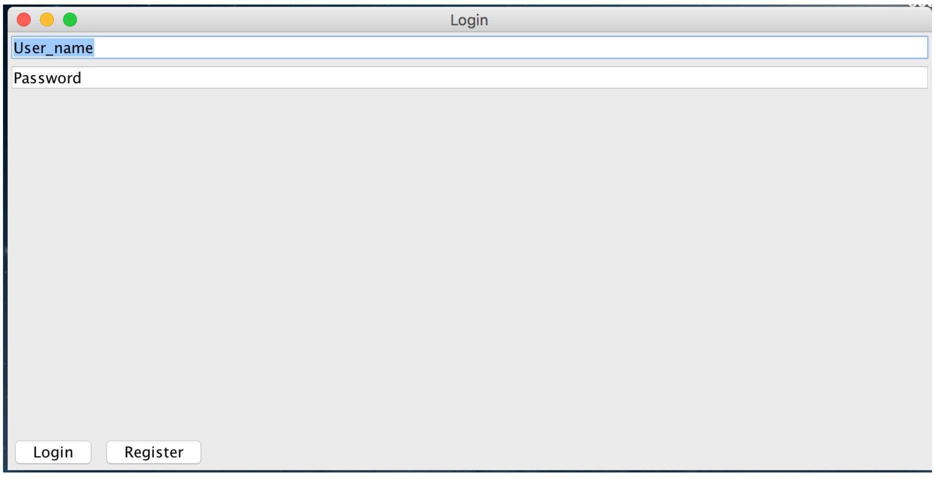
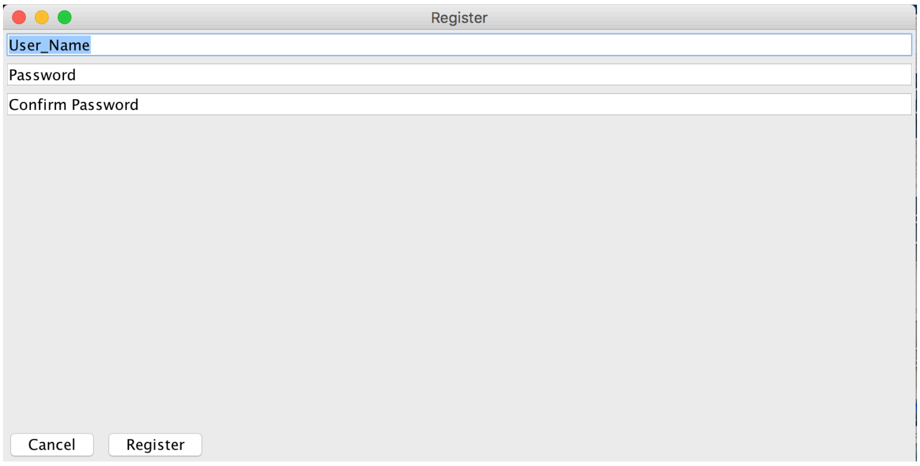
COMP3402 HW01

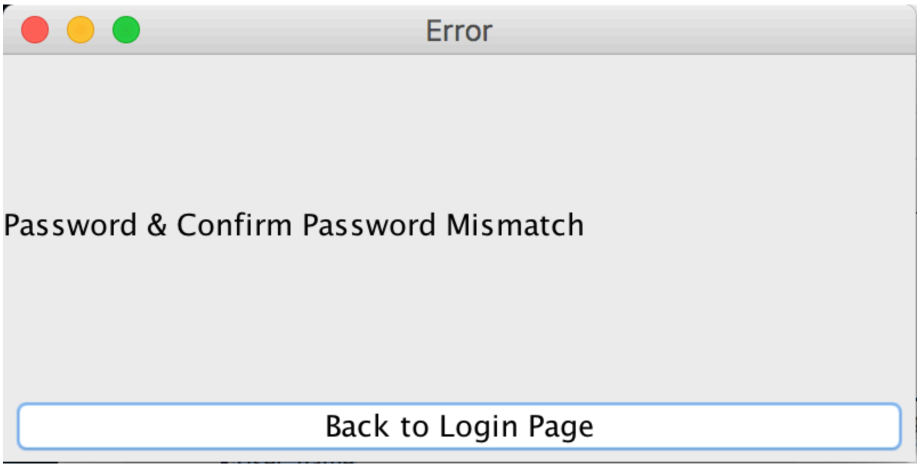
1. Screenshots of client program:
   1. Login Page:



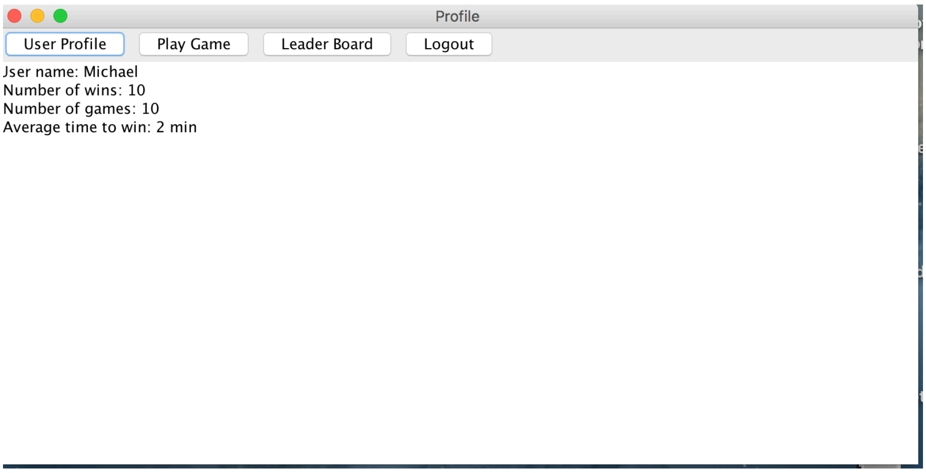
   2. Register Page:



   3. Error Page: Notice that “Error Page” pops up when a user types mismatch password & confirm password on Registration Page or the requesting user name has been taken.

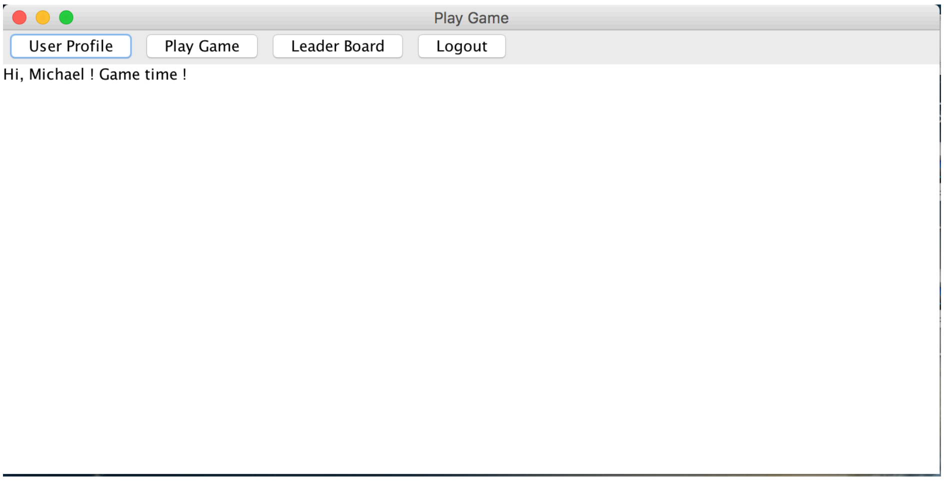


Password & Confirm Password Mismatch



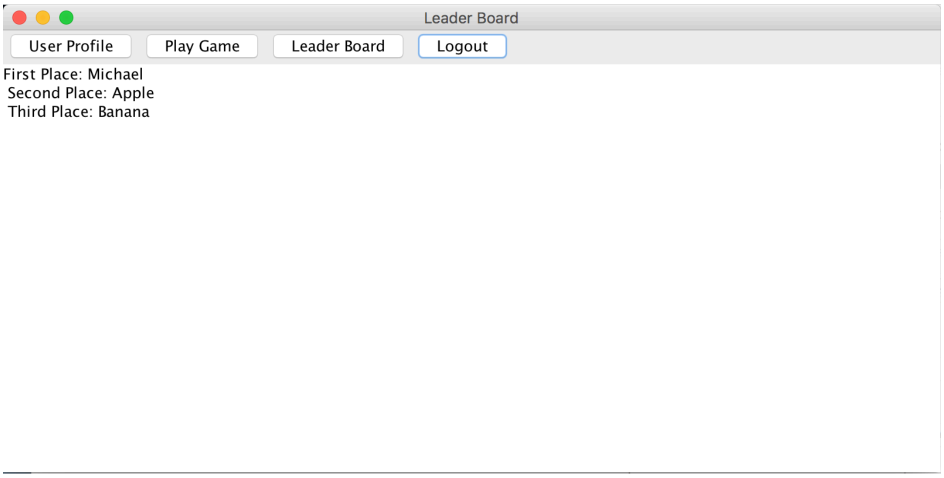
Jser name: Michael
Number of wins: 10
Number of games: 10
Average time to win: 2 min

4. Profile Page:
5. Play Game Page:



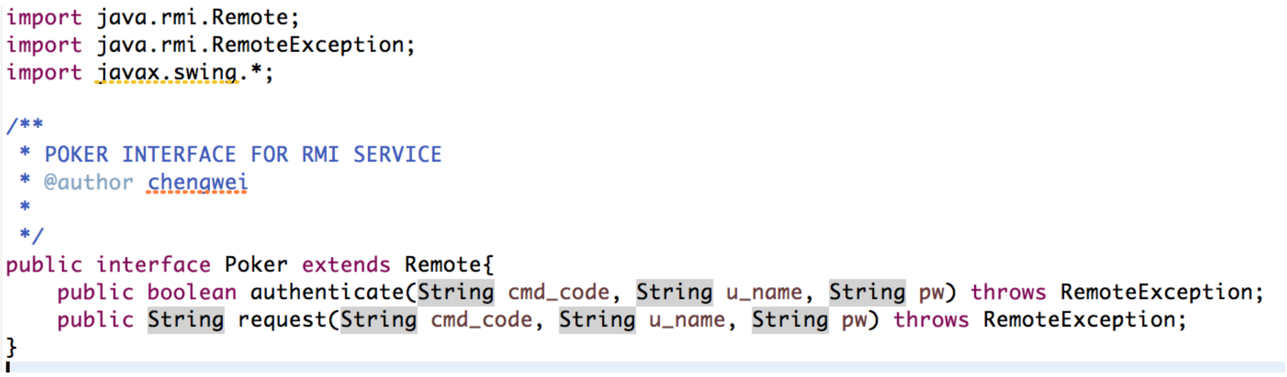
Hi, Michael ! Game time !

6. Leader Page:



First Place: Michael
Second Place: Apple
Third Place: Banana

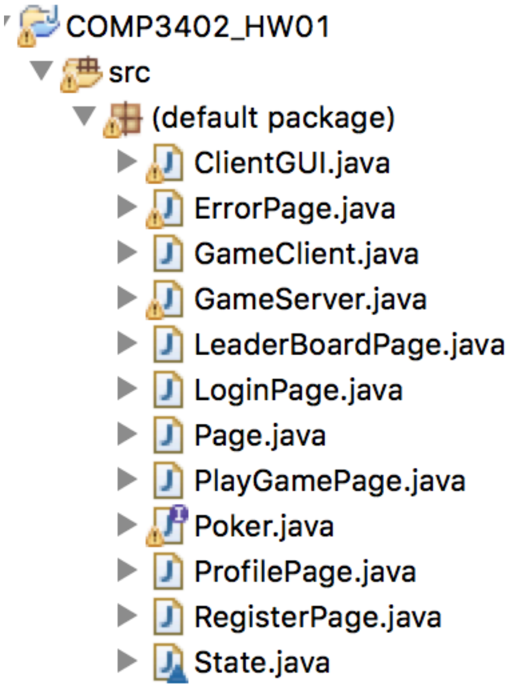
2. **Server program:**
    1. Poker interface



```
import java.rmi.Remote;
import java.rmi.RemoteException;
import javax.swing.*;

/**
 * POKER INTERFACE FOR RMI SERVICE
 * @author chengwei
 *
 */
public interface Poker extends Remote{
    public boolean authenticate(String cmd_code, String u_name, String pw) throws RemoteException;
    public String request(String cmd_code, String u_name, String pw) throws RemoteException;
}
```

    2. This interface is implemented by both server and client program.
    3. “authenticate” method provides various of authentication services given difference cmd_code. For instance, login verification service ,logout service …
    4. “request” method retrieves server response given difference cmd_code. For instance, this method returns user’s profile given “profile” command and corresponding user_name and password.
3. The following section briefly describes the feature of each Java classes used in this phase
    1. **Catalog**:



```
COMP3402_HW01
  src
    (default package)
      ClientGUI.java
      ErrorPage.java
      GameClient.java
      GameServer.java
      LeaderBoardPage.java
      LoginPage.java
      Page.java
      PlayGamePage.java
      Poker.java
      ProfilePage.java
      RegisterPage.java
      State.java
```

    2. **ClientGUI class**: This class models the user interface and logic of client program.
    3. **Page class & all sub-classes**: The top-level parent of following classes (LoginPage.java, ErrorPage.java, LeaderBoarderPage.java, PlayGamePage.java, ProfilePage.java, RegisterPage.java). The design methodology behind this is polymorphism. These sub-classes will be stored in a ArrayList during initialization, then the client will decide which Page to show based of different state transition logic.
    4. **Poker interface**: This interface handles all RMI functionality
    5. **State class**: This class modes the state transition logic on the client side.