ZHEJIANG UNIVERSITY

Undergraduate Report

A WeChat Multi Thread Chat Application

Course	Application of Java Programming
Academic Advisor	Professor Weiming Lu
Teaching Assistants	
Name	Leming Shen
ID Number	3180103654
College	College of Computer Science and Technology
Major	Software Engineering

December 31, 2020

Contents

1 Introduction to the Project

1.1 Background

In the course Java Application Technology, we are asked to write a program that implements a simple chat application. We need to use socket to implement communication between any two users. Java Swing library should be used to implement a simple chat interface.

Besides all this, the user's personal information and chat records are all stored in a database through the connection between Java and MySQL.

1.2 Goal of Design

1. A simple user interface to:
    - Sign in & Sign up (Using Email address and verify through email) to the system.
    - View the friend list.
    - Chat with friends with multiple windows.
2. A C/S socket model is developed to implement communication between users.
3. Link the program with the local database to:
    - Record the user's personal information
    - Record the user's chat message

1.3 Development & Running Configuration Environment

1.3.1 Development Environment

1. Operating System: Linux Ubuntu 20.04 LTS
2. Java version "1.8.0_251"
3. Java(TM) SE Runtime Environment (build 1.8.0_251-b08)
4. Java HotSpot(TM) Client VM (build 25.251-b08, mixed mode, sharing)
5. JDK 15.0
6. IDE: Jetbrains IntelliJ IDEA 2020.2.3
7. Chrome Driver
8. Third-party Jar
    - activation.jar
    - javax.mail.jar
    - mail.jar
    - mysql-connector-java-8.0.16.jar

1.3.2 Running Configuration Environment

1. Make sure that you have already correctly installed MySQL and the MySQL-Server is normally running.
2. Make sure that you have already created a database named "wechat" and create all tables provided below:
    - User.sql

```
create table user
(
    user_id varchar(255) not null,
```

```sql
    password varchar(20) not null,
    user_name varchar(255) not null,
    name varchar(255),
    phone varchar(20),

    primary key ( user_id )
);
```

- Friend_pair.sql

```sql
create table friend_pair
(
    user1_id varchar(255) not null,
    user2_id varchar(255) not null,

    primary key ( user1_id, user2_id )
);
```

- Chat_record_single.sql

```sql
create table chat_record_single
(
    id int not null auto_increment,
    send_user_id varchar(255),
    receive_user_id varchar(255),
    chat_time datetime not null,
    content text,

    primary key ( id )
);
```

3. The user’s name and password to the database should be "slm" and "123456". And if you want to connect to your local database or remote database, please modify concerned parameters in Database.java.
4. Open the whole folder with IntelliJ IDEA

2 Overall Design

2.1 Project Structure

|-wechat

|----.idea

|----img

    |----background.jpg

    |----profile

        |----3180103654@zju.edu.cn.jpg

        |----zjuslm@163.com.jpg

        |----zjuslm@126.com.jpg

```
|----lib
        |----activation.jar
        |----javax.mail.jar
        |----mail.jar
        |----mysql-connector-java-8.0.16.jar
|----out
|----src
        |----communication
                |----Client.java
                |----Client_Reader.java
                |----Server.java
                |----Server_Handler.java
        |----GUI
                |----Chat_Window.java
                |----Friend_List.java
                |----Login.java
                |----Register.java
                |----Session_Client.java
                |----Session_Server.java
        |----jdbc
                |----Database.java
                |----SendEmail.java
        |----Main.java (run the main program)
|----wechat.imi
```

2.2 Class Design

1. Communication
    1) Client
       The client class can generate a thread that individually handles the communication between the client and the server.
    2) Client_Reader

The client_reader class mainly deals with the message that the current client receives. After receive the message, the class decrypts it and post it on the user interface to let the user see the message he/she receives.

3) Server

   The server class generates a thread that handles the information the server gets.

4) Server_Handler

   The server_handler class mainly deals with the encrypted message from client and decides who to re-send.

2. GUI

   1) Chat_Window

      The Chat_Window class mainly provides a comfortable user interface of chatting.

   2) Friend_List

      The Friend_List class mainly provides a comfortable user interface that presents all the user's friend.

   3) Login

      The Login class mainly provides a user interface for user to input user id and password to login, and can generate a new window for user to create a new account.

   4) Register

      The Register class mainly provides a user interface for user to input his/her email, password, user name, real name, phone to create a new account. Before signing up, the inputted email address will be verified through the authentication code in the email sent by the program.

   5) Session_Client

      Every time the user clicks a friend in the friend list, a Chat_Window class and a Session_Client class will be created that generate a client thread.

   6) Session_Server

      Every time the program is started, the session_server will be created.

3. JDBC

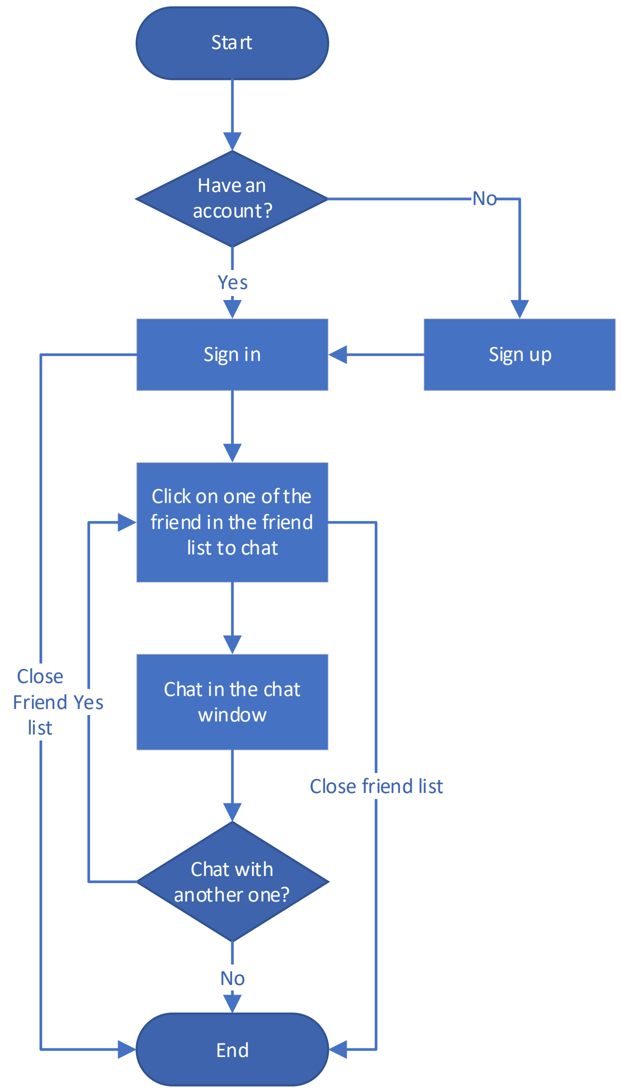
   1) Database

      The Database class provides a well encapsulated interface to implement insertion, deletion, query, modification on the Database according to the inputted SQL statements.

   2) SendEmail

      The SendEmail class provides a static method that send a email with a random authentication code according to the given email address.

2.3 Flowchart



3 Detailed Design

3.1 Communication

3.1.1 Client.java

```java
public class Client
{
    private int port = 9000;             /* the port to listen */
    private String ip = "127.0.0.1";     /* the ip address */
    private static Socket socket;        /* the communication socket */
    private String client_id;          /* set the client thread's id as client_id */
    private final int BUFFER_LENGTH = 8192; /* the maximum length of message */
    private JPanel chat_record_area;/* the chat record area from the chat window */

    /* the constructor function helps to initialize the client */
    public Client(String client_id, JPanel chat_record_area);

    /* initialization */
    private void init(String client_id, JPanel chat_record_area);

    /* create a new thread of client_reader according to the client_id */
    public void handle(JPanel chat_record_area);

    /* send a single message to the server, with declaration of target id */
    public void send(String message, String target_id);

    /* end the socket */
    public boolean cancel()
}
```

3.1.2 Client_Reader.java

```java
public class Client_Reader implements Runnable
{
    private InputStream is; /* the input stream of the socket */
    private final int BUFFER_LENGTH = 8192; /* the maximum length of message */
    private JPanel chat_record_area;    /* the chat record area */
    /* vaerify the send user id and receive user id */
    /* according to the encryptted message */
    private String user_id, user_id_source;

    /* the constructor to initialize */
    public Client_Reader(InputStream is, JPanel chat_record_area, String user_id);

    /* override the runnable interface's run method */
    /* which display the message the client receives */
    @Override
    public void run();
}
```

3.1.3 Server.java

```java
public class Server
{
    private int port = 9000;    /* the listening port */
    private ServerSocket server;/* the server socket */
    private Socket socket;      /* the captured client socket */
    private String server_name; /* the server name */
    private final int BUFFER_LENGTH = 8192;

    /* a hash map that stores the list of clients */
    private HashMap<String, Socket> client_list = new HashMap<String, Socket>();

    /* initialize the server */
    public Server(String server_name);

    /* initialize the server */
    private void init(String server_name);

    /* create a new server handler thread */
    /* that deals with the listened socket */
    private void handle(Socket socket);

    /* return the current client list */
    public HashMap<String, Socket> getClient_list();
}
```

3.1.4 Server_Handler.java

```java
public class Server_Handler implements Runnable
{
    private Socket socket;      /* the listened client socket */
    private OutputStream os;    /* the output stream of socket */
    private InputStream is;     /* the input stream of socket */
    private final int BUFFER_LENGTH = 8192;
    private HashMap<String, Socket> client_list;

    /* initialize the private variables */
    public Server_Handler(Socket socket, HashMap<String, Socket> client_list)

    /* the override run() function mainly deals with */
    /* the message the server receives. */
    /* it can recoginize the send user id, receive user id */
    /* and the message by decryptting the raw byte stream */
    /* and then decide who to send the message */
    @Override
    public void run()
}
```

3.2 GUI

3.2.1 Chat_Window.java

```java
public class Chat_Window
{
    private Database sql;   /* the jdbc interface */
    private String user_id; /* the current user's id */
    private String target_user_id;  /* the send target user id */
    private JFrame frame = new JFrame();    /* the window frame */

    /* initialize the window, search in the database */
    /* to get all chat history between the current user and target user */
    Chat_Window(String user_id, String target_user_id, Database sql);

    /* display the chat record area and send message area */
    /* every time the chat window is created, a new thread of client session */
    /* will be created that deals with sending and receiving messages */
    public void init(String user_name, String target_user_name);
}
```

3.2.2 Friend_List.java

```java
public class Friend_List
{
     private JFrame frame = new JFrame("Friend List");
    private Database sql;
    private String user_id; /* the current user id */
    private int friend_number = 0;  /* the number of friends the user has */
    private ArrayList<String> friends = new ArrayList<>();  /* friend list */

    /* initialize the frame */
    Friend_List(String user_id, Database sql);

    /* search from the database and display the user's information */
    public void set_personal_information();

    /* search from the database and display a list of friends' info */
    /* if the user click on one the list item, a new chat window will be created */
    public void init_friend();
}
```

3.2.3 Login.java

```java
public class Login
{
    private JFrame frame = new JFrame("Login");
    private Database sql;

    /* initialize */
    public Login(Database sql);

    /* set the background picture of the fram */
    public void set_background();

    /* dispaly user id and password field */
    public void set_login_panel();

    /* search in the database to authenticate */
    /* the inputted user id and password */
    public void handle_login(String user_id, String password)
}
```

3.2.4 Register.java

```java
public class Register
{
    private JFrame frame = new JFrame("Register");
    private Database sql;
    private int code;

    /* initialize */
    Register(Database sql);

    /* display the register information field */
    /* send an authentication email and register */
    public void set_register_panel();
}
```

3.2.5 Session_Client.java

```java
public class Session_Client implements Runnable
{
    private int length;     /* length of friends */
    private String user_id;
    private String user_name;
    private JButton send_button;
    private JTextArea send_message_area;
    private String target_user_id;
    private JPanel chat_record_area;

    /* initialize */
    Session_Client(
        String user_name,
        String user_id,
        String target_user_id,
        JPanel chat_record_area,
        JTextArea send_message_area,
        JButton send_button,
        int length);

    /* handle the received message */
    @Override
    public void run();
}
```

3.2.6 Session_Server.java

```
public class Session_Server implements Runnable
{
    /* Run a server session */
    @Override
    public void run()
    {
        Server server = new Server("leming");
    }
}
```

3.3 JDBC

3.3.1 Database.java

```
public class Database
{
    /* the configuration parameters */
    static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/wechat?useSSL=false&a
llowPublicKeyRetrieval=true&serverTimezone=UTC";

    static final String USER = "slm";
    static final String PASS = "123456";

    private Connection connection = null;

    /* initialize */
    public Database();

    /* execute insert, update, delete in database */
    public void modify(String sql, String[] args, String[] types);

    /* execute select in database */
    public ResultSet query(String sql, String[] args, String[] types);
}
```

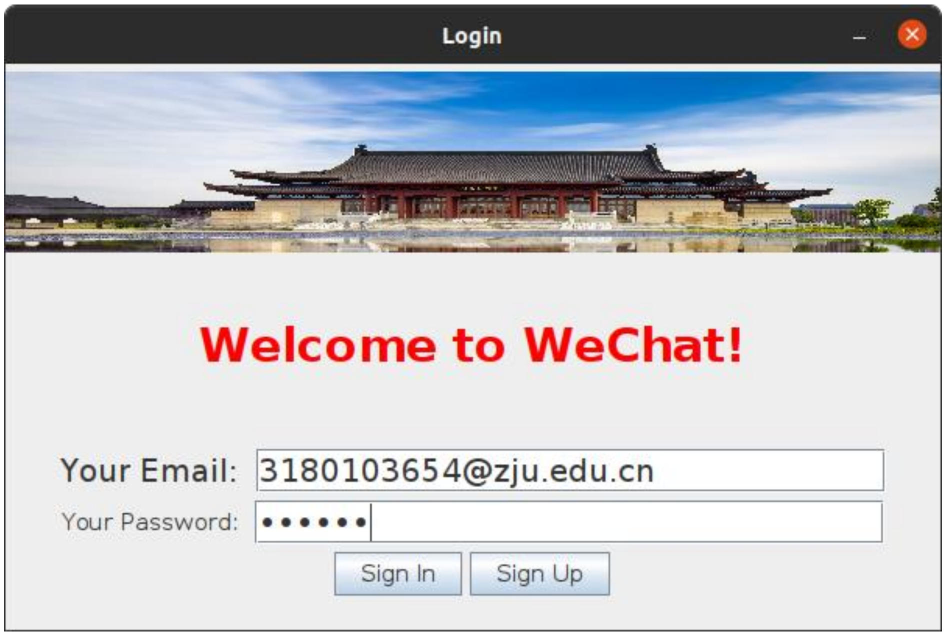
3.3.2 SendEmail.java

```java
public class SendEmail
{
    /* send email to the recipient's email address */
    public static void send(String recipient, String code);
}
```

4 Testing & Running

4.1 Login

The login window:

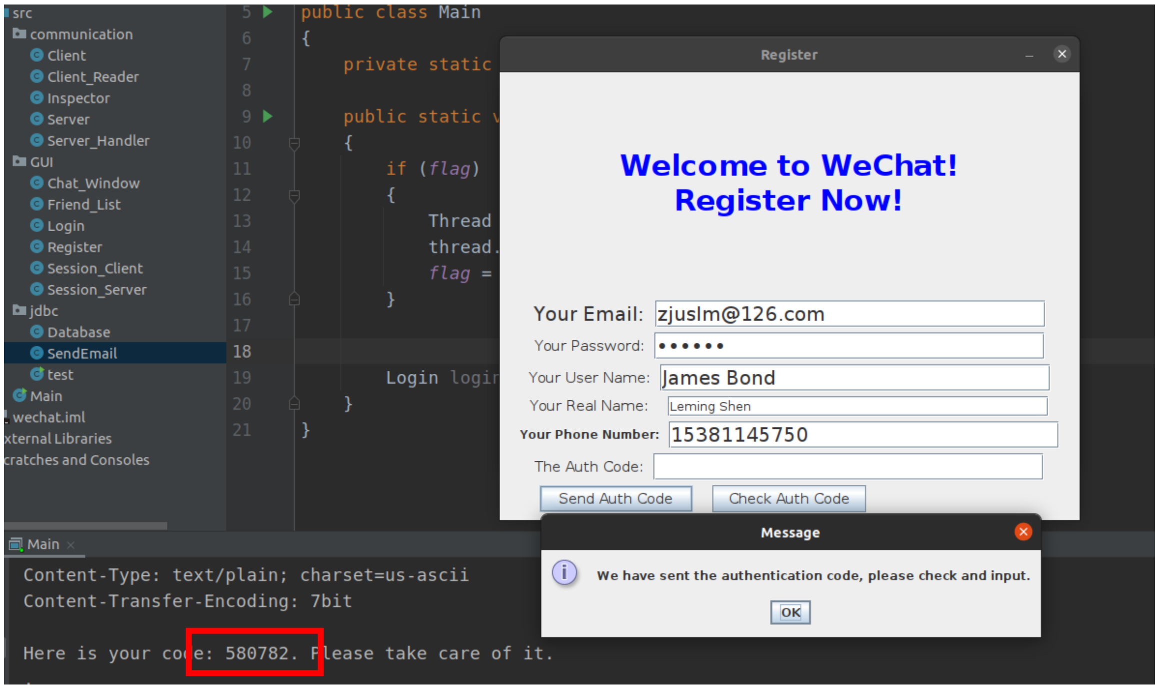


Your Email: 3180103654@zju.edu.cn

Your Password: ●●●●●●

4.2 Register

The register window and sending an email to verify:



```
public class Main
{
    private static

    public static v
    {
        if (flag)
        {
            Thread
            thread.
            flag =
        }

        Login logir
    }
}
```

Welcome to WeChat!
Register Now!

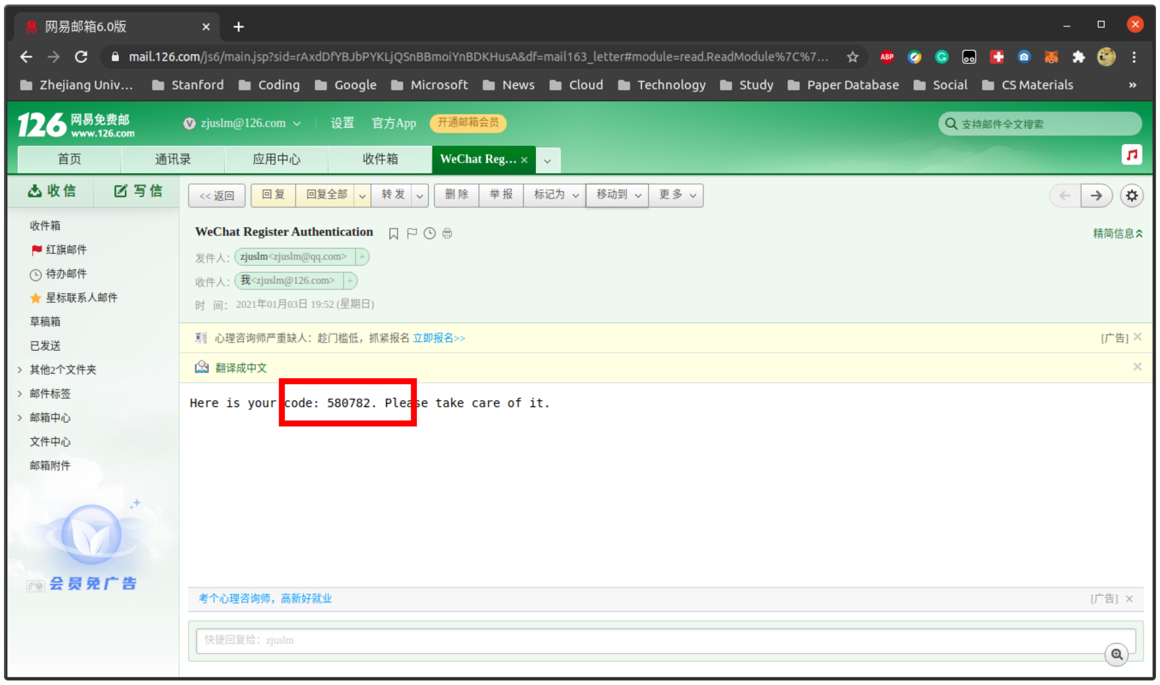
Your Email: zjuslm@126.com
Your Password: ••••••
Your User Name: James Bond
Your Real Name: Leming Shen
Your Phone Number: 15381145750
The Auth Code:

We have sent the authentication code, please check and input.

```
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Here is your code: 580782. Please take care of it.
```

The email received the email:



WeChat Register Authentication

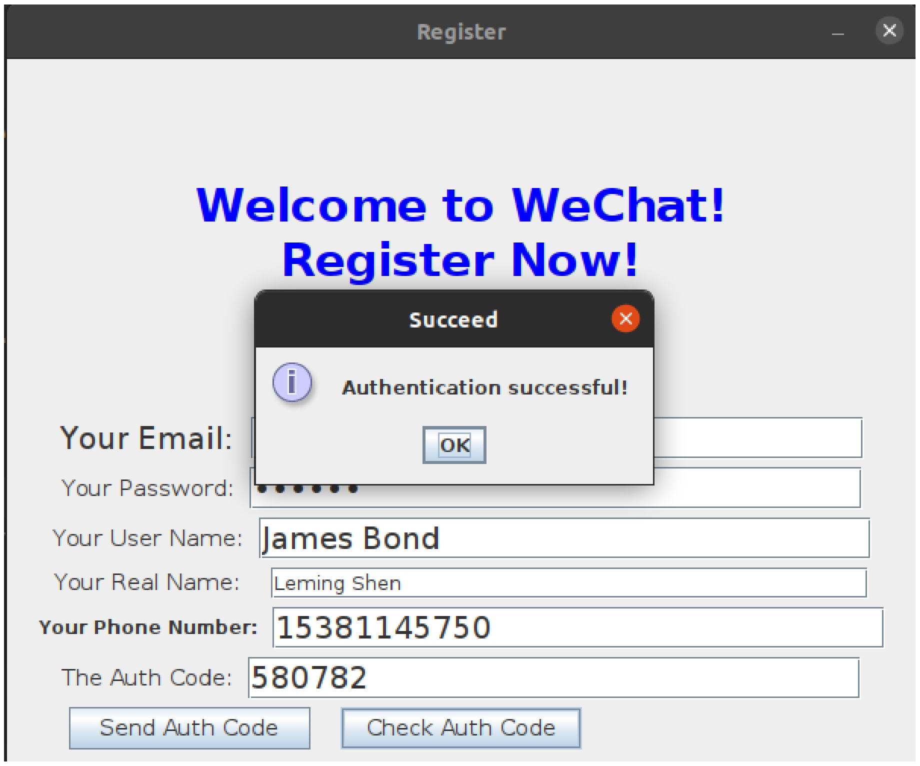
发件人：zjuslm<zjuslm@qq.com>
收件人：我<zjuslm@126.com>
时　间：2021年01月03日 19:52 (星期日)

心理咨询师严重缺人：趁门槛低，抓紧报名 立即报名>> [广告]

```
Here is your code: 580782. Please take care of it.
```

考个心理咨询师，高薪好就业 [广告]

Input the code and verify:



Register

Welcome to WeChat!
Register Now!

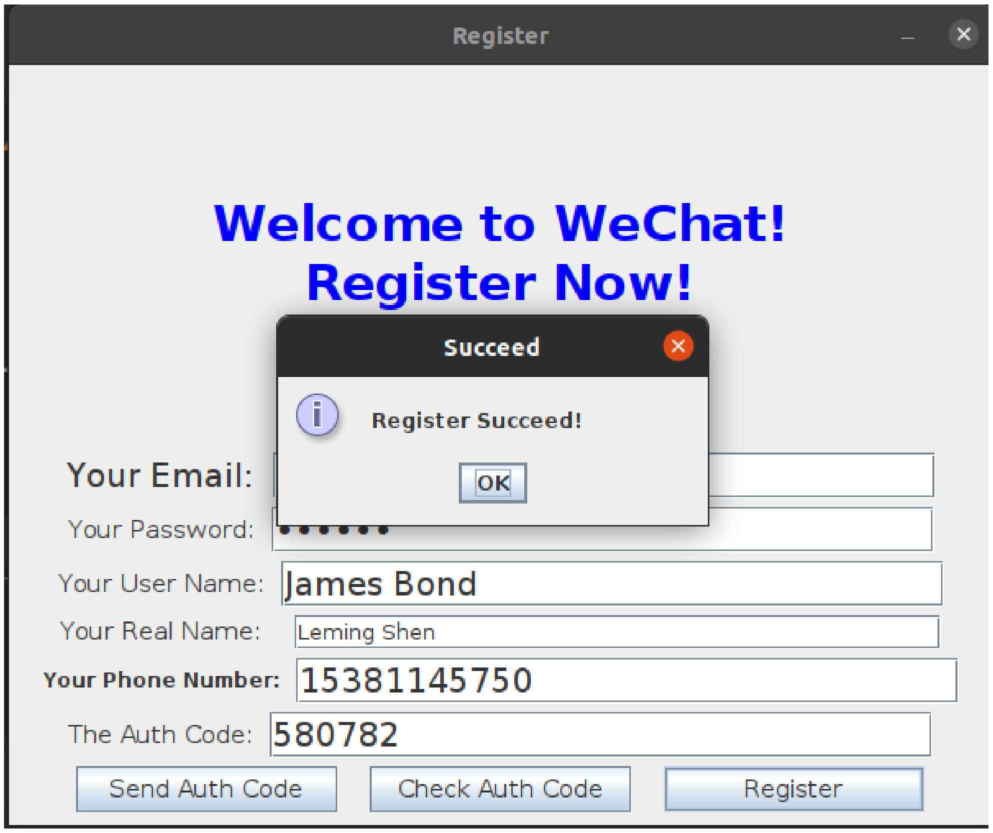
Succeed

Authentication successful!

OK

Your Email:	
Your Password:	••••••
Your User Name:	James Bond
Your Real Name:	Leming Shen
Your Phone Number:	15381145750
The Auth Code:	580782

Register:



Register

Welcome to WeChat!
Register Now!

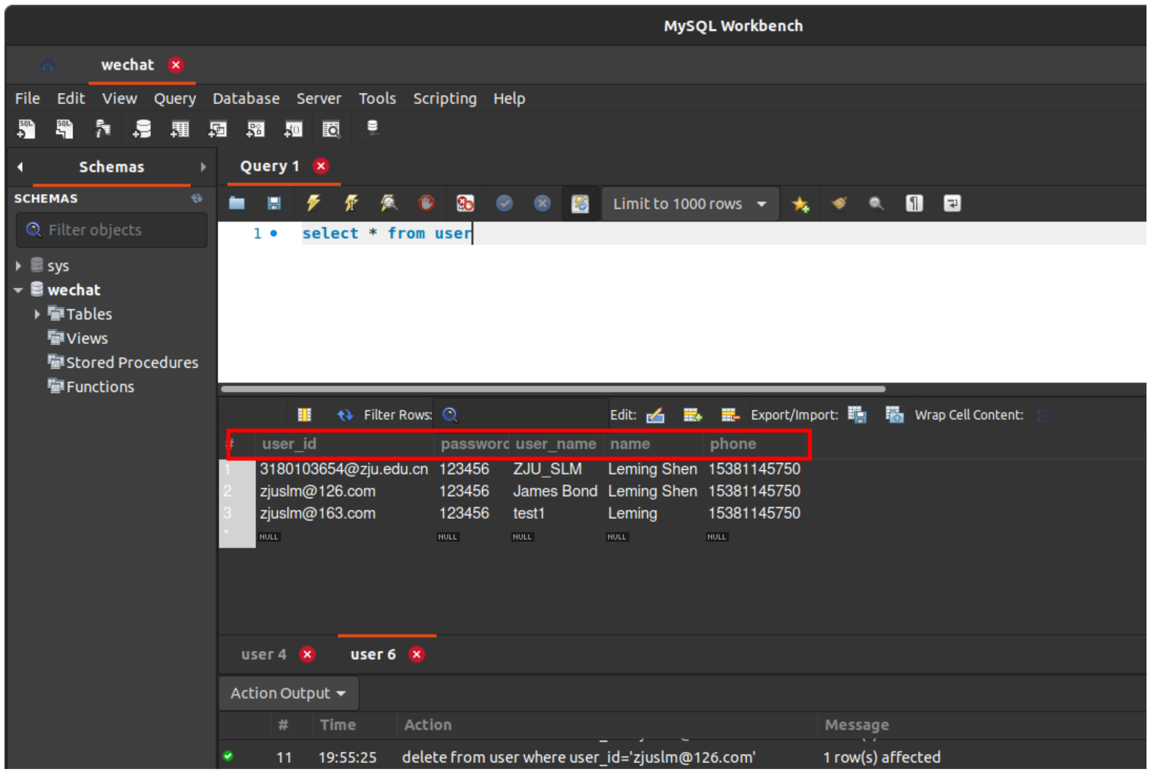
Succeed

Register Succeed!

OK

Your Email:	
Your Password:	••••••
Your User Name:	James Bond
Your Real Name:	Leming Shen
Your Phone Number:	15381145750
The Auth Code:	580782

And the database has a new user:



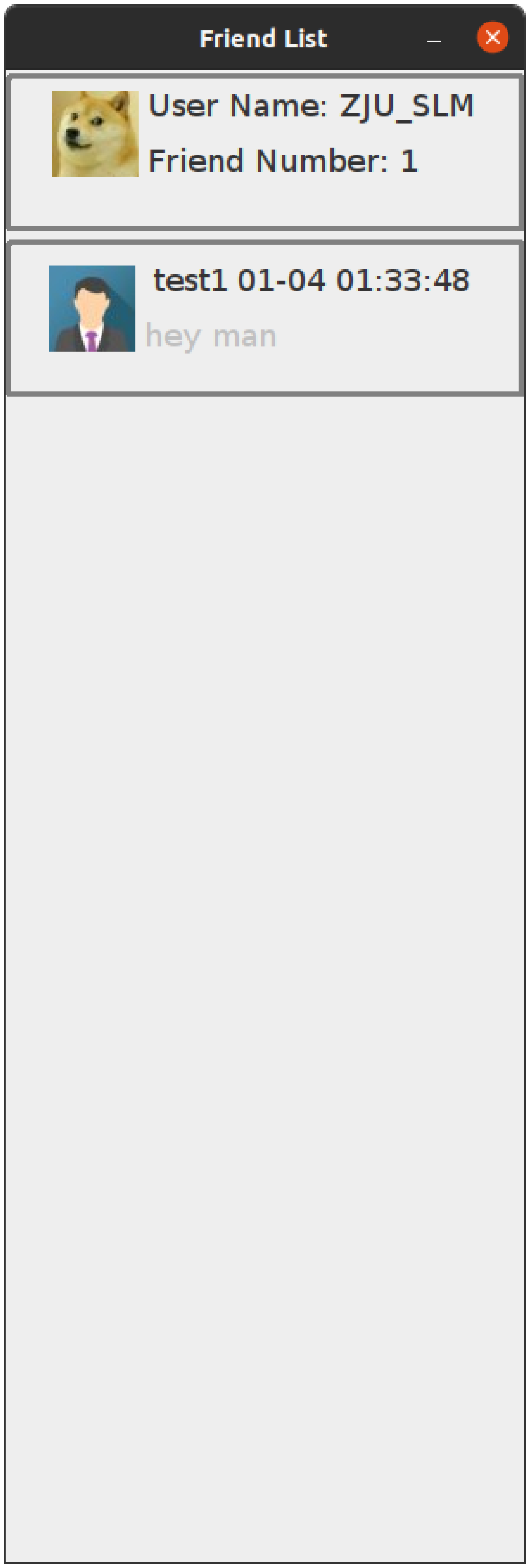
```
select * from user
```

#	user_id	password	user_name	name	phone
1	3180103654@zju.edu.cn	123456	ZJU_SLM	Leming Shen	15381145750
2	zjuslm@126.com	123456	James Bond	Leming Shen	15381145750
3	zjuslm@163.com	123456	test1	Leming	15381145750
*	NULL	NULL	NULL	NULL	NULL

#	Time	Action	Message
11	19:55:25	delete from user where user_id='zjuslm@126.com'	1 row(s) affected

4.3 Friend List

Friend List:



Friend List

User Name: ZJU_SLM

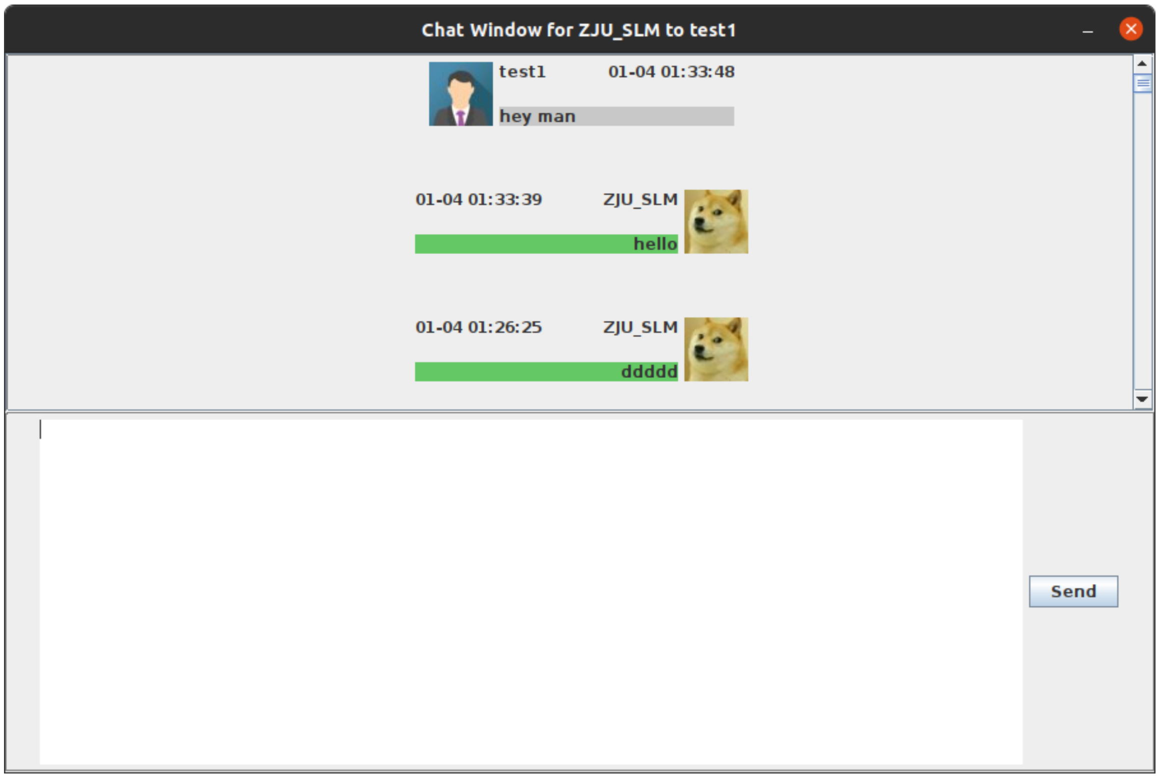
Friend Number: 1

test1 01-04 01:33:48

hey man

4.4 Chat Window

A simple chat window:


Chat Window for ZJU_SLM to test1

test1 01-04 01:33:48

hey man

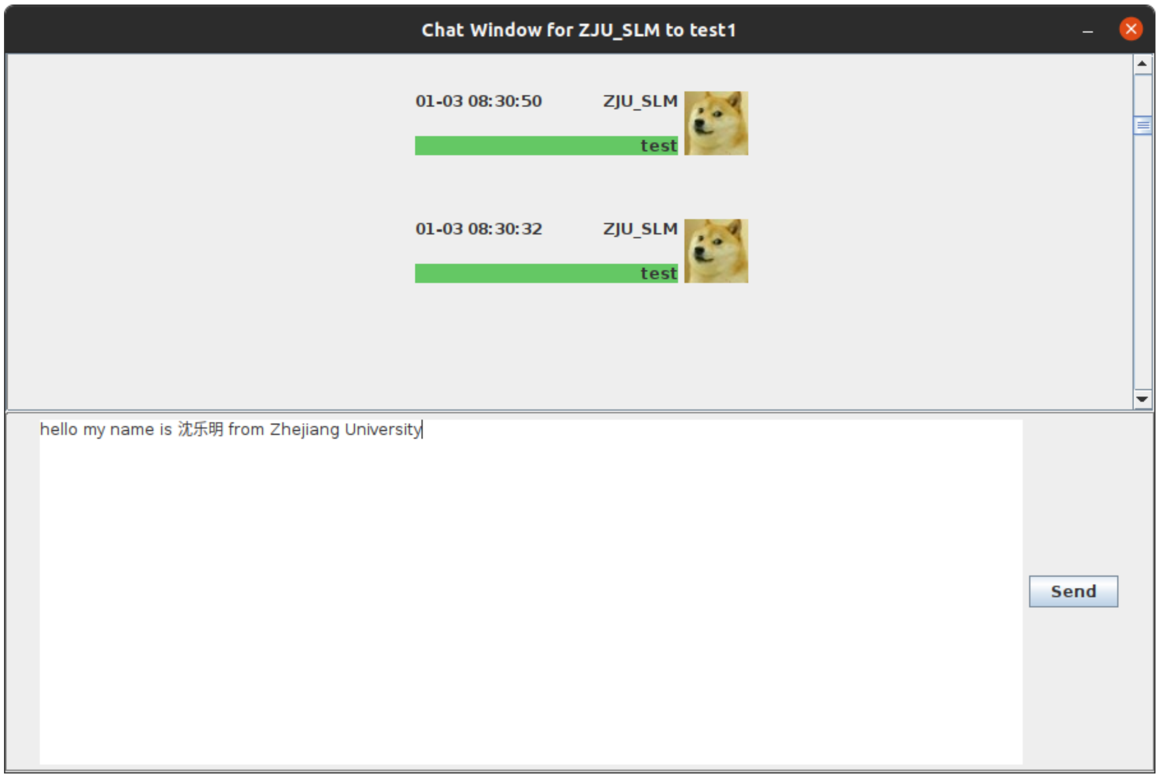
01-04 01:33:39 ZJU_SLM

hello

01-04 01:26:25 ZJU_SLM

ddddd

Input a message


Chat Window for ZJU_SLM to test1

01-03 08:30:50 ZJU_SLM

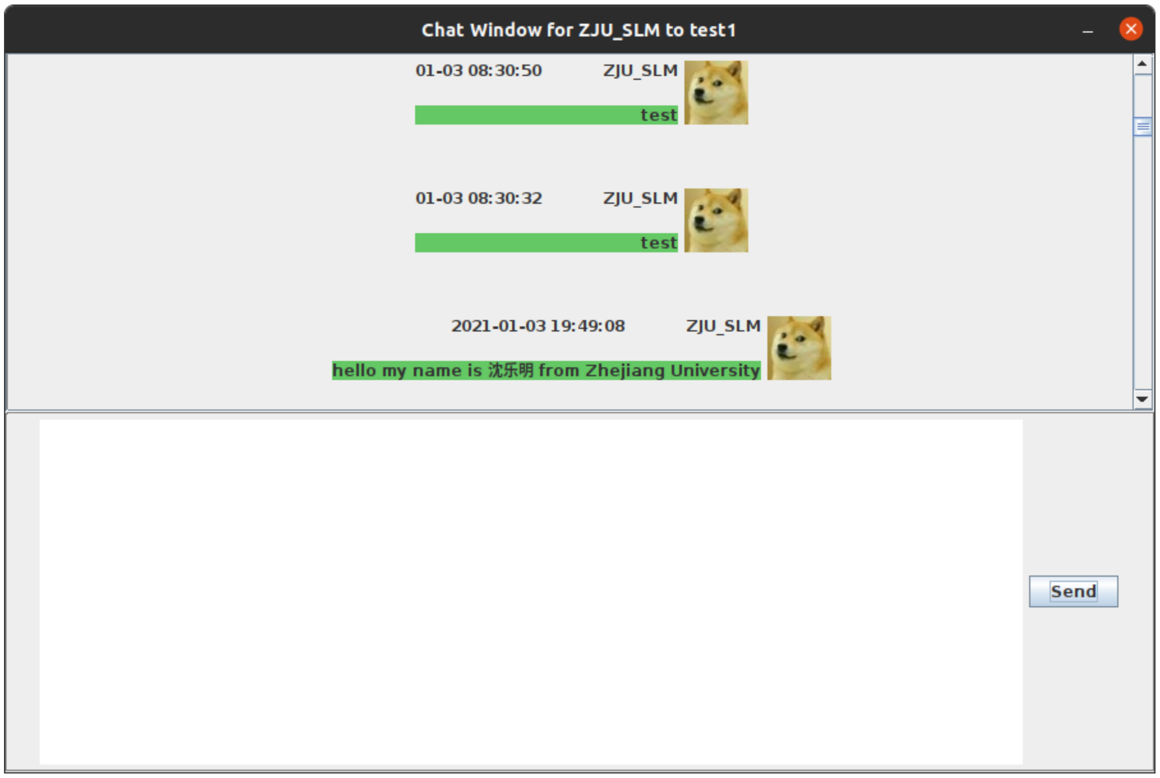
test

01-03 08:30:32 ZJU_SLM

test

hello my name is 沈乐明 from Zhejiang University

Send the message:


Chat Window for ZJU_SLM to test1

01-03 08:30:50 ZJU_SLM

test

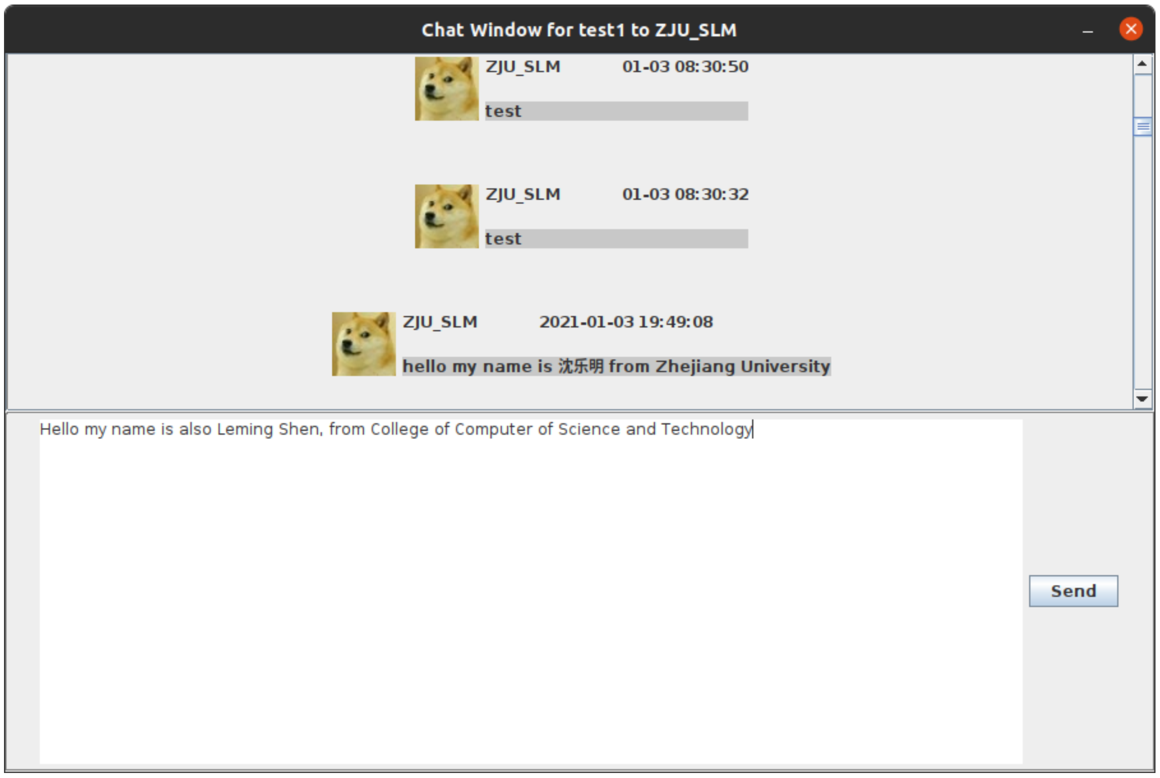
01-03 08:30:32 ZJU_SLM

test

2021-01-03 19:49:08 ZJU_SLM

hello my name is 沈乐明 from Zhejiang University

The other user receives the message:


Chat Window for test1 to ZJU_SLM

ZJU_SLM 01-03 08:30:50

test

ZJU_SLM 01-03 08:30:32

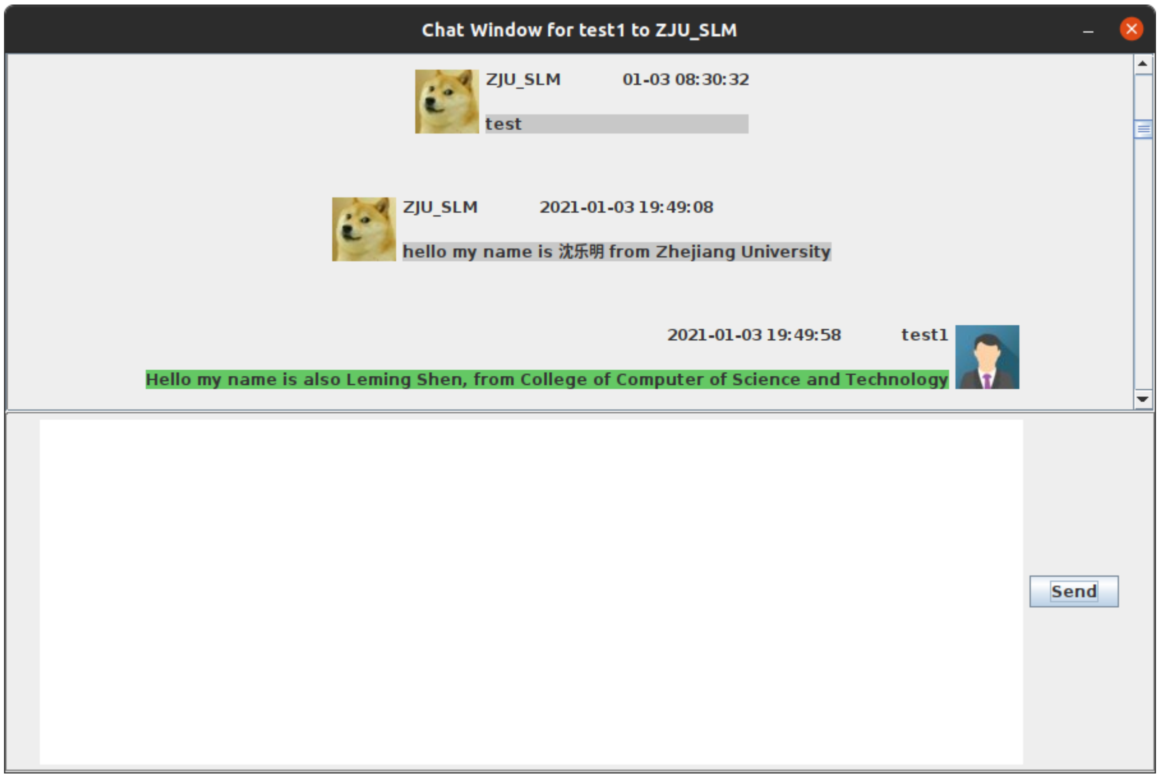
test

ZJU_SLM 2021-01-03 19:49:08

hello my name is 沈乐明 from Zhejiang University

Hello my name is also Leming Shen, from College of Computer of Science and Technology

The other user send back a message:



Chat Window for test1 to ZJU_SLM

ZJU_SLM 01-03 08:30:32

test

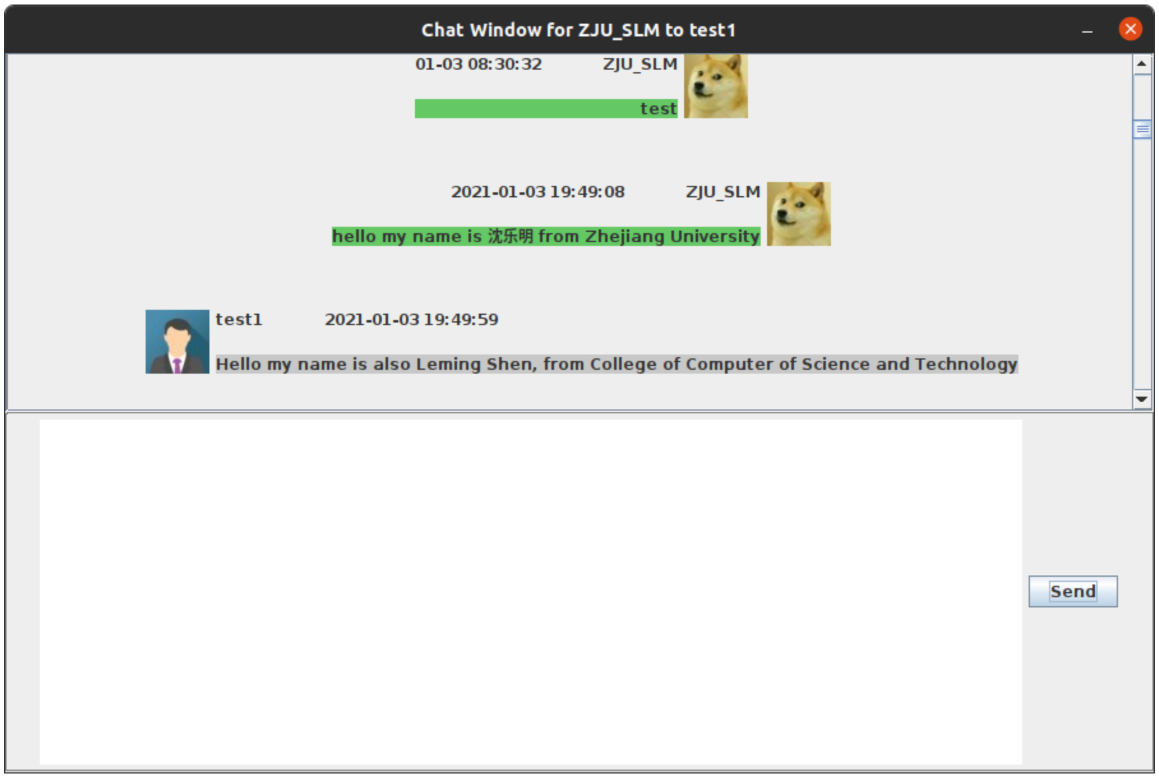
ZJU_SLM 2021-01-03 19:49:08

hello my name is 沈乐明 from Zhejiang University

2021-01-03 19:49:58 test1

Hello my name is also Leming Shen, from College of Computer of Science and Technology

And the first user receives the message:



Chat Window for ZJU_SLM to test1

01-03 08:30:32 ZJU_SLM

test

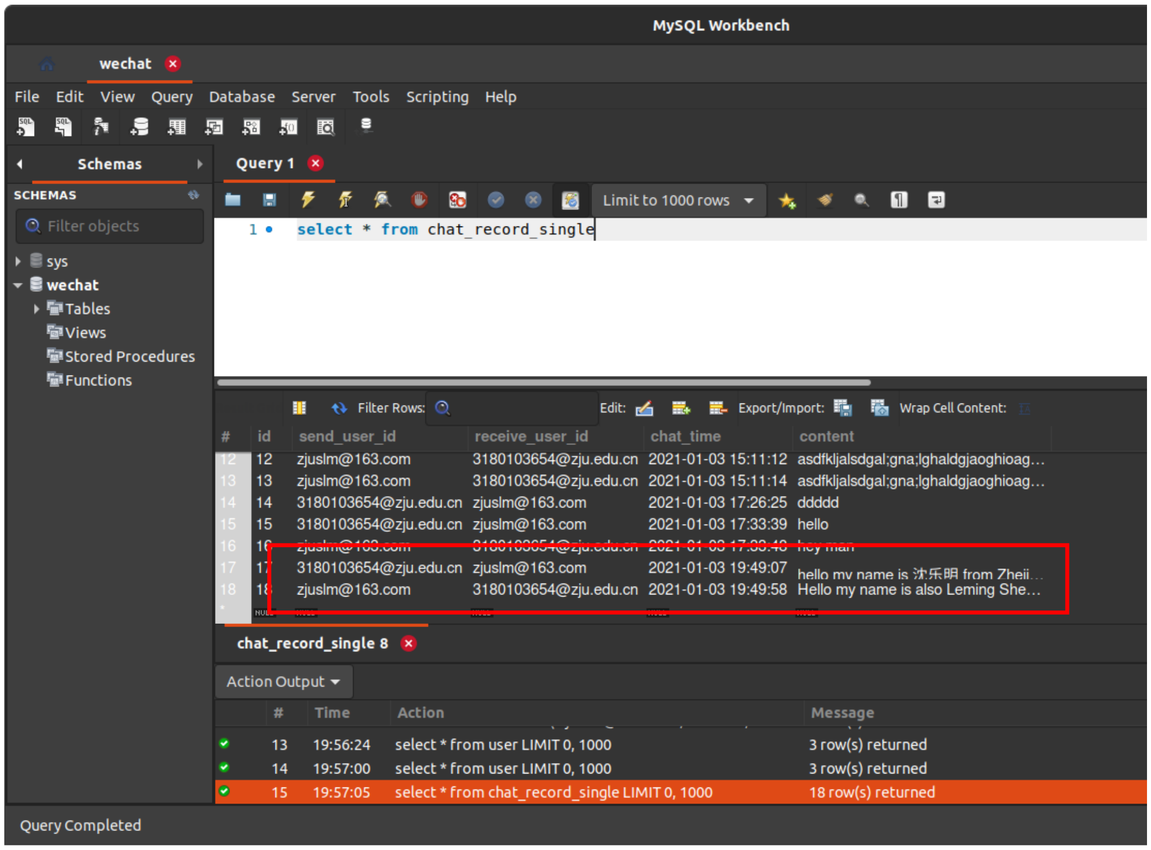
2021-01-03 19:49:08 ZJU_SLM

hello my name is 沈乐明 from Zhejiang University

test1 2021-01-03 19:49:59

Hello my name is also Leming Shen, from College of Computer of Science and Technology

The chat records are all inserted into the database table:

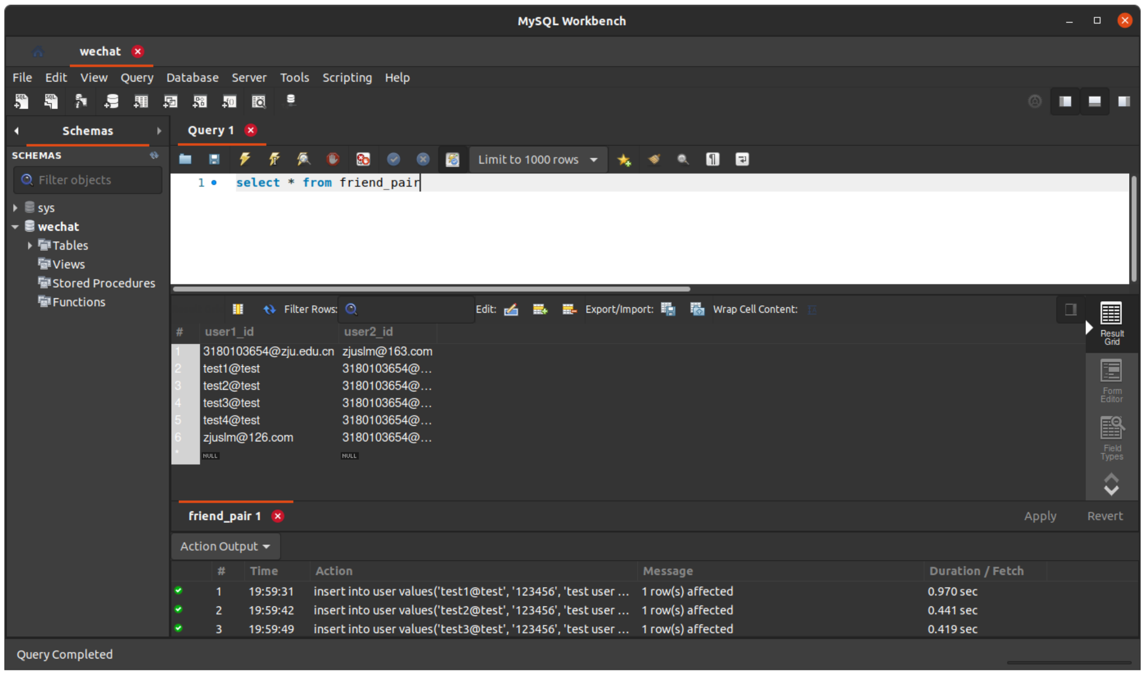


```
select * from chat_record_single
```

#	id	send_user_id	receive_user_id	chat_time	content
12	12	zjuslm@163.com	3180103654@zju.edu.cn	2021-01-03 15:11:12	asdfkljalsdgal;gna;lghaldgjaoghioag…
13	13	zjuslm@163.com	3180103654@zju.edu.cn	2021-01-03 15:11:14	asdfkljalsdgal;gna;lghaldgjaoghioag…
14	14	3180103654@zju.edu.cn	zjuslm@163.com	2021-01-03 17:26:25	ddddd
15	15	3180103654@zju.edu.cn	zjuslm@163.com	2021-01-03 17:33:39	hello
16	16	zjuslm@163.com	3180103654@zju.edu.cn	2021-01-03 17:33:48	hey man
17	17	3180103654@zju.edu.cn	zjuslm@163.com	2021-01-03 19:49:07	hello my name is 沈乐明 from Zheji…
18	18	zjuslm@163.com	3180103654@zju.edu.cn	2021-01-03 19:49:58	Hello my name is also Leming She…
*	NULL	NULL	NULL	NULL	NULL

	#	Time	Action	Message
✓	13	19:56:24	select * from user LIMIT 0, 1000	3 row(s) returned
✓	14	19:57:00	select * from user LIMIT 0, 1000	3 row(s) returned
✓	15	19:57:05	select * from chat_record_single LIMIT 0, 1000	18 row(s) returned

Add more friends to the user:

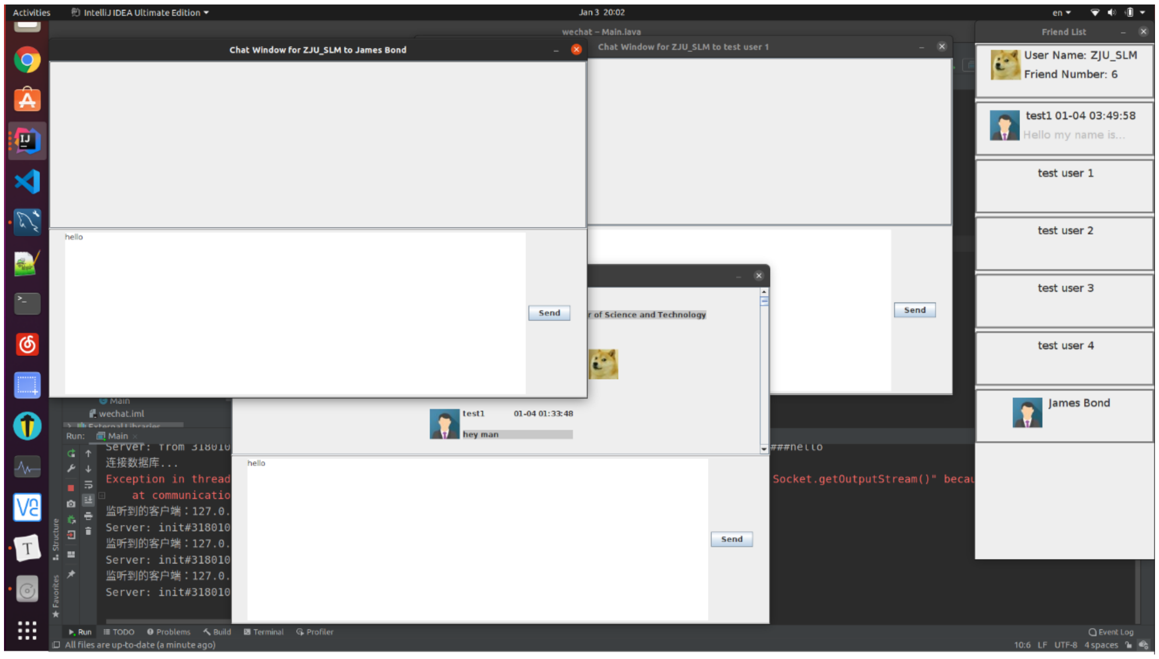


```
select * from friend_pair
```

#	user1_id	user2_id
1	3180103654@zju.edu.cn	zjuslm@163.com
2	test1@test	3180103654@…
3	test2@test	3180103654@…
4	test3@test	3180103654@…
5	test4@test	3180103654@…
6	zjuslm@126.com	3180103654@…
*	NULL	NULL

	#	Time	Action	Message	Duration / Fetch
✓	1	19:59:31	insert into user values('test1@test', '123456', 'test user …	1 row(s) affected	0.970 sec
✓	2	19:59:42	insert into user values('test2@test', '123456', 'test user …	1 row(s) affected	0.441 sec
✓	3	19:59:49	insert into user values('test3@test', '123456', 'test user …	1 row(s) affected	0.419 sec

And then multiple window chat can be implemented:



User Name: ZJU_SLM
Friend Number: 6

test1 01-04 03:49:58
Hello my name is...

test user 1

test user 2

test user 3

test user 4

James Bond

hello

r of Science and Technology

test1 01-04 01:33:48
hey man

hello

Main
wechat.iml
External Libraries

```
Server: from 318010
连接数据库...
Exception in thread
    at communicatio
监听到的客户端：127.0.
Server: init#318010
监听到的客户端：127.0.
Server: init#318010
监听到的客户端：127.0.
Server: init#318010
```

```
###hello

Socket.getOutputStream()" becau
```

All files are up-to-date (a minute ago)

5 Summary

The socket and database communication are easy to program.

However, I got really really bad impression on Java Swing, especially for its layout rule, which nearly drives me mad! 😫

6 References

1. *Introduction to JAVA Programming*, comprehensive version, 10th edition, Y. Daniel Liang.
2. *JAVA Concepts Early Objects*, 7th edition, Cay Horstmann, San Jose State University.
3. *Java Software Solutions*, Foundations of Program Design, Global Edition, John Lewis, William Loftus.
4. *Starting out with JAVA Early Objects*, 5th edition, Tony Daddis.

7 Source Code

7.1 Communication

1. Client.java

```java
package communication;

import javax.swing.*;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.nio.charset.StandardCharsets;
import java.util.Scanner;

public class Client
{
    private int port = 9000;
    private String ip = "127.0.0.1";
    private static Socket socket;
    private String client_id;
    private final int BUFFER_LENGTH = 8192;
    private JPanel chat_record_area;

    public Client(String client_id, JPanel chat_record_area)
    {
        try
        {
            init(client_id, chat_record_area);
            handle(this.chat_record_area);
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    private void init(String client_id, JPanel chat_record_area) throws Exception
    {
        this.chat_record_area = chat_record_area;
        this.client_id = client_id;
        socket = new Socket(ip, port);

        try
        {
            String information = "init#" + client_id;
            socket.getOutputStream().write(information.getBytes(StandardCharsets.UT
F_8));
            socket.getOutputStream().flush();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public void handle(JPanel chat_record_area) throws Exception
    {
        Thread thread = new Thread(new Client_Reader(socket.getInputStream(), chat_
record_area, client_id));
```

```java
        thread.setName(client_id);
        thread.start();
    }

    public void send(String message, String target_id) throws IOException
    {
        try
        {
            String data_sent = "from " + client_id + " to " + target_id + "####message####" + message;
            socket.getOutputStream().write(data_sent.getBytes(StandardCharsets.UTF_8));
            socket.getOutputStream().flush();
            System.out.println(data_sent);
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public boolean cancel() throws IOException
    {
        try
        {
            socket.close();
        }
        catch (Exception e)
        {
            e.printStackTrace();
            return false;
        }

        return true;
    }
}
```

2. Client_Reader.java

```java
package communication;

import jdbc.Database;

import javax.swing.*;
import java.awt.*;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.Date;

public class Client_Reader implements Runnable
{
    private InputStream is;
    private final int BUFFER_LENGTH = 8192;
    private JPanel chat_record_area;
    private String user_id, user_id_source;

    public Client_Reader(InputStream is, JPanel chat_record_area, String user_id)
    {
        this.is = is;
        this.user_id_source = user_id.split("_")[1];
        this.user_id = user_id.split("_")[0];
        this.chat_record_area = chat_record_area;
    }

    @Override
    public void run()
    {
        try
        {
            while (true)
            {
                byte[] b = new byte[BUFFER_LENGTH];
                int length = is.read(b);
                String message = new String(b, 0, length);
                System.out.println(message);
                System.out.println(user_id + "     " + user_id_source);

                message = message.split("####message####")[1];

                String query = "select user_name from user where user_id=?";
                ResultSet result = (new Database()).query(query, new String[]{user_id_source}, new String[]{"String"});
                result.next();

                JPanel single_chat = new JPanel();
                single_chat.setSize(900, 100);
                single_chat.setPreferredSize(new Dimension(900, 100));
                //single_chat.setLayout(new GridLayout(1, 2));
                ImageIcon user_image = new ImageIcon("img/profile/" + user_id_source + ".jpg");
                JLabel profile_label = new JLabel();
                profile_label.setIcon(user_image);
                profile_label.setSize(100, 100);

                JPanel chat_panel = new JPanel();
                chat_panel.setLayout(new GridLayout(2, 1, 0, 20));
                JLabel user_name_time = new JLabel(result.getString("user_name") + "             " + (new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")).format(new Date()));
                JLabel chat_content = new JLabel(message);
                chat_content.setOpaque(true);
                chat_panel.add(user_name_time);
                chat_panel.add(chat_content);

                profile_label.setHorizontalAlignment(SwingConstants.LEFT);
                user_name_time.setHorizontalAlignment(SwingConstants.LEFT);
                chat_content.setHorizontalAlignment(SwingConstants.LEFT);
                chat_content.setBackground(new Color(200, 200, 200));

                single_chat.add(profile_label);
                single_chat.add(chat_panel);

                chat_record_area.setLayout(new GridLayout(100, 1));
                chat_record_area.add(single_chat);
            }
        }
        catch (IOException | SQLException e)
        {
            e.printStackTrace();
        }
    }
}
```

3. Server.java

```java
package communication;

import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class Server
{

    private int port = 9000;
    private ServerSocket server;
    private Socket socket;
    private String server_name;
    private final int BUFFER_LENGTH = 8192;

    private HashMap<String, Socket> client_list = new HashMap<String, Socket>();

    public Server(String server_name)
    {
        try
        {
            init(server_name);
        }
        catch (IOException | InterruptedException e)
        {
            e.printStackTrace();
        }
    }

    private void init(String server_name) throws IOException, InterruptedException

    {
        this.server = new ServerSocket(port);
        this.server_name = server_name;

        while (true)
        {
            socket = server.accept();
            handle(socket);
        }
    }

    private void handle(Socket socket) throws IOException, InterruptedException
    {
        String key = socket.getInetAddress().getHostAddress() + ":" + socket.getPort();
        System.out.println("监听到的客户端： " + key);

        Thread thread = new Thread(new Server_Handler(socket, this.client_list));
        thread.start();
    }

    public HashMap<String, Socket> getClient_list()
    {
        return this.client_list;
    }
}
```

4. Server_Handler.java

```
package communication;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.nio.charset.StandardCharsets;
import java.util.HashMap;

public class Server_Handler implements Runnable
{
    private Socket socket;
    private OutputStream os;
    private InputStream is;
    private final int BUFFER_LENGTH = 8192;
    private HashMap<String, Socket> client_list;

    public Server_Handler(Socket socket, HashMap<String, Socket> client_list) throw
s IOException
    {
        this.socket = socket;
        this.is = socket.getInputStream();
        this.os = socket.getOutputStream();
        this.client_list = client_list;
    }

    @Override
    public void run()
    {
        try
        {
            while (true)
            {
                byte[] b = new byte[BUFFER_LENGTH];
                int length = is.read(b);
                String message = new String(b, 0, length);

                System.out.println("Server: " + message);

                Thread.sleep(1000);

                if (message.contains("init#"))
                {
                    byte[] init_information = new byte[BUFFER_LENGTH];
                    int init_len = length;
                    String client_id = message.split("#")[1];

                    this.client_list.put(client_id, socket);
```

```java
                    continue;
                }

                String[] source = message.split("####message####");
                String[] header = source[0].split(" ");
                String client_target = header[3];

                Socket target_socket = this.client_list.get(client_target);
                target_socket.getOutputStream().write(message.getBytes(StandardCharsets.UTF_8));
                target_socket.getOutputStream().flush();
            }
        }
        catch (IOException | InterruptedException e)
        {
            e.printStackTrace();
        }
    }
}
```

7.2 GUI

1. Chat_Window.java

```java
package GUI;

import jdbc.Database;
import jdk.dynalink.Operation;

import javax.swing.*;
import java.awt.*;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Chat_Window
{
    private Database sql;
    private String user_id;
    private String target_user_id;
    private JFrame frame = new JFrame();

    Chat_Window(String user_id, String target_user_id, Database sql) throws SQLException
    {
        this.sql = sql;
        this.user_id = user_id;
        this.target_user_id = target_user_id;

        String query = "select user_name from user where user_id=?";
        ResultSet result = sql.query(query, new String[]{user_id}, new String[]{"String"});
        result.next();
        String user_name = result.getString("user_name");
        query = "select user_name from user where user_id=?";
        result = sql.query(query, new String[]{target_user_id}, new String[]{"String"});
        result.next();
        String target_user_name = result.getString("user_name");

        frame.setTitle("Chat Window for " + user_name + " to " + target_user_name);

        frame.setSize(900, 600);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
        frame.setLayout(new GridLayout(2, 1));
        frame.setVisible(true);
        frame.setResizable(false);

        init(user_name, target_user_name);
    }

    public void init(String user_name, String target_user_name) throws SQLException
    {
        JPanel chat_record_area = new JPanel();
        JPanel send_message_area = new JPanel();
        JScrollPane scrollPane = new JScrollPane(chat_record_area, ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED, ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER);
        frame.add(scrollPane);
        frame.add(send_message_area);

        chat_record_area.setBorder(BorderFactory.createLineBorder(Color.gray, 1, true));
        send_message_area.setBorder(BorderFactory.createLineBorder(Color.gray, 1, true));

        //chat_record_area.setLayout(new GridLayout());

        JTextArea message_area = new JTextArea();
        message_area.setColumns(70);
        message_area.setLineWrap(true);
        message_area.setRows(18);

        JButton send = new JButton("Send");
        send.setSize(100, 50);
        send.setVerticalAlignment(SwingConstants.BOTTOM);

        send_message_area.add(message_area);
        send_message_area.add(send);

        String query = "select chat_time, content, send_user_id, receive_user_id from chat_record_single where (send_user_id=? and receive_user_id=?) or (send_user_id=? and receive_user_id=?) order by chat_time desc";
        ResultSet result = sql.query(query, new String[]{user_id, target_user_id, target_user_id, user_id}, new String[]{"String", "String", "String", "String"});
        int length = 0;
        while (result.next())
        {
            length++;
            JPanel single_chat = new JPanel();
            single_chat.setSize(900, 100);
            single_chat.setPreferredSize(new Dimension(900, 100));
            //single_chat.setLayout(new GridLayout(1, 2));
            ImageIcon user_image = new ImageIcon("img/profile/" + user_id + ".jpg");
            ImageIcon target_user_image = new ImageIcon("img/profile/" + target_user_id + ".jpg");
            JLabel profile_label = new JLabel();
            profile_label.setSize(100, 100);

            JPanel chat_panel = new JPanel();
            chat_panel.setLayout(new GridLayout(2, 1, 0, 20));
            JLabel user_name_time = new JLabel();
            JLabel chat_content = new JLabel(result.getString("content"));
            chat_content.setOpaque(true);
            chat_panel.add(user_name_time);
            chat_panel.add(chat_content);

            if (result.getString("send_user_id").equals(user_id))
            {
                profile_label.setIcon(user_image);
                user_name_time.setText(result.getTimestamp("chat_time").toString().substring(5, result.getTimestamp("chat_time").toString().indexOf('.')) + "    " + user_name);

                profile_label.setHorizontalAlignment(SwingConstants.RIGHT);
                user_name_time.setHorizontalAlignment(SwingConstants.RIGHT);
                chat_content.setHorizontalAlignment(SwingConstants.RIGHT);
                chat_content.setBackground(new Color(100, 200, 100));

                single_chat.add(chat_panel);
                single_chat.add(profile_label);
            }
            else
            {
                profile_label.setIcon(target_user_image);
                user_name_time.setText(target_user_name + "            " + result.getTimestamp("chat_time").toString().substring(5, result.getTimestamp("chat_time").toString().indexOf('.')));

                profile_label.setHorizontalAlignment(SwingConstants.LEFT);
                user_name_time.setHorizontalAlignment(SwingConstants.LEFT);
                chat_content.setHorizontalAlignment(SwingConstants.LEFT);
                chat_content.setBackground(new Color(200, 200, 200));

                single_chat.add(profile_label);
                single_chat.add(chat_panel);
            }

            chat_record_area.setLayout(new GridLayout(100, 1));
            single_chat.setLayout(new FlowLayout());

            chat_record_area.add(single_chat);
        }

        Thread thread = new Thread(new Session_Client(user_name, user_id, target_user_id, chat_record_area, message_area, send, length));
        thread.start();
    }
}
```

2. Friend_List.java

```
package GUI;

import jdbc.Database;

import javax.swing.*;
```

```java
import java.awt.*;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.util.ArrayList;
import java.util.Collections;

public class Friend_List
{
    private JFrame frame = new JFrame("Friend List");
    private Database sql;
    private String user_id;
    private int friend_number = 0;
    private ArrayList<String> friends = new ArrayList<>();

    Friend_List(String user_id, Database sql) throws SQLException
    {
        this.sql = sql;
        this.user_id = user_id;

        String query = "select * from friend_pair where user1_id=? or user2_id=?";

        ResultSet result = sql.query(query, new String[]{user_id, user_id}, new String[]{"String", "String"});

        while (result.next())
        {
            friend_number++;

            if (result.getString("user1_id").equals(user_id))
            {
                this.friends.add(result.getString("user2_id"));
            }
            else
            {
                this.friends.add(result.getString("user1_id"));
            }
        }

        frame.setLocationRelativeTo(null);
        frame.setLayout(new GridLayout(9, 1, 0, 5));
        frame.setSize(300, 900);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        set_personal_information();
        init_friend();

        frame.setVisible(true);
    }

    public void set_personal_information() throws SQLException
    {
        String query = "select user_name from user where user_id=?";
        ResultSet result = sql.query(query, new String[]{user_id}, new String[]{"String"});
        result.next();
        String user_name = result.getString("user_name");

        JPanel personal_information = new JPanel();

        personal_information.setLayout(new GridLayout(1, 2, 20, 20));
        ImageIcon profile = new ImageIcon("img/profile/" + user_id + ".jpg");
        JLabel profile_label = new JLabel();
        profile_label.setSize(100, 100);
        profile_label.setIcon(profile);
        personal_information.add(profile_label);

        JPanel right_panel = new JPanel();
        right_panel.setLayout(new GridLayout(2, 1, 0, 10));
        JLabel user_name_label = new JLabel("User Name: " + user_name);
        user_name_label.setFont(new Font(null, Font.PLAIN, 18));
        right_panel.add(user_name_label);

        JLabel user_friend_number = new JLabel("Friend Number: " + friend_number);

        user_friend_number.setFont(new Font(null, Font.PLAIN, 18));
        right_panel.add(user_friend_number);

        personal_information.add(right_panel);

        personal_information.setLayout(new FlowLayout());
        personal_information.setBorder(BorderFactory.createLineBorder(Color.gray, 3, true));
        frame.add(personal_information);
    }

    public void init_friend() throws SQLException
    {
        for (int i = 0; i < friend_number; i++)
        {
            String current_friend_id = friends.get(i);
            String current_friend_name = "";
            String recent_chat_content = "";
            Timestamp chat_time = null;
            String query = "select content, chat_time from chat_record_single where (send_user_id=? and receive_user_id=?) or (send_user_id=? and receive_user_id=?) order by chat_time desc";
            ResultSet result = this.sql.query(query, new String[]{user_id, current_friend_id, current_friend_id, user_id}, new String[]{"String", "String", "String", "String"});

                if (result.next())
                {
                    recent_chat_content = result.getString("content");
                    if (recent_chat_content.length() > 20)
                    {
                        recent_chat_content = recent_chat_content.substring(0, 16) + "...";
                    }
                    else
                    {
                        recent_chat_content += String.join("", Collections.nCopies(20 - recent_chat_content.length(), " "));
                    }

                    chat_time = result.getTimestamp("chat_time");
                }

                query = "select user_name from user where user_id=?";
                result = this.sql.query(query, new String[]{current_friend_id}, new String[]{"String"});
                result.next();
                current_friend_name = result.getString("user_name");

                JPanel single_friend = new JPanel();

                single_friend.setSize(290, 100);
                single_friend.setPreferredSize(new Dimension(290, 100));
                single_friend.setBorder(BorderFactory.createLineBorder(Color.gray, 3, true));
                single_friend.setLayout(new GridLayout(1, 2, 0, 0));

                ImageIcon profile = new ImageIcon("img/profile/" + current_friend_id + ".jpg");
                JLabel profile_label = new JLabel();
                profile_label.setIcon(profile);
                profile_label.setSize(100, 100);
                profile_label.setHorizontalAlignment(SwingConstants.CENTER);
                profile_label.setVerticalAlignment(SwingConstants.CENTER);
                single_friend.add(profile_label);

                JPanel main_panel = new JPanel();
                main_panel.setLayout(new GridLayout(2, 1, 0, 0));
                JPanel user_name_time = new JPanel();
                user_name_time.setLayout(new GridLayout(1, 2, 20, 0));
                JLabel user_name = new JLabel(current_friend_name);
                user_name.setSize(90, 50);
                user_name.setFont(new Font(null, Font.PLAIN, 18));
                user_name.setHorizontalAlignment(SwingConstants.LEFT);
                user_name_time.add(user_name);
                JLabel user_time = new JLabel(chat_time == null ? "" : chat_time.toString().substring(5, chat_time.toString().indexOf('.')));
                user_time.setSize(100, 50);
                user_time.setFont(new Font(null, Font.PLAIN, 18));
                user_time.setHorizontalAlignment(SwingConstants.RIGHT);
                user_name_time.add(user_time);
                main_panel.add(user_name_time);
                user_name_time.setLayout(new FlowLayout());

                JLabel chat_record = new JLabel(recent_chat_content);
                chat_record.setHorizontalAlignment(SwingConstants.LEFT);
                chat_record.setForeground(Color.LIGHT_GRAY);
                chat_record.setFont(new Font(null, Font.PLAIN, 18));
                main_panel.add(chat_record);

                single_friend.add(main_panel);
                single_friend.setLayout(new FlowLayout());

                single_friend.addMouseListener(new MouseListener()
                {
                    @Override
                    public void mouseClicked(MouseEvent e)
                    {
                        try
                        {
                            Chat_Window chat_window = new Chat_Window(user_id, current_friend_id, sql);
                        }
                        catch (SQLException throwables)
                        {
                            throwables.printStackTrace();
                        }
                    }

                    @Override
                    public void mousePressed(MouseEvent e)
                    {

                    }

                    @Override
                    public void mouseReleased(MouseEvent e)
                    {

                    }

                    @Override
                    public void mouseEntered(MouseEvent e)
                    {

                    }

                    @Override
                    public void mouseExited(MouseEvent e)
                    {

                    }
                });

                frame.add(single_friend);
            }
        }
    }
```

3. Login.java

```
package GUI;

import communication.Client;
import communication.Server;
import jdbc.*;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.sql.ResultSet;

public class Login
{
    private JFrame frame = new JFrame("Login");
    private Database sql;

    public Login(Database sql)
    {
        this.sql = sql;

        frame.setLayout(new GridLayout(3, 1, 0, 0));
        frame.setSize(600, 400);
```

```java
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        frame.setResizable(false);

        set_background();
        set_login_panel();

        frame.setVisible(true);
    }

    public void set_background()
    {
        JPanel background_panel = new JPanel();
        background_panel.setSize(500, 350);

        ImageIcon bg = new ImageIcon("img/background.jpg");
        bg.setImage(bg.getImage().getScaledInstance(600, 200, Image.SCALE_DEFAULT))
;
        JLabel background_label = new JLabel();
        background_label.setIcon(bg);
        background_panel.add(background_label);

        frame.add(background_panel);
    }

    public void set_login_panel()
    {
        JLabel welcome = new JLabel("Welcome to WeChat!");
        welcome.setFont(new Font(null, Font.BOLD, 30));
        welcome.setHorizontalAlignment(SwingConstants.CENTER);
        welcome.setForeground(Color.RED);
        frame.add(welcome);

        JPanel login_panel = new JPanel(new GridLayout(3, 2, 10, 10));

        JLabel login_hint = new JLabel("Your Email: ");
        login_hint.setFont(new Font(null, Font.PLAIN, 20));
        login_panel.add(login_hint);

        JTextField user_id = new JTextField(20);
        user_id.setFont(new Font(null, Font.PLAIN, 20));
        login_panel.add(user_id);

        JLabel password_hint = new JLabel("Your Password: ");
        password_hint.setFont(new Font(null, Font.PLAIN, 15));
        login_panel.add(password_hint);

        JPasswordField password = new JPasswordField(20);
        password.setFont(new Font(null, Font.PLAIN, 20));
        login_panel.add(password);

        JButton login = new JButton("Sign In");
        login.setFont(new Font(null, Font.PLAIN, 15));
        login.addActionListener(new ActionListener()
        {
            @Override
            public void actionPerformed(ActionEvent e)
            {
                handle_login(user_id.getText(), new String(password.getPassword()))
;
            }
        });
        login_panel.add(login);

        JButton register = new JButton("Sign Up");
        register.setFont(new Font(null, Font.PLAIN, 15));
        login_panel.add(register);

        login_panel.setLayout(new FlowLayout());

        register.addActionListener(new ActionListener()
        {
            @Override
            public void actionPerformed(ActionEvent e)
            {
                Register register1 = new Register(sql);
            }
            });

            frame.add(login_panel);
        }

        public void handle_login(String user_id, String password)
        {
            try
            {
                String query = "select * from user where user_id=? and password=?";

                ResultSet result = sql.query(query, new String[]{user_id, password}, new String[]{"String", "String"});
                result.last();

                if (result.getRow() > 0)
                {
                    System.out.println("Login success");
                    Friend_List friend_list = new Friend_List(user_id, sql);
                    this.frame.setVisible(false);
                }
                else
                {
                    System.out.println("fail");
                }
            }
            catch (Exception e)
            {
                e.printStackTrace();
            }
        }
    }
```

4. Register.java

```
package GUI;

import jdbc.Database;
import jdbc.SendEmail;

```

```java
import javax.mail.MessagingException;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.nio.charset.StandardCharsets;
import java.security.GeneralSecurityException;
import java.sql.SQLException;

public class Register
{
    private JFrame frame = new JFrame("Register");
    private Database sql;
    private int code;

    Register(Database sql)
    {
        this.sql = sql;

        frame.setLayout(new GridLayout(2, 1, 0, 0));
        frame.setSize(600, 500);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
        frame.setResizable(false);

        set_register_panel();

        frame.setVisible(true);
        frame.setVisible(true);
    }

    public void set_register_panel()
    {
        JLabel welcome = new JLabel("<html><body><p align='center'>Welcome to WeCha
t!<br>Register Now!</p></body></html>");
        welcome.setFont(new Font(null, Font.BOLD, 30));
        welcome.setHorizontalAlignment(SwingConstants.CENTER);
        welcome.setForeground(Color.BLUE);
        frame.add(welcome);

        JPanel register_panel = new JPanel(new GridLayout(7, 2, 10, 10));

        JLabel register_hint = new JLabel("Your Email: ");
        register_hint.setFont(new Font(null, Font.PLAIN, 20));
        register_panel.add(register_hint);

        JTextField user_id = new JTextField(20);
        user_id.setFont(new Font(null, Font.PLAIN, 20));
        register_panel.add(user_id);

        JLabel password_hint = new JLabel("Your Password: ");
        password_hint.setFont(new Font(null, Font.PLAIN, 15));
        register_panel.add(password_hint);

        JPasswordField password = new JPasswordField(20);
        password.setFont(new Font(null, Font.PLAIN, 20));
        register_panel.add(password);

        JLabel user_name_hint = new JLabel("Your User Name: ");
        user_name_hint.setFont(new Font(null, Font.PLAIN, 15));
        register_panel.add(user_name_hint);

        JTextField user_name = new JTextField();
        user_name.setFont(new Font(null, Font.PLAIN, 20));
        user_name.setColumns(20);
        register_panel.add(user_name);

        JLabel name_hint = new JLabel("Your Real Name:   ");
        name_hint.setFont(new Font(null, Font.PLAIN, 15));
        register_panel.add(name_hint);

        JTextField name_field = new JTextField();
        name_field.setFont(new Font(null, Font.PLAIN, 30));
        name_field.setColumns(30);
        register_panel.add(name_field);

        JLabel phone_hint = new JLabel("Your Phone Number: ");
        name_field.setFont(new Font(null, Font.PLAIN, 13));
        register_panel.add(phone_hint);

        JTextField phone = new JTextField();
        phone.setFont(new Font(null, Font.PLAIN, 20));
        phone.setColumns(20);
        register_panel.add(phone);

        JLabel auth_code_hint = new JLabel("The Auth Code: ");
        auth_code_hint.setFont(new Font(null, Font.PLAIN, 15));
        register_panel.add(auth_code_hint);

        JTextField auth_code = new JTextField();
        auth_code.setFont(new Font(null, Font.PLAIN, 20));
        auth_code.setColumns(20);
        register_panel.add(auth_code);

        JPanel button_panel = new JPanel();
            button_panel.setLayout(new GridLayout(1, 3, 20, 0));
            register_panel.add(button_panel);

            JButton send_auto_code = new JButton("Send Auth Code");
            send_auto_code.setFont(new Font(null, Font.PLAIN, 15));
            button_panel.add(send_auto_code);

            send_auto_code.addActionListener(new ActionListener()
            {
                @Override
                public void actionPerformed(ActionEvent e)
                {
                    code = (int)(Math.random() * 1000000) + 100000;
                    System.out.println(code);

                    if (!user_id.getText().equals("") && user_id.getText().contains("@"))
                    {
                        try
                        {
                            SendEmail.send(user_id.getText(), Integer.toString(code));
                            JOptionPane.showMessageDialog(null, "We have sent the authentication code, please check and input.");
                        }
                        catch (MessagingException messagingException)
                        {
                            messagingException.printStackTrace();
                        }
                        catch (GeneralSecurityException generalSecurityException)
                        {
                            generalSecurityException.printStackTrace();
                        }
                    }
                }
            });

            JButton check = new JButton("Check Auth Code");
            check.setFont(new Font(null, Font.PLAIN, 15));
            JButton register = new JButton("Register");
            register.setFont(new Font(null, Font.PLAIN, 15));
            register.setVisible(false);
            check.addActionListener(new ActionListener()
            {
                @Override
                public void actionPerformed(ActionEvent e)
                {
                    String input_code = auth_code.getText();

                    if (input_code.equals(Integer.toString(code)))
                    {
                        JOptionPane.showMessageDialog(null, "Authentication successful!", "Succeed", JOptionPane.INFORMATION_MESSAGE);
                        register.setVisible(true);
                    }
                    else
                    {
                        JOptionPane.showMessageDialog(null, "Authentication failed!", "Error", JOptionPane.ERROR_MESSAGE);
                    }
                }
            });

            register.addActionListener(new ActionListener()
            {
                @Override
                public void actionPerformed(ActionEvent e)
                {
                    String email = user_id.getText();
                    String passcode = new String(password.getPassword());
                    String username = user_name.getText();
                    String real_name = name_field.getText();
                    String phone_number = phone.getText();

                    String insert = "insert into user values(?, ?, ?, ?, ?)";
                    try
                    {
                        sql.modify(insert, new String[]{email, passcode, username, real_name, phone_number}, new String[]{"String", "String", "String", "String", "String"});

                        JOptionPane.showMessageDialog(null, "Register Succeed!", "Succeed", JOptionPane.INFORMATION_MESSAGE);
                    }
                    catch (SQLException throwables)
                    {
                        throwables.printStackTrace();
                    }
                }
            });

            button_panel.add(check);
            button_panel.add(register);

            register_panel.setLayout(new FlowLayout());

            frame.add(register_panel);
        }
    }
```

5. Session_Client.java

```java
package GUI;


import communication.Client;
import jdbc.Database;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.text.SimpleDateFormat;
import java.util.Date;

public class Session_Client implements Runnable
{
    private int length;
    private String user_id;
    private String user_name;
    private JButton send_button;
    private JTextArea send_message_area;
    private String target_user_id;
    private JPanel chat_record_area;

    Session_Client(String user_name, String user_id, String target_user_id, JPanel chat_record_area, JTextArea send_message_area, JButton send_button, int length)
    {
        this.user_id = user_id;
        this.user_name = user_name;
        this.send_button = send_button;
        this.send_message_area = send_message_area;
        this.target_user_id = target_user_id;
        this.chat_record_area = chat_record_area;
    }

    @Override
    public void run()
    {
        Client client = new Client(user_id + "_" + target_user_id, chat_record_area);

        send_button.addActionListener(new ActionListener()
        {
            @Override
```

```java
      public void actionPerformed(ActionEvent e)
      {
        String message = send_message_area.getText();

        if (message.equals(""))
        {
          return;
        }

        try
        {
          client.send(message, target_user_id + "_" + user_id);

          String query = "insert into chat_record_single values(null, ?, ?, now(), ?)";
          (new Database()).modify(query, new String[]{user_id, target_user_id, message}, new String[]{"String", "String", "String"});

          JPanel single_chat = new JPanel();
          single_chat.setSize(900, 100);
          single_chat.setPreferredSize(new Dimension(900, 100));
          //single_chat.setLayout(new GridLayout(1, 2));
          ImageIcon user_image = new ImageIcon("img/profile/" + user_id + ".jpg");
          JLabel profile_label = new JLabel();
          profile_label.setIcon(user_image);
          profile_label.setSize(100, 100);

          JPanel chat_panel = new JPanel();
          chat_panel.setLayout(new GridLayout(2, 1, 0, 20));
          JLabel user_name_time = new JLabel((new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")).format(new Date()) + "      " + user_name);
          JLabel chat_content = new JLabel(message);
          chat_content.setOpaque(true);
          chat_panel.add(user_name_time);
          chat_panel.add(chat_content);

          profile_label.setHorizontalAlignment(SwingConstants.RIGHT);
          user_name_time.setHorizontalAlignment(SwingConstants.RIGHT);
          chat_content.setHorizontalAlignment(SwingConstants.RIGHT);
          chat_content.setBackground(new Color(100, 200, 100));

          single_chat.add(chat_panel);
          single_chat.add(profile_label);

          chat_record_area.setLayout(new GridLayout(100, 1));
          //single_chat.setLayout(new FlowLayout());

          chat_record_area.add(single_chat);
          send_message_area.setText("");
        }
        catch (IOException | SQLException ioException)
        {
          ioException.printStackTrace();
        }
      }
    });
  }
}
```

6. Session_Server.java

```java
package GUI;


import communication.Client;
import communication.Server;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.nio.charset.StandardCharsets;
import java.util.HashMap;

public class Session_Server implements Runnable
{
    @Override
    public void run()
    {
        Server server = new Server("leming");
    }
}
```

7.3 JDBC

1. Database.java

```java
package jdbc;

import java.awt.*;
import java.sql.*;
import java.util.ArrayList;
import java.util.Map;
import java.util.Date;

public class Database
{
    static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/wechat?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";

    static final String USER = "slm";
    static final String PASS = "123456";

    private Connection connection = null;

    public Database()
    {
        try
        {
            Class.forName(JDBC_DRIVER);

            System.out.println("连接数据库...");
            connection = DriverManager.getConnection(DB_URL, USER, PASS);
        }
        catch (SQLException se)
        {
            se.printStackTrace();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public void modify(String sql, String[] args, String[] types) throws SQLException
    {
        try
        {
            PreparedStatement statement = this.connection.prepareStatement(sql);
            for (int i = 0; i < args.length; i++)
            {
                if (types[i].equals("String"))
                {
                    statement.setString(i + 1, args[i]);
                }
                else if (types[i].equals("Datetime"))
                {
                    statement.setTimestamp(i + 1, new Timestamp((new Date()).getTime()));
                }
                else if (types[i].equals("Int"))
                {
                    statement.setInt(i + 1, Integer.parseInt(args[i]));
                }
            }

            statement.executeUpdate();
        }
        catch (SQLException e)
        {
            e.printStackTrace();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public ResultSet query(String sql, String[] args, String[] types)
    {
        try
        {
            PreparedStatement statement = this.connection.prepareStatement(sql);
            for (int i = 0; i < args.length; i++)
            {
                if (types[i].equals("String"))
                {
                    statement.setString(i + 1, args[i]);
                }
                else if (types[i].equals("Datetime"))
                {
                    statement.setTimestamp(i + 1, new Timestamp((new Date()).getTime()));
                }
                else if (types[i].equals("Int"))
                {
                    statement.setInt(i + 1, Integer.parseInt(args[i]));
                }
            }

            ResultSet result = statement.executeQuery();

            return result;
        }
        catch (SQLException e)
        {
            e.printStackTrace();
        }
           catch (Exception e)
           {
               e.printStackTrace();
           }

           return null;
       }
   }
```

2. SendEmail.java

```
package jdbc;

import javax.mail.*;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import java.security.GeneralSecurityException;
import java.util.Properties;

public class SendEmail
{
    public static void send(String recipient, String code) throws MessagingException, GeneralSecurityException
    {
        try
        {
            Properties properties = new Properties();
            properties.put("mail.transport.protocol", "smtp");// 连接协议
            properties.put("mail.smtp.host", "smtp.qq.com");// 主机名
            properties.put("mail.smtp.port", 465);// 端口号
            properties.put("mail.smtp.auth", "true");
            properties.put("mail.smtp.ssl.enable", "true");// 设置是否使用 ssl 安全连接 ---一般都使用
            properties.put("mail.debug", "true");// 设置是否显示 debug 信息 true 会在控制台显示相关信息

            // 得到回话对象
            Session session = Session.getInstance(properties);
            // 获取邮件对象
            Message message = new MimeMessage(session);
            // 设置发件人邮箱地址
            message.setFrom(new InternetAddress("zjuslm@qq.com"));
            // 设置收件人邮箱地址
            message.setRecipient(Message.RecipientType.TO, new InternetAddress(recipient));//一个收件人
            // 设置邮件标题
            message.setSubject("WeChat Register Authentication");
            // 设置邮件内容
```

```java
            message.setText("Here is your code: " + code + ". Please take care of it.");
            // 得到邮差对象
            Transport transport = session.getTransport();
            // 连接自己的邮箱账户
            transport.connect("zjuslm@qq.com", "wohjryfplfvqhgjd");// 密码为 QQ 邮箱开通的 stmp 服务后得到的客户端授权码
            // 发送邮件
            transport.sendMessage(message, message.getAllRecipients());
            transport.close();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

浙江大學 计算机科学与技术学院

COLLEGE OF COMPUTER SCIENCE AND TECHNOLOGY
ZHEJIANG UNIVERSITY