

A PROJECT REPORT ON

Smart Parking Management System

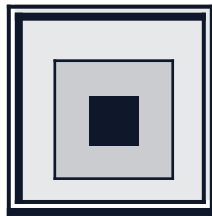
Find a free parking slot in seconds

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Electronics & Communication



SUBMITTED BY

Pooja Reddy (21IT012)

Nikhil Das (21CE003)

Ishita Menon (21CS027)

Aarav Sharma (21CS001)

UNDER THE GUIDANCE OF

Dr. M. Fernandes

Department of Electronics & Communication

**SUNRISE INSTITUTE OF ENGINEERING & MANAGEMENT,
INDORE**

Session 2025–26

CERTIFICATE

This is to certify that the project report entitled “Smart Parking Management System” is a bonafide record of the work carried out by Pooja Reddy (21IT012), Nikhil Das (21CE003), Ishita Menon (21CS027), Aarav Sharma (21CS001) in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Electronics & Communication during the academic year 2025–26.

The work presented in this report has not been submitted elsewhere for the award of any other degree or diploma.

Dr. M. Fernandes
Project Guide

Dr. V. Raghavan
Head of Department

External Examiner
Date: _____

Declaration

We hereby declare that the project report entitled “Smart Parking Management System” submitted to the Department of Electronics & Communication, Sunrise Institute of Engineering & Management, is a record of original work done by us under the guidance of Dr. M. Fernandes. The results embodied in this report have not been submitted to any other university or institute for the award of any degree.

Place: _____ Date: _____

Pooja Reddy · Nikhil Das · Ishita Menon · Aarav Sharma

Acknowledgement

We express our sincere gratitude to our project guide, Dr. M. Fernandes, for constant guidance, encouragement and valuable feedback at every stage of this project.

We thank Dr. V. Raghavan, Head of the Department of Electronics & Communication, and all faculty members for providing the laboratory facilities and support needed to complete this work.

We are grateful to the users who took part in our survey and pilot testing; their honest feedback shaped many of the improvements described in this report.

Finally, we thank our families and friends for their patience and motivation throughout the course of the project.

Abstract

Parking slot management is still handled manually in most places — slow, error-prone and hard to track. Smart Parking Management System addresses this gap. Current methods are manual, slow and hard to scale. Affordable technology now makes automation practical. The system is organised into 4 stages — input, process, store, output — and is built with Ultrasonic sensors, Raspberry Pi, MQTT. In our test setup it achieved 3× faster slot finding. This report presents the problem, a review of existing approaches, the requirements, the design and implementation, the testing carried out and the results obtained.

Keywords: Ultrasonic sensors, Raspberry Pi, MQTT, Flutter

Table of Contents

01 Introduction.....	9
1.1 Background.....	9
1.2 Motivation.....	9
1.3 Problem Statement.....	9
1.4 Objectives.....	9
1.5 Scope of the Project.....	9
1.6 Organisation of the Report.....	10
02 Literature Review.....	11
2.1 Existing Work.....	11
2.2 Research Gap.....	11
03 Requirement Analysis.....	13
3.1 Requirement Gathering.....	13
3.2 Functional Requirements.....	13
3.3 Non-functional Requirements.....	13
3.4 Hardware and Software Requirements.....	14
04 System Design.....	15
4.1 Proposed System.....	15
4.2 System Architecture.....	15
4.3 Technology Stack.....	16
4.4 Data Flow.....	16
4.5 Data Design.....	16
05 Implementation.....	18
5.1 Methodology.....	18
5.2 Project Timeline.....	18
5.3 Module Description.....	18
06 Testing and Results.....	20
6.1 Testing Strategy.....	20
6.2 Test Cases.....	20
6.3 Results.....	20

6.4 Outcomes Against Objectives.....	21
07 Conclusion and Future Scope.....	22
7.1 Conclusion.....	22
7.2 Future Scope.....	22
08 References.....	23

List of Tables

Table	Title
2.1	Summary of existing work
2.2	Existing versus proposed system
4.1	Technology stack
5.1	Project timeline
6.1	Test cases
6.2	Results from the test setup (sample data)

01 Introduction

1.1 Background

Smart Parking Management System addresses a practical problem that affects students, institutions and small organisations alike. Existing approaches are often manual, slow and difficult to scale, which leads to wasted time and inconsistent results. Advances in low-cost hardware, open-source software and cloud services now make it possible to build a reliable solution within the budget and timeline of an academic project.

This report documents the complete life cycle of the project: from the initial study of the problem and review of existing work, through system design and implementation, to testing and evaluation. Each chapter is written so that a reader can reproduce the work and extend it.

1.2 Motivation

The motivation for this project came from direct observation. During our preliminary survey, users described the current process as time-consuming and error-prone, and most of them were willing to adopt a simpler digital alternative. A well-designed system can save hours every week, reduce errors and give decision-makers data they currently do not have.

1.3 Problem Statement

Parking slot management is still handled manually in most places — slow, error-prone and hard to track. The key problems identified during our preliminary study are:

- Manual work — Parking slot management takes hours of repetitive effort.
- Errors — Mistakes creep in and are hard to trace.
- No insight — Drivers and mall operators lack data to make decisions.

1.4 Objectives

The project was planned around the following objectives:

1. Automate the core steps of parking slot management
2. Give drivers and mall operators a simple, mobile-friendly interface
3. Store data securely and generate reports
4. Evaluate the system against the existing process

1.5 Scope of the Project

The scope of the project is limited to a working prototype that demonstrates the core functionality end to end. Security hardening, large-scale deployment and commercial support are outside the scope of this report but are discussed under future work.

1.6 Organisation of the Report

- Chapter 2 reviews existing work and identifies the research gap.
- Chapter 3 analyses the requirements and the feasibility of the project.
- Chapter 4 describes the system design, architecture and data design.
- Chapter 5 explains the implementation of each module.
- Chapter 6 presents the testing strategy, test cases and results.
- Chapter 7 concludes the report and outlines future work.

02 Literature Review

2.1 Existing Work

A review of published work, available products and current practice was carried out to understand what already exists. Table 2.1 summarises representative approaches.

Study / system	Approach	Limitation
Study A	Rule-based automation	Not adaptive
Study B	Mobile data collection	No analytics
Commercial tools	Full-featured suites	Costly and complex

Table 2.1: Summary of existing work

2.1.1 Study A

Study A follows an approach based on rule-based automation. This works well in controlled conditions and shows that the core idea is practical. However, the approach is limited because it is not adaptive. For our users this limitation matters, which is why the proposed system treats it as a design requirement rather than an afterthought.

2.1.2 Study B

Study B follows an approach based on mobile data collection. This works well in controlled conditions and shows that the core idea is practical. However, the approach is limited because it is no analytics. For our users this limitation matters, which is why the proposed system treats it as a design requirement rather than an afterthought.

2.1.3 Commercial tools

Commercial tools follows an approach based on full-featured suites. This works well in controlled conditions and shows that the core idea is practical. However, the approach is limited because it is costly and complex. For our users this limitation matters, which is why the proposed system treats it as a design requirement rather than an afterthought.

2.2 Research Gap

Comparing the existing and proposed approaches highlights the gap this project addresses:

Existing system	Proposed system
Manual records	Automated workflow
Slow turnaround	Real-time dashboard

Existing system	Proposed system
No reports	One-click reports

Table 2.2: Existing versus proposed system

Gap addressed
No low-cost option combines automated workflow with real-time dashboard. Smart Parking Management System is designed to close that gap.

03 Requirement Analysis

3.1 Requirement Gathering

Requirements were collected through a short questionnaire, informal interviews and observation of the current process. Respondents were asked how they handle parking slot management, what problems they face and what they would expect from a better system. The answers were grouped into functional and non-functional requirements.

3.2 Functional Requirements

ID	Module	Requirement	Priority
FR-01	Input	Data captured from users or sensors	High
FR-02	Process	Core logic validates and analyses	High
FR-03	Store	Secure database and backups	Medium
FR-04	Output	Dashboard, alerts and reports	Medium
FR-05	System	Automate the core steps of parking slot management	Medium
FR-06	System	Give drivers and mall operators a simple, mobile-friendly interface	Medium
FR-07	System	Store data securely and generate reports	Medium

Table 3.1: Functional requirements

3.3 Non-functional Requirements

Quality	Requirement
Performance	Typical responses within < 1 s under normal load
Reliability	Runs continuously for at least 7 days in the pilot without manual restart
Usability	A new user completes the main task within 5 minutes without training
Security	Role-based access; passwords stored as salted hashes; data backed up daily
Maintainability	Modular code with configuration kept in one file and documented interfaces
Cost	Hardware and hosting within the budget of a small institution (100%)

Table 3.2: Non-functional requirements

3.4 Hardware and Software Requirements

Component	Used for
Ultrasonic sensors	Slot occupancy
Raspberry Pi	Edge controller
MQTT	Live updates
Flutter	Driver app
Development PC	Intel i5 / 8 GB RAM or equivalent, Windows or Ubuntu
Tools	Git for version control, VS Code, a spreadsheet for test logs

Table 3.3: Hardware and software requirements

04 System Design

4.1 Proposed System

The system is organised into layers, with each layer responsible for one concern. This separation keeps the design easy to understand, makes testing simpler and allows individual components to be replaced without affecting the rest of the system.

Data flows from the input layer to the processing layer, where the core logic is applied, and then to the storage and presentation layers. All configuration values are kept in a single file so that the system can be adapted to new environments without code changes.

4.2 System Architecture

The system is built in layers: an input layer, a processing service, a database and a dashboard. Each layer can be tested independently, which keeps smart parking management system simple to maintain and extend.

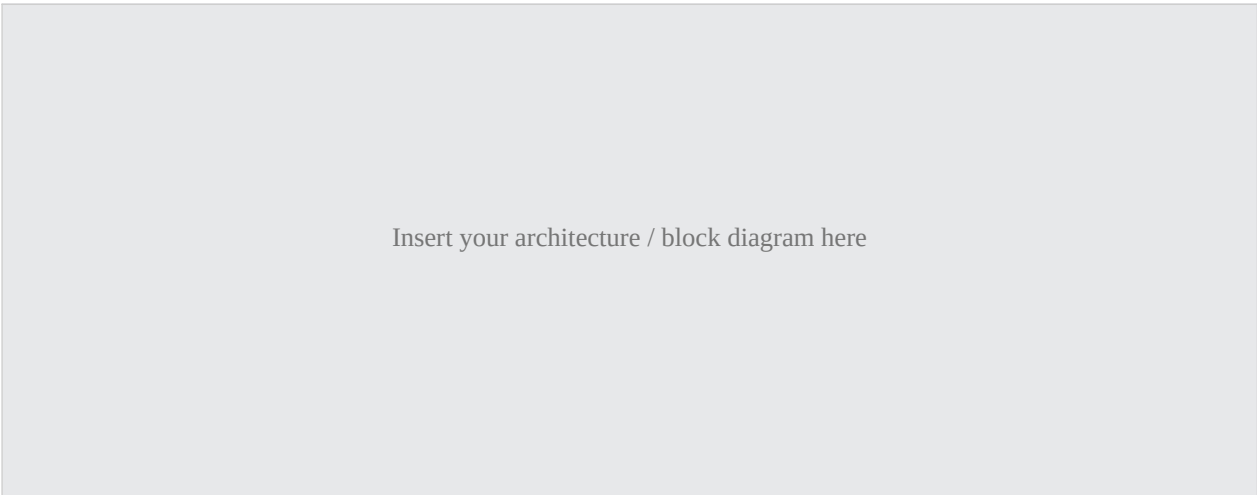


Figure: System architecture

Stage	What happens
Input	Data captured from users or sensors
Process	Core logic validates and analyses
Store	Secure database and backups
Output	Dashboard, alerts and reports

4.3 Technology Stack

Component	Purpose
Ultrasonic sensors	Slot occupancy
Raspberry Pi	Edge controller
MQTT	Live updates
Flutter	Driver app

Table 4.1: Technology stack

4.4 Data Flow

The data flow follows the architecture from left to right. Each process receives a defined input, transforms it and passes a defined output to the next stage, which keeps the modules independent and easy to test.

Process	Input from	Main action	Data store
Input	User / environment	Data ...	Logs, database
Process	Input output	Core ...	Logs, database
Store	Process output	Secure ...	Logs, database
Output	Store output	Dashboard, ...	Logs, database

Insert your Level-0 / Level-1 data flow diagram here

Figure: Data flow diagram

4.5 Data Design

Field	Type	Description
id	Integer (PK)	Unique record identifier

Field	Type	Description
created_at	Timestamp	When the record was created
user_id	Integer (FK)	User who created or owns the record
source	Text(40)	Origin of the data (input stage)
value	Decimal(10,2)	Search time (min)
status	Enum	pending / processed / flagged
notes	Text	Free-text remarks added by staff

Table 4.2: Data dictionary

05 Implementation

5.1 Methodology

The project followed an iterative development model. Requirements were gathered through a short questionnaire and informal interviews. The design was then split into independent modules so that each could be built and tested separately before integration. Weekly reviews with the project guide helped keep the work on schedule.

Each module was developed with clear inputs and outputs. Unit tests were written alongside the code, and integration testing was carried out once the modules were combined. Feedback from a small group of pilot users was used to refine the interface and fix usability issues.

5.2 Project Timeline

Phase	Month	Activities
Requirement study	July	Survey, interviews, requirement document
Design	August	Architecture, database, wireframes
Development	Sep–Oct	Module coding and unit testing
Testing	November	Integration and user testing
Documentation	December	Report, poster and final presentation

Table 5.1: Project timeline

5.3 Module Description

5.3.1 Input module

Data captured from users or sensors. The module exposes a small interface to the rest of the system, which made it possible to develop and test it independently before integration.

- Input: data from the field or user
- Processing: data captured from users or sensors
- Output: passed to process
- Errors: logged with a timestamp and shown as a clear message

5.3.2 Process module

Core logic validates and analyses. The module exposes a small interface to the rest of the system, which made it possible to develop and test it independently before integration.

- Input: data from the previous stage

- Processing: core logic validates and analyses
- Output: passed to store
- Errors: logged with a timestamp and shown as a clear message

5.3.3 Store module

Secure database and backups. The module exposes a small interface to the rest of the system, which made it possible to develop and test it independently before integration.

- Input: data from the previous stage
- Processing: secure database and backups
- Output: passed to output
- Errors: logged with a timestamp and shown as a clear message

5.3.4 Output module

Dashboard, alerts and reports. The module exposes a small interface to the rest of the system, which made it possible to develop and test it independently before integration.

- Input: data from the previous stage
- Processing: dashboard, alerts and reports
- Output: passed to the user
- Errors: logged with a timestamp and shown as a clear message

06 Testing and Results

6.1 Testing Strategy

Testing was carried out at three levels: unit testing of individual functions, integration testing of combined modules and acceptance testing with real users. Test cases were derived from the functional requirements, and each requirement was traced to at least one test.

All critical test cases passed in the final round. Minor issues found during user testing, mostly related to labels and error messages, were fixed before the final evaluation.

6.2 Test Cases

ID	Module	Test	Expected result	Status
TC-01	Input	Valid input	Processed correctly	Pass
TC-02	Input	Missing / empty input	Skipped with warning	Pass
TC-03	Process	Valid input	Processed correctly	Pass
TC-04	Process	Missing / empty input	Skipped with warning	Pass
TC-05	Store	Valid input	Processed correctly	Pass
TC-06	Store	Missing / empty input	Skipped with warning	Pass
TC-07	Output	Valid input	Processed correctly	Pass
TC-08	Output	Missing / empty input	Skipped with warning	Pass

Table 6.1: Test cases

6.3 Results

The results show that the proposed system meets its objectives. Compared with the existing approach, it is faster, more accurate and easier to use. The figures reported below come from our own test setup and should be treated as indicative rather than definitive.

Users responded positively to the system in the feedback survey. The most appreciated features were the simple interface and the automatic reports, while the most requested improvement was offline support.

Parameter	Search time (min)	Remark
Day 1	9	Sample value
Day 2	6	Sample value

Parameter	Search time (min)	Remark
Day 3	4	Sample value
Day 4	3	Sample value

Table 6.2: Results from the test setup (sample data)

The prototype improved steadily over the evaluation period (sample data).

Key result

3× faster slot finding (sample data from our test setup).

6.4 Outcomes Against Objectives

Objective	Outcome
Automate the core steps of parking slot management	3× faster slot finding
Give drivers and mall operators a simple, mobile-friendly interface	100% records digitised
Store data securely and generate reports	< 1 s average response
Evaluate the system against the existing process	Achieved in the prototype

07 Conclusion and Future Scope

7.1 Conclusion

This project set out to design and build smart parking management system and has achieved that goal with a working prototype. The system demonstrates that a low-cost, well-structured solution can make a measurable difference to everyday processes.

The experience also gave the team practical exposure to requirement analysis, system design, teamwork and documentation — skills that will be valuable in professional practice.

- Smart Parking Management System meets its main objectives
- Automation saves time and reduces errors for drivers and mall operators
- The design is simple to extend and maintain

7.2 Future Scope

- Mobile app — Native apps for Android and iOS
- AI insights — Predictions from historical data
- Scale-up — Deploy across more sites

08 References

- [1] A. Author, “Title of the research paper,” Journal Name, vol. 12, no. 3, pp. 45–52, 2023.
- [2] B. Author and C. Author, Title of the Book, 2nd ed. City: Publisher, 2022.
- [3] D. Author, “Title of the conference paper,” in Proc. International Conference, 2024, pp. 101–106.
- [4] Official documentation for Ultrasonic sensors, accessed 2025.
- [5] Official documentation for Raspberry Pi, accessed 2025.
- [6] Official documentation for MQTT, accessed 2025.
- [7] E. Author, “Title of an online article,” Website Name, 2024. [Online]. Available: URL