Project report

CYBER SHARE

Less fare, more friends!

Team Name

Cyber captain

Team Members

Feng Lei

Han KaiYang

Xu BingHui

Team Number

Group 9

Content

1 Executive Summary

Due to the large number of students at NUS and the fact that the university dormitory is not available to all, students are required to take the bus or MRT or a taxi home after class. Taking the bus or MRT can result in long waiting times and the cost of taking a taxi alone can be expensive. Hence, the need for carpooling between students is generated.

Our project is dedicated to carpooling by grouping student passengers from similar destinations together based on a clustering algorithm and then using the Dijkstra algorithm to figure out the shortest path. As the colleges are relatively far away, but the NUS bus route basically covers all colleges and Utown, the default pick-up point for students was chosen as Utown. The drop-off location is freely chosen by the passengers, but system will match the existing carpooling points by calculating the relative distance, so that system can give the user a station closer to his/her destination.

2 Business background analysis

2.1 Business pain & value

2.1.1 Taxi fare analysis

Taxi fares in Singapore are generally expensive. For normal taxis, the flag down fee is from 3$ to 3.4$ for the first 1 km, then 0.22$ every 400m thereafter up to 10km or every 45 seconds of waiting.

For morning or night commuting peak time, there is a congestion charge of 25% total fee. Besides, when it comes to midnight, there is a congestion charge of 50% total fee. Besides, from 5pm to 12pm, a 3$ location surcharge is asked in city area. From the data of statistics office, the average taxi fee is about 12$ and the average MRT fee is 2.4$ after 6pm.

2.1.2 NUS passengers' analysis

First, there is an increasing number of bachelor and Masters students at NUS, the demand of commuting at night, especially 9pm, is growing extremely fast.

Secondly, for university students, the Singapore has an average daily expense from 20$ to 40$ every day, and if one student will come back after 9pm, he/ she must pay about 1/3 to1/2 of his daily expense to call taxi.

Lastly, most NUS students do self-study at Utown, which is pretty far from MRT station, and the night buses needs a long waiting time. Many experienced students find that carpooling at night has a similar cost to bus or MRT.

2.1.3 Typical industry analysis

For other similar company such as 'Grab', it is larger and has a longer history of 10 years. Besides, Grab cars can be seen easily on road. However, its taxi share service is suspended, and it has no exclusive positioning for NUS students.

2.2 Market positioning

The Cyber share software is facing to NUS students, providing them a commute with less fare, and help them to make more friends.

The car will shuttle through multiple sites. When people request carpooling, the app will offer reasonable matching between passengers and calculate the shortest path to a carpool station. Finally, students receiving concessions will recommend this app to their friends.

2.3 Development planning

2.3.1 Pricing

Since it is able to maximize 4 carpooling passengers and due to the previously calculated optimal routing based on the respective destinations, each of them can enjoy the best price.

2.3.2 Upgrade decision

For the carpooling passengers' information, dynamic carpooling can be achieved. For the carpool waiting time, users could choose their own waiting time. For the user request function, multiple users can request and complete carpooling. For the carpool model, users can choose the low and high car types.

2.3.3 Upgrade social media

It is possible to communicate with each other online for passengers who share a carpooling. For users, it is possible to add friends and make friends.

3 System design

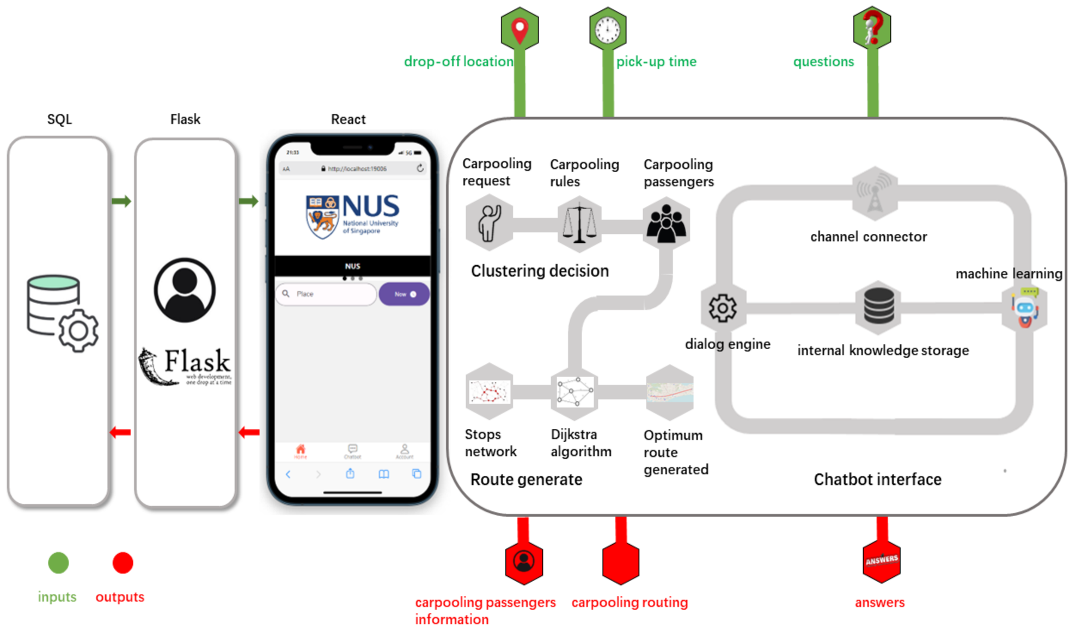


Figure 3-1: Carpooling system design

Firstly, from the Figure 3-1, it shown that the SQL database is set up to enable interoperability between SQL and the Flask and React frameworks. On the basis of the front and back-end work, three main functions of the system are completed.

3.1 clustering decision

The first is the clustering decision, using the clustering model and drop-off location, pick-up time input to generate rules to the next level and carpooling information to the whole system.

3.2 route generate

The second is route generate by unsupervised search method-the Dijkstra algorithm, which derives information about the start and end points from the clustering process and a map network of taxi to generates optimal route suggestions for the passenger.

3.3 Chatbot interface

The third is a chatbot, which provides answers to the user's questions and is implemented through a dialogue engine that combines an internal knowledge database with machine learning methods such as supervised learning.

Ultimately, the whole system can be completed with three main outputs: carpooling passenger information, carpooling routes and passenger question answers.

4 Decision

4.1 Processing Flowchart

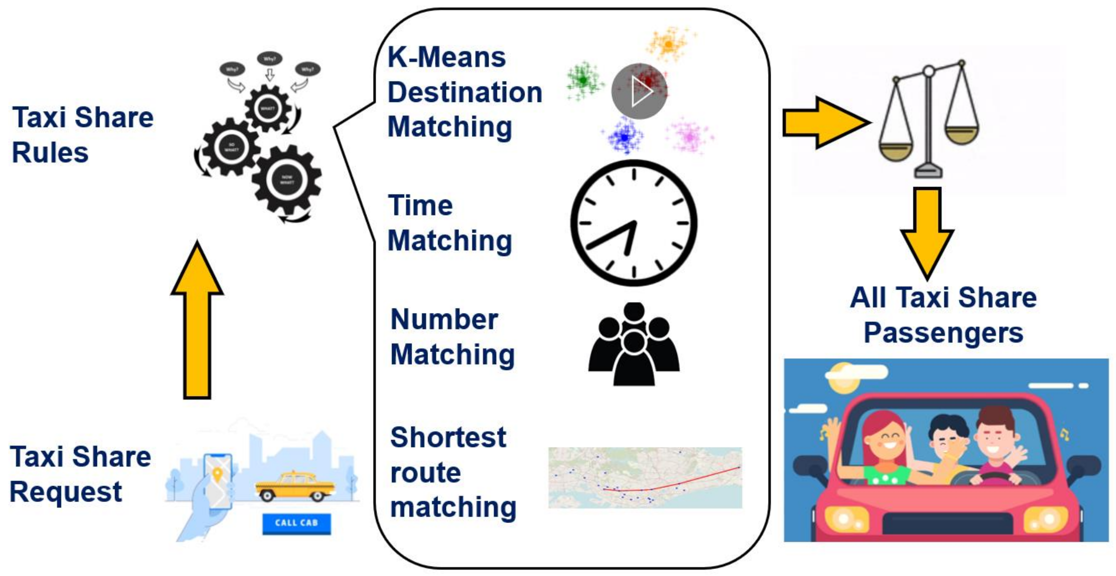


Figure 4-1: Processing Flowchart

4.2 Original Dataset

For the main purpose of this project is using operating vehicles to do carpooling, the drop off locations must follow local law. There are some rules which restricted the taxi and other operating vehicles to stop in certain areas. For example, taxi and other operating vehicles drivers are not allowed to pick-up or drop-off customers in Central Business District (CBD) areas between 7:00 and 22:00 to avoid influencing bus running. Besides, many roads such as Finlayson Green and Esplanade Drive are not allowed to stop by taxis and other operating vehicles.

In case the carpooling passengers can be sent to their home and ensure that their relatives can meet them at some fixed point, in this project, we use the taxi stops to be carpooling stops. There are many taxi stops in Singapore, which are set by the Singapore Land Transport Authority (LTA), and meant by the LTA to be the safest place to drop off passengers. So, in this project, we use the Singapore taxi stop dataset from OpenStreetMap Overpass API to get the taxi stops' names and locations. The shape of raw dataset after cleaning errors is shown as table 4-1. To some location with same names, the taxi stops' name will be appended with its unique location code by LTA.

Table 4-1: Part of taxi stops in Singapore

Latitude	Longitude	Name
1.286426	103.8014	Alexandra Hospital
1.275018	103.8434	Amara Hotel
1.369071	103.8482	Ang Mo Kio Hub
1.369827	103.8494	Ang Mo Kio Station
1.286878	103.8488	Archipelago Brewery
1.324684	103.8508	Balestier Hospital
1.316235	103.8358	Balmoral Plaza
1.342315	103.8805	Bartley Station
1.298414	103.8503	Bayview Hotel B05
1.324148	103.9295	Bedok Mall

4.3 Clustering

In this project, the people with similar routes can be taken in the same car. To make sure that the reasoning system is fast enough and easy to implement, the similarity of each route won't be calculated each time. Instead, the taxi stops which is mapped near each other and on similar directions will be divided in to groups in advance. In the reasoning process, if the user's destination in the same group as other users' destination in background, these users can be seen as potential carpooling passengers.

For the main features are longitudes and latitudes, there's limited labels in this dataset and can be seen as an unsupervised classification question. Clustering is a kind of unsupervised learning method to generate clusters of object dataset referring to its implicit structure. For it has no labels with the origin data, the number and standard of clustering result are unknown before training, so the quality of clustering results can be evaluated by data visualization.

From the from Singapore government website, taxi stops are set according to the passenger flow, which means where most taxi stops are densely distributed could have most drop off passengers. Thus, in this project, the clustering method is based on taxi stops distribution.

The clustering method can be divided into several categories, such as hierarchical clustering, partition clustering, density-based clustering and model-based clustering. Due to the actual circumstance, the clustering result wouldn't be too much in case that most people can find propriate carpoolers in a short time and share trip fares, the number of clusters should be set to a fixed number according to the distribution of taxi stops as figure 4-2. From figure 4-2, it can be seen that the taxi stops can be roughly separate to 5 directions from our start point Utown, respectively Joo Koon, Changi Airport, Woodlands, Harbourfront, and Punggol.

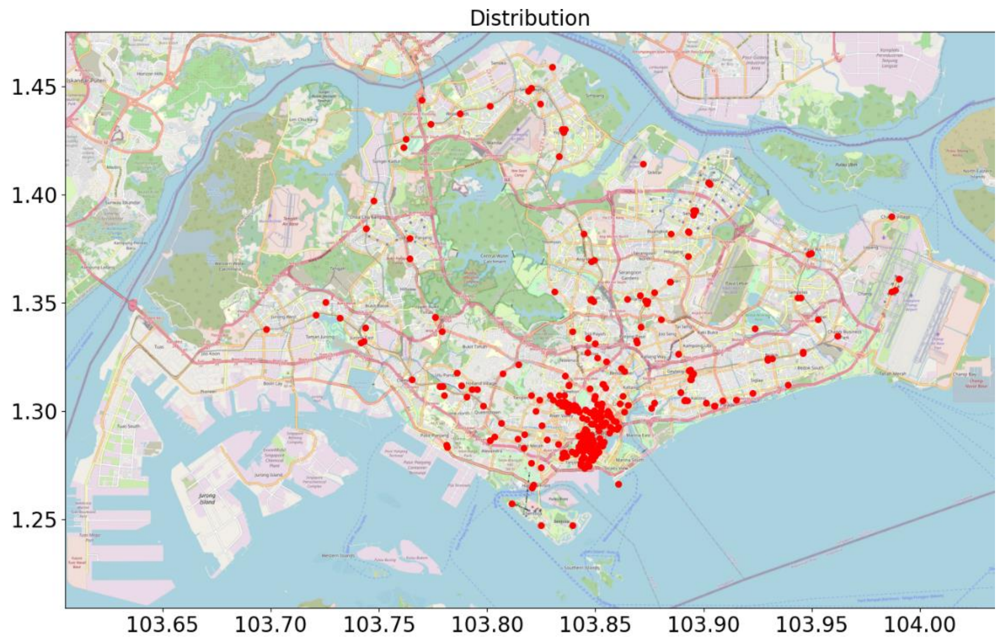


Figure 4-2: Taxi stops distributions in Singapore

Among all clustering methods, the hierarchical clustering, density-based clustering and model-based clustering could not input the cluster numbers and many hyperparameter will impact the cluster numbers, it's hard to achieve ideal clustering result with certain number and direction, and our pre-experiments have also proved this. Thus, the K-Means clustering is used.

The K-means algorithm is a top-down divisive clustering method, so it won't break old dividing border and enjoys a high efficient and simplicity. Thus, it is widely used in the practical application.

In this project, the function KMeans in sklearn.cluster library is used to implement the clustering algorithm. The function is as follow:

KMeans(n_clusters=a, random_state=b).fit_predict(dataset)

Where, ‘n_clusters’ stand for the number of clusters, ‘random_state’ stand for the random number generation for centroid initialization, which could better to be set as a constant when trying different cluster numbers. For most KMeans clustering result is round and won’t ignore outliers, some clusters need to be merged to fit the require of 5 big clusters. ‘Fit_predict()’ means to calculate the clustering centre and predict each function’s cluster attribute. After trying several parameters and change several point’s cluster according to practical directions, the clustering result by KMeans(n_clusters=10, random_state=2) is shown on figure 4-3.

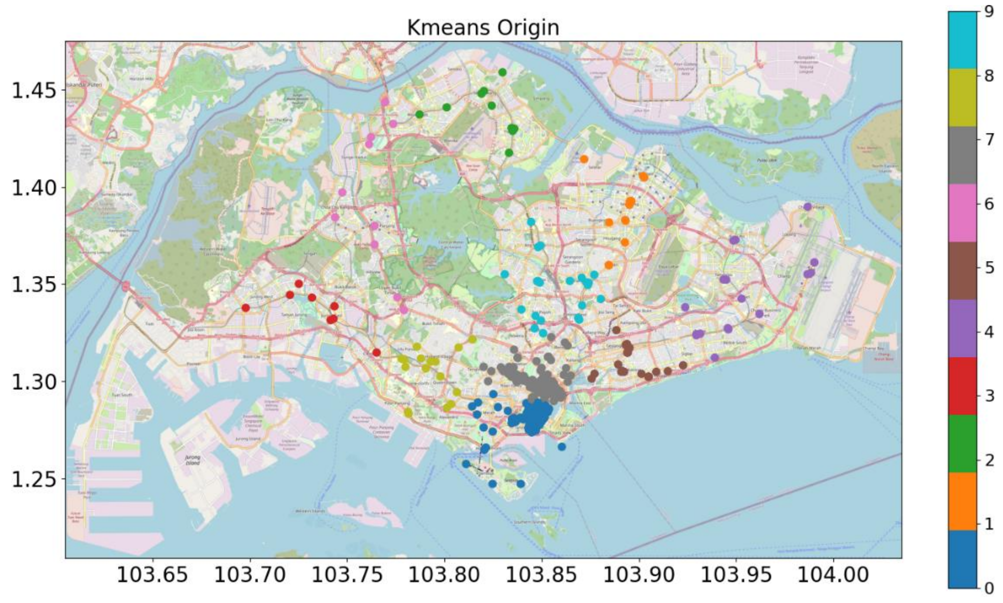


Figure 4-3: Clustering result by K-Means

From figure 2, all clusters are separately mapped on the desired direction, thus several clusters can be merged according to 5 directions, as is shown on figure 4-4. The clusters from 0 to 4 is separately stands the direction of Harbourfront, Punggol, Woodlands, Joo Koon, Changi Airport.

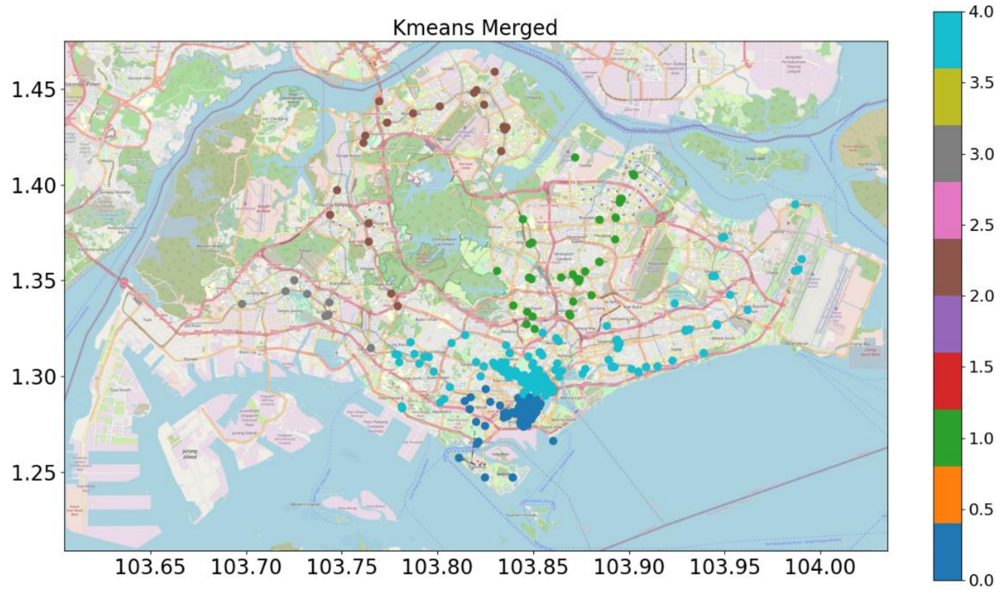


Figure 4-4: Clustering result after merging

4.4 Finding same route passengers

To realize the goal of sending passengers with the least cost and detours, some rules must be set in reasoning:

(1) The passenger number is from 1 to 4 and the more, the better;

(2) Each passenger could wait less than t minutes before they departure

(3) Each passenger's route will be assigned by the same cluster.

(4) Each passenger must be sent to their destination

(5) Each passenger's destination must be on the shortest route

Due to these limiting requirements, the rolling model with fixed window is used in the decisioning. Its working principle is as shown in Figure 4-5:

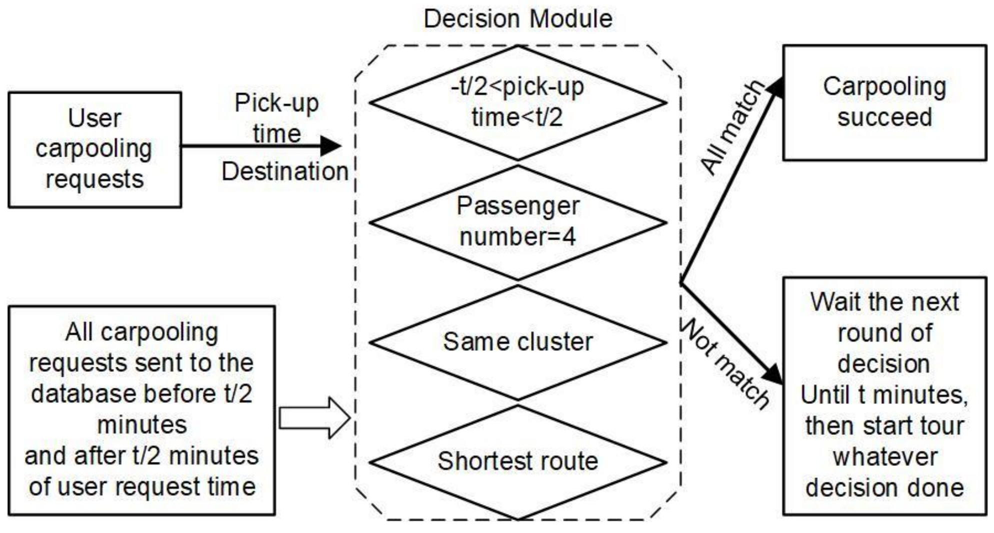

Decision Module
User carpooling requests
Pick-up time
Destination
-t/2<pick-up time<t/2
Passenger number=4
Same cluster
Shortest route
All carpooling requests sent to the database before t/2 minutes and after t/2 minutes of user request time
All match
Not match
Carpooling succeed
Wait the next round of decision Until t minutes, then start tour whatever decision done

Figure 4-5: Carpooling decision

4.5 Local virtual dataset

For the system has a requirement of running in local environment, the all carpooling requests need to be set to stimulate background users. For the user request has pick-up time and destination, the people can be generated by the time distribution from 5 p.m. to 12 p.m., which is a reasonable time for NUS students to come back home, and their destinations will be assigned from all taxi stops by ‘numpy.random’ function.

5 Route generate-Dijkstra algorithm

5.1 Processing Flowchart

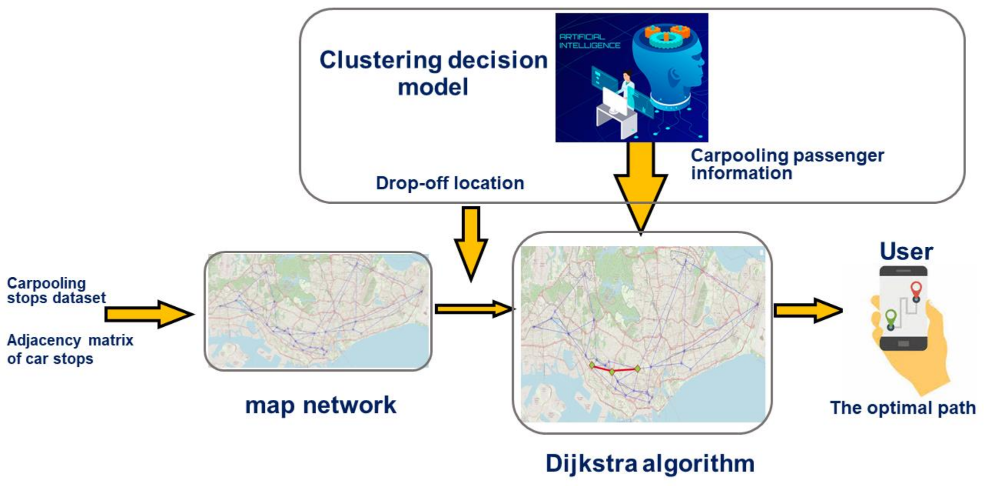


Figure 5-1: Processing Flowchart

5.2 Related Dataset

In this part, this app will give passengers advises about the shortest path, so, they can decide whether is this a worthy travel for them. And at this part, we choose some stops to represent the all stops, and these stops have the same distribution as all stops. The shape of stops dataset is shown as table 5-1.

Table 5-1: Part of car stops for route generate

node	id	Latitude	Longitude	Name
0	2.5E+09	1.30616	103.774	University town
1	1.6E+09	1.31473	103.765	Clementi Station
2	3.8E+09	1.33216	103.743	Jurong East Station
3	3.8E+09	1.33776	103.698	Pioneer Station
4	5.3E+09	1.34441	103.721	Lakeside Station
5	3.8E+09	1.35013	103.725	Blk 492 J10
6	3.8E+09	1.31131	103.779	Dover Station J06

7	3.7E+09	1.30662	103.791	Buona Vista Station
8	3.8E+09	1.31008	103.795	Holland Road Shopping Centre
9	3.8E+09	1.34323	103.776	Bukit Timah Shopping Centre

Additionally, using the uninformed search method-Dijkstra algorithm to find a short path need an Adjacency matrix, the shape of this matrix is shown as table 5-2.

Table 5-2: Part of Adjacency matrix of car stops

Node	0	1	2	3	4	5	6	7	8	9
0	0	1.36	inf	inf	inf	inf	0.82	1.84	inf	inf
1	1.36	0	3.13	inf	inf	inf	inf	inf	inf	3.37
2	inf	3.13	0	5.06	2.84	2.81	inf	inf	inf	3.85
3	inf	inf	5.06	0	2.63	inf	inf	inf	inf	inf
4	inf	inf	2.84	2.63	0	0.81	inf	inf	inf	inf
5	inf	inf	2.81	inf	0.81	0	inf	inf	inf	inf
6	0.82	inf	inf	inf	inf	inf	0	1.35	inf	3.55
7	1.84	inf	inf	inf	inf	inf	1.35	0	0.67	inf
8	inf	inf	inf	inf	inf	inf	inf	0.67	0	inf
9	inf	3.37	3.85	inf	inf	inf	3.55	inf	inf	0

In this table, the value of 0 raw and 1 column represents the distance from node 0 to node 1, it means the distance from University town to Clementi Station. And these distances are calculated by the geodesic distance, which can be obtained by import ‘geopy’ in python. The geodesic uses the currently internationally accepted method of representing the Earth on a rotating ellipsoid, which calculates the shortest distance between two points on the ellipsoid.

5.3 Building a map network of carpooling system

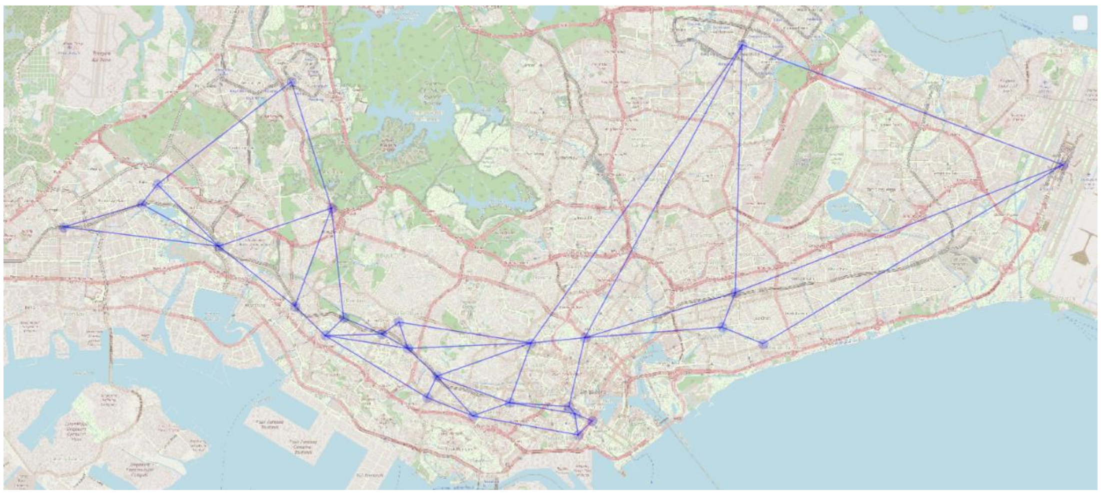


Figure 5-2: map network of carpooling system

Using the previous datasets, the matplotlib can draw the map Figure 5-1, this map mainly represent the operation route of cars in carpooling system.it is to simulate the real route after a car has picked passengers up. And in this map, due to the reason that generated routes are not clear when we put too much stop points on this map, it just picks 27 stops to run simulated. But the final network in the map can cover the main distribution of Singapore.

5.4 Uninformed search-Dijkstra algorithm

For better explaining how a shortest path are generated by Dijkstra algorithm. There will be an example network, it forbids the car to drive out the network consisted by these stops, which are shown in Table 2-3 and Figure 2-2. Meanwhile, we will set the start point as University town, and the end point as Lucky Plaza. Most importantly, in the original network, the process will have different updating Number of times and process even their start stops, end stop and principle are same.

Table 5-3: Example network's nodes

Node	Place
0	University town
6	Dover Station J06
7	Buona Vista Station
8	Holland Road Shopping Centre
11	Commonwealth Station
14	Lucky Plaza

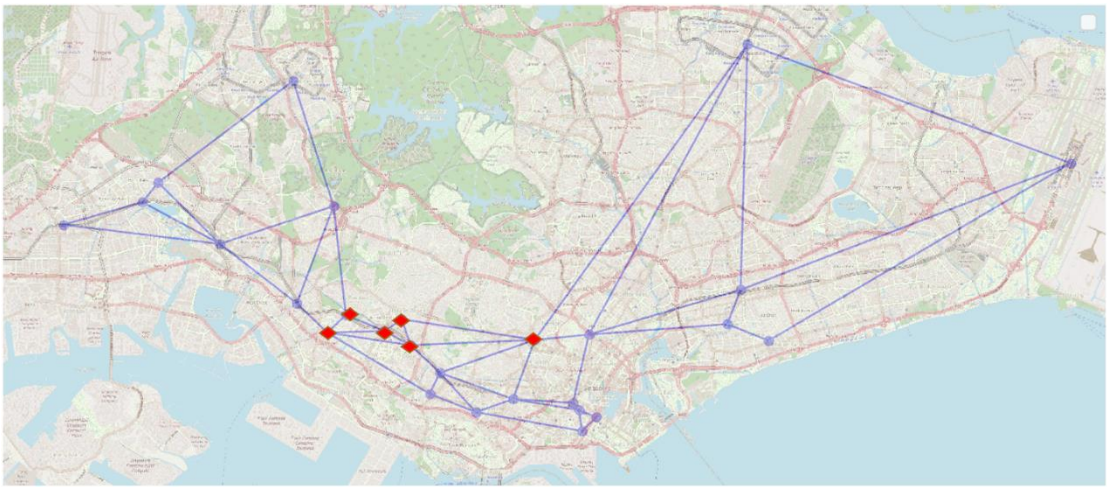


Figure 5-3: map network of carpooling system

Table 5-4: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
S	{ }					

In Table 5-4,5-5,5-6,5-7,5-8,5-9,5-10, the Q represents stops in example network and the S represents the list that the shortest path will pass. And the numerical values inside the table represents the cost from start point to the related stop. And Bolded, angled single figure represent the stop which have been completed, Correspondingly, it will be shown in Figure 5-3,5-4,5-5,5-6,5-7.

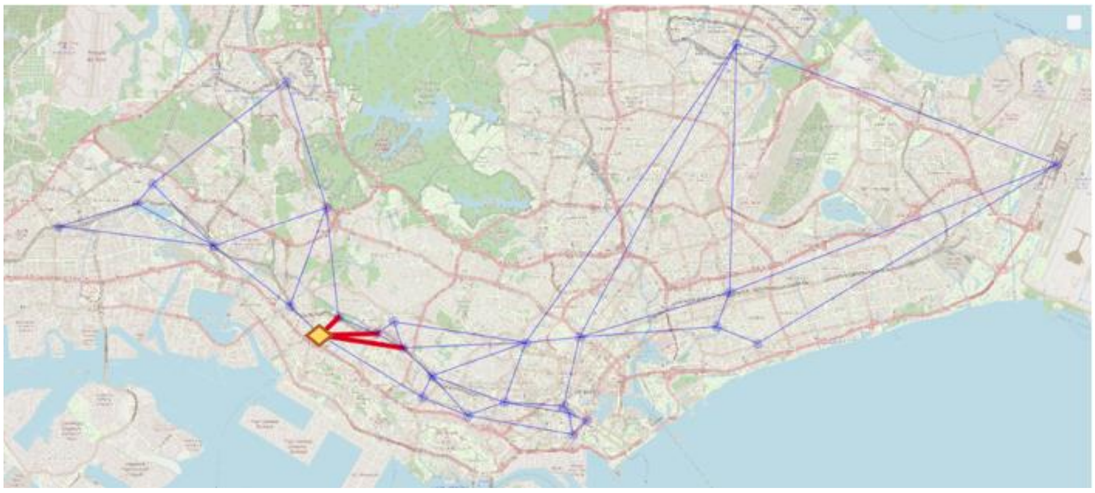


Figure 5-4: map network of carpooling system

Table 5-5: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	1.84		2.7	
S	{0}					

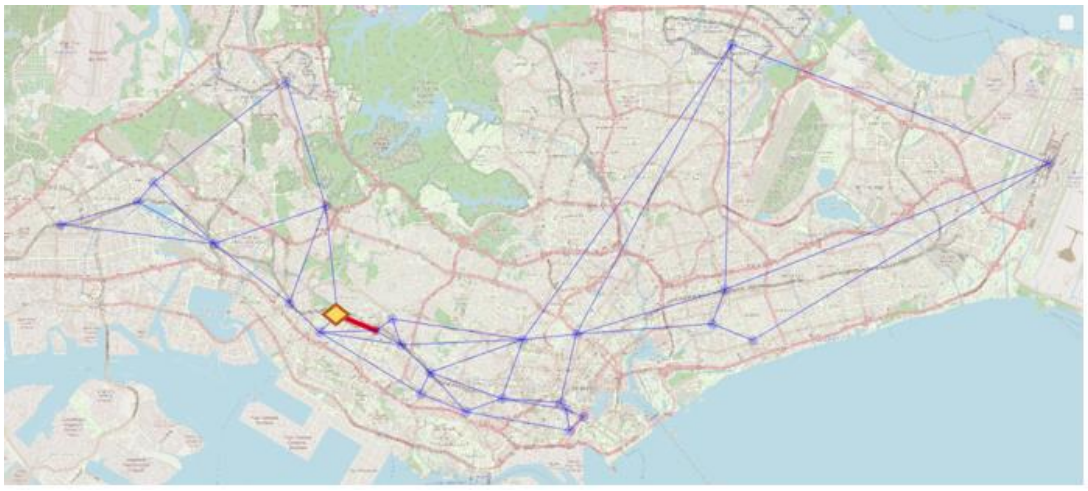


Figure 5-5: map network of carpooling system

Table 5-6: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	1.84		2.7	
S	{0}					

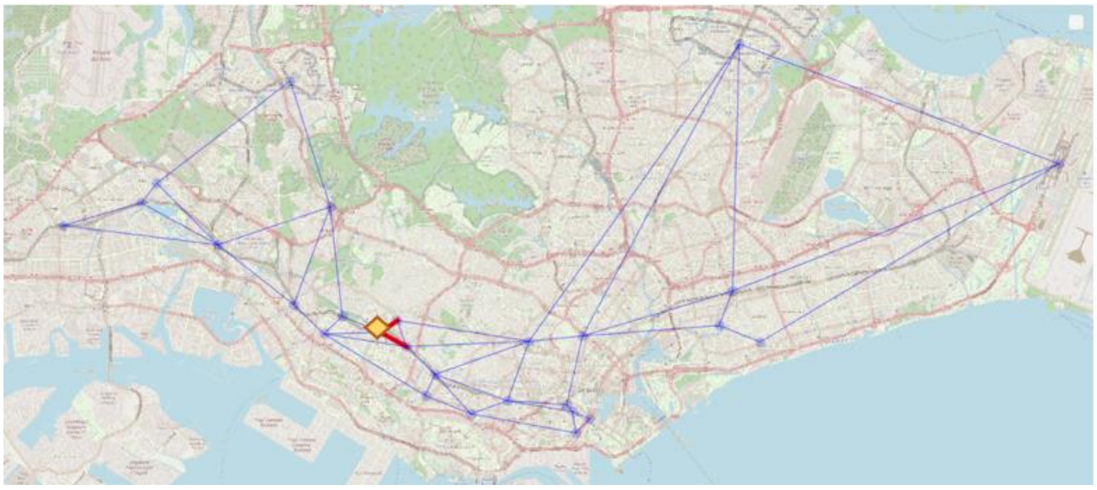


Figure 5-6: map network of carpooling system

Table 5-7: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	***1.84***		2.7	
				2.51		
S	{0}					

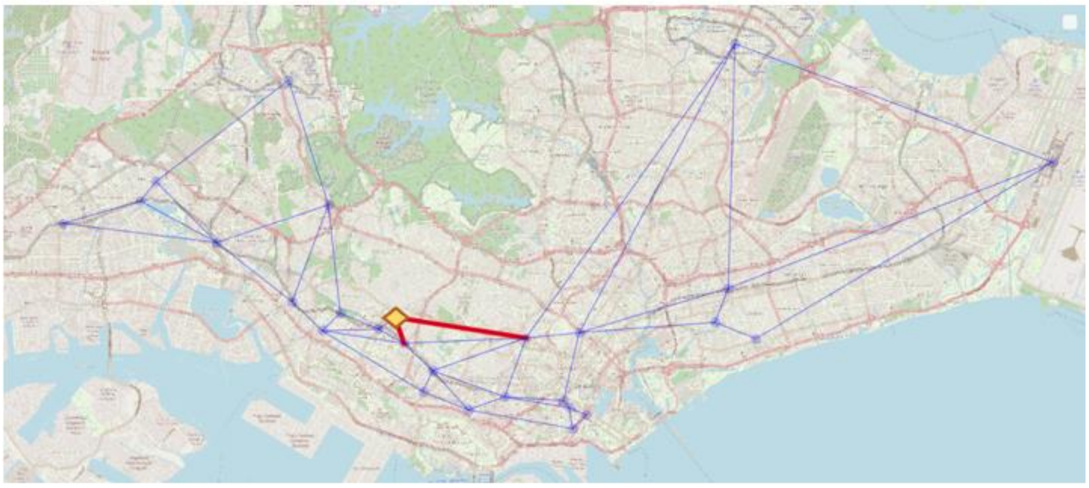


Figure 5-7: map network of carpooling system

Table 5-8: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	1.84		2.7	
				2.51		
						6.84
S	{0}					

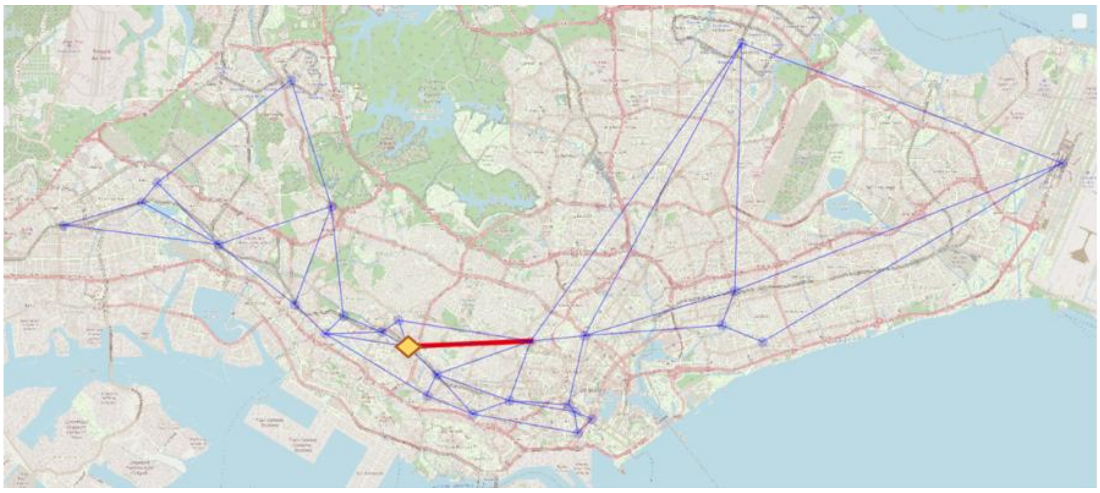


Figure 5-8: map network of carpooling system

Table 5-9: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	1.84		***2.7***	
				2.51		
						6.84
						6.69
S	{0→ 11}					

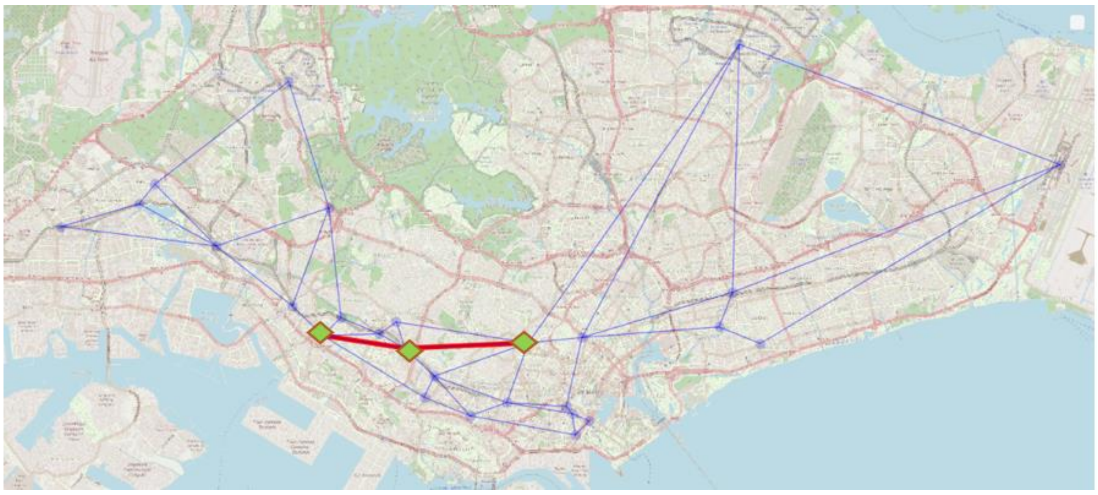


Figure 5-9: map network of carpooling system

Table 5-10: Cost and Shortest path

Q	0	6	7	8	11	14
	0	∞	∞	∞	∞	∞
		0.82	1.84		2.7	
				2.51		
						6.84
						6.69
S	{0 → 11 → 14}					

The Figure 5-8 represents the shortest path from University town to Lucky Plaza. And the Figure 5-8 represents the shortest distance from University town to Lucky Plaza.

5.5 Summary

For this part's algorithm only, it have Implemented a shortest search from random start stop to random end stop in the original map network

6 CHATBOT DESIGN

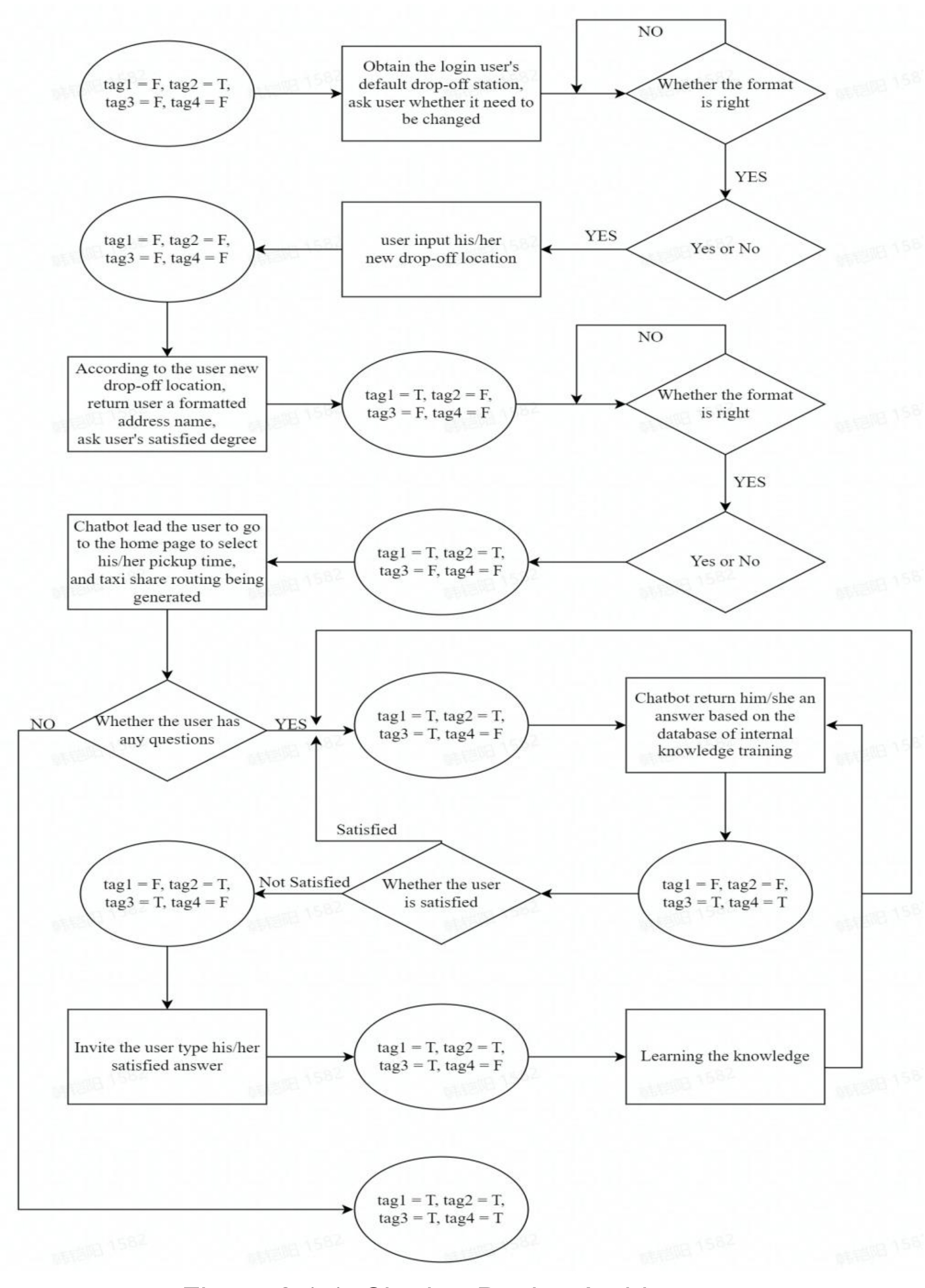
6.1 Overall Design Idea

The design flowchart is showing below. Chatbot named ‘Captain’ has three functions:

(1) Obtain the user drop-off location.

(2) Answer the user’s questions.

(3) Learn the user’s answers.



tag1 = F, tag2 = T, tag3 = F, tag4 = F
Obtain the login user's default drop-off station, ask user whether it need to be changed
NO
Whether the format is right
YES
Yes or No
YES
user input his/her new drop-off location
tag1 = F, tag2 = F, tag3 = F, tag4 = F
According to the user new drop-off location, return user a formatted address name, ask user's satisfied degree
tag1 = T, tag2 = F, tag3 = F, tag4 = F
NO
Whether the format is right
YES
Yes or No
tag1 = T, tag2 = T, tag3 = F, tag4 = F
Chatbot lead the user to go to the home page to select his/her pickup time, and taxi share routing being generated
NO
Whether the user has any questions
YES
tag1 = T, tag2 = T, tag3 = T, tag4 = F
Chatbot return him/she an answer based on the database of internal knowledge training
Satisfied
Not Satisfied
Whether the user is satisfied
tag1 = F, tag2 = F, tag3 = T, tag4 = T
tag1 = F, tag2 = T, tag3 = T, tag4 = F
Invite the user type his/her satisfied answer
tag1 = T, tag2 = T, tag3 = T, tag4 = F
Learning the knowledge
tag1 = T, tag2 = T, tag3 = T, tag4 = T

Figure 6.1.1: Chatbot Design Architecture

6.2 Function 1 – Obtain the user's drop-off location

6.2.1 Function overall description

After login, chatbot (Captain) could obtain the user name and default drop-off location according to his/her register information and ask whether the user want to change the default drop-off location.

If the user does not want to change it, 'Captain' will obtain the user's drop-off location.

If the user wants to change it, 'Captain' will let the user type his/her approximate wanted drop-off location and 'Captain' will return the user a formatted address. If the user is not satisfied with it, 'Captain' will let the user re-input his/her drop-off location until the user is satisfied with the result. Hence, 'Captain' will obtain the user's drop-off location.

After obtaining the user's drop-off location, 'Captain' will lead the user go to the app home page and select the pick-up time. Then, the carpooling related message and the routing will display.

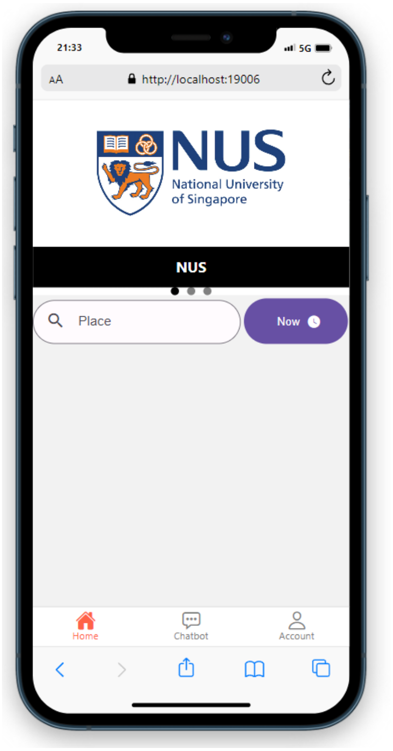


Figure 6.2.1: Home page

6.2.2 Formatted address technology

Use Google Maps Geocoding API and Google Places API Web Service to achieve this function based on 'googleplaces' and 'googlemap' in the python library.

Define the function 'obtain address recommendations', which returns a recommended address based on the address typed by the user. The function has three parameters, query, language and location. The language and location are given default values of English and Singapore. The formatted address is like this: '320 Clementi Ave 4, Block 320, Singapore 120320'.

6.3 Function 2 – Answer the user’s questions

6.3.1 Chatbot Q&A function description

When the user has obtained the carpooling message, the user could come back to the chatbot page and achieve questions & answers with ‘Captain’. Trigger condition is that user types anything, ‘Captain’ will capture whether the user wants to ask something. Then, chatbot chatting function is opening.

6.3.2 Rule-based and self-learning chatbot

Currently, there are currently two types of chatbot, rule-based and self-learning.

Rule-based chatbot will answer questions based on a number of rules that it has been trained to follow. The rules defined can be very simple or very complex. The chatbot could handle simple queries, but could not manage complex ones.

Self-learning chatbot use a number of machine learning based methods and tend to be more effective than rule-based chatbot. Both of them are further divided into the following two types, retrieval-based or generative.

In the retrieval-based model, chatbot uses a number of heuristics to select responses from a library of predefined responses. Chatbot uses message and conversation contexts to select the best response from a predefined list of chatbot messages. The context may include the current position in the dialogue tree, all previous messages in the conversation, previously saved variables. Heuristics for selecting responses could be designed in many different ways, ranging from rule-based if-else conditional logic to machine learning classifiers.

Generative chatbot could generate responses, but do not always answer with one of a set of answers. This makes them smarter, as they extract and generate answers word by word from queries.

In function 2, 'Captain' is a generative chatbot. If the user wants to ask questions, 'Captain' will answer it based on the database of the internal knowledge training.

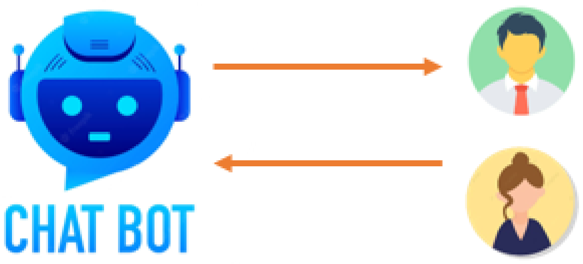


Figure 6.3.1: Chatbot & User interaction

6.3.3 Loading the training corpus

When training a chatbot, the training corpus stored in the database is loaded into memory segment by segment until it is fully loaded, thus preparing it for dialogue training.

6.3.4 Dialogue training

The key to chatbot ability to train chatbot that support "any question" is the training process. Instead of processing the training corpus deeply like word separation, the process converts the training corpus directly into statement, response pairs and stores them.

The training process for chatbot involves loading example dialogs into the chatbot's database. This allows the construction of graph data structures representing known sets of utterances and responses. When a chatbot trainer is provided with a dataset, it creates the necessary entries in the chatbot's knowledge graph to correctly represent the utterance inputs and responses.

A statement is constructed using contextual sentences, and the statement is equivalent to storing a relationship between the upper and lower conversations. When looking up, the most appropriate upper is found first, and the following is the answer. This is a training process, and this process of training is mainly in constructing the statement and putting it into storage. However, this piece has the disadvantage of low performance.

6.3.5 Storing training results

After completing the dialogue training, the training results need to be stored in a database so that when a chat request is received from a user the answer can be quickly found from the previous training results and fed back to the user.

6.3.6 Dialogue stage

The dialogue process is to find the answer statement from the previous training results and send it to the requesting party based on the current dialogue request.

If the best match strategy is used, when chatbot receives a dialogue request, it will iterate through all the training results and let it calculate the similarity with the current conversation request statement, and return the answer of the statement with the highest similarity to the conversation request as the answer to the current request statement. Hence, it could be seen that chatbot may take too long to answer a request when the training statement is too large.

6.4 Function 3 – Learn the user’s answers

6.4.1 Chatbot learning function description

When the user is not satisfied with the results of a question returned by the chatbot, ‘Captain’ will ask the user to type a satisfied answer to this question and writes the answer to a training set, which is retrained based on machine learning. After training, ‘Captain’ will have knowledge of the answer to this question and will tell the user to re-type the question to test Captain's learning ability.

6.4.2 Learning process

‘Captain’ will write the user's questions and answers to the training files and retrain them to store the results in the database, so that when the user is asked such questions again, captain will have a high probability of answering them satisfactorily.

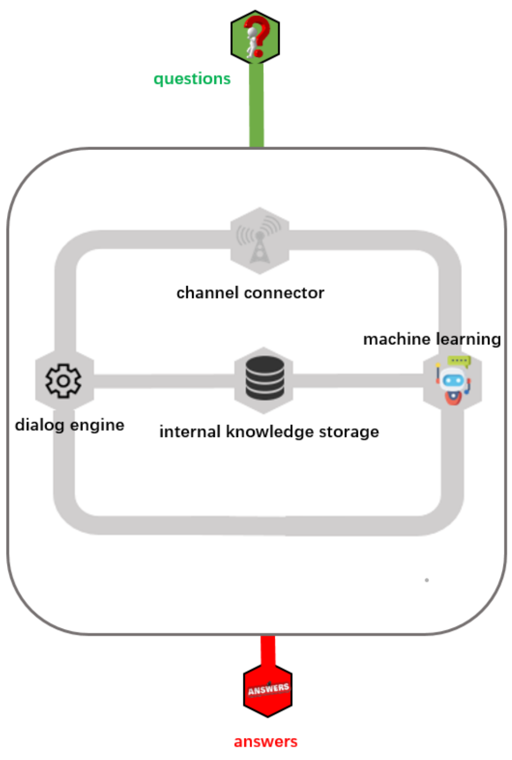


Figure 6.4.1: Learning process

7 Conclusion

Through the project, we use following three technique aspects:

(1) Decision automation: Knowledge based reasoning

(2) Business resource optimization: Informed search by Dijkstra algorithm

(3) System designed with cognitive techniques or tools: Chatbot

Our group have designed an app named ‘Cyber Share’ that first requires the user to register with a username, password and default drop-off location. Once registered, the user is automatically logged in. After logging out of the app, the user has to log in manually and will be prompted with an incorrect password or username.

Once logged in, the user will first need to click on the chatbot button at the bottom to enter the chat screen. The chatbot named ‘Captain’ will automatically capture the default drop off location in the user's database and ask whether the user wants to change it. If it needs to be changed, the user will have to re-type the drop-off location. if not, the user's default drop-off location will be used.

Once the user's drop-off location is obtained, the user is directed to the home page and given the desired pick-up time. Once the pick-up time and drop-off location are confirmed, the system will make the best match and calculate the shortest route based on the users waiting for carpooling in the background. The interface then shows the number of carpoolers, trip direction, optimal path length and passing stations.

If the user has any further questions, he/she can go back to the chatbot interface and continue the conversation with the ‘Captain’ chatbot.

8 Reference & Dataset

Taxi Singapore | Taxi Stands in CBD
https://www.taxisingapore.com/taxi-stands/

LTA | Getting a Ride
https://www.lta.gov.sg/content/ltagov/en/getting_around/taxis_private_hire_cars/getting_a_ride.html

overpass turbo
http://overpass-turbo.eu/#

9 APPENDIX

9.1 Appendix A – Project Proposal

Date of proposal:
28 Sep 2022

Project Title:

ISS Project - Intelligent carpooling system

Group Name
Cyber Captain

Group Members (name , Student ID):
Feng Lei, A0261852A

Han Kaiyang, A0261622M

Xu Binghui, A0262008N

Background/Aims/Objectives:

The Cyber share software is facing to NUS students, providing them a commute with less fare, and help them to make more friends.

The car will shuttle through multiple sites. When people request carpooling, the app will offer reasonable matching between passengers and calculate the shortest path to a carpool station. Finally, students receiving concessions will recommend this app to their friends.

Project Descriptions:

Firstly, the SQL database is set up to enable interoperability between SQL and the Flask and React frameworks. On the basis of the front and back-end work, three main functions of the system are completed.

The first is the clustering decision, using the clustering model and drop-off location, pick-up time input to generate rules to the next level and carpooling information to the whole system.

The second is route generate by unsupervised search method-the Dijkstra algorithm, which derives information about the start and end points from the clustering process and generates optimal route suggestions for the passenger.

The third is a chatbot, which provides answers to the user's questions and is implemented through a dialogue engine that combines an internal knowledge database with machine learning methods such as supervised learning.

9.2 Appendix B – Mapped System Functionalities against knowledge, techniques, and skills of module courses.

Module Courses	System Functionalities/ Techniques Applied
Machine Reasoning (MR)	Knowledge Representation Rule Based System Supervised Learning
Reasoning Systems (RS)	Generative Reasoning Systems (Uninformed search)

9.3 Appendix C– Installation and User Guide.

The link:

https://github.com/cybercaptaingroup9/CyberCaptain/blob/main/User%20Guide/Installation%20%26%20User%20Guide_Team%20Cyber%20Captain.pdf

9.4 Appendix C– Individual Report

9.4.1 Personal Report: Feng Lei

Personal Contribution

clustering decision, Dijkstra algorithm, Chatbot and the related Front and back-end works.

Lessons Learnt

Team work

Programming skills

Better understanding of intelligent reasoning systems

Future

The Intelligent Reasoning System contains a lot of interesting and practical knowledge. In the process of debugging the program, my understanding of machine learning and artificial intelligence has deepened, and I will encounter many scenarios where I can use this knowledge in my work..

9.4.2 Personal Report: Han Kaiyang

Personal Contribution

clustering decision, Dijkstra algorithm, Chatbot and the related Front and back-end works.

Lessons Learnt

Team work

Python Programming debug skills

The previous chatbot workshop was most useful for me

Future

This project, which achieved the initial product design then code implementation and finally the product launch, was a very challenging process. In my future research, I will use the thinking of intelligent reasoning systems to solve problems in daily life and use machine learning methods to deduce difficulties in distress. I will actively debug and continuously improve code capabilities.

9.4.3 Personal Report: Xu Binghui

Personal Contribution

clustering decision, Dijkstra algorithm, Chatbot and the related Front and back-end works.

Lessons Learnt

Team work

Self-study ability

Skills in programming and debugging

Future

For myself, it’s a fresh experience to finish a pure programming program rather than using any hardware or machine equipment. Thus, it’s really a treasurable project to practice my skills in artificial intelligence, software programming, and help me to understand the usage and meanings of hyper parameters of machine learning algorithms.