Design Document

for

Ride With Us

Version 1.0

Prepared by

Group #: 20 **Group Name: TITANS**

AYUSH YADAV	210251	ayushy21@iitk.ac.in
HARSHIT PATEL	210424	harshitptl21@iitk.ac.in
JATOTH SHASHI VARDAN	230501	shashivj23@iitk.ac.in
KANDULA AMARNADHU	230522	amarkn23@iitk.ac.in
PAL AJAY RAMSAGAR	230725	palajayr23@iitk.ac.in
ROHIT VINOD ATKURKAR	230872	rohitv23@iitk.ac.in
SANGA BADRI	230911	sangabadri23@iitk.ac.in
SUGALI YASHWANTH NAIK	231046	synaik23@iitk.ac.in
SUNANDINI BANSAL	EXY24032	sunandinib@iitk.ac.in
V HARIVANSH	231109	vhari23@iitk.ac.in

Course: CS253

Mentor TA: ASHISH SINGH (ashishsg24@iitk.ac.in)

Date: 07/02/2025

Contents II

Revisions II

1 Context Design 1

1.1 Context model 1
1.2 human Interface Design 1

2 Architecture design 5

3 object-oriented design 6

3.1 Use case diagram 6
3.2 Class diagram 7
3.3 sequence diagram 8
3.4 State diagram 9

4 Project Plan 10

5 Other Details 11

Appendix A - Group Log 12

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	AYUSH YADAV HARSHIT PATEL JATOTH SHASHI VARDAN KANDULA AMARNADHU PAL AJAY RAMSAGAR ROHIT VINOD ATKURKAR SANGA BADRI SUGALI YASHWANTH NAIK SUNANDINI BANSAL V HARIVANSH	Initial Designs of the project	07/02/2025

1 Context Design

1.1 Context Model

The context model defines how Ride With Us interacts with external entities and manages data flow.

External Entities:

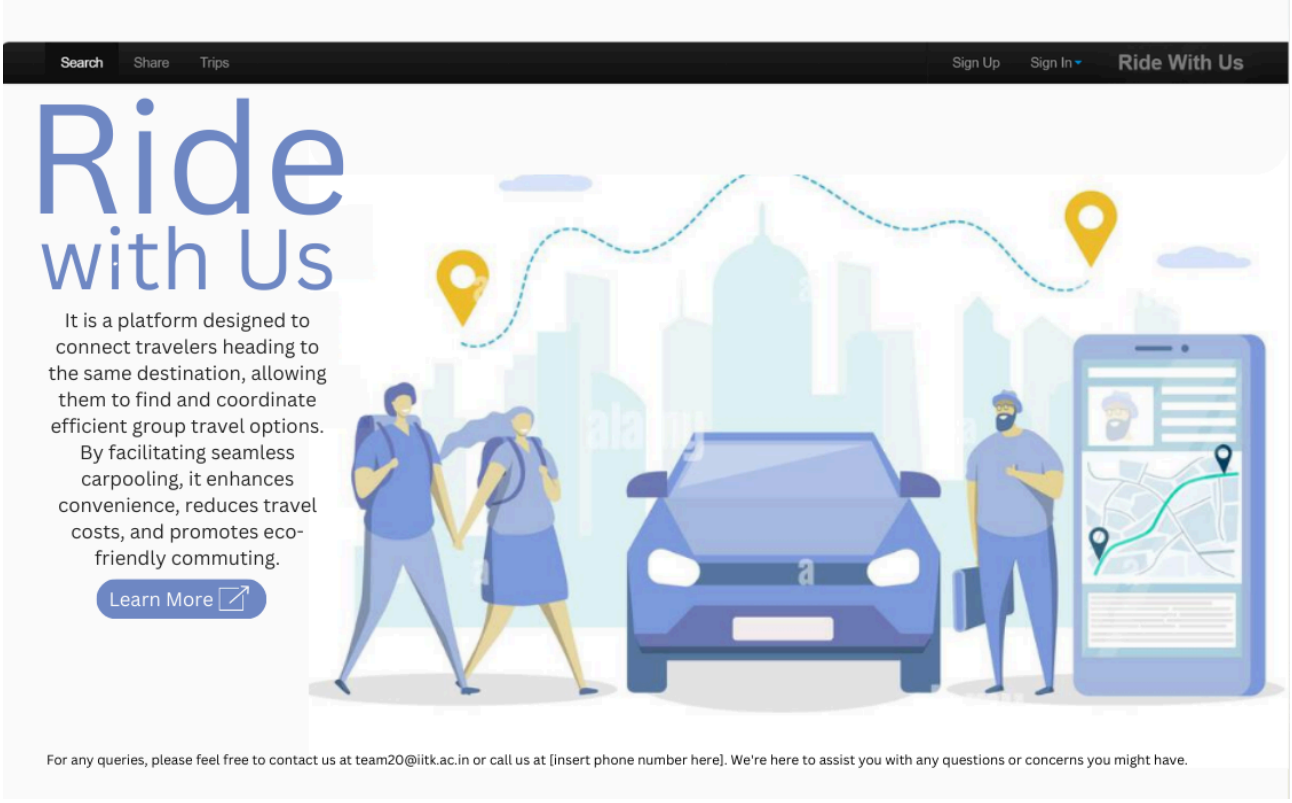
- Users (Drivers & Passengers): Share and search for rides.
- Database (MySQL): Stores user data, ride details, and bookings.
- Mapping API (Leaflet): Displays routes and ride locations.
- Web Server (PHP & AJAX): Handles user requests, authentication, and real-time updates.

Data Flow:

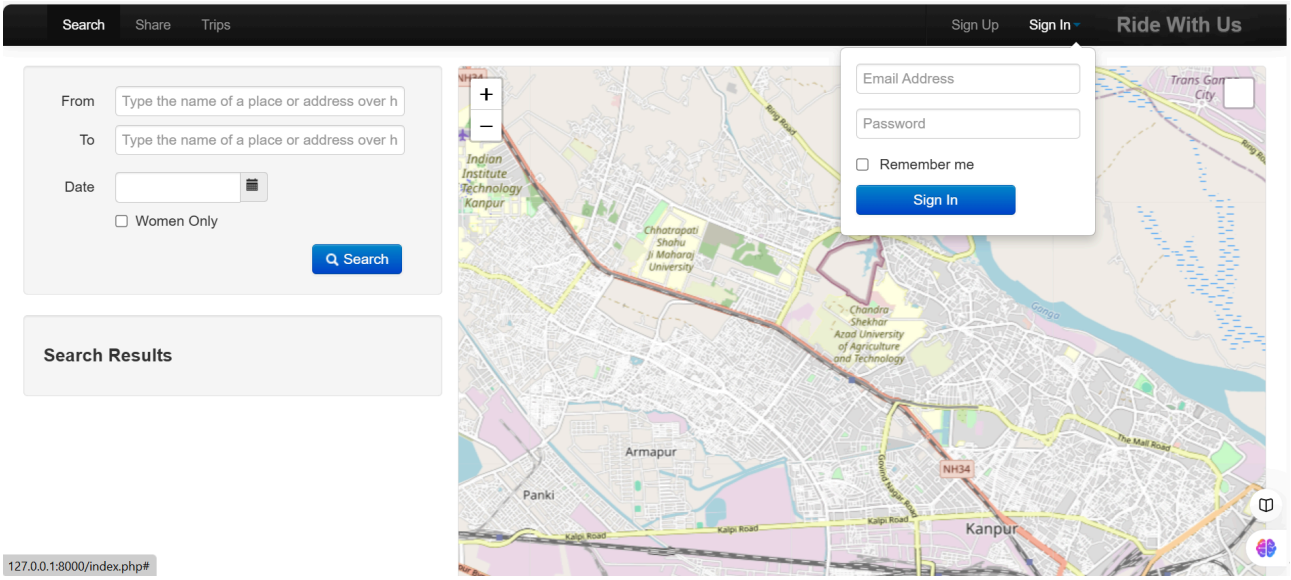
- Users interact via the web interface.
- Ride data is processed via PHP and stored in MySQL.
- Maps display location-based ride details using Leaflet.
- AJAX ensures dynamic updates for ride availability.

1.2 Human Interface Design

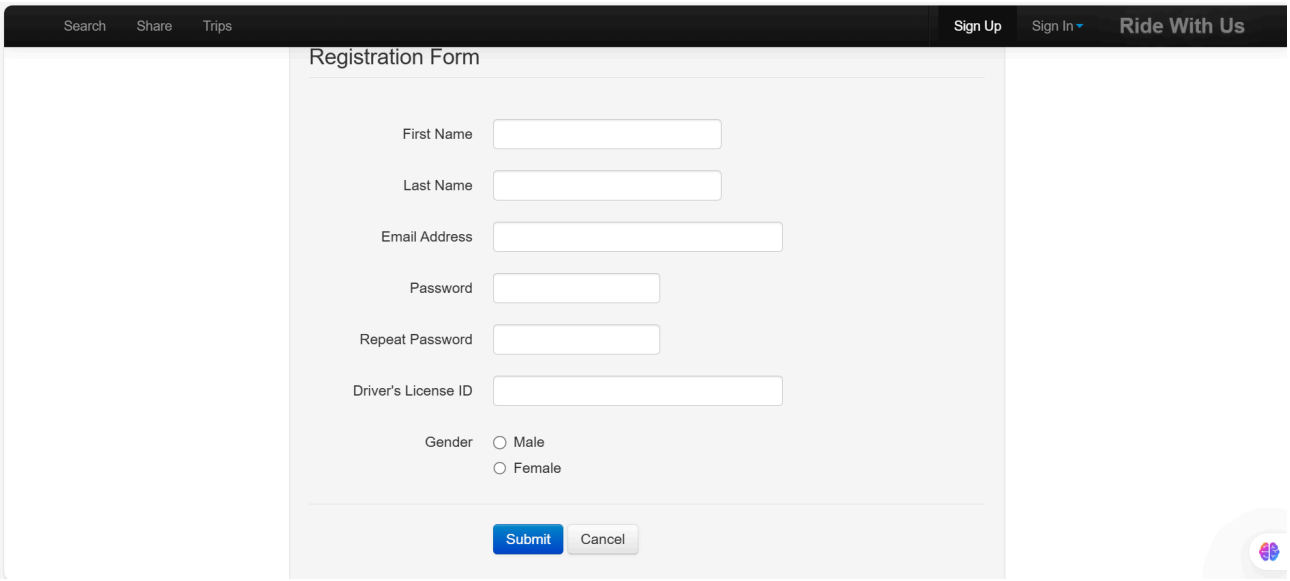
1.2.1 Home Page:



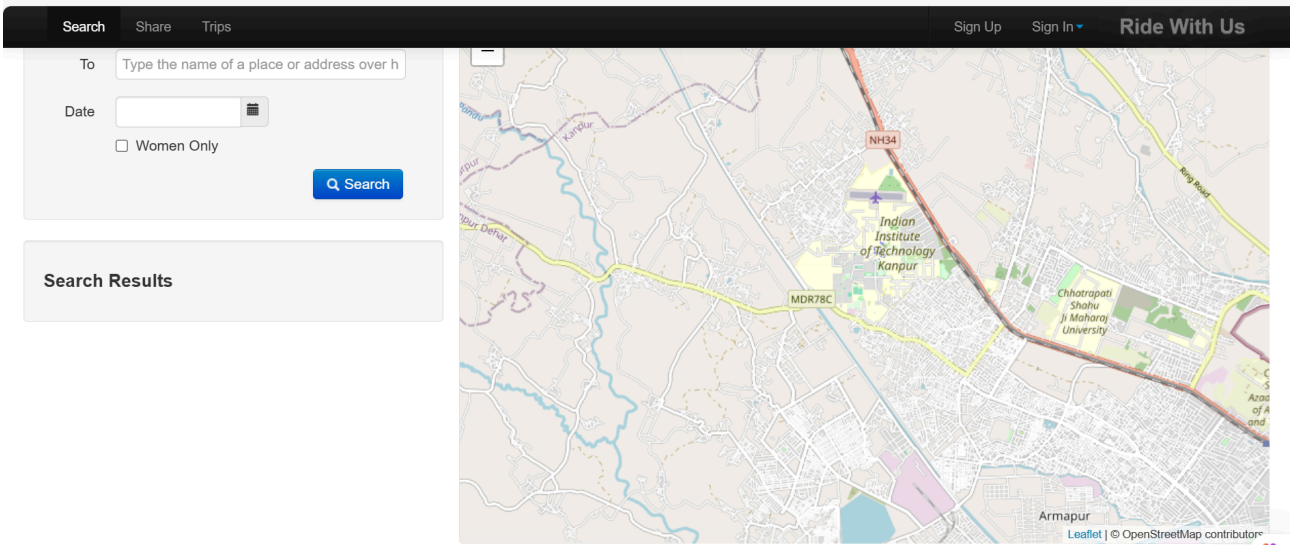
1.2.2 Sign in Page:(Pop Up)



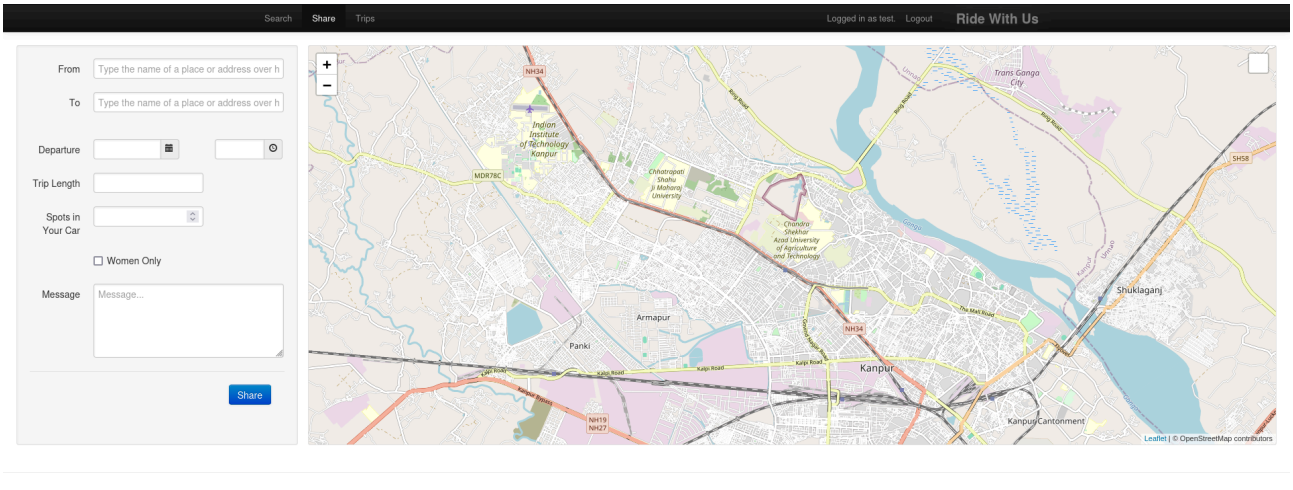
1.2.3 Sign up:



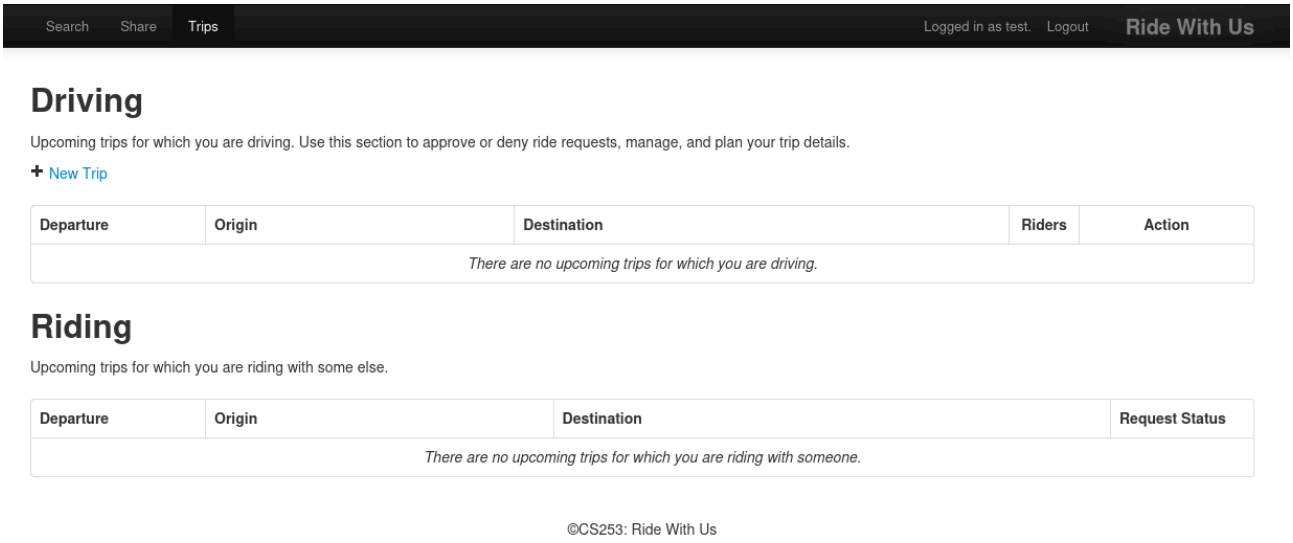
1.2.4 Search Page:



1.2.5 Share Page:



1.2.6 Trips Page:

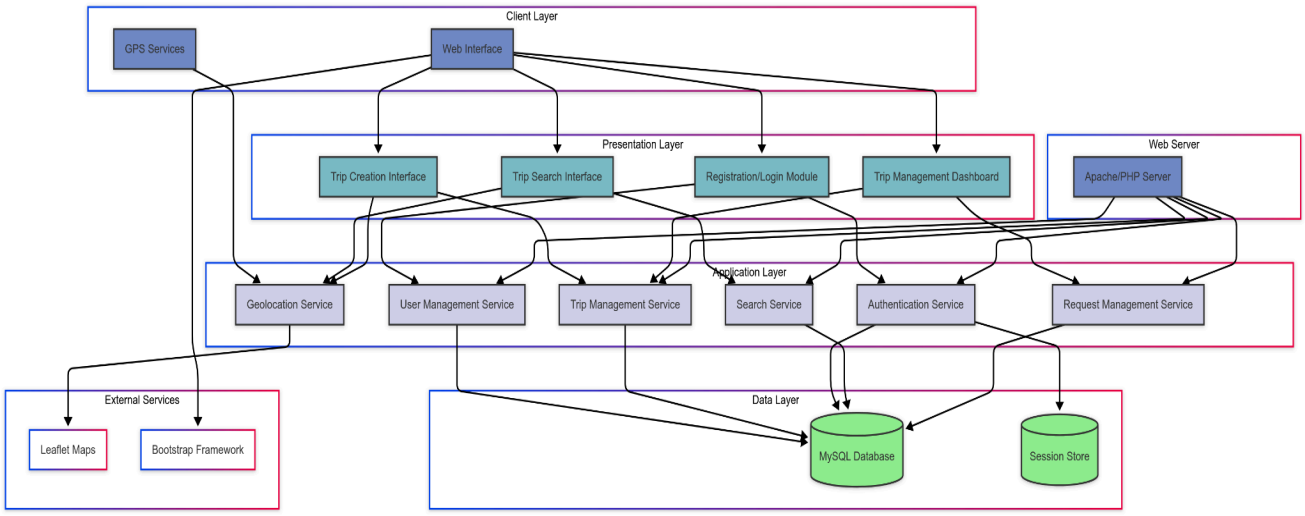


2 Architecture Design

We have chosen the “layered architecture model” for Ride With Us because it provides a clear separation of concerns, making the system more modular, maintainable, and scalable.

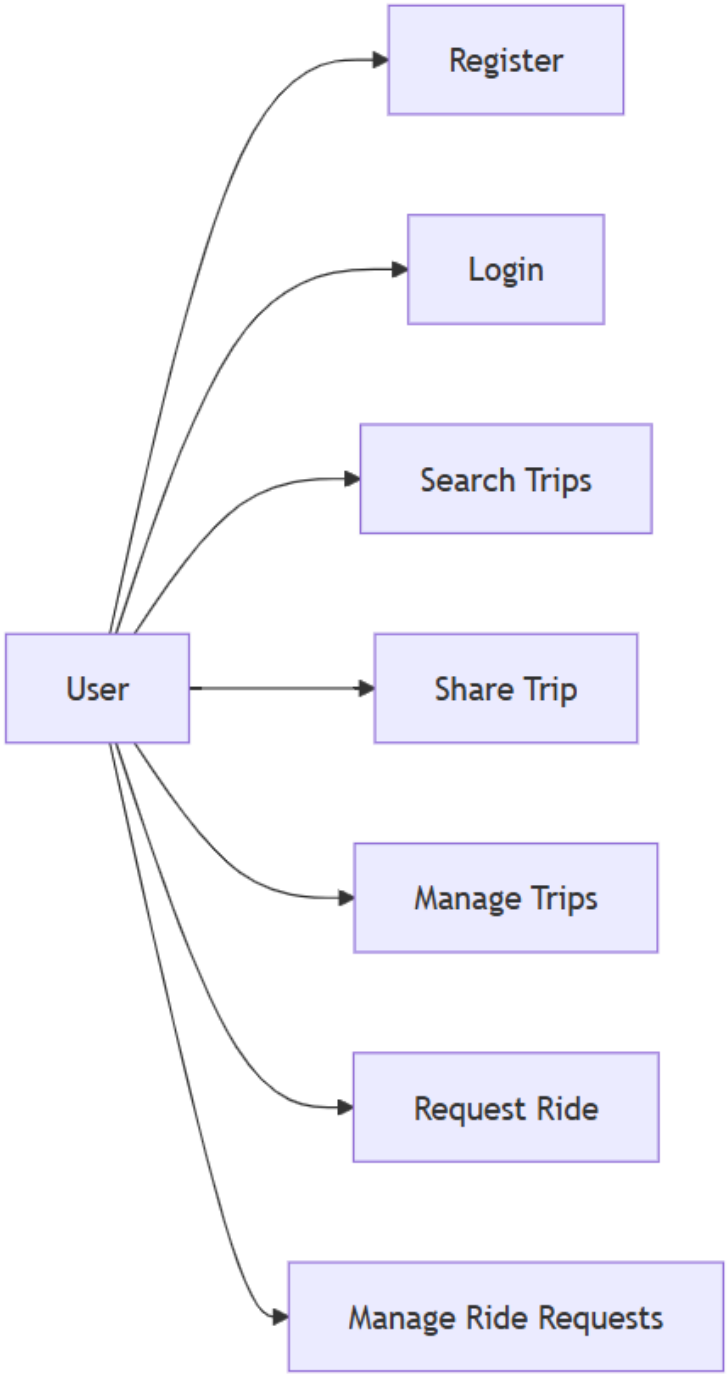
This architecture aligns well with our project's “Model-View-Controller (MVC)” approach, ensuring that each layer has a distinct responsibility. The client layer handles user interactions, the presentation layer manages UI components, the application layer processes business logic, and the data layer ensures efficient storage and retrieval of ride and user information. By structuring the system in this way, we enhance code reusability, ease of debugging, and future scalability, allowing seamless integration of additional features like notifications or ride history. Additionally, using a dedicated authentication service for IITK webmail verification strengthens security, while external services like Leaflet Maps improve usability. This model and diagram effectively illustrate how different system components interact, ensuring a well-organized and efficient implementation of Ride With Us.

ARCHITECTURE DESIGN

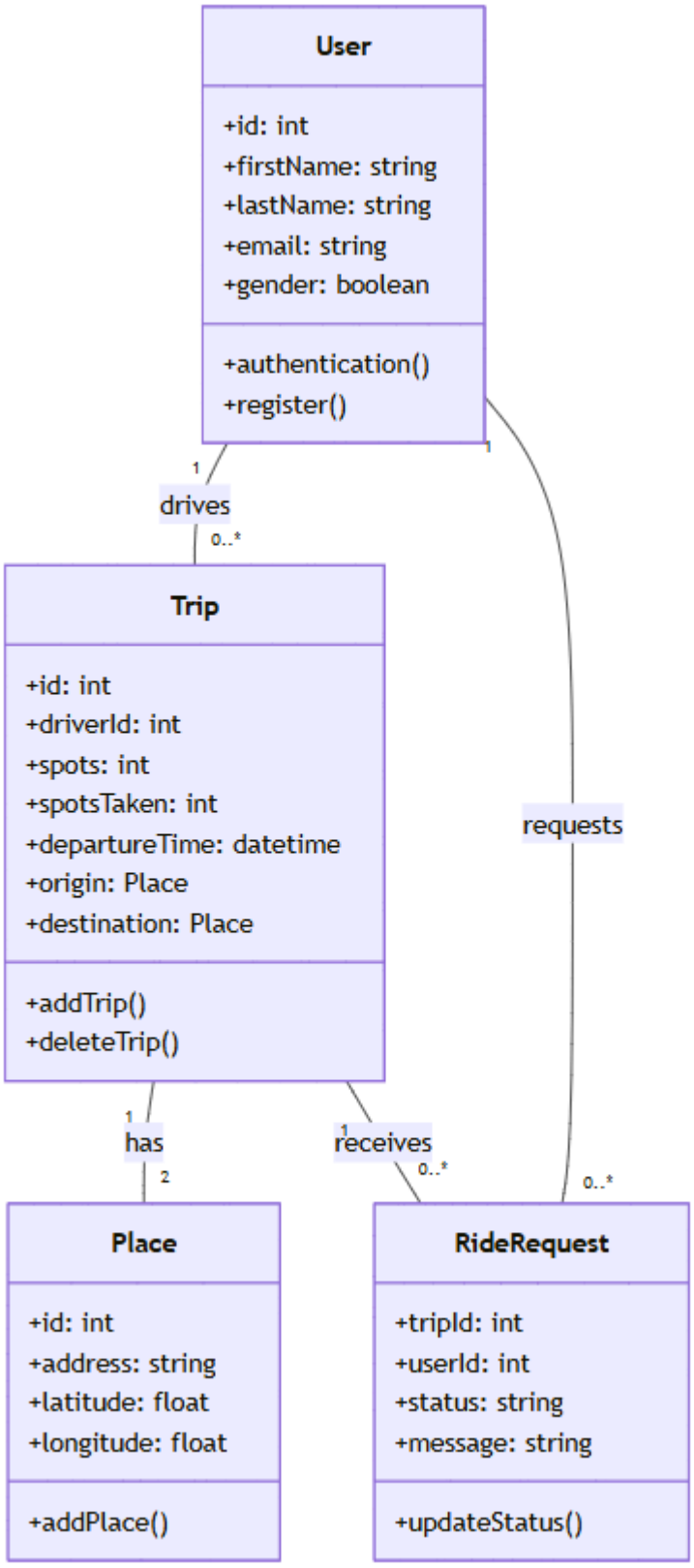


3 Object Oriented Design

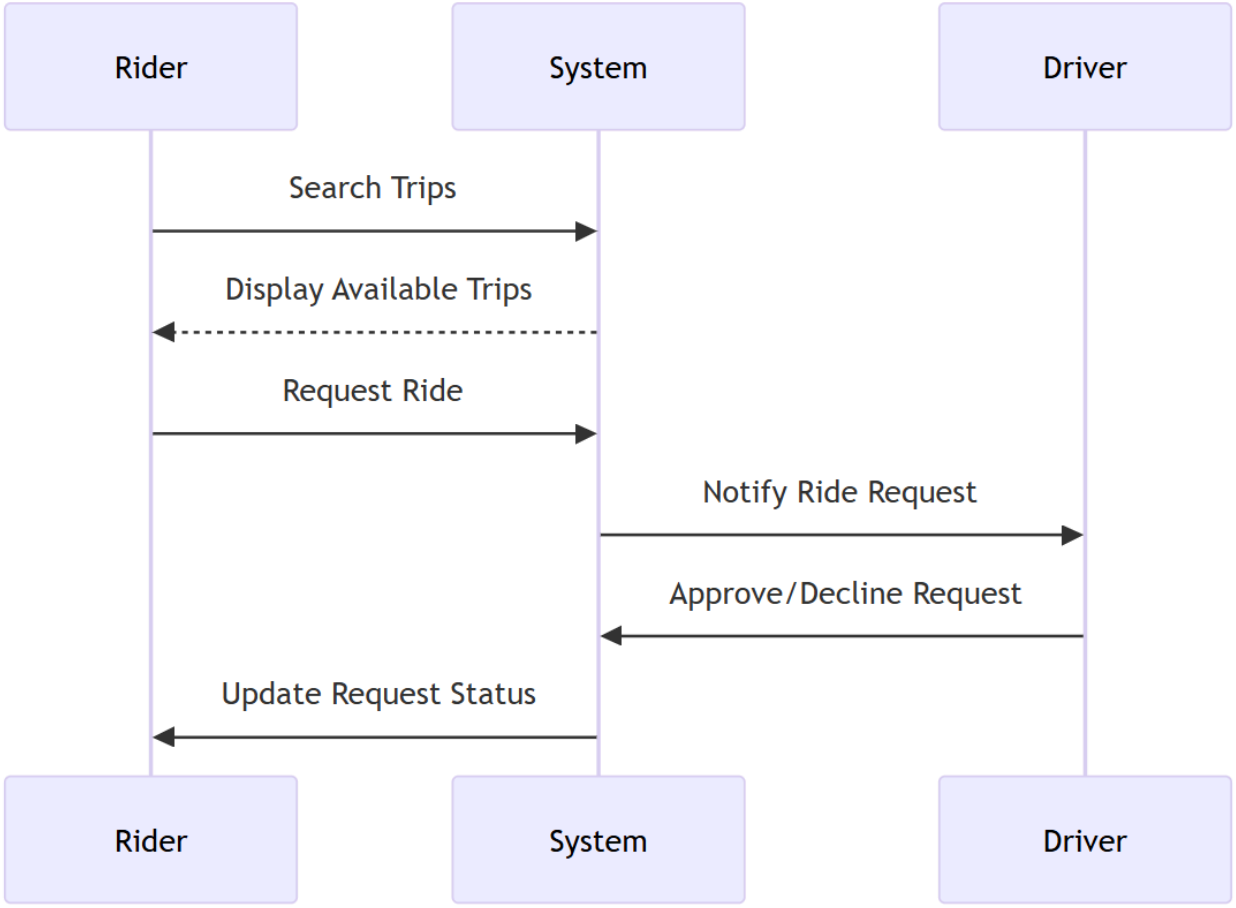
3.1 Use Case Diagrams



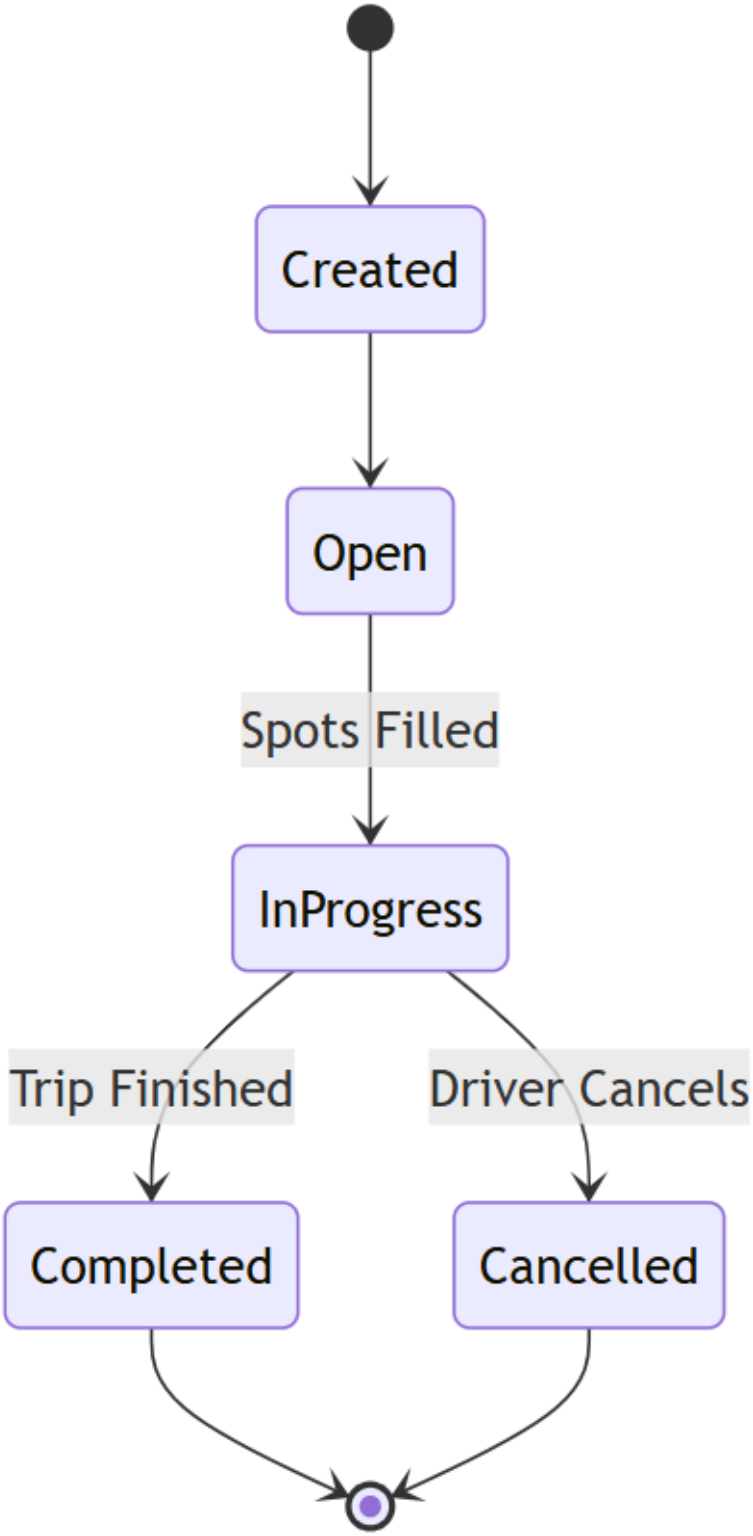
3.2 Class Diagrams



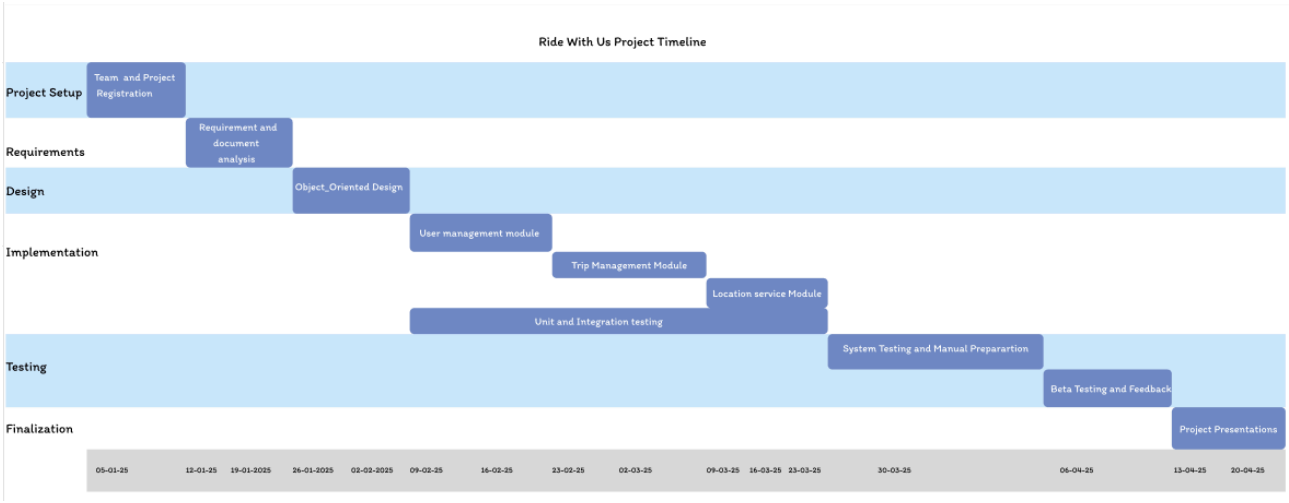
3.3 Sequence Diagrams



3.4 State Diagrams



4 Project Plan



Project Phases and Tasks:

Project Setup (Early January - Mid-January)

- The project kicks off with team and project registration, establishing the foundation for collaboration and role assignments.

Requirements (Mid-January - Late January)

- A crucial phase involving requirement and document analysis, ensuring clarity in project goals and functionalities.

Design (Late January - Mid-February)

- Object-Oriented Design is created to define system architecture.
- Development begins with the User Management Module, focusing on user authentication and profiles.

Implementation (February - Late March)

- The Trip Management Module is developed to handle ride booking and scheduling.
- The Location Service Module integrates GPS tracking and mapping features.

Testing (Mid-March - Mid-April)

- Unit and Integration Testing ensures that individual components and modules function correctly together.
- System Testing and Manual Preparation validate the system's overall performance and usability.
- Beta Testing and Feedback help refine the system based on user inputs.

Finalization (Late April)

- The project concludes with Project Presentations, showcasing the final product and its features.

5 Other Details

5.1 Key Technical Choices:

PHP for server-side logic – Ensures dynamic content generation, seamless user authentication, and efficient request handling. Widely supported with extensive libraries for web applications.

MySQL for data storage – Provides a reliable and scalable relational database system, ensuring efficient data retrieval and secure storage. Supports large datasets with optimized indexing.

Leaflet.js for mapping – Lightweight and highly customizable mapping library, ideal for displaying real-time location data. Works efficiently across devices with interactive map features.

Bootstrap for frontend design – Simplifies responsive web development with a mobile-first approach. Offers a vast collection of pre-styled components for a modern and clean UI.

AJAX for dynamic interactions – Enables smooth, asynchronous data updates without page reloads. Enhances user experience by providing real-time feedback and interactivity.

5.2 Security Measures:

Password encryption – Safeguards user credentials by securely hashing and storing passwords, preventing unauthorized access.

Input sanitization – Reduces vulnerabilities by filtering user input, preventing cross-site scripting (SHA-256) and other attacks.

Session management – Ensures secure user sessions by properly handling authentication tokens and preventing session hijacking.

Prepared statements to prevent SQL injection – Protects the database from malicious queries by separating SQL logic from user input, preventing data breaches.

5.3 Scalability Potential:

Modular design – Allows easy expansion and maintenance by organizing code into independent, reusable components.

Potential for adding features like:

- **Rating system** – Enhances trust and reliability by allowing users to rate and review their travel experiences.
- **Payment integration** – Enables seamless transactions for cost-sharing, making carpooling more convenient and efficient.
- **More advanced matching algorithms** – Improves ride-sharing accuracy by considering user preferences, past behavior, and real-time availability.

Appendix A - Group Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document>