## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE DESCRIPTION** | **PAGE NO** |
| **2.1** | System Architecture for Car-Pooling System | 11 |
| **4.1** | Data Flow Diagram Level 0 for the entire  Car-Pooling System | 16 |
| **4.2** | DFD Level 1 for Module Representation | 17 |
| **4.3** | DFD Level 2 for Data Processing | 18 |
| **4.4** | DFD Level 3 for Offering a Ride | 18 |
| **4.5** | DFD Level 4 for Emergency | 19 |
| **4.6** | Use Case Diagram of Car-Pooling System | 20 |
| **4.7** | Sequence Diagram of Car-Pooling System | 20 |
| **4.8** | Class Diagram of Car-Pooling System | 21 |
| **7.1** | White Box Testing of Error Code | 30 |
| **7.2** | White Box Testing of Corrected Code | 31 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TABLE DESCRIPTION** | **PAGE NO** |
| **7.2.1** | Black Box Testing for Face Recognition | 32 |
| **7.2.2** | Black Box Testing for Mapping Location | 33 |

## CHAPTER 1 INTRODUCTION

A supermarket is an expansive retail facility carrying a wide range of products under one roof, including full groceries lines and general merchandise. In theory, supermarkets allow customers to satisfy all their routine shopping needs in one trip. This project has all the basic and necessary functions that is involved in the working of product management software. The user can choose from a number of features such as add product(s), modify product(s) with the help of various unique identifiers, delete product(s), or simply search product(s). This supermarket management system has realized the transmission and control of large goods, so as to facilitate the management and decision of sales, and reduce a big burden for supermarkets and supermarket managers. It also can help to improve the work efficiency of supermarket. Its requirements is to provide the basic information maintenance function of employees, memberships and products so that managers can through the function to add, delete, and modify the basic information of employees and the employees can through it to add, modify and delete the basic information of memberships and goods. Supermarket management system is very convenient for manage, input, output, and find the data so as to make the messy supermarket data to specific, visualizations, rationalization.

The management system will provide all the administrative functionality and will have full command to manage all the branches on the single management system. Super market services include a wide variety of different items like foods and many different household and other items. The management system will provide details for both sales and purchase of the items on a single screen. The database will be setup that will contain all the necessary data like product categories that will help is easy selection of items available that will be convenient for the customers and administration also. In short, the objective of super market management system is to manage the activities that doing in super market like tracking records of monthly or annual sales, maintaining the stock details, adding or removing the products on admin side, displaying the items that are up for sale, discounts on the items etc. This management system will be a complete package for maintaining super market activities.

## OVERVIEW OF BACKEND

* + 1. **WEB FRAMEWORK**

Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc. With Streamlit, no callbacks are needed since widgets are treated as variables. Data caching simplifies and speeds up computation pipelines. Streamlit watches for changes on updates of the linked Git repository and the application will be deployed automatically in the shared link.

## DATA ANALYSIS

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

## DATA VISUALIZATION

The Plotly Python library is an interactive open-source library. This can be a very helpful tool for data visualization and understanding the data simply and easily. plotly graph objects are a high-level interface to plotly which are easy to use. It can plot various types of graphs and charts like scatter plots, line charts, bar charts, box plots, histograms, pie charts, etc.

## GPS

A matplotlib-like interface to render all the data on top of Google Maps. Python library gmplot allows us to plot data on google maps. gmplot generates the HTML and JavaScript to deliver all the additional data on Google Maps.

# CHAPTER 2

## SYSTEM ANALYSIS

With the increasingly fierce competition, how to reduce the cost has become the supermarket's vital problem. For ordinary supermarkets, involving the management of the sources of raw materials, sales and inventory, and the good or bad management is very important to the durability of the supermarket. Generally speaking, the user's demand for procurement, sales and inventory system is universal.

## EXISTING SYSTEM

Data appropriate to the visualization of the shopping path in existing system is brief-shopping session data characterized by minimal duplicated movement. Where customers move back to the same section numerous times, resulting in multiple overlapping shopping paths, the identity is lost. In addition, visualization of the shopping path clarifies customer rounds made inside the store as well as movements between sections, but does not allow for visualization of the length of time spent in a section.

## LITERATURE SURVEY

In the paper [1] discussed that Data Analytics in retail now helps grocers remove the uncertainty factor from Inventory Management. Ask any retailer and they’ll tell you how painful it is to have products that few customers buy. Data Analytics in retail accurately predicts demand and suggests better replenishment strategies. It does not end there. By deploying Data Analytics, shop owners can identify where offering up a new product might increase revenue. Inventory imbalances are removed because of Data Analytics. The overall result is a decrease in inventory costs and an increase in sales.

Papers that fall in this category aim to provide a visual design that improves the understanding of a business’ internal or external environment. The emphasis is that the resulting visual system is created for the use of a single business as opposed to a whole economy or ecosystem. In the “Business Intelligence Guidebook”, Sherman states that BI turns data into “actionable” information. It is this output that businesses strive for through whatever means are available to them. BI is seen as both a process as well as a saleable product.

In the paper [2] discussed that The main purpose of paper deals with Internal Intelligence involves the knowledge of internal business processes. Papers in this category often aim to improve business process efficiency or gain a better understanding of the internal structure of the company. This perspective is inward facing. For example, Kandel et al. explored the role that visualization plays in day-to-day business operations [18]. The focus is placed on a company’s internal operations.

In the paper [3] discussed the paper reports, External intelligence examines the business ecosystem from the perspective of a single business. The focus is often placed on the business competitors to aid in competitive development or in identifying business operations out of the businesses control. This perspective is outward facing. For example, Hao et al. used visualization to explore fraud detection data in the banking industry. Customer-centric literature focuses on visualizing customer data. Businesses are moving towards a customer focused method of operating. This focus ensures that the customers’ interests are seen as the highest priority and therefore benefits the business through customer loyalty and superior product development. The differences between primary and secondary data sources in visualisation.

Research involving primary data proposes the visualisation hypothesis before the data is collected, whereas research involving secondary data proposes the visualisation hypothesis with the knowledge that the data has already been collected (a posteriori hypothesis). The hybrid pipeline contains two stages of data collection pre and post hypothesis. The initial creation of the data is often in the context of social media. At this point, the data are not collected to support the hypothesis posed by the visualisation research. Once the visualisation hypothesis is formulated, the data can then be scraped and collated into a second structured dataset for the purpose of visual research.

In the paper [4] discussed that the goal is to provide Study data are collected to support an a priori hypothesis first hand through interviews, questionnaires, and reviews. The most popular use of study data in this survey lies in the internal intelligence classification. These data are typically collected as part of the visualisation research. For example, Kandel et al. presented an interview study with data visualisation analysts working with industry to characterize the process of industrial data analysis.

Pre-existing databases are often used as a case study to demonstrate new visualisation techniques. They are databases that are created for the purpose of previous analysis, and not for the visualisation research in which it is currently used. Roberts et al. used a pre-existing database for their tree map-based research on call centre data, provided by their industry partner.

## MOTIVATION

The application under consideration, the Supermarket management application, is a novel idea which has never been implemented before. This became our source of motivation for going ahead with this project. These simple, color-coded data visualizations allow retailers to turn a store floor into an analytics narrative. A different source of motivation behind the development of this application is the integration of Dashboard and Graphical Representation of product line.

## PROPOSED SYSTEM

Our proposed system overcomes the drawbacks of the existing system. It has advance facilities to make it more user-friendly. It provides details of the products and their purchase to maintain transparency between users of the system. It will track the location of supermarkets situated in San Francisco. It also tracks the product data that are been registered beforehand.

ALGORITHM

Step1: Start

Step2: Register through the website and Login

Step3: If user wants to analysis the data, then Step4.1: User must go to Dashboard

Step4.1.1: On Sidebar, We can filter requirement we wants.

Step4.1.2: Filter the customer type by selecting the different references customer type.

Step4.1.3: Filter the city by selecting the different references cities.

Step4.1.4: Filter the Gender by selecting male or female or both.

Step4.2: Choose About.

Step4.3: Check for Supermarkets nearby the location.

Step4.4: Choose Register Page.

Step4.5: Enter the details which are needed to be Update the Database.

Step5: If user has updated the database, then

Step5.1: User can view the Updated Database in Updated Table page.

Step5.2: After updating, the user can view the Visualized Graph.

Step6: Stop

## MODULES

In the Supermarket Visualization application, User needs to register data. The verification of user’s license will be done by live image capture for authentication. User needs to map the location then look for other users to drop them at nearby location and share their ride. User who plans a ride accepts the request and confirms the ride to reach the destination.

## Presenting Data:

The user after login can view the product data that has been entered. The steps to view the data are mentioned below,

## Steps:

Step1: Start

Step2: Admin login with username and password

Step3: We need to register by giving all the details Step4: Visit the Homepage

Step5: User searching for their product data

Step6: Selecting the product

Step7: Viewing user details and product details

Step8: Stop

## Dashboard analysis:

The user can analysis the visualized data and filter the options .

The steps to Offer a ride are mentioned below,

## Steps:

Step 1: Start

Step 2: Total Transaction amount will be displayed.

Step 3: Two Graph will be show and one based on gender analysis.

Step 4: Another one based on Product line Analysis.

Step 5: Viewing Dashboard details.

Step 6: Filter the city or gender or Customer type.

Step 7: Check the changes occur on the Graph.

Step 8: Stop

## Data Entry:

Data Entry changes apply in database of the products. The steps to Data Entry before viewing the Updated Table are mentioned below,

## Steps:

Step 1: Start

Step 2: Go to the register entry pages to enter the data.

Step 3: Enter the data for the purchased products.

Step 4: Upon Completion, Submit the form to review the entries.

Step 5: The Updated data can be viewed on Updated Table.

Step 6: Stop.

## Locate Supermarket:

Locations of Supermarkets in the map Box which are the franchise of SaveMart. The steps to locate supermarket are mentioned below,

## Steps:

Step 1: Start

Step 2: Fetch the location of the Supermarket present in San Francisco.

Step 3: Can Zoom In or Out of the Located Branch.

Step 4: Stop

## View Update:

While offering a ride user’s face needs to be matched with their license photo for security purposes. The steps for user authentication are mentioned below,

## Steps:

Step 1: Start

Step 2: Import face\_recognition module.

Step 3: face\_encoding() is used to encode the facial features which can be compared with any other picture of a face.

Step 4: Authenticate the encoding of face in driving license with the face captured in web cam using compare\_faces().

Step 5: Stop

## Mapping Locations:

After the user decided whether to plan or decide a ride, they can map their location through GPS. The steps to map the locations are mentioned below,

## Steps:

Step 1: Start

Step 2: Get the latitude and longitude for the Pick and Drop location using OpenStreetMap API.

Step 3: Create a base map with the obtained latitude and longitude.

Step 4: Geodesic distance (in kms) is calculated from latitude-longitude data using geopy module.

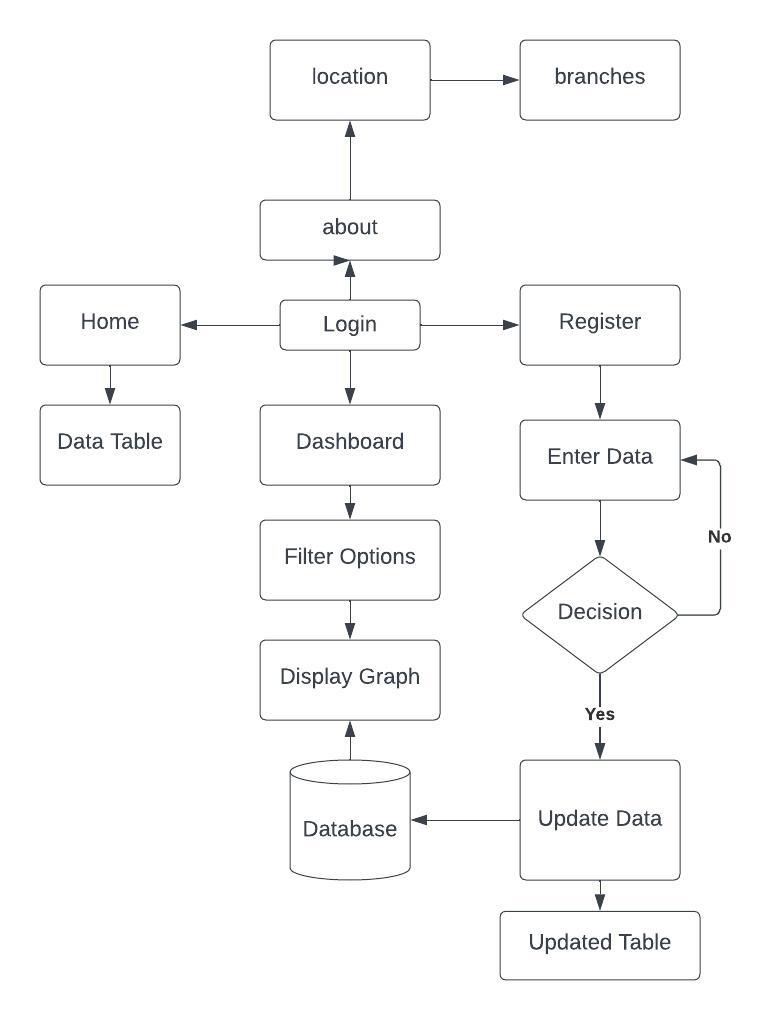
Step 5: Stop

## ADVANTAGES OF PROPOSED SYSTEM

* Forecasting Shifts in Pattern of Consumer Demands.
* The Amalgamation of Big Data and Specific Information of Organization.
* Understanding and Managing Retail Flow.
* Enhance Inventory Visibility.
* User-Friendly.
* Improve Merchandising Competitive Positions.
* Explore Omni channel Marketing Impacts.
* Enabling IT Stakeholders to Visualize Data Better.

## SYSTEM ARCHITECTURE

The Figure 2.1 shows the architectural design of a system emphasizes the design of the [system architecture](https://en.wikipedia.org/wiki/System_architecture) that describes the [structure](https://en.wikipedia.org/wiki/Structure), [behavior](https://en.wikipedia.org/wiki/Behavior) and more [views](https://en.wikipedia.org/wiki/View_model) of the proposed system and analysis.



**Figure 2.1 System Architecture of Car-Pooling**

User registers to the website that stores the credentials in the Database. The User should Login through the website and can access the website in 4 ways,

* Home
* About
* Dashboard
* Register

# CHAPTER - 3

## SYSTEM REQUIREMENTS

A System requirement specifies the project hardware and software requirements for developing the project. Supported operating system and run time environments where the project will run is on windows OS with python IDE. Since python IDE provides efficient way to implement face data Visualization, verification of user and mapping location using various python Packages.

## HARDWARE REQUIREMENTS

RAM : 4 GB and above Processor : 2.80 GHz and above Hard Disk : 120 GB and above CPU type : Intel Pentium 4 Clock speed : 3.0 GH

Monitor type : 15 Inch color monitor Keyboard type : Internet keyboard

CD -drive type : 52xmax

## SOFTWARE REQUIREMENTS

Operating system : Windows 7, Windows 8, Windows 10, Window 11 Language : Python

Documentation tool : Microsoft word 2019 Back end : Streamlit

Simulation Tool : Pandas

## SOFTWARE FEATURES

* + 1. **PYTHON**

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming. In Python, we don't need to declare the type of variable because it is a dynamic typed language.

**EASY TO CODE**

Python is a very developer-friendly language which means that anyone and everyone can learn to code it in a couple of hours or days. As compared to other object- oriented programming languages like Java, C, C++, and C#, Python is one of the easiest to learn.

**OPEN SOURCE AND FREE**

Python is an open-source programming language which means that anyone can create and contribute to its development. Python has an online forum where thousands of coders gather daily to improve this language further. Along with this [Python](https://www.edureka.co/blog/python-basics/) is free to download and use in any operating system, be it Windows, Mac or Linux.

## STREAMLIT

Streamlit is an open source app framework in Python language. It helps us create web apps for data science and machine learning in a short time. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc..

## PANDAS

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

## PLOTLY

The plotly Python library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.

## GMPLOT

A matplotlib-like interface to render all the data you'd like on top of Google Maps.

Several plotting methods make creating exploratory map views effortless.

## DATA VISUALIZATION

Data visualization is the representation of data through use of common graphics,

Such as charts, plots, infographics, and even animations. These visual displays of

information communicate complex data relationships and data-driven insights in a

way that is easy to understand.

## CHAPTER 4 SYSTEM DESIGN

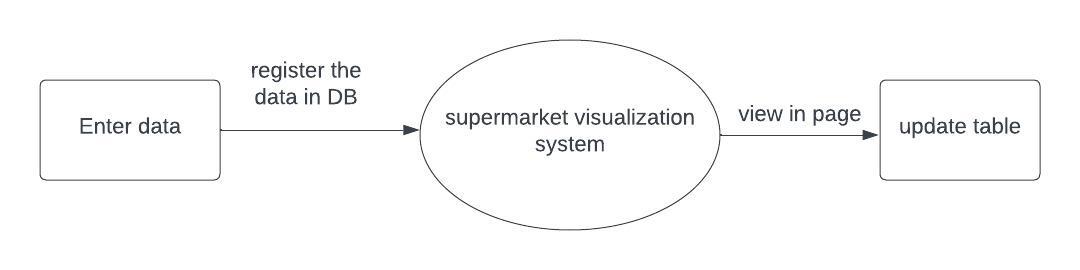
The term “Design” is defined as the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application.

## DATA FLOW DIAGRAM

DFD are used to Specify Functions of the Information System and how data flow from function to function. A data flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart. The data flow diagram is part of the structured analysis modeling tools.

## DATA FLOW DIAGRAM AT THE INITIAL LEVEL

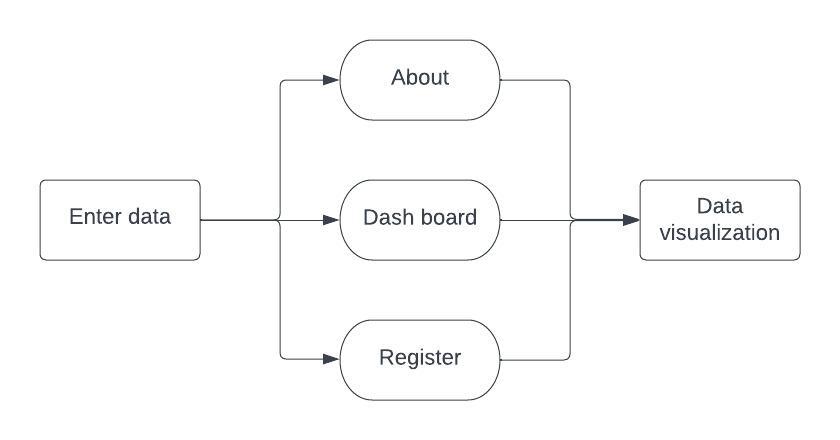
Figure 4.1 explains the overall functionality of the proposed system application which is to enter the data and register on Database to the supermarket visualization system and view the updated page in updated



**Figure 4.1 Data Flow Diagram Level 0 For the Initial Level of Supermarket Visualization System**

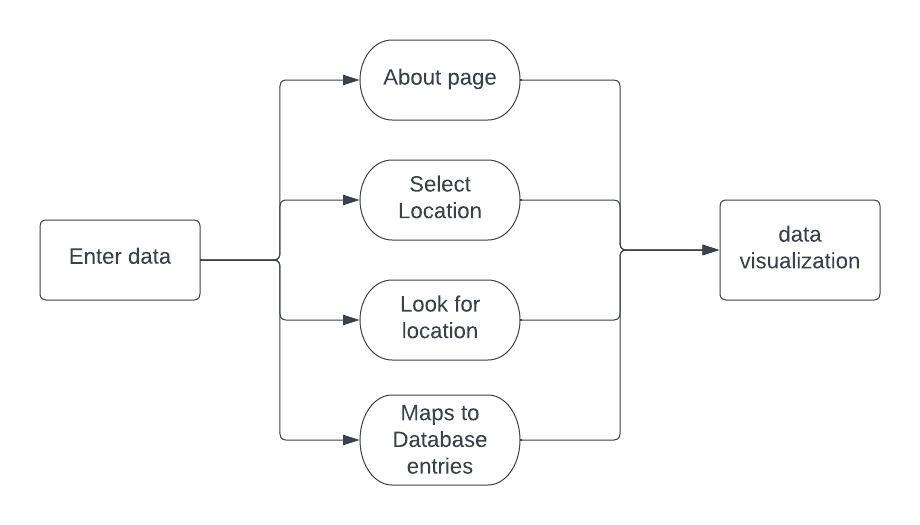
## DATA FLOW DIAGRAM LEVEL 1

Figure 4.2 describes about the overall representation of each module and their functions. The level one data flow diagram has various modules and the respective results.



**Figure 4.2 Data Flow Diagram Level 1 Module Representation**

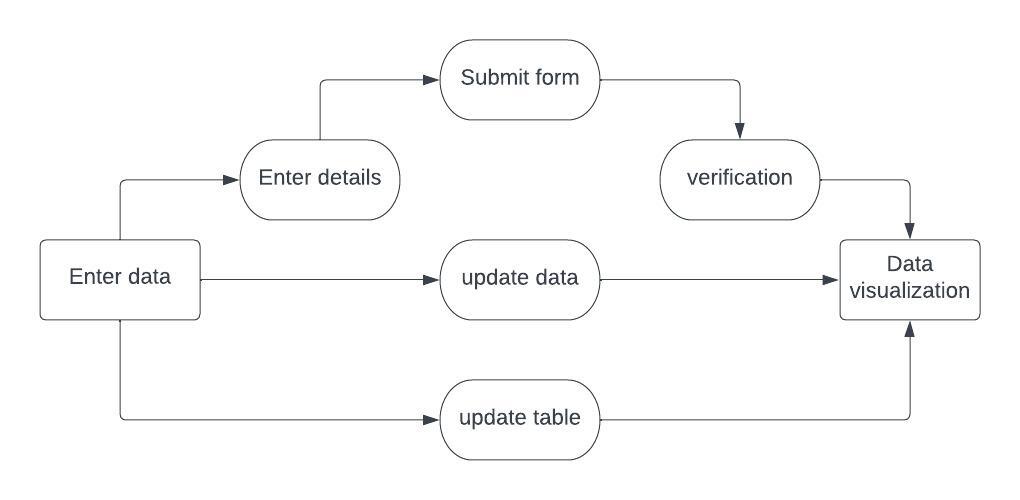
* 1. **DATA FLOW DIAGRAM LEVEL 2 FOR**

Figure 4.3 shows the Data Flow diagram level 2 of entire project. It describes the planning of the ride by the user. The user needs to provide the pickup and drop location and search for cabs. After finding a cab the user needs to confirm the ride and he/she can track the location during travel. After reaching destination payment is done.

**Figure 4.3 Data Flow Diagram Level 2 for Data Processing**

* 1. **DATA FLOW DIAGRAM LEVEL 3 FOR OFFER A RIDE**

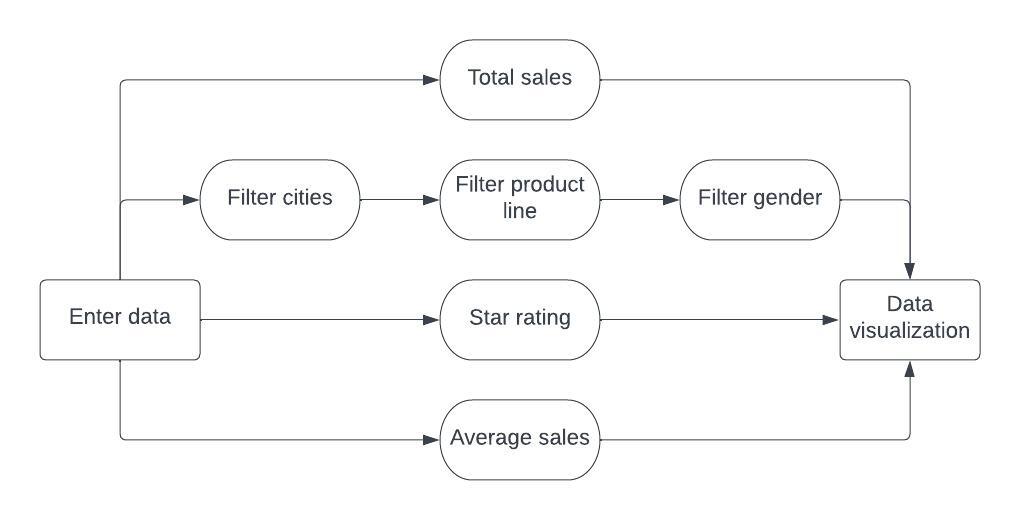
Figure 4.4 of Data Flow Diagram Level 2 for Offering a ride where the user must upload the license and a live video capture takes place to verify whether the face verifies with the license. After verification the user must enter the current location and destination to select a ride and drop the customer. Payment must be done.



**Figure 4.4 Data Flow Diagram Level 3 for Offering a ride**

## DATA FLOW DIAGRAM LEVEL 4 FOR EMERGENCY

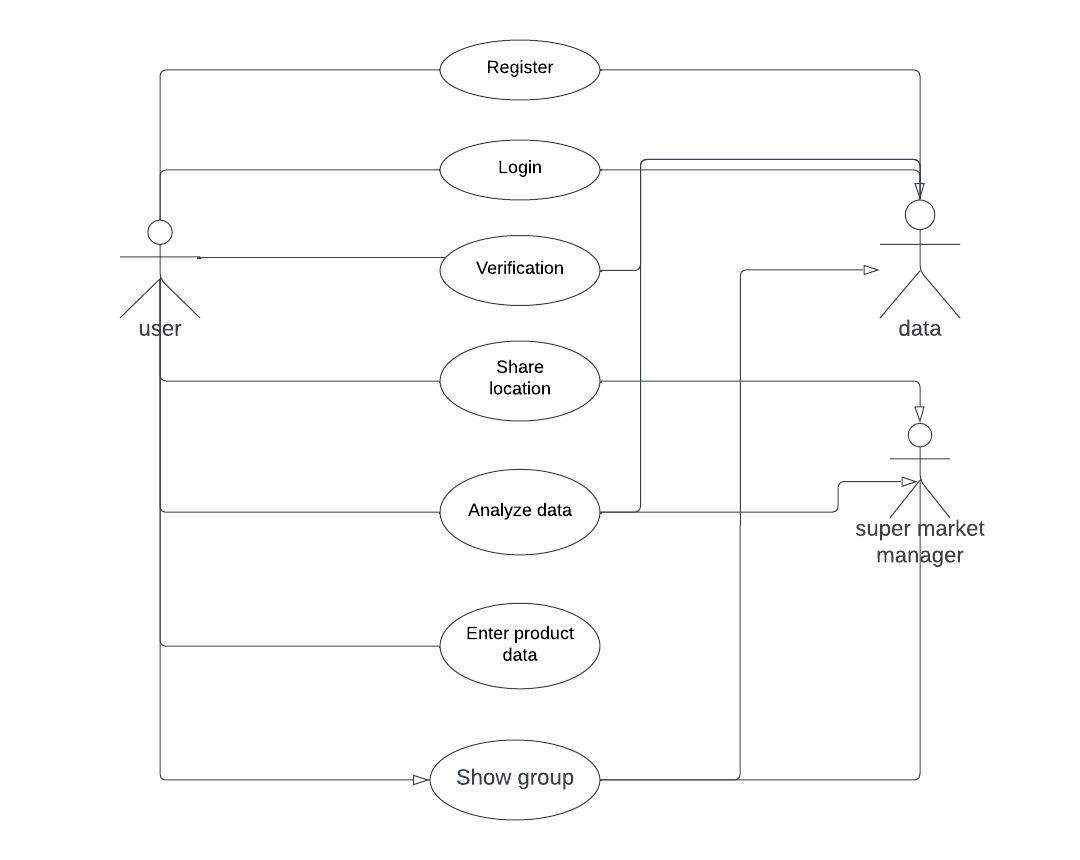
Figure 4.5 of Data Flow Diagram Level 4 for Emergency, when emergency occurs the current location of the user is tracked using GPS and nearby hospital can be searched. The request for ambulance is sent to the hospital management to save lives.



**Figure 4.5 Data Flow Diagram Level 4 for Emergency**

## USE CASE DIAGRAM

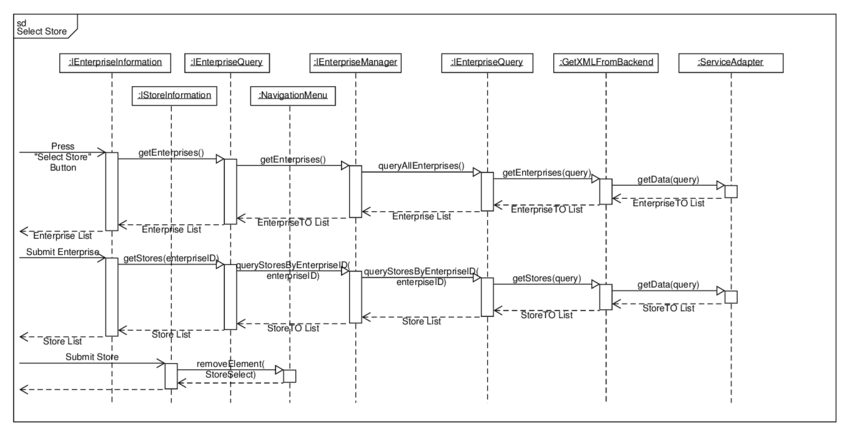
Figure 4.6 of Use Case Diagram depicts the relationship between all the modules. It includes phases such as Register through the website, Login using credentials, share a ride, verify user, Plan a ride, Payment Gateway and Emergency purposes.



**Figure 4.6 Use Case Diagram of Car-Pooling System**

## SEQUENCE DIAGRAM

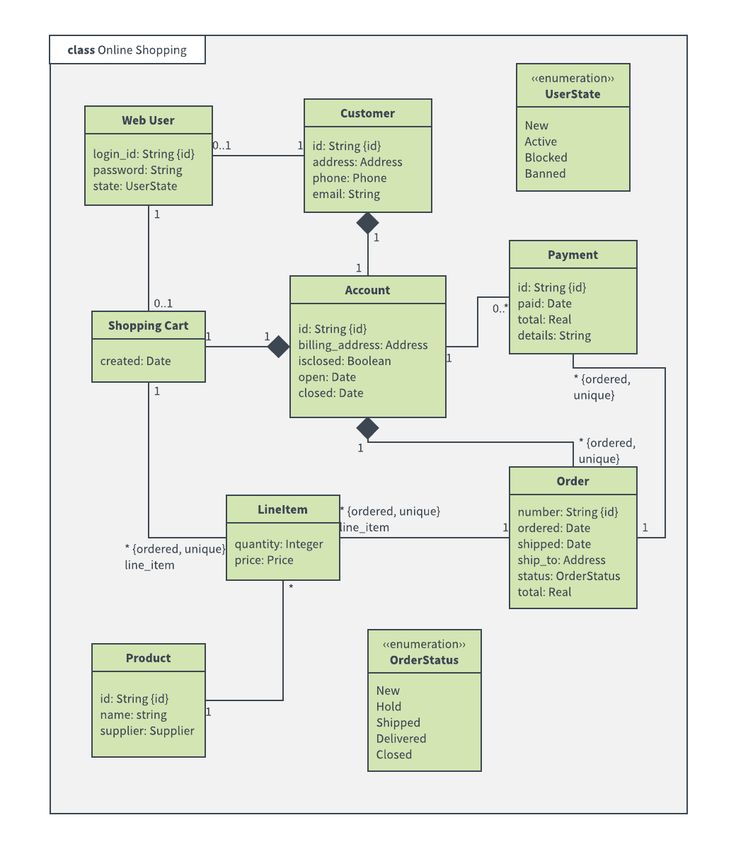
Figure 4.7 of Sequence diagram depicts the flow of logic within system in a visual manner, enabling the database with classifying the modules and providing the methods to access the application.



**Figure 4.7 Sequence diagram of Car-Pooling Application**

## CLASS DIAGRAM

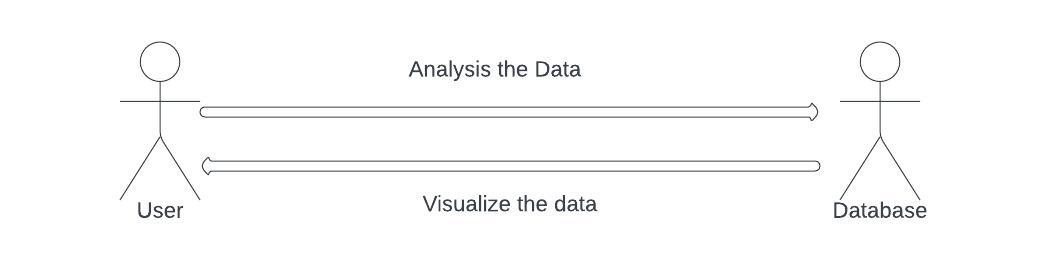
Figure 4.8 of Class diagram of Car-Pooling System divides into five classes known as User, find ride, Offer ride, Verification and Emergency. User gives name and password to register and login themselves. They can Offer a ride, Find a ride and Avail emergency. To offer a ride, authentication of user is done. To find a ride user gives pickup and drop location to search a ride and after finding a suitable ride he/she can track the live location through GPS. While offering a ride when the destination location matches, the user can drop off the customer at their respective location and avail the payment. For emergency purposes, the system requests the hospital management for an ambulance and later the ambulance arrives.



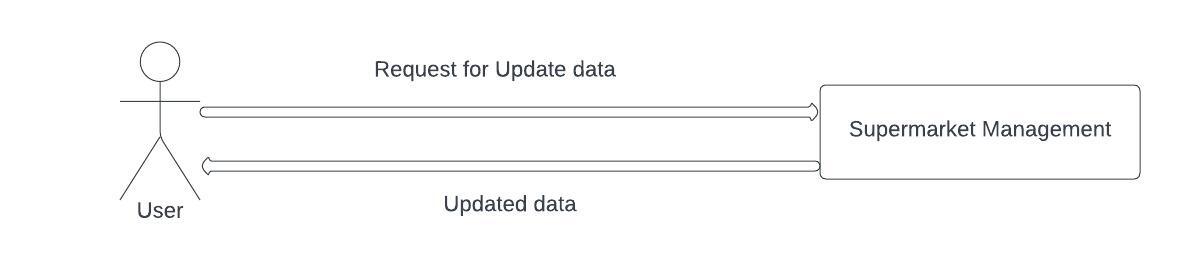
**Figure 4.8 Class Diagram**

## CHAPTER 5 SYSTEM DEVELOPMENT

Car-pooling allows sharing rides based on mutual acceptance algorithm, which consist of two steps. In the first step, verified user plans a ride and makes request to the registered users who planned and posted their rides with same destination. Second step makes sure that the registered users accept the request they received. After completion of the above steps the ride will be confirmed for both the users.



Emergency management helps the verified users to make request for ambulance service. This request is sent to the nearby hospitals. Once the hospital(s) notifies the availability of ambulance service, the service offered by the first hospital is accepted and the rest of the requests are rejected based on FIFO algorithm.



## DATABASE CONNECTIVITY

def get\_data\_from\_excel():

df = pd.read\_excel(

io="supermarkt\_sales.xlsx",

engine="openpyxl",

sheet\_name="Sales",

skiprows=0,

usecols="B:R",

nrows=1000,

)

* 1. **USER REGISTRATION** @app.route('/regsub',methods=['POST','GET']) def regsub():

print("gf")

name = request.form['username'] email = request.form['email'] pwd = request.form['psw'] phone = request.form['phone'] add = request.form['add']

sql = "INSERT INTO register (username,email,password,phnum,address) VALUES (%s, %s, %s, %s, %s)"

val = (name,email,pwd,phone,add) cur.execute(sql, val) mydb.commit()

return render\_template('index.html',n=name)

## LOGIN

@app.route('/log1', methods=['POST','GET']) def log():

email = request.form['email'] pwd = request.form['psw'] cur.callproc("login",[email,pwd]) for result in cur.stored\_results():

uname = result.fetchall()

return render\_template('index.html',n=uname[0][0])

## UPLOAD LICENSE

@app.route('/upload', methods = ['GET', 'POST']) def uploadfile():

if request.method == 'POST': f = request.files['file'] f.save("id.jpg")

print(f.filename)

return render\_template('upload.html')

* 1. **VERIFY UPDATED DOCUMENT**

if submit:

st.markdown('<p class="submit" style="color:#36E110;text-align: center;">Successfully Submitted</p>',unsafe\_allow\_html=True)

rows = [[id, branch, city,type,gender,product\_line,unit\_price,

quantity,tax,total,date,time,payment,cogs,

gmp,gi,rating

]]

df=pd.DataFrame(rows)

st.write(df)

workbook\_name = 'supermarkt\_sales.xlsx'

wb = load\_workbook(workbook\_name)

page = wb.active

for info in rows:

page.append(info)

wb.save(filename=workbook\_name)

st.markdown(hide\_style,unsafe\_allow\_html=True)

st.title("Currently Updated Data Of the Super Market:")

df = get\_data\_from\_excel()

st.write(df.tail())

## MAP LOCATION

def mapping(lat1,lon1,lat2,lon2): latitude\_list = [ lat1,lat2] longitude\_list = [ lon1,lon2 ]

gmap3 = gmplot.GoogleMapPlotter(lat1, lon1, 18)

gmap3.scatter( latitude\_list, longitude\_list, '#FF0000', size = 40, marker = False ) gmap3.plot(latitude\_list, longitude\_list, 'cornflowerblue', edge\_width = 2.5) gmap3.draw( "templates\map13.html" )

st.map(gmap3)

@app.route('/dist',methods = ['POST','GET'])

## TRACK DISTANCE

def dist():

curr = request.form['curr'] dest = request.form['dest'] lat1,lon1 = lat(curr) lat2,lon2 = lat(dest)

print(type(lat1),lon1,lat2,lon2) mapping(lat1,lon1,lat2,lon2)

return render\_template('result.html',dist=geodesic(lat(curr), lat(dest)).km)

## CHAPTER 6 SYSTEM IMPLEMENTATION

Performance analysis is used to find which algorithm is smart according to the specific criteria. Depending on the nature of application there are various criteria to measure the performance of prediction in terms of accuracy. The ultimate goal of this Car-Pooling Application is that any user with prior documents can share rides with other users. This application allows the users to perform tasks like search the car on desired route, date and timing. The aim of this source is to describe the Car-Pooling Application which benefits the individual travelling who can monetize their resource. The online carpooling system is a web-based application is to provide us with a simple riding platform between the car owner and car user. This project enables users to access mobility assets own by others exactly when they need. People who have a car can travel together to save cost and to promote other social-environmental benefits. People who don't have a car can also contact people with car and commute together. Carpooling is also seen as a more environmentally friendly and helpful way to travel as sharing journeys reduces air pollution, traffic congestion on the roads as well as the need for parking spaces also.

## PREREQUISITES:

* + - Flask Framework – Python
    - MySQL Workbench – Database
    - Face\_recognition, OpenCV, gmplot, mysqlconnector, flask\_session – Python libraries

## REQUIREMENTS:

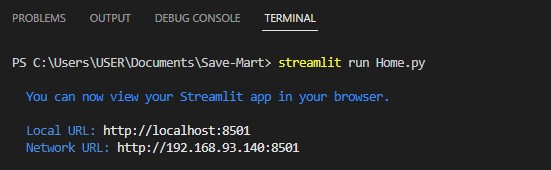
Type the following command to install all the required libraries, pip install -r requirements.txt

## DEPLOYMENT:

1. Clone car-pooling-system Repository In Local Your Machine.
2. With VS Code, Open Downloaded Save-Mart.
3. Build And Run

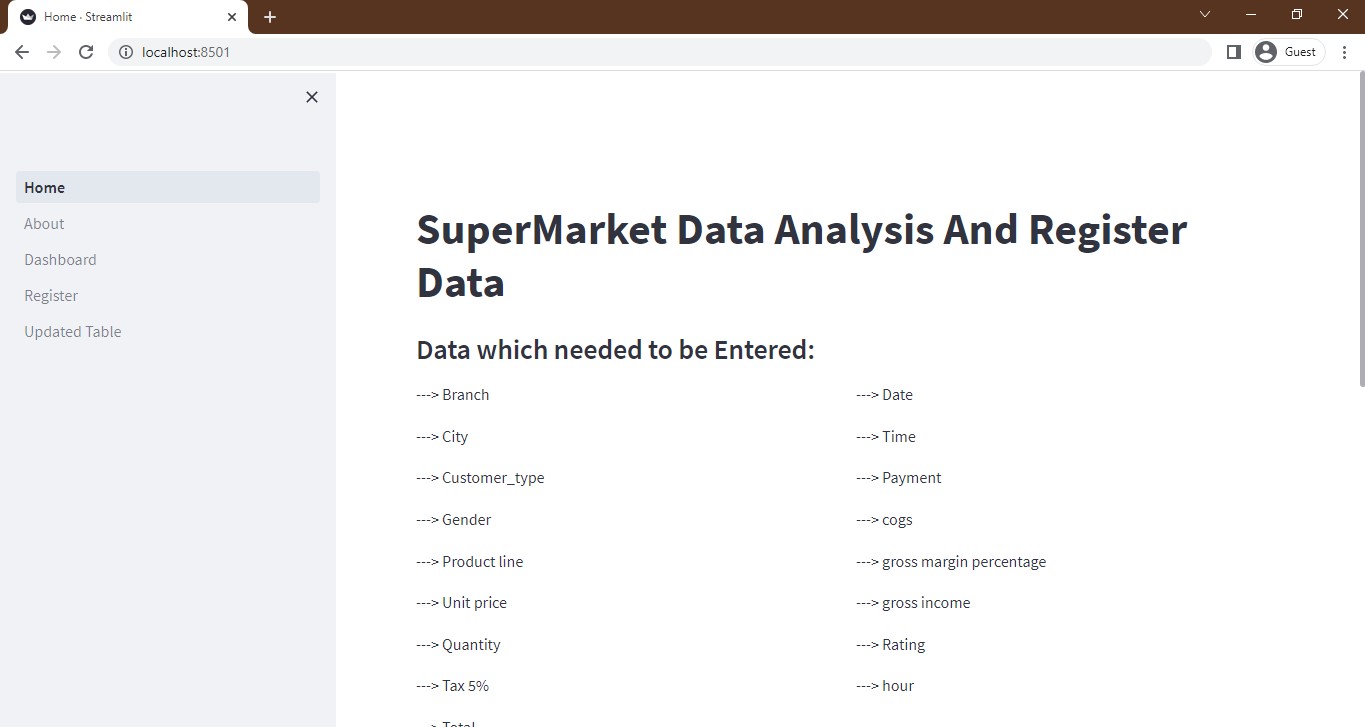
## RUN SERVER:

Type the following command in the terminal, streamlit run Home.py



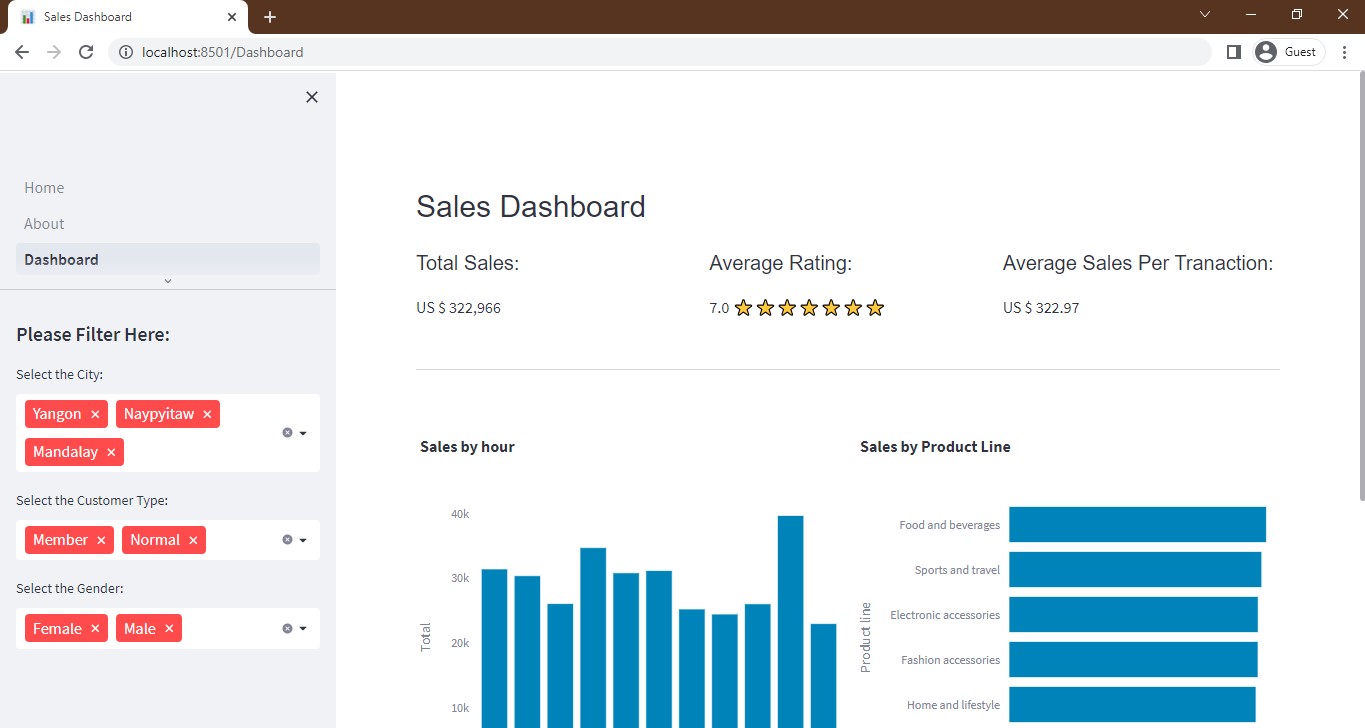
## HOSTING BROWSER:

Copy and paste the final url displayed into the browser to access the application.



## READY TO USE:

Now the application can be used as according to the user’s needs, they can book or share a ride and also for emergency purposes the user can book an ambulance.



# CHAPTER 7

**SYSTEM TESTING**

System testing is the testing of complete and fully integrated software product. Usually, software is only one element of a larger computer-based system. Ultimately, Software is interfaced with other software/hardware systems. System testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

## WHITE BOX TESTING

White box testing is a testing which includes the error in the coding section. It includes the error occurred during compilation and also during the development of the project.

## WHITE BOX TESTING FOR READING FILE

The below code explains white box testing which describes the 1-D array instead of multi-D array.

**ERROR CODE IN THE LIST OF DIRECTORIES**

k\_image = face\_recognition.load\_image\_file("id.jpg") u\_image = face\_recognition.load\_image\_file("cam.jpg")

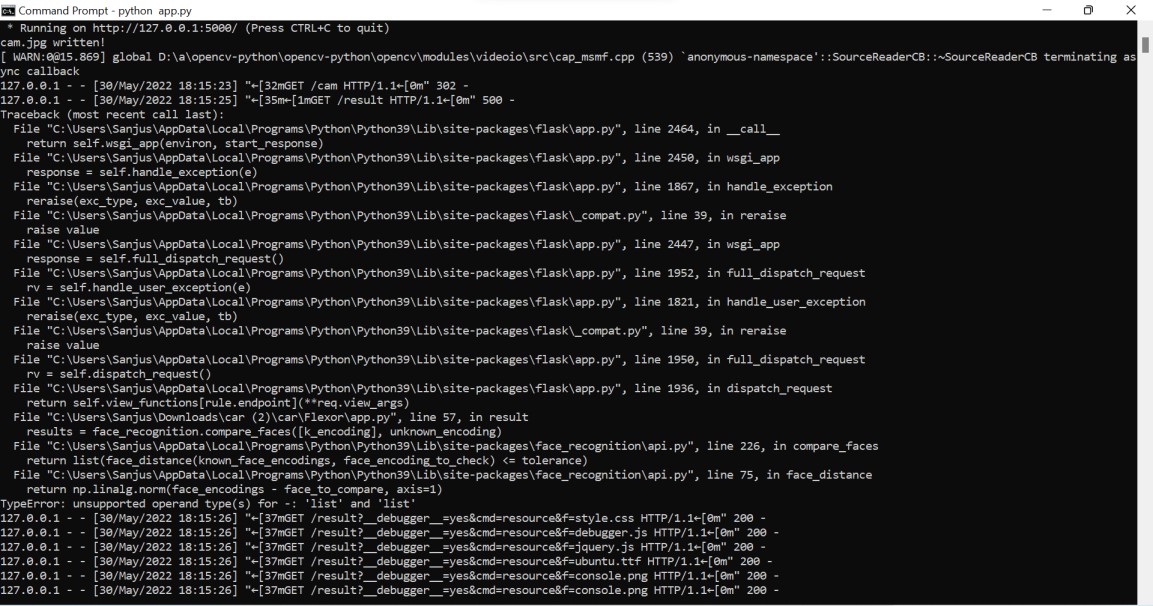
k\_en = face\_recognition.face\_encodings(k\_image) u\_en = face\_recognition.face\_encodings(u\_image)

Missing index position to abstract 1-D array

results = face\_recognition.compare\_faces([k\_en], u\_en) if(results[0]):

return render\_template('result.html',rs="Valid user")

**OUTPUT FOR THAT ERROR CODE IN FACE ENCODING**



**Figure 7.1 : Output for the Error Code**

**CORRECTED CODE IN LIST OF DIRECTORY**

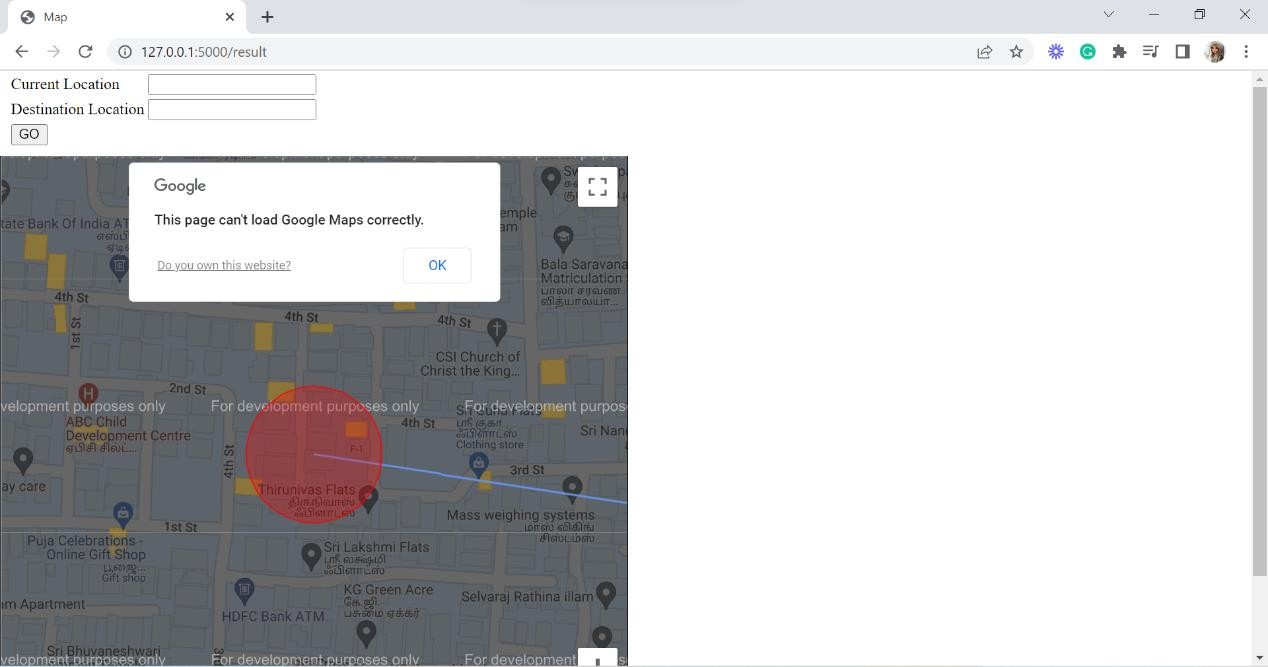
k\_image = face\_recognition.load\_image\_file("id.jpg") u\_image = face\_recognition.load\_image\_file("cam.jpg")

k\_en = face\_recognition.face\_encodings(k\_image)[0] u\_en = face\_recognition.face\_encodings(u\_image)[0]

results = face\_recognition.compare\_faces([k\_en], u\_en) if(results[0]):

return render\_template('result.html',rs="Valid user")

**OUTPUT**



**Figure 7.2 : Output for the Corrected Code**

## BLACK BOX TESTING OF CAR POOLING

Black box testing is a table analysis which helps the analyzer to verify the project Test cases are built based on the specification and requirements of the project. This project is verified and tested using Test cases. This table includes the test description, input case, expected result, actual result and status. It also tells the status of the test briefing whether it is failed or pass. Black box testing for face recognition for authenticated user and mapping the coordinates of the location.

## BLACK BOX TESTING FOR FACE RECOGNITION

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Description** | **Test Input** | **Expected Result** | **Actual Result** | **Stat us** |
| ID- 01 | Multiple faces in a video capture | Frame obtained from camera read. | Matches the correct face | Result based on the first image detected | Fail |
| ID- 02 | Encode the image data into numerical | Image file | Data values update to a numerical value | Data values update in the dataset | Pass |
| ID- 03 | Space key  press for video capture | Video capturing frame | Prompt showing valid key press | Frame is not affected with the key press | Fail |
| ID- 04 | Uploading driving license | Image of driving license in any image format | Automatica lly converted to jpg  format | Automatically converted to jpg format | Pass |

* + 1. **BLACK BOX TESTING FOR MAPPING COORDINATES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Description** | **Test Input** | **Expected Result** | **Actual Result** | **Stat us** |
| ID- 01 | Map the pick-up and drop location | Pick-up and drop address | Generate Coordinates and map the location | Generate Coordinates and map the location | Pass |
| ID- 02 | Address Case- sensitive | Pick-up and drop address | Map the location with the  appropriate address | Location is not mapped properly | Fail |

# CHAPTER 8 CONCLUSION

## CONCLUSION

Carpooling system is very effective means to reduce pollution and the congestion of vehicles in cities. It also provides an ecofriendly way to travel and an opportunity to meet new people. As today most people prefer private vehicle to travel due to delay caused in public transport system and luxuries provided by private vehicles. Pre- registration ensures that only identified people get into the vehicle so that trust can be established. The people registered are allotted specific days on which they should take their private vehicle, so that no inconvenience is caused to its registered passengers for daily commute. Thus, the proposed carpooling system will be effective in reducing environment pollution. It will also provide a security to citizens and gives the accurate pick-up time.

We have developed our project to handle emergency health situations and to evacuate the patient to a nearby and communicated hospital. The approach is extended toward rescuing a sufferer’s life in a more accelerated approach as feasible. It is extremely essential for sufferers in the matter of crises since it conserves time. With the help of this Project, the emergency vehicle can contact the user or victim as the position is followed or supplied within the application furthermore can produce the essential tools that are expected for the patient’s well-being.

## LIMITATION

Face Recognition in Video Capture may cause trouble as when person shows a photo of the registered user, it may also confirm as valid user. The Hospital needs to accept the request of the patient immediately as it is for emergency purposes. These limitations can be converted into cons by adding new features in the upcoming development.

## FUTURE ENHANCEMENT

The future enhancement of the project was developed taking in mind the benefits of the users. The developed system will be user-friendly for blind & lack of knowledge people. Bike can also be used in future for pooling. Pooling system can be for transportation goods in sharing manner (Truck Pooling). Future works include the collection of a larger multi-national driver license dataset that can be used for model generalization. The accuracy and the speed of the system can be increased by use of Google GPU for processing.

## APPENDICES

* + 1. **SAMPLE SOURCE CODE**

import streamlit as st

import pandas as pd

def get\_data\_from\_excel():

df = pd.read\_excel(

io="supermarkt\_sales.xlsx",

engine="openpyxl",

sheet\_name="Sales",

skiprows=0,

usecols="B:R",

nrows=1000,

)

df["hour"] = pd.to\_datetime(df["Time"], format="%H:%M:%S").dt.hour

return df

def main():

hide\_style="""

<style>

#MainMenu {visibility:hidden;}

header{visibility:hidden;}

footer{visibility:hidden;}

</style>

"""

st.markdown(hide\_style,unsafe\_allow\_html=True)

st.title("SuperMarket Data Analysis And Register Data ")

df = get\_data\_from\_excel()

st.subheader("Data which needed to be Entered:")

col1, col2 = st.columns(2)

list\_of\_column\_names = []

for row in df:

list\_of\_column\_names.append(row)

with col1:

for i in range(0, 9):

st.write("---> "+list\_of\_column\_names[i])

with col2:

for i in range(9, 17):

st.write("---> "+list\_of\_column\_names[i])

st.header("Data Which are entered in the DataBase Are:")

st.write(df)

st.markdown(hide\_style,unsafe\_allow\_html=True)

st.sidebar.header("Please Filter Here:")

city = st.sidebar.multiselect(

"Select the City:",

options=df["City"].unique(),

default=df["City"].unique()

)

customer\_type = st.sidebar.multiselect(

"Select the Customer Type:",

options=df["Customer\_type"].unique(),

default=df["Customer\_type"].unique(),

)

gender = st.sidebar.multiselect(

"Select the Gender:",

options=df["Gender"].unique(),

default=df["Gender"].unique()

)

df\_selection = df.query(

"City == @city & Customer\_type ==@customer\_type & Gender == @gender"

)

new\_title = '<p style="font-family:sans-serif; font-size: 30px;">Sales Dashboard</p>'

st.markdown(new\_title, unsafe\_allow\_html=True)

total\_sales = int(df\_selection["Total"].sum())

average\_rating = round(df\_selection["Rating"].mean(), 1)

star\_rating = ":star:" \* int(round(average\_rating, 0))

average\_sale\_by\_transaction = round(df\_selection["Total"].mean(), 2)

left\_column, middle\_column, right\_column = st.columns(3)

with left\_column:

new\_title = '<p style="font-family:sans-serif; font-size: 20px;">Total Sales:</p>'

st.markdown(new\_title, unsafe\_allow\_html=True)

st.markdown(f"US $ {total\_sales:,}")

with middle\_column:

new\_title = '<p style="font-family:sans-serif; font-size: 20px;">Average Rating:</p>'

st.markdown(new\_title, unsafe\_allow\_html=True)

st.markdown(f"{average\_rating} {star\_rating}")

with right\_column:

new\_title = '<p style="font-family:sans-serif; font-size: 20px;">Average Sales Per Tranaction:</p>'

st.markdown(new\_title, unsafe\_allow\_html=True)

st.markdown(f"US $ {average\_sale\_by\_transaction}")

st.markdown("""---""")

sales\_by\_product\_line = (

df\_selection.groupby(by=["Product line"]).sum()[["Total"]].sort\_values(by="Total")

)

sales\_by\_hour = df\_selection.groupby(by=["hour"]).sum()[["Total"]]

fig\_hourly\_sales = px.bar(

sales\_by\_hour,

x=sales\_by\_hour.index,

y="Total",

title="<b>Sales by hour</b>",

color\_discrete\_sequence=["#0083B8"] \* len(sales\_by\_hour),

template="plotly\_white",

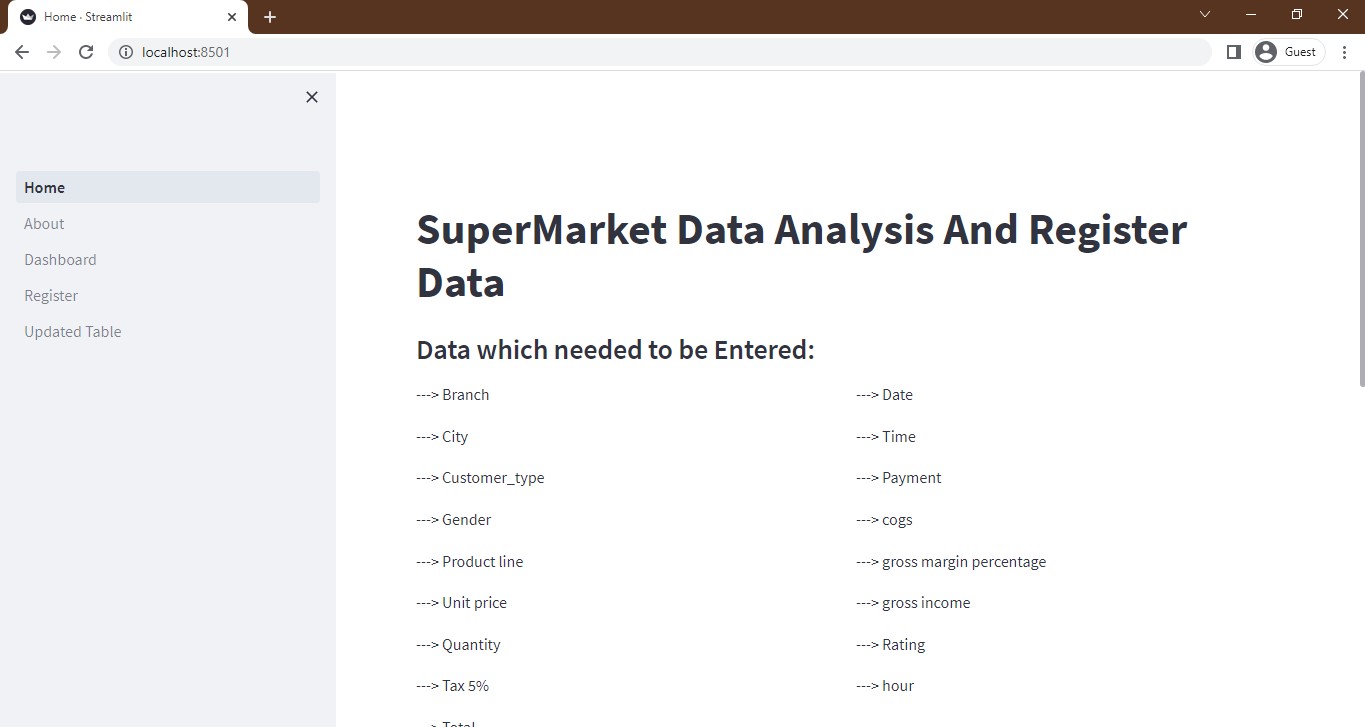
)

left\_column, right\_column = st.columns(2)

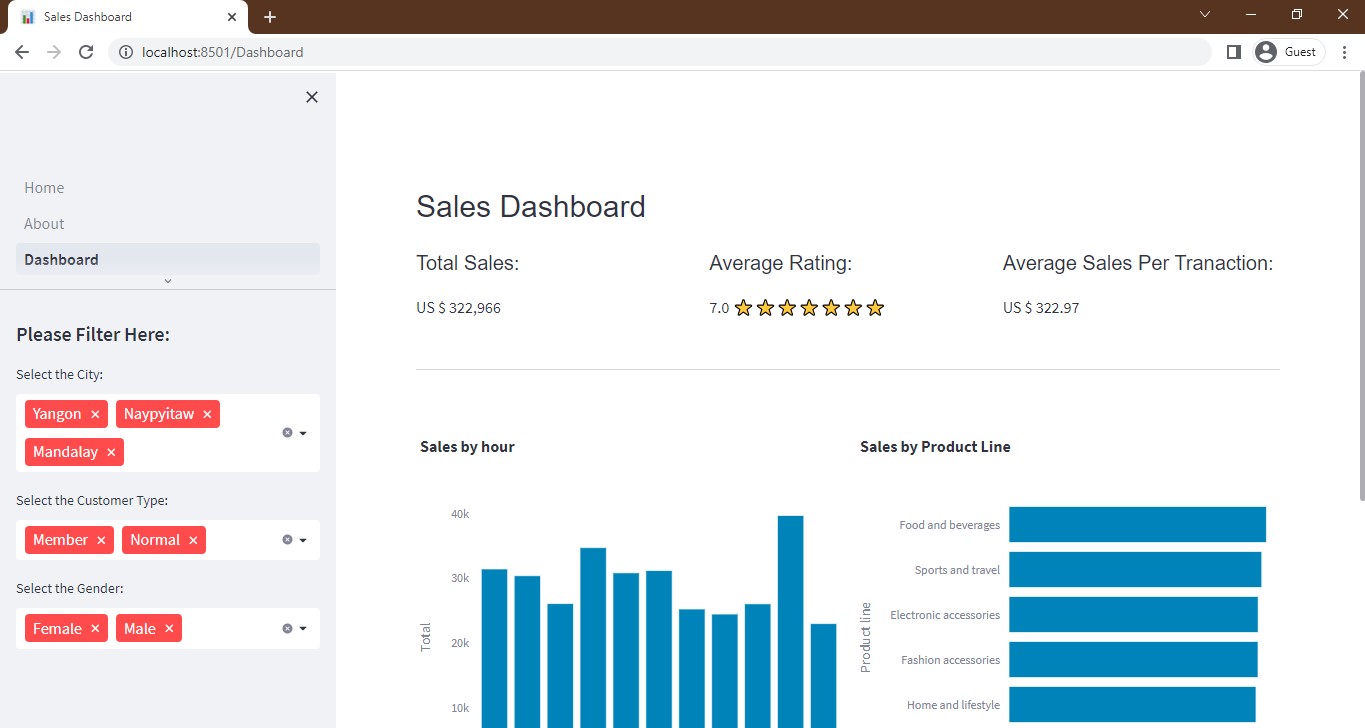
left\_column.plotly\_chart(fig\_hourly\_sales, use\_container\_width=True)

right\_column.plotly\_chart(fig\_product\_sales, use\_container\_width=True)

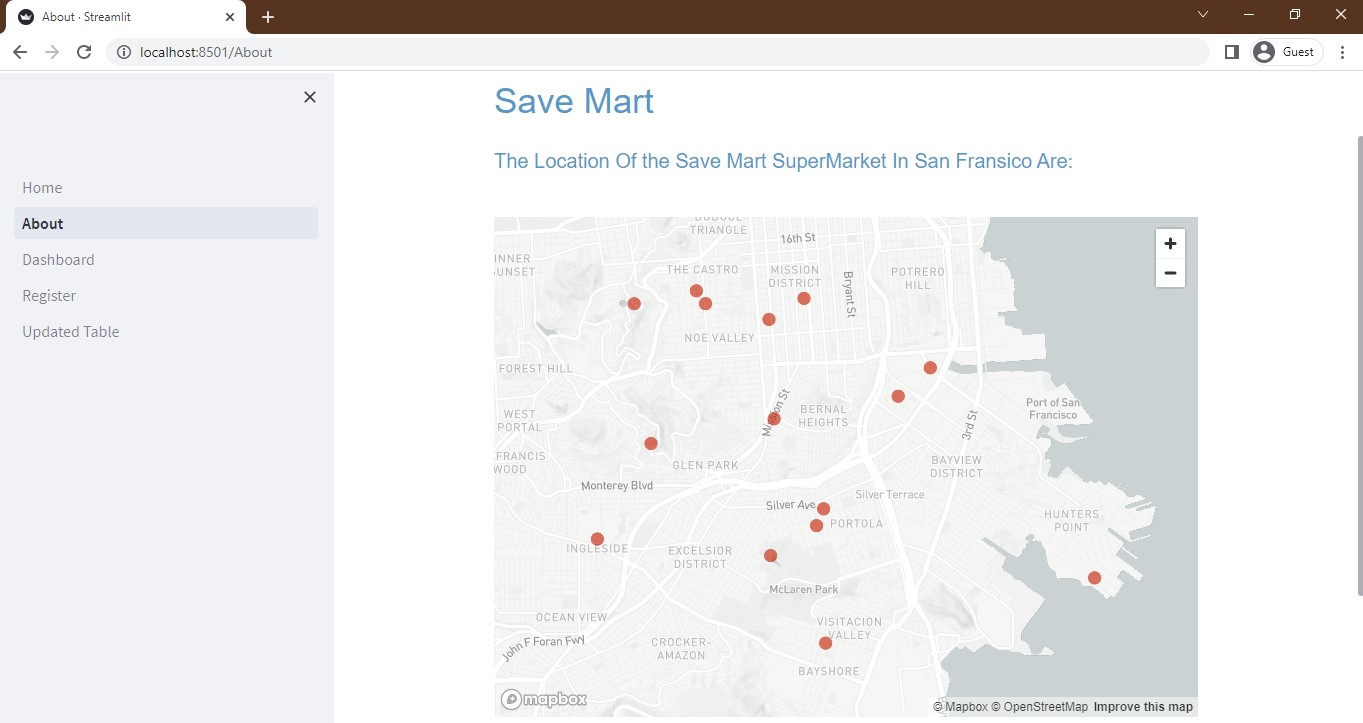
* + 1. **SCREENSHOTS**



**Figure A: Output of UI**



**Figure B: Output of Registration**



**Figure C: Output of Location Tracking**

## REFERENCES

1. Mayur Thorat, Rahul Lohakare, Prof. Nilesh N. Thorat , “Car Pooling System with SMS Alerts “ Proceeding of International Journal of Engineering Research & Technology (IJERT), 2013.
2. Shyamsundar Pralhad Magar, Vinayak Jadhav , “Ambuitec: Ambulance Booking Application for Emergency Health Response, Blood Inventory” Proceeding of Test Engineering and Management, 2020.
3. Chun-Ming Tsai, Jun-Wei Hsieh, Ming-Ching Chang, and Yu-Chen Lin, ” Driver License Field Detection using Real-Time Deep Networks “ , Proceeding of Department of Computer Science, University of Taipei, Taiwan College of Artificial Intelligence and Green Energy, National Chiao Tung University, Taiwan Department of Computer Science, University at Albany, State University of New York, USA, 2016.
4. Shervin Emami and Valentin Petruț Suciu , “Facial Recognition using OpenCV” Proceeding of University of Queensland, 2012.
5. https://pypi.org/project/face-recognition/
6. https://codingshiksha.com/tutorials/python-opencv-tutorial-to-capture-images- from-webcam-full-project/
7. https://towardsdatascience.com/camera-app-with-flask-and-opencv-bd147f6c0eec
8. https:/[/www.ge](http://www.geeksforgeeks.org/python-calculate-distance-between-two-places-)e[ksforgeeks.org/python-calculate-distance-between-two-places-](http://www.geeksforgeeks.org/python-calculate-distance-between-two-places-) using-geopy/?ref=lbp
9. https:/[/www.ge](http://www.geeksforgeeks.org/python-plotting-google-map-using-gmplot-)e[ksforgeeks.org/python-plotting-google-map-using-gmplot-](http://www.geeksforgeeks.org/python-plotting-google-map-using-gmplot-) package/