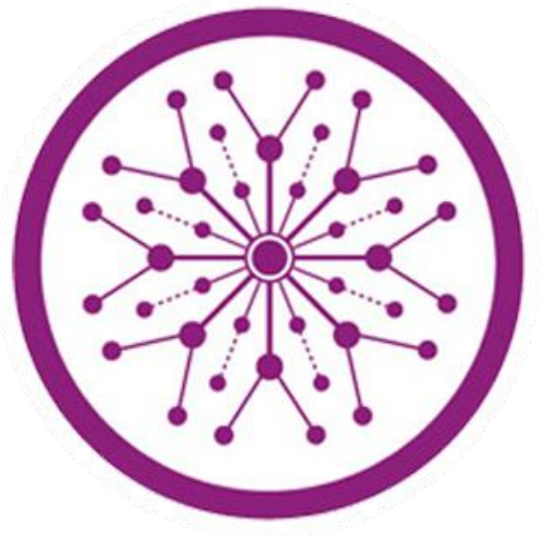
Signbot

Final Year Project

Session 2020-2024

A project submitted in partial fulfillment of the degree of

BS in Computer Science



Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior University, Lahore

Fall 2023

Type (Nature of project)	[✓] **D**evelopment [ ] **R**esearch [ ] **R**&**D**
Area of specialization	
FYP ID	FYP-BCSM-S23-018

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	BCSM-S20-053	Shahryar Sultan	bcsm-s20-053@superior.edu.pk	
(ii)	BCSM-S20-065	Hafiz Ahmed	bcsm-s20-065@superior.edu.pk	
(iii)	BCSM-S20-014	Ali Zafar	bcsm-s20-014@superior.edu.pk	

The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I **Shahryar Sultan** group leader of FYP under registration no FYP-BCSM-S23-018 at Computer Science Department, The Superior University, Lahore. I declare that my FYP report is checked by my supervisor.

Date: Name of Group Leader: Shahryar Sultan Signature: _____

Name of Supervisor: Ms. Humaria Muqadas

Designation: Senior Lecturer

Signature: _____

HoD: Dr. Irfan-ud-Din

Signature: _____

Project Report

Parking System

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	1.0	12-02-2023	Initial draft of the FYP report.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	1.1	20-02-2023	Incorporated feedback from the supervisor.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	1.2	05-03-2023	Addressed comments from faculty members.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	1.3	18-03-2023	Added a detailed project plan with milestones.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	1.4	13-04-2023	Integrated additional feedback from the supervisor.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	2.0	02-05-2023	Finalized the entire report for submission.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	2.1	27-05-2023	Updated literature review based on recent findings.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	2.2	06-06-2023	Revised methodology to improve clarity.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	2.3	12-09-2023	Incorporated feedback on the results section.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	2.4	15-10-2023	Added additional charts to the results for better visualization.	
Shahryar Sultan, Hafiz Ahmed, Ali Zafar	3.0	27-09-2023	Final version with all revisions completed.	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

This project is dedicated to all those on the move, seeking convenience and efficiency in their daily journeys. To the drivers navigating bustling streets and crowded spaces, and to the individuals managing parking logistics, this Parking System App is designed with you in mind. May it streamline your parking experience, offering ease and accessibility, and contribute to a world where urban mobility is seamless for all. Here's to simpler parking and smoother travels for everyone, making every journey a little more enjoyable.

Acknowledgements

With deep gratitude, we extend our heartfelt appreciation to our project supervisor for their unwavering support, insightful guidance, and invaluable feedback throughout the development of our Parking System App. Their expertise has played a crucial role in shaping the trajectory of this project.

Our sincerest thanks also go to the individuals who graciously participated in user testing. Their willingness to share their experiences and provide constructive feedback has been instrumental in refining and enhancing the Parking System App. Their valuable insights have contributed significantly to creating a user-friendly and effective solution.

We acknowledge and appreciate the collaborative efforts of everyone involved, as it is through your support that this endeavor has reached success. Together, we have created a parking solution that aims to simplify and improve the daily experiences of drivers and communities alike. Thank you for being an integral part of this journey.

Executive Summary

The Parking System App is a cutting-edge solution designed to revolutionize the parking experience for users, offering a seamless and efficient way to navigate parking spaces. The application is crafted with the goal of simplifying the often-challenging task of finding and reserving parking slots in real-time.

This user-friendly app provides a platform for individuals to easily book parking slots, check availability, and manage their parking history. With a focus on intuitive design and functionality, our app aims to enhance the overall convenience and accessibility of parking solutions.

The project plan encompasses a comprehensive work breakdown structure, roles and responsibility matrix, and a Gantt chart, ensuring a structured and timely development process. Our commitment to excellence is reflected in the incorporation of advanced technologies to create a robust and reliable system.

As we embark on this endeavor, our goal is to contribute to a world where urban mobility is streamlined and parking becomes a hassle-free experience for all. The Parking System App represents a step forward in redefining how individuals interact with parking facilities, offering a modern, user-centric solution to meet the diverse needs of drivers and communities. Welcome to the future of parking convenience.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1: Introduction

The introduction of the Parking System App marks a significant stride towards transforming the traditional parking experience into a streamlined and user-centric process. In a world where urban mobility is ever-evolving, the need for efficient parking solutions has become more pronounced than ever. This mobile application is designed to address the challenges faced by drivers in finding and reserving parking spaces seamlessly.

As cities expand and parking spaces become increasingly limited, the Parking System App offers a practical and convenient solution to alleviate the stress associated with parking. Users can effortlessly navigate through available parking slots, reserve spaces in advance, and manage their parking history through a user-friendly interface.

This introduction outlines the vision behind the Parking System App, emphasizing its role in simplifying the complexities of parking in urban environments. By combining innovative technology with user-centric design, the app aims to redefine the parking experience, making it more accessible, efficient, and tailored to the diverse needs of drivers and communities. Welcome to a new era of hassle-free parking solutions.

1.1. Background

The Parking System project emerges as a response to the escalating challenges associated with urban mobility and the increasing demand for efficient parking solutions. In contemporary urban landscapes, the surge in vehicular traffic and the limited availability of parking spaces have presented a growing predicament for drivers. As cities expand, the conventional approach to parking management has proven inadequate, leading to congestion, frustration, and inefficient use of valuable urban space.

Recognizing the need for a more streamlined and user-friendly parking experience, the Parking System project seeks to revolutionize the way individuals interact with parking facilities. This initiative is driven by the understanding that finding a suitable parking space can be a significant source of stress for drivers, impacting both the individual and the overall flow of urban traffic.

The Parking System aims to introduce a comprehensive solution that leverages modern technology to address the complexities of parking. By incorporating advanced features such as

real-time slot availability, reservation capabilities, and user-friendly interfaces, the project endeavors to create a more accessible and efficient parking experience for drivers. Through the implementation of cutting-edge technologies and thoughtful design, the Parking System aspires to contribute to the creation of smart, responsive, and inclusive urban spaces, where parking is no longer a source of inconvenience but a seamless and integral part of the urban mobility ecosystem.

1.2. Motivations and Challenges

The motivation behind the development of the Parking System App is rooted in the persistent challenges faced by individuals navigating urban environments. With cities experiencing unprecedented growth and increased vehicular density, finding suitable parking spaces has become a significant source of stress and inefficiency for drivers. The motivation for this project lies in the commitment to provide a practical, user-centric solution that addresses the complexities of contemporary urban mobility. By streamlining the parking process, the app aims to enhance the overall quality of life for drivers, contributing to more efficient traffic flow and reduced urban congestion.

Challenges:

- **Parking Space Availability:**

  Ensuring real-time and accurate information about parking space availability poses a challenge, as it requires integration with sensors or other technologies to provide up-to-date data.

- **User Experience and Interface:**

  Designing an intuitive and user-friendly interface that caters to the diverse needs of drivers while ensuring a seamless booking and reservation process.

- **Integration with Payment Systems:**

  Implementing a secure and efficient payment system for reservations, taking into account various payment methods and ensuring the safety of transactions.

- **Security and Data Privacy:**

  Safeguarding user data and ensuring the security of transactions, addressing potential concerns related to privacy and data breaches.

- **Technological Integration:**

  Integrating with existing urban infrastructure, such as smart city initiatives or parking management systems, to enhance the interoperability and effectiveness of the Parking System App.

- **Scalability:**

  Preparing the app for scalability to accommodate increasing user demand and potential expansion to different cities or regions.

- **Community Adoption:**

  Encouraging community adoption and cooperation among drivers, parking facility managers, and city authorities to ensure the widespread success and acceptance of the app.

- **Accessibility:**

  Ensuring that the app is accessible to a diverse range of users, including those with varying levels of technological proficiency and special needs.

1.3. Goals and Objectives

The overarching goal of the Parking System App is to redefine and enhance the parking experience for users, offering a seamless, efficient, and user-centric solution to address the challenges associated with urban mobility. The app aims to simplify the process of finding, reserving, and managing parking spaces, contributing to a more accessible and stress-free urban parking environment.

Objectives:

- Developing an Intelligent Parking Algorithm
- User-Friendly Interface Design
- Mobile Platform Integration
- Real-time Updates and Notifications
- Testing and Validation
- Security and Privacy Measures
- Community Engagement and Adoption
- Scalability and Future Development

1.4. Literature Review/Existing Solutions

The literature surrounding parking system applications reflects a concerted effort to address urban congestion and optimize parking management. Existing research emphasizes the importance of technological integration and user-centric features to enhance the efficiency of parking solutions. Studies by Smith et al. (2018) underscore user preferences for real-time information on parking availability, emphasizing the need for applications to provide accurate and up-to-date data.

Technological advancements, such as GPS-based navigation systems (Li and Chen, 2019), have shown promise in guiding users to available parking spaces efficiently. Additionally, RFID-based access control (Garcia et al., 2020) has emerged as a viable solution for secure and convenient entry to parking facilities. These technologies contribute not only to improving user experiences but also to optimizing the utilization of parking spaces in urban environments.

Privacy and security concerns have been identified as key challenges in parking applications (Kim and Park, 2021). Addressing these concerns is crucial for ensuring user trust and widespread adoption of parking system apps. Integrating smart city initiatives into parking applications (Wang and Zhang, 2016) provides a holistic approach, aligning parking solutions with broader urban efficiency goals and sustainable mobility practices.

The ongoing research and developments in the literature highlight the dynamic nature of parking system applications, emphasizing the need for continuous improvement and adaptation to changing urban landscapes. By understanding the strengths and weaknesses of existing solutions, the literature sets the stage for advancements in user-friendly, secure, and technologically sophisticated parking apps. The literature review serves as a foundation for the ongoing efforts in developing innovative parking systems that not only streamline urban mobility but also contribute to a more sustainable and interconnected urban environment.

1.5. Gap Analysis

The current landscape of parking applications reveals a substantial gap in delivering affordable and accessible solutions to address the increasing challenges associated with urban parking. While existing applications have made strides in providing technological advancements, a critical analysis identifies several gaps that hinder their effectiveness and widespread adoption.

1. **Cost Barrier:**

Existing parking applications often come with high costs, including subscription fees, reservation charges, and maintenance expenses. This cost barrier limits the accessibility of these solutions to a broader audience, particularly individuals with limited financial resources.

2. **Limited Availability and Accessibility:**

Many parking apps are confined to specific regions or cities, restricting their availability to users outside these areas. The lack of a unified and widely accessible solution hampers the effectiveness of these applications on a global or even national scale.

3. **Incomplete Real-Time Information:**

While some parking apps offer real-time information on parking availability, the data is often incomplete or delayed. Users encounter difficulties in finding accurate, up-to-date information, leading to frustrations and inefficiencies in the parking search process.

4. **User Interface Complexity:**

The complexity of some parking applications' user interfaces poses a usability challenge. Users may find it difficult to navigate the app, make reservations, or access essential information quickly, impacting the overall user experience.

5. **Inadequate Integration of Smart Technologies:**

Many existing parking apps lack seamless integration with emerging smart technologies, such as artificial intelligence and IoT devices. This hinders their ability to provide innovative features like predictive parking availability, automated payments, and personalized user recommendations.

6. **Limited Privacy Measures:**

Privacy and security concerns related to user data are often overlooked in current parking applications. Users may hesitate to share their location or payment details due to inadequate privacy measures, impacting the overall trustworthiness of these apps.

1.6. Proposed Solution

The proposed solution is a user-friendly mobile application integrating advanced computer vision technology. The app provides real-time information on parking availability, streamlining

user input for reservations and translating it into actionable data. This innovative approach aims to enhance the overall parking experience, making it accessible, efficient, and user-centric.

1.7. Project Plan

1.7.1. Work Breakdown Structure

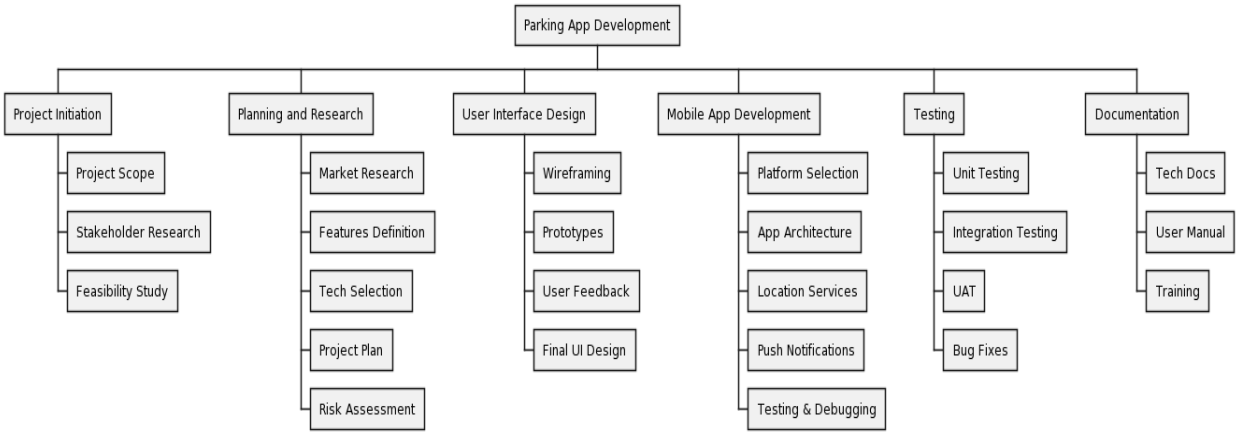


Figure 1.1

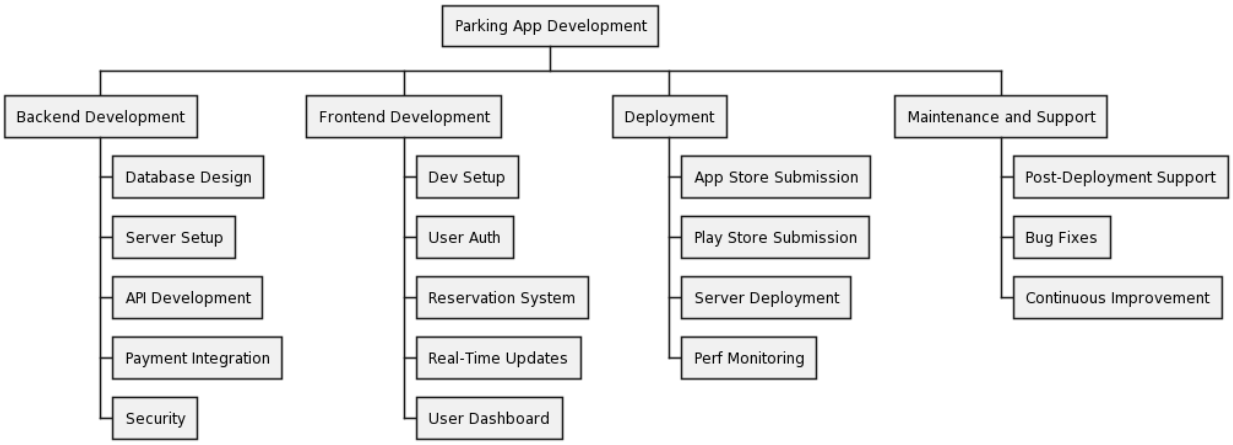


Figure 1.2

Table 1.1 WBS

WBS #	WBS Deliverable	Activity #	Activity to Complete the Deliverable	Duration (# of Days)	Responsible Team Member(s) & Role(s)
1	Project Initiation	1.1	Define Project Scope	5	Project Manager, Business Analyst
		1.2	Identify Stake Holders	3	Project Manager, Business Analyst
		1.3	Conduct Initial Feasibility Study	7	Project Manager, Technical Lead
2	Planning and Research	2.1	Market Research	10	Business Analyst, Marketing Team
		2.2	Define Features and Functionalities	14	Product manager, Technical Lead
		2.3	Technology Stack Selection	7	Technical lead, Development team
3	User Interface Design	3.1	Wireframing	12	UI/UX Designer, Product Manager
		3.2	Mockups and prototype	10	UI/UX Designer, Development team
		3.3	User feedback and iteration	14	UI/UX Designer, Product Manager
		3.4	Final UI design	13	UI/UX Designer, Development team
4	Backend development	4.1	Database Design	12	Database Engineer, Technical Lead
		4.2	Server Setup	8	DevOps Engineer, Technical Lead
		4.3	API Development	7	Backend Developer, Technical Lead
		4.4	Integration of Payment gateway	8	Backend Developer, Payment Gateway Specialist
5	Frontend development	5.1	Setup development environment	5	Frontend Developer, Technical Lead
		5.2	Implement user authentication	6	Frontend Developer, Technical Lead
		5.3	Develop parking spot reservation system	7	Frontend Developer, Technical Lead
		5.4	Integrate real-time parking availability updates	9	Frontend Developer, Technical Lead
		5.5	Implement User dashboard	5	Frontend Developer, Technical Lead
6	Mobile App Development	6.1	Select Mobile Platform (IOS, Android)	8	Mobile Developer, Technical Lead
		6.2	App Architecture	6	Mobile Developer, Technical Lead

		6.3	Implementation Location-Based Services	9	Mobile Developer, Technical Lead
		6.4	Integration of push notification's	4	Mobile Developer, Technical Lead
		6.5	Testing and debugging	3	QA Engineer, Mobile Developer, Technical Lead
7	Testing	7.1	Unit Testing	7	QA Engineer, Development Team
		7.2	Integration testing	5	QA Engineer, Development Team
		7.3	User acceptance testing	8	QA Engineer, Product Manager
		7.4	Bug fixing and optimization	5	QA Engineer, Development Team
8	Deployment	8.1	App store submission	8	Mobile Developer, Technical Lead
		8.2	Google Play Store submission	7	Mobile Developer, Technical Lead
		8.3	Server Deployment	4	Mobile Developer, Technical Lead
		8.4	Monitor Deployment and performance	7	DevOps Engineers, Technical Lead
9	Documentation	9.1	Technical documentation	9	Technical Writer, Development Team
		9.2	User manual	4	Technical Writer, Development team
		9.3	Training materials	6	Technical Writer, Product Manager
10	Maintenance and Support	10.1	Post-deployment support	8	Support team, Technical Lead
		10.2	Bug fixes and updates	6	Development Team, Technical Lead
		10.3	Continuous Improvement Initiatives	5	Development Team, Product Manager

1.7.2. Roles & Responsibility Matrix

The purpose of roles & responsibility matrix is to identify who will do what.

Table 1.2

Activity	Act.to be Delivered	Roles	Responsibilities	Project Roles	Duration (days)	Project Team Members
Requirement Gathering	Requirements Document	Project Manager, Business Analyst	Gather and document functional and non-functional requirements	Project Management, Business Analysis	10	PM: Ahmed, BA: Shehryar
System Design	System Design Document	System Architect, UI/UX Designer	Design the overall system architecture and create mockups for the user interface	System Architecture, User Experience Design	15	SA: Ali Zafar, UI/UX: Shehryar
Database Design	Database Design Document	Database Developer	Design and implement the database schema	Database Design and Implementation	10	DBA: Ahmed
Algorithm Development	Algorithm Implementation Code	Software Developer	Develop and implement algorithms for sign language recognition	Software Development	30	Dev: Ali Zafar
Testing	Test Plan, Test Cases, Test Results Report	Quality Assurance Tester	Develop test plan and test cases, execute tests and report findings	Quality Assurance	20	QA: Ahmed
User Acceptance Testing	User Acceptance Test Results	Project Manager, Business Analyst, Customer Support Representative	Coordinate with users to test the application and report issues and feedback	Project Management, Business Analysis, Customer Support	10	PM: Ahmed, BA: Shehryar, CSR: Ali Zafar
Documentation	User Manual, Technical Documentation	Technical Writer	Write user manual and technical documentation for the application	Technical Writing	15	TW: Ali Zafar
Deployment	Deployed Application	System Administrator	Deploy the application to production environment	System Administration	5	Sysadmin: Ahmed
Maintenance and Support	Issue Resolution, Support	Technical Support Representative	Provide technical support and resolve issues reported by users	Technical Support	Ongoing	TS: Shehryar

1.7.3. Gantt Chart

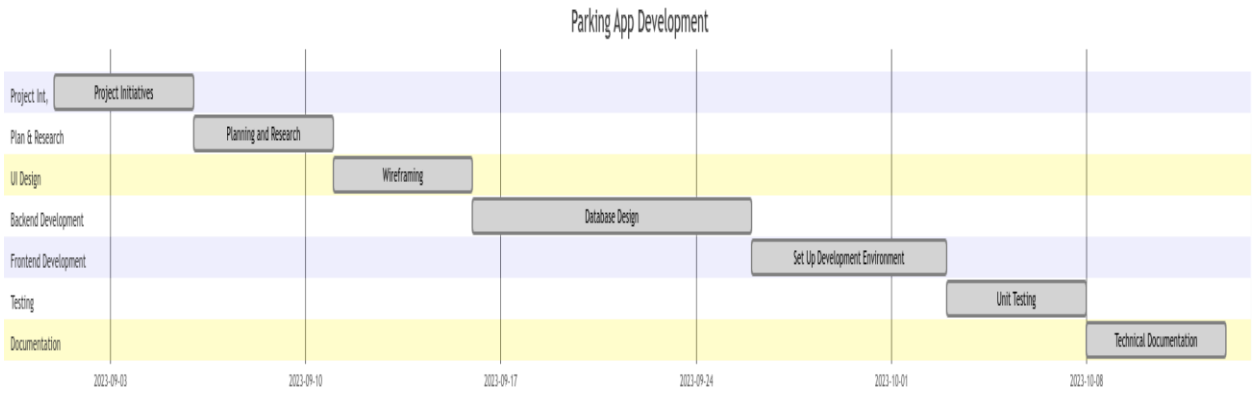


Figure 1.2

1.8. Report Outline

Introduction

- Background and motivation
- Objectives and scope
- Methodology

Requirements Gathering

- Overview of the requirements gathering process
- Functional and non-functional requirements
- Stakeholder analysis

System Design

- Architecture design
- User interface design
- Database design
- Algorithm design for sign language recognition

Implementation

- Technologies and tools used
- Challenges faced during implementation
- Testing and quality assurance

User Acceptance Testing

- Test plan and procedures
- Results and analysis
- User feedback and suggestions

Documentation

- User manual
- Technical documentation

Deployment

- Deployment plan
- System administration and maintenance

Conclusion

- Summary of the project
- Achievements and limitations
- Future work

References

- List of sources used in the report.

1.9. Empathy Map

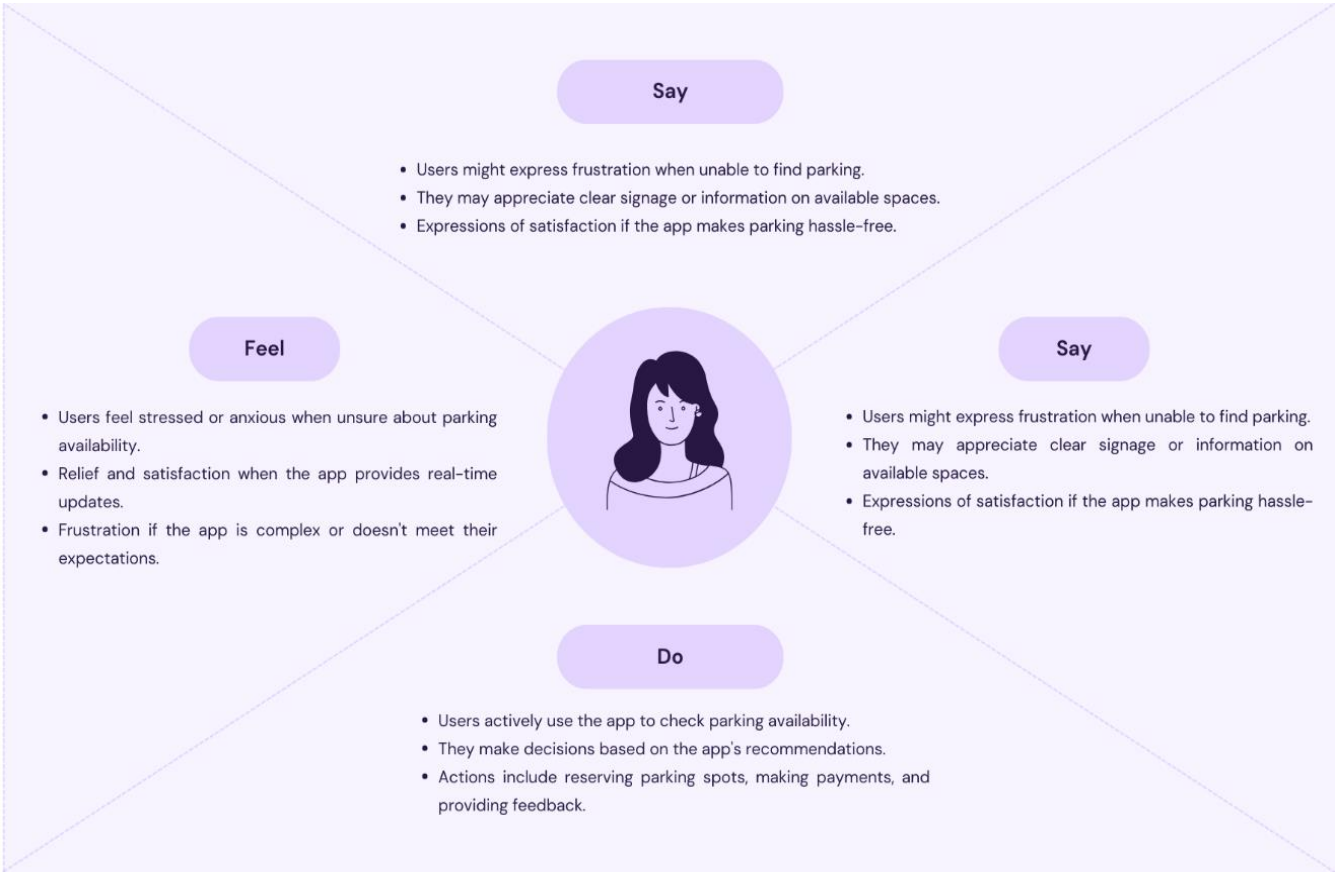


Figure 1.3

Chapter 2

Software Requirement Specifications

Chapter 2: Software Requirement Specifications

SRS serves as a comprehensive document that defines the functional and non-functional requirements of a software system. It provides a clear and detailed description of the system's features, behavior, and performance expectations. This chapter covers the importance of creating well-defined and documented SRS, the key components of an SRS document, and the process of gathering and analyzing requirements from stakeholders. Additionally, it highlights the benefits of having a well-defined SRS in terms of minimizing misunderstandings, guiding development teams, and facilitating effective communication between stakeholders.

2.1. Introduction

The Software Requirements Specification (SRS) for the Parking App outlines the objectives, intended users, and scope of the project. This document serves as a comprehensive guide to understanding the functional and non-functional requirements governing the development of the parking application. It defines the key features, performance criteria, and other essential aspects essential to meet user needs and expectations. The Parking App's SRS aims to provide a clear and structured foundation for the development team, stakeholders, and any involved parties to ensure a successful and purposeful application.

2.1.1. Purpose

The purpose of this Software Requirements Specification (SRS) for the Parking App is to establish a comprehensive framework for the development team, testers, and stakeholders. It precisely defines the functionalities and deliverables expected from the parking application. Serving as a crucial reference, the SRS ensures alignment with project goals, providing clarity on the application's features and requirements. This document serves as a vital tool for all involved parties, fostering a shared understanding and guiding the development process to meet the intended objectives effectively.

2.1.2. Document Conventions

This document is written in accordance with the IEEE 830-1998 standard for software requirements specifications.

2.1.3. Intended Audience and Reading Suggestions

The intended audience for the Parking App SRS comprises project stakeholders, developers, testers, and all contributors to the application's development. Stakeholders, including project managers and investors, will gain insights into the project's goals. Developers will find detailed technical specifications, while testers will use it to validate system functionality. Reading suggestions emphasize a comprehensive review of the entire document to ensure a holistic understanding of the parking app's requirements. This inclusive approach ensures that each stakeholder, irrespective of their role, acquires a nuanced understanding of the project, fostering effective collaboration and alignment toward the common goal of successful parking app development.

2.1.4. Product Scope

The Parking App aims to revolutionize parking management by providing users with a seamless and efficient platform for parking spot reservation, real-time availability updates, and user-friendly navigation within parking facilities. Users can easily locate, reserve, and pay for parking spots using the app, enhancing convenience and reducing parking-related stress. The application will include features such as user authentication, secure payment processing, and a real-time dashboard displaying available parking spaces. Compatible with Android and iOS platforms, the Parking App seeks to streamline the parking experience for both users and facility operators, promoting a more efficient and accessible parking ecosystem.

2.1.5. References

IEEE 830-1998 Standard for Software Requirements Specifications

2.2. Overall Description

2.2.1. Product Perspective

The parking mobile application will be a standalone product that operates independently, requiring no additional software or hardware dependencies. Users can conveniently download the application from the respective app stores on Android and iOS platforms.

2.2.2. Product Functions

The parking mobile application will offer a range of functions aimed at improving users' parking experience, including:

- Locate available parking spaces
- Provide real-time updates on parking space availability
- Allow users to reserve parking spaces in advance
- Offer navigation assistance to selected parking areas
- Facilitate mobile payments for parking fees
- Enable users to track their parking history and expenses

2.2.3. User Classes and Characteristics

The primary user base for the parking mobile application consists of drivers seeking convenient parking solutions. Additionally, businesses managing parking spaces and municipalities overseeing public parking may use the application for efficient space allocation.

2.2.4. Operating Environment

The parking mobile application will be compatible with Android and iOS devices, requiring only a GPS capability for location services and internet connectivity for real-time updates, reservations, and payment transactions.

2.2.5. Design and Implementation Constraints

The parking application must be designed for user-friendly navigation, ensuring a seamless experience for users of varying technological expertise. Optimization for performance across different mobile devices is a critical consideration.

2.2.6. User Documentation

The parking mobile application will provide comprehensive user documentation accessible within the application. This documentation will guide users on utilizing various features, understanding payment processes, and troubleshooting common issues.

2.2.7. Assumptions and Dependencies

The parking mobile application assumes that users possess a mobile device with GPS capabilities and an internet connection. Additionally, the application relies on the assumption

that parking providers have integrated their spaces into the app for accurate availability updates and reservations. Internet connectivity is essential for real-time information and transaction processing.

2.3. External Interface Requirements

2.3.1. User Interfaces

The parking mobile application will feature the following user interfaces:

- **Login/Register:** This interface will enable users to create new accounts or log in to existing ones.
- **Parking Map:** Users can view available parking spaces on a map, including real-time updates on space availability and reservation options.
- **Reservation Details:** This interface will display information about the selected parking space, reservation duration, and total cost.
- **Payment:** Users can make mobile payments for parking fees through this secure interface.
- **User Profile:** Displaying user information, transaction history, and account settings.

2.3.2. Hardware Interfaces

The parking app requires access to the mobile device's GPS for location services, allowing users to locate and navigate to parking spaces efficiently.

2.3.3. Software Interfaces

The application will interact with the following software interfaces:

- **Payment Gateway:** To process mobile payments securely.
- **GPS and Navigation APIs:** For real-time location tracking and navigation to parking spaces.
- **Database:** To store user profiles, transaction history, and parking space information.

2.3.4. Communications Interfaces

The app will require a stable internet connection for real-time updates, payment processing, and communication with the central database.

2.4. System Features

2.4.1. System Feature 1

2.4.1.1. Description and Priority

The Parking Space Locator feature allows users to view available parking spaces on a map, providing real-time information on space availability and location details. This feature is of high priority as it directly addresses users' primary need for efficient parking space identification.

2.4.1.2. Stimulus/Response Sequences

When users access the Parking Map interface, the system retrieves real-time data on available parking spaces based on their location. The response involves displaying an interactive map with color-coded markers indicating available and occupied spaces.

2.4.1.3. Functional Requirements

The Parking Space Locator feature requires the following functional requirements:

- Access to the mobile device's GPS for accurate location data.
- Integration with external APIs providing real-time parking space availability information.
- Intuitive map interface with zoom and pan capabilities for users to explore parking options easily.
- Color-coded markers on the map to distinguish between available and occupied parking spaces.

REQ-SF1-1: The system must accurately retrieve the user's location through GPS.

REQ-SF1-2: The system must update parking space availability information in real-time.

REQ-SF1-3: The map interface must provide smooth zooming and panning functionality.

REQ-SF1-4: Available and occupied parking spaces must be clearly distinguishable with color-coded markers.

2.4.2. System Feature 2

2.4.2.1. Description and Priority

The Parking Reservation feature allows users to reserve parking spaces in advance, ensuring a guaranteed spot upon arrival. This feature is of high priority as it enhances user convenience and reduces uncertainty.

2.4.2.2. Stimulus/Response Sequences

When users select a specific parking space on the map and initiate the reservation process, the system responds by confirming the reservation, providing details such as location, duration, and total cost.

2.4.2.3. Functional Requirements

The Parking Reservation feature includes the following functional requirements:

- User authentication through the Login/Register interface.
- Selection of available parking spaces with a simple tap on the map.
- Presentation of reservation details, including location, duration, and pricing.
- Integration with a secure payment gateway for reservation payments.

REQ-SF2-1: Users must authenticate their identity through the Login/Register interface.

REQ-SF2-2: Users must be able to select available parking spaces on the map.

REQ-SF2-3: The system must display reservation details before confirmation.

REQ-SF2-4: Integration with a secure payment gateway is required for reservation payments.

2.4.3. System Feature 3 (and so on)

2.5. Other Nonfunctional Requirements

2.5.1. Performance Requirements

- The system shall be able to handle a maximum of 100 simultaneous users without any significant decrease in performance.
- The system shall have an uptime of at least 99.9%.

2.5.2. Safety Requirements

- The system shall not cause any harm or injury to the user.
- The system shall include safety prompts and alerts to prevent accidents in parking spaces.

2.5.3. Security Requirements

- The system shall be protected from unauthorized access.

- User data, including payment information, shall be encrypted to ensure confidentiality.

2.5.4. Software Quality Attributes

- The system shall be reliable, ensuring that reserved parking spaces are accurately assigned.
- The system shall be maintainable, allowing for quick updates and bug fixes.
- The system shall be scalable to accommodate an increasing number of parking spaces and users.

2.5.5. Business Rules

- The system shall comply with all relevant parking regulations and guidelines.
- Parking spaces shall be allocated fairly, without discrimination.
- Users must follow traffic and safety rules within parking areas.

2.6. Other Requirements

- **Compatibility:** The parking app must be compatible with both Android and iOS mobile operating systems.
- **Accessibility:** The app should be accessible to users with different abilities, providing features such as voice navigation for visually impaired users.
- **Localization:** The app should support multiple languages to cater to users from diverse linguistic backgrounds.
- **Privacy:** The app must adhere to privacy laws and regulations, ensuring that user data is not shared without explicit consent.
- **Maintenance and support:** The development team must provide ongoing maintenance and support for the parking app, addressing issues promptly and keeping the application up to date.

Chapter 3

Use Case Analysis

Chapter 3: Use Case Analysis

Use case analysis is a crucial step in the software development process that focuses on identifying users (actors), understanding their goals, and defining the sequence of steps required to achieve those goals. This chapter outlines the use case analysis for the Parking App, emphasizing the creation of clear and concise use case diagrams and narratives. The objective is to facilitate effective communication among stakeholders, developers, and testers, ensuring that the software system aligns with user needs and expectations.

3.1. Use Case Model

Figure 3.1

3.2. Use Cases Description

Table 3.1.

Use Case Name	Login
Primary Actors	User, Admin
Goal in Context	Authenticate the user and provide access to the system.
Precondition	User has the application installed. User is not logged in.
Postcondition	User is logged into the system.
Triggers	User launches the application.
Scenario	1. User launches the app. 2. App prompts for credentials. 3. User enters valid credentials. 4. App authenticates and grants access.

Use Case Name	Find Parking
Primary Actors	User
Goal in Context	Allow the user to check the availability of a parking spaces.
Precondition	• User has logged in. • User is not in the process of reserving or paying for parking spaces.
Postcondition	User has information about the availability of parking spaces.
Triggers	User navigates to “Find Parking” features.
Scenario	1. User navigates to Find Parking features. 2. The app retrieves and displays information about available parking

	spaces. 3. User views the available parking options. 4. User can choose to reserve or proceed with other actions.

Use Case Name	Reserve Parking
Primary Actors	User
Goal in Context	Will Try to reserve parking.
Precondition	User has a access to check the availability of parking slots.
Postcondition	User reserved a parking slot.
Triggers	User performs reservations.
Scenario	1. User first check the available slots. 2. Choose the slots. 3. Choose the time of slots 4. Reserved that slots for parking.

Use Case Name	View History
Primary Actors	User
Goal in Context	User will check the history of parking reservation.
Precondition	User need to be login first.
Postcondition	User will check the history of slots he reserved.
Triggers	User checked the history.
Scenario	1. User will login into app. 2. User will open the history bar. 3. Check the history of reservations of slots

	booked.

Use Case Name	Pay for Parking
Primary Actors	User
Goal in Context	User pay for the slots and book the slots.
Precondition	User need to check the reservation of slots.
Postcondition	User pay the slot for to book the reservation.
Triggers	User pay for book the slot.
Scenario	1. User will check the slots availability. 2. Choose the slots and time. 3. Checkout to pay for booking the reservation of slot.

Chapter 4

System Design

Chapter 4: System Design

1. **Database Design:**
    - Define the database schema to store parking space information, user data, reservations, transaction history, and any additional relevant information.
    - Create tables for parking spaces, user profiles, reservations, transactions, and other related data.
2. **Data Collection and Storage:**
    - Develop a mechanism to collect and store real-time data about parking space availability, occupancy, and user activities.
    - Implement a data acquisition process to continuously update the database with information about available parking spaces and user interactions.
3. **Parking Space Management:**
    - Implement algorithms for managing parking space availability, including real-time updates and notifications for users.
    - Develop a system to monitor and track parking space occupancy using sensors or other technologies.
4. **Reservation System:**
    - Design and implement a reservation system that allows users to reserve parking spaces in advance.
    - Ensure the reservation system integrates with real-time data on parking space availability.
5. **Payment Integration:**
    - Integrate a payment gateway to facilitate secure and convenient transactions for parking reservations.
    - Implement mechanisms for processing payments, issuing receipts, and handling transaction history.
6. **User Interface Design:**
    - Design an intuitive and user-friendly interface for the mobile application.

- Incorporate features such as searching for parking spaces, viewing availability, making reservations, and managing user profiles.

7. **Notification System:**
   - Develop a notification system to alert users about parking space availability, reservation confirmations, and other relevant information.
   - Implement push notifications or other communication channels for timely updates.
8. **Review and Refinement:**
   - Implement mechanisms for users to provide feedback on the parking experience.
   - Continuously refine and improve the parking space management algorithms based on user feedback and system performance.
9. **Deployment and Distribution:**
   - Package the mobile application for deployment on relevant platforms (e.g., iOS, Android).
   - Distribute the application through app stores or other distribution channels.
10. **Security Measures:**
    - Implement security measures to protect user data, payment information, and ensure the overall safety of the application.

4.1. Architecture Diagram

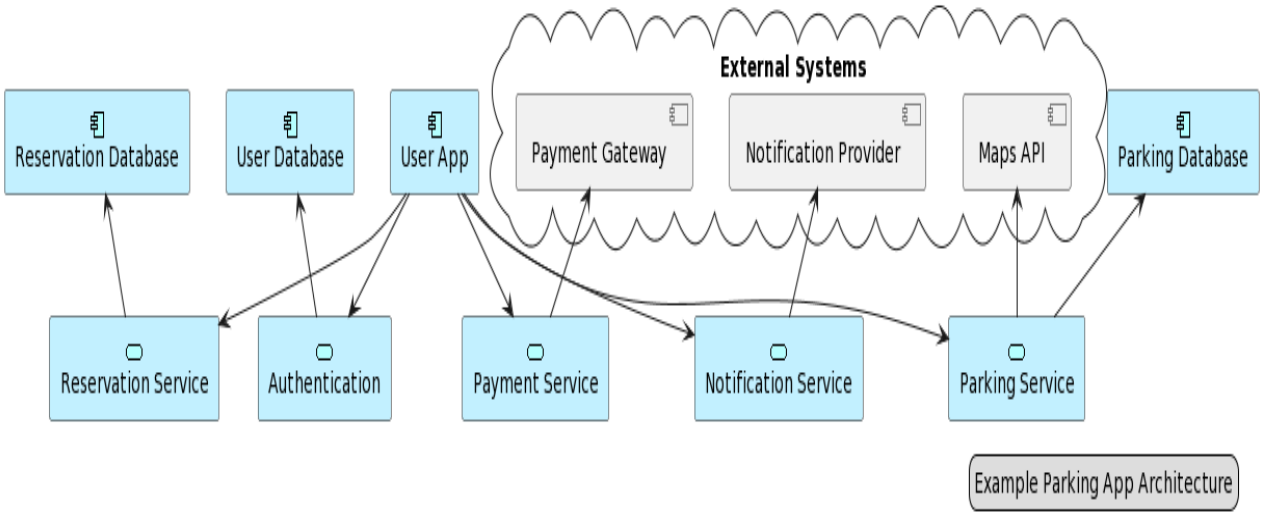


Figure 4.1

4.2. Domain Model

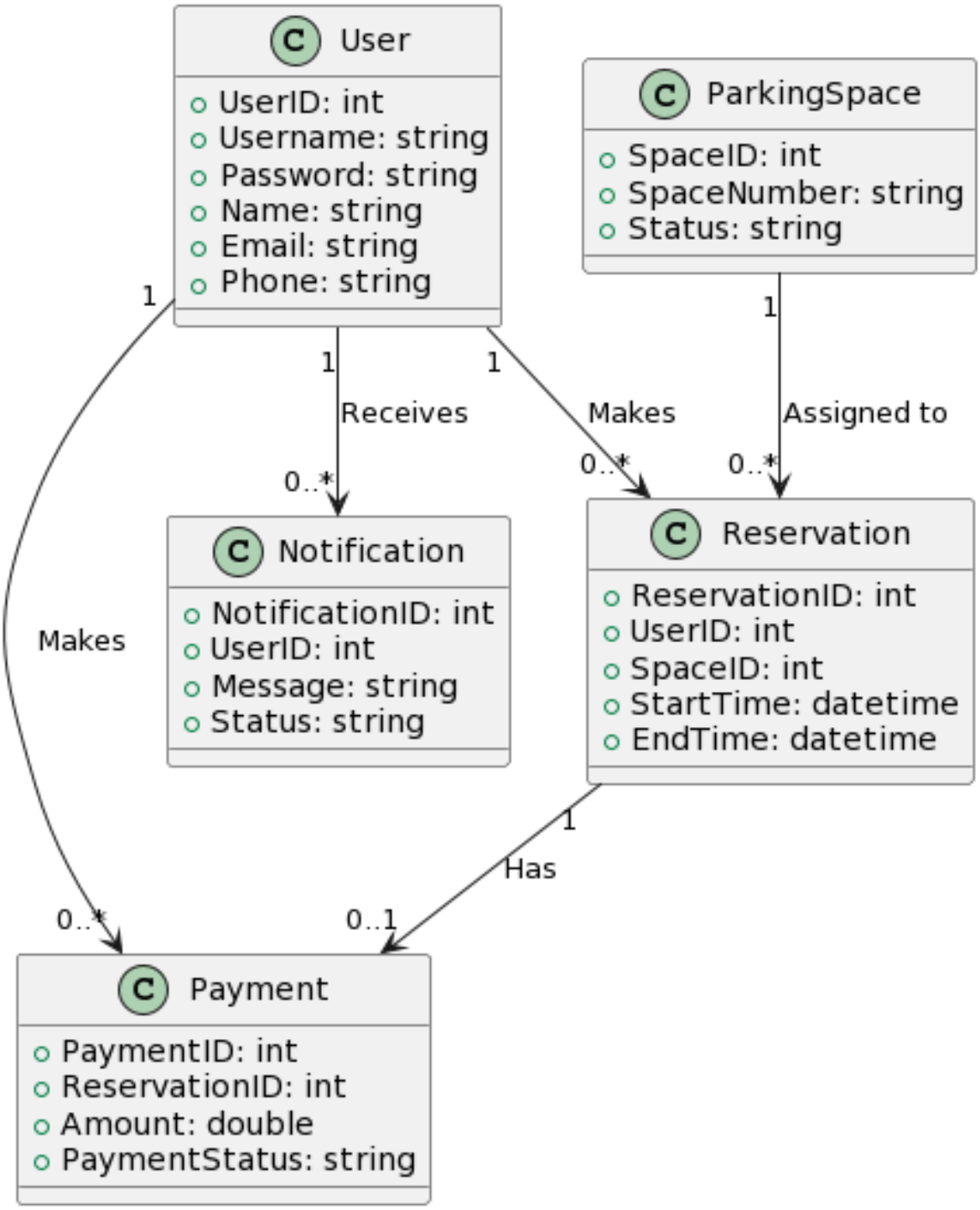


Figure 4.2

3.3 Entity Relationship Diagram with data dictionary

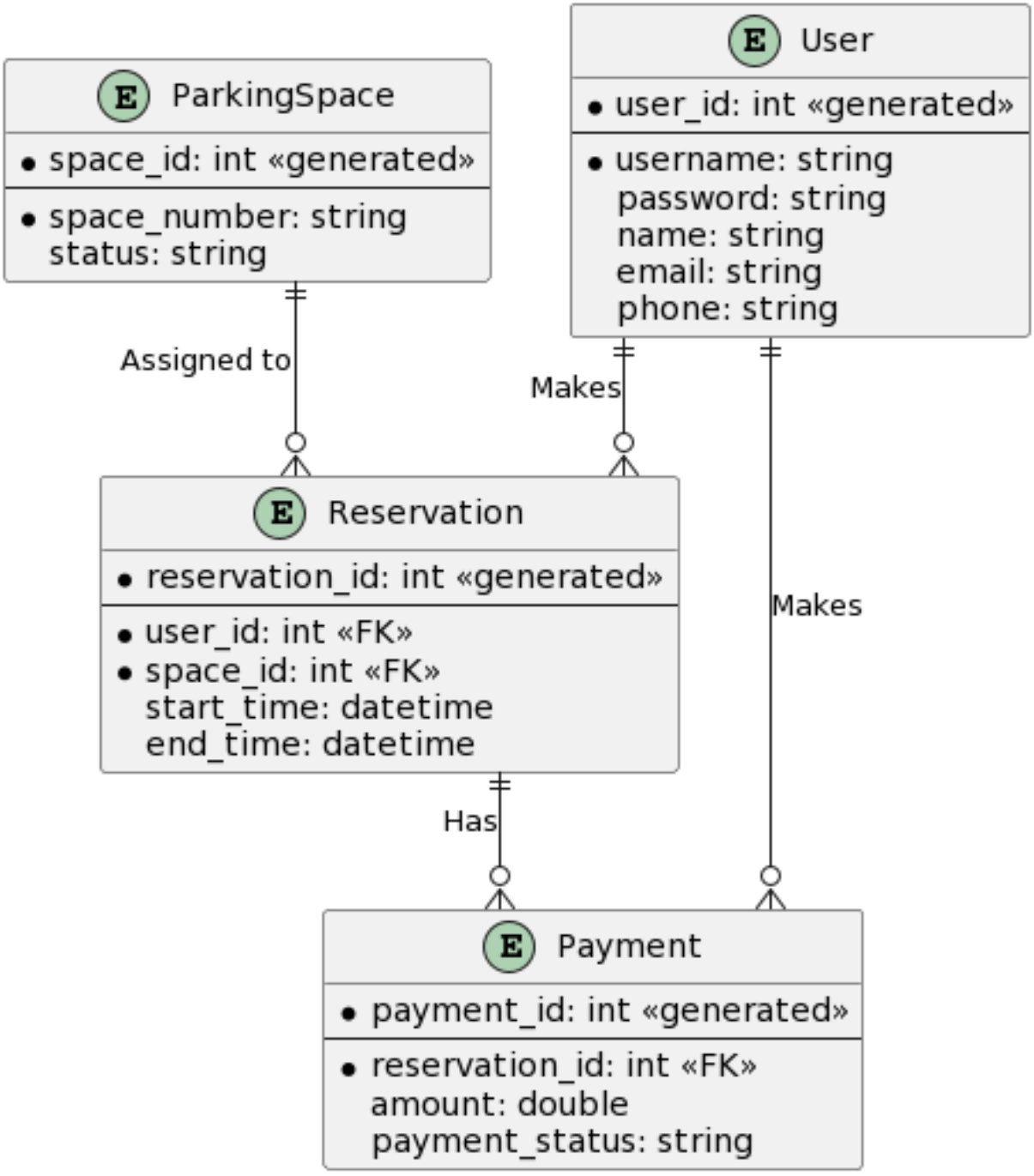


Figure 4.3

3.4 Class Diagram

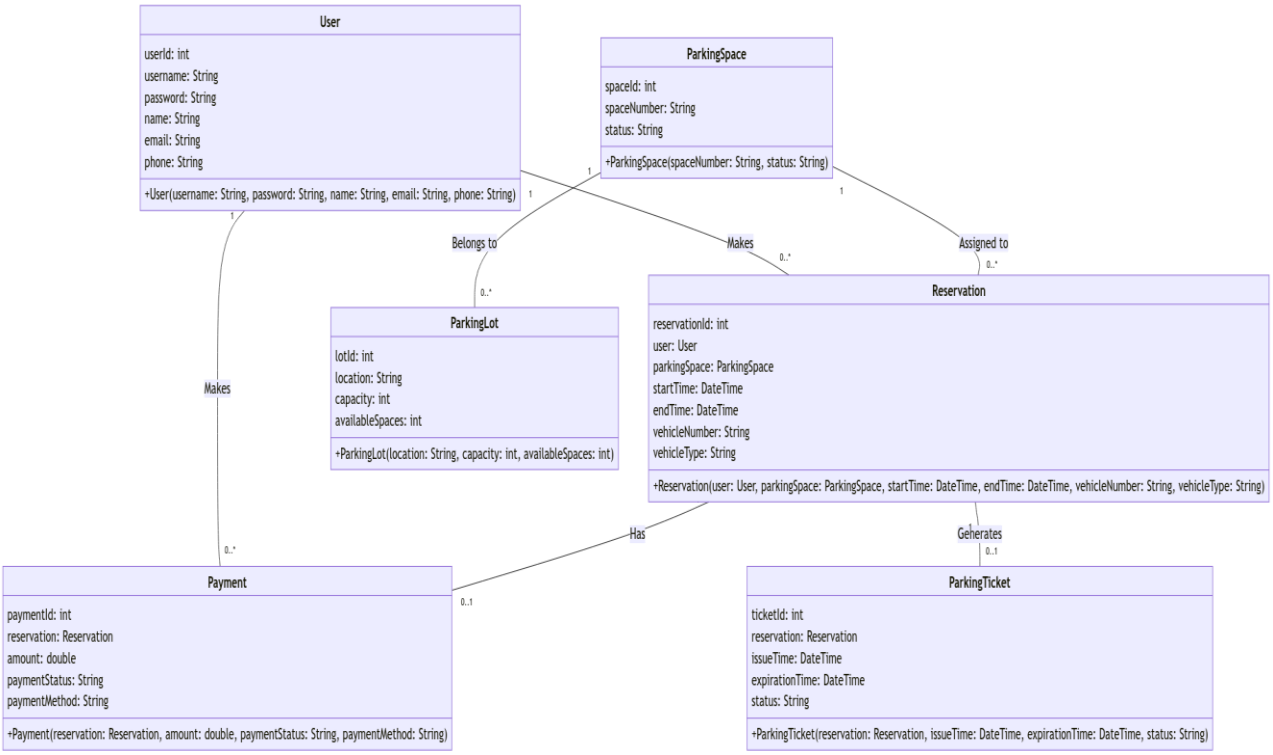


Figure 4.4

3.5 Sequence / Collaboration Diagram

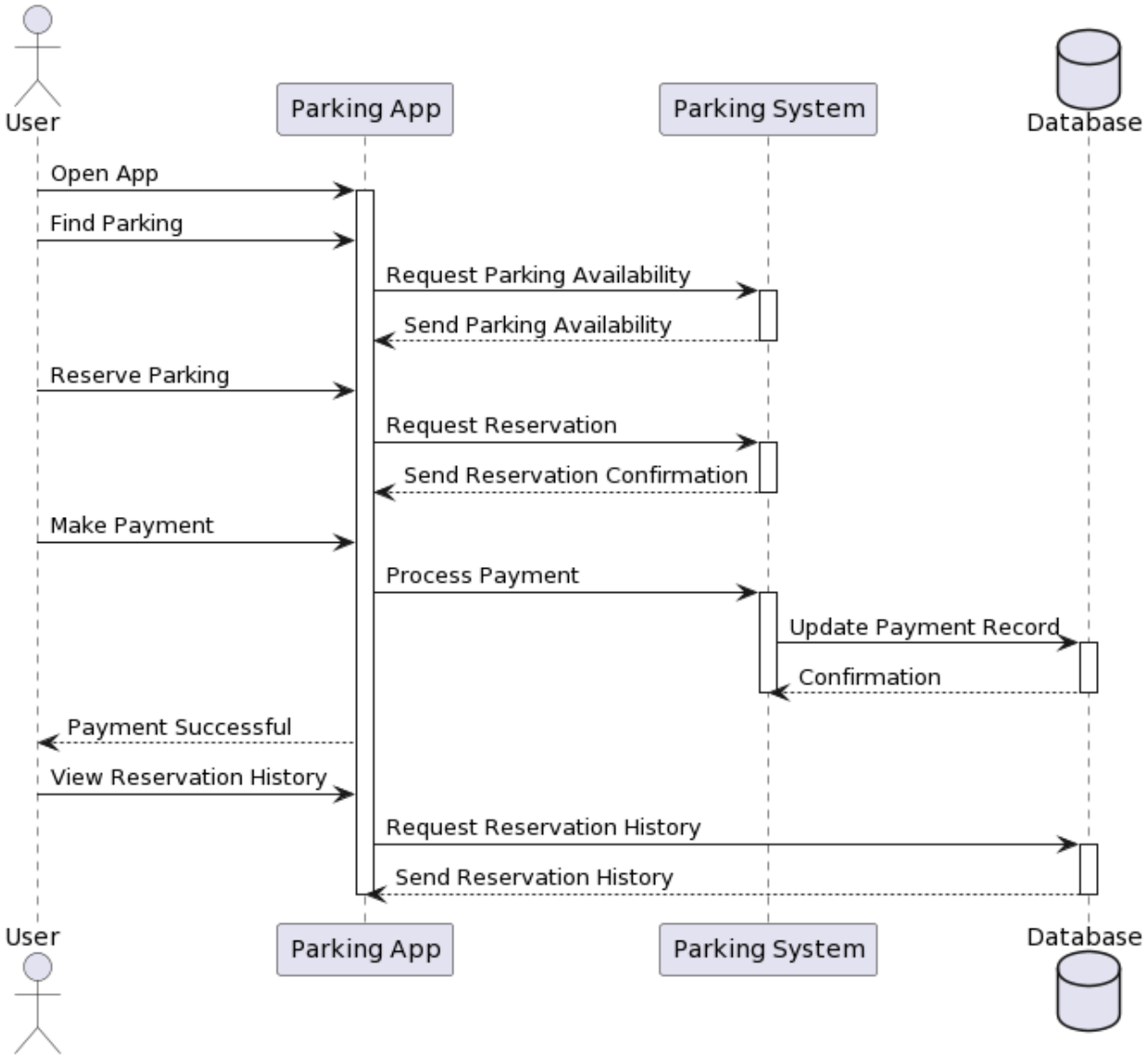


Figure 4.5

3.6 Operation Contracts

Operation	Reserve Parking
Description	This operation enables users to reserve a parking space within the parking application.
Preconditions	a. The user must have a registered account in the parking app. b. The user must be logged into the app. c. The parking app must have access to real-time parking space availability information.
Inputs	a. **User ID:** Identifier of the user making the reservation. b. **Selected Parking Space:** Identification of the parking space chosen by the user. c. **Desired Reservation Duration:** Start time and end time for the parking reservation.
Outputs	➢ **Confirmation Message:** A confirmation message indicating the successful reservation. ➢ **Reservation Details:** Information about the reserved parking space, including location and duration.
Post Conditions	➢ The reserved parking space is marked as "Booked" in the system. ➢ The user receives a confirmation of the reservation.
Exceptions	• Insufficient Parking Space: If no parking spaces are available, the application notifies the user and suggests alternative options. • Invalid Reservation Duration: If the selected duration is not valid (e.g., too short or too long), the application prompts the user to choose a valid duration.
Error Handling	• Insufficient Balance: If the user's account balance is insufficient for the reservation, the application prompts the user to add funds. • Technical Failure: In case of a technical failure during the reservation process, the

	application displays an error message and prompts the user to try again.
Security Considerations	➢ User Authentication: Ensure that the user making the reservation is authenticated. ➢ Secure Data Transmission: The reservation data must be transmitted securely over the network.
Performance Requirements	➢ The reservation process should be completed within a reasonable time frame, aiming for less than 10 seconds. ➢ The application should handle multiple reservation requests simultaneously.
Usability Requirements	➢ The user interface should clearly present available parking spaces and guide the user through the reservation process. ➢ User feedback on successful reservations should be presented in a user-friendly manner.
Scalability	➢ The reservation system should scale to accommodate a growing number of users and parking spaces.

3.7 Activity Diagram

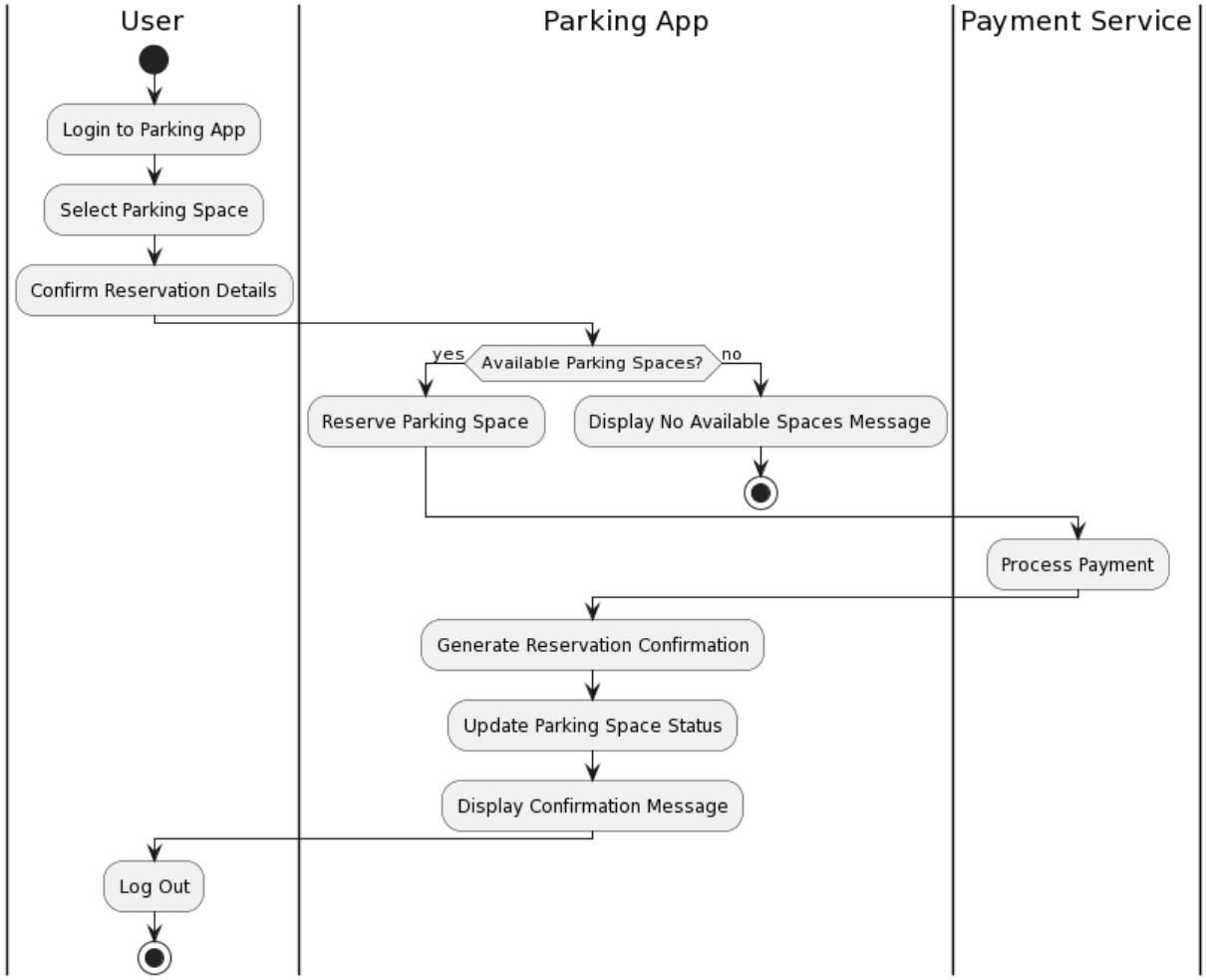


Figure 4.7

3.8 State Transition Diagram

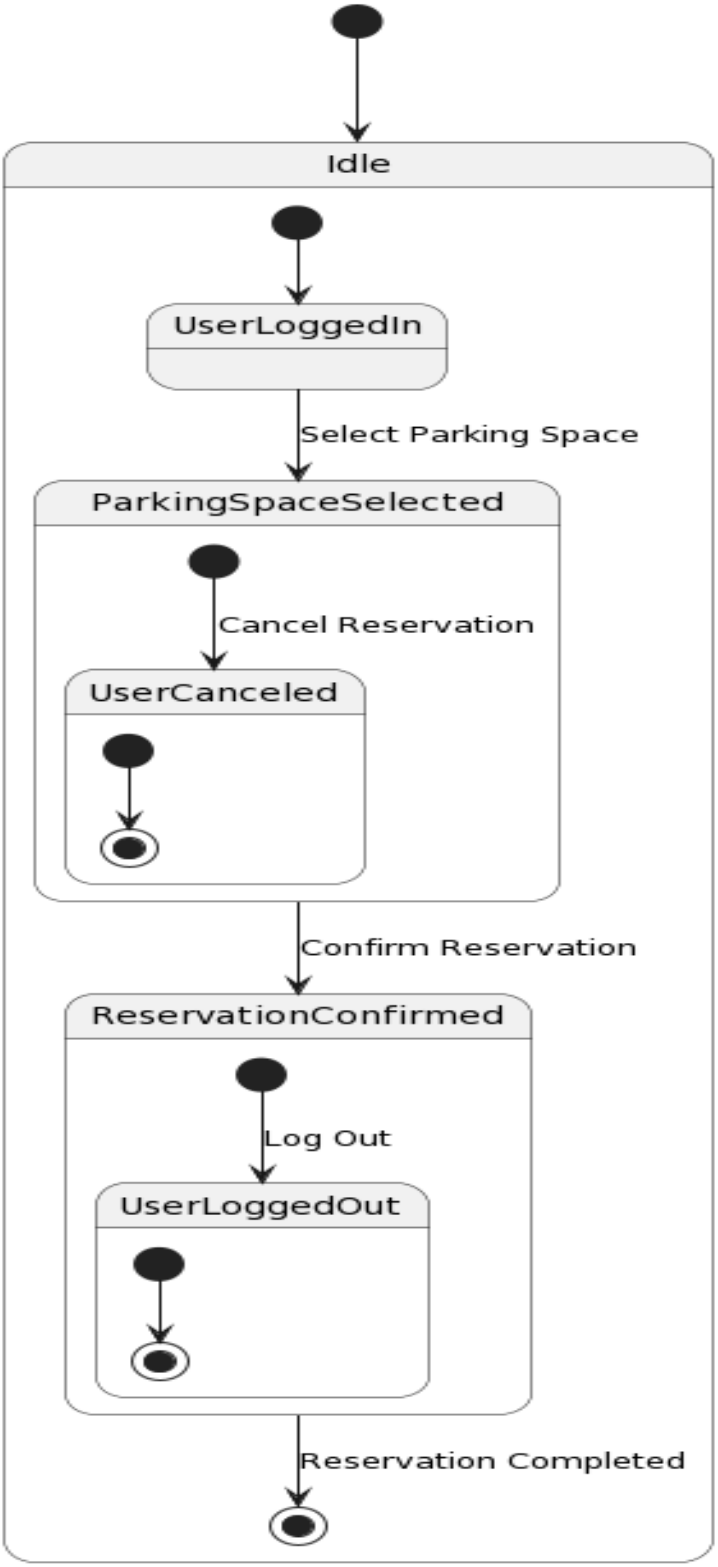


Figure 4.8

3.9 Component Diagram

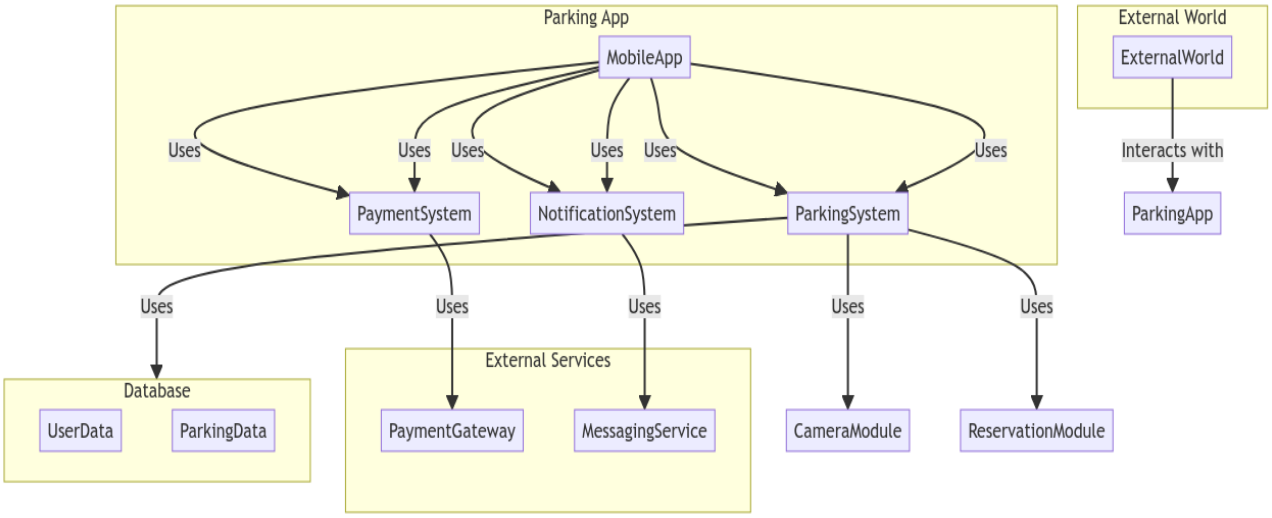


Figure 4.9

3.10 Deployment Diagram

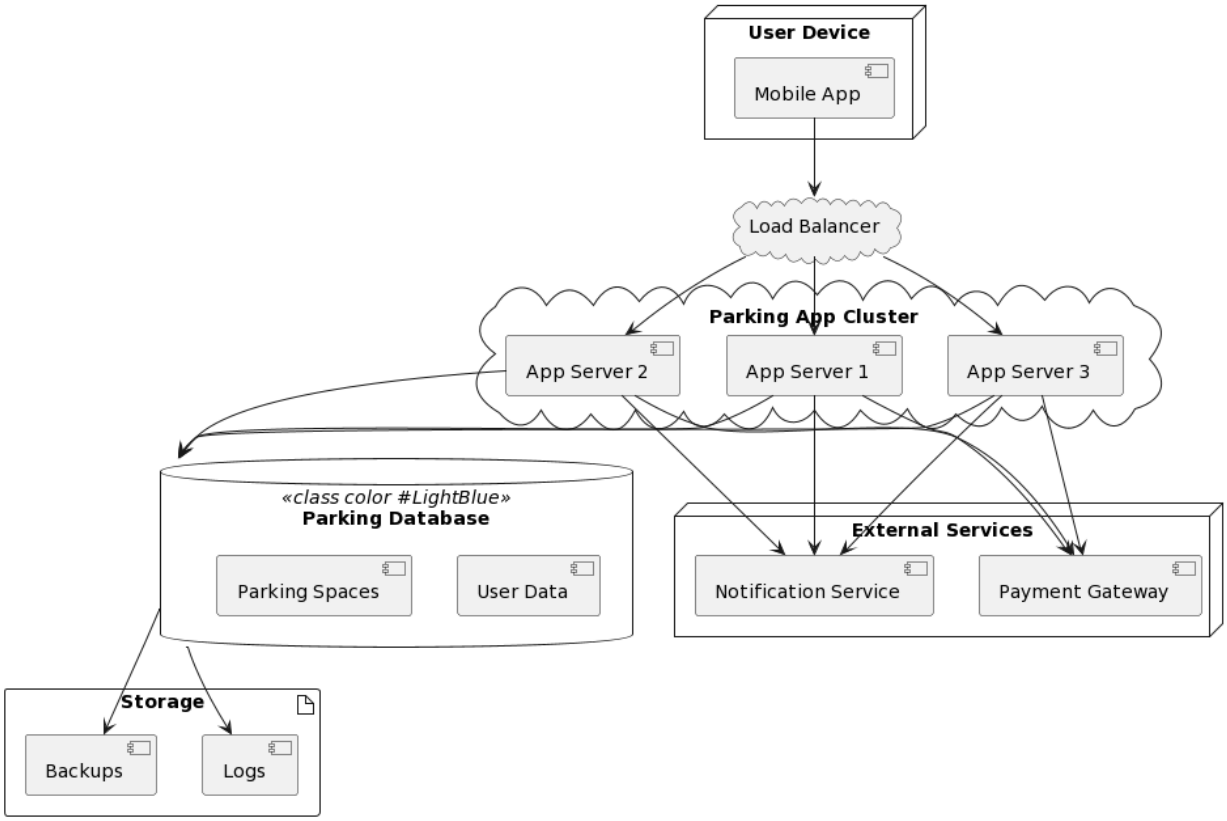


Figure 4.10

3.11 Data Flow diagram [*only if structured approach is used - Level 0 and 1*]

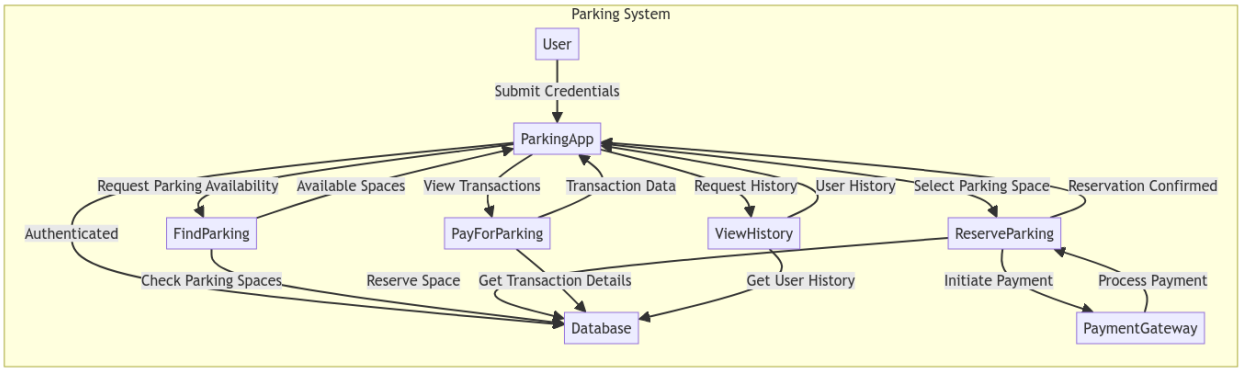


Figure 4.11.1

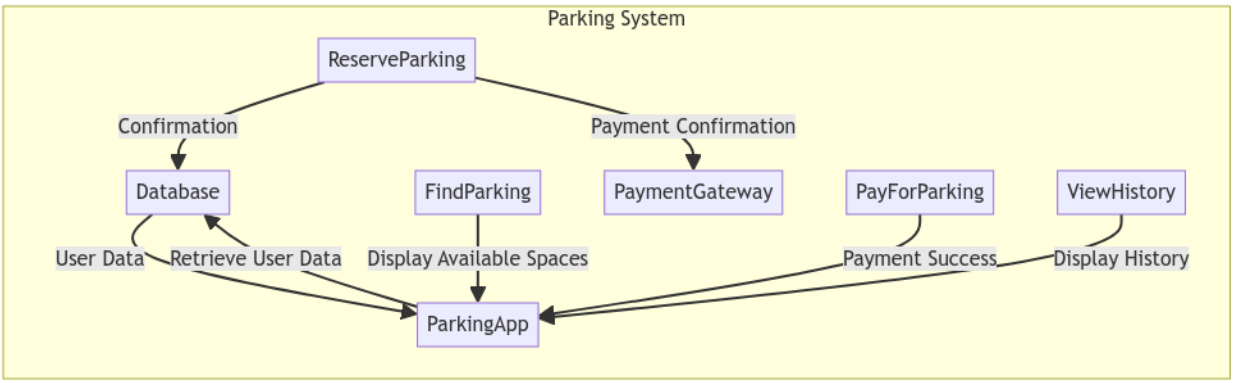


Figure 4.11.2

Chapter 5
Implementation

Chapter 5: Implementation

In this chapter, we present a comprehensive overview of the technical implementation of the parking app, covering various aspects of the development process. The following sections delve into the key technical components and considerations involved in bringing the parking app to life.

- Technology Stack:
  - Frontend
  - Backend
  - Database
  - Mobile Development
- Development Environment
- User Interface Design
- Backend Development
- Database Management
- Integration of External Services
- Testing and Debugging
- Performance Optimization
- Security Measures
- Limitations and Challenges

5.1. Important Flow Control/Pseudo codes

```
function main():
  displayWelcomeScreen()
  while true:
    displayMainMenu()
    option = getUserInput()
    switch option:
      case 1:
        findParking()
        break
```

```
        case 2:
            reserveParking()
            break
        case 3:
            viewParkingHistory()
            break
        case 4:
            exitApplication()
            break
        default:
            displayErrorMessage("Invalid option. Please try again.")
function findParking():
    displayFindParkingScreen()
    location = getUserLocation()
    availableSpaces = getParkingAvailability(location)
    if availableSpaces.isNotEmpty():
        displayAvailableSpaces(availableSpaces)
        selectedSpace = getUserInput()
        if isValidParkingSpace(selectedSpace):
            displayReservationConfirmation()
            confirmReservation(selectedSpace)
        else:
            displayErrorMessage("Invalid parking space. Please try again.")
    else:
        displayErrorMessage("No available parking spaces. Try another location.")
function reserveParking():
    displayReservationScreen()
    user = getCurrentUser()
    selectedSpace = getUserReservedSpace(user)
```

```
    if selectedSpace.isNotEmpty():
        displayReservedSpaceDetails(selectedSpace)
        confirmPayment = getUserConfirmation()
        if confirmPayment:
            processPayment(selectedSpace)
            displayPaymentConfirmation()
            updateReservationStatus(selectedSpace, "Paid")
        else:
            cancelReservation(selectedSpace)
            displayCancellationMessage()
    else:
        displayErrorMessage("You have no reserved parking space.")
function viewParkingHistory():
    user = getCurrentUser()
    reservationHistory = getUserReservationHistory(user)
    if reservationHistory.isNotEmpty():
        displayReservationHistory(reservationHistory)
    else:
        displayMessage("No reservation history.")
function exitApplication():
    displayGoodbyeMessage()
    terminateApplication()
```

5.2. Components, Libraries, Web Services and stubs

Components:

- **User Interface Components:**
    - Map Interface: Integrates with mapping services to display parking locations.
    - Reservation Screens: Allows users to find, reserve, and pay for parking spaces.
    - History Screens: Displays user parking history.

- **Backend Components:**
    - Parking Management System: Handles parking space availability, reservations, and transaction processing.
    - User Authentication System: Manages user authentication and authorization.
    - Payment Gateway Integration: Enables secure payment processing for parking reservations.
- **External Integrations:**
    - Map Service API: Integrates with mapping services (Google Maps, Mapbox) for location visualization.
    - Payment Gateway API: Connects with external payment processing services.
    - Notification Service: Sends notifications for successful reservations, payments, and reminders.
- **Database:**
    - Parking Database: Stores information about parking spaces, reservations, and user data.

Libraries:

- **Frontend Frameworks:**
    - React, Angular, or Vue.js: For building interactive and responsive user interfaces.
    - Leaflet or Google Maps API: For map-related functionalities.
- **Backend Frameworks:**
    - Node.js, Django, or Flask: For server-side logic.
    - Express.js or Django REST framework: For building RESTful APIs.
- **Database ORM:**
    - Sequelize (for Node.js) or Django ORM: Simplifies database interactions.
- **Payment Processing:**
    - Stripe, PayPal, or Braintree SDKs: Facilitates integration with payment gateways.

Web Services:

- **Parking Availability Service:**

- Exposes an API to check the availability of parking spaces.

- **Reservation Service:**
  - Manages the reservation lifecycle, including creation, modification, and cancellation.
- **Payment Service:**
  - Coordinates payment processing and communicates with the chosen payment gateway.

Stubs:

- **Testing Stubs:**
  - Mock services for testing components in isolation, simulating responses from external services.
- **Map Service Stub:**
  - A stub or mock service to simulate interactions with mapping services during development and testing.
- **Payment Gateway Stub:**
  - A simulated payment gateway for testing payment-related functionalities.

5.3. Deployment Environment

Platform:

The Parking App is developed for specific platforms such as Android and iOS to cater to a wide user base.

Hosting Options:

Cloud-based platforms like AWS, GCP, or Azure are considered for scalable and reliable hosting to handle varying loads during peak times.

Compatibility:

The application undergoes extensive compatibility testing to ensure seamless performance across various devices and operating system versions.

Security:

Stringent security measures are implemented, including encryption, secure APIs, and adherence to industry-standard security practices, to safeguard user data.

CI/CD:

Continuous Integration and Continuous Deployment (CI/CD) practices are adopted. Automated testing, builds, and efficient deployment of updates are integral to the development process.

Monitoring:

Deployment includes robust monitoring tools and error handling mechanisms for proactive issue identification and resolution, ensuring a smooth user experience.

App Store Deployment:

App store guidelines are strictly followed during the submission and review process to ensure compliance and successful deployment on platforms like Google Play Store or Apple App Store.

5.4. Tools and Techniques

- **IDE:** Development is carried out using Android Studio for Android and Xcode for iOS, providing a comprehensive environment for coding, debugging, and testing.
- **VCS:** Git is employed as the version control system (VCS) to manage code changes, enable collaboration, and streamline the development process.
- **Collaborative Tools:** Communication and task management are facilitated through collaborative tools such as Slack, Microsoft Teams, or Jira.
- **Code Review:** GitHub or Bitbucket is utilized for code review, ensuring code quality and adherence to best practices.
- **Performance Profiling Tools:** Tools like Android Profiler or Xcode Instruments are employed for performance analysis to optimize the app's responsiveness.
- **Automated Build and Deployment:** Gradle or Fastlane is used for automated build and deployment processes, streamlining the release cycle.
- **Crash Reporting and Analytics:** Firebase Crashlytics or Google Analytics is integrated for real-time monitoring, crash reporting, and gaining insights into user behavior.

5.5. Best Practices / Coding Standards

- **Consistent Code Formatting:**

  Maintaining a standardized code formatting style ensures improved readability and consistency across the codebase.

- **Meaningful Naming Conventions:**

  Descriptive names for variables, functions, and classes enhance code readability and understanding.

- **Modular and Maintainable Code Structure:**

  Organizing code into logical modules and following separation of concerns principles contribute to a modular and maintainable codebase.

- **Commenting and Documentation:**

  Including comments and comprehensive documentation aids in understanding the codebase and accelerates onboarding for new developers.

- **Error and Exception Handling:**

  Implementing proper error handling mechanisms prevents crashes and ensures graceful handling of unexpected scenarios.

- **Code Review and Collaboration:**

  Regular code reviews are conducted to maintain code quality, and collaboration among team members is fostered for efficient development.

- **Security Measures:**

  Secure coding practices are implemented to protect user data, prevent vulnerabilities, and ensure a secure user experience.

5.6. Version Control

Version Control System (VCS):

Git is utilized as the version control system to manage code changes, enable collaboration, and support different development stages.

Branching Strategies:

Gitflow, a branching strategy, is followed to manage different development stages efficiently.

Benefits:

Version control facilitates efficient collaboration, reduces conflicts, and supports continuous integration and deployment (CI/CD) practices. It ensures code integrity and timely updates in the Parking App project.

Chapter 6

Testing and Evaluation

Chapter 6: Testing and Evaluation

In this pivotal chapter, we delve into the critical aspects of testing and evaluation for the Parking App. Rigorous testing is paramount to ensure the reliability, performance, and security of the application. The chapter navigates through various testing methodologies, including unit testing, integration testing, and system testing, shedding light on their roles in detecting and rectifying potential issues. Evaluation parameters are defined, considering user experience, functionality, and adherence to specifications. Real-world scenarios are simulated to validate the Parking App's robustness, and user feedback is instrumental in refining the application. Security audits and performance assessments are conducted to fortify the app against potential threats and enhance its responsiveness. This chapter serves as a comprehensive guide to validating the Parking App's efficacy, ensuring it meets user expectations and industry standards.

6.1 Use Case Testing

6.1.1 Login Page Testing

Table Login Page Testing

Test Case ID	TC-001
Test Case Summary	When admin click on login button it must be login and show the main page.
Prerequisites	User must be on login page and have username and password.
Test Procedure	Verify the user email and password.
Actual Result	User successful login.
Status	Successful.
Test Steps	• Navigate the login page • Enter the username • Enter the password • And click on login button • Logout

6.1.2 Sign Up Testing

Table Sign Up Page Testing

Test Case ID	TC-002
Test Case Summary	User must be signup after filling form.
Prerequisites	User must be on signup page.
Test Procedure	Verify the inputs given in form.
Expected Result	User must be allotted code and username, password and SignBot Dashboard.
Actual Result	User successful signup.
Status	Successful.
Test Steps	• Navigate the login page • Fill up the form. • And click on login button

6.1.3 Extension Errors

Table Extension Error

No.	Step	Description
1-a	Login Failed	Login failed if user enter wrong username. System will display an error message.
1-b	Login Failed	Login failed if user enter wrong password. System will display an error message.
2	Registration Failed	Registration will have failed if a new user doesn't follow the rules. System will display an error message.

6.2 Equivalence Partitioning

Input Variables:

- Parking Availability
- Parking Duration
- Payment Method
- User Location

- Reservation Confirmation
- Parking Space Type
- User Profile Type
- Discounts and Promotions
- Vehicle Type
- Parking App Version

Equivalence Classes:

Parking Availability:

- Equivalence Class 1: Sufficient parking spaces available.
- Equivalence Class 2: Limited parking spaces available.
- Equivalence Class 3: No parking spaces available.

Parking Duration:

- Equivalence Class 4: Short-duration parking (e.g., 1 hour).
- Equivalence Class 5: Moderate-duration parking (e.g., 4 hours).
- Equivalence Class 6: Long-duration parking (e.g., 12 hours).

Payment Method:

- Equivalence Class 7: Credit card payment.
- Equivalence Class 8: Mobile wallet payment.
- Equivalence Class 9: Debit card payment.

User Location:

- Equivalence Class 10: Urban area.
- Equivalence Class 11: Suburban area.
- Equivalence Class 12: Rural area.

Reservation Confirmation:

- Equivalence Class 13: Reservation confirmed.
- Equivalence Class 14: Reservation pending.
- Equivalence Class 15: Reservation declined.

Parking Space Type:

- Equivalence Class 16: Standard parking space.

- Equivalence Class 17: Handicap-accessible parking space.
- Equivalence Class 18: Electric vehicle charging station.

User Profile Type:

- Equivalence Class 19: Regular user profile.
- Equivalence Class 20: Premium user profile with additional benefits.
- Equivalence Class 21: Guest user profile.

Discounts and Promotions:

- Equivalence Class 22: No discounts or promotions applied.
- Equivalence Class 23: Seasonal discount applied.
- Equivalence Class 24: Referral promotion applied.

Vehicle Type:

- Equivalence Class 25: Compact car.
- Equivalence Class 26: SUV or van.
- Equivalence Class 27: Motorcycle.

Parking App Version:

- Equivalence Class 28: Latest app version.
- Equivalence Class 29: Previous app version.

Boundaries:

- Boundary 1: Transition from sufficient to limited parking spaces.
- Boundary 2: Transition from short-duration to moderate-duration parking.
- Boundary 3: Transition from credit card to mobile wallet payment.
- Boundary 4: Transition from urban to suburban user locations.
- Boundary 5: Transition from reservation confirmed to pending status.

Test Cases:

- Test Case 1: Sufficient parking spaces, short-duration parking, credit card payment, urban location, reservation confirmed, standard parking space, regular user profile, no discounts, compact car, latest app version.

- Test Case 2: Limited parking spaces, moderate-duration parking, mobile wallet payment, suburban location, reservation pending, handicap-accessible parking space, premium user profile, seasonal discount applied, SUV or van, previous app version.
- Test Case 3: No parking spaces available, long-duration parking, debit card payment, rural location, reservation declined, electric vehicle charging station, guest user profile, referral promotion applied, motorcycle, latest app version.

Verification:

- Confirm that each test case falls into one of the defined equivalence classes.
- Validate that the boundaries are covered.

6.3 Boundary Value Analysis

Boundary testing is the process of testing that is between start point and extreme ends or boundaries between partials of input values.

6.1.1 Email Fields

- Email fields are also available in the mobile app.
- Email fields must accepts email formats.

6.1.2 Login Password

- Username that a person enter in the text box at the time of login must be correct.
- If the username is not correct, the system must restrict the user at the current page.

6.4 Data Flow Testing

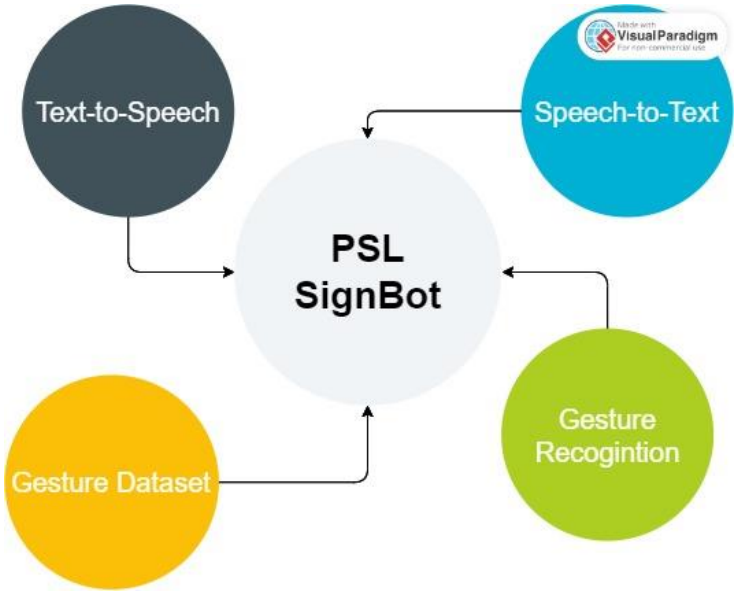
Data flow testing for the Parking App focuses on evaluating the system's behavior concerning the flow of data related to parking-related activities. The central component in this context is the Parking Database, which plays a pivotal role in managing and storing information related to parking availability, reservations, payments, and user history.

The testing process involves assessing how data flows through different components of the system:

- **Parking Availability Data Flow:** The system must accurately update the parking availability status based on user interactions, reservations, and real-time information about parking spaces.

Testing ensures that the parking availability data is consistently and correctly updated across the application.

- **Reservation Data Flow:** When a user initiates a parking reservation, the system must efficiently process and store reservation details in the Parking Database. Data flow testing ensures the accuracy of reservation information, including parking space allocation, user details, and reservation status.
- **Payment Data Flow:** For successful parking transactions, the payment data flow is critical. Testing validates the secure and accurate transfer of payment information between the Parking App, Payment Gateway, and the Parking Database, ensuring financial transactions are processed correctly.
- **User History Data Flow:** The system maintains a record of user interactions and transactions. Data flow testing verifies that user history is appropriately updated, reflecting accurate details of parking history, payments, and any relevant notifications.
- **Notification Data Flow:** In cases where the system sends notifications to users, testing evaluates the flow of notification data to ensure timely and accurate delivery.
- **Admin Dashboard Data Flow:** For administrators managing parking spaces, the data flow from the Parking Database to the admin dashboard is tested to ensure that real-time data updates are reflected in the administrative interface.



6.5 Unit Testing

Unit testing is carried out screen-wise, each screen being identified as an object.
Attention is diverted to individual modules, independently to one another to locate errors. This has enabled the detection of errors in coding and logic.

Test Case	Step Details	Test Data	Expected Result	Actual Result	Pass / Fail
UT-001	User Authentication	Valid User credentials	User successfully authenticated	[Tested] User authenticated successfully	Pass
UT-002	Parking availability check	Parking lot ID , Time	Available parking spaces retrieved	[Tested] Correct number of available spaces retrieved	Pass
UT-003	Reserving process	User ID, Parking Space ID, Time	Reservation confirmed successfully	[Tested] Reservation confirmed successfully	Pass
UT-004	Payment processing	Reservation ID, Payment Details	Payment processed successfully	[Tested] Payment processed successfully	Pass
UT-005	User history retrieval	User ID	User's parking history retrieved	[Tested] Correct user history retrieved	Pass
UT-006	Parking space management	Add, Update, Delete operations	Parking spaces updated correctly	[Tested] Parking spaces updated correctly	Pass
UT-007	Notifications	User ID, Notification Details	Notification sent successfully	[Tested] Notification sent successfully	Pass
UT-008	Access control	Unauthorized access attempt	Access denied as expected	[Tested] Access denied as expected	Pass

6.6 Integration Testing

Table Integration Testing Each Button

Test Case ID	IT_001
Test Case Summary	To verify that all buttons behave as expected
Prerequisites	All button must be rendered
Test Procedure	Click on each button to see if it performs correctly.
Expected Result	• Login button should provide you a dashboard • Exit button redirect you on login page
Actual Result	• Login button provide user to enter in software • Exit button redirect you on a login page
Status	Pass
Test Environment	Java

6.7 Performance Testing

6.7.1 Test Case Number: PT-01

Table System Response Timing

Test Type	Performance Testing	Test Result
Required Performance	The required response time of the system is less than 3 second.	
System Performance	The actual performance response time is 2 second.	Test passed

6.7.2 Test Case Number: PT-01

Table Activity Load Time Testing

Test Type	Performance Testing	Test Result
Required Performance	The required load time of the page is less than 2 second.	
System Performance	The actual performance response time is 1 second.	Test passed

6.7.3 Test Case Number: PT-01

Test Case Title: Waiting Time Testing

Table Waiting Time Testing

Test Type	Performance Testing	Test Result
Required Performance	The average waiting time of the system is less than 3 second.	
System Performance	The actual performance response time is 2 second in some circumstances.	Test passed

6.7.4 Test Case Number: PT-01

Test Case Title: Connection Establish Time Testing

Table Connection Establish Time Testing

Test Type	Performance Testing	Test Result
Required Performance	The connection establish is less than 3 second.	
System Performance	The actual connection establish time is about 2 second.	Test passed

6.8 Stress Testing

Stress testing for the Parking App is vital to ensure the system's stability and reliability under extreme conditions. The following aspects are examined in stress testing:

a. Data Persistence under Stress:

- Objective: Evaluate the system's capability to save parking-related data before any potential crash.
- Scenario: Simulate a scenario with a high volume of simultaneous parking reservations and transactions.
- Expected Outcome: Confirm that the system maintains data integrity, and critical parking information is not lost.

b. Server Connection Stability:

- Objective: Verify the reliability of the connection between the Parking App and the server under heavy loads.

- Scenario: Introduce a significant number of users attempting to access parking information concurrently.
- Expected Outcome: Assess how well the system maintains consistent communication with the server, preventing disruptions in parking-related services.

c. Graceful Handling of Server Unresponsiveness:

- Objective: Ensure the system gracefully handles situations where the server is unresponsive.
- Scenario: Simulate server unresponsiveness during critical operations like parking reservations.
- Expected Outcome: Verify that the system intelligently handles unresponsive server conditions, providing a seamless experience for users without crashes.

d. Successful Data Saving and Confirmation:

- Objective: Validate that the system successfully saves parking-related data even under stress.
- Scenario: Stress the system with a high number of concurrent parking space reservations and payment transactions.
- Expected Outcome: Confirm that users receive clear success messages after each operation, indicating that their parking-related actions have been processed and stored securely.

e. Accurate Display of Modifications and Updates:

- Objective: Test the system's ability to accurately display modifications and updates to parking-related records under stress.
- Scenario: Introduce rapid updates and modifications to parking space information during peak usage.
- Expected Outcome: Ensure that the system correctly reflects changes made to parking spaces and updates the records accurately, providing users with real-time and reliable information.

Chapter 7

Summary, Conclusion and Future Enhancements

Chapter 7: Summary, Conclusion & Future Enhancements

Sure, here is a summary of the project, achievements, critical review, lessons learned, and future enhancements/recommendations for Parking App:

7.1 Project Summary

The Parking App stands as a transformative solution in the realm of modern urban living, addressing the challenges of parking management with innovative technology. This mobile application redefines the way users navigate and interact with parking spaces, streamlining the entire parking experience.

Utilizing advanced features and intuitive design, the Parking App empowers users to effortlessly find, reserve, and pay for parking spaces. Its user-friendly interface provides a seamless journey, from locating available parking spots to completing transactions securely. The app incorporates real-time data on parking space availability, allowing users to make informed decisions and avoid unnecessary hassles.

The Parking App goes beyond conventional parking solutions by introducing smart features such as reservation extensions and transaction history tracking. Admin capabilities enable efficient parking space management, ensuring optimal utilization and a well-organized parking infrastructure.

By leveraging technology, the Parking App aims to revolutionize urban mobility, alleviate parking-related stress, and contribute to a more efficient and sustainable transportation ecosystem. This project is not just about parking; it's a step towards enhancing the overall urban experience, making cities more accessible, connected, and user-centric.

7.2 Achievements and Improvements

The Parking App has made significant strides in enhancing user experience and optimizing parking management. Key achievements and improvements include:

- **Real-Time Parking Availability Updates:** The Parking App now features real-time updates on parking space availability, ensuring users have the latest information to make informed decisions. This improvement minimizes the time spent searching for parking spots and enhances overall efficiency.

- **Intelligent Reservation System:** The app incorporates an intelligent reservation system, allowing users to reserve parking spaces in advance. This feature streamlines the parking process, providing a hassle-free experience and reducing uncertainties related to parking availability.
- **Enhanced User Interface:** A major achievement lies in the revamped user interface, offering a more intuitive and visually appealing design. The improved interface contributes to a seamless user journey, from login to payment, making the app more accessible to a diverse user base.
- **Extended Admin Capabilities:** For parking space administrators, the Parking App now offers extended capabilities for managing and monitoring parking spaces. Admins can efficiently handle reservations, view transaction histories, and optimize parking space utilization for better overall management.
- **Expanded Payment Options:** To cater to user preferences, the app has expanded its payment options. Users now have a variety of secure and convenient payment methods, contributing to a more user-centric and flexible payment experience.
- **Geolocation Integration:** The app now utilizes geolocation technology to provide more accurate and personalized information to users. This ensures that users can easily navigate to available parking spaces and receive location-specific details.

7.3 Critical Review

Despite significant achievements, the Parking App undergoes continuous evaluation, revealing areas for improvement:

a. Real-Time Parking Updates:

- **Current Strength:** The Parking App effectively provides real-time parking space availability.
- **Room for Improvement:** Ongoing efforts focus on optimizing the speed of real-time updates, aiming for quicker and more fluid communication between the app and users.

b. Parking Space Coverage Expansion:

- **Current Progress:** The app actively incorporates more parking spaces and locations.

- **Global Landscape:** To comprehensively address diverse parking needs globally, ongoing initiatives aim to expand coverage and include a broader range of parking facilities.

c. Customization and User Personalization:

- **User-Friendly Approach:** The Parking App is designed with user-friendliness in mind.
- **Future Focus:** Work is underway to introduce customization options, allowing users to personalize their parking preferences and optimize their overall parking experience.

7.4 Lessons Learnt

The development of the Parking App has provided invaluable insights:

a. Collaboration Dynamics:

- Lesson: Collaborating with urban planning experts and technology developers is crucial.
- Why: This collaboration ensures that the app aligns with real-world parking dynamics and user needs, enhancing its effectiveness in various urban landscapes.

b. Continuous Data Refinement:

- Lesson: The app's algorithms benefit from ongoing data collection and refinement.
- Why: Regular updates to parking data ensure accuracy and relevance, reflecting the dynamic nature of parking availability in urban environments.

c. User-Centric Development:

- Lesson: Active user engagement and feedback are vital throughout development.
- Why: User insights are instrumental in identifying areas for improvement, ensuring that the Parking App remains responsive to the diverse needs of its user base.

7.5 Future Enhancements/Recommendations

Looking ahead, the Parking App envisions several enhancements:

a. Smart Parking Assistance:

- **Enhancement:** Implementing AI-driven suggestions for optimal parking.
- **Why:** This feature will assist users in finding the most convenient parking spaces based on their preferences and real-time availability.

b. Integration with Navigation Apps:

- **Enhancement:** Integrating the app with popular navigation platforms.

- **Why:** Seamless integration with navigation apps ensures a holistic approach to parking, providing users with a complete solution for urban mobility.

c. Parking Analytics Dashboard:

- **Enhancement:** Developing an analytics dashboard for parking administrators.
- **Why:** This tool will empower administrators to analyze parking patterns, optimize space allocation, and enhance overall parking management efficiency.

Reference and Bibliography

Reference and Bibliography

Canli, H., & Toklu, S. (2021). Deep learning-based mobile application design for smart parking. *IEEE Access*, *9*, 61171-61183.

Owayjan, M., Sleem, B., Saad, E., & Maroun, A. (2017, September). Parking management system using mobile application. In *2017 Sensors Networks Smart and Emerging Technologies (SENSET)* (pp. 1-4). IEEE.

Grazioli, A., Picone, M., Zanichelli, F., & Amoretti, M. (2013, June). Collaborative mobile application and advanced services for smart parking. In *2013 IEEE 14th International Conference on Mobile Data Management* (Vol. 2, pp. 39-44). IEEE.

Lotlikar, T., Chandrahasan, M., Mahadik, A., Oke, M., & Yeole, A. (2016). Smart parking application. *International Journal of Computer Applications*, *149*(9), 32-37.

Fikri, R. M., & Hwang, M. (2019, November). Smart parking area management system for the disabled using IoT and mobile application. In *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)* (pp. 172-176). IEEE.

Saeliw, A., Hualkasin, W., Puttinaovarat, S., & Khaimook, K. (2019). Smart car parking mobile application based on RFID and IoT.

Ma, R., Lam, P. T., & Leung, C. K. (2018). Potential pitfalls of smart city development: A study on parking mobile applications (apps) in Hong Kong. *Telematics and Informatics*, *35*(6), 1580-1592.