

A MAJOR PROJECT REPORT ON VEHICLE RENTAL SYSTEM

Prepared By

Krishna Bajpayee

6TH SEM MCA (L.E)

BPUT REGED NO. :1925201006



**Under Supervision of Beta Centauri Pvt. Ltd
Bhubaneswar**

MR. RAKESH KUMAR BHOL

**Project Report Submitted to SEEMANTA ENGINEERING COLLEGE,
Jharpokharia, In Partial Fulfilment of 6th Semester**

**MASTER IN COMPUTER APPLICATIONS (MCA)
Of**

BPUT ODISHA 2021

MAJOR PROJECT REPORT
ON
“VEHICLE RENTAL SYSTEM”

(USING PYTHON DJANGO & SQLITE)

*A report submitted in partial fulfillment of the requirements for the award of
degree in*

MASTER
IN
COMPUTER APPLICATIONS

Submitted by:

KRISHNA BAJPAYEE

REGD. NO. : 1925201006

JAGANNATH PATRA

REGD. NO. : 1925201005

Session: 2019-2021

Under Supervision of Beta Centauri Pvt. Ltd
Bhubaneswar

MR. RAKESH KUMAR BHOL

(PROJECT GUIDE)



DEPARTMENT OF COMPUTER APPLICATIONS
SEEMANTA ENGINEERING COLLEGE

Approved by AICTE, permanently affiliated to BPUT, Rourkela
MAYURBHANJ, ODISHA

DECLARATION



We do hereby declare that the project report entitled “**Vehicle Rental System**” submitted to “**Seemanta Engineering College, Jharpokharia**” for the award of the degree of **MASTER IN COMPUTER APPLICATIONS (MCA)**, is an authentic and original work carried out by us at **BETA CENTAURI PVT. LTD.** under the guidance of **Mr. Rakesh Kumar Bhol** (Software Trainer in Beta Centauri Pvt. Ltd.).

Krishna Bajpayee

REGD. NO.: 1925201006

ROLL NO.: 19MCAD-12

Ref. No: BCPL/PROJ/ 112242

Date: 10th April 2021

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr. Krishna Bajpayee-[Registration No. 1925201006], a student of MCA, Seemanta Engineering College(SEC), Mayurbhanj under Biju Pattnaik University of Technology, has successfully completed **Internship Program**(From 8th Oct 2020 to 06th April 2021) **at Beta Centauri Pvt.Ltd.** in **Vehicle Rental System** using **Python with Django** Technology under the guidance and supervision of trainer/mentor Rakesh Kumar Bhol.

During the internship period, he was found punctual, hardworking and inquisitive.



Rakesh Kumar Bhol

Software Trainer

Beta Centauri Pvt. Ltd

Contact No : 9777552206 | **Email ID :** admin@betacentauri.in | **Website :** www.betacentauri.in

Office Address: Balarampur, Ghadiamal, Rajnagar, Kendrapara, Pin-754248

CIN : U72900OR2020PTC033243

CERTIFICATE



This is to certify that, the study entitled “**VEHICLE RENTAL SYSTEM**” is being submitted by **JAGANNATH PATRA** and **KRISHNA BAJPAYEE** in practical fulfillment of the requirement for **THE MASTER IN COMPUTER APPLICATION** of Biju Pattnaik University of Technology, is a record of benefited work carried out by them under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of External

**Signature Of HOD
(MCA)**

ACKNOWLEDGEMENT



We express our thanks and gratitude to Almighty GOD, our Parents and other family members and friends without whose unconstrained support; we could not make this career in Master in Computer Application. We are grateful to our project guide, **Mr. Rakesh Kumar Bhol** (Software Trainer in Beta Centauri Pvt.Ltd.). for his constant motivation and valuable help throughout the project work. We express our gratitude to **Mrs. Madhusmita Panda, HOD of Dept. Of MCA**, for her valuable suggestions and advices throughout the MCA course. We also extend our thanks to other faculties for their Co-operation during our course. Finally, we would like to thank all our well-wishers for their co-operation to complete this project.

Krishna Bajpayee

ROLL NO.: 19MCAD-12

REGD. NO.: 1925201006

ABSTRACT



Our Aim is to design and create a data management System for a Vehicle Rental Company. This enables admin can rent a vehicle that can be used by a customer. By paying the money during a Specified Period of time. This system increases customer retention and simplify vehicle and staff Management in an efficient way.

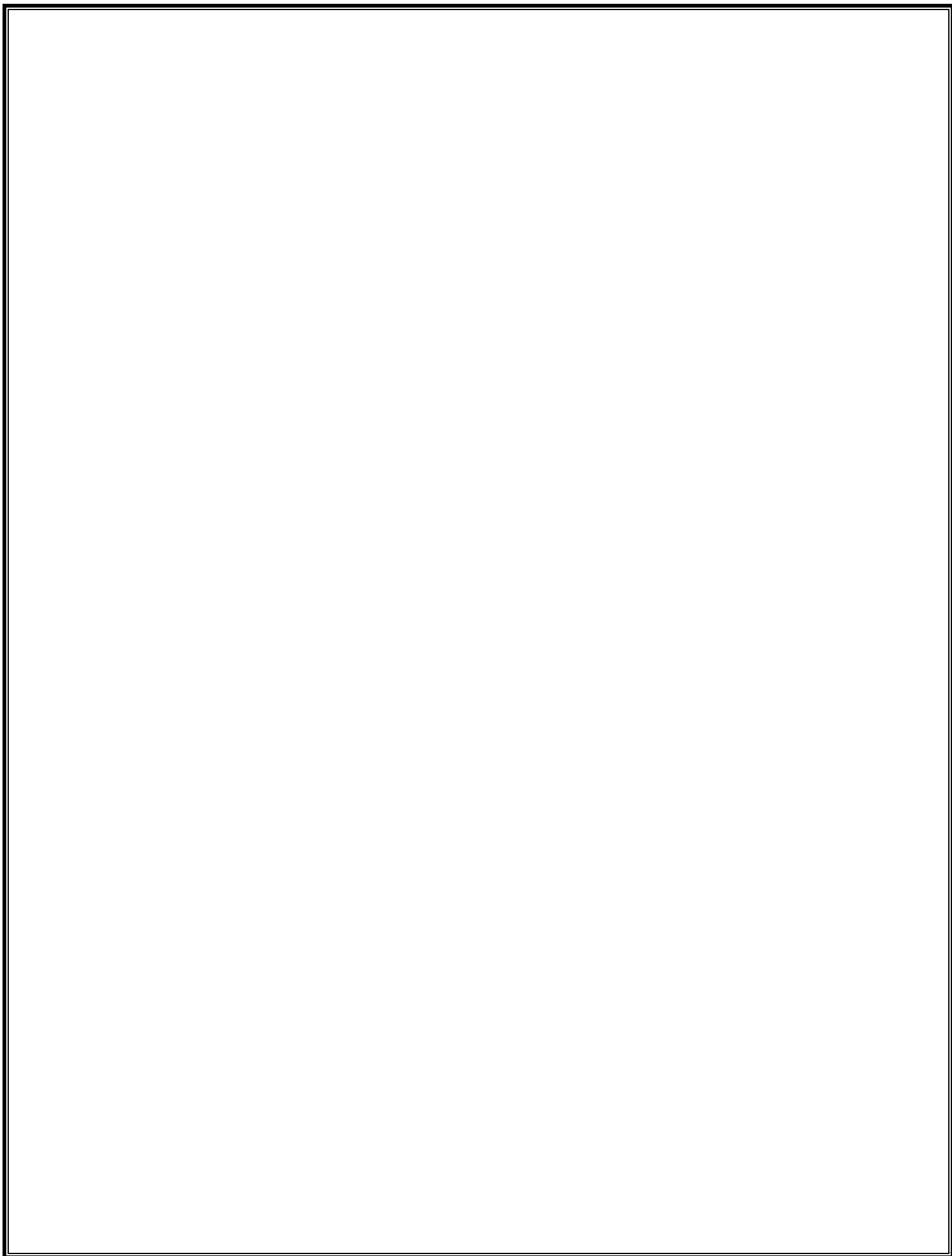
This software Vehicle Rental System has a very user friendly interface. Thus the users will feel very easy to work on it. By using this system admin can manage their rental, payment, employment issues and vehicle issues such as and insurance. The vehicle information can be added to the system. Or existed vehicle information can be edited or deleted too by Administrator. The transaction reports of the vehicle rental system can be retrieved by the admin, when its required. Thus, there is no delay in the availability of any vehicle information, whenever needed, vehicle information can be Captured very quickly and easily.

The customers can also use the system to get vehicle rent. The customer should create a new account(register) before logging in or he / she can log into the System with his/her created account. Then he/she can view the available vehicles in a branch and make a reservation for a Vehicle. This system will helpful to the admin as well as to the customer also.

CONTENT

Chapters	Page No.
1. INTRODUCTION	1
1.1 Aim of the project	
1.2 Scope of the project	
2. PROBLEM ANALYSIS	2
3. FEASIBILITY STUDY	3-4
3.1 Technical Feasibility	
3.2 Operational Feasibility	
3.3 Economic Feasibility	
3.4 Management Feasibility	
3.5 Social Feasibility	
4. PROJECT MANAGEMENT	5-8
4.1 Project Planning & Scheduling	
4.2 Methodology	
4.2.1 Project Management Life Cycle	
4.2.2 Project Plan	
4.2.3 Schedule Representation	
4.3 Risk Management	
4.4 Software Requirements	
4.5 Hardware Requirements	
5. SYSTEM DESIGN	9-23
5.1 UML Diagrams	
5.2 Data Flow Diagram	
5.3 E-R Diagram	
5.4 Database Design	

6. CODING AND IMPLEMENTATION	24-31
6.1 Technology Used	
6.1.1 HTML	
6.1.2 CSS	
6.1.3 Python	
6.1.4 Django	
6.1.5 Database	
8. SNAPSHOTS OF SCREENS	32-40
9. CODING	41-67
9. TESTING	68-69
7.1 Integration Testing	
7.2 Unit Testing	
7.3 System Testing	
7.4 Acceptance Testing	
7.5 Recovery Testing	
7.6 Functional Testing	
7.7 Hardware/Software Testing	
7.8 Security Testing	
9. CONCLUSION	70
10. REFERENCE	71



INTRODUCTION

This is an online vehicle rental system application software project that serves the functionality of an event manager. The system allows only registered users to login and new users are allowed to register on the application. This is a web application developed in **HTML, CSS, Python** language, **Django** framework and **SQLite** database. This project provides most of the basis functionality required for booking vehicle on rent. It allows user to book vehicles from a list of vehicles. Once user enters to this software to book vehicle, it shows all the details of vehicles and provides all options to choose. After users book vehicle, he/she can track his/her booking status on tracking criteria. As well as a vehicle owner can add vehicle after registering to this project. All these data are logged in the database and the user is given a receipt number for his booking. These data are then sent to administrator (website owner) and they interact with the client as per their requirements and their contact data stored in the database.

1.1 AIM OF THE PROJECT:

This project is aimed at developing an online application for those users who want to book a vehicle on rent. **Vehicle Rental System (VRS)** is a software system for adding and booking vehicles online for the vehicle owner and the users respectively. This system is an online application that can be accessed throughout the organization and others as well as proper login provided.

1.2 SCOPE OF THE PROJECT:

The system scope includes the following:

Vehicle owner logging in will have feature to add vehicles, manage them and delete their vehicles. The same features to Vehicle users logging in will have feature to book vehicles on rent, track their status and ride them. All these can be done easily by this software.

PROBLEM ANALYSIS

The Existing System is a computerized system but is developed on Lotus. It doesn't provide multiple user accessibility and also doesn't have different user privileges. So the system is not accessible for all the employees of the organization.

Since Vehicle is associated with the lives of common people and their day-to-day routines so I decided to work on this project. The purpose of this project is to automate or make online, the process of day-to-day activities like registering new customers, registering of new vehicle owner, booking vehicles on rent, adding new vehicles, and finally give up the vehicles. I have tried my best to make the complicated process Vehicle Rental System as simple as possible using Structured & Modular technique & Menu oriented interface. I have tried to design the software in such a way that user may not have any difficulty in using this package & further expansion is possible without much effort. Even though I cannot claim that this work to be entirely exhaustive. I am confident that this software package can be readily used by non-programming personal avoiding human handled chance of error.

FEASIBILITY STUDY

TECHNICAL FEASIBILITY:

This is concerned with specifying equipment and software that will successfully satisfy the user requirement; the technical needs of the system may vary considerably, but might include: The facility to produce outputs in a given time:

1. Response time under conditions.
2. Ability to process a certain volume of transaction at a particular seep.
3. Facility to communicate data to distant location.

OPERATIONAL FEASIBILITY:

It is mainly related to human organization and political aspects. The points to be considered are:

1. What changes will be brought with the system?
2. What organizational structures are distributed?
3. What new skills will be required? Do the existing staff members have these skills? If not, can then the trained due course of time.

ECONOMIC FEASIBILITY:

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More frequently known as cost/benefit system and compare them with costs. If benefits outweigh costs, a decision is taken to design and implement the system.

MANAGEMENT FEASIBILITY:

It is a determination of whether a proposed project will be acceptable to management. If does not accept a project of gives a negligible support to it; the analyst will tend to view the project as a no feasible one.

SOCIAL FEASIBILITY:

Social feasibility is a determination of whether the project will be acceptable to the people or not. This determination typically examines the probability of the project accepted by the group directly affected by the proposed system change.

PROJECT MANAGEMENT

PROJECT PLANNING AND SCHEDULING:

Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment. Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the duration for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path.

METHODOLOGY:

We have used Iterative and Incremental Development Model (IID) for our project development. This development approach is also referred to as Iterative Waterfall Development approach. Iterative and Incremental Development is a software development process developed in response to the more traditional waterfall model. This model is designed to take care of such big project. The large and complicate project chiefly demand better development and testing procedure. The waterfall model is well known for its repeated testing process. Hence I choose the waterfall model for developing my software.

Some Advantages of Waterfall Model:

- Simple and easy to understand and use.
- Easy to manage due to the rigidity of the model.
- Phases are processed and completed one at a time
- Works well for smaller projects where requirements are very well understood.

Project Management Life Cycle:

The Project Management Life Cycle has four phases. Each project life cycle phase is described along with the tasks need to complete it

The four phases are

1. Initiation
2. Planning
3. Execution
4. Closure.

Project Plan:

Once we examine that the project is feasible, I undertake project planning. The table below describes how we planned my project.

Table 2.1 Project Plan

SL No.	Task Name	Duration	Start Date	Finish Date
1	Planning	16 days	20/01/2021	05/02/2021
2	Design	24 days	03/02/2021	27/02/2021
3	Coding	32 days	28/02/2021	01/04/2021
4	Testing	8 days	02/04/2021	09/04/2021

Schedule Representation:

Scheduling the project tasks is an important project planning activity. It involves deciding which tasks would be taken up when. In order to schedule the project activities, a software project manager needs to do the following this rules.

RISK MANAGEMENT:

Software Risk Management is a proactive approach for minimizing the uncertainty and potential loss associated with a project. Some categories of risk include product size, business impact, customer-related, process, technology, development environment, staffing (size and experience), schedule, and cost. Risk Management is a practice with processes, methods, and tools for managing risks in a project?

Risk identification is a systematic attempt to specify threats to the project plan. By identifying known and predictable risks, we can take a first step toward avoiding them when possible and controlling them when necessary. To perform the risk identification, we categorized the risk into different categories as:

1. Project Risk
2. Technical Risk
3. Business Risk
4. Known Risk
5. Predictable Risk

SOFTWARE REQUIREMENTS:

Platform	: Windows 10, Ubuntu
Language	: Python, HTML, CSS, JavaScript
Framework	: Django
Database	: SQLite3
Web Server	: Django Default Web Server

HARDWARE REQUIREMENTS:

Processor : Intel Core i3

Hard Disk : 500 Gb

RAM : 4 Gb

SYSTEM DESIGN

UML DIAGRAMS:

Unified Modelling Language:

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- **User Model View**
- **Structural Model View**
- **Behavioural Model View**
- **Implementation Model View**
- **Environmental Model View**

DATA FLOW DIAGRAM:

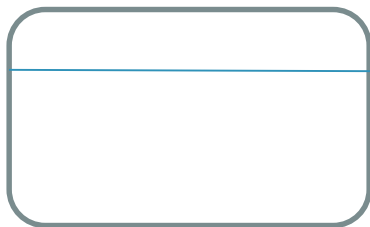
A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data input and output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams.

The physical data flow diagrams show the actual implementation and movement of data between people, departments and workstations. A full descriptions of a system actually consists of a set of data flow diagrams. Using three familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose.

DFD SYMBOLS:

In the DFD, there are four symbols

- A square defines a source or destination of system data.
- An arrow identifies data flow. It is the pipeline through which the information flows.
- A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows
- An open rectangle is a data store, data at rest or a temporary repository of data.



Process that transforms Data Flow.



Source or Destination of Data



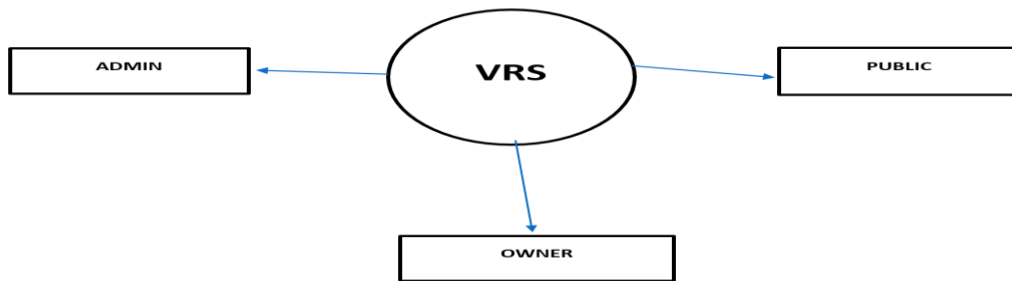
Data Flow



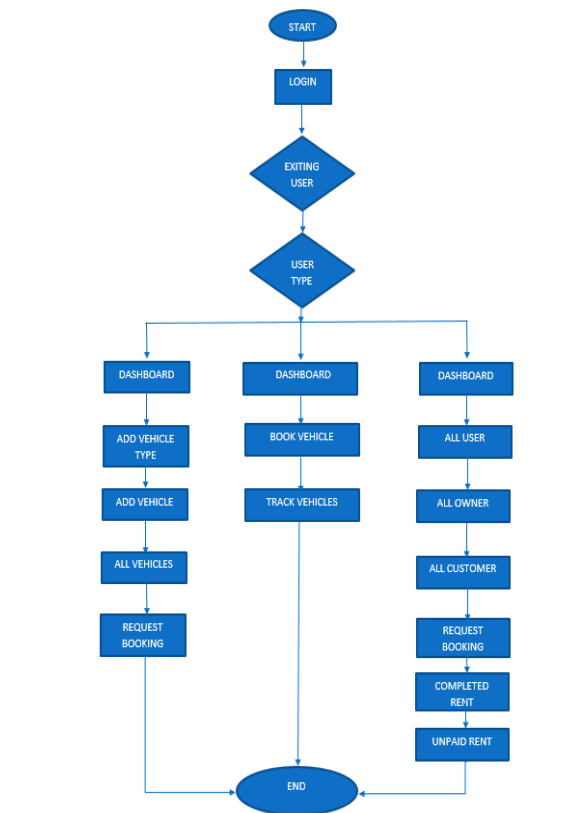


Data Store

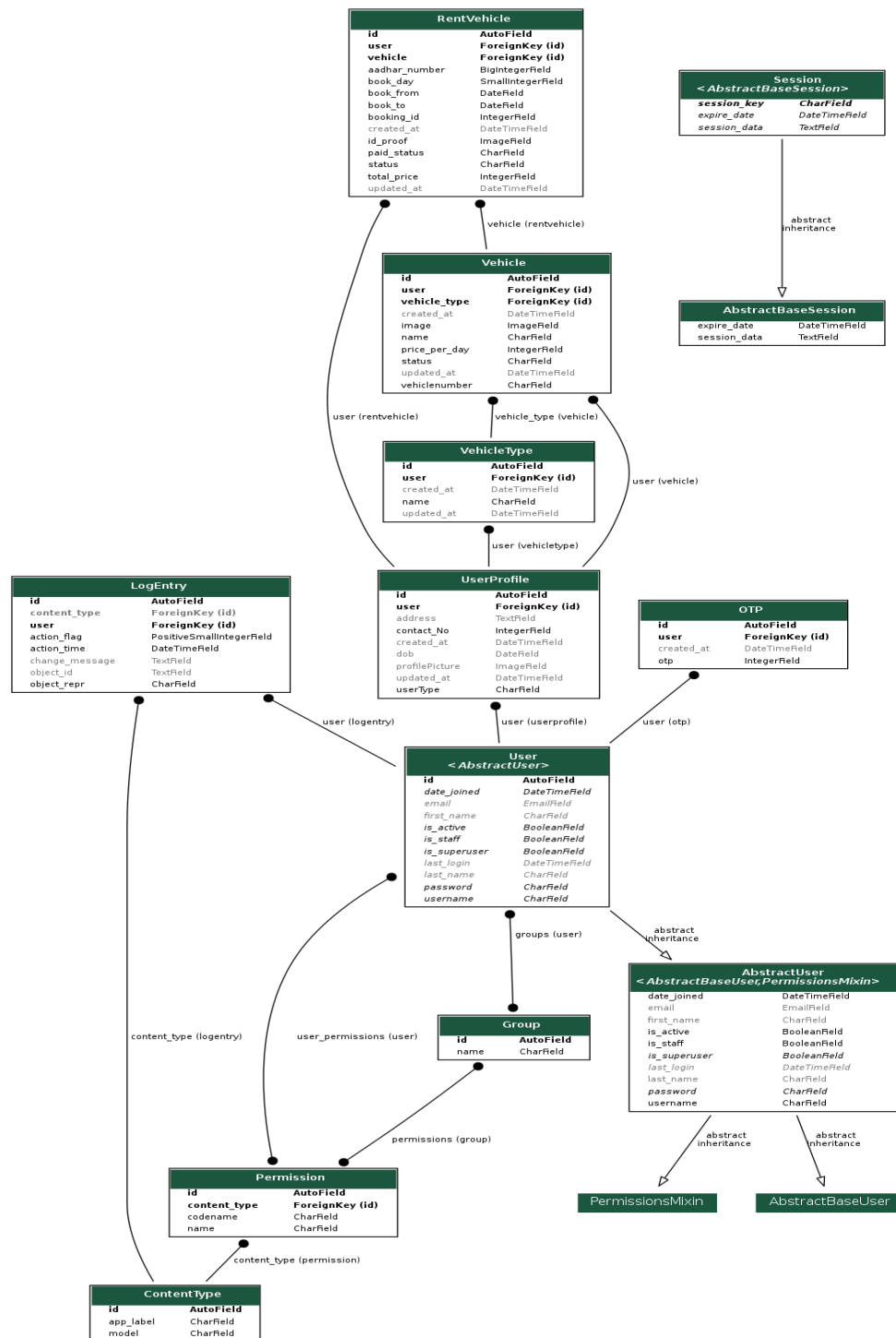
Zero Level DFD:



Final Level DFD:



E-R DIAGRM:

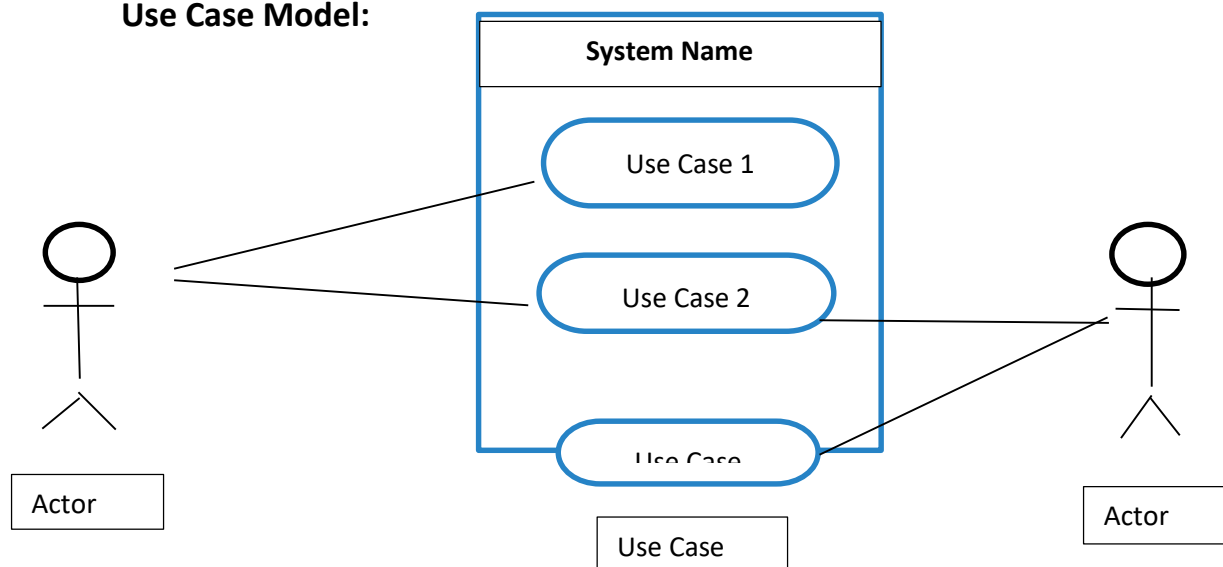


UML DIAGRAMS:

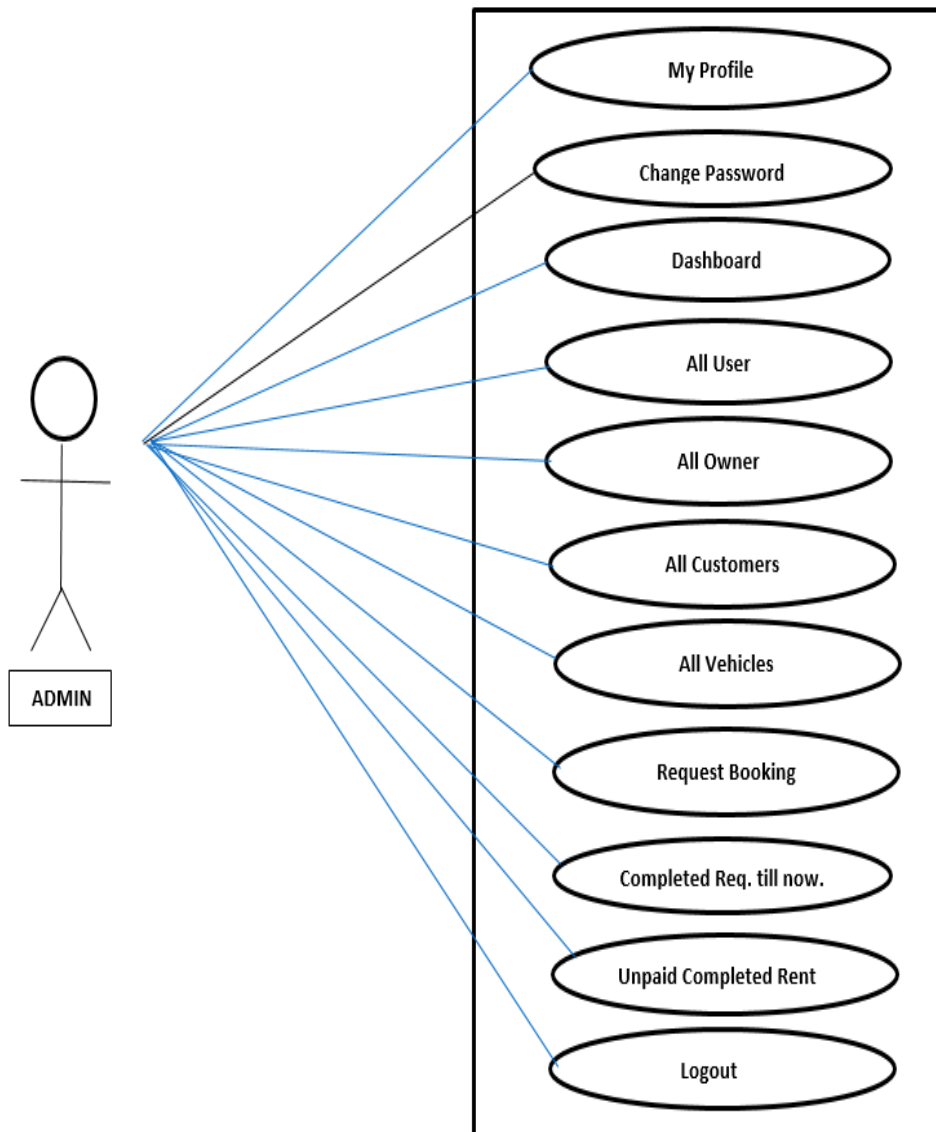
Use Case Diagram:

- The unified modelling language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.
- A UML is represented using five different views that describe the system from distinctly perspective. Each view is defined by a set of diagram, which is a follows.
 - User Model View
 - Structural Model View
 - Behavioral Model View
 - Implementation Model View
 - Environmental Model View

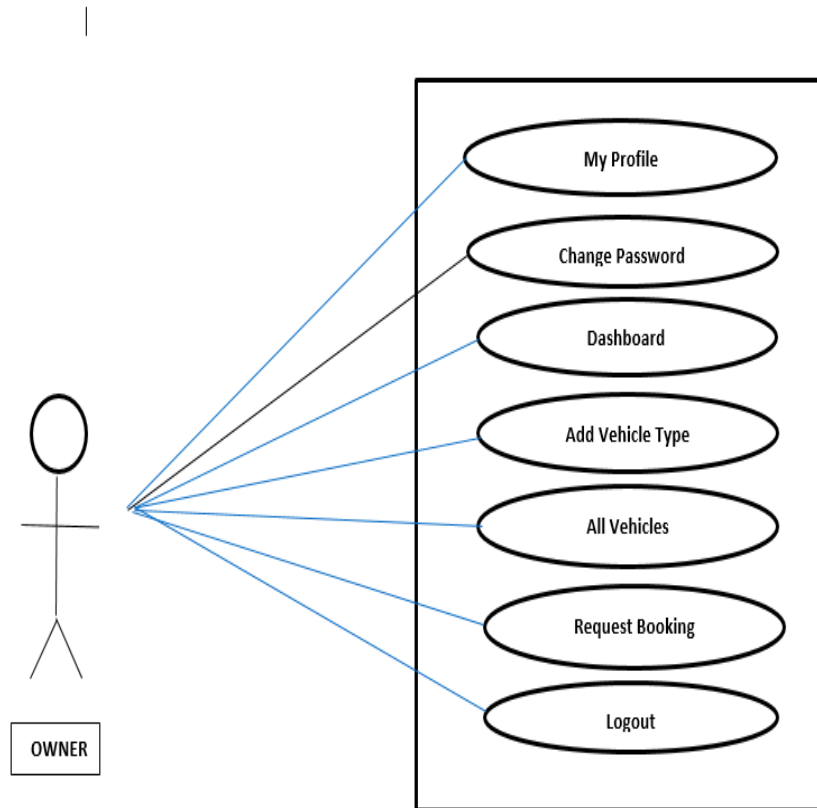
Use Case Model:



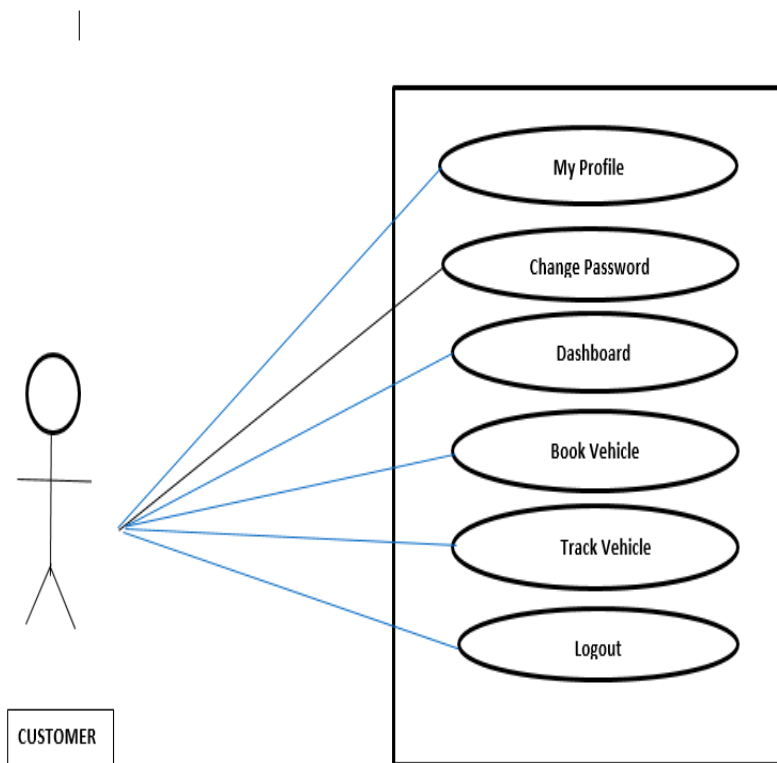
Use Case Diagrams for Admin:



Use Case Diagrams for Owner:



Use Case Diagrams for Customer:



5.4 DATABASE DESIGN:

Table 1: Admin_otp

Column Name	Data Type	Size	Schema
id	integer	50	NOT NULL
otp	integer	50	NOT NULL
Created_at	datetime	50	NOT NULL
User_id	integer	50	NOT NULL

Table 2: Admin_rentvehicle

Column Name	Data Type	Size	Schema
id	Integer	50	NOT NULL
Booking_id	Integer	50	NOT NULL
Aadhar_number	Bigint	50	NOT NULL
Id_proof	Varchar	100	NOT NULL
status	Varchar	15	NOT NULL
created_at	Datetime	50	NOT NULL
Updated_at	Datetime	50	NOT NULL
User_id	Integer	50	NOT NULL
Vehicle_id	Integer	50	NOT NULL
Book_day	Smallint	50	NOT NULL

Paid_status	Varchar	15	NOT NULL
Total_price	Integer	50	NOT NULL
Book_from	Date	50	NOT NULL
Book_to	date	50	NOT NULL

Table 3: Admin_userprofile

Column Name	Data Type	Size	Schema
Id	Integer	50	NOT NULL
profilePicture	varchar	100	NOT NULL
Contact_no	Integer	50	NOT NULL
Dob	Date	50	NOT NULL
Address	Text	50	NOT NULL
userType	varchar	15	NOT NULL
Created_at	Datetime	50	NOT NULL
Updated_at	Datetime	50	NOT NULL
User_id	integer	50	NOT NULL

Table 4: Admin_vehicle

Column Name	Data Type	Size	Schema
Id	Integer	50	NOT NULL
Vehiclenumber	Varchar	255	NOT NULL
Name	Varchar	255	NOT NULL

Image	Varchar	100	NOT NULL
status	Varchar	15	NOT NULL
Created_at	Datetime	50	NOT NULL
Updated_at	Datetime	50	NOT NULL
User_id	Integer	50	NOT NULL
Vehicle_type_id	Integer	50	NOT NULL
Price_per_day	Integer	50	NOT NULL

Table 5: Admin_vehicletype

Column Name	Data Type	Size	Schema
Id	Integer	50	NOT NULL
Name	Varchar	100	NOT NULL
Created_at	Datetime	50	NOT NULL
Updated_at	Datetime	50	NOT NULL
User_id	integer	50	NOT NULL

Table 6: Auth_group

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
name	varchar	150	NOT NULL

Table 7: Auth_group_permissions

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
Group_id	Integer	50	NOT NULL
Permission_id	integer	50	NOT NULL

Table 8: Auth_permissions

Column Name	Data Type	Size	Schema
id	integer	50	NOT NULL
Content_type_id	integer	50	NOT NULL
Codename	Varchar	100	NOT NULL
Name	varchar	255	NOT NULL

Table 9: Auth_user

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
Password	Varchar	128	NOT NULL
Last_login	Datetime	50	NOT NULL
Is_superuser	Bool	50	NOT NULL
Username	Varchar	150	NOT NULL

Last_name	Varchar	150	NOT NULL
email	Varchar	254	NOT NULL
Is_staff	Bool	50	NOT NULL
Is_active	Bool	50	NOT NULL
Date_joined	Datetime	50	NOT NULL
First_name	varchar	150	NOT NULL

Table 10: Auth_user_group

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
User_id	Integer	50	NOT NULL
Group_id	Integer	50	NOT NULL

Table 11: Auth_user_user_permission

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
User_id	Integer	50	NOT NULL
Permission_id	Integer	50	NOT NULL

Table 12: Django_admin_log

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
Action_time	Datetime	50	NOT NULL
Object_id	Text	50	NOT NULL
Object_repr	Varchar	200	NOT NULL
Change_message	Text	50	NOT NULL
Content_type_id	Integer	50	NOT NULL
User_id	Integer	50	NOT NULL
Action_flag	Smallint unsigned	50	NOT NULL

Table 13: Django_content_type

Column Name	Data Type	Size	Schema
Id	integer	50	NOT NULL
App_label	Varchar	100	NOT NULL
model	Varchar	100	NOT NULL

Table 14: Django_migration

Column Name	Data Type	Size	Schema
Id	Integer	50	NOT NULL
app	varchar	225	NOT NULL
Name	varchar	225	NOT NULL

applied	Datetime	50	NOT NULL
---------	----------	----	----------

Table 15: Django_session

Column Name	Data Type	Size	Schema
Session_key	varchar	40	NOT NULL
Session_data	text	50	NOT NULL
Expire_date	datetime	50	NOT NULL

TECHNOLOGY USED

HTML:

HTML, an initialism of Hypertext Markup Language, is the predominant markup language for web pages. It provides a means to describe the structures of text-based information in a document by denoting certain text as headings, paragraphs, lists, and so on – and to supplement that text with interactive forms, embedded images, and other objects. HTML is written in the form of labels (known as tags), surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code which can the behavior of web browsers and other HTML processors.

Hypertext Markup Language:

HTML is also often used to refer to content of the MIME type text/html or even more broadly as a generic term for HTML whether in its XML descended form or its form descended directly from SGML.

Hypertext Markup Language, the language of the World Wide Web, allows users to produces Web pages that include text, graphics and pointer to other Web pages.

HTML is not a programming language but it is an application of ISO Standard 8879, SGML, but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

ADVANTAGES:

- ❖ A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- ❖ HTML is platform in dependent.
- ❖ HTML tags are not case-sensitive.

CSS:

CSS is used to control the style of a web document in a simple and easy way.

CSS is the acronym for "**Cascading Style Sheet**". This tutorial covers both the versions CSS1, CSS2 and CSS3, and gives a complete understanding of CSS, starting from its basics to advanced concepts.

Cascading Style Sheets, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.

CSS is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning CSS:

- **Create Stunning Web site** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
- **Become a web designer** - If you want to start a career as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like JavaScript, php, or angular are become easier to understand.

Applications of CSS:

- **CSS saves time** - You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- **Pages load faster** - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** - To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** - CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** - Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- **Global web standards** - Now HTML attributes are being deprecated and it is being recommended to use CSS. So it's a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

Advantages of CSS:

- **CSS saves time** – You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- **Pages load faster** – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

- **Global web standards** – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it's a good idea to start using CSS in all the HTML pages to make them compatible to future browsers

Python:

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL).

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

ADVANTAGES:

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Characteristics of Python:

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python:

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintaining.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Django:

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Django makes it easier to build better web apps quickly and with less code.

Django is a registered trademark of the Django Software Foundation, and is licensed under BSD License

History of Django:

- **2003** – Started by Adrian Holovaty and Simon Willison as an internal project at the Lawrence Journal-World newspaper.
- **2005** – Released July 2005 and named it Django, after the jazz guitarist Django Reinhardt.
- **2005** – Mature enough to handle several high-traffic sites.
- **Current** – Django is now an open source project with contributors across the world.

Advantages of Django:

- **Object-Relational Mapping (ORM) Support** – Django provides a bridge between the data model and the database engine, and supports a large set of database systems including MySQL, Oracle, Postgres, etc. Django also supports NoSQL database through Django non-rel fork. For now, the only NoSQL databases supported are MongoDB and Google App Engine.
- **Multilingual Support** – Django supports multilingual websites through its built-in internationalization system. So you can develop your website, which would support multiple languages.
- **Framework Support** – Django has built-in support for Ajax, RSS, Caching and various other frameworks.
- **Administration GUI** – Django provides a nice ready-to-use user interface for administrative activities.
- **Development Environment** – Django comes with a lightweight web server to facilitate end-to-end application development and testing.

DATABASE:

SQLite:

SQLite is an in-process library that implements a self-contained, server less, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

- SQLite does not require a separate server process or system to operate (server less).
- SQLite comes with zero-configuration, which means no setup or administration needed.
- A complete SQLite database is stored in a single cross-platform disk file.
- SQLite is very small and light weight, less than 400KiB fully configured or less than 250KiB with optional features omitted.
- SQLite is self-contained, which means no external dependencies.
- SQLite transactions are fully ACID-compliant, allowing safe access from multiple processes or threads.
- SQLite supports most of the query language features found in SQL92 (SQL2) standard.
- SQLite is written in ANSI-C and provides simple and easy-to-use API.
- SQLite is available on UNIX (Linux, Mac OS-X, Android, iOS) and Windows (Win32, WinCE, WinRT).

SQLite A Brief History:

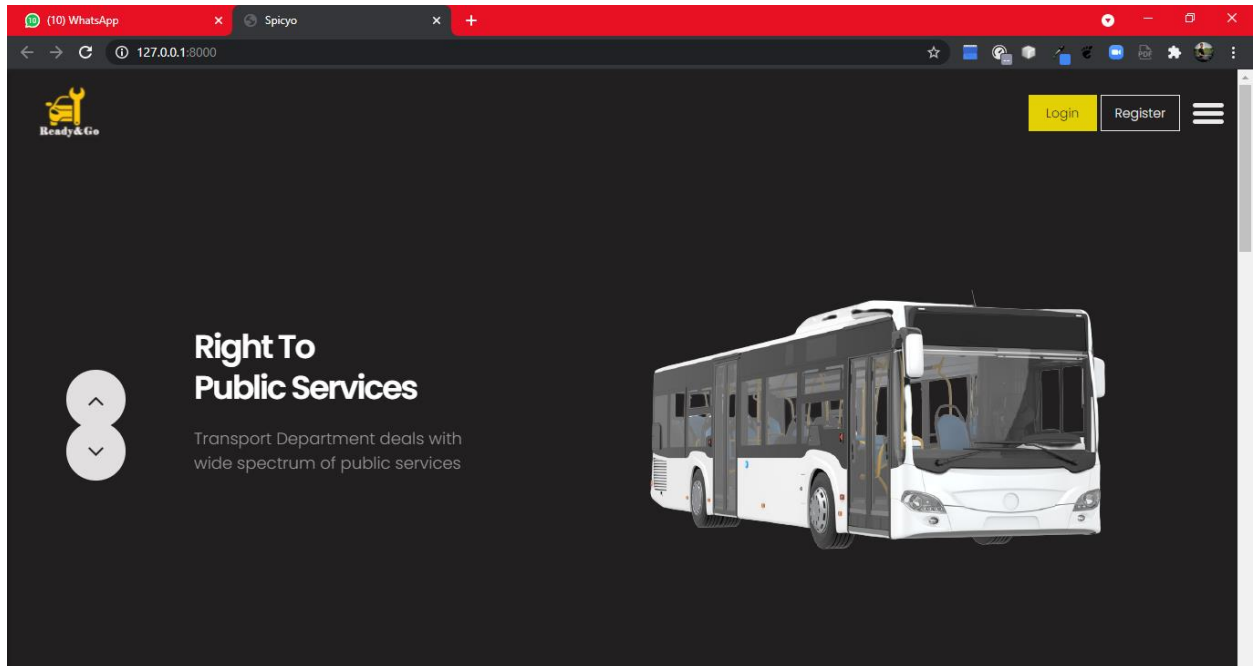
- 2000 - D. Richard Hipp designed SQLite for the purpose of no administration required for operating a program.
- 2000 - In August, SQLite 1.0 released with GNU Database Manager.
- 2011 - Hipp announced to add UNQL interface to SQLite DB and to develop UNQLite (Document oriented database).

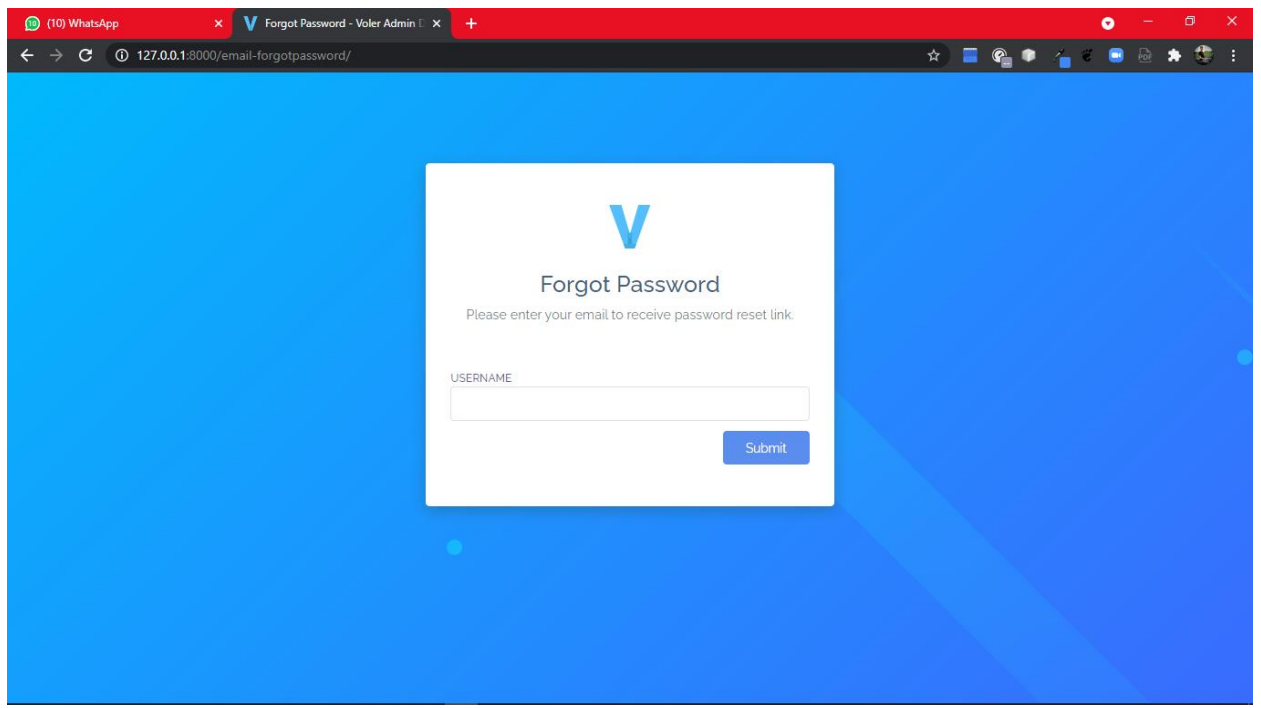
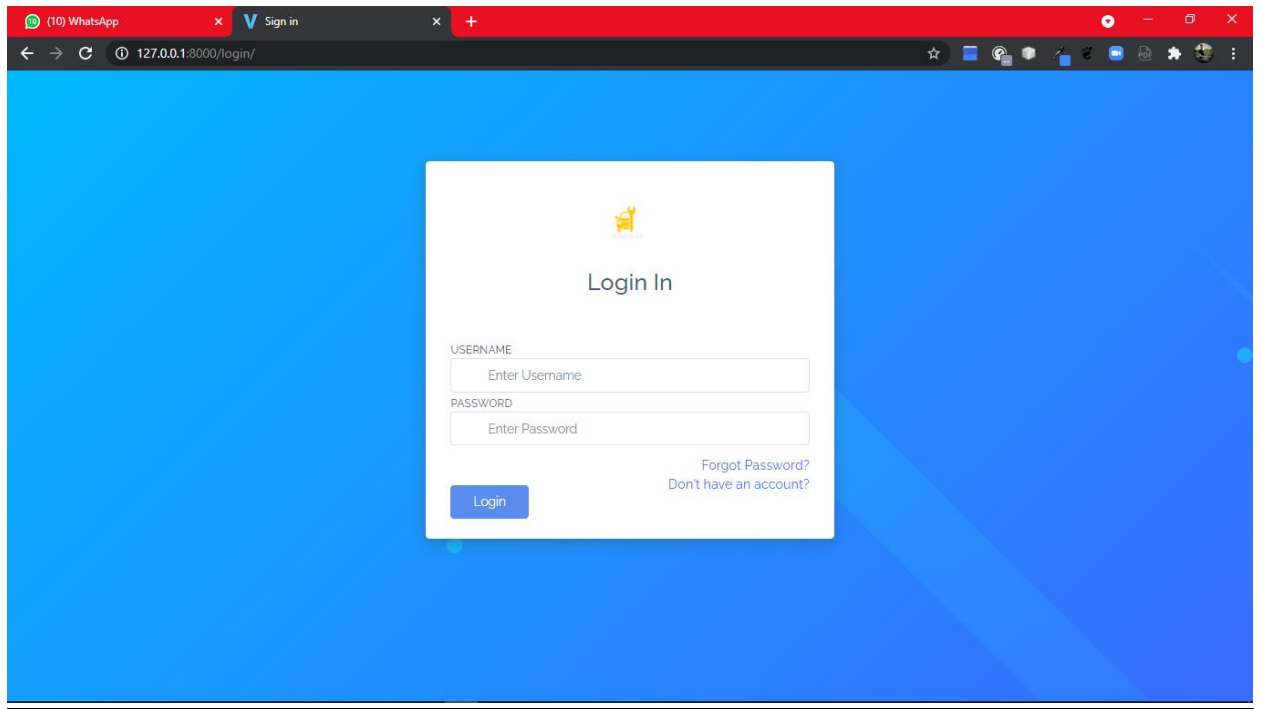
SQLite Commands:

The standard SQLite commands to interact with relational databases are similar to SQL. They are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into groups based on their operational nature.

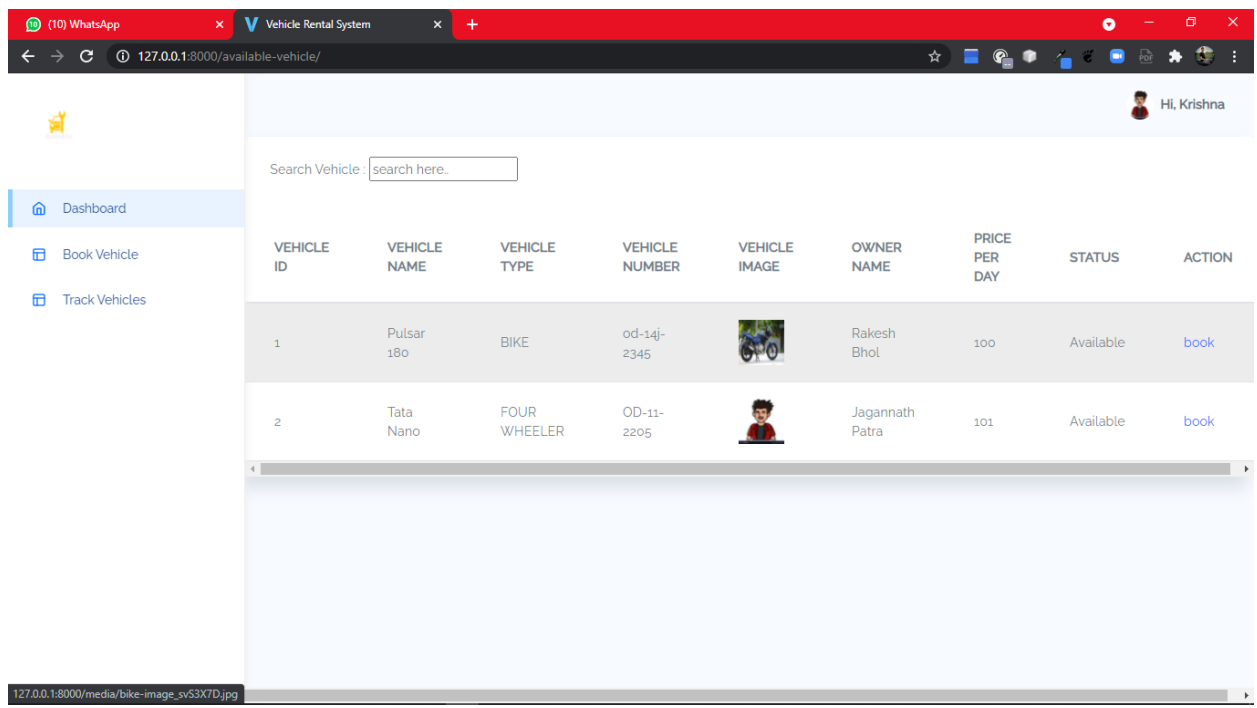
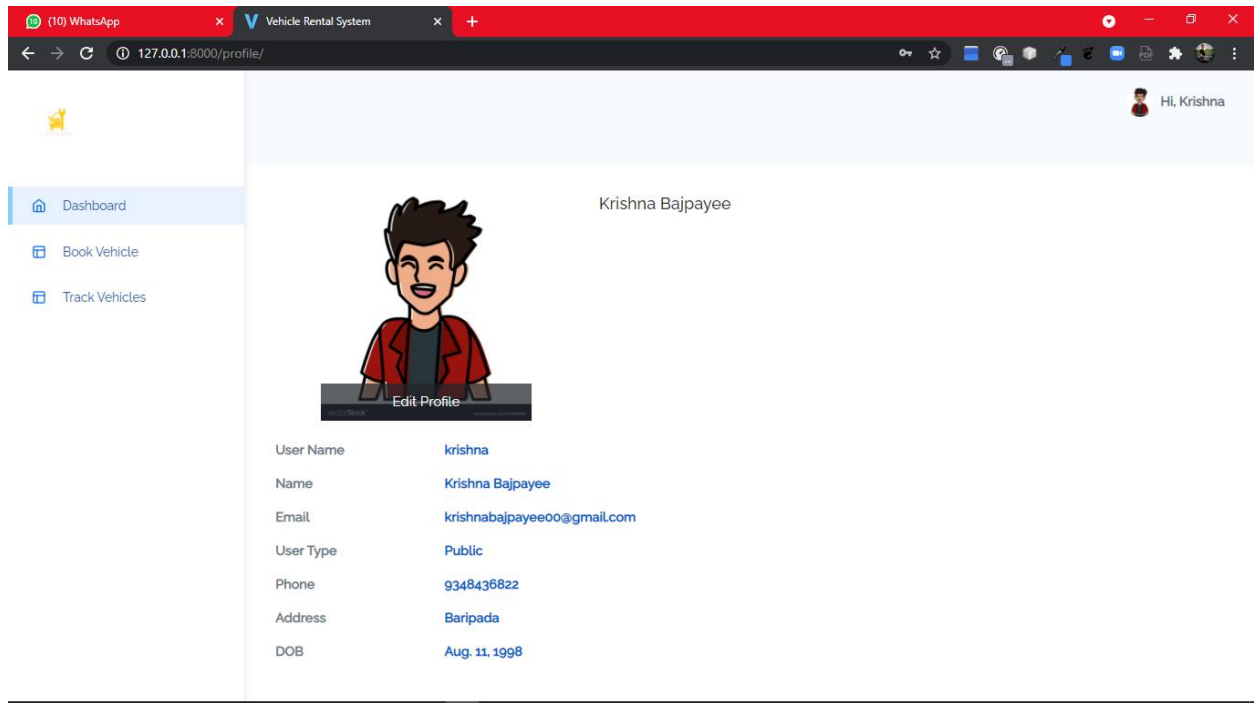
SNAPSHOTS OF SCREENS


- For all User

A screenshot of the 'Sign Up' page on the 'Ready & Go' website. The browser's address bar shows '127.0.0.1:8000/register/'. The page has a blue background. In the center, there is a white box containing the 'Sign Up' form. The form includes the following fields: 'FIRST NAME' (with placeholder 'Enter First Name *'), 'LAST NAME' (with placeholder 'Enter Last Name *'), 'USER NAME' (with placeholder 'Enter Username *'), 'CONTACT NO' (with placeholder 'Enter Contact Number *'), 'EMAIL-ID' (with placeholder 'Enter Email *'), 'DOB' (with placeholder 'dd-mm-yyyy' and a calendar icon), 'PROFILE PICTURE' (with a 'Choose file' button and 'No file chosen' text), 'PASSWORD' (with placeholder 'Enter Password *'), 'CONFIRM PASSWORD' (with placeholder 'Confirm Password *'), and 'ADDRESS' (with placeholder 'Enter Address *'). At the bottom left of the form is a blue 'Signup' button. At the bottom right, there are three links: 'Have an account? Login', 'Want to signup as Owner click here', and 'Want to signup as Admin click here'.



- **Public User**






[Dashboard](#)

[Book Vehicle](#)

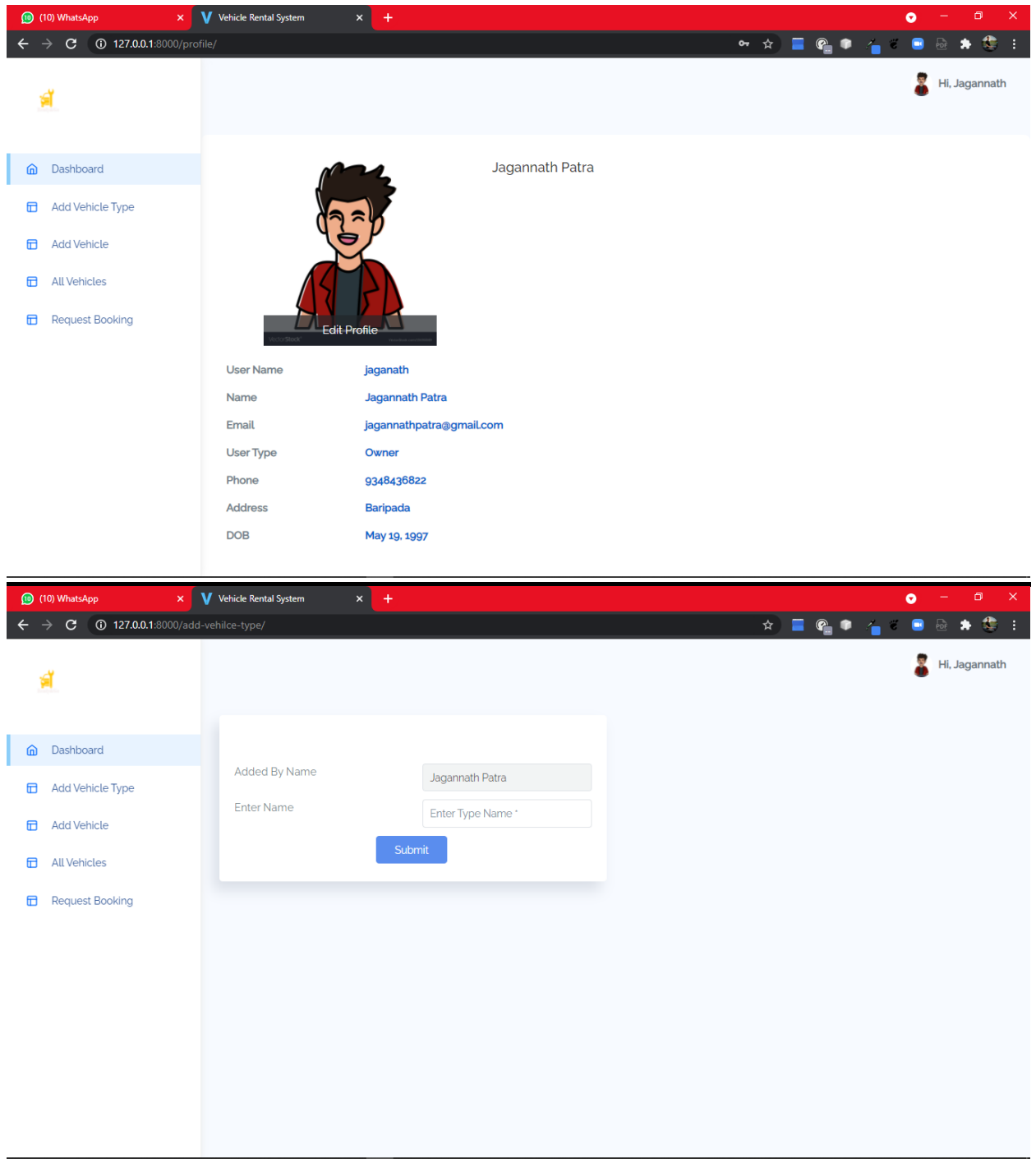
[Track Vehicles](#)

Hi, Krishna

Search Booking ID :

BOOKING ID	VEHICLE NAME	VEHICLE NUMBER	VEHICLE IMAGE	OWNER NAME	PRICE	PAID/UNPAID	STATUS	ACTION
12929581	Tata Nano	OD-11-2205		Jagannath Patra	404	Paid	Approved	track

- **Owner User**



• Admin

Hi, Admin

Admin Admin

[Edit Profile](#)

User Name: [admin](#)

Name: [Admin Admin](#)

Email: [admin@gmail.com](#)

User Type: [Admin](#)

Phone: [9878987675](#)

Address: [BBSR](#)

DOB: [March 7, 2021](#)

Hi, Admin

USER ID	USER NAME	DOB	Phone No	User Type	Action
2	Public User	March 3, 2021	9876564765	Public	view edit delete
3	Rakesh Bhol	March 7, 2021	9876456789	Owner	view edit delete
4	Chintu Chintu	March 8, 2021	9876456789	Owner	view edit delete
5	Krishna Bajpayee	Aug. 11, 1998	9348436822	Public	view edit delete
6	Jagannath Patra	May 19, 1997	9348436822	Owner	view edit delete

Vehicle Rental System

Hi, Admin


User Dashboard

TOTAL BOOKING	1	COMPLETED BOOKING	0	PENDING BOOKING	0
ON RENT BOOKING	0	APPROVED BOOKING	1	REJECTED BOOKING	0
UNPAID BOOKING	0	PAID BOOKING	0	TOTAL CUSTOMER	2

Vehicle Rental System

Hi, Admin

Search Booking ID:

BOOKING ID	VEHICLE NAME	VEHICLE NUMBER	VEHICLE IMAGE	OWNER NAME	PRICE	PAID/UNPAID	STATUS	ACTION
12929581	Tata Nano	OD-11-2205		Jagannath Patra	404	Paid	Approved	track edit delete

CODING

Sighn.html Register HTML code

```
<!DOCTYPE html>
<html lang="en">
{% load static %}

<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Sign up</title>
<link rel="stylesheet" href="{% static 'AdminStatic/assets/css/bootstrap.css' %}">

<link rel="shortcut icon" href="{% static 'AdminStatic/assets/images/favicon.svg' %}" type="image/x-icon">
<link rel="stylesheet" href="{% static 'AdminStatic/assets/css/app.css' %}">
</head>

<body>
<div id="auth">

<div class="container">
<div class="row">
<div class="col-md-7 col-sm-12 mx-auto">
<div class="card pt-4">
<div class="card-body">
<div class="text-center mb-5">

<h3>Sign Up</h3>
<p>Please fill the form to register.</p>
</div>
<form action="/register/" method="POST" enctype="multipart/form-data">
{% csrf_token %}
{% if msg %}

<center style="color: red;">{{msg}}</center>

{% endif %}
```

```

<div class="row">
<div class="col-md-6 col-12">
<div class="form-group">
<label for="first-name-column">First Name</label>
<input type="text" id="first-name-column" class="form-control"
placeholder="Enter First Name *" name="first_name" required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="last-name-column">Last Name</label>
<input type="text" id="last-name-column" class="form-
control" placeholder="Enter Last Name *"
name="last_name" required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="username-column">User Name</label>
<input type="text" id="username-column" class="form-
control" name="username" placeholder="Enter Username *"
required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="country-floating">Contact No</label>
<input type="number" id="country-floating" class="form-
control" placeholder="Enter Contact Number *"
name="contact" required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="company-column">Email-Id</label>
<input type="email" id="company-column" class="form-
control" name="email" placeholder="Enter Email *"
required>

```

```

</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="email-id-column">DOB</label>
<input type="date" id="email-id-column" class="form-control" name="dob"
required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="country-floating">Profile Picture</label>
<input type="file" id="country-floating" class="form-control" name="pic">
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="country-floating">Password</label>
<input type="password" id="country-floating" class="form-
control" placeholder="Enter Password *"
name="password" required>
</div>
</div>

```

```

<div class="col-md-6 col-12">
<div class="form-group">
<label for="company-column">Confirm Password</label>
<input type="password" id="company-column" class="form-
control" placeholder="Confirm Password *"
name="confirm_password" required>
</div>
</div>
<div class="col-md-6 col-12">
<div class="form-group">
<label for="company-column">Address</label>
<textarea class="form-
control" id="exampleFormControlTextarea1" placeholder="Enter Address *"

```

```
name="address" rows="1" required></textarea>
</div>
</div>
</div>
<div class="float-right">
Have an account? <a href="/login/">Login</a><br>
</div>
<div class="clearfix">
<button class="btn btn-primary float-center">Signup</button>
</div>
</form>

</div>
</div>
</div>
</div>
</div>

</div>
<script src="{% static 'AdminStatic/assets/js/feather-icons/feather.min.js' %}"></script>
<script src="{% static 'AdminStatic/assets/js/app.js' %}"></script>

<script src="{% static 'AdminStatic/assets/js/main.js' %}"></script>
</body>

</html>
```

Add-vehicle.html

```
{% extends 'adminbase.html' %}
{% block content %}
{% load static %}
<div class="main-content container-fluid">
  <div class="page-title">
    <div class="row">
      <div class="col-12 col-md-6 order-md-1 order-last">

        </div>
      </div>

    </div>

    <!-- Borderless table end -->

    <!-- Hoverable rows start -->
    <div class="row" id="table-hover-row">
      <div class="col-md-6">
        <div class="card">

          <div class="card-content">

            <div class="card-body">
              <h5>Add Vehicle</h5>
            </div>

            <form action="/add-vehicle/" method="POST" class="form-
body" style="padding: 20px;" enctype="multipart/form-data">
              {% csrf_token %}
              {% if msg %}
              <p style="color: red;">{{msg}}</p>
              {% endif %}
              <div>

                <div class="col-md-6">
                  <label>Vehicle Added By</label>
                </div>
                <div class="col-md-6 form-group">
                  <input class="form-control"
                    value="{{request.user.first_name}} {{request.user.last_name}}" disabled>
                </div>
                <div class="col-md-6">
                  <label>Select Vehicle Type</label>
                </div>
                <select class="col-md-4 form-
group" name='vehicle_type' required style="height: 30px; overflow-y: scroll;">
```

```

        {% for t in types %}
        <option class="form-control" value="{{t.id}}">{{t}}</option>
        {% endfor %}
    </select>
    <br><br>
    <div class="col-md-6">
        <label>Enter Vehicle Name</label>
    </div>
    <div class="col-md-6 form-group">
        <input type="text" id="email-id" class="form-control" name="name"
            placeholder="Enter Vehicle Name *" required>
    </div>
    <br>
    <div class="col-md-6">
        <label>Enter Vehicle Number</label>
    </div>
    <div class="col-md-6 form-group">
        <input type="text" id="email-id" class="form-control" name="vehicle_number"
            placeholder="Enter Vehicle Number *" required>
    </div>
    <br>
    <div class="col-md-6">
        <label>Enter Vehicle Image</label>
    </div>
    <div class="col-md-6 form-group">
        <input type="file" id="email-id" class="form-control" name="image" required>
    </div>
    <br>
    <div class="col-md-6">
        <label>Price Per Day</label>
    </div>
    <div class="col-md-6 form-group">
        <input type="text" id="email-id" class="form-control" name="price"
            placeholder="Enter Price Per Day *" required>
    </div>
    <br>
    <div class="col-sm-12 d-flex justify-content-center">
        <button type="submit" class="btn btn-primary mr-1 mb-1">Submit</button>

    </div>
</div>
</div>
</form>
</div>
</div>
</div>
</div>
</div>

```



```
</div>
{% endblock %}
```

Dashboard.html

```
{% extends 'adminbase.html' %}
{% block content %}
{% load static %}
<div class="main-content container-fluid">
  <div class="page-title">
    <h3> User Dashboard</h3>
    <p class="text-subtitle text-muted"></p>
  </div>
  <section class="section">
    {% if request.user.userprofile_set.first.userType == "Public" %}
    <div class="row mb-2">
      <div class="col-12 col-md-4">
        <div class="card card-statistic">
          <div class="card-body p-0">
            <div class="d-flex flex-column">
              <div class='px-3 py-3 d-flex justify-content-between'>
                <h3 class='card-title'>Total Booked Vehicle</h3>
                <div class="card-right d-flex align-items-center">
                  <p>{{booked_vehicle}}</p>
                </div>
              </div>
              <div class="chart-wrapper">
                <canvas id="canvas1" style="height:100px !important"></canvas>
              </div>
            </div>
          </div>
        </div>
      </div>
      <div class="col-12 col-md-4">
        <div class="card card-statistic">
          <div class="card-body p-0">
            <div class="d-flex flex-column">
              <div class='px-3 py-3 d-flex justify-content-between'>
                <h3 class='card-title'>total Paid Booking</h3>
                <div class="card-right d-flex align-items-center">
                  <p>{{user_paid_vehicle}}</p>
                </div>
              </div>
              <div class="chart-wrapper">
                <canvas id="canvas2" style="height:100px !important"></canvas>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
    </div>
  </div>
  <div class="col-12 col-md-4">
    <div class="card card-statistic">
      <div class="card-body p-0">
        <div class="d-flex flex-column">
          <div class='px-3 py-3 d-flex justify-content-between'>
            <h3 class='card-title'>total Paid Booking</h3>
            <div class="card-right d-flex align-items-center">
              <p>{{user_paid_vehicle}}</p>
            </div>
          </div>
          <div class="chart-wrapper">
            <canvas id="canvas2" style="height:100px !important"></canvas>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

```

        </div>
    </div>
    <div class="col-12 col-md-4">
        <div class="card card-statistic">
            <div class="card-body p-0">
                <div class="d-flex flex-column">
                    <div class='px-3 py-3 d-flex justify-content-between'>
                        <h3 class='card-title'>Total Unpaid Booking</h3>
                        <div class="card-right d-flex align-items-center">
                            <p>{{user_unpaid_vehicle}}</p>
                        </div>
                    </div>
                    <div class="chart-wrapper">
                        <canvas id="canvas3" style="height:100px !important"></canvas>
                    </div>
                </div>
            </div>
        </div>
    </div>
    </div>
    {% endif %}
    {% if request.user.userprofile_set.first.userType == "Owner" %}
    <div class="row mb-2">
        <div class="col-12 col-md-4">
            <div class="card card-statistic">
                <div class="card-body p-0">
                    <div class="d-flex flex-column">
                        <div class='px-3 py-3 d-flex justify-content-between'>
                            <h3 class='card-title'>My Total Vehicle</h3>
                            <div class="card-right d-flex align-items-center">
                                <p>{{owner_total_vehicle}} </p>
                            </div>
                        </div>
                        <div class="chart-wrapper">
                            <canvas id="canvas1" style="height:100px !important"></canvas>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    </div>
    <div class="col-12 col-md-4">
        <div class="card card-statistic">
            <div class="card-body p-0">
                <div class="d-flex flex-column">
                    <div class='px-3 py-3 d-flex justify-content-between'>
                        <h3 class='card-title'>Completed Booking</h3>
                        <div class="card-right d-flex align-items-center">

```

```

        <p>{{owners_total_completed_booking}}</p>
    </div>
</div>
<div class="chart-wrapper">
    <canvas id="canvas2" style="height:100px !important"></canvas>
</div>
</div>
</div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>My On Rent Vehicles</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{owners_total_onrent_booking}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas2" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
</div>
</div>
{% endif %}
{% if request.user.userprofile_set.first.userType == "Admin" %}
<div class="row mb-2">
    <div class="col-12 col-md-4">
        <div class="card card-statistic">
            <div class="card-body p-0">
                <div class="d-flex flex-column">
                    <div class='px-3 py-3 d-flex justify-content-between'>
                        <h3 class='card-title'>Total Booking</h3>
                        <div class="card-right d-flex align-items-center">
                            <p>{{total_booking}} </p>
                        </div>
                    </div>
                    <div class="chart-wrapper">
                        <canvas id="canvas1" style="height:100px !important"></canvas>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

```

```

</div>
<div class="col-12 col-md-4">
  <div class="card card-statistic">
    <div class="card-body p-0">
      <div class="d-flex flex-column">
        <div class='px-3 py-3 d-flex justify-content-between'>
          <h3 class='card-title'>Completed Booking</h3>
          <div class="card-right d-flex align-items-center">
            <p>{{completed_booking}}</p>
          </div>
        </div>
      </div>
      <div class="chart-wrapper">
        <canvas id="canvas2" style="height:100px !important"></canvas>
      </div>
    </div>
  </div>
</div>
<div class="col-12 col-md-4">
  <div class="card card-statistic">
    <div class="card-body p-0">
      <div class="d-flex flex-column">
        <div class='px-3 py-3 d-flex justify-content-between'>
          <h3 class='card-title'>Pending Booking</h3>
          <div class="card-right d-flex align-items-center">
            <p>{{total_pending_booking}}</p>
          </div>
        </div>
      </div>
      <div class="chart-wrapper">
        <canvas id="canvas3" style="height:100px !important"></canvas>
      </div>
    </div>
  </div>
</div>
<div class="col-12 col-md-4">
  <div class="card card-statistic">
    <div class="card-body p-0">
      <div class="d-flex flex-column">
        <div class='px-3 py-3 d-flex justify-content-between'>
          <h3 class='card-title'>On Rent Booking</h3>
          <div class="card-right d-flex align-items-center">
            <p>{{onrent_booking}}</p>
          </div>
        </div>
      </div>
      <div class="chart-wrapper">
        <canvas id="canvas3" style="height:100px !important"></canvas>
      </div>
    </div>
  </div>
</div>

```

```

        </div>
    </div>
</div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Approved Booking</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{approved_booking}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Rejected Booking</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{rejected_booking}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>UnPaid Booking</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{unpaid_booking}}</p>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

```

```

        <div class="chart-wrapper">
            <canvas id="canvas3" style="height:100px !important"></canvas>
        </div>
    </div>
</div>
</div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Paid Booking</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{paid_booking}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Total Customer</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{total_public_user}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Total Vehicle Owner</h3>
                    <div class="card-right d-flex align-items-center">

```

```

        <p>{{total_owner_user}}</p>
    </div>
</div>
<div class="chart-wrapper">
    <canvas id="canvas3" style="height:100px !important"></canvas>
</div>
</div>
</div>
</div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Total Admin</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{total_admin_user}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-12 col-md-4">
    <div class="card card-statistic">
        <div class="card-body p-0">
            <div class="d-flex flex-column">
                <div class='px-3 py-3 d-flex justify-content-between'>
                    <h3 class='card-title'>Total Earned Till now</h3>
                    <div class="card-right d-flex align-items-center">
                        <p>{{total_earned}}</p>
                    </div>
                </div>
                <div class="chart-wrapper">
                    <canvas id="canvas3" style="height:100px !important"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>
</div>
</div>
    {% endif %}
{% endblock %}

```

Edit-profile.html

```
{% extends 'adminbase.html' %}

{% block content %}
{% load static %}
<div class="container emp-profile">
  <div class="row">
    <div class="col-md-4">
      <div class="profile-img">
        {% if request.user.userprofile_set.first.profilePicture %}
        
        {% else %}
        
        {% endif %}
      </div>
    </div>
    <div class="col-md-6">
      <div class="profile-head">
        <h5>
          {{request.user.first_name}} {{request.user.last_name}}
        </h5>
      </div>
    </div>
  </div>

  <div class="col-md-4">
    <div class="tab-content profile-tab" id="myTabContent">
      <form action="/edit-profile/" method="POST" enctype="multipart/form-data">
        {% csrf_token %}
        {% if msg %}
        <p style="color: red;">{{msg}}</p>
        {% endif %}
        <div class="tab-pane fade show active" id="home" role="tabpanel" aria-labelledby="home-
tab">
          <br>
          <div class="row">
            <div class="col-md-6">
              <label>First Name</label>
```



```

</div>
<div class="col-md-6">
  <input type="text" name="first_name" value='{{request.user.first_name}}' required />
</div>
</div><br>
<div class="row">
  <div class="col-md-6">
    <label>Last Name</label>
  </div>
  <div class="col-md-6">
    <input type="text" name="last_name" value='{{request.user.last_name}}' required />
  </div>
</div><br>
<div class="row">
  <div class="col-md-6">
    <label>Email</label>
  </div>
  <div class="col-md-6">
    <input type="email" name="email" value='{{request.user.email}}' required />
  </div>
</div><br>
<div class="row">
  <div class="col-md-6">
    <label>Phone</label>
  </div>
  <div class="col-md-6">
    <input type="number" name='contact'
      value='{{request.user.userprofile_set.first.contact_No}}' required />
  </div>
</div><br>
<div class="row">
  <div class="col-md-6">
    <label>Address</label>
  </div>
  <div class="col-md-6">
    <input type="text" name="address" value='{{request.user.userprofile_set.first.address}}'
      required />
  </div>
</div><br>
<div class="row">
  <div class="col-md-6">
    <label>DOB</label>
  </div>
  <div class="col-md-6">
    <input type="text" name="dob" value='{{request.user.userprofile_set.first.dob}}' required />
  </div>
</div>

```

```

        </div>
        <div class="col-md-6">
            <input type="date" name="dob" value="{{request.user.u
serprofile_set.first.dob |date:'Y-m-d'}}"
                required />
        </div>
    </div><br>
    <div class="row">
        <div class="col-md-6">
            <label>Profile Picture</label>
        </div>
        <div class="col-md-6">
            <input type="file" name="pic" />
        </div>
    </div><br>
    </div>
    <input class="btn btn-success" type="submit" value="Update" />
</form>
</div>
</div>

{% endblock %}

```

book-vehicle.html

```

{% extends 'adminbase.html' %}
{% block content %}
{% load static %}

<section id="basic-vertical-layouts">
    <div class="row match-height">
        <div class="col-md-8 col-12">
            <div class="card">

                <div class="card-content">
                    <div class="card-body">
                        <form action="/book-vehicle/{{details.id}}/" method="POST" class="form form-vertical"
                            enctype="multipart/form-data">
                            {% csrf_token %}
                            <div class="form-body">
                                <div class="row">
                                    <div class="col-md-12">
                                        <div class="form-group">
                                            <label for="first-name-vertical">Booking by User </label>
                                            <input type="text" id="first-name-vertical" class="form-control"
                                                name="fname"

```

```

        value="{{request.user.first_name}} {{request.user.last_name}}" disabled>

</div>
<div class="form-group">
    <label for="first-name-vertical">Book Vehicle Type</label>
    <input type="text" id="first-name-vertical" class="form-control"
        name="fname" value="{{details.vehicle_type.name}}" disabled>

</div>
<div class="form-group">
    <label for="first-name-vertical">Book Vehicle Name</label>
    <input type="text" id="first-name-vertical" class="form-control"
        name="fname" value="{{details.name}}" disabled>

</div>
<div class="form-group">
    <label for="first-name-vertical">Book Vehicle Number</label>
    <input type="text" id="first-name-vertical" class="form-control"
        name="fname" value="{{details.vehiclenumber}}" disabled>

</div>
<div class="form-group">
    <label for="first-name-vertical">Price Per Day</label>
    <input type="text" id="first-name-vertical" class="form-control"
        name="fname" value="{{details.price_per_day}}" disabled>

</div>
</div>
<div class="form-group">
    <label for="first-name-vertical">Book For Number of Day's *</label>
    <input type="number" id="first-name-vertical" class="form-control" name="days"
        required>
</div>

<div class="form-group">
    <label for="first-name-vertical">Book From Date *</label>
    <input type="date" id="first-name-vertical" class="form-
control" name="book_from"
        required>
</div>

<div class="form-group">
    <label for="first-name-vertical">Book upto Date *</label>
    <input type="date" id="first-name-vertical" class="form-control" name="book_to"
        required>
</div>
<div class="form-group">
    <label for="first-name-vertical">Enter ID Card Number *</label>

```

```
        <input type="number" maxlength="12" minlength="12" id="first-name-vertical"
            class="form-control" name="aadhar" required>
    </div>

    <div class="form-group">
        <label for="first-name-vertical">Enter ID Card Proof *</label>
        <input type="file" class="form-control" name="id_proof" required>
    </div>
</div>
</div>
<button type="submit" class="btn btn-primary">Book</button>
</form>
</div>
</div>

</div>
</div>

</div>
</section>

{% endblock %}
```

View Functions

```
def Signup(request):
    if request.method == 'POST':
        username = request.POST['username']
        pass1 = request.POST.get('password')
        pass2 = request.POST.get('confirm_password')
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        email = request.POST.get('email')
        contact = request.POST.get('contact')
        address = request.POST.get('address')
        dob = request.POST.get('dob')
        profile_pic = request.FILES.get('pic')
        if pass1!=pass2:
            msg='Password should be same.'
            return render(request,'auth-register.html',{'msg':msg})

        if len(contact)!=10:
            msg = 'Phone Number Should be equal to 10 digit'
            return render(request,'auth-register.html',{'msg':msg})

        try:
            user=User.objects.create_user(username=username,email=email,password=pass1,first_
            name=first_name,last_name=last_name)
        except:
            msg='Username already exist.'
            return render(request,'auth-register.html',{'msg':msg})

        UserProfile.objects.create(
            user=user,profilePicture=profile_pic,
            contact_No=contact,
            address=address,
            dob=dob)
        return redirect('login')
    return render(request, 'auth-register.html')
```

User-register.py view function

```
def OwnerSignup(request):
    if request.method == 'POST':
        username = request.POST['username']
        pass1 = request.POST.get('password')
        pass2 = request.POST.get('confirm_password')
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        email = request.POST.get('email')
        contact = request.POST.get('contact')
        address = request.POST.get('address')
        dob = request.POST.get('dob')
```

```

        profile_pic = request.FILES.get('pic')

    if pass1!=pass2:
        msg='Password should be same.'
        return render(request,'owner-signup.html',{'msg':msg})

    if len(contact)!=10:
        msg = 'Phone Number Should be equal to 10 digit'
        return render(request,'owner-signup.html',{'msg':msg})

    try:
        user=User.objects.create_user(username=username,email=email,password=pass1,first_n
ame=first_name,last_name=last_name)
    except:
        msg='Username already exist.'
        return render(request,'owner-signup.html',{'msg':msg})
    UserProfile.objects.create(user=user,profilePicture=profile_pic,contact_No=contact, address=ad
dress, dob=dob, userType="Owner")
    return redirect('login')
    return render(request, 'owner-signup.html')

def AdminSignup(request):
    if request.method == 'POST':
        username = request.POST['username']
        pass1 = request.POST.get('password')
        pass2 = request.POST.get('confirm_password')
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        email = request.POST.get('email')
        contact = request.POST.get('contact')
        address = request.POST.get('address')
        dob = request.POST.get('dob')
        profile_pic = request.FILES.get('pic')

        if pass1!=pass2:
            msg='Password should be same.'
            return render(request,'admin-signup.html',{'msg':msg})

        if len(contact)!=10:
            msg = 'Phone Number Should be equal to 10 digit'
            return render(request,'admin-signup.html',{'msg':msg})

        try:
            user=User.objects.create_user(username=username,email=email,password=pass1,first_
name=first_name,last_name=last_name)
        except:
            msg='Username already exist.'
```

```

        return render(request, 'admin-signup.html', {'msg': msg})

    UserProfile.objects.create(user=user, profilePicture=profile_pic, contact_No=contact, address=address, dob=dob, userType="Admin")
    return redirect('login')
    return render(request, 'admin-signup.html')

def Login(request):

    if request.user.is_authenticated:
        logout(request)
        return redirect('login')

    msg=""
    if request.method=='POST':
        uname=request.POST['username']
        pwd=request.POST['password']
        data=authenticate(username=uname,password=pwd)
        if data !=None:
            login(request,data)
            return redirect('profile')
            msg='Incorrect Username or Password'
    return render(request, 'auth-login.html', {'msg':msg})

def Logout(request):
    logout(request)
    return render(request, 'index.html')

def SendEmailForForgotPassword(request):

    if request.method == "POST":
        username = request.POST.get('username')

    try:
        user = User.objects.get(username=username)
    except:
        msg='Invalid Username.'
        return render(request, 'ForgotPassEmail.html', {'msg':msg})

    otp = OTP.objects.create(user=user)

    body = f'Forgot Password?? This is your OTP to get your password reset {otp.otp}'
    subject = 'Forgot Password for Vehicle Project Services Account'
    from_email = settings.EMAIL_HOST_USER

```

```

to_email = [user.email]

send_mail(subject, body, from_email, to_email, fail_silently=True)
return redirect('forgot-password')
return render(request, 'ForgotPassEmail.html')

def ForgotPassword(request):
    msg = ""
    if request.method == "POST":
        username = request.POST.get('username')
        user_otp = request.POST.get('otp')
        password1 = request.POST.get('password1')
        password2 = request.POST.get('password2')

        if password1!=password2:
            msg='Password should be same.'
            return render(request,'auth-forgot-password.html', {'msg':msg})
        try:
            user = User.objects.get(username=username)
        except:
            msg='Invalid Username.'
            return render(request, 'auth-forgot-password.html', {'msg':msg})

        otp = OTP.objects.filter(user=user).order_by('-created_at').first()
        if str(otp.otp) == user_otp:
            user.set_password(password1)
            user.save()
            return redirect('login')
        msg = 'Please Enter Correct OTP'
    return render(request, 'auth-forgot-password.html', {'msg':msg})

```


URLs.py code

```
from django.contrib import admin
from django.urls import path
from Admin.views import (
    Search,
    Profile,
    ChangePassword,
    AllUsers,
    AllOwner,
    AllPublic,
    EditProfile,
    UserDetails,
    DeleteUser,
    EditSingleUser,
    Dashboard,
    AddVehicleType,
    AddVehicle,
    EditVehicle,
    DeleteVehicle,
    TotalVehicle,
    TrackVehicle,
    Track,
    AvailableVehicle,
    BookVehicle,
    EditBooking,
    DeleteBooking,
    TrackCompletedVehicle,
    UnpaidCompletedRent
)
from Public.views import Login, Logout, Signup, Home, Contact ,About, OwnerSignup
, AdminSignup, SendEmailForForgotPassword, ForgotPassword
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', Home, name=''),
    path('about/', About, name='about'),
    path('contact/', Contact, name='contact'),
    path('login/', Login, name='login'),
    path('logout/', Logout, name='logout'),
    path('register/', Signup, name='register'),
    path('ownersignup/', OwnerSignup, name='ownersignup'),
    path('adminsSignup/', AdminSignup, name='adminsSignup'),

    path('users/', AllUsers, name='users'),
    path('all-owner/', AllOwner, name='all-owner'),
    path('all-public/', AllPublic, name='all-public'),

    path('profile/', Profile, name='profile'),
```

```

    path('dashboard/', Dashboard, name='dashboard'),
    path('user-details/<int:id>/', UserDetails, name='user-details'),
    path('edit-single-user/<int:id>/', EditSingleUser, name='edit-single-user'),
    path('delete-user/<int:id>/', DeleteUser, name='delete-user'),

    path('edit-profile/', EditProfile, name='edit-profile'),
    path('email-forgotpassword/', SendEmailForForgotPassword, name='email-
forgotpassword'),
    path('forgot-password/', ForgotPassword, name='forgot-password'),
    path('change-password/', ChangePassword, name='change-password'),

    path('add-vehilce-type/', AddVehicleType, name='add-vehilce-type'),
    path('add-vehicle/', AddVehicle ,name="add-vehicle"),
    path('delete-vehicle/<int:id>/', DeleteVehicle, name='delete-vehicle'),
    path('total-vehicle/', TotalVehicle, name='total-vehicle'),

    path('edit-vehicle/<int:id>/', EditVehicle, name='edit-vehicle'),

    path('available-vehicle/', AvailableVehicle, name='available-vehicle'),

    path('book-vehicle/<int:id>/', BookVehicle, name='book-vehicle'),
    path('edit-booking/<int:id>/', EditBooking, name="edit-booking"),
    path('delete-booking/<int:id>/', DeleteBooking, name='delete-booking'),
    path('completed-rent/', TrackCompletedVehicle, name='completed-rent'),
    path('unpaid-completed-rent/', UnpaidCompletedRent, name='unpaid-completed-
rent'),
    path('track-vehicle/', TrackVehicle, name='track-vehicle'),
    path('track/<int:id>/', Track, name='track'),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

Models.py (Object Relational Modeling)

```
import random

from django.db import models
from django.contrib.auth.models import User
# Create your models here.
class UserProfile(models.Model):
    USER_TYPES = [
        ('Owner', 'Owner'),
        ('Admin', 'Admin'),
        ('Public', 'Public'),
    ]
    user=models.ForeignKey(User,on_delete=models.CASCADE)
    profilePicture = models.ImageField(blank=True, null=True)
    contact_No = models.IntegerField()
    dob = models.DateField(null=True, blank=True)
    address = models.TextField(null=True, blank=True)
    userType = models.CharField(max_length=15, choices=USER_TYPES, default='Public')
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.user.username

class OTP(models.Model):
    def Get_OTP():
        return random.randint(100000, 999999)

    user = models.ForeignKey(User, on_delete=models.CASCADE)
    otp = models.IntegerField(default=Get_OTP)
    created_at = models.DateTimeField(auto_now_add=True)

class VehicleType(models.Model):
    user = models.ForeignKey(UserProfile, on_delete=models.CASCADE)
    name = models.CharField(max_length=100)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.name
```

```

class Vehicle(models.Model):
    STATUS = (
        ('Not Available', 'Not Available'),
        ('Available', 'Available')
    )
    user = models.ForeignKey(UserProfile, on_delete=models.CASCADE)
    vehicle_type = models.ForeignKey(VehicleType, on_delete=models.CASCADE)
    vehiclenumber = models.CharField(max_length=255)
    name = models.CharField(max_length=255)
    image = models.ImageField()
    price_per_day = models.IntegerField()
    status = models.CharField(max_length=15, choices=STATUS, default='Available')
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.name

class RentVehicle(models.Model):
    STATUS = (
        ('Pending', 'Pending'),
        ('Approved', 'Approved'),
        ('Rejected', 'Rejected'),
        ('On Rent', 'On Rent'),
        ('Completed Rent', 'Completed Rent')
    )

    PAID_STATUS = (
        ('UnPaid', 'UnPaid'),
        ('Paid', 'Paid')
    )

    def BookingId():
        return int(random.randint(10000000,99999999))

    user = models.ForeignKey(UserProfile, on_delete=models.CASCADE)
    vehicle = models.ForeignKey(Vehicle, on_delete=models.CASCADE)
    booking_id = models.IntegerField(default=BookingId)
    aadhar_number = models.BigIntegerField()
    id_proof = models.ImageField()

```

```
book_day = models.SmallIntegerField()
book_from = models.DateField()
book_to = models.DateField()
total_price = models.IntegerField(null=True)
paid_status = models.CharField(max_length=15, choices=PAID_STATUS, default="UnPaid")
status = models.CharField(max_length=15, choices=STATUS, default='Pending')
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)

def __str__(self):
    return f'{self.booking_id} by {self.user.user.id}'
```

TESTING

7.1 Integration Testing:

Integration testing done before, during and after integration of a new module into the main software package. This involves testing of each individual code module. One piece of software can contain several modules which are often created by several different programmers. It is crucial to test each modules effect on the entire program model. After integration testing the project works successfully.

7.2 Unit Testing:

Unit testing performed on each module or block of code during development. Unit testing is normally done by the programmer who writes the code.

7.3 System Testing:

System testing done by a professional testing agent on the completed software product before it is introduced to the market.

7.4 Acceptance Testing:

Acceptance testing is a beta testing of the product done by the actual end user.

7.5 Recovery Testing:

Recovery testing is done to demonstrate a software salutation is reliable, trustworthy and can successfully recoup form possible crashes.

7.6 Functional Testing:

Functional Testing also known as functional completeness testing. Functional Testing involves trying to think of any possible missing functions. Testers might make a list of additional functionalities that a product could to improve it during functional testing.

7.7 Hardware/Software Testing:

IBM refers to Hardware/Software testing as “HW/SW Testing”. This is when the tester focuses his/her attention on the interactions between the hardware and software during system testing.

7.8 Security Testing:

Security Testing is a variant of Software Testing which ensures, that system and applications in an organization, are free from any loopholes that may cause a big loss. Security testing of any system is about finding all possible loopholes and weaknesses of the system which might result into a loss of information at the hands of the employees or outsiders of the Organization.

CONCLUSION

Our Aim is to design and create a data management System for a Vehicle Rental Company. This enables admin can rent a vehicle that can be used by a customer. By paying the money during a Specified Period of time. This system increases customer retention and simplify vehicle and staff Management in an efficient way.

This software Vehicle Rental System has a very user friendly interface. Thus the users will feel very easy to work on it. By using this system admin can manage their rental, payment, employment issues and vehicle issues such as and insurance. The vehicle information can be added to the system. Or existed vehicle information can be edited or deleted too by Administrator. The transaction reports of the vehicle rental system can be retrieved by the admin, when its required. Thus, there is no delay in the availability of any vehicle information, whenever needed, vehicle information can be Captured very quickly and easily.

The customers can also use the system to get vehicle rent. The customer should create a new account(register) before logging in or he / she can log into the System with his/her created account. Then he/she can view the available vehicles in a branch and make a reservation for a Vehicle. This system will helpful to the admin as well as to the customer also.

REFERENCE

- **Django documentation and References**

<https://docs.djangoproject.com>

<https://www.tutorialspoint.com/django/index.htm>

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django>

- **Python Documentation and References**

<https://docs.python.org/3/tutorial/index.html>

<https://www.tutorialspoint.com/python3/index.htm>

<https://www.w3schools.com/python/>

- **HTML ,CSS, JavaScript**

<https://www.w3schools.com/html/default.asp>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/js/default.asp>