International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 04 April-2026 | Impact Factor: 3.5

Let's Go – Driver Booking App

Yashashwini K C
Nagarjuna College of
Engineering and Technology
Beedaganahalli, Venkatagiri Kote, Devanahalli, Bengaluru, Karnataka 562110
yashashwini.kc.acharya@gmail.com
Vyshnavi S
Nagarjuna College of
Engineering and Technology
Beedaganahalli, Venkatagiri Kote, Devanahalli, Bengaluru, Karnataka562110
vyshnavireddys999@gmail.com
Tejas R C
Nagarjuna College of
Engineering and Technology Beedaganahalli, Venkatagiri Kote, Devanahalli, Bengaluru, Karnataka 562110
tejasnaik996@gmail.com

Prof. Raghavendra B
Nagarjuna College of
Engineering and Technology Beedaganahalli, Venkatagiri Kote,
Devanahalli, Bengaluru, Karnataka 562110 raghavendra.b@ncetmail.com

How to Cite this Article:

C, Y. K., S, V. & C, T. R. (2026). Let's Go – Driver Booking App. International Journal of Creative and Open Research in Engineering and Management, 02(04). https://doi.org/10.55041/ijcope.v2i4.576

https://doi.org/10.55041/ijcope.v2i4.576

Abstract - All traditional ride-booking platforms have been encountering some challenges with regards to inefficiency associated with ride booking and tracking. The Driver Booking App presents an efficient ride-sharing solution with a focus on emphasizing salient aspects of ride-sharing services, including user authentication, location tracking, ride booking, and tracking.

It is created with Kotlin as its frontend and Node.js/Express.js with MongoDB on the backend side for efficient data handling.

Its architectural designs provide it with an excellent user interface that enhances ease of convenience for ride booking. It presents a technological remedy for the current transport problem.

Keywords- Ride-hailing, Kotlin, Node.js, MongoDB, Real-time Tracking.

I. INTRODUCTION

The traditional methods for taxi and driver booking, such as contacting them manually or through physical taxi stands, have been seen to be inefficient and inadequate. The core issue with traditional methods is that they do not allow for real-time communication and transparency. As a result, issues such as waiting times, unavailability of drivers, and approximate accuracy in calculation of estimated fares stand prominent. It is an imperative need for today's modern technological world to have a quicker, smarter, and more convenient solution for effectively tackling daily commutes.

To deal with these challenges, Driver Booking App brings an automated and technology-based solution for ride-sharing operations. The app is designed with an objective to provide a smooth service not only for customers but also for drivers. The app contains facilities for customer registration and login, location tracking, and online booking functionality. Open Street Map works as a supportive tool for proper path tracking and mapping. It facilitates tracking and calculation of arrival times for drivers' movement.

The app will be developed with the Kotlin programming language and will be created using Android Studio. It will be implemented with a modern and dynamic UI. It will be created with a backend using Node.js and Express.js and will have a MongoDB database. The app will be scalable and will be able to handle multiple users and bookings without stuttering.

By employing secure APIs and efficient communication between the client and server, it ensures reliability and safety of data throughout the booking process. The system will also be easy to extend for other functionalities, such as cost estimation, rating, and payment functionality.

The Driver Booking Application not only increases user convenience but also establishes a platform for drivers that will enable them to manage bookings effectively. The focus on speed, accuracy, and user convenience makes the proposed system a cost-effective and efficient solution to existing ride-sharing services. As a whole, the project aims at revolutionizing the digital transport sector using the power of real-time technology**RELATED WORKS**

During the previous decade, various research works were conducted with an objective of improving efficiency and accuracy within ride hailing and ridesharing services. Early projects were mainly aimed at optimizing routes and managing fleets using mathematical and algorithm-based modeling. For example, [17] **Justin Miller and Jonathan P. How (2017)** designed an estimation platform and a chance-constrained fleet control strategy for ride hailing.

In [14] **2020**, **Josh Angelo Escalona et al.** suggested a ride-sharing system that uses an expansive search-based algorithm with a focus on improving user experience and overall productivity through proper allocation and organization of drivers and riders.

Later, [10] **Tingting Kang, Lei Zhang, and Yan Huang (2021)** examined the integration of digital wallets with online transport networks, focusing on cashless payment services and efficiency enhancements, but encountering problems such as technical problems and fraud.

Further progress was made by [9] **Haining Yu et al. (2021)**, who introduced **PGRide**, a privacy-preserving group ride-sharing model combining user data safety and match accuracy.

Similarly, [7] **Haining Yu, Xiaohua Jia, and Hongli Zhang (2022)** have created a ride-matching model that preserves user privacy with an exact road distance, thus enhancing ride efficiency and scalability.

In [4] **2023**, **Haining Yu et al.** proposed an efficient ride-sharing model leveraging **machine learning** based on rider reviews, improving reliability and reducing traffic congestion.

More recently, [5] **Wang Peng and Lili Du (2023)** introduced a coordinated real-time route choice mechanism that enhances ridesharing efficiency and reduces congestion through dynamic routing.

From the reviewed literature, it is evident that most existing systems focus on optimization, privacy, and scalability. However, challenges such as complexity, high data requirements, and lack of real-time adaptability remain. The Driver Booking App is set to provide solutions for these prevailing gaps through a real-time, scalable, user-friendly ride-booking app that integrates live tracking with efficient driver allocation and secure communication between users and drivers.

II. MATERIAL AND METHODS

A. Andriod App

The Driver Booking App is created for the Android operating system using Android Studio with Kotlin as the programming language. Android operating system is an open-source operating system commonly used for developing mobile app interfaces due to its flexible nature.

The use of Kotlin Programming makes it possible for the app's code to be short and safe while being more modern and with better performance compared to an app created using Java. The app also uses Google Maps, thus offering precise location tracking and estimation of distance. It is even able to get the coordinates of both users and drivers.

B. The Proposed System

The proposed Driver Booking App is an efficient, automated book-a-ride mobile application.The system consists of two interfaces: a user interface and a driver interface. The user can register and log in to the user interface, where he can book the ride by selecting the pickup and drop locations. The application showcases the nearby available drivers with the help of live locations and confirms the booking immediately. The driver interface allows the driver to log in and see the incoming ride requests, and it updates his status (available/busy).

Once a user confirms a ride, the backend server—made with Node.js and Express.js—manages the request by assigning the nearest available driver. The system provides secure and fast communication between the app and the server using RESTful APIs. All the data regarding rides, users, and drivers, along with trip histories, are stored in MongoDB, a NoSQL database famous for its scalability and high performance.

Its design balances usability with responsiveness to ensure smooth interaction and data flow between the client and the server. By combining modern back-end technologies with an Android-based front-end, correctness in ride tracking, reduction in waiting time, and enhancement of the overall commute experience are assured in the proposed system.

The design of the application is focused on the convenience of the users: it should be responsive and allow for smooth interaction, assuring a seamless flow of data from the client to the server. With the integration of modern backend technologies with a reliable Android-based frontend, the proposed system ensures correct tracking of rides, minimizes waiting times, and offers a better commuting experience.

.

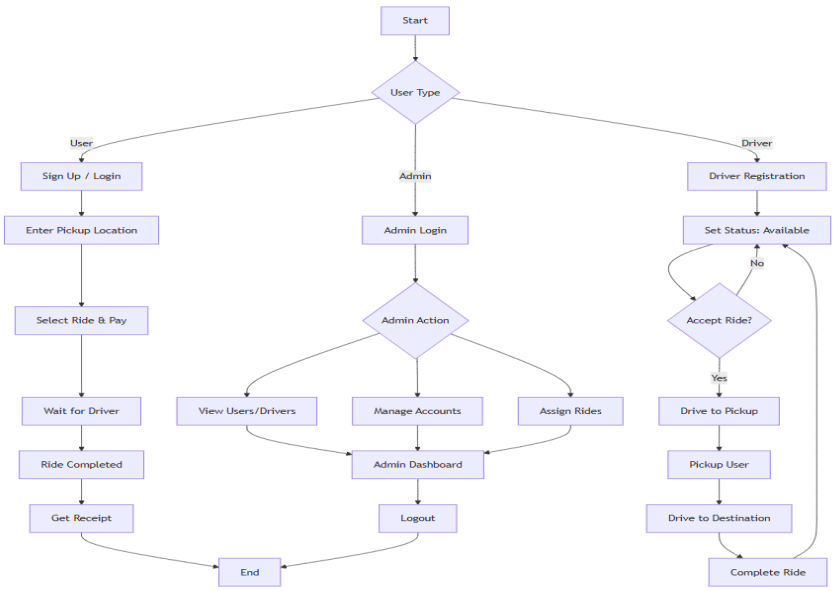

Fig 1. Overall flow diagram

The methodology of the Driver Booking App focuses on combining advanced technologies for easy and automated processing of booking rides. Users register, log in, and put pickup and drop locations that are processed using a Open Street Map to find nearby drivers. The backend, developed with Node.js and Express.js, assigns the nearest available driver and is also responsible for managing communication between the user and driver.
Thus, both parties are able to track each other's live location in real time until the ride is completed. All details of the trip will be accurately saved in MongoDB. The system will also include encrypted data transfer and authentication for secure access. Hence, the methodology on efficiency, user experience, real-time tracking, and a seamless booking experience is depicted in.

1. User Module:

The user module functions as a main link connecting customers and the system. It is a platform on which the user can log into the system and arrange for a ride. Once logged into the system, the user can be able to enter the source and destination points for pick up. The system will then process the request and link the user with drivers who are near and available. Once a driver Accepts and confirms the ride, it will be booked. The user module will also allow tracking of drivers' locations so that they can be aware of the timing for arrival. It will also allow automatic generation of an electronic receipt indicating costs and duration. This module focuses on convenience, transparency, and safety. As a result, the entire ride booking process will be smooth and reliable. By using a user-friendly interface and a secure transaction system, the User Module will ensure a smooth and hassle-free traveling experience for customers.

2. Driver Module:

The Driver module aims at assisting drivers effectively to deal with ride requests and carry out a smooth flow of communications with the system. The driver logs into the system and receives notifications about new ride requests. Depending on the schedules and locations of drivers, they are capable of accepting or rejecting ride requests. Once they accept, they receive information about the user and the destination so that they can arrive at the user’s location as soon as possible.

After dropping off the passenger, the driver begins the journey within the system interface. As soon as he reaches the destination, he ends the journey. The Driver module makes sure that drivers work efficiently with fewer efforts and with transparency within operations. It promotes synchronization and confirms safe communication, as well as contributing toward an efficient and properly structured transportation process. By doing so, it reduces risks and problems associated with passengers as well.

3. Admin Module:

The Admin module acts as the control center of the system, making it responsible for controlling and monitoring the activities of users and drivers. The admin can securely log in to the system in order to check the status of the registered users, verify driver details, and operate ride data management. This module manages the flow in a smooth way through maintaining data accuracy and resolving any issues that arise.

It will also allow the admin to generate reports, analyze performance, and keep the system secure and efficient. The Admin module, overseeing users and drivers alike, ensures overall transparency, coordination, and reliability within the entire ride-booking platform.

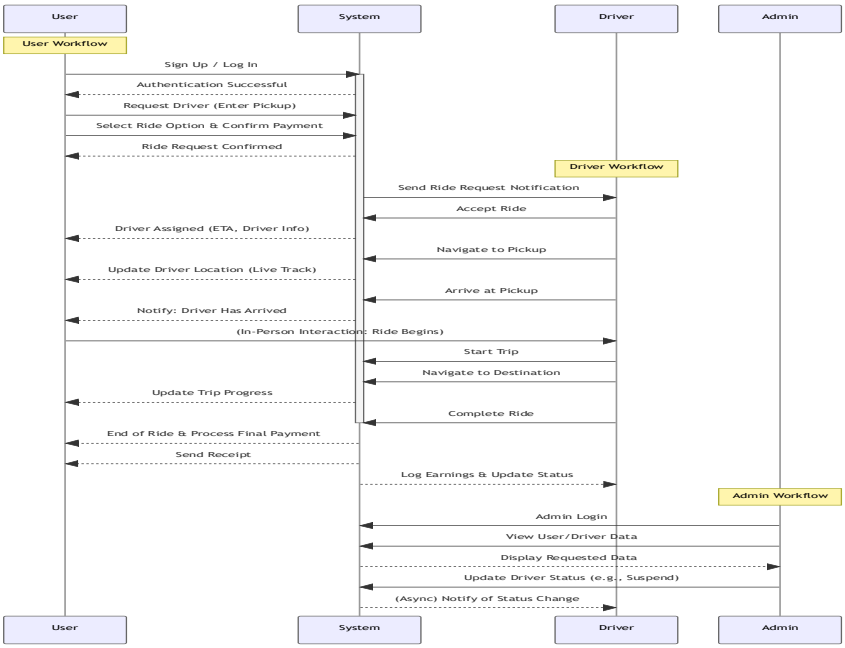

Fig 2. Data Flow Diagram.

III.SYSTEM IMPLEMENTATION

The implementation of the **Driver Booking App** focuses on integrating a user-friendly Android interface with a robust and scalable backend architecture. The system ensures seamless interaction among users, drivers, and administrators through real-time communication and efficient data processing. Based on the prototype demonstrated in the project synopsis, the system is implemented as a fully functional mobile application built using **Kotlin**, with backend services running on **Node.js/Express.js** and data stored in **MongoDB**. Each module is designed to perform reliably under real-time ride-booking scenarios, ensuring accuracy, responsiveness, and usability across all components.

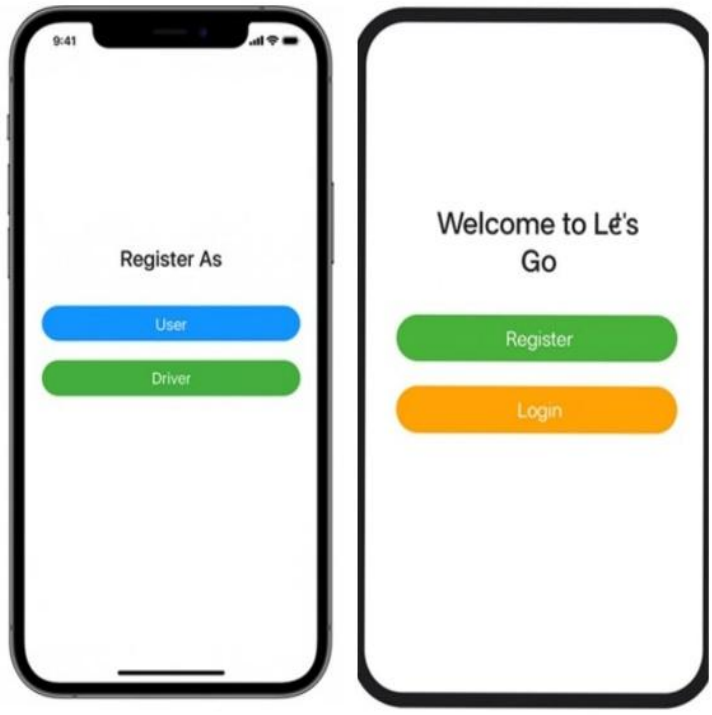

Fig 3. Register page for both users and Drivers

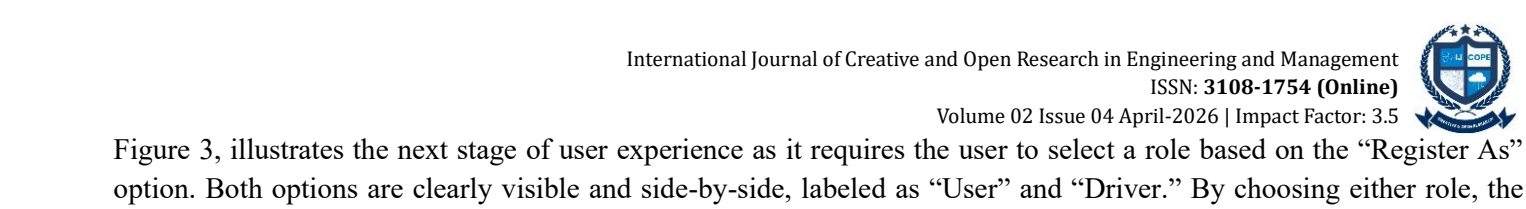

Figure 3, illustrates the next stage of user experience as it requires the user to select a role based on the "Register As" option. Both options are clearly visible and side-by-side, labeled as "User" and "Driver." By choosing either role, the user will be directed to either a user registration form or a driver registration form. Note how clean and simple it is.

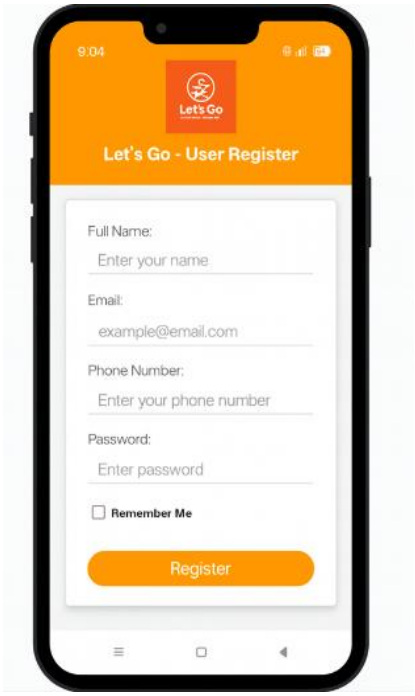

Fig 4. User Registration

As shown in Fig 4, users will have to input necessary information like 'name, 'email, 'phone number, and 'password' with a convenient top-to-bottom form layout. The form also includes necessary 'placeholder' entries with an added 'remember me' facility. The 'register' button at the end finishes this page.

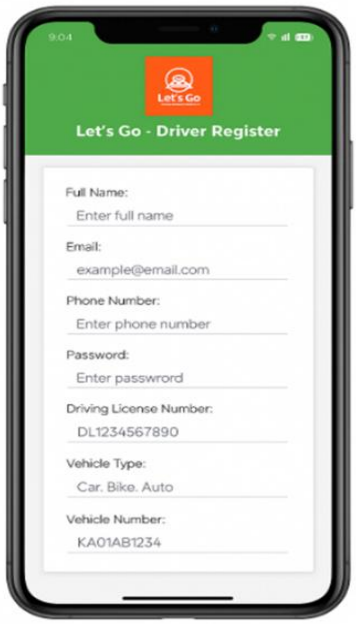

Fig 5. Driver Registration.

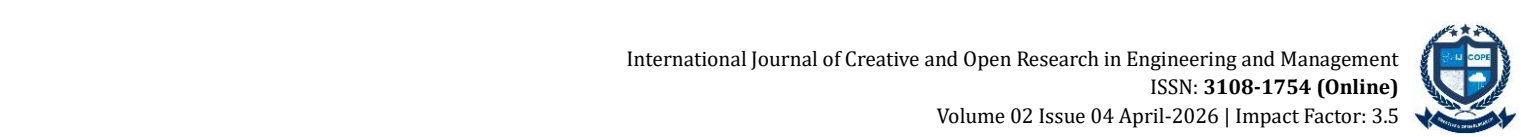

Figure 5 shows the "Driver Register" form that goes beyond simple user data to capture some very important professional information like Driving License Number, Type of Vehicle, and Vehicle Number that can act to validate the authenticity of the driver and make the platform safe. Given the structured input-taking-arguably dropdown for vehicle type-the form retains consistency and accuracy. Overall, this onboarding process will ensure that only a validated and qualified driver is able to get into the system.

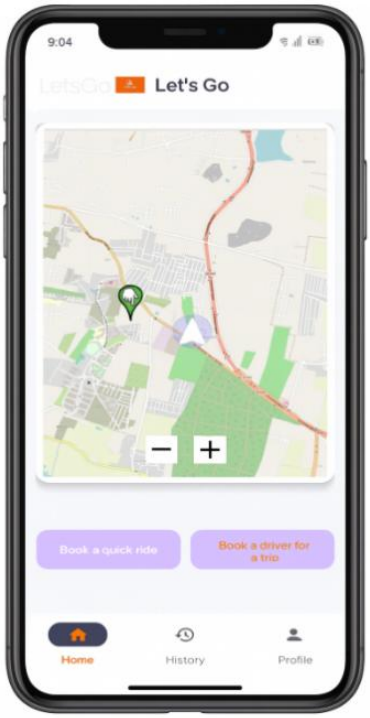

Fig 6. User Dashboard

Figure 6 confirms that all the required information concerning the driver is captured in a very neat, well-structured form that ensures smooth and secure onboarding. Such organization makes data entry easier and quicker to verify. Post-registration, Figure 4 also depicts the main user dashboard, complete with branding of the app, with the prompt "Book a quick ride." A neuroscience-backed, brightly colored "Book a driver for a trip" button acts as the main call-to-action for a user and seamlessly navigates them through destination selection, driver selection, and confirmation of the ride.

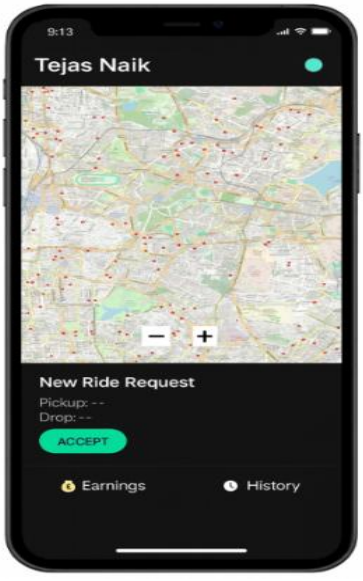

Fig 7. Driver Dashboard

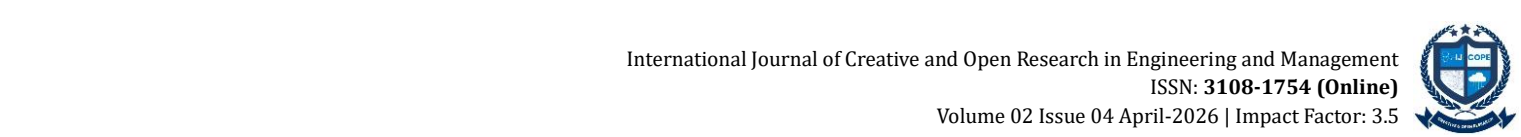

Figure 7 shows the active dashboard for a driver, which is viewed once there is a new ride request. The display begins with the driver's name at the top, followed by a "New Ride Request" window with pick-up and drop-off information, updating on a real-time basis. There is also a prominent "ACCEPT" button that enables the driver to act immediately. Other functions include "Earnings" and "History" at the bottom.

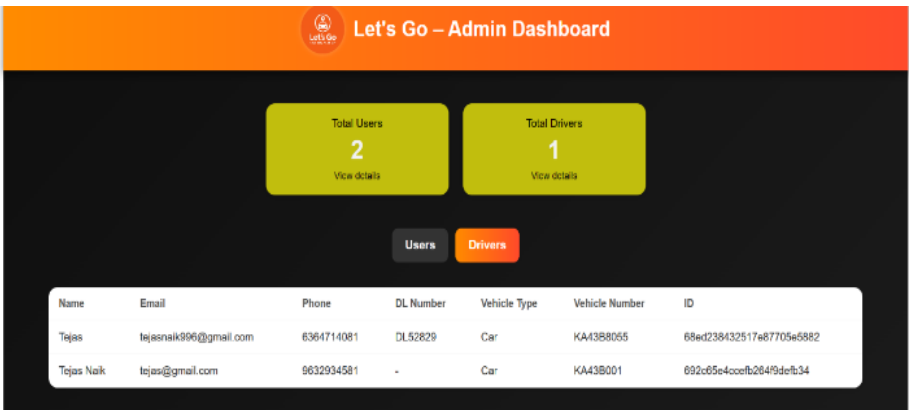

Fig 8. Admin Dashboard

Figure 8 Admin Dashboard. It has provided a very lucid view into the respective platform's activities regarding the management of users. The top allows for an overview of summary metrics, such as Total Users and Total Drivers, which can be directly utilized by administrators to understand the usage of the system. Underneath, a tabular presentation of structured driver details shows names, email addresses, phone numbers, license numbers, types of vehicles, and driver IDs for easy verification and monitoring.

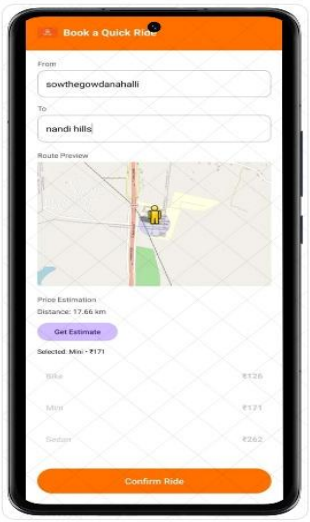

Fig 9. User Booking Ride Interface

Figure 9 presents the interface for ride-booking, where the user inputs the pickup and drop locations-in this case, Sowthegowdanahalli to Nandi Hills. Once the inputs are given, the app shows a preview of the route visually, which enables the user to confirm his path. Below the map, it presents estimated distance and fare options for Bike, Mini, and Sedan rides in a clear manner for easy comparability. One tap on the "Confirm Ride" button confirms the booking in an seamless manner.

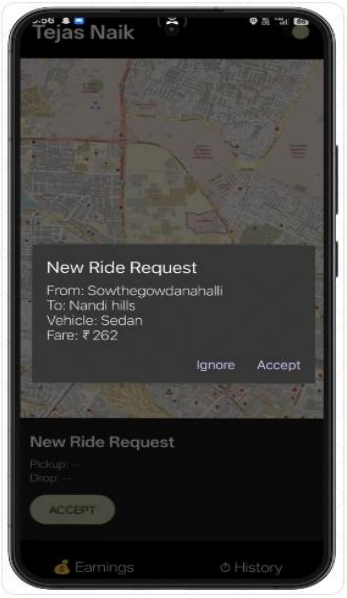

Fig 10. Driver Ride Acceptance

Figure 10 shows how a new ride request would look to a driver in a pop-up with key information: pickup point (Sowthegowdanahalli), destination (Nandi Hills), the selected ride type, and fare. In one glance, the driver can understand the requirement of the trip. Two basic options-"Ignore" and "Accept"-can enable a quick response based on availability. There is no clutter in this UI, which enables the driver to respond quickly and effectively for a seamless ride assignment process.

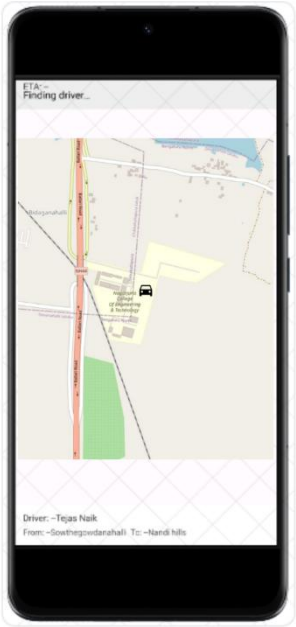

Fig 11. User Interface to Track the Driver

Figure 11 This is the “Finding Driver” interface once the user confirms the booking. The system is already finding the nearest available driver, with an appropriate status message as "ETA: Finding driver." at the top of the screen. The result is a centrally placed map preview using OpenStreetMap to help the user visualise the area in which search of a driver is currently being conducted. Below the map, key details of the trip are included, such as driver information, both pickup and drop locations, that helps give clarity and transparency to the user during this time-consuming assignment.

Fig 12. Driver On Going Activity

Figure 12 shows the Ride Details screen, which provides real-time tracking of the driver's movement and ride status. Its interface, at the top, displays an ETA like "~200m away," which tells the user just about how far the driver is away from the pickup point. In the middle of the screen, there is a dynamic preview of an OpenStreetMap by means of GPS data. Below the map, status buttons such as "On the way," "Arrived," "Start Trip," and "Complete" can be used by the driver to mark the progress of the ride after every stage. These updates keep both the backend and the user informed, allowing neat communication throughout the trip.

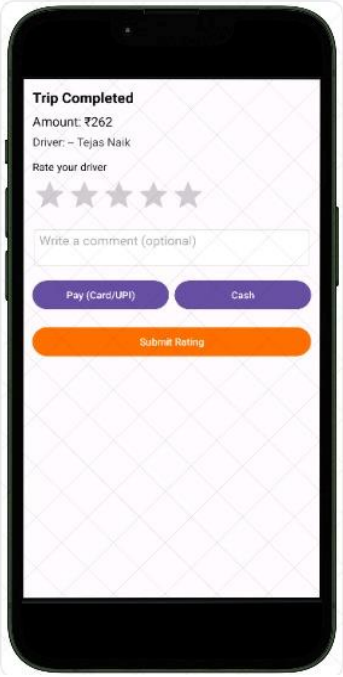

Fig 13. User Ride Complete Activity

Figure 13 shows the summary screen that would appear once the ride is over with the final fare in big font on the screen, and then the driver's name. User can give quick feedback by selecting star ratings out of five stars; there is also an optional comment box to provide more information. Just below it, the system is offering another option - Card/UPI or Cash. Once

the mode has been selected and payment has been done, the user will be done with tapping the "Submit Rating" option, thus making the experience after the ride smooth and user-friendly.

IV. CONCLUSION

The proposed **Driver Booking App** is an advanced and innovative solution that overcomes the limitations of traditional transportation systems. It provides an efficient, secure, and user-friendly platform for booking and tracking rides in real time. By integrating technologies such as **Open Street Map**, **Node.js**, and **MongoDB**, the system ensures accurate ride management, live tracking, and smooth communication between users and drivers.

The application emphasizes **reliability, transparency, and scalability**, offering features like location-based driver assignment, secure data handling, and a responsive design for easy interaction. Its architecture supports high performance and can be extended with additional modules such as fare estimation, in-app payment gateways, and ride history analytics.

Beyond convenience, the system is **cost-effective and accessible**, requiring only a smartphone with internet connectivity, making it adaptable for both urban and rural transportation needs. By automating the ride-booking process and minimizing manual intervention, the app saves time for users and drivers while ensuring accuracy and efficiency.

In summary, the **Driver Booking App** transforms the commuting experience by combining modern technology with user-centric design. It represents a significant step toward smarter, digitalized transportation—offering a **seamless, efficient, and scalable** solution that redefines the way people travel.

V. REFERENCES

[1] Vidushi Lukhoo, Raj Kishen Moloo, “A Carpooling System To Ease Traffic In Mauritius”, 2024.
[2] “Xuan Wang, Yaojie Li, Scott Smith, Helmut Schneider, “Data-driven decision-making: the case of ridesharing with implications for engineering managers”, 2024.
[3] Abhishek Shinde, Prajyot Bhoir, Sahil Shinde, Ms. Bushra Shaikh, “An Efficient Ridesharing Model using Machine Learning Based on Riders Reviews”, 2023.
[4] Haining Yu et al., “A Machine-Learning-Based Ride-Sharing Model Using Rider Reviews for Efficiency and Reliability,” 2023.
[5] Wang Peng and Lili Du, “A Novel Real-Time Coordinated Ridesharing Route Choice Mechanism”, 2023.
[6] Callista Ivana Mogie, Ricky, Gunawan Wang, Sfenrianto, Azlee bin Zabidi, Anderes Gui, “Improving Driver Loyalty through using Gamification Approach”, 2023.
[7] Haining Yu, Xiaohua Jia, Fellow, IEEE, Hongli Zhang, and Jiangang Shu, “Efficient and Privacy-Preserving Ride Matching Using Exact Road Distance in Online Ride Hailing Services”, 2022.
[8] Saidur Rahman, Apostolos Kalatzis, Mike P. Wittie, David L. Millman, Laura Stanley, “Dynamic Checkpoint Initiation in Serverless MEC”, 2022.
[9] Haining Yu, Xiangzhan Yu, Xiaojiang Du, Hongli Zhang and Mohsen Guizani, “PGRide: Privacy-Preserving Group Ridesharing Matching in Online Ride Hailing Services”, 2021.
[10] Tingting Kang, Lei Zhang, Yan Huang, “Development of App-based Ride-hailing Calculating Mileage and Time Detection Device”, 2021.
[11] Anushka Vijay Sant, Atharva Shrikant Naik, Anirban Sarkar, Vandana Dixit, “Driver Drowsiness Detection and Alert System: A Solution for Ride-Hailing and Logistics Companies”, 2021.
[12] Sumaiya Binte Akther, Md Anik Hasan, Nazifa Tasneem and Mohammad Monirujjaman Khan, “An Interactive Android Application to Share Rides with NSUers”, 2021.
[13] Yakob Utama Chandra, Meyliana, Michael Jhonsons, “Analysis of Digital Wallet for Higher Education Student using Online Transportation Network Services”, 2020.
[14] Josh Angelo Escalona, Benjamin Manalo, Wilbert Jethro R. Limjoco, Carl C. Dizo, “A Ride Sharing System based on An Expansive Search-Based Algorithm”, 2020.
[15] Kai Zhang, Shuang Zhang, “Dynamic Ridesharing Strategy for Shared Autonomous Vehicles Based on Maximum Similarity of Trajectories”, 2020.
[16] Alexey Kashevnik, Nikolay Teslya, Sergey Mikhailov, Mikhail Petrov, Anton Shabaev, and Andrey Krasov, “Ridesharing for Carsharing Service Provider: Driver and Pedestrian Route Matching”, 2020.