# BUS TRACKING AND SEAT OCCUPANCY MONITORING SYSTEM

PROJECT REPORT 2025-2026

Submitted By


**24591650 VINCILIMARY**

**24507211 PAVITHRA**

**24507206 MANIKUMAR**

**24507179 ANJITHA**


Under the guidance of

**Mrs. SRIDEVI .C B.E.,(CSE)**

HOD - COMPUTER & IT

Diploma in Computer Engineering

of the Directorate of Technical Education, Government of Tamil Nadu



DEPARTMENT OF COMPUTER ENGINEERING

**AKT MEMORIAL POLYTECHNIC COLLEGE**

KALLAKURICHI-606202.

# AKT MEMORIAL POLYTECHNIC COLLEGE

KALLAKURICHI-606202

## Department of Computer Engineering

**BONAFIDE CERTIFICATE**

This is certified that this project work entitled **Bus Tracking and Seat Occupancy Monitoring System** has been submitted **VINCILIMARY, PAVITHRA, MANIKUMAR, ANJITHA** in the partial fulfilment of the requirements for the award of Diploma in Computer Engineering during the academic year 2025-2026, who carried out the project work under our supervision.

| | |
|---|---|
| **Project Guide** | **Head of the Department** |
| **Mrs. C.SRIDEVI B.E.,(CSE)** | **Mrs. C.SRIDEVI B.E.,(CSE)** |
| **HOD-COMPUTER & IT** | **HOD-COMPUTER & IT** |

This is to certify that_____
was examined for the project work viva-voce held on_____

INTERNAL EXAMINER EXTERNAL EXAMINER

# Acknowledgement

We express our sincere and profound thanks to our chairman **Mr.A.K.T.Mahendiran B.Com.,** for creating this magnificent edifice of learning by providing all necessary facilities to complete diploma engineering course successfully.

We express our gratitude to our principal **Mr.P.K.Kabilar M.E.,MBA.,** for constant encouragement and facilities provided towards the successful completion of our project work.

At the same time we would like to express our heartfelt thanks to our head of the department **Mrs.C.Sridevi.,B.E.,(CSE).,** for guiding and needful advices and time to complete this project.

We would like to show our gratitude to our department staffs for all instructions and guidance given throughout.

Our sincere thanks and affection to our parents and friends who gave hand in all our steps.

# CONTENTS

# ABSTRACT

The Bus Tracking and Seat Occupancy Monitoring System is a smart transportation management solution designed to improve the efficiency, transparency, and convenience of public and private bus operations. The system combines bus ticket booking, live vehicle monitoring, seat availability tracking, passenger management, and administrative control into a single digital platform. It enables passengers to search routes, check schedules, view available seats, book tickets online, and receive instant confirmations. At the same time, transport operators can manage fleets, routes, trip timings, staff roles, fares, and booking records through a centralized system. This project addresses common problems in traditional bus transport such as manual booking errors, overbooking, lack of live information, poor seat utilization, and limited communication between passengers and operators.

A major strength of the proposed system is its ability to provide real-time seat occupancy monitoring along with route and trip updates. Each bus trip can be associated with a seat layout, booking count, passenger list, and status information so that users and administrators always know current seat availability. The platform supports multi-role access for administrators, supervisors, counter staff, and drivers, allowing each user type to perform authorized tasks efficiently. Admins can oversee the entire operation, counters can perform walk-in bookings, supervisors can monitor schedules and occupancy, and drivers can access trip-related details. This structured access improves operational control and reduces confusion in daily transport management.

The system also includes a professional ticket booking platform with advanced features such as ticket holding, cancellation policies, automated fare calculation, schedule-based trip assignment, and instant ticket generation. Passengers benefit from a user-friendly experience where they can search buses by source, destination, and date, choose seats visually, make secure payments, and receive digital tickets by email or SMS. Integration with multiple payment gateways makes the platform flexible for users across different regions. Additional features such as multilingual support, notification services, verification mechanisms, and transport analytics further enhance usability and security.

Overall, the Bus Tracking and Seat Occupancy Monitoring System provides a secure, scalable, and technology-driven solution for modern transport services. It reduces manual workload, improves booking accuracy, increases seat utilization, and enhances passenger satisfaction through live updates and digital convenience. The project is highly suitable as a final-year software engineering system because it demonstrates the practical implementation of database management, role-based access control, online booking workflows, real-time monitoring, and reporting dashboards. By digitizing both passenger services and backend transport operations, the system contributes to smarter and more reliable bus transportation management.

# 1. INTRODUCTION

The rapid growth of urbanization and population has significantly increased the demand for efficient and reliable public transportation systems. Among various modes of transport, buses remain one of the most widely used and affordable options for daily commuting as well as long-distance travel. However, traditional bus management systems still rely heavily on manual processes such as paper-based ticketing, unorganized scheduling, and lack of real-time monitoring. These limitations often lead to issues like overbooking, inaccurate seat allocation, delays in communication, and poor passenger experience. To overcome these challenges, there is a need for a modern, technology-driven solution that can streamline operations and enhance service quality.

The Bus Tracking and Seat Occupancy Monitoring System is designed to address these problems by integrating digital booking, real-time monitoring, and centralized management into a single platform. This system allows passengers to easily search for buses, check seat availability, view schedules, and book tickets online from anywhere. At the same time, transport operators can manage fleets, routes, schedules, and pricing efficiently through a user-friendly dashboard. By automating key operations, the system minimizes human errors and improves the overall efficiency of transport services.

One of the key features of this system is seat occupancy monitoring, which ensures that both passengers and administrators have accurate, up-to-date information about available and booked seats. This helps prevent overbooking and allows better utilization of resources. Additionally, the system supports real-time bus tracking, enabling users to know the current status and location of buses. This feature enhances transparency and helps passengers plan their journeys more effectively. The integration of multiple user roles—such as admin, supervisor, counter staff, and driver—ensures that each stakeholder can perform their tasks with proper access control and accountability.

Furthermore, the system incorporates advanced functionalities such as online payment integration, ticket cancellation policies, instant ticket generation, and automated

notifications via SMS or email. These features provide convenience and flexibility to passengers while also improving operational control for administrators. The platform is designed to be secure, scalable, and adaptable, supporting multiple languages and high data volumes, making it suitable for both small transport agencies and large-scale bus networks.

In conclusion, the Bus Tracking and Seat Occupancy Monitoring System represents a significant step towards digitizing transportation services. It not only enhances the user experience but also optimizes operational efficiency and decision-making for transport providers. This project is highly relevant in today's digital era, as it demonstrates the practical application of web technologies, database systems, and real-time data handling in solving real-world transportation challenges.

# 2. LITERATURE SURVEY

he development of a Bus Tracking and Seat Occupancy Monitoring System is supported by research and practical implementations in the fields of intelligent transportation systems, online reservation platforms, fleet management, and real-time monitoring technologies. Existing literature shows that transportation organizations across the world are moving from manual operations to digital systems in order to improve scheduling accuracy, passenger convenience, and resource utilization. Traditional bus reservation methods, which depend on counters, handwritten logs, and disconnected trip management, are often inefficient and error-prone. Researchers and developers have therefore proposed various software-based solutions that combine route planning, passenger booking, digital ticket generation, and operational control within centralized platforms.

One important area covered in earlier studies is online bus ticket booking systems. These systems were introduced to reduce the burden on physical booking counters and to provide passengers with the convenience of advance reservation. Literature on e-ticketing systems explains how web-based booking platforms can improve accessibility by allowing users to search routes, compare timings, view fare details, and reserve seats through the internet. Many of these systems focus on the use of relational databases for storing passenger records, schedules, and transactions. They also highlight the importance of secure login, booking confirmation, cancellation handling, and payment gateway integration. While these booking systems significantly improve convenience, many earlier models do not provide real-time seat occupancy visibility or live trip monitoring, which limits their operational effectiveness.

Another major area of study is fleet and route management systems. Several transportation management solutions have been developed to help operators manage bus fleets, assign vehicles to routes, define stoppages, and monitor daily operations. Literature in this domain emphasizes the need for structured data models to manage buses, drivers, routes, seat layouts, and trip schedules efficiently. These systems often include administrative dashboards for transport managers to monitor bookings and trip

performance. However, many route management systems are designed primarily for internal use and may not offer a strong passenger-facing interface. In addition, some systems lack integration between schedule management and live seat booking, resulting in disconnected workflows that reduce efficiency.

Research on real-time bus tracking systems has also contributed significantly to the proposed project. GPS-based tracking has become a popular approach in transportation monitoring, allowing administrators and passengers to view current bus locations and estimated arrival times. Various studies show that real-time tracking improves passenger satisfaction by reducing uncertainty and wait times. Mobile and web-based tracking platforms have been proposed for schools, public transport agencies, and private fleet operators. These systems often rely on GPS devices, mapping APIs, and communication modules to send vehicle position data to a central server. While effective in location monitoring, many tracking systems focus only on vehicle movement and do not integrate booking information, seat occupancy, or ticket issuance into the same platform.

The concept of seat occupancy monitoring is another important research area relevant to this project. In modern transport systems, occupancy tracking helps transport agencies understand demand patterns, avoid overbooking, and improve trip planning. Some studies discuss sensor-based occupancy detection, while others use booking and check-in data to estimate occupied seats. In software-oriented models, seat occupancy is often managed through database-driven booking updates, where each confirmed reservation changes seat status from available to booked. This method is practical for web-based ticketing systems and is especially useful when combined with visual seat layouts. Literature indicates that occupancy data can support analytics, pricing strategies, and better fleet allocation. However, many previous systems treat seat booking as a static process and do not fully connect it with real-time administrative monitoring or passenger communication.

Another relevant field is multi-role transport management software. Research and case studies on enterprise transport solutions show that different stakeholders—administrators, supervisors, counter operators, and drivers—require separate access rights and workflows. Admins may need control over buses, routes, schedules, pricing, and reports. Counter staff may require fast booking and ticket printing for walk-in customers. Drivers may need trip assignments and passenger lists, while supervisors may monitor trip progress and occupancy. Literature supports the use of role-based authentication and dashboard-driven interfaces to ensure security, accountability, and ease of use. Multi-role systems improve organizational efficiency, but earlier implementations often lack a complete integration of all operational levels within one scalable platform.

Studies on payment integration and secure digital transactions are also highly relevant to bus booking platforms. Online ticket reservation systems increasingly depend on secure payment gateways to process bookings instantly. Literature highlights the importance of transaction logging, booking confirmation, refund handling, and fraud prevention. Many modern systems support multiple payment options such as cards, wallets, bank transfers, and mobile payments. Secure transaction frameworks, data encryption, and verification mechanisms are critical components in these systems. However, in some earlier transport projects, payment modules are either simulated or limited to a single payment method, which does not reflect the flexibility needed in large-scale practical deployment.

In recent years, smart transportation systems have moved toward integrated platforms that combine booking, tracking, notifications, analytics, and customer service. This evolution is supported by advancements in web development, cloud hosting, API integration, and real-time communication tools. Literature suggests that a complete transport management solution should not only support ticket booking but also provide passenger alerts, cancellation policies, analytics dashboards, and administrative reports. These features help operators make data-driven decisions while improving passenger trust and convenience. The proposed Bus Tracking and Seat Occupancy Monitoring

System builds upon this modern perspective by combining multiple functional areas into one professional platform.

From the literature surveyed, it is clear that existing systems have addressed individual aspects such as ticket booking, GPS tracking, route management, or payment processing. However, there remains a need for a unified and scalable web-based solution that integrates all these components together. The proposed project fills this gap by offering a complete digital bus management system with live seat status, booking workflows, route and fleet administration, role-based dashboards, and secure passenger services. Therefore, the literature strongly supports the relevance and importance of developing such a system as a practical and advanced final-year software project.

# 3. PROPOSED SYSTEM

The proposed **Bus Tracking and Seat Occupancy Monitoring System** is a complete web-based transport management platform developed to automate bus operations, improve passenger convenience, and provide real-time visibility into fleet movement and seat usage. The system is designed to replace manual reservation, fragmented scheduling, and disconnected reporting methods with a centralized digital solution. It supports end-to-end transport activities including bus registration, route creation, schedule planning, seat layout configuration, passenger booking, payment processing, ticket generation, occupancy monitoring, and trip supervision. By integrating all these features into one platform, the proposed system ensures better efficiency, transparency, and service quality for both transport providers and passengers.

At the core of the system is an intelligent **bus ticket booking module** that allows passengers to search available buses based on source, destination, date, and timing. After selecting a trip, users can view a graphical seat layout that displays real-time seat availability. Booked seats, available seats, and held seats are visually differentiated so that users can make accurate reservations. Once the seat is selected, the passenger can enter travel details, proceed with payment, and receive instant ticket confirmation through the system. This process eliminates manual errors, avoids seat duplication, and significantly reduces the workload at physical booking counters. For transport offices, the system also supports walk-in booking through a dedicated counter panel, ensuring that offline and online reservations are managed in a synchronized manner.

The proposed system introduces a powerful **seat occupancy monitoring mechanism** that continuously updates the seat status for every bus trip. Occupancy is tracked based on confirmed bookings, cancellations, check-in records, and trip completion data. This helps administrators and supervisors understand the current load of each bus in real time. Visual occupancy indicators can be used in dashboards to show total seats, booked seats, vacant seats, reserved blocks, and cancellation counts. This information is highly useful for operational planning because it helps transport managers identify underutilized routes, peak travel periods, and high-demand schedules. Instead of depending on assumptions or manual phone updates, decision-makers can rely on live data to optimize bus assignments and improve seat utilization.

Another important component of the proposed system is **bus tracking and trip supervision**. Each trip can be associated with route information, assigned driver, departure point, intermediate stoppages, and estimated arrival timing. Supervisors and administrators can monitor trip progress using status updates such as scheduled, boarding, departed, en route, delayed, arrived, or completed. When integrated with GPS or location update APIs, the system can also show the current bus location and expected timing for the next stop. Passengers benefit from this transparency because they receive reliable travel updates, while operators benefit from improved monitoring and accountability. Even in cases where full GPS hardware integration is not used, the system can still support manual or API-based trip status updates to simulate live movement tracking in a practical software environment.

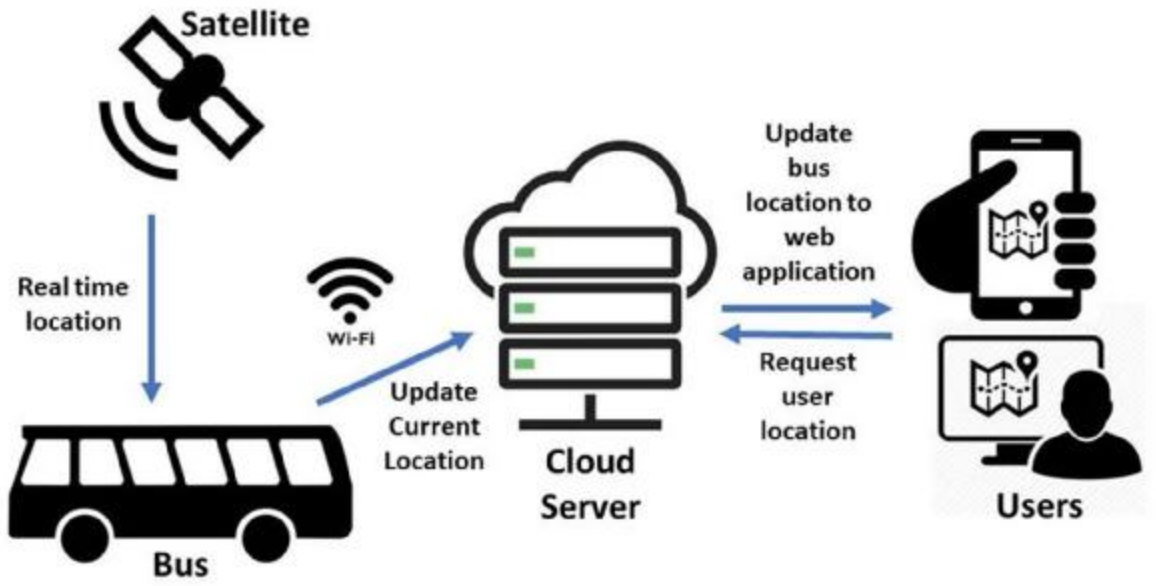
The proposed system is built with a **multi-role architecture** to support different users involved in transport operations. The **admin panel** provides complete control over buses, seat layouts, routes, stoppages, schedules, pricing, users, payment logs, cancellation rules, and reports. The **supervisor panel** focuses on live trip observation, occupancy analysis, and transport coordination. The **counter panel** allows staff to create instant bookings for customers visiting the office directly, print tickets, and handle travel inquiries. The **driver panel** displays assigned trip details, passenger count, boarding data, and status update options. This role-based design ensures data security, proper responsibility distribution, and efficient workflow management. Each user can access only the information and actions relevant to their duties.

To enhance user convenience and business flexibility, the proposed system includes a **secure payment and ticket issuance framework**. The platform can be integrated with multiple payment gateways to support cards, wallets, bank transfers, mobile money, and other regional or international options. Once payment is successful, the system automatically generates a unique booking ID and ticket reference. Tickets can be downloaded, printed, or sent through email and SMS notifications. The system also supports cancellation rules, ticket hold options, refund eligibility settings, and terms and conditions acceptance during booking. These features make the platform suitable for real-world deployment where customer policies and secure financial handling are essential.

The system is also designed with **analytics and reporting capabilities** to support transport business management. Administrative users can view booking trends, route-wise revenue, occupancy percentages, cancellation statistics, passenger history, counter performance, and trip completion summaries. Daily, weekly, and monthly reports help the organization measure service efficiency and identify operational gaps. This analytical capability transforms the platform from a simple booking application into a complete decision-support system. Reports can also help management plan route expansions, revise fares, and improve service during high-demand periods such as weekends, holidays, and festival seasons.

From a technical perspective, the proposed system can be developed using **Core PHP, MySQL, Bootstrap, JavaScript, and API integrations** for payments, notifications, and optional location services. The frontend interface is designed to be responsive and professional, ensuring smooth access across desktops, tablets, and mobile devices. The backend uses a relational database structure to store bus details, schedules, passengers, bookings, users, payments, and trip logs in an organized and scalable way. Security measures such as authentication, role-based access control, input validation, session management, and activity logging are incorporated to protect sensitive data and ensure reliable operation.

## 3.1 ARCHITECTURE




The architecture of the Bus Tracking and Seat Occupancy Monitoring System is designed as a multi-layer web-based architecture that integrates passenger services, transport administration, booking management, trip supervision, payment processing, and reporting into a unified platform. The system follows a structured design so that each component performs a specific function while remaining connected to the central database and control logic. This architecture ensures smooth data flow between users, buses, schedules, bookings, and monitoring modules. It is built to support real-time updates, multiple user roles, and scalable transport operations.

At the top level, the architecture consists of four major layers: the Presentation Layer, Application Layer, Service/Integration Layer, and Database Layer. The Presentation Layer is the user interface through which different actors interact with the system. These actors include passengers, administrators, supervisors, counter staff, and drivers. Passengers use the frontend portal to search routes, check schedules, view seat

availability, make payments, and download tickets. Administrators and staff use secure dashboards with role-based menus to manage buses, routes, schedules, fares, bookings, reports, and trip updates. The interface can be developed using HTML, CSS, Bootstrap, JavaScript, and AJAX to provide a responsive and professional design.

The second layer is the Application Layer, which acts as the core processing unit of the system. This layer handles all business logic related to booking validation, seat allocation, fare calculation, schedule management, cancellation rules, role-based authentication, and status monitoring. When a passenger selects a route and seat, the application layer verifies whether the seat is available, checks the selected trip timing, calculates ticket cost, applies booking rules, and confirms the reservation after successful payment. Similarly, when an admin creates a new route or trip schedule, this layer processes the input, validates the data, and stores it correctly in the system. This layer also ensures that user permissions are enforced, so each role accesses only authorized functions.

The Service and Integration Layer is responsible for handling communication with external or supporting services. This includes payment gateway integration, SMS and email notification services, and optional GPS or live location APIs for bus tracking. When a user completes a payment, this layer communicates with the payment provider and returns the transaction status to the application. If payment is successful, the system automatically issues the ticket and sends a confirmation message. In the case of live bus tracking, location updates from a GPS module or external API are fetched and displayed to supervisors and passengers. This layer makes the system flexible and ready for real-world transport operations where third-party services are necessary.

The Database Layer forms the backbone of the architecture. It stores all important information such as user accounts, bus details, seat layouts, routes, stoppages, schedules, passengers, bookings, payments, ticket records, trip logs, occupancy status, and reports. A relational database such as MySQL is suitable for this project because the system involves structured relationships between multiple entities. For example, one

bus can have one seat layout, one route can contain multiple stoppages, one trip schedule can produce many bookings, and one booking can be linked to one payment and one passenger. Proper normalization of tables helps reduce redundancy and ensures data consistency across modules.

A central architectural concept in this system is the seat occupancy engine. This component continuously updates seat status based on booking transactions, ticket cancellation, counter reservations, and trip completion records. Whenever a booking is made, the related seat is marked as occupied for that particular trip schedule. If a booking is cancelled, the seat becomes available again based on cancellation policy and system rules. This engine is connected directly with the booking module, seat layout module, and schedule module, ensuring that seat availability is always updated accurately in real time. This feature is essential for preventing overbooking and for offering trustworthy information to both passengers and management.

Another important architectural component is the trip tracking and status management module. Each bus trip moves through several operational stages such as scheduled, boarding, departed, en route, delayed, arrived, and completed. These statuses are maintained in the system through driver updates, supervisor inputs, or automated API-based location feeds. The architecture supports live status reflection on both passenger and admin dashboards. This helps passengers know whether the bus is on time and helps transport managers react quickly to delays, route changes, or occupancy issues. Thus, trip tracking acts as a monitoring bridge between operational activity and user communication.

The architecture also includes a role-based access control framework. This framework authenticates every user through secure login and assigns permissions based on role type. The Admin has full system access, including master data and reports. The Supervisor monitors schedules, occupancy, and trip status. The Counter Staff handles offline bookings, passenger service, and ticket printing. The Driver views trip assignments and updates travel status. The Passenger can search, book, cancel, and track

tickets through the frontend portal. This layered access improves security, accountability, and ease of operation.

In terms of system workflow, the architecture begins when the admin configures the master data, including buses, routes, seat structures, stoppages, prices, and schedules. Once these details are published, passengers and counters can make bookings. Booking data updates the database instantly, and the seat occupancy module reflects the latest status. Payment confirmation triggers ticket generation and notification delivery. During journey execution, trip tracking updates are recorded and displayed in dashboards. Finally, analytics and reporting modules collect operational data from all system activities and present them in charts, summaries, and logs for management decisions.

## 3.2 MODULES DESCRIPTION

**1. User Management Module**

The User Management Module handles all user-related activities in the system. It supports multiple roles such as Admin, Supervisor, Counter Staff, Driver, and Passenger. Each user must register and log in securely using authentication mechanisms like session control and input validation. The admin has full authority to create, update, or delete user accounts and assign roles accordingly.

This module ensures **role-based access control**, meaning each user can only access features relevant to their responsibilities. For example, passengers can book tickets, while admins manage system data. It also allows users to update their profiles and change passwords. Additionally, login history and activity logs can be maintained for security and monitoring purposes. This module is essential for maintaining system security, ensuring authorized access, and managing interactions between different stakeholders effectively.

**2. Bus & Fleet Management Module**

This module manages all bus-related information within the system. Admin can add new buses, define bus types (AC, Non-AC, Sleeper), set seating capacity, and configure seat layouts. Each bus is assigned a unique identification number to track operations efficiently.

The module helps in organizing fleet details and ensures buses are properly allocated to routes and schedules. It also allows updating bus availability status and maintaining records for operational use. By managing fleet data effectively, this module supports accurate scheduling and booking processes. It prevents duplication and ensures that buses are assigned without conflicts. Overall, this module plays a crucial role in managing transportation resources efficiently and maintaining structured fleet operations.

**3. Route & Stoppage Management Module**

The Route & Stoppage Management Module allows administrators to define travel routes between different locations. Each route includes a source, destination, and

multiple stoppages with proper sequence order. The module also allows setting estimated arrival and departure times for each stop.

This information is displayed to passengers during booking, helping them understand the journey clearly. It ensures that buses follow structured travel paths and improves route planning. Admin can add, update, or delete routes as required. This module plays a key role in organizing transport operations and ensuring accurate mapping of travel routes within the system.

**4. Schedule & Trip Management Module**

This module is responsible for managing bus schedules and trip planning. Admin can create schedules by assigning buses to routes with specific departure dates and times. Each schedule represents a trip instance.

The module allows updating trip status such as scheduled, boarding, departed, or completed. It ensures that buses are assigned without time conflicts. Supervisors can monitor trip progress, and drivers can view assigned trips. This module helps in maintaining organized transport operations and ensures timely execution of trips.

**5. Seat Layout & Allocation Module**

The Seat Layout & Allocation Module defines the seating arrangement of each bus. It provides a visual representation of seats, allowing passengers to select preferred seats during booking.

The system automatically updates seat status as available, booked, or reserved. This prevents double booking and ensures accurate seat allocation. Admin can design seat layouts based on bus type. This module enhances user experience and ensures efficient seat management.

**6. Ticket Booking & Reservation Module**

This module allows passengers to search for buses, select routes, choose seats, and book tickets. It supports both online booking and counter booking.

Users can enter travel details, confirm bookings, and receive tickets instantly. The system generates a unique booking ID for each reservation. This module ensures smooth and efficient booking operations while reducing manual errors.

**7. Payment & Transaction Module**

The Payment Module handles all financial transactions within the system. It supports integration with multiple payment gateways for secure online payments.
Users can pay using cards, wallets, or bank transfers. The module records transaction details, payment status, and booking references. It also supports refunds and cancellations. This ensures safe and reliable payment processing.

**8. Seat Occupancy Monitoring Module**

This module tracks real-time seat availability for each trip. It updates automatically when bookings are made or cancelled.

Admins and supervisors can view total seats, booked seats, and available seats through dashboards. This helps in improving seat utilization and avoiding overbooking. It provides accurate and live data for decision-making.

**9. Bus Tracking & Trip Status Module**

This module monitors the real-time status of buses. It updates trip stages such as boarding, departed, en route, and arrived.

If integrated with GPS, it can display live bus location. Passengers can track their journey, while admins monitor trip progress. This improves transparency and service reliability.

**10. Reports & Analytics Module**

This module generates reports on bookings, revenue, occupancy, and cancellations. Admin can view daily, weekly, and monthly reports.
It provides graphical insights and summaries for better decision-making. This module helps in analyzing system performance and improving operational efficiency.

# 4. IMPLEMENTATION

The Bus Tracking and Seat Occupancy Monitoring System is implemented as a web-based application using modern yet simple technologies such as Core PHP, MySQL, HTML, CSS, Bootstrap, and JavaScript. The system is designed following a modular architecture where each functional component operates independently but is connected through a centralized database. The main objective of the implementation is to create a user-friendly, secure, and efficient platform that supports real-time booking, seat monitoring, and transport management.

The development begins with setting up the environment using a local server such as XAMPP or WAMP. The backend logic is written in PHP, which handles all server-side operations including data processing, validation, and communication with the database. The frontend interface is developed using HTML and Bootstrap to provide a responsive layout, ensuring compatibility across devices such as desktops, tablets, and mobile phones. JavaScript and AJAX are used to enhance interactivity, especially for dynamic features like seat selection and real-time availability updates.

The core of the system lies in its database design, implemented using MySQL. The database consists of multiple interrelated tables such as Users, Buses, Routes, Stoppages, Schedules, Seat Layout, Bookings, Payments, and Trip Logs. Each table is structured using primary and foreign keys to maintain relationships and ensure data integrity. For example, the booking table is linked to the user, schedule, and payment tables, allowing accurate tracking of each reservation. Proper normalization techniques are applied to reduce redundancy and improve performance.

The implementation process starts with the Admin Panel, where the administrator configures the entire system. The admin can add bus details including bus number, type, and seating capacity. Next, routes and stoppages are defined with proper sequencing and timing. After that, schedules are created by assigning buses to routes along with departure dates and fares. These configurations form the foundation for all booking and tracking operations.

The User Management System is implemented to handle authentication and role-based access control. Users such as Admin, Supervisor, Counter Staff, Driver, and Passenger are assigned specific roles. Login functionality is secured using session management and password hashing techniques. This ensures that only authorized users can access the system, and each user interacts with features relevant to their role.

The Ticket Booking Module is one of the most important parts of the implementation. Passengers can search for buses by selecting source, destination, and travel date. The system retrieves available schedules from the database and displays them in a user-friendly format. Once a schedule is selected, a dynamic seat layout is presented. This layout is developed using HTML and JavaScript, where each seat is visually represented and color-coded based on availability.

When a user selects a seat, the system performs a real-time check using AJAX to ensure that the seat is still available. This prevents conflicts and double booking. After selecting seats, the user enters passenger details and proceeds to the payment stage. The booking details are temporarily stored until the payment is completed successfully.

The Payment Module is implemented using third-party payment gateway APIs such as Razorpay or PayPal. When the user initiates payment, the system securely sends transaction details to the payment gateway. After payment completion, the gateway returns a response indicating success or failure. Only successful transactions are recorded in the database, and the booking is confirmed. A unique ticket ID is generated, and the ticket is displayed on the screen as well as sent via email or SMS.

The Seat Occupancy Monitoring System is implemented by continuously updating seat status in the database. Every booking marks a seat as occupied, and cancellations revert the seat back to available status. Admin and supervisors can view occupancy details such as total seats, booked seats, and available seats for each trip. This data is displayed

using tables and graphical dashboards, helping in better decision-making and resource optimization.

The Bus Tracking and Trip Status Module is implemented to monitor the journey of each bus. Drivers or supervisors can update trip status manually, such as boarding, departed, en route, delayed, and arrived. Optionally, the system can be integrated with GPS APIs to provide real-time location tracking. These updates are reflected instantly on both admin dashboards and passenger interfaces, improving transparency and communication.

The Reports and Analytics Module is implemented using SQL queries and visualization libraries like Chart.js. It generates reports on bookings, revenue, occupancy rates, cancellations, and trip performance. These reports are displayed in graphical formats such as bar charts and pie charts, making it easier for administrators to analyze system performance and make informed decisions.

Security is a critical part of the implementation. The system uses input validation, prepared statements, and session control to prevent common vulnerabilities such as SQL injection and unauthorized access. Passwords are stored securely using hashing techniques, and sensitive operations require proper authentication.

## 4.1 SOFTWARE REQUIEMENTS

**1. Operating System**

- Windows 10 / 11 (for development)
- Linux (Ubuntu/CentOS) recommended for production server

**2. Web Server**

- Apache Server (with mod_rewrite enabled)

  **OR**

- Nginx Server

**3. Backend Technology**

- PHP >= 8.2
- Laravel Framework (Latest Version)

**4. Database**

- MySQL Server (5.7 or higher)

  **OR**

- MariaDB

**5. Frontend Technology**

- HTML5
- CSS3
- JavaScript
- Bootstrap (for responsive UI design)
- Blade Template Engine (Laravel)

**6. Required PHP Extensions**

- PDO PHP Extension
- OpenSSL PHP Extension
- Mbstring PHP Extension
- Exif PHP Extension
- Fileinfo Extension
- XML PHP Extension
- Ctype PHP Extension
- JSON PHP Extension
- Tokenizer PHP Extension

- cURL PHP Extension

**7. Additional Tools & Software**

- Composer (Dependency Manager for PHP)
- Git (Version Control System)
- Node.js & NPM (for frontend asset compilation, optional)

**8. Server Configuration**

- Enable **mod_rewrite** (Apache)
- Enable **HTTPS (SSL Certificate)** for secure communication
- Configure .env file for database and app settings
- Set proper file permissions for storage and cache

**9. Browser Compatibility**

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Safari

**10. Hosting Environment**

- VPS Server / Cloud Hosting (AWS, DigitalOcean, etc.)
- Minimum recommended configuration:
    - 2+ CPU Cores
    - 4 GB RAM
    - 50 GB Storage

## 4.2 SYSTEM REQUIREMENTS

### 1. Hardware Requirements

**Development System Requirements**

These are the minimum hardware requirements for developing and testing the project:

- **Processor:** Intel Core i3 / i5 or higher
- **RAM:** 8 GB minimum
- **Hard Disk:** 256 GB SSD or higher
- **Monitor:** 14-inch or above
- **Keyboard and Mouse:** Standard input devices
- **Internet Connection:** Stable broadband connection

These specifications are sufficient for coding, database handling, local server testing, and UI development.

**Server Requirements (VPS Hosting)**

For live deployment, the application requires a **Virtual Private Server (VPS)** with the following configuration:

| Component | Specification |
|---|---|
| **Server Type** | Virtual Private Server (VPS) |
| **CPU** | 8 Core Processor |
| **RAM** | 32 GB |
| **Storage** | 300 GB NVMe SSD |
| **Bandwidth** | High-speed / Unlimited preferred |
| **Operating System** | Ubuntu / CentOS / AlmaLinux |
| **Web Server** | Apache or Nginx |
| **Database Server** | MariaDB |
| **Control Panel** | WHM / cPanel |
| **Backup Support** | Daily / Weekly Backup Recommended |
| **SSL Certificate** | Required for secure access |

## 4.3 SAMPLE CODING

Login

```
@php
    $content = getContent('contact.content', true);
    $language = App\Models\Language::all();
    $selectedLang = $language->where('code', session('lang'))->first();
    $pages = App\Models\Page::where('tempname', $activeTemplate)
        ->where('is_default', Status::NO)
        ->get();
@endphp
<!-- Header Section Starts Here -->

<div class="header-top">
    <div class="container">
        <div class="header-top-area">
            <ul class="left-content">
                <li>
                    <i class="las la-phone"></i>
                    <a href="tel:{{ __(@$content->data_values->contact_number) }}">
                        {{ __(@$content->data_values->contact_number) }}
                    </a>
                </li>
                <li>
                    <i class="las la-envelope-open"></i>
                    <a href="mailto:{{ __(@$content->data_values->email) }}">
                        {{ __(@$content->data_values->email) }}
                    </a>
                </li>
            </ul>

            <div class="right-content d-flex flex-wrap" style="gap:10px">
```

```
@if (gs('multi_language'))
  <div>
    <div class="language dropdown">
      <button class="language-wrapper" data-bs-toggle="dropdown" aria-expanded="false">
        <div class="language-content">
          <div class="language_flag">
            <img src="{{ getImage(getFilePath('language') . '/' . @$selectedLang->image, getFileSize('language')) }}" alt="flag">
          </div>
          <p class="language_text_select">{{ __(@$selectedLang->name) }}</p>
        </div>
        <span class="collapse-icon"><i class="las la-angle-down"></i></span>
      </button>
      <div class="dropdown-menu langList_dropdow py-2">
        <ul class="langList">
          @foreach ($language as $item)
            <li class="language-list langSel" data-code="{{ $item->code }}">
              <div class="language_flag">
                <img src="{{ getImage(getFilePath('language') . '/' . $item->image, getFileSize('language')) }}" alt="flag">
              </div>
              <p class="language_text">{{ $item->name }}</p>
            </li>
          @endforeach
        </ul>
      </div>
    </div>
```

```
                </div>
            @endif
            @guest
                <ul class="header-login">
                    <li><a class="sign-in" href="{{ route('user.login') }}"><i class="fas fa-sign-in-alt"></i>@lang('Sign In')</a></li>
                    <li>/</li>
                    <li><a class="sign-up" href="{{ route('user.register') }}"><i class="fas fa-user-plus"></i>@lang('Sign Up')</a></li>
                </ul>
            @endguest
            @auth
                <ul class="header-login">
                    <li>
                        <a href="{{ route('user.home') }}">@lang('Dashboard')</a>
                    </li>
                </ul>
            @endauth
        </div>
    </div>
</div>
<div class="header-bottom">
    <div class="container">
        <div class="header-bottom-area">
            <div class="logo">
                <a href="{{ route('home') }}">
                    <img src="{{ siteLogo() }}" alt="@lang('Logo')">
                </a>
            </div> <!-- Logo End -->
            <ul class="menu">
```

```
            <li>
                <a href="{{ route('home') }}">@lang('Home')</a>
            </li>
            @foreach ($pages as $k => $data)
                <li>
                    <a href="{{ route('pages', [$data->slug]) }}">{{ __($data->name) }}</a>
                </li>
            @endforeach

            <li>
                <a href="{{ route('blog') }}">@lang('Blog')</a>
            </li>
            <li>
                <a href="{{ route('contact') }}">@lang('Contact')</a>
            </li>
        </ul>
        <div class="d-flex flex-wrap algin-items-center">
            <a href="{{ route('ticket') }}" class="cmn--btn btn--sm">@lang('Buy Tickets')</a>
            <div class="header-trigger-wrapper d-flex d-lg-none ms-4">
                <div class="header-trigger d-block d-lg-none">
                    <span></span>
                </div>
                <div class="top-bar-trigger">
                    <i class="las la-ellipsis-v"></i>
                </div>
            </div><!-- Trigger End-->
        </div>
    </div>
</div>
```

```
</div>

@push('style')
    <style>
        .language-wrapper {
            display: flex;
            align-items: center;
            justify-content: space-between;
            gap: 12px;
            width: max-content;
            margin-left: 12px;
            padding: 0;
            background-color: transparent;
            border: 0;
        }

        .language_flag {
            flex-shrink: 0;
            display: flex;
        }

        .language_flag img {
            height: 20px;
            width: 20px;
            object-fit: cover;
            border-radius: 50%;
        }

        .language-wrapper.show .collapse-icon {
            transform: rotate(180deg)
        }
```

```
.collapse-icon {
    font-size: 14px;
    display: flex;
    transition: all linear 0.2s;
    color: #111
}

.language_text_select {
    font-size: 14px;
    font-weight: 400;
    color: #111;
}

.language-content {
    display: flex;
    align-items: center;
    gap: 6px;
}

.language_text {
    color: #111
}

.language-list {
    display: flex;
    align-items: center;
    gap: 6px;
    padding: 6px 12px;
    cursor: pointer;
}
```

```
        .language-list:hover {
            background-color: rgba(0, 0, 0, 0.04);
        }

        .language .dropdown-menu {
            position: absolute;
            opacity: 0;
            visibility: hidden;
            top: 100%;
            display: unset;
            background: #ffffffea;
            box-shadow: 0px 0px 4px 0px rgba(0, 0, 0, 0.04), 0px 8px 16px 0px rgba(0, 0,
0, 0.08);
            min-width: 150px;
            padding: 7px 0 !important;
            border-radius: 8px;
            border: 1px solid rgb(255 255 255 / 10%);
        }

        .language .dropdown-menu.show {
            visibility: visible;
            opacity: 1;
        }
    </style>
@endpush

<!-- Header Section Ends Here -->

@push('script')
    <script>
```

```
        $(document).ready(function() {
            "use strict";
            $(".langSel").on("click", function() {
                window.location.href = "{{ route('home') }}/change/" + $(this).data('code');
            });
        });
    </script>
@endpush
```

**Header**

```
@php
    $content = getContent('contact.content', true);
    $language = App\Models\Language::all();
    $selectedLang = $language->where('code', session('lang'))->first();
@endphp
<!-- Header Section Starts Here -->
<div class="header-top">
    <div class="container">
        <div class="header-top-area">
            <ul class="left-content">
                <li>
                    <i class="las la-phone"></i>
                    <a href="tel:{{ __(@$content->data_values->contact_number) }}">
                        {{ __(@$content->data_values->contact_number) }}
                    </a>
                </li>
                <li>
                    <i class="las la-envelope-open"></i>
                    <a href="mailto:{{ __(@$content->data_values->email) }}">
                        {{ __(@$content->data_values->email) }}
                    </a>
```

```
            </li>
        </ul>
        <div class="right-content">
            <div>
                @if (gs('multi_language'))
                    <div>
                        <div class="language dropdown">
                            <button class="language-wrapper" data-bs-toggle="dropdown" aria-expanded="false">
                                <div class="language-content">
                                    <div class="language_flag">
                                        <img src="{{ getImage(getFilePath('language') . '/' . @$selectedLang->image, getFileSize('language')) }}" alt="flag">
                                    </div>
                                    <p class="language_text_select">{{ __(@$selectedLang->name) }}</p>
                                </div>
                                <span class="collapse-icon"><i class="las la-angle-down"></i></span>
                            </button>
                            <div class="dropdown-menu langList_dropdow py-2">
                                <ul class="langList">
                                    @foreach ($language as $item)
                                        <li class="language-list langSel" data-code="{{ $item->code }}">
                                            <div class="language_flag">
                                                <img src="{{ getImage(getFilePath('language') . '/' . $item->image, getFileSize('language')) }}" alt="flag">
                                            </div>
                                            <p class="language_text">{{ $item->name }}</p>
                                        </li>
```

```
                        @endforeach
                    </ul>
                </div>
            </div>
        </div>
    @endif
</div>
</div>
</div>
</div>
<div class="header-bottom">
    <div class="container">
        <div class="header-bottom-area">
            <div class="logo">
                <a href="{{ route('home') }}">
                    <img src="{{ siteLogo() }}" alt="@lang('Logo')">
                </a>
            </div> <!-- Logo End -->
            <ul class="menu">
                <li>
                    <a href="{{ route('user.home') }}">@lang('Dashboard')</a>
                </li>
                <li>
                    <a href="javascript::void()">@lang('Booking')</a>
                    <ul class="sub-menu">
                        <li>
                            <a href="{{ route('ticket') }}">@lang('Buy Ticket')</a>
                        </li>
                        <li>
```

```
                    <a href="{{ route('user.ticket.history') }}">@lang('Booking History')</a>
                </li>
            </ul>
        </li>
        <li>
            <a href="javascript::void()">@lang('Support Ticket')</a>
            <ul class="sub-menu">
                <li>
                    <a href="{{ route('ticket.open') }}">@lang('Create New')</a>
                </li>
                <li>
                    <a href="{{ route('ticket.index') }}">@lang('Tickets')</a>
                </li>
            </ul>
        </li>
        <li>
            <a href="#0">@lang('Profile')</a>
            <ul class="sub-menu">
                <li>
                    <a href="{{ route('user.profile.setting') }}">@lang('Profile')</a>
                </li>
                <li>
                    <a href="{{ route('user.change.password') }}">@lang('Change Password')</a>
                </li>
                <li>
                    <a href="{{ route('user.logout') }}">@lang('Logout')</a>
                </li>
            </ul>
        </li>
```

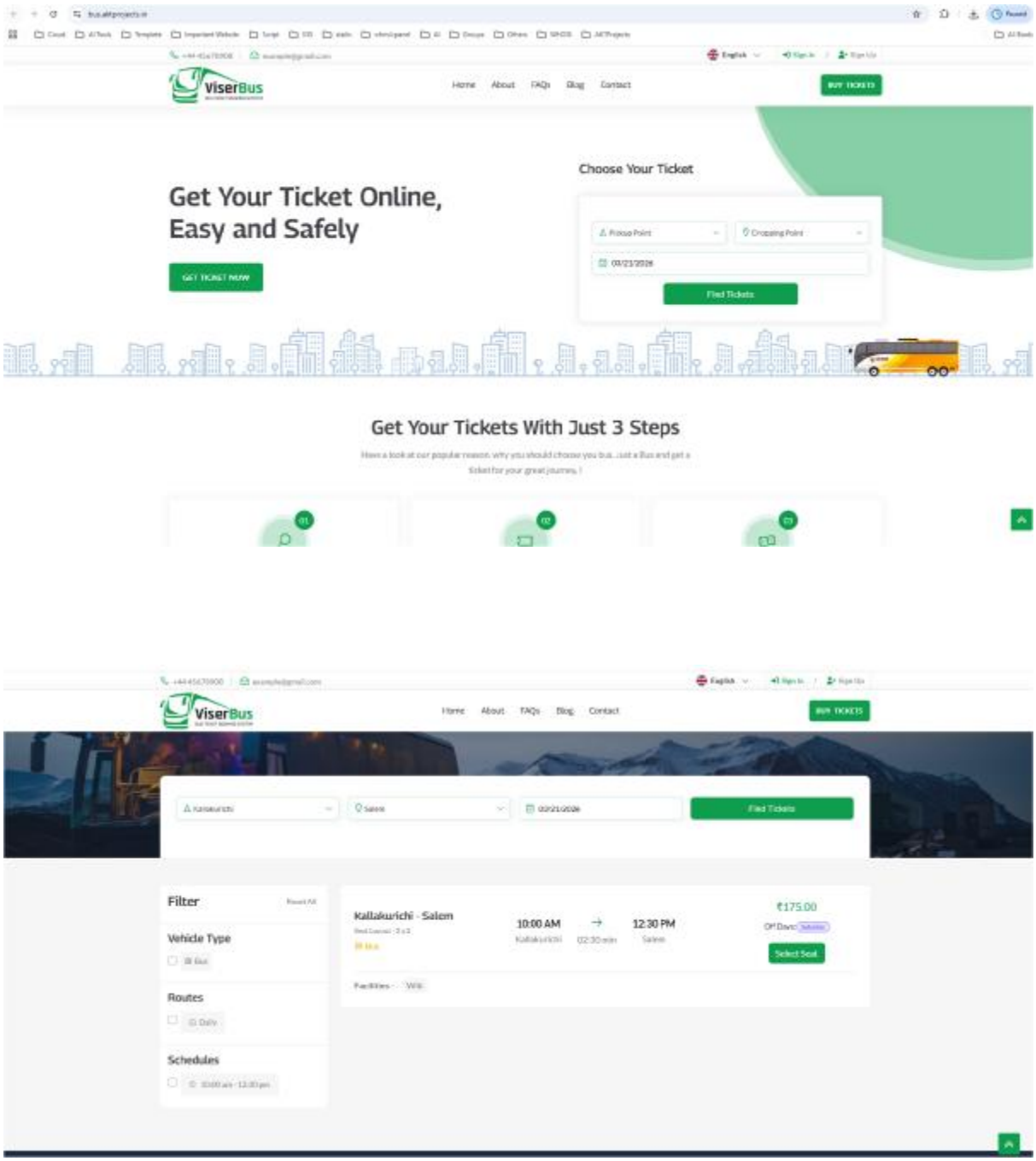
```
            </ul>
            <div class="d-flex flex-wrap algin-items-center">
                <a href="{{ route('ticket') }}" class="cmn--btn btn--sm">@lang('Buy Tickets')</a>
                <div class="header-trigger-wrapper d-flex d-lg-none ms-4">
                    <div class="header-trigger d-block d-lg-none">
                        <span></span>
                    </div>
                    <div class="top-bar-trigger">
                        <i class="las la-ellipsis-v"></i>
                    </div>
                </div><!-- Trigger End-->
            </div>
        </div>
    </div>
</div>
<!-- Header Section Ends Here -->

@push('style')
    <style>
        .language-wrapper {
            display: flex;
            align-items: center;
            justify-content: space-between;
            gap: 12px;
            width: max-content;
            margin-left: 12px;
            padding: 0;
            background-color: transparent;
            border: 0;
        }
```

```
.language_flag {
    flex-shrink: 0;
    display: flex;
}

.language_flag img {
    height: 20px;
    width: 20px;
    object-fit: cover;
    border-radius: 50%;
}

.language-wrapper.show .collapse-icon {
    transform: rotate(180deg)
}

.collapse-icon {
    font-size: 14px;
    display: flex;
    transition: all linear 0.2s;
    color: #111
}

.language_text_select {
    font-size: 14px;
    font-weight: 400;
    color: #111;
}

.language-content {
```

```
        display: flex;
        align-items: center;
        gap: 6px;
    }

    .language_text {
        color: #111
    }

    .language-list {
        display: flex;
        align-items: center;
        gap: 6px;
        padding: 6px 12px;
        cursor: pointer;
    }

    .language-list:hover {
        background-color: rgba(0, 0, 0, 0.04);
    }

    .language .dropdown-menu {
        position: absolute;
        opacity: 0;
        visibility: hidden;
        top: 100%;
        display: unset;
        background: #ffffffea;
        box-shadow: 0px 0px 4px 0px rgba(0, 0, 0, 0.04), 0px 8px 16px 0px rgba(0, 0,
0, 0.08);
        min-width: 150px;
```

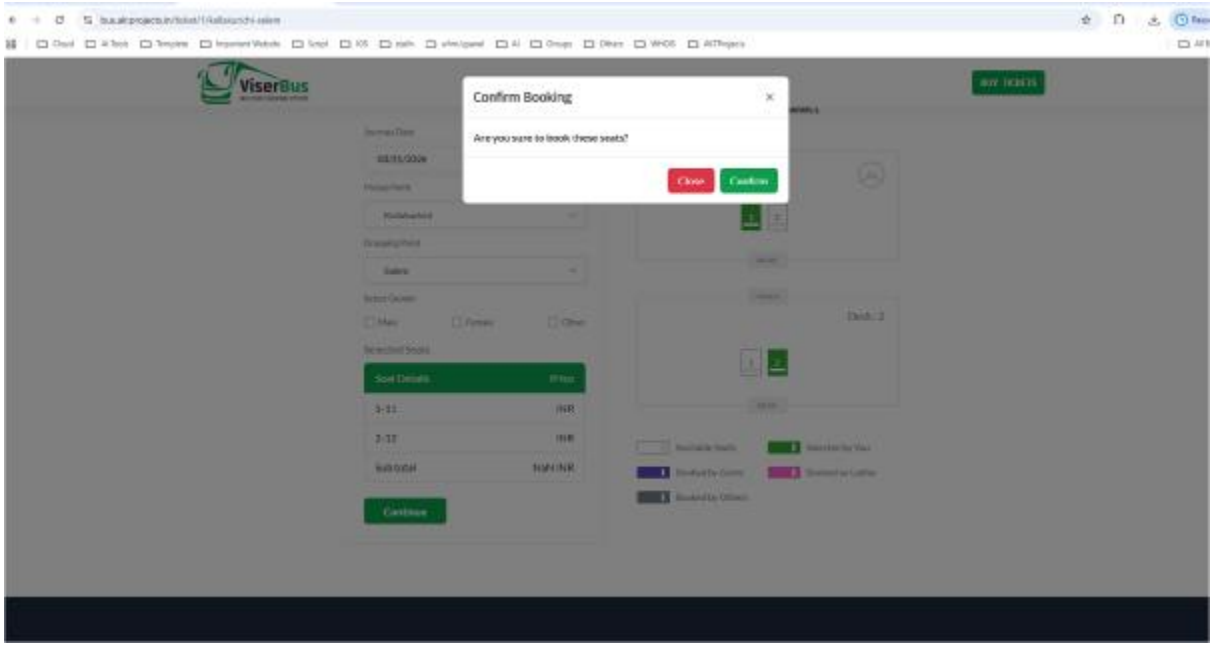
```
            padding: 7px 0 !important;
            border-radius: 8px;
            border: 1px solid rgb(255 255 255 / 10%);
        }

        .language .dropdown-menu.show {
            visibility: visible;
            opacity: 1;
        }
    </style>
@endpush

@push('script')
    <script>
        $(document).ready(function() {
            "use strict";
            $(".langSel").on("click", function() {
                window.location.href = "{{ route('home') }}/change/" + $(this).data('code');
            });
        });
    </script>
@endpush
```

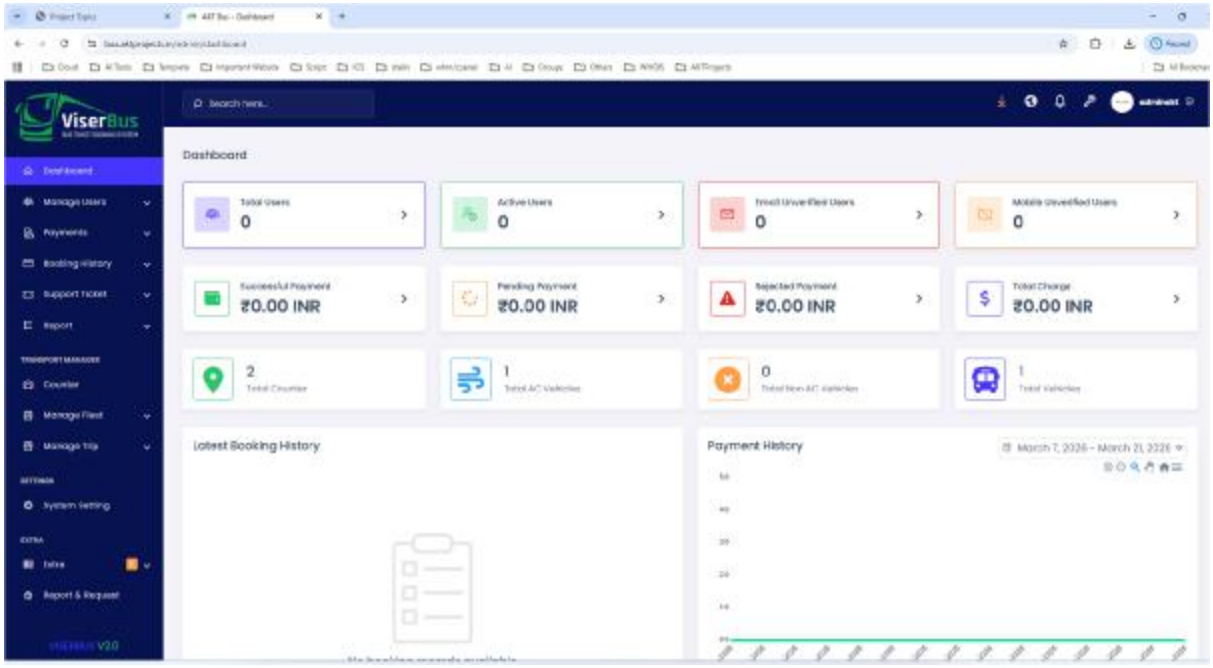
# 4.4. SCREEN SHOT

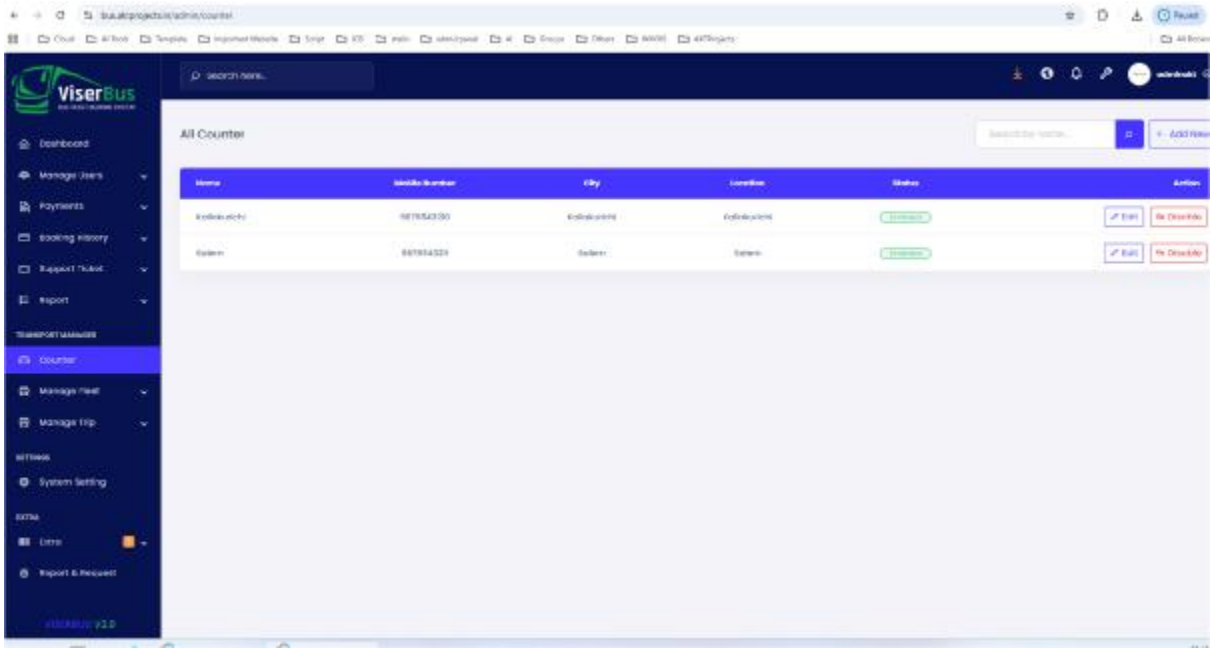


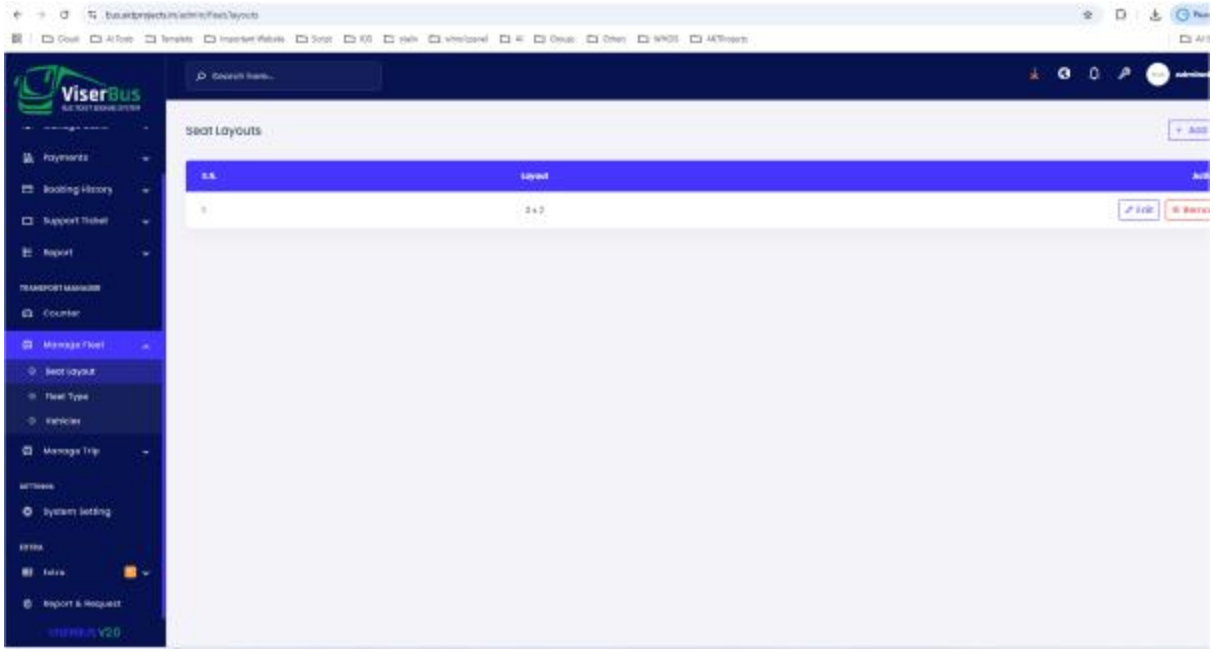

ViserBus
Home
About
FAQs
Blog
Contact
BUY TICKETS
Kallakurichi - Salem

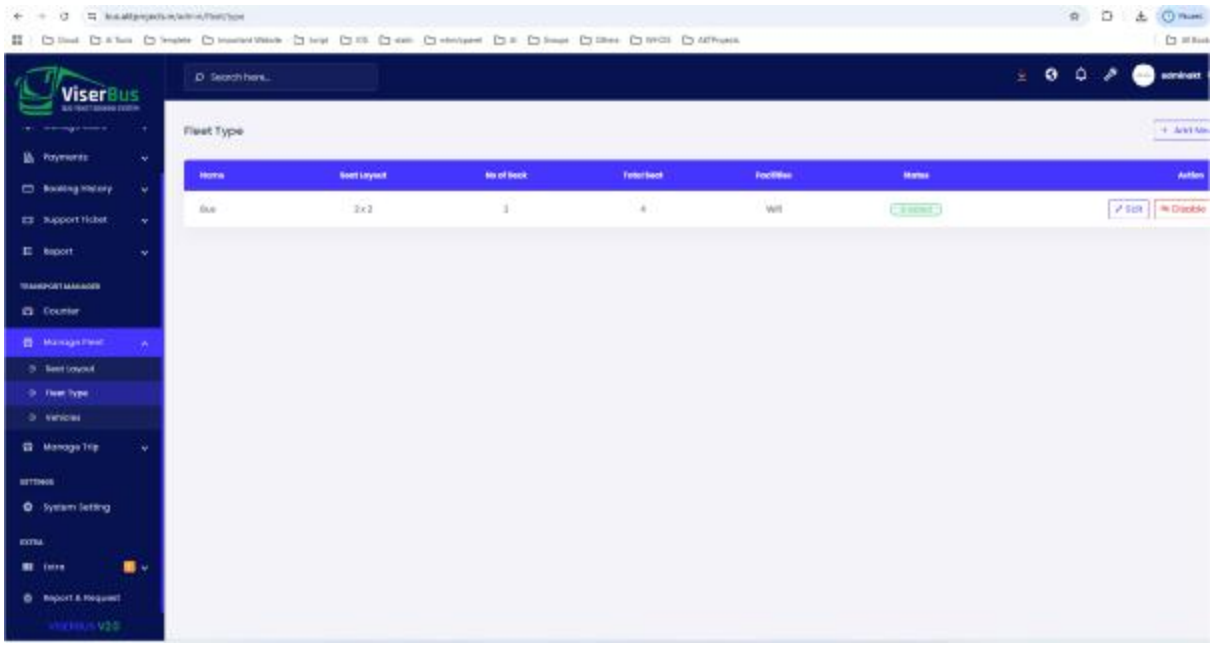

Click on Seat to select or deselect
DOOR
Continue

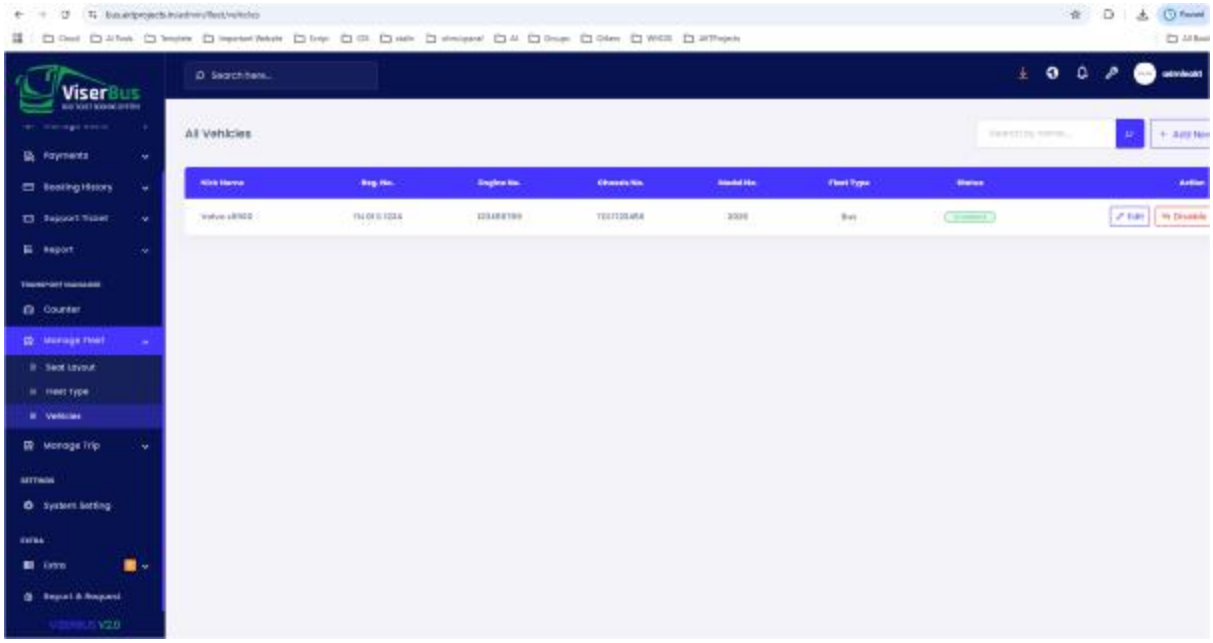

ViserBus
BUY TICKETS
Confirm Booking
Are you sure to book these seats?
Close
Confirm
Continue

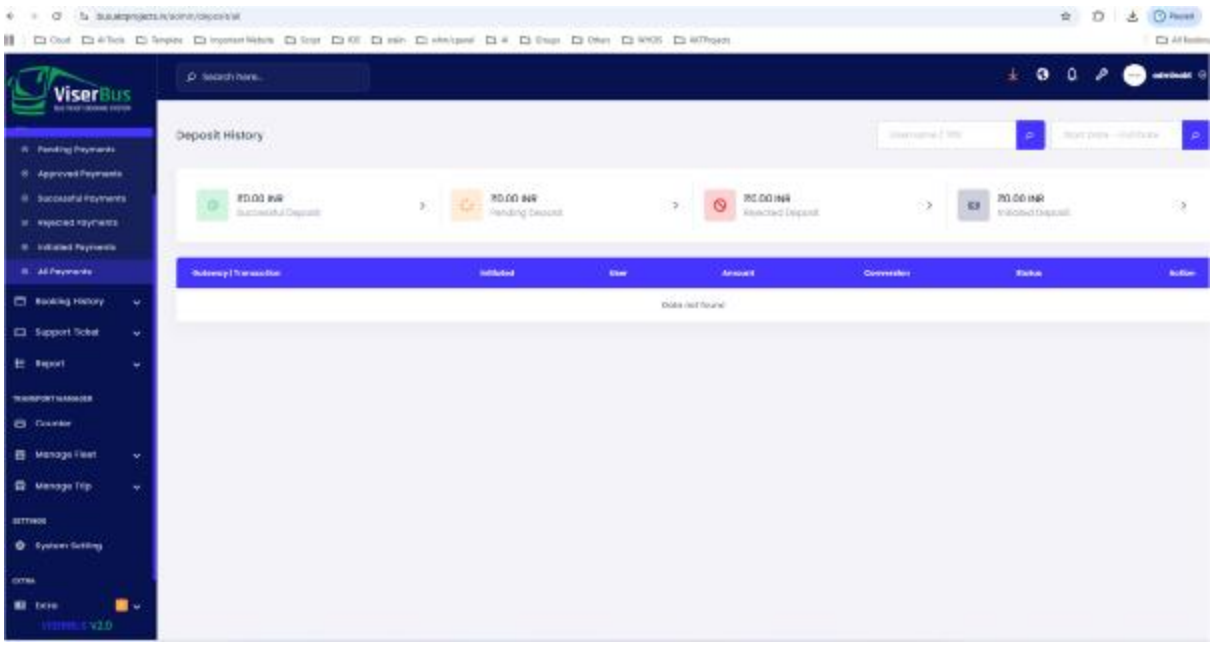

ViserBus
Dashboard
Total Users
0
Active Users
0
Email Unverified Users
0
Mobile Unverified Users
0
Successful Payment
₹0.00 INR
Pending Payment
₹0.00 INR
Rejected Payment
₹0.00 INR
Total Charge
₹0.00 INR
2
1
0
1
Latest Booking History
Payment History


ViserBus
All Counter


ViserBus
Seat Layouts
Payments
Booking History
Support Ticket
Report
Counter
Manage Fleet
Seat Layout
Fleet Type
Vehicles
Manage Trip
System Setting
Report & Request


ViserBus
Fleet Type
Payments
Booking History
Support Ticket
Report
Counter
Manage Fleet
Seat Layout
Fleet Type
Vehicles
Manage Trip
System Setting
Report & Request


ViserBus
All Vehicles
Payments
Booking History
Support Ticket
Report
Counter
Manage Fleet
Seat Layout
Fleet Type
Vehicles
Manage Trip
System Setting
Extra
Report & Request


ViserBus
Deposit History
Pending Payments
Approved Payments
Successful Payments
Rejected Payments
Initiated Payments
All Payments
Booking History
Support Ticket
Report
Counter
Manage Fleet
Manage Trip
System Setting
Extra

# 5. CONCLUSION

The Bus Tracking and Seat Occupancy Monitoring System successfully demonstrates the development of a modern, efficient, and user-friendly transportation management platform. The system integrates multiple functionalities such as online ticket booking, real-time seat availability tracking, trip monitoring, payment processing, and administrative control into a single unified application. By replacing traditional manual processes with an automated digital solution, the system significantly reduces human errors, improves operational efficiency, and enhances the overall travel experience for passengers.

One of the key achievements of this project is the implementation of real-time seat occupancy monitoring, which ensures accurate seat allocation and prevents overbooking. Passengers can easily view available seats and make informed booking decisions, while administrators can monitor seat utilization and optimize resource allocation. The inclusion of multi-role access control allows different users such as admin, supervisors, counter staff, and drivers to perform their respective tasks efficiently within a secure environment. This structured workflow improves coordination and accountability in transport operations.

The system also enhances passenger convenience through features like instant ticket generation, secure online payments, booking history, and automated notifications. These features provide a seamless and reliable booking experience. Additionally, the integration of trip status updates and optional real-time tracking increases transparency, allowing passengers to stay informed about their journey. From an administrative perspective, the availability of reports and analytics helps in better decision-making, route planning, and performance evaluation.

Technically, the project demonstrates effective use of web development technologies such as PHP, MySQL, Bootstrap, and JavaScript, along with API integration for payments and notifications. The modular design ensures scalability, making it possible

to extend the system with additional features like mobile applications, AI-based demand prediction, or advanced GPS tracking in the future.

# REFERENCES


1. Kurose, J. F., & Ross, K. W., Computer Networking: A Top-Down Approach, Pearson Education, 7th Edition, 2017.
2. Elmasri, R., & Navathe, S. B., Fundamentals of Database Systems, Pearson, 7th Edition, 2016.
3. Pressman, R. S., Software Engineering: A Practitioner's Approach, McGraw-Hill, 8th Edition, 2014.
4. Duckett, J., PHP and MySQL: Server-Side Web Development, Wiley Publications, 2014.
5. Nixon, R., Learning PHP, MySQL & JavaScript, O'Reilly Media, 5th Edition, 2018.
6. MDN Web Docs, HTML, CSS, and JavaScript Documentation, Available: https://developer.mozilla.org
7. PHP Official Documentation, PHP Manual, Available: https://www.php.net/manual/en/
8. MySQL Official Documentation, MySQL Reference Manual, Available: https://dev.mysql.com/doc/
9. Bootstrap Documentation, Bootstrap Framework Guide, Available: https://getbootstrap.com
10. W3Schools, Web Development Tutorials, Available: https://www.w3schools.com
11. Razorpay Documentation, Payment Gateway Integration Guide, Available: https://razorpay.com/docs/
12. PayPal Developer Docs, Online Payment Integration, Available: https://developer.paypal.com
13. Google Maps Platform, Maps and Location API Documentation, Available: https://developers.google.com/maps
14. Chart.js Documentation, JavaScript Chart Library, Available: https://www.chartjs.org
15. Sommerville, I., Software Engineering, Pearson, 10th Edition, 2015.