Data Communication Network Assignment-1

Rahul Dhayal
B19EE069

In this assignment I am going to develop a simple network app which uses appropriate protocols and different network layers.

(a)

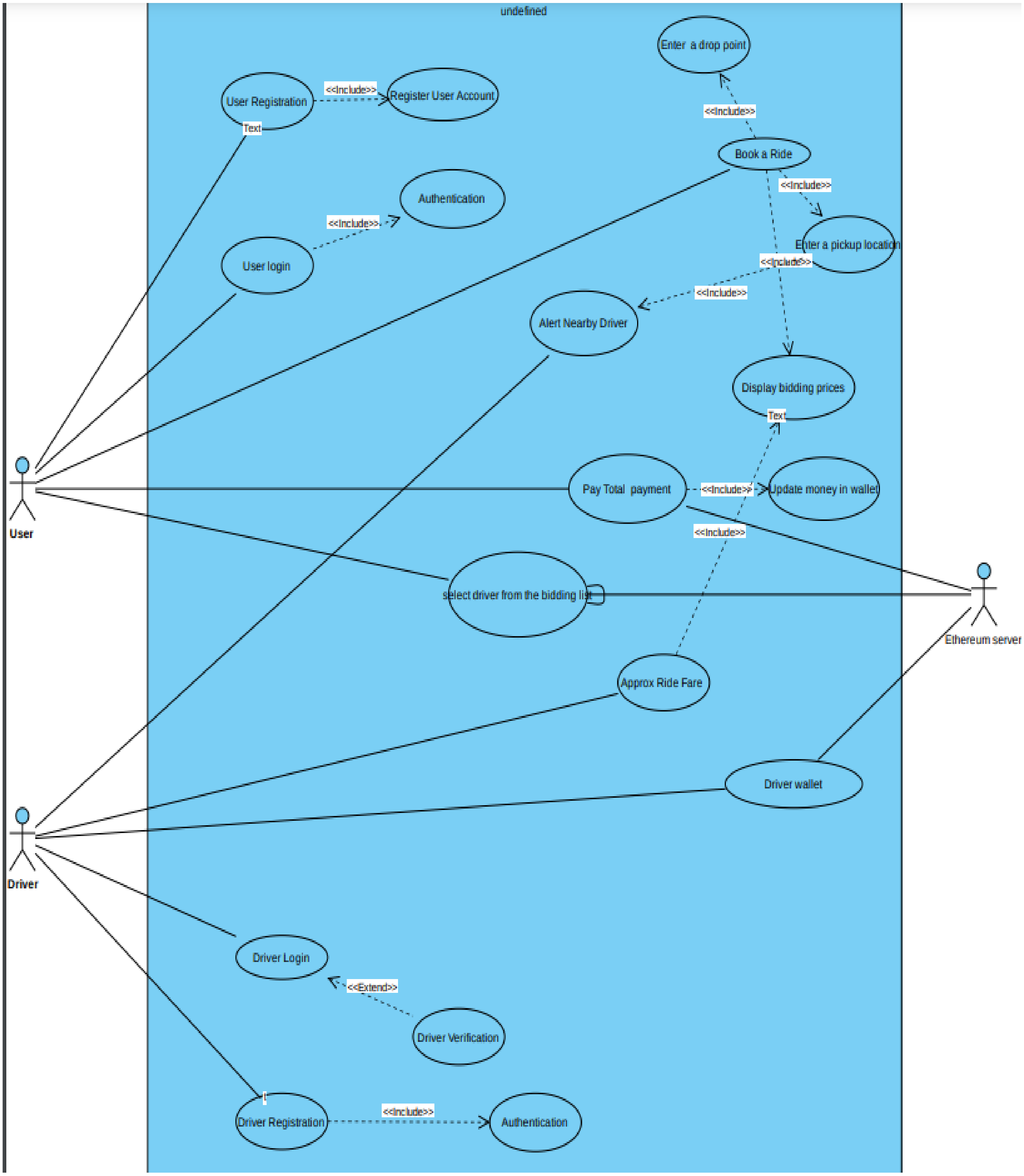
Originality of the idea:-

So here I propose my idea for making transportation easier. As I named this app "*RIDE DEKHO" .* In this app we can book rides to go from one place to another. But this app is slightly different from existing apps like Uber and Ola. In this app:-

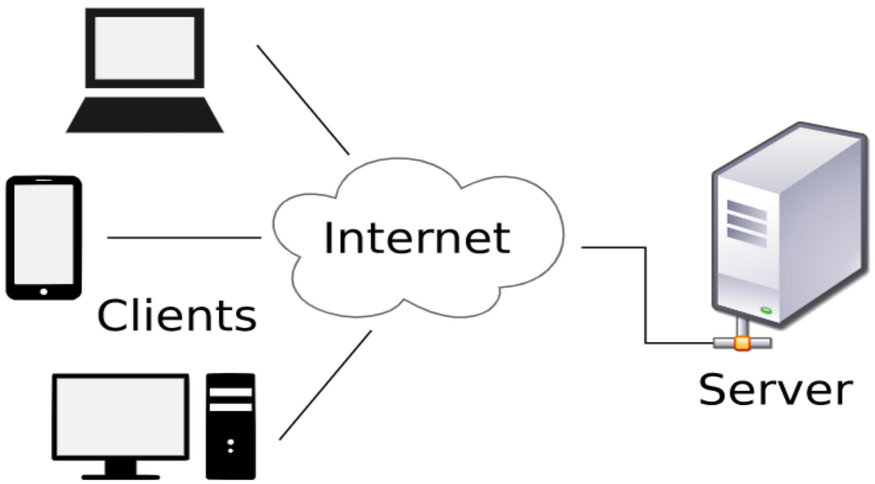
- User first login or register account on this app.
- Then the user books the ride and enters his/her pickup or destination location.
- Then notification about the ride goes to every nearby driver.
- Drivers have to also register to the app.
- Then drivers start bidding about the user's ride.
- And this all bidding shows to the user's display.
- Then the server(or we can say app management team) selects the appropriate ride for the user.
- Then on completing the ride the user pays to the server and then server pays to the driver.

Use-case requirements:-

Here below I attach a full body diagram of use-case how this app "*RIDE DEKHO*" works.


undefined
Enter a drop point
User Registration
<<Include>>
Register User Account
Text
<<Include>>
Book a Ride
Authentication
<<Include>>
<<Include>>
User login
Enter a pickup location
<<Include>>
<<Include>>
Alert Nearby Driver
Display bidding prices
Text
Pay Total payment
<<Include>>
Update money in wallet
User
<<Include>>
select driver from the bidding list
Ethereum server
Approx Ride Fare
Driver wallet
Driver
Driver Login
<<Extend>>
Driver Verification
<<Include>>
Driver Registration
Authentication

<u>System architecture</u>:-
I use client-server **paradigm** system architecture here.



As we can see in our use-case there is no direct connection between two clients. Each and every time the user has to request the server of the app for any help.
Here is the need to always be on server and anyone needs to ride at any time. And also in its server IP address is permanent.
In this app the client sends a request to the server and then the server responds accordingly.

(b)
Compare against any competitors already existing?:-
There are many apps available similar to this like Uber,Ola,Rapido which provide the same riding service.
So in this app I use WebSocket because WebSocket updates the application as it gets any update from the server, but other competitors like Uber use the HTTP/2 over TLS/TCP.
In comparison our app uses TCP and other competitors also use TCP.
And many apps choose peer-to-peer architecture but in this app I use the client-server architecture.

(c)

Choice and usage of appropriate protocols:-

In this section I briefly describe which appropriate protocols we use in this app to work it properly.

- ***For application layer protocol***:-

  For sending any message I use WebSocket because it is the foundation of any data exchange on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser. And as any user sends a constant request, with websockets, updates are sent immediately when they are available. Like if a user is constantly changing his/her location so prices also change with it and number of drivers also change so that's why I use WebSocket instat of other application layer protocol so that any instant update about users' ride can be updated.

  And I also use **DNS(Domain Name System)** as a translator between IP address and name. The main use of DNS is that it makes it easy to understand the message as if in "*RIDE DEKHO*" app a user doesn't need to write the full IP address of the app on the browser he/she can simply write RIDEDEKHO.com.

  But in WebSocket at some places I use HTTP(HyperText Transfer protocol) to solve some problems like:-

  - Users and servers can talk to each other at the same time by using full-duplex communication in http.
  - I also use http as if any user's ride is complete, the server doesn't need to maintain the request of the past client.
  - And in it I use a persistent http connection so that multiple information can be sent over through the single TCP.

- ***Transport layer protocol:-***

  In this app I use TCP(Transmission Control Protocol) as a transport layer protocol.

  TCP is more reliable as there is no chance of any data lost, data damaged and delivering data at different IP addresses and it ensures

that all segments are received in order and any lost segments are retransmitted.
TCP also detects any kind of error.
And also in TCP sender receives confirmation that the receiver receives the information. But in UDP the sender sends the message and never gets any confirmation.

- ***Network protocol:-***
  I use IP(Internet protocol) as a network protocol.
  Users or servers send messages or data in the form of packets.
  As if due to network congestion if any packet is lost I use TCP as a backup.

Web Cache:-
We also set up web cache in the big city and countries so that on each and every request of the user there is no need to go to the main server of the app.

(d)
Performance consideration:-
So to work our app in the proper manner we will see an example of any city.
Let's take the example of Jodhpur and let's assume that the total user of this app is 1000 in Jodhpur. And the required rate for each user to work this app properly is 1Mbps.
So every user needs a connection whose data speed is greater or equal to 1Mbps to send their request to the router which is set-up in Jodhpur. And the link rate of link between router and server/web cache should be greater than 1000 so if all users use the app together there will not be any kind of delay or any kind of congestion/queue at the router.
And if we want to decrease the link rate, I use in this app the packet switching method. So that we can break the data in small packets.
So the performance of this app depends on the number of users.

(e)

Sample code:-

Sample code:-https://github.com/RahulDhayal/Ride-Sharing