“ShareMyRide : A Car Pooling Application Using Flutter”

1 Shitole Adtiya,2 Akshit Surkutlawar,3 Suraj Shirke,4 Udaysinha Rasal

Abstract:- This carpooling web application is designed to link drivers who have spare seats with passengers traveling in the same direction. The platform seeks to optimize transportation efficiency by facilitating shared rides, which in turn can help reduce traffic congestion, decrease carbon emissions, and lower travel costs. Users can create accounts to enter personal details, with drivers specifying trip details such as start point, destination, time, and seat availability. Passengers can search for trips based on their preferences and request to book seats. Drivers have the discretion to approve or decline these requests. The app includes a messaging feature to facilitate communication between drivers and passengers, as well as an option to exchange contact details. After each trip, users can rate each other, contributing to a feedback system that builds community trust. Additional features, like advanced search filters, recommendation algorithms, trip notifications, and potential integration with other transport services, further enhance the platform’s functionality.

Keywords: Carpooling, transport efficiency, in-app messaging, recommendation algorithm

Introduction:-

Carpooling is a mode of transportation where individuals travel together, often sharing a vehicle to reach a common or nearby destination. It provides an environmentally conscious and cost-effective approach to commuting, reducing both resource use and roadway congestion. Carpooling aims to maximize vehicle occupancy by allowing individuals traveling similar routes to share rides. This reduces the total number of vehicles on the road, cutting down on fuel usage, carbon emissions, and traffic-related environmental impacts. For commuters, carpooling offers an economical alternative to solo driving, helping to reduce fuel, toll, and parking expenses, while enabling a more relaxing or productive commute. On a larger scale, carpooling alleviates road congestion, particularly during peak travel times, promoting smoother traffic flow and shorter travel durations. Fewer vehicles also lessen the demand for parking in crowded areas.

With technological advancements, carpooling has evolved through mobile and web applications, which streamline the process of matching drivers and passengers based on shared travel preferences, schedules, and routes. These applications offer user-friendly interfaces, secure payment systems, and features like real-time ride tracking, ratings, and reviews to enhance user experience. Carpooling not only provides practical benefits but also fosters social connections and community interaction.

Objective :

The ride-sharing web application is a platform that connects drivers who have extra seats in their vehicles with passengers traveling in the same direction. This system aims to enhance transportation efficiency by promoting shared rides, which can reduce traffic, cut down on carbon emissions, and lower travel expenses. The app requires users to register and provide personal information. Drivers can post trip details, including starting location, destination, time, and seat availability, while passengers can search for rides that match their preferences and request bookings. Drivers may accept or reject these requests based on availability and preferences. The application includes a messaging feature for communication between drivers and passengers, along with an option for contact sharing. Users can provide ratings and feedback after each trip, fostering a trustworthy community. Advanced features such as filtered searches, recommendation algorithms, trip notifications, and integration with other transportation services may also be included.

Keywords: Ride-sharing, transportation efficiency, messaging feature, recommendation algorithms

Motivation

The core motivation of this project is to solve the problem of finding people to share rides with, as it can be challenging to find someone traveling to the same destination at the same time. Current ride-sharing platforms often have limitations, such as restricted coverage areas and language barriers, which hinder accessibility in many regions. This project aims to develop an application that is user-friendly, universally accessible, and seamlessly integrated into daily routines, offering a sustainable and environmentally friendly transport solution.

The application's primary goal is to provide an easy-to-use, generic platform for everyday use, contributing to a sustainable and eco-conscious transportation system. By simplifying the process of finding travel companions, the app can foster a more connected community and promote an active, eco-friendly lifestyle.

Methodology

The development process of this carpooling app follows these key stages:

1. **Requirements Gathering**: Identify both user and business needs, focusing on the features and functionalities essential for a comprehensive application.
2. **Design and Architecture**: Create a high-level design, covering the user interface, data models, and backend services. This stage includes creating wireframes, flowcharts, and user stories.
3. **Development**: Implement the frontend using React.js, the backend with Node.js, and use MongoDB for data storage. This phase involves coding, integrating APIs, and developing test cases to ensure functionality and performance.
4. **Testing**: Perform rigorous testing, including unit, integration, and system tests, to verify the application's performance and resolve any issues.
5. **Deployment**: Launch the application in a live environment, setting up servers, databases, and resources to ensure optimal access and user experience.
6. **Maintenance and Support**: Continuously monitor the application, fixing bugs and releasing updates to meet user requirements and evolving market trends.

Code and implementation

The ride-sharing application is developed using Flutter for the frontend, Spring Boot for the backend, and Firebase as the database. This process involves creating both the frontend and backend components, integrating third-party APIs, and performing comprehensive testing to ensure functionality and performance.

1. **Design the Frontend**: Create a user interface with Flutter, focusing on a responsive and dynamic experience.
2. **Build the Frontend**: Develop the Flutter-based UI components, ensuring cross-platform compatibility.
3. **Set Up the Persistence Layer**: Use Firebase for real-time data management and storage.
4. **Build the API**: Implement the backend logic using Spring Boot, connecting with Firebase for data handling.

Frontend(Flutter)

The application leverages Flutter to build a dynamic and responsive user interface. Flutter’s cross-platform capabilities enable consistent experiences across mobile devices. Real-time updates are crucial for modern ride-sharing applications. Traditional methods like polling and AJAX requests are inefficient due to high server and network resource consumption. Instead, Firebase's real-time database is utilized, enabling seamless data synchronization without frequent requests.

Flutter's reactive programming model enhances real-time data flow, allowing the UI to respond immediately to changes such as ride requests or updates in ride availability. This approach ensures a smooth user experience, making it easier for users to book rides, view ride statuses, and communicate with other users.

Backend(Spring Boot)

Spring Boot, a popular Java-based framework, is chosen for the backend due to its robust features and compatibility with modern microservice architectures. It simplifies the development of server-side applications and integrates well with Firebase for real-time data handling.

The backend is designed using the Model-View-Controller (MVC) architecture:

1. **Models**: Define the structure of the data, including user profiles, ride requests, ride offers, and booking details.
2. **Views**: The frontend components developed in Flutter act as the views, interacting with the backend via API requests.
3. **Controllers**: Manage the business logic and handle requests from the frontend. This includes user authentication, ride searches, ride offerings, and booking management.

Key backend features include:

- **Data Models**: The data models are designed to work seamlessly with Firebase's NoSQL structure, storing user data, ride information, and booking details in a JSON-like format.
- **RESTful API**: The backend API, built with Spring Boot, follows RESTful principles and communicates with the frontend using JSON. API endpoints handle user authentication, ride searching, and booking processes.
- **Security and Authentication**: Spring Security is used for handling user authentication and authorization. Firebase Authentication is integrated for a secure login experience.

Database(Firebase)

Firebase, a cloud-based NoSQL database, is utilized for its real-time data capabilities and ease of integration with both Flutter and Spring Boot. Firebase's document-based storage format allows for flexible data handling, which is essential for scaling the application.

Key benefits of using Firebase include:

- **Real-Time Updates**: Firebase's real-time database provides instantaneous data synchronization, enabling features like live ride status updates.
- **NoSQL Structure**: Firebase uses a JSON-like data format, reducing the need for data transformation and increasing performance.
- **Cross-Platform Support**: Firebase integrates seamlessly with Flutter and Spring Boot, simplifying data management and reducing development complexity.

User Interface

1. **Login/Register**: Users can either log into their existing accounts or register to create a new one, providing essential information to participate in ride-sharing services.
2. **Post Ride**: Users post ride details, such as departure location, destination, and time, to connect with passengers. Drivers can review vehicle details, like model and rental conditions, and check feedback from other users to ensure a reliable experience.
3. **Ride Status**: This feature allows users to view the current ride status. Once the trip is completed, the driver or passenger can mark the ride as finished, officially ending the journey.
4. **Chat Page**: A built-in chat feature facilitates real-time communication between drivers and passengers, allowing them to coordinate trip details or address any immediate concerns.

Merits and Demerits

Advantages:

1. **Cost Savings**: By sharing rides, users can split transportation costs, lowering individual expenses on fuel, tolls, and other travel fees.
2. **Reduced Traffic Congestion**: Ride-sharing helps decrease the number of vehicles on the road, which can reduce travel times, ease congestion, and diminish pollution levels.

3. **Environmental Benefits**: Fewer cars on the road contribute to a reduction in greenhouse gas emissions, supporting a cleaner environment.
4. **Convenience**: The carpooling platform allows users to quickly find or offer rides, making it easier to arrange shared transportation.
5. **Increased Social Interaction**: Carpooling promotes socialization, as passengers can interact and connect with others on their journey.
6. **Real-Time Tracking**: Users can track ride status in real-time, enabling easy ride coordination and offering flexibility for potential changes.

Disadvantages:

1. **Varied Timing Requirements**: Passengers may have different schedules or need to make stops along the way, which can disrupt a smooth, uninterrupted trip.
2. **Reduced Privacy**: Sharing a vehicle requires passengers to give up some level of privacy, which may be uncomfortable for certain users.
3. **Ongoing Maintenance**: Continuous updates and support are necessary to maintain the application's usability and performance, which can require significant resources.

Future Scope

Future advancements in carpooling applications could include enhanced matching algorithms that consider factors beyond basic criteria like location and timing. For example, preferences for conversation, music, or driving styles could be included, potentially improving user satisfaction and accuracy in matches through machine learning techniques.

Integration with smart city infrastructures and public transportation systems could also offer real-time data sharing for route optimization, reduced traffic, and incentivized carpooling. Partnerships with municipalities and transit authorities could enable such a connected ecosystem.

As autonomous vehicle technology develops, carpooling apps might incorporate self-driving cars to provide on-demand ride-sharing, further reducing traffic and enhancing transportation access for users unable to drive. Additionally, future carpooling apps could offer environmental and social impact insights, showing users their contributions to sustainability, such as CO2 emissions saved or distance reduced through shared rides.

There is also potential to merge carpooling with other transport modes, like public transit, cycling, or ride-hailing, to provide seamless multi-modal travel options. These innovations would give users greater flexibility in trip planning. Enhanced safety features, including more thorough background checks, identity verification, real-time tracking, and emergency assistance, could improve security and foster trust among users, encouraging a safer carpooling environment.

Conclusion

In summary, this carpooling web application provides a practical and eco-friendly solution for users to share rides, reducing individual carbon footprints and optimizing resource use. By connecting drivers and passengers heading in similar directions, the platform encourages efficient travel and community building.

Throughout its development, the application utilized technologies such as Flutter for the frontend and Javascript for the backend, alongside Firebase for data management, creating a reliable and user-friendly experience. This carpooling platform addresses common challenges in transportation, offering an accessible space for individuals to connect and share journeys, fostering a more sustainable approach to commuting.

References

1]Xingyuan Li ,Yan Chang,Huijie Peng,Xiaomin Dai: "Can Ridesharing Improve the Reserve Capacity ofTransportation Network?" IEEE Access(2024).

2]Agielia luydia , Manush Aditya , Mithun K : "Car Pooling System Using flutter". IJNRD.ORG(2024).

3]David Zar , Noam Hazon , Amos Azaria: “Information Disclosure for Increasing User Satisfaction From a Sharing Ride “. IEEE Access(2023).

4]Wenbo Xhang ,Satish V. Ukkusuri: “Share-aCab: Scalable Clustering Taxi Group Ride Stand From Huge Geolocation Data”. IEEE Access(2021).

5]Ajinkya Ghorpade , Apurva Joshi , sumit Mali , Harsh Agarwal : “Pool : A Peer-to-Peer Ride Sharing App” . IJEAST(2021).

6]Hajra Qadir , Osman Khalid , Raheem Nawaz , Atta Ur Rehman Khan: “An Optimal Ride Sharing Recommendation Framework for Carpooling services” . IEEE Access(2018).