*Ryan Hoffman*

**Project Documentation**



# 

# **Executive Summary**

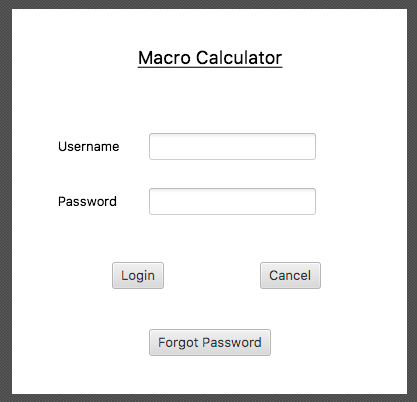
The Macro Calculator that I plan to make will incorporate other aspects of diets into one program. This project will not set and advise what a person’s diet or macros should be but will allow a user to enter their own numbers. This allows users to have full customizability to play with their levels of proteins, carbs and fats that they need to consume per day.

Once a user has set their diet they can then go ahead and start logging what they have eaten. There are no preloaded meals or food items in the database. There is going to be a template that a user can fill out and then have that information stored for both their current meal as well as future food logging. As the user adds foods to their daily food diary their macro tracker will keep track of what has been eaten and what is left to eat for the day. Having this graph auto adjust to the entries allows the user to know right away how much food is left to eat for the day which will allow him/her to make better decisions with their remaining meals.

There will also be a Fasting function for those that wish to do so. This function is just a running clock that is set to the duration of time the user wishes to fast. Think of a kitchen timer, the user can set the time they want to fast and once that time has expired an alert will happen to notify the user can stop fasting and start eating.

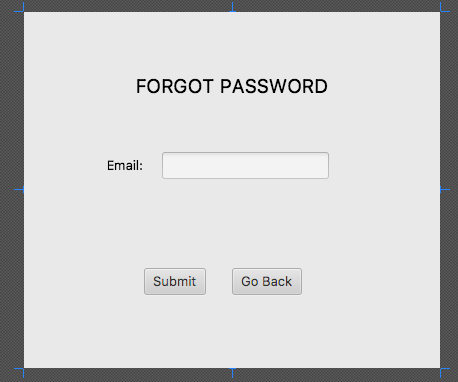
People keeping track of their diets by macro counting who also fast may find this program very useful. This macro calculator will be useful to those who are adjusting their macronutrient numbers and want to find their optimum numbers for performance and health. Having an application with the fasting functionality built in removes the need for one more application and can keep the users needs all in one place.

**Scenario | Program Usage Description**

****

From this screen the user can enter the login information to enter the account. The user has an added option to select Forgot Password which will take them to a password recovery screen.

# **Scenario | Program Usage Description**

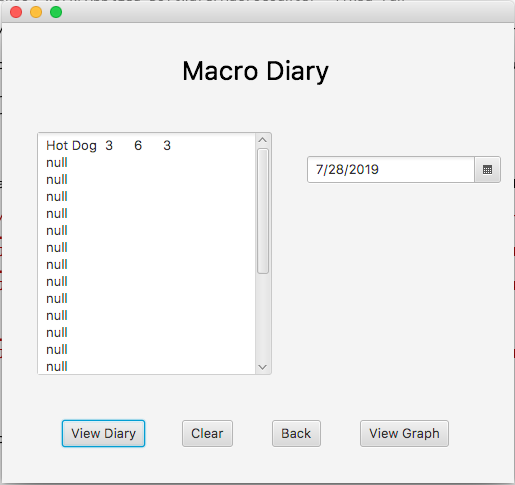
****

The password recovery scene will take the users email and check to see if it is present in the database. If the email is found in the database then the password associated with it will be sent to that email address. When the email is not found then an alert will be displayed to the user.

# **Scenario | Program Usage Description**

The Food Log scene is where users can enter their foods information. The food item along with the macronutrient information is then saved to the database with the selected date from the DatePicker field. From here the user can select to View their Diary which will allow them to view all their entries for a given day.

# **Scenario | Program Usage Description**



In the Macro Diary Scene the user can view the entries for a given day. The user can then go to View Graph to see the breakdown of the day selected and see how a pie chart with the macronutrient breakdown.

# 

# 

# 

# 

# **System Architecture**



## **Source Code Structure**

Source code structure introduction. The following is a summary of the source code directories and their contents:

|  |  |
| --- | --- |
| **Directory** | **Usage** |
| Source Packages | Holds all FXML and Controllers as well as the java files that hold reusable code. |
| Libraries | Holds libraries and jar files in order to perform.  The following jar files are :  Jfoenix-8.0.8.jar  MySQL JDBC Driver - mysql-connector-java-5.1.23-bin.jar  Mail.jar  Smtp.jar |

# 

# **Code Architecture**

With the MySQL connector included the program is able to communicate with the database which is hosted by phpMyAdmin.

## **Database or Data Store**

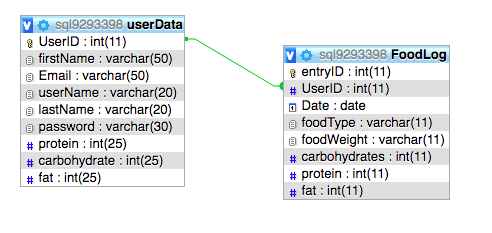
The database is stored in phpMyAdmin

Server : sql9.freemysqlhosting.net

Username: sql9293398

Password: 5UPXYpEPHW

The design is very basic and hold two tables; a userData table and a FoodLog table. These tables are linked with a primary key in userData named UserID and is found in the FoodLog table as a foreign key.



Programming Language | JAVA

This program is written using Java language and written using the NetBeans IDE. There are a number of jar files associated with this in order for it to run properly; they are:

Jfoenix-8.0.8.jar

MySQL JDBC Driver - mysql-connector-java-5.1.23-bin.jar

Mail.jar

Smtp.jar

Project Classes

Classes within the project are used to abstract re-usable pieces of code. Classes are also used to group related values, known as properties. The project utilizes these classes:

### **Connectivity Class | Connectivity.java**

This class holds some reusable functions used when accessing the database. There is a ConnectDB function that is called upon in the other controllers when and then once that connection is made the validation functions and updating the database functions are called from this class.

### **Input Validator Class | InputValidator.java**

In this class the user login is passed to these functions. The functions in InputValidator make sure that the correct information is being entered into user name fields and password fields. This class also holds the information when creating usernames and passwords and making sure they follow the correct format for these entries.

### **New User Class | NewUser.java**

In this class the get and set functions are located. This is where the user is being created and the user information will be passed to the database to be stored.

**Mail Class | Mail.java**

Mail class holds the information login information for the password recovery email. This is where the template is set when the user needs to recover their password.

Project Modules

Modules are used for procedural based code that does not require state data like class modules do. Complete the introduction to modules.

### **FXMLDocumentController | FXMLDocumentController.java**

Holds the scene logic to perform actions when the buttons are clicked.

### **ForgotPasswordController | ForgotPasswordController.java**

Used to collect the user’s email. validateEmail() makes sure that the email is valid and is present in the Database. When validated its passed to the Mail.java functions for password recovery.

### **ForgotPassword2Controller | ForgotPassword2Controller.java**

This holds the buttons logic to travel back to the login screen. The FXML associated with this displays a success screen that tells the user the email recovery was sent.

### **NewUser1 | NewUser1.java**

This accepts firstName, lastName, email, proteinInput, carbInput and fatInput. The values are then verified through the methods in InputValidator.java

### **NewUser2 | NewUser2.java**

This accepts inputID and Passwords. The values are then verified by calling the functions in the InputValidator.java

### **NewUserConfirmController | NewUserConfirmController.java**

This is successful scene that shows the user was created successfully. Only logic present is to travel back to the login screen to login with the newly created account.

### **MacroLogController | MacroLogController.java**

This scene accepts the variables foodItem, foodWeight, foodProtein, foodCarb, foodFat and dateEntry. The variables are then added to the DataBase along with the userID who logged it.

### **MacroDiaryController | MacroDiaryController.java**

User is able to view a log of entries from a selected date. The userID is passed along with the date to the database and the entries on that date are displayed to the text area.

### **MacroGraphController | MacroGraphController.java**

User can select a given date and a pieChart will populate with the values from the database for that date.

Summary

This program has the capability to save new user data as well as their log information for diet information. Users are also able to retrieve password information view the password recovery functions. The user can log diet information then access that information with the diary controller. The user should also be able to visually see the data for a particular day with the diary graph,however, the functionality is not working for that at this time.

There are a number of reusable functions for fetching user data, validating entries and storing the user input.

# **APPENDIX B (BUILD AND RELEASE PROCESS)**

Building of the app is done in Netbeans Ide and will be stored in the GitHub repository. The current building program will be found in the main branch of the repository. Once there is going to be updates made their will be additional branches being utilized. This allows the current iteration to continue working for its users. Once new updates are made on that branch and functionality is working properly that branch can then be merged to the main branch of the repository. This will then require users to update their local version to the new updated main branch of the repository.

# **APPENDIX D (DEVELOPER SETUP INSTRUCTIONS)**

The project is written in Java and the IDE being used is Netbeans. The developer will need to install the following libraries and jar files: Jfoenix-8.0.8.jar, MySQL JDBC Driver - mysql-connector-java-5.1.23-bin.jar, Mail.jar and Smtp.jar.