User UI

This is the interface that users will use to interact with SAML. The user mostly navigates through the interface by looking at a numbered list of action and type in their numerical choice. There are some instances where the user does not do that (for example edit e-mail) the user will be prompted to type in a string that has some restrictions (for example, an email must have a "@" sign and "."). Through this program we provide the user a select amount of actions. These include View/Edit personal info, View/Edit friends, Discover new media (Buy/Rent media) and list the media the user has and allows them to rate them on a scale of 0-10.

The User UI is based upon six components. The main one is Welcome.java which contains the main method. Welcome handles the Oracle Login, the user login and the main user interface where the user is presented with option on what they can do with SAML. Depending on which option they choose, another class is called to complete their actions. For example, if the user selects the first option, "View/Edit personal info", Welcome calls the class "one" to complete the user's action. Similarly if the user selects option two, class "two" is called to complete the user's action. This pattern repeats for options three and four. Options five and six are handled by welcome itself as Welcome handles the Oracle login and user login, therefore, the only class that can handle option five (logout) or six (exit) is Welcome itself. In addition to those classes, that interact with the user, there is a behind the scene class called InputOutput. It is important as it handles commonly used functions such as get user input to display warning messages when an error happens. Not only does it contain functions that are used in Java, it also contain methods that deal with SQL io's. Therefore, even though it does not interact with the user directly, this class provides functions that make the program work.

For testing, the User UI (written in Java) is completed before any SQL is implemented. To test the functionality of the User UI, we hard-coded in some data values to simulate a really small database. We then stress the code with both expected input and unexpected input (to test the code's capabilities in terms of error handling). An example of an expected input test is when the menu presents the user options one through six, we give an input within the range, for example "3" and see if the code reacts properly. An example of an unexpected input test is when the menu presents the user options one through six and we give an input that is out of the range, for example "a","-1","&" to see if the program can gracefully handle the error. This ensures that the Java code is as stable as possible. After we were convinced that the code is stable, we remove the hard-coded data values and fill those variables with data that will be filled by SQL queries. By making the Java code as stable as possible, any error that pops up at this stage we could be sure that the problem exists our SQL related lines. This is a huge time saver as we do not have to guess if the problem is in the User UI end or the SQL end.

Details/relationships between classes:
Welcome.class
Function: Oracle Login/Logout, User Login/Logout, Main Menu
Menu:

Option 1: Invokes one.class, when it terminates, we come back to welcome
Option 2: Invokes two.class, when it terminates, we come back to welcome
Option 3: Invokes three.class, when it terminates, we come back to welcome
Option 4: Invokes four.class, when it terminates, we come back to welcome
Option 5: User Logout, this basically brings us back to the place where we prompt for user login
Option 6: Oracle logout, exits program, Welcome is terminated

one.class - Invoked by Welcome
Function: View personal info, edit E-mail, edit E-mail visibility

two.class - Invoked by Welcome
Function: View friends, add friends suggested friends, remove friends

three.class - Invoked by Welcome
Function: Purchase, Rent

four.class - Invoked by Welcome
Function: View owned movie, rate owned movie

Construction of the admin UI exactly reflects the user's UI for consistency in debugging and pair programming efficacy. The structure in classes is also reflected, but there is no option for users to log in, and there is no fourth class/option. A logout from the home screen is a logout from Oracle.