TRAVELIX

Mini Project Report

Submitted by

Muhammed Hashim M

Reg. No.: AJC22MCA-2064

In Partial fulfillment for the Award of the Degree of

MASTER OF COMPUTER APPLICATIONS

(MCA TWO YEAR)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2023-2024

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, "**TRAVELIX"** is the bona fide work of **MUHAMMED HASHIM M (Regno: AJC22MCA-2064)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Dr. Paulin Paul
Internal Guide

Ms. Merra Rose Mathew
Coordinator

Rev. Fr. Dr. Rubin Thottupurathu Jose
Head of the Department

DECLARATION

I hereby declare that the project report **“TRAVELIX”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

Date: 04/12/2023	**MUHSMMED HASHIM M**
KANJIRAPPALLY	**Reg: AJC22MCA-2064**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Dr. Paulin Paul** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

MUHAMMED HASHIM M

ABSTRACT

.

Travelix is a user-friendly digital platform designed to automate a company's transportation and logistic functions. The various functions proposed system, include managing the employees in the transportation system which includes driver authentication and supporting staff. The other functions are for various processes involved in the logistics business. The driver or supporting staff authentication is proposed to be implemented using an IOT module. The other relevant details of a logistic transportation is recorded for inventory purpose. The activities user details (passenger), where every passenger is a registered user who can view the schedule book for transportation, make payment, and view and share the live location of the journey and all the activities. Are supervised and monitored by the admin module. In a graphical representation and can also cancel the journey before the departure time. The customer experience also be rated and a feedback option is included. The total purpose system has three main modules which include admin, staff, and user. The total proposed system will include in user module, booking and reservation module, driver, maintenance staff, real-time tracking, route optimization module, payment integration module, admin module, feedback and rating module, warehouse Management module, and driver access control module. admin functionalities and transportation management functions also be completed in the mini-project.

CONTENT

List of Abbreviation

HTML	Hyper Text Markup Language
COBOL	Common Business Oriented Language
UML	Unified Modelling Language
CSS	Cascading Style Sheet
AJAX	Asynchronous JavaScript and XML Environment
GB	GigaBytes
SSD	Solid-State Drive
JS	JavaScript

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The Travelix, developed with Django, is a user-friendly digital platform designed to revolutionize transportation and logistics management. Comprising three main modules User, Admin, and Staff the system ensures a seamless experience for users through features like easy booking, real-time tracking, and payment integration. The Admin module plays a pivotal role by managing users, adding locations, buses, and scheduling charts. With advanced commitment to data privacy and security, it prioritizes authentication for drivers and staff while overseeing user interactions and feedback. The scalable design of the system caters to the growing demands of users securely. In conclusion, the Travelix promises a secure and efficient solution for logistics companies, enhancing user satisfaction and operational effectiveness.

PROJECT SPECIFICATION

The Travelix aspires to establish an innovative digital platform leveraging Django for the automation and optimization of transportation and logistics processes. The project encompasses three primary modules, namely User, Admin, and Staff, each contributing to a seamless and secure transportation experience. In the User Module, individuals can easily book buses following a straightforward process of registration and login, ensuring a user-friendly journey. The Admin Module, central to the system, manages users, authenticates drivers and staff, and oversees security protocols. Meanwhile, the Staff Module is dedicated to the authentication and maintenance responsibilities of supporting staff. The system prioritizes data security through robust measures, offering a reliable and scalable solution for logistics companies. In conclusion, the Travelix endeavors to redefine transportation and logistics management, prioritizing both user satisfaction and operational efficiency.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The system study for Travelix is a comprehensive examination of the project's requirements, objectives, and technical specifications. Travelix, our innovative E-Transport System, stands at the forefront of digital transformation in transportation and logistics. With a user-centric design, it comprises three core modules Admin, Staff, and User each playing a pivotal role in automating and optimizing operations., while the User Module empowers passengers with booking, and feedback capabilities. This comprehensive system integrates cutting-edge features such as route optimization, payment integration promising a seamless experience for all stakeholders. Travelix is poised to redefine e-transport management, offering a robust solution that prioritizes efficiency, transparency, and a superior user experience in the dynamic realm of transportation and logistics.

2.2 EXISTING SYSTEM

The current state of the transportation and logistics industry is characterized by manual and fragmented processes, leading to inefficiencies and a lack of transparency. Traditional methods involve paper-based booking systems, limited real-time tracking capabilities, and minimal customer interaction. Driver and staff management often rely on manual records, posing security and accuracy challenges. The absence of a centralized platform contributes to operational silos and hinders comprehensive data analysis. Additionally, customer feedback mechanisms are limited, hindering companies from gaining valuable insights into service quality and user satisfaction.

The Travelix user Module offers passengers a user-friendly interface for seamless booking and interactive features such as payment integration, and feedback options. The transition from manual processes to a digital platform ensures data accuracy, reduces operational costs, and enhances overall efficiency. The system's feedback and rating module empowers companies to assess and improve service quality. The introduction of route optimization contributes to fuel efficiency and cost savings. Overall, Travelix not only addresses the existing gaps in the transportation and logistics sector but also propels it into a new era of innovation, connectivity, and customer satisfaction

2.2.1 NATURAL SYSTEM STUDIED

The natural system surrounding Travelix, the E-Transport Management System, encompasses the intricate dynamics of the transportation and logistics industry. In the existing natural system, manual processes and fragmented methodologies dominate, leading to inefficiencies and a lack of holistic connectivity. The shift proposed by Travelix introduces a transformative element, harmonizing with the ecosystem by leveraging digitalization and IoT technology. This integration not only addresses the inefficiencies present in the natural system but also augments it with real-time tracking, enhanced security through IoT-based authentication, and streamlined operational processes. The evolution from a conventional to a digitalized natural system heralds a new era in the transportation and logistics landscape, promising increased efficiency, reduced environmental impact, and a more interconnected and sustainable ecosystem.

2.2.2 DESIGNED SYSTEM STUDIED

The designed system for the Travelix booking transportation platform introduces a comprehensive digital framework to facilitate booking activities efficiently. With an integrated web-based platform, users can easily access and manage booking tasks, including event organization and data management. The platform's design ensures user-friendly navigation and intuitive functionalities for seamless interaction and collaboration. Through features such as personalized profiles, booking scheduling, the platform enables effective engagement and streamlined communication among scouts and administrators.

2.3 DRAWBACKS OF EXISTING SYSTEM

- Manual Processes: The existing system heavily relies on manual processes, leading to inefficiencies, errors, and time-consuming operations.
- Limited Transparency: Lack of real-time tracking and centralized control results in limited visibility into the status and location of transportation resources, hindering effective management.
- Security Concerns: Authentication and verification of drivers and supporting staff are prone to lapses in security, posing risks to the overall integrity of the transportation system.
- Operational Silos: Fragmented processes and data in the existing system create operational silos, making it challenging to achieve a comprehensive and integrated view of the entire transportation and logistics workflow.
- Customer Interaction Barriers: Traditional booking methods and minimal customer interaction limit the user experience, hindering the ability to provide personalized services and gather valuable feedback.
- Inefficient Route Planning: Lack of route optimization features results in suboptimal planning, leading to increased fuel costs and longer transit times.
- Paper-Based Records: Reliance on paper-based records for inventory and logistics purposes

not only contributes to environmental waste but also introduces the risk of data loss and inaccuracies.

- Limited Analytics: The absence of a centralized system restricts the ability to gather and analyze comprehensive data, hindering informed decision-making for continuous improvement.
- Inadequate Feedback Mechanism: The existing system lacks a robust feedback mechanism, making it challenging for companies to understand customer sentiments, address concerns, and enhance service quality.
- Resistance to Technological Advances: The reluctance to embrace modern technologies contributes to stagnation in the industry, preventing the realization of potential efficiencies and improvements achievable through digitalization and automation.

2.4 PROPOSED SYSTEM

The proposed system, Travelix, envisions a transformative leap from the drawbacks inherent in the existing transportation and logistics framework. By embracing cutting-edge technologies and a user-centric design, Travelix seeks to eradicate the inefficiencies associated with manual processes and limited transparency. it addresses security concerns, ensuring the integrity of driver and supporting staff verification. The centralized Admin Module provides optimizing route planning and streamlining operations. The proposed system introduces a seamless User Module, enhancing customer interactions through features such as live journey tracking, integrated payment systems, and a comprehensive feedback mechanism. By leveraging digitalization, Travelix not only overcomes the limitations of the current system but also sets the stage for a connected, efficient, and customer-centric future in the realm of e-transport management. Leveraging the power of Django in the backend and HTML, CSS, and Bootstrap in the frontend, the system will be highly responsive and dynamic, ensuring an intuitive and engaging user journey. Through an efficient deployment strategy, the proposed system aims to deliver a reliable and efficient platform.

2.5 ADVANTAGES OF PROPOSED SYSTEM

- Streamlined Operations: Travelix automates manual processes, minimizing errors and improving the overall efficiency of transportation and logistics operations.
- Robust Security Measures: The system's IoT-based authentication ensures secure driver and staff verification, enhancing the overall integrity of the transportation system.
- Real-Time Visibility: With centralized tracking, Travelix provides real-time visibility into the status and location of transportation resources, enabling better management and decision-making.

- User-Friendly Interface: The proposed User Module offers a user-centric design, allowing passengers to track journeys, make seamless payments, and provide feedback for an enhanced overall user experience.
- Optimized Route Planning: Travelix introduces route optimization features, reducing fuel costs and transit times, leading to cost savings and a more environmentally friendly approach.
- Data-Driven Decision Making: The centralized Admin Module enables comprehensive data analysis, empowering companies with valuable insights for informed decision-making and continuous improvement.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

The feasibility study for Travelix, encompasses a thorough examination across diverse dimensions to ensure the viability and success of the project. In evaluating economic feasibility, we analyze the cost-effectiveness of implementing Travelix against the anticipated benefits and operational efficiencies it promises. Technical feasibility involves an in-depth assessment of the system's compatibility with existing infrastructure and the integration of cutting-edge technologies, such as IoT modules and real-time tracking. Furthermore, the study explores the operational feasibility by examining how Travelix will streamline transportation and logistics processes, optimizing routes and providing real-time visibility. Lastly, user feasibility gauges the user-friendliness of the platform, ensuring that passengers, drivers, and administrative staff can easily navigate and benefit from its features. By addressing these aspects comprehensively, the feasibility study aims to provide a holistic understanding of Travelix potential to revolutionize the transportation and logistics industry.

3.1.1 Economic Feasibility

The project shows promise from an economic perspective. While there are initial development costs, these are balanced against potential cost savings and revenue generation. Ongoing maintenance costs are manageable, and the project offers long-term ROI potential.

3.1.2 Technical Feasibility

The technical feasibility of the E-Transport System is strong. It utilizes standard web technologies like Django, HTML, CSS, and JavaScript and in case using Firebase making it accessible and easy to develop and maintain. The modular architecture supports the integration of various components, ensuring scalability and adaptability.

3.1.3 Behavioral Feasibility

Travelix is a critical aspect of the overall project assessment, focusing on the system's acceptance and usability by the individuals who will interact with it. This feasibility dimension explores how well users, including passengers, drivers, and administrative staff, will adapt to and embrace the proposed E-Transport System. Travelix is designed with a user-centric approach, prioritizing a seamless and intuitive interface to ensure ease of navigation and interaction. Behavioral feasibility also considers the readiness of the involved stakeholders to embrace digital transformation in their day-to-day activities. The system aims to enhance user experiences by providing real-time tracking, convenient booking, and efficient feedback mechanisms, contributing to positive behavioral outcomes. Understanding and addressing the behavioral aspects are integral to the success of Travelix, as it strives to create a shift in user behavior towards adopting a more streamlined and technologically advanced approach to transportation and logistics.

3.1.4 Feasibility Study Questionnaire

Project Overview?

The E-Transport System is a user-friendly digital platform designed to simplify transportation and logistics. It offers an intuitive interface for passengers, shippers, and transport operators. Whether you're booking a seat or shipping cargo, the process is straightforward. A standout feature is real-time tracking, providing precise location updates for buses and cargo shipments. Behind the scenes, an admin module manages user accounts, routes, and schedules, ensuring security. For staff, including drivers and warehouse managers, the system streamlines tasks and communication.

To what extend the system is proposed for?

The proposed Travelix appears to be a comprehensive and ambitious project designed to address a wide range of transportation and logistics needs. It covers various aspects of transportation management, including passenger services, cargo transportation, driver management, real-time tracking, and administrative functions. Here's an overview of the extent to which the system is proposed:

- Warehouse Staff Management Module
- Driver Access Control

Specify the Viewers/Public which is to be involved in the System?

The Travelix is designed to cater to a diverse set of users, both within the organization managing the transportation system and external users who interact with the system's services. Here are the primary viewers/public who are intended to be involved in the system.

- Passengers
- Shippers
- Transport Operators
- Drivers
- Administrators
- Dispatchers

- Maintenance Staff
- General Public

List the Modules included in your System?

- Admin
- Staff
- User

Identify the users in your project?

My project is used in the general public, Maintenance Staff and driver.

Who owns the system?

The system is Owner by the administrator.

System is related to which firm/industry/organization?

The E-Transport System is related to the transportation and logistics industry. It is designed to streamline and optimize transportation and logistics operations, making it relevant to various firms, organizations, and industries involved in the movement of passengers and goods. Here are some of the industries and organizations that could benefit from this system

- Public Transportation Agencies
- Trucking Companies
- Travel and Tour Operators
- Schools and Educational Institutions
- Private Transportation Providers

Details of person that you have contacted for data collection?

Abhibus

Abhibus contact: 040-61656789

Questionnaire to collect details about the project? (min 10 questions, include descriptive answers, attach additional docs (e.g. Bill receipts, certificate models), if any?)

Q: What are the specific goals and objectives of the Travelix project?

A: The Travelix project aims to streamline transportation and logistics through specific goals and objectives. It seeks to automate manual processes for improved visibility, and deliver a user-friendly interface for passengers. By optimizing route planning, integrating payment systems, and enabling comprehensive data analysis, Travelix aims to revolutionize the industry, ensuring efficiency, security, and a seamless experience for all stakeholders.

Q: Who are the primary users of the system, and what are their specific needs and requirements?

A: The primary users of the Travelix system include passengers, drivers, and administrative staff. Passengers seek a user-friendly platform for booking, real-time tracking, and feedback. Drivers require secure authentication and route optimization, while administrative staff need centralized control for efficient management and data analysis.

Q: What are the essential features and functionalities that the system must include to meet user needs?

A: The system must include features such as user-friendly booking interfaces. Additionally integrated payment systems, and a centralized admin module are essential functionalities to meet the diverse needs of users and ensure efficient operations in the Travelix system.

Q: How will you ensure the security and privacy of user data within the system?

A: To ensure the security and privacy of user data, Travelix will employ robust encryption protocols, secure authentication methods, and regular security audits. Access controls and permissions will be implemented, and compliance with data protection regulations will be strictly adhered to, fostering a secure environment for user information within the system.

Q: What technology stack and development tools will you use for the front-end and back-end of the system?

A: For the front-end, Travelix will utilize modern web technologies such as HTML, CSS, JavaScript, ajax and responsive user interfaces. The back-end will be powered by a robust combination of Django for server-side development and SQLite the database to ensure scalability and flexibility in managing data within the system. Development tools like Visual Studio Code and Git will facilitate efficient collaboration and version control throughout the project.

Q: How will you handle real-time tracking and GPS integration for vehicles?

A: Real-time tracking and GPS integration in Travelix will be accomplished using advanced GPS modules and APIs, ensuring accurate location data for vehicles. The system will leverage real-time data processing to provide seamless tracking, enhancing overall efficiency and user experience.

Q: What is the system's scalability plan to accommodate potential growth and increased usage?

A: Travelix will adopt a scalable architecture by utilizing cloud-based services such as AWS or Azure, allowing seamless expansion of resources based on demand. Load balancing and horizontal scaling will be implemented to ensure optimal performance and accommodate increased usage without compromising system efficiency.

Q: What kind of testing and quality assurance processes will you implement to ensure a robust and bug-free system?

A: The Travelix system will undergo rigorous testing, including unit testing, integration testing, and user acceptance testing, to identify and rectify any bugs or issues. Continuous quality assurance processes, automated testing tools, and regular code reviews will be integral to maintaining a robust and reliable system throughout its development and updates.

Q: What is the project timeline, including milestones and deadlines for different phases of development?

A: The project timeline for Travelix involves an initial phase of requirements gathering and planning, followed by development, testing, and deployment. Milestones, including alpha and beta releases, are set with deadlines to ensure a systematic and timely progression of the project.

Q: How will you gather and incorporate user feedback and make continuous improvements to the system?

A; Travelix will implement user feedback mechanisms, such as surveys and reviews, and regularly assess analytics data. Continuous improvement will be achieved through agile development practices, incorporating user suggestions and addressing emerging needs in iterative development cycles.

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- Intel Core i5
RAM	- 8 GB or higher
Hard disk	- 256 GB SSD or higher

3.2.2 Software Specification

Front End	-	HTML5, CSS3, Bootstrap, JavaScript, jQuery
Back End	-	Django (Python)
Database	-	SQLite
Client on PC	-	Windows 7 and above.
Technologies used	-	JS, HTML5, AJAX, jQuery, Python, CSS, Django

3.3 SOFTWARE DESCRIPTION

3.3.1 Django

Django serves as a high-level web framework in Python, streamlining the creation of secure and easily maintainable websites. Developed by skilled professionals, Django simplifies many complex aspects of web development, allowing you to focus on building your application without the need to start from scratch. It's an open-source framework, meaning you can use it without any financial obligations. With a vibrant community, thorough documentation, and free or paid support options, Django was created to expedite the development of sophisticated web applications that heavily depend on data.

3.3.2 SQLite

SQLite, a popular choice for web development, is a self-contained, serverless, and open-source relational database management system. Its unique design allows it to operate without the need for a separate server process, making it an excellent option for local or client-side storage in web applications. With its widespread use and reputation for simplicity and reliability, SQLite is renowned for its seamless integration capabilities. It efficiently handles data in various formats, making it a versatile and user-friendly solution for managing your website's information. Its robust features ensure optimal performance, data integrity, and scalability, providing a solid foundation for your web development needs.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

In the realm of product and system development, the design phase marks the initial and critical stage. This creative process is instrumental in ensuring the seamless and effective functioning of any system. Design, in this context, refers to the meticulous application of diverse techniques and principles to meticulously outline a process or system, facilitating its tangible realization. It involves the comprehensive description of the intricacies of a machine or system to enable its accurate implementation. In software development, the design phase holds exceptional significance, serving as the cornerstone of creating efficient and precise software solutions. System design, specifically, involves meticulous planning for the construction of a well-optimized and reliable software framework. It signifies a shift from user-focused documentation to a more programmer-oriented approach, outlining the logical and physical design aspects of the system. This meticulous software design process ensures the seamless integration of functionality and precision, laying the foundation for a robust and user-friendly final product.

4.2 UML DIAGRAM

UML, the Unified Modeling Language, is an industry-standard visual language utilized for specifying, constructing, and documenting software system artifacts. It was established by the Object Management Group (OMG), with the initial UML 1.0 draft presented in January 1997. Unlike traditional programming languages such as C++, Java, or COBOL, UML serves as a graphical language, facilitating the creation of software blueprints. Its primary function is to provide a versatile and comprehensive modeling approach, enabling the visualization, specification, construction, and documentation of software systems. While UML finds extensive use in modeling software systems, its applications are not confined to this domain exclusively, extending to areas like process flows in manufacturing units and beyond. While UML itself is not a programming language, it supports code generation in various programming languages through UML diagrams.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

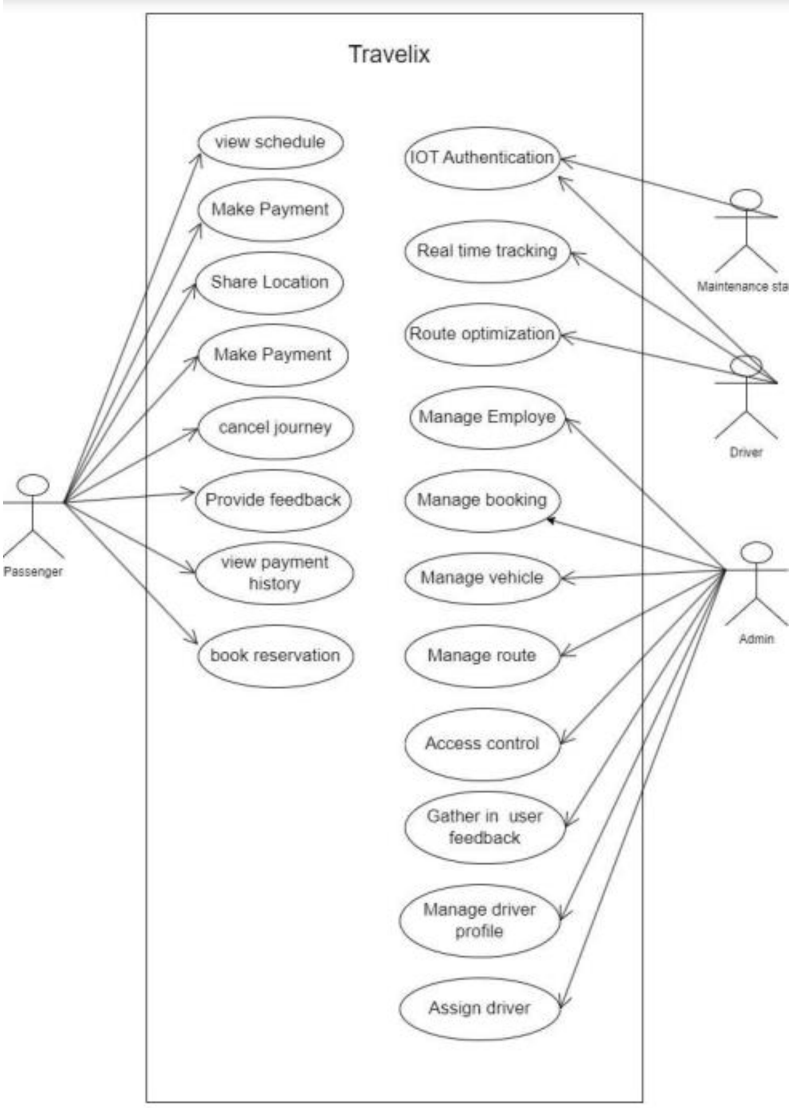
4.2.1 USE CASE DIAGRAM

Explanation

A use case is instrumental in identifying and organizing the essential requirements of a system, facilitating a structured approach to manage and prioritize these elements. Applied within the Unified Modeling Language (UML), use case diagrams serve as visual representations of real-world systems. They enable the comprehensive modeling of diverse systems, outlining their key functionalities and interactions. A use case diagram generally comprises four fundamental components, contributing to a clear and concise depiction of system requirements and behavior.

Use case diagrams aid in the documentation of a system's functional specifications. They demarcate the system's boundaries, delineating its distinctions from surrounding elements. Actors represent the individuals involved in distinct roles within the system. These diagrams facilitate a comprehensive understanding of the interactions between various entities or components within a specific context or problem. Constructing a proficient use case diagram necessitates adherence to certain rules. Ensuring appropriate nomenclature for use cases and actors is vital. The diagram should clearly illustrate relationships and dependencies while only displaying the essential connections for optimal diagram functionality. Supplementary notes may be utilized when necessary to elucidate critical points.

Diagram


Travelix
view schedule
Make Payment
Share Location
Make Payment
cancel journey
Provide feedback
view payment history
book reservation
IOT Authentication
Real time tracking
Route optimization
Manage Employe
Manage booking
Manage vehicle
Manage route
Access control
Gather in user feedback
Manage driver profile
Assign driver
Passenger
Maintenance sta
Driver
Admin

4.2.1 SEQUENCE DIAGRAM

Explanation

A sequence diagram serves to illustrate the systematic interaction between objects in a predetermined order. It elucidates the chronological flow of activities within a system. Referred to as event diagrams or event scenarios, sequence diagrams provide a comprehensive overview of the interplay among various system components, showcasing the specific sequence of their actions. These diagrams are valuable tools for both business professionals and software developers, enabling them to comprehend and depict the requirements of both new and existing systems effectively.

Notations in Sequence Diagrams:

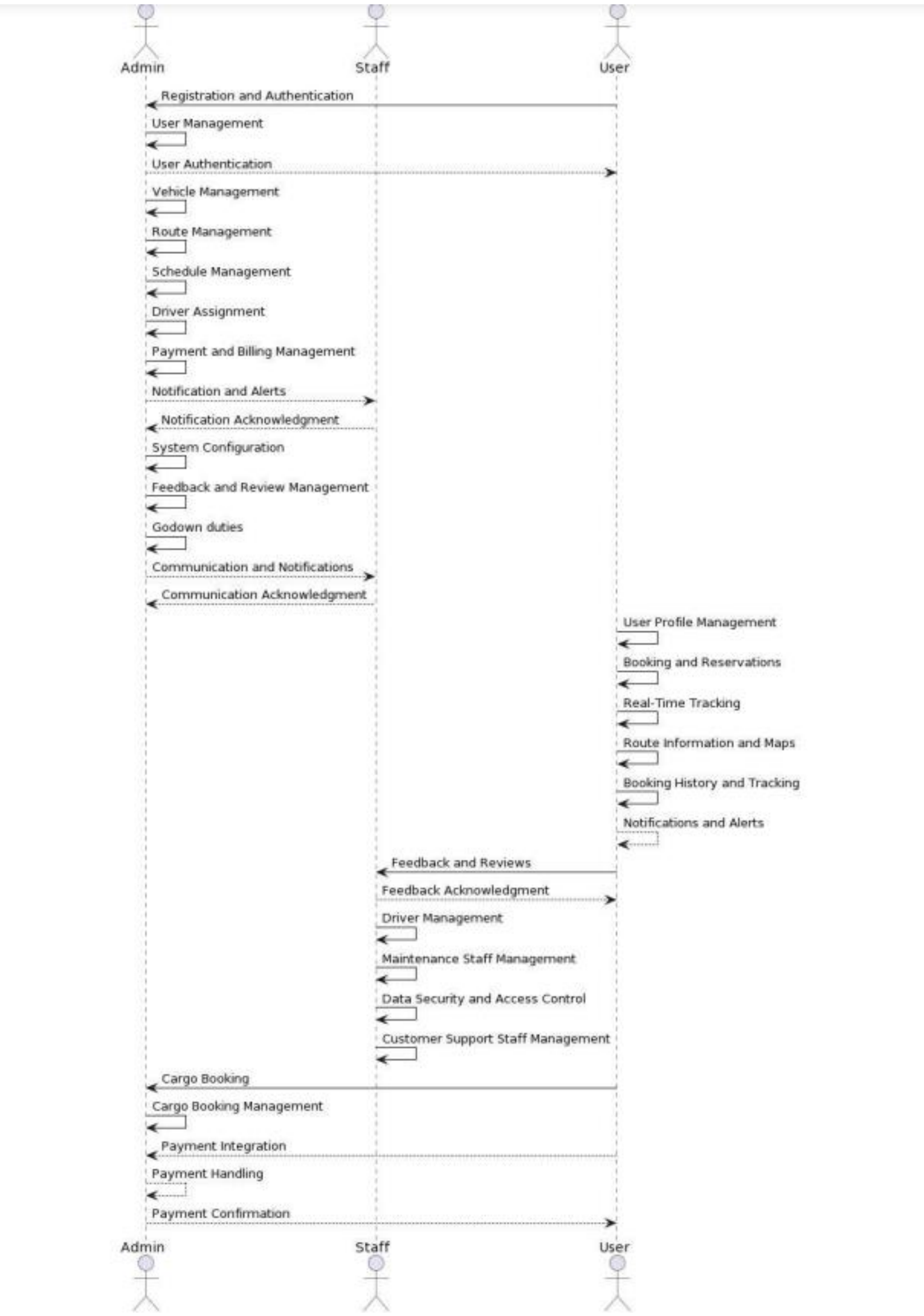
i. **Actors**: Represented as non-system individuals who utilize the system, actors play distinct roles within the diagram, depicted as simple stick figures.

ii. **Lifelines**: Each component within the system is denoted as a lifeline at the top of the sequence diagram.

iii. **Messages**: Objects communicate through the exchange of messages, depicted as arrows following the order of the lifeline. These messages form the core components of the sequence diagram.

iv. **Guards**: Utilized to define various conditions, guards restrict message flow based on specific conditions. They provide guidelines for software developers, outlining the rules and processes within the system.

Diagram


Admin
Staff
User
Registration and Authentication
User Management
User Authentication
Vehicle Management
Route Management
Schedule Management
Driver Assignment
Payment and Billing Management
Notification and Alerts
Notification Acknowledgment
System Configuration
Feedback and Review Management
Godown duties
Communication and Notifications
Communication Acknowledgment
User Profile Management
Booking and Reservations
Real-Time Tracking
Route Information and Maps
Booking History and Tracking
Notifications and Alerts
Feedback and Reviews
Feedback Acknowledgment
Driver Management
Maintenance Staff Management
Data Security and Access Control
Customer Support Staff Management
Cargo Booking
Cargo Booking Management
Payment Integration
Payment Handling
Payment Confirmation
Admin
Staff
User

4.2.2 State Chart Diagram

Explanation

The state machine diagram, also known as a state chart, effectively represents the various states of an object within a system and their sequential transitions. It provides a comprehensive overview of the system's behavioral patterns, delineating the object's progression through different states. It is instrumental in illustrating the interworking dynamics of various elements within a system, be it a group of individuals, a team, a larger collective, or an entire organization. The state machine diagram serves as a strategic blueprint, elucidating how objects evolve during different events and encapsulating the specific states that each element assumes within the system.

Notations in a State Machine Diagram:

• **Initial** state: Signifying the system's initiation, this notation is depicted as a black circle.

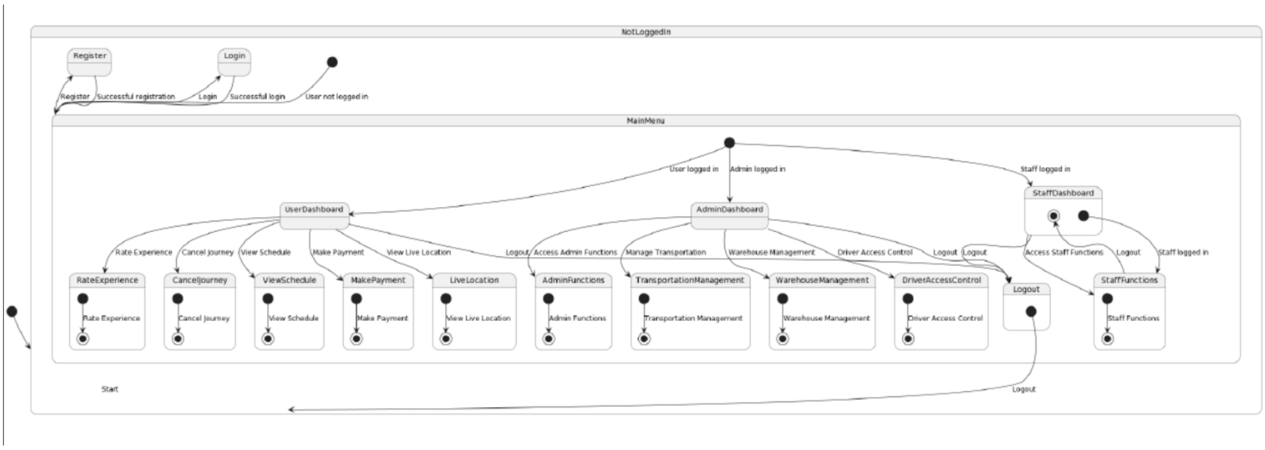
• **Final state**: Denoting the system's culmination, this notation is depicted as a filled circle within another circle.

• **Decision box**: Shaped like a diamond, it aids in making decisions based on the evaluation of a guard.

• **Transition**: Represents the shift of power or control from one state to another, illustrated as an arrow with a label indicating the cause of the transition.

• **State box**: Reflects the current status of an element within a group, represented as a rectangular shape with rounded corners.

Diagram



4.2.2 Activity Diagram

Explanation

The activity diagram visually portrays the sequential or simultaneous occurrence of events and activities, effectively illustrating how tasks unfold in a predetermined order. It serves as a comprehensive representation of the flow of activities, demonstrating the interconnectedness of different processes. The activity diagram showcases a variety of flows, encompassing actions occurring sequentially, in parallel, or along distinct pathways. To facilitate these dynamic flows, activity diagrams are equipped with tools such as fork and join, enabling the depiction of multiple simultaneous activities. This type of diagram is often referred to as an object-focused illustration, emphasizing the systematic portrayal of a process's workflow and operational intricacies.

Components of an Activity Diagram:

a) Activities: These refer to the grouping of behaviors into a sequence of actions, visually represented as interconnected nodes. The actions progress from the initial point to the end point, outlining the various tasks, control mechanisms, and resources utilized throughout the process.

b) Activity partition / swim lane: This organizational tool groups similar tasks into distinct categories, enhancing the modularity and comprehensibility of the activity diagram. It aids in the categorization and visualization of different activity sets, offering a clear representation of their interdependencies and interactions.

c) Forks: Fork nodes enable the concurrent execution of various components of a task, allowing the simultaneous progression of multiple activities. They facilitate the divergent flow of information, ensuring that data is dispersed efficiently across different pathways.

d) Join Nodes: Distinguished from fork nodes, join nodes enable the convergence of data from multiple sources, consolidating the incoming data streams into a unified output. These nodes serve to synchronize and coordinate the flow of information within the activity diagram.

Notations in an Activity Diagram:

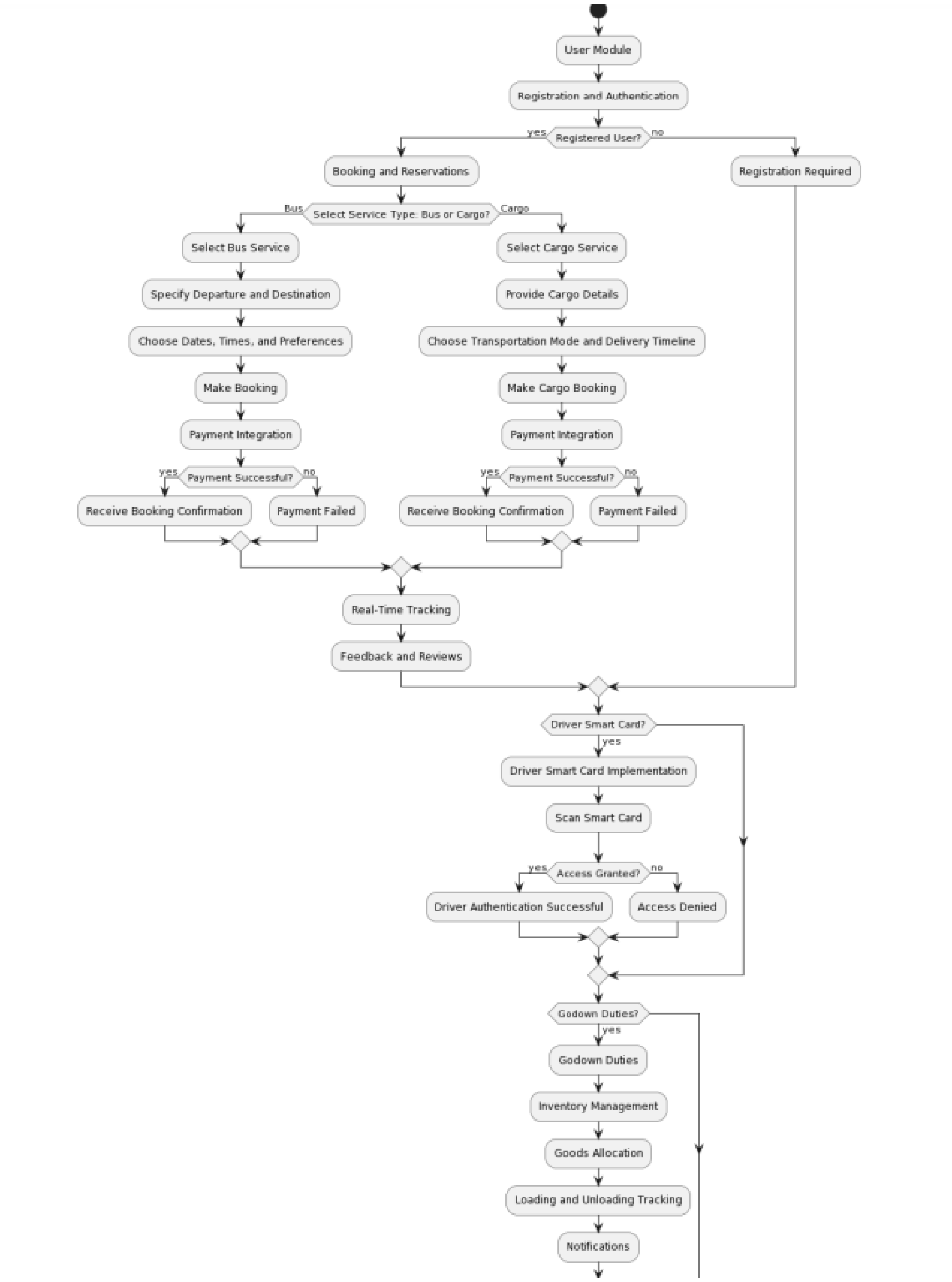
Initial State: Signifies the commencement or initial step of a process.

Final State: Represents the conclusion or endpoint of a process, indicating the completion of all associated activities.

Decision Box: Ensures that the progression of activities adheres to a predetermined course.

Action Box: Denotes the specific tasks or actions that need to be executed as part of the overall process

Diagram


User Module
Registration and Authentication
yes
Registered User?
no
Booking and Reservations
Registration Required
Bus
Select Service Type: Bus or Cargo?
Cargo
Select Bus Service
Select Cargo Service
Specify Departure and Destination
Provide Cargo Details
Choose Dates, Times, and Preferences
Choose Transportation Mode and Delivery Timeline
Make Booking
Make Cargo Booking
Payment Integration
Payment Integration
yes
Payment Successful?
no
yes
Payment Successful?
no
Receive Booking Confirmation
Payment Failed
Receive Booking Confirmation
Payment Failed
Real-Time Tracking
Feedback and Reviews
Driver Smart Card?
yes
Driver Smart Card Implementation
Scan Smart Card
yes
Access Granted?
no
Driver Authentication Successful
Access Denied
Godown Duties?
yes
Godown Duties
Inventory Management
Goods Allocation
Loading and Unloading Tracking
Notifications

4.2.3 Class Diagram

Explanation

Explanation

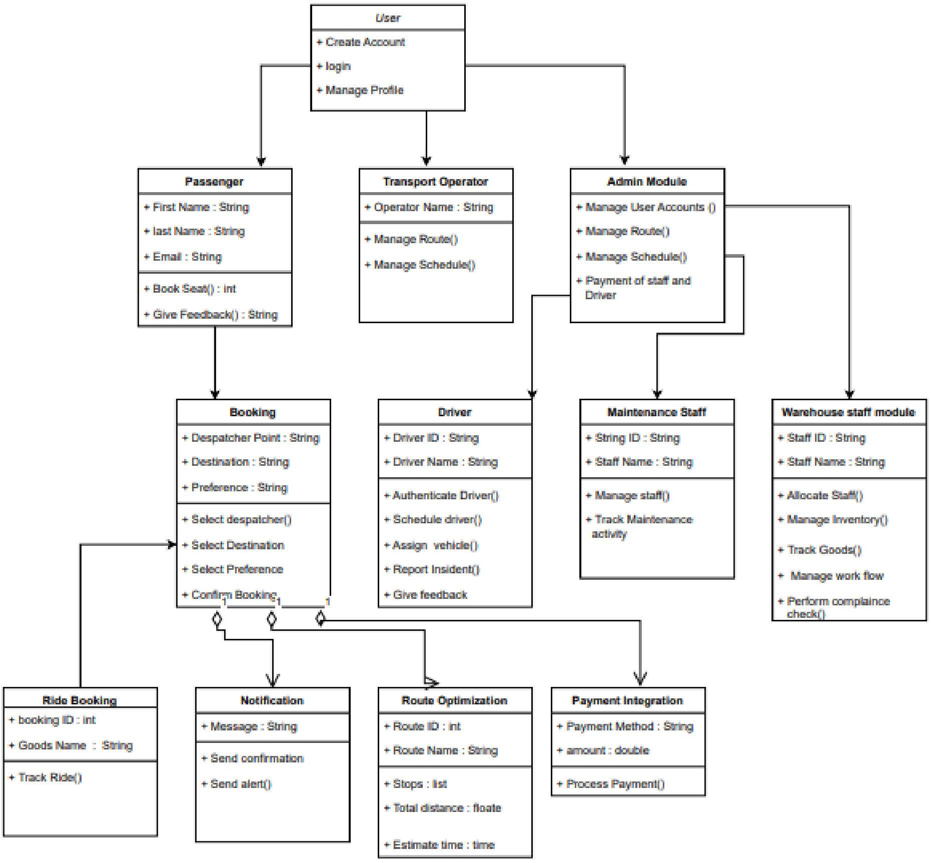
The class diagram is a static representation that illustrates the constituent elements and their associations within the system, providing a visual overview of its structure and components. It serves as a blueprint for organizing various attributes, relationships, and actions, streamlining the software development process by encapsulating critical information about the system's design and functionality. Essentially, a class diagram presents a holistic depiction of the system, offering insights into its composition, interdependencies, and behavior through a structured representation.

Components of a Class Diagram:

- **Upper Section**: This segment entails the class name, serving as a distinct identifier for a group of similar entities within the system. The class name is highlighted in bold letters, positioned centrally at the top. For an abstract class, the title is presented in slanted writing style, emphasizing its abstract nature.
- **Middle Section**: This section delineates the class attributes, demonstrating its properties and visibility factors. Attributes are depicted alongside their corresponding visibility indicators, such as public (+), private (-), protected (#), and package (~), reflecting the accessibility levels of each attribute.
- **Lower Section**: The lower segment encompasses the class methods or operations, outlined in a comprehensive list format. Each method is depicted as a discrete line item, emphasizing the interactions between the class and its associated data.
- Within the context of UML, the relationships depicted in the class diagram adhere to specific conventions:
- **Dependency**: Signifies the impact of one element's changes on another.
- **Generalization**: Illustrates the hierarchical relationship between classes, where one class acts as a parent while another assumes the role of the child.
- **Association**: Indicates the connections between different elements within the system.
- **Multiplicity**: Sets constraints on the permissible quantities associated with specific attributes or relationships.

- **Aggregation**: Represents a collection or group forming part of an association.
- **Composition**: Describes a specialized form of aggregation, emphasizing the interdependence between parent and child elements.

Diagram



4.2.4 Object Diagram

Explanation

Object diagrams are closely related to class diagrams and are derived from them. They offer a snapshot of a specific moment within an object-based system, illustrating a group of elements represented by a class. While they share similarities with class diagrams, object diagrams provide a more concrete depiction by showcasing specific instances of objects at a particular instant, enhancing the comprehension of the system's functionality and structure.

Diagram

4.2.5 Component Diagram

Explanation

A component diagram is instrumental in partitioning a complex system composed of objects into more manageable segments. It provides an overview of the internal components such as programs, documents, and tools within the node. By illustrating the connections and organization of elements in the system, it facilitates the creation of a usable system. Components are discreet sections of a system capable of autonomous function and conceal their internal operations, akin to a confidential box that operates only when used correctly.

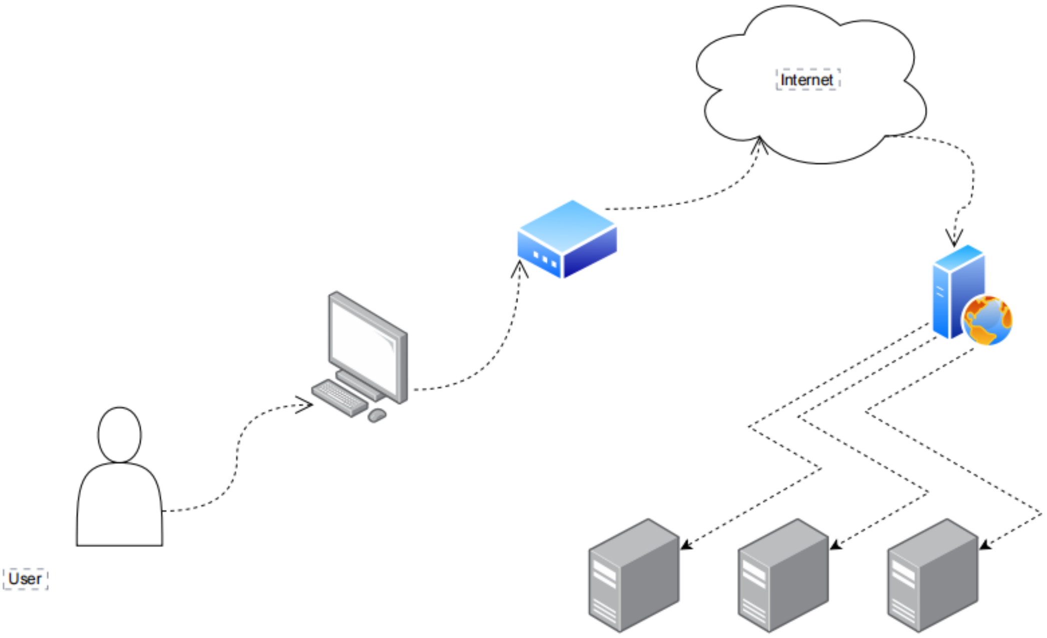
Diagram

4.2.8 Deployment Diagram

Explanation

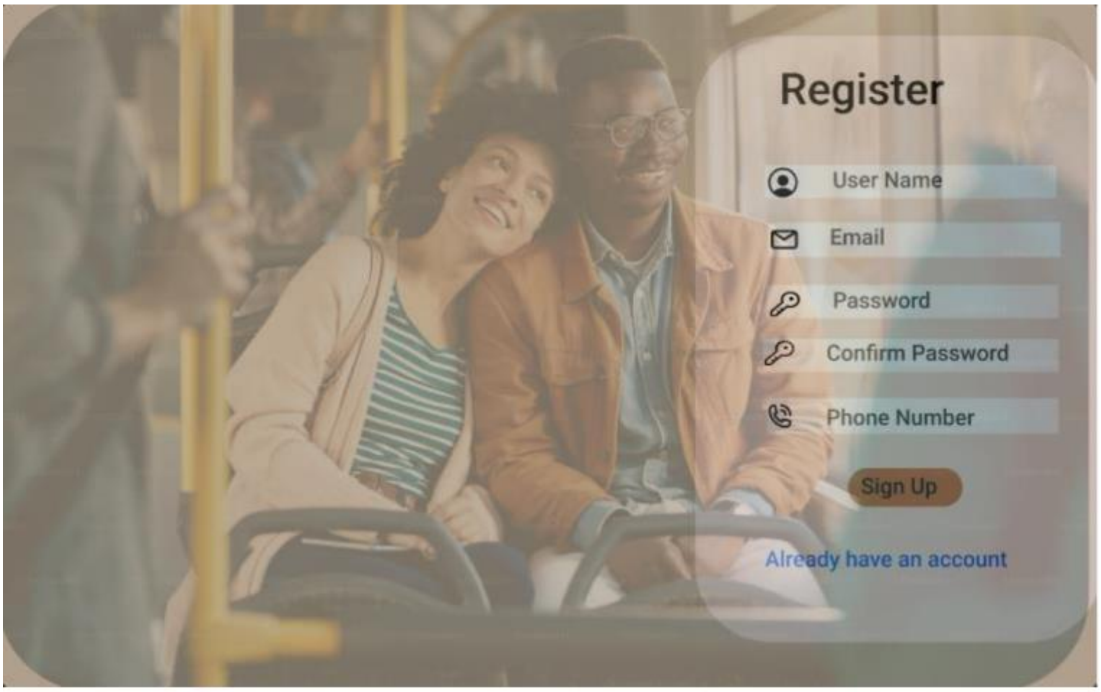
The deployment diagram visually represents the placement of software components within the physical computer or server. It provides insights into the static arrangement of the system's elements, including nodes and their interconnections. The diagram outlines the implementation of software on the computer system, detailing the architecture's composition and organization.

Diagram

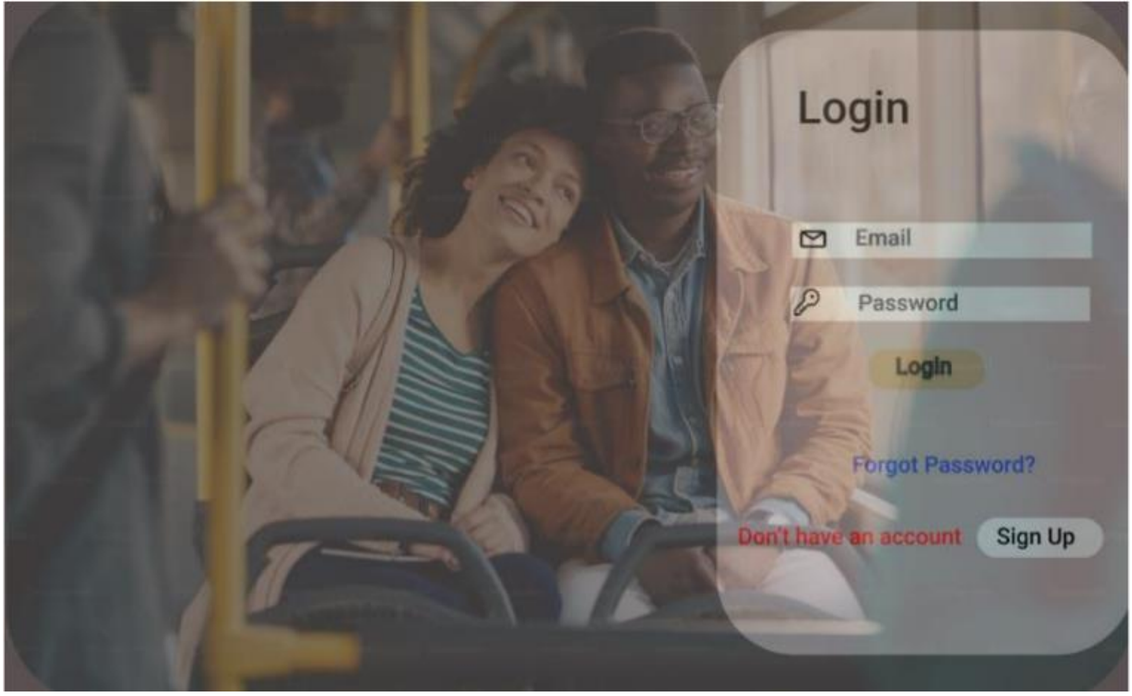


4.3 USER INTERFACE DESIGN USING FIGMA

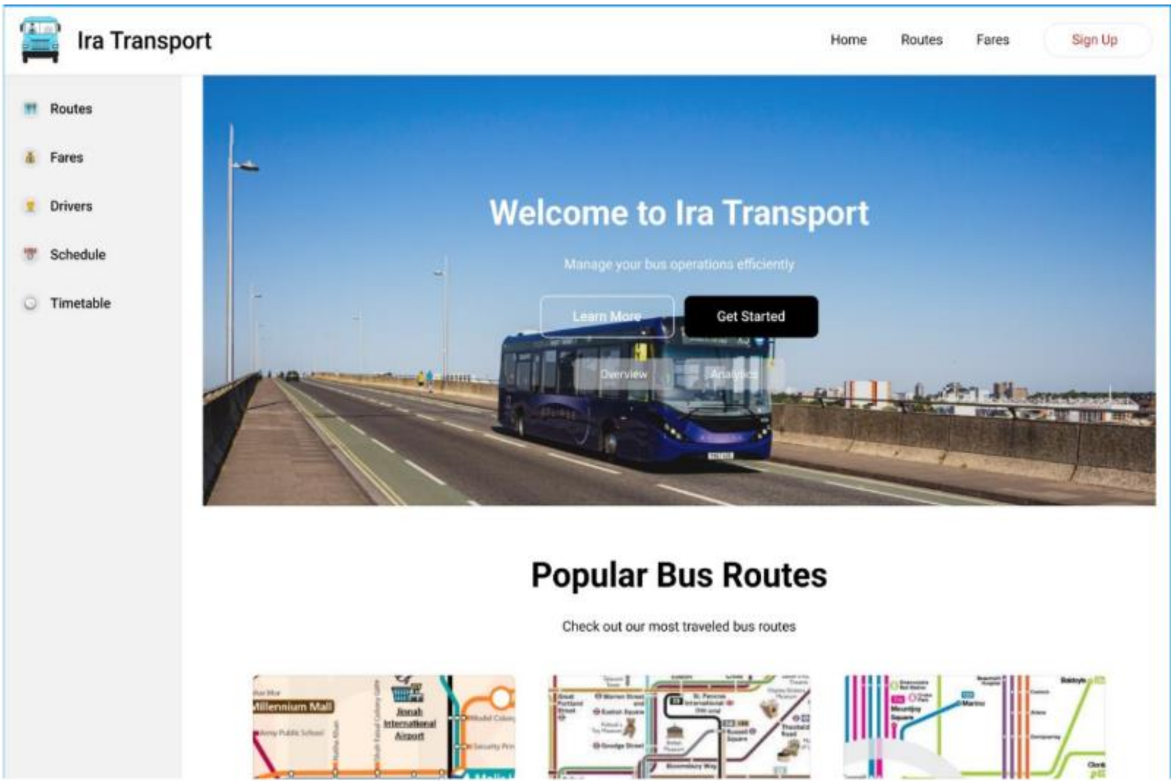
Form Name: Sign Up



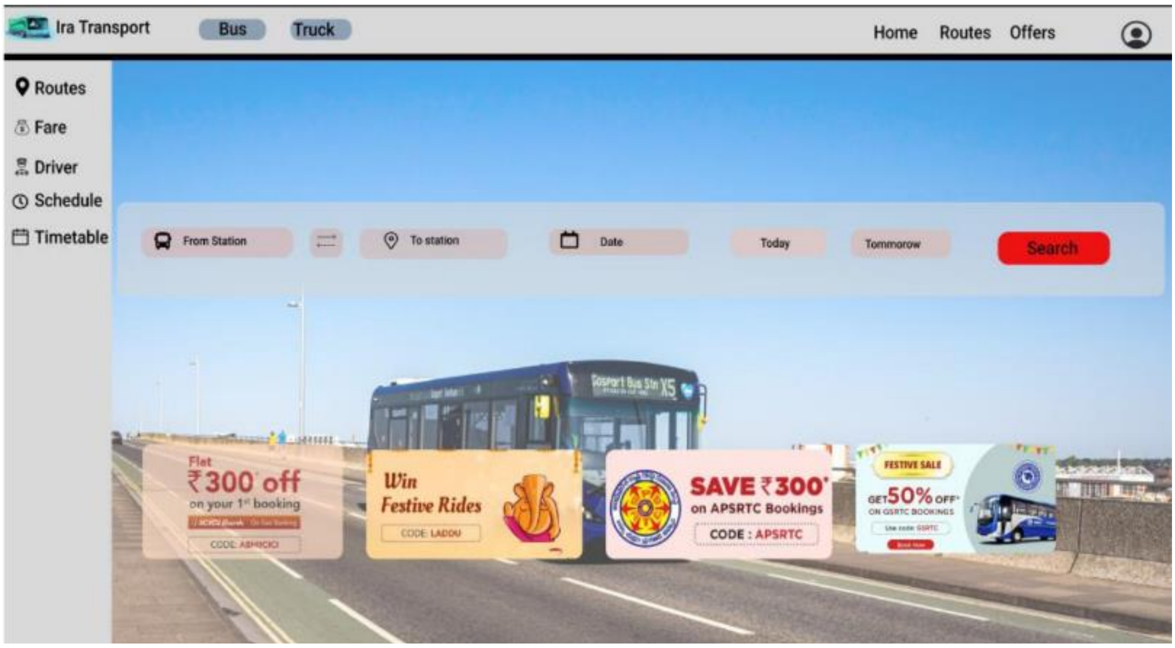
Form Name: Login



Form Name: Home



Form Name: booking



4.4 DATABASE DESIGN

4.4.1 Relational Database Management System (RDBMS)

- A relational database management system (RDBMS) is a structured system designed to efficiently store and retrieve information, ensuring its accessibility and security for users.
- Developing a database involves understanding user requirements and tailoring the database system to meet their specific needs.
- The process begins with creating an organizational plan for the data, independent of any particular computer program.
- Subsequently, this plan is translated into a specific design for the chosen computer program, focusing on the effective storage of information within the database system.
- Key aspects in this process include ensuring data integrity, maintaining the accuracy and consistency of data, and achieving data independence, allowing data to be accessed without being affected by changes in the database schema or structure.

4.4.2 Normalization

Normalization is the process of organizing a database into tables and ensuring that the data is efficiently stored and free from redundancy. It involves structuring the tables to minimize data duplication and dependency. Each table consists of rows and columns, where a row, also known as a tuple, represents a specific data set, and a column header, termed an attribute, identifies the type of data being stored. In the context of the relational model, a group of interrelated tables constitutes a relational database, facilitating effective management and utilization of data.

4.4.3 Sanitization

Django incorporates several pre-existing tools for data cleansing. It provides a variety of field types that come with built-in validation and cleaning capabilities. For instance, the Char Field automatically verifies and purifies text input, while the Email Field ensures the correct formatting of email addresses. These field types serve to prevent the storage of harmful or invalid data within the database. Moreover, Django emphasizes the use of form validation to sanitize user input. Its forms are equipped with predefined validation methods and validators that can be applied to different form fields. These validators conduct various checks and data cleansing procedures, such as confirming that numeric values fall within specified ranges and verifying the formats of uploaded files.

4.4.4 Indexing

Indexing involves organizing specific data or data groups in a database in a structured order based on their values. This ordered arrangement of index entries facilitates the quick and efficient retrieval of exact matches or data falling within a particular range. By using indexes, users can easily access information in a database without the need to search through every record during database operations. An index acts as a navigational tool for locating information within a database, enabling swift data lookup and facilitating the identification of records based on specific ordering. Additionally, an index can be established based on one or multiple columns within the table.

4.5 TABLE DESIGN

1.signup

.Primary key: **user_id**

.Foreign key: **userid_id** references table

No:	Field	Datatype (Size)	Key Constraints	Description of the field
1	userid	int	Primary key	User ID
2	username	Varchar(30)	NOT NULL	Username id
3	email	Varchar(20)	Unique	Email id
4	Phone_number	number	Not Null	Phone_number

1.profile

.Primary key: **id**

.Foreign key: **id** references **table user_id**

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	int	Primary key	id
2	age	number	NOT NULL	Age
3	gender	Varchar(20)	NOT NULL	gender
4	city	Varchar(20)	NOT NULL	city
5	Date_of_birth	date	NOT NULL	Date_of_birth
6	Profile_picture	Varchar(30)	NOT NULL	Profile_picture
7	User_id	int	Foreign Key	User_id

Bus_category

Primary key: **id** category_id

.

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	Category_id	int	Primary key	id
2	Category_name	Varchar(20)	NOT NULL	Category_name
3	description	Varchar(20)	NOT NULL	discription
4	status	Varchar(20)	NOT NULL	status
5	Date_created	date	NOT NULL	Date_created
6	Date_updated	date	NOT NULL	Date_updated

Primary key: **id**

.Foreign key: **id** category_id references **table category_id**

No:	Field	Datatype (Size)	Key Constraints	Description of the field
1	id	int	Primary key	id
2	Category_id	int	Foreign Key	Category_id
3	location	Varchar(30)	NOT NULL	location
4	status	Varchar(20)	Not Null	status
5	Date_created	date	Not Null	Date_created
6	Date_updated	date	Not Null	Date_updated

bus

Primary key: **id** bus_id

.Foreign key: **id** category_id references **table** category_id

No:	Field	Datatype (Size)	Key Constraints	Description of the field
1	Bus_id	int	Primary key	id
2	Bus_number	Varchar(30)	NOT NULL	Bus_number
3	Seat	Varchar(20)	int	Seat
4	Date_created	date	Not Null	Date_created
5	Date_updated	date	Not Null	Date_updated
6	Category_id	int	Foreign Key	Category_id

Schedule

Primary key: **id** schedule_id

.Foreign key: **id** schedule_id references **table** bus_id,

Location_id

No:	Field	Datatype (Size)	Key Constraints	Description of the field
1	Schedule_id	int	Primary key	Id
2	Code	unique	NOT NULL	Code
3	schedule	Date time	Unique	Schedule
4	fare	float	Not Null	Fare
5	status	Varchar(20)	Not Null	Status
6	Date_created	Date	Not Null	Date_created
7	Date_updated	Date	Not Null	Date_updated
8	Bus_id	Int	Foreign key	Bus_id
9	Location_id	Int	Foreign key	Location_id

Message

Primary key: **id** id

Foreign key: **id** id references **table** user_id

No:	Field	Datatype (Size)	Key Constraints	Description of the field
1	Id	int	Primary key	User ID
2	Message	Varchar(30)	NOT NULL	Username id
3	Timestamp	date	Not Null	Email id
4	User_id	int	Foreign key	User_id

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software testing serves as a critical means of ensuring that a computer program operates in accordance with its intended functionality. It is employed to verify that the software performs as expected, meeting the specified requirements and standards. Validation, in this context, involves the thorough examination and testing of software to confirm that it aligns with the user's desired outcomes. Software testing is a comprehensive process that, alongside methods such as inspection and program walkthroughs, aims to identify potential errors and discrepancies within the software. The objectives of testing can be guided by several principles, including the execution of a program with the aim of uncovering errors, the creation of effective test cases, and the successful identification of previously undiscovered errors. When conducted successfully, testing can pinpoint flaws in the software, ensuring that the computer program operates efficiently and as intended. Additionally, three fundamental methods are employed for assessing a computer program: correctness, implementation efficiency, and computational complexity.

5.2 TEST PLAN

A test plan serves as a comprehensive set of guidelines for conducting various types of tests, functioning as a navigational tool that outlines specific steps to follow. Software developers create instructions for both program usage and the organization of essential information to ensure the program's seamless operation. They meticulously verify each component of the program to confirm its intended functionality. The responsibility of thoroughly testing the built software lies with the ITG (Information Technology Group), ensuring comprehensive and rigorous testing beyond the initial development phase. The objectives of testing should be well-defined and quantifiable, with the test plan encompassing details about the frequency of malfunctions, associated costs for rectification, occurrence rates, and the time required for comprehensive reevaluation.

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Software components or modules, which are the smallest units of a software design, are tested through unit testing. Using the design guidance, test key control pathways in a module to identify issues. This implies the level of difficulty of the tests for each minor component of a program and the components that haven't been put to the test. One kind of testing called unit testing examines the internal workings of the code and can include completed concurrently for several program components.

We must first determine whether data is correctly flowing between the various components of the computer software before beginning any more tests. All other checks are useless if the data isn't entering and exiting the system appropriately. When creating anything, it's critical to consider potential difficulties and develop a plan to address them. This could entail rerouting the procedure or ending it altogether.

To test the Sell-Soft System, individual components were examined and subjected to various testing. A few errors in the modules' design were found and corrected. Following the drafting of the instructions for various components, each component is examined and tested independently. We removed unnecessary code and ensured that everything functions as it should.

5.2.2 Integration Testing

Integration testing is a technique for developing a program while simultaneously searching for errors in the interoperability of its many components. Using tested components to develop a software according to plan is the goal. To ensure proper operation, the entire software is tested. Errors are corrected, but then mistakes occur again, and so on. Following the inspection of every component of the system, the components were assembled to ensure optimal performance. Furthermore, they created uniform programs rather than varying them.

5.2.3 Validation Testing or System Testing

This concludes the testing phase. During this test, the entire system was examined to ensure that all of the various building blocks and instructions interacted as intended. This type of testing is known as system testing or black box testing. One technique to make sure the software performs as intended is to use black box testing. Through the use of several input formats, black box testing assists software engineers in identifying every issue present in a program. Black box testing examines code for errors in startup and termination, performance, data access, functions, and interfaces.

5.2.4 Output Testing or User Acceptance Testing

The system is being evaluated to see if users like it and if it satisfies the business' requirements. While it is being developed or updated, the computer program must remain connected to the user.

The following factors are taken into consideration:

- Input screen designs
- Output screen designs

The aforementioned is tested using various types of data. For system testing, having the test data ready is crucial. After gathering test data, they use that information to check the system they are researching. We look for errors in the system and use the procedures we are already familiar with to find and correct them. In order to use these fixes in the future, we keep a record of them.

5.2.5 Automation Testing

Before a piece of software is officially used, automated testing determines whether it functions properly and complies with standards. This testing technique makes use of tools that execute written instructions. When a computer system employs a specialized tool to test things automatically, this is known as UI automation testing. We write scripts that perform this task automatically for each different test instead of relying on users to click around the application to ensure everything is in working order. When you verify information and then add it, there are a few things you should do. When you need to run the same test simultaneously on numerous computers, automatic testing is required.

5.2.6 Selenium Testing

Selenium is a free and practical tool that automatically tests websites. It's critical that web developers are aware of it. Selenium automation testing is the practice of testing things using the Selenium tool. Selenium is a collection of tools that each perform unique tasks for automation testing, not just one single tool. Although manual testing is a crucial step in the development of an application, it has flaws. Its potential to be monotonous and repetitive is a major issue. Jason Huggins, a ThoughtWorks employee, developed a method to automatically test things rather than doing it by hand to make things simpler. He created a program called the JavaScriptTestRunner to aid in automatically testing websites. The program's name was changed to Selenium in 2004.

Example:

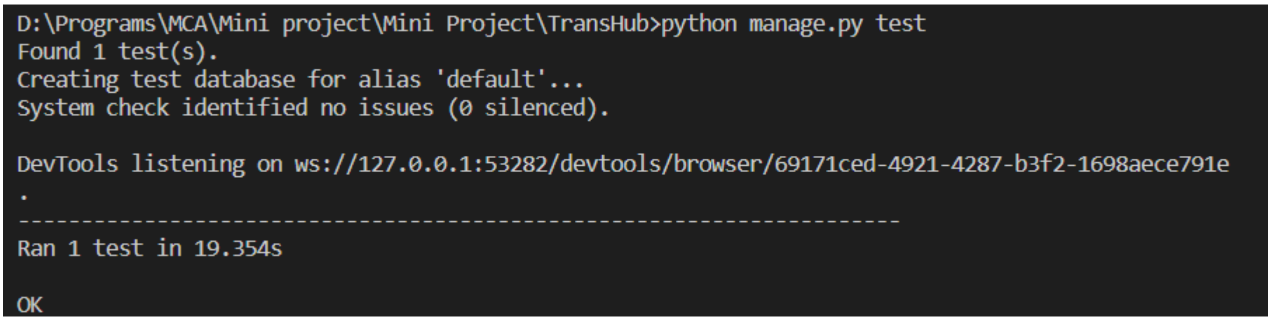
Test Case 1

Code

```
from datetime import datetime
from os import name
from django.forms import Select
from django.test import TestCase
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

class Hosttest(TestCase):

    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.implicitly_wait(10)
        self.live_server_url = 'http://127.0.0.1:8000/'

    def tearDown(self):
        self.driver.quit()
```

```
login = driver.find_element(By.CSS_SELECTOR, "a#login")
        login.click()
        time.sleep(2)
        username = driver.find_element(By.CSS_SELECTOR, "input#username")
        username.send_keys("stephin@001")
        password = driver.find_element(By.CSS_SELECTOR, "input#password")
        password.send_keys("Stephin@001")
        time.sleep(1)
        submitc = driver.find_element(By.CSS_SELECTOR, "button#login-btn")
        submitc.click()
        time.sleep(2)
```

Screenshot



```
D:\Programs\MCA\Mini project\Mini Project\TransHub>python manage.py test
Found 1 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:53282/devtools/browser/69171ced-4921-4287-b3f2-1698aece791e
.
----------------------------------------------------------------------
Ran 1 test in 19.354s

OK
```

Test Case 1

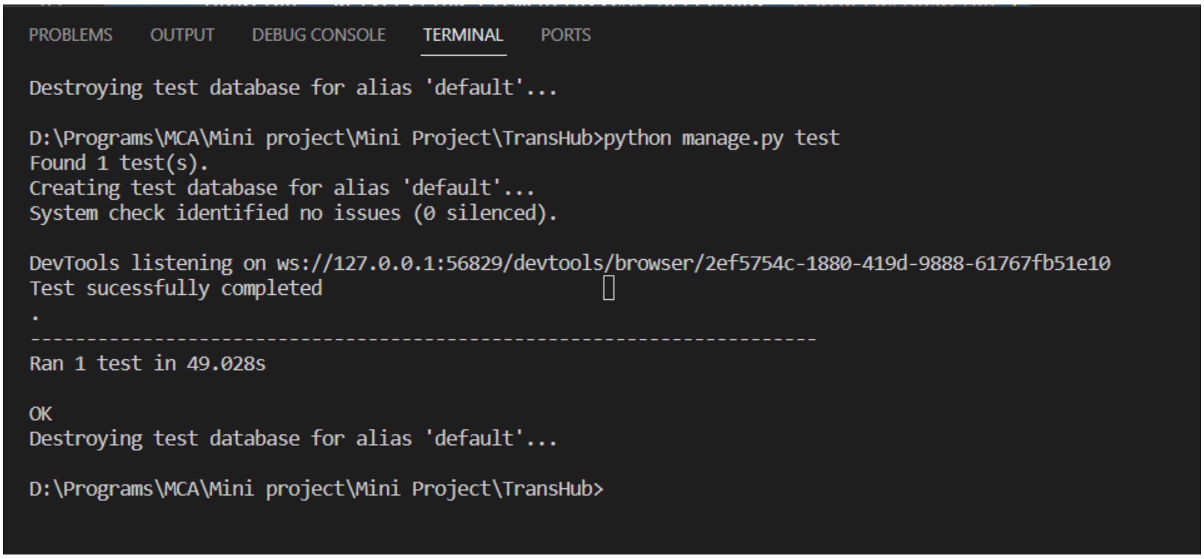
Code

```
from datetime import datetime
from os import name
from django.forms import Select
from django.test import TestCase
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

class Hosttest(TestCase):

    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.implicitly_wait(10)
        self.live_server_url = 'http://127.0.0.1:8000/'

    def tearDown(self):
        self.driver.quit()



    def test_01_login_page(self):
        driver = self.driver
        driver.get(self.live_server_url)
        driver.maximize_window()
        time.sleep(1)



        login = driver.find_element(By.CSS_SELECTOR, "a#login")
        login.click()
        time.sleep(2)
        username = driver.find_element(By.CSS_SELECTOR, "input#username")
        username.send_keys("Hashim")
        password = driver.find_element(By.CSS_SELECTOR, "input#password")
        password.send_keys("H@$h1m")
        time.sleep(1)
        submitc = driver.find_element(By.CSS_SELECTOR, "button#login-btn")
        submitc.click()
        time.sleep(2)
```

```
#trip catagery

trip_category = driver.find_element(By.CSS_SELECTOR, "a#trip_category")
trip_category.click()
time.sleep(2)

add_category = driver.find_element(By.CSS_SELECTOR, "button#add_new")
add_category.click()
time.sleep(2)

category_name = driver.find_element(By.CSS_SELECTOR, "input#name")
category_name.send_keys("AC")
description = driver.find_element(By.CSS_SELECTOR, "textarea#description")
description.send_keys("Non-stop bus")

status = driver.find_element(By.CSS_SELECTOR, "select#status")
status_option = status.find_elements(By.TAG_NAME, 'option')[0]  # Assuming the first option is a placeholder
status_option.click()

submit_category = driver.find_element(By.CSS_SELECTOR, "button#submit")
submit_category.click()
time.sleep(2)

driver.back()



#bus location

bus_location = driver.find_element(By.CSS_SELECTOR, "a#bus_location")
bus_location.click()
time.sleep(2)
```

```
    add_location = driver.find_element(By.CSS_SELECTOR, "button#add_new")
    add_location.click()
    time.sleep(2)

    location = driver.find_element(By.CSS_SELECTOR, "textarea#location")
    location.send_keys("Kollam")

    status = driver.find_element(By.CSS_SELECTOR, "select#status")
    status_option = status.find_elements(By.TAG_NAME, 'option')[0] # Assuming
the first option is a placeholder
    status_option.click()

    submit_trip_shedule = driver.find_element(By.CSS_SELECTOR,
"button#submit")
    submit_trip_shedule.click()
    time.sleep(2)

    driver.back()

    logout = driver.find_element(By.CSS_SELECTOR, "a#logout")
    logout.click()
    time.sleep(2)

    #login as user

    login = driver.find_element(By.CSS_SELECTOR, "a#login")
    login.click()
    time.sleep(2)
    username = driver.find_element(By.CSS_SELECTOR, "input#username")
    username.send_keys("Nintu")
    password = driver.find_element(By.CSS_SELECTOR, "input#password")
    password.send_keys("Nintu@123")
    time.sleep(1)
    submitc = driver.find_element(By.CSS_SELECTOR, "button#login-btn")
    submitc.click()
    time.sleep(2)

    booknow = driver.find_element(By.CSS_SELECTOR, "a#booknow")
    booknow.click()
    time.sleep(2)

    feedback = driver.find_element(By.CSS_SELECTOR, "a#feedback")
    feedback.click()
    time.sleep(2)

    feedback_form = driver.find_element(By.CSS_SELECTOR,
"textarea#feedback_form")
    feedback_form.send_keys("Misbehaviour of driver")

    feedback_submit = driver.find_element(By.CSS_SELECTOR,
"button#feedback_submit")
    feedback_submit.click()
    time.sleep(2)

    print("Test sucessfully completed")



if name == 'main':
  import unittest
  unittest.main()
```

Screenshot



```
Destroying test database for alias 'default'...

D:\Programs\MCA\Mini project\Mini Project\TransHub>python manage.py test
Found 1 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:56829/devtools/browser/2ef5754c-1880-419d-9888-61767fb51e10
Test sucessfully completed                          ▯
.
----------------------------------------------------------------------
Ran 1 test in 49.028s

OK
Destroying test database for alias 'default'...

D:\Programs\MCA\Mini project\Mini Project\TransHub>
```

Test Report

Test Case 1

Project Name:Travelix					
Login Test Case					
Test Case ID: Test_1			**Test Designed By: Muhammed Hashim M**		
Test Priority(Low/Medium/High):			**Test Designed Date:**		
Module Name: Login screen			**Test Executed By : Dr. Paulin Paul**		
Test Title : Login			**Test Execution Date:**		
Description: Verify login with valid username and password					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Home should be displayed	Login page displayed	Pass
2	Provide Valid username	Username:stephin@001	User should be able to Login	User Logged in and navigated to Player Home	Pass

3	Provide Valid Password	Password: Stephin@001			
Post-Condition: Post-Condition: The user's credentials are validated against the database, granting successful login access to the user's account. The account session details are accurately recorded and logged in the database for future reference and tracking purposes.					

Test Case 2:

Test Case 1

Project Name:Travelix					
Login Test Case					
Test Case ID: Test_2			**Test Designed By: Muhammed Hashim M**		
Test Priority(Low/Medium/High):			**Test Designed Date: 04/12/2023**		
Module Name: Login screen			**Test Executed By : Dr. Paulin Paul**		
Test Title : bus_category, location, message			**Test Execution Date: 04/12/2023**		
Description: category, location, messaging			Texting with category, location and messaging all functionality		
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	First of Home page		Home should be displayed	Admin page page displayed	Pass
2	Provide Category_bus		Admin should able to add data	User Logged in And admin category fuctionality	Pass
3	Provide Valid Password	Password: Stephin@001			
Post-Condition: Post-Condition: The user's credentials are validated against the database, granting successful login access to the user's account. The account session details are accurately recorded and logged in the database for future reference and tracking purposes.					

CHAPTER 6

IMPLEMENTATION

6.1INTRODUCTION

Implementation is the crucial phase during which the planned system materializes into a tangible, operational entity. Building user trust and confidence in the newly developed system is vital for its successful integration. The focus lies on user training and the creation of supportive resources. Conversion typically occurs during or after user training. In essence, implementation signifies the transition from a conceptual system design to its practical, functional existence. It involves the process of replacing or modifying existing systems to ensure a seamless shift towards the new system. Thorough planning, comprehensive system analysis, and the formulation of effective transition strategies are integral to the implementation process.

6.2 IMPLEMENTATION PROCEDURES

During the software implementation process, the software is installed in its intended environment and functions as expected. In certain organizations, an individual not directly involved in software usage oversees and greenlights the project's development. Initial hesitations may arise, and it is crucial to address any uncertainties before they escalate into formidable resistance. It is essential to demonstrate to users why the new system is superior to its predecessor and to instill a sense of trust in the software. Users should receive comprehensive training to ensure their comfort and confidence with the application. To evaluate the outcome, one must ensure that the server program is operational on the server before assessing the system's functionality.

6.2.1 User Training

User training is essential for imparting the necessary skills to individuals in utilizing and adapting to the system. It is imperative that users feel at ease and confident in their ability to navigate and operate the computer system effectively. Particularly when faced with complex functionalities, the significance of comprehensive user training becomes even more pronounced. This training equips users with the ability to input information, manage errors, query the database, and utilize various tools for generating reports and accomplishing other critical tasks.

6.2.2 Training on the Application Software

Training on the application software involves an in-depth understanding of how to operate a new program after familiarizing oneself with the fundamentals of computer usage. This training aims to provide comprehensive guidance on navigating the new system, encompassing the utilization of various screens, accessing support resources, troubleshooting mistakes, and rectifying errors. Its primary objective is to equip users with the requisite knowledge and skills to effectively engage with the system, tailored to the specific needs and roles of different user groups.

6.2.3 System Maintenance

System maintenance is a complex challenge within the sphere of system development. It encompasses vital tasks and ensures the effective functioning of software during the maintenance phase of the software life cycle. Once a system is operational, it requires ongoing attention to ensure its continued smooth operation. Prioritizing software maintenance throughout the development process is essential to enable the system to adapt to environmental changes. The practice of software maintenance extends beyond mere error detection within the code.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

Travelix emerges as a robust and user-centric e-transport management system designed to revolutionize transportation and logistics operations. Through its intuitive platform, the system seamlessly integrates essential functionalities across three main modules: admin, staff, and user. The meticulous system study revealed the drawbacks of the existing transportation systems, prompting the need for a sophisticated solution like Travelix. The proposed system introduces innovative features such as real-time tracking, route optimization, and a comprehensive admin module. The advantages of Travelix lie in its ability to enhance user experiences, automate logistics processes, and optimize transportation management. The feasibility study underscores Travelix potential to meet the specific needs of various stakeholders, while the behavioral feasibility analysis ensures a user-friendly and efficient platform. With a comprehensive understanding of the system's goals, features, and security measures, Travelix positions itself as a cutting-edge solution poised to elevate the efficiency and effectiveness of transportation and logistics management.

.

7.2 FUTURE SCOPE

The future scope of Travelix extends beyond its current functionalities, presenting an exciting trajectory for continuous improvement and expansion. As technology evolves, Travelix can leverage advancements such as machine learning and artificial intelligence to further enhance route optimization and predictive maintenance capabilities. Integration with emerging technologies like could fortify the security and transparency of the system, ensuring the integrity of user data and transactions. Additionally, expanding the system to accommodate multimodal transportation options, including partnerships with emerging electric and autonomous vehicle technologies, would position Travelix at the forefront of the evolving transportation landscape. Collaboration with smart city initiatives and the integration of IoT devices could lead to a more interconnected and data-driven transportation ecosystem. Furthermore, continuous user feedback and evolving industry requirements can drive the introduction of new features and improvements, ensuring that Travelix remains a dynamic and cutting-edge solution for the ever-evolving demands of the transportation and logistics sector.

.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Travelix Documentation. (Year). Title of the Documentation. Publisher or URL.

- Author, A. A., & Author, B. B. (Year). Title of the Research Paper or Technical Report related to Travelix. Journal Name, Volume(Issue), Page Range. DOI or URL.
- Company Name. (Year). Title of White Paper or Technical Report related to Travelix. URL.
- Relevant industry standards or specifications related to Travelix.
- Lastname, C. (Year). Title of the Article on [Specific Feature or Aspect] in Travelix. Magazine Name, Volume(Issue), Page Range.
- Researcher, R. R., & Scientist, S. S. (Year). Title of the Research Paper on [Specific Technology or Module] in Travelix. Conference Name, Proceedings, Page Range. DOI or URL.

WEBSITES:

- Abhi Bus
- Red Bus
- Kerala RTC

CHAPTER 9

APPENDIX

9.1 Sample Code

Following

Signup

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration</title>
  <!-- Include Bootstrap CSS -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <!-- Custom CSS for Background Image -->
  <style>
    body {
      background-image: url({% static "MainApp/images/bus1.jpg" %});
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
    }
    .container {
      padding: 20px;
      border-radius: 10px;
    }
  </style>
</head>
<body>
  <div class="container mt-5">
    <div class="row justify-content-center">
      <div class="col-md-6">
        <div class="card">
          <div class="card-header">
            <h4>Register</h4>
```

```
                    </div>
                    <div class="card-body">
                        <!-- Registration Form -->
                        <form action="" method="POST" onsubmit="return validateForm()">
                            {% csrf_token %}
                            <!-- Username -->
                            <div class="form-group">
                                <label for="username">Username</label>
                                <input type="text" class="form-control" id="username" name="username"
required onkeyup="validateUsername()" >
                                <span id="username-error" class="text-danger"></span>
                            </div>
                            <!-- Email -->
                            <div class="form-group">
                                <label for="email">Email</label>
                                <input type="email" class="form-control" id="email" name="email" required
onkeyup="validateEmail()" onblur="sendOtp()">
                                <span id="email-error" class="text-danger"></span>
                            </div>
                            <!-- OTP -->
                            <div class="form-group" id="otp-container">
                                <label for="otp">OTP</label>
                                <input type="text" class="form-control" id="otp" name="otp" required
onblur="validateOTP()">
                                <span id="otp-error" class="text-danger"></span>
                            </div>
                            <!-- Phone Number -->
                            <div class="form-group">
                                <label for="phone">Phone Number</label>
                                <input type="text" class="form-control" id="phone" name="phone" required
onkeyup="validatePhone()">
                                <small class="form-text text-muted">Format: +919123456789</small>
                                <span id="phone-error" class="text-danger"></span>
                            </div>
                            <!-- Password -->
                            <div class="form-group">
                                <label for="password">Password</label>
                                <input type="password" class="form-control" id="password"
name="password" required onkeyup="validatePassword()">
                                <span id="password-error" class="text-danger"></span>
                                <input type="checkbox" onclick="togglePasswordVisibility()"> Show
Password
                            </div>
                            <!-- Confirm Password -->
                            <div class="form-group">
                                <label for="confirm_password">Confirm Password</label>
                                <input type="password" class="form-control" id="confirm_password"
name="confirm_password" required onkeyup="validateConfirmPassword()">
                                <span id="confirm-password-error" class="text-danger"></span>
                            </div>
                            <div class="card-footer">
                                Already have an account? <a href="{% url 'login' %}">Sign in</a>
                            </div>
                            <!-- Submit Button -->
                            <button type="submit" class="btn btn-primary btn-block"
id="sub_btn">Register</button>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <!-- Include Bootstrap JS and jQuery -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
```

```
<!-- JavaScript Validation -->
<script>
    var generatedOTP = 0;
    var allCorrect = 1;
    $("#otp-container").hide();
    function validateUsername() {
        var usernameInput = document.getElementById("username");
        var usernameError = document.getElementById("username-error");
        var usernamePattern = /^[a-zA-Z][a-zA-Z0-9]*$/; // First character should be a letter, no spaces

        if (!usernamePattern.test(usernameInput.value)) {
            usernameError.textContent = "Username should start with a letter and contain only letters and numbers.";
            allCorrect = 0;
            return false;
        } else if (usernameInput.value.length > 15) {
            usernameError.textContent = "Username should be no longer than 15 characters.";
            allCorrect = 0;
            return false;
        } else {
            // send ajax request to check if the username is available
            fetch(`/check_username/?username=${usernameInput.value}`)
                .then(response => response.json())
                .then(data => {
                    if (data.exists) {
                        usernameError.textContent = 'Username is already taken.';
                        usernameError.style.color = 'red';
                    } else {
                        usernameError.textContent = '';

                    }
                })
                .catch(error => {
                    console.error('error:', error);
                    // handle the error here, e.g., show an error message
                });
        }
    }

    function validateOTP(){
        const otp = document.getElementById("otp").value;
        const otpError = document.getElementById("otp-error");
        const sub=document.getElementById("sub_btn");
        if (otp != generatedOTP){
            otpError.textContent = "OTP is Not Correct";
            allCorrect = 0;
            sub.disabled = true;
        }else{
            otpError.textContent = '';
            allCorrect = 1;
            sub.disabled = false;

        }
    }

    function sendOtp(){
        var emailInput = document.getElementById("email");
        var emailError = document.getElementById("email-error");
        fetch(`/validateGlobalEmail/?email=${emailInput.value}`)
            .then(response => response.json())
            .then(data => {
                if (data.otp){
                    emailError.classList = "text-success";
                    emailError.textContent = "OTP is Send to Email, Please Enter it;";
                    $("#otp-container").show();
                    generatedOTP = data.otp;
                console.log(generatedOTP);
            }
        })
        .catch(error => {
            console.error('An error occurred:', error);
        });
    }

    function validateEmail() {
        var emailInput = document.getElementById("email");
        var emailError = document.getElementById("email-error");
        var emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/; // Email format

        if (!emailPattern.test(emailInput.value)) {
            emailError.textContent = "Invalid email format.";
            allCorrect = 0;
            return false;
        } else {
          //send an ajax request to check if the email is available
          fetch(`/check_email/?email=${emailInput.value}`)
              .then(response => response.json())
              .then(data => {
                  console.log(data);
                  if (data.exists) {
                      emailError.textContent = 'Email is already registered.';
                      emailError.style.color = 'red';
                  } else {
                      emailError.textContent = '';
                  }
              })
              .catch(error => {
                  console.error('Error:', error);
                  // Handle the error here, e.g., show an error message to the user
              });
        }
    }

    function validatePhone() {
        var phoneInput = document.getElementById("phone");
        var phoneError = document.getElementById("phone-error");
        var phonePattern = /^\+91\d{10}$/; // Phone number format: +91 followed by 10 digits

        if (!phonePattern.test(phoneInput.value)) {
          phoneError.textContent = "Phone number should start with +91 and contain 10 digits.";
          allCorrect = 0;
          return false;
        } else {
          phoneError.textContent = "";
          allCorrect = 1;
          return true;
        }
    }

    function validatePassword() {
        var passwordInput = document.getElementById("password");
        var passwordError = document.getElementById("password-error");
        var passwordPattern = /^(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[@#$%^&!])[A-Za-
    z\d@#$%^&!]{6,}$/; // Password requirements

        if (!passwordPattern.test(passwordInput.value)) {
          passwordError.textContent = "Password should contain at least 6 characters, including
    one uppercase letter, one lowercase letter, one digit, and one special character.";
          allCorrect = 0;
          return false;
        } else {
          passwordError.textContent = "";
          allCorrect = 1;
          return true;
```

```
        }
    }

    function validateConfirmPassword() {
        var confirmPasswordInput = document.getElementById("confirm_password");
        var confirmPasswordError = document.getElementById("confirm-password-error");
        var passwordInput = document.getElementById("password");

        if (confirmPasswordInput.value !== passwordInput.value) {
            confirmPasswordError.textContent = "Passwords do not match.";
            allCorrect = 0;
            return false;
        } else {
            confirmPasswordError.textContent = "";
            var cr = validateUsername() + validateEmail() + validateOTP() + validatePassword() +
validatePhone();
            console.log(cr);
            allCorrect = 1;
            return true;
        }
    }

    function togglePasswordVisibility() {
        var passwordInput = document.getElementById("password");
        if (passwordInput.type === "password") {
            passwordInput.type = "text";
        } else {
            passwordInput.type = "password";
        }
    }

    function validateForm() {
        console.log(cr);
    }
    </script>
</body>
</html>
```

Login

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <!-- Include Bootstrap CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <!-- Custom CSS for Page Styling -->
    <style>
        .body {
            background-image: url({% static 'MainApp/images/login.jpg' %});
            background-size: cover;
            background-repeat: no-repeat;
            background-attachment: fixed;
        }
        .login-container {
            max-width: 400px;
            margin: 0 auto;
            padding: 20px;
            background-color: rgba(255, 255, 255, 0.8);
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0,0,0,0.1);
            margin-top: 100px;
        }
    </style>
</head>
```

```
<body>
  <div class="container login-container">
    <h2 class="text-center">Login</h2>
    <form  method="post">
      {% csrf_token %}
      <!-- Email Field -->
      <div class="form-group">
        <label>Username</label>
        <input type="text" class="form-control" id="username" name="username"
placeholder="Enter your username" required >

      </div>
      <!-- Password Field -->
      <div class="form-group">
        <label for="password">Password</label>
        <input type="password" class="form-control" id="password" name="password"
placeholder="Enter your password" required onkeyup="validatePassword()">
        <span id="password-error" class="text-danger"></span>
      </div>
      <!-- New Password Field (For Forgot Password) -->
      <!-- Forgot Password Link -->
      <p class="text-right"><a href="{% url 'password_reset'%}"
id="forgotPasswordLink">Forgot Password?</a></p>
      <!-- Submit Button -->
      <p>{{Error_message}}</p>
      <button type="submit" class="btn btn-primary btn-block" id="login-btn">Login</button>
    </form>
  </div>

</body>
</html>
```

Admin

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">



        <!-- Boxicons -->
        <link href='https://unpkg.com/boxicons@2.0.9/css/boxicons.min.css' rel='stylesheet'>



        <!-- My CSS -->
        <link rel="stylesheet" href="style.css">

        <title>AdminHub</title>
   <link href="{% static 'MainApp/styles/adminstyle.css' %}" rel="stylesheet">

   {% block style %}

   {% endblock style %}

</head>
<body>

        <!-- SIDEBAR -->
        <section id="sidebar">
                <a href="#" class="brand">
                        <i class='bx bxs-smile'></i>
                        <span class="text">AdminHub</span>
                </a>
                <ul class="side-menu top">
```

```
                    <li class="active">
                              <a href="#">
                                        <i class='bx bxs-dashboard' ></i>
                                        <span class="text">Dashboard</span>
                              </a>
                    </li>
                    <li>
                              <a href="{% url "user_account" %}">
                                        <i class='bx bxs-group' ></i>
                                        <span class="text">show users</span>
                              </a>
                    </li>
          <li>
                              <a href="{% url "category-page" %}">
                                        <i class='bx bx-category'></i>
                                        <span class="text">Trip Category</span>
                              </a>
                    </li>

                    <li>
                              <a href="{% url "location-page" %}">
                                        <i class='bx bx-map'></i>
                                        <span class="text">Bus Location</span>
                              </a>
                    </li>
                    <li>
                              <a href="{% url "bus-page" %}">
                                        <i class='bx bx-bus'></i>
                                        <span class="text">Bus List</span>
                              </a>
                    </li>
                    <li>
                              <a href="{% url "schedule-page" %}">
                                        <i class='bx bx-calendar-event'></i>
                    <span class="text">Trip Schedule</span>
                </a>
            </li>
            <li>
                <a href="{% url "adminfeedback" %}">
                    <i class='bx bx-message-rounded-detail'></i>
                    <span class="text">user feedback</span>
                </a>
            </li>
        </ul>
        <ul class="side-menu">
            <li>
                <a href="#">
                    <i class='bx bxs-cog' ></i>
                    <span class="text">Settings</span>
                </a>
            </li>
            <li>
                <a href="{% url "logout" %}" class="logout">
                    <i class='bx bxs-log-out-circle' ></i>
                    <span class="text">Logout</span>
                </a>
            </li>
        </ul>
    </section>
    <!-- SIDEBAR -->
    <!-- CONTENT -->
    <section id="content">
        <!-- NAVBAR -->
        <nav>
            <i class='bx bx-menu' ></i>
            <a href="#" class="nav-link">Categories</a>
            <form action="#">
                <div class="form-input">
```

```
                    <input type="search" placeholder="Search...">
                    <button type="submit" class="search-btn"><i class='bx bx-
search' ></i></button>
                </div>
            </form>
            <input type="checkbox" id="switch-mode" hidden>
            <label for="switch-mode" class="switch-mode"></label>
            <a href="#" class="notification">
                <i class='bx bxs-bell' ></i>
                <span class="num">8</span>
            </a>
            <a href="#" class="profile">
                <img src="img/people.png">
            </a>
        </nav>
        <!-- NAVBAR -->

        <!-- MAIN -->
        <main>
            <div class="head-title">
                <div class="left">
                    <h1>Dashboard</h1>
                    <ul class="breadcrumb">
                        <li>
                            <a href="#">Dashboard</a>
                        </li>
                        <li><i class='bx bx-chevron-right' ></i></li>
                        <li>
                            <a class="active" href="#">Home</a>
                        </li>
                    </ul>
                </div>
                {% comment %} <a href="#" class="btn-download">
                    <i class='bx bxs-cloud-download' ></i>
```

```
                                        <span class="text">Download PDF</span>
                                </a> {% endcomment %}
                        </div>

        {% block body %}

        {% endblock body %}

                </main>
                <!-- MAIN -->
        </section>
        <!-- CONTENT -->

 <script src="{% static 'MainApp/js/homescript.js' %}"></script>
   {% block scripts %}

   {% endblock scripts %}
</body>
</html>
```

Base

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

<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0"> {% if page_title %}
   <title>{{ page_title }} | Inventory Management System</title>
   {% else %}
   <title>Inventory Management System</title>
   {% endif %}
```

```
    <link rel="icon" href="{{ MEDIA_URL }}/default/logo.png">
    <link rel="stylesheet" href="{% static 'MainApp/font-awesome/css/all.min.css' %}">
    <link rel="stylesheet" href="{% static 'MainApp/bootstrap/css/bootstrap.min.css' %}">
    <link rel="stylesheet" href="{% static 'MainApp/select2/dist/css/select2.min.css' %}">
    <link rel="stylesheet" href="{% static 'MainApp/mdb-blogtemplate/css/mdb.min.css' %}" />
    <link rel="stylesheet" href="{% static 'MainApp/DataTables/datatables.min.css' %}" />
    <link rel="stylesheet" href="{% static 'MainApp/default/css/style.css' %}">

    <script src="{% static 'MainApp/font-awesome/js/all.min.js' %}"></script>
    <script src="{% static 'MainApp/default/js/jquery-3.6.0.min.js' %}"></script>
    <script src="{% static 'MainApp/bootstrap/js/bootstrap.min.js' %}"></script>
    <script src="{% static 'MainApp/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
    <script src="{% static 'MainApp/DataTables/datatables.min.js' %}"></script>
    <script src="{% static 'MainApp/bootstrap/js/popper.min.js' %}"></script>
    <script type="text/javascript" src="{% static 'MainApp/mdb-blogtemplate/js/mdb.min.js'
%}"></script>
    <script type="text/javascript" src="{% static 'MainApp/default/js/script.js' %}"></script>

    {% block headerContent %} {% endblock headerContent %}
</head>

<body class="">

    <main class="py-5">
        <div class="container mb-3">
            {% if messages %}
            <div class="row">
                <div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
                    {% for message in messages %}
                    <div class="alert alert-{% if message.extra_tags %}{{
message.tags|replaceBlank:message.extra_tags|replaceBlank:' ' }}{% else %}{{ message.tags
}}{% endif %} w-100 rounded-0 mb-2 redirect-msg">
                        <div class="d-flex w-100">
                            <div class="col-auto flex-shrink-1 flex-grow-1">{{ message|safe }}</div>
                            <div class="col-auto text-center">
                                <button class="btn-close btn-sm text-sm" type="button"
onclick="$(this).closest('.alert').remove()"></button>
                            </div>
                        </div>

                    </div>
                    {% if message.extra_tags != 'stay' %}
                    <script>
                        $(function() {
                            if ($('.redirect-msg').length > 0) {
                                setTimeout(() => {
                                    $('.redirect-msg').hide('slideUp')
                                    setTimeout(() => {
                                        $('.redirect-msg').remove()
                                    }, 500)
                                }, 3500)
                            }

                        })
                    </script>
                    {% endif %}
                    {% endfor %}
                </div>
            </div>{% endif %} {% block pageContent %} {% endblock pageContent %}

        </div>
    </main>
    {% block ScriptBlock %} {% endblock ScriptBlock %}
    <div class="modal fade" id="uni_modal" role='dialog'>
        <div class="modal-dialog modal-md modal-dialog-centered" role="document">
            <div class="modal-content rounded-0">
                <div class="modal-header">
                    <h5 class="modal-title"></h5>
                </div>
                <div class="modal-body">
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn  btn-sm btn-flat btn-primary rounded-0" id='submit'
onclick="$('#uni_modal form').submit()">Save</button>
                    <button type="button" class="btn btn-sm btn-flat btn-light border rounded-0" data-bs-
dismiss="modal">Cancel</button>
                </div>
            </div>
        </div>
    </div>
    <div class="modal fade" id="confirm_modal" role='dialog'>
        <div class="modal-dialog modal-md modal-dialog-centered" role="document">
            <div class="modal-content rounded-0">
                <div class="modal-header">
                    <h5 class="modal-title">Confirmation</h5>
                </div>
                <div class="modal-body">
                    <div id="delete_content"></div>
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn btn-sm btn-flat btn-primary rounded-0"
id='confirm' onclick="">Continue</button>
                    <button type="button" class="btn btn-sm btn-flat btn-light border rounded-0" data-bs-
dismiss="modal">Close</button>
                </div>
            </div>
        </div>
    </div>

    <!--Footer-->
    <footer class="bg-light text-lg-start">
        <!-- <div class="py-4 text-center">
```

```
          <a role="button" class="btn btn-primary btn-lg m-2"
href="https://www.youtube.com/channel/UC5CF7mLQZhvx8O5GODZAhdA" rel="nofollow"
target="_blank">
      Learn Bootstrap 5
    </a>
          <a role="button" class="btn btn-primary btn-lg m-2"
href="https://mdbootstrap.com/docs/standard/" target="_blank">
      Download MDB UI KIT
    </a>
      </div>

      <hr class="m-0" /> -->

      <!-- <div class="text-center py-4 align-items-center">
          <p>Follow MDB on social media</p>
          <a href="https://www.youtube.com/channel/UC5CF7mLQZhvx8O5GODZAhdA"
class="btn btn-primary m-1" role="button" rel="nofollow" target="_blank">
              <i class="fab fa-youtube"></i>
          </a>
          <a href="https://www.facebook.com/mdbootstrap" class="btn btn-primary m-1"
role="button" rel="nofollow" target="_blank">
              <i class="fab fa-facebook-f"></i>
          </a>
          <a href="https://twitter.com/MDBootstrap" class="btn btn-primary m-1" role="button"
rel="nofollow" target="_blank">
              <i class="fab fa-twitter"></i>
          </a>
          <a href="https://github.com/mdbootstrap/mdb-ui-kit" class="btn btn-primary m-1"
role="button" rel="nofollow" target="_blank">
              <i class="fab fa-github"></i>
          </a>
      </div> -->

      <!-- Copyright -->
```

```
        <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.2);">
            <a class="text-dark" href="" target="_blank">© 2023 E-Transport System</a>
        </div>
        <!-- Copyright -->
    </footer>
    <script>
        const loader = $('<div>')
        loader.attr('id', 'pre-loader')
        loader.html('<div class="lds-
default"><div></div><div></div><div></div><div></div><div></div><div></div><div></div>
<div></div><div></div><div></div><div></div><div></div></div>')

        window.start_loader = function() {
            $('body').removeClass('loading')
            if ($('#pre-loader').length > 0)
                $('#pre-loader').remove();
            $('body').append(loader)
            $('body').addClass('loading')
        }
        window.end_loader = function() {
            if ($('#pre-loader').length > 0)
                $('#pre-loader').remove();
            $('body').removeClass('loading')
        }
        window.uni_modal = function($title = '', $url = '', $size = "") {
            start_loader()
            $.ajax({
                url: $url,
                error: err => {
                    console.log()
                    alert("An error occured")
                },
                success: function(resp) {
                    if (resp) {
```

```
            $('#uni_modal .modal-title').html($title)
            $('#uni_modal .modal-body').html(resp)
            if ($size != ") {
              $('#uni_modal .modal-dialog').addClass($size + '  modal-dialog-centered')
            } else {
              $('#uni_modal .modal-dialog').removeAttr("class").addClass("modal-dialog
modal-md modal-dialog-centered")
            }
            $('#uni_modal').modal({
              backdrop: 'static',
              keyboard: false,
              focus: true
            })
            $('#uni_modal').modal('show')
            end_loader()
          }
        }
      })
    }
    window._conf = function($msg = ", $func = ", $params = []) {
      $('#confirm_modal #confirm').attr('onclick', $func + "(" + $params.join(',') + ")")
      $('#confirm_modal .modal-body').html($msg)
      $('#confirm_modal').modal('show')
    }

  </script>



  <script src="{% static 'MainApp/select2/dist/js/select2.full.js' %}"></script>
</body>

</html>
```

Category_mgt

```
{% extends 'base.html' %} {% block pageContent %}
    <div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
      <div class="card card-default rounded-0 shadow ">
        <div class="card-header">
          <div class="d-flex w-100 align-items-center justify-content-between">
            <h4 class="card-title fw-bold">Trip Categories</h4>
            <div class="tools">
              <button type="button" class="btn btn-primary rounded-0 bg-gradient btn-sm" 
id='add_new'><i class="fa fa-plus"></i> Add New</button>
            </div>
          </div>
        </div>
        <div class="card-body">
          <div class="container-fluid">
            <table class="table table-bordered" id="category-list">
              <colgroup>
                <col width="5%">
                <col width="15%">
                <col width="20%">
                <col width="30%">
                <col width="15%">
                <col width="15%">
              </colgroup>
              <thead>
                <tr class="bg-gradient bg-primary bg-opacity-50 text-light">
                  <th class="px-2 py-2 text-center">#</th>
                  <th class="px-2 py-2 text-center">Date/Time</th>
                  <th class="px-2 py-2 text-center">Name</th>
                  <th class="px-2 py-2 text-center">Description</th>
                  <th class="px-2 py-2 text-center">Status</th>
                  <th class="px-2 py-2 text-center">Action</th>
                </tr>
              </thead>
```

```
                            <tbody>
                                {% for category in categories %}
                                <tr>
                                    <td class="px-2 py-1 align-middle">{{ forloop.counter }}</td>
                                    <td class="px-2 py-1 align-middle">{{ category.date_created|date:"Y-m-d h:i A" }}</td>
                                    <td class="px-2 py-1 align-middle">{{ category.name }}</td>
                                    <td class="px-2 py-1 align-middle">
                                        <p class="m-0 text-truncate">{{ category.description }}</p>
                                    </td>
                                    <td class="px-1 py-1 align-middle text-center">
                                        {% if category.status == '1' %}
                                        <span class="badge bg-primary bg-gradient rounded-pill px-2">Active</span> {% else %}
                                        <span class="badge bg-secondary bg-gradient rounded-pill px-2">Inactive</span> {% endif %}
                                    </td>
                                    <td class="px-2 py-1 align-middle text-center">
                                        <a class="btn btn-outline-primary btn-sm edit-data" href="javascript:void(0)" data-id="{{ category.pk }}" title="Edit">
                                            <i class="fa fa-edit"></i>
                                        </a>
                                        <button class="btn btn-outline-danger btn-sm delete-data" type="button" data-id="{{ category.pk }}" title="Delete">
                                            <i class="fa fa-trash"></i>
                                        </button>
                                    </td>
                                </tr>
                                {% endfor %}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
```

```
    </div>

{% endblock pageContent %} {% block ScriptBlock %}
<script>
  $(function() {
    $('#add_new').click(function() {
      uni_modal('<i class="fa fa-plus"></i> Add Category', '{% url "manage-category" %}',
'modal-md')
    })
    $('.edit-data').click(function() {
      uni_modal('<i class="fa fa-edit"></i> Edit Category', '{% url "manage-category" %}/' +
$(this).attr('data-id'), 'modal-md')
    })
    $('.delete-data').click(function() {
      _conf("Are you sure to delete this Category permanently?", "delete_category",
[$(this).attr('data-id')])
    })

    $('#category-list').DataTable({
      columnDefs: [{
        orderable: false,
        targets: 5
      }],
      initComplete: function(settings, json) {
        $('table td, table th').addClass('px-2 py-1')
      },
      drawCallback: function(settings) {
        $('table td, table th').addClass('px-2 py-1')
      }
    })
  })

  function delete_category($id) {
    start_loader();
```

```
var _this = $(this)
$('.err-msg').remove();
var el = $('<div>')
el.addClass("alert alert-danger err-msg")
el.hide()
$.ajax({
    url: '{% url "delete-category" %}',
    headers: {
        'X-CSRFToken': "{{csrf_token}}"
    },
    method: 'POST',
    data: {
        id: $id
    },
    dataType: 'json',
    error: err => {
        console.log(err)
        el.text('An error occurred.')
        el.show('slow')
        end_loader()
    },
    success: function(resp) {
        if (resp.status == 'success') {
            location.reload()
        } else if (!!resp.msg) {
            el.text('An error occurred.')
            el.show('slow')
        } else {
            el.text('An error occurred.')
            el.show('slow')
        }
        end_loader()
    }
})
```

```
    }

    function edit_category(id) {
        $.ajax({
            url: '{% url "manage-category" %}',
            data: {
                'pk': id,
            },
            dataType: 'html',
            success: function(response) {
                $('#category-modal .modal-content').html(response);
                $('#category-modal').modal('show');
            },
            error: function(response) {
                alert('An error occurred while trying to edit the category.');
            }
        });
    }
</script>
{% endblock ScriptBlock %}
```

Manage_category

```
{% load customfilter %}
<div class="container-fluid">
    <form action=" " id="category-form">
        {% csrf_token %}
        <input type="hidden" name="id" value="{{ category.id }}">
        <div class="form-group mb-3 ">
            <label for="name" class="control-label">Category Name</label>
            <input type="text" class="form-control rounded-0" id="name" name="name" value="{{
category.name }}" required>
        </div>
        <div class="form-group mb-3">
            <label for="description" class="control-label">Description</label>
            <textarea class="form-control rounded-0" name="description" id="description" rows="5"
required>{{ category.description }}</textarea>
        </div>
        <div class="form-group mb-3 ">
            <label for="status" class="control-label">Status</label>
            <select name="status" id="status" class="form-select rounded-0">
                {% if not category.status or category.status == '1' %}
                <option value="1" selected>Active</option>
                {% else %}
                <option value="1" >Active</option>
                {% endif %}
                {% if category.status == '2' %}
                <option value="2" selected>Inactive</option>
                {% else %}
                <option value="2" >Inactive</option>
                {% endif %}
            </select>
        </div>
    </form>
</div>
<script>
    $(function() {
        $('#category-form').submit(function(e) {
            e.preventDefault();
            var _this = $(this)
            $('.err-msg').remove();
            var el = $('<div>')
            el.addClass("alert alert-danger err-msg")
            el.hide()
            if (_this[0].checkValidity() == false) {
                _this[0].reportValidity();
                return false;
            }
            start_loader();
            $.ajax({
                url: "{% url 'save-category' %}",
                data: new FormData($(this)[0]),
                cache: false,
                contentType: false,
                processData: false,
                method: 'POST',
                type: 'POST',
                dataType: 'json',
                error: err => {
                    console.log(err)
                    alert("An error occured ", 'error');
                    end_loader();
                },
                success: function(resp) {
                    if (typeof resp == 'object' && resp.status == 'success') {
                        el.removeClass("alert alert-danger err-msg ")
                        location.reload()
                    } else if (resp.status == 'failed' && !!resp.msg) {
                        el.html(resp.msg)
                    } else {
                        el.text("An error occured ", 'error');
                        end_loader();
                        console.err(resp)
                    }
                    _this.prepend(el)
                    el.show('slow')
                    $("html, body, .modal ").scrollTop(0);
                    end_loader()
                }
            })
        })
    })
</script>
```

Location

```
{% extends 'base.html' %} {% block pageContent %}
<div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
    <div class="card card-default rounded-0 shadow ">
        <div class="card-header">
            <div class="d-flex w-100 align-items-center justify-content-between">
                <h4 class="card-title fw-bold">Bus Stop Locations</h4>
                <div class="tools">
                    <button type="button" class="btn btn-primary rounded-0 bg-gradient btn-sm"
id='add_new'><i class="fa fa-plus"></i> Add New</button>
                </div>
            </div>
        </div>
        <div class="card-body">
            <div class="container-fluid">
                <table class="table table-bordered" id="location-list">
                    <colgroup>
                        <col width="5%">
                        <col width="15%">
                        <col width="40%">
                        <col width="15%">
                        <col width="15%">
                    </colgroup>
                    <thead>
                        <tr class="bg-gradient bg-primary bg-opacity-50 text-light">
                            <th class="px-2 py-2 text-center">#</th>
                            <th class="px-2 py-2 text-center">Date/Time Created</th>
                            <th class="px-2 py-2 text-center">Location</th>
                            <th class="px-2 py-2 text-center">Status</th>
                            <th class="px-2 py-2 text-center">Action</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for location in locations %}
                        <tr>
                            <td class="px-2 py-1 align-middle">{{ forloop.counter }}</td>
                            <td class="px-2 py-1 align-middle">{{ location.date_created|date:"Y-m-d h:i A"
}}</td>
                            <td class="px-2 py-1 align-middle">{{ location.location }}</td>
                            <td class="px-1 py-1 align-middle text-center">
                              {% if location.status == '1' %}
                              <span class="badge bg-primary bg-gradient rounded-pill px-
2">Active</span> {% else %}
                              <span class="badge bg-secondary bg-gradient rounded-pill px-
2">Inactive</span> {% endif %}
                            </td>
                            <td class="px-2 py-1 align-middle text-center">
                              <a class="btn btn-outline-primary btn-sm edit-data" href="javascript:void(0)"
data-id="{{ location.pk }}" title="Edit">
                  <i class="fa fa-edit"></i>
                              </a>
                              <button class="btn btn-outline-danger btn-sm delete-data" type="button" data-
id="{{ location.pk }}" title="Delete">
                  <i class="fa fa-trash"></i>
                              </button>
                            </td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
    </div>
</div>
{% endblock pageContent %} {% block ScriptBlock %}
<script>
    $(function() {
        $('#add_new').click(function() {
            uni_modal('<i class="fa fa-plus"></i> Add Location', '{% url "manage-location" %}',
'modal-md')
        })
        $('.edit-data').click(function() {
            uni_modal('<i class="fa fa-edit"></i> Edit Location', '{% url "manage-location" %}/' +
$(this).attr('data-id'), 'modal-md')
        })
        $('.delete-data').click(function() {
            _conf("Are you sure to delete this Location permanently?", "delete_location",
[$(this).attr('data-id')])
        })

        $('#location-list').DataTable({
            columnDefs: [{
                orderable: false,
                targets: 4
            }],
            initComplete: function(settings, json) {
                $('table td, table th').addClass('px-2 py-1')
            },
            drawCallback: function(settings) {
                $('table td, table th').addClass('px-2 py-1')
            }
        })
    })

    function delete_location($id) {
        start_loader();
        var _this = $(this)
        $('.err-msg').remove();
        var el = $('<div>')
        el.addClass("alert alert-danger err-msg")
        el.hide()
        $.ajax({
            url: '{% url "delete-location" %}',
            headers: {
                'X-CSRFToken': "{{csrf_token}}"
            },
            method: 'POST',
            data: {
                id: $id
            },
            dataType: 'json',
            error: err => {
                console.log(err)
                el.text('An error occurred.')
                el.show('slow')
                end_loader()
            },
            success: function(resp) {
                if (resp.status == 'success') {
                    location.reload()
                } else if (!!resp.msg) {
                    el.text('An error occurred.')
                    el.show('slow')
                } else {
                    el.text('An error occurred.')
                    el.show('slow')
                }
                end_loader()
            }
        })
    }
</script>
{% endblock ScriptBlock %}
```

Manage_location

```
{% load customfilter %}
```

```
<div class="container-fluid">
    <form action=" " id="location-form">
        {% csrf_token %}
        <input type="hidden" name="id" value="{{ location.id }}">
        <div class="form-group mb-3">
            <label for="location" class="control-label">Location</label>
            <textarea class="form-control rounded-0" name="location" id="location" rows="5"
required>{{ location.location }}</textarea>
        </div>
        <div class="form-group mb-3 ">
            <label for="status" class="control-label">Status</label>
            <select name="status" id="status" class="form-select rounded-0">
                {% if not location.status or location.status == '1' %}
                <option value="1" selected>Active</option>
                {% else %}
                <option value="1" >Active</option>
                {% endif %}
                {% if location.status == '2' %}
                <option value="2" selected>Inactive</option>
                {% else %}
                <option value="2" >Inactive</option>
                {% endif %}
            </select>
        </div>
    </form>
</div>
<script>
    $(function() {
        $('#location-form').submit(function(e) {
            e.preventDefault();
            var _this = $(this)
            $('.err-msg').remove();
            var el = $('<div>')
            el.addClass("alert alert-danger err-msg")
            el.hide()
            if (_this[0].checkValidity() == false) {
                _this[0].reportValidity();
                return false;
            }
            start_loader();
            $.ajax({
                url: "{% url 'save-location' %}",
                data: new FormData($(this)[0]),
                cache: false,
                contentType: false,
                processData: false,
                method: 'POST',
                type: 'POST',
                dataType: 'json',
                error: err => {
                    console.log(err)
                    alert("An error occured ", 'error');
                    end_loader();
                },
                success: function(resp) {
                    if (typeof resp == 'object' && resp.status == 'success') {
                        el.removeClass("alert alert-danger err-msg ")
                        location.reload()
                    } else if (resp.status == 'failed' && !!resp.msg) {
                        el.html(resp.msg)
                    } else {
                        el.text("An error occured ", 'error');
                        end_loader();
                        console.err(resp)
                    }
                    _this.prepend(el)
                    el.show('slow')
                    $("html, body, .modal ").scrollTop(0);
```

```
                end_loader()
            }
        })
    })
  })
</script>
```

Schedule_mgt

```
{% extends 'base.html' %} {% load humanize %} {% block pageContent %}
<div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
    <div class="card card-default rounded-0 shadow ">
        <div class="card-header">
            <div class="d-flex w-100 align-items-center justify-content-between">
                <h4 class="card-title fw-bold">Trip Schedules List</h4>
                <div class="tools">
                    <button type="button" class="btn btn-primary rounded-0 bg-gradient btn-sm"
id='add_new'><i class="fa fa-plus"></i> Add New</button>
                </div>
            </div>
        </div>
        <div class="card-body">
            <div class="container-fluid">
                <table class="table table-bordered" id="schedule-list">
                    <colgroup>
                        <col width="5%">
                        <col width="15%">
                        <col width="20%">
                        <col width="30%">
                        <col width="10%">
                        <col width="10%">
                        <col width="10%">
                    </colgroup>
                    <thead>
                        <tr class="bg-gradient bg-primary bg-opacity-50 text-light">
                            <th class="px-2 py-2 text-center">id</th>
                            <th class="px-2 py-2 text-center">Schedule Time</th>
                            <th class="px-2 py-2 text-center">Bus</th>
                            <th class="px-2 py-2 text-center">Route (From - To)</th>
                            <th class="px-2 py-2 text-center">Price</th>
                            <th class="px-2 py-2 text-center">Status</th>
                            <th class="px-2 py-2 text-center">Action</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for schedule in schedules %}
                        <tr>
                            <td class="px-2 py-1 align-middle">{{ forloop.counter }}</td>
                            <td class="px-2 py-1 align-middle">{{ schedule.schedule|date:"Y-m-d h:i A"
}}</td>
                            <td class="px-2 py-1 align-middle">
                                <div class="lh-1">
                                    <div>{{ schedule.bus.bus_number }}</div>
                                    <small class="text-muted">{{ schedule.bus.category }}</small>
                                </div>
                            </td>
                            <td class="px-2 py-1 align-middle">
                                <div class="lh-1">
                                    <div>{{ schedule.depart }}</div>
                                    <div>{{ schedule.destination }}</div>
                                </div>
                            </td>
                            <td class="px-2 py-1 align-middle">{{ schedule.fare|intcomma }}</td>
                            <td class="px-1 py-1 align-middle text-center">
                                {% if schedule.status == '1' %}
                                <span class="badge bg-primary bg-gradient rounded-pill px-
2">Active</span> {% else %}
                                <span class="badge bg-danger bg-gradient rounded-pill px-
2">Cancelled</span> {% endif %}
```

```
                    </td>
                    <td class="px-2 py-1 align-middle text-center">
                        <a class="btn btn-outline-primary btn-sm edit-data" href="javascript:void(0)"
data-id="{{ schedule.pk }}" title="Edit">
                            <i class="fa fa-edit"></i>
                        </a>
                        <button class="btn btn-outline-danger btn-sm delete-data" type="button" data-
id="{{ schedule.pk }}" title="Delete">
                            <i class="fa fa-trash"></i>
                        </button>
                    </td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
      </div>
    </div>
  </div>
</div>
{% endblock pageContent %} {% block ScriptBlock %}
<script>
  $(function() {
    $('#add_new').click(function() {
      uni_modal('<i class="fa fa-plus"></i> Add Schedule', '{% url "manage-schedule" %}',
'modal-md')
    })
    $('.edit-data').click(function() {
      uni_modal('<i class="fa fa-edit"></i> Edit Schedule', '{% url "manage-schedule" %}/' +
$(this).attr('data-id'), 'modal-md')
    })
    $('.delete-data').click(function() {
      _conf("Are you sure to delete this Schedule permanently?", "delete_schedule",
[$(this).attr('data-id')])
    })
```

```
        $('#schedule-list').DataTable({
            columnDefs: [{
                orderable: false,
                targets: 6
            }],
            initComplete: function(settings, json) {
                $('table td, table th').addClass('px-2 py-1')
            },
            drawCallback: function(settings) {
                $('table td, table th').addClass('px-2 py-1')
            }
        })
    })

    function delete_schedule($id) {
        start_loader();
        var _this = $(this)
        $('.err-msg').remove();
        var el = $('<div>')
        el.addClass("alert alert-danger err-msg")
        el.hide()
        $.ajax({
            url: '{% url "delete-schedule" %}',
            headers: {
                'X-CSRFToken': "{{csrf_token}}"
            },
            method: 'POST',
            data: {
                id: $id
            },
            dataType: 'json',
            error: err => {
                console.log(err)
                el.text('An error occurred.')
                el.show('slow')
                end_loader()
            },
            success: function(resp) {
                if (resp.status == 'success') {
                    location.reload()
                } else if (!!resp.msg) {
                    el.text('An error occurred.')
                    el.show('slow')
                } else {
                    el.text('An error occurred.')
                    el.show('slow')
                }
                end_loader()
            }
        })
    }
</script>
{% endblock ScriptBlock %}
```

Manage_schedule

```
{% load customfilter %}
<div class="container-fluid">
    <form action=" " id="schedule-form">
        {% csrf_token %}
        <input type="hidden" name="id" value="{{ schedule.id }}">
        <input type="hidden" name="code" value="1">
        <div class="form-group mb-3 ">
            <label for="bus" class="control-label">Bus</label>
            <select name="bus" id="bus" class="form-select select2 rounded-0">
                {% if not bus.bus_number %}
                <option selected></option>
                {% else %}
                <option ></option>
                {% endif %}
                {% for bus in buses %}
                {% if schedule.bus.id == bus.id %}
                <option value="{{ bus.id }}" selected>{{ bus }}</option>
                {% else %}
                <option value="{{ bus.id }}" >{{ bus }}</option>
                {% endif %}
                {% endfor %}
            </select>
        </div>
        <div class="form-group mb-3 ">
            <label for="depart" class="control-label">Depart</label>
            <select name="depart" id="depart" class="form-select select2 rounded-0">
                {% if not schedule.location %}
                <option selected></option>
                {% else %}
                <option ></option>
                {% endif %}
                {% for location in locations %}
                {% if schedule.depart.id == location.id %}
                <option value="{{ location.id }}" selected>{{ location }}</option>
                {% else %}
                <option value="{{ location.id }}" >{{ location }}</option>
                {% endif %}
                {% endfor %}
            </select>
        </div>
        <div class="form-group mb-3 ">
            <label for="destination" class="control-label">Destination</label>
            <select name="destination" id="destination" class="form-select select2 rounded-0">
                {% if not schedule.location %}
                <option selected></option>
                {% else %}
```

```
            <option ></option>
            {% endif %}
            {% for location in locations %}
            {% if schedule.destination.id == location.id %}
            <option value="{{ location.id }}" selected>{{ location }}</option>
            {% else %}
            <option value="{{ location.id }}" >{{ location }}</option>
            {% endif %}
            {% endfor %}
        </select>
    </div>
    <div class="form-group mb-3">
        <label for="schedule" class="control-label">Schedule</label>
        <input class="form-control rounded-0" name="schedule" id="schedule" type="datetime-local" value="{{ schedule.schedule|date:'Y-m-d\TH:i' }}" required>
    </div>
    <div class="form-group mb-3">
        <label for="fare" class="control-label">Fare</label>
        <input class="form-control rounded-0" name="fare" id="fare" type="number" step="any" value="{{ schedule.fare }}" required>
    </div>
    <div class="form-group mb-3 ">
        <label for="status" class="control-label">Status</label>
        <select name="status" id="status" class="form-select rounded-0">
            {% if not schedule.status or schedule.status == '1' %}
            <option value="1" selected>Active</option>
            {% else %}
            <option value="1" >Active</option>
            {% endif %}
            {% if schedule.status == '2' %}
            <option value="2" selected>Cancelled</option>
            {% else %}
            <option value="2" >Cancelled</option>
            {% endif %}
            </select>
        </div>
    </form>
</div>
<script>
    $(function() {
        $('.select2').select2({
            width:"100%",
            placeholder: "Please Select Here",
            dropdownParent:$('#uni_modal')
        })
        $('#schedule-form').submit(function(e) {
            e.preventDefault();
            var _this = $(this)
            $('.err-msg').remove();
            var el = $('<div>')
            el.addClass("alert alert-danger err-msg")
            el.hide()
            if (_this[0].checkValidity() == false) {
                _this[0].reportValidity();
                return false;
            }
            start_loader();
            $.ajax({
                url: "{% url 'save-schedule' %}",
                data: new FormData($(this)[0]),
                cache: false,
                contentType: false,
                processData: false,
                method: 'POST',
                type: 'POST',
                dataType: 'json',
                error: err => {
                    console.log(err)
                    alert("An error occured ", 'error');
                    end_loader();
                },
                success: function(resp) {
                    if (typeof resp == 'object' && resp.status == 'success') {
                        el.removeClass("alert alert-danger err-msg ")
                        location.reload()
                    } else if (resp.status == 'failed' && !!resp.msg) {
                        el.html(resp.msg)
                    } else {
                        el.text("An error occured ", 'error');
                        end_loader();
                        console.err(resp)
                    }
                    _this.prepend(el)
                    el.show('slow')
                    $("html, body, .modal ").scrollTop(0);
                    end_loader()
                }
            })
        })
    })
</script>
```

profile

```
<!DOCTYPE html>
<html>
<head>
    <title>Profile</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
    <style>
        body {
            background-image: url("");
            background-size: cover;
            background-position: center;
            background-attachment: fixed;
            background-repeat: no-repeat;
        }
        .circular-image {
            border-radius: 50%;
            width: 150px; /* Adjust the size as needed */
            height: 150px; /* Adjust the size as needed */
            object-fit: cover; /* To maintain aspect ratio and cover the circular area */
        }
        .profile-card {
            background-color: rgba(135, 206, 235, 0.7); /* Sky blue with 70% opacity */
            padding: 20px;
            text-align: center; /* Center the content horizontally */
            margin: 0 auto; /* Center the content horizontally */
            margin-top: 50px; /* Adjust the top margin as needed */
        }
        .card-text {
            margin-top: 10px; /* Adjust the top margin for other fields */
        }
    </style>
</head>
<body>
    <div class="container">
        <h1 class="my-4"></h1>
        <div class="card w-50 profile-card">
            <div class="card-body">
              {% if request.user.userprofile.profile_picture %}
              <img src="{{ request.user.userprofile.profile_picture.url }}" alt="Profile Picture"
class="circular-image img-fluid">
              {% else %}
              <p class="card-text">No profile picture uploaded.</p>
              {% endif %}
```

```
                <!-- Your existing profile information -->
                <p class="card-text">Username: {{ request.user.username }}</p>
                <p class="card-text">Email: {{ request.user.email }}</p>
                <p class="card-text">Phone Number: {{ request.user.phone_number }}</p>
                <p class="card-text">Age: {{ request.user.userprofile.age }}</p>
                <p class="card-text">Gender: {{ request.user.userprofile.gender }}</p>
                <p class="card-text">City: {{ request.user.userprofile.city }}</p>
                <p class="card-text">Date of Birth: {{ request.user.userprofile.date_of_birth }}</p>
            </div>
        </div>
    </div>
</body>
</html>
```

Update_profile

```
<!DOCTYPE html>
<html>
<head>
    <title>Update Profile</title>
    <!-- Add Bootstrap CSS link -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <!-- Add your custom CSS file if needed -->
    <link rel="stylesheet" href="custom.css">
    <style>
        /* Custom CSS for form styling */
        .custom-form {
            max-width: 400px;
            margin: 0 auto;
        }
    </style>
</head>
<body>
    <div class="container">
        <center><h1 class="my-4">Update Your Profile</h1></center>
        <form method="post" enctype="multipart/form-data" class="custom-form" onsubmit="return validateForm()">
            {% csrf_token %}

            <div class="form-group">
              <label for="id_username">Username:</label>
              <input type="text" class="form-control" name="username" value="stephin" maxlength="150" required="" id="id_username" disabled>
              <small class="form-text text-muted">Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.</small>
            </div>

            <div class="form-group">
              <label for="id_email">Email address:</label>
              <input type="email" class="form-control" name="email" value="hashimmuhammed2001@gmail.com" maxlength="254" id="id_email" disabled>
            </div>

            <div class="form-group">
              <label for="id_phone_number">Phone number:</label>
              <input type="text" class="form-control" name="phone_number" value="+913288747633" maxlength="15" id="id_phone_number" oninput="validatePhoneNumber(this)">
              <small class="form-text text-danger" id="phone_error"></small>
            </div>

            <div class="form-group">
              <label for="id_age">Age:</label>
              <input type="number" class="form-control" name="age" value="22" min="0" id="id_age" oninput="validateAge(this)">
              <small class="form-text text-danger" id="age_error"></small>
            </div>
            <div class="form-group">
              <label for="id_gender">Gender:</label>
              <select class="form-control" name="gender" id="id_gender">
                <option value="male">Male</option>
                <option value="female">Female</option>
              </select>
            </div>

            <div class="form-group">
              <label for="id_city">City:</label>
              <input type="text" class="form-control" name="city" value="palakkad" 
      maxlength="100" id="id_city" oninput="validateCity(this)">
              <small class="form-text text-danger" id="city_error"></small>
            </div>

            <div class="form-group">
              <label for="id_date_of_birth">Date of birth:</label>
              <input type="text" class="form-control" name="date_of_birth" value="2022-12-15" 
      id="id_date_of_birth" oninput="validateDateOfBirth(this)">
              <small class="form-text text-danger" id="dob_error"></small>
            </div>

            <div class="form-group">
              <label for="id_profile_picture">Profile picture:</label>
              Currently: <a 
      href="/media/profile_pictures/profile_icon_XkqlBIc.png">profile_pictures/profile_icon_XkqlBIc.
      png</a>
              <input type="checkbox" name="profile_picture-clear" id="profile_picture-clear_id">
              <label for="profile_picture-clear_id">Clear</label><br>
              Change:
              <input type="file" class="form-control-file" name="profile_picture" accept="image/*" 
      id="id_profile_picture">
            </div>
```

```
            <button type="submit" class="btn btn-primary">Update</button>
        </form>
    </div>

    <!-- Add Bootstrap JS and jQuery scripts at the end of the body if needed -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.3/dist/umd/popper.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

    <script>
        function validatePhoneNumber(input) {
            // Validation logic for phone number
            var phone = input.value;
            var phonePattern = /^\+91\d{12}$/;
            var errorElement = document.getElementById("phone_error");

            if (!phonePattern.test(phone)) {
                errorElement.innerHTML = "Invalid phone number format.";
            } else {
                errorElement.innerHTML = "";
            }
        }

        function validateAge(input) {
            // Validation logic for age
            var age = input.value;
            var errorElement = document.getElementById("age_error");

            if (isNaN(age) || age < 0) {
                errorElement.innerHTML = "Age must be a positive number.";
            } else {
                errorElement.innerHTML = "";
            }
```

```
}

function validateCity(input) {
   // Validation logic for city
   var city = input.value;
   var errorElement = document.getElementById("city_error");

   if (city.length < 2) {
      errorElement.innerHTML = "City name is too short.";
   } else {
      errorElement.innerHTML = "";
   }
}

function validateDateOfBirth(input) {
   // Validation logic for date of birth
   var dob = input.value;
   var datePattern = /^\d{4}-\d{2}-\d{2}$/;
   var errorElement = document.getElementById("dob_error");

   if (!datePattern.test(dob)) {
      errorElement.innerHTML = "Invalid date format (YYYY-MM-DD).";
   } else {
      errorElement.innerHTML = "";
   }
}

function validateForm() {
   // Additional validation logic for the entire form, if needed
   // Return true if the form is valid, or false if there are validation errors
   // Example: You can check if all error elements are empty
   return document.getElementById("phone_error").innerHTML === "" &&
        document.getElementById("age_error").innerHTML === "" &&
        document.getElementById("city_error").innerHTML === "" &&
                document.getElementById("dob_error").innerHTML === "";
       }
   </script>
</body>
</html>
```

Find_trip

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Search Form</title>
  <!-- Bootstrap CSS -->
  <link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Select2 CSS -->
  <link href="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.13/css/select2.min.css"
rel="stylesheet">
  <!-- jQuery -->
  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <!-- Bootstrap JS, Popper.js, and jQuery -->
  <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.1/dist/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
  <!-- Select2 JS -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.13/js/select2.min.js"></script>
  <style>
    body {
      display: flex;
      min-height: 100vh;
      justify-content: center;
      align-items: center;
      margin: 0;
      /* Set multiple background images */
      background-image: url('/static/MainApp/images/bus_booking3.jpg'),
                        url('/static/MainApp/images/bus_booking4.jpg');
      background-size: cover;
      background-position: center center;
      background-repeat: no-repeat;
    }

    nav.navbar {
      background-color: white;
      width: 100%;
      position: fixed;
      top: 0;
      z-index: 1000;
    }

    .navbar-nav li.nav-item a.nav-link {
      color: red !important;
      font-weight: bold;
      cursor: pointer; /* Change the mouse pointer */
      transition: color 0.3s, box-shadow 0.3s; /* Add a smooth transition for color and box-shadow
changes on hover */
      box-shadow: none; /* Remove the initial box-shadow */
      border-radius: 20px; /* Set border-radius for oval shape */
      padding: 10px 20px; /* Adjust padding for the oval shape */
    }

    .navbar-nav li.nav-item a.nav-link:hover {
      color: darkred; /* Change the color on hover */
      background-color: skyblue; /* Set hover background color to sky blue */
    }

    .search_panel {
      margin-top: 70px; /* Adjust the margin to make space for the fixed navbar */
      width: 80%;
    }

    .search_item {
      margin-bottom: 10px;
      display: flex;
      flex-direction: column; /* Display children (label and input/select) in a column */
    }

    /* Custom CSS for Select2 */
    .select2-container--bootstrap4 .select2-selection--single {
      background-color: white;
      border: 1px solid #ced4da;
      border-radius: 0.25rem;
      height: calc(2.25rem + 2px);
      padding: 0.375rem 0.75rem;
      width: 100%; /* Set the width to 100% or adjust as needed */
      min-width: 200px; /* Set a minimum width if needed */
    }

    /* Adjust the width of the container to make the dropdown wider */
    .select2-container--bootstrap4 .select2-selection--single .select2-selection__arrow {
      height: calc(2.25rem + 2px);
    }

    /* Hide error messages initially */
    .form-text.text-danger {
      display: none;
    }
  </style>
</head>
<body>

  <nav class="navbar navbar-expand-lg navbar-light">
    <a class="navbar-brand" href="#">Your Logo</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item active">
          <a class="nav-link" href="#">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Contact Us</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Cancellation</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{% url "submit_feedback" %}">Feedback</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">My Account</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Status Check</a>
        </li>
      </ul>
    </div>
</nav>

<div class="container search_panel">
  <form action="{% url 'find_trip' %}" method="GET" id="search_form_2"
class="search_panel_content d-flex flex-wrap justify-content-between align-items-center">
    {% csrf_token %}
    <div class="search_item">
      <label for="depart">Depart</label>
      <select name="depart" id="depart" class="form-select select2 rounded-0">
        <option selected></option>
        {% for location in locations %}
          <option value="{{ location.id }}">{{ location }}</option>
        {% endfor %}
      </select>
      <span id="departError" class="form-text text-danger">Please select a location</span>
    </div>
    <div class="search_item">
      <label for="destination">Destination</label>
      <select name="destination" id="destination" class="form-select select2 rounded-0">
        <option selected></option>
        {% for location in locations %}
          <option value="{{ location.id }}">{{ location }}</option>
        {% endfor %}
      </select>
      <span id="destinationError" class="form-text text-danger">Please select a location</span>
    </div>
    <div class="search_item">
      <label for="journeyDate">Journey Date</label>
      <input type="date" id="journeyDate" class="check_in search_input form-control" min="{{
today }}" required="required">
      <span id="journeyDateHelp" class="form-text text-danger">Invalid journey date</span>
    </div>
    <div class="search_item">
      <label for="returnDate">Return Date (optional)</label>
      <input type="date" id="returnDate" class="check_out search_input form-control" min="{{
today }}">
      <span id="returnDateHelp" class="form-text text-danger">Invalid return date</span>
    </div>

    <button type="submit" class="btn btn-primary search_button">Search</button>
  </form>
</div>

<script>
  $(document).ready(function() {
    // Initialize Select2 for the "Depart" field
    $('#depart').select2({
      placeholder: 'Select location',
      allowClear: true,
      theme: 'bootstrap4',
    });
    $('#destination').select2({
      placeholder: 'Select location',
      allowClear: true,
      theme: 'bootstrap4',
    });

    // Array of background images
    var images = [
        '/static/MainApp/images/bus_booking4.jpg',
        '/static/MainApp/images/bus_booking3.jpg',
        // Add more image URLs as needed
    ];

    var currentImageIndex = 0;

    // Function to change the background image
    function changeBackground() {
        $('body').css('background-image', 'url("' + images[currentImageIndex] + '")');
        currentImageIndex = (currentImageIndex + 1) % images.length;
    }

    // Change the background image every 5 seconds (adjust as needed)
    setInterval(changeBackground, 5000);

    // Hide error messages initially
    $('.form-text.text-danger').hide();

    // Your other script code goes here

  });
</script>
</body>
</html>
```

Schedule_manage_page

```
{% load static %}{% load humanize %}
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Schedule View Page</title>
  <!-- Bootstrap CSS -->
  <link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Add any additional styles as needed -->
  <style>
    body {
      background-color: #f8f9fa;
    }

    .container {
      margin-top: 30px;
    }

    h2 {
      color: #007bff;
    }

    table {
      width: 100%;
      margin-bottom: 1rem;
      color: #212529;
      border-collapse: collapse;
    }

    th,
    td {
      padding: 0.75rem;
      vertical-align: top;
      border: 1px solid #dee2e6;
    }

    th {
      background-color: #007bff;
      color: #fff;
    }

    tbody tr:hover {
      background-color: #f5f5f5;
      cursor: pointer;
    }

    .bus-number {
      font-weight: bold;
    }

    .category {
      font-size: 0.8rem;
      color: #6c757d;
    }

    .no-schedules {
      color: #dc3545;
    }
  </style>
</head>
<body>
<div class="container">
  {% if schedules %}
    <h2>Search Results</h2>
    <table class="table">
      <thead>
        <tr>
          <th>Schedule Time</th>
          <th>Bus</th>
          <th>Route (From - To)</th>
          <th>Price</th>
          <th>Action</th>
        </tr>
      </thead>
      <tbody>
        {% for schedule in schedules %}
          <tr>
            <td>{{ schedule.schedule|date:"Y-m-d h:i A" }}</td>
            <td>
              <div class="bus-number">{{ schedule.bus.bus_number }}</div>
              <div class="category">{{ schedule.bus.category }}</div>
            </td>
            <td>{{ schedule.depart }} - {{ schedule.destination }}</td>
            <td>{{ schedule.fare|intcomma }}</td>
            <td>
              <button type="button" class="btn btn-primary">Book Now</button>
            </td>
          </tr>
        {% endfor %}
      </tbody>
    </table>
  {% else %}
    <p class="no-schedules">No schedules found.</p>
  {% endif %}
</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.1/dist/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
<!-- Add any additional scripts as needed -->
</body>
</html>
```

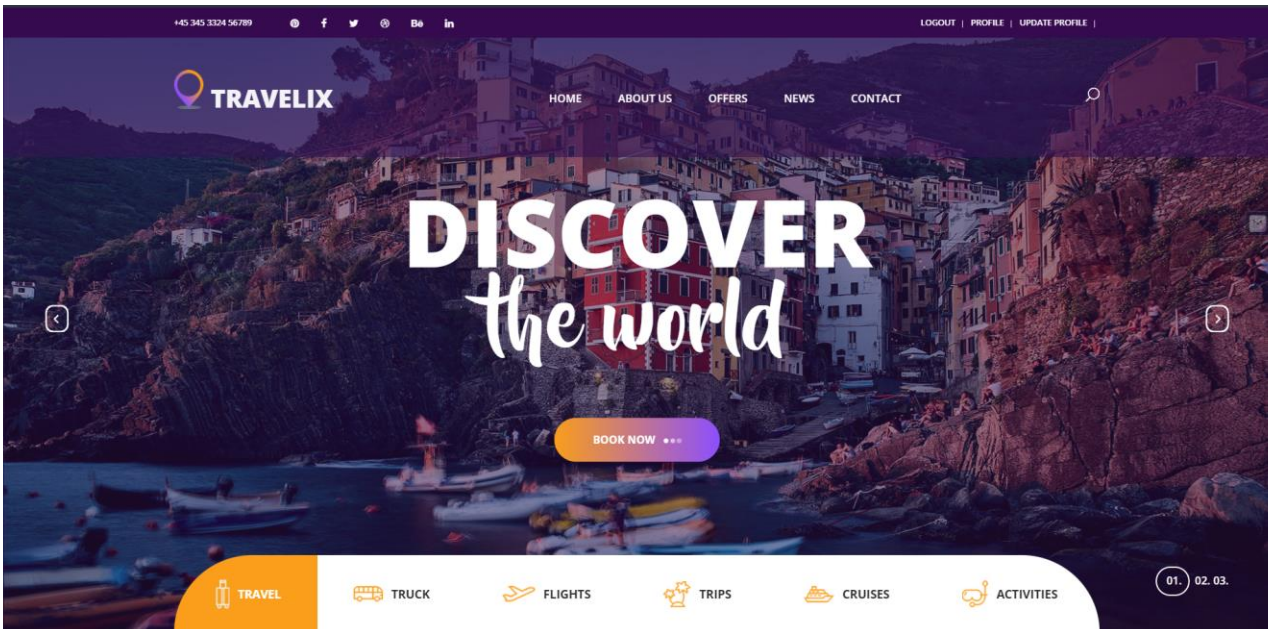
Views.py

```
from collections import UserDict
import json
from django.http import HttpRequest, HttpResponse, JsonResponse
from django.shortcuts import get_object_or_404, redirect, render
from django.contrib.auth import login, authenticate, logout
from django.views.decorators.cache import never_cache
from django.views.decorators.csrf import csrf_protect
from django.contrib.auth.decorators import login_required
from .models import Bus, Category, Location, Schedule, Users,Seat
from TransHub.settings import EMAIL_HOST_USER
from .models import Users
from django.core.mail import send_mail
from django.db.models import Q
from django.contrib import messages
from .models import Users, UserProfile
from datetime import datetime

# from django.http import HttpResponse

@never_cache
def showIndex(request):
    return render(request, 'index.html')

def about(request):
    return render(request, 'about.html')

@csrf_protect
def SignUp(request):
    if request.method == 'POST':
        username = request.POST['username']
        email = request.POST['email']
        phone = request.POST.get('phone')
        password = request.POST['password']
        confirm_password = request.POST['confirm_password']
        role='user'

        if password != confirm_password:
            return render(request, SignUp.html)
        user = Users(username=username,phone_number=phone, email=email,role=role)
        # password set
        user.set_password(password)
        #save the user to database
        user.save()
        UserProfile.objects.create(user=user)

        subject = 'Hello, '+username
        message = 'Your registration has been Successfully completed'
        from_email = EMAIL_HOST_USER
        recipient_list = [email]
        send_mail(subject, message, from_email, recipient_list)
        return redirect('login')

    return render(request,'SignUp.html')

def Log(request):
    if request.method == "POST":
        username = request.POST['username']
        user_password = request.POST['password']
        user = authenticate(username=username, password=user_password)
        if user is not None:
            if user.is_superuser:
                login(request, user)
                request.session['username'] = username
                return redirect("Admin_Home")
            login(request, user)
            return redirect('Home')
        else:
            return render(request, 'login.html', {'Error_message': 'invalid creadential!!'})

    return render(request,'login.html')

def logout_user(request):
    if request.user.is_authenticated:
        logout(request)
    return redirect('showindex')

import random
def generateOTP():
    generatedOTP = "".join(str(random.randint(0, 9)) for _ in range(6))
    return generatedOTP


#@login_required(login_url='login')
@never_cache
def Home(request):
    return render(request, 'home.html')

def validateGlobalEmail(request):
    email = request.GET['email']
    cpnm = "TransHub Corp. Ltd."
    otp = generateOTP()
    subject = 'Hello, Django Email!'
    message = 'Here is Your OTP.'+otp
    from_email = EMAIL_HOST_USER
    recipient_list = [email]
    data = {
        "exists": send_mail(subject, message, f"{cpnm} <{from_email}>", recipient_list),
        "otp": otp
    }
    return JsonResponse(data)


def check_username(request):
    username = request.GET.get('username', '')
    user_exits = Users.objects.filter(username=username).exists()
    return JsonResponse({'exists': user_exits})

def check_email(request):
    email = request.GET.get('email', '').lower
    email_exists = Users.objects.filter(email=email).exists()
    return JsonResponse({'exists': email_exists})

def Staff_signUp(request):
    if request.method == 'POST':
        # Similar to the SignUp view, include the role field
        username = request.POST['username']
        email = request.POST['email']
        phone = request.POST.get('phone')
        password = request.POST['password']
        confirm_password = request.POST['confirm_password']
        role = 'Staff'  # Set the role to 'staff' for staff registration

        if password != confirm_password:
            return render(request, 'StaffSignUp.html')

        user = Users(username=username, phone_number=phone, email=email, role=role)
        user.set_password(password)
        user.save()
        return redirect('login')
    return render(request, 'StaffSignUp.html')

def Admin_Home(request):
     if not request.user.is_authenticated:
         return redirect ('login')
     return render(request, 'adminhome.html')

def user_account(request):
    role_filter = request.GET.get('role')
    users = Users.objects.filter(~Q(is_superuser=True))  # Exclude superusers by default

    if role_filter:
        users = users.filter(role=role_filter)

    context = {'User_profiles': users, 'role_filter': role_filter}
    return render(request, 'usertable.html', context)

from django.template.loader import render_to_string
from django.utils.html import strip_tags


#def ActivateAccount(request):
#     return render(request, 'activate.html')

def activate_user(request, user_id):
    user = get_object_or_404(Users, id=user_id)

    if not user.is_active:
        user.is_active = True
        user.save()
        messages.success(request, f"User '{user.username}' has been activated by the admin, and an
email has been sent.")

        # Send activation email to the user
        subject = "Account Activation"
        html_message = render_to_string('activation_email.html', {'user': user})
        plain_message = strip_tags(html_message)
        from_email = "transhubcorporationltd@gmail.com"  # Update with your email
        recipient_list = [user.email]
        send_mail(subject, plain_message, from_email, recipient_list, html_message=html_message)

    else:
        messages.warning(request, f"User '{user.username}' is activated.")

    return redirect('user_account')


def activatation_email(request):
     return render(request, 'activation_email.html')

def deactivate_user(request, user_id):
    user = get_object_or_404(Users, id=user_id)
    if user.is_active:
        user.is_active = False
        user.save()

        # Send deactivation email
        subject = 'Account Deactivation'
        message = 'Your account has been deactivated by the admin.'
        from_email = 'transhubcorporationltd@gmail.com'  # Replace with your email
        recipient_list = [user.email]
        html_message = render_to_string('deactivation_email.html', {'user': user})

        send_mail(subject, message, from_email, recipient_list, html_message=html_message)

        messages.success(request, f"User '{user.username}' has been deactivated, and an email has
been sent.")
    else:
        messages.warning(request, f"User '{user.username}' is deactivated.")
    return redirect('user_account')

def deactivation_email(request):
     return render(request, 'deactivation_email.html')

# views.py
# views.py
from django.shortcuts import render, redirect
from .forms import SaveBus, SaveCategory, SaveLocation, SaveSchedule, UserProfileForm,
AdditionalProfileForm

def update_profile(request):
    # Check if the UserProfile exists for the user, and create it if it doesn't
    try:
        user_profile = UserProfile.objects.get(user=request.user)
    except UserProfile.DoesNotExist:
        user_profile = UserProfile(user=request.user)
        user_profile.save()

    user_form = UserProfileForm(request.POST, instance=request.user)
    profile_form = AdditionalProfileForm(request.POST, request.FILES, instance=user_profile)

    if request.method == 'POST':
        if user_form.is_valid() and profile_form.is_valid():
            user_form.save()
            profile_form.save()
            return redirect('profile')
    else:
        user_form = UserProfileForm(instance=request.user)
        profile_form = AdditionalProfileForm(instance=user_profile)

    return render(request, 'update_profile.html', {'user_form': user_form, 'profile_form':
profile_form})
```

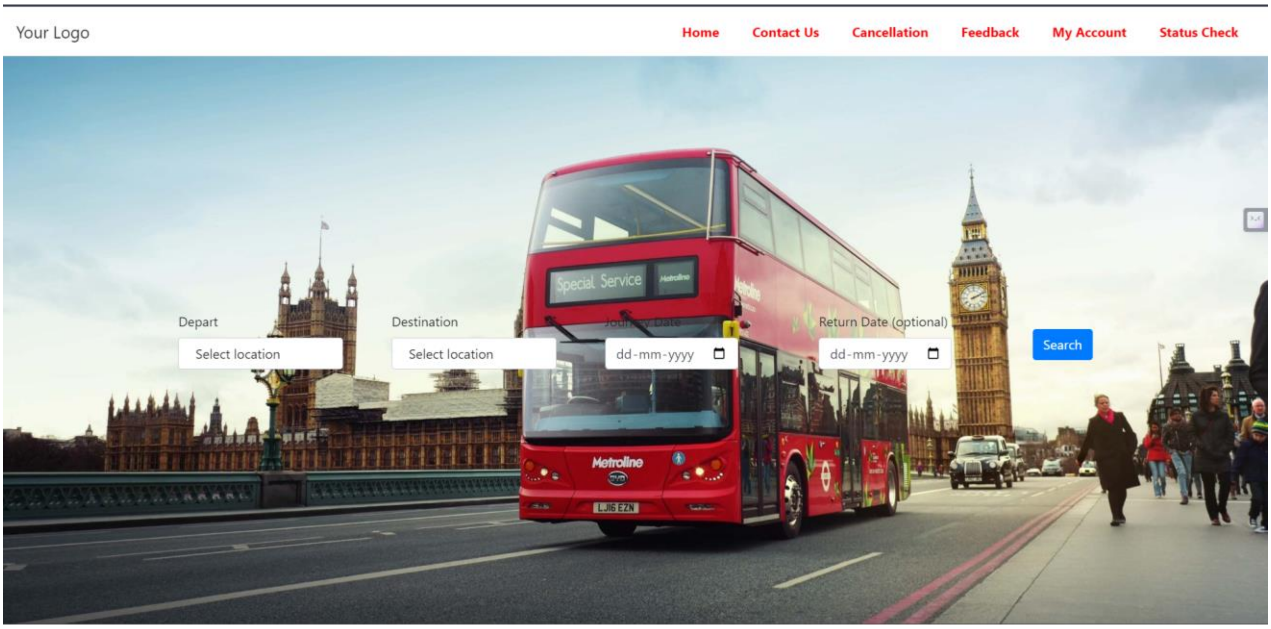
```
def profile(request):
    return render(request, 'profile.html')

#context text
context = {
    'page_title' : 'File Management System',
}

#category
@login_required
def category_mgt(request):
    context['page_title'] = "Bus Categories"
    categories = Category.objects.all()
    context['categories'] = categories

    return render(request, 'category_mgt.html', context)

#save_category
@login_required
def save_category(request):
    resp = {'status':'failed','msg':''}
    if request.method == 'POST':
        if (request.POST['id']).isnumeric():
            category = Category.objects.get(pk=request.POST['id'])
        else:
            category = None
        if category is None:
            form = SaveCategory(request.POST)
        else:
            form = SaveCategory(request.POST, instance= category)
        if form.is_valid():
            form.save()
            messages.success(request, 'Category has been saved successfully.')
            resp['status'] = 'success'
        else:
            for fields in form:
                for error in fields.errors:
                    resp['msg'] += str(error + "<br>")
    else:
        resp['msg'] = 'No data has been sent.'
    return HttpResponse(json.dumps(resp), content_type = 'application/json')

#manage_category
@login_required
def manage_category(request, pk=None):
    context['page_title'] = "Manage Category"
    if not pk is None:
        category = Category.objects.get(id = pk)
        context['category'] = category
    else:
        context['category'] = {}

    return render(request, 'manage_category.html', context)

#delete_category
@login_required
def delete_category(request):
    resp = {'status':'failed', 'msg':''}

    if request.method == 'POST':
        try:
            category = Category.objects.get(id = request.POST['id'])
            category.delete()
            messages.success(request, 'Category has been deleted successfully')
            resp['status'] = 'success'
        except Exception as err:
            resp['msg'] = 'Category has failed to delete'
            print(err)

    else:
        resp['msg'] = 'Category has failed to delete'

    return HttpResponse(json.dumps(resp), content_type="application/json")

# Location
@login_required
def location_mgt(request):
    context['page_title'] = "Locations"
    locations = Location.objects.all()
    context['locations'] = locations

    return render(request, 'location_mgt.html', context)

#save location
@login_required
def save_location(request):
    resp = {'status':'failed','msg':''}
    if request.method == 'POST':
        if (request.POST['id']).isnumeric():
            location = Location.objects.get(pk=request.POST['id'])
        else:
            location = None
        if location is None:
            form = SaveLocation(request.POST)
        else:
            form = SaveLocation(request.POST, instance= location)
        if form.is_valid():
            form.save()
            messages.success(request, 'Location has been saved successfully.')
            resp['status'] = 'success'
        else:
            for fields in form:
                for error in fields.errors:
                    resp['msg'] += str(error + "<br>")
    else:
        resp['msg'] = 'No data has been sent.'
    return HttpResponse(json.dumps(resp), content_type = 'application/json')

#manage location
@login_required
def manage_location(request, pk=None):
    context['page_title'] = "Manage Location"
    if not pk is None:
        location = Location.objects.get(id = pk)
        context['location'] = location
    else:
        context['location'] = {}

    return render(request, 'manage_location.html', context)

#delete location
@login_required
def delete_location(request):
    resp = {'status':'failed', 'msg':''}

    if request.method == 'POST':
        try:
            location = Location.objects.get(id = request.POST['id'])
            location.delete()
            messages.success(request, 'Location has been deleted successfully')
            resp['status'] = 'success'
        except Exception as err:
            resp['msg'] = 'location has failed to delete'
            print(err)

    else:
        resp['msg'] = 'location has failed to delete'

    return HttpResponse(json.dumps(resp), content_type="application/json")

# bus
@login_required
def bus_mgt(request):
    context['page_title'] = "Buses"
    buses = Bus.objects.all()
    context['buses'] = buses

    return render(request, 'bus_mgt.html', context)

@login_required
def save_bus(request):
    resp = {'status':'failed','msg':"}
    if request.method == 'POST':
        if (request.POST['id']).isnumeric():
            bus = Bus.objects.get(pk=request.POST['id'])
        else:
            bus = None
        if bus is None:
            form = SaveBus(request.POST)
        else:
            form = SaveBus(request.POST, instance= bus)
        if form.is_valid():
            form.save()
            messages.success(request, 'Bus has been saved successfully.')
            resp['status'] = 'success'
        else:
            for fields in form:
                for error in fields.errors:
                    resp['msg'] += str(error + "<br>")
    else:
        resp['msg'] = 'No data has been sent.'
    return HttpResponse(json.dumps(resp), content_type = 'application/json')

@login_required
def manage_bus(request, pk=None):
    context['page_title'] = "Manage Bus"
    categories = Category.objects.filter(status = 1).all()
    context['categories'] = categories
    if not pk is None:
        bus = Bus.objects.get(id = pk)
        context['bus'] = bus
    else:
        context['bus'] = {}

    return render(request, 'manage_bus.html', context)

@login_required
def delete_bus(request):
    resp = {'status':'failed', 'msg':"}

    if request.method == 'POST':
        try:
            bus = Bus.objects.get(id = request.POST['id'])
            bus.delete()
            messages.success(request, 'Bus has been deleted successfully')
            resp['status'] = 'success'
        except Exception as err:
            resp['msg'] = 'bus has failed to delete'
            print(err)

    else:
        resp['msg'] = 'bus has failed to delete'
    return HttpResponse(json.dumps(resp), content_type="application/json")

# schedule
@login_required
def schedule_mgt(request):
    context['page_title'] = "Trip Schedules"
    schedules = Schedule.objects.all()
    context['schedules'] = schedules

    return render(request, 'schedule_mgt.html', context)

@login_required
def save_schedule(request):
    resp = {'status':'failed','msg':"}
    if request.method == 'POST':
        if (request.POST['id']).isnumeric():
            schedule = Schedule.objects.get(pk=request.POST['id'])
        else:
            schedule = None
        if schedule is None:
            form = SaveSchedule(request.POST)
        else:
            form = SaveSchedule(request.POST, instance= schedule)
        if form.is_valid():
            form.save()
            messages.success(request, 'Schedule has been saved successfully.')
            resp['status'] = 'success'
        else:
            for fields in form:
                for error in fields.errors:
                    resp['msg'] += str(error + "<br>")
    else:
        resp['msg'] = 'No data has been sent.'
    return HttpResponse(json.dumps(resp), content_type = 'application/json')

@login_required
def manage_schedule(request, pk=None):
    context['page_title'] = "Manage Schedule"
    buses = Bus.objects.filter(status = 1).all()
    locations = Location.objects.filter(status = 1).all()
    context['buses'] = buses
    context['locations'] = locations
    if not pk is None:
        schedule = Schedule.objects.get(id = pk)
        context['schedule'] = schedule
    else:
        context['schedule'] = {}

    return render(request, 'manage_schedule.html', context)
```

```
@login_required
def delete_schedule(request):
    resp = {'status':'failed', 'msg':''}

    if request.method == 'POST':
        try:
            schedule = Schedule.objects.get(id = request.POST['id'])
            schedule.delete()
            messages.success(request, 'Schedule has been deleted successfully')
            resp['status'] = 'success'
        except Exception as err:
            resp['msg'] = 'schedule has failed to delete'
            print(err)

    else:
        resp['msg'] = 'Schedule has failed to delete'

    return HttpResponse(json.dumps(resp), content_type="application/json")


class Seat:
    def __init__(self, number, is_reserved, is_women_seat):
        self.number = number
        self.is_reserved = is_reserved
        self.is_women_seat = is_women_seat

def bus_seat_map(request):
    # Example: Creating a list of 40 seats with alternating reserved and available status,
    # and 5 women seats
    num_seats = 40
    num_women_seats = 5
    seats = [Seat(number=i, is_reserved=i % 2 == 0, is_women_seat=i < num_women_seats) for i
in range(1, num_seats + 1)]

    return render(request, 'bus_grid_seat.html', {'seats': seats})

def book_seat(request):
    if request.method == 'POST':
        selected_seats = request.POST.getlist('selected_seats')
        # Handle booking logic here
        return HttpResponse(f'Selected Seats: {", ".join(selected_seats)}')
    else:
        return HttpResponse('Invalid request method')


# find trip set

from django.shortcuts import render
from django.http import Http404
from datetime import datetime
from .models import Location, Schedule
from django.shortcuts import render, redirect
from django.urls import reverse
from datetime import datetime
from .models import Schedule, Location

def find_trip(request):
    context = {}
    context['page_title'] = 'Find Trip Schedule'
    context['locations'] = Location.objects.filter(status=1).all()
    today = datetime.today().strftime("%Y-%m-%d")
    context['today'] = today

    if request.method == 'GET':
        depart = request.GET.get('depart')
        destination = request.GET.get('destination')
        journey_date = request.GET.get('journeyDate')
        return_date = request.GET.get('returnDate')

        # Basic input validation
        if not depart and not destination and not journey_date:
            context['error_message'] = 'Please provide at least one search parameter.'
            return render(request, 'find_trip.html', context)

        # Additional validation can be added here if needed

        # Validate depart and destination locations
        try:
            depart_location = Location.objects.get(pk=depart)
        except Location.DoesNotExist:
            raise Http404('Depart location does not exist')

        try:
            destination_location = Location.objects.get(pk=destination)
        except Location.DoesNotExist:
            raise Http404('Destination location does not exist')

        # Query Schedule model based on search parameters
        schedules = Schedule.objects.all()

        if depart:
            schedules = schedules.filter(depart=depart_location)
        if destination:
            schedules = schedules.filter(destination=destination_location)
        if journey_date:
            # Validate journey_date format (YYYY-MM-DD)
            try:
                journey_date = datetime.strptime(journey_date, '%Y-%m-%d').date()
            except ValueError:
                context['error_message'] = 'Invalid journey date format.'
                return render(request, 'find_trip.html', context)
            # Filter schedules for the selected journey_date
            schedules = schedules.filter(schedule__date__contains=journey_date)

        context['schedules'] = schedules

        # Render the schedule_view_page template with the filtered schedules
        return render(request, 'schedule_view_page.html', context)

    return render(request, 'find_trip.html', context)
def schedule_view_page(request):
    context = {}

    if request.method == 'GET':
        depart = request.GET.get('depart')
        destination = request.GET.get('destination')
        journey_date = request.GET.get('journeyDate')

        # Additional validation can be added here if needed

        # Filter schedules based on the user's search parameters
        schedules = Schedule.objects.all()

        if depart:
            schedules = schedules.filter(depart__location__icontains=depart)

        if destination:
            schedules = schedules.filter(destination__location__icontains=destination)

        if journey_date:
            # Filter schedules for the selected journey_date
            schedules = schedules.filter(schedule__date__contains=journey_date)

        # Create a dictionary to store unique buses based on bus_number
        unique_buses = {}

        # Iterate through schedules and keep only the first occurrence of each bus
        for schedule in schedules:
            bus_number = schedule.bus.bus_number
            if bus_number not in unique_buses:
                unique_buses[bus_number] = schedule

        # Extract the unique schedules from the dictionary
        unique_schedules = list(unique_buses.values())

        context['schedules'] = unique_schedules

    return render(request, 'schedule_view_page.html', context)

from django.shortcuts import render, redirect
from .models import Feedback
from django.contrib.auth.decorators import login_required

@never_cache
@login_required(login_url="login")
def submit_feedback(request):
    if request.method == "POST":
        feedback_message = request.POST.get('feedback_message')
        if feedback_message:
            Feedback.objects.create(User=request.user, message=feedback_message)
            # You can add additional logic here (e.g., sending a confirmation email)
            return redirect('feedback_thankyou')

    return render(request, 'feedback_form.html')


def feedback_thankyou(request):
     return render(request,'feedback_thankyou.html')


from django.shortcuts import render
from .models import Feedback

def adminfeedback(request):
    feedback_list = Feedback.objects.all()
    return render(request, 'adminfeedback.html', {'feedback_list': feedback_list})

from django.shortcuts import render

def seat_reservation(request):
    rows = range(1, 5)
    cols = range(1, 11)

    return render(request, 'seat_reservation.html', {'rows': rows, 'cols': cols})
```

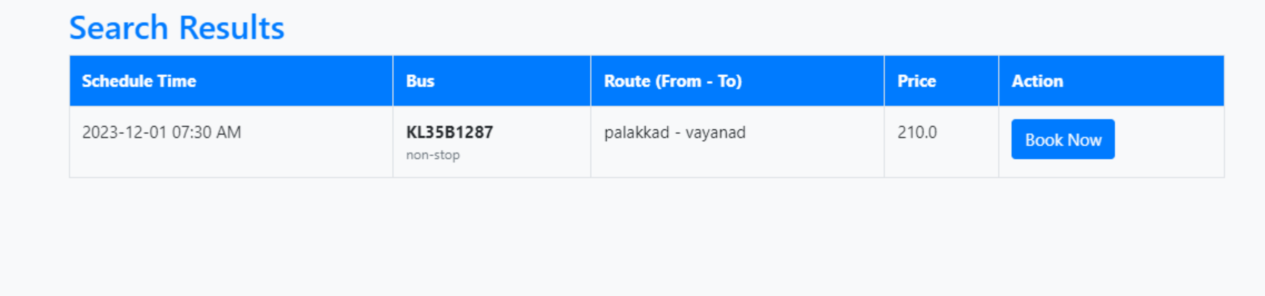
Home



Booking



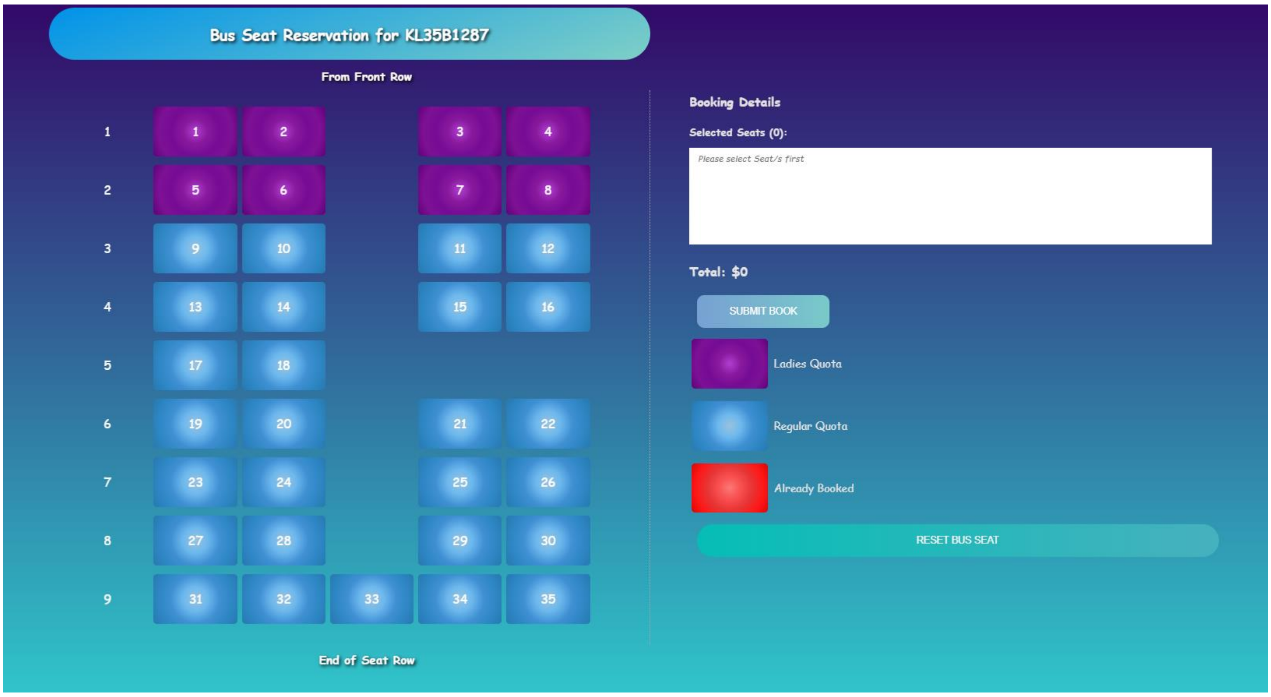
Schedule booking



Search Results

Schedule Time	Bus	Route (From - To)	Price	Action
2023-12-01 07:30 AM	**KL35B1287** non-stop	palakkad - vayanad	210.0	Book Now

Reservation seat



Bus Seat Reservation for KL35B1287

From Front Row

1	1	2		3	4
2	5	6		7	8
3	9	10		11	12
4	13	14		15	16
5	17	18			
6	19	20		21	22
7	23	24		25	26
8	27	28		29	30
9	31	32	33	34	35

End of Seat Row

Booking Details

Selected Seats (0):

Please select Seat/s first

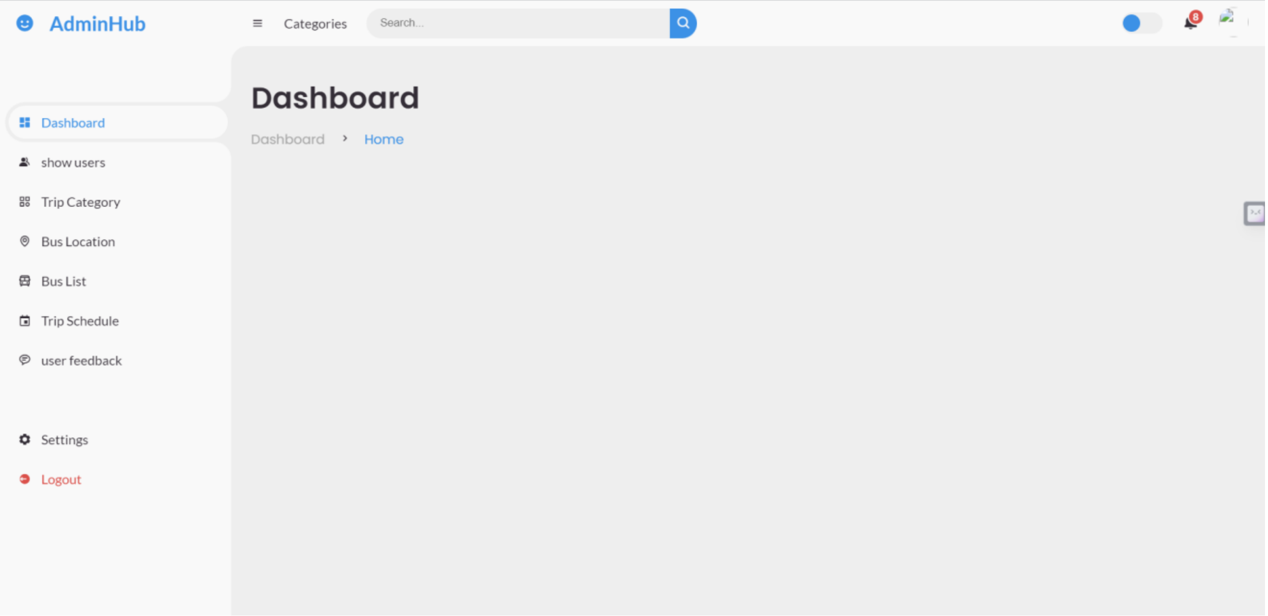
Total: $0

🟪 Ladies Quota

🟦 Regular Quota

🟥 Already Booked

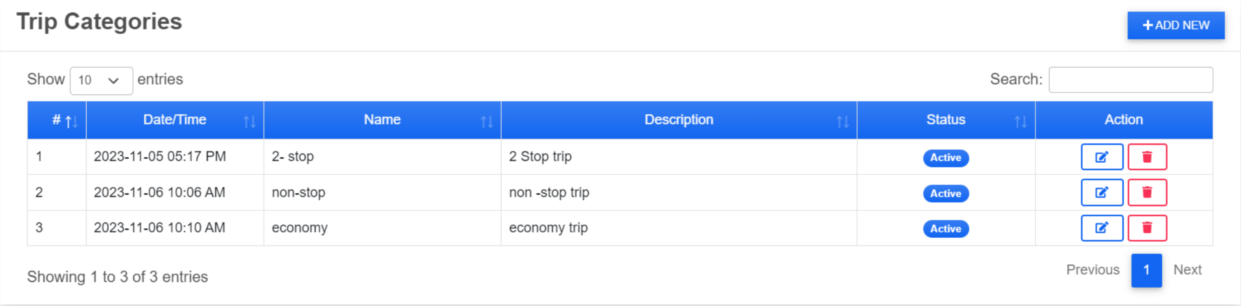
Admin Home



Dashboard

Dashboard › Home

Trip_category

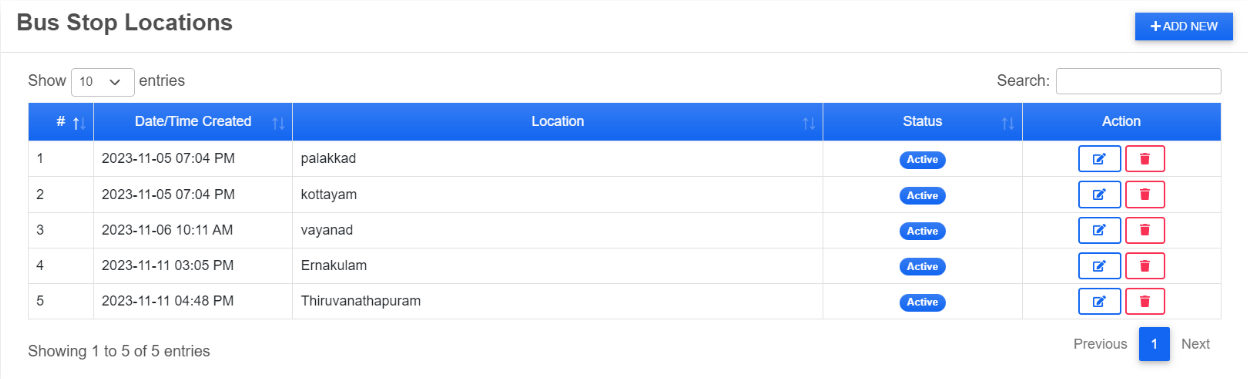


Trip Categories

#	Date/Time	Name	Description	Status	Action
1	2023-11-05 05:17 PM	2- stop	2 Stop trip	Active	✎ 🗑
2	2023-11-06 10:06 AM	non-stop	non -stop trip	Active	✎ 🗑
3	2023-11-06 10:10 AM	economy	economy trip	Active	✎ 🗑

Showing 1 to 3 of 3 entries

Location

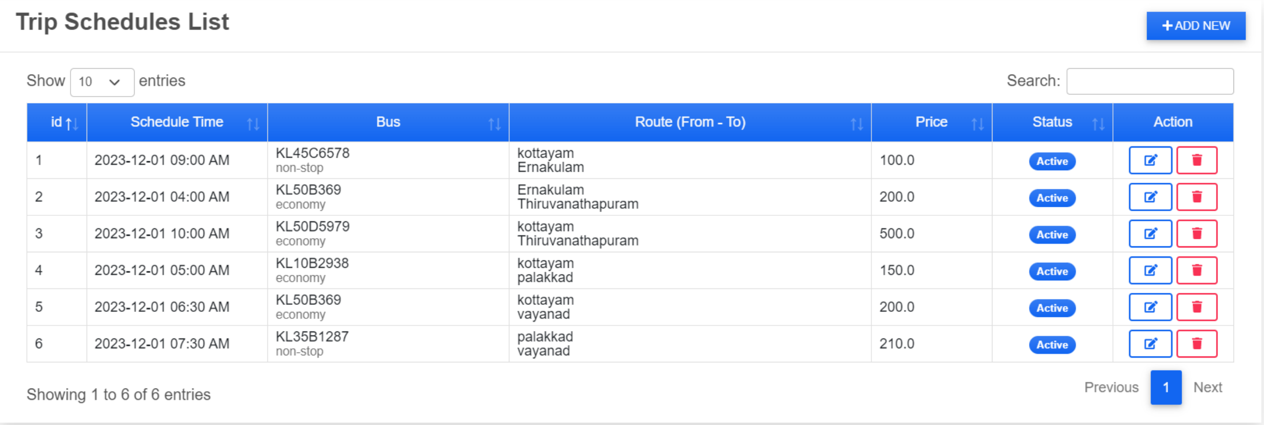


Bus Stop Locations

#	Date/Time Created	Location	Status	Action
1	2023-11-05 07:04 PM	palakkad	Active	✎ 🗑
2	2023-11-05 07:04 PM	kottayam	Active	✎ 🗑
3	2023-11-06 10:11 AM	vayanad	Active	✎ 🗑
4	2023-11-11 03:05 PM	Ernakulam	Active	✎ 🗑
5	2023-11-11 04:48 PM	Thiruvanathapuram	Active	✎ 🗑

Showing 1 to 5 of 5 entries

Schedule trip



Trip Schedules List

id	Schedule Time	Bus	Route (From - To)	Price	Status	Action
1	2023-12-01 09:00 AM	KL45C6578 non-stop	kottayam Ernakulam	100.0	Active	
2	2023-12-01 04:00 AM	KL50B369 economy	Ernakulam Thiruvanathapuram	200.0	Active	
3	2023-12-01 10:00 AM	KL50D5979 economy	kottayam Thiruvanathapuram	500.0	Active	
4	2023-12-01 05:00 AM	KL10B2938 economy	kottayam palakkad	150.0	Active	
5	2023-12-01 06:30 AM	KL50B369 economy	kottayam vayanad	200.0	Active	
6	2023-12-01 07:30 AM	KL35B1287 non-stop	palakkad vayanad	210.0	Active	

Showing 1 to 6 of 6 entries