DOCUMENTATION

Problem Description:

Truck going from city A to city B charges for both sides, it comes back empty , leading to wastage of fuel. Their may be another citizen who needs to transport goods from city B to city A , so he will use another truck , paying him for both sides and then this truck will come back empty. So, there is enormous wastage of fuel and money , which can be saved if the citizen going from city B to city A instead of a new truck uses the same truck as that going empty.

Design issues and provided solutions:

- **Remove outdated entry** : To maintain the correct entry of the time left in departure , the time should be updated regularly.

  Initial Idea : To update the entry on the client side , whenever the client open the site , but to update the whole data may take time , if the database , leading to fall of bandwidth for the client , so this should be done .

  Implementation on server side : A php script is written to update the entries and to run the script every minute crontab feature of the ubuntu is used .

- **Remove User from database** : Users can register for the email service and so users must be removed from the database who have confirmed with the driver .

  Initial Idea : To maintain a form on the site so that user can unsubscibe . But a user would not like to come back to site.
  So, a link is send in the mail itself , to unsubscribe.

- **To retrieve user details:** Retrieve user details like name , phone number , email Id etc.

  Possible Solution : Use android API , to directly retreive these data from the phone , but phone number was not accessible in every phone also we need permission to access phone state and things get ugly there .

  Implemented Solution : We used android shared preferences API to save the user preferences. User need not add the details every time .

- **Route**:If the user route is not exactly same as that of truck driver , but is a subroute .

  Possible Solution :Using google API maps v3 contains loaction , to check if query location lies in the route , but that was running on the client side , so we have to send the whole database to the user . Alternative was to implement this on server using node.js , but a single query was taking nearly 2 seconds.

  Implemented Solution : We have not made both the initial and the final city compulsory , so the user can query according to his needs [For example : all trucks starting from ropar] . Then he can see all the results based on the query , we have also added the google map view ,so that he can check whether his city lies in the path or not.

- **Database:** we have maintained two seperate tables for trucks and clients.
  Table 1 : Trucks :
  It consists of 5 columns for storing the name , phone number , initial place , Destination and Time left in departure .

  Table 2 : Clients :
  It also consists of 5 columns for storing the Name , EmailId , Initial Place , Destination and Time left in departure .

Possible solution: Since both the columns have nearly same columns , they could have been merged into a single table , adding a new column , to store the role if it is client or a driver . But than the time of query and sending mail would increase .When the user is querying , we only need entries of the truck so time would increase while scanning for the registered clients and vice-versa.

So we decided to make 2 different tables.

Design Details:

- **Server side design:**

  The server side is implemented using xampp for ubuntu .

  Mysql is used for maintaing the database and php is used as the server side scripting language .

  Different php scripts are written to serve different use based on the purpose of use. Php scripts are written to serve purposes like registering clients , adding trucks , deleting trucks who have received customers etc.These php scripts then connect to the database and then sql queries are run depending on the task.

- **Database Management :**

  Database is maintained using Mysql .
  Database consists of two tables , one for storing the details of the truck and other for storing the details of the clients .

- Table 1 : Trucks :
  It consists of 5 columns for storing the name , phone number , initial place , Destination and Time left in departure .

- Table 2 : Clients :
  It also consists of 5 columns for storing the Name , EmailId , Initial Place , Destination and Time left in departure .
  Since both the columns have nearly same columns , they could have been merged into a single table , adding a new column , to store the role if it is client or a driver .
  But than the time of query and sending mail would increase .
  When the user is querying , we only need entries of the truck so tiem would increase while scanning for the registered clients and vice-versa.
  So we decided to make 2 different tables.

- **Php Scripts :**

  1. config.php :
  This script consists of all the variables used to connect all the php with the database like IP , username , password and database name . Just changing anything in this file will change in every other file.

  2. Truckregister .php :
  This script is written for adding new truck in the database . Whenever a driver send request , if there is already a entry in the database of the same phone number then the entry is updated , else a new entry is added in the database . Also emails are send to the clients who have registered for the service , with the same initial and final city .

  3. Androidquery.php :
  This script is written for the android app , query for the result . Whenever a new client query for the initial city to the final city , the query is done and the output is send in the form of json encoded string .

  4. Clientregister.php :
  This script is written to add the new registered client in the database . Whenever a new client register for the service , a new mail is send to them . They are added to the database so that whenever a new driver is going on the same path , he/she is informed by e-mail.

5. Hourly.php : [CRONTAB]
This php script is written to maintain the time of the entries in the database such that there is no entry in the database which is outdated or the client is shown wrong time. This can be done using timestamp but that will only delete the outdated entry , while the user will shown wrong time . For example , if the truck will leave after 4 hours then the entry in the table will be 4 hour in the starting , and using timestamp the entry will be deleted after 4 hour , but the user will continue to see the truck will leave after 4 hour , even if there is only few minutes remaining for the departure of the truck .
This could also have been done at the client side , whenever a user onpens the slide , but if the database is large that will take time and also the bandwidth of the user , so this is done at the server side itself.

6. Deletetruck.php :
This script is written to delete the trucks from the database who have received the client . It is necessary to delete the client from the database , so that further clients should not try calling him , and also they should not see their entry while query .

7. Deleteclient.php :
This script is written to delete the clients from the databse who have received truck , and is confirmed to go with the truck .
Whenever a mail is send to the registered client , a link is also send to unsubscribe , so that further mail is not send to them .

MyTruck Android Application :-

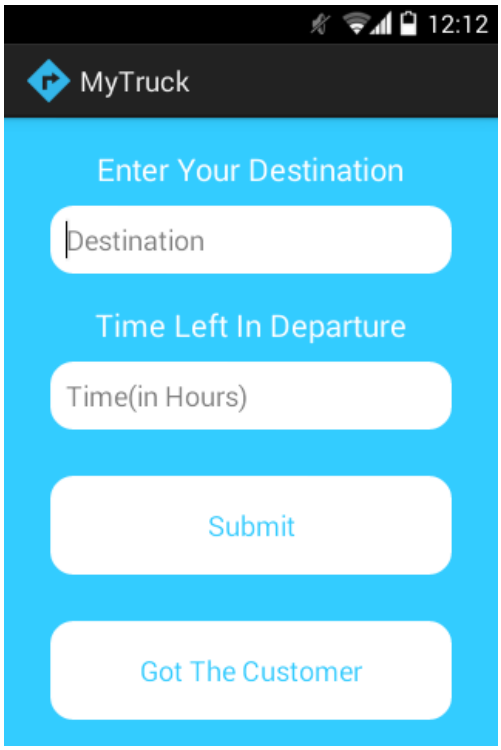
- Description :- This application is for Drivers .When the driver has successfully completed his job, He will use our application. He will enter his final location and time left in departure. Other information will be retrieved and this will be sent to the server via an Async task and added to the table. When he gets the Customer, He is to click on “Got The Customer” button so as to delete his entry from the table.

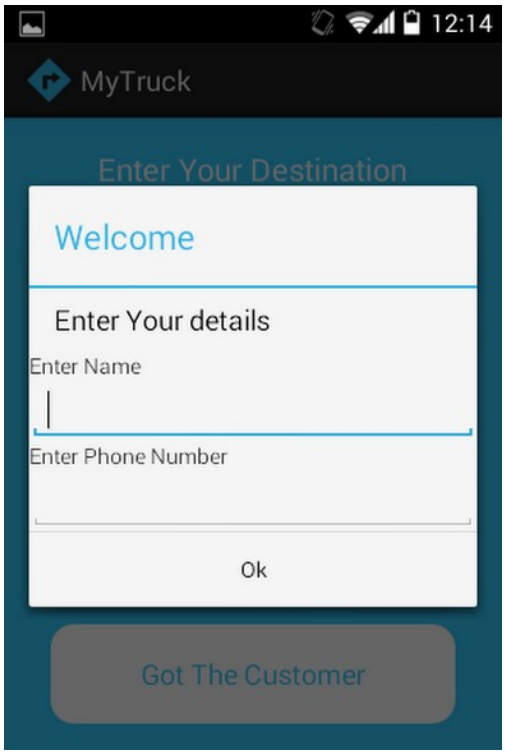
- Permissions :-
    - Permission for Location and Internet
        - android.permission.ACCESS_FINE_LOCATION
        - android.permission.INTERNET

- APIs Used :-

    - Shared Preferences - This is Used To save the driver’s details so that the he need not add his details everytime. The driver enter his details only once and they are saved in his preference.
    - Location - This is used to get Driver’s current location either through GPS or Network Provider whichever is available.
    - Geocoder – This is used to get the name of the location from coordinates(i.e. latitude ,longitude)
    - HttpPost,HttpClient – These are used for intercommunication between the Server and the phone.
    - Async Task :- The Communication with the Server can’t be done on the main UI Thread and hence this is used.

Screenshots





- **MyTruckClient Application :-**

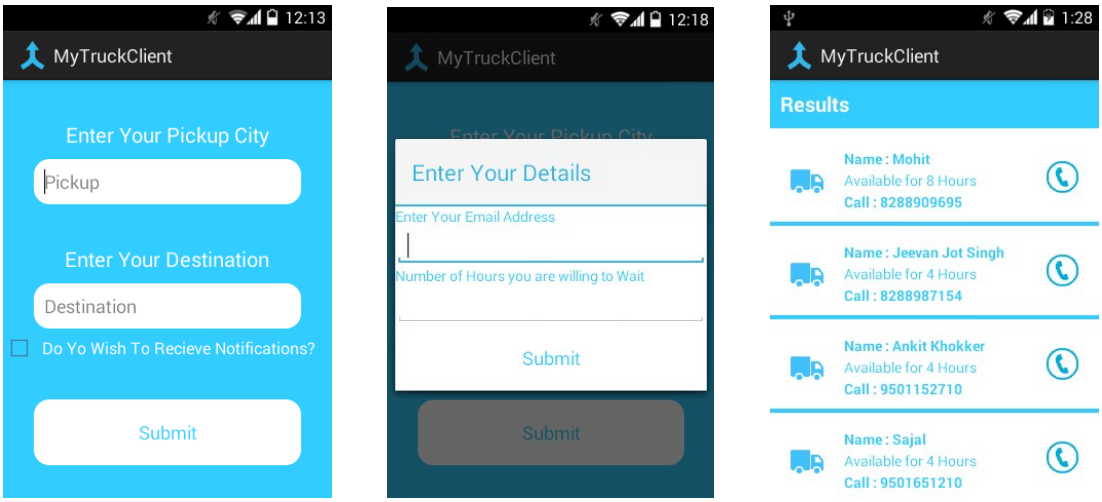
  Description :- This application is for the clients, client can make a query by entering his pick-up location and destination . the results will be displayed in a list alongwith their contact no. and availability details, he can make a call by directly clicking on the respective entry in the list. in case there is no truck currently available for his destined location he can register his email to receive notifications when a truck is available for his destination within his given waiting time.

- Permissions :-
  - Permission for calling and Internet
    - android.permission.CALL_PHONE
    - android.permission.INTERNET

- APIs Used :-

  - HttpPost,HttpClient – These are used for intercommunication between the Server and the phone.
  - Async Task :- The Communication with the Server can't be done on the main UI Thread and hence this is used.
  - JSONArray and JSONObject :- The response from the server will be a JSON and hence these are required to parse the response .
  - ListActivity(ListView) :- This is used to display the parsed results in a form of List.
  - ArrayAdapter :- This is used for initialization of the custom ListView.

- Screenshots:-



The Website :

The website makes use of HTML (Hypertext Mark Up Language) , PHP, mySQL and Javascript. PHP and mySQL are used for connecting to the server and perform queries on the basis of the information fed in by the Client. The Website also make use of GOOGLE MAPS and JQUERY.

1) Index.html : This is the main file and build up the home page of the website. It consists of two forms “Registration” and “Query”. A Snapshot of the site is given below :

2) query.php : This PHP script is written to perform queries with the server database and output the output according. This php script is included in an another file named index3.php

3) index3.php : This PHP + HTML file builds up the page where a table is made according to the query of the user. The page also includes a search box where the client can search for any particular driver. This search box gives info to another file named searcher.php. This page makes use of GOOGLE MAPS to show the route of each query. The map can be viewed by clicking on the MAP

VIEW button. This page also makes use of JQUERY TABLESORT feature. It can be done by clicking on the name of the column. A screenshot is given below :

4) searcher.php : This file recieves some query the client wants to search about. On the basis of that this file searches for that keyword in the table of the previous page. This page also makes use of JQUERY TABLESORT feature.