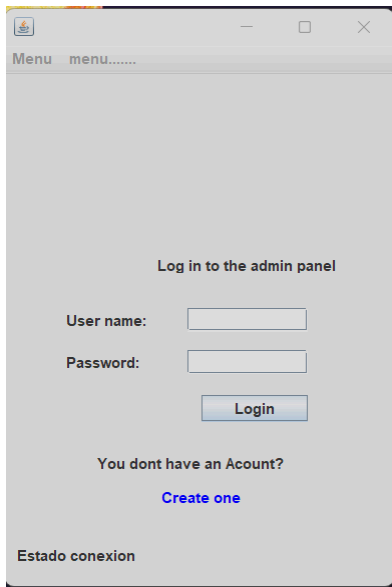


Entrega - Funcionalidad de prestamos

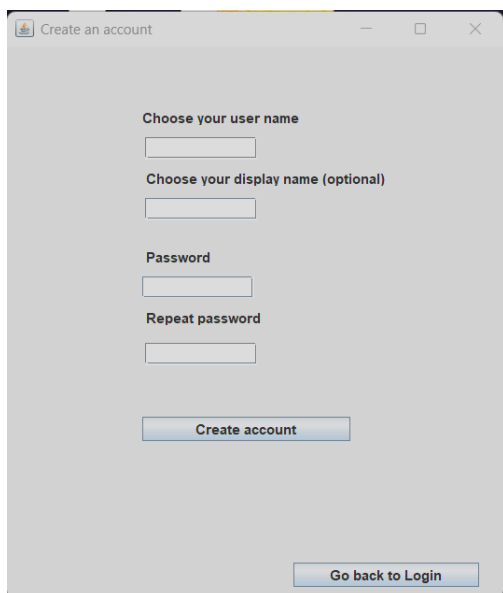
Hasta que no se inicia sesión, no se habilita el menú del chat:



The screenshot shows a web application window titled 'Menu menu.....'. Inside, there is a section titled 'Log in to the admin panel'. It contains two input fields: 'User name:' and 'Password:'. Below these is a 'Login' button. Underneath the login button, there is a link that says 'You dont have an Account?' followed by a blue link 'Create one'. At the bottom left, there is a label 'Estado conexion'.

NUEVO:

Si en login clickas en 'Create one', te dejará crear otro usuario, lo insertará en la base de datos, y le dará sus respectivos permisos como usuario normal (leer y editar lo suyo, solo borrar lo suyo), habilitando escapado en los textbox:



The screenshot shows a web application window titled 'Create an account'. It contains four input fields: 'Choose your user name', 'Choose your display name (optional)', 'Password', and 'Repeat password'. Below these is a 'Create account' button. At the bottom right, there is a link 'Go back to Login'.

```
tfPassword1.addKeyListener(new KeyAdapter() {  
    @Override  
    public void keyReleased(KeyEvent e) {  
  
        char[] passwordChars1 = tfPassword1.getPassword();  
        String password1 = new String(passwordChars1);  
        char[] password = password1.toCharArray();  
        System.out.println("Password1= " + password1);  
  
        char[] caracteresAEscapar = {  
            '\'', '\"', '\\', '\0', ';', '-', '/', '%', '_', '$'  
        };  
  
        // Verifica si el password no está vacío  
        if (password.length > 0) {  
            // Comprueba el último carácter del password  
            char ultimoCaracter = password[password.length-1];  
            System.out.println("ultimoCaracter= " + ultimoCaracter);  
  
            for (char caracterProhibido : caracteresAEscapar) { //Funcion de escapado de datos para evitar inyeccion SQL  
                if (ultimoCaracter == caracterProhibido) {  
                    System.out.println("Caracter prohibido");  
                    // Elimina el último carácter  
                    password = Arrays.copyOf(password, password.length - 1);  
                    password1 = new String(password);  
                    tfPassword1.setText(password1);  
                    lblWrongCharacter.setText("Invalid character: "+ultimoCaracter);  
                    lblWrongCharacter.setVisible(true);  
                }  
            }  
        }  
    }  
});
```

Con este código, si el usuario inserta una serie de caracteres prohibidos, el programa los borrará, y le informará al usuario si hay algun error.

Two screenshots of the 'Create an account' form. The left screenshot shows validation errors: 'You must enter a user name' and 'You must enter a password'. The right screenshot shows an error: 'Invalid character: \''.

Si al registrarte clickas en 'Profile', dejará editar los datos del usuario:

A screenshot of the user profile menu. The menu is open, showing options: 'Menu', 'Database', 'Chat', and 'Profile'. 'Profile' is selected and highlighted.

No deja cambiar el nombre del usuario, porque hará el Update de SQL con eso:

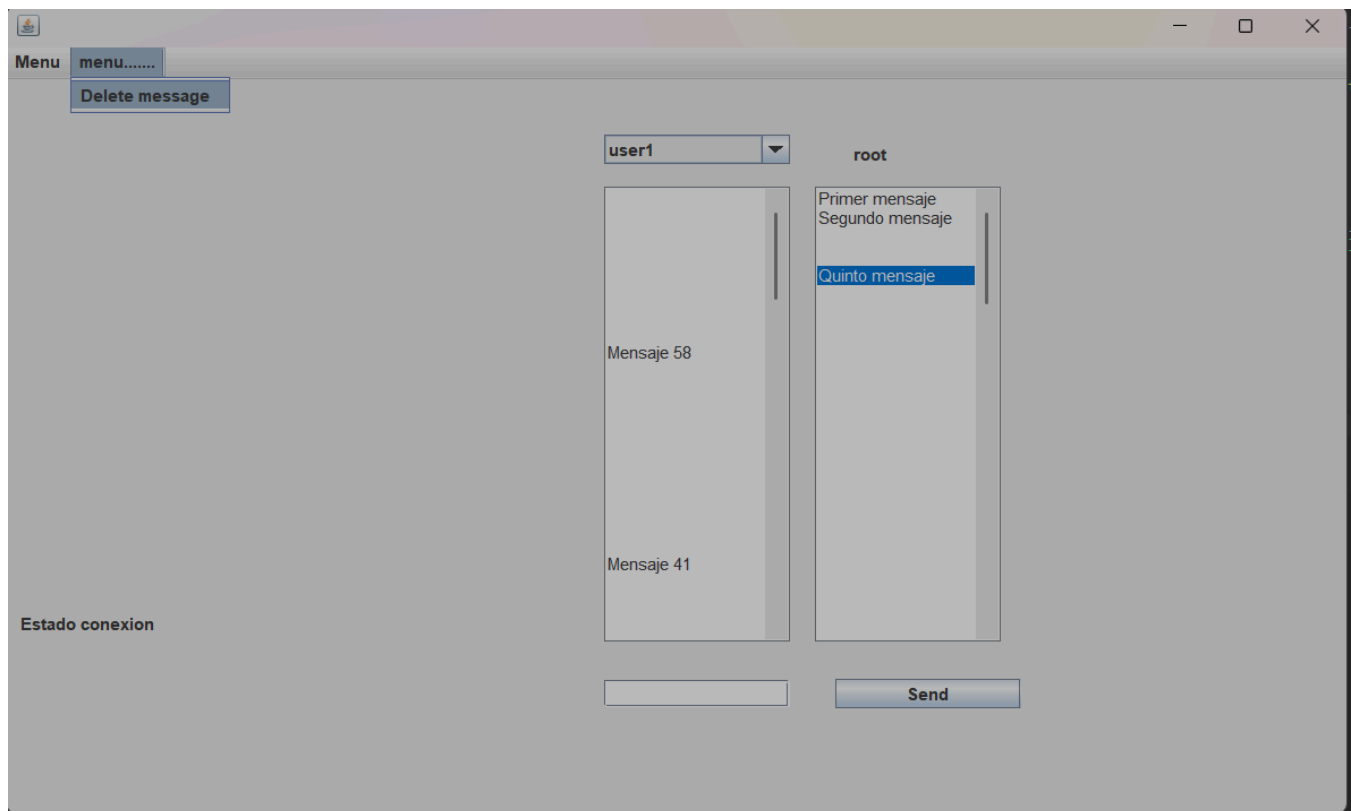
A screenshot of the user profile edit form. The form contains the following fields and buttons:

- Image:** A placeholder image box.
- User name:** A text input field containing 'root'.
- Display name:** A text input field containing 'root'.
- Location:** A text input field containing 'c/root'.
- Description:** A text area containing 'root'.
- Password:** A text input field containing 'root'.
- Buttons:** 'Go back', 'Undo changes', and 'Save'.

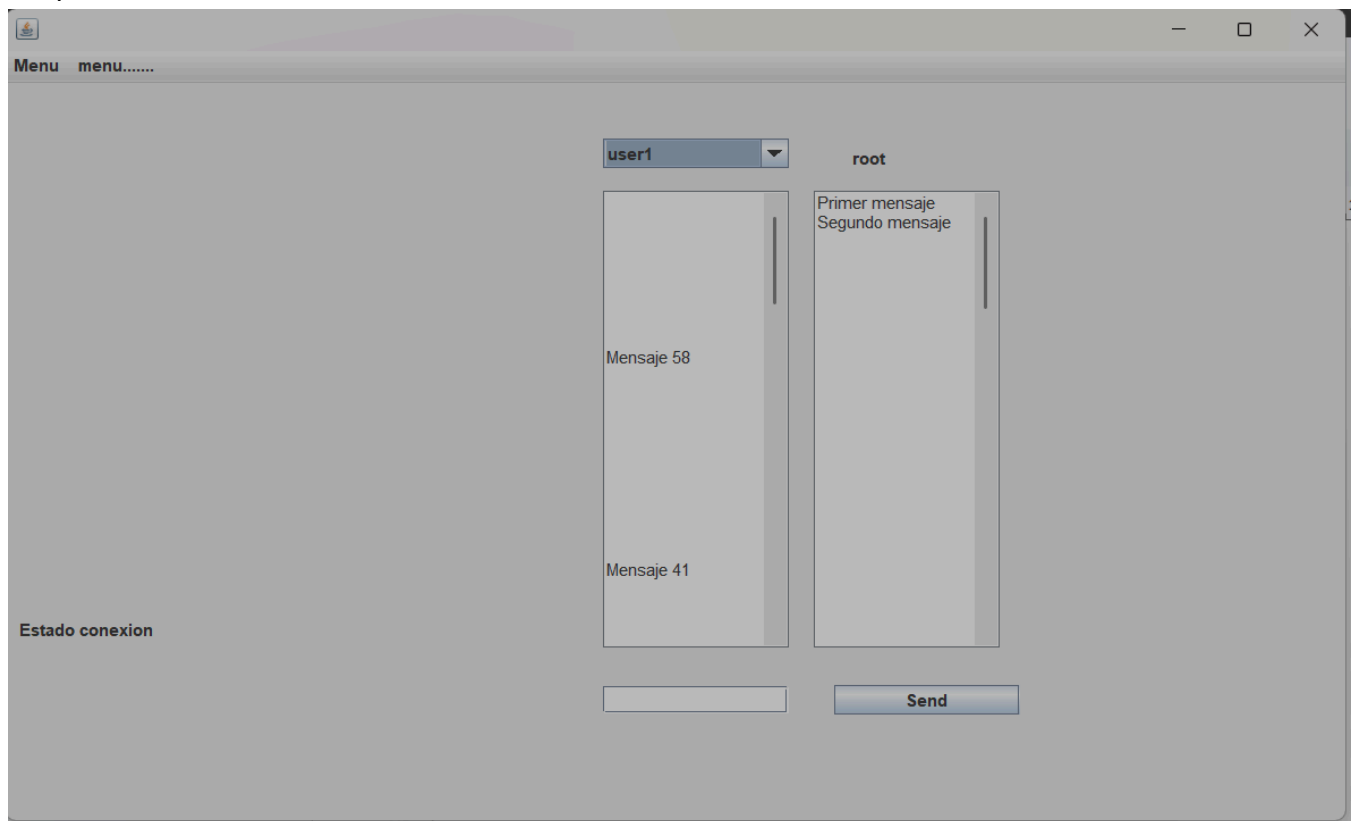
He usado este código para actualizar el usuario, usando PreparedStatements y un try catch con resources, para hacer la conexión mas segura:

```
String sql = "UPDATE users SET user_name = ?, user_password = ?, display_name = ?, image = ?, user_description = ?, location = ?  
WHERE user_name = ?";  
  
try(Connection con = DriverManager.getConnection("jdbc:mysql://localhost/mydb",userName,"");  
    Statement stmt = con.createStatement();  
){  
  
    PreparedStatement pst = con.prepareStatement(sql);  
  
    userName = tfUsername.getText();  
    displayName = tfDisplayName.getText();  
    location = tfLocation.getText();  
    password = BCrypt.withDefaults().hashToString(12, tfPassword.getText().toCharArray());  
    description = taDescription.getText();  
  
    pst.setString(1, userName); //todavía no se podrá cambiar el nombre de usuario;  
    pst.setString(2, password);  
    pst.setString(3, displayName);  
    pst.setString(4, imagePath);  
    pst.setString(5, description);  
    pst.setString(6, location);  
    pst.setString(7, userName);  
  
    int rs = pst.executeUpdate();  
  
    if(rs==1) {  
        JOptionPane.showMessageDialog(null, "Datos actualizados correctamente.",  
"Actualizar datos", JOptionPane.INFORMATION_MESSAGE);  
    }else {  
        JOptionPane.showMessageDialog(null, "Error al actualizar datos.", "Actualizar  
datos", JOptionPane.ERROR_MESSAGE);  
    }  
}
```

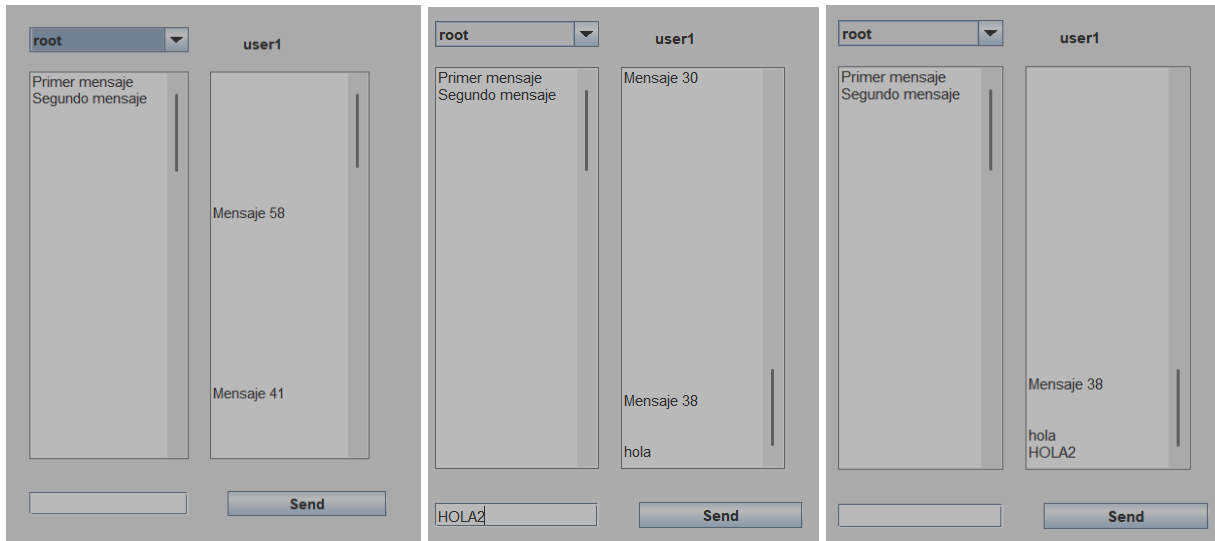
Ahora, deja borrar mensajes, y los borra de la BBDD también, con delete, try with resources, y PreparedStatements. Antes de borrarlo:



Despues:



Además, si iniciamos sesión con user1 y vamos al chat con root, nos mostrará adecuadamente los chats, cada uno en su lugar, y nos deja mandar mensajes, insertándolo en la BBDD:



Si se intenta borrar un mensaje que no es el tuyo, saltará este trigger de la base de datos MySQL:

```

95 DELIMITER $$
96 • CREATE TRIGGER delete_just_my_messages
97 BEFORE DELETE ON messages FOR EACH ROW
98 BEGIN
99
100 DECLARE userName VARCHAR(255);
101 DECLARE usernameLength INTEGER; #coge el usuario registrado (todos tienen @localhost)
102 SET usernameLength = length(CURRENT_USER) -10; #Resto longitud de @localhost
103
104 SET userName = LEFT(CURRENT_USER(),usernameLength); #Coge todos los caracteres del usuario y le resta @localhost
105
106 IF (userName != OLD.sender && OLD.sender != 'root') THEN #Solo deja borrar mensajes ajenos a 'root'
107 SIGNAL SQLSTATE '45000'
108 SET MESSAGE_TEXT="Solo puedes borrar tus mensajes";
109 END IF;
110
111 END$$
112
113 DELIMITER ;

```

ADICIONAL: Uso la librería BCrypt para encriptar todas las contraseñas, y almacenar las contraseñas encriptadas e ilegibles en la base de datos:

```

String passwordBCrypt = BCrypt.withDefaults().hashToString(12, password1.toCharArray());
pst.setString(2, passwordBCrypt);

```

He introducido lo mismo en user_name y user_password, pero la password se ha transformado:

	id	user_name	user_password
▶	1	root	\$2a\$12\$Tso.azDKzCh.K.iO0TEjl.glf9EKdIXCcH.4Q6PsSb42ER0geUTq
	2	user1	\$2a\$12\$2F7ST.lzM9VgXDTPLhSLWuDmCAUsdh55KNIT3AYGf0WrZT8Fqe0.
	9	hash	\$2a\$12\$ZlUJjJncpOqXVfRucvCcucY0cNMhu/IwQOCemi9hXnSWIIGSlkZK
	10	userContraseñauser1	\$2a\$12\$flo.oQdeABv8dyyZFoTjbOTgvr.tzoK9kfHn2a3sdULa8YnXSNCku
	12	user3	\$2a\$12\$MewC3KM.zpme4qHsGUCCJeGRRWqNTWBzy8pJLQHfwH2muAlU3oGyS
★	NULL	NULL	NULL

Códigos usados:

```
package Proyecto;
```

```
import java.awt.EventQueue;
```

```
import javax.swing.JFrame;  
import javax.swing.JPanel;  
import javax.swing.border.EmptyBorder;
```

```
import at.favre.lib.crypto.bcrypt.BCrypt;  
import at.favre.lib.bytes.Bytes;
```

```
import javax.swing.JButton;  
import java.awt.event.ActionListener;  
import java.awt.event.ActionEvent;  
import javax.swing.JMenuBar;  
import javax.swing.JMenu;  
import javax.swing.JMenuItem;  
import javax.swing.JOptionPane;  
import javax.swing.JTextField;  
import javax.swing.JLabel;  
import javax.swing.JList;  
import java.awt.List;
```

```
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.ResultSet;  
import java.sql.Statement;  
import java.sql.SQLException;  
import javax.swing.JComboBox;  
import java.awt.event.ItemListener;  
import java.awt.event.MouseEvent;
```

```

import java.awt.event.ItemEvent;
import javax.swing.JPasswordField;
import java.awt.Label;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

public class mainScreen extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField tfUserName;
    private JTextField tfPassword1;
    private JLabel lblUser, lblPassword, lblLogin, lblSender, lblAccount, lblCreateOne;
    private JButton btnLogin, btnSend;

    public int messageCount= 5;

    private JTextField tfNewMessage;
    private JComboBox<String> cbChatOther;
    private List chatMine, chatOther;

    private String userName = "";
    private String password="";
    private String displayName="";
    private String userDescription="";
    private String location="";
    private String imagePath="";
    private JPasswordField tfPassword;
    JMenu menuChatOptions;

    //200 filas 2 columnas
    private int[][] messages_Idsl = new int[300][2]; //Columna uno almacena donde esta cada mensaje(ira
    del 0 al maximo sin saltarse nada)
    private int[][] messages_IdSD = new int[300][2]; //Columna 2 almacena id del
    mensaje. asi al borrar por el index coge el id

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    mainScreen frame = new mainScreen();
                    frame.setVisible(true);
                } catch (Exception e) {

```

```

                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public mainScreen() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 325, 470); //setBounds(100, 100, 300, 500); setBounds(100, 100, 300, 500);

    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(null);

    JMenuBar menuBar = new JMenuBar();
    setJMenuBar(menuBar);

    JMenu mainMenu = new JMenu("Menu");
    menuBar.add(mainMenu);
    mainMenu.setEnabled(false);

    JMenuItem miDatabase = new JMenuItem("Database");
    mainMenu.add(miDatabase);

    chatOther = new List();
    chatOther.setBounds(422, 76, 131, 322);
    contentPane.add(chatOther);

    chatMine = new List();
    chatMine.setBounds(571, 76, 131, 322);
    contentPane.add(chatMine);

    cbChatOther = new JComboBox<>();
    cbChatOther.addItemListener(new ItemListener() {
        @Override
        public void itemStateChanged(ItemEvent e) {
            if (e.getStateChange() == ItemEvent.SELECTED) {
                String selectedItem = (String) cbChatOther.getSelectedItem();
                System.out.println("Item seleccionado: " + selectedItem);

                chatOther.removeAll();
                chatMine.removeAll();
            }
        }
    });
}

```



```

int chatLongitude = 0;

String otherUser = cbChatOther.getSelectedItemAt().toString();
System.out.println(otherUser);
try {

    Connection
conexion=DriverManager.getConnection("jdbc:mysql://localhost/mydb","root","");
    Statement sentencia= conexion.createStatement();
    ResultSet registro=sentencia.executeQuery("SELECT * FROM
messages WHERE sender = '"+userName+"' AND receiver = '"+otherUser+"'");

    int jump=0;
    int prevMes = 0;
    while(registro.next()) {

        System.out.println(registro.getString("content"));

        if(registro.getInt("message_id") == prevMes + 1) {
            chatMine.add(registro.getString("content"));
            messages_IdsD[chatLongitude][0] = chatLongitude;
            messages_IdsD[chatLongitude][1] =

registro.getInt("message_id");

            chatLongitude++;

        }else {

            jump = registro.getInt("message_id") - prevMes;
            for(int x =0; x<=jump-2;x++) {
                chatMine.add(" ");
                messages_IdsD[chatLongitude][0] =

chatLongitude;

                messages_IdsD[chatLongitude][1] = 0;
                chatLongitude++;

            }

            chatMine.add(registro.getString("content"));
            messages_IdsD[chatLongitude][0] = chatLongitude;
            messages_IdsD[chatLongitude][1] =

registro.getInt("message_id");

            chatLongitude++;

        }

        prevMes = registro.getInt("message_id");

    }
}

```

```

        prevMes = 0;
        jump=0;

        Statement sentencia2= conexion.createStatement();
        ResultSet registro2=sentencia2.executeQuery("SELECT * FROM
messages WHERE sender = '"+otherUser+"' AND receiver = '"+userName+"'");
        chatLongitude = 0;
        while(registro2.next()) {

            if(registro2.getInt("message_id") == prevMes + 1) {
                chatOther.add(registro2.getString("content"));
                messages_Idsl[chatLongitude][0] = chatLongitude;
                messages_Idsl[chatLongitude][1] =

registro2.getInt("message_id");

                chatLongitude++;
            }else {

                jump = registro2.getInt("message_id") - prevMes;
                for(int x =0; x<=jump-2;x++) {
                    chatOther.add(" ");
                    messages_Idsl[chatLongitude][0] =

chatLongitude;

                    messages_Idsl[chatLongitude][1] = 0;
                    chatLongitude++;
                }

                chatOther.add(registro2.getString("content"));
                messages_Idsl[chatLongitude][0] = chatLongitude;
                messages_Idsl[chatLongitude][1] =

registro2.getInt("message_id");

                chatLongitude++;
            }

            prevMes = registro2.getInt("message_id");
        }
        menuChatOptions.setEnabled(true);

        conexion.close();

    }catch(SQLException e1) {
        e1.printStackTrace();
    }

}
}

```

```

});

cbChatOther.setBounds(422, 39, 131, 21);
cbChatOther.setVisible(true);
contentPane.add(cbChatOther);

JMenuItem miChat = new JMenuItem("Chat");
miChat.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        cbChatOther.removeAllItems();
        cbChatOther.addItem("user1");
        cbChatOther.addItem("user2");
        cbChatOther.addItem("root");
        cbChatOther.addItem("user3");

        cbChatOther.setVisible(true);
        chatOther.setVisible(true);
        chatMine.setVisible(true);
        btnSend.setVisible(true);
        lblSender.setVisible(true);
        tfNewMessage.setVisible(true);
        menuChatOptions.setEnabled(false);

    }
});
mainMenu.add(miChat);

JMenuItem miProfile = new JMenuItem("Profile");
miProfile.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        Profile profile = new Profile();
        profile.setSize(811, 345);
        profile.setLocationRelativeTo(null);
        profile.setData(userName, password, displayName, location, userDescription, imagePath);
        profile.setVisible(true);
        dispose();
    }
});
mainMenu.add(miProfile);

tfUserName = new JTextField();
tfUserName.setToolTipText("login");
tfUserName.setBounds(146, 187, 96, 19);
contentPane.add(tfUserName);
tfUserName.setColumns(10);

```

```

tfPassword = new JPasswordField();
tfPassword.setBounds(146, 221, 96, 19);
tfPassword.setColumns(10);
contentPane.add(tfPassword);

JLabel lblConnection = new JLabel("Estado conexion");
lblConnection.setBounds(10, 359, 156, 51);
contentPane.add(lblConnection);

btnLogin = new JButton("Login");
btnLogin.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        userName = tfUserName.getText().toString();
        char[] passwordChars = tfPassword.getPassword();
        password = new String(passwordChars);

        try (Connection
conexion=DriverManager.getConnection("jdbc:mysql://localhost/mydb",userName,"");
        Statement sentencia= conexion.createStatement();
        ResultSet registro=sentencia.executeQuery("SELECT *
FROM users WHERE user_name = '"+userName+"';")){
            String passwordFromDb="";
            while(registro.next()) {
                passwordFromDb= registro.getNString("user_password");
                displayName = registro.getNString("display_name");
                location=registro.getNString("location");
                userDescription=registro.getNString("user_description");
                imagePath = registro.getNString("image");

            }

            conexion.close();

            boolean passwordMatch =
BCrypt.verifier().verify(password.toCharArray(), passwordFromDb.verified);

            if(passwordMatch) {
                System.out.println("Login correcto");

                tfPassword.setVisible(false);
                tfUserName.setVisible(false);
                lblUser.setVisible(false);
                lblPassword.setVisible(false);
                lblLogin.setVisible(false);
                btnLogin.setVisible(false);
            }
        }
    }
});

```

```

        lblCreateOne.setVisible(false);
        lblAccount.setVisible(false);

        mainMenu.setEnabled(true);
        setBounds(100, 100, 960, 579);
        lblSender.setText(userName);
        JOptionPane.showMessageDialog(null, "Conexion correcta",
"Conexion", JOptionPane.INFORMATION_MESSAGE);
    }else {
        JOptionPane.showMessageDialog(null, "Conexion
incorrecta", "Conexion", JOptionPane.WARNING_MESSAGE);
    }

    }catch(SQLException SQLe) {

        SQLe.printStackTrace();
        lblConnection.setText("No conectado");
    }
}

});
btnLogin.setBounds(157, 257, 85, 21);
contentPane.add(btnLogin);

lblUser = new JLabel("User name:");
lblUser.setBounds(49, 190, 69, 13);
contentPane.add(lblUser);

lblPassword = new JLabel("Password:");
lblPassword.setBounds(49, 224, 69, 13);
contentPane.add(lblPassword);

lblLogin = new JLabel("Log in to the admin panel");
lblLogin.setBounds(122, 146, 196, 13);
contentPane.add(lblLogin);

tfNewMessage = new JTextField();
tfNewMessage.setBounds(422, 425, 131, 19);
contentPane.add(tfNewMessage);
tfNewMessage.setColumns(10);

btnSend = new JButton("Send");
btnSend.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String sender =userName;
        String receiver=cbChatOther.getSelectedItem().toString();
        String content = tfNewMessage.getText().toString();
        int user_id = 1;

```

```

        if(!(content.equals(""))) {

            chatMine.add(content);

            tfNewMessage.setText("");

            try (Connection
conexion=DriverManager.getConnection("jdbc:mysql://localhost/mydb",userName,"");){

                Statement stmt = conexion.createStatement();
                int rs = stmt.executeUpdate("INSERT INTO messages (user_id,
content, sender, receiver) VALUES (" +user_id+", ""+content+", ""+sender+", ""+receiver+"");");
                if(rs>0) {
                    System.out.println("Insertado");
                }
            } catch (SQLException e1) {

                e1.printStackTrace();
            }

            System.out.println("Mensaje enviado");

            messageCount++;
        }else {
            System.out.println("Mensaje no enviado");
        }

    }

});
btnSend.setBounds(585, 424, 131, 21);
contentPane.add(btnSend);

lblSender = new JLabel("root");
lblSender.setBounds(598, 46, 45, 13);
contentPane.add(lblSender);

lblAccount = new JLabel("You dont have an Account?");
lblAccount.setBounds(74, 301, 168, 21);
contentPane.add(lblAccount);

lblCreateOne = new JLabel("Create one");
lblCreateOne.setForeground(Color.BLUE);
lblCreateOne.setBounds(125, 328, 102, 21);
lblCreateOne.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));

lblCreateOne.addMouseListener(new MouseAdapter() {
    @Override

```

```

public void mouseClicked(MouseEvent e) {
    // Acción al hacer clic
    System.out.println("Label clickeado");
    CreateAccount ca = new CreateAccount();
    ca.setSize(440, 510);
    ca.setLocationRelativeTo(null);
    ca.setVisible(true);
    dispose();
}
});
contentPane.add(lblCreateOne);

menuChatOptions = new JMenu("menu.....");
menuChatOptions.setBounds(147, 395, 115, 26);
menuChatOptions.setEnabled(false);
menuBar.add(menuChatOptions);

JMenuItem miDeleteSelectedMessage = new JMenuItem("Delete message");
miDeleteSelectedMessage.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        int messageContent = chatMine.getSelectedIndex();
        int messageId = messages_IdsD[messageContent][1];
        System.out.println(messageId);

        String sql = "DELETE from messages where message_id = '"+messageId+"'";

        try(Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/mydb",userName,"");
            Statement stmt = con.createStatement();){

            int rs = stmt.executeUpdate(sql);
            if(rs>=1) {

                //loadChat
                System.out.println("Mensaje borrado");
            }

        }catch(SQLException sqle) {
            sqle.printStackTrace();
        }

    }
});
menuChatOptions.add(miDeleteSelectedMessage);

```

```

        lblSender.setVisible(false);
        btnSend.setVisible(false);
        tfNewMessage.setVisible(false);

        cbChatOther.setVisible(false);
        chatOther.setVisible(false);
        chatMine.setVisible(false);

    }
}

```

```

package Proyecto;

```

```

import java.awt.EventQueue;

```

```

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

```

```

import at.favre.lib.crypto.bcrypt.BCrypt;

```

```

import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JButton;
import javax.swing.JTextField;
import javax.swing.JTextArea;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.Statement;
import java.awt.event.ActionEvent;

```

```

public class Profile extends JFrame {

```

```

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField tfUsername, tfPassword, tfDisplayName;
    private JTextArea taDescription;
    private JTextField tfLocation;
    private String userName, displayName, password, location, imagePath, description;

```



```

/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Profile frame = new Profile();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public Profile() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 811, 345);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(null);

    JLabel lblNewLabel = new JLabel("Image");
    lblNewLabel.setBounds(48, 33, 45, 13);
    contentPane.add(lblNewLabel);

    JLabel lblNewLabel_1 = new JLabel("User name");
    lblNewLabel_1.setBounds(48, 117, 85, 13);
    contentPane.add(lblNewLabel_1);

    JLabel lblNewLabel_2 = new JLabel("Display name");
    lblNewLabel_2.setBounds(48, 168, 85, 13);
    contentPane.add(lblNewLabel_2);

    JLabel lblNewLabel_3 = new JLabel("Description");
    lblNewLabel_3.setBounds(375, 33, 85, 13);
    contentPane.add(lblNewLabel_3);

    JLabel lblNewLabel_4 = new JLabel("Password");
    lblNewLabel_4.setBounds(375, 168, 85, 13);
    contentPane.add(lblNewLabel_4);
}

```

```

JButton btnGoBack = new JButton("Go back");
btnGoBack.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        mainScreen ms = new mainScreen();
        ms.setSize(325, 470);
        ms.setLocationRelativeTo(null);
        ms.setVisible(true);
        dispose();

    }
});
btnGoBack.setBounds(683, 29, 85, 21);
contentPane.add(btnGoBack);

tfUsername = new JTextField();
tfUsername.setEditable(false);
tfUsername.setBounds(182, 114, 96, 19);
contentPane.add(tfUsername);
tfUsername.setColumns(10);

tfDisplayName = new JTextField();
tfDisplayName.setBounds(182, 162, 96, 19);
contentPane.add(tfDisplayName);
tfDisplayName.setColumns(10);

tfPassword = new JTextField();
tfPassword.setBounds(470, 168, 131, 19);
contentPane.add(tfPassword);
tfPassword.setColumns(10);

JButton btnSave = new JButton("Save");
btnSave.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String sql = "UPDATE users SET user_name = ?, user_password = ?,
display_name = ?, image = ?, user_description = ?, location = ? WHERE user_name = ? ";

        try(Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/mydb",userName,"");
            Statement stmt = con.createStatement();
        ){

            PreparedStatement pst = con.prepareStatement(sql);

            userName = tfUsername.getText();
            displayName = tfDisplayName.getText();

```

```

        location = tfLocation.getText();
        password = BCrypt.withDefaults().hashToString(12,
tfPassword.getText().toCharArray());
        description = taDescription.getText();

        pst.setString(1, userName); //todavia no se podra cambiar el nombre de
usuario;

        pst.setString(2, password);
        pst.setString(3, displayName);
        pst.setString(4, imagePath);
        pst.setString(5, description);
        pst.setString(6, location);
        pst.setString(7, userName);

        int rs = pst.executeUpdate();

        if(rs==1) {
            JOptionPane.showMessageDialog(null, "Datos actualizados
correctamente.", "Actualizar datos", JOptionPane.INFORMATION_MESSAGE);
        }else {
            JOptionPane.showMessageDialog(null, "Error al actualizar datos.",
"Actualizar datos", JOptionPane.ERROR_MESSAGE);
        }

    }catch(SQLException sqle) {
        sqle.printStackTrace();
    }

}

});
btnSave.setBounds(516, 260, 85, 21);
contentPane.add(btnSave);

JButton btnUndo = new JButton("Undo changes");
btnUndo.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        int okCancel = JOptionPane.showConfirmDialog(null, "¿Seguro que quieres
restaurarlos?", "Deshacer cambios", JOptionPane.OK_CANCEL_OPTION,
JOptionPane.QUESTION_MESSAGE );

        if(okCancel == JOptionPane.OK_OPTION) {

            tfUsername.setText(userName);
            tfDisplayName.setText(displayName);
            tfLocation.setText(location);
            tfPassword.setText(password);
            taDescription.setText(description);

```

```

        }

    }

});
btnUndo.setBounds(360, 260, 117, 21);
contentPane.add(btnUndo);

taDescription = new JTextArea();
taDescription.setBounds(470, 27, 131, 102);
contentPane.add(taDescription);

JTextArea textArea_1 = new JTextArea();
textArea_1.setBounds(170, 27, 45, 43);
contentPane.add(textArea_1);

JLabel lblNewLabel_5 = new JLabel("Location");
lblNewLabel_5.setBounds(48, 220, 85, 13);
contentPane.add(lblNewLabel_5);

tfLocation = new JTextField();
tfLocation.setBounds(182, 217, 96, 19);
contentPane.add(tfLocation);
tfLocation.setColumns(10);

}

public void setData(String prevUsername, String prevPassword, String prevDisplayname, String
prevLocation, String prevUserdescription, String prevImagepath) {

    tfUsername.setText(prevUsername);
    tfPassword.setText(prevPassword);
    tfDisplayName.setText(prevDisplayname);
    taDescription.setText(prevUserdescription);
    tfLocation.setText(prevLocation);

    userName = prevUsername;
    password = prevPassword;
    displayName = prevDisplayname; //Guardamos en la clase los antiguos valores
    location = prevLocation;
    imagePath = prevImagepath;
    description = prevUserdescription;

}
}

package Proyecto;

```

```

//Meter longitud minima de contraseña

import java.awt.EventQueue;

import at.favre.lib.crypto.bcrypt.BCrypt;
import at.favre.lib.bytes.Bytes;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import com.mysql.cj.jdbc.exceptions.CommunicationsException;
import com.mysql.cj.jdbc.exceptions.MySQLDataTruncation;

import javax.swing.JTextField;
import javax.swing.JLabel;
import javax.swing.JPasswordField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Color;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Arrays;
import java.sql.SQLException;
import java.sql.SQLIntegrityConstraintViolationException;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;

public class CreateAccount extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField tfUserName;
    private JTextField tfDisplayName;
    private JPasswordField tfPassword2;
    private JPasswordField tfPassword1;
    private JLabel wrongPassword, wrongPassword2, lblwrongUser, lblUsernameGotten,
    lblWrongCharacter;

    /**
     * Launch the application.

```

```

*/
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                CreateAccount frame = new CreateAccount();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

```

```

/**

```

```

 * Create the frame.

```

```

*/

```

```

public CreateAccount() {
    setTitle("Create an account");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 440, 510);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(null);

    tfUserName = new JTextField();
    tfUserName.setBounds(119, 78, 96, 19);
    contentPane.add(tfUserName);
    tfUserName.setColumns(10);

    JLabel lblUsername = new JLabel("Choose your user name");
    lblUsername.setBounds(116, 55, 179, 13);
    contentPane.add(lblUsername);

    JLabel lblNewLabel_1 = new JLabel("Password");
    lblNewLabel_1.setBounds(119, 174, 96, 13);
    contentPane.add(lblNewLabel_1);

    JLabel lblNewLabel_2 = new JLabel("Repeat password");
    lblNewLabel_2.setBounds(119, 226, 143, 13);
    contentPane.add(lblNewLabel_2);

    tfDisplayName = new JTextField();
    tfDisplayName.setBounds(119, 130, 96, 19);
    contentPane.add(tfDisplayName);
    tfDisplayName.setColumns(10);
}

```

```
JLabel lblNewLabel = new JLabel("Choose your display name (optional)");
lblNewLabel.setBounds(119, 107, 265, 13);
contentPane.add(lblNewLabel);
```

```
tfPassword2 = new JPasswordField();
tfPassword2.setBounds(119, 254, 96, 19);
contentPane.add(tfPassword2);
```

```
tfPassword1 = new JPasswordField();
tfPassword1.addKeyListener(new KeyAdapter() {
    @Override
    public void keyReleased(KeyEvent e) {
```

```
        char[] passwordChars1 = tfPassword1.getPassword();
        String password1 = new String(passwordChars1);
        char[] password = password1.toCharArray();
        System.out.println("Password1= " + password1);
```

```
        char[] caracteresAEscapar = {
            '\", \"'\', '\0', ';', '-', '/', '%', '_', '$'
        };
```

```
        // Verifica si el password no está vacío
        if (password.length > 0) {
            // Comprueba el último carácter del password
            char ultimoCaracter = password[password.length-1];
            System.out.println("ultimoCaracter= "+ ultimoCaracter);
```

```
            for (char caracterProhibido : caracteresAEscapar) { //Funcion de escapado de datos
                para evitar inyeccion SQL
```

```
                if (ultimoCaracter == caracterProhibido) {
                    System.out.println("Caracter prohibido");
                    // Elimina el último carácter
                    password = Arrays.copyOf(password, password.length - 1);
                    password1 = new String(password);
                    tfPassword1.setText(password1);
                    lblWrongCharacter.setText("Invalid character: "+ultimoCaracter);
                    lblWrongCharacter.setVisible(true);
```

```
                }else {
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
});
```

```

tfPassword1.setBounds(116, 197, 96, 19);
contentPane.add(tfPassword1);

JButton btnCreateAccount = new JButton("Create account");
btnCreateAccount.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String userName = tfUserName.getText();
        String displayName = tfDisplayName.getText();

        char[] passwordChars1 = tfPassword1.getPassword();
        String password1 = new String(passwordChars1);

        char[] passwordChars2 = tfPassword2.getPassword();
        String password2 = new String(passwordChars2);

        System.out.println(userName+ " : "+displayName + " : " + password1 + " : " +
password2);

        Boolean usercorrect = !(userName.equals("")), passwordcorrect =
!(password1.equals("")), passwordCorrect2 = password1.equals(password2);

        if (!passwordcorrect) {
            wrongPassword2.setVisible(true);
        }else {
            wrongPassword2.setVisible(false);
            repaint();
        }

        if(!passwordCorrect2) {
            wrongPassword.setVisible(true);
        }else {
            wrongPassword.setVisible(false);
            repaint();
        }

        if(!usercorrect) {
            lblwrongUser.setVisible(true);
        }else {
            lblwrongUser.setVisible(false);
            repaint();
        }

        String createUser = "CREATE USER '?'@'localhost' IDENTIFIED BY ";
        + "GRANT select, update, insert on mydb.messages TO
'?'@'localhost';"
        + "GRANT select, update on mydb.users TO '?'@'localhost';"
        + "FLUSH privileges;";

```



```

        if(usercorrect && passwordcorrect && passwordCorrect2) {

            try(Connection
conexion=DriverManager.getConnection("jdbc:mysql://localhost/mydb","root",""); //Este try es para
añadirlo a la tabla

                Statement sentencia= conexion.createStatement();{

                    PreparedStatement pst = conexion.prepareStatement("INSERT INTO
users (user_name, user_password, display_name) VALUES (?, ?, ?)");
                    pst.setString(1, userName);
                    // Para hashear una contraseña
                    String passwordBCrypt = BCrypt.withDefaults().hashToString(12,
password1.toCharArray());

                    pst.setString(2, passwordBCrypt);

                    if(displayName.equals("")) {
                        pst.setObject(3, null);
                    }else {
                        pst.setString(3, displayName);
                    }
                    int rs = pst.executeUpdate();

                    if(rs==1) {
                        System.out.println("Insertado en tabla");

                        // Crear usuario en MySQL y otorgar privilegios
                        String createUserSQL = String.format("CREATE USER '%s'@'localhost'
IDENTIFIED BY ''", userName);
                        String grantMessagesSQL = String.format("GRANT SELECT, UPDATE,
INSERT ON mydb.messages TO '%s'@'localhost'", userName);
                        String grantUsersSQL = String.format("GRANT SELECT, UPDATE ON
mydb.users TO '%s'@'localhost'", userName);

                        rs = sentencia.executeUpdate(createUserSQL);
                        rs = sentencia.executeUpdate(grantMessagesSQL);
                        rs = sentencia.executeUpdate(grantUsersSQL);
                        rs = sentencia.executeUpdate("FLUSH PRIVILEGES");

                    }else {System.out.println("Error al insertar");}

```

```

        }catch(SQLIntegrityConstraintViolationException icv) {
            lblUsernameGotten.setVisible(true);
        }catch(CommunicationsException CE) {
            setTitle("No connection");
        }catch(MysqlDataTruncation tr) {
            setTitle("Paasowrd too long");
        }catch(SQLException sqle) {
            sqle.printStackTrace();
        }
    }

}

});
btnCreateAccount.setBounds(116, 317, 179, 21);
contentPane.add(btnCreateAccount);

wrongPassword = new JLabel("The passwords must match");
wrongPassword.setForeground(Color.RED);
wrongPassword.setBounds(119, 283, 159, 13);
wrongPassword.setVisible(false);
contentPane.add(wrongPassword);

wrongPassword2 = new JLabel("You must enter a password");
wrongPassword2.setForeground(Color.RED);
wrongPassword2.setBounds(222, 200, 159, 13);
wrongPassword2.setVisible(false);
contentPane.add(wrongPassword2);

lblwrongUser = new JLabel("You must enter a user name");
lblwrongUser.setForeground(Color.RED);
lblwrongUser.setBounds(225, 81, 159, 13);
lblwrongUser.setVisible(false);
contentPane.add(lblwrongUser);

lblUsernameGotten = new JLabel("This user name already exists!");
lblUsernameGotten.setForeground(Color.RED);
lblUsernameGotten.setBounds(225, 81, 191, 13);
lblUsernameGotten.setVisible(false);
contentPane.add(lblUsernameGotten);

JButton btnLogin = new JButton("Go back to Login");
btnLogin.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

```

```
        mainScreen ms = new mainScreen();
        ms.setSize(325, 470);
        ms.setLocationRelativeTo(null);
        ms.setVisible(true);
        dispose();
    }
});
btnLogin.setBounds(246, 442, 159, 21);
contentPane.add(btnLogin);

lblWrongCharacter = new JLabel("Invalid character");
lblWrongCharacter.setForeground(Color.RED);
lblWrongCharacter.setBounds(222, 200, 130, 13);
lblWrongCharacter.setVisible(false);
contentPane.add(lblWrongCharacter);

// Para verificar una contraseña
//BCrypt.Result result = BCrypt.verifyer().verify(password.toCharArray(), bcryptHashString);
//boolean passwordMatches = result.verified;

}
}
```