Princess Sumaya جامعـــة
University الأميـــرة سميّـــة
for Technology للتكنولوجيا

King Hussein School of Computing Sciences

Software Engineering Department

SwiftSpot

Team Members:

Omar Obeid - 20201009
Khalil Samara - 20200836
Ibrahim Abu Laban - 20200887

Supervised by:
Dr. Hazem Qattous

Graduation Project submitted in partial fulfillment for the degree of Bachelor of Science in Software Engineering at Princess Sumaya University of Technology.

Abstract

All over the world and especially in Jordan people face the problem of finding parking spaces in crowded areas, government departments, and educational institutions, particularly during working hours (8 am to 5 pm), on the other hand, some people have private garages in their homes, businesses, etc. which may not be in use by them or by others for long periods, rendering them to be wasted space.

From this problem arose our application which will have two main screens; one for people who own the unused parking space (the “Owners”) and another for people in need of a parking space (the “Tenants”) whereby the Owners post their garages on the marketplace with accompanying pictures and information like if the garage is protected from direct sunlight, does it have a car-charger, what’s the rent per hour, how long will it be free, etc. and the potential Tenants view the Owners’ posts, which are sorted by proximity to their current location, and choose to rent it, pay, then are shown the route on their maps application.

Brief Scenario

There are residential buildings with private garages all over PSUT. For instance, a student who has lectures from 10 a.m. to 4 p.m. will rent and pay for a parking space that’s close and available for these times from an Owner who will be at work during the time stated, he/she will then be given the location to park and when finished both the Owner & Tenant will be asked to give a rating for each other and have an opportunity to write a review.

List of Abbreviations

SDLC: Software Development Life Cycle

SWOT: Strengths, Weaknesses, Opportunities, and Threats

SRS: Software Requirements Specification

UR: User Requirements

FR: Functional Requirements

NFR: Non-Functional Requirements

RMP: Risk Management Plan

WBS: Work Breakdown Structure

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Introduction

Introducing “SwiftSpot”, a mobile application designed to connect users and parking spots. Providing convenience and saving time, this innovative application allows users to search for parking spots in areas that are made available with ease. Providing a platform that connects users with local providers of parking spaces in Amman, Jordan. The application offers real-time information and scheduling options to users, making it a reliable and easy solution for those who require a parking space.

1.2 Project Motivation

The motivation behind “SwiftSpot” is the problem that we have been facing as students at Princess Sumaya University for Technology (PSUT) during the years we received education in it. The unavailability of parking spaces in the university campus and university surroundings, made us come up with this innovative idea to save time and money.

SwiftSpot is driven by the desire to improve access to parking spaces for users, while also utilizing free spaces to create an opportunity to make money which would increase the contribution of the community to solve multiple issues faced in the past years in regards to parking spaces. It also motivates people to stop parking illegally and this will show a great impact on traffic issues.

1.3 Problem Statement

There is always a challenge in regards to finding a parking spot, it’s hard with the rapid increase of cars around Jordan. The main challenge is the amount of time wasted searching for a parking space, especially in highly populated cities such as Amman and Irbid. This is a well known problem among people and businesses, as some businesses face the same issue with parking spaces and it has shown a huge effect on their revenue.

An open shopping market is a great example of how many accidents happen because of illegally parked cars across the whole market and that’s not the only issue, another issue is the amount of traffic generated because of the people looking for parking spaces across the shopping market streets. These types of challenges are faced daily by people in Jordan and solving them using modern technology can show a high increase in satisfaction. The problem expands in shopping centers as well, on peak days such as weekends shoppers may take up so much time to find a parking spot so the alternative option is to find a free space outside the shopping center, which leads to confusion among shoppers and could be costly as the prices aren’t fixed or lead to a parking ticket.

1.4 Aims and Objectives

We aim to develop a user friendly application that offers a solution to the huge parking problem. The main objective is to solve users' problems in finding parking spaces in the city of Amman. By doing so, the mobile application aims to enhance convenience, save time, and contribute to a reduction in the number of vehicles that park illegally. SwiftSpot seeks to provide a reliable and efficient solution to the high parking issues, fostering a more organized and accessible parking environment in the city of Amman and potentially expanding to other areas and regions.

1.5 SWOT Analysis

1.5.1 Strengths

- SwiftSpot provides time-saving solutions for users searching for parking spots. The application's primary strength is in simplifying the often frustrating task of finding parking.
- Users can access up to date data on parking space availability.
- The application's ease of use can be a significant strength, attracting a high amount of users.
- Offering parking solutions across Amman can enhance the application appeal.

1.5.2 Weaknesses

- The success of the application depends on the participation of local parking providers.
- Handling user data and payment requires robust security measures.
- User satisfaction can be a huge issue, as the application works as a third party between the parking provider and the user.

1.5.3 Opportunities

- Promotes sustainable transportation
- Collaborations with local businesses and event organizers can open up opportunities for mutually beneficial partnerships.
- Embracing technologies such as AI for predicting parking availability.

1.5.4 Threats

- Economic downturn may lead to reduced spending on non-essential services.
- Negative reviews shared by users can quickly spread through social media, impacting the application’s reputation.
- If the concept is successful, competitors may quickly enter the market with a similar application.

1.6 Stakeholder Analysis

ID	Stakeholder	Role
S01	Project Supervisor	- Supervises the team and grades their contribution to the project - Authorize main ideas and critical changes
S02	Project Manager	- Monitor and plan the whole project
S03	Team Members	- Document the project - Program the project - Find defects and bugs in the system - Design the user interface - Present the project
S04	Parking Provider	- Approve / Decline users request for parking rental
S05	Users	- Rent parking space
S06	College Management	- Accept / Decline the project
S07	Project Sponsor	- Finance the project
S08	Business Analyst	- Study the market for potential competitors - Research the best approach for investors

Table 1: Stakeholder Analysis

1.7 Expected Output

The expected output is to enhance our client's experience searching for a parking space. It will allow clients to search for an open parking space in a specific area appointed by the client, then the client will be appointed to the parking space via a map third-party application. Clients can create accounts and add their most visited spots to give a quick scan of parking spaces in that area, which will increase the usability of the application. This kind of ease is expected to have a huge increase in user attention and their time efficiency during day to day activities.

Chapter 2: Literature Review

2.1 Literature Review

This literature review aims to explore the existing knowledge on the challenges associated with parking in densely populated areas, with a specific focus on the capital of Jordan. This review will also examine the potential of mobile applications as the solution to the parking issues with individuals and businesses.

2.1.1 Parking Challenges in Urban Environments

The huge growth of vehicles in urban areas, particularly in the city of Amman, has led to a significant increase in the demand for parking spaces. The issue of parking spots does not only result in time wastage for individuals but also poses great challenges for businesses, which shows an effect on their revenue streams. The literature highlights the effects of this issue on daily life, especially in marketplaces and shopping centers.

2.1.2 Impact on Traffic and Safety

Illegally parked vehicles contribute to traffic congestion and safety hazards, especially in open shopping markets. Studies have shown a huge increase in accidents because of illegally parked vehicles. The literature covers the need for efficient parking solutions to fix these issues and enhance overall road safety.

2.1.3 Economic Implications

Businesses, mainly who are located in high traffic areas, are significantly affected by the parking issue. Research that has been conducted shows that there is a correlation between the availability of parking spaces and customer satisfaction. A case has shown that customers may leave or not even pursue to look for a parking space because of laziness or to save time. The literature emphasizes the economic implications of addressing parking challenges for both customers and businesses.

2.1.4 Technological Solutions

As technology continues to advance rapidly, a huge interest has been shown in mobile applications as a solution to modern parking issues. The literature discusses various technological solutions, including important solutions such as real time information and scheduling options offered by mobile applications. These solutions focus on providing users with a convenient, efficient, and easy way to locate and secure parking spaces in busy urban environments.

2.2 Competitor Analysis

2.2.1 Neighbour

This business idea helps users store anything from boxes, furniture, vehicles, or business supplies. This gives them a bigger scope to users, but according to their website, the pricing is high for vehicle spaces which gives us an advantage over them. This competitor does not show a high threat as this business is located in the United States of America.

2.2.2 Parkable

Parkable is an application that can be found on App Store and Google Play. This application can be a strong competitor if it expands to the Middle East, it only targets users that are looking for a parking space and it also supports EV vehicles. The main two advantages that we have are the scheduling option and a better user interface that could highly draw the attention of users. We also have a better understanding of the local market needs and issues faced in today's local community, these needs and issues will be solved frequently which will be costly for our competitor to keep up.

2.2.3 Just Park

This competitor has the most features for vehicle parking spaces, they have lots of options such as the best deal in regards to pricing, the closest to the current location, and a very unique scheduling option. They also have a very good website explaining what their business is, how to use the application, create an account, and rent out your driveway. The main strong advantage that we have is giving more variety to users such as parking spaces in malls, open shopping centers, universities, and hospitals. The Just Park business isn't currently available in the Middle East which gives us time to push our project to users.

2.2.4 Transportation Companies

There are some indirect competitors such as Uber, Careem, and Jeeny. They have nothing similar to our application but they have a very big audience in today's world because they are safe and provide ease of use to users. Users may choose to use these indirect competitors to avoid any complications or to ensure a seamless transportation experience. While Uber, Careem, and Jeeny primarily focus on ride-hailing services, they have cultivated a significant user base and established a reputation for safety and user-friendly interfaces. Marketing strategies should be established to ensure user's acceptance of our application.

Chapter 3: Software Project Management Plan

3.1 Project Charter

Project Title: SwiftSpot
Date of Project Approval: Oct 14, 2023
Project Start Date: Oct 21, 2023 **Project Finish Date:** June 2024

Project Sponsor: Princess Sumaya University for Technology

Project Manager: Omar Obeid (oma20201009@std.psut.edu.jo)

Key Schedule Milestone:

- Develop a SWOT Analysis by October 2023
- Develop a GUI by November 2023
- Develop WBS and a Gantt Chart by December 2023
- Develop SRS by December 2023
- Develop a High-Level Design by January 2023
- Develop a working prototype by March 2023

Budget Information: The budget allocated for this project is 153 USD.

- 25 USD to publish the application on the Play Store
- 99 USD to get the publishing license on the App Store
- 17 USD for the integration with Google Maps
- 12 USD for Figma App Designer

Project Objective: Develop and launch "SwiftSpot," a mobile app addressing the acute parking space shortage in Amman, Jordan. The app aims to provide users with real-time parking information, scheduling options, and a collaborative platform connecting individuals with available parking spaces, thereby improving accessibility, reducing illegal parking, and fostering community engagement.

Project Success Criteria: To develop a functioning application that ensures adoption by users, receives positive user feedback, and reduces unauthorized parking incidents. To ensure improved accessibility and community collaboration.

Project Approach:

- Learn the necessary tools, programming languages, and techniques to implement the project.
- Schedule weekly meetings with the project supervisor.
- Schedule two meetings per week with team members.
- Use Figma to develop the GUI.
- Use Google docs, forms, and drive to manage documentation.
- Use GitHub to share code files with team members.
- Integrate the mobile application with parking space providers.

Roles and Responsibilities:

Name	**Role**	**Responsibility**	**Contact Info**
Dr. Hazem Qattous	Supervisor	Supervise the project	h.qattous@psut.edu.jo
Omar Obeid	Project Manager, Development, Document and Test	Manage team, Developer and Tester	oma20201009@std.psut.edu.jo

Khalil Samara	Development, Test, and Design Team	Developer and Tester	kha20200836@std.psut.edu.jo
Ibrahim Abu Laban	Development, Document, and Design Team	Developer and Designer	ibr20200887@std.psut.edu.jo

Table 2: Project Charter

3.2 Project Scope Statement

Project Title: SwiftSpot
Date of Authorization: Oct 14, 2023
Project Start Date: Oct 21, 2023
Project Finish Date: June 2024
Prepared by:

- Ibrahim Abu Laban
- Omar Obeid
- Khalil Samara

Project Justification: "SwiftSpot" is a mobile application that connects private garage or parking space owners with people willing to rent them. Its purpose is to help mediate the problem of scarce parking spots in Jordan by unlocking the potential of vacant garages, driveways, and private parking areas. Our mobile application aims to optimize parking accessibility, reduce congestion, and promote a more efficient use of urban space.

Project Scope: The application must have the following features.

- Search feature for the user
- Filter feature for the user
- Option to give feedback by the user
- Parking history can be accessed by the user
- Login should be available for the user
- Payment must be available to user
- Parking duration can be specified by the parking provider
- Entering available parking by the parking provider
- Approve/Reject parking provider request by admin

Project Deliverables:

Product-related Deliverables:

- System Requirements Specification (SRS)
- System Design Document (SDD)
- System Test Document (STD)
- Risk Management Plan
- Mobile Application

Project Management-related Deliverables:

- Project Charter
- Project Scope Statement
- Work Breakdown Structure (WBS)
- Software Development Lifecycle (SDLC)
- Gantt Chart
- Risk Management Plan

Project Success Criteria: To submit a correct, functioning, and thoroughly tested mobile application that serves Jordanian citizens when trying to find parking in the nearest location to them especially in congested areas without exceeding the budget and fulfilling the requirements in the SRS by June 2024.

Project Constraints:

- Cost: must not exceed the allocated budget.
- Resources: three team members will be working on the project and one supervisor to supervise the team.
- Risk: to ensure success, the application must perform its main features.
- Scope: the project must remain in the specified scope that's assigned by the Software Engineering department.
- Quality: the application must have quality and high customer satisfaction.
- Time: phase one must be submitted by January 2024 and phase two must be submitted by June 2024.

Table 3: Project Scope Statement

3.3 Software Development Lifecycle (SDLC)

To determine the best path for our project methodology, we researched and compared various software development lifecycles. Strategies included the waterfall, agile, iterative, and incremental strategies.

Since we are both the developers and the customers of this application, we can clearly say that there will be no major changes in the project and that the requirements and initial design are appropriate to the scope and goals that were set initially, the fact that change is an essential part of any software project is completely true, however, if the changes aren't extreme the waterfall SDLC model will still be viable and suitable.

It was determined that the Waterfall model will be used due to the nature of our project. This choice is supported by the project's clearly defined scope and relatively stable requirements. Each phase of the Waterfall model must be finished before going on to the next, and it follows a sequential and linear approach.

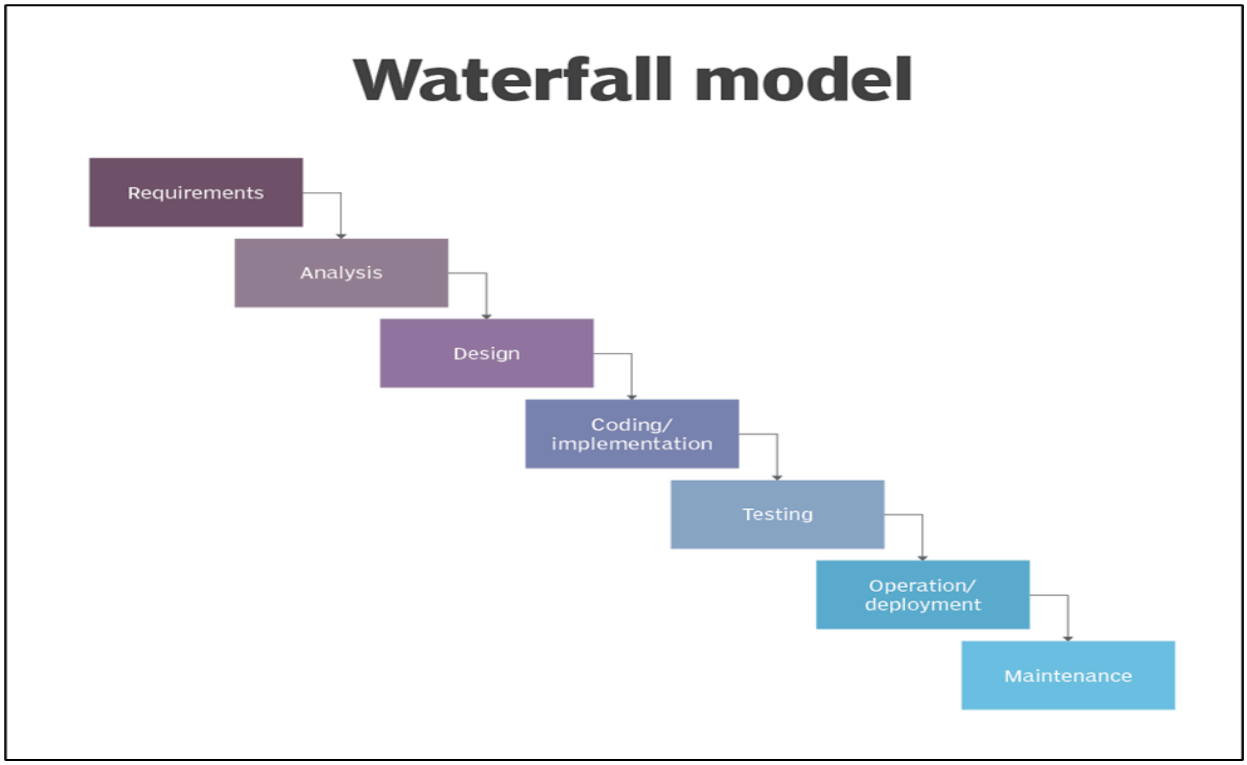


Figure 1: Waterfall Model

The Waterfall model is especially well-suited for projects with clear and fixed requirements, in contrast to the Incremental Iterative technique. It consists of distinct phases, each building on the results of the last, including requirements gathering, design, implementation, testing, deployment, and maintenance.

The Waterfall model ensures a clear comprehension of project requirements and a well-defined development process by providing an organized and systematic framework. The Waterfall strategy gives project timelines and deliverables clarity and predictability by following a linear progression.

The Waterfall strategy, as shown in *Figure 1*, will be implemented with an emphasis on completing each phase before proceeding to the next. This approach matches with the project's consistency in requirements and provides for a comprehensive and well-documented development process. Aiming to produce a final product that precisely and rigidly satisfies the set specifications, the Waterfall model emphasizes a methodical and step-by-step evolution, whereas Incremental Iterative models thrive on flexibility and continual input.

3.4 Work Breakdown Structure (WBS)

Task ID	Task Name	Task Description	Task Duration	Task Allocation	Task Predecessor
Initialization Phase & Research Analysis					
T01	Brainstorming for Project Idea	Finding a solution to a problem we faced in our university years	5 Days	Ibrahim Abulaban, Khalil Samara, Omar Obeid	-
T02	Researching Competitors	Researching current and future competitors	5 Days	Khalil Samara , Ibrahim Abulaban	T01
T03	SWOT Analysis	Define the Strengths, Weaknesses, Opportunities and Threats of the project	1 Day	Omar Obeid	T01
Planning Phase					
T04	Project Charter	Develop a comprehensive project charter, detailing project authorization, project sponsor, project manager, key schedule milestones, budget information, objectives, and responsibilities of team members	2 days	Omar Obeid	T02
T05	Software Development Lifecycle	Process of planning, creating, testing, and deploying software systems effectively.	2 Days	Khalil Samara, Omar Obeid	T04
T06	Project Scope Statement	Develop a comprehensive project scope statement, detailing the project start and finish date, justification, deliverables, success criteria, and constraints	2 Days	Khalil Samara , Ibrahim Abulaban	T04
T07	Work Breakdown Structure	Develop WBS to identify task duration and allocation	4 Days	Omar Obeid	T04, T06
T08	Gantt Chart	Develop a Gantt chart to visualize activities that are created in WBS	3 Days	Khalil Samara, Omar Obeid	T07
T09	Risk Management Plan	Identify risks and the impacts of each risk.	5 Days	Omar Obeid	T06, T07

Analyzing Phase					
T10	Feasibility Study	Thorough analysis to evaluate if a project is practical and economically viable.	3 Days	Omar Obeid	T01
T11	Requirements Elicitation Technique	Conduct surveys, interviews, and focus groups to identify requirements.	5 Days	Omar Obeid	T10
T12	Specify Requirements	Identify system and user requirements of the system	7 Days	Khalil Samara, Omar Obeid	T11
T13	Requirements Interdependency Matrix	Detailing relationships and dependencies among project requirements for comprehensive understanding.	4 Days	Khalil Samara, Omar Obeid	T12
T14	Prioritize Requirements	Prioritize the most important requirements based on the system.	4 Days	Omar Obeid	T12
T15	Use Case Diagram	Visual representation illustrating interactions between actors and a system's use cases.	5 Days	Khalil Samara, Omar Obeid	T12
T16	Activity Diagram	Graphical representation showing workflows and activities of the system.	5 Days	Khalil Samara, Ibrahim Abulaban	T12, T15
Designing Phase					
T17	High Level Design	An overview detailing the architecture and components of a software system.	3 Days	Ibrahim Abulaban	T12
T18	User Interface Diagram	Visual representation illustrating the layout and interaction of user elements.	4 Days	Khalil Samara	T15
T19	Sequence Diagram	Graphical representation illustrating interactions and order of system components or objects.	5 Days	Khalil Samara, Omar Obeid	T12, T17
T20	Class Diagram	Visual representation of system classes and their relationships.	5 Days	Khalil Samara	T12, T15
T21	Graphical User Interface	Visual interface for user interaction with software applications.	14 Days	Ibrahim Abulaban, Omar Obeid	T12, T19

Implementation Phase					
T22	Develop Flutter Mobile App	Implement the Specify Requirements to develop a mobile application.	115 Days	Ibrahim Abulaban, Khalil Samara, Omar Obeid	T12, T21
Testing Phase					
T23	Create Test Plan	A document outlining strategies for software testing and quality assurance processes.	1 Day	Khalil Samara, Omar Obeid	T22
T24	White Box Test	Examining internal structures and code logic for thorough testing.	3 Days	Khalil Samara	T23
T25	Black Box Test	Assessing the software's functionality without knowledge of its internal code.	2 Days	Omar Obeid	T23
T26	Ad Hoc Test	Informal and unplanned testing without predefined test cases.	2 Days	Omar Obeid	T23
T27	Performance Test	Assessing the system's responsiveness, speed, and overall performance.	2 Days	Khalil Samara	T23
T28	Integration Test	Verifying interactions between integrated components or systems.	1 Day	Ibrahim Abulaban	T23
T29	Security Test	Using the Microsoft Threat Modeling tool to identify threats in the application	1 Day	Omar Obeid	T23
Deployment Phase					
T30	Execute Deployment Plan	Implement a deployment plan to ensure smooth system transition and functionality.	1 Day	Ibrahim Abulaban, Khalil Samara, Omar Obeid	T22, T23

Table 4: Work Breakdown Structure

3.5 Gantt Chart

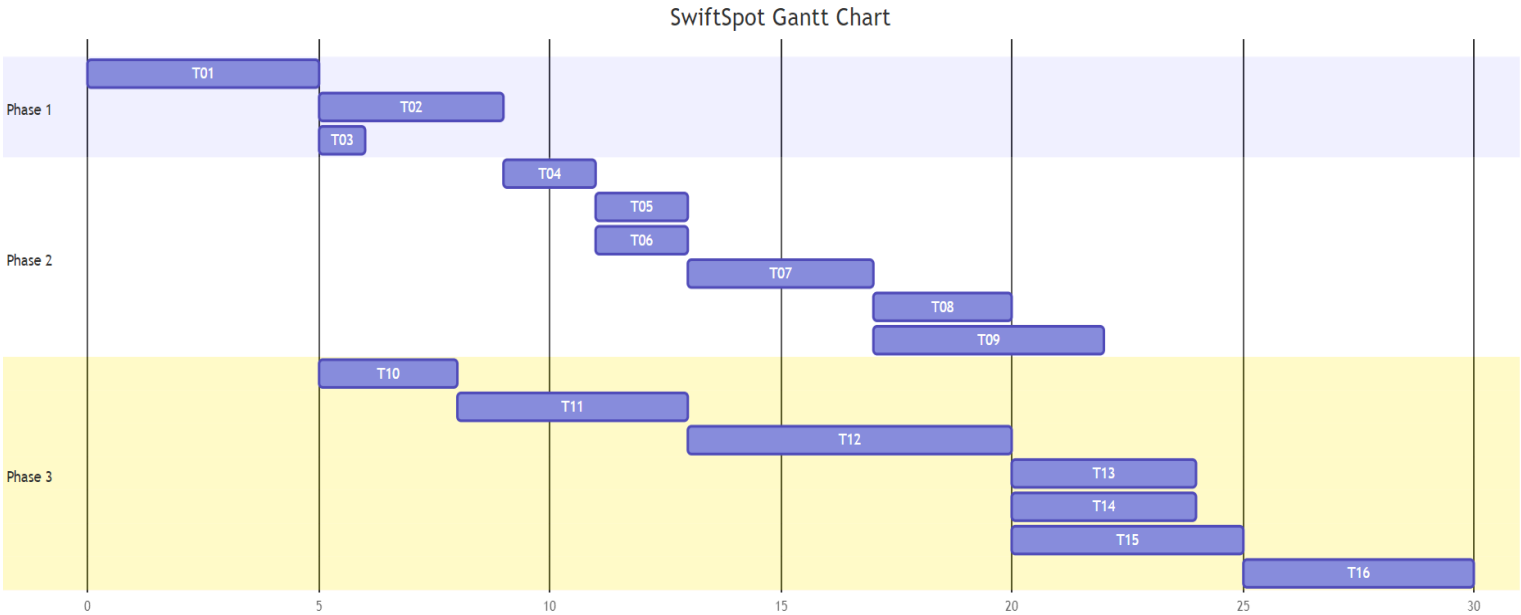


Figure 2: WBS illustrated using Gantt Chart from Phase 1 to 3

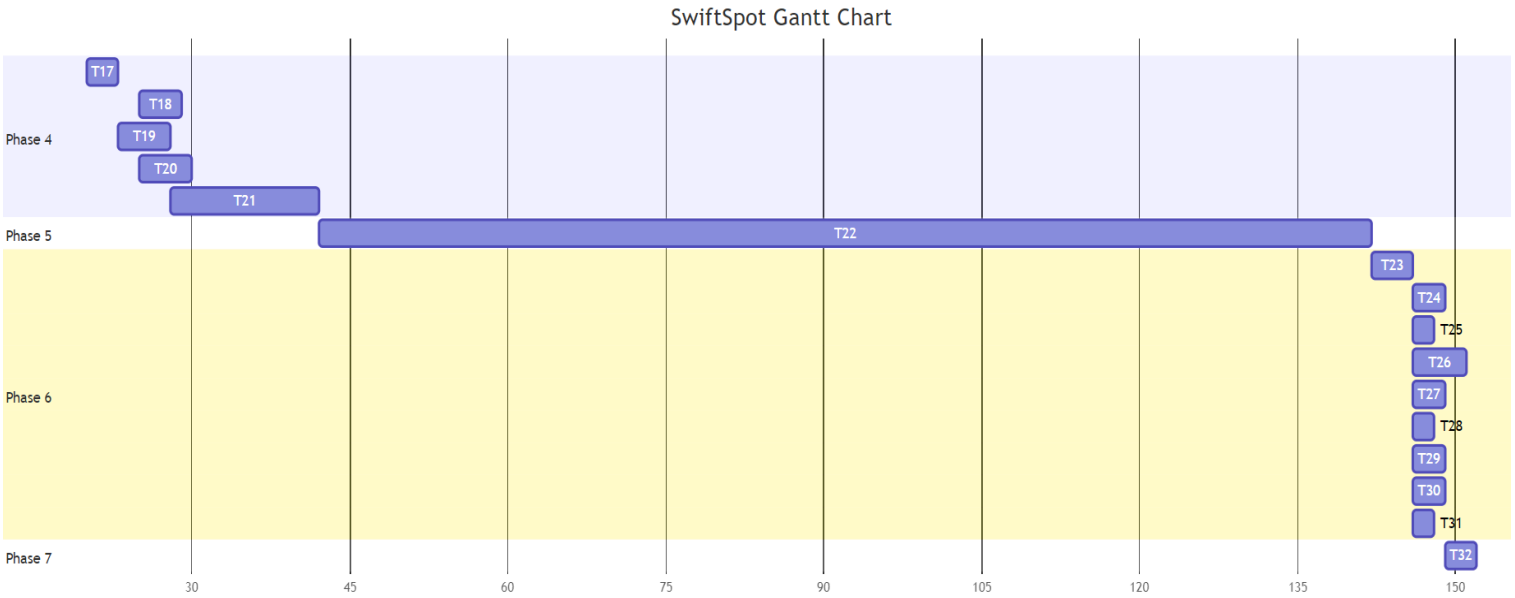


Figure 3: WBS illustrated using Gantt Chart from Phase 4 to 7

3.6 Project Management Tools

Tools	Description
FIGMA	A website for creating the user interface design.
Google Docs	Creating and sharing the documentation of the project among team members.
Google Drive	Sharing and accessing files among team members.
Google Forms	Creating a survey and disturbing it to create a study among people.
GitHub	Hosting the graduation project codes for control and collaboration.
Evernote	Note-taking and task management platform.
Discord	A platform used for live streaming and online conferences with team members.
Gmail	An email platform is used to schedule and communicate with our supervisor.
Draw.io	A tool used to draw the diagrams that are required for the project.
Zoom	A platform used for online conferences to communicate with our supervisor.

Table 5: Project Management Tools

3.7 Risk Management Plan

3.7.1 Risk Identification

Risk ID	Risk Name	Risk Description	Risk Category
R01	Project Deadline	Failing to finish the project according to the assigned deadline given by the software engineering department.	Business
R02	Financial Risk	Failing to stay within the allocated budget.	Business
R03	Legal Risk	Failure to comply with information protection could result in legal action.	Business
R04	Dependency Risk	Relying on third party dependencies that may fail or experience issues.	Business
R05	Market Risk	The application may not meet the needs of the target audience, resulting in low user adoption.	Business
R06	Display Parking Spaces	Failing to display parking spaces to the user.	Product
R07	Search for Parking Spaces	Failing to allow the user to search for parking spaces in a specific area.	Product
R08	Filtering Parking Spaces	Failure to create a filter feature to make searching easier for the user.	Product
R09	Displaying Parking Time	Failing to display the duration of the user's vehicle parking time.	Product
R10	Feedback	Failure to integrate feedback feature.	Product
R11	Applying Fines	The application must apply fines to users who exceed the time limit while parking their vehicles.	Product
R12	Parking Provider Privileges	Failing to give privileges to the parking provider such as time of parking for every vehicle and the amount of vehicles that can park at once.	Product
R13	Parking Provider Approval	Failure to approve for parking providers to accept/decline vehicles.	Product

R14	Barcode Scan/ Code Generator	Failing to implement a barcode scan or a code generator to indicate the starting time of the user's use of parking space.	Product
R15	Application Submit	Failing to submit a working application	Product
R16	Application Design	Failure to create an eye catching and easy-of-use design for users and parking providers.	Design
R17	Losing Team Member	Risk of losing a team member.	Organizational
R18	Conflict Among Team	The threat of a conflict that may appear among team members that can't be resolved.	Organizational
R19	Inexperienced Team Member	Risk of inexperienced team members to deliver the documentation or application within the project scope.	Organizational
R20	Packages Import	Failing to import packages that must be used in the system.	Technical
R21	System Integration	Failing to integrate the system with Google Maps and other integration systems.	Technical

Table 6: Risk Identification

3.7.2 Risk Analysis

3.7.2.1 Risk Probability-Impact Metric

Category	Probability Chance	Impact Description
Low	< 50% to occur	The project activities will not be interrupted, but the resolution of the risk will be addressed at a later time.
Medium	50% to occur	Project activities will come to a halt if the underlying risk is not resolved.
High	> 50% to occur	Project activities will come to a halt until a solution is implemented.

Table 7: Risk Probability-Impact Metric

3.7.2.2 Risk Impact/Probability

Impact	Probability		
	Low	Medium	High
Low	R04		
Medium	R03, R17	R07, R10	R14, R20
High	R02, R06, R09, R13, R15	R01, R05, R08, R11, R12, R21	R16, R18, R19

Table 8: Risk Impact/Probability

3.7.2.3 Risk Priority

Risk ID	Probability	Impact	Priority
R01	Medium	High	High
R02	Low	High	Medium
R03	Low	Medium	Medium
R04	Low	Low	Low
R05	Medium	High	High
R06	Low	High	Medium
R07	Medium	Medium	Medium
R08	Medium	High	High
R09	Low	High	Medium
R10	Medium	Medium	Medium
R11	Medium	High	High
R12	Medium	High	High
R13	Low	High	Medium
R14	High	Medium	High
R15	Low	High	Medium
R16	High	High	High
R17	Low	Medium	Medium
R18	High	High	High
R19	High	High	High
R20	High	Medium	High
R21	Medium	High	High

Table 9: Risk Priority

3.7.3 Risk Response

To address any potential risks associated with "SwiftSpot," our mobile application linking users with parking spaces, we will implement a robust risk response plan. Rigorous testing and quality assurance will be employed to ensure the application's reliability, tackling glitches or interruptions before its launch. Privacy considerations will be prioritized through regular security audits, aligning with regulations to foster user trust. Clear and detailed agreements with local parking providers, coupled with ongoing communication and contingency plans, will mitigate potential partnership disruptions. User satisfaction takes priority, involving continuous testing, feedback systems, and targeted marketing efforts. In the competitive landscape, SwiftSpot aims to maintain its edge by monitoring industry trends, innovating features, and strategically positioning itself through marketing initiatives and partnerships.

3.7.3.1 Risk Response Actions

Response Action	Response Action Definition
Avoidance	Change or eliminate the project plan to reduce the likelihood or impact of the risk.
Mitigation	Take action to reduce the likelihood or impact of the risk without eliminating it.
Transfer	Shift the impact of the risk to another party, such as outsourcing or contracts.
Acceptance	Acknowledge the risk and decide not to take action to address it.

Table 10: Risk Response Action Types

3.7.3.2 Risk Response Strategy

Risk ID	Response Action	Response Description
R01	Mitigation	Reduction of deadline risk is ensured by early risk reduction, resource optimization, updated schedule, and ongoing monitoring.
R02	Mitigation/Transfer	Adopt measurements to reduce financial risk and financial risk and assign accountability when it's practical.
R03	Transfer	Convey legal risk to an external party through contracts and insurance.
R04	Mitigation	Limiting the dependency on the allocated party to decrease issues and failure.
R05	Transfer	Assign high measures to the allocated party to meet the marketing requirements and goals.
R06	Mitigation	Allocate more resources and time to the specific functionality, seeking professional help and providing training sessions.
R07	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R08	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R09	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R10	Acceptance	Accept the issue and apply a change on the next release.
R11	Mitigation	Allocate more resources and time to the specific functionality, seeking professional help and providing training sessions.
R12	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R13	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R14	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.

R15	Mitigation	Allocate more resources and time to the functionality, seek professional help, and provide training sessions.
R16	Mitigation	Conduct surveys, focus groups, and interviews. Prioritize requirements by using the SDLC approach.
R17	Acceptance/Mitigation	Accept the issue and divide the teamwork to ensure the project continues smoothly and on track.
R18	Mitigation	Identify the point of conflict and assign the project manager to resolve the conflict with minimal losses.
R19	Mitigation	Provide training sessions and seek professional help.
R20	Mitigation	Conduct thorough testing, use alternative packages and monitoring to identify issues.
R21	Mitigation	Conduct thorough testing, use alternative integration options and monitoring to identify issues.

Table 11: Risk Response Strategy

Chapter 4: System Requirements Specifications

4.1 Feasibility Study

4.1.1 Technical Feasibility

SwiftSpot will operate on both IOS and Android by using the Flutter programming language. Figma design website will be used to design high level, low level, and UI/UX designs.

4.1.2 Operational Feasibility

SwiftSpot is a user-friendly application that eases the process of searching for a parking space. The application is highly dependent on user adoption and the involvement of parking providers since it saves time and money for users to ensure their comfort and convenience while finding parking spaces. The system involves users from the beginning by conducting surveys and interviews to understand their needs and concerns. Taking into consideration that this is the first system in Jordan to solve such an issue, it has been roughly studied.

4.1.3 Financial Feasibility

Financially, our project won't have a high cost because it will be implemented remotely from the team member's laptops. The only financial considerations are getting the warrants, Google Maps integration, third party payment fee, deployment fee for IOS and Android, and domain name registry.

Financial Expense	Price
GoDaddy Domain Name	$70
Google Maps Integration	$17 for every 100,00 searches
IOS Deployment	$99
Android Deployment	$26
Warrants	Unknown until the deployment of the project
Figma	$12 per month

Table 12: Financial Expense Breakdown

4.1.4 Schedule Feasibility

The project will take approximately 32 weeks to be completed. It was approved on the 14th of October 2023 and is expected to be finished around June 2024. In regards to risks, we have developed an extensive risk management plan that includes a response strategy that will reduce any factors that may cause delays in our scheduled timeline. As scheduled in the Work Breakdown Structure the Initialization Phase & Research Analysis Phase will take 11 days to complete taking into consideration that there aren't any delays. The Planning Phase will take 18 days to complete taking into consideration that there aren't any delays. The Analyzing Phase will take 33 days to complete and the Designing Phase will take 31 days to complete if there aren't any delays. The Implementation Phase will take 100 days until it's fully completed taking into consideration that there aren't any delays. At last, the Testing Phase will take 27 days to complete and the Deployment Phase will take 3 days to complete without any delays. If any issues are allocated we will implement the requirements prioritization first, then gradually implement the remaining requirements.

Phase Name	Days to Complete
Initialization Phase & Research Analysis Phase	11 Days
Planning Phase	18 Days
Analyzing Phase	33 Days
Designing Phase	31 Days
Implementation Phase	115 Days
Testing Phase	13 Days
Deployment Phase	3 Days

Table 13: Phase Schedule Breakdown

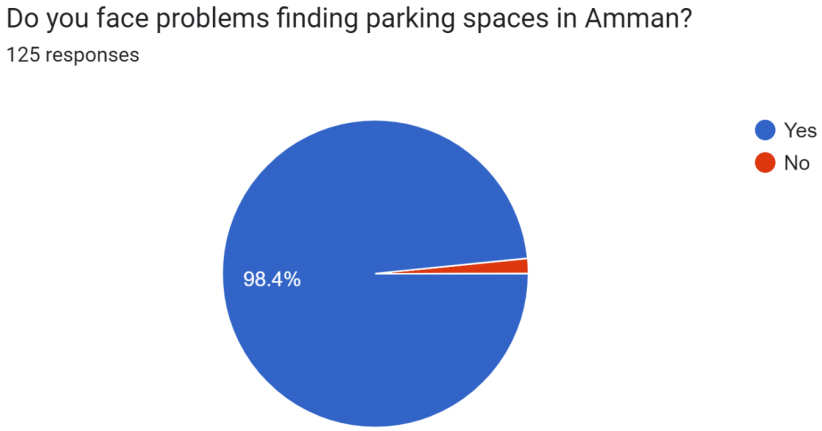
4.2 Requirements Elicitation

The main requirement elicitation techniques that were used were survey and interviews. Using this kind of technique allows us to gather the user requirements and address the main issues faced by users.

4.2.1 Survey

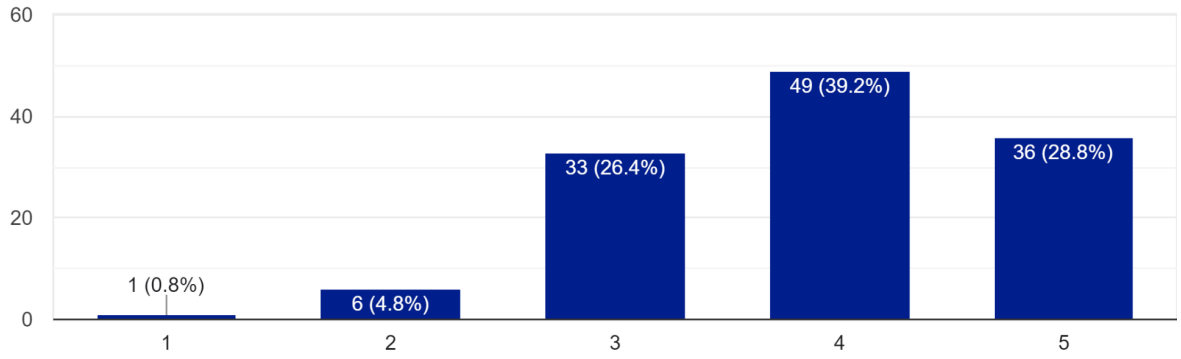
We used a survey to gather quantitative data that helped us present our user needs and wants for our application. We aligned our user requirements with the most important features selected by users, it confirmed the features that are most wanted by users and their priority.

The result of the survey is shown below:



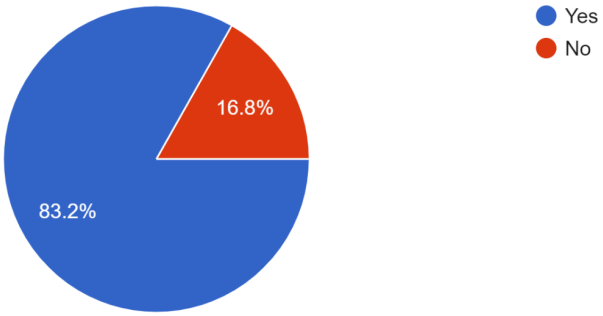
On scale of 1 to 5, how severe is the parking issue in Amman?

125 responses



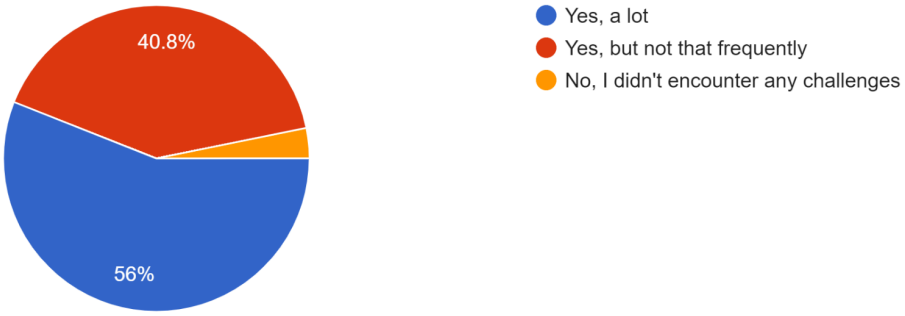
To the best of your knowledge, do you think one of the main reasons that there is a huge traffic issue in Amman is because of the unavailability of parking spaces?

125 responses



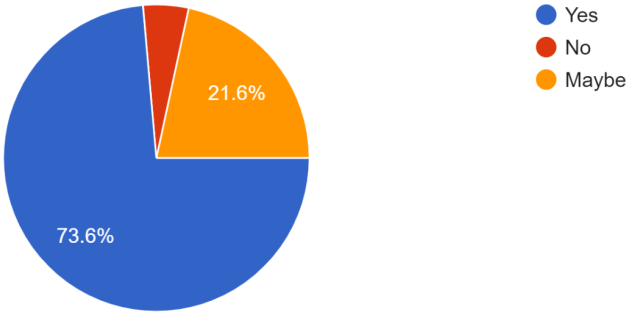
In Amman, have you ever encountered a challenge finding a parking space when visiting popular locations such as malls, shopping centers, or hospitals?

125 responses



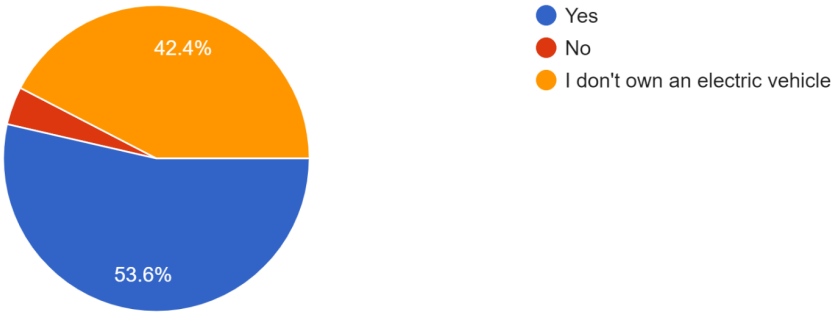
If a mobile application is created to solve the parking issue in Amman would you be interested in using it?

125 responses



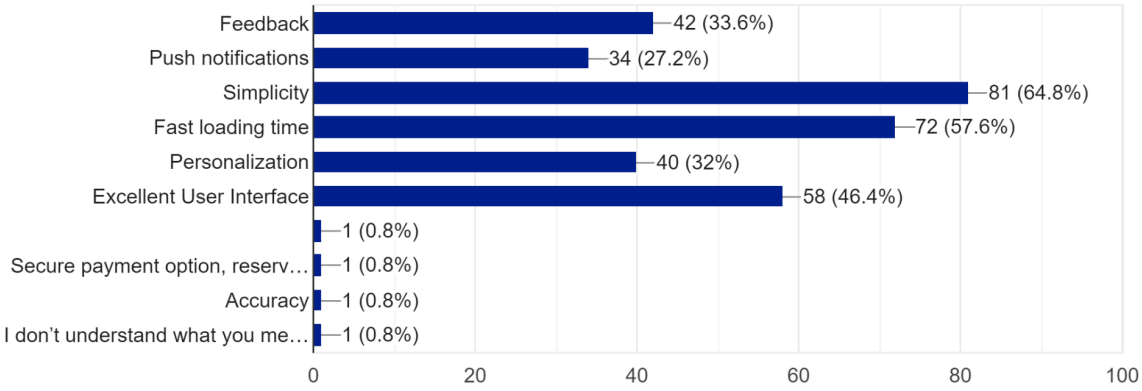
As an electric vehicle owner, would you appreciate the option to have a charging station available for your vehicle?

125 responses



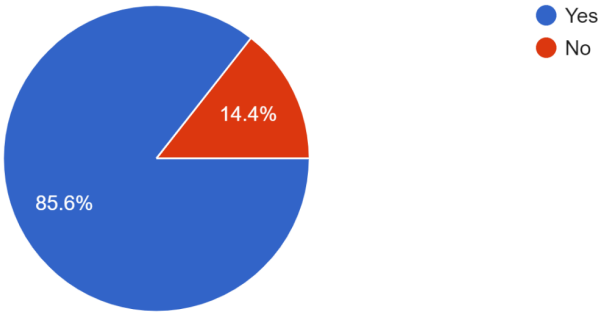
What are the most important features that would you like to be added in the mobile application?

125 responses



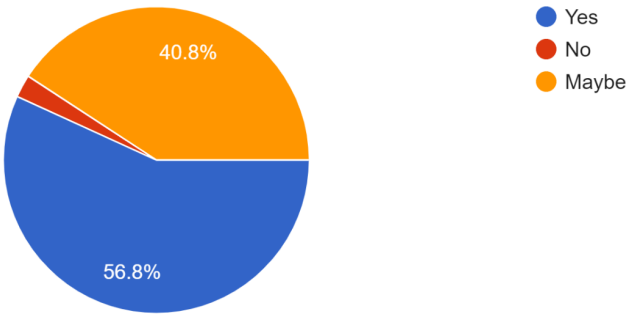
Do you believe it is appropriate to impose fines on individuals who consistently disobey the rules? (e.g. If a user exceeds the scheduled parking time, a fine applies.)

125 responses



Do you believe the introduction of this application could significantly contribute to resolving the parking challenges in Amman?

125 responses



4.2.2 Interviews

We have conducted six interviews with users to ensure that the best user requirements are provided. The interviews were conducted using a series of 11 questions and an approximate duration of 10 minutes.

4.2.2.1 Interview Questions

- **Introduction**
  - Name?
  - Age?
  - Profession?
  - How long have you lived in Amman, Jordan?
- **Questions**
  - Do you face any challenges in finding parking spaces in Amman?
  - What are the challenges that you usually face?
  - To the best of your knowledge, what is the main issue of unavailability of parking spaces in Amman?
  - How often do you find yourself in need of a parking space?
  - Are there any safety considerations while parking in unfamiliar parking spots?
- **Conclusion**
  - Do you believe the introduction of SwiftSpot will significantly contribute to resolving the parking challenges in Amman?

4.2.2.2 Interview Results

After conducting interviews with users, we were able to understand really important information about user's preferences and requirements. This enabled us to reach the required requirements that ensure user satisfaction and comfort. We were also able to address the most common issues that they have witnessed and faced, which simultaneously helped in prioritizing their needs and determining additional requirements. Interviews also helped us gain more insight into the project and its positive effect on the community.

4.3 User Requirements

UR-01 The user shall be able to search for a parking space.

UR-02 The user shall be able to filter parking spaces.

UR-03 The user shall be able to find the nearest parking space.

UR-04 The user shall be able to cancel their request for a parking space.

UR-05 The user shall be able to set location preferences.

UR-06 The user shall be able to log into the application using their account.

UR-07 The user shall be able to edit their profile information.

UR-08 The user shall be able to give feedback on their experience.

UR-09 The user shall be able to request additional time for parking if needed.

UR-10 The user shall be able to access their parking history.

UR-11 The user shall be able to register into the application.

4.4 System Requirements

4.4.1 Functional Requirements

FR-01 The system shall have a splash screen with the logo displayed.

FR-02 The system shall have an onboarding screen with three distinct instructions.

FR-03 The user shall be able to log into the system using their email and password.

FR-04 The user shall be able to register a new account.

FR-04.1 The system shall allow the user to register by entering their email, name, password, and mobile number.

FR-04.2 The system shall authenticate newly registered users using OTP through the given phone number.

FR-04.3 The system shall ask for the user's permission to access their current location or to set their location manually.

FR-04.4 The system shall ask the user to choose the language.

FR-05 The system shall display a header on the home screen.

FR-05.1 The user shall be able to search by text.

FR-05.2 The user shall be able to filter their search results.

FR-05.2.1 The user should be able to filter by available today, popularity, star ratings, price, and duration.

FR-06 The system shall display a list of available parking spaces sorted by their distance from the user on the home screen.

FR-07 The system shall display all available details of a parking space on a separate screen that is shown when the user taps on a parking space item.

FR-08 The system shall allow the user to enter the date, time, duration, and if their vehicle is an EV of their parking.

FR-9 The system shall allow the user to give feedback on their parking experience using the five-star rating and a written review.

FR-10 The system shall issue a receipt to the user after they are done with the parking.

FR-11 The system shall allow the user to add and remove payment methods.

FR-12 The system shall display the time left to the user in the spot they're parked in.

FR-13 The system shall allow users to view details of both their completed and pending bookings.

FR-14 The system shall allow the user to add a parking space to their favorites list.

FR-15 The system shall show the history of notifications sent to the user in a separate notification screen.

FR-16 The user shall be able to access their profile screen.

FR-16.1 The user should be able to change their password, profile picture, or system language if needed.

FR-16.2 The user shall be able to sign out of their account.

FR-17 The user should be able to apply to become a parking provider.

FR-17.1 The user shall enter registration information in the application.

FR-17.2 The application shall be sent to the administrator.

FR-18 The parking provider shall be able to add, edit, and delete parking spaces.

FR-19 The parking provider shall specify if the parking space supports EV vehicles.

FR-20 The parking provider shall be able to review the user ratings and comments.

4.4.2 Non Functional Requirements

NFR-01 The system shall operate on Android and IOS devices.

NFR-02 The system shall not exceed 3 seconds of load time.

NFR-03 The system shall be available 99% of its run-time.

NFR-04 The system shall store passwords using a hashing algorithm.

NFR-05 The system shall be easily understandable for a new user within an hour of learning.

NFR-06 The system shall employ end-to-end encryption on all its transactions.

NFR-07 The system shall be able to handle 35000 transactions per day.

4.5 Interdependency Matrix

Requirement ID	Refined to	Change to	Similar to	Requires	Conflicts with	Increase cost of	Decrease cost of	Increase value of	Decrease value of
FR-01			**FR-02**						
FR-02	**FR-01**								
FR-03				**FR-04**					
FR-04	**FR-03,** **FR-04.1** **FR-04.2** **FR-04.3** **FR-04.4**		**FR-17**						
FR-04.1								**FR-04.2**	
FR-04.2				**FR-04.1**					
FR-04.3									
FR-04.4									
FR-05	**FR-05.1** **FR-05.2** **FR-05.2.1**								
FR-05.1									

FR-05.2	**FR-05.2.1**							**FR-05.1**	
FR-05.2.1									
FR-06			**FR-07**						
FR-07								**FR-06**	
FR-08									
FR-09								**FR-06**	
FR-10									
FR-11				**FR-10**					
FR-12				**FR-11**					
FR-13									
FR-14									
FR-15			**FR-13**						
FR-16	**FR-16.1** **FR-16.2**								
FR-16.1									
FR-16.2									
FR-17	**FR-17.1** **FR-17.2**								
FR-17.1									
FR-17.2				**FR-17.1**					
FR-18				**FR-17**					

Table 14: Interdependency Matrix

4.6 Requirements Prioritization

For our project we have chosen to prioritize our requirements by using the **Top-Ten Requirements** approach is a prioritization strategy often used in scenarios with multiple stakeholders with varying priorities. It simplifies decision making by focusing on the ten most crucial requirements per stakeholder, minimizing conflicts, and balancing different stakeholder interests. This strategy ensures the most critical functionalities are developed and delivered, acting as a tool for negotiation in resource-constrained situations.

Top-Ten Requirements

FR-03: The system must allow the user to login so they can use the features in the application.

FR-04: The user's ability to register is the most important requirement because all the other requirements and the whole system depend on it.

FR-05: This requirement is important because it allows the user to search for a parking space which is the main foundation of the system.

FR-06: This functionality is important because it increases the ease of use of the system.

FR-08: This is a really important feature for the user because not all users need the same duration for a parking space.

FR-11: This is essential for ensuring multiple financial transactions are available for the user.

FR-12: This is a really important requirement because the user must be aware of the time left for the duration of the parking to have the option of extending or ending the rental process.

FR-16: This allows the user to apply change to their account if needed which is necessary because it happens a lot.

FR-17: This is an essential requirement because providing a parking space is the key aspect of the system.

FR-18: This is really important to ensure that the parking provider can share their parking spaces and edit the required settings if needed.

Chapter 5: System Design & Architecture

5.1 Context Diagram

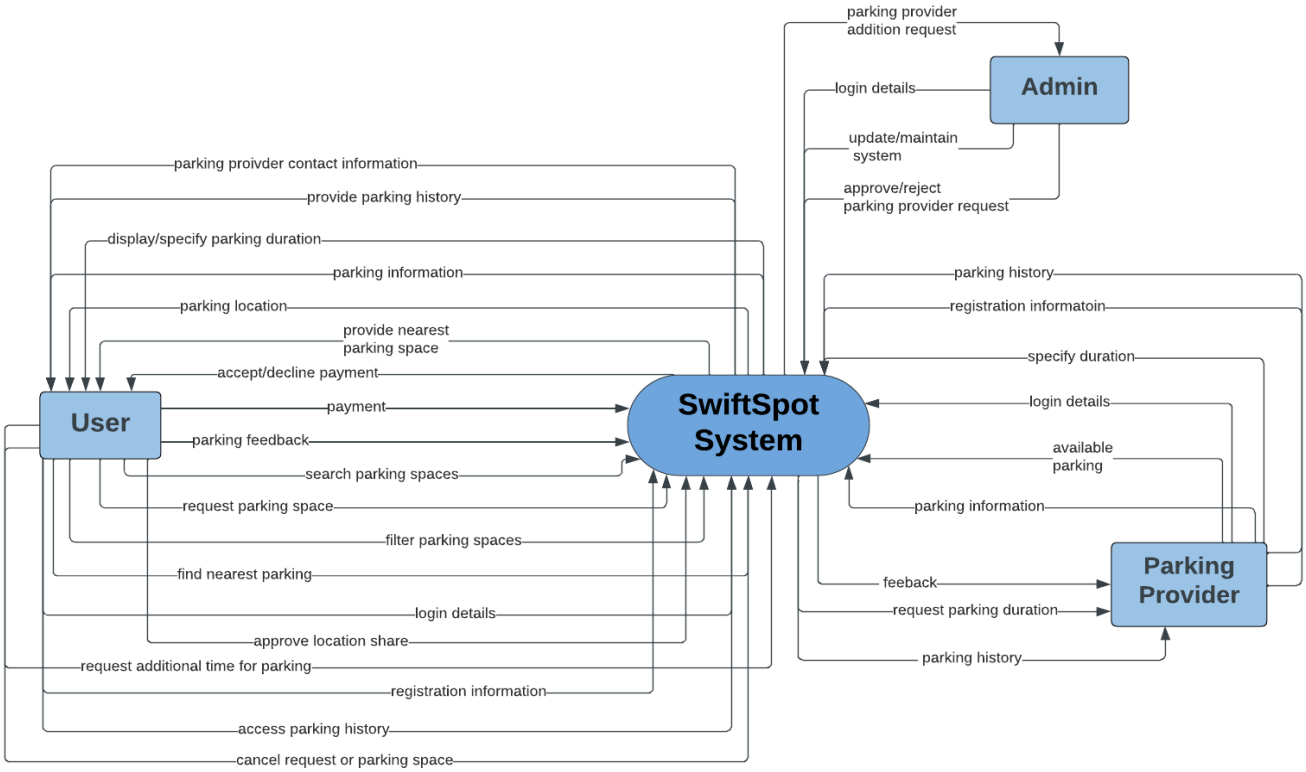


Figure 4: Context Diagram

To illustrate our system we chose a context diagram to represent the boundaries and interactions between our system and their interacting entities as shown in *Figure 4*. Our SwiftSpot System will interact with the SwiftSpot user entity by providing the entity with data such as providing the nearest parking space, displaying/specifying parking duration, accept/decline payment, parking provider contact information, and parking location. The SwiftSpot user will return their approval to share location, login details, filter parking spaces, search parking spaces, parking feedback, request additional time for parking, and their registration information. The SwiftSpot system will also interact with parking provider entities such as user feedback, request parking space, request parking duration, and user contact information. In return, the parking provider will provide data such as parking information, available parking spaces, accept/decline requests, login details, registration information, and specified duration. The final entity that will interact with the SwiftSpot system is the Admin entity which requests access for adding a parking provider. In return, the Admin will provide the SwiftSpot system with the login details, update and maintain the system, and approve/reject parking provider requests.

5.2 Use Case Diagram

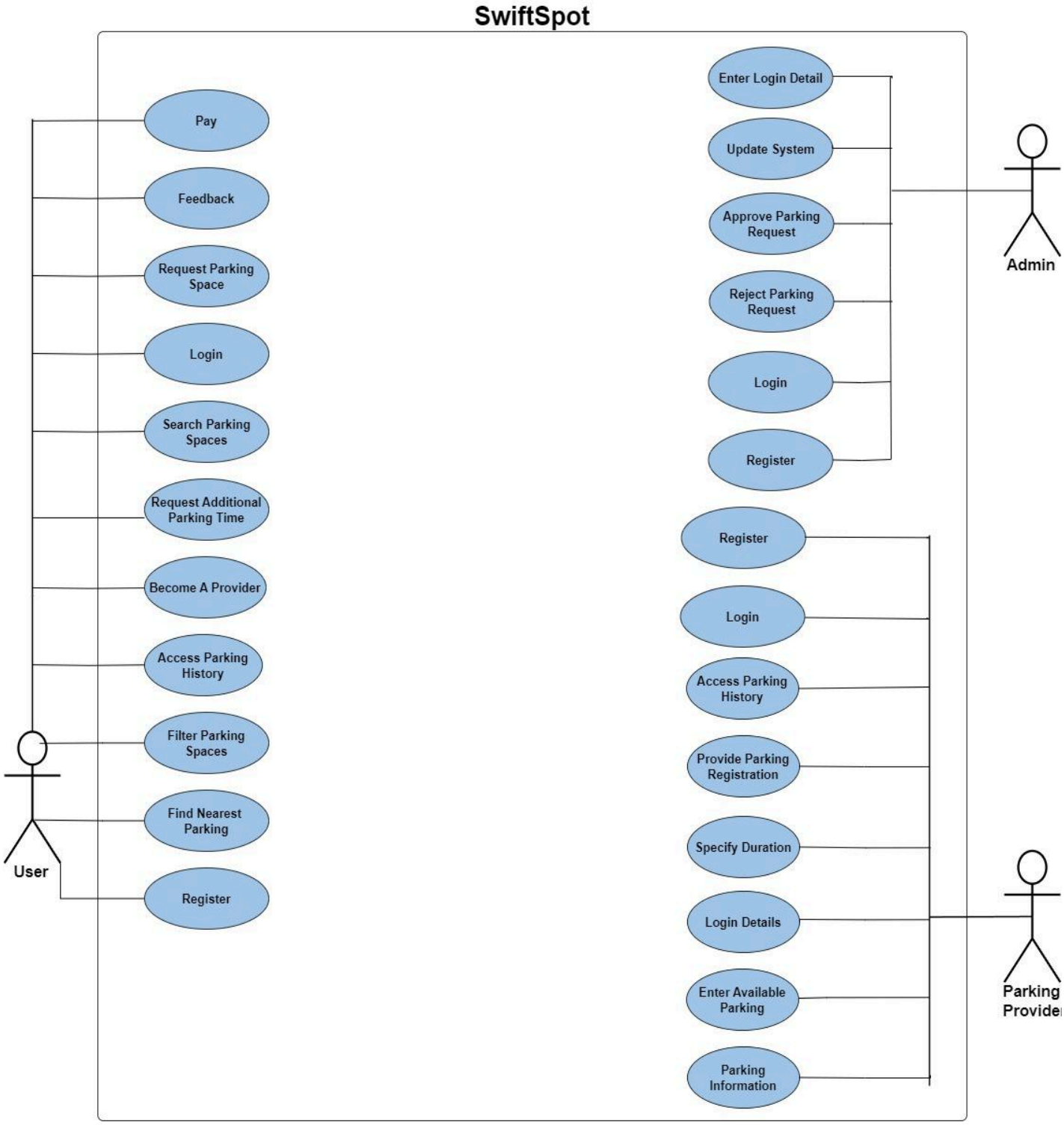


Figure 5: Use Case Diagram

5.3 Activity Diagram

5.3.1 Become a Provider

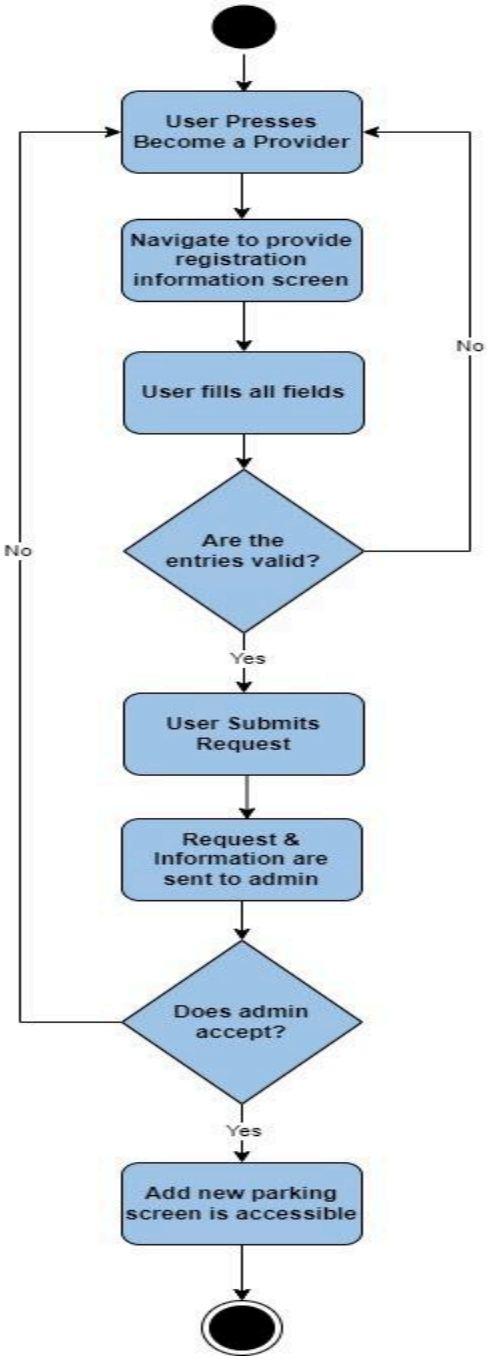


Figure 6: User option to become a Provider

5.3.2 User Search Activity

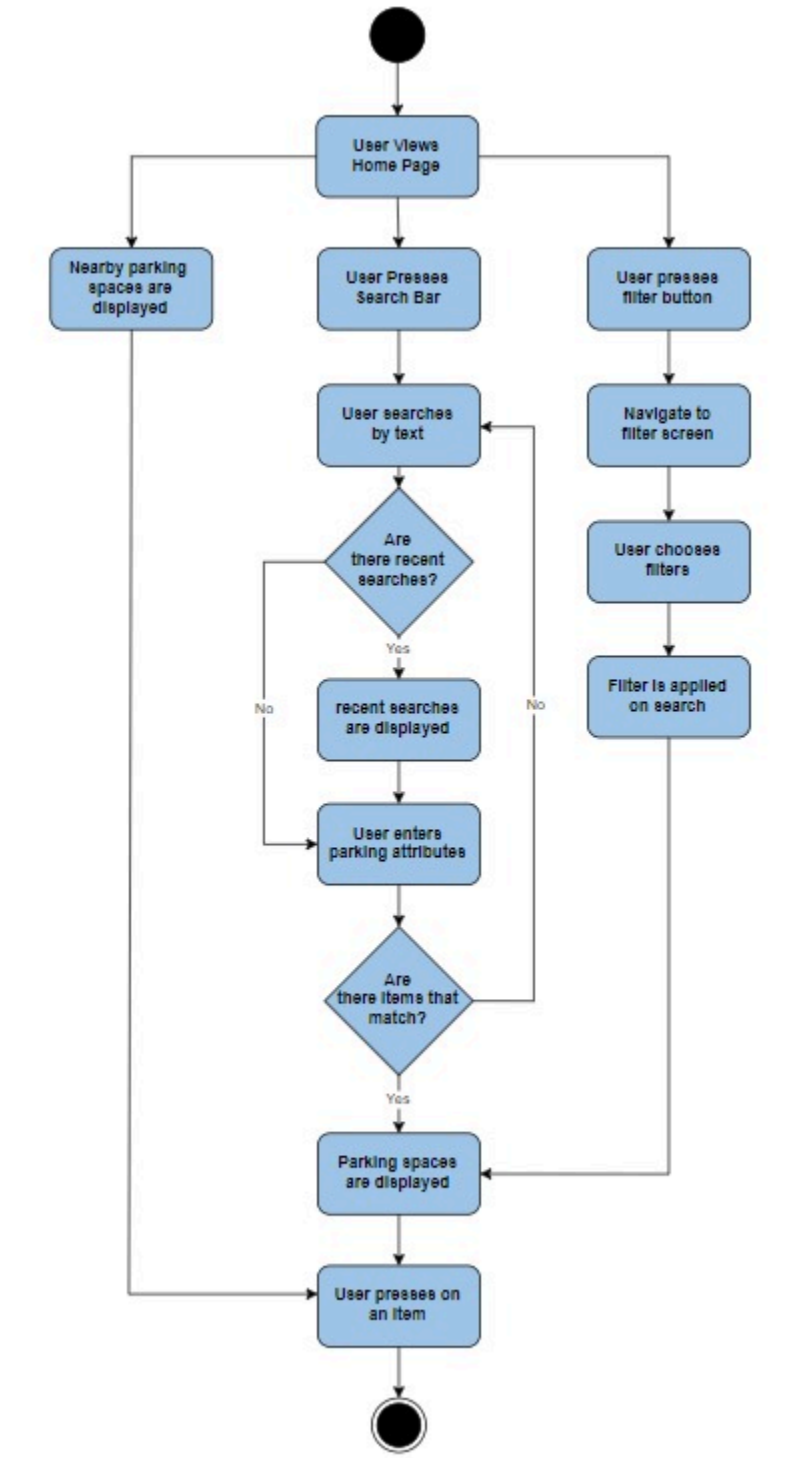


Figure 7: Search Activity for User

5.3.3 Access Parking History

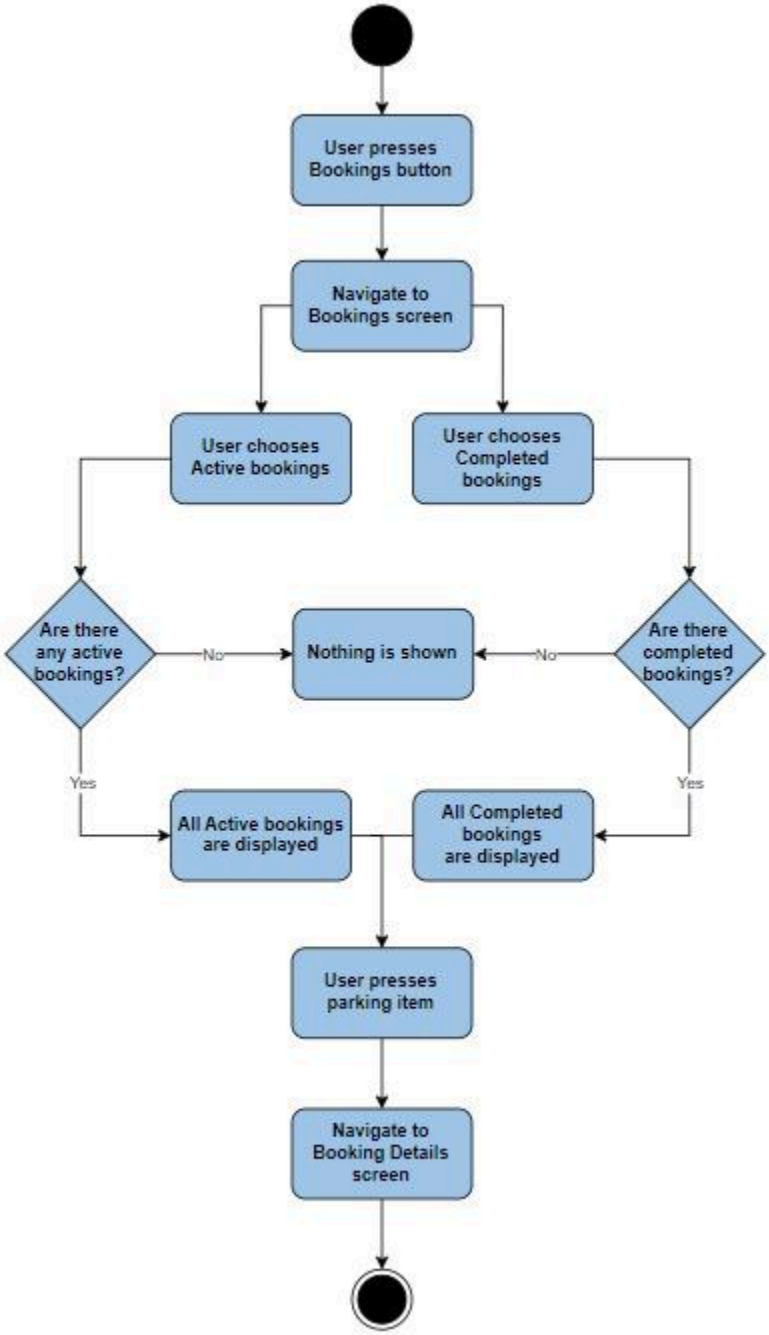


Figure 8: Access Parking History

5.3.4 Booking Parking Space

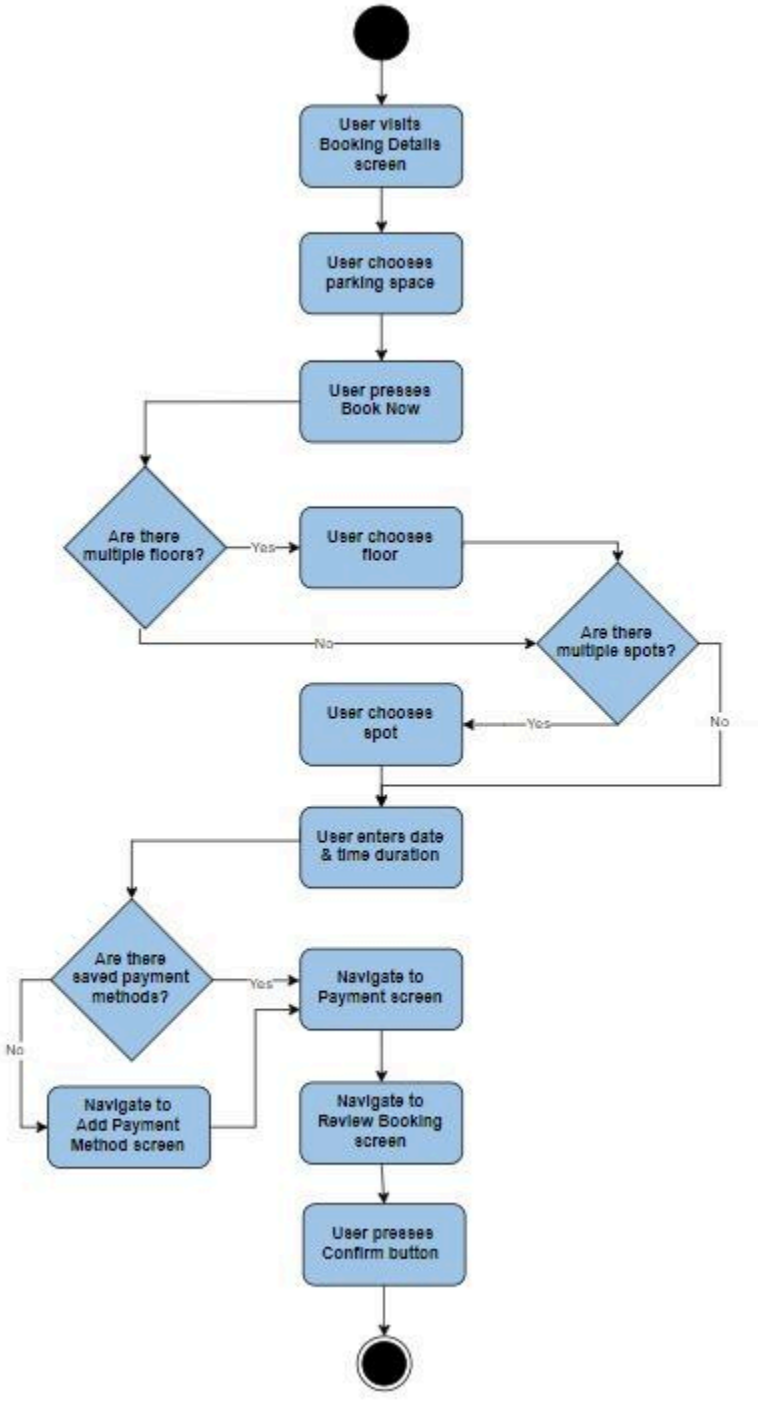


Figure 9: User Booking Parking Space

5.3.5 Admin Accept Parking Provider Request

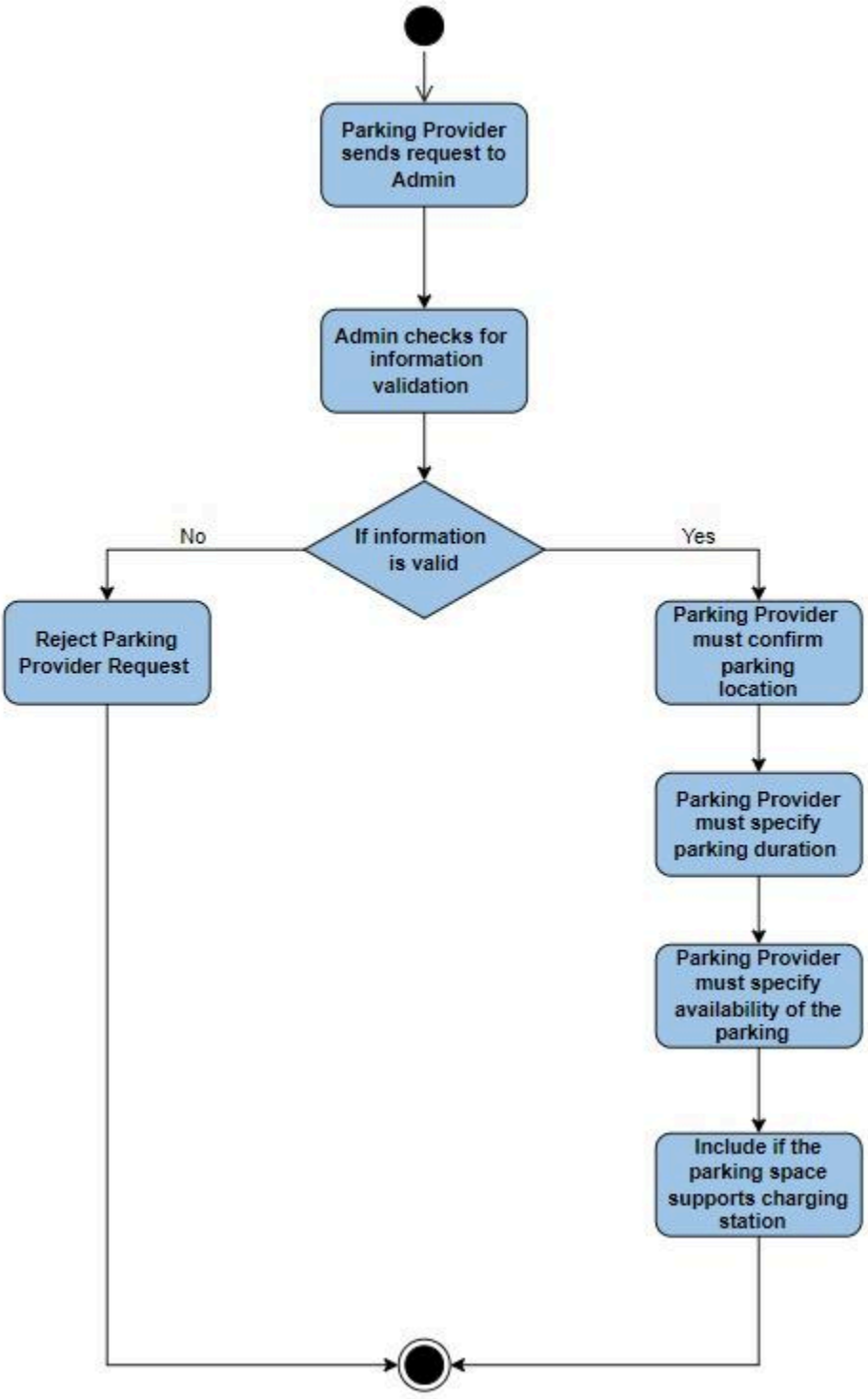


Figure 10: Admin Accept Parking Provider Request

5.3.6 Admin Reject Parking Provider Request

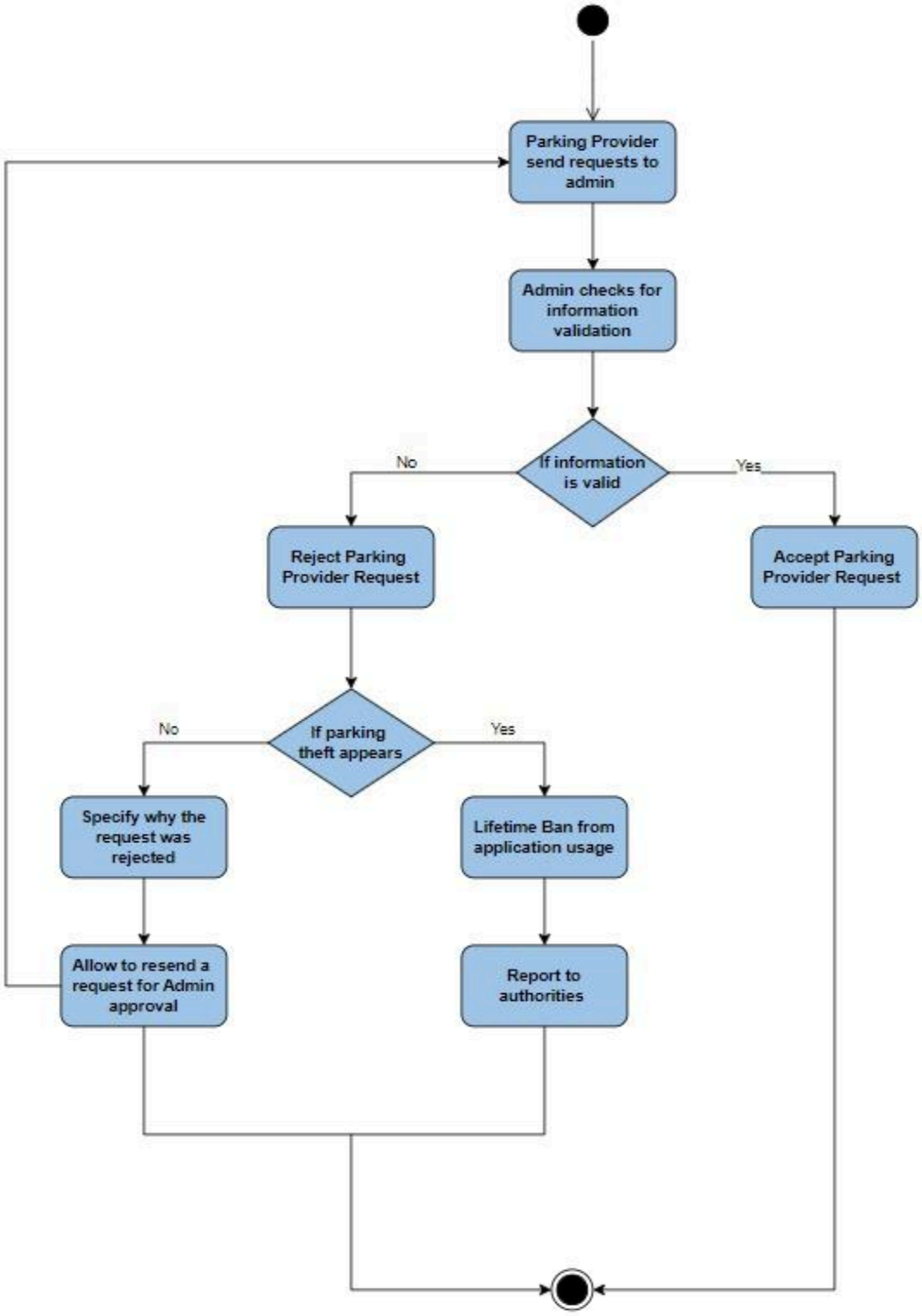


Figure 11: Admin Reject Parking Provider Request

5.3.7 Register

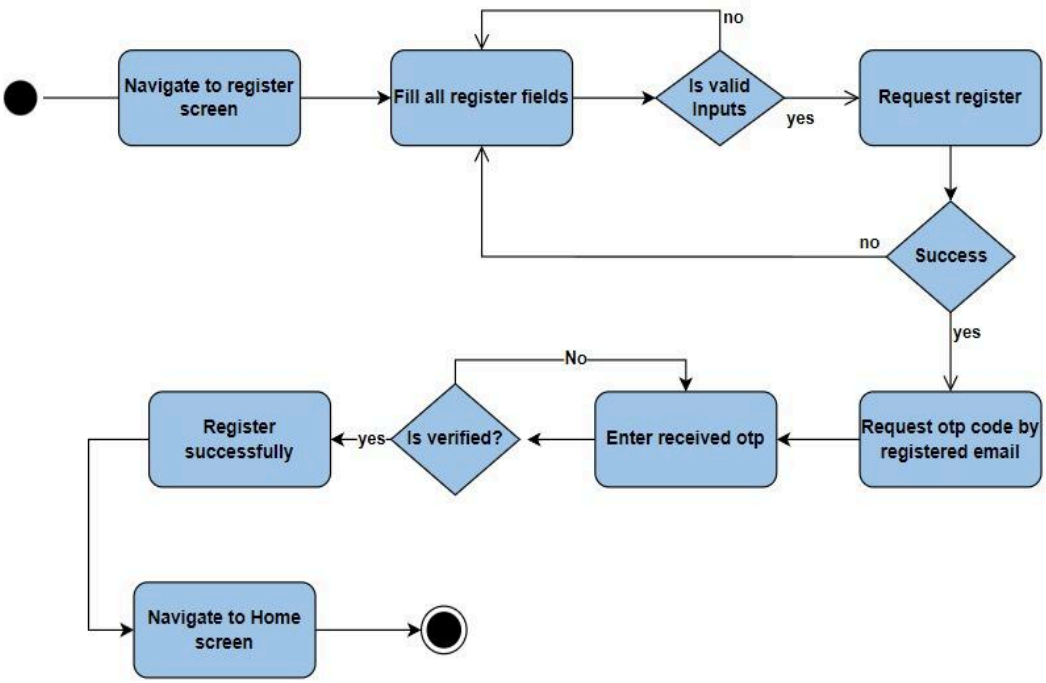


Figure 12: Register

5.3.8 Payment

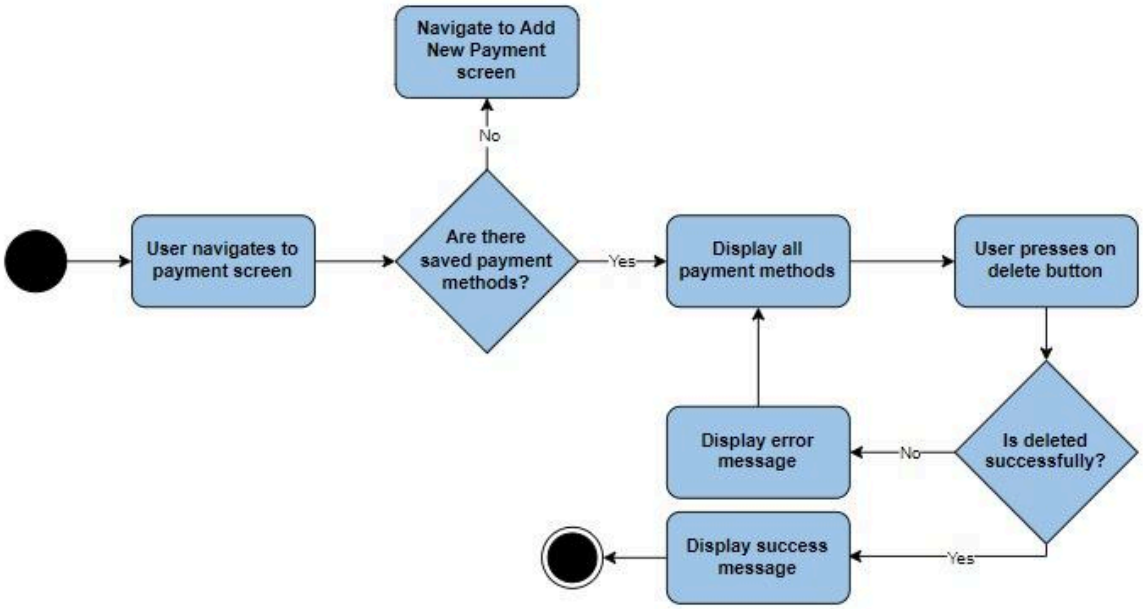


Figure 13: Payment

5.3.9 Feedback

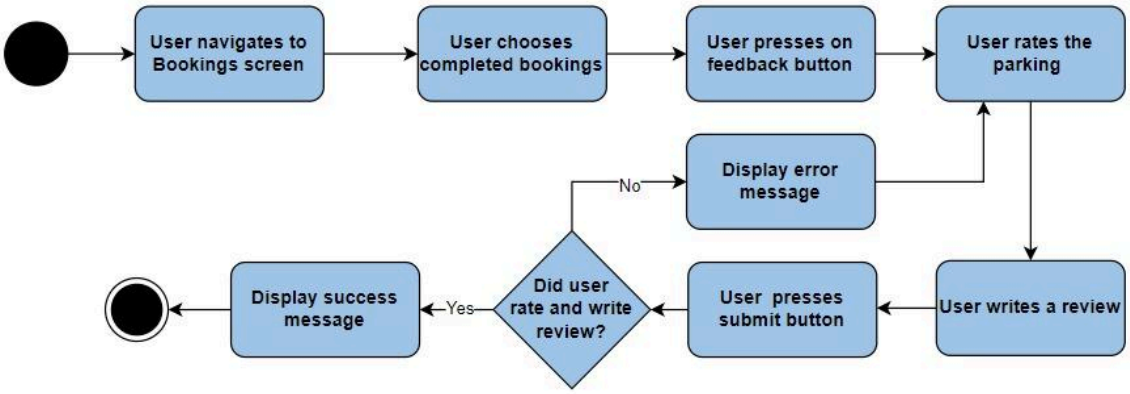


Figure 14: Feedback

5.3.10 Login

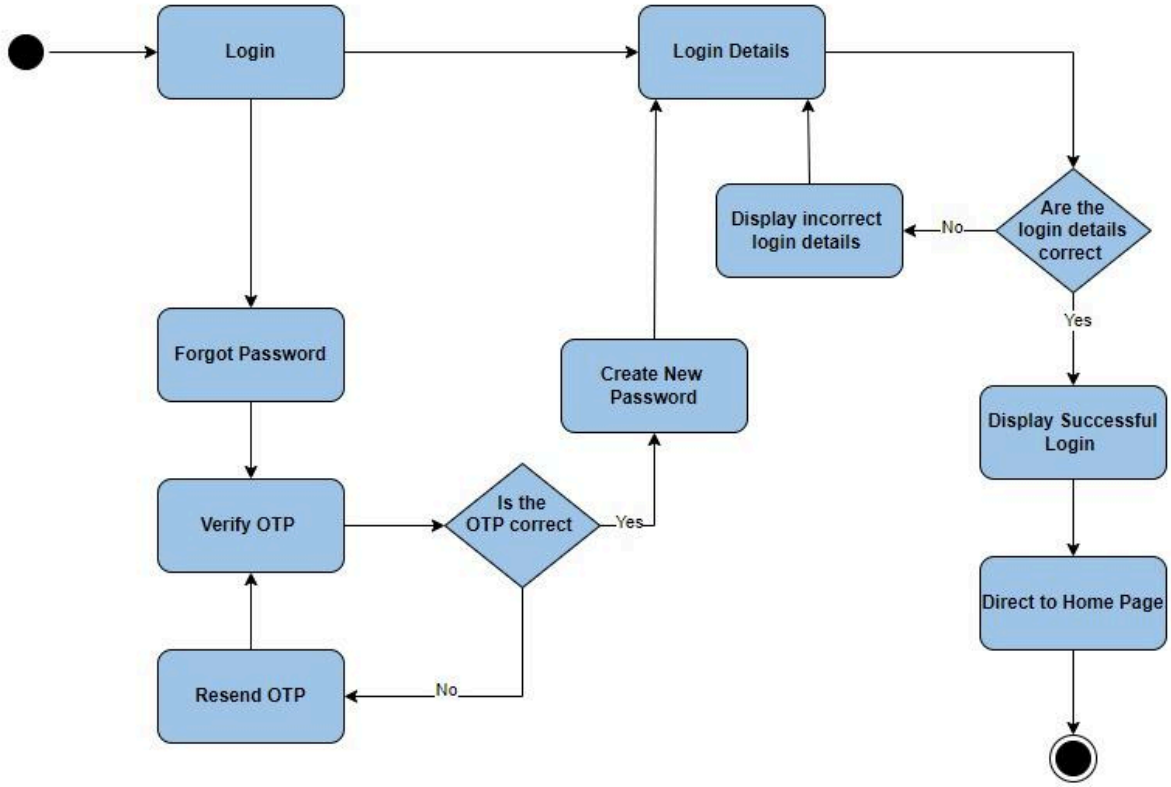


Figure 15: Login

5.4 Class Diagram

The following Class Diagram describes the relationship between the classes and how they interact. Our Class Diagram consists of 12 classes:

- **ParkingSpace:** This class represents the physical parking spot location where users park their vehicles.
- **OpenHours:** This class represents the time availability of the parking space.
- **Feedback:** This class is responsible for providing the user with the ability to Star and type a written Review of their experience.
- **Location:** This class represents the geographical location of each parking space.
- **Notification:** This class's responsibility is to provide the user with a push notification that contains a title, image, and description.
- **Person:** This class provides the **User** and **ParkingProvider** classes with identical attributes.
- **User:** This class is inherited from **Person**, it handles the attributes favorite parking space selected and the payment method.
- **ParkingProvider:** This class is inherited from **Person,** it is responsible for managing and providing the parking space.
- **Admin:** This class represents an administrator who has privileged access to approve or decline parking spaces.
- **PaymentMethod:** This class is responsible for filling out the credit card information which includes Card Number, Card Holder, Expiration Date, and CVC.
- **PaymentRequest:** This class represents the payment attributes and approving payment.
- **Transaction:** This class is responsible for the transactions that happen between the **User** and **ParkingProvider**.

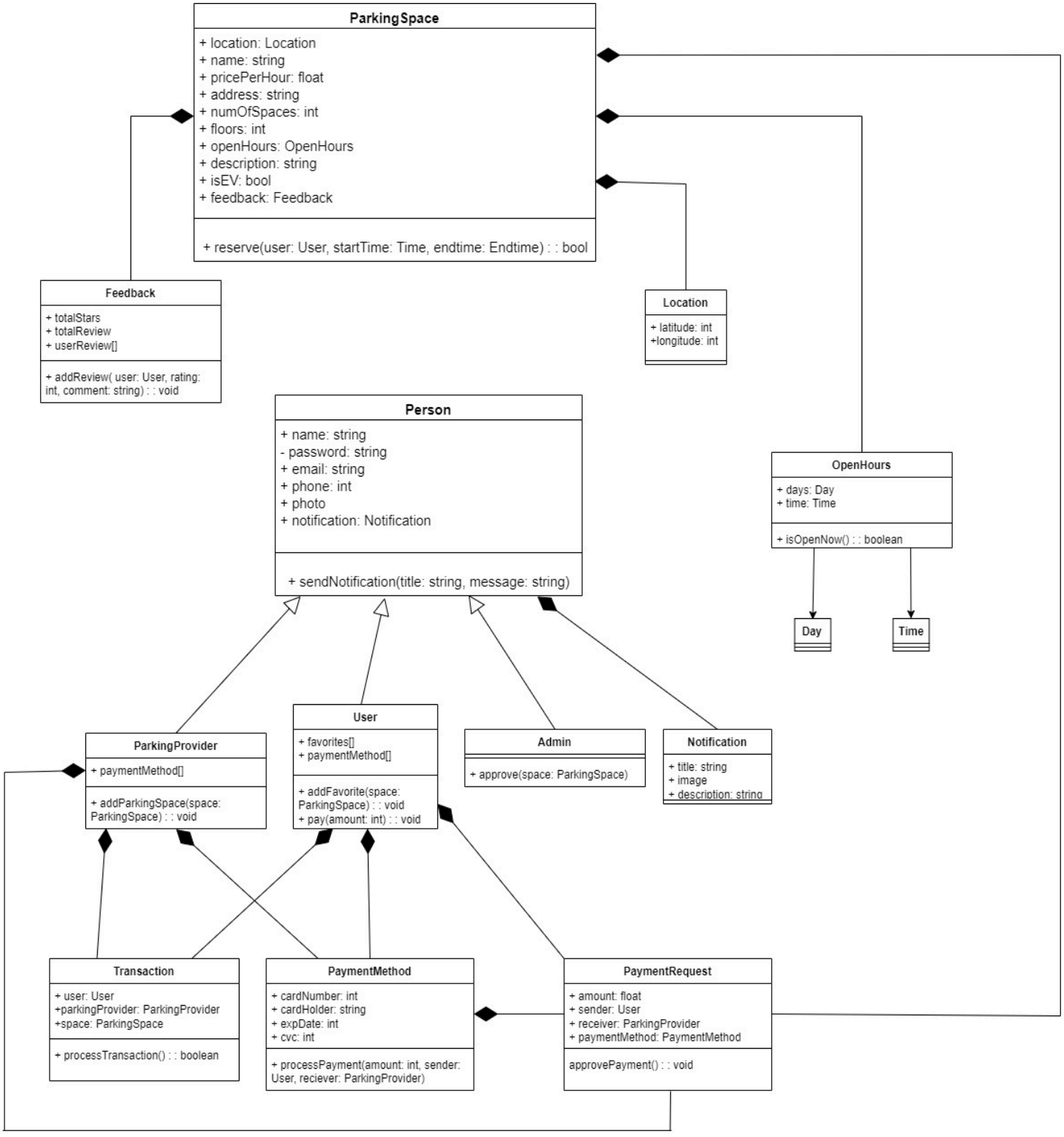
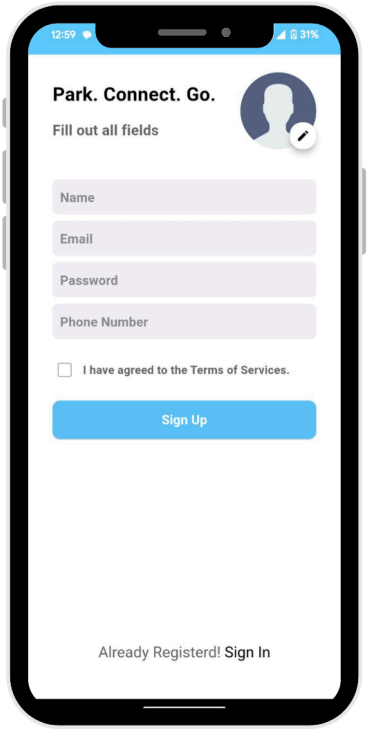
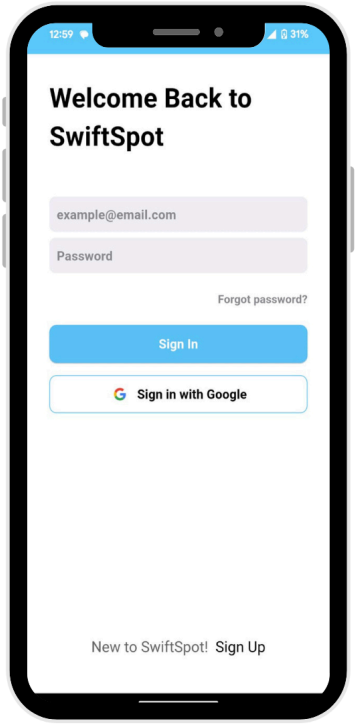


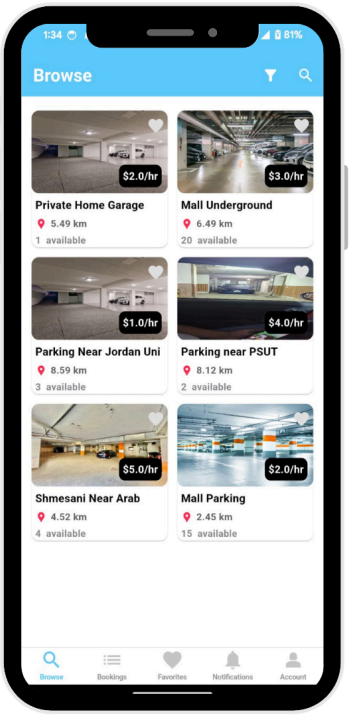
Figure 16: Class Diagram

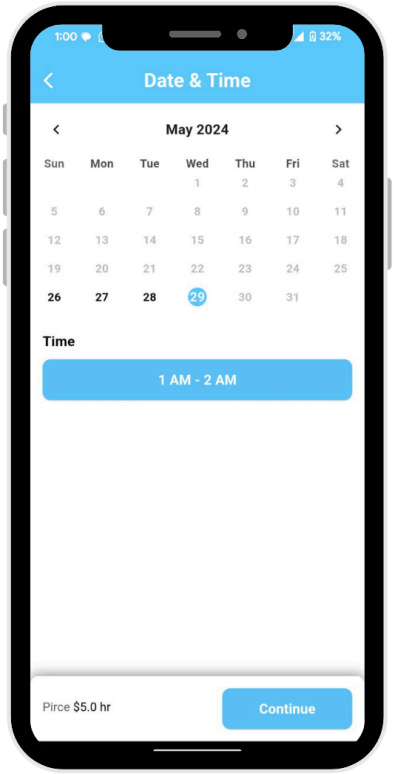
Chapter 6: Software Implementation

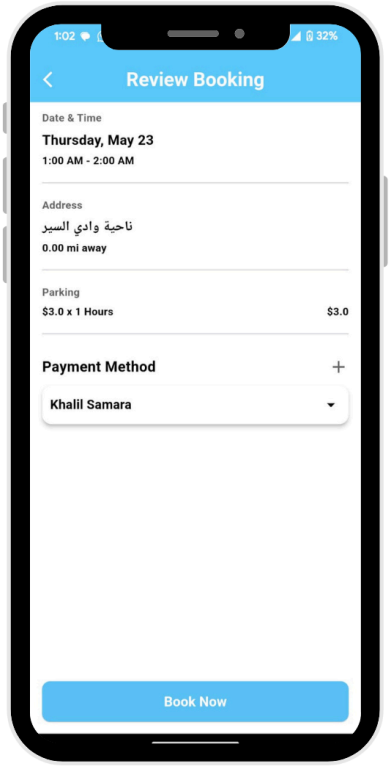
6.1 Graphical User Interface

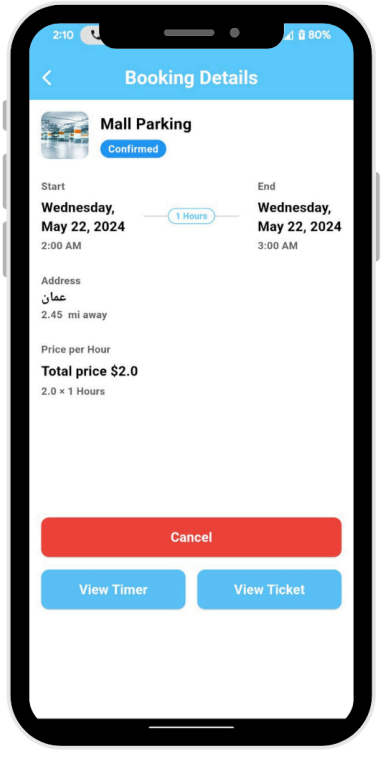


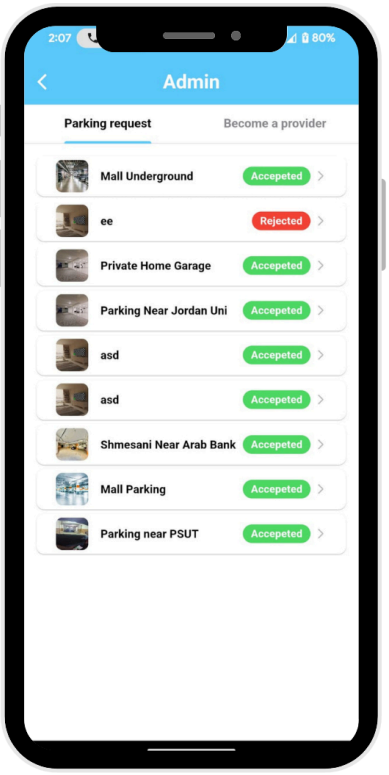


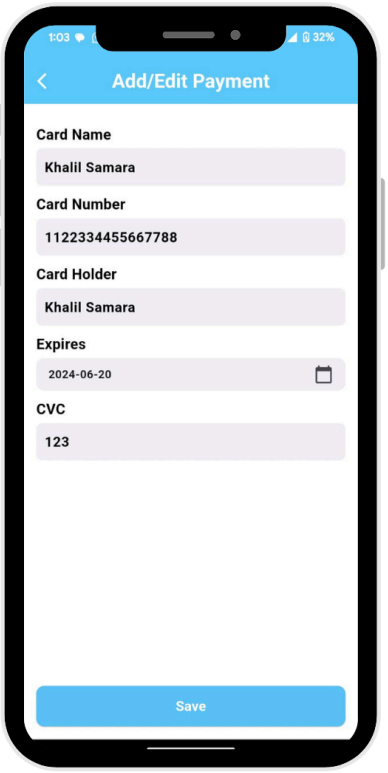


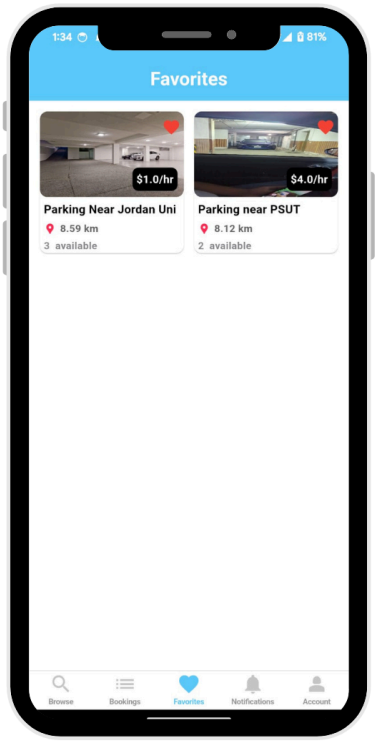


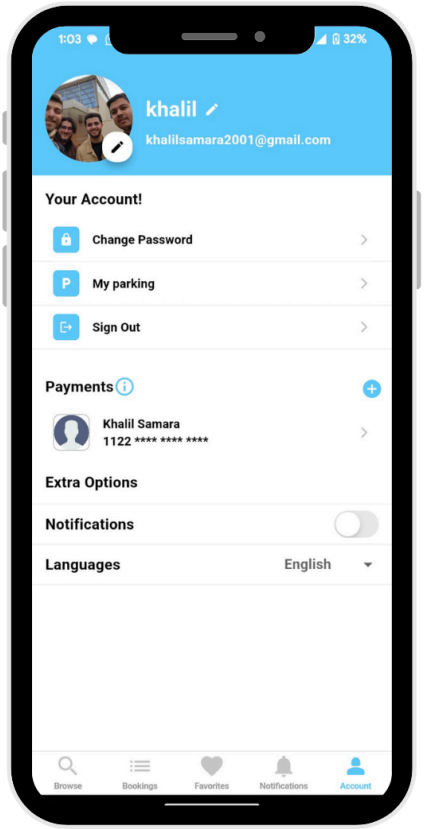


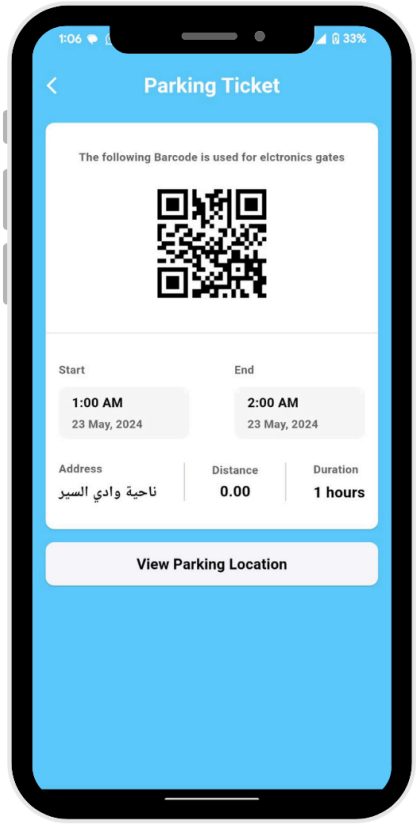



Admin
Parking request
Become a provider
Mall Underground
Accepeted
ee
Rejected
Private Home Garage
Accepeted
Parking Near Jordan Uni
Accepeted
asd
Accepeted
asd
Accepeted
Shmesani Near Arab Bank
Accepeted
Mall Parking
Accepeted
Parking near PSUT
Accepeted


Add/Edit Payment
Card Name
Khalil Samara
Card Number
1122334455667788
Card Holder
Khalil Samara
Expires
2024-06-20
CVC
123
Save


Favorites
$1.0/hr
Parking Near Jordan Uni
8.59 km
3 available
$4.0/hr
Parking near PSUT
8.12 km
2 available
Browse
Bookings
Favorites
Notifications
Account


khalil
khalilsamara2001@gmail.com
Your Account!
Change Password
My parking
Sign Out
Payments
Khalil Samara
1122 **** **** ****
Extra Options
Notifications
Languages
English
Browse
Bookings
Favorites
Notifications
Account


Parking Ticket
The following Barcode is used for elctronics gates
Start
1:00 AM
23 May, 2024
End
2:00 AM
23 May, 2024
Address
ناحية وادي السير
Distance
0.00
Duration
1 hours
View Parking Location

6.2 Code

6.2.1 Sign Up

Allows any new user to create an account by entering email, password, name, and phone number.

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    resizeToAvoidBottomInset: false,
    body: Form(
      key: _formKey,
      child: Container(
        width: double.maxFinite,
        padding: EdgeInsets.symmetric(
          horizontal: 32,
          vertical: 50,
        ), // EdgeInsets.symmetric
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            SizedBox(height: 25),
            _buildWelcomeUserSection(context),
            SizedBox(height: 39),
            _buildNameSection(context),
            SizedBox(height: 8),
            _buildEmailSection(context),
            SizedBox(height: 8),
            _buildPasswordSection(context),
            SizedBox(height: 8),
            _buildMobileLabelSection(context),
            SizedBox(height: 24),
            _buildTermsOfServiceSection(context),
            SizedBox(height: 24),
            _buildSignUpSection(context),
            Spacer(),
            Center(
              child: RichText(
                text: TextSpan(
                  children: [
                    TextSpan(
                      text: translation(context).msg_have_an_account2,
                      style: TextStyle(
                        fontSize: 17,
                        color: Color(0xFF666666),
                      ), // TextStyle
                    ), // TextSpan
                    TextSpan(
                        text: translation(context).lbl_sign_in,
                        style: TextStyle(
                          fontSize: 17,
                          color: Colors.black,
                        ), // TextStyle
                        recognizer: TapGestureRecognizer()
```

6.2.2 Sign In

To allow users that have Signed Up to access their accounts.

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    resizeToAvoidBottomInset: false,
    body: Form(
      key: _formKey,
      child: Container(
        width: double.maxFinite,
        padding: EdgeInsets.symmetric(horizontal: 30, vertical: 50),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            SizedBox(height: 16),
            Text(
              translation(context).lbl_welcome_back,
              maxLines: 2,
              overflow: TextOverflow.ellipsis,
              style: TextStyle(
                fontSize: 30,
                color: Colors.black,
              ), // TextStyle
            ), // Text
            SizedBox(height: 15),
            Text(
              translation(context).msg_sign_in_to_continue,
              style: TextStyle(
                fontSize: 17,
                color: Color(0xFF666666),
              ), // TextStyle
            ), // Text
            SizedBox(height: 39),
            CustomTextFormField(
              controller: emailController,
              hintText: "example@example.com",
              textInputType: TextInputType.emailAddress,
              validator: (value) {
                if (value == null ||
                    (!isValidEmail(value, isRequired: true))) {
                  return translation(context)
                      .err_msg_please_enter_valid_email;
                }
                return null;
              },
            ), // CustomTextFormField
```

6.2.3 Admin

Accepts and rejects parking provider requests for becoming a parking provider or adding a parking space.

```
        children: [
          Container(
            padding: EdgeInsets.symmetric(
                horizontal: 10, vertical: 2), // EdgeInsets.symmetric
            decoration: BoxDecoration(
              color: parkingRequestData[index]['status'] == 0
                  ? Colors.blue
                  : parkingRequestData[index]['status'] == 1
                      ? Color(0xFF4CD964)
                      : parkingRequestData[index]['status'] ==
                              -1
                          ? Colors.red
                          : Colors.white,
              borderRadius: BorderRadius.circular(100),
            ), // BoxDecoration
            child: Text(
              parkingRequestData[index]['status'] == 0
                  ? 'Pending'
                  : parkingRequestData[index]['status'] == 1
                      ? 'Accepeted'
                      : parkingRequestData[index]['status'] ==
                              -1
                          ? 'Rejected'
                          : '',
              style: TextStyle(
                fontSize: 13,
                color: Colors.white,
              ), // TextStyle
            ), // Text
          ), // Container
          SizedBox(width: 5),
          Icon(
            Icons.arrow_forward_ios_rounded,
            size: 15,
            color: Colors.grey.shade400,
          ), // Icon
        ],
      ), // Row
      buttonStyle: ButtonStyle(
        backgroundColor:
            MaterialStatePropertyAll(Colors.white),
        surfaceTintColor:
            MaterialStatePropertyAll(Colors.white),
        shape:
            MaterialStatePropertyAll<RoundedRectangleBorder>(
          RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10),
          ), // RoundedRectangleBorder
```

6.2.4 Bookings

Displays all details of the bookings screen.

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      elevation: 0,
      backgroundColor: Color(0xFF5AC8FA),
      centerTitle: true,
      title: Text(
        translation(context).lbl_bookings,
      ), // Text
    ), // AppBar
    body: SingleChildScrollView(
      padding: EdgeInsets.symmetric(horizontal: 5, vertical: 15),
      child: StreamBuilder<QuerySnapshot>(
        stream: FirebaseFirestore.instance
            .collection('bookings')
            .where('userId', isEqualTo: AuthService().userId)
            .snapshots(),
        builder: (context, AsyncSnapshot<QuerySnapshot> snapshot) {
          if (snapshot.connectionState == ConnectionState.waiting) {
            return Center(child: CircularProgressIndicator());
          }
          if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {
            return Center(child: Text('No bookings found'));
          }

          return Column(
            children: snapshot.data!.docs.map((DocumentSnapshot document) {
              Map<String, dynamic> data =
                  document.data() as Map<String, dynamic>;
              return FutureBuilder<DocumentSnapshot>(
                future: FirebaseFirestore.instance
                    .collection('products')
                    .doc(data['parkingId'])
                    .get(),
                builder: (BuildContext context,
                    AsyncSnapshot<DocumentSnapshot> snapshot) {
                  if (snapshot.connectionState == ConnectionState.waiting) {
                    return Center(
                      child: CircularProgressIndicator(),
                    ); // Center
                  }
                  if (snapshot.hasError) {
                    return Text("Error: ${snapshot.error}");
                  }
                  if (!snapshot.hasData && !snapshot.data!.exists) {
                    return Center(
                      child: Text('something error with this parking'),
                    ); // Center
                  }
```

6.2.5 Become Provider

Sends request to Admin for becoming a parking provider.

```
CustomElevatedButton(
  text: translation(context).lbl_send,
  isLoading: isSubmitLoading,
  onPressed: () {
    setState(() {
      isSubmitLoading = true;
    });
    showMsg(
      context,
      'by continuing you agree to terms and conditions',
      showCancle: true,
      oncancel: () {
        setState(() {
          isSubmitLoading = false;
        });
      },
      onpress: () async {
        if (_formKey.currentState!.validate()) {
          if (sortOptions == 'NationalID') {
            if (areAllImagesUploaded(nationalId)) {
              await AuthService().requestBecomeProvider({
                "documents": nationalId,
                "name": nameController.text,
                "email": emailController.text,
                "mobile": mobileController.text,
                "status": 0,
              }).then((value) {
                setState(() {
                  isSubmitLoading = false;
                });
                if (value.isNotEmpty) {
                  clearValues();
                  showMsg(
                    context,
                    'Your request was sent successfully',
                    onpress: () {
                      Navigator.pop(context);
                    },
                  );
                } else {
                  showMsg(context, 'Something went error');
                }
              });
```

6.2.6 Search

Allows users to search for parking spaces from the browse screen.

```
String searchKey = "";
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      elevation: 0,
      backgroundColor: Colors.white,
      automaticallyImplyLeading: false,
      title: Row(
        children: [
          Expanded(
            child: Container(
              margin: EdgeInsets.only(right: 10.0),
              height: 40,
              child: Hero(
                tag: "searchInput",
                child: Material(
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10),
                  ), // RoundedRectangleBorder
                  child: SearchInput(
                    controller: _searchController,
                    hintText: translation(context).lbl_search,
                    bgColor: const Color(0xFFEFEFF4),
                    onChanged: (value) {
                      setState(() {
                        searchKey = value;
                      });
                    },
                    onClear: () {
                      setState(() {
                        searchKey = '';
                      });
                    },
                  ), // SearchInput
                ), // Material
              ), // Hero
            ), // Container
          ), // Expanded
          InkWell(
            onTap: () {
              Navigator.pop(context);
            },
```

6.3 Future Work

In an ideal business model, our system should be divided into multiple systems. At first, the user would have his own application to book parking spaces, view notifications, give feedback, pay for renting a parking space, and provide if the vehicle is an EV. Then the parking provider would have a separate application to add parking space, delete parking space, update parking space, view user feedback, contact admin in case any issues appear, and receive payment for providing the parking space. The admin would access all the data needed through a web browser for accepting and rejecting parking provider requests, accepting and rejecting parking space requests, maintaining and updating the system, providing customer support if needed, and retrieving data for future improvement.

Due to time constraints, we have combined the User, Parking Provider, and Admin in one application. The functionality on the account screen can be changed from the Firebase to switch between the parking provider and admin. We weren't able to complete our requirements because of the lack of time but we were able to finish the main functionalities of the application such as booking a parking space, searching or filtering, adding a parking space, requesting to become a parking provider, accepting/rejecting parking provider and parking space. But some sub-functionalities weren't completed yet such as notifications, adding if the vehicle is an EV, adding if the parking space supports EV vehicles, providing an onboarding screen for first-time use of the application, star rating function, feedback from the user, providing contact information of parking provider to the user, and deleting parking provider added parking space.

We plan to finish all the remaining unfinished functionalities before deploying the application for beta and user testing. In an estimated four months, the application could be ready for the final deployment on both IOS and Android. More testing will be required which will need more time but our priority is user satisfaction and meeting such important aspects needs patience and time. With the rise of technology, the competition is getting harder every day, so we also plan to improve our payment method, duration of parking space, and account screen to satisfy our users and meet the market expectations.

Chapter 7: Software Testing

When it comes to the world of software creation testing is a key phase to achieving user acceptance and a working environment for users. So the following testing techniques were decided for this project **White Box**, **Black Box**, **Ad Hoc**, **Performance**, and **Security** Testing.

Due to time constraints, we couldn't implement User, Acceptance, and Beta Testing as they require users to test the application to give feedback on the application and how it could be improved in the future.

As software engineering students our background in security testing is narrow, so we chose to implement the Microsoft Threat Modeling Tool that was given to us in the Secure Software Development course. The version that was used is (Microsoft Threat Modeling Tool 2016).

To ensure that software testing was conducted correctly and thoroughly we have created a test plan to allocate the task to each team member:

Test ID	Test Name	Test Allocation	Test Predecessor	Test Status
T1	White Box	Khalil Samara	-	Completed
T2	Black Box	Omar Obeid	-	Completed
T3	Ad Hoc	Omar Obeid	-	Completed
T4	Performance	Khalil Samara	-	Completed
T5	Security	Omar Obeid	-	Completed

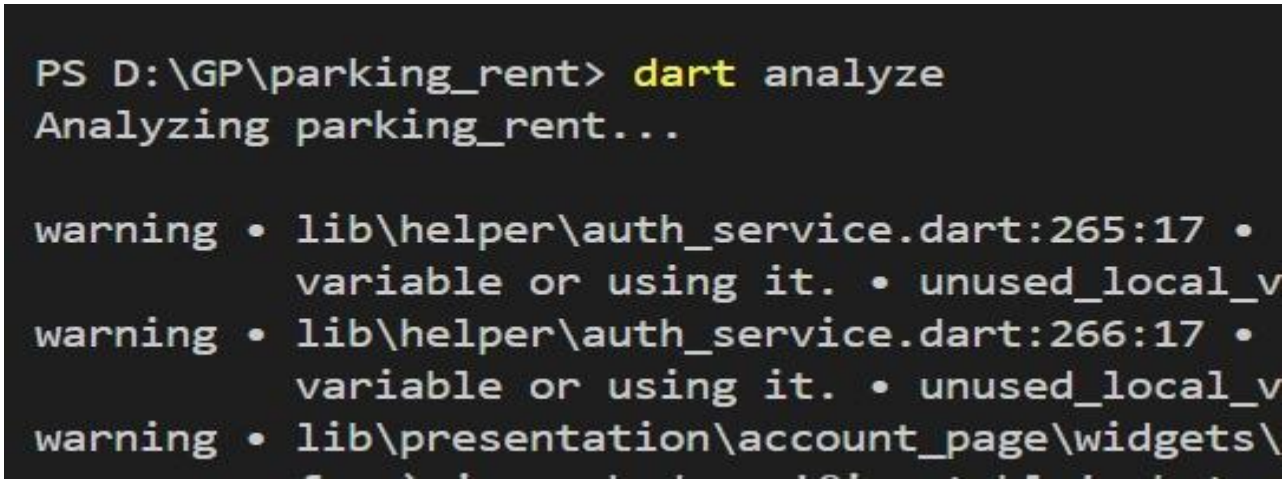
Table 15: Test Plan

7.1 White Box Testing

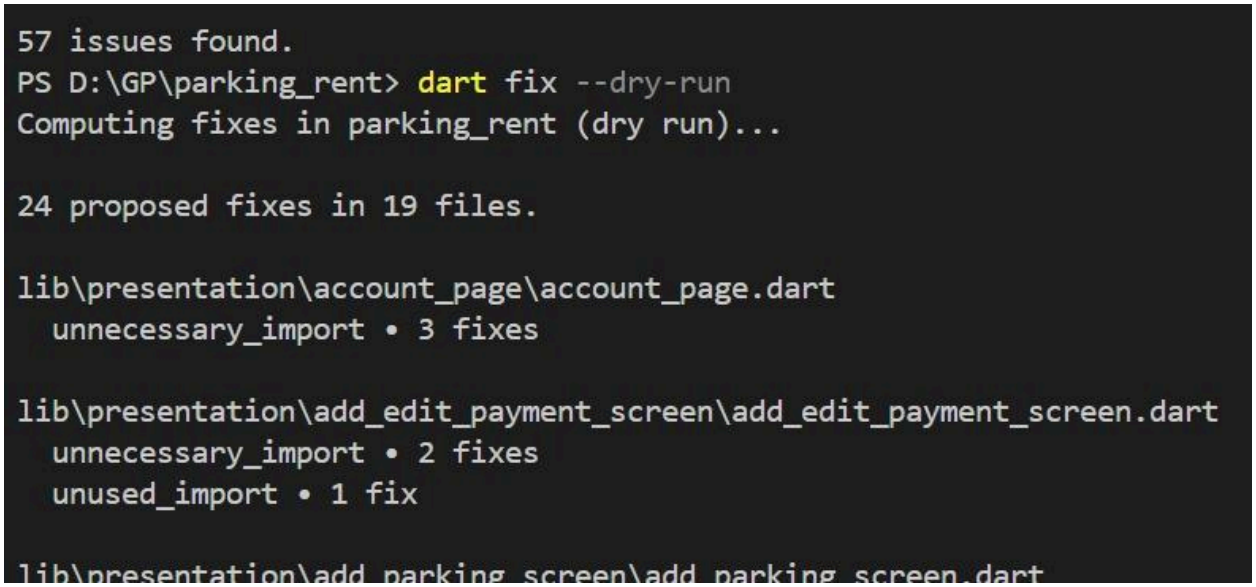
Throughout the development process we conducted unit testing, each function underwent unit testing to ensure it served its intended purpose and functionality before being integrated with the rest of the project. This was an important step for identifying and addressing issues and bugs in the early stages of development. Additionally regression and confirmation testing were applied after ever change made to the application.

Dart Command Line Tool:

Initial Analysis



Suggested Fixes



Apply Fixes

```
PS D:\GP\parking_rent> dart fix --apply
Computing fixes in parking_rent...
Applying fixes...

lib\presentation\account_page\account_page.dart
  unnecessary_import • 3 fixes

lib\presentation\add_edit_payment_screen\add_edit_payment_screen.dart
  unnecessary_import • 2 fixes
  unused_import • 1 fix

lib\presentation\add_parking_screen\add_parking_screen.dart
```

Final Result

```
  unused_import • 1 fix

lib\widgets\custom_text_form_field.dart
  unused_import • 1 fix

pubspec.yaml
  MISSING_DEPENDENCY • 1 fix

24 fixes made in 19 files.
PS D:\GP\parking_rent>
```

7.2 Black Box Testing

When conducting black box testing, the testing is done without any knowledge of its internal code. Test cases are provided to ensure a working functional application, the following black box testing is done for the most important functions in the application.

Test Case 1: Signin to the application

Test Case Description: Verify that login is successful with valid credentials.
Test Precondition: User must be registered (signed up) to the Application.
Test Steps:

1. Access the sign in screen.
2. Enter email and password.
3. Click on the sign in button provided.

Test Data 1:

- Email: omarobeid136@gmail.com
- Password: Omar@2001

Test Result: User gains access to the application and allowed to use its functionality.

Test Data 2:

- Email: (left empty)
- Password: (left empty)

Test Result: Asks for valid email and password.

Test Data 3:

- Email: omarobeid136@gmail.com
- Password: 1111111

Test Result: Asks for valid password.

Test Case 2: Signup to the application

Test Case Description: Verify that user can signup by filling correct credentials.
Test Precondition: None
Test Steps:

1. Access the sign up screen.
2. Fill name, email, password, and mobile.
3. Agree to the term of service.
4. Click on sign up button provided.

Test Data 1:

- Name: Omar Obeid
- Email: omarobeid136@gmail.com
- Password: Omar@2001
- Mobile: 0796019477
- Agreement for Terms and Services: [checked]

Test Result: User is signed up and is accessed to login.

Test Data 2:

- Name: Omar Obeid
- Email: omarobeid136@gmail.com
- Password: Omar@2001
- Mobile: 0796019477
- Agreement for Terms and Services: [un-checked]

Test Result: User must agree to the terms and services of the application.

Test Data 3:

- Name: Omar Obeid
- Email: omarobeid136@gmail.com
- Password: 111111
- Mobile: 0796019477
- Agreement for Terms and Services: [checked]

Test Result: Asks for valid password.

Test Data 4:

- Name: Omar Obeid
- Email: omarobeid136gmailcom
- Password: Omar@2001
- Mobile: 0796019477
- Agreement for Terms and Services: [checked]

Test Result: Asks to enter the correct format of the email.

Test Data 5:

- Name: Omar Obeid
- Email: omarobeid136@gmail.com
- Password: Omar@2001
- Mobile: try1
- Agreement for Terms and Services: [checked]

Test Result: Must enter a valid number without any strings.

Test Case 3: Search for available parking spaces

Test Case Description: Verify that user can search for parking spaces.
Test Precondition: User must login to the application.
Test Steps:

1. Access the browse screen.
2. Click on the search icon.
3. Enter string for searching.
4. Click on the search bar.

Test Data 1:

- Entry: Parking

Test Result: All parking names that start with parking and end with parking are displayed.

Test Data 2:

- Entry: 11111

Test Result:No parking is displayed because their is no such parking space

Test Case 4: Parking provider adds new parking space

Test Case Description: Verify that parking provider can add a new parking space.
Test Precondition: Parking provider must be accepted by admin.
Test Steps:

1. Access the account screen.
2. Request to become a provider.
3. Add the input for adding a parking space.
4. Select open hours.
5. Set location via button provided.
6. Click on the send button.

Test Data 1:

- Parking Name: The Great Mall
- Parking Description: This parking is available for you!
- Available Parking Space: 15DE
- Price per Hour: 12
- Phone Number: 0776123477
- Add Image: 1 image
- Select open hours: [open] 9 am - 10 pm

Test Result: Available parking space only accepts digits.

Test Data 2:

- Parking Name: The Great Mall
- Parking Description: This parking is available for you!
- Available Parking Space: 15
- Price per Hour: 12D
- Phone Number: 0776123477
- Add Image: 1 image
- Select open hours: [open] 9 am - 10 pm

Test Result: Price per hour only accepts digits.

Test Data 3:

- Parking Name: The Great Mall
- Parking Description: This parking is available for you!
- Available Parking Space: 15
- Price per Hour: 12
- Phone Number: 07761234ee
- Add Image: 1 image
- Select open hours: [colsed]

Test Result: Phone number only accepts digits.

Test Data 4:

- Parking Name: The Great Mall
- Parking Description: This parking is available for you!
- Available Parking Space: 15
- Price per Hour: 12
- Phone Number: 0776123477
- Add Image: None
- Select open hours: [open] 9 am - 10 pm

Test Result: Atleast one image must be uploaded to send request.

Test Case 5: Add payment

Test Case Description: Verify that user can add a card.
Test Precondition: User must login to the application.
Test Steps:

1. Access the account page.
2. Click on the add blue button.
3. Fill card name, number, holder, expires, and CVC.
4. Click save.

Test Data 1:

- Card Name: Omar Visa
- Card Number: 12345678912345 (14 digits)
- Card Holder: OMAR OBEID
- Expires: 13/08/2027
- CVC: 123

Test Result: Error appears the validation only allows 16 digits.

Test Data 2:

- Card Name: Omar Visa
- Card Number: 1234567891234534
- Card Holder: OMAR OBEID22
- Expires: 13/08/2027
- CVC: 123

Test Result: Error appears card holder should be only string.

Test Data 3:

- Card Name: Omar Visa
- Card Number: 1234567891234534
- Card Holder: OMAR OBEID
- Expires: 13/08/2027
- CVC: 1234

Test Result: Error appears CVC shouldonly contain 3 digits.

Test Data 4:

- Card Name: Omar Visa
- Card Number: 1234567891234534
- Card Holder: OMAR OBEID
- Expires: 13/08/2027
- CVC: 123FG

Test Result: Error appears CVC shouldonly contain 3 digits and should be digits.

7.3 Ad Hoc Testing

We conducted multiple types of testing for our project but there is a lot of testing to do and learn. So the best solution for our time constraint in the project was to apply Ad Hoc testing. Ad Hoc testing is categorized as a software testing technique and it's performed without any test cases, plans, or predefined set of steps. Instead, we must use our experience, intuition, and creativity to identify bugs, issues, and defects that other forms of testing methods may not identified.

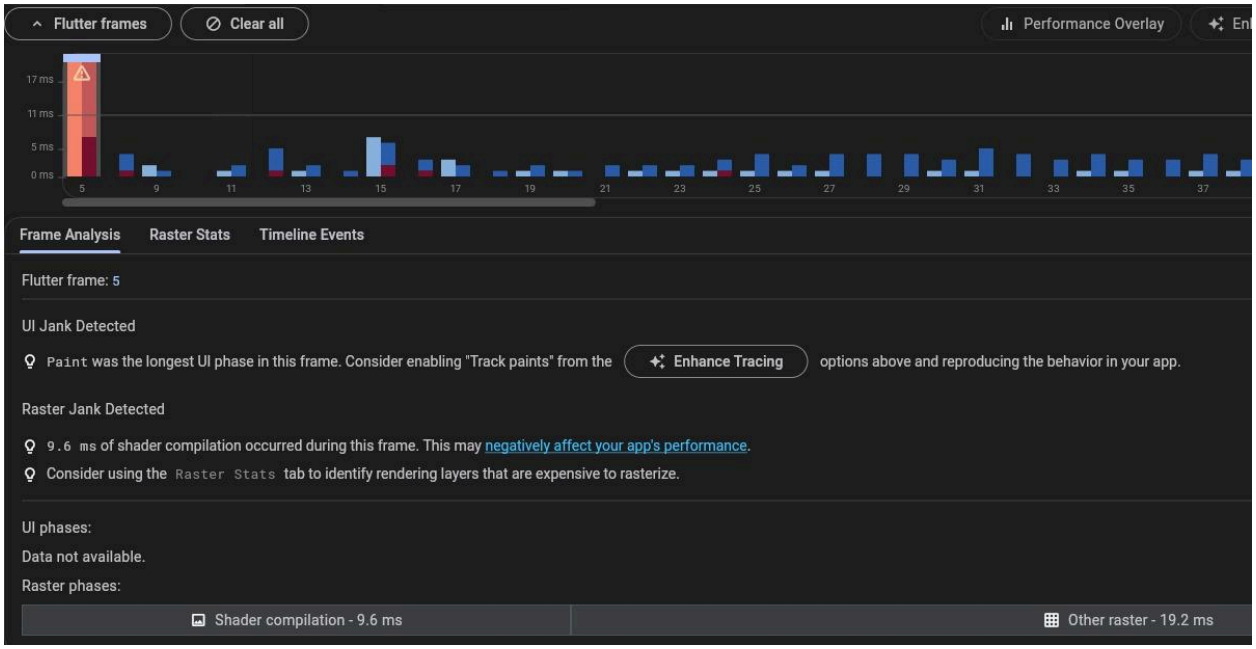
Screen	Issue	Solution
add_edit_payment_screen	The card number wasn't validated to 16, the user was able to enter more or less.	Validated the input of the user to only 16 digits.
add_edit_payment_screen	The cardholder's name could only be one word and didn't allow any spaces.	Allowed the user to add multiple words and numbers.
add_edit_payment_screen	The CVC wasn't validated to 3, the user was able to enter more or less.	Validated the input of the user to only 3 digits.
bookings_details_screen	There was no button for canceling the parking space.	A cancel button was added so the user can cancel their request if needed.
sign_in_screen	There was no validation on the password.	A validation was added to ensure security safety.
admin_screen	Didn't show rejected parking spaces.	Shows rejected parking spaces.
account_page	The user couldn't turn off notifications.	A toggle was added to allow the user to turn notifications on or off.
add_parking_screen	The parking provider can't delete or update parking spaces.	Added to the future work part to resolve the issue before deployment.
enter_date_time_screen	The timer range didn't have a constraint to show what time the parking space was closed.	A constraint was added in the color red to prohibit the user from picking a closed time.

Table 16: Ad Hoc Testing

7.4 Preformance Testing

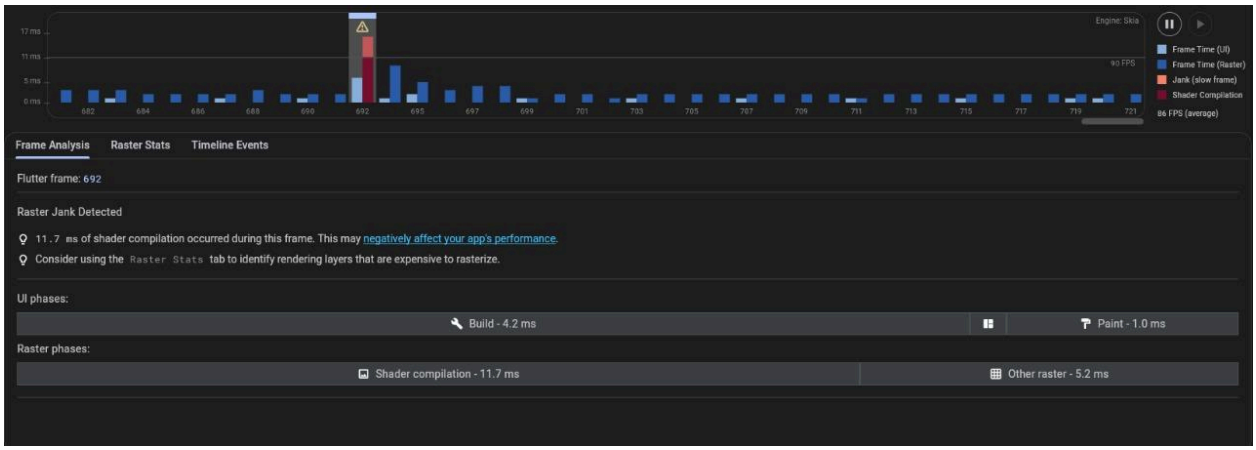
When assessing an application performance is a key factor, we conducted a performance testing to assess the applications responsiveness, speed, and overall performance. The following are the frames that took the longest to load, their loading time were still relatively fast and efficient.

Application Startup



Parking Details Screen

Select Payment Method from Review Booking Screen



7.5 Security Testing

We have conducted our security testing by using the Microsoft Threat Modeling Tool, by providing a data flow diagram that presents our system and how it's main functionality work. While conducting our testing we encountered 42 total threats such as:

- Spoofing
- Elevation Of Privilege
- Denial of Service (DoS)
- Tampering
- Repudiation
- Information Disclosure

To ensure such threats are resolved a professional security team must be appointed to handle such threats and resolve them before the final deployment of the application.

7.5.1 Security System Design

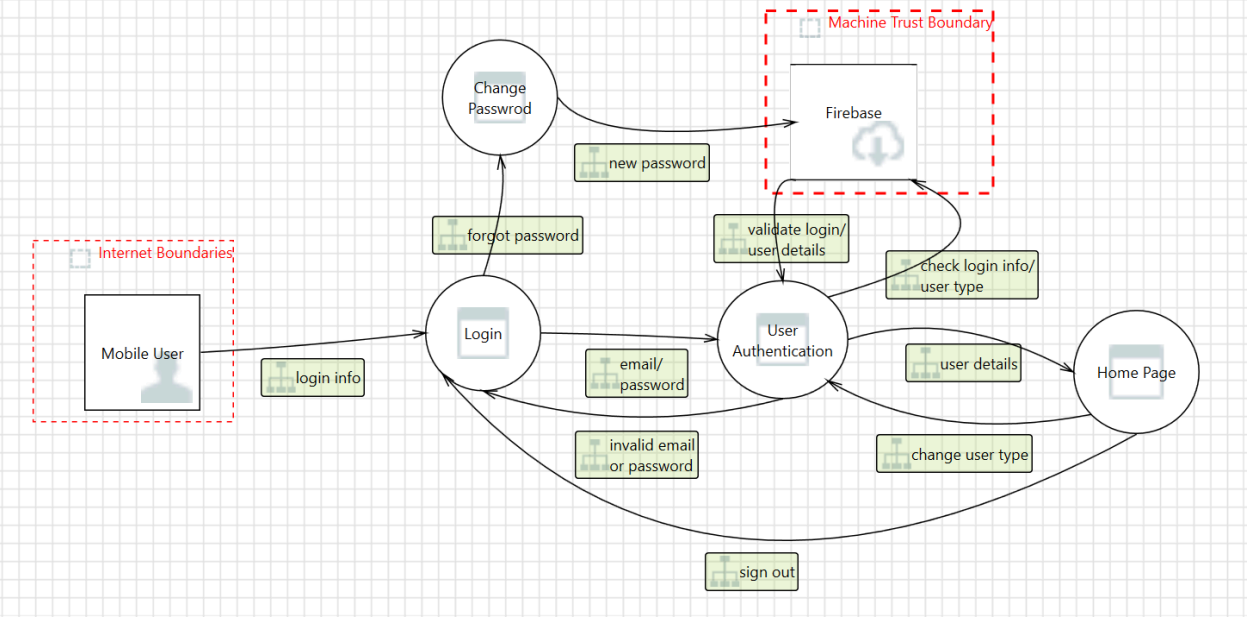


Figure 17: Security System Design

7.5.2 Security Threats

Threat List

ID	Diagram	Changed By	Last Modified	State	Title	Category	Description	Justification	Interaction	Priority
9	Diagram 1		Generated	Not Started	Spoofing the...	Spoofing	Mobile User m...		login info	High
15	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	Login may be...		login info	High
19	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	Change Passwr...		forgot password	High
20	Diagram 1		Generated	Not Started	Spoofing of D...	Spoofing	Firebase may...		new password	High
21	Diagram 1		Generated	Not Started	Potential Exces...	Denial Of Servi...	Does Change...		new password	High
29	Diagram 1		Generated	Not Started	Spoofing the L...	Spoofing	Login may be...		login info	High
30	Diagram 1		Generated	Not Started	Potential Lack...	Tampering	Data flowing a...		login info	High
31	Diagram 1		Generated	Not Started	Potential Data...	Repudiation	Login claims t...		login info	High
32	Diagram 1		Generated	Not Started	Data Flow Snif...	Information Di...	Data flowing a...		login info	High
33	Diagram 1		Generated	Not Started	Potential Proc...	Denial Of Servi...	Login crashes,...		login info	High
34	Diagram 1		Generated	Not Started	Data Flow logi...	Denial Of Servi...	An external ag...		login info	High
35	Diagram 1		Generated	Not Started	Login May be...	Elevation Of Pr...	Mobile User m...		login info	High
36	Diagram 1		Generated	Not Started	Elevation by C...	Elevation Of Pr...	An attacker m...		login info	High
37	Diagram 1		Generated	Not Started	Cross Site Req...	Elevation Of Pr...	Cross-site req...		login info	High
39	Diagram 1		Generated	Not Started	Spoofing of So...	Spoofing	Firebase may...		validate login/...	High
41	Diagram 1		Generated	Not Started	Weak Access C...	Information Di...	Improper data...		validate login/...	High
47	Diagram 1		Generated	Not Started	Spoofing the C...	Spoofing	Change Passwr...		new password	High
48	Diagram 1		Generated	Not Started	The Firebase...	Tampering	Data flowing a...		new password	High
49	Diagram 1		Generated	Not Started	Data Store De...	Repudiation	Firebase claim...		new password	High
50	Diagram 1		Generated	Not Started	Data Flow Snif...	Information Di...	Data flowing a...		new password	High
51	Diagram 1		Generated	Not Started	Data Flow new...	Denial Of Servi...	An external ag...		new password	High
52	Diagram 1		Generated	Not Started	Data Store Ina...	Denial Of Servi...	An external ag...		new password	High
53	Diagram 1		Generated	Not Started	Spoofing the...	Spoofing	User Authentic...		validate login/...	High
54	Diagram 1		Generated	Not Started	Potential Data...	Repudiation	User Authentic...		validate login/...	High
55	Diagram 1		Generated	Not Started	Potential Proc...	Denial Of Servi...	User Authentic...		validate login/...	High
56	Diagram 1		Generated	Not Started	Data Flow user...	Denial Of Servi...	An external ag...		validate login/...	High
57	Diagram 1		Generated	Not Started	Data Store Ina...	Denial Of Servi...	An external ag...		validate login/...	High
58	Diagram 1		Generated	Not Started	User Authenta...	Elevation Of Pr...	Firebase may...		validate login/...	High
59	Diagram 1		Generated	Not Started	Elevation by C...	Elevation Of Pr...	An attacker m...		validate login/...	High
60	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	User Authentic...		email/ password	High
61	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	Login may be...		invalid email o...	High
62	Diagram 1		Generated	Not Started	Spoofing the...	Spoofing	User Authentic...		check login inf...	High
63	Diagram 1		Generated	Not Started	Spoofing of D...	Spoofing	Firebase may...		check login inf...	High
64	Diagram 1		Generated	Not Started	The Firebase...	Tampering	Data flowing a...		check login inf...	High
65	Diagram 1		Generated	Not Started	Data Store De...	Repudiation	Firebase claim...		check login inf...	High
66	Diagram 1		Generated	Not Started	Data Flow Snif...	Information Di...	Data flowing a...		check login inf...	High
67	Diagram 1		Generated	Not Started	Potential Exces...	Denial Of Servi...	Does User Aut...		check login inf...	High
68	Diagram 1		Generated	Not Started	Data Flow ALP...	Denial Of Servi...	An external ag...		check login inf...	High
69	Diagram 1		Generated	Not Started	Data Store Ina...	Denial Of Servi...	An external ag...		check login inf...	High
70	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	Home Page m...		user details	High
71	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	User Authentic...		change user ty...	High
72	Diagram 1		Generated	Not Started	Elevation Usin...	Elevation Of Pr...	Login may be...		sign out	High

Figure 18: Security Threats

References

1. ‘Neighbor’, 2017. [Online]. Available: https://www.neighbor.com/. [Accessed: 06- Nov- 2023].
2. Toby. Littin, ‘Parkable’, 2015. [Online]. Available: https://parkable.com/. [Accessed: 06- Nov- 2023].
3. Anthony. Eskinazi, ‘JustPark’, 2006. [Online]. Available: https://www.justpark.com/. [Accessed: 07- Nov- 2023].
4. Ibrahim. El-Din, Car Parking Problem in Urban Areas. Cairo: 1st International Conference on Towards a Better Quality of Life, 2017, p. 4, 5, 8, 9.
5. Angelica. Batrakova, Influence of Road Conditions on Traffic Safety. Kharkiv: Kharkiv National Automobile and Highway University, 2016, p. 1, 2.
6. Kathy. Schwalbe, Information Technology Project Management. Boston: Cengage, 2019, p. 473, 480, 482, 483, 484, 490.