GenIE-Sys Documentation

Release 0.1

Chanaka Mannapperuma

Oct 16, 2020

Administrator Documentation

CHAPTER 1

Getting Started

1.1 Overview of GenIE-Sys

The Genome Integrative Explorer System (GenIE-Sys) is dedicative in-house system 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-Sys has its basic content stored in text files. MySQL database server is required to load the genomic data and integrate with GenIE-Sys plugins.

GenIE-Sys 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-Sys 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-Sys 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-Sys 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.

1.2 Requirements

We recommended using Firefox or Google Chrome web browsers for testing of the GenIE-Sys. Please do not use Microsoft Edge or Internet Explorer.

To run GenIE-Sys we recommend your host supports the following

- PHP version 5.4+
- MySQL version 5.6+ or MariaDB version 10.0+

To fulfil the above requirements, we have tested the GenIE-Sys under the following infrastructures.

- MAMP/LAMP
- Apache, PHP and MySQL standalone servers
- Docker (recommended for development purpose)

CHAPTER 2

Installation & Updates

2.1 Download the GenIE-Sys

You can download the latest version of GenIE-Sys by using the official download link:

Please note that the above link will onlu download the source code for the GenIE-Sys. 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 GenIE-Sys and parsing scripts:

```
git clone --recursive https://github.com/plantgenie/geniesys.git
```

2.2 Folder structure

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

2.3 Install with Docker

For Developers and Contricutors

```
# Please comment the supporting_files/run.sh line to avoid download the geniesys.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
```

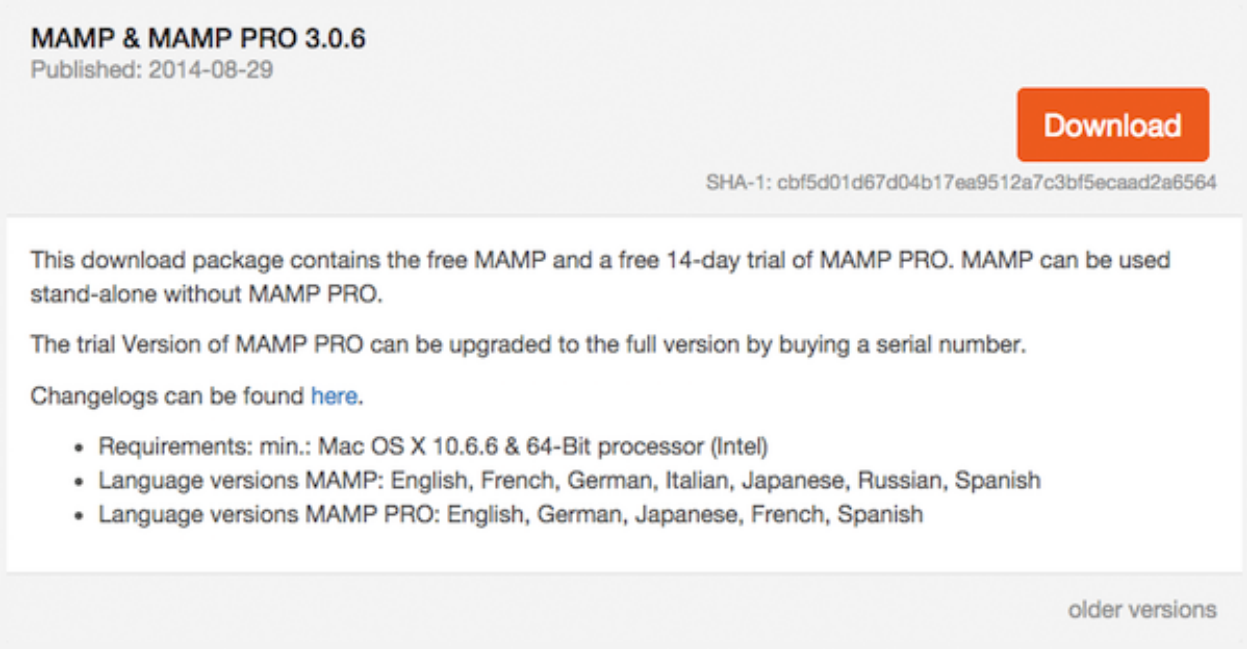
When we need to commit changes, please go to `cd docker4geniecms/genie` folder. Never commit from `docker4geniecms` folder. Because it will add genie as a submodule. Incase you mistakenly pushed from `docker4geniecms` folder, please `cd docker4geniecms` and `git rm genie`. You can access MySQL using `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
```

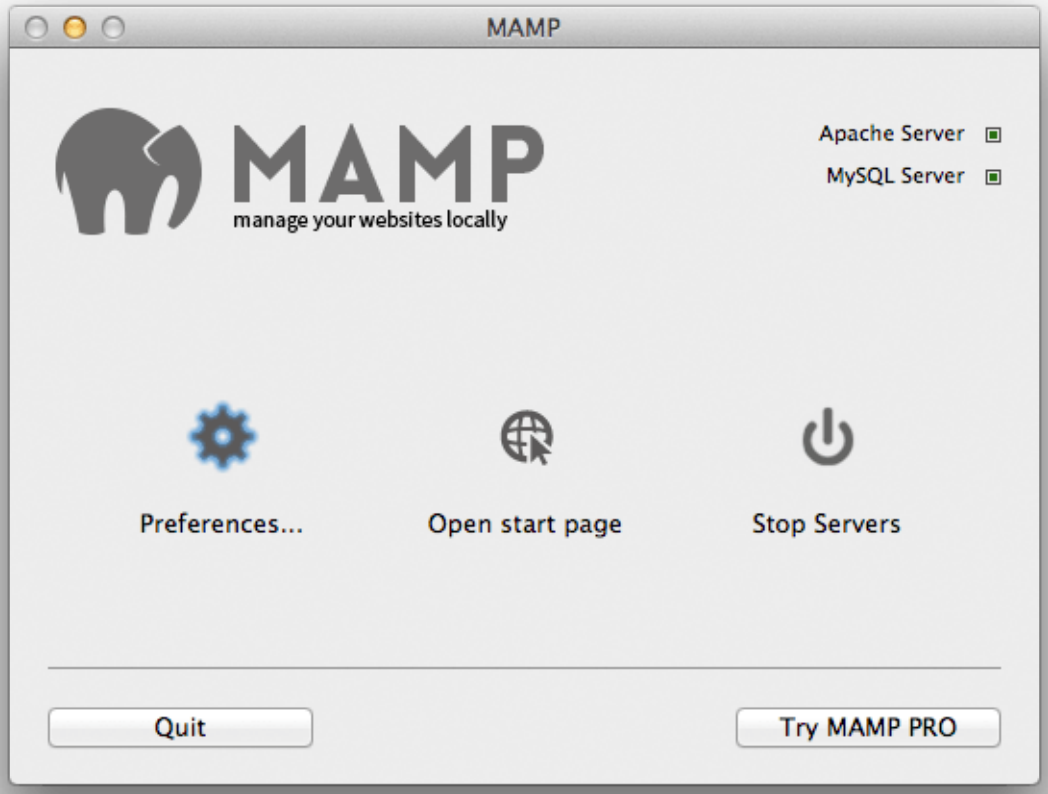
Now we can start the real development and push changes into genie.

2.4 Install with MAMP/XAMPP

Most Mac users will probably try GenIE-Sys with MAMP/LAMP.

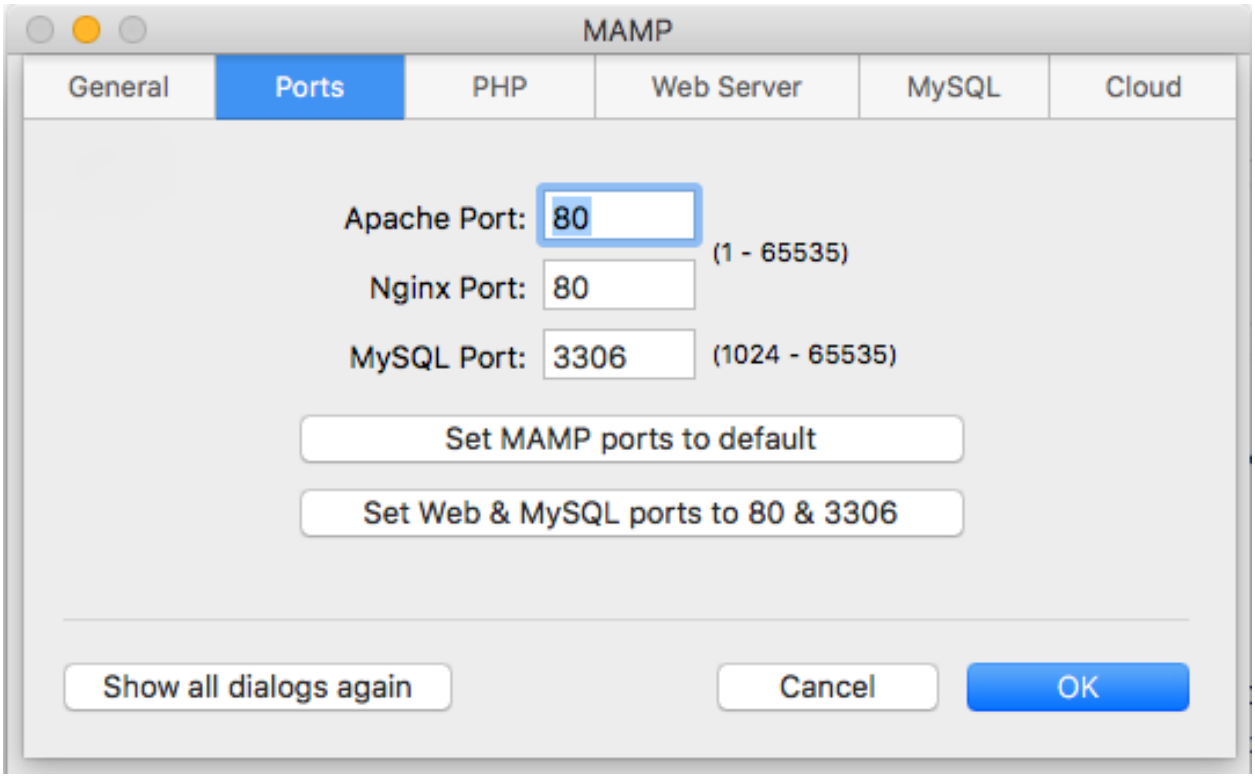


Installing MAMP is just a matter of downloading the app from the MAMP website and running the installer. It will install a MAMP app in your Applications folder.



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`.

By default, MAMP uses port 8888 for Apache and port 8889 for MySQL. It is convenient to change the MySQL and Apache ports to 3306 and 80 respectively to use default MySQL and Apache ports.



Download GenIE-Sys

Download

Copy GenIE-Sys to MAMP Web server

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

That is basically what you need to do in order to install GenIE-Sys on your Mac's local server. You should now be able to access it at: `http://localhost:[port number]/geniesys` in your browser. Now you can see the essential website up and running. However to configure it correctly please update the configuration file as described in the next section.

2.5 Configuration file

We should update the settings file(`geniesys/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:[port number]/geniesys/';

OR

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

2.6 Troubleshooting

GenIE-Sys 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 GenIE-Sys as effortless as possible. Please send us an email if you still get trouble with installation or updates: `contact@geniecms.org`

2.6.1 Running from Command Line

If you want to use PHP's built-in server (**not recommended**), just use following lines to install GenIE-Sys. 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/geniesys.git
cd geniesys
php -S localhost:3000
```

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

2.6.2 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 GenIE-Sys.:

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

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

2.6.3 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 GenIE-Sys 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 `GenIE-Sys/.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.

2.6.4 Example input files

```
#head  input/example.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
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

#head annotation_file.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
```

2.6.5 Updates

Manual updates

GenIE-Sys can be updated manually using latest ZIP file from GitHub. Please backup your older version of `geniesys/plugins/settings.php` and `geniesys/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 `geniesys/plugins/settings.php` and `geniesys/genie_files` into latest version of GenIE-Sys.

Updates using Git

Here is the easy way to update GenIE-Sys using git submodules:

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

2.6.6 More problems?

Please contact us:contact@geniecms.org

CHAPTER 3

Administration

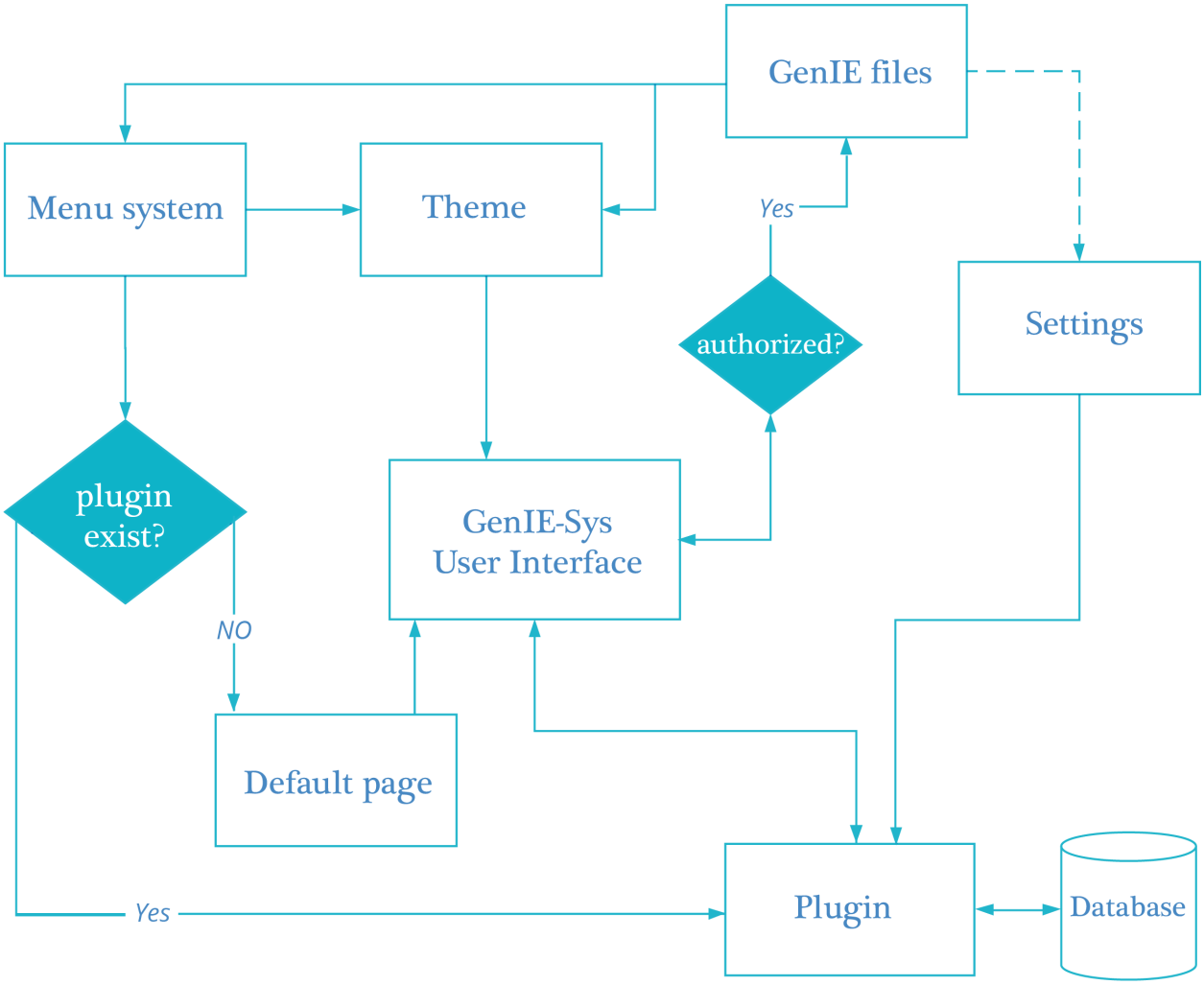
3.1 Plugin architecture

GenIE-Sys comes with a stable but yet simple-to-start plugin architecture. A plugin is a simple folder with a bunch of PHP files. This would be the simplest form:

```
site/plugins/my-plugin/index.php
```

GenIE-Sys automatically loads all plugins from folders within `/site/plugins` and tries to find an `index.php`.

Analysis, expression or genomic tools (Sundell et al., 2015) are integreated into a GenIE-Sys as external plugins. GenIE-Sys contains 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-Sys.



As shown in the above figure. Genie files can be changed by the administrator using GenIE-Sys user interface. Basic site configurations including passwords, menus and themes information stored in the GenIE files. First, check the availability of the plugin for the given menu name. If the plugin is available for the given menu name, the equivalent plugin will be displayed; otherwise, the user will only see the default blank page. Plugins will use the Database and storage files to visualize genomic, analysis and expression data. The corresponding plugin will render inside the GenIE user interface with the combination of the selected theme. The plugin will also have access to the settings files to get database name, username, password, blast directories and other similar settings. Settings have to access the configuration files to get default database name unless if it's not already mentioned in the settings files.

3.2 Database design

GenIE-Sys has its basic content stored in text files. MySQL database server is required to load the genomic data and integrate with GenIE-Sys plugins.

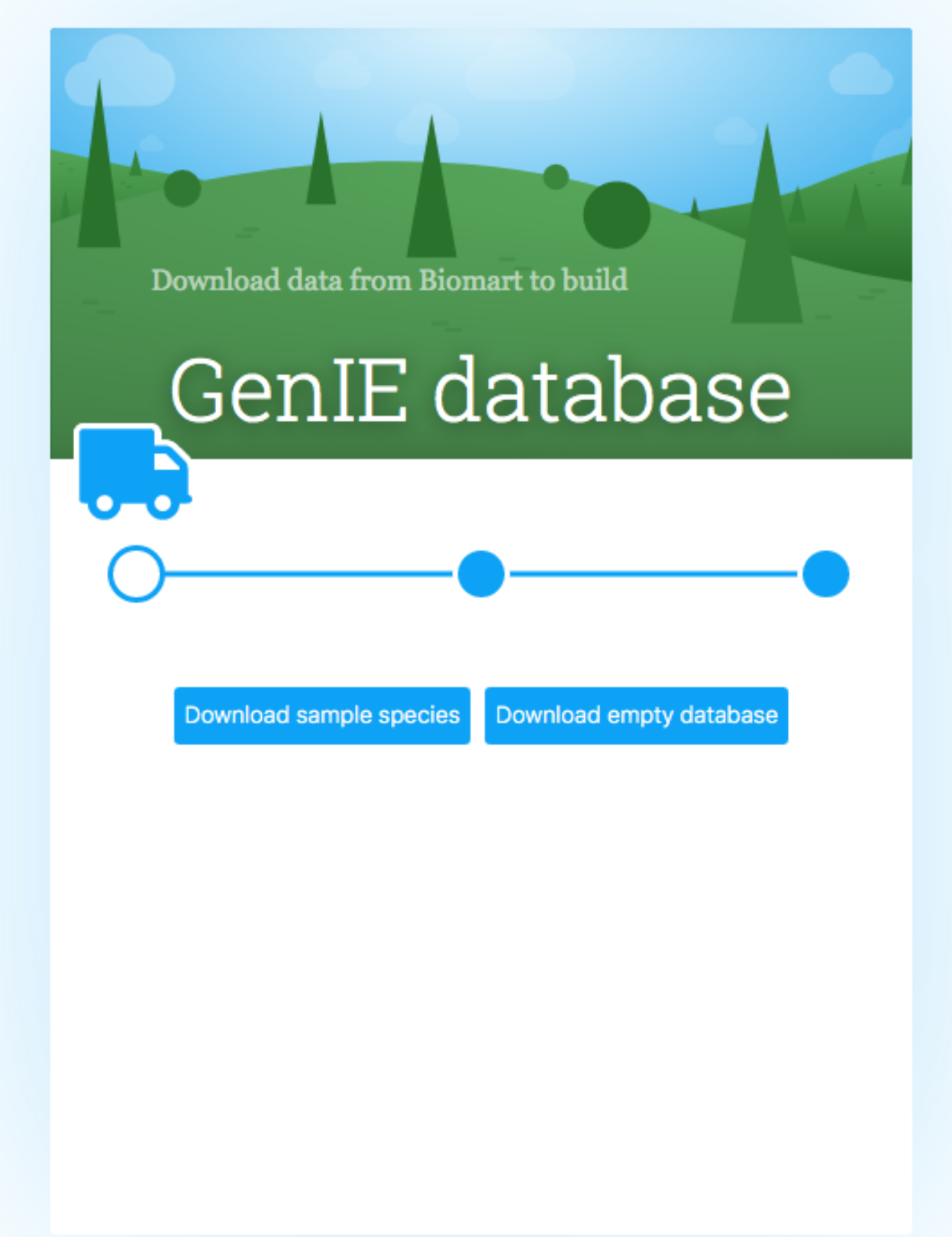
You can create a database using a graphical user interface or command line.

3.2.1 Creating a new database using GUI

Once you navigate to the Home page, you will see options to install the database. There are two options available, to begin with, database installation, as stated below.

1.) install Arabidopsis thaliana model species

2.) install an empty database.



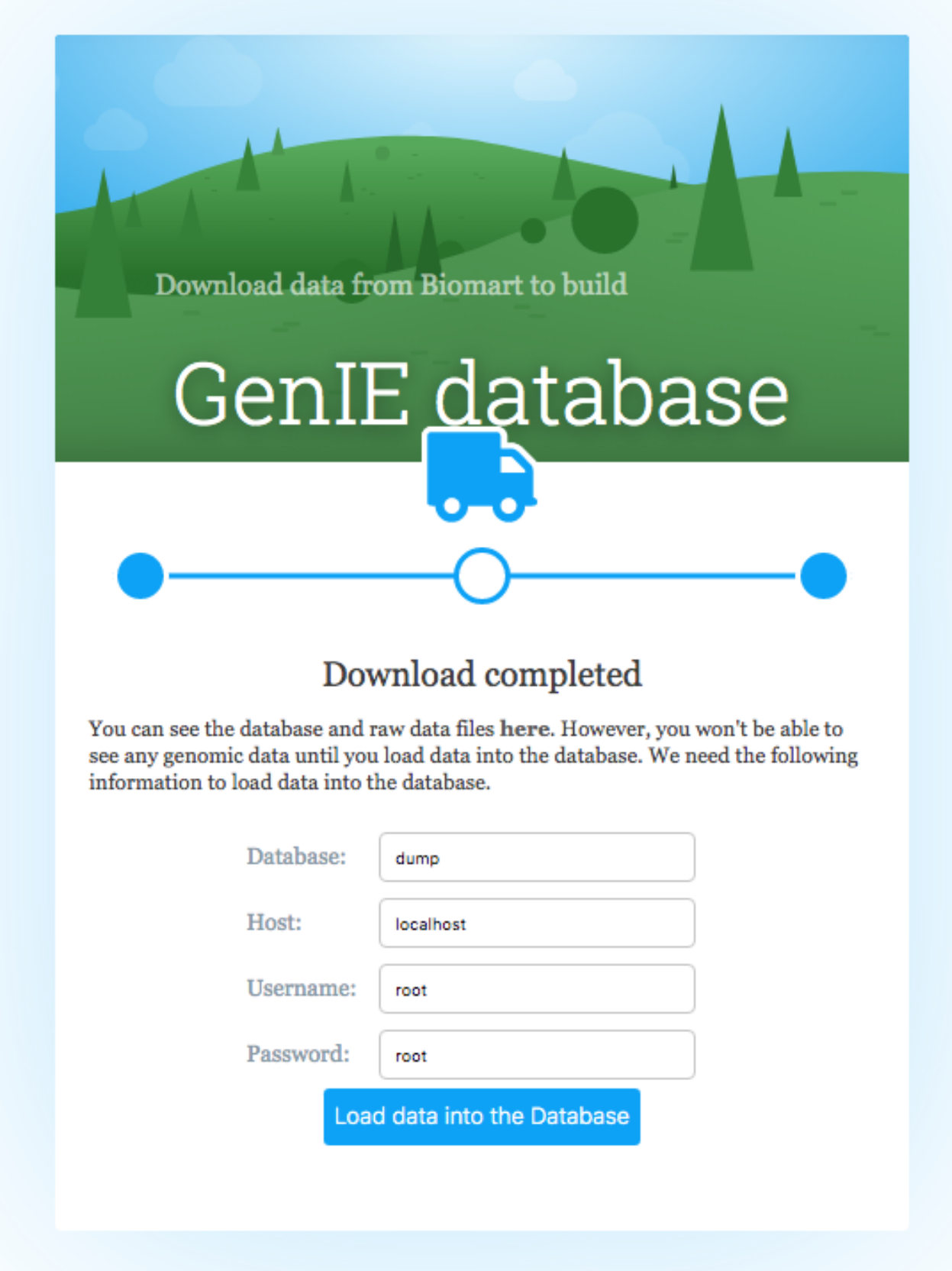
1.) install Arabidopsis thaliana model species

You need to type in the database name, MySQL host, username and password and then click the button “Load Data

into the Database."

2.) install an empty database.

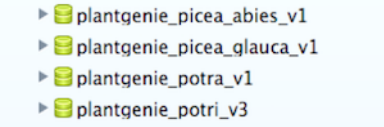
You need to type in the empty database name, MySQL host, username and password and then click the button “Load Data into the Database.”



Once the above processes are completed, you can be able to access the newly created database in MySQL server.

3.2.2 Creating a new database using CMD

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.

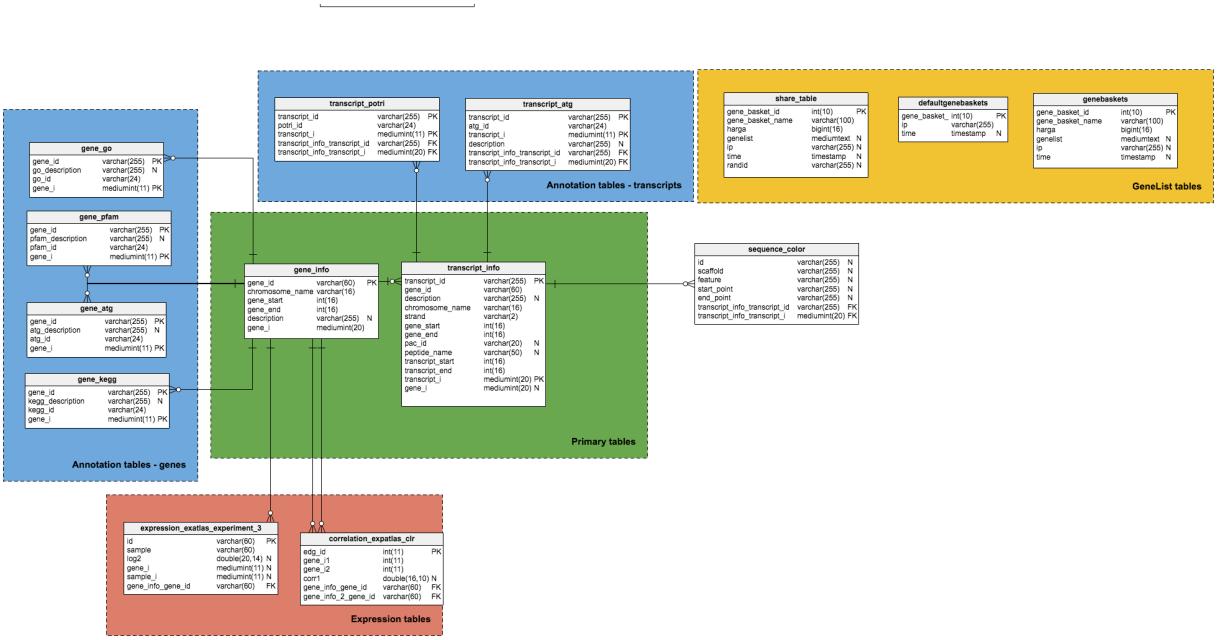
```
#Create MySQL user:
CREATE USER newuser@'localhost' IDENTIFIED BY 'newpassword';

#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
$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 geniesys 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.

Configuring genome database

All configuration settings in geniesys need to be added into `/geniesys/plugins/settings.php` file. You need to update `/geniesys/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.

3.3 Load novel genome

Here is a quick guide to describe how to load a novel genome into GenIE-Sys database.

3.3.1 Download

Let's assume we need to integrate Populus tremula v2.0 genome into GenIE-System. First, we need to download the required files. The latest version of the GFF3 and FASTA files are available on PlantGenIE FTP.

```
$ curl -O ftp://plantgenie.org/Data/PopGenIE/Populus_tremula/v2.2/gff/Potra02_genes.
↪gff.gz
$ curl -O ftp://plantgenie.org/Data/PopGenIE/Populus_tremula/v2.2/fasta/Potra02_
↪genome.fasta.gz
$ gzip -d Potra02_genes.gff.gz
$ gzip -d Potra02_genome.fasta.gz

$ awk '!/##/' Potra02_genes.gff  |head
chr1        maker        gene        8865        11259        .        -        .
↪        ID=Potra2n1c1;Name=Potra2n1c1
chr1        maker        mRNA        8865        10802        .        -        .
↪        ID=Potra2n1c1.3;Parent=Potra2n1c1;Name=Potra2n1c1.3;_AED=0.39;_eAED=0.37;_
↪QI=192|0.66|0.75|1|0|0|4|0|115
chr1        maker        CDS        8865        9054        .        -
↪        1        ID=Potra2n1c1.3:cds;Parent=Potra2n1c1.3
```

(continues on next page)

(continued from previous page)

```
chr1        maker        CDS        9487        9559        .        -
↪        2        ID=Potra2n1c1.3:cds;Parent=Potra2n1c1.3
chr1        maker        CDS        9669        9753        .        -
↪        0        ID=Potra2n1c1.3:cds;Parent=Potra2n1c1.3
chr1        maker        exon        9669        9764        .        -        .
↪        ID=Potra2n1c1.3:exon:300;Parent=Potra2n1c1.3
chr1        maker        five_prime_UTR        9754        9764        .        -
↪        .        ID=Potra2n1c1.3:five_prime_utr;Parent=Potra2n1c1.3
chr1        maker        exon        10622        10802        .        -        .
↪        ID=Potra2n1c1.3:exon:299;Parent=Potra2n1c1.3
chr1        maker        five_prime_UTR        10622        10802        .        -
↪        .        ID=Potra2n1c1.3:five_prime_utr;Parent=Potra2n1c1.3
chr1        maker        mRNA        8865        11259        .        -        .
↪        ID=Potra2n1c1.1;Parent=Potra2n1c1;Name=Potra2n1c1.1;_AED=0.22;_eAED=0.21;_
↪QI=896|0.66|0.75|1|0|0|4|0|115

$ head Potra02_genome.fasta
>chr1
AGAGAGCTCTGTGGGTCATTACTGTCACAACTCCTAGCCAGCTTGAATAT
TCCATATAGCACATATCCTGGATGGGAAAGTTTGGTTAATGTGTGCTATT
CTTGCTCGCCTTCAACACGATTATTTCGTTCATACCACAAGAAATAAACA
GTAGTGGATAGTAGAAGGCGAGCTAGCATGTGATCACTGTTATTCTTCTT
CGTGTAGTGAGTGACTGACCAATGAAGCAATTGTGTCCACGGTTTGCATG
GCCAATAATGGTTGGCTCTGCGACAAATGGACTTCCAAACCAAGCTGGTG
TAACTGCATTCAAAAAAGAGGTGTTCATATGTTTCCATGTAAATTCCATA
TAGTATGCAAGTTGTATCTGTGACTCCTCCATGCAATCTATCCATCGTTC
TTAGTCTACCAAGGCTGGCTAACCAGAGTATAAATGAGTGACGAGGGATA
```

3.3.2 Parse genome

Now we need to parse GFF3 and FASTA files into required formats. There are two primary tables(transcript_info and gene_info) in the database.

```
## generate file for gene_info table
awk '/gene/{split($9,a,"ID=");split(a[2],b,";");print b[1],$1,$4,$5,$7}' FS='\t' OFS=
↪'\t' Potra02_genes.gff > gene_info.txt
$ head gene_info.txt
Potra2n1c1        chr1        8865        11259        -
Potra2n1c2        chr1        21121        21603        +
Potra2n1c3        chr1        22295        24697        -
Potra2n1c4        chr1        30731        32811        +
Potra2n1c5        chr1        33508        33833        +
Potra2n1c6        chr1        50823        54726        -
Potra2n1c7        chr1        50901        51116        +
Potra2n1c8        chr1        54928        62450        -
Potra2n1c9        chr1        69471        73884        -
Potra2n1c10        chr1        74717        75583        +

## create file for transcript_info table
awk 'BEGIN{ OFS = "\t"; }$3~/gene/{g=$4"\t"$5}$3~/RNA$/{split($9,a,/[;=]/);for(i=1;i␣
↪in a;i+=2)k[a[i]]=a[i+1]; print k["Name"], k["Parent"], "desc", $1, $7, g, "PAC",
↪"PEP", $4,$5}' Potra02_genes.gff > transcript_info.txt
$ head transcript_info.txt
Potra2n1c1.3        Potra2n1c1        desc        chr1        -
↪        8865        11259        PAC        PEP        8865        10802
```

(continues on next page)

(continued from previous page)

```
Potra2n1c1.1        Potra2n1c1        desc        chr1        -
↪        8865        11259        PAC        PEP        8865        11259
Potra2n1c1.2        Potra2n1c1        desc        chr1        -
↪        8865        11259        PAC        PEP        8865        11259
Potra2n1c2.
↪1        Potra2n1c2        desc        chr1        +        21121        21603        PAC        PE
Potra2n1c3.1        Potra2n1c3        desc        chr1        -
↪        22295        24697        PAC        PEP        22295        24697
Potra2n1c4.
↪1        Potra2n1c4        desc        chr1        +        30731        32811        PAC        PE
Potra2n1c5.
↪1        Potra2n1c5        desc        chr1        +        33508        33833        PAC        PE
Potra2n1c6.1        Potra2n1c6        desc        chr1        -
↪        50823        54726        PAC        PEP        50823        54726
Potra2n1c7.
↪1        Potra2n1c7        desc        chr1        +        50901        51116        PAC        PE
Potra2n1c8.3        Potra2n1c8        desc        chr1        -
↪        54928        62450        PAC        PEP        54928        61609
```

3.3.3 Create a database

Now we need to create a database. To do this, you need a MySQL username and password. If you use the MAMP installation default username and password would be `root`.

```
## Download all required scripts and dump database
$ git clone https://github.com/irusri/scripts.git
## Create database for default root user and root password
$ mysql -u root -proot
mysql> create database my_genie_sys_database;
Query OK, 1 row affected (0.01 sec)
mysql> use my_genie_sys_database;
Database changed
mysql> source scripts/dump.sql;
```

3.3.4 Loading primary tables

Now we need to load above two files(gene_info.txt and transcript_info.txt) into the newly created database. There is a script(`load_data.sh`) to do this. We can download the script and enter the correct `username, password and database` information to `DB_USER, DB_PASS and DB` parameters respectively.

```
$ nano scripts/load_data.sh
#load_data.sh script
#!/bin/bash
#load_data.sh
#USAGE: sh load_data.sh [table_name] [filename]
#sh load_data.sh transcript_info_x /tmp/transcript_info.tsv

DB_USER='root' #'your_db_username'
DB_PASS='root' #'your_password'
DB='my_genie_sys_database' #'database_name'

mysql  --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 --
↪database=$DB <<EOFMYSQL
```

(continues on next page)

(continued from previous page)

```
TRUNCATE TABLE $1;
ALTER TABLE $1 AUTO_INCREMENT = 1;
load data local infile '$2' ignore INTO TABLE $1 CHARACTER SET UTF8 fields terminated␣
↪by '\t' LINES TERMINATED BY '\n' ignore 0 lines;
EOFMYSQL
```

Run following commands to load `gene_info.txt` and `transcript_info.txt` into respective tables.

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

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

Now we need to update the `gene_i` parameter in `transcript_info` table. There is a script(`update_gene_i.sh`) in the scripts directory, we just need to enter the correct `username, password and database` information to `DB_USER, DB_PASS and DB` parameters respectively as we did in previous step.

```
$ nano scripts/update_gene_i.sh
#update_gene_i.sh script
#!/bin/bash
#update_gene_i.sh

DB_USER='root' #'your_db_username'
DB_PASS='root' #'your_password'
DB='my_genie_sys_database' #'database_name'

#USAGE: sh update_gene_i.sh

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
```

Let's run the following command to update `gene_i` in `transcript_info` table.

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

If above script takes time please try following command on MySQL. This will update the `gene_i` column in `transcript_info` table.

```
$ nano scripts/update_gene_i.sh
#update_gene_i_dev.sh script
#!/bin/bash
#update_gene_i_dev.sh

DB_USER='root' #'your_db_username'
DB_PASS='root' #'your_password'
DB='my_genie_sys_database' #'database_name'
```

(continues on next page)

(continued from previous page)

```
#USAGE: sh update_gene_i_dev.sh

mysql --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 --
↪database=$DB <<EOFMYSQL
update transcript_info,gene_info set transcript_info.gene_i=gene_info.gene_i where␣
↪gene_info.gene_id=transcript_info.gene_id;
EOFMYSQL
```

Run following command to execute the above script(`update_gene_i_dev.sh`)

```
sh scripts/update_gene_i_dev.sh
```

Great! We have loaded transcript and gene infortmation properly into the database. Now can we load additional information. For example; description to the `transcript_info` table.

```
$ curl -O ftp://plantgenie.org/Data/PopGenIE/Populus_tremula/v2.2/annotation/blast2go/
↪Potra22_blast2go_description.txt
$ head Potra22_blast2go_description.txt
Potra2n765s36715.1        UniRef90_B9GWJ3F-box domain-containing protein n=2␣
↪Tax=Populus TaxID=3689 RepID=B9GWJ3_POPTR
Potra2n765s36713.1        Populus trichocarpa uncharacterized LOC112326797␣
↪(LOC112326797), ncRNA
Potra2n765s36713.2        Populus trichocarpa uncharacterized LOC112326797␣
↪(LOC112326797), ncRNA
Potra2n765s36714.1        UniRef90_A0A2K2BA33FAD-binding PCMH-type domain-containing␣
↪protein n=40 Tax=Populus TaxID=3689 RepID=A0A2K2BA33_POPTR
Potra2n1433s37070.1        UniRef90_U7E173Protein kinase domain-containing protein␣
↪(Fragment) n=1 Tax=Populus trichocarpa TaxID=3694 RepID=U7E173_POPTR
Potra2n581s36023.1        UniRef90_UPI00057ABC08probable LRR receptor-like serine/
↪threonine-protein kinase At4g08850 isoform X1 n=1 Tax=Populus euphratica␣
↪TaxID=75702 RepID=UPI00057ABC08
Potra2n581s36025.1        UniRef90_UPI00057B3C83probable LRR receptor-like serine/
↪threonine-protein kinase At4g08850 n=1 Tax=Populus euphratica TaxID=75702␣
↪RepID=UPI00057B3C83
Potra2n581s36024.1        UniRef90_U5GE99Zeta-carotene desaturase n=10 Tax=fabids␣
↪TaxID=91835 RepID=U5GE99_POPTR
Potra2n707s36547.1        UniRef90_A0A2K1X8T3AMPKBI domain-containing protein n=5␣
↪Tax=Populus TaxID=3689 RepID=A0A2K1X8T3_POPTR
Potra2n409s35556.1        UniRef90_UPI000B5D6D9FE3 ubiquitin-protein ligase SHPRH␣
↪isoform X3 n=1 Tax=Manihot esculenta TaxID=3983 RepID=UPI000B5D6D9F
```

Here is the script to load `description` into `transcript_info` column.

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

DB_USER='root' #'your_db_username'
DB_PASS='root' #'your_password'
DB='my_genie_sys_database' #'database_name'

# if less than two arguments supplied, display  error message
        if [  $# -le 0 ]
        then
                start='\033[0;33m'
                start_0='\033[0;33m'
```

(continues on next page)

(continued from previous page)

```
                start_2='\033[0;31m'
                end='\033[0m'
                echo  "\nUsage:\n$0 ${start}[gene_info/transcript_info] [file_name]$
↪{end}\nEx: ${start_2}sh update_description.sh transcript_info/gene_info potra_
↪description.tsv${end}\n\nWhat it does?\n${start_0}This script will create a two␣
↪columns(ids, description) temporary table and load the [file_name] into it.\nThen␣
↪it will match ids column in temporary table with transcript_ids/gene_ids and update␣
↪the gene/transcript description.\nFinally delete the temporary table.\n${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 just need to run the script to load `description` into `transcript_info` table.

```
sh scripts/update_description.sh transcript_info Potra22_blast2go_description.txt
```

3.3.5 Loading secondary tables

Following are the tables available with GenIE-Sys default database. However, it is easy to add more tables depending on the user demands. Secondary table conatins annotation related to the primary tables.

```
$ curl -O ftp://plantgenie.org/Data/PopGenIE/Populus_tremula/v2.2/annotation/blast2go/
↪Potra22_blast2go_GO.txt
$ head Potra22_blast2go_GO.txt
Sequence Name        Annotation GO ID-Annotation GO Term
Potra2n765s36715.1        GO:0005515-protein binding
Potra2n765s36714.1        GO:0009690-cytokinin metabolic process
Potra2n765s36714.1        GO:0016021-integral component of membrane
Potra2n765s36714.1        GO:0019139-cytokinin dehydrogenase activity
Potra2n765s36714.1        GO:0055114-oxidation-reduction process
Potra2n765s36714.1        GO:0071949-FAD binding
Potra2n1433s37070.1        GO:0004674-protein serine/threonine kinase activity
Potra2n1433s37070.1        GO:0005509-calcium ion binding
Potra2n1433s37070.1        GO:0005524-ATP binding

$ awk 'BEGIN{FS="\t";OFS="\t"}{a[$1]=a[$1]?a[$1]";"$2:$2;}END{for (i in a)print i"\t
↪"a[i];}'   Potra22_blast2go_GO.txt > Potrav22_go_desc.txt
$ head Potrav22_go_desc.txt
Potra2n5c11384.4        GO:0019904-protein domain specific binding
Potra2n1c2900.1        GO:0006118-obsolete electron transport;GO:0009055-electron␣
↪transfer activity;GO:0016021-integral component of membrane;GO:0022900-electron␣
↪transport chain
Potra2n12c24161.1        GO:0005789-endoplasmic reticulum membrane;GO:0016021-
↪integral component of membrane
```

(continues on next page)

(continued from previous page)

```
Potra2n6c13118.1        GO:0003677-DNA binding;GO:0004724-magnesium-dependent protein␣
↪serine/threonine phosphatase activity;GO:0005963-magnesium-dependent protein serine/
↪threonine phosphatase complex;GO:0006470-protein dephosphorylation;GO:0046872-metal␣
↪ion binding
Potra2n6c13118.2        GO:0003677-DNA binding;GO:0004724-magnesium-dependent protein␣
↪serine/threonine phosphatase activity;GO:0005963-magnesium-dependent protein serine/
↪threonine phosphatase complex;GO:0006470-protein dephosphorylation;GO:0046872-metal␣
↪ion binding
Potra2n6c13118.3        GO:0003677-DNA binding;GO:0004724-magnesium-dependent protein␣
↪serine/threonine phosphatase activity;GO:0005963-magnesium-dependent protein serine/
↪threonine phosphatase complex;GO:0006470-protein dephosphorylation;GO:0046872-metal␣
↪ion binding
Potra2n9c19679.1        GO:0016021-integral component of membrane;GO:0016117-
↪carotenoid biosynthetic process;GO:0016166-phytoene dehydrogenase activity;
↪GO:0016757-transferase activity, transferring glycosyl groups;GO:0055114-oxidation-
↪reduction process
Potra2n14c27340.1        GO:0046872-metal ion binding
Potra2n9c19679.2        GO:0016021-integral component of membrane;GO:0016117-
↪carotenoid biosynthetic process;GO:0016166-phytoene dehydrogenase activity;
↪GO:0016757-transferase activity, transferring glycosyl groups;GO:0055114-oxidation-
↪reduction process
Potra2n14c27340.2        GO:0046872-metal ion binding
```

As you see the annotation are based on transcript IDs. Therefore, Following script can be used to load secondary table into `transcript_go` table. Then update `transcript_i` column using another script as described below.

```
sh scripts/load_data.sh transcript_go Potrav22_go_desc.txt
```

Then update the `gene_i` or `transcript_i` depending on the primary usint of the annotation dataset using following script.

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

DB_USER='root' #'your_db_username'
DB_PASS='root' #'your_password'
DB='my_genie_sys_database' #'database_name'

#USAGE sh update_annotation_gene_i.sh transcript_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

count=$(mysql --host=localhost --user=$DB_USER --password=$DB_PASS --database=$DB -
↪sse "SHOW COLUMNS FROM $1 LIKE 'transcript_id';")
if [ ${#count} -gt 0 ]
then
     mysql --host=localhost --user=$DB_USER --password=$DB_PASS --local_infile=1 --
↪database=$DB <<EOFMYSQL
```

(continues on next page)

(continued from previous page)

```
UPDATE $1 INNER JOIN transcript_info on transcript_info.transcript_id = $1.transcript_
↪id SET $1.transcript_i = transcript_info.transcript_i;
EOFMYSQL
else
     mysql --host=localhost --user=$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
fi
```

Let's run following command to fill the `transcript_i` or `gene_i` column.

```
sh scripts/update_annotation_gene_i.sh transcript_go
```

Similalrly when we have annotation based on gene IDs, we have to fill gene_`annotation` tables.

You may also load additional annotation as secondary tables to the GenIE-Sys database. If there is a transcript-based annotation, please use the following script to create a corresponding table (please replace `annotation` with the name of the annotation).

```
-- ----------------------------
--  Table structure for `transcript_annotation`
-- ----------------------------
DROP TABLE IF EXISTS `transcript_annotation`;
CREATE TABLE `transcript_annotation` (
  `transcript_id` varchar(255) NOT NULL,
  `annotation_description` varchar(1000)  DEFAULT '' NOT NULL,
  `transcript_i` mediumint(20) unsigned DEFAULT 0 NOT NULL,
  PRIMARY KEY (`transcript_i`,`transcript_id`),
) ENGINE=MyISAM DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

If there is a gene-based annotation, please use the following script to create a corresponding table (please replace `annotation` with the name of the annotation).

```
-- ----------------------------
--  Table structure for `gene_annotation`
-- ----------------------------
DROP TABLE IF EXISTS `gene_annotation`;
CREATE TABLE `gene_annotation` (
  `gene_id` varchar(255) NOT NULL,
  `annotation_description` varchar(1000)  DEFAULT '' NOT NULL,
  `gene_i` mediumint(20) unsigned DEFAULT 0 NOT NULL,
  PRIMARY KEY (`gene_i`,`gene_id`),
) ENGINE=MyISAM DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

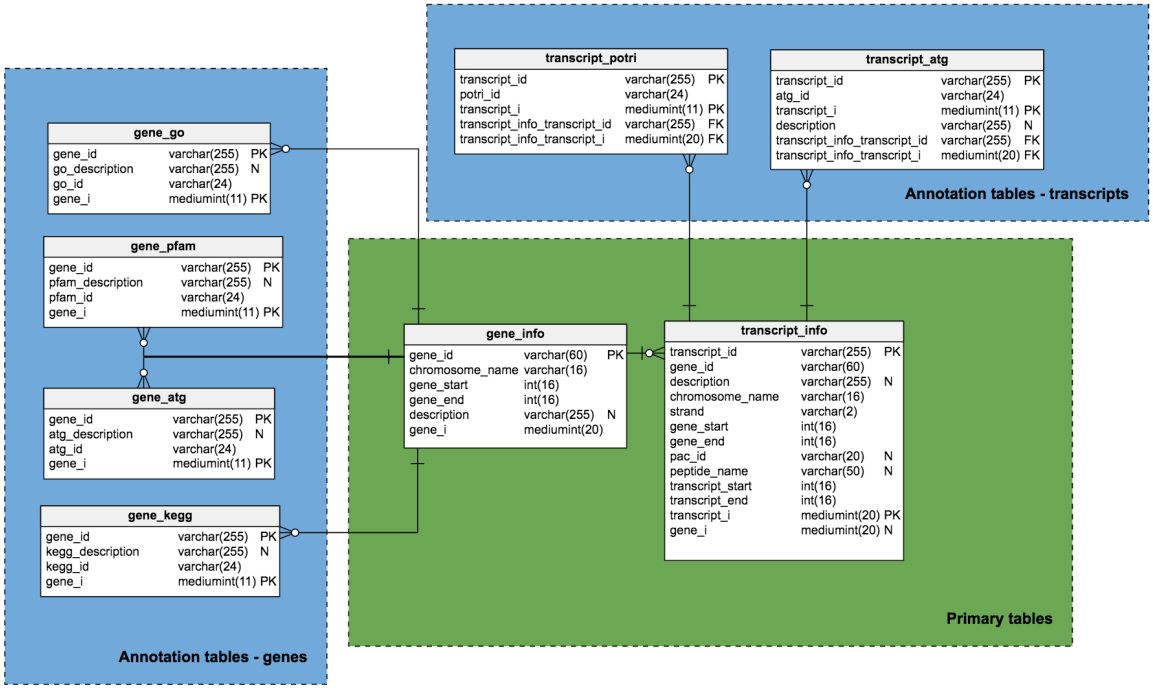
Finally you may need to add the new annotation into `/plugins/genelist/genelist/service/config.php` to make it searchable in the GeneSearch tool.

3.4 GeneList

Overview

GeneList is the heart of the GenIE-Sys; this will be the entry point to many of the tools and workflows. Foundation to entire GenIE-Sys 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 GenIE-Sys 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 geniesys/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-Sys.

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`)
```

(continues on next page)

(continued from previous page)

```
);
#Describe transcript_info table
mysql> explain transcript_info;
+-----------------+-------------------------+------+-----+---------+----------------+
| 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
```

(continues on next page)

(continued from previous page)

```
Potra000001        leafV2        mRNA        9066        10255        .        -
↪        .        ID=Potra000001g00001.1;Parent=Potra000001g00001;
↪Name=Potra000001g00001;cdsMD5=71c5f03f2dd2ad2e0e00b15ebe21b14c;primary=TRUE
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 '{if(g3=="gene"){split($9,a,"=");split(a[2],b,";");split(g9,ga,"=");split(ga[2],
↪gb,";");print b[1]"\t"gb[1]"\tDesc\t"$1"\t"$7"\t"g4"\t"g5"\tPAC\tPEP\t"$4"\t"$5};g3=
↪$3;g1=$1;g2=$2;g4=$4;g5=$5;g9=$9}' input/Potra01-gene-mRNA-wo-intron.gff3 > input/
↪transcript_info.txt

#results file(transcript_info.txt) looks like following
Potra000001g00001.1        Potra000001g00001        desc        Potra000001        -
↪        9066        10255        9066        10255
Potra000001g00002.
↪1        Potra000001g00002        desc        Potra000001        +        13567        14931
Potra000002g00003.
↪1        Potra000002g00003        desc        Potra000002        +        6593        10325
```

(continues on next page)

(continued from previous page)

```
Potra000002g00003.
↪2        Potra000002g00003        desc        Potra000002        +        6593        10325
Potra000002g00003.
↪3        Potra000002g00003        desc        Potra000002        +        7874        8369
Potra000002g00004.2        Potra000002g00004        desc        Potra000002        -
↪        8395        12794                        8395        12794
Potra000002g00004.3        Potra000002g00004        desc        Potra000002        -
↪        9033        12733                        9033        12733
Potra000002g00004.1        Potra000002g00004        desc        Potra000002        -
↪        9033        12733                        9033        12733
Potra000002g35060.1        Potra000002g35060        desc        Potra000002        -
↪        10226        12730                        10226        12730
Potra000002g00005.3        Potra000002g00005        desc        Potra000002        -
↪        19301        21913                        19301        21913
```

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 `geniesys/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
```

(continues on next page)

(continued from previous page)

```
Potra000002g00006.5        DnaJ homolog subfamily

#head potra_gene_description.txt
Potra000001g00001        Germin-like protein subfamily 1 member
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_description.sh` in `geniesys/scripts` folder. The script looks like following.

```
#!/bin/bash
#update_description.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: ${start_2}sh update_description.sh transcript_info/gene_info potra_
↪description.tsv${end}\n\nWhat it does?\n${start_0}This script will create a two␣
↪columns(ids, description) temporary table and load the [file_name] into it.\nThen␣
↪it will match ids column in temporary table with transcript_ids/gene_ids and update␣
↪the gene/transcript description.\nFinally delete the temporary table.\n${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_description.sh` script to load description into gene_info and transcript_info tables.

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

(continues on next page)

(continued from previous page)

```
#Load transcript description
./update_description.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 geniesys/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` (
```

(continues on next page)

(continued from previous page)

```
  `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`)
);

#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-Sys 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
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
```

(continues on next page)

(continued from previous page)

```
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]/geniesys/genelist`

3.5 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` 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 `geniesys/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",
```

(continues on next page)

(continued from previous page)

```
                    "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.

3.6 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
``
```

3.7 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 [genome fasta file]
bin/flatfile-to-json.pl --gff [GFF3 file]] --trackLabel [Label name] --trackType␣
↪CanvasFeatures
bin/generate-names.pl -v
```

Usage

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

For more information please go to JBrowse documentation

3.8 How to create a plugin?

How to create a plugin

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

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

Hello World! Plugin

```
/geniesys/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]) ||
```

(continues on next page)

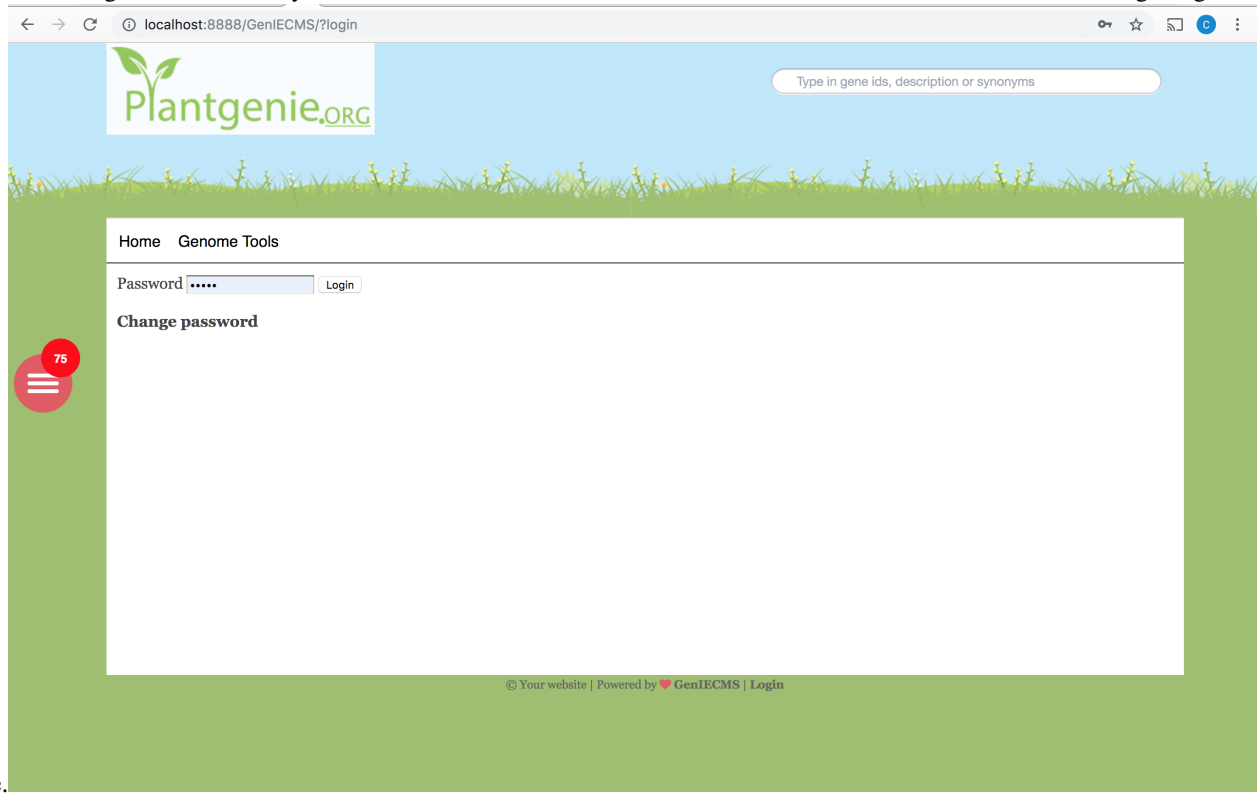
(continued from previous page)

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

3.) Add tool.php into the hello_world directory. `tool.php` file will be used to write all the function related to the plugin.

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

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



5.) Navigate to `http://[server name]/geniesys/hello`

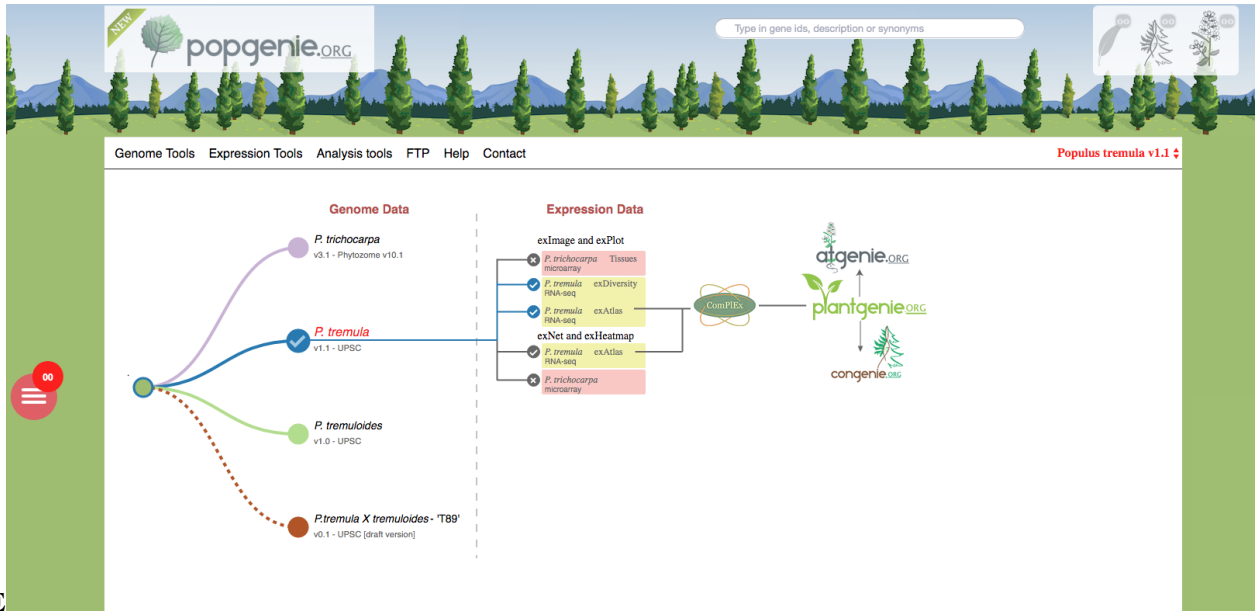
More details about the GenIE-Sys API is available on https://api.plantgenie.org

3.9 Change Theme

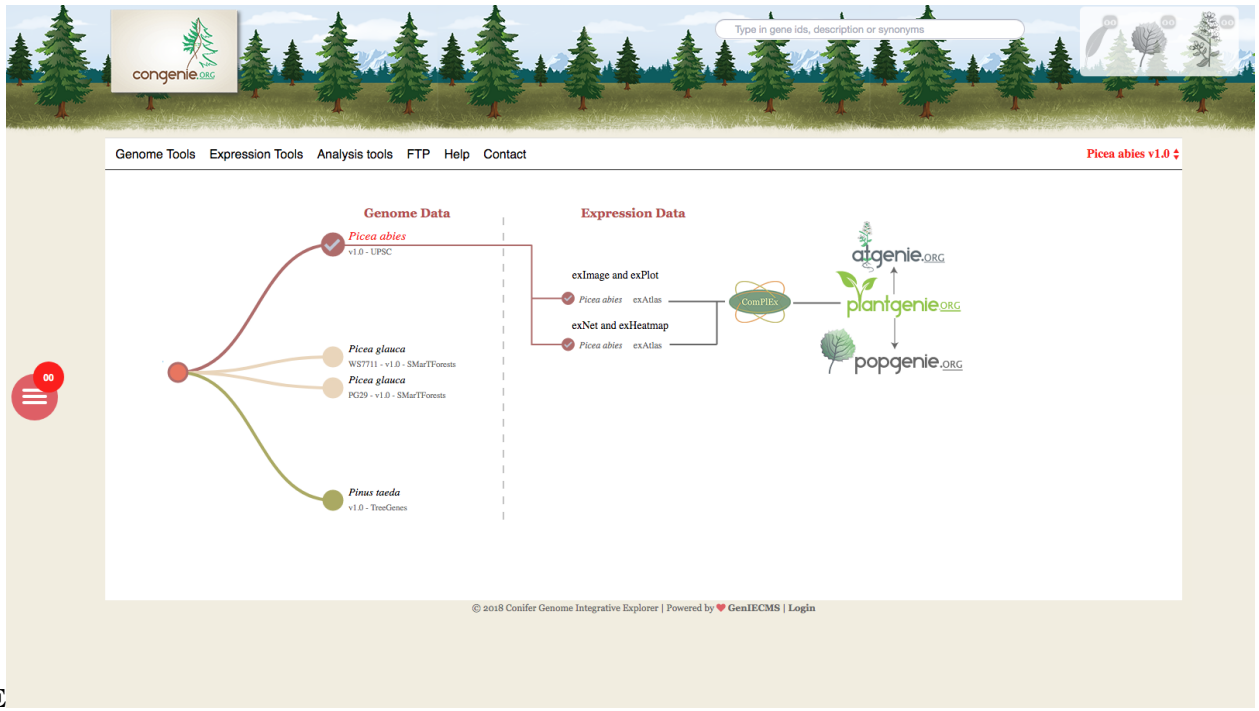
Header image, Logo and Background colour can easily be changed once user login to the system as shown in the following screencast.

CHAPTER 4

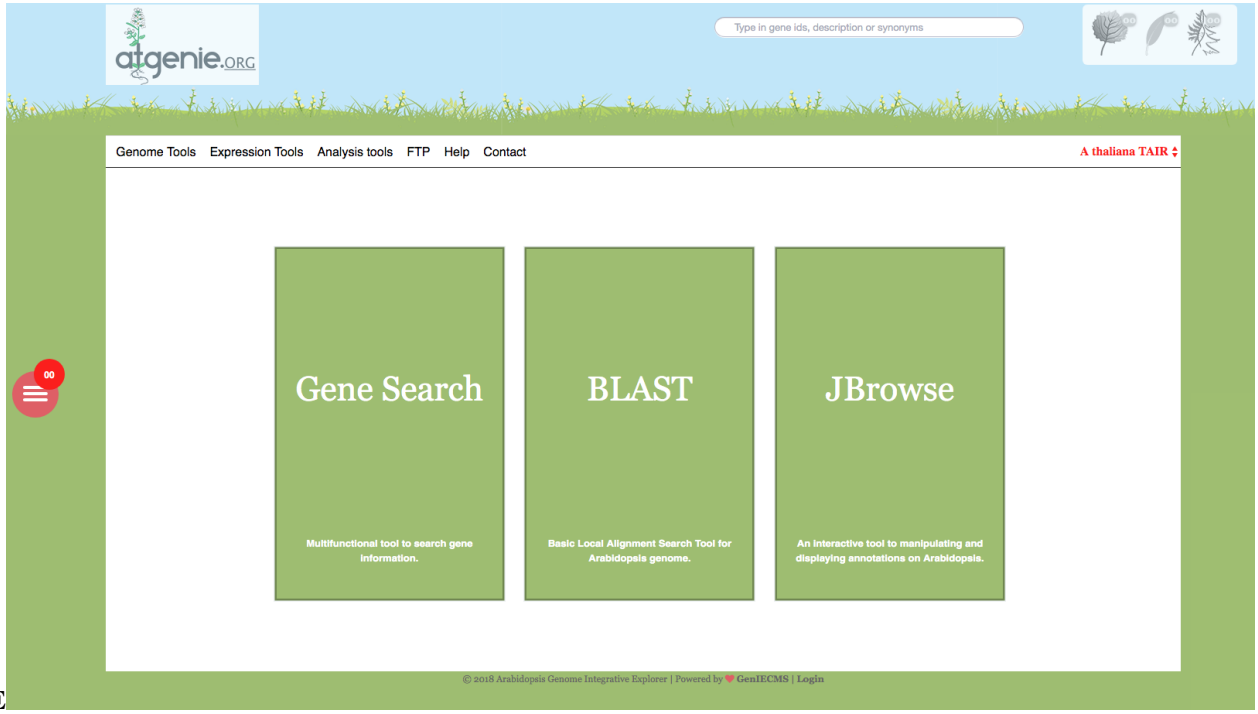
Who Uses GenIE-Sys



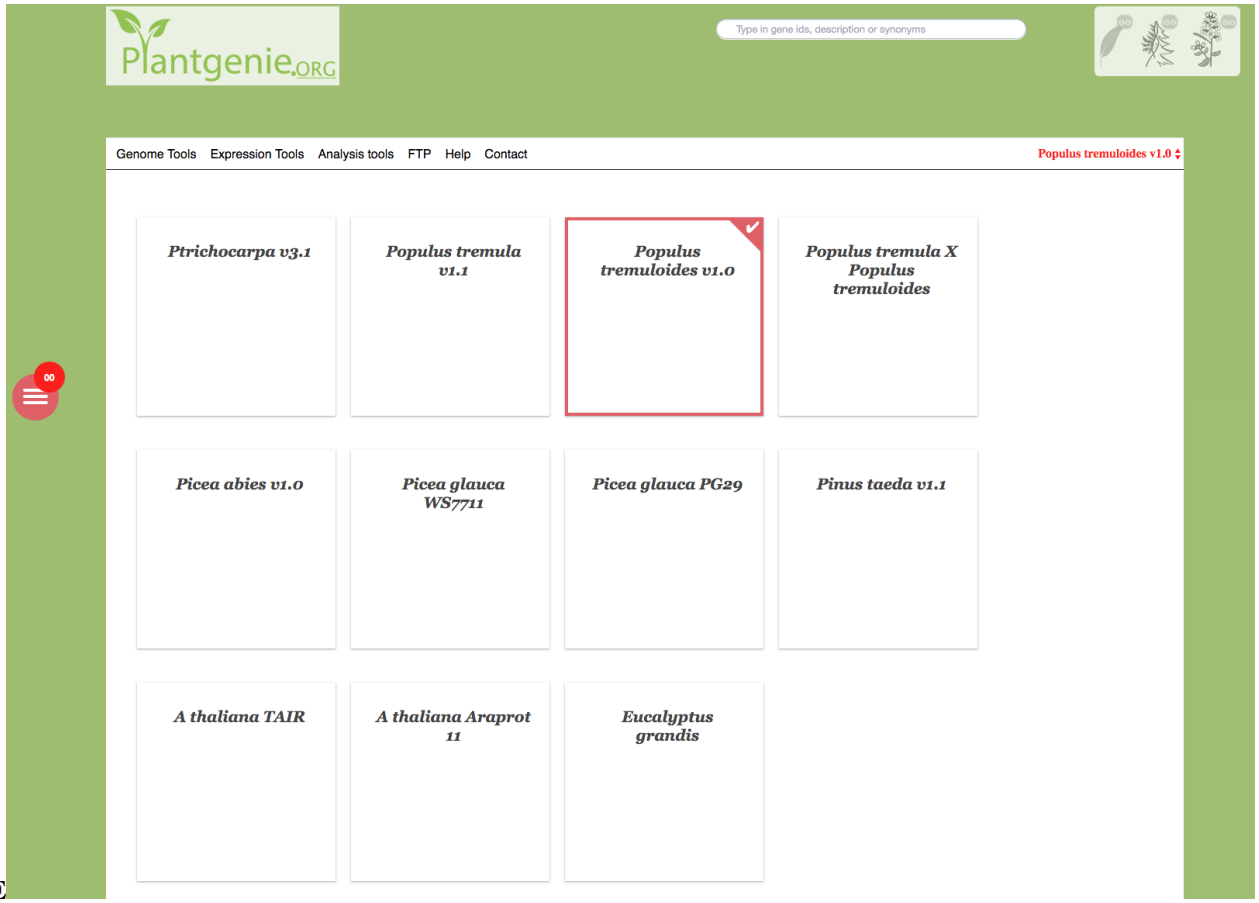
PopGenIE



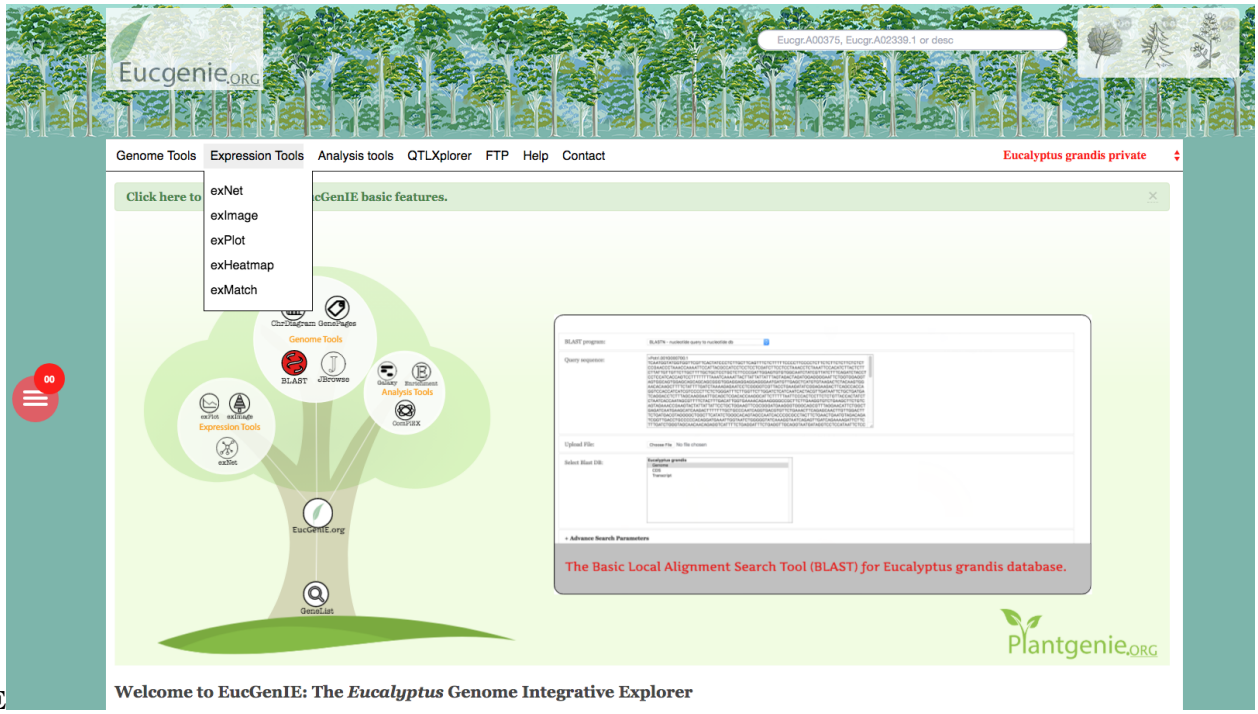
ConGenIE



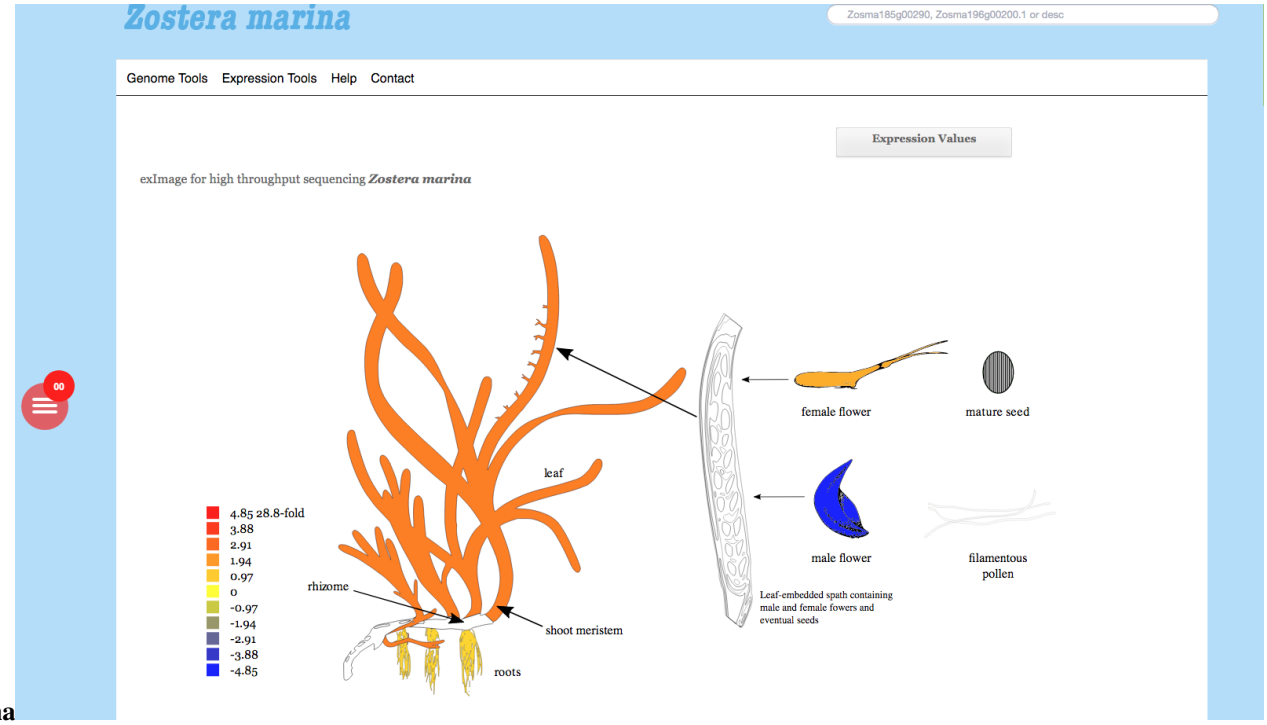
AtGenIE



PlantGenIE



EucGenIE



Zostera marina

CHAPTER 5

Basic Usage

5.1 GeneList

The GeneList tool allows the user to search for genes using **gene IDs, descriptions, experiments, GO ids and different annotations** then saves the result in a list that can be used by other tools.

Basic Usage

Simply type in gene ids, descriptions, or different annotations. The matching genes will be displayed with selected annotations. The result can be customized by clicking the Select Displayed Annotations button. There are three buttons to **Save all to Gene List**, **Remove selected from Gene List** or **Empty Gene List**. The **Share table** button allows sharing the current GeneList with other users by way of an auto generated URL. The GeneList tool is the starting point for most PlantGenIE workflow.

Multiple GeneList

The GeneList tool is capable of holding several named gene lists for use in other tools. These lists can be Added, Renamed or Deleted. Once clicking on a GeneList name, it will become the active GeneList and will be displayed in all other tools. GeneLists will remain for seven days while shared GeneList will be saved for 30 days.

5.2 Gene Information Page

Overview

The Gene information page contains basic information about a gene including sequence, function and family information.

Basic Usage

The Gene Information Page consists of dedicated tabs names: Basic Information (including GBrowse details), Sequence, Functional Information, Expression Overview, Gene Family and Publications (including community annotations). The Basic tab gives a quick overview (Chromosome, Description, Synonyms, Arabidopsis id and GBrowse image map) of the gene or the gene model. The Sequence tab contains Genomic-, CDS-, Transcript- and Protein-sequences. You can easily BLAST any of the above sequences by clicking the related BLAST button, and you can

extract Upstream or Downstream Genomic sequence by adjusting the upstream/downstream input boxes. 5' UTR, CDS, 3' UTR regions are highlighted with dedicated colors. The Functional information tab includes annotations from different data sources including GO, PFAM, PANTHER, KO, EC and KOG. The Expression Overview tab displays the exImage image for the gene, and gives a visual overview of the tissues where the gene is expressed. More tissues/samples are avaiable at the dedicated exImage tool. The Gene Family tab contains gene family information across several different species. You can select a species and download fasta file or create a phylogenetic tree using either Galaxy or Phylogeny.fr. You can also send a gene families to the PlantGenIE GeneList or visualize expression conservation/divergence using the ComPlEX tool. The Community Annotation tab will display the user submitted annotation of gene models. You can edit the current annotation using the WebApollo annotation editor. Once members of PlantGenIE team approved the new submission, it will display inside the Community Annotation tab.

The Gene Information page is the starting point to WebApollo and this will also be the final destination for many of the PlantGenIE tools, for example GBrowse, GeneList or exPlot. There are dedicated pages for both genes and transcripts information.

Implementation

The Gene Information page uses JavaScript, PHP, MySQL, PostgreSQL, JQuery and d3js.

5.3 BLAST

Overview

The BLAST (Basic Local Alignment Search Tool) tool compares input sequences with datasource to identify homologous sequence matches.

Basic Usage

Paste your sequence (with or without a FASTA header) into the query input text box. Transcript sequence can be extracted by type in a gene ID into the Load example text box or upload a sequence file (Less than 100 MB)with the upload file function. Then click and select the desired dataset from the lists of available BLAST databases and click the BLAST! Button at the bottom of the page.

BLAST plugin uses standard default NCBI BLAST options. However users can change the following advanced options:

BLAST Results page will be automatically reloaded until the search results are successfully retrieved. BLAST results are organized into a table containing Query ID, Hit ID, Average bit score (top), Average e-value (lowest), Average identity (av. similarity) and Links. Clickable BLAST results display the corresponding region of identified homology within the JBrowse tool, where the matching region is shown.

Implementation

BLAST search tool is implemented using NCBI Blast (v2.2.26) and a backend PostgresSQL Chado database. 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.

5.4 GBrowse

GBrowse is an open-source, genome annotation viewer.

Basic Usage

To find particular region of the chromosome, type a gene name, a short sequence (minimum of 15 bp), or a nucleotide range in the Landmark or Region box located near the top left of the page and click on the Search button. The area shown in the Details panel is highlighted by a box. You can grab the box and slide it left or right within limits (it can't

slide over the whole genome). Once you get to a particular location, you can fine-tune the view with the Scroll/Zoom buttons to move along the chromosome or change magnification.

Data

GBrowse uses in house annotation data and data from Phytozome and Plaza.

Implementation

PlantGenIE GBrowse uses customized version of Generic Genome Browser version 2.49. We use dedicated GBrowse servers for each of our PlantGenIE resources.

5.5 exImage

exImage provides an intuitive pictographic view of expression data across a diverge range of expression datasets.

Basic Usage

Users can either enter a gene ID in the input text area (and hit the "GO" button) or create a gene list which then will appear as an interactive list in the tool. exImage will shade the samples according to expression levels across multiple samples using either absolute or relative values. Relative values displays expression relative to the mean expression across all samples. The current view can be exported in various vector formats including publication ready PDFs or as expression values. The 'Take a tour' feature will provide a brief introduction to the basic functionalities in exImage.

Data

exImage uses VST (Variance-Stabilizing Transformation) or TPM (Transcripts Per Million) values for absolute expression, and no unit for the relative values. Absolute expression values were generated by aligning RNA-Seq reads to the reference genome and gene annotation with aligned read numbers then used to calculate VST values.

Implementation

exImage was developed using PHP, Javascript, d3js, rsvg-convert, imagemacgick, librvg and batik. exImage uses a MySQL database as a backend data source. exImage was inspired by the eFP resource.

5.6 Sequence search

The Sequence search tool extracts CDS, Transcript, Protein or Genomic sequences from the GenIE-Sys database.

Basic Usage

Simply type in gene ids in the large text box and select the desired type of sequence from the dropdown lit. Then click the Submit button. Now you can use the Download button to download the sequence or the phylogeny tree button to create a phylogenic tree using phylogeny.fr. By default, the sequence search tool shows the genes from the active GeneList.

Data

Sequence search uses both in house assembly data and data from Phytozome .

Implementation The Sequence search tool was developed using JavaScript, PHP, MySQL and fastacmd.

CHAPTER 6

Indices and tables

- genindex
- search