International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Riding Partner Sharing System

DR.T.JASPERLINE [1], RUTHRATHNAKUMARI A[2], SIVA S[3],BELL YABAS T [4]

[1]Professor -Department of Computer Science and Engineering & Dr.G.U.Pope College of Engineering-India.
[2]Assistant Professor -Department of Computer Science and Engineering & Dr.G.U.Pope College of Engineering-India.
[3,4]Department of Computer Science and Engineering & Dr.G.U.Pope College of Engineering-India.

Abstract - The **Riding Partner Sharing System** is a web-based application designed to help individuals find and connect with others traveling on similar routes. This system aims to promote carpooling and reduce traffic congestion, fuel consumption, and environmental impact. By entering details such as start location, destination, travel date, and time, users can either create a ride or search for available rides that match their route. The platform allows users to view profiles, contact potential ride partners, and confirm ride-sharing arrangements.

Developed using **Python (Flask)** for backend processing and **JavaScript** for client-side interaction, the application maintains a user-friendly interface for smooth navigation and real-time matching. The system stores ride data securely in a backend database and uses basic matching logic to recommend suitable partners based on location and time preferences. This project serves as a prototype for sustainable urban commuting and demonstrates the integration of web technologies to solve real-world mobility problems.

1. INTRODUCTION

In recent years, urban areas have witnessed a significant rise in the number of vehicles on the road. This increase has led to several challenges such as traffic congestion, longer travel times, higher fuel consumption, and greater environmental pollution. Most personal vehicles are underutilized, often carrying only one passenger, which contributes to inefficient use of transport resources. To address these issues, ride-sharing has emerged as a practical and sustainable solution.

The **Riding Partner Sharing System** is a web-based platform designed to connect people who are traveling on the same route. It allows users to register their trip details, including the starting point, destination, travel time, and date. Based on this information, the system matches users with others who have similar travel plans, offering an easy way to share rides. This not only helps reduce the number of vehicles on the road but also promotes cost-saving, social interaction, and environmental benefits.

The system is developed using **Python with Flask** for the backend to handle data and logic, and **HTML, CSS, and JavaScript** for the frontend to create a responsive and user-friendly interface. Users can create ride offers or search for available rides, and the platform displays the best possible matches. This project demonstrates how simple web technologies can be combined to create a functional, real-world application that supports sustainable transportation.

1.1 Domain Introduction

The **Riding Partner Sharing System** is a modern solution designed to connect individuals who travel similar routes and wish to share their journey for convenience, cost-effectiveness, and environmental benefits. This system primarily serves as a **ride-matching platform**, allowing users to find and connect with other commuters or travelers who are heading in the same direction.

As urban populations grow and transportation costs rise, the need for **efficient ride-sharing solutions** has become increasingly important. Traditional ride-hailing services often focus on passenger-to-driver relationships, whereas this system emphasizes **peer-to-peer** connections — enabling users to share rides as **co-passengers** or **alternating drivers** in private or shared vehicles.

This domain integrates aspects of:

- **Location-based services** (to find nearby riders),
- **User profiling and preferences** (to match compatible partners),
- **Scheduling systems** (to plan and coordinate rides),
- **Secure communication** (chat or call between matched users),
- **Review and rating systems** (to ensure trust and safety).

The system is particularly useful for **daily commuters**, **college students**, **corporate employees**, and **long-distance travelers** who want to reduce travel costs, minimize vehicle usage, and promote sustainable transportation practices.

1.2 Objectives

The main objectives of this project are:

- **To provide a platform** where users can find and connect with riding partners who share similar travel routes and timings.
- **To reduce travel costs** by enabling users to share expenses such as fuel, toll, or parking fees through ride-sharing.
- **To improve travel convenience** by allowing users to schedule, manage, and coordinate shared rides easily using the system.
- **To promote environmental sustainability** by reducing the number of vehicles on the road, thereby lowering carbon emissions and traffic congestion.
- **To ensure safety and trust** through features like user verification, rating and review systems, and in-app communication.

- **To support daily commuters** such as students, employees, and travelers in managing their transportation needs more efficiently.
- **To build a community-based travel network** where users can interact, share travel updates, and form regular ride-sharing partnerships.

1.3 Scope of the Project

The Riding Partner Sharing System is designed to create a user-friendly platform that connects individuals with similar travel routes and timings. The primary focus is to simplify the process of ride-sharing among users for daily commuting or long-distance travel. The system will allow users to register, create profiles, set preferred travel routes and schedules, and find matching partners for shared rides.

The scope includes features such as location-based ride matching, real-time tracking, secure in-app communication, and a feedback/rating system to ensure trust and reliability. The platform can be accessed via web or mobile applications, providing convenience to users on different devices.

This project aims to benefit students, employees, and travelers by offering an economical and eco-friendly alternative to solo travel. It also lays the foundation for future integration of advanced features like fare splitting, route optimization using AI, and integration with public transportation systems.

2.SYSTEM ANALYTICS

The Riding Partner Sharing System uses analytics to monitor user behavior, ride-matching efficiency, and system performance. It tracks data such as the number of ride requests, successful matches, user activity, peak travel times, and route preferences. This information helps in optimizing ride suggestions, improving user experience, and identifying areas for system improvement. Basic statistics and visual reports can also assist admins in understanding usage patterns and managing the platform effectively

2.1 Existing Problem

In today's fast-paced world, daily commuting has become a major challenge, especially in urban and semi-urban areas. A large number of people, including students, office workers, and employees, travel similar routes every day using individual vehicles or unreliable public transportation. This leads to several ongoing problems that affect both individuals and society as a whole.

One of the primary issues is the underutilization of vehicle capacity. Most private vehicles on the road carry only one person, while the vehicle has space for three or four more passengers. This results in a massive waste of fuel and resources. It also contributes to increased traffic congestion, especially during peak hours. Roads become overcrowded, travel times increase, and people end up spending a significant portion of their day stuck in traffic. This not only reduces productivity but also leads to mental stress and fatigue.

Another key problem is the financial burden of solo travel. Fuel prices, parking charges, and vehicle maintenance costs are constantly rising. For daily commuters, especially students and middle-income individuals, these expenses can be hard to manage. At the same time, public transport is often overcrowded, irregular, or unavailable in many areas. People are forced to either depend on expensive ride-hailing services or travel long distances on their own.

Furthermore, existing ride-sharing platforms mostly operate on a commercial driver-to-passenger model (like Uber or Ola), where the user is dependent on a paid driver. These platforms do not support true peer-to-peer ride sharing, where ordinary users can connect with one another to share rides as equals. There is also a lack of trust and safety features in informal ride-sharing, making people hesitant to share rides with strangers without a verified system in place.

In summary, the lack of a dedicated, safe, and user-friendly platform for connecting daily travelers with similar routes leads to increased costs, traffic, pollution, and commuter stress. There is a clear need for a system that allows users to find reliable riding partners easily and promotes efficient, eco-friendly travel.

2.2 Proposed Methodology

The proposed Riding Partner Sharing System aims to provide a user-friendly, efficient, and secure platform for individuals to find and share rides with others traveling along the same or nearby routes. The methodology follows a structured approach involving several key phases: requirement analysis, system design, implementation, testing, and deployment.

Requirement Analysis
In this phase, both functional and non-functional requirements of the system are collected. Functional requirements include user registration, login, profile management, ride creation, ride search, matching algorithm, messaging, and rating systems. Non-functional requirements cover performance, security, and scalability. User feedback and surveys may be used to understand user expectations and behavior.

System Design
The system will be designed using a modular architecture. The core components include:

- User Module: Registration, login, profile, and preferences.
- Ride Management Module: Creating rides, searching for rides, matching based on route and time.
- Communication Module: In-app chat or message system for coordination.
- Admin Module: Managing users, monitoring system activity, handling reports and reviews.

The system design also includes database schema design for storing user details, ride information, and history.

Implementation
The platform will be developed using modern web technologies (e.g., HTML, CSS, JavaScript for frontend; PHP, Python, or Node.js for backend). The backend server

will handle logic for matching rides and managing user data, while the frontend provides an intuitive interface. GPS and mapping APIs (like Google Maps) will be integrated for real-time location and route suggestions.

Testing
Various levels of testing will be performed, including unit testing (individual components), integration testing (modules working together), and user acceptance testing (real users interacting with the system). Security testing will also be conducted to ensure data protection and safe communication between users.

Deployment and Maintenance
Once testing is successful, the system will be deployed on a web server or cloud platform. User feedback will be continuously collected to make improvements. Regular maintenance will ensure system performance, bug fixing, and updates based on evolving user needs.

This structured methodology ensures the system is robust, scalable, and user-friendly, and directly addresses the problems of costly solo travel, lack of ride-sharing opportunities, and traffic congestion. It encourages safe and community-driven transportation.

3.SYSTEM REQUIREMENTS

3.1. Hardware Requirements

- A server or cloud hosting environment to deploy the application backend and database.
- User devices such as smartphones, tablets, or computers with internet connectivity.
- GPS-enabled devices for location tracking (smartphones, tablets).

3.2. Software Requirements

Frontend:

- Web technologies: HTML5, CSS3, JavaScript (frameworks like React, Angular, or Vue.js can be used optionally).
- Mobile app platforms (optional): Android (Java/Kotlin), iOS (Swift) if mobile apps are developed.

Backend:

- Server-side language: Python (Django/Flask), Node.js, PHP, or Java.
- Database management system: MySQL, PostgreSQL, or MongoDB.
- RESTful API to communicate between frontend and backend.
- Integration with mapping and location services (e.g., Google Maps API).

Other Tools:

- Version control system: Git/GitHub or GitLab.
- Testing frameworks for automated and manual testing.
- Security tools for authentication (OAuth, JWT) and data protection.

3.3. Functional Requirements

- User registration and authentication.
- Profile management with travel preferences.
- Ride creation and scheduling.
- Search and match rides based on location and time.
- In-app communication between matched users.
- Rating and feedback system.
- Admin dashboard for monitoring users and activities.

3.4. Non-Functional Requirements

- Performance: The system should provide fast response times for ride searches and matching.
- Scalability: Able to support growing numbers of users without performance loss.
- Security: Secure user data and communication, with proper authentication and encryption.
- Usability: Simple and intuitive user interface for easy navigation.
- Reliability: High uptime with minimal downtime.
- Privacy: Ensure user location and personal data are handled with privacy considerations.

4.MODULES AND UML DIAGRAMS

4.1. Modules of the System

The Riding Partner Sharing System is divided into several key functional modules:

4.1.1 User Module

- User registration and login
- Profile creation and editing
- Set travel preferences (route, time, seat availability)

4.1.2 Ride Management Module

- Create new ride entries
- Search for available rides based on source, destination, and time
- Request to join a ride

4.3 Matching Module

- Algorithm to match users based on route similarity, time, and preferences
- Prioritizes users based on distance and rating

4.4 Communication Module

- In-app messaging or contact options for matched users
- Ride confirmation notifications

4.5 Rating and Feedback Module

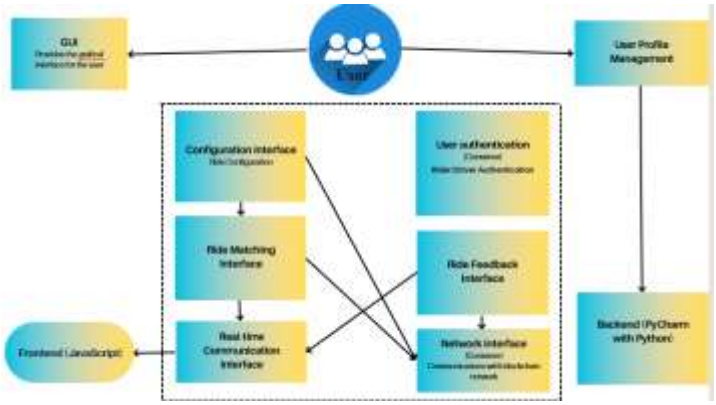
- After ride completion, users can give feedback
- Rating system helps build trust

4.6 Admin Module

- Monitor user activity
- Manage reported users or issues
- Control over system settings and ride data

4.2. UML Diagrams

Below are the UML diagrams that represent the structure and behavior of the system:



4.2 Use Case Diagram

Purpose: Shows interactions between users and system functionalities.

Actors:

- User
- Admin

Use Cases:

- Register/Login
- Create Ride
- Search Rides
- Request Ride
- Chat with Partner
- Rate Ride
- View User Reports (Admin)
- Manage Rides (Admin)

4.2.2 Class Diagram

Purpose: Describes the structure of the system by showing system classes and their relationships.

Main Classes:

- User: userId, name, email, password, preferences
- Ride: rideId, source, destination, time, driverId, seatsAvailable
- Match: matchId, rideId, userId, status
- Message: messageId, senderId, receiverId, content, timestamp
- Rating: ratingId, fromUserId, toUserId, rating, comment

Relationships:

- One User can create many Rides
- One Ride can have multiple Matches
- User sends and receives many Messages
- User gives and receives Ratings

4.2.3 Sequence Diagram

Use Case Example: Book a Ride

- User logs in
- User searches for rides
- System matches rides
- User sends ride request
- Driver accepts the request
- Ride is confirmed

4.2.4 Activity Diagram

Use Case Example: Ride Matching Process

- Start → Enter route → Search rides → Match results → Send request → Wait for approval → Confirm ride → End

4.2.5 Data Flow Diagram (Level 1)

- Shows how data moves between user input, matching logic, and output (confirmed ride)

5. IMPLEMENTATION

The **Riding Partner Sharing System** was implemented using modern web technologies and follows a modular development approach. The goal of implementation is to convert the system design into a working software solution that is efficient, user-friendly, and scalable.

5.1. Technology Stack

Frontend:

- **HTML5, CSS3** – For structuring and styling web pages.
- **JavaScript** – For interactivity and user input validation.
- **Bootstrap** – To create responsive, mobile-friendly layouts.

Backend:

- **PHP / Python (Flask/Django)** – For server-side logic, user authentication, ride matching, and data processing.
- **MySQL / PostgreSQL** – For database storage of user profiles, ride details, messages, and ratings.

APIs & Tools*:*

- **Google Maps API** – For location services and route display.
- **JWT (JSON Web Tokens)** – For secure user authentication.
- **Git** – For version control and collaborative development.

5.2. Implementation Steps

a. User Authentication:

- Created a login and registration system with secure password hashing.
- Users must verify their email/mobile number before accessing the platform.

b. Ride Creation and Search:

- Logged-in users can create ride offers by entering route details, travel time, and seat availability.
- Other users can search for rides by entering their source and destination. A matching algorithm suggests rides based on location proximity and time overlap.

c. Matching Algorithm:

- The backend compares entered source/destination with existing rides using a basic location radius and time window logic.
- Future improvements may include integrating Google Distance Matrix API for smarter route matching.

d. Communication:

- Once a ride is matched, users can send messages through an internal chat system or receive email notifications for ride confirmations.

e. Rating and Feedback:

- After a completed ride, users can rate their partner. Ratings are stored in the database and displayed on user profiles to help future users build trust.

f. Admin Panel:

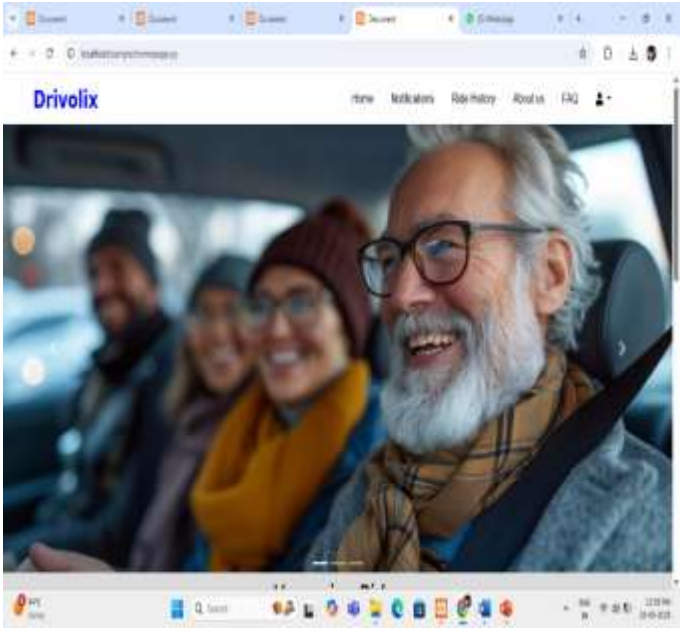
- An admin dashboard was developed to manage user accounts, moderate feedback, and handle reports or inappropriate behavior.

5.3. Testing and Debugging

- Each module was tested independently (unit testing) and then integrated.
- Common issues like form validation, session handling, and API failures were handled through proper error messages and logging.

5.4. Deployment

- The system was deployed on a local server for testing and demonstration.
- For live deployment, platforms like Heroku, Vercel, or cPanel-based hosting can be used.
- A domain name and SSL certificate will ensure secure access.



TESTING

Testing is a crucial phase in software development, ensuring that all components of the system work as expected and meet user requirements. The **Riding Partner Sharing System** underwent various levels of testing to validate its functionality, reliability, performance, and user experience.

1. Types of Testing Performed

1.1 Unit Testing

- Individual modules such as user login, ride creation, and ride search were tested separately.
- Ensured that each function works correctly in isolation.

- Example: Testing if the system allows only valid email formats during registration.

1.2 Integration Testing

- Verified that different modules work together properly.
- Example: Checked if a user who creates a ride can be matched and contacted by another user.

1.3 System Testing

- End-to-end testing of the complete application to verify it behaves as a whole.
- Simulated real-world usage scenarios such as ride creation, matching, and feedback submission.

1.4 User Acceptance Testing (UAT)

- Conducted with a group of sample users (e.g., students, office commuters) to gather feedback on usability.
- Verified whether the system meets real user expectations and is easy to navigate.

1.5 Security Testing

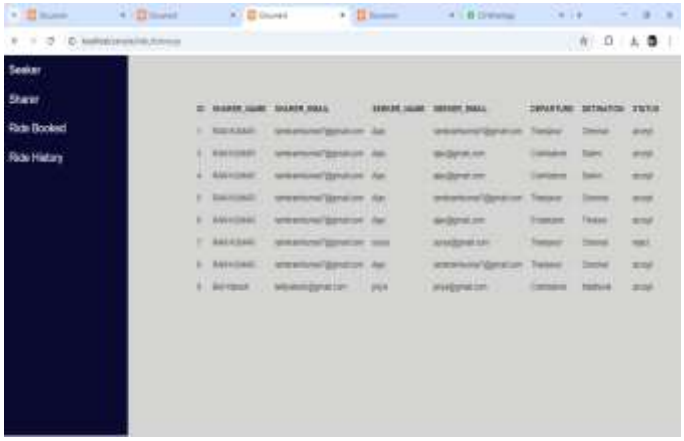
- Ensured secure handling of user data such as passwords and profile information.
- Verified that users cannot access other users' data without proper authorization.
- Checked for common vulnerabilities like SQL injection and session hijacking.

2. Testing Tools Used

- Manual testing using various user accounts.
- Browser developer tools (Chrome DevTools) to test responsiveness and identify frontend issues.
- Database queries checked using MySQL interface for data accuracy.
- Logs and console messages used to detect backend errors.

3. Test Cases – Examples

Test Case ID	Description	Input	Expected Output	Result
TC01	User login with valid credentials	Email + Password	Redirect to dashboard	Pass
TC02	Create new ride	Source, Destination, Time	Ride saved and listed	Pass
TC03	Search rides by location	Source = "Anna Nagar"	List of nearby rides shown	Pass
TC04	Send message to matched partner	Ride match ID	Message sent confirmation	Pass
TC05	Submit rating	Rating = 4 stars	Rating saved in database	Pass



RESULTS

The development and implementation of the **Riding Partner Sharing System** have successfully fulfilled the primary objectives of providing a reliable and user-friendly platform for individuals to find and share rides. The system offers key features such as user registration, ride creation, intelligent ride matching, communication between users, and feedback submission after ride completion.

Key Outcomes:

- Users are able to **register and log in securely** using their credentials.
- Ride givers can **post ride offers** with accurate source, destination, time, and seat availability.
- Passengers can **search and match** rides based on their route and time preferences.
- A **basic ride-matching algorithm** successfully filters and displays suitable rides based on proximity.
- Users can **chat with each other**, confirm rides, and **rate partners** after completion.
- The admin dashboard allows for **system monitoring and user management**, ensuring platform safety and reliability.
- The platform is **mobile-friendly**, making it accessible for users on the go.

The system has been tested thoroughly and is found to be **functionally stable** under normal usage scenarios. User feedback during testing was positive, highlighting the simplicity and usefulness of the platform. This indicates that the system is **ready for real-world deployment** with minor improvements in future updates.

CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion

The **Riding Partner Sharing System** was successfully developed as a practical solution to promote shared commuting, reduce individual travel costs, and lower traffic congestion. The system allows users to create and search for ride offers, get matched with suitable partners, communicate securely, and provide post-ride feedback.

The implementation focused on building a user-friendly interface, a functional backend, and secure data handling. The system has been tested for various use cases and shows satisfactory performance in terms of usability, reliability, and accuracy. It meets the initial objectives of the project and demonstrates potential for real-world application in urban and campus environments.

Overall, the project proves that a ride-sharing platform can be effectively built using current web technologies, providing a sustainable and community-based transport alternative.

Future Enhancements

Although the system is functional, several improvements can be made in future versions:

Mobile Application Development:

Create dedicated Android/iOS apps for better user convenience and on-the-go access.

Advanced Matching Algorithm:

Use AI or ML techniques to improve ride-matching based on user history, traffic data, and real-time GPS tracking.

Live GPS Tracking:

Enable real-time tracking of rides and users for better coordination and safety.

Payment Integration:

Add digital payment gateways (e.g., UPI, PayPal) for ride cost sharing.

Ride Scheduling and Notifications:

Allow users to schedule future rides and receive automatic reminders or notifications.

Enhanced Security Features:

Include user verification (KYC), emergency contact features, and ride history tracking for improved safety.

Carbon Footprint Tracking:

Show users how much CO_2 they've saved by ride-sharing, promoting eco-friendly behavior.

By implementing these enhancements, the system can evolve into a powerful and reliable ride-sharing platform for cities, universities, and large organizations, making transportation more affordable, social, and sustainable.

REFERENCES

1. **Google Maps Platform Documentation**
Google Developers. (n.d.). *Google Maps APIs and Services*.
Retrieved from https://developers.google.com/maps
Used to integrate map-based ride selection and route matching in the application.
2. **W3Schools Online Web Tutorials**
W3Schools. (n.d.). *HTML, CSS, JavaScript, PHP, MySQL Tutorials*.
Retrieved from https://www.w3schools.com
A comprehensive source for learning and referencing frontend and backend development basics.
3. **Mozilla Developer Network (MDN)**
Mozilla. (n.d.). *Web Technology Documentation*.
Retrieved from https://developer.mozilla.org
Referenced for HTML5 semantic elements, JavaScript ES6 features, and responsive web design practices.
4. **GeeksforGeeks**
GeeksforGeeks. (n.d.). *Data Structures, Algorithms, and Programming*.
Retrieved from https://www.geeksforgeeks.org
Used for understanding backend logic, login system implementation, and database connectivity in PHP/Python.
5. **Stack Overflow – Community Support**
Stack Overflow. (n.d.). *Developer Q&A Forum*.
Retrieved from https://stackoverflow.com
Assisted in solving common programming issues and debugging errors.
6. **MySQL Documentation**
Oracle Corporation. (n.d.). *MySQL 8.0 Reference Manual*.
Retrieved from https://dev.mysql.com/doc
Used for database table creation, query writing, and normalization principles.
7. **IEEE Xplore Digital Library**
IEEE. (2021). *A Survey on Dynamic Ride-Sharing: Approaches, Challenges, and Opportunities*.
Retrieved from https://ieeexplore.ieee.org
Provided insights into ride-matching algorithms and safety mechanisms in ride-sharing systems.
8. **Bootstrap Documentation**
Bootstrap Team. (n.d.). *Bootstrap 5 Components and Layouts*.

Retrieved from https://getbootstrap.com
Used to implement responsive design, layout grids, and UI components.

9. **Open Source Projects and GitHub Repositories**
GitHub. (n.d.). *Ride-Sharing System Projects*.
Retrieved from https://github.com
Examined to understand the structure of open-source ride-sharing systems.
10. **Coursera & Udemy Courses**
Coursera & Udemy. (n.d.). *Full Stack Web Development Courses*.
Accessed for understanding MVC design patterns, API usage, and full-stack integration.
11. **ResearchGate – Academic Research**
ResearchGate. (n.d.). *Ride Sharing Algorithms & Urban Mobility Studies*.
Retrieved from https://www.researchgate.net
Used for reference on real-time matching and user experience improvements in ride-sharing.
12. **NPTEL Online Courses**
NPTEL. (n.d.). *Database Management Systems & Software Engineering*.
Retrieved from https://nptel.ac.in
Helped understand ER modeling, DFD, and software development life cycle (SDLC).
13. **OWASP Foundation**
OWASP. (n.d.). *Top 10 Web Application Security Risks*.
Retrieved from https://owasp.org
Referred to implement secure login and data protection mechanisms.
14. **Journal of Transportation Research**
"Ride-Sharing and Carpooling Trends in Urban India" (2022).
Journal of Urban Transportation & Mobility.
Provided context on how carpooling systems work in Indian cities and user behavior.
15. **Google Firebase Documentation (Optional Use)**
Google Firebase. (n.d.). *Authentication and Real-Time Database*.
Retrieved from https://firebase.google.com/docs
Considered as an alternative for real-time ride updates and secure login management.