SOLID PRINCIPLES ASSIGNMENT

SINGLE RESPONSIBILITY PRINCIPLE (SRP)

User Class

The User class is responsible for encapsulating user information, providing methods to retrieve the users username, email and password. It adheres to the SRP by focusing solely on the management of User and associated data.

```
package com.game.entity;

public class User {

 private String username;
 private String email;
 private String password;

 public String getUsername() {
return username;
}

public void setUsername(String username) {
this.username = username;
}

public String getEmail() {
return email;
}

public void setEmail String email) {
this.email = email;
}

public String getPassword() {
return password;
```

```
}

public void setPassword(String password) {
this.password = password;
}

public User(String username, String email, String password) {
 this.username = username;
 this.email = email;
 this.password = password;
 }
```

OPEN-CLOSED PRINCIPLE (OCP)

Regular Account Service Class, Login Service Interface, Register User Interface

The `Regular Account Service` Class implements `login` and `register_user` interface for normal users. To add more types of users you can just implement the interfaces. For a premium user the same interfaces can be implemented for them without modifying anything.All user types are sub classes of `Account Service` class which can host all the common methods.

```
package com.game.service;

import java.util.Scanner;

import com.game.entity.User;
import com.game.interfaces.LoginUser;
import com.game.interfaces.PhoneOtp;
import com.game.interfaces.RegisterUser;

public class RegularAccountService extends AccountService
implements RegisterUser,LoginUser,PhoneOtp{
```

```java
@Override
public User registerUser() {
String username=null;
String email=null;
String password=null;
String Otp=null;
Scanner scanner = new Scanner(System.in);
System.out.println("Enter User Name");
username=scanner.next();
System.out.println("Enter Email");
email= scanner.next();
System.out.println("Enter Password");
password =scanner.next();
phoneOtp();
System.out.println("Enter OTP");
Otp=scanner.next();
User user = new User(username,email,password);
System.out.println("Account Created..");
return user;
}

@Override
public void loginUser() {
String username=null;
String password=null;
Scanner scanner = new Scanner(System.in);
System.out.println("Enter User Name");
username=scanner.next();
System.out.println("Enter Password");
password= scanner.next();
System.out.println("Login Completed\n"+username+" loged in");
}

@Override
public void phoneOtp() {
int randomNumber= (int) Math.floor(Math.random()*1000);
System.out.println("Your OTP is "+randomNumber);
```

```java
package com.game.interfaces;

public interface LoginUser {
public void loginUser();
}
```

```java
package com.game.interfaces;

import com.game.entity.User;

public interface RegisterUser {
public User registerUser();


}
```

LISKOV SUBSTITUTION PRINCIPLE (LSP)

PremiumUser Class

The `PremiumUser` class extends the `User` class. The premium user class can be substituted where ever the user class is used. An example for this is the show user method which user is logged in and users type.

```java
package com.game.entity;

public class PremiumUser extends User{

private boolean isPremium;

public PremiumUser(String username, String email, String
password, boolean isPremium) {
super(username, email, password);
```

```
this.isPremium = isPremium;
}

public boolean isPremium() {
return isPremium;
}

public void setPremium(boolean isPremium) {
this.isPremium = isPremium;
}

}
```

INTERFACE SEGREGATION PRINCIPLE (ISP)

PhoneOtp Interface

The `phoneOtp` Interface is implemeted to `Regular Account Service` Class . The otp authentication methods can be clubbed together but different user types like premium members can only access other otp methods like `emailOtp` so differnt interfaces are created for them instead of a single one.

```
package com.game.interfaces;

public interface PhoneOtp {

public void phoneOtp();


}
```

```
package com.game.service;

import java.util.Scanner;
```

```java
import com.game.entity.User;
import com.game.interfaces.LoginUser;
import com.game.interfaces.PhoneOtp;
import com.game.interfaces.RegisterUser;

public class RegularAccountService extends AccountService
implements RegisterUser,LoginUser,PhoneOtp{

@Override
public User registerUser() {
String username=null;
String email=null;
String password=null;
String Otp=null;
Scanner scanner = new Scanner(System.in);
System.out.println("Enter User Name");
username=scanner.next();
System.out.println("Enter Email");
email= scanner.next();
System.out.println("Enter Password");
password =scanner.next();
phoneOtp();
System.out.println("Enter OTP");
Otp=scanner.next();
User user = new User(username,email,password);
System.out.println("Account Created..");
return user;
}

@Override
public void loginUser() {
String username=null;
String password=null;
Scanner scanner = new Scanner(System.in);
System.out.println("Enter User Name");
username=scanner.next();
System.out.println("Enter Password");
password= scanner.next();
System.out.println("Login Completed\n"+username+" loged in");
}
```

```
@Override
public void phoneOtp() {
int randomNumber= (int) Math.floor(Math.random()*1000);
System.out.println("Your OTP is "+randomNumber);
```

DEPENDENCY INVERSION PRINCIPLE (DIP)

NotificationManager class

The `NotificationManager` class adheres to the Dependency Inversion Principle by depending on abstraction of `NotificationService` Interface rather than concrete implementations like `EmailNotificationService`. This promotes flexibility and ease of extension.

```
package com.game.service;

import com.game.entity.User;
import com.game.interfaces.NotificationService;

public class NotificationManager {

private NotificationService notificationService;

public NotificationManager(NotificationService
notificationService) {
this.notificationService = notificationService;
}

public void sendNotificationToUser(User user, String message) {
```

```
String fullMessage = "Hello, " + user.getUsername() + " " +
message;
notificationService.sendNotification(fullMessage);
}


}
```

```
package com.game.interfaces;

public interface NotificationService {
public void sendNotification String messsage);

}
```

```
package com.game.service;

import com.game.interfaces.NotificationService;

public class EmailNotificationService implements
NotificationService {

@Override
public void sendNotification String messsage) {
System.out.println("Notification message:"+messsage);

}
```

```


}
```