Software Requirements Specification

For

< CAR RENTAL SYSTEM AND ROUTE SUGGESTION USING DIJKSTRA'S ALGORITHM >

<16-oct-2021>

Prepared by :

Specialization	SAP ID	Name
Artificial Intelligence and Machine Learning	500075151	Aradhya Prakash Mahant
Artificial Intelligence and Machine Learning	500075342	Arjun Kapoor
Artificial Intelligence and Machine Learning	500075266	Harsh Kukretee
Artificial Intelligence and Machine Learning	500075951	Arun Gera

UPES
UNIVERSITY WITH A PURPOSE

Department of Informatics
School Of Computer Science
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,
DEHRADUN- 248007. Uttarakhand

Table of Contents

1.INTRODUCTION

1.1 Purpose of the Project:

The purpose of this project is to provide a smart and efficient way of renting cars and providing the most appropriate route to the user with navigation system. During these hard times of covid when you cannot risk your life by going in a public transport such as bus, taxi or auto rickshaws you rely only on personal transport but the people who don't have a personal vehicle really suffer a lot so for them this system will be very useful. Helping the person in need is the main purpose of our project.

1.2 Target Beneficiary :

In this Project, the target groups are

- Those people who don't have personal vehicles and want to go places so that they can explore their country(India).
- One who is lazy and wants to book cars by the comfort of his home.
- Common people who doesn't know any car rental service provider can use our app.

1.3 Project Scope:

Through the implementation of this project, we aim to assist in the ease of understandability of various algorithms and C++ concepts with the help of Dev C++ software. Here, in this project the user will be able to choose their ride and help themselves to navigate throughout their destination. The system would provide us with a main screen where we have to fill our login details to get access. The user has to provide the system with a password to make a successful login. Now, the system would provide the customer with the list of cars according to the availability including details like model, color, minimum advance fare to book the car , etc.

The user will input his initial location and a final location then the system will calculate rent for total kilometers and the path for the users destination. Finally, the customer will have the money he needs to pay and the route to his input destination.
Benefits :

- Better Efficiency than tradition car rental methodology
- Computer based invoice
- 2 in 1 app as it helps in renting car and suggesting routes.
- Contact-free system (beneficial for corona period)
- Transparent car renting approach
- Promotes tourism (state to state travelling)

Objective : During this covid-19 pandemic it was need of the hour to follow social distancing and avoid as much contact as possible. This has motivated us to make a Simple car rental system that will help the common masses. This project will act as an application and would help small rental companies and shops to grow their business.
Goal :With the approach of helping common people, we wanted to build this project and obviously our skills in C++ have made our work easy to fulfill our dreams by helping others.

1.4 References :

1. Wikipidia: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

2. Geeksforgeeks : https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/

3. https://www.projecttopics.org/design-and-implementation-of-a-car-rental-system.html

4. Let Us C++ a book by Yashavant Kanetkar.

2.PROJECT DESCRIPTION

2.1 reference algorithm :

The reference algorithm used for this project is Dijkstra's algorithm of path suggestion.Dijkstra's algorithm is the iterative algorithmic process to provide us with the shortest path from one specific starting node to all other nodes of a graph. It is different from the minimum spanning tree as the shortest distance among two vertices might not involve all the vertices of the graph. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later. In this project we have modified the Dijkstra's algo. So that it takes the source node and final node input from the user then suggestion the shortest path and giving the shortest distance as the output. The algorithm for this is :

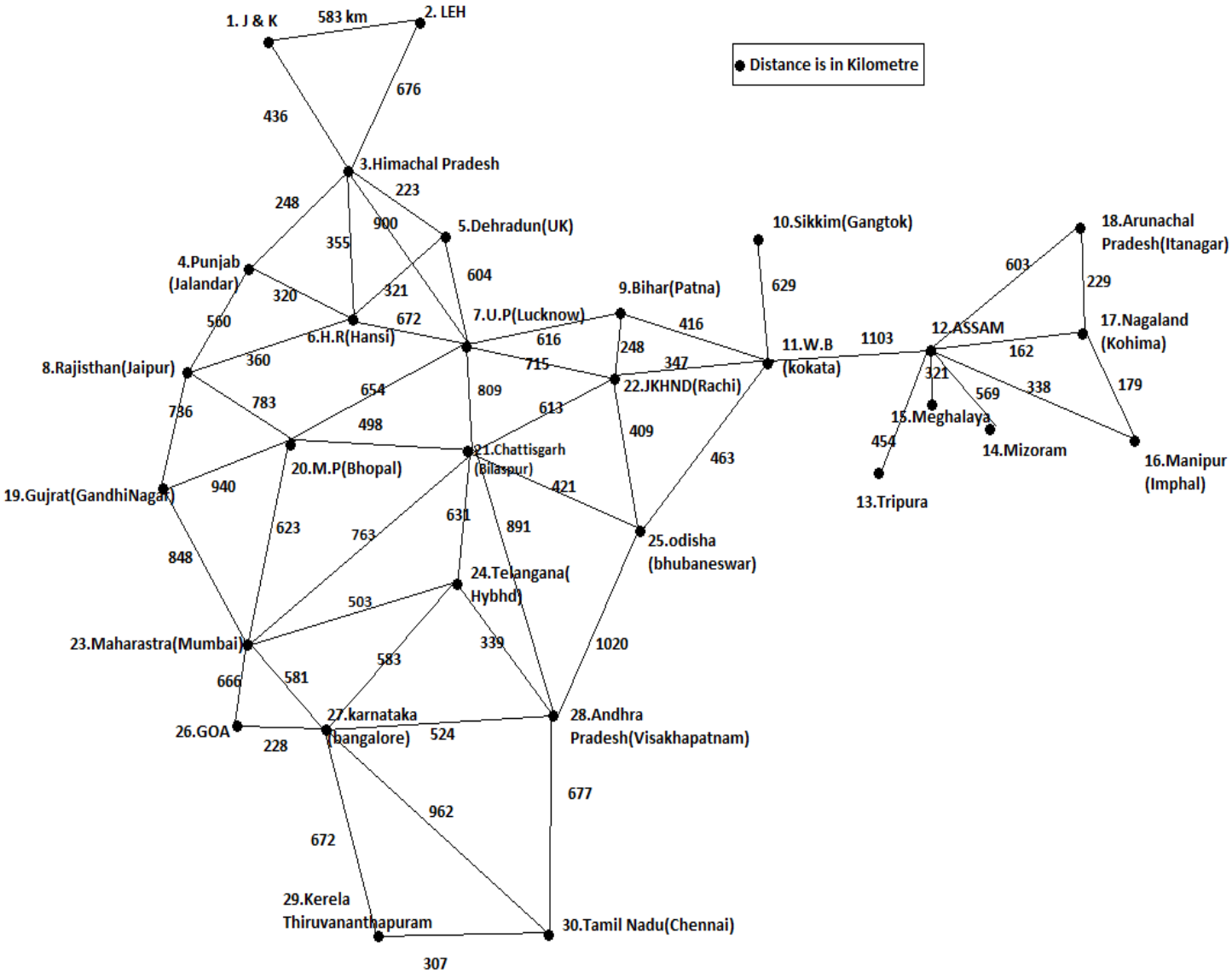
```
function Dijkstra(Graph, source):

    create vertex set Q
     for each vertex v in Graph:
                    dist[v] ← INFINITY
                     prev[v] ← UNDEFINED
                     add v to Q
       dist[source] ← 0
       while Q is not empty:
                    u ← vertex in Q with min dist[u]
                    remove u from Q
        for each neighbor v of u still in Q:
                    alt ← dist[u] + length(u, v)
                    if alt < dist[v]:
                                dist[v] ← alt
                                prev[v] ← u
     return dist[], prev[]
```

With the help of this algorithm we were able to calculate the minimum distance in between the states of India and suggesting a optimal route for travelling in our project car rental system and route suggestion using Dijkstra algorithm .Time Complexity : theta(|E| + |V|log|V|).

2.2 Characteristic of Data :

The dataset used in this project is a self created graph of the map of india with different states connected to each other by dots and lines which is further declared in the code so that it can be used for path suggestion . The dataset has accurate distances in between each state capitals as its is made with the help of google maps. No pre-processing was required in the data as it is a self made data or rather we should say graph .The given below is the graph used in the project for path suggestion :



2.3 SWOT Analysis:

Strengths	Weaknesses
• Designed for the comfort of users. • User friendly application. • 2 in 1 application as it helps in getting a self driven car as well as shortest route to the destination . • Largely scalable . • Free of cost. • Easy to use application.	• Success depends highly on user. • No marketing scheme is there so the spread of application only depends on suggestion of one user to other. • High workload and lack of work distribution. • Some bugs in the application.
Opportunities	**Threats**
• Local government wants to encourage local businesses. • Our competitors may be slow to adopt new technologies. • Small project like this acts as an inspiration for youth. • Product based market is always searching for user friendly apps like this.	• Further development of the application might be difficult for the users to adapt. • Availability of some of the closed source similar projects with access to a large scale of users. • Large competitors in the market.

2.4 Project Features :

The major features of our project are:-

- Login screen for the user to use the app.
- A list of available cars along with color, model , price per kilometre and the minimum advance money required to rent the car.
- 30 different state capitals to select one source destination and one final destination so that the minimum distance and path can be provided to the user.
- Finally a invoice is printed with amount due and also the path suggested.

2.5 User Classes and Characteristics :

Mainly users that are in search for a self driven car will use this app to book a car for themselves. This system will give them options with different cars so that they can book a car of their choice . People who want to travel and explore their country but do not own a car will be very happy to use our simple and easy system of renting cars.

2.6 Design and Implementation Constraints :

In this Project, the basic requirements for project to be successfully executed in users machines are -
Hardware:
Windows 7 - Processor - Intel Core 2 Duo E7500 2.93 GHz / RAM - 4GB
HDD - 20GB(free space)
MacOSX - Processor - 2.7GHz dual-core Intel Core i5 processor / RAM - 4GB
SSD - 10GB(free space)

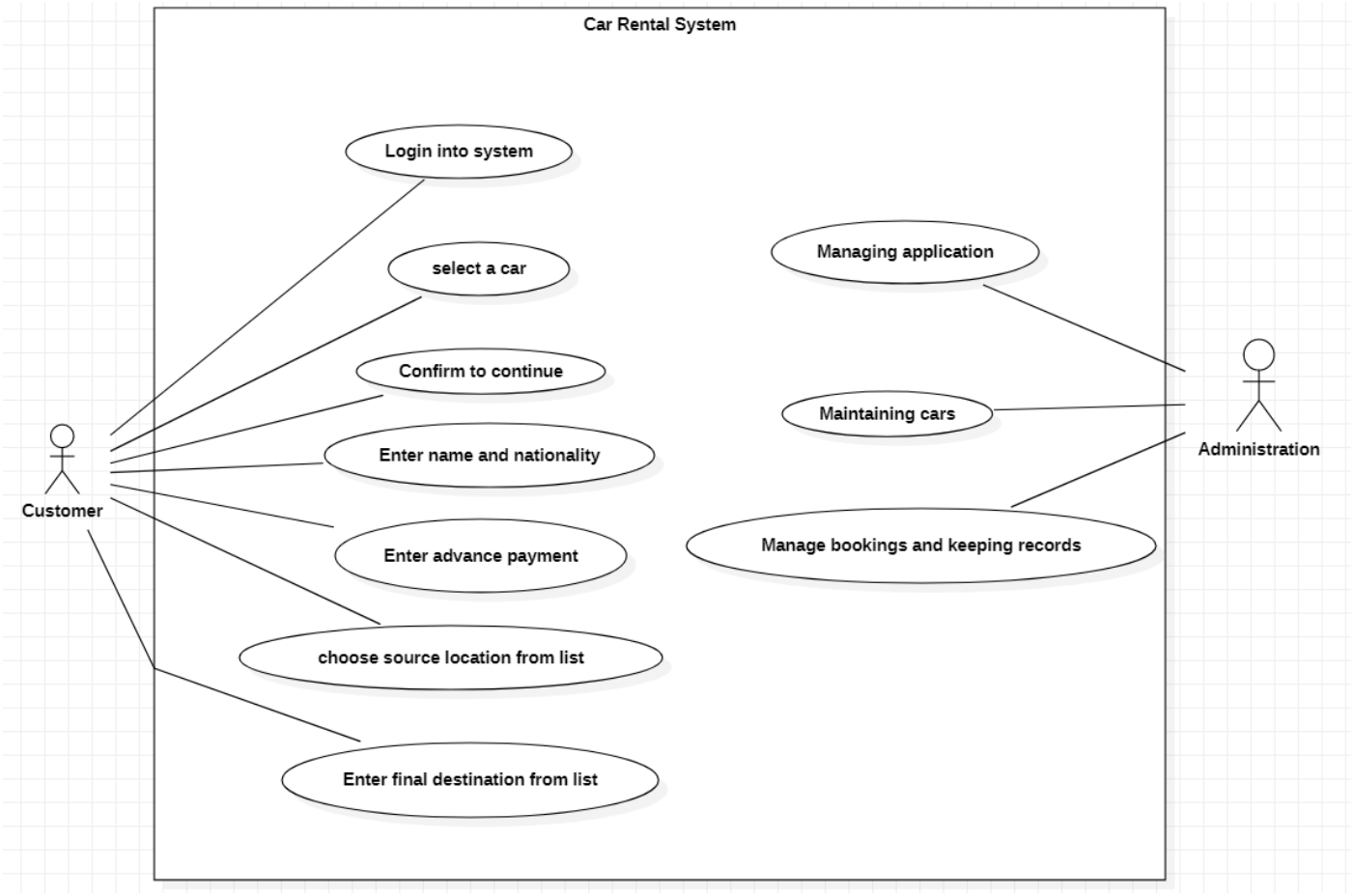
The execution of this project will be different for windows and mac users as both will be using different compilers :

GCC – for windows users
Xcode - This is a compiler for mac users.

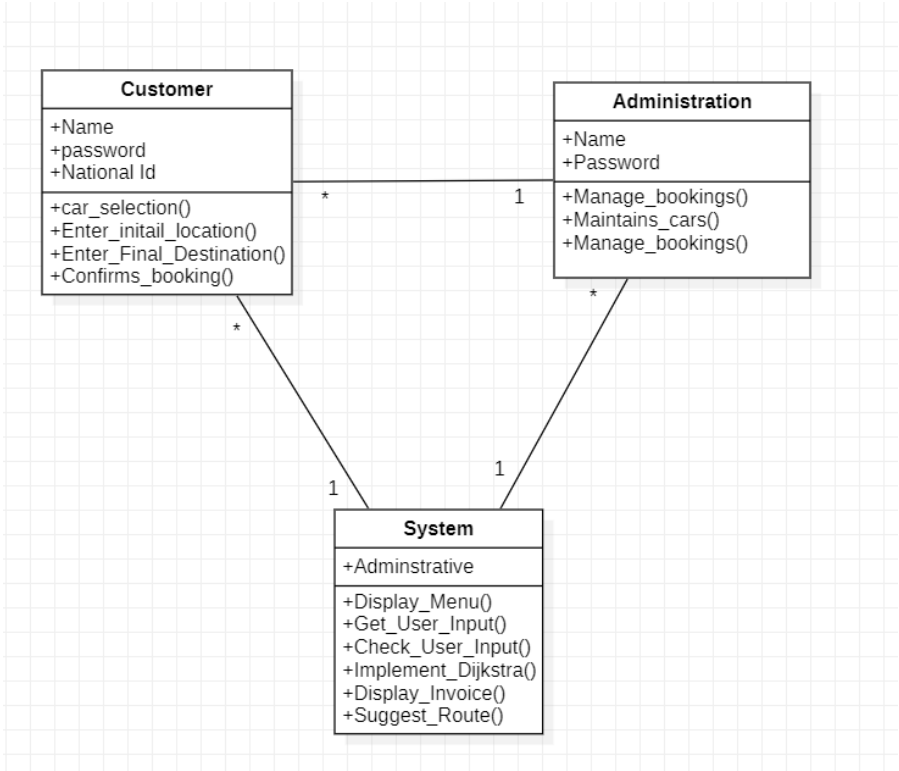
Our source code is written in C++ language so the users have to install a C++ IDE in their system to run our code without any worries . We suggest that windows users should use Dev C++ and Mac users should be Xcode IDE.

2.7 DESIGN DIAGRAMS

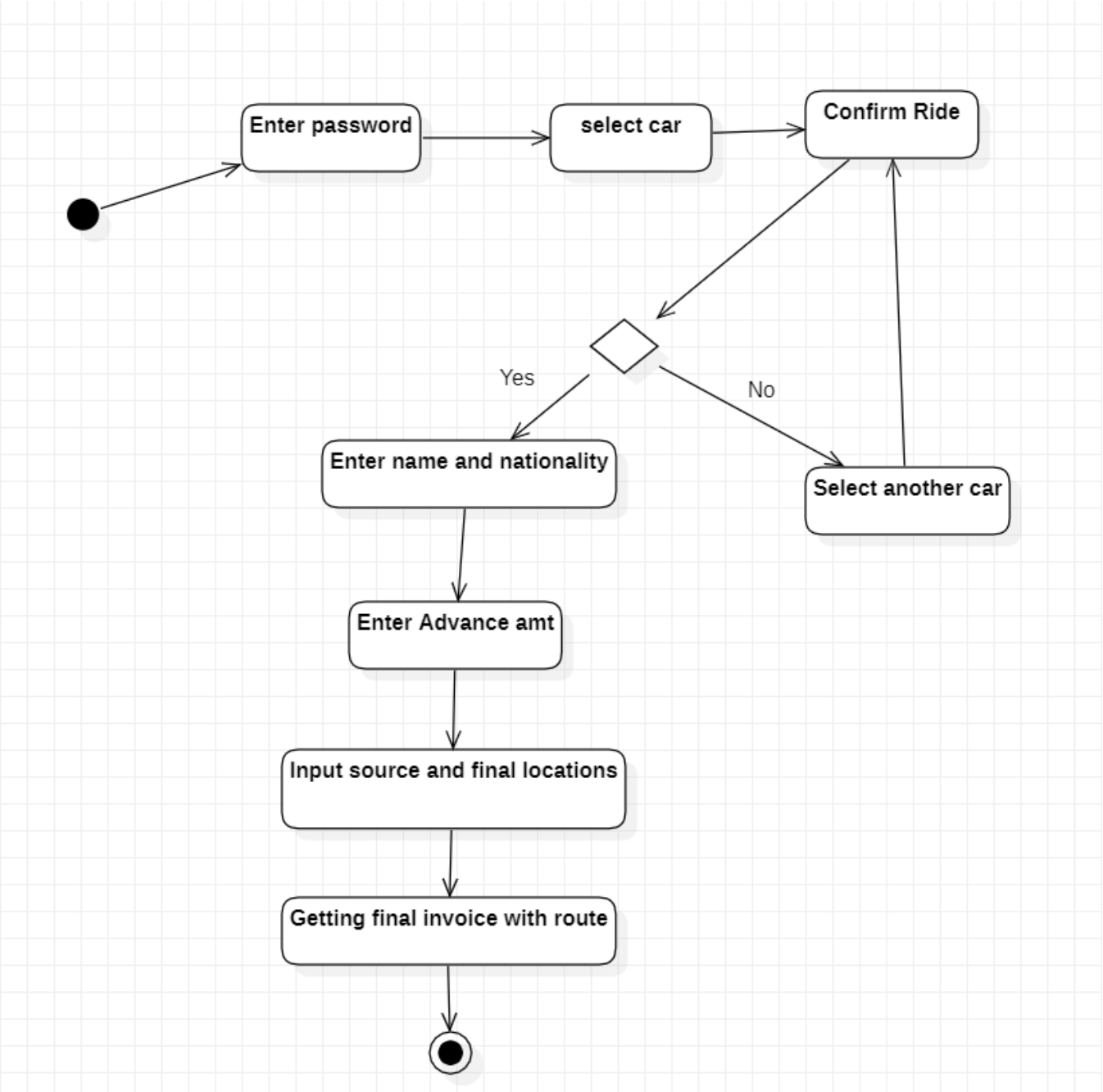
1. USE-CASE DIAGRAM :



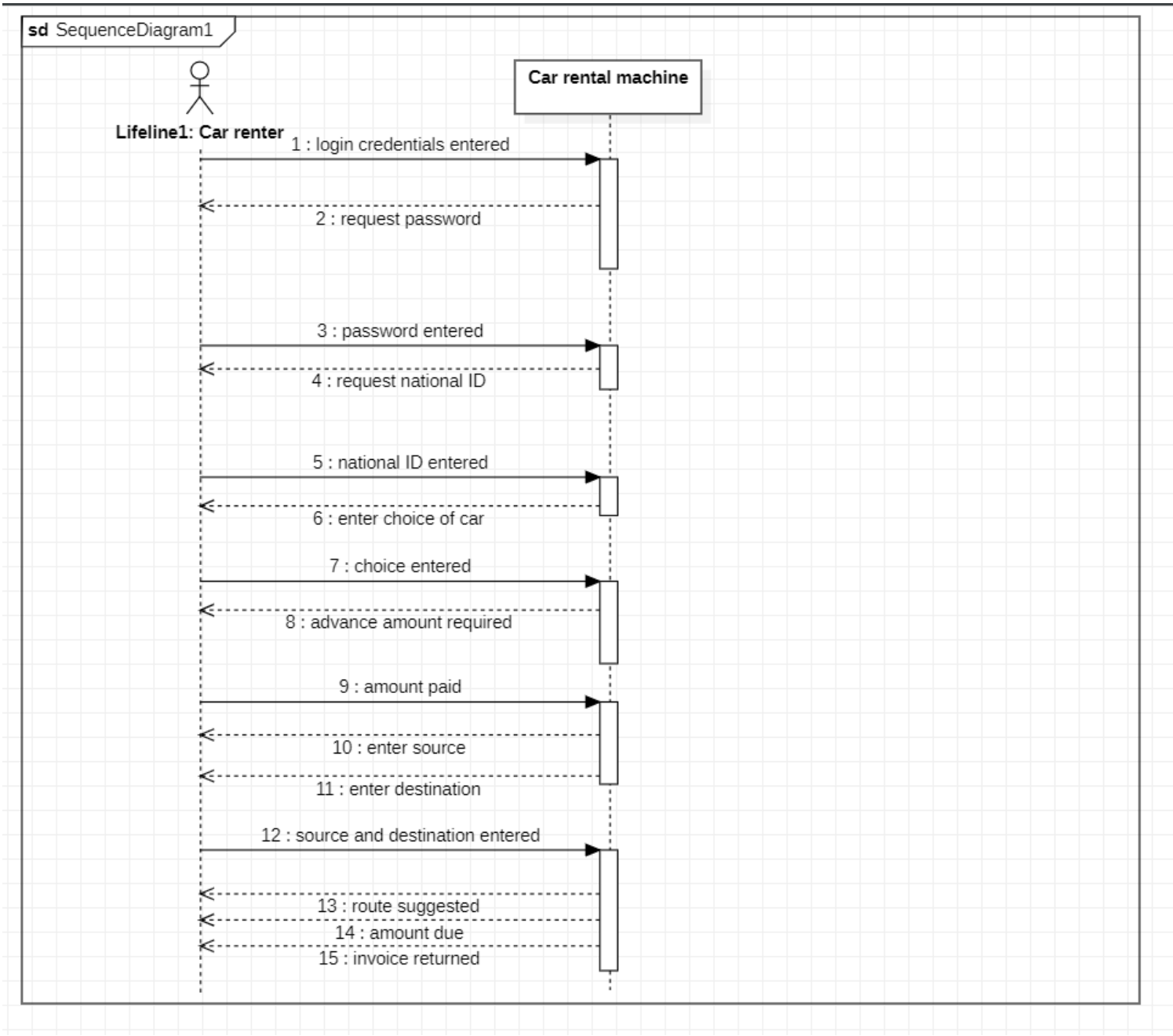
2. Class diagram :



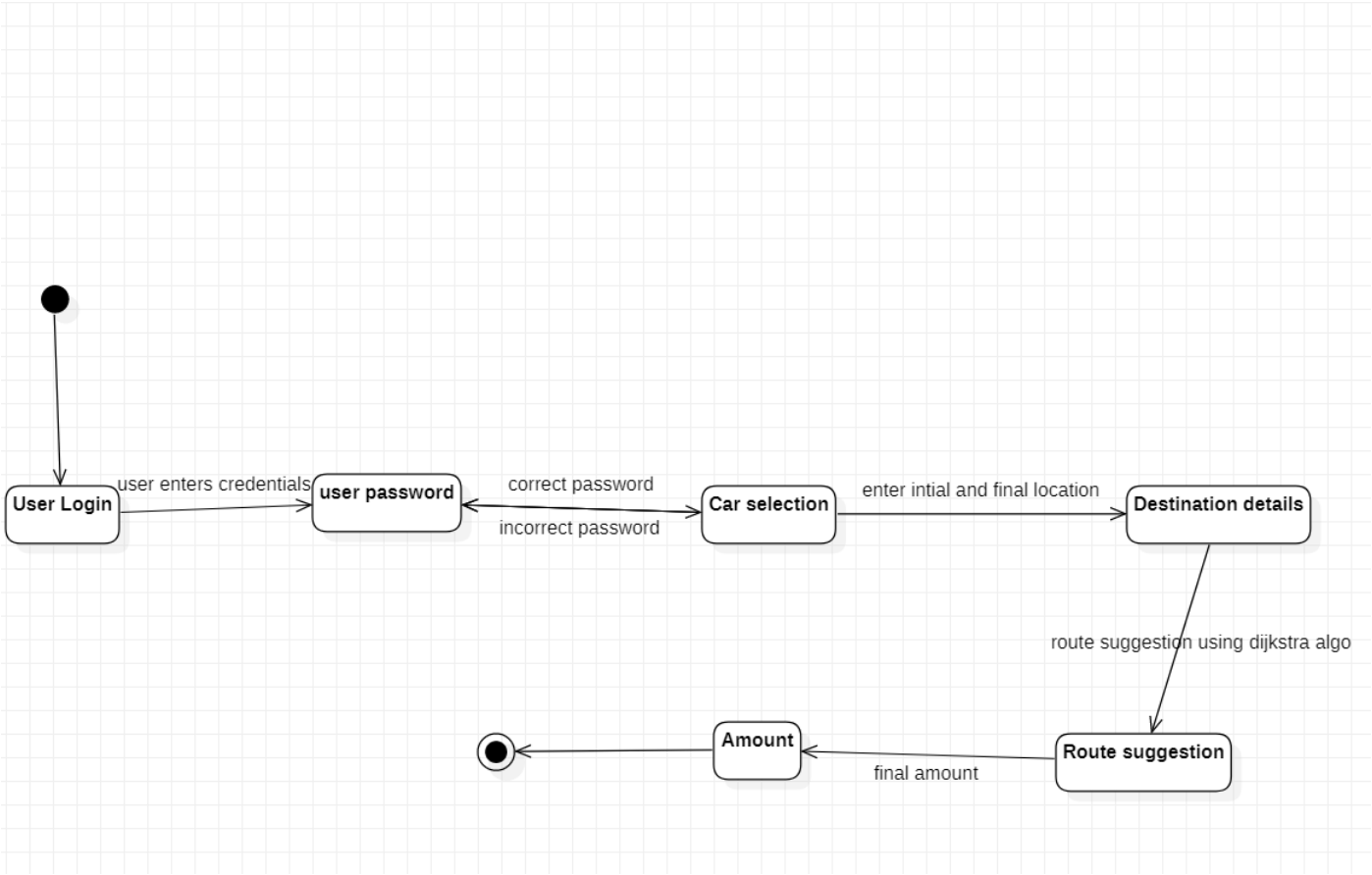
3.Activity diagram :



4.Sequence diagram :



5. State Diagram :



2.8 Assumption and Dependencies :

Assumptions made from our side are that we thought the users have a basic knowledge of how to use a computer any person who can not use a computer may not be able to take advantage of this system.
Other than that we thought that user will remember our index number generated path as our car rental system totally depends on the graph map of India as shown above.
Users should also know how to use command line for a better usage of our system.

3. SYSTEM REQUIREMENTS

3.1 <u>User Interface :</u>

The User Interface for project only requires a compiler.Dev C++, Turbo C++ are preferred.The project only requires user interface for selection of cars and other related details after which an invoice is printed.The project also requires user interaction for source and destination in order to suggest shortest route.

3.2 <u>Software Interface :</u>

The modules/library used for this following project include-

CONIO.h-This library It is a header file used in C and C++ and it includes inbuilt functions like getch() and clrscr().It stand for console input output i.e. it takes input from keyboard and displays it on screen.

STLIB.h-This library defines four variable types, several macros, and various functions for performing general functions.

3.3 <u>Database Interface</u> :

Our project contains self made dataset which will be stored and used in the form of 2- dimensional array and hence will be used for Dijkstra's algorithm.The dataset contains 900 elements in the form of weights associated with states and UT's. The dataset is the following one -

```
{ 0, 583, 436, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 583, 0, 676, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 436, 676, 0, 248, 223, 355, 900, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 248, 0, 9999, 320, 9999, 560, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 223, 9999, 0, 321, 604, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 355, 320, 321, 0, 672, 360, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 900, 9999, 604, 672, 0, 794, 616, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 654, 809, 715,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 560, 9999, 360, 794, 0, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 736, 783, 9999, 9999,
9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 616, 9999, 0, 9999, 416, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,
9999, 248, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
```

{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 0, 629, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 416, 629, 0, 110, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 347, 9999, 9999, 463, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 110, 0, 454, 569, 328, 338, 162, 603, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 454, 0, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 569, 9999, 0, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 328, 9999, 9999, 0, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 338, 9999, 9999, 9999, 0, 178, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 162, 9999, 9999, 9999, 178, 0, 229, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 603, 9999, 9999, 9999, 9999, 229, 0, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 736, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 0, 940, 9999, 9999, 848, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 654, 783, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 940, 0, 9999, 498, 623, 9999, 9999, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 809, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 498, 0, 614, 763, 630, 421, 9999, 9999, 891, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 715, 9999, 248, 9999, 347, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 614, 0, 9999, 9999, 409, 9999, 9999, 9999, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 848, 623, 763, 9999, 0, 503, 9999, 666, 581, 9999, 9999, 9999},
{ 9999, 631, 9999, 763, 0, 9999, 9999, 583, 339, 9999, 9999},
{ 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 463, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 421, 409, 9999, 9999, 0, 9999, 9999, 1020, 9999, 9999},
{ 9999, 666, 9999, 9999, 0, 228, 9999, 9999, 9999},
{ 9999, 581, 583, 9999, 228, 0, 524, 672, 692},
{ 9999, 891, 9999, 9999, 339, 1020, 9999, 524, 0, 9999, 677},
{ 9999, 672, 9999, 0, 307},
{ 9999, 677, 307, 0}

This dataset will be used for path recommendation as our project does not require the use of SQL, therefore no database management system was used.

3.4 **Protocols :**

As this project is based on Dijkstra's algorithm and rental system therefore no communication protocols were used in this project.

4. NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements, as the name suggests, are requirements that are not directly concerned with the specific services delivered by the system to its users. They may relate to emergent system properties such as reliability, response time, and store occupancy. Alternatively, they may define constraints on the system implementation such as the capabilities of I/O devices or the data representations used in interfaces with other systems. Non-functional requirements, such as performance, security, or availability, usually specify or constrain characteristics of the system as a whole.

4.1 Performance requirements:

The system response time for every instruction conducted by the user must not exceed more than a minimum of 10 seconds. The system should have high performance rate when executing user's input and should be able to provide response within a short time span usually 50 second for highly complicated task and 20 to 25 seconds for less complicated task.

4.2 Security requirements:

The system provides a password to prevent the system from unauthorized access. The staffs' password must be 4 characters. The subsystem should provide a high level of security and integrity of the data held by the system, just approved work force of the organization can access the organization's gotten page on the system; and just clients with substantial password and username can login to see client's page.

4.3 Software quality attributes:

Software Quality Attributes can be explained as the characteristics of the software application system that are kept in check, for meeting the needs of the software application to be eligible for being in good quality. Here, for the software quality standards, the term quality can be defined as a scale for distinction, for meeting the needs of the customer or end-user, for satisfying the principles of the application is expected to follow, to hold the user-friendly aspects, etc. These attributes are categorized to be the non-functional requirement specification, which needs to be met in order to make the software system quality exceptional.

Adaptability:

Adaptability can be defined as the case where a software system is assessed for interactive ability, and it adapts every action performed on the application to be matching with the individual user's needs.

Availability:

The system should always be available for access at 24 hours, 7 days a week. Also in the occurrence of any major system malfunctioning, the system should be available in 1 to 2 working days, so that business process is not severely affected.

Correctness:

The application should be correct in terms of its functionality, calculations used internally and the navigation should be correct. This means that the application should adhere to functional requirements.

Flexibility:

Should be flexible enough to modify. Adaptable to other products with which it needs interaction. Should be easy to interface with other standard 3rd party components.

Interoperability:
Interoperability of one system to another should be easy for the product to exchange data or services with other systems. Different system modules should work on different operating system platforms, different databases, and protocol conditions.

Maintainability:
Different versions of the product should be easy to maintain. For development, it should be easy to add code to the existing system, should be easy to upgrade for new features and new technologies from time to time. Maintenance should be cost-effective and easy. The system is easy to maintain and correct defects or make a change in the software.

Portability:
This can be measured in terms of Costing issues related to porting, Technical issues related to porting, and Behavioral issues related to porting.

Reliability:
Measure if the product is reliable enough to sustain in any condition. Should give the correct results consistently. Product reliability is measured in terms of working of the project under different working environments and different conditions.

Reusability:
Software reuse is a good cost-efficient and time-saving development method. Different code library classes should be generic enough to be easily used in different application modules. Divide the application into different modules so that modules can be reused across the application.

Robustness:
Robustness can be defined as the software quality assurance aspect, which is used for evaluating `the application’s capacity to handle the errors that occur during execution.

Testability:
The system should be easy to test and find defects. If required, it should be easy to divide into different modules for testing.

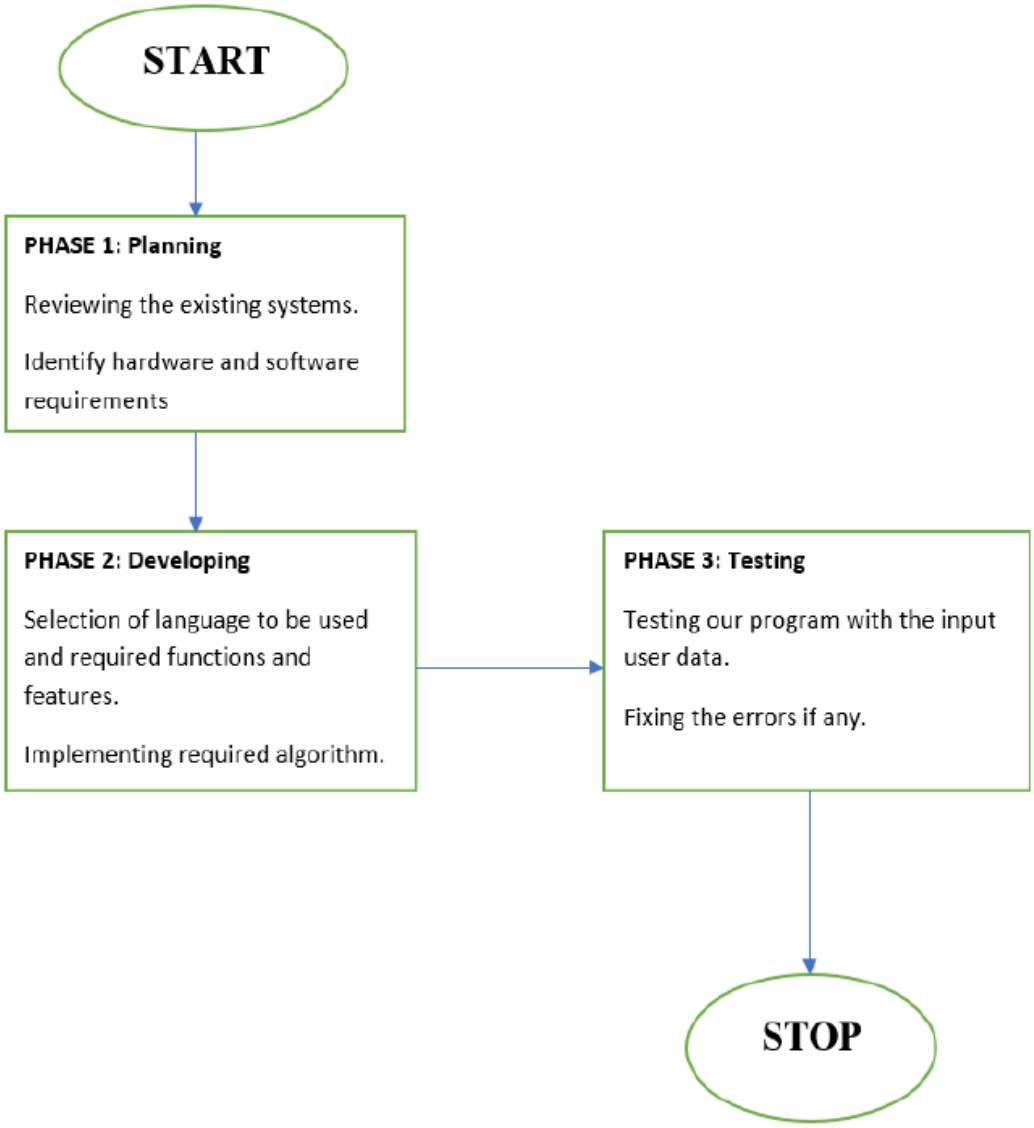
Usability:
The system provides a help and support menu in all interfaces for the user to interact with the system. The user can use the system by reading help and support.

OTHER REQUIREMENTS

A. Glossary –

There is no acronyms and abbreviations used in our project as our project is made in simple English language so we haven't used any acronyms and abbreviations as in such .

B: Analysis Model-



C: Issues List –

Revision History

Date	Change	Reason for Changes	Mentor Signature