Appendix 3: Source Code

Use the search tool to navigate the .java file the name is in relation with **Criteria C** Figure heading, inside the bracket it will tell which .java file the code is from, so it can be search here for the full code and to see the comment that are unable to be load in Criteria C.

Main

```
package sample;
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class Main extends Application {

  @Override
  public void start(Stage primaryStage) throws Exception{
    Parent root = FXMLLoader.load(getClass().getResource("login.fxml")); //Exentially load up the login page
    primaryStage.initStyle(StageStyle.UNDECORATED);
    primaryStage.setScene(new Scene(root, 520, 400)); // This respond to the size for the program page
    primaryStage.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```

Database Connection

```
package sample;

import java.sql.*;


public class DatabaseConnection {
   public Connection databaseLink;
   public Connection getConnection() { //Use to connect to the database when
called, only work if host locally
       String databaseUser = "root";
       String databasePassword = "skyhunter1921";
       String url = "jdbc:mysql://localhost/demo_db";
//    public Connection getConnection() { //Use to connect to the database when
called, using hosting service to held database
//        String databaseUser = "epiz_31299866";
//        String databasePassword = "T6n2WtKVmTdsjIW";
//        String url = "sql209.epizy.com/epiz_31299866_demo_db";

       try {
           Class.forName("com.mysql.cj.jdbc.Driver");
           databaseLink = DriverManager.getConnection(url, databaseUser,
databasePassword);
       } catch (Exception e) {
           e.printStackTrace();
           e.getCause();
       }
       return databaseLink;
   }

   public static boolean CheckUsernameExists(String username) //Command to check if
the username that is register already exist
   {
       DatabaseConnection connectNow = new DatabaseConnection();
       Connection connectDB = connectNow.getConnection();
       boolean usernameExists = false;
       try
       {
           //loop through the database sort by username descending
           //individually check  with the newly created
           PreparedStatement st = connectDB.prepareStatement("select * from
user_account order by username desc");
           ResultSet r1=st.executeQuery();
           String usernameCounter;
           if(r1.next())
           {
               usernameCounter =  r1.getString("username");
               if(usernameCounter.equalsIgnoreCase(username))
               {
                   System.out.println("It already exists"); // warning
                   usernameExists = true;
               }
           }
       }
```

```
        catch (SQLException e)
        {
            System.out.println("SQL Exception: "+ e.toString());
        }
        return usernameExists;
    }
}
```

InsideApp

```
package sample;

import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.stage.Stage;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class insideApp{
   PreparedStatement preparedStatement = null;
   ResultSet resultSet = null;
   @FXML
   private Button homeButton;
   @FXML
   private Button close;

   //Close the tab with button
   public void closeOnAction() {
      Stage stage = (Stage) close.getScene().getWindow();
      stage.close();
   }

   //Go to the home page from other page
   public void homeOnAction() {
      new loadTab("lobby.fxml",1100,650);
      Stage stage = (Stage) homeButton.getScene().getWindow();
      stage.close();
   }

   //Method to check if it numeric
   public static boolean isNotNumber(String str){
      try{
            Double.parseDouble(str);
            return false;
        }catch(NumberFormatException e){
            return true;
        }
    }

    //Method to get the account_id (key) from log
    public int getAccount(){
        int account_id = 0; //create variable so can be change later
        DatabaseConnection connectNow = new DatabaseConnection();
        Connection connectDB = connectNow.getConnection();
        try {
            //load statement into database by looking at log database descending order of log_id
            preparedStatement = connectDB.prepareStatement("SELECT * FROM log ORDER BY log_id DESC
LIMIT 1");
            resultSet = preparedStatement.executeQuery();
            //loop while there more log to look at
            while (resultSet.next()) {
                account_id = resultSet.getInt("account_id");
            }
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
        System.out.println(account_id);
        return account_id;
    }
}
```

loadTab

```
package sample;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

// Method to open new tab with minimal line of code and input
public class loadTab {
    public loadTab(String fxmlFile, int width, int height){
        try{
            //Getting fxml from the code
            Parent root = FXMLLoader.load(getClass().getResource(fxmlFile));
            Stage Stage = new Stage();
            Stage.initStyle(StageStyle.UNDECORATED);
            //Setting it size following the set width and height
            Stage.setScene(new Scene(root, width, height));
            Stage.show();
        }catch(Exception e){
            e.printStackTrace();
            e.getCause();
        }
    }
}
```

LobbyController

```
package sample;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Button;
import javafx.scene.control.DatePicker;
import javafx.scene.control.Label;
import javafx.scene.control.ProgressBar;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.stage.Stage;

import java.io.File;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.util.Date;
import java.util.ResourceBundle;

public class LobbyController extends insideApp implements Initializable {

    @FXML
    private ImageView topLeftIconView;
    @FXML
    private ImageView botRightIconView;
    @FXML
    private ImageView botLeftIconView;

    @FXML
    private Button nutrientButton;
    @FXML
    private Button trackerButton;
    @FXML
    private Button plannerButton;

    @FXML
    private ProgressBar myProgressBar;
    @FXML
    private Label progressLabel;
    @FXML
    private DatePicker myDatePicker;
    @FXML
    private Label dayLabel;
    double progress;

    @Override
    public void initialize(URL url, ResourceBundle resourceBundle){
        //Pregnant Logo
        File topLeftFile = new File("Image/PREG.png");
        Image topLeftImage = new Image(topLeftFile.toURI().toString());
        topLeftIconView.setImage(topLeftImage);
        //Pie Chart Logo
        File botRightFile = new File("Image/PIE_CHART.png");
        Image botRightImage = new Image(botRightFile.toURI().toString());
        botRightIconView.setImage(botRightImage);
        //Calendar Logo
        File botLeftFile = new File("Image/CALENDAR.png");
        Image botLeftImage = new Image(botLeftFile.toURI().toString());
        botLeftIconView.setImage(botLeftImage);
        myProgressBar.setStyle("-fx-accent: red;"); //Colour of progress Bar
//        dailyAlert();
    }

    @FXML //Activate on Date Picker
    public void getDateNow() {
        //Load today date in "yyyy-MM-DD" format and parse it
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        String dateNow = sdf.format(new Date());
        //Get the value from the Date Picker and parse it
        LocalDate myDate = myDatePicker.getValue();
        String assign = myDate.toString();
        //Set the text to the different between two date
        dayLabel.setText(pregCount(assign, dateNow));
        //Calculate the difference and divide by 9 month to see how long until labor
        progress = Float.parseFloat(pregCount(assign, dateNow))/280;
        if (progress<0.9){
            //Set the progress bar percentage of progress
            myProgressBar.setProgress(progress);
            progressLabel.setText(((int)Math.round(progress * 100)) + "%");
        }
        }
    //Method to calculate difference between two given date
    private String pregCount(String past, String present){
        //Parsing it to float for calculation after picking substring from the date String
        float pastYear = Float.parseFloat(past.substring(0,4));
        float pastMonth = Float.parseFloat(past.substring(5,7));
        float pastDay = Float.parseFloat(past.substring(8,10));
        float presentYear = Float.parseFloat(present.substring(0,4));
        float presentMonth = Float.parseFloat(present.substring(5,7));
        float presentDay = Float.parseFloat(present.substring(8,10));
        float diffYear = presentYear-pastYear;
        float diffMonth = (presentMonth-pastMonth)+(diffYear*12);
        float diffDay = (presentDay-pastDay)+(diffMonth*30);
        if (diffDay>0){
            return String.valueOf((int)diffDay);
        } else {
            return "Error";
        }
    }

    public void calendarOnAction(){
        new loadTab("tracker.fxml", 1100, 650);
        Stage stage = (Stage) trackerButton.getScene().getWindow();
        stage.close();
    }
    public void nutrientOnAction(){
        new loadTab("nutrient.fxml",1100,650);
        Stage stage = (Stage) nutrientButton.getScene().getWindow();
        stage.close();
    }
    public void plannerOnAction(){
        new loadTab("planner.fxml",1100,650);
        Stage stage = (Stage) plannerButton.getScene().getWindow();
        stage.close();
    }

}
```

LoginController

```
package sample;

import javafx.fxml.FXML;
```

```java
import javafx.fxml.Initializable;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.stage.Stage;

import java.io.File;
import java.net.URL;
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.ResourceBundle;

public class LoginController implements Initializable {
  @FXML
  public Button loginButton;
  @FXML
  private Button signupButton;
  @FXML
  private Button cancelButton;

  @FXML
  private ImageView lockImageView;
  @FXML
  private TextField usernameTextField;
  @FXML
  private PasswordField enterPasswordField;

  PreparedStatement preparedStatement = null;
  ResultSet resultSet = null;
  Integer giveUserID = null;

  @Override
  public void initialize(URL url, ResourceBundle resourceBundle) {
  }

  @FXML
  public void loginButtonOnAction() {
     //Check if the field is blank
     if (!usernameTextField.getText().isBlank() && !enterPasswordField.getText().isBlank()) {
        validateLogin();

     } else {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setHeaderText(null);
        alert.setContentText("Please fill in the data!");
        alert.showAndWait();
        //if field is blank it alert the user with appropriate message
      }
    }
  @FXML
  public void cancelButtonOnAction() {
    Stage stage = (Stage) cancelButton.getScene().getWindow();
    stage.close();
  }

  public void signupButtonOnAction() {
    new loadTab("register.fxml", 520, 445);
    Stage stage = (Stage) signupButton.getScene().getWindow();
    stage.close();
  }

  public void validateLogin() {
    //Connect to Database
    DatabaseConnection connectNow = new DatabaseConnection();
    Connection connectDB = connectNow.getConnection();

    //Getting username and password from the field
    String username = usernameTextField.getText();
    String password = enterPasswordField.getText();
    //Select and find number of row from user_account that have the entered username and password
    String verifyLogin = "SELECT count(*) FROM user_account WHERE username = '" +
            username + "' AND password ='" + password + "'";
    try {
      //Load up the SQL Command
      Statement statement = connectDB.createStatement();
      ResultSet queryResult = statement.executeQuery(verifyLogin);
      //Loop through all the select result
      while (queryResult.next()) {
        //Check if the total number of row that is found is equal to one
        //Validating that username and password exist within database
        if (queryResult.getInt(1) == 1) {
          try {
            preparedStatement = connectDB.prepareStatement("SELECT * FROM user_account WHERE
username = '" +
                username + "'");
            resultSet = preparedStatement.executeQuery();
            while (resultSet.next()) {
              giveUserID = resultSet.getInt("account_id");
            }
          } catch (SQLException throwables) {
```

```java
                        throwables.printStackTrace();
                    }
                    String storeData = "INSERT INTO log(username, account_id) VALUES('"+username +"','"+
giveUserID +"')";
                    try {
                        Statement statementOne = connectDB.createStatement();
                        statementOne.executeUpdate(storeData);
                    } catch (Exception e) {
                        e.printStackTrace();
                        e.getCause();
                    }
                    //Today Date
                    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
                    String dateNow = sdf.format(new Date());
                    boolean FLAG = false; //Declaring FLAG

                    try {
                        //Select data from tracker table descending as we want the earilest date
                        preparedStatement = connectDB.prepareStatement("SELECT * FROM tracker WHERE user_id
="+giveUserID+" ORDER BY dateCal DESC;");
                        resultSet = preparedStatement.executeQuery();
                        while (resultSet.next()&& !FLAG) {
                        if((dateNow.equals(String.valueOf(resultSet.getDate("dateCal"))))){
                            FLAG = true;
                        }
                        }
                        if (!FLAG){
                            Alert alertEmpty = new Alert(Alert.AlertType.ERROR);
                            alertEmpty.setHeaderText(null);
                            alertEmpty.setContentText("Fill in tracker data for today!");
                            alertEmpty.showAndWait();
                        }
                    } catch (SQLException throwables) {
                        throwables.printStackTrace();
                    }

                    //Go to main page
                    new loadTab("lobby.fxml", 1100, 650);
                    Stage stage = (Stage) loginButton.getScene().getWindow();
                    stage.close();
                } else { //Login Failed with alert Message
                    Alert alert = new Alert(Alert.AlertType.ERROR);
                    alert.setHeaderText(null);
                    alert.setContentText("Invalid Login please try again!");
                    alert.showAndWait();
                }
            }
```

```
        } catch (Exception e) {
            e.printStackTrace();
            e.getCause();
        }
    }

}
```

RegisterController

```
package sample;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.stage.Stage;

import java.io.File;
import java.net.URL;
import java.sql.Connection;
import java.sql.Statement;
import java.util.ResourceBundle;


public class RegisterController extends DatabaseConnection implements Initializable {

    //This is to indicate the @FXML that I'm using
    @FXML
    private ImageView shieldImageView;
    @FXML
    private Button closeButton;
    @FXML
    private PasswordField setPasswordField;
    @FXML
    private PasswordField confirmPasswordField;
    @FXML
    private TextField firstnameTextField;
    @FXML
```

```
private TextField lastnameTextField;
@FXML
private TextField usernameTextField;

@Override
public void initialize(URL url, ResourceBundle resourceBundle){
    //Loading Image
    File shieldFile = new File("OMO/logo.png");
    Image shieldImage = new Image(shieldFile.toURI().toString());
    shieldImageView.setImage(shieldImage);
}
@FXML
public void close(ActionEvent event){
    new loadTab("login.fxml",520,440);
    Stage stage = (Stage) closeButton.getScene().getWindow();
    stage.close();
}

@FXML
public void registerButtonOnAction(ActionEvent event) {
    //Check if any field is empty
    if (firstnameTextField.getText().isBlank() && lastnameTextField.getText().isBlank() &&
            confirmPasswordField.getText().isBlank() && setPasswordField.getText().isBlank() &&
            usernameTextField.getText().isBlank()) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setHeaderText(null);
        alert.setContentText("Please fill in the data!");
        alert.showAndWait();
    //Check if password is equal to confirm password
    }else if(!(setPasswordField.getText().equals(confirmPasswordField.getText()))) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setHeaderText(null);
        alert.setContentText("Password not match!");
        alert.showAndWait();
    //Check if username is already used, recalling method from DatabaseConnection
    }else if((CheckUsernameExists(usernameTextField.getText()))) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setHeaderText(null);
        alert.setContentText("Username already exist!");
        alert.showAndWait();
    }else{
        registerUser();
    }
}
//Method to put registry data that entered from field into database
public void registerUser(){
    //Connect Datbase
```

```
        DatabaseConnection connectNow = new DatabaseConnection();
        Connection connectDB = connectNow.getConnection();
        //Getting Data from TextField
        String firstname = firstnameTextField.getText();
        String lastname = lastnameTextField.getText();
        String username = usernameTextField.getText();
        String password = setPasswordField.getText();
        //SQL commands to enter this data into new row on user_account table in database
        String insertFields = "INSERT INTO user_account(lastname, firstname, username, password) VALUES ('";
        String insertValues = firstname +"','"+ lastname +"','"+ username +"','"+ password +"')";
        String insertToRegister = insertFields + insertValues;
        try{
            //Execute Command
            Statement statement = connectDB.createStatement();
            statement.executeUpdate(insertToRegister);
            //Give Alert to show completion
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setHeaderText(null);
            alert.setContentText("User enter data successfully");
            alert.showAndWait();
        }catch (Exception e){
            e.printStackTrace();
            e.getCause();
        }
    }
}
```

PlannerController

```
package sample;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.geometry.HPos;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.stage.Stage;

import java.net.URL;
import java.sql.Connection;
```

```java
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.ResourceBundle;
import java.util.logging.Level;
import java.util.logging.Logger;

public class plannerController extends insideApp implements Initializable {
  @FXML
  private TextField taskField;
  @FXML
  private TextField locationField;
  @FXML
  private TextField addField;

  @FXML
  private ComboBox<String> timeEnter;
  @FXML
  private ComboBox<String> dayEnter;
  @FXML
  private ComboBox<String> timeDelete;
  @FXML
  private ComboBox<String> dayDelete;
  @FXML
  private Button deleteButton;
  @FXML
  private Button enterButton;
  @FXML
  private ListView<String> commonTaskList;
  @FXML
  private ListView<String> commonLocationList;

  @FXML
  private ColorPicker colorPicker;

    int account_id = getAccount();
    private int DE = 0;
    private int TE = 0;
    private int TD = 0;
    private int DD = 0;
    @FXML
    private GridPane matrix;
    private final Label[][] label = new Label[7][14]; //Declaring size of the Label Array
    private final Pane[][] pane = new Pane[7][14]; // Creating Pane with size of Array

    //Setting up value for the ComboBox time and day
    ObservableList<String> digTime = FXCollections.observableArrayList(
                "7:00", "8:00", "9:00","10:00","11:00","12:00","13:00","14:00","15:00","16:00",
                "17:00","18:00","19:00","20:00");
        ObservableList<String> daily = FXCollections.observableArrayList(
                "Monday", "Tuesday", "Wednesday","Thursday","Friday","Saturday","Sunday");

        @Override
        public void initialize(URL url, ResourceBundle resourceBundle) {
            timeEnter.setItems(digTime);
            dayEnter.setItems(daily);
            timeDelete.setItems(digTime);
            dayDelete.setItems(daily);
            setMatrix();
            generateLocationList();
            generateTaskList();
            chooseTask();
            chooseLocation();
        }
    //Method to load the matrix grid
    private void setMatrix(){
        //Adding pane and text to the multidimensional array, to later store item
        for (int i = 0; i < label.length; i++) {
            for (int j = 0; j < label[i].length; j++) {
                label[i][j] = new Label();
                pane[i][j] = new Pane();
                matrix.add(pane[i][j], i, j);
                //Set default background color
                pane[i][j].setStyle("-fx-background-color: #F2C2C2;");
            }
        }
        try {
            //Connect Database
            DatabaseConnection connectNow = new DatabaseConnection();
            Connection connectDB = connectNow.getConnection();
            //Select data from planner table
            preparedStatement = connectDB.prepareStatement("SELECT * FROM planner WHERE user_id
="+account_id);
            resultSet = preparedStatement.executeQuery();
            //Loop through result
            while (resultSet.next()) {
                int theDay = resultSet.getInt("day"); // Position of Horizontal Array
                int theTime = resultSet.getInt("time"); // Position of Vertical Array
                //Using the position key set the text there
                label[theDay][theTime].setText(resultSet.getString("task") +"\n" +
                        resultSet.getString("location") +"\n"+
                        resultSet.getString("addition"));
                //Display on the matrix grid
                matrix.add(label[theDay][theTime], theDay, theTime);
                //Look up the color column in database and get the hex id of it then make the position that color
                String colorHex = resultSet.getString("color").substring(2,8);
                pane[theDay][theTime].setStyle("-fx-background-color: #" +colorHex+ ";");
                GridPane.setHalignment(label[theDay][theTime], HPos.CENTER);
            }
        } catch (SQLException ex) {
            Logger.getLogger(plannerController.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    @FXML
    private void save() {
        DatabaseConnection connectNow = new DatabaseConnection(); //Connect Database
        Connection connectDB = connectNow.getConnection();
        //Get the data from the field entered
        String task = taskField.getText();
        String location = locationField.getText();
        String addition = addField.getText();
        Color paneColor = colorPicker.getValue();

        //Check all value is entered
        if (timeEnter.getValue() == null && dayEnter.getValue() == null
                || task.isEmpty() || location.isEmpty() || addition.isEmpty()) {
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setHeaderText(null);
            alert.setContentText("Please Fill in the Data!");
            alert.showAndWait();
        } else {
            //Recall Method that convert ComboBox value to that of position array(INTEGER)
            timeToArray();
            dayToArray();
            //Initializing value to the value get from the method
            int day = DE;
            int time = TE;
            boolean FLAG = false; //Setting up flag
            try {
                //Select the table planner
                preparedStatement = connectDB.prepareStatement("SELECT * FROM planner WHERE user_id
="+account_id);
                resultSet = preparedStatement.executeQuery();
                //Loop through result for the value checking if both DAY and TIME are identical to database
                //Avoiding overlapping of data
                while (resultSet.next() && !FLAG) {
                    int theDay = resultSet.getInt("day");
                    int theTime = resultSet.getInt("time");
                    if ((DE == theDay) && (TE == theTime)) {
                        FLAG = true; //When there is identical match stop this loop
                    }
                }
                //Identical match will result in alert message
                if (FLAG) {
                    Alert alert = new Alert(Alert.AlertType.ERROR);
                    alert.setHeaderText(null);
                    alert.setContentText("Date and Time already used!");
                    alert.showAndWait();
                } else {
                    //SQL commands to enter this data into new row on planner table in database
                    String insertFields = "INSERT INTO planner(task, location, addition, day, time, color, user_id)
VALUES ('";
                    String insertValues = task + "','" + location + "','" + addition + "','" + day + "','" + time + "','"
+paneColor + "','"+ account_id + "')";
                    String insertToPlanner = insertFields + insertValues;
                    try {
                        Statement statement = connectDB.createStatement();
                        statement.executeUpdate(insertToPlanner);

                        new loadTab("planner.fxml",1100,650);//Refresh the page
                        Stage stage = (Stage) enterButton.getScene().getWindow();
                        stage.close();
                    } catch (Exception e) {
                        e.printStackTrace();
                        e.getCause();
                    }
                }
            } catch (SQLException throwables) {
                throwables.printStackTrace();
            }
        }
    }
    @FXML //On action button to delete the select array
    private void deleteArray() {
        //Check if user entered both value for position array
        if(timeDelete.getValue() != null && dayDelete.getValue() != null) {
            //Recall Method that convert ComboBox value to that of position array(INTEGER)
            dayToDelete();
            timeToDelete();
            try {
                //Connect Database
                DatabaseConnection connectNow = new DatabaseConnection();
                Connection connectDB = connectNow.getConnection();
                //Selected deleting only place with the position array input
                preparedStatement = connectDB.prepareStatement("DELETE FROM planner WHERE day = " + DD +
" AND time = " + TD+ " AND user_id = " + account_id);
                preparedStatement.execute();
                new loadTab("planner.fxml", 1100, 650); //Refresh Page
                Stage stage = (Stage) deleteButton.getScene().getWindow();
                stage.close();
            } catch (SQLException ex) {
                Logger.getLogger(plannerController.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }

    //Switch for the value entered with ComboBox changing it value to position array
    private void timeToArray() {
        //Get value from the comboBox
        String x = timeEnter.getValue();
        switch (x) {
            case "7:00" -> TE = 0;
            case "8:00" -> TE = 1;
            case "9:00" -> TE = 2;
            case "10:00" -> TE = 3;
            case "11:00" -> TE = 4;
            case "12:00" -> TE = 5;
            case "13:00" -> TE = 6;
            case "14:00" -> TE = 7;
            case "15:00" -> TE = 8;
            case "16:00" -> TE = 9;
            case "17:00" -> TE = 10;
            case "18:00" -> TE = 11;
            case "19:00" -> TE = 12;
            case "20:00" -> TE = 13;
        }
    }
    private void dayToArray() {
        String y = dayEnter.getValue();
        switch (y) {
            case "Monday" -> DE = 0;
            case "Tuesday" -> DE = 1;
            case "Wednesday" -> DE = 2;
            case "Thursday" -> DE = 3;
            case "Friday" -> DE = 4;
            case "Saturday" -> DE = 5;
            case "Sunday" -> DE = 6;
        }
    }
    private void timeToDelete() {
        String x = timeDelete.getValue();
        switch (x) {
            case "7:00" -> TD = 0;
            case "8:00" -> TD = 1;
            case "9:00" -> TD = 2;
            case "10:00" -> TD = 3;
            case "11:00" -> TD = 4;
            case "12:00" -> TD = 5;
            case "13:00" -> TD = 6;
            case "14:00" -> TD = 7;
            case "15:00" -> TD = 8;
            case "16:00" -> TD = 9;
            case "17:00" -> TD = 10;
            case "18:00" -> TD = 11;
            case "19:00" -> TD = 12;
            case "20:00" -> TD = 13;
        }
    }
    private void dayToDelete() {
        String y = dayDelete.getValue();
        switch (y) {
            case "Monday" -> DD = 0;
            case "Tuesday" -> DD = 1;
            case "Wednesday" -> DD = 2;
            case "Thursday" -> DD = 3;
            case "Friday" -> DD = 4;
            case "Saturday" -> DD = 5;
            case "Sunday" -> DD = 6;
        }
    }
    //ArrayList to add task that are commonly use
    private void generateTaskList(){
        ArrayList<String> commonTask = new ArrayList<>();
        commonTask.add("Sleep");
        commonTask.add("Work");
        commonTask.add("Breakfast");
        commonTask.add("Lunch");
        commonTask.add("Dinner");
        commonTask.add("Exercise");
        commonTask.add("Leisure");
        commonTask.add("Shopping");
        //Display Task
        for (String useTask : commonTask){
            commonTaskList.getItems().add(useTask);
        }
    }
    private void generateLocationList(){
        ArrayList<String> commonLocation = new ArrayList<>();
        commonLocation.add("Home");
        commonLocation.add("Office");
        commonLocation.add("Central");
```

```
        for (String useLocation : commonLocation){
            commonLocationList.getItems().add(useLocation);
        }
    }
    //MouseOnClick action to choose the combobox
    private void chooseTask(){
        commonTaskList.setOnMouseClicked(event ->{
            String item = commonTaskList.getSelectionModel().getSelectedItem();
            taskField.setText(item);
        });
    }
    private void chooseLocation(){
        commonLocationList.setOnMouseClicked(event ->{
            String item = commonLocationList.getSelectionModel().getSelectedItem();
            locationField.setText(item);
        });
    }
}
```

NutrientController

```
package sample;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.chart.PieChart;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.paint.Color;

import java.net.URL;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.ResourceBundle;
import java.util.logging.Level;
import java.util.logging.Logger;

public class nutrientController extends insideApp implements Initializable {
```

```
@FXML
private TextField foodField;
@FXML
private TextField calField;
@FXML
private TextField carbField;
@FXML
private TextField proteinField;
@FXML
private TextField fibreField;
@FXML
private TextField fatField;
@FXML
private Label indicator;


@FXML
private TableColumn<modelNutStorage, String> colTotalCalories;
@FXML
private TableColumn<modelNutStorage, String> colTotalMass;
@FXML
private TableColumn<modelNutStorage, String> colDateAlone;
@FXML
private TableView<modelNutStorage> totalTable;

@FXML
private TableView<modelNutTable> table;
@FXML
private TableColumn<modelNutTable, String> colDate;
@FXML
private TableColumn<modelNutTable, String> colFood;
@FXML
private TableColumn<modelNutTable, String> colCalories;
@FXML
private TableColumn<modelNutTable, String> colMass;
@FXML
private TableColumn<modelNutTable, String> colCarb;
@FXML
private TableColumn<modelNutTable, String> colProtein;
@FXML
private TableColumn<modelNutTable, String> colFibre;
@FXML
private TableColumn<modelNutTable, String> colFat;
@FXML
private PieChart pieChart;
modelNutTable someNut = null;
```

```
  modelNutStorage someNutStorage = null;
  int account_id = getAccount();
  ObservableList<modelNutTable> nutList = FXCollections.observableArrayList();
  ObservableList<modelNutStorage> totalNutList = FXCollections.observableArrayList();

  @Override
  public void initialize(URL url, ResourceBundle resourceBundle) {
    load();
    loadStorage();
  }

  @FXML //Getting data from database and adding it to table
  private void refreshTable() {
    try {
      nutList.clear();
      //Connect to Database
      DatabaseConnection connectNow = new DatabaseConnection();
      Connection connectDB = connectNow.getConnection();
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
      String today = sdf.format(new Date());
      //Select data from nutrient table showing only today date
      preparedStatement = connectDB.prepareStatement("SELECT * FROM nutrient WHERE date =
'"+today+"' AND user_id = "+account_id);
      resultSet = preparedStatement.executeQuery();
      //Loop through the result from table while getting data from each row
      while (resultSet.next()){
        nutList.add(new modelNutTable(
            resultSet.getDate("date"),
            resultSet.getString("food"),
            resultSet.getDouble("calories"),
            resultSet.getDouble("mass"),
            resultSet.getDouble("carb"),
            resultSet.getDouble("protein"),
            resultSet.getDouble("fibre"),
            resultSet.getDouble("fat")));
        table.setItems(nutList);
      }
    } catch (SQLException ex) {
      Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
    }
  }
  //Method to load the table and display each data from database to column
  private void load(){
    //Connect to Database
    DatabaseConnection connectNow = new DatabaseConnection();
    connectNow.getConnection();
    refreshTable();
```

```java
        //Setting the value of each column to the respected data from method above
        colDate.setCellValueFactory(new PropertyValueFactory<>("date"));
        colFood.setCellValueFactory(new PropertyValueFactory<>("food"));
        colCalories.setCellValueFactory(new PropertyValueFactory<>("calories"));
        colMass.setCellValueFactory(new PropertyValueFactory<>("mass"));
        colCarb.setCellValueFactory(new PropertyValueFactory<>("carb"));
        colProtein.setCellValueFactory(new PropertyValueFactory<>("protein"));
        colFibre.setCellValueFactory(new PropertyValueFactory<>("fibre"));
        colFat.setCellValueFactory(new PropertyValueFactory<>("fat"));
        table.setItems(nutList);//Display the data
    }

    //Getting data from database and adding it to table
    private void loadStorage(){
        //Connect to Database
        DatabaseConnection connectNow = new DatabaseConnection();
        Connection connectDB = connectNow.getConnection();
        totalNutList.clear();
        try{
            //Select data from nutrientSummary table
            preparedStatement = connectDB.prepareStatement("SELECT * FROM nutrientSummary WHERE user_id
="+account_id);
            resultSet = preparedStatement.executeQuery();

            //Loop through the result from table while getting data from each row
            while (resultSet.next()) {
                totalNutList.add(new modelNutStorage(
                        resultSet.getDate("dateAlone"),
                        resultSet.getDouble("totalCalories"),
                        resultSet.getDouble("totalMass")));
                totalTable.setItems(totalNutList);
            }
        }catch (SQLException ex) {
            Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
        }
        //Setting the value of each column to the respected data
        colDateAlone.setCellValueFactory(new PropertyValueFactory<>("dateAlone"));
        colTotalCalories.setCellValueFactory(new PropertyValueFactory<>("totalCalories"));
        colTotalMass.setCellValueFactory(new PropertyValueFactory<>("totalMass"));
        totalTable.setItems(totalNutList); //Display the Data onto Table
    }
    @FXML
    private void save() {
        pieChart.getData().clear();//Reset Database
        //Connect Database
        DatabaseConnection connectNow = new DatabaseConnection();
        Connection connectDB = connectNow.getConnection();
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        String datenow = sdf.format(new Date());
        //Getting Data from Field
        String food = foodField.getText();
        String calories = calField.getText();
        String carb = carbField.getText();
        String protein = proteinField.getText();
        String fibre = fibreField.getText();
        String fat = fatField.getText();

        //Check if field is empty
        if (food.isEmpty() || calories.isEmpty() || carb.isEmpty()|| protein.isEmpty()|| fibre.isEmpty()|| fat.isEmpty()) {
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setHeaderText(null);
            alert.setContentText("Please Fill All DATA");
            alert.showAndWait();
        } else if(isNotNumber(calories) || isNotNumber(carb) || isNotNumber(protein) || isNotNumber(fibre)||
isNotNumber(fat)){
            Alert alertEmpty = new Alert(Alert.AlertType.ERROR);
            alertEmpty.setHeaderText(null);
            alertEmpty.setContentText("Data is not numeric!");
            alertEmpty.showAndWait();
            clean();
        }

        else {
            clean();
            //Parsing to Integer
            int tempCarb = Integer.parseInt(carb);
            int tempProtein = Integer.parseInt(protein);
            int tempFibre = Integer.parseInt(fibre);
            int tempFat = Integer.parseInt(fat);
            int temporaryMass = tempCarb + tempProtein + tempFibre + tempFat;//Calculating total Mass
            //Calculate mass percentage respected to the total mass
            int carbPercent = tempCarb/temporaryMass;
            int proteinPercent = tempProtein/temporaryMass;
            int fatPercent = tempFat/temporaryMass;
            String mass = String.valueOf(temporaryMass);

            //SQL commands to enter this data into new row on nutrient table in database
            String insertFields = "INSERT INTO nutrient(date, food, calories, mass, carb, protein, fibre, fat, user_id)
VALUES ('";
            String insertValues = datenow +"','"+ food +"','"+ calories +"','"+ mass +"','"+carb+"','"+protein+"','"+
                    fibre+"','"+fat+"','"+account_id+"')";
            String insertToNutrient = insertFields + insertValues;

            //Checking if meal is healthy and displaying message using condition from health website
```

```
            if (((0.45<carbPercent)||(carbPercent<0.65))&&((0.2<proteinPercent)||(proteinPercent<0.35))&&
                    (tempFibre>30)&&((0.2<fatPercent)||(fatPercent<0.35))){
                indicator.setText("This meal is balanced \n and healthy");
                indicator.setTextFill(Color.web("#008450",0.8));
            }else{
                indicator.setText("This meal is unbalanced");
                indicator.setTextFill(Color.web("#B81D13",0.8));
            }
            //Setting the piechart with given data
            PieChart.Data slice1 = new PieChart.Data("Carbohydrate", tempCarb);
            PieChart.Data slice2 = new PieChart.Data("Protein", tempProtein);
            PieChart.Data slice3 = new PieChart.Data("Fibre", tempFibre);
            PieChart.Data slice4 = new PieChart.Data("Fat", tempFat);
            pieChart.getData().add(slice1);
            pieChart.getData().add(slice2);
            pieChart.getData().add(slice3);
            pieChart.getData().add(slice4);

            try{
                Statement statement = connectDB.createStatement();
                statement.executeUpdate(insertToNutrient);
                refreshTable();
            }catch (Exception e){
                e.printStackTrace();
                e.getCause();
            }
            clean();
        }
    }
    @FXML
    private void deleteNutCell() {
        try {
            someNut = table.getSelectionModel().getSelectedItem();//Get value of the selected row
            DatabaseConnection connectNow = new DatabaseConnection();//Connect Database
            Connection connectDB = connectNow.getConnection();
            //SQL commands to delete this data into new row on nutrient table in database
            preparedStatement = connectDB.prepareStatement("DELETE FROM nutrient WHERE fat = 
"+someNut.getFat()+ " AND calories = "+someNut.getCalories()+ " AND user_id = " + account_id);
            preparedStatement.execute();
            refreshTable();
        } catch (SQLException ex) {
            Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    @FXML
    private void deleteNutStorage() {
        try {
```

```
        someNutStorage = totalTable.getSelectionModel().getSelectedItem();//Get value of the selected row
        DatabaseConnection connectNow = new DatabaseConnection();//Connect Database
        Connection connectDB = connectNow.getConnection();
        //SQL commands to delete this data into new row on nutrientSummary table in database
        preparedStatement = connectDB.prepareStatement("DELETE FROM nutrientSummary WHERE
totalCalories = "+someNutStorage.getTotalCalories()+
            " AND totalMass = "+someNutStorage.getTotalMass()+ " AND user_id = " + account_id);
        preparedStatement.execute();
        loadStorage();
      } catch (SQLException ex) {
        Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
      }
    }
    @FXML //Generate Table for the the total intake that day
    public void generateDailyTable(){
      //Generating today date
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
      String datenow = sdf.format(new Date());
      try {
        DatabaseConnection connectNow = new DatabaseConnection();
        Connection connectDB = connectNow.getConnection();
        //Selecting data from nutrient table from today only
        preparedStatement = connectDB.prepareStatement("SELECT * FROM nutrient WHERE
date='"+datenow+"' AND user_id = "+account_id);
        resultSet = preparedStatement.executeQuery();

        double tempTotalCalories = 0;
        double tempTotalMass = 0;
        //Loop through result of today nutrient and sum them all up
        while(resultSet.next()){
          tempTotalCalories = tempTotalCalories+resultSet.getDouble("calories");
          tempTotalMass = tempTotalMass+resultSet.getDouble("mass");
        }
        //SQL commands to enter this data into new row on nutrientSummary table in database
        String insertFields = "INSERT INTO nutrientSummary(dateAlone, totalCalories, totalMass, user_id)
VALUES ('";
        String insertValues = datenow +"','"+ tempTotalCalories +"','"+ tempTotalMass+"','"+ account_id+"')";
        String insertToTotalNutrient = insertFields + insertValues;
        Statement statement = connectDB.createStatement();
        statement.executeUpdate(insertToTotalNutrient);

      } catch (SQLException ex) {
        Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
      }
      loadStorage();
    }
    @FXML //Method to remove text in the textField
```

```
  private void clean() {
      foodField.setText(null);
      calField.setText(null);
      carbField.setText(null);
      proteinField.setText(null);
      fibreField.setText(null);
      fatField.setText(null);
  }
}
```

ModelNutTable

```
package sample;
import java.sql.Date;

public class modelNutTable {

  private Date date;
  private final String food;
  private final Double calories;
  private final Double mass;
  private final Double carb;
  private final Double protein;
  private final Double fibre;
  private final Double fat;

  public modelNutTable(Date date, String food, Double calories, Double mass, Double carb, Double protein,
Double fibre, Double fat) {
      //Setting this to the parameter, otherwise it will be confuse with same name
      this.date = date;
      this.food = food;
      this.calories = calories;
      this.mass = mass;
      this.carb = carb;
      this.protein = protein;
      this.fibre = fibre;
      this.fat = fat;
  }
  //Getter method, to get and use the value after it being stored
  public Date getDate(){
      return date;
  }
  public void setDate(Date date){
      this.date = date;
  }
```

```
    public String getFood(){
        return food;
    }
    public Double getCalories(){
        return calories;
    }
    public Double getCarb(){
        return carb;
    }
    public Double getProtein(){
        return protein;
    }
    public Double getFibre(){
        return fibre;
    }
    public Double getFat(){
        return fat;
    }
    public Double getMass(){
        return mass;
    }
}
```

ModelNutStorage

```
package sample;
import java.sql.Date;

public class modelNutStorage{

    private final Date dateAlone;
    private final double totalCalories;
    private final double totalMass;

    public modelNutStorage(Date dateAlone, double totalCalories, double totalMass) {
        //Setting this dateCal to the parameter, otherwise it will be confuse with same name
        this.dateAlone = dateAlone;
        this.totalCalories = totalCalories;
        this.totalMass = totalMass;
    }
    //Getter method, to get and use the value after it being stored
    public Date getDateAlone(){
        return dateAlone;
    }
```

```
  public Double getTotalCalories(){
      return totalCalories;
  }
  public Double getTotalMass(){
      return totalMass;
  }
}
```

TrackerController

```
package sample;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;

import java.net.URL;
import java.sql.Connection;
import java.sql.Date;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.util.ResourceBundle;
import java.util.logging.Level;
import java.util.logging.Logger;

public class trackerController extends insideApp implements Initializable {
  @FXML
  private DatePicker myDatePicker;
  @FXML
  private CheckBox menstruation;
  @FXML
  private TextField tempField;
  @FXML
  private TextField weightField;

  @FXML
  private TableView<modelTrackTable> table;
  @FXML
```

```
    private TableColumn<modelTrackTable, String> colDate;
    @FXML
    private TableColumn<modelTrackTable, String> colTemp;
    @FXML
    private TableColumn<modelTrackTable, String> colWeight;
    @FXML
    private TableColumn<modelTrackTable, String> colMens;

    modelTrackTable trackTable = null;
    int account_id = getAccount();
    ObservableList<modelTrackTable> trackerList = FXCollections.observableArrayList();

    @Override
    public void initialize(URL url, ResourceBundle resourceBundle) {
        load();
    }

    public void tempOnAction(){
        new loadTab("temp.fxml",1000,550);
    }

    //Method to load the table and display each data from database to column
    private void load(){
        //Connect to Database
        DatabaseConnection connectNow = new DatabaseConnection();
        connectNow.getConnection();
        refreshTable();
        //Setting the value of each column to the respected data from method above
        colDate.setCellValueFactory(new PropertyValueFactory<>("dateCal"));
        colTemp.setCellValueFactory(new PropertyValueFactory<>("temp"));
        colWeight.setCellValueFactory(new PropertyValueFactory<>("weight"));
        colMens.setCellValueFactory(new PropertyValueFactory<>("mens"));
        table.setItems(trackerList);//Display the data
    }

    @FXML //Getting data from database and adding it to table
    private void refreshTable() {
        try {
            trackerList.clear();
            //Connect to Database
            DatabaseConnection connectNow = new DatabaseConnection();
            Connection connectDB = connectNow.getConnection();
            //Select data from tracker table sort by date
            preparedStatement = connectDB.prepareStatement("SELECT * FROM tracker WHERE user_id
="+account_id+" ORDER by dateCal DESC" );
            resultSet = preparedStatement.executeQuery();
            //Loop through the result from table while getting data from each row
```

```java
        while (resultSet.next()){
            trackerList.add(new modelTrackTable(
                    resultSet.getDate("dateCal"),
                    resultSet.getDouble("temp"),
                    resultSet.getDouble("weight"),
                    resultSet.getString("mens")));
            table.setItems(trackerList);
        }
    } catch (SQLException ex) {
        Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
    }
}

@FXML
private void save() {
    //Connect Database
    DatabaseConnection connectNow = new DatabaseConnection();
    Connection connectDB = connectNow.getConnection();
    String mens = "No"; //Declaring Variable for menstruation
    if(menstruation.isSelected()){ mens = "Yes"; } //Check if box is tick
    boolean FLAG = false; //Declaring FLAG

    //Check if value is insert in field
    if(myDatePicker.getValue() == null || weightField.getText() == null || tempField.getText() == null){
        Alert alertEmpty = new Alert(Alert.AlertType.ERROR);
        alertEmpty.setHeaderText(null);
        alertEmpty.setContentText("Please fill in the data");
        alertEmpty.showAndWait();
    //Making sure that the enter value is number and not integer
    } else if(isNotNumber(weightField.getText()) || isNotNumber(tempField.getText())){
        Alert alertEmpty = new Alert(Alert.AlertType.ERROR);
        alertEmpty.setHeaderText(null);
        alertEmpty.setContentText("Data is not numeric!");
        alertEmpty.showAndWait();
        clean();
    }
    else{
        Double tempStore= Double.parseDouble(tempField.getText()); //Getting from field
        tempStore = Math.round(tempStore*10.0)/10.0; //Rounding it to 2 sf
        String temp = String.valueOf(tempStore); //Parsing back to String to be insert

        Double weightStore= Double.parseDouble(weightField.getText()); //Getting from field
        weightStore = Math.round(weightStore*10.0)/10.0; //Rounding it to 2 sf
        String weight = String.valueOf(weightStore); //Parsing back to String to be insert

        String dateCal = String.valueOf(myDatePicker.getValue()); //Parsing back to String to be insert
        try {
                //Select data from tracker table sort by date and user_id via SQL command
                preparedStatement = connectDB.prepareStatement("SELECT * FROM tracker WHERE user_id
="+account_id+" ORDER by user_id, datecal DESC");
                resultSet = preparedStatement.executeQuery();
                //Loop through result to check if Date already exist
                while(resultSet.next() && !FLAG) {
                    Date date = resultSet.getDate("dateCal");
                    DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
                    String calDay = dateFormat.format(date);
                    if (calDay.equals(dateCal)) {
                        FLAG = true;
                    }
                //If date already exist
                }if(FLAG){
                    Alert alertEmpty = new Alert(Alert.AlertType.ERROR);
                    alertEmpty.setHeaderText(null);
                    alertEmpty.setContentText("Date Already been used");
                    alertEmpty.showAndWait();
                }
                //If date doesn't already exist
                else {
                    clean();
                    //SQL commands to enter this data into new row on tracker table in database
                    String insertFields = "INSERT INTO tracker(dateCal, temp, weight, mens, user_id) VALUES ('";
                    String insertValues = dateCal +"','"+ temp +"','"+ weight +"','"+ mens+"','"+ account_id +"')";
                    String insertToCalendar = insertFields + insertValues;
                    try{
                        Statement statement = connectDB.createStatement();
                        statement.executeUpdate(insertToCalendar);
                        refreshTable();
                    }catch (Exception e){
                        e.printStackTrace();
                        e.getCause();
                    }
                }
            } catch (SQLException throwables) {
                throwables.printStackTrace();
            }
        }
    }
    @FXML
    private void deleteCell() {
        try {
            trackTable = table.getSelectionModel().getSelectedItem(); //Get value of the selected row
            DatabaseConnection connectNow = new DatabaseConnection(); //Connect Database
            Connection connectDB = connectNow.getConnection();
            //SQL commands to delete this data into new row on tracker table in database
            preparedStatement = connectDB.prepareStatement("DELETE FROM tracker WHERE temp
="+trackTable.getTemp()+
                    " AND weight ="+trackTable.getWeight()+ " AND user_id = " + account_id);
            preparedStatement.execute();
            //Load table again with new value
            refreshTable();
        } catch (SQLException ex) {
            Logger.getLogger(trackerController.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    @FXML //Method to remove text in the textField
    private void clean() {
        tempField.setText(null);
        weightField.setText(null);
    }
    public void getDateNow(){
        LocalDate myDate = myDatePicker.getValue();
    }

}
```

ModelTrackTable

```
package sample;
import java.sql.Date;

public class modelTrackTable {
    private final Date dateCal;
    private final Double temp;
    private final Double weight;
    private final String mens;

    public modelTrackTable(Date dateCal, Double temp, Double weight, String mens) {
        //Setting this dateCal to the parameter, otherwise it will be confuse with same name
        this.dateCal = dateCal;
        this.temp = temp;
        this.weight = weight;
        this.mens = mens;
    }

    //Getter method, to get and use the value after it being stored
    public Date getDateCal(){ return this.dateCal; }
    public Double getTemp(){ return this.temp; }
    public Double getWeight(){ return this.weight; }
    public String getMens(){ return this.mens; }
}
```

tempGraphController

```
package sample;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.chart.LineChart;
import javafx.scene.chart.XYChart;

import java.net.URL;
import java.sql.Connection;
import java.sql.Date;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ResourceBundle;

public class tempGraphController extends insideApp implements Initializable {

   @FXML
   private LineChart<?, ?> tempGraph; //Declaring LineChart
   int account_id = getAccount();

   @Override
   public void initialize(URL url, ResourceBundle rb) {
         generate();
   }

   //Generate Temperature Graph
   public void generate(){
       //Connect to Database
       DatabaseConnection connectNow = new DatabaseConnection();
       Connection connectDB = connectNow.getConnection();
       try {
           //Selecting data from database order by date
           preparedStatement = connectDB.prepareStatement("SELECT * FROM tracker WHERE user_id
="+account_id+" ORDER by dateCal");
           resultSet = preparedStatement.executeQuery();
           XYChart.Series temperature = new XYChart.Series();
           //Loop through result and set X axis to date and Y axis to temperature
           while(resultSet.next()) {
               temperature.getData().add(new XYChart.Data(convertDateToString(resultSet.getDate("dateCal")),
resultSet.getDouble("temp")));
```

```
            }
            tempGraph.getData().addAll(temperature);
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
    }
    //Method to convert Date to String
    public static String convertDateToString(Date indate) {
        String dateString = null;
        SimpleDateFormat sdfr = new SimpleDateFormat("dd/MMM/yyyy");
        try{
            dateString = sdfr.format(indate);
        }catch (Exception ex ){
            System.out.println(ex);
        }
        return dateString;
    }
}
```

.FXML Source Code

Lobby.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.image.*?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.*?>
<AnchorPane prefHeight="650.0" prefWidth="1100.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.LobbyController">
  <children>
     <AnchorPane prefHeight="220.0" prefWidth="1100.0" style="-fx-background-color: #F2C2C2;">
        <children>
           <Button fx:id="close" layoutX="1025.0" layoutY="14.0" mnemonicParsing="false"
onAction="#closeOnAction" prefHeight="65.0" prefWidth="66.0" style="-fx-background-color: #F2C2C2;"
text="X" textFill="WHITE">
              <font>
                 <Font size="26.0" />
              </font></Button>
           <ProgressBar fx:id="myProgressBar" layoutX="478.0" layoutY="141.0" prefHeight="55.0"
prefWidth="586.0" progress="0.0" />
```

```
        <Label fx:id="progressLabel" layoutX="788.0" layoutY="149.0" prefHeight="39.0" prefWidth="160.0"
text="\%">
          <font>
            <Font size="24.0" />
          </font>
        </Label>
        <DatePicker fx:id="myDatePicker" layoutX="478.0" layoutY="88.0" onAction="#getDateNow"
prefHeight="26.0" prefWidth="205.0" />
        <Label fx:id="dayLabel" layoutX="967.0" layoutY="79.0" prefHeight="39.0" prefWidth="58.0">
          <font>
            <Font size="24.0" />
          </font>
        </Label>
        <Label layoutX="51.0" layoutY="78.0" text="Welcome Back!">
          <font>
            <Font size="60.0" />
          </font>
        </Label>
        <Label layoutX="700.0" layoutY="79.0" prefHeight="39.0" prefWidth="275.0" text="Day since your
pregnant:">
          <font>
            <Font size="24.0" />
          </font>
        </Label>
      </children></AnchorPane>
    <AnchorPane layoutY="220.0" prefHeight="430.0" prefWidth="1100.0">
      <children>
        <AnchorPane prefHeight="430.0" prefWidth="1100.0" style="-fx-background-color: #FFFFFF;">
          <children>
            <Button fx:id="trackerButton" contentDisplay="TOP" layoutX="366.0" mnemonicParsing="false"
onAction="#calendarOnAction" prefHeight="430.0" prefWidth="366.0" style="-fx-background-color:
#FFFFFF;" text="Ovulation&#10;Tracker" textAlignment="CENTER">
              <font>
                <Font name="System Font" size="50.0" />
              </font>
              <graphic>
                <ImageView fx:id="topLeftIconView" fitHeight="250.0" fitWidth="250.0" pickOnBounds="true"
preserveRatio="true">
                  <image>
                    <Image url="@../../Image/PREG.png" />
                  </image>
                </ImageView>
              </graphic>
            </Button>
            <Button fx:id="nutrientButton" contentDisplay="TOP" mnemonicParsing="false"
onAction="#nutrientOnAction" prefHeight="430.0" prefWidth="366.0" style="-fx-background-color: #FFFFFF;"
text="Nutrient" textAlignment="CENTER">
```

```
              <font>
                <Font name="System Font" size="50.0" />
              </font>
              <graphic>
                <ImageView fx:id="botRightIconView" fitHeight="250.0" fitWidth="250.0" pickOnBounds="true"
preserveRatio="true">
                  <image>
                    <Image url="@../../Image/PIE_CHART.png" />
                  </image>
                </ImageView>
              </graphic>
            </Button>
            <Button fx:id="plannerButton" contentDisplay="TOP" layoutX="732.0" mnemonicParsing="false"
onAction="#plannerOnAction" prefHeight="430.0" prefWidth="366.0" style="-fx-background-color: #FFFFFF;"
text="Day Planner" textAlignment="CENTER">
              <font>
                <Font name="System Font" size="50.0" />
              </font>
              <graphic>
                <ImageView fx:id="botLeftIconView" fitHeight="250.0" fitWidth="250.0" pickOnBounds="true"
preserveRatio="true">
                  <image>
                    <Image url="@../../Image/CALENDAR.png" />
                  </image></ImageView>
              </graphic>
            </Button>
          </children></AnchorPane>
      </children>
    </AnchorPane>
  </children>
</AnchorPane>
```

Login.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.*?>
<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"
prefHeight="400.0" prefWidth="520.0" style="-fx-background-color: #EEEEEE;"
xmlns="http://javafx.com/javafx/16" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="sample.LoginController">
  <left>
```

```
      <AnchorPane prefHeight="400.0" prefWidth="110.0" BorderPane.alignment="CENTER" />
   </left>
  <right>
      <AnchorPane prefHeight="400.0" prefWidth="110.0" BorderPane.alignment="CENTER" />
  </right>
  <center>
    <AnchorPane prefHeight="200.0" prefWidth="200.0" BorderPane.alignment="CENTER">
       <children>
            <Label layoutX="7.0" layoutY="140.0" text="Username" />
            <TextField fx:id="usernameTextField" layoutX="78.0" layoutY="135.0" prefHeight="26.0"
prefWidth="173.0" promptText="Username" />
            <ImageView fx:id="lockImageView" fitHeight="70.0" fitWidth="256.0" layoutX="108.0"
layoutY="17.0" pickOnBounds="true" preserveRatio="true">
                <image>
                    <Image url="@../../Image/logo.png" />
                </image>
            </ImageView>
            <Label layoutX="8.0" layoutY="192.0" text="Password" />
            <PasswordField fx:id="enterPasswordField" layoutX="78.0" layoutY="187.0" prefHeight="26.0"
prefWidth="173.0" promptText="Password" />
            <Button fx:id="loginButton" alignment="CENTER" layoutX="55.0" layoutY="247.0"
mnemonicParsing="false" onAction="#loginButtonOnAction" prefHeight="25.0" prefWidth="200.0"
style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2;" text="Login" textFill="WHITE" />
            <Button fx:id="cancelButton" alignment="CENTER" layoutX="55.0" layoutY="284.0"
mnemonicParsing="false" onAction="#cancelButtonOnAction" prefHeight="25.0" prefWidth="200.0"
style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2;" text="Cancel" textFill="WHITE" />
          <Button fx:id="signupButton" alignment="CENTER" layoutX="55.0" layoutY="320.0"
mnemonicParsing="false" onAction="#signupButtonOnAction" prefHeight="25.0" prefWidth="200.0"
style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2;" text="Sign Up" textFill="WHITE" />
       </children>
    </AnchorPane>
  </center>
</BorderPane>
```

Register.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
```

```
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane prefHeight="445.0" prefWidth="520.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.RegisterController">
  <children>
    <AnchorPane prefHeight="82.0" prefWidth="520.0" style="-fx-background-color: #FFC0CB;">
      <children>
        <ImageView fx:id="shieldImageView" fitHeight="42.0" fitWidth="69.0" layoutX="32.0" layoutY="20.0"
pickOnBounds="true" preserveRatio="true">
          <image>
            <Image url="@../../Image/logo.png" />
          </image>
        </ImageView>
        <Label layoutX="97.0" layoutY="33.0" text="User Registration" textFill="WHITE">
          <font>
            <Font size="18.0" />
          </font>
        </Label>
      </children>
    </AnchorPane>
    <Label layoutX="93.0" layoutY="106.0" text="Lastname" />
    <TextField fx:id="firstnameTextField" layoutX="93.0" layoutY="123.0" prefHeight="25.0"
prefWidth="336.0" promptText="Lastname" />
    <Label layoutX="94.0" layoutY="159.0" text="Firstname" />
    <TextField fx:id="lastnameTextField" layoutX="93.0" layoutY="176.0" prefHeight="25.0"
prefWidth="336.0" promptText="Firstname" />
    <Label layoutX="93.0" layoutY="207.0" text="Username" />
    <TextField fx:id="usernameTextField" layoutX="93.0" layoutY="224.0" prefHeight="25.0"
prefWidth="336.0" promptText="Username" />
    <Label layoutX="93.0" layoutY="257.0" text="Password" />
    <PasswordField fx:id="setPasswordField" layoutX="93.0" layoutY="274.0" prefHeight="25.0"
prefWidth="336.0" promptText="Password" />
    <Label layoutX="93.0" layoutY="309.0" text="Confirm Password" />
    <PasswordField fx:id="confirmPasswordField" layoutX="93.0" layoutY="326.0" prefHeight="25.0"
prefWidth="336.0" promptText="Confirm Password" />
    <Button fx:id="registerButton" layoutX="94.0" layoutY="364.0" mnemonicParsing="false"
onAction="#registerButtonOnAction" prefHeight="26.0" prefWidth="336.0" style="-fx-background-color:
#FFC0CB#FFC0CB;" text="Register" textFill="WHITE" />
    <Button fx:id="closeButton" layoutX="93.0" layoutY="398.0" mnemonicParsing="false" onAction="#close"
prefHeight="26.0" prefWidth="336.0" style="-fx-background-color: #FFC0CB#FFC0CB;" text="Back"
textFill="WHITE" />
  </children>
</AnchorPane>
```

Planner.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.text.*?>
<AnchorPane prefHeight="650.0" prefWidth="1100.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.plannerController">
   <children>
     <ScrollPane prefHeight="1486.0" prefWidth="833.0">
        <content>
             <AnchorPane prefHeight="2141.0" prefWidth="810.0" translateY="100.0">
              <children>
                 <AnchorPane layoutX="10.0" layoutY="65.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2;" translateY="10.0">
                    <children>
                       <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="7:00"
textAlignment="CENTER" translateX="-10.0">
                          <font>
                             <Font size="32.0" />
                          </font>
                       </Text>
                    </children>
                 </AnchorPane>
                 <AnchorPane layoutX="115.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
                    <children>
                       <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Mon"
textAlignment="CENTER">
                          <font>
                             <Font size="30.0" />
                          </font>
                       </Text>
                    </children>
                 </AnchorPane>
                 <GridPane fx:id="matrix" gridLinesVisible="true" hgap="5.0" layoutX="115.0" layoutY="75.0"
prefHeight="1115.0" prefWidth="695.0" vgap="5.0">
                   <columnConstraints>
                     <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
                       <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
                     <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
                       <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
                       <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
                       <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
```

```
                <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
              </columnConstraints>
              <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
              </rowConstraints>
            </GridPane>
            <AnchorPane layoutX="10.0" layoutY="155.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
               <children>
                  <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="8:00"
textAlignment="CENTER" translateX="-10.0">
                     <font>
                        <Font size="32.0" />
                     </font>
                  </Text>
               </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="235.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
               <children>
                  <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="9:00"
textAlignment="CENTER" translateX="-10.0">
                     <font>
                        <Font size="32.0" />
                     </font>
                  </Text>
               </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="315.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
               <children>
                  <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="10:00"
textAlignment="CENTER" translateX="-10.0">
                     <font>
```

```
                    <Font size="32.0" />
                  </font>
                </Text>
              </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="395.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
              <children>
                <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="11:00"
textAlignment="CENTER" translateX="-10.0">
                  <font>
                    <Font size="32.0" />
                  </font>
                </Text>
              </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="475.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
              <children>
                <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="12:00"
textAlignment="CENTER" translateX="-10.0">
                  <font>
                    <Font size="32.0" />
                  </font>
                </Text>
              </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="555.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
              <children>
                <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="13:00"
textAlignment="CENTER" translateX="-10.0">
                  <font>
                    <Font size="32.0" />
                  </font>
                </Text>
              </children>
            </AnchorPane>
            <AnchorPane layoutX="10.0" layoutY="635.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
              <children>
                <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="14:00"
textAlignment="CENTER" translateX="-10.0">
                  <font>
                    <Font size="32.0" />
                  </font>
                </Text>
```

```
      </children>
    </AnchorPane>
    <AnchorPane layoutX="10.0" layoutY="715.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
      <children>
        <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="15:00"
textAlignment="CENTER" translateX="-10.0">
          <font>
            <Font size="32.0" />
          </font>
        </Text>
      </children>
    </AnchorPane>
    <AnchorPane layoutX="10.0" layoutY="795.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
      <children>
        <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="16:00"
textAlignment="CENTER" translateX="-10.0">
          <font>
            <Font size="32.0" />
          </font>
        </Text>
      </children>
    </AnchorPane>
    <AnchorPane layoutX="10.0" layoutY="875.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
      <children>
        <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="17:00"
textAlignment="CENTER" translateX="-10.0">
          <font>
            <Font size="32.0" />
          </font>
        </Text>
      </children>
    </AnchorPane>
    <AnchorPane layoutX="10.0" layoutY="955.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
      <children>
        <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="18:00"
textAlignment="CENTER" translateX="-10.0">
          <font>
            <Font size="32.0" />
          </font>
        </Text>
      </children>
    </AnchorPane>
    <AnchorPane layoutX="10.0" layoutY="1035.0" prefHeight="75.0" prefWidth="100.0"
```

```
style="-fx-background-color: #F2C2C2 #F2C2C2;">
             <children>
               <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="19:00"
textAlignment="CENTER" translateX="-10.0">
                 <font>
                   <Font size="32.0" />
                 </font>
               </Text>
             </children>
           </AnchorPane>
           <AnchorPane layoutX="10.0" layoutY="1115.0" prefHeight="75.0" prefWidth="100.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
             <children>
               <Text layoutX="20.0" layoutY="50.0" strokeType="OUTSIDE" strokeWidth="0.0" text="20:00"
textAlignment="CENTER" translateX="-10.0">
                 <font>
                   <Font size="32.0" />
                 </font>
               </Text>
             </children>
           </AnchorPane>
           <AnchorPane layoutX="215.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
             <children>
               <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Tue"
textAlignment="CENTER">
                 <font>
                   <Font size="30.0" />
                 </font>
               </Text>
             </children>
           </AnchorPane>
           <AnchorPane layoutX="315.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
             <children>
               <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Wed"
textAlignment="CENTER">
                 <font>
                   <Font size="30.0" />
                 </font>
               </Text>
             </children>
           </AnchorPane>
           <AnchorPane layoutX="415.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
             <children>
               <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Thur"
```

```
textAlignment="CENTER">
                    <font>
                        <Font size="30.0" />
                    </font>
                </Text>
            </children>
        </AnchorPane>
        <AnchorPane layoutX="515.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
            <children>
                <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Fri"
textAlignment="CENTER">
                    <font>
                        <Font size="30.0" />
                    </font>
                </Text>
            </children>
        </AnchorPane>
        <AnchorPane layoutX="615.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
            <children>
                <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Sat"
textAlignment="CENTER">
                    <font>
                        <Font size="30.0" />
                    </font>
                </Text>
            </children>
        </AnchorPane>
        <AnchorPane layoutX="715.0" layoutY="20.0" prefHeight="50.0" prefWidth="95.0"
style="-fx-background-color: #F2C2C2 #F2C2C2;">
            <children>
                <Text layoutX="15.0" layoutY="35.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Sun"
textAlignment="CENTER">
                    <font>
                        <Font size="30.0" />
                    </font>
                </Text>
            </children>
        </AnchorPane>
      </children>
    </AnchorPane>
  </content>
</ScrollPane>
   <AnchorPane prefHeight="100.0" prefWidth="1100.0" style="-fx-background-color: #F2C2C2 #F2C2C2;">
     <children>
        <Text layoutX="401.0" layoutY="62.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Planner">
```

```
                <font>
                    <Font size="50.0" />
                </font>
            </Text>
        </children>
    </AnchorPane>
    <TextField fx:id="locationField" layoutX="860.0" layoutY="168.0" prefHeight="30.0" prefWidth="200.0"
promptText="Location" />
    <TextField fx:id="taskField" layoutX="860.0" layoutY="118.0" prefHeight="30.0" prefWidth="200.0"
promptText="Task" />
    <TextField fx:id="addField" layoutX="860.0" layoutY="218.0" prefHeight="30.0" prefWidth="200.0"
promptText="Additional Info" />
    <Button fx:id="enterButton" layoutX="1003.0" layoutY="309.0" mnemonicParsing="false"
onAction="#save" prefHeight="26.0" prefWidth="63.0" text="Enter" />
    <ComboBox fx:id="timeEnter" layoutX="859.0" layoutY="264.0" prefHeight="26.0" prefWidth="83.0"
promptText="Time" visibleRowCount="15" />
    <ComboBox fx:id="dayEnter" layoutX="948.0" layoutY="264.0" prefHeight="26.0" prefWidth="120.0"
promptText="Day" visibleRowCount="15" />
    <ComboBox fx:id="timeDelete" layoutX="861.0" layoutY="351.0" prefHeight="26.0" prefWidth="80.0"
promptText="Time" visibleRowCount="15" />
    <ComboBox fx:id="dayDelete" layoutX="948.0" layoutY="351.0" prefHeight="26.0" prefWidth="120.0"
promptText="Day" visibleRowCount="15" />
    <Button fx:id="deleteButton" layoutX="910.0" layoutY="394.0" mnemonicParsing="false"
onAction="#deleteArray" prefHeight="26.0" prefWidth="83.0" text="Delete" />
    <Button fx:id="homeButton" layoutX="37.0" layoutY="28.0" mnemonicParsing="false"
onAction="#homeOnAction" prefHeight="50.0" prefWidth="69.0" style="-fx-background-color: #F2C2C2
#F2C2C2 #F2C2C2;" text="&lt;---" textAlignment="CENTER" textFill="WHITE"
textOverrun="CENTER_ELLIPSIS">
        <font>
            <Font name="System Bold" size="20.0" />
        </font>
    </Button>
    <ListView fx:id="commonTaskList" layoutX="841.0" layoutY="437.0" prefHeight="163.0"
prefWidth="120.0" />
    <ListView fx:id="commonLocationList" layoutX="966.0" layoutY="437.0" prefHeight="163.0"
prefWidth="120.0" />
    <ColorPicker fx:id="colorPicker" layoutX="861.0" layoutY="309.0" prefHeight="26.0" prefWidth="131.0" />
  </children>
</AnchorPane>
```

Nutrient.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```xml
<?import javafx.scene.chart.PieChart?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>
<?import javafx.scene.text.Text?>

<AnchorPane prefHeight="650.0" prefWidth="1100.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.nutrientController">
   <children>
      <AnchorPane prefHeight="100.0" prefWidth="1100.0" style="-fx-background-color: #F2C2C2 #F2C2C2;">
         <children>
            <Text layoutX="432.0" layoutY="78.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Nutrient
Tracker">
               <font>
                  <Font size="50.0" />
               </font>
            </Text>
         </children>
      </AnchorPane>
      <AnchorPane layoutY="100.0" prefHeight="550.0" prefWidth="1100.0">
       <children>
         <TableView fx:id="table" layoutX="11.0" layoutY="24.0" prefHeight="196.0" prefWidth="820.0">
           <columns>
             <TableColumn fx:id="colDate" prefWidth="111.66212177276611" text="date" />
             <TableColumn fx:id="colFood" minWidth="4.246917724609375" prefWidth="151.8378677368164"
text="Food" />
             <TableColumn fx:id="colCalories" prefWidth="97.02169799804688" text="calories" />
             <TableColumn fx:id="colMass" minWidth="0.0" prefWidth="79.2825927734375" text="mass" />
             <TableColumn fx:id="colCarb" prefWidth="81.2027587890625" text="carb" />
             <TableColumn fx:id="colProtein" prefWidth="87.3623046875" text="protein" />
             <TableColumn fx:id="colFibre" prefWidth="83.2825927734375" text="fibre" />
             <TableColumn fx:id="colFat" prefWidth="84.8623046875" text="fat" />
           </columns>
         </TableView>
         <TextField fx:id="foodField" layoutX="852.0" layoutY="26.0" prefHeight="30.0" prefWidth="200.0"
promptText="Course" />
         <TextField fx:id="calField" layoutX="851.0" layoutY="65.0" prefHeight="30.0" prefWidth="152.0"
promptText="Calories" />
         <TableView fx:id="totalTable" layoutX="13.0" layoutY="245.0" prefHeight="252.0" prefWidth="550.0">
           <columns>
             <TableColumn fx:id="colDateAlone" prefWidth="192.41211318969727" text="date" />
             <TableColumn fx:id="colTotalCalories" prefWidth="137.70553588867188" text="total calories" />
             <TableColumn fx:id="colTotalMass" minWidth="0.0" prefWidth="217.0" text="total mass" />
      </columns>
    </TableView>
    <Button layoutX="309.0" layoutY="510.0" mnemonicParsing="false" onAction="#generateDailyTable"
prefHeight="26.0" prefWidth="83.0" text="Generate" />
    <TextField fx:id="carbField" layoutX="849.0" layoutY="106.0" prefHeight="30.0" prefWidth="152.0"
promptText="Carb Mass" />
    <TextField fx:id="proteinField" layoutX="850.0" layoutY="145.0" prefHeight="30.0" prefWidth="152.0"
promptText="Protein Mass" />
    <TextField fx:id="fibreField" layoutX="849.0" layoutY="187.0" prefHeight="30.0" prefWidth="152.0"
promptText="Fibre Mass" />
    <TextField fx:id="fatField" layoutX="851.0" layoutY="229.0" prefHeight="30.0" prefWidth="152.0"
promptText="Fats Mass" />
    <Button layoutX="907.0" layoutY="314.0" mnemonicParsing="false" onAction="#deleteNutCell"
prefHeight="26.0" prefWidth="83.0" text="Delete" />
    <Button layoutX="906.0" layoutY="276.0" mnemonicParsing="false" onAction="#save"
prefHeight="26.0" prefWidth="83.0" text="Save" />
    <Button layoutX="907.0" layoutY="353.0" mnemonicParsing="false" onAction="#clean"
prefHeight="26.0" prefWidth="83.0" text="Clear" />
    <PieChart fx:id="pieChart" labelLineLength="100.0" layoutX="573.0" layoutY="235.0"
prefHeight="301.0" prefWidth="264.0" title="Nutrition Pie Chart" />
    <Text layoutX="1009.0" layoutY="86.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/cals">
      <font>
        <Font size="16.0" />
      </font>
    </Text>
    <Text layoutX="1008.0" layoutY="125.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/gram">
      <font>
        <Font size="16.0" />
      </font>
    </Text>
    <Text layoutX="1008.0" layoutY="165.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/gram">
      <font>
        <Font size="16.0" />
      </font>
    </Text>
    <Text layoutX="1008.0" layoutY="207.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/gram">
      <font>
        <Font size="16.0" />
      </font>
    </Text>
    <Text layoutX="1008.0" layoutY="248.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/gram">
      <font>
        <Font size="16.0" />
      </font>
    </Text>
    <Button layoutX="197.0" layoutY="510.0" mnemonicParsing="false" onAction="#deleteNutStorage"
prefHeight="26.0" prefWidth="83.0" text="Delete" />
```

```
        <Label fx:id="indicator" alignment="CENTER" layoutX="829.0" layoutY="402.0" prefHeight="140.0"
prefWidth="258.0">
          <font>
            <Font size="20.0" />
          </font>
        </Label>
      </children></AnchorPane>
    <Button fx:id="homeButton" layoutX="40.0" layoutY="27.0" mnemonicParsing="false"
onAction="#homeOnAction" prefHeight="50.0" prefWidth="69.0" style="-fx-background-color: #F2C2C2
#F2C2C2 #F2C2C2;" text="&lt;---" textAlignment="CENTER" textFill="WHITE"
textOverrun="CENTER_ELLIPSIS">
      <font>
        <Font name="System Bold" size="20.0" />
      </font>
    </Button>
  </children>
</AnchorPane>
```

Tracker.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.*?>
<AnchorPane prefHeight="650.0" prefWidth="1100.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.trackerController">
  <children>
    <AnchorPane prefHeight="100.0" prefWidth="1100.0" style="-fx-background-color: #F2C2C2 #F2C2C2;">
      <children>
        <Text layoutX="432.0" layoutY="78.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Calendar">
          <font>
            <Font size="50.0" />
          </font>
        </Text>
        <Button fx:id="homeButton" layoutX="30.0" layoutY="25.0" mnemonicParsing="false"
onAction="#homeOnAction" prefHeight="50.0" prefWidth="69.0" style="-fx-background-color: #F2C2C2
#F2C2C2 #F2C2C2;" text="&lt;---" textAlignment="CENTER" textFill="WHITE"
textOverrun="CENTER_ELLIPSIS">
          <font>
            <Font name="System Bold" size="20.0" />
            </font></Button>
        </children>
      </AnchorPane>
      <AnchorPane layoutY="100.0" prefHeight="550.0" prefWidth="1100.0">
        <children>
          <TableView fx:id="table" layoutX="36.0" layoutY="47.0" prefHeight="442.0" prefWidth="776.0">
            <columns>
              <TableColumn fx:id="colDate" prefWidth="220.1175537109375" text="Date" />
              <TableColumn fx:id="colTemp" minWidth="4.246917724609375" prefWidth="199.8824005126953"
text="temp" />
                <TableColumn fx:id="colWeight" prefWidth="103.72097778320312" text="weight" />
                <TableColumn fx:id="colMens" minWidth="0.0" prefWidth="250.5" text="mens" />
            </columns>
          </TableView>
          <CheckBox fx:id="menstruation" layoutX="878.0" layoutY="158.0" mnemonicParsing="false"
prefHeight="16.0" prefWidth="175.0" text="Menstruation" />
          <DatePicker fx:id="myDatePicker" layoutX="868.0" layoutY="192.0" onAction="#getDateNow"
prefHeight="26.0" prefWidth="205.0" />
          <Button fx:id="tempButton" layoutX="829.0" layoutY="371.0" mnemonicParsing="false"
onAction="#tempOnAction" prefHeight="50.0" prefWidth="237.0" style="-fx-background-color: #F2C2C2
#F2C2C2 #F2C2C2 #F2C2C2;" text="Temperature Analysis" textAlignment="CENTER"
textOverrun="CENTER_ELLIPSIS">
            <font>
              <Font size="20.0" />
            </font>
          </Button>
          <Button layoutX="908.0" layoutY="323.0" mnemonicParsing="false" onAction="#deleteCell"
prefHeight="26.0" prefWidth="83.0" style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2 #F2C2C2;"
text="Delete" />
          <Button layoutX="908.0" layoutY="240.0" mnemonicParsing="false" onAction="#save"
prefHeight="26.0" prefWidth="83.0" style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2 #F2C2C2;"
text="Save" />
          <Button layoutX="910.0" layoutY="280.0" mnemonicParsing="false" onAction="#clean"
prefHeight="26.0" prefWidth="83.0" style="-fx-background-color: #F2C2C2 #F2C2C2 #F2C2C2 #F2C2C2;"
text="Clear" />
          <Text layoutX="1044.0" layoutY="78.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/°C" />
          <Text layoutX="1044.0" layoutY="126.0" strokeType="OUTSIDE" strokeWidth="0.0" text="/Kg" />
        </children></AnchorPane>
      <TextField fx:id="tempField" layoutX="870.0" layoutY="160.0" prefHeight="30.0" prefWidth="169.0"
promptText="Temperature" />
      <TextField fx:id="weightField" layoutX="870.0" layoutY="210.0" prefHeight="30.0" prefWidth="169.0"
promptText="Weight" />
  </children>
</AnchorPane>
```

Temp.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.chart.CategoryAxis?>
<?import javafx.scene.chart.LineChart?>
<?import javafx.scene.chart.NumberAxis?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>
<?import javafx.scene.text.Text?>
<AnchorPane prefHeight="550.0" prefWidth="1000.0" xmlns="http://javafx.com/javafx/16"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.tempGraphController">
  <children>
      <AnchorPane prefHeight="100.0" prefWidth="1100.0" style="-fx-background-color: #F2C2C2 #F2C2C2;">
         <children>
            <Text layoutX="380.0" layoutY="73.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Temperature
Graph">
               <font>
                  <Font size="50.0" />
               </font>
            </Text>
         <Button fx:id="close" layoutX="59.0" layoutY="12.0" mnemonicParsing="false"
onAction="#closeOnAction" prefHeight="75.0" prefWidth="96.0" style="-fx-background-color: #F2C2C2
#F2C2C2 #F2C2C2 #F2C2C2;" text="X" textFill="WHITE">
            <font>
               <Font size="40.0" />
            </font></Button>
         </children>
      </AnchorPane>
      <AnchorPane layoutY="100.0" prefHeight="550.0" prefWidth="1100.0">
       <children>
         <LineChart fx:id="tempGraph" alternativeColumnFillVisible="true" layoutX="143.0" layoutY="41.0"
nodeOrientation="LEFT_TO_RIGHT" prefHeight="373.0" prefWidth="813.0" title="Temperature Graph">
           <xAxis>
             <CategoryAxis label="Date" side="BOTTOM" tickLabelRotation="39.5" fx:id="x" />
           </xAxis>
           <yAxis>
             <NumberAxis fx:id="y" autoRanging="false" label="Temperature/C" lowerBound="35.0"
minorTickCount="10" prefWidth="48.0" side="LEFT" tickLabelGap="2.0" upperBound="40.0" />
           </yAxis>
         </LineChart>
       </children></AnchorPane>
  </children>
</AnchorPane>
```