# **1. Introduction**

## 1.1 Purpose of the SRS

The purpose of this Software Requirements Specification (SRS) is to define the functional and non-functional requirements for the Campus Ride-Sharing Platform with Parking System Integration. This document serves as a guide for developers, stakeholders, and users, ensuring a clear understanding of the platform’s objectives and expected functionalities. It provides a comprehensive description of the system requirements, use case models, and design diagrams, following the ISO/IEC/IEEE 29148:2018 standard. The SRS will help ensure consistency, clarity, and completeness throughout the development process.

## 1.2 Scope of the system

The Campus Ride-Sharing Platform is designed to facilitate carpooling among university community members, including students, faculty, and staff. The system integrates with the campus parking management system and digital ID verification to reduce parking demand and provide real-time parking availability. The platform aims to promote sustainable transportation practices while improving campus parking efficiency.

a) Software Product: Campus Ride-Sharing Platform with Parking System Integration

b) Functionality: Facilitates carpooling, displays real-time parking availability, verifies users via digital ID, and supports parking slot booking during carpooling.

c) Application: Aimed at university community members to promote sustainable transportation, reduce traffic congestion, and enhance parking efficiency on campus.

d) Consistency: The scope of this software aligns with the university’s sustainable transportation and parking management initiatives. The Campus Ride-Sharing Platform is designed to facilitate carpooling among university community members, including students, faculty, and staff. The system integrates with the campus parking management system and digital ID verification to reduce parking demand and provide real-time parking availability. The platform aims to promote sustainable transportation practices while improving campus parking efficiency.

## 1.3 Definitions, acronyms, abbreviations

This section provides definitions, acronyms, and abbreviations used throughout the SRS to ensure clarity and consistency.

* SRS: Software Requirements Specification - A document that outlines the requirements of the software system.
* FR: Functional Requirement - A specification of what the system must perform.
* NFR: Non-Functional Requirement - A specification of how the system performs certain functions.
* API: Application Programming Interface - A set of functions allowing software to interact with other software.
* ID: Identification - A means of recognizing or verifying an entity.
* ERD: Entity-Relationship Diagram - A visual representation of data relationships.
* MVC: Model-View-Controller - An architectural pattern used to separate the internal representation of information from the way it is presented to and accepted from the user.

## 1.4 References

# **2. Overall Description**

## 2.1 Product perspective

The Campus Ride-Sharing Platform is an independent software application designed to work seamlessly with the university’s existing digital ID and parking management systems. It serves as a coordination tool for ride-sharing and parking availability updates. The platform is part of the university's broader sustainable transportation and parking management initiative. It directly interfaces with the digital ID verification system to ensure secure user authentication. Additionally, it integrates with the campus parking management system to provide real-time availability updates.

System Interfaces:

* Integrates with the university digital ID system for user authentication.
* Connects to the campus parking management system for availability data.
* Facilitates real-time data synchronization between parking availability and carpool coordination.

User Interfaces:

* Mobile and web applications with a user-friendly graphical interface.
* Provides a dashboard displaying ride options, carpool history, and parking availability.
* User authentication and profile management interfaces.

Hardware Interfaces:

* Supports mobile devices, desktops, and parking sensor devices.
* Connects to mobile devices through standard mobile OS interfaces (Android/iOS).
* Uses QR code scanners for digital ID verification.

Software Interfaces:

* Digital ID API: Integrates with the university’s ID verification system.
* Parking Management API: Interfaces with real-time parking availability services.
* Mapping Service: Uses external API for route suggestions and carpool tracking.

Communication Interfaces:

* Uses HTTPS for secure data communication.
* Supports push notifications via Firebase for ride updates.

Memory Constraints:

* Requires at least 2GB of device RAM for optimal performance.
* Data storage requirements depend on the number of users and ride records.

Operations:

* User-initiated ride searches, carpool creation, and parking slot booking.
* Automatic update of parking availability when users check in.
* Backup and recovery through cloud storage services.

Site Adaptation Requirements:

* Configurable for specific campus policies, including reserved parking zones.
* Customizable UI to reflect the university’s branding.

Interfaces with Services:

* Cloud-based data management for ride and parking data.
* Authentication services via university digital ID API.
* Students: Primary users who will share rides and check parking availability.
* Faculty/Staff: Secondary users, similar to students but may have reserved parking zones.
* System Administrators: Manage user data, monitor system performance.

## 2.2 Product functions (overview)

The Campus Ride-Sharing Platform provides the following major functions:

* User Authentication: The system verifies users through the university digital ID system, ensuring that only authorized members of the university community can access the platform.
* Ride Coordination: Users can create and join carpool groups through the platform, selecting preferred routes, pick-up points, and co-riders. The system provides suggestions based on proximity and user preferences.
* Parking Management: The system displays real-time parking availability on campus, allowing users to view and book parking slots as part of the carpooling process. Integration with the campus parking management system ensures that updates are accurate and timely.
* Route Planning: Integrates with a mapping service to provide optimal route suggestions, taking into account factors such as distance, traffic, and carpool member locations.
* Notifications: The platform sends notifications about ride statuses, carpool confirmations, parking updates, and any changes related to scheduled rides. These notifications are sent via push notifications on mobile devices.
* Data Management: The system maintains user profiles, ride history, and parking data, ensuring that information is stored securely and can be retrieved when needed.
* User Interface: A clean, intuitive interface that supports both web and mobile access, making it easy for users to navigate ride options, parking details, and notifications.
* Reporting and Analytics: Administrators can access data on carpool usage, parking occupancy, and system performance to make data-driven decisions related to campus transportation management.

## 2.3 User characteristics

The primary users of the Campus Ride-Sharing Platform include students, faculty, and staff members within the university community. These users will vary in terms of educational level, technical expertise, and accessibility needs. Understanding the characteristics of these users helps to develop an intuitive and inclusive platform.

* Students: Typically possess intermediate to advanced digital literacy, familiar with mobile apps and web interfaces. They primarily use the platform to find carpools, book parking slots, and coordinate rides with peers. Students may have varying schedules and often prefer flexible ride options.
* Faculty/Staff: Generally proficient with digital tools and mobile applications. Their usage focuses on parking availability, booking reserved spots, and occasionally participating in carpools. Faculty members may have specific accessibility needs, such as support for assistive technologies.
* System Administrators: Technically proficient users who manage platform settings, monitor data, and resolve issues. They require a robust, user-friendly administrative dashboard to efficiently manage user profiles, parking data, and ride coordination.
* Accessibility Considerations: Some users may have disabilities requiring assistive technologies, such as screen readers or voice commands. The platform must be compliant with accessibility standards to ensure usability for all members of the university community.

## 2.4 Constraints (e.g., university policies, digital ID APIs)

The development and implementation of the Campus Ride-Sharing Platform are subject to several constraints:

* Compliance with University Policies: The platform must adhere to university regulations regarding transportation, parking, and data privacy. Any changes to these policies may affect the platform's functionality or operations.
* Integration with Digital ID APIs: The system must securely integrate with the university’s digital ID verification system to authenticate users. This requires adherence to API usage policies and data protection guidelines.
* Network Availability: The system relies on stable internet connectivity on campus. Limited or no internet access may hinder platform functionality, particularly in remote or underground parking areas.
* Mobile Device Compatibility: The platform must support a range of devices with varying operating systems (e.g., Android, iOS) and screen sizes, ensuring a consistent user experience.
* Data Security Compliance: User data, including ride history and parking information, must be stored securely, complying with local data protection regulations.
* Interoperability: The platform must work seamlessly with existing campus parking management systems and external mapping services without disrupting their current operations.
* Resource Limitations: The platform must function efficiently within the existing IT infrastructure, minimizing the need for additional hardware or software resources.

## 2.5 Limitations

The Campus Ride-Sharing Platform is subject to the following limitations:

* Regulatory Requirements and Policies: The platform must comply with all applicable university policies and regulations related to transportation, parking, and data protection.
* Hardware Limitations: The platform is dependent on mobile device capabilities, including GPS accuracy and internet connectivity. Limited hardware resources may affect performance, especially on older devices.
* Interfaces to Other Applications: The platform must integrate seamlessly with the university’s digital ID system, parking management system, and external mapping APIs. Disruptions or updates to these systems may affect functionality.
* Parallel Operation: The platform must support simultaneous use by multiple users, including managing carpool creation, parking bookings, and real-time updates without performance degradation.
* Audit Functions: The system will maintain logs of user interactions and data updates for monitoring and troubleshooting.
* Control Functions: Administrative functions will include user management, carpool moderation, and data integrity checks.
* Higher-Order Language Requirements: The platform will be developed using standard web and mobile programming languages and frameworks to ensure compatibility and maintainability.
* Signal Handshake Protocols: The system will use standard communication protocols (e.g., HTTPS) to ensure secure data transfer between the client and server.
* Quality Requirements: The platform aims to maintain high reliability and performance, particularly during peak usage hours.
* Criticality of the Application: The platform is considered critical for daily campus transportation coordination, and any downtime may significantly impact users.
* Safety and Security Considerations: The system must prevent unauthorized access and protect user data through robust authentication and encryption methods.
* Physical/Mental Considerations: The user interface must be accessible, with options for visually impaired users and easy navigation.
* Dependencies on Other Systems: The platform's real-time data accuracy depends on the availability and stability of the university’s parking management and digital ID systems.

## 2.6 Assumptions and dependencies

The development and operation of the Campus Ride-Sharing Platform rely on several assumptions and dependencies that influence its requirements and performance. Any change in these factors may affect the platform's functionality or require modifications to the SRS.

* Operating System Compatibility: It is assumed that the platform will be compatible with the latest versions of Android and iOS. If these versions change significantly, updates may be needed.
* Network Connectivity: The platform assumes stable internet access on campus. Limited or no connectivity will affect ride coordination and parking availability updates.
* Integration with Campus Systems: The system's effectiveness relies on seamless integration with the university's digital ID verification system and parking management system. Any changes or outages in these systems could impact functionality.
* Mapping and Navigation Services: The platform depends on reliable third-party mapping services for route planning and carpool coordination. Disruptions in these services may impair the route suggestion feature.
* User Data Accuracy: It is assumed that users will provide accurate personal and vehicle information during registration. Inaccurate data may affect ride-matching and parking booking accuracy.
* Device Support: The platform is assumed to support a range of devices, including smartphones, tablets, and desktop browsers. Compatibility issues may arise with outdated or unsupported devices.

# **3. Specific Requirements**

## 3.1 Functional Requirements (FRs)

Numbered (e.g., FR1, FR2) with descriptions

## 3.2 Non-functional Requirements (NFRs)

Performance, Security, Usability, Availability, Maintainability

# **4. Use Case Model**

## 4.1 Use Case Diagram

A diagram of a diagram

AI-generated content may be incorrect.

## 4.2 List of Use Case Specifications

**1. Login With ID**

Actors: User, Admin

Description:

User login using university-issued digital ID, Admin with constant id and password

Preconditions:

* User must have a valid digital ID.

Postconditions:

* User is authenticated and redirected to the main interface.

Normal Flow:

* 1. User opens the application.
  2. System prompts for digital ID authentication.
  3. Digital ID verified via University ID System.
  4. User is granted access.

**2. Request Ride**

Actor: User

Description:

User requests to join or create a carpool ride.

Preconditions:

* User must be logged in.

Postconditions:

* Ride request is submitted and matched.

Normal Flow:

1. User selects "Request Ride."
2. System requests ride details (origin, destination, time).
3. System uses GPS to match or suggest carpools.
4. User receives ride confirmation.

**3. Check Parking Availability**

Actor: User

Description:

User views available parking slots in real-time.

Preconditions:

* User is logged in.

Postconditions:

* Available slots displayed; option to reserve shown if ride is confirmed.

Normal Flow:

1. User accesses parking availability section.
2. System fetches real-time data from Campus Parking System.
3. Display available slots.

**4. Receive Notifications**

Actor: User

Description:

System notifies user of ride updates, parking alerts, and announcements.

Preconditions:

* User has an active session or app notifications enabled.

Postconditions:

* User receives timely updates.

Normal Flow:

1. System sends notification.
2. User reads and acts if necessary.

**5. Manage User Data**

Actor: Admin

Description:

Admin manages user registration data, flags misuse, etc.

Preconditions:

* Admin is authenticated.

Postconditions:

* User data updated in database.

Normal Flow:

1. Admin logs in to dashboard.
2. Views or edits user profiles.

**6. Monitor Ride Data**

Actor: Admin

Description:

Admin reviews statistics and logs of rides and carpools.

Preconditions:

* Admin is logged in.

Postconditions:

* Reports generated or anomalies flagged.

Normal Flow:

1. Admin selects ride monitoring panel.
2. System fetches and displays ride logs.

**7. Check Parking Logs**

Actor: Admin

Description:

Admin views historical and current parking usage logs.

Preconditions:

* Admin is authenticated.

Postconditions:

* Logs reviewed or exported.

Normal Flow:

1. Admin opens parking log page.
2. System retrieves logs from database.

**8. Book Car Park**

Actor: User

Description:

User reserves a parking slot in advance upon joining or initiating a carpool ride.

Preconditions:

* User must be logged in.
* A ride request must be confirmed.

Postconditions:

* Parking slot is reserved and recorded in the system.

Normal Flow:

1. User completes a ride request.
2. System checks availability with Campus Parking System.
3. User confirms booking.
4. System updates parking database and confirms reservation.

# **5. Other Diagrams and Models**

## 5.1 ERD or Class Diagram

## 5.2 Activity or Sequence Diagram

## 5.3 Data Dictionary (optional)

# **6. Appendices**

## 6.1 Change history

## 6.2 Glossary

## 6.3 Supporting data or notes