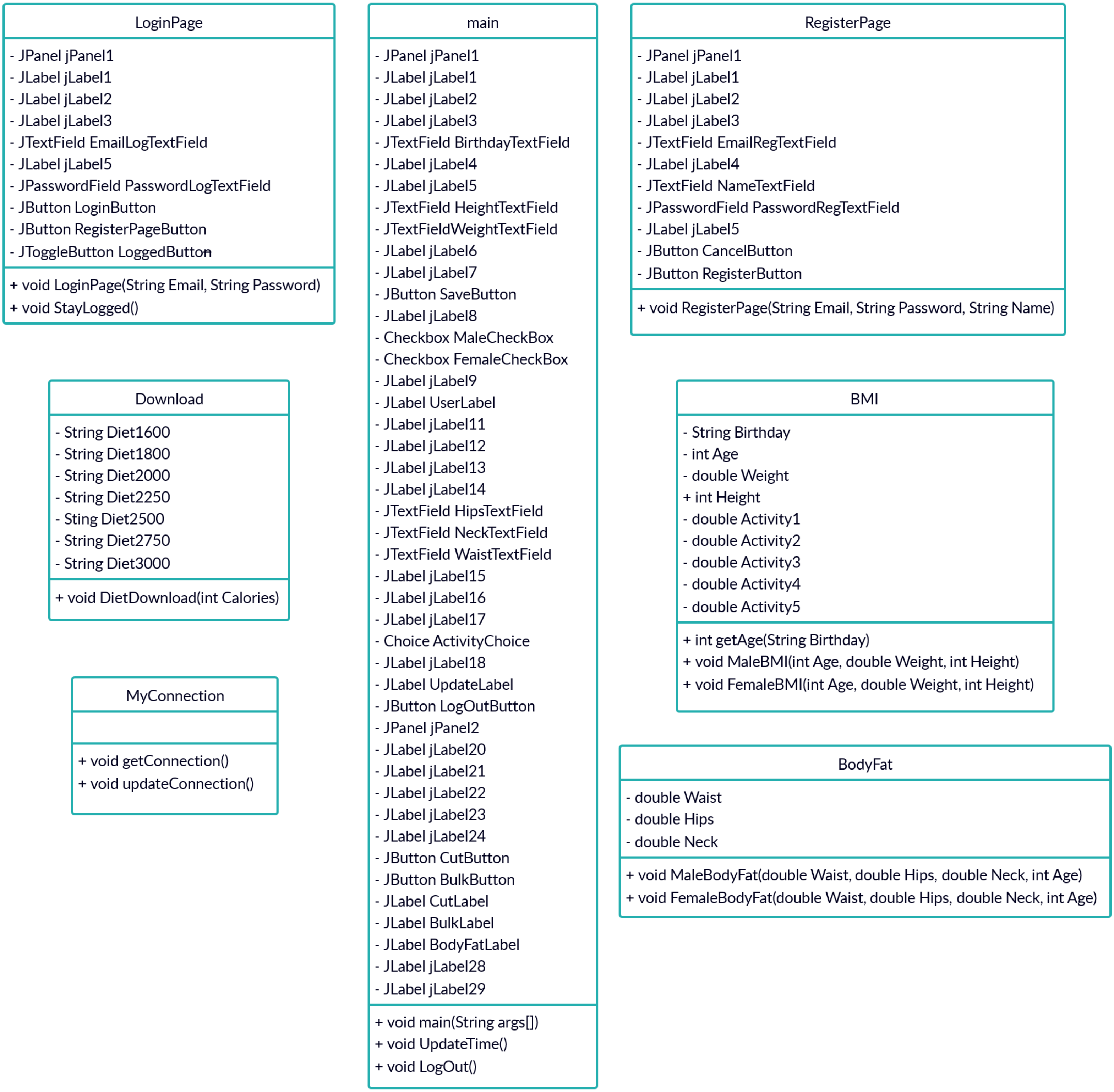
Criterion B: Design



```
LoginPage
- JPanel jPanel1
- JLabel jLabel1
- JLabel jLabel2
- JLabel jLabel3
- JTextField EmailLogTextField
- JLabel jLabel5
- JPasswordField PasswordLogTextField
- JButton LoginButton
- JButton RegisterPageButton
- JToggleButton LoggedButton
+ void LoginPage(String Email, String Password)
+ void StayLogged()
```

```
main
- JPanel jPanel1
- JLabel jLabel1
- JLabel jLabel2
- JLabel jLabel3
- JTextField BirthdayTextField
- JLabel jLabel4
- JLabel jLabel5
- JTextField HeightTextField
- JTextFieldWeightTextField
- JLabel jLabel6
- JLabel jLabel7
- JButton SaveButton
- JLabel jLabel8
- Checkbox MaleCheckBox
- Checkbox FemaleCheckBox
- JLabel jLabel9
- JLabel UserLabel
- JLabel jLabel11
- JLabel jLabel12
- JLabel jLabel13
- JLabel jLabel14
- JTextField HipsTextField
- JTextField NeckTextField
- JTextField WaistTextField
- JLabel jLabel15
- JLabel jLabel16
- JLabel jLabel17
- Choice ActivityChoice
- JLabel jLabel18
- JLabel UpdateLabel
- JButton LogOutButton
- JPanel jPanel2
- JLabel jLabel20
- JLabel jLabel21
- JLabel jLabel22
- JLabel jLabel23
- JLabel jLabel24
- JButton CutButton
- JButton BulkButton
- JLabel CutLabel
- JLabel BulkLabel
- JLabel BodyFatLabel
- JLabel jLabel28
- JLabel jLabel29
+ void main(String args[])
+ void UpdateTime()
+ void LogOut()
```

```
RegisterPage
- JPanel jPanel1
- JLabel jLabel1
- JLabel jLabel2
- JLabel jLabel3
- JTextField EmailRegTextField
- JLabel jLabel4
- JTextField NameTextField
- JPasswordField PasswordRegTextField
- JLabel jLabel5
- JButton CancelButton
- JButton RegisterButton
+ void RegisterPage(String Email, String Password, String Name)
```

```
Download
- String Diet1600
- String Diet1800
- String Diet2000
- String Diet2250
- Sting Diet2500
- String Diet2750
- String Diet3000
+ void DietDownload(int Calories)
```

```
BMI
- String Birthday
- int Age
- double Weight
+ int Height
- double Activity1
- double Activity2
- double Activity3
- double Activity4
- double Activity5
+ int getAge(String Birthday)
+ void MaleBMI(int Age, double Weight, int Height)
+ void FemaleBMI(int Age, double Weight, int Height)
```

```
MyConnection
+ void getConnection()
+ void updateConnection()
```

```
BodyFat
- double Waist
- double Hips
- double Neck
+ void MaleBodyFat(double Waist, double Hips, double Neck, int Age)
+ void FemaleBodyFat(double Waist, double Hips, double Neck, int Age)
```

Figure 1 - UML diagram with classes the program uses.

Functionality of each class:
main – Workspace where all the variables of the program are inputted and outputted (communicates with the MySQL database). It has the gui and connects the other classes to each other.

LoginPage – Login system that connects to MySQL database.

RegisterPage – Register system that connects to MySQL database with the option to cancel, in order to get back to the login page.

MyConnection – It contains all the methods needed to connect and update from the MySQL database.

BMI – Calculates the Body Mass Index using the formula provided by my client (attached in the Appendix) and connects to BodyFat, in order to get more accurate results.

BodyFat – Calculates the body fat percentage using the formula provided by US Navy. It gets some attributes from BMI.

Download – Reads the calculated burned calories generated by BMI and suggests diets in that range, updating the download button to the corresponding link from the website.

*More classes may be added in the development, but the above ones are the most important.

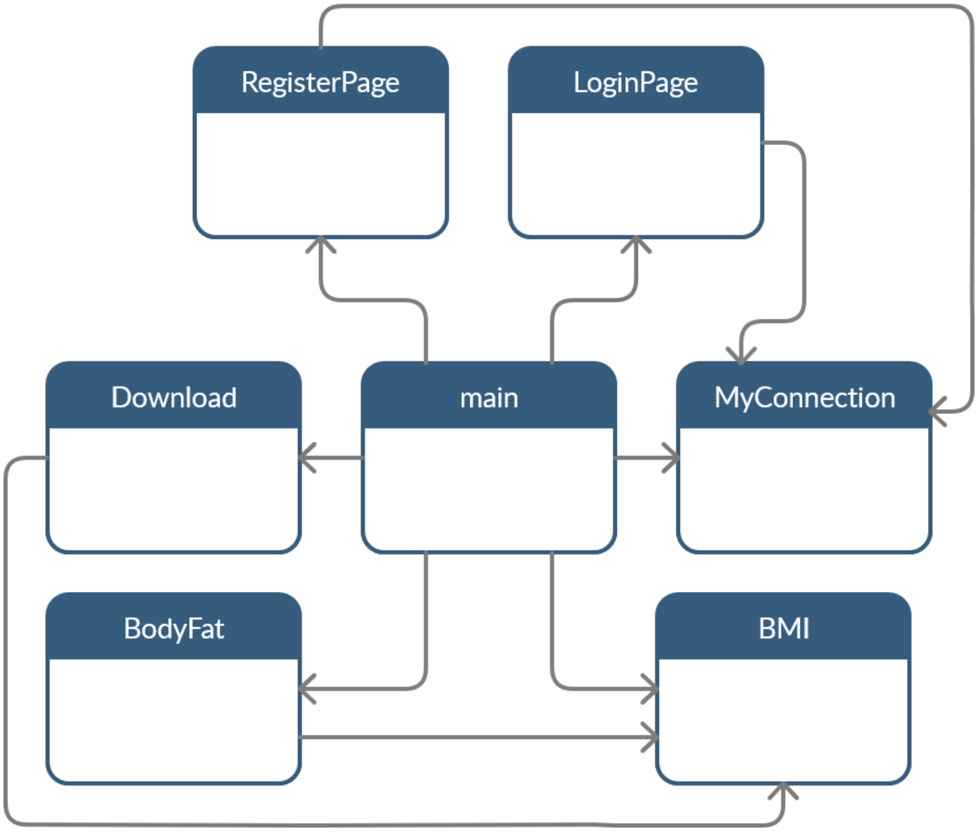
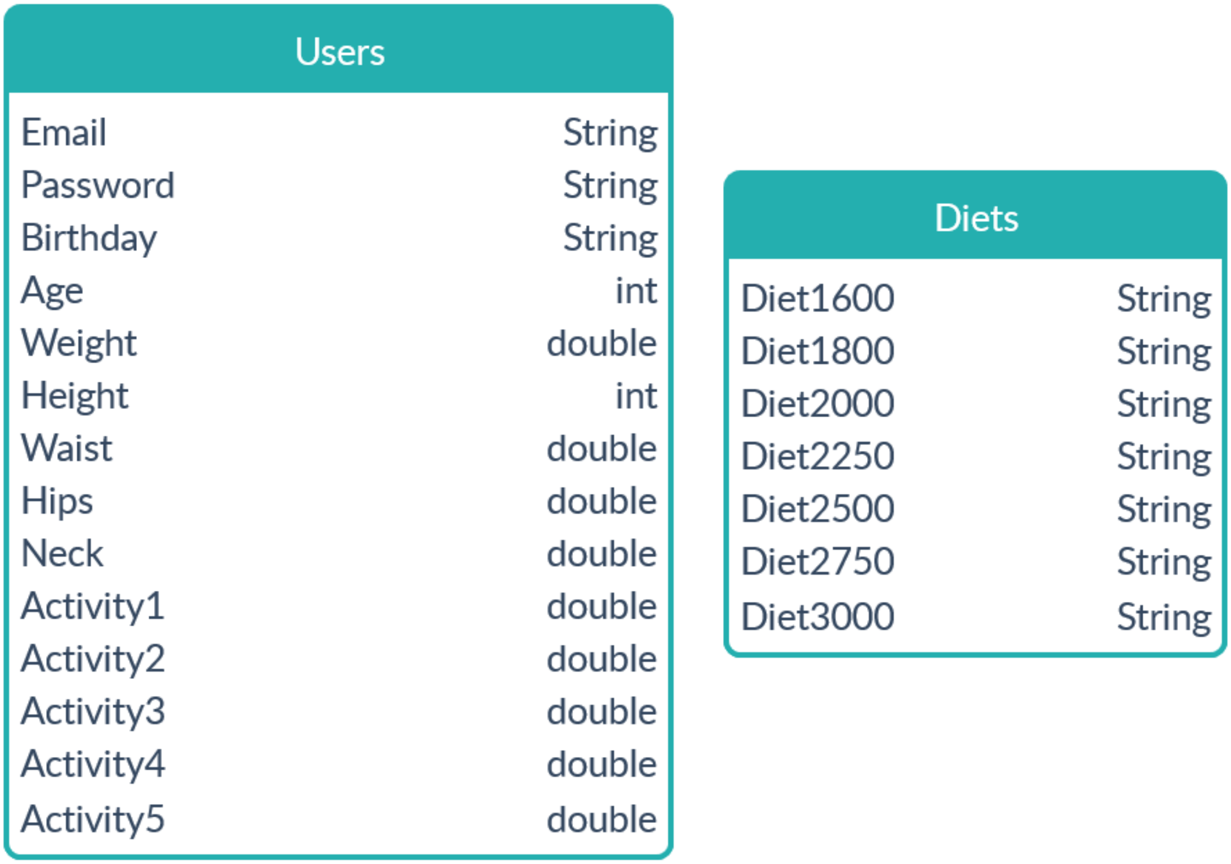


Figure 2 – Classes of the program and their links.



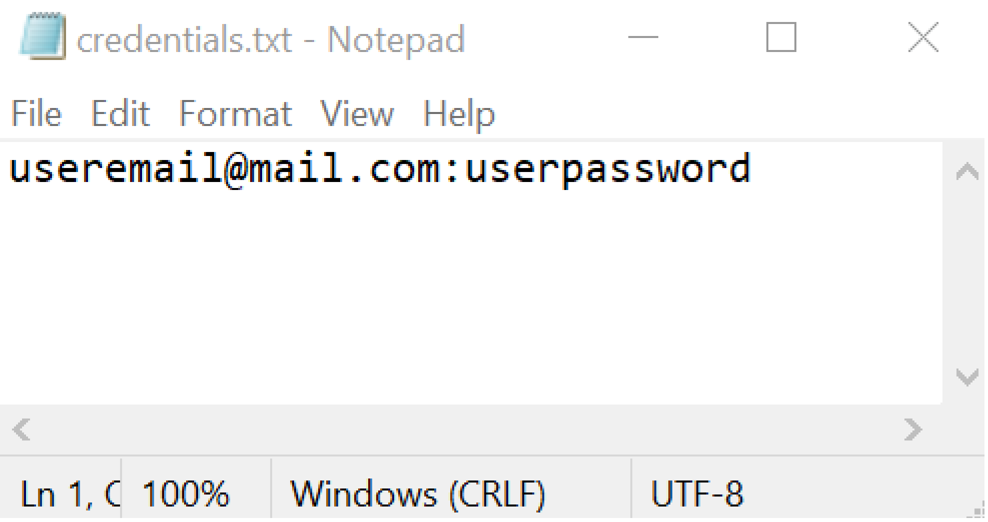
Users	
Email	String
Password	String
Birthday	String
Age	int
Weight	double
Height	int
Waist	double
Hips	double
Neck	double
Activity1	double
Activity2	double
Activity3	double
Activity4	double
Activity5	double

Diets	
Diet1600	String
Diet1800	String
Diet2000	String
Diet2250	String
Diet2500	String
Diet2750	String
Diet3000	String

Figure 3 - Databases and files that program will require.

My key field for the MySQL database is Email.

Program will create a text file with the login credentials if the option “Stay Logged” is chosen, so the program can log in automatically. You can see the intended syntax below.



credentials.txt - Notepad
File Edit Format View Help

```
useremail@mail.com:userpassword
```

Ln 1, C 100% Windows (CRLF) UTF-8

Figure 4 – Text file created by the program to store credentials encrypted.

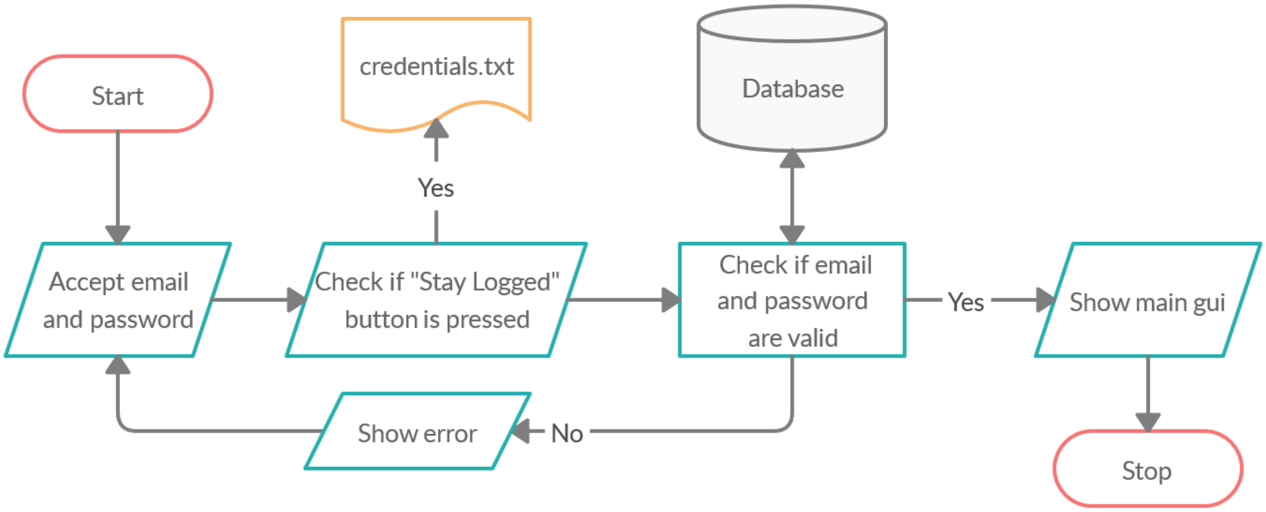

Start
credentials.txt
Database
Yes
Accept email and password
Check if "Stay Logged" button is pressed
Check if email and password are valid
Yes
Show main gui
Show error
No
Stop

Figure 5 – Login system flowchart diagram.

User types email and password in EmailLogTextField and PasswordLogTextField, these strings are then put in LoginPage() method. If user wants the program to auto login by using StayLogged() method, presses the LoggedButton.

Login function

```
LoginSystem()
    // function to login to MySQL database
    PreparedStatement ps
    ResultSet rs
    String email = get text from text field
    String pass = get text from text field
    String query = "SELECT * FROM `users` WHERE `email` =? AND `pass` =?"

    try
      pass = EncyptPasswordWithSHA-256(pass)
      ps = ConnectToMySQLdb.prepareStatement(query)

      ps.setString(1, email)
      ps.setString(2, pass)

      rs = ps.executeQuery()

      if rs.next()
        try to write file "credentials.txt" in "utf-8" in the local directory
          if LoggedButton is Selected = true //if stay logged box is pressed
```

```
            Write file email + ":" + pass +":1"

          else
            Write file email + ":" + pass + ":0"
             Open the main page
             Close the login page

      catch SQL Exception
          Log the exception
      else
        show Message Dialog "Incorrect email or password."
```

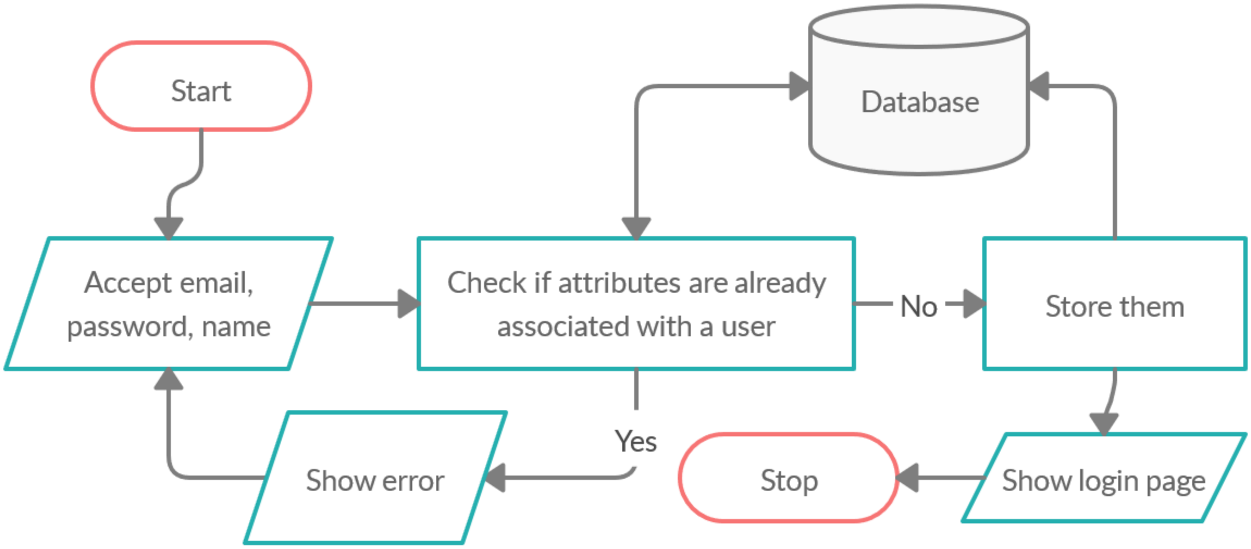

Start
Accept email, password, name
Check if attributes are already associated with a user
Database
No
Store them
Yes
Show error
Stop
Show login page

Figure 6 – Register system flowchart diagram.

User types email, password and name in EmailRegTextField, PasswordRegTextField, NameTextField, these strings are then put in RegisterPage.

Register function

```
RegisterSystem()
    // Sign in the user to the MySQL database
    String name = get text from text field
    String email = get text from text field
    String pass = get text from text field
    String re_pass = get text from text field
    String bdate = null
```

```
//error checking the string to have to appropriate data
if name = null
   show Message Dialog "Add a name."

else if email = null or email contains "@" = false
   show Message Dialog "Add a valid email."

else if pass.length() < 8 or pass = re_pass
   show Message Dialog "Retype the password again.”

else if checkEmail(email) = true
   show Message Dialog "This email already exist."

else if data = null or incorrect format
   show Message Dialog "Input correct birthday format."

   // check for the default MySQL data format
   bdate = select data using a GUI
PreparedStatement ps
String query = "INSERT INTO `users` (`name`, `email`, `pass`, `birthday`) VALUES (?,?,?,?)"

try
   pass = EncyptPasswordWithSHA-256(pass)
   ps = ConnectToMySQLdb.prepareStatement(query)

   ps.setString(1, name)
   ps.setString(2, email)
   ps.setString(3, pass)

   If changes were successfully made on database
      show Message Dialog "New user added.”

catch SQL Exception
    Log the exception
```

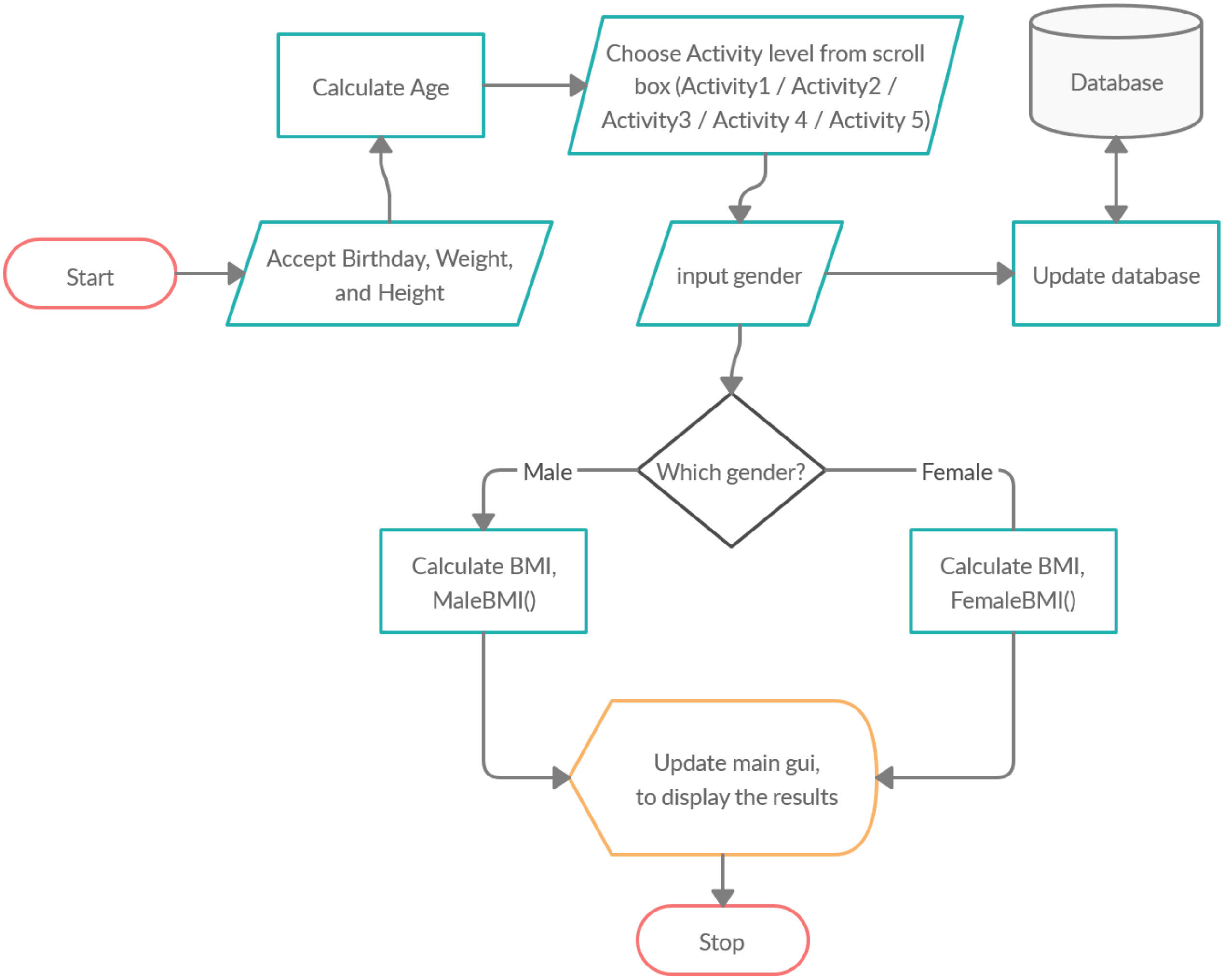

Calculate Age
Choose Activity level from scroll box (Activity1 / Activity2 / Activity3 / Activity 4 / Activity 5)
Database
Start
Accept Birthday, Weight, and Height
input gender
Update database
Male
Which gender?
Female
Calculate BMI, MaleBMI()
Calculate BMI, FemaleBMI()
Update main gui, to display the results
Stop

Figure 7 – BMI calculator flowchart.

User types Birthday date, weight and height. It calculates user's age then ask user to choose his activity level (1 – low to 5 – high). User ticks the box with the gender, this then updated to the MySQL database, then calculates the Body Mass Index, displaying the results (cut and bulk calories).

BMI calculator function

```
public int MaleBMI (int Age, int Weight, int Height, double Activity)
   double calories = (66 + (13.7*Weight) + (5*Height) - (6.8*Age))*Activity
   return int Calories

public int FemaleBMI (int Age, int Weight, int Height, double Activity)
   double calories = (655 + (9.6*Weight) + (1.8*Height) - (4.7*Age))*Activity
   return int Calories
   // Save details to MySQL database
   // get details from TextFields and save to local variables
```

```
open the file
try
   scan read it
   delimit variables by ":"
   while read.hasNext() //into below variable
      email = read.next()
      pass = read.next()
      String Logged = read.next()

catch SQL Exception
    Log the exception

// switch case to associate strings with numbers
        // for the db


if null != activity switch activity
   case ...
    activity = 1.2/1.37/1.55/1.725/1.9

if Male = gender
   gender = 0
else if Female = gender
   gender = 1

PreparedStatement ps
String query = "UPDATE users SET weight='"+ weight +"' , height='"
        + height +"' , waist='"+ waist +"' , hips='"+ hips +"' , neck='"
        + neck +"' , gender='"+ gender +"' , activity='" + activity
        + "' WHERE email='" + email +"' AND pass='" + pass + "'"
try
   ps = ConnectToMySQLdb.prepareStatement(query)
   ps.executeUpdate()
   show Message Dialog "Details updated."

catch SQL Exception
    Log the exception

   show Message Dialog "Please check your input."
   // error checking if the inputs are not accordingly
   // for example: inputing string into int

   Update() //refresh GUI
Update() // Live update the details from MySQL database
    open the file
    try
       scan read it
      delimit variables by ":"
      while read.hasNext()
        email = read.next()
        pass = read.next()
        String Logged = read.next()

   catch SQL Exception
      Log the exception

String query = "SELECT height,waist,hips,neck,weight,activity,gender,name FROM "
        + "users WHERE email ='" + email + "' AND pass ='" + pass + "'"


try
  ps = ConnectToMySQLdb.prepareStatement(query)
  if ps.execute()
    rs = ps.getResultSet()
  else
    output "Failed."

  while rs.next() //loops while db sends data
    Save the data fetch into variables (name, height, waist, hips, neck, weight)

    Set text on GUI (name, height, waist, hips, neck, weight) //below updates the GUI

    if activity=1.2 //associate numbers with choices
      Set combo list choice on GUI
    if gender=0
      Set combo list choice on GUI
    else if (gender=1
      Set combo list choice on GUI

     catch SQL Exception
      Log the exception


int age = getAge(email)


    if gender=0
    calories = cals.MaleBMI (age, weight, height, activity)
    bodyfat = fats.MaleBodyFat (waist, hips, neck, age, height)
  else if gender=1
    calories = cals.FemaleBMI (age, weight, height, activity)
    bodyfat = fats.FemaleBodyFat (waist, hips, neck, age, height)

  Set text on GUI (bodyfat)
  CutCalories=CutCalories-300
  int BulkCalories=CutCalories+600
  Set text on GUI (CutCalories)
  Set text on GUI (BulkCalories)

getAge (String email) // Calculate the user's age from his birthday
  PreparedStatement ps
  ResultSet rs = null
  int age = 0

  try
     String query = "SELECT DATE_FORMAT (FROM_DAYS (DATEDIFF (CURDATE
     (),birthday)), + "'%Y')+0 AS age FROM users WHERE email='"+ email +"'"
           // change date to days and then make the difference
           // ending with converting it to years

  ps = ConnectToMySQLdb.prepareStatement(query)
    if ps.execute()
      rs=ps.getResultSet()
    else
      output "Failed."

    while rs.next()
    age=rs.getInt(1)

    catch SQL Exception
     Log the exception

   return age
```

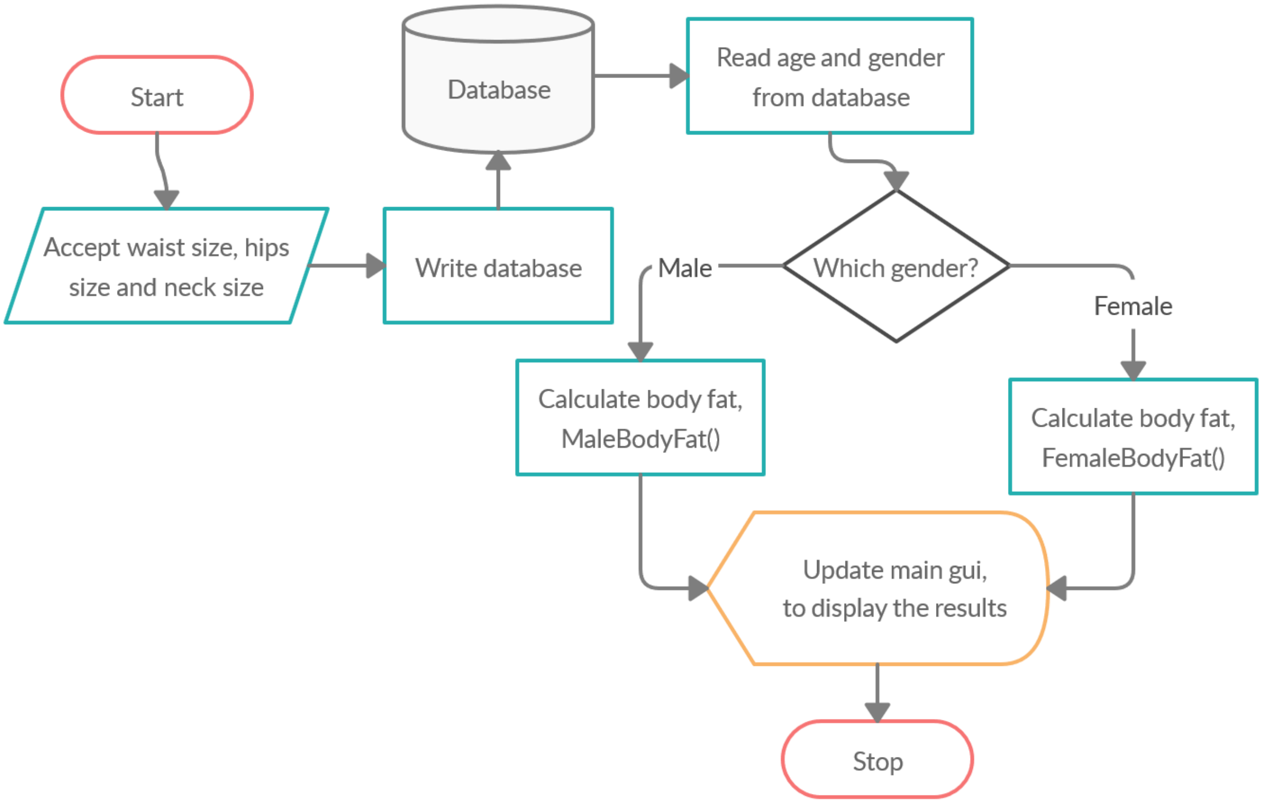

Start
Database
Read age and gender from database
Accept waist size, hips size and neck size
Write database
Male
Which gender?
Female
Calculate body fat, MaleBodyFat()
Calculate body fat, FemaleBodyFat()
Update main gui, to display the results
Stop

Figure 8 – Body fat percentage calculator flowchart.

User types waist size, hips size and neck size, It writes user's information to the database and then reads user's age and gender. It calculates the body fat and displays the results.

Body fat percentage calculator function

```
int MaleBodyFat (int Waist, int Hips, int Neck, int Age, int Height)
    double bodyfat = (495/(1.0324-0.19077*( log (Waist-Neck)) + 0.15456 * (log
    (Height))))-450 //calculate body fat percentage for male
    return BodyFat

int FemaleBodyFat (int Waist, int Hips, int Neck, int Age, int Height)
    double bodyfat = (495/(1.29579-0.35004*(log(Waist+Hips-Neck)) + 0.22100 *
    (log (Height))))-450 //calculate body fat percentage for female
    int BodyFat=(int)bodyfat; //convert double to int
    return BodyFat
```

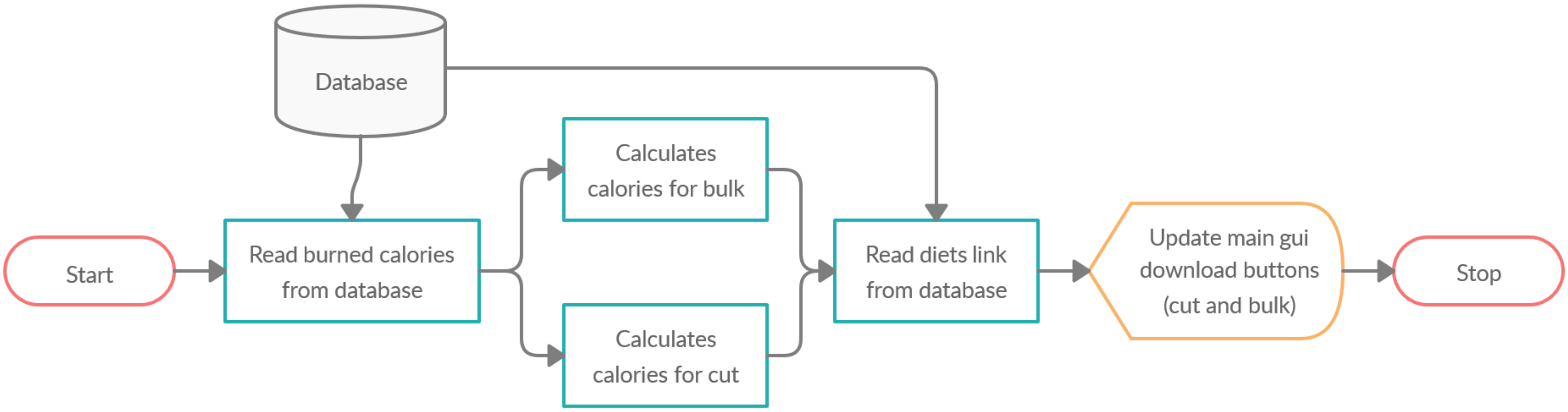

Database
Start
Read burned calories from database
Calculates calories for bulk
Calculates calories for cut
Read diets link from database
Update main gui download buttons (cut and bulk)
Stop

Figure 9 – Downloading diets function flowchart.

Downloading diets function

```
DietDownloadCut (int x) //function to open the url
    try //according to the inputed calories, Cut version.
        If x < 1800 //range of calories
          x = 1
        else if x > 1800 | x < 2000
          x = 2
        else if x > 2000 | x < 2250
          x = 3
        else if x > 2250 | x < 2500
          x = 4
        else if x > 2500 | x < 2750
          x = 5
        else if x > 2750
          x = 6

      PreparedStatement ps
      ResultSet rs = null

      String query = "SELECT link FROM diets WHERE id = '" + s + "'"
      ps = ConnectToMySQLdb.prepareStatement(query)

     if ps.execute()
        rs=ps.getResultSet() //send the query to db

      while rs.next()
        CutLink = rs.getString(1) //receive from db once
```

```
       OpenLink(CutLink) //pop the link in browser
     catch SQL Exception
        Log the exception



DietDownloadBulk (int x) //function to open the url
     try //according to the inputed calories, Bulk version.
         if x < 1800 //range of calories
           x = 1
         else if x > 1800 | x < 2000
           x = 2
        else if x > 2000 | x < 2250
           x = 3
         else if x > 2250 | x < 2500
           x = 4
         else if x > 2500 | x < 2750
           x = 5
         else if x > 2750
           x = 6

       PreparedStatement ps;
       ResultSet rs = null;

       x = x+1;
       String query = "SELECT link FROM diets WHERE id = '" + s + "'"

       ps = ConnectToMySQLdb.prepareStatement(query)

       if(ps.execute()
         rs=ps.getResultSet() //send the query to db

       while rs.next()
         BulkLink = rs.getString(1) //receive from db once

       OpenLink(BulkLink) //pop the link in browser

    catch SQL Exception
        Log the exception
```

Check Logged function

```
checkLogged()
    File f = new File("credentials.txt") //read file where login credentials are stored
try
    Scanner read = new Scanner(f)
    read.useDelimiter(":") //separates the email, pass and Logged status into different variables
    loop while read.hasNext())
    email = read.next()
    pass = read.next()
    Logged = read.next()

catch FileNotFoundException
    Log error

If Logged = 1
    try
        String query = "SELECT * FROM `users` WHERE `email` =? AND `pass` =?"
        ps = Connect to MySQL database and send query
        ps.setString(1, email)
        ps.setString(2, pass)
        rs = ps.executeQuery()

    if rs.next()
        Open the main page
        Close the login page

    catch SQLException
        Log error
```

Design of Graphical User Interface:

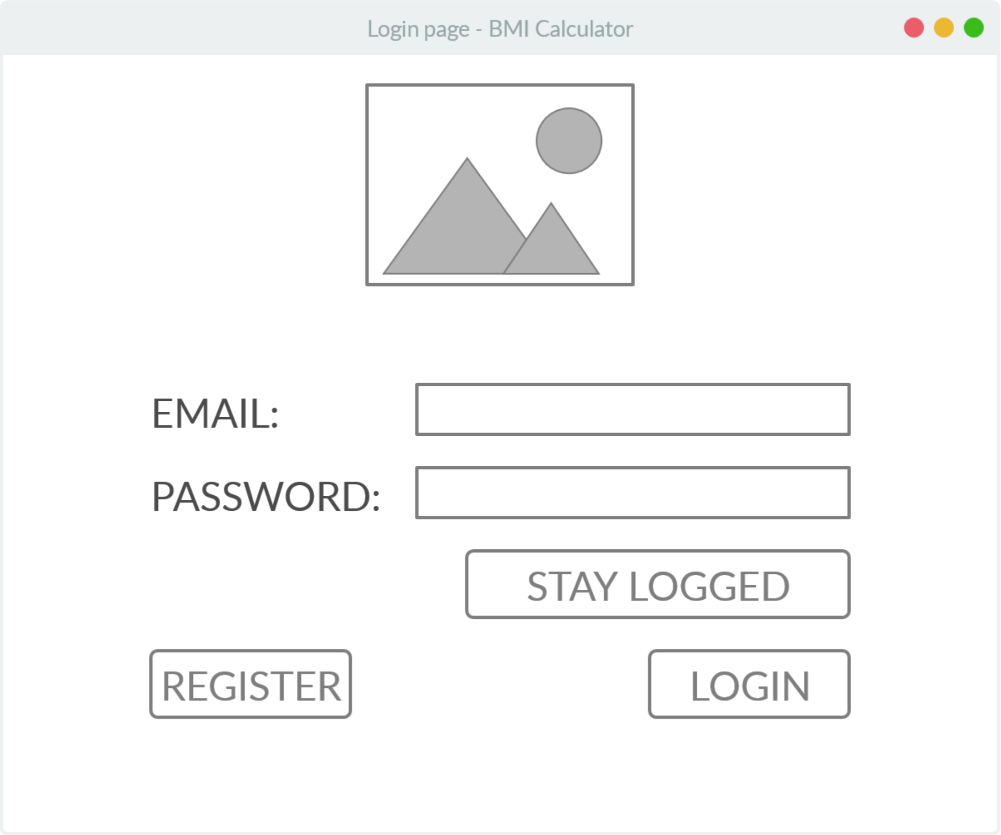


Figure 10 – Login page

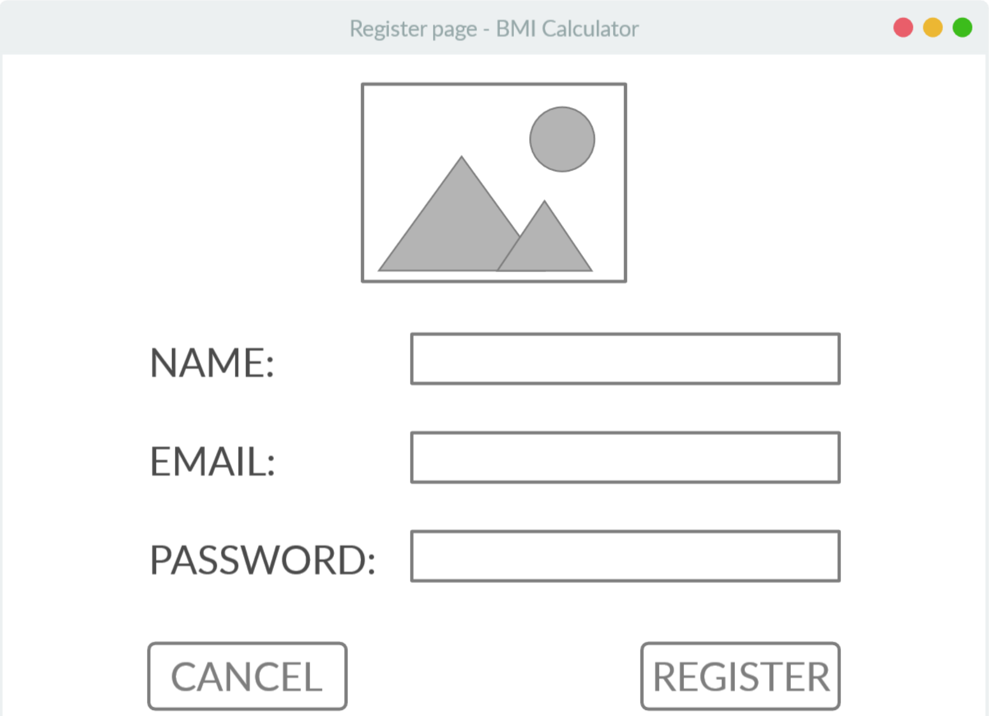
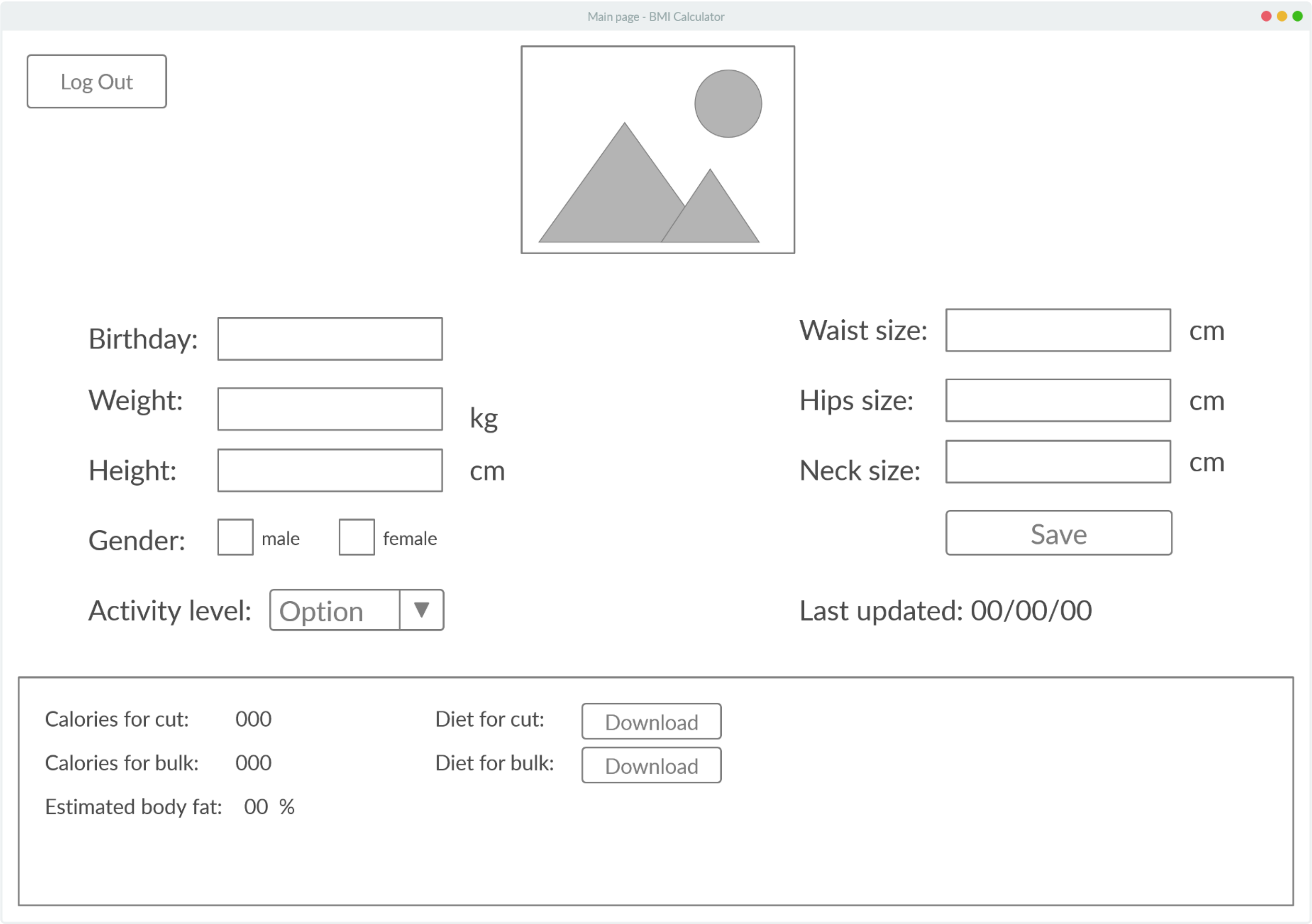


Figure 11 – Register Page



Main page - BMI Calculator

Birthday:

Weight: kg

Height: cm

Gender: □ male □ female

Activity level: Option ▼

Waist size: cm

Hips size: cm

Neck size: cm

Last updated: 00/00/00

Calories for cut: 000

Calories for bulk: 000

Estimated body fat: 00 %

Diet for cut: Download

Diet for bulk: Download

Figure 12 – Main page

I have asked Mrs. I. Sochirca if the GUI is friendly enough and she had the suggestion of having all the functions in one window (the main one) instead of different pages. I thought the idea was great, so I implemented it and sent her the final sketch (**Figure 8**). She approved it. You can see the emails between us in the appendix.

Test plan:

ACTION TO BE TESTED	TEST METHOD
User is able to register and login, with error checking for both.	Input the details to the register and login page.
Remain logged function works.	Reopen the program after log in choosing the "stay logged".
Program is able to add and edit user information (name, email, birthday, height, weight, activity level, waist size, hips size and neck size) to the MySQL databases.	Input these details into the program, checking with the database if all the information is written correct. Error checking if the email and password are valid (or meet the minimum requirements).
Using MySQL as the database, program might be vulnerable to SQL injections.	Program will be tested with several SQL injection payloads, which will be then patched.
Check that the BMI and Body fat percentage calculator work as expected.	Compare them with manually using the formulas.
The algorithm updates the download buttons (for cut and bulk diets) with the correct link.	Input different BMI details in order to get all the calories range (1600 – 3000) and check that the download buttons update to the correct links.
All the graphical interfaces are showing the correct output (without any visual glitches).	Run the program multiple times to test individual features linked to graphical elements with various inputs and check if they are performing the desired functions.
The details remain in the text fields, to give the user the possibility to track his measurements.	Run the program multiple times with internet connection drops to see if the details still update.
MySQL database which is hosted on a web server can be accessed by Mrs. I. Sochirca.	Test the website locally and then send the link to Mrs. I. Sochirca for further testing.