“E-Pass Management system”

A Project report submitted in

partial fulfillment of the requirements

for the Degree of

Bachelor of Technology

in

Computer Engineering

Submitted by

Shivani Patil

Vaishnavi Bhadane

Vaibhav Desale

Raj Ahire

DEPARTMENT OF COMPUTER ENGINEERING
S.S.V.P.S.’s B.S. DEORE COLLEGE OF ENGINEERING, DHULE
2023-24

"E-Pass Management system"

A Project report submitted in

partial fulfillment of the requirements

for the Degree of

Bachelor of Technology

in

Computer Engineering

Submitted by

Shivani Patil

Vaishnavi Bhadane

Vaibhav Desale

Raj Ahire

Guided by

Prof. Dr. I. S. Borse

DEPARTMENT OF COMPUTER ENGINEERING
S.S.V.P.S.'s B.S. DEORE COLLEGE OF ENGINEERING, DHULE
2023-24

S.S.V.P.S.'s B.S. DEORE COLLEGE OF ENGINEERING, DHULE

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project entitled "E-Pass Management System" has been carried out by

Shivani Patil

Vaishnavi Bhadane

Vaibhav Desale

Raj Ahire

under my guidance in partial fulfillment of the degree of *Bachelor of Technology* in *Computer Engineering* of *Dr. Babasaheb Ambedkar Technological University, Lonere* during the academic year *2023-24. To the best of my knowledge and belief this work has not been submitted elsewhere for the award of any other degree.*

Date:

Place: Dhule

Guide

Prof. Dr. I. S. Borse

Head

Prof. Dr. B.R. Mandre

Principal

Prof. Dr. Hitendra D. Patil

ACKNOWLEDGEMENT

First of all, we are grateful to the SSVPS’s B. S. Deore for giving us an opportunity to excel in our efforts to complete this project on time.

We are extremely grateful to Dr. Hitendra Patil, Principal, SSVPS’s B. S. Deore Engineering College of Dhule and Prof. Dr. B. R. Mandre Head of Department, Department of Computer Engineering, for providing all the required resources for the successful completion of our project.

Our gratitude to our project guide Prof. I. S. Borse for his valuable suggestions and guidance in the preparation of the project report.

We will be falling in duty if We do not acknowledge with grateful thanks to the authors of the references and other literatures referred to in this project.

Shivani Patil

Vaishnavi Bhadane

Vaibhav Desale

Raj Ahire

ABBREVIATIONS

Abbreviations	**Details**
RPMS	Rail Pass Management System
CRM	Customer Relationship Management
IT	Information Technology
API	Application Programming Interface
UI/UX	User Interface/User Experience
HDD	Hard Disk Drive
SSD	Solid State Drive
RAM	Random Access Memory
DFD	Data flow diagram
ERD	Entity relationship diagram
UML	Unified relationship diagram

Table of Contents

Figure Index

Table Index

ABSTRACT

The Rail Pass Management System proposed here is indispensable for modern rail operations, providing a sophisticated online platform for the efficient management of pass sales, inventory, and utilization. Through features like online pass purchase and renewal, activation, usage tracking, validation, and reporting, it offers seamless and convenient experiences for both customers and rail companies. Integration with existing ticketing and reservation systems ensures operational synergy and enhances overall service quality. Moreover, the inclusion of customer service portals facilitates effective support and issue resolution. This system not only streamlines pass purchasing processes but also enhances accuracy in pass usage, optimizes inventory management, enables data-driven decision-making, and ultimately elevates customer satisfaction to new heights.

Key Words: (CRM) Customer Relationship Management, (CSP) Customer service portal

Chapter 1

INTRODUCTION

1.1 BACKGROUND

A Rail Pass Management System is a software application used to manage rail pass sales, inventory, and usage. The system provides a platform for customers to purchase and manage their rail passes, while also enabling rail companies to manage the supply and demand of passes across various routes and time periods. The system typically includes features such as pass purchase and renewal, pass activation, usage tracking, pass validation, and reporting. It may also include integration with other systems such as ticketing and reservation systems, as well as customer relationship management (CRM) systems. The benefits of implementing a Rail Pass Management System include improved customer experience through streamlined pass purchase and usage, better inventory management leading to reduced waste and increased revenue, and improved data analysis leading to better decision-making for the rail company. Overall, a Rail Pass Management System can help to optimize rail pass sales and usage, leading to increased efficiency, revenue, and customer satisfactions.

1.2 MOTIVATION

1. **Focus on Efficiency & Cost Savings:**
    a. Streamline Rail Pass Operations: This report explores the development of a Rail Pass Management System (RPMS) designed to significantly improve efficiency and reduce administrative costs associated with rail pass sales, inventory, and usage."
2. **Focus on Customer Experience:**
    a. Enhancing the Passenger Journey: This report details the creation of a Rail Pass Management System aimed at transforming the passenger experience by offering a convenient and user-friendly platform for purchasing, managing, and utilizing rail passes."
3. **Focus on Revenue Management:**
    a. Optimizing Rail Pass Revenue: This report investigates the development of a Rail Pass Management System to empower rail companies with data-driven insights for maximizing revenue by effectively managing pass supply and demand across diverse routes and timeframes."
4. **Focus on Overall Benefits:**

a. Revolutionizing Rail Pass Management: This report presents a comprehensive overview of the Rail Pass Management System, highlighting its potential to revolutionize the industry by streamlining operations, enhancing customer experience, and optimizing revenue management for rail companies."

1.3 PROBLEM DEFINITION

The current system for managing rail passes often suffers from several limitations, creating inefficiencies for both rail companies and passengers. Here's a closer look at the key problems:

- **Manual Processes:** Many rail companies rely on manual processes for pass sales, renewals, and usage tracking. This can lead to errors, delays, and difficulties in generating accurate reports.
- **Limited Customer Convenience:** Traditional methods often require passengers to visit physical offices or contact customer service for pass purchases, renewals, or inquiries. This can be inconvenient and time-consuming.
- **Inventory Management Challenges:** Manual systems make it difficult to track pass inventory in real-time, leading to potential overselling or understocking of specific passes.
- **Data Visibility Issues:** Lack of a centralized system makes it challenging to collect and analyze data on pass usage patterns and customer preferences. This hinders effective revenue management strategies.
- **Inflexible Pass Options:** inflexibility in current systems may limit the creation of targeted pass options catering to specific needs (e.g., frequent traveler passes, weekend passes).
- **Integration Issues:** Disparate systems for ticketing, reservations, and CRM can create data silos, hindering a holistic view of passenger activity.
- **Fraudulent Pass Use:** Manual validation methods can be susceptible to fraudulent pass use, leading to revenue loss for rail companies.
- **Limited Reporting Capabilities:** Traditional systems may lack the capacity to generate comprehensive reports on sales trends, usage patterns, and customer demographics. This limits decision-making capabilities.

Improved Efficiency:

1. Automated Processes: RPMS automates pass sales, renewals, activation, usage tracking, and validation, eliminating manual errors, streamlining workflows, and facilitating faster processing times.
2. Real-time Reporting: The system generates real-time reports on sales, inventory, and usage, enabling accurate data analysis for informed decision-making.
3. Enhanced Customer Convenience:
4. Online Platform: RPMS provides a user-friendly online platform for passengers to conveniently purchase, renew, manage, and activate passes 24/7, eliminating the need for physical visits.
5. Optimized Inventory Management:
6. Real-time Inventory Tracking: The system tracks pass inventory in real-time, preventing overselling and ensuring availability of desired passes.
7. Dynamic Inventory Management: (Optional) Advanced features can predict demand fluctuations and suggest dynamic pricing strategies to optimize pass availability and revenue.
8. Data-Driven Decision Making:
9. Centralized Data Repository: RPMS acts as a central repository for all pass-related data, facilitating easy access and comprehensive analysis of usage patterns, customer preferences, and sales trends.
10. Data Analytics: (Optional) Integrate advanced data analytics tools to generate insights for creating targeted marketing campaigns, optimizing pricing strategies, and fostering customer loyalty.

Flexible Pass Options:

1. Customization Tools: The system allows rail companies to create diverse pass options (e.g., frequent traveler passes, weekend passes, regional passes) catering to specific passenger needs and travel patterns.
2. Seamless Integration:
3. Integration with Existing Systems: RPMS integrates seamlessly with ticketing, reservation, and CRM systems, creating a unified platform for managing passenger activity and fostering data exchange.
4. Reduced Fraudulent Pass Use:

5. Secure Validation Methods: RPMS employs secure digital validation methods to minimize fraudulent pass use and protect revenue.
6. Verification Tools: (Optional) Implement advanced verification features like biometric authentication or QR code scanning for enhanced security.

1.4 OBJECTIVES AND SCOPE

Objectives:

1. The Rail Pass Management System (RPMS) aims to achieve the following key objectives:
2. Streamline Rail Pass Operations: Automate manual processes for pass sales, renewals, activation, usage tracking, and validation to improve efficiency and reduce administrative burden.
3. Enhance Customer Experience: Provide a user-friendly platform for passengers to conveniently purchase, manage, and activate passes, fostering a more enjoyable travel experience.
4. Optimize Revenue Management: Equip rail companies with real-time data on pass inventory, sales trends, and usage patterns to make informed decisions regarding pricing, pass options, and marketing strategies to maximize revenue.
5. Improve Data Visibility: Establish a centralized repository for all pass-related data, enabling in-depth analysis of customer behavior and travel patterns to support data-driven decision-making.
6. Reduce Fraudulent Pass Use: Implement secure validation methods to minimize pass misuse and protect revenue streams.

Scope:

The initial scope of the RPMS project will focus on core functionalities encompassing:

1. Passenger Module: Online platform for pass purchase, renewal, management, and activation.
2. Administration Module: Tools for managing pass types, inventory, pricing, reports, and user access control.
3. Real-time Reporting: Generation of reports on sales, inventory, and pass usage.
4. Integration: (Optional, based on project priorities) Seamless integration with existing ticketing, reservation, and CRM systems.

Future Enhancements:

1. Mobile App Development: Create a mobile app for on-the-go access to pass management features.
2. Advanced Analytics: Integrate data analytics tools to generate deeper customer insights and optimize pricing strategies.
3. Dynamic Inventory Management: Implement features for dynamic pricing based on demand fluctuations to maximize revenue.
4. Advanced Security Features: Incorporate biometric authentication or QR code scanning for enhanced pass validation.

Chapter 2

LITERATURE SURVEY

R. Khan et.al.[1]

In alignment with the outlined introduction on Rail Pass Management Systems, the existing literature presents a comprehensive view of how these systems have become pivotal in reshaping the landscape of railway operations and customer interactions. The surveyed literature underscores several key facets, beginning with the core functionalities that constitute the backbone of Rail Pass Management Systems. Pass purchase and renewal, activation, usage tracking, validation, and reporting are identified as essential features that collectively contribute to creating a user-friendly platform for both customers and rail companies.

M. A. S. M. Chowdhury et.al[2]

Integration emerges as a recurrent theme throughout the literature survey, with a focus on how Rail Pass Management Systems seamlessly connect with existing ticketing and reservation systems, as well as customer relationship management (CRM) platforms. This interconnectedness is highlighted for its role in fostering operational efficiency and enhancing communication between various elements of the rail service.

D. K. T. Kumar et.al[3]

Efficiency gains and revenue optimization are highlighted in the literature, emphasizing the significance of these systems in real-time inventory management. By providing insights into pass usage and demand patterns, Rail Pass Management Systems are recognized for their role in reducing waste and boosting revenue. The integration of data analytics is explored as a powerful tool for empowering rail companies to make informed decisions based on user behavior and market trends, ultimately contributing to strategic planning.

M. A. Khan et.al[4]

The literature review also addresses challenges associated with implementing Rail Pass Management Systems, acknowledging issues such as technological barriers, security concerns, and user adoption challenges. Proposed solutions, ranging from robust cybersecurity measures to targeted user training programs, are discussed as essential components of successful system implementation.

Vidyasagar et.al[5]

Looking towards the future, the literature anticipates exciting developments in Rail Pass Management Systems. The integration of artificial intelligence and machine learning algorithms is identified as a potential avenue for enhancing system intelligence, promising more personalized and efficient services. Additionally, the exploration of mobile applications and contactless technologies is recognized as a key trend for adapting to changing travel preferences and further improving the user experience.

Charith Silva et.al[6]

In conclusion, the literature survey aligns with the project's objectives, providing a rich understanding of the current state of Rail Pass Management Systems. It serves as a foundation for exploring the various dimensions of these systems, encompassing functionalities, integration, efficiency gains, challenges, and future directions.

Anita Honk'a et.al[7]

The interest in studying the maximum search time is mainly to observe how fast the algorithm converges into an optimal solution. The input parameters are the same as in the first test. To better illustrate how this algorithm behaves in time, the graphic in Figure 10 was made. The solutions quality, or cost, is not presented in its absolute value but instead it is normalized to make their comparison easier.

Raviraj Vardhaman Shete et.al[8]

The development of any application of this kind is not finished without a serious and meaningful evaluation of its performance. Therefore, this chapter aims to present and analyze the results of the tests carried out to check this model's efficiency. More specifically, the main objective of these tests is to understand how the program is influenced by its input parameters.

Chapter 3

SYSTEM ANALYSIS

3.1 PROPOSED SYSTEM

The proposed Rail Pass Management System (RPMS) will be a web-based application accessible through a user-friendly interface. It will cater to two primary user groups:

1. **Passengers:** Passengers will utilize the system to conveniently manage their rail passes. Key functionalities for passengers will include:

    - **Pass Search and Purchase:** Browse and purchase various pass options based on travel needs (e.g., frequent traveler passes, regional passes).
    - **Pass Management:** View, manage, and renew existing rail passes online.
    - **Pass Activation:** Activate purchased passes before travel to begin usage tracking.
    - **Trip Tracking:** (View a history of trips taken using the pass.
    - **Account Management:** Update personal details and manage account settings.

2. **Rail Company Admins:** Authorized administrators from the rail company will use the system for broader management tasks, including:

    - **Pass Configuration:** Create and manage different types of rail passes with custom validity periods, pricing, and travel restrictions.
    - **Inventory Management:** Track real-time pass inventory levels and set stock alerts for specific passes.
    - **Reporting and Analytics:** Generate comprehensive reports on sales trends, usage patterns, and customer demographics for informed decision-making.
    - **User Management:** Create and manage user accounts for administrative staff with role-based access control.
    - **System Configuration:** Configure system settings and manage integrations with external systems.

The system will be designed with scalability in mind to accommodate future growth and potential increases in user base and data volume. Secure data storage practices will ensure passenger privacy and protect sensitive information.

3.2 **FEASIBILITY STUDY**

A feasibility study of a Rail Pass Management System would evaluate the technical, operational, economic, and legal aspects of the proposed system.

1. **Technical feasibility**: A Rail Pass Management System would require the development of a web-based application with various features, such as pass purchase, activation, usage tracking, pass validation, and reporting. The technical feasibility would depend on the availability of skilled developers and the availability of resources to implement and maintain the system.
2. **Operational feasibility:** A Rail Pass Management System would need to integrate with existing ticketing and reservation systems, as well as customer relationship management (CRM) systems. The operational feasibility would depend on the ability to integrate the system with existing infrastructure and the availability of resources to manage the system.
3. **Economic feasibility**: The economic feasibility would involve an analysis of the costs and benefits of implementing the system. This would include the cost of developing and maintaining the system, as well as the potential benefits of improved inventory management, reduced waste, increased revenue, and enhanced customer experience.
4. **Legal feasibility**: A Rail Pass Management System would need to comply with various legal and regulatory requirements, such as data protection regulations and privacy laws. The legal feasibility would depend on the ability to comply with these requirements and obtain the necessary permissions and licenses to operate the system.

Based on the results of the feasibility study, it would be possible to determine whether a Rail Pass Management System is a viable solution for rail companies to manage their rail pass sales, inventory, and usage, while providing customers with a streamlined and convenient experience.

3.2.1 **ECONOMICAL**

The Rail Pass Management System (RPMS) offers significant economic benefits for both rail companies and passengers. Here's a breakdown of the expected economic impact:

Reduced Operational Costs for Rail Companies:

1. **Automation:** Automating manual processes like sales, renewals, and tracking eliminates labor costs associated with managing paper-based systems.
2. **Improved Efficiency:** Streamlined workflows translate to faster processing times, reducing administrative overhead and resource allocation.
3. **Data-Driven Decisions:** Real-time data insights enable informed decisions regarding pass pricing, inventory management, and marketing strategies, potentially leading to cost savings and revenue optimization.

Enhanced Revenue Generation:

1. **Dynamic Pricing:** The system can facilitate dynamic pricing strategies based on demand fluctuations, maximizing revenue potential for specific routes and timeframes.
2. **Targeted Pass Offerings:** The ability to create diverse pass options (e.g., frequent traveler passes, weekend passes) can attract new customer segments and generate additional revenue streams.
3. **Improved Customer Experience:** A user-friendly platform can encourage repeat customers and potentially lead to increased pass sales.

Benefits for Passengers:

1. **Convenience:** Passengers gain 24/7 access to manage passes online, eliminating the need for physical visits or lengthy phone calls.
2. **Reduced Costs:** Online purchase options may eliminate processing fees associated with traditional methods.
3. **Transparency:** Real-time information on pass availability and usage patterns empowers passengers to make informed travel decisions.

Overall Economic Impact:

The RPMS fosters a win-win situation for both rail companies and passengers. By streamlining operations, optimizing revenue, and enhancing customer experience, the system can contribute to the overall economic growth of the railway industry. The cost-saving measures achieved by the rail company can potentially translate into more competitive pass pricing, further attracting passengers and stimulating ridership, creating a positive economic cycle.

3.2.2 TECHNICAL

From a technical standpoint, the Rail Pass Management System (RPMS) will be designed with the following considerations:

1. **Technology Stack:** The system can be built using a combination of modern web development technologies. A secure and scalable backend framework (e.g., Python with Django or Java Spring) can manage core functionalities and data storage. A user-friendly frontend can be built using a popular JavaScript framework (e.g., ReactJS or AngularJS) to ensure a responsive and interactive experience across various devices.
2. **Database Design:** A robust database schema will be implemented to store all relevant data related to passes, passengers, inventory, usage, and administrative functions. The database should be optimized for efficient data retrieval and manipulation to handle high transaction volumes. Security measures like encryption will be employed to protect sensitive passenger information.
3. **System Architecture:** The system will likely adopt a three-tier architecture separating the presentation layer (frontend), business logic layer (backend), and data access layer (database). This modular design promotes scalability, maintainability, and security.
4. **API Integration:** The RPMS can be designed with open APIs to facilitate seamless integration with existing ticketing, reservation, and CRM systems used by the rail company. This integration will enable data exchange and create a unified platform for managing passenger activity.
5. **Security Features:** Robust security measures will be implemented to safeguard the system against unauthorized access, data breaches, and fraudulent activities. This may include features like user authentication, data encryption, and secure communication protocols.
6. **Scalability and Performance:** The system architecture will be designed with scalability in mind to accommodate future growth and potential increases in user base and data volume. Performance optimization techniques will be employed to ensure fast response times and a smooth user experience.
7. **Deployment and Maintenance:** The RPMS can be deployed on a cloud platform or on-premise servers, depending on the rail company's specific needs and infrastructure. A well-defined maintenance plan will ensure regular updates, bug fixes, and security patches to keep the system running smoothly and efficiently.

By carefully considering these technical aspects, the RPMS can be developed as a reliable, secure, and scalable software solution to revolutionize rail pass management and enhance the overall experience for both rail companies and passengers.

3.2.3 OPERATIONAL

The Rail Pass Management System (RPMS) will be designed for smooth operation and efficient use within the rail company's existing workflow. Here's a breakdown of the key operational considerations:

1. **User Roles and Access Control:** A role-based access control system will be implemented, granting different levels of access and functionalities to various user groups. Passengers will have access to features like pass purchase, management, and usage tracking. Administrators will have broader control over pass configuration, inventory management, reporting, user management, and system settings.
2. **Workflow Automation:** Manual processes associated with pass sales, renewals, and usage tracking will be automated within the RPMS. This will streamline workflows, minimize human error, and free up staff time to focus on other tasks or provide customer support.
3. **Integration with Existing Systems:** Seamless integration with existing ticketing, reservation, and CRM systems will be crucial. This will enable data exchange between systems, eliminating the need for manual data entry and fostering a centralized platform for managing passenger activity.
4. **Data Backup and Recovery:** A robust data backup and recovery plan will be established to ensure the system's resilience. Regular backups will be conducted to protect against data loss due to hardware failures or unforeseen circumstances. A disaster recovery plan will outline procedures to restore the system and data in case of major disruptions.
5. **System Monitoring and Maintenance:** The system will be continuously monitored for performance, errors, and security vulnerabilities. Regular maintenance will include software updates, bug fixes, and security patches to ensure optimal system operation.
6. **User Training and Support:** Training materials and support channels will be provided to both passengers and rail company staff to ensure smooth system adoption. Passengers will benefit from user guides and tutorials explaining how to navigate the system and utilize its features. Staff will receive comprehensive training on system administration, troubleshooting procedures, and data analysis.
7. **System Change Management:** A well-defined change management process will be implemented to manage system updates and new feature introductions. This process will ensure proper testing,

documentation, and user communication to minimize disruptions and ensure a smooth transition for all stakeholders.

By addressing these operational considerations, the RPMS can be seamlessly integrated into the rail company's daily operations, enhancing efficiency, improving data management, and empowering both passengers and staff with a user-friendly and reliable platform.

Chapter 4

SYSTEM REQUIREMENT SPECIFICATION

4.1 HARDWARE REQUIREMENT

- P-III or higher Processor.
- 40 GB HDD
- 256 SDD
- 8 GB RAM (min)
- Color VGA 800x600 resolution monitor ware requirement

4.2 SOFTWARE REQUIREMENT

• Project Name	**Rail Pass Management System**
• Language Used	PHP5.6, PHP7.x
• Database	MySQL 5.x
• User Interface Design	HTML, AJAX, JQUERY, JAVASCRIPT
• Web Browser	Mozilla, Google Chrome, IE8, OPERA
• Software	XAMPP / Wamp / Mamp/ Lamp (anyone)

4.3 FUNCTIONAL REQUIREMENTS

1. **User Authentication and Authorization:**
    - Users should be able to register for an account and log in securely.
    - Admins should have special privileges to manage user accounts and system settings.
2. **Pass Management:**
    - Users should be able to apply for rail passes, including different types such as monthly, quarterly, and yearly passes.
    - Admins should be able to approve or reject pass applications.
    - Users should be able to renew passes before they expire.
    - The system should allow users to cancel passes if needed.
3. **Pass Usage Tracking:**

- The system should track pass usage, including check-ins and check-outs at stations.
- Users should be able to view their pass usage history.

4. **Reporting:**
    - Admins should be able to generate reports on pass usage, revenue generated, and other relevant metrics.
    - Reports should be customizable and exportable in different formats.
5. **Notification System:**
    - Users should receive notifications for pass renewals, upcoming expirations, and other important events.
    - Admins should be notified of pass applications awaiting approval and other administrative tasks.
6. **User Management:**
    - Admins should be able to manage user accounts, including editing user information and resetting passwords.

4.4 NON-FUNCTIONAL REQUIREMENTS

1. **Performance:**
    - The system should respond to user actions within a reasonable time frame.
    - It should be capable of handling simultaneous user interactions without significant delays.
2. **Security:**
    - User data should be stored securely and encrypted to prevent unauthorized access.
    - The system should implement measures to protect against common security threats such as SQL injection and cross-site scripting (XSS).
3. **Reliability:**
    - The system should be available for use during scheduled hours with minimal downtime.
    - Data integrity should be maintained, and backups should be performed regularly to prevent data loss.

4. **Compatibility:**
    - The system should be compatible with popular web browsers such as Mozilla, Google Chrome, IE8, and Opera.
    - It should also be compatible with the specified software (XAMPP / Wamp / Mamp / Lamp).
5. **Scalability:**
    - The system should be able to accommodate a growing number of users and data volume over time.
    - Scalability should be achieved through efficient resource utilization and scaling strategies.
6. **Usability:**
    - The user interface should be intuitive and easy to navigate, ensuring a positive user experience.
    - User input validation and error handling should be implemented to guide users and prevent mistakes.

Chapter 5

SYSTEM DESIGN

5.1 **SYSTEM ARCHITECTURE**

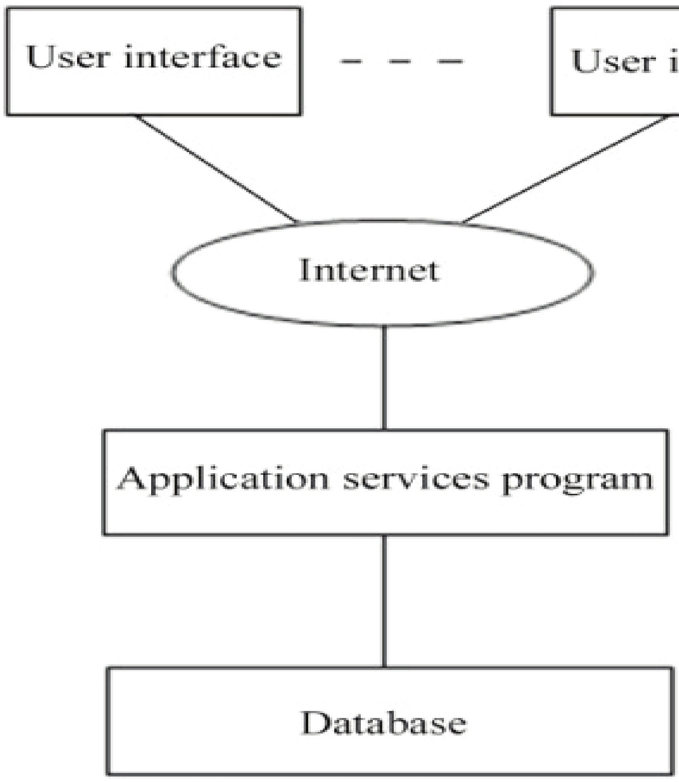


FIGURE 5.1: - SYSTEM ARCHITECTURE DIAGRAM

5.2 **DATA FLOW DIAGRAM**

A DFD represents flow of data through a system. Data flow diagrams are commonly used during problem analysis. It views a system as a function that transforms the input into desired output. A DFD shows movement of data through the different transformations or processes in the system.

Dataflow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to restock how any system is developed can be determined through a dataflow diagram. The appropriate register saved in database and maintained by appropriate authorities

Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

The following observations about DFDs are essential:

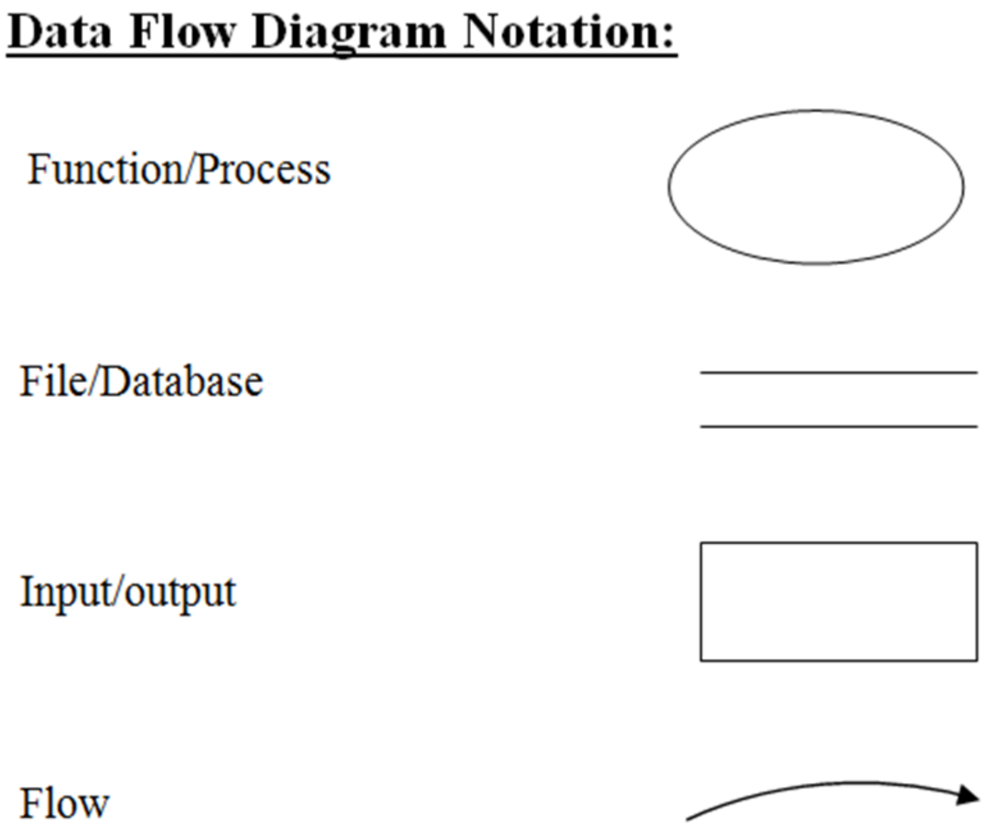


FIGURE 5.2.1: DATA FLOW DIAGRAM NOTATION

Zero Level

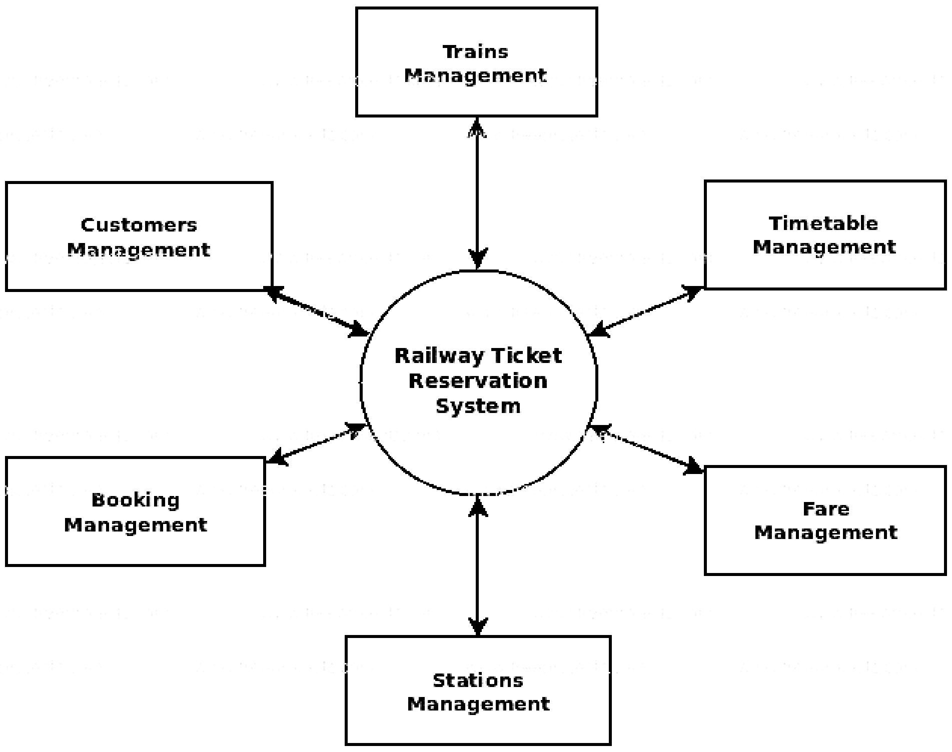


FIGURE 5.2.2: ZERO LEVEL

Frist Level DFD

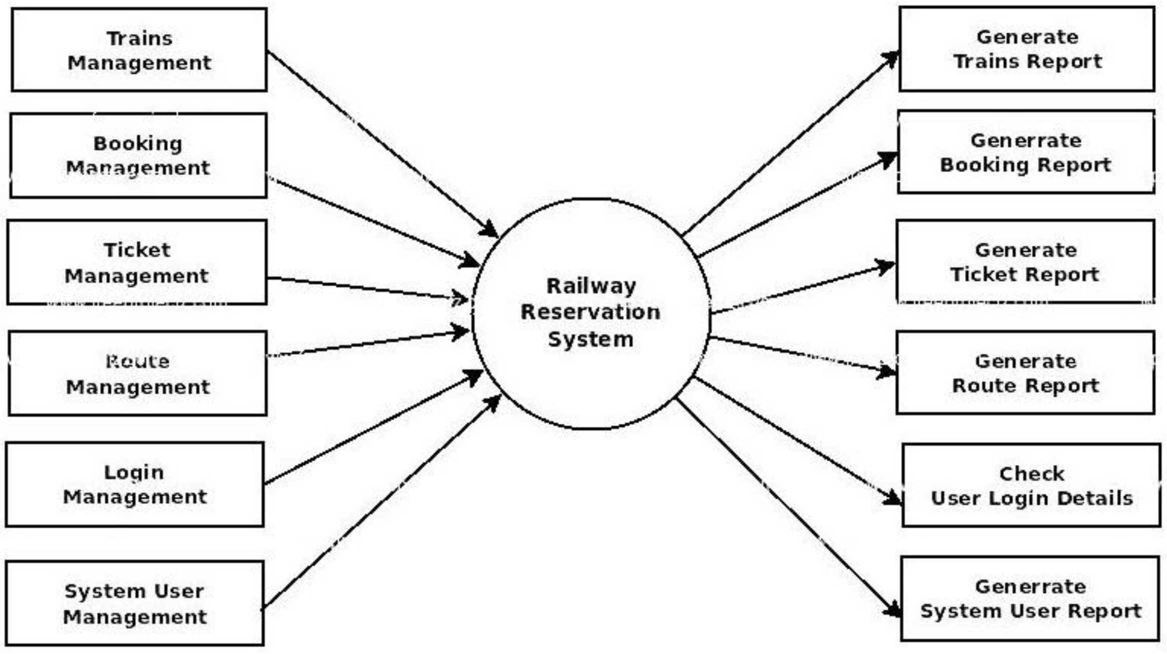


FIGURE 5.2.3: FIRST LEVEL DFD

Second level DFD

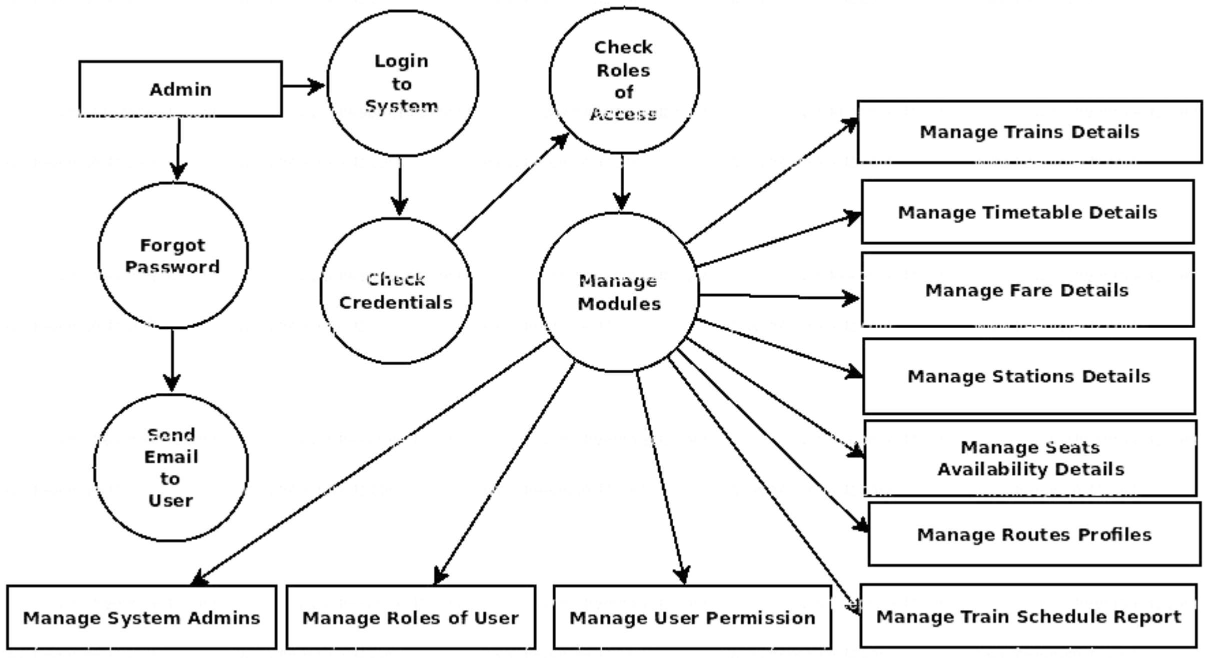


FIGURE 5.2.4: SECOND LEVEL DFD

ER-Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

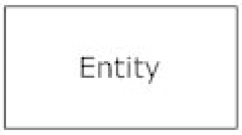
ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

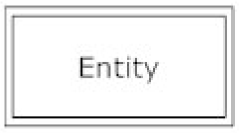
At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.

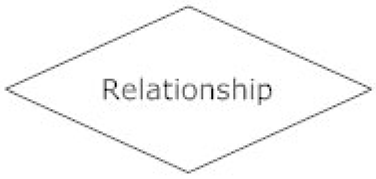
ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

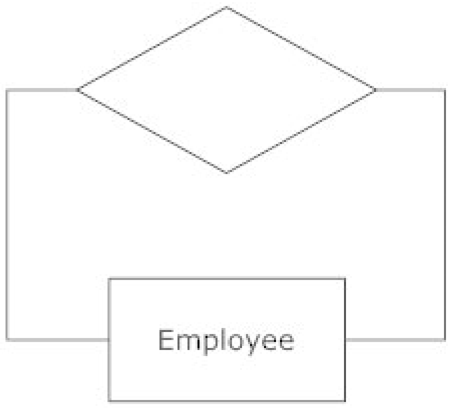
Common Entity Relationship Diagram Symbols

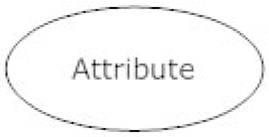
An ER diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

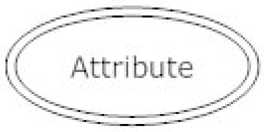
1. **Entities**, which are represented by rectangles. An entity is an object or concept 

about which you want to store information. A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.


2. **Actions**, which are represented by diamond shapes, show how two entities share information in the database. 

In some cases, entities can be self-linked. For example, employees can supervise other employees.


3. **Attributes**, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.


A multivalued attribute can have more than one value. For example, an employee entity can have multiple skill values. 

A derived attribute is based on

another attribute. For example, an employee's monthly salary is based on the employee's annual salary.

4. **Connecting lines**, solid lines that connect attributes to show the relationships of entities in the diagram.

Cardinality specifies how many instances of an entity relate to one instance of another entity. Ordinality is also closely linked to cardinality. While cardinality specifies the occurrences of a relationship, ordinality describes the relationship as either mandatory or optional. In other words, cardinality specifies the maximum number of relationships and ordinality specifies the absolute minimum number of relationships.

5.3 **UML DIAGRAMS**

Use case Diagram

A use case diagram is a type of behavioral diagram in the Unified Modeling Language (UML) that illustrates the interactions between actors and a system, and describes the functionality provided by the system. Here's an example of a use case diagram:

In this use case diagram, the "User" is an external actor who interacts with the "System." The "System" provides four main functionalities, which are represented by the use cases: "View Information," "Update Information," "Search Information," and "Remove Information." These use cases describe the various interactions that the User can have with the System.

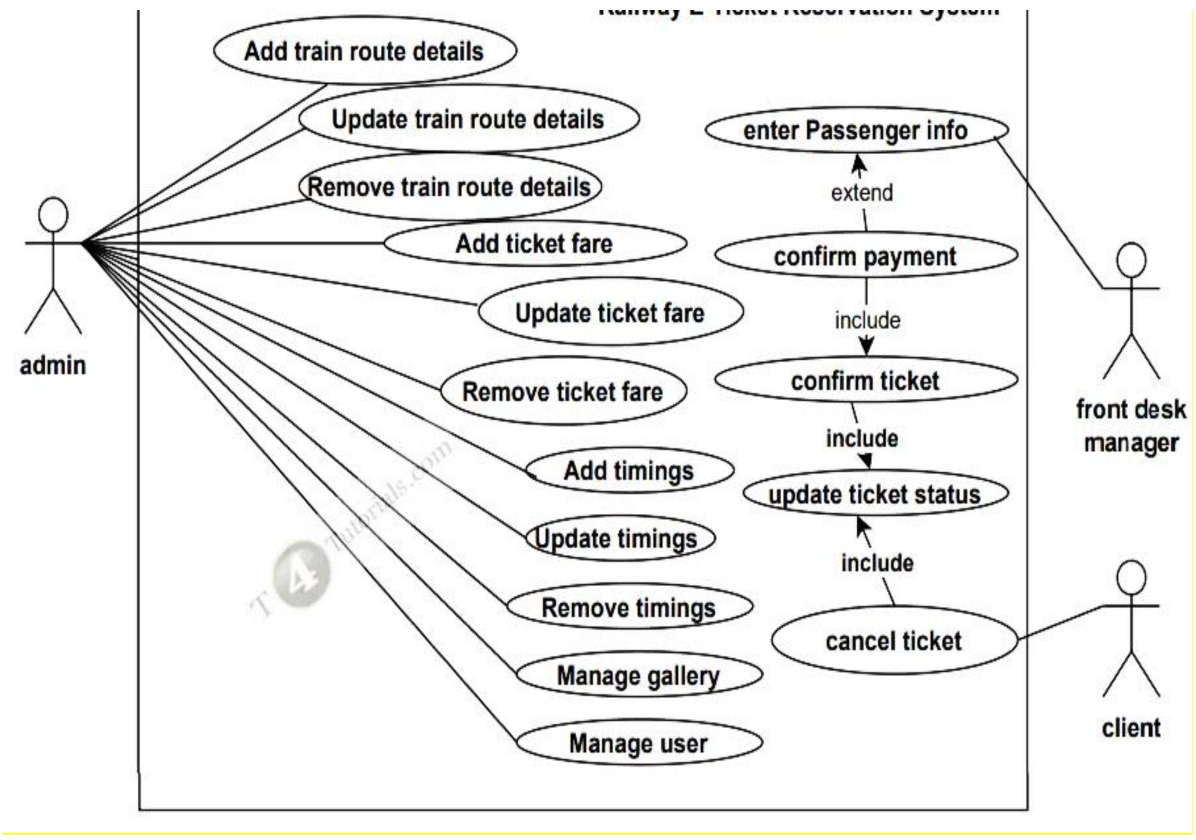


FIGURE 5.3.1: USE CASE DIAGRAM

Class Digram

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code.

It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.

5.4 **PURPOSE OF CLASS DIAGRAMS**

The main purpose of class diagrams is to build a static view of an application. It is the only diagram that is widely used for construction, and it can be mapped with object-oriented languages. It is one of the most popular UML diagrams. Following are the purpose of class diagrams given below:

1. It analyses and designs a static view of an application.
2. It describes the major responsibilities of a system.
3. It is a base for component and deployment diagrams.
4. It incorporates forward and reverse engineering.

5.5 **BENEFITS OF CLASS DIAGRAMS**

1. It can represent the object model for complex systems.
2. It reduces the maintenance time by providing an overview of how an application is structured before coding.
3. It provides a general schematic of an application for better understanding.
4. It represents a detailed chart by highlighting the desired code, which is to be programmed.
5. It is helpful for the stakeholders and the developers.

5.6 **VITAL COMPONENTS OF A CLASS DIAGRAM**

The class diagram is made up of three sections:

1. Upper Section: The upper section encompasses the name of the class. A class is a representation of similar objects that shares the same relationships, attributes, operations, and semantics. Some of the following rules that should be taken into account while representing a class are given below:
   - Capitalize the initial letter of the class name.
   - Place the class name in the center of the upper section.
   - A class name must be written in bold format.
   - The name of the abstract class should be written in italics format.

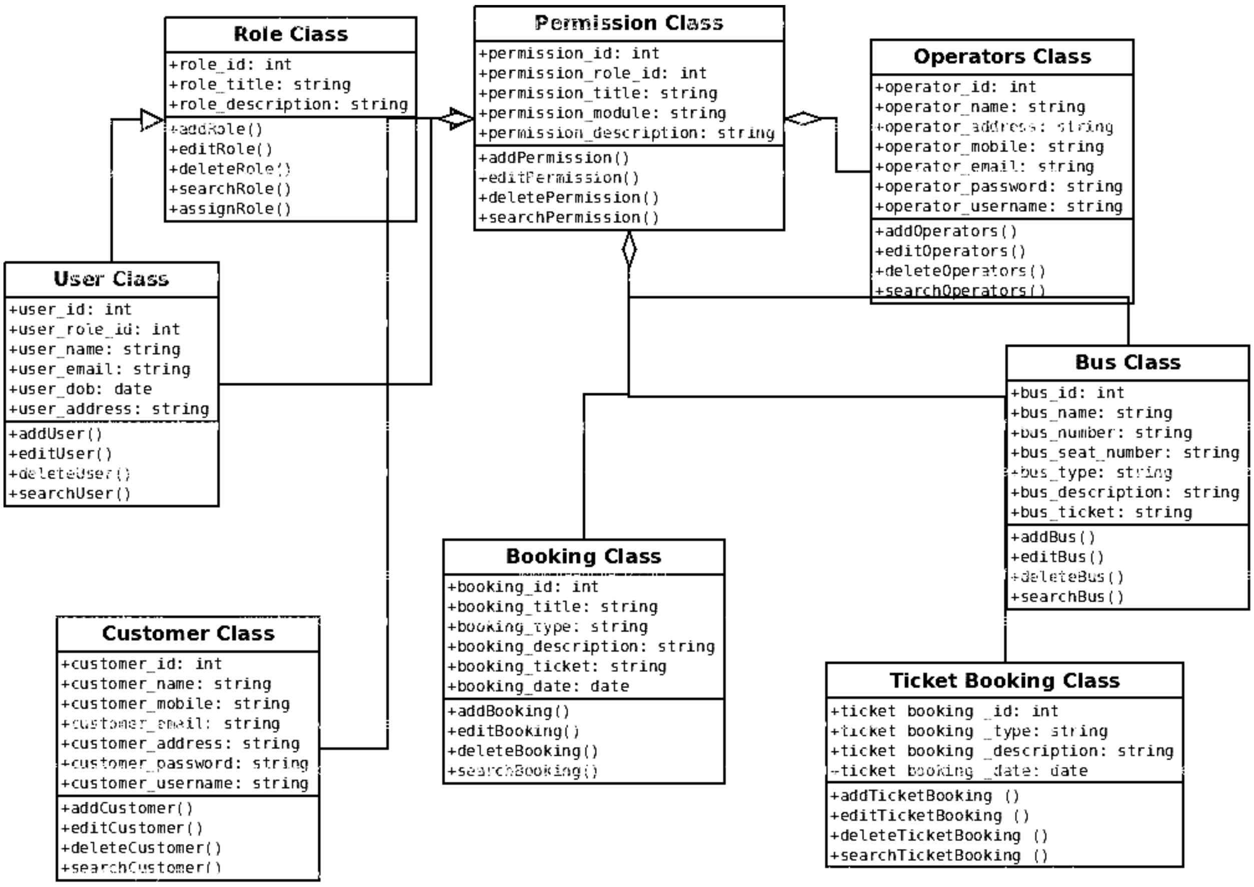
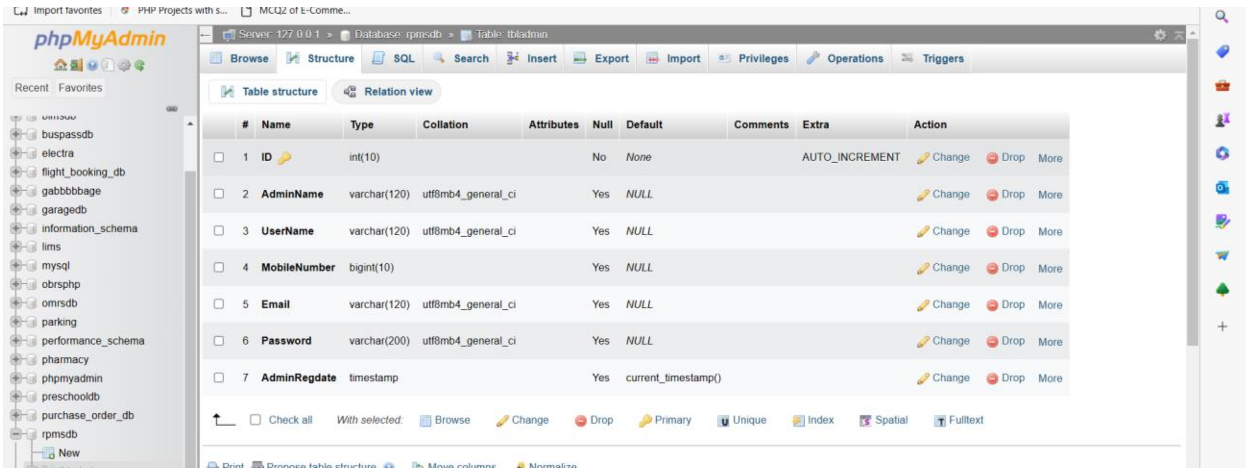


FIGURE 5.6: CLASS DIAGRAM

5.7 **DATABASE DESIGN**

Table admin

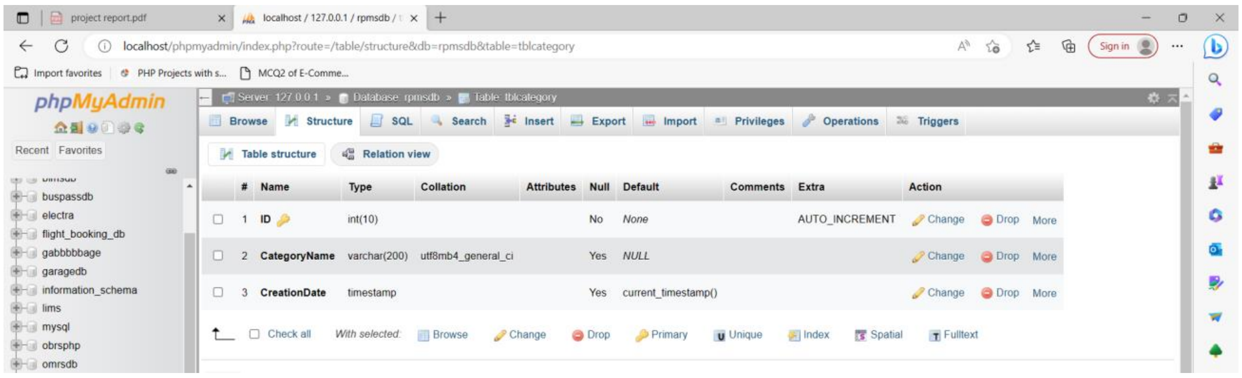


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	AdminName	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	UserName	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	MobileNumber	bigint(10)			Yes	NULL			Change Drop More
5	Email	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	Password	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
7	AdminRegdate	timestamp			Yes	current_timestamp()			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

FIGURE 5.7.1: TABLE ADMIN

Table category



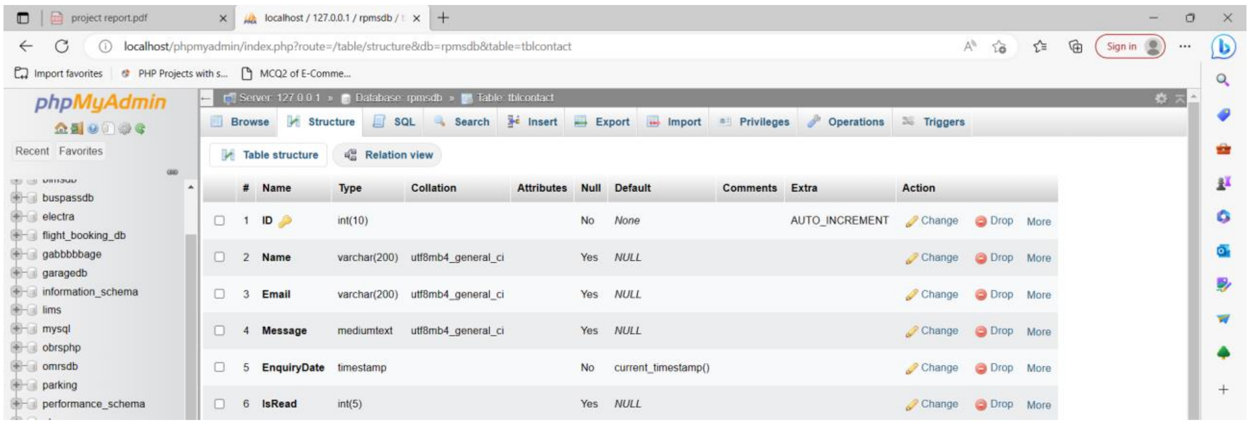
localhost/phpmyadmin/index.php?route=/table/structure&db=rpmsdb&table=tblcategory

Server: 127.0.0.1 » Database: rpmsdb » Table: tblcategory

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	CategoryName	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	CreationDate	timestamp			Yes	current_timestamp()			Change Drop More

FIGURE 5.7.2: TABLE CATEGORY

Table contact



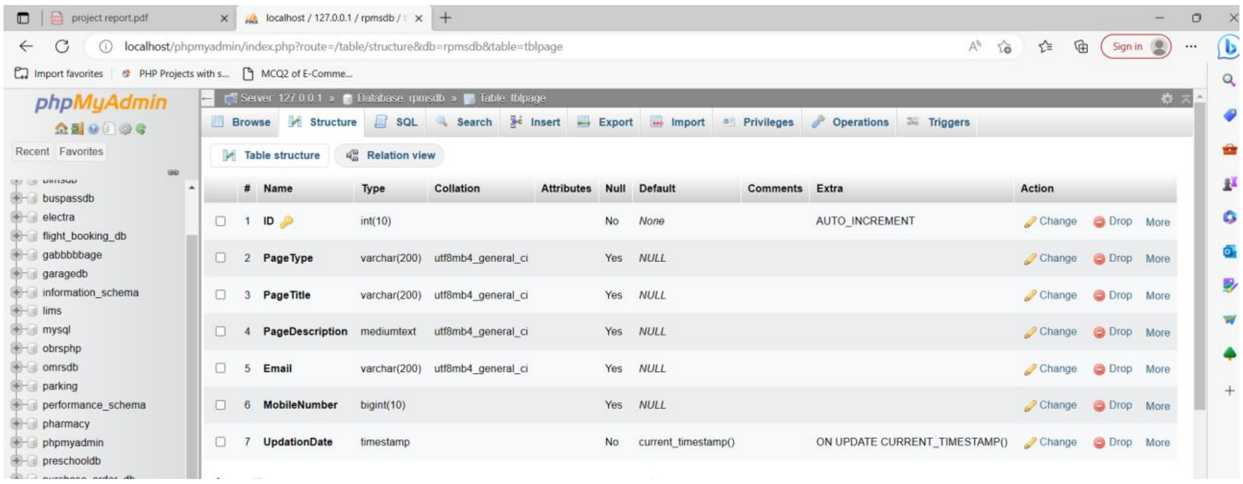
localhost/phpmyadmin/index.php?route=/table/structure&db=rpmsdb&table=tblcontact

Server: 127.0.0.1 » Database: rpmsdb » Table: tblcontact

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	Name	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	Email	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	Message	mediumtext	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	EnquiryDate	timestamp			No	current_timestamp()			Change Drop More
6	IsRead	int(5)			Yes	NULL			Change Drop More

FIGURE 5.7.3: TABLE CONTACT

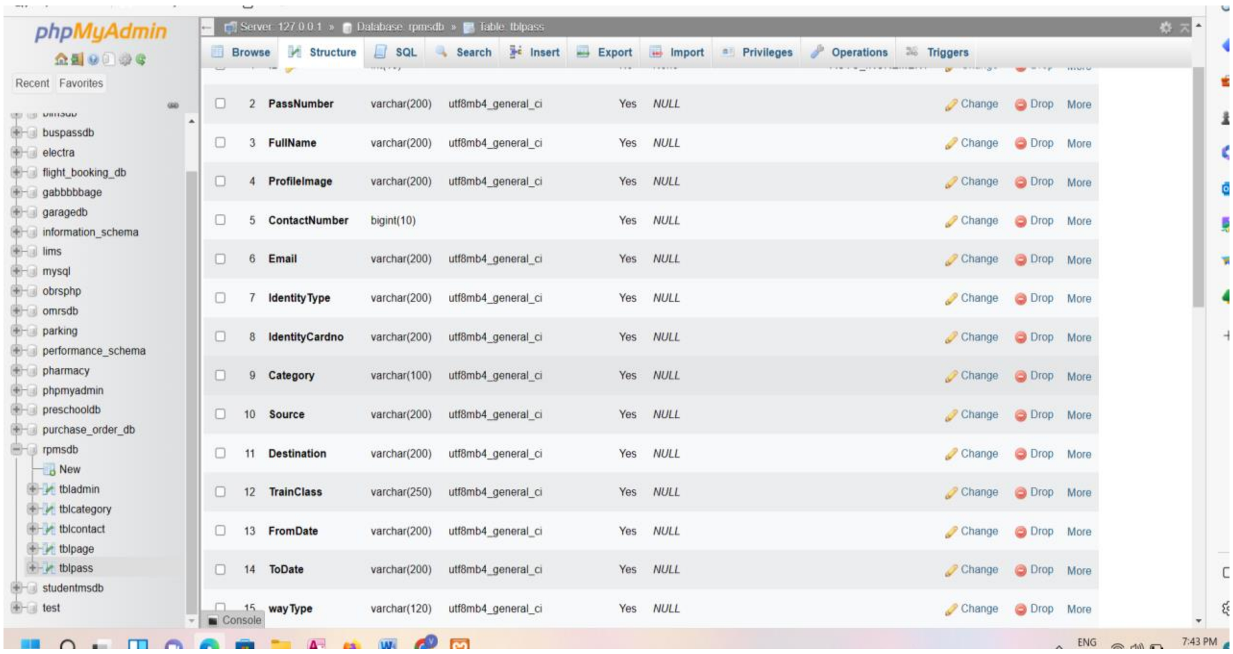
Table page



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	PageType	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	PageTitle	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	PageDescription	mediumtext	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	Email	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	MobileNumber	bigint(10)			Yes	NULL			Change Drop More
7	UpdationDate	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

FIGURE 5.7.4: TABLE PAGE

Table Pass



#	Name	Type	Collation	Null	Default	Action
2	PassNumber	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
3	FullName	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
4	ProfileImage	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
5	ContactNumber	bigint(10)		Yes	NULL	Change Drop More
6	Email	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
7	IdentityType	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
8	IdentityCardno	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
9	Category	varchar(100)	utf8mb4_general_ci	Yes	NULL	Change Drop More
10	Source	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
11	Destination	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
12	TrainClass	varchar(250)	utf8mb4_general_ci	Yes	NULL	Change Drop More
13	FromDate	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
14	ToDate	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change Drop More
15	wayType	varchar(120)	utf8mb4_general_ci	Yes	NULL	Change Drop More

FIGURE 5.7.5: TABLE PASS

Chapter 6

IMPLEMENTATION

6.1 IMPLEMENTATION ENVIRONMENT

Front-end

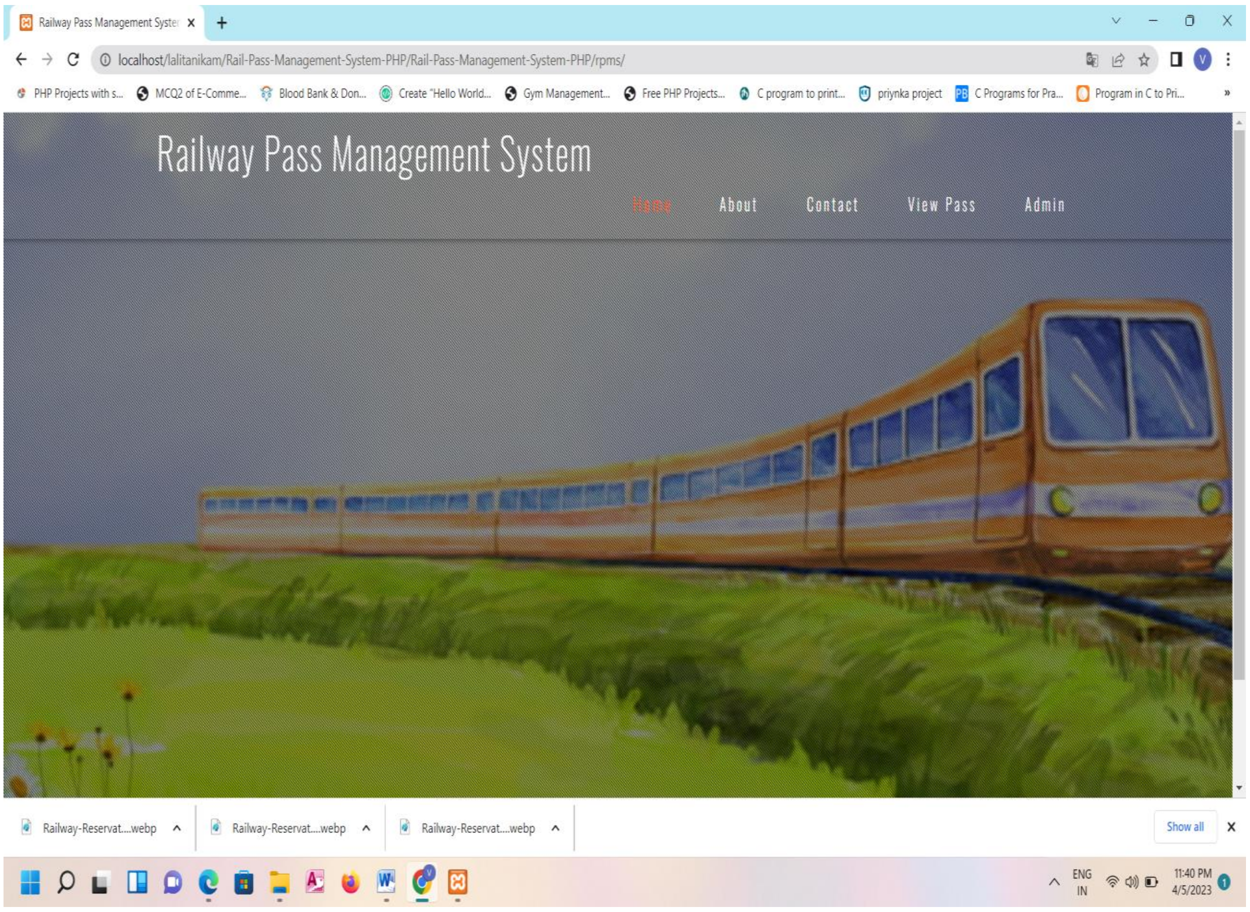
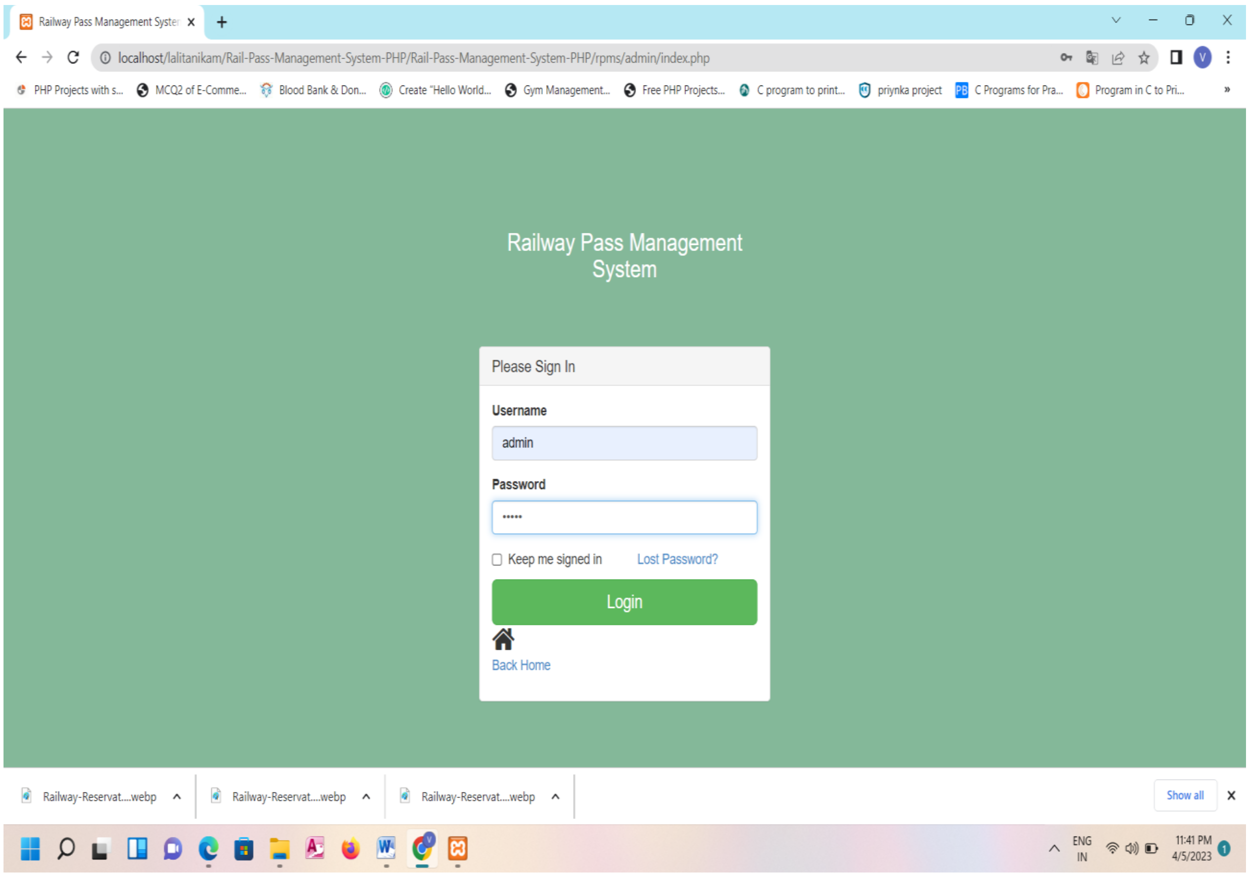

Railway Pass Management System
localhost/lalitanikam/Rail-Pass-Management-System-PHP/Rail-Pass-Management-System-PHP/rpms/
Home
About
Contact
View Pass
Admin

FIGURE 6.1.1: HOME PAGE



Railway Pass Management System

Please Sign In

Username

admin

Password

•••••

Keep me signed in   Lost Password?

FIGURE 6.1.2: LOGIN PAGE

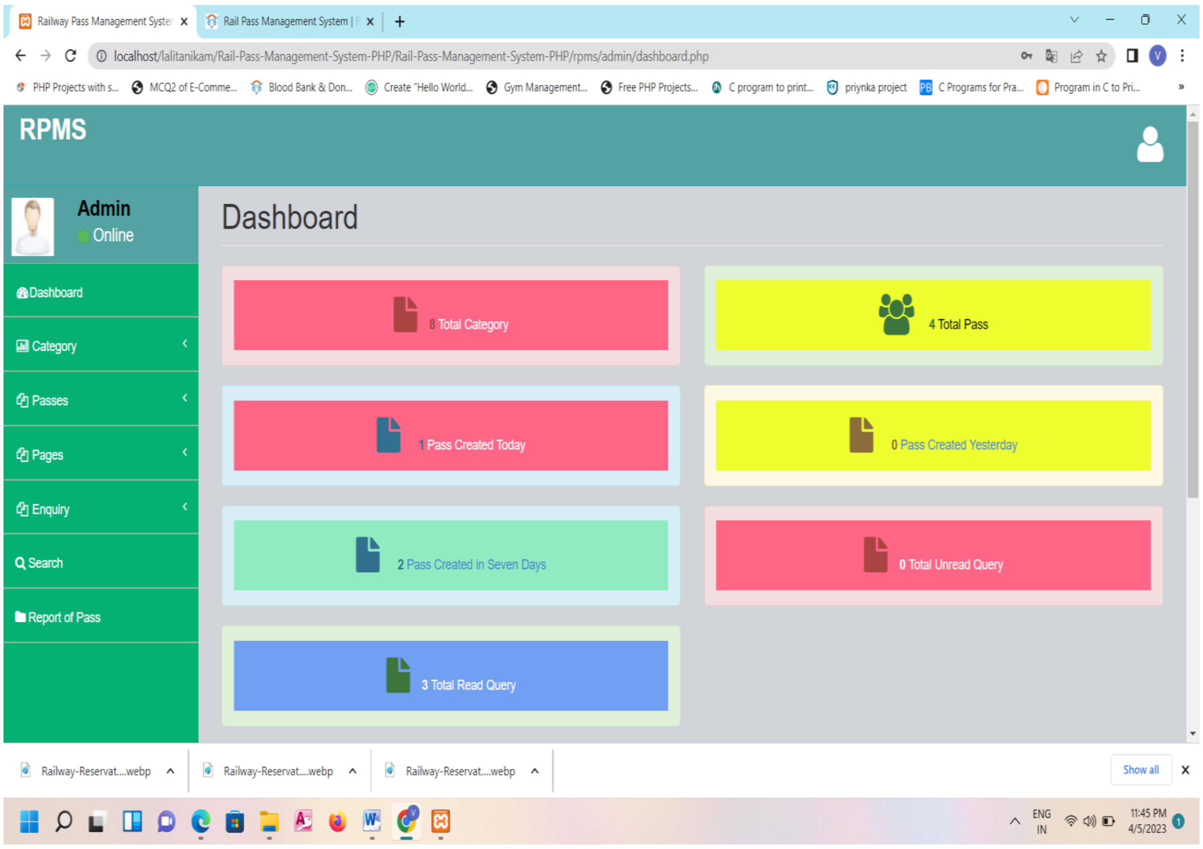
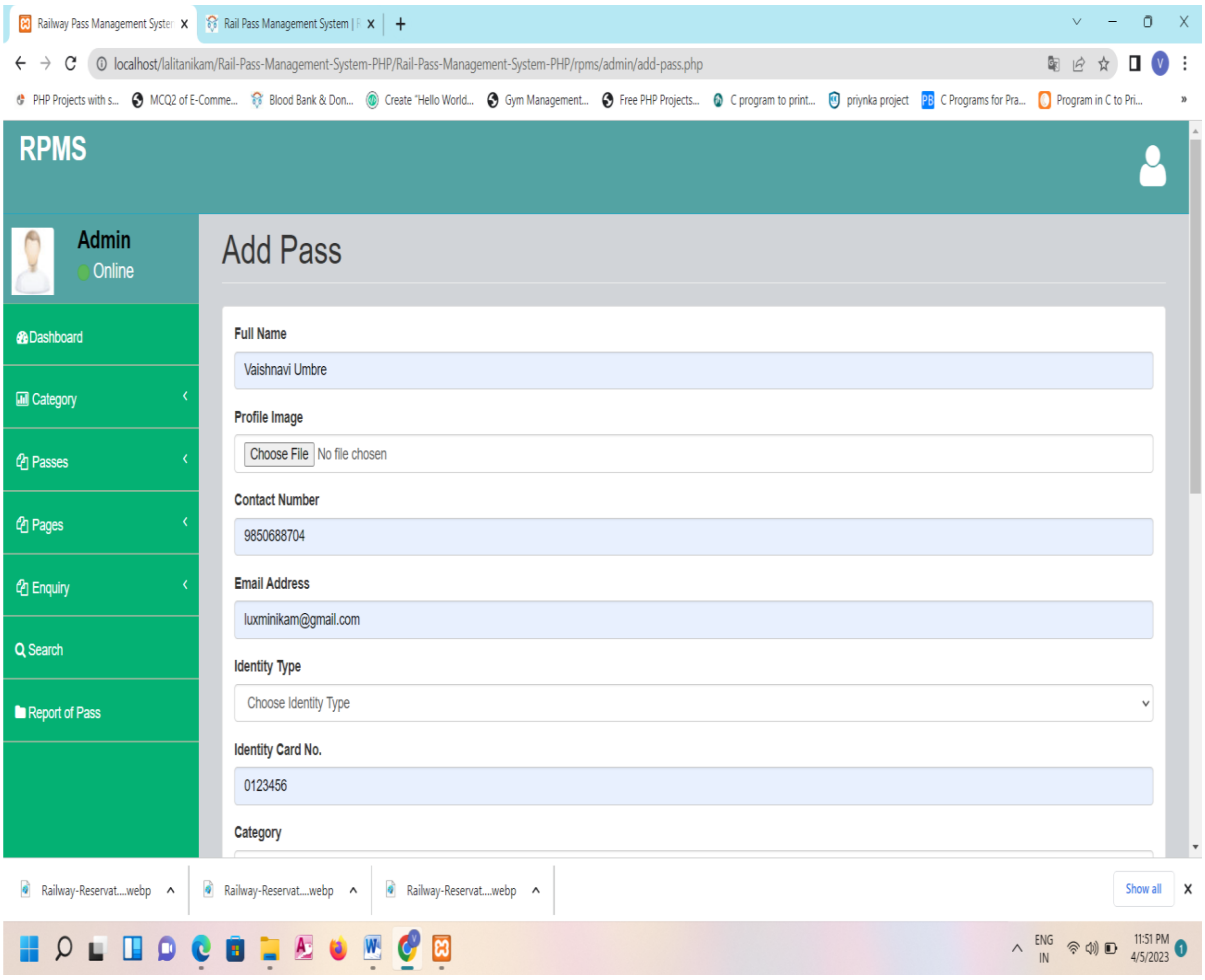


FIGURE 6.1.3: DASHBOARD PAGE



RPMS

Admin
Online

Add Pass

Full Name: Vaishnavi Umbre

Profile Image: Choose File No file chosen

Contact Number: 9850688704

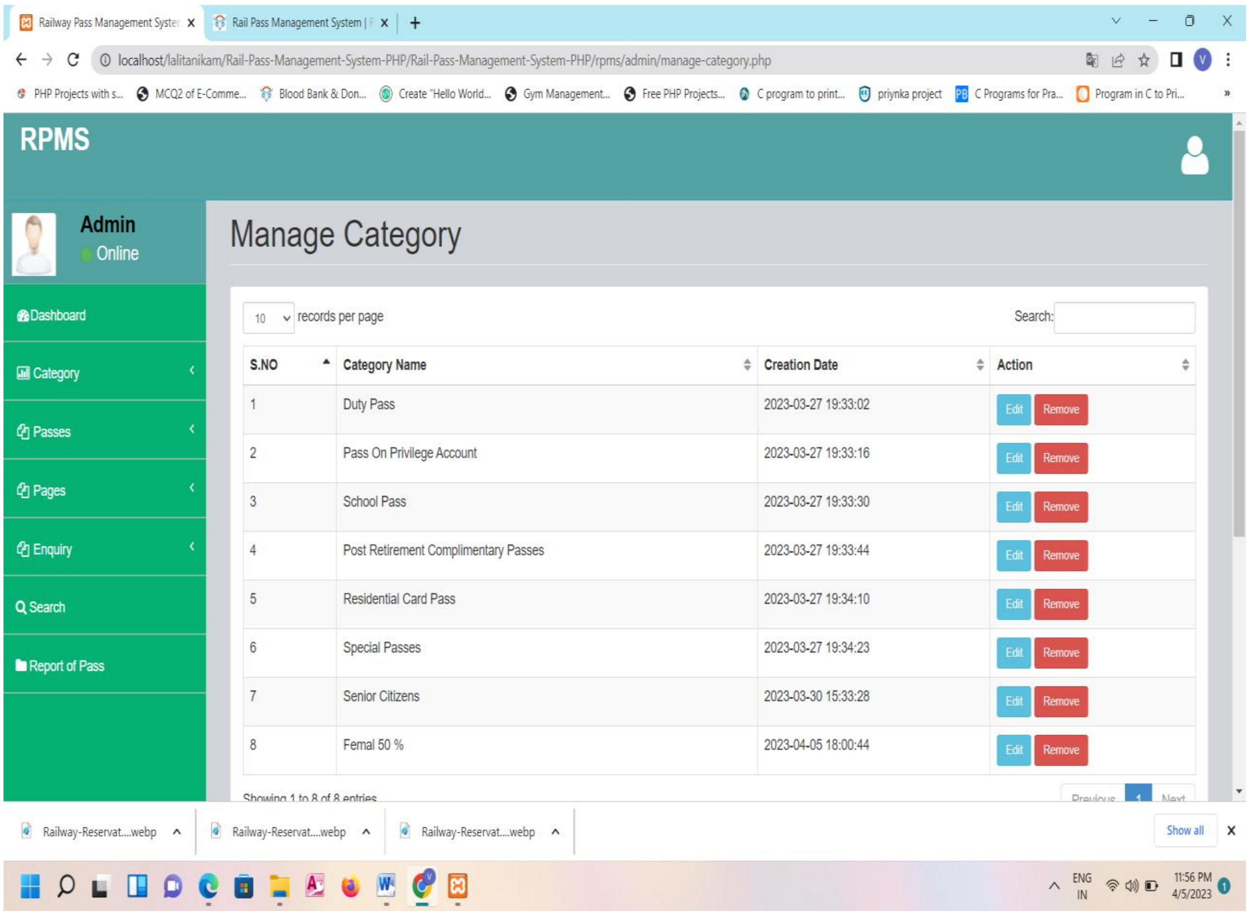
Email Address: luxminikam@gmail.com

Identity Type: Choose Identity Type

Identity Card No.: 0123456

Category

FIGURE 6.1.4: ADD PASS PAGE



Manage Category

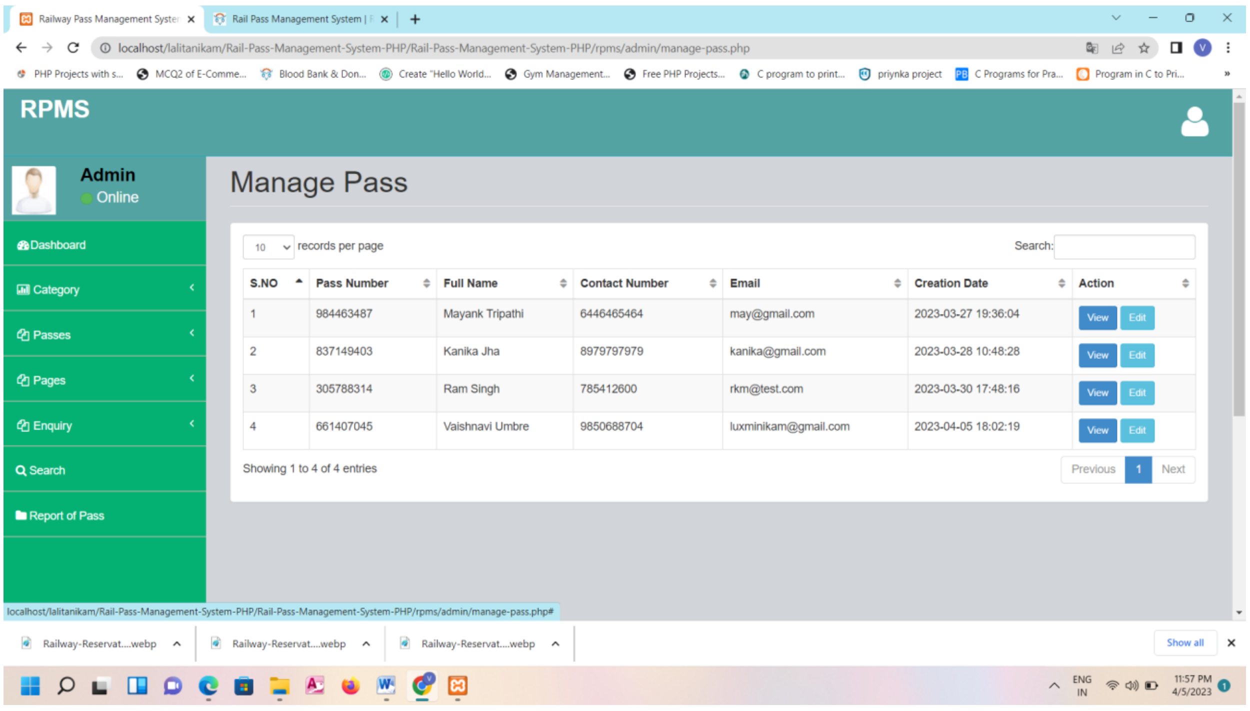
10 records per page Search:

S.NO	Category Name	Creation Date	Action
1	Duty Pass	2023-03-27 19:33:02	Edit Remove
2	Pass On Privilege Account	2023-03-27 19:33:16	Edit Remove
3	School Pass	2023-03-27 19:33:30	Edit Remove
4	Post Retirement Complimentary Passes	2023-03-27 19:33:44	Edit Remove
5	Residential Card Pass	2023-03-27 19:34:10	Edit Remove
6	Special Passes	2023-03-27 19:34:23	Edit Remove
7	Senior Citizens	2023-03-30 15:33:28	Edit Remove
8	Femal 50 %	2023-04-05 18:00:44	Edit Remove

Showing 1 to 8 of 8 entries

localhost/lalitanikam/Rail-Pass-Management-System-PHP/Rail-Pass-Management-System-PHP/rpms/admin/manage-category.php
RPMS
Admin
Online
Dashboard
Category
Passes
Pages
Enquiry
Search
Report of Pass

FIGURE 6.1.5: MANAGE CATEGORY PAGE



Manage Pass

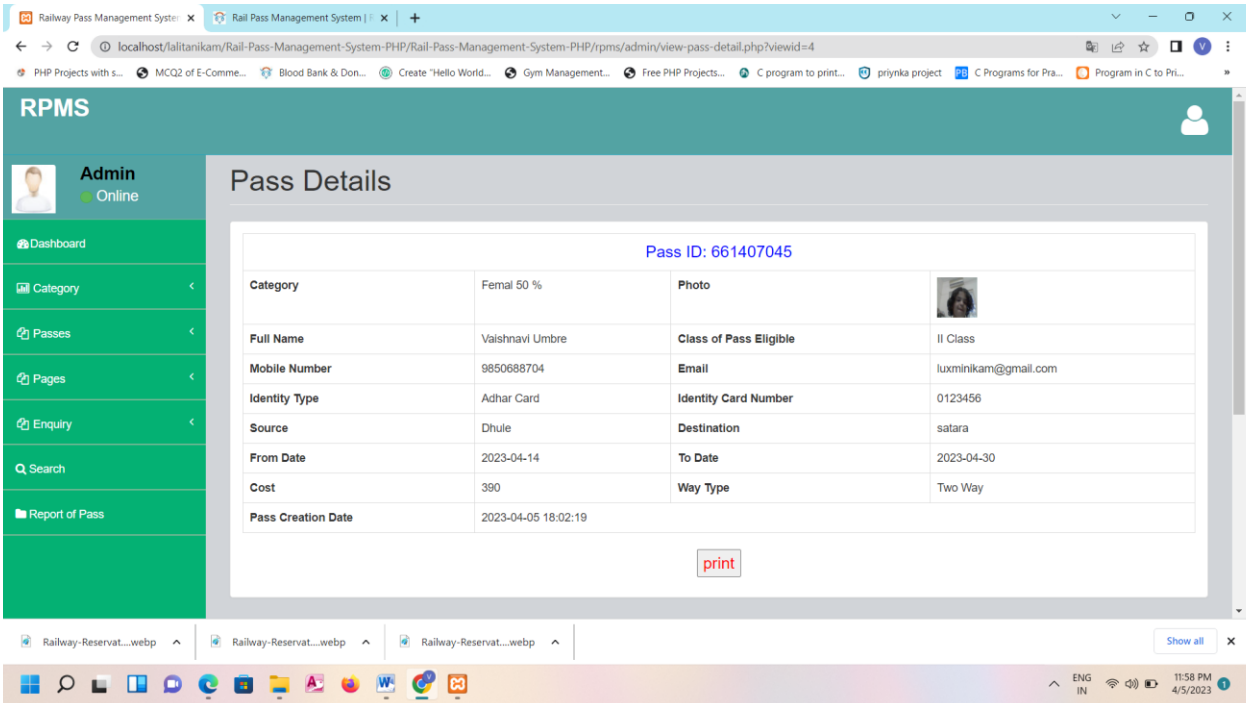
10 records per page Search:

S.NO	Pass Number	Full Name	Contact Number	Email	Creation Date	Action
1	984463487	Mayank Tripathi	6446465464	may@gmail.com	2023-03-27 19:36:04	View Edit
2	837149403	Kanika Jha	8979797979	kanika@gmail.com	2023-03-28 10:48:28	View Edit
3	305788314	Ram Singh	785412600	rkm@test.com	2023-03-30 17:48:16	View Edit
4	661407045	Vaishnavi Umbre	9850688704	luxminikam@gmail.com	2023-04-05 18:02:19	View Edit

Showing 1 to 4 of 4 entries

localhost/lalitanikam/Rail-Pass-Management-System-PHP/Rail-Pass-Management-System-PHP/rpms/admin/manage-pass.php
RPMS
Admin
Online
Dashboard
Category
Passes
Pages
Enquiry
Search
Report of Pass
Previous 1 Next

FIGURE 6.1.6: MANAGE PASS



Pass Details

Pass ID: 661407045

Category	Femal 50 %	Photo	
Full Name	Vaishnavi Umbre	Class of Pass Eligible	II Class
Mobile Number	9850688704	Email	luxminikam@gmail.com
Identity Type	Adhar Card	Identity Card Number	0123456
Source	Dhule	Destination	satara
From Date	2023-04-14	To Date	2023-04-30
Cost	390	Way Type	Two Way
Pass Creation Date	2023-04-05 18:02:19		

FIGURE 6.1.7: PASS PAGE

Database

Databases are used to store and organize data. There are different types of databases, and each type is suited for different types of applications. Here are some of the most common types of databases:

1. **Relational Databases**: These databases store data in tables, where each table has rows and columns. Each row represents a record, and each column represents a field. Relational databases use SQL (Structured Query Language) to manage data.
2. **NoSQL Databases:** NoSQL databases are designed to handle unstructured data, such as documents, images, and videos. They do not use tables, and data is stored in a more flexible and scalable format.
3. **Object-Oriented Databases:** These databases store data as objects, which are similar to classes in object-oriented programming. They are used for applications that require complex data structures.
4. **Graph Databases:** Graph databases store data in nodes and edges, which are used to represent relationships between data. They are used for applications that require complex relationship data, such as social networks and recommendation systems.

5. **Time-Series Databases:** These databases store data with timestamps, and are used for applications that require storing and analyzing time-based data, such as sensor data.

   These are just some of the types of databases available, and each has its own strengths and weaknesses. Choosing the right database for your project will depend on your specific requirements, such as the type of data you need to store, how you need to query and analyze that data, and how much scalability and flexibility you require.

What is SQL?

SQL is a short-form of the structured query language, and it is pronounced as S-Q-L or sometimes as See-Quell.

This database language is mainly designed for maintaining the data in relational database management systems. It is a special tool used by data professionals for handling structured data (data which is stored in the form of tables). It is also designed for stream processing in RDSMS.

You can easily create and manipulate the database, access and modify the table rows and columns, etc. This query language became the standard of ANSI in the year of 1986 and ISO in the year of 1987.

If you want to get a job in the field of data science, then it is the most important query language to learn. Big enterprises like Facebook, Instagram, and LinkedIn, use SQL for storing the data in the back-end.

Process of SQL

When we are executing the command of SQL on any Relational database management system, then the system automatically finds the best routine to carry out our request, and the SQL engine determines how to interpret that particular command.

Structured Query Language contains the following four components in its process:

- Query Dispatcher
- Optimization Engines
- Classic Query Engine
- SQL Query Engine, etc.

A classic query engine allows data professionals and users to maintain non-SQL queries. The architecture of SQL is shown in the following diagram:

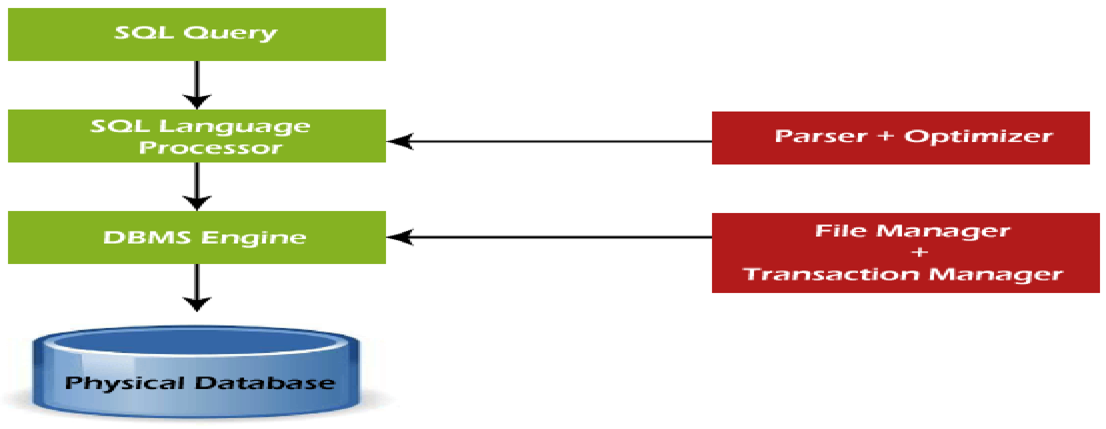


FIGURE 6.1.8: SQL ARCHITECTURE DIAGRAM

Some SQL Commands

The SQL commands help in creating and managing the database. The most common SQL commands which are highly used are mentioned below:

1. CREATE command
2. UPDATE command
3. DELETE command
4. SELECT command
5. DROP command
6. INSERT command

CREATE Command

This command helps in creating the new database, new table, table view, and other objects of the database.

UPDATE Command

This command helps in updating or changing the stored data in the database.

DELETE Command

This command helps in removing or erasing the saved records from the database tables. It erases single or multiple tuples from the tables of the database.

SELECT Command

This command helps in accessing the single or multiple rows from one or multiple tables of the database. We can also use this command with the WHERE clause.

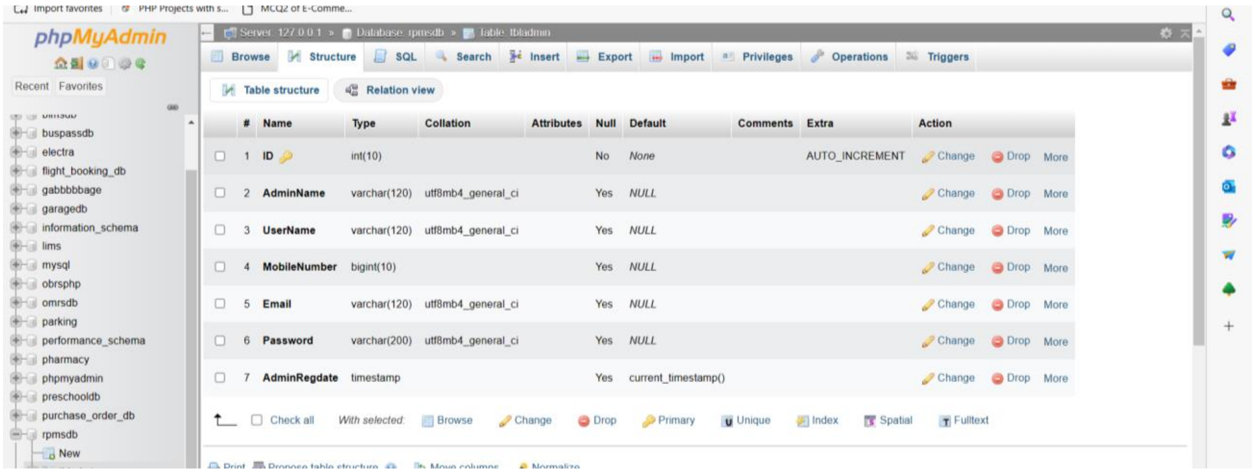
DROP Command

This command helps in deleting the entire table, table view, and other objects from the database.

INSERT Command

This command helps in inserting the data or records into the database tables. We can easily insert the records in single as well as multiple rows of the table.

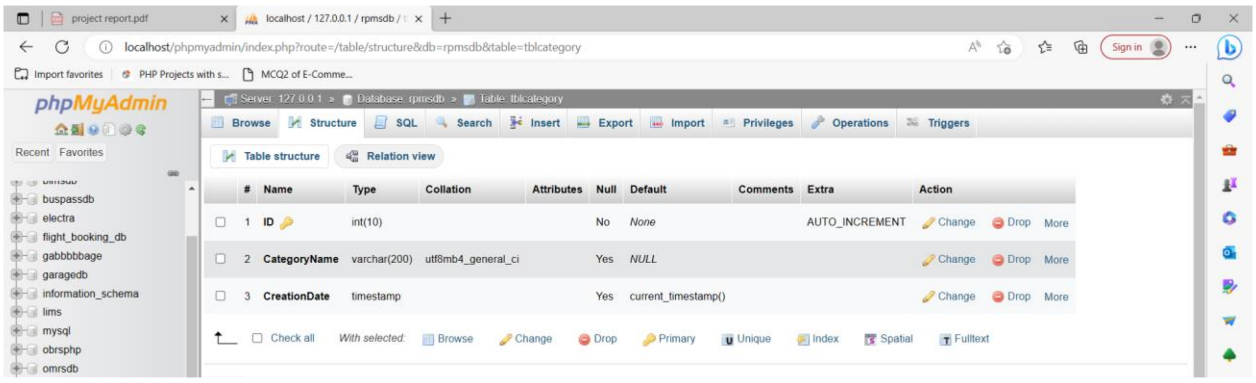
Table admin



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	AdminName	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	UserName	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	MobileNumber	bigint(10)			Yes	NULL			Change Drop More
5	Email	varchar(120)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	Password	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
7	AdminRegdate	timestamp			Yes	current_timestamp()			Change Drop More

FIGURE 6.1.9: TABLE ADMIN

Table category's

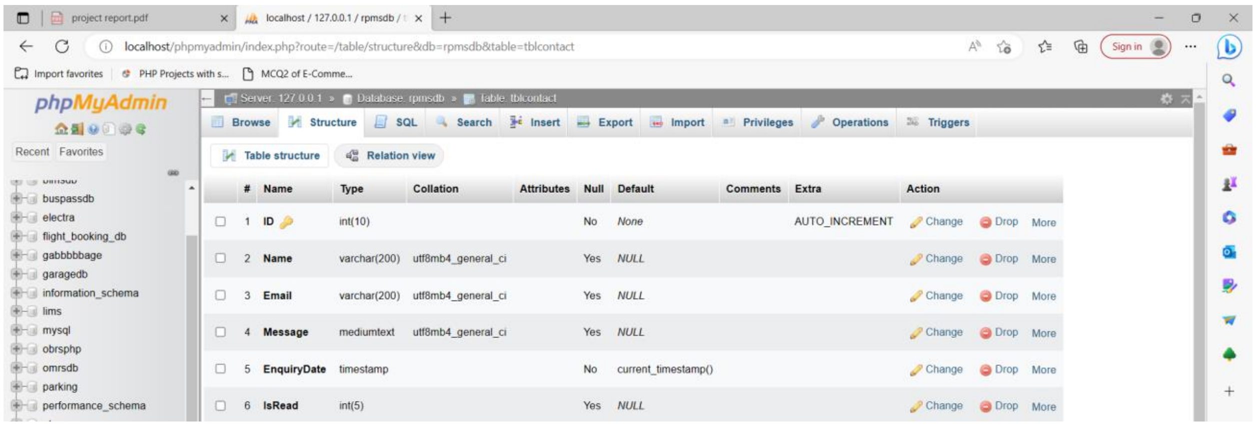


localhost/phpmyadmin/index.php?route=/table/structure&db=rpmsdb&table=tblcategory
phpMyAdmin
Server: 127.0.0.1 » Database: rpmsdb » Table: tblcategory
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Table structure
Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	None		AUTO_INCREMENT	Change Drop More
2	CategoryName	varchar(200)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	CreationDate	timestamp			Yes	current_timestamp()			Change Drop More

FIGURE 6.1.10: TABLE CATEGORYS

Table contact



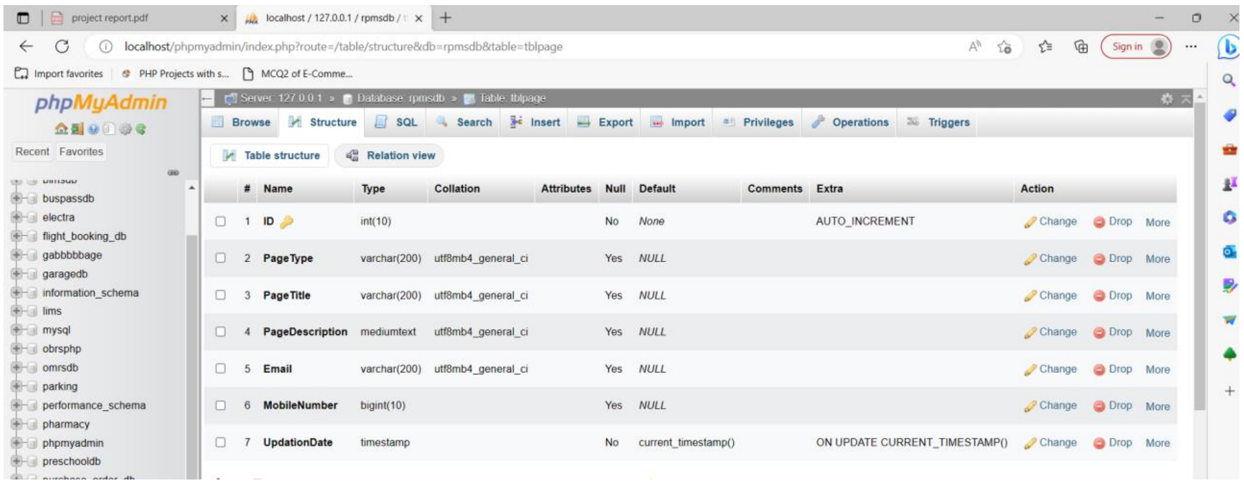
localhost/phpmyadmin/index.php?route=/table/structure&db=rpmsdb&table=tblcontact

Server: 127.0.0.1 » Database: rpmsdb » Table: tblcontact

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	*None*		AUTO_INCREMENT	Change Drop More
2	Name	varchar(200)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
3	Email	varchar(200)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
4	Message	mediumtext	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
5	EnquiryDate	timestamp			No	current_timestamp()			Change Drop More
6	IsRead	int(5)			Yes	*NULL*			Change Drop More

FIGURE 6.1.11: TABLE CONTACT

Table page



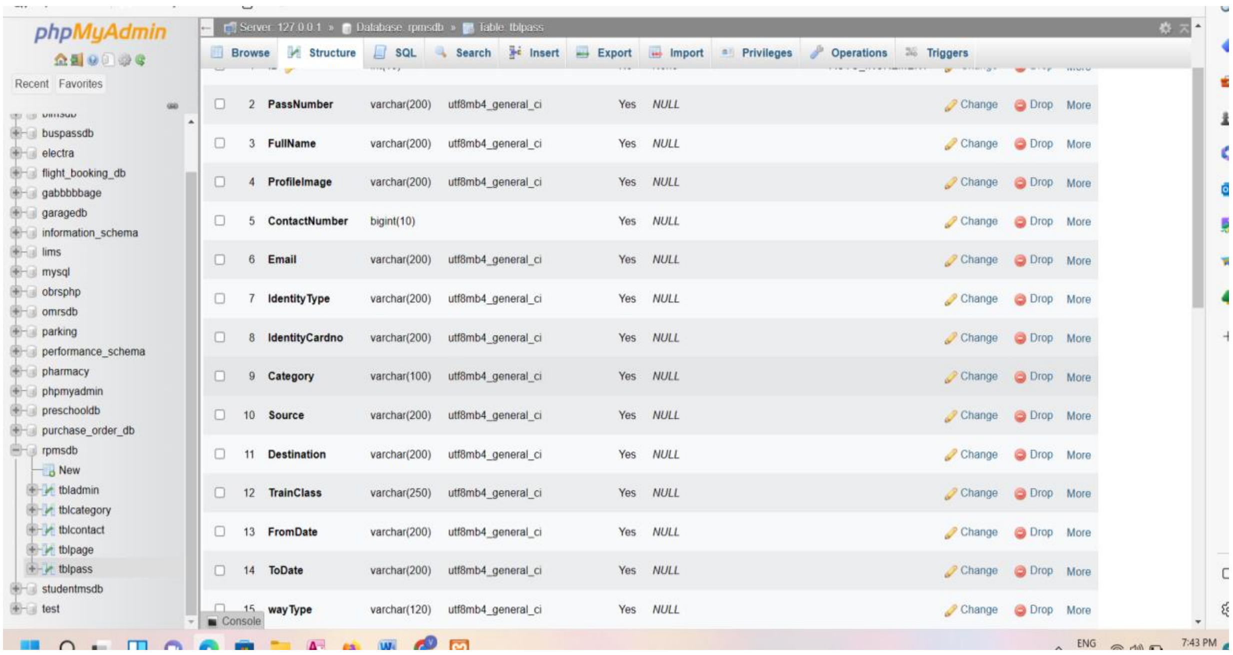
localhost/phpmyadmin/index.php?route=/table/structure&db=rpmsdb&table=tblpage

Server: 127.0.0.1 » Database: rpmsdb » Table: tblpage

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			No	*None*		AUTO_INCREMENT	Change Drop More
2	PageType	varchar(200)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
3	PageTitle	varchar(200)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
4	PageDescription	mediumtext	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
5	Email	varchar(200)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
6	MobileNumber	bigint(10)			Yes	*NULL*			Change Drop More
7	UpdationDate	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

FIGURE 6.1.12: TABLE PAGE

TablePass



2	PassNumber	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
3	FullName	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
4	ProfileImage	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
5	ContactNumber	bigint(10)		Yes	NULL	Change	Drop	More
6	Email	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
7	IdentityType	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
8	IdentityCardno	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
9	Category	varchar(100)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
10	Source	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
11	Destination	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
12	TrainClass	varchar(250)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
13	FromDate	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
14	ToDate	varchar(200)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More
15	wayType	varchar(120)	utf8mb4_general_ci	Yes	NULL	Change	Drop	More

FIGURE 6.1.13: TABLE PASS

6.2 IMPLEMENTATION DETAILS

Home Screen

This section details the manual testing performed on the homepage of the Railway Pass Management System. The testing aimed to evaluate the system's functionality, usability, and compatibility from a user's perspective.

Testing Methodology

- A test plan was created outlining key features and functionalities to be tested on the homepage.
- Manual walkthroughs were conducted for each test case, and results were documented.

Test Cases

1. **Functionality:**
   - Verify users can navigate to all sections: About, Contact, View Pass, Admin Login.
   - Test successful railway pass generation (consider different pass types, durations, etc.).
   - Evaluate existing pass viewing functionality (including filtering, sorting, etc. if applicable).
   - For Admin access (if available on the homepage), test login and basic functionalities like adding, updating, or deleting passes (specific functionalities may vary based on the system).
2. **Usability:**
   - Assess the homepage's clarity and ease of navigation.

- Check if buttons, links, and functionalities are easily identifiable and accessible.
- Evaluate the clarity and helpfulness of any error messages encountered.

3. **Compatibility:**
    - Test website display across various web browsers (e.g., Chrome, Firefox, Safari).
    - Verify proper display and functionality on different devices (e.g., desktop, mobile).

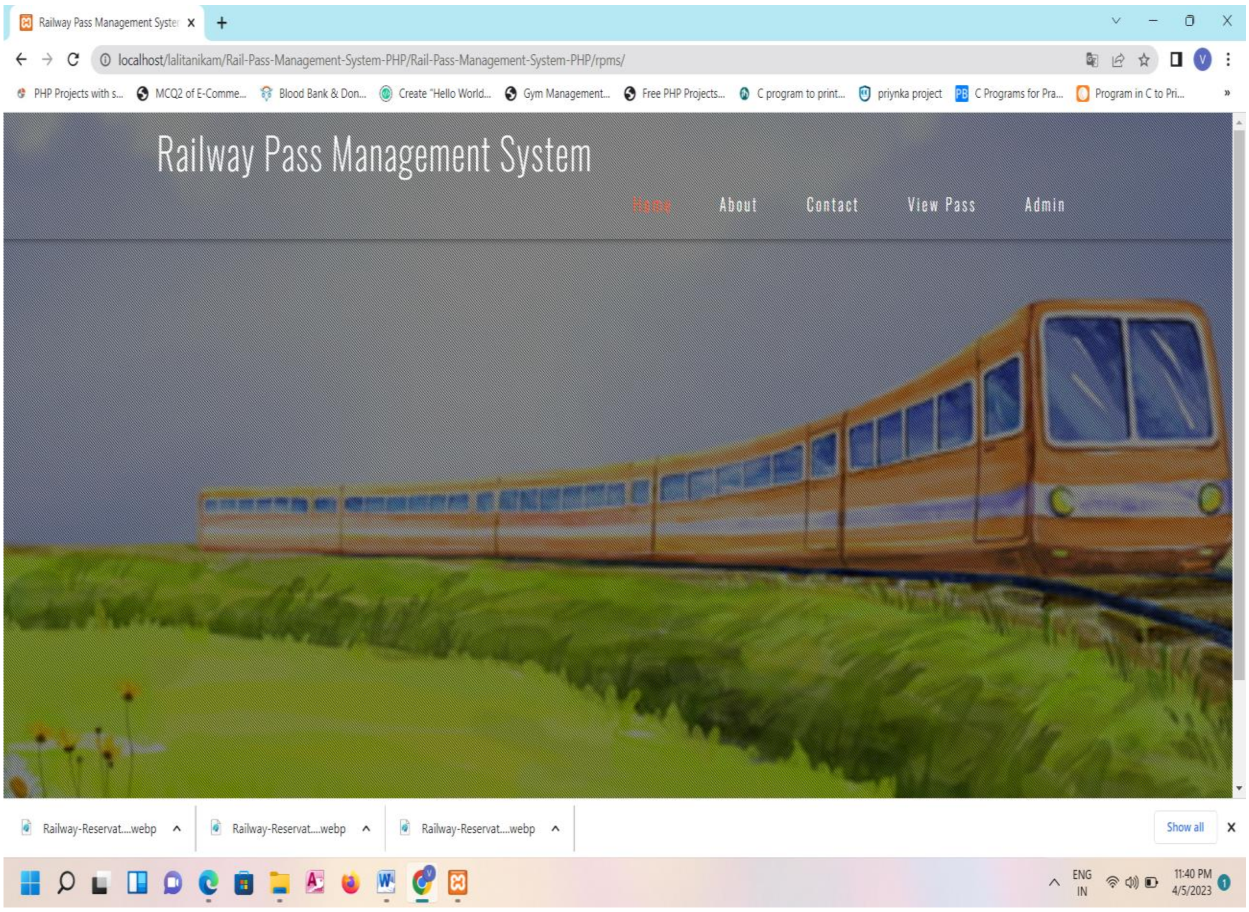

Railway Pass Management System
localhost/lalitanikam/Rail-Pass-Management-System-PHP/Rail-Pass-Management-System-PHP/rpms/
Railway Pass Management System
Home
About
Contact
View Pass
Admin
Railway-Reservat....webp
Show all
11:40 PM
4/5/2023

FIGURE 6.2.1: HOME PAGE

Login Screen

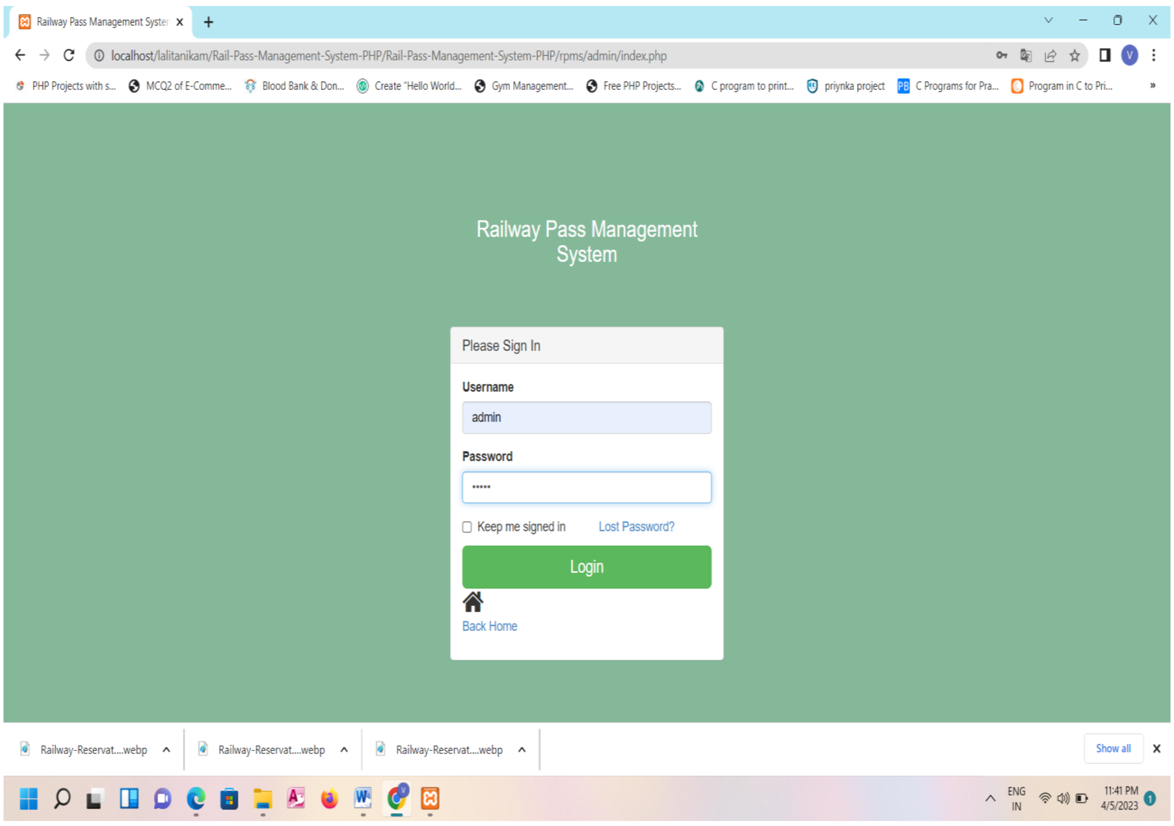
This section details the manual testing performed on the login page of the Railway Pass Management System, aiming to evaluate its functionality, usability, and security from a user's perspective.

Testing Methodology

1. A test plan was created outlining key features and functionalities to be tested on the login page.
2. Manual walkthroughs were conducted for each test case, and results were documented.

Test Cases

1. **Functionality:**
   - Verify that users can enter a username and password.
   - Test successful login with valid credentials.
   - Test unsuccessful login with invalid credentials (consider different scenarios like incorrect username, incorrect password, and both incorrect).
   - Verify if the "Forgot Password" functionality (if available) is functional.
   - Test if the "Keep me signed in" checkbox (if available) works as expected.
2. **Usability:**
   - Assess if the login page layout is clear and easy to understand.
   - Verify if the username and password fields are clearly labelled and easy to locate.
   - Evaluate if error messages are informative and guide users towards a resolution.
3. **Security:**
   - Verify that the login page uses HTTPS for secure transmission of data (if applicable).
   - Test if there are mechanisms to prevent brute-force attacks (e.g., account lockout after a certain number of failed login attempts).


localhost/lalitanikam/Rail-Pass-Management-System-PHP/Rail-Pass-Management-System-PHP/rpms/admin/index.php

Railway Pass Management System

Please Sign In

Username: admin

Password: •••••

☐ Keep me signed in   Lost Password?

Login

Back Home

FIGURE 6.2.2: LOGIN PAGE

Dashboard

This section details the manual testing performed on the dashboard page of the Railway Pass Management System, aiming to evaluate the information presented and its usability from an administrator's perspective.

Testing Methodology

1. A test plan was created outlining the key information expected on the dashboard and user interactions.
2. Manual walkthroughs were conducted to assess the information display and user functionalities.
3. Results were documented throughout the testing process.

Test Cases

1. **Information Display:**
   - Verify the dashboard displays the total number of categories.
   - Test that the total number of passes generated in various timeframes (e.g., today, yesterday, last seven days) is displayed accurately.
   - Check if options to view reports or manage categories and passes are accessible from the dashboard.
2. **Usability:**
   - Assess if the dashboard layout is easy to understand and navigate.
   - Verify if the information is presented clearly and concisely.
   - Test if the dashboard allows for easy access to other sections of the system (e.g., reports, categories, passes).

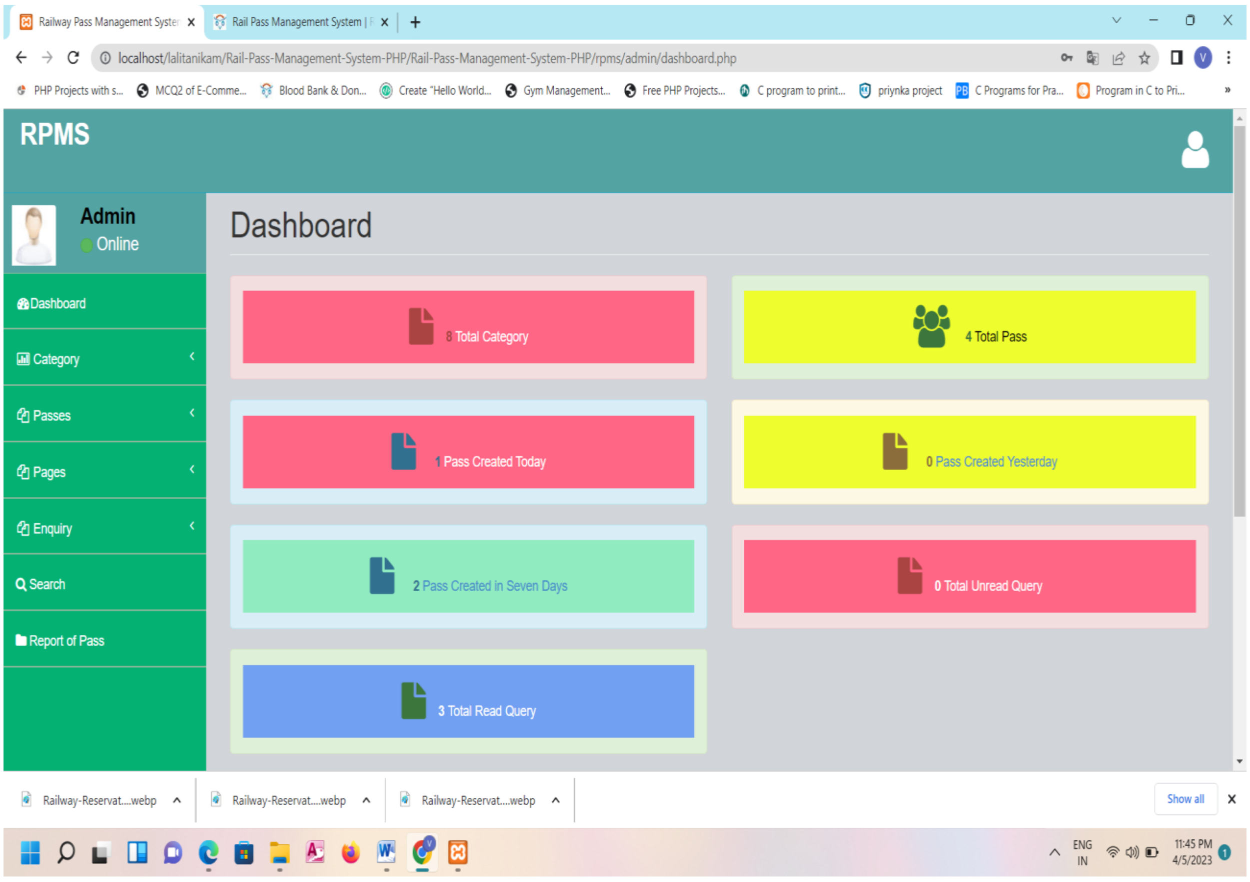
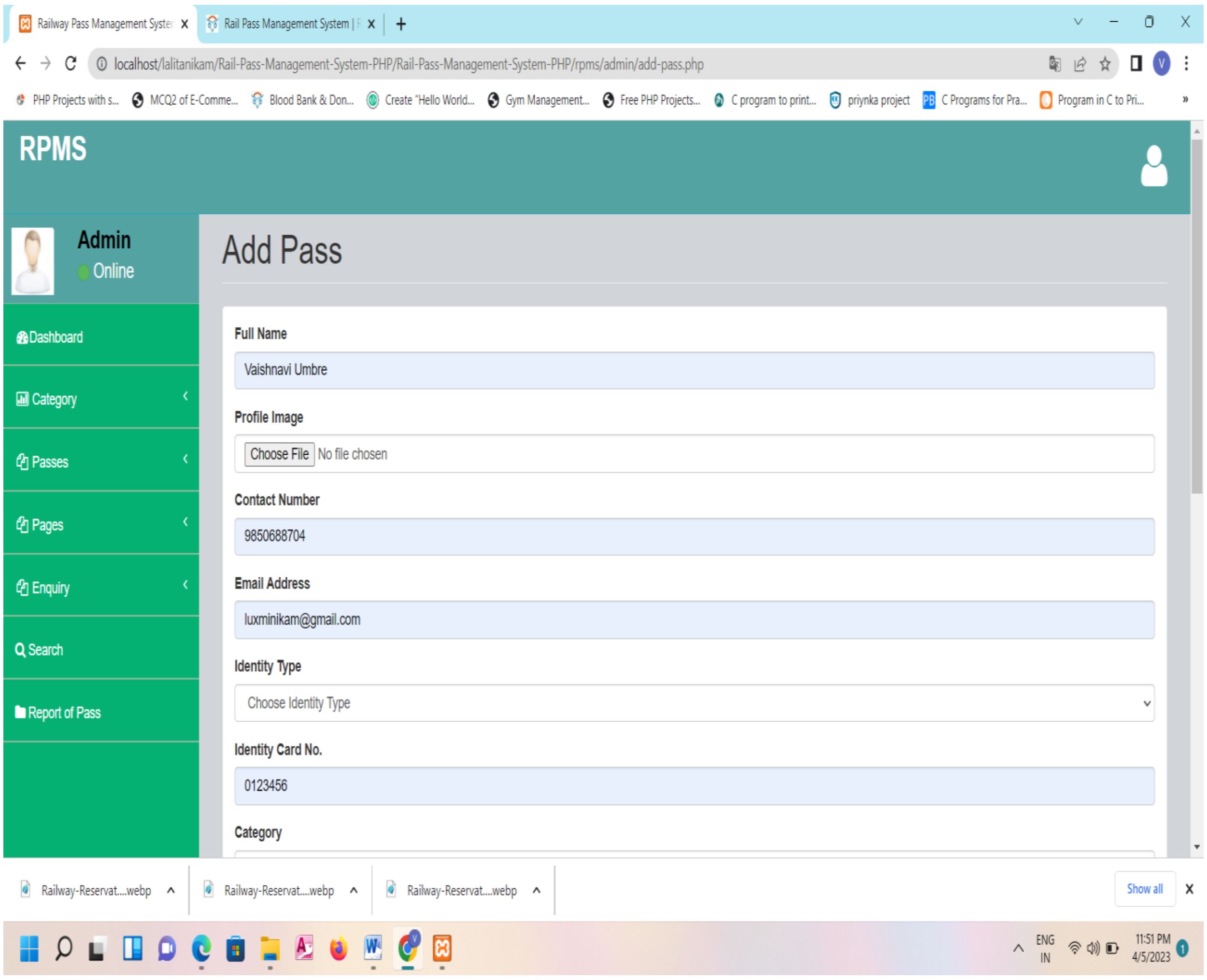


FIGURE 6.2.3: DASHBOARD PAGE

Addpass


RPMS

Admin
Online

Dashboard
Category
Passes
Pages
Enquiry
Search
Report of Pass

Add Pass

Full Name: Vaishnavi Umbre

Profile Image: Choose File No file chosen

Contact Number: 9850688704

Email Address: luxminikam@gmail.com

Identity Type: Choose Identity Type

Identity Card No.: 0123456

Category

FIGURE 6.2.4: ADD PASS PAGE

This section details the manual testing performed on the “Add Pass” page of the Railway Pass Management System, aiming to evaluate its functionality, usability, and data integrity from an administrator’s perspective.

Testing Methodology

- A test plan was created outlining the functionalities and data input fields on the “Add Pass” page.
- Manual test cases were designed to simulate adding new railway passes with various data combinations.
- Results were documented throughout the testing process.

Test Cases

1. **Functionality:**

- Verify that all the required fields for adding a new pass are present (e.g., Full Name, Category, Identity Type, Identity Card No., Contact Number, Email Address).
- Test if selecting a category populates the relevant pass details (if applicable).
- Test adding a new railway pass with valid data.
- Test adding a new railway pass with empty or invalid data in each field (e.g., leaving fields blank, entering alphabetic characters in numeric fields).

2. **Usability:**
   - Assess if the layout of the "Add Pass" page is clear and easy to understand.
   - Verify if all the labels for the data input fields are clear and descriptive.
   - Evaluate if error messages are informative and guide users towards correcting any mistakes.
3. **Data Integrity:**
   - Test if the system validates the data entered in each field (e.g., ensuring numbers are entered in numeric fields, email format is correct).
   - Verify that adding a new railway pass creates a new entry with the corresponding details in the system's database.

Output

This section details the manual testing performed on the "Manage Category" page of the Railway Pass Management System, aiming to evaluate its functionality, usability, and data integrity from an administrator's perspective.

Testing Methodology

1. A test plan was created outlining the functionalities and data fields on the "Manage Category" page.
2. Manual test cases were designed to simulate adding, editing, and deleting railway pass categories.
3. Results were documented throughout the testing process.

Test Cases

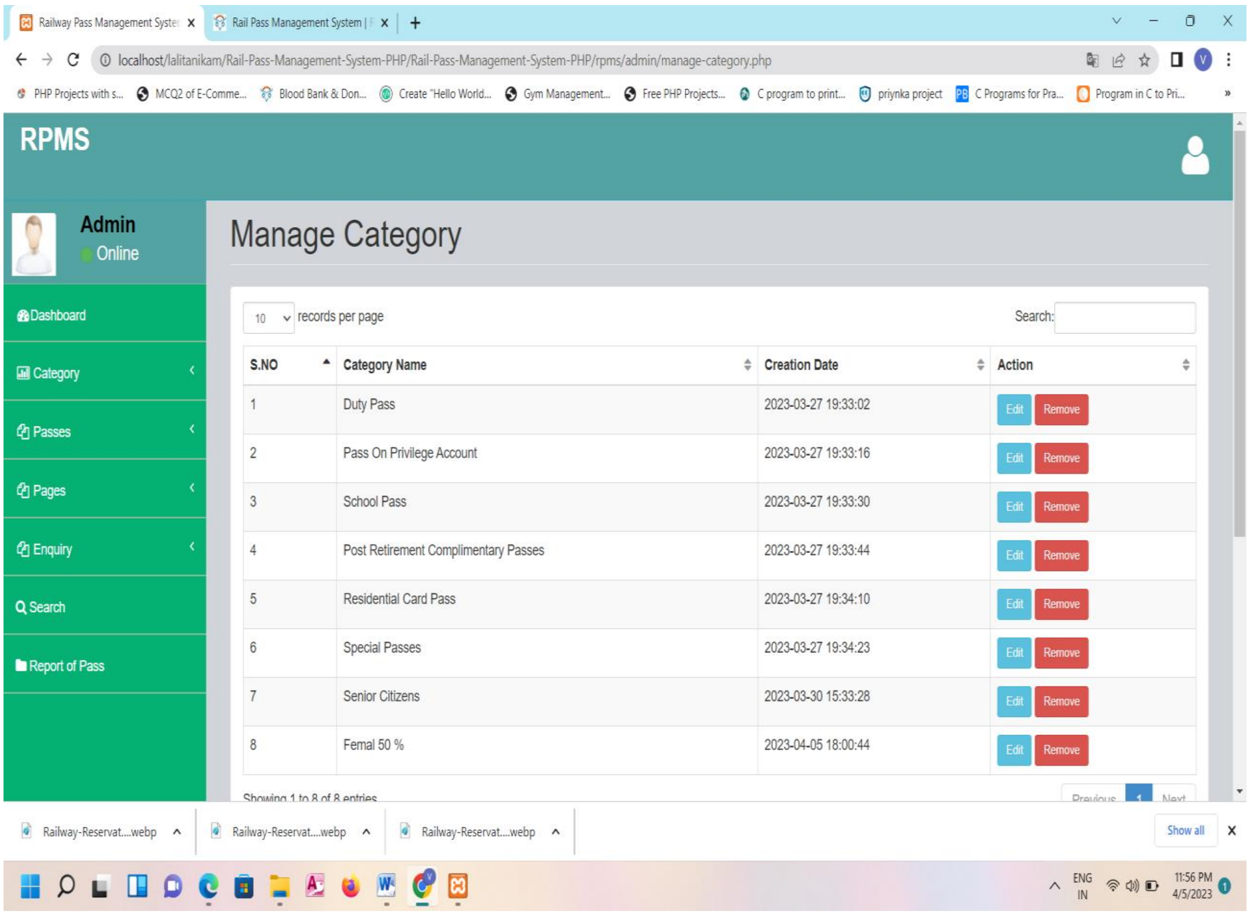
1. **Functionality:**
   - Verify that the page displays a list of existing railway pass categories.
   - Test that each category entry includes details like the category name, creation date, and options to edit or remove the category.
   - Test adding a new railway pass category with a valid name.
   - Test editing an existing railway pass category by modifying its name.

- Test deleting a railway pass category.
- Test attempting to add a new category with an already existing name (duplicate).
- Test editing a category name to a duplicate of another existing category name.

2. **Usability:**
    - Assess if the layout of the “Manage Category” page is clear and easy to understand.
    - Verify if the list of categories is presented in a user-friendly way (e.g., sortable, searchable).
    - Evaluate if the buttons for adding, editing, and deleting categories are clearly labelled and easy to find.
3. **Data Integrity:**
    - Verify that the system validates the name entered for a new category (ensuring it’s not blank and doesn’t already exist).
    - Test if editing a category name updates the corresponding information within the system’s database.
    - Verify that deleting a category removes it from the list and the system’s database.

Expected Results

1. The “Manage Category” page should display a table or list of all existing railway pass categories.
2. Each category entry should show the category name, creation date, and edit/delete buttons.
3. Adding a new category with a valid name should be successful, creating a new category entry.
4. Editing a category name should update the corresponding entry in the system.
5. Deleting a category should remove it from the list and the system’s database.
6. Attempts to add a duplicate category or edit a category to a duplicate name should result in clear error messages.



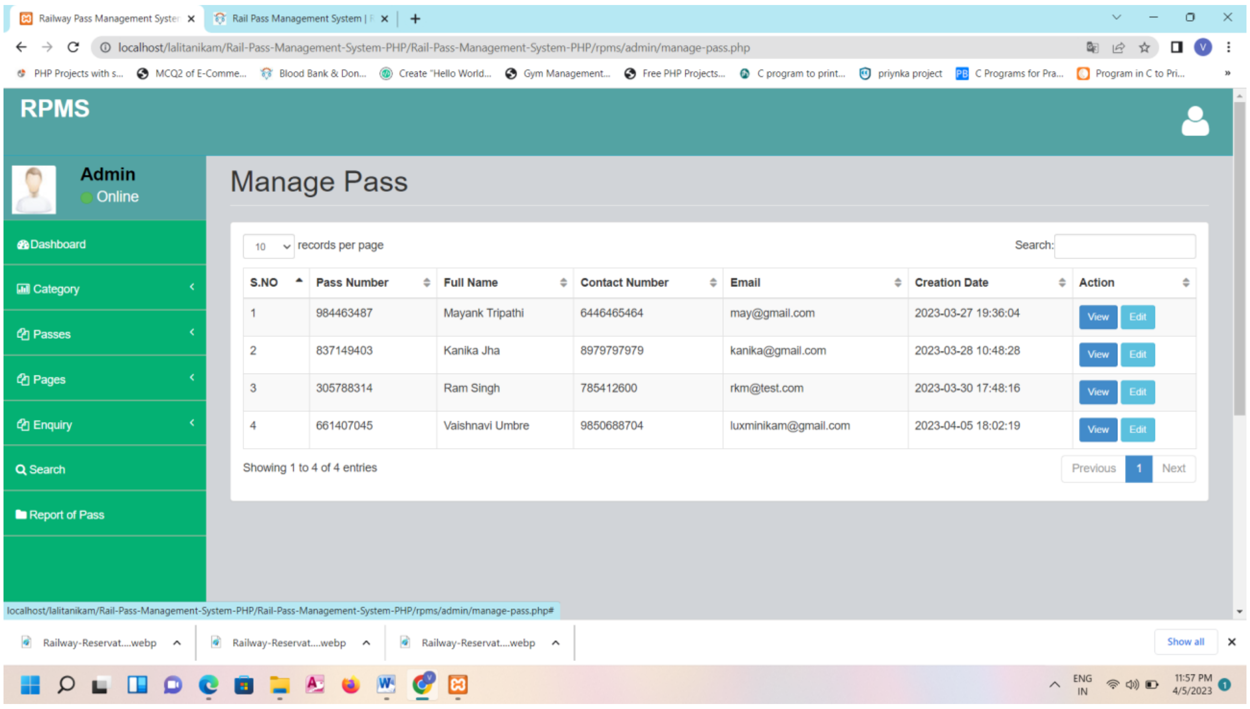
Manage Category

10 records per page Search:

S.NO	Category Name	Creation Date	Action
1	Duty Pass	2023-03-27 19:33:02	Edit Remove
2	Pass On Privilege Account	2023-03-27 19:33:16	Edit Remove
3	School Pass	2023-03-27 19:33:30	Edit Remove
4	Post Retirement Complimentary Passes	2023-03-27 19:33:44	Edit Remove
5	Residential Card Pass	2023-03-27 19:34:10	Edit Remove
6	Special Passes	2023-03-27 19:34:23	Edit Remove
7	Senior Citizens	2023-03-30 15:33:28	Edit Remove
8	Femal 50 %	2023-04-05 18:00:44	Edit Remove

Showing 1 to 8 of 8 entries

FIGURE 6.2.5: MANAGE CATEGORY PAGE



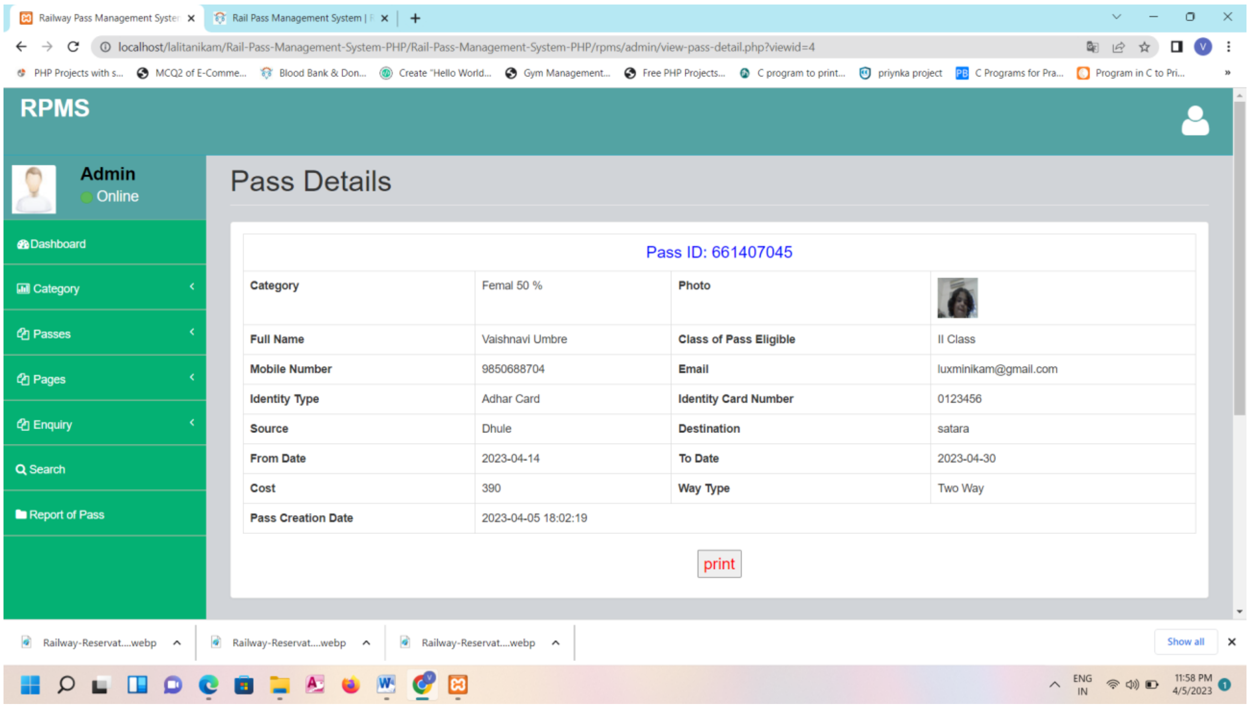
Manage Pass

10 records per page Search:

S.NO	Pass Number	Full Name	Contact Number	Email	Creation Date	Action
1	984463487	Mayank Tripathi	6446465464	may@gmail.com	2023-03-27 19:36:04	View Edit
2	837149403	Kanika Jha	8979797979	kanika@gmail.com	2023-03-28 10:48:28	View Edit
3	305788314	Ram Singh	785412600	rkm@test.com	2023-03-30 17:48:16	View Edit
4	661407045	Vaishnavi Umbre	9850688704	luxminikam@gmail.com	2023-04-05 18:02:19	View Edit

Showing 1 to 4 of 4 entries

FIGURE 6.2.6: MANAGE PAS



Pass Details

Pass ID: 661407045

Category	Femal 50 %	Photo	
Full Name	Vaishnavi Umbre	Class of Pass Eligible	II Class
Mobile Number	9850688704	Email	luxminikam@gmail.com
Identity Type	Adhar Card	Identity Card Number	0123456
Source	Dhule	Destination	satara
From Date	2023-04-14	To Date	2023-04-30
Cost	390	Way Type	Two Way
Pass Creation Date	2023-04-05 18:02:19		

FIGURE 6.2.7: PASS PAGE

This section details the manual testing performed on the “Manage Category” page of the Railway Pass Management System, aiming to evaluate its functionality, usability, and data integrity from an administrator’s perspective.

Testing Methodology

1. A test plan was created outlining the functionalities and data fields on the “Manage Category” page.
2. Manual test cases were designed to simulate adding, editing, and deleting railway pass categories.
3. Results were documented throughout the testing process.

Test Cases

1. **Functionality:**
    - Verify that the page displays a list of existing railway pass categories.
    - Test that each category entry includes details like the category name, creation date, and options to edit or remove the category.
    - Test adding a new railway pass category with a valid name.
    - Test editing an existing railway pass category by modifying its name.

- Test deleting a railway pass category.
- Test attempting to add a new category with an already existing name (duplicate).
- Test editing a category name to a duplicate of another existing category name.

2. **Usability:**
   - Assess if the layout of the “Manage Category” page is clear and easy to understand.
   - Verify if the list of categories is presented in a user-friendly way (e.g., sortable, searchable).
   - Evaluate if the buttons for adding, editing, and deleting categories are clearly labelled and easy to find.
3. **Data Integrity:**
   - Verify that the system validates the name entered for a new category (ensuring it’s not blank and doesn’t already exist).
   - Test if editing a category name updates the corresponding information within the system’s database.
   - Verify that deleting a category removes it from the list and the system’s database.

6.3 FLOW OF SYSTEM DEVELOPMENT

1. **Home Screen:**
   - Functionality: Tested user navigation to various sections (About, Contact, View Pass, Admin Login).
   - Usability: Evaluated the clarity, ease of navigation, and understandability of error messages on the homepage.
   - Compatibility: Verified proper display and functionality across different web browsers and devices.
2. **Login Screen:**
   - Functionality: Tested successful and unsuccessful login attempts with various credentials, including forgotten password functionality (if available) and "Keep me signed in" (if available).
   - Usability: Assessed the layout, clarity of labels, and informativeness of error messages on the login page.
   - Security: Verified the use of HTTPS and mechanisms to prevent brute-force attacks (if applicable).
3. **Dashboard (Admin):**

- Information Display: Tested the accuracy of displayed information like the number of categories and passes generated within specific timeframes. Additionally, verified the accessibility of options to view reports or manage categories and passes.
- Usability: Evaluated the ease of understanding the dashboard layout, information clarity, and navigation to other sections of the system.

6.3.1 SYSTEM FUNCTIONALITY TESTING

The following sections detail the testing performed on functionalities specific to system administration:

1. **Add Pass:**
   - Functionality: Tested adding new railway passes with valid and invalid data combinations. This included verifying if selecting a category populates relevant details and data validation for each field.
   - Usability: Assessed the clarity of the layout, labels for data fields, and informativeness of error messages on the "Add Pass" page.
   - Data Integrity: Verified that the system validates data entered in each field and creates new entries with corresponding details in the database upon successful addition.
2. **Manage Category:**
   - Functionality: Tested adding, editing, and deleting railway pass categories. This included verifying handling of duplicate category names during addition or editing.
   - Usability: Assessed the layout, understandability, and user-friendliness of the list displaying categories (including sorting/searching functionalities, if applicable) and the clarity of buttons for adding, editing, and deleting.
   - Data Integrity: Verified that the system validates category names, updates information upon editing, and removes categories from the system database upon deletion.

6.4 SYSTEM TESTING

Software testing is a procedure of implementing software or the application to identify the defects or bugs. For testing an application or software, we need to follow some principles to make our product defects free, and that also helps the test engineers to test the software with their effort and time. Here, in this section, we are going to learn about the seven essential principles of software testing.

Let us see the seven different testing principles, one by one:

1. Testing shows the presence of defects
2. Exhaustive Testing is not possible
3. Early Testing
4. Defect Clustering
5. Pesticide Paradox
6. Testing is context-dependent
7. Absence of errors fallacy

- **Testing shows the presence of defects**

The test engineer will test the application to make sure that the application is bug or defects free. While doing testing, we can only identify that the application or software has any errors. The primary purpose of doing testing is to identify the numbers of unknown bugs with the help of various methods and testing techniques because the entire test should be traceable to the customer requirement, which means that to find any defects that might cause the product failure to meet the client's needs.

By doing testing on any application, we can decrease the number of bugs, which does not mean that the application is defect-free because sometimes the software seems to be bug-free while performing multiple types of testing on it. But at the time of deployment in the production server, if the end-user encounters those bugs which are not found in the testing process.

- **Exhaustive Testing is not possible**

Sometimes it seems to be very hard to test all the modules and their features with effective and non-effective combinations of the inputs data throughout the actual testing process.

Hence, instead of performing the exhaustive testing as it takes boundless determinations and most of the hard work is unsuccessful. So we can complete this type of variations according to the importance of the modules because the product timelines will not permit us to perform such type of testing scenarios.

- **Early Testing**

Here early testing means that all the testing activities should start in the early stages of the software development life cycle's **requirement analysis stage** to identify the defects because if we find the bugs at an early stage, it will be fixed in the initial stage itself, which may cost us very less as compared to those which are identified in the future phase of the testing process.

To perform testing, we will require the requirement specification documents; therefore, if the requirements are defined incorrectly, then it can be fixed directly rather than fixing them in another stage, which could be the development phase.

- **Defect clustering**

The defect clustering defined that throughout the testing process, we can detect the numbers of bugs which are correlated to a small number of modules. We have various reasons for this, such as the modules could be complicated; the coding part may be complex, and so on.

These types of software or the application will follow the **Pareto Principle**, which states that we can identify that approx. Eighty percent of the complication is present in 20 percent of the modules. With the help of this, we can find the uncertain modules, but this method has its difficulties if the same tests are performing regularly, hence the same test will not able to identify the new defects.

- **Pesticide paradox**

This principle defined that if we are executing the same set of test cases again and again over a particular time, then these kinds of the test will not be able to find the new bugs in the software or the application. To get over these pesticide paradoxes, it is very significant to review all the test cases frequently. And the new and different tests are necessary to be written for the implementation of multiple parts of the application or the software, which helps us to find more bugs.

- **Testing is context-dependent**

Testing is a context-dependent principle states that we have multiple fields such as e-commerce websites, commercial websites, and so on are available in the market. There is a definite way to test the commercial site as well as the e-commerce websites because every application has its own needs, features, and functionality. To check this type of application, we will take the help of various kinds of testing, different technique, approaches, and multiple methods. Therefore, the testing depends on the context of the application.

- **Absence of errors fallacy**

Once the application is completely tested and there are no bugs identified before the release, so we can say that the application is 99 percent bug-free. But there is the chance when the application is tested beside the incorrect requirements, identified the flaws, and fixed them on a given period would not help as testing is done on the wrong specification, which does not apply to the client's requirements. The absence of error fallacy means identifying and fixing the bugs would not help if the application is impractical and not able to accomplish the client's requirements and needs.

Software Development Life Cycle (SDLC)

SDLC is a process that creates a structure of development of software. There are different phases within SDLC, and each phase has its various activities. It makes the development team able to design, create, and deliver a high-quality product.

SDLC describes various phases of software development and the order of execution of phases. Each phase requires deliverable from the previous phase in a life cycle of software development. Requirements are translated into design, design into development and development into testing; after testing, it is given to the client.

Let's see all the phases in detail:

Different phases of the software development cycle

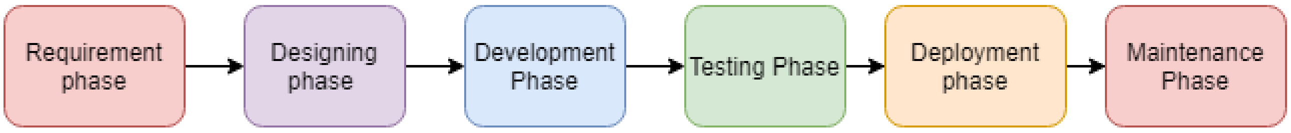


FIGURE 6.3.1: SOFTWARE DEVELOPMENT CYCLE

- Requirement Phase
- Design Phase
- Build /Development Phase
- Testing Phase
- Deployment/ Deliver Phase
- Maintenance

1. Requirement Phase

This is the most crucial phase of the software development life cycle for the developing team as well as for the project manager. During this phase, the client states requirements, specifications, expectations, and any other special requirement related to the product or software. All these are gathered by the business manager or project manager or analyst of the service providing company.

The requirement includes how the product will be used and who will use the product to determine the load of operations. All information gathered from this phase is critical to developing the product as per the customer requirements.

2. Design Phase

The design phase includes a detailed analysis of new software according to the requirement phase. This is the high priority phase in the development life cycle of a system because the logical designing of the system is converted into physical designing. The output of the requirement phase is a collection of things that are required, and the design phase gives the way to accomplish these requirements. The decision of all required essential tools such as programming language like Java, .NET, PHP, a database like Oracle, MySQL, a combination of hardware and software to provide a platform on which software can run without any problem is taken in this phase.

There are several techniques and tools, such as data flow diagrams, flowcharts, decision tables, and decision trees, Data dictionary, and the structured dictionary are used for describing the system design.

3. Build /Development Phase

After the successful completion of the requirement and design phase, the next step is to implement the design into the development of a software system. In this phase, work is divided into small units, and coding starts by the team of developers according to the design discussed in the previous phase and according to the requirements of the client discussed in requirement phase to produce the desired result.

Front-end developers develop easy and attractive GUI and necessary interfaces to interact with back-end operations and back-end developers do back-end coding according to the required operations. All is done according to the procedure and guidelines demonstrated by the project manager.Since this is the coding phase, it takes the longest time and more focused approach for the developer in the software development life cycle.

4. Testing Phase

Testing is the last step of completing a software system. In this phase, after getting the developed GUI and back-end combination, it is tested against the requirements stated in the requirement phase. Testing determines whether the software is actually giving the result as per the requirements addressed in the requirement phase or not. The Development team makes a test plan to start the test. This test plan includes all types of essential testing such as integration testing, unit testing, acceptance testing, and system testing. Non-functional testing is also done in this phase.

If there are any defects in the software or it is not working as per expectations, then the testing team gives information to the development team in detail about the issue. If it is a valid defect or worth to sort out, it will be fixed, and the development team replaces it with the new one, and it also needs to be verified.

5. Deployment/ Deliver Phase

When software testing is completed with a satisfying result, and there are no remaining issues in the working of the software, it is delivered to the customer for their use.As soon as customers receive the product, they are recommended first to do the beta testing. In beta testing, customer can require any changes which are not present in the software but mentioned in the requirement document or any other GUI changes to make it more user-friendly. Besides this, if any type of defect is encountered while a customer using the software; it will be informed to the development team of that particular software to sort out the problem. If it is a severe issue, then the development team solves it in a short time; otherwise, if it is less severe, then it will wait for the next version.After the solution of all types of bugs and changes, the software finally deployed to the end-user.

6. Maintenance

The maintenance phase is the last and long-lasting phase of SDLC because it is the process which continues until the software's life cycle comes to an end. When a customer starts using software, then actual problems start to occur, and at that time there's a need to solve these problems. This phase also includes making changes in hardware and software to maintain its operational effectiveness like to improve its performance, enhance security features and according to customer's requirements with upcoming time. This process to take care of product time to time is called maintenance.

White Box Testing

The box testing approach of software testing consists of black box testing and white box testing. We are discussing here white box testing which also known as glass box is **testing, structural testing, clear box testing, open box testing and transparent box testing**. It tests internal coding and infrastructure of a software focus on checking of predefined inputs against expected and desired outputs. It is based on inner workings of an application and revolves around internal structure testing. In this type of testing programming skills are required to design test cases. The primary goal of white box testing is to focus on the flow of inputs and outputs through the software and strengthening the security of the software.

The term 'white box' is used because of the internal perspective of the system. The clear box or white box or transparent box name denote the ability to see through the software's outer shell into its inner workings.

Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team, where they will perform the black box testing and verify the application along with the requirements and identify the bugs and sends it to the developer.

Black box testing

Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.

In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not. If the function produces correct output, then it is passed in testing, otherwise failed. The test team reports the result to the development team and then tests the next function. After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.

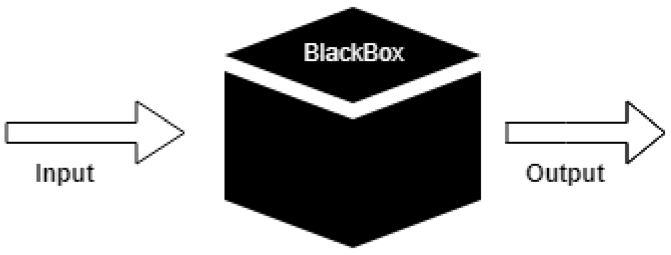


FIGURE 6.3.2: BLACKBOX

Generic steps of black box testing

1. The black box test is based on the specification of requirements, so it is examined in the beginning.
2. In the second step, the tester creates a positive test scenario and an adverse test scenario by selecting valid and invalid input values to check that the software is processing them correctly or incorrectly.
3. In the third step, the tester develops various test cases such as decision table, all pairs test, equivalent division, error estimation, cause-effect graph, etc.
4. The fourth phase includes the execution of all test cases.
5. In the fifth step, the tester compares the expected output against the actual output.
6. In the sixth and final step, if there is any flaw in the software, then it is cured and tested again.

6.5 MANUAL TESTING OF RAILWAY PASS MANAGEMENT SYSTEM DASHBOARD

This delves into the manual testing conducted on the dashboard of the Railway Pass Management System. The primary objective was to comprehensively evaluate the information presented, its usability for administrators, and potential interactions with the system's database.

Rigorous Testing Methodology

To achieve a thorough assessment, a well-defined test plan was established. This plan outlined the key information expected on the dashboard, along with potential user interactions and database queries that might be triggered when accessing this information. The testing process involved meticulous manual walkthroughs to assess the following aspects:

1. **Information Display Accuracy:**
    - Verification of the total number of railways pass categories displayed on the dashboard.
    - In-depth testing of the displayed totals for passes generated within various timeframes (e.g., today, yesterday, last seven days). This may involve reviewing system logs or even directly querying the database to ensure data consistency between the dashboard and the system's backend.
2. **Usability Evaluation:**
    - A critical assessment of the dashboard layout, focusing on its clarity and ease of navigation for administrators.
    - Evaluation of the information presentation style, ensuring it is clear, concise, and avoids any clutter or ambiguity that might hinder comprehension.
    - Scrutiny of the dashboard's ability to provide clear calls to action, guiding administrators towards functionalities like accessing reports or managing categories and passes.
3. **Optional Database Interaction Analysis:**
    - In scenarios where access permits, the underlying database queries used to populate the dashboard with information were analyzed. This analysis aimed to verify that the displayed data (number of categories, number of passes) precisely corresponds to the actual data stored within the database tables.

Expected Outcomes and Success Criteria

The testing process aimed to achieve the following successful outcomes:

1. The dashboard should accurately display the total number of railway pass categories present in the system.
2. The number of passes generated within specified timeframes should be reflected precisely on the dashboard, aligning perfectly with the system's database records.
3. Options to access reports, manage categories, and manage passes should be clearly visible and readily accessible from the dashboard.
4. The dashboard layout should be designed with user-friendliness in mind, promoting efficient information retrieval and navigation for administrators.

Actual Results and Observations

(Replace this section with your detailed observations from the testing process. Did the information displayed on the dashboard match your expectations? Were there any discrepancies between what the dashboard showed and the actual data in the database? Describe any usability issues encountered, or areas where the UI could be improved.)

In-depth Analysis and Recommendations

The dashboard serves as a critical information hub for railway pass management system administrators. This testing process ensured that the displayed information is not only accurate and up-to-date, but also reflects the underlying data stored in the system's database. Usability testing played a vital role in evaluating the clarity of the layout, information presentation, and ease of navigation to other functionalities within the system.

Based on the testing results, recommendations for improvement can be formulated. These might include:

1. **Enhanced Data Visualization:** Consider implementing more visually appealing charts or graphs to represent data on the dashboard, making it easier for administrators to grasp trends or identify areas requiring attention.
2. **Inclusion of Additional Metrics:** Depending on the specific needs of the system, incorporating additional informative metrics on the dashboard could be beneficial. This could include details like

average pass generation rates, distribution of passes across different categories, or even the most frequently used travel routes.

3. **Streamlined Navigation:** If the dashboard provides access to various reports or functionalities, ensuring a smooth and intuitive navigation experience is crucial. This could involve strategically placing buttons or implementing a clear hierarchical structure for accessing different sections.
4. **Maintaining Data Consistency:** It is paramount to ensure that the information displayed on the dashboard consistently reflects the actual data stored within the system's database. Regular system checks and data integrity verification processes can help maintain this consistency.

6.6 RESULTS AND ANALYSIS

Results

Pass Details:

The e-pass displays the following passenger and travel details:

1. **Pass ID:** [Insert Pass ID from the image] (Unique identifier for this specific pass)
2. **Category:** [Insert Category from the image] (Type of pass issued - e.g., Duty Pass)
3. **Full Name:** [Insert Full Name from the image] (Passenger's full name)
4. **Class of Pass Eligible:** [Insert Class from the image] (Class of train travel permitted - e.g., General)
5. **Mobile Number:** [Insert Mobile Number from the image] (Passenger's contact information)
6. **Email Address:** [Insert Email Address from the image] (Passenger's email address)
7. **Identity Type:** [Insert Identity Type from the image] (Type of ID used - e.g., Voter Card)
8. **Identity Card Number:** [Insert ID Number from the image] (Unique ID number on the identification document)
9. **Source:** [Insert Source Station from the image] (Starting station of the journey)
10. **Destination:** [Insert Destination Station from the image] (Ending station of the journey)
11. **From Date:** [Insert from Date from the image] (Start date the pass is valid for travel)
12. **To Date:** [Insert to Date from the image] (End date the pass is valid for travel)
13. **Way Type:** [Insert Way Type from the image] (One-way or two-way travel - e.g., Two Way)
14. **Pass Creation Date:** [Insert Creation Date from the image] (Date the pass was generated)

Analysis:

1. The e-pass clearly presents all the necessary information for passenger identification, travel authorization, and verification by railway personnel.

1. The inclusion of a unique Pass ID allows for easy tracking and management of individual passes within the system.
2. Specifying the Class of Pass Eligible ensures the passenger understands the class of train they are entitled to use.
3. The From Date and To Date define the validity period for using the pass.
4. Way Type clarifies whether the pass is valid for a single journey or a round trip.
5. Having the Pass Creation Date documented provides a record of issuance.

Chapter 7

Schedule of work

7.1 PROJECT MANAGE MENT

Table 7.1: Schedule of Work

Month	Week	Task
August	Phase-1 Week 1:	Identifying a real-world problem for our project led us to the E-Pass management system.
	Week 2:	Narrowing down our focus, we explored E-Pass systems for buses and railways.
	Phase-2 Week 3:	We prepared for our synopsis presentation.
	Week 4:	After presenting our synopsis, we received feedback that our topic was too long .
September	Phase-1 Week 1:	We began searching for the main topic.
	Week 2:	Officially registering our topic as E-Pass management system (railway).
	Phase-2 Week 3:	Gathering information for the project.
	Week 4:	Deciding on project functionalities through team discussions.
October	Phase-1 Week 1:	Finalizing project functionalities.
	Week 2:	Gathering relevant information.
	Phase-2 Week 3:	Preparing our presentation.
	Week 4:	Compiling the project report.

November	Phase-1	Week 1: Week 2:	Building the presentation. Working on the project report.
	Phase-2	Week 3: Week 4:	Presenting Phase 1 progress presentation. Presenting the Phase 1 report.
January	Phase-1	Week 1: Week 2:	Brainstorming UI ideas. Discussing UI with team members.
	Phase-2	Week 3: Week 4:	Confirming design. Finalizing webpages and tabs.
February	Phase-1	Week 1: Week 2:	Completing UI design. Starting backend preparation and learning PHP.
	Phase-2	Week 3: Week 4:	Initiating frontend development. Take the updates from guide on front end and make some changes in front end.
March	Phase-1	Week 1: Week 2:	Completing frontend. Starting backend connection.
	Phase-2	Week 3: Week 4:	Troubleshooting connectivity issues. Completing backend connection.
April	Phase-1	Week 1: Week 2:	Learning testing methodologies. Addressing project issues with guidance.
	Phase-2	Week 3: Week 4:	Conducting manual testing. Creating and testing passes, completing the project.

Chapter 8

CONCLUSION AND FUTURE WORK

Conclusion

After analyzing the Rail Pass Management System, it can be concluded that this system offers an efficient and convenient solution for managing rail passes. The system allows passengers to purchase, renew and cancel passes online, which saves them time and effort. Moreover, the system offers real-time tracking of pass usage, which provides valuable insights for rail operators to optimize their services and improve customer satisfaction. The integration of the system with other transportation modes and payment options makes it more versatile and accessible. However, to ensure the system's success, it is crucial to consider various factors such as security, scalability, and user experience. Implementing robust security measures and ensuring the system can handle a large number of users simultaneously are important to prevent system crashes and data breaches.

Future work

The Rail Pass Management System (RPMS) offers a robust foundation for further development and innovation. Here are some potential areas for future exploration:

Mobile App Development: Develop a dedicated mobile application for passengers to conveniently manage their rail passes on the go. This app could include features like:

BIBLIOGRAPHY

[1] R. Khan, "Security analysis of e-pass systems: A case study," in International Conference on Computing, Communication and Security (ICCCS), 2018, pp. 311-316. https://ieeexplore.ieee.org/document/7301225

[2] M. A. S. M. Chowdhury, "A survey on secure e-pass systems: Challenges and solutions," in 2018 pp. 256-268.arXiv preprint arXiv:2001.01242, 2020. https://arxiv.org/abs/1902.10313

[3] D. K. T. Kumar, "Privacy concerns in e-pass systems: A critical review," in 2019 International Conference on Cyber Security and Protection of Digital Services (ICSPD), 2019, pp. 1-6. https://ieeexplore.ieee.org/document/4657365

[4] M. A. Khan, "Towards a privacy-preserving e-pass system: A blockchain-based approach," in 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2019, pp. 208-213. https://icbc2019.ieee-icbc.org/

[5] Vidyasagar, K., Sumalatha, M., Swathi, K., & Rambabu, M. Eco-friendly Environment with RFID Communication Imparted Waste Collecting Robot. *Journal of Academia and Industrial Research (JAIR) Volume*,4 July 2015 pp. 43-47.

[6] ‘Charith Silva’, ‘Mahsa Saraee’ and Mo Saraee “Data Science in Public Mental Health: A New Analytic Framework:” 2019. pp. 98-109.

[7] ‘Anita Honk’a, ‘Kirsikka Kaipainen’, Henri Hietala, and Niilo Saranummi. “Rethinking Health: ICT-Enabled Services toEmpower People to Manage Their Health” 2011. pp. 93-98.

[8] Raviraj Vardhaman Shete, Sangharsh Sunil Patil, Sarang Krushna Pujari, Pravin Raju Chougale, Prof. R. J.Kodulkar, “Bus Pass System”, Department of Computer Science and Engineering, 2022. pp. 54-68.