Flip Coin Game

Submitted By

Group A

Md.Azizur Rahman- 14153203054

Md.Mirazur Rahman-14153203053

Md.Nazmul Haque- 14153203061

Mst.moarium Akter- 14153203059

Submitted To

Susmita kor

Department of CSE

Contents

Bangladesh University of Business and Technology (BUBT)
Department of Computer science engineering

Project Name: Flip Coin Game.

Objective: Coin flipping, coin tossing, or heads or tails is the practice of throwing a coin in the air or using program checking which side is showing when it lands to choose between two alternatives, sometimes to resolve a dispute between two parties. It is a form of sortation which inherently has only two possible and equally likely outcomes Head or Tails.

Overview:

The programming assignments during the semester will comprise parts of a single application a Game Project. The Game Project is to be done in teams of from one to four people per team .Over the course of the semester each team will implement a complete game with capabilities as outlined below. At the end of the semester we will have a "Game Party" where each team demonstrates its game and has a chance to see and play those of other teams.

Language:

Java
Java Fx

Flip Coin Game Rules

1. A user must have authorization to play game.
2. A player must have an Account.
3. A user needs Valid Email address.
4. An email address user can use only once.
5. User can change password by sending request to admin.
6. Minimum bet is 25$.
7. Maximum bet is 5000$.
8. For withdrawal user must have minimum balance 10k$.
9. Winner will get amount 200% included 5% Vat.
10. Loser wills loss Total investment.

Flip Coin Software Process Models

Incremental Development

® In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a “multi-waterfall” cycle.

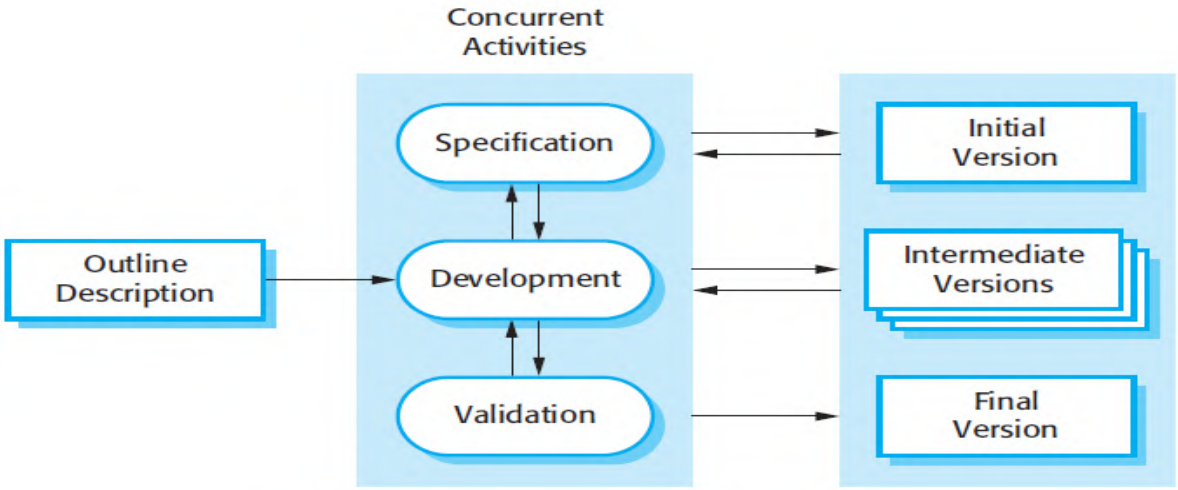
® Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases.

® A working version of software is produced during the first module, so you have working software early on during the software life cycle.

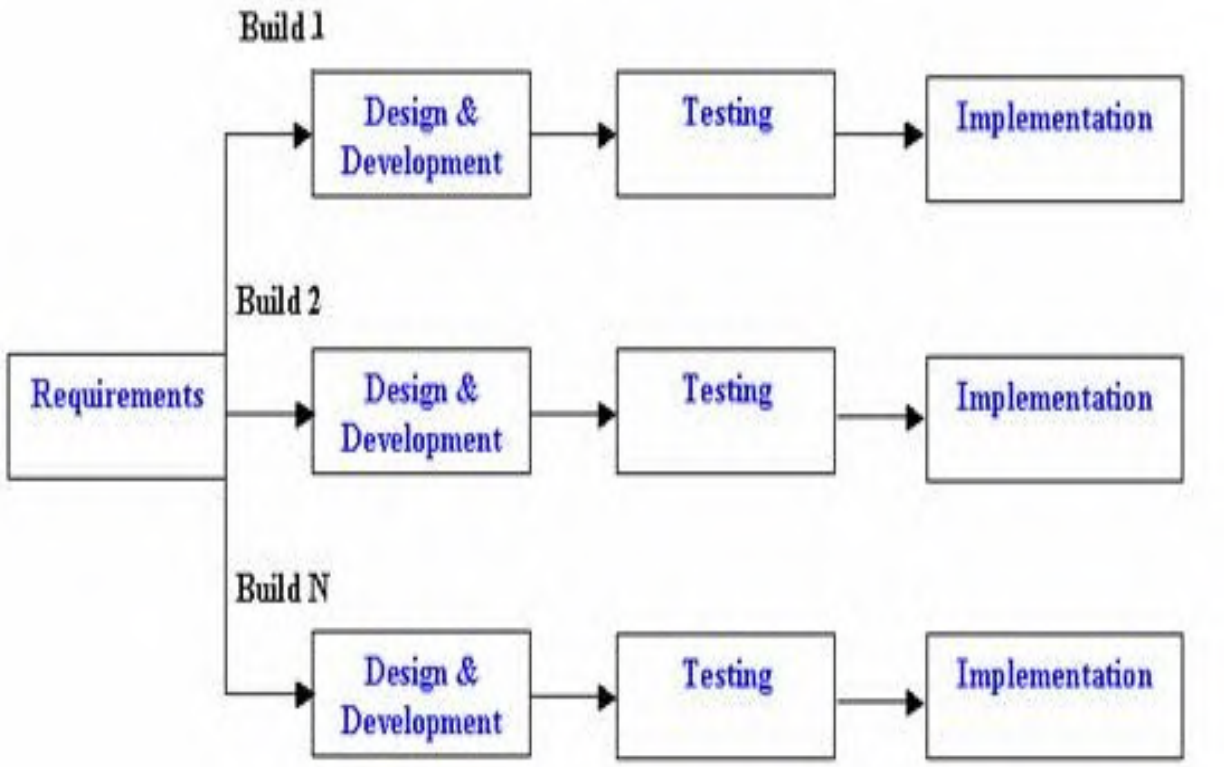
® Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved.

✧ For example:

Incremental Model:

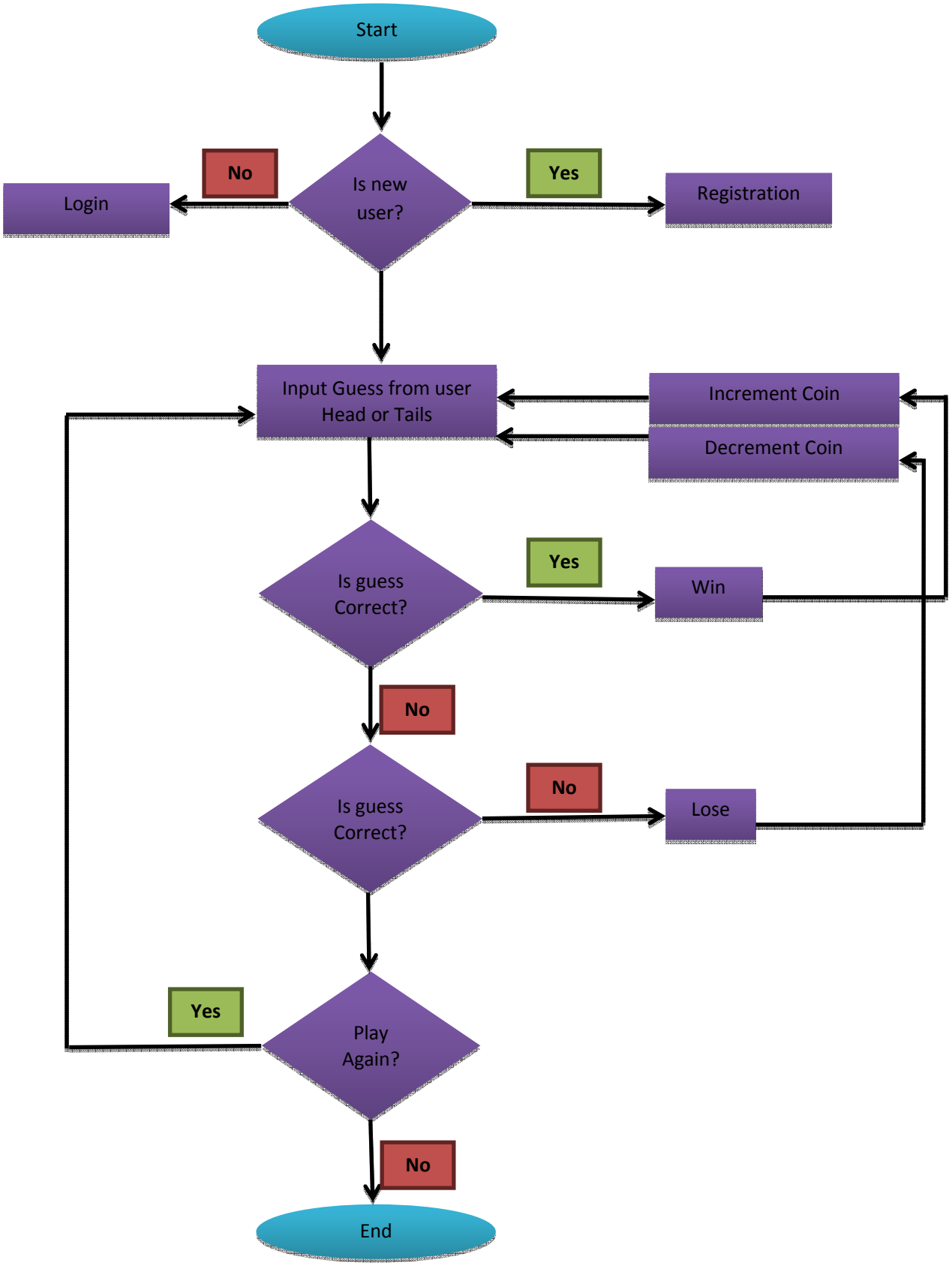


How does it work?



Incremental Life Cycle Model

Flip Coin Game Algorithm

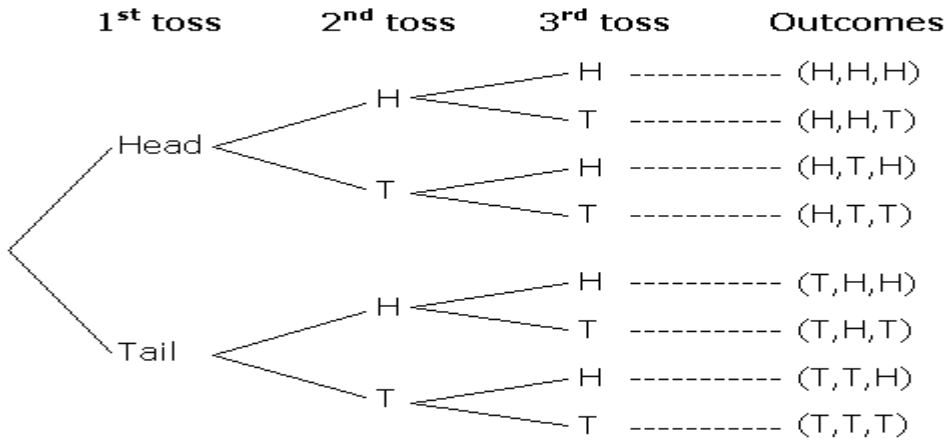


Flip Coin Games Tree Diagram

Clare tossed a coin three times.

a) Draw a tree diagram to show all the possible outcomes.
b) Find the probability of getting:
(i) Three tails.
(ii) Exactly two heads.
(iii) At least two heads.

a) A tree diagram of all possible outcomes.



b) The probability of getting:

(i) **Three tails.**

Let S be the sample space and A be the event of getting 3 tails.

$n(S) = 8$; $n(A) = 1$

$P(A) = \frac{1}{8}$

ii) **Exactly two heads.**

Let B be the event of getting exactly 2 heads.

$n(B) = 3$

$P(B) = \frac{3}{8}$

iii) **At least two heads.**

Let C be the event of getting at least two heads.

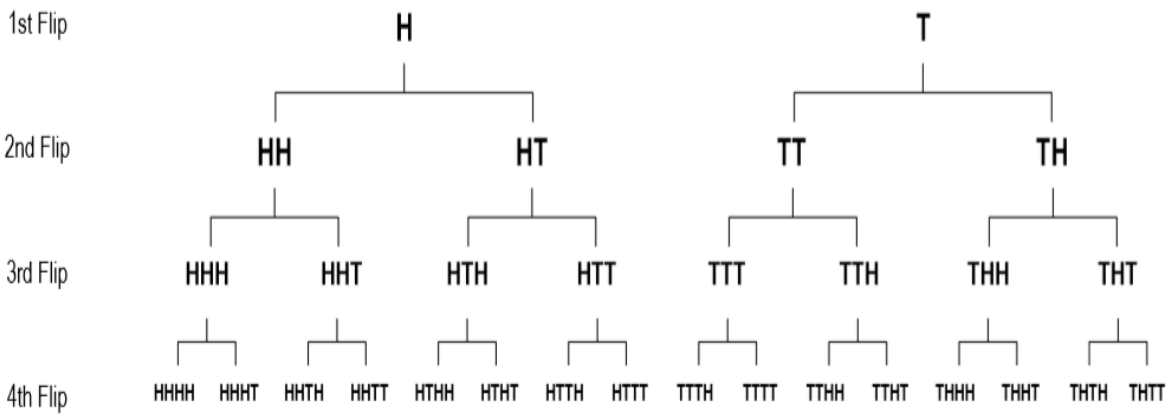
$n(C) = 4$

$P(C) = \frac{4}{8} = \frac{1}{2}$

Calculating probability

A simple diagram like the one below can be used to calculate the probability of any outcome. The sum of the probabilities of all possible outcomes for a random event must always add to 1. For a coin flip, there are two possible outcomes, either heads or tails. So if the coin is unbiased, and the coin flipping technique is also unbiased, then the probabilities are exactly half and half (½ or 0.5 or 50%).

So, using the diagram, there are four possible outcomes on the second flip. The probability of getting HH on the second flip is ¼ (or 0.25). The probability of HT, TH, or TT are all also ¼. The probability of HHH on the third flip is 1/8 (or 0.125) and HHHH on the fourth is 1/16 (or 0.0625).

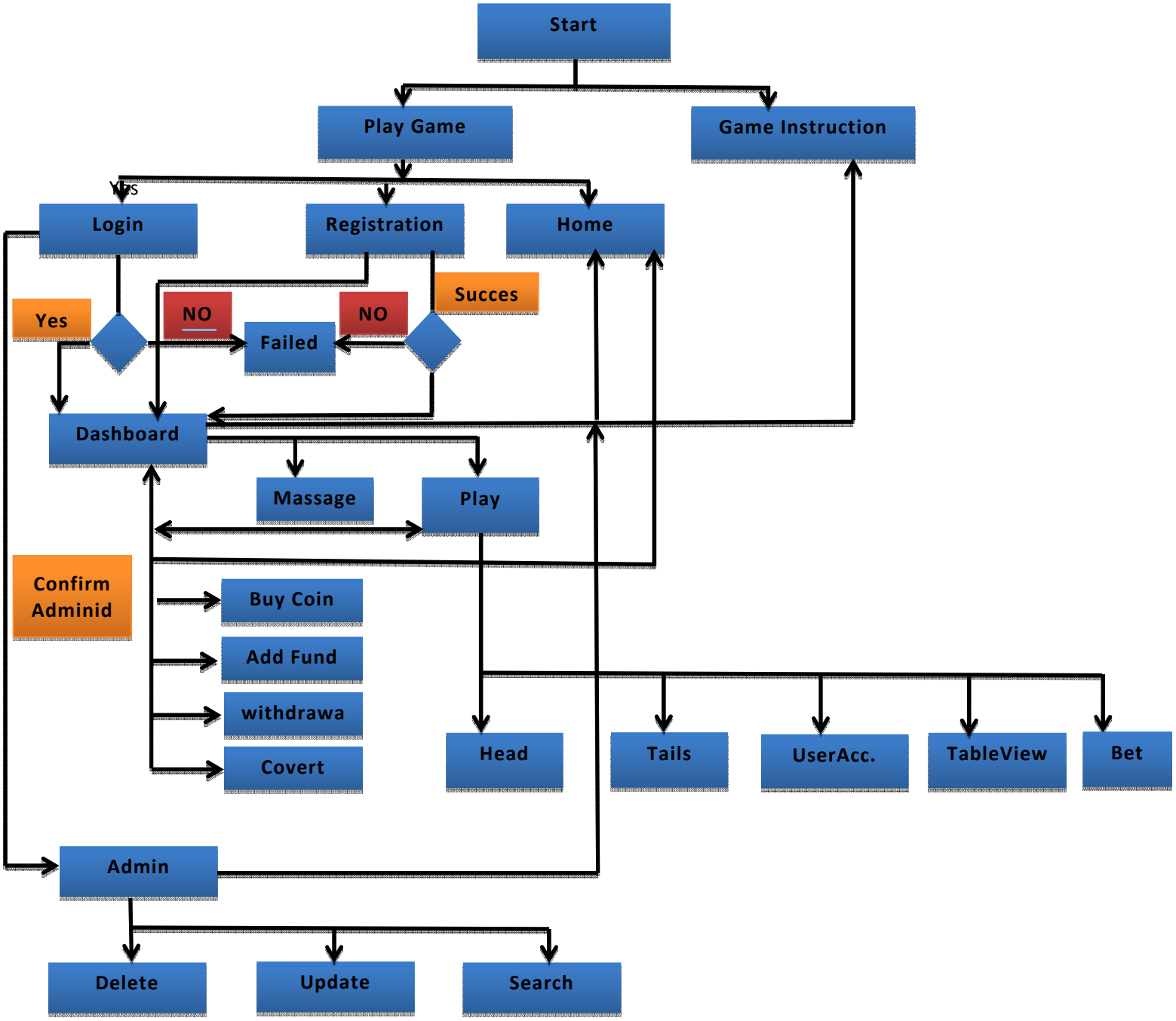


A faster way to calculate the probability of an outcome after any number of flips is to multiply the probabilities of all the outcomes together. So the probability of HH is ½ × ½ = ¼. The probability of HHH is ½ × ½ × ½ = 1/8 and of HHHH is ½ × ½ × ½ × ½ = 1/16.

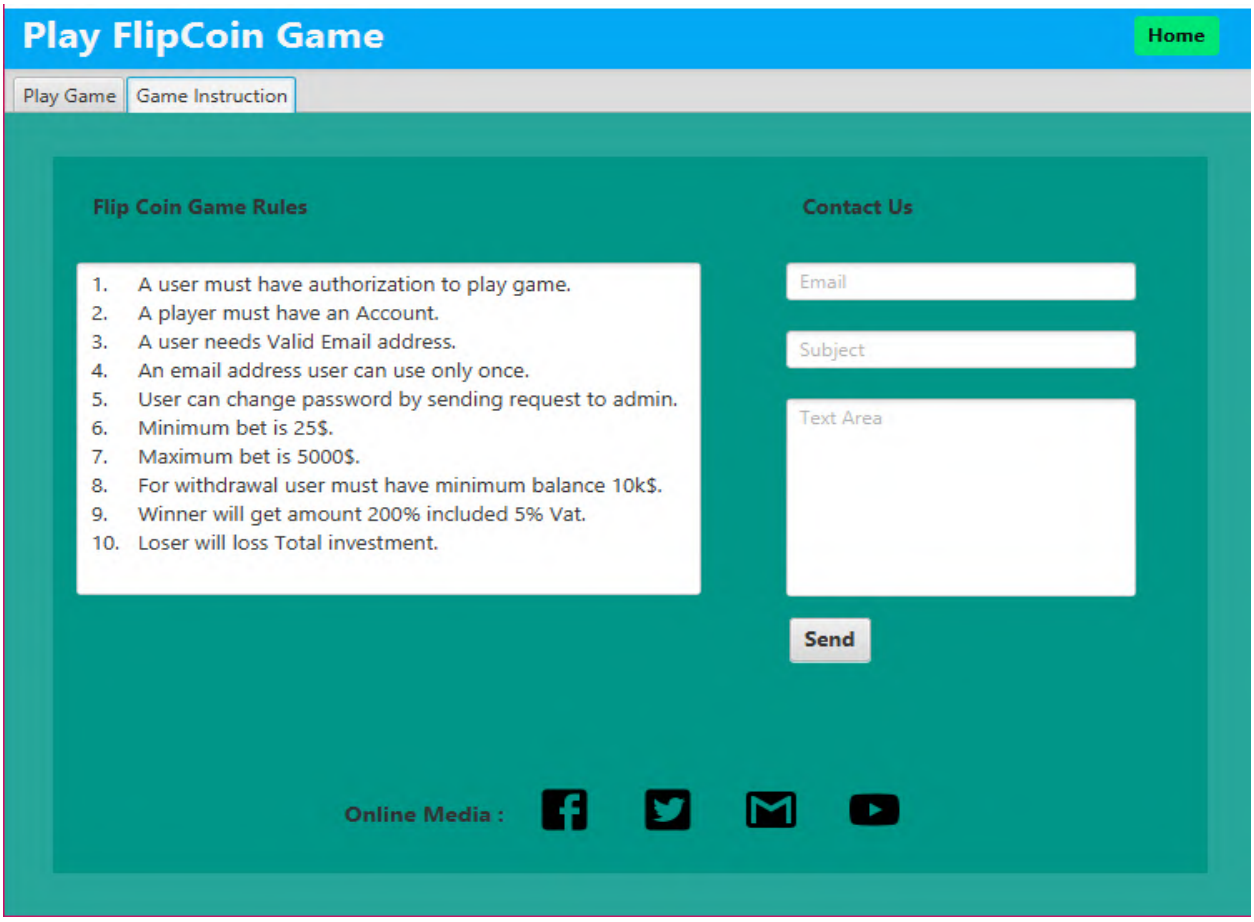
Calculating the probabilities and expected frequencies of clusters is beyond the scope of primary school mathematics but students should be able to grasp the concept of their expected frequency and notice that their combined results match these predictions almost perfectly.

The expected frequencies for each cluster size are provided in Table 5 in the Student Worksheet. Each frequency in the table represents the average number of times you would expect to see these clusters in a large number of sets of results from one hundred coin flips.

Flip Coin Game System View



Game Instruction

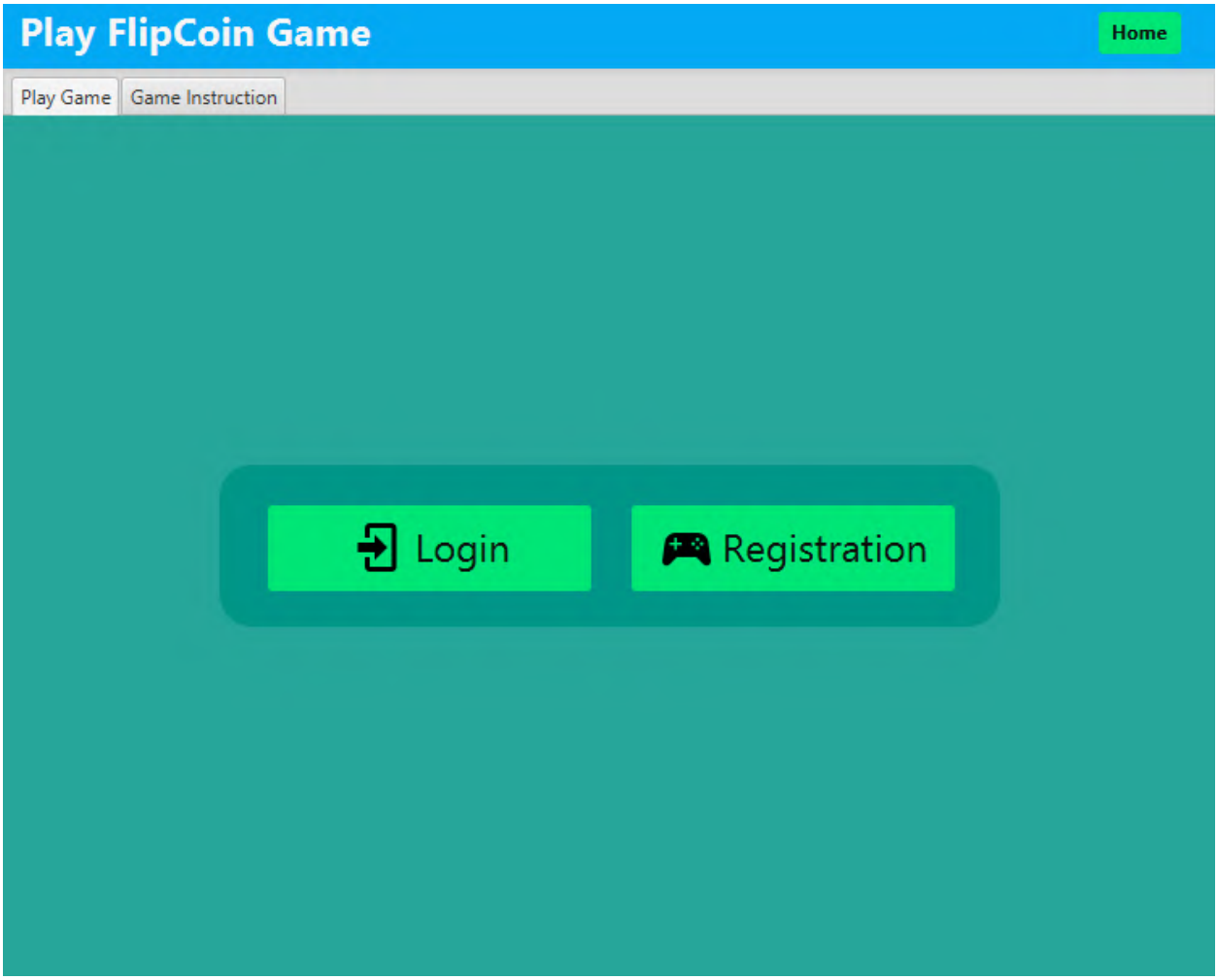


Flip Coin Game Rules

1. A user must have authorization to play game.
2. A player must have an Account.
3. A user needs Valid Email address.
4. An email address user can use only once.
5. User can change password by sending request to admin.
6. Minimum bet is 25$.
7. Maximum bet is 5000$.
8. For withdrawal user must have minimum balance 10k$.
9. Winner will get amount 200% included 5% Vat.
10. Loser will loss Total investment.

Above those rules must followed by a user of Flipcoin Games.

Authorizetion Form

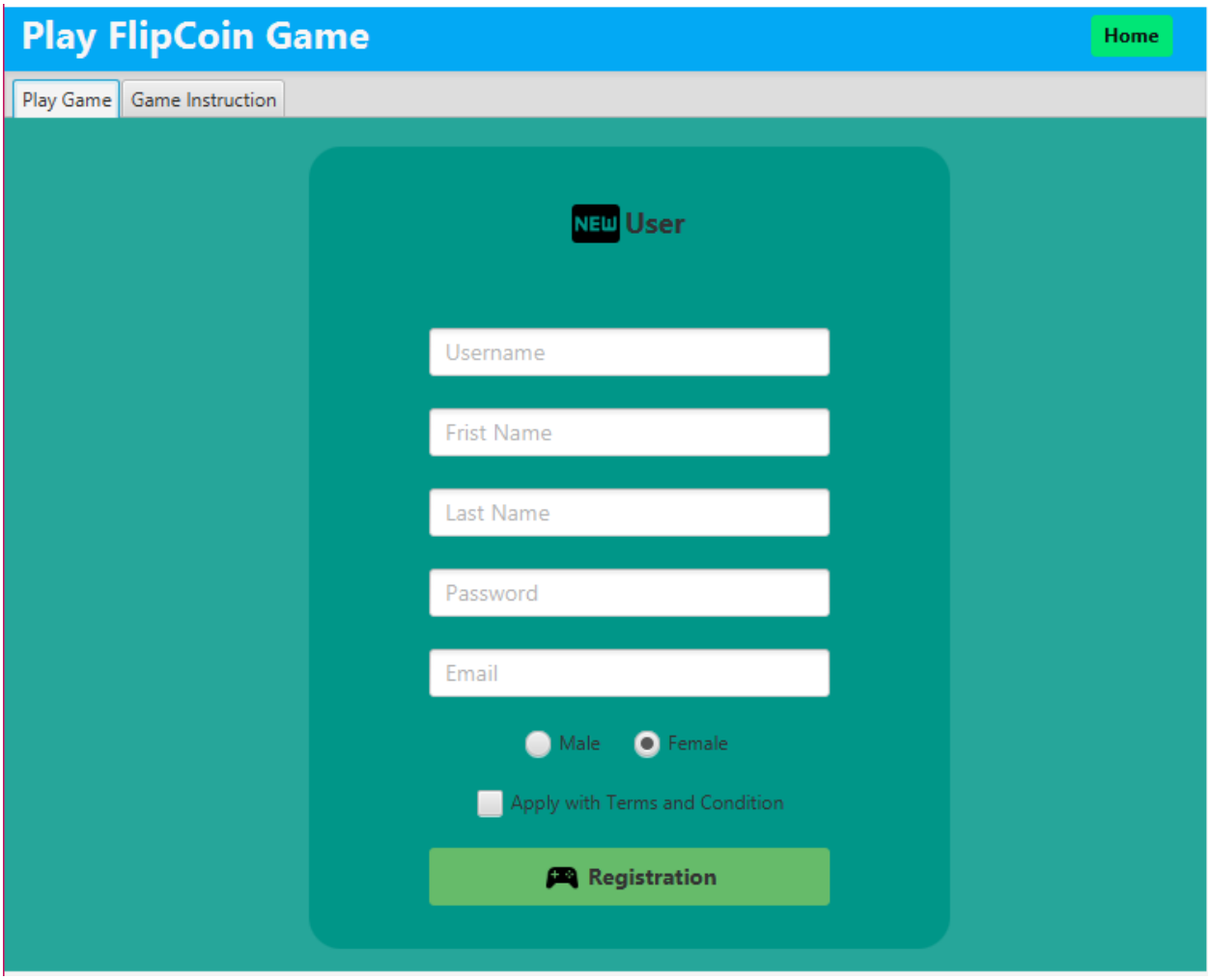


After a Welcome Screen a user will get a screen where have two options

1. Login
2. Registration

If a user be a new person in this game, user must registration first, after the user can login into the Game without registration a user can't access main form where user can play the Flip Coin Game.

Registration Form



In the registration form a user must have fill 5 information such as

1.Username

2.First name

3.Last name

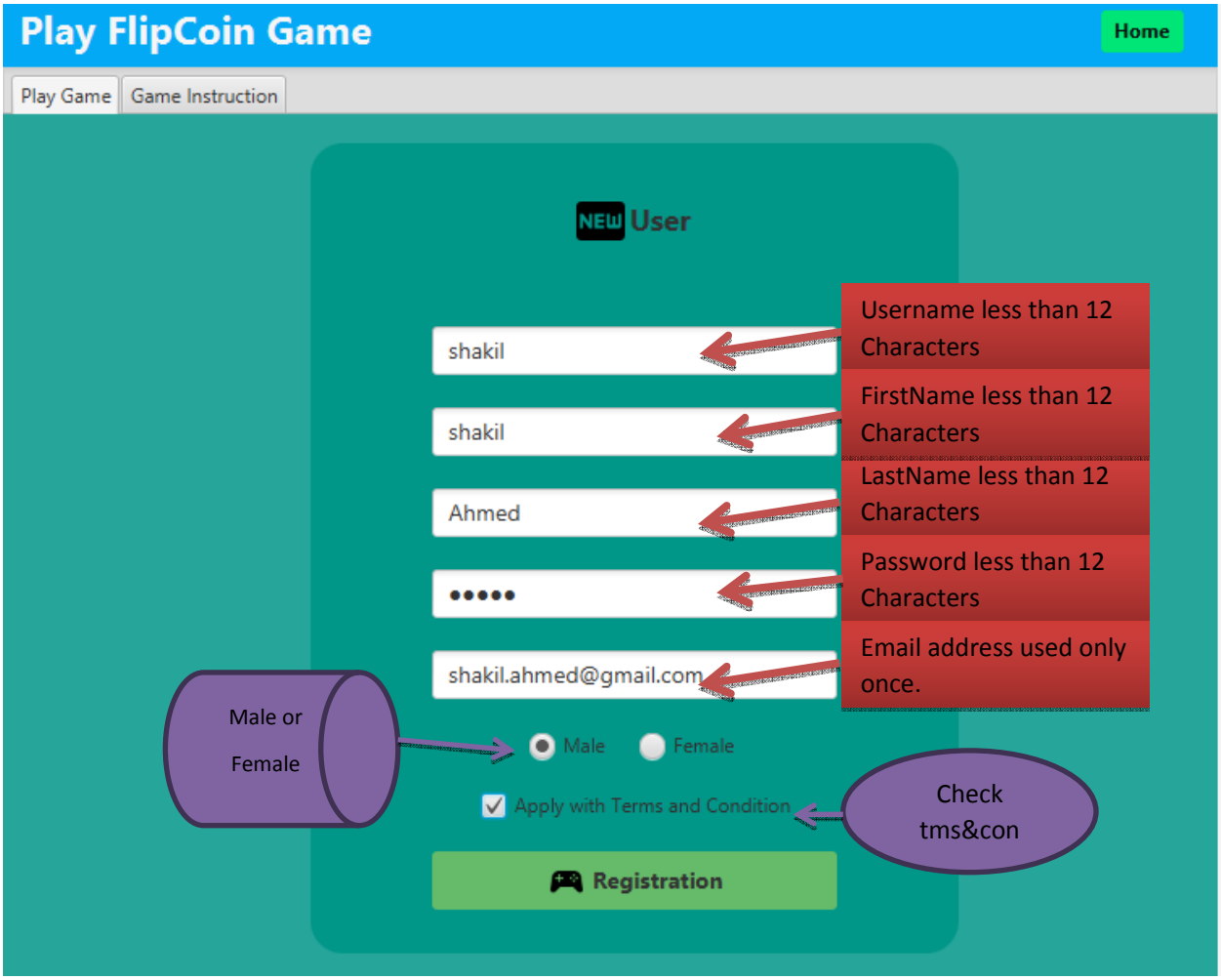
4. Password

5.Email

And also Two clickable checked point one is Radio Button another is CheckBox

Then press Registration.

Registration procedures



1.Username: User can gives as a user name text and number.

2.First name: user sur name

3.Last name : last name of user

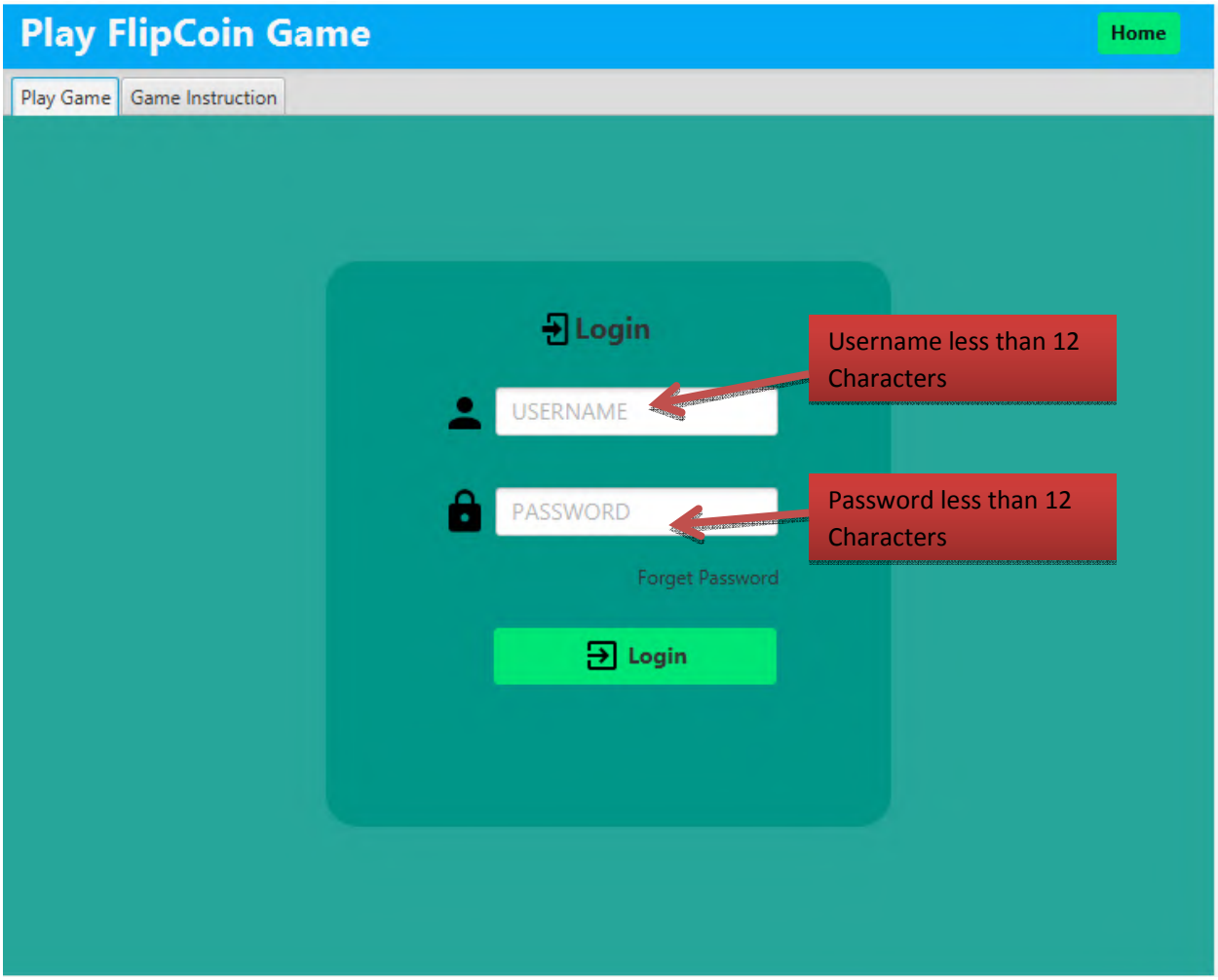
4. Password: Number or text combination

5.Email: User should use valid Email address. One more thing is that a user can't able to open two Flip Coin Game id by One email Address.

** Click Male or Female Button according to your Gender.

** Check above all terms and condition that is given by a user.

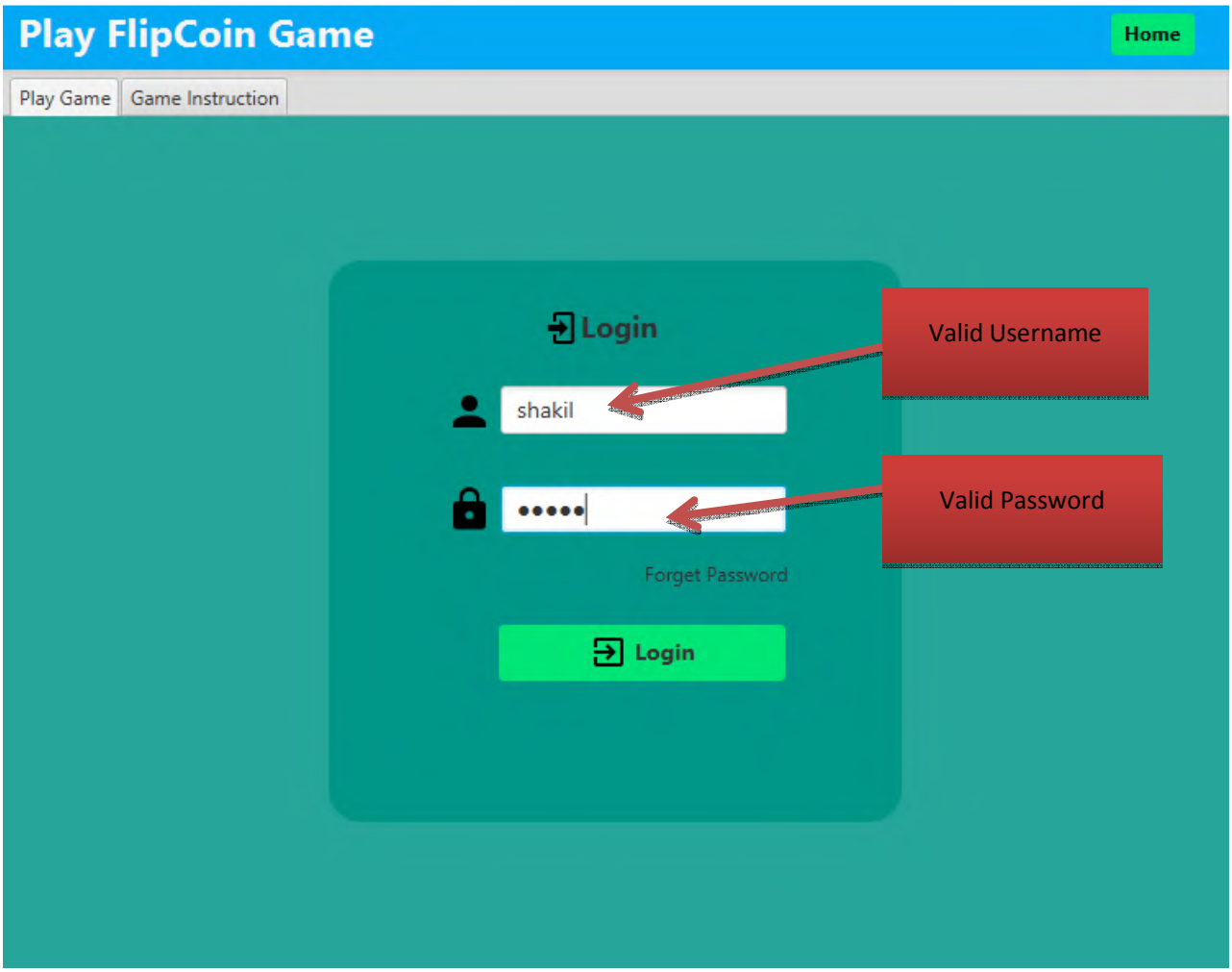
Login Form



In login form has username and password

1. Username: user name will be less than 12 characters.
2. Password: password will be less than 12 characters.

Login procedures



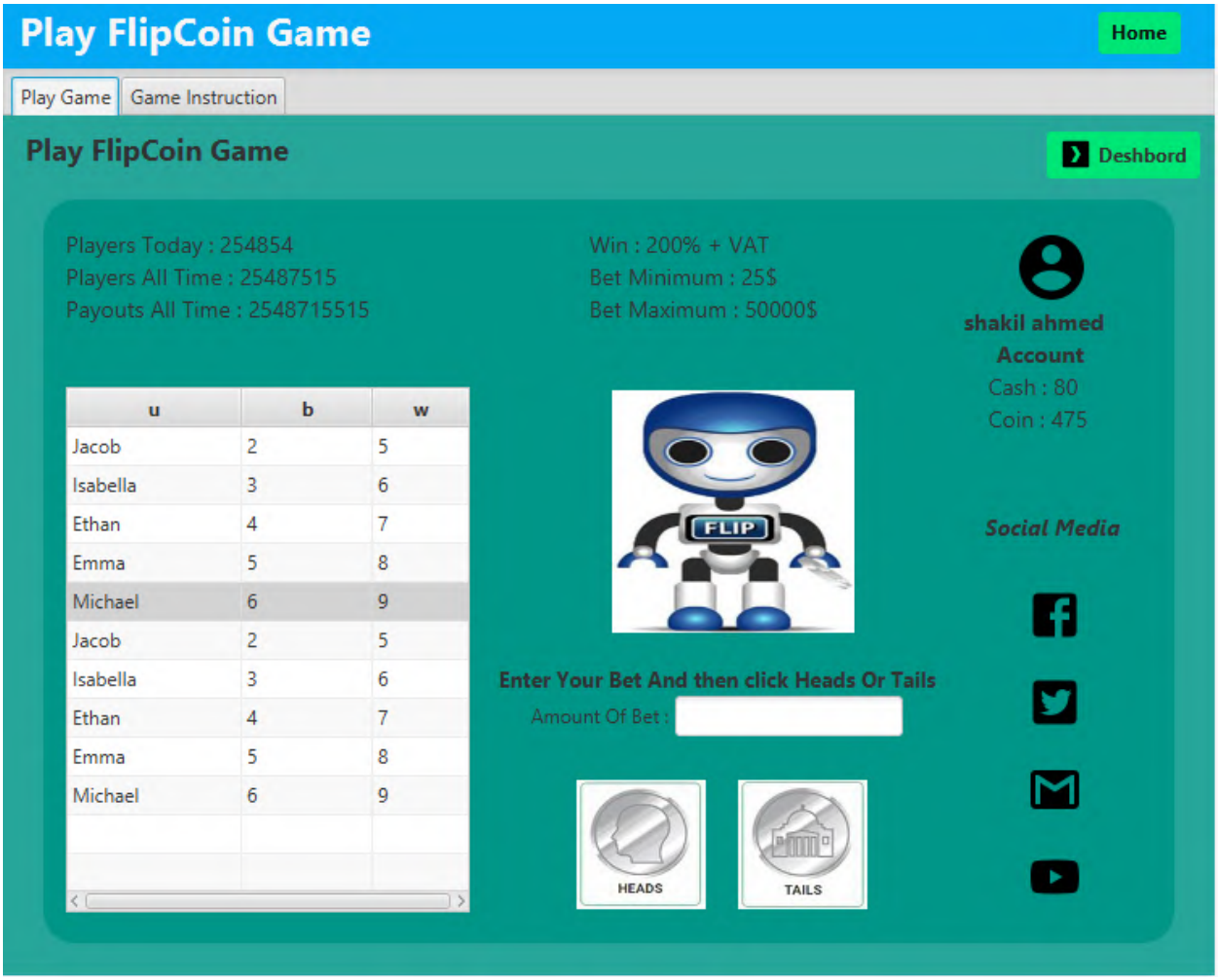
In Login Form
Only a registered user can login, a registered user will give valid username and password that is gave by user when user got registration.

After that a user can successfully Login.

Deshbaord

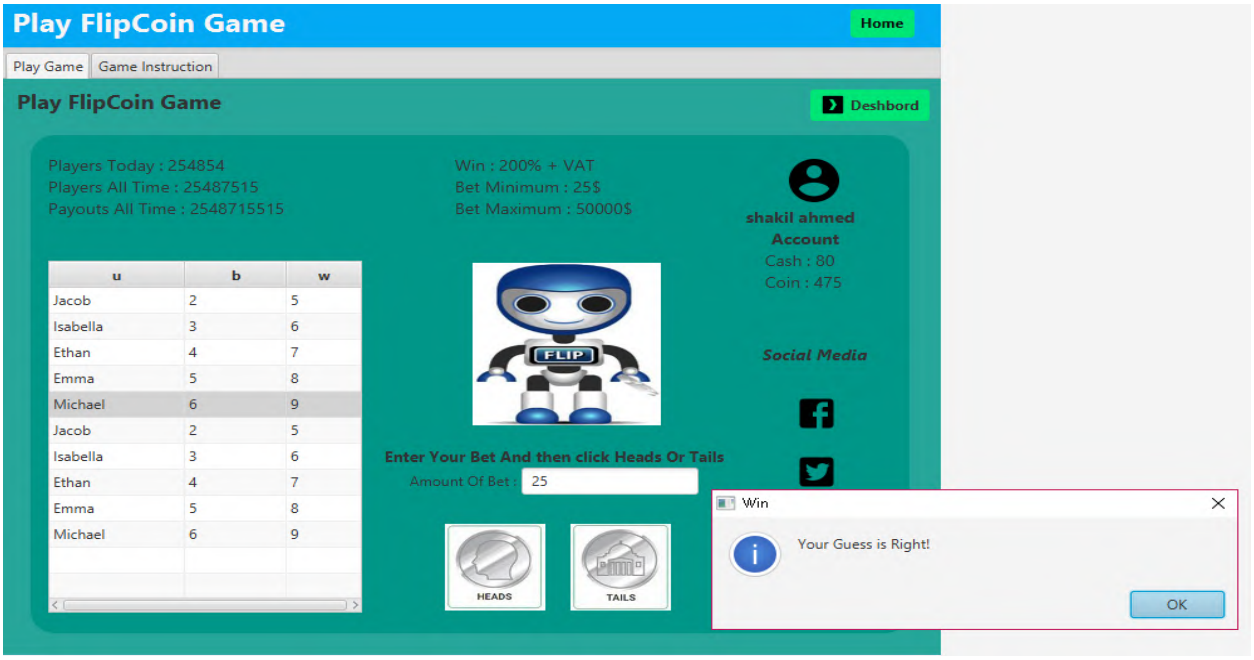
Dashboard, it will show five new user and their coins. In X axis it shows last Five individual user. In Y axis it shows amount of Coin last five individual users.

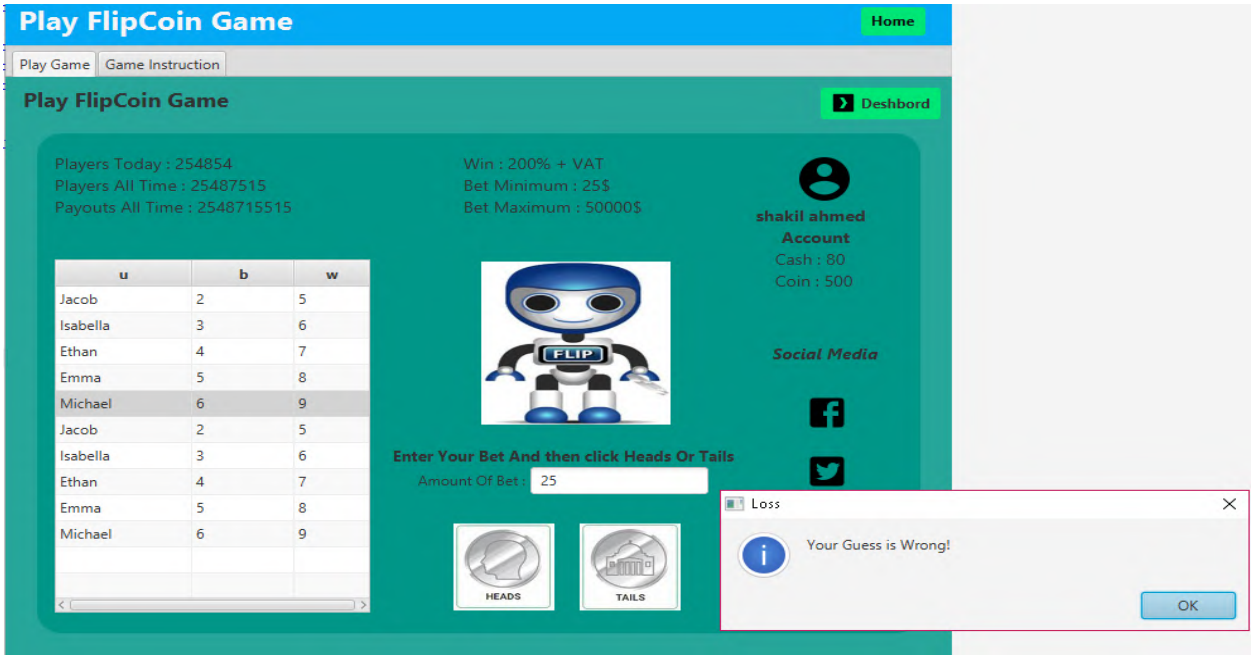
Play Games



This surface is for the user to playing game. User can randomly choose heads or tails option accordingly user opinion .here guess pattern is more important things for playing game. User must choose right one for earn money.

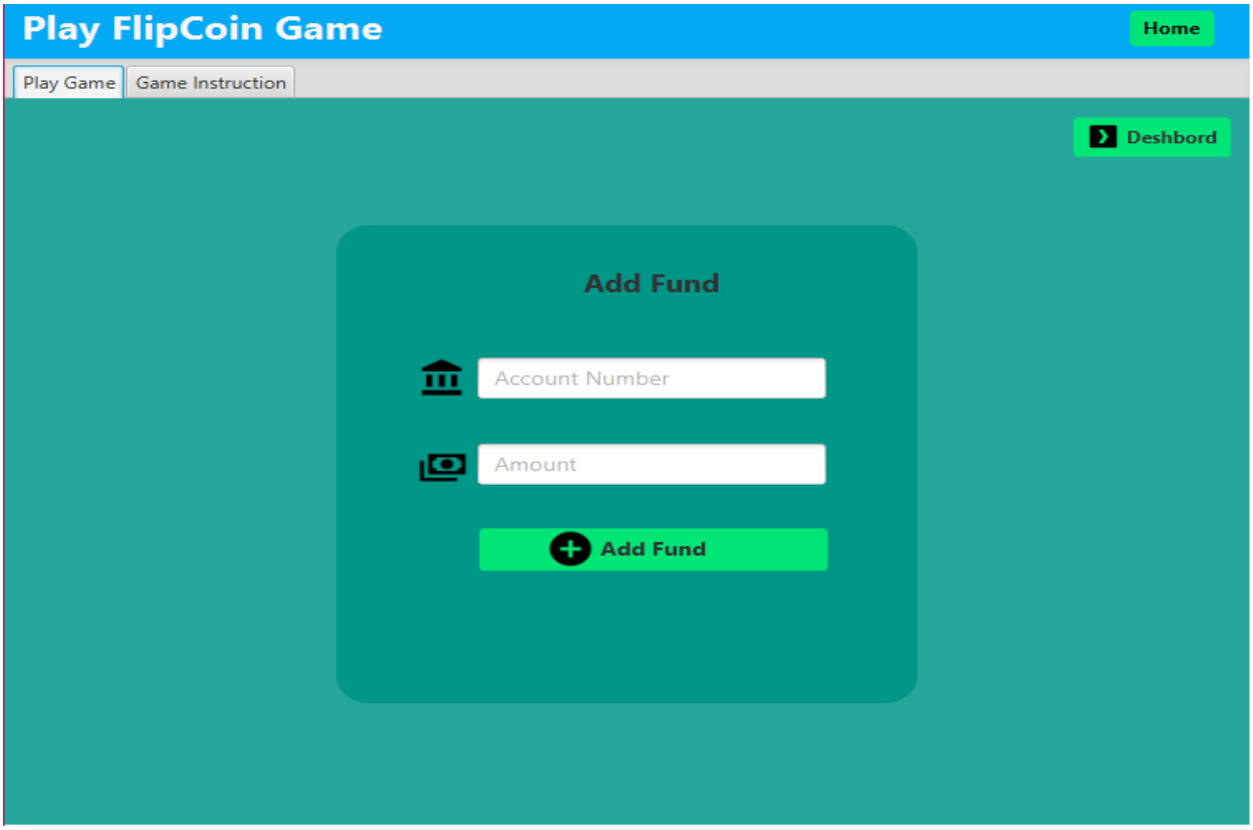
Random selection /Toss





A user can choose only one option head or tails.
If user guess is correct then user will get 200% Bonus but user will gives 5% VAT to service provider of Game.
On the other hand if user lose the toss .user will lose the toss money that user bet.

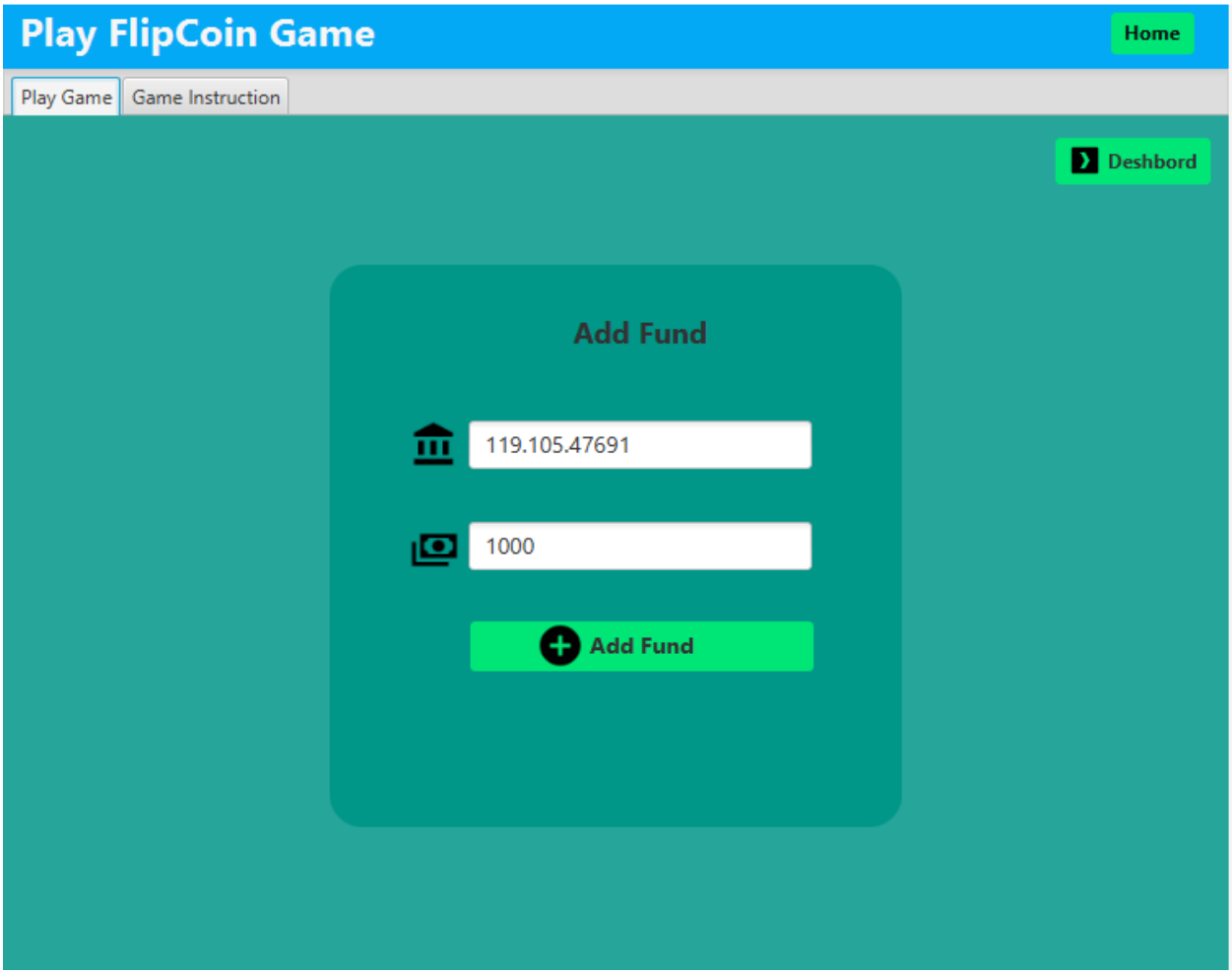
Adding Game Fund



Sometimes a user need amount to paly game. But user doesn't have money on his/her account to play game.
At that time user can transfer money to Flip coin games account from his/her valid Bank Account number.
This is the easy and quickest way to transfer money and play game.

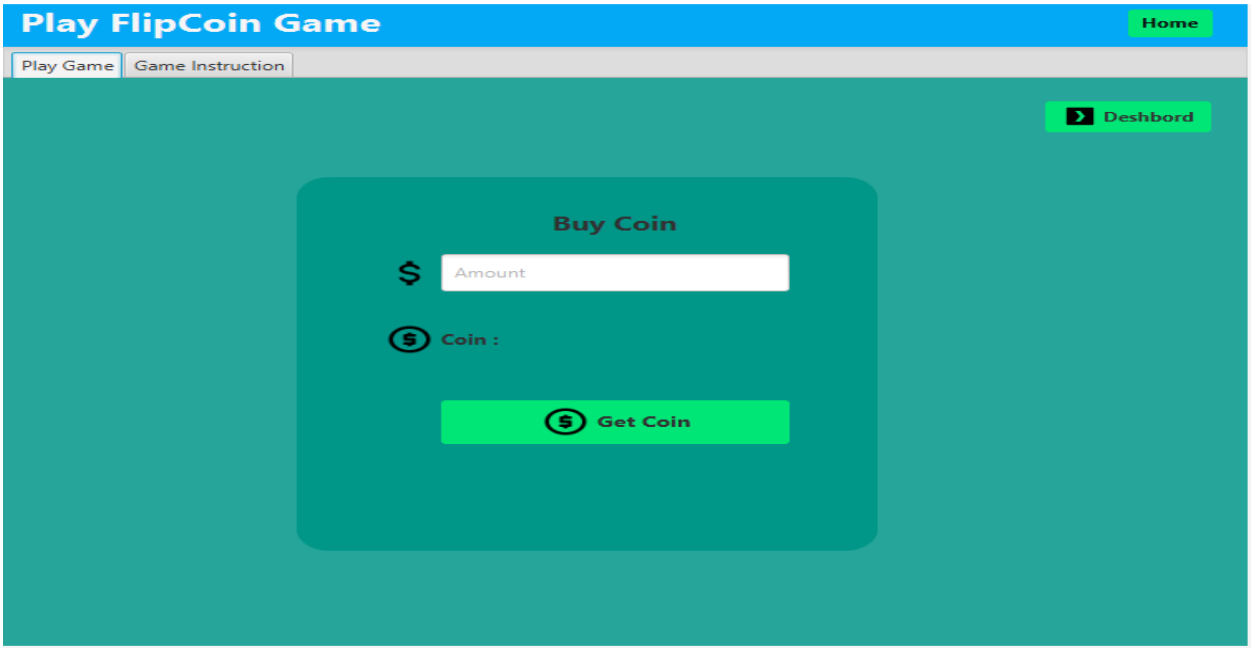
User also can use his/her bKash account number. This is upcoming process. Flip coin game developer team working on it .user will access bkash account very soon…….

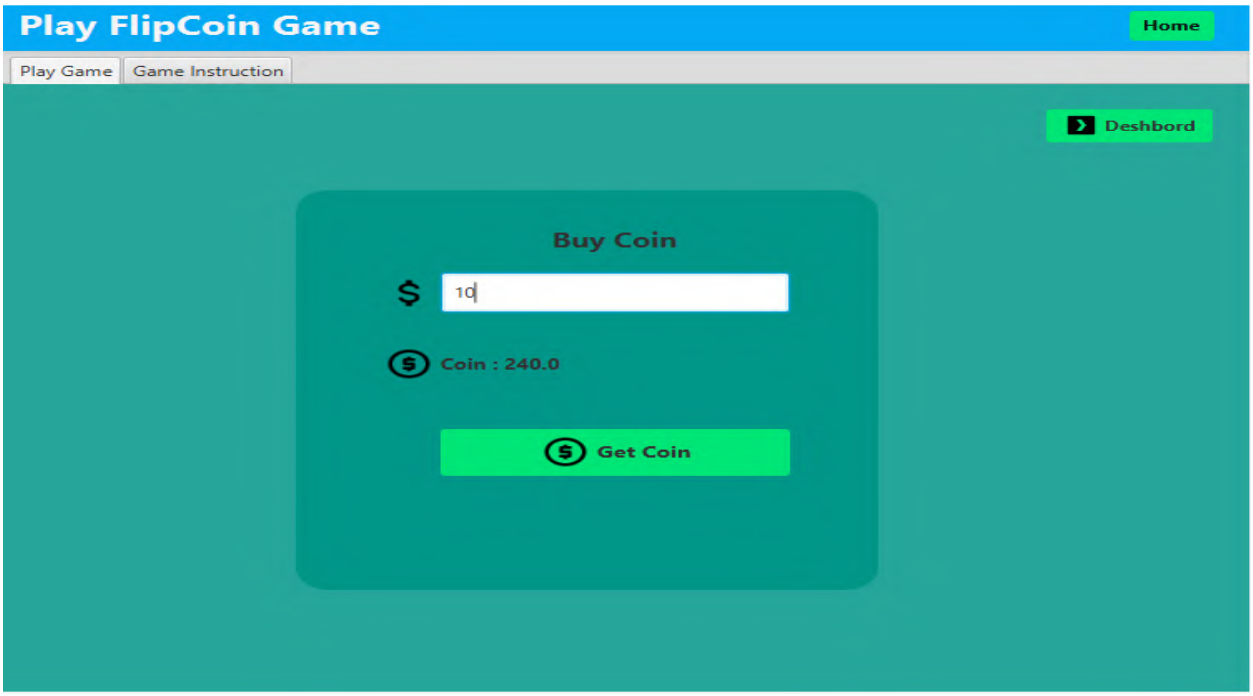
Adding Game Fund Procedure



This is the procedure how to Transfer money from the Bank account. There is two options one gives valid 12 digits account number and another one is gives how much amount wants to transfer from account.

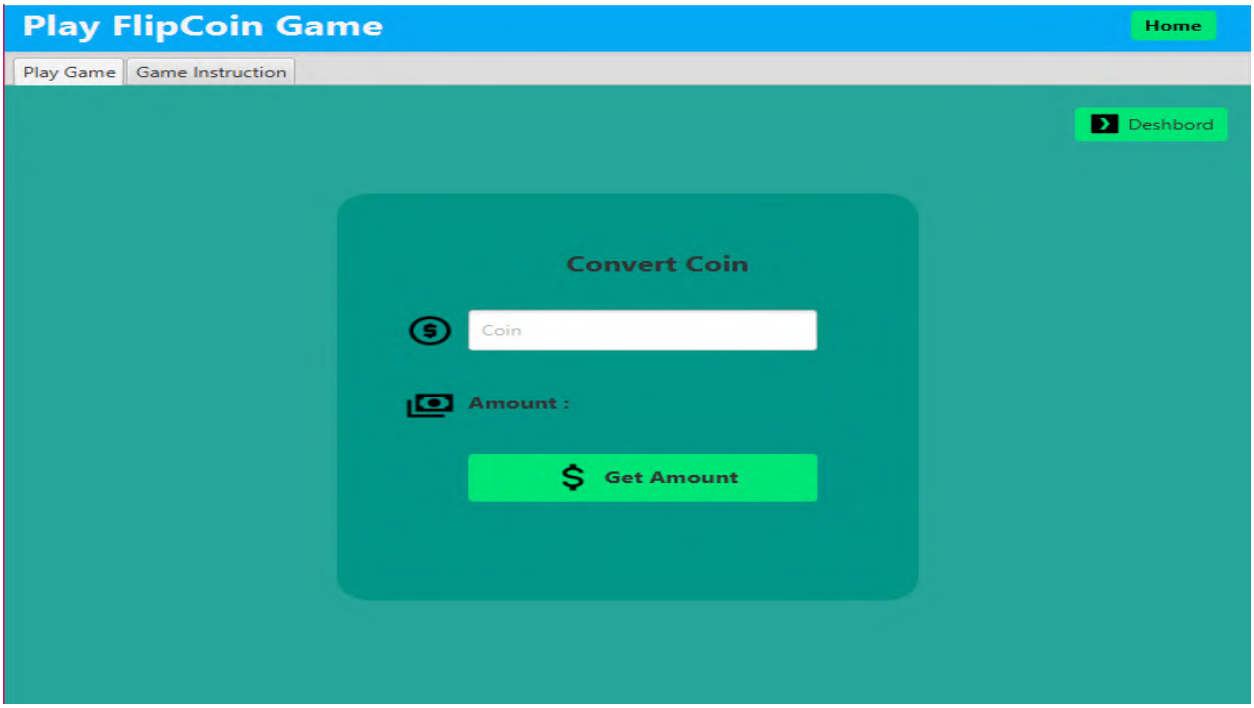
Buy Coin

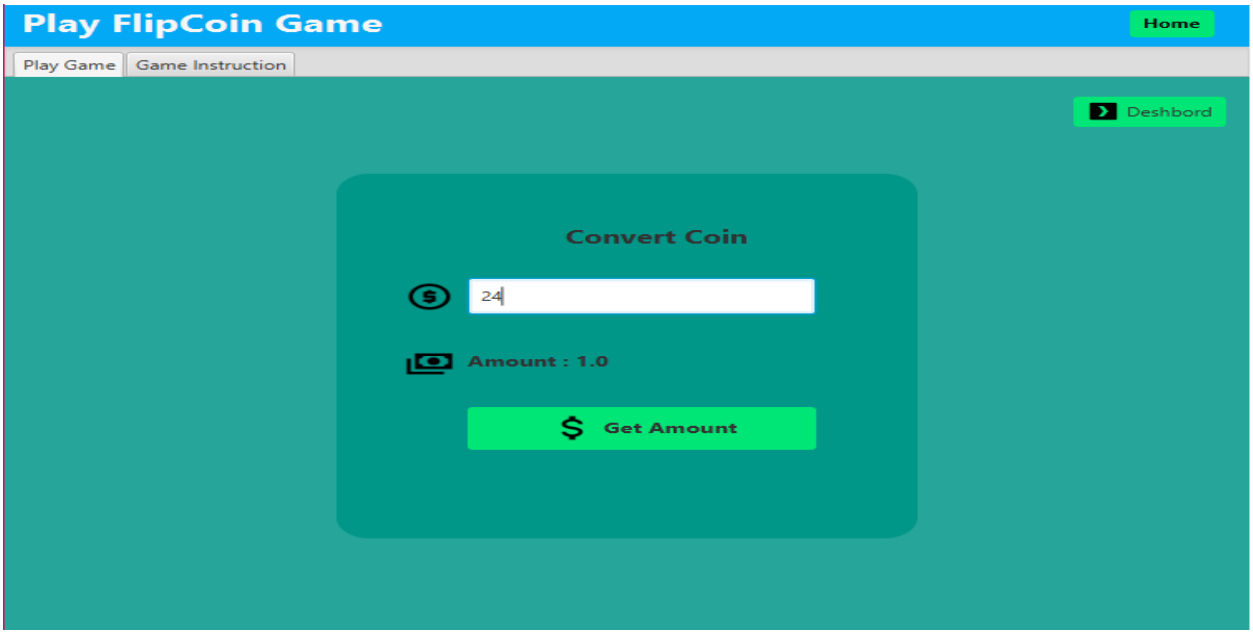




Buy coin is the process by following this process a user can buy coin to play game.
A user will get 24 coins by one dollar. Ten dollar user will get 240 coins.
At first a user should buy two dollar coin because of user must have minimum 25 coins to play game.

Convert Coin



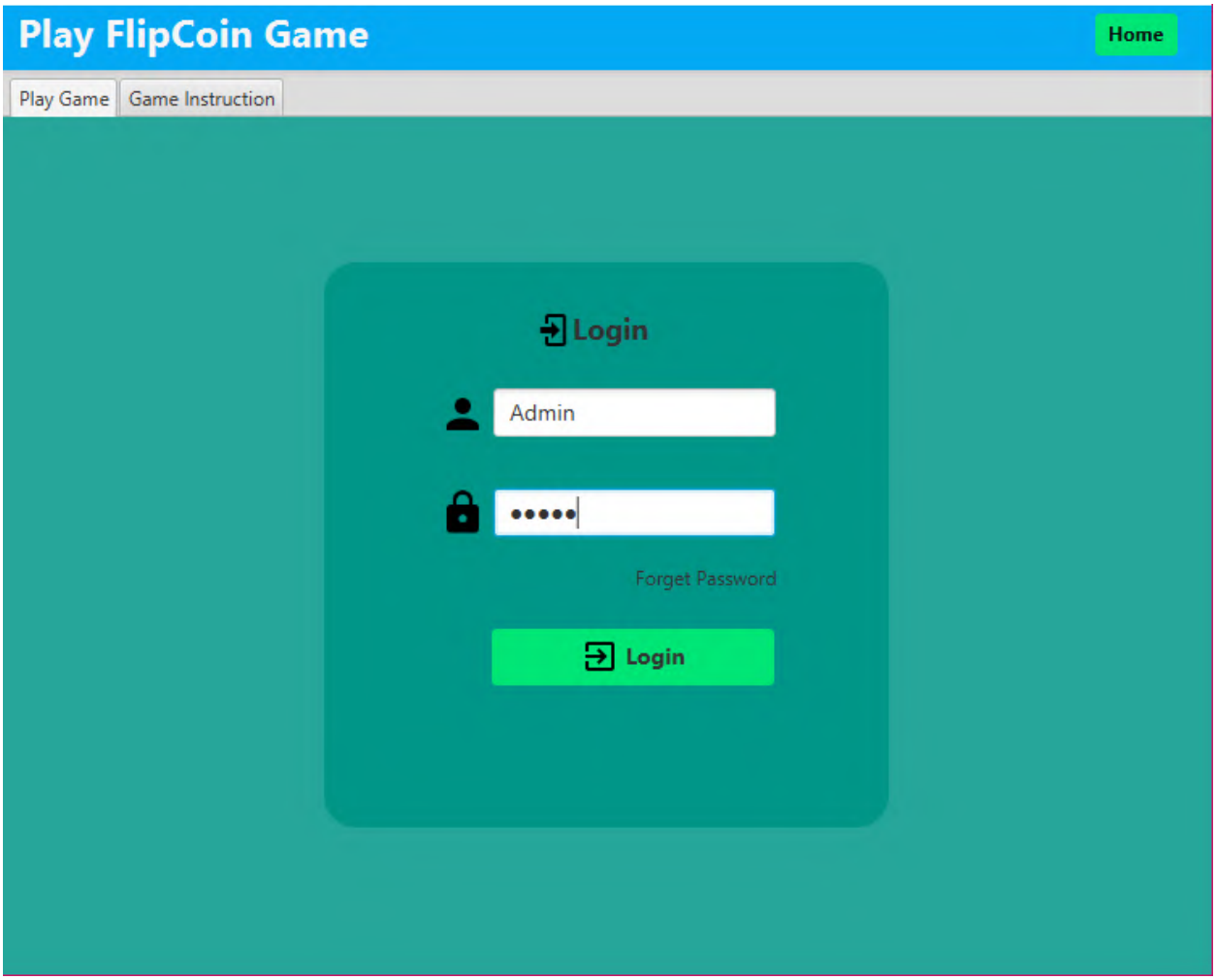


After playing games user will wish to get money from flip coin game account.
At first user must convert amount to dollar. Following this procedure a user can convert whole coins to dollar. It's depending on user how much he/she wants to convert.

Cash Out

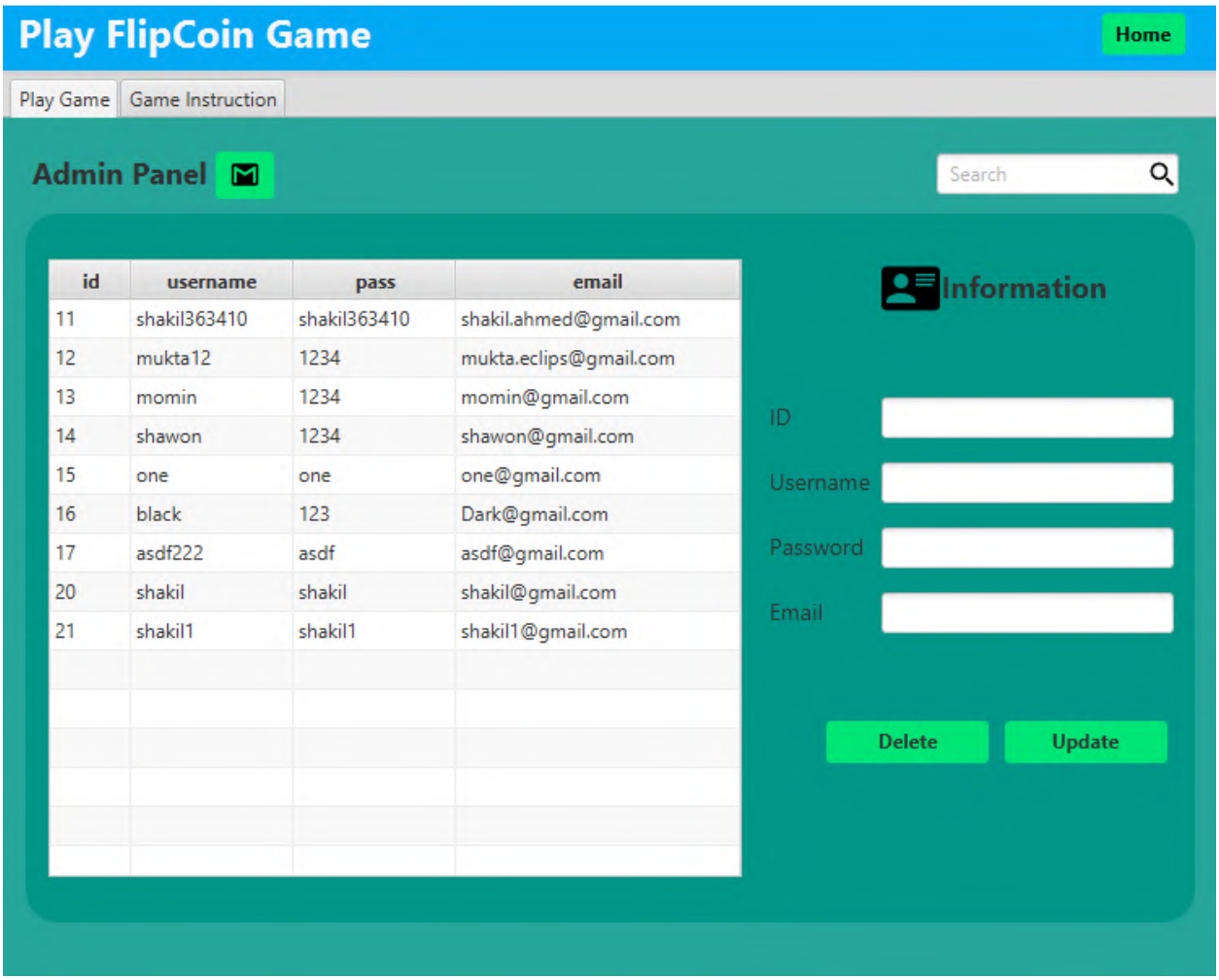
After playing games user will wish to get money from flip coin game account. After converting coin into dollar user can send money to his /her valid Bank account .money will send after confirmation of flip coin game authority.

Admin Login



Admin user has whole power of the games. Admin can delete any users who are not abiding by rules and regulations of flip Coin game. Admin can update user information according to user valid request, find out by searching by the username of a user.

Table View of Admin Panel



After admin login view of a admin panel. There is a table view where has all of the information of every single user. Delete, update, search options available.

Delete user

Before deleting id 17 view of admin table. In the right-bottom side of the table has delete button.

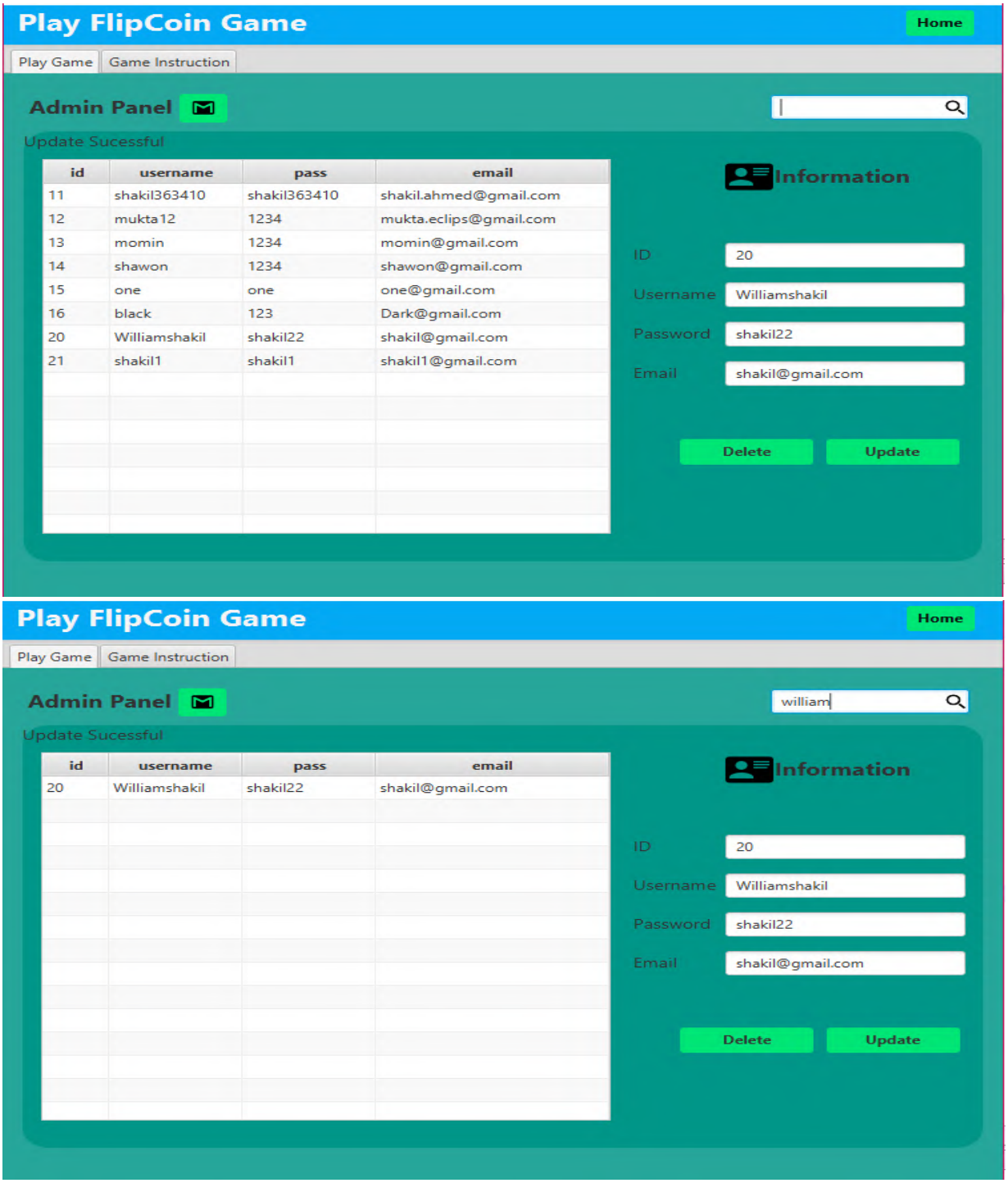
After deleting id 17 is gone. Others id's are same as like as before was.

Update User

For updating id, admin selected id was 20.

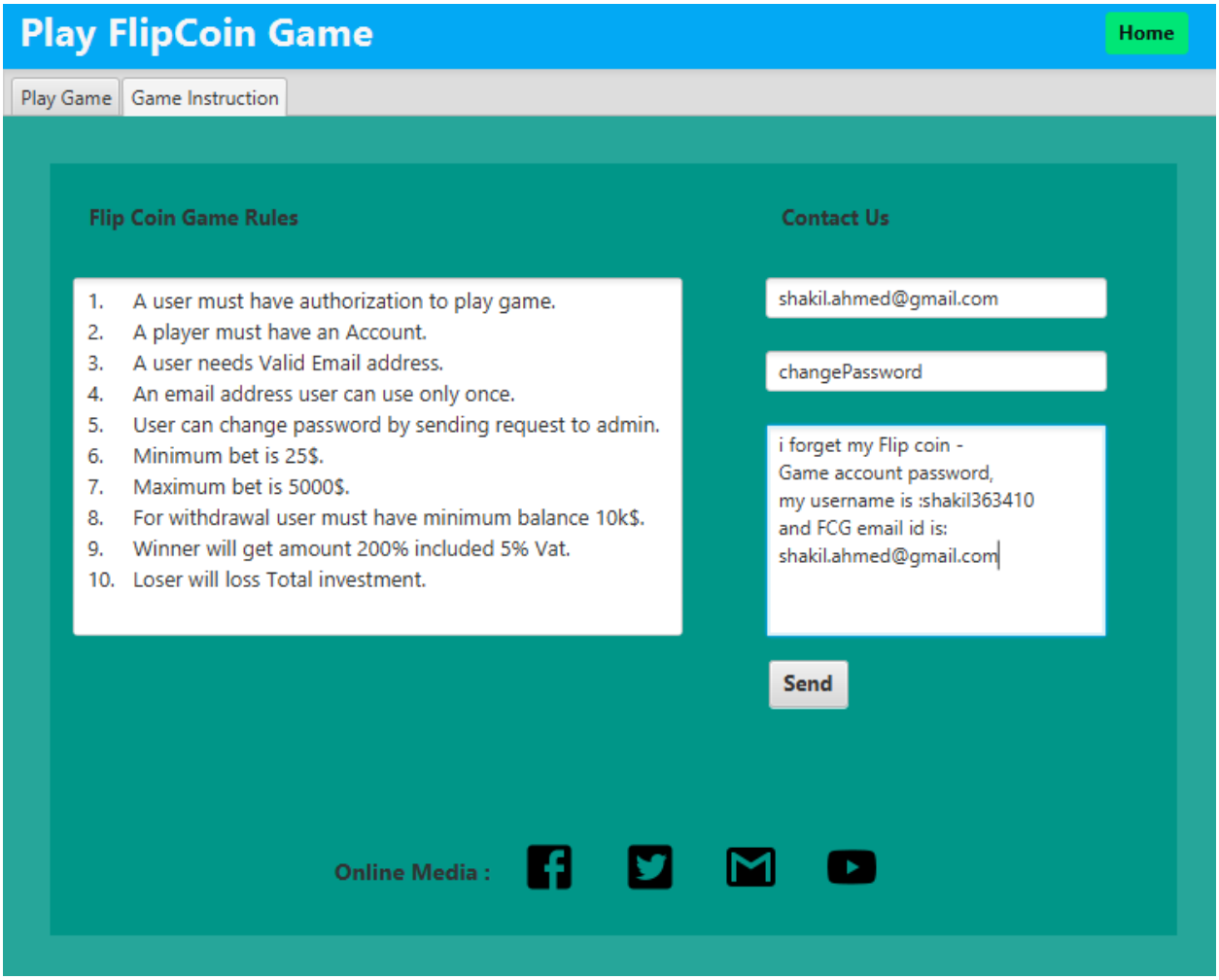
After updating here we find that admin change username from **shakil to Williamshakil**, password **shakil to shakil22** and others thinks kept same as like before.

Search User



Admin searching for username williamshakil. in the first image where this id position is 20.in the second image view admin only write down William in the search box of image view two. In the table view it's shown that id 20, username, pass and email.

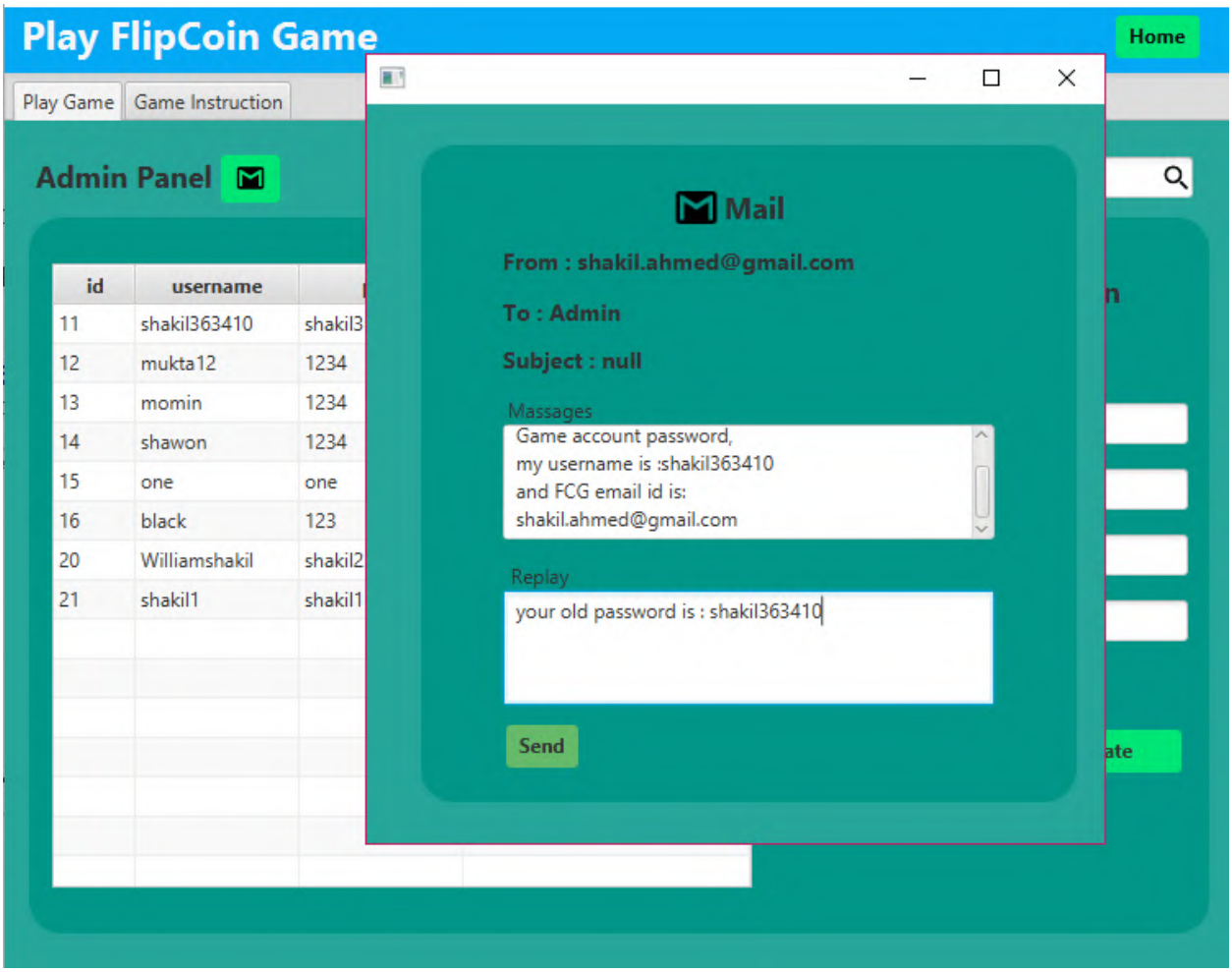
User request



For Forget password user sending text request

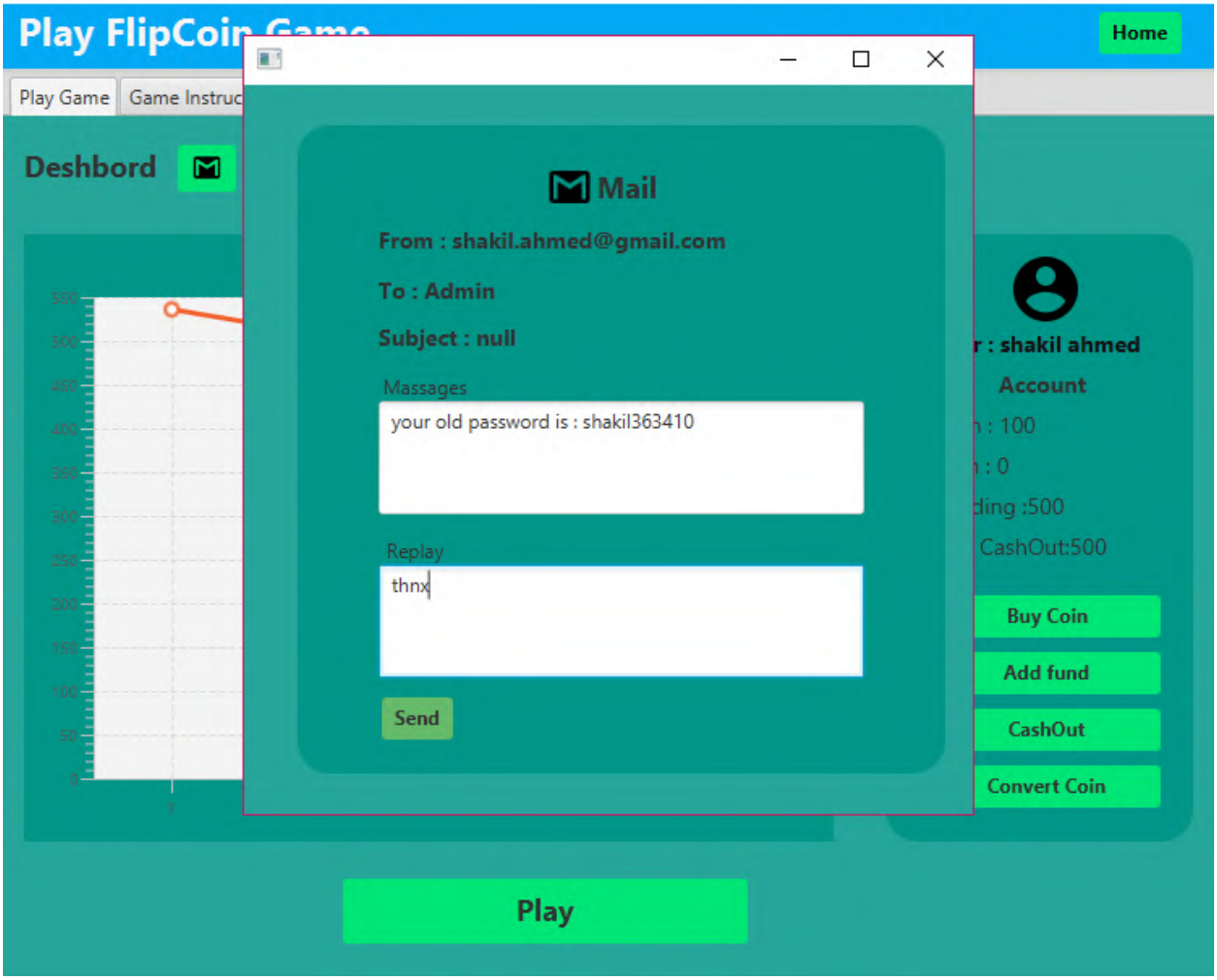
I forget my Flip coin-Games account password
My username is : shakil363410
And FCG email id is :
Shakil.ahmed@gmail.com

Admin Response



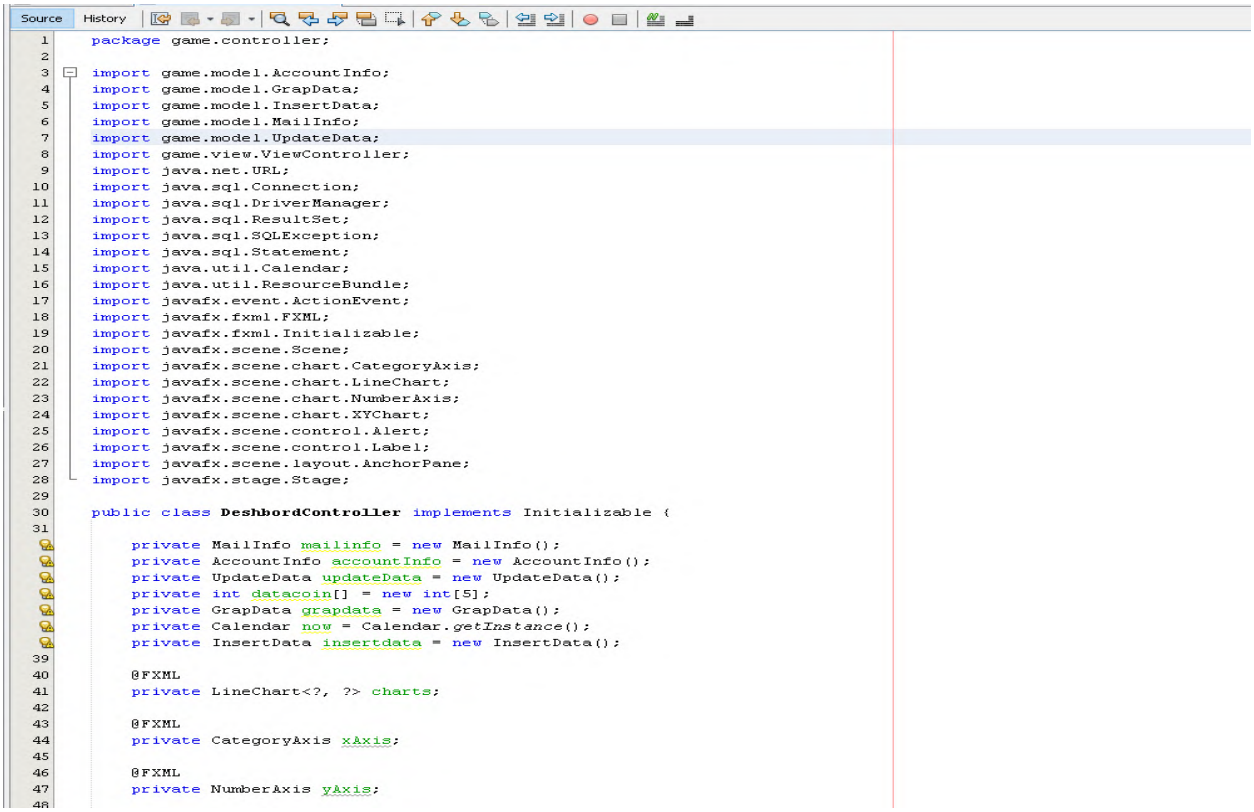
Admin gives response to user that your old password is :shakil363410
And sending to user via email address.

User gets Feedback



User gets feedback from admin that user old password is :shakil363410

Dashboard Control Views Code



```
Source  History
1  package game.controller;
2
3  import game.model.AccountInfo;
4  import game.model.GrapData;
5  import game.model.InsertData;
6  import game.model.MailInfo;
7  import game.model.UpdateData;
8  import game.view.ViewController;
9  import java.net.URL;
10 import java.sql.Connection;
11 import java.sql.DriverManager;
12 import java.sql.ResultSet;
13 import java.sql.SQLException;
14 import java.sql.Statement;
15 import java.util.Calendar;
16 import java.util.ResourceBundle;
17 import javafx.event.ActionEvent;
18 import javafx.fxml.FXML;
19 import javafx.fxml.Initializable;
20 import javafx.scene.Scene;
21 import javafx.scene.chart.CategoryAxis;
22 import javafx.scene.chart.LineChart;
23 import javafx.scene.chart.NumberAxis;
24 import javafx.scene.chart.XYChart;
25 import javafx.scene.control.Alert;
26 import javafx.scene.control.Label;
27 import javafx.scene.layout.AnchorPane;
28 import javafx.stage.Stage;
29
30 public class DeshbordController implements Initializable {
31
       private MailInfo mailinfo = new MailInfo();
       private AccountInfo accountInfo = new AccountInfo();
       private UpdateData updateData = new UpdateData();
       private int datacoin[] = new int[5];
       private GrapData grapdata = new GrapData();
       private Calendar now = Calendar.getInstance();
       private InsertData insertdata = new InsertData();
39
40     @FXML
41     private LineChart<?, ?> charts;
42
43     @FXML
44     private CategoryAxis xAxis;
45
46     @FXML
47     private NumberAxis yAxis;
```

```
48
49      @FXML
50      private Label name;
51
52      @FXML
53      private Label cash;
54
55      @FXML
56      private Label coin;
57
58      @FXML
59      private AnchorPane deshbord;
60
61      @FXML
62      void play(ActionEvent event) {
63          ViewController vc = new ViewController();
64          vc.setFxmlFile("Play.fxml");
65          deshbord.getChildren().setAll(vc.getFxmlFile());
66      }
67
68      @FXML
69      void addfund(ActionEvent event) {
70          ViewController vc = new ViewController();
71          vc.setFxmlFile("AddFund.fxml");
72          deshbord.getChildren().setAll(vc.getFxmlFile());
73      }
74
75      @FXML
76      void buycoin(ActionEvent event) {
77          ViewController vc = new ViewController();
78          vc.setFxmlFile("BuyCoin.fxml");
79          deshbord.getChildren().setAll(vc.getFxmlFile());
80      }
81
82      @FXML
83      void cashout(ActionEvent event) {
84          ViewController vc = new ViewController();
85          vc.setFxmlFile("CashOut.fxml");
86          deshbord.getChildren().setAll(vc.getFxmlFile());
87      }
88
89      @FXML
90      void convartcoin(ActionEvent event) {
91          ViewController vc = new ViewController();
```

```
91          ViewController vc = new ViewController();
92          vc.setFxmlFile("ConvertCoin.fxml");
93          deshbord.getChildren().setAll(vc.getFxmlFile());
94      }
95
96      @FXML
97      void mail(ActionEvent event) {
98          if (mailinfo.getReciver() != null && mailinfo.getCount() == 3) {
99              Stage stage = new Stage();
100             mailinfo.setMailStage(stage);
101             ViewController getPane = new ViewController();
102             getPane.setFxmlFile("Mail.fxml");
103             AnchorPane root = getPane.getFxmlFile();
104             Scene scene = new Scene(root);
105             stage.setScene(scene);
106             stage.show();
107
108         } else {
109             Alert alert = new Alert(Alert.AlertType.INFORMATION);
110             alert.setTitle("Message");
111             alert.setHeaderText(null);
112             alert.setContentText("No Message Here!");
113             alert.showAndWait();
114
115         }
116
117     }
118
119     @Override
120     public void initialize(URL url, ResourceBundle rb) {
121
122         chartData();
123         getCoinDate();
124         XYChart.Series series = new XYChart.Series();
125         series.getData().add(new XYChart.Data("1", datacoin[0]));
126         series.getData().add(new XYChart.Data("2", datacoin[1]));
127         series.getData().add(new XYChart.Data("3", datacoin[2]));
128         series.getData().add(new XYChart.Data("4", datacoin[3]));
129         series.getData().add(new XYChart.Data("5", datacoin[4]));
130
131         charts.getData().add(series);
132         name.setText("User : " + accountInfo.getName());
133         cash.setText("Cash : " + String.valueOf(accountInfo.getAmount()));
134         coin.setText("Coin : " + String.valueOf(accountInfo.getCoin()));
135         mailinfo.setPage(3);
136     }
137
138     public void chartData() {
139
```

Admin Controller Views Code

```java
package game.controller;

import game.model.MailInfo;
import game.model.UserInfo;
import game.view.ViewController;
import java.net.URL;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ResourceBundle;
import java.util.function.Predicate;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.collections.transformation.FilteredList;
import javafx.collections.transformation.SortedList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Label;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.input.KeyEvent;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;

public class AdminController implements Initializable {

    MailInfo mailinfo = new MailInfo();
    private ObservableList<UserInfo> data = FXCollections.observableArrayList();
    FilteredList<UserInfo> filteredData=new FilteredList<>(data,e->true);
    @FXML
    private TableView<UserInfo> adminTabel;

    @FXML
    private TableColumn<UserInfo, Integer> idCol;

    @FXML
    private TableColumn<UserInfo, String> userCol;

    @FXML
    private TableColumn<UserInfo, String> passCol;
      private TableColumn<UserInfo, String> passCol;

      @FXML
      private TableColumn<UserInfo, String> emailCol;

      @FXML
      private TextField idField;

      @FXML
      private TextField usernameField;

      @FXML
      private TextField passwordField;

      @FXML
      private TextField emailField;

      @FXML
      private TextField searchField;

      @FXML
      void onSearchField(KeyEvent event) {
          searchField.textProperty().addListener((observableValue, oldValue, newValue) -> {
              filteredData.setPredicate((Predicate<? super UserInfo>) user -> {
                  if (newValue == null || newValue.isEmpty()) {
                      return true;
                  }
                  String lowerCaseFilter = newValue.toLowerCase();
                   if (user.getUser().toLowerCase().contains(lowerCaseFilter)) {
                      return true;
                  }
                  return false;
              });
          });
          SortedList<UserInfo> sortedData = new SortedList<>(filteredData);
          sortedData.comparatorProperty().bind(adminTabel.comparatorProperty());
          adminTabel.setItems(sortedData);
      }

      @FXML
      void onDelete(ActionEvent event) {

          UserInfo userInfo = adminTabel.getSelectionModel().getSelectedItem();

          if (userInfo != null) {
              adminTabel.getItems().removeAll(adminTabel.getSelectionModel().getSelectedItem());
              delete(userInfo.getId());
              lavel.setText("Delete Sucessfull ");
          } else {
```

```
97          } else {
98              lavel.setText("Your not Secleted Any Item ");
99          }
100     }
101     @FXML
102     private Label lavel;
103
104     @FXML
105     void onUpdate(ActionEvent event) {
106         UserInfo userInfo = adminTabel.getSelectionModel().getSelectedItem();
107         userInfo.setEmail(emailField.getText());
108         userInfo.setId(Integer.parseInt(idField.getText()));
109         userInfo.setPass(passwordField.getText());
110         userInfo.setUser(usernameField.getText());
111         if (userInfo != null) {
112             updateUser(userInfo.getId(), userInfo.getUser(), userInfo.getPass(), userInfo.getEmail());
113             lavel.setText("Update Sucessful ");
114
115         } else {
116             lavel.setText("Your not Secleted Any Item ");
117         }
118         data.clear();
119         UserDataTabel();
120     }
121
122     @FXML
123     void onGetdata(MouseEvent event) {
124         UserInfo userInfo = adminTabel.getSelectionModel().getSelectedItem();
125         if (userInfo != null) {
126             idField.setText(String.valueOf(userInfo.getId()));
127             usernameField.setText(userInfo.getUser());
128             passwordField.setText(userInfo.getPass());
129             emailField.setText(userInfo.getEmail());
130
131         } else {
132             lavel.setText("Your not Secleted Any Item ");
133         }
134
135     }
136
137     @FXML
138     void mail(ActionEvent event) {
139         if (mailinfo.getReciver() != null && mailinfo.getCount() == 2) {
140             Stage stage = new Stage();
141             mailinfo.setMailStage(stage);
142             ViewController getPane = new ViewController();
143             getPane.setFxmlFile("Mail.fxml");
144             AnchorPane root = getPane.getFxmlFile();
145             Scene scene = new Scene(root);
145                 Scene scene = new Scene(root);
146                 stage.setScene(scene);
147                 stage.show();
148
149             } else {
150                 Alert alert = new Alert(Alert.AlertType.INFORMATION);
151                 alert.setTitle("Message");
152                 alert.setHeaderText(null);
153                 alert.setContentText("No Message Here!");
154                 alert.showAndWait();
155
156             }
157
158     }
159
160     @Override
161     public void initialize(URL url, ResourceBundle rb) {
162         idCol.setCellValueFactory(
                        new PropertyValueFactory<UserInfo, Integer>("id")
164         );
165         userCol.setCellValueFactory(
                        new PropertyValueFactory<UserInfo, String>("user")
167         );
168
169         passCol.setCellValueFactory(
                        new PropertyValueFactory<UserInfo, String>("pass")
171         );
172         emailCol.setCellValueFactory(
                        new PropertyValueFactory<UserInfo, String>("email")
174         );
175
176         UserDataTabel();
177         adminTabel.setItems(data);
178         mailinfo.setPage(3);
179     }
180
181     private Connection connect() {
182         String url = "jdbc:sqlite:game.db";
183         Connection conn = null;
184         try {
185             try {
186                 Class.forName("org.sqlite.JDBC");
187             } catch (ClassNotFoundException ex) {
188             }
189             conn = DriverManager.getConnection(url);
190         } catch (SQLException e) {
191             System.out.println(e.getMessage());
192         }
193         return conn;
```

```
193         return conn;
194     }
195 
196     public void UserDataTabel() {
197         String sql = "SELECT * FROM user_tbl";
198         try (Connection conn = this.connect();
199                 Statement stmt = conn.createStatement();
200                 ResultSet rs = stmt.executeQuery(sql)) {
201             // loop through the result set
202             while (rs.next()) {
203                 data.add(new UserInfo(rs.getInt("id"), rs.getString("username"), rs.getString("password"), rs.getString("email")));
204             }
205         } catch (SQLException e) {
206             System.out.println(e.getMessage());
207         }
208 
209     }
210 
211     public void delete(int id) {
212         String sql = "DELETE FROM user_tbl WHERE id = ?";
213 
214         try (Connection conn = this.connect();
215                 PreparedStatement pstmt = conn.prepareStatement(sql)) {
216             pstmt.setInt(1, id);
217             pstmt.executeUpdate();
218 
219         } catch (SQLException e) {
220             System.out.println(e.getMessage());
221         }
222     }
223 
224     public void updateUser(int id, String user, String pass, String email) {
225         String sql = "UPDATE user_tbl SET username = ? ,"
226                 + "password = ?,"
227                 + "email = ?"
228                 + "WHERE id = ?";
229 
230         try (Connection conn = this.connect();
231                 PreparedStatement pstmt = conn.prepareStatement(sql)) {
232             // set the corresponding param
233             pstmt.setString(1, user);
234             pstmt.setString(2, pass);
235             pstmt.setString(3, email);
236             pstmt.setInt(4, id);
237             // update
238             pstmt.executeUpdate();
239         } catch (SQLException e) {
240             System.out.println(e.getMessage());
```