RAILWAY TICKET MANAGEMENT SYSTEM

A Mini Project Report

submitted in partial fulfillment of requirements for the award of the degree

Bachelor of Engineering (B.E)

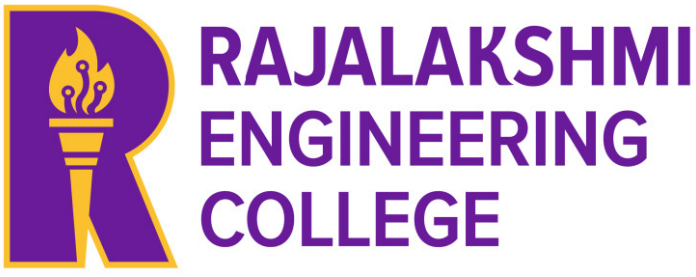
in

Computer Science Engineering and Cyber Security

by

SIVASANKARI T (Reg. No: 241901108)

SUBATHRA DEVI R (Reg. No: 241901113)



DEPARTMENT OF COMPUTER SCIENCE ENGINEERING AND CYBER SECURITY

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

November 2025

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

AND CYBER SECURITY

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

2025–2026

CERTIFICATE

This is to certify that the mini project report entitled **" RAILWAY TICKET AND MANAGEMENT SYSTEM"** submitted as part of the curriculum requirements for the Bachelor of Engineering (B.E) degree affiliated to Anna University, is a bonafide record of the project work carried out by **SIVASANKARI T** (Reg. No: 241901108), **SUBATHRA DEVI R** (Reg. No: 241901113) under the supervision of Ms. R. Rupmala, Assistant Professor, Department of Computer Science Engineering and Cyber Security, Rajalakshmi Engineering College, Chennai.

This report has not been submitted to any other University or Institute for any purpose and represents original work carried out by the students.

Ms. R. Rupmala
Assistant Professor
Department of Computer Science
Engineering and Cyber Security
Rajalakshmi Engineering College

The mini project report is submitted for the viva-voce examination to be held on .....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We hereby declare that the mini project report **RAILWAY TICKET MANAGEMENT SYSTEM**, submitted as part of the curriculum requirements for the Bachelor of Engineering (B.E) degree affiliated to Anna University, is a bonafide work carried out by us under the supervision of Ms. R. Rupmala, Assistant Professor, Department of Computer Science Engineering and Cyber Security, Rajalakshmi Engineering College, Chennai.

This submission represents our ideas in our own words, and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources.

We also declare that we have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact, or source in our submission. We understand that any violation of the above will be grounds for disciplinary action by the institute and/or the University and may also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been obtained. This report has not previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Rajalakshmi Engineering College,
Chennai
November 2025

SIVASANKARI T
SUBATHRA DEVI R

ABSTRACT

The Railway Reservation System is a comprehensive web-based application designed to modernize and streamline the train ticket booking process. Built using Jakarta Servlets, MySQL database, and modern web technologies, this system provides a secure, efficient, and user-friendly platform for railway ticket reservations.

The system implements enterprise-level security features including JWT-based authentication, two-factor authentication using TOTP, and password hashing with BCrypt. It offers real-time train search capabilities, secure ticket booking with QR code generation, PNR status checking, and comprehensive booking management features.

Key technological achievements include the implementation of connection pooling for database optimization, responsive web design using Bootstrap, comprehensive logging with SLF4J and Logback, and automated database management tools. The system follows MVC architecture principles and incorporates modern software engineering practices for maintainability and scalability.

Keywords: Railway Reservation, Web Application, Jakarta Servlets, MySQL, JWT Authentication, TOTP, QR Code, Real-time Booking

ACKNOWLEDGEMENT

We like to convey our sincere appreciation to all who have supported and mentored us during the successful completion of this project work.

We express our profound gratitude to **Mr. Benedict J.N.**, Associate Professor (SG) and Head of the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for furnishing us with essential resources, support, and an enabling environment to execute this project.

We express our profound gratitude to **Ms. R. Rupmala**, Assistant Professor in the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for her unwavering support, invaluable insights, and collaboration throughout our endeavor.

We would like to convey our gratitude to our faculty members and colleagues for their valuable feedback and encouragement.

We express our gratitude to our families and friends for their steadfast support, patience, and encouragement, which were instrumental in the effective execution of this seminar.

SIVASANKARI T - 241901108

SUBATHRA DEVI R - 241901113

Contents

List of Figures

List of Tables

Listings

Chapter 1: INTRODUCTION

1.1 Evolution of Online Booking Current Technologies

The evolution of online reservation systems has transformed significantly over the last two decades. Initially, these systems were limited to desktop-based applications with minimal connectivity and functionality. However, with the advent of web technologies, cloud computing, and mobile platforms, modern systems have become more accessible, scalable, and user-friendly. Railway booking platforms, in particular, have adopted advanced technologies to ensure secure, efficient, and seamless user experiences.

Modern railway systems integrate multiple layers of functionality to improve performance and security. Authentication mechanisms such as token-based authentication (JWT/OAuth), multi-factor authentication (MFA), and biometric verification enhance user identity protection. Databases are designed using relational models to ensure transaction integrity, supported by techniques like connection pooling, replication for high availability, and caching for faster response times. From a design perspective, user interfaces now follow responsive layouts for accessibility across devices, employ Progressive Web App (PWA) standards, and support real-time data updates through WebSocket connections. Moreover, adherence to accessibility standards like WCAG ensures inclusivity for all users.

1.2 Security in Online Booking Systems

Security plays a crucial role in maintaining the trust and reliability of online booking platforms. Data protection measures such as encryption of sensitive information, use of secure transmission protocols (HTTPS/TLS), and input validation help safeguard user data from breaches and unauthorized access. To prevent attacks such as SQL injection, systems enforce strict sanitization of inputs and compliance with standards like PCI DSS.

In terms of authentication security, strong password policies, two-factor authentication mechanisms, and session timeout settings are implemented to prevent unauthorized access. Additionally, account lockout policies act as a safeguard against brute-force attacks. By combining these security practices, online railway booking systems ensure a safe, efficient, and reliable

experience for users while maintaining data integrity and system resilience.

1.3 Impact and Future Scope of Online Railway Systems

The introduction of digital railway booking systems has greatly enhanced convenience, transparency, and efficiency in ticketing operations. Passengers can now check availability, book tickets, and receive real-time updates from anywhere, eliminating the need for long queues at counters. Additionally, integration with payment gateways and digital wallets has simplified financial transactions, making the entire process faster and more user-centric.

Looking ahead, the future of railway reservation systems lies in artificial intelligence (AI), machine learning (ML), and predictive analytics. These technologies can help forecast travel demand, personalize user recommendations, and detect fraudulent activities in real time. Moreover, the adoption of blockchain for ticket verification and cloud-native architectures for scalability will further strengthen the reliability and security of online booking platforms. Hence, the continuous evolution of these systems will play a key role in shaping the future of smart and sustainable transportation.

Chapter 2: SYSTEM ARCHITECTURE

2.1 Overall Architecture

The **Railway Reservation System** follows a **Model-View-Controller (MVC)** architecture pattern implemented using **Jakarta Servlets**. This design ensures modularity, scalability, and easy maintenance by separating the application into three main layers: Model, View, and Controller.

2.2 System Architecture Diagram

2.2.1 Layer Representation

The system architecture consists of three major interconnected layers as shown in Table **??**. Each layer is responsible for distinct functionalities, ensuring modularity and scalability of the overall design.

Presentation Layer	**Business Layer**	**Data Layer**
JavaScript	Servlets	MySQL
Bootstrap	Services	DAOs
Responsive UI	Filters	Models
	Utilities	Connection Pooling

Table 2.1 Representation of the System Architecture Layers

2.2.2 Layer Explanation

Presentation Layer

The presentation layer is responsible for user interaction and dynamic content display. It is developed using **JavaScript** and **Bootstrap 5**, ensuring a responsive and visually appealing design for users across different devices.

Business Layer

The business layer serves as the core logic controller that manages data flow between the frontend and backend. It includes key components such as **Servlets**, **Filters**, and **Services**, which together implement the core business logic and handle user requests efficiently.

Data Layer

The data layer focuses on data persistence and database management. It uses **MySQL** along with **Data Access Objects (DAOs)** and connection pooling mechanisms to ensure optimized database performance and secure access to stored information.

2.2.3 Component Architecture

Presentation Layer

This layer uses frontend technologies such as **JavaScript** and **Bootstrap 5** to build an interactive user interface. It handles client-side logic, including form validation, API calls, and user interactions, following a mobile-first design approach to ensure cross-browser compatibility.

Business Logic Layer

The business logic layer implements the functional core of the system. **Servlets** are used to manage client requests and generate dynamic responses, while **Services** handle business operations and transaction management. **Filters** are employed to perform authentication, authorization, and preprocessing of HTTP requests. Additionally, a set of **Utility Classes** provides reusable helper functions for common application tasks.

Data Access Layer

The data access layer abstracts the database operations through **DAOs (Data Access Objects)**. It incorporates **Connection Pooling** using Apache DBCP2 for efficient resource utilization and employs **Model Classes** to represent database entities as Java objects. Database scripts are also included to handle schema creation and initial data population.

2.2.4 Technology Stack Details

Backend Technologies

The backend of the system is built using **Java 11**, providing a robust and modern programming foundation. **Jakarta Servlets 6.1** manage HTTP requests and responses, while **Maven 3** handles build automation and dependency management. The application runs on **Apache Tomcat 9**, which serves as the servlet container and application server.

Database Layer

The database layer uses **MySQL 8.0** as the primary relational database for structured data management. Connection pooling is achieved using **Apache DBCP2**, with **HikariCP** as an alternative high-performance framework ensuring optimized database connectivity.

Security Framework

Security in the system is implemented using several modern frameworks and tools. **JWT (JJWT 0.11.5)** provides secure token-based authentication, while **BCrypt** ensures password hashing for data protection. The integration of **Google Authenticator** enables two-factor authentication (TOTP), and **Apache Commons Codec** supports encoding and decoding operations essential for maintaining data confidentiality.

Frontend Framework

The frontend is developed using **Bootstrap 5.1.3**, which ensures a responsive and visually consistent design. **Font Awesome 6.0** is used for incorporating modern UI icons, and **Vanilla JavaScript (ES6+)** provides the interactivity and client-side scripting necessary for a dynamic and seamless user experience.

Chapter 3: DATABASE SCHEMA:

3.1 Database Design

3.1.1 Entity–Relationship Diagram

The database schema of the **Railway Reservation System** is designed to handle users, trains, bookings, schedules, and seat availability efficiently. Figure 3.1 illustrates the relationships between the major entities in the system.

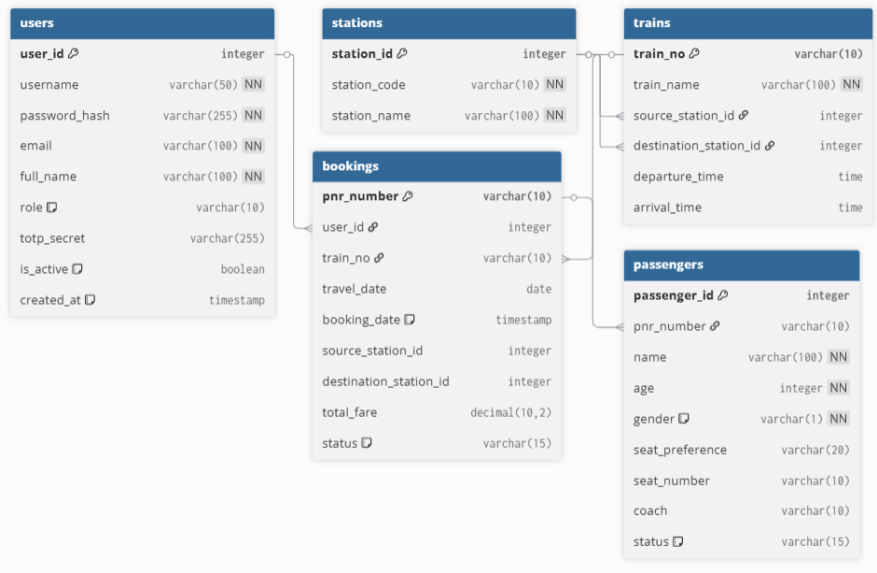


Fig. 3.1 System Architecture Diagram

3.2 Database Design

3.2.1 Overview

The **Railway Reservation System** database is designed to efficiently manage users, trains, bookings, schedules, and seat availability. It ensures data integrity, scalability, and security through normalization and relational constraints.

3.3 Entity Tables

Column Name	Data Type	Description
user_id (PK)	INT	Unique identifier for each user
username	VARCHAR(50)	Login name of the user
password	VARCHAR(255)	Encrypted password (BCrypt hashed)
email	VARCHAR(100)	Registered email ID
full_name	VARCHAR(100)	Full name of the user
role	VARCHAR(20)	Role of the user (Admin/User)
totp_secret	VARCHAR(255)	Secret key for TOTP authentication
is_active	BOOLEAN	Account status flag
created_at	DATETIME	Account creation timestamp

Table 3.1 Users Table Structure

Column Name	Data Type	Description
train_no (PK)	INT	Unique train number
train_name	VARCHAR(100)	Name of the train
source_id (FK)	INT	Source station ID
dest_id (FK)	INT	Destination station ID
dept_time	TIME	Departure time
arriv_time	TIME	Arrival time

Table 3.2 Trains Table Structure

Column Name	Data Type	Description
pnr_number (PK)	VARCHAR(15)	Unique PNR number for each booking
user_id (FK)	INT	References the user making the booking
train_no (FK)	INT	Train associated with the booking
travel_date	DATE	Date of travel
booking_dt	DATETIME	Date and time of booking
source_id (FK)	INT	Source station of travel
dest_id (FK)	INT	Destination station of travel
total_fare	DECIMAL(8,2)	Total fare amount
status	VARCHAR(20)	Booking status (Confirmed, RAC, WL)

Table 3.3 Bookings Table Structure

Column Name	Data Type	Description
passenger_id (PK)	INT	Unique passenger identifier
pnr_number (FK)	VARCHAR(15)	References associated booking
name	VARCHAR(100)	Passenger name
age	INT	Passenger age
gender	CHAR(1)	Passenger gender
seat_number	VARCHAR(10)	Allocated seat number
coach	VARCHAR(5)	Coach name/number
status	VARCHAR(20)	Seat status (Confirmed, RAC, WL)

Table 3.4 Passengers Table Structure

Column Name	Data Type	Description
station_id (PK)	INT	Unique station identifier
station_code	VARCHAR(10)	Official code for the station
station_name	VARCHAR(100)	Full name of the station

Table 3.5 Stations Table Structure

Column Name	Data Type	Description
schedule_id (PK)	INT	Unique schedule identifier
train_no (FK)	INT	References the train
station_id (FK)	INT	References the station
arrival_time	TIME	Arrival time at station
depart_time	TIME	Departure time from station
platform_no	INT	Platform number at station
day_of_week	VARCHAR(10)	Day of operation

Table 3.6 Schedules Table Structure

Column Name	Data Type	Description
availability_id (PK)	INT	Unique identifier for availability record
train_no (FK)	INT	Train associated with the record
travel_date	DATE	Date of journey
class_type	VARCHAR(10)	Type of class (1A, 2A, SL, etc.)
total_seats	INT	Total number of seats
available	INT	Number of available seats
rac_seats	INT	RAC seats available
waitlist	INT	Waitlisted seats count

Table 3.7 SeatAvailability Table Structure

- **Users:** Stores registered user information and authentication details.
- **Trains:** Contains train details including name, source, destination, and timing.
- **Bookings:** Records all booking transactions with corresponding PNR and train numbers.
- **Passengers:** Maintains passenger details linked to each booking.
- **Stations:** Holds station codes, names, and their schedules.
- **SeatAvailability:** Tracks available, RAC, and waitlist seats for each train and date.

Chapter 4: DATABASE FEATURES

4.1 Connection Pooling Configuration

The system uses **Apache DBCP2** for efficient database connection management. Connection pooling minimizes the overhead of creating and closing connections for every user request, enhancing overall performance and scalability.

Listing 4.1 DBCP2 Connection Pool Configuration

```
BasicDataSource dataSource = new BasicDataSource();
dataSource.setUrl("jdbc:mysql://localhost:3306/
´´´´´´´´´´´´´´´´´´´´´´´´´´´railway_reservation_system");
dataSource.setUsername("root");
dataSource.setPassword("");
dataSource.setInitialSize(5);
dataSource.setMaxTotal(20);
dataSource.setMaxIdle(10);
dataSource.setMinIdle(2);
dataSource.setValidationQuery("SELECT´1");
dataSource.setTestOnBorrow(true);
```

4.1.1 Transaction Management

The database operations in the **Railway Reservation System** strictly adhere to **ACID** principles to maintain data consistency and reliability.

- **Atomicity:** Ensures that all steps of a booking transaction are completed successfully or none at all.
- **Consistency:** Maintains valid data state before and after every transaction.
- **Isolation:** Uses optimistic locking to avoid conflicts in concurrent seat booking operations.

✓ **Durability:** Guarantees data persistence even in case of a system crash.

4.2 Core Servlets

4.2.1 LoginServlet.java

Listing 4.2 Login Servlet Implementation

```
@WebServlet(name = "LoginServlet", urlPatterns={"/api/auth/login"})
public class LoginServlet extends HttpServlet {

    private final AuthService authService = new AuthService();

    @Override
    protected void doPost(HttpServletRequest request,
                HttpServletResponse response) throws IOException {


        LoginRequest loginRequest = parseLoginRequest(request);


        AuthService.AuthResult result = authService.authenticate(
            loginRequest.getUsername(),
            loginRequest.getPassword()
        );


        if (result.isSuccess()) {
            String token = JwtUtil.generateToken(result.getUser());
            sendSuccessResponse(response, token, result.getUser());
        } else {
            sendErrorResponse(response, "Invalid credentials");
        }
    }
}
```

4.2.2 Registration Servlets

RegisterServlet.java

Listing 4.3 Register Servlet Implementation

```
@WebServlet(name = "RegisterServlet", urlPatterns ={"/api/auth
    /register"})
public class RegisterServlet extends HttpServlet {
private final UserService userService =new UserService();

    @Override
protected void doPost(HttpServletRequest request,
            HttpServletResponse response) throws IOException {

RegisterRequest registerRequest = parseRegisterRequest(request
    );



if (!validateRegistration(registerRequest)) {
    sendErrorResponse(response, "Invalid input data");
    return;
}

User user = userService.createUser(registerRequest);

    String totpSecret = TotpUtil.generateSecret();
    String qrCodeUri = TotpUtil.generateQRCodeUri(user.
       getUsername(), totpSecret);

    sendRegistrationResponse(response, user, qrCodeUri);
 }
}
```

4.3 Booking Servlets

4.3.1 BookTicketServlet.java

Listing 4.4 Book Ticket Servlet Implementation

```
@WebServlet(name = "BookTicketServlet",
    urlPatterns = {"/api/bookings/book"})
public class BookTicketServlet extends HttpServlet {

 private final BookingService bookingService = new BookingService();

    @Override
    protected void doPost(HttpServletRequest request,
                HttpServletResponse response) throws IOException {
    User user = JwtAuthFilter.getCurrentUser(request);
    if (user == null) {
        response.setStatus(HttpServletResponse.SC_UNAUTHORIZED);
        return;
    }

    BookingRequest bookingRequest = parseBookingRequest(request);
    bookingRequest.setUserId(user.getUserId());
    BookingService.BookingResult result =
    bookingService.bookTicket(bookingRequest);
    if (result.isSuccess()) {
        sendBookingSuccessResponse(response, result.getDetails());
    } else {
        sendBookingErrorResponse(response, result.getMessage());
        }
    }
}
```

Chapter 5: SNAPSHOTS OF PROJECTS:

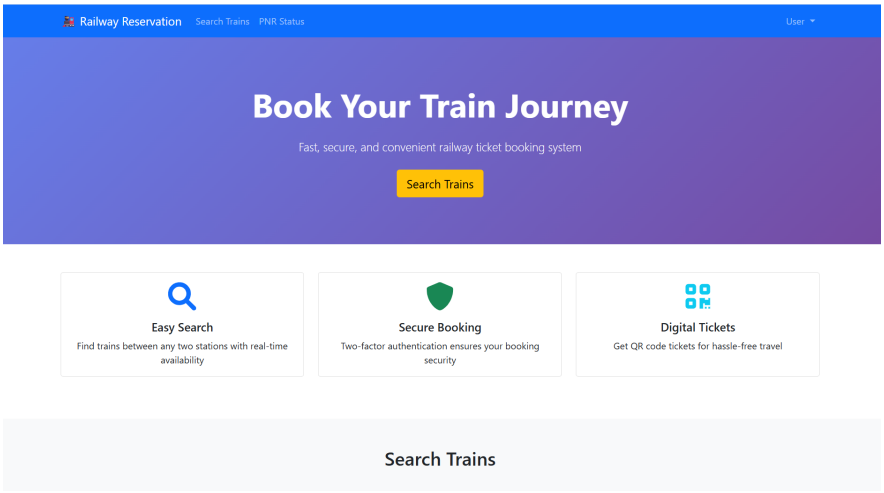


Fig. 5.1 Login page

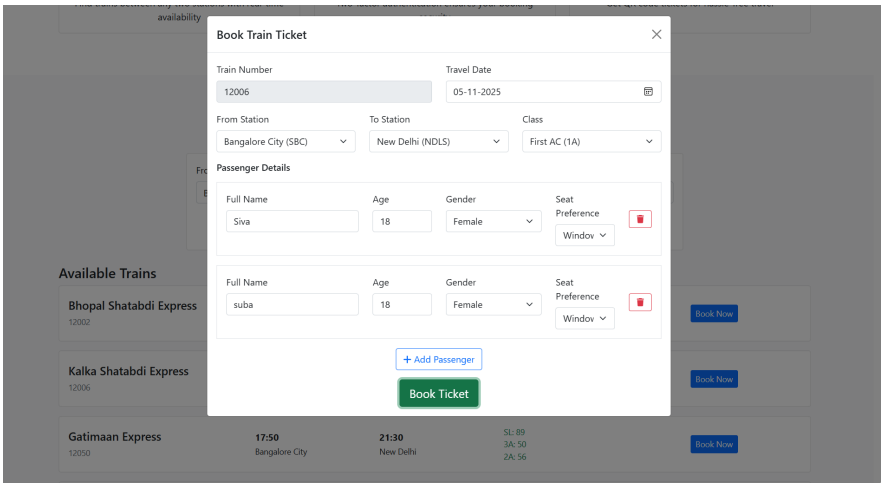


Fig. 5.2 Ticket Booking

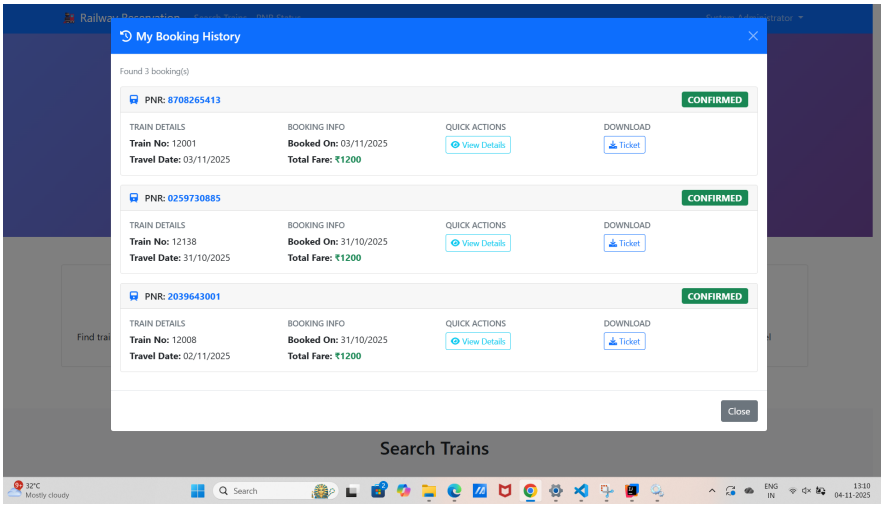


Fig. 5.3 Booking History

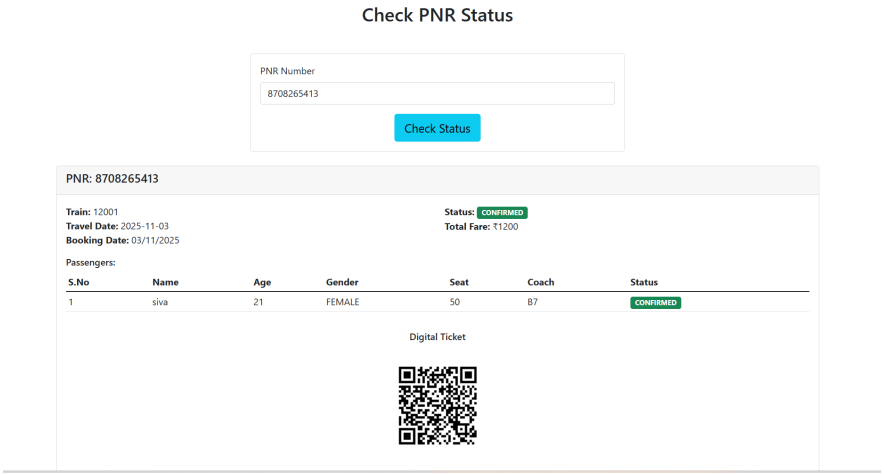


Fig. 5.4 PNR Status

Fig. 5.5 QR Code Verification

Chapter 6: CONCLUSION

The Railway Reservation System represents a comprehensive solution for modern transportation booking needs. Through the implementation of robust security measures, efficient database design, and user-centric features, the system successfully addresses the challenges of traditional booking methods.

6.1 Future Roadmap

The Railway Reservation System provides a solid foundation for future enhancements.

6.1.1 Immediate Opportunities

- ✓ Mobile application development for enhanced user accessibility.
- ✓ Payment gateway integration for complete booking workflow.
- ✓ Advanced search and filtering capabilities.
- ✓ Multi-language support for diverse user base.

6.1.2 Strategic Development

- ✓ Artificial Intelligence integration for price prediction and recommendations.
- ✓ Microservices architecture for improved scalability.
- ✓ Cloud deployment for global accessibility.
- ✓ IoT integration for smart ticketing solutions.

6.2 Final Remarks

This Railway Reservation System demonstrates the successful application of modern software engineering principles to solve real-world problems. The combination of robust backend architecture, secure authentication mechanisms, and intuitive user interface creates a comprehensive solution that meets both current requirements and future scalability needs.

The project serves as a testament to the power of well-designed software systems in transforming traditional business processes. Through careful planning, implementation of best practices, and attention to user needs, the system delivers significant value to all stakeholders while establishing a foundation for continued innovation and improvement.

The success of this project reinforces the importance of:

- Comprehensive system design and architecture planning.
- Implementation of robust security measures from project inception.
- User-centric design and development approaches.
- Continuous testing and quality assurance processes.
- Documentation and knowledge transfer for long-term maintenance.

As transportation systems continue to evolve and digitize, this Railway Reservation System provides a robust platform that can adapt and grow with changing requirements, ensuring long-term value and utility for users, administrators, and the broader transportation ecosystem.

REFERENCES AND RESOURCES

1. Eclipse Foundation, *Jakarta Servlet Specification 6.0 - Official Documentation*, (2023). Available: https://jakarta.ee/specifications/servlet/6.0/

2. Oracle Corporation, *MySQL 8.0 Reference Manual*, 2024. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/

3. Bootstrap Team, *Bootstrap 5.0 Documentation*, 2021. [Online]. Available: https://getbootstrap.com/docs/5.0

4. M. Jones, J. Bradley, N. Sakimura, *JSON Web Tokens (JWT) - RFC 7519 Specification*, IETF (2015). DOI: 10.17487/RFC7519

5. D. M'Raihi, S. Machani, M. Pei, J. Rydell, *TOTP: Time-Based One-Time Password Algorithm - RFC 6238*, IETF (2011). DOI: 10.17487/RFC6238

6. OWASP Foundation, *OWASP Web Application Security Testing Guide*, (2024). Available: https://owasp.org/www-project-web-security-testing-guide/

7. National Institute of Standards and Technology (NIST), *Cybersecurity Framework*, (2023). Available: https://www.nist.gov/cyberframework

8. PCI Security Standards Council, *PCI DSS Requirements for Payment Processing*, (2022). Available: https://www.pcisecuritystandards.org/

9. International Organization for Standardization, *ISO/IEC 27001 Information Security Management*, (2022). Available: https://www.iso.org/isoiec-27001-information-security.html

10. Apache Software Foundation, *Apache Maven Documentation - Build Automation*, (2023). Available: https://maven.apache.org/guides/

11. Apache Software Foundation, *Apache Tomcat Configuration Guide*, (2024). Available: https://tomcat.apache.org/tomcat-9.0-doc/

12. QOS.ch, *SLF4J Manual*, 2023. [Online]. Available: http://www.slf4j.org/manual.html

13. BCrypt Project, *BCrypt Documentation - Password Hashing*, (2023). Available: https://en.wikipedia.org/wiki/Bcrypt

14. R. C. Martin, *Clean Code: A Handbook of Agile Software Craftsmanship*, Prentice Hall, (2008).

15. E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, (1994).

16. S. Newman, *Building Microservices*, O'Reilly Media, (2015).

17. B. Sullivan, V. Liu, *Web Application Security: A Beginner's Guide*, McGraw-Hill Education, (2011).