

```

*
* @author Darian M. Martinez ESPE DCCO
*/
public class UserController {

    private final Gson gson = new GsonBuilder().setPrettyPrinting().create();

    private String secretKey = "SDevelopers";

    public User Validate(String username, String password){
        ArrayList<Object> objects;

        ArrayList<User> users = new ArrayList<>();
        objects = JsonManager.readObjects();
        User user = null;
        for (int i = 0; i < objects.size(); i++) {
            User userJson;
            String jUser = gson.toJson(objects.get(i));
            userJson = gson.fromJson(jUser, User.class);

            users.add(userJson);
        }
        for (int i = 0; i < users.size(); i++) {
            if(users.get(i).getUserName().equals(username) &&
                SecurityPassword.deecnode(secretKey,
users.get(i).getPassword()).equals(password)){
                user = users.get(i);
            }
        }
        return user;
    }
}
/**
*

```

```

* @author Darian M. Martinez ESPE DCCO
*/
public class JsonManager {

    static Gson gson = new GsonBuilder().setPrettyPrinting().create();

    public static void addToFile(User object) {
        ArrayList<User> users = new ArrayList<>();
        users.add(object);
        UserList userList = new UserList(users);

        try(FileWriter fw = new FileWriter("User1.json")){
            gson.toJson(userList, fw);
        }catch(IOException e){
            JOptionPane.showMessageDialog(null, "User not entered successfully..",
"FAILURE", JOptionPane.ERROR_MESSAGE);
        }
    }

    public static ArrayList<Object> readObjects() {
        JsonParser parser = new JsonParser();
        UserList userList = new UserList(new ArrayList<>());
        ArrayList<Object> objects = new ArrayList<>();
        try (Reader reader = new FileReader("User1.json")){
            userList = gson.fromJson(reader, UserList.class);
        } catch (IOException e) {
            JOptionPane.showMessageDialog(null, "File not Found....", "FAILURE",
JOptionPane.ERROR_MESSAGE);
        }

        for(User user : userList getUsers()){
            objects.add(user);
        }

        return objects;
    }
}
/**
 *
 * @author Darian M. Martinez ESPE DCCO
 */
public class SecurityPassword {

    public static String ecnode(String secretKey, String password) {
        String encryption = "";
        try {
            MessageDigest md5 = MessageDigest.getInstance("MD5");
            byte[] keyPassword = md5.digest(secretKey.getBytes("utf-8"));
            byte[] BytesKey = Arrays.copyOf(keyPassword, 24);
            SecretKey key = new SecretKeySpec(BytesKey, "DESede");

```

```

        Cipher cifrado = Cipher.getInstance("DESede");
        cifrado.init(Cipher.ENCRYPT_MODE, key);
        byte[] plainTextBytes = password.getBytes("utf-8");
        byte[] buf = cifrado.doFinal(plainTextBytes);
        byte[] base64Bytes = Base64.encodeBase64(buf);
        encryption = new String(base64Bytes);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Something went wrong...",
"FAILURE", JOptionPane.ERROR_MESSAGE);
    }
    return encryption;
}

public static String deecnode(String secretKey, String password){
    String decryption = "";
    try {
        byte[] message = Base64.decodeBase64(password.getBytes("utf-8"));
        MessageDigest md5 = MessageDigest.getInstance("MD5");
        byte[] digestOfPassword = md5.digest(secretKey.getBytes("utf-8"));
        byte[] keyBytes = Arrays.copyOf(digestOfPassword, 24);
        SecretKey key = new SecretKeySpec(keyBytes, "DESede");
        Cipher decipher = Cipher.getInstance("DESede");
        decipher.init(Cipher.DECRYPT_MODE, key);
        byte[] plainText = decipher.doFinal(message);
        decryption = new String(plainText, "UTF-8");

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(null, "Something went wrong",
"FAILURE", JOptionPane.ERROR_MESSAGE);
        }
        return decryption;
    }
}
/**
 *
 * @author Darian M. Martinez ESPE DCCO
 */
public class User implements Serializable{

    private String userName;
    private String password;

    public User(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }

    public User() {
    }
}

```

```

/**
 * @return the userName
 */
public String getUserName() {
    return userName;
}

/**
 * @param userName the userName to set
 */
public void setUserName(String userName) {
    this.userName = userName;
}

/**
 * @return the password
 */
public String getPassword() {
    return password;
}

/**
 * @param password the password to set
 */
public void setPassword(String password) {
    this.password = password;
}
}
/**
 *
 * @author Darian M. Martinez ESPE DCCO
 */
public class UserList implements Serializable{

    private ArrayList<User> users;

    public UserList(ArrayList<User> users) {
        this.users = users;
    }

    public ArrayList<User> getUsers() {
        return users;
    }

    public void setUsers(ArrayList<User> users) {
        this.users = users;
    }

}

```

```

public class FrmLogin extends javax.swing.JFrame {

    /**
     * Creates new form FrmLogin
     */
    public FrmLogin() {
        initComponents();
    }
    private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String userName = txtUsername.getText();
        String password = txtPassword.getText();
        UserController userC = new UserController();
        User user = userC.Validate(userName, password);

        if(user != null){
            FrmNewUser menu = new FrmNewUser();
            menu.setVisible(true);
            this.setVisible(false);
        }else {
            JOptionPane.showMessageDialog(this, "Incorrect Username or Password",
"FAILURE", JOptionPane.ERROR_MESSAGE);
            txtPassword.setText("");
            txtUsername.setFocusable(true);
        }
    }
    private void txtUsernameKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char inputChar = evt.getKeyChar();
        if(!Character.isAlphabetic(inputChar)){
            evt.consume();
        }
    }
}

public class FrmNewUser extends javax.swing.JFrame {

    /**
     * Creates new form FrmNewUser
     */
    public FrmNewUser() {
        initComponents();
        this.setLocationRelativeTo(null);
    }
    private void btnCreateActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String secretKey = "SDevelopers";
        String userName = txtUsername.getText();
        String password = SecurityPassword.ecnode(secretKey, txtPassword.getText());

        User user = new User(userName, password);
    }
}

```

```

        JsonManager.addToFile(user);

        FrmLogin login = new FrmLogin();
        this.setVisible(false);
        login.setVisible(true);
    }

    private void txtUsernameKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char inputChar = evt.getKeyChar();
        if(!Character.isAlphabetic(inputChar)){
            evt.consume();
        }
    }
}

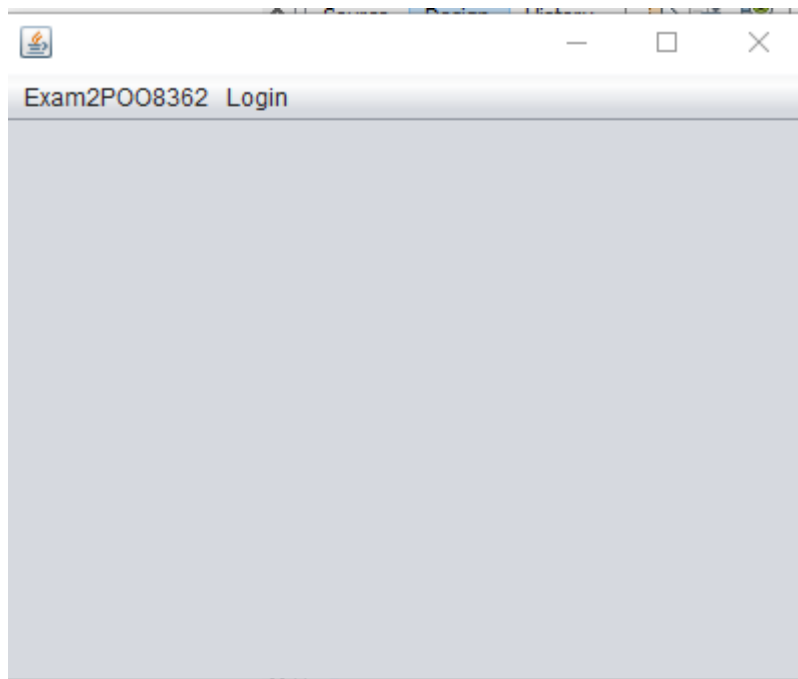
public class FrmPrincipal extends javax.swing.JFrame {

    /**
     * Creates new form FramePrincipal
     */
    public FrmPrincipal() {
        initComponents();
    }

    private void jMenuItem2MouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        FrmLogin frmLogin = new FrmLogin();
        this.setVisible(false);
        frmLogin.setVisible(true);
    }

    private void itmQuitMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        System.exit(0);
    }
}


```

A screenshot of the same Java Swing window, now populated with a login form. The form has a title "LOGIN", two input fields labeled "Username" and "Password", and a "LOGIN" button. The "Username" field contains the text "Karen".

LOGIN

Username

Password

—□×

NEW USER

Username

Password

Create