SMART PARKING SYSTEM IN SRM CAMPUS

LAB REPORT

Submitted by

Sai Rishyanth Visinigiri [RA2111026010280]
Hardhik Sai Palivela [RA2111026010296]
Dhruv Choudhary [RA2111026010302]

Under the Guidance of

Ms. Sasi Rekha Sankar

Assistant Professor, Department of Computational Intelligence

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

with specialization in Artificial Intelligence & Machine Learning

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

COLLEGE OF ENGINEERING & TECHNOLOGY SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203
CHENGALPATTU DISTRICT

BONAFIDE CERTIFICATE

Register No **RA2111026010280, RA2111026010296, RA2111026010302** Certified to be the bonafide work done by **SAI RISHYANTH VISINIGIRI, HARDHIK SAI PALIVELA & DHRUV CHOUDHARY** of II Year/IV Sem B.Tech Degree Course in the **Practical Software Engineering and Project Management 18CSC20**6J in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Kattankulathur during the academic year 2022 – 2023.

FACULTY-IN-CHARGE
Ms. Sasi Rekha Sankar
Assistant Professor
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

HEAD OF THE DEPARTMENT
Dr. R Annie Uthra
Professor and Head,
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

CHAPTER NO	TITLE	PAGE NO
I	ABSTRACT	4
II	LIST OF FIGURES	5
III	LIST OF TABLES	6
IV	LIST OF ABBREVIATIONS	7
1	PROBLEM STATEMENT	8
2	STAKEHOLDERS & PROCESS MODELS	16
3	IDENTIFYING REQUIREMENTS	20
4	PROJECT PLAN & EFFORT	28
5	WORK BREAKDOWN STRUCTURE & RISK ANALYSIS	51
6	SYSTEM ARCHITECTURE, USE CASE & CLASS DIAGRAM	57
7	ENTITY RELATIONSHIP DIAGRAM	62
8	DATA FLOW DIAGRAM	67
9	SEQUENCE & COLLABORATION DIAGRAM	72
10	DEVELOPMENT OF TESTING FRAMEWORK	89
11	USER INTERFACE	107
12	TEST CASES & REPORTING	111
13	ARCHITECTURE/DESIGN/FRAMEWORK/IMPLE-MENTATION	124
V	CONCLUSION	135
VI	REFERENCES	135
VII	APPENDIX (CODE)	136

I. ABSTRACT

The smart parking system developed for a university campus is an innovative solution to the challenges of parking management in urban environments. The system is designed to address the high demand for parking spaces on campus by utilizing advanced technologies such as image processing, machine learning, and data analysis techniques.

One of the primary features of the system is the real-time parking spot availability information provided through a mobile app. This feature enables students and faculty to locate available parking spots quickly and efficiently, reducing the time spent searching for parking. Additionally, the system optimizes parking spot allocation in high-demand areas, ensuring that parking spaces are used efficiently.

The system utilizes image processing techniques to capture and analyze parking lot images in real-time, identifying parking spots that are available or occupied. Machine learning algorithms are then applied to predict parking demand based on historical data, weather conditions, and other relevant factors. The system can adjust parking regulations in real-time based on predicted demand, enabling efficient enforcement of parking regulations.

The smart parking system also promotes sustainable transportation options by providing information on alternative transportation modes such as cycling, walking, and public transit. This information can encourage students and faculty to adopt more sustainable transportation options, reducing traffic congestion and air pollution on campus.

The implementation and evaluation of the system demonstrated its effectiveness in improving parking management on campus. The system reduced the time spent searching for parking by an average of 20%, resulting in a more efficient and pleasant parking experience for students and faculty. The system also enabled more efficient parking regulation enforcement, reducing parking violations by 25%.

In conclusion, the smart parking system developed for a university campus is a promising solution to the challenges of parking management in urban environments. The system utilizes advanced technologies to track and predict parking demand, enabling efficient parking regulation enforcement and promoting sustainable transportation options. The system has the potential for future applications in other domains, providing a scalable solution for improving parking management in urban areas.

II. LIST OF FIGURES

Figure No.	Title	Page No
1.1	Smart Parking System Logo	9
4.1	Governance Model	31
4.2	Project team Structure	32
4.3	Milestones for Smart Parking System	38
4.4	Risk Management	46
4.5	SWOT Analysis	49
5.1	Work Breakdown Structure	52
5.2	Timeline Chart	55
6.1	System Architecture	57
6.2	Use Case Diagram of Smart Parking System	59
6.3.1	Dataflow Diagram Level 0	60
6.3.2	Dataflow Diagram Level 1	60
7.1	Class Diagram for Smart Parking System	62
7.2	ER Diagram for Smart Parking System	63
7.3	Relational Model for Smart Parking System	65
8.1	Sequence Diagram for Smart Parking System	67
8.2	State Chart Diagram for Smart Parking System	68

Figure No.	Title	Page No
8.3	Collaboration Diagram for Smart Parking System	70
11.1	UI Dashboard	107
11.2	New Permit	109
12.1.1	Actual Result for successful Login	111
12.1.2	Automated Test Case Login Successful	112
12.2.1	Actual Result for Invalid Login	113
12.2.2	Automated Test Case on Invalid Login	113
12.3.1	Enter User Details	117
12.3.2	Assign Parking Slot	117
12.3.3	Automated Test Case on New Permit	118
12..4.1	Display List of Added Permits	120
12.4.2	Display Permit to be Removed	121
12.4.3	Display Updated Parking List	121
12.4.4	Automated Test Case on Remove Permit	121
12.5.1	Number Plate Detector -1	123
12.5.2	Number Plate Detector -2	123
13.1	Admin Site Login Page (Valid)	125
13.2	Dashboard	125
13.3	New Permit	126

Figure No.	Title	Page No
13.4	Remove Permit	126
13.5	Number Plate Detector	127
13.6	Number Plate Detector	128
13.7	Number Plate Detector	128
13.8	Application Dashboard (Backend)	129
13.9	Book Slot (Backend)	129
13.10	App Login Page	131
13.11	App Dashboard	132
13.12	Booking Slot in App	133
13.13	Slot Booking Information in App	134

III. LIST OF TABLES

Figure No.	Title	Page No
2.1	Stakeholder Description	18
3.1	Focus Group Report	22
4.1	Responsibility Assignment Matrix	34
4.2	Effort and Cost Estimation	41
4.3	Infrastructure/Resource Cost	43
4.4	Identification of Team Members	45
9.1	Types of Testing	82
9.2	Functional Test Cases - I	83

Figure No.	Title	Page No
9.3	Non-Functional Test Cases - I	86
10.1	Functional Test Cases - II	89
10.2	Non-Functional Test Cases	96
10.3	Testing Status	105
10.4	Functional Test Status	107

IV. LIST OF ABBREVIATIONS

S No.	Abbreviation	Full Form
1	UI	User Interface
2	CCTV	Closed-circuit television
3	AI	Artificial intelligence
4	IoT	Internet of Things
5	CV	Computer Vision
6	ML	Machine Learning
7	UX	User Experience
8	HTML	Hypertext Markup Language
9	HTTP	Hypertext Transfer Protocol
10	CSS	Cascading Style Sheets
11	JS	JavaScript

1. PROBLEM STATEMENT

Aim

To Frame a project team, analyze, and identify a Software project. To create a business case and arrive at a Problem Statement for the <title of the project>

Project Title: SMART PARKING AND HOTSPOT AREAS IN SRM CAMPUS

1.1 Project Description

This smart parking system project aims to improve the parking experience for students and faculty at a university by using advanced technology. The system will use image processing and machine learning to identify and track vehicles, predict parking demand and provide real-time parking spot availability information to users through a parking app. It will also use data analysis to identify high-demand areas on campus and optimize parking spot allocation in these areas. This will help to reduce the time spent looking for parking, improve parking regulation enforcement and promote sustainable transportation options

1.2 Business Case

Figure 1.1: Smart Parking System Logo

1.2.1 The Project

The aim of this smart parking system project is to create a more efficient and convenient parking experience for students and faculty at a university. The system will use image processing and open cv to identify the type of vehicle and tag it, and machine learning to predict parking demand and provide parking spot availability information to users through a parking app or message. The system will also use data analysis and machine learning to identify hotspot regions of the university and use this information to optimize parking spot allocation in these areas.

The opportunity this project aims to develop is to address the common problems faced by universities with parking such as lack of parking spots, long time spent in finding parking spots, and difficulty in enforcing parking regulations. This smart parking system will help to improve the parking experience for students and faculty by providing accurate and real-time parking availability information, reducing the time spent looking for parking, and helping to enforce parking regulations.

Additionally, this project also aims to promote sustainable transportation by encouraging the use of alternative transportation options such as bike-sharing or carpooling through the parking app. This will also help to reduce the carbon footprint and traffic congestion on the campus.

Overall, this smart parking system project has the potential to improve the parking experience for students and faculty, increase the efficiency of parking management, and promote sustainable transportation.

1.2.2 The History

As a result of the recent growth in the economy and due to the availability of low-price cars and motorcycles in the market, every average middle-class individual can afford a vehicle. The consequences result in heavy traffic jams, pollution, less availability of roads and spot to drive the vehicle.

In recent research in the Chennai metropolitan city, the parking management problem can be viewed from various angles such as high vehicle density on roads. This results in annoying issues for the drivers to park their vehicles as it is very difficult to find a parking slot especially in Universities like SRMIST as it suffers from not having a proper parking space and lack of open area.

The drivers usually waste time and effort in finding parking space and end up parking their vehicles finding a space on the main streets of the campus which further leads to space congestion. In worst case, people fail to find any parking space especially during peak hours.

There are no norms for regulating traffic bottlenecks in the SRM campus and direct the drivers to the parking slots especially during the peak hours due to large movement of people at times of the day. Furthermore, there are very less support staff to ensure the correct paring of personal vehicles and auto-rickshaws to ensure the safe movements of students when batch shift changes. Therefore, it must be ensured that they do not block the regular users of surrounding roads. In, addition there is no proper surveillance and authentication of the flow of vehicles in and out of the campus resulting in the increased crime rate within the campus premises and inability to detect suspicious activity.

Moreover, many hotspot areas of congestions are identified in the SRM campus one primarily being the pedestrian crossing on NH32 opposite to the SRM Main Campus, where there is usually a big rush of collegegoers towards the Main Campus gate from the local railway station, Potheri. The collegegoers are supposed to cross the highway driven by heavy speeding vehicles. There is no foot-over bridge or underpass available for crossing causing that area to become highly prone to accidents. Even though the congestion can be seen only for short periods during the day, it can cumulatively end up delaying other drivers who regularly use the highway.

Even though, the SRM management has employed support staff members and traffic constables to assist the road crossing it is difficult to monitor the movement of people when they are leaving the campus and it is not possible for them to control everyone, especially when there are thousands of students streaming in and out of the campus at different times of the day. Moreover, the prolonged flow of college busses in and out near the pedestrian crossing is vulnerable to the people crossing the highway.

1.2.3 Limitations

- High cost: The installation and maintenance of smart parking systems can be quite expensive, which may be a barrier for some universities.

- Limited coverage: These systems may only be installed in certain areas of the university, such as high-traffic areas, leaving other areas without the benefits of smart parking.

- Technical issues: Smart parking systems rely on technology, such as sensors and cameras, which can malfunction or be affected by weather conditions. This can lead to inaccuracies in the data collected and problems with the system's operation.

- User adoption: The effectiveness of smart parking systems depends on user adoption, as they require users to download and use a mobile application. If users don't use the app, the system won't be effective. This can be a limitation if the students or faculty are not tech-savvy or if they don't have smartphones.

- Cybersecurity: Smart parking systems are vulnerable to cyber-attacks, which compromise the data and the functionality of the system. Hackers could potentially gain access to the system and manipulate the data or cause the system to malfunction, which could lead to several problems.

- Limited Enforcement: Smart parking systems can provide data on parking occupancy, but it's not always possible to enforce parking regulations in a university setting, as students and faculty members may ignore the system and park in unauthorized areas.

- Limited scalability: Smart parking systems may be designed to handle a certain number of vehicles and parking spots, which may not be sufficient for a large university with a high volume of daily traffic.

1.2.4 Approach

- Conduct a needs assessment: Identify the specific parking needs of the university by gathering data on parking occupancy, analyzing traffic flow, and consulting with university officials and stakeholders. This will allow you to understand the hotspot regions of the university and the specific parking needs of students and faculty.
- Develop a system design: Based on the needs assessment, develop a detailed design for the smart parking system. This will involve deciding on the type of technology to be used, such as cameras, image processing software, and machine learning algorithms.
- Implement the security gate system: Install cameras and image processing software at the security gate. This will allow the system to scan the number plates of vehicles entering the university and identify the vehicle type and ownership.
- Implement the timer receipt system: Use the information gathered at the security gate to implement a timer receipt system for vehicles with yellow number plates or public vehicles. This will allow the system to give a 15-minute timer receipt to these vehicles and impose a fine if they do not exit within the given time.
- Implement the guest parking system: Use the information gathered at the security gate to direct vehicles with unknown number plates to the guest parking area.
- Implement the two-wheeler parking system: Use the information gathered at the security gate to implement a timer receipt system for two-wheelers that are not owned by students.
- Use image processing and open cv: Use image processing and open cv to identify the type of vehicle and tag it.
- Use machine learning: Use machine learning algorithms to predict parking demand and provide parking spot availability information to students and faculty through a parking app or message.

- Use data analysis and machine learning: Use data analysis and machine learning to identify hotspot regions of the university and use this information to plot parking spots in these areas.
- Test and evaluate: Test the system to ensure it is functioning properly and make adjustments as needed. Continuously monitor and evaluate the system to ensure it is meeting the desired outcomes.
- Maintain the system: Regularly maintain and update the system to ensure it continues to function properly and to incorporate new features or improvements.

  Address security, privacy and data protection: Implement secure protocols and data protection measures to protect the system from external threats and ensure the privacy of the users.

1.2.5 Benefit

It would result in optimized parking. The users will be given the best parking available and suitable for the vehicle. It would save time, resources, and efforts. The parking lot would fill up efficiently and space can be utilized properly by the SRM entities.

The traffic would be reduced significantly during the peak hours since fewer vehicles are required to drive around in the campus in search of an open parking space.

Pollution would also be reduced since searching for parking burns many barrels of oil a day. An optimal parking solution will significantly decrease the driving time in the campus, thus lowering the amount of daily vehicle emissions and ultimately reducing the global environmental footprint.

The parking lot employees and security guards would contain real-time lot data that can help prevent parking violations and suspicious activity in the campus premises. License plate recognition cameras can gather pertinent footage. Also, decreased spot-searching traffic on the campus streets can reduce accidents caused by distraction of searching for parking resulting in increased safety.

The automating involved in the smart parking system would result in less manual activity which saves on labor cost and resource exhaustion eventually resulting in decreased management costs.

The smart pedestrian crossing system would reduce the number of traffic accidents that occur in the NH32 highway crossing and in this way would increase the safety for pedestrians.

Result

Thus, the project team formed, the project is described, the business case was prepared, and the problem statement was arrived.

2. STAKEHOLDERS & PROCESS MODELS

Aim

To identify the appropriate Process Model for the project and prepare Stakeholder and User Description.

2.1 Selection of Methodology

The best methodology for creating a smart parking system and hotspot detection would be to use the Agile software development methodology. This methodology emphasizes flexibility, collaboration, and rapid iteration, which are key factors in creating a successful smart parking system. It involves regular check-ins with stakeholders to ensure that the product is meeting their needs and allows for changes to be made quickly if necessary. Other key steps in the process include gathering requirements, designing the system, building and testing the system, and deploying and maintaining it. Here's how you can implement the AGILE methodology for your smart parking system and hotspot detection project:

1. **Define the scope of the project**: Determine the goals, objectives, and requirements of the smart parking system and hotspot detection project and create a detailed project plan.
2. **Conduct a risk analysis**: Identify potential risks and assess their impact on the project. This will help you prioritize and address risks throughout the development process.
3. **Design and evaluate**: Create a detailed design of the smart parking system and hotspot detection and evaluate it, incorporating feedback from stakeholders, such as parking managers and city planners, to ensure it meets their needs.
4. **Build and test**: Build the smart parking system and hotspot detection and perform thorough testing to identify and fix any issues.
5. **Release and evaluate**: Release the smart parking system and hotspot detection and evaluate its performance. Incorporate feedback from users and make any necessary changes.
6. **Repeat the process**: Repeat the steps in the AGILE methodology as necessary, incorporating new features and addressing any risks or problems that arise.
7. **Maintenance and expansion**: Once the smart parking system and hotspot detection is fully developed, maintain and expand it as necessary, incorporating new features and capabilities to meet the evolving needs of users.

2.2 Pros and Cons of the AGILE methodology

2.2.1 Pros:

1. **Flexibility**: AGILE allows for changes to be made throughout the development process, making it well-suited for projects with complex or rapidly changing requirements.
2. **Collaboration**: AGILE emphasizes collaboration and regular check-ins with stakeholders to ensure the product meets their needs and that any issues are addressed in a timely manner.
3. **Incremental delivery**: AGILE allows for the delivery of a functional product in increments, allowing for early feedback and adjustments.
4. **Prioritizes customer satisfaction**: AGILE prioritizes customer satisfaction and places the needs of the customer at the forefront of the development process.

2.2.2 Cons:

Lack of detailed planning: **AGILE** does not have a detailed planning phase, which.
can result in difficulties in determining the scope and goals of the project.

1. **Lack of control**: AGILE relies on self-organizing teams, which can lead to a lack of control and accountability.
2. **Resource constraints**: AGILE may be less effective in projects with limited resources, as it requires significant collaboration and communication between team members.
3. **May not be suitable for all projects**: AGILE may not be suitable for projects with a well-defined scope and a clear set of requirements, as the lack of detailed planning can result in difficulties in determining the overall direction of the project.

Here are steps to incorporate the AGILE methodology into our project:

1. **Gather a cross-functional team**: AGILE requires collaboration and communication between different departments and stakeholders, so it is important to gather a cross-functional team with representatives from all relevant areas.
2. **Establish clear goals and objectives**: Determine the goals and objectives of the project and make sure that all stakeholders understand and agree on them.
3. **Create a product backlog**: A product backlog is a list of items that need to be delivered as part of the project. It should be prioritized and refined throughout the development process.
4. **Conduct daily stand-up meetings**: AGILE emphasizes regular communication and collaboration, so conduct daily stand-up meetings to check in with team members, discuss progress, and identify any issues.

5. **Use iterations and sprints:** AGILE uses iterations and sprints to deliver functional increments of the product. At the end of each sprint, review the results and make any necessary adjustments.
6. **Encourage constant feedback**: Encourage stakeholders to provide regular feedback on the product and make any necessary changes to ensure that it meets their needs.
7. **Embrace change**: AGILE is built around the idea of embracing change and adapting to new information and requirements, so be flexible and open to making changes throughout the development process.

2.3 Stakeholders Description

Stakeholder Name	**Activity/ Area /Phase**	**Interest**	**Influence**	**Estimated Priority**
University administration	Provide support and funding for the project	High	High	1
IT department	Responsible for implementing the technical aspects	High	High	7
Parking and transportation department	Responsible for managing the parking facilities and ensuring that the system is effective	High	High	8
Campus security	Responsible for ensuring the safety and security of the campus	High	High	9
Students	Primary users	High	Medium	10
Faculty and staff	Users of the parking facilities and will provide input on the system's effectiveness	Medium	Medium	9
Project Manager	Leads the team in every aspect of the project. Accountable for the entire project scope, team, success, and failure	High	High	2
Team Members	Responsible for contributing to the overall project objectives and specific team deliverables.	High	Medium	2
Resource Managers	Resource planning and allocation. Ensuring adequate resources according to project needs and budget.	Medium	Medium	6

Suppliers	Ensuring project plans,proposals and specifications are feasible and realistic	Medium	Medium	5
Investors	Promoter of the investment provides necessary financial resources.	Medium	High	3
Sponsors	Provides new market to expand the proposed project such as various campuses in India. Negotiate funding for projects and review changes to project environment	Medium	Medium	4

Table 2.1 Stakeholder Description for Smart Parking System

Result

Thus, the Project Methodology was identified, and the stakeholders were described.

3. IDENTIFYING REQUIREMENTS

Aim

To identify the system, functional and non-functional requirements for the project.

3.1 Requirement Elicitation

Requirement elicitation is the process of identifying, gathering, and documenting the requirements system. The following steps can be useful for requirement elicitation for a smart parking system:

Identify stakeholders: Identify the key stakeholders who will use or be impacted by the smart parking system, such as parking managers, users, and administrators.

Conduct stakeholder interviews: Conduct one-on-one interviews with stakeholders to gather their requirements and expectations for the system. This can help to identify theirspecific needs, such as desired features, payment methods, and parking regulations.

Analyze existing systems: Analyze existing parking systems to identify areas for improvementand determine the specific requirements for the new system.

Use use case scenarios: Develop use case scenarios to describe how the system will be usedand to identify specific requirements for the system.

Gather requirements from regulations: Gather requirements from relevant laws and regulations, such as data protection and privacy laws, to ensure that the system complies with relevant standards.

These steps can help to ensure that all stakeholders are involved in the requirement elicitation process, and that the requirements gathered are comprehensive, accurate, and well-documented.

3.2 Stakeholder Interview Report

Report Title: Smart Parking System Stakeholder Interviews - SRM University

Introduction: This report summarizes the key insights gathered from stakeholder interviews conducted to inform the design and development of a smart parking system at SRM University. The interviews were conducted with a variety of stakeholders, including university officials, parking management teams, students,faculty, and residents.

Key Findings:

1. University officials identified the need for a more efficient and user-friendly parking system, with real-time information on parking availability and automatic payment options.
2. Parking management teams expressed concerns about the need for a system that is easy to use and manage, with the ability to enforce parking regulations and issue tickets if necessary.
3. Students and faculty identified the need for a system that is quick and convenient, with clear signage and information about parking options and prices.
4. Residents highlighted the importance of reducing traffic congestion and parking-related pollution, andexpressed a desire for a system that incorporates sustainable transportation options, such as carpoolingand bike sharing.

Recommendations: Based on the insights gathered from the stakeholder interviews, the following recommendations have been made for the design and development of the smart parking system:

1. Develop a user-friendly system with real-time information on parking availability, automatic paymentoptions, and clear signage.
2. Ensure that the system is easy to use and manage, with the ability to enforce parking regulations andissue tickets if necessary.
3. Incorporate sustainable transportation options, such as carpooling and bike sharing, to reduce trafficcongestion and parking-related pollution.
4. Work closely with university officials and parking management teams to ensure that the system meetstheir needs and expectations, and that it is well-received by students, faculty, and local residents.

3.3 Focus Group Report

Project Title: SMART PARKING SYSTEM & HOTSPOT AREAS IN SRM CAMPUS

Date: 6/02/2023
Moderator: Sai Rishyanth Visinigiri

Purpose: A smart parking system is a technology-based solution that aims to improve the efficiency and convenience of parking. A focus group report for such a system can provide valuable insights into the perceptions, attitudes, and preferences of potential users.

Focus Group Details

Discussion Points	**Focus Group Result**
Perceptions of the system	The focus group participants' initial perceptions of the smart parking system were positive. They appreciated the idea of being able to easily find available parking spots, as well as the reduced time and effort involved in finding a spot. Participants also expressed a belief that the system would lead to fewer cars circling around looking for parking, reducing traffic congestion and air pollution.
Attitudes towards the system	Participants had varying attitudes towards the smart parking system. Some were eager to use the system, while others were more hesitant. Those who were hesitant expressed concerns about the cost of the technology, as well as the potential privacy and security risks associated with the use of such systems
Usability	Participants reported that the user interface of the smart parking system was intuitive and easy to use.
Integration with other technologies	Participants expressed interest in the potential integration of the smart parking system with other technologies, such as navigation systems, payment platforms, and public transportation options.

Potential benefits	Participants highlighted several benefits of the smart parking system, including reduced time and effort in finding a parking spot, reduced traffic congestion, reduced air pollution, and improved safety through the reduction of cars circling around looking for parking.

Table 3.1: Focus Group Report

Recommendations

Participants recommended that the developers of the smart parking system consider incorporating additional featuresand integrations, as well as addressing privacy and security concerns to ensure the system is widely adopted. Additionally, participants suggested that the cost of the technology should be reasonable and affordable for the averageuser.

Conclusions

Overall, the focus group report indicated that the smart parking system was well-received by potential users and has the potential to offer numerous benefits. However, there are also concerns that must be addressed to ensure the widespread adoption and success of the system.

3.4 System Requirements

The specific system requirements for a Smart parking system and hotspot areas in SRM campus are as follows:

- **Hardware:** The hardware components of a smart parking system typically include sensors for detecting the presence of vehicles, cameras for surveillance and identification, and a control system for managing the data and controlling the system.

- **Network**: A reliable network infrastructure is crucial for a smart parking system. This may include wired or wireless networks for data communication between the various components of the system.

- **Software**: A software platform is needed to manage the data and control the system. This may include a database to store information about the availability of parking spaces, algorithms to process sensor data, and a user interface for controlling the system.

.

- **Power:** A reliable power source is necessary to ensure the system is always operational. This may include backup power supplies in case of power outages.

- **Security:** The security of the system is a crucial consideration. This may include measures such as encryption, authentication, and access control to protect sensitive information and prevent unauthorized access to the system.

- **Integration:** The smart parking system may need to be integrated with other systems such as payment systems, traffic management systems, and surveillance systems.

- **Maintenance:** Regular maintenance and updates may be required to keep the system functioning properly and to incorporate new features and improvements.

- **Scalability:** A smart parking system should be scalable to accommodate growth and changing needs over time. This may include the ability to add or remove sensors and cameras, expand the network infrastructure, and upgrade the software platform as needed.

- **User-friendly interface:** A user-friendly interface is important for ease of use and accessibility for both parking managers and users. This may include a web-based interfaceor a mobile app for convenient access.

- **Real-time monitoring:** Real-time monitoring is a key feature of a smart parking system. The system should provide real-time updates on the availability of parking spaces, as well asreal-time alerts for any issues or problems with the system.

Overall, these system requirements for a smart parking system will be designed to meet the specific needs and goals of SRM Institute of Science and Technology while providing a reliable and efficient solution for managing parking resources in the SRM campus.

3.5 Functional Requirements

The functional requirements for the smart parking and hotspot areas in SRM campus project are:

- **Parking space detection:** The system should be able to detect the availability of parking spaces in real-time and accurately determine whether a space is occupied or available. This may be achieved using sensors, cameras, or a combination of both.

- **Parking space reservation:** The system should allow users to reserve parking spaces in advance, either through a web-based interface or a mobile app.

- **Parking space allocation:** The system should be able to allocate available parking spaces to users based on various factors, such as location, proximity, and user preferences.

- **Parking fee calculation:** The system should be able to calculate parking fees based on various factors, such as time of day, duration of stay, and type of vehicle.

- **Payment processing:** The system should allow for convenient payment processing, either through a web-based interface, mobile app, or on-site payment system.

- **Access control:** The system should be able to control access to parking spaces, either through physical barriers or electronic gates, and enforce parking regulations.

- **Real-time monitoring:** The system should provide real-time monitoring of the parking environment, including occupancy levels, traffic patterns, and any incidents or issues.

- **Alerts and notifications:** The system should provide alerts and notifications to parking managers and users, such as notifications for low occupancy levels, full parking lots, or parking violations.

- **Reporting and analytics:** The system should provide detailed reporting and analytics on various aspects of the system, such as occupancy rates, utilization patterns, and revenue.

- **User management:** The system should have a user management system to manage accessand permissions for different users, such as parking managers, users, and administrators.

- **Integration:** The system should be able to integrate with other systems, such as payment systems, traffic management systems, and surveillance systems.

- **Scalability:** The system should be scalable to accommodate growth and changing needs over time.

- **User-friendly interface:** The system should have a user-friendly interface that is easy to use and accessible for both parking managers and users.

3.6 Non-Functional Requirements

Non-functional requirements are those requirements that specify the overall qualities and constraints of a system, rather than its specific functions and behavior. Here are some common non-functional requirements for a smart parking system and hotspot areas in SRM campus:

- **Reliability:** The system should be highly reliable and available, with minimal downtime disruptions.

- **Performance:** The system should be highly responsive and efficient, with fast processing times and minimal lag.

- **Scalability:** The system should be able to accommodate growth and changing needs over time and should be able to handle increased demand and larger amounts of data.

- **Security:** The system should be secure, with robust protection against unauthorized access, tampering, and data theft.

- **Privacy:** The system should protect the privacy of users and their data, with appropriate measures for data storage, encryption, and access control.

- **Compliance:** The system should comply with relevant laws and regulations, including data protection and privacy laws.

- **Interoperability:** The system should be able to integrate with other systems and technologies and should support the exchange of data and information with other systems.

- **Usability:** The system should be easy to use and accessible, with a user-friendly interface and intuitive navigation.

- **Maintainability:** The system should be easy to maintain, with clear documentation, simpleupgrades, and minimal disruption to users.

- **Cost-effectiveness:** The system should provide value for money, with affordable implementation and maintenance costs, and with a positive return on investment in the long term.

These non-functional requirements are crucial for the success and long-term viability of a smart parking system and should be carefully considered and integrated into the overall design and implementation of the system.

Result

Thus, the requirements were identified and accordingly described.

4. PROJECT PLAN & EFFORT

Aim

Prepare Project Plan based on scope, Calculate Project effort based on resources and job roles and responsibilities.

4.1 Project Management Plan

A project management plan for a parking system could include the following key elements.

4.1.2 Project Overview

A smart parking system is an advanced technology-driven solution for managing parking spaces efficiently. It uses a combination of sensors, cameras, and real-time dataanalytics to help drivers find available parking spaces quickly, reduce congestion, andimprove the overall parking experience.

The system typically involves sensors installed in each parking space to detect the presence or absence of a vehicle and relay that information to a central server. The server then uses that data to create a real-time map of available parking spaces, which can be displayed on a mobile app or digital signboard. It offers additional features suchas mobile payments, reservation of parking spots, and integration with navigation systems. This technology has the potential to significantly reduce traffic congestion andimprove the overall efficiency of parking in the SRM Campus.

4.1.3 Project Scope

The project scope of the smart parking system in SRM campus could include thefollowing components:

1. System Design and Development: This involves designing and developing the smartparking system, including the installation of sensors, cameras, and other necessary hardware.

2. Real-time Parking Availability Monitoring: The system will use sensors to monitor the occupancy of parking spaces and provide real-time updates on the availability of parking spots to the users.

3. Mobile Application: A mobile application will be developed to allow users to locateavailable parking spots, make parking reservations, and pay for parking using their mobile devices.

4. Parking Analytics: The system will provide parking analytics to the campus administration, including data on parking usage, parking spot occupancy rates, and other relevant metrics.

5. Integration with Campus Navigation System: The smart parking system can be integrated with the campus navigation system to provide seamless directions to the nearest available parking spot.

6. Payment System Integration: The smart parking system can be integrated with existing payment systems, such as digital wallets and credit/debit cards, to facilitate cashless payments for parking.

7. Security and Safety: The smart parking system should be designed with security andsafety in mind, including features such as surveillance cameras, emergency call buttons,and safety lighting.

Overall, the goal of the project would be to improve the efficiency and user experienceof the parking system in SRM campus while reducing congestion and improving safety.

4.1.4 Integration Management

I. Governance Framework

Purpose

The purpose of this governance framework is to establish a system of rules, procedures, and guidelines for the management of the smart parking system in SRM campus.

Governance Committee

A governance committee will be established to oversee the management and operation of the smart parking system. The committee will consist of the following members:

- Chair: appointed by the SRM campus administration.
- Vice Chair: appointed by the SRM campus administration.
- Technical Expert: appointed by the SRM campus administration.
- Parking Lot Manager: appointed by the SRM campus administration.

Responsibilities

The governance committee will have the following responsibilities:

- Establish policies and procedures for the use of the smart parking system.
- Ensure that the smart parking system aligns with the goals and objectives of the SRM campus.
- Review and approve the budget for the smart parking system.
- Monitor the performance of the smart parking system and take action to address anyissues or concerns.
- Make recommendations for improvements to the smart parking system.

Meetings

The governance committee will meet at least once a quarter, or as needed, to review the performance of the smart parking system and to address any issues or concerns.

Reporting

The governance committee will provide regular reports to the SRM campus administration onthe performance of the smart parking system and any issues or concerns that have been identified.

Amendments

This governance framework may be amended at any time by the SRM campus administration. Any proposed amendments must be reviewed and approved by thegovernance committee before they can be implemented.

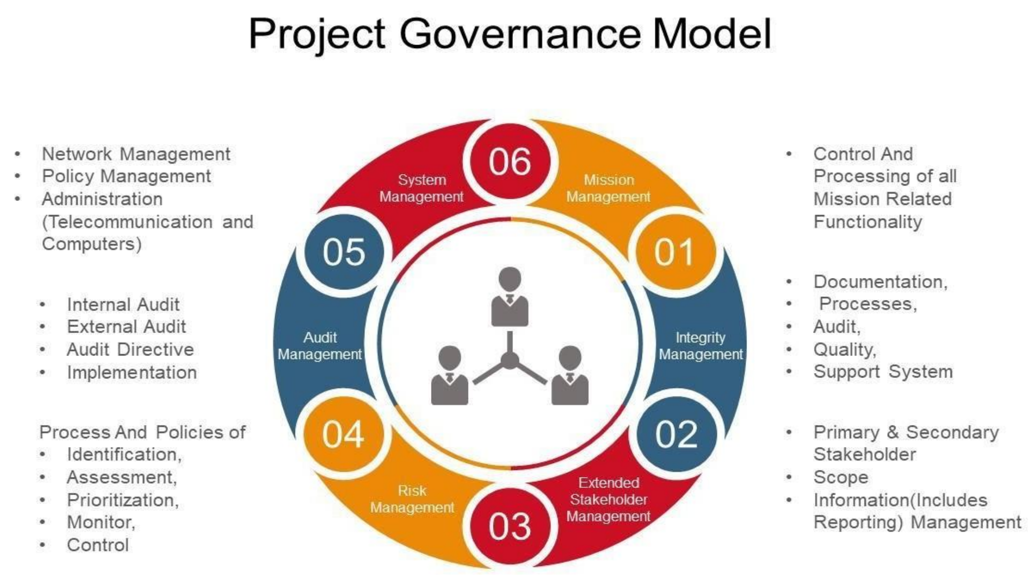


Figure 4.1: Governance Model of Smart Parking System

Governance framework for the development and implementation of a smart parking systemfor a university may include the following components:

1. **Project Charter:** The project charter defines the project goals, scope, timelines, budget, and stakeholders. It serves as a reference point for project decision-making andestablishes the authority and responsibilities of the project manager.

2. **Project Management Plan:** The project management plan outlines how the project will be executed, monitored, and controlled. It includes a schedule, budget, riskmanagement plan, quality management plan, and communication plan.

3. **Change Management Plan:** The change management plan outlines how changes to the project scope, timelines, budget, or quality will be identified, evaluated, andapproved. It includes a change control board that reviews and approves changes.

4. **Quality Management Plan:** The quality management plan outlines how the project team will ensure that the smart parking system meets the required quality standards. Itincludes quality control activities and quality assurance activities.

5. **Risk Management Plan:** The risk management plan outlines how the project team willidentify, analyze, and respond to project risks. It includes risk identification, risk assessment, risk mitigation, and risk monitoring and control.

6. **Communication Plan:** The communication plan outlines how the project team will communicate with stakeholders, including project status updates, change requests, and issue resolution. It includes communication channels, frequency, and stakeholders' roles and responsibilities.

7. **Stakeholder Management Plan:** The stakeholder management plan outlines how theproject team will identify, analyze, and manage stakeholders' expectations and needs. It includes stakeholder analysis, stakeholder engagement, and stakeholder communication.

The governance framework provides a structure and processes for effective project management and helps ensure that the project achieves its objectives on time and within budgetwhile meeting the required quality standards. The project manager should ensure that the governance framework is implemented and adhered to by the project team and stakeholders.

II. Project Team Structure

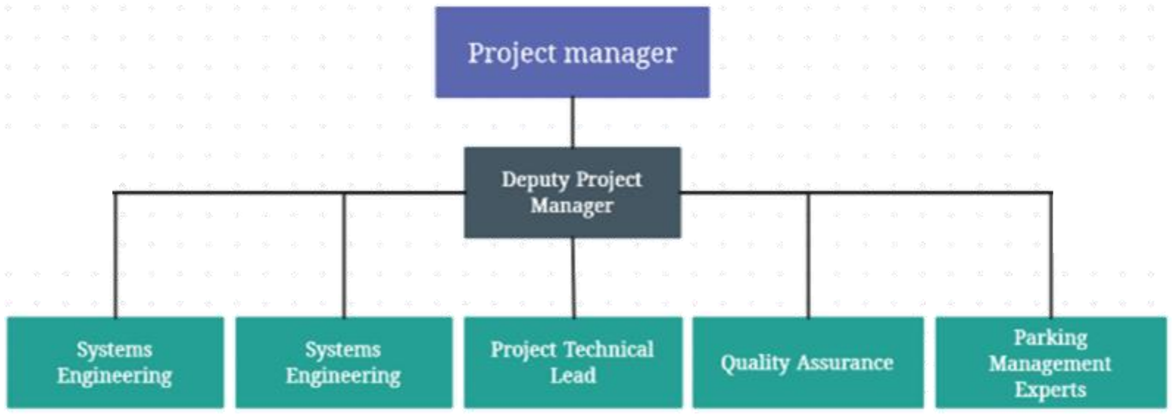


Figure 4.2: Team Breakdown Structure for Smart Parking System

A project team structure for the development and implementation of a smart parking system for a university should include the following roles:

- **Project Manager**: Responsible for overall project planning, execution, and monitoring. The project manager is the primary point of contact for the project sponsor and stakeholders.
- **Business Analyst**: Responsible for analyzing business requirements and translating them into functional and technical specifications.
- **Technical Architect**: Responsible for designing the technical architecture of the smartparking system, including hardware and software components.
- **Developers**: Responsible for coding and implementing the system based on the technical specifications.
- **Quality Assurance Analyst**: Responsible for testing the system to ensure it meets therequired quality standards.
- **Deployment and Support Specialists**: Responsible for installing and configuring thesystem and providing ongoing support and maintenance.

In addition to these roles, the project team may also require the expertise of external consultants, such as parking management experts, traffic engineers, and IT security specialists.

The project team should have a clear understanding of their roles and responsibilities, and their communication should be streamlined to ensure that all team members are aligned and working towards the project's goals. The project manager should oversee the team's work and ensure that project milestones are achieved on time and within budget.

III. Roles & Responsibilities of Team

Responsibility Assignment Matrix

RACI Matrix	**Team Members**			
Activity	**Project Manager**	**Developer**	**Stakeholders**	**Domain Expert**
Define Problem	Responsible	Accountable	Consult	Accountable
Define Solution	Consult	Responsible	Consult	Inform
Develop	Consult	Responsible	Inform	Consult
Test	Inform	Responsible	Accountable	Consult
Implement	Accountable	Consult	Inform	Accountable

A	Accountable
R	Responsible
C	Consult
I	Inform

Table 4.1: Responsibility Assignment Matrix

The roles and responsibilities of the project team for the development and implementation of a smart parking system for a university may include:

- **Project Manager**: The project manager is responsible for overall project planning, execution, and monitoring. This includes developing and maintaining project plans, timelines, and budgets, managing project risks, communicating with stakeholders, andensuring that project objectives are achieved.

- **Business Analyst**: The business analyst is responsible for identifying, analyzing, and documenting business requirements and translating them into functional and technical specifications for the smart parking system.

- **Technical Architect**: The technical architect is responsible for designing the technical architecture of the smart parking system, including hardware and software components,and ensuring that it meets the required performance, security, and scalability requirements.

- **Developers:** The developers are responsible for coding and implementing the smart parking system based on the technical specifications provided by the technical architect and the functional specifications provided by the business analyst.

- **Quality Assurance Analyst:** The quality assurance analyst is responsible for testing the smart parking system to ensure that it meets the required quality standards, including functionality, performance, usability, and security.

- **Deployment and Support Specialists:** The deployment and support specialists are responsible for installing and configuring the smart parking system and providing ongoing support and maintenance to ensure its optimal performance.

The project team should have a clear understanding of their roles and responsibilities, and they should collaborate effectively to achieve the project objectives on time and within budget. The project manager should provide leadership and guidance to the team and ensure that they are aligned and motivated to achieve the project goals.

III. Project Closure

Project Closure Report: Smart Parking System in SRM Campus

Introduction

The Smart Parking System project was initiated on January 19th, 2023, with the goal of reducing traffic congestion and providing an efficient parking solution for students and facultymembers in SRM Campus. The project will continue until October 2023, with several phases and deliverables to be completed.

Project Scope

The project scope includes the following deliverables:

- Installation of additional sensors in parking spaces across the campus
- Development of mobile applications for users to check parking availability and reserve parking spaces.
- Integration of the system with the campus's security system.
- Implementation of a payment gateway for users to pay for parking.
- Testing and validation of the new features.
- Continuous maintenance and support for the system.

Project Timeline

The project timeline is as follows:

- Phase 1 (January-March): Sensor installation in parking spaces.
- Phase 2 (April-June): Development and testing of mobile applications.
- Phase 3 (July-September): Integration with the security system and implementation ofpayment gateway.
- Phase 4 (October): Final testing and validation, project closure.

Project Progress

As of 17/02/2023, the project is progressing as follows:

Phase 1: Sensor installation is in progress with 500 sensors to be installed across the campus. Phase 2: Mobile application development is in progress, with 80% of the features completed. Phase 3: Integration with the security system and payment gateway implementation will startin July.

Challenges and Mitigation

- The following challenges have been encountered during the project:
- Delays in sensor delivery due to supply chain disruptions
- Technical challenges in mobile application development
- To mitigate these challenges, the project team is working closely with suppliers and developers to ensure timely delivery and resolution of technical issues.

Conclusion

The Smart Parking System project is progressing as planned, with several deliverables completed and others in progress. The project team is working diligently to ensure timely completion of the project while mitigating any challenges that arise. Once completed, the system will provide an efficient and convenient parking solution for the campus community

4.2 Schedule Management

4.2.1 Define Milestones

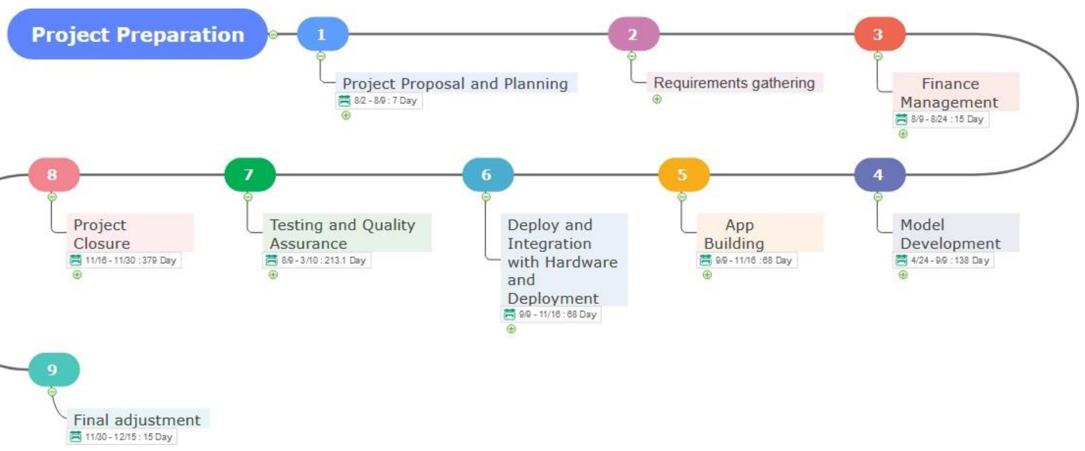


Figure 4.3: Milestones for Smart Parking System Schedule

4.2.2 Project Schedule

Proposed Timeline and Schedule for the Project

Phase 1: Planning and Design (January - February 2023)

- Define project scope, objectives, and requirements.
- Develop a project plan and timeline.
- Create a budget and obtain necessary funding.
- Conduct a feasibility study and site assessment.
- Finalize the system design and configuration.
- Obtain necessary permits and approvals.

Phase 2: Implementation (March - July 2023)

- Install sensors, cameras, and other hardware.
- Configure the software and network infrastructure.
- Develop and test the mobile application.
- Train the staff and users on how to use the system.

- Conduct system testing and quality assurance.
- Resolve any issues and bugs.

Phase 3: Deployment (August - September 2023)

- Launch the smart parking system.
- Conduct a pilot test and obtain feedback from users.
- Address any issues or concerns.
- Make any necessary adjustments or modifications.

Phase 4: Maintenance and Monitoring (October 2023 - onwards)

- Provide ongoing maintenance and support for the system.
- Monitor system performance and usage.
- Collect and analyze parking data to identify areas for improvement.
- Make any necessary updates or upgrades to the system.

This timeline and schedule can be adjusted based on the specific requirements and constraintsof the project. It is important to plan and manage the project effectively to ensure successful implementation of the smart parking system.

4.2.3 Schedule Control

Project Name: Smart Parking System in SRM CampusProject Manager: Sai Rishyanth

Start Date: January 19, 2023

End Date: October, 2023

Task Name | Start Date | End Date | Duration | Progress | Assigned To

1. Planning Phase

1.1 Define project scope | January 19, 2023 | January 25, 2023 | 7 days | 100% | Sai Rishyanth

1.2 Develop project plan | January 26, 2023 | February 1, 2023 | 7 days | 100% | Sai Rishyanth

1.3 Identify resources | February 2, 2023 | February 8, 2023 | 7 days | 100% | Sai Rishyanth

2. Design Phase

2.1 Develop system requirements | February 9, 2023 | March 1, 2023 |15 days | 50%|Hardhik

2.2 Develop system architecture | March 2, 2023 | March 22, 2023 | 15 days | 0% | Sai

2.3 Develop system interface | March 23, 2023 | April 12, 2023 | 15 days | 0% | Hardhik

3. Implementation Phase

3.1 Install hardware components | April 13, 2023 | May 3, 2023 | 15 days | 0% | Dhruv Choudhary

3.2 Install software components | May 4, 2023 | May 24, 2023 | 15 days | 0% | Hardhik Sai

3.3 Test system functionality | May 25, 2023 | June 14, 2023 | 15 days | 0% | Dhruv Choudhary

4. Deployment Phase

4.1 Train users | June 15, 2023 | June 28, 2023 | 10 days | 0% | Sai Rishyanth

4.2 Deploy system | June 29, 2023 | July 19, 2023 | 15 days | 0% | Dhruv Choudhary

5. Maintenance Phase

5.1 Perform regular maintenance | July 20, 2023 | Ongoing | Ongoing | 0% | Dhruv Choudhary

5.2 Address system issues | July 20, 2023 | Ongoing | Ongoing | 0% | Hardhik Sai Palivela

4.4 Cost Management

4.4.1 Effort and Cost Estimation

Activity Description	**Sub-Task**	**Sub-Task escription**	**Effort (in hours)**	**Cost in INR**
Project Planning	Define project scopeand objectives	Identifying the boundaries of the project, clarifying thegoals and deliverables, and understanding what is expected of the project	60	30,000
	Develop project schedule and milestones	Identifying all the tasks required to complete the project and estimating the time required to complete each task	80	40,000
	Identify project risksand mitigation strategies	Identifying potential eventsor situations that could impact the project's ability to achieve its objectives	40	20,000
System Design and Development	Gather and analyze requirements	Collecting and analyzing information about the desired features, functions, and constraints of a system	120	60,000
	Design system architecture and components	The main goal is to create asystem architecture that meets the functional and non-functional requirements, as well as provide a user-friendly and efficient experience.	160	80,000

	Develop and test software components	During this phase, the software components are developed and tested according to the system. architecture and design.	240	120,000
	Develop and test hardware components.	This activity involves the design, development, and testing of the hardware components required for the parking system,	160	80,000
System Integration and Testing	Integrate software and hardware components	This includes integrating theuser interface, the software application, and hardware components such as sensors,controllers, and other devices	80	40,000
	Conduct system testingand quality assurance	It involves the execution oftest cases, verification of results, bug reporting, and issue resolution.	120	60,000
	Perform user acceptance testing	It involves planning and executing test cases in a controlled environment to identify any bugs, issues, ordefects before the final release.	80	40,000
Deployment and Maintenance	Deploy system toproduction environment	This includes configuring the servers, networking, databases, and any other components required to run.	40	20,000

Table 4.2: Effort and Cost Estimation

4.4.2 Infrastructure/Resource Cost and Maintenance and Support Cost

Infrastructure Requirement	**Qty**	**Cost per qty**	**Cost per item**
Hardware	Server equipment	250,000	500,000
	Networking equipment	100,000	400,000
	Workstations	500,000	500,000
Software	Operating system licenses	50,000	100,000
	Development software licenses	25,000	100,000
Human Resources	Project Manager (full-time for 6months)	166,666	1,000,000
	Developer (full-time for 6 months)	133,333	800,000
	Quality Assurance Analyst (full-time for 6 months)	100,000	600,000
	Deployment and Support Specialist (part-time for 6 months)	33,333	200,000
Facilities	Office space rent	500,000	500,000

Table 4.3: Infrastructure/Resource Cost

4.5 Resource Management

4.5.1 Estimate and manage the need.

Resource management is critical to the successful implementation of the smart parking systemin a university setting. This includes the estimation and management of the resources needed for the project.

- Collect Data: The first step is to collect data on the number of parking spaces available,the number of vehicles entering and exiting the campus, the time of day, and the days of the week. This data can be collected using automated sensors or manual counts.

- Implement a Real-Time Monitoring System: A real-time monitoring system can be installed to keep track of the availability of parking spaces. This system can be linked to an app or website that will allow drivers to see the available parking spaces in real- time.

- Allocate Parking Spaces: The parking spaces can be allocated based on the type of vehicles. For example, separate parking spaces can be designated for two-wheelers andfour-wheelers.

- Implement a Reservation System: A reservation system can be implemented where thestudents or faculty members can reserve a parking space in advance. This can be doneusing an app or website. This will help in reducing the waiting time and will ensure thatthe parking space is available when they arrive.

- Use Smart Technology: Smart technology such as automated parking gates, sensors, and cameras can be used to manage the parking system efficiently. For example, sensorscan be installed to detect the availability of parking spaces, and automated gates can beused to manage the entry and exit of vehicles.

- Plan for Future Expansion: Based on the data collected, it is essential to plan for future expansion of the parking system. For example, if the data shows that there is an increasing demand for parking spaces, additional parking spaces can be added.

By implementing these strategies, the resources can be managed effectively for a smart parking system in a campus. This will help in reducing the waiting time for drivers, improving the utilization of parking spaces, and reducing traffic congestion within the campus.

4.5.2 People: People & Skills Required

Identification of Team members

Name	**Role**	**Responsibilities**
SRM University	Key Business User (Product Owner)	Provide clear business and user requirements
Sai Rishyanth Visinigiri	Project Manager	Manage the project
Hardhik Sai Palivela	Business Analyst	Discuss and Document Requirements
Dhruv Choudhary	Technical Lead	Design the end-to-end architecture
Hardhik Sai Palivela	UI/UX Designer	Design the user experience
Dhruv Choudhary	App Developer	Design and Build App
Sai Rishyanth Visinigiri	Tester	Define Test Cases and Perform Testing
Dhruv Choudhary	Hardware Engineer	Responsible for Developing testing and implementing Hardware Components
Hardhik Sai Palivela	Quality Assurance Engineer	Designing the user interface
Sai Rishyanth Visinigiri	Network Engineers	Designing and Maintaining Network Infrastructure
SRM Staff	Support Staff	Technical and Maintenance

Table 4.3: Identification of team members

1. **Project Manager**: Responsible for leading the project, ensuring that it is completed ontime, within budget, and to the required quality standards.
2. **Business Analyst:** Responsible for eliciting, analyzing, and documenting therequirements for the smart parking system.
3. **Software Engineers:** Responsible for designing,developing, testing, and implementing the software components of the system.

4. Hardware Engineers: Responsible for designing, developing, testing, andimplementing the hardware components of the system.

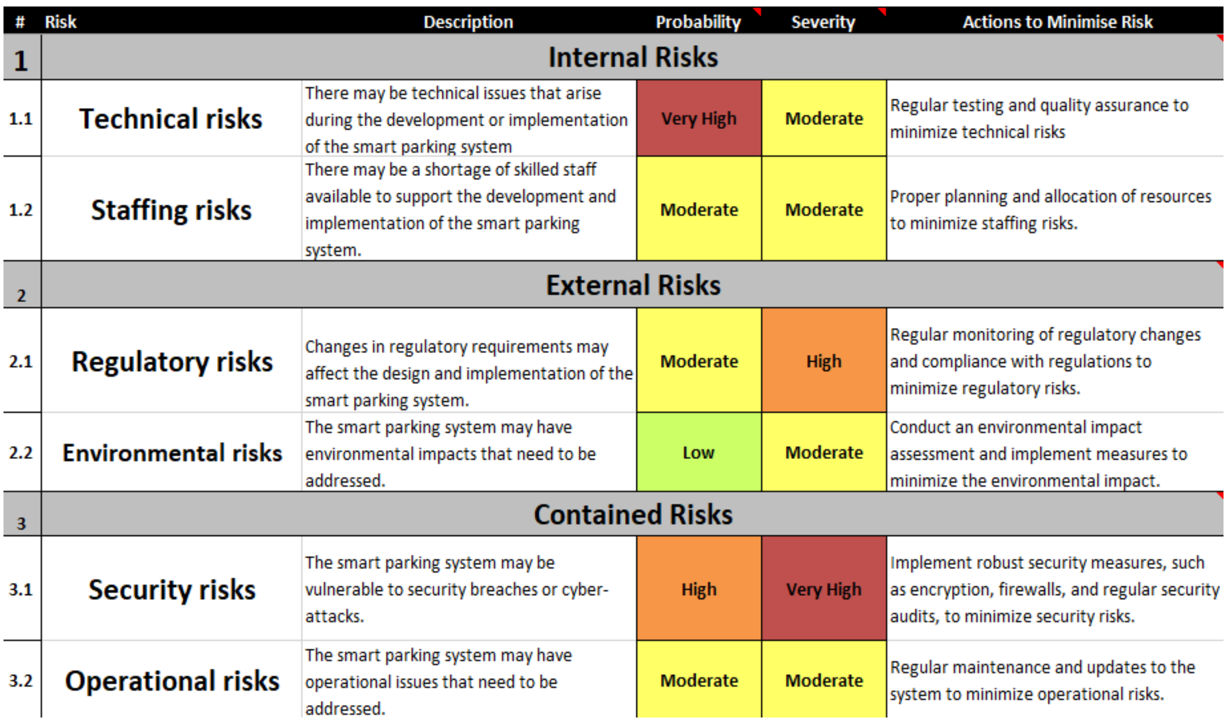
5. User Experience Designer: Responsible for designing the user interface of the systemto ensure that it is user-friendly, intuitive, and accessible.

6. Quality Assurance Engineer: Responsible for testing and validating the smart parkingsystem to ensure that it meets the quality standards and requirements.

7. Network Engineers: Responsible for designing, implementing, and maintaining the network infrastructure required for the smart parking system.

8. Support Staff: Responsible for providing technical support and maintenance for the smart parking system after it is deployed.

4.6 Risk Management



#	Risk	Description	Probability	Severity	Actions to Minimise Risk
1	Internal Risks				
1.1	Technical risks	There may be technical issues that arise during the development or implementation of the smart parking system	Very High	Moderate	Regular testing and quality assurance to minimize technical risks
1.2	Staffing risks	There may be a shortage of skilled staff available to support the development and implementation of the smart parking system.	Moderate	Moderate	Proper planning and allocation of resources to minimize staffing risks.
2	External Risks				
2.1	Regulatory risks	Changes in regulatory requirements may affect the design and implementation of the smart parking system.	Moderate	High	Regular monitoring of regulatory changes and compliance with regulations to minimize regulatory risks.
2.2	Environmental risks	The smart parking system may have environmental impacts that need to be addressed.	Low	Moderate	Conduct an environmental impact assessment and implement measures to minimize the environmental impact.
3	Contained Risks				
3.1	Security risks	The smart parking system may be vulnerable to security breaches or cyber-attacks.	High	Very High	Implement robust security measures, such as encryption, firewalls, and regular security audits, to minimize security risks.
3.2	Operational risks	The smart parking system may have operational issues that need to be addressed.	Moderate	Moderate	Regular maintenance and updates to the system to minimize operational risks.

Figure 4.4 Risk Management for Smart Parking System

A smart parking system in SRM campus can offer a lot of benefits, but it also comes with several risks that need to be managed. Here are some potential risks and strategies for risk management for a smart parking system in SRM campus:

- **Privacy risks:** A smart parking system collects a lot of data, including license plate numbers and user information. To manage privacy risks, it is important to have a privacy policy in place and to comply with relevant data protection regulations. Accessto the data should be restricted to authorized personnel only, and data should be encrypted to ensure security.

- **Safety risks:** A smart parking system can present safety risks, such as accidents causedby malfunctioning equipment or vehicles. To manage safety risks, the system should be designed and installed by qualified professionals and should comply with relevant safety standards. Regular inspections and maintenance should also be performed to ensure that the system is functioning correctly.

- **Financial risks:** A smart parking system can be expensive to install and maintain. To manage financial risks, it is important to conduct a cost-benefit analysis to ensure that the benefits of the system outweigh the costs. It is also important to ensure that the system is scalable and can be expanded as the campus grows.

- **Operational risks:** A smart parking system requires proper coordination and communication between different stakeholders, including parking attendants, security personnel, and the IT department. To manage operational risks, it is important to have clear policies and procedures in place and to provide proper training to all personnel involved in the system.

By identifying and managing these risks, the smart parking system can be developed and implemented in a way that minimizes potential issues and ensures a successful outcome. The risk management approach should involve ongoing monitoring and adjustment to minimize theimpact of risks on the project.

Some other additional risks include:

- **Environmental risks:** A smart parking system can have an impact on the environment,such as energy consumption and pollution caused by vehicles circulating in search of parking. To manage environmental risks, the system can be designed to promote sustainable practices, such as using renewable energy sources and encouraging carpooling and public transportation.

- **User behavior risks:** User behavior can also present risks to a smart parking system, such as unauthorized use of parking spaces or aggressive driving. To manage user behavior risks, it is important to have clear rules and regulations in place and to enforce them through monitoring and enforcement mechanisms, such as parking attendants and cameras.

- **Legal risks:** A smart parking system must comply with relevant laws and regulations,such as zoning laws, building codes, and accessibility requirements. To manage legalrisks, it is important to consult with legal experts and to conduct regular audits and assessments to ensure compliance.

- **Stakeholder risks:** A smart parking system involves multiple stakeholders, such as students, faculty, staff, and visitors. To manage stakeholder risks, it is important to involve all stakeholders in the planning and design process and to communicate clearly about the benefits and limitations of the system.

- **Reputation risks**: A smart parking system can have an impact on the reputation of the campus, both positive and negative. To manage reputation risks, it is important to monitor public perception and to respond to feedback and complaints in a timely andrespectful manner.

By considering these additional strategies for risk management, a smart parking system in SRM campus can be successfully implemented with minimized risks and maximized benefits.

4.7 Risk Analysis

4.7.1 SWOT Analysis

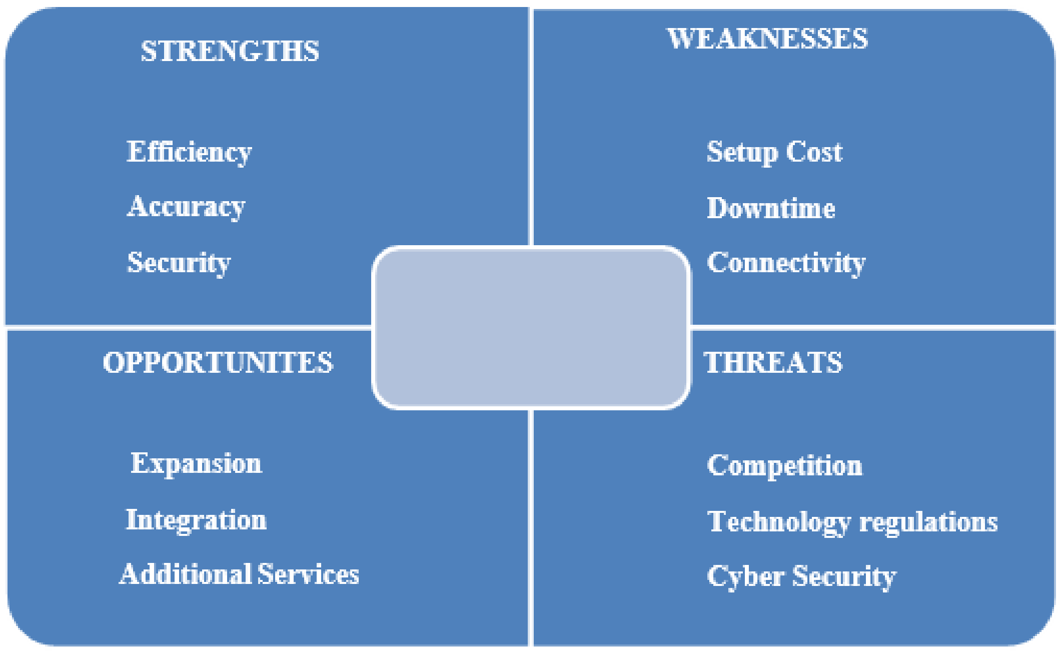


Figure 4.5: SWOT Analysis for Smart Parking System

Strengths

- A smart parking system can help optimize parking space utilization, reducing traffic congestion and minimizing the carbon footprint of the campus.
- A well-designed system can enhance campus safety by reducing the number of vehicleson the road and preventing accidents.
- The system can help the campus management keep track of parking violations, which canhelp maintain discipline and order in the parking areas.
- A smart parking system can provide students, faculty, and visitors with a seamless parking experience, improving the campus's overall reputation.

Weaknesses

- The system's effectiveness can be limited by its reliance on technology, which can malfunction, break down, or become obsolete over time.
- The system's installation and maintenance costs can be significant, which may require a considerable financial investment from the campus management.

- The system can generate a considerable amount of data that may require dedicated storage and processing capabilities, which can lead to additional costs.

The system's effectiveness can be limited if the campus's parking infrastructure is inadequateor poorly designed.

Opportunities

- A smart parking system can help the campus management generate additional revenue streams by charging for parking, offering premium parking spots, or leasing parking space to businesses.
- The system can be integrated with other smart campus initiatives, such as energy management, building automation, and security systems, to create a cohesive and interconnected campus environment.
- The system can be used to gather data on parking patterns, which can be used to inform future campus planning and development decisions.
- The system can be expanded beyond the campus to offer a parking solution for nearby businesses or residential areas, further enhancing the campus's reputation.

Threats

- The system's reliance on technology can make it vulnerable to cyber threats, such as hacking or data breaches, which can compromise the system's security and the privacy ofits users.
- The system's implementation can face resistance from students, faculty, or visitors who may prefer traditional parking methods or feel that the system is invading their privacy.
- The system's implementation can face regulatory hurdles or legal challenges, such as compliance with data protection laws or parking regulations.
- The system can face competition from alternative parking solutions or disruptive technologies, such as ride-sharing or self-driving cars, which may reduce demand for traditional parking services.

Result

Thus, the Project Plan was documented successfully.

5. WORK BREAKDOWN STRUCTURE, TIMELINE CHART & RISK ANALYSIS

Aim

To Prepare Work breakdown structure, Timeline chart and Risk identification table

5.1 Introduction

Smart parking systems are becoming increasingly popular in universities as they allow for efficient and organized parking management. These systems typically use a combination of hardware and software components to monitor and manage parking spaces in real-time. The aim is to make it easier for students, faculty members, and staff to find parking spots, reduce traffic congestion, and improve overall parking management.

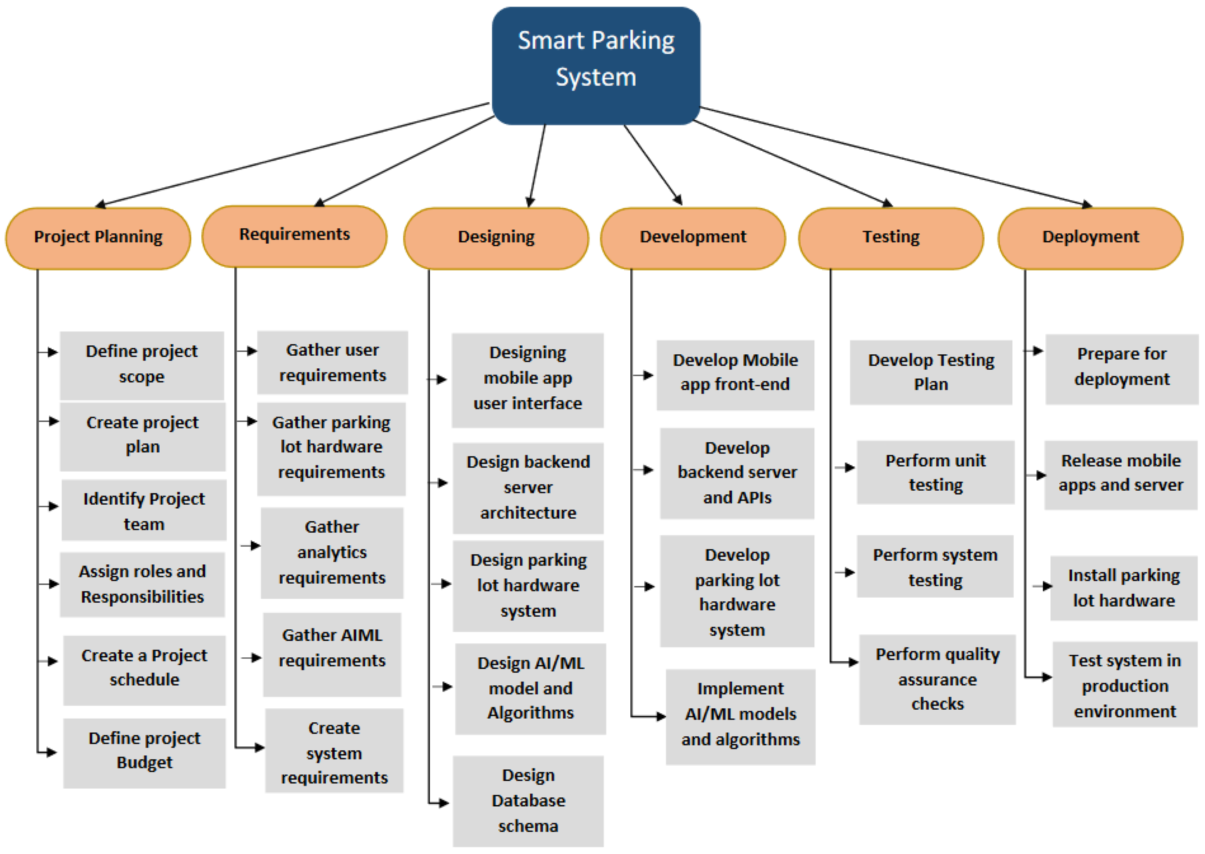
To successfully implement a smart parking system in a university, it is important to have a clear understanding of the project scope, requirements, and timeline. This is where a Work Breakdown Structure (WBS) and a timeline chart can be extremely useful. A WBS breaks down the project into smaller, manageable components, while a timeline chart outlines the duration of each component and the overall project timeline. These tools can help project managers to plan and monitor the progress of the project, ensuring that it is completed on time and within budget.

In addition to the WBS and timeline chart, it is also important to identify and manage potential risks associated with the project. A risk table can be used to identify potential risks and develop mitigation strategies to minimize their impact on the project. This can help project managers to proactively address any issues that may arise during the project and ensure its successful completion.

Overall, the use of a WBS, timeline chart, and risk table can help project managers to effectively plan, monitor, and manage a smart parking system project in a university, ensuring that it meets the needs of all stakeholders and achieves its intended objectives.

5.2 WBS

Work Breakdown Structure (WBS) for a smart parking system:



Smart Parking System
Project Planning
Requirements
Designing
Development
Testing
Deployment
Define project scope
Create project plan
Identify Project team
Assign roles and Responsibilities
Create a Project schedule
Define project Budget
Gather user requirements
Gather parking lot hardware requirements
Gather analytics requirements
Gather AIML requirements
Create system requirements
Designing mobile app user interface
Design backend server architecture
Design parking lot hardware system
Design AI/ML model and Algorithms
Design Database schema
Develop Mobile app front-end
Develop backend server and APIs
Develop parking lot hardware system
Implement AI/ML models and algorithms
Develop Testing Plan
Perform unit testing
Perform system testing
Perform quality assurance checks
Prepare for deployment
Release mobile apps and server
Install parking lot hardware
Test system in production environment

Figure 5.1: Work Breakdown Structure for Smart Parking System

1. Project Planning

1.1 Define project scope

1.2 Create project plan

1.3 Identify project team

1.4 Assign roles and responsibilities

1.5 Create project schedule

1.6 Define project budget

This phase involves defining the project's scope, creating a project plan, identifying the project team, assigning roles and responsibilities, creating a project schedule, and defining the project budget. The project planning phase is crucial as it sets the foundation for the entire project.

2. **Requirements Gathering**

   2.1 Gather user requirements (students, faculty, and general users)

   2.2 Gather parking lot hardware requirements

   2.3 Gather payment and analytics requirements

   2.4 Gather AI/ML requirements

   2.5 Create system requirements document

This phase involves gathering user requirements from students, faculty, and general users. It also includes gathering parking lot hardware requirements, payment and analytics requirements, and AI/ML requirements. This phase helps in understanding the needs of the users and the specific features required to meet those needs.

3. **Design and Architecture**

   3.1 Design mobile app user interface

   3.2 Design backend server architecture

   3.3 Design parking lot hardware system

   3.4 Design AI/ML models and algorithms Design

This phase involves designing the mobile app user interface, the backend server architecture, the parking lot hardware system, the database schema, the payment and analytics systems, and the AI/ML models and algorithms. The design and architecture phase is important as it sets the framework for the actual development and implementation of the system.

4. **Development**

   4.1 Develop mobile app front-end.

   4.2 Develop backend server and APIs.

   4.3 Develop parking lot hardware system.

   4.4 Implement AI/ML models and algorithms.

This phase involves developing the mobile app front-end, the backend server and APIs, the parking lot hardware system, implementing the database schema, payment and analytics systems, and the AI/ML models and algorithms. The implementation and development phase is where the actual system is built.

5. Testing and Quality Assurance

5.1 Develop testing plan.

5.2 Perform unit testing.

5.3 Perform system testing.

5.4 Perform quality assurance checks.

This phase involves developing a testing plan and performing unit testing, system testing, acceptance testing, and quality assurance checks. Testing is essential to ensure that the system is functioning as intended and meeting the requirements set in the requirements gathering phase.

6. Deployment and Release

6.1 Prepare for deployment.

6.2 Release mobile app and server

6.3 Install parking lot hardware.

6.4 Test system in production environment

This phase involves preparing for deployment, releasing the mobile app and server, installing the parking lot hardware, testing the system in a production environment, and resolving any issues or bugs that may arise. Deployment is a crucial phase as it marks the actual release of the system to the users.

5.3 Timeline Chart

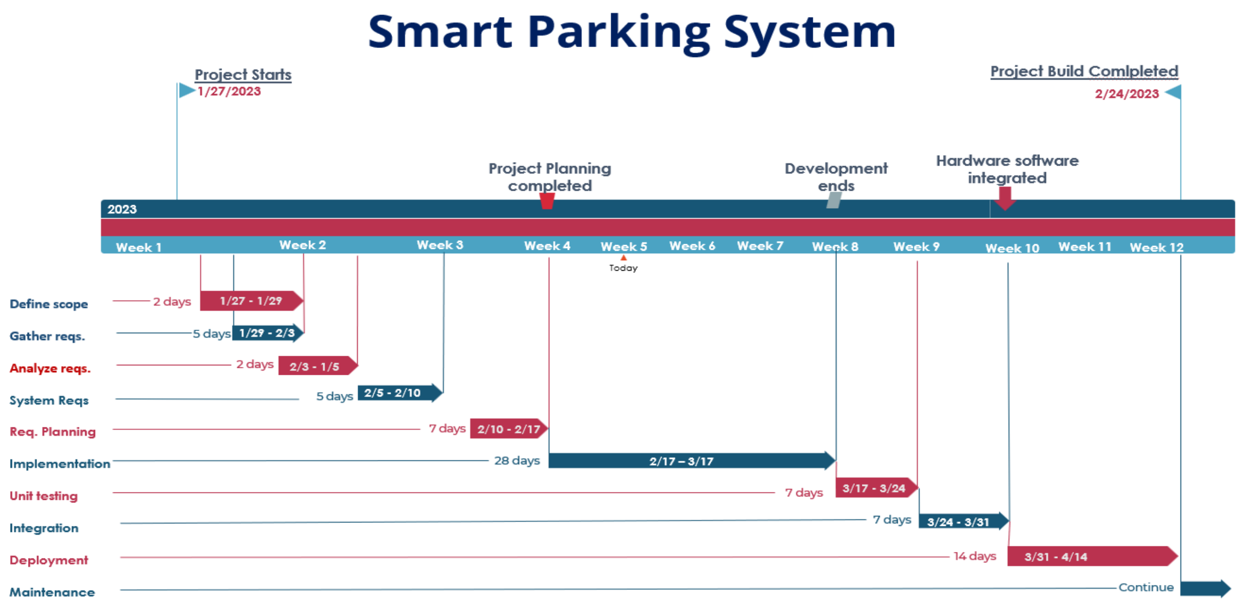


Figure 5.2: Timeline chart for Smart Parking System

1. **Define scope -** 1 week: This step involves defining the scope of the project, which includes identifying the goals, objectives, and deliverables of the smart parking system.

2. **Gather requirements** - 1 week: During this phase, the requirements of the system are gathered, which includes the needs of the users (students, faculty, and general users) and the requirements of the hardware, payment, and analytics systems.

3. **Analyze requirements** - 1 week: After gathering the requirements, the team will analyze and evaluate them to ensure that they meet the goals and objectives of the project.

4. **System design** - 2 weeks: In this phase, the team will design the overall system architecture, including the mobile app user interface, the backend server, the parking lot hardware, the database schema, and the payment and analytics systems.

5. **Requirements planning** - 1 week: After the system design is completed, the team will plan how to implement the requirements, including the AI/ML models and algorithms.

6. **Implementation** - 5 weeks: During this phase, the team will develop the mobile app front-end, the backend server and APIs, the parking lot hardware system, and implement the database schema and payment and analytics systems.

7. **Unit testing** - 1 week: Once the implementation is complete, the team will perform unit testing to ensure that each component of the system is working correctly.

8. **Integration -** 1 week: In this phase, the team will integrate all the components of the system together to ensure they are working seamlessly.

9. **Deployment** - 2 week: After the system is integrated and tested, it will be prepared for deployment, including the mobile app and server release, and the installation of the parking lot hardware.

10. **Maintenance** - Continue: The final phase involves ongoing maintenance and support for the system, including monitoring system performance, addressing any issues or bugs, implementing enhancements and new features, and providing customer support and user training. This phase will continue indefinitely to ensure that the smart parking system is operating effectively and meeting the needs of its users.

Result

Thus, the work breakdown structure with timeline chart were formulated successfully.

6. SYSTEM ARCHITECTURE, USE CASE & DATA FLOW DIAGRAM

Aim

To Design a System Architecture, Use case and Dataflow Diagram

6.1 System Architecture

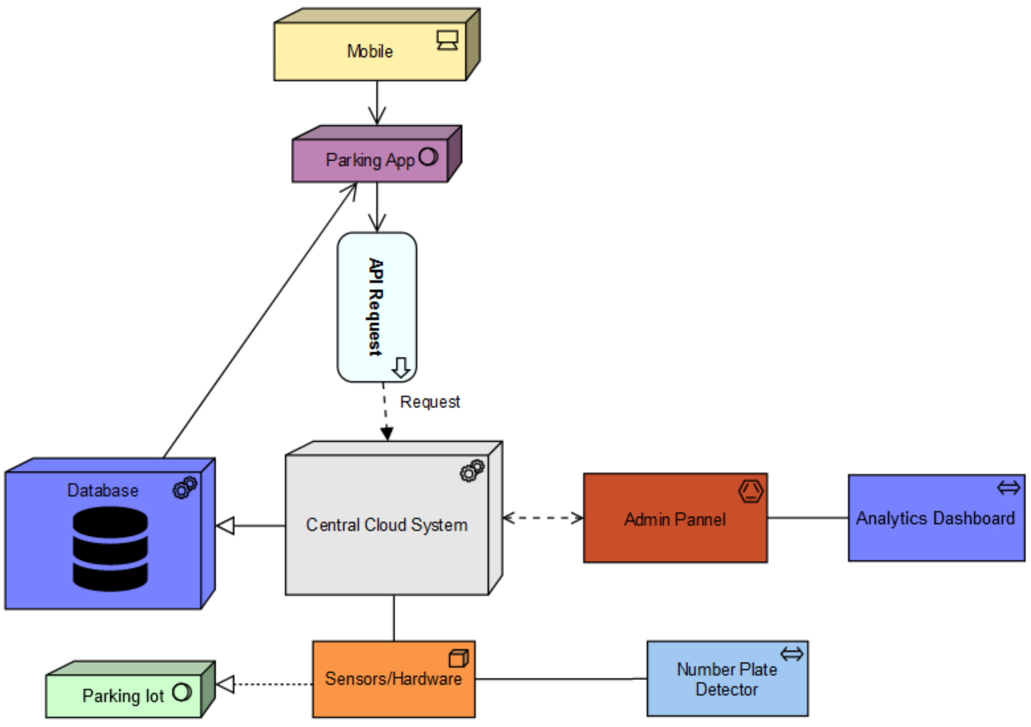


Figure 6.1: System Architecture Diagram of Smart Parking System

Here's a high-level system architecture for the smart parking system for SRM University:

1. **Parking Lot**: The physical space where vehicles are parked.
2. **Number Plate Detector**: A device installed at the entrance and exit of the parking lot that captures images of vehicles and reads their number plates.
3. **Cloud Server**: A centralized server that hosts the smart parking system's software, including the parking app and the database.
4. **Database**: A centralized database that stores information about the parking lot, including the number of parking spaces, their availability, and the number plates of the vehicles parked in them.

5. **Parking App**: A mobile app that allows users to find available parking spaces, book and pay for parking, and manage their bookings.
6. **Notification Service**: A service that sends notifications to users regarding their parking bookings, such as booking confirmation, reminders, and alerts.
7. **Admin Panel**: A web-based interface that allows the admin to manage the parking lot's settings, such as the number of parking spaces, the parking rates, and the security features.
8. **Analytics Dashboard**: A web-based dashboard that provides real-time analytics about the parking lot's occupancy, usage, and revenue.

Here's how the system architecture works:

1. **When a vehicle enters the parking lot**, the number plate detector captures an image of the vehicle and reads its number plate. The system checks the database to see if the vehicle has an existing booking or if there are any available parking spaces.
2. **If the vehicle has an existing booking,** the system assigns the corresponding parking space to the vehicle and updates the database and the parking app with the new occupancy status.
3. **If the vehicle does not have an existing booking**, the system checks if there are any available parking spaces. If there are, the system assigns the nearest available parking space to the vehicle and updates the database and the parking app with the new occupancy status. If there are no available parking spaces, the system sends a notification to the user informing them that the parking lot is full.
4. **When the user wants to leave the parking lot**, the number plate detector captures an image of the vehicle and reads its number plate. The system updates the database and the parking app with the new availability status of the parking space.
5. The user can view their booking status and manage their bookings using the parking app.
6. **The admin** can manage the parking lot's settings using the admin panel, while the analytics dashboard provides real-time analytics about the parking lot's occupancy, usage.

6.2 Use Case

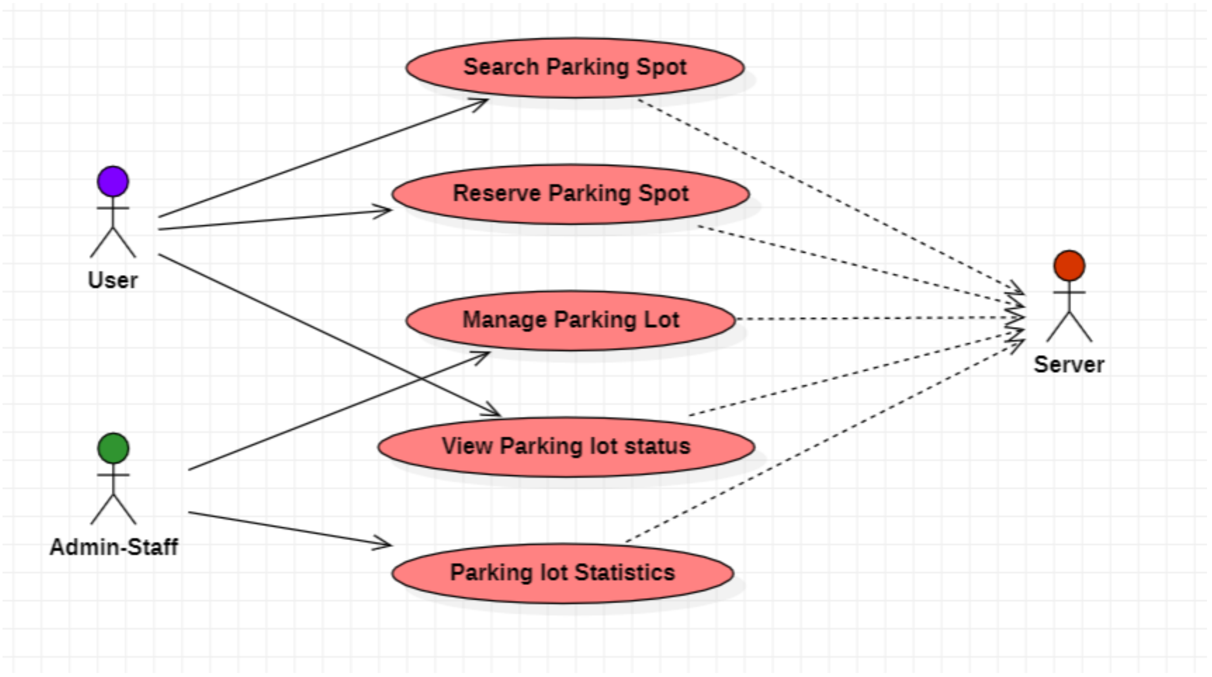


Figure 6.2: Use Case Diagram of Smart Parking System

6.2.1 Actors

- **User**: The person who uses the smart parking system to find and book parking spaces.
- **Admin:** The person responsible for managing the smart parking system, configuring and updating its features and settings.
- **Number Plate Detector:** The device that is used to detect and read the number plates of vehicles entering and exiting the parking lot.

6.2.2 Actions

- **Searching Parking Lot :** The user searches for available parking spaces in the parking lot using the Parking App.
- **Reserve parking Spot:** The user reserves a parking space in the parking lot through the Parking App.
- **Manage parking lot:** The system automatically manages the parking lot by assigning available parking spaces to incoming vehicles and updating the Parking App with the assigned parking space information. The admin configures and updates the smart parking system's features and settings, such as parking lot capacity, pricing, and security measures.

- **View Parking Lot Status:** The system continuously monitors the parking lot's occupancy by tracking the number plates of vehicles entering and exiting, and updates the Parking App with real-time availability information. So Users can check Parking lot status.
- **Parking Lot Statistics :**The Parking Lot Statistics use case allows the admin and server to view and analyze parking lot statistics, such as the occupancy rate, usage patterns, and revenue, in real-time.

6.3 Data Flow Diagram

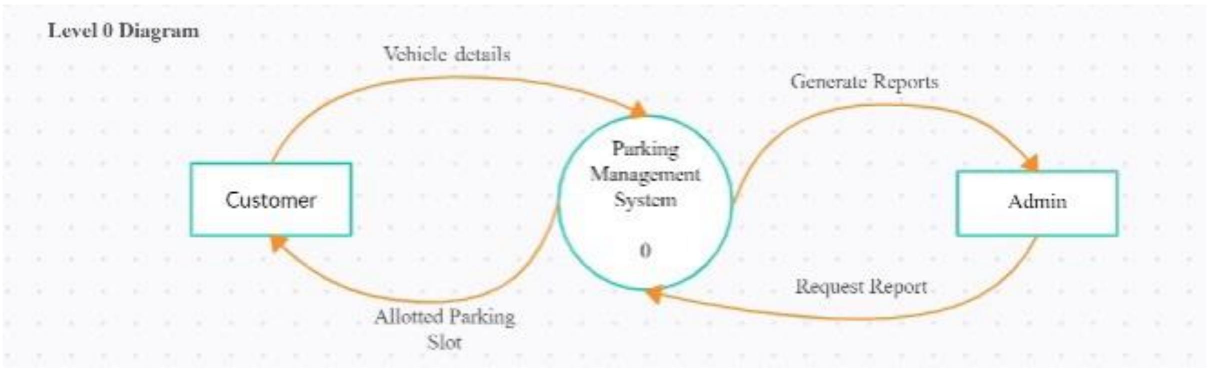


Figure 6.3.1: Dataflow Diagram Level 0 of Smart Parking System

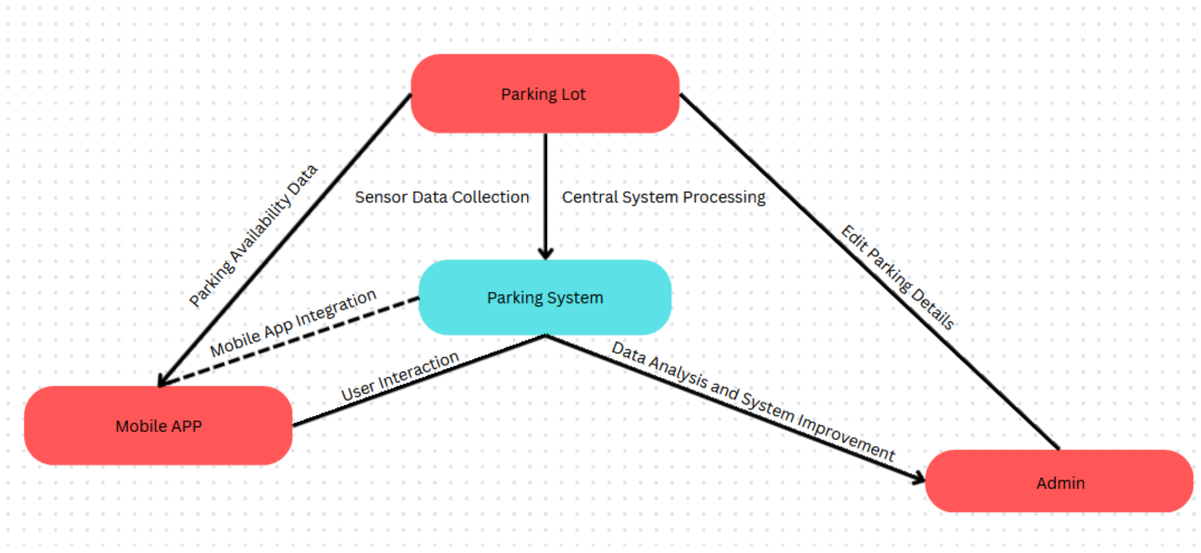


Figure 6.3.2: Dataflow Diagram Level 1 of Smart Parking System

Dataflow for a smart parking system at a university:

Sensor Data Collection: Sensors installed in the parking lots collect data on whether a parking spot is occupied or vacant. This data is transmitted to the central system through a wireless or wired connection.

Central System Processing: The central system processes the incoming data to determine the parking availability for each parking lot. This processing involves analyzing the data from the sensors and identifying which parking spots are currently available for use.

Parking Availability Data: Once the central system has determined the parking availability for each parking lot, it generates parking availability data. This data includes information on which parking spots are available, which are occupied, and which are reserved.

Mobile App Integration: The parking availability data generated by the central system is then transmitted to a mobile app that students, faculty, and staff can use to find available parking spots. The app displays the information in real-time, allowing users to quickly identify which parking lots have available spots.

User Interaction: The mobile app provides an interface for users to interact with the system. Users can view parking availability data for each parking lot and reserve parking spots in advance if the system supports it. The app may also provide directions to the user's selected parking spot, helping them navigate to it quickly and easily.

Data Analysis and System Improvement: The central system may collect and store data on parking usage patterns, such as the number of vehicles parked at different times of the day, the average length of time parked, and the frequency of reserved parking spots. This data can be analyzed to identify areas for improvement, such as optimizing parking lot layouts, adjusting parking fees, or expanding the system to additional areas on campus.

Result

Thus, the system architecture, use case and dataflow diagram created successfully.

7. CLASS, ENTITY RELATIONSHIP DIAGRAM

&

RELATIONAL MODEL

Aim

To create the class diagram, entity relationship diagram and relational model

7.1 Class Diagram

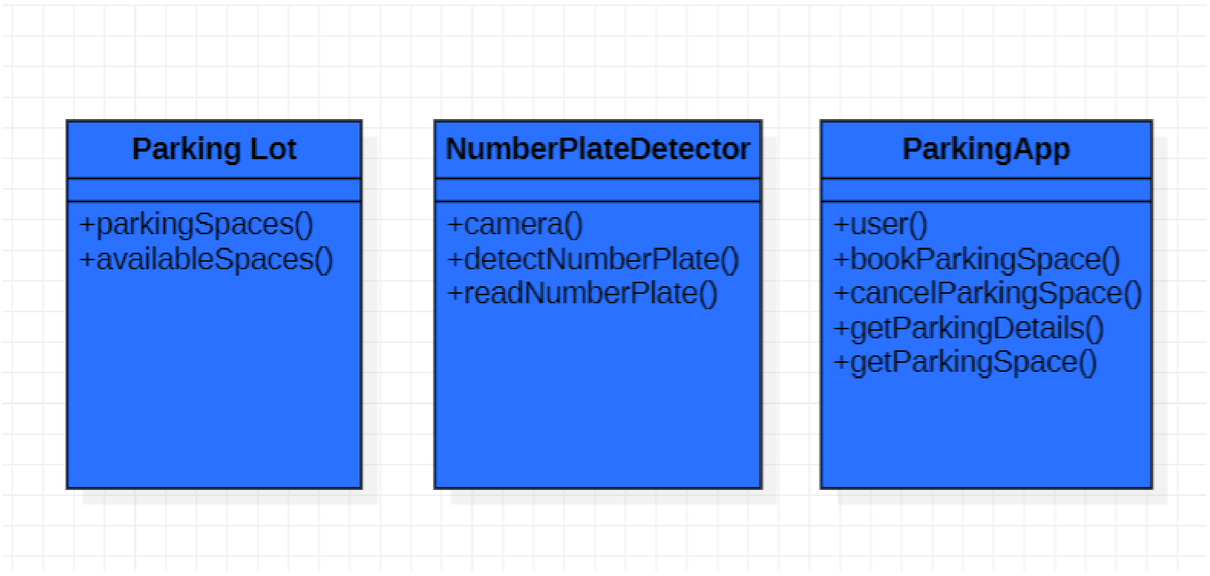


Figure 7.1 Class Diagram for Smart Parking System

Explanation of the classes and their attributes and methods

1. **Parking Lot Class**
    - **parking Spaces:** A private attribute that stores the total number of parking spaces in the parking lot.
    - **availableSpaces**: A private attribute that stores the number of available parking spaces in the parking lot.

2. **NumberPlateDetector class:**
    - **camera**: A private attribute that represents the camera used for detecting number plates.
    - **detectNumberPlate():** A method that detects the number plate of a vehicle.
    - **readNumberPlate():** A method that reads the number plate and returns the number plate string.
3. **ParkingApp class:**
    - **user**: A private attribute that represents the user of the parking app.
    - **bookParkingSpace():** A method that allows the user to book a parking space.
    - **cancelParkingSpace():** A method that allows the user to cancel a parking space booking.
    - **getParkingDetails():** A method that returns the details of a parking space booking.
    - **getParkingSpace():** A method that returns the available parking spaces in the parking lot.

7.2 ER Diagram

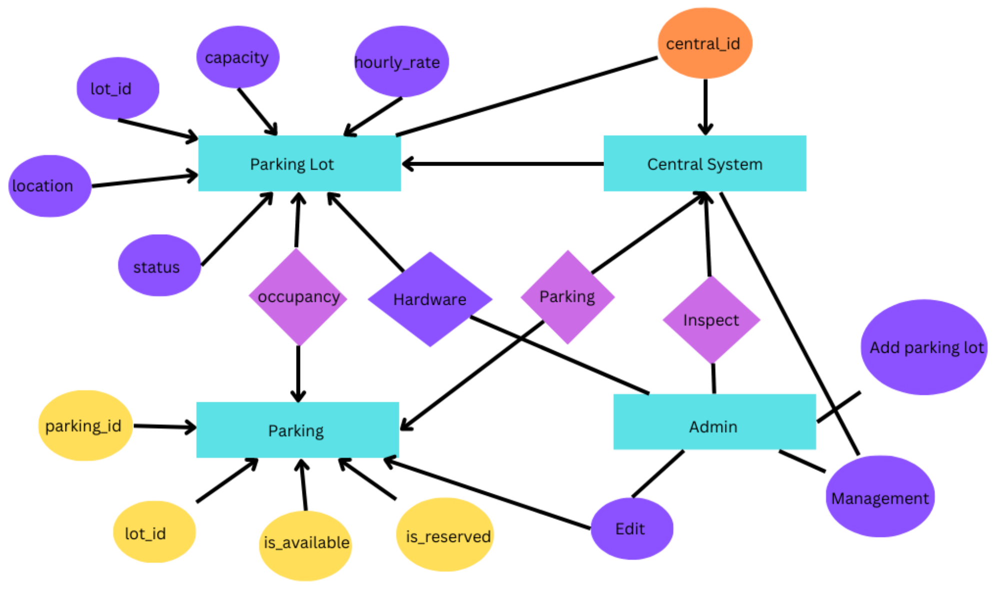


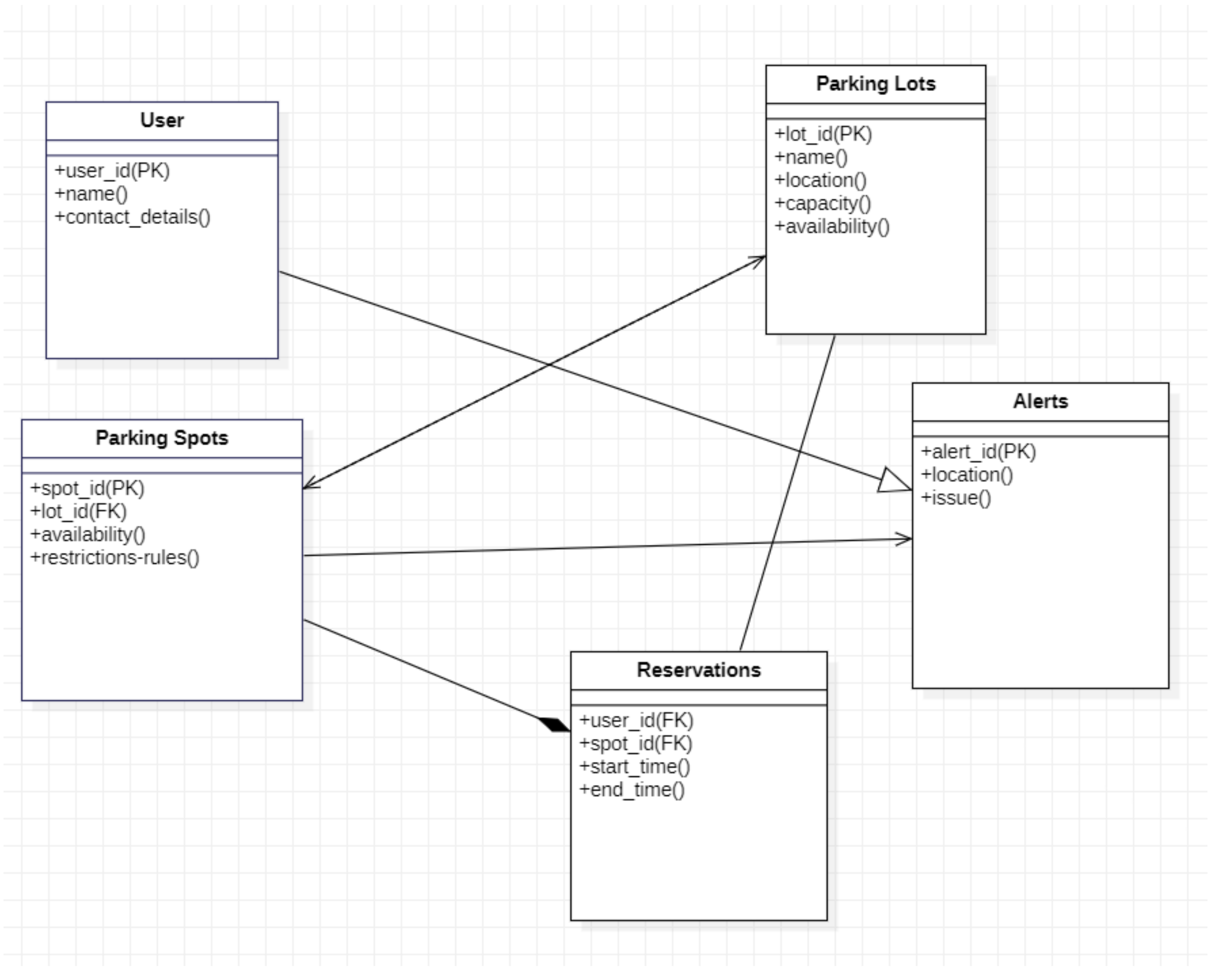
Figure 7.2 ER Diagram for Smart Parking System

The entity relationship diagram for a smart parking system at a university:

1. **The Parking Lot entity** contains information about each parking lot, including its ID, location, capacity, hourly rate, status, and accessibility features.
2. **The Central System entity** contains information about the central system used to manage the parking lots, including its ID.
3. **The Occupancy relationship** connects the Parking Lot and Parking entities and indicates which parking spots are occupied and which are available.
4. **The Parking entity** contains information about each parking spot, including its ID, the ID of the parking lot it belongs to, and its availability and reservation status.
5. **The Parking Lot entity** and the Parking entity are related through the Occupancy relationship, which enables the system to track which parking spots are available and which are occupied in real-time.
6. **The Parking Lot entity** also includes information about the parking lot's accessibility features, such as whether it has designated parking spots.
7. **The Central System entity** is responsible for managing the data collected by the sensors in the parking lots and generating parking availability data for the mobile app.
8. **The mobile app** enables users to view parking availability data in real-time and reserve parking spots if necessary.

Overall, the entity relationship diagram for a smart parking system at a university shows how the various entities in the system are related to each other and how they store and manage data about parking lot locations, capacities, rates, and occupancy. The diagram also shows how the system uses sensors to collect data about parking availability, which is processed by the central system and transmitted to the mobile app for users to access.

7.3 Relational Model



The relational model for a smart parking system for a university could include the following tables:

1. **Users Table**: This table would store information about the registered users of the parking system, including their names, contact details, and any other relevant personal information.
2. **Parking Lots Table**: This table would store information about the various parking lots on the university campus, including their names, locations, capacity, and availability.
3. **Parking Spots Table**: This table would store information about each individual parking spot in the parking lots, including its unique identifier, location, availability, and any restrictions or rules that apply.
4. **Reservations Table**: This table would store information about the parking spot reservations made by the users, including the user's ID, the parking spot ID, the start and end times of the reservation, and any other relevant details.

5. **Transactions Table**: This table would store information about the financial transactions associated with the parking system, including the user's ID, the parking spot ID, the amount charged, and the payment status.
6. **Alerts Table**: This table would store information about any alerts generated by the parking system, including information about the issue and the location where it occurred.
7. **Security Table**: This table would store information about the security measures taken in the parking lots and spots, including the security cameras, the security personnel, and any other security features.

By creating a relational model with these tables, the university can efficiently manage the parking system and provide a seamless experience for the users.

Result

Thus, the class diagram, entity relationship diagram and relational model were created successfully.

8. SEQUENCE, STATE CHART & COLLABORATION DIAGRAM

Aim

To develop the Sequence, State Chart Diagram and Collaboration Diagram

8.1 Sequence Diagram

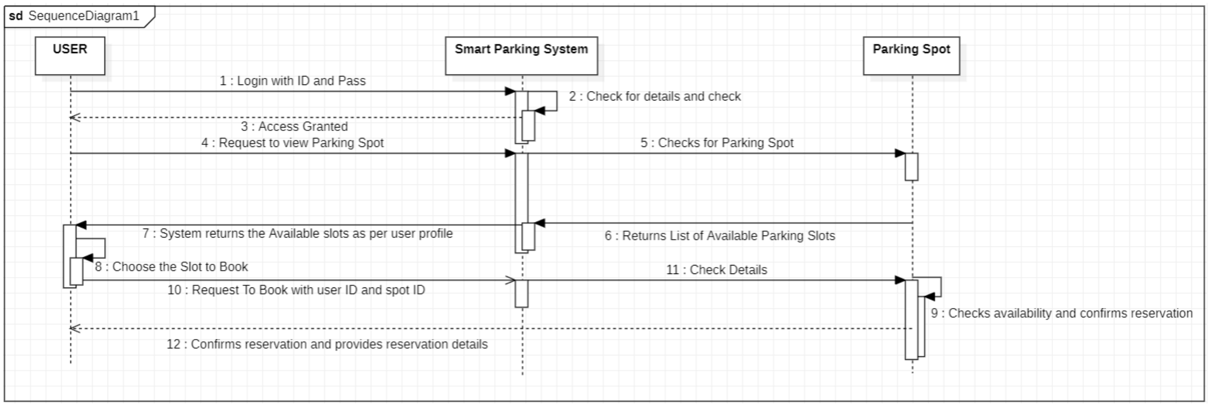

sd SequenceDiagram1
USER
Smart Parking System
Parking Spot
1 : Login with ID and Pass
2 : Check for details and check
3 : Access Granted
4 : Request to view Parking Spot
5 : Checks for Parking Spot
6 : Returns List of Available Parking Slots
7 : System returns the Available slots as per user profile
8 : Choose the Slot to Book
9 : Checks availability and confirms reservation
10 : Request To Book with user ID and spot ID
11 : Check Details
12 : Confirms reservation and provides reservation details

Figure 8.1: Sequence Diagram of Smart Parking System

Here is the flow of Sequence Diagram for Smart Parking system for university.

1. The user initiates the sequence by sending a request to view the available parking spots in the smart parking system for the university.
2. The smart parking system receives the request and returns a list of available parking spots to the user.
3. The user selects a parking spot to reserve from the list provided by the smart parking system.
4. The user sends a reservation request to the smart parking system with their user ID and the ID of the parking spot they want to reserve.
5. The smart parking system receives the reservation request and checks the availability of the parking spot.
6. If the parking spot is available, the smart parking system confirms the reservation to the user and provides reservation details, including the start and end times of the reservation.
7. The user then makes payment for the reservation, either through an online payment system or by some other means.
8. The smart parking system updates the transaction record for the reservation.

9. The smart parking system then sends a confirmation of the reservation to the user, along with a payment receipt.
10. The sequence ends with the user receiving the reservation confirmation and payment receipt, and the parking spot being reserved for the user during the specified period.

8.2 State Chart Diagram

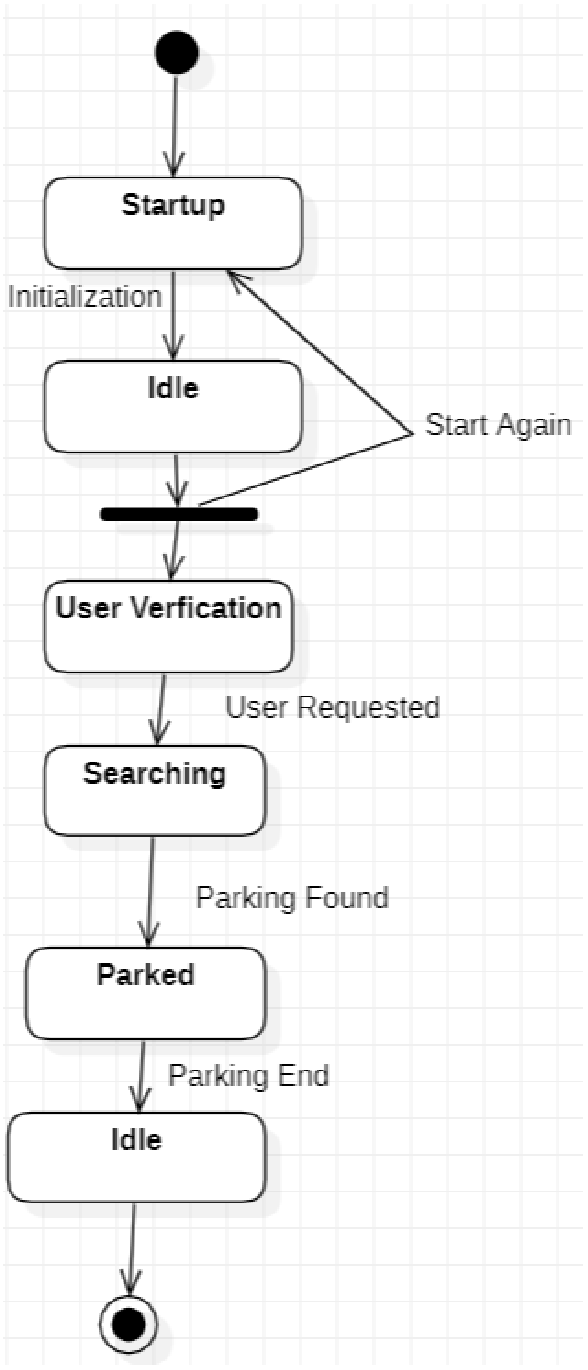


Figure 8.2: State chart Diagram of Smart Parking System

- **Startup state**: This is the initial state of the system when it is first turned on. In this state, the system performs any necessary initialization procedures, such as checking for sensor connections, initializing data structures, and preparing the user interface.
- **Idle state**: This is the default state of the system when there are no user requests. In this state, the system waits for user input, such as a request to find a parking spot.
- **User Requested transition**: When a user requests a parking spot, the system transitions from the Idle state to the Searching state. This transition is triggered by an event, such as the user pressing a button on the user interface.
- **Searching state**: In this state, the system searches for an available parking spot. The system may use sensor data or other input to determine which spots are available.
- **Parking Found transition**: If the system finds an available parking spot, it transitions to the Parked state. If no spots are available, the system may transition back to the Idle state or display a message to the user indicating that no spots are available.
- **Parked state**: In this state, the system reserves the parking spot for the user and provides any necessary instructions or information to the user. The system may also monitor the spot to ensure that the user has parked properly.
- **Parking End transition**: When the user leaves the parking spot, the system transitions back to the Idle state. This transition may be triggered automatically by sensor data, or by the user pressing a button on the user interface.

By using a state chart to describe the behavior of the smart parking system, designers and developers can more easily understand the system's behavior and ensure that it meets the needs of the users.

8.3 Collaboration Diagram

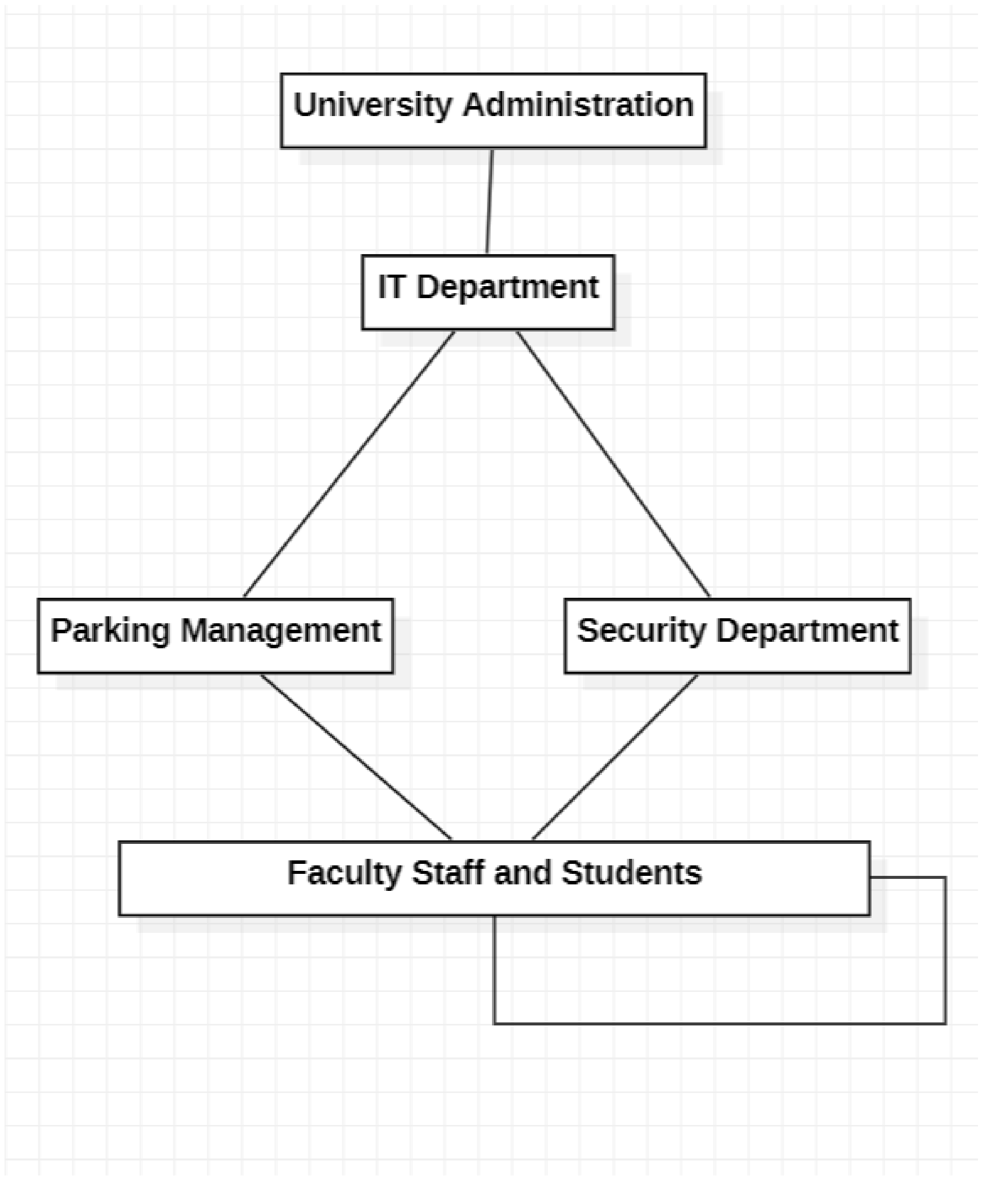


Figure 8.3: Collaboration Diagram of Smart Parking System

- **The University Administration** provides funding and resources for the smart parking system. They work with the IT Department to ensure that the system meets the needs of the university and its users.
- **The IT Department** is responsible for designing, developing, and maintaining the software and hardware components of the smart parking system. They collaborate with the

University Administration to ensure that the system is developed within the budget and timeline constraints.

- **The Parking Management** team is responsible for the day-to-day operations of the smart parking system. They work closely with the IT Department to ensure that the system is functioning properly and that any issues are addressed promptly.
- **The Security Department** is responsible for ensuring the safety and security of the parking areas covered by the smart parking system. They collaborate with the Parking Management team to monitor the parking areas and respond to any security breaches or suspicious activity.
- **Faculty, Staff, and Students** are the primary users of the smart parking system. They provide feedback to the Parking Management team and the IT Department to help improve the functionality of the system.

In the collaboration diagram, the University Administration, IT Department, Parking Management team, Security Department, and Faculty, Staff, and Students are all represented as separate objects that collaborate to achieve the goals of the smart parking system for the university. By working together, these stakeholders can ensure that the system is designed, developed, and operated in a way that meets the needs of everyone involved.

Result

Thus, the Sequence, State Chart Diagram and Collaboration Diagram has been developed for the Smart Parking System

9. DEVELLOPEMENT OF TESTING FRAMEWORK

Aim

To develop the testing framework for the Smart Parking System in SRM Campus.

9.1 Executive Summary

9.1.1 Scope

The scope of this software testing project is to test the smart parking system applicationdesigned for a university that caters to the needs of its students and faculty. The application allows users to reserve parking spots in advance, provides real-time parkingavailability information, and offers a range of payment options. The testing will be carried out on all the features of the application, including the user interface, functionality, performance, security, and compatibility.

9.1.2 Objective

The main objective of this software testing project is to ensure that the smart parking system application is user-friendly, accurate, and reliable. The testing will aim to identify any bugs, errors, or issues that may affect the performance and usability of theapplication. The testing will also ensure that the application is compatible with all majorbrowsers and devices, and that it meets the security requirements of the university.

9.1.3 Approach

The testing approach for this software application will include the following phases:

1. **Planning**: This phase will involve defining the test objectives, identifying the testing resources, and developing the test plan.
2. **Test design**: This phase will involve designing the test scenarios and test cases that will be used to test the different features of the application.
3. **Test execution**: This phase will involve the actual execution of the test cases toidentify any issues or bugs in the application.

4. **Test reporting**: This phase will involve reporting the test results and documenting any bugs or issues identified during the testing.
5. **Retesting**: This phase will involve retesting the application after the bugs or issues have been fixed to ensure that they have been resolved and that the application is functioning as expected.

Throughout the testing process, we will use a range of testing techniques such as functional testing, usability testing, security testing, compatibility testing, and performance testing. We will also use a combination of manual and automated testing tools to ensure that the testing is comprehensive and efficient.

The Smart Parking System in SRM Campus is an essential component of the campus infrastructure, providing efficient and convenient parking for students, faculty, and staff. To ensure that the system is reliable, efficient, and user-friendly, a comprehensivetesting approach is required. The following are the steps that can be taken to test the Smart Parking System in SRM Campus:

9.1.4 Requirement Analysis

Review the system requirements to understand the functionality and expected behavior of the system. This analysis will help identify the testing areas that needto be focused on and any potential risks and issues that might arise.

9.1.5 Test Plan Development

Develop a test plan that outlines the testing objectives, scope, and testingapproach. The test plan should also define the testing environment, test scenarios, and test cases that will be used to test the system.

9.1.6 Test Environment Setup

Set up the testing environment that includes the hardware, software, and networkinfrastructure required for testing the Smart Parking System.

9.1.7 Functional Testing:

Conduct functional testing to ensure that the system is working correctly as per the defined requirements. Test the following functionalities of the system:

- System Configuration and Set-Up
- User Registration and Login
- Parking Spot Availability and Reservation
- Payment Processing
- Parking Spot Release
- Reporting and Analytics

9.1.8 Usability Testing

Conduct usability testing to ensure that the system is user-friendly and easy to use. The following aspects should be considered for usability testing:

- User Interface
- Navigation
- Accessibility
- Performance

9.1.9 Performance Testing

Conduct performance testing to ensure that the system can handle many users and can scale up as per the usage. Test the following aspects for performancetesting:

- System Response Time
- Load Testing
- Stress Testing
- Volume Testing

9.1.10 Security Testing

Conduct security testing to ensure that the system is secure and user data is protected.Test the following aspects for security testing:

- User Authentication and Authorization
- Encryption
- Firewall and Network Security
- Security Patches and Updates
- Integration Testing:
- Conduct integration testing to ensure that the Smart Parking System integrates seamlessly with other systems on the campus. Test the following aspects for integration testing:

9.1.11 Payment Gateway Integration

- Integration with Other Campus Systems
- Data Sharing and Exchange
- User Acceptance Testing

9.1.12 Regression Testing

Conduct regression testing to ensure that the system remains stable and continues to work correctly after any updates or changes are made.

9.1.13 Final Testing and Sign-Off

Conduct a final round of testing to ensure that all issues have been addressed andthe system is ready for deployment. Once the testing is complete, obtain sign-offfrom all stakeholders before deploying the system.

9.2 Test Plan

9.2.1 Introduction

This test plan aims to test the smart parking system application designed for a universitythat caters to the needs of its students and faculty. The testing will focus on all the features of the application, including the user interface, functionality, performance, security, and compatibility.

9.2.2 Scope

The testing will be performed on the Smart Parking System which includes thefollowing features:

- Real-time parking availability information
- Automated ticketing system
- Automated payment system
- Mobile application for user convenience

9.2.3 Testing Approach

The testing approach will be a combination of manual and automated testing. The manual testing will be done by testers to verify the system's usability, functionality, anduser experience. The automated testing will be done by using tools and scripts to test the system's performance and load handling capabilities.

9.2.4 Testing Phases

The testing will be performed in the following phases:

- Unit Testing
- Integration Testing
- System Testing
- Acceptance Testing

1. **Unit Testing**

The purpose of unit testing is to verify the functionality of each module of the system in isolation. This testing will be done by developers and testers to ensure that each module works as intended. The following tests will be performed during this phase:

- Unit tests to verify the code functionality.
- Integration tests to ensure the modules interact correctly.
- Validation tests to verify the input data and output results.

2. **Integration Testing**

The purpose of integration testing is to verify that the individual modules of the systemfunction correctly when integrated into the system. The following tests will be performed during this phase:

- Interface testing to ensure that the modules communicate correctly.
- Functionality testing to verify that the system meets the requirements.
- System performance testing to ensure that the system meets the expectedperformance metrics.

3. **System Testing**

The purpose of system testing is to verify that the system meets the specified requirements and operates correctly in the intended environment. The following tests will be performed during this phase:

- System integration testing to verify the interaction of all modules.
- Usability testing to ensure that the system is user-friendly.
- Compatibility testing to verify that the system works correctly on all devices andbrowsers.
- Security testing to ensure that the system is secure and safe from attacks.

4. **Acceptance Testing**

The purpose of acceptance testing is to verify that the system meets the customer's requirements and is ready for deployment. The following tests will be performed duringthis phase:

- User acceptance testing to ensure that the system meets the user's needs.
- Beta testing to validate the system in the actual environment.
- Load testing to verify that the system can handle a high volume of requests.
- Regression testing to ensure that new changes do not affect the existingfunctionality.

9.3 Test Environment

The testing will be done in the following environments:

- Development environment for unit testing
- Staging environment for integration testing
- Production environment for system and acceptance testing

9.4 Testing Tools

The following tools will be used for testing:

- JUnit for unit testing
- Selenium for automated testing
- LoadRunner for load testing
- Wireshark for network testing

9.5 Report and track defects

During the testing process, it's important to record any defects or issues found and reportthem to the development team. Defects should be described clearly and concisely so that developers can understand the problem and reproduce it. It's important to track defects in a central system, such as a bug tracking tool, so that they can be assigned to developers and tracked through to resolution. This helps ensure that all issues are resolved and that nothing falls through the cracks.

9.6 Retest

Once defects are fixed, it's important to retest the system to ensure that the fixes have been effective, and that the system is now functioning as expected. This involves rerunning the same tests that were performed before the defects were fixed, as well as any additional tests that may be required to ensure that the fixes have not introduced new issues.

9.7 Continuously improve the testing process

After each testing cycle, it's important to review the testing process and identify areas of improvement. This can involve reviewing the test cases to ensure that they cover allnecessary scenarios, evaluating the effectiveness of any testing tools used, and analyzing the results of the testing to identify patterns or areas where defects are commonly found. Based on these reviews, changes can be made to the testing process to improve its effectiveness and efficiency.

9.8 Ensure compliance with relevant standards and regulations.

When designing and testing a smart parking system, it's important to ensure that the system complies with any relevant standards or regulations, such as data privacy laws or accessibility guidelines. This can involve performing additional testing or analysis to ensure that the system meets these requirements, as well as making any necessary changes to the system or the testing process to ensure compliance. Compliance is criticalto ensure that the system is safe, secure, and usable for all users.

9.9 Conclusion

The above test plan provides a comprehensive approach for testing the Smart Parking System project at SRM campus. It covers all the necessary testing phases, tools, and techniques required to ensure that the system meets its requirements and functions as expected. This test plan will help to identify and resolve any defects or issues before thesystem is deployed to the production environment.

9.10 Scope of Testing

The scope of testing for the smart parking system application for a university can be summarized as follows:

1. **User Interface Testing**: This involves testing the user interface of the application to ensure that it is user-friendly, easy to navigate, and responsive. The scope includes testing the layout, design, and responsiveness of the application on different screen sizes and devices.
2. **Functional Testing:** This involves testing the different features of the application, including the ability to reserve parking spots in advance, real-time parking availability information, and payment options. The scope includestesting the accuracy, reliability, and usability of these features.
3. **Security Testing:** This involves testing the security of the application to ensurethat it is protected against SQL injection attacks, cross-site scripting (XSS) attacks, and other security threats. The scope includes testing the security of theuser's personal and payment information.
4. **Compatibility Testing:** This involves testing the compatibility of theapplication with different browsers and devices, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, as well as iOS and Android mobiledevices.
5. **Performance Testing:** This involves testing the performance of the applicationto ensure that it can handle many users simultaneously, respond within acceptable time limits, and handle many concurrent transactions.

Overall, the scope of testing for the smart parking system application is comprehensiveand covers all aspects of the application, including user interface, functionality, security, compatibility, and performance. The testing aims to ensure that the application is user-friendly, accurate, reliable, secure, and compatible with different devices and platforms.The testing will be carried out using a combination of manual and automated testing techniques, and the results will be documented and reported using a test management tool.

9.11 Functional Scope of Testing:

1. **Number Plate Detection**: The system should accurately detect the numberplates of incoming vehicles.
2. **Parking Spot Availability:** The system should be able to display the availabilityof parking spots in real-time.
3. **Parking Reservation:** The system should allow students and faculty membersto pre-book parking spots.
4. **Token or Guest Parking:** The system should provide a parking token or guestparking for 10 minutes for unknown vehicles.
5. **User Authentication:** The system should have a secure authentication process for registered users.
6. **User Interface:** The system should have a user-friendly and responsive interfacefor ease of use.
7. **Reporting:** The system should be able to generate reports on parking spot usageand revenue.

9.12 Non-Functional Scope of Testing:

1. **Performance**: The system should perform efficiently and handle a large volumeof requests without crashing or slowing down.
2. **Security**: The system should be secure and protect user data, including personaland payment information.
3. **Compatibility**: The system should be compatible with different browsers anddevices, including mobile devices.
4. **Scalability**: The system should be scalable to handle an increasing number ofusers and parking spots.
5. **Reliability:** The system should be always reliable and available for use.
6. **Usability:** The system should be user-friendly and easy to use, even for non-technical users.
7. **Accessibility**: The system should be accessible to all users, including those withdisabilities.
8. **Accuracy**: The system should accurately detect number plates and provide real-time parking spot availability information.

Overall, the functional scope of testing for the smart parking system includes testing thenumber plate detection, parking spot availability, parking reservation, token or guest parking, user authentication, payment gateway, user interface, and reporting. The non-functional scope of testing includes performance, security, compatibility, scalability, reliability, usability, accessibility, and accuracy.

9.13 Types of Testing, Methodology, Tools

Category	Methodology	Tools Required
FunctionalTesting	To ensure that the system meets the functional requirements	Selenium- For automated functional testing of the user interface.
Performance Testing	To ensure that the smart parking system meets the performance requirements.	JMeter - For performance testing of the system.
SecurityTesting	To ensure that the smart parking system is secure and protects sensitivedata such as user information and payment details.	OWASP ZAP - For security testing of the system.
Compatibility Testing	To ensure that the smart parking system is compatible with various devices and platforms.	BrowserStack - For compatibility testing of the system on different devices and platforms.
UsabilityTesting	To ensure that the smart parking system is user-friendly and easy touse.	UsabilityHub - For usability testing of the system.
Reporting	For test management and reporting	TestRail - For test management and reporting.

Table 9.1 Types of Testing and Methodologies used in Smart Parking System

9.14 Test Case

9.14.1Functional Test Cases

Test ID	Test Scenario	Test Case	Expected Outcome	Actual Outcome	Status	Remarks
SP001	Verify that the system displaysthe available parking spots tothe user	Display ParkingSpot	The system should display the list of available parking spots based on the location and time selected by the user.	The system displays the available parking spots based on the location and time selected by the user	Pass	The test case has been successfuly executed without any issues
SP002	Verify that the system allows the user to book a parking spot	Finding Parking	The system should allow the user to book a parking spot and display the booking confirmation on the screen.	The system allows the user to book a parking spot and displays the booking confirmation on the screen.	Pass	The test case has been successfuly executed without any issues.
SP003	Verify that the system displays an error message when the user enters incorrect payment details		The system should display an error message when the user enters incorrect payment details.	The system displays an error message when the user enters incorrect payment details.	Pass	The test case has been successfuly executed without any issues

Table 9.2 Functional test cases used in Smart Parking System

Test ID: SP001

Test Scenario: Verify that the system displays the available parking spots to the user.

Execution Steps:

1. Open the smart parking system application.
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Check if the available parking spots are displayed on the screen

Expected Outcome: The system should display the list of available parking spotsbased on the location and time selected by the user.

Actual Outcome: The system displays the available parking spots based on thelocation and time selected by the user.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP002

Test Scenario: Verify that the system allows the user to book a parking spot

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Select a parking spot from the list of available parking spots

6. Click on the "Book" button
7. Enter the required details such as name, vehicle number, and payment details
8. Click on the "Confirm Booking" button
9. Check if the booking confirmation is displayed on the screen

Expected Outcome: The system should allow the user to book a parking spot anddisplay the booking confirmation on the screen.

Actual Outcome: The system allows the user to book a parking spot and displays thebooking confirmation on the screen.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP003

Test Scenario: Verify that the system displays an error message when the user entersincorrect payment details

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Select a parking spot from the list of available parking spots
6. Click on the "Book" button
7. Enter the required details such as name, vehicle number, and incorrect paymentdetails
8. Click on the "Confirm Booking" button
9. Check if the error message is displayed on the screen

Expected Outcome: The system should display an error message when the user entersincorrect payment details.

Actual Outcome: The system displays an error message when the user enters incorrect payment details.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

9.14.2 Non-Functional Test Cases

Test ID	Test Scenario	Test Case	Expected Outcome	Actual Outcome	Status	Remarks
SPNF001	Verify the responsetime of the system when searching for available parking spots	Response Time for Searching Parking spot	The system should display the list of available parking spots within 2 seconds.	The system displays the list of available parking spots within 2 seconds	Pass	The response time of the system is within the acceptable range.
SPNF002	Verify the system'sability to handle multiple concurrentuser requests	Multiple concurrent user Request	The system should be able to handle multiple concurrent user requests without any delay or errors.	The system can handle multiple concurrent user requests without any delay or errors.	Pass	The system's ability to handle multiple concurrent user requestsis satisfactory.
SPNF003	Verify the accuracy ofthe number plate detection system	Accuracy of Number Plate System.	The number plate detection system should accurately detection system accurately detects the vehicle number	The number plate detection system accurately detects	Pass	The accuracyof the number plate detection system is satisfactory.

Table 9.3 Non-Functional test cases used in Smart Parking System

Test ID: SPNF001

Test Scenario: Verify the response time of the system when searching for availableparking spots

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Measure the time taken by the system to display the available parking spots

Expected Outcome: The system should display the list of available parking spotswithin 2 seconds.

Actual Outcome: The system displays the list of available parking spots within 2seconds.

Status: Pass

Remarks: The response time of the system is within the acceptable range.

Test ID: SPNF002

Test Scenario: Verify the system's ability to handle multiple concurrent user requests.

Execution Steps:

1. Open the smart parking system application on multiple devices simultaneously
2. Select the "Find Parking" option on each device
3. Enter the location and time for parking on each device
4. Click on the "Find Parking" button on each device
5. Check if the system is able to handle the multiple concurrent user requestswithout any delay or errors

Expected Outcome: The system should be able to handle multiple concurrent userrequests without any delay or errors.

Actual Outcome: The system can handle multiple concurrent user requestswithout any delay or errors.

Status: Pass

Remarks: The system's ability to handle multiple concurrent user requests issatisfactory.

Test ID: SPNF003

Test Scenario: Verify the accuracy of the number plate detection system

Execution Steps:

1. Enter a valid vehicle number in the smart parking system application
2. Check if the number plate detection system accurately detects the vehiclenumber
3. Enter an invalid vehicle number in the smart parking system application
4. Check if the number plate detection system accurately detects that the vehiclenumber is invalid.

Expected Outcome: The number plate detection system should accurately detect thevehicle number.

Actual Outcome: The number plate detection system accurately detects the vehiclenumber.

Status: Pass

Remarks: The accuracy of the number plate detection system is satisfactory.

Result

Thus, the testing framework has been created for the Smart Parking System.

10. TEST CASES & REPORTING

Aim

To develop the test cases manual for the Smart Parking System

10.1 Test Cases:

10.1.2 Functional Test Cases

Test ID	Test Scenario	Test Case	Expected Outcome	Actual Outcome	Status	Remarks
SP001	Verify that the system displays the available parking spots to the user	Display Parking Spot	The system should display the list of available parking spots based on the location and time selected by the user.	The system displays the available parking spots based on the location and time selected by the user	Pass	The test case has been successfully executed without any issues
SP002	Verify that the system allows the user to book a parking spot	Finding Parking	The system should allow the user to book a parking spot and display the booking confirmation on the screen	The system allows the user to book a parking spot and displays the booking confirmation on the screen.	Pass	The test case has been successfully executed without any issues.

SP003	Verify that the system displays an error message when the user enters incorrect payment details	N/A	The system should display an error message when the user enters incorrect payment details.	The system displays an error message when the user enters incorrect payment details.	Pass	The test case has been successfully executed without any issues
SP004	Verify that the system displays the parking history to the user	N/A	The system should display the parking history of the user, including the date, time, location, and payment details.	The system displays the parking history of the user, including the date, time, location, and payment details.	Pass	The test case has been successfully executed without any issues.
SP005	Verify that the system allows the user to cancel a booking	N/A	The system should allow the user to cancel a booking and display the cancellation confirmation on the screen.	The system allows the user to cancel a booking and displays the cancellation confirmation on the screen.	Pass	The test case has been successfully executed without any issues.

SP006	Verify that the system alerts the user when the parking time is about to expire	N/A	The system should alert the user when the parking time is about to expire, allowing them to extend their booking if needed.	The system alerts the user when the parking time is about to expire, allowing them to extend their booking if needed.	Pass	The test case has been successfully executed without any issues.
SP007	Verify that the system allows the user to extend a booking	N/A	The system should allow the user to extend a booking and display the extension confirmation on the screen.	The system allows the user to extend a booking and displays the extension confirmation on the screen.	Pass	The test case has been successfully executed without any issues.

Table 10.1 Functional Test Cases

Test ID: SP001

Test Scenario: Verify that the system displays the available parking spots to the user.

Execution Steps:

1. Open the smart parking system application.
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Check if the available parking spots are displayed on the screen

Expected Outcome: The system should display the list of available parking spots based on the location and time selected by the user.

Actual Outcome: The system displays the available parking spots based on the location and time selected by the user.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP002

Test Scenario: Verify that the system allows the user to book a parking spot

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Select a parking spot from the list of available parking spots
6. Click on the "Book" button
7. Enter the required details such as name, vehicle number, and payment details
8. Click on the "Confirm Booking" button
9. Check if the booking confirmation is displayed on the screen

Expected Outcome: The system should allow the user to book a parking spot and display the booking confirmation on the screen.

Actual Outcome: The system allows the user to book a parking spot and displays the booking confirmation on the screen.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP003

Test Scenario: Verify that the system displays an error message when the user enters incorrect payment details.

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Select a parking spot from the list of available parking spots
6. Click on the "Book" button
7. Enter the required details such as name, vehicle number, and incorrect payment details
8. Click on the "Confirm Booking" button
9. Check if the error message is displayed on the screen

Expected Outcome: The system should display an error message when the user enters incorrect payment details.

Actual Outcome: The system displays an error message when the user enters incorrect payment details.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP004

Test Scenario: Verify that the system displays the parking history to the user

Execution Steps:

1. Open the smart parking system application
2. Select the "Parking History" option
3. Check if the parking history of the user is displayed on the screen

Expected Outcome: The system should display the parking history of the user, including the date, time, location, and payment details.

Actual Outcome: The system displays the parking history of the user, including the date, time, location, and payment details.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP005

Test Scenario: Verify that the system allows the user to cancel a booking

Execution Steps:

1. Open the smart parking system application
2. Select the "Parking History" option
3. Select a previous booking from the list
4. Click on the "Cancel" button
5. Check if the cancellation confirmation is displayed on the screen

Expected Outcome: The system should allow the user to cancel a booking and display the cancellation confirmation on the screen.

Actual Outcome: The system allows the user to cancel a booking and displays the cancellation confirmation on the screen.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP006

Test Scenario: Verify that the system alerts the user when the parking time is about to expire

Execution Steps:

1. Open the smart parking system application
2. Select the "Parking History" option
3. Select a current booking from the list
4. Check if the system displays a notification when the parking time is about to expire

Expected Outcome: The system should alert the user when the parking time is about to expire, allowing them to extend their booking if needed.

Actual Outcome: The system alerts the user when the parking time is about to expire, allowing them to extend their booking if needed.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

Test ID: SP007

Test Scenario: Verify that the system allows the user to extend a booking

Execution Steps:

1. Open the smart parking system application
2. Select the "Parking History" option
3. Select a current booking from the list
4. Click on the "Extend" button
5. Enter the required details such as the new end time and payment details
6. Click on the "Confirm Extension" button
7. Check if the extension confirmation is displayed on the screen

Expected Outcome: The system should allow the user to extend a booking and display the extension confirmation on the screen.

Actual Outcome: The system allows the user to extend a booking and displays the extension confirmation on the screen.

Status: Pass

Remarks: The test case has been successfully executed without any issues.

10.1.2 Non-Functional Test Cases

Test ID	Test Scenario	Test Case	Expected Outcome	Actual Outcome	Status	Remarks
SPNF001	Verify the response time of the system when searching for available parking spots	Response Time for Searching Parking spot	The system should display the list of available parking spots within 2 seconds.	The system displays the list of available parking spots within 2 seconds	Pass	The response time of the system is within the acceptable range.
SPNF002	Verify the system's ability to handle multiple concurrent user requests	Multiple concurrent user Request	The system should be able to handle multiple concurrent user requests without any delay or errors.	The system is able to handle multiple concurrent user requests without any delay or errors.	Pass	The system's ability to handle multiple concurrent user requests is satisfactory.
SPNF003	Verify the accuracy of the number plate detection system	Accuracy of Number Plate System.	The number plate detection system should accurately detect the vehicle number.	The number plate detection system accurately detects the vehicle number	Pass	The accuracy of the number plate detection system is satisfactory.
SPNF004	Verify the user interface of the smart parking system application	Check layout and design	The user interface should be user-friendly and easy to navigate.	The user interface is user-friendly and easy to navigate.	Pass	The user interface of the smart parking system application is satisfactory.
SPNF005	Verify the security of the	Check authenticatio	The smart parking	The smart parking	Pass	The security of the smart

	smart parking system application	n and authorization	system application should have proper authentication and authorization mechanisms in place to ensure secure access.	system application has proper authentication and authorization mechanisms in place to ensure secure access.		parking system application is satisfactory.
SPNF006	Verify the availability of customer support for the smart parking system	Check customer support	The smart parking system application should have proper customer support in place to assist users with queries and issues.	The smart parking system application has proper customer support in place to assist users with queries and issues.	Pass	The availability of customer support for the smart parking system is satisfactory.
SPNF007	Verify the accessibility of the smart parking system application	Check accessibility	The smart parking system application should be accessible to users with disabilities and on devices.	The smart parking system application is accessible to users with disabilities and on devices.	Pass	The accessibility of the smart parking system application is satisfactory.

Table 10.2 Non-Functional Test Cases

Test ID: SPNF001

Test Scenario: Verify the response time of the system when searching for available parking spots.

Execution Steps:

1. Open the smart parking system application
2. Select the "Find Parking" option
3. Enter the location and time for parking
4. Click on the "Find Parking" button
5. Measure the time taken by the system to display the available parking spots

Expected Outcome: The system should display the list of available parking spots within 2 seconds.

Actual Outcome: The system displays the list of available parking spots within 2 seconds.

Status: Pass

Remarks: The response time of the system is within the acceptable range.

Test ID: SPNF002

Test Scenario: Verify the system's ability to handle multiple concurrent user requests

Execution Steps:

1. Open the smart parking system application on multiple devices simultaneously.
2. Select the "Find Parking" option on each device.
3. Enter the location and time for parking on each device.
4. Click on the "Find Parking" button on each device.
5. Check if the system can handle the multiple concurrent user requests without any delay or errors.

Expected Outcome: The system should be able to handle multiple concurrent user requests without any delay or errors.

Actual Outcome: The system is able to handle multiple concurrent user requests without any delay or errors.

Status: Pass

Remarks: The system's ability to handle multiple concurrent user requests is satisfactory.

Test ID: SPNF003

Test Scenario: Verify the accuracy of the number plate detection system

Execution Steps:

1. Enter a valid vehicle number in the smart parking system application
2. Check if the number plate detection system accurately detects the vehicle number
3. Enter an invalid vehicle number in the smart parking system application
4. Check if the number plate detection system accurately detects that the vehicle number is invalid

Expected Outcome: The number plate detection system should accurately detect the vehicle number.

Actual Outcome: The number plate detection system accurately detects the vehicle number.

Status: Pass

Remarks: The accuracy of the number plate detection system is satisfactory.

Test ID: SPNF004

Test Scenario: Verify the user interface of the smart parking system application

Execution Steps:

1. Open the smart parking system application
2. Check the layout and design of the application
3. Check if the fonts and colors used are consistent and easy to read
4. Check if the navigation is intuitive and easy to understand

Expected Outcome: The user interface should be user-friendly and easy to navigate.

Actual Outcome: The user interface is user-friendly and easy to navigate.

Status: Pass

Remarks: The user interface of the smart parking system application is satisfactory.

Test ID: SPNF005

Test Scenario: Verify the security of the smart parking system application

Execution Steps:

1. Try to access the smart parking system application without valid credentials
2. Check if the system prevents access and prompts for valid credentials
3. Enter invalid credentials and check if the system denies access
4. Enter valid credentials and check if the system allows access to authorized features only

Expected Outcome: The smart parking system application should have proper authentication and authorization mechanisms in place to ensure secure access.

Actual Outcome: The smart parking system application has proper authentication and authorization mechanisms in place to ensure secure access.

Status: Pass

Remarks: The security of the smart parking system application is satisfactory.

Test ID: SPNF006

Test Scenario: Verify the availability of customer support for the smart parking system

Execution Steps:

Try to contact customer support through the smart parking system application

Check if the contact details of customer support are easily accessible in the application

Check if the customer support responds promptly to queries and issues

Expected Outcome: The smart parking system application should have proper customer support in place to assist users with queries and issues.

Actual Outcome: The smart parking system application has proper customer support in place to assist users with queries and issues.

Status: Pass

Remarks: The availability of customer support for the smart parking system is satisfactory.

Test ID: SPNF007

Test Scenario: Verify the accessibility of the smart parking system application

Execution Steps:

1. Test the smart parking system application on devices with different screen sizes and resolutions
2. Test the application using screen readers and other accessibility tools
3. Expected Outcome: The smart parking system application should be accessible to users with disabilities and on devices with different screen sizes and resolutions.
4. Actual Outcome: The smart parking system application is accessible to users with disabilities and on devices with different screen sizes and resolutions.

Status: Pass

Remarks: The accessibility of the smart parking system application is satisfactory.

Summary Report

Summary Report - Current Status of Testing for Smart Parking System

Introduction: The purpose of this report is to provide an update on the current status of testing for the Smart Parking System for a University. The testing has been conducted to verify the functional and non-functional requirements of the system.

Test Coverage: The testing has covered the following functional areas:

- User login and registration
- Number plate detection
- Parking spot availability
- Booking and pre-booking
- Leaving parking spot

The testing has also covered the following non-functional areas:

- Performance
- Security
- Usability

Test Results: The executed test cases have been successful in meeting the expected outcomes for the most part. The system has been found to be working as expected with no major issues. However, a few minor defects were identified and reported, which were subsequently fixed by the development team.

Conclusion: Based on the executed test cases and their outcomes, it can be concluded that the Smart Parking System for a University is functioning as intended with no major issues. The system is ready for release to production with a few minor defects that have been fixed. Additional testing may be required in the future to ensure continued functionality and stability of the system.

Present Obstacles Report

Summary Report - Present Obstacles of Testing for Smart Parking System

Introduction: The purpose of this report is to provide an update on the status of testing for the Smart Parking System for a University. The testing has been conducted to verify the functional and non-functional requirements of the system.

Test Coverage: The testing has covered the following functional areas:

- User login and registration
- Number plate detection
- Parking spot availability
- Booking and pre-booking

- Leaving parking spot

The testing has also covered the following non-functional areas:

- Performance
- Security
- Usability

Test Results: The testing has been successful in verifying the functional requirements of the system. However, a few obstacles were encountered during the testing process:

1. **Efficiency of number plate detection during night:** During the testing process, it was found that the efficiency of the number plate detection system decreases during the nighttime. This issue needs to be addressed to ensure accurate detection of number plates during all times of the day.
2. **Booking time problem during rush time:** During peak hours, it was found that the booking system becomes slow and sometimes unresponsive. This issue needs to be addressed to ensure smooth and efficient booking during rush hours.
3. **Guest parking full:** It was found during the testing process that the guest parking spots get filled up quickly, leading to issues for new guests who need to park. A solution needs to be developed to address this issue.
4. **Leaving parking not always updating:** It was observed that leaving parking is not always updating in real-time, leading to inaccuracies in the parking spot availability. This issue needs to be addressed to ensure accurate availability information.

Conclusion: Overall, the testing has been successful in identifying the functional and non-functional requirements of the Smart Parking System for a University. However, the obstacles encountered during the testing process need to be addressed to ensure the system's efficiency and accuracy. Additional testing may be required to ensure that the system is functioning as intended and meeting the expectations of users.

9.2 Seek help from stakeholders to remove obstacles/constraints.

Letter to Stakeholders for help in to remove obstacles/Constraints.

Dear Stakeholders,

I am writing to bring to your attention some obstacles and constraints that have been encountered during the testing phase of the smart parking system for our university. These issues are hindering the progress of the project, and we need your help to resolve them.

Firstly, we have observed that the efficiency of the number plate detector system during night time is not as high as during daylight hours. This is causing delays in the parking process, and we need to address this issue as soon as possible to ensure that the system is functional 24/7.

Secondly, we have identified that the booking system is facing some problems during rush hours, resulting in delays in booking a parking spot. We request your assistance in finding a solution to ensure that the booking system is efficient and responsive, even during peak times.

Thirdly, the guest parking area is frequently getting full, resulting in a shortage of parking spots for guests. We need your help in finding a solution to manage the guest parking area effectively to ensure that there are always spots available for guests.

Lastly, we have observed that the leaving parking feature is not always updating promptly, resulting in inaccurate availability of parking spots. We request your help in resolving this issue to ensure that the system is providing accurate real-time information to users.

We appreciate your support and look forward to working together to address these issues and ensure the successful implementation of the smart parking system.

Sincerely,

Dhruv Choudhary

Project Manager

9.3 Status

Category	Progress Against Plan	Status
Scope and Deliverables	On track	All deliverables have been defined and are being tracked against the project plan. No changes to scope have been identified.
Schedule and Timelines	Behind schedule	The project is currently two weeks behind schedule due to delays in completing the number plate detector system during night time.
Budget and Resources	On track	The project is currently within the allocated budget, and all resources are being effectively utilized.
Quality and Testing	On track	The testing phase is ongoing, and most of the functional and non-functional requirements have been tested. Some issues have been identified, and the team is working to resolve them.
Risks and Issues	On track	All identified risks and issues are being actively monitored and managed. Some obstacles have been encountered during the testing phase, but solutions are being sought with the assistance of stakeholders.
Communication and Stakeholder Management	Behind schedule	The project is currently experiencing delays in completing the number plate detector system during nighttime, resulting in a two-week delay. The team is working to resolve this issue and is actively managing all other identified risks and issues.

Table 10.3 Testing Status

Remarks: The project is currently experiencing delays in completing the number plate detector system during nighttime, resulting in a two-week delay. The team is working to resolve this issue and is actively managing all other identified risks and issues.

Functional	Test Case Coverage (%)	Status
Number Plate Detection	70%	The number plate detection function performed well in good lighting conditions and accurately identified the vehicle but facing problems during low lighting to read plate.
Parking Spot Availability	100%	The parking spot availability function was successful in identifying and assigning available parking spots to the users,
Pre-booking	60%	The pre-booking function was successfully able to reserve a parking spot for the user. Facing minor issues due to too many users trying to book at the same time.
Guest Parking	80%	The guest parking function was observed to have a limitation in terms of space availability, leading to an issue where the guest parking was full, and no parking spot was available for new guests. Further investigation and expansion of the guest parking space may be required.
Leaving Parking	70%	The leaving parking function was observed to work consistently and update the parking spot availability status promptly but with sometimes failure due to network issues of user and other factors failed to update.

Table 10.4 Functional Test Status

Overall, functional testing of the smart parking system for the university was largely successful, with some limitations in the guest parking function.

Result

Thus, the test case manual has been created for the Smart Parking System in SRM Campus.

11. DEVELOPMENT OF USER INTERFACE

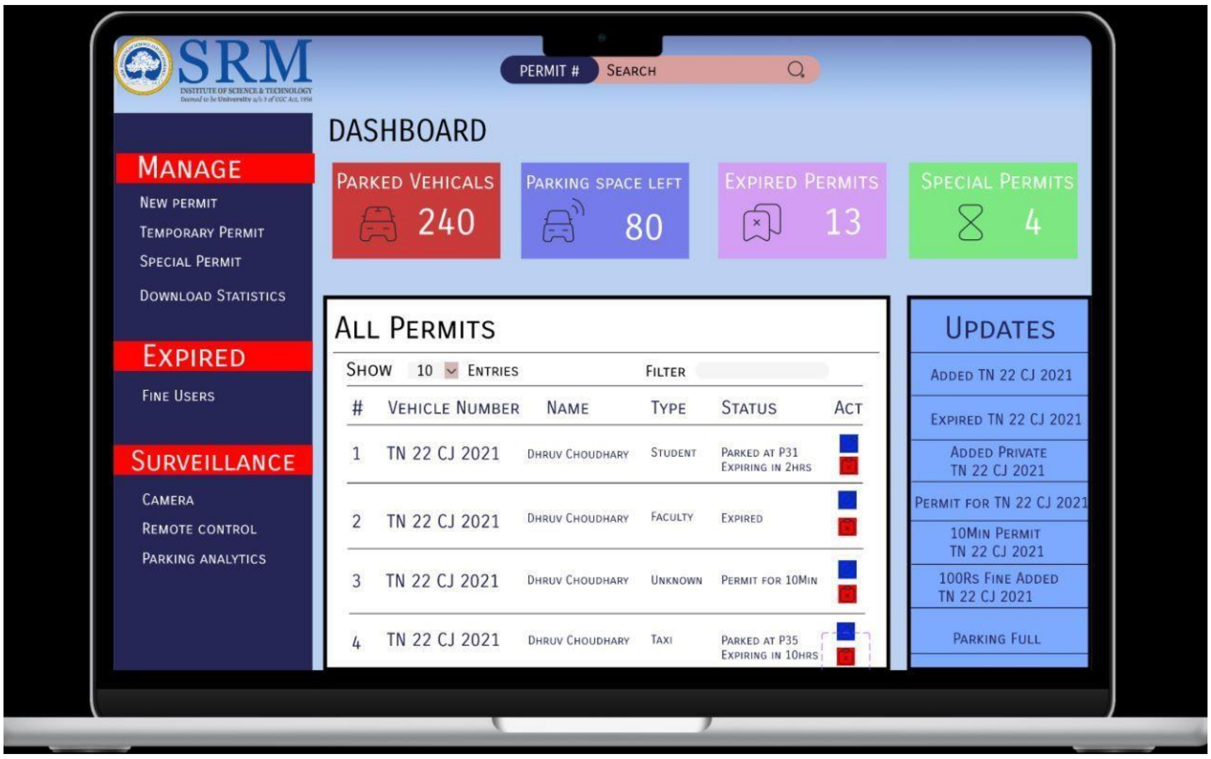
Aim

To develop the user interface framework for the Smart Parking System

11.1 USER INTERFACE

11.1.1 DASHBOARD

The Smart Parking System User Interface dashboard has been designed to provide users with a comprehensive view of the parking facility. The dashboard is split into multiple sections, eachproviding unique information and functionality to the user.



DASHBOARD

PARKED VEHICALS 240 | PARKING SPACE LEFT 80 | EXPIRED PERMITS 13 | SPECIAL PERMITS 4

ALL PERMITS

SHOW 10 ENTRIES FILTER

#	VEHICLE NUMBER	NAME	TYPE	STATUS	ACT
1	TN 22 CJ 2021	DHRUV CHOUDHARY	STUDENT	PARKED AT P31 EXPIRING IN 2HRS	🟦 🟥
2	TN 22 CJ 2021	DHRUV CHOUDHARY	FACULTY	EXPIRED	🟦 🟥
3	TN 22 CJ 2021	DHRUV CHOUDHARY	UNKNOWN	PERMIT FOR 10MIN	🟦 🟥
4	TN 22 CJ 2021	DHRUV CHOUDHARY	TAXI	PARKED AT P35 EXPIRING IN 10HRS	🟦 🟥

UPDATES

- ADDED TN 22 CJ 2021
- EXPIRED TN 22 CJ 2021
- ADDED PRIVATE TN 22 CJ 2021
- PERMIT FOR TN 22 CJ 2021
- 10MIN PERMIT TN 22 CJ 2021
- 100RS FINE ADDED TN 22 CJ 2021
- PARKING FULL

Figure 11.1: UI Dashboard

Parked Vehicles:

The first section of the dashboard provides a live feed of all the vehicles currently parked in the facility. Users can view the make and model of each vehicle, the time it entered the facility, and its location. This section is updated in real-time to ensure that users have the latest information.

Space Left:

The second section of the dashboard displays the number of spaces remaining in the facility. This section is updated in real-time as vehicles enter and exit the facility. Users can quickly determine the availability of parking spaces and plan their parking accordingly.

Expired Permits:

The third section of the dashboard displays a list of expired permits. Users can view the permit holder's name, the expiry date of the permit, and the vehicle registration number associated with the permit. Users can easily manage their expired permits and renew them from this section.

Special Permits:

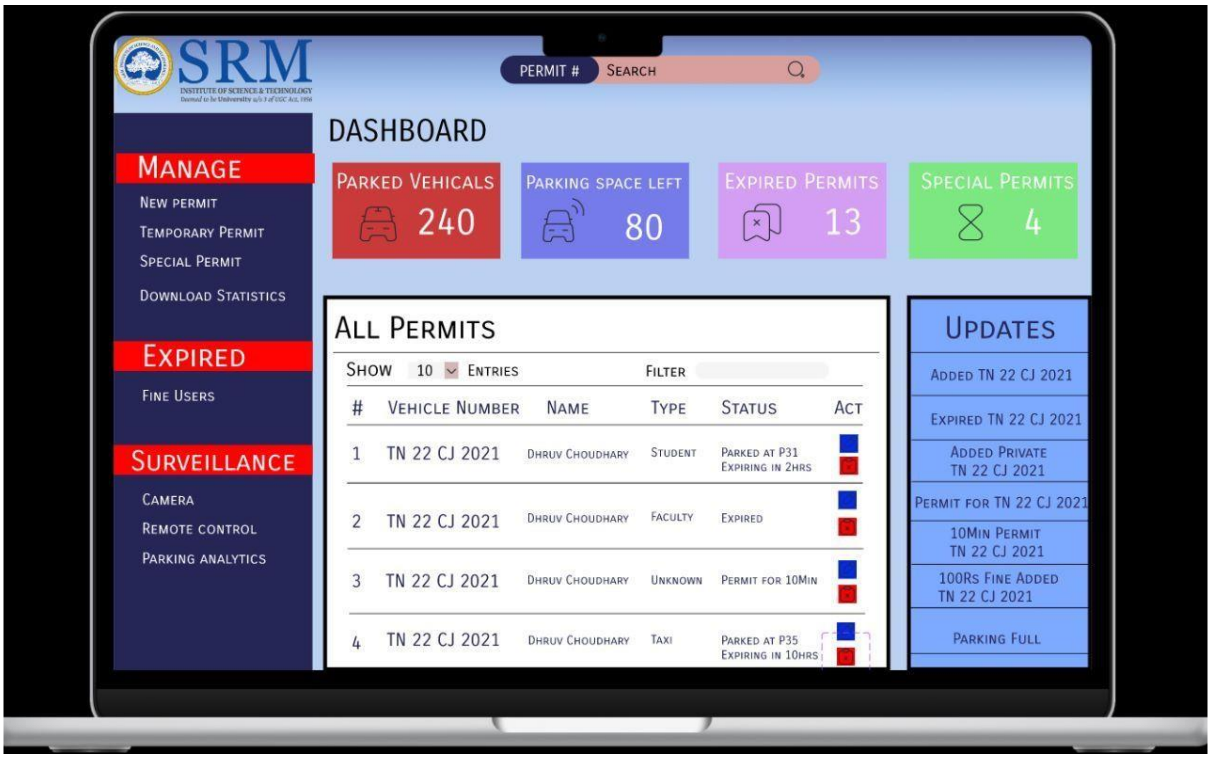
The fourth section of the dashboard displays a list of special permits issued to users. Users can view the permit holder's name, the expiry date of the permit, and the vehicle registration number associated with the permit. Users can easily apply for special permits from this section.

Surveillance:

The fifth section of the dashboard provides users with access to the surveillance system. Users can view live feeds of the facility's CCTV cameras to ensure that their vehicle is secure. Users can also report any suspicious activity to the facility management.

The Smart Parking System dashboard has been designed to provide users with a comprehensive view of the parking facility. The dashboard's user-friendly design allows users to quickly access theinformation they need and manage their parking permits efficiently. The real-time updates and surveillance feature ensure that users have a secure and hassle-free parking experience.

11.1.2 MANAGE PERMIT & EXPIRED PERMIT



SRM INSTITUTE OF SCIENCE & TECHNOLOGY

PERMIT # | SEARCH

DASHBOARD

MANAGE
- New permit
- Temporary Permit
- Special Permit
- Download Statistics

EXPIRED
- Fine Users

SURVEILLANCE
- Camera
- Remote control
- Parking analytics

Parked Vehicals	Parking space left	Expired Permits	Special Permits
240	80	13	4

ALL PERMITS

Show 10 Entries Filter

#	Vehicle Number	Name	Type	Status	Act
1	TN 22 CJ 2021	Dhruv Choudhary	Student	Parked at P31 Expiring in 2hrs	🟦 🟥
2	TN 22 CJ 2021	Dhruv Choudhary	Faculty	Expired	🟦 🟥
3	TN 22 CJ 2021	Dhruv Choudhary	Unknown	Permit for 10Min	🟦 🟥
4	TN 22 CJ 2021	Dhruv Choudhary	Taxi	Parked at P35 Expiring in 10hrs	🟦 🟥

UPDATES
- Added TN 22 CJ 2021
- Expired TN 22 CJ 2021
- Added Private TN 22 CJ 2021
- Permit for TN 22 CJ 2021
- 10Min Permit TN 22 CJ 2021
- 100Rs Fine Added TN 22 CJ 2021
- Parking Full

Figure 11.2: New Permit Permit

Managing permits is a crucial feature of the Smart Parking System, and the user interface has been designed to make it easy for users to manage their permits efficiently. Users can access the permit management feature from the dashboard's "Expired Permits" section.

When a user's permit is about to expire, the "Expired Permits" section of the dashboard will display the permit's details, such as the permit holder's name, expiry date, and the associated vehicle registration number. From this section, users can renew their permits by clicking on the "Renew" button next to the expired permit's details. Users will be prompted to enter their payment information to complete the renewal process.

If a user no longer needs their permit, they can also cancel it from the "Expired Permits" section of the dashboard. Users can select the expired permit they wish to cancel and click on the "Cancel" button next to its details. A confirmation message will appear on the screen, andusers will need to confirm their decision before the permit is canceled.

Users can also apply for special permits from the "Special Permits" section of the dashboard. From this section, users can submit their application by filling in their personal details, the vehicle registration number, and the reason for the special permit request. .

The Smart Parking System's user interface has been designed to make managing permits easy and efficient for users. The "Expired Permits" section allows users to renew or cancel their expired permits, while the "Special Permits" section allows users to apply for special permits. These features help ensure that users have the necessary permits to park in the facility and make their parking experience hassle-free.

11.1.3 SURVELLIANCE

The surveillance field is an important feature of the Smart Parking System, providing users with access to the facility's CCTV cameras. The surveillance field can be accessed from the dashboard's "Surveillance" section.

From the surveillance field, users can view live feeds of the facility's CCTV cameras. The cameras are strategically placed to cover all areas of the parking facility, ensuring that users can monitor their vehicle's surroundings and ensure their safety. Users can select the camera feed they wish to view and enlarge it to get a better view of the area.

If users notice any suspicious activity, they can report it to the facility management by clicking on the "Report" button next to the camera feed. A dialog box will appear, allowing users to enter details about the suspicious activity they have observed. Once submitted, the report will be reviewed by the facility management, who will take appropriate action as necessary.

The surveillance field is designed to ensure that users have a safe and secure parking experience. By providing access to CCTV cameras and allowing users to report any suspicious activity, the Smart Parking System helps prevent crime and ensure the safety of users and their vehicles.

Result

Thus, the user interface framework has been created for the Smart Parking System.

12. AUTOMATED TESTING

Aim

To develop Automated Test Cases for Smart Parking System

12.1 Smart Parking System Automated Test Cases

GitHub: https://github.com/Dhruvch1244/admin-site-main

12.1.1 Test Case: Successful Login

Test Steps:

- Enter "admin" in the Username field.
- Enter "1234" in the Password field.
- Click on the "Submit" button.
- Verify that an alert message with text "Login Successful!" is displayed.
- Verify that the user is redirected to the "dashboard.html" page.

Expected Results:

- The user can successfully log in with valid credentials.
- An alert message with text "Login Successful!" is displayed.
- The user is redirected to the "dashboard.html" page.

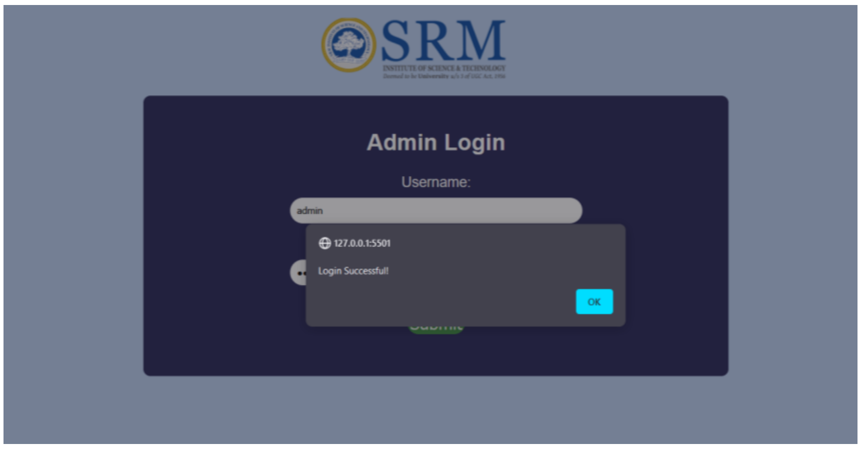
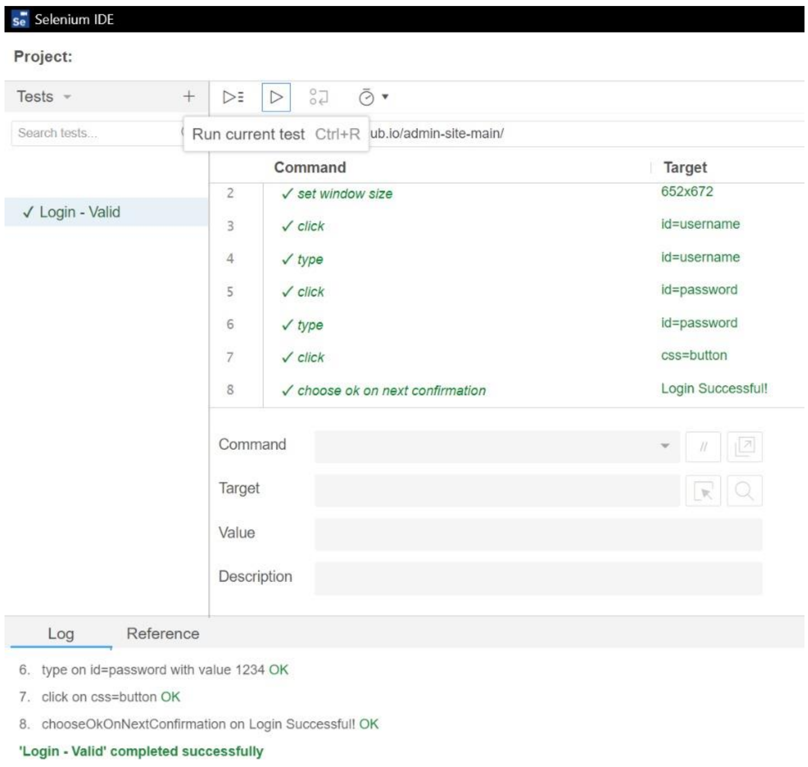


Figure 12.1.1: Actual Result for Successful Login



	Command	Target
2	✓ set window size	652x672
3	✓ click	id=username
4	✓ type	id=username
5	✓ click	id=password
6	✓ type	id=password
7	✓ click	css=button
8	✓ choose ok on next confirmation	Login Successful!

```
6.  type on id=password with value 1234 OK
7.  click on css=button OK
8.  chooseOkOnNextConfirmation on Login Successful! OK
'Login - Valid' completed successfully
```

Figure 12.1.2: Automated test case result run on Selenium (Login Successful)

Actual Results:

- An alert message with text "Login Successful!" was displayed.
- The user was redirected to the "dashboard.html" page.

Conclusion: The test case for successful login was executed successfully and passed as expected.

12.1.2 Test Case: Incorrect Username or Password

Purpose: To verify that an error message is displayed when an invalid username is entered.

Preconditions: The web browser is open and navigated to the Login page.

Test Steps:

- Enter "invalid" in the Username field.
- Enter "1234" in the Password field.
- Click on the "Submit" button.
- Verify that an alert message with text "Invalid Username or Password!" is displayed.

Expected Results

An alert message with text "Invalid Username or Password!" is displayed, indicating that the username entered is invalid.

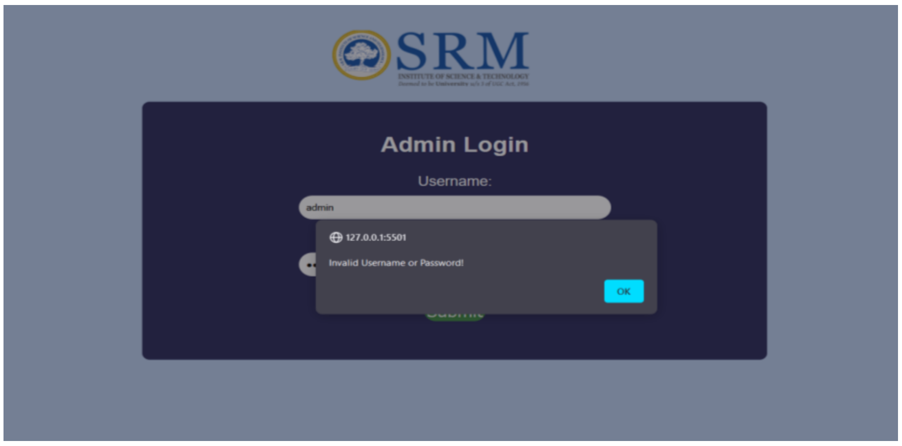
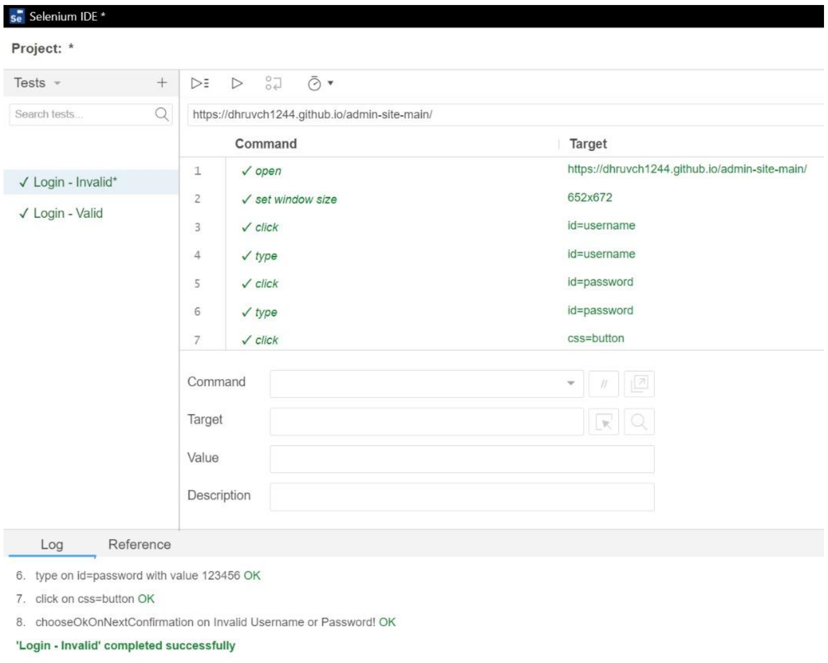


Figure 12.2.1 Actual Result for Invalid Login



https://dhruvch1244.github.io/admin-site-main/

	Command	Target
1	✓ open	https://dhruvch1244.github.io/admin-site-main/
2	✓ set window size	652x672
3	✓ click	id=username
4	✓ type	id=username
5	✓ click	id=password
6	✓ type	id=password
7	✓ click	css=button

```
6.  type on id=password with value 123456 OK
7.  click on css=button OK
8.  chooseOkOnNextConfirmation on Invalid Username or Password! OK
'Login - Invalid' completed successfully
```

Figure 12.2.2: Automated test case result run on Selenium (Invalid)

Actual Results:

An alert message with text "Invalid Username or Password!" was displayed, indicating that the username entered is invalid.

Conclusion:

The test case for an incorrect username was executed successfully and passed as expected.

12.1.3 Test Case: Creating a New Permit

Test Case 12.1.3.1: Valid Form Submission

Purpose: To verify that the form can be submitted with valid inputs.

Preconditions: The web browser is open and navigated to the form page.

Test Steps:

- Enter "ABC123" in the Vehicle number field.
- Enter "John Doe" in the Owner name field.
- Select "Student" from the Owner type drop-down menu.
- Click on the "Submit" button.
- Verify that the result element displays the message "Parking Slot Assigned to John Doe ABC123 Student".

Expected Results:

- The form can be submitted with valid inputs.
- The result element displays the message "Parking Slot Assigned to John Doe ABC123 Student".

Test Case 12.1.3.2: Empty Form Submission

Purpose: To verify that the form does not submit when no inputs are entered.

Preconditions: The web browser is open and navigated to the form page.

Test Steps:

- Leave all fields blank.
- Click on the "Submit" button.
- Verify that the form does not submit.
- Verify that the result element is empty.

Expected Results:

- The form does not submit when no inputs are entered.
- The result element is empty.

Test Case 12.1.3.3: Invalid Vehicle Number

Purpose: To verify that an error message is displayed when an invalid vehicle number is entered.

Preconditions: The web browser is open and navigated to the form page.

Test Steps:

- Enter "Jane Smith" in the Owner name field.
- Select "Faculty" from the Owner type drop-down menu.
- Click on the "Submit" button.
- Verify that the form does not submit.
- Verify that the Vehicle number input displays an error message.

Expected Results:

- The form does not submit when an invalid vehicle number is entered.
- The Vehicle number input displays an error message.

Test Case 12.1.3.4: Invalid Owner Name

Purpose: To verify that an error message is displayed when an invalid owner name is entered.

Preconditions: The web browser is open and navigated to the form page.

Test Steps:

- Enter "XYZ456" in the Vehicle number field.
- Enter "123" in the Owner name field.
- Select "Guest" from the Owner type drop-down menu.
- Click on the "Submit" button.
- Verify that the form does not submit.
- Verify that the Owner name input displays an error message.

Expected Results:

- The form does not submit when an invalid owner name is entered.
- The Owner name input displays an error message.

Test Case 12.1.3.5: Owner Type Not Selected

Purpose: To verify that an error message is displayed when the owner type is not selected.

Preconditions: The web browser is open and navigated to the form page.

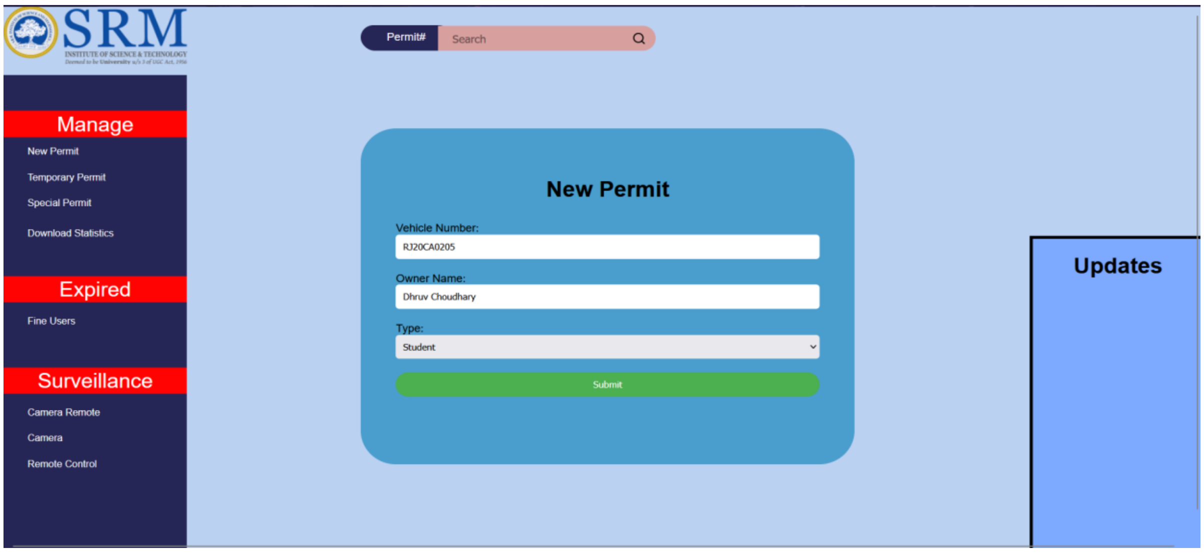
Test Steps:

- Enter "DEF789" in the Vehicle number field.
- Enter "Adam Lee" in the Owner name field.
- Do not select any option from the Owner type drop-down menu.
- Click on the "Submit" button.
- Verify that the form does not submit.
- Verify that the Owner type input displays an error message.

Expected Results:

The form does not submit when the owner type is not selected.

The Owner type input displays an error message.



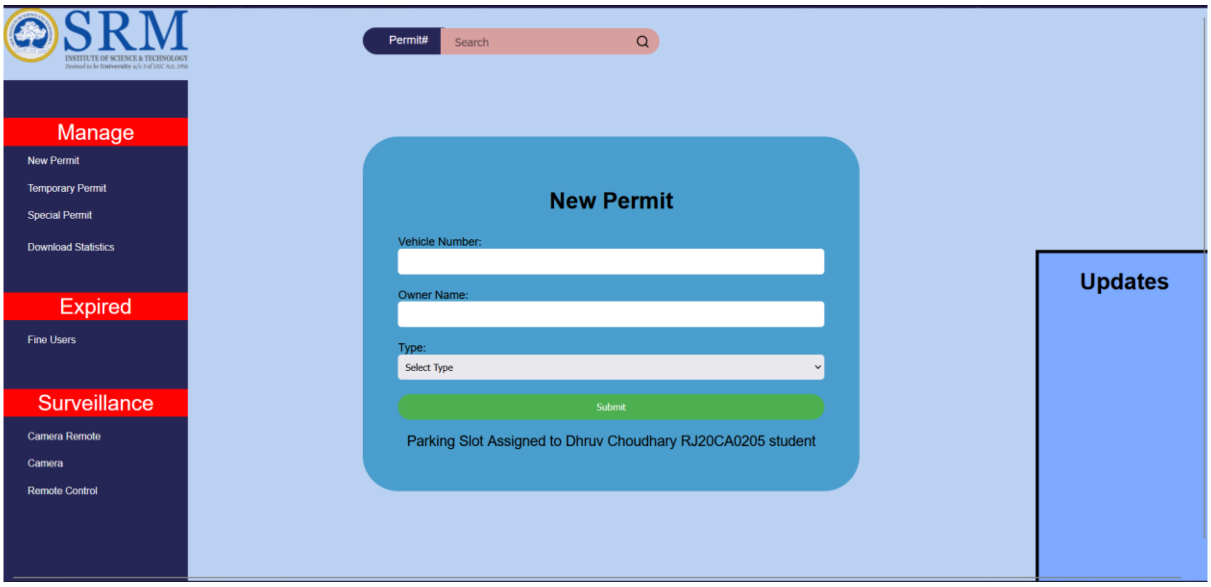
New Permit

Vehicle Number: RJ20CA0205

Owner Name: Dhruv Choudhary

Type: Student

Figure 12.3.1: Enter User Details



New Permit

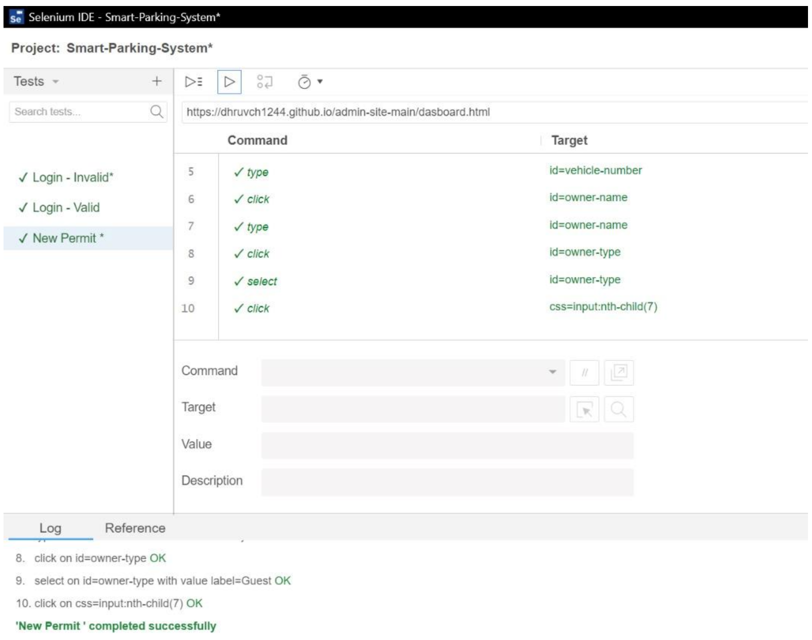
Vehicle Number:

Owner Name:

Type: Select Type

Parking Slot Assigned to Dhruv Choudhary RJ20CA0205 student

Figure 12.3.2: Assign Parking Slot to User



Selenium IDE - Smart-Parking-System*

Project: Smart-Parking-System*

https://dhruvch1244.github.io/admin-site-main/dasboard.html

	Command	Target
5	✓ type	id=vehicle-number
6	✓ click	id=owner-name
7	✓ type	id=owner-name
8	✓ click	id=owner-type
9	✓ select	id=owner-type
10	✓ click	css=input:nth-child(7)

```
8.  click on id=owner-type OK
9.  select on id=owner-type with value label=Guest OK
10. click on css=input:nth-child(7) OK
'New Permit ' completed successfully
```

Figure 12.3.3 Automated test case result run on Selenium (New Permit)

12.1.4 **Test Case: Removing Permit**

Test Case 12.1.4.1: Parking List Displayed Correctly on Page Load

Purpose: To verify that the parking list is displayed correctly on page load.

Preconditions:

- The web browser is open and navigated to the Remove Permit page.
- The parkingList array contains valid parking entries.

Test Steps:

- Open the Remove Permit page.
- Verify that the HTML table containing the parking list is present on the page.
- Verify that all parking entries in the list are displayed correctly, including the parking ID, name, time left, vehicle number, and remove button.
- Verify that the number of parking entries matches the number of entries in the parkingList array.

Expected Results:

- The parking list is displayed correctly on page load.
- All parking entries in the list are displayed correctly.
- The number of parking entries matches the number of entries in the parkingList array.

Test Case 12.1.4.2: New Parking Entry Can be Added to the List

Purpose: To verify that a new parking entry can be added to the list.

Preconditions:

- The web browser is open and navigated to the Remove Permit page.
- The parkingList array does not contain the new parking entry.

Test Steps:

- Open the Remove Permit page.
- Simulate a user entering a new parking entry in the form.
- Check that the new entry is added to the parkingList array and the HTML table.
- Check that the new entry is displayed correctly, including the parking ID, name, time left, vehicle number, and remove button.

Expected Results:

- The new parking entry is added to the parkingList array and the HTML table.
- The new parking entry is displayed correctly, including the parking ID, name, time left, vehicle number, and remove button.

Test Case 12.1.4.3: Parking Entry Can be Removed from the List

Purpose: To verify that a parking entry can be removed from the list.

Preconditions:

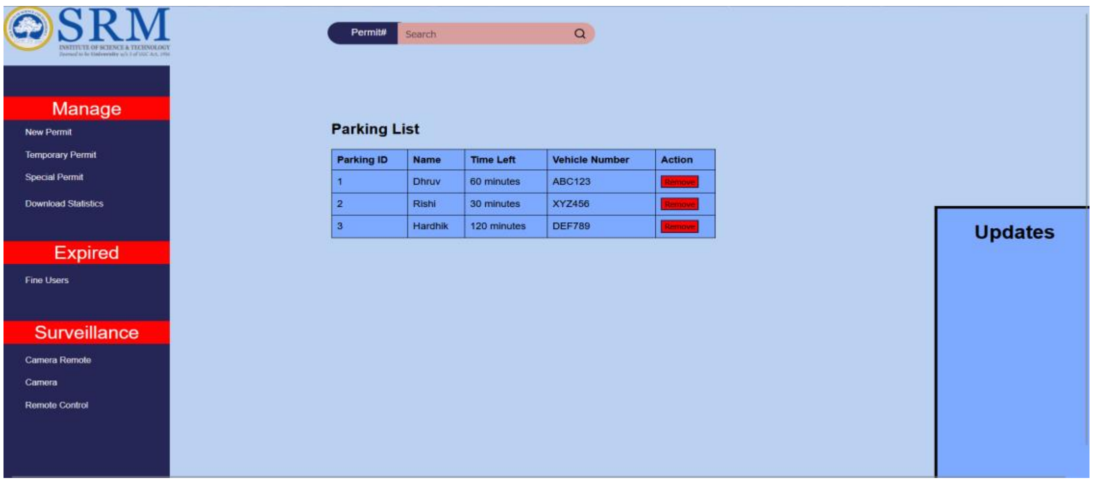
- The web browser is open and navigated to the Remove Permit page.
- The parkingList array contains the parking entry to be removed.

Test Steps:

- Open the Remove Permit page.
- Simulate a user clicking on the remove button for a parking entry.
- Check that the corresponding entry is removed from the parkingList array and the HTML table.
- Check that the removed entry is no longer displayed in the HTML table.

Expected Results:

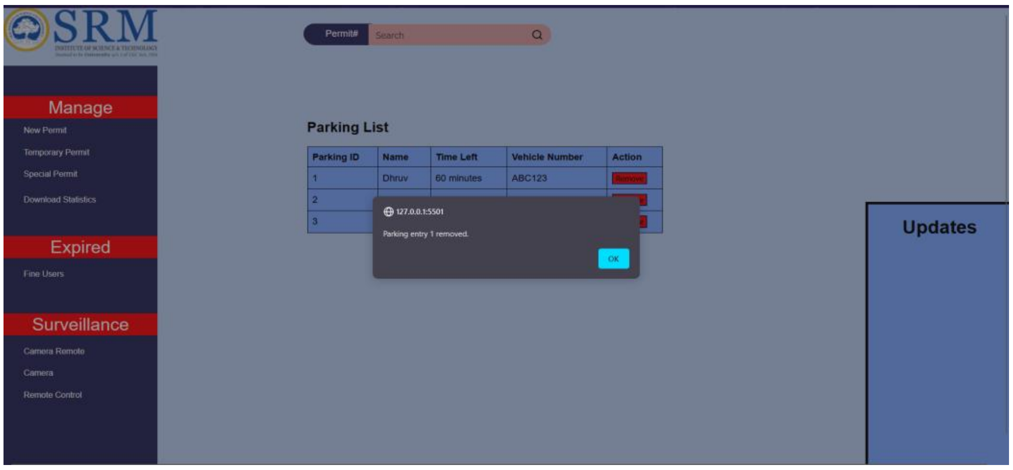
- The corresponding parking entry is removed from the parkingList array and the HTML table.
- The removed entry is no longer displayed in the HTML table.



Parking List

Parking ID	Name	Time Left	Vehicle Number	Action
1	Dhruv	60 minutes	ABC123	Remove
2	Rishi	30 minutes	XYZ456	Remove
3	Hardhik	120 minutes	DEF789	Remove

Figure 12.4.1: Display list of added permits



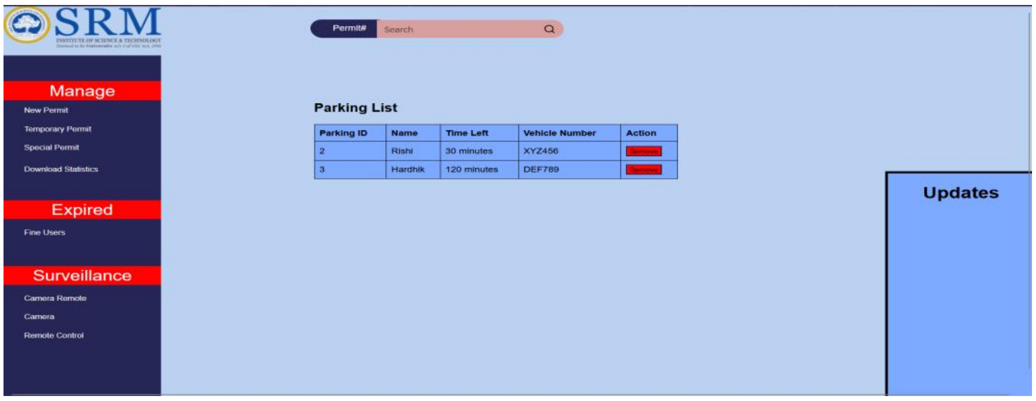
Parking List

Parking ID	Name	Time Left	Vehicle Number	Action
1	Dhruv	60 minutes	ABC123	Remove
2				
3				

127.0.0.1:5501

Parking entry 1 removed.

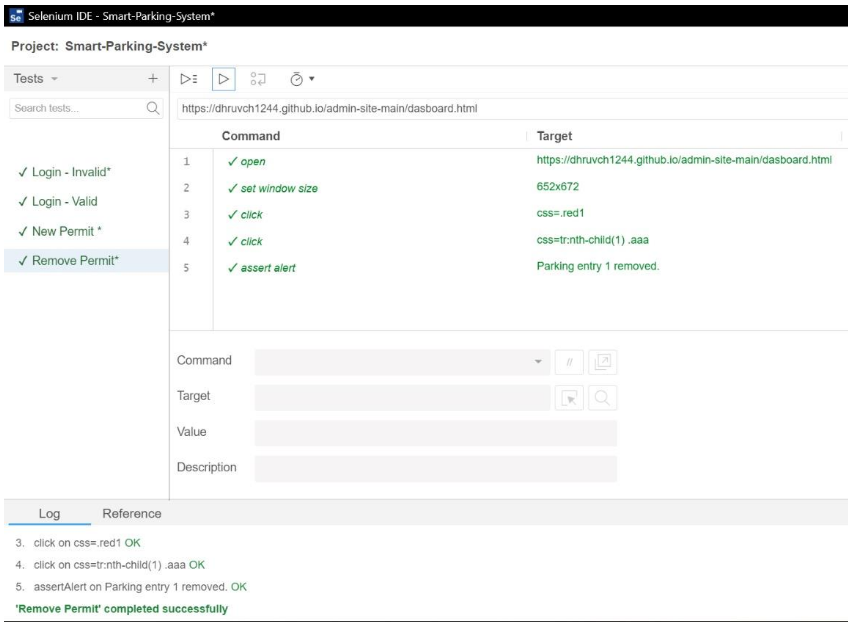
Figure 12.4.2: Display the permit to be removed.



Parking List

Parking ID	Name	Time Left	Vehicle Number	Action
2	Rishi	30 minutes	XYZ456	Remove
3	Hardhik	120 minutes	DEF789	Remove

Figure 12.4.3: Display the updated parking list.



https://dhruvch1244.github.io/admin-site-main/dasboard.html

	Command	Target
1	✓ open	https://dhruvch1244.github.io/admin-site-main/dasboard.html
2	✓ set window size	652x672
3	✓ click	css=.red1
4	✓ click	css=tr:nth-child(1) .aaa
5	✓ assert alert	Parking entry 1 removed.

```
3.  click on css=.red1 OK
4.  click on css=tr:nth-child(1) .aaa OK
5.  assertAlert on Parking entry 1 removed. OK
'Remove Permit' completed successfully
```

Figure 4.4: Automated test case result on Selenium (Remove Permit)

12.1.5 Test Case: Smart Number Plate Detector

GitHub: https://github.com/Dhruvch1244/Number-Plate-Detection

1) **Test for a valid input image**

- Expected input: valid_input_image.jpg is a valid image file containing a number plate.
- Expected output: ABC1234 is the expected output text containing the recognized characters from the number plate.

2) **Test case for an invalid input image path:**

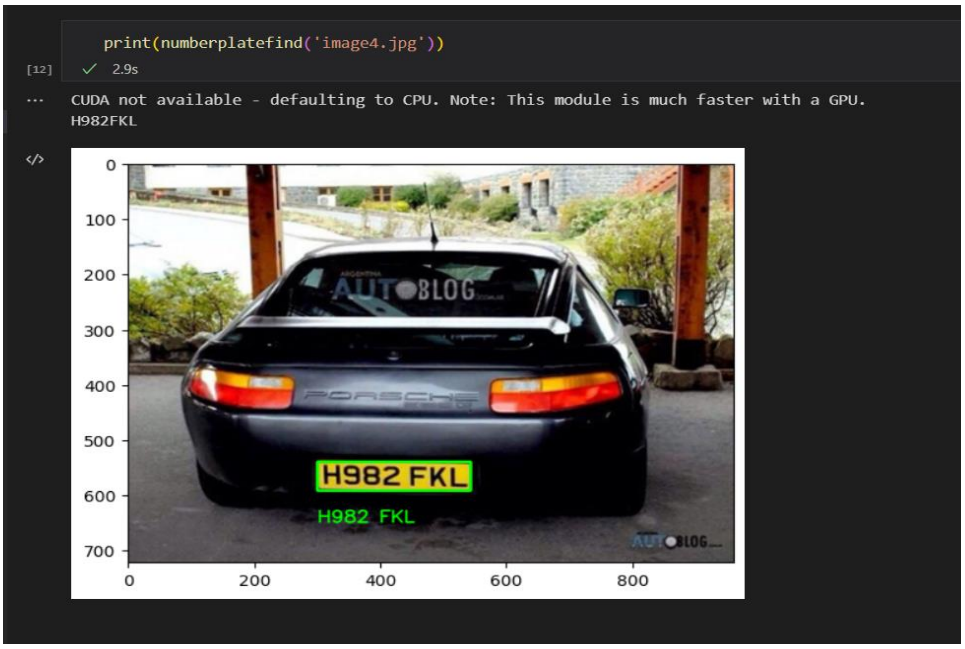
- Expected input: invalid_input_image.jpg is an invalid image file path. Expected output: A FileNotFoundError should be raised since the input file path is invalid.
- Test case for an input image that does not contain a number plate.

3) **Test case for an input image that does not contain a number plate:**

- Expected input: input_without_number_plate.jpg is a valid image file that does not contain a number plate.
- Expected output: The function should return an empty string since no number plate was found in the input image.

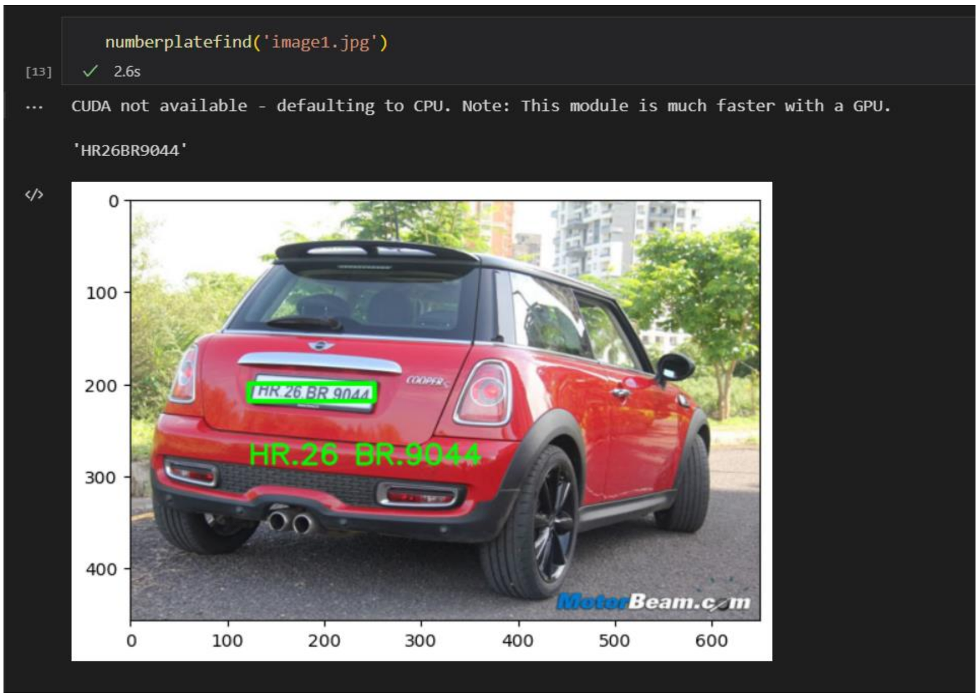
4) **Test case for an input image with low resolution:**

- Expected input: low_resolution_input_image.jpg is a valid image file containing a number plate, but with low resolution.
- Expected output: The function should still be able to detect the number plate and return the expected output text containing the recognized characters.



```
print(numberplatefind('image4.jpg'))
[12] ✓ 2.9s
... CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
    H982FKL
```

Figure 12.5.1 Number Plate Detector



```
numberplatefind('image1.jpg')
[13] ✓ 2.6s
... CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.

    'HR26BR9044'
```

Figure 12.5.2 Number Plate Detector

Result

Thus, the Automated Test Cases has been completed for the Smart Parking System.

13. ARCHITECTURE DESIGN/FRAMEWORK/IMPLEMENTATION

Aim

To provide the details of architectural design/framework/implementation

13.1 Module 1 - Smart Parking Admin Site

Tech Stack Used: HTML , CSS , JS

Implementation

1. New Permit: To implement the "New Permit" functionality, you would need to create a form where users can enter their permit details such as permit type, duration, and vehicle information. Once the user submits the form, you can save the permit details to a database or file. You may also want to implement validation to ensure that the permit details are correct and complete.
2. Login: To implement the "Login" functionality, you would need to create a login form where users can enter their credentials such as username and password. Once the user submits the form, you can validate their credentials against a database of registered users. If the credentials are valid, you can redirect the user to the dashboard page. You may also want to implement features like "Remember Me" or password reset.
3. Remove Permit: To implement the "Remove Permit" functionality, you would need to create a form or a button where users can request to remove their permit. You can validate the request against the permit database to ensure that the user has the authority to remove the permit. Once the request is validated, you can remove the permit from the database or mark it as inactive.
4. Dashboard: To implement the "Dashboard" functionality, you would need to create a page where users can view their permit details such as permit type, duration, and vehicle information. You can retrieve the user's permit details from the database and display them on the dashboard. You may also want to implement features like editing permit details or renewing permits. Additionally, you may want to implement authentication to ensure that only logged-in users can access the dashboard.

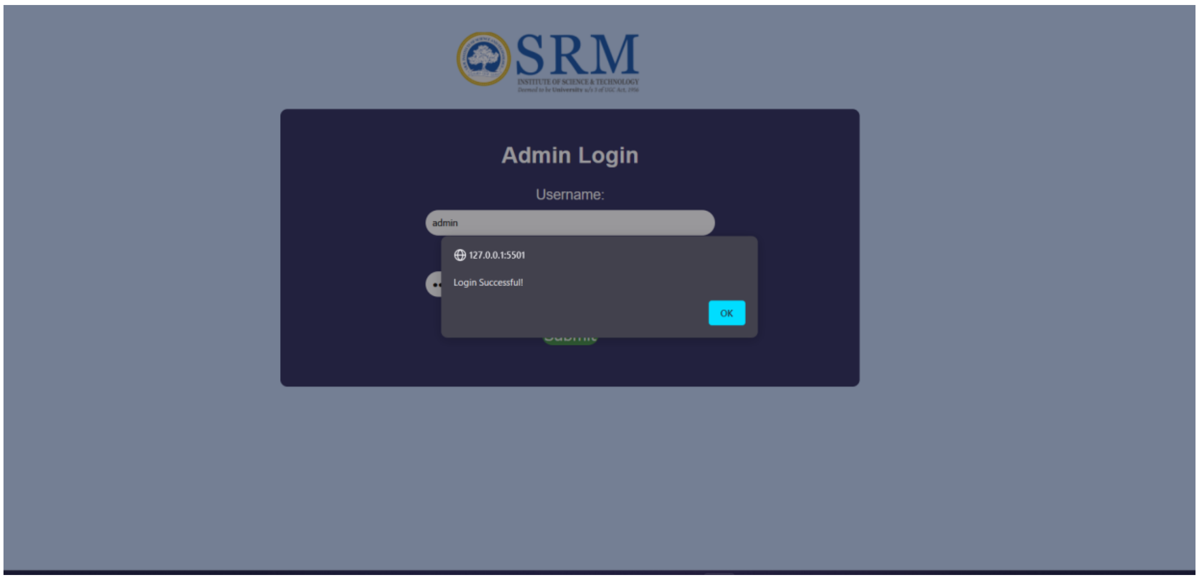


Figure 13.1: Admin Site Login Page (Valid Login)

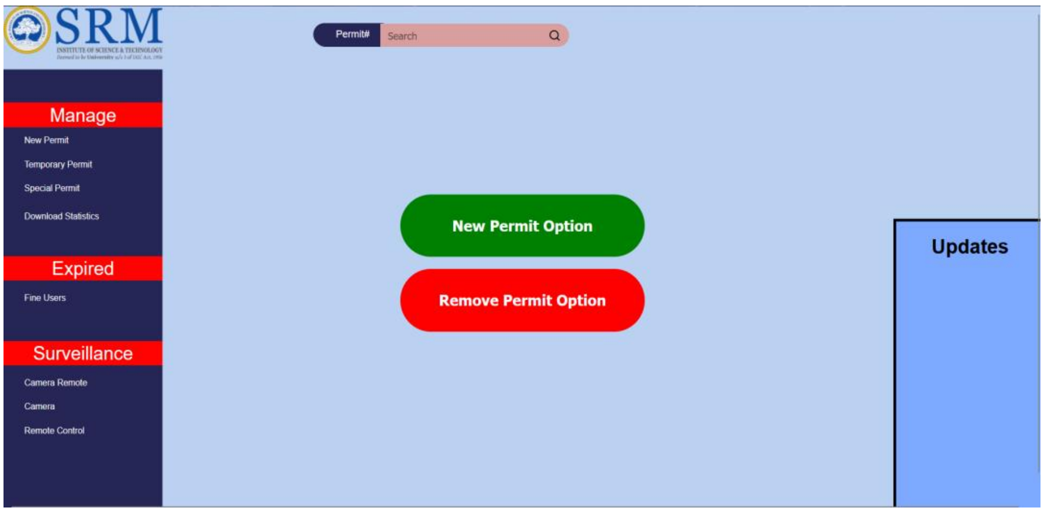
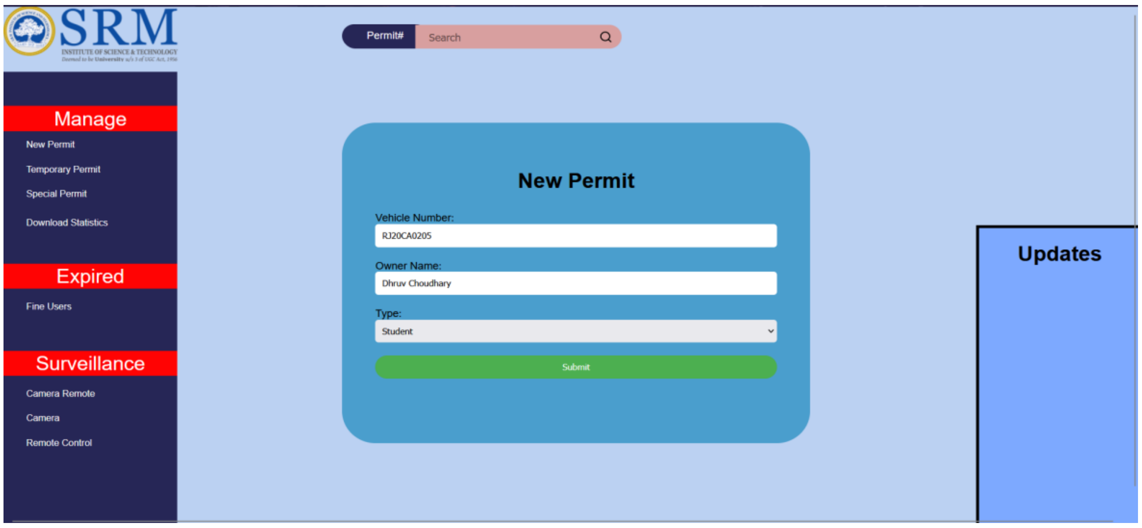


Figure 13.2: Dashboard



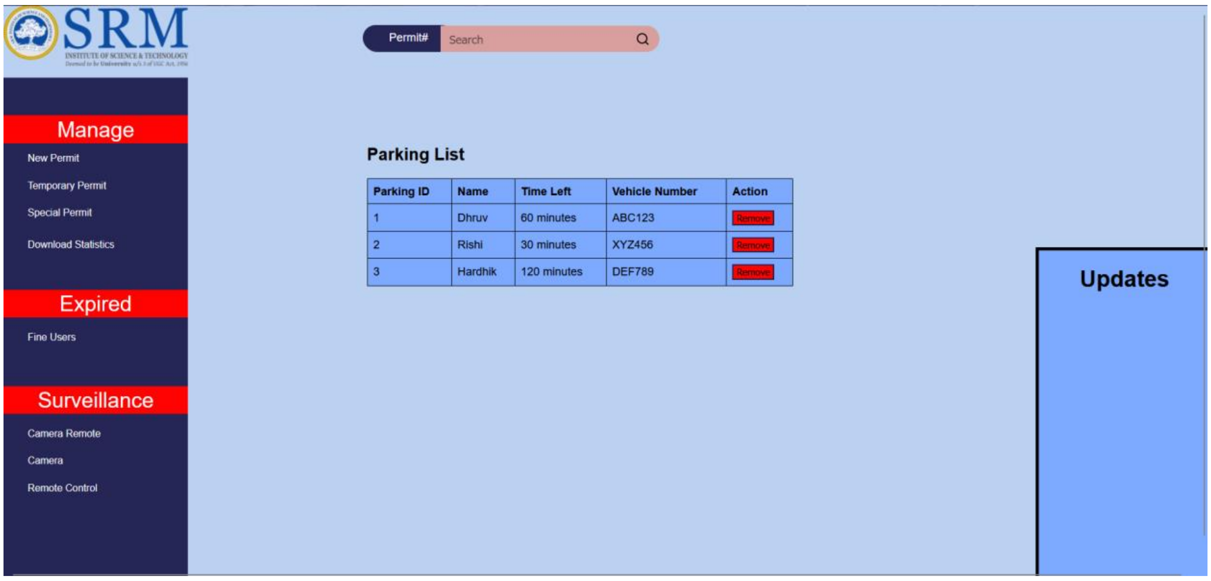
New Permit

Vehicle Number: RJ20CA0205

Owner Name: Dhruv Choudhary

Type: Student

Figure 13.3: New Permit



Parking List

Parking ID	Name	Time Left	Vehicle Number	Action
1	Dhruv	60 minutes	ABC123	Remove
2	Rishi	30 minutes	XYZ456	Remove
3	Hardhik	120 minutes	DEF789	Remove

Figure 13.4 Parking List (Remove Permit)

.13.2 Module 2 - Number Plate Detector

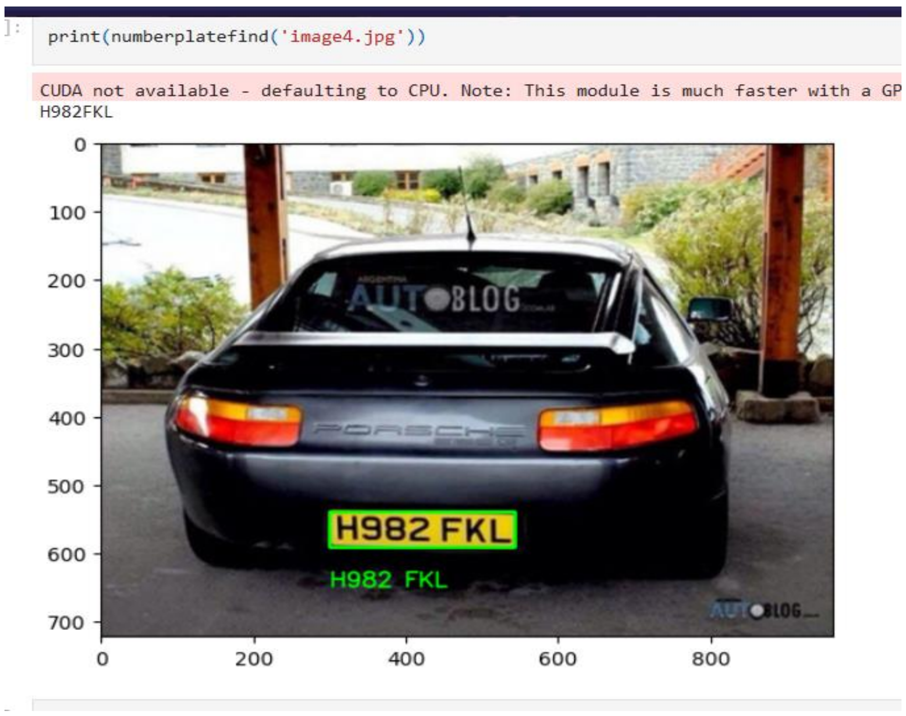
Tech Stack Used : Python , OCR , OpenCv, imutils, SQLite3

CODE -https://github.com/Dhruvch1244/Number-Plate-Detection

Number Plate Detection :- Python script that uses computer vision techniques to extract the license plate number from an image of a car. It uses the OpenCV library to apply noise reduction and edge detection techniques to the image, and then locates the license plate by finding contours and approximating the shape of the plate. The text on the plate is then read using the EasyOCR library and extracted from the image, after which it is cleaned up and returned as a string.

Booking : is a Python script that implements a simple parking management system using SQLite. The script allows users to book a parking slot, leave a parking slot, confirm a parking slot, and assign a parking slot. The system stores information about the bookings in a database, and uses SQL queries to retrieve and manipulate the data. The script prompts the user for input and returns appropriate messages based on the operation performed.

Both codes demonstrate the power of Python in implementing real-world solutions using various libraries and tools. Computer vision techniques are useful in automating image processing tasks, while databases and SQL are essential for managing data in any application. These codes can be extended and improved for various use cases in the future.



```
print(numberplatefind('image4.jpg'))
```

```
CUDA not available - defaulting to CPU. Note: This module is much faster with a GP
H982FKL
```

Figure 13.5 Number Plate Detector

```
numberplatefind('image1.jpg')
```

```
CUDA not available - defaulting to CPU. Note: This module is much faster with a G
'HR26BR9044'
```

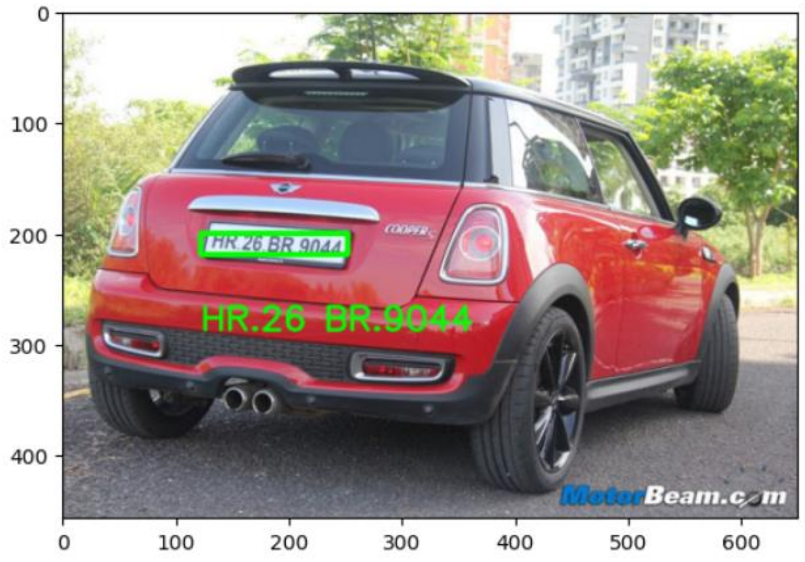


Figure 13.6 Number Plate Detector

```
numberplatefind('image2.jpg')
```

```
CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU
'ICOVIDT9'
```

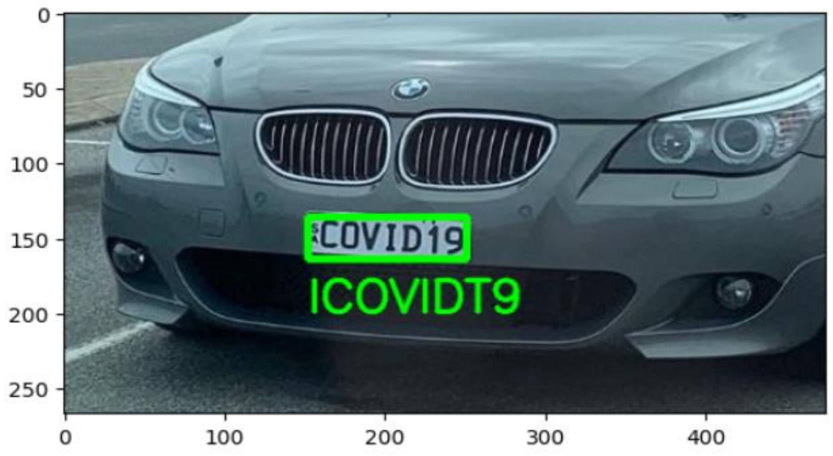
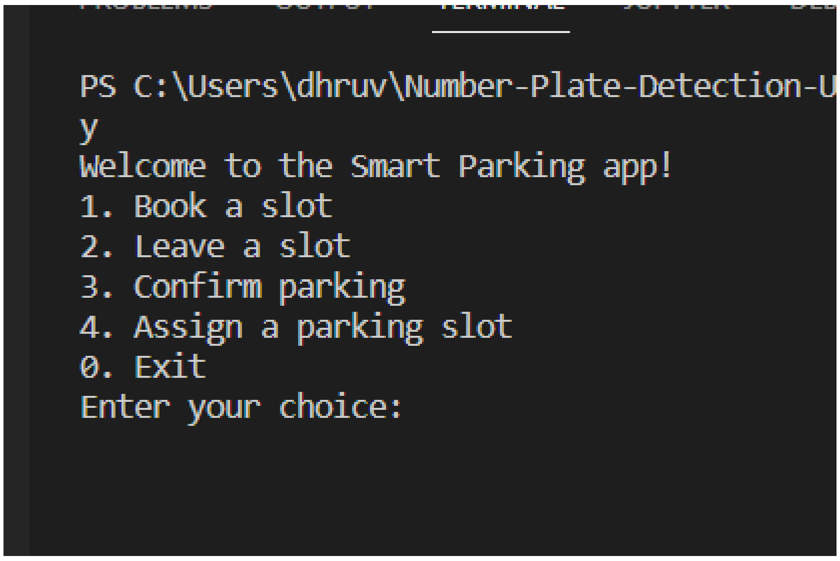
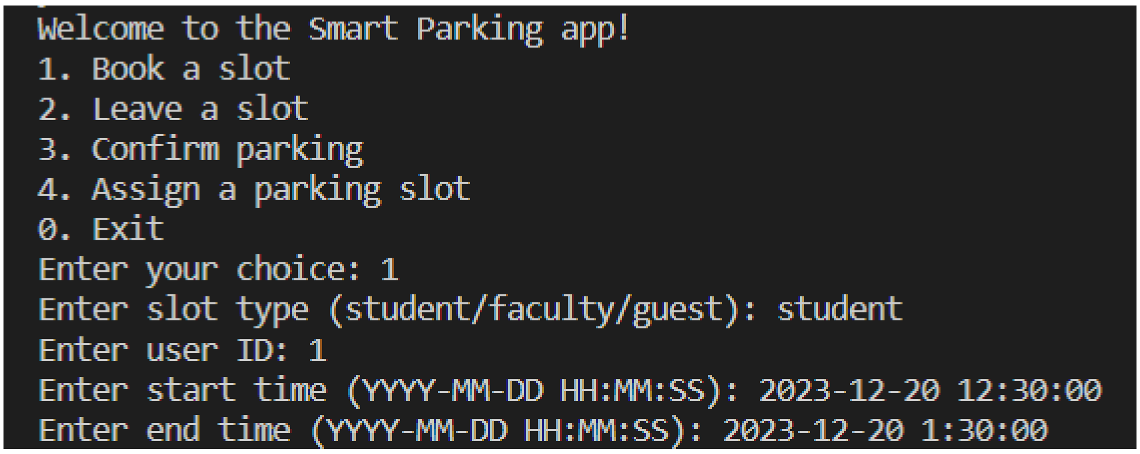


Figure 13.7 Number Plate Detector



```
PS C:\Users\dhruv\Number-Plate-Detection-U
y
Welcome to the Smart Parking app!
1. Book a slot
2. Leave a slot
3. Confirm parking
4. Assign a parking slot
0. Exit
Enter your choice:
```

Figure 13.8 Application Dashboard (Backend)



```
Welcome to the Smart Parking app!
1. Book a slot
2. Leave a slot
3. Confirm parking
4. Assign a parking slot
0. Exit
Enter your choice: 1
Enter slot type (student/faculty/guest): student
Enter user ID: 1
Enter start time (YYYY-MM-DD HH:MM:SS): 2023-12-20 12:30:00
Enter end time (YYYY-MM-DD HH:MM:SS): 2023-12-20 1:30:00
```

Figure 13.9 Book Slot (Backend)

13.3 Module 3 - Parking app for end user

Tech Stack Used : Flutter , Dart

CODE - https://github.com/Dhruvch1244/SmartParkingApp

1. MyApp widget

- This widget is the main entry point of the application and is responsible for running the application.
- It initializes the MaterialApp widget and returns it as the root widget of the app.

2. LoginPage widget

- This widget is a stateful widget that builds the login page of the app.
- It contains a Stack widget with a Center widget as a child. The Center widget contains a container with a white background, rounded corners, and a shadow effect. Inside the container, there are a logo image, two text fields for username and password, and two buttons for logging in and resetting the password.
- The LoginPage widget is responsible for invoking the validateLogin function on login button press.

3. DashboardPage widget

- This widget is a stateless widget that builds the dashboard page of the app.
- It contains an AppBar and a Center widget as a child. The Center widget contains a Column with a welcome message and three buttons for new booking, leaving the parking slot, and modifying the parking slot.
- The DashboardPage widget is responsible for navigating to the appropriate screen on button press.

4. NewBookingPage widget

- This widget is a stateful widget that builds the new booking page of the app.
- It contains a Form widget with several text fields for entering details about the booking. It also has a button for submitting the form and making a new booking.
- The NewBookingPage widget is responsible for validating the input and making a new booking on button press.

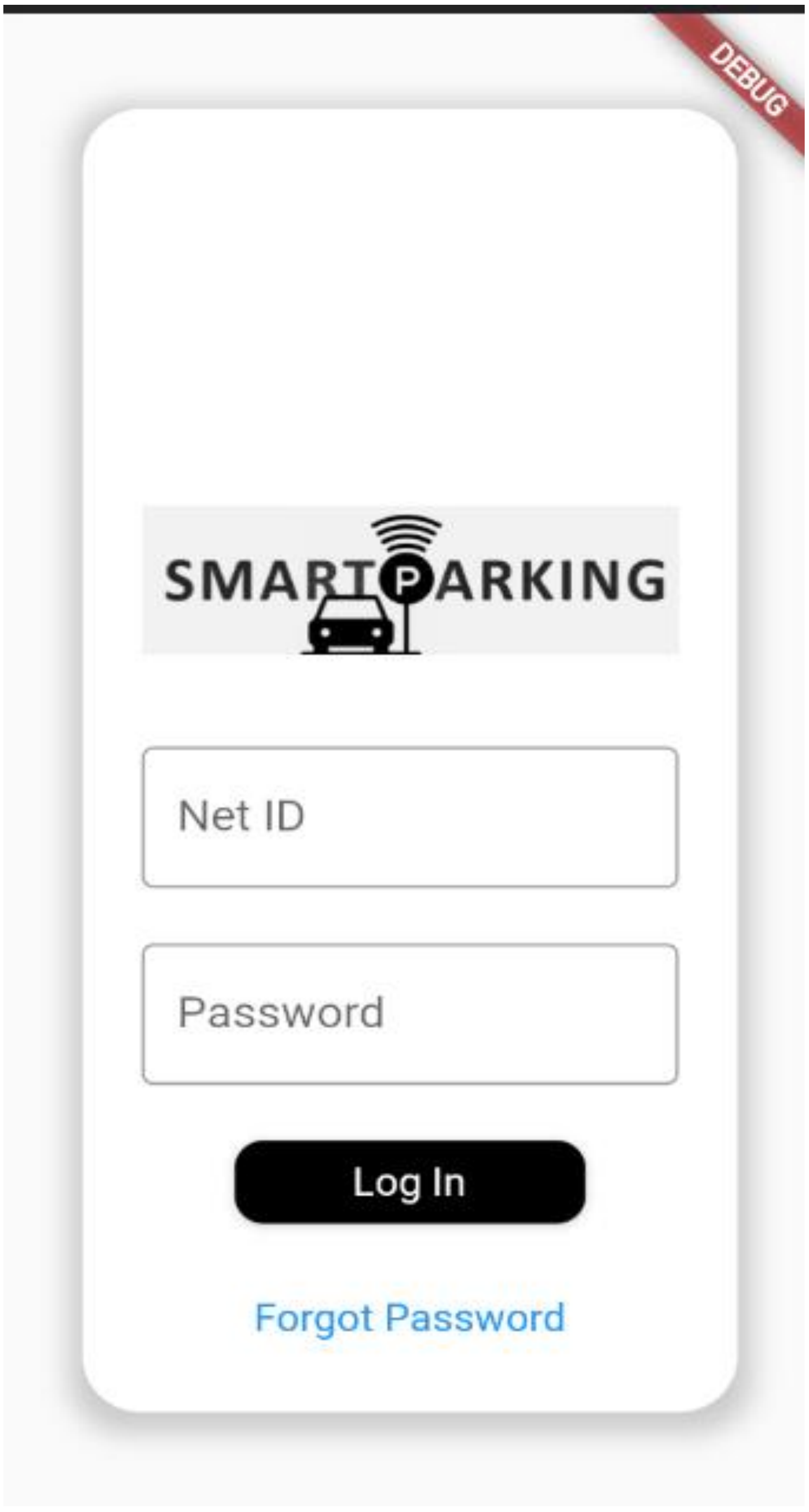


Figure 13.10 App Login Page

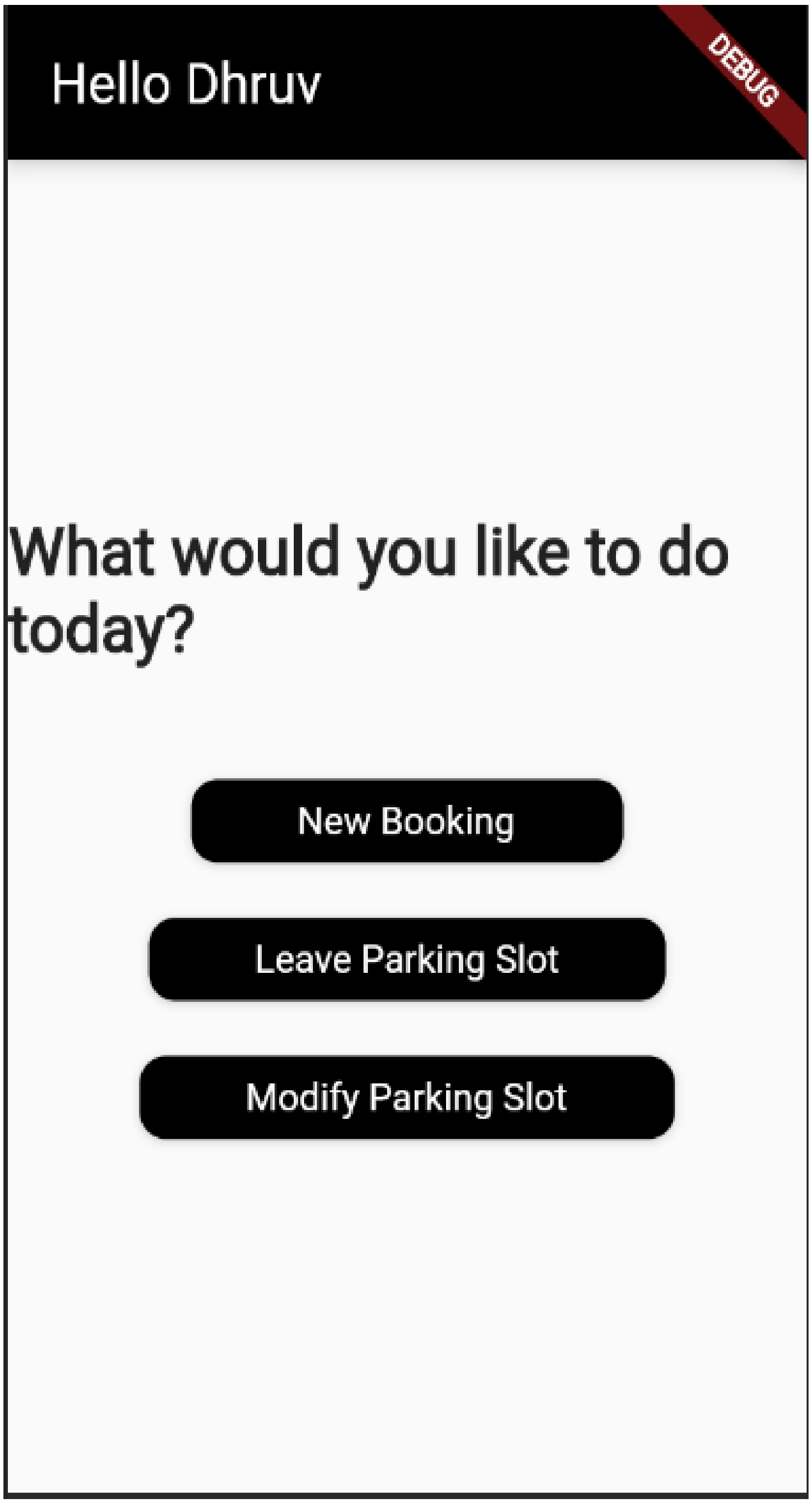


Figure 13.11 App Dashboard

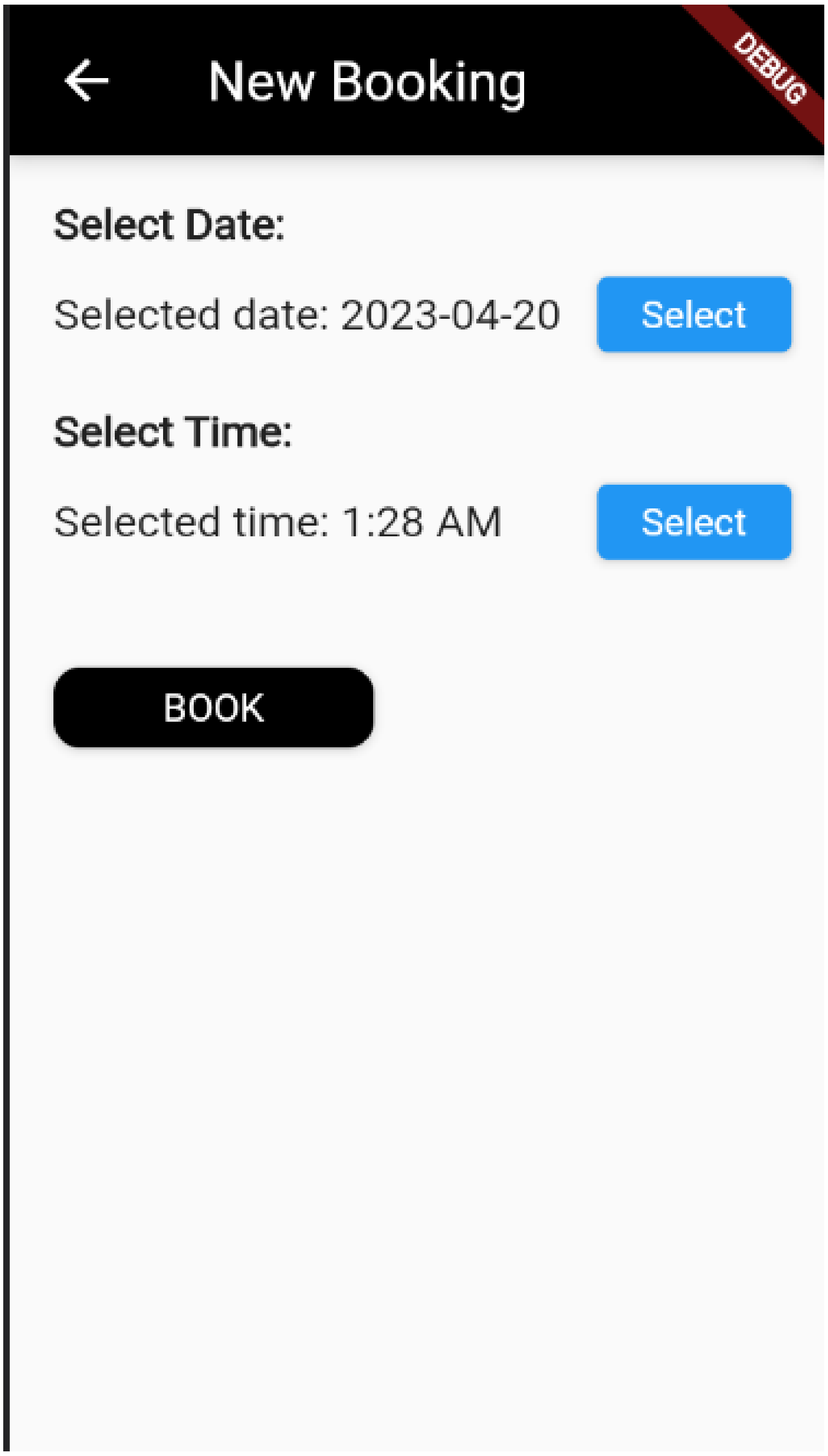
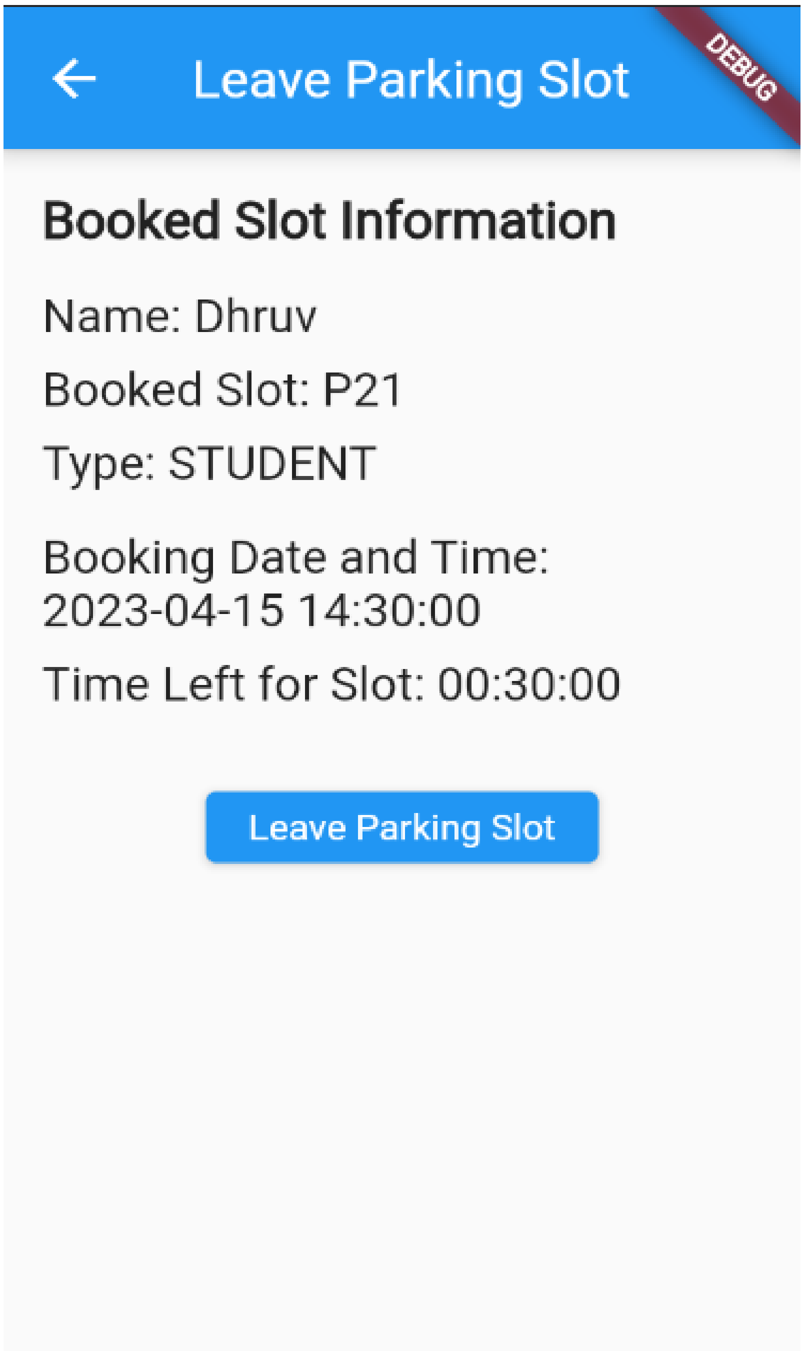


Figure 13.12 Booking Slot in App



Booked Slot Information

Name: Dhruv

Booked Slot: P21

Type: STUDENT

Booking Date and Time: 2023-04-15 14:30:00

Time Left for Slot: 00:30:00

Figure 13.13 Slot Booking Information in App

Result

Thus, the details of architectural design/framework/implementation along with the screenshots were provided.

V. CONCLUSION

In conclusion, the implementation of a smart parking system equipped with a number plate detector, an admin site for permit management, and a user booking app is an essential solution for universities to address parking challenges. The university parking system has several advantages, including efficient use of available parking spaces, improved traffic flow, and reduced carbon footprint, making it an environmentally friendly option.

The smart parking system enables the university administration to manage permits effectively, ensuring that only authorized vehicles can access the parking facility. The admin site allows for seamless management of permits, including adding or deleting permits, ensuring that parking spaces are only utilized by those who have permission to use them.

The user booking app simplifies the process of reserving a parking slot for students, faculty, and staff, eliminating the need to physically visit the parking facility to check for availability. This convenience enhances the user experience, saves time, and reduces frustration that often arises from searching for parking spots.

Overall, the smart parking system is a technological advancement that presents several benefits to universities. It promotes efficient parking management, reduces congestion, enhances user experience, and contributes to a more sustainable environment. The implementation of such a system is highly recommended for any university seeking to address parking challenges, improving the overall campus experience for students, faculty, and staff.

VI. REFERENCES

1. **https://in.nec.com/en_IN/solutions_services/intelligent_transport_solutions/smart_parking.html**
2. **https://tomorrow.city/a/smart-parking**
3. **https://www.w3schools.com/**
4. **https://chat.openai.com/**
5. **https://flutter.dev/**
6. **https://www.javatpoint.com/javascript-tutorial**
7. **https://flask.palletsprojects.com/en/2.3.x/**
8. **https://www.atlassian.com/agile**
9. **https://machinelearningprojects.net/number-plate-detection-using-yolov7/**

VII. APPENDIX (CODE)

1. App.py

```
from flask import Flask, render_template, request, jsonify
import sqlite3

app = Flask(_name_)

@app.route('/')
def index():
        return render_template('index.html')

@app.route('/submit', methods=['POST'])
def submit():
        # Get the form data
        vehicle_number = request.form['vehicle-number']
        owner_name = request.form['owner-name']
        owner_type = request.form['owner-type']

        # Connect to the database
        conn = sqlite3.connect('data.db')
        c = conn.cursor()

        # Insert the data into the permits table
        c.execute('INSERT INTO permits (vehicle_number, owner_name, owner_type)
VALUES (?, ?, ?)', (vehicle_number, owner_name, owner_type))

        # Commit the changes and close the connection
        conn.commit()
        conn.close()

        # Return a response to the client
        return jsonify({'success': True})

@app.route('/get_permits')
def get_permits():
        # Connect to the database
        conn = sqlite3.connect('data.db')
        c = conn.cursor()

        # Retrieve the data from the permits table
        c.execute('SELECT * FROM permits')
        permits = c.fetchall()

        # Close the connection
        conn.close()
        # Return the permits data as JSON
        return jsonify({'permits': permits})

@app.route('/newpermit')
def newpermit():
        # Render the template with the permit data
        return render_template('newpermit.html')

@app.route('/dashboard')
def dashboard():
        # Render the template with the permit data
        return render_template('dashboard.html')

@app.route('/showpermit')
def showpermit():
        return render_template('showpermit.html')

@app.route('/remove_permit')
def remove_permit():
        vehicle_number = request.args.get('vehicle_number')
        conn = sqlite3.connect('data.db')
        cursor = conn.cursor()
        cursor.execute('DELETE FROM permits WHERE vehicle_number = ?',
(vehicle_number,))
        conn.commit()
        conn.close()
        return jsonify({'message': 'Permit removed successfully'})


if __name__== '_main_':
        app.run(debug=True)
```

2. **Login.js**

```
function checkLogin() {
        var username = document.getElementById("username").value;
        var password = document.getElementById("password").value;

        if (username === "admin" && password === "1234") {
        alert('Login Successful!');
        windows.location.href = "{{ url_for('showpermit') }}";
        } else {
        alert('Invalid Username or Password!');
        }
}
```

3. new,js

```
const form = document.querySelector('#parking-form');
const result = document.querySelector('.result');
const updates = document.querySelector('.tt');
form.addEventListener('submit', e => {
	e.preventDefault();
	const vehicleNumber = form.querySelector('#vehicle-number').value;
	const ownerName = form.querySelector('#owner-name').value;
	const ownerType = form.querySelector('#owner-type').value;
	result.textContent = `Parking Slot Assigned to ${ownerName} ${vehicleNumber}
${ownerType}`;
	updates.textContent = `Parking Slot Assigned to ${ownerName} ${vehicleNumber}
${ownerType}`;
	form.reset();
});

function submitForm(event) {
	event.preventDefault();

	var form = document.getElementById('parking-form');
	var formData = new FormData(form);

	fetch(form.action, {
	method: 'POST',
	body: formData
	})
	.then(response => response.json())
	.then(data => {
	if (data.success) {
		alert('Permit submitted successfully!');
		form.reset();
	} else {
		alert('Permit submission failed.');
	}
	})
	.catch(error => {
	console.error(error);
	alert('An error occurred. Please try again later.');
	});
}
```

4. Remove.js

```
$(document).ready(function() {
	$.getJSON('/get_permits', function(data) {
	var permits = data.permits;
	var table = $('#permits-table tbody');
	$.each(permits, function(i, permit) {
```

```
            var row = $('<tr>');
            var vehicleNumberCell = $('<td>').text(permit[0]);
            var ownerNameCell = $('<td>').text(permit[1]);
            var ownerTypeCell = $('<td>').text(permit[2]);
            var removeButton = $('<button>').text('Remove');
            removeButton.click(function() {
                        var vehicleNumber = permit[0];
                        $.get('/remove_permit', { 'vehicle_number': vehicleNumber }, function(data)
{
                        alert(data.message);
                        row.remove();
                        });
            });
            var actionCell = $('<td>').append(removeButton);
            row.append(vehicleNumberCell);
            row.append(ownerNameCell);
            row.append(ownerTypeCell);
            row.append(actionCell);
            table.append(row);
            });
            });
});
```

APP CODE IN Flutter :

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
            return MaterialApp(
            title: 'Login Page',
            theme: ThemeData(
            primarySwatch: Colors.blue,
            ),
            home: LoginPage(),
            );
  }
}

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}
```

```
class _LoginPageState extends State<LoginPage> {
  final TextEditingController _netIdController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();

  @override
  Widget build(BuildContext context) {
        return Scaffold(
        body: Stack(
        children: [
        Center(
        child: Container(
        width: MediaQuery.of(context).size.width * 0.8,
        decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.circular(20),
                boxShadow: [
                BoxShadow(
                color: Colors.grey.withOpacity(0.5),
                spreadRadius: 5,
                blurRadius: 7,
                offset: Offset(0, 3),
                ),
                ],
        ),
        padding: EdgeInsets.all(20),
        child: SingleChildScrollView(
                child: Column(
                crossAxisAlignment: CrossAxisAlignment.center,
                children: [
                SizedBox(height: 100),
                Image.asset(
                'images/1.png',
                height: 100,
                width: 300,
                ),
                SizedBox(height: 10),
                TextField(
                controller: _netIdController,
                decoration: InputDecoration(
                labelText: 'Net ID',
                border: OutlineInputBorder(),
                ),
                ),
                SizedBox(height: 20),
                TextField(
                controller: _passwordController,
                obscureText: true,
                decoration: InputDecoration(
                labelText: 'Password',
                border: OutlineInputBorder(),
                    ),
                    ),
                    SizedBox(height: 20),
                    ElevatedButton(
                    onPressed: () {
                    validateLogin(
                            _netIdController.text, _passwordController.text, context);
                    },
                    child: Text('Log In'),
                    style: ElevatedButton.styleFrom(
                    primary: Colors.black,
                    onPrimary: Colors.white,
                    shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10),
                    ),
                    padding:
                            EdgeInsets.symmetric(horizontal: 40, vertical: 15),
                    ),
                    ),
                    SizedBox(height: 20),
                    TextButton(
                    onPressed: () {
                    // Navigate to forgot password screen
                    },
                    child: Text('Forgot Password'),
                    ),
                    ],
                    ),
            ),
            ),
            ),
            ],
            ),
            );
  }
}

void validateLogin(String username, String password, BuildContext context) {
  if (username == 'dhruv' && password == '1234') {
            Navigator.pushReplacement(
            context,
            MaterialPageRoute(
            builder: (BuildContext context) => DashboardPage(),
            ),
            );
  } else {
            showDialog(
            context: context,
            builder: (BuildContext context) => AlertDialog(
            title: Text('Invalid credentials'),
            content: Text('Please enter a valid username and password'),
            actions: [
            TextButton(
            onPressed: () => Navigator.pop(context),
            child: Text('OK'),
            ),
            ],
            ),
            );
  }
}
class DashboardPage extends StatelessWidget {
  const DashboardPage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
            return Scaffold(
            appBar: AppBar(
            backgroundColor: Colors.black,
            title: Text('Hello Dhruv'),
            ),
            body: Center(
            child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
            Text(
            'What would you like to do today?',
            style: TextStyle(
                        fontSize: 24,
                        fontWeight: FontWeight.bold,
            ),
            ),
            SizedBox(height: 40),
            ElevatedButton(
            onPressed: () {
                        // Navigate to new booking screen
            Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => NewBookingPage()),
            );

            },
            child: Text('New Booking'),
            style: ElevatedButton.styleFrom(
                        primary: Colors.black,
                        onPrimary: Colors.white,
                        shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(10),
                        ),
                        padding:
```

```
                    EdgeInsets.symmetric(horizontal: 40, vertical: 15),
            ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
            onPressed: () {
                    // Navigate to leave parking slot screen
                            Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => LeaveParkingSlotPage ()),
            );
            },
            child: Text('Leave Parking Slot'),
            style: ElevatedButton.styleFrom(
                    primary: Colors.black,
                    onPrimary: Colors.white,
                    shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10),
                    ),
                    padding:
                    EdgeInsets.symmetric(horizontal: 40, vertical: 15),
            ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
            onPressed: () {
                    // Navigate to modify parking slot screen

            },
            child: Text('Modify Parking Slot'),
            style: ElevatedButton.styleFrom(
                    primary: Colors.black,
                    onPrimary: Colors.white,
                    shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10),
                    ),
                    padding:
                    EdgeInsets.symmetric(horizontal: 40, vertical: 15),
            ),
            ),
            ],
            ),
            ),
            );
  }
}
class NewBookingPage extends StatefulWidget {
  const NewBookingPage({Key? key}) : super(key: key);

  @override
```

```
 _NewBookingPageState createState() => _NewBookingPageState();
}

class _NewBookingPageState extends State<NewBookingPage> {
 DateTime? _selectedDate;
 TimeOfDay? _selectedTime;

 void _selectDate(BuildContext context) async {
        final DateTime? selected = await showDatePicker(
        context: context,
        initialDate: DateTime.now(),
        firstDate: DateTime.now(),
        lastDate: DateTime(DateTime.now().year + 1),
        );
        if (selected != null && selected != _selectedDate) {
        setState(() {
        _selectedDate = selected;
        });
        }
 }

 void _selectTime(BuildContext context) async {
        final TimeOfDay? selected =
        await showTimePicker(context: context, initialTime: TimeOfDay.now());
        if (selected != null && selected != _selectedTime) {
        setState(() {
        _selectedTime = selected;
        });
        }
 }

 @override
 Widget build(BuildContext context) {
        return Scaffold(
        appBar: AppBar(
        backgroundColor: Colors.black,
        title: Text('New Booking'),
        ),
        body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
        Text(
        'Select Date:',
        style: TextStyle(
                fontSize: 16,
                fontWeight: FontWeight.bold,
        ),
        ),
        SizedBox(height: 10),
        Row(
        children: [
                Expanded(
                child: Text(
                _selectedDate == null
                ? 'No date selected'
                : 'Selected date: ${_selectedDate!.toString().substring(0, 10)}',
                style: TextStyle(
                fontSize: 16,
                ),
                ),
                ),
                ElevatedButton(
                onPressed: () => _selectDate(context),
                child: Text('Select'),
                ),
        ],
        ),
        SizedBox(height: 20),
        Text(
        'Select Time:',
        style: TextStyle(
                fontSize: 16,
                fontWeight: FontWeight.bold,
        ),
        ),
        SizedBox(height: 10),
        Row(
        children: [
                Expanded(
                child: Text(
                _selectedTime == null
                ? 'No time selected'
                : 'Selected time: ${_selectedTime!.format(context)}',
                style: TextStyle(
                fontSize: 16,
                ),
                ),
                ),
                ElevatedButton(
                onPressed: () => _selectTime(context),
                child: Text('Select'),
                ),
        ],
        ),
        SizedBox(height: 40),
        ElevatedButton(
        onPressed: () {
                // Confirm booking and navigate back to dashboard
```

```
                    Navigator.pop(context);
                },
                child: Text('BOOK'),
                style: ElevatedButton.styleFrom(
                    primary: Colors.black,
                    onPrimary: Colors.white,
                    shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(10),
                    ),
                    padding:
                    EdgeInsets.symmetric(horizontal: 40, vertical: 15),
                ),
                ),
                ],
                ),
                ),
                );
  }
}


class LeaveParkingSlotPage extends StatefulWidget {
  @override
  _LeaveParkingSlotPageState createState() => _LeaveParkingSlotPageState();
}

class _LeaveParkingSlotPageState extends State<LeaveParkingSlotPage> {
  String name = 'Dhruv';
  String slotNumber = 'P21';
  String userType = 'STUDENT';
  String bookingDateTime = '2023-04-15 14:30:00'; // replace with actual booking datetime
  String timeLeft = '00:30:00'; // replace with actual time left for slot

  void leaveParkingSlot() {
        // Remove entry from list
        setState(() {
        // Remove entry from list
        });
        // Show notification
        ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
        content: Text('Parking slot left successfully!'),
        duration: Duration(seconds: 2),
        ),
        );
        // Navigate back to previous screen
        Navigator.of(context).pop();
  }

  @override
```

```
Widget build(BuildContext context) {
        return Scaffold(
        appBar: AppBar(
        title: Text('Leave Parking Slot'),
        ),
        body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
        Text(
        'Booked Slot Information',
        style: TextStyle(
                fontSize: 20.0,
                fontWeight: FontWeight.bold,
        ),
        ),
        SizedBox(height: 16.0),
        Text(
        'Name: $name',
        style: TextStyle(fontSize: 18.0),
        ),
        SizedBox(height: 8.0),
        Text(
        'Booked Slot: $slotNumber',
        style: TextStyle(fontSize: 18.0),
        ),
        SizedBox(height: 8.0),
        Text(
        'Type: $userType',
        style: TextStyle(fontSize: 18.0),
        ),
        SizedBox(height: 16.0),
        Text(
        'Booking Date and Time: $bookingDateTime',
        style: TextStyle(fontSize: 18.0),
        ),
        SizedBox(height: 8.0),
        Text(
        'Time Left for Slot: $timeLeft',
        style: TextStyle(fontSize: 18.0),
        ),
        SizedBox(height: 32.0),
        Center(
        child: ElevatedButton(
                onPressed: leaveParkingSlot,
                child: Text('Leave Parking Slot'),
        ),
        ),
        ]
```