BIOL 300 Wiki Documentation

Release 1.0

Jeffrey Gill

Jan 06, 2019

Instructor Manual

These instructions are written for Spring 2017.

CHAPTER 1

Creating Student Accounts and Teams

For the second day of class, you will need to assign students to teams. During the first two weeks of classes, before the add/drop deadline has passed, students may join or leave the course, and team assignments will need to be adjusted accordingly.

Additionally, if you need to assign grades to a student who does not yet have an account on the wiki, you will need to create one for them before you will be able to do so.

These instructions will help you accomplish all of these tasks using one script.

1.1 Installing the Student Account and Team Creation Script

The script needs to be installed only once. Skip to *Using the Student Account and Team Creation Script* if installation is complete.

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install this new package,

Package	Description
python-yaml	YAML parser and emitter for Python

using the following:

```
sudo apt-get install python-yaml
```

4. Download and install the script for creating student accounts and assigning students to teams:

```
sudo wget -O /usr/local/sbin/register-students-and-teams https://biol-300-wiki-
↪docs.readthedocs.io/en/latest/_downloads/register-students-and-teams
```

Set the MySQL password inside the script:

```
read -s -r -p "MySQL password: " DBPASS && sudo sed -i "/^sql_pass =/s|= .*|= '
↪$DBPASS'|" /usr/local/sbin/register-students-and-teams; DBPASS= ; echo
```

Protect the password:

```
sudo chown root:www-data /usr/local/sbin/register-students-and-teams
sudo chmod ug=rwx,o= /usr/local/sbin/register-students-and-teams
```

If you are curious about the contents of the script, you can view it here:

register-students-and-teams

`Direct link`

```
#!/usr/bin/python

# Place this script in /usr/local/sbin and make it executable (chmod +x).
#
# This script will ...
#
# NOTE: The input file must use spaces for indentation, not tabs.
#
# TODO: also dump a CSV file that can be used for checkoffs


from __future__ import print_function
import datetime      # for naming a file with a timestamp
import ldap          # for querying CWRU servers for real names and network IDs
import MySQLdb       # for modifying the MediaWiki and Django databases
import random        # for randomly assigning teams
import re            # for string matching using regular expressions
import sys, getopt   # for processing command line arguments
import yaml          # for reading/writing human-readable input/output files

# import Django modules
import os
sys.path += [os.path.abspath('/var/www/django')]
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'CourseDjango.settings')
import django
django.setup()
from CourseDjango.credit.models import Team
from django.contrib.auth.models import User

# database connection credentials
sql_user = 'root'
sql_pass = '<MySQL password>'
sql_wiki_db   = 'wikidb'
sql_django_db = 'djangodb'
```

(continues on next page)

(continued from previous page)

```
# defaults and globals
verbose = False
dry_run = False
output_file = 'output-' + datetime.datetime.now().strftime('%Y-%m-%d-%H%M%S') + '.
↪txt'
default_grading_group = 'All students' # TODO: should determine grading student␣
↪group (undergad vs. grad vs. all students) from file
TEAM_UNASSIGNED = 0
TEAM_INSTRUCTOR = -1



def usage():
    print('usage: sudo register-students-and-teams [-o output_file] input_file')
    print()
    print('       -h, --help      display this help and exit')
    print('       -o, --output    specify a name for output file')
    print('       -v, --verbose   print debugging messages')
    print('       -d, --dry-run   create an output file but make no database␣
↪changes')

def say(str = ''):
    global verbose
    if verbose:
        print(str)

def warn(msg):
    print('warning: ' + str(msg))



def main(argv = None):

    global verbose, dry_run, output_file

    try:

        ########################
        # PROCESS OPTS AND ARGS #
        ########################

        # parse command line options and arguments
        if argv is None:
            argv = sys.argv
        try:
            opts, args = getopt.gnu_getopt(argv[1:], 'ho:vd', ['help','output=',
↪'verbose','dry-run'])
        except getopt.error, msg:
            raise UsageError(msg)

        # process options
        for opt, arg in opts:
            if opt in ('-h', '--help'):
                usage()
                return 0  # exit code
            elif opt in ('-o', '--output'):
                output_file = arg
```

(continues on next page)

(continued from previous page)

```
        elif opt in ('-v', '--verbose'):
            verbose = True
        elif opt in ('-d', '--dry-run'):
            dry_run = True
        else:
           raise Error('unhandled option')

    # require that the user is root
    if os.geteuid() != 0:
        raise Error("superuser privileges needed (rerun with sudo)")

    if dry_run:
        say('dry run: will make no database changes')
    say('output file: ' + str(output_file))
    say()

    # process arguments
    if len(args) == 1:
        input_file = args[0]
    elif len(args) < 1:
        raise UsageError('missing argument input_file')
    elif len(args) > 1:
        raise UsageError('too many arguments')


    ###################
    # READ INPUT FILE #
    ###################

    try:
        with open(input_file, 'r') as file:
            input = validate_input(yaml.safe_load(file))
    except IOError, msg:
        raise Error('cannot open input file "' + str(input_file) + '"')
    except yaml.parser.ParserError, msg:
        raise ParsingError(msg)
    except yaml.scanner.ScannerError, msg:
        raise ParsingError(msg)

    say('Result of reading input:')
    say(input)
    say()


    ##########################################
    # PARSE INPUT AND LOOK UP PERSON DETAILS #
    ##########################################

    all_users = []
    teams_defined = []

    for block in input:

        block_title, team_num = validate_block_title(str(block.keys()[0]))
        block_items = block.values()[0]

        # make sure a team is not defined more than once
```

(continues on next page)

(continued from previous page)

```
            if team_num > 0:
                if team_num in teams_defined:
                    raise Error('team %d is defined more than once' % team_num)
                else:
                    teams_defined.append(team_num)

            for person in block_items:

                # if no person details were provided...
                if type(person) == str:
                    person_label = person
                    person_details_from_file = {}
                # if some person details were provided...
                elif type(person) == dict and len(person) == 1:
                    person_label = str(person.keys()[0])
                    person_details_from_file = person.values()[0]
                else:
                    raise Error('bad person specification slipped past validator')

                # look up the individual in the LDAP database
                person_details_from_ldap = lookup_names_in_ldap(person_label)

                # merge the person details from LDAP and those provided (provided␣
↪details supersede LDAP)
                merged_person_details = person_details_from_ldap.copy()
                merged_person_details.update(person_details_from_file)

                # add person_label and team_num
                merged_person_details.update({"io_label": person_label, "team_num
↪": team_num})

                # save the person details
                all_users.append(merged_person_details)
                say(merged_person_details)

            say()

        # test for duplicate users
        all_user_ids = [user['uid'] for user in all_users]
        duplicate_user_ids = list(set([uid for uid in all_user_ids if all_user_
↪ids.count(uid) > 1]))
        if duplicate_user_ids:
            raise Error('individuals with the following ids appear in the input␣
↪more than once (they may possibly have different labels, e.g., "hjc" and
↪"hillel.chiel@case.edu"): ' + str(duplicate_user_ids))


        #####################################
        # FIND TEAMS FOR UNMATCHED STUDENTS #
        #####################################

        users_needing_a_team = [user for user in all_users if user['team_num'] ==␣
↪TEAM_UNASSIGNED]
        total_teams_needed = len(teams_defined) + len(users_needing_a_team)/2
        say('There will need to be a total of ' + str(total_teams_needed) + '␣
↪teams')
        say()
```

(continues on next page)

(continued from previous page)

```
        say('Teams defined in input:')
        say(sorted(teams_defined))
        say()

        teams_with_large_nums = [team_num for team_num in teams_defined if team_
↪num > total_teams_needed]
        if teams_with_large_nums:
            warn('the following teams have numbers greater than necessary given␣
↪the size of enrollment (you can manually reduce them and then rerun this script,
↪ or ignore this warning): ' + str(teams_with_large_nums))
            say()

        teams_to_fill = list(set(range(1, total_teams_needed + 1)) - set(teams_
↪defined))[:total_teams_needed-len(teams_defined)]
        say('The teams to fill are:')
        say(teams_to_fill)
        say()

        if len(users_needing_a_team) == 1:
            raise Error('because there are no other unassigned students to pair␣
↪him or her with, "' + users_needing_a_team[0]['io_label'] + '" will need to be␣
↪assigned to a team manually; edit the input directly and then rerun this script
↪')
            pass
        elif len(users_needing_a_team) > 1:
            uids_needing_a_team = [user['uid'] for user in users_needing_a_team]
            random.shuffle(uids_needing_a_team)

            new_team_assignments = dict(zip(uids_needing_a_team, sorted(2*teams_
↪to_fill) + [max(teams_to_fill)])) # adding max(teams_to_fill) will make the␣
↪last team a team of three if there are an odd number of unassigned students
            for i, user in enumerate(all_users):
                if user['uid'] in new_team_assignments.keys():
                    user_copy = user.copy()
                    user_copy['team_num'] = new_team_assignments[user['uid']]
                    all_users[i] = user_copy
            say('The new team assignments will be:')
            say(new_team_assignments)
            say()


        ####################
        # DUMP OUTPUT FILE #
        ####################

        # export the team assignments in a form that can be reimported by this␣
↪script
        if output_file:
            dump_output_file(all_users)
            say('Formatted output written to file')
            say()


        ####################################
        # REGISTER ACCOUNTS WITH MEDIAWIKI #
        ####################################
```

(continues on next page)

(continued from previous page)

```
        print()
        print('*** MediaWiki registration ***')
        print()
        # add users to the MediaWiki database and update real names
        db  = MySQLdb.connect(host='localhost', user=sql_user, passwd=sql_pass,␣
↪db=sql_wiki_db)
        cur = db.cursor()
        try:
            # query for existing MediaWiki users
            cur.execute('SELECT user_name FROM user')
            uids_already_registered = [uid[0] for uid in cur.fetchall()]
            say('Existing MediaWiki users:')
            say(uids_already_registered)
            say()

            say('-- MEDIAWIKI REGISTRATION --')
            say()
            for user in all_users:
                print(user['uid'])
                uid = user['uid'].capitalize()  # MediaWiki user names are␣
↪capitalized
                real_name = user['first'] + ' ' + user['last']

                # registration and real name
                if uid not in uids_already_registered:
                    print('needs registered')
                    print('registering with real name "%s"...' % real_name)
                    if not dry_run:
                        cur.execute('INSERT INTO user (user_name, user_real_name,␣
↪user_password, user_newpassword, user_email, user_touched, user_registration)'
                                    'VALUES("%s", "%s", "", "", "%s@case.edu",␣
↪DATE_FORMAT(UTC_TIMESTAMP(), "%%Y%%m%%d%%H%%i%%s"), DATE_FORMAT(UTC_TIMESTAMP(),
↪ "%%Y%%m%%d%%H%%i%%s"))'
                                    % (uid, real_name, uid.lower()))
                else:
                    print('already registered')
                    cur.execute('SELECT user_real_name FROM user WHERE user_name␣
↪= "%s"' % uid)
                    print('current real name is       "%s"' % cur.fetchall()[0])
                    print('changing real name to      "%s"...' % real_name)
                    if not dry_run:
                        cur.execute('UPDATE user SET user_real_name="%s" WHERE␣
↪user_name="%s"' % (real_name, uid))

                if user['team_num'] == TEAM_INSTRUCTOR:
                    print('granting instructor privileges...')
                    if not dry_run:
                        cur.execute('INSERT IGNORE INTO user_groups (ug_user, ug_
↪group)'
                                    'VALUES('
                                      '(SELECT user_id FROM user WHERE user_name␣
↪= "%s"),'
                                      '"bureaucrat")' % uid)
                        cur.execute('INSERT IGNORE INTO user_groups (ug_user, ug_
↪group)'
                                    'VALUES('
```

(continues on next page)

(continued from previous page)

```
                                        '(SELECT user_id FROM user WHERE user_name␣
↪= "%s"),'
                                        '"sysop")' % uid)
                            cur.execute('INSERT IGNORE INTO user_groups (ug_user, ug_
↪group)'
                                      'VALUES('
                                        '(SELECT user_id FROM user WHERE user_name␣
↪= "%s"),'
                                        '"grader")' % uid)
                    #### TODO: should also remove instructors from grading system
                    #### in case they were accidentally marked as students once
                else:
                    print('adding student to grading system...')
                    if not dry_run:
                        cur.execute('INSERT IGNORE INTO scholasticgrading_
↪groupuser (sggu_group_id, sggu_user_id)'
                                      'VALUES('
                                        '(SELECT sgg_id FROM scholasticgrading_
↪group WHERE sgg_title = "%s"),'
                                        '(SELECT user_id FROM user WHERE user_name␣
↪= "%s"))' % (default_grading_group, uid))
                    #### TODO: should determine grading student group (undergad␣
↪vs. grad vs. all students) from file

                    #### TODO: should also remove any grader privileges from␣
↪students
                    #### in case they were accidentally marked as instructors once

                    #### TODO: should also remove unlisted students from grading␣
↪system
                print()
        except MySQLdb.Error, err:
            try:
                print('MySQL Error [%d]: %s' % (err.args[0], err.args[1]))
            except IndexError:
                print('MySQL Error: %s' % str(err))
        db.commit()


        #################################
        # REGISTER ACCOUNTS WITH DJANGO #
        #################################

        print()
        print()
        print('*** Django registration ***')
        print()
        for user in all_users:
            print(user['uid'])
            try:
                u = User.objects.get(username = user['uid'])
                print('already registered')
                print('current real name is       "%s %s"' % (u.first_name, u.
↪last_name))
                print('changing real name to      "%s %s"...' % (user['first'],␣
↪user['last']))
                u.first_name = user['first']
```

(continues on next page)

(continued from previous page)

```
                u.last_name  = user['last']
                u.is_active = True
                if not dry_run:
                    u.save()
            except User.DoesNotExist, err:
                print('needs registered')
                print('registering with real name "%s %s"...' % (user['first'],␣
↪user['last']))
                u = User(username = user['uid'], first_name = user['first'], last_
↪name = user['last'], is_active = True)
                if not dry_run:
                    u.save()
            if user['team_num'] == TEAM_INSTRUCTOR:
                print('granting instructor privileges...')
                u.is_staff = True
                u.is_superuser = True
                if not dry_run:
                    u.save()
            else:
                print('removing instructor privileges (if necessary)...')
                u.is_staff = False
                u.is_superuser = False
                if not dry_run:
                    u.save()
            print()

        print('destroying estisting teams')
        for team in Team.objects.all():
            print(team)
            if not dry_run:
                team.delete()
        print()
        print('creating new teams')
        team_num_list = set([user['team_num'] for user in all_users if user['team_
↪num'] > 0])
        for team_num in team_num_list:
            team = Team(number = team_num, active = True)
            if not dry_run:
                team.save()
                for member in [User.objects.get(username = user['uid']) for user␣
↪in all_users if user['team_num'] == team_num]:
                    team.members.add(member)
                team.save()
                print(team)
        print()


        #####################
        # DUMP STUDENT LIST #
        #####################

        print()
        print('*** paste the following into the "Student list" wiki page ***')
        print()
        print()
        print('{| class="wikitable sortable" style="text-align:center"')
        print('|-')
```

(continues on next page)

(continued from previous page)

```
        print('! Team !! Members')
        print('|-')
        print()
        for team_num in sorted(team_num_list):
            team_members = [user for user in all_users if user['team_num'] ==␣
↪team_num]
            print('| rowspan="%d"| %d' % (len(team_members), team_num))
            for user in sorted(team_members, key = lambda user: user['last']):
                print('| [[User:%s]]' % user['uid'].capitalize())
                print('|-')
            print()
        print('|}')
        print()


        return 0  # exit code


    except Error, err:
        print("error: " + str(err.msg), file = sys.stderr)
        return 1  # exit code

    except ParsingError, err:
        print("parser error: " + str(err.msg), file = sys.stderr)
        return 1  # exit code

    except UsageError, err:
        print("error: " + str(err.msg), file = sys.stderr)
        usage()
        return 1  # exit code



# validate the data structures of input
def validate_input(input):

    if type(input) != list or len(input) < 1:
        raise ParsingError('the following was expected to be parsed as a non-
↪empty list (did you forget a hyphen?): ' + str(input))

    for block in input:

        if type(block) != dict or len(block) != 1:
            raise ParsingError('the following was expected to be parsed as a␣
↪dictionary of length 1 (did you forget a colon?): ' + str(block))

        block_title = str(block.keys()[0])
        block_items = block.values()[0]

        if type(block_items) != list or len(block_items) < 1:
            raise ParsingError('the following under heading "' + block_title + '"␣
↪was expected to be parsed as a non-empty list (did you forget a hyphen?): ' +␣
↪str(block_items))

        for item in block_items:
            if type(item) == str:
                pass
```

(continues on next page)

(continued from previous page)

```
            elif type(item) == dict and len(item) == 1:
                item_name = str(item.keys()[0])
                item_properties = item.values()[0]
                if item_properties is None:
                    raise ParsingError('if no properties are provided for "' +␣
↪item_name + '", the trailing colon should be removed')
                elif type(item_properties) != dict:
                    raise ParsingError('the following properties provided for "'␣
↪+ item_name + '" were expected to be parsed as a dictionary (did you insert␣
↪unneeded hyphens in front of the property names?): ' + str(item_properties))

            else:
                raise ParsingError('the following item under heading "' + block_
↪title + '" was expected to be parsed as a string or dictionary of length 1: ' +␣
↪str(item))

    return input



# validate block titles
# team_num > 0 corresponds to a real team of students
# team_num == TEAM_UNASSIGNED indicates block 'Needs team'
# team_num == TEAM_INSTRUCTOR indicates block 'Instructors'
def validate_block_title(block_title):

    say('Entering block "%s"' % block_title)
    match = re.match(r'(Team) (\d+)|(Needs team)|(Instructors)', block_title)
    if not match:
        raise ParsingError('"%s" is an unrecognized block title; only "Team N"␣
↪(where N is a positive integer), "Needs team", and "Instructors" are permitted'
↪% block_title)
    elif match.group(1) == 'Team':
        team_num = int(match.group(2))
        if team_num < 1:
            raise Error('"%s" is not permitted; team numbers must be positive␣
↪integers' % block_title)
    elif match.group(3) == 'Needs team':
        team_num = TEAM_UNASSIGNED
    elif match.group(4) == 'Instructors':
        team_num = TEAM_INSTRUCTOR
    else:
        raise Error('unhandled block title')

    return block_title, team_num



# dumped files are designed to be reusable as input files
def dump_output_file(all_users):
    if output_file:

        formatted_output = []

        # ---------- TEAM N ----------
```

(continues on next page)

(continued from previous page)

```
        team_num_list = set([user['team_num'] for user in all_users if user['team_
↪num'] > 0])
        for team_num in sorted(team_num_list):

            # filter by team_num
            team_members = [user for user in all_users if user['team_num'] ==␣
↪team_num]

            # format using same label as original input and drop some items
            team_members = [{user['io_label']: {k:v for k,v in user.items() if k␣
↪not in ('io_label','team_num')}} for user in team_members]

            # sort by last name
            team_members = sorted(team_members, key = lambda user: user.
↪values()[0]['last'])

            formatted_output.append({'Team ' + str(team_num): team_members})

        # ---------- NEEDS TEAM ----------

        # filter by team_num
        users_needing_a_team = [user for user in all_users if user['team_num'] ==␣
↪TEAM_UNASSIGNED]

        if users_needing_a_team:
            raise Error('some users were not placed into a group who should have␣
↪been')

        # ---------- INSTRUCTORS ----------

        # filter by team_num
        instructors = [user for user in all_users if user['team_num'] == TEAM_
↪INSTRUCTOR]

        # format using same label as original input and drop some items
        instructors = [{user['io_label']: {k:v for k,v in user.items() if k not␣
↪in ('io_label','team_num')}} for user in instructors]

        # sort by uid
        instructors = sorted(instructors, key = lambda user: user.values()[0]['uid
↪'])

        if instructors:
            formatted_output.append({'Instructors': instructors})


        # write to file
        try:
            with open(output_file, 'w') as file:
                os.chmod(output_file, 0o666) # make sure the output is easily␣
↪editable even though the script was run as root
                file.write(yaml.dump(formatted_output, default_flow_style=False))
        except IOError, msg:
            raise Error('cannot open output file "' + str(output_file) + '"')
```

(continues on next page)

(continued from previous page)

```
def ldap_search(searchstr):
    """Use a search string to fetch a {uid,first,last} dict using LDAP"""

    # login to the LDAP server
    l = ldap.init('ldap.case.edu')
    l.simple_bind('anonymous','')

    # look up the user's name by user id
    res_id = l.search('ou=People,o=cwru.edu,o=isp',
            ldap.SCOPE_SUBTREE, searchstr)
    res_t, res_d = l.result(res_id, 1000)

    if len(res_d) > 0:
        result = {
                  'uid':   res_d[0][1]['uid'][0],
                  'first': res_d[0][1]['givenName'][0],
                  'last':  res_d[0][1]['sn'][0]
                 }
    else:
        result = None

    # log out of the server
    l.unbind_s()

    return result



def lookup_names_in_ldap(uid_or_email):
    """Translate the username or email to a {uid,first,last} dict using LDAP"""

    # the case ldap server seems to throttle complex searches, so try the
    # several possibilities one at a time.
    result = ldap_search("(uid={0})".format(uid_or_email))

    if not result:
        result = ldap_search("(mail={0})".format(uid_or_email))
    if not result:
        result = ldap_search("(mailAlternateAddress={0})".format(uid_or_email))
    if not result:
        result = ldap_search("(mailEquivalentAddress={0})".format(uid_or_email))
    if not result:
        result = ldap_search("(mail={0})".format(
            uid_or_email.replace('case','cwru')))
    if not result:
        result = ldap_search("(mailAlternateAddress={0})".format(
            uid_or_email.replace('case','cwru')))
    if not result:
        result = ldap_search("(mailEquivalentAddress={0})".format(
            uid_or_email.replace('case','cwru')))

    # if the individual was not found...
    if not result:
        raise Error('person "' + uid_or_email + '" not found')

    # if the individual was found...
    else:
```

(continues on next page)

(continued from previous page)

```
        return result



class Error(Exception):
    def __init__(self, msg):
        self.msg = msg

class ParsingError(Exception):
    def __init__(self, msg):
        self.msg = msg

class UsageError(Exception):
    def __init__(self, msg):
        self.msg = msg



if __name__ == "__main__":
    sys.exit(main())
```

1.2 Using the Student Account and Team Creation Script

To use the script, you must first create an input file (a text file with a simple syntax, see below) containing a list of the emails of students enrolled in the course and of the instructors. Students who have selected their teammates can be placed together, and all others will be assigned to teams randomly.

When executed, the script does the following:

- Reads the input file
- Looks up real name information for each email address if a name is not specified
- Creates wiki and Django accounts for students and instructors who do not already have one
- Assigns students to teams who do not already have one
- Grants privileges to instructors
- Writes an output file containing the name details and modified team configuration
- Prints to the screen wikitext that should be copied into the Student list wiki page

The output file has the same syntax as the input file, so it can be run through the script again with modifications to make changes to the roster and team assignments. You can even change someone's real name in the output file to a preferred alternative and rerun the script to change their name on the wiki.

To use the script for the first time this semester, first log into the virtual machine:

```
ssh hjc@biol300.case.edu
```

Check for any old input or output files in your home directory that were created by running the script last year:

```
ls ~
```

If any are present, delete them using `rm`.

Next, create a file to serve as the input to the script:

```
vim ~/input.txt
```

You should create something that looks like this:

```
- Needs team:
    - george.washington@case.edu
    - john.adams@case.edu
    - thomas.jefferson@case.edu
    - james.madison@case.edu
    - james.monroe@case.edu
    - john.quincy.adams@case.edu
    - andrew.jackson@case.edu

- Instructors:
    - hillel.chiel@case.edu
    - jeffrey.gill@case.edu
```

Note that the indentations must be spaces, not tabs. Pay attention to punctuation and white space, e.g., there needs to be a space after each hyphen. Emails may be of the form `first.last@case.edu` or `abc123@case.edu`.

If some students have requested to be partnered together, you may specify that now:

```
- Team 1:
    - john.adams@case.edu
    - john.quincy.adams@case.edu

- Needs team:
    - george.washington@case.edu
    - thomas.jefferson@case.edu
    - james.madison@case.edu
    - james.monroe@case.edu
    - andrew.jackson@case.edu

- Instructors:
    - hillel.chiel@case.edu
    - jeffrey.gill@case.edu
```

To run the script, execute the following:

```
sudo register-students-and-teams ~/input.txt
```

An output file called `output-XXX.txt` will be created, where `XXX` will be a timestamp, and changes to the database will be made. You may use the `--dry-run` flag with the script to create the output file without actually making changes to the MediaWiki or Django databases.

In addition to creating an output file, the script will print messages to the screen. The last set of messages include wikitext that you should copy into the Student list wiki page.

The output file will look something like this:

```
- Team 1:
  - john.adams@case.edu:
      first: John
      last: Adams
      uid: jxa
  - john.quincy.adams@case.edu:
      first: John
      last: Adams
```

(continues on next page)

(continued from previous page)

```
      uid: jqa
- Team 2:
  - andrew.jackson@case.edu:
      first: Andrew
      last: Jackson
      uid: axj
  - james.monroe@case.edu:
      first: James
      last: Monroe
      uid: jxm2
- Team 3:
  - james.madison@case.edu:
      first: James
      last: Madison
      uid: jxm
  - thomas.jefferson@case.edu:
      first: Thomas
      last: Jefferson
      uid: txj
  - george.washington@case.edu:
      first: George
      last: Washington
      uid: gxw
- Instructors:
  - hillel.chiel@case.edu:
      first: Hillel
      last: Chiel
      uid: hjc
  - jeffrey.gill@case.edu:
      first: Jeffrey
      last: Gill
      uid: jpg18
```

In this example, because there were an odd number of students, a team of three was created.

If you later need to make changes to the roster, **you should use the output file as the new input**. Make a copy of the file first, modify it, and then re-run the script.

Suppose John Quincy Adams tells you that he prefers to go by "Johnny". You just need to modify JQA's first name in the output file and re-run the script.

Suppose James Monroe drops the course. The script isn't clever enough to see that one of the members of the team of three needs to be moved to Team 2 to rebalance them. To resolve this, you should modify the output file by deleting James Monroe and moving one of the members of Team 3 into Team 2 manually before re-running the script.

If a large number of new students enrolls in the class after some teams already exist, you can randomly assign them to new teams by adding the `Needs team:` heading used in the original input file.

CHAPTER 2

Form Emails

Todo: Provide form emails for instructors (e.g., absences, tardies).

CHAPTER 3

Patrolling Student Comments

On each instructor's personal page (reached by clicking your name in the top-right corner), there are two links that lead to lists of student comments.

The first is "Comments on course materials".

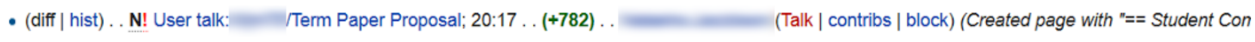
This page lists any content that students post on Talk pages in the Main namespace. For example, if a student were to click the "Discussion" link at the top of the Course Policies page, she could leave comments about the course policies. This is more relevant to BIOL 373, where we encouraged students to post concerns or corrections on the unit Talk pages. For BIOL 300, students generally shouldn't be doing this. However, this link is still worth checking from time to time, since students may incorrectly post things in this space.

The other link found on instructors' personal pages is "Comments on term papers".

This page lists any content that students post on Talk pages in the User namespace. That is, when a student leaves a comment on a benchmark, it will appear here.

When you are grading comments, you will want to mark them as "patrolled", which will eliminate them from these lists. There are two ways to mark comments as patrolled; depending on whether the comment was the first left on the page or not, you must use one or the other (which is fairly annoying and inconvenient).

If the comment left by a student resulted in the creation of a new Talk page, i.e., the student was the first individual to leave a comment, then the entry in the listing will look like this:



• (diff | hist) . . **N!** User talk:[redacted]/Term Paper Proposal; 20:17 . . **(+782)** . . [redacted] (Talk | contribs | block) *(Created page with "== Student Con*

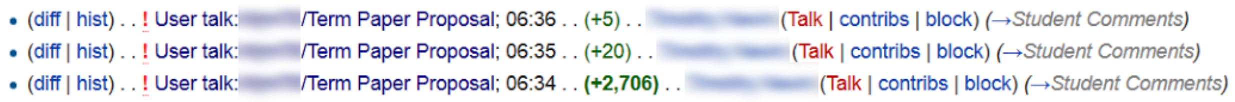
Things to take note of:

- Be careful about confusing the author of the comment and the recipient of the comment.
- The edit description says "Created page…", indicating that this student was the first to leave a comment on this page.
- There is a capital, bold N in front of the page name, which again indicates that this edit represents the creation of a new page.
- The word "diff" at the start of the line is not clickable. This is because there is no earlier edit to compare this first edit to.

Marking first edits as patrolled works differently than marking subsequent edits as patrolled. To mark a first edit as patrolled, navigate to the Talk page by clicking the page name (in this example, “User talk:. . . /Term Paper Proposal”), scroll to the very bottom of the page, and click the “Mark this page as patrolled” link. Doing this will eliminate the first edit entry from the listing.

Note that when you click on the page name in this way, you will be taken to the *current* version of the page, so you will see any late edits the student made, as well as edits made by other students. To disambiguate which edits the first-edit poster left before or after a deadline, look at the page history.

Subsequent edits made by students to existing User Talk pages appear a little differently:

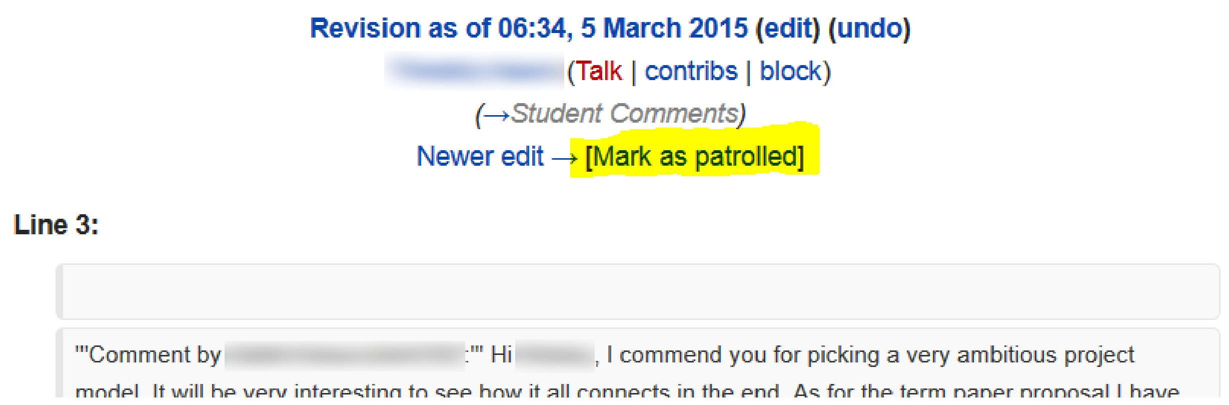


- (diff | hist) . . ! User talk:[redacted]/Term Paper Proposal; 06:36 . . (+5) . . [redacted] (Talk | contribs | block) (→Student Comments)
- (diff | hist) . . ! User talk:[redacted]/Term Paper Proposal; 06:35 . . (+20) . . [redacted] (Talk | contribs | block) (→Student Comments)
- (diff | hist) . . ! User talk:[redacted]/Term Paper Proposal; 06:34 . . **(+2,706)** . . [redacted] (Talk | contribs | block) (→Student Comments)

Things to take note of:

- In this example, the same person performed three consecutive edits on one term paper proposal page. This student could have left more later. Each edit will need to be marked as patrolled separately. Once you’ve decided on a score for this student’s comment, you should scan the list of edits for additional edits by the same person to the same benchmark and mark each as patrolled so that you do not have to deal with them again.
- The bold N is absent, since the page already existed.
- The “diff” link is clickable because the page already existed.

To mark these edits as patrolled, click a “diff” link and then click “Mark as patrolled” near the top right, beneath the editor’s name:



Revision as of 06:34, 5 March 2015 (edit) (undo)
[redacted] (Talk | contribs | block)
(→Student Comments)
Newer edit → [Mark as patrolled]

Line 3:

'''Comment by [redacted]:''' Hi [redacted], I commend you for picking a very ambitious project model. It will be very interesting to see how it all connects in the end. As for the term paper proposal I have

Clicking this will remove that single entry from the edits listing.

Rinse and repeat. By the end of this round of comment grading, the list of unpatrolled edits should be empty.

CHAPTER 4

Granting Student Privileges

Todo: Add instructions for giving students privileges to access the private areas of the wiki (relevant to BIOL 373 only) and for getting a term paper deadline extension.

CHAPTER 5

Modifying Student Real Names

5.1 Changing Real Names on the Wiki

Todo: Install phpMyAdmin for a web interface for changing real names.

To edit user real names manually so they will appear with the Realnames extension, edit the wiki database directly using the following command:

```
mysql -u root -p wikidb
```

Enter the <MySQL password> when prompted. Edit entries for individuals on the wiki using

```
UPDATE user SET user_real_name='<realname>' WHERE user_name='<username>';
```

Note that user names always begin with a capital letter. Type `exit` to quit. After changing a real name, the server must be restarted:

```
sudo apache2ctl restart
```

5.2 Changing Real Names in the Survey System

Todo: Add instructions for changing real names in Django.

CHAPTER 6

Introduction to Vim

Vim is a text editor for the command line that you will use often when following these instructions. Vim can be very intimidating for new users, but for the purposes of these instructions you only need to know a few commands.

To start Vim, type `vim /path/to/file` on the command line to edit a new or existing file. Vim has two modes that you will use regularly: command mode and insert mode. **Command mode** allows you to execute commands like saving and quitting. **Insert mode** lets you actually edit text.

When you first start Vim you will be in command mode, so you cannot immediately begin editing text. You must first switch into insert mode by pressing `i`. When you do this, you should notice the last line in the terminal window says "`-- INSERT --`". When in insert mode, you can type text and it will appear on screen as you would expect. Press Esc to return to command mode, and the insert mode indicator at the bottom of the terminal window will disappear.

If you press the wrong keys in command mode, unexpected and confusing things may happen. If you get stuck in some other mode, press Esc a few times to return to command mode. If that fails, try typing `:q` followed by the Enter key.

Below is a list of useful commands. Notice that commands beginning with a colon (`:`) or slash (`/`) appear on the bottom line in the terminal window when typed in command mode.

- **Save and quit**: In command mode, type `:w` followed by the Enter key to save (write) the file. Type `:q` followed by the Enter key to quit Vim. If you have made unsaved changes to the file before trying to quit, you will get an error, "`No write since last change`". To quit and discard unsaved changes, type `:q!` followed by the Enter key.
- **Undo and redo**: In command mode, press `u` to undo your last action. Press Ctrl + `r` to redo.
- **Search**: In command mode, type `/`, followed by a word or phrase, followed by the Enter key to begin searching for the phrase. Press `n` to cycle to the next instance of the phrase or `N` to cycle to the previous instance.
- **Line numbers**: In command mode, type `:set number` followed by the Enter key to turn on line numbering. Use `:set nonumber` to turn it off again.
- **Jump**: In command mode, press `gg` to jump to the top of the file. Type a line number before this command to jump to that line. Press `G` to jump to the end of the file.
- **Paste**: You can use your terminal application's normal paste command to paste text while in insert mode. However, sometimes when you paste a multi-line block of text into Vim from another source (such as from this

document), the pasted content will be auto-indented or auto-commented in undesirable ways. To prevent this behavior, in command mode, type `:set paste` followed by the Enter key before entering insert mode and pasting.

- **Mouse**: In command mode, type `:set mouse=a` followed by the Enter key to enable interaction with the text using your mouse. This will allow you to click anywhere to place the cursor or to select blocks of text.

6.1 Vim Configuration File

The following instructions will customize your Vim configuration so that syntax highlighting and line numbers are turned on by default. It will also enable interaction with text using the mouse, so you can place the cursor by clicking with the mouse. It also enables a "persistent undo" feature that allows Vim to retain the history of edits for a file after it is closed, so you can undo or redo edits even after closing and reopening a file.

The persistent undo feature depends on the existence of a directory. Create this directory now:

```
mkdir -p ~/.vim/undodir
```

Finally, download and install the `.vimrc` configuration file:

```
wget -O ~/.vimrc https://biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/.vimrc
```

If you are curious about the contents of `.vimrc`, you can view it here:

.vimrc

`Direct link`

```
" An example for a vimrc file.
"
" Maintainer:   Bram Moolenaar <Bram@vim.org>
" Last change:  2002 Sep 19
"
" To use it, copy it to
"     for Unix and OS/2:  ~/.vimrc
"             for Amiga:  s:.vimrc
"  for MS-DOS and Win32:  $VIM\_vimrc
"           for OpenVMS:  sys$login:.vimrc

" When started as "evim", evim.vim will already have done these settings.
"if v:progname =~? "evim"
"  finish
"endif

" Use Vim settings, rather then Vi settings (much better!).
" This must be first, because it changes other options as a side effect.
set nocompatible

" allow backspacing over everything in insert mode
set backspace=indent,eol,start

if has("vms")
  set nobackup          " do not keep a backup file, use versions instead
else
  set backup            " keep a backup file
endif
set history=50          " keep 50 lines of command line history
```

(continues on next page)

(continued from previous page)

```
set ruler               " show the cursor position all the time
set showcmd             " display incomplete commands
set incsearch           " do incremental searching

" For Win32 GUI: remove 't' flag from 'guioptions': no tearoff menu entries
" let &guioptions = substitute(&guioptions, "t", "", "g")

" Don't use Ex mode, use Q for formatting
map Q gq

" This is an alternative that also works in block mode, but the deleted
" text is lost and it only works for putting the current register.
"vnoremap p "_dp

" Switch syntax highlighting on, when the terminal has colors
" Also switch on highlighting the last used search pattern.
if &t_Co > 2 || has("gui_running")
  syntax on
  set hlsearch
endif

if has("gui_running")
  " turn off the toolbar
  set guioptions-=T
  set columns=84
endif

" Only do this part when compiled with support for autocommands.
if has("autocmd")

  " Enable file type detection.
  " Use the default filetype settings, so that mail gets 'tw' set to 72,
  " 'cindent' is on in C files, etc.
  " Also load indent files, to automatically do language-dependent indenting.
  filetype plugin indent on

  " Put these in an autocmd group, so that we can delete them easily.
  augroup vimrcEx
  au!

  " For all text files set 'textwidth' to 78 characters.
  autocmd FileType text setlocal textwidth=78

  " When editing a file, always jump to the last known cursor position.
  " Don't do it when the position is invalid or when inside an event handler
  " (happens when dropping a file on gvim).
  autocmd BufReadPost *
    \ if line("'\"") > 0 && line("'\"") <= line("$") |
    \   exe "normal g`\"" |
    \ endif

  augroup END

else

  set autoindent                " always set autoindenting on
```

(continues on next page)

(continued from previous page)

```
endif " has("autocmd")
set tabstop=4
set shiftwidth=4
color desert
set guifont=Dejavu\ Sans\ Mono\ 12
hi Normal guifg=Green guibg=Black
hi Constant guifg=Yellow
hi Statement guifg=Yellow gui=none
hi Type guifg=Yellow gui=none
hi Special guifg=SeaGreen
hi Search guibg=DarkBlue

"set lines=46
set expandtab
"setlocal spell spelllang=en_us
set number
set mouse=a

function! s:insert_gates()
  let gatename = substitute(toupper(expand("%:t")), "\\.", "_", "g")
  execute "normal! i#ifndef " . gatename
  execute "normal! o#define " . gatename . " "
  execute "normal! Go#endif /* " . gatename . " */"
  normal! kk
endfunction
autocmd BufNewFile *.{h,hpp} call <SID>insert_gates()

"Spell checking
"set spell spelllang=en_us

" persistent undo support
" " note: you need to mkdir ~/.vim/undodir before this will work
set undofile " turn on persistent undo's
set undodir=~/.vim/undodir " where to store the undo files
set undolevels=1000  " max number of changes to undo
set undoreload=10000 " max number of lines to save for undo

" allow switching to a different buffer without saving
set hidden
```

CHAPTER 7

Introduction to Git

Todo: Write an intro to Git.

24. Create a Git profile for yourself:

```
git config --global user.name "<your full real name>"
git config --global user.email <your email>
git config --global core.editor vim
git config --global push.default simple
```

24. Generate an SSH key on the virtual machine if you do not have one already. This will generate an identity for your account on this virtual machine that can later be granted privileges to the Git repository on GitHub. If you have an SSH key already, this command will recognize that and do nothing. Otherwise, press Enter three times when asked about the file location and passphrase:

```
[ -e ~/.ssh/id_rsa ] || ssh-keygen
```

CHAPTER 8

Port Forwarding & Bridged Networking

DynamicsHJC is a computer (an actual computer, not a virtual machine) in the Chiel lab. VirtualBox and all our virtual machines are running on it.

DynamicsHJC is plugged into the wired network via ethernet. Associated with every computer's ethernet port is a unique MAC address, which is basically a number that it uses to identify itself to the network.

When a new computer is first plugged into the wired network on campus, it can't use the internet until you complete a registration wizard that pops up in the browser. This registration process is triggered by the fact that the MAC address is not recognized by Case. Registration links a computer's name (hostname) to its MAC address.

In our case, the hostname is "dynamicshjc". I've set up a web server on that computer, and that's where these wiki setup instructions are hosted (notice that `dynamicshjc.case.edu` is in the address). When you visit these instructions in a web browser, the university network routes you to our lab computer by doing a lookup of the hostname in its registration database to determine which ethernet port to connect you with.

Since our virtual machines are running on DynamicsHJC, they all share a single ethernet port. How can the network properly route web traffic to the right machine if more than one lives at the same address? There are two mutually exclusive ways of solving this problem with VirtualBox: port forwarding and bridged networking.

Port forwarding assigns to a virtual machine a unique port number (analogous to having multiple apartments at one street address). To connect to a virtual machine using port forwarding, you'd type something like this into your web browser: https://dynamicshjc.case.edu:8014. More precisely, individual ports on the host (host ports) are mapped to individual ports on the virtual machine (guest ports). In this example, the host port 8014 on DynamicsHJC is mapped to the guest port 443 (the standard HTTPS port) on the virtual machine. We use this method throughout these instructions while set up is in progress to avoid conflicts with the existing wiki site.

With **bridged networking**, the virtual machine is assigned a randomly generated, faux MAC address, which the computer running VirtualBox presents to the network as if it's actually a different ethernet port. When this is done, the network can't tell the difference between a virtual machine using bridged networking and a real computer using its own unique ethernet port. Using the network registration process outlined *here*, you can associate a hostname with the virtual machine's phony MAC address (this only ever needs to be done once), and thus it can be accessed like a real machine at an address like `neurowiki.case.edu`. This is the permanent solution that we switch to when the virtual machine is all set up.

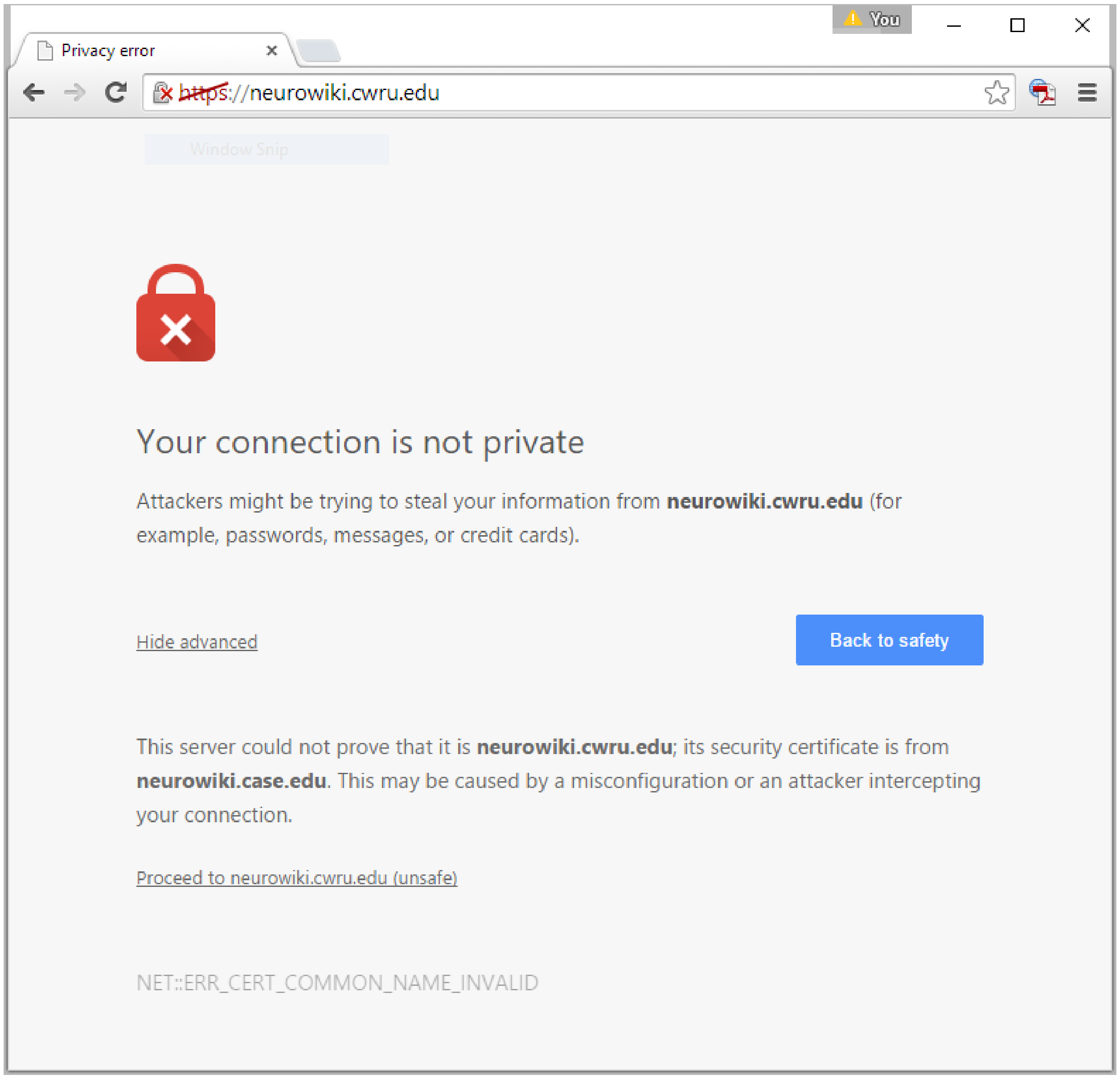
CHAPTER 9

HTTPS & SSL

Web browsers can make insecure (HTTP) or secure (HTTPS) connections to websites. Since the wiki requires passwords, it should be set up to use HTTPS.

When a web browser tries to create an HTTPS connection with a website, it requests to see an SSL certificate that proves the server it is talking to is the real deal. If the server can't provide one that matches the address the web browser is trying to connect to, the web browser will warn the user that something is wrong.

For example, although https://neurowiki.cwru.edu is a perfectly valid alias for https://neurowiki.case.edu, a browser visiting the former should warn the user that the connection is not secure, preventing them from visiting it unless they click through some warnings. Here's what Chrome does when I visit that address:


You
Privacy error
~~https~~://neurowiki.cwru.edu
Window Snip

Your connection is not private

Attackers might be trying to steal your information from **neurowiki.cwru.edu** (for example, passwords, messages, or credit cards).

Hide advanced
Back to safety

This server could not prove that it is **neurowiki.cwru.edu**; its security certificate is from **neurowiki.case.edu**. This may be caused by a misconfiguration or an attacker intercepting your connection.

Proceed to neurowiki.cwru.edu (unsafe)

NET::ERR_CERT_COMMON_NAME_INVALID

The SSL certificate we obtained from the university for that virtual machine certifies that our site is `neurowiki.case.edu`, but not `neurowiki.cwru.edu`. Lacking an appropriate SSL certificate does not prevent people from using the site (i.e., in Chrome, I could click "Proceed to neurowiki.cwru.edu (unsafe)"), but your visitors will likely be scared away, or at least inconvenienced, if it's absent.

As described in *Port Forwarding & Bridged Networking*, when a virtual machine is using port forwarding, it is accessed using an address like https://dynamicshjc.case.edu:8014. For this reason, the virtual machine needs to present DynamicsHJC's SSL certificate while it is using port forwarding. When it is switched to bridged networking, it is accessed at an address like https://neurowiki.case.edu, so it must present a different SSL certificate. When the virtual machine is cloned and moved to https://neurowikidev.case.edu, it must present another different SSL certificate.

Instructions for creating and renewing SSL certificates can be found in *SSL Certificate Management*.

CHAPTER 10

Building from Scratch

Follow these instructions to build the wiki from scratch:

10.1 Install Operating System

1. Log into the VirtualBox Machines (vbox) account on DynamicsHJC in front of the keyboard and monitor, or using interactive screen sharing (see *Screen Sharing Using SSVNC* for details).
2. On DynamicsHJC, download the installation disk image (ISO file) for the latest version of Ubuntu Server with long term support (LTS).

   LTS versions of Ubuntu Server receive guaranteed support (e.g., security updates) for 5 years. As of December 2016, the latest LTS version is Ubuntu Server 16.04.1 (Xenial Xerus) 64-bit, which will be supported until April 2021. The next LTS release should be available in April 2018.[1]

   You can download the Ubuntu Server ISO file here.
3. In VirtualBox[2], press "New" to create a new virtual machine, and choose the following settings:
   - **Name:** biol300_2017
   - **Type:** Linux
   - **Version:** Ubuntu (64 bit)
   - **Memory size:** 2048 MB
   - **Hard drive:** Create a virtual hard drive now
   - **Hard disk file type:** VDI (VirtualBox Disk Image)
   - **Storage on physical hard drive:** Dynamically allocated
   - **File location:** biol300_2017
   - **File size:** 30.00 GB

[1] You can visit this page to see the release schedule for LTS versions of Ubuntu.
[2] As of this writing, we are using VirtualBox version 5.0.30 r112061.

Warning: Do not immediately start the virtual machine! If you do, you should delete it and start over, since this may result in an improperly configured network interface on the virtual machine.[3]

4. After creating the virtual machine, select it and choose Machine > Group. Click on the new group's name ("New group"), click Group > Rename Group, and rename the group to "2017". Finally, drag-and-drop the new group into the "BIOL 300" group.

5. Set up port forwarding for the virtual machine (see *Port Forwarding & Bridged Networking* for a more detailed explanation). This will allow you to access the new virtual machine at a temporary URL as you build it, and the wiki from the previous year can remain accessible at https://biol300.case.edu until this setup is complete.

   In VirtualBox, select the virtual machine and choose Settings > Network. With the "Adapter 1" tab selected, change the "Attached to" setting to "NAT".

   Under Advanced, click Port Forwarding. Here you can create rules that bind a port on the host machine (DynamicsHJC) to a port on the guest machine (biol300_2017). You need to create two rules (click the "+" button)[4]:

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
HTTPS	TCP	[leave blank]	8014	[leave blank]	443
SSH	TCP	[leave blank]	8015	[leave blank]	22

6. At the end of this instruction set, you will *Activate Bridged Networking* (again see *Port Forwarding & Bridged Networking* for background). To make that process as easy as possible, you should set the MAC address for the virtual machine now.

   Follow the instructions in *Looking Up MAC Addresses* to identify the MAC address for the BIOL 300 Wiki. In VirtualBox, open Settings > Network again, and under Advanced replace the randomly generated MAC address with the BIOL 300 Wiki's.

   Until bridge networking is activated, the MAC address will not actually do anything other than ensure that the network interface on the virtual machine is properly configured when the operating system is installed.[3]

7. Start the new virtual machine. You will immediately be asked to provide a start-up disk. Click the file browser icon (small folder) and select the Ubuntu Server disk image (ISO file) that you downloaded in step 2. Finally, press Start.

8. The virtual machine will boot from the start-up disk. After choosing English as the installer language, choose "Install Ubuntu Server" and select the following settings:

   - **Select a language**
     - English
   - **Select your location**
     - United States
   - **Configure the keyboard**
     - Detect keyboard layout: No
     - Country of origin for the keyboard: English (US)

[3] In earlier versions of Ubuntu, I encountered this issue where a virtual machine initialized with the wrong MAC address required some extra work to correct the mistake when it came time to activate bridged networking. I think in more recent versions of Ubuntu the MAC address might be detected at boot time, which means ensuring it is set properly now is not actually necessary, but I haven't tested this extensively.

[4] Although the guest ports must have exactly these values (443 and 22 are the default ports for HTTPS and SSH, respectively), the host ports were, in fact, chosen arbitrarily. You can choose any numbers you like for the host ports (up to 65536), but they must not conflict with standard ports that DynamicsHJC needs for its own services (ports 0-1024 are reserved for common applications, so you should not use values within that range) or any other virtual machine that is running using port forwarding. To access the virtual machine from off campus, you should also consider using host ports that have univeristy firewall exceptions (see *University Firewall* for details). For the purposes of this document, I will assume you are using the host ports listed above.

  - Keyboard layout: English (US)
- **Configure the network**
  - Hostname: biol300
- **Set up users and passwords**
  - Full name: Hillel Chiel
  - Username: hjc
  - Password: <system password>
  - Encrypt your home directory: No
- **Configure the clock**
  - Timezone: America/New York
- **Partition disks**
  - Partitioning method: Guided - use entire disk and set up LVM
  - Select disk to partition: [there should only be one choice]
  - Write the changes to disks and configure LVM: Yes
  - Amount of volume group to use for guided partitioning: 100%
  - Write the changes to disks: Yes
- **Configure the package manager**
  - HTTP proxy information: [leave blank]
- **Configuring tasksel**
  - How do you want to manage upgrades on this system: Install security updates automatically
- **Software selection**
  - **Choose software to install [use Space to select and Enter to finish]:**
    * LAMP server
    * Mail server
    * standard system utilities
    * OpenSSH server
  - When prompted, provide a <MySQL password>
  - **Postfix Configuration**
    * General type of mail configuration: Internet Site
    * System mail name: biol300.case.edu
- **Install the GRUB boot loader on a hard disk**
  - Install the GRUB boot loader to the master boot record: Yes

After the installation completes, the virtual machine will restart.

9. The virtual machine should immediately be accessible via SSH from the internet. Connecting to the virtual machine remotely via SSH, rather than at DynamicsHJC's physical keyboard or using screen sharing will allow you to conveniently enter the remaining commands found in these instructions by copying and pasting them into the Terminal window on your own machine.

To setup SSH connectivity from your personal computer, as well as complete some additional account configuration, visit *Creating Virtual Machine Accounts* and follow the instructions starting with step 2.

10. If you are not already logged in, do so now:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

11. Create an account for Jeff (username: `gill`) and any other TAs you would like to have sudo privileges on the server by completing the first step in *Creating Virtual Machine Accounts*. Instruct them to complete the remaining steps in that document to finish configuring their accounts. Account creation and configuration can also be done at a later time.

12. Upgrade all packages:

```
sudo apt-get update
sudo apt-get dist-upgrade
```

13. Install these new packages,

Package	Description
ntp	Network Time Protocol daemon for automatic system time sync
virtualbox-guest-utils	VirtualBox guest utilities for communicating with VM host for time sync and folder sharing

using the following:

```
sudo apt-get install ntp virtualbox-guest-utils
```

14. Shut down the virtual machine:

```
sudo shutdown -h now
```

15. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**OS installed**".

10.2 Configure Web Server with HTTPS

1. If the virtual machine is running, log in and shut it down:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
sudo shutdown -h now
```

2. Share the SSL certificate directory owned by the VirtualBox Machines (vbox) account on DynamicsHJC (`~vbox/ssl-certificates`) with the virtual machine as a shared folder. Shared folders are accessible on the virtual machine under the mount point `/media`. In VirtualBox, select Machine > Settings > Shared Folders, press the "+" button, and use the following settings:

   - Folder Path: `/Users/vbox/ssl-certificates`
   - Folder Name: ssl-certificates
   - Check "Read-only"
   - Check "Auto-mount"

   Press "OK" twice to close the Settings windows.

3. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

4. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

5. The Apache web server operates as user www-data in group www-data. Give the web server ownership of and access to the web directory:

```
sudo chown -R www-data:www-data /var/www/
sudo chmod -R ug+rw /var/www/
sudo find /var/www -type d -exec chmod g+s {} \;
```

6. Download and install the `check-ssl-cert-expiration` script:

```
sudo wget -O /usr/local/sbin/check-ssl-cert-expiration https://biol-300-wiki-docs.
↪readthedocs.io/en/latest/_downloads/check-ssl-cert-expiration
sudo chmod +x /usr/local/sbin/check-ssl-cert-expiration
```

The script looks for the shared folder set up in step 2 and prints the expiration dates of any certificates found there. Check that this is working and that the certificates are current:

```
sudo check-ssl-cert-expiration
```

If you are curious about the contents of `check-ssl-cert-expiration`, you can view it here:

check-ssl-cert-expiration

`Direct link`

```
#!/bin/bash

# Place this script in /usr/local/sbin and make it executable (chmod +x).
#
# This script will print out the expiration dates for all SSL certificates
# located in CERTDIR or its subdirectories.


# Function for aborting with an error message

die () {
    echo >&2 "$@"
    exit 1
}


# Require that the user is root.

[ "$UID" -eq 0 ] || die "Aborted: superuser privileges needed (rerun with sudo)"


# The certificate files are expected to be found in CERTDIR with a specific
# naming scheme.

```

(continues on next page)

(continued from previous page)

```
CERTDIR="/media/sf_ssl-certificates"


# Find and report expiration dates for certificates.

find $CERTDIR -name "*_cert.cer" -print -exec bash -c "openssl x509 -noout -
↪enddate -in {} | sed -e 's/\(.*\)=\(.*\)/  \2/'" \;

exit 0
```

7. Disable some default Apache configuration files, and download and install a custom Apache configuration file for handling SSL certificates:

```
sudo a2dissite 000-default default-ssl
sudo wget -O /etc/apache2/sites-available/smart-ssl.conf https://biol-300-wiki-
↪docs.readthedocs.io/en/latest/_downloads/smart-ssl.conf
sudo a2enmod rewrite ssl
sudo a2ensite smart-ssl
sudo apache2ctl restart
```

The determination of which SSL certificate to use is done automatically by looking at the URL used to access the site. If port forwarding is enabled and the virtual machine is accessed using https://dynamicshjc.case.edu:8014, the certificate for DynamicsHJC will be selected automatically. If bridged networking is enabled and the virtual machine is accessed using https://biol300.case.edu, the certificate for the BIOL 300 Wiki will be selected automatically. Later, when the virtual machine is cloned and converted to BIOL300Dev, its certificate will be selected automatically.

If you are curious about the contents of `smart-ssl.conf`, you can view it here:

smart-ssl.conf

`Direct link`

```
#
#
# GLOBAL SETTINGS
#
#


# Globally specify ServerName to satisfy requirement, will be replaced by
# matching virtual host's ServerName

ServerName localhost


# Except where aliases are used, all URLs are relative to DocumentRoot, e.g.,
# https://example.com/dir1/page.html points to /var/www/html/dir1/page.html

DocumentRoot /var/www/html


# Deny access to everything on the server unless overridden by other Directory
# directives, and allow access to the DocumentRoot

<Directory ~ "/">
    Options -Indexes
```

(continues on next page)

(continued from previous page)

```
    Require all denied
</Directory>

<Directory ~ "/var/www/html">
    Require all granted
</Directory>


# Disallow access to .git directories and .gitignore files

RedirectMatch 404 /\.git


# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined


#   SSL Engine Options:
#   Set various options for the SSL engine.
#   o FakeBasicAuth:
#         Translate the client X.509 into a Basic Authorisation.  This means that
#         the standard Auth/DBMAuth methods can be used for access control.  The
#         user name is the `one line' version of the client's X.509 certificate.
#         Note that no password is obtained from the user. Every entry in the user
#         file needs this password: `xxj31ZMTZzkVA'.
#   o ExportCertData:
#         This exports two additional environment variables: SSL_CLIENT_CERT and
#         SSL_SERVER_CERT. These contain the PEM-encoded certificates of the
#         server (always existing) and the client (only existing when client
#         authentication is used). This can be used to import the certificates
#         into CGI scripts.
#   o StdEnvVars:
#         This exports the standard SSL/TLS related `SSL_*' environment variables.
#         Per default this exportation is switched off for performance reasons,
#         because the extraction step is an expensive operation and is usually
#         useless for serving static content. So one usually enables the
#         exportation for CGI and SSI requests only.
#   o OptRenegotiate:
#         This enables optimized SSL connection renegotiation handling when SSL
#         directives are used in per-directory context.
#SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
<FilesMatch "\.(cgi|shtml|phtml|php)$">
        SSLOptions +StdEnvVars
</FilesMatch>
<Directory /usr/lib/cgi-bin>
        SSLOptions +StdEnvVars
</Directory>


#
#
```

(continues on next page)

(continued from previous page)

```
# VIRTUAL HOSTS
#
# ServerName specifies what hostname must appear in the request's Host: header
# to match a virtual host
#
#


# Matches any http://* and redirects to https://*

<VirtualHost *:80>
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>


<IfModule mod_ssl.c>


# Matches only https://dynamicshjc.case.edu

<VirtualHost *:443>
    ServerName dynamicshjc.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/dynamicshjc/dynamicshjc_
↪case_edu_cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/dynamicshjc/dynamicshjc_
↪case_edu.key
    SSLCertificateChainFile /media/sf_ssl-certificates/dynamicshjc/dynamicshjc_
↪case_edu_interm.cer
</VirtualHost>


# Matches only https://neurowiki.case.edu

<VirtualHost *:443>
    ServerName neurowiki.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/neurowiki/neurowiki_case_
↪edu_cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/neurowiki/neurowiki_case_
↪edu.key
    SSLCertificateChainFile /media/sf_ssl-certificates/neurowiki/neurowiki_case_
↪edu_interm.cer
</VirtualHost>


# Matches only https://neurowikidev.case.edu

<VirtualHost *:443>
    ServerName neurowikidev.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/neurowikidev/neurowikidev_
↪case_edu_cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/neurowikidev/neurowikidev_
↪case_edu.key
    SSLCertificateChainFile /media/sf_ssl-certificates/neurowikidev/neurowikidev_
↪case_edu_interm.cer
```

(continues on next page)

(continued from previous page)

```
</VirtualHost>


# Matches only https://biol300.case.edu

<VirtualHost *:443>
    ServerName biol300.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/biol300/biol300_case_edu_
↪cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/biol300/biol300_case_edu.
↪key
    SSLCertificateChainFile /media/sf_ssl-certificates/biol300/biol300_case_edu_
↪interm.cer
</VirtualHost>


# Matches only https://biol300dev.case.edu

<VirtualHost *:443>
    ServerName biol300dev.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/biol300dev/biol300dev_case_
↪edu_cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/biol300dev/biol300dev_case_
↪edu.key
    SSLCertificateChainFile /media/sf_ssl-certificates/biol300dev/biol300dev_case_
↪edu_interm.cer
</VirtualHost>


# Matches only https://slugwiki.case.edu

<VirtualHost *:443>
    ServerName slugwiki.case.edu
    SSLEngine on
    SSLCertificateFile      /media/sf_ssl-certificates/slugwiki/slugwiki_case_edu_
↪cert.cer
    SSLCertificateKeyFile   /media/sf_ssl-certificates/slugwiki/slugwiki_case_edu.
↪key
    SSLCertificateChainFile /media/sf_ssl-certificates/slugwiki/slugwiki_case_edu_
↪interm.cer
</VirtualHost>


</IfModule>


# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

8. The web server should now be active. Open a web browser and navigate to

   https://dynamicshjc.case.edu:8014

   You should see a default page provided by Apache.

9. Delete that default page:

```
rm /var/www/html/index.html
```

10. Discourage bots, such as Google's web crawler, from visiting some parts of the site. Download and install `robots.txt`:

```
wget -O /var/www/html/robots.txt https://biol-300-wiki-docs.readthedocs.io/en/
↪latest/_downloads/robots.txt
```

If you are curious about the contents of `robots.txt`, you can view it here:

robots.txt

`Direct link`

```
User-agent: *
Allow: /w/load.php?
Disallow: /w/
Disallow: /django
Disallow: /JSNeuroSim
```

11. Shut down the virtual machine:

```
sudo shutdown -h now
```

12. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Web server configured with HTTPS**".

10.3 Install MediaWiki

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install these new packages,

Package	Description
php	server-side, HTML-embedded scripting language
php-pear	PEAR - PHP Extension and Application Repository
php-mbstring	MBSTRING module for PHP
php-intl	Internationalisation module for PHP
php-dev	Files for PHP module development (needed for `phpize` for compiling apcu)
imagemagick	image manipulation programs
python-mysqldb	Python interface to MySQL
python-ldap	LDAP interface module for Python
Mail	Class that provides multiple interfaces for sending emails
php-apcu	APC (Alternative PHP Cache) User Cache for PHP[1]

using the following (when asked about enabling internal debugging, just press Enter):

```
sudo apt-get install php php-pear php-mbstring php-intl php-dev imagemagick␣
↪python-mysqldb python-ldap
sudo pear install --alldeps Mail
sudo pecl install apcu_bc-beta
sudo bash -c "echo extension=apcu.so > /etc/php/7.0/mods-available/apcu.ini"
sudo bash -c "echo extension=apc.so > /etc/php/7.0/mods-available/z_apc.ini"
sudo phpenmod apcu z_apc
```

4. Download the source code for the latest version of MediaWiki with long term support (LTS).

   LTS versions of MediaWiki receive guaranteed support (e.g., security updates) for 3 years, although these updates must be applied manually (see *Updating MediaWiki*). As of December 2016, the latest LTS version is MediaWiki 1.27.1, which will be supported until June 2019. The next LTS release should be available in mid-2018.[2]

   Run these commands to download and install the source code:

```
wget -P ~ https://releases.wikimedia.org/mediawiki/1.27/mediawiki-1.27.1.tar.gz
tar -xvzf ~/mediawiki-*.tar.gz -C /var/www
mkdir -p /var/www/mediawiki
mv /var/www/mediawiki-*/* /var/www/mediawiki
rm -rf /var/www/mediawiki-* ~/mediawiki-*.tar.gz
sudo chown -R www-data:www-data /var/www/mediawiki
sudo chmod -R ug+rw /var/www/mediawiki
```

5. Download and install a custom Apache configuration file for MediaWiki:

```
sudo wget -O /etc/apache2/conf-available/mediawiki.conf https://biol-300-wiki-
↪docs.readthedocs.io/en/latest/_downloads/mediawiki.conf
sudo a2enconf mediawiki
sudo apache2ctl restart
```

   If you are curious about the contents of `mediawiki.conf`, you can view it here:

   mediawiki.conf

   `Direct link`

```
#
#
# MEDIAWIKI CONFIGURATION
#
#


# Aliases allow short URLs for the wiki
```

(continues on next page)

[1] PHP version 7.0 is packaged with this version of Ubuntu; in earlier versions of Ubuntu, PHP 5 was used (fun fact: PHP 6 never existed). The PHP module APC provides the wiki with object caching, which should improve its performance. This module is compatible with PHP 5.4 and earlier, but not PHP 5.5, 5.6, or 7.0. For these newer versions of PHP, a replacement module called APCu exists. For the purposes of object caching, it can do the job, but it uses different function calls than the MediaWiki source code expects (e.g., `apcu_fetch()` rather than `apc_fetch()`). The versions of APCu that work with PHP 5.5 and 5.6 include backwards compatibility code that handles this. However, the version of APCu that works with PHP 7.0, which is what you get if you use `sudo apt-get install php-apcu`, dropped the backwards compatibility. Instead, a forked version of APCu needs to be used which restores backwards compatibility. To install that, I followed the instructions found here, which have been incorporated into the instructions above.

This was so much fun to figure out. If you could see my face right now, you would know that I am totally serious and not at all being sarcastic.

[2] You can visit this page to see the release schedule for versions of MediaWiki. The MediaWiki download page provides the URL for downloading the tar archive.

(continued from previous page)

```
Alias /wiki /var/www/mediawiki/index.php
Alias /w    /var/www/mediawiki


# Redirect to the wiki when a URL lacks a path, e.g.,
# https://example.com redirects to https://example.com/wiki

RedirectMatch ^/$ /wiki


# Allow large form submissions (for grading) and large file transfers
# (for wiki uploads)

php_admin_value max_input_vars 50000
php_admin_value memory_limit 256M
php_admin_value post_max_size 105M
php_admin_value upload_max_filesize 100M


# Allow wiki files to be served

<Directory ~ "/var/www/mediawiki">
    Require all granted
</Directory>


# Prevent scripts uploaded to the wiki from being executed

<Directory ~ "/var/www/mediawiki/images">
    # Serve HTML as plaintext, don't execute SHTML
    AddType text/plain html htm shtml php phtml php5

    # Don't run arbitrary PHP code
    php_admin_flag engine off
</Directory>


# Forbid access to deleted files

<Directory ~ "/var/www/mediawiki/images/deleted">
    Require all denied
</Directory>


# Forbid access to the cache

<Directory ~ "/var/www/mediawiki/cache">
    Require all denied
</Directory>


# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

6. Create and initializing the wiki database (do not forget to fill in the passwords):

```
php /var/www/mediawiki/maintenance/install.php --dbname wikidb --dbuser root --
↪dbpass <MySQL password> --pass <wiki admin password> "BIOL 300 Wiki" Hjc
```

7. Download and install the MediaWiki configuration file:

```
wget -O /var/www/mediawiki/LocalSettings.php https://biol-300-wiki-docs.
↪readthedocs.io/en/latest/_downloads/LocalSettings.php
```

Set the passwords and randomize the secret keys inside the configuration file:

```
read -s -r -p "MySQL password: " DBPASS && sed -i '/^\$wgDBpassword/s|".*";|"'
↪$DBPASS'";|' /var/www/mediawiki/LocalSettings.php; DBPASS= ; echo

read -s -r -p "Gmail password: " GMPASS && sed -i '/^    .password./s|".*"|"'
↪$GMPASS'"|' /var/www/mediawiki/LocalSettings.php; GMPASS= ; echo

sed -i '/^\$wgSecretKey/s|".*";|"'$(openssl rand -hex 32)'";|' /var/www/mediawiki/
↪LocalSettings.php

sed -i '/^\$wgUpgradeKey/s|".*";|"'$(openssl rand -hex 8)'";|' /var/www/mediawiki/
↪LocalSettings.php
```

Protect the passwords in the configuration file:

```
sudo chown www-data:www-data /var/www/mediawiki/LocalSettings.php*
sudo chmod ug=rw,o= /var/www/mediawiki/LocalSettings.php*
```

If you are curious about the contents of `LocalSettings.php`, you can view it here:

LocalSettings.php

`Direct link`

```
<?php
# This file was automatically generated by the MediaWiki 1.27.0
# installer. If you make manual changes, please keep track in case you
# need to recreate them later.
#
# See includes/DefaultSettings.php for all configurable settings
# and their default values, but don't forget to make changes in _this_
# file, not there.
#
# Further documentation for configuration settings may be found at:
# https://www.mediawiki.org/wiki/Manual:Configuration_settings

# Protect against web entry
if ( !defined( 'MEDIAWIKI' ) ) {
        exit;
}

## Uncomment this to disable output compression
# $wgDisableOutputCompression = true;

$wgSitename = "BIOL 300 Wiki";
$wgMetaNamespace = "BIOL_300_Wiki";

## The URL base path to the directory containing the wiki;
## defaults for all runtime URL paths are based off of this.
## For more information on customizing the URLs
## (like /w/index.php/Page_title to /wiki/Page_title) please see:
## https://www.mediawiki.org/wiki/Manual:Short_URL
$wgScriptPath = "/w";
```

(continues on next page)

(continued from previous page)

```
## The protocol and server name to use in fully-qualified URLs
## Note: Leaving $wgServer undefined will allow it to be determined
## automatically, which is necessary when switching from port forwarding to
## bridged networking or when cloning the dev server
#$wgServer = "https://dynamicshjc.case.edu:8014";

## The URL path to static resources (images, scripts, etc.)
$wgResourceBasePath = $wgScriptPath;

## The URL path to the logo.  Make sure you change this from the default,
## or else you'll overwrite your logo when you upgrade!
$wgLogo = "$wgScriptPath/images/a/a0/Wiki-logo.png";

## UPO means: this is also a user preference option

$wgEnableEmail = true;
$wgEnableUserEmail = true; # UPO

$wgEmergencyContact = "biol300.case@gmail.com";
$wgPasswordSender = "biol300.case@gmail.com";

$wgEnotifUserTalk = true; # UPO
$wgEnotifWatchlist = true; # UPO
$wgEmailAuthentication = true;

## Database settings
$wgDBtype = "mysql";
$wgDBserver = "localhost";
$wgDBname = "wikidb";
$wgDBuser = "root";
$wgDBpassword = "<MySQL password>";

# MySQL specific settings
$wgDBprefix = "";

# MySQL table options to use during installation or update
$wgDBTableOptions = "ENGINE=InnoDB, DEFAULT CHARSET=binary";

# Experimental charset support for MySQL 5.0.
$wgDBmysql5 = false;

## Shared memory settings
$wgMainCacheType = CACHE_ACCEL;
$wgMemCachedServers = [];

## To enable image uploads, make sure the 'images' directory
## is writable, then set this to true:
$wgEnableUploads = true;
$wgUseImageMagick = true;
$wgImageMagickConvertCommand = "/usr/bin/convert";

# InstantCommons allows wiki to use images from https://commons.wikimedia.org
$wgUseInstantCommons = true;

## If you use ImageMagick (or any other shell command) on a
## Linux server, this will need to be set to the name of an
```

(continues on next page)

(continued from previous page)

```
## available UTF-8 locale
$wgShellLocale = "en_US.utf8";

## Set $wgCacheDirectory to a writable directory on the web server
## to make your wiki go slightly faster. The directory should not
## be publically accessible from the web.
$wgCacheDirectory = "$IP/cache";

# Site language code, should be one of the list in ./languages/data/Names.php
$wgLanguageCode = "en";

$wgSecretKey = "<random string>";

# Changing this will log out all existing sessions.
$wgAuthenticationTokenVersion = "1";

# Site upgrade key. Must be set to a string (default provided) to turn on the
# web installer while LocalSettings.php is in place
$wgUpgradeKey = "<random string>";

## For attaching licensing metadata to pages, and displaying an
## appropriate copyright notice / icon. GNU Free Documentation
## License and Creative Commons licenses are supported so far.
$wgRightsPage = ""; # Set to the title of a wiki page that describes your license/
↪copyright
$wgRightsUrl = "https://creativecommons.org/licenses/by-nc-sa/4.0/";
$wgRightsText = "Creative Commons Attribution-NonCommercial-ShareAlike";
$wgRightsIcon = "$wgResourceBasePath/resources/assets/licenses/cc-by-nc-sa.png";

# Path to the GNU diff3 utility. Used for conflict resolution.
$wgDiff3 = "/usr/bin/diff3";

# The following permissions were set based on your choice in the installer
$wgGroupPermissions['*']['createaccount'] = false;
$wgGroupPermissions['*']['edit'] = false;

## Default skin: you can change the default skin. Use the internal symbolic
## names, ie 'vector', 'monobook':
$wgDefaultSkin = "vector";

# Enabled skins.
# The following skins were automatically enabled:
wfLoadSkin( 'Vector' );


# Enabled extensions. Most of the extensions are enabled by adding
# wfLoadExtensions('ExtensionName');
# to LocalSettings.php. Check specific extension documentation for more details.
# The following extensions were automatically enabled:
wfLoadExtension( 'Cite' );
wfLoadExtension( 'Gadgets' );
wfLoadExtension( 'Interwiki' );
wfLoadExtension( 'Nuke' );
wfLoadExtension( 'ParserFunctions' );
wfLoadExtension( 'Renameuser' );
wfLoadExtension( 'SyntaxHighlight_GeSHi' );
wfLoadExtension( 'WikiEditor' );
```

(continues on next page)

(continued from previous page)

```
# End of automatically generated settings.
# Add more configuration options below.


# Disable one form of caching so that newly uploaded files are immediately
# visible on pages
$wgParserCacheType = CACHE_NONE;

# Set default time zone for display
$wgDefaultUserOptions['timecorrection'] = 'ZoneInfo|-240|America/New_York';

# Enable short URLs
$wgUsePathInfo = true;
$wgArticlePath = "/wiki/$1";

# Favicon
$wgFavicon = "$wgScriptPath/images/7/77/Wiki-favicon.ico";

# Fix site customization on Special:Preferences page
$wgAllowSiteCSSOnRestrictedPages = true;

# Configure email sent from the wiki
$wgPasswordSenderName = $wgSitename;
$wgSMTP = array(
    'host'      => "smtp.gmail.com",
    'IDHost'    => "gmail.com",
    'port'      => 587,
    'auth'      => true,
    'username'  => "biol300.case",
    'password'  => "<gmail password>"
);

# Allow uploads of additional file types
$wgFileExtensions[] = "nb";
$wgFileExtensions[] = "cdf";
$wgFileExtensions[] = "m";
$wgFileExtensions[] = "wl";
$wgFileExtensions[] = "pdf";
$wgFileExtensions[] = "ico";  # for uploading the wiki favicon

# MediaWiki is suspicious of Import and Export commands in Mathematica
# notebooks, e.g., Chapter02.nb from BIOL 300. Disable script checks before
# uploading notebooks, then re-enable them for security purposes. Toggle between
# states by running
#   sudo disable-upload-script-checks
# which will change the value of this variable.
$wgDisableUploadScriptChecks = false;

# Some Mathematica notebooks with file sizes smaller than the PHP memory limit
# will still cause PHP to run out of memory when the notebooks are uploaded
# (perhaps data compressed in the notebook is being expanded?). A blank white
# page will appear when uploading the file. Enable PHP error reporting for
# debugging.
#error_reporting(E_ALL);
#ini_set('display_errors', 1);
```

(continues on next page)

(continued from previous page)

```
# When all else fails, enable debugging details to track down issues with
# extensions, etc.  Useful when you get the MWException error.
#$wgShowExceptionDetails = true;
```

8. Create a script for toggling a security variable by downloading and installing a file:

```
sudo wget -O /usr/local/sbin/disable-upload-script-checks https://biol-300-wiki-
↪docs.readthedocs.io/en/latest/_downloads/disable-upload-script-checks
sudo chmod +x /usr/local/sbin/disable-upload-script-checks
```

If you are curious about the contents of the script, you can view it here:

disable-upload-script-checks

`Direct link`

```
#!/bin/bash

# Place this script in /usr/local/sbin and make it executable (chmod +x).
#
# This script will toggle the MediaWiki variable $wgDisableUploadScriptChecks.
# This variable should be set to false to improve security, but occasionally it
# must temporarily be set to true so that certain files (e.g., Chapter02.nb from
# BIOL 300) can be uploaded, since MediaWiki mistakenly thinks it contains
# malicious code.


# Function for aborting with an error message

die () {
    echo >&2 "$@"
    exit 1
}


# Require that the user is root.

[ "$UID" -eq 0 ] || die "Aborted: superuser privileges needed (rerun with sudo)"


# Get the current state of the $wgDisableUploadScriptChecks variable

STATE=$(grep ^\$wgDisableUploadScriptChecks /var/www/mediawiki/LocalSettings.php␣
↪| awk '{ print $3 }')


# Determine the new state

if [ "$STATE" == "false;" ]; then
    NEWSTATE="true;"
    echo "Security checks on wiki file uploads are now DISABLED. RE-ENABLE WHEN␣
↪YOU'RE DONE!"
else
    NEWSTATE="false;"
    echo "Security checks on wiki file uploads are now ENABLED."
fi
```

(continues on next page)

(continued from previous page)

```


# Set the new state.

sed -i '/^\$wgDisableUploadScriptChecks/s|= \(.*\);|= '$NEWSTATE'|' /var/www/
↪mediawiki/LocalSettings.php

exit 0
```

9. Allow MediaWiki to recognize Mathematica notebooks and package files so that they can be uploaded:

```
echo "text nb cdf m wl" >> /var/www/mediawiki/includes/mime.types
```

10. Download and install a script for fetching real names for MediaWiki users. Since the CASAuth extension, which will be installed later, automatically fetches real names, this script should not need to be run regularly.

```
sudo wget -O /usr/local/sbin/set-real-names-in-mediawiki https://biol-300-wiki-
↪docs.readthedocs.io/en/latest/_downloads/set-real-names-in-mediawiki
```

Set the MySQL password inside the script:

```
read -s -r -p "MySQL password: " DBPASS && sudo sed -i "/^sql_pass =/s|= .*|= '
↪$DBPASS'|" /usr/local/sbin/set-real-names-in-mediawiki; DBPASS= ; echo
```

Protect the password:

```
sudo chown root:www-data /usr/local/sbin/set-real-names-in-mediawiki
sudo chmod ug=rwx,o= /usr/local/sbin/set-real-names-in-mediawiki
```

Run the script to fetch your real name for your account:

```
sudo set-real-names-in-mediawiki
sudo apache2ctl restart
```

If you are curious about the contents of the script, you can view it here:

set-real-names-in-mediawiki

`Direct link`

```
#!/usr/bin/python

# Place this script in /usr/local/sbin and make it executable (chmod +x).
#
# This script will fetch real names for any MediaWiki user lacking a real name.


from __future__ import print_function
import MySQLdb
import ldap

def ldap_search(searchstr):
    """Use a search string to fetch a uid and displayName using LDAP"""

    # login to the LDAP server
    l = ldap.init('ldap.case.edu')
    l.simple_bind('anonymous','')
```

(continues on next page)

(continued from previous page)

```
    # look up the user's name by user id
    res_id = l.search('ou=People,o=cwru.edu,o=isp',
            ldap.SCOPE_SUBTREE, searchstr)
    res_t, res_d = l.result(res_id, 1000)

    if len(res_d) > 0:
        result = [res_d[0][1]['uid'][0],
                  res_d[0][1]['displayName'][0]]
                  #res_d[0][1]['cn'][0],
                  #res_d[0][1]['givenName'][0],
                  #res_d[0][1]['sn'][0]]
    else:
        result = []

    # log out of the server
    l.unbind_s()

    return result

# database connection credentials
sql_user = 'root'
sql_pass = '<MySQL password>'
sql_db   = 'wikidb'

# connect to the wiki database
db  = MySQLdb.connect(host='localhost', user=sql_user, passwd=sql_pass, db=sql_db)
cur = db.cursor()

# query for ids of users lacking a real name
cur.execute('SELECT user_name FROM user WHERE (user_real_name = "" OR␣
↪LOWER(CONVERT(user_real_name USING latin1)) = LOWER(CONVERT(user_name USING␣
↪latin1)))')
ids = [row[0] for row in cur.fetchall()]

if len(ids) == 0:
    print("Aborting: All wiki users already have real names.")
    exit(0)

# print the user ids
print("Wiki users without a real name:")
print(' '.join(ids) + '\n')

# query ldap for user real names
results = []
for id in ids:
    result = ldap_search("(uid={0})".format(id))
    if len(result) > 0:
        results.append(result)
    else:
        print('%s does not appear in the LDAP database!' % (id))

if len(results) == 0:
    print("Aborting: All wiki users without real names cannot be identified.")
    exit(0)

# construct a list of queries for updating user real names
```

(continues on next page)

(continued from previous page)

```
sql_queries = [];
for user in results:
    sql_queries.append('UPDATE user SET user_real_name="%s" WHERE user_name="%s"'
↪% (user[1], user[0].title()))

# print the queries
print('\n'.join(sql_queries) + '\n')

# prompt for confirmation
raw_input('Press [Enter] to execute these SQL commands.')

# execute each query
for query in sql_queries:
    try:
        cur.execute(query)
    except MySQLdb.Error, e:
        try:
            print('MySQL Error [%d]: %s' % (e.args[0], e.args[1]))
        except IndexError:
            print('MySQL Error: %s' % str(e))
db.commit()
print('Done!')

# close the database connection
cur.close()
db.close()
```

11. The wiki should now be accessible. Open a web browser and navigate to

    https://dynamicshjc.case.edu:8014/wiki

    You should see a default page provided by MediaWiki. The wiki logo and favicon should be missing for now.

12. Shut down the virtual machine:

```
sudo shutdown -h now
```

13. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**MediaWiki installed**".

10.4 Install Django

Django is used for conducting class surveys.

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install these new packages,

Package	Description
python-pip	Alternative Python package installer
python-mysqldb	Python interface to MySQL
python-ldap	LDAP interface module for Python
python-imaging	Python Imaging Library
python-docutils	Text processing system for reStructuredText
libapache2-mod-wsgi	Python WSGI adapter module for Apache

using the following:

```
sudo apt-get install python-pip python-mysqldb python-ldap python-imaging python-
↪docutils libapache2-mod-wsgi
sudo pip install --upgrade pip
```

4. Install the latest version of Django with long term support (LTS)[1].

   LTS versions of Django receive guaranteed support (e.g., security updates) for at least 3 years, although these updates must be applied manually (see *Updating Django*). As of December 2016, the latest LTS version is Django 1.8.17, which will be supported until at least April 2018. The next LTS release should be available in April 2017.

   Run this command to install Django:

```
sudo pip install Django==1.8.17
```

5. Install django-cas-ng to allow integration between the Django site and the CWRU login servers:

```
sudo pip install django-cas-ng==3.5.6
```

6. Download our code for implementing our Django site—called CourseDjango—to the virtual machine (you will be asked for your password for DynamicsHJC):

```
git clone ssh://dynamicshjc.case.edu/~kms15/gitshare/CourseDjango.git /var/www/
↪django
```

7. Store the database password and a random "secret key" (which you do not need to write down) in a file accessible to Django:

```
read -s -r -p "MySQL password: " DBPASS && echo "db_password = '$DBPASS'" > /var/
↪www/django/CourseDjango/passwords.py; DBPASS= ; echo

echo "secret_key  = '$(openssl rand -hex 32)'" >> /var/www/django/CourseDjango/
↪passwords.py
```

   Protect the file:

```
sudo chmod ug=rw,o= /var/www/django/CourseDjango/passwords.py*
```

8. Export the survey forms from last semester's Django database. Surveys for both BIOL 373 and BIOL 300 are contained in a single database. You should export from last semester's BIOL 373 server because modifications to any surveys need to be preserved for future offerings of both courses.

   Run the following commands to export the survey forms (you will be asked to accept the unrecognized fingerprint for the BIOL 373 server – this is expected – and you will need to enter your password for the BIOL 373 server three times):

[1] You can visit this page to see the release schedule for LTS versions of Django.

```
ssh hjc@neurowiki.case.edu "/var/www/django/manage.py dumpsurveyforms > ~/
↪surveyforms.json"
scp hjc@neurowiki.case.edu:surveyforms.json ~/
ssh hjc@neurowiki.case.edu "rm ~/surveyforms.json"
```

9. Create the Django database, import the survey forms, and delete the exported survey forms file since it is no longer needed and contains the answers to quiz questions (enter the <MySQL password> when prompted):

```
echo "CREATE DATABASE djangodb;" | mysql -u root -p
/var/www/django/manage.py migrate
/var/www/django/manage.py loaddata ~/surveyforms.json
rm ~/surveyforms.json
```

10. Place static assets (CSS, JS, image files) used by the Django site (student-facing and admin) in a directory that can be served by Apache:

```
mkdir -p /var/www/html/django/static
/var/www/django/manage.py collectstatic --noinput --clear
```

11. Give the web server ownership of and access to the new files:

```
sudo chown -R www-data:www-data /var/www/
sudo chmod -R ug+rw /var/www/
```

12. Download and install a custom Apache configuration file for Django:

```
sudo wget -O /etc/apache2/conf-available/django.conf https://biol-300-wiki-docs.
↪readthedocs.io/en/latest/_downloads/django.conf
sudo a2enconf django
sudo apache2ctl restart
```

If you are curious about the contents of `django.conf`, you can view it here:

django.conf

`Direct link`

```
#
#
# DJANGO CONFIGURATION
#
#


# Run Django in daemon mode and add the CourseDjango project to the Python path

WSGIDaemonProcess django-daemon python-path=/var/www/django
WSGIProcessGroup django-daemon


# Allow access to the Django site

WSGIScriptAlias /django /var/www/django/CourseDjango/wsgi.py

<Directory "/var/www/django/CourseDjango">
    <Files wsgi.py>
        Require all granted
    </Files>
```

(continues on next page)

(continued from previous page)

```
</Directory>


# Bypass the WSGIScriptAlias for static items (CSS, JS, images)

Alias /django/static /var/www/html/django/static


# Alias needed for django-cas to work

WSGIScriptAlias /django/accounts /var/www/django/CourseDjango/wsgi.py


# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

13. Log into the Django site to automatically create an account for yourself. Visit

    https://dynamicshjc.case.edu:8014/django

14. Edit the Django database to make yourself an administrator. Access the database:

    ```
    mysql -u root -p djangodb
    ```

    Enter the <MySQL password> when prompted. Execute these SQL commands:

    ```
    UPDATE auth_user SET is_superuser=1 WHERE username='hjc';
    UPDATE auth_user SET is_staff=1 WHERE username='hjc';
    ```

    Type `exit` to quit.

15. The Django administration tools are now accessible at

    https://dynamicshjc.case.edu:8014/django/admin

    You should promote Jeff and any other TAs to superuser and staff status using this interface after they log into the Django site for the first time, which will create their accounts.

16. Shut down the virtual machine:

    ```
    sudo shutdown -h now
    ```

17. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Django installed**".

10.5 Configure Automatic Backup

1. Start the virtual machine and log in:

   ```
   ssh -p 8015 hjc@dynamicshjc.case.edu
   ```

2. Check for and install system updates on the virtual machine:

   ```
   sudo apt-get update
   sudo apt-get dist-upgrade
   sudo apt-get autoremove
   ```

3. Download and install the backup script:

```
sudo wget -O /usr/local/sbin/backup-wiki https://biol-300-wiki-docs.readthedocs.
↪io/en/latest/_downloads/backup-wiki
```

Set the MySQL password inside the script:

```
read -s -r -p "MySQL password: " DBPASS && sudo sed -i '/^SQLPASS=/s|=.*|='$DBPASS
↪'|' /usr/local/sbin/backup-wiki; DBPASS= ; echo
```

Protect the password:

```
sudo chown root:www-data /usr/local/sbin/backup-wiki
sudo chmod ug=rwx,o= /usr/local/sbin/backup-wiki
```

If you are curious about the contents of the script, you can view it here:

backup-wiki

`Direct link`

```
#!/bin/bash

# Copyright (c) 2016, Kendrick Shaw, Jeffrey Gill, Hillel Chiel
# All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions are met:
#
#   * Redistributions of source code must retain the above copyright notice,
#     this list of conditions and the following disclaimer.
#   * Redistributions in binary form must reproduce the above copyright notice,
#     this list of conditions and the following disclaimer in the documentation
#     and/or other materials provided with the distribution.
#
# THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
# AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
# IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
# ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
# LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
# CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
# SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
# INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
# CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
# ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
# POSSIBILITY OF SUCH DAMAGE.


DEFAULTNAME=$(hostname)-backup-$(date +'%Y%m%dT%H%M%S%z') # ISO-8601 timestamp

SQLUSER=root
SQLPASS=<MySQL password>
WIKIDB=wikidb
DJANGODB=djangodb

WWWDIR=/var/www
WIKIDIR=$WWWDIR/mediawiki
DJANGODIR=$WWWDIR/django
```

(continues on next page)

(continued from previous page)

```
function usage {
    echo "Usage:"
    echo "  backup-wiki backup  [-q] [backup_filename]"
    echo "  backup-wiki restore [-q]  backup_filename"
    echo
    echo "Options:"
    echo "  -q  Quiet mode, print only errors"
    exit
}


# read in options
QUIET=false
while getopts ":q" opt; do
    case $opt in
        q)
            # the flag -q was set
            QUIET=true
            ;;
        \?)
            # an unrecognized flag was set
            echo "Invalid option: -$OPTARG" >&2
            usage
            ;;
    esac
done
shift $((OPTIND-1))


# read in backup_file argument
if [ -z "$2" ]; then
    # second argument not provided -- use default file name
    BACKUPNAME=$DEFAULTNAME
else
    BACKUPNAME=$(dirname $2)/$(basename $2 .tar.bz2)
fi


##### BACKUP MODE #####

if [ "backup" == "$1" ]; then

    if [ $QUIET = false ]; then
        echo "Starting backup to archive $BACKUPNAME.tar.bz2 ..."
    fi

    # create a temporary directory
    if [ $QUIET = false ]; then
        echo "Creating temporary directory ..."
    fi
    TEMPDIR=$(mktemp -d)

    # copy the MediaWiki, Django, and JSNeuroSim source code
    if [ $QUIET = false ]; then
        echo "Copying files ..."
    fi
    cp -R $WWWDIR $TEMPDIR
```

(continues on next page)

(continued from previous page)

```
    # dump the wiki database in sql format
    if [ $QUIET = false ]; then
        echo "Exporting the wiki database in SQL format ..."
    fi
    mysqldump --user=$SQLUSER --password=$SQLPASS $WIKIDB --complete-insert --
↪result-file $TEMPDIR/wikidb.sql 2>&1 | grep -v "\[Warning\] Using a password"

    # dump the django database in sql format
    if [ $QUIET = false ]; then
        echo "Exporting the Django database in SQL format ..."
    fi
    mysqldump --user=$SQLUSER --password=$SQLPASS $DJANGODB --complete-insert --
↪result-file $TEMPDIR/djangodb.sql 2>&1 | grep -v "\[Warning\] Using a password"

    # dump the django database in json format (a more portable backup of the␣
↪content)
    if [ $QUIET = false ]; then
        echo "Exporting the Django database in JSON format ..."
    fi
    python $DJANGODIR/manage.py dumpdata > $TEMPDIR/djangodb.json

    # compress everything into a single file
    if [ $QUIET = false ]; then
        echo "Compressing directory into an archive file ..."
    fi
    mkdir -p $(dirname $BACKUPNAME)
    tar -cjf $BACKUPNAME.tar.bz2 -C $TEMPDIR .
    chmod o= $BACKUPNAME.tar.bz2

    # delete the temporary directory
    if [ $QUIET = false ]; then
        echo "Deleting temporary directory ..."
    fi
    rm -rf $TEMPDIR

    if [ $QUIET = false ]; then
        echo "Done!"
        echo
        echo "NOTE: The backup you just created contains sensitive student␣
↪information, quiz"
        echo "answers, and passwords. Keep this file in a safe place!"
        echo
    fi


##### RESTORE MODE #####

elif [ "restore" == "$1" ]; then

    if [ -z "$2" ]; then

        echo "Missing argument: backup_filename" >&2
        usage

    elif [ -e "$BACKUPNAME.tar.bz2" ]; then
```

(continues on next page)

(continued from previous page)

```
        if [ "$(whoami)" != "root" ]; then

            echo "Aborting: superuser privileges needed" >&2
            usage

        else

            if [ $QUIET = false ]; then
                echo "Starting restoration from archive $BACKUPNAME.tar.bz2 ..."
            fi

            # extract the files
            if [ $QUIET = false ]; then
                echo "Unpacking archive ..."
            fi
            TEMPDIR=$(mktemp -d)
            tar -xjf $BACKUPNAME.tar.bz2 -C $TEMPDIR

            # stop the server
            if [ $QUIET = false ]; then
                echo "Stopping the web server ..."
            fi
            apache2ctl stop

            # copy the files, fixing permissions and owners
            if [ $QUIET = false ]; then
                echo "Copying files ..."
            fi
            rm -rf $WWWDIR
            cp -R $TEMPDIR/$(basename $WWWDIR) $WWWDIR
            chown -R www-data:www-data $WWWDIR
            chmod -R ug+rw $WWWDIR
            find $WWWDIR -type d -exec chmod g+s {} \;

            # restore the wiki database
            if [ $QUIET = false ]; then
                echo "Restoring the wiki database ..."
            fi
            echo "DROP DATABASE IF EXISTS $WIKIDB;" | mysql --user=$SQLUSER --
↪password=$SQLPASS 2>&1 | grep -v "\[Warning\] Using a password"
            echo "CREATE DATABASE $WIKIDB;" | mysql --user=$SQLUSER --password=
↪$SQLPASS 2>&1 | grep -v "\[Warning\] Using a password"
            mysql --user=$SQLUSER --password=$SQLPASS $WIKIDB < $TEMPDIR/wikidb.
↪sql 2>&1 | grep -v "\[Warning\] Using a password"

            # restore the django database
            if [ $QUIET = false ]; then
                echo "Restoring the Django database ..."
            fi
            echo "DROP DATABASE IF EXISTS $DJANGODB;" | mysql --user=$SQLUSER --
↪password=$SQLPASS 2>&1 | grep -v "\[Warning\] Using a password"
            echo "CREATE DATABASE $DJANGODB;" | mysql --user=$SQLUSER --password=
↪$SQLPASS 2>&1 | grep -v "\[Warning\] Using a password"
            mysql --user=$SQLUSER --password=$SQLPASS $DJANGODB < $TEMPDIR/
↪djangodb.sql 2>&1 | grep -v "\[Warning\] Using a password"

            # restart the server and clean-up
```

(continues on next page)

(continued from previous page)

```
            if [ $QUIET = false ]; then
                echo "Restarting the web server ..."
            fi
            apache2ctl start
            rm -rf $TEMPDIR

            if [ $QUIET = false ]; then
                echo "Done!"
            fi

        fi

    else

        echo "Bad argument: file $BACKUPNAME not found" >&2
        usage

    fi


else
    usage
fi
```

4.

Todo: Make the backup script that lives on DynamicsHJC and the cron job list downloadable.

In a different Terminal window, log into the VirtualBox Machines (vbox) account on DynamicsHJC:

```
ssh vbox@dynamicshjc.case.edu
```

Create a script that remotely executes the backup script and moves the archive to the backup drive. Create the file

```
mkdir -p /Volumes/CHIELWIKI/backups/biol300/2017
vim /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh
```

and fill it with the following:

```
#!/bin/bash

#REMOTESSH="ssh hjc@biol300.case.edu"
REMOTESSH="ssh -p 8015 hjc@dynamicshjc.case.edu"

#REMOTESCP="scp -q hjc@biol300.case.edu"
REMOTESCP="scp -q -P 8015 hjc@dynamicshjc.case.edu"

REMOTEFILE=biol300-backup-`date +'%Y%m%dT%H%M%S%z'`.tar.bz2
if [ -z "$1" ]; then
    LOCALFILE=/Volumes/CHIELWIKI/backups/biol300/2017/$REMOTEFILE
else
    LOCALFILE=/Volumes/CHIELWIKI/backups/biol300/2017/$1
fi

$REMOTESSH backup-wiki -q backup $REMOTEFILE
$REMOTESCP:$REMOTEFILE $LOCALFILE
```

(continues on next page)

(continued from previous page)

```
chmod go= $LOCALFILE
$REMOTESSH rm $REMOTEFILE
```

5. Make the script executable:

```
chmod u+x /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh
```

6. Copy the public SSH key from the vbox account on DynamicsHJC to the virtual machine to allow automatic authentication (you will be asked to accept the unrecognized fingerprint of the virtual machine — this is expected — and you will be asked for your password to the virtual machine twice):

```
ssh-keygen -R [dynamicshjc.case.edu]:8015
scp -P 8015 ~/.ssh/id_rsa.pub hjc@dynamicshjc.case.edu:
ssh -p 8015 hjc@dynamicshjc.case.edu "mkdir -p .ssh && cat id_rsa.pub >> .ssh/
↪authorized_keys && rm id_rsa.pub"
```

Test whether automatic authentication is working by trying to log into the virtual machine from the vbox account on DynamicsHJC — you should NOT need to enter your password:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

If this works without you needing to enter a password, automatic authentication is properly configured. You should `logout` to return to the vbox account on DynamicsHJC.

7. In the vbox account on DynamicsHJC, create a backup schedule. Create the file

```
vim /Volumes/CHIELWIKI/backups/biol300/2017/crontab
```

and fill it with the following:

```
################################################################################
#                                    BIOL 300                                  #
################################################################################
# Make hourly backups on the odd-numbered hours (except 1 AM and 3 AM)
0 5,9,13,17,21  * * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh␣
↪hourA.tar.bz2
0 7,11,15,19,23 * * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh␣
↪hourB.tar.bz2
# Make daily backups at 1 AM
0 1 * * 0 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh sunday.tar.bz2
0 1 * * 1 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh monday.tar.bz2
0 1 * * 2 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh tuesday.tar.
↪bz2
0 1 * * 3 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh wednesday.tar.
↪bz2
0 1 * * 4 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh thursday.tar.
↪bz2
0 1 * * 5 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh friday.tar.bz2
0 1 * * 6 /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh saturday.tar.
↪bz2
# Make weekly backups at 3 AM on the 1st, 8th, 15th, and 22nd of the month
0 3 1  * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh `date +'\%Y-\
↪%m'`.tar.bz2
0 3 8  * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh 8th.tar.bz2
0 3 15 * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh 15th.tar.bz2
0 3 22 * * /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh 22nd.tar.bz2
```

8. Schedule the backups. In the vbox account on DynamicsHJC, dump the existing scheduled jobs to a temporary file:

   ```
   crontab -l > /tmp/crontab.old
   ```

   Edit the temporary file, and delete the backup jobs for last year's BIOL 300 Wiki (leave the jobs for BIOL300Dev for now). You can use `Shift+v` in Vim to enter Visual Line mode, the up and down arrow keys to select a block of lines, and `d` to delete them all at once.

   ```
   vim /tmp/crontab.old
   ```

   Now append the new jobs to the old and schedule them:

   ```
   cat {/tmp/crontab.old,/Volumes/CHIELWIKI/backups/biol300/2017/crontab} | crontab
   ```

   Verify that the backup jobs for this year's BIOL 300 Wiki are properly scheduled, and that backup jobs for last year's BIOL 300 Wiki are absent:

   ```
   crontab -l
   ```

9. Log out of the vbox account:

   ```
   logout
   ```

   You can now return to your original Terminal window.

10. Shut down the virtual machine:

    ```
    sudo shutdown -h now
    ```

11. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Automatic backups configured**".

10.6 Install MediaWiki Extensions

1. Start the virtual machine and log in:

   ```
   ssh -p 8015 hjc@dynamicshjc.case.edu
   ```

2. Check for and install system updates on the virtual machine:

   ```
   sudo apt-get update
   sudo apt-get dist-upgrade
   sudo apt-get autoremove
   ```

3. Install these new packages,

Package	Description
phpCAS	PHP implementation of Central Auth Service
php-ldap	LDAP module for PHP for looking up real names when wiki accounts are created
parsoid	Web service converting HTML+RDFa to MediaWiki wikitext and back (prereq for VisualEditor)
dvipng	convert DVI files to PNG graphics (prereq for Math)
ocaml	ML language implementation with a class-based object system (prereq for Math)
texlive-latex-recommended	TeX Live: LaTeX recommended packages (prereq for Math)
texlive-latex-extra	TeX Live: LaTeX additional packages (prereq for Math)
texlive-fonts-recommended	TeX Live: Recommended fonts (prereq for Math)
texlive-lang-greek	TeX Live: Greek (prereq for Math)

using the following:

```
sudo apt-get install php-cas php-ldap dvipng ocaml texlive-latex-recommended␣
↪texlive-latex-extra texlive-fonts-recommended texlive-lang-greek
sudo apt-key advanced --keyserver pgp.mit.edu --recv-keys 90E9F83F22250DD7
sudo apt-add-repository "deb https://releases.wikimedia.org/debian jessie-
↪mediawiki main"
sudo apt-get update
sudo apt-get install parsoid
```

Todo: Change parsoid install command to use a specific version known to work with MW 1.27 (version “0.5.3all” in Fall 2016 or “0.6.1all” in Spring 2017).

4. Restart the web server:

```
sudo apache2ctl restart
```

5. Download these extensions,

Extension	Description
CASAuthentication	integration with university Single Sign-On
EmbedVideo	embed online videos in wiki pages
ImportUsers[1]	create user accounts from CSV files
Lockdown	restrict access to specific namespaces
Math	math typesetting
MobileFrontend	better wiki interface for smartphones
NewArticleTemplate	auto-fill new pages with template text
Realnames	replace user names with real names
ScholasticGrading	assign and report grades
Scribunto	embed scripts in Module namespace (prereq for citation and license templates)
UserMerge	merge contributions and delete users
VisualEditor	WYSIWYG editor for wiki atricles

using the following:

```
cd /var/www/mediawiki/extensions/

# CASAuthentication
git clone https://github.com/CWRUChielLab/CASAuth.git

# EmbedVideo
git clone https://github.com/HydraWiki/mediawiki-embedvideo.git EmbedVideo
git -C EmbedVideo checkout -q v2.5.2  # latest release as of Dec 2016

# ImportUsers
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/ImportUsers.git
git -C ImportUsers checkout -q REL1_27

# Lockdown
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/Lockdown.git
git -C Lockdown checkout -q REL1_27

# Math
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/Math.git
git -C Math checkout -q REL1_27
make -C Math  # build texvc

# MobileFrontend
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/MobileFrontend.git
git -C MobileFrontend checkout -q REL1_27

# NewArticleTemplate
git clone https://github.com/mathiasertl/NewArticleTemplates.git␣
↪NewArticleTemplate
git -C NewArticleTemplate checkout -q cff90b32  # latest commit as of Dec 2016

# Realnames
wget -O Realnames.tar.gz http://ofbeaton.com/releases/Realnames/Realnames_0.3.1_
↪2011-12-25.tar.gz
tar -xf Realnames.tar.gz && rm Realnames.tar.gz

# ScholasticGrading
git clone https://github.com/CWRUChielLab/ScholasticGrading.git

# Scribunto
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/Scribunto.git
git -C Scribunto checkout -q REL1_27

# UserMerge
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/UserMerge.git
git -C UserMerge checkout -q REL1_27

# VisualEditor
git clone https://gerrit.wikimedia.org/r/p/mediawiki/extensions/VisualEditor.git
git -C VisualEditor checkout -q REL1_27
git -C VisualEditor submodule update --init
```

Todo: Consider adding Mathoid installation instructions. In its current form, this instruction set utilizes the Math extension's PNG mode. By changing `$wgDefaultUserOptions['math']` to `'mathml'` in `LocalSettings.php`, an alternate MathML mode can be used. This requires a Mathoid server for runtime equation rendering. The `$wgMathFullRestbaseURL` setting specifies a server in Germany that is free to use but is occasionally unresponsive, causing the wiki to either load slowly or fail to render equations. By building a local Mathoid server, responsiveness and reliability could be guarenteed. However, after much effort, Jeff has not been able to get a Mathoid installation working yet. Since MathML offers few advantages over PNG mode, getting this working is a low priority.

6. Download and install the CAS (Single Sign-On) configuration file:

```
wget -O /var/www/mediawiki/extensions/CASAuth/CASAuthSettings.php https://biol-
↪300-wiki-docs.readthedocs.io/en/latest/_downloads/CASAuthSettings.php
```

Randomize the secret key inside the configuration file:

```
sed -i '/^\$CASAuth\["PwdSecret"\]/s|=".*";|="'$(openssl rand -hex 32)'";|' /var/
↪www/mediawiki/extensions/CASAuth/CASAuthSettings.php
```

Protect the secret key:

```
sudo chmod ug=rw,o= /var/www/mediawiki/extensions/CASAuth/CASAuthSettings.php*
```

If you are curious about the contents of `CASAuthSettings.php`, you can view it here:

CASAuthSettings.php

`Direct link`

```
<?php
# This file controls the configuration of the CASAuth extension.  The defaults
# for these options are defined in CASAuth.php, and are only defined here as
# well for the sake of documentation.  It is highly suggested you should not
# modify the CASAuth defaults unless you know what you are doing.

# Location of the phpCAS library, which is required for this Extension.  For
# more information, see https://wiki.jasig.org/display/CASC/phpCAS for more
# information.
#
# Default: $CASAuth["phpCAS"]="$IP/extensions/CASAuth/CAS";
$CASAuth["phpCAS"]="/usr/share/php";

# Location of the CAS authentication server.  You must set this to the location
# of your CAS server.  You have a CAS server right?
#
# Default: $CASAuth["Server"]="auth.example.com";
$CASAuth["Server"]="login.case.edu";

# An array of servers that are allowed to make use of the Single Sign Out
# feature.  Leave to false if you do not support this feature, of if you dont
# want to use it.  Otherwise, add servers on individual lines.
#  Example:
#    $CASAuth["LogoutServers"][]='cas-logout-01.example.com';
#    $CASAuth["LogoutServers"][]='cas-logout-02.example.com';
#
# Default: $CASAuth["LogoutServers"]=false;
$CASAuth["LogoutServers"]=false;

# Server port for communicating with the CAS server.
#
# Default: $CASAuth["Port"]=443;
```

(continues on next page)

(continued from previous page)

```
$CASAuth["Port"]=443;

# URI Path to the CAS authentication service
#
# Default: $CASAuth["Url"]="/cas/";
$CASAuth["Url"]="/cas/";

# CAS Version.  Available versions are "1.0" and "2.0".
#
# Default: $CASAuth["Version"]="2.0";
$CASAuth["Version"]="2.0";

# Enable auto-creation of users when signing in via CASAuth. This is required
# if the users do not already exist in the MediaWiki use database.  If accounts
# are not regularly being creating, it is recommended that this be set to false
#
# Default: $CASAuth["CreateAccounts"]=false
$CASAuth["CreateAccounts"]=true;

# If the "CreateAccounts" option is set "true", the string below is used as a
# salt for generating passwords for the users.  This salt is not used by
# the normal Mediawiki authentication and is only in place to prevent someone
# from cracking passwords in the database.  This should be changed to something
# long and horrendous to remember.
#
# Default: $CASAuth["PwdSecret"]="Secret";
$CASAuth["PwdSecret"]="<random string>";

# The email domain is appended to the end of the username when the user logs
# in.  This does not affect their email address, and is for aesthetic purposes
# only.
#
# Default: $CASAuth["EmailDomain"]="example.com";
$CASAuth["EmailDomain"]="case.edu";

# Restrict which users can login to the wiki?  If set to true, only the users
# in the $CASAuth["AllowedUsers"] array can login.
#
# Default: $CASAuth["RestrictUsers"]=false
$CASAuth["RestrictUsers"]=false;

# Should CAS users be logged in with the "Remember Me" option?
#
# Default: $CASAuth["RememberMe"]=true;
$CASAuth["RememberMe"]=true;

# If $CASAuth["RestrictUsers"] is set to true, the list of users below are the
# users that are allowed to login to the wiki.
#
# Default: $CASAuth["AllowedUsers"] = false;
$CASAuth["AllowedUsers"] = false;

# If a user is not allowed to login, where should we redirect them to?
#
# Default: $CASAuth["RestrictRedirect"]="http://www.example.com";
$CASAuth["RestrictRedirect"]="http://www.example.com";
```

(continues on next page)

(continued from previous page)

```
# If you dont like the uid that CAS returns (ie. it returns a number) you can
# modify the routine below to return a customized username instead.
#
# Default: Returns the username, untouched
function casNameLookup($username) {
  return $username;
}

# If your users aren't all on the same email domain you can
# modify the routine below to return their email address
#
# Default: Returns $username@EmailDomain
function casEmailLookup($username) {
  global $CASAuth;
  return $username."@".$CASAuth["EmailDomain"];
}

# If you dont like the uid that CAS returns (ie. it returns a number) you can
# modify the routine below to return a customized real name instead.
#
# Default: Returns the username, untouched
#function casRealNameLookup($username) {
#  return $username;
#}

# If you would like to use ldap to retrieve real names, you can use these
# functions instead. Remember to fill in appropriate parameters for ldap.
function casRealNameLookup($username) {
  return @casRealNameLookupFromLDAP($username);
}

function casRealNameLookupFromLDAP($username) {
  try {
    # login to the LDAP server
    $ldap = ldap_connect("ldap://ldap.case.edu");
    $bind = ldap_bind($ldap, "anonymous", "");

    # look up the user's name by user id
    $result = ldap_search($ldap, "ou=People,o=cwru.edu,o=isp", "(uid=$username)");
    $info = ldap_get_entries($ldap, $result);

    $first_name = $info[0]["givenname"][0];
    $last_name  = $info[0]["sn"][0];

    # log out of the server
    ldap_unbind($ldap);

    $realname = $first_name . " " . $last_name;
  } catch (Exception $e) {}

  if ($realname == " " || $realname == "" || $realname == NULL) {
    $realname = $username;
  }

  return $realname;
}
```

7. Modify the Parsoid configuration for the VisualEditor extension. The Parsoid server running on the virtual machine needs to communicate with the web server on the same machine. To simplify this, Parsoid is configured to connect simply to `http://localhost...`. For this to work, the protocol must be changed to HTTPS, and Parsoid must be permitted to ignore mismatched SSL certificates, since it will expect a certificate for "localhost", rather than one for DynamicsHJC, BIOL 300 Wiki, or BIOL300WikiDev. Make the changes by running these commands:

```
sudo sed -i '/^\s*uri:/s|http://|https://|' /etc/mediawiki/parsoid/config.yaml
sudo sed -i '/strictSSL/s|\(\s*\)#\(.*\)|\1\2|' /etc/mediawiki/parsoid/config.yaml
sudo service parsoid restart
```

8. **Todo:** Update the patch for MobileFrontend. For now, skip this step during installation.

   Customizing the mobile site navigation menu by downloading a patch file and applying it:

```
wget -O /var/www/mediawiki/extensions/MobileFrontend/MobileFrontend_customize-
↪icons.patch https://biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/
↪MobileFrontend_customize-icons.patch
patch -d /var/www/mediawiki/extensions/MobileFrontend < /var/www/mediawiki/
↪extensions/MobileFrontend/MobileFrontend_customize-icons.patch
```

   If you are curious about the contents of the patch file, you can view it here:

   MobileFrontend_customize-icons.patch

   `Direct link`

```
diff --git a/includes/skins/SkinMinerva.php b/includes/skins/SkinMinerva.php
index b17ada0..c83d489 100644
--- a/includes/skins/SkinMinerva.php
+++ b/includes/skins/SkinMinerva.php
@@ -312,10 +312,30 @@ class SkinMinerva extends SkinTemplate {
                global $wgMFNearby;

                $items = array(
-                       'home' => array(
-                               'text' => wfMessage( 'mobile-frontend-home-button
↪' )->escaped(),
-                               'href' => Title::newMainPage()->getLocalUrl(),
-                               'class' => 'icon-home',
+                       'syllabus' => array(
+                               'text' => 'Syllabus',
+                               'href' => Title::newFromText( 'nw:Course syllabus
↪' )->getLocalUrl(),
+                               'class' => 'icon-syllabus',
+                       ),
+                       'policies' => array(
+                               'text' => 'Policies',
+                               'href' => Title::newFromText( 'nw:Course policies
↪' )->getLocalUrl(),
+                               'class' => 'icon-policies',
+                       ),
+                       'grades' => array(
+                               'text' => 'Grades',
+                               'href' => Title::newFromText( 'nw:Special:Grades'␣
↪)->getLocalUrl(),
```

(continues on next page)

(continued from previous page)

```
+                               'class' => 'icon-grades',
+                       ),
+                       'labnotebook' => array(
+                               'text' => 'Lab Notebook',
+                               'href' => Title::newFromText( 'Special:UserLogin'␣
↪)->getLocalUrl( 'returnto=nw:Special:MyPage' ),
+                               'class' => 'icon-home',
+                       ),
+                       'termpapers' => array(
+                               'text' => 'Term Papers',
+                               'href' => Title::newFromText( 'nwp:Main Page' )->
↪getLocalUrl(),
+                               'class' => 'icon-termpapers',
                        ),
                        'random' => array(
                                'text' => wfMessage( 'mobile-frontend-random-
↪button' )->escaped(),
diff --git a/less/common/hacks.less b/less/common/hacks.less
index 59fb6b3..6496480 100644
--- a/less/common/hacks.less
+++ b/less/common/hacks.less
@@ -16,10 +16,10 @@ FIXME: Review all of these hacks to see if they still apply.
 .content {
        // Hide cleanup templates by default to non-javascript users as these␣
↪stop them from reading the article itself
        // Note not in issues.less as that is only loaded via JavaScript
-       .ambox,
+       .ambox:not(.simbox),
        #coordinates,
        // Hide article badges, clean-up notices, stub notices, and navigation␣
↪boxes
-       .navbox, .vertical-navbox, .topicon, .metadata {
+       .navbox, .vertical-navbox, .topicon, .metadata:not(.simbox) {
                // It's important as some of these are tables which become␣
↪display: table on larger screens
                display: none !important;
        }
diff --git a/less/common/mainmenu.less b/less/common/mainmenu.less
index 8ccf1ba..5a46631 100644
--- a/less/common/mainmenu.less
+++ b/less/common/mainmenu.less
@@ -104,6 +104,22 @@
                        .background-image('images/menu/home.png');
                }

+               &.icon-syllabus a {
+                       .background-image('/CourseMaterials/Images/mobile-icon-
↪syllabus.png');
+               }
+
+               &.icon-policies a {
+                       .background-image('/CourseMaterials/Images/mobile-icon-
↪policies.png');
+               }
+
+               &.icon-grades a {
+                       .background-image('/CourseMaterials/Images/mobile-icon-
↪grades.png');
```

(continues on next page)

(continued from previous page)

```
+               }
+
+               &.icon-termpapers a {
+                       .background-image('/CourseMaterials/Images/mobile-icon-
↪termpapers.png');
+               }
+
                &.icon-random a {
                        .background-image('images/menu/random.png');
                }
```

9. **Todo:** Explain the function of NewArticleTemplate better here and in terms of the subpages actually used by students.

   Modify the NewArticleTemplate extension so that subpage templates will be used even if the parent page does not exist.

   For example, without this modification, a new page *User:Foo/Bar* will use the User namespace subpage template *MediaWiki:NewArticleTemplate/User/Subpage* if the parent page *User:Foo* already exists, but it will use the User namespace template *MediaWiki:NewArticleTemplate/User* if *User:Foo* does not already exist.

   This modification will force the subpage template to always be used for subpages, regardless of whether the parent page exists or not.

   Download a patch file and apply it:

```
wget -O /var/www/mediawiki/extensions/NewArticleTemplate/NewArticleTemplate_
↪always-use-subpage-template.patch https://biol-300-wiki-docs.readthedocs.io/en/
↪latest/_downloads/NewArticleTemplate_always-use-subpage-template.patch
patch -d /var/www/mediawiki/extensions/NewArticleTemplate < /var/www/mediawiki/
↪extensions/NewArticleTemplate/NewArticleTemplate_always-use-subpage-template.
↪patch
```

   If you are curious about the contents of the patch file, you can view it here:

   NewArticleTemplate_always-use-subpage-template.patch

   `Direct link`

```
--- a/NewArticleTemplate.php
+++ b/NewArticleTemplate.php
@@ -52,12 +52,13 @@ function newArticleTemplates( $newPage ) {
         $isSubpage = false;

         if ( $title->isSubpage() ) {
-                $baseTitle = Title::newFromText(
-                        $title->getBaseText(),
-                        $title->getNamespace() );
-                if ( $baseTitle->exists() ) {
-                        $isSubpage = true;
-                }
+                #$baseTitle = Title::newFromText(
+                #        $title->getBaseText(),
+                #        $title->getNamespace() );
+                #if ( $baseTitle->exists() ) {
+                #        $isSubpage = true;
```

(continues on next page)

(continued from previous page)

```
+                 #}
+                 $isSubpage = true;
         }

         /* we might want to return if this is a subpage */
```

Todo: Add hooks to NewArticleTemplate so that it works with VisualEditor.

Todo: Fork NewArticleTemplate on GitHub and incorporate the “always use subpage template” patch and the hooks for VisualEditor.

10. Fix a bug in the Realnames extension (version 0.3.1)[2]. The Realnames extension includes a bug that causes subpages in the User namespace to lack titles. Download a patch file and apply it:

```
wget -O /var/www/mediawiki/extensions/Realnames/Realnames_ignore-subpage-titles.
↪patch https://biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/Realnames_
↪ignore-subpage-titles.patch
patch -d /var/www/mediawiki/extensions/Realnames < /var/www/mediawiki/extensions/
↪Realnames/Realnames_ignore-subpage-titles.patch
```

If you are curious about the contents of the patch file, you can view it here:

Realnames_ignore-subpage-titles.patch

`Direct link`

```
--- a/Realnames.body.php
+++ b/Realnames.body.php
@@ -265,7 +265,7 @@
       if (in_array($title->getNamespace(), array(NS_USER, NS_USER_TALK))) { //␣
↪User:
         // swap out the specific username from title
         // this overcomes the problem lookForBare has with spaces and␣
↪underscores in names
-        $out->setPagetitle(static::lookForBare($out->getPageTitle(),'/'.
↪static::getNamespacePrefixes().'\s*('.$title->getText().')(?:\/.+)?/'));
+        $out->setPagetitle(static::lookForBare($out->getPageTitle(),'~'.
↪static::getNamespacePrefixes().'\s*('.$title->getText().')(?:/.+)?~'));
       }

       // this should also affect the html head title
@@ -331,7 +331,7 @@
     if (empty($pattern)) {
       // considered doing [^<]+ here to catch names with spaces or underscores,
       // which works for most titles but is not universal
-      $pattern = '/'.static::getNamespacePrefixes().'([^ \t]+)(:\/.+)?/';
+      $pattern = '~'.static::getNamespacePrefixes().'([^ \t]+)(:/.+)?~';
     }
     wfDebugLog('realnames', __METHOD__.": pattern: ".$pattern);
     return preg_replace_callback(
```

[2] This solution is documented here.

Todo: Fork Realnames on GitHub and incorporate the "ignore subpage titles" patch.

11. Fix a bug in the MediaWiki core that causes Lockdown to conflict with certain API calls[3]. In particular, this patch is needed to prevent the "Marking as patrolled failed" error and a silent failure when using the `action=userrights` API module. Download a patch file and apply it:

```
wget -O /var/www/mediawiki/includes/user/Lockdown_api-compatibility.patch https://
↪biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/Lockdown_api-
↪compatibility.patch
patch -d /var/www/mediawiki/includes/user < /var/www/mediawiki/includes/user/
↪Lockdown_api-compatibility.patch
```

If you are curious about the contents of the patch file, you can view it here:

Lockdown_api-compatibility.patch

`Direct link`

```
--- b/User.php
+++ a/User.php
@@ -1355,8 +1355,8 @@
          */
         protected function loadFromUserObject( $user ) {
                 $user->load();
-                $user->loadGroups();
-                $user->loadOptions();
+                #$user->loadGroups();
+                #$user->loadOptions();
                 foreach ( self::$mCacheVars as $var ) {
                         $this->$var = $user->$var;
                 }
```

12. Give the web server ownership of and access to the new files:

```
sudo chown -R www-data:www-data /var/www/
sudo chmod -R ug+rw /var/www/
```

13. Download and install the extension configuration settings:

```
wget -P ~ https://biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/
↪LocalSettings_extensions.php
cat ~/LocalSettings_extensions.php >> /var/www/mediawiki/LocalSettings.php
rm ~/LocalSettings_extensions.php
```

If you are curious about the contents of the configuration file, you can view it here:

LocalSettings_extensions.php

`Direct link`

```
# EXTENSIONS

# CASAuthentication
require_once("$IP/extensions/CASAuth/CASAuth.php");
```

(continues on next page)

[3] This issue is described here, and this patch (used in these instructions) was proposed for fixing it. Later, another patch was proposed, but I have not tested it since the first one works for our purposes. One or both of these patches was expected to be included in 1.27.2+, but as of July 2017 the bug is reported to still be present.

(continued from previous page)

```
# EmbedVideo
wfLoadExtension( 'EmbedVideo' );

# ImportUsers
require_once("$IP/extensions/ImportUsers/ImportUsers.php");

# Interwiki -- automatically enabled above
$wgGroupPermissions['sysop']['interwiki'] = true;

#####################################################################
# Lockdown

# Non-admin users (students) can be prevented from editing the wiki
# after the final deadline for the term paper. Toggle between states
# by running
#   sudo lock-wiki
# which will change the value of this variable.
$wikiLocked = false;

# This line is included just to get this group to appear among the
# list of available user groups.
$wgGroupPermissions['term-paper-deadline-extension']['read'] = true;

# The VisualEditor extension and its companion Parsoid server will
# fail if a combination of MediaWiki's core permission system
# ($wgGroupPermissions) and the Lockdown extension are used to
# restrict access to the wiki. Here we remove all restrictions by
# making the entire wiki readable and writable by anyone (including
# users not logged in), but we will create restrictions below using
# Lockdown.
$wgGroupPermissions['*']['read'] = true;
$wgGroupPermissions['*']['edit'] = true;

# Load the Lockdown extension unless the request to access the wiki
# comes from localhost (IP address 127.0.0.1, i.e., the virtual
# machine itself). This will load Lockdown for all users accessing the
# wiki through the web but not for the Parsoid server running on the
# virtual machine, which is used by the VisualEditor extension. If
# Lockdown is loaded when Parsoid tries to access the wiki (even if no
# configuration variables are set), VisualEditor will fail.
if ( !(array_key_exists('REMOTE_ADDR',$_SERVER) && $_SERVER['REMOTE_ADDR'] ==
↪'127.0.0.1') ) {

    # Load the extension
    require_once("$IP/extensions/Lockdown/Lockdown.php");

    # Restrict editing of all pages to admins (instructors), with a
    # few exceptions if the wiki is unlocked or if a student has a
    # term paper extension ('user' includes all logged in users)
    $wgNamespacePermissionLockdown['*']['edit'] = array('sysop');
    if ( !$wikiLocked ) {
        $wgNamespacePermissionLockdown[NS_TALK]['edit']      = array('user');
        $wgNamespacePermissionLockdown[NS_USER]['edit']      = array('user');
        $wgNamespacePermissionLockdown[NS_USER_TALK]['edit'] = array('user');
        $wgNamespacePermissionLockdown[NS_FILE]['edit']      = array('user');
    } else {
```

(continues on next page)

(continued from previous page)

```
        $wgNamespacePermissionLockdown[NS_USER]['edit']       = array('sysop',
↪'term-paper-deadline-extension');
    }

    # Restrict moving of all pages to sysop, with a few exceptions if
    # the wiki is unlocked ('user' includes all logged in users)
    $wgNamespacePermissionLockdown['*']['move'] = array('sysop');
    if ( !$wikiLocked ) {
        $wgNamespacePermissionLockdown[NS_USER]['move'] = array('user');
        $wgNamespacePermissionLockdown[NS_FILE]['move'] = array('user');
    } else {
        $wgNamespacePermissionLockdown[NS_USER]['move'] = array('sysop', 'term-
↪paper-deadline-extension');
    }
}

# END Lockdown
##########################################################################

# Math
wfLoadExtension( 'Math' );
$wgMathFullRestbaseURL = 'https://api.formulasearchengine.com/';
$wgMathValidModes = array('mathml', 'png', 'source');
$wgDefaultUserOptions['math'] = 'png';
$wgHiddenPrefs[] = 'math';

# MobileFrontend
wfLoadExtension( 'MobileFrontend' );
$wgMFAutodetectMobileView = true;

# NewArticleTemplate
require_once ("$IP/extensions/NewArticleTemplate/NewArticleTemplate.php");
$wgNewArticleTemplatesEnable = true;
$wgNewArticleTemplatesOnSubpages = true;
$wgNewArticleTemplatesNamespaces = array(
    NS_USER      => 1,
    NS_USER_TALK => 1
);
$wgNewArticleTemplates_PerNamespace = array(
    NS_USER      => "MediaWiki:NewArticleTemplate/User",
    NS_USER_TALK => "MediaWiki:NewArticleTemplate/User Talk"
);

# Realnames
require_once("$IP/extensions/Realnames/Realnames.php");

# ScholasticGrading
require_once("$IP/extensions/ScholasticGrading/ScholasticGrading.php");
$wgGroupPermissions['grader']['editgrades'] = true;

# Scribunto
require_once("$IP/extensions/Scribunto/Scribunto.php");
$wgScribuntoDefaultEngine = 'luastandalone';

# UserMerge
wfLoadExtension( 'UserMerge' );
$wgGroupPermissions['*']['usermerge'] = false;
```

(continues on next page)

(continued from previous page)

```
$wgGroupPermissions['sysop']['usermerge'] = true;
$wgUserMergeProtectedGroups = array('sysop');

# VisualEditor
wfLoadExtension( 'VisualEditor' );
$wgDefaultUserOptions['visualeditor-enable'] = 0;
$wgDefaultUserOptions['visualeditor-hidebetawelcome'] = 1;
$wgVirtualRestConfig['modules']['parsoid'] = array(
    'url' => 'http://localhost:8142',    # use http, not https
    'prefix' => 'localhost',
    'domain' => 'localhost',
    'forwardCookies' => true,
);
$wgVisualEditorAvailableNamespaces = array(
    NS_MAIN          => true,
    NS_TALK          => true,
    NS_USER          => true,
    NS_USER_TALK     => true,
    NS_HELP          => true,
    NS_PROJECT       => true,
);

# WikiEditor -- automatically enabled above
$wgDefaultUserOptions['usebetatoolbar'] = true;
$wgDefaultUserOptions['usebetatoolbar-cgd'] = true;
$wgDefaultUserOptions['wikieditor-preview'] = true;
```

Todo: Post my solution for the VisualEditor + Lockdown extension conflict to the discussion board, and then provide a "This solution is documented here" footnote in these instructions.

14. Create database tables for the Math and ScholasticGrading extensions:

```
php /var/www/mediawiki/maintenance/update.php
```

15. Create aliases for URLs to the surveys using "interwiki" links. Interwiki links allow MediaWiki sites to easily create links to one another. For example, using `[[wikipedia:Neuron]]` on our wiki will create a link directly to the Wikipedia article on neurons. Here we use this capability to create aliases, such as `[[survey:1]]`, that make linking to the surveys easier.

    Run the following (you will be prompted for the MySQL password):

```
echo "INSERT INTO interwiki (iw_prefix,iw_url,iw_api,iw_wikiid,iw_local,iw_trans)␣
↪VALUES ('survey','/django/survey/\$1/','','',0,0)" | mysql -u root -p wikidb
```

16. Create a script for toggling the locked state of the wiki by downloading and installing a file:

```
sudo wget -O /usr/local/sbin/lock-wiki https://biol-300-wiki-docs.readthedocs.io/
↪en/latest/_downloads/lock-wiki
sudo chmod +x /usr/local/sbin/lock-wiki
```

    If you are curious about the contents of the script, you can view it here:

    lock-wiki

    `Direct link`

```
#!/bin/bash

# Place this script in /usr/local/sbin and make it executable (chmod +x).
#
# This script will toggle the locked state of the wiki. When the wiki is locked,
# only admins and term paper authors with a deadline extension can edit the
# wiki.


# Function for aborting with an error message

die () {
    echo >&2 "$@"
    exit 1
}


# Require that the user is root.

[ "$UID" -eq 0 ] || die "Aborted: superuser privileges needed (rerun with sudo)"


# Get the current state of the $wikiLocked variable

STATE=$(grep ^\$wikiLocked /var/www/mediawiki/LocalSettings.php | awk '{ print $3␣
↪}')


# Determine the new state

if [ "$STATE" == "true;" ]; then
    NEWSTATE="false;"
    echo "Wiki is now UNLOCKED. Non-admins CAN make edits."
else
    NEWSTATE="true;"
    echo "Wiki is now LOCKED. Non-admins CANNOT make edits (unless they have a␣
↪term paper deadline extension, set using Special:UserRights and listed at␣
↪Special:ListUsers)."
fi


# Set the new state.

sed -i '/^\$wikiLocked/s|= \(.*\);|= '$NEWSTATE'|' /var/www/mediawiki/
↪LocalSettings.php

exit 0
```

17. At this point, individuals with CWRU accounts can log into the wiki, which will create a wiki account for them. Invite the TAs to do this now. After their accounts are created, you should visit (while logged in)

    https://dynamicshjc.case.edu:8014/wiki/Special:UserRights

    For each TA, enter their wiki user name (which should match their CWRU user name), and add them to the following groups:

    - ***administrator***
        - Administrators have special powers on the wiki, such as moving, deleting, or protecting pages.

- ***bureaucrat***
  - Bureaucrats have the ability to change group membership (using the *Special:UserRights* page) for any user, including administrators and other bureaucrats (making this the most powerful group).
- ***grader***
  - Graders can view and edit all grades on the *Special:Grades* page.

Add yourself to the *grader* group as well (you should already be a member of the other groups).

18. Check that all extensions are installed and working properly. Visit Special:Version and compare the list of installed extensions to the list of extensions at the beginning of this section of the instructions.

You can test each of the essential extensions by doing the following:

Extension	Test
CASAuthentication	log in using CWRU's Single Sign-On
EmbedVideo	try adding `{{#ev:youtube|8zRtXBrmvyc||center|Survival Guide}}` to a page
ImportUsers	visit Special:ImportUsers
Lockdown	remove yourself from the *administrator* group using Special:UserRights and try editing a policy page (restore privileges when done!)
Math	try adding `<math>x=\sqrt{2}</math>` to a page
MobileFrontend	click the "Mobile view" link at the bottom of any page (click "Desktop" to return to normal)
NewArticleTemplate	will test later. . .
Realnames	log in and look for your full name in the top right of the page
ScholasticGrading	visit Special:Grades
UserMerge	visit Special:UserMerge
VisualEditor	enable VE under Preferences > Editing, and then try editing a page by clicking "Edit" (not "Edit source")

Todo: Embedded videos don't render in VisualEditor. Fix?

Todo: When attempting to upload files through VisualEditor, it thinks I'm not logged in. Fix!

19. Shut down the virtual machine:

```
sudo shutdown -h now
```

20. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**MediaWiki extensions installed**".

10.7 Install Citation Tools

Todo: Get Wikipedia's advanced citation tools for VisualEditor, especially automatic citation completion from URLs, DOIs, and PMIDs.

Todo: Add screenshots showing the various citation tools in action.

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install the citation templates. Citation templates are plain text templates that allow wiki users to create standardized reference citations by filling in parameters.

   For example, citation templates will transform wiki text such as this,

```
{{cite journal|last1=Cullins|first1=Miranda J.|last2=Gill|first2=Jeffrey P.
↪|last3=McManus|first3=Jeffrey M.|last4=Lu|first4=Hui|last5=Shaw|first5=Kendrick␣
↪M.|last6=Chiel|first6=Hillel J.|title=Sensory Feedback Reduces Individuality by␣
↪Increasing Variability within Subjects|journal=Current Biology|date=October␣
↪2015|volume=25|issue=20|pages=2672-2676|doi=10.1016/j.cub.2015.08.044}}
```

   into properly formatted wiki text, complete with italics and bold markup and hyperlinks:

```
Cullins, Miranda J.; Gill, Jeffrey P.; McManus, Jeffrey M.; Lu, Hui; Shaw,␣
↪Kendrick M.; Chiel, Hillel J. (October 2015). "Sensory Feedback Reduces␣
↪Individuality by Increasing Variability within Subjects". ''Current Biology'' ''
↪'25''' (20): 2672-2676. [[doi]]:[https://dx.doi.org/10.1016%2Fj.cub.2015.08.044␣
↪10.1016/j.cub.2015.08.044].
```

   To install the citation templates, first download this XML file (`citation-templates-1.27.xml`), and import it into the wiki using Special:Import (choose "Import to default locations").

   Complete the installation by editing this wiki page and appending the following:

   - MediaWiki:Common.css

     citation template styles

```
/***********************************************
 *          CITATION TEMPLATE STYLES           *
 ***********************************************/

/* Reset italic styling set by user agent */
cite, dfn {
    font-style: inherit;
}

/* Consistent size for <small>, <sub> and <sup> */
small {
    font-size: 85%;
```

(continues on next page)

(continued from previous page)

```
}
.mw-body sub,
.mw-body sup,
span.reference /* for Parsoid */ {
    font-size: 80%;
}

/* Highlight linked elements (such as clicked references) in blue */
body.action-info :target,
.citation:target {
    background-color: #DEF;  /* Fallback */
    background-color: rgba(0, 127, 255, 0.133);
}

/* Styling for citations. Breaks long urls, etc., rather than overflowing box␣
↪*/
.citation {
    word-wrap: break-word;
}

/* For linked citation numbers and document IDs, where the number need not be␣
↪shown
   on a screen or a handheld, but should be included in the printed version */
@media screen, handheld {
    .citation .printonly {
        display: none;
    }
}

/* Make the list of references smaller */
ol.references,
div.reflist,
div.refbegin {
    font-size: 90%;            /* Default font-size */
    margin-bottom: 0.5em;
}
div.refbegin-100 {
    font-size: 100%;           /* Option for normal fontsize in {{refbegin}}␣
↪*/
}
div.reflist ol.references {
    font-size: 100%;           /* Reset font-size when nested in div.reflist␣
↪*/
    margin-bottom: 0;          /* Avoid double margin when nested in dev.
↪reflist */
    list-style-type: inherit;  /* Enable custom list style types */
}

/* Ensure refs in table headers and the like aren't bold or italic */
sup.reference {
    font-weight: normal;
    font-style: normal;
}

/* Allow hidden ref errors to be shown by user CSS */
span.brokenref {
    display: none;
```

(continues on next page)

(continued from previous page)

```
}
```

Details about how the XML file was created can be found here[1]. The CSS code above was extracted from a version of *MediaWiki:Common.css* on Wikipedia contemporary with the templates included in the XML file.

4. Install refToolbar. refToolbar is a MediaWiki "gadget": a JavaScript-powered add-on for MediaWiki managed by the Gadgets extension. It adds a "Cite" button to the edit toolbar that makes filling citation templates easier by providing a pop-up wizard with text fields for each citation template parameter.

   To install refToolbar, first download this XML file (`refToolbar-1.27.xml`), and import it into the wiki using Special:Import (choose "Import to default locations").

   Complete the installation by editing this wiki page and appending the following:

   - MediaWiki:Gadgets-definition

   ```
   * refToolbar[ResourceLoader|default|dependencies=user.options,mediawiki.
   ↪legacy.wikibits]|refToolbar.js
   * refToolbarBase[ResourceLoader|hidden|rights=hidden]|refToolbarBase.js
   ```

   Details about how the XML file was created can be found here[2].

[1] The XML file `citation-templates-1.27.xml` contains versions of the following pages, originally exported from Wikipedia, that are compatible with MediaWiki 1.27 and contemporary extensions (e.g., Scribunto on branch `REL1_27`):

```
Module:Citation/CS1
Module:Citation/CS1/COinS
Module:Citation/CS1/Configuration
Module:Citation/CS1/Date validation
Module:Citation/CS1/Identifiers
Module:Citation/CS1/Suggestions
Module:Citation/CS1/Utilities
Module:Citation/CS1/Whitelist
Template:Cite AV media
Template:Cite AV media notes
Template:Cite book
Template:Cite conference
Template:Cite DVD notes
Template:Cite encyclopedia
Template:Cite episode
Template:Cite interview
Template:Cite journal
Template:Cite mailing list
Template:Cite map
Template:Cite news
Template:Cite newsgroup
Template:Cite podcast
Template:Cite press release
Template:Cite report
Template:Cite serial
Template:Cite sign
Template:Cite speech
Template:Cite techreport
Template:Cite thesis
Template:Cite web
Template:Reflist
```

Versions compatible with MediaWiki 1.27 were found by first trying the versions of these pages currently on Wikipedia, exported using Special:Export on 2016-08-20. At the time, Wikipedia was running an alpha-phase version of MediaWiki 1.28.0. Luckily, this just worked without errors or rendering glitches. If it had not, I would have used the method described in *Updating Templates Exported from Wikipedia* to find working versions of these pages.

Finally, to eliminate the red links that would otherwise appear in some citations, I manually edited the XML contents of *Module:Citation/CS1/Configuration*. Specifically, in the variable `id_handlers`, I added the interwiki prefix `wikipedia:` to each `link` field; I added the same prefix to the value of `['help page link']`.

The final result is the file `citation-templates-1.27.xml`.

[2] The XML file `refToolbar-1.27.xml` contains versions of the following pages, originally exported from Wikipedia, that are compatible

5. Install reference tooltips. Like refToolbar, reference tooltips are implemented using a MediaWiki gadget. When enabled, placing the mouse cursor over any citation will either provide the reference in a tooltip (if the bibliography is not currently visible on-screen) or highlight the reference in the bibliography (if the bibliography is visible).

   To install reference tooltips, first download this XML file (`reference-tooltips-1.27.xml`), and import it into the wiki using Special:Import (choose "Import to default locations").

   Complete the installation by editing this wiki page and appending the following:

   - MediaWiki:Gadgets-definition

   ```
   * ReferenceTooltips[ResourceLoader|default]|ReferenceTooltips.
   ↪js|ReferenceTooltips.css
   ```

   Details about how the XML file was created can be found here[3].

6. Test that the citation tools are working by pasting this text into any wiki page and pressing the "Show Preview" button:

   ```
   <ref name=note />
   <ref name=book />
   <ref name=journal />
   <ref name=news />
   <ref name=web />

   <div style="height:200px"></div>

   {{reflist|refs=
   <ref name=note>This is a simple note, not a full citation.</ref>
   <ref name=book>{{cite book|last1=Doe|first1=John|last2=Doe|first2=Jane|title=Book␣
   ↪Title|date=1 January 1999|publisher=Publisher|location=New York|isbn=978-1-23-
   ↪456789-7|pages=1-99|edition=1st|url=http://www.example.com|accessdate=1 January␣
   ↪2015}}</ref>
   ```

(continues on next page)

with MediaWiki 1.27 and contemporary extensions (e.g., Scribunto on branch `REL1_27`):

```
MediaWiki:Gadget-refToolbar
MediaWiki:Gadget-refToolbar.js
MediaWiki:Gadget-refToolbarBase.js
MediaWiki:RefToolbar.js
MediaWiki:RefToolbarConfig.js
MediaWiki:RefToolbarLegacy.js
MediaWiki:RefToolbarMessages-en.js
MediaWiki:RefToolbarNoDialogs.js
```

Versions compatible with MediaWiki 1.27 were found by first trying the versions of these pages currently on Wikipedia, exported using Special:Export on 2016-08-20. At the time, Wikipedia was running an alpha-phase version of MediaWiki 1.28.0. Luckily, this just worked without errors or rendering glitches. If it had not, I would have used the method described in *Updating Templates Exported from Wikipedia* to find working versions of these pages.

The final result is the file `refToolbar-1.27.xml`.

[3] The XML file `reference-tooltips-1.27.xml` contains versions of the following pages, originally exported from Wikipedia, that are compatible with MediaWiki 1.27:

```
MediaWiki:Gadget-ReferenceTooltips
MediaWiki:Gadget-ReferenceTooltips.css
MediaWiki:Gadget-ReferenceTooltips.js
```

Versions compatible with MediaWiki 1.27 were found by first trying the versions of these pages currently on Wikipedia, exported using Special:Export on 2017-03-29. At the time, Wikipedia was running MediaWiki 1.29.0. Luckily, this just worked without errors or rendering glitches. If it had not, I would have used the method described in *Updating Templates Exported from Wikipedia* to find working versions of these pages.

I then made one small modification. On Wikipedia, the reference tooltips are confined to the Main, Project, and Help namespaces. We need the tooltips to work anywhere on our wiki, and most critically in the User namespace. To accomplish this, in the XML file I replaced line 7 of *MediaWiki:Gadget-ReferenceTooltips.js* with

```
if (true) {
```

The final result is the file `reference-tooltips-1.27.xml`.

(continued from previous page)

```
<ref name=journal>{{cite␣
↪journal|last1=Doe|first1=John|last2=Doe|first2=Jane|title=Journal article␣
↪title|journal=Journal|date=1 January 1999|volume=1|issue=1|pages=1-99|doi=10.
↪1234/56789|pmid=123456|url=http://www.example.com|accessdate=1 January 2015}}</
↪ref>
<ref name=news>{{cite news|last1=Doe|first1=John|last2=Doe|first2=Jane|title=News␣
↪article title|url=http://www.example.com|accessdate=1 January␣
↪2015|work=Newspaper|agency=Agency|issue=1|publisher=Publisher|date=1 January␣
↪1999}}</ref>
<ref name=web>{{cite web|last1=Doe|first1=John|last2=Doe|first2=Jane|title=Web␣
↪article title|url=http://www.example.
↪com|website=Website|publisher=Publisher|accessdate=1 January 2015}}</ref>
}}
```

Perform these tests:

- Inspect the reference list. If the citations look alright, there are no script errors, and there are no red links, then the citation templates are properly installed.
- Try inserting a new citation template into the page using refToolbar's "Cite" menu, found on the toolbar to the right of "Help" (perform a hard refresh in your browser if you do not see it). You should also be able to insert additional citations to existing named references by clicking "Named references".
- Test the reference tooltips by hovering your cursor over one of the citations (bracketed numbers). If the reference list is simultaneously visible, you should see a box appear in the reference list around the corresponding reference. If the list is not visible, you should see a tooltip containing the reference. You can adjust the height of the `<div>` or resize your browser window to test both cases.

If these tests are successful, you can press "Cancel" instead of "Save page" to avoid saving the test code.

Todo: Add tests for VisualEditor's citation tools.

7. Shut down the virtual machine:

```
sudo shutdown -h now
```

8. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Citation tools installed**".

10.8 Install Message Box Templates

Message box templates are a prerequisite for license templates, and they will be used later to customize the wiki.

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install the message box templates. Message boxes highlight important information by placing a colorful text box on the page (examples can be seen here). These appear frequently on Wikipedia to indicate, for example, that a

page needs attention or that it presents a biased viewpoint. They are highly customizable and could be used to draw students' attention to key ideas. On NeuroWiki, we use message boxes to highlight links to simulations.

To install the message box templates, first download this XML file (`message-box-templates-1.27.xml`), and import it into the wiki using Special:Import (choose "Import to default locations").

Complete the installation by editing this wiki page and appending the following:

- MediaWiki:Common.css

  message box template styles

```
/*********************************************
 *        MESSAGE BOX TEMPLATE STYLES        *
 *********************************************/

/* Messagebox templates */
.messagebox {
    border: 1px solid #aaa;
    background-color: #f9f9f9;
    width: 80%;
    margin: 0 auto 1em auto;
    padding: .2em;
}
.messagebox.merge {
    border: 1px solid #c0b8cc;
    background-color: #f0e5ff;
    text-align: center;
}
.messagebox.cleanup {
    border: 1px solid #9f9fff;
    background-color: #efefff;
    text-align: center;
}
.messagebox.standard-talk {
    border: 1px solid #c0c090;
    background-color: #f8eaba;
    margin: 4px auto;
}
/* For old WikiProject banners inside banner shells. */
.mbox-inside .standard-talk,
.messagebox.nested-talk {
    border: 1px solid #c0c090;
    background-color: #f8eaba;
    width: 100%;
    margin: 2px 0;
    padding: 2px;
}
.messagebox.small {
    width: 238px;
    font-size: 85%;
    /* @noflip */
    float: right;
    clear: both;
    /* @noflip */
    margin: 0 0 1em 1em;
    line-height: 1.25em;
}
.messagebox.small-talk {
```

(continues on next page)

(continued from previous page)

```
    width: 238px;
    font-size: 85%;
    /* @noflip */
    float: right;
    clear: both;
    /* @noflip */
    margin: 0 0 1em 1em;
    line-height: 1.25em;
    background: #F8EABA;
}

/* Cell sizes for ambox/tmbox/imbox/cmbox/ombox/fmbox/dmbox message boxes */
th.mbox-text, td.mbox-text {   /* The message body cell(s) */
    border: none;
    /* @noflip */
    padding: 0.25em 0.9em;     /* 0.9em left/right */
    width: 100%;               /* Make all mboxes the same width regardless␣
↪of text length */
}
td.mbox-image {                /* The left image cell */
    border: none;
    /* @noflip */
    padding: 2px 0 2px 0.9em;  /* 0.9em left, 0px right */
    text-align: center;
}
td.mbox-imageright {           /* The right image cell */
    border: none;
    /* @noflip */
    padding: 2px 0.9em 2px 0;  /* 0px left, 0.9em right */
    text-align: center;
}
td.mbox-empty-cell {           /* An empty narrow cell */
    border: none;
    padding: 0;
    width: 1px;
}

/* Article message box styles */
table.ambox {
    margin: 0 10%;                  /* 10% = Will not overlap with other␣
↪elements */
    border: 1px solid #aaa;
    /* @noflip */
    border-left: 10px solid #1e90ff;  /* Default "notice" blue */
    background: #fbfbfb;
}
table.ambox + table.ambox {      /* Single border between stacked boxes. */
    margin-top: -1px;
}
.ambox th.mbox-text,
.ambox td.mbox-text {            /* The message body cell(s) */
    padding: 0.25em 0.5em;       /* 0.5em left/right */
}
.ambox td.mbox-image {           /* The left image cell */
    /* @noflip */
    padding: 2px 0 2px 0.5em;    /* 0.5em left, 0px right */
}
```

(continues on next page)

(continued from previous page)

```
.ambox td.mbox-imageright {      /* The right image cell */
    /* @noflip */
    padding: 2px 0.5em 2px 0;    /* 0px left, 0.5em right */
}

table.ambox-notice {
    /* @noflip */
    border-left: 10px solid #1e90ff;    /* Blue */
}
table.ambox-speedy {
    /* @noflip */
    border-left: 10px solid #b22222;    /* Red */
    background: #fee;                   /* Pink */
}
table.ambox-delete {
    /* @noflip */
    border-left: 10px solid #b22222;    /* Red */
}
table.ambox-content {
    /* @noflip */
    border-left: 10px solid #f28500;    /* Orange */
}
table.ambox-style {
    /* @noflip */
    border-left: 10px solid #f4c430;    /* Yellow */
}
table.ambox-move {
    /* @noflip */
    border-left: 10px solid #9932cc;    /* Purple */
}
table.ambox-protection {
    /* @noflip */
    border-left: 10px solid #bba;       /* Gray-gold */
}

/* Image message box styles */
table.imbox {
    margin: 4px 10%;
    border-collapse: collapse;
    border: 3px solid #1e90ff;    /* Default "notice" blue */
    background: #fbfbfb;
}
.imbox .mbox-text .imbox {  /* For imboxes inside imbox-text cells. */
    margin: 0 -0.5em;       /* 0.9 - 0.5 = 0.4em left/right.        */
    display: block;         /* Fix for webkit to force 100% width.  */
}
.mbox-inside .imbox {       /* For imboxes inside other templates.  */
    margin: 4px;
}

table.imbox-notice {
    border: 3px solid #1e90ff;    /* Blue */
}
table.imbox-speedy {
    border: 3px solid #b22222;    /* Red */
    background: #fee;             /* Pink */
}
```

(continues on next page)

(continued from previous page)

```
table.imbox-delete {
    border: 3px solid #b22222;    /* Red */
}
table.imbox-content {
    border: 3px solid #f28500;    /* Orange */
}
table.imbox-style {
    border: 3px solid #f4c430;    /* Yellow */
}
table.imbox-move {
    border: 3px solid #9932cc;    /* Purple */
}
table.imbox-protection {
    border: 3px solid #bba;       /* Gray-gold */
}
table.imbox-license {
    border: 3px solid #88a;       /* Dark gray */
    background: #f7f8ff;          /* Light gray */
}
table.imbox-featured {
    border: 3px solid #cba135;    /* Brown-gold */
}

/* Category message box styles */
table.cmbox {
    margin: 3px 10%;
    border-collapse: collapse;
    border: 1px solid #aaa;
    background: #DFE8FF;    /* Default "notice" blue */
}

table.cmbox-notice {
    background: #D8E8FF;    /* Blue */
}
table.cmbox-speedy {
    margin-top: 4px;
    margin-bottom: 4px;
    border: 4px solid #b22222;    /* Red */
    background: #FFDBDB;          /* Pink */
}
table.cmbox-delete {
    background: #FFDBDB;    /* Red */
}
table.cmbox-content {
    background: #FFE7CE;    /* Orange */
}
table.cmbox-style {
    background: #FFF9DB;    /* Yellow */
}
table.cmbox-move {
    background: #E4D8FF;    /* Purple */
}
table.cmbox-protection {
    background: #EFEFE1;    /* Gray-gold */
}

/* Other pages message box styles */
```

(continues on next page)

(continued from previous page)

```
table.ombox {
    margin: 4px 10%;
    border-collapse: collapse;
    border: 1px solid #aaa;       /* Default "notice" gray */
    background: #f9f9f9;
}

table.ombox-notice {
    border: 1px solid #aaa;       /* Gray */
}
table.ombox-speedy {
    border: 2px solid #b22222;    /* Red */
    background: #fee;             /* Pink */
}
table.ombox-delete {
    border: 2px solid #b22222;    /* Red */
}
table.ombox-content {
    border: 1px solid #f28500;    /* Orange */
}
table.ombox-style {
    border: 1px solid #f4c430;    /* Yellow */
}
table.ombox-move {
    border: 1px solid #9932cc;    /* Purple */
}
table.ombox-protection {
    border: 2px solid #bba;       /* Gray-gold */
}

/* Talk page message box styles */
table.tmbox {
    margin: 4px 10%;
    border-collapse: collapse;
    border: 1px solid #c0c090;    /* Default "notice" gray-brown */
    background: #f8eaba;
}
.mediawiki .mbox-inside .tmbox { /* For tmboxes inside other templates. The
↪"mediawiki" class ensures that */
    margin: 2px 0;               /* this declaration overrides other styles␣
↪(including mbox-small above)   */
    width: 100%;                 /* For Safari and Opera */
}
.mbox-inside .tmbox.mbox-small { /* "small" tmboxes should not be small when ␣
↪*/
    line-height: 1.5em;          /* also "nested", so reset styles that are  ␣
↪*/
    font-size: 100%;             /* set in "mbox-small" above.               ␣
↪*/
}

table.tmbox-speedy {
    border: 2px solid #b22222;    /* Red */
    background: #fee;             /* Pink */
}
table.tmbox-delete {
    border: 2px solid #b22222;    /* Red */
```

(continues on next page)

(continued from previous page)

```
}
table.tmbox-content {
    border: 2px solid #f28500;    /* Orange */
}
table.tmbox-style {
    border: 2px solid #f4c430;    /* Yellow */
}
table.tmbox-move {
    border: 2px solid #9932cc;    /* Purple */
}
table.tmbox-protection,
table.tmbox-notice {
    border: 1px solid #c0c090;    /* Gray-brown */
}

/* Disambig and set index box styles */
table.dmbox {
    clear: both;
    margin: 0.9em 1em;
    border-top: 1px solid #ccc;
    border-bottom: 1px solid #ccc;
    background: transparent;
}

/* Footer and header message box styles */
table.fmbox {
    clear: both;
    margin: 0.2em 0;
    width: 100%;
    border: 1px solid #aaa;
    background: #f9f9f9;     /* Default "system" gray */
}
table.fmbox-system {
    background: #f9f9f9;
}
table.fmbox-warning {
    border: 1px solid #bb7070;  /* Dark pink */
    background: #ffdbdb;        /* Pink */
}
table.fmbox-editnotice {
    background: transparent;
}
/* Div based "warning" style fmbox messages. */
div.mw-warning-with-logexcerpt,
div.mw-lag-warn-high,
div.mw-cascadeprotectedwarning,
div#mw-protect-cascadeon,
div.titleblacklist-warning,
div.locked-warning {
    clear: both;
    margin: 0.2em 0;
    border: 1px solid #bb7070;
    background: #ffdbdb;
    padding: 0.25em 0.9em;
}
/* Div based "system" style fmbox messages.
   Used in [[MediaWiki:Readonly lag]]. */
```

(continues on next page)

(continued from previous page)

```
div.mw-lag-warn-normal,
div.fmbox-system {
    clear: both;
    margin: 0.2em 0;
    border: 1px solid #aaa;
    background: #f9f9f9;
    padding: 0.25em 0.9em;
}

/* These mbox-small classes must be placed after all other
   ambox/tmbox/ombox etc classes. "html body.mediawiki" is so
   they override "table.ambox + table.ambox" above. */
html body.mediawiki .mbox-small {   /* For the "small=yes" option. */
    /* @noflip */
    clear: right;
    /* @noflip */
    float: right;
    /* @noflip */
    margin: 4px 0 4px 1em;
    box-sizing: border-box;
    width: 238px;
    font-size: 88%;
    line-height: 1.25em;
}
html body.mediawiki .mbox-small-left {   /* For the "small=left" option. */
    /* @noflip */
    margin: 4px 1em 4px 0;
    box-sizing: border-box;
    overflow: hidden;
    width: 238px;
    border-collapse: collapse;
    font-size: 88%;
    line-height: 1.25em;
}

/* Style for compact ambox */
/* Hide the images */
.compact-ambox table .mbox-image,
.compact-ambox table .mbox-imageright,
.compact-ambox table .mbox-empty-cell {
    display: none;
}
/* Remove borders, backgrounds, padding, etc. */
.compact-ambox table.ambox {
    border: none;
    border-collapse: collapse;
    background: transparent;
    margin: 0 0 0 1.6em !important;
    padding: 0 !important;
    width: auto;
    display: block;
}
body.mediawiki .compact-ambox table.mbox-small-left {
    font-size: 100%;
    width: auto;
    margin: 0;
}
```

(continues on next page)

(continued from previous page)

```
/* Style the text cell as a list item and remove its padding */
.compact-ambox table .mbox-text {
    padding: 0 !important;
    margin: 0 !important;
}
.compact-ambox table .mbox-text-span {
    display: list-item;
    line-height: 1.5em;
    list-style-type: square;
    list-style-image: url(/w/skins/MonoBook/bullet.gif);
}
.skin-vector .compact-ambox table .mbox-text-span {
    list-style-type: disc;
    list-style-image: url(/w/skins/Vector/images/bullet-icon.svg);
    list-style-image: url(/w/skins/Vector/images/bullet-icon.png)\9;
}
/* Allow for hiding text in compact form */
.compact-ambox .hide-when-compact {
    display: none;
}
```

Details about how the XML file was created can be found here[1]. The CSS code above was extracted from a version of *MediaWiki:Common.css* on Wikipedia contemporary with the templates included in the XML file.

4. Test that the message box templates are working by pasting this text into any wiki page and pressing the “Show Preview” button:

```
{{ambox
 | type  = notice
 | small = {{{small|left}}}
 | text  = This is a test using a small, left-aligned message box.
}}
```

(continues on next page)

[1] The XML file `message-box-templates-1.27.xml` contains versions of the following pages, originally exported from Wikipedia, that are compatible with MediaWiki 1.27 and contemporary extensions (e.g., Scribunto on branch `REL1_27`):

```
Module:Arguments
Module:Category handler
Module:Category handler/blacklist
Module:Category handler/config
Module:Category handler/data
Module:Category handler/shared
Module:Message box
Module:Message box/configuration
Module:Namespace detect/config
Module:Namespace detect/data
Module:No globals
Module:Yesno
Template:Ambox
Template:Cmbox
Template:Fmbox
Template:Imbox
Template:Mbox
Template:Ombox
Template:Tmbox
```

Versions compatible with MediaWiki 1.27 were found by first trying the versions of these pages currently on Wikipedia, exported using Special:Export on 2016-08-21. At the time, Wikipedia was running an alpha-phase version of MediaWiki 1.28.0. Luckily, this just worked without errors or rendering glitches. If it had not, I would have used the method described in *Updating Templates Exported from Wikipedia* to find working versions of these pages.

The final result is the file `message-box-templates-1.27.xml`.

(continued from previous page)

```
{{ambox
 | type = content
 | text = This is a test using a large, centered message box. It even has an
↪exclamation point!
}}

{{ambox
 | type = speedy
 | text = Danger, Will Robinson!
}}
```

The boxes that appear should be colored blue, orange, and red sequentially, with circle-"i", circle-"!", and triangle-"!" icons (perform a hard refresh in your browser if you do not see the boxes).

If this test is successful, you can press "Cancel" instead of "Save page" to avoid saving the test code.

Todo: Add screenshots showing what the message box test result should look like.

5. Shut down the virtual machine:

```
sudo shutdown -h now
```

6. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Message box templates installed**".

10.9 Install License Templates

Copyright licenses can be assigned to files when they are uploaded to the wiki. Note that *Install Message Box Templates* is a prerequisite for the license templates.

1. Start the virtual machine and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Install the license templates. License templates are selected for each file at time of upload.

   To install the license templates, first download this XML file (`license-templates-1.27.xml`), and import it into the wiki using Special:Import (choose "Import to default locations").

   Complete the installation by editing these wiki pages and filling them with the following:

   - MediaWiki:Wm-license-cc-zero-text

   ```
   This file is made available under the [[wikipedia:Creative Commons|Creative
   ↪Commons]] [//creativecommons.org/publicdomain/zero/1.0/deed.en CC0 1.0
   ↪Universal Public Domain Dedication].
   ```

   - MediaWiki:Wm-license-cc-zero-explanation

```
The person who associated a work with this deed has dedicated the work to the␣
↪[[wikipedia:public domain|public domain]] by waiving all of his or her␣
↪rights to the work worldwide under copyright law, including all related and␣
↪neighboring rights, to the extent allowed by law. You can copy, modify,␣
↪distribute and perform the work, even for commercial purposes, all without␣
↪asking permission.
```

Details about how the XML file was created can be found here[1]. The two *MediaWiki* pages listed above had to be manually filled because they would not export from Wikipedia properly.

4. Test that the license templates are working by visiting Special:Upload, selecting each license from the drop-down menu one at a time, and inspecting the message boxes that appear. If the licenses look alright, there are no script errors, and there are no red links, then the license templates are working properly.

5. Shut down the virtual machine:

```
sudo shutdown -h now
```

6. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**License templates installed**".

[1] The XML file `license-templates-1.27.xml` contains versions of the following pages, originally exported from Wikipedia, that are compatible with MediaWiki 1.27 and contemporary extensions (e.g., Scribunto on branch `REL1_27`):

```
MediaWiki:Licenses
Module:Math
Template:Category handler
Template:Cc-by-4.0
Template:Cc-by-sa-4.0
Template:Cc-by-sa-4.0,3.0,2.5,2.0,1.0
Template:Cc-zero
Template:Center
Template:File other
Template:Free media
Template:GFDL
Template:Image other
Template:ImageNoteControl
Template:Lang
Template:License migration
Template:License migration announcement
Template:Max
Template:PD-art
Template:PD-old
Template:PD-old-100
Template:PD-US
Template:PD-USGov
Template:R from move
Template:Self
Template:Template other
```

Versions compatible with MediaWiki 1.27 were found by first trying the versions of these pages currently on Wikipedia, exported using Special:Export on 2016-08-21. At the time, Wikipedia was running an alpha-phase version of MediaWiki 1.28.0. Luckily, this just worked without errors or rendering glitches. If it had not, I would have used the method described in *Updating Templates Exported from Wikipedia* to find working versions of these pages.

I then made some modifications. On Wikipedia, the list of licenses is quite long and includes many that will not be useful to our students. To remove these, in the XML file I deleted several lines from *MediaWiki:Licenses*.

Finally, to eliminate the red links that would otherwise appear in some licenses, I manually edited the XML contents of many templates. Specifically, wherever there would be a red link, I added the interwiki prefix `wikipedia:`. For example, `[[Creative Commons]]` appears several times across a few templates. I replaced each instance with `[[wikipedia:Creative Commons|Creative Commons]]`.

The final result is the file `license-templates-1.27.xml`.

10.10 Customize the Wiki

1. Start the virtual machine and log in:

   ```
   ssh -p 8015 hjc@dynamicshjc.case.edu
   ```

2. Check for and install system updates on the virtual machine:

   ```
   sudo apt-get update
   sudo apt-get dist-upgrade
   sudo apt-get autoremove
   ```

3. Change the name of the front page of the wiki. Move the existing Main Page to *Course syllabus* (capital “C”, lowercase “s”) and check “Leave a redirect behind” (the Move button is located near the top-right of the page, next to “View history”).

   Next, delete the moved page (now called *Course syllabus*) so that last year’s version of the page will import properly.

   Finally, edit the following page to make clicking on the wiki logo direct the user to the right place. Replace its contents with the following:

   - MediaWiki:Mainpage

   ```
   Course syllabus
   ```

4. Modify the navigation sidebar. Edit the following page, and replace its contents with the following:

   - MediaWiki:Sidebar

   ```
   * navigation
   ** Course syllabus|Course syllabus
   ** Course policies|Course policies
   ** Special:Grades|Course grades
   ** Student list|Student list
   ** recentchanges-url|recentchanges
   ** randompage-url|randompage
   ** Help:Editing|help
   * SEARCH
   * TOOLBOX
   * LANGUAGES
   ```

5. Add a reminder for students to the file upload form to include CWRU IDs in filenames. Edit the following page, and append the following:

   - MediaWiki:Uploadtext

   ```
   <font color="red">'''''Please prefix all your filenames with your CWRU ID!''''
   ↪'</font>
   ```

6. Add a reminder for students to the WikiEditor dialog box to include CWRU IDs and file extensions in filenames. Edit the following page, and replace its contents with the following:

   - MediaWiki:Wikieditor-toolbar-file-target

   ```
   Filename: (PLEASE INCLUDE YOUR CWRU ID AND THE FILE EXTENSION!)
   ```

Todo: Add a similar message about prefixing uploaded file names with a student's user name to VisualEditor.

7. Add custom styling. Edit the following page, and append the following:

   - MediaWiki:Common.css

```
/**********************************************
*          CUSTOM BIOL 300 WIKI STYLES          *
**********************************************/

table.course-info {
    float: right;
    border: 1px solid #aaa;
    width:300px;
    text-align: center;
    border-collapse: collapse;
}
table.course-info caption {
    background: #2c659c;
    color: #fff;
    font-weight: bold;
}
table.course-info th {
    background: #dcdcdc;
}
div.course-example {
    background: #f9f9f9;
    border: 1px solid #ddd;
    padding: 1em;
    margin-top: 8px;
}
```

8. Create the term paper benchmark template. Edit the following page, and fill it with the following:

   - Template:Termpaper

```
'''{{#if: {{{1|}}} | [[User:{{{1}}}]]'s | My}} Term Paper Benchmarks'''

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Term Paper Proposal | Term Paper␣
↪Proposal]]. Due ? at 12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Model Plan | Model Plan]]. Due ? at␣
↪12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Benchmark I: Introduction | Benchmark I:␣
↪Introduction]]. Due ? at 12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Benchmark II: Model Description |␣
↪Benchmark II: Model Description]]. Due ? at 12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Benchmark III: Results | Benchmark III:␣
↪Results]]. Due ? at 12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Benchmark IV: Discussion | Benchmark IV:␣
↪Discussion]]. Due ? at 12:45 PM.

* [[{{#if: {{{1|}}} | User:{{{1}}}}}/Final Term Paper | Final Term Paper]].␣
↪Due ? at 5 PM.
```

(continues on next page)

(continued from previous page)

```
<noinclude>
<hr>
'''Arguments'''
# ''Username'' (optional): Links are created as subpages to this user's page.␣
↪Defaults to using the current page as parent to linked subpages if omitted.␣
↪Also replaces "My Term Paper Benchmarks" with "<nowiki>[[User:<username>]]</
↪nowiki>'s Term Paper Benchmarks".
</noinclude>
```

9. Create a template for student user pages. Edit the following page, and fill it with the following:

   - MediaWiki:NewArticleTemplate/User

   ```
   {{termpaper}}

   <!--
       ATTENTION: DO NOT MAKE ANY CHANGES TO THIS PAGE!
       PRESS THE SAVE BUTTON BELOW, AND YOUR PERSONAL TERM
       PAPER TEMPLATE WILL BE CREATED. YOU SHOULD USE THE
       LINKS THAT APPEAR THERE FOR SAVING YOUR WORK.
   -->
   ```

   Todo: Consider what should be done about the User namespace template if I can't get VisualEditor to work with NewArticleTemplate.

10. Create a blank template for term paper benchmarks. Create the page MediaWiki:NewArticleTemplate/User/Subpage and leave it blank (you will first need to place some initial content on the page and save for it to be created, and then delete that content and save again).

11. Create templates for instructor and student comments. Edit the following page, and fill it with the following:

    - MediaWiki:NewArticleTemplate/User Talk

    ```
    == Student Comments ==

    == Instructor Comments ==
    ```

12. Hide the "Category: Articles using small message boxes" message that appears at the bottom of some pages. Edit the following page and fill it with the following:

    - Category:Articles using small message boxes

    ```
    __HIDDENCAT__
    ```

13. Shut down the virtual machine:

```
sudo shutdown -h now
```

14. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Wiki customization complete**".

10.11 Copy Old Wiki Contents

1. Start biol300_2017 and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Start the old (2016) virtual machine if it is not already running.

4. Export the policy and unit pages from last year's wiki. You may have decided after last spring's semester ended to make some changes to these pages on either the live wiki (biol300.case.edu) or the development wiki (biol300dev.case.edu). To determine where the latest changes can be found, visit these pages to view the latest edits on each wiki:

   - Live wiki recent changes
   - Dev wiki recent changes

   If you made changes in parallel to both wikis that you want to keep, you may need to manually propogate some of those edits.

   After determining where you should export pages from, visit *Special:Export* on that wiki (live wiki | dev wiki). Select these options,

   - Check "Include only the current revision, not the full history"
   - Do NOT check "Include templates"
   - Check "Save as file"

   and paste the names of the following pages into the text field:

   BIOL 300 Wiki pages to export

```
Course policies
    Class attendance
    Team evaluations
    Concepts and attitudes surveys
    Problem benchmarks
    Comments on other students' term paper benchmarks
    Term Paper Template
    Model Plan
        Exemplary Model Plan
    Selecting a published model to reproduce
    Term paper proposal
        Exemplary Term Paper Proposal 1
        Exemplary Term Paper Proposal 2
    Benchmark I: Introduction
        Exemplary Introduction Draft 1
        Exemplary Introduction Draft 2
        Exemplary Introduction Draft 3
        Exemplary Introduction Draft 4
        Exemplary Introduction Draft 5
    Benchmark II: Model Description
        Exemplary Model Description Draft 1
        Exemplary Model Description Draft 2
        Exemplary Model Description Draft 3
        Exemplary Model Description Draft 4
        Exemplary Model Description Draft 5
```

(continues on next page)

(continued from previous page)

```
    Benchmark III: Results
        Exemplary Results Draft 1
        Exemplary Results Draft 2
        Exemplary Results Draft 3
        Exemplary Results Draft 4
        Exemplary Results Draft 5
    Benchmark IV: Discussion
        Exemplary Discussion Draft 1
        Exemplary Discussion Draft 2
        Exemplary Discussion Draft 3
        Exemplary Discussion Draft 4
        Exemplary Discussion Draft 5
    Term paper
        Exemplary Final Term Paper 1
        Exemplary Final Term Paper 2
        Exemplary Final Term Paper 3
        Exemplary Final Term Paper 4
        Exemplary Final Term Paper 5
        Exemplary Final Term Paper 6
    Student Presentations
        Presentation Guidelines
    Rationale for modeling and modeling tools
    Plagiarism
    Don't Cheat
    The rules apply to everyone, including you

Course syllabus
    List of Discussion benchmarks
    List of final term papers
Student list
Help:Editing
BIOL 300 Wiki:Copyrights
Template:Instructor links
User:Hjc
User:Jpg18
Sandbox
Wiki Todo list
Hjc Todo list
```

Export the pages and save the XML file when given the option.

5. On the 2017 virtual machine, visit Special:Import and upload the XML file obtained from the 2016 wiki (choose "Import to default locations").
6. Since it is possible that the list above is incomplete, visit Special:WantedPages to determine which pages are still missing.

   There will be several missing pages related to the class that should be ignored. These are the pages begining with the slash ("/") character, such as */Model Plan*. These appear in the list because the *Template:Termpaper* page uses relative links for the term paper benchmanks.

   If necessary, repeat steps 4-5 until no relevant pages are missing.
7. The following pages need to be updated with new dates, personnel, office hours times, etc., or out-dated contents need to be cleared:
   - Course syllabus
   - Course policies

   - Student list
   - Student Presentations
   - Template:Termpaper

8. If you'd like to add or remove term paper benchmark exemplars, now is a good time to do so. If you remove any, be sure to also delete associated files and images from the "Files to Import" directory.

   Todo: The "Files to Import" directory is now hosted online. Add instructions for modifying it.

9. On the virtual machine, download and then import into the wiki a collection of images and files. This includes the wiki logo, favicon, and figures from benchmark exemplars:

   ```
   wget -P ~ https://biol-300-wiki-docs.readthedocs.io/en/latest/_downloads/BIOL-300-
   ↪Files-to-Import.tar.bz2
   tar -xjf ~/BIOL-300-Files-to-Import.tar.bz2 -C ~
   php /var/www/mediawiki/maintenance/importImages.php --user=Hjc ~/BIOL-300-Files-
   ↪to-Import
   sudo apache2ctl restart
   rm -rf ~/BIOL-300-Files-to-Import*
   ```

   If you'd like to view the collection of files, you can download it to your personal machine here: `BIOL-300-Files-to-Import.tar.bz2`

10. **Todo:** Update this step with instructions for adding files to the online "BIOL-300-Files-to-Import.tar.bz2" archive, and move the `fetch_wiki_files.sh` script to an external file in the docs source.

    Visit Special:WantedFiles to determine which files are still missing. Files on this list that are struckthrough are provided through Wikimedia Commons and can be ignored.

    If there are only a few files missing, download them individually from the old wiki, add them to the "Files to Import" directory, and upload them manually.

    If there are many files missing (which is likely to happen if you added a new exemplar), you can use the following script to download them from the old wiki in a batch.

    On your personal machine, create the file

    ```
    vim fetch_wiki_files.sh
    ```

    and fill it with the following:

    fetch_wiki_files.sh

    ```
    #!/bin/bash

    # This script should be run with a single argument: the path to a file
    # containing the names of the files to be downloaded from the wiki,
    # each on its own line and written in the form "File:NAME.EXTENSION".
    INPUT="$1"
    if [ ! -e "$INPUT" ]; then
        echo "File \"$INPUT\" not found!"
        exit 1
    fi

    # MediaWiki provides an API for querying the server. We will use it
    ```

(continues on next page)

(continued from previous page)

```
# to determine the URLs for directly downloading each file.
WIKIAPI=https://biol300.case.edu/w/api.php

# The result of our MediaWiki API query will be provided in JSON and
# will contain some unnecessary meta data. We will use this Python
# script to parse the query result. It specifically extracts only the
# URLs for directly downloading each file.
SCRIPT="
import sys, json

data = json.loads(sys.stdin.read())['query']['pages']

for page in data.values():
    if 'invalid' not in page and 'missing' not in page:
        print page['imageinfo'][0]['url']
"

# Create the directory where downloaded files will be saved
DIR=downloaded_wiki_files
mkdir -p $DIR

# While iterating through the input line-by-line...
while read FILENAME; do
    if [ "$FILENAME" ]; then
        echo -n "Downloading \"$FILENAME\" ... "

        # ... query the server for a direct URL to the file ...
        JSON=`curl -s -d "action=query&format=json&prop=imageinfo&iiprop=url&
↪titles=$FILENAME" $WIKIAPI`

        # ... parse the query result to obtain the naked URL ...
        URL=`echo $JSON | python -c "$SCRIPT"`

        if [ "$URL" ];
        then
            # ... download the file
            cd $DIR
            curl -s -O $URL
            cd ..
            echo "success!"
        else
            echo "not found!"
        fi
    fi
done < "$INPUT"
```

Make the script executable:

```
chmod u+x fetch_wiki_files.sh
```

Copy the bulleted list of missing files found at Special:WantedFiles and paste them into this file:

```
vim wanted_files_list.txt
```

You can use this Vim command to clean up the list:

```
:%s/^\s*File:\(.*\)\%u200f\%u200e (\d* link[s]*)$/File:\1/g
```

Finally, execute the script to download all the files in the list:

```
./fetch_wiki_files.sh wanted_files_list.txt
```

The downloaded files will be saved in the `downloaded_wiki_files` directory. Copy these to the "Files to Import" directory and upload them to the new wiki manually or using the `importImages.php` script used in step 9.

11. Protect every image and media file currently on the wiki from vandalism. Access the database:

```
mysql -u root -p wikidb
```

Enter the <MySQL password> when prompted. Execute these SQL commands (the magic number 6 refers to the File namespace):

```
INSERT IGNORE INTO page_restrictions (pr_page,pr_type,pr_level,pr_cascade,pr_
↪expiry)
    SELECT p.page_id,'edit','sysop',0,'infinity' FROM page AS p WHERE p.page_
↪namespace=6;
INSERT IGNORE INTO page_restrictions (pr_page,pr_type,pr_level,pr_cascade,pr_
↪expiry)
    SELECT p.page_id,'move','sysop',0,'infinity' FROM page AS p WHERE p.page_
↪namespace=6;
INSERT IGNORE INTO page_restrictions (pr_page,pr_type,pr_level,pr_cascade,pr_
↪expiry)
    SELECT p.page_id,'upload','sysop',0,'infinity' FROM page AS p WHERE p.page_
↪namespace=6;
```

Type `exit` to quit.

You do not need to protect any wiki pages because the Lockdown extension for MediaWiki does this for you!

12. Shut down the virtual machine:

```
sudo shutdown -h now
```

13. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**2016 wiki contents migrated**".

10.12 Create Django Survey Sessions

Although the forms for the Django surveys are already in the Django database, dated instances of the surveys, called sessions, are not yet created.

1. Start biol300_2017 and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Start biol300_2016 if it is not already running.

4. Export the Django survey sessions from the previous year (you will be prompted three times for last year's BIOL 300 Wiki password):

```
ssh hjc@biol300.case.edu "/var/www/django/manage.py dumpdata survey.surveysession␣
↪> ~/surveysessions_2016.json"
scp hjc@biol300.case.edu:surveysessions_2016.json ~/
ssh hjc@biol300.case.edu "rm ~/surveysessions_2016.json"
```

5. Import the survey sessions (note that this will delete any survey sessions that you have created manually, which would appear on this admin page). The first command (`sed`) will do a search-and-replace on the year to make updating the dates in step 7 a little faster.

```
sed -i 's/2016/2017/g' ~/surveysessions_2016.json
/var/www/django/manage.py loaddata ~/surveysessions_2016.json
rm ~/surveysessions_2016.json
```

6. Close all the imported survey sessions so that students cannot begin accessing them:

```
echo "UPDATE survey_surveysession SET open=0;" | mysql -u root -p djangodb
```

   Enter the <MySQL password> when prompted.

7. If you haven't done so already, update the dates on the syllabus:

   https://dynamicshjc.case.edu:8014/wiki/Course_syllabus

8. Update the dates assigned to each survey session in the Django database. For each survey listed on the syllabus, click its link to navigate to the survey. From there, click the "Edit Survey" button. Change the survey date to match the new date listed on the syllabus, and then save the survey session. Do this for every survey on the syllabus.

9. Navigate to the Django admin page for managing all survey sessions:

   https://dynamicshjc.case.edu:8014/django/admin/survey/surveysession/

   Look over the list of sessions for any that still need to be updated. Fix these now.

10. Test the links to surveys found on the Course syllabus to make sure they point to the right sessions and that they have the correct dates.

11. Shut down the virtual machine:

```
sudo shutdown -h now
```

12. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Django survey sessions created**".

10.13 Create ScholasticGrading Assignments

The ScholasticGrading MediaWiki extension must be populated with all the assignments for the entire semester.

1. Start biol300_2017 and log in:

```
ssh -p 8015 hjc@dynamicshjc.case.edu
```

2. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

3. Start biol300_2016 if it is not already running.
4. If you haven't done so already, give yourself (and any other TAs) privileges to edit grades. Navigate to

   https://dynamicshjc.case.edu:8014/wiki/Special:UserRights

   and add each instructor to the "grader" group.
5. Export the ScholasticGrading student groups, assignments, and assignment-group-membership tables from the previous year (remember to add the MySQL password from last Spring to the first command, and note that you will be prompted three times for last year's BIOL 300 Wiki password):

```
ssh hjc@biol300.case.edu "mysqldump --user=root --password=<BIOL 300 Wiki 2016␣
↪MySQL password> -c biol300_wiki scholasticgrading_group scholasticgrading_
↪assignment scholasticgrading_groupassignment > ~/scholasticgrading_2016.sql"
scp hjc@biol300.case.edu:scholasticgrading_2016.sql ~/
ssh hjc@biol300.case.edu "rm ~/scholasticgrading_2016.sql"
```

   Todo: For exporting content from the 2016 wiki, the commands above are fine, but they should be revised for the future by using the `wikidb` database name.

6. Import the ScholasticGrading tables (note that this will delete any groups or assignments you have created manually, which would appear on the Manage groups page and the Manage assignments page).

   The first command (`sed`) will do a search-and-replace on the year to make updating the dates in the next step a little faster.

```
sed -i 's/2016/2017/g' ~/scholasticgrading_2016.sql
mysql -u root -p wikidb < ~/scholasticgrading_2016.sql
rm ~/scholasticgrading_2016.sql
```

   Enter the <MySQL password> when prompted.
7. Update the dates for each assignment. Navigate to the assignments management page,

   https://dynamicshjc.case.edu:8014/w/index.php?title=Special:Grades&action=assignments

   First, check that the point total (listed at the bottom of the page) is 100 for the "All students" group.

   Next, for each assignment, determine the appropriate new date using the syllabus and update it. Remember to "Apply changes" often so you don't lose work!
8. To see what the assignment list looks like to a student, navigate to the groups management page,

   https://dynamicshjc.case.edu:8014/w/index.php?title=Special:Grades&action=groups

   Add yourself to the "All students" group and press "Apply changes". Next, navigate to the "View all user scores" page,

   https://dynamicshjc.case.edu:8014/w/index.php?title=Special:Grades&action=viewalluserscores

   You should see the complete list of assignments under your name, along with the racetrack. Look this over to see if everything looks correct. All the runners should be all the way on the left. Once you are done, remove yourself from the "All students" group.
9. Shut down the virtual machine:

```
sudo shutdown -h now
```

10. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**ScholasticGrading assignments created**".

10.14 Activate Bridged Networking

1. If biol300_2017 is running, log in and shut it down:

   ```
   ssh -p 8015 hjc@dynamicshjc.case.edu
   sudo shutdown -h now
   ```

2. If biol300_2016 is running, shut it down as well:

   ```
   ssh hjc@biol300.case.edu
   sudo shutdown -h now
   ```

3. In VirtualBox, select biol300_2017 and choose Settings > Network. Change the "Attached to" setting to "Bridged Adapter". Under Advanced, verify that the MAC address matches the BIOL 300 Wiki's (see *Looking Up MAC Addresses* for instructions).

4. Start biol300_2017 and log in using the new address:

   ```
   ssh-keygen -R biol300.case.edu
   ssh hjc@biol300.case.edu
   ```

5. Fix the backup script. Log into the vbox account on DynamicsHJC:

   ```
   ssh vbox@dynamicshjc.case.edu
   ```

   and edit the file

   ```
   vim /Volumes/CHIELWIKI/backups/biol300/2017/backup-biol300.sh
   ```

   comment out the definitions for `REMOTESSH` and `REMOTESCP` containing `dynamicshjc.case.edu` and uncomment the definitions containing `biol300.case.edu`.

6. Fix SSH authentication into the BIOL 300 Wiki from the vbox account. You will be asked to accept the unrecognized fingerprint of the virtual machine — this is expected — but you should NOT need to enter your password. The IP address is the static IP for biol300.case.edu, obtained using `host biol300.case.edu`.

   ```
   ssh-keygen -R biol300.case.edu -R 129.22.139.44
   ssh hjc@biol300.case.edu
   ```

   If this works without you needing to enter a password, automatic authentication is properly configured. You should `logout` to return to the vbox account on DynamicsHJC, and `logout` again to return to the virtual machine.

7. Shut down the virtual machine:

   ```
   sudo shutdown -h now
   ```

8. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Bridged networking enabled**".

10.15 Delete VirtualBox Snapshots

Todo: Walkthrough deleting all but the most recent snapshot to save disk space.

CHAPTER 11

Building from Last Year

1. If last year's virtual machine is running, log in and shut it down:

```
ssh hjc@biol300.case.edu
sudo shutdown -h now
```

2. In VirtualBox, create a clone of last year's virtual machine by selecting it and Machine > Clone, and choosing the following settings:

   - **New machine name**
     - Name: biol300_YYYY (new year here)
     - Do NOT check "Reinitialize the MAC address of all network cards"
   - **Clone type**
     - Full clone
   - **Snapshots**
     - Current machine state

3. After cloning the virtual machine, select it and choose Machine > Group. Click on the new group's name ("New group"), click Group > Rename Group, and rename the group to the current year. Finally, drag-and-drop the new group into the "BIOL 300" group.

4. Using VirtualBox, take a snapshot of the current state of the new virtual machine. Name it "**Cloned from biol300_YYYY**" (old year).

5. Start the new virtual machine and log in:

```
ssh hjc@biol300.case.edu
```

6. Check for and install system updates on the virtual machine:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

7. Install this new package if necessary,

Package	Description
jq	lightweight and flexible command-line JSON processor

using the following:

```
sudo apt-get install jq
```

8. If it is not already installed, download and install the wiki reset script:

```
sudo wget -O /usr/local/sbin/reset-wiki https://biol-300-wiki-docs.readthedocs.io/
↪en/latest/_downloads/reset-wiki
```

Set the MySQL password inside the script:

```
read -s -r -p "MySQL password: " DBPASS && echo && sudo sed -i '/^SQLPASS=/s|=.*|=
↪'$DBPASS'|' /usr/local/sbin/reset-wiki; DBPASS=
```

Choose a password for a new wiki account that will be created by the reset script and store it inside the script (you will need this again in step 11):

```
read -s -r -p "Wiki password for new bot account (min 8 chars): " BOTPASS && echo␣
↪&& sudo sed -i '/^BOTPASS=/s|=.*|='$BOTPASS'|' /usr/local/sbin/reset-wiki;␣
↪BOTPASS=
```

Protect the passwords:

```
sudo chown root:www-data /usr/local/sbin/reset-wiki
sudo chmod ug=rwx,o= /usr/local/sbin/reset-wiki
```

If you are curious about the contents of the script, you can view it here:

reset-wiki

`Direct link`

```
#!/bin/bash


######################################################################
##                                                                  ##
## Global variables                                                 ##
##                                                                  ##
######################################################################

# The user name and, optionally, the password of a wiki account that
# will be used to interact with the wiki through the MediaWiki API.
# User names and passwords are case-sensitive. If the password is left
# blank here, you will be prompted for it when it is needed.

BOTNAME=SemesterResetBot
BOTPASS=

# The names of the MediaWiki and Django databases and, optionally, the
# MySQL password. If the password is left blank here, you will be
# prompted for it when it is needed.
```

(continues on next page)

(continued from previous page)

```
WIKIDB=wikidb
DJANGODB=djangodb
SQLUSER=root
SQLPASS=

# The user names of wiki accounts that should be ignored by this
# script. The following will be preserved for accounts in this list:
# MediaWiki and Django accounts, files uploaded, User and Private
# pages. If you have a TA from last semester who is continuing to work
# with you in the upcoming semester, you can include them here so that
# you do not need to re-setup their account privileges. User names are
# case-sensitive and should be separated with spaces.

IGNOREUSERS="Hjc Jpg18"

# The titles of pages that should be ignored by this script. User and
# Private pages of accounts in the IGNOREUSERS list are automatically
# preserved, as are all pages outside of the User, User talk, Private,
# and Private talk namespaces. Use this variable to preserve important
# pages in these namespaces. Page titles are case-sensitive. Spaces in
# titles should be replaced with underscores, and titles should be
# separated by spaces.

IGNOREPAGES="Private:Term_papers"

# The user names of wiki accounts that will be used for merging old
# accounts. User names are case-sensitive.

MERGEDSTUDENTNAME=FormerStudent
MERGEDINSTRUCNAME=FormerInstructor

# MediaWiki provides an API for querying the server. We will use it
# to log into the bot account.

WIKIAPI="https://$(hostname).case.edu/w/api.php"

# Since the UserMerge extension lacks an API, to use it we must
# simulate human actions through a browser using the normal access
# point used by human visitors to the wiki.

WIKIINDEX="https://$(hostname).case.edu/w/index.php"

# Maintaining a login session requires that we store an HTTP cookie
# file.

COOKIE="/tmp/cookie.txt"

# MediaWiki namespace constants

NS_TALK=1
NS_USER=2
NS_USER_TALK=3
NS_PRIVATE=100
NS_PRIVATE_TALK=101

# The functions below set and use the following additional global
```

(continues on next page)

(continued from previous page)

```
# variables

BOTISLOGGEDIN=false
EDITTOKEN=
USERRIGHTSTOKEN=


######################################################################
##                                                                  ##
## Logging                                                          ##
##                                                                  ##
######################################################################

# Create a log directory if it does not exist.

LOGDIR="/var/log/reset-wiki"
mkdir -p $LOGDIR

# Log files are dated.

LOGFULLPATH="$LOGDIR/reset-wiki-$(date +'%Y-%m-%d-%H%M%S').log"

# Redirect stdout ( > ) into a named pipe ( >() ) running tee, which
# allows text printed to the screen to also be written to a file.

exec > >(tee -i "$LOGFULLPATH")

# Also redirect stderr ( 2> ) to stdout ( &1 ) so that it too is
# printed to the screen and written to the file.

exec 2>&1


######################################################################
##                                                                  ##
## Function: userexists                                             ##
##                                                                  ##
## Checks whether a user account exists on the wiki. Returns 0 if   ##
## it exists or 1 otherwise. Prompts for the account name if one    ##
## was not provided as an argument when the function was called.    ##
##                                                                  ##
######################################################################

function userexists {

    local USER=$1
    local RESPONSE
    local MISSING
    local USERID


    # If the name of account is not passed as a function argument, ask
    # for it now.

    if [ -z "$USER" ]; then

        read -r -p "User name (to check for existence): " USER
```

(continues on next page)

(continued from previous page)

```
        if [ -z "$USER" ]; then
            echo >&2 "User existence check aborted: You must enter a username"
            return 1
        fi
    fi


    # Request basic information about the account.

    RESPONSE=$(curl -s $WIKIAPI \
        -d "action=query" \
        -d "format=json" \
        -d "list=users" \
        -d "ususers=$USER")

    MISSING=$(echo $RESPONSE | jq '.query.users[0].missing')
    USERID=$( echo $RESPONSE | jq '.query.users[0].userid')

    if [ "$MISSING" == null -a "$USERID" != null ]; then

        # User exists, so return true (0)

        return 0

    else

        # User is missing, so return false (1)

        return 1

    fi

} # end userexists


######################################################################
##                                                                  ##
## Function: usergroups                                             ##
##                                                                  ##
## Prints out the list of groups to which a user on the wiki        ##
## belongs. The result will be in the form of a JSON array of       ##
## strings if the user exists, or "null" otherwise. Prompts for the ##
## account name if one was not provided as an argument when the     ##
## function was called.                                             ##
##                                                                  ##
######################################################################

function usergroups {

    local USER=$1
    local RESPONSE
    local USERGROUPS


    # If the name of account is not passed as a function argument, ask
    # for it now.
```

(continues on next page)

(continued from previous page)

```
    if [ -z "$USER" ]; then

        read -r -p "User name (to check for groups): " USER

        if [ -z "$USER" ]; then
            echo >&2 "User group check aborted: You must enter a username"
            return 1
        fi
    fi


    # Request group information about the account.

    RESPONSE=$(curl -s $WIKIAPI \
        -d "action=query" \
        -d "format=json" \
        -d "list=users" \
        -d "ususers=$USER" \
        -d "usprop=groups")

    USERGROUPS=$(echo $RESPONSE | jq '.query.users[0].groups')

    echo $USERGROUPS

} # end usergroups


######################################################################
##                                                                  ##
## Function: loginbot                                               ##
##                                                                  ##
## Logs the bot into the wiki so that it can perform automated      ##
## tasks. Prompts for the bot account password if one was not       ##
## provided as an argument when the function was called. If         ##
## successful, the function saves an HTTP cookie associated with    ##
## the login session and updates the BOTISLOGGEDIN global variable. ##
##                                                                  ##
######################################################################

function loginbot {

    local BOTPASS=$1
    local RESPONSE
    local LOGINTOKEN
    local LOGINSTATUS
    local WARNING
    local ERROR


    # If the bot account password is not passed as a function
    # argument, ask for it now.

    if [ -z "$BOTPASS" ]; then
        read -s -r -p "Enter $BOTNAME's password: " BOTPASS
        echo
        echo
```

(continues on next page)

(continued from previous page)

```
    fi


    # Delete any old cookie files.

    rm -f "$COOKIE"


    # Logging into the wiki is a two-step process. This first step
    # should result in the receipt of an HTTP cookie (saved to a file
    # using -c) and a login token (a random string) that is paired to
    # the cookie.

    RESPONSE=$(curl -s -c "$COOKIE" $WIKIAPI \
        -d "action=query" \
        -d "meta=tokens" \
        -d "type=login" \
        -d "format=json")

    LOGINTOKEN=$(echo $RESPONSE | jq '.query.tokens.logintoken | @uri' | tr -d '"
↪')

    if [ "$LOGINTOKEN" == "null" ]; then
        WARNING=$(echo $RESPONSE | jq '.warnings.tokens | .["*"]' | tr -d '"')
        echo >&2 "Login token retrieval failed: $WARNING"
        return 1
    fi

    # The second step for logging in submits the cookie (submitted
    # from a file using -b) and login token, along with the username
    # and password, and should result in the receipt of a modified
    # HTTP cookie (saved to the same file using -c). A valid return
    # URL is required to log in but is not used by this script.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIAPI \
        -d "action=clientlogin" \
        -d "format=json" \
        -d "username=$BOTNAME" \
        -d "password=$BOTPASS" \
        -d "loginreturnurl=http://localhost" \
        -d "logintoken=$LOGINTOKEN")

    LOGINSTATUS=$(echo $RESPONSE | jq '.clientlogin.status' | tr -d '"')

    if [ "$LOGINSTATUS" == "FAIL" ]; then
        ERROR=$(echo $RESPONSE | jq '.clientlogin.message' | tr -d '"')
        echo >&2 "Login failed: $ERROR"
        return 1
    fi

    if [ "$LOGINSTATUS" == "PASS" ]; then
        echo "Login successful."
        BOTISLOGGEDIN=true
        return 0
    else
        echo >&2 "Login failed: Result was expected to be 'PASS' but got '
↪$LOGINSTATUS' instead"
```

(continues on next page)

(continued from previous page)

```
        BOTISLOGGEDIN=false
        return 1
    fi

} # end loginbot


######################################################################
##                                                                  ##
## Function: createandpromoteaccount                                ##
##                                                                  ##
## Creates a new account on the wiki. Requires that the username of ##
## the new account is passed as the first argument when the         ##
## function is called. Prompts for a password. Can optionally       ##
## accept any of the following flags for promoting the account to a ##
## user group: --bot --bureaucrat --sysop                           ##
##                                                                  ##
######################################################################

function createandpromoteaccount {

    local NEWUSER=$1
    local FLAGS=${@:2} # all args after the first
    local NEWPASS1=
    local NEWPASS2=


    # Ask for a password

    read -s -r -p "Choose a password for $NEWUSER (min 8 chars): " NEWPASS1
    echo
    read -s -r -p "Retype the password: " NEWPASS2
    echo
    echo

    until [ "$NEWPASS1" == "$NEWPASS2" -a "${#NEWPASS1}" -ge "8" ]; do
        echo "Passwords did not match or are too short, try again."
        echo
        retryprompt
        read -s -r -p "Choose a password for $NEWUSER (min 8 chars): " NEWPASS1
        echo
        read -s -r -p "Retype the password: " NEWPASS2
        echo
        echo
    done


    # Actually create the account and promote it to the appropriate
    # user groups

    php /var/www/mediawiki/maintenance/createAndPromote.php --force $FLAGS "
↪$NEWUSER" "$NEWPASS1"

} # end createandpromoteaccount


######################################################################
```

(continues on next page)

(continued from previous page)

```
##                                                                   ##
## Function: getedittoken                                            ##
##                                                                   ##
## Requests an edit token from the wiki. Edit tokens are random      ##
## strings of letters and numbers needed to take most actions on     ##
## the wiki, including merging users. Stores the edit token in the   ##
## global variable EDITTOKEN.                                        ##
##                                                                   ##
#######################################################################

function getedittoken {

    local RESPONSE
    local WARNING


    # Request the edit token.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIAPI \
        -d "action=tokens" \
        -d "format=json")

    if [ "$(echo $RESPONSE | jq '.tokens')" == "[]" ]; then
        WARNING=$(echo $RESPONSE | jq '.warnings.tokens | .["*"]' | tr -d '"')
        echo >&2 "Edit token retrieval failed: $WARNING"
        return 1
    fi

    EDITTOKEN=$(echo $RESPONSE | jq '.tokens.edittoken | @uri' | tr -d '"')
    return 0

} # end getedittoken


#######################################################################
##                                                                   ##
## Function: getuserrightstoken                                      ##
##                                                                   ##
## Requests a userrights token from the wiki. Userrights tokens are ##
## random strings of letters and numbers needed to make changes to   ##
## user properties, such as group membership. Stores the userrights ##
## token in the global variable USERRIGHTSTOKEN.                     ##
##                                                                   ##
#######################################################################

function getuserrightstoken {

    local RESPONSE
    local WARNING


    # Request the userrights token.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIAPI \
        -d "action=query" \
        -d "meta=tokens" \
        -d "type=userrights" \
```

(continues on next page)

(continued from previous page)

```
        -d "format=json")

    USERRIGHTSTOKEN=$(echo $RESPONSE | jq '.query.tokens.userrightstoken | @uri'␣
↪| tr -d '"')

    if [ "$USERRIGHTSTOKEN" == "null" ]; then
        WARNING=$(echo $RESPONSE | jq '.warnings.tokens | .["*"]' | tr -d '"')
        echo >&2 "Userrights token retrieval failed: $WARNING"
        return 1
    fi

    return 0

} # end getuserrightstoken


######################################################################
##                                                                  ##
## Function: demotesysop                                            ##
##                                                                  ##
## Removes a wiki user from the sysop group. Requires that the bot  ##
## is already logged in and an edit token is already acquired, so   ##
## run the loginbot and getedittoken functions first. Prompts for   ##
## the username of the account to be demoted if one was not         ##
## provided as an argument when the function was called.            ##
##                                                                  ##
######################################################################

function demotesysop {

    local USER=$1
    local RESPONSE
    local BOTISBUREAUCRAT


    # If the name of the sysop account to be demoted is not passed as
    # a function argument, ask for it now.

    if [ -z "$USER" ]; then

        read -r -p "User name (to demote): " USER

        if [ -z "$USER" ]; then
            echo >&2 "Demote sysop aborted: You must enter a username"
            return 1
        fi
    fi


    # Verify that the bot can edit user rights.

    BOTISBUREAUCRAT=$(usergroups $BOTNAME | jq '. | contains(["bureaucrat"])')

    if [ "$BOTISBUREAUCRAT" != "true" ]; then
        echo >&2 "Demote sysop aborted: Bot must be added to the bureaucrat group"
        return 1
    fi
```

(continues on next page)

(continued from previous page)

```
    # Get a userrights token.

    until getuserrightstoken; do
        echo
        retryprompt
    done


    # Request the demotion.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIAPI \
        -d "action=userrights" \
        -d "format=json" \
        -d "user=$USER" \
        -d "remove=sysop" \
        -d "token=$USERRIGHTSTOKEN")

    if [ "$(echo $RESPONSE | jq '.userrights.removed[]' | tr -d '"')" == "sysop"␣
↪]; then
        return 0
    else
        echo >&2 "Demote sysop failed: User may have already been demoted"
        return 1
    fi

} # end demotesysop


######################################################################
##                                                                  ##
## Function: usermerge                                              ##
##                                                                  ##
## Merges one wiki account ("old") into another ("new") and deletes ##
## the former. All contributions belonging to the old account       ##
## (edits, uploads) are reassigned to the new account. The logs are ##
## revised as well. Depends on the UserMerge MediaWiki extension.   ##
## Requires that the bot is already logged in and an edit token is  ##
## already acquired, so run the loginbot and getedittoken functions ##
## first. Sysop users cannot be merged into another account, so use ##
## demotesysop first if necessary. User names of the old and new    ##
## accounts can be passed as the first and second function          ##
## arguments, respectively. If either argument is missing, the       ##
## function will prompt for the user names. User names are          ##
## case-sensitive.                                                  ##
##                                                                  ##
######################################################################

function usermerge {

    local OLDUSER=$1
    local NEWUSER=$2
    local RESPONSE
    local ERROR
    local SUCCESS
    local OUTPUT="/tmp/response.html"
```

(continues on next page)

(continued from previous page)

```
    # If either the old or new user was not passed as a function
    # argument, ask for both now.

    if [ -z "$OLDUSER" -o -z "$NEWUSER" ]; then

        read -r -p "Old user (merge from): " OLDUSER

        if [ -z "$OLDUSER" ]; then
            echo >&2 "User merge aborted: You must enter a username"
            return 1
        fi

        read -r -p "New user (merge to): " NEWUSER

        if [ -z "$NEWUSER" ]; then
            echo >&2 "User merge aborted: You must enter a username"
            return 1
        fi
    fi


    # Request to merge users.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIINDEX \
        -d "title=Special:UserMerge" \
        -d "wpolduser=$OLDUSER" \
        -d "wpnewuser=$NEWUSER" \
        -d "wpdelete=1" \
        -d "wpEditToken=$EDITTOKEN")

    # Attempt to detect any error messages in the response.

    ERROR=$(echo $RESPONSE | sed -n -e "s/.*\(<span class=\"error\">\)\s*\([^<>
↪]*\)\s*\(<\/span>\).*/\2/ p")

    if [ -n "$ERROR" ]; then
        echo >&2 "User merge aborted: $ERROR"
        return 1
    fi

    # Attempt to detect a success message in the response.

    SUCCESS=$(echo $RESPONSE | sed -n -e "s/.*\(Merge from [^<>]* is complete\.\).
↪*/\1/ p")

    if [ -n "$SUCCESS" ]; then
        echo "Success: $SUCCESS"
        return 0
    fi

    # The function would have returned by now if either the error or
    # success pattern matching steps had found something.

    echo $RESPONSE > $OUTPUT
    echo >&2 "User merge aborted: The server responded in an unexpected way."
```

(continues on next page)

(continued from previous page)

```
    echo >&2 "I've saved the response in $OUTPUT if you'd like to inspect it."
    return 1

} # end usermerge


######################################################################
##                                                                  ##
## Function: validatesqlpass                                        ##
##                                                                  ##
## The first time this function is executed, it will prompt for the ##
## MySQL password if none was provided at the top of this file (it  ##
## is recommended that this file is kept free of passwords for      ##
## improved security). It then tests the password. This repeats if  ##
## the password was incorrect until a correct password is given or  ##
## the user aborts. If this function is executed again later after  ##
## the correct password was obtained, it will silently double check ##
## that the password is still working and return.                   ##
##                                                                  ##
######################################################################

function validatesqlpass {

    # If the password is not provided at the top of this file
    # (it is recommended that this file is kept free of passwords for
    # improved security) and this function has not been executed
    # already, prompt for the password now.

    if [ -z "$SQLPASS" ]; then

        read -s -r -p "Enter the MySQL password: " SQLPASS
        echo
        echo


    # If the password was provided at the top of this file, or if it
    # was acquired when this function was previously executed, test
    # the password, and if it works, return.

    elif $(echo "" | mysql --user=$SQLUSER --password=$SQLPASS >/dev/null 2>&1);␣
↪then

        return 0

    fi


    # No password or an incorrect password was provided at the top of
    # this file, and this function has not been executed previously to
    # obtain the correct password, so enter this loop.

    while true; do

        # Check again whether the password works.

        if $(echo "" | mysql --user=$SQLUSER --password=$SQLPASS >/dev/null 2>&1);
↪ then
```

(continues on next page)

(continued from previous page)

```
            # If it works this time, provide feedback to the user and
            # return.

            echo "The MySQL password you entered is correct."
            echo
            return 0

        else

            # If it does not work this time, provide feedback to the
            # user and ask again.

            echo "The MySQL password you entered is incorrect."
            echo
            retryprompt
            read -s -r -p "Enter the MySQL password: " SQLPASS
            echo
            echo

        fi

    done

} # end validatesqlpass


######################################################################
##                                                                  ##
## Function: querywikidb                                            ##
##                                                                  ##
## Submits a MySQL query to the MediaWiki database. validatesqlpass ##
## should be executed at least once before submitting a query.      ##
##                                                                  ##
######################################################################

function querywikidb {

    local QUERY=$1


    # Submit the query and suppress a warning about using passwords on
    # the command line.

    echo "$QUERY" | mysql --user=$SQLUSER --password=$SQLPASS -N $WIKIDB 2>&1 |␣
↪grep -v "\[Warning\] Using a password"

} # end querywikidb


######################################################################
##                                                                  ##
## Function: querydjangodb                                          ##
##                                                                  ##
## Submits a MySQL query to the Django database. validatesqlpass    ##
## should be executed at least once before submitting a query.      ##
##                                                                  ##
```

(continues on next page)

(continued from previous page)

```
######################################################################

function querydjangodb {

    local QUERY=$1


    # Submit the query and suppress a warning about using passwords on
    # the command line.

    echo "$QUERY" | mysql --user=$SQLUSER --password=$SQLPASS -N $DJANGODB 2>&1 |␣
↪grep -v "\[Warning\] Using a password"

} # end querydjangodb


######################################################################
##                                                                  ##
## Function: listnonsysops                                          ##
##                                                                  ##
## Prints out a list of all users on the wiki who are not           ##
## admins/sysops (usually students or recently demoted TAs).        ##
## Usernames provided as arguments to the function will be filtered ##
## out of the list.                                                 ##
##                                                                  ##
######################################################################

function listnonsysops {

    local EXCLUSIONS


    # Create a regular expression that matches any user names that
    # were passed as arguments to this function call.

    EXCLUSIONS="^($(echo $* | tr -s ' ' '|'))$"


    # Query for all users who are not members of the sysop group and
    # who are not among the excluded user name list.

    querywikidb "SELECT user_name FROM user WHERE user_id NOT IN (SELECT ug_user␣
↪FROM user_groups WHERE ug_group = 'sysop') AND user_name NOT REGEXP '$EXCLUSIONS
↪';"

} # end listnonsysops


######################################################################
##                                                                  ##
## Function: listsysops                                             ##
##                                                                  ##
## Prints out a list of all users on the wiki who are admins/sysops ##
## (instructors and the bot account). Usernames provided as         ##
## arguments to the function will be filtered out of the list.      ##
##                                                                  ##
######################################################################
```

(continues on next page)

(continued from previous page)

```
function listsysops {

    local EXCLUSIONS


    # Create a regular expression that matches any user names that
    # were passed as arguments to this function call.

    EXCLUSIONS="^($(echo $* | tr -s ' ' '|'))$"


    # Query for all users who are members of the sysop group and who
    # are not among the excluded user name list.

    querywikidb "SELECT user_name FROM user WHERE user_id IN (SELECT ug_user FROM␣
↪user_groups WHERE ug_group = 'sysop') AND user_name NOT REGEXP '$EXCLUSIONS';"

} # end listsysops


######################################################################
##                                                                  ##
## Function: listpages                                              ##
##                                                                  ##
## Prints out a list of all pages from specified namespaces.        ##
## Prompts for one or more namespace constants (integers separated  ##
## by spaces) if one was not provided as an argument when the       ##
## function was called.                                             ##
##                                                                  ##
######################################################################

function listpages {

    local NS_CONSTANTS="$*"


    # If namespace constants are not passed as function arguments, ask
    # for them now.

    if [ -z "$NS_CONSTANTS" ]; then

        read -r -p "Namespace constants (integers separated by spaces): " NS_
↪CONSTANTS

        if [ -z "$NS_CONSTANTS" ]; then
            echo >&2 "List pages aborted: You must enter one or more namespace␣
↪constants"
            return 1
        fi
    fi


    # Query for all pages in the specified namespaces

    for NS in $NS_CONSTANTS; do
```

(continues on next page)

(continued from previous page)

```
        if [ $NS == $NS_TALK ]; then
            NSTITLE="Talk"
        elif [ $NS == $NS_USER ]; then
            NSTITLE="User"
        elif [ $NS == $NS_USER_TALK ]; then
            NSTITLE="User talk"
        elif [ $NS == $NS_PRIVATE ]; then
            NSTITLE="Private"
        elif [ $NS == $NS_PRIVATE_TALK ]; then
            NSTITLE="Private talk"
        else
            NSTITLE="UNKNOWNNAMESPACE"
        fi

        querywikidb "SELECT CONCAT('$NSTITLE:', page_title) FROM page WHERE page_
↪namespace=$NS;"

    done

} # end listpages


######################################################################
##                                                                  ##
## Function: listfiles                                              ##
##                                                                  ##
## Prints out a list of all files uploaded to the wiki. Usernames   ##
## provided as arguments to the function will have their uploaded   ##
## files be filtered out of the list.                               ##
##                                                                  ##
######################################################################

function listfiles {

    local EXCLUSIONS


    # If any user names were passed as arguments with this function
    # call, construct a MySQL phrase that will exclude their uploaded
    # files from the query

    if [ -n "$*" ]; then
        EXCLUSIONS=$(echo "$*" | tr -s ' ' '|')
        EXCLUSIONS="WHERE img_user NOT IN (SELECT user_id FROM user WHERE user_
↪name REGEXP '$EXCLUSIONS')"
    else
        EXCLUSIONS=""
    fi


    # Query for all files uploaded by anyone not on the excluded user
    # list

    querywikidb "SELECT CONCAT('File:', img_name) FROM image $EXCLUSIONS;"

} # end listfiles
```

(continues on next page)

(continued from previous page)

```
######################################################################
##                                                                  ##
## Function: deletepageorfile                                       ##
##                                                                  ##
## Deletes a wiki page or uploaded file. Requires that the bot is   ##
## already logged in and an edit token is already acquired, so run  ##
## the loginbot and getedittoken functions first. Prompts for the   ##
## page or file title if one was not provided as an argument when   ##
## the function was called. For pages in namespaces other than      ##
## Main, include the namespace prefix. For files, include "File:".  ##
##                                                                  ##
######################################################################

function deletepageorfile {

    local TITLE=$1
    local TITLEESCAPED
    local RESPONSE


    # If the page title is not passed as a function argument, ask for
    # it now.

    if [ -z "$TITLE" ]; then

        read -r -p "Page/file title (to delete): " TITLE

        if [ -z "$TITLE" ]; then
            echo >&2 "Page/file delete aborted: You must enter a page or file␣
↪title"
            return 1
        fi
    fi


    # Replace underscores in the title with spaces

    TITLE=$(echo $TITLE | tr '_' ' ')


    # Create a safe-for-URL version of the title with escaped special
    # characters

    TITLEESCAPED=$(echo "{\"title\":\"$TITLE\"}" | jq '.title | @uri' | tr -d '"')


    # Request the deletion.

    RESPONSE=$(curl -s -b "$COOKIE" -c "$COOKIE" $WIKIAPI \
        -d "action=delete" \
        -d "format=json" \
        -d "title=$TITLEESCAPED" \
        -d "token=$EDITTOKEN" \
        -d "reason=Mass deletion of former student content")

    if [ "$(echo $RESPONSE | jq '.delete.title' | tr -d '"')" == "$TITLE" ]; then
```

(continues on next page)

(continued from previous page)

```
        echo "Successful deletion: $TITLE"
        return 0
    else
        echo >&2 "Failed: $TITLE NOT deleted: $RESPONSE"
        return 1
    fi

} # end deletepageorfile


######################################################################
##                                                                  ##
## Function: continueprompt                                         ##
##                                                                  ##
## Asks the user if they want to continue with the script. Aborts   ##
## if they press any key other than 'c'.                            ##
##                                                                  ##
######################################################################

function continueprompt {

    local PROMPT

    read -r -n 1 -p "Press 'c' to continue, or any other key to quit: " PROMPT
    echo
    if [ "$PROMPT" != "c" ]; then
        exit 0
    else
        echo
    fi

} # end continueprompt


######################################################################
##                                                                  ##
## Function: retryprompt                                            ##
##                                                                  ##
## Asks the user if they want to retry some action that failed.     ##
## Aborts if they press any key other than 'r'.                     ##
##                                                                  ##
######################################################################

function retryprompt {

    local PROMPT

    read -r -n 1 -p "Press 'r' to retry, or any other key to quit: " PROMPT
    echo
    if [ "$PROMPT" != "r" ]; then
        exit 0
    else
        echo
    fi

} # end retryprompt
```

(continues on next page)

(continued from previous page)

```
######################################################################
##                                                                  ##
## Main: Functions are actually called here                         ##
##                                                                  ##
######################################################################


# Since this script is very powerful, require sudo

if [ "$(whoami)" != "root" ]; then
    echo >&2 "Aborted: superuser priveleges needed (rerun with sudo)"
    exit 1
fi


# Acquire the MySQL password

validatesqlpass


echo "\
This script can be used to clean up the wiki in preparation for a new
semester. It will *irreversibly* remove all student content, including
lab notebooks, term papers, files uploaded by students, grades (but
not assignments), survey responses (but not surveys), and all related
log entries.

       *******************************************************
       **  BEFORE PROCEEDING, YOU SHOULD CLONE THE VIRTUAL  **
       **     MACHINE TO PRESERVE LAST SEMESTER'S DATA!     **
       *******************************************************

This script will perform the following actions:

    1.  Log into the wiki using a bot account. The bot will perform
        many of the operations on the wiki.

    2.  Merge all student accounts into the \"$MERGEDSTUDENTNAME\" account.

    3.  Merge the accounts of all former TAs into the
        \"$MERGEDINSTRUCNAME\" account.

    4.  Reversibly delete pages in the following namespaces:
            - User & Private            (lab notebooks and term papers)
            - User talk & Private talk  (comments on student work)

    5.  Reversibly delete files uploaded by students.

    6.  Permanently delete the pages, files, and related log entries.

    7.  Delete grades.

    8.  Clean up wiki logs.

    9.  Delete survey responses.
```

(continues on next page)

(continued from previous page)

```
You will be prompted at every step for permission to continue.

All output from this script will be recorded in the file
$LOGFULLPATH.
"
continueprompt


echo "\
**********************************************************************
**                 STEP 1: LOG INTO THE BOT ACCOUNT                 **
**********************************************************************
"

# Ensure that the bot account exists

until userexists "$BOTNAME"; do
    echo "\
The bot account, $BOTNAME, does not exist on the wiki. This
script will create it now. The account will also be promoted to the
bot, bureaucrat, and administrator (sysop) groups.
"
    continueprompt
    until createandpromoteaccount "$BOTNAME" --bot --bureaucrat --sysop; do
        echo
        retryprompt
    done
    echo
done


# Ensure that the bot account has the correct privileges, in case it
# was manually demoted

until $(usergroups $BOTNAME | jq '. | contains(["bot", "bureaucrat", "sysop"])')␣
↪== "true"; do
    echo "\
The bot account must belong to specific user groups so that it can
have necessary privileges. This script will promote it now to the bot,
bureaucrat, and administrator (sysop) groups. You will be asked to
select a new password for the account. You may reuse the existing
password if you like.
"
    continueprompt
    until createandpromoteaccount "$BOTNAME" --bot --bureaucrat --sysop; do
        echo
        retryprompt
    done
    echo
done


# Log into the account

if [ -n "$BOTPASS" ]; then
    echo "Attempting bot login using password stored in this script ..."
    echo
```

(continues on next page)

(continued from previous page)

```
    loginbot "$BOTPASS"
else
    loginbot
fi

until $BOTISLOGGEDIN; do
    echo
    retryprompt
    loginbot
done
echo
continueprompt


echo "\
**********************************************************************
**                  STEP 2: MERGE FORMER STUDENTS                   **
**********************************************************************
"

# Ensure that the account for merging students exists

until userexists "$MERGEDSTUDENTNAME"; do
    echo "\
The account that will be used to merge students, $MERGEDSTUDENTNAME, does
not exist on the wiki. This script will create it now.
"
    continueprompt
    until createandpromoteaccount "$MERGEDSTUDENTNAME"; do
        echo
        retryprompt
    done
    echo
done


# Acquire the list of former students

USERLIST=$(listnonsysops $IGNOREUSERS $BOTNAME $MERGEDSTUDENTNAME
↪$MERGEDINSTRUCNAME)
USERCOUNT=$(echo "$USERLIST" | wc -l)


if [ -n "$USERLIST" ]; then

    echo "\
The following non-administrator accounts are assumed to be either
students from last semester or accounts of random people who once
logged into the wiki. These will be merged into the $MERGEDSTUDENTNAME
account, and anything they ever did on the wiki will be destroyed in a
later step. The merging process will delete the original accounts.
"
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit usernames by line breaks,␣
↪not spaces
    for USER in $USERLIST; do
        REALNAME=$(querywikidb "SELECT user_real_name FROM user WHERE user_name='
↪$USER';")
```

(continues on next page)

(continued from previous page)

```
        echo -e "$USER \t($REALNAME)"
    done
    IFS=$OLDIFS
    echo

    echo "\
Look over the list carefully. Continue only if everything looks right
to you. If an account appears here that you do not want merged, you
may edit this script and add the user name to the IGNOREUSERS global
variable at the top of the file.

Merging accounts can take a long time, so please be patient. You can
press Ctrl+c to abort this script at any time.
"
    continueprompt
    until getedittoken; do
        echo
        retryprompt
    done
    ITER=1
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit usernames by line breaks,␣
↪not spaces
    for USER in $USERLIST; do
        echo -n "[$ITER/$USERCOUNT] "
        until usermerge "$USER" "$MERGEDSTUDENTNAME"; do
            echo
            retryprompt
        done
        let ITER++
    done
    IFS=$OLDIFS
    echo

    echo "\
Merging of former student accounts complete.
"

else

    echo "\
All former student accounts have already been merged and deleted.
"

fi
continueprompt


echo "\
**********************************************************************
**                  STEP 3: MERGE FORMER INSTRUCTORS                **
**********************************************************************
"

# Ensure that the account for merging instructors exists

until userexists "$MERGEDINSTRUCNAME"; do
    echo "\
```

(continues on next page)

(continued from previous page)

```
The account that will be used to merge instructors, $MERGEDINSTRUCNAME,
does not exist on the wiki. This script will create it now.
"
    continueprompt
    until createandpromoteaccount "$MERGEDINSTRUCNAME"; do
        echo
        retryprompt
    done
    echo
done


# Acquire the list of former instructors

USERLIST=$(listsysops $IGNOREUSERS $BOTNAME $MERGEDSTUDENTNAME $MERGEDINSTRUCNAME)
USERCOUNT=$(echo "$USERLIST" | wc -l)


if [ -n "$USERLIST" ]; then

    echo "\
The following administrator accounts are assumed to be TAs from last
semester. These will be merged into the $MERGEDINSTRUCNAME account.
Their contributions to the wiki will be preserved under the merged
account. The merging process will delete the original accounts.
"
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit usernames by line breaks,␣
↪not spaces
    for USER in $USERLIST; do
        REALNAME=$(querywikidb "SELECT user_real_name FROM user WHERE user_name='
↪$USER';")
        echo -e "$USER \t($REALNAME)"
    done
    IFS=$OLDIFS
    echo

    echo "\
Look over the list carefully. Continue only if everything looks right
to you. If an account appears here that you do not want merged, you
may edit this script and add the user name to the IGNOREUSERS global
variable at the top of the file.

Merging accounts can take a long time, so please be patient. You can
press Ctrl+c to abort this script at any time.
"
    continueprompt
    until getedittoken; do
        echo
        retryprompt
    done
    ITER=1
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit usernames by line breaks,␣
↪not spaces
    for USER in $USERLIST; do
        echo -n "[$ITER/$USERCOUNT] "
        until demotesysop "$USER"; do
            echo
```

(continues on next page)

(continued from previous page)

```
            retryprompt
        done
        until usermerge "$USER" "$MERGEDINSTRUCNAME"; do
            echo
            retryprompt
        done
        let ITER++
    done
    IFS=$OLDIFS
    echo

    echo "\
Merging of former instructor accounts complete.
"

else

    echo "\
All former instructor accounts have already been merged and deleted.
"

fi
continueprompt


echo "\
**********************************************************************
**           STEP 4: DELETE LAB NOTEBOOKS AND TERM PAPERS           **
**********************************************************************
"

# List all pages in the User, User talk, Private, and Private talk
# namespaces.

PAGELISTALL=$(listpages $NS_USER $NS_USER_TALK $NS_PRIVATE $NS_PRIVATE_TALK)

# Create a regular expression that matches all pages in the
# IGNOREPAGES list, as well as the User and Private pages (including
# subpages, e.g., User:Foo/Bar) of all users in the IGNOREUSERS list.

REGEXPIGNORE="^$(echo $IGNOREPAGES | tr -s ' ' '|')|((Private:|)User:($(echo
↪$IGNOREUSERS | tr -s ' ' '|'))(/.*|_'.*|))$"

# List the User and Private pages of all users in the IGNOREUSERS
# list, which will not be deleted.

PAGELISTIGN=$(echo "$PAGELISTALL" | grep -E "$REGEXPIGNORE")

# List the remaining pages, which will be deleted, and count them.

PAGELISTDEL=$(echo "$PAGELISTALL" | grep -E "$REGEXPIGNORE" -v)
PAGECOUNT=$(echo "$PAGELISTDEL" | wc -l)


if [ -n "$PAGELISTDEL" ]; then

    echo "\
```

(continues on next page)

(continued from previous page)

```
This step will reversibly delete pages in the following namespaces:
  - User & Private            (lab notebooks and term papers)
  - User talk & Private talk  (comments on student work)

Pages listed in the IGNOREPAGES global variable at the top of this
file will be ignored, as will the User and Private pages of users
listed in the IGNOREUSERS global variable. The following pages will be
ignored:
"
    echo "$PAGELISTIGN"
    echo

    echo "\
The method used here is equivalent to clicking the \"Delete\" link on
each page. The options to view the histories of these pages and
undelete them will still be present on the wiki to instructors. This
content will be permanently deleted in a later step.

The total number of pages that will be deleted is: $PAGECOUNT

Deleting pages can take a long time, so please be patient. You can
press Ctrl+c to abort this script at any time.
"
    continueprompt
    until getedittoken; do
        echo
        retryprompt
    done
    ITER=1
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit page titles by line breaks,␣
↪not spaces
    for PAGE in $PAGELISTDEL; do
        echo -n "[$ITER/$PAGECOUNT]  "
        deletepageorfile "$PAGE"
        let ITER++
    done
    IFS=$OLDIFS
    echo

    echo "\
Deletion of lab notebooks and term papers complete.
"

else

    echo "\
All lab notebooks and term papers have already been deleted.
"

fi
continueprompt


# List all pages in the Talk namespace and count them.

PAGELISTALL=$(listpages $NS_TALK)
PAGECOUNT=$(echo "$PAGELISTALL" | wc -l)
```

(continues on next page)

(continued from previous page)

```
if [ -n "$PAGELISTALL" ]; then

    echo "\
There is a namespace in which students can provide comments that may
be worth reading now:
  - Talk  (comments on course materials)

Since some of these pages may be comment exemplars provided with the
term paper exemplars, and others might be student feedback that you
should look at, they will not be deleted by this script. It is
recommended that you look at each page now and delete it manually if
that is appropriate.

$PAGELISTALL
"
    continueprompt

fi


echo "\
**********************************************************************
**                    STEP 5: DELETE STUDENT FILES                  **
**********************************************************************
"

# List all files not uploaded by instructors and count them

FILELIST=$(listfiles $IGNOREUSERS $MERGEDINSTRUCNAME)
FILECOUNT=$(echo "$FILELIST" | wc -l)


if [ -n "$FILELIST" ]; then

    echo "\
This step will delete all files uploaded by non-instructors.

The method used here is equivalent to clicking the \"Delete\" link on
each file page. The options to view the histories of these files and
undelete them will still be present on the wiki to instructors. Image
thumbnails and resized versions of images are deleted in this step,
freeing up potentially tens of gigabytes of hard drive space. The
original files will be permanently deleted in a later step.

The total number of files that will be deleted is: $FILECOUNT

Deleting files can take a *VERY* long time (~0.6 sec per file = ~6000
files per hour). If you are remotely connected to the server, it is
highly recommended that you use the command line 'screen' utility
while running this portion of the script. This will allow you to
disconnect from the server without interrupting the script. To use it,
quit this script and run the following:

    screen -dRR

You will be placed in a pre-existing screen session if there is one;
```

(continues on next page)

(continued from previous page)

```
otherwise a new session will be created. From there you can run this
script as before. If you want to disconnect from the server while the
script is still running, press Ctrl+a d (the key combination Ctrl+a
followed by the 'd' key alone). This will \"detach\" you from the screen
session, and you will be returned to the normal command line where you
can log out. To return to the screen session later, log into the
server and enter the 'screen -dRR' command again. This works even if
your connection to the server was accidentally interrupted without you
manually logging out.

You can press Ctrl+c to abort this script at any time.
"
    continueprompt

    echo "Current disk usage:"
    echo "$(df -H /)"
    echo

    echo -n "Resetting filesystem permissions... "
    chown -R www-data:www-data /var/www/
    chmod -R ug+rw /var/www/
    echo "done"

    until getedittoken; do
        echo
        retryprompt
    done
    ITER=1
    OLDIFS=$IFS; IFS=$'\n' # tell for-loop to delimit file titles by line breaks,␣
↪not spaces
    for FILE in $FILELIST; do
        if [ $(($ITER % 200)) -eq 0 ]; then
            # fetch a new edit token periodically so it does not expire
            until getedittoken; do
                echo
                retryprompt
            done
        fi
        echo -n "[$ITER/$FILECOUNT] "
        deletepageorfile "$FILE"
        let ITER++
    done
    IFS=$OLDIFS

    echo -n "Resetting filesystem permissions... "
    chown -R www-data:www-data /var/www/
    chmod -R ug+rw /var/www/
    echo "done"
    echo

    echo "Current disk usage:"
    echo "$(df -H /)"
    echo

    echo "\
Deletion of student files complete.
"
```

(continues on next page)

(continued from previous page)

```
else

    echo "\
All student files have already been deleted.
"

fi
continueprompt


echo "\
**********************************************************************
**             STEP 6: PERMANENTLY DELETE STUDENT CONTENT            **
**********************************************************************

When a page or file is deleted on the wiki, it becomes inaccessible to
normal users, and a pink box appears on the deleted page stating that
the page was deleted. However, it is more accurate to say that the
item was archived, since administrators can still review the revision
history or restore the item.

In this step, the pages and files deleted in prior steps will be
permanently deleted, removing them and their revision histories
completely from the wiki, and freeing up potentially gigabytes of hard
drive space.

Furthermore, all entries in the \"Recent changes\" and Special:Log lists
that pertain to the deleted items will be removed, such as the
thousands of edits made by students to their pages and instructors
marking student comments as patrolled.

These actions will apply not only to items deleted by this script, but
also to items deleted manually.

NOTE: Warnings may appear stating that files are \"not found in group
'deleted'\". These should be ignored.
"
continueprompt


echo "Current disk usage:"
echo "$(df -H /)"
echo

echo -n "Resetting filesystem permissions... "
chown -R www-data:www-data /var/www/
chmod -R ug+rw /var/www/
echo "done"

php /var/www/mediawiki/maintenance/deleteArchivedRevisions.php --delete
echo
php /var/www/mediawiki/maintenance/deleteArchivedFiles.php --delete --force
echo

echo -n "Resetting filesystem permissions... "
chown -R www-data:www-data /var/www/
```

(continues on next page)

(continued from previous page)

```
chmod -R ug+rw /var/www/
echo "done"

apache2ctl restart

echo "Deleting relevant log entries..."
echo

# Remove records about deleting pages, files, and users. Necessary to
# remove the pink "This page has been deleted" boxes.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'delete';"
querywikidb "DELETE FROM logging WHERE log_type = 'delete';"

# Remove records about student edits and uploads.

querywikidb "DELETE FROM recentchanges WHERE rc_user = (SELECT user_id FROM user␣
↪WHERE user_name = '$MERGEDSTUDENTNAME');"
querywikidb "DELETE FROM logging WHERE log_user = (SELECT user_id FROM user WHERE␣
↪user_name = '$MERGEDSTUDENTNAME');"

# Remove records about patrolling student comments on term paper benchmarks.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'patrol';"
querywikidb "DELETE FROM logging WHERE log_type = 'patrol';"

echo "Current disk usage:"
echo "$(df -H /)"
echo


echo "\
Permanent deletion of archived pages and files complete.
"
continueprompt


echo "\
*********************************************************************
**                       STEP 7: DELETE GRADES                     **
*********************************************************************

This step will delete the scores that former students received on
assignments. The assignments themselves will not be changed. Entries
in Special:Log/grades will be deleted as well.
"
continueprompt


# Remove grades for former students.

querywikidb "TRUNCATE TABLE scholasticgrading_adjustment;"
querywikidb "TRUNCATE TABLE scholasticgrading_evaluation;"
querywikidb "TRUNCATE TABLE scholasticgrading_groupuser;"

# Remove records about assigning grades.
```

(continues on next page)

(continued from previous page)

```
querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'grades';"
querywikidb "DELETE FROM logging WHERE log_type = 'grades';"


echo "\
Deletion of grades complete.
"
continueprompt


echo "\
**********************************************************************
**                      STEP 8: WIKI LOG CLEANUP                    **
**********************************************************************

This step will delete all remaining entries in the \"Recent changes\"
and Special:Log lists that are remnants of the last semester or this
script.
"
continueprompt


# Remove records about actions taken by the bot.

querywikidb "DELETE FROM recentchanges WHERE rc_user = (SELECT user_id FROM user␣
↪WHERE user_name = '$BOTNAME');"
querywikidb "DELETE FROM logging WHERE log_user = (SELECT user_id FROM user WHERE␣
↪user_name = '$BOTNAME');"


# Remove records about manually merging user accounts.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'usermerge';"
querywikidb "DELETE FROM logging WHERE log_type = 'usermerge';"


# Remove records about manually creating user accounts.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'newusers';"
querywikidb "DELETE FROM logging WHERE log_type = 'newusers';"


# Remove records about renaming user accounts.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'renameuser';"
querywikidb "DELETE FROM logging WHERE log_type = 'renameuser';"


# Remove records about adjusting group membership (e.g., term paper authors).

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'rights';"
querywikidb "DELETE FROM logging WHERE log_type = 'rights';"


# Remove records about moving/renaming pages and files.

querywikidb "DELETE FROM recentchanges WHERE rc_log_type = 'move';"
```

(continues on next page)

(continued from previous page)

```
querywikidb "DELETE FROM logging WHERE log_type = 'move';"


# Remove watchlist entries for merged accounts.

querywikidb "DELETE FROM watchlist WHERE wl_user = (SELECT user_id FROM user␣
↪WHERE user_name = '$MERGEDSTUDENTNAME');"
querywikidb "DELETE FROM watchlist WHERE wl_user = (SELECT user_id FROM user␣
↪WHERE user_name = '$MERGEDINSTRUCNAME');"


# Remove watchlist entries for pages and files that no longer exist.

querywikidb "DELETE wl.* FROM watchlist AS wl LEFT JOIN page AS p ON (p.page_
↪namespace = wl.wl_namespace AND p.page_title = wl.wl_title) WHERE p.page_id IS␣
↪NULL;"


echo "\
Deletion of old log entries complete.
"
continueprompt


echo "\
**********************************************************************
**                 STEP 9: DELETE SURVEY RESPONSES                  **
**********************************************************************
"

# Create a regular expression that matches the user names in the
# IGNOREUSERS list

REGEXPIGNORE="^($(echo $IGNOREUSERS | tr -s ' ' '|'))$"


echo "\
This step will delete all responses from students to surveys in the
Django system and close all open surveys. All Django accounts will
also be deleted, except those listed in the IGNOREUSERS global
variable at the top of this file. The accounts that will be ignored
are:
"
querydjangodb "SELECT CONCAT(username, ' (', first_name, ' ', last_name, ')')␣
↪FROM auth_user WHERE username REGEXP '$REGEXPIGNORE';"
echo
continueprompt


querydjangodb "TRUNCATE django_admin_log;"
querydjangodb "TRUNCATE django_session;"

querydjangodb "TRUNCATE survey_choiceanswer;"
querydjangodb "TRUNCATE survey_ratinganswer;"
querydjangodb "TRUNCATE survey_textanswer;"
querydjangodb "TRUNCATE survey_surveycredit;"
```

(continues on next page)

(continued from previous page)

```
querydjangodb "TRUNCATE credit_team_members;"
querydjangodb "DELETE FROM credit_team; ALTER TABLE credit_team AUTO_INCREMENT =␣
↪1;" # cannot be truncated because of a foreign key constraint
querydjangodb "DELETE FROM auth_user WHERE username NOT REGEXP '$REGEXPIGNORE';"


# Close all survey sessions.

querydjangodb "UPDATE survey_surveysession SET open=0;"


echo "\
Deletion of survey responses complete.
"
continueprompt


echo "\
**********************************************************************
**                              FINISHED                            **
**********************************************************************

This script is now finished. You should check that it has done its job
by visiting these pages:

    - Special:ListUsers

      The only accounts that should be listed are the
      $BOTNAME, $MERGEDSTUDENTNAME, and $MERGEDINSTRUCNAME accounts,
      as well as those accounts that were listed in the IGNOREUSERS
      variable.

    - Special:AllPages

      Check that the Main, Talk, User, User talk, Private, Private
      talk, and File namespaces are all absent of student content.

    - Special:ListFiles

      Check that this list is absent of student content.

    - Special:Log

      You should see only instructor actions listed on the front page.
      Using the drop-down menu, you can view specific logs, such as
      the grades log. Most of these should be empty, but a few will
      list instructor actions. This is intended.

    - Special:Log/$BOTNAME
    - Special:Log/$MERGEDSTUDENTNAME
    - Special:ListFiles/$MERGEDSTUDENTNAME
    - Special:Contributions/$MERGEDSTUDENTNAME
    - Special:DeletedContributions/$MERGEDSTUDENTNAME

      These lists should be empty.

    - Special:Grades
```

(continues on next page)

(continued from previous page)

```
        There should not be any student grades listed.

      - Django admin page:
        https://$(hostname).case.edu/django/admin

        The list of users should contain only those accounts listed in
        the IGNOREUSERS variable. There should be no teams. All survey
        sessions should be closed.
"
continueprompt
```

9. **Todo:** Add instructions for updating ignored users in the reset-wiki script and for first saving exemplars.

10. Start a `screen` session:

    ```
    screen -dRR
    ```

    The screen session will allow you to disconnect from the server without interrupting the script as it runs.

11. Run the script and follow the step-by-step instructions:

    ```
    sudo reset-wiki
    ```

    If this is the first time the script is run, three new wiki accounts will be created. You will be asked to choose passwords for each. It is fine to use the same password for all three. The password for the first account (the bot) must match the password stored in the script, which you specified in step 8. The passwords for the other two accounts will never be needed after they are created.

    Running this script can take a long time (hours). If you need to disconnect from the server while the script is running, press `Ctrl-a d` (that's the key combination `Ctrl-a` followed by the `d` key alone) to detach from the screen session. You can then log out of the server. To return to the screen session later, just run `screen -dRR` again after logging in.

12. Once the wiki has been successfully reset, shut down the virtual machine:

    ```
    sudo shutdown -h now
    ```

13. Using VirtualBox, take a snapshot of the current state of the virtual machine. Name it "**Former students' wiki content deleted**".

14. Delete the first snapshot, created in step 4, to save disk space.

15. Restart the virtual machine and log in:

    ```
    ssh hjc@neurowiki.case.edu
    ```

16. Unlock the wiki so that students can make edits if it is still locked from the end of the last semester (running the command will tell you whether it is locked or unlocked):

    ```
    sudo lock-wiki
    ```

17. **Todo:** Need to add instructions for updating miscellaneous wiki pages, syllabus dates, assignment dates, survey session dates after resetting the wiki.

CHAPTER 12

Configuring the Development Server

1. If biol300_2017 is running, log in and shut it down:

```
ssh hjc@biol300.case.edu
sudo shutdown -h now
```

2. If biol300dev_2016 is running, shut it down as well:

```
ssh hjc@biol300dev.case.edu
sudo shutdown -h now
```

3. Using VirtualBox, take a snapshot of the current state of biol300_2017. Name it “**Last shared state between biol300 and biol300dev**”.

4. In VirtualBox, create a clone of biol300_2017 by selecting the virtual machine and Machine > Clone, and choosing the following settings:

   - **New machine name**
     - Name: biol300dev_2017
     - Do NOT check “Reinitialize the MAC address of all network cards”
   - **Clone type**
     - Full clone
   - **Snapshots**
     - Current machine state

   Drag-and-drop the new virtual machine into the “2017” group under “BIOL 300”.

5. In VirtualBox, select biol300dev_2017 and choose Settings > Network. Verify that the “Attached to” setting is set to “Bridged Adapter”. Under Advanced, change the MAC address to match BIOL300Dev’s (see *Looking Up MAC Addresses* for instructions).

6. Restart biol300_2017. Start biol300dev_2017 for the first time and log in:

```
ssh-keygen -R biol300dev.case.edu
ssh hjc@biol300dev.case.edu
```

7. Change the name that the development server uses to identify itself to the network:

```
sudo hostnamectl set-hostname biol300dev
```

8. You should now be able to access it in a browser:

    https://biol300dev.case.edu

9. **Todo:** Make the backup script that lives on DynamicsHJC and the cron job list downloadable.

    Log into the vbox account on DynamicsHJC:

```
ssh vbox@dynamicshjc.case.edu
```

Create a script that remotely executes the backup script on the development virtual machine and moves the archive to the backup drive. Create the file

```
mkdir -p /Volumes/CHIELWIKI/backups/biol300dev/2017
vim /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh
```

and fill it with the following:

```
#!/bin/bash

REMOTESSH="ssh hjc@biol300dev.case.edu"
#REMOTESSH="ssh -p 8015 hjc@dynamicshjc.case.edu"

REMOTESCP="scp -q hjc@biol300dev.case.edu"
#REMOTESCP="scp -q -P 8015 hjc@dynamicshjc.case.edu"

REMOTEFILE=biol300dev-backup-`date +'%Y%m%dT%H%M%S%z'`.tar.bz2
if [ -z "$1" ]; then
    LOCALFILE=/Volumes/CHIELWIKI/backups/biol300dev/2017/$REMOTEFILE
else
    LOCALFILE=/Volumes/CHIELWIKI/backups/biol300dev/2017/$1
fi

$REMOTESSH backup-wiki -q backup $REMOTEFILE
$REMOTESCP:$REMOTEFILE $LOCALFILE
chmod go= $LOCALFILE
$REMOTESSH rm $REMOTEFILE
```

10. Make the script executable:

```
chmod u+x /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh
```

11. In the vbox account on DynamicsHJC, create a backup schedule. Create the file

```
vim /Volumes/CHIELWIKI/backups/biol300dev/2017/crontab
```

and fill it with the following:

```
################################################################################
#                                  BIOL300DEV                                  #
################################################################################
# Make hourly backups on the odd-numbered hours (except 1 AM and 3 AM)
0 5,9,13,17,21  * * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-
↪biol300dev.sh hourA.tar.bz2
0 7,11,15,19,23 * * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-
↪biol300dev.sh hourB.tar.bz2
# Make daily backups at 1 AM
0 1 * * 0 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh sunday.
↪tar.bz2
0 1 * * 1 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh monday.
↪tar.bz2
0 1 * * 2 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh tuesday.
↪tar.bz2
0 1 * * 3 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh␣
↪wednesday.tar.bz2
0 1 * * 4 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh␣
↪thursday.tar.bz2
0 1 * * 5 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh friday.
↪tar.bz2
0 1 * * 6 /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh␣
↪saturday.tar.bz2
# Make weekly backups at 3 AM on the 1st, 8th, 15th, and 22nd of the month
0 3 1  * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh `date +
↪'\%Y-\%m'`.tar.bz2
0 3 8  * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh 8th.
↪tar.bz2
0 3 15 * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh 15th.
↪tar.bz2
0 3 22 * * /Volumes/CHIELWIKI/backups/biol300dev/2017/backup-biol300dev.sh 22nd.
↪tar.bz2
```

12. Schedule the backups. In the vbox account on DynamicsHJC, dump the existing scheduled jobs to a temporary file:

    ```
    crontab -l > /tmp/crontab.old
    ```

    Edit the temporary file, and delete the backup jobs for last year's BIOL300Dev. You can use `Shift+v` in Vim to enter Visual Line mode, the up and down arrow keys to select a block of lines, and `d` to delete them all at once.

    ```
    vim /tmp/crontab.old
    ```

    Now append the new jobs to the old and schedule them:

    ```
    cat {/tmp/crontab.old,/Volumes/CHIELWIKI/backups/biol300dev/2017/crontab} |␣
    ↪crontab
    ```

    Verify that the backup jobs for this year's BIOL300Dev are properly scheduled, and that backup jobs for last year's BIOL300Dev are absent:

    ```
    crontab -l
    ```

13. Fix SSH authentication into BIOL300Dev from the vbox account. You will be asked to accept the unrecognized fingerprint of the virtual machine — this is expected — but you should NOT need to enter your password. The IP address is the static IP for biol300dev.case.edu, obtained using `host biol300dev.case.edu`.

```
ssh-keygen -R biol300dev.case.edu -R 129.22.139.48
ssh hjc@biol300dev.case.edu
```

If this works without you needing to enter a password, automatic authentication is properly configured. You should `logout` to return to the vbox account on DynamicsHJC, and `logout` again to return to the virtual machine.

14. Shut down the development virtual machine:

```
sudo shutdown -h now
```

15. Using VirtualBox, take a snapshot of the current state of biol300dev_2017. Name it "**biol300dev configured**".

16. Restart biol300dev_2017.

CHAPTER 13

Using a Backup Archive to Recover from Loss of the Virtual Machine

These instructions describe how to recover the wiki if the virtual machine is lost but backups are retained. The instructions are untested and possibly incomplete. Additional breathing may be required. The wiki backups should contain everything needed to run the server except for the Apache configuration settings, which are contained in this document.

1. Breathe.
2. *Create a replica virtual machine*.
3. *Configure the web server*.
4. Install the prerequisite packages for *MediaWiki*, the *MediaWiki extensions*, and *Django*.
5. Install the web server (Apache) configuration files for *MediaWiki* and *Django*.
6. Restore the machine from an archived backup. The restore script `backup-wiki` is included in the archive.
7. *Activate bridged networking*. The wiki should be accessible now.
8. Go through the instruction set for *building the wiki from scratch* carefully and complete any minor steps that have been skipped, such as placing scripts in `/usr/local/sbin` (e.g., one is needed for automatic backups to work).
9. Verify that automatic backups are working.

CHAPTER 14

Creating Virtual Machine Accounts

These instructions will guide you through all the steps needed to create a new account for someone (e.g., a TA) on the virtual machine. Throughout these instructions, you should replace the placeholder `<name>` with the actual name of the new account.

Warning: The created account will have unlimited administrative privileges on the virtual machine. Be careful who you give an account to.

1. An existing sysadmin must run the following commands to create the account. The first command creates the new account and adds it to the sudo (sysadmin) group. The second command sets an initial password for the account. Be prepared to share this with the owner of the new account.

   ```
   sudo useradd -m -s /bin/bash -G sudo <name>
   sudo passwd <name>
   ```

Note: The remaining steps should be performed by the account owner.

2. To connect to the virtual machine remotely, you must have access to a command line interface on your personal machine. If you own a Mac or Linux computer, Terminal should already be installed. If you own a Windows computer, you should install the Windows Subsystem for Linux (Windows 10 only) or Cygwin.

3. At the command line on your personal machine, log into the virtual machine using one of the following pairs of commands:

   ```
   ssh-keygen -R [dynamicshjc.case.edu]:8015
   ssh -p 8015 <name>@dynamicshjc.case.edu
   ```

   if the virtual machine is still under construction and is connected to the network using port forwarding, or

   ```
   ssh-keygen -R biol300.case.edu
   ssh <name>@biol300.case.edu
   ```

if the virtual machine is publicly visible and using bridged netorking.

First, you will be asked to accept the unrecognized fingerprint of the virtual machine; enter "yes". Second, you will need to enter your password. Finally, you should be greeted by a welcome message, and the command prompt should change to `<name>@biol300:~$`.

If you would like to change your password, you may do so now using the following command:

```
passwd
```

4. On the virtual machine, add yourself to the www-data (Apache web server) group to make accessing files in the web directory easier:

```
sudo usermod -a -G www-data <name>
```

You must `logout` and back in for this change to take effect (you do not need to do so now).

Note: The following steps are optional.

5. If you would like to be able to log into the virtual machine without needing to enter your password, do the following.

First, logout of the virtual machine to return to your personal computer:

```
logout
```

Second, generate an SSH key on your personal computer if you do not have one already. If you do, this command will recognize that and do nothing. Otherwise, press Enter three times when asked about the file location and passphrase:

```
[ -e ~/.ssh/id_rsa ] || ssh-keygen
```

Third, copy your public SSH key to the virtual machine using one of the following commands (you will need to enter your password for the virtual machine once or twice).

If the virtual machine is still under construction and is connected to the network using port forwarding, and if `ssh-copy-id` is available on your personal computer, use

```
ssh-copy-id -p 8015 <name>@dynamicshjc.case.edu
```

If `ssh-copy-id` is unavailable, use

```
scp -P 8015 ~/.ssh/id_rsa.pub <name>@dynamicshjc.case.edu:
ssh -p 8015 <name>@dynamicshjc.case.edu "mkdir -p .ssh && cat id_rsa.pub >> .ssh/
↪authorized_keys && rm id_rsa.pub"
```

If the virtual machine is publicly visible and using bridged netorking, and if `ssh-copy-id` is available on your personal computer, use

```
ssh-copy-id <name>@biol300.case.edu
```

If `ssh-copy-id` is unavailable, use

```
scp ~/.ssh/id_rsa.pub <name>@biol300.case.edu:
ssh <name>@biol300.case.edu "mkdir -p .ssh && cat id_rsa.pub >> .ssh/authorized_
↪keys && rm id_rsa.pub"
```

You should now be able to log into the virtual machine without entering your password. Do so now using using one of the following commands:

```
ssh -p 8015 <name>@dynamicshjc.case.edu
```

if the virtual machine is still under construction and is connected to the network using port forwarding, or

```
ssh <name>@biol300.case.edu
```

if the virtual machine is publicly visible and using bridged netorking.

6. Create a Vim configuration file on the virtual machine by following the directions in *Vim Configuration File*.

CHAPTER 15

SSL Certificate Management

For an introduction to SSL certificates, see *HTTPS & SSL*.

A collection of all our SSL certificates is maintained in a secure directory on DynamicsHJC: `/Users/vbox/ssl-certificates`. Since Fall 2016, this directory is shared with the virtual machines as a virtual file system mounted at `/media/sf_ssl-certificates`. Virtual machines created since that time serve their SSL certificates directly from this central collection. (Older virtual machines have local copies of the certificates.) This makes certificate renewal easier, as updating the certificate in one place will take care of expiring certificates on all clones of that virtual machine.

15.1 Checking Expiration Dates

SSL certificates have limited lifespans. Certificates provided by the university usually last 3 years. You can check the expiration date of a certificate on a live website using your browser (click the lock icon in the address bar). You can also check the expiration dates for all the certificates in the aforementioned collection using this script on DynamicsHJC:

```
sudo ~vbox/ssl-certificates/check-ssl-cert-expiration
```

15.2 Creating a New Certificate

Note: These are probably not the instructions you are looking for! In general, you should not need to create SSL certificates from scratch. If you received a warning in an email about an expiring certificate, you should instead consult *Renewing an Expiring Certificate*.

Normally, the only time you would use the following process for creating a certificate from scratch would be if you need a certificate for a machine with a new name (e.g., because you want to start hosting a wiki at `awesomewiki.case.edu`). You might also be asked by the university's certificate administrators to create a new certificate, rather than renew an existing one, if there is a problem with the old one (e.g., if the old certificate was created with the deprecated SHA1 encryption algorithm).

Below is a list of steps showing how one would create a certificate from scratch. Throughout these instructions, you should replace `<host>` with the name of the machine for which you are creating the certificate. The `<CSR password>` is something you need to make up in step 3, and which will be needed in step 6. You can discard it after that as it will not be needed again.

1. Log into the VirtualBox Machines (vbox) account on DynamicsHJC:

   ```
   ssh vbox@dynamicshjc.case.edu
   ```

2. Create a new directory for the new certificate:

   ```
   mkdir -p ~/ssl-certificates/<host>/new
   ```

3. Generate a private key and certificate signing request (CSR) pair for the new certificate:

   ```
   openssl req -sha512 -new -newkey rsa:2048 -nodes -keyout ~/ssl-certificates/<host>
   ↪/new/<host>_case_edu.key -out ~/ssl-certificates/<host>/new/<host>_case_edu.csr
   ```

   Enter the following details when prompted:

   - Country Name: US
   - State or Province Name: Ohio
   - Locality Name: Cleveland
   - Organization Name: Case Western Reserve University
   - Organizational Unit Name: Biology
   - Common Name: <host>.case.edu
   - Email Address: hjc@case.edu
   - A challenge password: <CSR password>
   - An optional company name: .

4. Protect the new files:

   ```
   find ~/ssl-certificates/*/ -type f -exec chmod u=rw,go= {} \;
   ```

5. Dump the contents of the CSR to the screen and copy all of it to the clipboard:

   ```
   cat ~/ssl-certificates/<host>/new/<host>_case_edu.csr
   ```

6. Visit the following web site,

   https://cert-manager.com/customer/InCommon/ssl?action=enroll

   and supply the following on the front page:

   - Access Code: <access code> (see email dated 12/27/2015 from certificate-admin@case.edu with subject line "Fwd: WARNING: Expiration Notice: InCommon SSL certificate for dynamicshjc.case.edu")
   - E-mail: hjc@case.edu

   Fill in the remaining fields:

   - Don't edit address details
   - Certificate Type: InCommon SSL (SHA-2)
   - Certificate Term: 3 years

- Server Software: Apache/ModSSL
- CSR: <paste contents of CSR>
- Common Name: <host>.case.edu
- Pass-phrase: <CSR password>
- External Requester: hjc@case.edu

Press Enroll.

7. After an evaluation period, an email with a subject line beginning "Enrollment Successful" will arrive for Dr. Chiel containing links for downloading files in various formats. He can forward the email to you.

   Using Safari in the VirtualBox Machines (vbox) account on DynamicsHJC, log into your email and download the following using the links provided in the email:

   - X509 Certificate only, Base64 encoded
   - X509 Intermediates/root only, Base64 encoded

   The first file, which will end in `_cert.cer`, is the new certificate belonging to your site. The second file, which will end in `_interm.cer`, contains a chain of certificate authorities that the server will use to corroborate its claim that its certificate isn't fake.

8. Move these two files into `~ssl-certificates/<host>/new` on DynamicsHJC with the matching private key and CSR files. If you downloaded the files while sitting in front of DynamicsHJC or using screen sharing, this can be done simply by dragging-and-dropping. If instead you downloaded the files onto your local machine, you can use `scp` to copy the files:

```
scp *.cer vbox@dynamicshjc.case.edu:ssl-certificates/<host>/new
```

9. Once again, secure the files:

```
find ~/ssl-certificates/*/ -type f -exec chmod u=rw,go= {} \;
```

10. You can now move the new files into the location where the virtual machines will expect to find them (and retire the old certificate if one exists):

```
mkdir -p ~/ssl-certificates/<host>/old
rm -f ~/ssl-certificates/<host>/old/*
mv ~/ssl-certificates/<host>/*.{cer,key,csr} ~/ssl-certificates/<host>/old
mv ~/ssl-certificates/<host>/new/*.{cer,key,csr} ~/ssl-certificates/<host>
```

    Double check that you did this right, and then remove the `new` directory:

```
rm -rf ~/ssl-certificates/<host>/new
```

11. If any of the currently running virtual machines were using the retired certificate (if it even existed), you will need to log into them and restart their web servers before they will begin using the new certificate. Use the following to restart the web server:

```
sudo apache2ctl restart
```

    If the new certificate was for DynamicsHJC, you should also restart its web server. The command is a little different on the Mac:

```
sudo apachectl restart
```

Check that the web server is now using the new certificate by visiting the site in a browser and clicking the lock icon that appears next to the address bar. You should be able to view the certificate details. Verify that the expiration date is far in the future.

15.3 Renewing an Expiring Certificate

When an SSL certificate is going to expire soon, you will receive an expiration notice in an email. Follow these steps to renew it. Throughout these instructions, you should replace `<host>` with the name of the machine for which you are renewing the certificate.

1. Forward the email notice to Tareq Alrashid at certificate-admin@case.edu and ask that the certificate be renewed for another 3 years.
2. After an evaluation period, an email with a subject line beginning "Enrollment Successful" will arrive for Dr. Chiel containing links for downloading files in various formats. He can forward the email to you.

   Using Safari in the VirtualBox Machines (vbox) account on DynamicsHJC, log into your email and download the following using the links provided in the email:

   - X509 Certificate only, Base64 encoded
   - X509 Intermediates/root only, Base64 encoded

   The first file, which will end in `_cert.cer`, is the renewed certificate belonging to your site. The second file, which will end in `_interm.cer`, contains a chain of certificate authorities that the server will use to corroborate its claim that its certificate isn't fake.
3. Retire the old certificate and certificate chain files:

```
mkdir -p ~/ssl-certificates/<host>/old
rm -f ~/ssl-certificates/<host>/old/*
mv ~/ssl-certificates/<host>/*.cer ~/ssl-certificates/<host>/old
```

4. Move the two files downloaded in step 2 into `~ssl-certificates/<host>` on DynamicsHJC. The matching private key and CSR files should already be in there, with extensions `.key` and `.csr` respectively (if the latter is missing, that's OK). If you downloaded the files while sitting in front of DynamicsHJC or using screen sharing, this can be done simply by dragging-and-dropping. If instead you downloaded the files onto your local machine, you can use `scp` to copy the files:

```
scp *.cer vbox@dynamicshjc.case.edu:ssl-certificates/<host>
```

5. Protect the new files:

```
find ~/ssl-certificates/*/ -type f -exec chmod u=rw,go= {} \;
```

6. If any of the currently running virtual machines were using the retired certificate, you will need to log into them and restart their web servers before they will begin using the new certificate. Use the following to restart the web server on the virtual machines:

```
sudo apache2ctl restart
```

   If the renewed certificate was for DynamicsHJC, you should also restart its web server. The command is a little different on the Mac:

```
sudo apachectl restart
```

Check that the web server is now using the new certificate by visiting the site in a browser and clicking the lock icon that appears next to the address bar. You should be able to view the certificate details. Verify that the expiration date is far in the future.

CHAPTER 16

MAC Addresses

See *Port Forwarding & Bridged Networking* for additional background.

16.1 Looking Up MAC Addresses

Each of our virtual machines has associated with it a random string of numbers called its MAC address. Because of the risk of MAC spoofing, these number are secured on DynamicsHJC.

If you are logged into the VirtualBox Machines (vbox) account on DynamicsHJC, you view the file using

```
cat ~/mac-addresses.txt
```

or, if you are sitting in front of the monitor, opening the file through Finder. It is stored in the home directory.

If you are not logged into the vbox account, you can obtain the list through SSH:

```
ssh vbox@dynamicshjc.case.edu "cat ~/mac-addresses.txt"
```

16.2 Registering New MAC Addresses

Additional machine names with new MAC addresses can be registered with the university network by following these steps:

1. Generate a new random MAC address by selecting the virtual machine in VirtualBox, choosing Settings > Network > Advanced, and clicking the blue arrows. Write this number down and press OK. If the virtual machine is configured to use bridged networking ("Attached to: Bridged Adapter"), it will not be able to connect to the CWRU wired network until the remaining steps are completed.
2. Ask Dr. Chiel to visit (using a different machine) UTech Self-Service Tools, choose Student Device Registration, and enter his CWRU credentials. He should then submit the machine name of the virtual machine (chosen during OS installation and retrievable by typing `hostname` on the command line) and the new MAC address.

If Dr. Chiel is unavailable, you can start this process yourself and later submit a request to help@case.edu to transfer ownership of the registered machine to him.

Todo: Verify that Student Device Registration works for non-students.

3. Registration may take several hours to complete. Once done, internet access will be restored to the virtual machine if it is configured for bridged networking. If it is using port forwarding (“Attached to: NAT”), there should have been no interruption of internet access.
4. Add the new hostname and MAC address to the file `/Users/vbox/mac-addresses.txt` on DynamicsHJC.

CHAPTER 17

University Firewall

In July 2016, Case instituted a new computer network security policy, called Default Protect All, that affects how on-campus computers are accessed from off campus. A firewall was activated that prevents connections to computers on the university's network (computers accessed with the case.edu domain name) from computers that aren't connected to the wired network, CaseGuest, CaseWireless, or VPN.

This affects web servers, remote shell connections (SSH), and remote desktop sharing (VNC). If you need remote access to a lab computer from somewhere off-campus, you should always be able to obtain it if you first connect to VPN. However, to host public websites like the wikis, it is necessary to obtain firewall exceptions for specific ports on specific host machines. This is done by contacting the Help Desk (help@case.edu or 216-368-4357). Exceptions only need to be granted once, and this has already been done for the class wikis and many other machines in our lab. A complete list of the exceptions obtained for our lab computers can be found here.

In case this system changes or new exceptions are needed in the future, here I provide background information and procedures for dealing with the university firewall.

17.1 Technical Background

When a computer attempts to connect to the university network for the first time (e.g., after rebooting), it must contact a central server, called the Dynamic Host Configuration Protocol (DHCP) server, and request a numerical IP address, which looks something like 129.22.139.42 (it seems that all IP addresses in our building begin with 129.22.139.*). The IP address is needed so that other computers know where to find it.

Unless a permanent, "static" IP address is specifically requested, the DHCP server will temporarily lease a "dynamic" IP address to the machine. This is useful to the network managers since there are a limited number of available addresses, and devices (e.g., mobile phones) connect to and disconnect from the network all the time. If a permanent IP was assigned to a device every time it connected to the network, the university would quickly run out of addresses.

Leases on IP addresses have expiration dates. A computer that has been leased an IP address must check in with the central server after a certain period of time (roughly 10 hours) to maintain its connection to the network. If the computer was leased a dynamic IP address, the lease on the original IP address will usually just be renewed; however, it is possible that a different IP address will be leased (hence "dynamic").

Changes in IP address are usually invisible to end users since they don't need to access computers using IP addresses directly. Normally, an end user knows the unchanging "fully qualified domain name" (FQDN) of the computer they want to connect to, such as neurowiki.case.edu. In this example, "neurowiki" is the "host name" for a computer on the university network, and "case.edu" is the "domain name" for the university network. Whenever an end users requests to access a computer using its FQDN, another server, called a Domain Name System (DNS) server, translates the FQDN to its corresponding IP address. The database that the DNS server reads to perform these translations needs to be updated whenever a dynamically allocated IP address changes.

Case's Default Protect All policy places all computers on the university network behind a firewall that prohibits access to these computers from all but trusted nodes, namely other computers on the university network or on VPN. This makes it much harder for a hacker to gain access to university computers from off-campus. However, it also means that web and data servers cannot be accessed by the public or off-campus colleagues. Case allows for exceptions to the policy to be requested for specific ports on specific computers for this reason.

Apparently, the way the firewall is implemented requires that the computers granted exceptions have static IP addresses. A static IP address can be requested for a particular computer through a form on UTech's website. After the request is granted, the computer will be reliably leased the same IP address.

Since the assignment of a static IP address to a computer reduces the finite pool of available IP addresses for dynamic assignment, reclaiming static IP addresses when they are no longer needed is a priority for the network managers. Consequently, requests for static IP addresses must be renewed each year, or they will be returned to the pool.

17.2 Transitioning to a Static IP Address

Some time after a request for a static IP address is granted (roughly 20 minutes, supposedly), the DNS database is updated so that the FQDN (e.g., neurowiki.case.edu) points to the new static IP address. However, until the hours-long lease on the original, dynamic IP address currently in use by the computer ends, the computer will not update its address. After that time, it will check in with the DHCP server and be assigned its new, static IP address. However, in this interval, the server will be inaccessible because the DNS server has moved its pointer before the server has moved to its new address.

Instead of waiting for the old lease to expire, you can force a computer to request a new one at any time. If you've requested a static IP address recently and notice that you can no longer access your computer, you should first verify that the situation described above is the problem.

First, check at which address the DNS server says your computer should be located. In the Terminal (command line) on any Mac or Ubuntu machine, enter:

```
host <name>
```

where `<name>` is the FQDN of the inaccessible computer (e.g., neurowiki.case.edu). The result will look something like this:

```
neurowiki.case.edu is an alias for neurowiki.BIOL.CWRU.edu.
neurowiki.BIOL.CWRU.edu has address 129.22.139.42
```

If the static IP request has taken effect, this is it.

Second, check the current IP address lease on the inaccessible machine. At the Terminal for that machine, enter (on either Mac or Ubuntu):

```
ifconfig -a
```

The result will look like this:

```
eth0      Link encap:Ethernet  HWaddr 08:00:27:44:bb:5a
          inet addr:129.22.139.71  Bcast:129.22.139.127  Mask:255.255.255.128
          inet6 addr: fe80::a00:27ff:fe44:bb5a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21998180 errors:0 dropped:0 overruns:0 frame:0
          TX packets:61298281 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2125183715 (2.1 GB)  TX bytes:161301856038 (161.3 GB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:5751 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5751 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:857989 (857.9 KB)  TX bytes:857989 (857.9 KB)
```

Here, 129.22.139.71 is the current, dynamically allocated IP address leased to the computer. It is in use by the active network interface labeled “eth0” (Ethernet socket 0). For the next step, you must correctly identify the active network interface. On other machines, this may be labeled differently, or there may be more than one interface, only one of which is active. For example, on DynamicsPJT, the result of the command is this:

```
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet 127.0.0.1 netmask 0xff000000
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 00:25:00:ed:8e:c2
        media: autoselect (<unknown type>)
        status: inactive
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 00:23:df:e0:2f:e8
        inet6 fe80::223:dfff:fee0:2fe8%en1 prefixlen 64 scopeid 0x5
        inet 129.22.139.43 netmask 0xffffff80 broadcast 129.22.139.127
        media: autoselect (1000baseT <full-duplex,flow-control>)
        status: active
fw0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 4078
        lladdr 00:23:df:ff:fe:dc:f5:ac
        media: autoselect <full-duplex>
        status: inactive
```

This machine has two Ethernet sockets, labeled “en0” and “en1”. The interface “en1” is identifiable as the active interface because it has an IP address (129.22.139.43); in this case, it also says “status: active”. The reason this interface is active is because whoever originally set up that computer happened to plug the Ethernet cable into one socket and not the other.

Finally, if you’ve verified that there is a mismatch between the IP address in the DNS database and the currently leased IP address, you can resolve the situation by having the inaccessible computer request a new lease. This should update it to the new static IP address. To do this, you can restart the computer or use another command, which depends on the operating system:

- Mac:

  ```
  sudo ipconfig set <interface> DHCP
  ```

- Ubuntu:

```
sudo dhclient -v -r <interface> && sudo dhclient -v <interface>
```

where `<interface>` is the name of the active network interface identified earlier.

After doing this, you should be able to rerun `ifconfig -a` and see that the IP address has updated. If the new IP address matches the one in the DNS database, the computer should be accessible again.

CHAPTER 18

Screen Sharing Using SSVNC

The program SSVNC allows you to control DynamicsHJC remotely. The advantage of using this program over Mac OS X's native screen sharing program is that it does not require someone sitting in front of the monitor to press the "Share my screen" button. It does, however, require that the account has been logged into once already since the last reboot, so that the VNC server is running.

If you are using a Mac and SSVNC is already installed and configured on your machine, you will find the executable located at `/Applications/ssvnc/MacOSX/ssvnc`. After launching the program, you simply need to load the correct profile, press "Connect", and enter the vbox password.

If SSVNC is not already installed, download the tar ball for SSVNC here:

> http://sourceforge.net/projects/ssvnc/

On a Mac, unpack the archive into the `Applications` directory. In theory, a Mac user should be able to simply double-click on `/Applications/ssvnc/MacOSX/ssvnc` to start the program.

However, as of July 2015, the SSVNC tar ball for Mac (`ssvnc_no_windows-1.0.29.tar.gz`) lacks binaries for the "Darwin x86_64" platform. (You can run `uname -sm` on the command line to see what platform you are running.) If you try to run the executable, it will detect your platform automatically, and it will fail when it does not find a directory named `/Applications/ssvnc/bin/Darwin.x86_64` containing needed binaries.

Fortunately, the binaries provided for "Darwin i386" work on "Darwin x86_64", so we can work around this error. In `/Applications/ssvnc/bin`, duplicate the `Darwin.i386` directory and rename the copy to `Darwin.x86_64`.

You should now be able to run `/Applications/ssvnc/MacOSX/ssvnc`. If it complains about the developer not being trusted, try Control-clicking on the executable and selecting "Open".

In the field "VNC Host:Display", enter `vbox@dynamicshjc.case.edu:8`, and select "Use SSH". Click the "Options" button, then the "Advanced" button, and finally the "Unix ssvncviewer" button. In the field "Scaling", enter `0.75`. Click the "Done" button. Click the "Save" button to save this profile so that you will not need to re-enter this information in the future. Finally, press "Connect" and enter the vbox password.

Note that the display port "8" must match the parameter associated with the VNC server running on DynamicsHJC for the vbox account. This will not change unless the VNC server is reconfigured.

CHAPTER 19

Cleaning the Boot Partition

Sometimes you will log into the virtual machine and see a message like this one:

```
=> /boot is using 94.5% of 235MB
```

This should NOT be ignored.

`/boot` is the mount point for the boot partition: a small partition (roughly 250 MB) of the hard drive that stores the boot loader (GRUB) and multiple versions of the Linux kernel (core of the operating system). When a new version of the Linux kernel becomes available, it will be downloaded and installed in the boot partition automatically, but the system needs to be rebooted before it can switch over to using the newest kernel.

Because a reboot is required to cease using the old kernel, it cannot be removed automatically. Consequently, over time the boot partition will fill up with several outdated versions of the Linux kernel until a sysadmin does something about it. Worse yet, critical security updates and non-critical software updates will eventually require newer versions of the kernel, and the `apt-get upgrade` command will fail until room is made in the boot partition for the version of the kernel that they depend on. **If the boot partition is not maintained, security updates will eventually cease, and manual system updates will fail until the problem is fixed.**

To free up space in the boot partition, first reboot the virtual machine so that it will switch to using the latest installed version of the kernel:

```
sudo shutdown -r now
```

Next, remove old versions of the kernel (this command will remove all but the two most recent versions in case there is something wrong with the latest):

```
sudo apt-get remove $(dpkg --list 'linux-image-[0-9]*-generic' | grep ^ii | awk '
↪{print $2}' | sort | head -n -2 | grep -v $(uname -r))
sudo apt-get autoremove
```

You can now check disk usage in the boot partition:

```
df -h /boot
```

It should be much lower than it was before, probably below 50%.

CHAPTER 20

Expanding the Virtual Hard Drive

If a lot of material is uploaded to the wiki (e.g., problem set screenshots), backups may begin to fail due to insufficient hard drive space. The virtual hard drive can be grown to a larger size if necessary.

1. Shut down the virtual machine.
2. Export a VirtualBox appliance as a backup.
3. Delete all snapshots. These silently interfere with the virtual machine's ability to recognize the increased storage.
4. In VirtualBox, under Settings > Storage, note the file location of the VDI storage file. Locate it now in Finder and make a backup of it.
5. Remove the storage attachment.
6. In Terminal, run

   ```
   VBoxManage modifyhd <location> --resize <size>
   ```

   where `<location>` is the path to the original VDI file, and `<size>` is an integer number of megabytes that the resized storage should have.

   For example, to expand to 40 GB use something like this:

   ```
   VBoxManage modifyhd /Users/vbox/VirtualBox\ VMs/BIOL\ 373/2016/neurowiki_2016/
   ↪neurowiki_2016.vdi --resize 40960
   ```

7. In VirtualBox, under Settings > Storage, re-attach the VDI file under "Controller: SATA".
8. On DynamicsHJC, download the GParted Live Bootable Image (ISO).
9. In VirtualBox, under Settings > Storage, add the ISO, and make sure the boot order under Settings > System has "Optical" checked and prioritized above "Hard Disk".
10. Start the virtual machine to boot into GParted, and press Enter a few times to get through the initial prompts.
11. In the GParted interface, you should see some unallocated space that corresponds to the extra space added using `VBoxManage`.
12. If the main partition and/or its child partitions are locked, "Deactivate" them.

13. Expand the main partition and its child partitions to include the unallocated space.
14. Restart the virtual machine. It should boot normally.
15. Check the disk space using

    ```
    df -h /
    ```

    The “Size” probably won’t reflect the new space added. If it doesn’t, run

    ```
    sudo lvextend -l +100%FREE <device>
    sudo resize2fs <device>
    ```

    where `<device>` is the path given in the first column of the result of `df -h /`.

    Verify that this worked by checking the disk space again.

CHAPTER 21

Updating MediaWiki

The MediaWiki-announce mailing list provides announcements for security updates and new releases. Emails from this list provide links to patches that make upgrading easy. The patches can also be found here. General instructions for upgrading can be found here.

The following provides an example of how to upgrade using a patch file:

```
wget -P ~ http://releases.wikimedia.org/mediawiki/1.27/mediawiki-1.27.2.patch.gz
gunzip ~/mediawiki-1.27.2.patch.gz
patch -p1 -d /var/www/mediawiki < ~/mediawiki-1.27.2.patch
rm ~/mediawiki-1.27.2.patch
sudo chown -R www-data:www-data /var/www/mediawiki
sudo chmod -R ug+rw /var/www/mediawiki
sudo chmod ug=rw,o= /var/www/mediawiki/LocalSettings.php*
php /var/www/mediawiki/maintenance/update.php
```

CHAPTER 22

Updating Django

The django-announce mailing list provides announcements for security updates and new releases. General instructions for upgrading can be found here.

To see which version of Django is *currently* installed, run

```
pip show Django
```

To see which versions of Django are *available*, run

```
pip install Django==null
```

Upgrading the Django installation should be as simple as running

```
sudo pip install Django==1.8.X
sudo apache2ctl restart
```

where `1.8.X` should be replaced with a valid version number.

Once this is completed, you should try visiting our Django site, including the admin side of the site, to look for anything obviously broken. You can run some simple tests, such as opening and closing surveys, or changing a user's privileges.

If bugs are discovered after upgrading, it is possible that you will need to make changes to *our* code—the CourseDjango application. Do not forget to commit and push any fixes to the Git repository (see *Introduction to Git* for help with this)!

CHAPTER 23

Changing the Wiki Logo or Favicon

Todo: Provide instructions for changing the wiki logo or favicon, including skin adjustments to accomodate logo size.

CHAPTER 24

Wiki Maintenance Cheatsheet

Todo: The instructions for new TAs ought to be moved to their own page.

- Steps to take for new TAs:
  - Ask the TA to log into the wiki to automatically create their MediaWiki account. You should then visit Special:UserRights and add them to the *administrator*, *bureaucrat*, and *grader* user groups.
  - Instruct the TA to replace the contents of their personal page (accessed by clicking their name in the top-right corner while logged in) with the following (they should ignore the warning intended for students):

    ```
    {{instructor links}}
    ```

  - Ask the TA to log into the Django site by visiting any survey to automatically create their Django account. You should then visit the Django user admin page and grant them the *Staff status* and *Superuser status* permissions.
  - Update the biol300@case.edu mailing list subscribers:

    https://groups.google.com/a/case.edu/forum/#!managemembers/biol300/members/active

  - Update the instructor and office hours listing on the *Course policies* page:

    https://biol300.case.edu/wiki/Course_policies

- Some useful commands:
  - Logging in, with port forwarding:

    ```
    ssh -p 8015 hjc@dynamicshjc.case.edu
    ```

  - Logging in, with bridged networking:

    ```
    ssh hjc@biol300.case.edu
    ```

  - Copying files from your local machine to your home folder on the server, with port forwarding:

```
scp -P 8015 /path/to/local/files hjc@dynamicshjc.case.edu:
```

– Copying files from your local machine to your home folder on the server, with bridged networking:

```
scp /path/to/local/files hjc@biol300.case.edu:
```

– Testing Django code as non-staff (do not forget to undo when done with `is_staff=1`!):

```
mysql -u root -p djangodb
UPDATE auth_user SET is_staff=0 WHERE username='hjc';
```

– Emptying the ScholasticGrading grade log (useful for development):

```
echo "DELETE FROM logging WHERE log_type='grades';" | mysql -u root -p wikidb
```

– Viewing APT package descriptions:

```
apt-cache search foo
```

– Cleaning up directories:

```
find . -name \*.pyc -exec rm \{\} \;
find . -name \*~ -exec rm \{\} \;
```

• Important file locations:

Apache error and access logs	`/var/log/apache2/`
Apache configuration directory	`/etc/apache2/`
Apache configuration files	`/etc/apache2/conf-available/`
SSL certificates	`/media/sf_ssl-certificates/`
Web root directory	`/var/www/html/`
MediaWiki source directory	`/var/www/mediawiki/`
MediaWiki extensions directory	`/var/www/mediawiki/extensions/`
MediaWiki configuration file	`/var/www/mediawiki/LocalSettings.php`
Django directory	`/var/www/django/`

CHAPTER 25

Updating this Documentation

25.1 Source Repository and Read the Docs

The source code for this documentation is hosted on GitHub. Changes made there will automatically be compiled and appear on Read the Docs.

25.2 Compiling the Docs Locally

Todo: Add general instructions for compiling the docs. Note that these can also be found in the README.

25.3 Updating the MediaWiki Configuration File

Todo: Document method of creating a pre-configured `LocalSettings.php` file, which is just a modified version of the file created by default by the script `maintenance/install.php`.

25.4 Updating Templates Exported from Wikipedia

The XML files used in *Install Citation Tools* and similar pages were exported from Wikipedia and may be too old to work with future releases of MediaWiki and/or some extensions (especially Scribunto). If you intend to update the docs to use a much newer version of MediaWiki, you may need to generate new versions of the XML files.

It is possible that you can just re-export the necessary *Module* and *Template* pages from Wikipedia, as is decribed in the footnotes on the relevant doc pages. However, if Wikipedia is running a version of MediaWiki or an extension that is newer than the version you plan to upgrade to, and backwards-incompatible changes have been made to the pages you need to export from Wikipedia, simply importing the current versions of these pages may fail in unusual ways (e.g., odd citation rendering errors).

By digging through the histories of these pages and using a binary search guess-and-check strategy, is possible to identify older versions of the pages that are compatible with whichever version of MediaWiki you want to upgrade to. It may help to review the MediaWiki version timeline to get a good initial guess for a date.

Once you have identified a particular moment in time when Wikipedia was using versions of the pages that are compatible with your upgraded MediaWiki installation, you can export a snapshot of all the pages you need from that time using the command line. Here is an example:

```
curl -d "&action=submit&offset=2014-06-01T00:00:00Z&limit=1&dir=desc&pages=
    Module:Arguments
    Module:Category handler
    Module:Category handler/blacklist
    etc. ...
    " http://en.wikipedia.org/w/index.php?title=Special:Export -o "message-box-
↪modules.xml"
```

CHAPTER 26

To-Do

Todo: Add instructions for updating ignored users in the reset-wiki script and for first saving exemplars.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-last-year.rst, line 81.)

Todo: Need to add instructions for updating miscellaneous wiki pages, syllabus dates, assignment dates, survey session dates after resetting the wiki.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-last-year.rst, line 129.)

Todo: Make the backup script that lives on DynamicsHJC and the cron job list downloadable.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/config-backups.rst, line 38.)

Todo: The "Files to Import" directory is now hosted online. Add instructions for modifying it.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/copy-contents.rst, line 146.)

Todo: Update this step with instructions for adding files to the online "BIOL-300-Files-to-Import.tar.bz2" archive, and move the `fetch_wiki_files.sh` script to an external file in the docs source.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/copy-contents.rst, line 165.)

Todo: For exporting content from the 2016 wiki, the commands above are fine, but they should be revised for the future by using the `wikidb` database name.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/create-assignments.rst, line 34.)

Todo: Add a similar message about prefixing uploaded file names with a student's user name to VisualEditor.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/customize-wiki.rst, line 70.)

Todo: Consider what should be done about the User namespace template if I can't get VisualEditor to work with NewArticleTemplate.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/customize-wiki.rst, line 152.)

Todo: Walkthrough deleting all but the most recent snapshot to save disk space.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/delete-snapshots.rst, line 4.)

Todo: Get Wikipedia's advanced citation tools for VisualEditor, especially automatic citation completion from URLs, DOIs, and PMIDs.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-citation-tools.rst, line 4.)

Todo: Add screenshots showing the various citation tools in action.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-citation-tools.rst, line 9.)

Todo: Add tests for VisualEditor's citation tools.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-citation-tools.rst, line 144.)

Todo: Change parsoid install command to use a specific version known to work with MW 1.27 (version "0.5.3all" in Fall 2016 or "0.6.1all" in Spring 2017).

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 46.)

Todo: Consider adding Mathoid installation instructions. In its current form, this instruction set utilizes the Math extension's PNG mode. By changing `$wgDefaultUserOptions['math']` to `'mathml'` in `LocalSettings.php`, an alternate MathML mode can be used. This requires a Mathoid server for runtime equation rendering. The `$wgMathFullRestbaseURL` setting specifies a server in Germany that is free to use but is occasionally unresponsive, causing the wiki to either load slowly or fail to render equations. By building a local Mathoid server, responsiveness and reliability could be guarenteed. However, after much effort, Jeff has not been able to get a Mathoid installation working yet. Since MathML offers few advantages over PNG mode, getting this working is a low priority.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 143.)

Todo: Update the patch for MobileFrontend. For now, skip this step during installation.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 195.)

Todo: Explain the function of NewArticleTemplate better here and in terms of the subpages actually used by students.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 218.)

Todo: Add hooks to NewArticleTemplate so that it works with VisualEditor.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 252.)

Todo: Fork NewArticleTemplate on GitHub and incorporate the "always use subpage template" patch and the hooks for VisualEditor.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 256.)

Todo: Fork Realnames on GitHub and incorporate the "ignore subpage titles" patch.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 280.)

Todo: Post my solution for the VisualEditor + Lockdown extension conflict to the discussion board, and then provide a "This solution is documented here" footnote in these instructions.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 329.)

Todo: Embedded videos don't render in VisualEditor. Fix?

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 415.)

Todo: When attempting to upload files through VisualEditor, it thinks I'm not logged in. Fix!

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-extensions.rst, line 419.)

Todo: Add screenshots showing what the message box test result should look like.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/build-from-scratch/install-mbox.rst, line 72.)

Todo: Provide instructions for changing the wiki logo or favicon, including skin adjustments to accomodate logo size.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/change-logo-favicon.rst, line 4.)

Todo: Make the backup script that lives on DynamicsHJC and the cron job list downloadable.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/config-dev-server.rst, line 51.)

Todo: Provide form emails for instructors (e.g., absences, tardies).

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/form-emails.rst, line 4.)

Todo: Write an intro to Git.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/git-intro.rst, line 4.)

Todo: Add instructions for giving students privileges to access the private areas of the wiki (relevant to BIOL 373 only) and for getting a term paper deadline extension.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/grant-student-privileges.rst, line 4.)

Todo: Verify that Student Device Registration works for non-students.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/mac-addresses.rst, line 59.)

Todo: The instructions for new TAs ought to be moved to their own page.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/maintenance-cheatsheet.rst, line 4.)

Todo: Install phpMyAdmin for a web interface for changing real names.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/modify-real-names.rst, line 9.)

Todo: Add instructions for changing real names in Django.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/modify-real-names.rst, line 36.)

Todo: Add general instructions for compiling the docs. Note that these can also be found in the README.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/update-docs.rst, line 19.)

Todo: Document method of creating a pre-configured `LocalSettings.php` file, which is just a modified version of the file created by default by the script `maintenance/install.php`.

(The original entry is located in /home/docs/checkouts/readthedocs.org/user_builds/biol-300-wiki-docs/checkouts/latest/source/update-docs.rst, line 30.)