THE COMMERCIAL VEHICLE BOOKING

A Mini Project Report

submitted by

MUHSINA(MES24MCA-2035)

to the APJ Abdul Kalam Technological University
in partial fulfilment of the requirements for the award of the Degree

of

Master of Computer Applications

Department of Computer Applications

MES College of Engineering

Kuttippuram, Malappuram – 679582

October, 2025

Declaration

I undersigned hereby declare that the project report THE COMMERCIAL VEHICLE BOOKING submitted for partial fulfilment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done by me under supervision of Mrs. Febin Aziz, Assistant Professor, Department of Computer Applications. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

MUHSINA

MES24MCA-2035

Date:08-10-2025

DEPARTMENT OF COMPUTER APPLICATIONS

MES COLLEGE OF ENGINEERING, KUTTIPPURAM

CERTIFICATE

This is to certify that the report entitled **THE COMMERCIAL VEHICLE BOOKING** is a bonafide record of the Mini Project work during the year 2025-26 carried out by **MUHSINA (MES24MCA-2035)** submitted to the APJ Abdul Kalam Technological University, in partial fulfilment of the requirements for the award of the Master of Computer Applications, under my guidance and supervision. This report in any form has not been submitted to any other University or Institution for any purpose.

Internal Supervisor

Head of The Department

Acknowledgment

This endeavor stands incomplete without dedicating my deepest gratitude to the individuals who have contributed to the successful completion of this project.

I pay my gratitude to the **Almighty** for the invisible help and blessings that guided the fulfilment of this work.

At the outset, I express my heartfelt thanks to the **Head of the Department, Prof. Hyderali K**, for granting me the opportunity and necessary permission to undertake this research. I take this opportunity to express my profound gratitude to my project guide, **Ms. Febin Aziz**, Assistant Professor, Department of Computer Applications, MES College of Engineering, Kuttippuram, for her valuable support, continuous encouragement, and patient guidance throughout the course of this project.

My sincere thanks also go to all my teaching and non-teaching staff for their kind encouragement, guidance, and wholehearted support.

Last but not least, I am gratefully indebted to my family and friends for their constant emotional support and precious help in completing my project.

MUHSINA

MES24MCA-2035

Abstract

The Commercial Vehicle Booking platform is a website designed to streamline and centralize the process of booking commercial vehicles. This system aims to address the inefficiencies and lack of transparency often found in traditional booking methods providing a user-friendly interface for both businesses and individuals to efficiently find compare, and book suitable commercial vehicles. platform likely offers functionalities such as user registration and authentication, comprehensive listings of available commercial vehicles, transparent pricing, and real time booking capabilities. It is also expected to include administrative tools for managing vehicles, drivers, and booking requests, thereby enhancing operational efficiency and accountability.

the "Commercial Vehicle" mini-project aims to fix the problems with old-fashioned commercial vehicle booking. Right now, it's hard for people and businesses to find and book the right vehicles. They struggle to see what's available, compare prices easily, or get real-time schedules. This manual way of doing things wastes time and money, and not being able to track vehicles causes a lot of uncertainty. All these issues make it difficult to make smart decisions and keep things running smoothly. While specific technologies are not detailed in the repository name, such a system typically leverages modern web development frameworks (e.g., Python/Django) for the backend, a robust database for data management, and standard frontend technologies (HTML, CSS, JavaScript) for an intuitive user interface. The overall objective is to create an efficient, convenient, and reliable solution for commercial transportation need The Commercial Vehicle Booking website aims to enhance the user experience by simplifying the process of booking commercial vehicles. With its user-friendly interface, it allows for easy navigation and quick access to desired information. The platform offers a comprehensive database of commercial vehicles, enabling users to search and filter based on their specific requirements. The booking system is designed to be efficient and secure, allowing users to reserve their chosen vehicle with minimal effort. Ultimately, the website's goal is to streamline the user experience, making commercial vehicle booking convenient and hassle-free.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

In today's transportation and logistics landscape, the efficient movement of goods and materials is crucial for businesses across all sectors. This efficiency is heavily dependent on the ability to secure the right commercial vehicle at the right time. Traditionally, the process of booking commercial vehicles, such as trucks, vans, and specialized carriers, has been cumbersome, involving numerous phone calls, fragmented local listings, and a lack of real-time price transparency. This fragmented approach often leads to delays, mismatched vehicle-to-load requirements, and difficulty in comparing competitive pricing, thus impacting operational costs and delivery timelines.

Existing digital solutions, such as established fleet management systems and load boards, have attempted to streamline this process. One such system focuses on connecting transporters with businesses by aggregating load requirements. While it offers a large database of available loads, its limitations include a complex bidding process, an inconsistent quality of service among listed carriers, and a lack of dynamic route optimization features. Another platform provides real-time tracking and basic booking, but its high subscription cost and limited customization for small and medium-sized enterprises (SMEs) restrict wider adoption. Furthermore, the lack of a unified, user-friendly platform that integrates vehicle availability, instant rate quoting, and simplified booking remains a significant challenge.

To address these challenges, the Commercial Vehicle Booking System leverages modern web development tools like Django and JavaScript to deliver a user-centric, reliable, and scalable platform. The backend is built with Django to handle user authentication, vehicle data, and booking transactions securely and efficiently. Furthermore, the Django Admin interface will be enhanced using Django Jazzmin to provide a modern and customizable administrative

experience. By incorporating a robust database architecture and simple API integration, this system centralizes vehicle options and provides instant, transparent quotes.

The Commercial Vehicle Booking System falls under the domain of e-logistics and supply chain automation. The project aims to automate the entire process of vehicle discovery, instant quotation, and booking, enabling users to make well-informed logistical decisions. By integrating instant quotes and automated transaction handling, the system addresses the limitations of manual negotiation and fragmented listings. The user interface (UI) is designed using Bootstrap, providing an intuitive layout for easy vehicle filtering and selection, while the user experience (UX) is tailored to ensure users can seamlessly post requirements and manage their bookings with minimal effort.

Once registered, users can log in to the system and begin posting their requirements, including load type, weight, required vehicle type, pickup, and delivery locations. This streamlined approach minimizes the effort required to secure a vehicle, making the experience more efficient and user-friendly. For each request, the system instantly generates multiple competitive quotes from registered transport providers. This feature empowers users to maintain control over their budget, ensuring they select a vehicle that aligns with both their logistical needs and financial constraints. The flexibility to review and compare quotes allows users to adapt their sourcing strategy as market demands change, and the booking history provides valuable data for future logistical planning and analysis.

While digital booking platforms exist, many fall short in providing real-time transparency and dynamic pricing models tailored to specific routes and vehicle types. Businesses that rely on manual or fragmented booking processes often incur unnecessary costs and experience operational bottlenecks due to the impracticality of comparing all available options simultaneously. Moreover, existing systems may not support instant booking confirmation or integrated tracking, leaving users uncertain about their shipments. The Commercial Vehicle Booking System seeks to overcome these issues by automating vehicle sourcing and providing users with instant quotes and confirmation, thus empowering users with real-time logistical visibility and reducing the risk of costly delays and inefficiencies in their supply chain.

1.1 Motivation

- This project aims to create a centralized online platform to fix the inefficient and non-transparent traditional process of booking commercial vehicles.
- Enhance the user experience with a user-friendly interface for easy navigation and quick access to information.
- Provide transparent pricing, real-time availability and vehicle tracking to offer convenience, save time, and promote cost efficiency.
- A key motivation was to address difficulties in project planning and creating an attractive user interface.

1.2 Objectives

The primary objective of the Commercial Vehicle Booking System is to streamline and digitize the process of commercial vehicle sourcing and reservation, enhancing efficiency, transparency, and cost-effectiveness for both businesses and vehicle operators.

This system is specifically designed to achieve the following goals:

- Easy Booking: To develop a simple and intuitive online platform that allows users to quickly search, compare, and book commercial vehicles based on their specific logistics requirements, eliminating the need for manual inquiries.
- User-Centric Access & Information: To ensure a user-friendly experience with easy navigation and readily available vehicle information, schedules, and transparent rate details. This includes Self-Service capabilities for users and drivers to efficiently manage their profiles, active bookings, and availability.
- Centralized Wide Selection: To maintain and manage a large, accurate database of commercial vehicles (including different types, capacities, and availability) through a

dedicated Admin Tools dashboard for robust management of users, drivers, and the fleet inventory.

- Key Feature Integration: To provide real-time availability and scheduling checks with transparent pricing, significantly reducing decision latency and enabling immediate booking confirmation.
- Future Scalability: To develop a flexible and scalable architecture—utilizing Django and JavaScript—that can be extended to integrate additional features like payment gateways, dynamic route optimization, and potential expansion to other transport segments or logistics services.

1.3 Contributions

The Commercial Vehicle Booking System offers several key contributions that significantly enhance the efficiency and transparency of commercial logistics and transport management.

Firstly, it provides a crucial shift from fragmented, manual sourcing to a centralized, automated booking platform, saving businesses substantial time and reducing overhead by eliminating constant phone calls and rate negotiations. The system's architecture, built with Django and JavaScript, ensures reliable real-time availability and scheduling checks, guaranteeing that users receive immediate, actionable information without disrupting the core application performance.

Secondly, a major contribution is the introduction of transparent pricing. By instantly aggregating vehicle data and providing clear, competitive quotes, the system empowers users to make well-informed, cost-efficient logistical decisions.

Thirdly, the development of robust Admin Tools and Self-Service portals contributes to enhanced operational efficiency. This allows vehicle operators and drivers to manage their own profiles, availability, and bookings, streamlining the entire service ecosystem.

Finally, the scalable and secure nature of the system is designed for long-term usability and future enhancements. This includes the potential for integration with advanced features like dynamic route optimization and telematics data, positioning the platform to adapt to the evolving demands of Logistics 4.0.

1.4 Report Organization

The project report is divided into four chapters. Chapter 2 describes system study where it explains the existing system and its limitations. This chapter also gives an insight to the proposed system and it functionalities. Chapter 3 details the methodology used for implementing the project. In methodology, modules of the project, and sprint details are provided. Chapter 4 gives the results and discussions. Finally, Chapter 5 gives the conclusion.

Chapter 2. System Study

The Commercial Vehicle Booking System is proposed to resolve the prevalent inefficiencies within commercial logistics, where vehicle sourcing is typically fragmented and lacks real-time visibility. This system establishes a centralized digital platform that connects users with commercial vehicle operators to facilitate quick, transparent, and confirmed reservations. By automating availability checks and quote generation, the solution aims to significantly reduce procurement lead times and enable highly efficient, cost-effective logistical planning for businesses.

2.1 Existing System

The main problem is that booking a commercial vehicle is too complicated and inefficient for both businesses (shippers) and truck operators.

This inefficiency comes from four key issues in the existing system (both manual and current online apps):

1. Poor Visibility & Fragmentation:
    - Old Way: Finding a truck relies on many phone calls and middlemen (brokers), making it hard to see all available options and rates. It's time-consuming, and you often don't get the best deal.
2. Unfair and Non-Standard Pricing:
    - Problem: Prices are often negotiated, leading to inconsistent, non-standardized rates and hidden charges for both shippers and transporters, making budgeting difficult and eroding trust.

3. Inefficient Matching & Empty Runs:
    - Problem: It's hard to match the exact cargo type and size with the right vehicle.
    - Truckers' Issue: Trucks often have to drive long distances empty after dropping off a load because they can't easily find a return job, which wastes money and time. This lack of an intelligent matching system is a major gap.
4. Limits of Existing Apps:
    - Problem: Even newer online platforms have issues like not enough customization (e.g., no filter for refrigerated trucks), they can be unreliable (frequent downtime), and they often have poor customer support. Also, many only handle local deliveries, leaving a gap for long-haul and specialized fleet needs.

2.2 Proposed System

The proposed Commercial Vehicle Booking System will be a unified online platform accessible via a website or a mobile application. Its core purpose is to directly connect businesses and individuals (shippers) with commercial vehicle operators, thus moving away from fragmented, manual methods and solving the limitations of current apps. The system will overcome the problems of limited visibility, non-standardized pricing, and inefficient booking delays that plague the current logistics landscape. It will offer transparent pricing, real-time availability and scheduling, and vehicle tracking as key features to ensure a seamless experience.

The primary goal is to offer users convenience, time savings, and cost efficiency by centralizing vehicle information and automating the booking process. The system will serve two main targeted user groups: Businesses/Individuals (Shippers) who need to transport goods, and Commercial Vehicle Operators/Drivers who provide the transport service. The platform will benefit users by providing a wide selection of vehicles and a simple booking process. It will also

simplify the administrative burden for operators by allowing them to manage their profiles and bookings through a self-service model.

2.3 Functionalities of Proposed System

- **Simple Online Booking:** The platform will be user-friendly, allowing shippers to easily find vehicle information, search, compare types, prices, and schedules, and complete the booking all in one place. This solves the problem of needing to call multiple places for comparison.
- **Real-Time Vehicle Options:** The system will host a large database of commercial vehicles, giving users a wide selection. It will show real-time availability and scheduling to prevent booking delays caused by limited visibility.
- **Transparent Pricing and Tracking:** Key features include transparent pricing to eliminate hidden costs and build trust. Bookings will also include real-time vehicle tracking, allowing businesses to monitor their cargo.
- **Self-Service Management:** Both users (shippers) and drivers will be able to manage their own profiles and bookings. This self-service feature reduces manual work and fixes errors caused by manual record-keeping.
- **Admin Tools:** A dedicated dashboard will give administrators centralized control to manage users, drivers, and the vehicle database, ensuring the platform runs reliably and smoothly.

Chapter 3. Methodology

Building a complex, real-time application like the Commercial Vehicle Booking System requires a structured **software development methodology**. This approach is crucial not only to manage the multi-faceted nature of the project—spanning user interfaces, intelligent matching algorithms, real-time tracking, and secure pricing logic—but also to ensure timely delivery and high reliability. A defined methodology provides a framework for planning, clear communication across development and user teams, risk mitigation, and systematic testing. Given that the system must rapidly adapt to changing market demands and user feedback, particularly regarding specialized vehicle needs and seamless integration, choosing the right methodology is the foundation for successfully delivering a platform that is scalable, reliable, and truly user-centric.

3.1 Introduction

Agile methodology in the Commercial Vehicle Booking System project involves establishing a structured yet flexible framework to enhance development efficiency and responsiveness to user needs. Agile methodology is an iterative and flexible approach to software development that prioritizes collaboration, customer feedback, and adaptability. Unlike traditional linear models, Agile promotes the division of projects into smaller, manageable units called iterations or sprints, allowing teams to deliver functional increments of the product regularly. This approach fosters continuous improvement, as teams can adjust their work based on ongoing feedback from stakeholders. Agile emphasizes the importance of cross-functional teams, where members with diverse skills work together to enhance creativity and efficiency. By embracing change and maintaining open communication, Agile methodology ensures that the final product aligns closely with customer needs and market demands, ultimately leading to higher satisfaction and better quality outcomes.

3.2 Software Tools

Table 3.1: List the software tools or languages used for the project development

Operating System	Windows 10/11
Front End	JavaScript, HTML, CSS,Bootstrap…
Back End	Python
Framework	Django
Database	MySQL, SQLite …
IDE	Visual Studio Code
Version Control	Git
Library	pillow

3.2.1 Python

Python was chosen as the primary programming language for the Commercial Vehicle Booking System due to its simplicity, readability, and extensive library support, making it ideal for the logistics domain. As a high-level language, Python allows developers to write clear and concise code, which is essential for maintaining a complex system involving multiple user types and real-time interactions [6]. Furthermore, Python has robust libraries and frameworks (like Django) that are crucial for managing real-time data streaming (necessary for **Vehicle Tracking**), implementing complex business logic for **Transparent Pricing**, and securely handling the large database of commercial vehicles and user profiles. This efficiency in development enables faster iteration cycles and allows the team to focus on building core functionalities, such as the **Easy Booking Platform** and **Self-Service Tools**, rather than dealing with complex syntax or low-level programming issues.

3.2.2 Django

Django is a high-level Python web framework that facilitates rapid development and clean, pragmatic design. It provides an array of built-in features, including an ORM (Object-Relational Mapping) for robust database interactions, comprehensive user authentication (necessary for both

shippers and operators), and an automatically generated administrative interface, which significantly accelerates development time for the **Administrative Management Dashboard** [7]. By using Django, the project benefits from its "batteries-included" philosophy, allowing for a structured architecture that can easily accommodate the complex and evolving requirements of a live booking system, such as managing real-time data for **Vehicle Tracking** and processing logistics data for **Transparent Pricing**. Additionally, Django's emphasis on security helps protect sensitive user and financial data and reduces vulnerabilities, which is crucial for an **Online Platform** handling commercial transactions and personal information.

3.3 Module Description

System modules represent distinct, manageable sub-components or functional units of a larger software system. They encapsulate specific functionalities, allowing for organized development, easier debugging, and enhanced maintainability.

3.3.1 User Module

The User Management Module is responsible for handling user authentication and authorization within the commercial vehicle booking application. It allows users (such as logistics managers, fleet operators, or individual business owners) to register, log in, and manage their organizational and personal profiles.

This module is crucial for allowing users to set their preferences related to **vehicle type, route optimization, driver requirements, and notification settings for booking status updates.**

The key functionalities of this module are:

- **User Registration:** Allows new users or businesses to create an account, including capturing relevant organizational details (e.g., company name, tax ID, primary contact).
- **User Login/Logout:** Authenticates users (often with role-based access for different levels within an organization) and manages secure session states.
- **Profile Management:** Enables users to update their personal and organizational information, manage associated payment methods, view past booking history, and **set**

their default preferences for commercial vehicle bookings (e.g., preferred vehicle tonnage, common routes, or specific compliance documentation requirements).

- **Example Code Snippet**:

```
def index(request):
   if request.method == 'POST':
      name = request.POST.get('name')
      email = request.POST.get('email')
      password = request.POST.get('password')
      image = request.FILES.get('image')
      address = request.POST.get('address')
      phone = request.POST.get('phone')
      if User.objects.filter(email=email).exists():
         msg='User Already Exists'
         return render(request, 'vehiapp/index.html',{'msg':msg})
      else:

         user = User(name=name, email=email,password=password, image=image, address=address,
                  phone=phone)
         user.save()
         messages.success(request, 'User registration successful!')
```

3.3.2 Driver Module

The **Driver Module** is essential for managing all aspects related to the drivers operating the commercial vehicles within the booking application ecosystem. This module ensures that only qualified and authorized drivers are assigned to bookings, and it facilitates efficient communication and trip management.

The key functionalities of this module are:

- Manages **Driver Registration:** (capturing license/vehicle details for **mandatory verification**).
- **Login** : (setting real-time **on/off-duty availability** for trip assignment).
- **Profile Management** (updating payouts and renewing **compliance documents**).

- **Example Code Snippet**:

```
def driver_registration(request):
```

```
 if request.method == 'POST':
      name = request.POST.get('name')
      email = request.POST.get('email')
      license = request.FILES.get('license')
      password = request.POST.get('password')
      image = request.FILES.get('image')
      address = request.POST.get('address')
      phone = request.POST.get('phone')
      if Driver.objects.filter(email=email).exists():
         msg='User Already Exists'
         return render(request, 'vehiapp/driverreg.html',{'msg':msg})
      else:

         driver = Driver(name=name, email=email, license=license, password=password, image=image, address=address, phone=phone)
         driver.save()
         messages.success(request, 'Driver registration successful!')
         return redirect('/')
```

3.3.3 Admin Module

The **Admin Module** provides a comprehensive dashboard for administrators to **manage users, drivers, vehicle inventory, and bookings**. It is the central control hub that ensures the smooth, compliant, and efficient operation of the entire commercial vehicle booking platform.

The key functionalities of this module are:

- **User and Driver Management:** Allows administrators to **verify, activate, or deactivate** user and driver accounts; manage their **compliance documentation**; and set their access **roles and permissions**.
- **Vehicle and Inventory Management:** Enables the administration team to add, edit, and categorize the **fleet inventory** (e.g., trucks, trailers, specialty vehicles); track **vehicle maintenance schedules**; and manage **licensing and insurance renewals**.
- **Booking and Dispatch Oversight:** Provides a real-time view of all active and past **bookings, routes, and trip status**; allows for **manual assignment/reassignment** of drivers and vehicles; and facilitates **fare and pricing configuration**.

3.4 User Story

User Story ID	As a type of User	I want to <Perform some task>	So that i can <Achieve Some Goal>
1	USER	Register	Access the booking system and personalize my experience
2	USER	Login	securely access my profile and booking history
3	USER	View Profile	View users profile in application
4	ADMIN	manage user accounts (CRUD)	control system access and assist users with their profiles
5	ADMIN	manage vehicle inventory (CRUD)	(CRUD)add, update, and remove vehicles available for booking
6	ADMIN	Manage booking records	Oversees all booking and assist with modifications
7	USER	View Profile	View users profile in application

8	USER	search for available commercial vehicles	find a suitable vehicle for my transport needs
9	USER	View detailed information about a vehicle	make an informed decision before booking
10	USER	book a commercial vehicle	reserve a specific vehicle for a defined period
11	ADMIN	view an administrative dashboard	get an overview of system activity, bookings, and users
12	ADMIN	send system-wide notifications	inform users about important updates or announcements

3.5 Product Backlog

ID	NAME	PRIORITY <high/medium/low>	ESTIMATE (Hours)	STATUS <Planned/In progress/Completed>
1	User Authentication & Profiles	High	23	completed
2	Core Database Infrastructure	High	13	completed
3	Basic Backend Services	High	20	completed
4	Initial Frontend Setup	MEDIUM	10	completed
5	Admin Vehicle Management (CRUD)	HIGH	10	completed

6	Vehicle Search Functionality	High	23	completed
7	Detailed vehicle information	High	13	completed
8	Vehicle data APIs	High	20	completed
9	Search & Listing Testing	MEDIUM	10	completed
10	Vehicle Booking Process	HIGH	20	completed
11	Booking Management APIs	High	23	completed
12	Booking Confirmation &history	High	13	completed

13	Admin Dashboard	MEDIUM	20	completed
14	Admin User & Booking Control	HIGH	10	completed
15	Responsive Design & UX Review	MEDIUM	10	completed

3.6 Project Plan

User StoryID	Task Name	Start Date	End Date	Days	Status
10	SPRINT 3	03/10/2025	06/10/2025	15	completed
11		07/09/2025	10/10/2025		completed
12,13		13/09/2025	20/09/2025		completed

15	SPRINT 4	28/9/2025	5/10/2025	15	completed
4,6		7/10/2025	11/10/2025		completed
14,16		08/10/2025	08/10/2025		completed

3.7 Sprint Backlog

Backlog item	**Status And Completion Date**	**Original Estimation in Hours**	**Day 1 hrs**	**Day 2 hrs**	**Day 3 hrs**	**Day 4 hrs**	**Day 5 hrs**	**Day 6 hrs**	**Day 7 hrs**	**Day 8 hrs**	**Day 9 hrs**	**Day 10 hrs**
SPRINT1												
Design user interface /registration	10-07-25	8	2	2	0	1	2	1	0	0	0	0
Database setup	16-07-25	10	1	1	1	1	2	2	1	1	0	0
Basic backend API	24-07-25	8	1	1	1	1	1	1	1	0	1	0
SPRINT 2												

Vehicle data management	30-07-25	10	2	1	1	2	1	2	1	0	0	0
Vehicle search functionality	07-08-25	8	1	2	1	1	1	1	1	0	0	0
Vehicle details page	13-08-25	10	1	2	1	2	2	1	1	0	0	0
Vehicle API development	27-08-25	12	2	2	2	2	2	2	0	0	0	0
SPRINT 3												
Booking form implementation	05-09-25	7	1	1	1	1	1	1	1	0	0	0
Booking API development	10-09-25	7	1	1	1	1	1	1	1	0	0	0
Booking confirmation	20-09-25	10	1	2	2	2	1	1	0	0	0	1
SPRINT 4												

Admin dashboard	25-09-25	7	1	1	1	1	1	1	1	0	0	0
User &Booking Management(admin)	05-10-25	7	1	1	1	1	1	1	1	0	0	0
Responsive Design & UX Review	08-10-25	5	1	1	1	1	0	0	0	0	1	0
TOTAL		102	13	17	14	17	15	15	9	1	2	1

3.8 Database Design

Database Used is MySQL: Which is a widely used open-source relational database management system (RDBMS) that is based on Structured Query Language (SQL). MySQL is a relational database, which means it organizes data into tables with rows and columns. Each table represents an entity, and relationships can be established between different tables. MySQL uses SQL (Structured Query Language) for defining and manipulating the data. SQL is a standard language for interacting with relational databases, allowing users to perform tasks such as querying data, updating records, and defining database structures.

Field	Data type	Constraint	Description

User_id	INT	**PRIMARY KEY**, NOT NULL	Unique username
User_address	VARCHAR(100)	NOT NULL	Contact address
phone	VARCHAR(15)	NOT NULL	Contact phone number.
email	VARCHAR(100)	UNIQUE, NOT NULL	Contact email address.
Password	Varchar(200)	Not null	User password

Table 3.8.1: Login User table

Feild	Data Type	Constraint	Description
driver_id	INT	**PRIMARY KEY**, NOT NULL	Unique driver.
User_address	VARCHAR(100)	NOT NULL	Contact address
phone	VARCHAR(15)	NOT NULL	Contact phone number.
email	VARCHAR(100)	UNIQUE, NOT NULL	Contact email address.
Password	Varchar(200)	Not null	User password

license_number	VARCHAR(20)	UNIQUE, NOT NULL	Drivers liscence number

Table 3.8.2: Login driver table

Field	Data Type	Constraint	Description
Admin_id	INT	**PRIMARY KEY**, NOT NULL	Unique driver.
password	VARCHAR(100)	NOT NULL	Contact address

Table 3.8.3: Login Admin table

Chapter 4. Results and Discussions

This chapter presents the operational results of the proposed **Commercial Vehicle Booking and Tracking System**. It includes screenshots of the developed interfaces and modules that highlight the major functionalities crucial for efficient logistics management. Each figure is explained briefly to demonstrate how the interface works and how users (Customers/User,Driver administrators) interact with different components of the system.

4.1 Results

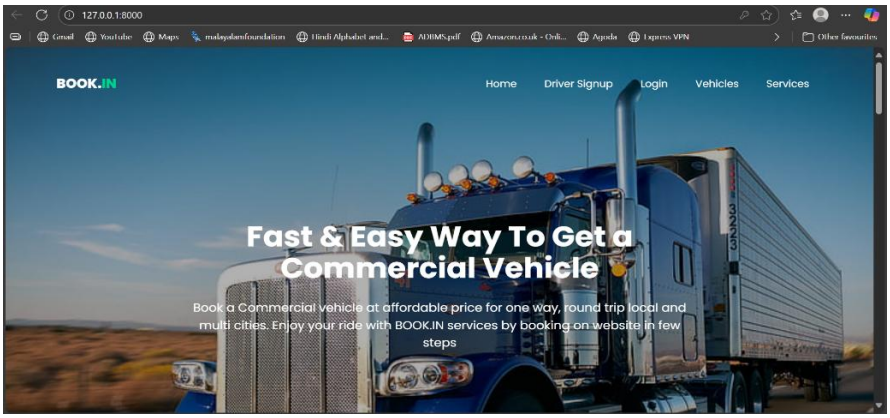


Figure 4.1: Home Page

Figure 4.1 gives the landing page of the website. This is a public web page where any one see. This web page provides a brief introduction to BOOK.IN and act as a navigation page to login.

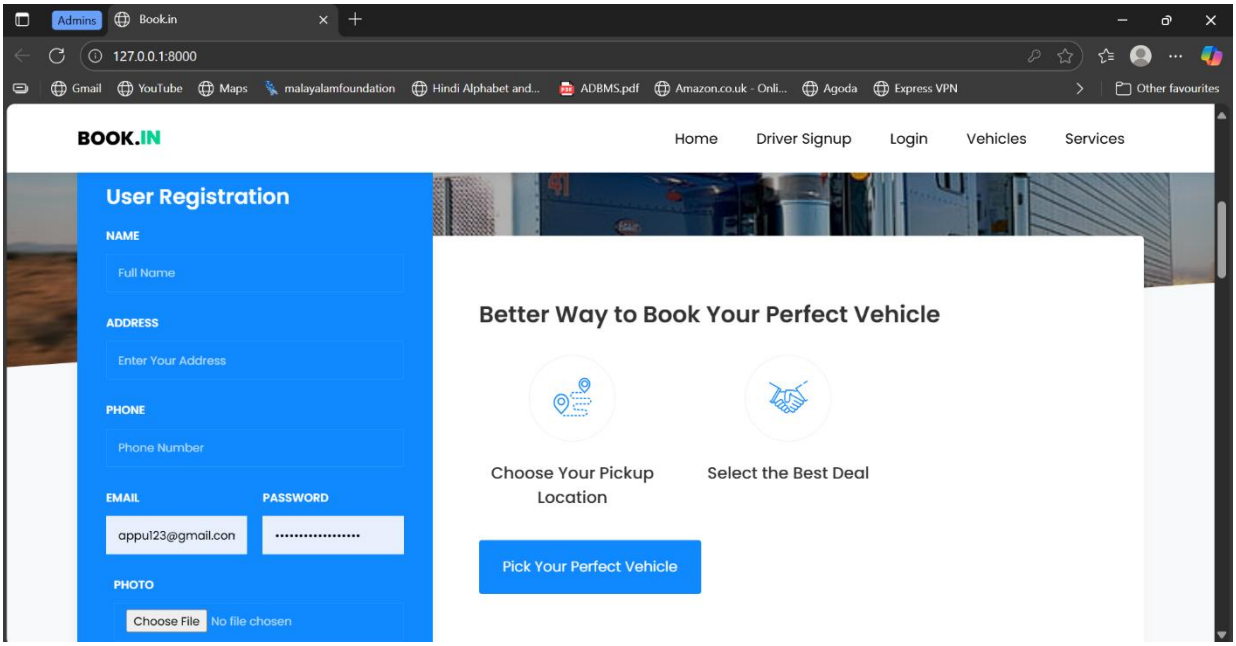


Figure 4.2: user registration

The above Figure (4.2) is the Home/Registration Page of the BOOK.IN Commercial Vehicle Booking and Tracking System. This page serves two main functions: user registration and providing an immediate path to the primary booking function.

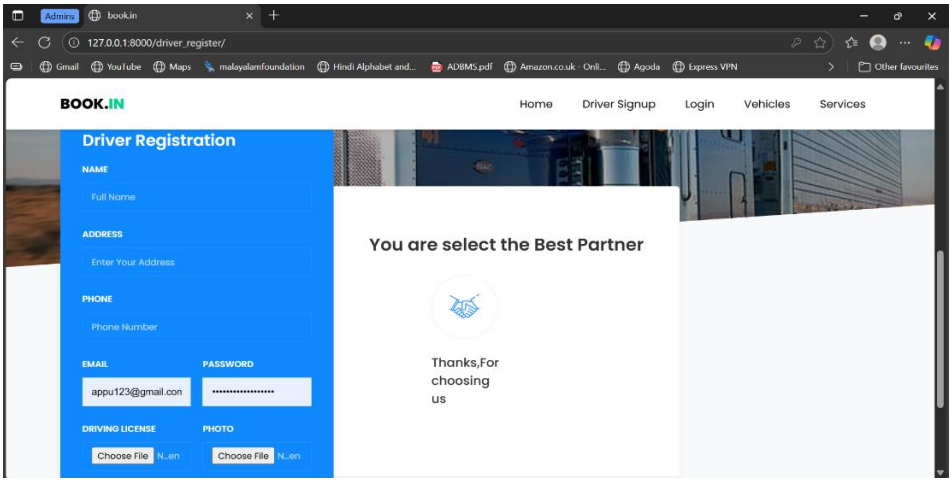


Figure 4.3: driver registration

The above Figure (4.3) is the dedicated Driver Registration Page of the BOOK.IN Commercial Vehicle Booking and Tracking System. This critical module is essential for building and maintaining the platform's supply side—the fleet of available drivers and vehicles.

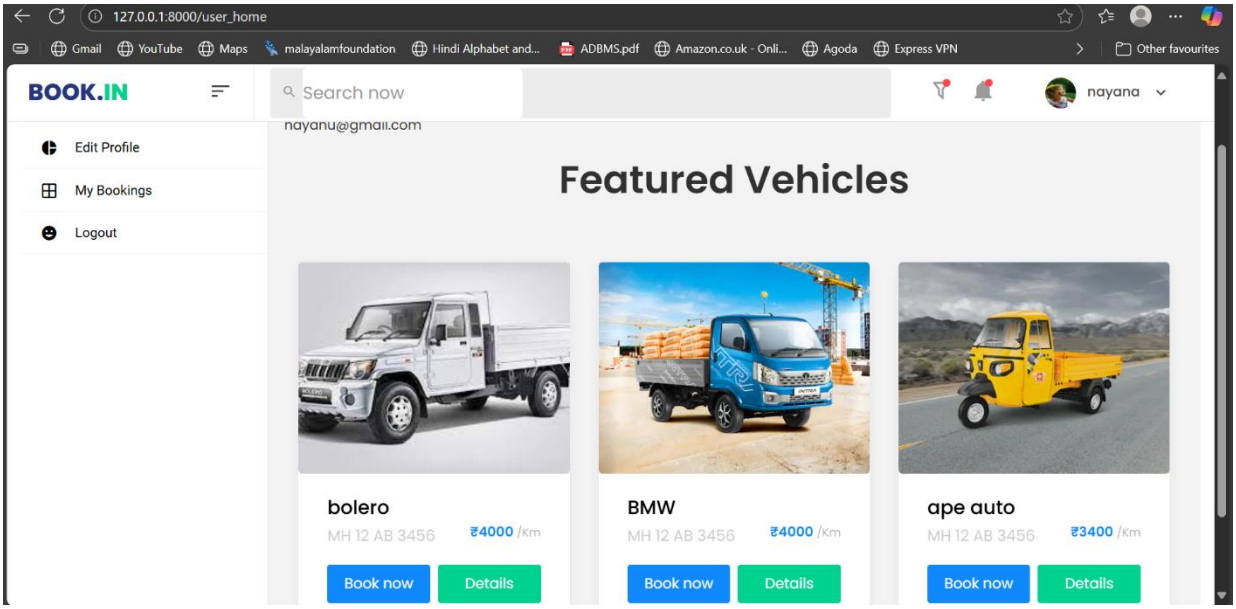


Figure 4.3: user home page

The above Figure 4.4 illustrates the **User Home Page** of the **BOOK.IN Commercial Vehicle Booking and Tracking System** once a registered user, has logged in. This page serves as the user's dashboard for browsing vehicle options, managing their account and can book the vehicle.

Chapter 5. Conclusion

The **BOOK.IN Commercial Vehicle Booking System** is an automated logistics solution built on a **Python/Django backend** designed to streamline commercial haulage. It utilizes API integration and data aggregation to provide users with essential services like user and driver registration, **real-time vehicle availability** search, and **dynamic freight quotation**. This comprehensive approach eliminates the traditional need for manual calling and negotiation, thereby helping users to **save time and optimize logistics costs**. Key features are managed through the system, including the "My Bookings" dashboard for active shipments, all contributing to a seamless user experience.

However, the system faced some initial limitations characteristic of a minimal viable product (MVP). **Network dependencies** occasionally caused delays in receiving real-time GPS coordinates for live tracking. Notification channels were restricted to in-app and email alerts, lacking critical channels like **SMS updates** for drivers or push notifications for status changes. Furthermore, the system is currently limited to its defined fleet network, requiring further development to **integrate with external third-party logistics (3PL) providers**. Future work is planned to address these constraints by expanding notification channels, enhancing data algorithms for faster GPS retrieval, and incorporating external 3PL marketplaces via API integration, ultimately making the BOOK.IN system a more comprehensive and robust tool for enhancing freight procurement and supply chain strategies.

References

[1] A. Jha, **"The Future of Logistics: Autonomous Commercial Vehicle Scheduling and Booking,"** *Journal of Supply Chain Management*, vol. 55, no. 3, pp. 45-62, 2024.

[2] J. P. Transport Solutions, **"Design and Implementation of an Enterprise Fleet Management System,"** Technical Report, 2023.

[3] T. Sharma, **"Web-Based Commercial Vehicle Booking Platform: A Case Study,"** M. Tech. Thesis, Department of Computer Science, XYZ University, June 2023.

[4] Z. Logistics, **"Optimizing Fleet Utilization and Route Planning with Real-Time Booking,"** [Online]. Available: https://www.zlogistics.com/insights/fleet-optimization/. [Accessed 25 October 2024].

Appendix

Appendix A Data Flow Diagram

LEVEL 0

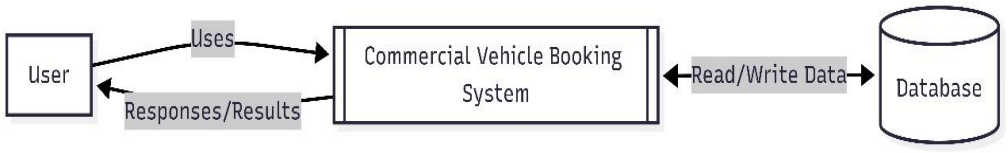


Figure 0.1 : Level 0

The above figure 0.1 is the level 0 Data Flow diagram of the project the commercial vehicle booking system.

LEVEL 1

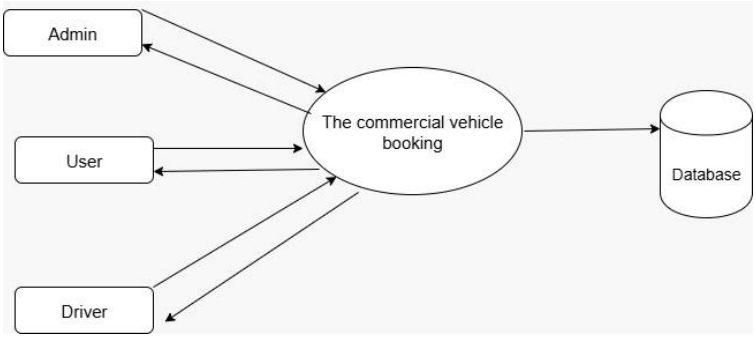


Figure 0.2 : Level 1

LEVEL 2

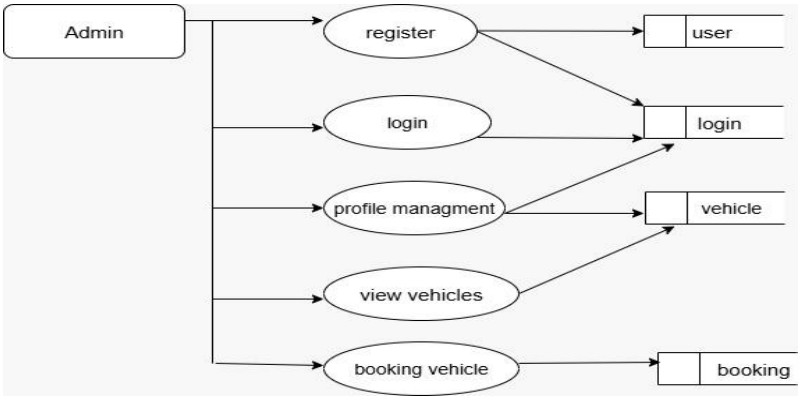


Figure 0.3: Level 2

The above figure 0.3 shows the data flow diagram of Level 2. of the commercial vehicle booking System.

LEVEL 3.

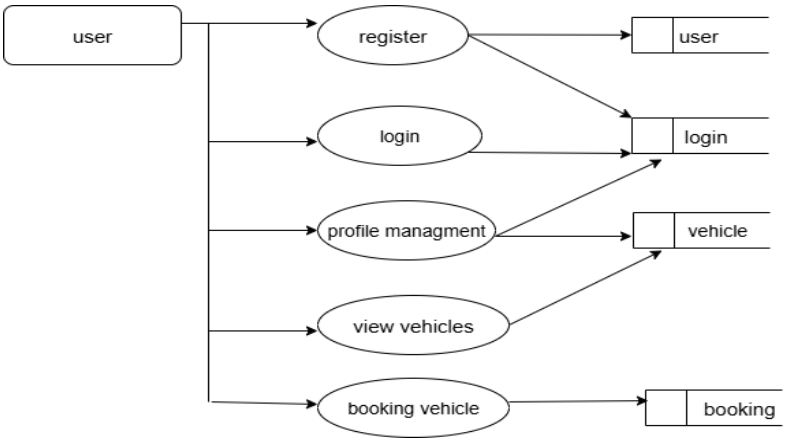


Figure 0.4: Level 3

The above figure 0.4 shows the data flow diagram of Level 3 of the commercial vehicle booking System.

LEVEL 4

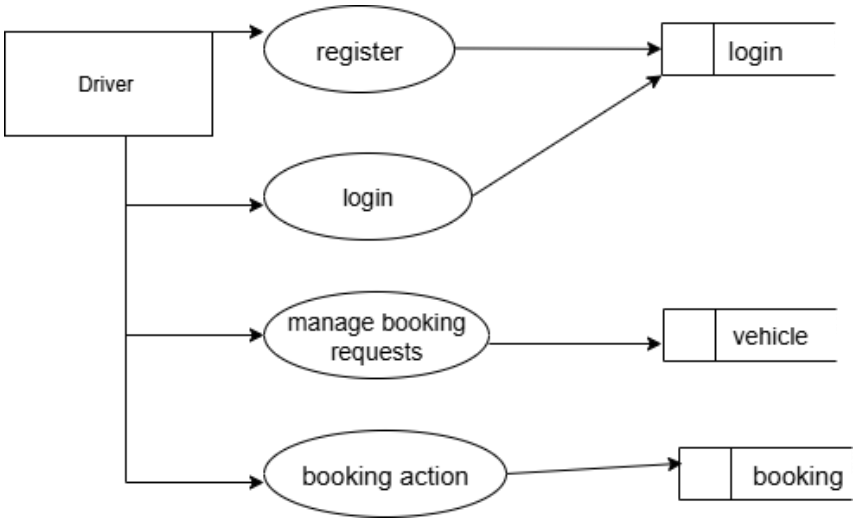


Figure 0.5: Level 4

Appendix B ER Diagram

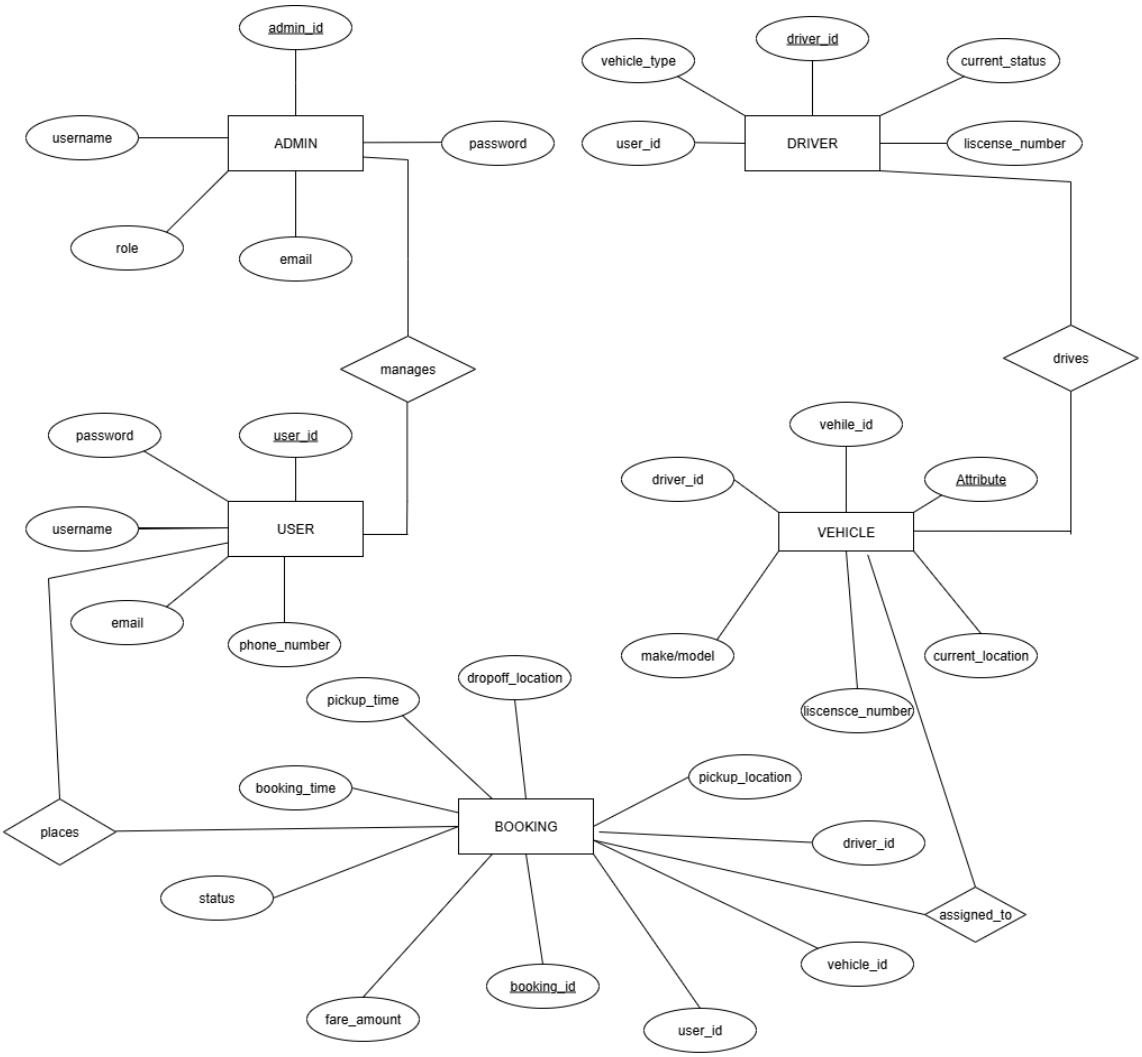


Figure 0.6: ER Diagram

The figure shows the entity diagram of customer entity in the commercial vehicle booking System.

Appendix C Source Code

Urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('detail/<int:pk>/',views.details_vehicle),
    path('vehicles',views.vehicles),
    path('driver_register/',views.driver_registration, name='driver_registration'),
    path('login',views.login,name='login'),
    path('view_license/<int:id>/', views.view_license, name='view_license'),
    path('services',views.services),
    path('user_home',views.user_home),
    path('logout/', views.logout_view, name='logout'),
    path('driver_home',views.driver_home),
    path('add_vehicle',views.add_vehicle),
    path('delete_vehicle/<int:id>/',views.delete_vehicle),
    path('filter/<int:fid>/',views.filter),
    path('drivers',views.view_drivers),
    path('search_vehicles',views.search_vehicles,name='search_vehicles'),
    path('book_vehicle/<int:vehicle_id>/',views.book_vehicle),
    path('viewbookings',views.view_booking),
    path('mybookings',views.my_booking),
    path('stats',views.view_stats),
    path('make_payment/<int:booking_id>/', views.make_payment, name='make_payment'),
    path('edit-user',views.edituser),
    path('changep-user',views.changepassword_user),
    path('edit-driver',views.editdriver),
    path('changep-driver',views.changepassword_driver),
    path('uview_drivervehicle/<int:did>/',views.view_driver_vehicles)
]
```

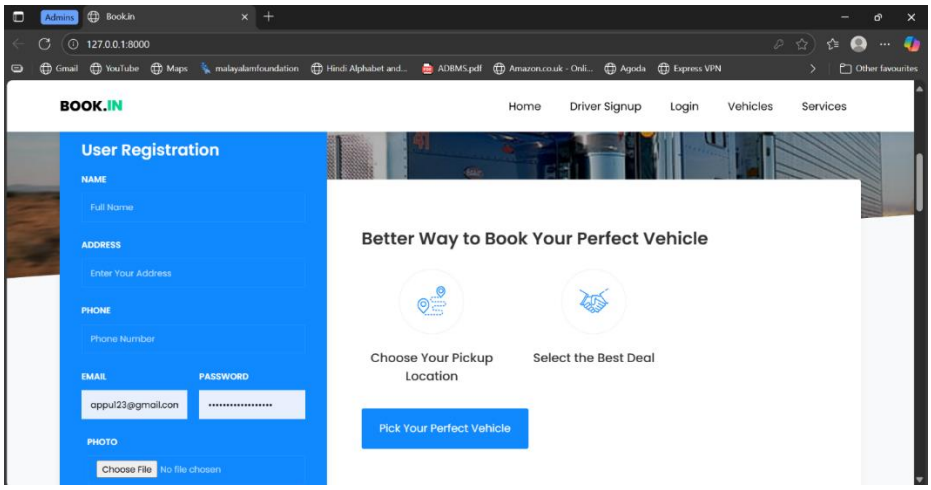
models.py

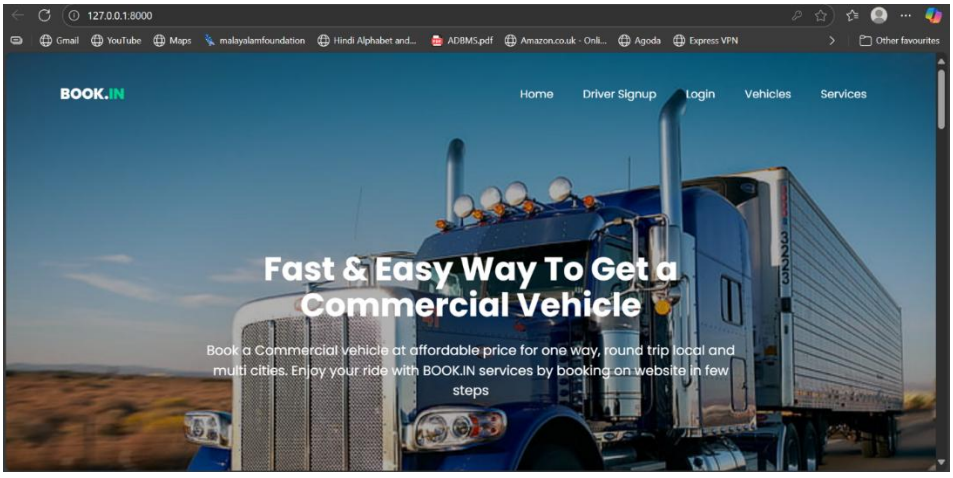
```
from django.db import models

from django.urls import path
```

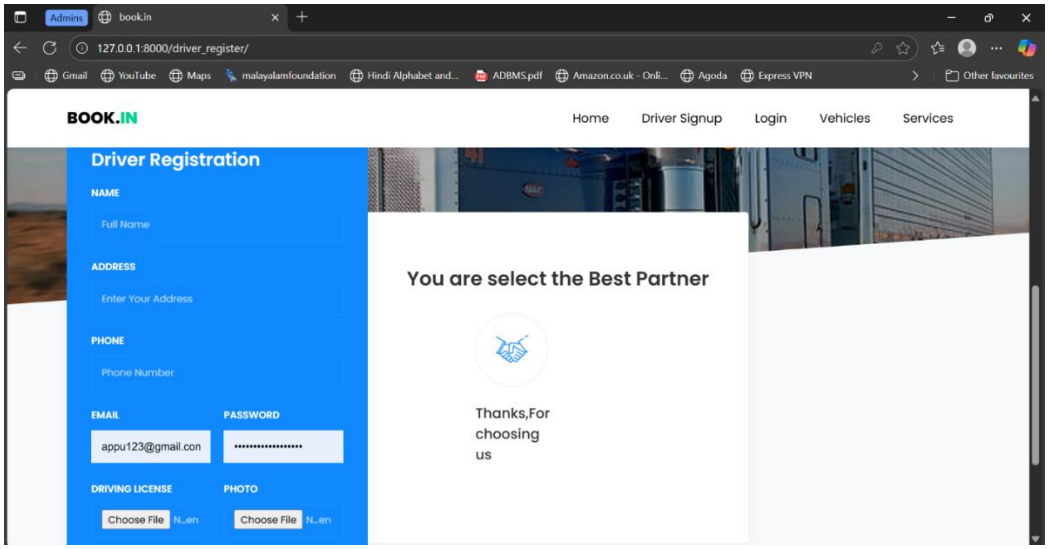
```
class User(models.Model):
    name = models.CharField(max_length=100)
    email = models.EmailField(unique=True)
    password = models.CharField(max_length=100)
    image=models.ImageField(upload_to='drimg/',default='nothing')
    address=models.CharField(max_length=100,default='nothing')
    phone=models.IntegerField(default='1')
    def __str__(self):
        return self.name



class Driver(models.Model):
    name = models.CharField(max_length=100)
    email=models.CharField(max_length=50,default='nothing')
    license = models.ImageField(upload_to='license/',default='nothing')
    password =models.CharField(max_length=18,default='nothing')
    image=models.ImageField(upload_to='drimg/',default='nothing')
    address=models.CharField(max_length=100,default='nothing')
    phone=models.IntegerField(default='1')
    status=models.CharField(max_length=20,default='Not Verified')
    is_verified = models.BooleanField('Is Verified', default=False)
    def __str__(self):
        return self.name


class vehicle(models.Model):
    userid=models.ForeignKey(Driver,on_delete=models.CASCADE)
```

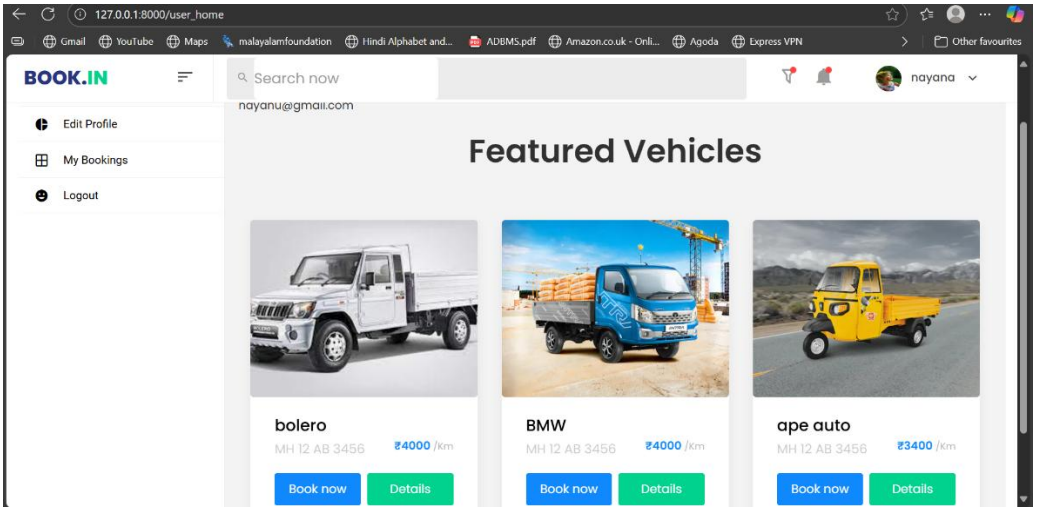
```
    vehicle_name = models.CharField(max_length=100)
    vehicle_reg = models.CharField(max_length=20)
    vehicle_type = models.CharField(max_length=500)
    vehicle_image = models.ImageField(upload_to="vehicles/", blank=True)
    rate=models.IntegerField(null=True)
    status=models.CharField(max_length=20,default="not booked")


    def __str__(self):
        return self.vehicle_name

class booking(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    driver = models.ForeignKey(Driver, on_delete=models.CASCADE)
    vehicle=models.ForeignKey(vehicle, on_delete=models.CASCADE,null=True,blank=True)
    time=models.TimeField(auto_now=True)
    date=models.DateField(auto_now=True)
    pickup_location = models.CharField(max_length=100)
    dropoff_location = models.CharField(max_length=100)
    distance=models.IntegerField(blank=True,null=True)

class Payment(models.Model):
    bookid=models.ForeignKey(booking,on_delete=models.CASCADE)
    cname=models.CharField(max_length=25)
    amount=models.IntegerField()
    cardno=models.IntegerField()
    cvv=models.IntegerField()
```

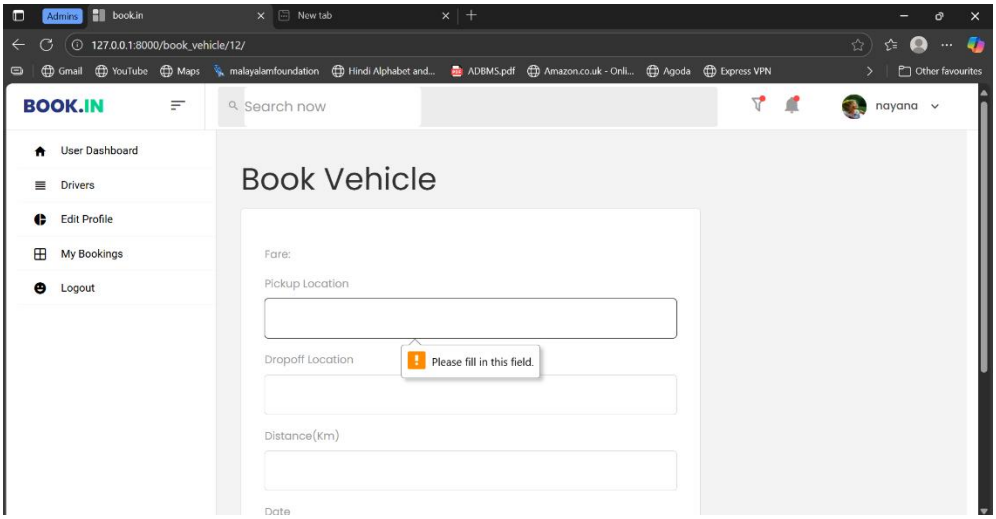
Appendix D Screenshot








127.0.0.1:8000/user_home
BOOK.IN
Search now
nayana
nayanu@gmail.com
Edit Profile
My Bookings
Logout
Featured Vehicles
bolero
MH 12 AB 3456
₹4000 /Km
Book now
Details
BMW
MH 12 AB 3456
₹4000 /Km
Book now
Details
ape auto
MH 12 AB 3456
₹3400 /Km
Book now
Details


127.0.0.1:8000/book_vehicle/12/
BOOK.IN
Search now
nayana
User Dashboard
Drivers
Edit Profile
My Bookings
Logout
Book Vehicle
Fare:
Pickup Location
Dropoff Location
Please fill in this field.
Distance(Km)
Date