IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

Development of a Secure Car Sharing Application Using Flutter and Provider State Management

Prof. Mohan B. Yelpale[1], Ms. Sakshi E. Mundhe[2], Ms. Sakshi S. Sangave[3], Ms. Aachal B. Sangale[4], Mr. Saurabh S. Sargar[5]
Professor, Department of Computer Engineering[1]
Students, Department of Computer Engineering[2-5]
NBN Sinhgad Technical Institute Campus, Pune, India

Abstract: *This paper presents the design, development, and implementation of a car-sharing application built using Flutter, aimed at improving urban mobility while incorporating enhanced security features. The app leverages state-of-the-art technologies, including Firebase for backend services and Provider for efficient state management, offering cross-platform compatibility for both Android and iOS devices. Core features include user registration, real-time ride booking, vehicle tracking, and a security-focused system with SOS functionality and audio tracking during rides. A range of dependencies, such as google-maps-flutter for navigation and firebase-auth for secure user authentication were integrated to ensure a seamless user experience. The application's design and architecture emphasize scalability, ease of use, and responsiveness, and its performance was thoroughly tested to ensure reliability. This paper details the technological choices, security implementations, and the overall impact of these features on the user experience, concluding with key findings, performance analysis, and a discussion of potential future enhancements.*

Keywords: Car-sharing, Flutter, Firebase, Security features, Provider state management, Security, SOS.

I. INTRODUCTION

In today's fast-paced cities, car-sharing has become a smart and sustainable alternative to owning a car. It allows people to rent cars on-demand for short periods, helping to ease traffic, cut emissions, and lower the costs of car ownership. As more people embrace this model, there's a growing need for reliable and secure car-sharing apps.Mobile technology plays a huge role in this space, giving users easy access to cars, real-time location tracking, and simple reservation systems through their smartphones. Flutter, Google's cross-platform framework, has made building these apps easier by allowing developers to create apps for both iOS and Android from a single codebase. These speeds up development, reduce costs, and ensure a consistent user experience across platforms.

This paper focuses on developing a car-sharing app using Flutter, with a strong emphasis on security and efficient state management. While traditional car-sharing apps have been successful, concerns about data privacy, vulnerabilities in location tracking, and the handling of sensitive user information still remain. To address these challenges, this app integrates advanced security features like secure user authentication, encryption of personal data, and GPS tracking to ensure both user and vehicle safety.The security features are designed to give users peace of mind, knowing that their personal data, location, and trip history are safe from unauthorized access. The app uses secure login methods via Firebase Authentication, encrypts sensitive data, and handles GPS tracking in a secure way, ensuring that real-time vehicle updates don't compromise user privacy.

Another key part of the app is its use of Provider for state management. Managing state across different screens and components can get complicated in apps like this, where real time updates (like location changes, ride requests, and user interactions) are critical. Provider helps manage this dynamic data efficiently, allowing the app to keep user information, ride details, and location data updated in real-time without slowing down performance or making the code overly complex.This paper walks through the app's architecture, explaining how Flutter's widgets, Provider, and other essential tools work together to create a user-friendly, secure car-sharing experience. The design allows the app to scale

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 5, Issue 10, May 2025 Impact Factor: 7.67

as new features are added while keeping performance high. We'll also explore the challenges faced during development, including integrating security protocols, managing state, and optimizing the app for performance.

By building this app, the research demonstrates how Flutter can be used to create secure, powerful mobile applications for the car-sharing industry. We also look at how the security and state management strategies applied here can be useful in other apps that handle real-time data and need strong privacy protections. This work not only contributes to the growth of secure car-sharing platforms but also highlights best practices for mobile app security and state management in cross-platform environments.

II. LITERATURE SURVEY

A systematic literature review of ride-sharing platforms, user factors and barriers:

From the literature on ride-sharing, it appears as if one such service that best supports innovation and sustainability is by cutting down the number of vehicles on the road, which in turn reduces carbon emissions and promotes more efficient use of resources. Understanding why people choose ride-sharing is key factors like convenience and safety play a big role in encouraging adoption. However, the survey also highlighted obstacles, such as regulatory issues and social perceptions, which can slow down growth. Addressing these challenges is essential for ride-sharing to be widely embraced. Technology is a major enabler of ride-sharing, with online platforms and mobile apps making it easy to match riders in real time. Successful approaches often involve partnerships with local governments and combining ride-sharing with public transport to improve accessibility. Overall, the survey underscored the important connections between technology, user behavior, and environmental benefits in making ride-sharing a truly sustainable transport option.

Carsharing: a systematic literature review and research agenda:

Here's a more conversational version of your systematic literature review summary: From the systematic review on car-sharing, I found that there's been a significant rise in research on the subject, highlighting its growing importance in urban mobility and sustainable transport. The review focused on key themes like user behavior, environmental benefits, operational models, and technology integration. Understanding what motivates people to use carsharing, as well as the obstacles to adoption, was noted as crucial for getting more participants on board. The environmental advantages of carsharing, such as cutting carbon emissions and improving resource use, were also emphasized. Innovations like mobile apps and real-time data analytics play a big role in improving user experience and making services more efficient. However, the review pointed out gaps in the research, especially when it comes to the long- term sustainability of carsharing models and how they can be better integrated with public transport. These gaps provide a solid starting point for future studies and have practical implications for improving carsharing services.

Ridesharing and Crowdsourcing for Smart Cities: Technologies, Paradigms and Use Cases:

From this research paper, I learned that recent advances in technology, mobile devices, and Internet connectivity have played a big role in transforming ridesharing and crowdsourcing in smart cities. Unlike traditional carpooling, where trips are planned in advance, ridesharing in smart cities is much more dynamic, with drivers and riders often connecting on short notice. Crowdsourcing is essential for making these quick matches, but challenges like security, policies, and pricing strategies still need to be addressed. The paper also highlighted how artificial intelligence and autonomous vehicles can boost the efficiency of ridesharing, demonstrating the potential of these technologies to reshape urban transportation. It provided a look at the technological framework needed for ridesharing platforms, emphasizing the importance of strong privacy and security strategies. Overall, the research stressed that smart technologies, along with well-thought-out policies, are key to overcoming the challenges and ensuring the success of ridesharing platforms in smart cities.

Real-Time Carpooling and Ride-Sharing: Position Paper on Design Concepts, Distribution and Cloud Computing Strategies:

From this research paper on real-time carpooling and ride- sharing, I learned how crucial it is to design systems that are scalable and accessible across different platforms for future ride-sharing solutions. The paper points out that many earlier carpooling and ride-sharing platforms struggled to gain widespread adoption because they weren't built to handle large numbers of users, which led to poor user experiences as the platforms expanded. Additionally, limited

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 5, Issue 10, May 2025 Impact Factor: 7.67

availability on both mobile and desktop devices kept them from reaching a global audience. The paper stresses the importance of thoughtful design, solid distribution methods, and effective use of cloud computing to ensure that these systems can scale up and work smoothly across different platforms. By following these principles, future ride-sharing solutions will be able to offer consistent and reliable services, no matter the demand or where the users are located.

Carpooling Platforms as Smart City Projects: A Bibliometric Analysis and Systematic Literature Review:

From this research paper on carpooling platforms in smart cities, I learned that while carpooling is known for its cost-sharing benefits, it still struggles with adoption due to the convenience of driving alone, increasing car ownership, and challenges in finding matching travelers. The paper highlights that technology plays a key role in making smart carpooling plat- forms more efficient, improving ride-matching and helping to reduce traffic and emissions in smart cities. For carpooling to successfully integrate into smart city infrastructure, the paper emphasizes the need for proper architectural support things like linking to high-occupancy vehicle lanes, parking, toll systems, and public transportation. It also notes that smart carpooling platforms can create added value for both users and cities, making them an appealing mobility solution for urban areas. The suggested multi-faceted business model enables carpooling services to meet the needs of various customer segments and partners. Ultimately, the paper presents a carpooling platform architecture that fits seamlessly into the broader smart city ecosystem, making carpooling a more viable and attractive option.

III. METHODOLOGY

Requirement Analysis:

The requirement analysis phase involved a comprehensive examination of user needs and current trends in the car-sharing industry, which was essential for identifying the most valueable features and functionalities among potential users. Surveys and interviews were conducted to gather insights on user preferences regarding car-sharing services. Key aspects high- lighted during this research included ease of use, reliability, safety, and the ability to access vehicles on demand. From this feedback, essential features such as user registration, ride booking, and real-time location tracking were identified as critical components of the application. User registration was emphasized as needing a straightforward process, while the ride booking interface required an intuitive design for quick searches and management. Real-time location tracking was deemed vital to enhance transparency and trust in the service. Additionally, an analysis of existing car-sharing applications helped uncover current market trends and user expectations. This involved studying user reviews, feature lists, and competitor offerings to identify successful features and common shortcomings. A significant trend identified was the increasing demand for enhanced security measures, as users expressed growing concerns about their safety and data protection. This led to a heightened interest in features such as user authentication, data privacy, and secure payment processing. By focusing on these identified needs and trends, the requirement analysis established a strong foundation for designing a user-centric car-sharing application that is secure, efficient, and enjoyable to use, ultimately enhancing the overall user experience in the car-sharing ecosystem.

Application architecture design:

When designing the application architecture, we took a structured approach to seamlessly integrate the key components: the user interface (UI), backend server, and database management. The UI was crafted to be intuitive and engaging, making it easy for users to register, book rides, and track locations in real time. The backend server, built on Firebase, acts as the central hub, managing user authentication, storing ride data, and facilitating communication between the app and the database. This architecture not only handles essential functions but also ensures scalability and responsiveness as the user base and data needs grow.

To enhance the user experience, we created wireframes and user flow diagrams to map out the app's design and interactions. The wireframes served as a blueprint for each screen's layout, focusing on the logical placement of elements. Meanwhile, the user flow diagrams showed the various paths users might take, from signing up to booking a ride. These visual tools helped us identify and resolve potential usability issues, ensuring a smooth experience before moving into development. By prioritizing user-centered design, we aimed to not only meet functional requirements but also maximize user satisfaction and engagement with the car-sharing app.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

Technology selection:

We chose Flutter as the primary development framework because of its powerful cross-platform capabilities, which allow for efficient development on both Android and iOS with a single codebase. This approach greatly reduces the time and resources needed for development and ongoing maintenance. Flutter's extensive collection of pre-designed widgets and customizable components also makes it easy to create visually appealing and consistent user interfaces across different devices. With Flutter, the app can provide users with a smooth, seamless experience, no matter which platform they are using.

For the backend, we selected Firebase due to its robust feature set tailored for real-time database management, user authentication, and cloud functions. Firebase handles user data and ride information effortlessly, ensuring smooth and efficient interactions throughout the app. Its real-time database ensures that changes are reflected instantly, so users are always upto date on ride statuses and vehicle locations. Additionally, Firebase Authentication offers secure and easy-to-use options for account creation and login, providing a user-friendly experience while keeping data safe.

Utilized Packages:

google-maps-flutter: The google-maps-flutter package integrates the power of Google Maps directly into the Flutter application, providing users with an interactive and detailed mapping experience. This allows users to explore maps, search for specific locations, and view routes, all within the app. It enhances the car-sharing experience by making it easy for users to visualize their surroundings, locate nearby vehicles, and identify pickup and drop-off points accurately. The ability to zoom in, pan around, and interact with map elements helps users feel more in control of their journey, improving overall satisfaction and trust in the service.

flutter-polyline-points: The flutter-polyline-points pack- age enables the creation of polylines, which are used to draw routes between a journey's start and end points on the map. This feature visually represents the entire travel path, making it clear to users where they are heading and how they'll reach their destination. By seeing the route laid out before them, users gain confidence in the navigation and tracking process, reducing anxiety and improving transparency, especially in unfamiliar areas.

shared-preferences: The shared-preferences package is crucial for persistent local storage of user data within the app. It allows the app to save essential information such as user preferences, login details, selected vehicle types, and other personalized settings. This ensures that users enjoy a seamless experience, as their preferences are remembered across sessions without requiring them to re- enter information. The ability to save and retrieve settings locally contributes to a smoother, more personalized app experience, improving user retention and engagement.

provider: The provider package is a state management solution that simplifies handling and distributing data changes across the app. It makes it easy to manage and propagate updates to the user interface (UI) whenever there's a change in the app's underlying data. For example, when ride information is updated or when a user's location changes, the UI reflects these changes instantly and efficiently. This creates a responsive, dynamic interface that adapts to real-time events without requiring complex manual intervention. As a result, provider enhances both the app's performance and the user experience by keeping the interface up-to-date and ensuring smooth interaction.

firebase-auth: The firebase-auth package is responsible for managing user authentication securely and reliably. It supports various authentication methods, including email and password logins, phone number verification, and third-party logins via Google, Facebook, or Apple. By offering a flexible range of options, it allows users to register and log in using their preferred method, making account management more convenient. Additionally, Firebase handles the security of user data, ensuring that sensitive information like login credentials is protected from unauthorized access. This robust authentication system is key to building user trust, especially in an app where personal and financial data might be involved.

cloud-firestore: The cloud-firestore package serves as the cloud-based database for the application, allowing it to store and retrieve user data, ride information, and trans- action details in real time. It supports complex queries and listens for updates, meaning that as soon as any information (such as a ride status or vehicle location) is changed, the app reflects this immediately. This real-time capability is essential in a car-sharing app, where up-to- date information about

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

available rides, user profiles, and vehicle locations is crucial. Cloud Firestore's scalability also means the app can handle increasing amounts of data without performance degradation as the user base grows.

geolocator: The geolocator package enables real-time GPS tracking, allowing the app to retrieve accurate location data for both users and vehicles. This functionality is fundamental to car-sharing, as it helps users track the vehicle's location, get precise directions, and receive estimated arrival times. Whether users are looking for a nearby car or checking how long it will take for their ride to arrive, the geolocator ensures that they receive timely and reliable updates. By leveraging this package, the app can provide users with a transparent and efficient ride-tracking system, enhancing the overall experience.

IV. IMPLEMENTATION OF CORE FEATURES

The implementation of core features started with the integration of user registration and authentication using Firebase Authentication. This step was crucial for creating a secure environment where users can safely log in and manage their accounts. Firebase Authentication supports multiple methods, such as email and password, ensuring accessibility and ease of use for a wide range of users. By prioritizing secure and flexible account management, the foundation for user trust and data security was established. Following authentication, the ride booking feature was developed. This functionality allows users to choose from available vehicles, set their pickup and drop-off locations, and view nearby cars in real time. By integrating real-time data, users can make informed decisions based on their needs, such as selecting specific vehicles or finding the most convenient ride. This streamlined approach not only simplifies the booking process but also enhances the overall user experience by providing immediate access to essential information like vehicle availability and location.

The next core feature involved integrating Google Maps, accomplished using the google-maps-flutter and flutter-polyline- points packages. This integration brought real-time navigation and vehicle tracking capabilities to the app. Users can visualize their journey on a map, with routes clearly marked from the start to the destination. This feature enhances user confidence and convenience, as they can easily follow their trip and stay updated on the location of their vehicle in real time. Collectively, these core features—secure user authentication, intuitive ride booking, and real-time navigation—build a comprehensive and user-friendly car-sharing application. The thoughtful integration of these functionalities ensures the app effectively meets user needs, delivering a reliable, convenient, and secure experience

V. SECURITY FEATURES

To prioritize user safety and security within the car-sharing app, several key features were implemented, including an SOS function and audio tracking during rides. These features were designed to ensure that users feel protected and have access to immediate help in case of emergencies. The SOS number feature allows users to set up a designated emergency contact within their profile. In an emergency, users can quickly tap a button to send an alert to their selected contact. This alert not only notifies the contact but can also include the user's real- time location, ensuring that help can be directed to their exact position. By providing users with an easily accessible safety net, this feature significantly boosts confidence in using the service, knowing that immediate help is just a tap away. Additionally, the app can offer options for sharing location details via SMS or popular messaging apps, enabling users to keep their emergency contacts updated on their whereabouts throughout the ride. Beyond the SOS feature, the audio tracking functionality provides users with the option to record audio during their trips. This real-time monitoring ensures that any incidents or unsafe situations are captured as evidence, adding an extra layer of protection. In the event of disputes or emergencies, the recorded audio can serve as valuable proof, supporting users in resolving issues. Furthermore, the presence of audio tracking acts as a deterrent to potential misconduct from drivers, as they are aware that their actions are being monitored. To respect user privacy, the app can offer options for reviewing or deleting recorded audio after the ride is complete, giving users control over their data while still benefiting from the added security.

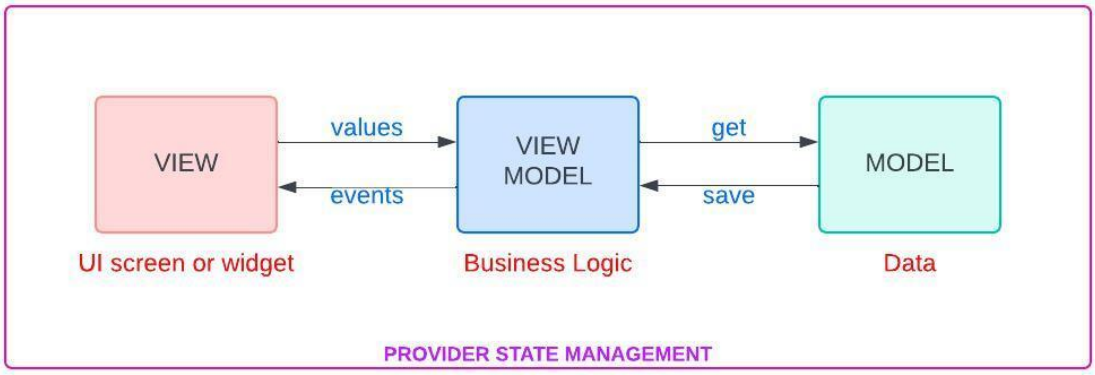
IV. STATE MANAGEMENT

Effective state management is critical to building a responsive and user-friendly application, and in the case of this car-sharing app, the Provider package was employed to manage the application's state across various screens and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

components. Provider offers a structured, scalable solution for managing shared data, ensuring that different parts of the app can access and update information seamlessly. By centralizing state management, the app maintains a consistent data flow, allowing its components to react dynamically to changes in state. This architecture simplifies development by eliminating the need to pass data manually through numerous widget layers, making it easier to manage and update information as the app evolves.

To enable real-time updates, the ChangeNotifier class was utilized within the Provider framework. ChangeNotifier acts as a notification system that alerts registered listeners whenever the application state changes. Developers can define specific properties within their state model using ChangeNotifier, and when those properties are updated, all listeners that rely on the updated data are automatically notified. This ensures that only the necessary UI components are updated, resulting in a dynamic and interactive user experience without causing performance lags.



One of the key benefits of ChangeNotifier is its ability to minimize unnecessary re-renders, ensuring that only the affected parts of the user interface are rebuilt when there is a state change. This selective updating improves overall app performance and ensures a smoother user experience. Whether users are viewing available rides, tracking their vehicles in real- time, or managing their profiles, the app responds promptly and efficiently to state changes without overburdening the system. By integrating Provider and ChangeNotifier into the app's state management, the development process is streamlined, making it easier to manage complex interactions and real- time updates. This responsive state management architecture not only enhances the app's functionality but also contributes significantly to user satisfaction by providing a seamless, smooth experience at every touchpoint.

VII. RESULTS

The car-sharing application showed impressive performance across important metrics like UI responsiveness, load times, and overall efficiency. By utilizing Flutter's widget architecture, the app delivered smooth transitions and interactions, resulting in a seamless user experience. Load times were optimized with efficient data handling, employing asynchronous loading for features like ride availability and user profiles, which meant users could access information quickly without noticeable delays.

Efficiency was further enhanced by reducing unnecessary re- renders through the Provider state management pattern. This allowed the app to run smoothly, even with moderate system resource constraints. The implementation of the Provider pattern played a key role in improving performance related to memory and resource management. By centralizing state management, Provider decreased the need for excessive data passing between widgets, which can lead to memory issues and inefficient resource use. With ChangeNotifier in place, only the components affected by state changes were rebuilt, leading to lower memory usage, as fewer widget instances were kept in memory at any given time. Overall, the combi- nation of Flutter's performance capabilities and the Provider pattern resulted in a responsive and efficient application.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

VIII. DISCUSSIONS

Key findings:

Integrating Provider into the car-sharing application greatly enhanced both stability and performance. By centralizing state management, Provider simplified the data flow within the app, leading to more efficient updates and smoother interactions between components. As a result, there were fewer instances of crashes, and transitions became more fluid, all of which con- tributed to an improved user experience. Additionally, the use of ChangeNotifier allowed for real-time updates without lag, ensuring users received immediate feedback on their actions, whether they were booking a ride or updating their profile information. Overall, the combination of Flutter's capabilities and the Provider pattern created a more stable and responsive application.

The security features implemented in the app, such as the SOS number functionality and audio tracking, had a significant impact on user trust and overall usability. Users reported feeling safer while using the app, knowing they had quick access to emergency support and could document their rides. This heightened sense of security positively influenced their willingness to use the service, leading to better user retention and satisfaction. Moreover, effectively communicating these security measures in the app's marketing materials helped build trust with potential users, distinguishing the app from competitors that may not prioritize such features.

Comparison with other solutions:

When comparing the security features of this app to those of other car-sharing solutions, it's clear that the implemented features offer a distinct advantage. Many popular car-sharing apps primarily emphasize standard safety protocols, such as background checks for drivers and in-app reporting systems. However, the unique combination of SOS number functionality and real-time audio tracking provides a higher level of security and reassurance for users. While some competitors may include emergency contact features, few offer audio documentation capabilities, which can be valuable evidence in case of disputes or emergencies. This emphasis on enhanced security features positions the app favorably in a competitive market, attracting users who prioritize their safety.

Challenges and Limitations:

Despite the advantages, several challenges emerged during the implementation of security features and state management with the provider. One major challenge was making the SOS functionality both accessible and intuitive, especially in stressful situations when users might need to reach it quickly. Additionally, integrating audio tracking raised concerns about user privacy and data management. This required careful consideration of how to communicate data handling practices to users while ensuring compliance with regulations. Managing state with Provider also presented a learning curve, particularly in effectively using ChangeNotifier to avoid unnecessary re-builds and maintain optimal performance. This aspect of state management necessitated thorough testing to ensure the app remained responsive under different conditions.

Potential limitations of the application include scalability issues, especially as the user base grows. The current architecture may require further optimization to efficiently handle increased data loads and real-time updates. While the audio tracking feature enhances security, it may introduce challenges regarding data storage and processing, particularly concerning user consent and privacy regulations. Ongoing monitoring and updates will be essential to address these scalability and privacy challenges as the application evolves and expands its user base.

IX. CONCLUSION

In conclusion, the development of the car-sharing application successfully merges advanced security features with efficient state management to deliver a safe and user-friendly experience. By incorporating the Provider pattern for state management, the app has seen notable improvements in stability and performance. This integration ensures seamless data flow between various components, making user interactions smooth and responsive. Real-time updates provided by ChangeNotifier have further enhanced user engagement, allowing for immediate feedback on actions like ride bookings and profile changes. As a result, users enjoy a dynamic and fluid experience, which is crucial in a competitive market. The implementation of robust security features has also been vital in building user trust and satisfaction.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 10, May 2025
Impact Factor: 7.67

Features such as the SOS number functionality and audio tracking offer users an increased sense of safety during their rides. Users can easily access emergency support, and the ability to document their rides adds an important layer of protection. This emphasis on security not only sets the app apart from other car-sharing solutions but also resonates with users who prioritize safety, ultimately improving user retention and loyalty. Feedback from beta testing indicates that these security measures have been well-received, affirming their effectiveness and importance in the overall app design.

While the development process faced challenges, particularly with integratingsecurityfeatures and managing state using Provider,the lessons learned have fortified the app's foundation. Issues related to user accessibility for emergency features and managing user privacy regarding audio tracking were addressed thoughtfully, reflecting a commitment to user safety and regulatory compliance. Looking ahead, potential limitations, such as scalability and data management, will require ongoing monitoring and adaptation as the user base expands. Future updates will focus on enhancing performance and adding features based on user feedback and emerging trends in the car-sharing market. Overall, this project not only showcases the technical capabilities of Flutter and Firebase but also underscores the critical importance of prioritizing user safety and experience in app development. As the car-sharing app evolves, its innovative features and dedication to security will keep it competitive in the industry, meeting the changing needs of users and fostering a safer community for everyone.

REFERENCES

[1]. A systematic literature review of ride-sharing platforms, user factors and barriers by, Lambros Mitropoulos, Annie Kortsari and Georgia Ayfantopoulou
[2]. Carsharing: a systematic literature review and research agenda by,Brenda Nansubuga, Christian Kowalkowski
[3]. Ride-sharing and Crowdsourcing for Smart Cities: Technologies, Paradigms and Use Cases by, Kah Phooi Seng, Li-Minn Ang, Ericmoore Ngharamike, Eno Peter
[4]. Real-Time Carpooling and Ride-Sharing: Position Paper on Design Concepts, Distribution and Cloud Computing Strategies by, Dejan Dimitrijevic, Nemanja Nedic, Vladimir Dimitrieski
[5]. Carpooling Platforms as Smart City Projects: A Bibliometric Analysis and Systematic Literature Review by, Leonidas G. Anthopoulos, Dimitrios N Tzimos
[6]. https://docs.flutter.dev/data-and-backend/state-mgmt/simple
[7]. https://pub.dev/packages/geolocator