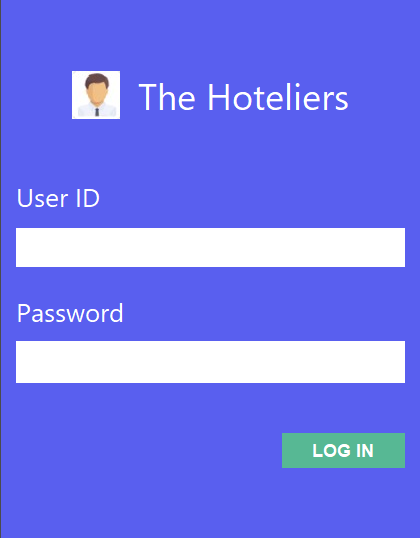
Criterion C – Development

**Fulfilling the Success Criteria**

I have achieved all success criteria by creating the application on NetBeans (Java) and all the source code is attached in **Appendix-2** for reference.

**Linked with success criteria 1:**



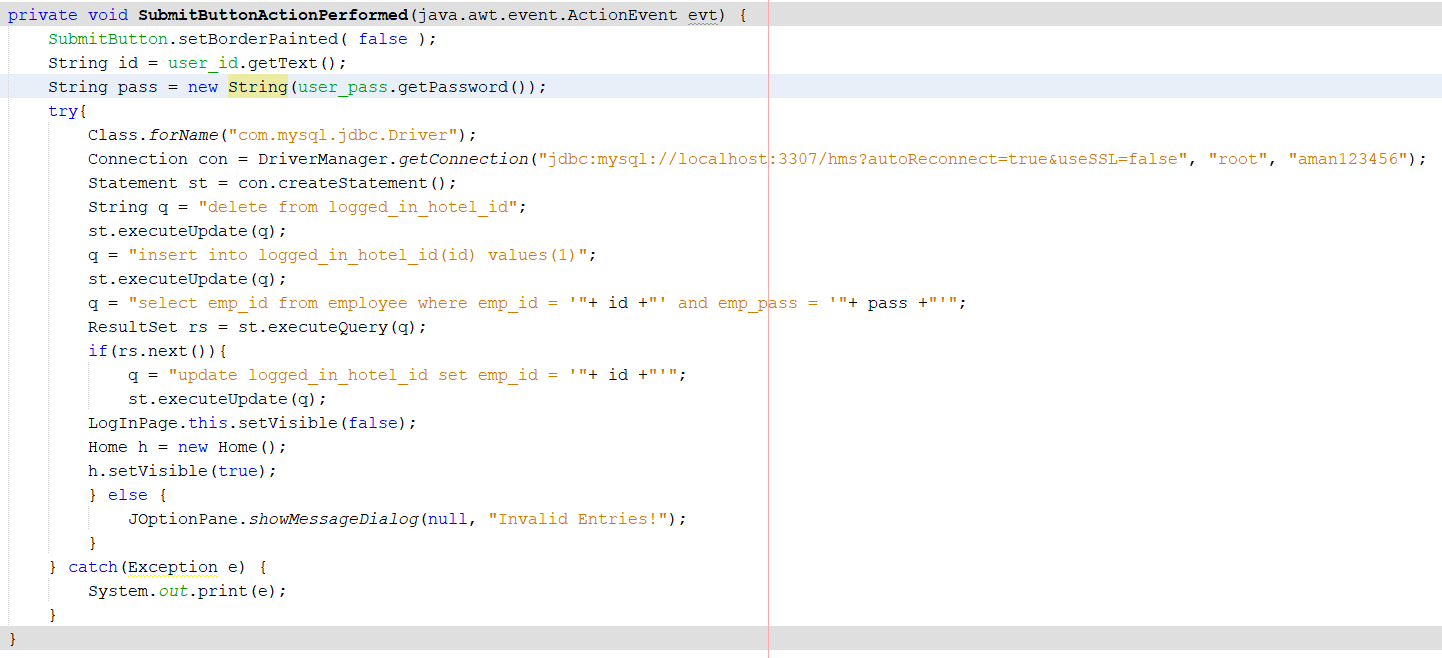
Text field to enter user ID

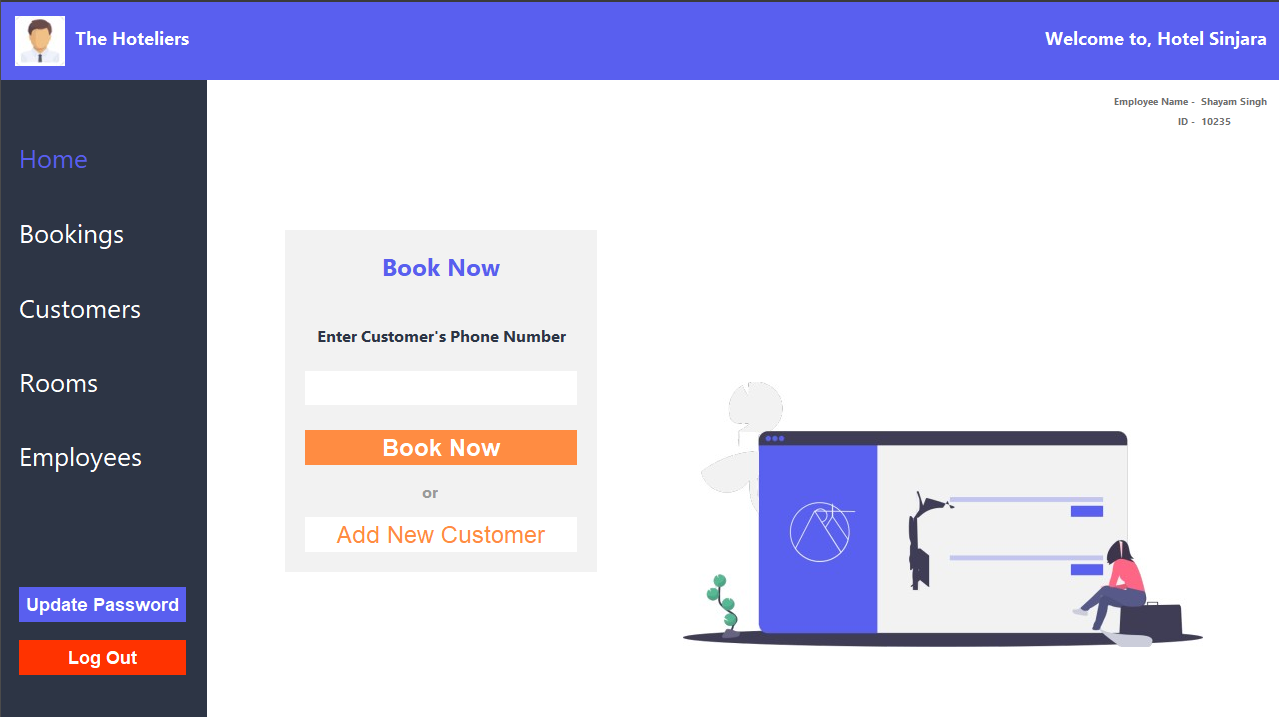
Button to Login account

Password field to enter user’s Password

The above form allows administrators and employees to login to the application. The login credentials are compared with those in the database (phpMyAdmin). There is also an option that allows the users to change their passwords in case they forget it (Success criteria 11). Here is the code that facilitates the login form:

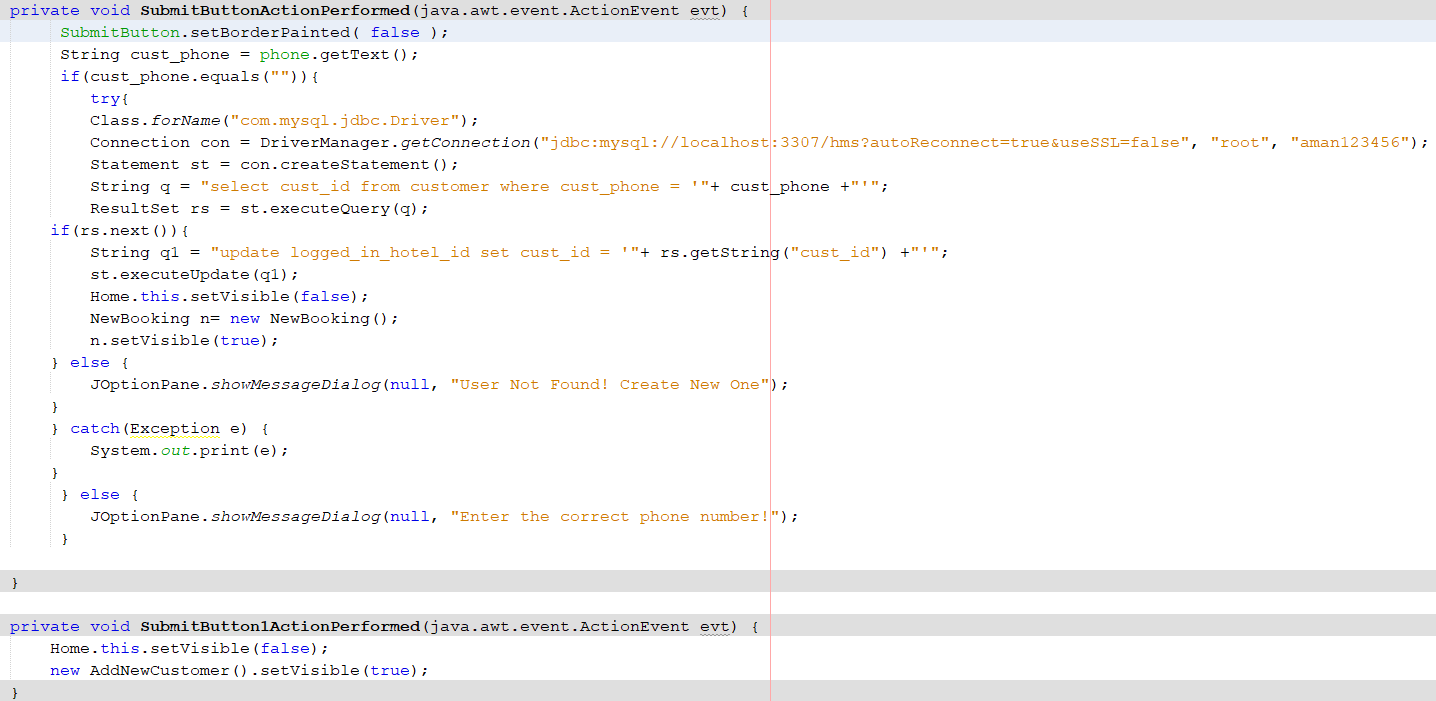
**Linked with success criteria 2, 3, 11, 17:**





Book Now Button on HOME page.

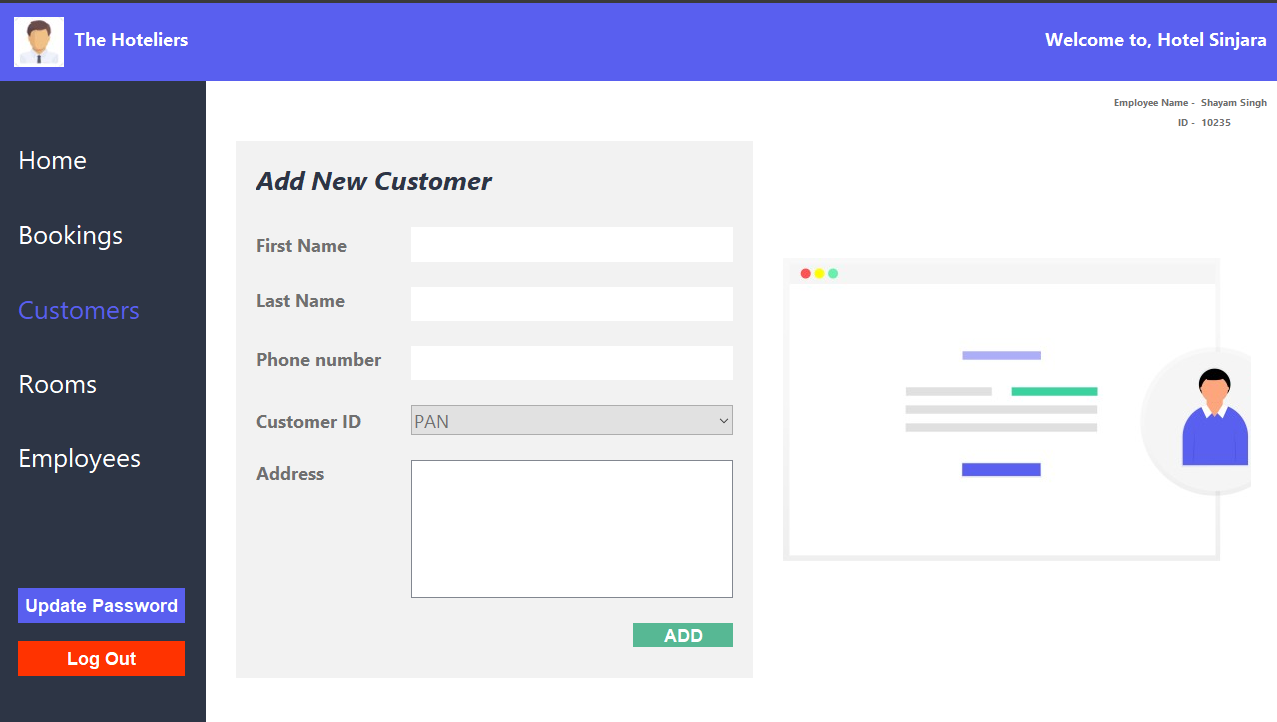
Add New Customer option on homepage

****Here is the code that facilitates Book Now and Add New Customer Option:

Validation Check.

Error message displayed as a JOptionPane warning that user not found / invalid phone number

**Linked with success criteria 4:**



Add to database button

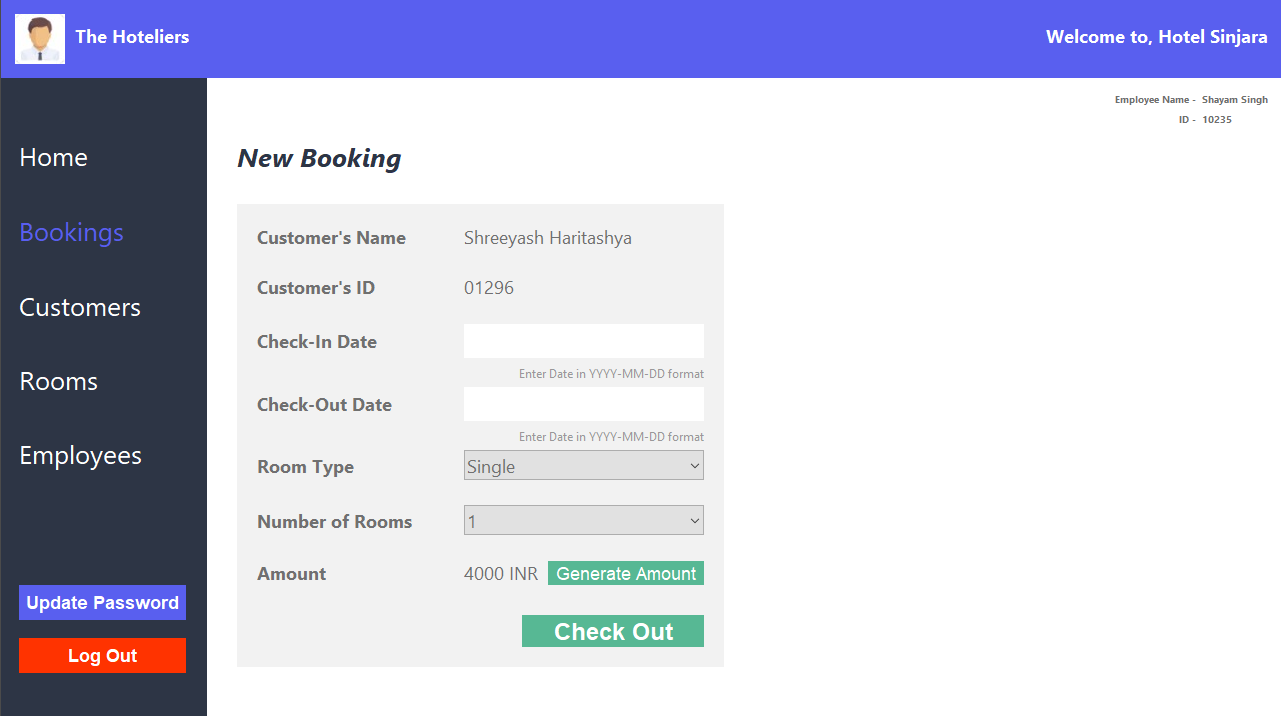
JTextFields to accept new customer’s details

JComboBox to accept new customer’s ID

Here is the code that facilitates Add New Customer:



**Linked with success criteria 5:**



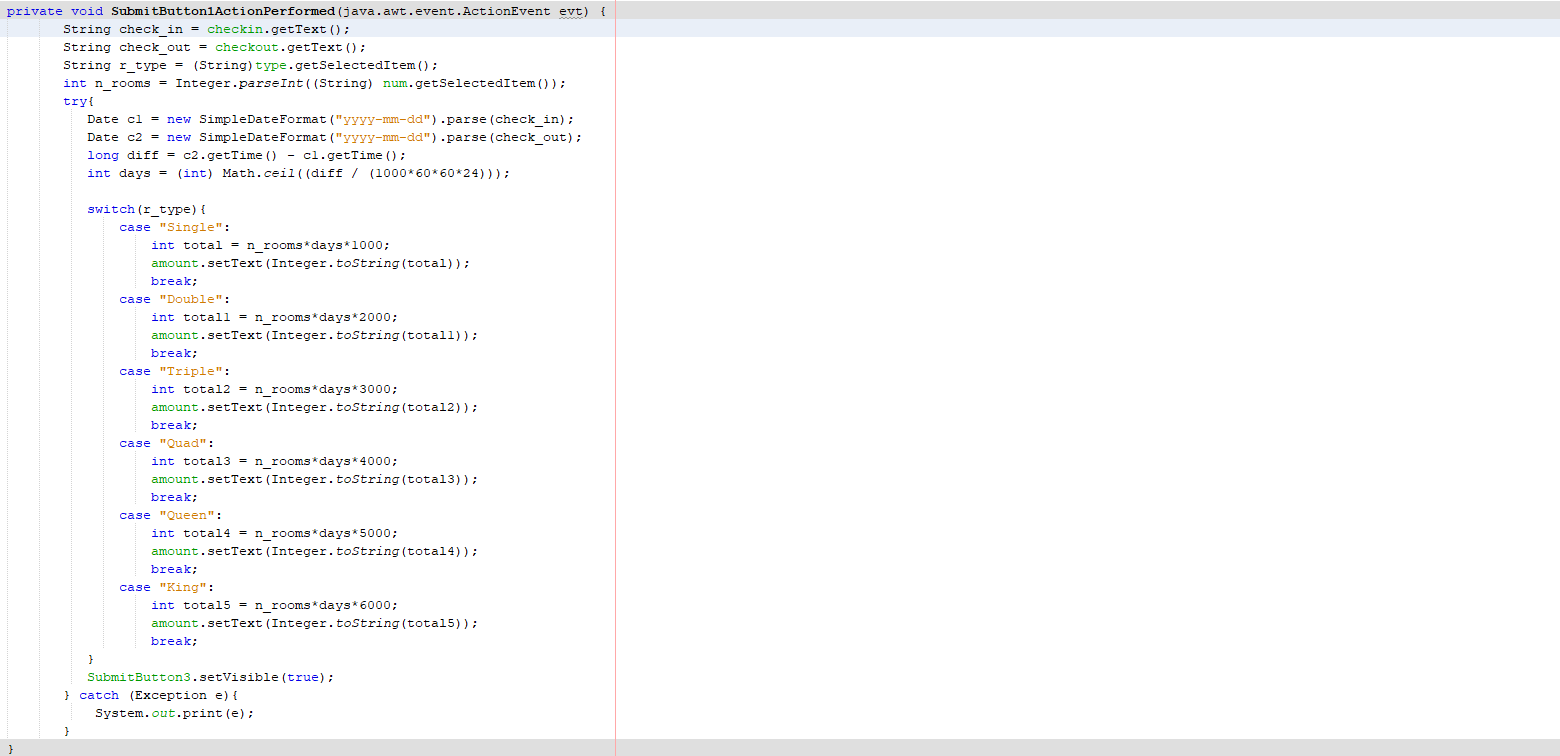
JButton to generate Amount

JButton to Confirm Booking

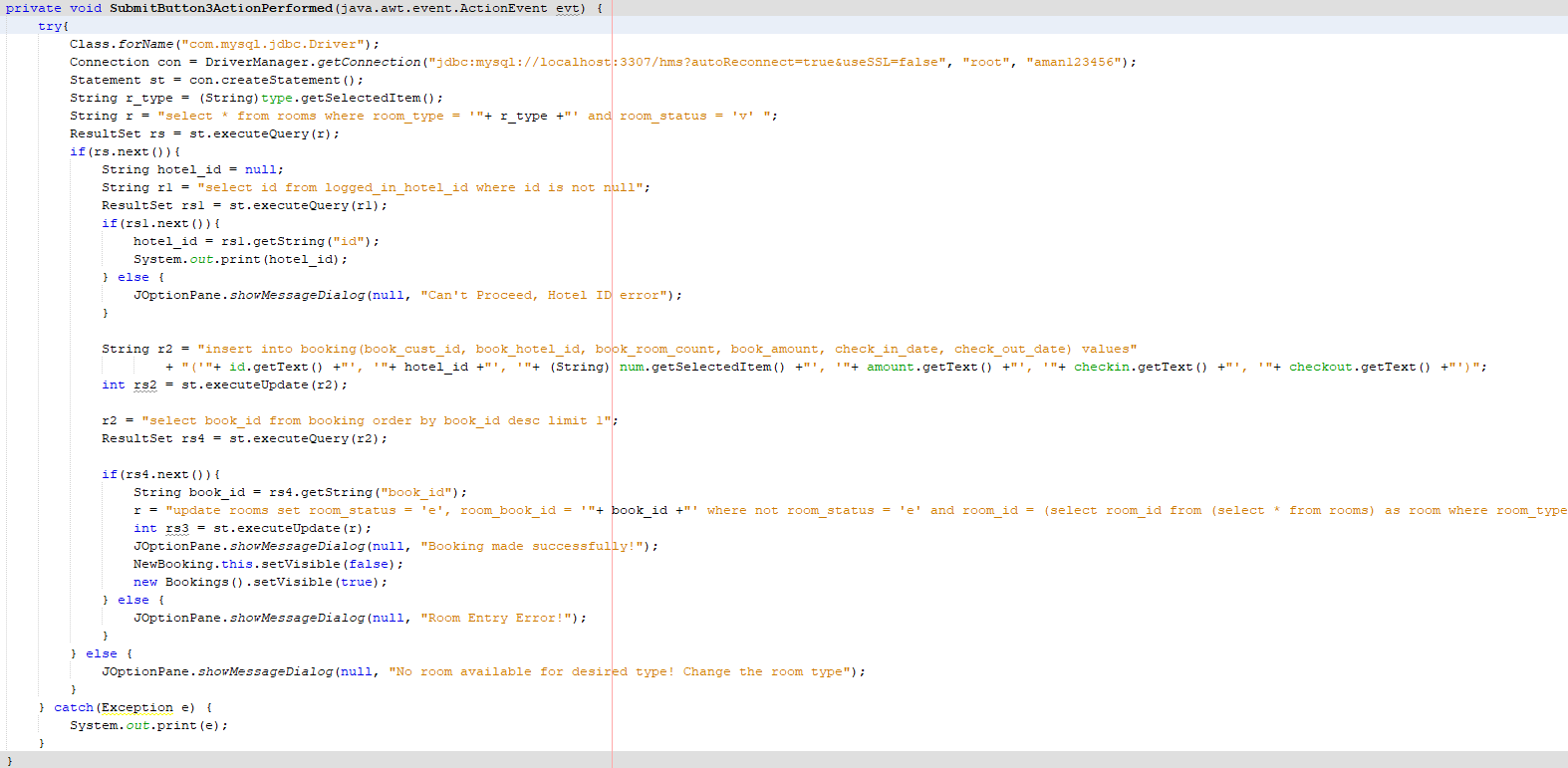
JTextFields to accept new customer’s details

JComboBox to accept Room Type and Number of rooms to book

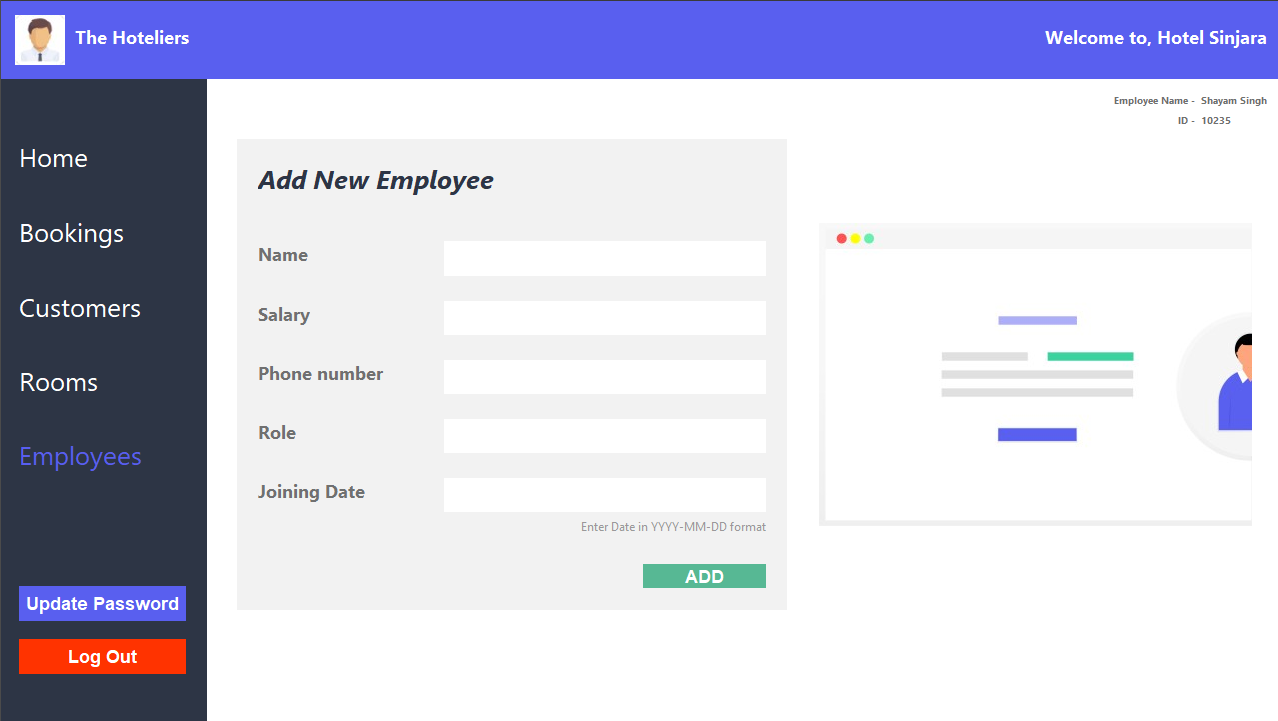
Here is the code that facilitates New Booking:

//Generate Amount Button Code:

//Check-Out Button Code:

****

**Linked with success criteria 6:**



JButton to Add employee to database

JTextFields to accept new Employee’s details

Here is the code that facilitates Add New Employee:



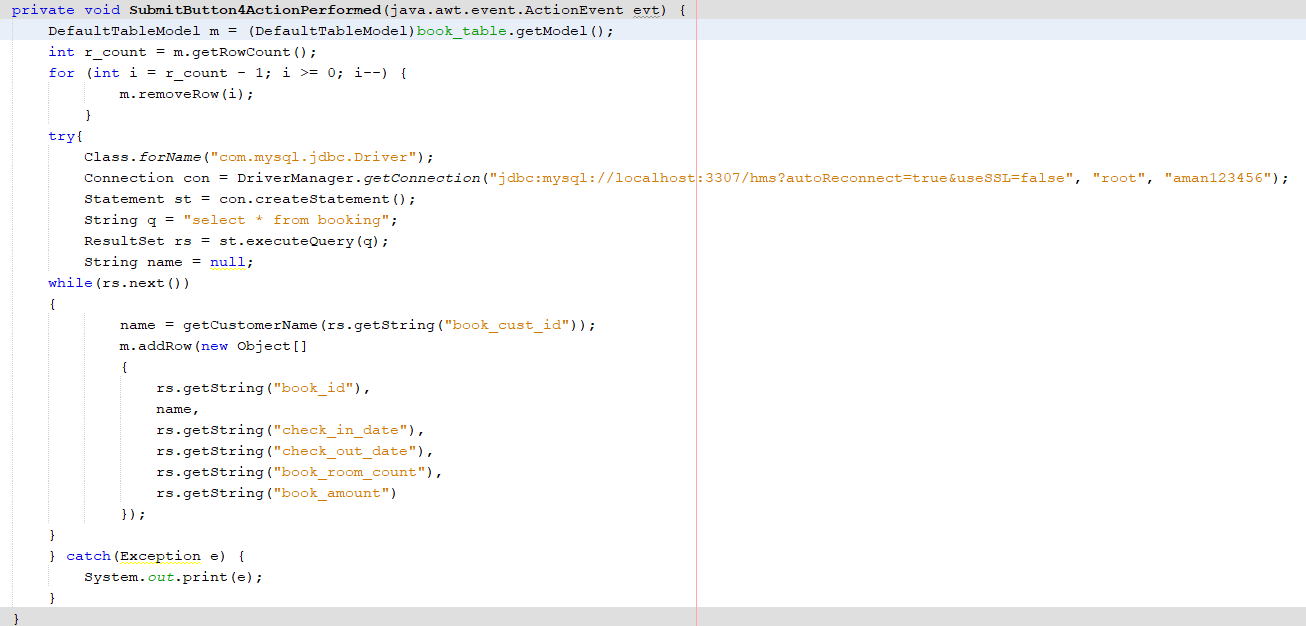
**Linked with success criteria 7, 14:**



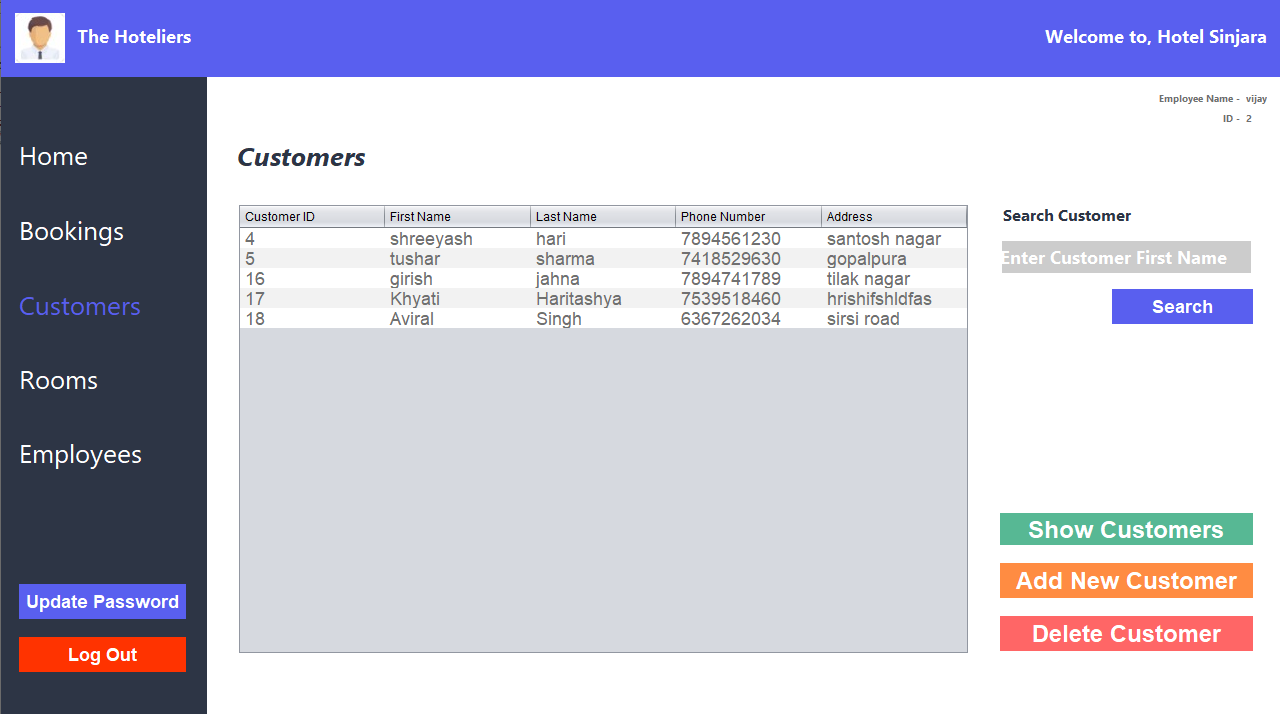
JTable that displays the different entities

Search among bookings

Here is the code that facilitates View Bookings:

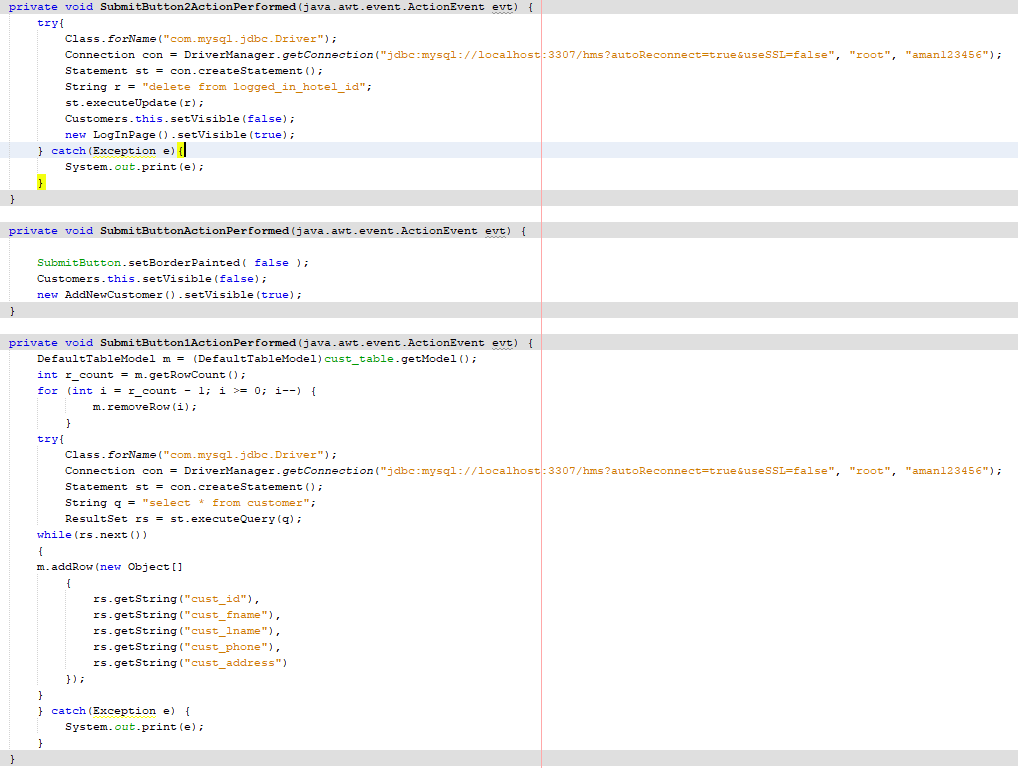
****

**Linked with success criteria 8, 13:**

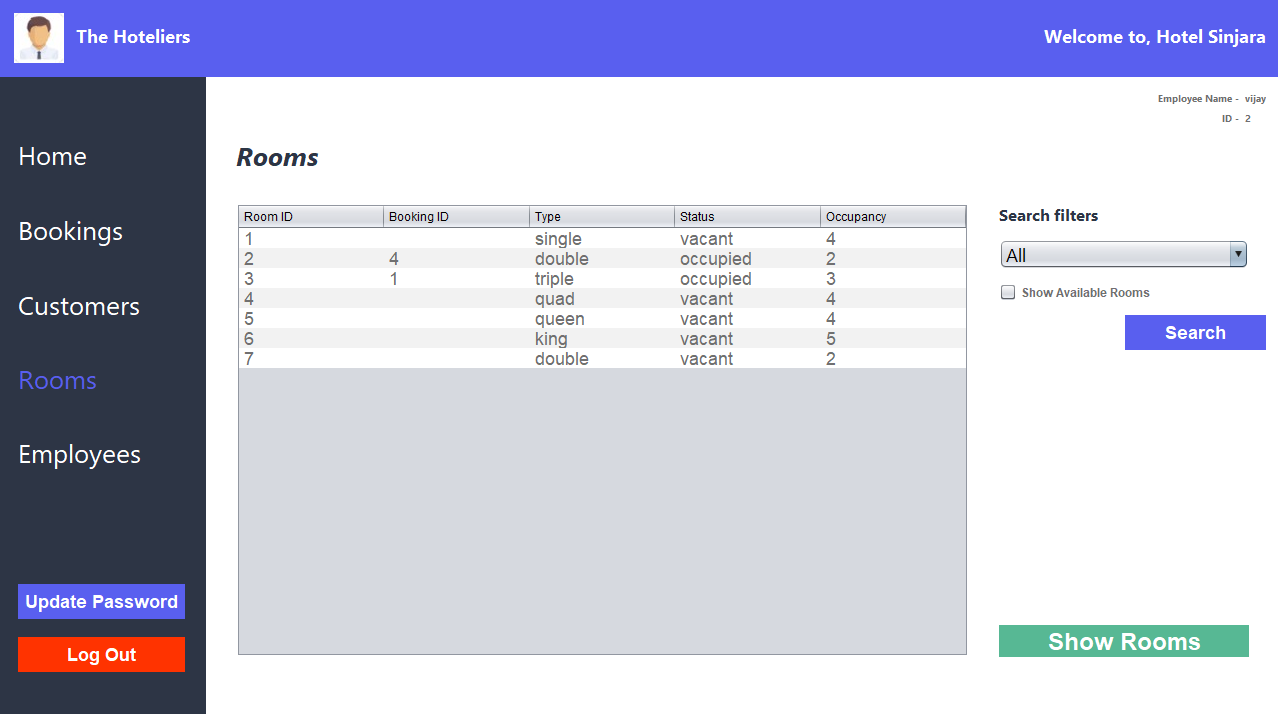


JTable that displays the different entities

Search among customers

****Here is the code that facilitates View Customers:

**Linked with success criteria 9, 16:**

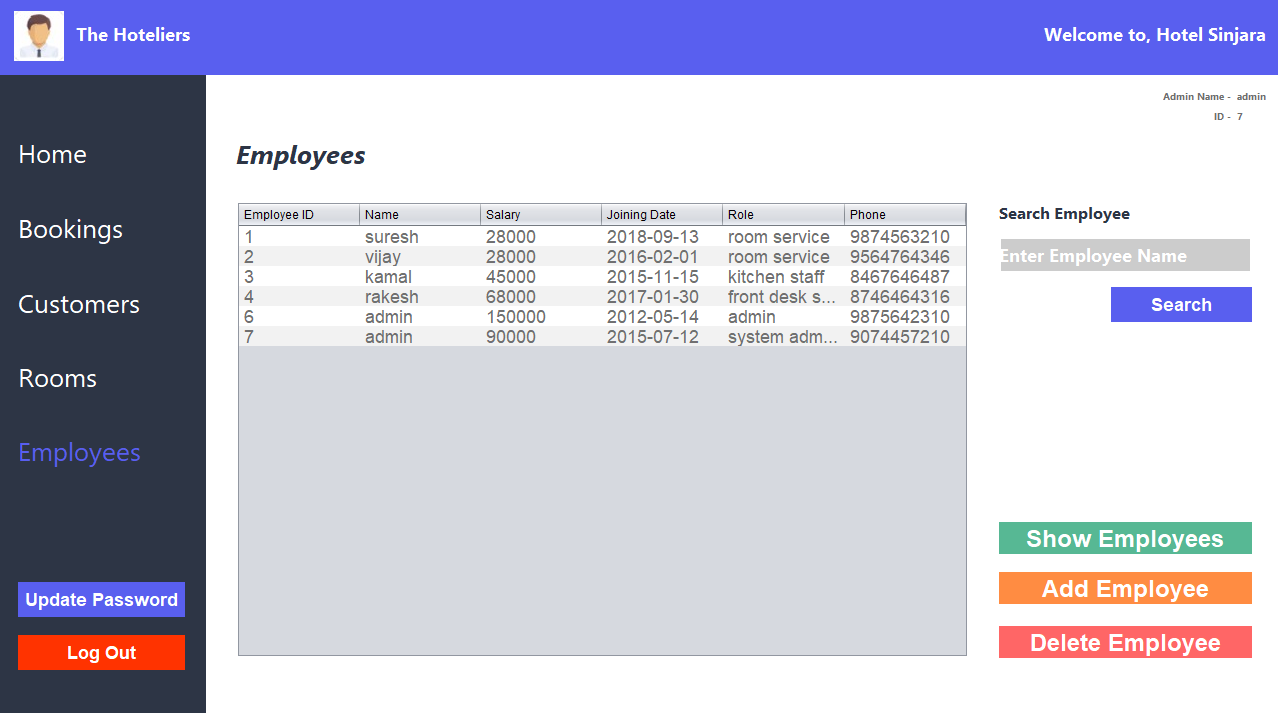


JTable that displays the different entities

To apply filter to see occupancy

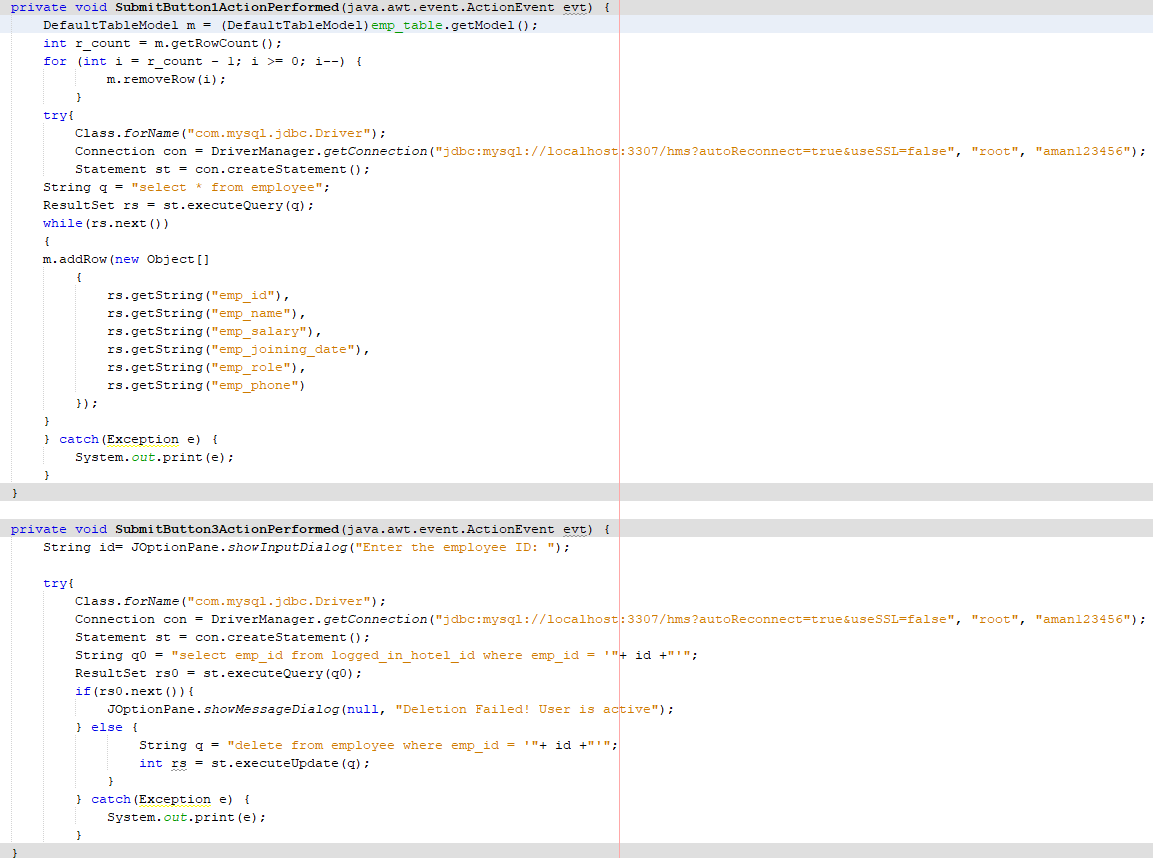
****Here is the code that facilitates View Rooms:

**Linked with success criteria 10, 15:**

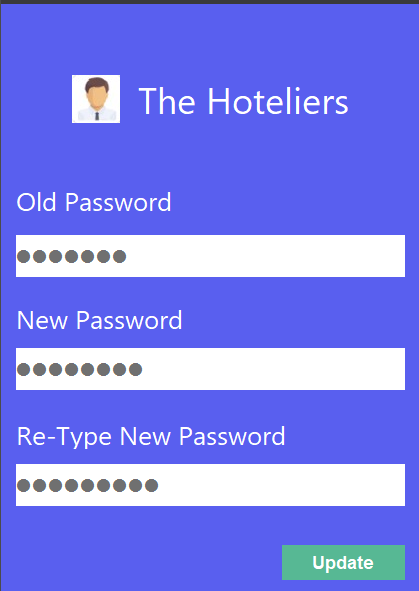


JTable that displays the different entities

To search employee by his name.

Here is the code that facilitates View Employees:

**Linked with success criteria 12:**



Button to update account

Password field to enter user’s Password

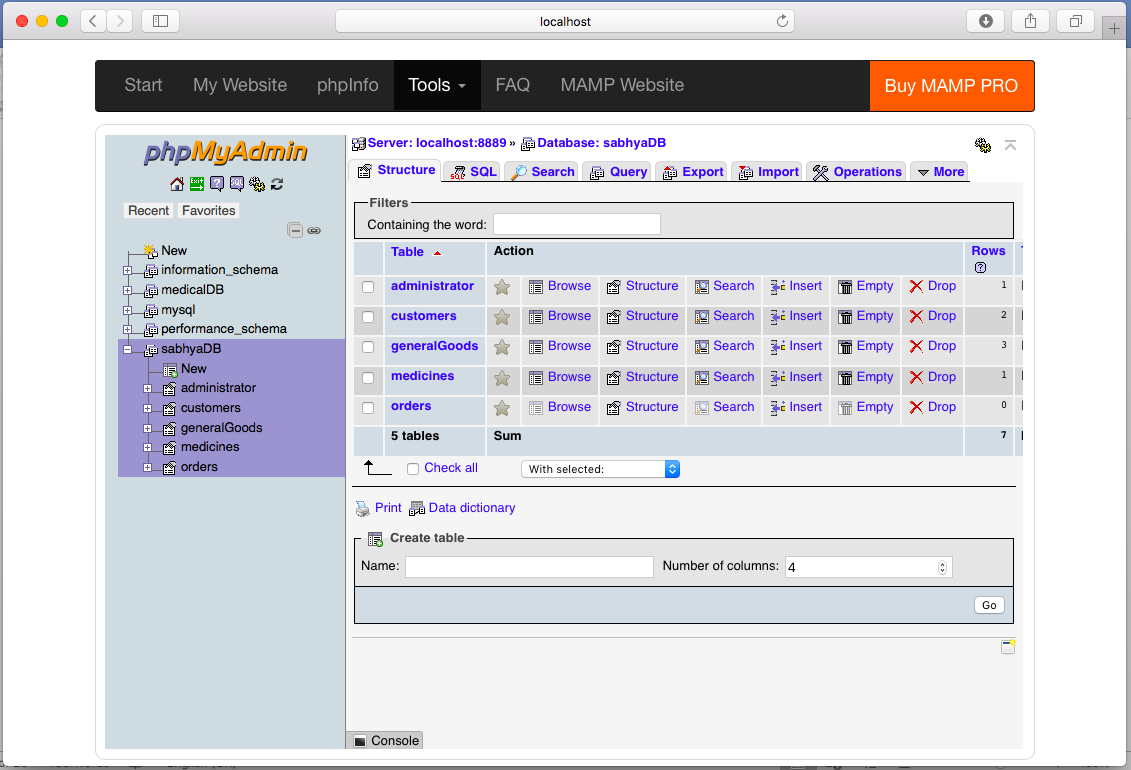
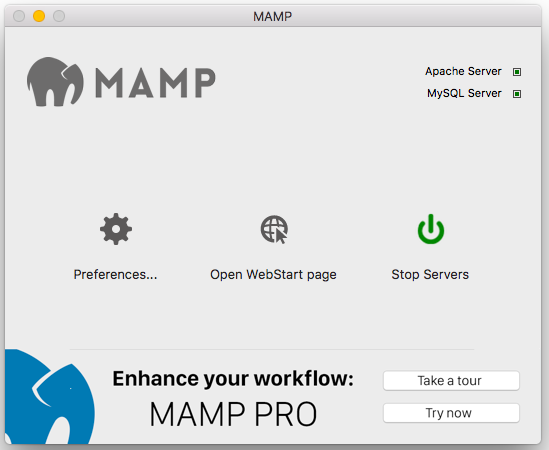
Here is the code that facilitates Update Password:

****

**Techniques Used**

|  |
| --- |
| Online Database Connection |
| Searching |
| Abstraction |
| Non-parameterized Constructors |
| Inheritance |
| Polymorphism |
| Global Variables |

**Online Database Connection – (Success Criterion 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)**



I decided to make use of MySQL database as my backend database for this program. In order to connect to a MySQL database, I had to connect to a local phpMyAdmin server. In order to do so, I made use of MAMP (Macintosh Apache MySQL PHP).

The green button represents the button which enables the servers and switches them on.

This is what the phpMyAdmin database system looks on the local web server on my computer’s web browser. It allows me to create tables, manually make changes to any data records within the tables etc.

This is the code I used to connect NetBeans to my phpMyAdmin database. I made use of port number 3307 for this purpose.

The technique of **Database Normalization** has also been used, wherein a database schema has been implemented in the form of a standard schema. Database normalization has been implemented as a continual process.

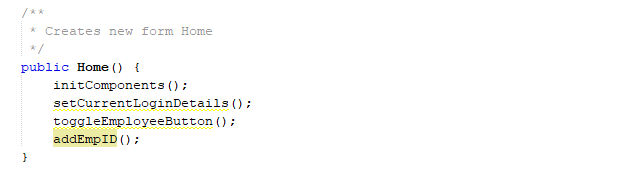
**Abstraction - (Success Criterion 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)**

Abstraction is a process in Java in which certain details of the program are hidden. Only the essential features of the program are visible. The rest of the code is tucked away. I have made use of abstraction on multiple instance in my application. The way in which I have implemented abstraction is that for all the necessary functionality such as displaying items in a JTable, automatically generating IDs, searching, sorting etc. have been given their own functions, which can be easily reused anywhere in the program. Here is an example of abstraction in my application:

The method empLoggedIn() has been coded separately and re hidden from the location in which they are being called.

**Efficient Use of Non Parameterized Constructors - (Success Criterion 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)**

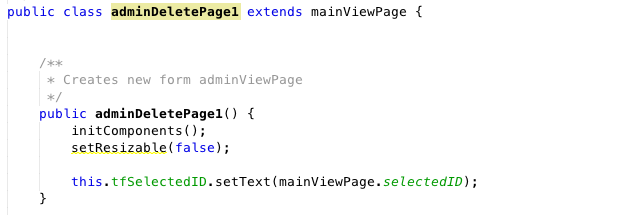
All forms use non-parameterized constructors. I have added certain lines of code to some of these constructors. On all forms I added a line that would prevent the size of the window from being changed. Furthermore, on the forms where I had to generate automatic employee session ID numbers, I added pieces of code in the constructors.



Constructor that generates automatic employee session ID.

**Inheritance – success criteria 9 and 11**

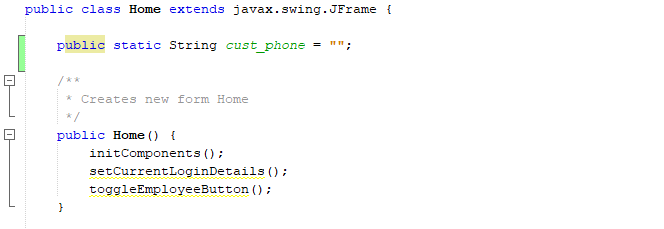
Inheritance proved to be a nifty tool that enabled me to implement my Graphical User Interface.It has been used to enable the admin deletion form – **success criteria 11**. An example is given below:

****

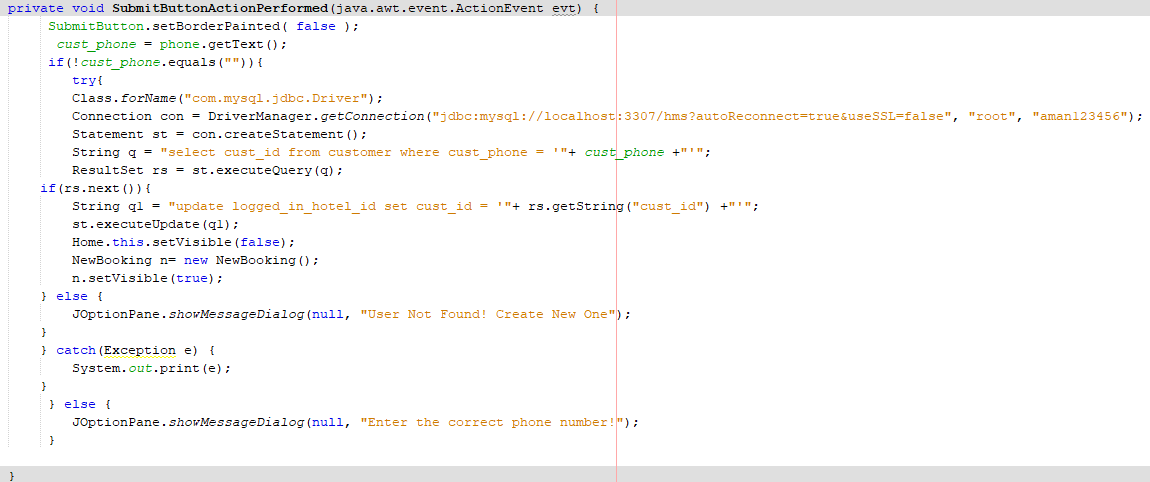
Inheritance has been used to inherit all the properties from main view page.

**Global Variables:**

I made use of global variables in order to keep track of the user who is logged into the system. This allowed to me to display the employee details automatically.



The global variable cust\_phone has been defined and belongs to Home class



The global variable cust\_phone has been called in another class.