i5k_doc Documentation

Release 1.0

Fish Lin

Sep 26, 2018

Table of Contents

Genomics workspace is a open-source project created by i5k workspace of NAL.

In this project, we produced a Django website with functionality of common sequence searchs including BLAST, HMMER, and Clustal.

Leveraging the admin page of Django and task queue by RabbitMQ and Celery, it's much easier to manage the sequence databases and provide services to end-users.

All source codes of genomics workspace are in our github repo.

Note: You can try genomics workspace on our live services:

- BLAST: https://i5k.nal.usda.gov/webapp/blast/
- HMMER: https://i5k.nal.usda.gov/webapp/hmmer/
- Clustal: https://i5k.nal.usda.gov/webapp/clustal/

In fact, the live services listed above are implemented by a customized version of genomics workspace. You can check the source code of it in another github repo: NAL-genomics-workspace.

CHAPTER 1

Pre-requisites

- git
- Python 2.7
- npm
- RabbitMQ
- PostgreSQL
- mod_wsgi (optional, only for production)

CHAPTER 2

Setup Guide

This is our introduction to this project.

2.1 Setup Guide (CentOS)

This setup guide is for CentOS. It's tested in CentOS 6.7 and CentOS 7.2, but it should also work on all modern linux distributions.

Note: The following variables may be used in path names; substitute as appropriate:

```
<user>      :  the name of the user doing a set up.
<user-home> :  the user's home directory, e.g., /home/<user>
<git-home>  :  the directory containing the genomics-workspace, and `.git/` folder␣
↪for `git` will be there.
```

2.1.1 Project Applications

Clone or refresh the genomics-workspace:

```
git clone https://github.com/NAL-i5K/genomics-workspace

# Or if the  repository exists:
cd <git-home>
git fetch
```

2.1.2 Yum

Generate metadata cache:

```
yum makecache
```

2.1.3 Python

Install necessary packages:

```
sudo yum -y groupinstall "Development tools"
sudo yum -y install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel
sudo yum -y install readline-devel tk-devel gdbm-devel db4-devel libpcap-devel xz-
↪devel python-devel
```

Install python 2.7.13 from source:

```
cd <user-home>
wget http://www.python.org/ftp/python/2.7.13/Python-2.7.13.tar.xz
tar -xf Python-2.7.13.tar.xz

# Configure as a shared library:
cd Python-2.7.13
./configure --prefix=/usr/local --enable-unicode=ucs4 --enable-shared LDFLAGS="-Wl,-
↪rpath /usr/local/lib"

# Compile and install:
make
sudo make altinstall

# Update PATH:
export PATH="/usr/local/bin:$PATH"

# Checking Python version (output should be: Python 2.7.13):
python2.7 -V

# Cleanup if desired:
cd ..
rm -rf Python-2.7.13.tar.xz Python-2.7.13
```

Install pip and virtualenv:

```
wget https://bootstrap.pypa.io/ez_setup.py
sudo /usr/local/bin/python2.7 ez_setup.py

wget https://bootstrap.pypa.io/get-pip.py
sudo /usr/local/bin/python2.7 get-pip.py

sudo /usr/local/bin/pip2.7 install virtualenv
```

Build a separate virtualenv:

```
cd <git-home>

# Create a virtual environment called py2.7 and activate:
virtualenv -p python2.7 py2.7
source py2.7/bin/activate
```

2.1.4 RabbitMQ

Install RabbitMQ Server:

```
cd <user-home>

# Install RHEL/CentOS 6.8 64-Bit Extra Packages for Enterprise Linux (Epel).
# The 6.8 Epel caters for CentOS 6.*:
wget https://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
sudo rpm -ivh epel-release-6-8.noarch.rpm

# For RHEL/CentOS 7.* :
# wegt http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-10.noarch.rpm
# and change other commands accordingly

# Install Erlang:
sudo yum -y install erlang

# Install RabbitMQ server:
sudo yum -y install rabbitmq-server

# To start the daemon by default when system boots run:
sudo chkconfig rabbitmq-server on

# Start the server:
sudo /sbin/service rabbitmq-server start

# Clean up:
rm epel-release-6-8.noarch.rpm
```

2.1.5 Memcached

Install and activate memcached:

```
sudo yum -y install memcached

# Set to start at boot time:
sudo chkconfig memcached on
```

2.1.6 Database

Install PostgreSQL:

```
# Add line to yum repository:
echo 'exclude=postgresql*' | sudo tee -a /etc/yum.repos.d/CentOS-Base.repo

# Install the PostgreSQL Global Development Group (PGDG) RPM file:
sudo yum -y install http://yum.postgresql.org/9.5/redhat/rhel-6-x86_64/pgdg-centos95-
↪9.5-2.noarch.rpm

# Install PostgreSQL 9.5:
sudo yum -y install postgresql95-server postgresql95-contrib postgresql95-devel

# Initialize (uses default data directory: /var/lib/pgsql):
sudo service postgresql-9.5 initdb

# Startup at boot:
sudo chkconfig postgresql-9.5 on
```

(continues on next page)

(continued from previous page)

```
# Control:
# sudo service postgresql-9.5 <command>
#
# where <command> can be:
#
#     start   : start the database.
#     stop    : stop the database.
#     restart : stop/start the database; used to read changes to core configuration␣
↪files.
#     reload  : reload pg_hba.conf file while keeping database running.

# Start:
sudo service postgresql-9.5 start

#
#  (To remove everything: sudo yum erase postgresql95*)
#

# Create django database and user:
sudo su - postgres
psql

# At the prompt 'postgres=#' enter:
create database django;
create user django;
grant all on database django to django;
ALTER USER django CREATEDB;

# Connect to django database:
\c django

# Create extension hstore:
create extension hstore;

# Exit psql and postgres user:
\q
exit

# Config in pg_hba.conf:
cd <git-home>
export PATH=/usr/pgsql-9.5/bin:$PATH

# Restart:
sudo service postgresql-9.5 restart
```

2.1.7 Python Modules and Packages

Install additional Python packages:

```
cd <git-home>
pip install -r requirements.txt
```

2.1.8 Chrome Driver

- Install ChromeDriver from https://sites.google.com/a/chromium.org/chromedriver/downloads
- Add to PATH

2.1.9 Celery

Configure celery:

```
# Run celery manually
celery -A i5k worker --loglevel=info --concurrency=3
# Run celery beat maually as well
celery -A i5k beat --loglevel=info
```

2.1.10 Migrate Schema to to PostgreSQL

Run migrate:

```
cd <git-home>
python manage.py migrate
```

2.1.11 Install Binary Files and Front-end Scripts

This step will instll binary files (for BLAST, HMMER and Clustal) and front-end scripts (*.js*, *.css* files):

```
npm run build
```

2.1.12 Start development server

To run developement server:

```
cd <git-home>
python manage.py runserver
```

2.2 Setup Guide (MacOS)

This setup guide is tested in MacOS Sierra (10.12) and MacOS High Sierra (10.13), but it should also work on all recent MacOS versions.

Note: The following variables may be used in path names; substitute as appropriate:

```
<user>      :  the name of the user doing a set up.
<user-home> :  the user's home directory, e.g., /home/<user>
<git-home>  :  the directory containing the genomics-workspace, and `.git/` folder␣
↪for `git` will be there.
```

2.2.1 Project Applications

Clone or refresh the genomics-workspace:

```
git clone https://github.com/NAL-i5K/genomics-workspace

# Or if the  repository exists:
cd <git-home>
git fetch
```

2.2.2 Homebrew

We recommend to use Homebrew as package manager. Installation steps can be found at https://brew.sh/.

2.2.3 Python

Install virtualenv:

```
pip install virtualenv
```

Build a separate virtualenv:

```
# Make root dir for virtualenv and cd into it:
cd genomics-workspace

# Create a virtual environment called py2.7 and activate:
virtualenv -p python2.7 py2.7
source py2.7/bin/activate
```

2.2.4 RabbitMQ

Install and run RabbitMQ Server:

```
brew install rabbitmq
# Make sure /usr/local/sbin is in your $PATH
rabbitmq-server
```

2.2.5 Memcached

Install and activate memcached:

```
brew install memcached
memcached
```

2.2.6 Database

Install PostgreSQL:

```
brew install postgres
psql postgres

# At the prompt 'postgres=#' enter:
create database django;
create user django;
grant all on database django to django;
ALTER USER django CREATEDB;

# Connect to django database:
\c django

# Create extension hstore:
create extension hstore;

# Exit psql and postgres user:
\q
exit
```

2.2.7 Python Modules and Packages

Install additional Python packages:

```
cd <git-home>
pip install -r requirements.txt
```

2.2.8 Chrome Driver

- Install ChromeDriver from https://sites.google.com/a/chromium.org/chromedriver/downloads
- Add to PATH

2.2.9 Celery

Configure celery:

```
# Run celery manually
celery -A i5k worker --loglevel=info --concurrency=3
# Run celery beat maually as well
celery -A i5k beat --loglevel=info
```

2.2.10 Migrate Schema to to PostgreSQL

Run migrate:

```
cd <git-home>
python manage.py migrate
```

2.2.11 Install Binary Files and Front-end Scripts

This step will instll binary files (for BLAST, HMMER and Clustal) and front-end scripts (*.js*, *.css* files):

```
npm run build
```

2.2.12 Start development server

To run developement server:

```
cd <git-home>
python manage.py runserver
```

2.3 Advanced Setup

2.3.1 JBrowse/Apollo Linkout Integration

In Genomics workspace, we have a linkout integration between BLAST and JBrowse/Apollo. You can directly go to corresponding sequence location through clicking entries in BLAST result table. To start using it, make change of `ENABLE_JBROWSE_INTEGRATION` in `i5k/settings.py`;

```
ENABLE_JBROWSE_INTEGRATION = True
```

CHAPTER 3

User Guide

BLAST, HMMER, and Clustal are main functionality of genomics-workspace. Each one of this, we implemented it as a single app under Django.

In this section, we will go through details about how to configure each one of them.

In short, you need to configure database for BLAST and HMMER, but you don't need to configure anything for Clustal.

Note: The page is for user that wants to set up genomics-workspace by creating new admin user and confuguring in admin page. **If you want to know how to use services provided by genomics-workspace, see these tutorials:**

- BLAST: https://i5k.nal.usda.gov/content/blast-tutorial
- HMMER: https://i5k.nal.usda.gov/webapp/hmmer/manual/
- CLUSTAL: https://i5k.nal.usda.gov/webapp/clustal/manual/

To get started, you need to setup an admin account:

```
python manage.py createsuperuser
```

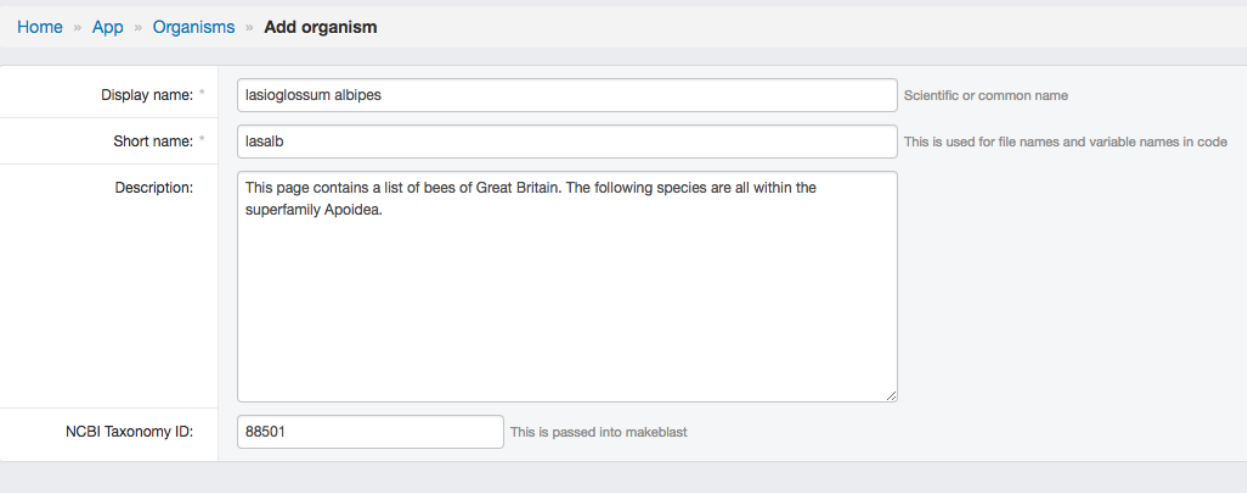
Follow the instruction shown on your terminal, then browse and login to the admin of genomics-workspace. Usually, the admin page should be at `http://127.0.0.1:8000/admin/`.

3.1 BLAST Database Configuration

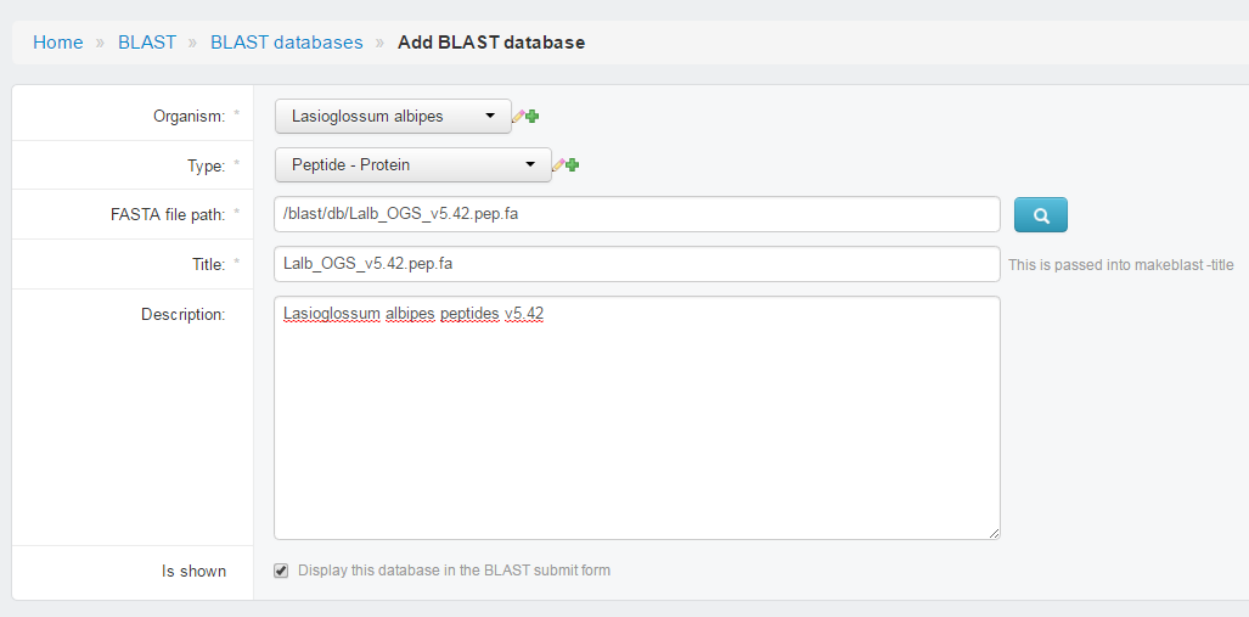
There are five steps to create a BLAST database.

- Add Organism (click the **Organism** icon at sidebar and click **Add organism**):
  - Display name should be scientific name.
  - Short name are used by system as a abbreviation.

  - Descriptions and NCBI taxa ID are automatically filled.



- Add Sequence types:
  - Used to classify BLAST DBs in distinct catagories.
  - Provide two kinds of molecule type for choosing, Nucleotide/Peptide.
- Add Sequence
- Add BLAST DB
  - Choose `Organsim`
  - Choose `Type` (Sequence type)
  - Type location of fasta file in `FASTA file path` (It should be in `<git-home>/media/blast/db/`)
  - Type `Title` name. (showed in HMMER page)
  - Type `Descriptions`.
  - Check `is shown`, if not check, this database would show in HMMER page.
  - Save



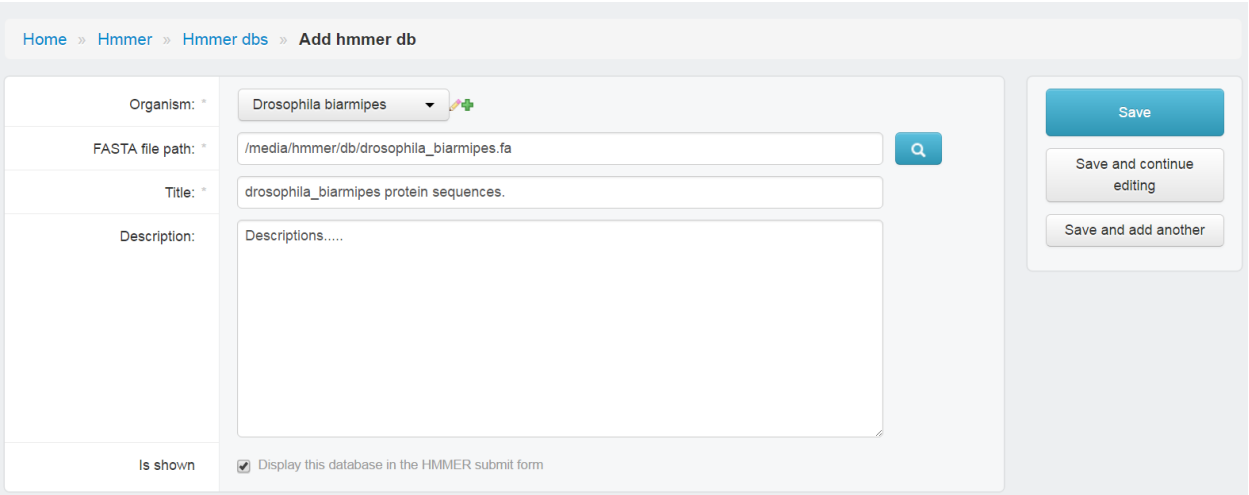
- Browse to `http://127.0.0.1:8000/blast/`, you should able to see the page with dataset shown there.

3.2 HMMER Database Configuration

Like BLAST, HMMER databases must be configured then they could be searched.

Go django admin page and click Hmmer on left-menubar. You need to create HMMER db instance (Hmmer dbs) for each fasta file.

- Choose `Organsim`
- Type location of peptide fasta file in `FASTA file path`
- Type `Title` name. (showed in HMMER page)
- Type `Descriptions`.
- Check `is shown`, if not check, this database would show in HMMER page.
- Save



CHAPTER 4

How to Deploy

In short, you need to setup following tools and services:

- Apache HTTP server
- mod_wsgi
- RabbitMQ
- Celery and celerybeat runs in daemon mode.

Because genomics workspace is a standard Django website, there is no large difference to deploy genomics workspace. We recommed to deploy genomics workspace through Apache and mod_wsgi.

You may want take a look the great documentation of Django project on deploying as well.

4.1 Apache HTTP server and mod_wsgi

See the document of Django. You can also see the example settings file of Apache and mod_wsgi in our github repo.

4.2 RabbitMQ

Use the rabbitmq-server command.

4.3 Celery and celerybeat

Here are example setup steps for linux,

1. Copy files:

```
# when using CentOS 7.*
# copy celeryd.sysconfig and celerybeat.sysconfig to /etc/default instead.
sudo cp celeryd /etc/init.d
sudo cp celerybeat /etc/init.d
sudo cp celeryd.sysconfig /etc/sysconfig/celeryd
sudo cp celerybeat.sysconfig /etc/sysconfig/celerybeat
```

2. edit '/etc/sysconfig/celeryd':

```
CELERYD_CHDIR="<git-home>"
CELERYD_MULTI="<git-home>/py2.7/bin/celery multi"
```

3. edit '/etc/sysconfig/celerybeat' as follows:

```
CELERYBEAT_CHDIR="<git-home>"
CELERY_BIN="<git-home>/py2.7/bin/celery"
```

4. set as daemon:

```
sudo chkconfig celeryd on
sudo chkconfig celerybeat on
```

For more details or setup on Mac, check the document of Celery. Example files mentioned above are also (celery*) in our github repo.

CHAPTER 5

Trouble Shooting

Q: I get an error message like: `FATAL: Ident authentication failed`. How can I fix this ?

A: It's because the setting of PostgreSQL database. Try to modify the config file `pg_hba.conf`. For example, in PostgreSQL 9.5, the file is at `/var/lib/pgsql/9.5/data/pg_hba.conf`. Make sure you change part of the content of it into something like:

```
local   all             all                                     peer
host    all             all             127.0.0.1/32            ident
host    all             all             ::1/128                 md5
```

CHAPTER 6

About i5k Workplace at NAL

The i5k Workspace at NAL is a platform for communities around 'orphaned' arthropod genome projects to access, visualize, curate and disseminate their data.

For more information, please see website of i5k Workspace@NAL.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search