MINISTRY OF EDUCATION
IMAM ABDULRAHMAN BIN
FAISAL UNIVERSITY
COLLEGE OF COMPUTER SCIENCE
& INFORMATION TECHNOLOGY

وزارة التعليم
جامعة الإمام
عبدالرحمن بن فيصل
كلية علوم الحاسب
وتقنية المعلومات

جامعة الإمام عبد الرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY

Kingdom of Saudi Arabia

Ministry of Higher Education
Imam Abdulrahman Bin Faisal University
College of Computer Sciences & Information Technology

Smart Parking Space Detection System for Imam Abdulrahman bin Faisal University Campus

A project submitted
in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Information Systems
by

Abeer Saad AlDkheel (2180000017)

Aisha Moraie Khubrani (2180005147)

Bashayer Mohammed AlYami (2180001758)

Ghadeer Saeed AlGhamdi (2200400065)

Ghadah Riyadh AlMuhaideb (2180000255)

Supervised by

Mr. Sardar Zafar Iqbal

Ms. Maryam AlNasser

Committee Member Names

Dr. Nesrine Isalah Mezhoudi

Dr. Saqeeb Saeed

April 2023

DECLARATION

We hereby declare that this project report entitled "Smart Parking Detection System for Imam Abdulrahman bin Faisal University" is a record of an original work done by us under the guidance of Supervisor Mr. Sardar Iqbal, Assistant Professor at College of CSIT, and that no part has been plagiarized without citations, which have been duly acknowledged. We also declare that it has not been previously or concurrently submitted for any other degree or award at any university or any other institution. This project work is submitted in the partial fulfillment of the requirements for the degree of "Bachelor of Science in Computer Information Systems at Computer Information Systems Department, College of Computer Sciences and Information Technology, Imam Abdulrahman Bin Faisal University.

Project Team:

Group Number: 2		
Aisha Moraie Khubrani	Aisha.	11 September 2022
Abeer Saad Aldkheel		11 September 2022
Bashayer Mohammed Alyami	Bashayer.	11 September 2022
Ghadah Riyadh AlMuhaideb		11 September 2022
Ghadeer Saeed Alghamdi	Ghadeer.	11 September 2022

Project Supervisor:

Name	Signature	Date
Mr. Sardar Zafar Iqbal		

ACKNOWLDGMENT

We would like to express our gratitude to everyone who implanted a passion in us during our study years. We are most grateful to our supervisors Mr. Sardar Zafar Iqbal, and our co- supervisor Ms. Maryam Alnasser for their continued support and guidance during the planning of this project. We extend our endless thanks to our project evaluators Dr. Nesrine Mezhoudi and Dr. Saqib Saeed for their cooperation, support, guidance, and suggestions to improve our project. A special thanks to all colleagues, friends, and workers in the technology field who provided advice, supported the idea, and expressed their kind opinion about our work contributing to obtaining better results.

ABSTRACT

In this report, we propose developing a smart Parking detection system using Computer vision and image-processing with the aid of the Internet of Things (IoT) to improve the quality of the parking lot area on the AlRakah campus. The report is divided into eight major sections. First, an introduction chapter will cover a problem statement, the project's goals and objectives, and the justification for them. The Project Management Plan, the second chapter, will describe how the project will be executed, monitored, controlled, and closed. The third chapter is the Requirements Specification, which will show what the software will do and how it will be expected to perform. The Design Specifications chapter will include both software and hardware design specifications for the proposed project. The fifth chapter is Implementation and Testing, which will explain the methods, tools, and techniques used to develop the project as well as how it will be tested and will conclude with the testing results. The sixth chapter will be about the implementation process and testing process. The seventh is about the deployment process and how to integrate everything together. Finally, the final chapter will be a report that concludes with lessons learned and future work. This project is the culmination of our knowledge and experience gained over the years; it has provided us with the opportunity to apply, improve, and apply this knowledge. We hope that this project will be an efficient solution to alleviating traffic, and fuel consumption, easing the trouble of finding available parking spots for drivers, and anyone else who may be interested in this mode of technology.

Table of Contents

LIST OF TABLES xii
LIST OF FIGURES xiv
List of ABBREVIATIONS xvii
Chapter 1: Introduction xix
1.1 Introduction 20
1.2 Problem Statement 21
1.3 Background 21
1.3.1 Smart parking system and parking monitoring 21
1.3.2 Deep Learning 22
1.3.3 Computer vision 22
1.3.3.1 Computer vision in transportation 22
1.3.4 Internet of Things -IOT- 22
1.4 Review of Literature 23
1.4.1 Smart Parking Monitoring 23
1.4.2 IoT Based Smart Parking System 24
1.4.3 Intelligent Parking Systems for Quasi-Close Communities 25
1.4.4 Deep Learning for Decentralized Parking Lot Occupancy Detection 25
1.4.5 Machine Vision Smart Parking Using Internet of Things (IoTs) 26
1.4.6 An Edge Based Smart Parking Solution Using Camera Networks and Deep Learning .28
1.5 Justification 33
1.6 Aims & Objectives 33
1.6.1 Aims 33
1.6.2 Objectives 33
1.7 Scope / Limitation of the Study 33
1.8 Professional and Ethical Implication References 34
1.9 Project Organization 34
Chapter 2: System Project Management Plan 36
2.1 Project Overview 37
2.1.1 Project Summary 37
2.1.1.1 Purpose, Scope, and Objectives 37
2.1.1.1 Purpose 37
2.1.1.2 Scope 37
2.1.1.3 Objectives 37

2.1.2 Assumptions and Constraints 38
2.1.3 Project Deliverables 38
2.1.4 Master Schedule and Budget Summary 39
2.1.5 Evolution of the Plan 39
2.2 Project Organization 39
2.2.1 Internal Structure 39
2.2.2 Role and Responsibilities 40
2.3 Project Roles and Responsibilities 41
2.4 Management Process Plans 42
2.4.1 Start-up 42
2.4.1.1 Estimation 42
2.4.1.2 Cost Estimation 42
2.4.1.3 Schedule Estimates 42
2.4.1.4 Staffing 42
2.4.1.5 Resource Acquisition 43
2.4.2 Staff Training 44
2.4.2.1 Work Planning 44
2.4.2.2 Work Activities 44
2.4.2.3 Schedule Allocation 45
2.4.2.4 Resource Allocation 46
2.4.2.5 Budget Allocation 46
2.5 Project Controls 47
2.5.1 Requirements Control 47
2.5.2 Schedule Control 47
2.5.2.1 Schedule Tracking 47
2.5.2.2 Schedule Performance Reports 47
2.5.2.3 Schedule Reviews 47
2.5.2.4 Progress Variance Monitoring 48
2.5.2.5 Progress Variance Resolution 48
2.5.2.6 Follow-up on Corrective Action 48
2.5.3 Budget Control 48
2.5.4 Quality Control 49
2.5.5 Project Reporting and Communication 49
2.5.5.1 Electronic Media 49
2.5.5.2 Meetings 49

5.5.5.3 Information Repository....50
2.5.5.4 Reviews....50
2.5.5.5 Status Reporting....50
2.6 Risk Management....50
2.7 TECHNICAL PROCESS....51
2.7.1 Process Model....51
2.7.2 Methods, Tools, and Techniques....53
2.7.3 Project Infrastructure....53
2.7.4 Product Acceptance....54
2.8 SUPPORTING PROCESSES....55
2.8.1 Independent Verification and Validation....55
2.8.1.1 Validation Scope....55
2.8.1.2 Tools and Techniques....55
2.8.1.3 Responsibilities....55
2.9 Documentation....55
2.10 Reviews and Audits....56
Chapter 3 Software Requirements Specification....57
3.1 Introduction....58
3.2 Purpose....58
3.3 Document Conventions....58
3.4 Intended Audience and Reading Suggestions....58
3.5 Product Scope....58
3.6 References....58
3.7 Overall Description....59
3.7.1 Product Perspective....59
3.7.2 Product Functions....60
3.7.3 User Classes and Characteristics....61
3.7.4 Operating Environment....61
3.7.5 Design and Implementation Constraints....61
3.7.6 User Documentation....61
3.7.7 Assumptions and Dependencies....62
3.8. External Interface Requirements....63
3.8.1 User Interfaces....63
3.8.2 Hardware Interfaces....63

3.8.3 Software Interfaces 63
3.8.4 Communications Interfaces 63
3.9 SYSTEM FEATURES 63
3.9.1 FUNCTIONAL REQUIREMENTS 63
3.10 USE CASE DIAGRAM 66
3.11 Other Nonfunctional Requirements 66
3.11.1 Performance Requirements 67
3.11.2 Safety Requirements 67
3.11.3 Security Requirements 68
3.11.4 Software Quality Attributes 68
3.12 Business Rules 70
Chapter 4 System Design Specification 70
4.1 Introduction 71
4.2 Purpose 71
4.3 Scope 71
4.4 Definitions, Acronyms and Abbreviations 71
4.5 Reference 72
4.6 System Overview 72
4.6.1 GENERAL FUNCTIONALITY 72
4.6.2 ESP32-CAM Camera 72
4.6.3 Database Functionality 72
4.7 DATABASE STORAGE 72
4.8 Design Considerations 72
4.8.1 ASSUMPTIONS AND DEPENDENCIES 73
4.8.2 GENERAL CONSTRAINTS 73
4.9 User Interface Design 74
4.9.1 Overview of User Interface 75
4.9.2 Interface Design Rules 75
4.9.3 Screen images for User 75
4.9.3.1 Walkthrough Interface 75
4.9.3.2 Welcome Interface 76
4.9.3.3 Register Interface 77
4.9.3.4 Login Interface 79
4.9.3.5 Recover Password Interface 79

4.9.3.6 Home Interface 80
4.9.3.7 Profile Interface 81
4.9.3.8 Edit Reservation Interface 83
4.9.3.9 Add car plate number interface. 84
4.9.3.10 select parking zone interface. 85
4.9.3.11 Select number of hours interface. 86
4.9.3.12 Reservation ticket interface 87
4.9.3.13 More interface 88
4.9.3.14 About interface 89
4.9.3.15. Settings interface 90
4.9.3.16. Contact Us 91
4.10 System Architecture 93
4.10.1 Architecture Design Approach 93
4.10.2 Architectural Design 93
4.10.3.Subsystem Architecture 94
4.10.4 General View of Parkin 94
4.11 Data Design 96
4.11.1. Data Description 96
4.11.1.1. User Data Table 96
4.11.1.2 ESP32-CAM Table 96
4.11.2. Data Dictionary 96
4.11.2.1. User Data Table 96
4.11.2.2 ESP32-CAM Table 97
4.11.3 Database Description 97
4.12 Component Design 99
4.13 Detailed System Design 100
4.13.1. classification 100
14.13.2. Definition / Responsibilities 100
4.13.3. Constraints 101
4.13. 4 RESOURCES 102
4.14 PROCESSING 103
4.14.1 System User Process 103
4.15. DETAILED SUBSYSTEM DESIGN 105
4.15.1. ESP32-CAM Data Flow Diagram 105
4.15.2. Database Data Flow Diagram 106

4.15.3. Mobile Application User 106
4.16 Requirements Traceability 107
CHAPTER 5: IMPLEMENTATION AND TESTING 109
5.1 Introduction 110
5.2 Testing plan 110
5.2.1 INTRODUCTION 110
5.2.1.1 Objectives 110
5.2.1.2 Testing Strategy 111
5.2.1.3 Scope 111
5.2.1.4 Reference Material 111
5.2.2 TEST ITEMS 111
5.2.2.1 Program Modules 111
5.2.2.2 Job Control Procedures 112
5.2.2.3 User Procedures 112
5.2.2.4 Operator Procedures 112
5.2.3 FEATURES TO BE TESTED 112
5.2.4 FEATURES NOT TO BE TESTED 113
5.2.5 APPROACH 113
5.2.5.1 Component Testing 114
5.2.5.2 Integration Testing 114
5.2.5.3 Interface Testing 116
5.2.5.4 Security Testing 116
5.2.5.5 Recovery Testing 119
5.2.5.6 Performance Testing 119
5.2.5.7 Regression Testing 121
5.2.5.8 Acceptance Testing 121
5.2.6 PASS / FAIL CRITERIA 121
5.2.6.1 Suspension Criteria 121
5.2.6.2 Resumption Criteria 122
5.2.6.3 Approval Criteria 122
5.2.7 TESTING PROCESS 122
5.2.7.1 Test Deliverables 122
5.2.7.2 Testing Tasks 122
5.2.7.3 Responsibilities 123
5.2.7.4 Resources 123

5.2.7.5 Schedule 123
5.3 Changes from the proposal phase and justifications. 124
5.4 Implementation progress 125
5.4.1 Implementation environment 134
5.4.2 Tasks progress during the implementation phase 135
5.6 Issues faced: 137
5.7 ENVIRONMENTAL REQUIREMENTS 137
5.7. 1 Hardware 137
5.7.2 Software 137
5.7.3 Security 138
5.7.4 Tools 138
5.7.5 Publications 138
5.7. 6 Risks and Assumptions 138
5.7.7 CHANGE MANAGEMENT PROCEDURES 140
5.7.8 PLAN APPROVALS 140
5.8 Software Test Report 140
5.8.1 Document overview 140
5.8.2References 140
5.8.2.1 Project References 140
5.8.2.2 Standard and regulatory References 141
5.8.3 Overview of Tests Results 141
5.8.3.1 Tests log 141
5.8.3.2 Rationale for decision 141
5.8.3.3 Overall assessment of tests 141
5.8.4 Impact of test environment 142
5.8.5 Detailed Tests Results 143
5.8.5.1 White-box Testing 144
5.8.5.2 Black-box Testing 151
CHAPTER 6: USER MANUAL 161
6.1 Overview of Parkin 162
6.2 Features 162
6.3 Requirements 162
6.4 Getting started 162
6.4.1 Hardware connection 162

6.4.1.1 ESP32-CAM connection 163
6.4.1.2 Wi-Fi Connection 163
6.4.2 Application Interfaces 163
6.5 Troubleshooting 165
6.5.1 (Forgot Password) 165
CHAPTER 7: DEPLOYMENT 166
7.1 Project life cycle 167
7.2 Deployment 167
7.2.1 Deployment constraints 168
CHAPTER 8: CONCLUSION 169
8.1 Conclusion 170
8.2 Lesson learned. 170
8.3 Future work 171
REFERENCES 172
Appendix A: Glossary 175
Appendix B: Response to Comments 176
Appendix C: BUSINESS MODEL CANVAS 177
Appendix D: Project Team Deliverables 178
Appendix E: Poster Acceptance 179

LIST OF TABLES

Table 1 APPRECIATIONS xvii
Table 2 summary of papers for the review of the literature. 29
Table 3 assumption and constraint 38
Table 4 project deliverables 38
Table 5 budget summary 39
Table 6 Project Roles and Responsibilities 41
Table 7 Staffing table. 43
Table 8 Staff Training 44
Table 9 budget of materials 46
Table 10 risk management plan 50
Table 11 Methods, Tools, and Techniques 53
Table 12 Product Acceptance 54
Table 13 Documentation Plan 56
Table 14 process of the project 60
Table 15 User Classes and Characteristics 61
Table 16 User Documentation 62
Table 17 FUNCTIONAL REQUIREMENTS 01 63
Table 18 FUNCTIONAL REQUIREMENTS 02 64
Table 19 FUNCTIONAL REQUIREMENTS 03 64
Table 23 FUNCTIONAL REQUIREMENTS 04 65
Table 24 FUNCTIONAL REQUIREMENTS 05 66
Table 25 Software Quality Attributes 68
Table 26 User Data Description 96
Table 27 ESP32 CAM Description Table 96
Table 28 User Data Dictionary Table 96
Table 29 Camera Data 97
Table 30 Component's Classification, Definition, And Responsibilities 100
Table 31 Component's Constraints and Composition 101
Table 32 Components Resources 102
Table 33 Requirements traceability 108
Table 34 features to be test. 112
Table 35 sign in successful test case 114
Table 36 Parking reservation successful test case 115
Table 37 Parking reservation unsuccessful test case 115
Table 38 Reset the forgotten password successful test case. 115
Table 39 Reset the forgotten password unsuccessful test case. 116
Table 40 Length of the Password 117
Table 41 successful test case of the password 117
Table 42 unsuccessful test case of the password 117
Table 43 successful test case of Authorized Admin 118
Table 44 unsuccessful test case of Authorized Admin 118
Table 45 Performance Test case 119
Table 46 load test Successful test case 120
Table 47 load test Unsuccessful test case 120

Table 48 Availability Successful test case 120
Table 49 Availability Unsuccessful test case 121
Table 50 resources 123
Table 51 schedule 124
Table 52 Changes and justification 124
Table 53 Implementation Environment 134
Table 54 Task Progress 135
Table 55 risks and assumptions 138
Table 56 plan approvals 140
Table 57 Project References 140
Table 58 Standard and regulatory References 141
Table 59 Overall assessment of tests 142
Table 60 difference between the expected condition and real condition 142
Table 61 Cyclomatic complexity Test case 1 144
Table 62 Full branch coverage Test case 1 144
Table 63 Cyclomatic complexity of Test case 2 145
Table 64 Full branch coverage Test case 2 145
Table 65 Cyclomatic complexity of test case 3 146
Table 66 Full branch coverage Test case 3 146
Table 67 Cyclomatic complexity of test case 4 147
Table 68 full branch coverage of test case 4 147
Table 69 Cyclomatic complexity of test case 5 148
Table 70 full branch coverage of test case 5 148
Table 71 Cyclomatic complexity of test case 6 149
Table 72 full branch coverage of test case 6 149
Table 73 Cyclomatic complexity of test case 7 150
Table 74 full branch coverage of test case 7 151
Table 75 requirements needed to operate ParkIn 162
Table 76 definitions 175
Table 77 Response comments 176
Table 78 Parkin Business model canvas 177
Table 79 Project Team Deliverables 178

LIST OF FIGURES

Figure 1 Project organization 35
Figure 2: Structure of the Team 40
Figure 3 Structure of the Organization 40
Figure 4 Work Breakdown Structure 44
Figure 5 Initial Phase Progress 45
Figure 6 Initial Phase Progress 45
Figure 7 Project Timeline 46
Figure 8 The System Development Life Cycle 52
Figure 9 Components of our project 59
Figure 10 use case 66
Figure 11 System Hiarchy 75
Figure walkthrough interfaces12 76
Figure 13 welcome interface 77
Figure 14 signup interface 78
Figure 15 confirmation sign up interface. 78
Figure 16 confirmation sign up interface 78
Figure 17 signup interface 78
Figure 18 login interface 79
Figure 19 confirming set password interface 80
Figure 20 reset password interface 80
Home interface Figure 21 81
Reservations interface Figure 22 82
Profile interface Figure 23 82
Figure 24 Vehicle interface 83
Figure 25 Edit profile 83
Figure26 successfully ending session message 84
Figure27 End session interface 84
Figure 28 car plate number interface 85
Figure 29 select parking zone interface. 86
Figure 30 select number of hours interface. 87
Figure 32 successfully reservation interface 88
Figure 31 reservation tickets interface 88
Figure 33 more interface 89
Figure 34 about interface 90
Figure 35 settings interface 91
Figure 36 contact us interface. 92
Figure37 Architecture Design of Parkin 94
Figure 38 Flowchart of Parkin 95
Figure 39 Database Diagram 97
Figure 40 Entity-Relationship Diagram 98
Figure 41 Relational Schema for User 98
Figure 42 Relational Schema for ESP32-CAM 98
Figure 43 system user process 104
Figure 44 system process user part1 104

Figure 45 system process user part2 105
Figure 46 camera data flow diagram 106
Figure 47 database data flow diagram 106
Figure 48 Sequence Diagram for The Mobile Application Users 107
Figure 49 Home Page Code Snippet -1 125
Figure 50 Home Page Code Snippet -2 126
Figure 51 Login Page Code Snippet.....126
Figure 52 Realtime database 127
Figure 53 Admin page code Snippet 127
Figure 54 Admin Interface of Parking App.....128
Figure 55 Login Interface of Parking App.....128
Figure 56 Add Parking Area Interface of Parking App 128
Figure 57 Admin Add Interface of Parking App 128
Figure 58 Add Sticker ID Interface of Parking App.....129
Figure 59 Home Interface of Parking App.....129
Figure 60 Select Number of Hours Interface of Parking App.....129
Figure 61 Select a Parking Area Interface of Parking App 129
Figure 62 Success Message Interface of Parking App.....130
Figure 63 Ticket Interface of Parking App.....130
Figure 64 Timer Interface of Parking App 130
Figure 65 My Reservation Interface of Parking App 130
Figure 66 No Reservation Interface of Parking App 131
Figure 67 End Session Message of Parking App.....131
Figure 68 User Info Interface of Parking App 131
Figure 69 Arduino IDE Code for ESP32 CAM.....132
Figure 70 Arduino IDE Code for ESP32 CAM.....132
Figure 71 (A + B) show frame of parking.....133
Figure 72 Main Hardware Code Snippet.....133
Figure 73 Main Hardware Code Snippet 2.....133
Figure 74 Control ESP32CAM.....134
Figure 75 writing and reading to firebase.....134
Figure 76 Arduino IDE 134
Figure 77 Android Studio 135
Figure 78 Python 135
Figure 79 Notepad++ 135
Figure 80 Flutter Platform.....135
Figure 81 Firebase NoSQL 135
Figure 82 (a) Flowchart and (b) Flow graph of Test case 1 144
Figure 83 (a) Flowchart and (b) Flow graph of Test case 2 145
Figure 84 (a) Flowchart and (b) Flow graph of Test case 3 146
Figure 85 (a) Flowchart and (b) Flow graph of Test case 4 147
Figure 86 (a) Flowchart and (b) Flow graph of Test case 5 148
Figure87 (a) Flowchart and (b) Flow graph of Test case 6.....149
Figure 88 (a) Flowchart and (b) Flow graph of Test case 7 150
Figure 89 camera connection 163

Figure 90 walkthrough page 163
Figure 91 sign up or sign in pages. 164
Figure 92 reservation pages 164
Figure 93 Forgot Password 165
Figure 94 Project life cycle 167
Figure 95 Deployment 168
Figure 96 Poster and Acceptance poster email 180

List of ABBREVIATIONS

Table 1 APPRECIATIONS

Abbreviation	Details
AI	Artificial Intelligence
CNN	Convolutional Neural Networks
LCD	Liquid Crystal Display
IoT	Internet of Things
RFID	Radio Frequency Identifiers
YOLO	You Only Look Once
SDS	Software Design Specification
SRS	Software Resource Specification
SPMP	Software Project Management Plan
DL	Deep Learning
DSI	Display Serial Interface
IDE	Integrated Development Environment
CPU	Central Processing Unit

USB	Universal Serial Bus
MQT	Materialized Query Table
LED	Light Emitting Diode
CSI	Camera Serial Interface

Chapter 1: Introduction

1.1 Introduction

Nowadays, we live in an era of advanced technology, with most manual jobs being replaced with automated solutions. As a result, these solutions have assisted many businesses in lowering their costs and error rates. The transportation sector, which has played a pivotal role in the world's economic development for more than a century, is one of the industries strongly impacted by technology. However, the industry is preparing to undergo a significant transformation. Transportation issues inside the city, such as traffic congestion, insufficient parking spaces, and road safety, were highlighted as a result of this development. Due to these issues, there have been several parking guidance solutions on the market. The expanding number of automobiles in metropolitan areas is rapidly exceeding the availability of parking spaces; as a result, parking has become a pervasive development concern [1].

This issue is readily apparent in urban areas, particularly in urban centers, shopping malls, open marketplaces, government facilities, healthcare facilities, schools, and academic institutions [2]. In these systems, many parking spots exist as local data silos, and drivers can't get information about them from afar [3]. In addition, data on parking in isolated areas cannot be communicated to a centralized platform for city parking administration. The parking management system enabled by computer vision technology can overcome the described challenges experienced by parking management systems of high quality. Due to the characteristics of the university campus in terms of size, internet infrastructure, and the presence of open spaces such as outdoor parking areas, image-processing based computer vision is a potential option for enabling parking management on the smart campus.

Initially, camera images and machine vision technologies are used to determine the parking status, as opposed to the prior method of employing embedded sensors to detect vehicles in the parking lot. Second, the system can provide a precise position, which is essential for locating parking spaces. Thirdly, a camera-based technique is typically deployed within a roadside or university parking zone. Consequently, unattended parking is advantageous in accurately identifying free, cost-effective automotive parking spaces.

This part of the report contains an introduction to the project, its history, the rationale behind its selection, the issues it aims to solve, and its anticipated results. In addition, a problem statement, project justification, goals and objectives, scope and

limitations, project structure, and a comprehensive project management plan will be included in the second chapter.

1.2 Problem Statement

Nowadays, with women driving, the daily arrivals at the AlRakah campus at Imam Abdulrahman University have increased to up to 10,000 students and staff driving. Therefore, parking spaces have become more occupied than ever. This problem poses a daily challenge to both students and staff alike, as they must make many rounds just to search out a vacant automobile parking space. As a result, drivers would waste their precious time and energy when they could be late for work or classes.

However, discovering parking spots around the university campus in real-time is important for the convenience of staff, students, administration, and even visitors. Furthermore, not just for convenience purposes, but such a sensible system can help vehicle owners within the AlRakah campus park in an available parking lot, saving time, improving parking utilization, reducing management costs, and alleviating holdups. In this context, the employment of IoT networks will allow the sharing of knowledge about the outdoor parking slots of the campus over the Internet, thus helping both drivers and security management.

1.3 Background

The Background discusses five subjects: Smart parking system and parking monitoring, Deep Learning, Computer vision, Computer vision in transportation, Internet of Things -IoT-, and ESP32-CAM with IoT. Also, we have sub subjects related to IoT like: Edge computing, Cloud computing, and Machine learning, and IoT in smart parking.

1.3.1 Smart parking system and parking monitoring

Briefly said, smart parking is a parking system that may incorporate counting sensors, cameras, or in-ground sensors. In order to determine if parking spaces are empty or occupied, these devices are typically placed adjacent to or implanted into parking spaces. Real-time data collection enables this. The information is subsequently sent to a smart parking mobile application or website, which informs its users of the availability. Some businesses additionally provide additional in-app data, like parking costs and locations. This provides you with the opportunity to investigate all of your parking options [4].

1.3.2 Deep Learning

Deep learning enables computational models with numerous processing layers to acquire knowledge and represent data at various levels of abstraction, simulating how the brain sees and comprehends multimodal information and implicitly capturing complex structures of big data. Neural networks, hierarchical probabilistic models, and other unsupervised and supervised feature learning algorithms are all part of the broad family of techniques known as deep learning. Deep learning approaches have seen a recent uptick in interest due to their demonstrated superior performance than previous state-of-the-art methods in several tasks as well as the amount of complicated data from various sources (e.g., visual, audio, medical, social, and sensor).[5]

1.3.3 Computer vision

Computer vision is a component of artificial intelligence systems that enables computers to extract useful information from a variety of input sources, including pictures, videos, and other types of media. In other words, computer vision enables computers to perceive, observe, and comprehend pictures like people, much as AI systems enable computers to think like humans. The transportation sector is not the only one to gain from computer vision technology. Many other industries, including healthcare, manufacturing, agriculture, retail, etc., are already being transformed significantly by computer vision systems [6].

1.3.3.1 Computer vision in transportation

Because of its numerous applications in fields like self-driving cars, traffic management, parking management, road condition monitoring, and other areas, computer vision or visual transport tracking is crucial to the future of the transportation sector. Better traffic management and increased road safety are the consequences as computer vision assists in delivering correct information such as traffic density, freeway traffic count, etc. Future public transit is likewise anticipated to rely heavily on computer vision technology [7].

1.3.4 Internet of Things -IOT-

The term "Internet of Things," or "IoT," describes the complete system of interconnected devices as well as the technology that allow for cloud-based and device-to-device communication. We now have billions of devices connected to the Internet as a result of the development of low-cost computer processors and high-bandwidth phones. As a result, commonplace equipment like robots, vehicles, and vacuum cleaners may

employ sensors to gather data and communicate with users directly. The Internet of Things links commonplace "things" to the web.

IoT systems can use the following technologies:

- **Edge computing:** The term "edge computing" describes technology that enables smart devices to perform tasks rather than just send or receive data to their IoT platforms. It increases computing power at the edge of IoT networks, reducing communication latency and speeding up response times.
- **Cloud computing:** Cloud technology is used for remote data storage and IoT device management, making data available to multiple devices on the network.
- **Machine learning:** The software and algorithms used to analyze data and make real-time decisions are called machine learning. These AI algorithms can be used at the edge or in the cloud [8].

IOT in smart parking

An IoT-based smart parking system is a centralized management platform that enables drivers to look for and book parking spaces using a smartphone app. Parking place availability may be detected using IoT technology both indoors and outside. Smart Parking is a system use cameras that can identify parking spots both inside and outside [9][10].

1.4 Review of Literature

1.4.1 Smart Parking Monitoring

Ezekiel et al. [12] develop and analyze an intelligent smart parking system based on computer vision and internet of things technologies, with applications to gridlock issues caused by vehicles in parking lots. The basic dataset for the proposed system begins with the collection of images of various vehicles gathered from the faculty of science parking lot at Rivers State University. The first technique uses a convolution neural network algorithm in conjunction with a Haar cascade classifier to detect many vehicles in a single image and place rectangular boxes on the identified vehicles. They achieved a 99.80% accuracy. The second model, the Mask R-CNN model, was used to identify various cars by drawing a bounding box around each detected vehicle and downloading a pre-trained model built on the coco dataset to distinguish various items in videos and images. Finally, utilizing less graphics cards, the trained model was able to detect vehicles and parking spaces in a high-quality video.

Furthermore, Suhaila et al. [13] developed a cost-effective IoT smart parking system to monitor city parking places and offer drivers real-time parking information. The

parking status is determined using camera images and machine vision technology. They used machine vision to create a fully linked prototype for motion tracking and canny edge detection using OpenCV technology. Besides, data is transferred to an IoT platform called Ubidots for prospective activity monitoring, and an Android mobile application is designed for customers to access real-time parking data. At the same time, the developed mobile application allows users to easily look for empty car parking spaces, thereby reducing traffic difficulties such as illegal double parking. Moreover, the system has been tested in an actual outdoor parking area in a variety of weather conditions, including clear skies and rain. The detection accuracy achieved was 96.40%, indicating that this method is effective. Finally, with adequate night illumination, nighttime detection can theoretically reach the same accuracy as daytime. The accuracy of this technology can be enhanced in the future by employing a better camera and a speedier processing unit. indicating that this method is effective. Finally, with adequate night illumination, nighttime detection can theoretically reach the same accuracy as daytime. The accuracy of this technology can be enhanced in the future by employing a better camera and a speedier processing unit.

1.4.2 IoT Based Smart Parking System

A. Khanna and R. Anand. [14] aimed to lower traffic congestion, limit car parking facilities and road safety by the help of IoT. The main objective of this project is creating a smart parking system based on IoT, which is a module that is used to track and signalize each parking space's availability on-site. With the aid of distant computers connected via the Internet, those gadgets might be followed, managed, or watched over. This project showed no dataset. However, they carried out the project by employing ultrasonic sensors to find cars and using the processor-on-a-chip Raspberry Pi. Also, the processing unit functions as a go-between for the sensors and cloud, and the cloud-hosted IBM MQTT server serves as a database to hold all the information about parking lots and system users who have access. The results they gained are information that is current about the availability of parking spaces in a parking area, and the usage of the smartphone application allowed users in far-off places to reserve a parking space for themselves. The limitation of this project is limited capacity for processing. However, the data gathered from various sensors is typically sent to stronger nodes where it may be gathered and processed.

1.4.3 Intelligent Parking Systems for Quasi-Close Communities

A. Faiyetole. [15] aimed to create an intelligent parking system for Quasi-Close Communities. The experimental design and need justifications for a localized intelligent parking system (L-IPS) are shown in this research. This system is suited for quasi-close communities with growing motor volumes that rely on fixed or limited parking spaces, by boosting the number of vehicles that rely on finite or permanent parking spaces and addressing concerns with infrastructure-to-driver (I2D) communication and parking detection. The first step in their methodology was to create a systematic questionnaire that was sent to staff and students on the FUTA campus and asked about how difficult it was to locate parking spots at the recommended facility and how easy it was to find spots at the preferred facility. The respondents were asked about their acquaintance with the IPS as well as their opinions about its capability to control the flow of vehicles into and out of the parking area.

Additionally, an Arduino Nano Microcontroller, Arduino Ultrasonic Range Detection Sensors, Jumper Wires, Liquid Crystal Display with an I2C Chip, LED, and regulated power supply were used. It was also revealed that IPS is useful in assisting with the parking problem, despite the fact that few students are unfamiliar with it due to their difficulty navigating it.

1.4.4 Deep Learning for Decentralized Parking Lot Occupancy Detection

This paper proposes a decentralized and efficient solution for visual parking lot occupancy detection, based on a deep Convolutional Neural Network (CNN) specifically designed for smart cameras. The performance of their proposed CNN architecture is comparable to the well-known AlexNet, which is three orders of magnitude larger.

A smart camera is a vision system that has enough computing power to process and extract application-specific information from the captured images. A single smart camera can simultaneously monitor several parking lots at a cost significantly lower than the cost required to install and maintain sensors in each parking lot.

Deep Learning is a branch of AI that aims at developing tech that allow computers to learn complex perception tasks, such as seeing and hearing, at hu man level of accuracy. It provides near-human level accuracy in image classification, object detection, speech recognition, natural language processing, and more. It can be employed not only to classify specific objects, but also to recognize previously unseen objects [16].

They have compared mAlexNet with the method proposed in de Almeida, a state-of-the-art approach for car parking occupancy detection that relies on RBF kernel SVMs. They performed different experiments to investigate different aspects of the proposed solution. They trained and tested their proposed model mAlexNet on each of the three training sets individually, and at the end of the training phase, they tested each trained model on all three testing sets. They computed two evaluation metrics as in de Almeidaet . The first is the accuracy on the test set when choosing the most confident class as output (which is using a threshold of 0.5 for a bi-nary classification problem) and the second one is the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curves.

A team of researchers has presented a solution for visualizing the occupancy of parking spaces, which exploits deep Convolutional Neural Networks (CNNs) to classify the parking space occupancy. The only information sent to a central server for visualization is the binary output [17].

1.4.5 Machine Vision Smart Parking Using Internet of Things (IoTs)

The authors were approached to create a short-term solution for on-campus parking while the location and construction of new parking lots are decided at Christopher Newport University. Other options, such as public transportation, were determined to be ineffective on the campus due to its small size. The average US driver wastes seventeen hours a year searching for parking, costing an estimated $72.7 billion a year from fuel and emissions, or $345 per year per driver. This is a large sum of money that drivers lose every year due to the struggles of parking and can be alleviated by a smart parking system. The Park Fast smart parking system uses a database to store data on the parking spots. Other researchers have used RFIDs in their parking applications. In a survey published by Lin et. Al, they claim that if drivers can have real-time parking availability information, they will be able to adjust their traveling schedule. Alam et. Al. suggest using smart real-time parking systems in their paper [18] [19].

Their system uses magnetic sensors instead of cameras to provide real time data. The paper did not provide the cost it would take to implement the system, so the cost effectiveness of their system cannot be compared.

Spark provides a feature that allows users to reserve parking spots in advance. Such a feature would be useful to implement in ParkFast. Bura et. Al. describes a smart parking solution using cam- eras with a full view of a parking lot connected to edge devices that

will gather the occupation status of parking spots. It uses a map of campus parking lots to indicate which lots are open and which ones are closed. The app stores this data for later, changing the availability of certain lots based on the user's choice. The researchers implemented a novel technique in their application in the form of a map overlay of the parking lots. A close-up view of a single parking lot is shown with each parking space highlighted and outlined. This will allow the user to easily identify and navigate directly to an open parking spot.

The app was developed using Swift 5.0 to accomplish the objective that the app be deployable on an iOS platform. The user's choices on the login screen are stored via core data, eliminating the need to store and later access the data from a separate database. The parking tracker software uses multiple programming languages, some more than others. Most of the program is written in Python 3.6 which will be necessary for the OpenCV

software. Each parking space displays a given color at any point in time and is liable to change if the requirements are met.

The process requires representative data sets to be used for training and validation. Each frame that is processed goes through a Gaussian Filter. This is a linear filter that will assist in blurring and reducing noise in the image (Gaussian Filter Equation). To detect shadows, they use background subtraction on top of the Gaussian and Laplacian filters.

They use a machine learning based approach where a cascade function is trained from a multitude of positive and negative images. It is then used to detect objects in other images. The computer being used to process the parking tracker program is the NVIDIA Jetson Nano, 70 x 35 mm (70 x 35 ins). The Wi-Fi network operates on 2.4GHz. The power source that runs the Nvidia Jetson and the Ubiquiti antenna is an ExpertPower 12V 12AH Sealed Lead Acid (SLA) battery. To regulate the charge capacity of the SLA battery from the solar panel, they attached a Renogy Wanderer 10A 12V/24V PWM solar controller. There are two classifiers used with two different image sets of cars. The reaction time for a region of interest to detect and switch contour colors took only a couple of frames. If they increase the height of the webcam, then the chances of a car cutting into an area of interest decreases.

Siri Shortcut feature would allow the user to say a customized phrase and be directed towards the nearest parking lot with an open space. It would only function if

the user is physically within a mile of the campus. Not all parking lots on the university campus are equipped with the technology required to become compatible with the system.

1.4.6 An Edge Based Smart Parking Solution Using Camera Networks and Deep Learning

As much as 30% of traffic congestion is caused by drivers looking for parking spaces in congested city areas, according to research. The system can automatically detect when a car enters the parking space and, the location of the parking spot.

Smart parking solutions are one of the basic and foremost requirements of smart cities. Installing sensors in every parking spot is a simple way to solve the automatic parking spot detection problem. Autonomous vehicles are another solution to solve parking problems, but they also pose significant challenges for infrastructure. If a car can't find a parking spot it will end up circling the lot looking for one, blocking other spaces and wasting fuel. Some solutions using vision-based computer vision have been proposed to solve the parking spot problem, but they are expensive and require highly complex equipment.[27]

Cameras with wide angle fish-eye lens will cover large parking spaces and detect the parked parking lot via artificial intelligence. Deep learning algorithms will be used to identify occupied, vacant, and special parking spots. The overall system will help motorists in navigating to a vacant spot with ease and will also help in charging them based on the exact length of time the vehicle has been parked in the parking space.[28]

They choose the low-cost Raspberry Pi (Rpi) camera and fit with a zoom lens. The edge devices used in their system are Raspberry Pi and Nvidia Jetson Tx2. The Rpi interfaced with the ground camera runs OpenALPR to perform license plate number recognition. The Jetson which receives the live feed time-stamps the entry of a vehicle and extracts its features [29].

A custom network model has been developed for identifying vacant parking spaces in parking lots. The inference time for this model is much faster than the standard AlexNet or mAlexNet. the future work includes increasing the speed and accuracy of the object detection model and establishing a link between ground cameras and top-view cameras to achieve object tracking [30].

Table 2 summary of papers for the review of the literature.

Study Name	Objectives	Dataset	Methods	Results	Limitations
Smart Vehicle Parking System Using Computer Vision and Internet of Things (IoT)	Recognize and detect both allocated and unallocated parking spaces using an intelligent system for smart car parking using internet of things and computer vision	Coco dataset	Haar cascade classifier, Mask R-CNN	A CNN model trained to identify various objects in videos and pictures. automatically detects vehicles. Tests were applied, and detection accuracy reached 99.8%	• The dataset was compiled from only one test environment. So, future work would include expanding the dataset. • The biggest limitation of R-CNN is that it takes a long time to train and cannot identify all vehicles tested on high-quality high-definition video.
Machine vision based smart parking system using Internet of Things	Presented a low-cost IoT smart parking system to monitor city parking spaces and send vehicles with real-time parking information	No dataset	Raspberry Pi 3 5-megapixel camera,	This method has been tested on an actual outdoor parking area in a variety of weather conditions, including clear skies and rain. The detection accuracy achieved 96.40%.	The accuracy of this method can be further improved by using better camera and faster processing unit.
Deep learning for decentralized parking lot occupancy detection	Decentralized and efficient solution for visual parking lot occupancy detection based on a deep Convolutional Neural Network (CNN) specifically designed for smart cameras.	It has a dataset; this dataset is publicly available to the scientific community and is another contribution of their research.	The approach that the paper proposes is to perform this task in real-time directly on smart cameras, without using a central server. It is a decentralized, effective,	A decentralized and efficient solution for visual detection of the parking status was presented, which exploits deep Convolutional Neural	• it is hard to think of general, robust, reliable features, which map to specific object types • it is a huge task to find the right combination of

			efficient, and scalable approach, based on deep learning. It relies on a deep Convolutional Neural Network (CNN) specifically designed to be executed on smart cameras.	Networks (CNNs) to classify the parking space occupancy. The solution employes smart cameras built using Raspberry Pi platform equipped with a camera module. Each smart camera can simultaneously monitor up to fifty parking spaces.	features for every type of object to classify • it is difficult to design functions that are robust to translations, rotations and scaling of objects in the image.
IoT Based Smart Parking System	On-site deployment of an IoT module that is used to monitor and signalize the state of availability of each single parking space.	No dataset	Using ultrasonic sensors to detect the presence of a car. Using Raspberry pi which is a processor on chip. The processing unit acts like an intermediate between the sensors and cloud. IBM MQTT server hosted on the cloud that acts as a data base to store all the records related to parking areas and end users	Real-time information regarding availability of parking slots in a parking area. Users from remote locations could book a parking slot for them by the use of our mobile application	Limited processing capabilities. However, data collected from various sensors is usually transmitted to more powerful nodes where its aggregation and processing

			that have access to the system.		
Intelligent Parking Systems for Quasi-Close Communities	Increasing vehicular volume that depends on limited or constant parking facilities and resolve aspects of infrastructure-to-driver (I2D) communication and parking detection issues.	No dataset	A structured questionnaire on the difficulty of finding parking spots on the recommended facility and ease of finding spots on the preferred facility was administered on the FUTA campus to staff and students. As well as familiarity with IPS, perceptions on the IPS' capacity in managing vehicular flow to and out of the parking zone were posed to the respondents. Also, Arduino Nano Microcontroller, Arduino Ultrasonic Range Detection Sensors, Jumper Wires, I2C Chip Liquid Crystal Display, LED and Regulated Power Supply were used..	IPS is valuable to help resolve the parking issue.	Few students are not familiar with IPS, and they found it difficult to deal with.

Machine Vision Smart Parking Using Internet of Things (IoTs)	create a short-term solution for on-campus parking while the location and construction of new parking lots.	It has dataset	presented a smart parking system that utilizes a server which collects data from ultrasonic sensors and transmits it to an application, called "ParkX," to help users expedite the parking process.	The Haar Cascade classifier was trained on images of Luter parking lot from a skewed angle. To test the algorithm, they mounted a webcam on top of one of the parking garages at CNU. Images from the third story of CNU parking garage were used to train a classifier that determines whether a parking space is open or closed. The drawback with testing at the height the camera is visualizing from is the	camera height, as it must be adjusted for various sized lots.

				chances of a car cutting into a region of interest decrease.	

1.5 Justification

The problem of congestion, traffic and less space in university can be reduced due to our project. It will improve parking by decreasing the rotational distance that drivers must drive to find a parking space in the parking of the IAU university, which will save time because the drivers will already be aware of their destination and the best route to take. Also. Our system will provide a user interface that allows users to access information such as parking availability, as well as reserve a parking space. Unlike the standard parking system, which only provides a screen that displays availability and does not address the rotation issue.

1.6 Aims & Objectives

1.6.1 Aims

- To develop an application where vehicles' drivers can be able to identify the available and busy parking spaces, by the help of a detective camera located near those parking areas.

1.6.2 Objectives

- Reserve a parking space when the vehicle driver is close to the parking area.
- Reduce fuel consumption.
- Reduce carbon dioxide emissions.
- Saving the driver's time.

1.7 Scope / Limitation of the Study

The first limitation is the uncertainty over the parking area, there must be a certain amount of knowledge of parking architecture map in order to utilize the parking management system, thus we didn't have the building map necessary to create the model. The second limitation is the car parking sheds, the camera can't detect the parking because of the sheds, so we need more cameras to put it at a different angels/corner. [20]

1.8 Professional and Ethical Implication References

Our system contains social and ethical vulnerabilities.

- **Social vulnerability:** People get frustrated when they go to a place, and they find out that the parking lot is full. When they, for example, have an important appointment and they arrive before the appointment, and they didn't find a parking spot, they get anxious and upset.
- **Ethical vulnerability:** based on the current parking system, the parking spot does not contain a reservation method; it is based on who comes first. Therefore, some problems might happen, such as traffic.

Our system has positive implications for different vulnerabilities in several fields, such as:

- **Social:** The system helps people go to the places they want while making sure that they will be able to arrive at their appointments on time and their parking spot is reserved for them.
- **Ethical:** The system supports the environment by reducing the amount of traffic that is caused while searching for a parking spot.
- **Security:** Protect data from unauthorized access by using security and privacy procedures.

1.9 Project Organization

This chapter sums up the document with a summary of the project's plan, the document will be organized as the following:

Chapter 1: This section provides a brief description of the project's primary idea (abstract), project introduction, problem statement, background and study of literature, justification, aims and objectives, scope, and limitation, Professional and Ethical Implications, and project organization.

Chapter 2: Project Management Plan (SPMP) This chapter discusses the project's Project Management Plan. It is concerned with the project model, work breakdown structure, schedule, budget, and risk management, project monitoring and control plans, resource requirements, and so on.

Chapter 3: Specification of Requirements (SRS) this chapter contains a full summary of the project's requirements. It will go over the system, functional and nonfunctional needs, limitations, and assumptions, as well as other details.

Chapter 4: Design Specification (SDS) this chapter is mostly concerned with software and hardware specifications. Design concerns, user interface design, system architecture, and component design are all covered.

Chapter 5: Testing and Implementation This chapter will be divided into three sections: Explain the methodologies, tools, and strategies utilized to construct the software in the first subsection. Explain how the suggested program will be tested in the second subsection (Test plan). Explain the test findings as well as the real results of test cases in the third subsection (Test report).

Chapter 6: Conclusion this is the final chapter; it will complete the entire project by providing information about the issues encountered, the resulting solutions to the problems, lessons learned, recommendations for future work, and so on. Figure 1 illustrate the timeline of the project cycle.

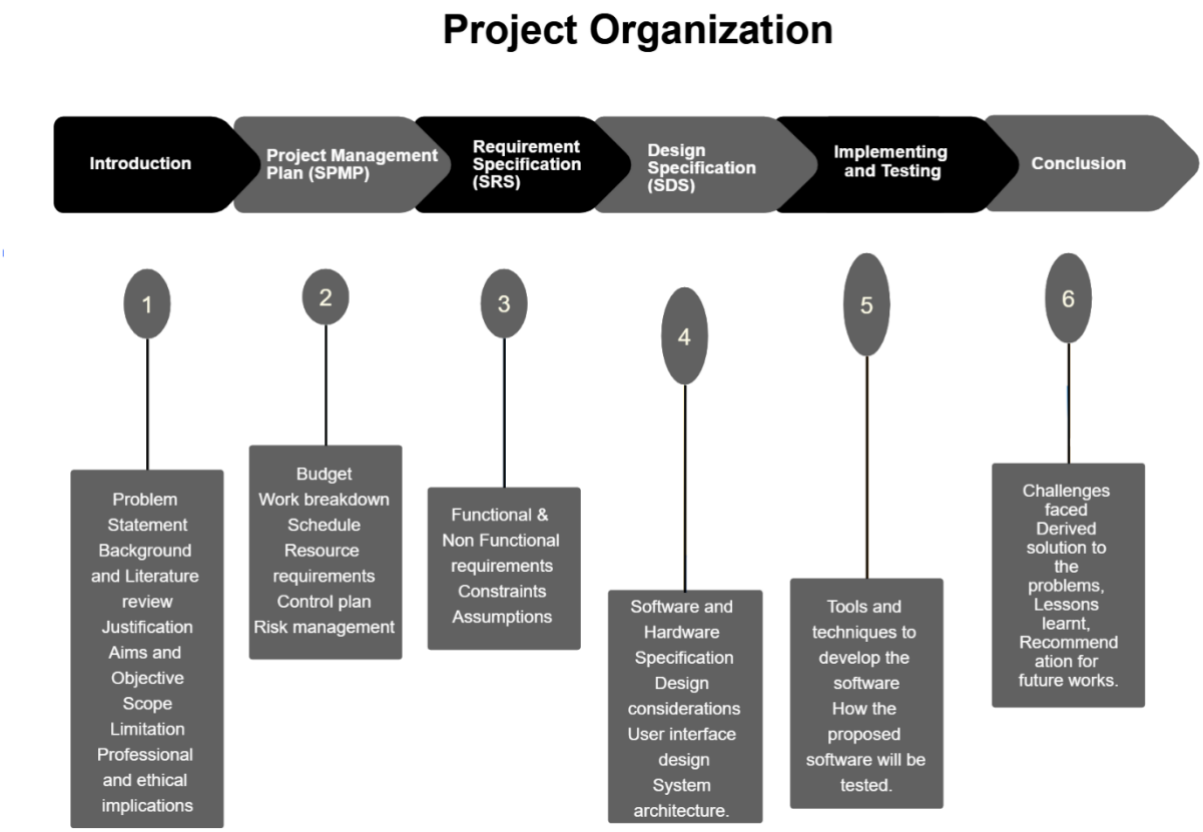


Figure 1 Project organization

Chapter 2: System Project Management Plan

2.1 Project Overview

This section provides an overview of the project's purpose, scope, and objectives, as well as project assumptions and limitations, a list of project deliverables, and a timeline and budget summary.

2.1.1 Project Summary

Using ESP32-CAM with help of computer vision and Image-processing, we propose to create a smart parking system that will be able monitor the availability of parking space in real-time, so the camera detect free parking spaces and direct drivers to the most convenient spot. This information will provide detailed, live information around hours to show real-time location through an application on a smart phone.

2.1.1.1 Purpose, Scope, and Objectives

2.1.1.1 Purpose

. Our project intends to develop an application that will allow vehicle drivers to identify available and occupied parking spots and detect parking space with the aid of a camera placed near parking spaces. We believe that smart parking detection systems can make a significant difference in the transportation field, given the importance of alleviating traffic congestion.

2.1.1.2 Scope

This section discusses essential components of the project, including its purpose, scope, and objectives, project organization, managing process plans, technical process plans, documentation, and other procedures. The goal is to develop a mobile application for a smart parking system using a computer vision model for collecting parking spots. The Hough circle function will be utilized in this model due to its high rate of accuracy achieved in prior work and compared to other methods. The used data is a collecting from our sides.

2.1.1.3 Objectives

- Automate the process of monitoring parking spaces to save time and effort for the security guard. Ensure the safety of the vehicle and the driver in the parking area.
- Using computer vision models to detect objects, the university can better manage parking spaces and reduce congestion without incurring significant costs.
- Achieve the highest possible level of accuracy.

2.1.2 Assumptions and Constraints

Table No.3 contains assumptions, constraints, for the success of the project.

Table 3 assumption and constraint

Assumptions	The project will be completed by the due date. The team is cooperative and capable of self-learning. The skill set of the team members will aid in the development of a reliable and precise mobile application. We anticipate that by the end of the project, both the application and the hardware components will be fully operational. Evaluate different computer vision algorithms and implement the one with the most efficient results. The driver will be able to reserve a spot for one car at a time.
Constraints	Insufficient time to complete the project. Few local businesses carried the essential hardware materials, indicating a lack of available resources. The availability of the data set could take longer than anticipated. Camera issues in the covered parking areas due to parking sheds.

2.1.3 Project Deliverables

Table No.4 displays the deliverables of the project's that are required to be completed.

Table 4 project deliverables

Deliverables	**To whom**	**Format**	**Date**
Proposal discussion	Mr. Sardar Iqbal	Oral – Zoom meeting	6 September, 2022
Problem statement, scope, aims and objectives	Mr. Sardar Iqbal	Softcopy	22 September, 2022
Project Plan (SPMP)	Mr. Sardar Iqbal	Softcopy	25 September, 2022
Mid-Semester Report	Mr. Sardar Iqbal	Softcopy	29 September, 2022
Project Requirement (SRS)	Mr. Sardar Iqbal	Softcopy	16 October, 2022
Project Design (SDS)	Mr. Sardar Iqbal	Softcopy	23 October, 2022
Final Design	Mr. Sardar Iqbal	Softcopy	30 October, 2022
Final Report	Mr. Sardar Iqbal and Committee Members	Softcopy	6 November 2022
Project Presentation	Mr. Sardar Iqbal and Committee Members	Oral – Zoom meeting	10 November, 2022

2.1.4 Master Schedule and Budget Summary

To estimate the cost of utilizing the tools as shown Table No.5, we calculated their prices, which yielded a total of 484 Saudi Riyals.

Table 5 budget summary

Tools	**Price**
ESP32 CAM	75 SR
OV7670 300KP Camera	42 SR
X2 Micro servo 9g	154 SR
Camera holder	40 SR
Jumper wires	27 SR
Breadboard	23 SR
GPS	60 SR
Resistors	34 SR
USB: A-B	10 SR
LED	19 SR
Subtotal	484 SR

2.1.5 Evolution of the Plan

Throughout the system development process, our strategy may be adjusted. Before making any modifications, Mr. Sardar Iqbal, our advisor, shall be contacted for his consent and counsel. The entire team will discuss the most recent developments. Because this is the first edition of the System, modifications are possible.

2.2 Project Organization

This section identifies the project's external interface, internal organizational structure, and each member's exact roles and responsibilities.

2.2.1 Internal Structure

The internal structure contains 5 members. Therefore, the team members will have to find a suitable communication method to be able to discuss and distribute the work equally between each other. The group will be able to meet each other face to face and online, online meetings are more efficient therefore using online applications such as Zoom, FaceTime, and Microsoft Word shared file are recommended.

2.2.2 Role and Responsibilities

This section of the internal structure will list the hierarchal order of the team and organization structures how team members divide and coordinate the work and tasks among each other as shown in the figures below (Figure No.2-3).

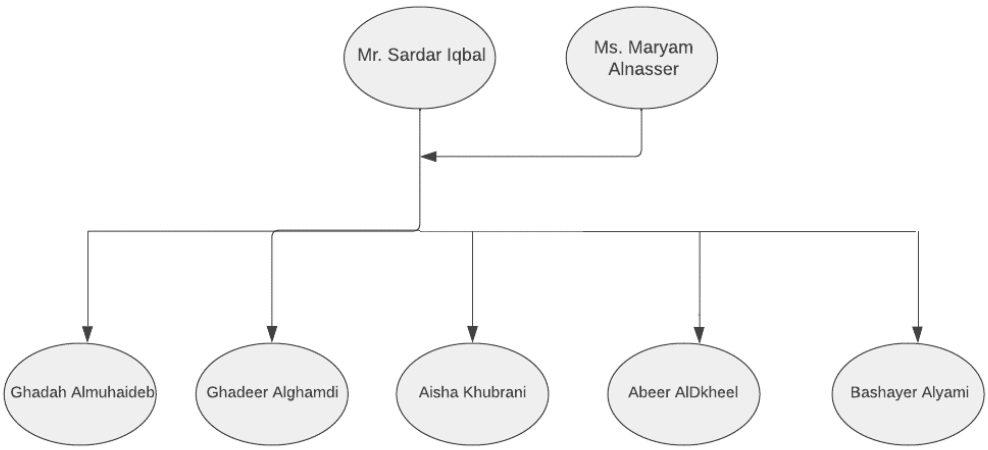


Figure 2: Structure of the Team

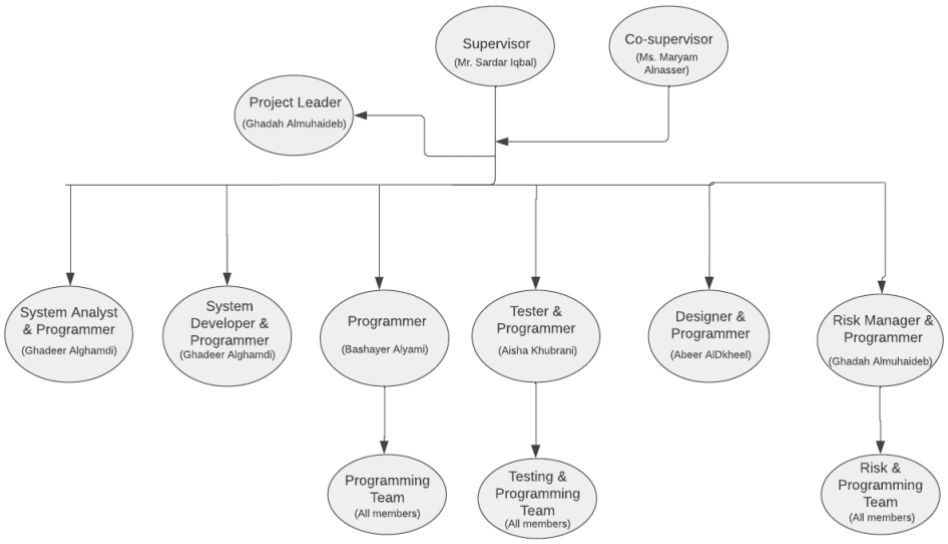


Figure 3 Structure of the Organization

2.3 Project Roles and Responsibilities

As roles and responsibilities is considered a crucial part in the project, each team member has been assigned several roles, with each role containing multiple responsibilities.

Table No.6 contains the roles and responsibilities of team member.

Table 6 Project Roles and Responsibilities

Role	Responsibilities	Name
Supervisor, Co-supervisor	• Leading the graduation project. • Outline the project goals. • Conduct weekly meetings with the members through zoom. • Review the member's tasks. • Provide recommendation based on his review. • Manage and minimize potential risk.	Mr. Sardar Iqbal. Ms. Maryam Alnasser.
Project Manager	• Divide work equally between team members. • Solve the member's conflict. • Encourage the team members. • Make sure that each member meets their specified deadline.	Ghadah Almuhaideb.
System Analyst	• Analyze the project. • Collect the requirement. • Analyze the budget and cost. • Analyze the system. • Document the system analysis.	Ghadeer Alghamdi.
Programmer, System Developer	• Develop the needed code for the system. • Test the code. • Connect the hardware with the software.	Bashayer Alyami. Ghadeer Alghamdi. Ghadah Almuhaideb. Aisha Khubrani. Abeer AlDkheel.
Tester	• Test the equipment. • Fix any bugs or errors. • Write the testing documentation.	Aisha Khubrani.
Designer	• Provide high quality design. • Design the interfaces. • Confirming the system requirements. • Document the final design.	Abeer AlDkeel.
Risk Manager	• Predict any crisis, problems, and difficulties. That might happen in the future. • Provide solution for future crisis. • Make decisions based on these crises.	Ghadah Almuhaideb.

2.4 Management Process Plans

The project management procedures for this project will be depicted in this section of the Management Process Plans. This part contains the plans for the start-up, work, and project tracking. In addition, the risk management and project close-out plans will be defined in this part.

2.4.1 Start-up

The start-up plan will take into consideration the expected time for completion of the project, the personnel required, and the cost. In along with assessing the abilities of the individual members

2.4.1.1 Estimation

2.4.1.2 Cost Estimation

The team members did research in order to find adequate hardware equipment for the system, either locally or online, and Geeks Valley was chosen based on multiple comparisons between different countries, websites, and stores. The rule for identifying the cost estimations:

(Quantity * price) = product price

product price + VATs= total price

Based on our calculations, team members decided that geeksvalley. products were largely appropriate in terms of availability, quality, and cost.

2.4.1.3 Schedule Estimates

This project preparation and implementation will take two semesters. The first semester of planning is spent on the design stage, while the second semester is spent on the implementation stage. Each deliverable will be allocated a precise date once the project's syllabus has been approved by the supervisor.

2.4.1.4 Staffing

Table No.7 will provide the following information: human resource type, how many working hours per week, the key period, phases including the number of staff that is working on the project.

Table 7 Staffing table.

Human Resource Type	**Work (hrs.) Per Week**	**Key Periods**	**Number of Staff**	**phases**
Supervisor	2	From 1/09/2022 to 25/12/2022	1	All phases
Project Manager	32		1	All phases
System analyst	30		1	- Initiating - Analysis
Programmer - system developer	32		1	- Programming - Executing
Tester	30		1	- Executing - Testing
Designer	30		1	- Designing - Documentation

2.4.1.5 Resource Acquisition

In this project, two main types of resources are required: hardware and application.

Hardware:

- Laptop
- ESP32-CAM
- Wifi

There will be a variety of web-based applications installed with the software. Several open source

applications such as:

- Arduino IDE
- Notpad++

2.4.2 Staff Training

In order to comply with the system requirements, some skills need to be learned or improved, shows in detail the needed tasks in this project, number of members that need to obtain this task, what source will be helpful to improve these tasks, the team members level in each task, and the training period.

Table 8 Staff Training

Task	Team members	Source	Skill level	Period
Using different programming languages (paython, C++)	**All members**	**Online tutorial**	**Intermediate to high**	**3 Years**
Building the hardware	**All members**	**Online tutorial**	**Intermediate**	**2 months**
Programming the cameras	**All members**	**Online tutorial**	**Beginner**	**1 month**
Developing the Application on Visual code studio	**All members**	**Previous course**	**Intermediate**	**5 months**
Developing the Database	**All members**	**Previous course Online tutorial**	**Intermediate to high**	**5 months**

2.4.2.1 Work Planning

The following paragraphs provide a working management plan for the acquisition of the Smart Parking Space Detection System.

2.4.2.2 Work Activities

So far, there are other phases explained in the plan. However, the figures below (Figure 4,5 and 6) are listing the planning steps and the work breakdown structure

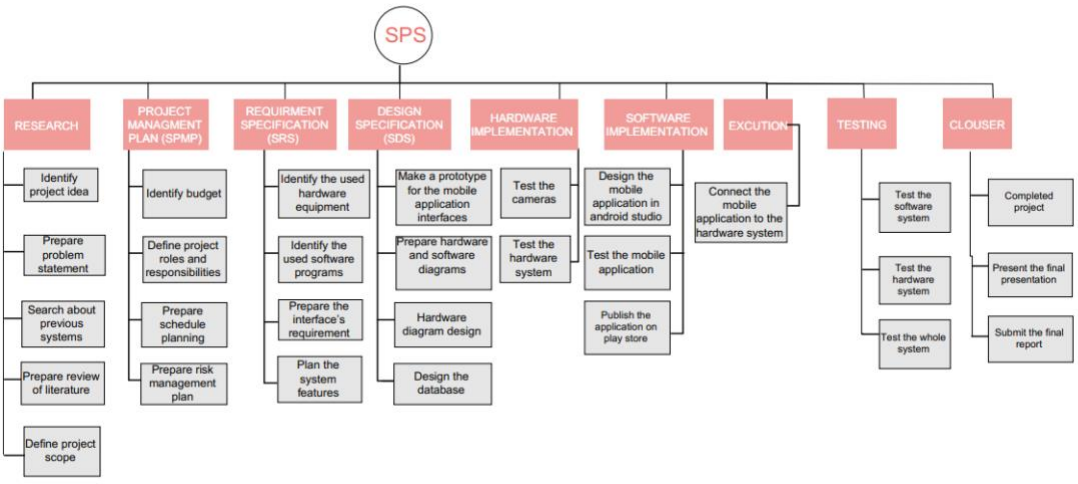


Figure 4 Work Breakdown Structure

Title	Type	Duration	Start	End	%	Assigned to
Discussion	T	7 days	01/09/2022	09/09/2022	100%	
Identify project idea	T	3 days	09/09/2022	13/09/2022	100%	
Project proposal	T	4 days	13/09/2022	16/09/2022	100%	
Problem statement	T	4 days	16/09/2022	21/09/2022	100%	
Project management plan (SPMP)	T	3 days	21/09/2022	25/09/2022	100%	
Midterm report	T	7 days	21/09/2022	29/09/2022	0%	
Project requirement (SRS)	T	5 days	29/09/2022	05/10/2022	0%	
Final design	T	3 days	05/10/2022	09/10/2022	0%	
Final report	T	9 days	03/10/2022	13/10/2022	0%	

Figure 5 Initial Phase Progress

Title	Type	Duration	Start	End	%	Assigned to
Project presentation #1	T	5 days	19/10/2022	25/10/2022	0%	
Hardware implementation	T	25 days	26/10/2022	29/11/2022	0%	
Test the hardware system	T	8 days	29/11/2022	08/12/2022	0%	
Software implementation	T	20 days	29/11/2022	26/12/2022	0%	
Publish the application on app store or play store	T	5 days	26/12/2022	01/01/2023	0%	
Connect the mobile application to the hardware system	T	11 days	02/01/2023	16/01/2023	0%	
Test the entire system	T	10 days	16/01/2023	27/01/2023	0%	
Project presentation #2	T	5 days	26/01/2023	01/02/2023	0%	
Completed project	M	0 days	02/02/2023	02/02/2023	0%	

Figure 6 Initial Phase Progress

2.4.2.3 Schedule Allocation

As part of the senior semester, the team will design the solution architecture theoretically.

(December 2022), and the implementation will take place next semester. Below (Figure 7) is the timeline for the project:

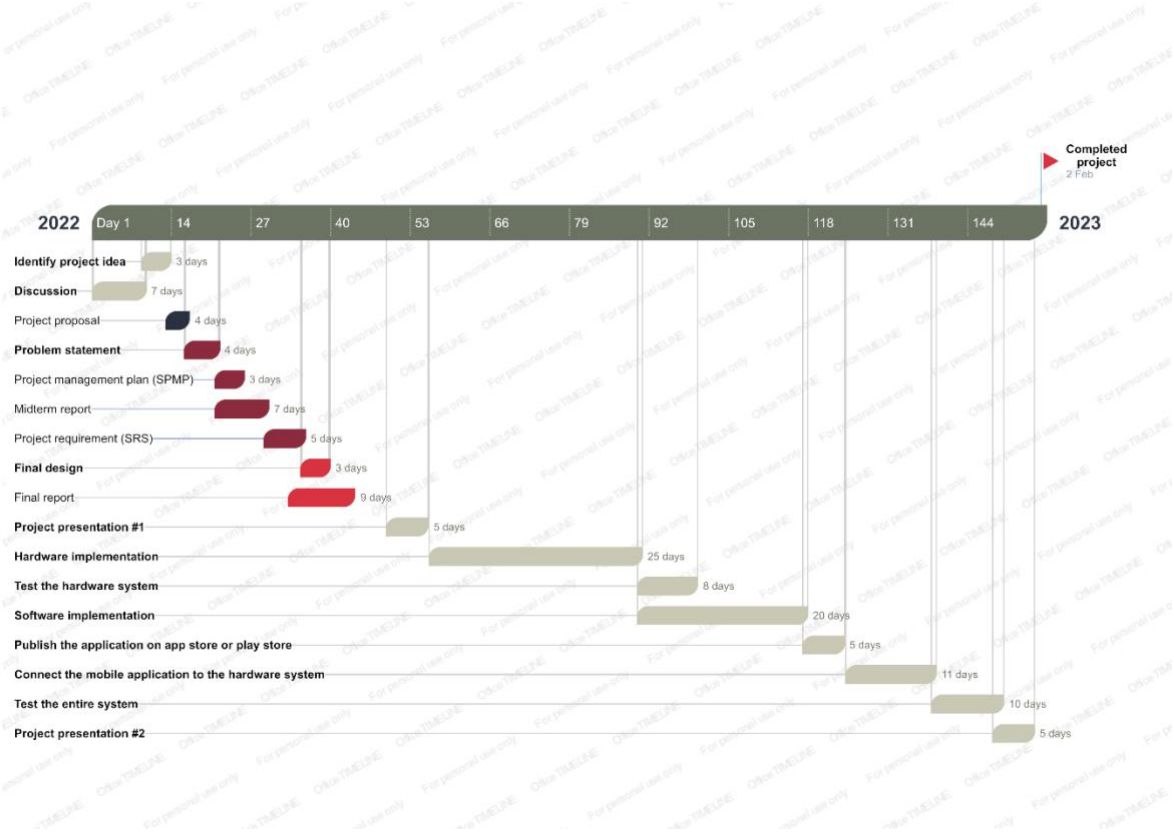


Figure 7 Project Timeline

2.4.2.4 Resource Allocation

Each team member will contribute equally to the project. Along with the other equipment, students will use their own devices for the project. Students will design and carry out physical tasks to finish the project.

2.4.2.5 Budget Allocation

A budget is necessary due to the resources that will be used in this IoT project, However, all the recourses that will be used in this project are available in Geeks valley therefore, our team members decided to purchase it online as shown in table No.9. The estimated budget of materials is:

Table 9 budget of materials

Tools	**Price**
Camera ESP32 CAM	75 SR
Camera holder	40.25 SR
USB: A-B	10 SR
Cars	36 SR
Subtotal	161.25 SR

2.5 Project Controls

To accomplish the desired results in this project, team members must follow a particular management and control criteria, which are listed below:

2.5.1 Requirements Control

This section describes the project control approach and techniques. Our project seeks to meet all the specifications of a smart parking system. If any changes to the outlined plans arise, our team will assess, discuss, resolve, and investigate the adjustments.

2.5.2 Schedule Control

Schedule control is carried out during the development of the project phases via a status report that is delivered to the project supervisor every two weeks to keep the project's plan flow up to date. The status report evaluates the project's progress based on completed and remaining tasks, team difficulties, and project meetings.

The management technique for project schedule control is defined in the following paragraphs.

2.5.2.1 Schedule Tracking

To ensure that the project has produced the desired results and performed the tasks on schedule, it is essential to regularly monitor its development. We have employed management tools like the Gantt Chart, which was already covered in section 5.2.2. This will assist us in maintaining control over the project's current status and any outstanding tasks. The Gantt Chart will give date-specific information about the project and tasks and manage the schedule.

2.5.2.2 Schedule Performance Reports

After each chapter, the team provides performance reports that show which tasks were completed within the projected time frame and which tasks are still outstanding. We as team members will debate these reports in the team leader's regular meetings, which will ideally improve the Schedule performance.

2.5.2.3 Schedule Reviews

The schedule review allows us to monitor the progress of the project by using a timeline with all assigned tasks among team members. The timeline represents the assigned tasks and each task deadline, and by monitoring the progress of the team members, we can determine whether or not they have completed their assigned tasks on time. The schedule review assists in addressing any anticipated hurdles due to any

incomplete tasks or any abrupt change in plan, as well as comprehending all future tasks and their deadlines.

2.5.2.4 Progress Variance Monitoring

We can distinguish between the real project deadline and the estimated deadline set by the team leader by using the progress variance monitoring. These details will assist the team members to identify any possible problems and give them the chance to revise the entire project and, if necessary, take remedial action.

2.5.2.5 Progress Variance Resolution

The progress variance plan can be implemented in different phases of the project utilizing estimated timelines. This will allow us to complete the chores quickly and easily, allowing us to go on to the next phase. However, if the projected time is insufficient to complete a work, we have a meeting to reschedule the time of the remaining task to fit in with the given timeframe set by the team leader to be completed accordingly.

2.5.2.6 Follow-up on Corrective Action

Several steps will be taken if team members are unable to fulfill the established deadline for several reasons.

For quicker implementation and connection, break down the tasks and equipment.

- Lengthen meetings and workdays.
- Seek guidance from someone who has more experience.
- The budget might be raised to hasten the shipping procedure.

2.5.3 Budget Control

The budget control plan will be carried out as follows in this part. If the budget exceeds the estimated budget specified in section 5.2.4, we shall hold a meeting to revise the current budget that was spent with the estimated budget. If any additional materials are required to complete the desired results, which may increase the budget, we will discuss and decide how crucial the material is to the project. Our team will make certain that there is an adequate budget for that material.

The following paragraphs outline the project's management method for budget control. The following paragraphs explain the project's management approach for schedule control.

2.5.4 Quality Control

The project may require adjustments or updates as it is implemented and developed in order to be completed more effectively and qualitatively; in this case, we will ensure the quality of the project outputs by training ourselves as a team to match our outputs with our project goals by doing some appropriate tests for the quality assurance of the outputs to give the exact outputs and outcomes of the project and removing any unpleasant consequences.

2.5.5 Project Reporting and Communication

Our project reporting and communication strategy will be carried out twice a week through actual face-to-face meetings and zoom virtual meetings held three times a week by the project leader. We will also interact using WhatsApp group chat, which allows all team members to send and receive documents, files, and progress updates, as well as report any problems that may arise. In addition, we are working on a shared word document in one drive so that we can work on it simultaneously for speedier outcomes. The communication between the team members and the supervisor will be conducted by the supervisor through zoom meetings once a week and through emails.

The next paragraphs establish the management plan for ensuring the broadest possible communication of project-related information.

2.5.5.1 Electronic Media

We communicate about projects using Microsoft Word shared files and WhatsApp group chat. It includes team members for them to be aware of the project's development. If the supervisor would communicate with the team members; he can do it via email rather than waiting for the next meeting to discuss the matter or putting notifications on Blackboard. This helps to shorten the project's timeline.

2.5.5.2 Meetings

Our project's meetings will be held twice a week between team members via actual face-to-face meetings and zoom sessions held twice a week by the project leader. The meeting between the team members and the supervisor will be held once a week by the supervisor via zoom meetings. The supervisor will notify the team members ahead of time to avoid any conflict. The team leader will present the current status of the project and any changes to the plans at this meeting.

5.5.5.3 Information Repository

It is advised to maintain the project repository's documentation throughout its many stages. These documents include the SPMP, Midterm Report, SRS, SDS, Final Design, and Final Report, which will be shared via Microsoft Word shared files. If any team member is unclear about any component of the project, they can all access this information repository (shared files). Additionally, it summarizes the project's status and explains each section in detail. This speeds up and improves the communication process. As a result, everyone on the team will have a much greater understanding of the project and the topics covered in meetings.

2.5.5.4 Reviews

Reviews improve in the execution of the monitoring procedure. This will assist team members in identifying current and potential project challenges. After the completion of each chapter of the project, a review meeting is organized to revise the entire work and determine whether we may end the chapter of the current phase and go on to the next.

2.5.5.5 Status Reporting

Every two weeks, the group leader gathers all team members for a meeting to discuss project progress and to follow up on task completion by documenting a status report. This will then be emailed to our supervisor for review.

2.6 Risk Management

In the risk management plan the project manager will identify, evaluate, and maintain any potential risks, and this plan will be followed, evaluated, and updated during the project's duration. As shown in table No.10 below:

Table 10 risk management plan

Risk	**Description**	**Probability (High – Low)**	**Effect (High – Low)**	**Procedure**
Miscommunication And Misinterpretation	Failing in communicating and delivering ideas and	High	High	Immediately convene a meeting of the members.

	opinions successfully			Involve the leader in resolving the problem.
Equipment malfunction	The equipment is corrupted or destroyed	Low	High	Make sure to have a backup for the broken tool
Loss of Data	Corruption or loss of data	Low	High	Make Data in the shared file on the cloud, and make a copy of them in different devices
connection issues	Unable to connect the hardware and software together	High	High	Ask an experienced/skilled expert for guidance or look for a solution on the internet.
Failure to fulfill deadlines	Inability to deliver the project within the specific deadline.	Low	High	Commitment to work within the project timeline and not to accumulate tasks
recognition problem	not understanding the operating system of the additional device, resulting in malfunction	High	High	Checking the operating systems used and their ability to identify other devices

2.7 TECHNICAL PROCESS

2.7.1 Process Model

Multiple things will be discussed in this section such as (process model, tools, methods, techniques, and lastly infrastructure and product acceptance). According to our project, the smart parking system based on IoT will be the most appropriate process to be implemented for several reasons. One of the key reasons is that the team won't be able to go on to the following phase of the project until the previous one has been examined, accomplished, and finished. The group members selected the necessary tools and requirements using this

model, scheduled dates and deadlines based on our supervisor's (Mr. Sardar Zafar Iqbal) instructions and established the expected results from the final implementation. The team will achieve a successful, contented outcome by successfully completing these steps.

The System Development Life Cycle (SDLC) of the smart parking model is shown in the figure below (Figure 8):

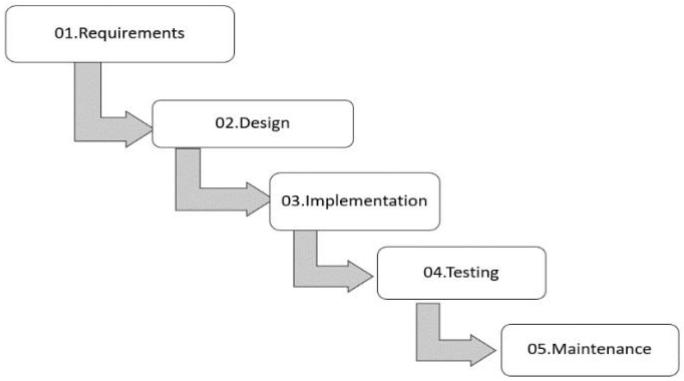


Figure 8 The System Development Life Cycle

01 Reequipments Definition: To create a smart parking model, all essential equipment must first be purchased. All project needs, including system inputs and outputs, are recorded, and documented in a requirement specification document at this phase (SRS document).

02 Design: Based on the requirements specifications document (SRS) that was developed in the first phase, the system design is created in this phase. Consequently, this system design may be used to identify the hardware specifications and system architecture.

03 Implementation: Based on feedback from the system design, a system is first developed into a tiny unit that is then integrated in the following phase.

04 Testing: All units are tested and integrated at this phase.

05 If the team members found any errors or defects, changes will be made in this phase to address the problem. Once all the stages have been successfully completed, the system will achieve our intended objective, which is to create a smart parking lot using IoT.

2.7.2 Methods, Tools, and Techniques

In the method, tools, and Techniques we will describe the phases of our project, the methods, tools, and techniques we will use is shown in table No.11 below.

Table 11 Methods, Tools, and Techniques

Phase	Methods	Tools	Techniques
Planning and Gathering requirements	• Using word and use case diagram	• Microsoft word • Google docs • Zoom	None
Design	• Interface mockup	• Microsoft word • Zoom • Microsoft PowerPoint • Draw.io • Figma	• Gantt charts and scheduling Techniques • Understand Arduino IDE programming language.
Implementation	• Program the microcontroller chip. • Program cameras. • Connect Arduino IDE with the hardware(cameras)	• Microsoft word • Zoom • Arduino IDE • OpenCV	• How to connect the Arduino IDE with hardware(cameras)

2.7.3 Project Infrastructure

The creation and maintenance of the development environment will have a significant impact on the productivity of the team, which will ultimately have an impact on the final deliverables. For this reason, all team members should have access to the necessary hardware, software, techniques, and network connection.

- **Hardware**
  - The team members will need to buy the needed equipment such as ESP32-CAM, Camera holder, Micro SD, -.
  - All team members should have a PC.

- **Software**
  - The needed software is Microsoft office, Visual code studio, Flutter Framework, Notepad++.
- **Operating system**
  - The Arduino Software (IDE) is a dependable and cost-effective Integrated Development Environment that offers a wide range of available sketches and shields, with the overall cost dependent on the required shields. The best part is that there is no need for an external programmer or power supply. Additionally, the Arduino-Python3 Command API is a lightweight Python library that facilitates communication with ESP32-CAM microcontroller boards from a connected computer through standard serial IO, be it over a physical wire or wirelessly. Therefore, to use it, team members should have Windows, Mac OSX, or Linux operating systems installed.

2.7.4 Product Acceptance

- Members of the group concurred that if the final deliverable achieves the stated aim and objectives determines whether a project is successful or unsuccessful.
- Each milestone/prototype of the project will be accepted by the SEC and the project manager by signing acceptance documentation.
- The project idea, plan, goals, timetable, and budget were all approved by the group members and the project supervisor, as shown in table No.12 below.

Table 12 Product Acceptance

Name	Signature	Date
Mr. Sardar Zafar Iqbal		11 September 2022
Aisha Moraie Khubrani	Aisha Moraie Khubrani	11 September 2022
Abeer Saad Aldkheel	Abeer Saad Aldkheel	11 September 2022
Bashayer Mohammed Alyami	Bashayer Mohammed Alyami	11 September 2022
Ghadah Riyadh AlMuhaideb	Ghadah Riyadh AlMuhaideb	11 September 2022
Ghadeer Saeed Alghamdi	Ghadeer Saeed Alghamdi	11 September 2022

2.8 SUPPORTING PROCESSES

2.8.1 Independent Verification and Validation

2.8.1.1 Validation Scope

This part gives a full analysis of the outputs and the outputs' completeness in a satisfactory and efficient manner. The process of verification and validation occurs regularly throughout the duration of the project. This will be achieved by taking the following steps:

- Requiring the supplies
- Designing the circuit
- Determining the circuit's principles of operation.
- Programming a ESP32 CAM
- Development of the Application
- Implementation and Testing

2.8.1.2 Tools and Techniques

The project will develop a mobile application using computer vision prediction model and the required software and hardware.

The following tools was utilized throughout the project:

Microsoft Word and MS Project for project development.

- Online meetings are held via WhatsApp and Zoom.
- For shared files, OneDrive is used.
- Outlook is used to send emails to the supervisor.

2.8.1.3 Responsibilities

All team members are responsible for ensuring the quality of the services provided throughout the project.

2.9 Documentation

In table No.13 below, the contents of each phase of the project are documented, from selecting the project to completing its implementation.

Table 13 Documentation Plan

Document Type	Format Standard	Estimated Page Count	Peer Review Type
Project Proposal	Provided by supervisor	1	Project supervisor/Co-supervisor:
Project Management Plan (SPMP)	IEEE Std 1058 – 1998	35	Mr. Sardar Iqbal.
Project Requirements (SRS)	IEEE Std 830 – 1998	25	Ms. Maryam Alnasser.
Project Design (SDS)	IEEE Std 1016 – 1998	36	
Software Test Plan (STP)	IEEE std 829-1998	20	
Software Test Report (STR)	IEEE 829-1998	10	
Conclusion	Provided by supervisor	2	
User Manual	Provided by supervisor	28	
Business Model	Provided by supervisor	1	
Final Report	Provided by supervisor	120	

2.10 Reviews and Audits

Team members will examine the entire project structure on a weekly basis. The auditing step will incorporate all results, whether successful or defective. Every week, the supervisor and team members discuss the model's weaknesses and strengths and come up with solutions and strategies to keep the model in good condition.

Chapter 3 Software Requirements Specification

3.1 Introduction

This section provides documentation of the software requirement specification's (SRS) objective and specifies the necessary requirements to construct Parkin, as well as a comprehensive description of the system's functions and tasks.

3.2 Purpose

The purpose of SRS documentation is to offer a comprehensive and clear description of Parkin functional and non-functional requirements, as well as the anticipated final system. This document will demonstrate the scope of the project, the system's constraints, and limitations, as well as the needs and characteristics of the system software and hardware in detail, to ensure that all team members have a comprehensive understanding of the system's operation.

3.3 Document Conventions

This section defines some unfamiliar terms that can be found in the glossary in Appendix A at the conclusion of the document.

3.4 Intended Audience and Reading Suggestions

- **The project manager (supervisor):** is responsible for reviewing and providing team members with instructions and feedback on how to optimize their system.
- **Project Development Team:** The team members will benefit from this document by gaining a comprehensive understanding of the project's needs and functionality, which will aid in its effective implementation.
- **Users:** The users are drivers who are interested in parking as well as the project evaluators of Imam Abdulrahman bin Faisal university who will read and review this document to ensure that the project requirements are met and to determine if any modifications and enhancements are necessary to improve the project prior to its implementation.

3.5 Product Scope

This part is already included in this document's second chapter.

3.6 References

All sources cited in this chapter are given at the conclusion of this document.

3.7 Overall Description

3.7.1 Product Perspective

The original design concept for the traditional parking system served as the basis for our approach. The project will assist in transforming the current parking system into a smart parking system by installing smart cameras that can identify parking spaces that are available, preventing time and resource waste. Additionally, we have enhanced the initial parking system by adding a feature that allows users to get information about available free parking. The key elements of our system are represented in the figure No. 9 below:

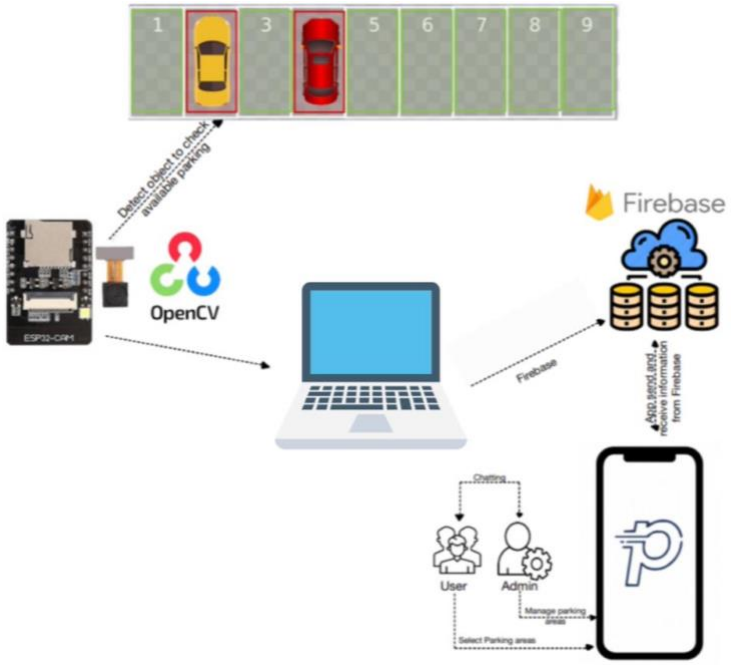


Figure 9 Components of our project

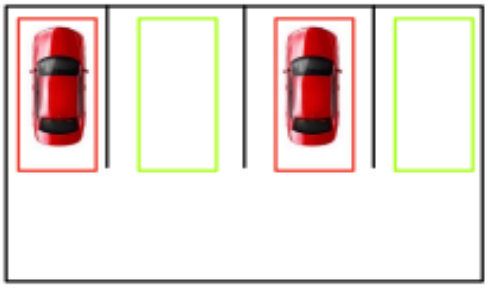
In the figure No.9 above we identified the major components of our system as follows:

- **ESP32 Camera model:** the camera is responsible for reading the status of the parking space such as full or empty.
- **Laptop:** will be connected to ESP32-CAM via USB and share data to firebase
- **Mobile application:** is the Application that will be used in multiplatform it will show a dashboard of the parking space, allow the user to reservation.
- **Arduino IDE:** is the software that will be installed in the laptop and will receive the readings from the camera and will enable us to program ESP32-CAM to decide how the project will run.

- **Firebase:** A database housed in the cloud is the Firebase Realtime Database. Data is synchronized in real-time to mobile applications as JSON storage.

The process for our project is shown in Table No. 14 below, where we initially show a parking space with one car. Once the ESP32-CAM recognizes the car, we label the available parking as green and the unavailable parking as red.

Table 14 process of the project

<table>
<tr><th colspan="2">Data collections from parking space</th></tr>
<tr><td colspan="2">  </td></tr>
<tr><td colspan="2">Training</td></tr>
<tr><td rowspan="2">Car detection</td><td>Car detection</td></tr>
<tr><td>Car recognition</td></tr>
<tr><td colspan="2">Testing</td></tr>
<tr><td>  </td><td>  </td></tr>
</table>

3.7.2 Product Functions

- Enables the management and reduction of parking search traffic on the streets, ensuring the security of the car and the driver inside the parking space.
- Reduces fuel consumption while ensuring the driver's and vehicle's safety in the parking space.
- Enable user interaction with the mobile application so they may interpret the data displayed in the user interface and complete the reservation process.

3.7.3 User Classes and Characteristics

In this section, we will describe the characteristics of ParkIn targeted user and then describe the characteristics of the user. There is only one type of user in our system: any person who are interested in found parking space. The user should have a smartphone device and know how to use the app. The table No.15 below contains a summarization of the user classes and characteristics.

Table 15 User Classes and Characteristics

User type	Requirements	Skills	Education level	Technical expertise
Any person who are interested in found parking space	Smartphone - Internet connection	Understanding English language	Not required	User has the basic knowledge of how to use the application. - Basic knowledge of driving.

3.7.4 Operating Environment

our project will be operated in an internet-based environment and can be run on any smartphone operating system. The requirements are having an internet connection.

3.7.5 Design and Implementation Constraints

- **Programming language:** our application will be programmed by python, in addition the Hardware will be implemented Arduino IDE, and Notepad++ is the only option that could be used.
- **Time restriction:** the project must be implemented in 15 weeks. As a result, the schedule is fixed and cannot be changed.
- **Parking space area**: Due to our lack of understanding of the parking architectural map, we are unsure about the parking area, and the camera is unable to see the parking due to the sheds that cover the parking area.

3.7.6 User Documentation

In this section, we will list all the deliverables of our project. The team will produce a final working system and will provide a single copy of each deliverable described in table No.16 below.

Table 16 User Documentation

Deliverables	To Whom	Document Format	Date
Defining Project	Mr. Sardar Iqbal	Softcopy	**6 September, 2022**
Problem statement	Mr. Sardar Iqbal	Softcopy	**22 September, 2022**
software Project Management Plan (SPMP)	Mr. Sardar Iqbal	Softcopy	**25 September, 2022**
Midterm report	Mr. Sardar Iqbal	Softcopy	**22 September, 2022**
Software Requirement Specification (SRS)	Mr. Sardar Iqbal	Softcopy	**16 October, 2022**
Software Design Specification (SDS)	Mr. Sardar Iqbal	Softcopy	**23 October, 2022**
Implementation and Testing	Mr. Sardar Iqbal	Softcopy	**23 February, 2022**
Conclusion	Mr. Sardar Iqbal	Softcopy	**30 February, 2022**

3.7.7 Assumptions and Dependencies

The assumptions are:

1. Assuming users have basic knowledge of smartphones.
2. Assuming that the system will be flexible and easy to use.
3. Assuming that the user has basic knowledge of driving.
4. Assuming that any time the user can access the application.
5. Assuming that the users have an internet connection.
6. Assuming that users must know their Email and password to access their account on mobile application.
7. Assuming the user can manage the hardware equipment cautiously.

The dependencies are:

1. It requires specific hardware and software for the system to operate.
2. The system should store information reports of the dashboard and parking on database.

3.8. External Interface Requirements

3.8.1 User Interfaces

The user will connect with the system via a mobile application that uses several interfaces to deliver the fundamental functionalities required to run the project. More information on user interfaces will be provided in Chapter 4.

3.8.2 Hardware Interfaces

In our project, simulations will be employed, but if there is a need for hardware, the ESP32-CAM will be used. It will be linked to the laptop and will recognize and analyze the elements that will be detected in the situation.

3.8.3 Software Interfaces

We will use a Dot language or Python, which are valid on all mobile operating systems. Additionally, they have some pre-instilled library that will help us to detect the object which will be useful and will be linked to all the necessary elements such as databases so that the program runs smoothly.

3.8.4 Communications Interfaces

In order for the program to function effectively, all hardware and software components must be connected to a secure and established internet connection to allow interaction within the system. Lan-Wan, wireless modem, and Ethernet wires may be used.

3.9 SYSTEM FEATURES

3.9.1 FUNCTIONAL REQUIREMENTS

In this section a detailed description will be given about the user who will use the system and features that are provided, where the main function is going to be performed by the project developers, building manager, faculty, users, and technical staff.

Table 17 FUNCTIONAL REQUIREMENTS 01

Requirement#: FR-01	**Requirement type: functional requirement**	**Title:** Develop the Mobile Interfaces
Actor: Project Developers		
Description: develop an application interface to display the parking spots and the reserving interfaces		
Normal Course of Events: • The project developers will create the application.		

• Project settings will be modified.
Exception(s): N/A

Table 18 FUNCTIONAL REQUIREMENTS 02

Requirement#: FR-02	**Requirement type: functional requirement**	**Title:** Create new personal account.
Actor: Project Developers / User.		
Description: The user will create an account which will provide him/her with a welcoming and explanation of reserving a parking spot		
Normal Course of Events: • sign-up page appears. • user will fill the required information. • verification link will be sent to the email. • the application will check if the password meets the requirement (longer than 8 symbols – uncomment words are used – non-standard uppercasing…etc.).		
Exception(s): • User cancels account creation. • If the password did not meet the required conditions.		

Table 19 FUNCTIONAL REQUIREMENTS 03

Requirement#: FR-03	**Requirement type: functional requirement**	**Title:** Access to an existed account
Actor: Project Developers / User		
Description: The Project Developers / user will log in to the account which will provide		

him/her with interfaces that contain all the reserved parking spots and shows the status of the spot
Normal Course of Events: • Sign-in page appears. • Project Developers / user will enter the email. • Project Developers / user will enter the password. • users click on "next button" to proceed to the next page. • The application will check if the data entered it corrects. • If the data is correct. • the Project Developers / user will successfully login and the created interface will be displayed.
Exception(s): • the Project Developers / user enters invalid/incorrect email. • the Project Developers / user enters invalid/incorrect password. • the Project Developers / user enters forget the password.

Table 20 FUNCTIONAL REQUIREMENTS 04

Requirement#: FR-04	**Requirement type: functional requirement**	**Title:** reserve parking spaces
Actor: User		
Description: the user can view the available parking spots and reserve		
Normal Course of Events: • Home page will appear • the reserving icon will be on the services page • the user can check the available parking spots and chooses one of them • confirm the reservation		

Exception(s):
• Displaying incorrect, corrupted data.

Table 21 FUNCTIONAL REQUIREMENTS 05

Requirement#: FR-05	**Requirement type: functional requirement**	**Title:** admin editing
Actor: admin		
Description: the amin can manage the users (add- update-delete)		
Normal Course of Events: • The admin will login and go to the admin interfaces. • The admin will be able to add the user. • Update the users. • Delete the users		
Exception(s): • N/A		

3.10 USE CASE DIAGRAM

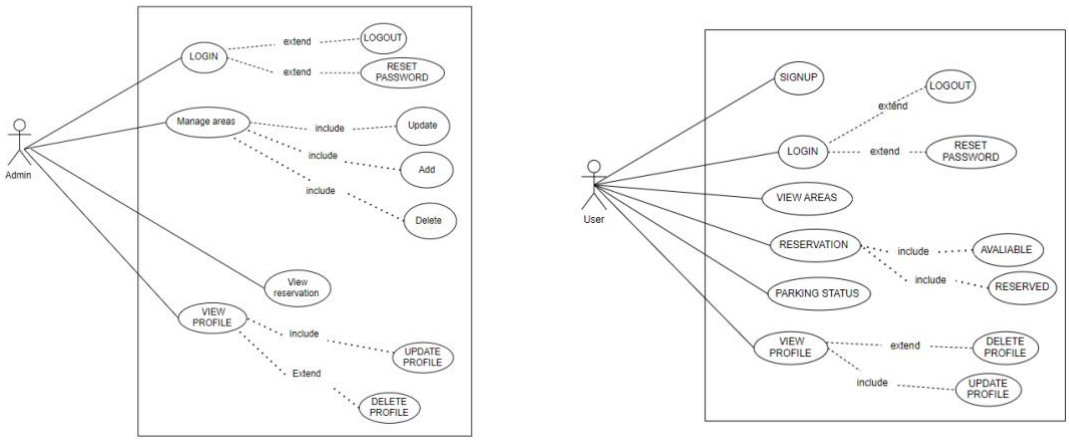


Figure 10 use case

3.11 Other Nonfunctional Requirements

A common way to distinguish non-functional requirements from functional requirements is to describe how a system shall accomplish a task as opposed to what the system should

accomplish. Therefore, this section will illustrate how the system performs, the design constraints, and the design attributes, as well as the non-functional requirements.

3.11.1 Performance Requirements

Since Parkin is an intelligent parking system that depends on vehicles and parking lots, it is required to have good quality camera that is able to precisely detect the available and busy parking lots, vehicles and so on. In addition, fast and accurate performance and having broadband internet access is crucial for the system.

- **Availability**

  Once the application has been installed, it should be available for the user 24/7, it needs to be connected to the internet. The availability will be calculated as $Availability = \frac{Uptime}{(Uptime/Downtime)} * 100$ and the weekly allowable downtime shouldn't be more than 3% of the total time.

- **Responsiveness**

  It's critical that the processing time for an operation does not exceed 10 seconds, as Parkin is a mobile app.

- **Fault Recovery**

  The system must be fixed in order to recover from the failure, using the Mean Down Time: $\frac{\sum Start\ of\ Uptime - Start\ of\ Downtime}{\text{Number of Failures}}$ and the maximum allowable downtime for the system is 5 hours, and it’s the maximum acceptable breakdown time.

3.11.2 Safety Requirements

Parkin mobile application is a self-reliant /application, it should not affect other systems on a personal computer. On the other hand, it should not harm the device. Also, other systems should not disturb Parkin mobile application. The concern is about the possibility that the database might crash because of operating system filature, viruses, or damage on the server. And to prevent that from happening, database backup is required and keeping

the server link updated to prevent any virus or hacking attacks in that way the data are kept safe from any kind of loss.

3.11.3 Security Requirements

- Email and password are required as identity authentication.
- Each user has their own account in the Parkin mobile application.
- Users can't edit or modify the interface where the only can view it.
- Users don’t have direct access to the camera. However, the parking lots’ availability will be always updated in the application according to the camera.

3.11.4 Software Quality Attributes

Table 22 Software Quality Attributes

SQ Attribute	The Concept of the Attribute	Description
Usability	An evaluation of how effectively, efficiently, and satisfactorily a product or design can enable a user in a given context to achieve a defined goal. Additionally, it ought to offer user guidance for invalid entry. User movement between interfaces should be seamless, and the error rate shouldn't be more than 10%, while the success rate should be about 90%.	The interfaces of Parkin application are clear and easy to use and understand for the users.
Reliability	Product, system, or service reliability refers to their capacity to function as intended over a period, or to operate in defined environments without failure. Utilizing the Mean Time Between Failures, reliability will be evaluated by the following formula: $MTBF = \frac{TWT - TBT}{Number\ of\ Breakdowns}$	The intended function of the system is to detect the available and busy parking lots by the help of ESP32-CAM located near those parking areas. To inform the users, that information will be displayed on the dashboard of the application, and these functions are accomplished by project

		developers with high degree of accuracy.
Richness	Describes the collection of features that the interface provides and includes.	The system that is provided to the user is full of features.
Robustness	A system's ability to cope with errors while it is being executed.	The system is robust in a way that can handle with inappropriate input, for example: wrong E-mail or password.
Correctness	The degree to which a software meets its requirements and fulfils the mission objectives of the user.	The system fulfils the mission objectives.
Flexibility	The effort needed to make a modification in the program.	The system has been developed to be flexible which will make developers mission easier.
Adaptability	The system can tolerate changes in its environment without external intervention. This software is adaptable IAU.	The system can be installed in any mobile platform.
Reusability	The capability of using the same code in another application (reuse it) to perform its function.	The system has been analyzed to be usable for more than one function.
Integrity	The integrity of a system refers to the capability of performing correctly according to the original specification of the system under various adversarial conditions.	Secure information will not be lost, and the system will be protected.
Testability	The effort required to test and ensure that system performs as intended.	The team members will test the system functionality and tasks to

		ensure we receive the expected output.

3.12 Business Rules

The main goal of business rules is meeting and achieving the systems insights. Business rules consist of various regulations, rules operations and constrains which the system users stand for. For example, users can't deal with the camera, but can see the availability of the parking lots in the dashboard of the application, and reserve one available lot only.

Chapter 4 System Design Specification

4.1 Introduction

The Software Design Specification (SDS) document's goal is to describe the architecture and user interfaces of the SPD system. As a result, this document will go into extensive detail about the interface, database, components, and constraints.

4.2 Purpose

The SDS document’s intention is to clarify how the system will be constructed to meet the system requirements.

4.3 Scope

The diagrams used in this documentation, the thorough design, and the system architecture are all included in this section. They are as follows:

- Use case diagram.
- Activity diagram

4.4 Definitions, Acronyms and Abbreviations

The glossary in Appendix A at the end of document will contain the definition of terms mentioned in this report.

4.5 Reference

All material that has been referenced in this chapter will be listed at the end of document.

4.6 System Overview

4.6 SYSTEM OVERVIEW

The project aims to create an intelligent parking system that will allow drivers of moving cars to locate occupied parking spaces and recognize license plates using an ESP32-CAM that will be placed close to such spaces.

The user can use a mobile application using a smartphone device Visual code studio will be used to develop the application.

4.6.1 GENERAL FUNCTIONALITY

The project functions will be controlled using ESP32 CAM. ESP32 will operate according to the given functions and requirements.

4.6.2 ESP32-CAM Camera

Is a small size, low power consumption camera module based on ESP32, utilized to record high quality video and capture images at parking spaces to display to users

4.6.3 Database Functionality

In this project, the database will be implemented using Firebase, which provides many useful functionalities. Moreover, database will be updated every 60 seconds. the project has been provided with the following useful features:

- Real time data retrieval / storage.
- Cloud storage.

4.7 DATABASE STORAGE

This system will use Firebase database to store user data and ESP32-CAM reading as well.

- Store user information
- Retrieve user information.
- Update user information
- Retrieve the readings from the cameras.
- Update the readings from the cameras.

4.8 Design Considerations

This section will consider the assumptions, dependencies, and general constraints of the design of this project.

4.8.1 ASSUMPTIONS AND DEPENDENCIES

This section will describe assumption and dependences of different part of the project.

Software and hardware:

- The cameras will detect the circle and if there are no cars covering this circle, which means that parking is available.
- The Arduino IDE will be used to program the ESP32-CAM with a c++ programming language.
- The Not[ad++ will be used to program the main program with python programming language.
- for the real time data retrieval and storage, the project will use database platform such as Firebase.
- To keep the track of the cameras reading, the project will work for 24 hours a day and 7 days a week.
- The mobile application can retrieve the data from the used database.

Operating system:

The mobile application will run on multiplatform.

End-user characteristics:

- Someone who can drive a car and wants to choose a parking spot in the easiest way.
- Users of all knowledge background can navigate the mobile application easily.
- The user must have an account to access the system.

4.8.2 GENERAL CONSTRAINTS

- **Hardware or software environment:**

The mobile application will run on multiplatform.

- **End-user environment:**

User must have internet connection to perform his/her activities and make use of the system.

- **Standards compliance:**

All the project documents follow IEEE standards.

- **Data repository and distribution requirements:**

The database used is firebase, therefore, the reading of the cameras will be registered in the database for a long time.

- **Security requirements (or other such regulations):**

The mobile application will be restricted by an authentication with an email and password and phone number.

- Memory and other capacity limitations:

The database of the user has a limited capacity.

- **Performance requirements:**

The system is expected to preform efficiently and the cameras reading will be sent to the database rapidly and in a timely manner, and without the user interaction.

- **Network communications:**

the system will not work efficiently without Wi-Fi connection.

- **Verification and validation requirements:**

Multiple tests will be performed to determine the correct parking space and how the cameras will detect the availability and unavailable parking.

4.9 User Interface Design

This section contains screenshots, prototypes, and a detailed discussion of the interfaces that will be offered to the user.

4.9.1 Overview of User Interface

Aside from developing an auto space detection Hardware Structure, Parkin will also include a software program with a variety of beneficial capabilities for the user to utilize as guidelines. When the user launches the application, the walk-through page appears, followed by the home page after clicking the end button, where the user can sign in or create an account if he or she does not have one figure 11 demonstrate the architecture of system hierarchy.

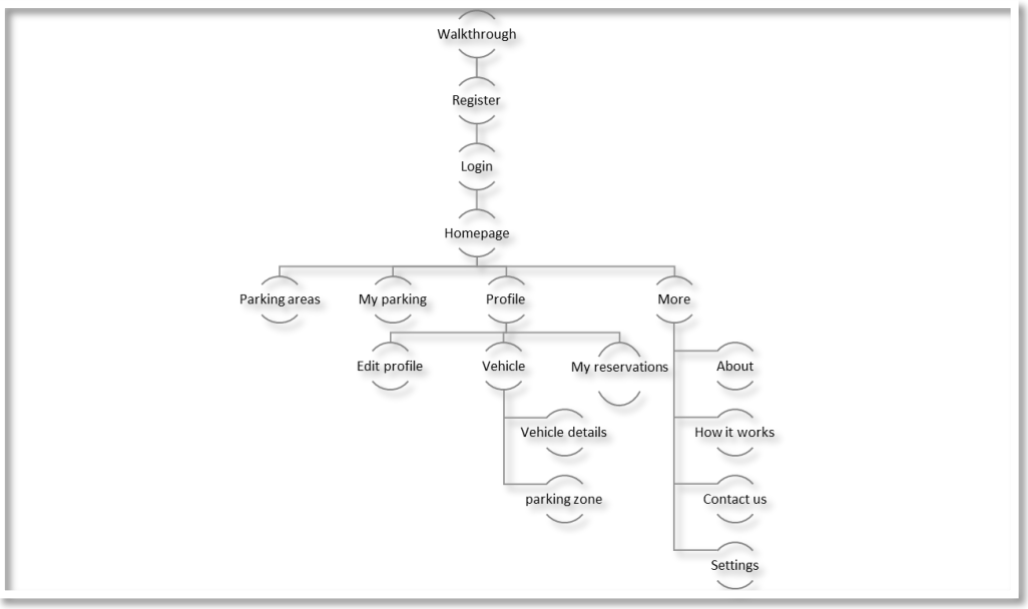


Figure 11 System Hiarchy

4.9.2 Interface Design Rules

- **Consistency:** Since the Parkin will provide a uniform design, the font size and type, colors, buttons, and other components will be comparable.
- **Dialogs yield to closure:** The sequence of actions will be ordered, and the user will receive informative feedback.
- **Universal usability:** The application can be utilized by people of all genders, ages, and levels of knowledge. The application interfaces will have clear features that will allow both novice and expert users to use it easily.

4.9.3 Screen images for User

4.9.3.1 Walkthrough Interface

Figure 12 shows interfaces consists of four slides it will be first displayed to the user when the user enters the Parkin app in the first time, the walkthrough gives an overview of Parkin.

12 Figure walkthrough interfaces

4.9.3.2 Welcome Interface

The figure 13 shows the application's Welcome interface, which has two buttons that determine the user's next location. If the user does not already have an account, they must click on the SIGN-UP button, which will send them to the next destination, the register page, where they may establish their own new account. If the user already has an account, he or she will click on the LOGIN button, which will send him or her to the login page, where they will be able to access their account.

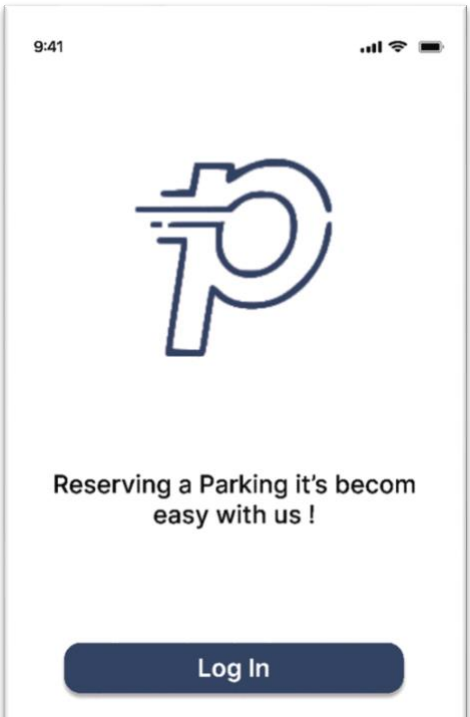


Figure 13 welcome interface

4.9.3.3 Register Interface

If the user does not already have an account, he or she can create one. The figures 14 and 15 showing the "signup page," where the user must enter information such as their full name, email address, and create a strong password that should contain capital, small letters, characters, and numbers, to protect their personal account from being accessed easily by unauthorized entity.

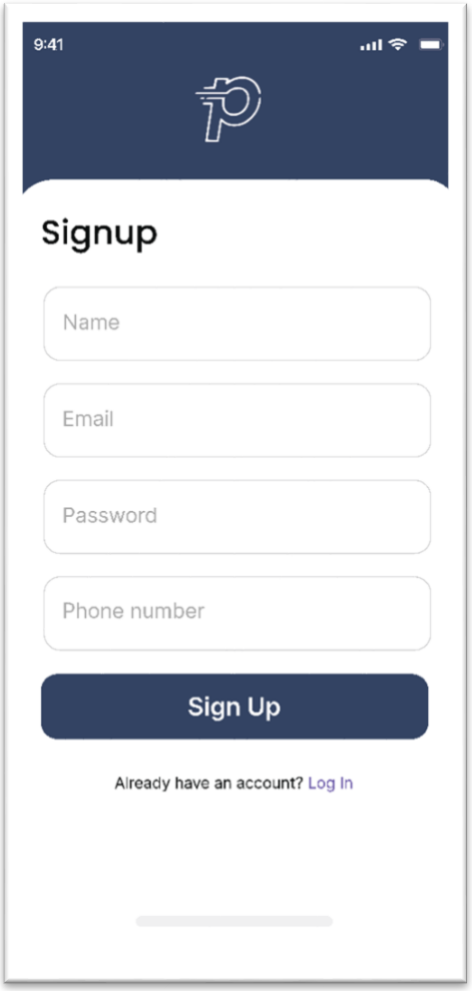


Figure 14 signup interface

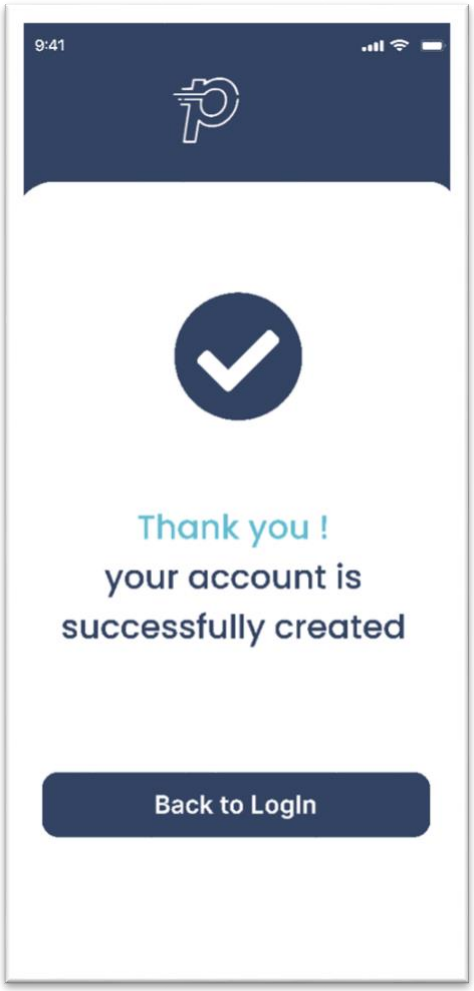


Figure 15 confirmation sign up interface.

4.9.3.4 Login Interface

The figure 18 shows the Login screen, where the user must input their email address and password. The program will compare the given data to the one recorded on the Database. If the users forget their password, they may click on the "Reset password" link, which will allow them to set a new password to avoid losing their account.

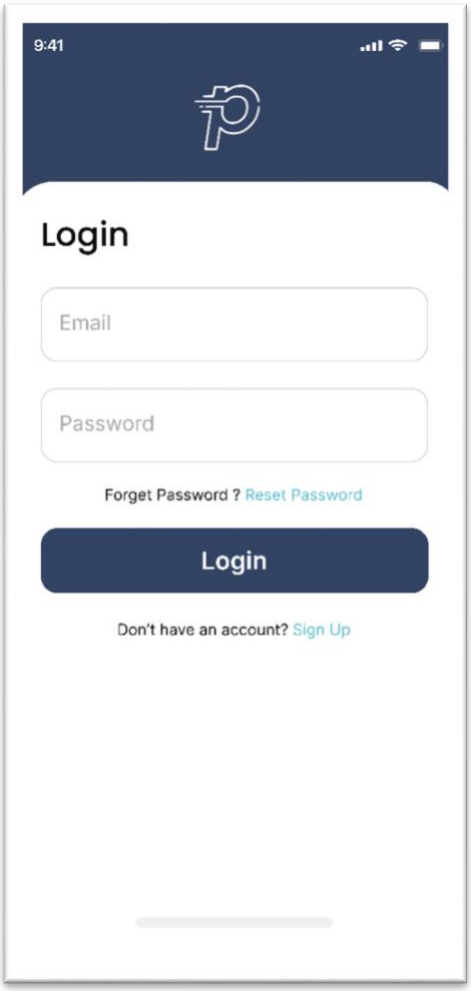


Figure 18 login interface

4.9.3.5 Recover Password Interface

If the users are unable to access their account or have forgotten their password, the program allows them to generate a new password to keep their account active. The figure shows 19 and 20 displays "recover account password" to assist the user in creating a new password where they must enter their email address, thus the application will check if the email address is registered in the database, if so, an email will be sent to the user to create a new password for their account.

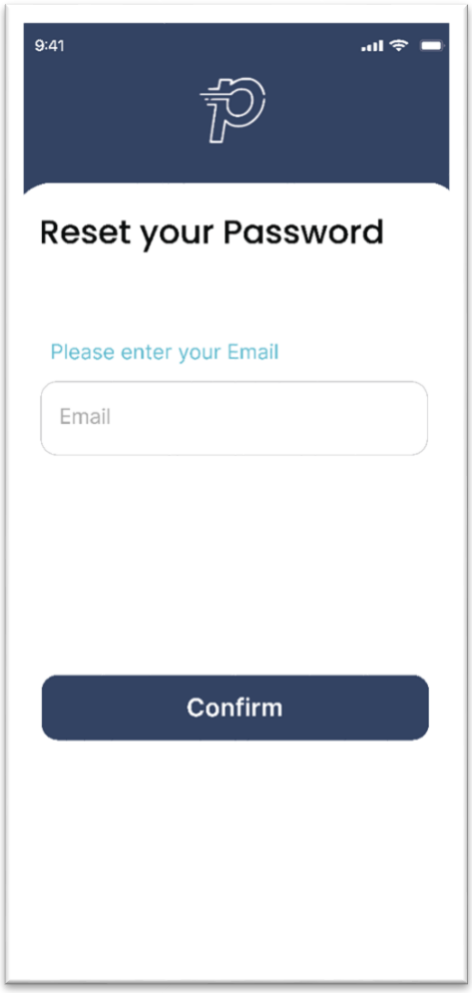


Figure 20 reset password interface

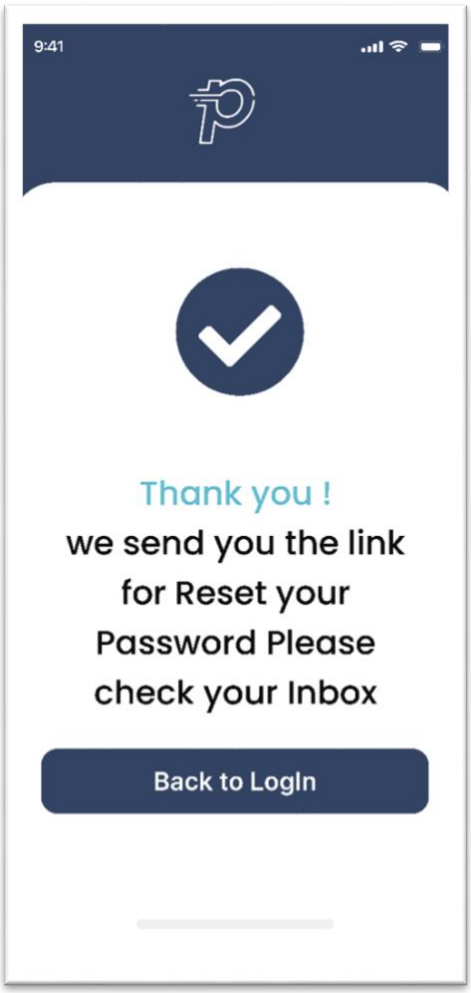
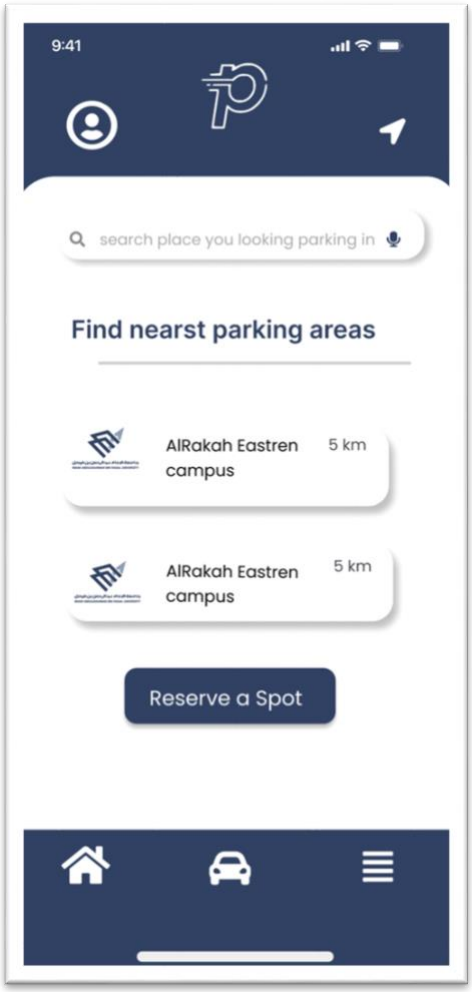


Figure 19 confirming set password interface

4.9.3.6 Home Interface

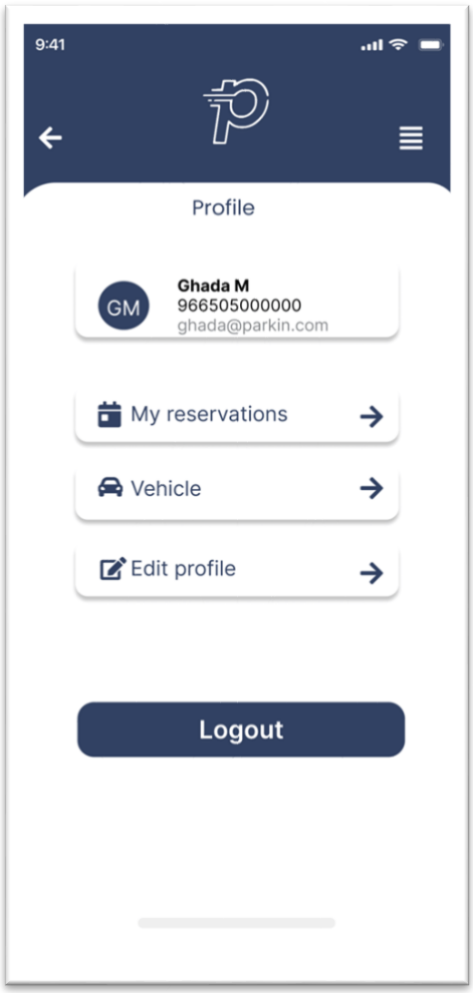
Figure 21 displays the page that will be shown to the user once he/she successfully sign in. In the top of the page, it contains profile icon that transfer the user to the profile page and precise location button, also, in the bottom the reserve a spot reservations button that allow user to book a parking to the user, and finally the navigation bar to easily move between pages.



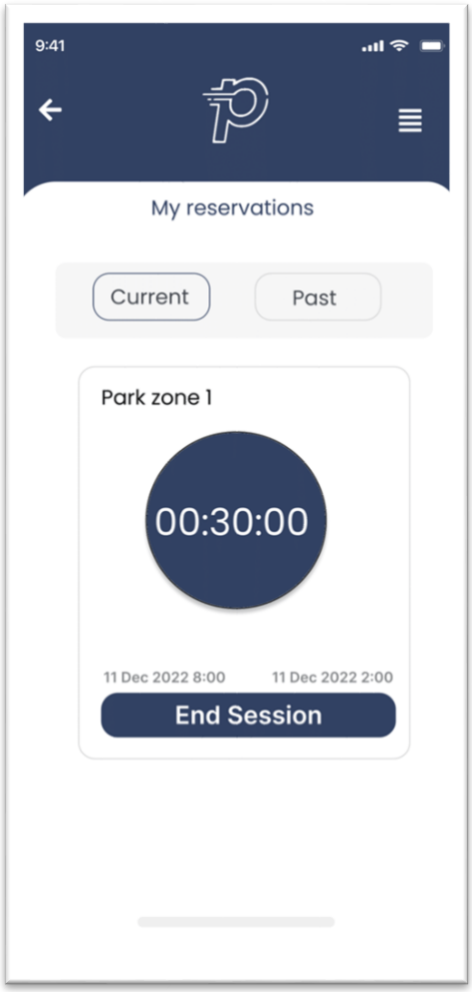
Home interface Figure 21

4.9.3.7 Profile Interface

This interface in figure 22 displays three buttons that contain a edit profile that move the user to edit profile page like what is shown in figure 25, and my reservations that is shows the vehicle current and previous reservations and vehicle where user can edit vehicle information (figure 24).



Profile interface Figure 23



Reservations interface Figure 22

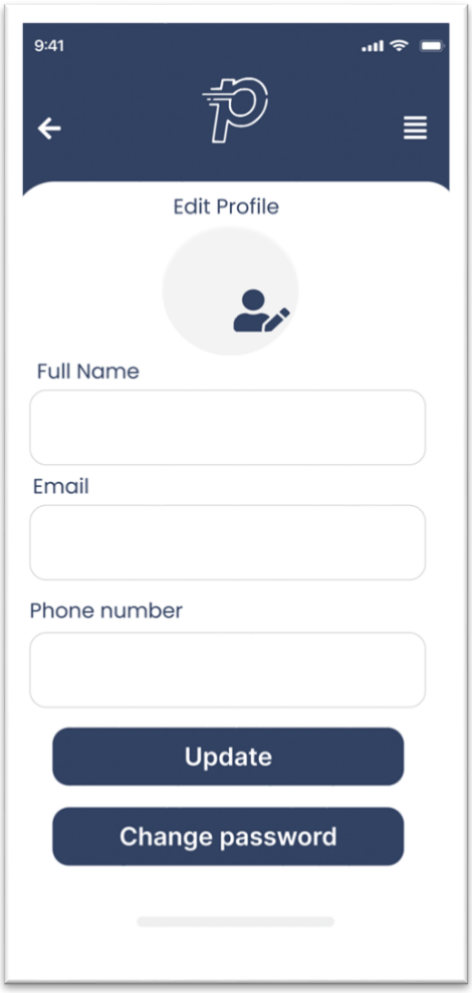


Figure 25 Edit profile

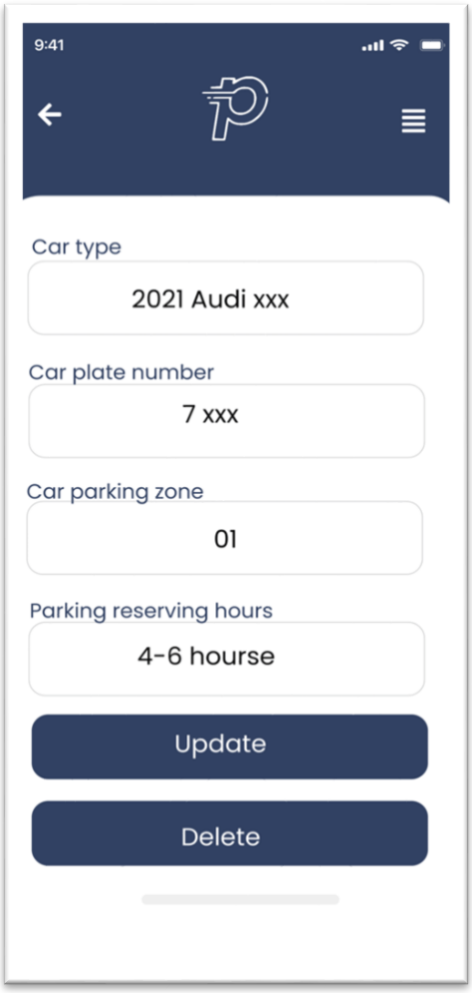


Figure 24 Vehicle interface

4.9.3.8 Edit Reservation Interface

User can end the session anytime he/she wants to terminate the reservation process, and as in figure 27 the user will ask if he sure wants to cancel the reservation after that a pop-up message will shows in figure 26 that the reservation has been cancel successfully.

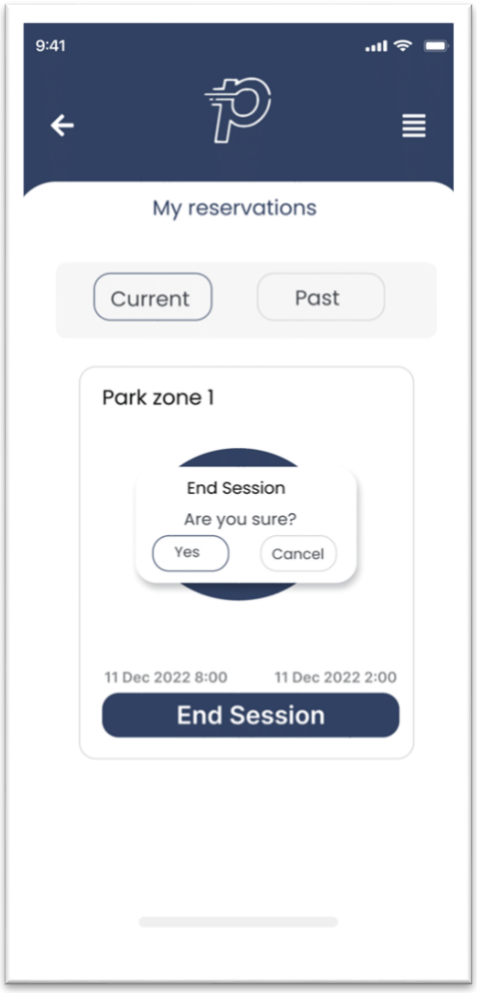


Figure 27 End session interface

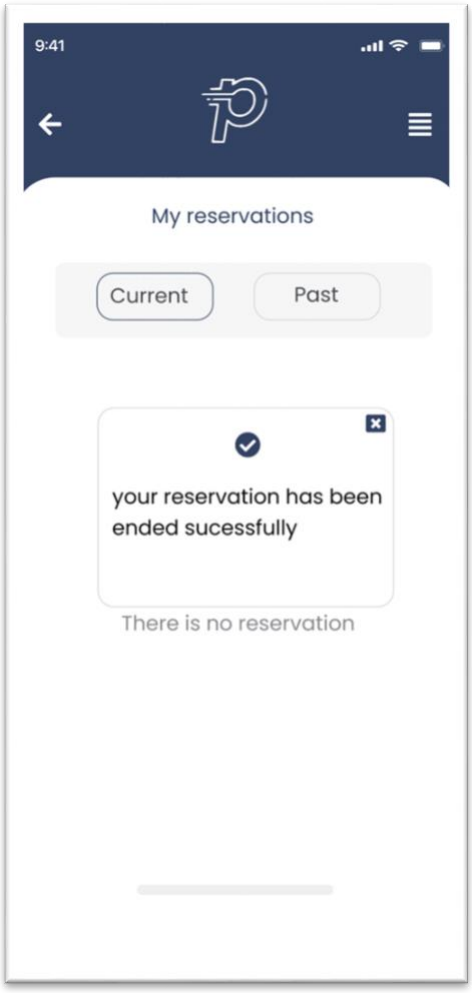


Figure 26 successfully ending session message

4.9.3.9 Add car plate number interface.

Each user must add the plate number and the car type to book correctly, as shown in the figure No. 28 below.

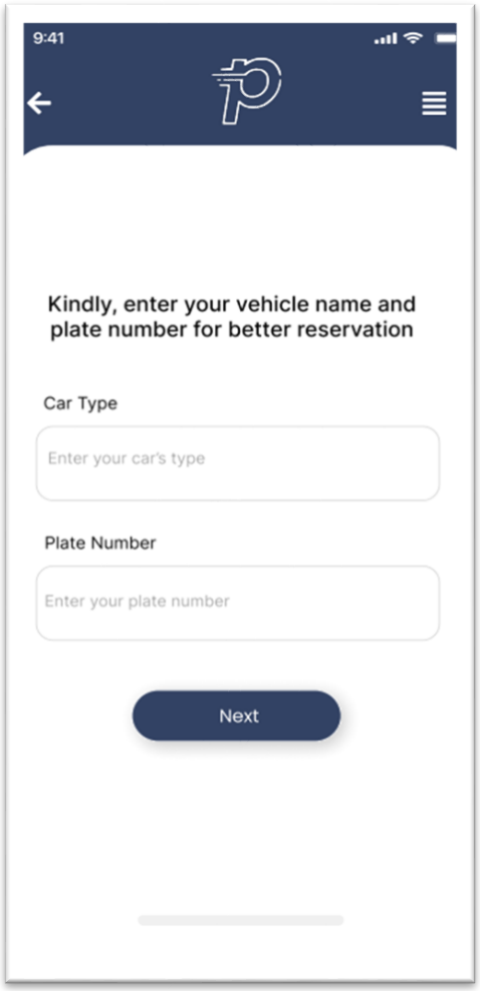


Figure 28 car plate number interface

4.9.3.10 select parking zone interface.

The user can choose the appropriate and available parking in the simplest way by clicking on the parking shown in numbers in the figure No. 29.

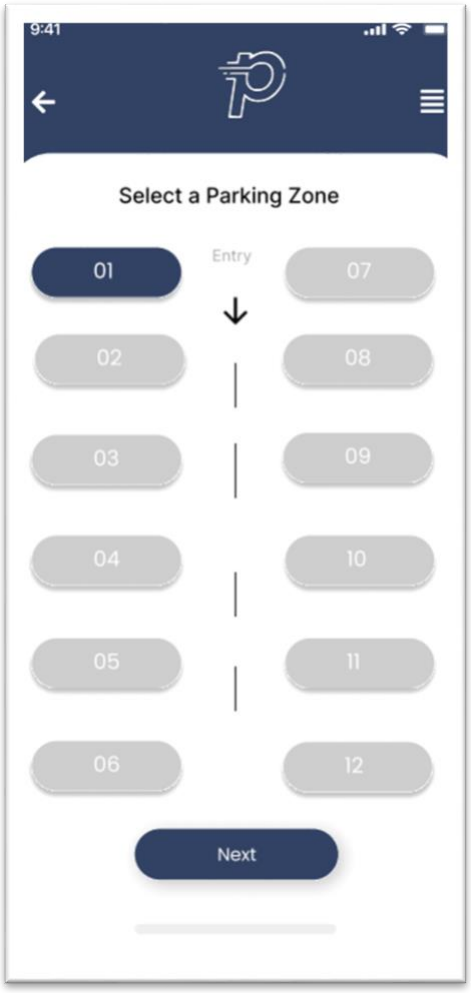


Figure 29 select parking zone interface.

4.9.3.11 Select number of hours interface.

The user must choose the number of hours as shown in the figure No.30 below .

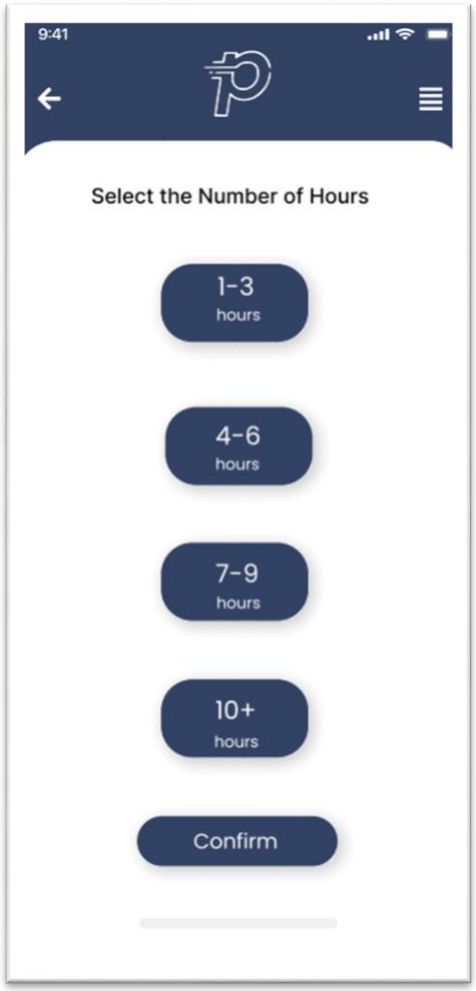


Figure 30 select number of hours interface.

4.9.3.12 Reservation ticket interface

After a successful reservation process as shoe in figure 32, the reservation ticket appears to us in the figure No. 31 The reservation details appear in the ticket.

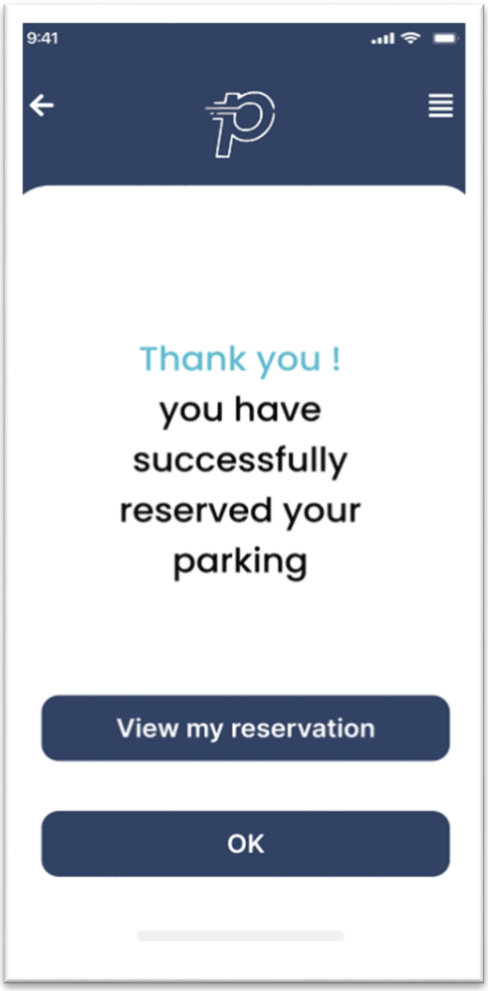


Figure 31 successfully reservation interface

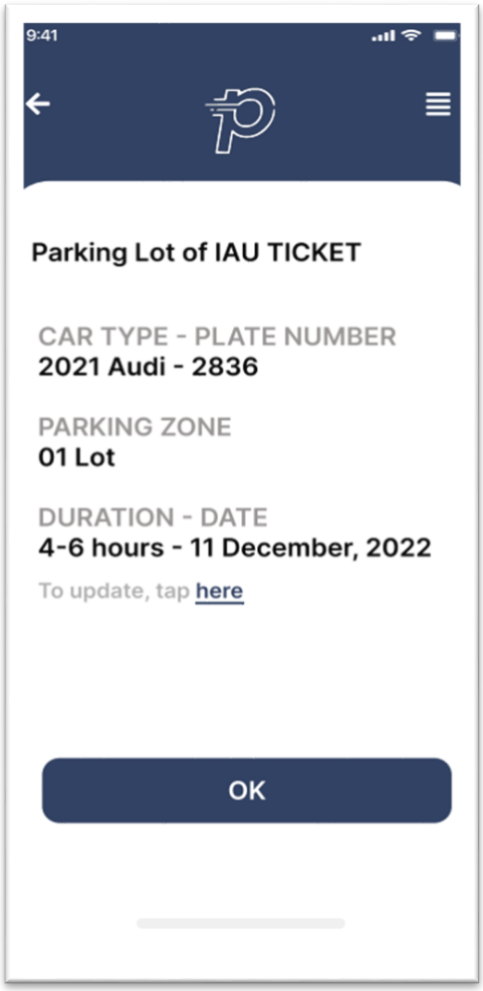


Figure 32 reservation tickets interface

4.9.3.13 More interface

The figure No. 33 displays the "More Page" which is considered one of the main pages, this page contains 4 sections, About, Settings, and Contact us. Each section will be described in detail below.

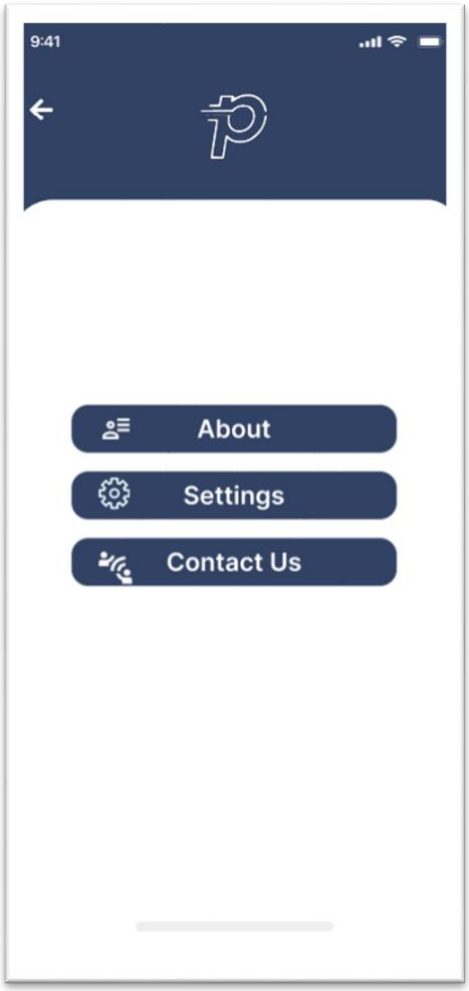


Figure 33 more interface

4.9.3.14 About interface

This page shown in the figure No.34 contains general information about Parkin app and a brief introduction about the system and the purpose of it.

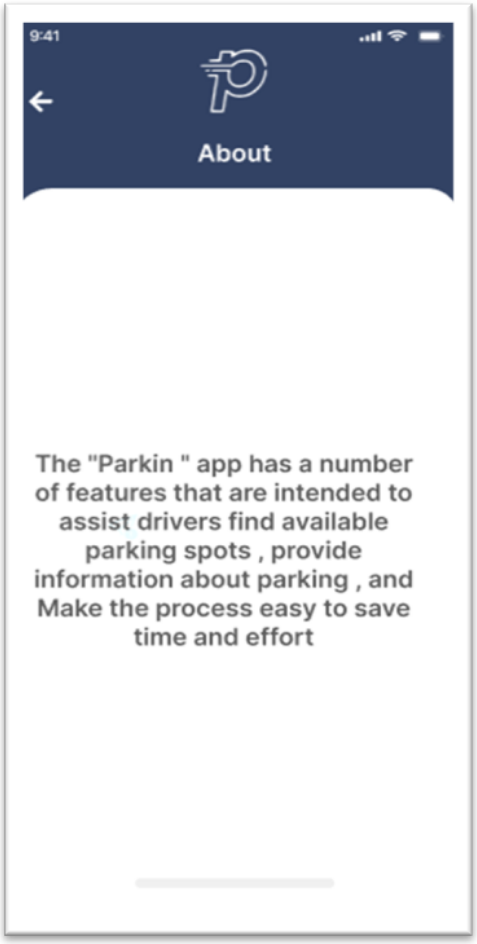


Figure 34 about interface

4.9.3.15. Settings interface

The figure No. 35 below displays the Settings page that contain various data such as the application version and the Delete account button. It also includes switches that behave based on the user preference such as, Notification Switch.

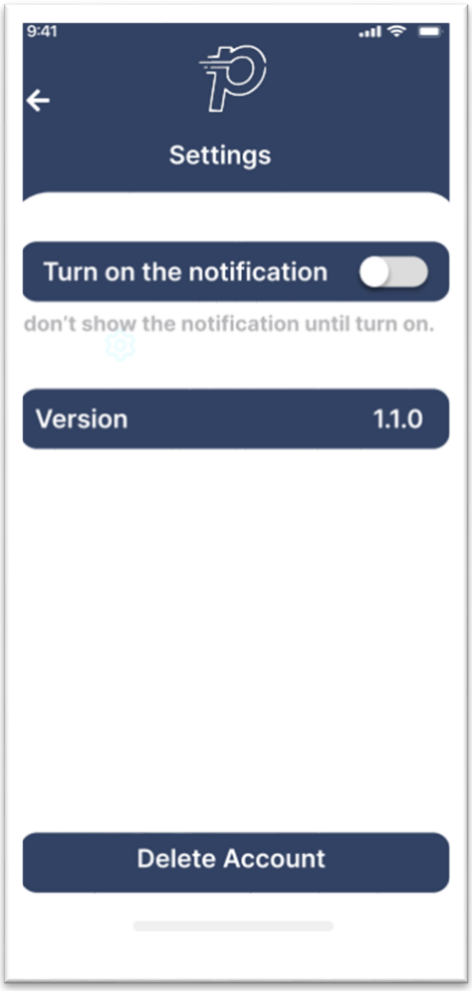


Figure 35 settings interface

4.9.3.16. Contact Us

In case the user has a problem to report or need further information he/she can got to this page for contact information, the figure No.36 below shows Contact Us Interface containing contact information which allows the user to communicate with the project developers either through social media or through email.

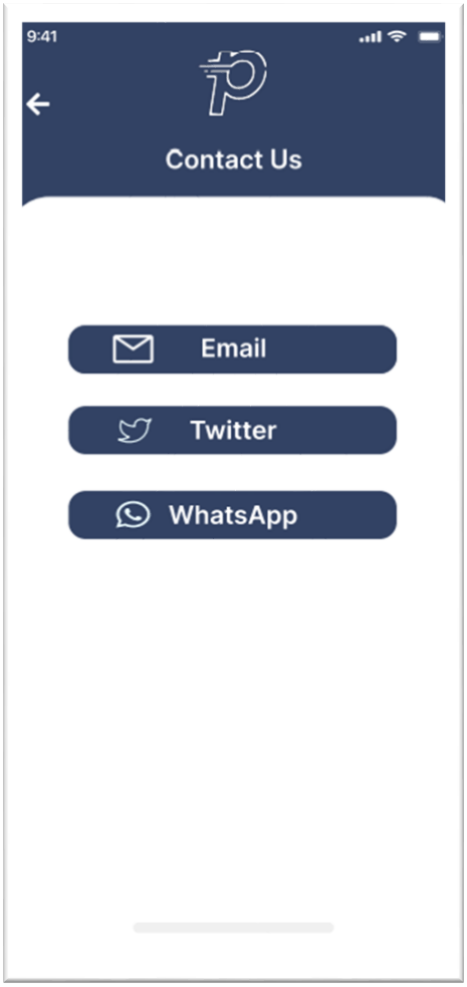


Figure 36 contact us interface.

4.10 System Architecture

This section will provide an overview of Parkin design architecture and its subsystem architecture.

4.10.1 Architecture Design Approach

Parkin will be using multi-tier Architecture, which will help the programmer to develop the system in easier and more flexible way since each tier will be developed simultaneously by different team members therefore, any updates on one tier will not affect other tiers. Parkin will include four architectural tiers:

- **User interface tier:** This is the highest tier. This tier is where the user can interact with the mobile application.
- **Software tier:** This is the middle to high tier. This tier is where the commands and code will be developed, logical decisions will be made using Python and Visual code Studio.
- **Data tier:** This is the middle to low tier. This tier is where the information will be stored and retrieved from the database.
- **Hardware tier:** This is the lowest tier; camera's readings will come from this tier.

4.10.2 Architectural Design

- **User interface tier:** This tier will provide the user with the data that is displayed from the database which will enable the user to interact with the system features, for instance, creating a new account, storing the car's name and plate's number, reserving a parking lot, selecting a period of time for reserving the lot and checking the app notifications.
- **Software tier:** This tier will enable the programmer to develop the user interface tier and will process the data coming from the data tier. This tier will include the data that has been retrieved from the data tier then it will be shown in the user interface tier.
- **Data tier:** The data tier will be working on the database. This tier will be used to store and collect user and hardware data therefore, it can be used in the development of Parkin app.
- **Hardware tier:** This tier will take the information from the ESP32-CAM to the Arduino, which will collect data such as the available and busy parking lots. After that the data that comes from the Arduino will be used in the software tier which then will process the incoming data to produce the final results that will be shown

on the user interface tier. The figure below will show the architectural design of Parkin.

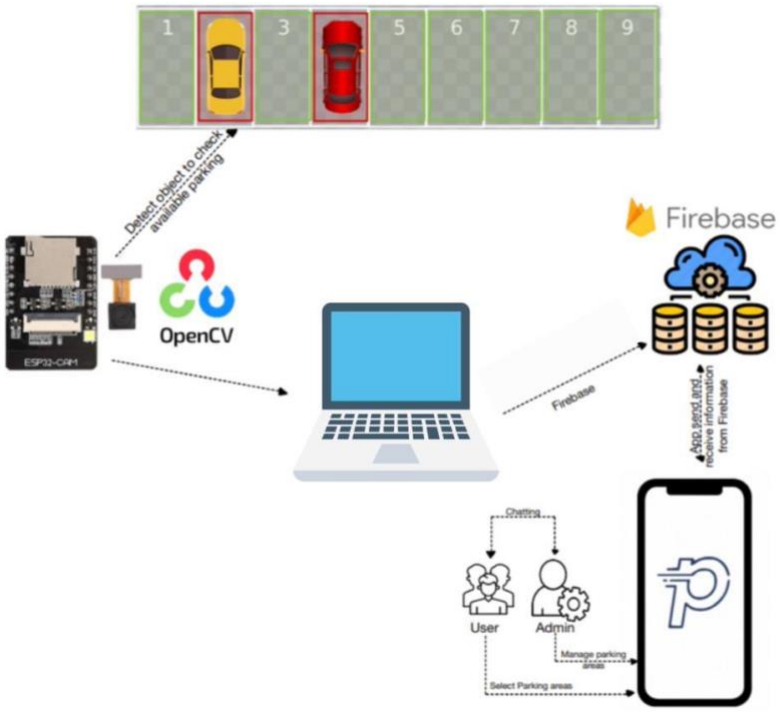


Figure37 Architecture Design of Parkin

4.10.3.Subsystem Architecture

This section will describe the functions and the processes of Parkin. it defines how the data is processed, how the hardware provides results, and how the data is stored. Which will be describes in the flowchart diagram of Parkin.

4.10.4 General View of Parkin

This section will describe the systems relationships and boundaries of Parkin. First, the user will run the Parkin application in their smartphones, then starting with ESP32 CAM it will collect the readings including available and busy parking lots.

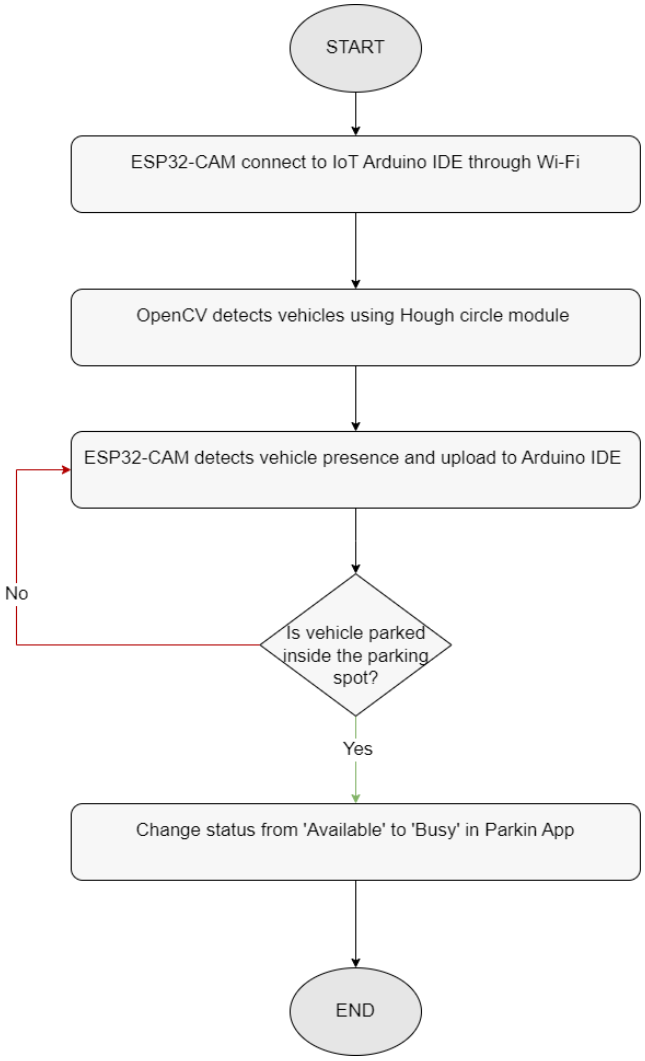


Figure 38 Flowchart of Parkin

Once the system starts, Arduino will be connected to the IoT Platform of the Parkin application through a Wi-Fi, then, the located ESP32-CAM will be detecting the availability of the vehicles and sends that information to the Arduino. OpenCV will be classifying the vehicles with the help of Hough circle function module. After that, if the vehicle is parked inside the parking slot, it will change the status from ‘Available’ to ‘Busy’, otherwise, the camera will keep on detecting the availability of vehicles.

4.11 Data Design

4.11.1. Data Description

4.11.1.1. User Data Table

The table below will address the user's data table that will be stored in database.

Table 23 User Data Description

Attributes	Type	Constraints
User_Name	VARCHAR (25)	Not NULL
User_Email	VARCHAR (20)	Not NULL
Password	VARCHAR (20)	Not NULL
Sticker_ID	VARCHAR (10)	Not NULL

4.11.1.2 ESP32-CAM Table

The table No.27 below will address the ESP32-CAM data table that will be stored in the Firebase database.

Table 24 ESP32 CAM Description Table

Attributes	Type	Constraints
AvailableLot_Number	INT (12)	Not NULL
BusyLot_Number	INT (12)	Not NULL
ParkingZone_Number	INT(12)	Not NULL

4.11.2. Data Dictionary

4.11.2.1. User Data Table

The table No.28 below will address the user’s data dictionary table that will be stored in the Firebase database.

Table 25 User Data Dictionary Table

Attributes	Description
User_Name	An attribute used to store the user's name.
User_Email	An attribute used to store the user’s email.
Password	An attribute used to store the user’s password.
Sticker_ID	An attribute used to create the user’s sticker’s ID.

4.11.2.2 ESP32-CAM Table

The table No.29 below will address the camera's data table that will be stored in the Firebase database.

Table 26 Camera Data

Attributes	Description
AvailableLot_Number	Attribute used to store the available lots' numbers.
BusyLot_Number	Attribute used to store the busy lots' numbers.
ParkingZone_Number	Attribute used to store the parking zone number.

4.11.3 Database Description

In Parkin Firebase will be created to store and manage the ESP32 CAM database such as, the available and busy lots. In addition, the Firebase database will be used to store user's data. The Firebase will be connected and established to the Parkin application.

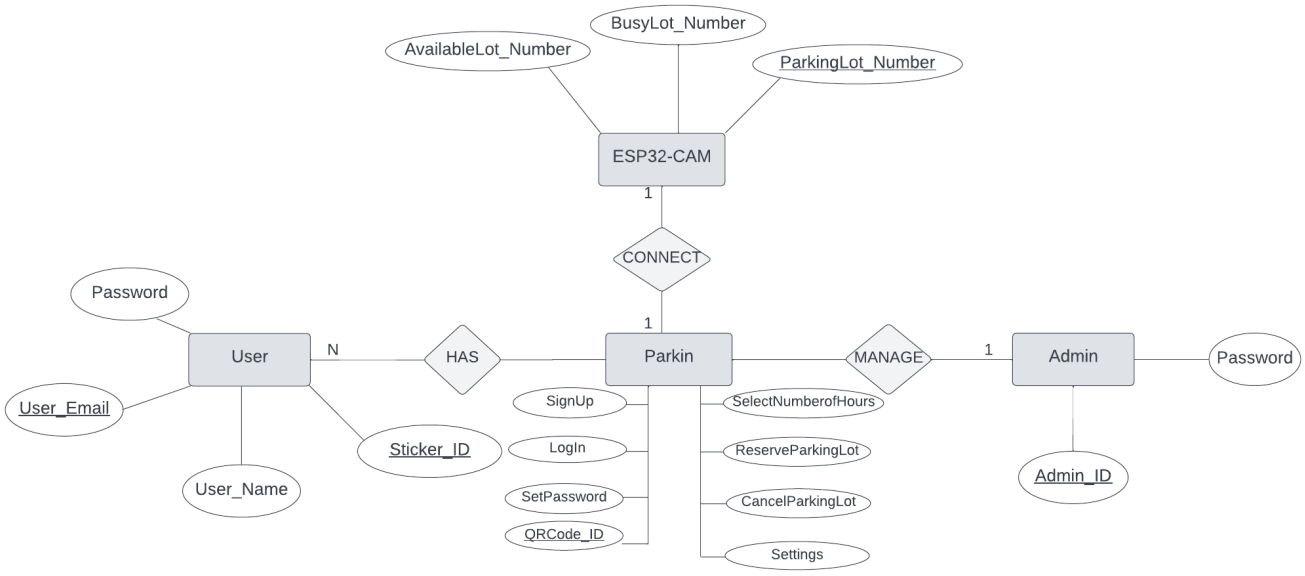


Figure 39 Database Diagram

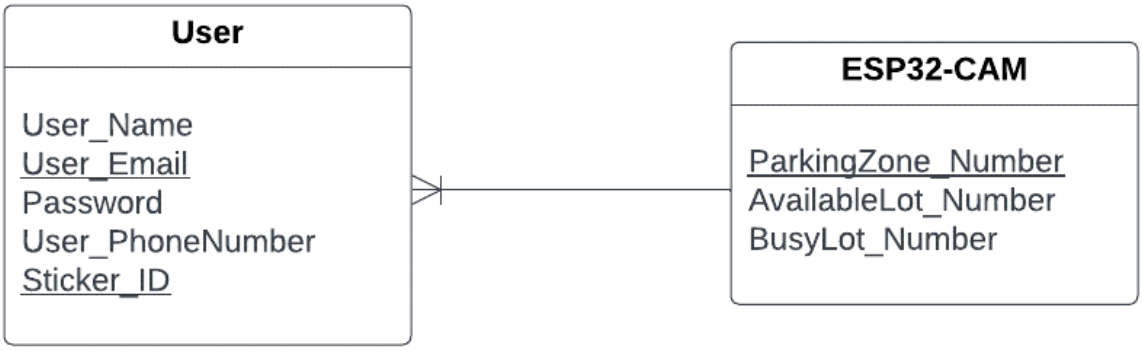


Figure 40 Entity-Relationship Diagram

The Entity-Relationship Diagram for the essential attributes is shown below.

The relational schema for the essential attributes is shown below.

User

User_Name	User_Email	Password	User_PhoneNumber	Sticker_ID

Figure 41 Relational Schema for User

ESP32-CAM

AvailableLot_Number	BusyLot_Number	ParkingZone_Number

Figure 42 Relational Schema for ESP32-CAM

4.12 Component Design

1) Object detection

```
def detectObject() {

receive object image from Camera

store data to ESP32-CAM

}
```

2) Send data to database

```
def sendData(){
receive data from ESP32-CAM

if(isEmpty != false){
send data to firebase
} }
```

3) Receive data from database

```
Db=firebase.database()

Value=db.child("child").get()

Print(Value.val())
```

4) Real-time data storage

```
def storeData(){
receive data from ESP32-CAM

store data to Firebase database
}
```

5) Display data on mobile application

```
def displayData():
def  recieveData():
print (Data)
```

4.13 Detailed System Design

This section describes the components of the system, each subsection of this section contains a detailed description of each component.

4.13.1. classification

The table 30 shows list the classification of each component in the Parkin.

14.13.2. Definition / Responsibilities

The table 30 shows list the components of the Parkin with its definition / responsibilities.

Table 27 Component's Classification, Definition, And Responsibilities

#	Components	Classification	Definition and Responsibilities
1	Receive data from database	Function	Reserve parking from available spots
2	Send data to database	Function	Collect the data from the sensor and send it to the database.
3	Real-time data storage	Function	Store real-time data to use it later.
4	Display data on mobile application	Function	Display the parking, available and reservation Area that are received from the database.
5	Detect the available spots in the parking lot	Function	Identify the witch spots are not resaved.
6	Reserve a parking space	Function	Users can reserve a parking space from the application
7	Manage account details	Function	Users can manage and edit their details account
8	Manage the users	Function	The admin can Manage all the user from application

9	Manage the spots parking area	Function	The admin can Manage all the spots area from application

4.13.3. Constraints

The table No.31 shows list the constrains of each component in the Parkin.

Table 28 Component's Constraints and Composition

#	Components	Limitation	Pre-condition	Post-condition
1	Receive data from database	-Data collected from the sensors -Database availability -Internet connection	Data availability	Data will be received from the database for further use.
2	Send data to database	-Data collected from the sensors -Database availability -Internet connection	Data availability	Data will be sent to the database for further use.
3	Real-time data storage	-Database availability -Internet connection	Data availability	Data will be stored temporarily for further use.
4	Display data on mobile application	-Internet connection	Working mobile application	Data received from the database will be displayed in the mobile application.
5	Detect the available spots in the parking lot	The accuracy of the sensor	Working sensor	Measuring and discovring the object
6	Reserve a parking space	-Data collected from the sensors	Working mobile application	Data received from the database will be

		-Database availability -Internet connection		displayed in the mobile application.
7	Manage account details	-Data collected from the sensors -Database availability -Internet connection	Working mobile application	Data received from the application will be stored in the database.
8	Manage the users	-Data collected from the sensors -Database availability -Internet connection	Working mobile application and database	Data received from the application will be stored in the database.
9	Manage the spots parking area	-Data collected from the sensors -Database availability -Internet connection	Working mobile application and database	Data received from the application will be stored in the database.

4.13. 4 RESOURCES

The table No.32 shows list the Resources of each component in the Parkin.

Table *29* Components Resources

#	Components	Resources	Description
1	Receive data from database	• Firebase database • visual studio code	Data will be retrieved from in firbase database and visual studio Receiving end of the data
2	Send data to database	Firebase database	Data will be sent in this real-time database
3	Real-time data storage	• Firebase database	Data will be stored temporarily in the database for further use

4	Display data on mobile application	• visual studio	Display data that are received from the database in graphs.
5	Detect the available spots in the parking lot	• ESP32-CAM	Read the data and transfer it to the database
6	Reserve a parking space	• visual studio • Firebase database	Reserving a parking from app and store it on the database
7	Manage account details	• visual studio • Firebase database	Reserving an action from app and store it on the database
8	Manage the users	• visual studio • Firebase database	Reserving an action from app and store it on the database
9	Manage the spots parking area	• visual studio • Firebase database	Reserving an action from app and store it on the database

4.14 PROCESSING

This section shows a detailed process design of Business Process Modeling for both the user and admin**.**

4.14.1 System User Process

The first interface that will appear for the user is the walkthrough interface, when the user clicks on finish the sign in interface will appear, if the user signs in, the system checks on the database **.**

If it did not find the user profile, it will notify him, and the user can register as a new user. Still **,**suppose the system is verified, and the entries match the file saved in the database. In that case **,**the user will be able to enter the system, and the home interface will appear to the user, The user can check the Setting interface that contain (General, help, contact us, about us) buttons , if the user want (sign out, open/close the relay, open close

the notification) click the general button. This figure 43, 44 and 45 demonstrates the process for system users.

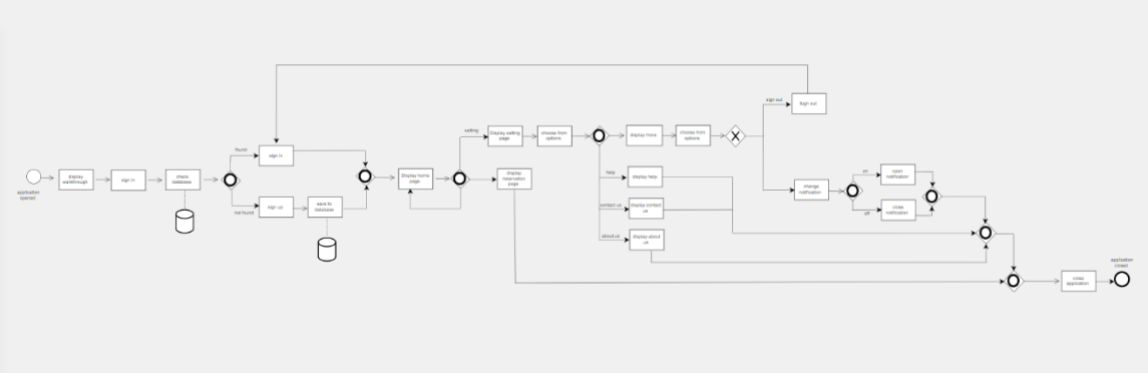


Figure 43 system user process

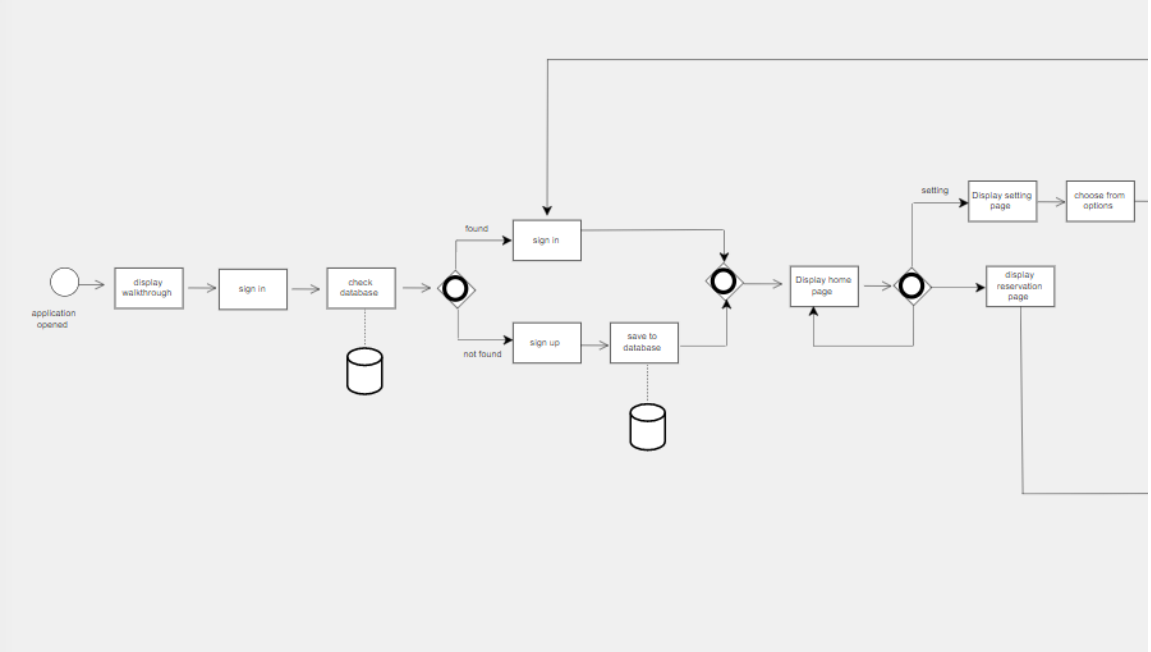


Figure 44 system process user part1

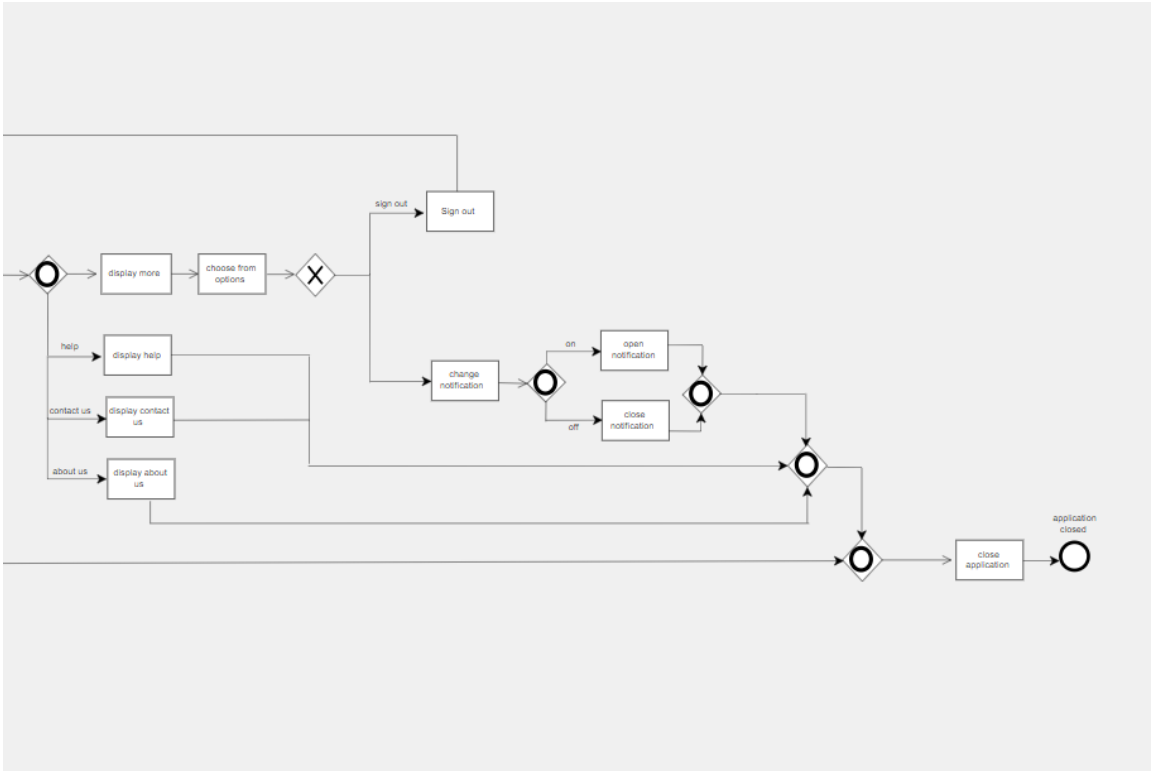


Figure 45 system process user part2

4.15. DETAILED SUBSYSTEM DESIGN

This section shows a detailed design of SPS subsystems with information flow as a data flow diagram the Database.

4.15.1. ESP32-CAM Data Flow Diagram

This figure 46 demonstrates the data transformation from the ESP32-CAM to the application.

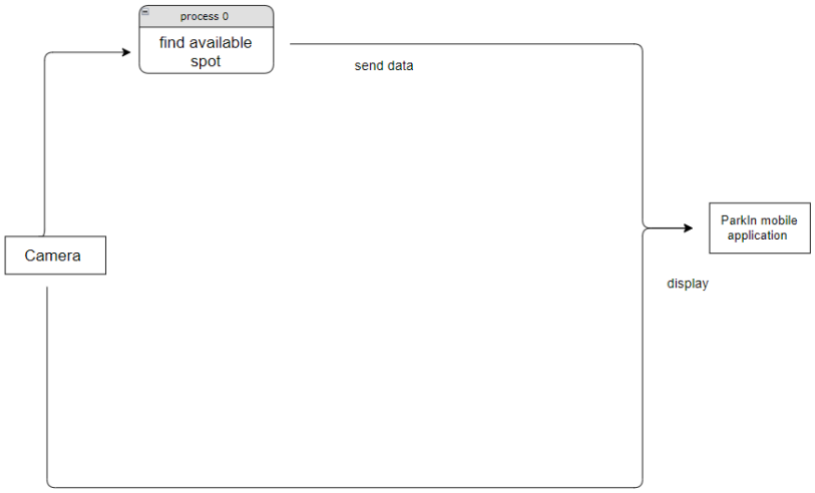


Figure 46 camera data flow diagram

4.15.2. Database Data Flow Diagram

This figure 47 demonstrates the data transformation from the application to the database

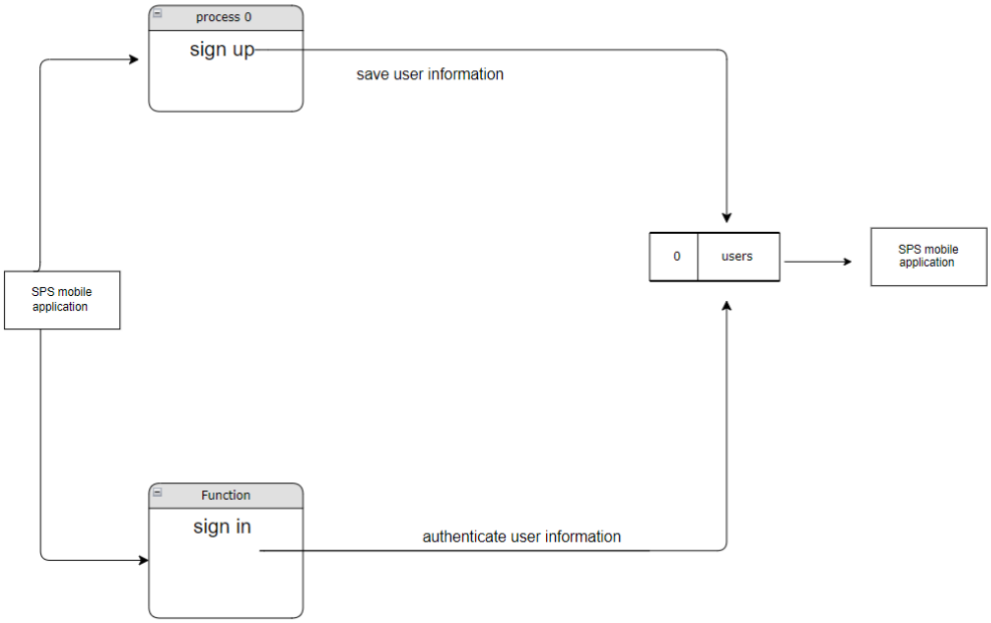


Figure 47 database data flow diagram

4.15.3. Mobile Application User

This figure describes the sequence diagram for the mobile application users that contains multiple Objects such as Homepage, sign in and sign up, in addition it also explains the interaction between objects and the mobile user.

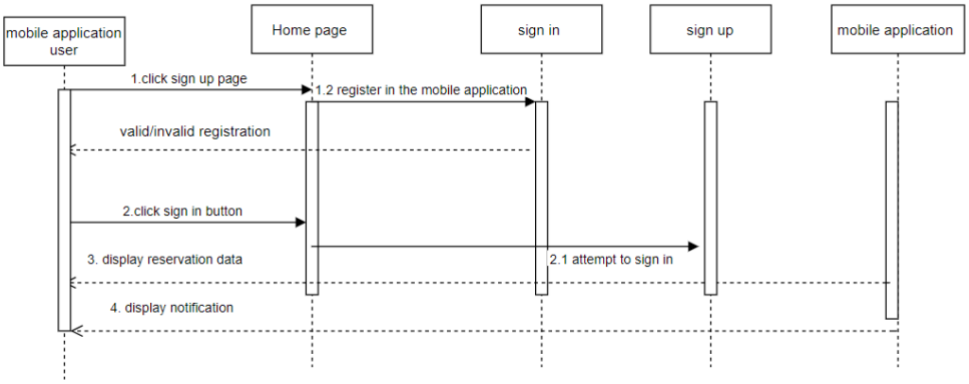


Figure 48 Sequence Diagram for The Mobile Application Users

4.16 Requirements Traceability

Table No. 33 shows the requirements traceability matrix, which displays the relationships between all Parkin interfaces for the software, hardware, mobile-application user, and administrator requirements.

Table *30* Requirements traceability

Requirements	**Hardware and Software**			**User**		**Admin**	**Database**	
Interfaces	Replace damaged hardware components	Send Park busy and availability information	Solve software technical problems	Create new personal account	Access an exited account	Manage Parkin	Store in database	Retrieve from database
Sign up				√		√	√	
Log in					√	√		√
Forget account password					√	√	√	
Home page		√				√		
Features		√				√	√	√
Help						√		
Parkin	√	√	√			√	√	√

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Introduction

Our project is intended to optimize vehicle parking reservations. This chapter is offered to comprehensively describe the project implementation process and track the team's progress. In addition, this chapter describes the modifications performed prior to the implementation phase. It also describes the challenges faced and the lessons learned throughout this period. In addition, an IEEE test plan has been developed to define the test strategy for the Parkin project during the testing phase. This chapter has been broken into sixth sections. In the first section, we will justify the changes made during the proposal process. The second section will discuss the team's implementation process progress and results to date. The third section addresses the problems we encountered in the second section. The fourth section outlines the test strategy for the project. The fifth part highlights the emerging programming languages, tools, and methodologies learned during the period of implementation. The final section eventually summarizes the chapter.

5.2 Testing plan

5.2.1 INTRODUCTION

This section will introduce the software test plan (STP) of Parkin application, a smart parking space detection system that offers a parking spot with many distinctive features which will be tested. Moreover, all system testing will be scheduled. Software test plans also list the products, features, types of testing, persons, resources, timetable, and risks connected with testing. This section of the Software Test Plan (STP) contains the main objectives, testing strategies, and scope.

5.2.1.1 Objectives

The purpose of writing a software test plan is to instruct team members on how to conduct testing. This includes what resources and tools are needed, what each member's responsibilities are, and how they should handle changes. Furthermore, it should enable us to adapt to changes and handle them. As part of the Parkin system software testing, which aims to identify errors and defects, and improve the performance and security of software, an integration process between the software and hardware will be implemented to provide a high-quality product that performs its intended functions as expected and meets user requirements while providing them with a high-quality experience. The tests will be done using a variety of testing methods and both human and non-human resources. Due to the fact that every feature is a high priority for the release of the system, all features will be tested.

5.2.1.2 Testing Strategy

The objective of the testing procedure is to distinguish between intended and current application functionalities. The Parkin application will utilize both white-box and black-box testing procedures. White-box testing is mostly used to study the internal structure of an application to verify the input/output flow, fix the design, and test each component of the platform. In addition, black-box testing will be utilized to evaluate the compliance of our platform components with SRS's established functional criteria.

5.2.1.3 Scope

During the development of the application, the testing procedure will be implemented at many stages of the application's life cycle. Primarily when functions are done and ready to be tested (i.e., "features to be tested" section). Testing is a dependent activity. Consequently, a testing strategy is a continuous activity conducted throughout the system development life cycle.

5.2.1.4 Reference Material

The following document was used as references for the development of the Software Test Plan:

- Software project management plan (SPMP)
- Software Requirement Specification (SRS)
- Software Design Specification (SDS)

5.2.2 TEST ITEMS

This section will describe the details of the testing process implemented in Parkin application which includes program modules, job control procedures, user procedures, and operator procedures.

5.2.2.1 Program Modules

In program modules, Parkin application includes 3 main user modules which are:

- **Admin interfaces:** ensuring that admin interfaces are well-designed and function properly.
- **User interfaces:** ensuring that the application's user interfaces are well-designed, user-friendly, convenient, and functioning appropriately.
- **Database:** the developer is responsible for storing and maintaining data to ensure that all data recorded by the ESP32-CAM is accurate depending on the OpenCV module.

These modules will be disassembled into small individual components, and all of them will be tested to ensure that they conform to the specifications and are free of critical errors. They will then be reassembled and subjected to a comprehensive test to ensure that they function effectively and efficiently.

5.2.2.2 Job Control Procedures

The team should establish a stable schedule for communicating via Zoom, WhatsApp, OneDrive, and face-to-face meetings in order to confirm each test stage. They must also agree on the standard of the interfaces as well as the restrictions of user capabilities and privileges, which includes the interactions between the user and the interfaces.

5.2.2.3 User Procedures

In the Parkin application, all documents mentioned in the test process, such as SPMP, SDS, and SRS, are utilized, and all team members will review the final report to verify that all chapters are clear, exhaustive, and conform to IEEE standards.

5.2.2.4 Operator Procedures

A test will be implemented on a prototype of the designed hardware to ensure that the image processing is done successfully. For the Application, the iOS operating system will be used to determine if all Features are successfully interactive.

Using software quality assurance techniques, strategies, and processes to test and debug any faults, the deliverables' quality will be validated.

5.2.3 FEATURES TO BE TESTED

Table No.34 below describes the features that will be tested, which includes the main functions.

Table 31 features to be test.

Req ID	Function Requirement	Applicable Role	Description
1	Sign up	User	Users must create an account; it is a mandatory step in order to get access to all the services available on the platform. In addition, basic information such as username, phone number, and e-mail must be provided.
2	Sign in	Amin/user	Username and password are required inputs for each time a user logs in to the platform. Therefore, the interface will be

			displayed based on the user's role, whether it is an admin/user.
3	Sign out	Amin/user	Able for both users and admin to log out successfully from their accounts on Parkin application.
4	Reset forgotten password	User	The user is able to retrieve their password in case they forgot it.
5	Search for areas to park in	User	User will have different areas to park at and he will be able to choose from.
6	Reservation	User	User will be able to reserve a spot
7	Send Notifications	Admin	The admin has the ability to send notifications.
8	Update/Delete/Display areas for parking spots	Admin	The admin has the ability to Update, Delete, and add the areas for the user to choose from
9	Update information	User	Users can change their account information. (For example, phone number, email address).

5.2.4 FEATURES NOT TO BE TESTED

There will be no (features not to be tested) since all the system's features will be tested.

5.2.5 APPROACH

This section will go through and describe the tests done on the Parkin application to make sure that all its features respond and perform as intended.

These tests will be covered in this section:

• Component Testing

• Integration Testing

• Interface Testing

• Security Testing

• Performance Testing

• acceptance Testing

• Performance Testing

• Recovery Testing

5.2.5.1 Component Testing

Before integration of each piece of code, component testing is used to test the smallest components of the program one by one to ensure that each component (such as registration, login, application) of the platform is operational and error-free. As a result, the team will run this test using white box testing to highlight the logic, data streams, and data structures of the platform. Additionally, the team will use Branch coverage which is an approach used by the white box testing to ensure the potential branch of the casing is checked at least once. Also, the team aims to cover pathways 100% coverage.

5.2.5.2 Integration Testing

The smallest software units are joined with other units, modules, or components during integration testing after having been tested independently to assure and confirm that each unit, module, and component of the platform is functioning. The top-down technique, in which the top module is tested first, followed by branches of the module, is used for integration testing until all the unit modules in the package are tested, at which point all the platform packages are tested. To evaluate the platform's functioning once it has been integrated.

The test cases in the tables below are intended for integration testing. These test cases are samples developed to allow us to run preliminary tests on the platform to see whether the functionality passed or failed with two cases (Successful/Unsuccessful), in addition to offering greater explanation and clarification of the concept. Big bang testing technique will be implemented by testing all the components as a single unit. Although, for testing purpose we will use "Abeer@gmail.com" as a valid admin email address with "Ab9@_021" as password. Table No.35 to Table No.39 will show the test case .

- **Sign in.**

Table 32 sign in successful test case

Successful test case				
Input	**Expected Output**	**Actual Output**	**Result**	**Comment**
Clicks on the "Sign in" button	The user will have the ability to log in in the account.	The user logged in to the account successfully.		The user is directed to the homepage.

- **Parking reservation.**

Table 33 Parking reservation successful test case

Successful test case				
Input	**Expected Output**	**Actual Output**	**Result**	**Comment**
- **Id number: "**2180000269**"** - **Reserved time: "**1 hour**"** - **Spot area**:"1"	A successful pop-up notification appears once the reservation has been successfully uploaded, saved, and stored in the database.	The user reserved a parking successfully.		The user is directed to the Summery of the reservation ticket.

Table 34 Parking reservation unsuccessful test case

Unsuccessful test case				
Input	**Expected Output**	**Actual Output**	**Result**	**Comment**
- **Id number: "**218annm69**"** - **Reserved time: "**1 hour**"** - **Spot area**:"1"	An error message will appear next to the incomplete labels and the request won't be saved or submitted.	The offer will not be completed or delivered, and an error message will appear next to the labels that aren't complete.		-

• Reset the forgotten password

Table 35 Reset the forgotten password successful test case.

Successful test case				
Input	**Expected Output**	**Actual Output**	**Result**	**Comment**

-Email: "Abeer@gmail.com " **-New password:** Ab9@_021" "	The user will be able to access their account and log in.	The user logged in to the account successfully.		The user is directed to the homepage.

Table 36 Reset the forgotten password unsuccessful test case.

Unsuccessful test case				
Input	**Expected Output**	**Actual Output**	**Result**	**Comment**
-Email: "Abeer@gmall.com " **-New password:** "Ab9@_021"	The user will Not be able to access their account and an error message will be shown near the labels with the wrong entry to	An error message will appear next to the labels with the incorrect entry, and the user will not receive the email.		The user will be navigated to the sign-in page.

5.2.5.3 Interface Testing

The team in the interface testing of Parkin application will assure and confirm that the interfaces will react and coordinate in an efficient manner, navigate smoothly, the error message will appear when it's needed, transfer data correctly and without errors. Also, usability issues in user interface designs may be found through heuristic evaluation. our group will analyze the interface during a heuristic assessment to determine whether or not it adheres to established usability guidelines (the "heuristics"). Each member examines the interface on their own to do heuristic assessment. This process is essential to ensure that each evaluator provides fair and objective assessments. and that they satisfy the design specifications given in the SRS previously.

5.2.5.4 Security Testing

To ensure users that their data and information are secure and secured on the Parkin application from being altered, updated, and accessed by an unauthorized user, security testing will be carried out. For instance, the database will encrypt the user's password. The security testing will also address the following key issues:

• Privacy: Ensure that any sensitive data relating to users is encrypted.

• Authentication: The platform conforms to and verifies the accessed user using a special username and password.

• Authorization: Check the platform's permissions systems for all users

The tables from No.40 to No.44 below are test cases designed for security testing.

- **Length of the Password**

Table 37 Length of the Password

Equivalence Partitioning		
Invalid	**Valid**	**Invalid**
Length < 8	8 ≤ Length ≤ 12	**Length > 12**
E.g., AbC357	**E.g., SbKtt6083**	**E.g., Rm33sRsM10047777***

Table 38 successful test case of the password

Successful test case				
Input	**Expected Output**	**Actual Output**	**Expected Result**	Comment
Enter password "ArCj135" in Sign up or reset password	**The password will not be accepted since the password is less than 8 characters.**	**Unacceptable password.**	**Pass.**	**-**

Table 39 unsuccessful test case of the password

Successful test case				
Input	**Expected Output**	**Actual Output**	**Expected Result**	**Comment**

Enter password "AbCd2324*" in Sign up or reset password	**The password will be accepted since the password is more than 8 characters.**	**Acceptable password.**	**Pass.**	**-**

- **Authorized Admin**

Table 40 successful test case of Authorized Admin

Successful test case				
Input	**Expected Output**	**Actual Output**	**Expected Result**	**Comment**
Username: "admmin1" **Password:"**ad11*2mm"	The admin has access and has been given authorization to see the data, which is accurate.	The admin was given authorization to access and perform tasks and the submitted data are accurate.	Pass.	-

Table 41 unsuccessful test case of Authorized Admin

Unsuccessful test case				
Input	**Expected Output**	**Actual Output**	**Expected Result**	**Comment**
Username: "admmin1" **Password:"**ad11*2mm"	The entered data is not correct on the database and the system will allow this unauthorized user to access	The error message does not appear, and the user will navigate to the home screen	Pass.	-

5.2.5.5 Recovery Testing

The Parkin Application's recovery testing makes sure that, in the event of a system breakdown or crash, all application functions continue flawlessly and properly. After identifying the areas of failure, the system will perform partial recovery testing before being evaluated completely and the team will ensure the recovery done in a good way.

5.2.5.6 Performance Testing

In this test, software performance is evaluated while it is being used as part of an integrated system. Testing is done at every stage of the procedure. It is possible to evaluate the effectiveness of each individual module when tests are run. However, until all components are integrated, a complete assessment of a system's performance cannot be made. The performance test is used for the Parkin project to find performance issues that may be brought on by a lack of server resources, slow network bandwidth, insufficient database capabilities, issues with sensor and Firebase connections, or other hardware or software problems that could impair client-server performance.

The table No.45 below shows the test cases conducted:

Table 42 Performance Test case

Test Case	Expected output	Actual Output
A good quality camera is connected to the other equipment correctly.	The free spots detected.	as expected**,** output
Poor quality camera is connected to the other equipment correctly.	The sensor will work for only short amount of time, and the reading will not be precisely detected.	as expected**,** output
Fast internet connection	The system will be able to do it functions successfully and efficiently.	as expected**,** output
Low or unavailable internet connection.	The system's performance will be slow, or the functions will not be working at all.	Same as Expected output

Load testing:

Is a type of performance testing that examines how well a system performs under real-world load situations. We are going to add stress to the program to test how well he can handle it.

The table No.46 and No.47 below shows the test cases conducted:

Table 43 load test Successful test case

Successful test case			
Input	Expected output	Expected result	Comment
200 users present in the platform in real-time. (Desired load)	The system will operate properly as expected	pass	-

Table 44 load test Unsuccessful test case

Unsuccessful test case			
Input	Expected output	Expected result	Comment
1000 users present in the platform in real-time. (Over the desired load)	The system will not be able to operate properly as expected	pass	-

Availability:

The table below will explain the availability of the software giving the MTTR which is the total time for maintenance over the total repairs. And the MTTF which is the total time over the number of units tested. Finding this will give us the percentage of how long the system is available. The table No.48 and No.49 below shows the test cases conducted:

Table 45 Availability Successful test case

Successful test case			
Input	Expected output	Expected result	Comment
(MTTR) Mean time repairs= 2 hours **(MTTF) Mean time to fail =800 hours**	Availability=[MTTF/(MTTF+MTTR)] X 100 Availability= [800 / (800+2)] X 100 = 99.7%	pass	The platform will be available 99.7% which means most of the time.

Table 46 Availability Unsuccessful test case

Unsuccessful test case			
Input	Expected output	Expected result	Comment
(MTTR) Mean time repairs= 22 hours **(MTTF) Mean time to fail =1500 hours**	Availability=[MTTF/(MTTF+MTTR)] X 100 Availability= [150 / (150+22)] X 100 = 87.20%	pass	The platform will be available 87.20% which means it is inefficient.

5.2.5.7 Regression Testing

Following a software system modification, a regression test is performed. In this project, regression tests should be run anytime new parts are introduced or the program is modified since they assist to guarantee that changes do not result in unexpected behavior or more problems. The team members are following the strategy outlined in the report on the software requirements specifications during the implementation phase (SRS).

5.2.5.8 Acceptance Testing

Acceptance tests are testing carried out by the user to find any potential flaws prior to accepting a product from the developer. Instead of software engineers, end users carry them out. The acceptance test for the Parkin project will be performed by the end user with the assistance of the development team, and it won't start until there aren't any major or critical flaws.

5.2.6 PASS / FAIL CRITERIA

This section determines the criteria used to determine if a testing component has passed or failed a test to determine whether the testing process should be stopped or continued. Parkin needs to fulfill specific criteria that are listed in the SRS document. To make sure the platform is functioning effectively with few errors and faults, several test cases will be run.

5.2.6.1 Suspension Criteria

There will be no further testing until the suspension criterion is determined. The suspension criteria will halt the execution of testing operations. Testing will be paused, and problems found will be repaired immediately to restart testing if more than 50% of test cases fail to pass owing to significant issues.

- The hardware components may contain different problems such as damaged sensors, ESP32-CAM not Recognized which could limit the testing process.
- Significant changes in the hardware and software requirements will make the testing process more challenging.
- The software may have certain challenges as the testing process moves forward, such as delayed system response and abrupt program crashes.
- Network connection: if the network is down, the entire project will be delayed because the project does not enable offline connections.
- Inadequate testing conditions.

5.2.6.2 Resumption Criteria

The testing operations will restart upon fulfilling the resumption requirements if the discovered mistakes and flaws that caused the testing to be paused have been corrected.

- Offer new hardware equipment to replace any that has been destroyed.
- Locate and fix any software flaws that might hinder testing or render the system inoperable.
- Adapt to any modifications to the hardware or software specifications.
- Fix network connectivity and create a reliable WIFI connection.
- Offer enough testing environments to support moving on with the testing process.

5.2.6.3 Approval Criteria

The following criteria will be considered to approve the testing:

- The outcome satisfies the standards for quality and operation.
- Getting the end-acceptance user's and pleasure with the system.
- Obtaining the consent of the system's creators.
- Verify that the system is error-free and operating properly.

5.2.7 TESTING PROCESS

Expected deliverables, testing tasks, responsibilities, resources used, and a projected schedule are all included in this section.

5.2.7.1 Test Deliverables

The following documents will be delivered both throughout the implementation of testing activities and once the testing phase has been completed:

• Software Test Plan (STP).

• Software Test Report (STR).

5.2.7.2 Testing Tasks

During the testing phase, the following tasks will be implemented:

• Gather all the paperwork needed for the testing process.

• Clearly define and explain the testing tasks, roles, and responsibilities to every team member.

• Examine and verify the equipment and supplies required to carry out the testing process.

• Set up testing environments for the network, software, and hardware.

• Establish a schedule for the test process and execute each test case.

• Carry out testing activities using a variety of methods and procedures.

• Verify that the platform complies with the established software specifications.

• Evaluate Parkin platform's performance in a variety of unique situations.

• Keep track of all flaws discovered during testing process and address them.

• Constantly keep an eye on Parkin platform in case something occurs.

5.2.7.3 Responsibilities

In fact, the testing activities must be planned, designed, carried out, reviewed, and resolved by every member of the Parkin team. The project testing leader Ghadah will oversee the testing of the platform's components during this step, which will also be carried out by testing team members. The testing process activities will also be evenly distributed among team members to examine the testing results.

5.2.7.4 Resources

The resources needed for carrying out testing activities are shown in the table No.50 below.

Table 47 resources

Hardware Resources	Software Resources	Human Resources
• **Laptops.** • **ESP32-CAM.**	• Android Studio. • Notepad++. • Arduino IDE • Firebase Database. • OpenCV.	Project testing leader and team members.

5.2.7.5 Schedule

The testing schedule for the Parkin platform is detailed in the table No.51 below.

Table 48 schedule

Test Activity Name	Start Date	End Date
Component Testing	28th of January, 2023	31st of January, 2023
Integrating Testing	1st of February, 2023	2nd of February, 2023
Interface Testing	3rd of February, 2023	5th of February, 2023
Usability Testing	3rd of February, 2023	5th of February, 2023
Security Testing	6th of February, 2023	6th of February, 2023
Performance Testing	7th of February, 2023	7th of February, 2023
Recovery Testing	8th of February, 2023	10th of February, 2023

5.3 Changes from the proposal phase and justifications.

Table No.52 below defines the changes that occur in the implementation phase beside to their justification.

Table 49 Changes and justification

Change	Justification
Dispense "Raspberry pi".	Due to the small area that we will cover, we are adopting a laptop using OpenCV instead that has a similar as possible for Raspberry Pi.
Dispense "Geany IDE"	Geany IDE provides coding in Raspberry Pi, Unfortunately, we dispense with Geany IDE, and we are building our application with Arduino IDE using Python for (back-end).
Adopt "ESP32-CAM mega"	As we can use low-cot hardware since it is a prototype and can give as same functionality.
Enhance "Reserve parking" feature.	One of the valuable comments in the last semester about this feature was that it is "time-consuming and inefficient for the user", therefore, the comment was taken into consideration and this feature was enhanced so that the reservation will be with email address instead of adding car plate number.
Dispense "Extended reservation"	Due to high demand o the parking area, extended reservation might lead users to use time more than they need. So, in order to avoid this problem, the user will be restricted to the hors that he is already reserved.
Enhance "Requirements analysis"	The requirement analysis needed to be reviewed also, an appropriate model should be used to depict them. So, we enhance the requirements with a better model.
Modifying and extend ER diagram	There were some errors and comments regarding the ER diagram, which was taken into consideration, thus, it was re-designed from scratch to support all scenarios.

Modifying the whole use case diagram	The use case was not compatible with the functional requirement aligned with the model, so we ensured that it is confirmed with all models.
Modifying Scope	We wanted to improve our scope. Therefore, a new methodology has been added that is clearly listed along with the assumptions.
Add Business Model	Unfortunately, the business impact was not discussed in the proposal, so we have adopted a business model.
Adopt Hough circle method	We have adopted the Hough circle function method for detecting the object as a method to identify the present objects.

5.4 Implementation progress

The project consists of two phases. The first phase involved assembling the hardware components by connecting the ESP32-CAM to the OpenCV, followed by its development and programming using the Arduino IDE. Each component was installed correctly and given instructions through programming code, and the device's features were tested using AI Thinker. In the second phase, the team programmed the application using Flutter, completed the construction of the main interfaces, and is currently working on connecting the app and the device to the Firebase database. Once this stage is complete, the team will ensure that the project meets all the necessary requirements. Here are some screenshots of the team's progress throughout the implementation phase.

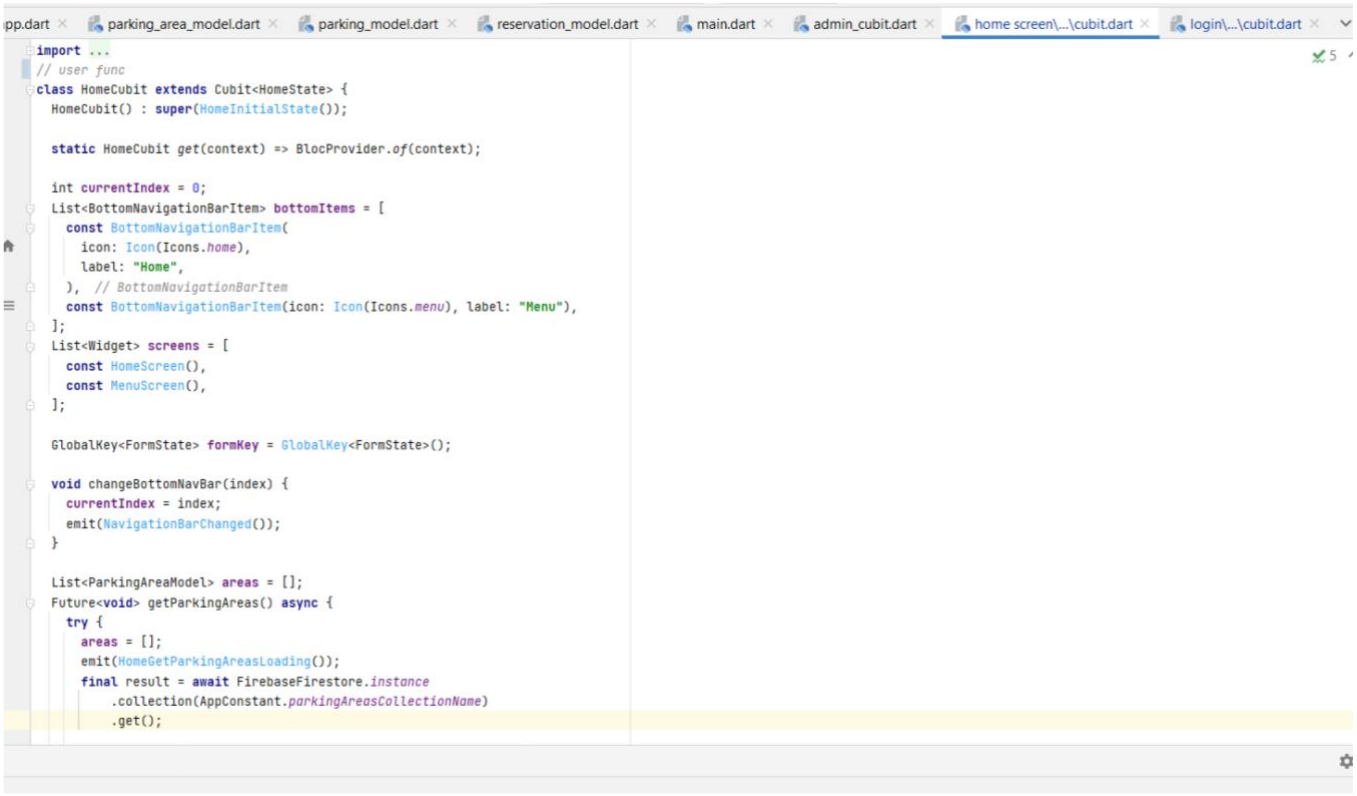


Figure 49 Home Page Code Snippet -1

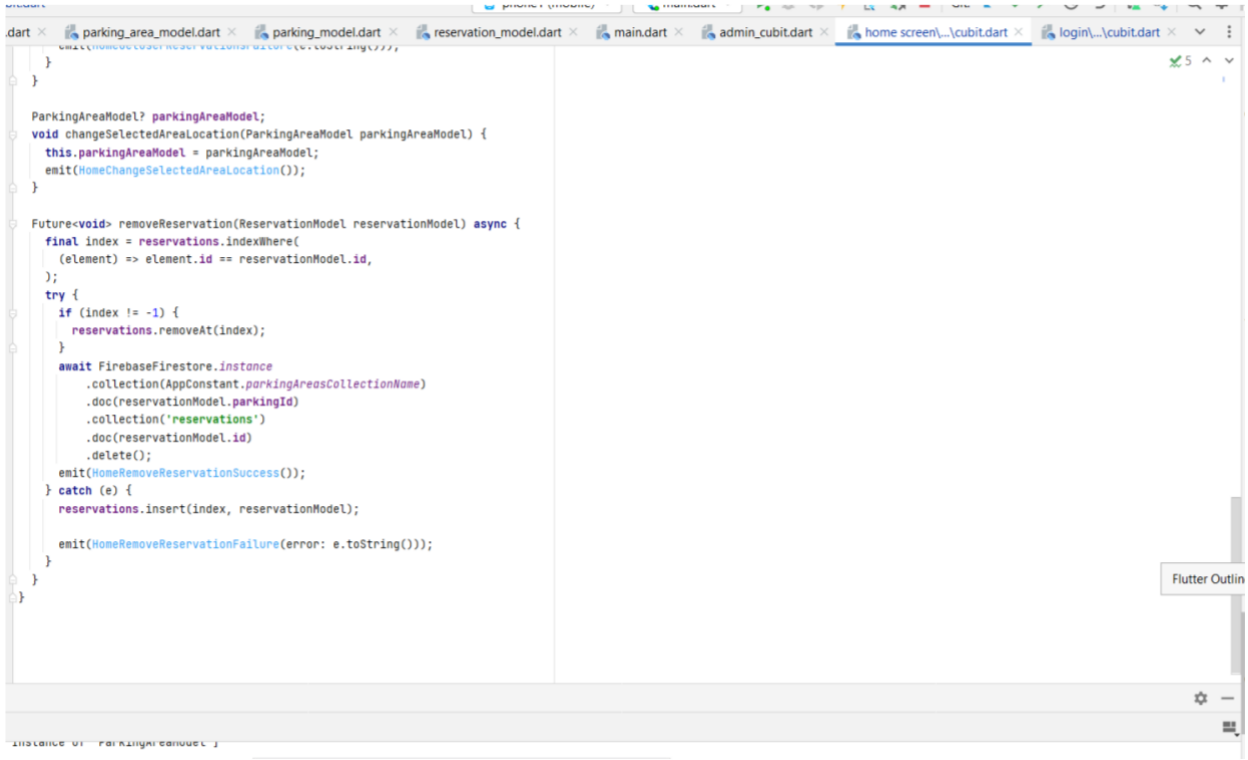


Figure 51 Login Page Code Snippet

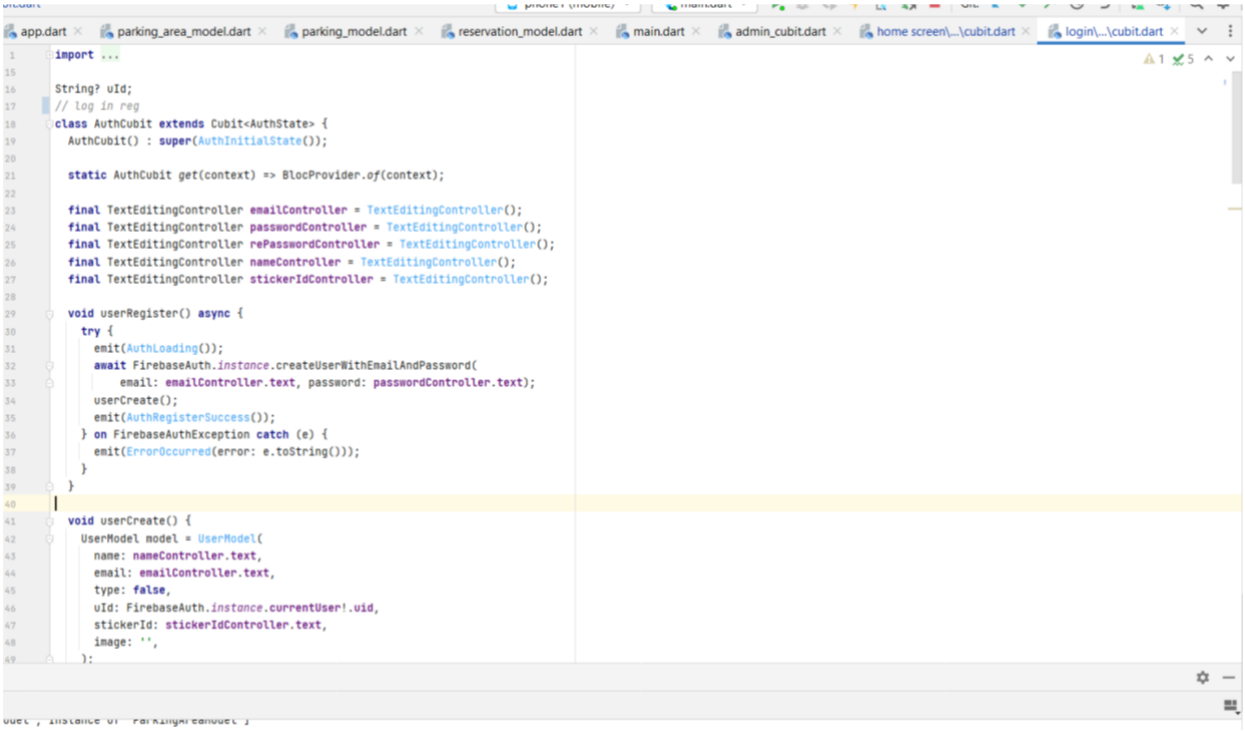


Figure 50 Home Page Code Snippet -2

```
AdminCubit() : super(AdminInitial());

static AdminCubit get(BuildContext context) => BlocProvider.of(context);

Future<void> upsertParkingArea(ParkingAreaModel parkingAreaModel) async {
  emit(AdminUpsertParkingAreaLoading());
  try {
    await FirebaseFirestore.instance
        .collection(AppConstant.parkingAreasCollectionName)
        .doc(parkingAreaModel.id)
        .set(parkingAreaModel.toJson());
    areas.map((area) {
      if (area.id == parkingAreaModel.id) {
        return parkingAreaModel;
      }
    });
    emit(AdminUpsertParkingAreaSuccess());
  } catch (e) {
    emit(AdminUpsertParkingAreaFailure(e.toString()));
  }
}

List<ParkingAreaModel> areas = [];
Map<String, List<int>> parkingAreas = {};
Map<String, Map<int, UserModel>> parkingAreasUsers = {};
Future<void> getParkingAreas() async {
  areas = [];
  parkingAreas = {};
  parkingAreasUsers = {};
  emit(AdminGetParkingAreasLoading());
  try {
```

Figure 53 Admin page code Snippet

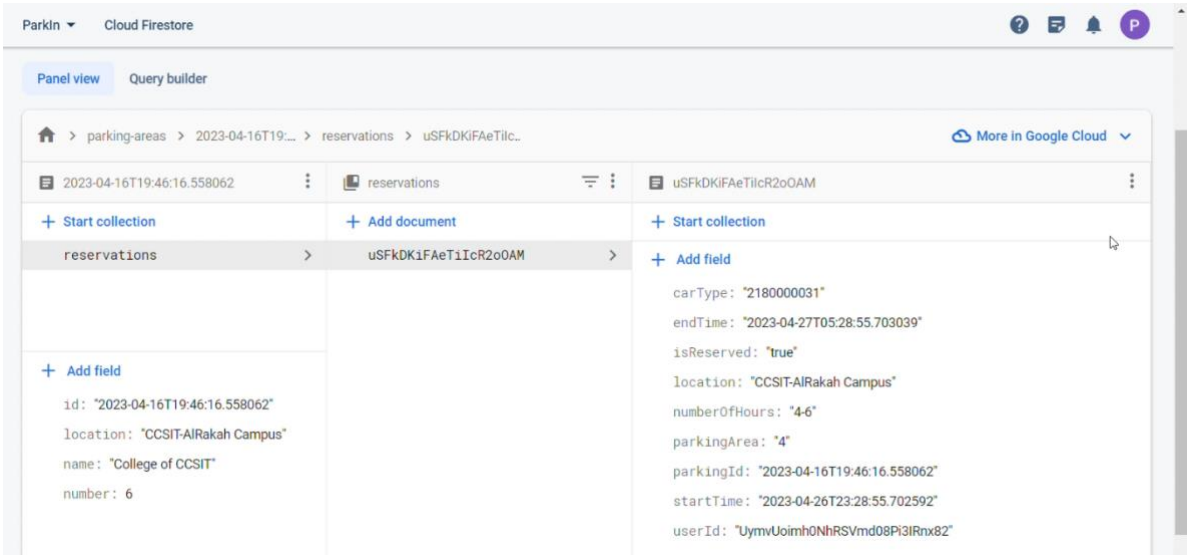


Figure 52 Realtime database

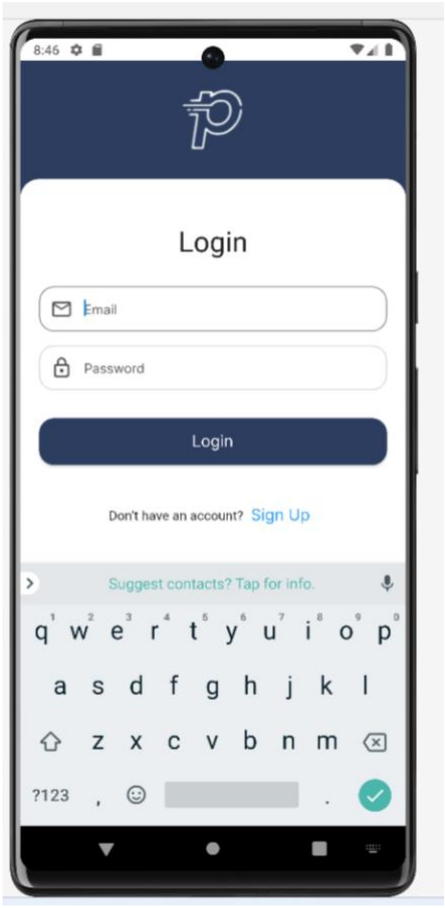


Figure 55 Login Interface of Parking App

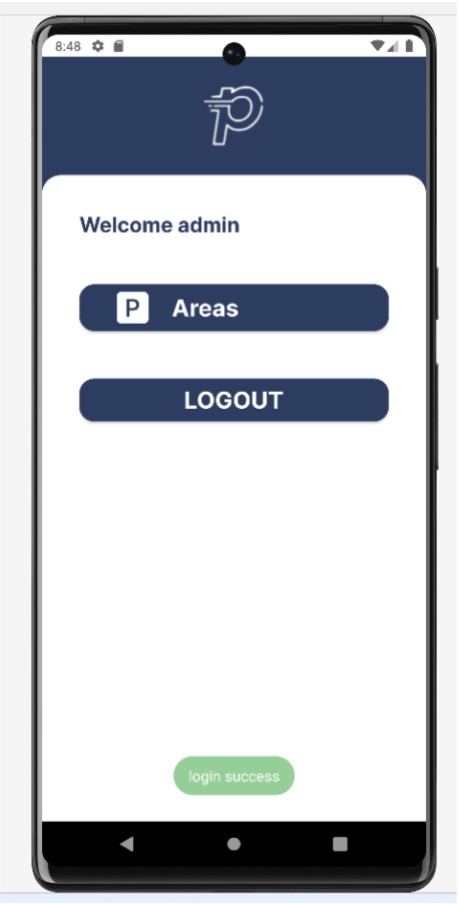


Figure 54 Admin Interface of Parking App

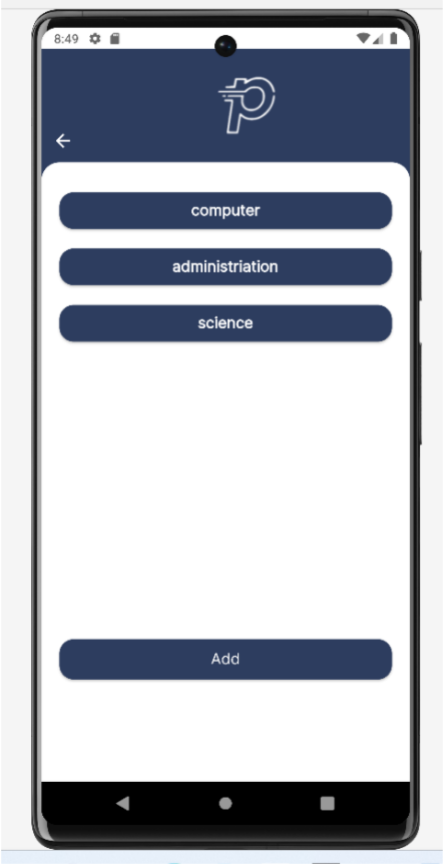


Figure 57 Admin Add Interface of Parking App

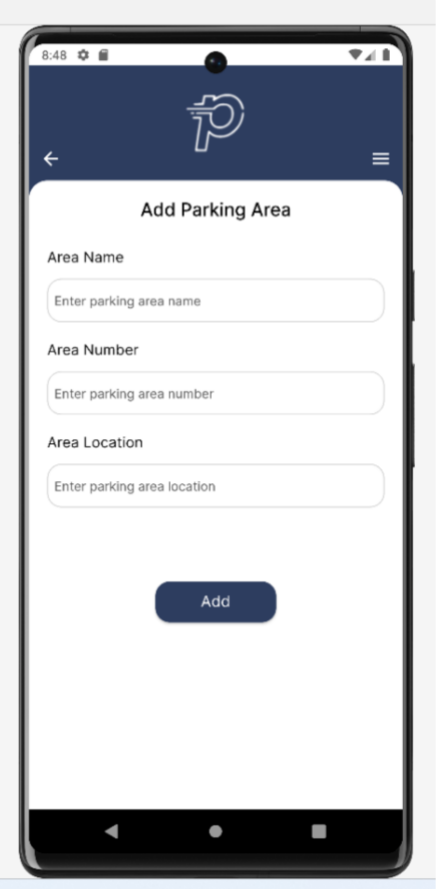


Figure 56 Add Parking Area Interface of Parking App

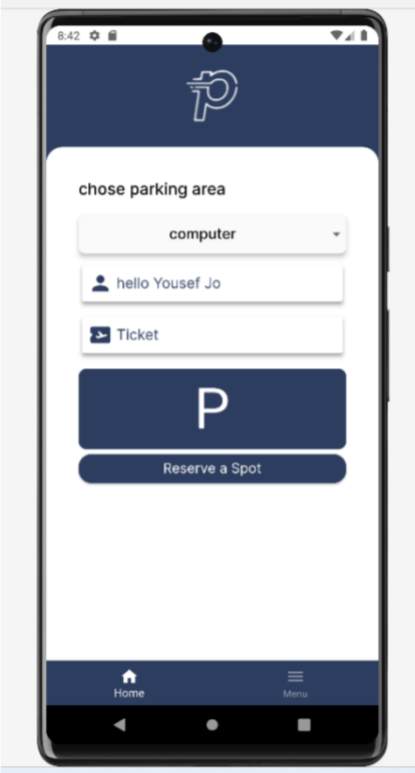


Figure 59 Home Interface of Parking App

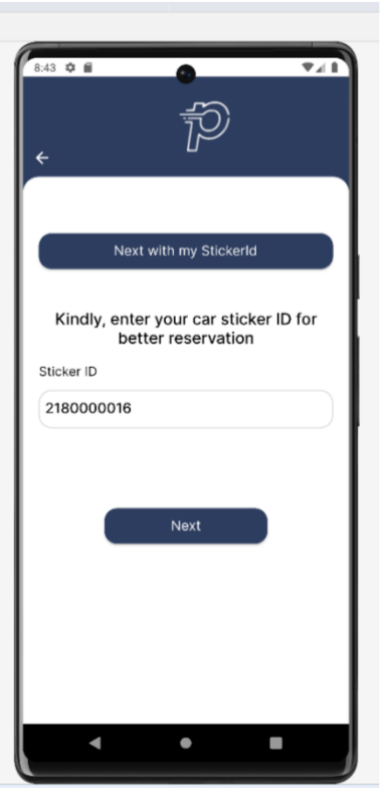


Figure 58 Add Sticker ID Interface of Parking App

Figure 61 Select a Parking Area Interface of Parking App

Figure 60 Select Number of Hours Interface of Parking App

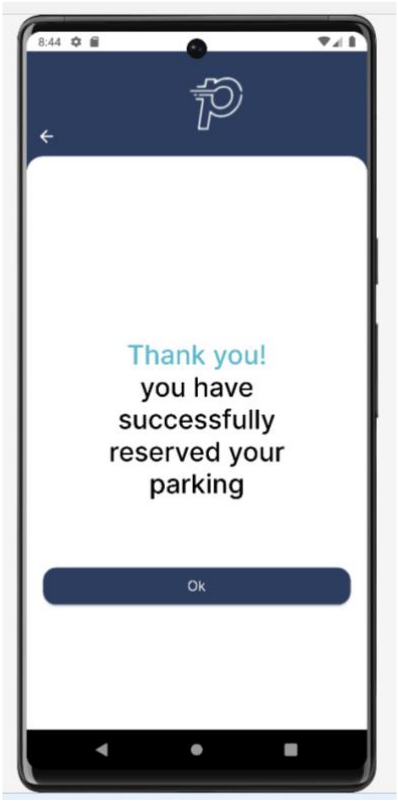


Figure 62 Success Message Interface of Parking App

Figure 63 Ticket Interface of Parking App

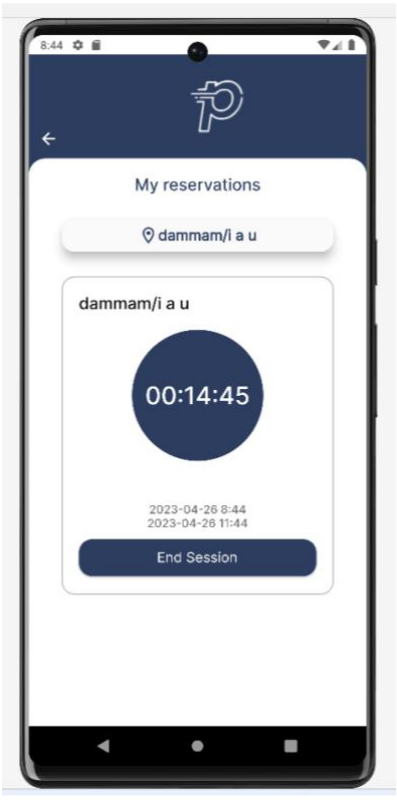


Figure 64 Timer Interface of Parking App

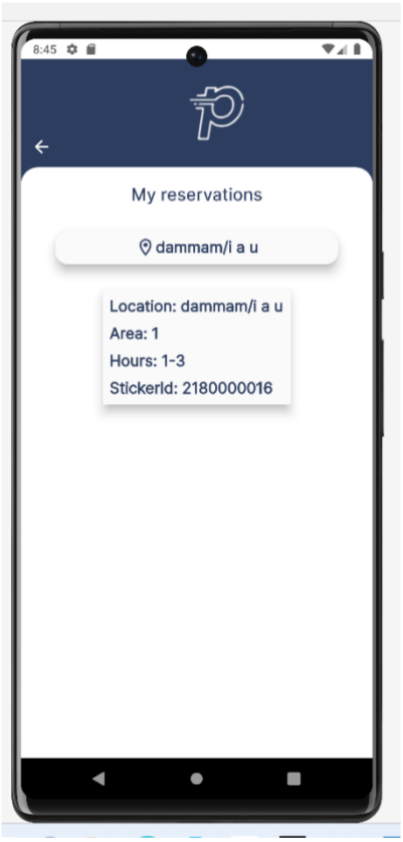


Figure 65 My Reservation Interface of Parking App

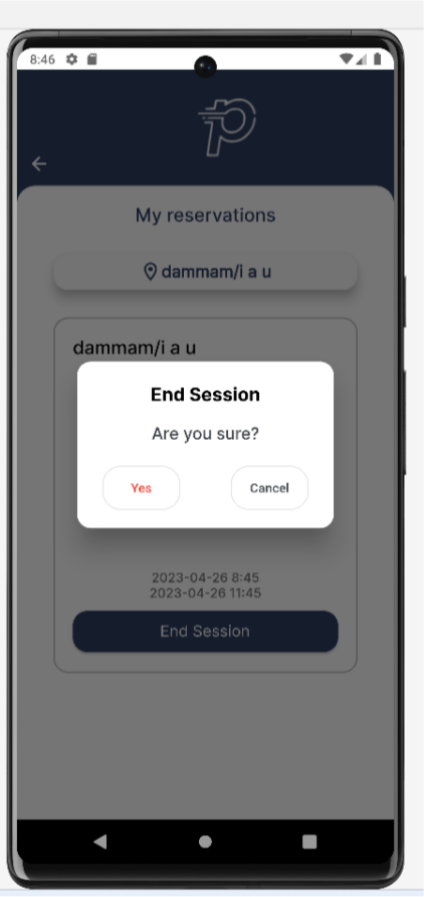


Figure 67 End Session Message of Parking App

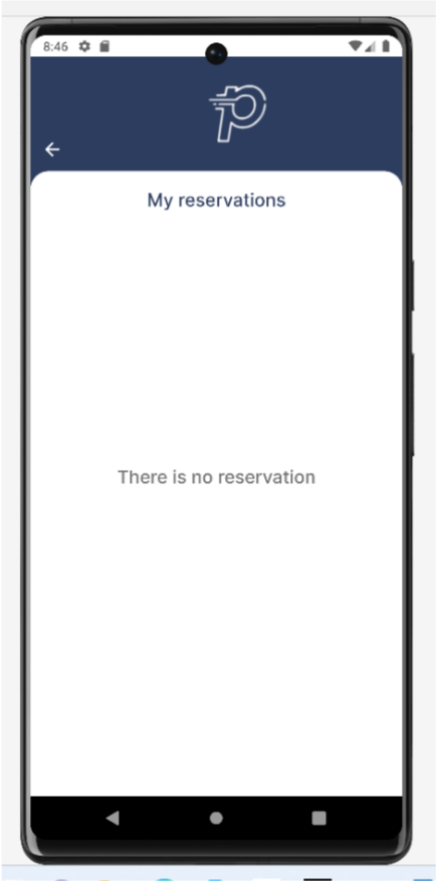


Figure 66 No Reservation Interface of Parking App

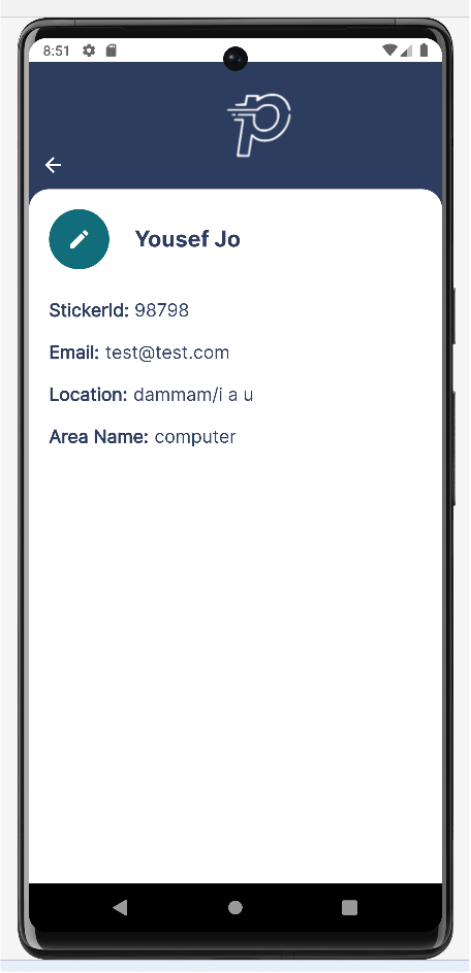


Figure 68 User Info Interface of Parking App

Hardware part:

```
CameraWebServer § | app_httpd.cpp | camera_index.h | camera_pins.h
//#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
//#define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
#include "camera_pins.h"

// ===========================
// Enter your WiFi credentials
// ===========================
const char* ssid = "**********";
const char* password = "**********";

void startCameraServer();
void setupLedFlash(int pin);

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
```

Figure 69 Arduino IDE Code for ESP32 CAM

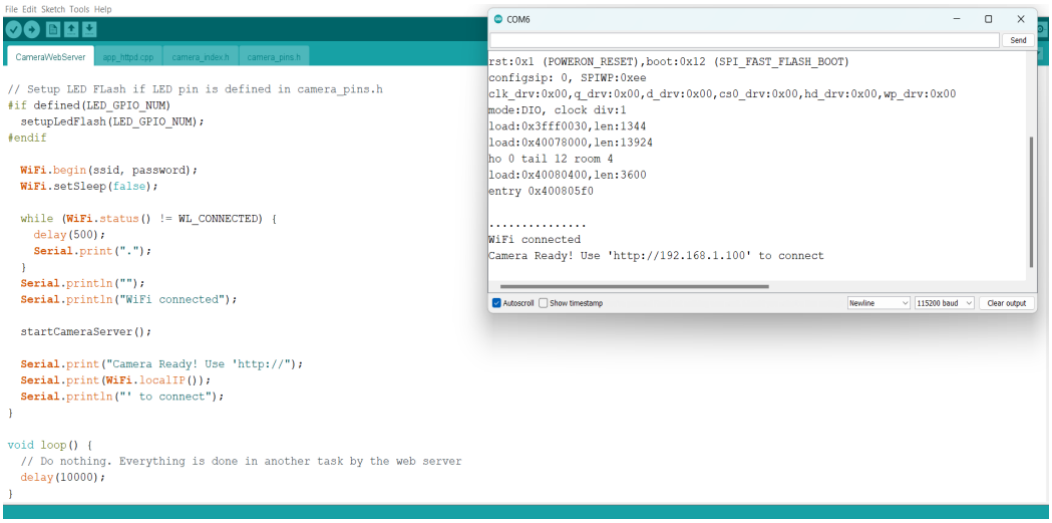


Figure 70 Arduino IDE Code for ESP32 CAM

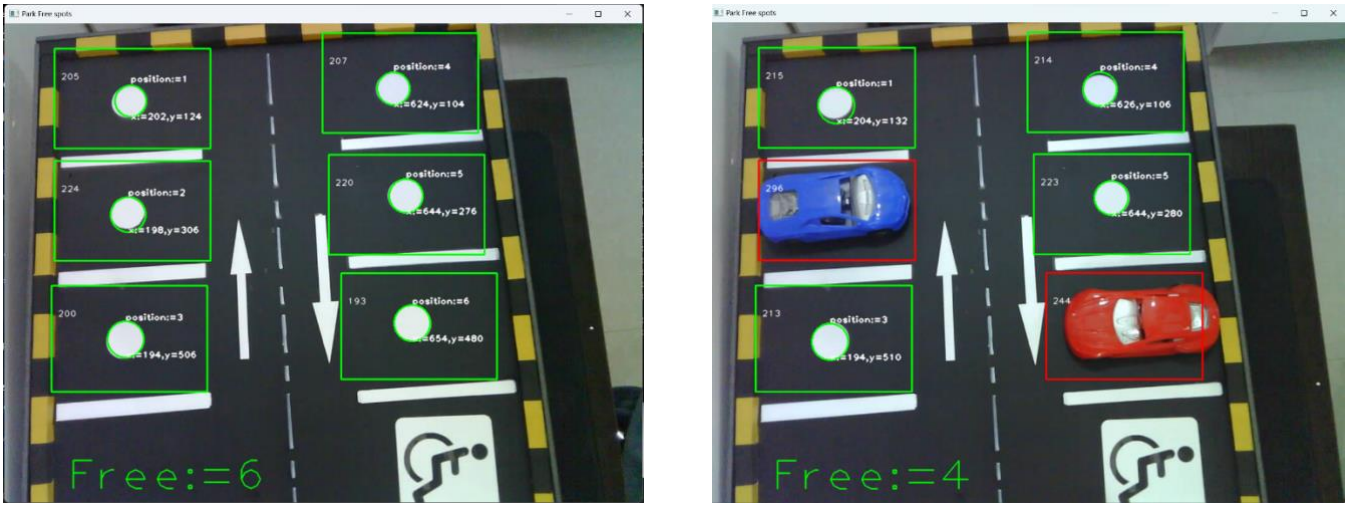


Figure 71 (A + B) show frame of parking

change.log custCode_Ver1.py

```python
import cv2
import numpy as np
import fireStoreHandler as fs
import controlEsp32Cam as espcam
import argparse

def handleArgs():
    global URL
    global IP
    parser = argparse.ArgumentParser(description="Free parking spot identifier using image processing",
                                     formatter_class=argparse.ArgumentDefaultsHelpFormatter)
    parser.add_argument("-c", "--cameraIP", action="store", help="camera module ip address")

    args = parser.parse_args()
    config = vars(args)

    #if the args has provided
    if config['cameraIP'] is not None:
        IP = "http://"+config['cameraIP']
        URL = 'http://' + config['cameraIP']+ ":81/stream"
    else:
        URL="http://192.168.1.3:81/stream" #if not args provided stay default
def setup():
    global cap
    # init firebase
    fs.init()

    # set resolution of the cam
    espcam.set_resolution(IP, index=10)
    # Face recognition and opencv setup
    try:
        cap = cv2.VideoCapture(URL)
    except Exception as e:
        print(f"Cannot find the camera feed:{e}")

```

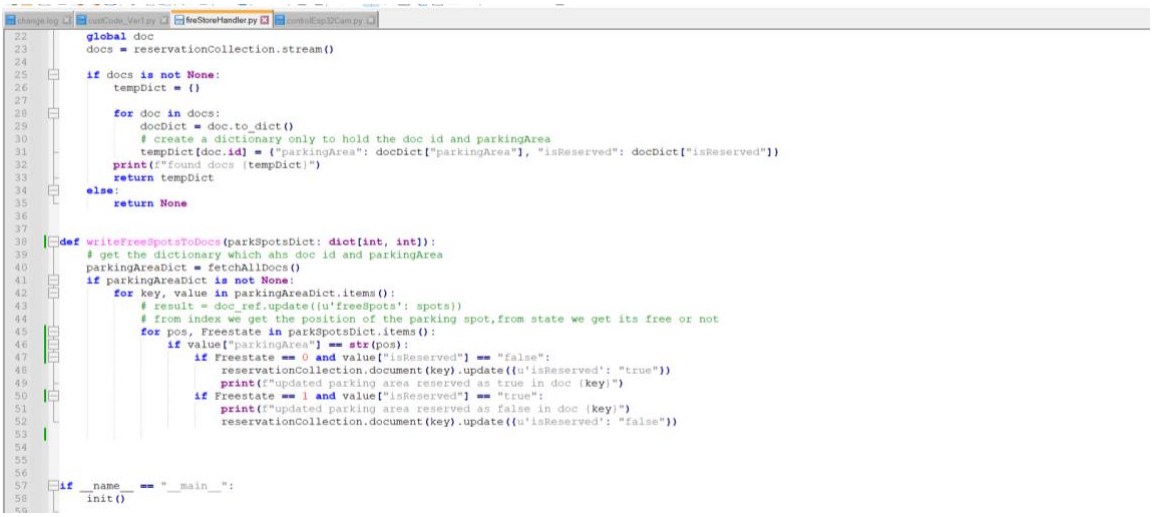
Figure 72 Main Hardware Code Snippet

```python
def loop():
    global cap

    # x,y,width,height of the 6 rectangles
    rectanglesDict = {1: (80, 90, 250, 120),  # position 1
                      2: (80, 250, 250, 120),  # position 2
                      3: (75, 420, 250, 160),  # position 3
                      4: (500, 65, 250, 120),  # position 4
                      5: (500, 200, 250, 120),  # position 5
                       6: (500, 400, 250, 160))  # position 6
    # after counting 100 times the average of the parking spots we send that to firebase

    count = 0
    total = {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0}

    while True:

        if cap.isOpened():
            ret, frame = cap.read()

            if ret:
                # Convert the image to grayscale
                gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

                # blurr image to remove noise
                blured = cv2.medianBlur(gray, 5)

                for (x, y, w, h) in rectanglesDict.values():
                    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2) #blue,green,red
                    roi = blured[y:y + h, x:x + w]
                    avg_color = np.asarray(cv2.mean(roi))

                    cv2.putText(frame, f"{round(np.sum(avg_color))}", (x + 10, y + 50), cv2.FONT_HERSHEY_PLAIN, 1,
                                (255, 255, 255), 1, cv2.LINE_AA)

                    # Apply Hough circle detection to find circles in the image
                    circles = cv2.HoughCircles(blured, cv2.HOUGH_GRADIENT, 1, 10, param1=10, param2=34, minRadius=15,
                                               maxRadius=30)
```

Figure 73 Main Hardware Code Snippet 2

```python
import requests

def set_resolution(url: str, index: int=1, verbose: bool=False):
    try:
        if verbose:
            resolutions = "10: UXGA(1600x1200)\n9: SXGA(1280x1024)\n8: XGA(1024x768)\n7: SVGA(800x600)\n6: VGA(640x480)\n5: CIF(400x296)\n4: QVGA(320x240)\n3: HQVGA
            print("available resolutions\n{}".format(resolutions))

        if index in [10, 9, 8, 7, 6, 5, 4, 3, 0]:
            requests.get(url + "/control?var=framesize&val={}".format(index))
        else:
            print("Wrong index")
    except:
        print("SET_RESOLUTION: something went wrong")

def set_quality(url: str, value: int=1, verbose: bool=False):
    try:
        if value >= 10 and value <=63:
            requests.get(url + "/control?var=quality&val={}".format(value))
    except:
        print("SET_QUALITY: something went wrong")

def set_awb(url: str, awb: int=1):
    try:
        awb = not awb
        requests.get(url + "/control?var=awb&val={}".format(1 if awb else 0))
    except:
        print("SET_QUALITY: something went wrong")
    return awb
```

Figure 74 Control ESP32CAM



```python
    global doc
    docs = reservationCollection.stream()

    if docs is not None:
        tempDict = {}

        for doc in docs:
            docDict = doc.to_dict()
            # create a dictionary only to hold the doc id and parkingArea
            tempDict[doc.id] = {"parkingArea": docDict["parkingArea"], "isReserved": docDict["isReserved"]}
        print(f"found docs {tempDict}")
        return tempDict
    else:
        return None

def writeFreeSpotsToDocs(parkSpotsDict: dict[int, int]):
    # get the dictionary which ahs doc id and parkingArea
    parkingAreaDict = fetchAllDocs()
    if parkingAreaDict is not None:
        for key, value in parkingAreaDict.items():
            # result = doc_ref.update({u'freeSpots': spots})
            # from index we get the position of the parking spot,from state we get its free or not
            for pos, Freestate in parkSpotsDict.items():
                if value["parkingArea"] == str(pos):
                    if Freestate == 0 and value["isReserved"] == "false":
                        reservationCollection.document(key).update({u'isReserved': "true"})
                        print(f"updated parking area reserved as true in doc {key}")
                    if Freestate == 1 and value["isReserved"] == "true":
                        print(f"updated parking area reserved as false in doc {key}")
                        reservationCollection.document(key).update({u'isReserved': "false"})


if __name__ == "__main__":
    init()
```

Figure 75 writing and reading to firebase.

5.4.1 Implementation environment

The software, programming languages, and platforms used during the implementation phase are described in table No.53 below.

Table 50 Implementation Environment

Logo	Name	Description
ARDUINO *Figure 76 Arduino IDE*	Arduino IDE	The software includes a code editor that links with Arduino equipment to transfer programs and communicate with them.

 *Figure 77 Android Studio*	Android Studio	An official integrated development environment for the Android operating system from Google, built on the IntelliJ IDEA software from JetBrains.
 *Figure 78 Python*	Python	a general-purpose, high-level programming language that improves the readability of the code (Python, n.d.).
 *Figure 79 Notepad++*	Notepad++	Open-source editor for text and source code that works on Microsoft Windows. It has the capability to use tabs for editing, enabling users to work with several open files within a single window.
 *Figure 80 Flutter Platform*	Flutter	A Google open-source UI software development kit. It is used to create cross-platform software from a single codebase for Google Fuchsia, Android, iOS, Linux, macOS, Windows, and the web.
 *Figure 81 Firebase NoSQL*	Firebase NoSQL	A NoSQL database stored in the cloud that enables real-time data synchronization between users.

5.4.2 Tasks progress during the implementation phase

Each task created since the start of this semester is listed below in table No.54 with its ID, description, and status.

Table 51 Task Progress

Task ID	Task Description	Task Status
General Functions		
1	Create the connection between the Firebase database and Android Studio.	Completed.
2	Create a “Homepage” interface to be available for all (i.e., users, admin)	Completed.

3	Provide reset forgotten password service.	Completed.
4	Create and design a “Sign in” interface and build a form containing two labels (username and password) besides able the password field to be hidden not visible.	Completed.
5	Create and design a “Sign up” interface and build its form.	Completed.
6	Build the connection between the database and “Sign in / Sign up” interfaces.	Completed.
7	Create and design “Sign out” button.	Completed.
8	Create and design a “Find nearest parking areas” interface, build its form with all necessary labels, and merge Google Maps within the interface.	Completed.
Hardware Functions		
9	Create the connection between ESP32-CAM and Arduino IDE.	Completed.
10	Import OpenCV library in Arduino IDE Successfully.	Completed.
11	Establish the connection between ESP33 cam.	Completed.
User Functions		
12	Create and design an “Reserve Parking Zone” interface, build its form with all necessary labels, and merge Google Maps within the interface.	Completed.
13	Create “Vehicle” to add new vehicle and view/update/delete existing vehicles.	Completed.
14	Create “My Reservations” to view current and previous reservations.	Completed.
15	Create and design ‘Edit profile’ and build its form.	Completed.
16	Create and design ‘Parking zone’ interface and build its form.	Completed.
Admin Functions		
17	Create and design admin’s dashboard.	Completed.

18	Create and design (Add/Edit/Delete) vehicles interface to enable admin for modifying	Completed.
19	Create and design (Add/Edit/Delete) user interface to enable admin for modifying	Completed.

5.6 Issues faced:

The following issues were highlighted by team members during the implementation phase:

1. learn a new programming language in a short amount of time (for example, Python).
2. Some problems were encountered while installing flutter and binding it within Visual Studio Code.
3. Time management because of scheduling disputes among team members.
4. It was nearly impossible to communicate and discuss verbally, much less even combine the work done by each team member.

5.7 ENVIRONMENTAL REQUIREMENTS

In this section will address the environmental requirements including the Hardware, Software,

Security, tools, publication and risk and assumption that are related to the project and system.

5.7. 1 Hardware

The needed hardware and network requirement to successfully perform the testing procedure effectively are:

- Laptops
- Cameras (ESP32 CAM)
- High speed Wi-Fi connection
- Smartphone.

5.7.2 Software

The needed software requirements for successfully preforming the testing procedure effectively are:

- Windows operating system with a suitable RAM space.
- Microsoft word for document preparation.
- Android Studio
- Firebase
- Notepad++

5.7.3 Security

The system will be secured by:

- Login authentication: All type of users' needs to be authorized.
- Parking authentication: the parking gate will remain closed until there's a QR code scanned, the QR code will be expired after 15 minutes.
- Database encryption: passwords on firebase database will be encrypted.

5.7.4 Tools

The tools requirement that are needed for preforming the test procedure efficiently are:

- Android Debug Bridge
- Android system
- Microsoft Word for document creation throughout the testing phase.
- Firebase

5.7.5 Publications

The publication requirement that are needed for preforming the test procedure efficiently are using IEEE test plan standard therefore, the references are:

- IEEE Recommended Practice Software Requirements Specification (SRS)
- Software Design Specifications (SDS)

5.7. 6 Risks and Assumptions

Table No. 55 below describes the possible risks that might occur in the testing process and how the team members can prevent it.

Table 52 risks and assumptions

Risk	Risk Possibility	Risk impact in the project	Description	Prevention
Item availability	Medium	High	If the required items are not available during testing. This can either cause the testing process to be delayed or result in test failure.	Before moving forward with the testing process, the team must ensure that all the required items are available.
Resource availability	High	High	• A lack of resources might cause a delay in the testing process, or the team might be unable to perform the	• Team members should possess the appropriate knowledge in order to implement

			testing process efficiently. • The testing process can also be adversely affected if team members do not have the appropriate knowledge.	the system or consult experts for guidance.
Time constrains	Medium	Medium	It is possible for unexpected circumstances to occur, such as a sudden absence from the team or a sudden change in the schedule and deadlines, which can cause the delivery and testing process to be delayed.	For the project to be completed on time, the team members have to cooperate efficiently in order to meet the deadline.
No control over project priorities	Medium	Medium	Lack of control over project priorities can lead to misalignment between the project goals and the resources used to achieve them. Consequently, there may be cost overruns, delays, and missed deadlines	• ensure that the project plan is well-defined and that all team members are aware of the project plan. • establish clear roles and responsibilities, so that all team members are aware of their part in the project. • Regular meetings and updates of the project plan should be conducted to meet the deadlines.
Budget control	Low	Medium	Ineffective budget management can result in a project going over budget. This risk can arise when there is not an effective system in place to monitor and track the budget for a project	• establish a budget monitoring system and that all team members are aware of the budget management. In order to keep the project within the budget

5.7.7 CHANGE MANAGEMENT PROCEDURES

The Change Management Processes: are a set of actions that help a change be implemented successfully.

Change Management Plans: Facilitate the process to implement a change. They are typically created during the planning stage.

If team members need to make changes or modifications to the test plan, they must follow several approval processes. As stated in the processes below:

Change Initiation: Determine the necessary changes, assess the impact of the changes, and document the plan.

Change Review: Allow team members to review and approve the change.

Change Authorization: Validate and authorize the change.

5.7.8 PLAN APPROVALS

Table 53 plan approvals

Name	Date	Signature
Ghada almuhaideb	27-12-2022	*Ghada almuhaideb*
Aisha khubrani	27-12-2022	*Aisha khubrani*
Abeer aldkheel	27-12-2022	*Abeer aldkheel*
Bashayer Alyami	27-12-2022	*Bashayer Alyami*
Ghadeer AlGhamdi	27-12-2022	*Ghadeer AlGhamdi*

5.8 Software Test Report

5.8.1 Document overview

This document provides the software test report for the mobile application software project's Parkin testing phase. It includes the results of all tests conducted during the testing phase. Parkin is an application that offers drivers with available parking spaces on the AlRakah campus and is suitable for novice users. Our objective is to save customers time and money by allowing them to reserve a parking spot through a single application and by directing them to the nearest parking lot.

5.8.2References

5.8.2.1 Project References

Table 54 Project References

#	Document Identifier	Document Title
[R1]	1	Parkin’s Software Development Plan
[R2]	2	Parkin’s IEEE Software Test Plan
[R3]	3	Test Cases Checklists

5.8.2.2 Standard and regulatory References

Table 55 Standard and regulatory References

#	Document Identifier	Document Title
[STD1]	1	IEEE Standard 1233 – 1998, IEEE Guide for Developing System Requirements Specifications

5.8.3 Overview of Tests Results

This section consists of four main parts, which are the tests log, the rationale for decision, overall assessment of tests, and impact of the test environment.

5.8.3.1 Tests log

The Parkin's Software (version 0.1) was tested on android studio test platform located in Eastern province in Khobar, from the 2023/01/28 to the 2023/02/10. The tests of the test phase (ref. software test plan) were executed.
Testers where:

- Ghadah Riyadh AlMuhaideb
- Abeer Saad AlDkheel
- Aisha Moraie Khubrani
- Bashayer Mohammed AlYami
- Ghadeer Saeed AlGhamdi

5.8.3.2 Rationale for decisio*n*

After executing a test, the decision is defined according to the following rules:

- **OK:** The test sheet is set to "OK" state when all steps are in "OK" state. The real result is compliant to the expected result.
- **NOK:** The test sheet is set to "NOK" state when all steps of the test are set to "NOK" state or when the result of a step differs from the expected result.
- **Partial OK:** The test sheet is set to "Partial OK" state when at least one step of the test is set to "NOK" state or when the result of a step is partially compliant to the expected result. Keep it or remove. Source of inconsistencies: criteria to set if result is Partial OK may be qualitative.
- **NOT RUN:** Default state of a test sheet not yet executed.
- **NOT COMPLETED:** The test sheet is set to "Not Completed" state when at least one step of the test is set "Not Run" state.

Tests results are listed in §3.

5.8.3.3 Overall assessment of tests

Qualitative overall assessment of tests: 100% tests passed with "OK".
The table below illustrates all the test cases with their names, testing category, and results.

Table 56 Overall assessment of tests

Test Case No.	Name	Category	Result
1	Log in.	White-box testing	OK
2	Register	White-box testing	OK
3	Forget password.	White-box testing	OK
4	User profile	White-box testing	OK
5	Search for parking.	White-box testing	OK
6	Reserve parking.	White-box testing	OK
7	Cancel parking.	White-box testing	OK
8	Search for parking	White-box testing	OK
9	Login	Black-box testing	OK
10	Register	Black-box testing	OK
11	Reset password	Black-box testing	OK
12	User Profile	Black-box testing	OK
13	Contact Us	Black-box testing	OK
14	Homepage	Black-box testing	OK
14.1	Search for parking	Black-box testing	OK
14.2	Reserve Parking	Black-box testing	OK
15	Cancel Reservation	Black-box testing	OK
16	Add/Update/Delete Areas	Black-box testing	OK
17	Update user's account information	Black-box testing	OK

5.8.4 Impact of test environment

The test environment, which acts as a setup for both the hardware and the software, had a favorable impact. Before implementing the system and interfacing the hardware with the program, the team members were able to avoid any unexpected errors and defects using the correct environment. The test environment is often constrained to a single platform. The system is made up of tangible parts that make it possible to construct and test it. In other words, it makes it possible to do tests utilizing simulators, hardware, and network settings, software test tools, etc.

The difference between the expected condition and real condition are describes in the table No.60 below:

Table 57 difference between the expected condition and real condition

Test Tool	Expected Condition	Real Condition
Software test tools	• Each interface works successfully. • the hardware and the software are connected successfully.	• Interfaces must function well and attain a high degree of usability, to be effectively used by individuals with varying levels of expertise.

	• Data are successfully stored and retrieved from the database. • Application must work on an android mobile phone.	• The data must be correct, free from error and complete before displaying it in the application or stored in the database. • Application must work on multiple vision of android.
Hardware test tools	• The ESP32-CAM needs to be able to detect the objective (parking space) correctly. • ESP32-CAM needs to send the data to the firebase successfully. • The servo motor ESP32-CAM needs to open and close the door based on the barcode reading, based on the condition, if the barcode is in the database or not.	• The Detecting of the object needs to be as accurate as possible. • ESP32-CAM has to send the data to the firebase successfully and update is continuously. • The servo motor ESP32-CAM needs to open and close the door based on the barcode reading, based on the condition, if the barcode is in the database or not, without a delay.
Database serve	• database server keeps essential users' information secure.	• database server keeps essential users' information secure and saves along with their historical data of adoption offers and requests details

5.8.5 Detailed Tests Results

For each executed test, this document contains:

- Test identification.
- Test title.
- Test decision.
- A comment containing additional information or problems encountered during execution and differences with the test procedure.

5.8.5.1 White-box Testing

Test Case 1: log in.

(a) **(b)**

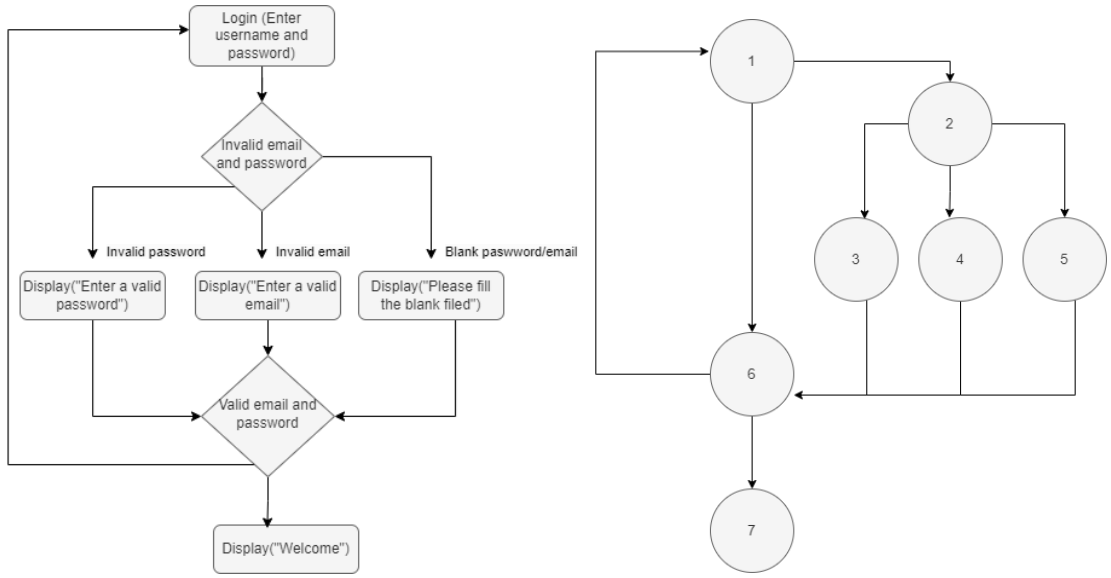


Figure *82* (a) Flowchart and (b) Flow graph of Test case 1

The Cyclomatic complexity:

Table *58* Cyclomatic complexity Test case 1

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)=10-7+2	V(G) = P+1 V(G)= 4+1
Result	V(G)=5	V(G)=5	V(G)=5

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table No.62 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

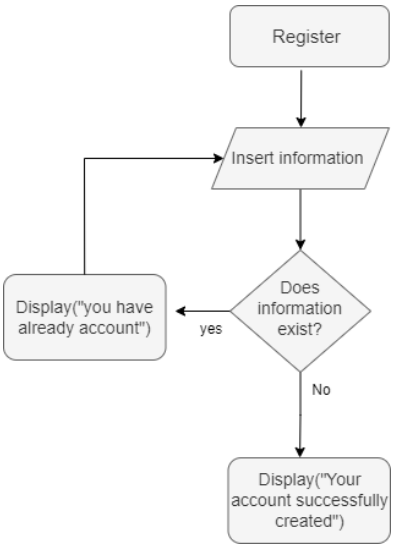
No. test= 2

Table *59* Full branch coverage Test case 1

No. 1	1, 2, 3, 4, 5, 6, ,7 Test case 1: (A=5, B=3, D=2)
No. 2	1,6,7 Test case 2: (A=2, B=5, D=8)

Test Case 2: Register

(a)



(b)

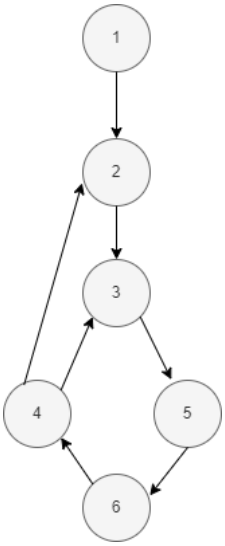


Figure 83 (a) Flowchart and (b) Flow graph of Test case 2

The Cyclomatic complexity:

Table 60 Cyclomatic complexity of Test case 2

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)= 7-6+2	V(G) = P+1 V(G)= 2+1
Result	V(G)=3	V(G)=3	V(G)=3

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table No.64 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 2

Table 61 Full branch coverage Test case 2

No. 1	1,2,3,5,6
No.2	1,2,3,4,6

Test Case 3: forget password.

(a) (b)

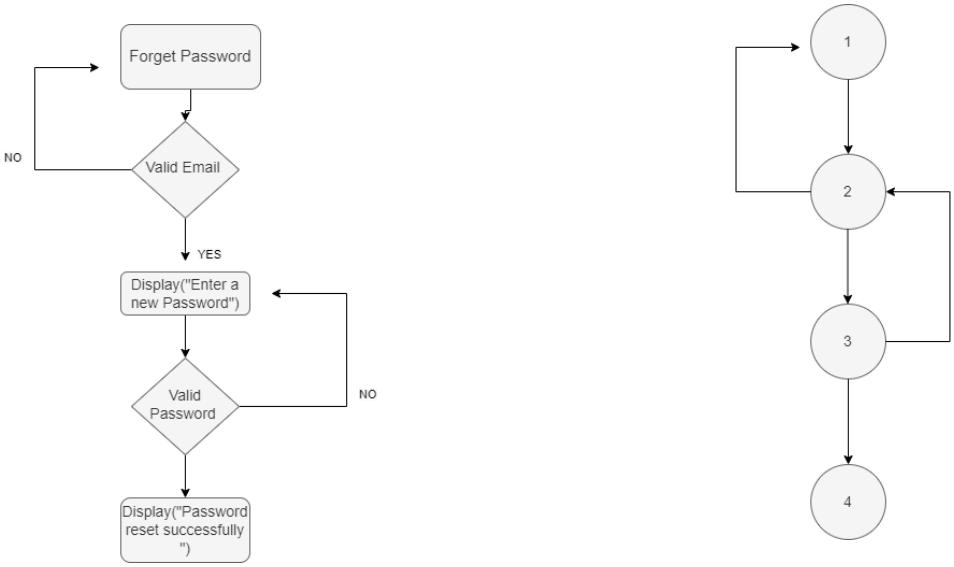


Figure 84 (a) Flowchart and (b) Flow graph of Test case 3

The Cyclomatic complexity:

Table 62 Cyclomatic complexity of test case 3

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)= 5 – 4 + 2	V(G) = P+1 V(G)= 2+1
Result	V(G)=3	V(G)=3	V(G)=3

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table No.66 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 1

Table 63 Full branch coverage Test case 3

No. 1	1, 2, 3, 4

Test Case 4: user profile (edit profile)

(a) (b)

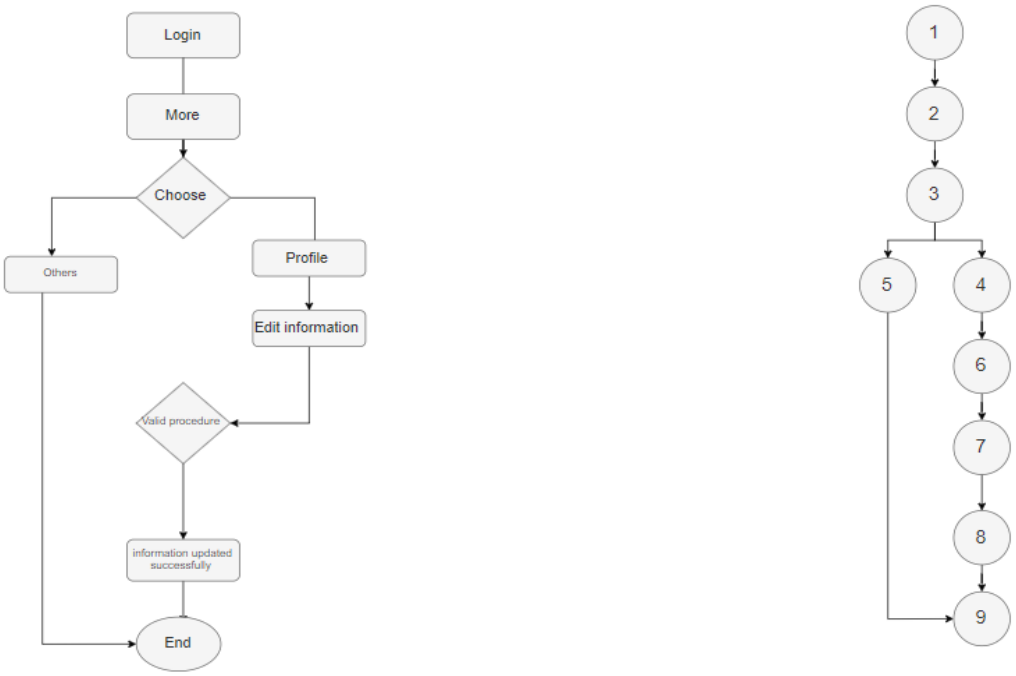


Figure 85 (a) Flowchart and (b) Flow graph of Test case 4

The Cyclomatic complexity:

Table 64 Cyclomatic complexity of test case 4

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)= 9-9+2	V(G) = P+1 V(G)= 1+1
Result	V(G)=2	V(G)=2	V(G)=2

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The tableNo.68 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 3

Table 65 full branch coverage of test case 4

No. 1	1,2,3,4,6,7,8,9
No.2	1,2,3,5,9

Test Case 5: Home page (Reserve parking)

(a) (b)

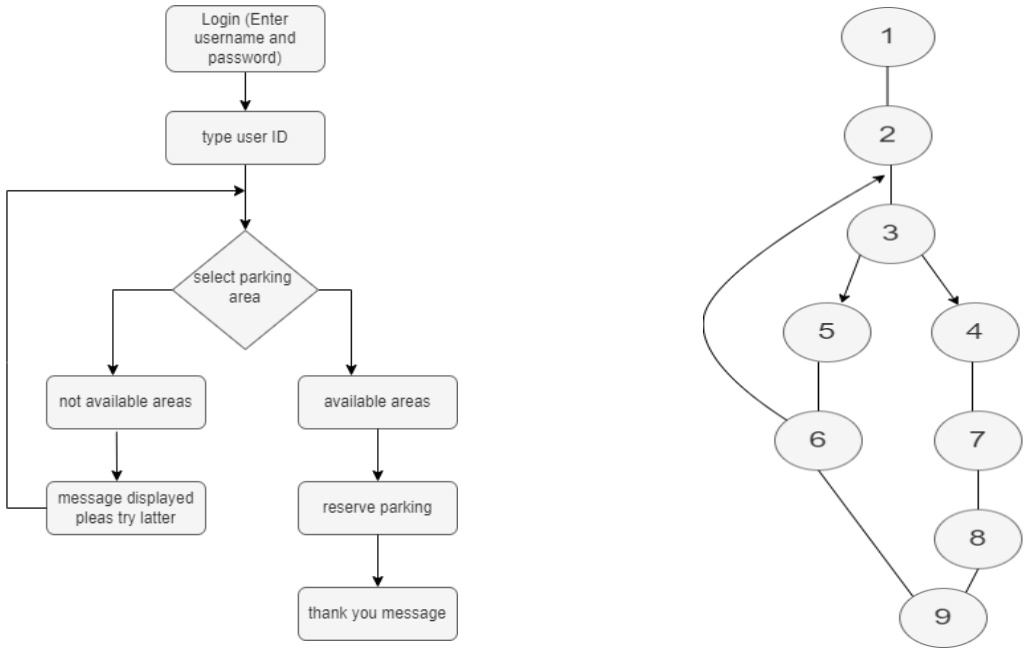


Figure 86 (a) Flowchart and (b) Flow graph of Test case 5

The Cyclomatic complexity:

Table 66 Cyclomatic complexity of test case 5

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)= 10-9+2	V(G) = P+1 V(G)= 2+1
Result	V(G)=3	V(G)=3	V(G)=3

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table No.70 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 3

Table 67 full branch coverage of test case 5

No. 1	1,2,3,4,7,8,9
No.2	1,2,3,5.6,9
No.3	1,2,3,5.6,2...9

Test Case 6: cancel parking.

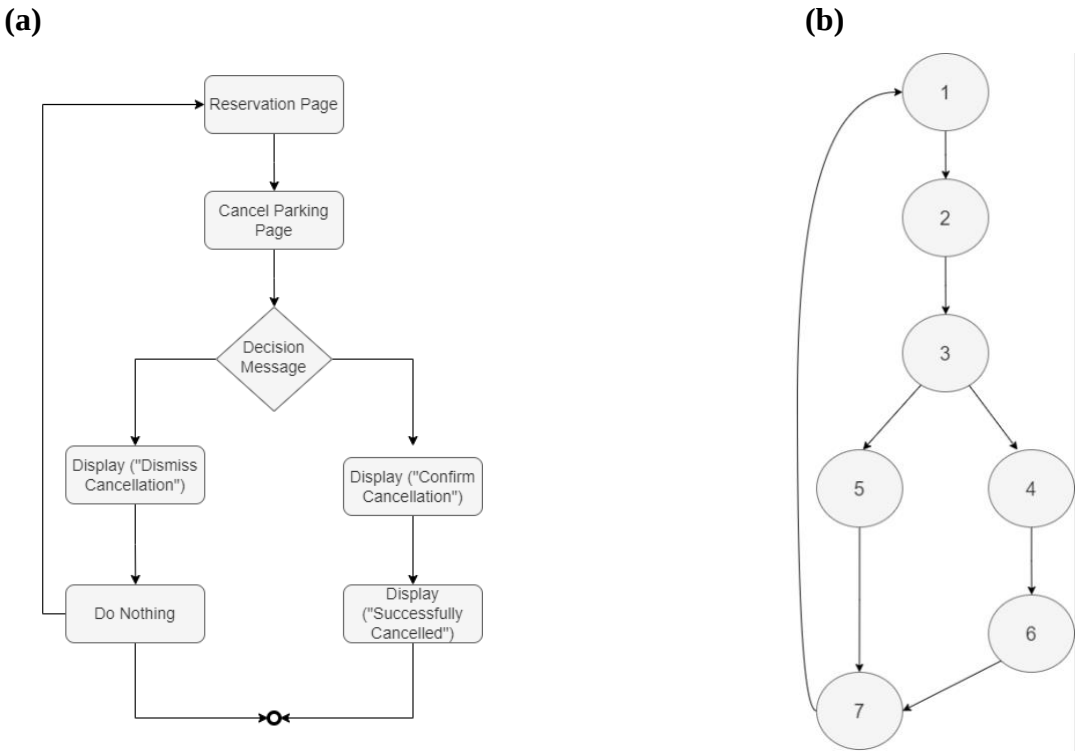


Figure87 (a) Flowchart and (b) Flow graph of Test case 6

The Cyclomatic complexity:

Table 68 Cyclomatic complexity of test case 6

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)=8-7+2	V(G) = P+1 V(G)= 2+1
Result	V(G)=3	V(G)=3	V(G)=3

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table No.72 below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 7

Table 69 full branch coverage of test case 6

No. 1	1, 2, 3, 4, 6, 7 Test case 1: (A=5, B=3, D=2)
No. 2	1, 2, 3, 5, 7

	Test case 2: (A=5, B=3, D=3)

Test Case 7: Add/Update/Delete Parking.

(a) **(b)**

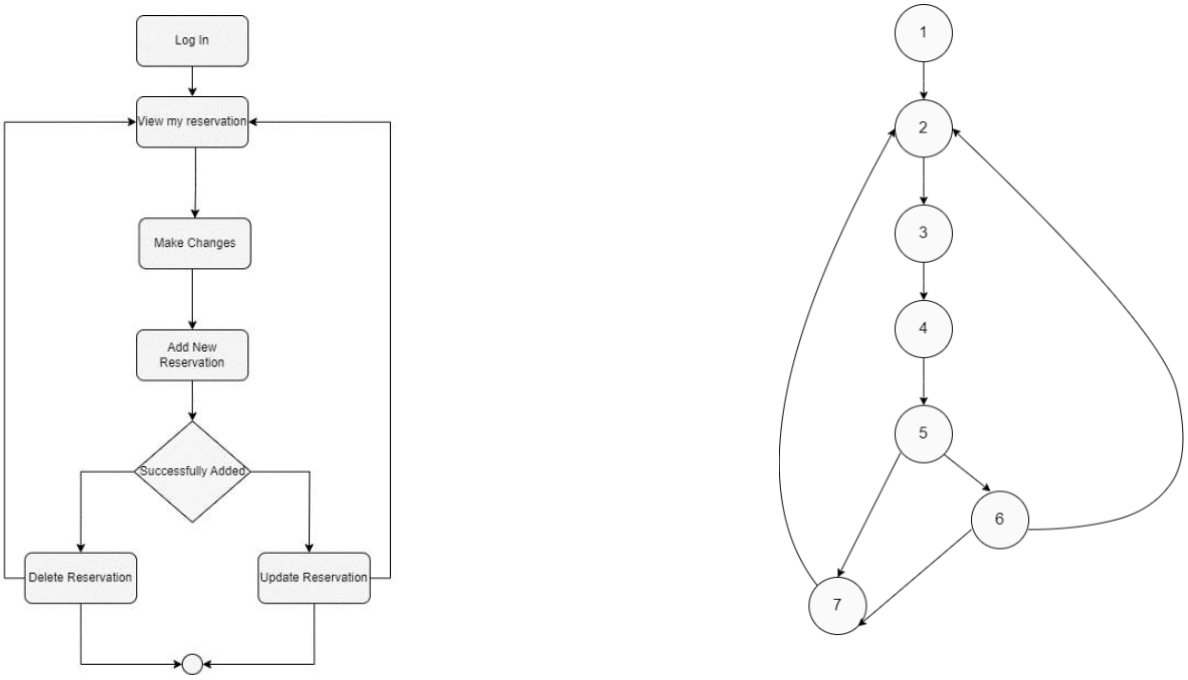


Figure 88 (a) Flowchart and (b) Flow graph of Test case 7

The Cyclomatic complexity:

Table 70 Cyclomatic complexity of test case 7

	Method 1	**Method 2**	**Method 3**
Formula	V(G)=R	V(G)=E-N+2 V(G)= 2+2	V(G) = P+1 V(G)= 3+1
Result	V(G)= 4	V(G)= 4	V(G)= 4

(R) = Number of regions (N) = Number of Nodes (E) = Number of edges (P) = Number of predicate nodes

The table below shows the testing methodology that was used for white box, and it was the branch coverage. All possible branches have been tested at least once; the coverage target for this testing is 100%.

The full branch coverage:

No. test= 8

Table 71 full branch coverage of test case 7

No. 1	1, 2, 3, 4, 5, 6, Test case 1: (A=6, B=4, D=2)
No. 2	1, 2, 3, 4, 5, 7 Test case 2: (A=6, B=4, D=3)

5.8.5.2 Black-box Testing

Test Case 8: Login Test

Test ID	**T9**	**Comment**	**Decision**
Test description	This test case goal is to test the login functionality in Parkin.	Application login successfully	OK
Verified Requirement	Email Password	Email = email Password = password	
Initial conditions	1. Fill All require fields. 2. Credentials must be registered to gain access.	Users should enter a valid email address. Password should include 8 characters containing uppercase and number.	
Tests inputs	Username Password		
Data collection actions	Parkin's will verify the user's credentials and cross check with the database		
Tests outputs	Redirect to the homepage		
Assumptions and constraints	Only registered users will gain access to the application.		
Expected results and criteria	1. Users enter their credentials. 2. Connection established. 3. User log in		
Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	Start filling the required fields	Check the user's input	OK
2	Clicks login button	Database checked for the user	OK
3	Directed to the homepage	User gain access	OK
Test Condition 1			
4	Repeat steps 1-2 and keep the email filed empty	The system will display an error message ("email address field is required")	Ok
Test Condition 2			
6	Repeat steps 1-2 and keep the password filed empty	The system will display an error message ("password field is required")	Ok

Test Condition 3			
7	Repeat steps 1-2 and entering wrong password	The system will display an error message (“email or password is incorrect”)	Ok
Test Condition 4			
8	Repeat steps 1-2 and entering wrong email	The system will display an error message (“email or password is incorrect”)	Ok

Test Case 9: Register Test

Test ID	T10	Comment	Decision
Test description	This test case goal is to test the Register functionality in Parkin.	User created successfully	OK
Verified Requirement	Username Password Email	Username = Stringf Password = Password Email = Email	
Initial conditions	1- Fill All require fields. 2- All input should be correct.		
Tests inputs	Username Password Email	All fields must be entered in the correct format.	
Data collection actions	Parkin will check the user's credentials and do a database cross-check.		
Tests outputs	The user will be registered in the system.		
Assumptions and constraints	1. Registered users cannot be registered again. 2. Administrators can't sign up		
Expected results and criteria	1. User enter their credentials. 2. Connection established. 3. User is logged in.		

Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	Start filling the required fields	Checks the user's input	OK
2	Clicks register button	Checks database for user	OK
3	Directed to the login page	Waits for the user's input	OK
4	Start filling the required fields	Checks the user's input	OK
5	Directed to the homepage	Gives the user access	OK
Test Condition 1			
7	Repeat steps 1-4 and keep the first name field empty	The system will display an error message "Please fill out the form!"	OK
Test Condition 2			
8	Repeat steps 1-2 with wrong email address	The system will display an error message "Invalid email address"	OK

Test Case 10: Reset password feature.

Test ID	**T11**	**Comment**	**Decision**
Test description	This test case goal is to test the forgotten passwords functionality in Parkin. How user can reset a new password		OK
Verified Requirement	Email		
Initial conditions	The user must have an account.		
Tests inputs	Email: "Abeer@gmail.com"		
Data collection actions	----------		
Tests outputs	Password reset successfully		
Assumptions and constraints	The user must have an account		
Expected results and criteria	An email will be sent to the provided email with a link that redirects the user to the reset password.		

Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	Click on log in		OK
2	Users click on the “forget password”	The system will redirect the user to the forget password interface	OK
3	Users enter email address	The system verifies the entered email address against those that are already in the database.	
4	Users click on confirm	The given email password will be used to send a link, and a message reading "Please check your email and click on the link to reset your password" will be displayed.	OK
5	Users receive the rest password link in the email	The user receive email to reset the passwords	OK
6	Users click on link and enter a new password	The system will record the new password on the database.	OK
Test Condition			
7	Repeat steps 1-3 with wrong email address	The system will display an error message “Invalid email address”	OK

Test Case 11: User Profile Test (edit profile)

Test ID	**T12**	**Comment**	**Decision**
Test description	User Profile Test		OK
Verified Requirement	Username Password Email	Username = Stringf Password = Password Email = Email	
Initial conditions	-the user must be registered. (have an account)		
Tests inputs	- Click on More - choose Profile then edit information.		
Data collection actions	Name, phone number ,email.		
Tests outputs	Pop up message the profile is updated successfully		
Assumptions and constraints	There is none.		
Expected results and criteria	The profile information is updated.		
Test procedure			

Step number	Operator actions	Expected result and evaluation criteria	Result
1	Click on More	The system displays More page	OK
2	Select the Profile option	System redirects the user to the profile page.	Ok
3	Select edit information option	System redirects the user to the edit information page	Ok
Test condition 1			
4	Repeat 1-2 steps and write the old information	The system will not display a message Because nothing is changed.	Ok

Test Case 12: Contact Us Test

Test ID	**T13**	**Comment**	**Decision**
Test description	The test aims to test the functionality if the user has any concerns or feedback while using the system.	Contact the Parkin's Software team	OK
Verified Requirement	No requirements needed.		
Initial conditions	- the user must be registered. - the feedback sent by the user via one of the options shall be reviewed by Parkin's Software team.		
Tests inputs	- Click on contact us. - choose to contact us option (Email, Twitter or WhatsApp)		
Data collection actions	Parkin's Software team will review the feedback and reply to the user via chosen contact option		

Tests outputs	Successfully redirect the user to the chosen contact us option.		
Assumptions and constraints	- Only registered users will be able to contact the Parkin's Software team		
Expected results and criteria	Contact the Parkin's Software team		
Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	Click on contact us	The system displays contact us page	OK
2	Select the preferred contact us option (Email, Twitter or WhatsApp)	System redirects the user to (Mail, Twitter) all links should be working	Ok
Test condition 1			
3	Repeat 1-2 steps and write wrong email	The system will display an error message "Invalid email"	Ok

Test Case 13: Homepage Test

13.1 Search for parking.

Test ID	**T14.1**	**Comment**	**Decision**
Test description	This test case test if the user can find the nearest parking area		OK
Verified Requirement	Search in the research bar		
Initial conditions	1. Write the name of the wanted. parking area 2. The user must be registered to search the parking area		
Tests inputs	Name character of the wanted parking area		
Data collection actions	Parkin will verify the application specific locations and match the name written with the entered name to reserve parking.		

Tests outputs	Redirect to reserve parking page		
Assumptions and constraints	Only registered users have the availability to search for the parking areas		
Expected results and criteria	Match the content of the entry by the user and transfer it to the reservation page to complete the reservation		
Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	1-Type the closest location into the search box.	Sometimes its Mach different area that is not in the bounders of the slots	OK
2	2-compare the parking lot's name with the one specified in the application.		OK
3	3. Display the locations nearby where parking spots are available for reservations.		OK
Test condition			
4	Repeat steps 1-3	The system will display the areas that are available.	Ok

13.2 Reserve Parking

Test ID	**T14.2**	**Comment**	**Decision**
Test description	This test case test if the user can reserve a parking		OK
Verified Requirement	1. Car stickers ID 2. Plate number		
Initial conditions	The user should be logged in to his account and have previous reservation(s).		
Tests inputs	1. Car stickers ID 2. Plate number		
Data collection actions	1. Parkin will check to see if the input is a numeric number and complies with		

	the field's requirements. 2. If the required fields are filled out, Parkin will transport the user to a page where they may choose the parking zone and the duration of hours needed to complete the reservation of the parking.		
Tests outputs	Redirect to reservation ticket page		
Assumptions and constraints	Only registered users have the availability to search for the parking areas		
Expected results and criteria	Match the content of the entry the user information and transfer it to the ticket page to see the reservation ticket summary		
Test procedure			
Step number			**Result**
1	1-Type the user ID and plate Number		OK
2	2-Select a Parking Zone and Number hours		OK
3	Thank you, message, will appear. and summary of the reservation will show		OK
Test condition			
4	Repeat steps 1-3	The system will display a message “you are successfully reserved a parking “	Ok

Test Case 15: Cancel Reservation Test

Test ID	T15	Comment	Decision
Test description	This function allows users to cancel reservations.		OK
Verified Requirement	None.		
Initial conditions	The user should be logged in to his account and have previous reservation(s).		
Tests inputs	User E-mail, Reservation Ticket.		
Data collection actions	User's E-mail and reservation(s) should exist in the firebase to match.		
Tests outputs	TEST IS NOT COMPLETED.		
Assumptions and constraints	User should receive a confirmation of cancellation message within 30s.		
Expected results and criteria	Successful message of cancellation should appear to user.		
Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	User log in to his account using E-mail and Password.	Home page	OK
2	Go to "Reservation" page.	Reservation page appears with a "Cancel Reservations" button on the bottom.	OK
3	Click on "Cancel Reservation" button.	Cancellation confirmation message box will appear.	OK
4	Confirm cancellation by clicking "Confirm Cancellation".	Successful Message will appear	OK
Test condition			
5	Repeat steps 1-4 and confirm cancellation.	The system will display a success message 'You have cancelled a reservation successfully'.	OK
6	Repeat steps 1-3 and click on 'dismiss' button in the 4th step.	The system will not make any changes.	OK

Test Case 16: Add/Update/Delete Areas Test

Test ID	T16	Comment	Decision
Test description	Test if the system allows the admin to make changes regarding adding, updating and deleting areas.		OK
Verified Requirement	Requires admin ID and password.		
Initial conditions	Admin must be logged in the system.		
Tests inputs	Admin ID and password.		
Data collection actions	Areas to make changes on exists and matches with the data in firebase.		
Tests outputs	NOT RUN TEST		
Assumptions and constraints	Only admin can add, update or delete the areas.		
Expected results and criteria	Changes to areas occur successfully.		
Test procedure			
Step number	**Operator actions**	**Expected result and evaluation criteria**	**Result**
1	Search for areas.	Areas displays successfully.	OK
2	Select the area to make changes.	Area displays on screen.	OK
3	Select “Add area” or “Update area” or “Delete area”.	New changes will be displayed successfully.	OK
Test condition			
4	Repeat steps 1-2 and select an area outside the college parking area.	The system will display an error message ‘Undefined area, please try again’.	OK
5	Repeat steps 1-3 and delete an area.	The system will display a success message ‘An area has been deleted successfully’.	OK
6	Repeat steps 1-3 and add an area that already existed.	The system will display an error message ‘This area already exists, add another area’.	OK

CHAPTER 6: USER MANUAL

6.1 Overview of Parkin

The Parkin system is a smart parking solution that can tackle the problem of spending excessive time searching for an available parking spot. It works by utilizing sensors, a computer system, and display panels to efficiently manage parking and minimize traffic congestion, time wastage, expenses, car emissions, and pollution. There are different techniques that can be employed to detect vehicles in the parking area, such as image processing, which uses cameras to capture several cars simultaneously. The images are then analysed using software that compares the variations between successive frames. The parking lot detection is achieved by recognizing the circular green icon in each parking space, and the system can be easily adjusted as needed. The project uses OpenCV software and real-time parking lot images.

6.2 Features

- Ability to choose parking spot.
- Ability to reserve a spot.
- Ability to cancel a reservation.
- Ability to add sticker ID.
- Ability to contact the admin via email in case of any issues.

6.3 Requirements

The table No.75 below specifies all the requirements needed to operate ParkIn.

Table 72 requirements needed to operate ParkIn

Requirements	Software	Hardware
Hardware System		Yes
Wi-Fi connection	Yes	Yes
Android Mobile	Yes	

6.4 Getting started

The below steps demonstrate how Parkin hardware work.

6.4.1 Hardware connection

In order to successfully use Parkin, it is vital that you have a complete hardware system before you begin using it.

6.4.1.1 ESP32-CAM connection

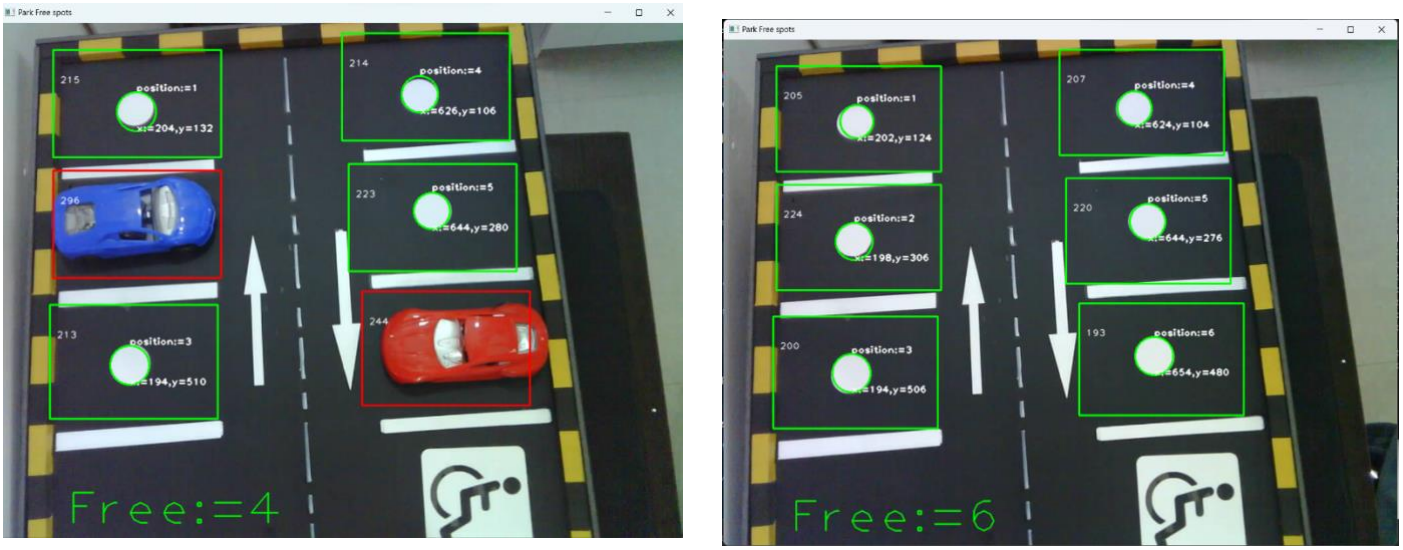


Figure 89 camera connection

6.4.1.2 Wi-Fi Connection

Stable internet connectivity is required for ParkIn to operate.

6.4.2 Application Interfaces

The below steps demonstrate how to use ParkIn application.

The first page that appears when opening the ParkIn app is the walkthrough page, this page gives a quick overview of ParkIn. Keep swiping to the right to get into the next slides.

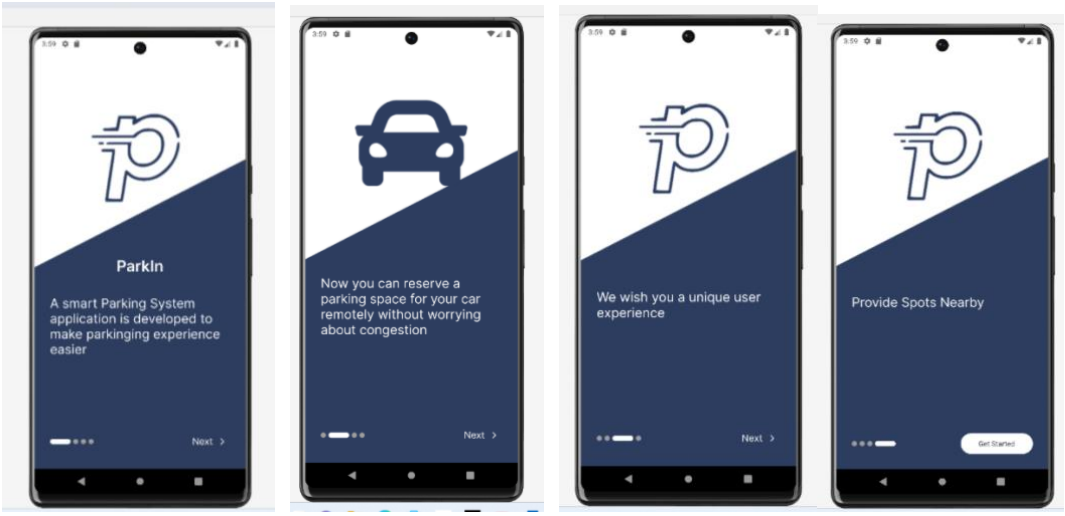


Figure 90 walkthrough page

Then the sign up or sign in pages

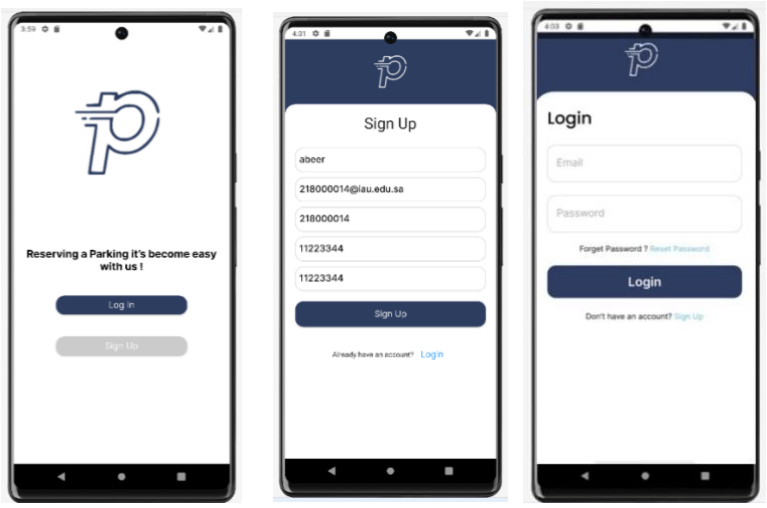


Figure 91 sign up or sign in pages.

Then the reservation pages

Figure 92 reservation pages

6.5 Troubleshooting

6.5.1 (Forgot Password)

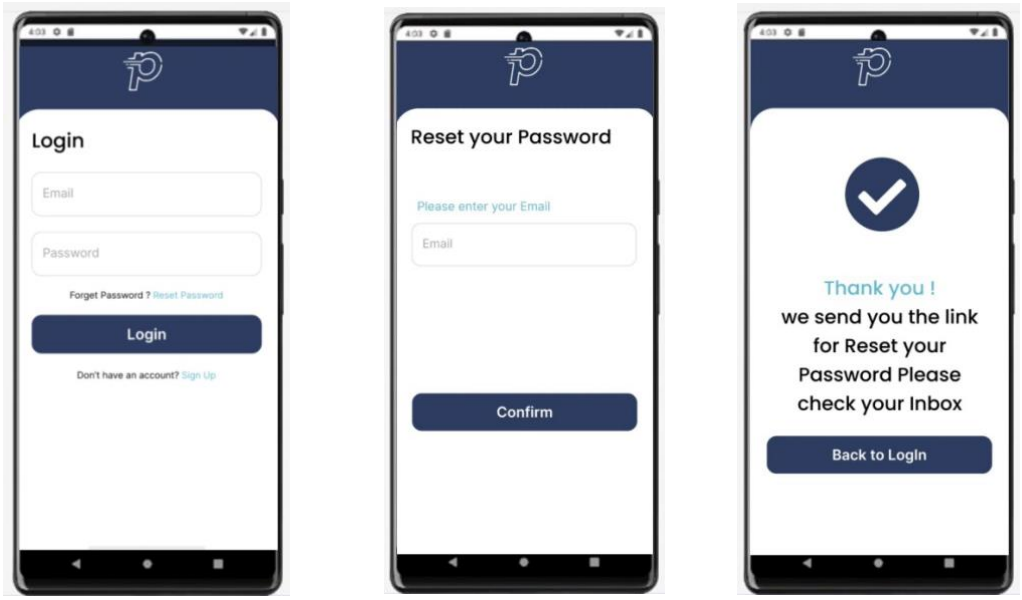


Figure 93 Forgot Password

ParkIn USER MANUAL
Version 1.0
Date: April 2023

CHAPTER 7: DEPLOYMENT

7.1 Project life cycle

The project life cycle illustrates the phases of mobile application development, and this section will briefly revise and explain the phases covered in the previous chapters. In addition, in this chapter, we will cover the final phase of the project life cycle. Furthermore, the waterfall method was followed on the project life cycle, which works through a sequential and logical progression of steps, so those specific goals are defined for each stage of software development and must be achieved in each stage. Then, move to the next stage, and none of the previous stages can be reconsidered after completing them and starting the next stage. Therefore, the waterfall model consists of several phases (Requirement Gathering, Analysis, Design, Coding, Testing, and Deployment).

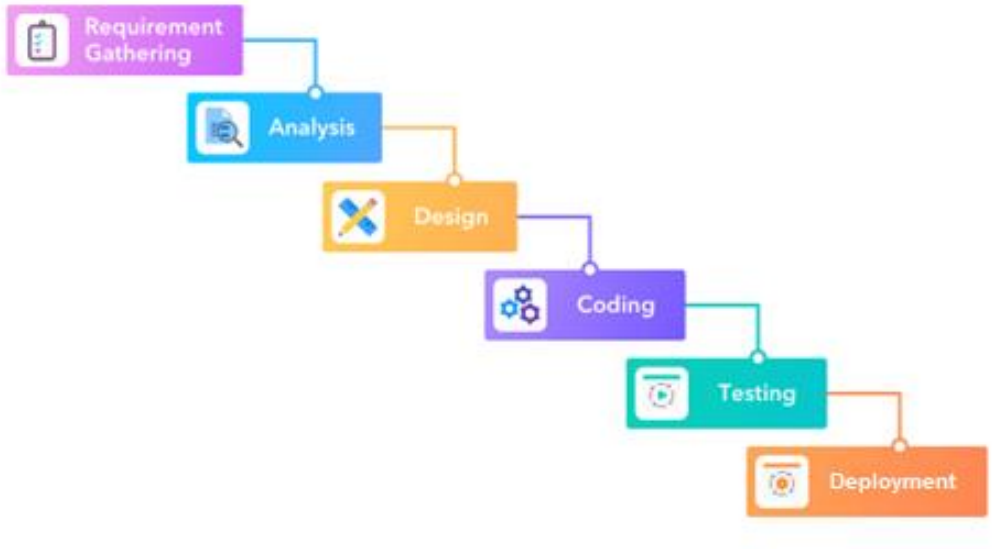


Figure 94 Project life cycle

- **Gathering Requirements:** This stage involves gathering information and comprehending the proposed system's requirements and limitations.
- **Analysis:** Once the requirements have been determined, this stage is the most significant in the SDLC since the objective is to comprehend the new system's requirements and develop a system that satisfies them.
- **Design**: In this stage, we create software and hardware conceptual models after analyzing the proposed system.
- **Coding:** The programming code is created in this stage after the design phase.
- **Testing**: The system's performance is assessed by testing, debugging, and validating its functions after the coding phase.
-

7.2 Deployment

After the project completes the testing phase, the last phase is deployment where the software is deployed to the production environment, and actual users are begun to operate. The figure below Illustrate the deployment; it is a simple diagram of the deployment. This

diagram illustrates a straightforward deployment process, beginning with the student developers and progressing to mobile application code development. Following code development of the hardware, extensive testing was conducted to verify that the application's functionality was working correctly and in the correct order. Finally, the mobile application was deployed for the users' use.

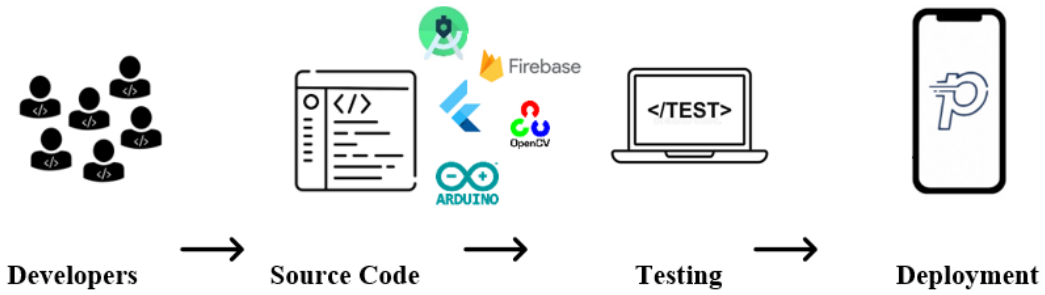


Figure 95 Deployment

7.2.1 Deployment constraints

The Deployment constraints include:

- **Connectivity:** The website application requires an internet connection to function.
- **Limited database access:** The admin has restricted access to the database and will only access it once changes have been made to Firebase for security reasons.
- **Vehicle reservation limits**: The Parkin application restricts each user to a limited number of reservations per vehicle reservation, allowing only one vehicle reservation.
- **Application limitations:** The Parkin app cannot display the status of parking that was not reserved using the Parkin application.
- **English language:** The mobile application is available only in the English language.
- **Database size:** The database server size is restricted.

CHAPTER 8: CONCLUSION

8.1 Conclusion

This report depicts the Parkin application, which allows users to book parking spots that uses a computer vision and image-processing technologies to detect parking space status. The primary concept behind our project's use of computer vision, Internet of Things, and image based. Thus, Parkin can measure whether the parking slot is available or busy by using an ESP32-CAM that sends information to the Arduino IDE, and this is how the availability can be displayed through the application. The team then used the IEEE SRS template to carry out the analysis step of gathering the requirements. The conceptual design and schematics were then created by the team. The team members also created a software test plan that detailed the goals and testing approach. This section includes a thorough explanation of the testing procedure as well as a list of every feature that must be tested. The Software Test Report was therefore created because of this, and it contains the outcomes of the tests that were carried out during the testing phase. Finally, the User Manual provides a summary of the reports.

8.2 Lesson learned.

As students, we have the opportunity to make significant improvements to our talents and selves, such as strengthening our capacity for critical thought, enhancing our interpersonal and teamwork abilities, and deepening our understanding of programming concepts and languages. Additionally, developing computer programming skills is a mental exercise that strengthens critical thinking. The team members have learnt and developed expertise in a variety of practical techniques, tools, and skills during the implementation process, including:

- Improve our ability to communicate, make decisions, solve problems, and think critically.
- Being able to write using the Flutter platform and the Dart programming language gives us the skills we need to understand the world and deal with the problems that come up frequently in our daily lives.
- Time management: Due to the current tri semester plan and the need to learn and use new tools and equipment, in addition to taking additional classes this semester, time management got a little bit more challenging. The team members, however, discovered via this project that time management is the most crucial aspect. As a result, deadlines should be respected, and enforcing stringent standards will help the project achieve its objectives and fulfill its requirements.
- Arduino IDE: Since the team members had never used Arduino IDE before, there were some reservations. However, the programming process grew simpler after viewing the tutorials and understanding how to use the Arduino IDE libraries.
- Firebase: The team members must figure out how to transport data from the Arduino IDE to Firebase in order for the application to show the saved data.
- The team members discovered that the ESP-32 CAM is a helpful tool for recording parking lots to be stored in Firebase with the aid of Arduino IDE code.

8.3 Future work

The following are a number of suggestions that could help to improve the platform:

- Transform the project into a real project, as it is a prototype for now.
- Link the system with the General Department of Traffic to monitor violations.
- Enable the user to add more than one car to the system.
- Add a chatting feature between the user and the admin.

REFERENCES

[1] CG. Hoehne, M.V. Chester, A.M. Fraser, D.A. King, Valley of the sun-drenched parking space: the growth, extent, and implications of parking infrastructure in Phoenix, Cities 89 (2019) 186-198.

[2] j. Arellano Verdejo, F. Alonso-Pecina, E Alba, A. Cuzman Arenas, Optimal allocation of public parking spots in a silt civ: problem characterisation and first algorithms, J. Exp. Theor. Artif. Intell. 31 (4) (2019) 575-597.

[3] J. Lu, J. Wu, L Sun, Control method of urban intelligent parking guidance system based on Internet of Things, Comput. Commun. 153 (2020) 279-285.

[4]J. Karsten, "What is Smart Parking? - Parkeagle - Smart Parking Solutions", *Parkeagle*, 2022. [Online]. Available: https://www.parkeagle.com/2018/05/12/what-is-smart-parking/. [Accessed: 20- Sep- 2022].

[5]I. Education, "What is Deep Learning?", *Ibm.com*, 2022. [Online]. Available: https://www.ibm.com/cloud/learn/deep-learning. [Accessed: 20- Sep- 2022].
[8]."What is IoT? - Internet of Things Beginner's Guide - AWS", *Amazon Web Services, Inc.*, 2022. [Online]. Available: https://aws.amazon.com/what-is/iot/?nc1=h_ls. [Accessed: 20- Sep- 2022].

[6]"Computer Vision in Transportation: What is its impact?", *DataToBiz*, 2022. [Online]. Available: https://www.datatobiz.com/blog/computer-vision-in-transportation/. [Accessed: 20- Sep- 2022].
Y. Wang, "A smart, efficient, and reliable parking sur
[7] Alred, G.J., Brusaw, C.T. & Oliu, W.E. Handbook of Technical Writing. Bedford/St. Martin's, 2006.

[8] Al- Naim M. & Al-Mudara N. Electronic Court. King Faisal University College of Computer Sciences and Information Technology, 2012.

[9]."Ultimate Guide to IoT Based Smart Parking System | CHINT Blog", *CHINT Blog*, 2022. [Online]. Available: https://chintglobal.com/blog/iot-based-smart-parking-system/. [Accessed: 20- Sep- 2022].

[10]."Smart Parking IoT technology Solution Provider - Parking control system", *Libelium*, 2022. [Online]. Available: https://www.libelium.com/iot-solutions/smart-parking. [Accessed: 20- Sep- 2022].

[11].*Irep.iium.edu.my*, 2022. [Online]. Available: http://irep.iium.edu.my/79794/1/79794_Smart%20parking%20guidance%20system.pdf. [Accessed: 20- Sep- 2022].

[12] O. Taylor, P. Ezekiel and V. Emmah, "Smart Vehicle Parking System Using Computer Vision and Internet of Things (IoT)", *European Journal of Information*

Technologies and Computer Science, vol. 1, no. 2, pp. 11-16, 2021. Available: 10.24018/compute.2021.1.2.12..

[13] D. Loong, S. Isaak and Y. Yusof, "Machine vision based smart parking system using Internet of Things", *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 4, p. 2098, 2019. Available: 10.12928/telkomnika.v17i4.12772.

[14] E. Al-Turjman, A. Malekloo, Smart parking in IoT-enabled cities: a survey, Sustain. Cities Soc. (2019) 101608.

[15] A. Faiyetole, "Intelligent Parking Systems for Quasi-Close Communities," *ACADEMIA*, Nov-2019. [Online]. Available: https://www.academia.edu/. [Accessed: Sep-2022].

[16] de Almeida, P. R., Oliveira, L. S., Britto, A. S., Silva, E. J., & Koerich, A. L. (2015). Pklot–a robust dataset for park[1]ing lot classification. Expert Systems with Applications, 42, 4937–4949.

[17] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Con[1]ference on Computer Vision and Pattern Recognition (pp. 580–587).

[18] M. Alam, et al. ,"Real-time smart parking systems integration in distributed ITS for smart cities," Journal of Advanced Transportation, 2018. [Online] Available: https://www.hindawi.com/journals/jat/2018/1485652/B19 [Accessed September 15, 2022].

[19] H. He, Z. Zhang, P. Yan, "A real-time reservation service for smart system," Proceedings of 2018 15th International Conference on Service Systems and Service Management (ICSSSM), Hangzhou, pp. 1–6, 2018.

[20]"Pros And Cons Of Parking Management Systems | BookingNinjas", *Booking Ninjas*, 2022. [Online]. Available: https://www.bookingninjas.com/blog/pros-and-cons-of-parking-management-systems#_____advantages_of_a_parking_management_system_. [Accessed: 20- Sep- 2022].

[21]"Convolutional Neural Network", *DeepAI*, 2022. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network. [Accessed: 25- Sep- 2022].

[22]"What is Computer Vision? | IBM", *Ibm.com*, 2022. [Online]. Available: https://www.ibm.com/sa-en/topics/computer-vision. [Accessed: 25- Sep- 2022].

[23]"What is Deep Learning and How Does It Work?", *SearchEnterpriseAI*, 2022. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network. [Accessed: 25- Sep- 2022].

[24]"Thonny," *Softonic*. [Online]. Available: https://thonny.en.softonic.com/. [Accessed: 23-Oct-2022].

[25]"Geany," *Wikipedia*, 26-Feb-2022. [Online]. Available: https://en.wikipedia.org/wiki/Geany. [Accessed: 23-Oct-2022].

[26]"Raspberry pi documentation," *Display*. [Online]. Available: https://www.raspberrypi.com/documentation/accessories/display.html. [Accessed: 02-Nov-2022].

[27] S. Kumar, "Smart city solutions smart parking lots - internet of things — iot india," 2018. [Online]. Available: https://electronicsofthings.com/ expert-opinion/smart-city-solutions-smart-parking-lots/

[28] R. Grodi, D. B. Rawat, and F. Rios-Gutierrez, "Smart parking: Parking occupancy monitoring and visualization system for smart cities," in SoutheastCon 2016, March 2016

[29] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition (alpr): A state-of-the-art review," IEEE Transactions on Circuits and Systems for Video Technology, vol. 23, no. 2, pp. 311–325, Feb 2013.

[30] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Meghini, and C. Vairo, "Deep learning for decentralized parking lot occupancy detection," Expert Systems with Applications, vol. 72, no. Supplement C, pp. 327 – 334, 2017. [Online]. Available: http://www.sciencedirect.com/science/ article/pii/S095741741630598X

Appendix A: Glossary

Table 73 definitions

Term	Definition
SPMP	Software Project Management Plan. A complete document covering the project plan and management tools for developing and managing the project.
SRS	A Software Requirements Specifications describes a to-be-developed software system. The structure is based on the business requirements specification.
SDS	A software design description is a representation of a software design used to detailed plan for developing a piece of software.
IoT	The Internet of Things, a newly emerging term, refers to the new generation of the Internet that enables understanding between interconnected devices. These devices include tools, sensors, sensors, various artificial intelligence tools, and others.
Deep learning	Deep learning is a type of machine learning and artificial intelligence (AI) that mimics the way humans acquire specific types of knowledge [23].
Computer vision	Computer vision is a subfield of artificial intelligence (AI) that enables computers and systems to extract information from digital images, videos, and make recommendations based on that information [22].
CNN	A convolutional neural network is a type of artificial neural network typically used for analyzing visual imagery [21].
ESP32-CAM	A camera that can capture both still images and high-definition video.
OpenCV	An open-source library dedicated to solving computer vision problems.
Thonny	It is an open-source integrated development environment (IDE) that can be used to create various applications using the Python programming language [24]
Geany	is a free and open-source lightweight GUI text editor using Scintilla and GTK, including basic IDE features [25].
DSI	Display Serial Interface (DSI): is a specification created by the Mobile Industry Processor Interface (MIPI) Alliance with the intention of making display controllers in mobile devices less expensive. It outlines a serial bus and a protocol for communication between the host, the device that is the destination, and the source of the picture data.

Appendix B: Response to Comments

Table 74 Response comments

Comment	**Comment was by**
Component testing coverage target should be fixed number.	Dr. Saqib Saeed
Response	
We have fixed it for each test case there was a fixed percentage.	
Comment	Comment was by
Test cases need to be improved e.g., no input values in sign in.	Dr. Saqib Saeed
Response	
Done. We have added input values for each test case	
Comment	Comment was by
In regression testing should be mentioned which test cases will be repeated at which stage of integration	Dr. Saqib Saeed
Response	
Done	
Comment	Comment was by
In acceptance testing clearly mention what are the conditions	Dr. Saqib Saeed
Response	
Done	
Comment	Comment was by
I recommend extending the part in the related works about deep learning	Nesrine mezhoudi
Response	
We have added study about that	

Appendix C: BUSINESS MODEL CANVAS

Table 75 Parkin Business model canvas

<table>
<tr><th colspan="2">Parkin Business Model Canvas</th><th>Designed by:</th><th>Date</th><th>Version</th></tr>
<tr><td colspan="2"></td><td>Parkin Team member</td><td>25/05/2023</td><td>1st</td></tr>
<tr><td>key Partners

• Department of Financial.
• Public relation department.</td><td>Key Activities

1- Back-end management: managing the back end of the platform (i.e., program code) and allowing it to function and operate without user access (e.g., Python, flutter, NoSQL).

2- The system is responsible for capture available parking, by using ESP32-CAM and based on it reading, the parking will be automatically capture the available parking by camera , moreover by using the application the available parking will appear by green light as user desire.
3- Platform maintenance: We need to maintain our software and hardware by regularly checking and repairing the necessary parts to keep the software and hardware condition good.

Key Resources
• Flutter application developers.
• Hardware and software.</td><td>Value Propositions

parkin aims to save customers time by:

1. Facilitate booking a parking process to users.
2. Provide users with useful and usable services.
3. Increase users' trustiness and loyalty.
4. Provide a convenient environment for chatting and discussing.
5. Provide services to users with no fees.</td><td>Customer Relationships

User Support through Parkin channels, which are Instagram, Twitter, email, or even through our page "Contact us" in the Application.

1. Social media accounts Twitter: @ Paarrkin

2. Parkin official e-mail: Paarrk.in@gmail.com

Channels

Available in Apple store and Google play.</td><td>Customer Segments

• Universities, institutions, shopping malls and medium and big companies in Kingdom of Saudi Arabia.</td></tr>
<tr><td colspan="3">Cost Structure
• Hardware and software cost.
• Database Cloud service.
• Marketing.</td><td colspan="2">Revenue Streams
• Book fees.
• Annual Subscription.</td></tr>
</table>

Smart parking application android/iOS using dart.

Smart Parking System: Parkin is a smart parking system utilizes an ESP-32 Cam and a Flutter mobile application to enable users to book parking spaces in advance and check their occupancy status. The ESP-32 Cam, installed at the entrance of parking area, captures real-time images and communicates the occupancy information to the mobile app. Users can search for available spots, make reservations, and receive navigation

guidance through the app. The ESP-32 Cam also indicates spot availability through a live web server or mobile application home page.

Appendix D: Project Team Deliverables

Table 76 Project Team Deliverables

Member Name	Tasks	Meeting deadlines	Quality	Issues faced
Abeer AlDkheel	• System Development • User interface mockups • File documentation	ü	Enjoyable work	Time conflict
Aisha Khubrani	• Hardware development • File documentation	ü	Risk sharing	Challenges of assessment
Bashayer AlYami	• System development • Testing planning	ü	Conflict management	Time conflict
Ghadah AlMuhaideb	• Hardware development • Database connection • File documentation	ü	Commitment	Prioritizing the tasks
Ghadeer AlGhamdi	• System Development • System Analysis • ER Diagram • File Documentation	ü	High response rate	Inability to face to face meetings

Appendix E: Poster Acceptance

Parkin: Smart Parking System using Image-Processing

Abeer AlDakheel [1], Aisha Khubrani [1], Bashayer AlYami [1], Ghadah AlMuhaideb [1], and Ghadeer AlGhamdi [1]

[1] College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia

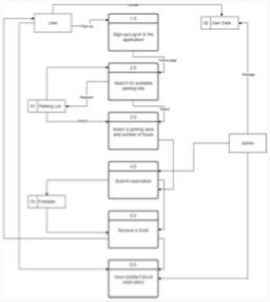
1. INTRODUCTION

The Parkin system is a smart parking solution that can tackle the problem of spending excessive time searching for an available parking spot. It works by utilizing sensors, a computer system, and display panels to efficiently manage parking and minimize traffic congestion, time wastage, expenses, car emissions, and pollution. There are different techniques that can be employed to detect vehicles in the parking area, such as image processing, which uses cameras to capture several cars simultaneously. The images are then analyzed using software that compares the variations between successive frames. The parking lot detection is achieved by recognizing the circular green icon in each parking space, and the system can be easily adjusted as needed. The project uses OpenCV software and real-time parking lot images.

2. OBJECTIVES

- Use IoT to improve existing solutions to reduce car congestion.
- Improve overall parking area of the College of Computer Science and Information Technology to increase efficiency of the parking.
- Help drivers to save time and book space through a mobile application remotely.

3. METHODS



- The user first logs in to the application by entering his information. information will be validated, and an email will be issued to the user.
- The user reserves a parking space and parks his or her vehicle there based on the available slots in the parking spaces.
- When a slot is filled, the database is updated, and the application displays the number of available slots.

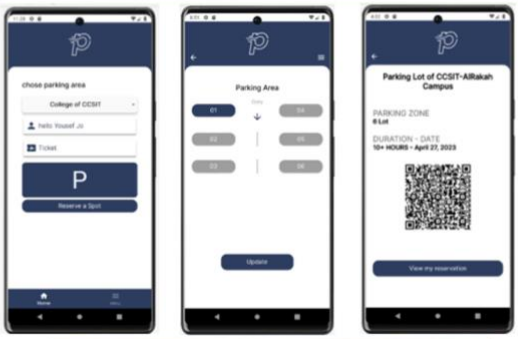
Parkin system is a prototype that was developed using IoT to improve and facilitate the movement of parking vehicles, reduce congestion and accidents in CCSIT parking areas particularly. Observation method was conducted for the movements of parking vehicles in CCSIT parking lots, as we have observed a random parking behaviors.

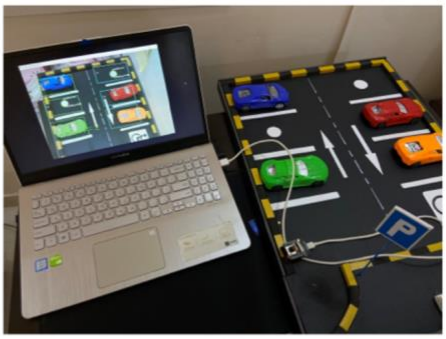
We have developed this project in three phases:

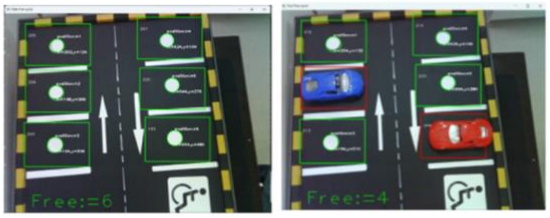
1. **Planning and Gathering Requirements:** The project's overall plan is built at this phase, and the needs have been identified and gathered.
2. **Design:** In the second phase, we have clarified how the system as a whole will be constructed to meet its requirements. This comprises the hardware components, their connections, and the hardware monitoring software.
3. **Implementation:** we divided this phase into two sections: development of hardware and development of software. The two sections were developed simultaneously then they were connected as the final end product. The hardware aspect consists of a ESP32-CAM located near the parking area and a laptop with the help of Arduino IDE to process the images. OpenCV was used to receive the data from the camera and read the status of the parking space such as full or empty. In addition, it was used to send the data to firebase. The application on the other hand was developed using Flutter, it receives the data from Firebase then display it to the user.

Our project was developed in a respectful and trusting environment between the team, in addition to the excellent experience of the team members regarding IoT.

4. RESULTS







5. CONCLUSION

This poster depicts the Parkin application, which allows users to book parking spots that uses a computer vision and image-processing technologies to detect parking space status. The primary concept behind our project's use of computer vision, Internet of Things, and image based. Thus, Parkin can measure whether the parking slot is available or busy by using an ESP32-CAM that sends information to the Arduino IDE, and this is how the availability can be displayed through the application.

REFERENCES

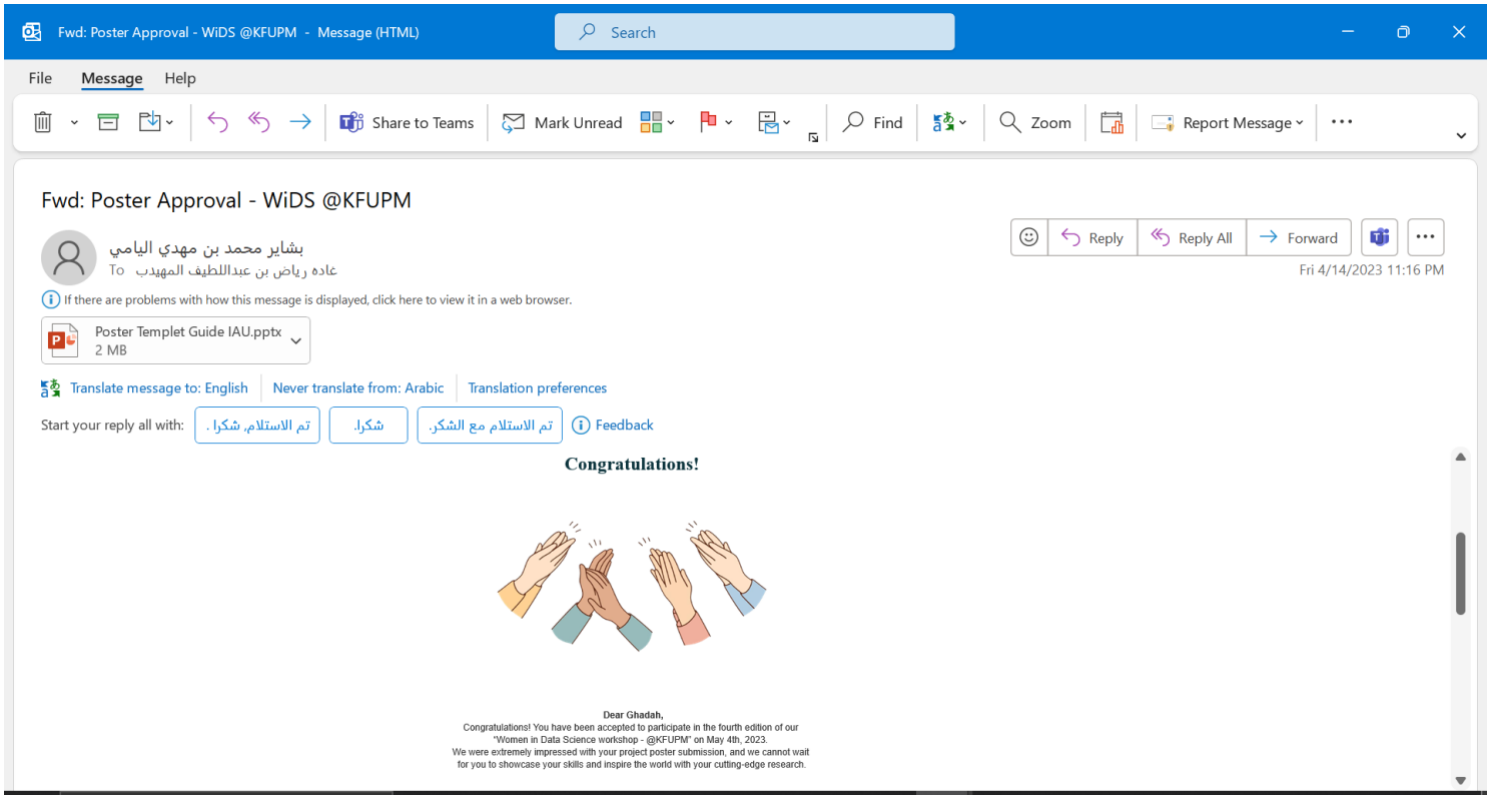


Figure 96 Poster and Acceptance poster email