Taxi Cab Management Software

Abhishek and Vishwas

Synopsis

TABLE OF CONTENTS

1. ABSTRACT

The project aims at developing software for a cab/taxi management and booking system for a client. The main purpose fulfilled by the Application will be:

1. Providing a user interface to book a cab from one place to another and show him the real-time location of the cab.
2. Providing the Administrator/owner an interface to manage the vehicles and monitor them whenever need in real-time.

2. REQUIREMENTS

The outlay of requirements in categorized into two primary categories:

2.1. ADMINISTRATIVE END

The Administrator need to perform following actions:

2.1.1. VEHICLE RECORD

In this module, the administrator needs to create an entry for each vehicle wherein he/she can fill in all the necessary details such as:

1. Type of Vehicle : AC, Non-AC, 3-wheeler
2. Driver's Name
3. Registration Number
4. Vehicle Make/Model
5. Seating Capacity
6. Last Serviced on : Maintainance details
7. K.M. Reading Till Now : Updated on day-to-day basis.

The following interface should also provide options for:

1. New Vehicle Record
2. Delete Vehicle Record

2.1.2. CITY MAP CREATION

The software should provide an interface to create a 2-D map of city with following functionality:

1. Add/Remove Landmarks (click somewhere and mark a landmark and Fill its properties like: Name)
2. Add/Remove Junctions (click somewhere and mark a junction)
3. ADD/Remove Roads (join any two landmarks or junction to create a road and name it (*optional*))
4. Save the Map into a file

2.1.3. REAL-TIME VEHICLE MONITORING

This interface should provide the administrator with real-time data; location of vehicle on map with color aspect (free, booked, busy/unavailable).

The administrator should be able to click on any cab and get all its details mentioned in "Vehicle Record" section of this document.

2.2. USER END

2.2.1. USER LOGIN

User should get an option of either 'Sign In' or 'Sign Up'. While 'Sign In' will be done by using the unique "Login ID" and "Password", the 'Sign Up' Interface/form should provide the following interface:

1. Name of Costumer
2. E-Mail ID
3. Password
4. Re-Type Password
5. Address
6. Contact No
7. **Submit**

After initial 'Sine Up' the page should redirect to a window to confirm phone number and email-id through an Authentication code send to the both the things.

After confirmation, the page should go to the 'Cab Booking' window.

2.2.2. USER CAB BOOKING

This inter-face will provide the user with functionality of booking a cab as per his needs

2.2.2.1. JOURNEY DETAILS

There will be the following things on this page:

1) User Welcome message and User Details
   a) This should include the user's name and his other relevant details
2) Live Map of the city showing all the cars at with their color status.
   a) Red – Booked
   b) Green – Vacant
   c) Grey – Unavailable (no need to show on user map)
3) Booking Menu with following details:
   a) Origin (from) : User chooses a landmark from a drop down menu showing list of all the landmarks.
   b) Destination (to) : User chooses a landmark from a drop down menu showing list of all the landmarks.

c) Type of Vehicle : Auto/AC Car/Non AC Car
d) Shared / unshared : Make rules for selection keeping the gender in mind.
e) Number of Travellers

2.2.2.2. BOOKING OPTIONS MENU

This menu is generated based on the input in 'Journey Details' menu. Generated output is seen as a list of available vehicles with following fields:

Here the person will choose a vehicle as per his needs. The shared / unshared here denotes that the vehicle is running in shared mode or unshared mode i.e. if there is any person travelling in it or not.

Note: The number of 'Seating Capacity' shows the number of available seats in the vehicle. May be less in case of a shared running mode taxi.

2.2.2.3. FINAL BOOKING MENU

In this menu the person will get the form to fill the details of passengers.

	Name	Gender
Passenger 1		
Passenger 2		
Passenger 3		

Add a **SUBMIT** button

2.2.2.4. TICKET

After this, a final Ticket will be generated with following details:

- Cab Model
- Registration Number
- Driver's Name
- Driver's Mobile Number
- Time to Reach
- Booked Travellers' Name

3. IMPLEMENTATION AND DESIGN

There are four classes:

- **Check**
- **CityMap**
- **Star**
- **Board**

3.1. CLASS – CHECK

It is the main user interface used by the Administrator for updating the databases, adding customers and offline booking. It is having interaction with databases like MySQL and File Handling.

Figure 1 GUI for adding a new Cab to the database

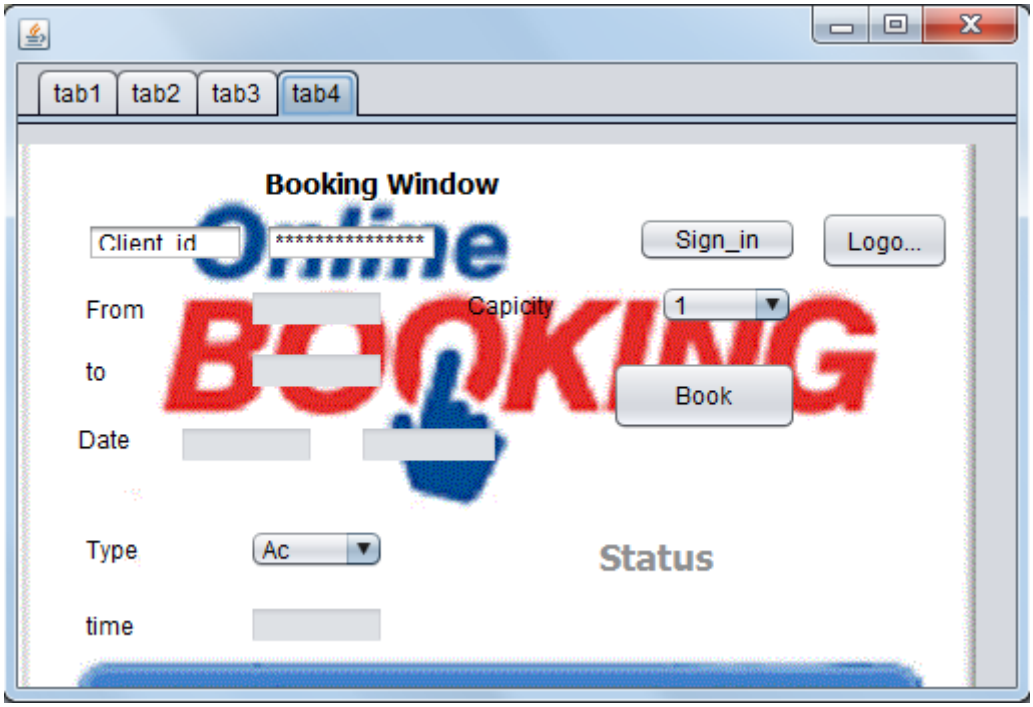

Figure 2 GUI : Offline booking portal

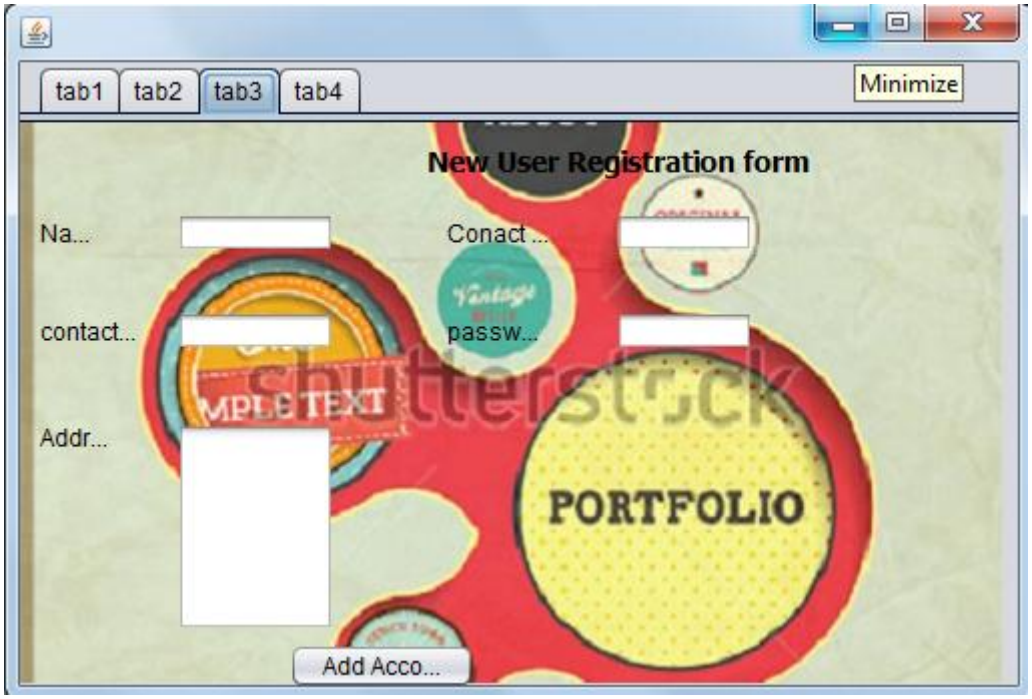

Figure 3 New User registration interface

Figure 4 Interface to delete an entry of a vehicle from the database

3.2. CLASS – CITYMAP

3.2.1. GETSQUARE();

This method checks whether or not a node exists over the place where the map creator wants to add a new node.

3.2.2. ADD()

This method is called when map creator clicks on the map to create a new node.

3.2.3. REMOVE();

This method is used for removing a node from a map. The method is invoked when a user double clicks a node while map creation. The node gets deleted.

3.2.4. PRINT();

On click of 'Save' button, this function is called and it uses the current drawing on the map to form an adjacency matrix locating the cab and naming the nodes.

3.3. CLASS – STAR

This class contains the formation of frame for class 'CityMap'.

3.4. CLASS – BOARD

This class constitutes methods for finding shortest paths from 'A' to 'B' and animation.

3.4.1. GETTINGPOINT()

It reads a file created by the 'Save' Button in the previous user interface and save the coordinates of nodes in an array.

3.4.2. GETTINGCABS()

It reads a file created by the 'Save' Button in the previous user interface and save the coordinates of 'Cars' in an array.

3.4.3. GETTINGNODE()

It is having the current information of the Graph at any point of time and it returns the graph

3.4.4. GETTINGNAME()

It returns the Name of Nodes on the previously mapped graph.

3.4.5. SHORTEST_PATH()

It will take the following arguments – an adjacency matrix, and coordinate of "From and To" and locate the shortest path between the given two points.

3.4.6. GETIT()

It will return the coordinate of each point in the path.

3.4.7. GETTINGPOINT ()

It takes two points as argument and returns hundred equally spaced points on the line connecting these two points.

4. DATABASE EMBEDDED

4.1. MYSQL

Database name : Taxi Service

4.1.1. TABLE : CLIENT

Entities :

Client_id	Name	Address	Contact Number	Password	Login ID

4.1.2. TABLE : BOOKING

Entities :

IDBooking	Registration Number(of cab)	Start Date	End Date	Source	Destination	Client	Status	Start Time	End Time

4.1.3. TABLE : TAXI

Entities :

Registration Number	Type	KM Reading	Model	Capacity	Service Date	Driver Name	Name

4.2. FILE

The file is having the information about the map created in the previous user interface. It is used to fetch information about the current map stored in the classes ‘Check’ and ‘Board’.

File Name : ‘out.txt’