COMP6699001

Object Oriented Programming Final Report

Rafael Angelo Christianto
2702342773
L2BC

I) Introduction

By making this project I do think I can get the coding logic and experience on how to make a login and sign up system, and additionally store the data. This can be a really useful experience for me in the future as a backend developer.

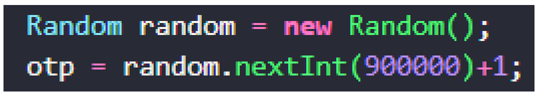
II) Problem Description

In this program, the user who does not have an account yet in the FARLEAF INDUSTRIES, they are allowed to sign up and make an account for themselves. But, not only that, I want to add something extra, so I decided to make it a 2-step-verification process that sends the user email to verify their account. Thus, log in to the program.

III) Solution

The solution I thought of is to store the data in a database, which in this case I used MySQL. And by connecting it with my Java code by making use of the Java Database Connectivity (JDBC), it can work.

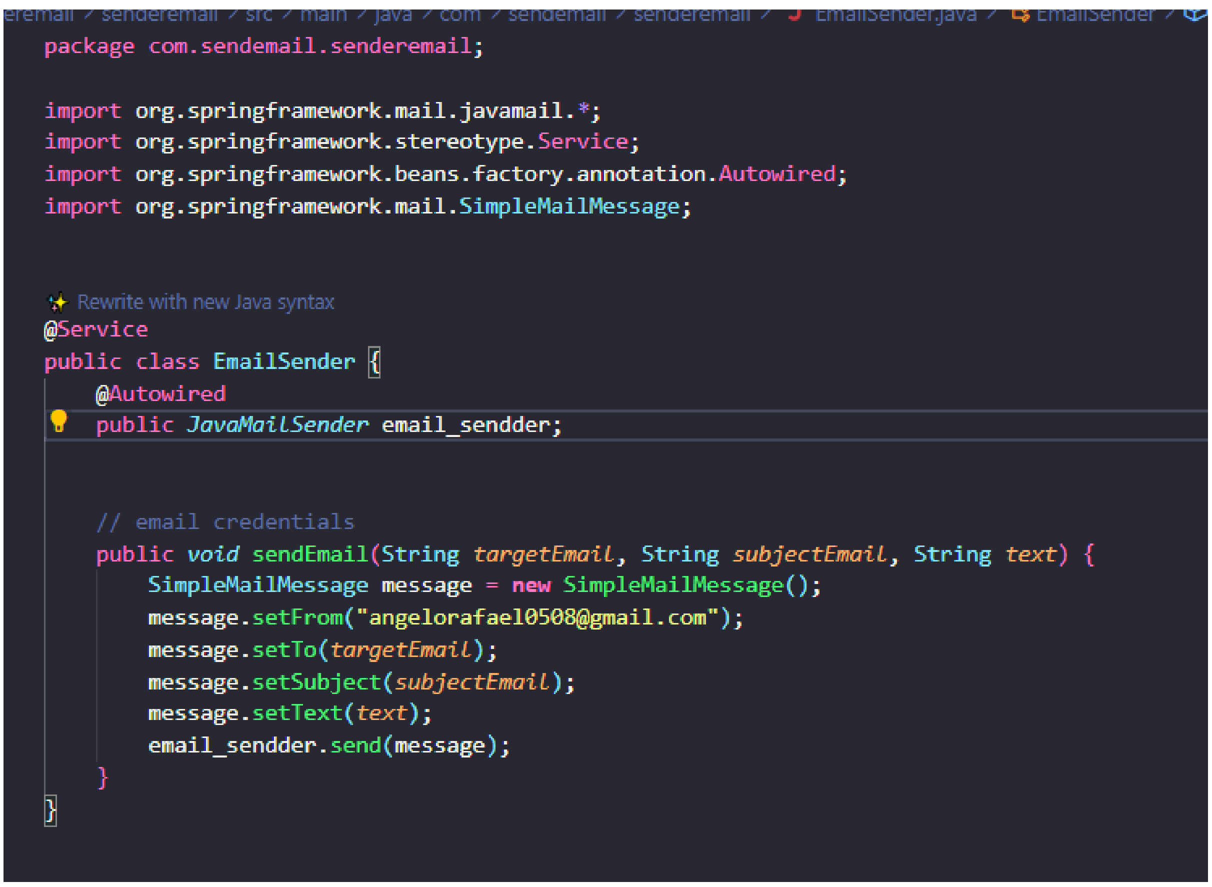
That was for storing the data. But, to send the user the verification code or OTP, I used Spring Initializr and a Maven project. Through that they have provided me with a lot of classes to assist me in creating the program. Plus, not forgetting to mention that they can allow imports from their packages as well. To create the 6 digit code randomly each time I just imported the Random library and used one of their built-in methods, just like how it is shown in the code below



```java
Random random = new Random();
otp = random.nextInt(900000)+1;
```

Upon signing up to the program, the user can't just immediately create an account. Instead, they have to insert their email first, click the verification button, and insert the OTP code in the space provided. If the user decided to create an account before verifying, then a pop-up in the form of a JOptionPane will show up.

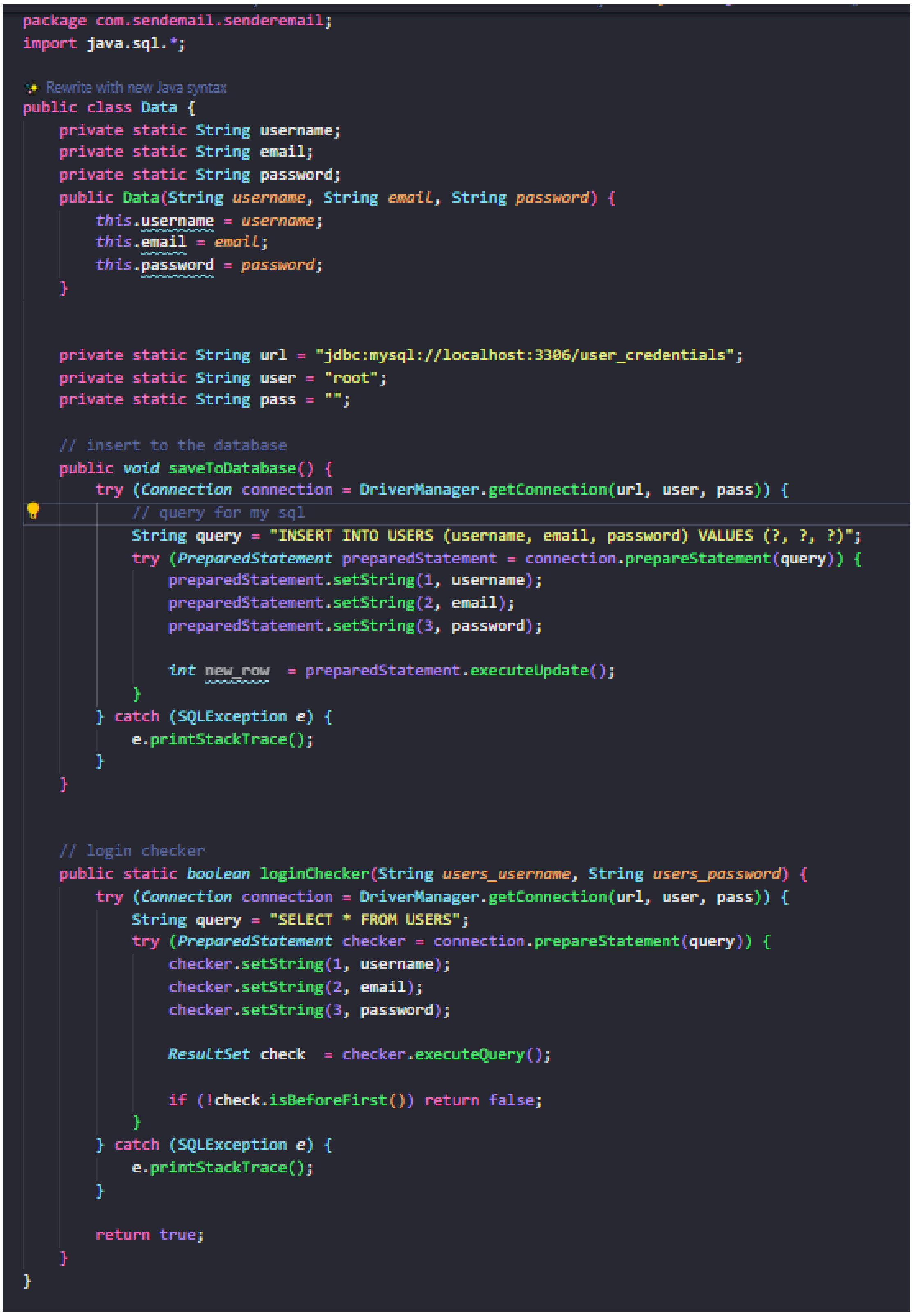
The code that was shown in the EmailSender class, like as can be shown below


eremail › senderemail › src › main › java › com › sendemail › senderemail › EmailSender.java › EmailSender ›
Rewrite with new Java syntax

```java
package com.sendemail.senderemail;

import org.springframework.mail.javamail.*;
import org.springframework.stereotype.Service;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.mail.SimpleMailMessage;


@Service
public class EmailSender {
    @Autowired
    public JavaMailSender email_sendder;

    // email credentials
    public void sendEmail(String targetEmail, String subjectEmail, String text) {
        SimpleMailMessage message = new SimpleMailMessage();
        message.setFrom("angelorafael0508@gmail.com");
        message.setTo(targetEmail);
        message.setSubject(subjectEmail);
        message.setText(text);
        email_sendder.send(message);
    }
}
```

This is the driver specifically for the sending email part, where I used an instance variable of JavaMailSender. Then construct a method with the user's email, subject of email, and the text of the email as the parameters, which will be important in another class. However, the rest of the code in this class is just configurations for the email, and lastly calling the email_sender to send the message. By which the message can be obtained from the SimpleMailMessage class.

Now for the code that can be found in the Data,java class, there are 2 methods that can be seen present. Both of them have different usages, but both are related with the database which is MySQL. For the saveToDatabase() method, it is to save the users' information to the table. It is done by making use of classes like Connection and Prepared Statement, and a query to update the SQL. For the loginChecker() method, it is the only method in the program that returns a value, as others are void. So, it returns a boolean value, if the users' username and password is found. The code snippet can be found below:



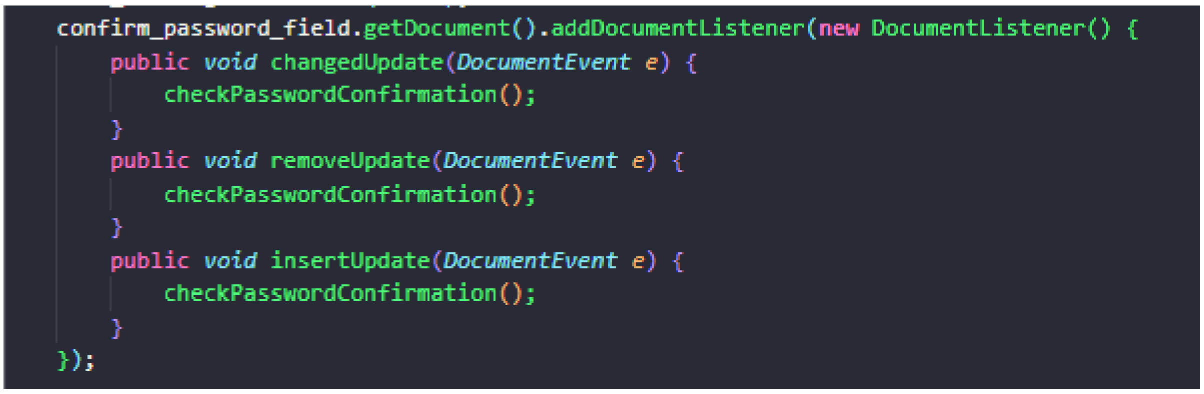
```java
package com.sendemail.senderemail;
import java.sql.*;

public class Data {
    private static String username;
    private static String email;
    private static String password;
    public Data(String username, String email, String password) {
        this.username = username;
        this.email = email;
        this.password = password;
    }


    private static String url = "jdbc:mysql://localhost:3306/user_credentials";
    private static String user = "root";
    private static String pass = "";

    // insert to the database
    public void saveToDatabase() {
        try (Connection connection = DriverManager.getConnection(url, user, pass)) {
            // query for my sql
            String query = "INSERT INTO USERS (username, email, password) VALUES (?, ?, ?)";
            try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {
                preparedStatement.setString(1, username);
                preparedStatement.setString(2, email);
                preparedStatement.setString(3, password);

                int new_row  = preparedStatement.executeUpdate();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }


    // login checker
    public static boolean loginChecker(String users_username, String users_password) {
        try (Connection connection = DriverManager.getConnection(url, user, pass)) {
            String query = "SELECT * FROM USERS";
            try (PreparedStatement checker = connection.prepareStatement(query)) {
                checker.setString(1, username);
                checker.setString(2, email);
                checker.setString(3, password);

                ResultSet check  = checker.executeQuery();

                if (!check.isBeforeFirst()) return false;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }

        return true;
    }
}
```

For the url that will be the local host of my MySQL, and so is the user and password.

For the inheritance and implementation in the code between the class relationships, I just inherited one of the classes to JFrame and both classes that is related with the GUI components implements the ActionListener interface, so that it allows us to write code that is able to detect when the user clicks that triggers an event, one of which is a button. I made use of the ActionEvent as well for the methods that detect the user button click in this program.

For the GUI components, they are pretty simple. I made use of mostly JLabel, JButton, JTextField, JPasswordField, JPanel, JOptionPane. However, it can be seen in the code that I declared the variables first in the class before using them, and that is because some of the other methods need to use it as well to detect events or read user inputs. And if I declared the variable inside of another method, then the ActionEvent methods cannot get access to the variable.

For some of the text fields, particularly the email and password confirmation fields, I used a DocumentListener in order to read the user input as a live event. This is so that the user knows what to include and what are the requirements for the field as quickly as possible, without having to click buttons. The DocumentListener library needs to be imported first, with the need of having 3 methods with 3 different usages.



```java
confirm_password_field.getDocument().addDocumentListener(new DocumentListener() {
    public void changedUpdate(DocumentEvent e) {
        checkPasswordConfirmation();
    }
    public void removeUpdate(DocumentEvent e) {
        checkPasswordConfirmation();
    }
    public void insertUpdate(DocumentEvent e) {
        checkPasswordConfirmation();
    }
});
```

Here is the confirm_password_field DocumentListener as an example, where the 3 methods can be seen with DocumentEvent as their parameters and checkPasswordConfirmation() method which is the one I made to check whether the regular password and confirm password field have equal values.

The 3 methods have self-explanatory names, but here are the uses:

1. changeUpdate = Detecting when changes is made in the field
2. insertUpdate = Detecting if a new insertion is made into the field
3. removeUpdate = Detecting if something is deleted in the field.

The GitHub link with the include of code, poster, and UML diagram is inserted below:
https://github.com/RafaelAngeloChristianto/2-StepVerification