Software System for Hailing Cab

UCS2201 – Fundamentals and Practice of Software Development

A PROJECT REPORT

Submitted By

ANANDHARAJ D 3122225001009

DINESH S 3122225001029

Department of Computer Science and Engineering

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

Kalavakkam – 603110

July 2023

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled "**Software System for Hailing Cab**" is the bonafide work of "D.ANANDHARAJ (3122225001009) and S.DINESH (3122225001029)" who carried out the project work in the UCS2201 – Fundamentals and Practice of Software Development during the academic year 2022-23.

Internal Examiner External Examiner

Date:

TABLE OF CONTENTS

ABSTRACT

The project “Software System for Hailing Cab” aims at developing an application for assigning cabs for customers based on their requests and locations. We have created this software for easy booking of rides and assignment of drivers. The software is made in such a way that even a naive user can easily handle it. The software displays the map of places wherever the service is available. The software can handle both advance and instant bookings. The software clearly maintains the past bookings of the customers, which can be viewed by the user.

We considered the privacy and security of the user. User’s details like past activity and advance bookings can only be seen by the user by entering their username and password.

Passengers can rate their ride experiences, enabling continuous improvement in service quality. This system leverages modern technology to provide a seamless, convenient, and reliable transportation service for users. The software also allows the customer to pay bill through online and by cash. The software sorts the driver and assign the nearest driver with high ratings, which reduces the waiting time of the customer and assigns the high rated driver.

Introduction

Transportation has evolved so much right from the ancient age. Transportation is fundamental for development of mankind. Cab booking systems provide users with the convenience of booking a ride anytime, anywhere, using their smartphones or other devices. This accessibility eliminates the need to wait on the streets for a taxi or rely on traditional dispatch services.

The system uses intelligent algorithms to match the nearest available driver to a passenger's location, reducing waiting times and optimizing the utilization of drivers. Seamless integration with GPS and navigation tools enables efficient routing and reduces the chance of getting lost. Cab booking systems have the potential to reduce traffic congestion and carbon emissions by promoting carpooling and shared rides.

The success of cab booking systems has opened up opportunities for entrepreneurs and companies to enter the ride-hailing market, contributing to economic growth and job creation. As cab booking systems grow, there may be opportunities to expand their services to rural areas, providing transportation solutions in underserved regions.

PROBLEM STATEMENT

Develop a software system for assigning cabs for customers based on their requests and locations. The customers are charged a fixed base fare plus fare based on the distance traveled. During peak demand time, a surge fee also will be charged. Apart from this, if the customer books the vehicle in advance, advance booking fees will be charged. If the customer cancels the ride for some reason, a cancellation fee will be charged.

Constraints

- Assign a driver so that
  - the customer can be picked up with minimum waiting time
  - the best among the available drivers based on the average rating is assigned

- The assigned driver should be driving the minimum distance to pick up the customer.

Input

- Set of customer requests, where customer request comprises location, destination and
  mode of travel such as auto, mini sedan, sedan, SUV, Innova and so on.

- Number of vehicles available in each category

Output

Allocation of suitable cabs to customers and bill generation

Exploration of the problem

- We started to work on the project with the motive to produce high quality software for the good cause.

- Beyond the problem statement, we tried to explore about lot of things which will benefit both customers and drivers.

- We've worked a lot on keeping the user's data safe and accessible only to user whenever he/she needs it.

- Additionally, we regularly get feedbacks and ratings from customers to improve our services.

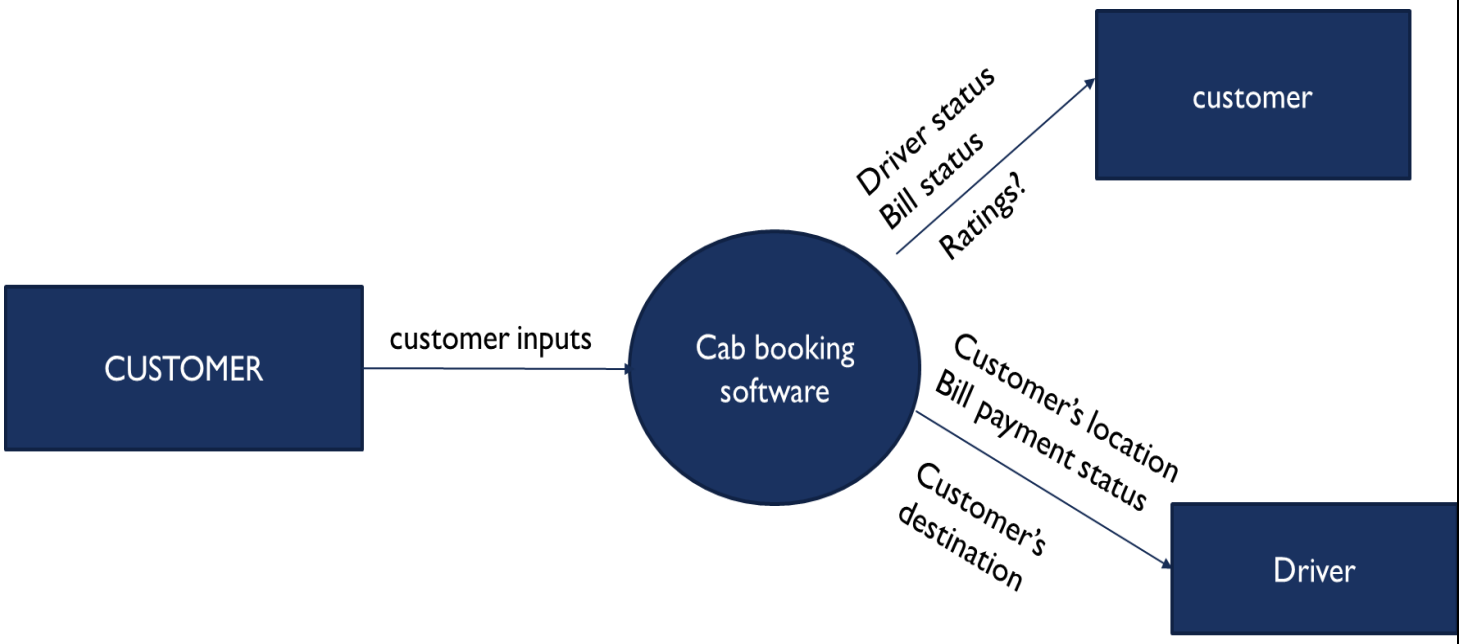
- Allocating the right driver to the right ride is essential for efficient operations. The system must take into account factors such as the driver's proximity to the pickup location, their current availability.

- Calculating fares fairly and transparently is crucial for building trust with customers. The system should be able to estimate costs accurately, considering factors like distance, time, and additional charges

- There will be cases where passengers need to cancel their bookings. We must handle it properly and act accordingly. Ex.Cancelling the ride after driver assignment leads to extra charges.

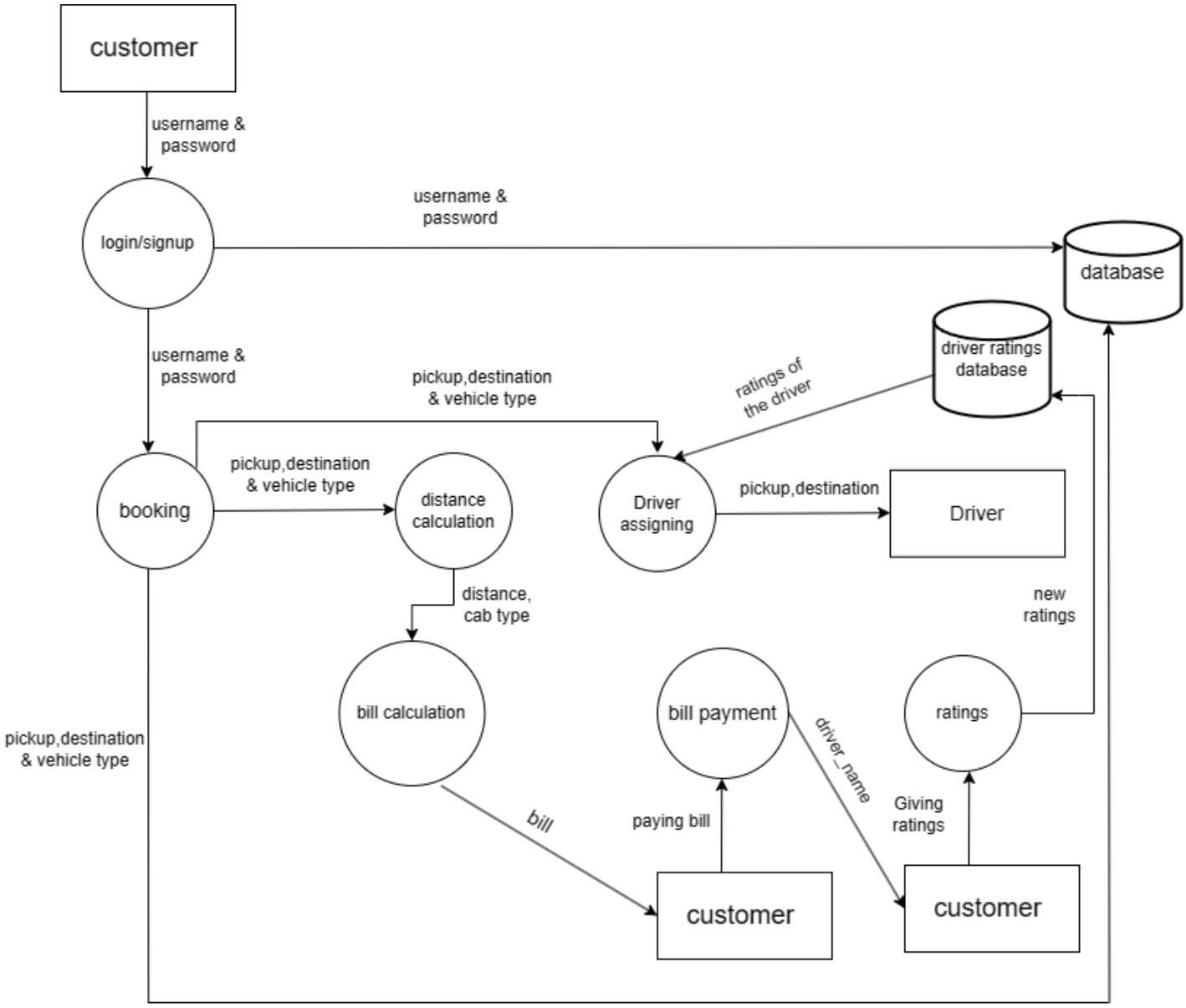
PROBLEM ANALYSIS

DFD LEVEL 0



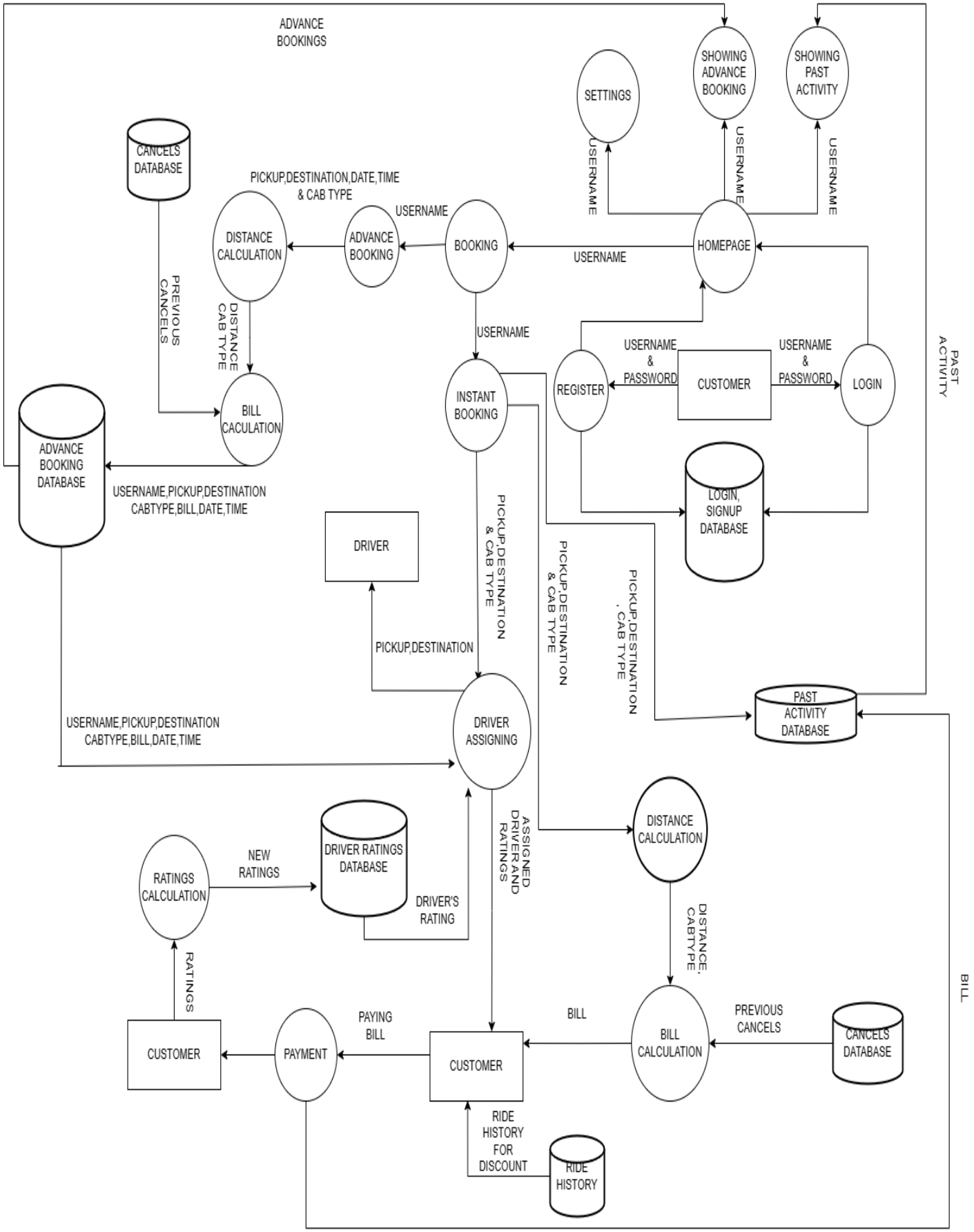
- CUSTOMER PROVIDES THE INPUTS SUCH AS PICK UP , DESTINATION,VEHICLE CATEGORY AND PICK UP TIME.
- THROUGH CAB BOOKING SOFTWARE ,CUSTOMER'S INPUTS ARE SENT TO DRIVER AND DRIVER IS BEING ALLOTED.
- AFTER DRIVER ASSIGNING,CUSTOMER IS REQUESTED TO PAY FOR THE RIDE AND GIVE RATINGS.

DFD LEVEL 1



- User login/sign up using username,password and this data flows to database and get stored there.
- And this data flows to booking module, where customer books the ride by giving pickup,destination and vehicle type as inputs.It gets saved in the database after successful ride.
- This data flows to distance calculation and driver assigning module where distance between pickup and destination is calculated and driver is assigned respectively.
- Driver is assigned to customer based on the constraints like nearest driver and high ratings.
- Distance from distance calculation flows to bill calculation and bill is generated.
- Bill is shown to the customer through payment module and customer the bill. From bill payment driver name flows to ratings module. Customer get to rate the driver here.
- The new rating gets stored in the database.

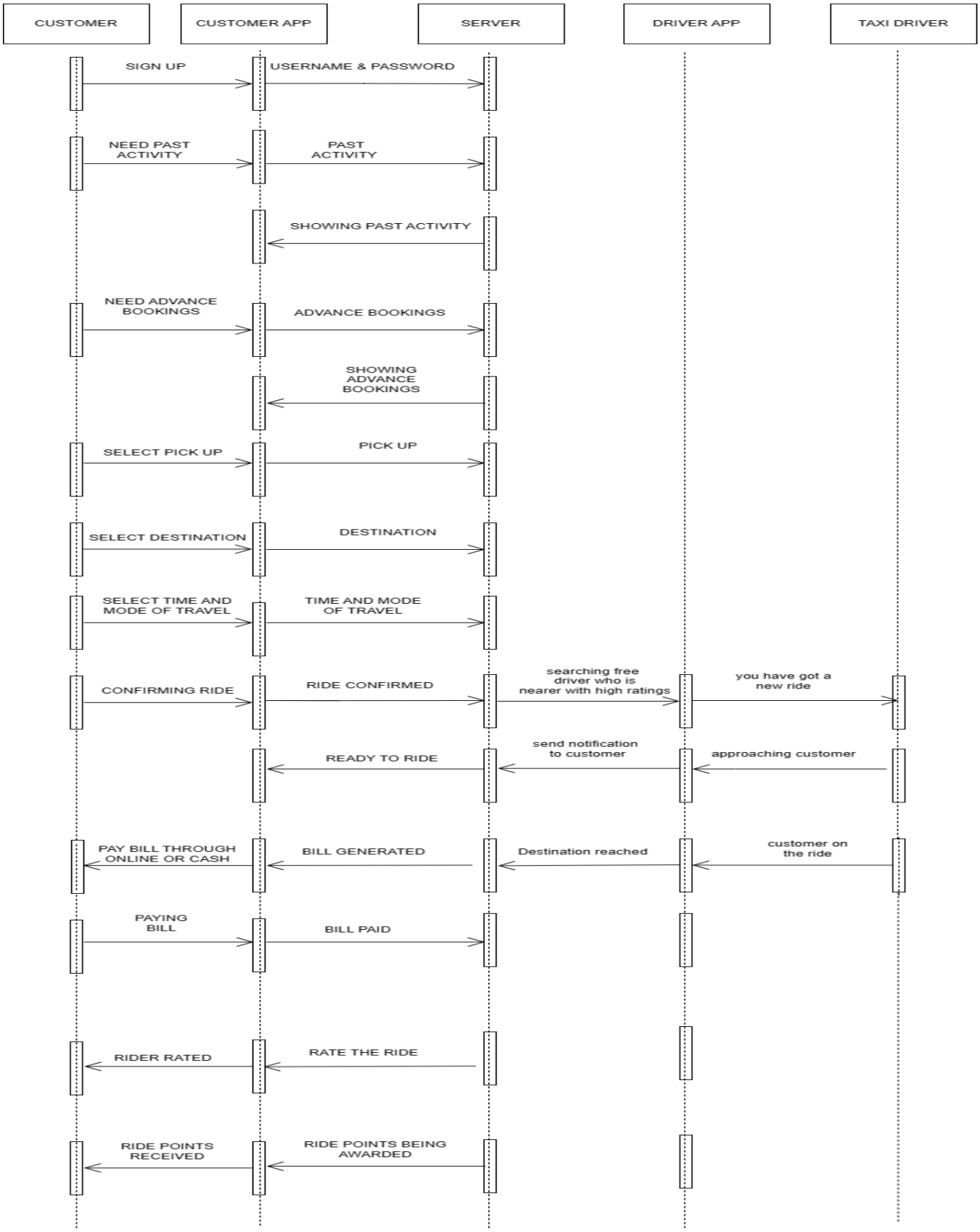
DFD LEVEL 2


ADVANCE BOOKINGS
SETTINGS
SHOWING ADVANCE BOOKING
SHOWING PAST ACTIVITY
USERNAME
USERNAME
USERNAME
CANCELS DATABASE
PICKUP,DESTINATION,DATE,TIME & CAB TYPE
USERNAME
DISTANCE CALCULATION
ADVANCE BOOKING
BOOKING
HOMEPAGE
USERNAME
PREVIOUS CANCELS
DISTANCE CAB TYPE
USERNAME
USERNAME & PASSWORD
CUSTOMER
USERNAME & PASSWORD
LOGIN
PAST ACTIVITY
INSTANT BOOKING
REGISTER
BILL CACULATION
ADVANCE BOOKING DATABASE
USERNAME,PICKUP,DESTINATION CABTYPE,BILL,DATE,TIME
LOGIN, SIGNUP DATABASE
DRIVER
PICKUP,DESTINATION & CAB TYPE
PICKUP,DESTINATION & CAB TYPE
PICKUP,DESTINATION , CAB TYPE
PICKUP,DESTINATION
PAST ACTIVITY DATABASE
USERNAME,PICKUP,DESTINATION CABTYPE,BILL,DATE,TIME
DRIVER ASSIGNING
DISTANCE CALCULATION
ASSIGNED DRIVER AND RATINGS
RATINGS CALCULATION
NEW RATINGS
DRIVER RATINGS DATABASE
DRIVER'S RATING
DISTANCE, CABTYPE
BILL
RATINGS
PAYING BILL
BILL
PREVIOUS CANCELS
CUSTOMER
PAYMENT
CUSTOMER
BILL CALCULATION
CANCELS DATABASE
RIDE HISTORY FOR DISCOUNT
RIDE HISTORY

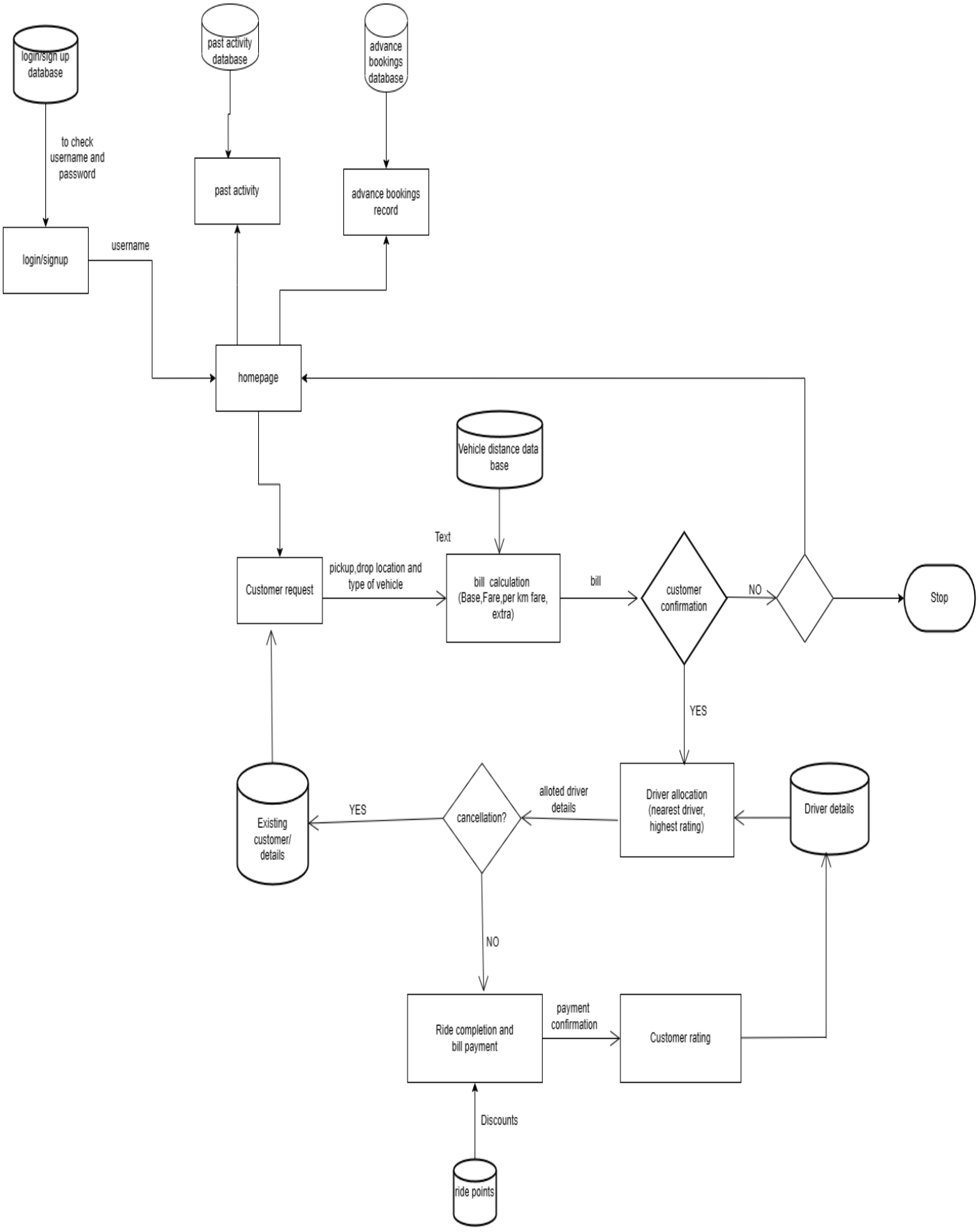
- The user login/sign up by entering the username and password.
- The username and password gets stored in the database(login/signup).
- And this data flows to homepage where user can navigate to settings, past activity, advance bookings and booking module.
- The data(username) flows to past activity, advance booking module to show the past rides and advance bookings booked by the customer who's currently logged in.
- The data from homepage is sent to booking module for the customer to book a ride. It has two options advance booking and instant booking.
- Considering advance booking, Inputs like pickup, destination and vehicle type are entered by the user which further flows to distance calculation and distance generated by it is sent to bill calculation alongwith cab type and data from cancels also reaches here to take previous cancels into consideration and additional fee is charged.
- After which data like username, pickup, destination, date, time, cabtype are stored in the database.
- For instant booking, inputs like pickup, destination, cab type flows to distance calculation where distance between pickup and destination is calculated. The distance alongwith cab type is sent to bill calculation to calculate the bill and this data is stored in past activity database also after successful completion of ride.
- Data from cancels also flows to bill calculation to take previous cancels into consideration.
- Data from instant booking flows to driver assigning to assign the driver based on constraints, here data from driver ratings is also reaches here.
- Assigned driver, ratings of the driver and bill generated is shown to the customer through payment module and also data flows from ride history to give discounts.
- The generated bill is sent to past activity database.
- The customer rates the driver, through rating calculation the new generated rating is stored in the database driver ratings.

DETAILED DESIGN

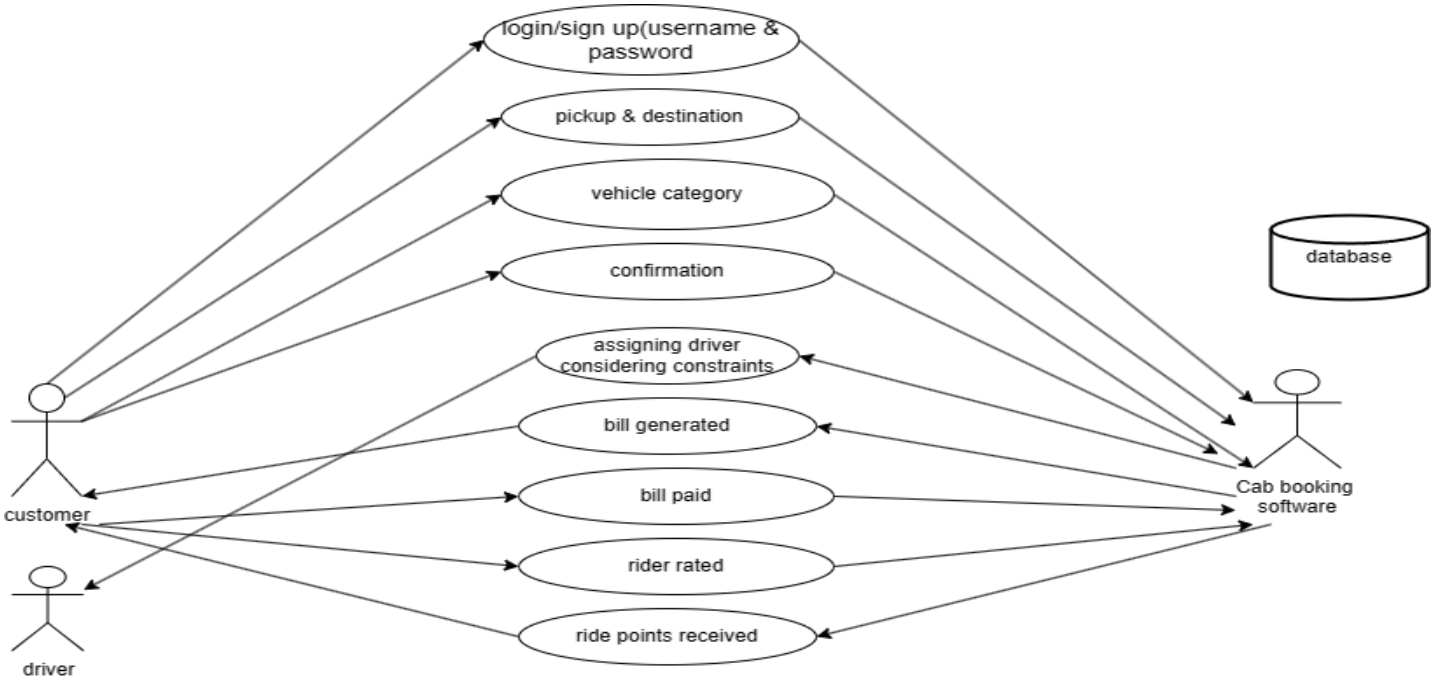
SEQUENCE DIAGRAM



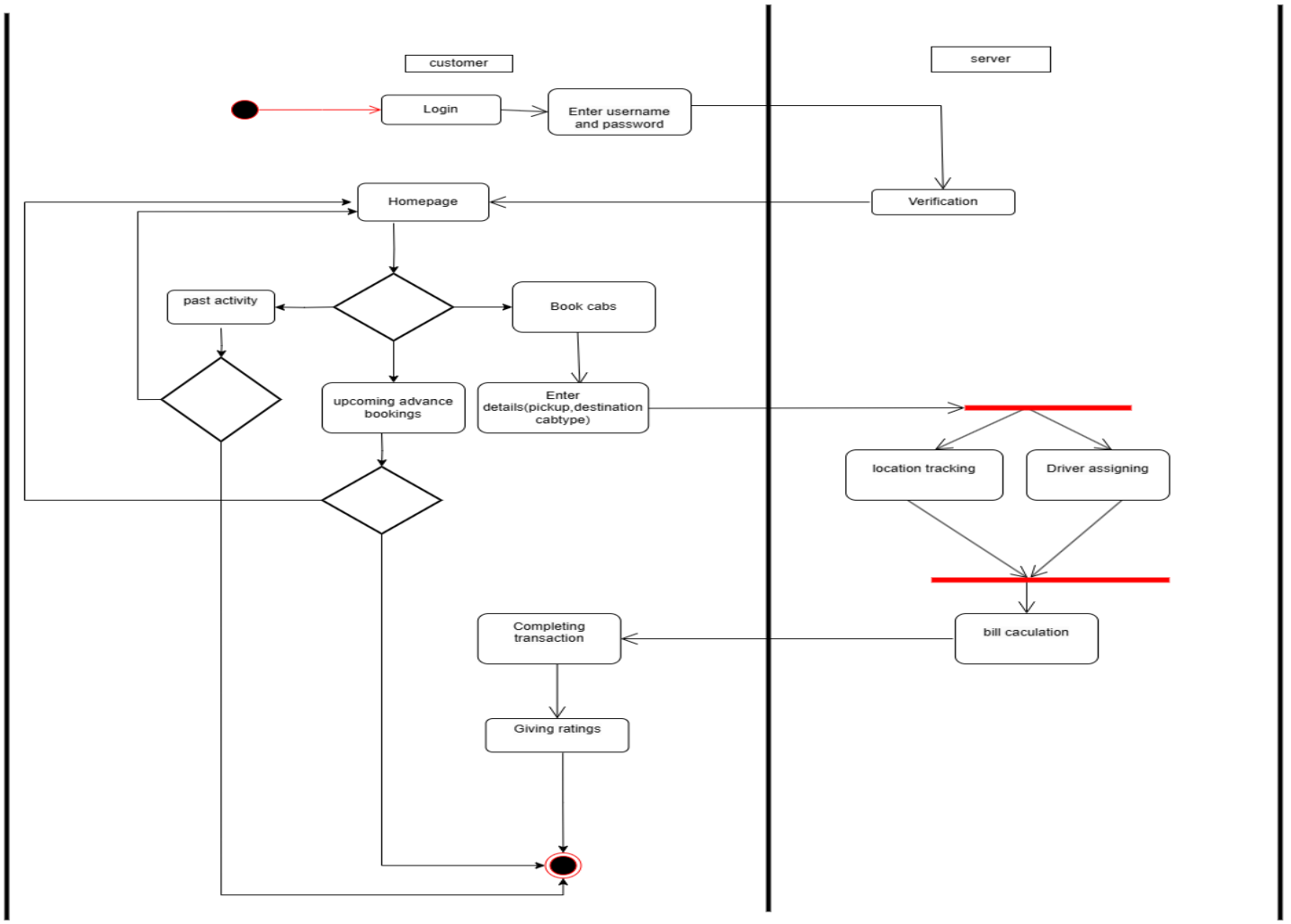
Architecture diagram



USE CASE DIAGRAM



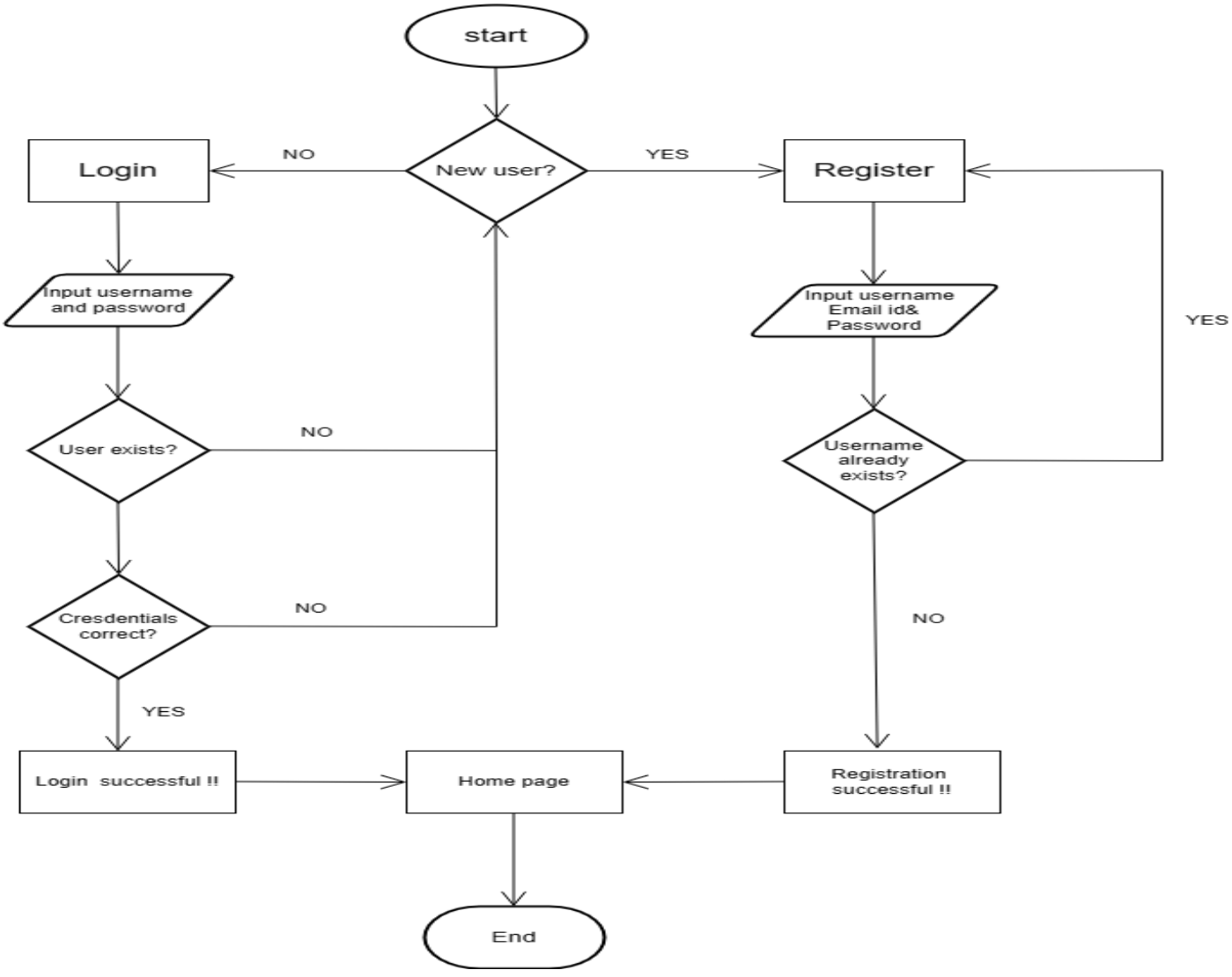
Activity diagram



MODULES

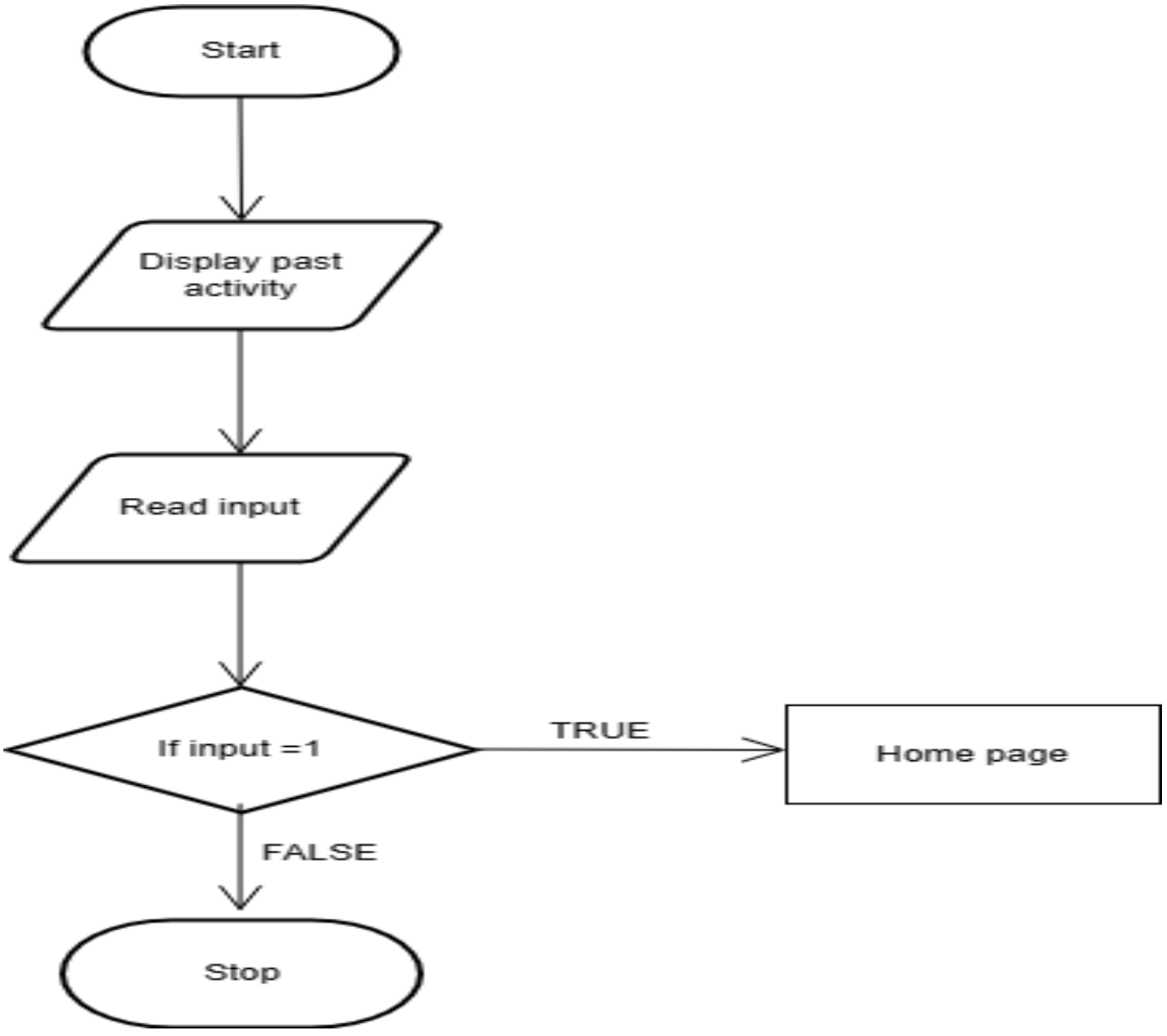
- LOGIN/SIGNUP MODULE - ANANDHARAJ
- PAST ACTIVITY MODULE - DINESH
- ADVANCE BOOKING MODULE - DINESH
- BOOKING MODULE– DINESH
- DRIVER ASSIGNING MODULE- ANANDHARAJ
- BILL CALCULATION MODULE - ANANDHARAJ
- PAYMENT MODULE - DINESH
- RATINGS MODULE - ANANDHARAJ

LOGIN MODULE



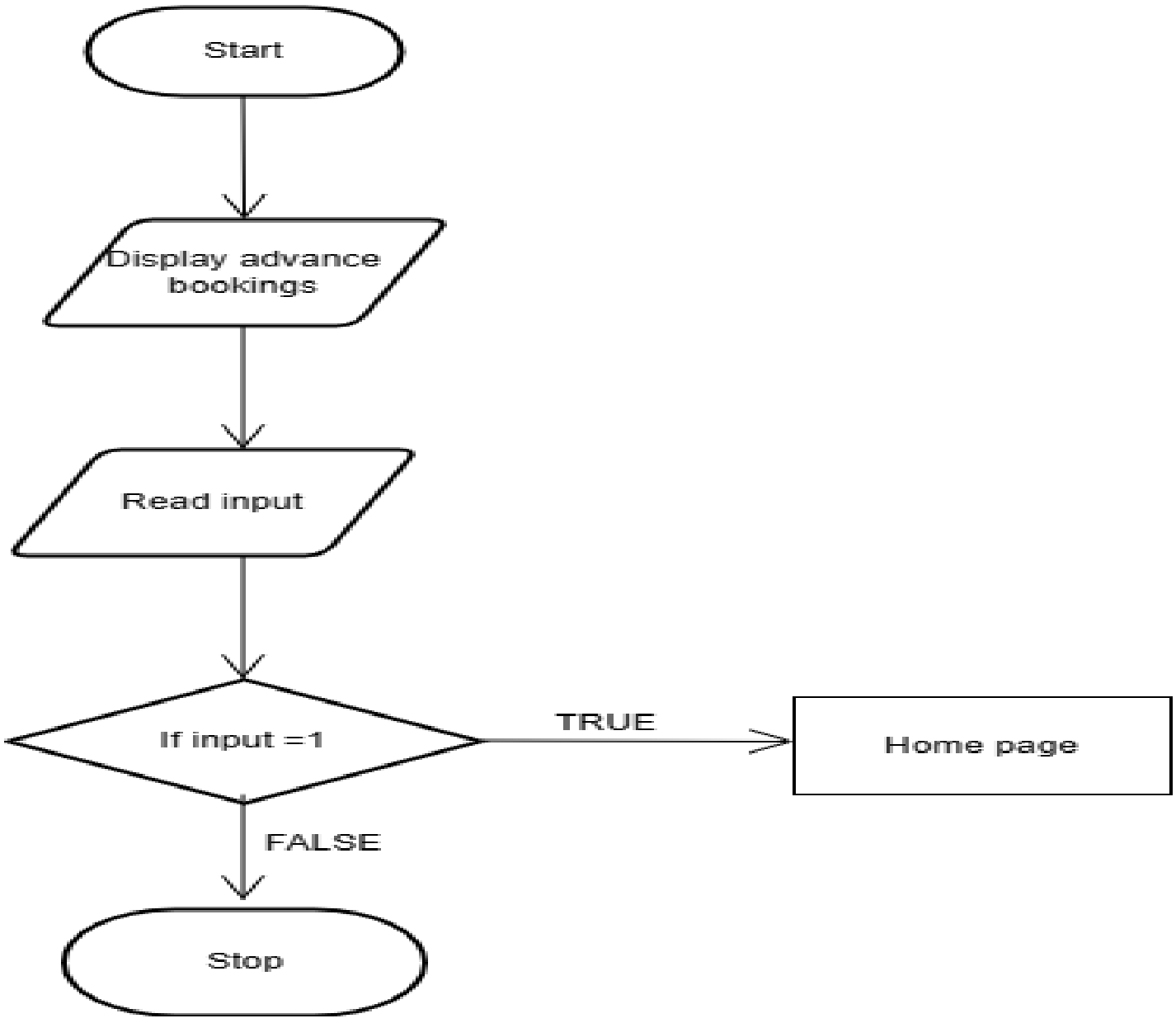
- Sign up to the system if the user is new.
- Else, login to the system.
- Checks for the credentials.
- If given credentials is wrong, login again.
- If given credentials is right, access to software is granted.

PAST ACTIVITY



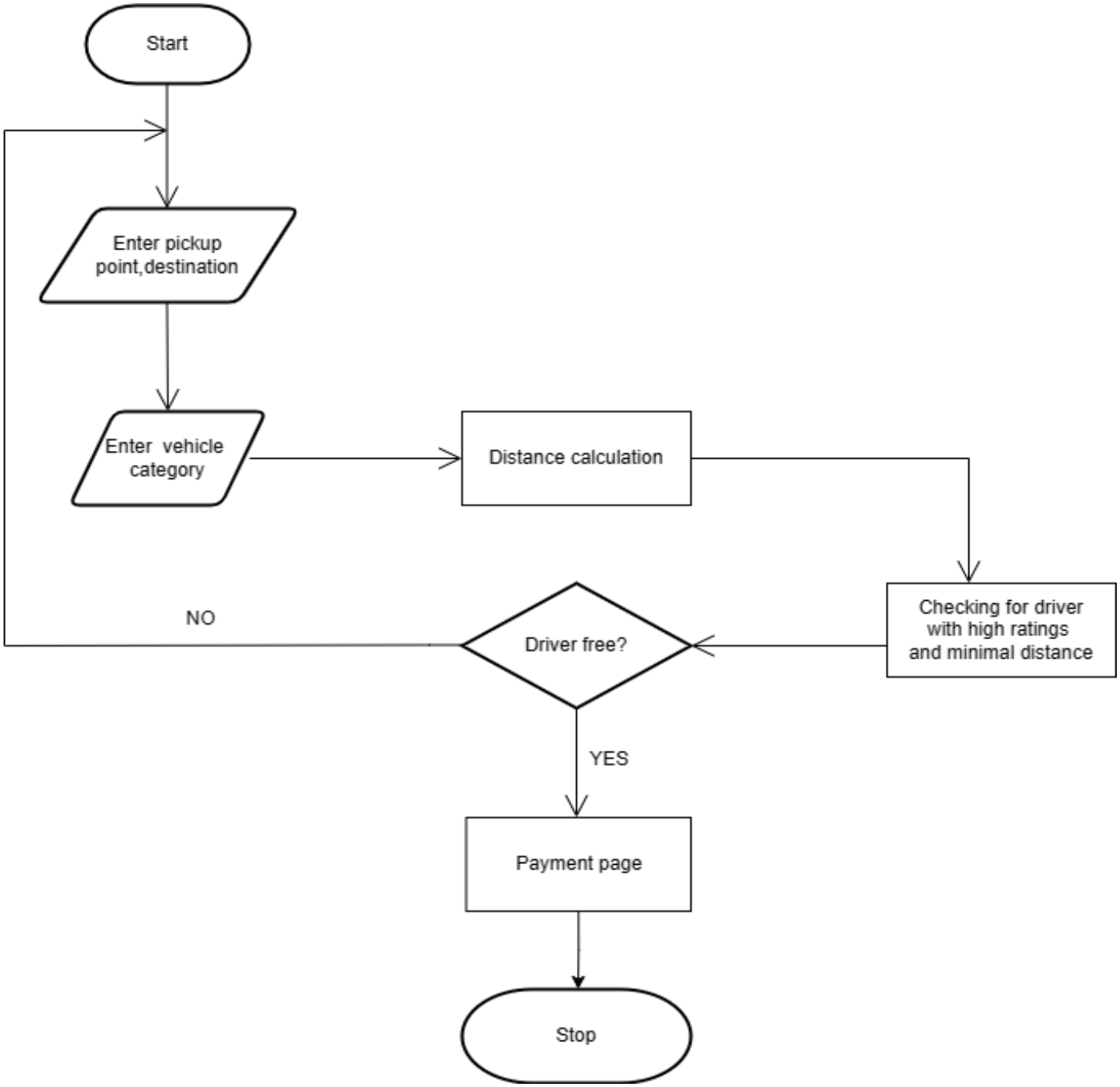
- This helps the user to see his past activity
- And it allows the user to go back to homepage or close the program by giving respective inputs.
- 1 is for going back to homepage.
- 2 is for closing the software.

ADVANCE ACTIVITY



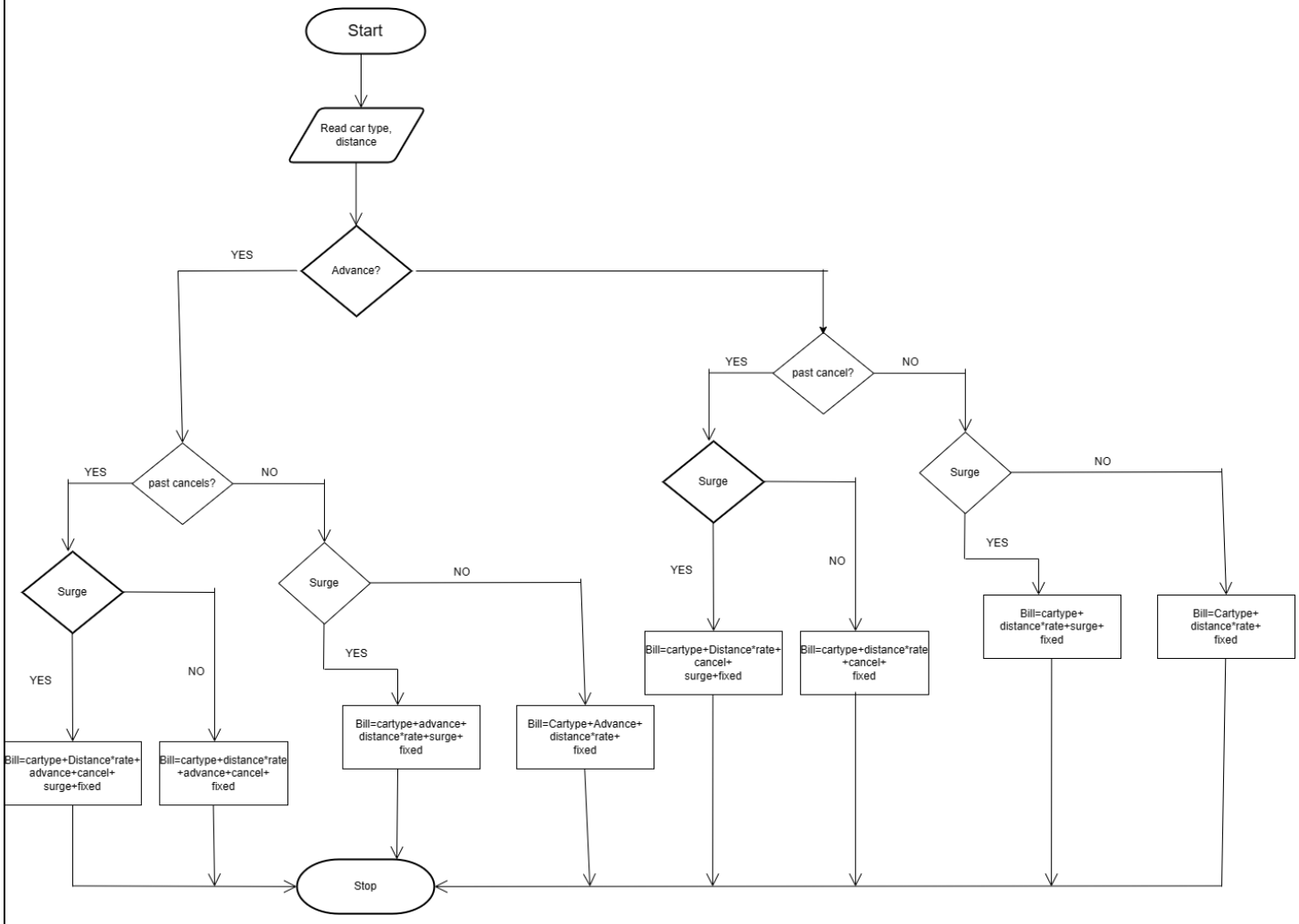
- This helps the user to see his/her advance bookings
- And it allows the user to go back homepage or close the program by giving respective inputs.
- 1 is for going back to homepage.
- 2 is for closing the software.

BOOKING AND DRIVER ASSIGNING MODULE



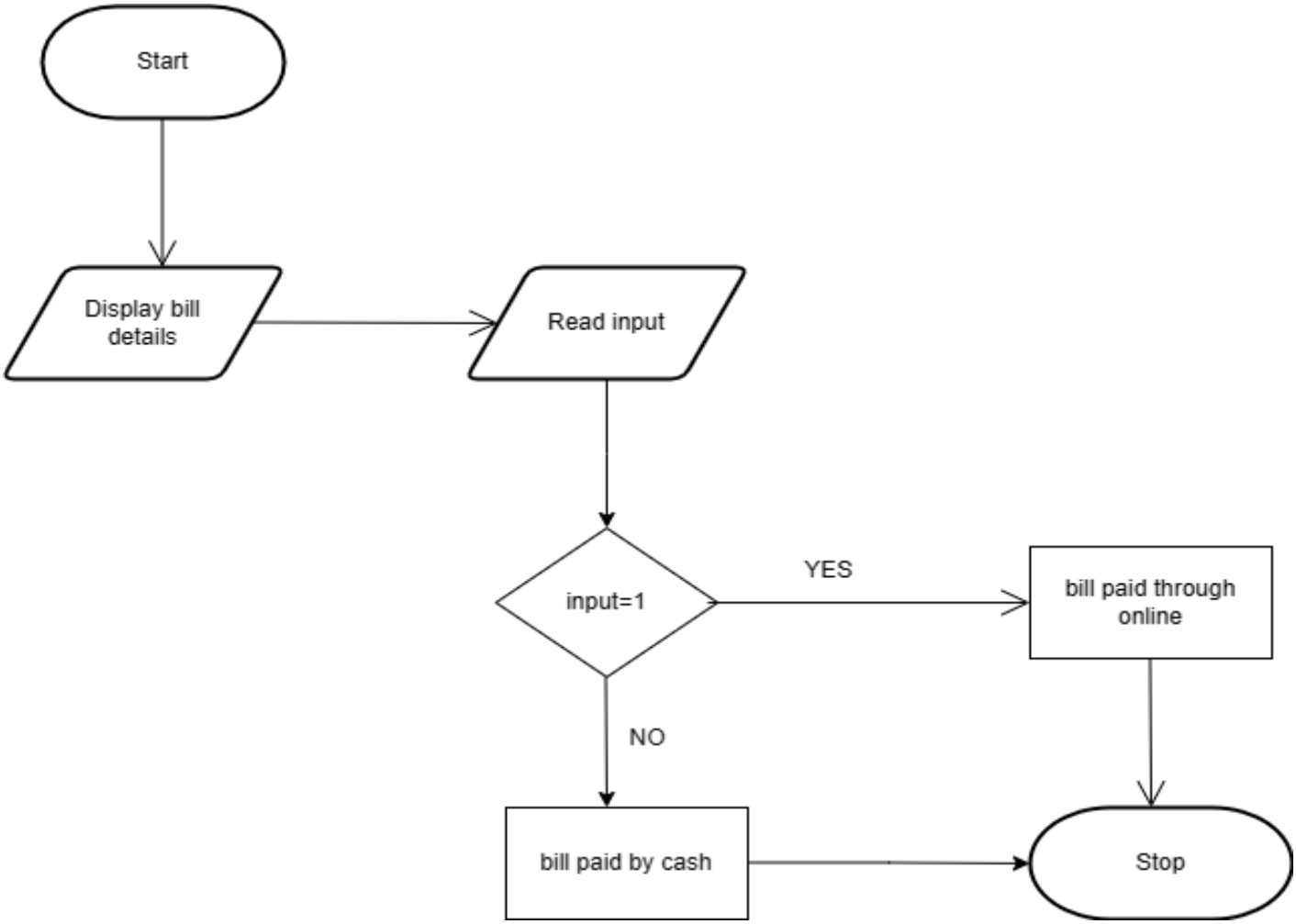
- This module asks the user for inputs like pickup,destination and vehicle category for booking the cab.
- Based on the given inputs distance is calculated.
- Based on the given inputs,driver nearest to customer is assigned.
- If two/more drivers are at the same distance from the customer, driver whose rating is higher is assigned.
- If the driver is found,further process takes place.
- If driver isn't available,book again.
- Both advanced and instant booking can be done here.
- Based on the inputs bill is generated.

Bill calculation module



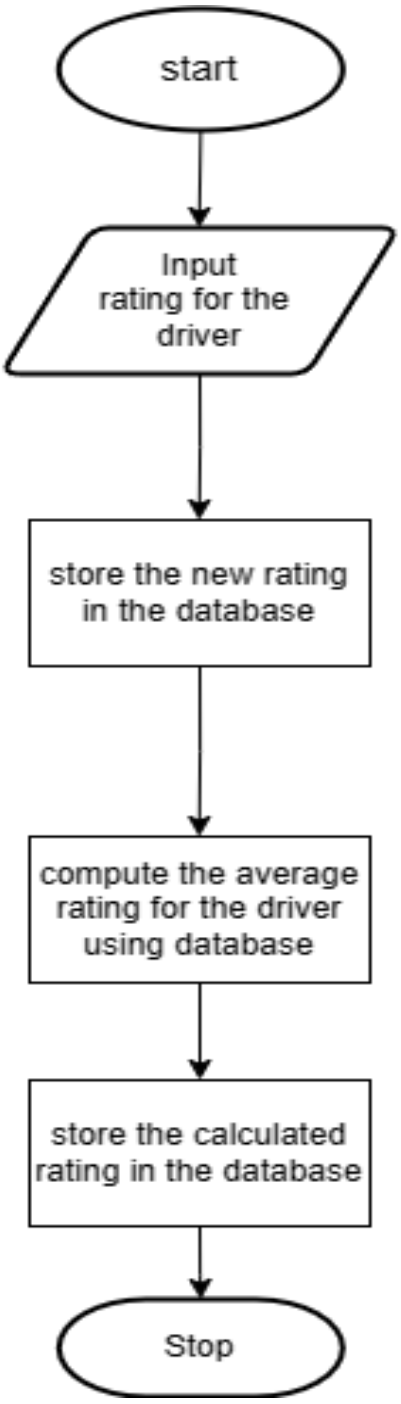
- To calculate the cab bill,distance and cab type are basic needs.
- For advance booking, advance fee is added to the total bill.
- During demand time,surge fee is charged.
- If there is any previous cancels by the user,extra fee is charged for that too.

Payment module



- Customer’s bill details is shown.
- The customer is asked to pay his/her bill through different mode
- Based on customer’s inputs,payment mode is chosen
- Ratings module is opened after successful payment.

Ratings module



- The booking comes to an end with the customer rating the driver
- The new rating is stored in the database(eachratings).
- Using the above database, new average rating is calculated.
- This new calculated average rating is stored to the database(ratingsfinal).

Implementation

Explanation of how the data is organized

- For this project we have used 13 files and 3 structures,1 array and 1 pointer.
- We have used files to store the following.
  - Login/sign up details(login.txt,signup.txt)
  - Previous cancels of the rider(cancels.txt)
  - Advance bookings of the customer(advancebookings.txt)
  - Distance between two locations(distance.txt)
  - Location of the driver(drivermap.txt)
  - The rating the driver gets each time by the customer(eachratings.txt)
  - Distance between two connected points(mapp.txt)
  - Nearest driver from customer during allocation(minimumdistance.txt)
  - Past rides of the customer(past.txt)
  - Ratings of the driver (ratingsfinal.txt)
  - Ride points(rides.txt)
  - Date and time before executing the code(date.txt)
- We've used structure for the following
  - Structure of array to display part of past activity.(past)

```
struct past
{
    char history[30];
};
```

where p is the structure variable.

```
strcpy(p[0].history,"Date:");
strcpy(p[1].history,"Time:");
strcpy(p[2].history,"Starting point:");
strcpy(p[3].history,"Destination:");
strcpy(p[4].history,"Driver:");
strcpy(p[5].history,"Mode of travel:");
strcpy(p[6].history,"Bill:");
```

  - Structure of array to store price for different mode of travel ,like bike, auto etc.

```
struct cars
{
     char modes[20];
     float cost;
};

    struct cars c[7];

    strcpy(c[0].modes,"Bike");
    strcpy(c[1].modes,"Auto");
    strcpy(c[2].modes,"Mini sedan");
    strcpy(c[3].modes,"Sedan");
    strcpy(c[4].modes,"SUV");
    strcpy(c[5].modes,"Innova");
    strcpy(c[6].modes,"Luxury car");

    c[0].cost=10;
    c[1].cost=20;
    c[2].cost=30;
    c[3].cost=35;
    c[4].cost=45;
    c[5].cost=60;
    c[6].cost=75;
```

- Structure of array to store place code for different places and display it.

```
struct map
{
     char place[50];
     int code;
};

    for(int i=1;i<14;i++)
    {
        m[i-1].code=i;
    }

    strcpy(m[0].place,"kelambakkam");
    strcpy(m[1].place,"SSN");
    strcpy(m[2].place,"thiruporur");
    strcpy(m[3].place,"kovalam");
    strcpy(m[4].place,"ECR");
    strcpy(m[5].place,"sholinganallur");
    strcpy(m[6].place,"koyambedu");
    strcpy(m[7].place,"chennaicentral");
    strcpy(m[8].place,"annanagar");
    strcpy(m[9].place,"chromepet");
    strcpy(m[10].place,"tambaram");
    strcpy(m[11].place,"vandalur");
    strcpy(m[12].place,"kattankulathur");

    printf("Place code and Place\n");

    for(int j=0;j<13;j++)
    {
        printf("%d   %s\n",m[j].code,m[j].place);
    }
```

- The one array is to store the bill for different mode of travel and display it for instant booking
- The one pointer is to store the bill for different mode of travel and display it for advance booking.

Rationale behind the selection of a particular language construct

FILES:

- The reason for using files is that the data is not erased after execution of the program.
- This ensures that important information, such as user preferences, ride history, and driver details, remains accessible even after the application is closed and reopened.

STRUCTURES:

- Structures ensures data integrity because related data elements are bundled together, reducing the risk of errors related to data inconsistency or mismatched values.
- Structures can be defined once and reused in different parts of the program,promoting code reusability.

ARRAYS:

- Arrays provide sequential access to elements, making it easy to traverse and process data in a linear manner. This feature is particularly useful for tasks such as searching, sorting, and iterating through data.
- Arrays enable direct access to elements using their indices, which allows for quick and efficient retrieval of data based on its position in the array

Explanation of any other libraries or APIs that have been used

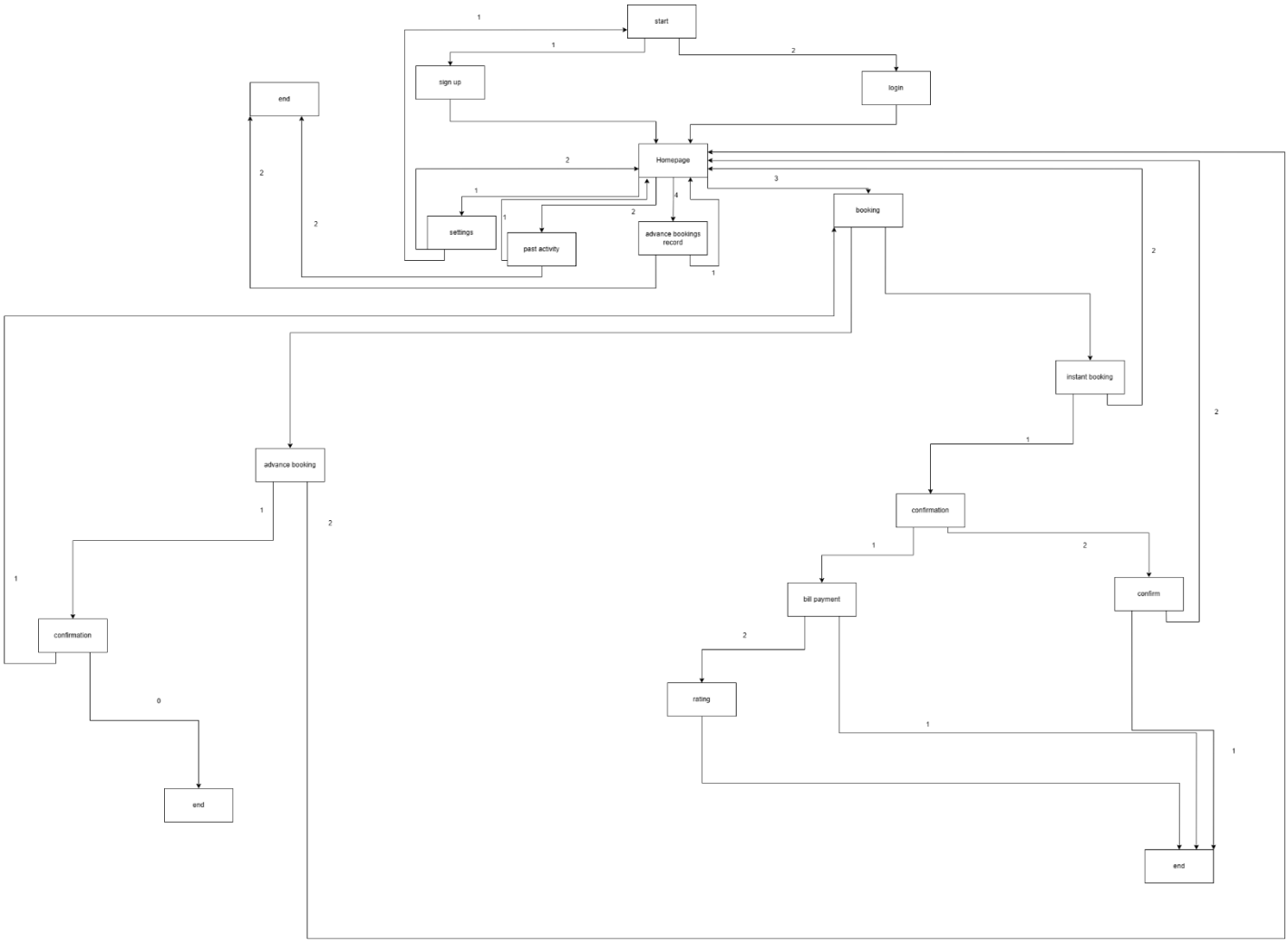
```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>
#include <unistd.h>
```

- stdio.h to handle with standard input and standard output and to handle file input and output. Ex. Printf, scanf, fopen, fclose etc.
- string.h to perform various string operations. Ex .strcmp,strcpy,strcat etc.
- stdlib.h to acess functions defined in stdlib.h functions. Ex exit, srand.
- time.h to provide a time-based seed for the random number generator.
- unistd.h to use sleep function.

Platform used for Code Development

1) codeblocks

User interface design



SAMPLE CODES

MAIN

```
int main()
{
 time_t t1;

srand((unsigned) time(&t1));

FILE *signup;
FILE *login;

printf("press 1 for signup\n");
printf("press 2 for login\n");

int register1=0;
scanf("%d",&register1);

if(register1==1)
{
sign_up(signup,login);
}

else if(register1==2)
{
log_ind(signup,login);
}
return 0;
}
```

LOGIN

```
void log_ind(FILE *signup,FILE *login)
{
int count=0,c=0;

printf("enter your username:");
scanf("%s",username);
strcpy(usernamepassword,username);

printf("enter your password:");
scanf("%s",password);

signup=fopen("signup.txt","r");

if(signup==NULL)
{
printf("user not found, sign up first\n");
log_in(signup,login);
}

else
{
char lines[100];
while (!feof(signup))
{
fscanf(signup,"%s",lines);
if (strcmp(lines,username)==0)
{
count++;
}
}
}
fclose(signup);
strcat(usernamepassword,password);
login=fopen("login.txt","r");
char line[100];

if (count!=0)
{
while(!feof(login))
{
fscanf(login,"%s",line);
if (strcmp(line,usernamepassword)==0)
{
c++;
}
}
fclose(login);
if (c!=0)
{
printf("logged in succesfully\n");
after_signup(signup,login);
}
else
{
printf("username and password don't match\n");
log_in(signup,login);
}
}
else
{
printf("user not found, sign up first\n");
log_in(signup,login);
}
}
```

HOMEPAGE

```
void after_signup(FILE *signup,FILE *login)
{
  int home=0,point=0;

  FILE *past;

  printf("press 1 to go to settings\n");
  printf("press 2 to know your past activity\n");
  printf("press 3 to book a ride\n");
  printf("press 4 to know advance bookings\n");
  scanf("%d",&point);

  if (point==1)
  {
    settings(signup,login);
  }

  else if(point==2)
  {
    activity(past,signup,login);
  }
  else if(point==3)
  {
    book(signup,login);
  }
  else if(point==4)
  {
    advancebookings(signup,login);
  }
}
```

BOOKING

```
void book(FILE *signup,FILE *login)
{
   FILE *advance;
   FILE *mappp;


   struct map m[15];
   int ride,mode,confirm,p1,p2,c1,c2,d1;
   float d2,d3;
   float bill;
   float bill1;


   char date[20],time[20];
   char type[30];


   for(int i=1;i<14;i++)
   {
      m[i-1].code=i;
   }


   strcpy(m[0].place,"kelambakkam");
   strcpy(m[1].place,"SSN");
   strcpy(m[2].place,"thiruporur");
   strcpy(m[3].place,"kovalam");
   strcpy(m[4].place,"ECR");
   strcpy(m[5].place,"sholinganallur");
   strcpy(m[6].place,"koyambedu");
   strcpy(m[7].place,"chennaicentral");
   strcpy(m[8].place,"annanagar");
   strcpy(m[9].place,"chromepet");
   strcpy(m[10].place,"tambaram");
   strcpy(m[11].place,"vandalur");
   strcpy(m[12].place,"kattankulathur");


   printf("places code and places\n");


   for(int j=0;j<13;j++)
   {
      printf("%d %s\n",m[j].code,m[j].place);
   }


   printf("nearby places/connected points\n");
   printf("1-2\n2-3\n1-4\n4-5\n1-6\n6-7\n7-8\n7-9\n7-10\n10-11\n11-12\n12-13\n");


   mappp=fopen("mapp.txt","r");


   while(fscanf(mappp,"%d %d %d",&c1,&c2,&d1)!=EOF)
   {
      printf("distance between two points %d %d is %d\n",c1,c2,d1);
   }




   printf("press 1 to book in advance\n");
   printf("press 2 for instant ride\n");


   scanf("%d",&ride);


      else if(ride==2)
      {
         printf("Around which region you want the driver to pick
you up(code)(leavingplacecode headingplacecode):");
         scanf("%d %d",&p1,&p2);

         printf("how far you want the pick from the %d:",p1);
         scanf("%f",&d2);


         printf("enter the pick up point(leavingplace):");
         scanf("%s",pickup);

         printf("enter your destination(place):");
         scanf("%s",destination);

         printf("how far you want the drop from destination: ");
         scanf("%f",&d3);

         amount(pickup,destination,d2,d3);

         printf("select your mode of travel\n");
         printf("press 3 for bike\n");
         printf("press 4 for auto\n");
         printf("press 5 for mini sedan\n");
         printf("press 6 for sedan\n");
         printf("press 7 for SUV\n");
         printf("press 8 for innova\n");
         printf("press 9 for luxury car\n");

         scanf("%d",&mode);


         if(mode==3)
         {
            printf("%f\n",bill_arr[0]);
            bill=bill_arr[0];
            strcpy(type,"bike");

            booking(type,p1,p2,d2,signup,login,bill);
         }

         else if(mode==4)
         {
            printf("%f\n",bill_arr[1]);
            bill=bill_arr[1];
            strcpy(type,"auto");

            booking(type,p1,p2,d2,signup,login,bill);
         }

         else if(mode==5)
         {
            printf("%f\n",bill_arr[2]);
            bill=bill_arr[2];
            strcpy(type,"minisedan");

            booking(type,p1,p2,d2,signup,login,bill);
         }

         else if(mode==6)
         {
            printf("%f\n",bill_arr[3]);
            bill=bill_arr[3];
            strcpy(type,"sedan");

            booking(type,p1,p2,d2,signup,login,bill);
         }
         else if(mode==7)
         {
            printf("%f\n",bill_arr[4]);
            bill=bill_arr[4];
            strcpy(type,"SUV");

            booking(type,p1,p2,d2,signup,login,bill);
         }

         else if(mode==8)
         {
            printf("%f\n",bill_arr[5]);
            bill=bill_arr[5];
            strcpy(type,"innova");

            booking(type,p1,p2,d2,signup,login,bill);
         }

         else if(mode==9)
         {
            printf("%f\n",bill_arr[6]);
            bill=bill_arr[6];
            strcpy(type,"luxury");

            booking(type,p1,p2,d2,signup,login,bill);
         }
      }
   }
```

DRIVER ASSIGNING

```
void booking(char type[],int p1,int p2,int d2,FILE *signup,FILE *login,float bill)
{
   FILE *overallrating;
   FILE* drivermapp;
   FILE *samedistance;

   int confirm;

   char
name[30],category[30],category1[30],driver[30],drivername[30],minname[30];
   char name_final[30];
   float rating,max,dr1,dr2,r1,r2,t,count=0;
   float min;
   drivermapp=fopen("drivermap.txt","r");
   while(fscanf(drivermapp,"%s %s %f %f %f %f
%f",drivername,category1,&r1,&r2,&dr1,&t,&dr2)!=EOF)
   {
      if(strcmp(category1,type)==0 && ((r1==p1&&r2==p2)||(r1==p2 &&
r2==p1)))
      {
         count++;
         if(p1==r1)
         {
            if(d2>dr1)
            {
               min=d2-dr1;
            }
            else if(dr1>d2)
            {
               min=dr1-d2;
            }
            else
            {
               min=dr1-d2;
            }
         }

         else if(p1==r2)
         {
            if(d2>dr2)
            {
               min=d2-dr2;
            }
            else if(dr2>d2)
            {
               min=dr2-d2;
            }
            else
            {
               min=dr2-d2;
            }
         }
         break;
      }

   }
         printf("%.1f\n",min);
         if(count==0)
         {
            printf("Drivers not found\n");
            printf("Please try again\n");
            book(signup,login);
         }
         fclose(drivermapp);

         drivermapp=fopen("drivermap.txt","r");

while(fscanf(drivermapp,"%s %s %f %f %f %f %f",drivername,category1,&r1,&r2,&dr1,&t,&dr2)!=EOF)
         {
            if(strcmp(category1,type)==0 && ((r1==p1&&r2==p2)||(r1==p2 && r2==p1)))
            {
               if(p1==r1)
               {
                  if(d2>dr1)
                  {
                     if(d2-dr1<min)
                     {
                        min=d2-dr1;
                     }

                  }

                  else if(dr1>d2)
                  {
                     if(dr1-d2<min)
                     {
                        min=dr1-d2;
                     }
                  }
                  else
                  {
                     if(dr1-d2<min)
                     {
                        min=dr1-d2;
                     }
                  }
               }

               else if(p1==r2)
               {
                  if(d2>dr2)
                  {
                     if(d2-dr2<min)
                     {
                        min=d2-dr2;
                     }

                  }

                  else if(dr2>d2)
                  {
                     if(dr2-d2<min)
                     {
                        min=dr2-d2;
                     }
                  }
                  else
                  {
                     if(dr2-d2<min)
                     {
                        min=dr2-d2;
                     }
                  }
               }
            }
         }

         fclose(drivermapp);

         printf("%.1f\n",min);

         drivermapp=fopen("drivermap.txt","r");
         samedistance=fopen("minimumdistance.txt","w");
```

```
while(fscanf(drivermapp,"%s %s %f %f %f %f
%f",drivername,category1,&r1,&r2,&dr1,&t,&dr2)!=EOF)
        {
          if(strcmp(category1,type)==0 && ((r1==p1&&r2==p2)||(r1==p2 && r2==p1)))
          {
             if(p1==r1)
             {
                if(dr1>d2)
                {
                   if(dr1-d2==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }

                else if(d2>dr1)
                {
                   if(d2-dr1==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }
                else if(d2==dr1)
                {
                   if(d2-dr1==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }
             }

             else if(p1==r2)
             {
                if(dr2>d2)
                {
                   if(dr2-d2==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }

                else if(d2>dr2)
                {
                   if(d2-dr2==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }
                else if(d2==dr2)
                {
                   if(d2-dr2==min)
                   {
                      fprintf(samedistance,"%s\n",drivername);
                      printf("%s\n",drivername);
                   }
                }
             }
          }
        }
        fclose(drivermapp);
        fclose(samedistance);
        samedistance=fopen("minimumdistance.txt","r");

        while(fscanf(samedistance,"%s",minname)!=EOF)
        {
           overallrating=fopen("ratingsfinal.txt","r");
           while(fscanf(overallrating,"%s %s %f",name,category,&rating)!=EOF)
           {
              if(strcmp(minname,name)==0)
              {
                 max=rating;
                 break;
              }
           }
           fclose(overallrating);
           break;
        }
        fclose(samedistance);
        fclose(overallrating);
        printf("%.1f\n",max);

        samedistance=fopen("minimumdistance.txt","r");

        while(fscanf(samedistance,"%s",minname)!=EOF)
        {
           overallrating=fopen("ratingsfinal.txt","r");
           while(fscanf(overallrating,"%s %s %f",name,category,&rating)!=EOF)
           {
              if(strcmp(minname,name)==0)
              {
                 if(rating>max)
                 {
                    max=rating;

                 }
              }
           }
           fclose(overallrating);
        }
        fclose(samedistance);
        fclose(overallrating);

        printf("%.1f\n",max);

        samedistance=fopen("minimumdistance.txt","r");

        while(fscanf(samedistance,"%s",minname)!=EOF)
        {
           overallrating=fopen("ratingsfinal.txt","r");
           while(fscanf(overallrating,"%s %s %f",name,category,&rating)!=EOF)
           {
              if(strcmp(minname,name)==0)
              {
                 if(max==rating)
                 {
                    printf("%s\n",name);
                    strcpy(name_final,name);
                 }

              }
           }
           fclose(overallrating);
        }
        fclose(samedistance);
        fclose(overallrating);

        printf("sending notification to driver\n");
        sleep(5);

        confirmation(name_final,max,signup,login,bill);
}
```

Bill calculation

```
void amount_ad(char pickup[],char destination[],float d2,float d3)
{
    FILE *distancecalc;
    FILE *cancelss;
    FILE *temp;


    char start[30],end[30];
    float distance=0,rate=3,price,distance0;


    float fixed=15;
    float surge;
    float total;
    float advance=10;
    int random;
    int count;
    char users[30],name1[30];
    struct cars c[7];


    strcpy(c[0].modes,"bike");
    strcpy(c[1].modes,"auto");
    strcpy(c[2].modes,"mini sedan");
    strcpy(c[3].modes,"sedan");
    strcpy(c[4].modes,"SUV");
    strcpy(c[5].modes,"innova");
    strcpy(c[6].modes,"luxury car");


    c[0].cost=10;
    c[1].cost=20;
    c[2].cost=30;
    c[3].cost=35;
    c[4].cost=45;
    c[5].cost=60;
    c[6].cost=75;


    distancecalc=fopen("distance.txt","r");


    while(fscanf(distancecalc,"%s %s %f",start,end,&distance0)!=EOF)
    {
        if(strcmp(pickup,start)==0||strcmp(pickup,end)==0)
        {
            if(strcmp(destination,end)==0 || strcmp(destination,start)==0)
            {
                distance=distance0;
            }
        }
    }
    distance=distance-d2-d3;
    fclose(distancecalc);
    printf("calculated distance %f\n",distance);
    surge=5+(distance*2);
    random=rand()%2+1;



    if (random==1)
    {
        printf("no surge\n");
        price=fixed+(distance*rate);
    }


    if(random==2)
    {
        printf("surge\n");
        price=fixed+surge+(distance*rate);


    }


    cancelss=fopen("cancels.txt","r");


    while(fscanf(cancelss,"%s",users)!=EOF)
    {
        if(strcmp(users,username)==0)
        {
            count++;
        }
    }
    fclose(cancelss);


    if(count!=0)
    {
        printf("You have cancelled previous orders %d times\n",count);
        printf("You will be charged for that\n");


        printf("car category and its cost of travel\n");
        fclose(cancelss);


        for(int i=0;i<7;i++)
        {
            total=c[i].cost+price+(count*10);
            *(p+i)=total+advance;
            printf("%s-%f\n",c[i].modes,*(p+i));
        }
    }


    else
    {
        printf("car category and its cost of travel\n");
        for(int i=0;i<7;i++)
        {
            total=c[i].cost+price;
            *(p+i)=total+advance;
            printf("%s-%f\n",c[i].modes,*(p+i));
        }
    }
}
```

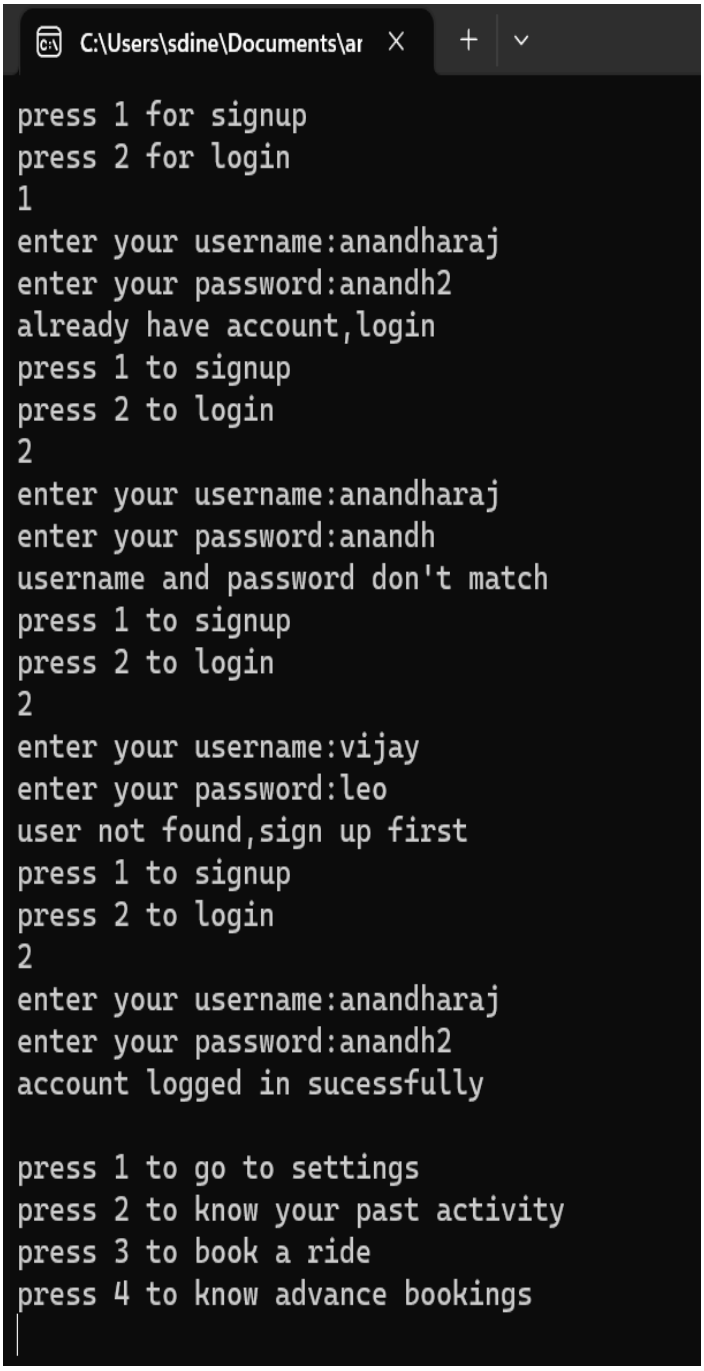
BILL PAYMENT

```
void bill_payment(char name_final[],float bill)
{
   FILE *ride;
   FILE *temp;
   char user[30],name[30];
   int count=0;
   float bill_final;
   int next;

   ride=fopen("rides.txt","r");
   while(fscanf(ride,"%s",user)!=EOF)
   {
      if(strcmp(user,username)==0)
      {
         count++;
      }
   }
   fclose(ride);

   if(count!=0)
   {

      printf("Your ride history shows previous bookings of %d times\n",count);
      printf("You will be discounted for it\n");
      bill_final=bill-(count*3);

      temp=fopen("temp.txt","w");
      ride=fopen("rides.txt","r");

      while(fscanf(ride,"%s",name)!=EOF)
      {
         if(strcmp(name,username)!=0)
         {
            fprintf(temp,"%s\n",name);
         }

         else
         {
            continue;
         }
      }

      fclose(ride);
      fclose(temp);

      remove("rides.txt");
      rename("temp.txt","rides.txt");

      pastrecord(bill_final);

      printf("Bill: %.1f\n",bill_final);

      printf("press 1 to pay through UPI\n");
      printf("press 2 to pay with cash\n");

      scanf("%d",&next);

      if(next==1)
      {
         printf("BILL paid sucessfully through UPI\n");
      }

      else if(next==2)
      {
         printf("BILL paid successfully by cash\n");
      }
   }

   else
   {
      bill_final=bill;
      printf("Bill: %.1f\n",bill_final);

      printf("press 1 to pay through UPI\n");
      printf("press 2 to pay with cash\n");

      scanf("%d",&next);

      if(next==1)
      {
         printf("BILL paid sucessfully through UPI\n");
      }

      else if(next==2)
      {
         printf("BILL paid successfully by cash\n");
      }
      pastrecord(bill_final);
   }

   ride=fopen("rides.txt","a");
   fprintf(ride,"%s\n",username);
   fclose(ride);

   ratings(name_final);
}
```

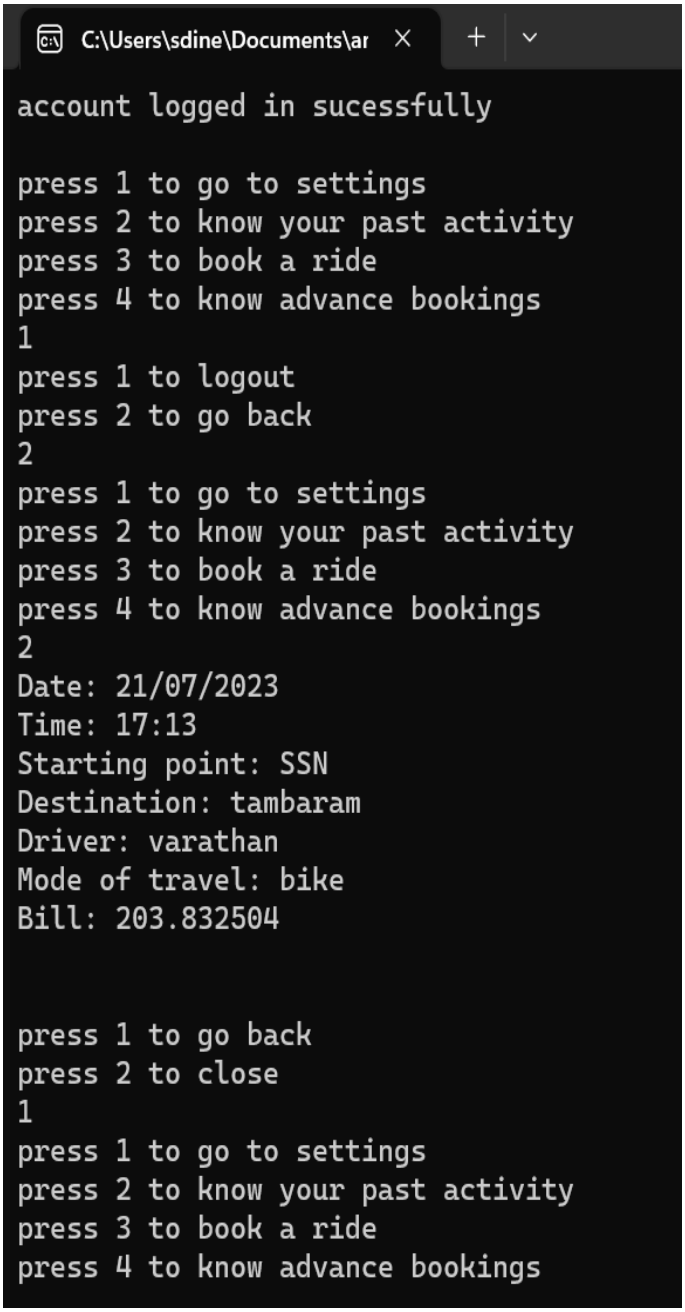
RATINGS

```
void ratings(char name_final[])
{
   FILE *allrating;
   FILE *overallrating;
   FILE *temp;
   double newrating=0,newratinginput=0,rating1=0;
   char name[50],name1[50],category[30];
   double ratingsum=0,calcrating=0;
   int count=0;

   printf("enter your rating for the driver:");
   scanf("%lf",&newratinginput);

   allrating=fopen("eachratings.txt","a");
   fprintf(allrating,"%s",name_final);
   fprintf(allrating," %.1f\n",newratinginput);
   fclose(allrating);

   allrating=fopen("eachratings.txt","r");

   while(fscanf(allrating,"%s %lf",name,&newrating)!=EOF)
   {
      if(strcmp(name,name_final)==0)
      {
         count++;
         ratingsum+=newrating;
      }
   }

   calcrating=ratingsum/count;
   printf("%f",calcrating);

   overallrating=fopen("ratingsfinal.txt","r+");
   temp=fopen("temporary.txt","w");


   while(fscanf(overallrating,"%s %s %lf",name1,category,&rating1)!=EOF)
   {
      if(strcmp(name1,name_final)==0)
      {
         fprintf(temp,"%s ",name1);
         fprintf(temp,"%s ",category);
         fprintf(temp,"%.1f\n",calcrating);
      }

      else
      {
         fprintf(temp,"%s ",name1);
         fprintf(temp,"%s ",category);
         fprintf(temp,"%.1f\n",rating1);
      }
   }

   fclose(allrating);
   fclose(overallrating);
   fclose(temp);

   remove("ratingsfinal.txt");
   rename("temporary.txt","ratingsfinal.txt");

}
```

TEST CASES

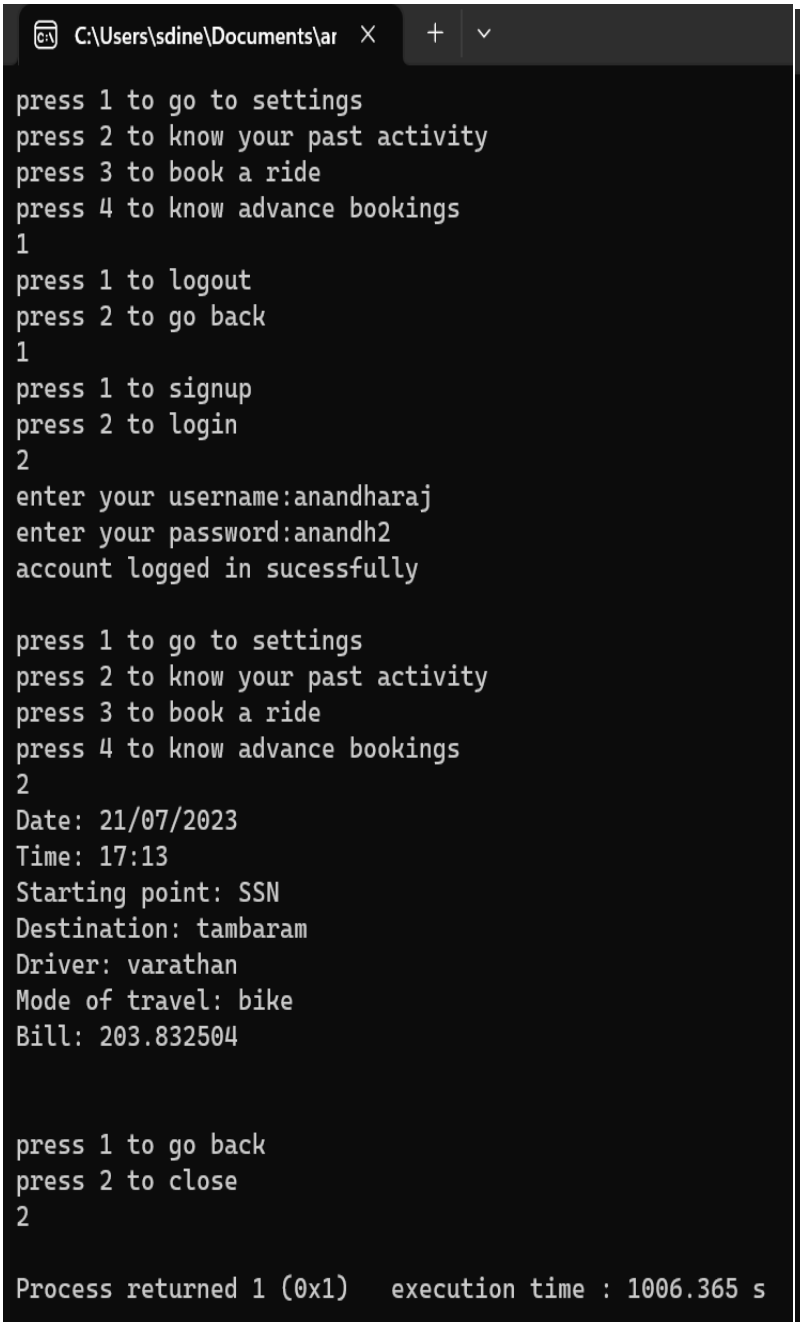


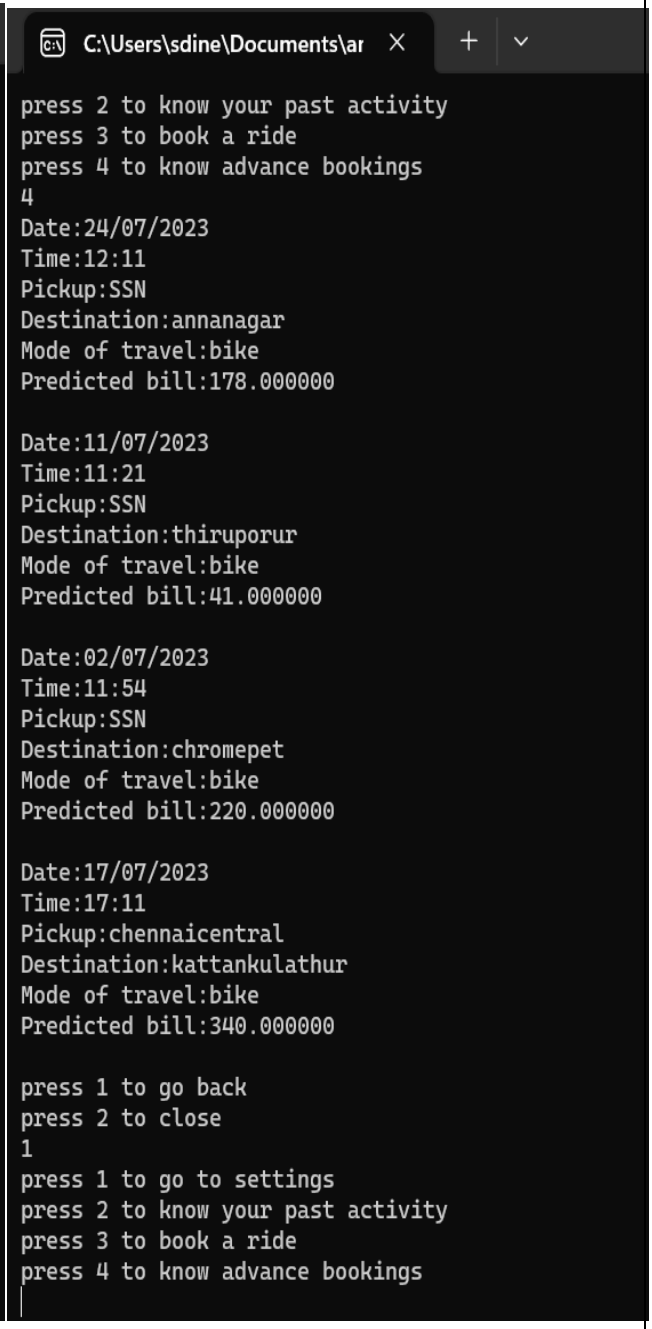
All possible ways of login/sign up is tried and actual output is same as expected output.



1 takes you to settings. From where you can try to log out or come back again to homepage. From homepage 2 lets you see the past bookings. Pressing 1 brings you back to homepage.

Actual output is verified with expected output.



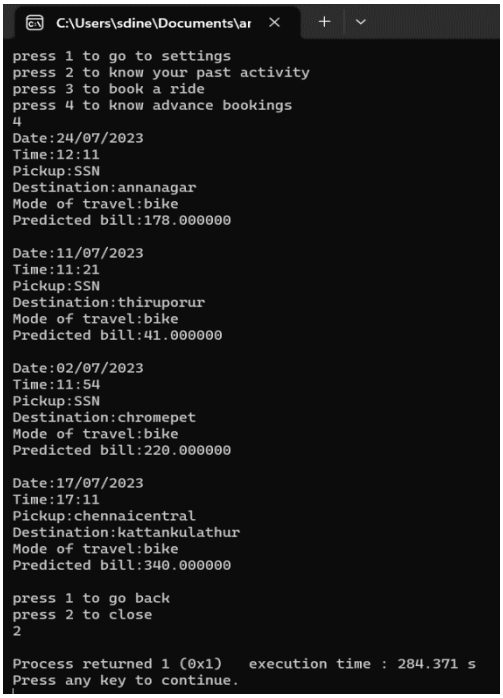


Pressing 1 allows you to log out from settings. Again you can log in and you can close the program by clicking 2 from past activity.

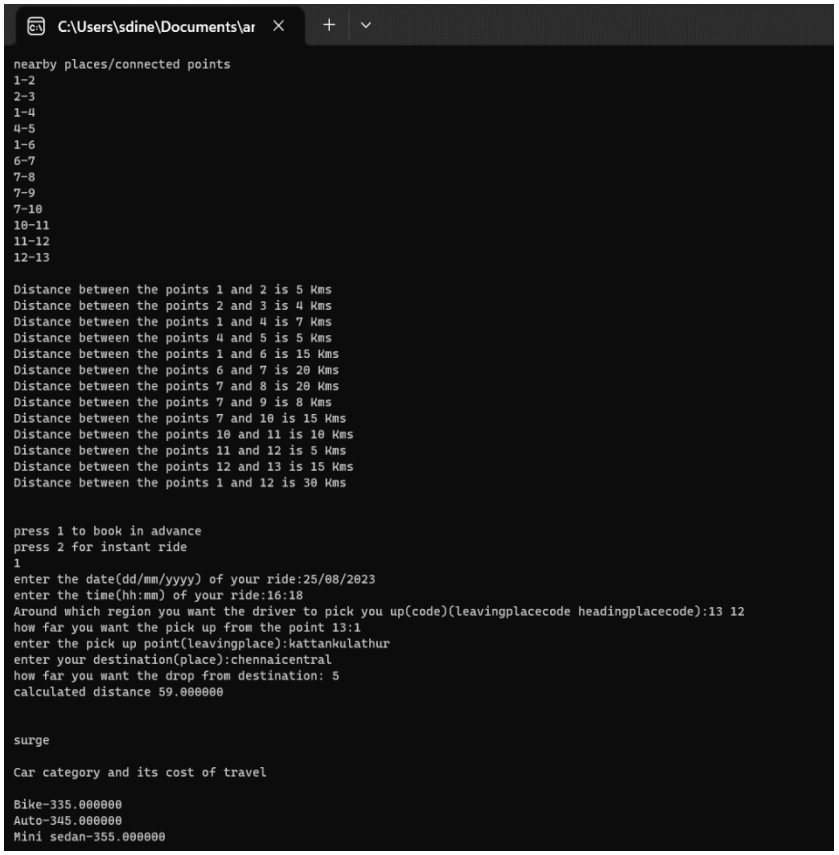
Actual output and expected output is verified.

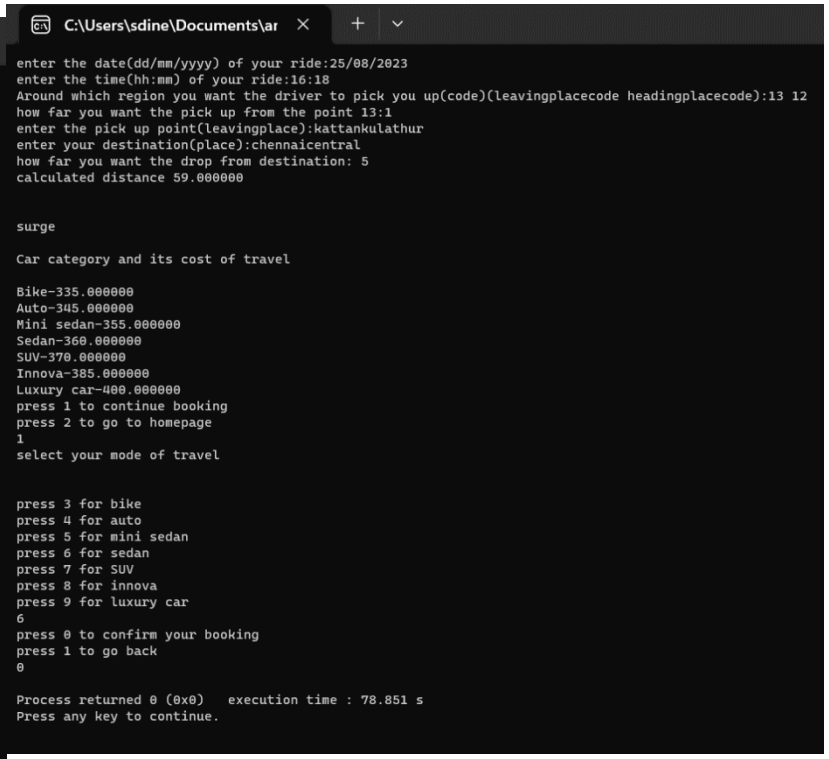
4 takes you to see advance bookings and from there you can come back to homepage.

Actual output and expected output matches.



Here also, closing the program option is given. Actual output is verified with expected output.





We tried to book in advance, we haven't cancelled anywhere, so the ride is booked and stored in the file actual and expected output are same.

```
C:\Users\sdine\Documents\ar X + v

press 1 to book in advance
press 2 for instant ride
1
enter the date(dd/mm/yyyy) of your ride:27/08/2023
enter the time(hh:mm) of your ride:17:18
Around which region you want the driver to pick you up(code)(leavingplacecode headingplacecode):6 11
how far you want the pick up from the point 6:1
enter the pick up point(leavingplace):sholinganallur
enter your destination(place):koyambedu
how far you want the drop from destination: 2
calculated distance 17.000000


surge

Car category and its cost of travel

Bike-125.000000
Auto-135.000000
Mini sedan-145.000000
Sedan-150.000000
SUV-160.000000
Innova-175.000000
Luxury car-190.000000
press 1 to continue booking
press 2 to go to homepage
2
press 1 to go to settings
press 2 to know your past activity
press 3 to book a ride
press 4 to know advance bookings
```

If we don’t confirm the ride, the advance booking doesn’t get store in the file. And it takes you to homepage.

Output is verified

```
C:\Users\sdine\Documents\ar X + v


press 1 to book in advance
press 2 for instant ride
2
Around which region you want the driver to pick you up(code)(leavingplacecode headingplacecode):8 7
how far you want the pick up fron the point 8:2
enter the pick up point(leavingplace):chennaicentral
enter your destination(place):SSN
how far you want the drop from destination: 1
calculated distance 57.000000


surge

Car category and its cost of travel


Bike-315.000000
Auto-325.000000
Mini sedan-335.000000
Sedan-340.000000
SUV-350.000000
Innova-365.000000
Luxury car-380.000000
press 1 to continue booking
press 2 to go to homepage
1
select your mode of travel


press 3 for bike
press 4 for auto
press 5 for mini sedan
press 6 for sedan
press 7 for SUV
press 8 for innova
press 9 for luxury car
5
sending notification to driver


Your ride is booked successfully


Driver name: umran
Ratings: 4.200000


press 1 to confirm your ride
press 2 to cancel the ride
1
Ride starts
Your destination is reached


Your ride history shows previous bookings

You will be discounted for it


Bill: 320.509583
press 1 to pay through online
```

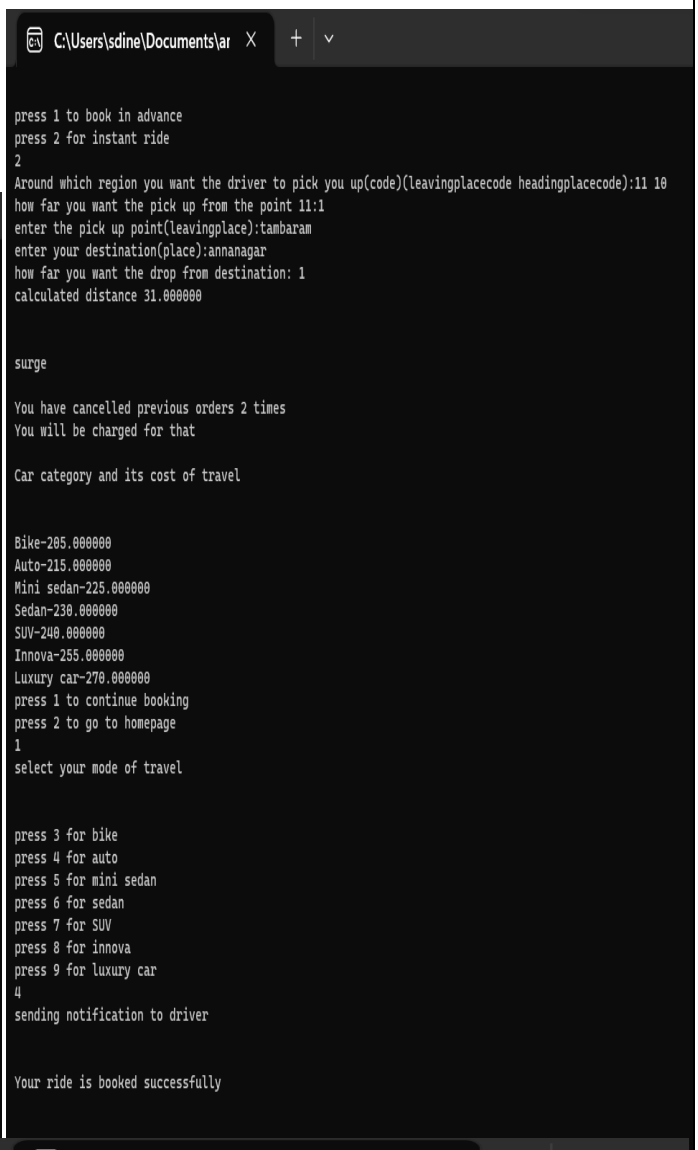
Here, we can stop booking in between if we aren’t okay with the bill.it takes you to homepage.

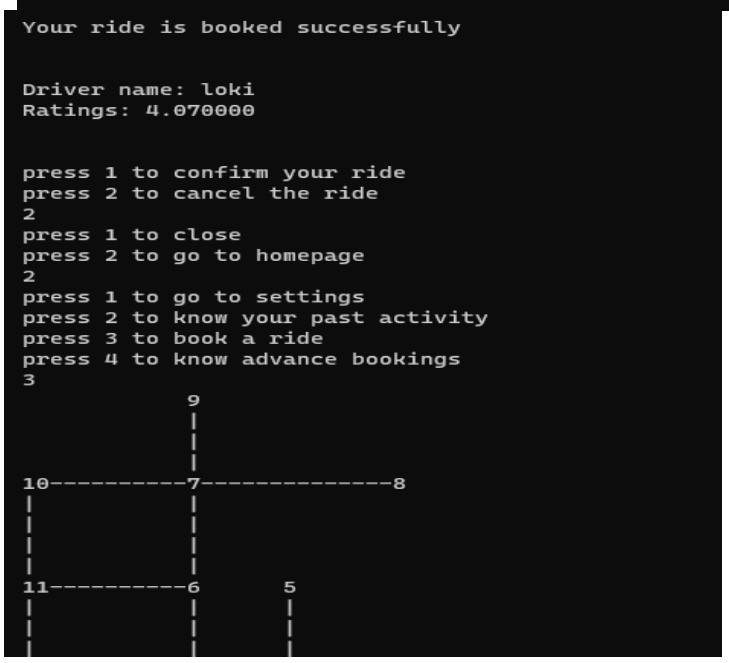
Actual output is same as expected output

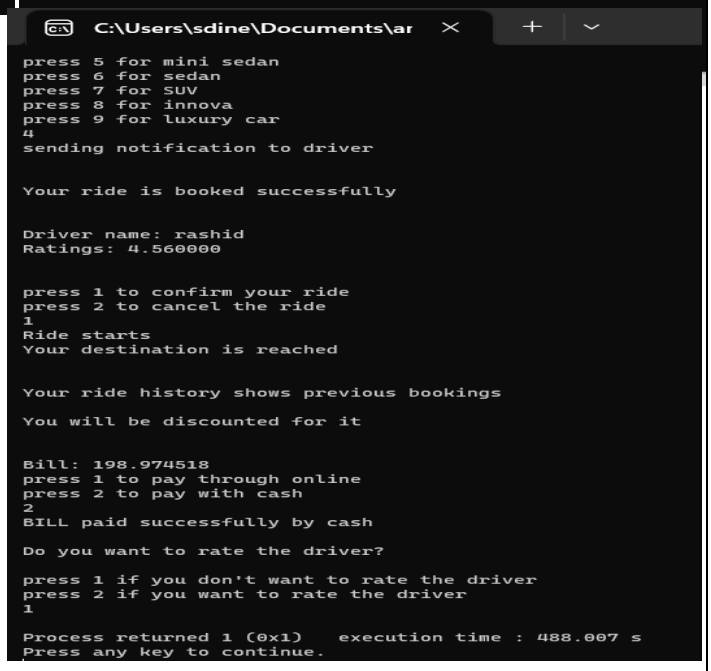
```
C:\Users\sdine\Documents\ar X + v

Distance between the points 1 and 12 is 30 Kms


press 1 to book in advance
press 2 for instant ride
1
enter the date(dd/mm/yyyy) of your ride:18/09/2023
enter the time(hh:mm) of your ride:17:18
Around which region you want the driver to pick you up(code)(leavingplacecode headingplacecode):10 11
how far you want the pick up from the point 10:2
enter the pick up point(leavingplace):chromepet
enter your destination(place):SSN
how far you want the drop from destination: 1
calculated distance 37.000000


surge

Car category and its cost of travel

Bike-225.000000
Auto-235.000000
Mini sedan-245.000000
Sedan-250.000000
SUV-260.000000
Innova-275.000000
Luxury car-290.000000
press 1 to continue booking
press 2 to go to homepage
1
select your mode of travel


press 3 for bike
press 4 for auto
press 5 for mini sedan
press 6 for sedan
press 7 for SUV
press 8 for innova
press 9 for luxury car
5
press 0 to confirm your booking
press 1 to go to homepage
1
press 1 to go to settings
press 2 to know your past activity
press 3 to book a ride
press 4 to know advance bookings
```

```
Your ride history shows previous bookings

You will be discounted for it


Bill: 320.509583
press 1 to pay through online
press 2 to pay with cash
2
BILL paid successfully by cash

Do you want to rate the driver?

press 1 if you don't want to rate the driver
press 2 if you want to rate the driver
2
enter your rating for the driver(0-5):4.98
new calculated rating for the driver 4.590000


Process returned 0 (0x0)   execution time : 247.348 s
Press any key to continue.
```

The instant booking is done without cancelling anywhere. Ride completed successfully . Actual output is same as expected output.







I have cancelled the order , from there I reached homepage. After that when I book, I'm charged extra for cancelling. I can also ignore giving ratings. Actual output is verified.

Limitations of the solution

- Limited availability of cab options in certain regions
- Increased demand may lead to longer wait times or unavailability of cabs
- Surge pricing during peak hours may result in higher fares for users
- It is impossible to use GPS facilities to locate customers and drivers.
- It is not possible to see the real time traffic.

Observations from the societal, legal and environmental perspectives

- Easy access to transportation, especially for those without personal vehicles
- Enhanced convenience for daily commuters and travelers
- Verified drivers and background checks ensure passenger safety
- Real-time tracking and monitoring systems provide a sense of security
- Reliable and professional service reduces the risk of incidents or accidents
- Job creation and income generation for drivers

LEARNING OUTCOMES

- We got to know about how cab booking apps like uber, ola and rapido works.
- We learnt the basics of c like array, structure, pointers, files, functions etc.
- we explored a lot about different diagrams like use case, sequence, activity, data flow diagram etc.
- we had to work with MS powerpoint and MS word. We got to know how to use those.
- This project has increased our logical and analytical thinking immensely.
- Since we worked as a team, we learnt how to work efficiently as a team.
- We learnt how to integrate structures,files,functions efficiently and get a proper output.
- The project has improved our algorithmic and problem solving skill.
- We got lot of errors while working on the project,we tried to find it and solve.(debugging & testing).
- We learnt how to document our code.