Cab4U

Team 14
Akhilesh Bharadwaj – 180010009
Yashwant – 180010010
Paritosh Gavali – 180010024
Rupesh Kalantre - 180010029

Creating website application for cab booking which handles the both driver and passenger.

Project Problem

Passenger should be notified whether the any driver has accepted his/her request or else *no cab available* should be displayed. After the ride is completed he/she should have the option to verify the same.

Driver should get all the requests from passenger near him along with information like Starting position, Destination , Passenger name, and fare. He/She should have the option to choose or not choose the passenger near him/her. As soon as driver has reached the end position he can end the ride.

Both passenger and driver should have some mechanism of payment. And as soon the ride gets completed fare should be deducted form the passenger and should be added to the driver.

Project Implementation

To solve the problem we created a website which serves as a platform for interaction of passengers and drivers.

Languages used:

Front end : HTML, JavaScript, CSS
Back end : PHP, MySQL

Brief Explanation:

If a passenger books his/her ride from the locations provided, fare of given ride is calculated. If sufficient balance in passenger's wallet is not present then request for cab is denied. In such a case, passenger has to add money in his wallet by payment portal. After the fare is checked then open job query is generated in SQL table with list of all the drivers who are eligible (drivers who are in 1 km range from the starting point). This notification is given to the driver for which he can accept it or leave it open. It driver accepts the job, the job is marked as assigned so that it will be removed from other drivers' list. Now, passenger will be provided with information about the driver who accepted the ride. After we reach at the destination both driver and passenger have to confirm this information. After conformation from both sides we deduct money from passenger's wallet and add it into driver's wallet. Now the job query which was generated is marked as closed and is add to the history.

Passenger

- Passenger should make an account on our website providing details of his/her name, unique username, a password.
- Passenger dashboard has the following features :
  - Profile - user can see his profile.
  - History - to view on going and past cab rides.
  - Wallet - to add money to his wallet.
  - Logout
  - Booking snippet
  - Visualisation of the journey

Driver

- Driver should make an account on our website if he doesn't have any.
- Driver dashboard has the following features :
  - Profile - user can view his profile.
  - History - to view his past service
  - Logout
  - Notification page

Structure of the Application

Databases Used:

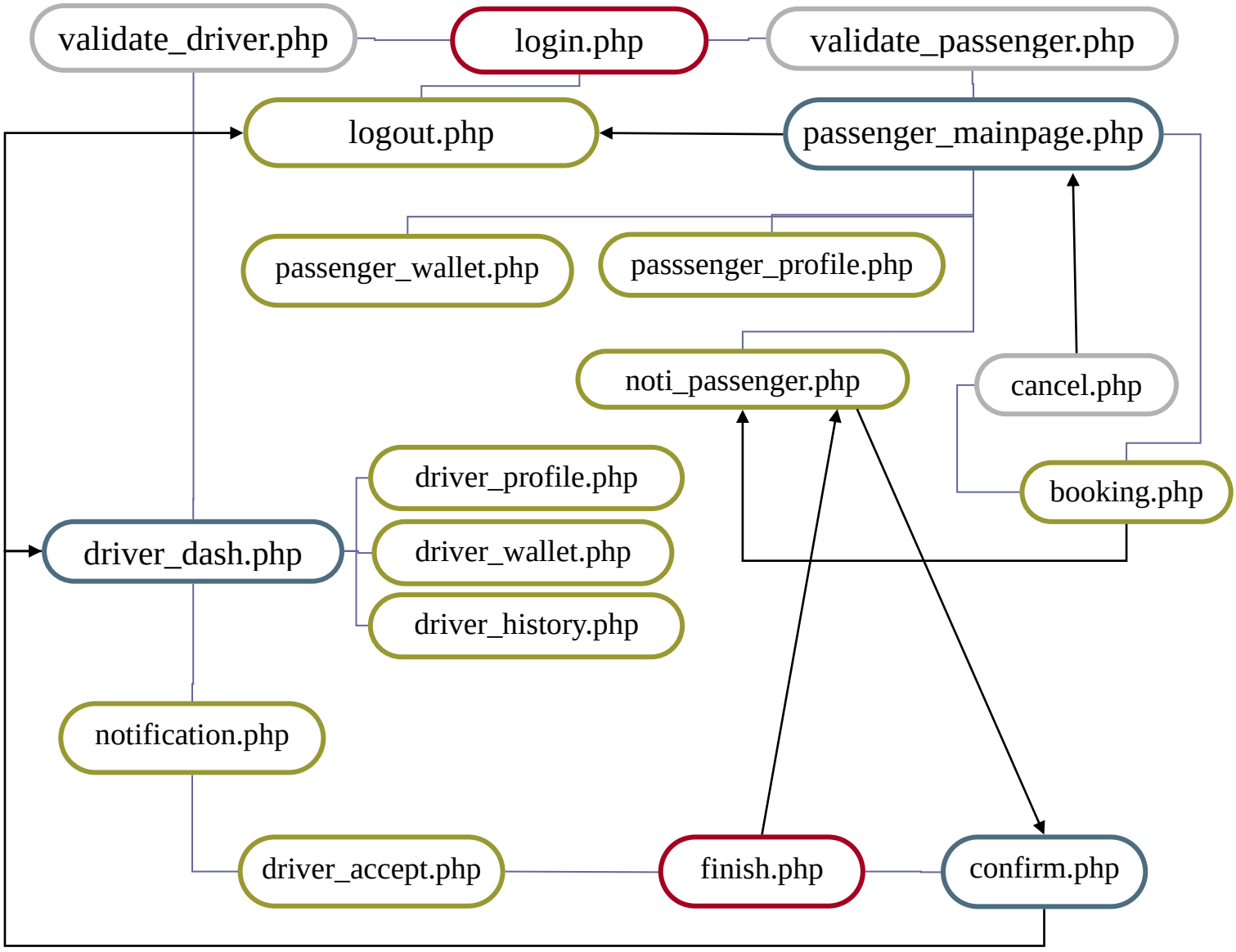
- Passenger table: used to store passengers details : username, name, password, wallet, and credit card details.
- Driver table : used to store driver details : name, username, password, location, wallet
- Location table : used to store name of a location along with its coordinates
- Notification table
    - used to store ride requests and ongoing rides
    - passenger’s name, driver’s name, start point, destination, fare
- History table
    - used to store details of the finished rides
    - passenger’s name, driver’s name, start point, destination, fare

Files and their activities

1. login.php : login page for both driver and passenger. Takes user name and password for login.
2. passenger_signup.php : takes credentials from passenger for registering on site for the first time.
3. driver_signup.php : takes credentials from driver for registering on site for the first time.
4. validate_passenger.php : checks whether passenger is registered and has given correct password then it directs passenger to his dashboard.
5. validate_driver.php : checks whether driver is registered and has given correct password then it directs driver to his dashboard. This page also takes location (latitude and longitude) of the driver and makes him online.
6. pari/passenger_mainpage.php : this is the dashboard of the passenger. It has buttons for passenger profile, history, wallet and logout. It has a booking form and a picture which gives visualisation of journey. Images are stored in maps folder.
7. show_map.php : it changes the map in passenger dashboard.
8. pari/passenger_mainpage.css : this is the CSS file for pari/passenger_mainpage.php
9. pari/paseenger_profile.php : displays the profile of passenger and has button for adding money in wallet.
10. pari/noti_passenger.php : displays ongoing and completed rides of the passenger. Also if the current ongoing ride of the passenger is completed it display the message to confirm money transaction and redirects to passenger_dashboard.php.

11. pari/passenger_wallet.php : displays the current amount in passenger's wallet and has a form to add money in wallet.
12. booking.php : this page is directed if passenger selects starting point and destination of journey. It checks whether any driver is available within 1 km range of the starting position. If no driver is available it shows inconvenience message. It also checks if passenger has enough money for the ride, if not it displays error message. If drivers are available it adds a query in notification table with name of the driver, name of passenger, from, to, and fare. Now it with an option to cancel ride which redirects to pari/cancel.php. Suppose a driver accepts the ride this page is redirected to pari/noti_passenger.php which refreshes every 10 sec.
13. pari/cancel.php : It removes all the booking request of this passenger in the notification table.
14. driver_dash.php : this is the dashboard of the driver. It has buttons for driver profile, history, logout and notification.
15. driver_profile.php : this shows the profile information of the driver.
16. driver_wallet.php : shows the amount in driver's wallet.
17. driver_history.php : shows the past rides of the driver.
18. notification.php : show all the rides which can be accepted by him/her. It has a text input from where he/she can select passenger. If he clicks accept he is redirected to driver_accept.php.
19. driver_accept.php : driver is marked as busy and it removes the all the notification query of the passenger as well as passenger from notification page. It also adds query in history page with marker as ongoing ride.
20. finish.php : The query in history table is marked as finished from driver side and waits until passenger also confirms that the ride is finished. After passenger confirms it redirects to confirm.php.
21. confirm.php : it deducts amount from passenger's wallet and adds to the driver's wallet and driver is redirected to his/her driver_dash.php.

Flowchart of the website



Scope of Improvement

- Rentals and reservations modes can be added.
- Real-time tracking service can be added easily if street maps are incorporated in the application.
- With real-time tracking we can remove fixed loactions for ‘from’ and ‘to’ tables.
- Message and emails can be linked with the user accounts so that notifications can be sent.
- OPT service can be encorporated.