گزارشکار پروژه نهایی

درس اصول طراحی پایگاه داده ها

تهیه کننده : اشکان مقرب رازلیقی

شماره دانشجویی : ۹۸۲۳۰۸۱

نام استاد : دکتر حمیدرضا شهریاری

توجه :

این پروژه در قالب client server پیاده سازی شده است اما در گزارشکاربه دلیل همپوشانی بخش server و client و وضوح بیشتر کد بخش server ، تنها این بخش توضیح داده شده است .

Server Class

```java
package Server;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.sql.*;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class Server {
    public static void main(String[] args) {
        ExecutorService pool = Executors.newCachedThreadPool();
        try (ServerSocket welcomingSocket = new ServerSocket( port: 7600)) {
            System.out.println("Server started");
            int count = 0;
            while (count != 20) {
                count++;
                Socket connectionSocket = welcomingSocket.accept();
                pool.execute(new ClientHandler(connectionSocket));
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
}
```

توضیحات : در این کلاس برای هرکاربر(تا سقف ۲۰ کاربر) که به سرور وصل شود، یک Socket در نظر گرفته می شود و سپس این socket به کلاس Client Handler داده می شود .

توضیحات کلاس Client Handler پس از بیان توضیحات کلاس های Services و AuthenticationService گفته خواهد شد .

AuthenticationService Class

```
private void act() {
    label:
    while (true) {
        if (!ID.equals("")) {
            break;
        }
        String msg = "Welcome " + "\n" + "1:sign up" + '\n' + "2:Sign in" + '\n' + "3:recover password" + '\n' + "4:Quit";
        try {
            out = connectionSocket.getOutputStream();
            in = connectionSocket.getInputStream();
            usefulmethods.send_message(out, msg);
            msg = usefulmethods.read_message(in);
            switch (msg) {
                case "1":
                    signUp();
                    break;
                case "2":
                    signIn();
                    break;
                case "3":
                    recover_password();
                    break;
                default:
                    usefulmethods.send_message(out, string "Goodbye.coming back soon");
                    Quit = true;
                    break label;
            }
        } catch (IOException ex) {
            ex.printStackTrace();
```

توضیحات : در Constructor این کلاس ، متد act وجود دارد که با هربارساخت یک شی از این کلاس اجرا می شود.

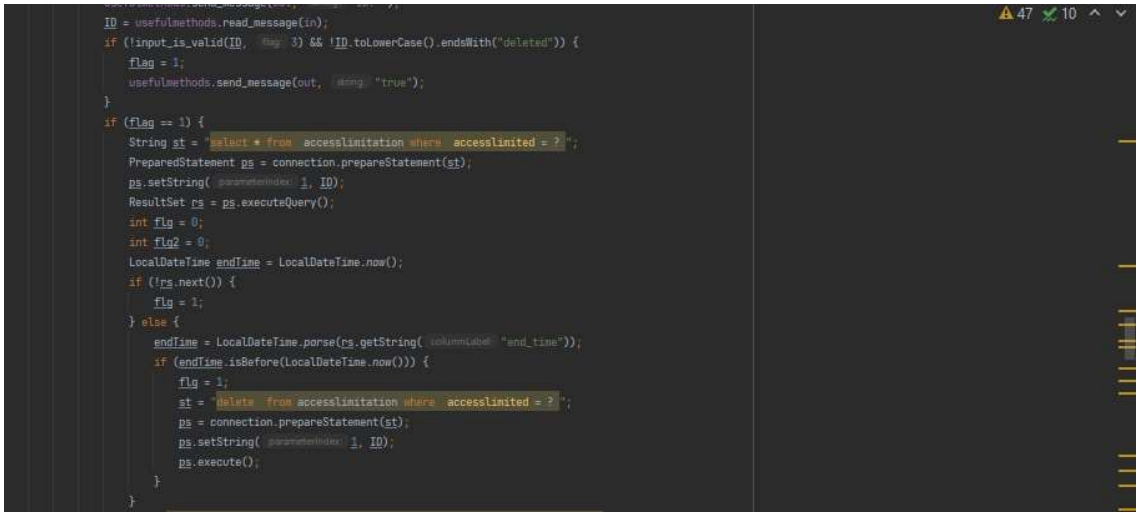
در متد act ابتدا به کاربریک Menu نمایش داده می شود که گزینه های آن به ترتیب مربوط به **ثبت نام** ، **واردشدن به حساب کاربری** ، **بازیابی رمزعبور** و**خروج از سامانه** می باشد . سپس باتوجه به انتخاب کاربر، یا به یکی از متدهای مربوطه خواهیم رفت یا از سامانه خارج می شویم .

signUp

بخشی از متد signUp

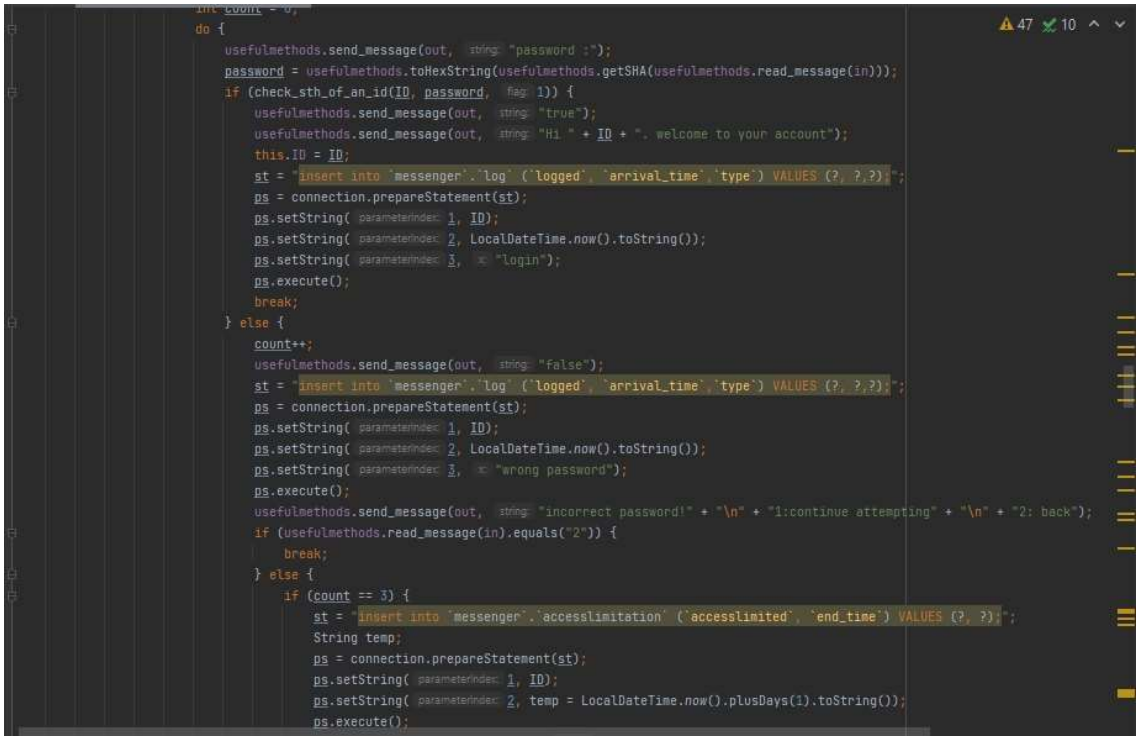
توضیحات : دراین متد نام، نام خانوادگی، شماره همراه(محدودیت شروع با ۰۹ ، ۱۱رقمی و غیرتکراری بودن برآن اعمال شده)، ایمیل(محدودیت غیرتکراری بودن بر آن اعمال شده) وآیدی (محدودیت غیرتکراری بودن بر آن اعمال شده)،
رمزعبور(محدودیت ترکیب حروف و عدد بودن برآن اعمال شده وبه صورت هش شده درمی آید) وسوال امنیتی(به صورت هش شده درمی آید) از کاربر گرفته می شود و درپایگاه داده ذخیره می شود .

signIn



توضیحات : در این بخش از این متد ، چک می شود که آیدی که کاربر برای ورود به حساب کاربری اش وارد کرده معتبر است یا نه ؟ و اگر بود ، دارای محدودیت دسترسی است یا نه ؟

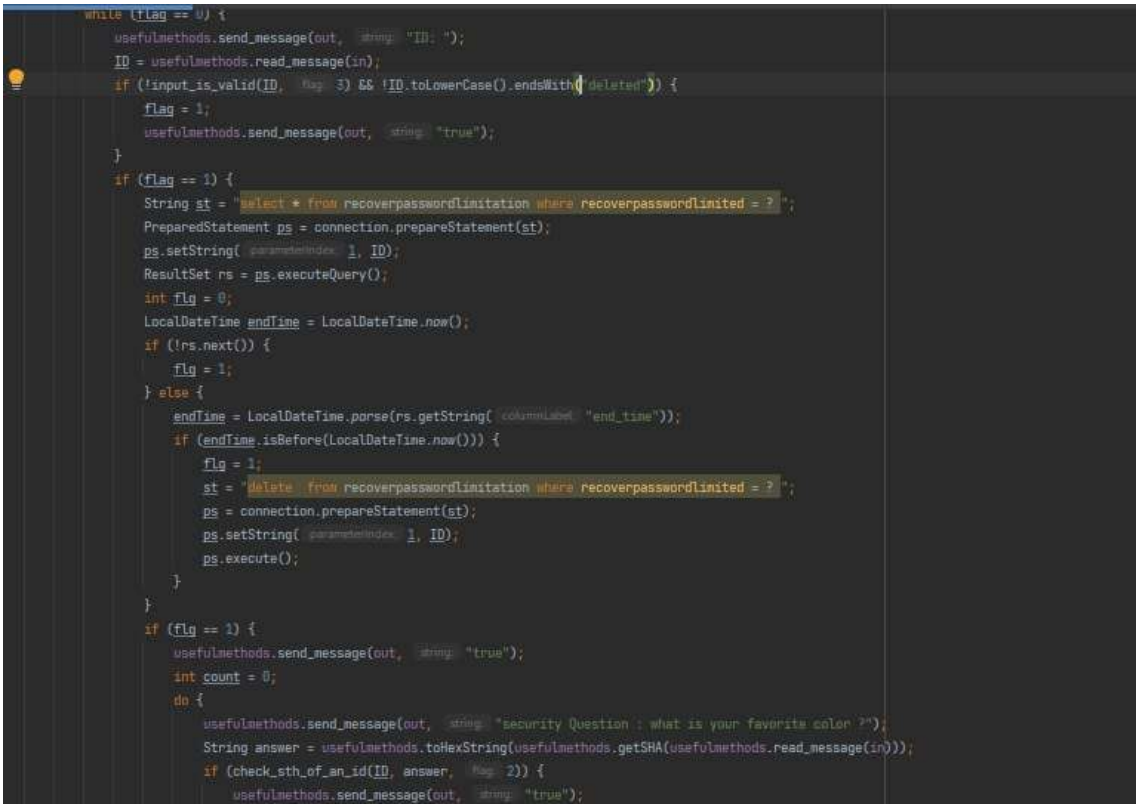
توضیحات : دراین بخش از این متد ، چک می شود که کاربر از جایی دیگر وارد حساب کاربری خود نشده باشد .



توضیحات : دراین بخش از این متد ، پس ازاینکه مطمئن شدیم کاربر محدودیت دسترسی ندارد و از جای دیگری وارد حساب کاربری اش نشده است ، ازاو رمز عبورش را می گیریم درصورتی که درست بود ، در جدول logged ورودش را درج می کنیم و در غیر این صورت تا ۳ بار به او اجازه واردکردن مجدد رمزعبورش را می دهیم .(هرباراشتباه واردکردن رمزعبوردرجدول logged درج می شود).

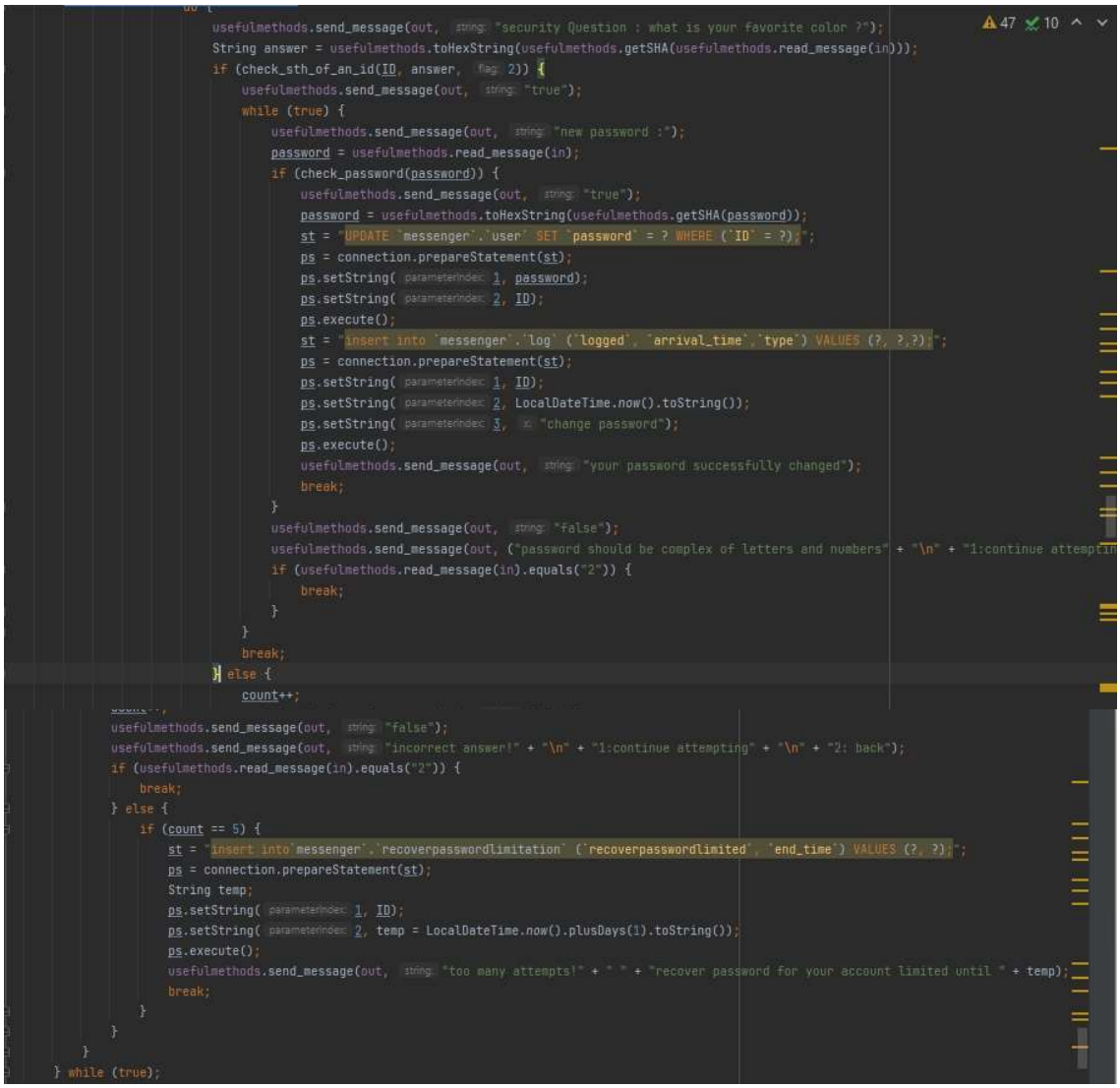
درصورتی که تا ۳ بار موفق به درست واردکردن رمزعبورش نشد ، او را به مدت یک روز از دسترسی به حساب کاربری اش منع می کنیم و این را در جدول accesslimited درج می کنیم .

recover_password



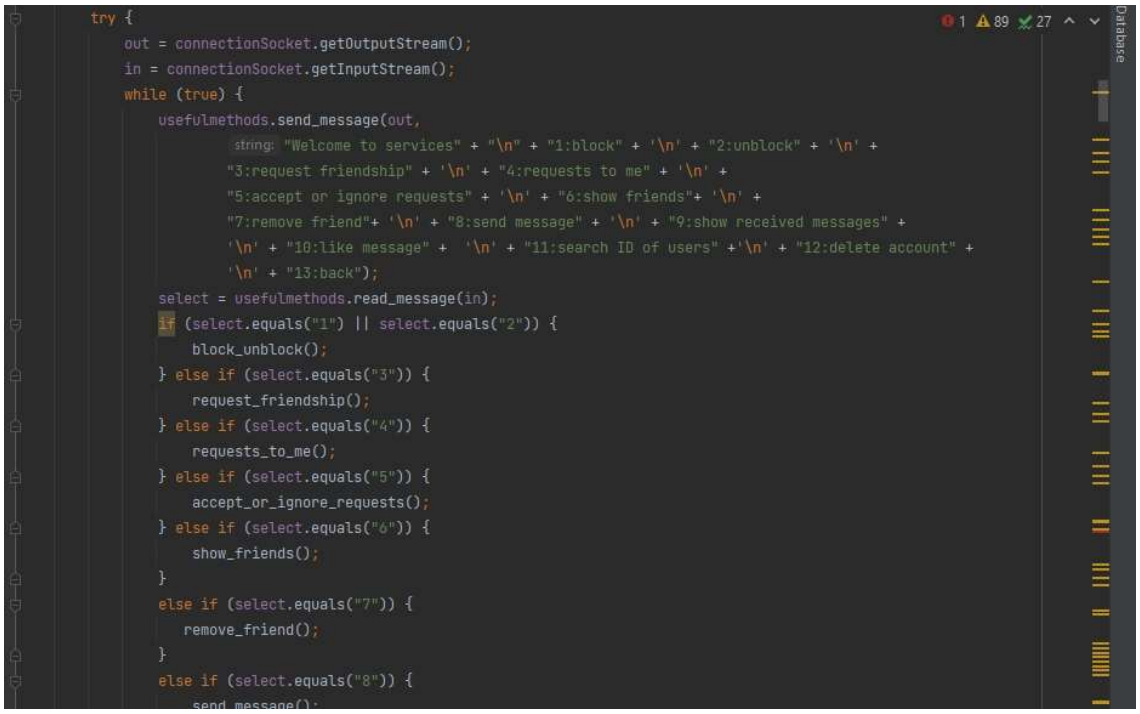
```
while (flag == 0) {
    usefulmethods.send_message(out, string: "ID: ");
    ID = usefulmethods.read_message(in);
    if (!input_is_valid(ID, flag: 3) && !ID.toLowerCase().endsWith("deleted")) {
        flag = 1;
        usefulmethods.send_message(out, string: "true");
    }
    if (flag == 1) {
        String st = "select * from recoverpasswordlimitation where recoverpasswordlimited = ? ";
        PreparedStatement ps = connection.prepareStatement(st);
        ps.setString( parameterIndex: 1, ID);
        ResultSet rs = ps.executeQuery();
        int flg = 0;
        LocalDateTime endTime = LocalDateTime.now();
        if (!rs.next()) {
            flg = 1;
        } else {
            endTime = LocalDateTime.parse(rs.getString( columnLabel: "end_time"));
            if (endTime.isBefore(LocalDateTime.now())) {
                flg = 1;
                st = "delete  from recoverpasswordlimitation where recoverpasswordlimited = ? ";
                ps = connection.prepareStatement(st);
                ps.setString( parameterIndex: 1, ID);
                ps.execute();
            }
        }
        if (flg == 1) {
            usefulmethods.send_message(out, string: "true");
            int count = 0;
            do {
                usefulmethods.send_message(out, string: "security Question : what is your favorite color ?");
                String answer = usefulmethods.toHexString(usefulmethods.getSHA(usefulmethods.read_message(in)));
                if (check_sth_of_an_id(ID, answer, flag: 2)) {
                    usefulmethods.send_message(out, string: "true");
```

توضیحات : دراین بخش از این متد ،چک می شود که آیدی که کاربر برای بازیابی رمزعبورش وارد کرده معتبر است یا نه ؟ و اگر بود ، دارای محدودیت بازیابی رمزعبور است یا نه ؟



توضیحات : دراین بخش از این متد ، سوال امنیتی پیش فرض از کاربر پرسیده می شود و درصورتی که درست جواب داد ، از او رمز عبور جدید گرفته می شود (با بررسی ترکیب حروف و عدد بودن آن، در جدول logged تغییررمزعبورش را درج می کنیم) . درغیر این صورت تا ۵ بار به او اجازه داده می شود به سوال امنیتی به درستی جواب دهد واگر نتوانست، او را به مدت ۱روزازبازیابی رمزعبورش منع می کنیم و این را در جدول recoverpasswordlimited درج می کنیم .

Services Class

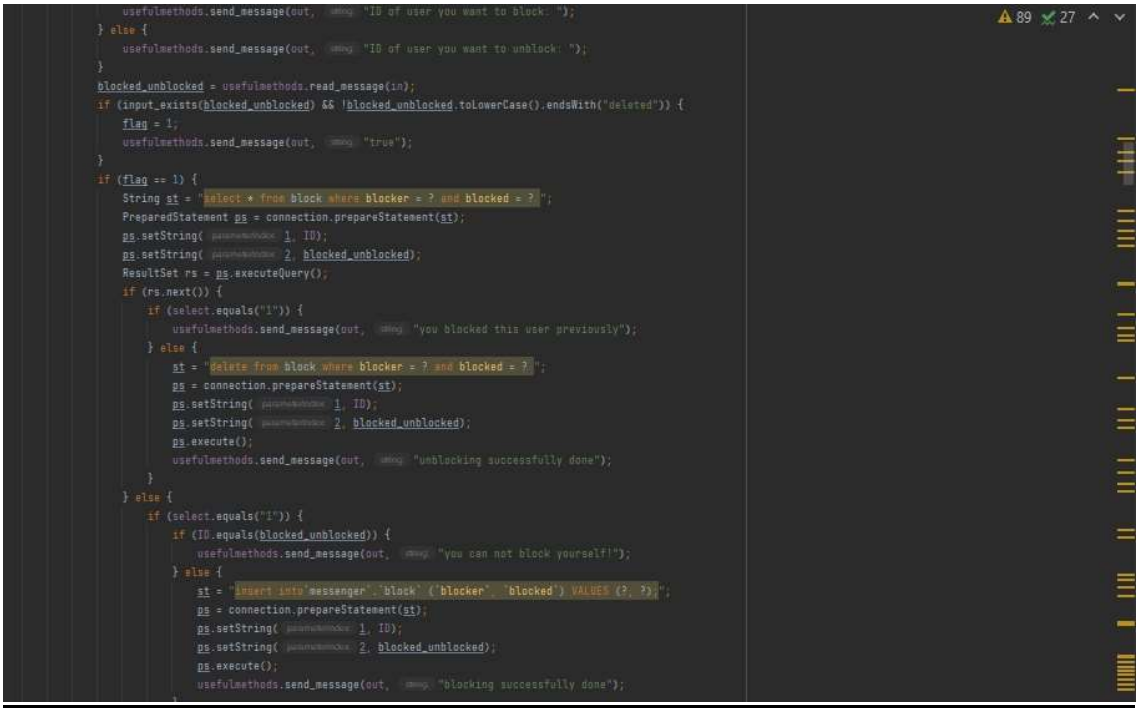


```
try {
    out = connectionSocket.getOutputStream();
    in = connectionSocket.getInputStream();
    while (true) {
        usefulmethods.send_message(out,
                string: "Welcome to services" + "\n" + "1:block" + '\n' + "2:unblock" + '\n' +
                "3:request friendship" + '\n' + "4:requests to me" + '\n' +
                "5:accept or ignore requests" + '\n' + "6:show friends"+ '\n' +
                "7:remove friend"+ '\n' + "8:send message" + '\n' + "9:show received messages" +
                '\n' + "10:like message" +  '\n' + "11:search ID of users" +'\n' + "12:delete account" +
                '\n' + "13:back");
        select = usefulmethods.read_message(in);
        if (select.equals("1") || select.equals("2")) {
            block_unblock();
        } else if (select.equals("3")) {
            request_friendship();
        } else if (select.equals("4")) {
            requests_to_me();
        } else if (select.equals("5")) {
            accept_or_ignore_requests();
        } else if (select.equals("6")) {
            show_friends();
        }
        else if (select.equals("7")) {
           remove_friend();
        }
        else if (select.equals("8")) {
            send_message();
```

توضیحات : در Constructor این کلاس ، متد act وجود دارد که با هربارساخت یک شی از این کلاس اجرا می شود.

در متد act ابتدا به کاربریک Menu نمایش داده می شود که گزینه های آن به ترتیب مربوط به **مسدودکردن کاربران**، **رفع مسدودیت**، **درخواست دوستی**، **دریافت درخواست های داده شده به کاربربرای دوستی**، **موافقت یا عدم موافقت با یک درخواست**، **دریافت لیست دوستان**، **حذف از لیست دوستان**، **ارسال پیام**، **لیست پیام های دریافتی**، **لایک یک پیام**، **امکان پیداکردن افراد**، **حذف حساب کاربری** و **بازگشت به عقب** می باشد. . سپس باتوجه به انتخاب کاربر، یا به یکی از متدهای مربوطه خواهیم رفت یا به عقب بازمی گردیم .

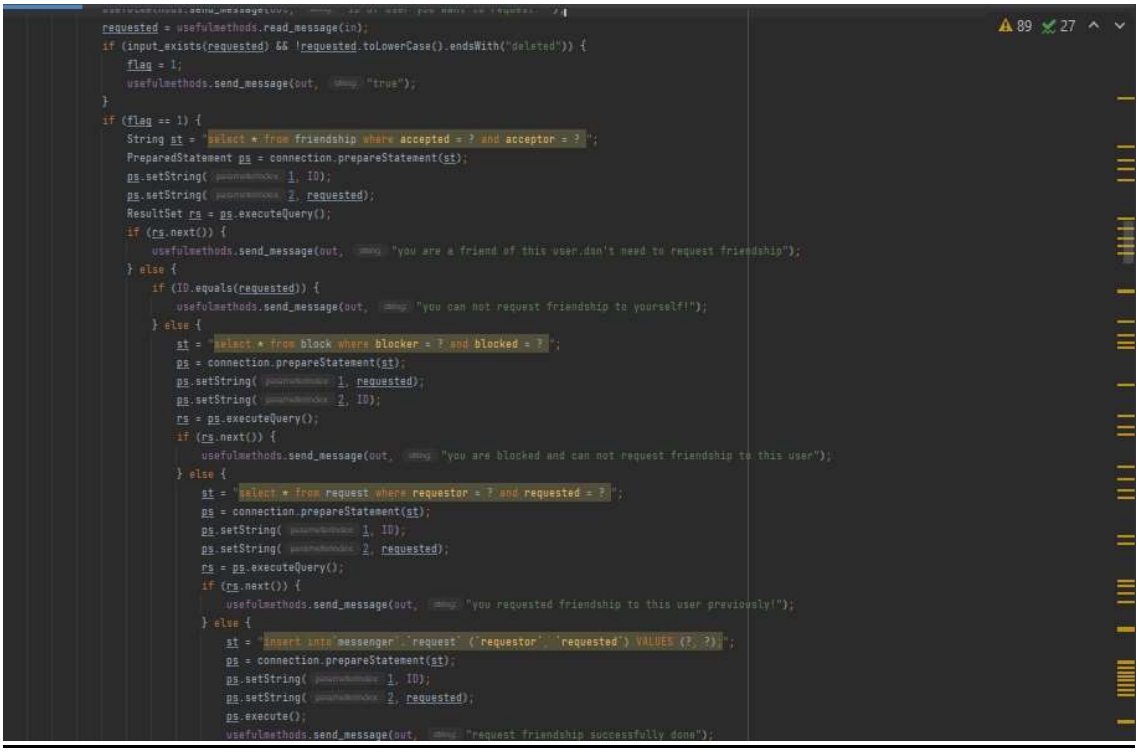
block_unblock



توضیحات : دراین متد، ابتدا آیدی فردی که کاربر می خواهد او را مسدود یا رفع مسدود کند گرفته می شود . پس از بررسی موجود بودن چنین آیدی ، درجدول بلاک می گردیم که آیا کاربر، این فرد را قبلا مسدود کرده است یا خیر. درصورتی که قبلا مسدود کرده باشد و درخواست مسدود کردن مجدد بدهد، به او اطلاع می دهیم که قبلا این کاربررا مسدودکرده است و درصورتی که درخواست رفع مسدودیت بدهد، آیدی کاربرو فرد مسدود شده را ازجدول بلاک حذف می کنیم .

درصورتی که قبلا مسدود نکرده باشد و درخواست مسدود کردن بدهد، آیدی کاربرو فرد مسدود شده را درجدول بلاک درج می کنیم ودرصورتی که درخواست رفع مسدودیت بدهد، به او اطلاع می دهیم که قبلا این کاربر را مسدود نکرده است .

request_friendship



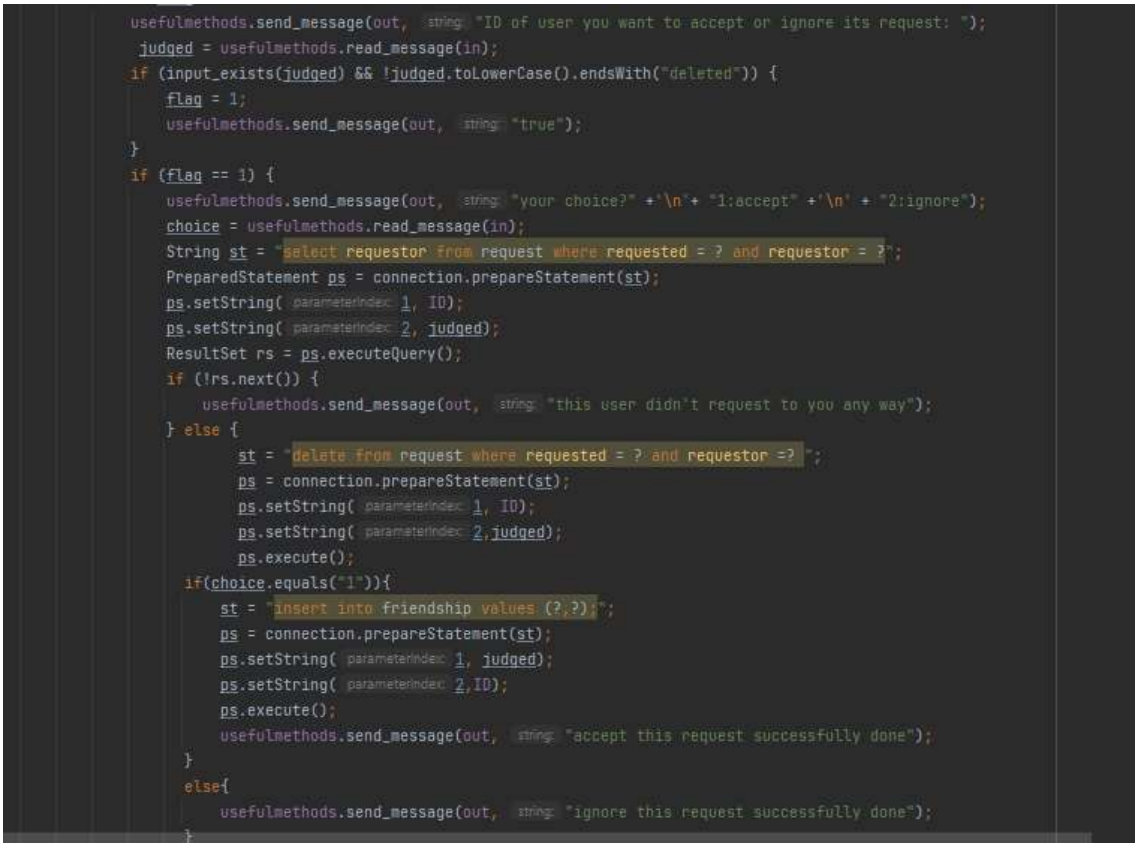
توضیحات : دراین متد، ابتدا آیدی فردی که کاربر می خواهد به اودرخواست دوستی دهد گرفته می شود. پس از بررسی موجود بودن چنین آیدی، با استفاده از جدول friendship بررسی می کنیم که آیا این کاربر دوست آن فرد بوده است یا خیر. درصورتی که بوده باشد، به اواطلاع می دهیم که دوست آن فرد است و نیاز به درخواست دوستی نیست . درغیراین صورت، چک می کنیم که آن فرد، کاربر را مسدود نکرده باشد . درصورتی که مسدود کرده بود، به او اطلاع می دهیم که نمی تواند درخواست دوستی دهد و درغیر این صورت چک می کنیم که کاربر قبلا درخواست دوستی دیگری را ثبت نکرده باشد . درصورت ثبت نشدن درخواست دیگری ، درخواست کاربر درجدول request درج می شود.

requests_to_me

```java
private void requests_to_me() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch (ClassNotFoundException cnfe) {
        System.out.println("Error loading driver");
    }
    try {
        Connection connection = DriverManager.getConnection(
                url "jdbc:mysql://localhost"
                ,
                user "ashkan1377",  password "aBc.123456");
        Statement statement = connection.createStatement();
        statement.execute( sql "use messenger");
        String st = "select requestor from request where  requested = ? ";
        PreparedStatement ps = connection.prepareStatement(st);
        ps.setString( parameterIndex 1, ID);
        ResultSet rs = ps.executeQuery();
        ArrayList<String> requestors = new ArrayList<>();
        while (rs.next()) {
            requestors.add(rs.getString( columnLabel "requestor"));
        }
        usefulmethods.send_message(out,  string "" + requestors.size());
        for (String requestor : requestors) {
            usefulmethods.send_message(out, requestor);
            Thread.sleep( millis 1);
        }
```

توضیحات: دراین متد، با استفاده از جدول request آیدی افرادی که درخواست دوستی به کاربر را داده اند ، به او نمایش داده می شود.

accept_or_ignore_requests



```
usefulmethods.send_message(out, string: "ID of user you want to accept or ignore its request: ");
 judged = usefulmethods.read_message(in);
if (input_exists(judged) && !judged.toLowerCase().endsWith("deleted")) {
    flag = 1;
    usefulmethods.send_message(out, string: "true");
}
if (flag == 1) {
    usefulmethods.send_message(out, string: "your choice?" +'\n'+ "1:accept" +'\n' + "2:ignore");
    choice = usefulmethods.read_message(in);
    String st = "select requestor from request where requested = ? and requestor = ?";
    PreparedStatement ps = connection.prepareStatement(st);
    ps.setString( parameterIndex: 1, ID);
    ps.setString( parameterIndex: 2, judged);
    ResultSet rs = ps.executeQuery();
    if (!rs.next()) {
        usefulmethods.send_message(out, string: "this user didn't request to you any way");
    } else {
           st = "delete from request where requested = ? and requestor =? ";
           ps = connection.prepareStatement(st);
           ps.setString( parameterIndex: 1, ID);
           ps.setString( parameterIndex: 2,judged);
           ps.execute();
      if(choice.equals("1")){
          st = "insert into friendship values (?,?)";
          ps = connection.prepareStatement(st);
          ps.setString( parameterIndex: 1, judged);
          ps.setString( parameterIndex: 2,ID);
          ps.execute();
          usefulmethods.send_message(out, string: "accept this request successfully done");
      }
      else{
          usefulmethods.send_message(out, string: "ignore this request successfully done");
      }
```

توضیحات : دراین متد، ابتدا آیدی فردی که کاربر می خواهد درخواست دوستی اش را رد یا قبول کند، گرفته می شود . پس از بررسی موجود بودن چنین آیدی، چک می شود که آیا همچین فردی درخواستی به کاربر داده است یا خیر. درصورتی که درخواست داده باشد، از جدول request درخواست او حذف می شود واگرکاربربا این درخواست موافقت کرده باشد، آیدی کاربر و آن فرد در جدول friendship درج می شود.

show_friends

```
private void show_friends() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch (ClassNotFoundException cnfe) {
        System.out.println("Error loading driver");
    }
    try {
        Connection connection = DriverManager.getConnection(
                url: "jdbc:mysql://localhost"
                ,
                user: "ashkan1377",  password: "aBc.123456");
        Statement statement = connection.createStatement();
        statement.execute( sql: "use messenger");
        String st = "select accepted from friendship where acceptor= ?";
        PreparedStatement ps = connection.prepareStatement(st);
        ps.setString( parameterIndex: 1, ID);
        ResultSet rs = ps.executeQuery();
        ArrayList<String> friends = new ArrayList<>();
        while(rs.next()){
            friends.add(rs.getString( columnLabel: "accepted"));
        }
        usefulmethods.send_message(out, string: "" +friends.size());
        for(String friend : friends){
            usefulmethods.send_message(out,friend);
            Thread.sleep( millis: 1);
        }
    }
    catch (SQLException | InterruptedException e1) {
        e1.printStackTrace();
    }
}
```

توضیحات : دراین متد، با استفاده از جدول friendship آیدی افرادی که دوست کاربر هستند، به او نمایش داده می شود.

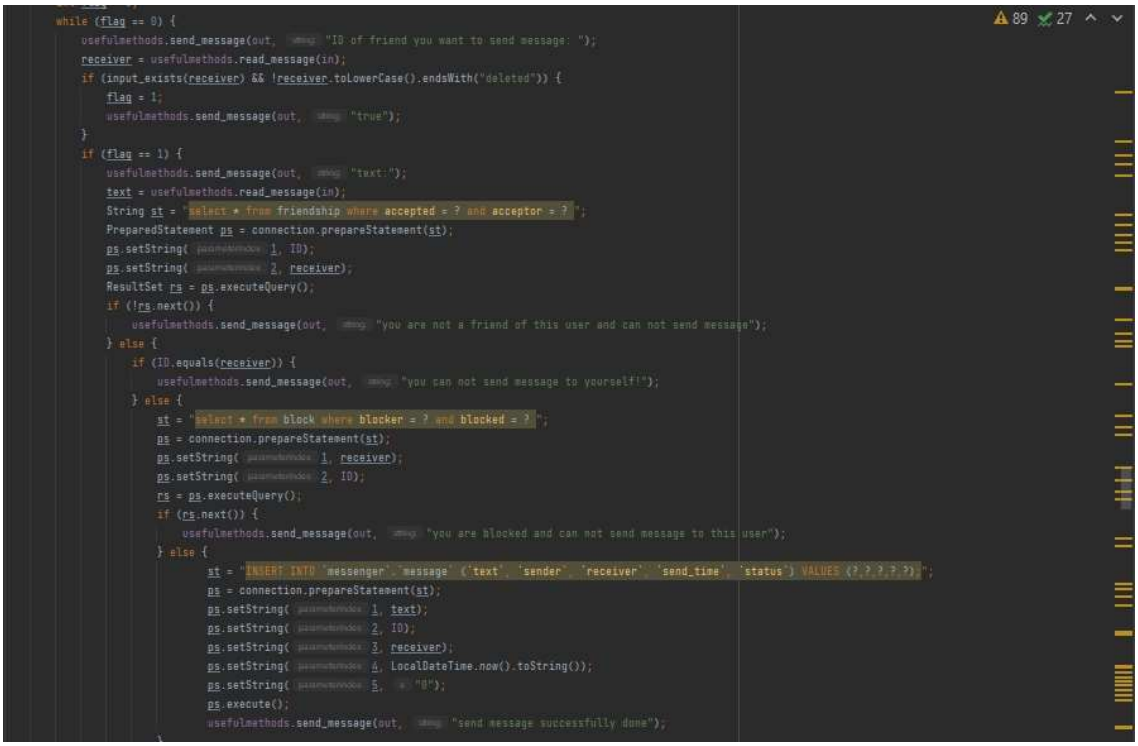
Remove_friend

```
usefulmethods.send_message(out,  string: "ID of friend you want to remove: ");
removed= usefulmethods.read_message(in);
if (input_exists(removed) && !removed.toLowerCase().endsWith("deleted")) {
    flag = 1;
    usefulmethods.send_message(out,  string: "true");
}
if (flag == 1) {
    String st = "select accepted from friendship where accepted = ? and acceptor = ?";
    PreparedStatement ps = connection.prepareStatement(st);
    ps.setString( parameterIndex: 1, removed);
    ps.setString( parameterIndex: 2, ID);
    ResultSet rs = ps.executeQuery();
    if (!rs.next()) {
        usefulmethods.send_message(out,  string: "this user isn't your friend");
    } else {
        st = "delete from friendship where accepted =? and  acceptor =? ";
        ps = connection.prepareStatement(st);
        ps.setString( parameterIndex: 1, removed);
        ps.setString( parameterIndex: 2,ID);
        ps.execute();
            usefulmethods.send_message(out,  string: "this user removed from your friends successfully");
    }
} else {
    usefulmethods.send_message(out,  string: "false");
    usefulmethods.send_message(out,  string: "There is no user with this username" + "\n" + "1:continue attempting" + "\n" + "2: back");
    if (usefulmethods.read_message(in).equals("2")) {
        break;
    }
```

توضیحات:دراین متد، ابتدا آیدی دوستی که کاربرمی خواهد حذفش کند،گرفته می شود.

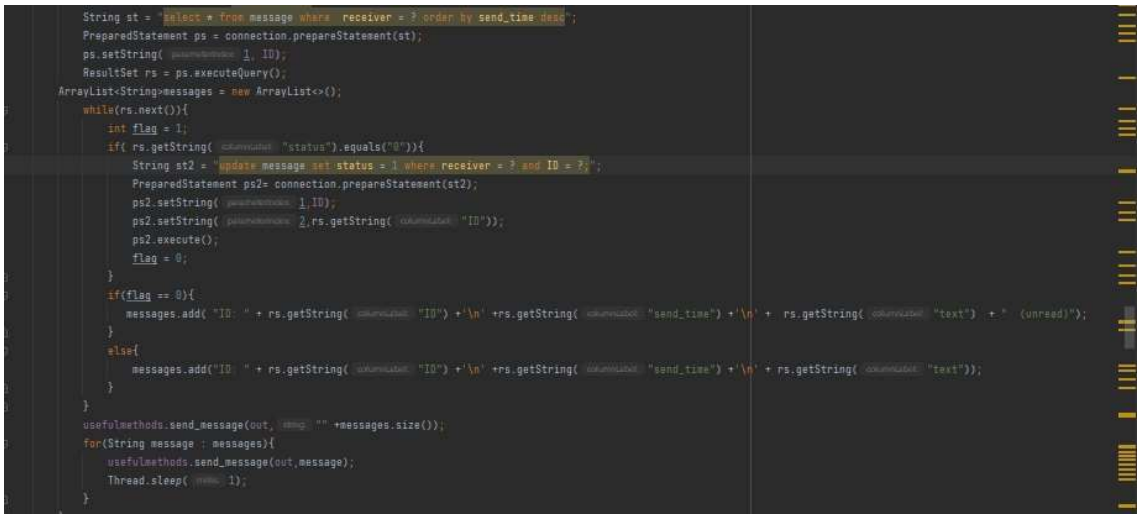
پس از بررسی موجود بودن چنین آیدی، با استفاده ازجدول friendship چک می شود که آیا این فرد دوست کاربر بوده است یا خیر. درصورتی که بوده باشد از جدول friendship تاپلی که مربوط به آیدی کاربر و آن فرد است حذف می شود.

send_message



توضیحات:دراین متد، ابتدا آیدی دوستی که کاربرمی خواهد به او پیام دهد،گرفته می شود. پس از بررسی موجود بودن چنین آیدی، با استفاده ازجدول friendship چک می شود که آیا کاربرجزو دوستان آن فرد است یا خیر. درصورتی که باشد، با استفاده ازجدول block ، چک می شود که آیا آن فرد، کاربر را مسدود کرده است یا خیر. درصورتی که نکرده باشد، متن پیام از کاربر گرفته می شود و درجدول message درج می شود .

show_received_messages

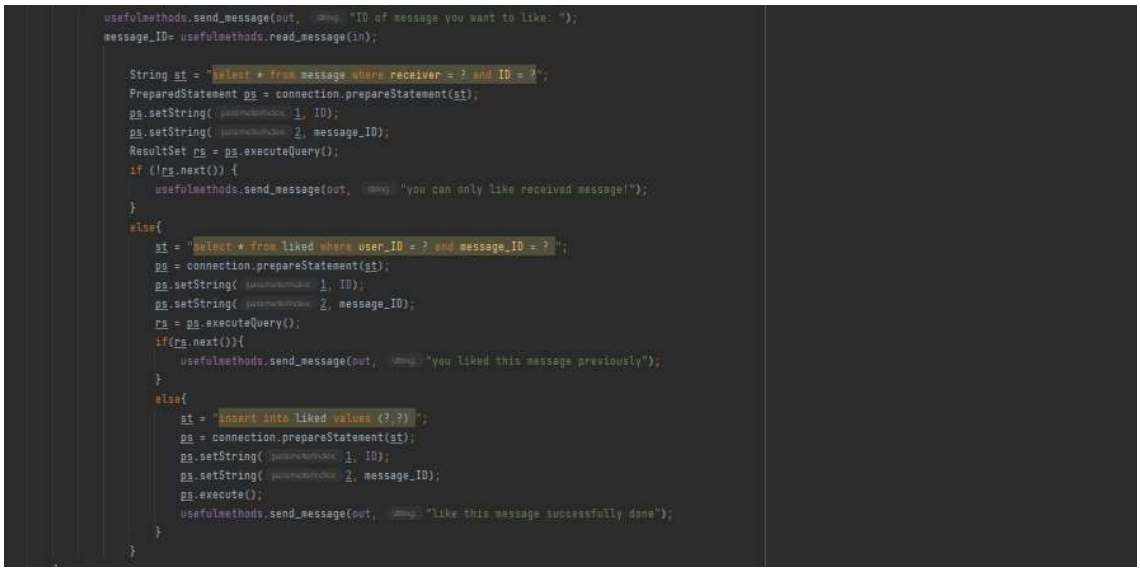


```
String st = "select * from message where  receiver = ? order by send_time desc";
PreparedStatement ps = connection.prepareStatement(st);
ps.setString( parameterIndex: 1, ID);
ResultSet rs = ps.executeQuery();
ArrayList<String>messages = new ArrayList<>();
while(rs.next()){
    int flag = 1;
    if( rs.getString( columnLabel: "status").equals("0")){
        String st2 = "update message set status = 1 where receiver = ? and ID = ?;";
        PreparedStatement ps2= connection.prepareStatement(st2);
        ps2.setString( parameterIndex: 1,ID);
        ps2.setString( parameterIndex: 2,rs.getString( columnLabel: "ID"));
        ps2.execute();
        flag = 0;
    }
    if(flag == 0){
       messages.add( e: "ID: " + rs.getString( columnLabel: "ID") +'\n' +rs.getString( columnLabel: "send_time") +'\n' +  rs.getString( columnLabel: "text")  + "  (unread)");
    }
    else{
        messages.add("ID  " + rs.getString( columnLabel: "ID") +'\n' +rs.getString( columnLabel: "send_time") +'\n' + rs.getString( columnLabel: "text"));
    }
}
usefulmethods.send_message(out, msg: "" +messages.size());
for(String message : messages){
    usefulmethods.send_message(out,message);
    Thread.sleep( millis: 1);
}
```

توضیحات: دراین متد، با استفاده از جدول message پیام هایی که دوستان کاربر برای او فرستاده اند به ترتیب نزولی برحسب زمان ارسال ، به او نمایش داده می شود.

باتوجه به اینکه پیامی که برای کاربر نمایش داده می شود قبلا خوانده شده یا خیربرچسب unread برای آن پیام نمایش داده می شود .

like_message



```
usefulmethods.send_message(out, msg: "ID of message you want to like: ");
message_ID= usefulmethods.read_message(in);

    String st = "select * from message where receiver = ? and ID = ?";
    PreparedStatement ps = connection.prepareStatement(st);
    ps.setString( parameterIndex: 1, ID);
    ps.setString( parameterIndex: 2, message_ID);
    ResultSet rs = ps.executeQuery();
    if (!rs.next()) {
        usefulmethods.send_message(out, msg: "you can only like received message!");
    }
    else{
        st = "select * from liked where user_ID = ? and message_ID = ? ";
        ps = connection.prepareStatement(st);
        ps.setString( parameterIndex: 1, ID);
        ps.setString( parameterIndex: 2, message_ID);
        rs = ps.executeQuery();
        if(rs.next()){
            usefulmethods.send_message(out, msg: "you liked this message previously");
        }
        else{
            st = "insert into liked values (?,?) ";
            ps = connection.prepareStatement(st);
            ps.setString( parameterIndex: 1, ID);
            ps.setString( parameterIndex: 2, message_ID);
            ps.execute();
            usefulmethods.send_message(out, msg: "like this message successfully done");
        }
    }
```

توضیحات: دراین متد، ابتدا آیدی پیامی که کاربر می خواهد لایک کند گرفته می شود .

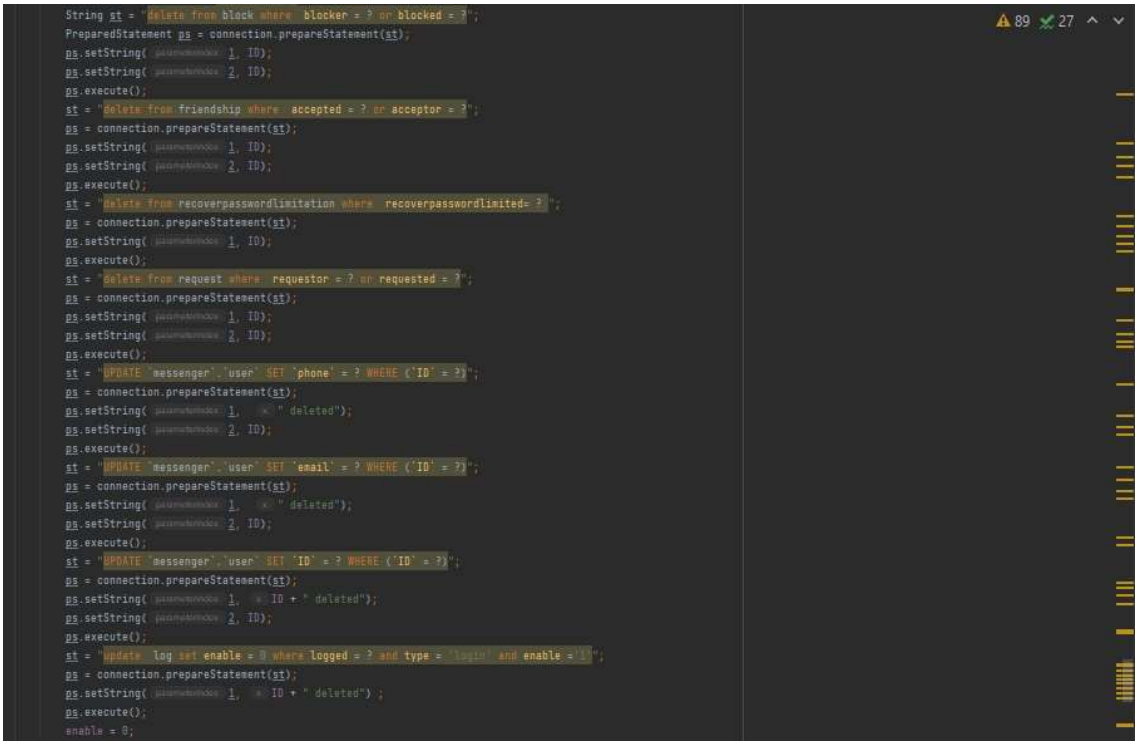
سپس با استفاده از جدول message، چک می شود که آیا این پیام جزو پیام های دریافتی کاربرهست یا خیر. درصورتی که باشد، با استفاده از جدول liked چک می کنیم که قبلا کاربراین پیام را لایک نکرده باشد . درصورتی که لایک نکرده بود، این لایک را در جدول liked درج می کنیم .

search

```
private void search() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch (ClassNotFoundException cnfe) {
        System.out.println("Error loading driver");
    }
    try {
        Connection connection = DriverManager.getConnection(
                url: "jdbc:mysql://localhost"
                ,
                user: "ashkan1377",  password: "aBc.123456");
        Statement statement = connection.createStatement();
        statement.execute( sql: "use messenger");
        usefulmethods.send_message(out,  string: "word: ");
        String word = usefulmethods.read_message(in);
        ResultSet rs = statement.executeQuery( sql: "select ID from user");
        ArrayList<String> nears = new ArrayList<>();
        while(rs.next()){
            String st = rs.getString( columnLabel: "ID");
            if(st.startsWith(word) && !st.endsWith("deleted") ){
                nears.add(st);
            }
        }
        usefulmethods.send_message(out, string: "" +nears.size());
        for(String near : nears){
            usefulmethods.send_message(out,near);
            Thread.sleep( millis: 1);
        }
    }
    catch (SQLException | InterruptedException e1) {
        e1.printStackTrace();
    }
}
```

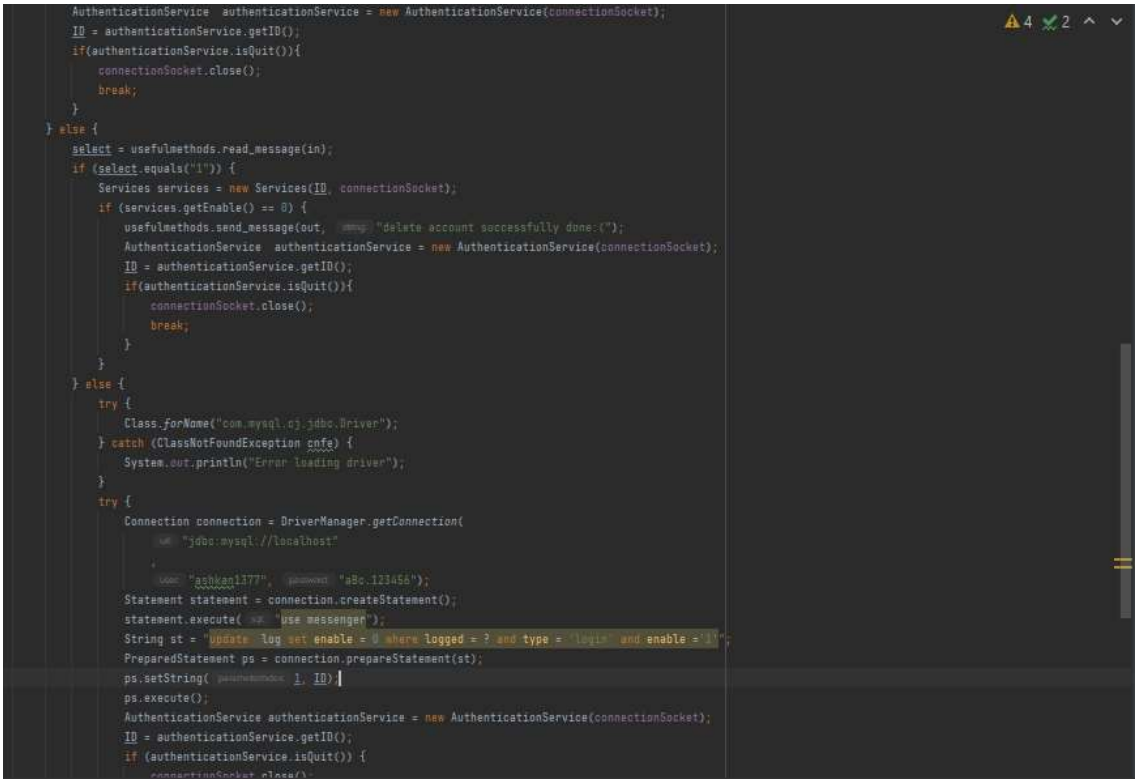
توضیحات: دراین متد، با استفاده از جدول user آیدی همه کاربرانی که با کلمه وارد شده از سمت کاربرشروع می شوند ، نمایش داده می شود.

delete_account



توضیحات: دراین متد، تاپل هایی از جداول block، friendship، recoverpasswordlimitation و request که آیدی کاربر درآن ها است ، پاک می شوند . درجدول user ، شماره همراه، ایمیل و آیدی به ترتیب به deleted ، deleted و deleted + نام کاربری کاربر تغییر می یابند.

ClientHandler Class



```
AuthenticationService  authenticationService = new AuthenticationService(connectionSocket);
ID = authenticationService.getID();
if(authenticationService.isQuit()){
    connectionSocket.close();
    break;
}
} else {
select = usefulmethods.read_message(in);
if (select.equals("1")) {
    Services services = new Services(ID, connectionSocket);
    if (services.getEnable() == 0) {
        usefulmethods.send_message(out,  msg: "delete account successfully done:(");
        AuthenticationService  authenticationService = new AuthenticationService(connectionSocket);
        ID = authenticationService.getID();
        if(authenticationService.isQuit()){
            connectionSocket.close();
            break;
        }
    }
} else {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch (ClassNotFoundException cnfe) {
        System.out.println("Error loading driver");
    }
    try {
        Connection connection = DriverManager.getConnection(
             url: "jdbc:mysql://localhost"
             ,
             user: "ashkan1377",  password: "aBc.123456");
        Statement statement = connection.createStatement();
        statement.execute( sql: "use messenger");
        String st = "update log set enable = 0 where logged = ? and type = 'login' and enable = '1'";
        PreparedStatement ps = connection.prepareStatement(st);
        ps.setString( parameterIndex: 1, ID);
        ps.execute();
        AuthenticationService authenticationService = new AuthenticationService(connectionSocket);
        ID = authenticationService.getID();
        if (authenticationService.isQuit()) {
            connectionSocket.close();
```

توضیحات: این کلاس یک متد run دارد که درابتدا هرکاربر را به یک شی از کلاس AuthenticationService ارجاع می دهد. وقتی کاربر رفت و ثبت نام کرد ووارد حساب کاربری اش شد، اورا به یک شی از کلاس services ارجاع می دهد.
ودرصورتی که یا حساب کاربری اش را حذف کند یا از حساب کاربری اش خارج شود، به یک شی از کلاس AuthenticationService ارجاع می دهد.

شمای پایگاه داده پروژه