An STSM Report for the creation and implementation of a database providing information about the use and development of vegetable rootstocks

Candidate: Richard Odame

Start Date: 3rd June, 2014

End Date: 3rd July,2014

Background and Accomplished Aims:

Many vegetable crops such as tomato, pepper, cucumber, melons and watermelons are produced on grafted plants where the rootstock and the scion are joined by grafting. This technology allows different traits to be bred into each half of the plant and is vital to protect plants from soil borne diseases and control plant vigor.

A database of information about the use and development of vegetable rootstocks, was required to support the "FA1204 Vegetable Grafting To Improve Yield And Fruit quality Under Biotic And Abiotic Conditions" project. The database created consist of bibliographic data, information about plant genotypes, information about plant genes, information about researchers and information about commercial activities.

The requirements for the database web application created was discussed with Professor Giuseppe Colla who is the coordinator for the Cost action FA1204 project and other members of the project in Tuscia University.

Netbeans

Materials

1. A Computer With Windows Operating System,
2. JDK Version 8.0
3. MySQL 5.6 (Open Source) which consists of the folllowing: 1. MySQL Server, MySQL Connectors which includes JDBC(Java Database Connectivity Drivers), MySQL Workbench and MySQL For Excel
4. Netbeans 7.4 (Open Source)

MySQL 5.6 and Netbeans 7.4 was installed on my personal working computer which uses Windows 8 Operating System. This was to allow all work to be done on my PC so it does not interfere with

University Of Tuscia's server since programming can affect University's server. This was mutually agreed between myself and Prof. Colla.

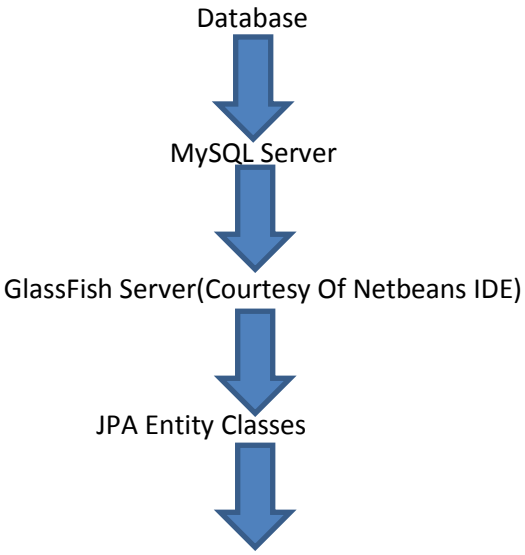
Methodology Employed

"vegrootstock web application" is the robust secured java/mysql data retrieving tool that has been created at the end of the STSM under Cost Action FA1204 in Tuscia University under Professor Giuseppe Colla, by me Richard Odame, who is a Bioinformatics student studying in Cranfield University.

STEPS USED IN THE CREATION OF WEB APPLICATION

- Created XHTML To Hold Data Table Using Prime Faces
- Created Required Database Tables
- Configured JDBC Realm in Glassfish
- Created Web Application using Netbeans Maven
- Configured Security Constraints in web.xml
- Configured Security Role mapping in glassfish web.xml
- Configured JDBC Connection Pool and JNDI Resource using glassfish-resources.xml
- Used SHA-256 Hashing To Encrypt Passwords in Database
- Secured the created vegrootstock java Server Faces Web Application created
- Seed Database with made up user accounts
- Created Entities For Database
- Created Persistence.xml
- Created Entity Manager
- Created Session Bean

SCHEMATIC PRESENATION OF HOW THE ENTIRE SECURED DATABASE WEB APPLICATION IS CREATED



Persistence.xml to manage Entities

Servlet To Test Server Connection

Entity Manager To Work On behalf of the persistence.xml and the Upper lighter layer of the web application

Now With created java classes for the web application , the web application with Data retrieving ability was created. This is made possible by linking the managed bean to the XHTML Pages created.

THE REASONING, CREATION , SCHEMATIC REPRESENTATION AND DEPLOYMENT OF DATABASE TO MYSQL SERVER

vegrootstockproject is the schema name for the project model vegrootstock.mwb created in MySQL platform. When db(database) is referred to, it represents the Schema name which contains the relationship between tables and not the model name.

Tables created for this schema are

1. User
2. Group
3. Vegetable Cultiver
4. Abiotic Factor
5. Biotic Factor
6. Response Type
7. Genotype
8. Genes
9. Research Work

DEFINITION OF OBJECTS IN DATABASE SCHEMA CREATION

User, is the individual or researcher who would access the vegrootstock web application that has been created.

The Group, contains two group names which are the admin group and the user group. The admin group has rights to edit and update the database of the created vegrootproject web application whilst the user has read only rights

Vegetable Cultiver refers to the plant that would be used in grafting. The plant referred to here would be either a rootstock or/and a scion in the context of vegetable grafting.

Abiotic Factor, is the non living factors that affect the growth of the vegetable cultivar.

The Biotic factor is the living factors that affect the growth of the vegetable cultivar

Response Type refers to how the plant vegetable rootstock performs with the scion when exposed to abiotic or/and biotic factors

Genotype refers to the genotypic constitution of the vegetable rootstock understudy

Gene refers to the genetic constitution of the vegetable roostock understudy.

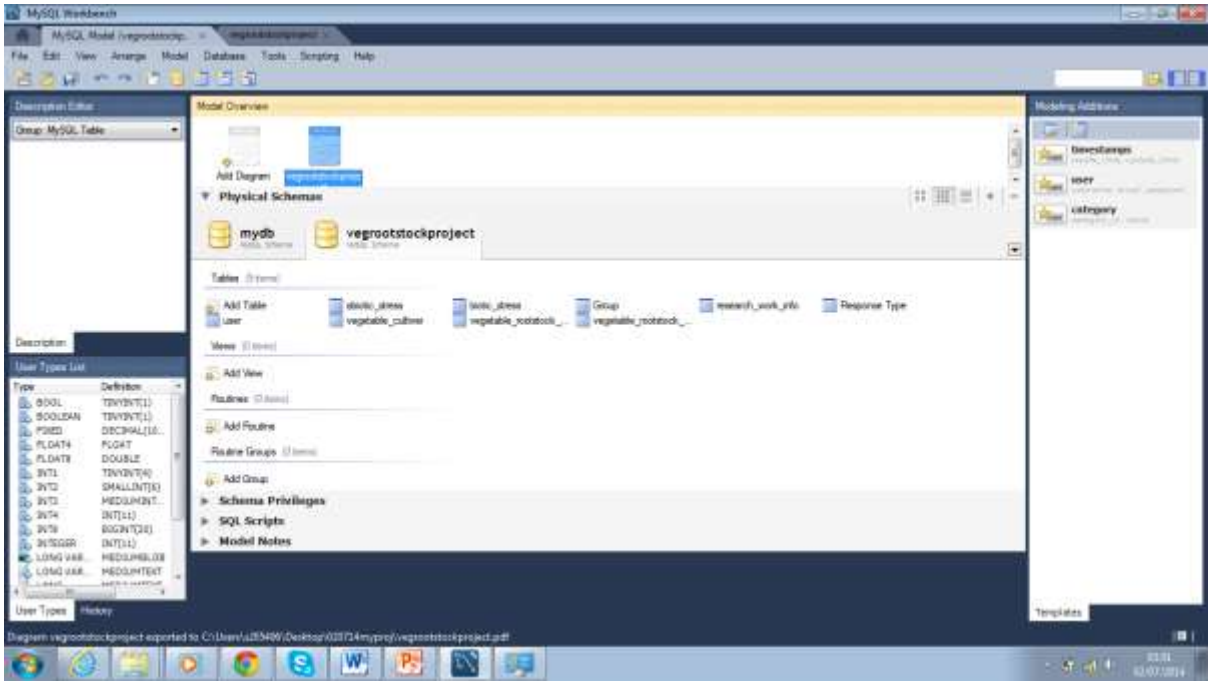
Research Work, refers to the ongoing research and completed research in some areas of vegetable grafting that would support an overall enhancement and usability of our data retrieving web application tool.

The objects defined above are called Tables in MySQL and each would have attributes that describe the respective tables.

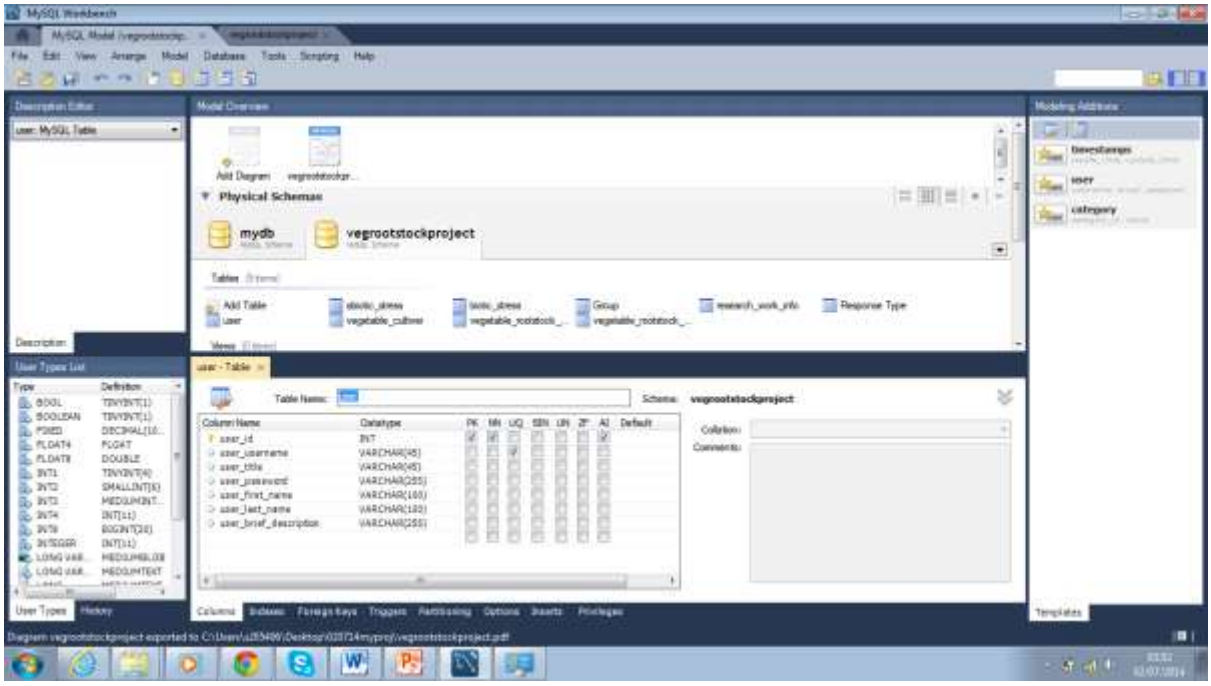
ATTRIBUTES OF RESPECTIVE TABLES

NB: The attributes listed for each table are expansive and hence only the attributes that are vital for the present Data retrieving web application have been included in the construction of the vegrootstock data retrieving project.

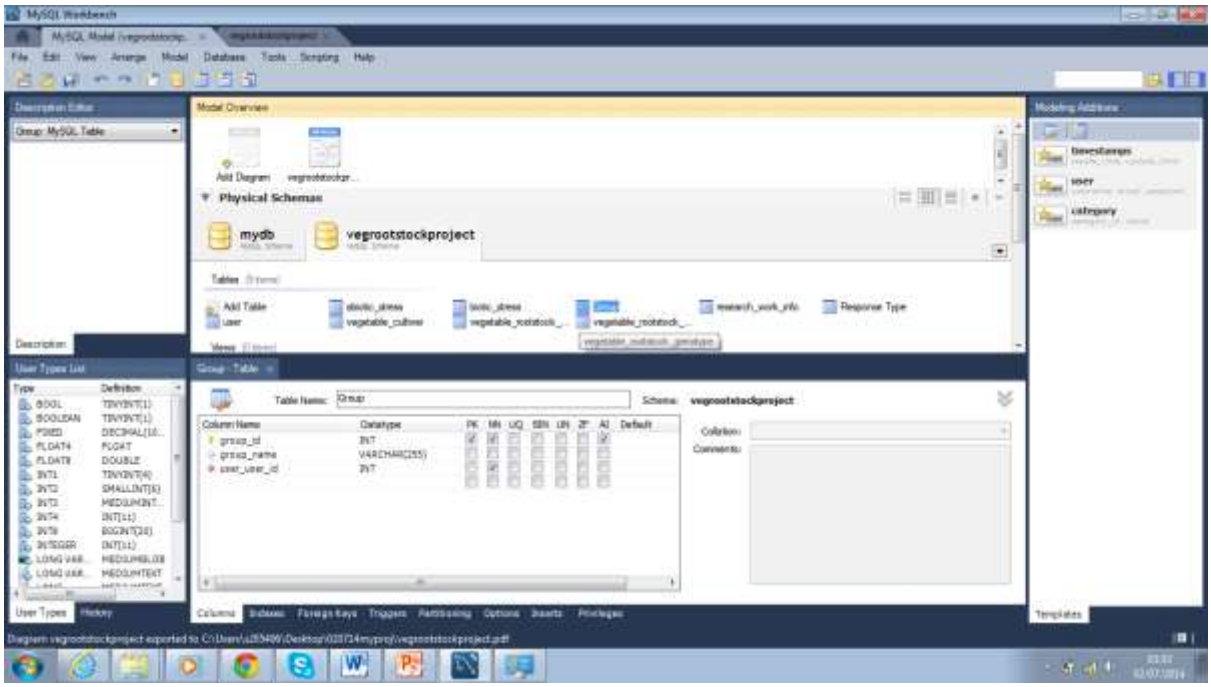
DATABASE SETUP



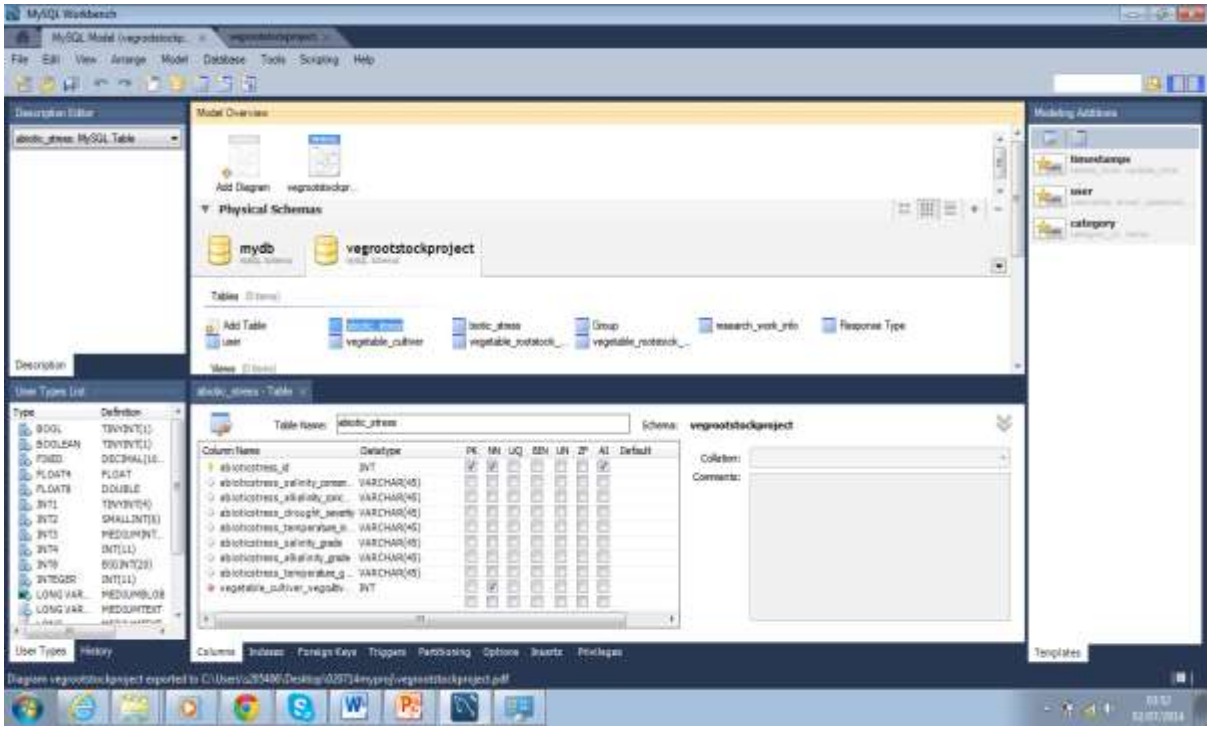
The above screenshot shows the created tables in MySQL



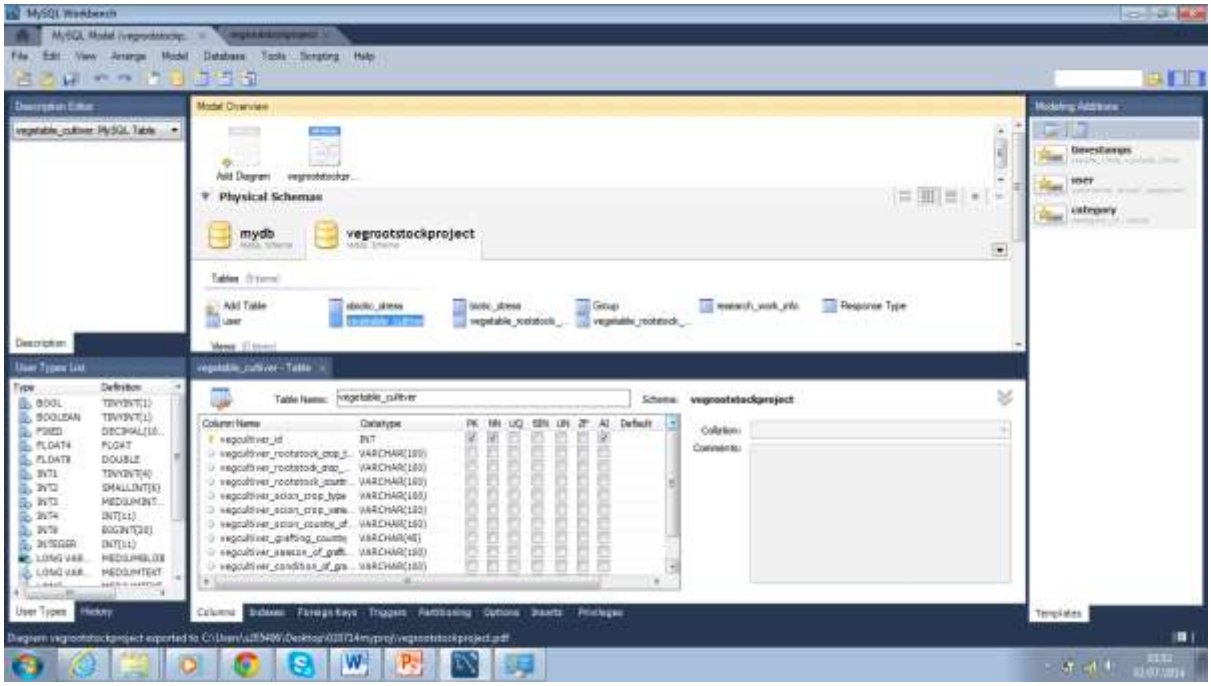
The above screenshot shows the attributes created under the table User. AI has been ticked to allow auto increment. User's username is a unique key and hence repetition in the table will not be allowed. User_id is the primary key and would not be null(NN) because it holds the link to the table User. The PK would be a FK in the Group table since the Group table depends on the user table to complete a meaningful statement of a group and a group member because of the user credentials required to identify a group member.



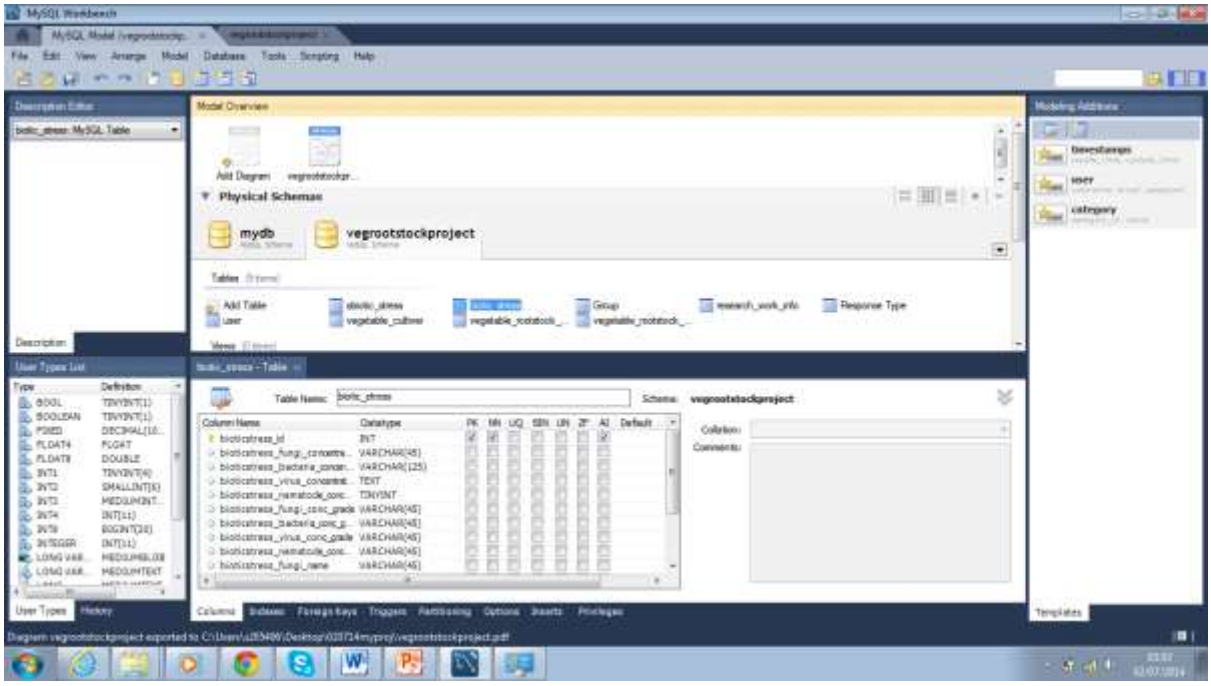
The above screenshot shows the attributes of the group table with a link to the user table through the user Foreign Key(FK)



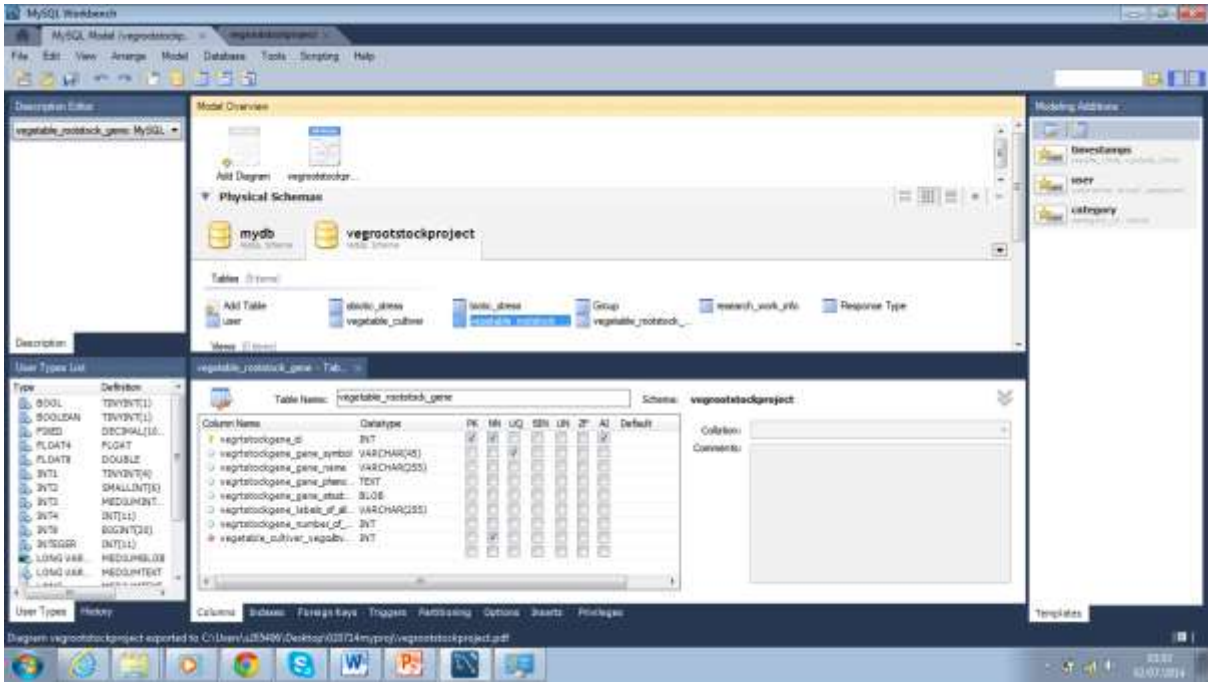
The above screenshot shows the attributes of the abiotic_stress table which would be a FK to Response Type to make a meaningful statement regarding the subject of vegetable rootstock.



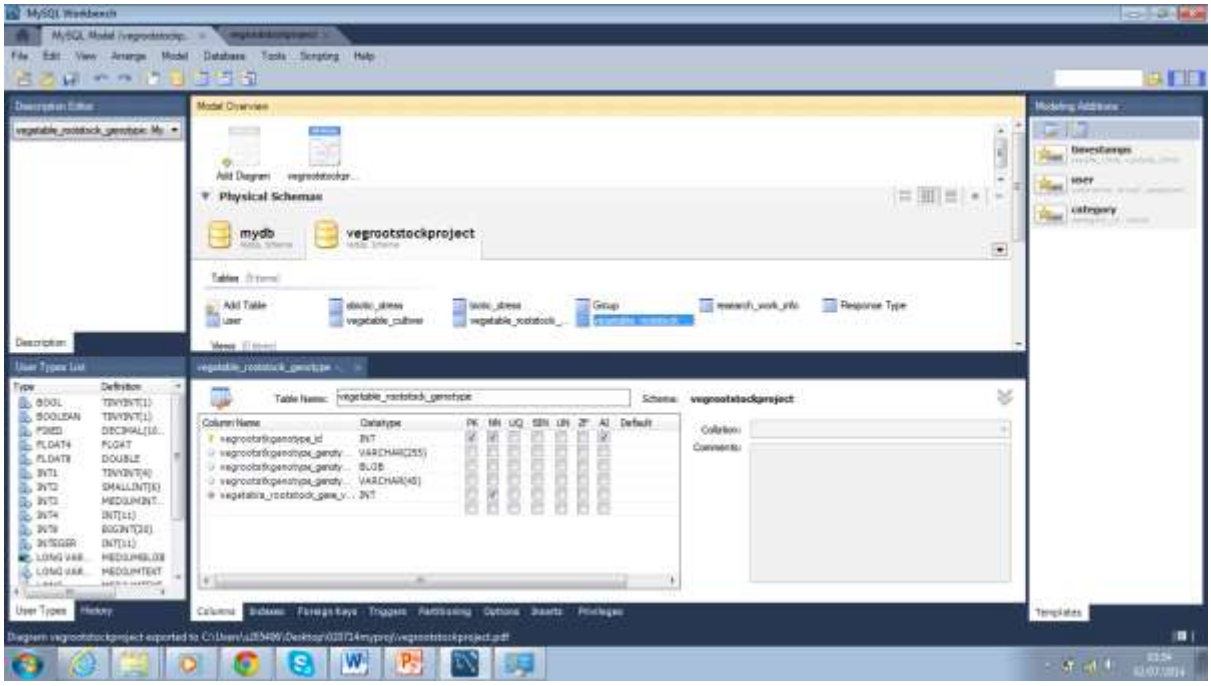
The above screenshot shows the attributes of vegetable cultivar. This makes a meaning statement on its own regarding the subject of vegetable rootstocks



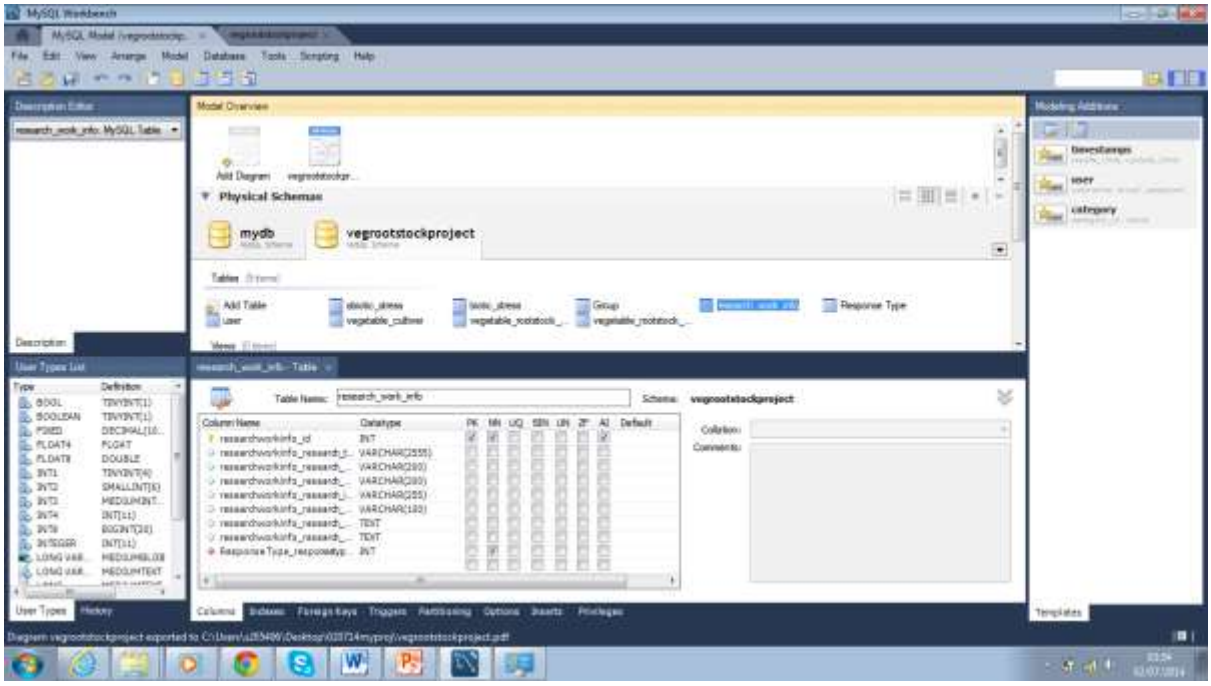
The above screenshot shows the attributes of biotic_stress which would be a foreign key Response type make a meaningful statement regarding the subject of vegetable rootstocks.



The above screenshot shows the attributes of vegetable_rootsock_gene which has a foreign key in vegetable cultivar table so this gene table can make a meaning statement regarding the subject of vegetable rootstocks

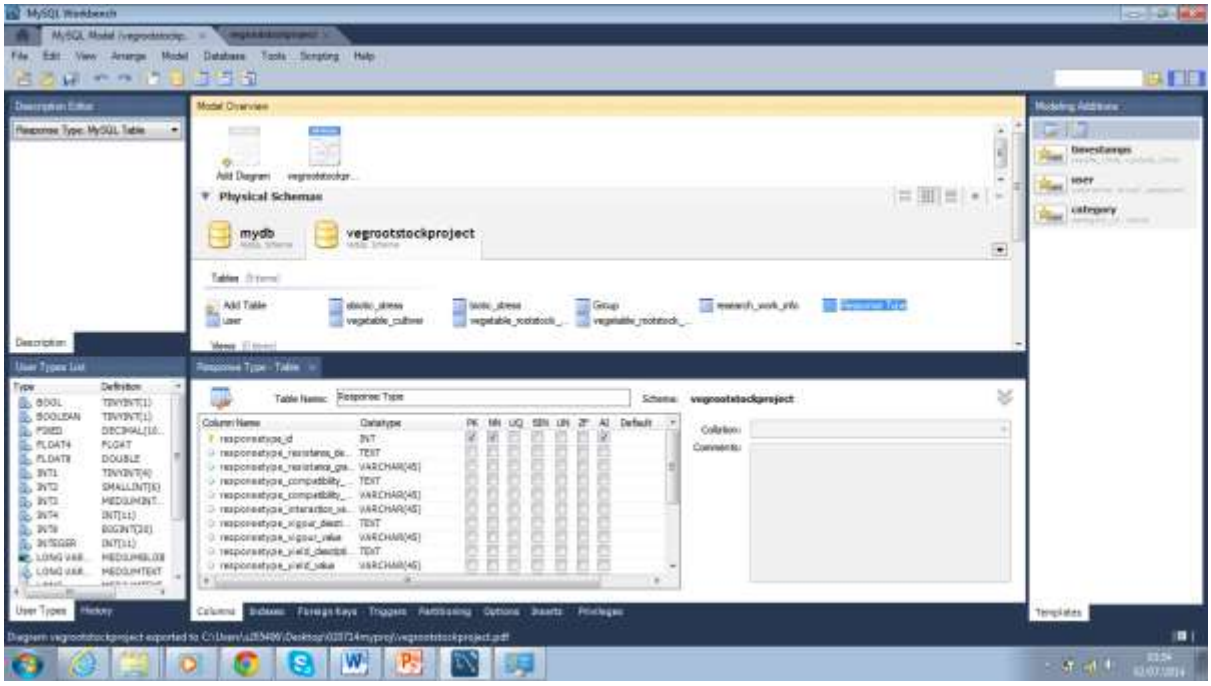


The above screenshot shows the attributes of the genotype table which has a foreign key in gene table that also has a FK in vegetable cultivar. This allows for a good link between the three tables to carry on the flow of meaningful information regarding the subject of vegetable rootstocks



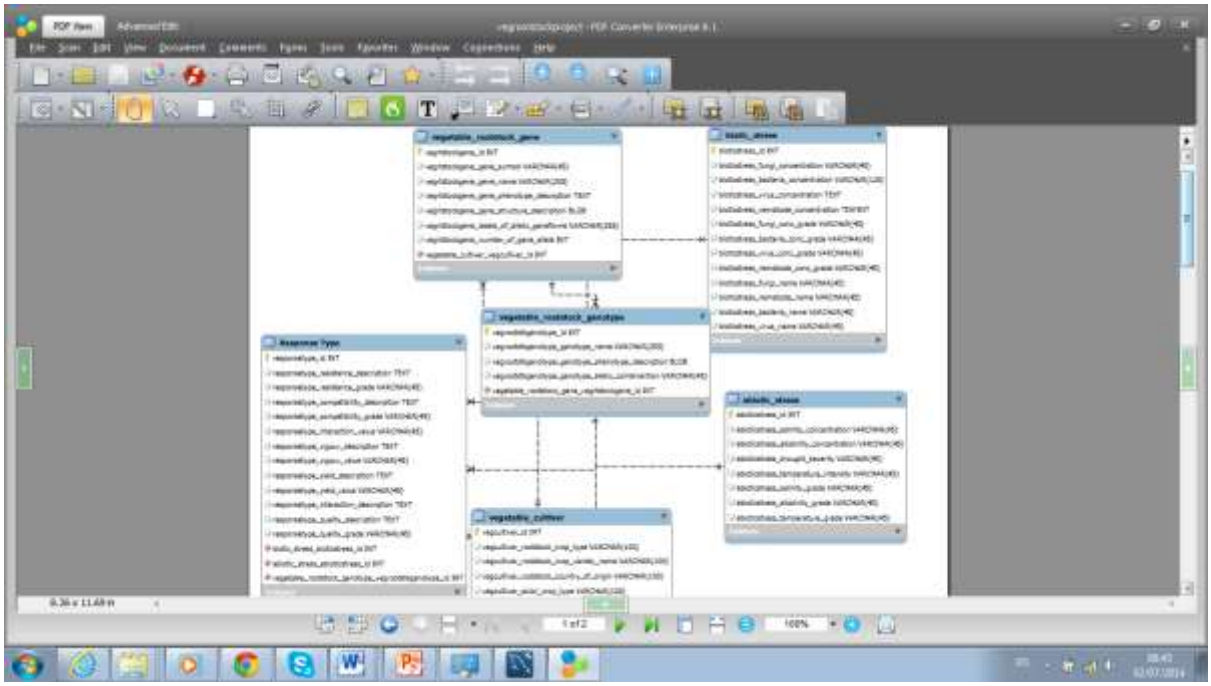
The above screenshot shows the attributes of research_work_info table with the PK not null and a forein key in response type so the research_work_info table can be linked to all the FKs in response type and the response type table as well. This allows an instance record of the research work info to be linked to the necessary attributes of response type and the other table sthat re foregn keys in the

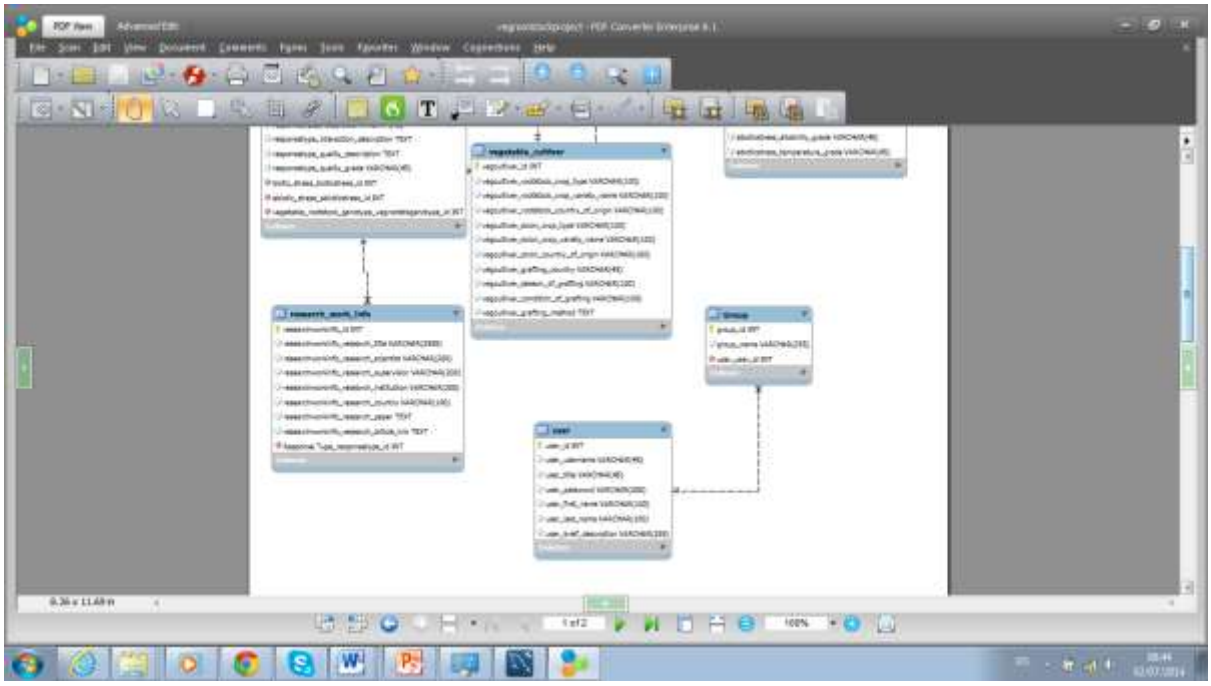
response table. This allows the interlinking of information on a lighter scale because of the use of integers to refer to the foreign data in research work info table by way of FKs.



The above screenshot shows the attributes of the response type table that has FKs in abiotic factor, biotic factor and vegetable rootstock genotype tables.

SCHEMATIC SHOT

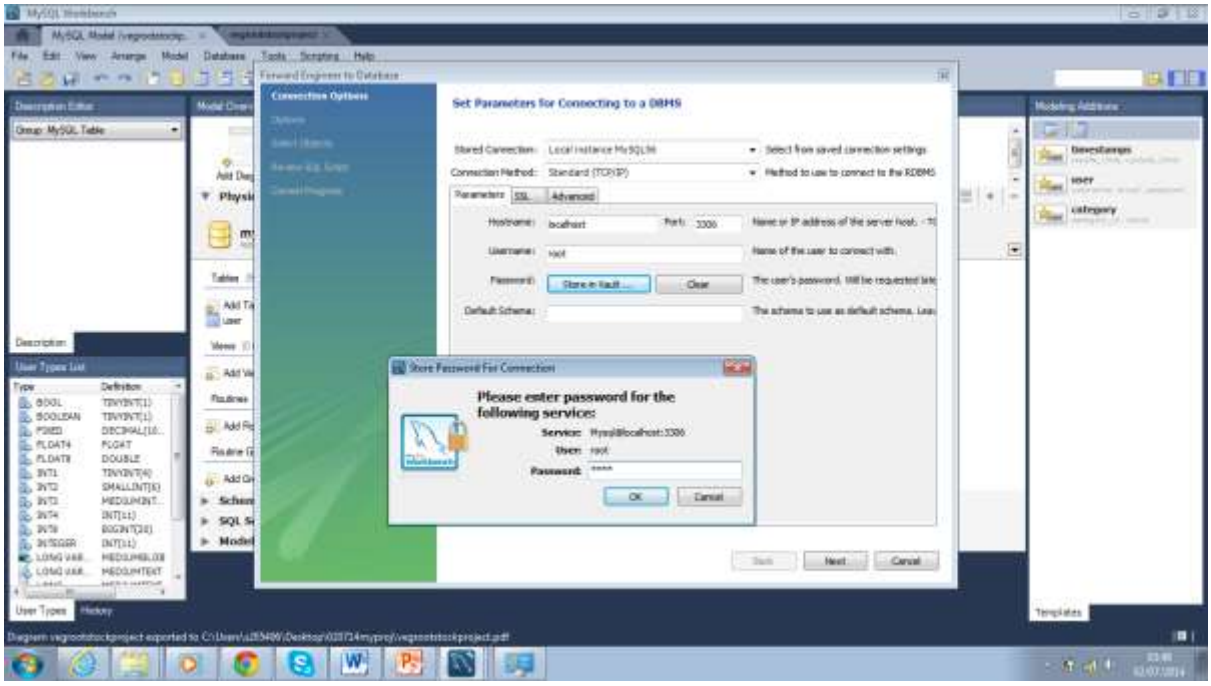




The two screenshots above is a pdf of the schematic representation of the database that has been created. This schema will be forwarded to MySQL server so our database can be accesed at all times via the server.

FORWARD ENGINEER IN MYSQL PLATFORM

To transfer schema to MySQL server click on database menu in MySQL platform where the database is opened. Click on Forward Engineer> Set Parameters for connecting to the DBMS



After the creation of the schema, the database is forwarded to the MySQL Server using the forward engineer as shown in the screen shot above. This required the input of right credentials to access the server. Stored Connection; here you choose from the saved connection settings. Local instance MySQL56 was the settings set.

Connection Method; here the method to use to connect to the RDBMS is specified. TCP/IP was used.

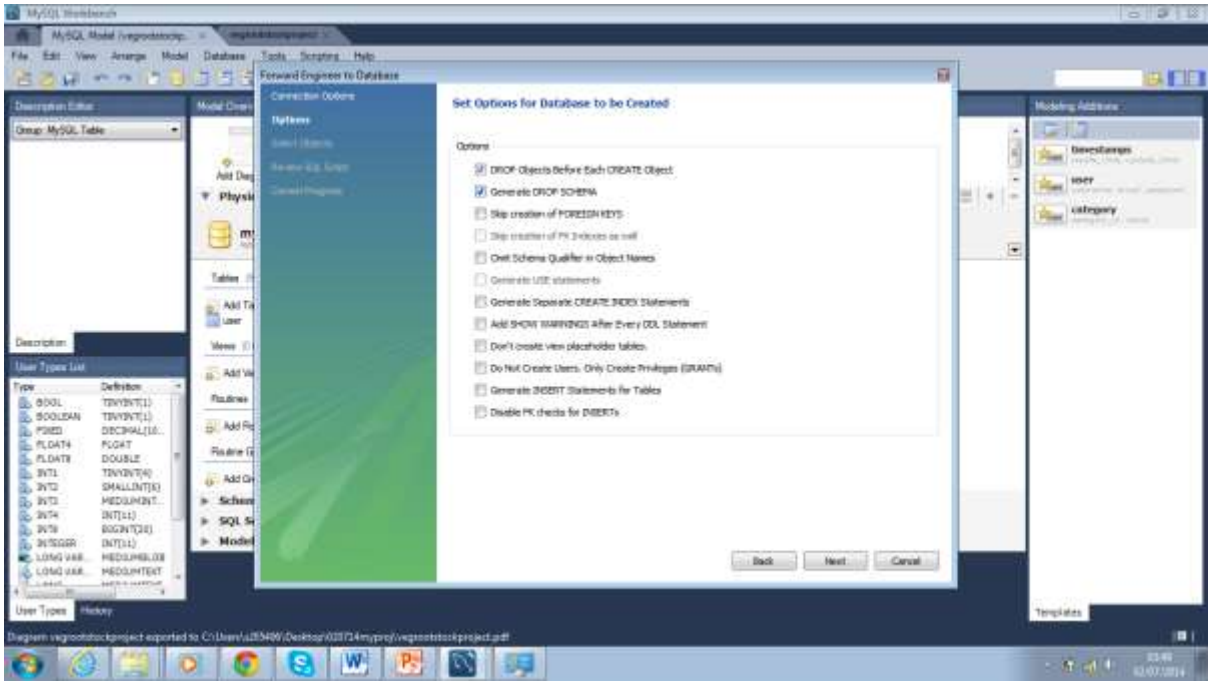
Parameters For Connecting To DBMS

Hostname: Localhost; Port=3306 (Name or IP address of the server host)

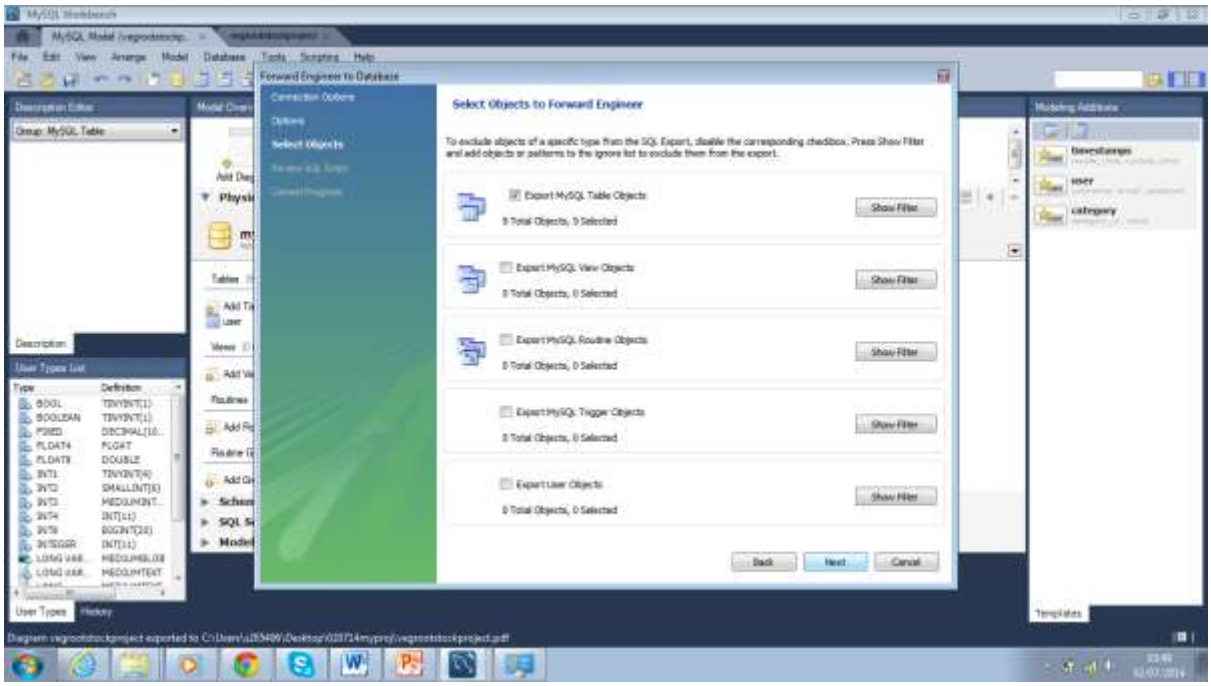
Username

Password

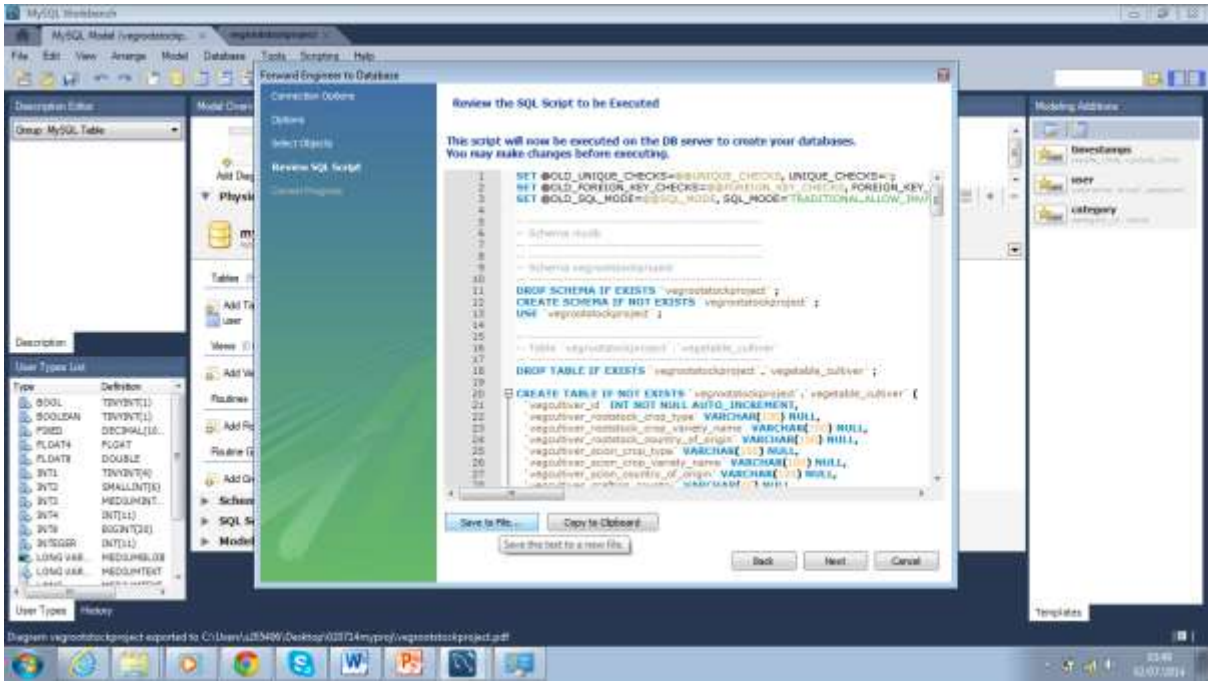
Default schema on ths page should be left blank



The above screenshot shows parameters that have to be ticked to prevent duplication of schema and schema objects. This can cause problems at the upper end when database has to be used and hence a good practice to tick the two boxes to prevent this from happening.



The above screenshot shows the objects that need to be engineered to the server being emphasised. Correction of how many objects that has to be sent can be made from here.



The created db schema script has to reviewed here before execution to prevent error. When ticks for Connect To DBMS; Execute Forward Engineered Script; Read Back Changes Made By Server; Save Synchronised state are all correct, then the forward Engineer finished successfully.

TO ENSURE DATABASE IS DEPLOYED TO MYSQL SERVER

In MySQL platform, under MySQL connections, click on Local instance MySQL56 and with username and with localhost 3306.

Scroll down on schemas on the left pane to select the schema created and view tables

To Create Connection Between MySQL Server And Netbeans

Go to services in IDE, the schema vegrootstock project will be in other database. It means it is in the IDE paltfform but we want to link it with glassfish server.

Click on drivers under databases.

Click on Drivers unders under databases.

Rightclick on MySQL(Connector/J Driver) > Connect Using> Customise Connection

Host= localhost Port:3306

Database: the dbname in the IDE bearing the name on the MySQL Server, vegrootstockproject

Drivername: MySQL(Connector/JDriver)

Test connection, when successful proceed.

The jdbc URL will be jdbc:mysql://localhost 3306/vegrootstockproject?zeroDateTimeBehaviour=convertToNull

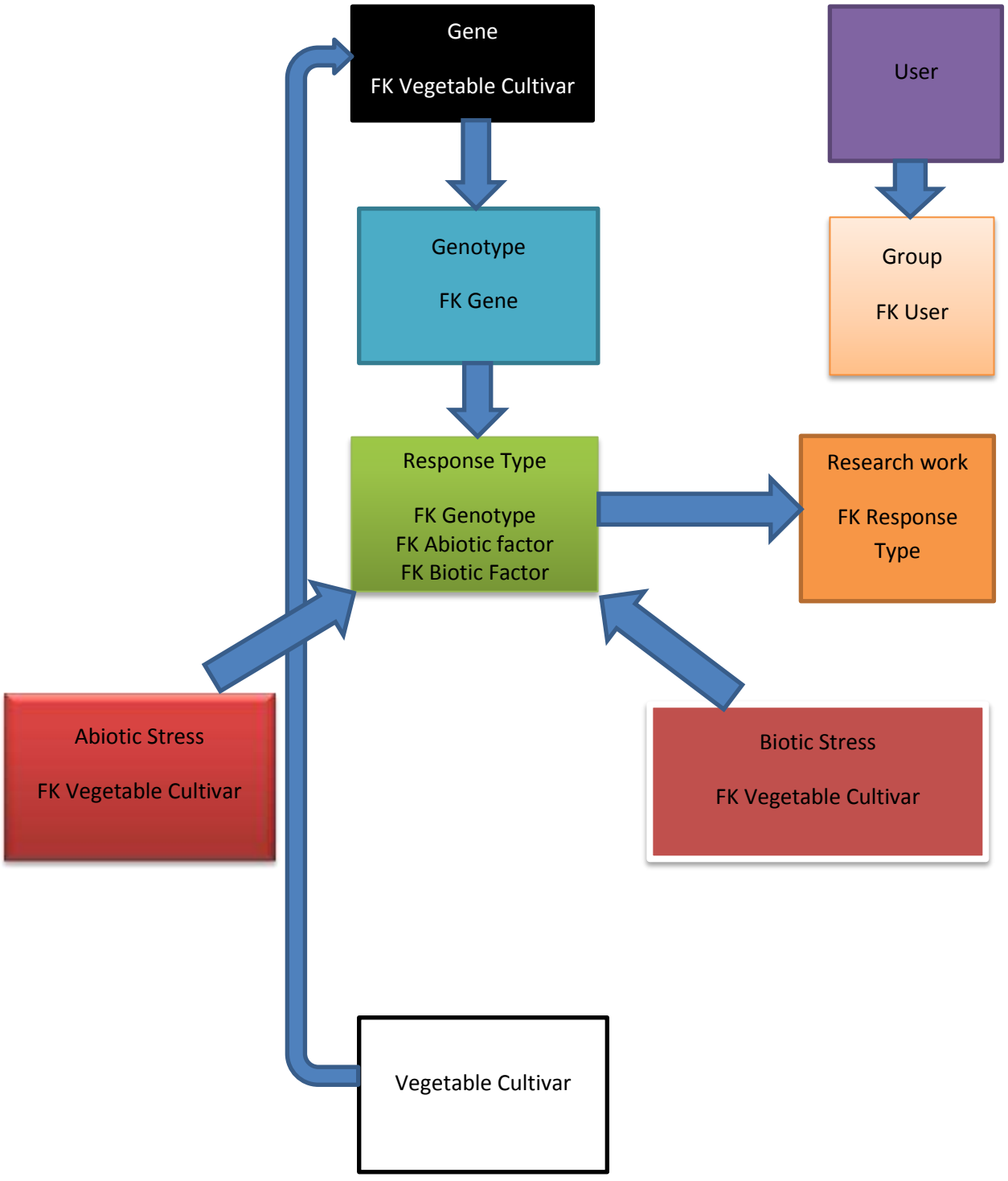
Click Next from here

Choose Database schema name is often greyed out. Click next

Input connect name which would be jbdc:mysql://localhost:3306/vegrootstockproject. Click finish.

Now when you observe the connection under Services with a joined design, when you click you should have the schema name and when you click further you should see the tables of the database which establishes link to the glassfish server. If the link is broken, double click to join it and here you may be required to input your password

SUMMARISED REPRESENTATION OF RELATIONSHIP BETWEEN THE TABLES CREATED



NB: Tables on their own have attributes that specifically describe the table header. The arrow in the diagram is configured as below:

The box with the arrow pointed to it contains the Table Header that is dependant on the Box defined table Header of the other box with the arrow pointed away from it, to make a meaning analysis.

Ideology here is though you may be far from me (box with arrow pointed away from it), you are still dependent on me, so to remind you, I will ask my PK which is a symbol of me in the table, to be in your table as a FK. You cannot exist without me. I am anchorage to your full existence.

Relationship Between Vegetable Cultivar, Abiotic ,Biotic Factors and Response Type

Without abiotic or biotic factor and vegetable cultivar, a user will hold no basis to discuss Response Type. Response Type of a vegetable rootstock is dependent on attributes listed in Vegetable Cultivar, Abiotic and Biotic Factor.

So to keep the relationship between the Response type table and the 3 other tables, the response type table would accommodate the primary keys of Vegetable Cultivar, Abiotic factor and Biotic factor as a foreign key.

Response Type table

+FK abiotic factor

+FK Biotic Factor

(+FK Vegetable Cultiver)---this will be inherited indirectly through the Vegetable_Rootstock_Genotype table

Relationship Between Response Type, Biotic Factor, Abiotic Factor, Genotype And Vegetable Cultiver

Since Response Type is closely related to Abiotic factor, Biotic factor and Genotype, we will drop the FK of VC in response Type is indirectly carried by Veetable_Rootstock_Genotype table.

On the otherhand, for a user to discuss the abiotic and biotic factors that affect a vegetable rootstock, response by vegetable rootstock to these factors must be the key subject for discussion. The existence and relevance of abiotic and biotic factors is dependent on Response of Vegetable Cultiver, which is the response type. But since Response type already inhabits Vegetable Cultiver table through the Genotype table, which is inherited from the Vegetable_Rootstock_Gene table. Biotic Factor and AbioticFactor will be the additional tables required by Response type by way of FK.

The response type table will have a Genotype FK because the response type of a vegetable cultivar will be discussed with the genotype in mind. If anything, from research we know that our phenotypic traits are as a result of the genetic composition we have. For that reason the response type which is the phenotypic trait exhibited by the vegetable rootstock will be dependent on the genotype. Response type will not be mentioned in close relation to the genotype. Response type is non-existent in a discussion relating to the two tables without genotype being mentioned hence Genotype will be a FK in Response type.

Response Type table

+FK Genotype

+FK Abiotic Factor

+FK Biotic Factor

Pertaining to the subject of vegetable rootstocks, a discussion on genotype and abiotic or biotic factors will be dependent on vegetable cultivar and the outcome of Response type to these factors to make a meaningful analysis, so there is no direct relationship between Genotype and Abiotic or Biotic Factors.

Genotype and vegetable cultivers have a direct relationship but genotype will be discussed with respect to the vegetable cultivar. Meaning a meaningful user discussion is non existent if the vegetable cultivar is not mentioned before the genotype, considering the subject of vegetable rootstocks under discussion. So Genotype is dependent on Vegetable cultivar table that holds vegetable rootstock as an attribute. Hence the PK of Vegetable Cultiver table will be a FK in Genotype table. This will only be indirectly through Gene table which has Vegetable Cultiver as a FK.

Vegetable Cultiver And Gene

With respect to the analysis above, VC and Genes will be the extremes that will have no direct connection with each other, yet these two are very vital for the related data. Abiotic and biotic factors are directly linked to VC but can be indirectly linked to Response type. Gene is directly linked to VC and Genotype and cannot be indirectly linked to any of Genotype and VC. So we will create our relationship as such so that the FK of VC in biotic and abiotic factor

A discussion of vegetable rootstock and gene will be directly related since this information can be all that a user needs. The gene of a vegetable cultivar will be irrelevant if the the vegetable cultivar is not mentioned first. So the Gene table will be dependent on the vegetable Cultivar table.

The response type will still be dependent on the Abiotic and Biotic factors and its FK which is suppose to be in the Response Type table ill be indirectly inherited through the FK of vegetable cultivar that the Gene table carries.

This way there is interconnectivity with all. Any search can retrieve information about all the other tables.

So for the generated user table, once the vegetable cultivar table is chosen and the biotic or abitic factors or both chosen, the response will be generated. Then to increase the quality or yield, the system will be filtered by the vegrootstock only since quality and yield can be affected by abiotic or biotic factors, the type of scion, the location of grafting, the country of grafting and the grafting conditions(green house or open space).

Genotype and Genes

Gene responsible for a phenotype can be in different forms called Alleles. The different combination of the different forms of a gene will give rise to different genotype of the same gene. Different genotypes of the same giene will give rise to variation in a particular phenotype. So genotypes cannot be discussed meaningly if the gene that form them is not mentioned. Genotypes are dependent on their gene and the different forms of the gene that combine to form the said genotype. A gene will therefore be a FK in genotype.

Research Work Info and response Type

Since research work will look at abiotic or biotic or genetic factors affecting yield or quality of rootstock, there will be a direct relationship between the table that holds direct and indirect information on all these and research work table which is the Response Type table.

Eventually the closely related data are how a user is likely to access the application and hence the ease of data retrieval.

When the data is retrieved from the response type which is heavily linked, data will be light weight since the data will be represented by integers.

User And Group tables

A group will not exist without a user. So a group table is dependent on the user table. The PK of user table will become the FK of the Group table.

SECURITY

CONFIGURING JDBC REALM ON GLASSFISH SERVER IN ADMINISTRATION CONSOLE

Realm is a security term that indicates that server has to be configured for handling security.

Steps.

Switch to services tab in netbeans

Find glassfish server and ensure it is started. The green chevron icon will show.

Right click on glassfish server icon to view Domain Admin Console.

Get the authentication password and username for Admin console by clicking on properties after right clicking. Use these details to login in. show password details to see the password of the glassfish server4.0 admin console login for the instance you have.

This will bring up the administrator screen for the server. In the navigation area, click on configurations> Server config> Security> Realms

Need to create jdbc-realm

You will see on the realm page default admin-realm, certificate and file. To create jdbc realm, click on New. Filling the dialog box with the details below:

Realm Name: jdbc-realm

Classname: Click from drop down com.sun.enterprise.ee.auth.realm.jdbc.JDBCRealm

Properties Specific To This Class:
JAAS Context: jdbcRealm (This is the identifier for the login module to use for this realm)

JNDI: jdbc/vegrootstockproject (JNDI name of the JDBC resource used by this realm)

User Table

Vegrootstockproject.user (This is the name of the database table that contains the list of authourised users for this realm. It I the schema name followed by the table name.

User Column Name

user-username (This will be the name of the column in the user table that contains the list of usernames)

Password Column

user-password (Name of the column in the user table that contains the user passwords)

Group Table

vegrootstockproject.group

Group Table UserName Column

user_user_id (name of the column in user group table that contains the list of group for this realm)

This is the PK of the user acting as FK in the group table.)

Group Name Column

group-name (Name of the column in the group table that contains the list of group names.

Password Encryption Algorithm

AES (This denotes the the algorithm for encrypting the passwords in the database. It is a security rik to leave this field empty.

We will use the default Digest Algorithm (SHA-256)

The default use to be MD5 in glassfish versioins prior to 3.1. so we will not store anything in here because it is default

On the charsets used type UTF-8

We can skip the rest of the fields.

Then click the save button.

Once changes are saved, we can exit screen and stop the glassfish server. Otherwise the changes wont take effect until the server is restarted. The JDBC realm is set for our database connection to glassfish.

CREATE MAVEN WEB APPLICATION AND CONFIGURE IT FOR JSF, JPA, DB CONNECTIOIN POOL, SECURITY AND UTILITY LIBRARIES

After the creation of the jdbc realm, a maven web application is created and named vegrootsockproject.

File> New Project> Maven> Web Application

We make sure server is Glassfish 4.0 and Java EE7. Finish

Then we have our project.

In project files in our web application project, we have a POM.xml that controls the libraries that are used.

Now we will configure our project to use java server faces. If we were not using glassfish as application server, we will need to modify pom.xml and add jsf libraries as dependency but because we are using glassfish we don't need to add these libraries because they are already built in. So all we have to do is tell our project we want to use jsf.

Right click on project>Properties>Frameworks>Click add. We will pick java server faces. Now we wait for jsf2.2 to appear as server libraries and for registered libraries to appear in greyed out text there. You can proceed into primefaces from components.

Delete the plane html file in project and welcomeprimefaces file. Now we will add any additional additioinal libraries that are neceassy in our pom. Open pom, look for google guava dependency.

Google guava is a utility library and is used for text manipulation and is used for encrypting for security.

<dependency>

<groupId> com.google.guava </groupId>

<artefact Id>guava</artifactId>

Version>13.0.1</version>

</dependency>

Write this dependency into our new project within the dependencies of pom.xml and Save

Next we need to consider the contents in our website that need to be secured. Web pages folder is the root directory for webpages. At present we will only have index.xhtml for our homepage. Anything in the webpages folder is public but if you want secured content for different people in different roles, then you would have to create separate directories for that. So we will create a new folder.

Right click web pages>

New>Other>other>Folder>Create admin folder and user folder.

You can create any folder you want but we will use admin folder for contents only admin group can view and user folder for contents only user group can view.

You can create many directories with as many terms as you want but the idea is that we segregate the content for different groups of login users.

So that way, when we do our configuration, we can say anything in the admin folder you need to be in the administrator group to view it or for anything in the user folder you need to be in the user group to view it. But then anything outside these 2 folders, you don't need to be logged in, it is public. So that is the easiest way to configure our project.

OPTIONAL CONFIGURATION

When we first created this maven project we had the web-inf directory and once we added the jsf capability we had the web.xml file.

This is one configuration file we need. It is used to configure our java server faces and the home page.

Faces Configuration

Right click on project> New>Other>JavaServerFaces>Faces Configuration>Accept defaults.Next >finish

At present there is nothing in there, but it can be used for Internationalisation. Now we will not use it, but we will have it available if we need it.

Glassfish-web-xml

Next configuration, New>Other>Glassfish>glassfishdescriptor>next>Defaults>Accept

It will add the glassfish -web-xml. This will be used for some of our security settings

Contexts And Dependency

Click on New>Other>Context and Dependency>bean.xml (CDI configuration File)>Next>Accept Defaults

Then you see your beans.xml. This file is necessary for CDI which is the new model of doing jsf and it also allows us to do more advance dependency injection and some other things. It is good for the future.

Another recommendation is under other resources where we will add directory to store our database backup files. If you want to restore database from backup it is nice to have them in your project under source control by get. Right click project> New>Other>Folder>Type in folder name>Browse for main under source and this add the new folder. Copy and paste items you want to add to this folder. This is a great way to protecting one's self of their hard work.

Now we have all the config files under WEB-INF

Always ensure not to create web pages in the web-inf folder, since this folder is only meant for .xml files

HASHED PASSWORD GENERATOR

Creating the HashPaswordGenerator:

In source package, create the java package by right clicking on source package.

Insert the java class body by right clicking the java class. Write the java code in here to generate encrypted passwords for the tables.

With the database tables created, we will have to replace the clear text passwords with Hashed PasswordGenerator.java. Open and run the code for Hashed Passwords

The following code was implemented:

Package vegrootstockproject.glassfish.jsfsecure.util

Import com.google.common.base.Charsets

Import com.google.hash.Hashings

Public class HashedPasswordGenerator{

Public static void generateHash(String password)

{String output = Hashing.sha256().hashString(

Password, Charsets.UTF_8).toString();

System.out.println(output);

}

Public static void main(String[]args){

generateHash("password");

The HashPasswordGenerator is mandatory. Google guava was downloaded as a jar but for the use of a Mavan Web application project, the guava libraries were obtained from guava-libraries and added as a dependency in the POM.xml file of the project vegrootstockproject Maven Web application created. SHA256 is a way that passwords are encrypted and cannot be decrypted. In this way, any sensitive data in the database will not be available for unauthorised users who want to retrieve data. So in the mysql databases tables created for vegrootstockproject(Schema Name), the users table that holds information on passwords will be replaced by the encrypted passwords generated by the HashPasswordGenerator.

CREATING JPA ENTITIES

Creating of JPA Entities is the way a connection is established between a web application and database on MySQL server. Our Maven web application uses the database on the MySQL Server through entities which are managed or configured at the lower level by persistence.xml.

JDBC establishes the connection between Glassfish server which is an open source server, and MySQL server which is also open source, to allow access to the the vegrootstockproject Database tables. These tables are called JPA entity classes between the glassfish server (using java ee7 api) and MySQL server. This is the language they mutually understand for the table in vegrootstockproject database hosted on the MySQL server. The MySQL server would have credentials of the computer on which the MySQL platform is installed to pass the database information. This way the full credentials of the database source will be known to the MySQL server.

CONFIGURE APPLICATION FOR DATABASE ACCESS USING JPA

We need to have firstly a connection pool on the server that connects to our database for performance efficiency.

We can create our configuration file that we can use to automatically install that on the server regardless of what computer you are on. This we can do by first clicking on Seervices tab in netbeas. Have MySQL Server connected (under databases in netbeans, MySQLServer at localhost:3306).

Ensure that Netbeans is connected to MySQL Server

We will want a connection to our database vegrootsockproject. From the click down node, you choose the db you want to connect to. Create the connection in netbeans by simply right clicking and say connect. You can also do this by Rightclicking on MySQL J-connector under drivers, enter the dbname which is vegrootstockproject as discussed earlier.

Enable the connection then once in there we can look at the tables

DATABASE CONNECTION IN WEB SECURITY

So once we have our connection established under the services tab, we want to connect to the jdbc realm for database security so we return to our project.

Right click project>New>Other>Glassfish>JDBC Resource>Create New JDBC Connection Pool

Fill in Custom JNDI Name. We will call this jdbc/vegrootstockproject. This will be the same as the JNDI name for our jdbc realm.

In Resource, we will name the connection pool and choose which jdbc connection pool we want to use for our project. Then we will need the resource that connects to our connection pool. JNDI name must be consistent because it is an identifier for our connection using the connection pool.

The JNDI name must always be preceeded by jdbc/* and you can customise the rest. Where * means the rest. Note that this information must be same as you have on the realm.

We will callt the connection pool vegrootstockprojectPool

Extract from existing connection, we will select the pool that connects to the vegrootstockproject database which is jdbc:mysql://localhost:3306/vegrootstockproject

Click Next

Resource Type

Choose javax.sql.ConnectionPoolDataSource from drop down and for

Datasource Classname:

Com.mysql.jdbc jdbc2.optional.MySQLConnectionPoolDataSource

Ensure all the properties are correct including the password and username to the MySQL server and the URL of the database to glassfish which is jdbc:mysql://localhost:3306/vegrootstockproject

Click Next

If you have loads of users simultaneously, then you want to up the default values. We will leave the defaults for now.

Now we have created the connection pool and jbdc resource

NB: Each project must have its own configuration on pertaining a particular jdbc-realm configuration. So the JNDI name and the URL must be unique to project, otherwise errors are bound to happen with multiple porojects have the same security configuration.

When you successfully apply your security configurations to a web application project, this project will not need any further configuration whenyou configure through the JDBC Resource on glassfish for the project. So it can be deployed on any other computer because the settings on the servers will remain same so far as the mysql server link is not destroyed on the initial host computer.

When we close our application and reopen it, we should see under Other Rsourcesa newly formed setup folder. In this folder you will see the glassfish-resources.xml

This xml file points to the poolname created, the jndi name, then the db connection, username and password to connect to that.

So what happens is this xml file is used when you ran the project to create the connection pool and resource on the server. If it does not exist on the glassfish server it creates it, if it already exist it does not. This way, no matter where you ran your project or what computer you ran it on it will automatically make sure you have a connection pool configured and running on the glassfish server.

The reason we want to use a connection pool is for performance efficiency because connections are expensive to create, so this glassfish-resources.xml will ensure we re-use existing connections for performance benefits.

PERSISTENCE UNIT

Now for JPA, we need a Persistence Unit that works with our connection Pool. Rightclick on project>New>Other>Persistence>Persistence Unit>Next>We can change the default name but ensure it ends in PU(we will use vegrootstockprojectPU.

For persistence provider we will keep Eclipselink as default

Datasource = This will be our JNDI name which is jdbc/vegrootstockproject

We will ensure that use Java Transaction APIs is ticked

If we wanted to create tables, we could have kept the create ticked under table Generation Strategy, but since we are not creating any tables, we click no. This is for creating tables automatically from enetity classes. Click finish

We will now have a new folder under other sources under main resources inMETA-INF folder.

You can look at this as an .xml file. The graphical view is default. And this is all we need to start using JPA

In summary, glassfish-resources.xml establishes our connection pool whilst persistence.xml uses our JNDI name to talk to that connection pool so that when we use JPA it will know which database to talk to.

NOW WE NEED A LOGIN PAGE AND AN ERROR PAGE THAT DIDSPLAYS ANY LOGIN PROBLEM

We will create the login and login error pages and add them under web pages. The .xhtml sheets are attached as files for viewing. We will use jsf tags and plain html tags. Normally we will want to avoid the use of plain html tags but it is mandatory if we want to make use of the built in glass fish security system. In that case we have use a normal html form tag and the advantage is that we can set the action property to point to j_security_check. It must be spelt exactly like that. This is the name of a hidden built in servlet that is in glassfish and what that means is we don't have to write anycode to process the login.

But this does not work from a jsf form tag. We can use some jsf tags for layout purposes. But we have to use a normal html input tag for password and username if we want to use the j_security servlet. <input type “password” value=”password”/> nad <input type “username” value=”password”/>

If you use the jsf version for submit button, it will not go to the j_security_check servlet. But by using <input type=”submit” value=”login”/> it will go to the j_security_check servlet.

ERROR PAGE

This will be the page you will see when someone’s login fails. We will configure in web.xml. Go to security. Under login configuration, Set type to form**.** Identify the login page by browsing for it and select login.xhtml. This shows up as /login.xhtml but we will have to change this to /faces/login.xhtml since this is a java server faces project and will not process correctly if it does not have this is in the lead, because in source and under servlet it expects all java serverfaces pages to start with faces/* for the url

Browse for error page, choose loginerror.xhtml. Correct to /faces/loginerror.xhtml

Realm Name= jdbc-realm

Under welcome files where for error login we will put the code 403 and choose thelogin error.xhtml file

Error page will have a text to value which will redirect users to the homepage ie unitus and this will be in a hyperlink.

To Home Page (for outside links)

Or for retry <h:link outcome=”faces/login.xhtml” value=Retry/>

We will not use <h:link outcome=”faces/index.xhtml” value=Homepage/> since this is used often to link to pages within the application.

SECURITY ROLES

These are the roles that our groups created in db will belong to. We will add a role named admin. It is advisable for the role names to match the folder names created in webpages. We will then add another role named user. So we have two roles that matches the two folders we have created.

SECURITY CONSTRAINTS

DATA CONSTRAINT

We will have one security constraint for admin role and one for user’s role

ADMIN AND USERCONSTRAINT

We will change the display name AdminConstraint

Click add under WebResource Collection. Resource name = admin

URL pattern:/faces/admin/* (meaning all jsf pages in the admin folder)

Enable Authentication Constraint

We choose this to enable the roles that are allowed to access content

We will link the admin role to have access to the URL above.

We will do the same for user role but the UserContraint will have /faces/user/* and the roles that will have access will be both admin and user. In this way we are giving admin overall access to the web application created and hence access to the user folder which user role is assigned to.

TRANSPORT GUARANTEE

We set transport guarantee to confidential. What this does is it enable secure socket layer, which is the thing that encrypts all our dataflow between client and server. We will do this for both UserConstraint and AdminConstraint.

Finish

GLASSFISH-WEB-XML

We will now click on WEB-INF to glassfish-web.xml, click on security

You will notice that you have security Role Mappings for admin and users that matches up with what we did earlier. But we need to configure these

Map these names(roles) to the group names in db

Add Groups user and admin to the roles user and admin respectively

Finish.

THE MAIN WEB APPLICATION FOR DATA RETRIEVAL

Now we have a MAVEN application. As in all Maven applications, the POM file defines the project structure which is an .xml file.

In there we can see that we have a dependency set on on java EE which is java-web –api which is seen when you click on dependencies, you see javaee-web-api-7.0 jar.

INDEX.XHTML FILE IS REGISTERED AS THE WWELCOME FILE IN THE WEB.XML FILE FOR OUR PROJECT

When you look in the web.xml we will see that the facelet servlet registered and also we have index.xhtml file as the first page we will see set as the welcome file. We can open the web.xml file by right click and edit.

THE INDEX.XHTML FILE POINTS TO ANOTHER FILE WHICH IS WHERE OUR SECURED FACES CONTENT IS FOUND

In the index.xhtml file, which is the welcome page the user sees welcoming them to the COST ACTION FA1204 Database site. In here a link is available to the user as CLICK TO ACCESS OUR OPEN SOURCE SITE ON COST ACTION FA1204. A brief description on the difference between the database site and the open source site.

On this Database Welcome Page, there is a LOGIN AS A USER link and LOGIN AS AN ADMINISTRATOR link which will direct user to the login page where they will login into the USER site or the administrator site respectively. The outcome of these values “LOGIN AS A USER” and “LOGIN AS AN ADMINISTRATOR” is linked to faces/user/index.xhtml which is a welcome page to show users that they have successfully logged in into the user’s site and on the user’s site page will be the link to either access the Research Work page or the Database On Vegetable rootstock page . These values will be linked to the faces/user/userviewresearchwork.xhtml or the faces/user/userviewvegrtskdb.xhtml page respectively.

When an Administrator logs in successfully into the Administrator’s site, there will be a welcome message that states that they have successfully logged into the Administrators site. Over here there will be links to research work for both admin and user view and also a user and admin view for vegetable rootstock database using the links

1. faces/user/userviewvegrtskdb.xhtml
2. faces/user/userviewresearchwork.xhtml
3. faces/admin/userviewvegrtskdb.xhtml
4. faces/admin/userviewresearchwork.xhtml

The primefaces interface which uses the primefaces component library is taken advantage of with respect to the data display pages from our database for both user and admin. A start to project with header, footer and a Big space to display the Welcome Text and the database.

GENERATE JAVA CLASSES REPRESENTING THE DATA IN OUR DATABASE

We will pull data from the database in Netbeans SQL Server following the guidelines and specifications provided by javaee7 into our primefaces application.

SPECIAL CODE GENERATORS ARE INCLUDED IN NETBEANS IDE FOR GENERATING JPA ENTITY CLASSES

We need to get hold of this data and represent it in some way in our application, following javEE7 and the way is via the JPA specification java persistence annotations. We do this by right clicking on the application in Netbeans> Click New>Entity Classes from database

NETBEANS IDE LETS YOU SELECT A DATABASE CONNECTION AND THEN SHOW YOU THE TABLES FOUND THERE

We find that there is a template called entity classes from the database and for each of the tables that are in our database, we can choose the tables that we want to use in our application. We will choose all the tables in our database. We will realise that for each table we select over, Netbeans pulls along its related foreign keys. It is necessary to note how the names of the classes are compared t the database table. Pay attention to the location which is source packages and the package name which is com.company.vegrootstockproject which is inherited from the project name vegrootstockproject. Click Next

SPECIFY A PACKAGE AND ALSO NOTE WE CAN REGERATE THE CLASSES LATER, WHEN/IF THE DATABASE CHANGES

After clicking next we will specify a package where for each table we have selected, a java class will be created. Following the JPA specification, there will be one so called ENTITY CLASS for each of our tables and via these entity classes our lower level data(attributes) will be accessed

NOTE THE “GENERATION TYPE” COLUMN , WHICH IF THE CLASSES ALREADY EXIST, CAN BE CHANGED TO “UPDATE” TO OVERWRITE

In the Entity classes Dialog box, there is a heading called Generation type. If our data actually changes(data under the tables) we can go back into the wizard and switch from new to a different setting ie UPDATE or CHANGE so we can override the classes that we are now creating when our database changes.

Click Next

HERE OTHER OPTIONS CAN BE SET, TO REALLY FINETUNE THE CODE THAT WILL BE GENERATED

You can see that we can specify what kind of collection are used (eg we can choose from java.util.collection eg fully qualified DatabaseTableNames, so we can fully customise what is going to be generated.

Click Finish

NOW THE JPA ENTITY CLASSES WILL BE GENERATED USING THE SETTINGS YOU PROVIDED

And for each of our tables we now have entity classes. The entity classes will show under source packages>com.mycompany.vegrootsockproject as .java which are the classes. So we will have

Vegetablecultivar.java, researchworkinfo.java, vegetablerootstockgenotype.java, vegetablerootstockgene.java, abioticstress.java, bioticstress.java and Responsetype.java.

Each of the tables now have a java class that represent it in the java application. Each java class has Annotations such as @Entity making this an Entity Class and there is a link from below and mapping from this class to the table named VegetableCultiver , VegetableRootstockGenotype etc

NAMED QUERIES ENABLE YOU TO EASILY ACCESS DATA, WITHOUT NEEDING TO REMEMBER LENGTHY SQL QUERY STRINGS

We have several so called named queries which are also part of the JPA specification, mapping the sql query to a more readable, more usable format.

WE WILL USE VegetableRootstockGenotype.FindAll As An Example

For instance we will use VegetableRootstockGenotype.findAll to pull all the data from genotype table. So instead of saying the whole SQL query we can use the jpa named query

THESE ANNOTATIONS COME FROM THE BEAN VALIDATION API e.g @Column specifies the number of character allowed

Moving down we can see also that there are annotations for validating the columns in the table. In otherwords if in the case of the VegetableRootstockGenotypeName Column, if an administrator fills in the column with more than the characters it is set out to contain, an error message will be displayed to the user

THE @PATTERN ANNOTATION LETS YOU SPECIFY A PATTERN THAT THE VALUE MUST MATCH TO BE VALID

Pattern matching is also supported, so in the case of a phone number foe eg netbeans suggest a pattern that should be matched inside the field that the user will be typing in when entering data back into the database which should match and if not, data would not be entered into the database.

FOR THE REST, ALL IS A PLAIN OLD JAVA CLASS, WITH GETTERS AND SETTERS

All kinds of validation checks can be set in other words, for the rest, this is a plain old java objects with getters and setters and right at the end the equal and the to String. This is the same for all the other java classes. Plain old java objects from below; standard domain objects or business objects; annotated with special JPA annotations and Bean validation annotation

FOLLOWING THE JPA (JAVA PERSISTENCE ANNOTATION) SPECIFICATION, THE IDE GENERATED A PERSISTENCE.XML FILE

As seen in earlier configuration the persistence.xml will now have a tick here that shows that the xml file includes all Entity classes in the project application. The persistence.xml is an important file that controls the classes created. The convenient thing about JPA as a tool is that it is very easy to identify for Netbeans which of the classes are actually Entity classes because they are marked with the @Entity annotation.

ALL CLASSES ANNOTATION WITH @ENTITY ARE ACCESSIBLE VIA THE PERSISTENCE.XML FILE

So everything inside the application with the @Entity annotation is by default accessible via this persistence.xml class and that is how we are able to access the 7 classes of of our project and the underlying data by referring to this .xml class.

CREATE A SERVLET TO CHECK WHETHER THE DATABASE CONNECTION IS WORKING AS EXPECTED

So to prove the work of of the persistence.xml in connecting the classes to the underlying data in database, we will create a servlet. Right click source packages>New>Servlet

The servlet I going to test our database connection

The servlet is called VerifyJPA and is made to check the application before work was began on the visual layer or the controller layer, to confirm that we have a good connection that is working the way we want.

Location is Source packages and the package we called called servlets.

ClickNext.

RATHER THAN REGISTERING THE SERVLET IN A WEB.XML FILE, WE ARE GOING TO LET THE IDE CREATE AN ANNOTATION IN THE SERVLET CLASS

In servlet 3.0 we will annotate the servlet with @WebServlet to register it, rather than using the web.xml to do so. We will run the created servlet and when we receive a message on the server saying verifyJPA Servlet then it means the servlet is deployed. We will now use the servlet to connect to the database.

To Support Filtering Data

VegetableCultiverService.java(VCS) and Filter.java work hand in hand. The VCS constructs the instance of the random value chosen by user and how the return pattern should be. This is the session bean or controller or Enterprise java bean(EJB). This is passed to the Filter.java which uses the session bean ie VCS which is incorporated into the Filter.java or the Entity Manager.

The Entity manager uses the @NamedQueries to retrieve the required instance value.

Since the request of the session bean is to produce the list as defined, the record for this value will be used to fill out the required list of the session bean which is how the tables are displayed in the front end.

So the java class VegetableCultiver will integrate as an object into the XHTML static object on the screen and retrieve all the values of the objects into the database using VeetableCultiver.'the attribute')

VegetableCultiverService will record the instance produced from the filter() and input value from the homescreen to create the instance. This is presented as a view to the dtFilter to use the EntityManager to collect this data value from the database. The requirement of the session bean on how this should be displayed is passed to the Filter.java which has Entity bean than retrieves data from database using persistence.xml.

The display is a record of the instance value in an output as defined in the VCS which is the session bean

Targets For The Table Met

Show values under the different attributes

Allow an individual to choose a value from a drop down list

The value chosen should populate the table with the record of the chosen value

Installation Of GlassFish Server And Netbeans

Java SE Development Kit 8.0 or 7.0 is required before installing Netbeans 7.4 and this JDK comes with some glassfish server bundles and Netbeans IDE bundles. The JDK configures glassfish and aids glassfish in carrying the API which is used by glassfish in the Netbeans IDE. The instance of Glassfish on one's local computer after installation uses the admin port 4848 to link to the admin console that has components that iinteract with the domain glass fish server. The security component of glassfish is also accessed through port 8080. Netbeans IDE is the platform that uses the dynamic services of

glassfish server which the local computer will have access through port 8080t connect to the domain server and port 4848 through which the localhostwith Netbeans will have access to the admin console.

We will use proxy server settings if we are usng a different server apart from glassfish to deploy the JDK and Netbeans.

But if we want JDK to use glassfish on Netbeans then we don't have to touch the proxy server settings. The JDK with glassfish is program written such that we only need the ports t get access to the domain server. But if we are using a proxy server with of a commercialsort to deploy Netbeans IDE and glassfish, then we enter the IP address and port to access the server. For windows 8 we need JDK8 or 7 to install default glassfish with JDK to use Netbeans.

Deliverables:

A created database holding information about the use and development of vegetables rootstocks has been satisfied. The database allows for access for retrieval of data and addition of data.

The database has been implemented into the web application built.

MySQL 5.6 to store relational database and Netbeans IDE with GlassFish Server 4.0 and JDK8 were installed on the University's Server under the instructions of Professor Colla and the project manager Massimo Romanelli.

The web application built will be further discussed with Dr. Andrew Thompson in Cranfield University,who is a working colleague of Prof. Colla, upon my return to UK. Then deployment and maintenance to the server which now has the installations of MySQL and Netbeans all installed and configured by me, would take place upon the agreement by Prof. Giuseppe Colla of any other changes that may arise after my meeting with Dr. Andrew Thompson.

Link and benefit To Cost

I firmly believe that this project will extend collaboration between the group of Dr Andrew Thompson (Cranfield University), The Agricultural University of Athens and Tusci University, Italy where the focus would be the application of the database application which consists of bibliographic data, information about plant genotypes, information about plant genes, information about researchers, and information about Commercial activities. Ultimately this research in Grafting vegetable crops would be facilitated with use of the web database application that has been created.

All 3 teams are active in rootstock research but operate in different disciplines. The collaboration would help to build a multidisciplinary network of active research within the FA1204 COST action. I am pleased with my involvement in this Research and would welcome any further inclusions.

The STSM has enabled the development of this database that would make the availability of data easily retrievable and so would be the addition of data to the research project database. Information about the use and development of vegetable rootstocks would be readily available since all would

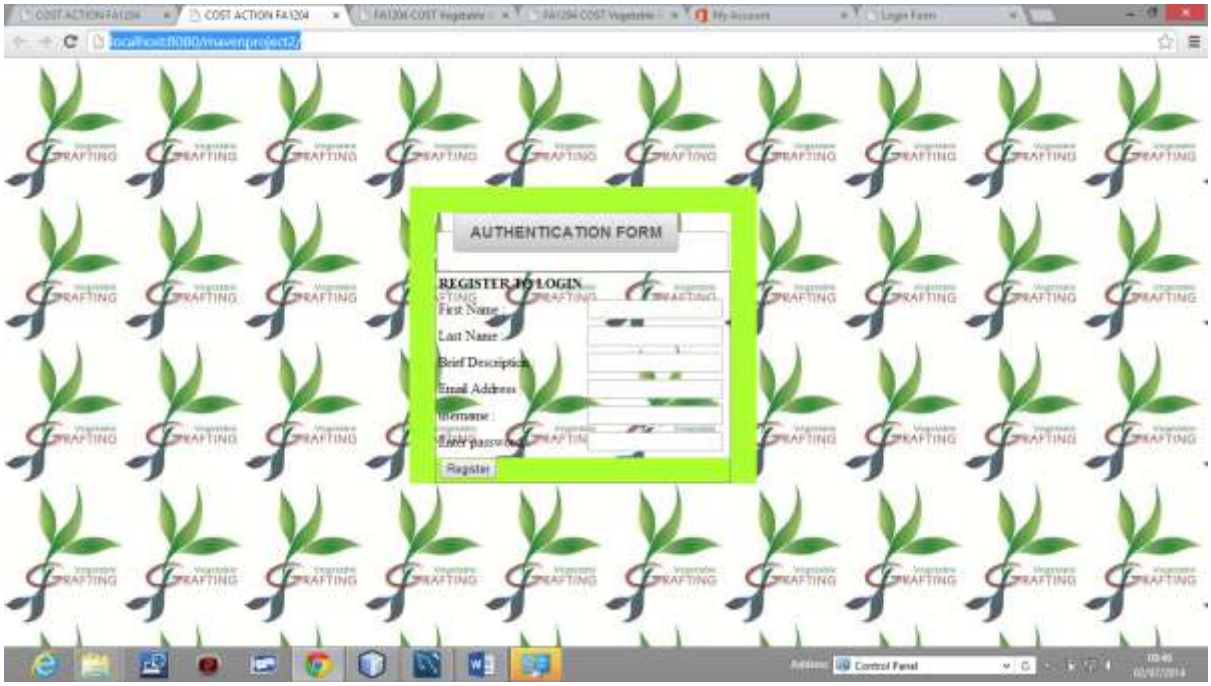
be gathered in one database that would be linked to a web application to allow easy authenticated user investigations.

How Does The Topic Fit Into The Goals Of The COST action?

This created database web application would be essential in research of grafting vegetables such that data can easily be assessed to enable comparison and analysis. Data can also be added to the database for later examination. Other research work related to vegetable grafting can also be analysed to ease studies. This would allow leads in the discovery of biomarkers that are key to the rootstock yield and resistance to soil borne diseases with respect to biotic and abiotic factors. Benefits from this research could be used by commercial breeders to select more vigorous tomato rootstocks using marker assisted selection. An understanding of the underlying genes could eventually be applied to other vegetable crops.

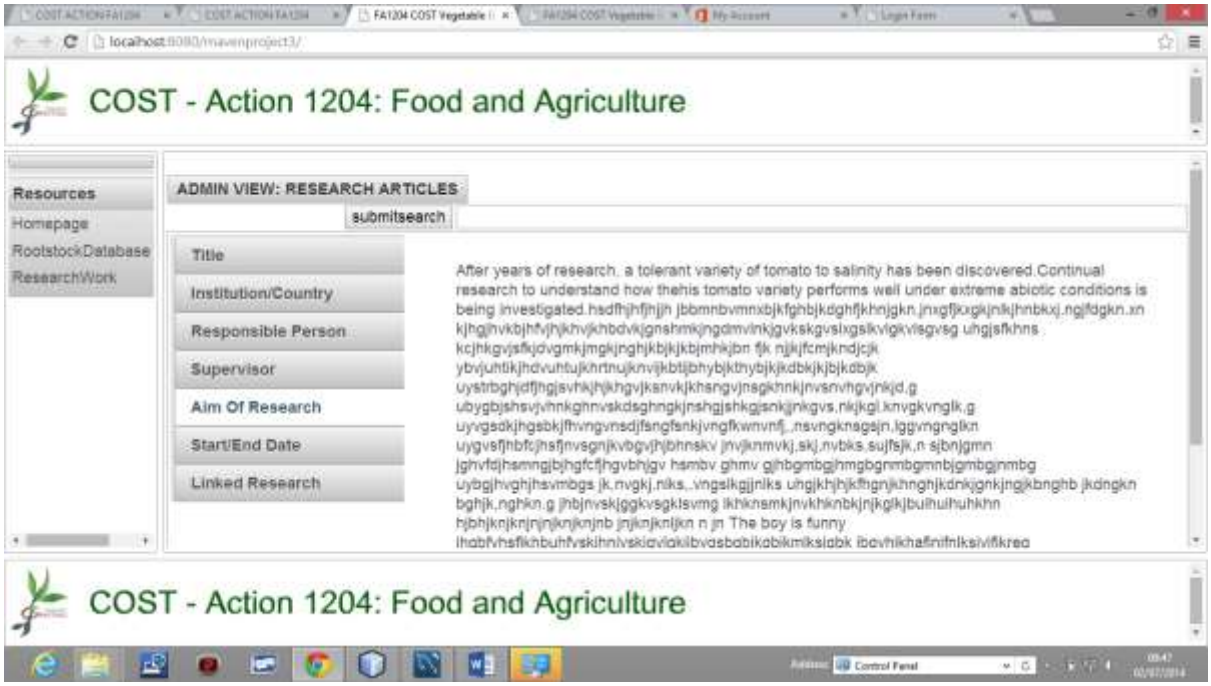
LOGIN PAGE

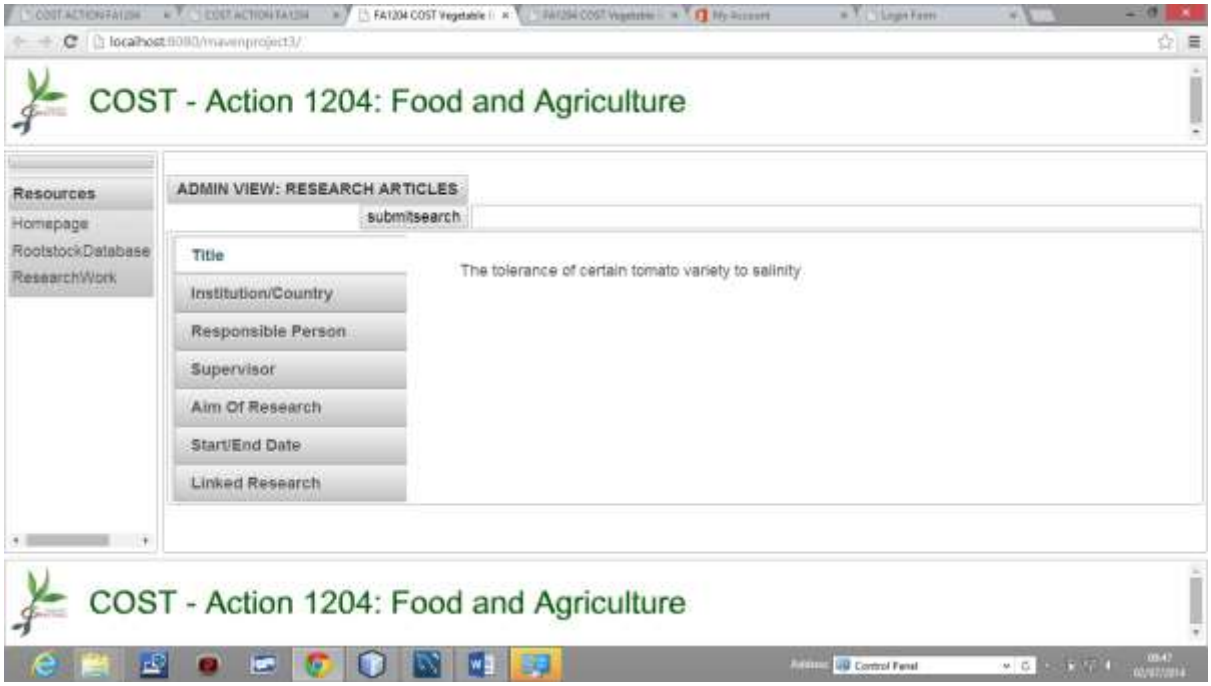
REGISTER FORM

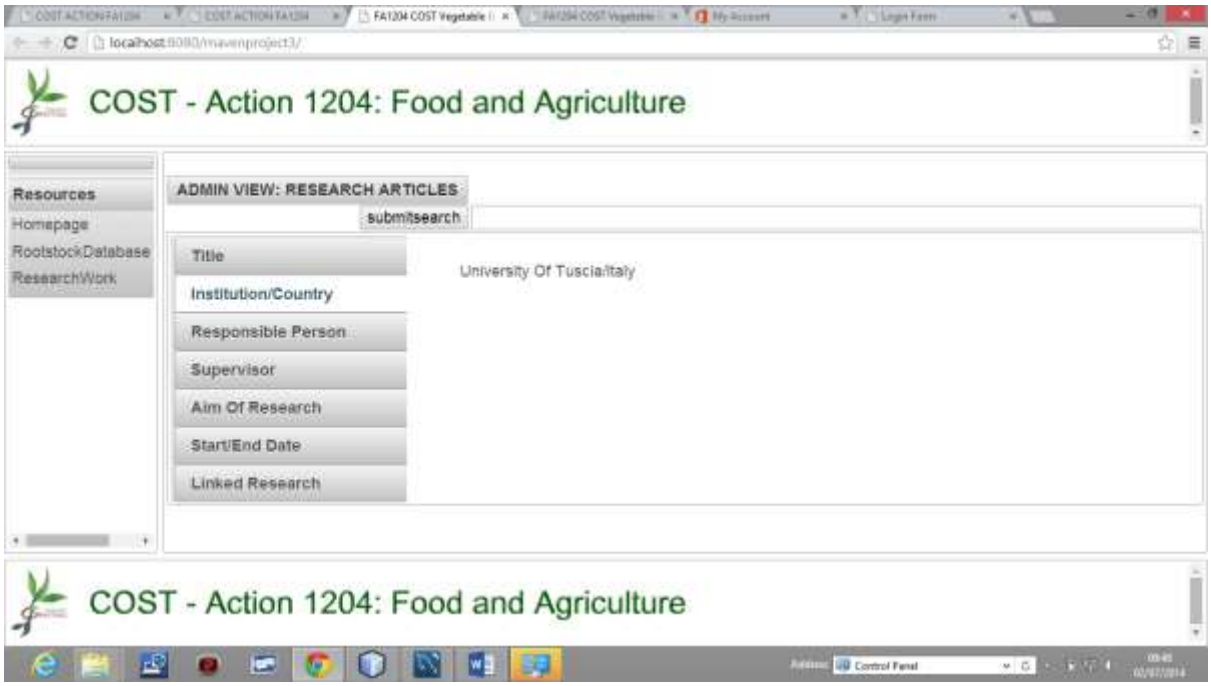


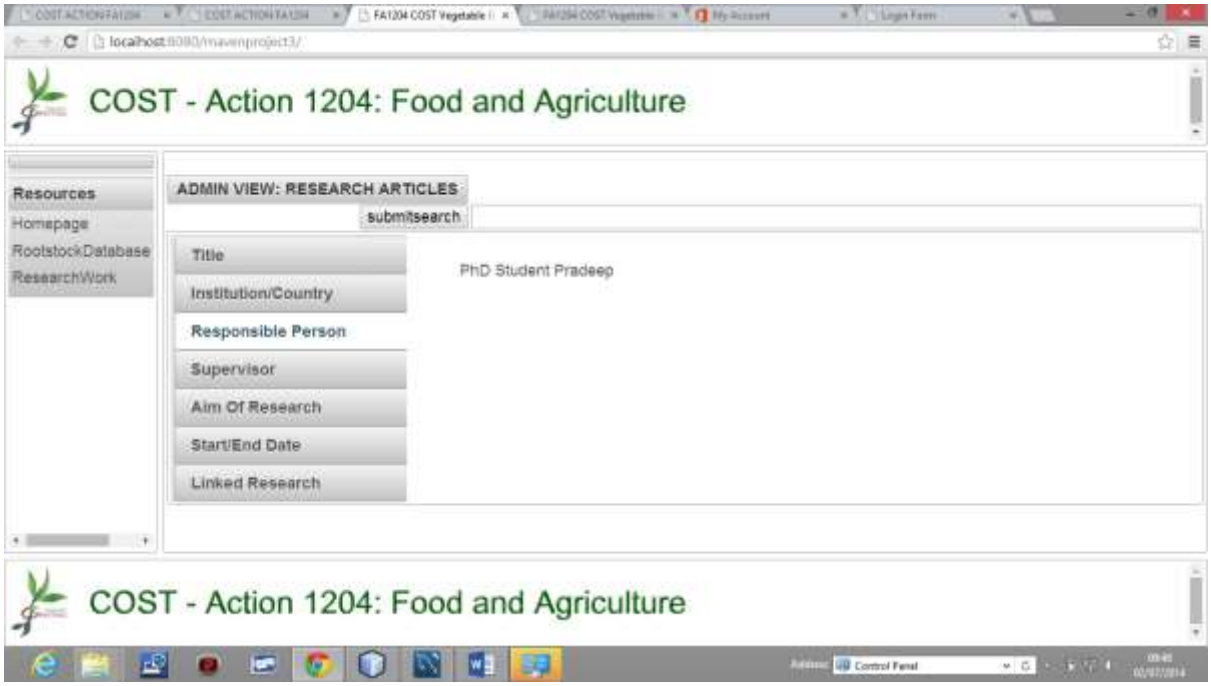
RESEARCH WORK PAGE

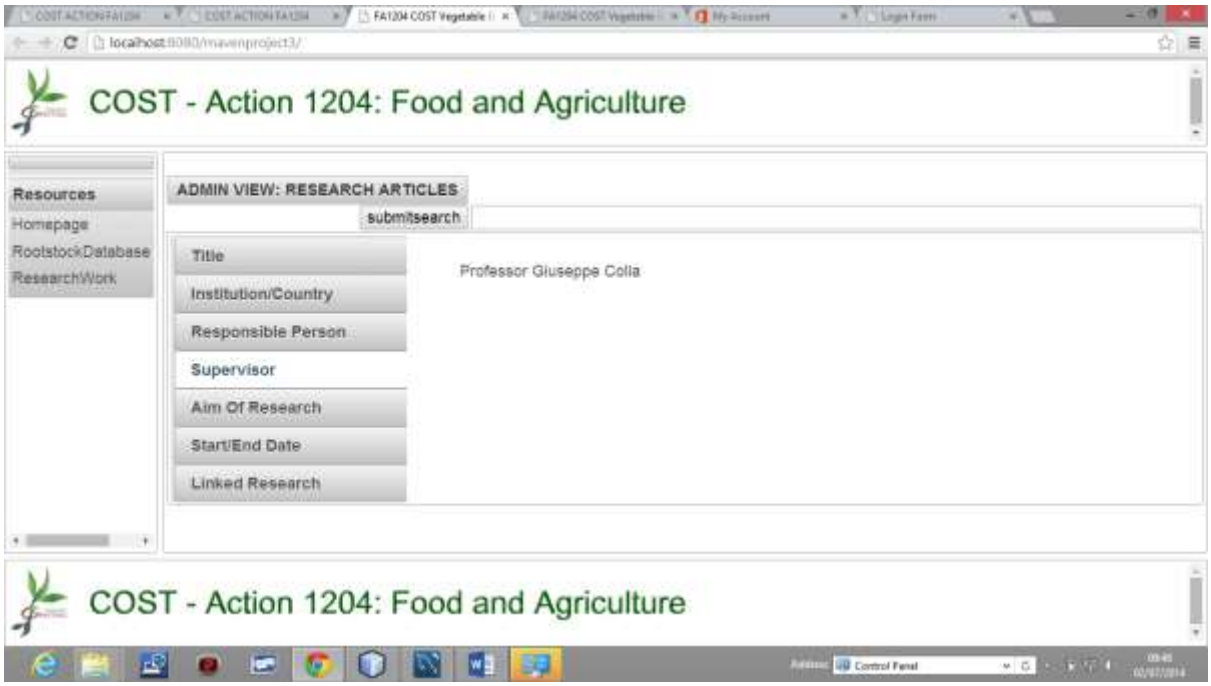
The pages below show how the tabs work in the research work page

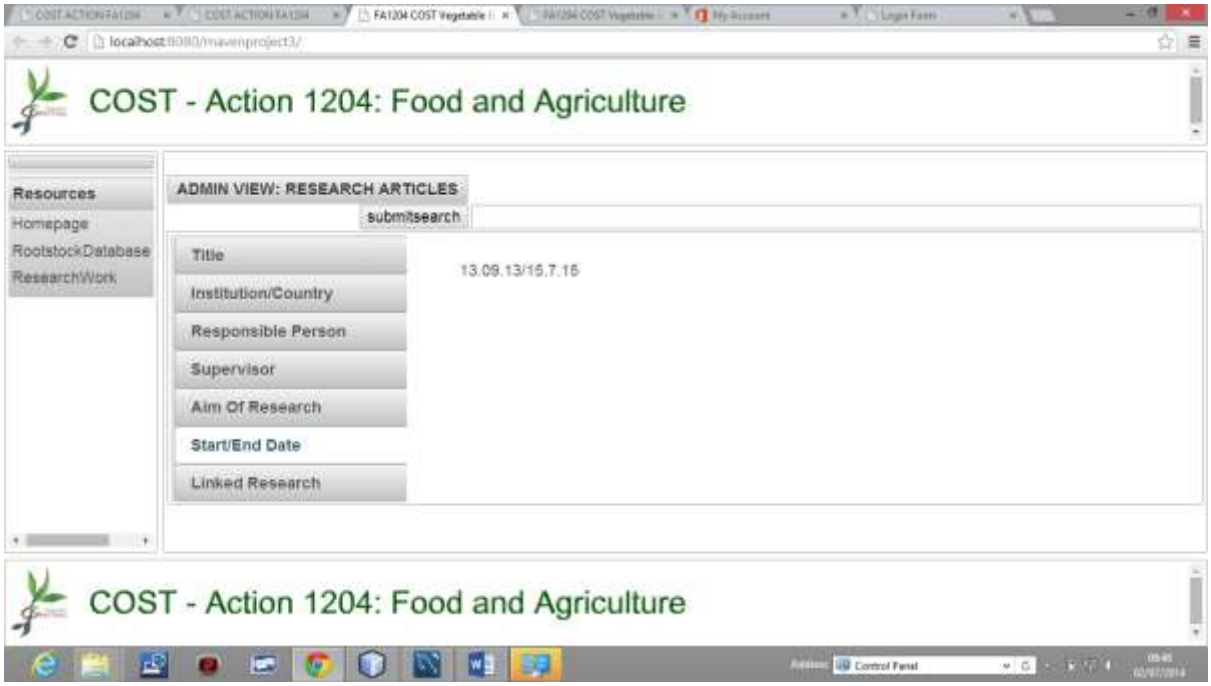


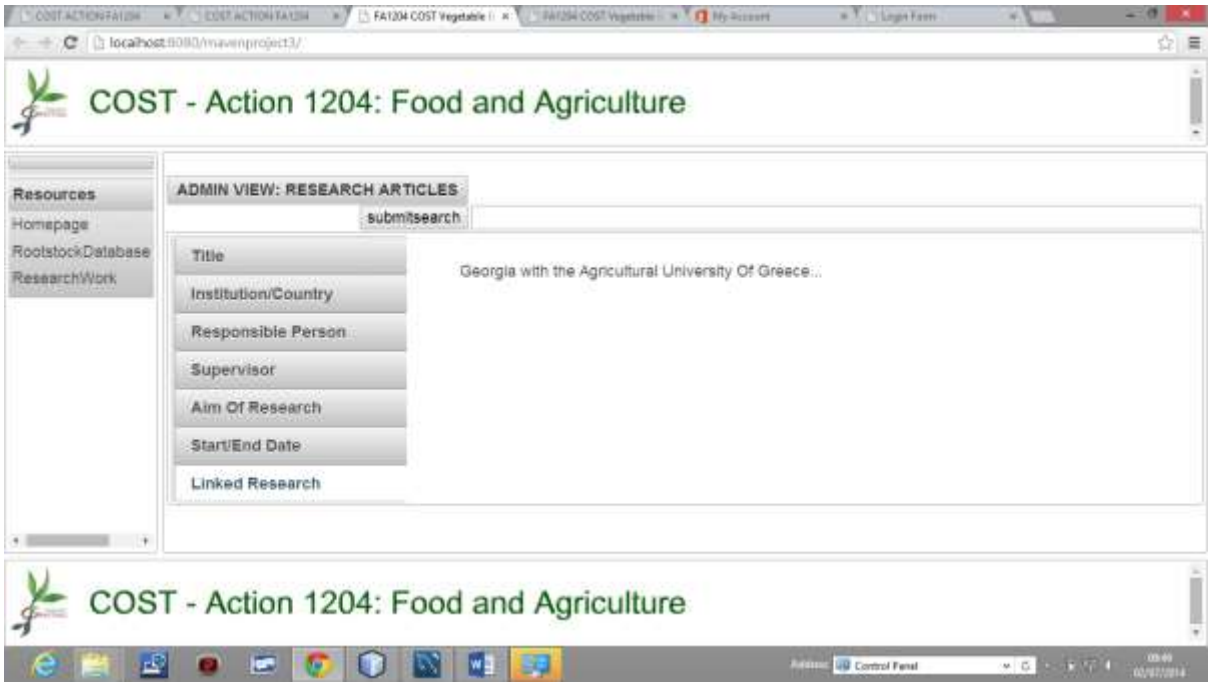

COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
The tolerance of certain tomato variety to salinity
COST - Action 1204: Food and Agriculture

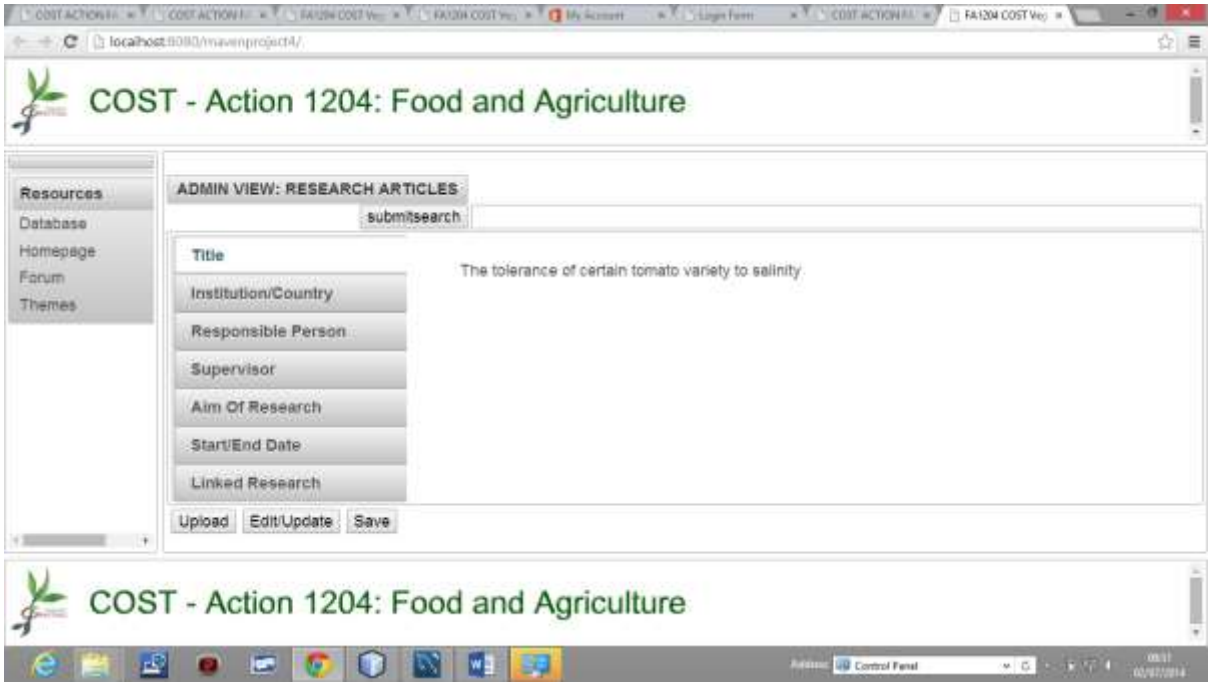

COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
University Of Tuscia/Italy
COST - Action 1204: Food and Agriculture


COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
PhD Student Pradeep
COST - Action 1204: Food and Agriculture

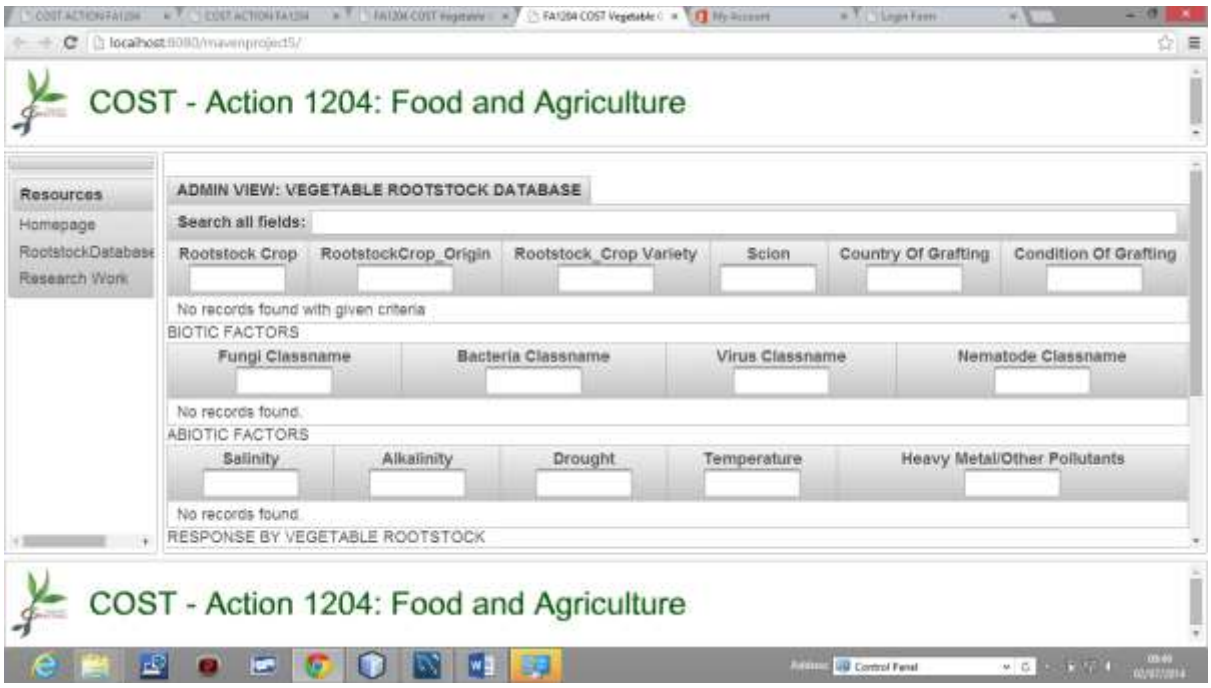

COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
Professor Giuseppe Colla
COST - Action 1204: Food and Agriculture

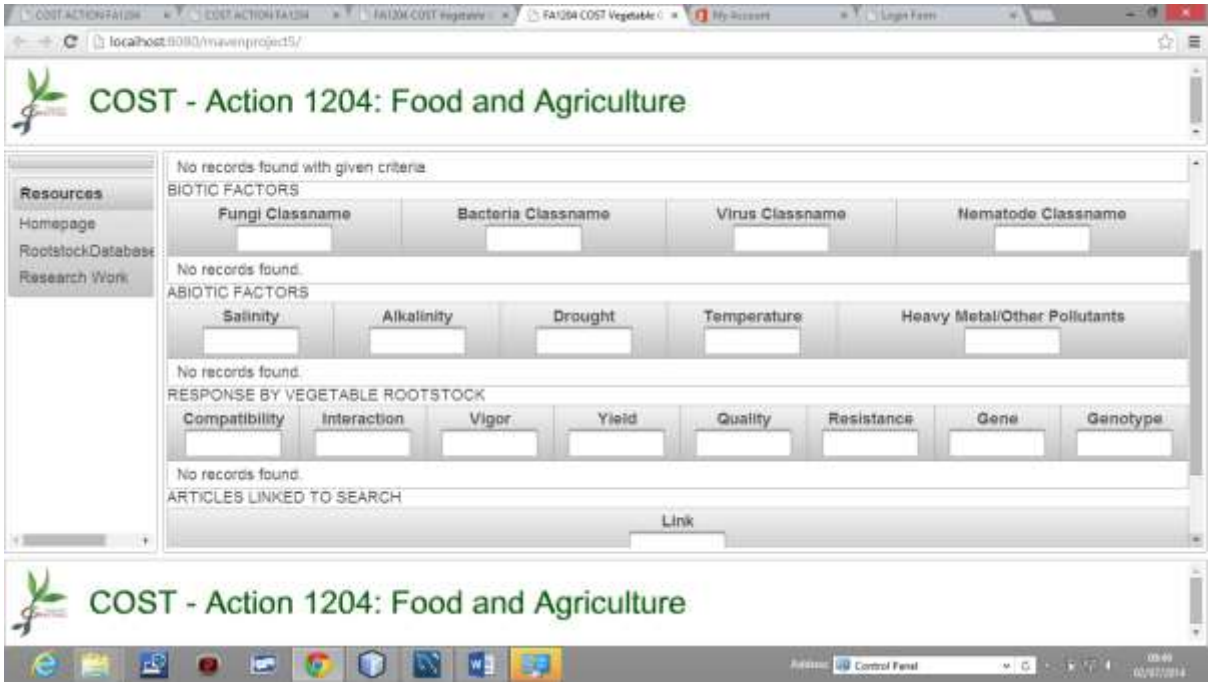

COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
13.09.13/15.7.15
COST - Action 1204: Food and Agriculture


COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
ResearchWork
ADMIN VIEW: RESEARCH ARTICLES
submitsearch
Title
Institution/Country
Responsible Person
Supervisor
Aim Of Research
Start/End Date
Linked Research
Georgia with the Agricultural University Of Greece...
COST - Action 1204: Food and Agriculture

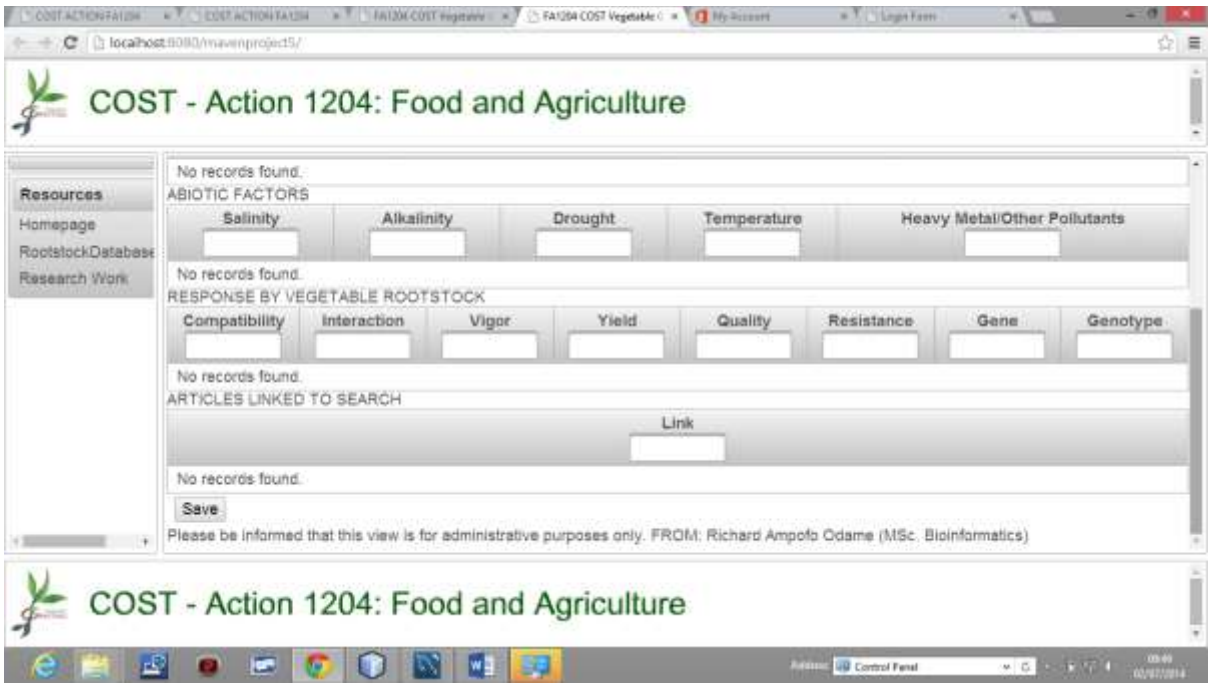


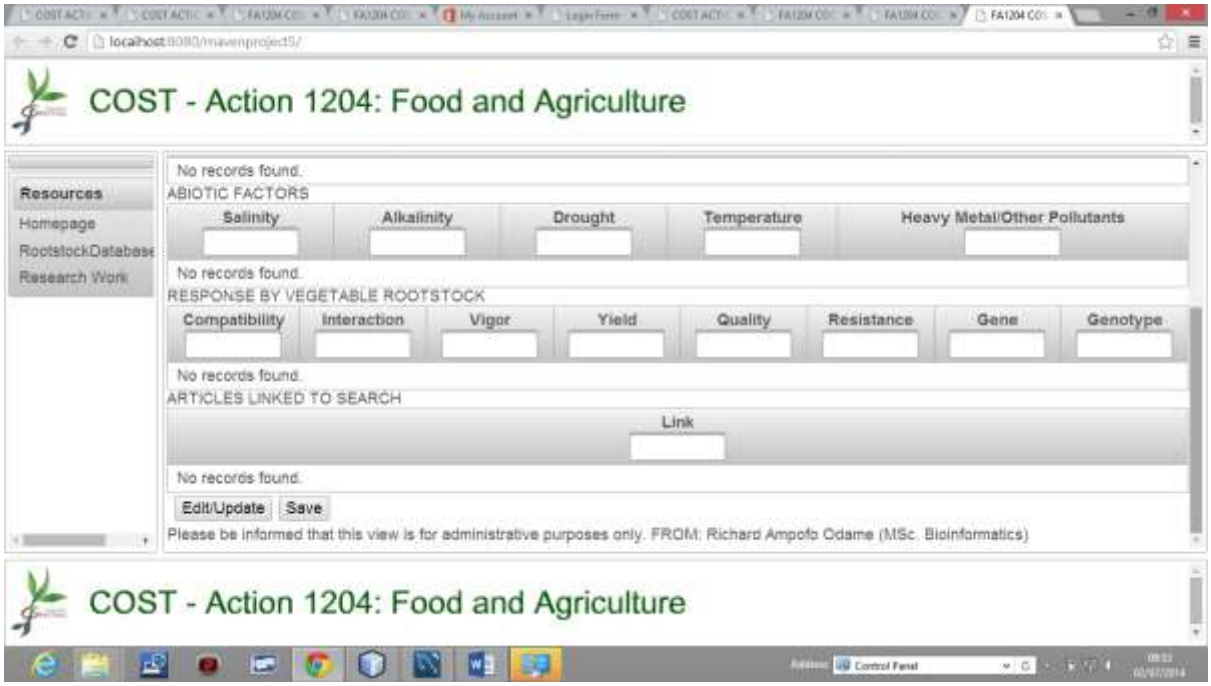
The Admin View For The Database Which Retrieves and Adds From/To Database





This Shows A User Page For View And Save Only




localhost:8080/mavenproject5/
COST - Action 1204: Food and Agriculture
Resources
Homepage
RootstockDatabase
Research Work
No records found.
ABIOTIC FACTORS
Salinity
Alkalinity
Drought
Temperature
Heavy Metal/Other Pollutants
No records found.
RESPONSE BY VEGETABLE ROOTSTOCK
Compatibility
Interaction
Vigor
Yield
Quality
Resistance
Gene
Genotype
No records found.
ARTICLES LINKED TO SEARCH
Link
No records found.
Edit/Update
Save
Please be informed that this view is for administrative purposes only. FROM: Richard Ampofo Odame (MSc. Bioinformatics)
COST - Action 1204: Food and Agriculture