Advanced International Journal for Research (AIJFR)
E-ISSN: 3048-7641 ● Website: www.aijfr.com ● Email: editor@aijfr.com

Dual-Interface Ride-Sharing Platform for Real-time Driver-Passenger Matching.

Aryan Gupta[1], Dr. Namrata Khade[2], Ashmit Yadav[3], Anand Borkar[4], Kaustubh Lodha[5], Rupesh Rahangdale[6].

[1]Research Scholar, Computer Science & Engineering, Priyadarshini College of Engineering.
[2]Professor, Computer Science & Engineering, Priyadarshini College of Engineering.
[3,4,5,6]Research Scholar, Computer Science & Engineering, Priyadarshini College of Engineering.

Abstract

With the rapid growth of the global population, the number of vehicles on urban roads has also increased rapidly. This surge has led some severe traffic congestion, resulting in substantial delays for travellers traveling to important destinations. A major contributor to this inefficiency is that many vehicles are occupied by a single driver, leaving multiple seats vacant, while numerous people are forced to walk or seek alternative transportation options.

Therefore the primary reasons for this problem include the high fares charged by conventional ride-hailing services, which require passengers to book an entire vehicle. Even when vacant seats are available, they cannot be shared because the ride is reserved exclusively for one passenger.

This problem is effectively addressed by our project, Ride Sharing. The platform allows drivers who are already traveling to their own destinations to offer their vacant seats to other passengers heading in the same or similar direction. By enabling real-time seat sharing within existing journeys, Ride Sharing eliminates the need to deploy additional vehicles for single passenger trips. Instead of offering an entire vehicle for hire, the driver acts as a partner traveller, simply sharing available seats along their planned route.

Keywords: Ride sharing, Cost and time efficient travel, On demand transportation.

1. Introduction

In an era of rapid population growth and escalating vehicle ownership, urban traffic congestion has intensified, resulting in significant delays for travellers and utilization of road space. Conventional ride-hailing services exacerbate this issue by requiring passengers to book entire vehicles at premium fares, leaving vacant seats unutilized even when multiple individuals share mixed destinations. This project introduces Ride Sharing, a smart ride pooling mobile application that transforms everyday private vehicles into dynamic shared vehicle mobility solutions.

Unlike traditional driver-as-service models, Ride Sharing enables genuine travellers who are already driving to their own destinations to offer their vacant seats to fellow passengers traveling along the same or overlapping routes. The platform intelligently matches drivers and passengers in real time based on origin, destination, route proximity, and preferred time windows, ensuring that no additional vehicle is

deployed for single-occupant trips. By the existing flow of traffic, Ride Sharing maximizes seat occupancy, reduces the total number of vehicles on the road, and significantly cuts travel costs for both drivers and passengers through fair, per-seat fare sharing.

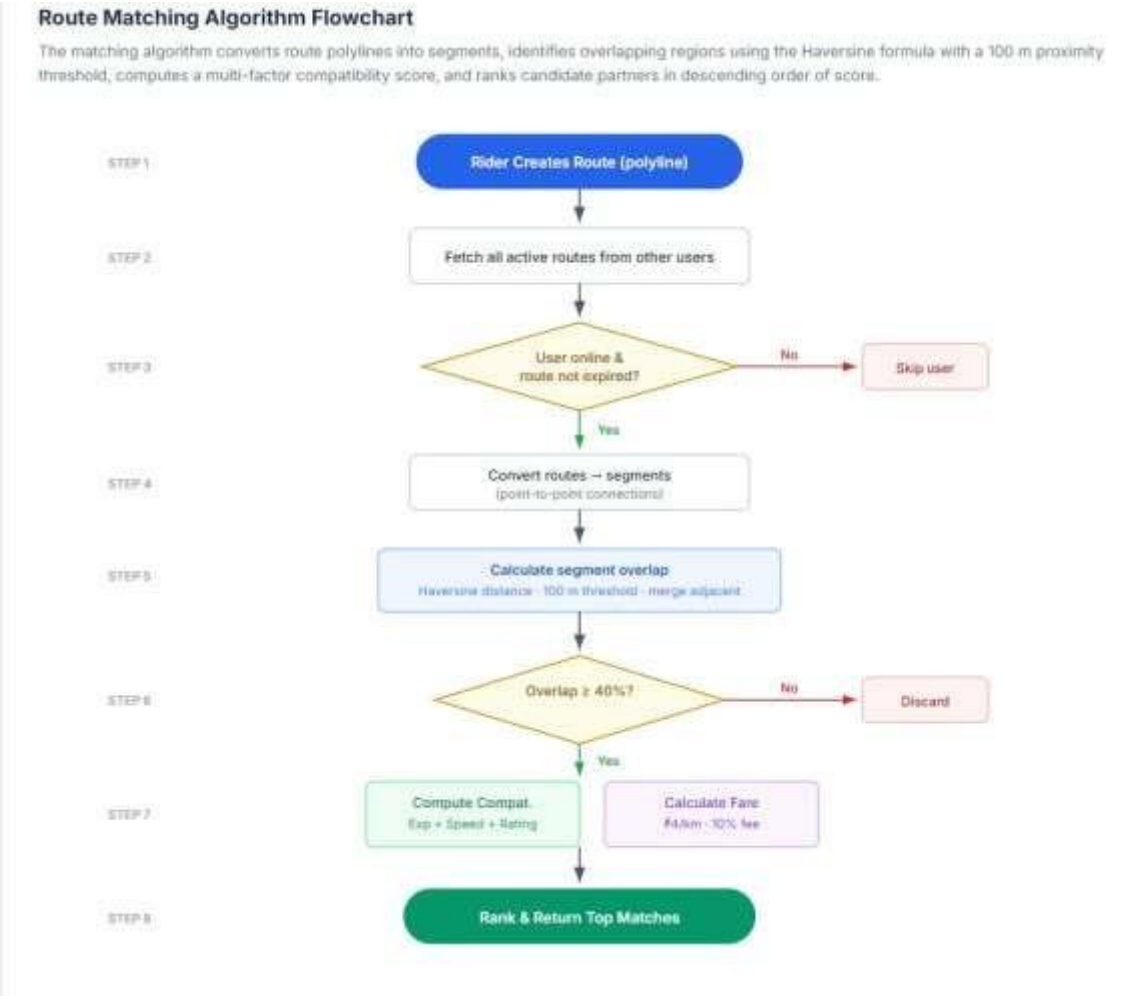
Key features include seamless route-based matching, secure in-app payments, real-time tracking, safety verification, and an intuitive interface that allows drivers to specify available seats and passengers to join ongoing journeys without disturbing the driver's original plan. This innovative peer-to-peer ride-pooling mode not only solves traffic congestion and lowers transportation expenses but also reduces carbon emissions, and increases a community-driven culture of efficient resource sharing. Ride Sharing represents a practical, scalable solution to the growing challenges of modern city commuting.

2. Abbreviations and Acronyms.

Sr. No.	Abbreviation / Acronym	Full Form
1.	API	Application Programming Interface
2.	GPS	Global Positioning System
3.	OTP	One-Time Password
4.	P2P	Peer-to-Peer
5.	Ride Sharing	Ride Sharing Platform for real-time matching
6.	UI	User Interface
7.	UX	User Experience
8.	JSON	JavaScript Object Notation
9.	REST	Representational State Transfer
10.	HTTPS	Hypertext Transfer Protocol Secure
11.	DBMS	Database Management System
12.	GMAPS	Google Maps API

3. FIGURES AND TABLES

Figure 1: FLOWCHART

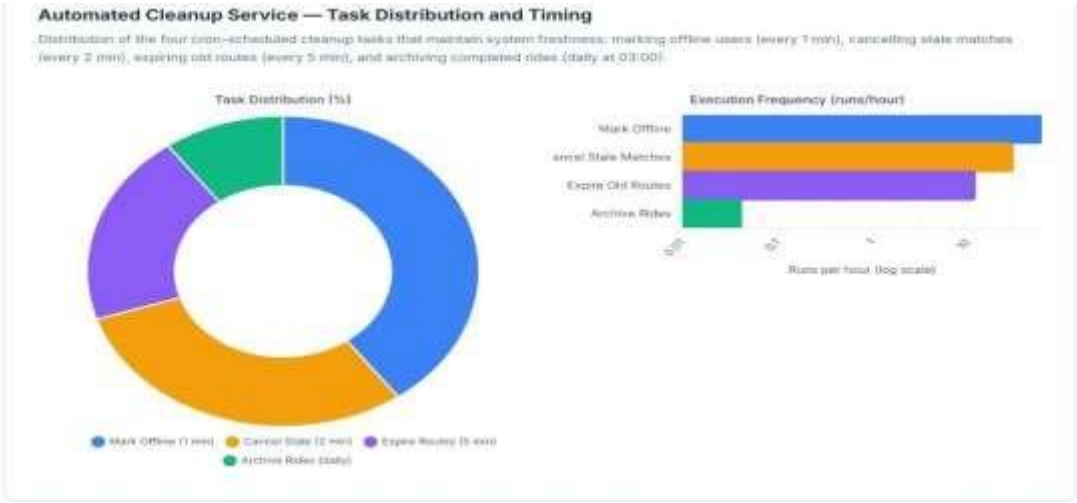


The flowchart shows how the route matching algorithm works in a ride-pooling system. First, a rider creates a route which is stored as polyline containing multiple GPS coordinates of the path. After that, the system fetches active routes of other users from the database. Then checks whether the user is online and whether the route is still valid or not. If the user is offline or the route has expired, that user is skipped. If the user is active, the system converts the routes into smaller segments, which are basically point-to-point connections between coordinates.

Next, algorithm calculates how much the routes overlap. For it uses the Haversine formula to measure the distance between route segments with a threshold of about 100 meters. If the overlap between two routes are less than 40%, the match is rejected. If it overlaps is more than 40%, the system continues the process by calculating a compatibility score based on factors like experience, speed, and user rating. It also calculates the estimated fare, for example ₹4 per kilometer with a 10% platform fee. Finally, the system ranks every suitable matches and returns the top matching riders for ride pooling.

1. Rider Creates Route (Polyline)
   The rider enters the starting point and destination. The system saves path as a polyline, which is a set of GPS coordinates representing the path.
2. Fetch Active Routes
   The system collects all currently active path's from other users in the database to compare with new route.
3. Check User Status
   The system checks if the user is online and if the route is still valid. If the user is offline or the route has expired, user is skipped.
4. Convert Routes into Segments
   Each route is divided into smaller segments so that the system can compare routes more easily.
5. Calculate Segment Overlap
   The system measures the distance between route segments using the Haversine formula with a threshold of about 100 meters to see if the routes are close to each other.
6. Check Overlap Percentage
   If the route overlap's less than 40%, the match is rejected. If the overlap is more than 40%, the process continues.
7. Calculate Compatibility and Fare
   The system calculates the compatibility score based on factors like speed and user rating. It also estimates the fare (for example ₹4 per km with a 10% fee).
8. Rank and Return Matches
   Finally, the system rank the best matches based on the scores and returns to top suitable riders for ride pooling.

Figure 2: Task Distribution



This shows how different automated cleanup tasks are distributed and how often it run's in the system to keep ride-pooling platform updated. The Mark Offline task runs every 1 minute and has the highest execution frequency because it continuously checks and marks inactive users. The Cancel Stale Matches task runs every 2 minutes to remove ride matches are no longer valid. The Expire Old Routes

task runs every 5 minutes to delete routes that are outdated or no longer needed. Finally, Archive Completed Rides task runs once daily at 03:00 storing completed ride data for records.

4. Study Methodology

The methodology of the study is divided into 2 main parts. The first part explains the system architecture, which describes how different modules of the ride-pooling system work together. The second part explains the technology stack used to build and implement the system.

4.1 System Architecture

The system architecture is composed of several modules so manage user interaction, route handling, ride matching, and safety.

4.1.1 User Registration Module

This module allows new users to create an account in system. During the registration, user selects their role either as a driver (vehicle owner) or a passenger (traveller). The user provides basic details such as name, contact information, and login credentials. After successful registration, user can access the platform according to their selected role.

4.1.2 Route and Schedule Entry

This section is mainly used by drivers. Drivers enter their starting location, destination, and planned departure time. The system stores this route information in the database so that it can later be compared with passenger requests. This helps the system identify passengers who are travelling on the same or similar routes.

4.1.3 Ride Matching Engine

This module is responsible for finding suitable ride partners. The system compares passenger requests with available driver routes and checks similarity between routes and travel times. If the routes overlap or are close enough, the system suggests a match so that both users can share’s ride.

4.1.4 Ride Confirmation

Once a suitable match is found, the system sends notifications to both the driver and the passenger. They can review the ride details and confirm the ride. If no match is available at that time, the passenger is informed about the next closest ride option.

4.1.5 Feedback and Safety Module

After the ride is completed, both drivers and passengers can provide ratings and feedback about their experience. This helps maintain safety and trust within system and improves the reliability of users on the platform.

4.2 Technology Stack

The following technologies are used to develop the system:

- Frontend: Built using React and JavaScript create an interactive user interface.
- Backend: Implemented using Node.js to handle server-side operations and APIs.
- Database: Firebase or MongoDB is used to store user data, routes, and ride information.

- Maps and Navigation: Google Maps Platform is used for location services, route detection, and distance calculation.
- Authentication: Firebase Authentication is used to manage secure user login through email, phone number, or Google sign-in.

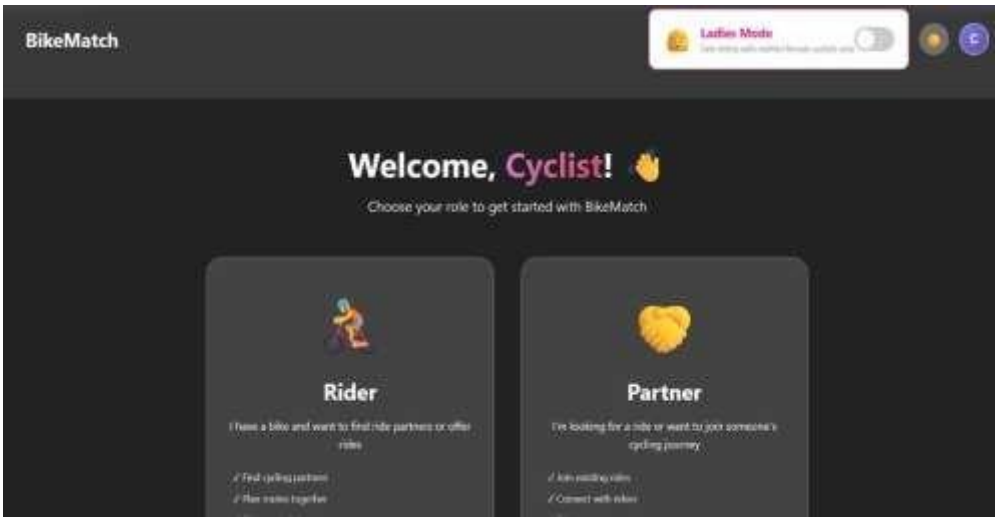
5. Appendix

Figure 1: Login Page

The figure shows the login page of the EKLiFT ride pooling application. It allows existing users to sign in to their account. The screen contains two input fields where users enter their email address and password. After entering the detail's the user can click's on Sign.

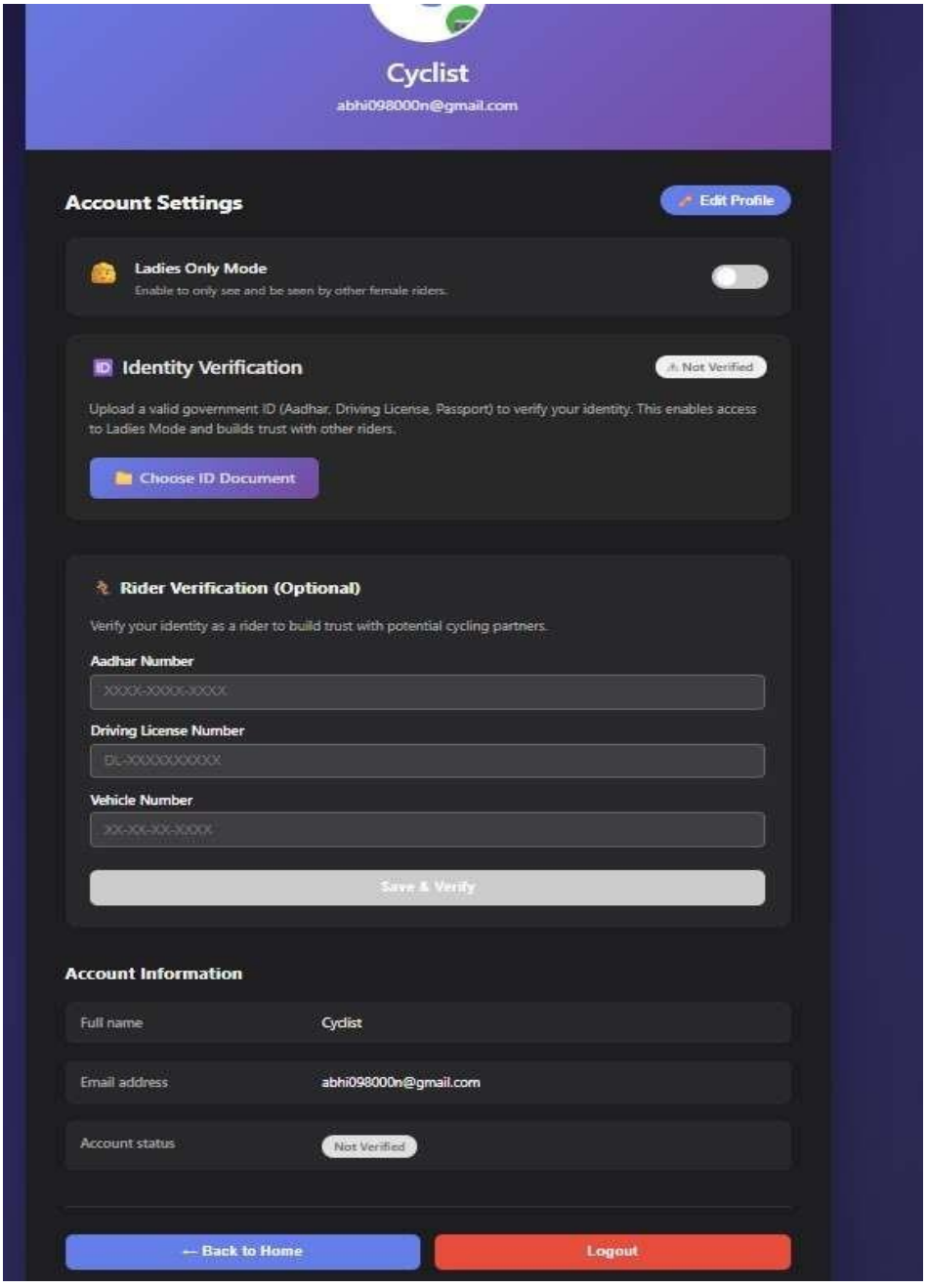
In button to access the application. There is also a Forgot Password option for users who need to reset their password and in Additionally, the system provides a Sign in with Google option using Google Sign-In for quick login. At the bottom, a Sign-Up link is available for new users who want to create an account interface uses a dark theme with green highlights for a clean and simple design.

Figure 2: Home page



The figure show's role selection screen of the Bike-Match application. After logging in, the user is welcomed and asked to choose their role in the system. Two options are available: Rider and Partner. The Rider option is for users who have a bike and want to offer rides or find cycling partners. The Partner option is for users to join someone else's ride. The screen also includes a Ladies Mode option for safer rides with verified female cyclists. This page helps system understand the user's role before starting the ride-matching process.

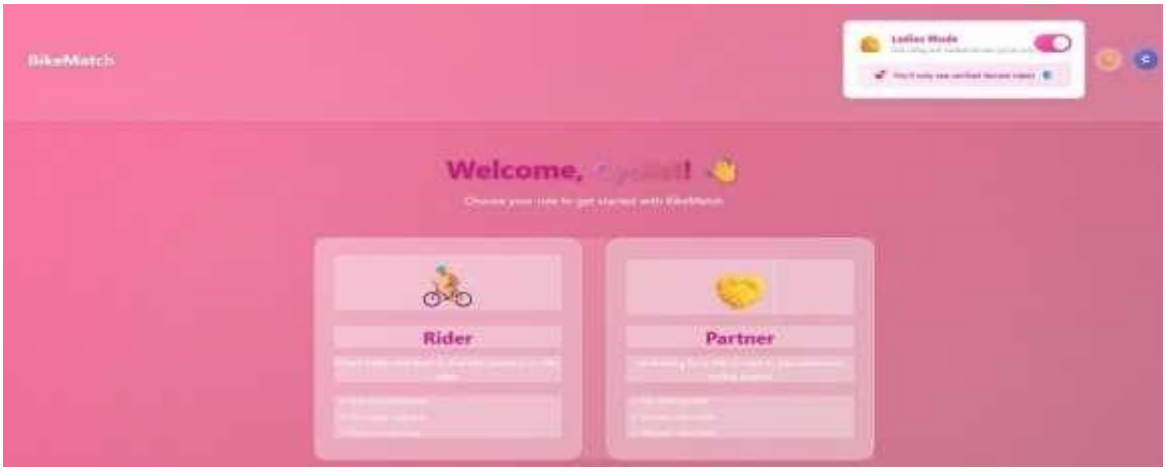
Figure 3: User page



The page shows the account settings page of the Bike Match application. This page allows user to manage their profile and verification. At the top, user profile information such as name and email is displayed,

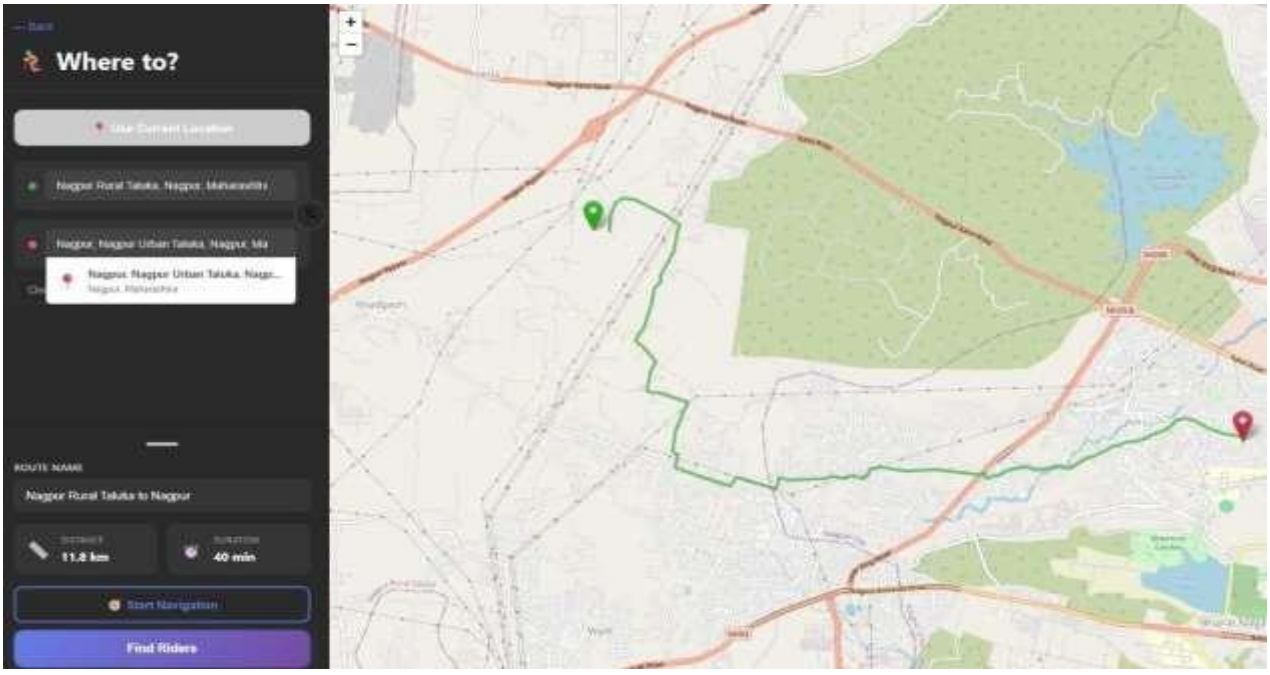
along with an Edit Profile option. The page includes a Ladies Only Mode toggle that allows female users to ride only with verified female riders for safety reason and purpose. There is also an Identity Verification section where users can upload a government ID like Aadhaar, driving license, or passport to verify their identity. Additionally, the Rider Verification section allows users to enter details such as Aadhaar number, driving license number, and vehicle number to build trust with other riders. At the bottom, the page shows account information and provides options to return to the home page or log out of the system.

Figure 4: Ladies Mode



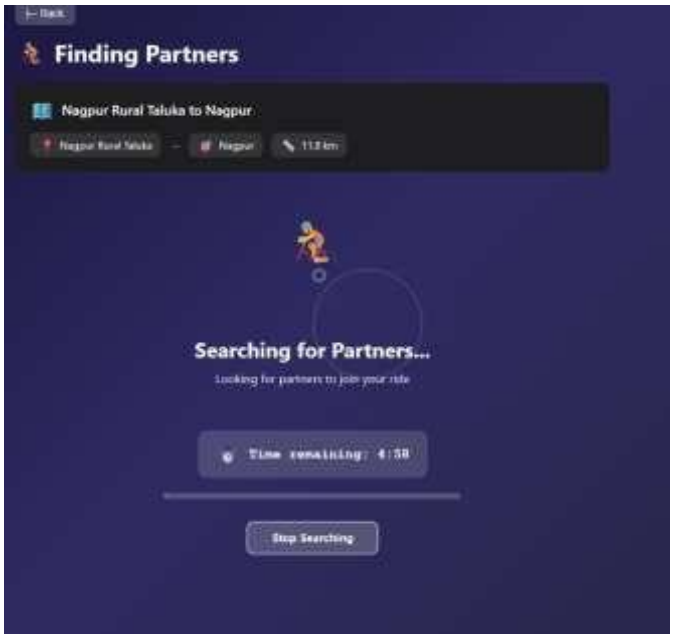
This page is the welcome page of our Bike-Match app where user selects their role to get started. There are two options: Rider, for users who have bike and can offer rides, and Partner, for users want to join someone's ride. It also shows basic features like planning routes and connecting with others. On the top right, there is a Ladies Mode option which is safety feature that allows female users to match only with female riders.

Figure 5: Navigation



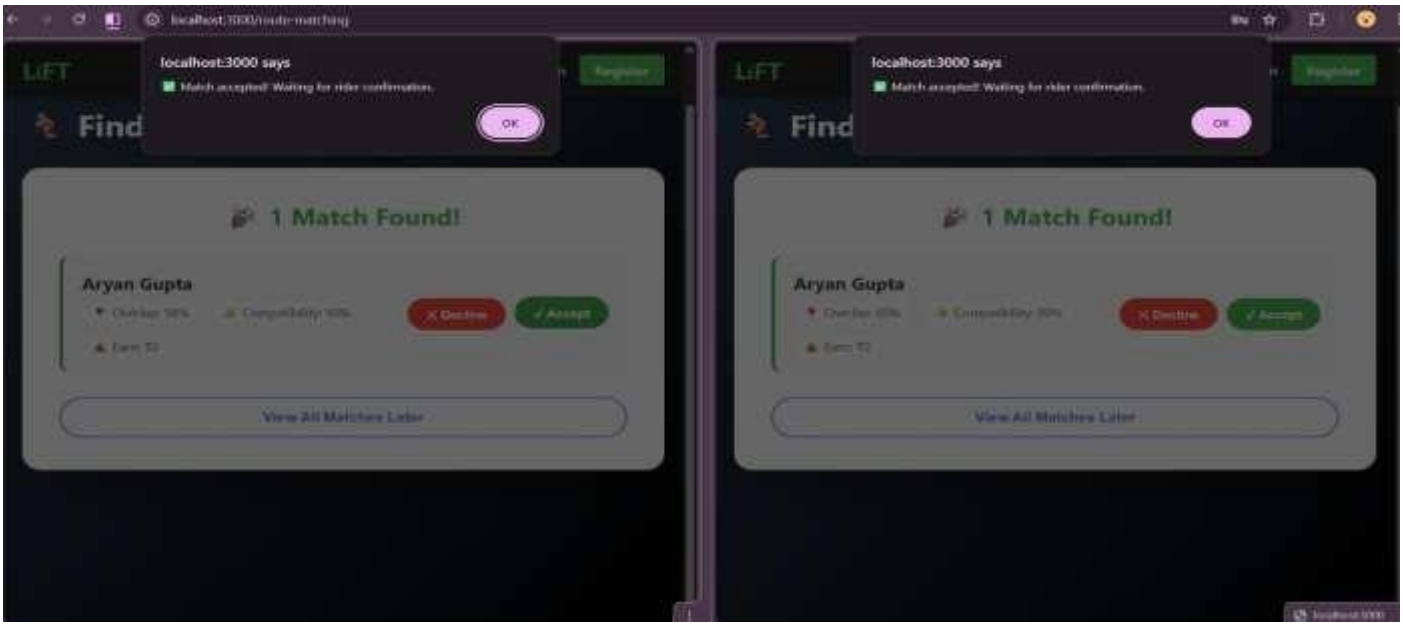
This page is used for selecting the route in our Bike-Match app. The user can either use their current location or manually enter source and destination. The map on the right side shows the selected route clearly with a path between the two points. It also displays details like distance and estimated time. At the bottom, the user can start navigation or click on “Find Riders” to search for people traveling on same route.

Figure 6: Route matching



This screen shows the partner searching process our Bike-Match app. After selecting path, the system starts finding suitable riders or partners going in the same direction. It displays route details at the top and shows a loading animation while searching. There is also timer indicating how long the search will continue, and user can stop the search anytime using the “Stop Searching” button.

Figure 7: Match found



The screen shows when match is found in our Bike-Match app. It displays the matched user's details along with route overlap and compatibility percentage. The user can either accept or decline the match. When both users accept, a confirmation message is shown indicating that the match is successful and waiting for final rider confirmation.

6. Acknowledgement

I would like to express my sincere gratitude to our project guide, **Dr. Namrata Khade,** for their valuable guidance, support, and encouragement throughout the development of this project. Their insights and suggestions helped us improve our work and successfully complete the project.

I also thank our faculty members and institution for providing us with the necessary resources and a good learning environment to carry out this project.

We are grateful to our friends and classmates for their support, ideas, and cooperation during the project development.

Lastly, I would like to thank all the online resources and tools that helped us understand concepts and implement this project effectively.

References

1. BENJAMIN KOMMEY, BENJAMIN ASANE ASANTE and ELVIS TAMAKLOE, "A Ride Sharing System for University Community", IJITIS, vol. 5, April 2022, pp. 907-924, doi:10.15157/IJITIS.2022.5.2.907-924.
2. ANDY HAM, "Dial-a-Ride Problem with Meeting Point Feature Known-as ExpressPool," IEEE, June 2021, doi:10.1109/ACCESS.2021.3089275.
3. LAMBROS K. MITROPOULOS, ANNIE KORTSARI and GEORGIA AYFANTOPOLOU, "A systematic literature review of ride-sharing platforms, user factors and barriers", European Transport Research Review, Dec. 2021, doi:10.1186/s1254402100522-1.
4. KAH PHOOI SENG, LI-MINN ANG, ERICMOORE NGHARAMIKE and ENO PETER, "Ridesharing and Crowdsourcing for Smart Cities: Technologies, Paradigms and Use Cases," IEEE, Feb 2023, doi:10.1109/ACCESS.2023.3243264.
5. JISHNU V JOSHY, NANDHU PRASAD, SIDHARTH PRAVEEN, NANDHU KANAKARAJAN and NEETHU MARIA JOHN, "Ride Sharing," IJERT, 2023, ISSN: 22780181.