

**Trimester March/April, 2025**

**CSE6224 SOFTWARE REQUIREMENTS ENGINEERING**

**Project Part 1**

**Topic: Campus Ride-Sharing Platform with**

**Parking System Integration**

**Software Requirements Specification**

|  |  |  |
| --- | --- | --- |
| Name | Student ID | Course |
| Chee Rui | 1211112287 | Bachelor of Computer Science |
| Teh Li Wei | 1211109581 | Bachelor of Computer Science |
| Sow Chien Yee | 1211210800 | Bachelor of Computer Science |
| Lai Zi Xuan | 1211109451 | Bachelor of Computer Science |

**Table of Contents**

[**1 Introduction 3**](#_8qhk8rbclkmo)

[1.1 Problem Statement 3](#_4v7znnun3zh4)

[1.2 Vision 3](#_o9f67vy27l3)

[1.3 Scope 4](#_wi4uzsbz4l3b)

[1.4 Purpose 4](#_5uy3u3vxj47b)

[1.5 Goals 4](#_68ac5kpjsfzr)

[1.6 References 5](#_fynhq0z0nam3)

[**2 Product Overview 5**](#_g156thw0ybsw)

[2.1 Product Perspective 5](#_clpaxk4g822q)

[2.2 Product Functions 6](#_hdiiialvyllf)

[2.3 Product Characteristics 8](#_fsoeyxwk6833)

[2.4 Limitations 9](#_9d3r0v7bqxr3)

[**3 Requirements 11**](#_odfxfrkvacet)

[3.1 Functions 11](#_jt8bb5fqpmi2)

[3.2 Performance Requirements 11](#_z6k751gx52st)

[3.3 Usability Requirements 11](#_nwvck47ltin8)

[3.4 Interface Requirements 11](#_5djepz9hbhoe)

[3.5 Logical Database Requirements 11](#_3eyo5qq74kla)

[3.6 Design Constraints 11](#_jp33usrxvqvi)

[3.7 Software System Attributes 11](#_325hrgwxbxeu)

[3.8 Supporting Information 11](#_75ck7psircmx)

[**4 Verification 11**](#_f237zn9f3707)

[4.1 Verification Approach 11](#_ltma1231bdog)

[4.2 Verification Criteria 11](#_xu7mpkwux0ej)

[**5 Appendix 11**](#_zenz8ijvhdw9)

[5.1 Assumptions and dependencies 11](#_9zq6gvnp7wol)

[5.2 Acronyms and abbreviations 12](#_en4rmp2cpbq9)

[5.3 Glossary 12](#_9qn5nlgdapc)

# 

# 1 Introduction

## Problem Statement

The Multimedia University (MMU) Cyberjaya campus often faces issues with limited parking availability and the lack of coordinated transportation options for students or staff. Parking spots are hard to find due to absence of a real-time monitoring system or poor carpool coordination. Additionally there is no centralized platform that enables trusted ride-sharing among campus members while ensuring the security and legitimacy of users through digital ID verification. As a result, campus members waste time on searching for parking, contribute to environmental pollution and experience inefficient travel around campus.

## Vision

To create a secure, user-friendly, and efficient campus ride-sharing platform integrated with a real-time parking management system. It aims to provide a secure, user-friendly, efficient and useful system that enables students or staff to coordinate carpools, promotes sustainable transportation and helps in reducing parking demands. By integrating with the university's digital ID verification system and real-time parking status checking, we aim to build a trusted, eco-friendly, and smarter mobility experience across campus.

## 

## 1.3 Scope

**The system will:**

* Allow university members to verify their identity using their digital student or staff ID during registration.
* Enable users to offer and request rides based on time, destination, and availability.
* Automatically match riders and drivers using customizable filters.
* Display real-time parking availability across campus.
* Indicate which parking spaces are currently occupied by verified users.
* Obtain users' locations (with permission) for more accurate matching and parking coordination.
* Allow users to claim and unclaim parking spaces to maintain parking legitimacy.

**The system will not:**

* Provide rides to individuals outside the university community.
* Handle financial transactions or facilitate payments for rides. ( have to decide?  possible?) *(I mean its possible, but like student carpooling around campus should be fixed, its not a big place after all)*

## 1.4 Purpose

The purpose of the Campus Ride-Sharing and Parking Management System is to address the transportation and parking challenges at MMU Cyberjaya by providing a secure, integrated system for campus ride-sharing and real-time parking management. It aims to facilitate trusted carpool coordination through integration with student ID , reduce time spent searching for parking by showing available parking spots, lower environmental impact, and promote more efficient and sustainable campus mobility for students and staff.

On the other hand, this document defines the software requirements for the Campus Ride-Sharing and Parking Management System at Multimedia University Cyberjaya. It’s purpose is to provide a clear and detailed description of the system's functionalities, constraints, and goals of the system. It serves as a reference for the development team, project stakeholders, and university administration to ensure the system is designed to meet user needs and institutional goals. This document also provides the foundation for future system design, development, and validation. It aims to ensure all parties share a common understanding of the system's expected behaviour, features, limitations and evaluation.

## 1.5 Goals

* Reduce campus parking congestion through coordinated ride-sharing.
* Provide secure and exclusive access to university members by using digital ID authentication.
* Offer real-time parking availability data to optimize parking usage.
* Encourage environmentally friendly commuting habits.

# 2 Product Overview

The Campus Ride-Sharing and Parking System is a module integrated within the existing MMU Mobile campus app. It provides additional functions to the app by coordinating secure, safe ride-sharing and providing real time parking for end users.

This integration supports reducing parking problems and promotes shared mobility while complementing other digital campus services. The system communicates with the campus app through APIs, sharing user data with the campus parking system, student authentication services with student id, and real-time mapping tools. It synchronizes with backend services for ride-matching and parking space management. These interfaces ensure seamless functionality within the larger campus infrastructure.

Accessible through mobile platforms, it offers a centralized solution to reduce parking congestion, lower carbon emissions, and enhance overall campus mobility.

## 2.1 Product Perspective

A screen shot of a black screen

AI-generated content may be incorrect.

The block diagram above shows the campus ride-sharing and Parking App is part of a campus mobile app that includes services like campus shuttles, public transport integration, and vehicle access control. It connects to the campus authentication system for secure login and to APIs for real-time parking data and ride-matching. The app also interfaces with GPS to track user location and space availability. Data flows between the app, cloud services, and external systems to provide a seamless user experience. These connections ensure efficient coordination between parking, carpooling, and campus-wide transport services.

The campus ride-sharing system with parking system interfaces with campus authentication system for secure user authentication and integrates with parking and mobility APIs for real-time data exchange. Its mobile friendly user interface allows users to book rides, view parking availability, and receive notifications. The app connects to hardware such as GPS modules and smartphones for accurate tracking and monitoring. It also works alongside software components like mapping APIs and the university’s backend systems. Communication relies on secure HTTPS protocols, with real-time updates delivered via pushing notifications. The app is optimized for typical mobile device memory limits and uses cloud storage to reduce local memory usage. Operationally, it includes an admin dashboard for monitoring and managing activity. Site adaptation allows for custom configurations based on specific campus policies, such as reserved zones or academic schedules. Finally, the app integrates with services like email, SMS, and potentially payment gateways to enhance functionality and user experience.

## 2.2 Product Functions

**User Capabilities:**

* Verify identity using StudentID and password
* Open and zoom into the school map
* View parking spaces and car plate details
* Report illegitimate parking
* Override or empty a parking space
* Book a ride to a faculty member

**Admin Capabilities:**

* All User Functions
* Log in using admin ID and password
* View reported parking spaces
* View car owner details

## 2.3 Product Characteristics

View ISO docs 9.6.6

## 

## 2.4 Limitations

The following table outlines the limitations that may impact the design, development, and deployment of the Campus Ride-Sharing Platform with Parking System Integration:

|  |  |
| --- | --- |
| **Limitation Category** | **Description** |
| **Regulatory Requirements and Policies** | Only registered university students and staff can use the platform. User data must comply with the university’s privacy policies and external data protection regulations (e.g., PDPA). |
| **Hardware Limitations** | Real-time parking information depends on the university’s parking management infrastructure, which may have latency, limited coverage, or data inaccuracies. |
| **Interfaces to Other Applications** | Integration with campus digital ID verification and parking management systems is required. External system limitations (e.g., API restrictions) may affect platform functionality. |
| **Parallel Operation** | The platform must operate alongside existing manual parking procedures during the transition phase. |
| **Audit Functions** | User login and ride-sharing activities must be logged, but real-time auditing is not mandatory. |
| **Control Functions** | The platform provides parking recommendations but does not directly control external devices like parking gates. |
| **Higher-Order Language Requirements** | No restrictions on programming languages; system interfaces should follow common standards (e.g., REST APIs). |
| **Signal Handshake Protocols** | Standard HTTPS communication and OAuth authentication are sufficient; no special signal protocols are required. |
| **Quality Requirements** | The platform must ensure at least 99% uptime during semesters. Parking data refresh intervals should not exceed 2 minutes. |
| **Criticality of the Application** | The system is important for convenience but is not safety-critical; failures should degrade gracefully without endangering users. |
| **Safety and Security Considerations** | Authentication must be secure. Personal data must be encrypted both during transmission and at rest. |
| **Physical/Mental Considerations** | The user interface should include accessibility features (e.g., support simple navigation) to assist users with disabilities. |
| **Limitations Sourced from Other Systems** | Accuracy and timeliness of parking availability and ID verification depend on external university systems, which may introduce occasional delays or errors. |

# 3 Requirements

## 3.1 Functions

## 3.2 Performance Requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Priority |
| REQ\_P001 | The average response time of searching for a ride is 3 milliseconds | HIGH |
| REQ\_P002 | The system shall support at least 500 concurrent users without significant performance degradation. | HIGH |
| REQ\_P003 | The average response time of searching for an available parking slot is less than 5 milliseconds | HIGH |
| REQ\_P004 | The average process time of updating the parking slot availability is less than 10 milliseconds | HIGH |
| REQ\_P005 | The average time for the process of user log in operation is less than 5 milliseconds | HIGH |

The requirements above show how well the system must perform under various conditions. The app must perform efficiently under both regular and peak usage conditions. It should provide real-time updates for ride-matching and parking availability, responding to user actions within 2 seconds and maintaining at least 99.9% uptime. The system backend should support high concurrency, handling over 1,000 API requests per minute without lag.

## 3.3 Usability Requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Priority |
| REQ\_U001 | The app should use clear, campus-branded visuals and icons for navigation and function clarity. | HIGH |
| REQ\_U002 | The interface shall be accessible and usable on common screen sizes (phones and tablets) with touch-optimized controls. | HIGH |
| REQ\_U003 | Feedback (e.g., confirmation or error messages) shall be provided to the user within 1 second of completing any major action. | MEDIUM |

The usability requirements show how easy it is for users to interact with the app. The application must be user-friendly, enabling first-time users to complete essential tasks like booking a ride or reserving a parking spot within 5 minutes. It should offer a clean, campus-branded interface compatible with mobile devices and accessible for users with disabilities. Clear visual cues and immediate feedback must guide users through all actions, enhancing ease of use and satisfaction.

## 3.4 Interface Requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Priority |
| REQ\_I001 | Users shall log in via the campus Single Sign-On (SSO) system. | HIGH |
| REQ\_I002 | The app shall interface with GPS sensors in smartphones to determine user location. | HIGH |
| REQ\_I003 | It shall interact with campus APIs for student data, parking rules, and access control. | MEDIUM |
| REQ\_I004 | The app shall integrate with mapping APIs (e.g., Google Maps) for route planning and navigation. | HIGH |
| REQ\_I005 | The system shall support push notifications for ride updates, reminders, and alerts. | MEDIUM |
| REQ\_I006 | All data transmissions shall use HTTPS protocols. | HIGH |

## These requirements specify how the app interacts with users, hardware and other systems. The app should integrate seamlessly with campus systems and external services. Users will interact through a mobile interface connected to GPS, parking sensors, and mapping APIs, while authentication will be handled via campus SSO. All data exchange should be secure, using HTTPS, and the system should support real-time communication through push notifications and cloud-based APIs.

## 3.5 Logical Database Requirements

## 3.6 Design Constraints

## 3.7 Software System Attributes

## 3.8 Supporting Information

# 4 Use Case

4.1 Use Case Diagram

4.2 Detailed Use Case Descriptions

# 6 Appendix

## 5.1 Assumptions and dependencies

## 5.2 Acronyms and abbreviations

## 5.3 Glossary