WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

VAZHIPOKAN

Dr Esther T, Mukesh Kumar T, Navinath M, Parthasarathy E

Sri Shakthi Institute of Engineering and Technology, Coimbatore, 641062, India.

A B S T R A C T

Vazhipokan is a smart ride-sharing mobile application designed to connect individuals who are traveling in the same direction. The app facilitates seamless communication between a person seeking a lift and a person willing to offer one. The user seeking a ride can register their destination and request assistance through the app. Simultaneously, users who are heading towards the same or nearby destination can view the request and choose to accept it. This innovative platform promotes community-driven travel, reduces fuel consumption, and supports sustainable transportation. By focusing on short-distance, need-based lifts, *Vazhipokan* not only enhances travel convenience but also encourages mutual help among daily commuters.

Keywords: Ride-sharing, Lift request, Destination matching, Carpooling, Mobile application, Eco-friendly travel, Real-time ride matching.

1. Introduction

In the modern world, transportation is a daily necessity, but it often comes with problems like traffic congestion, rising fuel prices, and environmental concerns. Many individuals travel alone in personal vehicles, leading to underutilization of space and increased pollution. Addressing these issues requires innovative, community-based solutions that promote shared travel and reduce the overall number of vehicles on the road.

Vazhipokan is a mobile application designed to connect people traveling in the same direction. The app allows users to register their destination and request a lift, while others going that way can view and accept the request. This real-time ride-matching system fosters a culture of helping one another, reduces travel costs, and supports eco-friendly commuting. By combining convenience with sustainability, *Vazhipokan* encourages smarter, shared mobility for a better future.

2. Literature Review

Several existing ride-sharing platforms like uber pool have proven the effectiveness of shared travel in reducing fuel consumption and traffic congestion. These systems use real-time matching and GPS tracking to connect riders with similar destinations. However, they are mostly commercial and focus on long-distance or urban travel.

Recent studies highlight the need for localized, community-based ride-sharing solutions. Projects like smart ride and quick lift emphasize peer-to-peer lift sharing but lack simplicity for everyday users in semi-urban or rural areas.

3. Methodology

The development of *Vazhipokan* started with gathering user requirements through surveys to understand common transportation problems. Based on these findings, the app was designed with two main features: a user module for registration and requesting rides, and a driver module for accepting these requests. The app was developed using technologies like Flutter/Android Studio and Firebase for real-time data management. After the development phase, the app underwent testing with users to ensure it worked smoothly, including features like ride requests and accepting offers.

3.1 System Overview

The *Vazhipokan* system follows a modular approach, consisting of the following major components:

1. **User Registration and Authentication** – Secure user sign-up and login process to ensure that only verified users can access the platform.
2. **Ride Request** – Users can request a lift by entering their destination and travel time.
3. **Ride Matching** – The system identifies nearby users who are traveling in the same direction and can potentially offer a ride.

4. **Ride Acceptance/Decline** – Users offering rides can accept or decline ride requests based on availability and route compatibility.
5. **Real-Time Notifications** – Notifications are sent to both riders and drivers when a request is made or accepted.
6. **Feedback and Rating System** – After completing a ride, users can rate their experience and provide feedback.
7. **Backend Processing (Firebase API)** – Integration of real-time data handling, user authentication, and notification services using Firebase.

3.2 Data Collection and Preprocessing

The data used by the system consists of:

- **User Information:** Collected during registration, including name, contact details, and location.
- **Ride Requests:** Users enter their desired destination, travel time, and current location.
- **Real-Time Location Tracking:** Location data is constantly updated to ensure ride requests and offers are matched accurately.

3.3 Ride Matching Algorithm

The core feature of the system is its ability to match ride requests with available drivers:

- **Destination Matching:** The system identifies users who are traveling towards the same or a similar destination.
- **Proximity Matching:** The system uses real-time GPS data to identify users in close proximity who may be able to offer rides.
- **Availability Matching:** Only users who are actively traveling at the requested time are shown as available drivers.

3.4 User Interface and Experience

The *Vazhipokan* app is designed for simplicity and ease of use:

- **Home Screen:** Displays a clear input form for users to enter their destination and request a lift.
- **Ride Offer Interface:** Displays available ride requests with options to accept or decline based on destination and availability.
- **Notifications:** Real-time alerts for both riders and drivers regarding ride requests, acceptances, and cancellations.

3.5 Backend Integration (Firebase)

The backend of *Vazhipokan* is developed using Firebase, providing essential services:

- **User Authentication:** Firebase Authentication is used to securely manage user sign-ups and logins.
- **Real-Time Database:** Firebase Realtime Database handles dynamic data exchange for ride requests and location updates.
- **Push Notifications:** Firebase Cloud Messaging is used for sending real-time notifications to users about ride updates.

4.Implementation Details

The *Vazhipokan* ride-sharing system was implemented using Python, integrating key technologies and frameworks to ensure a smooth user experience and efficient backend processing. The frontend interface was developed using Flutter (for cross-platform compatibility), while the backend was built using Firebase for real-time data handling and Flask for integrating additional functionalities. This section outlines the tools, libraries, frameworks, and environment used in the project.

4.1 Development Environment

The development environment was set up with the following specifications:

- Operating System: Windows 10 / Ubuntu 20.04
- Programming Language: Python 3.8 for backend logic and Firebase integration
- Mobile Development Framework: Flutter (for cross-platform development)
- Backend Framework: Firebase (for real-time database management and notifications)

- IDE: Android Studio or Visual Studio Code (for Flutter and Python development)

4.2 Backend Integration (Firebase)

The backend handles user authentication, ride requests, and the real-time database for ride matching. Firebase Cloud Functions were used to implement RESTful APIs, including:

- /register: User registration and authentication
- /login: User login and session management
- Accepts ride requests with details like user destination and time
- Provides the status of the ride, including whether a driver has accepted the request or not
- /notifications: Push notifications to alert users about updates (ride status, nearby drivers, etc.)

4.3 Frontend Design

The frontend was developed using Flutter for both Android and iOS platforms, providing a seamless and responsive user experience. The interface allows users to:

- Register and log in securely through Firebase Authentication
- Request a ride by entering the destination and time
- View available ride requests with nearby users and offers to give rides
- Accept or reject ride requests based on proximity and destination
- View ride status and notifications in real-time through a notification system

4.4 Output and Visualization

Upon successful ride matching, users are presented with key details regarding their ride request and the available drivers:

- Ride Request Information: Destination, ride time, and the number of potential drivers nearby
- Matching Results: A list of drivers heading in the same direction with their estimated arrival times
- Ride Status: Real-time updates on whether the ride request was accepted or pending
- Push Notifications: Alerts for ride confirmations, status updates, and cancellations

5.Result and Discussion

The *Vazhipokan* ride-sharing system was successfully developed and tested with various users. The system performed as expected in helping users find ride-sharing opportunities and offer rides to others traveling in the same direction. Here, we discuss the key results, user feedback, and areas for potential improvement.

6. Conclusion

Overall, *Vazhipokan* has the potential to transform how people commute, offering a simple, cost-effective, and eco-friendly solution to everyday travel. With ongoing improvements and real-world implementation, this system could become an essential tool for efficient ride-sharing and smarter city transportation.

Acknowledgements

Finally, I would like to acknowledge the resources provided by Firebase, Flutter, and other open-source technologies that made this project feasible and helped accelerate the development process.

REFERENCES

A Ride-Sharing System for Matching Riders and Drivers," 2025.

Firebase Documentation: Firebase, "Firebase Overview," *https://firebase.google.com/docs*, Accessed: May 2025.

Flutter Documentation: Flutter, "Flutter: Build natively compiled applications for mobile, web, and desktop from a single codebase," *https://flutter.dev/docs*, Accessed: May 2025.

Ultralytics YOLOv5: Ultralytics, "YOLOv5: State-of-the-Art Object Detection," *https://github.com/ultralytics/yolov5*, Accessed: May 2025.

Python Software Foundation: Python Software Foundation, "Python Programming Language," *https://www.python.org/doc/*, Accessed: May 2025.

Flask Documentation: Flask, "Flask: A Microframework for Python," *https://flask.palletsprojects.com/en/2.0.x/*, Accessed: May 2025.

Deep Learning with PyTorch: PyTorch, "PyTorch: An open-source deep learning framework," *https://pytorch.org/tutorials/*, Accessed: May 2025.