GenIE-CMS Documentation

Release 0.1

Chanaka Mannapperuma

Jun 02, 2019

User Documentation

CHAPTER 1

Installation & Updates

1.1 Download & Requirements

You can download the latest version of GenIECMS by using the official download link:

Please note that the above link will onlu download the source code for the GenIE-CMS. If you need to download the parsing scripts, you need to download it here.

If you prefer using the terminal please run to download both the CMS and parsing scripts:

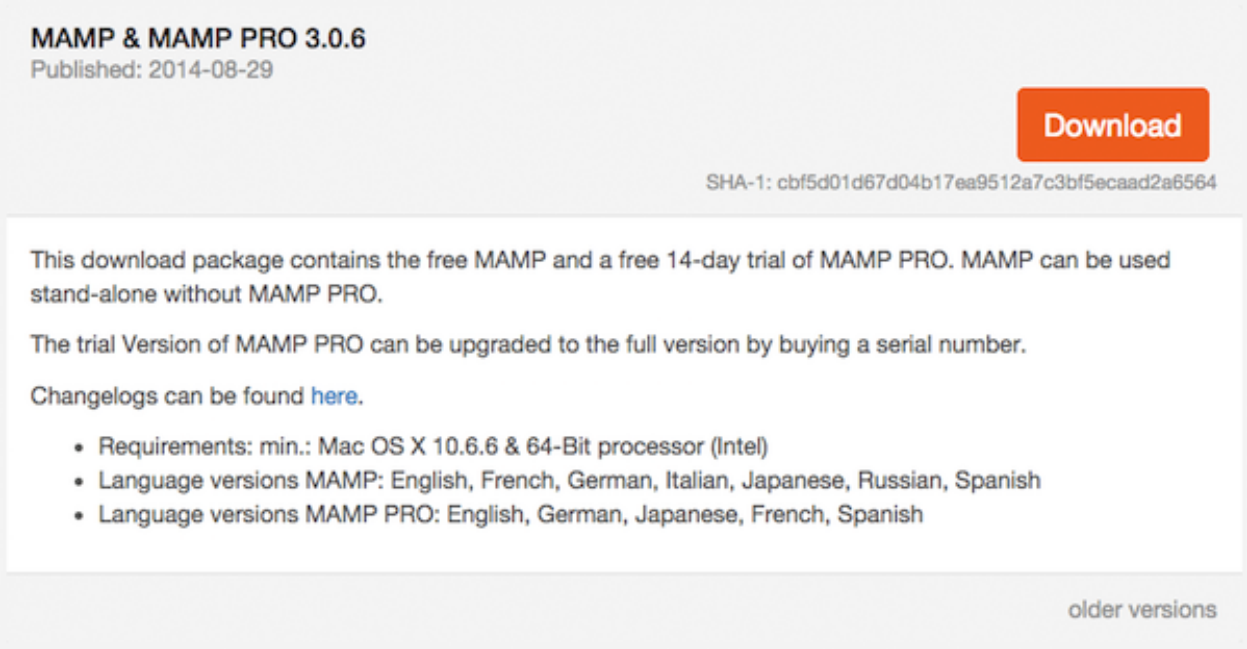
```
git clone --recursive https://github.com/irusri/GenIECMS.git
```

Requirements

- Apache 2 with URL rewriting (mod_rewrite) or nginx
- PHP 5.4+

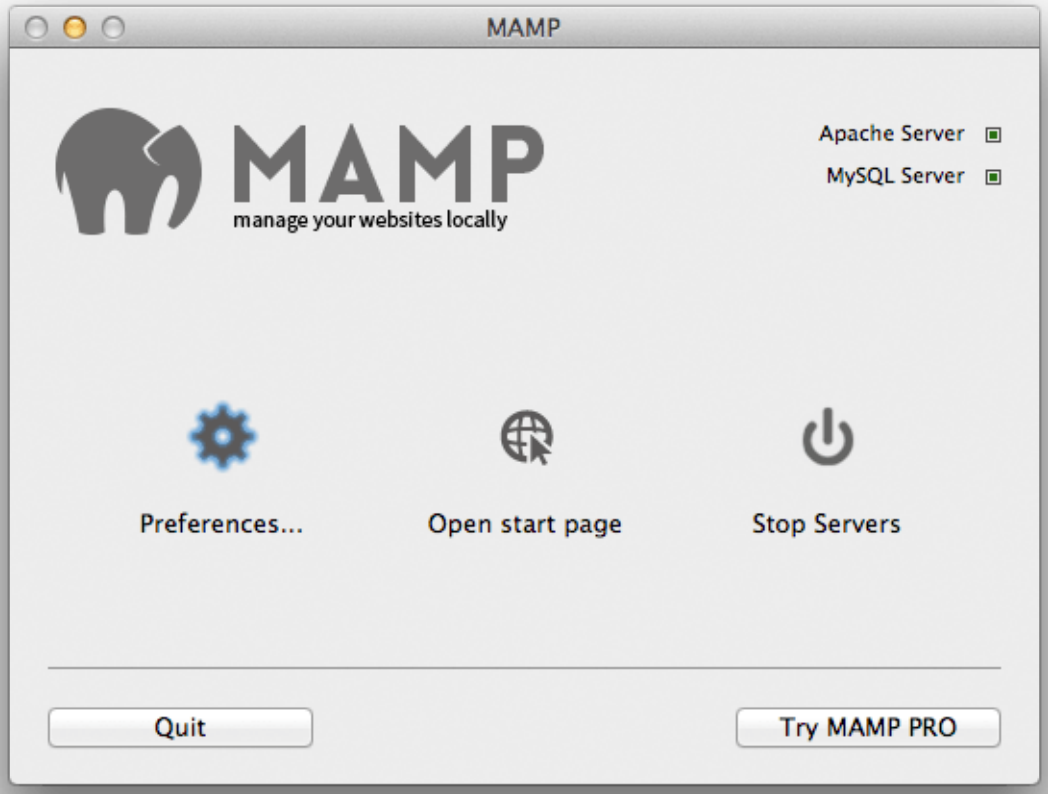
1.2 Installing GenIECMS on a Mac

Most Mac users will probably try GenIECMS with MAMP.



Installing MAMP

Installing MAMP is just a matter of downloading the app from the MAMP website and running the installer. It will install a MAMP app in your Applications folder.



By starting the MAMP app you are also starting your Apache and MySQL server. You should now be able to reach your local server at `http://localhost:8888`.

Download GenIECMS

Download

Copy GenIECMS to MAMP Web server

You will find the source of GenIECMS in your download folder. So you just need to Copy GenIECMS folder into corresponding `~/Applications/MAMP/htdocs/` folder.

That is basically what you need to do in order to install GenIECMS on your Mac's local server. You should now be able to access it at: `http://localhost:8888/GenIECMS` in your browser.

1.3 Update configuration file

We should update the settings file(`GenIECMS/plugins/settings.php`) right after the installtion. Especially the base URL depending on your webhost. For example:

```
/*Define your base url with trailing slash*/
$GLOBALS["base_url"]='http://localhost:8888/GenIECMS/';

OR

$GLOBALS["base_url"]='http://localhost:3000';
```

Next, we need to create a MySQL database and load our data.

1.4 Troubleshooting

GenIECMS can easily be installed without an effort. Unfortunately there is always space for problems due to multiple server setups and PHP versions. In this section, we try to answer most frequent issues in order to install GenIECMS as effortless as possible. Please send us an email if you still get trouble with installation or updates: `contact@geniecms.org`

Subfolder permissions

Web server runs in a different group than your user account on most servers. Following subfolder permissions will necessary to grant write access from GenIECMS.:

```
chgrp -R www-data GenIECMS
chmod -R 775 GenIECMS/genie_files
```

Please make sure that the root folder is also readable by the webserver.

Broken subpages

Whenever you have problems(can not open or a server error) with subpages, you can try following steps.

- Make sure that the .htaccess file is present inside GenIECMS folder.
- mod_rewrite should be enabled on your server.
- You need to check the .htaccess. You can test this by adding some extra characters into your .htaccess. If this cause an "Internal Server Error", the file gets loaded. Otherwise, you need to enable AllowOverride all in your Web server configuration file. An example of `GenIECMS/.htaccess` file shown below.

```
RedirectMatch 403 ^.*/genie_files/
ErrorDocument 403  
RewriteEngine on
Options -Indexes
ServerSignature Off
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond $1#%{REQUEST_URI} ([^#]*)#(.*)\1$
RewriteRule ^([^\.]+)$ %2?page=$1 [QSA,L]
ErrorDocument 404 /notfound.html
```

Please make sure that you are using the PHP 5.4 or higher.

More problems?

Please contact us:`contact@geniecms.org`

1.5 GenIECMS updates

Manual updates

GenIECMS can be updated manually using latest ZIP file from GitHub. Please backup your older version of `GenIECMS/plugins/settings.php` and `GenIECMS/genie_files` before you do the latest update. First unzip the genie.zip file from your download folder and move into the Web Server server. Finally copy the `GenIECMS/plugins/settings.php` and `GenIECMS/genie_files` into latest version of GenIECMS.

Updates using Git

Here is the easy way to update GenIECMS using git submodules:

```
cd GenIECMS
git checkout master
git pull
git submodule foreach --recursive git checkout master
git submodule foreach --recursive git pull
```

1.6 Docker installation

For Developers and Contricutors

```
# Please comment the supporting_files/run.sh line to avoid download the geniecms.git
git clone https://github.com/irusri/docker4geniecms.git
cd docker4geniecms
git submodule add -f https://github.com/irusri/genie.git
docker build -t genie -f ./Dockerfile .
docker run --rm -i -t -p "80:80" -p "3308:3306" -v ${PWD}/genie:/app -v ${PWD}/mysql:/
↪var/lib/mysql -e MYSQL_ADMIN_PASS="mypass" --name genie genie
cd genie
```

When we need to commit changes, please go to `cd docker4geniecms/genie` folder. Never commit from `docker4geniecms` folder. Because it will add genie as a submodule. Incase you mistakenly pushed from `docker4geniecms` folder, please `cd docker4geniecms` and `git rm genie`. You can access MySQL using `mysql -u admin -pmypass -h localhost -P 3308` or using phpMyAdmin. Some useful docker commands are as follows.

```
# Must be run first because images are attached to containers
docker rm -f $(docker ps -a -q)
# Delete every Docker images
docker rmi -f $(docker images -q)
# To see docker process
docker ps -l
# To see or remove all volumes
docker volume ls/prune
# To run bash inside the running docker container
docker exec -it 890fa15eeef6126b668f4b0fcb7a38b33eaff0 /bin/bash
or
docker attach 890fa15eeef6126b668f4b0fcb7a38b33eaff0
```

Now we can start the real development and push changes into genie.

1.7 Running from Command Line

If you want to use PHP's built-in server (not recommended), just use following lines to install GenIECMS. This is only for the initial test installation, in order to make a full functional website you have to install Webbserver package such as MAMP or LAMP.

```
git clone --recursive https://github.com/irusri/GenIECMS.git
cd GenIECMS
php -S localhost:3000
```

You should now be able to access GenIECMS at: `http://localhost:3000` in your browser.

CHAPTER 2

Getting Started

2.1 What is GenIECMS?

The Genome Integrative Explorer Content Management System (GenIE-CMS) is dedicative in-house CMS to facilitate external groups in setting up their own web resource for searching, visualizing, editing, sharing and reproducing their genomic and transcriptomic data while using project raw data(GFF3,FASTA,FASTQ) as an input.

GenIE-CMS will support cutting-edge genomic science, providing easily accessible, reproducible, and shareable science. The increasingly large size of many datasets is a particularly challenging aspect of current and future genomics based research; it is often difficult to move large datasets between servers due to constraints of time and finance. It is also important to keep the experimental datasets private among the group members until the project goals are accomplished or until after publication. In other words, it must provide a high level of security to ensure that the genomic web resource remains private without requiring the moving of data to unknown remote servers. Therefore, a locally hosted GenIE-CMS installation represents a more secure, less expensive and time consuming resource to implement.

In Addition, Researchers who are not specialized in bioinformatics or have limited computers skills are not currently able to gain maximal insight from the biological data typically produced by genomics projects. In order to overcome this limitation, GenIE-CMS will provide an ideal gateway with simple graphical user interfaces to those who have limited skills in bioinformatics.

Web resources such as Phytozome(Goodst et al., 2012) , iPlant(Goff. et al.,2011), TAIR (Rhee et al., 2003) and PLAZA (Proost et al., 2011). These collections of tools and services have been sources of inspiration to be and have contributed my desire to develop the GenIE-CMS as well as, and importantly, developing an understanding of their limitations to end users. None of these resources allow users to easily setup their own web resource without submitting their data to the resource developers and making them publicly available.

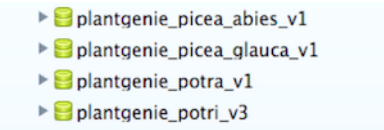
2.2 GenIECMS's folder structure

```
├── GenIECMS
│   ├── data
│   ├── docs
│   ├── genie_files
│   ├── index.php
│   ├── js
│   ├── LICENSE
│   ├── plugins
│   ├── README.md
│   ├── scripts
│   └── themes
```

2.3 Database design

Creating a new database

Due to increasing number of species in PlantGenIE we use standard naming convention to easily identify and maintain the databases. For example: `[website name]_[species name]_[version number]`



Log into the MySQL server and create a database.

```
#Create a database:
CREATE DATABASE new_database;
```

You can download the empty database here. Then load the database into the newly created database using following commands.

```
git show HEAD~1:scripts/dump.sql > dump.sql
mysql -u newuser -p newpassword new_database < dump.sql
```

Log into the MySQL server to create user and grant permissions.

```
#Create MySQL user:
CREATE USER newuser@'localhost' IDENTIFIED BY 'newpassword';

#User permissions:
GRANT SELECT ON new_database.* TO newuser@'localhost';
GRANT INSERT,UPDATE,DELETE ON new_database.genebaskets TO newuser@'localhost';
GRANT INSERT,UPDATE,DELETE ON new_database.defaultgenebaskets TO newuser@'localhost';
```

`newuser, newpassword and new_database` should be included in the plugins/settings.php similar to following example.

```
//Define the databasename names
$db_species_array=array("new_database"=>"new genome",...
//Define the databasename and background colours
```

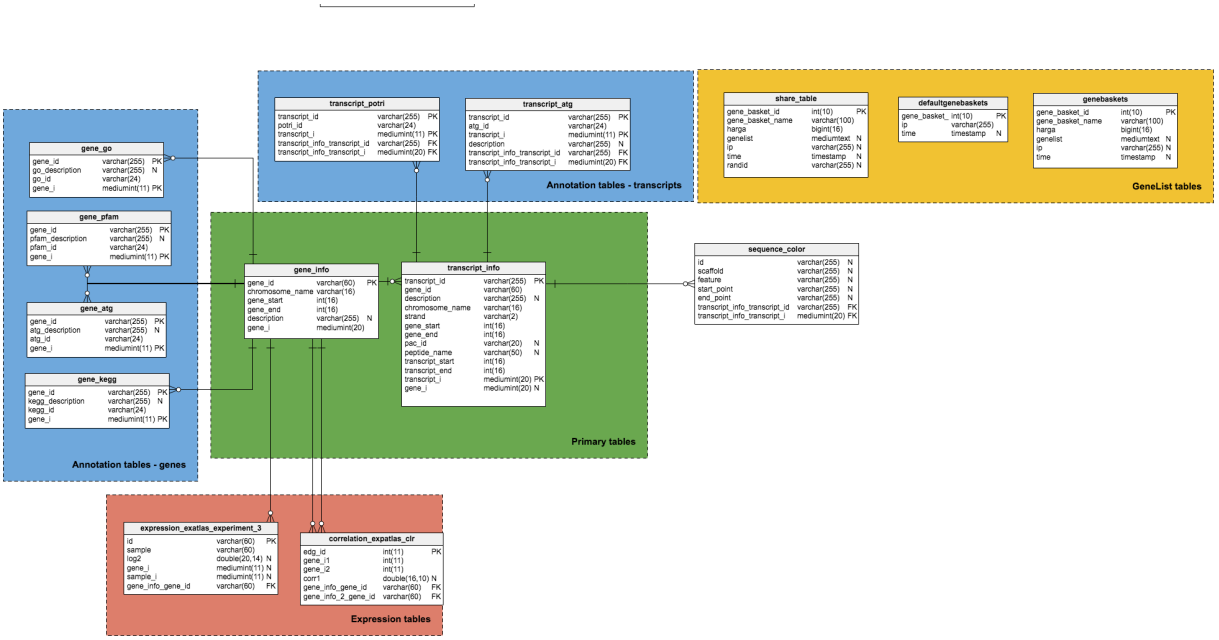
(continues on next page)

(continued from previous page)

```
$db_species_color_array=array("new_database"=>"#86c0a6",....
//Define the username, password and host here
$db_url=  array ('genelist'=>'mysqli://newuser:newpassword@localhost/'.$selected_
↪database);
//Define the base url with trailing slash
$GLOBALS["base_url"]='http://localhost:3000/';
```

Loading tables

Following database diagram shows the initial genie database architecture. It will be used with basic GenIECMS tools such as GeneList, gene information pages, autocomlete search and BLAST.



We have to to follow the data loading instructions in order to load data into the database tables.

2.4 Configuring genome database

All configuration settings in GenIECMS need to be added into `/GenIECMS/plugins/settings.php` file. You need to update `/GenIECMS/plugins/settings.php` file with your available details. You can find everything about the integration plugins and how to load data in the plugins section.

2.5 Plugins/Modules

Analysis, expression or genomic tools can be integreated into a CMS as external plugins. Detailed plugin development guidelines will be available under the plugins section. GenIE-CMS will contain JBrowse, GeneList, gene information pages and BLAST as standard default plugins. All additional tools(exImage, exNet, Enrichment) can be integrated as external plugins to the GenIE-CMS.

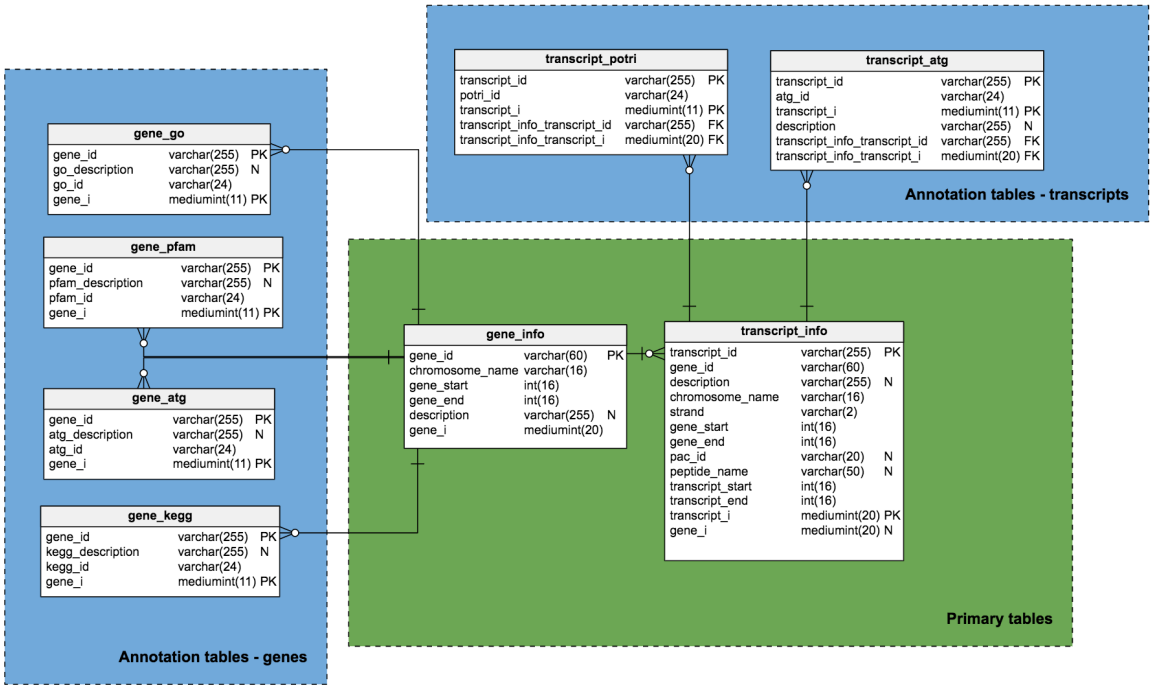
CHAPTER 3

Plugins

3.1 GeneList

Overview

GeneList is the heart of the GenIE-CMS; this will be the entry point to many of the tools and workflows. Foundation to entire CMS database has been designed based on GeneList tables. Tables that are started with *gene_* or *transcript_* prefixes are considered as GeneList tables. GeneList tables consist of two types of tables according to our vocabulary. The first one is primary tables and the second one is annotation tables. *transcript_info* and *gene_info* tables are considered as primary tables and rest of the GeneList tables are known as annotation tables.



Primary tables

There should only be two primary tables (transcript_info and gene_info) in GenIECMS database. Primary tables keep basic gene and transcript information. Since the smallest data unit is based on transcript ids or gene ids, all primary tables are used *transcript_i/gene_i* as a primary key.

Loading data into the primary tables can be easily accomplished using dedicated scripts listed on GenIECMS/scripts folder. First, we need to find corresponding GFF3 and FASTA files related to the species that we are going to load into the GenIE-CMS.

Creating Primary tables

!Important:You do not need to create following tables separately, instead use this script to create all tables at once. Then move to oading data into Primary tables section.

```
#Create transcript_info table
CREATE TABLE `transcript_info` (
  `transcript_id` varchar(60) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `chromosome_name` varchar(20) DEFAULT NULL,
  `transcript_start` int(16) unsigned DEFAULT NULL,
  `transcript_end` int(16) unsigned DEFAULT NULL,
  `strand` varchar(2) DEFAULT NULL,
  `gene_id` varchar(60) DEFAULT NULL,
  `description` varchar(1000) DEFAULT NULL,
  `gene_i` mediumint(16) unsigned DEFAULT NULL,
  `transcript_i` mediumint(16) unsigned NOT NULL AUTO_INCREMENT,
  PRIMARY KEY (`transcript_i`),
  KEY `transcript_id` (`transcript_id`),
  KEY `gene_id` (`gene_id`)
);
#Describe transcript_info table
mysql> explain transcript_info;
```

(continues on next page)

(continued from previous page)

```
+-----------------+-------------------------+------+-----+---------+-----------------+
| Field           | Type                    | Null | Key | Default | Extra           |
+-----------------+-------------------------+------+-----+---------+-----------------+
| transcript_id   | varchar(60)             | NO   | MUL |         |                 |
| chromosome_name | varchar(20)             | YES  |     | NULL    |                 |
| transcript_start | int(16) unsigned       | YES  |     | NULL    |                 |
| transcript_end  | int(16) unsigned        | YES  |     | NULL    |                 |
| strand          | varchar(2)              | YES  |     | NULL    |                 |
| gene_id         | varchar(60)             | YES  | MUL | NULL    |                 |
| description     | varchar(1000)           | YES  |     | NULL    |                 |
| transcript_i    | mediumint(16) unsigned  | NO   | PRI | NULL    | auto_increment  |
| gene_i          | mediumint(16) unsigned  | YES  |     | NULL    |                 |
+-----------------+-------------------------+------+-----+---------+-----------------+
9 rows in set (0.00 sec)
#Create gene_info table
CREATE TABLE `gene_info` (
  `gene_id` varchar(60) CHARACTER SET utf8 NOT NULL,
  `chromosome_name` varchar(20) DEFAULT NULL,
  `gene_start` int(16) unsigned DEFAULT NULL,
  `gene_end` int(16) unsigned DEFAULT NULL,
  `strand` varchar(2) DEFAULT NULL,
  `description` varchar(1000) DEFAULT NULL,
  `peptide_name` varchar(50) DEFAULT NULL,
  `gene_i` mediumint(16) unsigned NOT NULL AUTO_INCREMENT,
  PRIMARY KEY (`gene_i`),
  KEY `gene_id` (`gene_id`)
);
#Describe gene_info table
mysql> explain gene_info;
+-----------------+-------------------------+------+-----+---------+-----------------+
| Field           | Type                    | Null | Key | Default | Extra           |
+-----------------+-------------------------+------+-----+---------+-----------------+
| gene_id         | varchar(60)             | NO   | MUL | NULL    |                 |
| chromosome_name | varchar(20)             | YES  |     | NULL    |                 |
| gene_start      | int(16) unsigned        | YES  |     | NULL    |                 |
| gene_end        | int(16) unsigned        | YES  |     | NULL    |                 |
| strand          | varchar(2)              | YES  |     | NULL    |                 |
| description     | varchar(1000)           | YES  |     | NULL    |                 |
| peptide_name    | varchar(50)             | YES  |     | NULL    |                 |
| gene_i          | mediumint(16) unsigned  | NO   | PRI | NULL    | auto_increment  |
+-----------------+-------------------------+------+-----+---------+-----------------+
8 rows in set (0.00 sec)
#Adding indeices to transcript_info and gene_info tables is important when we update␣
↪and select tables.
mysql> ALTER TABLE transcript_info ADD INDEX `transcript_id` (`transcript_id`)
mysql> ALTER TABLE transcript_info ADD INDEX `gene_id` (`gene_id`)
mysql> ALTER TABLE gene_info ADD INDEX `gene_id` (`gene_id`)
```

The following example will show you how to load basic information into the primary tables.

Loading data into Primary tables

```
#head  input/Potra01-gene-mRNA-wo-intron.gff3
Potra000001        leafV2        gene        9066        10255        .        -
↪        .        ID=Potra000001g00001;Name=Potra000001g00001;potri=Potri.004G180000,
↪Potri.004G180200
Potra000001        leafV2        mRNA        9066        10255        .        -
↪        .        ID=Potra000001g00001.1;Parent=Potra000001g00001;
↪Name=Potra000001g00001;cdsMD5=71c5f03f2dd2ad2e0e00b15ebe21b14c;primary=TRUE
```

(continues on next page)

(continued from previous page)

```
Potra000001        leafV2        three_prime_UTR        9066        9291        .
↪        -        .        ID=Potra000001g00001.1.3pUTR1;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        exon        9066        9845        .        -
↪        .        ID=Potra000001g00001.1.exon2;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        CDS        9292        9845        .        -
↪        2        ID=Potra000001g00001.1.cds2;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        CDS        10113        10236        .        -
↪        0        ID=Potra000001g00001.1.cds1;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        exon        10113        10255        .        -
↪        .        ID=Potra000001g00001.1.exon1;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        five_prime_UTR        10237        10255        .
↪        -        .        ID=Potra000001g00001.1.5pUTR1;Parent=Potra000001g00001.1;
↪Name=Potra000001g00001.1
Potra000001        leafV2        gene        13567        14931        .
↪        +        .        ID=Potra000001g00002;Name=Potra000001g00002;potri=Potri.
↪004G179800,Potri.004G179900,Potri.004G180100
Potra000001        leafV2        mRNA        13567        14931        .
↪        +        .        ID=Potra000001g00002.1;Parent=Potra000001g00002;
↪Name=Potra000001g00002;cdsMD5=df49ed7856591c4a62d602fef61c7e37;primary=TRUE

#Use GFF3 file and generate source input file to load into gene_info mysql table
awk '/gene/{split($9,a,"ID=");split(a[2],b,";");print b[1],$1,$4,$5,$7}' FS='\t' OFS=
↪'\t' input/Potra01-gene-mRNA-wo-intron.gff3 > input/gene_info.txt

#results file(gene_info.txt) looks like following
Potra000001g00001        Potra000001        9066        10255        -
Potra000001g00002        Potra000001        13567        14931        +
Potra000002g00003        Potra000002        8029        9534        +
Potra000002g35060        Potra000002        10226        12730        -
Potra000002g00005        Potra000002        19301        25349        -
Potra000002g00006        Potra000002        33101        36247        +
Potra000002g00007        Potra000002        36609        41740        +
Potra000002g31575        Potra000002        42835        43635        +
Potra000002g31576        Potra000002        52539        53036        +
Potra000002g31577        Potra000002        55010        55465        +

#Use GFF3 and generate source input file to load into transcript_info mysql table
awk '/mRNA/{split($9,a,"ID=");split(a[2],b,";");split(b[1],c,".");print b[1],$1,$4,$5,
↪$7,c[1]}' FS='\t' OFS='\t' input/Potra01-gene-mRNA-wo-intron.gff3 > input/
↪transcript_info.txt

#results file(transcript_info.txt) looks like following
Potra000001g00001.1        Potra000001        9066        10255        -
↪        Potra000001g00001
Potra000001g00002.
↪1        Potra000001        13567        14931        +        Potra000001g00002
Potra000002g00003.
↪1        Potra000002        8029        9534        +        Potra000002g00003
Potra000002g35060.1        Potra000002        10226        12730        -
↪        Potra000002g35060
Potra000002g00005.3        Potra000002        19301        21913        -
↪        Potra000002g00005
```

(continues on next page)

(continued from previous page)

```
Potra000002g00005.2        Potra000002        19301        24937        -
↪        Potra000002g00005
Potra000002g00005.1        Potra000002        19301        25032        -
↪        Potra000002g00005
Potra000002g00005.5        Potra000002        19346        21913        -
↪        Potra000002g00005
Potra000002g00005.4        Potra000002        19346        25349        -
↪        Potra000002g00005
Potra000002g00006.
↪5        Potra000002        33101        35399        +        Potra000002g00006
```

Two files are ready for loading into the primary tables. `load_data.sh` script can be used to load them into the database and `load_data.sh` script can be found inside `GenIECMS/scripts` folder.

```
#!/bin/bash
#load_data.sh
#USAGE: sh load_data.sh [table_name] [filename]
#sh load_data.sh transcript_info transcript_info.txt

DB_USER='your_db_username'
DB_PASS='your_password'
DB='database_name'

/usr/bin/mysql --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 -
↪-database=$DB <<EOFMYSQL
TRUNCATE TABLE $1;
ALTER TABLE $1 AUTO_INCREMENT = 1;
load data local infile '$2' replace INTO TABLE $1 fields terminated by '\t' LINES␣
↪TERMINATED BY '\n' ignore 0 lines;
EOFMYSQL
```

Folowing two lines will load `transcript_info.txt` and `gene_info.txt` files into respective tables.

```
#Load above generated source file into gene_info table
./load_data.sh gene_info gene_info.txt

#Load previously generated source file into transcript_info table
./load_data.sh transcript_info transcript_info.txt
```

Now we just need to fill the description column in gene_info and transcript_info tables. Therefore, we need files similar to folliwng example.

```
#head potra_transcript_description.txt
Potra000001g00001.1        Germin-like protein subfamily 1 member
Potra000001g00002.1        Germin-like protein
Potra000002g00003.1        uncharacterized protein LOC105113244
Potra000002g35060.1        Pyruvate, phosphate dikinase regulatory
Potra000002g00005.3        Gibberellin 2-beta-dioxygenase
Potra000002g00005.2        Gibberellin 2-beta-dioxygenase
Potra000002g00005.1        Gibberellin 2-beta-dioxygenase
Potra000002g00005.5        Gibberellin 2-beta-dioxygenase
Potra000002g00005.4        Gibberellin 2-beta-dioxygenase
Potra000002g00006.5        DnaJ homolog subfamily

#head potra_gene_description.txt
Potra000001g00001        Germin-like protein subfamily 1 member
```

(continues on next page)

(continued from previous page)

```
Potra000001g00002        Germin-like protein
Potra000002g00003        uncharacterized protein LOC105113244
Potra000002g35060        Pyruvate, phosphate dikinase regulatory
Potra000002g00005        Gibberellin 2-beta-dioxygenase
Potra000002g00006        DnaJ homolog subfamily
Potra000002g00007        Tyrosyl-DNA phosphodiesterase
Potra000002g31575        uncharacterized protein LOC105115090
Potra000002g31576        conserved unknown protein
Potra000002g31577        conserved unknown protein
```

There is a script called `update_descriptions.sh` in `GenIECMS/scripts` folder. The script looks like following.

```
#!/bin/bash
#update_descriptions.sh

DB_USER='your_db_username'
DB_PASS='your_password'
DB='database_name'

# if less than two arguments supplied, display  error message
        if [  $# -le 0 ]
        then
                start='\033[0;33m'
                start_0='\033[0;33m'
                start_2='\033[0;31m'
                end='\033[0m'
                echo  "\nUsage:\n$0 ${start}[gene_info/transcript_info] [file_name]$
↪{end}\nEx: ${start_2}sh update_descriptions.sh transcript_info/gene_info potra_
↪descriptions.tsv${end}\n\nWhat it does?\n${start_0}This script will create a two␣
↪columns(ids, descriptions) temporary table and load the [file_name] into it.\nThen␣
↪it will match ids column in temporary table with transcript_ids/gene_ids and update␣
↪the gene/transcript descriptions.\nFinally delete the temporary table.\n${end}"
                exit 1
        fi

table_name=$(echo $1 | awk '{split($0,a,"_");print a[1]}');
tmp_field_name=$table_name"_id"
/usr/bin/mysql --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 -
↪-database=$DB<<EOFMYSQL
CREATE TEMPORARY TABLE tmp_tb(gene_name VARCHAR(60),annotation VARCHAR(1000));
load data local infile '$2' replace INTO TABLE tmp_tb fields terminated by '\t' LINES␣
↪TERMINATED BY '\n' ignore 0 lines;
UPDATE $1 INNER JOIN tmp_tb on tmp_tb.gene_name = $1.$tmp_field_name SET $1.
↪description = tmp_tb.annotation;
DROP TEMPORARY TABLE tmp_tb;
EOFMYSQL
```

We can use `update_descriptions.sh` script to load descriptions into gene_info and transcript_info tables.

```
#Load gene description
./update_descriptions.sh gene_info potra_transcript_description.txt

#Load transcript description
./update_descriptions.sh transcript_info potra_gene_description.txt
```

Finally update the `gene_i`in `transcript_info` table using `update_gene_i.sh`.

```
#!/bin/bash
#update_gene_i.sh

DB_USER='your_db_username'
DB_PASS='your_password'
DB='database_name'

#USAGE: sh update_gene_i.sh

/usr/bin/mysql --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 -
↪-database=$DB <<EOFMYSQL
create temporary table add_gene_i(gene_i MEDIUMINT NOT NULL AUTO_INCREMENT PRIMARY␣
↪KEY, genename VARCHAR(40));
ALTER TABLE add_gene_i AUTO_INCREMENT = 1;
INSERT INTO add_gene_i(genename) select DISTINCT(gene_id) from transcript_info;
UPDATE transcript_info INNER join add_gene_i ON add_gene_i.genename = transcript_info.
↪gene_id SET transcript_info.gene_i = add_gene_i.gene_i;
drop temporary table add_gene_i;
EOFMYSQL
```

Run following command

```
./update_gene_i.sh
```

Annotation tables

Whenever a user needs to integrate new annotation field into the GeneList, it is possible to create a new table which is known as annotation table. The user can create as many annotation tables depend on their requirements.

Loading data into the annotation tables can be easily done using corresponding scripts listed on GenIECMS/scripts folder. First, we need to create the source file to fill the annotation table. The source file should contain two fields. The first field should be either a gene_id or transcript_id and the other fields should be the annotation.

Load data into transcript_[go/pfam/kegg] tables

```
#Let's assume, if we have Best BLAST results similar to following example.
Potra000001g00001.1 AT5G39130.1
Potra000001g00002.1 AT5G39130.1
Potra000002g00003.1 AT4G21215.2
Potra000002g00005.1 AT4G21200.1 ATGA2OX8,GA2OX8
Potra000002g00005.2 AT4G21200.1 ATGA2OX8,GA2OX8
Potra000002g00005.3 AT4G21200.1 ATGA2OX8,GA2OX8
Potra000002g00005.4 AT4G21200.1 ATGA2OX8,GA2OX8
Potra000002g00005.5 AT4G21200.1 ATGA2OX8,GA2OX8
Potra000002g00006.1 AT1G61770.1
Potra000002g00006.2 AT1G61770.1
```

Now we need to create a MySQL Annotation table to load Best BLAST results.

```
#Create transcript_atg table
CREATE TABLE `transcript_atg` (
  `transcript_id` varchar(60) NOT NULL,
  `atg_id` varchar(60) NOT NULL,
  `description` varchar(1000) DEFAULT NULL,
  `transcript_i` mediumint(16) unsigned NOT NULL,
  PRIMARY KEY (`transcript_i`),
  KEY `transcript_id` (`transcript_id`),
  KEY `atg_id` (`atg_id`)
```

(continues on next page)

(continued from previous page)

```
);

#We will load above file into following table.
mysql> explain transcript_atg;
+---------------+------------------------+------+-----+---------+-------+
| Field         | Type                   | Null | Key | Default | Extra |
+---------------+------------------------+------+-----+---------+-------+
| transcript_id | varchar(60)            | NO   | MUL | NULL    |       |
| atg_id        | varchar(60)            | NO   | MUL | NULL    |       |
| description   | varchar(1000)          | YES  |     | NULL    |       |
| transcript_i  | mediumint(16) unsigned | NO   | PRI | NULL    |       |
+---------------+------------------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Previous `load_data.sh` script can be used to load Best BLAST results to `transcript_atg` table.

```
./load_data.sh transcript_atg potra_transcript_atg.txt
```

Finally update the `transcript_i` in `transcript_atg` table using following script.

```
#!/bin/bash
DB_USER='your_db_username'
DB_PASS='your_password'
DB='database_name'

#USAGE sh update.sh transcript_potri
display_usage() {
        echo  "\nUsage:\n$0 [table_name] \n"
        }

# if less than one arguments supplied, display usage
        if [  $# -le 0 ]
        then
                display_usage
                exit 1
        fi

/usr/bin/mysql --host=localhost  --user=$DB_USER --password=$DB_PASS --local_infile=1␣
↪--database=$DB <<EOFMYSQL
UPDATE $1 INNER JOIN transcript_info on transcript_info.transcript_id = $1.transcript_
↪id SET $1.transcript_i = transcript_info.transcript_i;
EOFMYSQL
```

Run following command to update `transcript_i`

```
./update_transcript_i.sh transcript_atg
```

Load data into gene_[go/pfam/kegg] tables

Although it is recommended to have all the annotation are based on transcript IDs, sometimes we may have annotation with gene IDs. Following example will show you how to load gene ID based annotation files into GenIE-CMS database.

Load data into gene_[go/pfam/kegg] tables

```
#Let's assume, if we have annotation file similar to following example.
Potra000001g00001   GO:0008565      protein transporter activity
```

(continues on next page)

(continued from previous page)

```
Potra000001g00001   GO:0031204      posttranslational protein targeting to membrane,␣
↪translocation
Potra000002g00006   GO:0005634      nucleus
Potra000002g00005   GO:0003677      DNA binding
Potra000002g00005   GO:0003824      catalytic activity
Potra000002g00006   GO:0015031      protein transport
Potra000002g00006   GO:0006457      protein folding
Potra000001g00002   GO:0003852      2-isopropylmalate synthase activity
Potra000001g00002   GO:0009098      leucine biosynthetic process
Potra000002g00008   GO:0008312      7S RNA binding
```

As you see in the above example, one gene ID associated with several Gene ontology IDs. Therfore, we need to format the above results into the right format. Following `parse.py` script can be used. Now we need to create MySQL Annotation table to load GO results.

```
#!/usr/bin/env python
#parse.py
def parse(file, store):
        f = open(file, 'r')
        dic = {}
        for i in f:
                i = i.strip("\n")
                val = i.split("\t")
                try:
                    if(val[1]!=""):
                        dic[val[0]] = dic[val[0]] + ";"+ val[1]+"-"+val[2]
                except KeyError:
                    if(val[0]!=""):
                        dic[val[0]]=val[1]+"-"+val[2]
        f.close()
        f = open(store, 'w')
        for i in dic.keys():
                string = i+"\t"+dic[i]+"\t0"
                f.write(string+"\n")
        f.close

if __name__=="__main__":
        import sys
        if len(sys.argv) > 1:
                file = sys.argv[1]
                store = sys.argv[2]
                parse(file, store)
        else:
                sys.exit("No input")
```

Then the output will be similar to following.

```
Potra000001g00001       GO:0008565-protein transporter activity;GO:0031204-
↪posttranslational protein targeting to membrane, translocation 0
Potra000001g00002       GO:0003852-2-isopropylmalate synthase activity;GO:0009098-
↪leucine biosynthetic process  0
Potra000002g00005       GO:0003677-DNA binding;GO:0003824-catalytic activity  0
Potra000002g00008       GO:0008312-7S RNA binding 0
Potra000002g00006       GO:0005634-nucleus  0
Potra000002g00006       GO:0015031-protein transport;GO:0006457-protein folding 0
```

Now we need to create a table to load newly generated annotation data.

```
#Create gene_go table
CREATE TABLE `gene_go` (
  `gene_id` varchar(60) NOT NULL,
  `go_description` varchar(2000) DEFAULT NULL,
  `gene_i` mediumint(16) unsigned DEFAULT '0',
  PRIMARY KEY (`gene_id`),
  KEY `gene_id` (`gene_id`)
);


#We will load above file into following table.
mysql> explain gene_go;
+----------------+-------------------------+------+-----+---------+-------+
| Field          | Type                    | Null | Key | Default | Extra |
+----------------+-------------------------+------+-----+---------+-------+
| gene_id        | varchar(60)             | NO   | PRI | NULL    |       |
| go_description | varchar(2000)           | YES  |     | NULL    |       |
| gene_i         | mediumint(16) unsigned | YES  |     | 0       |       |
+----------------+-------------------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Previousy used `load_data.sh` script can be used to load go_gene results to `gene_go` table.

```
./load_data.sh gene_go gene_go.txt
```

Finally update the `gene_i` in `gene_go` table using following script.

```
#!/bin/bash

DB_USER='your_db_username'
DB_PASS='your_password'
DB='database_name'

#USAGE sh update_annotation_gene_i.sh gene_go
display_usage() {
        echo  "\nUsage:\n$0 [table_name] \n"
        }

# if less than one arguments supplied, display usage
        if [  $# -le 0 ]
        then
                display_usage
                exit 1
        fi

/usr/bin/mysql --host=localhost  --user=$DB_USER --password=$DB_PASS --local_infile=1␣
↪--database=$DB <<EOFMYSQL
UPDATE $1 INNER JOIN transcript_info on transcript_info.gene_id = $1.gene_id SET $1.
↪gene_i = transcript_info.gene_i;
EOFMYSQL
```

Run following command to update `gene_i`

```
./update_annotation_gene_i.sh gene_go
```

Installation

1. Download the genelist.zip file and unzip into plugins directory.

2. Edit database details in services/settings.php file.

Usage

Navigate to `http://[your server name]/GenIECMS/genelist`

3.2 BLAST

Implementation

PlantGenIE BLAST search is implemented using NCBI Blast (v2.2.26) and no database will be used. config.json files contains all necessary. We use PHP, JavaScript, XSL, Perl and d3js, Drupal libraries to improve Open Source GMOD Bioinformatic Software Bench server to provide a graphical user interface.

Libraries

Makesure ubuntu taskspooler and blastall properly installed into /use/bin

```
use DBI;
use Bio::Tools::GFF;
use File::Basename;
use Bio::SearchIO;
use Bio::SearchIO::Writer::HTMLResultWriter;
use Bio::SearchIO::Writer::TextResultWriter;
use Bio::SearchIO::Writer::GbrowseGFF;
use Bio::Graphics;
use Bio::FeatureIO;
use Bio::SeqFeature::Generic;
```

Installation

Download the BLAST tool plugin from here. Then place it into your `CMS/plugins/` folder.

Adding Datasets

Adding a dataset into BLAST tool we must use formatdb or makeblastdb tools. `config.json` file contains all necessary configuration parameters to add new datasets into existing BLAST tool. An example of `config.json` file looks like following:

```
{
"selecttion_box":[{"height":180,"width":400}],
"datasets":[{
                    "number": 1,
                "user_friendly_name": "A Label which appears in the Tool",
                "dataset_path":"/path/to/the/blast/indices",
                    "molecule_type":"nucleotide/protein",
                    "group_name":"Group Name"
    },{
                "number": 2,
                "user_friendly_name": "A Label which appears in the Tool",
                "dataset_path":"/path/to/the/blast/indices",
                    "molecule_type":"nucleotide/protein",
                    "group_name":"Group Name"
    }],
"default_jbrowse_dataset_directory":"Fegr20"
}
```

- number: This is an incremental unique number to identify the dataset id.

- user_friendly_name: This name will be appeared as dataset name inside the BLAST tool.
- molecule_type: This value should be either nucleotide or protein.
- group_name: Group name helps to grouping the datasets based on similarity.

3.3 Gene Information Pages

Installation

1. Download the gene.zip file and unzip into plugins directory.
2. Edit database details in services/settings.php file.
3. Edit the conf.json file, if needed to display sequence information inside the gene pages.

Usage

Navigate to `http://[your server name]/genie/gene?id=[gene id]` or `http://[your server name]/genie/transcript?id=[transcript id]`

Sequence information Sequences will be displayed under the sequence tab once we configure the config.json file.

Sequence coloring

Following script will be used to load genome gff3 file into corresponding sequence coloring table(sequence_color) in GenIE database. This feature will be shaded the genomic,transcriptomic and cds sequence regions in gene information pages.

```
#!/bin/bash
#get the gene.gff3 file and loaded into database table calles sequence_color
#Usage: sh sequence_color.sh /data/Egrandis_297_v2.0.gene.gff3

awk '/mRNA/{split($2,a,"=");sub(/ID=./,a[2]";");print $1;next}/gene/{;next}{sub(/ID=./
↪,a[2]";");print $1}' FS=\; OFS=\; $1 | awk '!/#/{print $9"\t"$1"\t"$3"\t"$4"\t"$5}'␣
↪> tmp &&
sed -i 's/five_prime_UTR/5UTR/' tmp && sed -i 's/three_prime_UTR/3UTR/' tmp  &&
/usr/bin/mysql --host=localhost --user=[user] --password=[pass] --local_infile=1 --
↪database=egrandis<<EOFMYSQL
TRUNCATE TABLE  sequence_color;
LOAD DATA LOCAL INFILE "tmp" INTO TABLE sequence_color fields terminated by '\t'␣
↪LINES TERMINATED BY '\n' ignore 0 lines;
EOFMYSQL
rm tmp
``
```

3.4 JBrowse

Installation

1. Download the jbrowse.zip file and unzip into plugins directory.
2. Edit database details in `services/settings.php` file.

Manual installation from JBrowse.org - optional

Following steps are important when you need to convert existing JBrowse into GenIE module.

1. Copy JBrowse into plugins folder

2. Copy `index.php` into jbrowse folder
3. Create menu item called jbrowse
4. Change plugins `plugins/jbrowse/main.css` and `plugins/jbrowse/genome.css`
5. Copy `pugins/jbrowse/index.html` into `plugins/jbrowse/tool.php` from jbrowse.zip
6. Copy `plugins/jbrowse/src/dojo/dojo.css` from jbrowse.zip
7. Copy `plugins/jbrowse/src/dijit/theme/tundra/tundra.css` from jbrowse.zip

Loading data into JBrowse

```
bin/prepare-refseqs.pl --fasta ../../data/Egrandis297v2.0.fa
bin/flatfile-to-json.pl --gff ../../data/Egrandis297v2.0.gene.gff3 --trackLabel E.
↪Genes --trackType CanvasFeatures
bin/generate-names.pl -v
```

Usage

Navigate to `http://[your server name]/genie/jbrowse`

For more information please go to JBrowse documentation

3.5 How to create a plugin?

How to create a plugin

GenIECMS plugin can start as a simple file with a PHP function. All plugins are being installed in `/GenIECMS/plugins`. The only requirement for a plugin is that the foldername has to be the same as the menu name and `index.php php` file should be available inside the plugin folder.

```
/GenIECMS/plugins/{pluginname}/index.php
/GenIECMS/plugins/{pluginname}/tool.php
```

Hello World! Plugin

```
/GenIECMS/plugins/hello/tool.php
```

1. Creat hello directory inside the plugin directory
2. Place following index.php file inside hello directory

```
<?php
//index.php
$subdir_arr = explode("/", $_SERVER['REDIRECT_URL']);
$mennu_arr = explode("<br />", $c['menu']);
$menu_exist = false;
for ($search_num = 0; $search_num < count($mennu_arr); $search_num++) {
    if (trim(strtolower($mennu_arr[$search_num])) == strtolower($subdir_arr[count(
↪$subdir_arr) - 1]) ||
    trim(strtolower($mennu_arr[$search_num])) == "-".strtolower($subdir_arr[count(
↪$subdir_arr) - 1])) {
        $menu_exist = true;
    }
}
if(strtolower(basename(dirname(__FILE__)))== strtolower($subdir_arr[count($subdir_
↪arr)-1]) && $menu_exist==true){
```

(continues on next page)

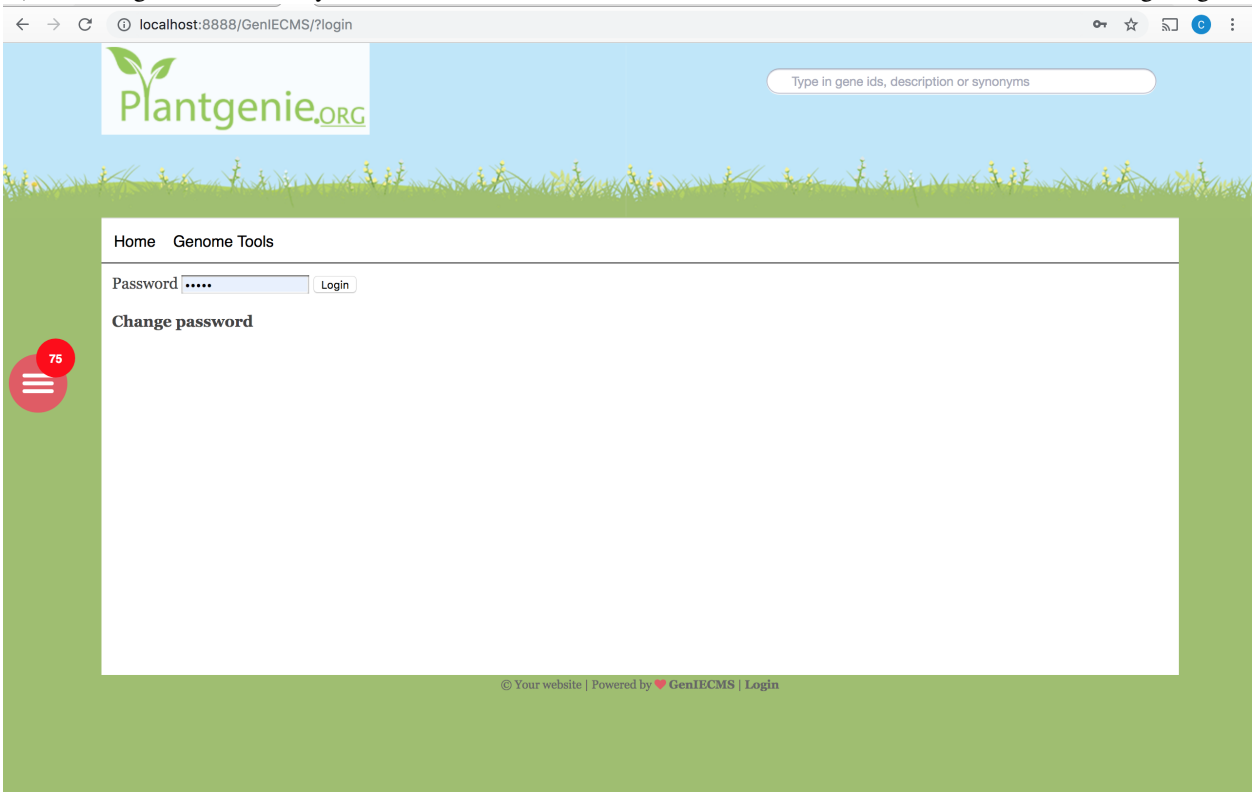
(continued from previous page)

```
    $c['initialize_tool_plugin'] = true;
    $c['tool_plugin'] = strtolower($subdir_arr[count($subdir_arr) - 1]);
}
?>
```

3.) Add tool.php into the hello_world directory

```
<?php
//tool.php
echo "Hello World!";
?>
```

4.) Log into the system and add hello into the menu like shown in following figure.



4.) Navigate to `http://[server name]/GenIECMS/hello`

CHAPTER 4

Indices and tables

- genindex
- search