Chat Application (personal project)

Rahul Kayithi

I built this chat server application to practice SQL database integration into a Java program, as well as a client-server interaction using sockets. It allows a new user to set up an account using a username, password, and email address, and these 3 fields are then recorded in a table in an SQL database. I used a Microsoft SQL server for this particular application, but I was also able to successfully test with a MySQL database as well.

In the UserRegistration class, after the user inputs the necessary fields, a confirmation email is sent to the email address they provided, confirming the creation of their new Chatroom account.

The ChatServer class allows them to log into the chatroom using their username and password, which will be checked against the saved information in the database table. They will only be able to log into the chatroom if the username and password match up.

The ChatClient class creates a socket that connects to the ServerSocket in the ChatServer application. After this is executed, the user can then communicate with the server/admin using the chat.

How to set up the application:

The UserRegistration and ChatServer class both require a connection URL to connect to the database server. This is the example URL that must be tweaked based on your system:

```
      private static final String url =
"jdbc:sqlserver://localhost\\SQLEXPRESS:<port
num>;databaseName=<dbname>;user=<username>;password=<password>; encrypt =
true;trustServerCertificate=true";
```

This example is for the Microsoft SQL server. Further information for creating a connection URL and establishing a connection can be found here:
https://learn.microsoft.com/en-us/sql/connect/jdbc/building-the-connection-url?view=sql-server-ver16
https://learn.microsoft.com/en-us/sql/connect/jdbc/working-with-a-connection?view=sql-server-ver16v

```
CREATE DATABASE chatdb2;

USE chatdb2;

CREATE TABLE users (
    id INT PRIMARY KEY IDENTITY,
    username VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,
    email VARCHAR(320) NOT NULL
);
```

For this project, I used Microsoft SQL Server Management Studio to set up the server database and table.
The relevant database “chatdb” can be created from an SQL query by executing this command:

CREATE DATABASE chatdb;

Then, the relevant table “users” can be created by executing this command:

USE chatdb;

CREATE TABLE users (
  id INT PRIMARY KEY IDENTITY,
  username VARCHAR(255) NOT NULL,
  password VARCHAR(255) NOT NULL,
      email VARCHAR(320) NOT NULL
);

At this point, the database is fully built for the project. You must also download a JDBC driver and add the JAR file to the Java project class path. The JAR file for the Microsoft JDBC driver can be downloaded from this link:
https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16

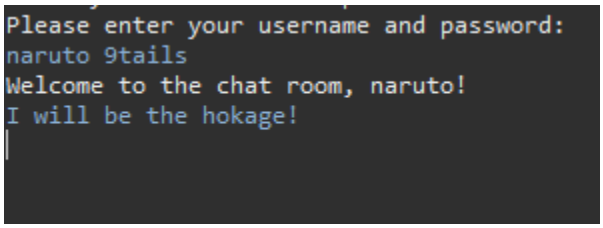
In addition to the JDBC driver, 2 more JAR files must be imported to the class path libraries; javax.mail.jar and activation.jar
These JAR files are necessary for the email functionality of the UserRegistration class which sends a confirmation email to the user after their account creation. Those can be found at these links:
https://javaee.github.io/javamail/
https://www.oracle.com/java/technologies/java-beans-activation.html

Next, we must input the email information for the account which will send out the confirmation email.

```
private static final String EMAIL_FROM = "email@example.com";  //input your email address
private static final String EMAIL_PASSWORD = "password";       //input your email password
private static final String EMAIL_HOST = "smtp.email.com";     //input host name
private static final String EMAIL_PORT = "465";                // input email port number
```

The hostname is unique to each email provider. The hostname and the port number can easily be found by googling. This link is a reference for the Gmail SMTP: https://kinsta.com/blog/gmail-smtp-server/

With that, the Chat application should be fully functional on your end. Open your SQL server and create the database and table. Then run the UserRegistration class to create an account which will be saved to the database. After, run the ChatServer class to start the server, and finally, run the ChatClient class to complete the connection. After logging into a saved account, you will be able to type in the chatroom. Here is some sample output:



Thanks for checking out my project!