

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590018



“A MINI PROJECT REPORT”
(Subject: File Structure Laboratory with Mini Project)
(Subject Code: 18ISL67)
ON
“Vehicle Parking Management System”

Submitted in partial fulfilment for the requirements for the Award of Degree of

BACHELOR OF ENGINEERING
IN
INFORMATION SCIENCE AND ENGINEERING
BY

SAUROV GHOSH	1EP20IS079
SUBRAMANYA M	1EP20IS087
SHIVRAJ V AWARADI	1EP20IS081
VASH RAJU MOTWANI	1EP20IS104

UNDER THE GUIDANCE OF
Prof. Kemparaju N Prof. Teena KB
Prof. & Head Assistant Professor
Dept. of ISE Dept. of ISE



**EAST
POINT COLLEGE OF ENGINEERING &
TECHNOLOGY**

(Affiliated to Visvesvaraya Technological University, Belagavi) Bangalore-560049

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **Mini Project Work** entitled “**Vehicle Parking Management System**” carried out by **Mr. SAUROV GHOSH, USN 1EP20IS079 , Mr. SUBRAMANYA M, USN 1EP20IS087 , Mr. SHIVARAJ V AWARADI, USN 1EP20IS081 , Mr. VASH RAJU MOTWANI, USN 1EP20IS104,** Bonafide students of **East Point College of Engineering and Technology** in partial fulfilment for the award of **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during the year **2022-23**. It is certified that all the corrections/suggestions indicated for the Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect of File Structures Laboratory with Mini Project (18IS167) prescribed for the award of the said degree.

GUIDE

Prof. Teena KB

Asst. Professor

HOD

Prof. N Kemparaju

EPCET

PRINCIPAL

Dr. Yogesh G S

EPCET

Examiners

Name of the Examiners

Signature with date

1.

2.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders, and friends. We would like to take this opportunity to thank them all.

First and foremost, we would like to express our sincere regards and thanks to **Mr. Pramod Gowda** and **Mr. Rajiv Gowda**, CEOs, East Point Group of Institutions, Bangalore, for providing necessary infrastructure and creating good environment.

We express our gratitude to **Dr. Prakash S**, Senior Vice President, EPGI and **Dr. Yogesh G S**, Principal, EPCET who has always been a great source of inspiration.

We express our sincere regards and thanks to **Prof. N Kemparaju**, Professor and Head, Department of Information Science and Engineering, EPCET, Bangalore, for his encouragement and support.

We are grateful to acknowledge the guidance and encouragement given to us by **Prof. N Kemparaju**, Prof. & Head and **Prof. Teena KB**, Assistant Professor, Department of Information Science and Engineering, EPCET, Bangalore, who has rendered a valuable assistance.

We also extend our thanks to the entire faculty of the Department of **Information Science and Engineering, EPCET**, Bangalore, who have encouraged us throughout the course of the Project.

Last, but not the least, we would like to thank our family and friends for their inputs to improve the Project.

SAUROV GHOSH 1EP20IS079

SUBRAMANYA M 1EP20IS087

SHIVARAJ V AWARADI 1EP20IS081

VASH RAJU MOTWANI 1EP20IS104

ABSTRACT

A Parking Management System is designed to manage and solve the parking problems of a company or institute. The aim of implementing such a system is to reduce time and increase efficiency of the current parking management process. In overpopulated cosmopolitan zones, parking strategies must be well-implemented for management of vehicles. The system can manage the records of incoming and outgoing vehicles in a parking house, making it easy for the admin to retrieve data.

CHAPTER NO.	INDEX TOPICS	PAGE NO.
1	Introduction 1.1 Introduction to Project 1.2 Organization of the Report 1.3 Proposed System	1
2	Literature Survey	2
3	Requirements Specification 3.1. Introduction 3.2 System architecture 3.2 Functional Requirements 3.3 Non-Functional Requirements 3.3.1 User Interface 3.3.2 Software Requirements 3.3.3 Hardware Requirements 3.4 System model	3 - 7
4	System Design 4.1 System Architecture and Decomposition 4.2. Data Models 4.3 Algorithmic Design 4.4 User Interface Design	8 - 19
5	Software Test Specification 5.1 Goals and Major constraints 5.2 Testing Strategies 5.2.1 Unit Testing 5.2.2 Integration Testing 5.2.3 System Testing	20 - 22
	Conclusion and Future Enhancement	23
6	References	24

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
1.1	Functional Requirements	4
5.1	Test case for Login	21
5.2	Test case for Search	21
5.3	Test case for Vehicle add	21
5.5	Test case for System Testing	22
5.4	Test case for delete.py	21

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	Architecture diagram	4
3.2	Use case diagram	6
3.3	Sequence diagram	7
4.1	Class diagram	12
4.2	Data flow diagram	13
4.3	State diagram	14
4.4	Index Page	17
4.5	Login Window	18
4.6	option Window	18
4.7	Add vehicle Menu	18
4.8	Search Vehicle	19
4.9	Checkout Vehicle	19

CHAPTER 1

INTRODUCTION

1.1 Introduction to project

Vehicle parking management systems play a crucial role in efficiently managing parking spaces and ensuring smooth operations in various locations such as parking lots, garages, and public parking areas. These systems utilize file structures to organize and store information related to vehicles, parking spots, and other relevant data. In a file structure-based vehicle parking management system, data is typically stored in files and directories, creating a hierarchical organization that allows for easy access and retrieval.

1.2 Organization of Report.

The chapter 1 is the brief introduction about the project, about the organisation of the report and the proposed system is described. The chapter 2 is regarding the literature survey we have described different other similar technologies in it. The chapter 3 is regarding the requirement specification it describes about the system requirements, functional requirements, nonfunctional requirements like user interface, software requirements, and hardware requirements etc and also system model is shown. The chapter 4 is regarding the system architecture and decomposition which consists of different data models, algorithm. And finally chapter 5 is regarding goals and major constraints, testing strategies, unit testing, integration testing and system design.

1.3 Proposed System

The salient features of “Vehicle Parking Management System” are.

- **Vehicle Registration:** Streamline the process of registering vehicles entering the parking area.
- **Parking Spot Allocation:** Efficiently assign parking spots to vehicles based on availability and predefined rules, optimizing space utilization.
- **Entry and Exit Management:** Track vehicle entry and exit timestamps to accurately calculate parking duration and manage parking fees.
- **Reporting and Analytics:** Generate comprehensive reports and analytics on parking utilization, revenue, and occupancy rates for data-driven decision-making.
- **User-Friendly Interfaces:** Provide intuitive interfaces for operators and users, facilitating easy management and accessibility for various tasks within the system.

CHAPTER 2

LITERATURE SURVEY

A parking management system is essential for efficient and well-organized parking operations. By implementing such a system, parking facilities can optimize the utilization of available space, ensuring that every parking spot is efficiently allocated and utilized. This helps to minimize congestion and maximize the number of vehicles that can be accommodated, particularly in high-demand areas. Additionally, a parking management system improves the overall traffic flow within the parking area and the surrounding roads by guiding vehicles to available parking spots, reducing search time, and minimizing unnecessary circulation. This not only enhances the user experience but also contributes to smoother traffic management in the vicinity. Furthermore, the system offers added convenience to users through features such as real-time spot availability updates, reservation options, and automated payment processing, saving time and reducing frustration. Lastly, a parking management system provides valuable data and reports on parking occupancy, revenue generation, and usage patterns. This information aids in effective planning, decision-making, and future expansion of parking infrastructure to meet growing demands.

Here are some commonly used systems for vehicle parking management:

- Pay-and-Display Systems.
- Ticketing Systems.
- Barrier/Gate Systems.
- Parking Guidance Systems.
- Mobile Parking Applications.
- Valet Parking Systems.

There are several technologies available for vehicle parking management systems. Here are some commonly used technologies in this field:

Sensors: Parking management systems often utilize various types of sensors, such as ultrasonic, magnetic, or infrared sensors, to detect the presence of vehicles in parking spots. These sensors provide real-time data on occupancy, allowing for efficient spot allocation and monitoring.

License Plate Recognition (LPR): LPR technology uses cameras and optical character recognition algorithms to capture and read license plate information of vehicles entering and exiting the parking area. This enables automated vehicle identification and tracking.

Mobile Applications: Mobile apps provide a convenient platform for users to search for parking locations, reserve spots, make payments, and receive real-time updates on parking availability. Integration with GPS and mapping technologies allows for easy navigation to selected parking areas.

Internet of Things (IoT): IoT devices can be deployed in parking facilities to collect and transmit data on occupancy, traffic flow, and spot availability. This data can be analyzed to optimize parking operations and provide valuable insights for decision-making.

Smart Parking Meters: Advanced parking meters incorporate technologies such as touchscreens, contactless payment options, and connectivity to centralized parking management systems. These meters streamline payment processing and facilitate efficient revenue management.

Data Analytics: Data analytics techniques are used to process and analyze parking-related data, such as occupancy rates, payment trends, and traffic patterns. This enables better decision-making, optimization of parking operations, and predictive modeling.

Parking Guidance Systems: These systems use a combination of sensors, dynamic signage, and wayfinding technology to guide drivers to available parking spots. Real-time information on spot availability is displayed, reducing search time and congestion within parking areas.

Automated Payment Systems: Technologies such as mobile payment platforms, NFC (Near Field Communication), and RFID (Radio-Frequency Identification) enable contactless and cashless payment options for parking fees. This improves convenience and speeds up the payment process.

Security and Surveillance Systems: CCTV cameras, access control mechanisms, and remote monitoring systems enhance security in parking facilities. These technologies help deter theft, ensure user safety, and provide evidence in case of incidents.

Integration with Smart City Infrastructure: Vehicle parking management systems can be integrated with broader smart city initiatives, such as intelligent transportation systems, connected infrastructure, and data sharing platforms. This enables improved coordination and optimization of parking in the context of urban mobility.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 Introduction

We aim to become a pioneer in the vehicle rental industry by completely focusing on customers, our employees, growth, innovation and efficiency. All of these elements will drive us towards success and show us as one company that can perform and give value for money.

When it comes to cab rental services, Cool Service is the most trusted and reliable name in the travel business. The most advanced travel agents offering cab rental and car hire in India, making full use of information technology to improve the level of our efficiency. However, this is only one aspect of services. And this project continually strive to offer the best of services - both in terms of man and machine, to our clients. Moreover, this project has a fleet of cars ranging from luxury to budget cabs. While, it offers online cab hire service for corporate houses. And this project claim to offer the best of rates, which are tailor-made depending upon the facilities, availed and offer both intercity and intra-city cab facilities. All cabs have proper permits and documentation so that the clients couldn't be hassled for the lack of documents. However, this project has strategic backup system for any eventuality. Cab drivers are educated, polite, and reliable and are trained to handle acute breakdowns. The cab service includes all categories of cars from luxury to budget. Further, this project's utmost priority is quality. To achieve this, vehicles are well maintained and tested for delivering optimum and uninterrupted performance. Team of professionals in the travel business enables this system to design trips that suits to all budgets and preferences of the travelers. In addition, workforce including drivers and administrative staff are well trained to discharge their duties with a lot of efficiency.

3.2 System architecture

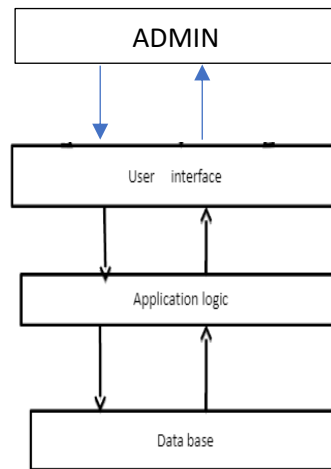


Figure 3.1: Architecture Diagram

3.2 Functional Requirements

Functional Requirements Requirement analysis is a software engineering approach that consists of a series of activities that establish the demands or conditions that must be satisfied for a new or updated product while taking into account the potential for competing requirements from different users. Functional requirements are

those that are used to demonstrate the system's internal functioning nature, as well as the system's description and explanation of each subsystem. It comprises the task that the system should accomplish, the processes involved, the data that the system should contain, and the user interfaces. The functional requirements discovered are as follows: 1) Admin Login- Admin should be able to get the access. 2) Add Vehicle. 3). Remove Vehicle- The system should be able to update the database without any further effort from the administrator whenever a new reservation or return is made. 4). view all vehicle from the interface. 5). checkout vehicle. 6) Search all Vehicle from List.

Functional Requirements: These are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. It specifies the application functionality that the developers must build into the product to enable users to accomplish their tasks. The system must allow the Admin to register for Vehicles.

3.3 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.

3.3.1 User Interface

We use tkinter for user interface in python programming languages. Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create GUI applications. Creating a GUI using tkinter is an easy task. tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

3.3.2 Software Requirements

We have used visual studio code as an IDE (integrated development environment) software to run our simple python program of ledger. And we have used some libraries such as tkinter, transactions, people etc. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

3.3.3 Hardware Requirements

The Windows 7+ operating system is used, intel x5 processor, memory size is 32 GB, available disk space is 100 GB.

3.4 System model

web-based car rental system integrated with SMS technology has a very user-friendly interface. By using this system, employees can manage bookings, payment, vehicle issues and SMS notification to the customers within a few clicks only. The new data can be added or an existed data can be edited or deleted too by administrators. Thus, there is no delay in the availability of any information, whether needed, can be captured very quickly and easily. For security purposes, all customers need to create a new account before logging in or he/she can log into the system with his/her created account before they can make a reservation for a car. Then, the customer will be notified the availability of the car reserved through SMS. This system becomes very helpful for employees, administrator and customers. Figure 1 shows the car rental system architecture for the proposed system.

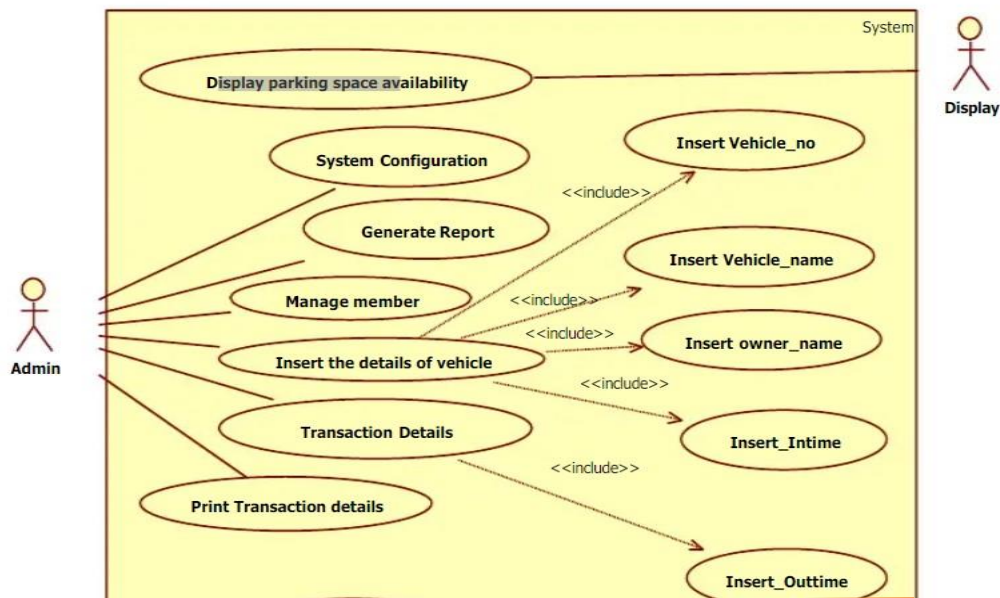


Figure 3.2: Flow chart

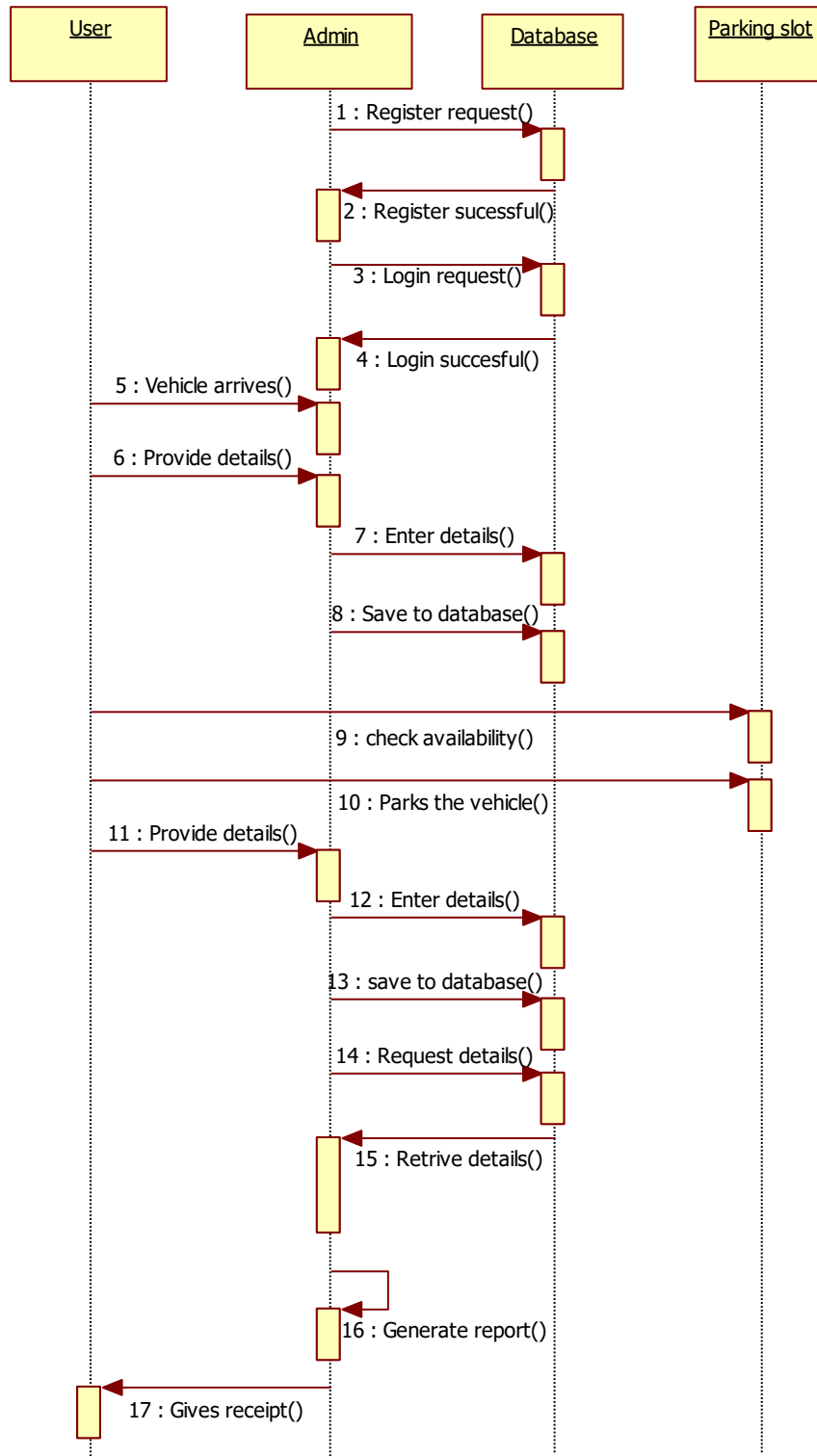


Figure 3.3: Sequence diagram

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture and Decomposition

A system architecture for a parking management system project for a file structure mini project would depend on the specific requirements and goals of the project. However, here are some general components that could be included in such a system:

1. **File:** A File to store information about parking spaces, vehicles, and transactions.
2. **User Interface:** A user interface for users to interact with the system, such as reserving parking spaces or making payments.
3. **Sensors:** Sensors to detect the presence of vehicles in parking spaces and update the database accordingly.
4. **Payment System:** A payment system to handle transactions for parking fees.
5. **Reporting:** A reporting system to generate reports on usage, revenue, and other metrics.

4.2 Data Models

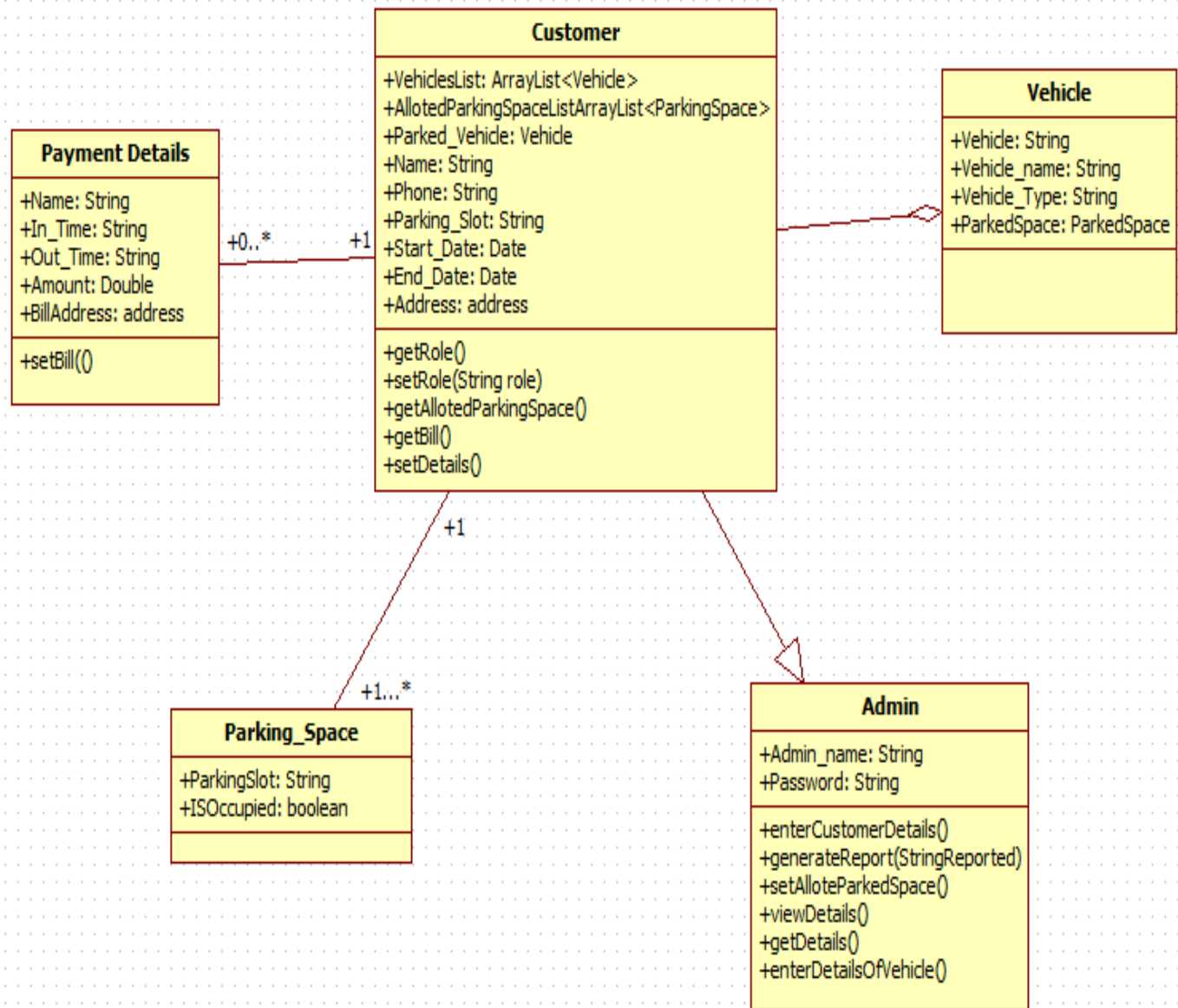


Figure 4.1: Class diagram

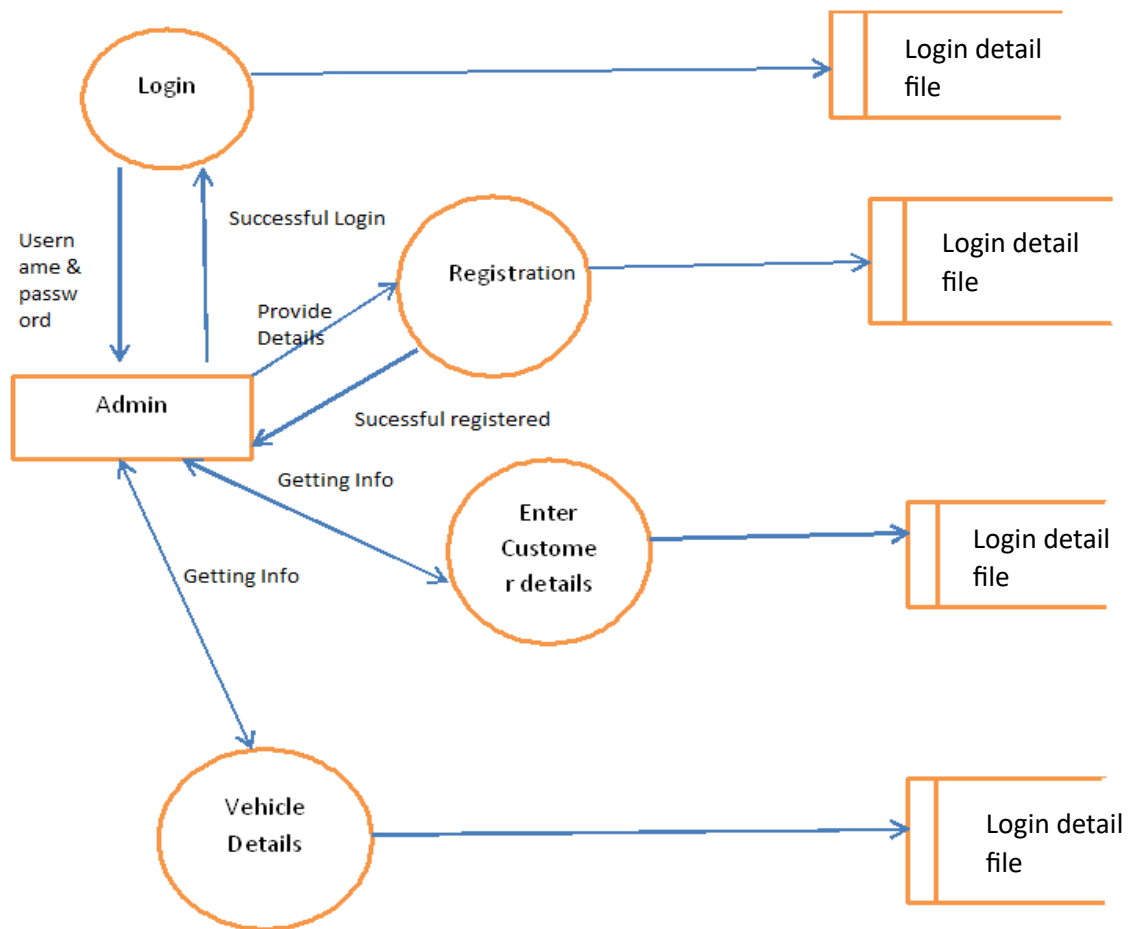


Figure 4.2: Data flow diagram

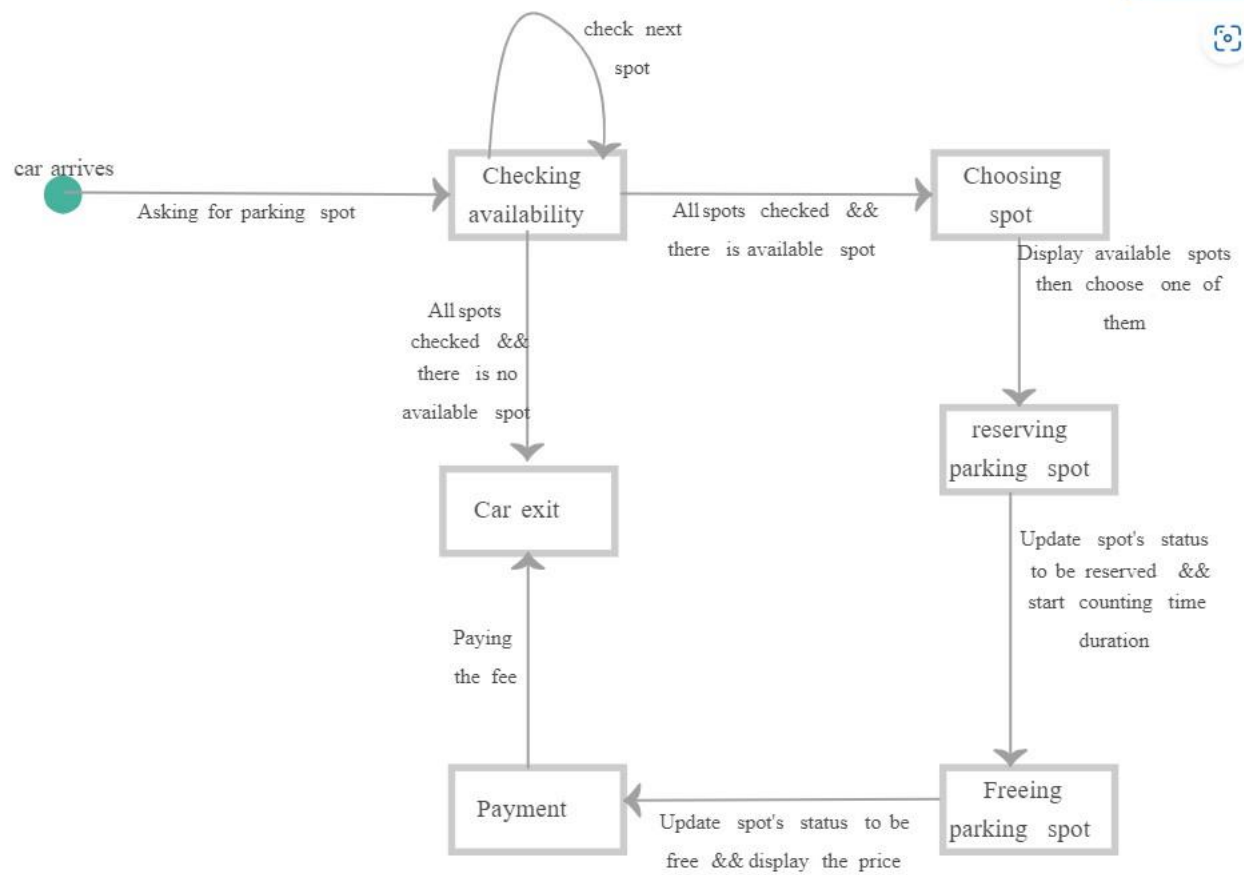


Figure 4.3: State diagram

4.3 Algorithmic Design

#Main login page

```
from tkinter import *

import os,index,datetime,sys

from PIL import Image, ImageTk

from tkinter import messagebox

def clse():

    sys.exit()

def pos():

    ret=verifier()

    if ret==0:

        h=open("admin.txt")

        lines = h.readlines()

        h.close()

        for i in lines:

            if i.find(eid.get())!=-1 and i.find(psw.get())!=-1:

                messagebox.showwarning("success","Authorization Success.")

                root.destroy()

                os.system('python option.py')

                break

            else:

                messagebox.showwarning("Warning","Employee id or password is incorrect.")

        else:
```

```
messagebox.showwarning("Warning","Type Employee id and password.")
```

```
def verifier():
```

```
    a=b=0
```

```
    if not eid.get():
```

```
        a=1
```

```
    if not psw.get():
```

```
    if __name__=="__main__":
```

```
        root=Tk()
```

```
        root.minsize(935, 455)
```

```
        root.maxsize(935, 455)
```

```
        root.title("Parking")
```

```
        root.configure(bg='#FFC0CB')
```

```
        eid=StringVar()
```

```
        psw=StringVar()
```

```
        label=Label(root,text="LOGIN",font="bold",fg="Red")
```

```
        label.place(x=450,y=50)
```

```
        label1=Label(root,text="Employee id :")
```

```
        label1.place(x=360,y=120)
```

```
        label2=Label(root,text="Password   :")
```

```
        label2.place(x=360,y=150)
```

```
        e1=Entry(root,textvariable=eid)
```

```
        e1.place(x=460,y=120)
```

```
        e2=Entry(root,show='*',textvariable=psw)
```

```
        e2.place(x=460,y=150)
```

```
b4=Button(root,text="Submit",command=pos,activebackground="pink",bg="#68BBE3",width=30)
```

```
b4.place(x=363,y=200)
```

```
b3=Button(root,text="Close",command=clse,bg="#68BBE3",activebackground="red",width=30)
```

```
b3.place(x=700,y=420)
```

```
root.mainloop()
```

RUN main event loop using mainloop()

FUNCTION admin_check():

GET entered admin ID and password from admin login form

IF admin ID and password match predefined values:

def ad():

```
if menu.get()=="TwoWheeler":
```

```
    t1=datetime.datetime.now()
```

```
    m1=t1.strftime("%Y:%m:%d")
```

```
    ls=name.get()+'|'+vno.get()+'|'+tno.get()+'|'+m1+"\n"
```

```
    p=open('twowheel.txt','a')
```

```
    p.write(ls)
```

```
    p.close()
```

```
if menu.get()=="ThreeWheeler":
```

```
    t1=datetime.datetime.now()
```

```
    m1=t1.strftime("%Y:%m:%d")
```

```
    ls=name.get()+'|'+vno.get()+'|'+tno.get()+'|'+m1+"\n"
```

```
    p=open('threewheel.txt','a')
```

```
    p.write(ls)
```

```
    p.close()
```

```
if menu.get()=="FourWheeler":  
    t1=datetime.datetime.now()  
  
    DEFINE add_check function for add car button action  
  
    RUN main event loop using mainloop()  
  
FUNCTION add_check():  
    GET entered car details from add car form  
  
    IF car registration number is empty:  
        DISPLAY error message  
  
        RETURN to add car screen  
  
    WRITE car details to Cars.txt file  
  
    DISPLAY success message  
  
    DESTROY add car window  
  
#Create a dropdown Menu  
  
drop= OptionMenu(root, menu,"TwoWheeler", "ThreeWheeler","FourWheeler","Others")  
  
drop.pack()  
  
drop.place(x=400,y=50)  
  
vno=StringVar()  
  
tno=StringVar()  
  
name=StringVar()  
  
label2=Label(root,text=" Name:")  
  
label2.place(x=300,y=120)  
  
label1=Label(root,text="Vehicle No:")  
  
label1.place(x=300,y=170)
```

```
label2=Label(root,text="Token No:")

label2.place(x=300,y=220)

e1=Entry(root,textvariable=name,width=40)

e1.place(x=420,y=120)

e1=Entry(root,textvariable=vno,width=40)

# Validate car ID and check if it is in the user's records

# Update car availability, remove record, and display messages

FUNCTION search_in():

    # Create and configure search_menu window

    # Create label, entry, and button

    # Position label, entry, and button

    search_menu.mainloop()

FUNCTION search_check():

    # Get search word from entry

    # Validate search word and perform binary search

    # Display search result or error message

# Call the Main_Menu function to start the program

Main_Menu()
```


4.4 User Interface Design

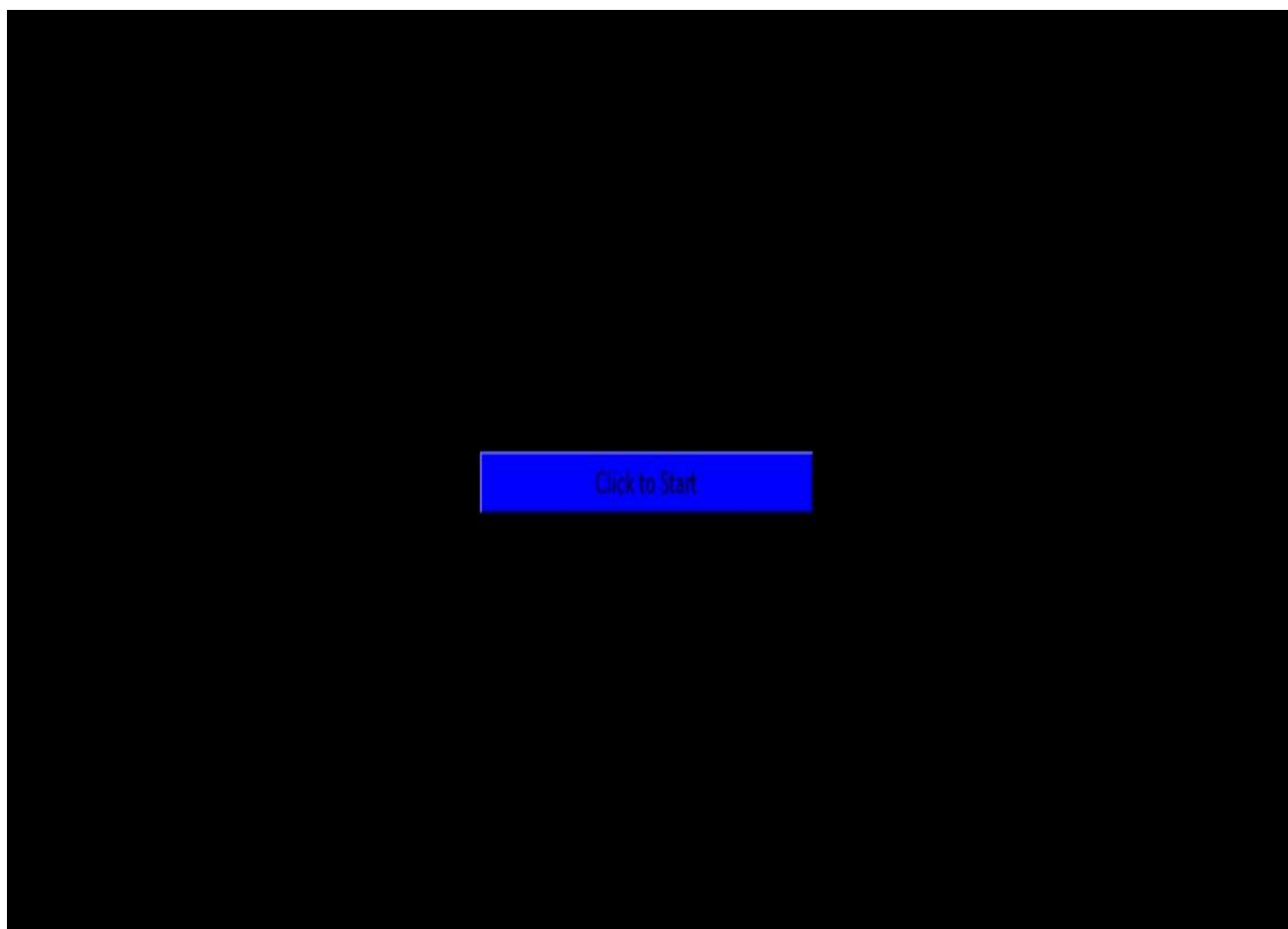


Figure 4.4: Index Page

The image shows a light pink rectangular area representing the login page. At the top center, there is a red rectangular button with the text "LOGIN" in white. Below this, there are two input fields: the first is labeled "Employee id :" and the second is labeled "Password :". Below these fields is a blue rectangular button with the text "Submit". In the bottom right corner, there is a blue rectangular button with the text "Close".

Figure 4.5: Login Page

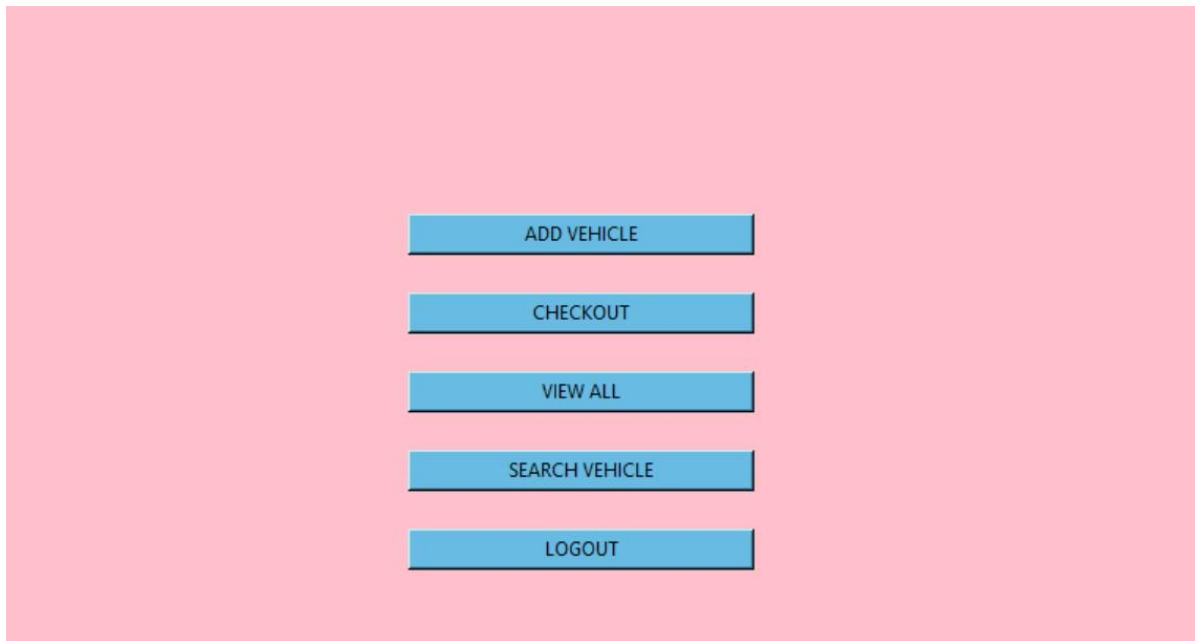


Figure 4.6: Options page

A screenshot of a web form titled 'Add Vehicle Window' on a light pink background. At the top, there is a dropdown menu with 'ThreeWheeler' selected. Below this are three input fields: 'Name:' with 'yamaha', 'Vehicle No:' with '12385', and 'Token No:' with '34'. At the bottom, there are two blue buttons: 'Submit' on the left and 'Back' on the right.

Figure 4.7: Add Vehicle Window

Name	Vehicle No	Token	Date	Vehicle Type
sharon	TN46H6754	11	2022:06:28	Other
jeevan	TN46H6776	10	2022:06:28	Four wheeler
yamaha	12385	34	2023:07:09	Three wheeler
minto	KL14J5643	5	2022:06:28	Three wheeler
saurov	1ep20is079	79	2023:07:06	Two wheeler
jobin	KL78H6858	9	2022:06:28	Two wheeler
mebin	KL78H6748	8	2022:06:28	Two wheeler

Back

Figure 4.8: View Window

CHECKOUT


ThreeWheeler

Token No: yamaha

Vehicle No: 86

Checkout

Succes


Checkout completed.

OK

Back

Figure 4.9: Checkout window

CHAPTER 5

SOFTWARE TEST SPECIFICATION

5.1 Goals and Major constraints

When we are performing testing, we do expect correct output.

5.2 Testing Strategies

5.2.1 Unit Testing

The different units/modules/functions used are.

1. Index.py
 2. Login.py
 3. Vehicleaddvehicle.py
 4. View.py
 5. Options.py
 6. Delete.py
 7. Search.py
- The Login/Register/Admin login is done with a set of functions to enter into the user interface.
 - The Admin menu is used to add/remove cars.
 - The Add Vehicle /Remove Remove menu used to add or remove Vehicle
 - Test case for Login is - id=Admin,Password=12345.
 - Test case for Search is- token no.=0007, click generate.

Table 5.1: Test case for Login.py

Name of the Module/Unit: Login				
Input	Expected Result	Actual result	Pass/Fail	Remarks
ID:Admin Password:12345	Authorization success	Authorization success	pass	pass

Table 5.2: Test case for unit search.py

Name of the Module/Unit: Search Car				
Input	Expected Result	Actual result	Pass/Fail	Remarks
Reg. No.: 079	Found +Details	Found +Details	pass	pass

Table 5.3: Test case for Vehicleadd.py

Name of the Module/Unit: add vehicle				
Input	Expected Result	Actual result	Pass/Fail	Remarks
Name:Yamaha Vehicle no:079 Token no:79	Added successfully	Added successfully	pass	pass

Table 5.4: Test case for unit delete.py

Name of the Module/Unit: Search Car				
Input	Expected Result	Actual result	Pass/Fail	Remarks
Reg. No.: 079	deleted Successfully	deleted Successfully	pass	pass

5.2.3 Integration Testing

After integration the units are also behaving the same and giving expected results.

5.2.4 System Testing

Table 5.5: Test case for System testing

Functional Requirements No.	Functional Requirements	Action	Expected Result	Actual Result	Pass/Fail	Remarks
FR1	Login	Login	Pass+ Main window	Pass+ Main window	pass	pass
FR2	Add vehicle	Add	Allocated	Allocated	pass	pass
FR3	Checkout Vehicle	Delete	Deleted	Deleted	pass	pass
FR4	Search Vehicle	Search	Vehicle no	Vehicle no+Details	pass	pass

Conclusion and Future Enhancement

This Project is minimizing the task of parking a vehicle by paying and saying some details about customer and vehicle to save data. In this the vehicle is parked as a safe and secure. This project is done as Efficient as possible

Hereby we, the Student of BE(ISE) 6th Semester concludes that the project was completely and slowly developed by me. I also conclude that this project has helped us gain more knowledge about the topic that we are indulged ourselves into “Visual Studio”, I would be glad to enhance and promote this project if given chance and help ourselves and society in the near future

The developed application is tested with sample inputs and outputs obtained in according to the requirement. Even though I have tried our level best to make it a dream project. Due to time constraints I could not add more facilities to it.

The efficiency of the developed system can be enhanced with some minor modifications. Future development can be made in proposed system by integration more services like:

- It can be implemented through web pages.
- New effective modules can be added time to time.

References

- [1] . The wiki page link: <https://en.wikipedia.org/wiki/zoomcarrental> .
- [2]. Python documentation.
- [3]. Tkinter Documentation.
- [4]. File Structures: An Object-Oriented Approach with C++ by Michael J. Folk, Bill Zoellick, and Greg.
- [5]. Data Structures and Algorithms in Python by Michael T. Goodrich, Roberto Tamassia.