"ONLINE TOLL NOTIFICATION SYSTEM"

A

Project Report

submitted

in partial fulfillment

for the award of the Degree of

Bachelor of Technology

in Department of Information Technology

Project Mentor:
Dr. S.R. Dogiwal
Associate Professor

Submitted By :
Harshit Kumar Jain, 21ESKIT053
Himansh Singh, 21ESKIT055
Chirag, 21ESKIT036

Department of Information Technology
Swami Keshvanand Institute of Technology, M & G, Jaipur
Rajasthan Technical University, Kota
Session 2024-2025

Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur

Department of Information Technology

CERTIFICATE

This is to certify that Mr. Harshit Kumar Jain, a student of B.Tech(Information Technology) VIII semester has submitted his Project Report entitled ”Online Toll Notification System” under my guidance.

Mentor

Dr. S.R. Dogiwal

Associate Professor

Signature: _____

Coordinator

Dr. Priyanka Yadav

Assistant Professor

Signature: _____

Swami Keshvanand Institute of Technology,

Management & Gramothan, Jaipur

Department of Information Technology

CERTIFICATE

This is to certify that Mr. Himansh Singh, a student of B.Tech(Information Technology) VIII semester has submitted his Project Report entitled ”Online Toll Notification System” under my guidance.

Mentor

Dr. S.R. Dogiwal

Associate Professor

Signature: _____

Coordinator

Dr. Priyanka Yadav

Assistant Professor

Signature: _____

Swami Keshvanand Institute of Technology,

Management & Gramothan, Jaipur

Department of Information Technology

CERTIFICATE

This is to certify that Mr. Chirag, a student of B.Tech(Information Technology) VIII semester has submitted his Project Report entitled ”Online Toll Notification System” under my guidance.

Mentor

Dr. S.R. Dogiwal

Associate Professor

Signature: _____

Coordinator

Dr. Priyanka Yadav

Assistant Professor

Signature: _____

DECLARATION

We hereby declare that the report of the project entitled **Online Toll Notification System** is a record of an original work done by us at Swami Keshvanand Institute of Technology, Management and Gramothan, Jaipur under the mentorship of **Dr. S.R. Dogiwal** (Associate Professor, Department of Information Technology). This project report has been submitted as the part of original work for the partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology (B.Tech)** in the Department of Information Technology. It has not been submitted anywhere else, under any other program to the best of our knowledge and belief.

Team Members **Signature**

Harshit Kumar Jain, 21ESKIT053

Himansh Singh, 21ESKIT055

Chirag, 21ESKIT036

Acknowledgement

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in presentation of the project and preparation of this report. It would not have been possible to present the project and report in this form without their valuable help, cooperation and guidance.

I am extremely grateful to **Dr. (Prof.) Anil Chaudhary**, HOD, Department of Information Technology, as well as our class coordinator for the guidance and encouragement and for providing me with best facilities and atmosphere for the creative work. I would like to thank my project guide, **Dr. S.R. Dogiwal**, Associate Professor, for the valuable guidance. care and timely support throughout the project work. He has always a constant source of encouragement. I would like to thank my family and friends for their encouragement, which helped me to keep my spirit alive and to complete this work successfully. I would also like to express my gratitude to our lab coordinator, **Dr. Priyanka Yadav**, Assistant Professor, for their support and assistance during this project.

At last I must express my sincere heartfelt gratitude to all the staff members of Department of Information Technology who helped me directly or indirectly during this course of study.

Team Members:

Harshit Kumar Jain, 21ESKIT053

Himansh Singh, 21ESKIT055

Chirag, 21ESKIT036

Table of Content

List of Figures

Chapter 1

INTRODUCTION

1.1 Purpose

The purpose of the **Online Toll Notification System** is to create a smart, efficient, and user-friendly platform that allows users to receive real-time notifications of toll charges along their travel routes. The system aims to simplify toll management by providing route-based toll alerts, enhancing user awareness, and helping commuters make informed decisions before and during their journeys. This project offers a seamless interface for both end-users and administrators. Users can view toll plazas along their route using location names, while administrators can manage toll records, user data, and monitor system activity through a dedicated backend dashboard.

1.2 Project Objectives

The primary objectives of the **Online Toll Notification System** are:

- To enable users to search for tolls between two locations using route mapping.
- To notify users in real-time about upcoming toll plazas and associated charges using geolocation and WebSocket communication.
- To support OTP-based authentication for user registration, login, and password recovery to ensure secure access.

- To provide an intuitive dashboard for administrators to manage users, toll records, and system activity.
- To build a scalable, modular platform using modern web technologies.

1.3 Project Tools and Technologies Used

The development of the **Online Toll Notification System** involved the use of the following tools and technologies:

- **Frontend:** React.js with Tailwind CSS for building a responsive and visually consistent user interface.
- **Backend:** Node.js with Express.js for API development and server-side logic.
- **Database:** MongoDB for storing user data, toll records, and session information.
- **Routing and Mapping:** Mapbox API for route visualization and toll plaza mapping.
- **Authentication:** Twilio API for OTP-based phone number verification.
- **Real-time Features:** WebSocket (Socket.IO) for sending toll notifications.
- **Hosting/Deployment:** Render.

1.4 Significance

The **Online Toll Notification System** plays a crucial role in enhancing the toll experience for road users by:

- Allowing travelers to pre-plan their journeys with clear insights into toll points and costs.

- Reducing confusion and delays by providing timely toll alerts along the selected route.
- Improving operational efficiency by enabling centralized toll management for administrators.
- Contributing to digital transformation efforts in transportation and infrastructure by leveraging modern location-based services and real-time data handling.

Chapter 2

SYSTEM REQUIREMENTS SPECIFICATION (SRS)

2.1 Functional Requirements

The functional requirements define the core features and capabilities that the Online Toll Notification System must provide for its users and administrators. These functionalities ensure the system fulfills its primary objective of managing toll notifications and user interactions effectively.

- **User Registration and Login:** Users must be able to securely register, log in, and manage their accounts using OTP-based verification.
- **OTP Verification:** The system should support OTP verification for new registrations, logins, and password resets to ensure secure access.
- **Toll Route Detection:** The system should identify and display toll plazas along a user's route using Mapbox routing APIs.
- **Notification Management:** Real-time notifications about toll booths encountered, charges, or upcoming tolls should be delivered via WebSockets.
- **Admin Management Panel:** Admins should be able to view user details, manage toll plaza data, and monitor route activity.
- **Password Reset:** Users must be able to initiate a "Forgot Password" flow using OTP authentication and update their password securely.

- **User Profile Management:** Users should be able to update profile information such as name and phone number.
- **Toll History Tracking:** Users can view their previous routes and toll history.
- **Resend OTP Feature:** Users should be able to request a new OTP if the previous one expires or is not received.
- **Logout:** Secure session handling should allow users to log out cleanly.

2.2 Non-Functional Requirements

These requirements describe the operational aspects of the system, ensuring it runs efficiently and meets user expectations in terms of security, performance, and usability.

- **Performance:** The system should handle real-time data updates and multiple concurrent users without noticeable delay.
- **Scalability:** The system must be scalable to support a growing user base and expansion of toll data.
- **Security:** Sensitive user data and route information must be protected using secure communication (e.g., HTTPS, encrypted OTPs).
- **Availability:** The system should maintain high uptime with robust monitoring and minimal downtime for maintenance.
- **Usability:** The UI should be responsive and intuitive across different screen sizes and devices.
- **Reliability:** The system should perform reliably under normal and peak usage conditions with minimal crashes.

- **Backup and Recovery:** Toll and user data should be regularly backed up, with mechanisms in place for disaster recovery.

2.3 Software and Hardware Requirements

2.3.1 Software Requirements

- **Operating System:** Windows, macOS, or Linux for both development and production.
- **Web Server:** Node.js server using Express.js framework.
- **Database:** MongoDB for storing user data, toll information, and session records.
- **Development Stack:**
  - Frontend: React.js, Tailwind CSS, JavaScript
  - Backend: Node.js, Express.js
- **APIs and Services:**
  - Mapbox API: For routing and toll plaza detection along the user's journey.
  - Twilio API: For OTP verification via SMS.
  - WebSocket (Socket.IO): For real-time user notifications.
- **Browsers Supported:** Chrome, Firefox, Edge, Safari

2.3.2 Hardware Requirements

- **Server:** Cloud-based deployment with at least 4 vCPUs, 8GB RAM, and 100GB storage.

- **Client Devices:** Desktop, laptop, or mobile devices capable of running a modern browser.
- **Network:** Stable internet connection with sufficient bandwidth to support real-time data exchange for multiple users.

Chapter 3

SYSTEM ANALYSIS AND DESIGN

Introduction

This chapter presents the system-level design of the Toll Notification System, focusing on real-time toll alerts, user management, and admin controls. It includes:

Use Case Diagram

The use case diagram provides a high-level overview of the system functionality from the user's perspective. It identifies the system's primary actors and their interactions with various modules.

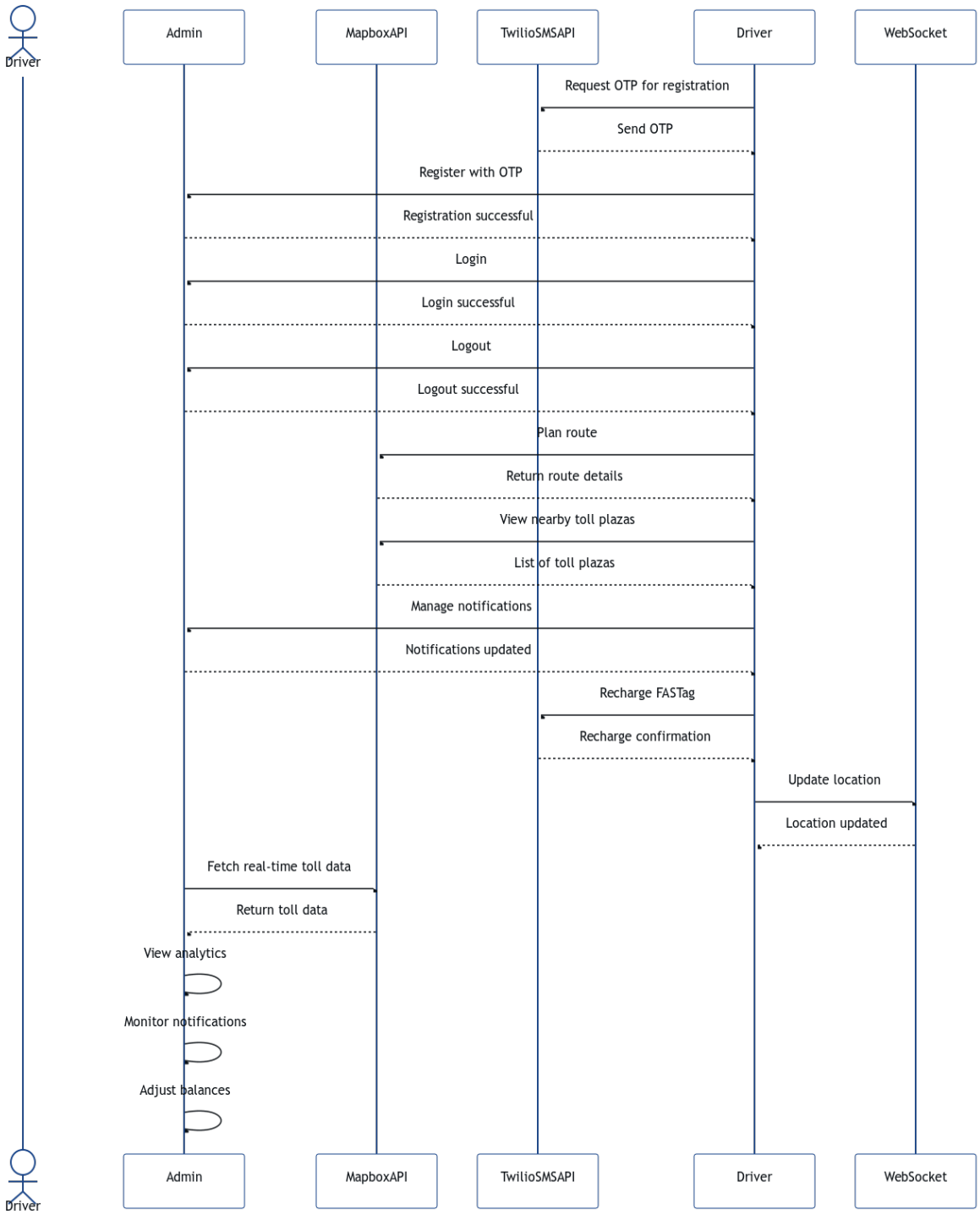

Figure 3.1: Use case diagram for Online Toll Notification System

Actors:

- **User (Driver):** Plans routes, receives toll alerts, manages profile.
- **Admin:** Manages toll plaza data, monitors users, views analytics.

- **External Systems:**
  - **Mapbox API** (Route calculation).
  - **Twilio API** (OTP SMS).
  - **Overpass API** (Real-time toll data).

Data Flow Diagram (DFD)

The DFD shows how data moves within the system. It explains the flow of inputs and outputs between processes and data stores.

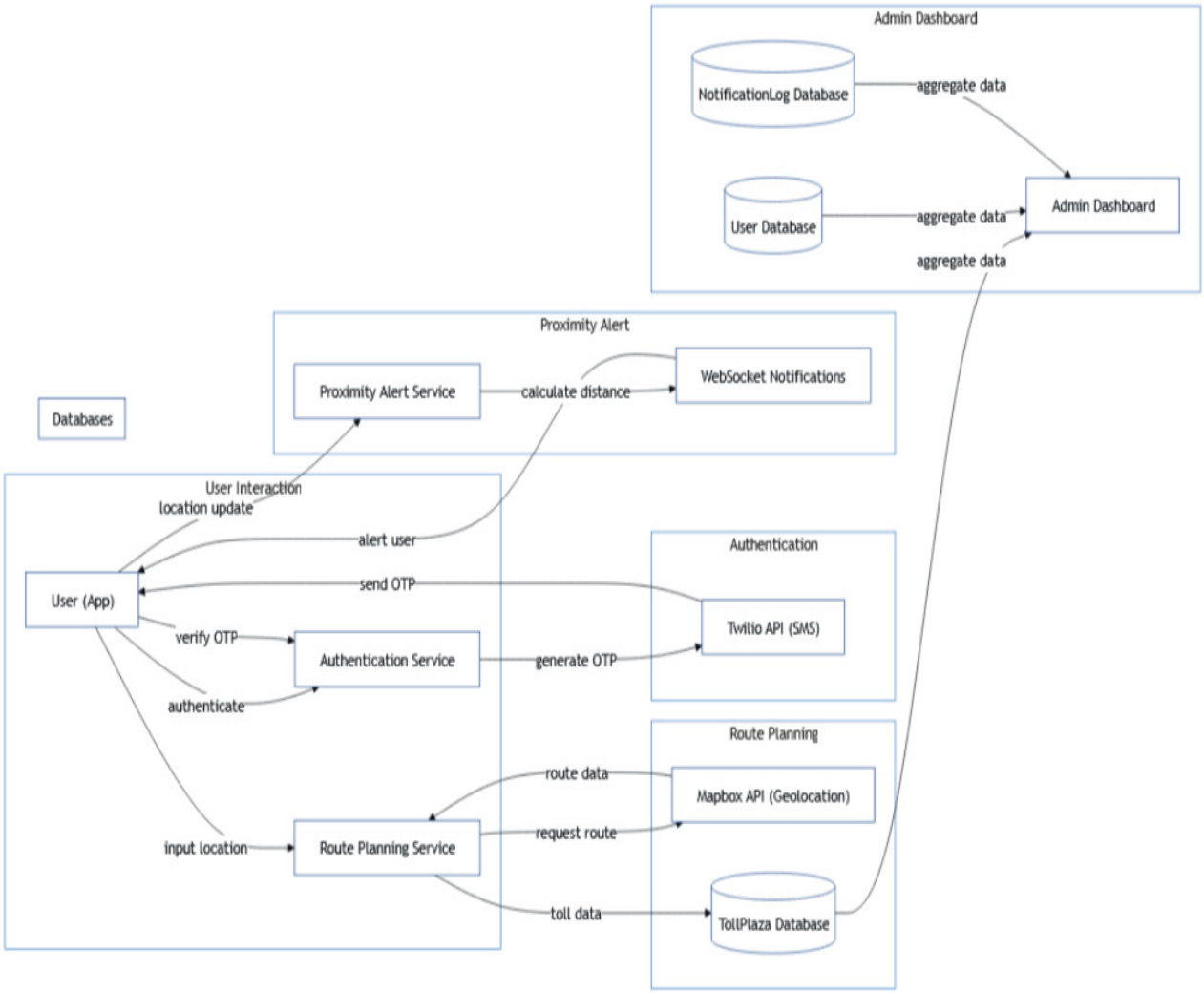

Figure 3.2: Data Flow diagram for Online Toll Notification System

Processes:

- **User Authentication:** OTP generation → Twilio SMS → Verification.
- **Route Planning:** User inputs → Mapbox API → Route + Toll Data.
- **Proximity Alert:** User location → Distance calculation → WebSocket notification.
- **Admin Dashboard:** Toll/User data → Aggregation → Analytics.

Data Stores:

- **User Database** (MongoDB).
- **TollPlaza Database** (Synced with Overpass API).
- **NotificationLog** (Alerts history).

External Entities:

- **User** (Mobile/Web App).
- **Twilio API** (SMS).
- **Mapbox/Overpass API** (Geolocation).

Entity-Relationship Diagram (ERD)

The ER diagram displays the logical structure of the database. It outlines the relationships between entities involved in the system.

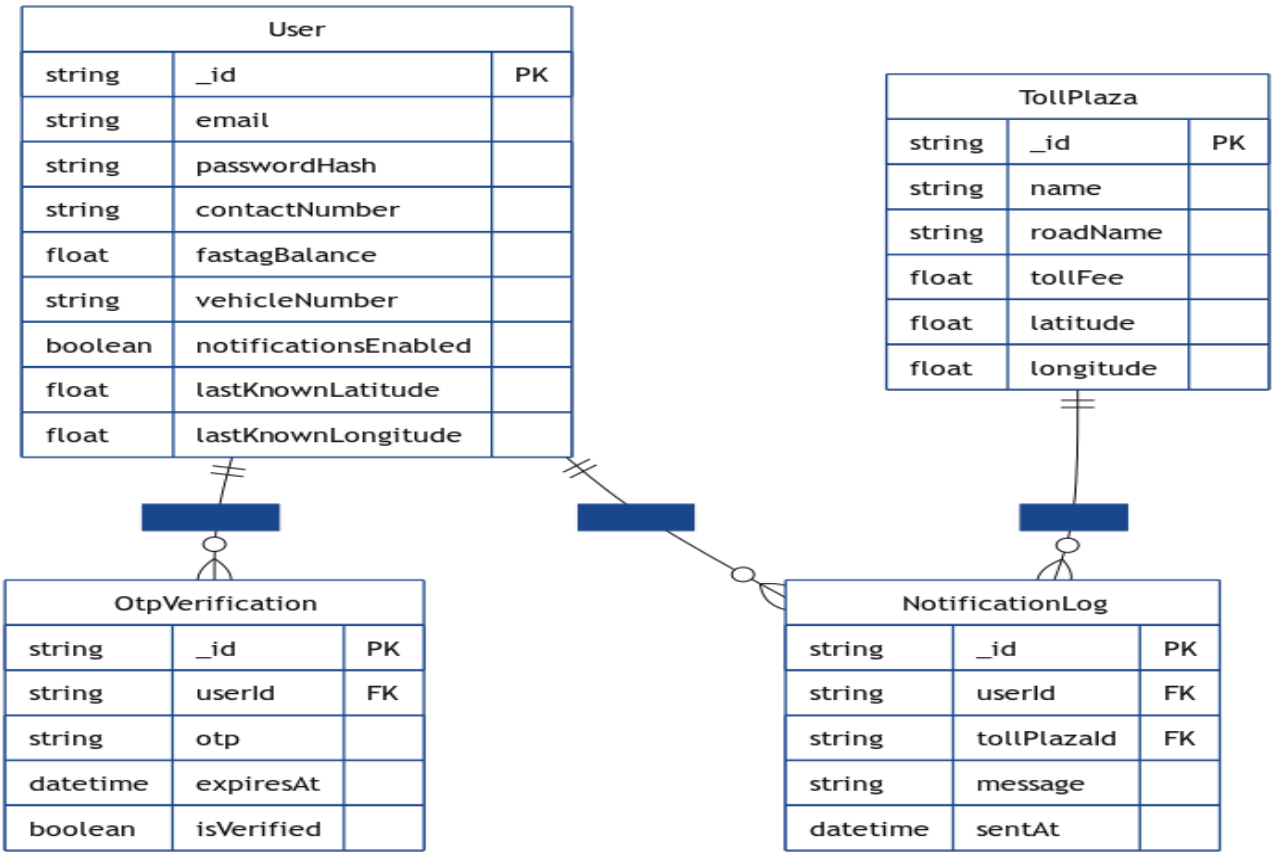

Figure 3.3: ER diagram for Online Toll Notification System

Key Entities:

- **User:** Contains personal details (email, contact number) and authentication data (password hash), with FASTag balance and notification preferences.
- **TollPlaza:** Stores toll plaza information (name, road, fee) and geographic coordinates (latitude/longitude).
- **OtpVerification:** Manages OTP codes for user verification, linked to user accounts with expiration timestamps.
- **NotificationLog:** Tracks all sent alerts, connecting users to toll plazas with timestamps and message content.

Chapter 4

MODULES AND FUNCTIONALITIES

4.1 Description of Proposed Modules

The Online Toll Notification System is structured around multiple integrated modules, each responsible for a distinct part of the system's functionality. These modules collaborate to deliver a real-time, responsive toll alert system for users and an efficient management interface for administrators. The key modules in the system include:

- **User Module:** Enables user registration, login, OTP verification, and toll viewing through a route-based interface.
- **Admin Module:** Allows administrators to manage users, toll data, and system analytics.
- **Notification Module:** Provides real-time toll alerts to users based on geolocation and routing.
- **Route Planning Module:** Facilitates route input using location names and displays toll plazas along the selected route using Mapbox.
- **Authentication Module:** Handles secure login and password recovery using OTP-based verification via Twilio.

4.2 Detailed Functionality

Each module in the system plays a critical role in ensuring a seamless and effective toll notification experience. The functionalities of each module are described below:

- **User Module:**
  - **Registration and Login:** Users can register with their phone numbers and verify using OTP. They can log in to access personalized features.
  - **Forgot Password:** Users can initiate a password reset using OTP verification.
  - **Toll Browsing:** Users can input their source and destination locations to view the tolls along the route
  - **Real-Time Toll Alerts:** Users receive notifications about upcoming toll plazas as they travel, based on route data and socket communication.
- **Admin Module:**
  - **User Management:** Admins can view all registered users, approve, deactivate, or manage user details.
  - **Toll Management:** Admins can add, edit, or delete toll plaza records including name, location, and charge.
  - **Activity Monitoring:** Admins have access to a dashboard displaying system usage metrics and recent activity logs.
  - **Notification Control:** Admins can broadcast system-wide alerts or updates to all users.
- **Notification Module:**

  - **WebSocket Integration:** Implements real-time communication to send toll alerts to users as they approach a toll plaza.
  - **Geolocation Triggering:** Based on user location and route tracking, toll alerts are pushed automatically.
  - **Notification History:** Maintains a history of sent notifications for admin tracking and analysis.

- **Route Planning Module:**
  - **Route Input:** Users can search for routes using place names instead of coordinates.
  - **Mapbox Integration:** Displays the complete route with marked toll plazas using the Mapbox API.
  - **Toll Filtering:** Dynamically filters and displays tolls that lie along the computed route.

- **Authentication Module:**
  - **OTP Verification:** Utilizes Twilio to send one-time passwords for secure user verification during registration and password recovery.
  - **Session Handling:** Maintains secure user sessions and prevents unauthorized access.
  - **Auto-Resend & Timer:** Allows users to request a new OTP after a timer expires, ensuring convenience and security.

Chapter 5

IMPLEMENTATION

5.1 Tools and Technologies Used

The Online Toll Notification System is developed using modern web technologies to deliver a responsive, real-time toll notification platform with a strong focus on usability, scalability, and maintainability. Below are the primary tools and technologies used in this project:

- **Frontend:**
  - **React.js:** A powerful JavaScript library used to build a dynamic and component-based user interface for the system.
  - **Tailwind CSS:** A utility-first CSS framework that enables rapid UI development with a consistent design system.
  - **JavaScript (ES6+):** Used to enhance interactivity and handle client-side logic.
  - **Mapbox GL JS:** Integrated for map visualization and route plotting using location-based services..
- **Backend:**
  - **Node.js:** A server-side JavaScript runtime used to handle backend processing and API endpoints.

  - **Express.js:** A lightweight web framework used to define RESTful routes, handle middleware, and manage backend logic.

- **Database:**
  - **MongoDB:** A NoSQL database used to store user details, toll plaza information, OTP sessions, and system notifications. Its document-oriented model supports scalable data handling.

- **Authentication**
  - **Twilio API:** Integrated for sending OTPs (One-Time Passwords) during user registration and password reset, ensuring secure authentication flows.

- **Real-Time Communication:**
  - **Socket.IO (WebSocket):** Enables real-time toll notifications by pushing alerts to users as they travel through mapped routes.

- **Hosting and Deployment:**
  - **Render:** Used to host the React frontend with CI/CD for smooth deployment. Also, used for hosting the Express backend and MongoDB database in a managed cloud environment.

- **Version Control:**
  - **Git and GitHub:** Used for version control and collaboration between team members during development.

5.2 Screenshots of the Application

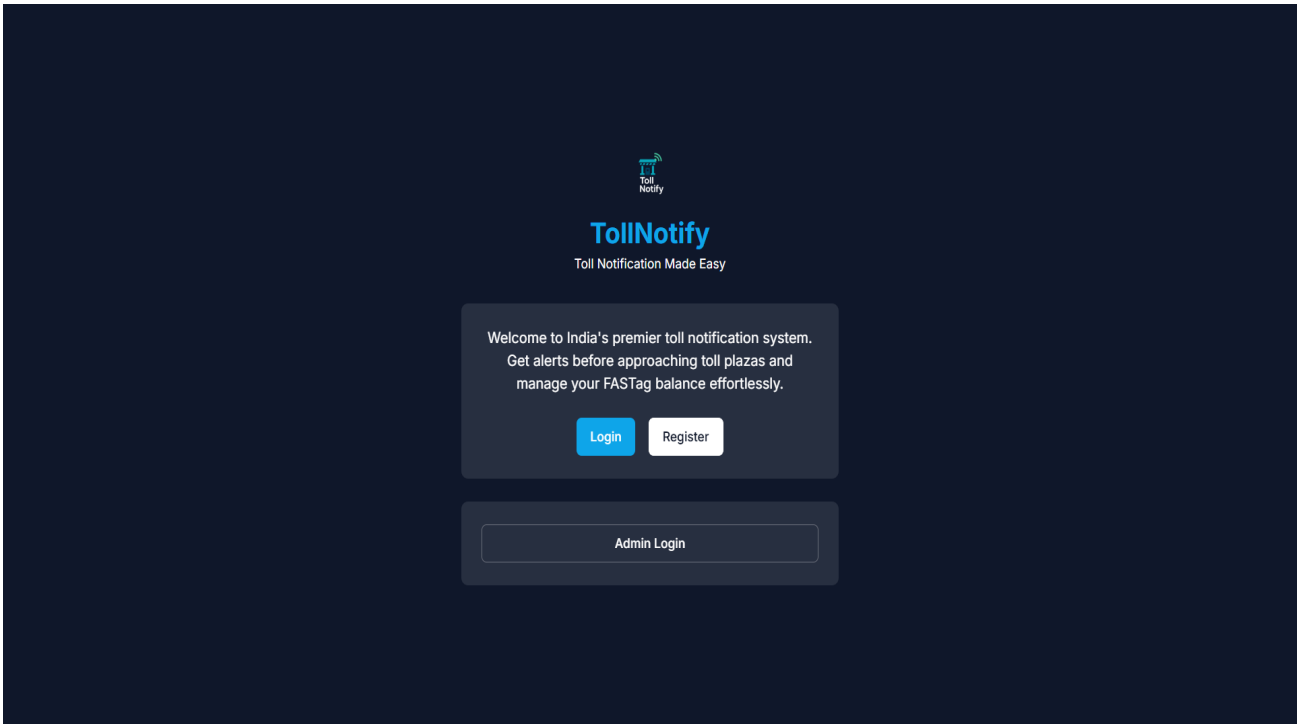

Figure 5.1: Welcome Page

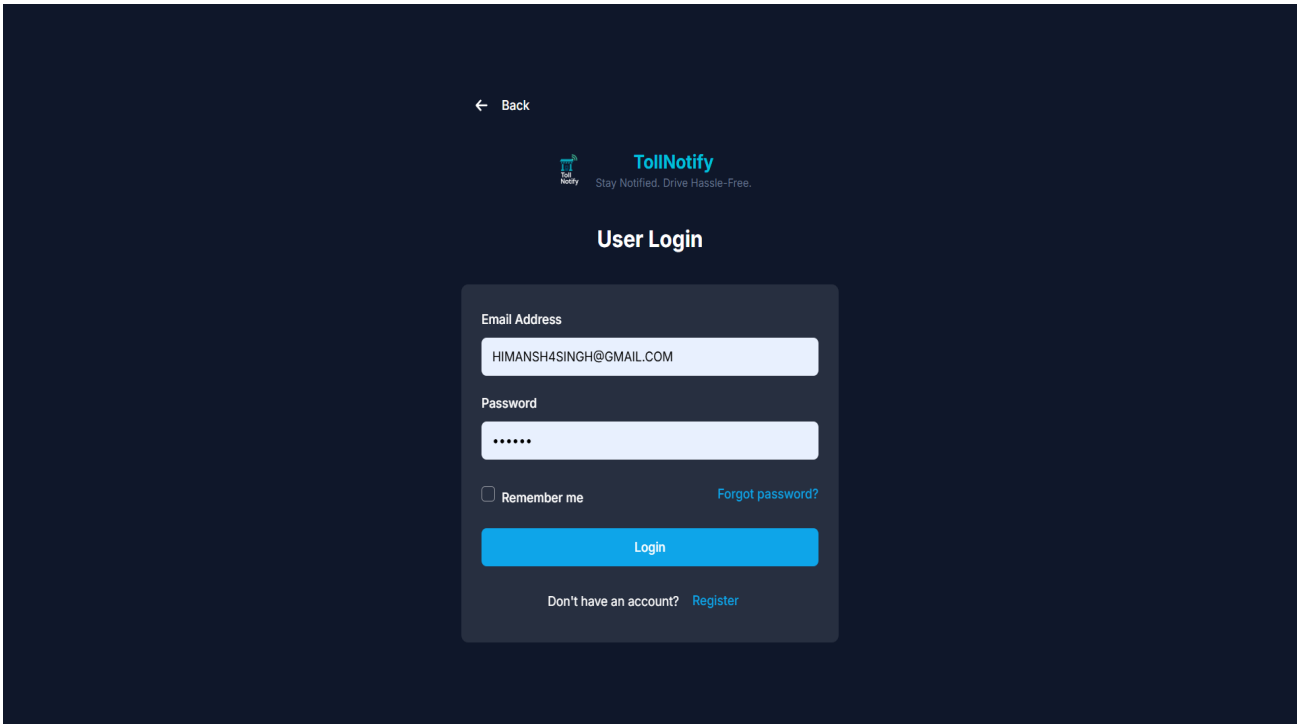

Figure 5.2: Login Page

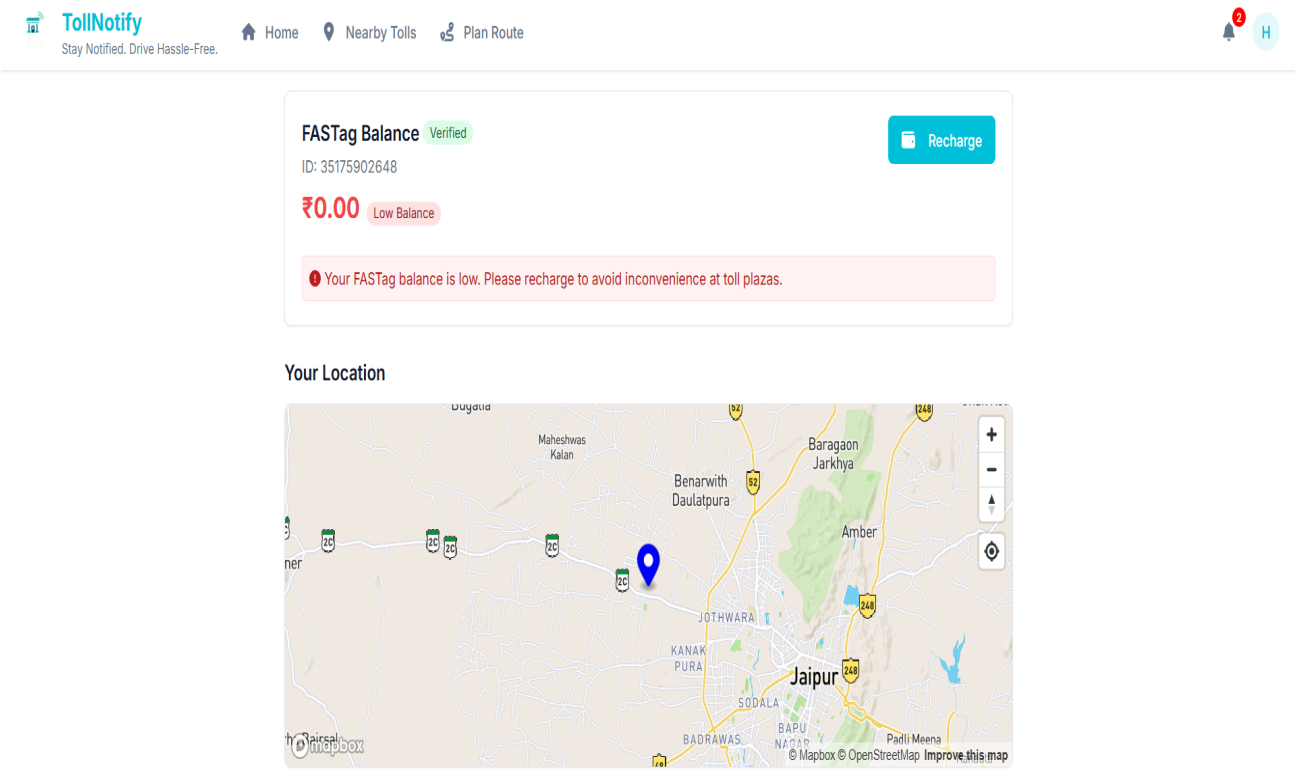

Figure 5.3: Home Page

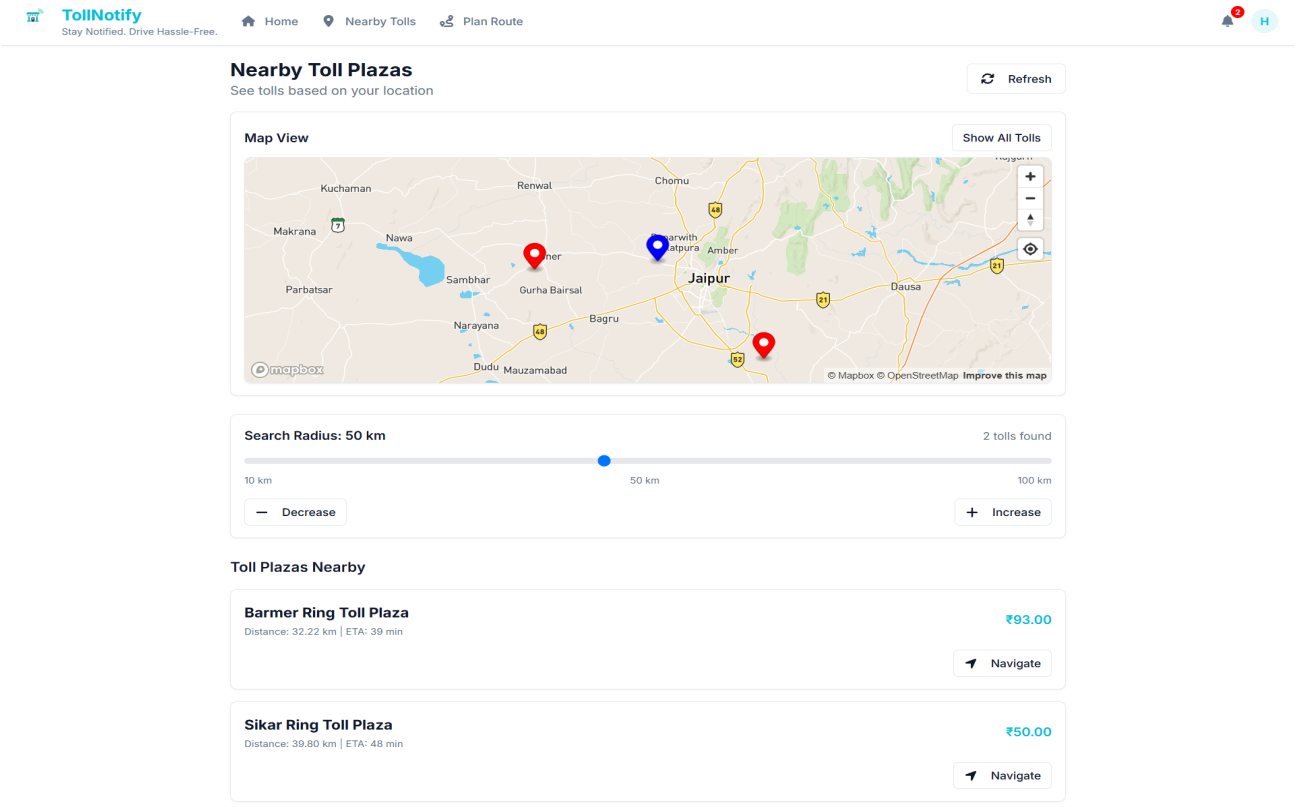

Figure 5.4: Nearby Tolls Page

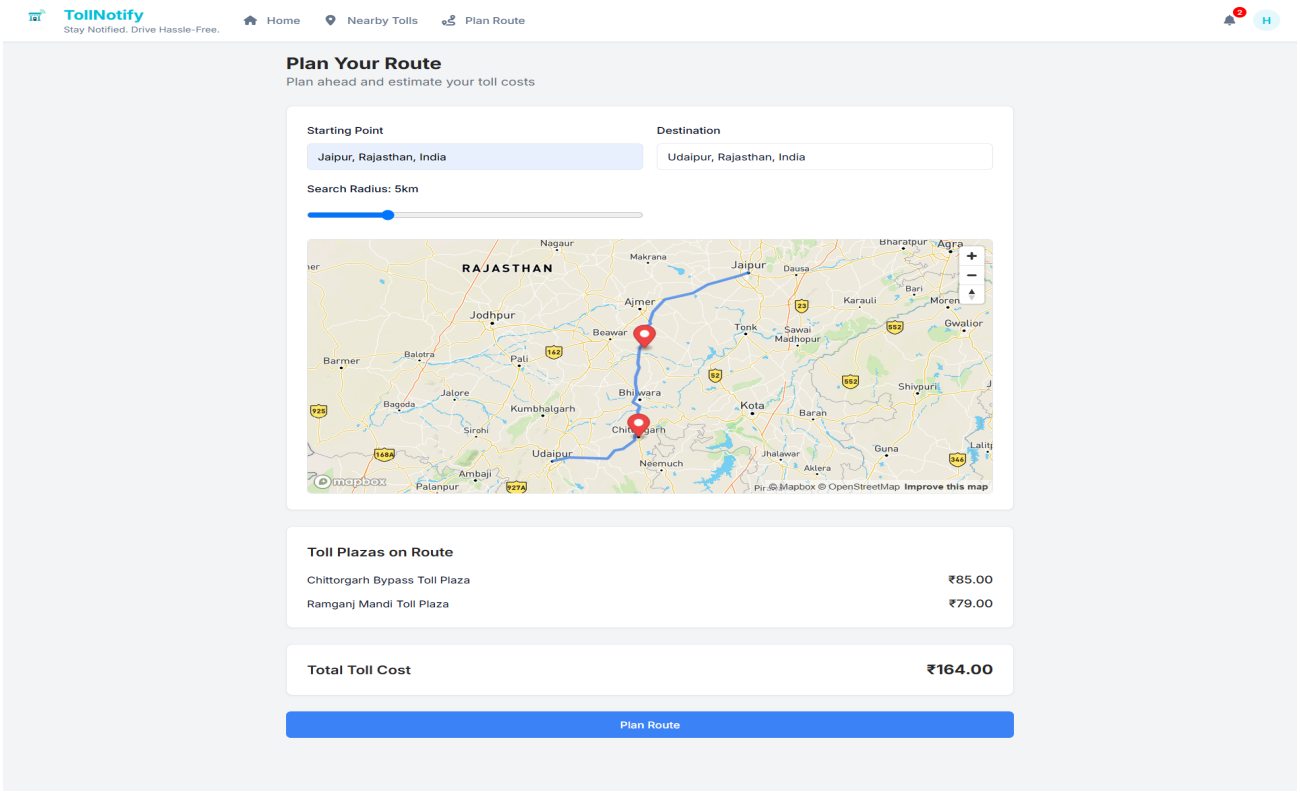

Figure 5.5: Plan Route Page

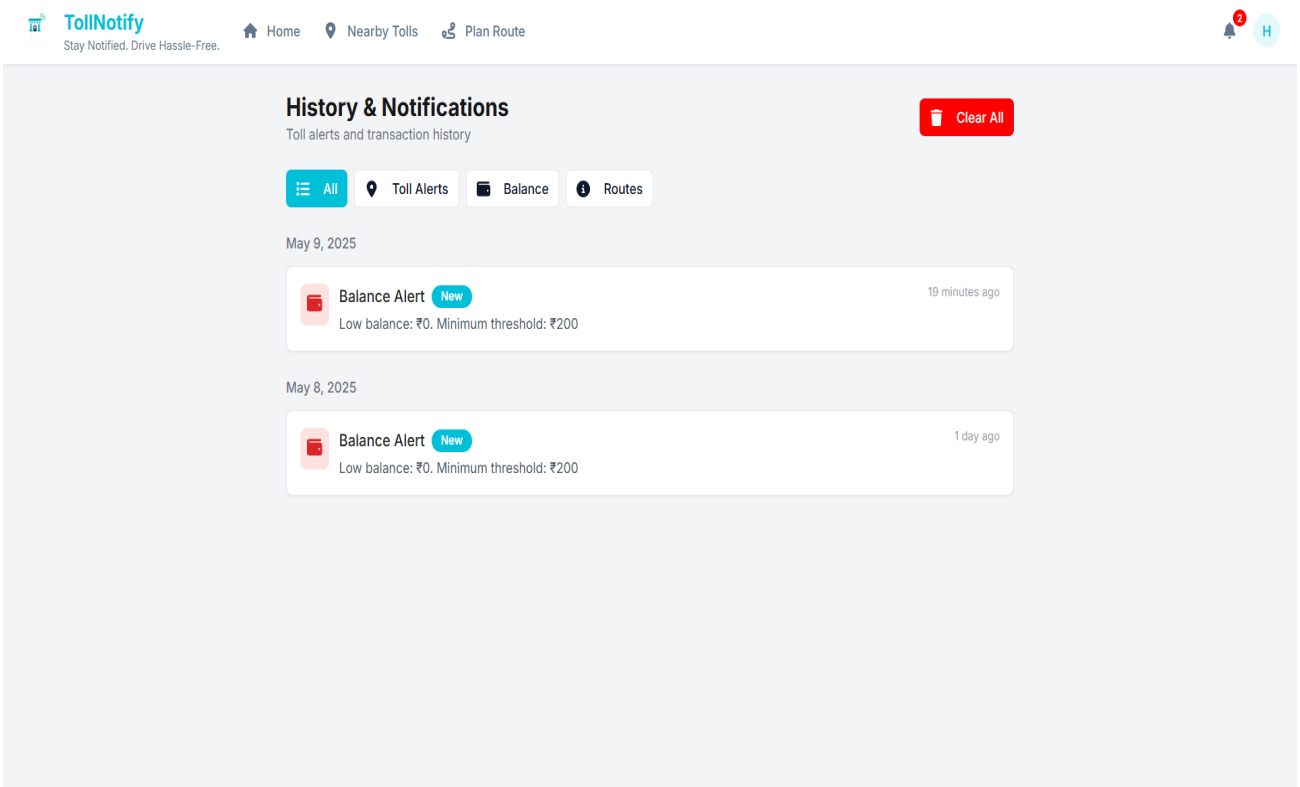

Figure 5.6: History and Notifications Page

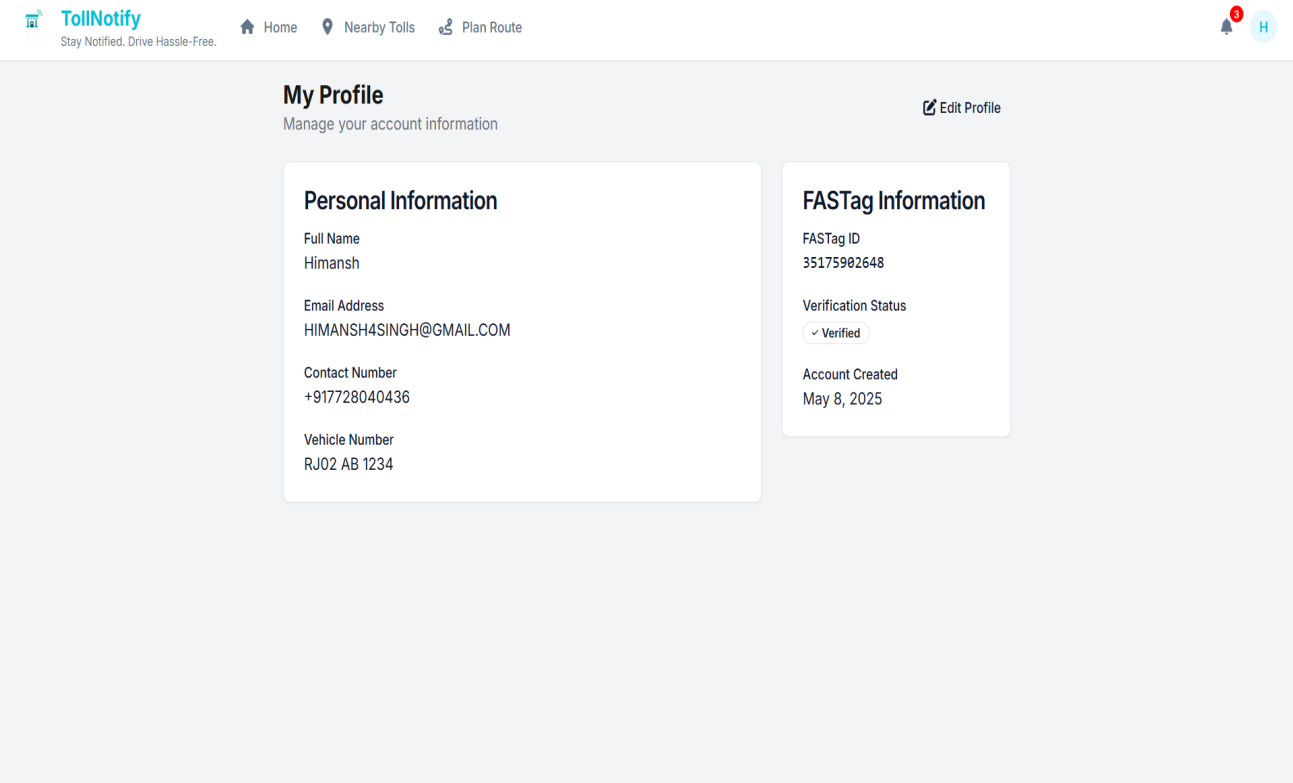

Figure 5.7: Profile Page

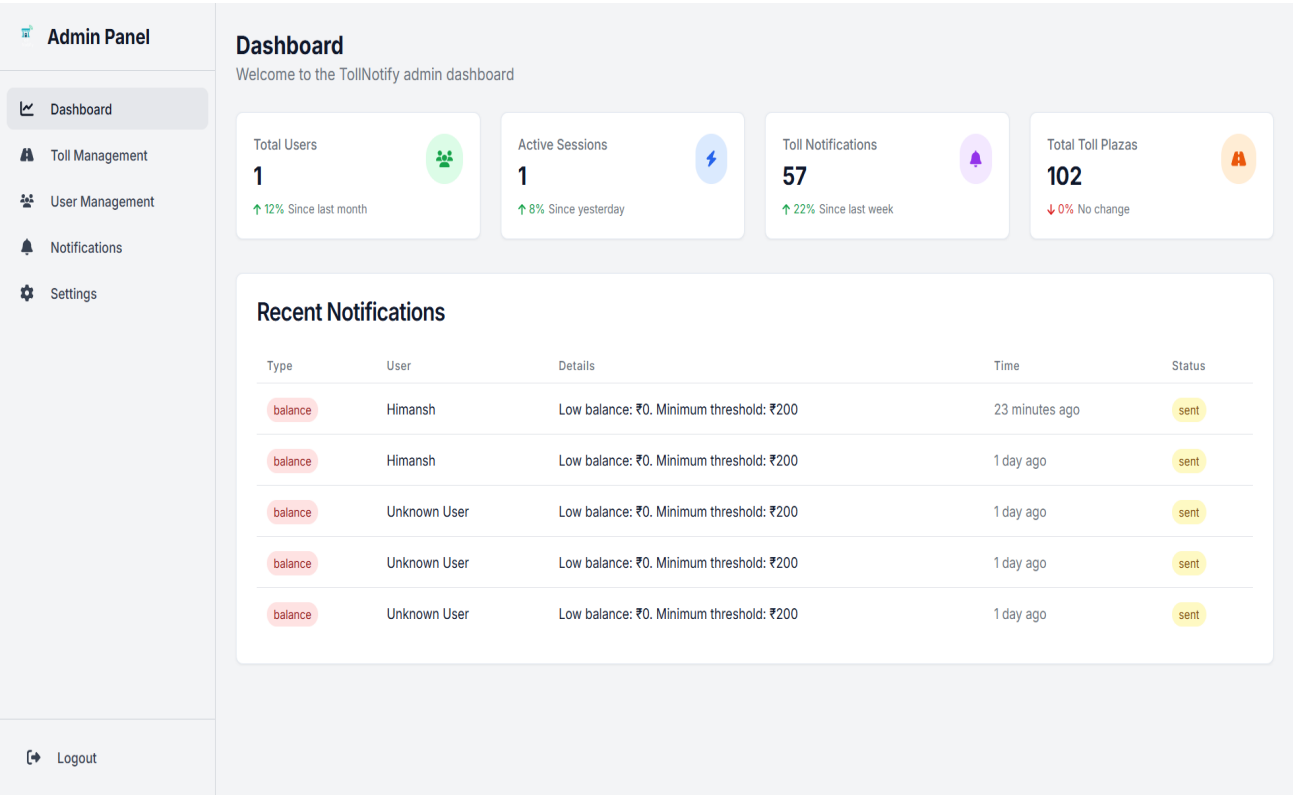

Figure 5.8: Admin Dashboard Page

Chapter 6

TESTING

6.1 Testing Methodologies

The testing phase of the Online Toll Notification System was conducted using a combination of manual and automated methods to ensure that the application functions correctly, reliably, and securely across all modules. The following testing methodologies were applied:

- **Unit Testing:** Each independent component, such as OTP authentication, toll fetching, and route mapping, was tested in isolation to ensure correctness and expected outputs.
- **Integration Testing:** Focused on verifying that all integrated parts of the system such as frontend interactions with backend APIs and WebSocket connections worked together seamlessly.
- **Functional Testing:** Ensured that all major features such as user registration, route-based toll detection, real-time notifications, and admin operations were working as per requirements.
- **Usability Testing:** Conducted to verify that the system interface was intuitive, responsive, and accessible for both users and administrators.
- **Security Testing:** Validated that sensitive operations such as OTP verification and password resets were secure against unauthorized access or tampering.

- **Performance Testing:** Stress tests were performed to check system behavior under simulated traffic conditions, especially focusing on real-time notifications and API response times.

6.2 Test Cases and Results

The following are some of the key test cases executed during the testing process:

6.2.1 Test Case 1: User Registration and OTP Verification

- **Description:** Verify that a user can register and verify their phone number using OTP.
- **Input:** Valid phone number, correct OTP.
- **Expected Result:** User account created and redirected to login page.
- **Actual Result:** Passed, user successfully registered and verified.
- **Status: Passed**

6.2.2 Test Case 2: Route Toll Detection

- **Description:** Verify that tolls along a route are correctly displayed when user enters source and destination.
- **Input:** ”Jaipur” as source and ”Delhi” as destination.
- **Expected Result:** Toll plazas along the selected route are displayed on the map.
- **Actual Result:** Passed, correct tolls were identified and shown.
- **Status: Passed**

6.2.3 Test Case 3: Real-Time Toll Notification

- **Description:** Verify that users receive real-time toll alerts while traveling.
- **Input:** Simulated movement along a route with tolls.
- **Expected Result:** Notifications appear as user reaches toll locations.
- **Actual Result:** Passed, toll alerts triggered in real-time via WebSocket.
- **Status: Passed**

6.2.4 Test Case 4: Admin Toll Management

- **Description:** Verify admin can add, update, or delete toll records.
- **Input:** Admin login $\rightarrow$ add a new toll with details $\rightarrow$ update it $\rightarrow$ delete it.
- **Expected Result:** Toll is added, updated, and deleted successfully.
- **Actual Result:** Passed, operations reflected correctly in the database.
- **Status: Passed**

6.2.5 Test Case 6: Performance Under Load

- **Description:** Test system performance under simulated high traffic.
- **Input:** 1000+ concurrent socket connections.
- **Expected Result:** System handles requests with no crash or significant delay.
- **Actual Result:** Passed, system remained stable and responsive.
- **Status: Passed**

All test cases passed successfully, confirming that the Online Toll Notification System performs reliably, meets functional requirements, and is ready for deployment under real-world conditions.

Chapter 7

PROJECT MANAGEMENT

7.1 Team Member Details

Student Name	Class & Group	Expertise Area	Role in Project
Harshit Kumar Jain	7ITA-G2	Frontend, UI/UX design	Team Leader, Front-end Developer
Himansh Singh	7ITA-G2	Backend	Back-end Developer
Chirag	7ITA-G2	Database	Database Administrator

Table 7.1: Roles and Responsibilities of Team Members

7.2 Roles of Team Members

The successful completion of the Toll Notification System was achieved through collaborative efforts. Each member contributed expertise to critical project areas:

- **Harshit Kumar Jain (Team Leader):**
  - Managed project timelines and coordination.
  - Designed the user interface for route planning and toll alerts.
  - Ensured seamless integration of Mapbox for real-time mapping.
- **Himansh Singh (Back-end Developer):**
  - Developed RESTful APIs for user authentication and toll data fetching.
  - Integrated Twilio for OTP verification and WebSocket for proximity alerts.
  - Optimized backend performance for real-time location tracking.

- **Chirag (Database Administrator):**
  - Designed MongoDB schemas for User, TollPlaza, and NotificationLog.
  - Implemented geospatial queries for toll proximity calculations.
  - Ensured data integrity and security for sensitive user information.

7.3 Timeline and Milestones

The project was executed in phases with key milestones:

- **Phase 1: Project Planning** (Week 1-2)
  - Finalized scope: Real-time toll alerts, OTP login, admin dashboard.
  - Identified APIs: Mapbox (routing), Overpass (toll data), Twilio (SMS).
- **Phase 2: System Design** (Week 3-4)
  - Designed ERD for User, TollPlaza, and NotificationLog.
  - Created wireframes for user and admin interfaces.
- **Phase 3: Core Development** (Week 5-7)
  - Implemented OTP authentication.
  - Built route planning with Mapbox.
  - Set up WebSocket for real-time location updates.
- **Phase 4: Advanced Features** (Week 8-9)
  - Added proximity alerts (1-2km threshold).
  - Developed admin CRUD for toll plazas.
- **Phase 5: Testing** (Week 10-11)

  - Unit tests: OTP verification, distance calculation.
  - Integration tests: Mapbox + Overpass API.
  - Security tests: User data encryption.

- **Phase 6: Documentation** (Week 12)
  - Prepared final report (Use Case, DFD, ERD diagrams).
  - Compiled API documentation (Postman).

- **Phase 7: Deployment** (Week 13)
  - Deployed the final application to the production environment.
  - Conducted review: Verified real-time alert accuracy.

Chapter 8

CHALLENGES AND LIMITATIONS

8.1 Challenges

The development of the Online Toll Notification System presented several technical and operational challenges that required strategic problem-solving and iterative improvements. Key challenges included:

- **Real-Time Notification with WebSockets:** Implementing real-time toll alerts using WebSockets required maintaining persistent and stable connections between the client and server. Handling connection drops and ensuring reliable delivery of messages was complex.
- **Route Mapping with Toll Detection:** Automatically identifying toll plazas along dynamically generated routes (using Mapbox) involved geospatial computations and ensuring accuracy in toll placement relative to the route path.
- **OTP-Based Authentication:** Integrating OTP verification using third-party APIs (such as Twilio) required careful handling of request timing, retries, and securing sensitive data during the verification process.
- **Frontend Consistency:** Ensuring a visually consistent, responsive interface across all screens, especially in both light and dark modes, was a UI/UX challenge, particularly when adapting multiple components like login, registration, and OTP entry forms.

- **Data Synchronization Between Modules:** Syncing toll data, user activity, and notifications in real time across both user and admin dashboards posed challenges, especially during high-frequency updates.

8.2 Limitations

Despite the overall success and functional implementation of the system, certain limitations remain and are identified for future consideration and enhancement:

- **Limited Admin Features:** The admin panel currently supports basic operations like toll plaza management and user account handling. Advanced analytics, route usage statistics, and real-time user tracking are yet to be implemented.
- **Scalability Concerns with WebSockets:** While the current system handles moderate traffic effectively, scalability under large concurrent user loads needs further optimization, possibly through load balancers or distributed architecture.
- **Absence of Native Mobile App:** The system is designed as a web application and is mobile-responsive, but a native mobile app could improve usability for drivers who prefer in-app notifications and offline route caching.
- **No Offline Mode:** The system is entirely dependent on internet connectivity. Users cannot view toll data or receive notifications while offline.

Chapter 9

CONCLUSION AND FUTURE ENHANCEMENTS

The Online Toll Notification System has successfully addressed the need for a smarter, more efficient way to inform drivers about toll plazas along their travel routes. Designed as a modern web-based application, it provides a seamless experience for users to register, log in, plan routes, and receive real-time toll alerts using Mapbox and WebSocket technologies. The system also equips administrators with tools to manage toll data and monitor user activity.

Throughout the project development, essential functionalities such as OTP-based user authentication, toll plaza management, and real-time route monitoring were implemented. The use of contemporary technologies such as React.js for the frontend, Node.js with Express for the backend, MongoDB for database storage, and Twilio for OTP services has enabled the creation of a responsive, secure, and scalable platform.

While the current version of the system fulfills the core objectives, there is significant room for enhancement to further elevate its utility and scalability.

Future Enhancements

The following potential enhancements could be made to improve the system's overall performance and user experience:

- **Toll Payment Integration:** Currently, the system does not support toll fee

payment. Future versions could integrate digital payment gateways to enable real-time toll fee transactions directly from the app.

- **Dedicated Mobile App:** Developing native Android and iOS apps would provide a more convenient and optimized experience for users on the go, particularly for those driving through toll routes frequently.
- **Voice and Push Notifications:** Implementing voice-based alerts or push notifications would improve driver safety and ensure that toll alerts are not missed while in transit.
- **Enhanced Admin Dashboard:** Additional features such as toll traffic analytics, heat maps, and route usage reports could be integrated into the admin panel to assist in operational planning.
- **Offline Route Caching:** Allowing users to cache route data and receive toll notifications even when offline would improve reliability in areas with poor network coverage.
- **Multilingual Support:** Supporting regional languages will make the application more accessible to a broader range of users across different states and regions in India.
- **Vehicle Type-Based Toll Info:** The system could be extended to provide toll fees based on vehicle categories (e.g., car, truck, bus), enabling more accurate cost estimation.
- **Scalability Optimization:** Enhancing WebSocket infrastructure and backend logic to support larger user bases and simultaneous connections, especially for highway authorities or logistics companies.

Incorporating these enhancements will significantly improve the system's functionality, user engagement, and adaptability, allowing it to better meet the needs of both users and administrators in an evolving digital landscape.

Conclusion

In conclusion, the Online Toll Notification System successfully delivers a real-time, interactive toll alerting platform that enhances the driving experience by informing users in advance of upcoming toll plazas. The project has met its initial objectives of improving route awareness, user engagement, and toll data management.

The development process provided significant technical learning opportunities in areas such as geolocation-based services, real-time communication, user authentication, and full-stack development. The platform lays a strong foundation for further innovation in the intelligent transportation domain.

With the integration of proposed future features, this system has the potential to evolve into a fully-fledged smart toll management platform—streamlining operations for highway authorities and offering a safer, more convenient experience for commuters and logistics providers alike.

REFERENCES

1. React, ”Homepage,” [Online]. Available: https://reactjs.org/. [Accessed: Apr. 15, 2025].

2. Node.js, ”Homepage,” [Online]. Available: https://nodejs.org/. [Accessed: Apr. 15, 2025].

3. Express.js, ”Homepage,” [Online]. Available: https://expressjs.com/. [Accessed: Apr. 16, 2025].

4. MongoDB, ”Homepage,” [Online]. Available: https://www.mongodb.com/. [Accessed: Apr. 18, 2025].

5. Mapbox, ”Homepage,” [Online]. Available: https://www.mapbox.com/. [Accessed: Apr. 20, 2025].

6. Twilio, ”Homepage,” [Online]. Available: https://www.twilio.com/. [Accessed: Apr. 22, 2025].

7. GitHub, ”Homepage,” [Online]. Available: https://github.com/. [Accessed: Apr. 10, 2025].

8. WebSocket.org, ”WebSocket Overview,” [Online]. Available: https://www.websocket.org/. [Accessed: Apr. 25, 2025].

9. Tailwind CSS, ”Homepage,” [Online]. Available: https://tailwindcss.com/. [Accessed: Apr. 12, 2025].

10. Visual Studio Code, ”Homepage,” [Online]. Available: https://code.visualstudio.com/. [Accessed: Apr. 10, 2025].

11. npm, ”Homepage,” [Online]. Available: https://www.npmjs.com/. [Accessed: Apr. 15, 2025].