

ONLINE PARKING SLOT BOOKING

Submitted in partial fulfilment of the required for the award of

Bachelor of Computer Science

By

Mohammed Malick. F (40290058)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC JEPPIAR NAGAR, RAJIV

GANDHI SALAI, CHENNAI 600199

MAY 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Mohammed Malick. F(40290058)** who carried out the project entitled **“ONLINE PARKING SLOT BOOKING”** under our -supervision from **JANUARY 2023 TO MAY 2023.**

INTERNAL GUIDE

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voce examination held on 08/05/2023

Internal Examiner

External Examiner

DECLARATION

I am, **Mohammed Malick. F (40290058)** hereby declare that the Project Report entitled "**ONLINE PARKING SLOT BOOKING**" done by us under the guidance of **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Arts and science** Department of Computer Science at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is submitted in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.



DATE:

Mohammed Malick.F

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Arts and Science** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Arts and Science** sincerement of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff member of the Department of Computer Science who were helpful in many ways for the completion of the project

Mohammed Malick.F

TABLE OF CONTENTS

Chapter No	Name of the topic	Page No
1	INTRODUCTION	1-7
	1.1 Project Review	
	1.2 Details about Parking Slot	
	1.3 Literature Review	
	1.4 Modules Description	
2	SYSTEM ANALYSIS	8-11
	2.1 Feasibility study	
	2.2 Existing System	
	2.3 Proposed System	
3	SYSTEM CONFIGURATION	12-26
	3.1 Hardware Requirments	
	3.2 Software Requirments	
	3.3 Python	
	3.4 Django == 4.2	
	3.5 Eclipse	
	3.6 PHP MySQL	
	3.7 Logical function	
	3.8 Mathematical equation	

4	SYSTEM DESIGN	27-32
	4.1 System Architecture	
	4.2 ER Diagram	
	4.3 Data Flow Diagram	
	4.4 Activity Diagram	
	4.5 Usecase Diagram	
5	CONCLUSION AND FUTURE WORK	33
6	APPENDIX	34-68
	6.1 Coding	
	6.2 Screenshots	
	6.3 Table Description	
7	REFERENCES	69-70

ABSTRACT

Here's an abstract for an online parking slot booking system that includes payment integration with popular services like Google pay , Phonepe, Paytm, Credit card and Debit card. Additionally, the system will generate a barcode that displays the booked parking slot. The system will be built using Python and will consist of two main components: a backend server and a frontend web application. The backend server will handle all the logic related to parking slot booking, payment processing, and barcode generation. It will use a database to store information about available parking slots, bookings, and payment transactions. Here mathematical equations and logical functions are used. The frontend web application will allow users to browse available parking slots and make bookings. The application will display real-time information about available parking slots and update the status of the slots as they are booked. Users will be able to select their preferred payment method and make payments securely using their preferred service. Upon successful payment, the system will generate a barcode that displays the booked parking slot, which users can scan to gain access to the parking lot. In conclusion, this online parking slot booking system will provide users with a convenient and secure way to book parking slot.

CHAPTER 1

INTRODUCTION

The parking place is very important all over the world especially in the cities of the countries. Every day thousands of car drivers spend a lot of the time to find where to park. The result of this situation is theft in urban areas, increasing traffic congestion and frustration of drivers. Inorder to solve this problem, the implementation of Online Vehicle Parking System in this city For managing parking places is mandatory. It will allow the drivers to Reserve a parking place on the Platform of People's Park Chennai (PPC) anytime, anywhere. this chapter gives an overview of the background of the study, statement of the problems, Objectives of the study, research questions, scope of the study and significance of the study. People's Park Chennai is located in Chennai district along Shollinganallur –OMRroad . It's located 1 mile from Accenture round about, People's Park Chennai Is constructed on 1 Acre of land (measurements 100ft-100ft) in Chennai, it provides Enough space for its well-wishing customers, It is equipped with high maximum security coupled With an electric fence , four gun men from a well training security company and well trained bull Dogs have been also used to control trespassing within the People's Park and with this maximum Security, expenses like compensation of customers' lost vehicles are brought down.

1.1 Project overview

People's Park Chennai had challenges concerning its safety of data in the store since they currently use paper based system physical struggle for parking by drivers, wastage of time, Congestion and collision. There was also a problem of monitoring the profit made for the company where by the company was losing money to its workers who receive the money (fraud). This system majorly solved the congestion, collision and save time during parking activities.

1.2 Detailed About Parking Slot

An online parking slot booking system using Python can be built using several frameworks and libraries such as Flask, Django, and SQLite for the database. Here is a brief overview of the steps involved in building such a system:

- Setup the development environment: Install Python and the required packages for the project, including Flask/Django, SQLite, and any additional libraries required.
- Design the database schema: Design the database schema to store the parking lot details, user details, and booking details. Create tables for each of these entities and establish relationships between them.
- Create the web application: Use the Flask/Django framework to create the web application. Define the routes for different pages, such as the home page, login page, booking page, and confirmation page.
- Create the user authentication system: Implement a user authentication system to allow users to create accounts, log in, and log out. Use Flask/Django's built-in authentication system or third-party libraries like Flask-Login or Django-Authentication to achieve this.
- Implement the booking system: Create a form to allow users to book a parking slot by specifying the parking lot, date, and time. Use Flask/Django's form handling capabilities to validate user input and store the booking details in the database.
- Display the booking details: Create a page to display the user's current and past bookings. Use SQL queries to fetch the booking details from the database and display them in a user-friendly format.
- Test and deploy the application: Test the application thoroughly to ensure that it works as expected. Once tested, deploy the application to a web server or cloud platform like Heroku.

This is a high-level overview of the steps involved in building an online parking slot booking system using Python.

Depending on the complexity of the system, there may be additional steps or requirements involved.

Parking Management includes a variety of strategies that encourage more efficient use of existing parking facilities, improve the quality of service provided to parking facility users and improve parking facility design. Our online reservation system to reserve parking spaces in the immediate parking, additional services and home purchase will increase your website by enabling customers to pay or go online. It is designed to make it easier for people to book parking spaces online. In today's parking lots there are no standard system to check for parking spaces. Searching for a vacant parking space in a metropolitan area is the daily concern for most people and it is time consuming. It commonly results more traffic load and air pollution in certain area only for an available parking space. This Python Project on Parking Management System is mostly concerned with dealing with client parking details such as number and slot. The system also allows vehicle owners to enter information such as their contact information, vehicle number, and vehicle category. However, after entering vehicle information, the system creates a reserve slot that lasts until the car leaves. When it comes to parking spaces, the system uses green and red to signify empty and occupied slots, accordingly. The system displays all parked vehicles under the manage vehicles area of the app, where the user can cancel the parking after it is completed.

1.3 LITERATURE SURVEY

1 .Date of Conference: 19-21 September 2019

Authors :Prasad Narode , Samruddhi Kalekar , Sanket Sanap , Harmeet Khanuja

The numbers of vehicles on roads are ever increasing, giving rise to a variety of problems. One of the problems is parking. Unorganized parking can lead to chaotic situations. As mentioned in [1] according to IBM Global Parking Survey, “30% of traffic in a city is caused by drivers searching for a parking spot. A lot of manpower is required to look after parking lots and hence a solution is presented to minimize the manpower, diverting it to more productive tasks and reducing the complexity involved in parking management task. This solution is a perfect combination of concepts of Internet of Things (IOT), machine learning and Android. Using REST based framework the system becomes loosely coupled from underlying architecture. The paper is organized as follows – section I contains introduction to our system, section II lists user side modules and parking lot management side modules one by one, section III contains traffic prediction module in greater depth focusing on technical implementation, section IV specifies server side management application, section V throws light on parking lot management side implementation, section VI gives the entire workflow, section VII gives the future scope and we conclude on section VIII.

2. Date of Conference: 14-17 October 2019

- Authors : Paul Melnyk , SoufieneDjahel , FaridNait-Abdesselam

Smart parking management aims to leverage advanced sensing and monitoring technologies to optimize the usage of parking spots within large car parks and ease parking for drivers by directing them to available parking spots, thereby reducing unnecessary stress and delay that further worsen the traffic conditions. The idea of smart parking development began to make parking easier, faster, with less of a drain on resources which are slowly disappearing. The aim of the Smart Parking Management System (SPMS) proposed in this paper is to assist drivers in alleviating several issues linked to parking. Our SPMS provides a sensing platform supported by a mobile app to enable real-time interaction of drivers with the car park infrastructure in a way that

minimizes drivers' search time for an empty parking spot within large multi-storey car parks and facilitates the localization of their parked cars. To validate the efficiency of our SPMS, a small scale test-bed was developed as a proof of concept and tested under five different scenarios. The obtained results highlight its effectiveness and call for its potential integration in real world car parks.

3. Date of conference: 17-20 February 2019

- Authors : NawsheenPromy, Sadman Islam.

Traffic congestion is a common phenomenon in developing countries like Bangladesh. This is the quotidian scenario in most of the metropolitan cities of the world. Because of heavy traffic, people lose their valuable time from their busy schedule. One of the prime reasons of this traffic congestion is parking on the roadside. So, a need arises to develop a parking system so that it can reduce the traffic congestion in near future. The paper focuses on a smart android based parking control application which will help you to find out a parking area in nearby your location. The mobile application will also help you to pay the parking charge by online payment system. A counter will be there in the application which will help you to count the amount of time your car have spent in the parking lot. This application will also track the entire process and all the previous records.

4 Date of conference: 29 November 2019 – 01 December 2019.

- Authors: LomatHaider Chowdhury, Z.N.M Zarif Mahmud, Intishar-UI Islam, Ishrat Jahan, Salekul Islam

The demand for vehicles increases with the increase of the population. A densely replenished country like Bangladesh faces many challenges in managing this increased number of vehicles. Especially in city areas, where the road-side parking is not legitimate. Parking management can play an important role to diminish congestion on the roads. In our country, most of the parking areas are maintained by a manual parking system. In this paper, our aim is to design and develop a smart car parking system to solve the chaos, perplexity and long queues at the entry and exit of a parking space located inside

public buildings including shopping malls and office spaces. In the smart car parking system, a radio frequency identification (RFID) card is used for every vehicle to store the information of the entrance. Time is automatically enumerated from the entry time to exit time, and thus fare will be shown to the client for his used parking space. In this paper, we have developed a system composed of both hardware and software components. Our system provides a user-friendly interface for the employees who maintain the system. The system also generates different reports including revenue and usage reports directly from the database using the software section. There will be an LED display in the parking that will show the number of available parking slots. Finally, we have measured the performance in terms of response time by deploying the system in the local server and remote Cloud server.

1.4 MODULE DESCRIPTION

User login:

Users have to first register themselves to login into the system. In the registration of user page should be filled with the details of Name, Data of Birth, Email ID, Gender, Phone Number, Address and Password. After clicking sign up, the user registration will be made successful. With the details of registration made, the user login should be done. Checking the authorization, the user will be logged in to the system. After logging it, the user can able to see the Parking Cost, Book Parking, Your Booking details.

Admin Login:

The system is under supervision of admin who manages the bookings made. Once after the admin logged it, the admin can see the parking cost, View User details and View Bookings. Only the admin has the privilege of modifying the parking cost. Also admin can view the user details. And also the admin can view the complete booking details with the cost etc.

Parking availability:

User can click on spaces to view the availability. If the space is already booked it will be marked yellow and the available ones will be seen in normal color. Parking booking online for date and time: Users can book parking space for their required date and time. User cannot select the slot if the booking is already done by some other earlier.

Automatic cost calculation:

The system calculates the total cost incurred for parking based on the time that user has asked for booking. In the user login, the option: Book Parking is available. In this the user enters the details of Date, time, and parking hours. After entering the details, the hours are calculated automatically and the final amount is displayed. The user no need to calculate the amount and enter it manually.

CHAPTER 2

SYSTEM ANALYSIS

2.1 Feasibility study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are:

- i. Economical Feasibility
- ii. Technical Feasibility
- iii. Social Feasibility
- iv. Operational Feasibility

2.1.1 Economic feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.1.2 Technical feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high

demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.1.3 Social feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

2.1.4 Operational feasibility

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantages of the opportunities identified during scope definition and how it satisfies the requirements identified in 8 the requirements identified in the requirements analysis phase of system development.

2.2 Existing system

Token Based Parking Management System (PMS) dispenses the parking token & records entry and exit information of the parking. These tokens can be reused hence eliminating the recurring cost of ticket roll. CASS token based PMS system are very popular among most of the Hypermarkets / Supermarkets, Hospitals, Shopping Malls, Hotels, parking lots & in mixed-use properties. In metropolitan cities where the population is at its peak and the roads are all messed with vehicle's and long traffics. In such over increasing demand for vehicle's it becomes a difficult job to find a parking spot for our wheels. Due to this people are forced to haphazardly park their vehicle wherever they find

space, which results in traffic offence like illegal, haphazard double or angular parking. In the existing system the parking management were all made in Manual mode, so it's unable to make the process easy and smooth. The existing system of parking management does not consider the parking timing slots where they can give bigger confusions and not efficiently utilized the space.

Disadvantages of existing system

Limited availability:

If a parking slot booking system is not widely adopted or popular in a given area, drivers may find it difficult to find an available parking space.

Inflexibility:

If a driver books a parking spot but then changes their plans or arrives late, they may miss their reservation and lose their money.

Technical issues:

If the booking system experiences technical difficulties, drivers may not be able to reserve a parking spot or may be charged incorrectly.

Privacy concerns:

Some people may have concerns about the privacy and security of their personal information when using a parking slot booking system.

Overbooking:

If a parking slot booking system overbooks the available parking spaces, drivers may find that their reserved spot is already occupied when they arrive.

2.3 PROPOSED SYSTEM

The proposed project is an Online parking booking system that provides customers an easy way of reserving a parking space online. It overcomes the problem of finding a parking space in commercial areas that unnecessary consumes time. Hence this project offers a web based reservation system where users can view various parking areas and select the space to view whether space is available or not. If the booking space is available then he can book it for specific time slot. The booked space will be marked yellow and will not be available for anyone else for the specified time. Users can even make payment online. The project can be implemented in commercial areas for employee parking. It can be utilized by companies and organizations (hospitals, schools, colleges) to automate their parking system. The system can also be used in public places for public parking like in malls, station, and so on.

Advantages of proposed system

Users can get learn about parking areas for particular locations.It saves user time in search of parking space available in such a long parking area.The system provides a graphical view of the parking spaces.User can pay online on the spot and confirm their space.It excludes the need of human efforts for managing parking spaces.Cost-effective.The proposed system provides better information for the users of this system easily they can see the available slots and book from anywhere without facing any difficulty.The proposed system does not require any physical presence during booking.The proposed system has good authentication so only authorized person can able to login and book it.

CHAPTER 3

SYSTEM CONFIGURATION

3.1 HARDWARE REQUIRMENTS

Hardware	Minimum Requirements	Reason
Processor speed	1.6GHZ	Accommodate most PC's
Memory of user PC	512MB RAM	Relatively fast
Disk space of user PC	5GB	Adequate storage capacity
Memory of server PC	8GB	Fast
Bandwidth (networkconnection)	15Mbps	Relatively good
Disk space of server	200GB	Adequate storage for data base and applications.

A fast processor is required because there is need to handle large amounts of data queries. 8 GB memory is required on the server for faster performance because it runs many processes simultaneously, while the memory in the computer should be relatively fast so as to run the processes required. 200 GB storage capacity in the server's hard disk is necessary for storage of huge amounts of data while the disk space in a user computer should be modest enough. Good network connection is vital because the application is majorly internet based and there is need for faster communication and retrieval of information.

3.2 SOFTWARE REQUIRMENTS

Software	Minimum Requirements	Reason
Operating systems for computer	Windows 8,10, Linux, Mac, Windows xp	Globally distributed and widely accessed
Database Management System	MySQL	Easy to use and scalable
Browser	Chrome, Internet explorer	Standard browser

3.3 PYTHON

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a “batteries included” language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely backward-compatible with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2. Python consistently ranks as one of the most popular programming languages. Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his “permanent vacation” from his responsibilities as Python's "benevolent dictator for life”, a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member Steering Council to lead the project. Python 2.0 was released on 16 October 2000, with many major new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released on 3 December 2008, with many of its major features backported to Python 2.6.x and 2.7.x. Releases of Python 3 include the 2to3 utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life was initially set for 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No further security patches or other improvements will be released for it. Currently only 3.7 and later are supported. In 2021, Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues leading to possible remote code execution

and web cache poisoning. In 2022, Python 3.10.4 and 3.9.12 were expedited and 3.8.13, and 3.7.13, because of many security issues. When Python 3.9.13 was released in May 2022, it was announced that the 3.9 series (joining the older series 3.8 and 3.7) would only receive security fixes in the future. On September 7, 2022, four new releases were made due to a potential denial-of-service attack: 3.10.7, 3.9.14, 3.8.14, and 3.7.14. As of November 2022, Python 3.11 is the stable release. Notable changes from 3.10 include increased program execution speed and improved error reporting. Python is meant to be an easily readable language. Its formatting is visually uncluttered and often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are allowed but



rarely used. It has fewer syntactic exceptions and special cases than C or Pascal.

Indentation Main article: [Python syntax and semantics § Indentation](#)

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation signifies the end of the current block. Thus, the program's visual structure accurately represents its semantic structure. This feature is sometimes termed the off-side rule. Some other languages use indentation this way; but in most, indentation has no semantic meaning. The recommended indent size is four spaces.

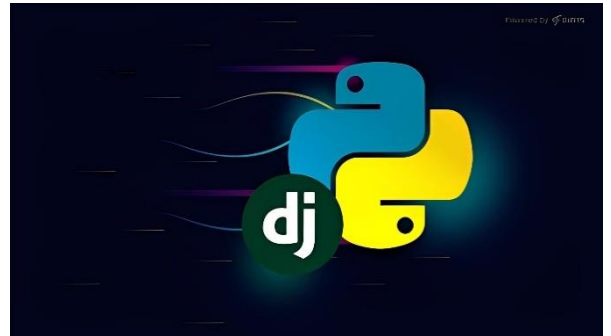
3.4 DJANGO == 4..2

Django (/ˈdʒæŋɡoʊ/ JANG-goh; sometimes stylized as django)[6] is a free and open-source, Python-based web framework that follows the model–template–views (MTV) architectural pattern.[7][8] It is maintained by the Django Software Foundation (DSF), an independent organization established in the US as a 501(c)(3) non-profit. Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "Pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself.[9] Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models. Some well-known sites that use Django include Instagram, Mozilla, Disqus, Bitbucket, Next door and Clubhouse. Django was created in the fall of 2003, when the web programmers at the Lawrence Journal-World newspaper, Adrian Holovaty and Simon Willison, began using Python to build applications. Jacob Kaplan-Moss was hired early in Django's development shortly before Simon Willison's internship ended. It was released publicly under a BSD license in July 2005. The framework was named after guitarist Django Reinhardt. Adrian Holovaty is a Romani jazz guitar player and a big fan Django Reinhardt.[citation needed]In June 2008, it was announced that a newly formed Django Software Foundation (DSF) would maintain Django in the future. Programmers have ported Django's template engine design from Python to other languages, providing decent cross-platform support. Some of these options are more direct ports; others, though inspired by Django and retaining its concepts, take the liberty to deviate from

Django's design:

- Liquid for Ruby
- Template::Swig for Perl
- Twig for PHP and JavaScript

- Jinja for Python
- ErlyDTL for Erlang.



Django as a framework is capable of building a complete CMS, however there are dedicated CMS projects which are built upon and extend the Django framework. Below is a list of a few of the more popular Django-based CMSs:

- Django CMS
- Wagtail.

3.5 ECLIPSE

Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development, and, until 2016, was the most popular. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#, Clojure, COBOL, D, Erlang, Fortran, Groovy, Haskell, JavaScript, Julia, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Rust, Scala, and Scheme. It can also be used to develop documents with LaTeX (via a TeXlipse plug-in) and packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others. The initial codebase originated from IBM VisualAge. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles. Eclipse software development kit (SDK) is free and open-source software, released under the terms of the Eclipse Public License, although it is incompatible with the GNU General Public License. It was one of the first IDEs to run under GNU Classpath and it runs without problems under IcedTea. Eclipse was inspired by the Smalltalk-based VisualAge family of integrated development environment (IDE) products. Although fairly successful, a major drawback of the VisualAge products was that developed code was not in a component-based software engineering model. Instead, all code for a project was held in a compressed database using SCID techniques (somewhat like a zip file but in a proprietary format called .dat). Individual classes could not be easily accessed, certainly not outside the tool. A team primarily at the IBM Cary, NC lab developed the new product as a Java-based replacement.[12][failed verification] In November 2001, a consortium was formed with a board of stewards to further the development of Eclipse as open-

source software. It is estimated that IBM had already invested nearly \$40 million by that time. The original members were Borland, IBM, Merant, QNX Software Systems, Rational Software, Red Hat, SuSE, TogetherSoft, and WebGain. The number of stewards increased to over 80 by the end of 2003. In January 2004, the Eclipse Foundation was created. Eclipse 3.0 (released on 21 June 2004) selected the OSGi Service Platform specifications as the runtime architecture. The Association for Computing Machinery recognized Eclipse with the 2011 ACM Software Systems Award on 26 April 2012. Eclipse uses plug-ins to provide all the functionality within and on top of the run-time system. Its run-time system is based on Equinox, an implementation of the OSGi core framework specification. In addition to allowing the Eclipse Platform to be extended using other programming languages, such as C and Python, the plug-in framework allows the Eclipse Platform to work with typesetting languages like LaTeX and networking applications such as telnet and database management systems. The plug-in architecture supports writing any desired extension to the environment, such as for configuration management. Java and CVS support is provided in the Eclipse SDK, with support for other version control systems provided by third-party plug-ins. With the exception of a small run-time kernel, everything in Eclipse is a plug-in.

Thus, every plug-in developed integrates with Eclipse in the same way as other plug-ins; in this respect, all features are "created



equal". Eclipse provides plug-ins for a wide variety of features, some of which are from third parties using both free and commercial models. Examples of plug-ins include for Unified Modeling Language (UML), for Sequence and other UML diagrams, a plug-in for DB Explorer, and many more. The Eclipse SDK includes the Eclipse Java development tools (JDT), offering an IDE with a built-in Java incremental compiler and a full model of the Java source files. This allows for advanced refactoring techniques and code analysis. The IDE also makes use of a workspace, in this case a set of metadata over a flat filesystem allowing external file modifications as long as the corresponding workspace resource is refreshed afterward. Eclipse implements the graphical control elements of the Java toolkit called Standard Widget Toolkit (SWT), whereas most Java applications use the Java standard Abstract Window Toolkit (AWT) or Swing. Eclipse's user interface also uses an intermediate graphical user interface layer called JFace, which simplifies the construction of applications based on SWT. Eclipse was made to run on Wayland during a Google Summer of Code (GSoC) Project in 2014. As of 2017, language packs being developed by the Babel Project provide translations into over 40 natural languages. Eclipse Web Tools Platform (WTP) project is an extension of the Eclipse platform with tools for developing Web and Java EE applications. It includes source and graphical editors for a variety of languages, wizards and built-in applications to simplify development, and tools and APIs to support deploying, running, and testing apps. Android Development Tools (ADT) was superseded in 2015 by the Eclipse foundation's own plugin, called Andmore: Development Tools for Android, after Google discontinued development of their plug-in for the Eclipse IDE, that is designed to provide an integrated environment in which to build Android applications. ADT/Andmore extends the abilities of Eclipse to let developers set up new Android projects, create an application UI, add packages based on the Android Framework API, debug their applications using the Android SDK tools, and export signed (or unsigned) .apk files in order to distribute their applications. It is freely available to download. Google's ADT was the official IDE for Android until 2015 but was replaced by Eclipse's Andmore and the official Android Studio. Eclipse supports development for Tomcat, GlassFish and many other servers and is often capable of installing the required server (for development) directly from the IDE. It supports remote debugging, allowing a user to watch variables and step through the code of an application that is running on the attached server.

3.6 PHP MYSQL

PHP (Hypertext Pre-processor) and MySQL are two widely used technologies for building dynamic websites and web applications. PHP is a server-side scripting language that is used to generate HTML pages on the fly, while MySQL is a relational database management system that is used to store and retrieve data. When used together, PHP and MySQL provide a powerful combination for building dynamic web applications. PHP is used to generate dynamic content, such as user profiles, product pages, and search results, while MySQL is used to store and retrieve data, such as user information, product details, and search queries. PHP provides a range of functions and libraries for working with MySQL databases, such as connecting to the database, executing queries, and handling errors. These functions and libraries make it easier to work with MySQL databases and reduce the amount of boilerplate code that is required.

Some of the key features of PHP and MySQL include:

Compatibility: PHP and MySQL are compatible with a wide range



of web servers and operating systems, making them a popular choice for building web applications.

Speed: PHP is a fast scripting language that is optimized for web development, while MySQL is a fast database management system that can handle large volumes of data.

Security: PHP and MySQL have built-in security features, such as password encryption, to help protect sensitive data and prevent unauthorized access.

Scalability: PHP and MySQL are scalable technologies that can handle a large number of concurrent users and requests.

Open-source: PHP and MySQL are both open-source technologies, meaning that their source code is freely available and can be modified and distributed by anyone. Overall, PHP and MySQL are powerful technologies for building dynamic websites and web applications. Their compatibility, speed, security, scalability, and open-source nature make them a popular choice for developers around the world.

3.7 LOGICAL FUNCTIONS

Suppose you want to design a parking system. A parking lot has three different kinds of parking spaces – big, medium, and small. And there are fixed number of slots for each size. Make a class called Our Parking System with of two methods –

constructor(big, medium, small) – This constructor is taking the number of slots available for different spaces and initializes object of the Our Parking System class.

addCar(carType) – This method checks whether there is a parking space of given car Type for the car that wants to put inside the parking lot.

The three slots big, medium, or small, are represented by 1, 2, and 3 respectively. The constraint is a car can only park in a parking space if car Type is matched. If there is no space available, return false, otherwise park the car in that size space and return true. If there are 2 spaces for big car, no space for medium car and 1 space for small car, then constructor call will be like OurParkingSystem(2, 0, 1), and if we call addCar like –

addCar(3) – add one small car and return true

addCar(2) – No space to add medium car so return false

addCar(3) – No space to add a new small car so return false

addCar(1) – add one big car and return true

addCar(1) – add another big car and return true

addCar(1) – No space for another big car so return false

To solve this, we will follow these steps –

Define a function constructor() . This will take big, medium, small
sp := a list like [0, big, medium, small]

Define a function addCar() . This will take carType

if sp[carType] > 0, then

```
sp[carType] := sp[carType] - 1
return True
return False
```

Example (Python)

Let us see the following implementation to get better understanding –

Live Demo

```
class OurParkingSystem:
def __init__(self, big, medium, small):
self.sp = [0, big, medium, small]

def addCar(self, carType):
    if(self.sp[carType] > 0 ):
self.sp[carType] -= 1
        return True
        return False
```

```
ps = OurParkingSystem(2, 0, 1)
print(ps.addCar(3))
print(ps.addCar(2))
print(ps.addCar(3))
print(ps.addCar(1))
print(ps.addCar(1))
print((1))
```

Input

```
ps.addCar(3)
ps.addCar(2)
ps.addCar(3)
ps.addCar(1)
ps.addCar(1)
```

```
ps.addCar(1)
```

Output:

True

False

False

True

True

False

3.8 MATHEMATICAL EQUATIONS

Count maximum number of cars parked at the same time

Given a 2d array `arr[][]` with each row representing a pair representing entry and exit time of a car in a parking lot, the task is to calculate the maximum number of cars that can be parked at the same time.

Examples:

Input: `arr[][] = {{1012, 1136}, {1317, 1417}, {1015, 1020}}`

Output: 2

Explanation:

1st car entered at 10:12 and exits at 11:36 and 3rd car entered at 10:15 and exits at 10:20.

Therefore, 1st and 3rd car are present at the same time.

Input: `arr[][] = {{1120, 1159}, {1508, 1529}, {1508, 1527}, {1503, 1600}, {1458, 1629}, {1224, 1313}}`

Output: 4

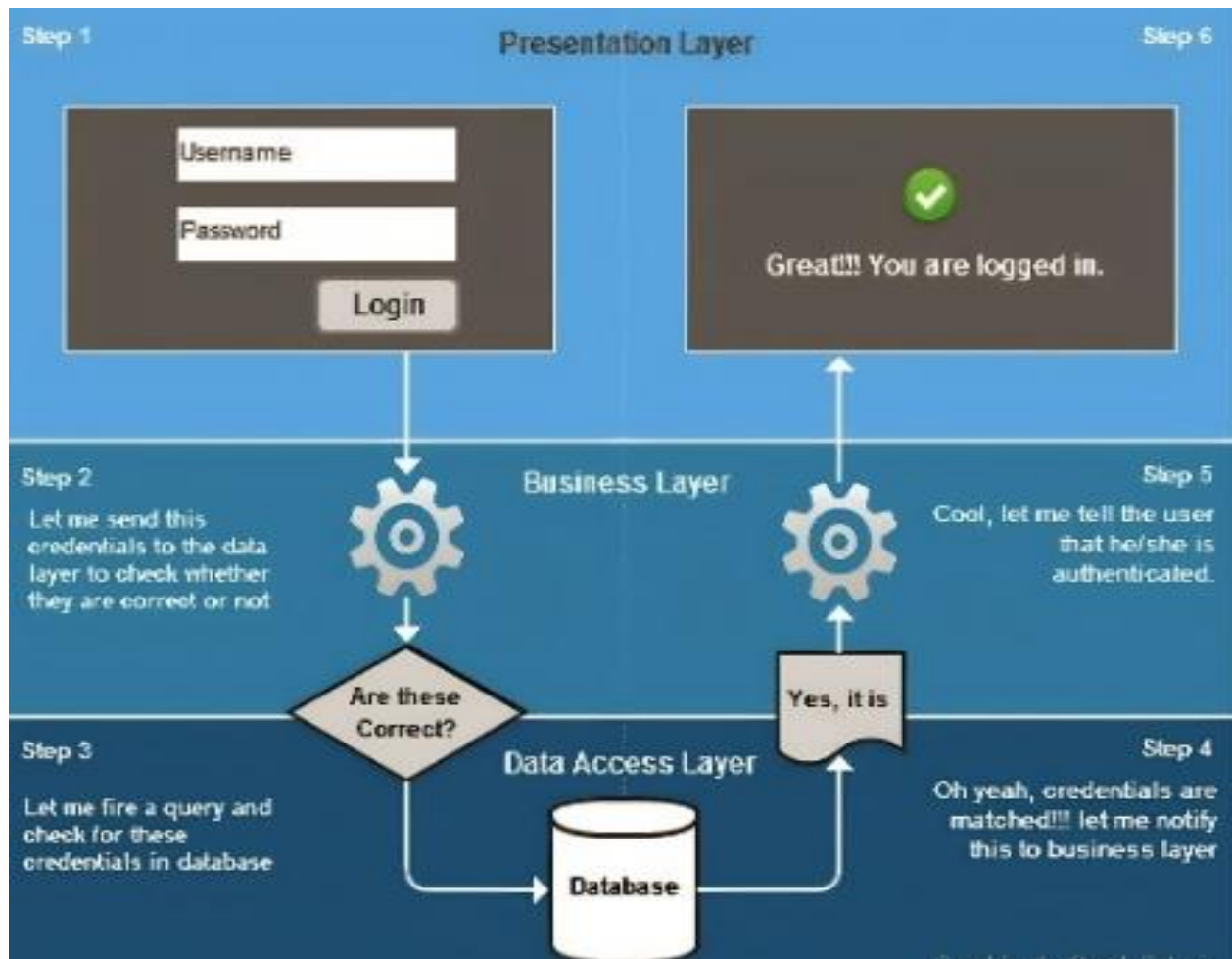
Explanation: 2nd, 3rd, 4th and 5th cars are present at the same time.

Find your parking ratio by dividing the number of spaces by the building's square footage (in thousands). For instance, take a 40,000-square-foot building with a 200-space parking lot. Divide 200 (spaces) by 40 (thousand square feet) to find a parking ratio of 5 spaces per 1,000 square feet.

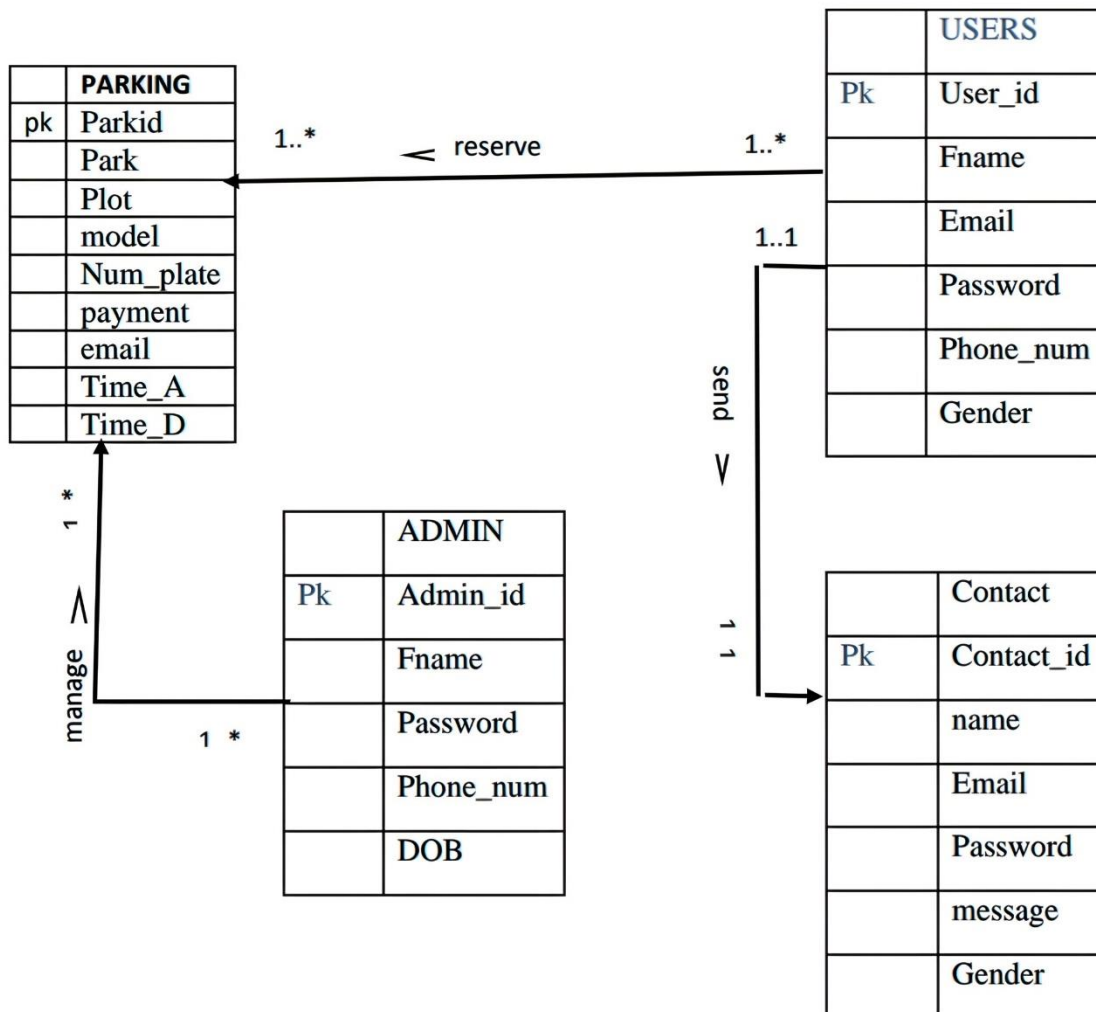
CHAPTER 4

SYSTEM DESIGN

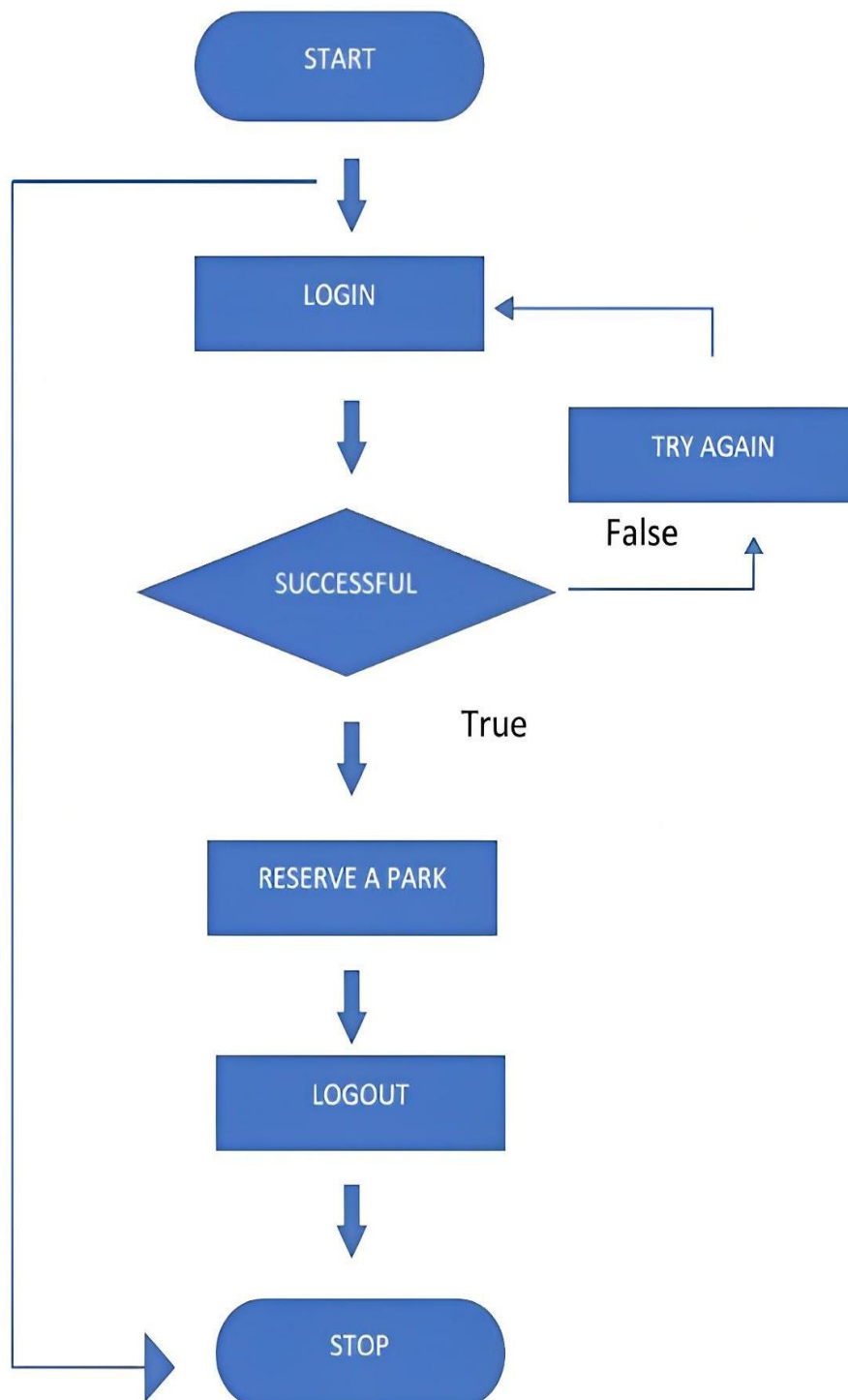
4.1 SYSTEM ARCHITECTURE



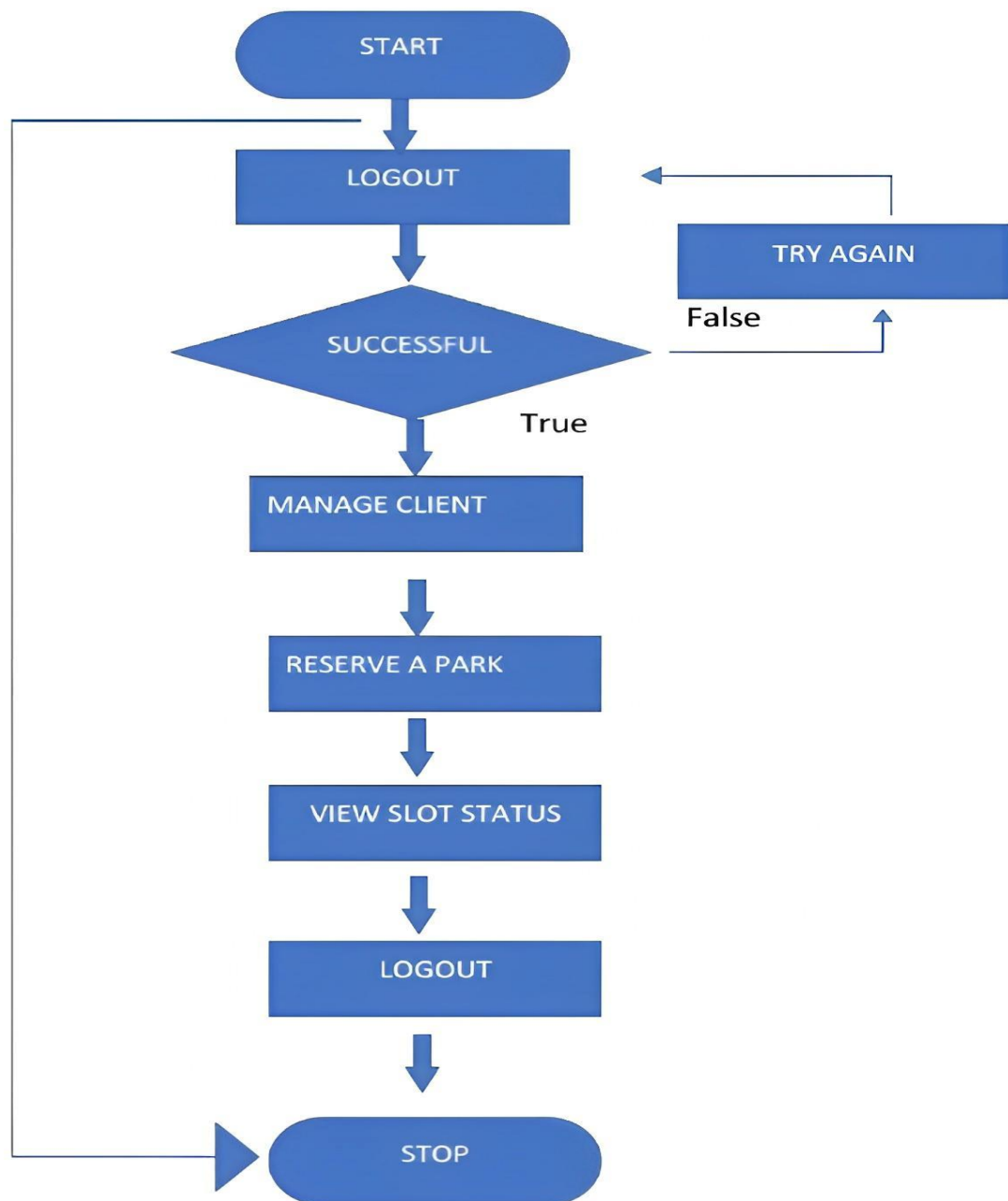
4.2 ER DIAGRAM



4.3 DATA FLOW DIAGRAM

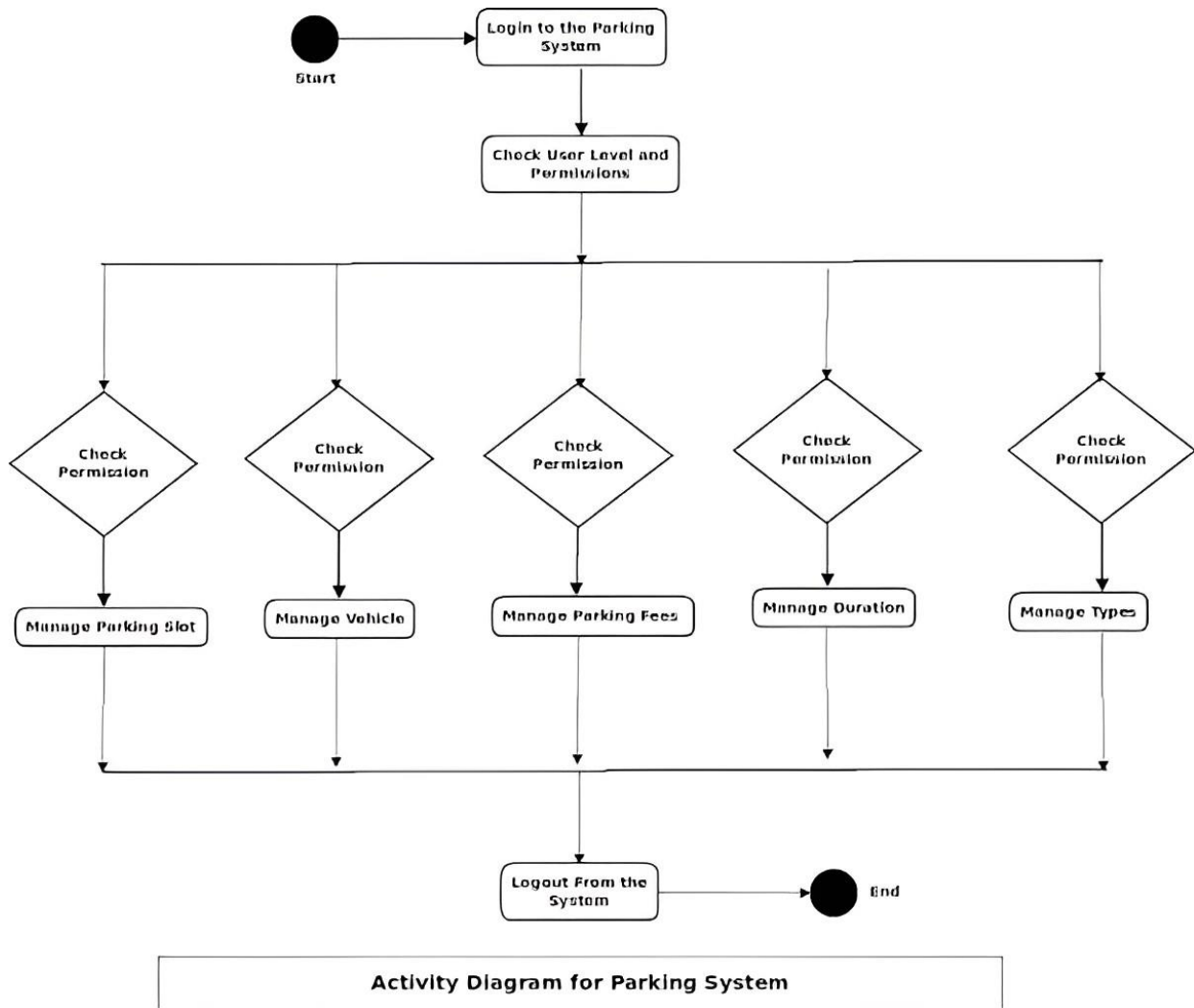


Client side Flow chart



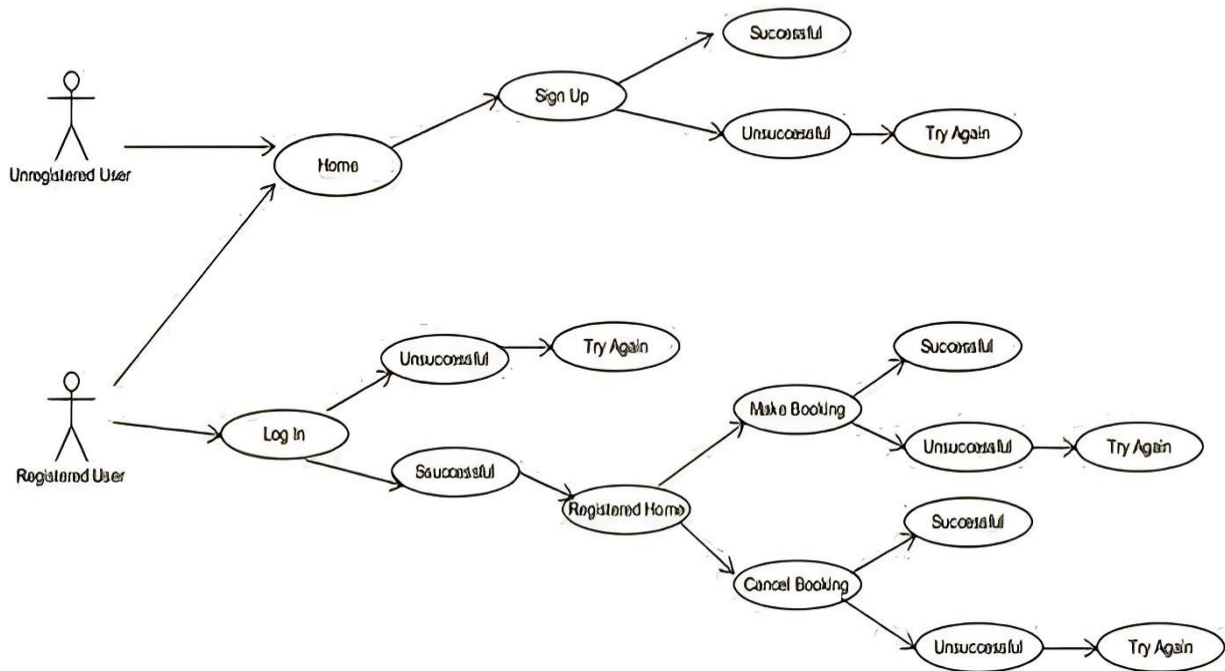
Admin side Flow chart

4.4 ACTIVITY DIAGRAM



4.5 USECASE DIAGRAM

Use Case Diagram



CHAPTER 5

CONCLUSION AND FUTURE WORK

Online vehicle parking reservation system improves the existing system since we are in Computerized world. With this new system is mandatory, it enables the user of the system (client,Employee, System administrator) to reserve a parking lot online and this reduces the wasting of Time of the clients looking for where to park, increase the safety of the property since the parking lot is numbering. The difficulty of searching available parking lots has been completely eliminated by receiving lots via the proposed system. User can get learn about parking areas for particular location. It saves user time in search of parking space available in such long parking area. In the further development of this Android app plan to make the app more advance by adding more support and additional functions to the system. Also plan to use credit card payment gateway instead of Gpay and Paytm.

CHAPTER 6

APPENDIX

6.1 CODING

Login:

```
{% load static %}
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>
```

Parking Management System

```
</title>
```

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
```

```
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
```

```
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlIQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
```

```
crossorigin="anonymous"></script>
```

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
```

```
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
```

```
GLhITQ8iRABdZLI6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
```

```
crossorigin="anonymous">
```

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

```
<!-- Link to Google Fonts in the HTML head -->
```

```
<link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap"
rel="stylesheet">
```

```
<link href="https://fonts.google.com/specimen/Crimson+Text">
```

```
<style>
```

```
/* Use the imported font in CSS */
```

```
body {
```

```
font-family: 'Open Sans', sans-serif;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<section class="vh-100">
```

```
<div class="container-fluid h-custom">
```

```
<div class="row d-flex justify-content-center align-items-center h-100">
```

```
<div class="col-md-9 col-lg-6 col-xl-5">
```

```
{% load static %}

</div>

<div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">

<form action="" method="POST">

{% csrf\_token %}

<div class="divider d-flex align-items-center my-4">

<p class="text-center fw-bold mx-3 mb-0">Parking application Log In </p>

</div>

<!-- username input -->

<div class="form-outline mb-4">

<input type="text" name="username" id="form3Example3" class="form-control

form-control-lg"

placeholder="Enter a Username" />

<label class="form-label" for="form3Example3">Username</label>

</div>

<!-- Password input -->

```

<div class="form-outline mb-3">

<input type="password" name="password" id="form3Example4" class="form-control
form-control-lg"
placeholder="Enter password" />

<label class="form-label" for="form3Example4">Password</label>

</div>

<div class="text-center text-lg-start mt-4 pt-2">

<button type="submit" class="btn btn-primary btn-lg"
style="padding-left: 2.5rem; padding-right: 2.5rem;">Login</button>

{{form.errors}}

<p class="small fw-bold mt-2 pt-1 mb-0">Don't have an account? <a href="{% url
'signup' %}"
class="link-danger">Register</p>

</div>

</form>

</div>

</div>

</div>

<div

```

```
class="d-flex flex-column flex-md-row text-center text-md-start justify-content-between
py-4 px-4 px-xl-5 bg-primary">
```

```
<!-- Copyright -->
```

```
<div class="text-white mb-3 mb-md-0">
```

```
Copyright © 2023. All rights reserved.
```

```
</div>
```

```
<!-- Copyright -->
```

```
<style>
```

```
.divider:after,
```

```
.divider:before {
```

```
content: "";
```

```
flex: 1;
```

```
height: 1px;
```

```
background: #eee;
```

```
}
```

```
.h-custom {
```

```
height: calc(100% - 73px);
```

```
}
```

```
@media (max-width: 450px) {
```

```
.h-custom {
```

```
height: 100%;
```

```
}
```

```
}
```

```
</style>
```

```
</section>
```

```
</body>
```

```
</html>
```

Sign up

```
{% load static %}
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>
```

Parking Management System

```
</title>
```

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
```

```
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
```

```
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlIQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
```

```

crossorigin="anonymous"></script>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhITQ8iRABdZLI6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">

<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">

<!-- Link to Google Fonts in the HTML head -->

<link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap"
rel="stylesheet">

<link href="https://fonts.google.com/specimen/Crimson+Text">

<style>

/* Use the imported font in CSS */

body {

font-family: 'Open Sans', sans-serif;

}

</style>

</head>

<body>

<section class="vh-100">

```

```
<div class="container-fluid h-custom">
```

```
<div class="row d-flex justify-content-center align-items-center h-100">
```

```
<div class="col-md-9 col-lg-6 col-xl-5">
```

```
{% load static %}
```

```
</div>
```

```
<div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
```

```
<form action="" method="POST">
```

```
{% csrf_token %}
```

```
<div class="divider d-flex align-items-center my-4">
```

```
<p class="text-center fw-bold mx-3 mb-0">Parking application Signup </p>
```

```
</div>
```

```
<!-- username input -->
```

```
<div class="form-outline mb-2">
```

```
<label class="form-label" for="form3Example3">Username</label>
```

```

<input type="text" name="username" id="form3Example3" class="form-control
form-control-lg"
placeholder="Enter a Username" />

</div>

<label class="form-label" for="form3Example3">Email</label>

<div class="form-outline mb-2">

<input type="email" name="email" id="form3Example3" class="form-control form-
control-lg"
placeholder="Enter a Email" />

</div>

<!-- Password input -->

<div class="form-outline mb-2">

<label class="form-label" for="form3Example4">Password</label>

<input type="password" name="password1" id="form3Example4" class="form-
control form-control-lg"
placeholder="Enter password" />

</div>

<div class="form-outline mb-2">

<label class="form-label" for="form3Example4">Re-Enter Password</label>

```

```

<input type="password" name="password2" id="form3Example4" class="form-
control form-control-lg"
placeholder="Re-Enter password" />

</div>

<div class="text-center text-lg-start mt-4 pt-2">

<button type="submit" class="btn btn-primary btn-lg"
style="padding-left: 2.5rem; padding-right: 2.5rem;">Signup</button>

<p class="small fw-bold mt-2 pt-1 mb-0">if you have an account? <a href="{% url 'login'
%}"
class="link-danger">Login</p>

</div>

</form>

</div>

</div>

</div>

<div
class="d-flex flex-column flex-md-row text-center text-md-start justify-content-between
py-4 px-4 px-xl-5 bg-primary">

<!-- Copyright -->

```

```
<div class="text-white mb-3 mb-md-0">
```

Copyright © 2023. All rights reserved.

```
</div>
```

```
<!-- Copyright -->
```

```
<style>
```

```
.divider:after,
```

```
.divider:before {
```

```
content: "";
```

```
flex: 1;
```

```
height: 1px;
```

```
background: #eee;
```

```
}
```

```
.h-custom {
```

```
height: calc(100% - 73px);
```

```
}
```

```
@media (max-width: 450px) {
```

```
.h-custom {
```

```
height: 100%;
```

```
}
```

```
}
```

```
</style>
```

```
</section>
```

```
</body>
```

```
</html>
```

Payment:

```
<div class="container p-0">
```

```
<style>
```

```
@import
```

```
url('https://fonts.googleapis.com/css?family=Montserrat:400,700&display=swap');
```

```
* {
```

```
margin: 0;
```

```
padding: 0;
```

```
box-sizing: border-box;
```

```
font-family: 'Montserrat', sans-serif;
```

```
}
```

```
body {
```

```
display: flex;
```

```
justify-content: center;
```

```
align-items: center;
```

```
min-height: 100vh;

background-color: #0C4160;

padding: 30px 10px;

}

.card {

max-width: 500px;

margin: auto;

color: black;

border-radius: 20 px;

}

p {

margin: 0px;

}

.container .h8 {

font-size: 30px;

font-weight: 800;

text-align: center;

}

.btn.btn-primary {
```

```
width: 100%;

height: 70px;

display: flex;

align-items: center;

justify-content: space-between;

padding: 0 15px;

background-image: linear-gradient(to right, #77A1D3 0%, #79CBCA 51%, #77A1D3

100%);

border: none;

transition: 0.5s;

background-size: 200% auto;

}

.btn.btn.btn-primary:hover {

background-position: right center;

color: #fff;

text-decoration: none;

}

.btn.btn-primary:hover .fas.fa-arrow-right {

transform: translate(15px);
```

```
transition: transform 0.2s ease-in;
```

```
}
```

```
.form-control {
```

```
color: white;
```

```
background-color: #223C60;
```

```
border: 2px solid transparent;
```

```
height: 60px;
```

```
padding-left: 20px;
```

```
vertical-align: middle;
```

```
}
```

```
.form-control:focus {
```

```
color: white;
```

```
background-color: #0C4160;
```

```
border: 2px solid #2d4dda;
```

```
box-shadow: none;
```

```
}
```

```
.text {
```

```
font-size: 14px;
```

```
font-weight: 600;
```

```
}
```

```
::placeholder {
```

```
font-size: 14px;
```

```
font-weight: 600;
```

```
}
```

```
</style>
```

```
<div class="card px-4">
```

```
<p class="h8 py-3">Payment Details</p>
```

```
<div class="row gx-3">
```

```
<div class="col-12">
```

```
<div class="d-flex flex-column">
```

```
<p class="text mb-1">Person Name</p>
```

```
<input class="form-control mb-3" type="text" placeholder="Name" value="">
```

```
</div>
```

```
</div>
```

```
<div class="col-12">
```

```
<div class="d-flex flex-column">
```

```
<p class="text mb-1">Card Number</p>
```

```
<input class="form-control mb-3" type="text" placeholder="1234 5678"
```

435678">

</div>

</div>

<div class="col-6">

<div class="d-flex flex-column">

<p class="text mb-1">Expiry</p>

<input class="form-control mb-3" type="text" placeholder="MM/YYYY">

</div>

</div>

<div class="col-6">

<div class="d-flex flex-column">

<p class="text mb-1">CVV/CVC</p>

<input class="form-control mb-3 pt-2 " type="password" placeholder="\*\*\*\*">

</div>

</div>

<div class="col-12">

<div class="btn btn-primary mb-3">

<a href="{% url 'success' %}"><span class="ps-3" >Pay ₹500.00</span></a>

<span class="fas fa-arrow-right"></span>

</div>

</div>

</div>

</div>

</div>

Models

```
from django.db import models
```

```
class ParkingSlot(models.Model):
```

```
 LOCATION_CHOICES = (
```

```
 ('middle', 'Middle'),
```

```
 ('right', 'Right'),
```

```
 ('left', 'Left'),
```

```
 ('center', 'Center'),
```

```
 ('UB', 'Upper Basement'),
```

```
 ('LB', 'Lower Basement'),
```

```
)
```

```
 slot_number = models.IntegerField(default=0)
```

```
 location = models.CharField(max_length=20, choices=LOCATION_CHOICES)
```

```
 def __str__(self):
```

```

return str(self.slot_number)

class Meta:

 ordering = ['slot_number']

 def save(self, *args, **kwargs):

 if not self.id:

 # This is a new instance, set the slot number

 last_slot = ParkingSlot.objects.order_by('-slot_number').first()

 if last_slot:

 self.slot_number = last_slot.slot_number + 1

 else:

 self.slot_number = 1

 super(ParkingSlot, self).save(*args, **kwargs)

class BookSlot(models.Model):

 name=models.CharField(max_length=100)

 make = models.CharField(max_length=100)

 model = models.CharField(max_length=100)

 slotnumber=models.OneToOneField(ParkingSlot, on_delete=models.SET_NULL,

 null=True, blank=True)

 license_plate = models.CharField(max_length=100, unique=True)

```

## Settings

.....

Django settings for parkingProject project.

Generated by 'django-admin startproject' using Django 4.1.5.

For more information on this file, see

<https://docs.djangoproject.com/en/4.1/topics/settings/>

For the full list of settings and their values, see

<https://docs.djangoproject.com/en/4.1/ref/settings/>

.....

```
import os
```

```
from pathlib import Path
```

```
Build paths inside the project like this: BASE_DIR / 'subdir'.
```

```
BASE_DIR = Path(__file__).resolve().parent.parent
```

```
Quick-start development settings - unsuitable for production
```

```
See https://docs.djangoproject.com/en/4.1/howto/deployment/checklist/
```

```
SECURITY WARNING: keep the secret key used in production secret!
```

```
SECRET_KEY = 'django-insecure-
```

```
86jt7o12%dfcws4jf0c@jywotr2@yuo_wzcqwbtaf805t_s^jk'
```

```
SECURITY WARNING: don't run with debug turned on in production!
```

```
DEBUG = True

ALLOWED_HOSTS = []

Application definition

INSTALLED_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'parkingmanagement.apps.ParkingmanagementConfig',

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

'django.contrib.messages.middleware.MessageMiddleware',
```

```
'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT_URLCONF = 'parkingProject.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [],

'APP_DIRS': True,

'OPTIONS': {

'context_processors': [

'django.template.context_processors.debug',

'django.template.context_processors.request',

'django.contrib.auth.context_processors.auth',

'django.contrib.messages.context_processors.messages',

],

},

},

]

WSGI_APPLICATION = 'parkingProject.wsgi.application'
```

```
Database
```

```
https://docs.djangoproject.com/en/4.1/ref/settings/#databases
```

```
DATABASES = {
```

```
 'default': {
```

```
 'ENGINE': 'django.db.backends.sqlite3',
```

```
 'NAME': BASE_DIR / 'db.sqlite3',
```

```
 }
```

```
}
```

```
Password validation
```

```
https://docs.djangoproject.com/en/4.1/ref/settings/#auth-password-validators
```

```
AUTH_PASSWORD_VALIDATORS = [
```

```
{
```

```
 'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
```

```
},
```

```
{
```

```
 'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
```

```
},
```

```
{
```

```
 'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
```

```
},

{

'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',

},

]
```

# Internationalization

# <https://docs.djangoproject.com/en/4.1/topics/i18n/>

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# <https://docs.djangoproject.com/en/4.1/howto/static-files/>

STATIC\_URL = 'static/'

# Default primary key field type

# <https://docs.djangoproject.com/en/4.1/ref/settings/#default-auto-field>

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

Manage

#!/usr/bin/env python

```

"""Django's command-line utility for administrative tasks."""

import os

import sys

def main():

 """Run administrative tasks."""

 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'parkingProject.settings')

 try:

 from django.core.management import execute_from_command_line

 except ImportError as exc:

 raise ImportError(

 "Couldn't import Django. Are you sure it's installed and "

 "available on your PYTHONPATH environment variable? Did you "

 "forget to activate a virtual environment?"

) from exc

 execute_from_command_line(sys.argv)

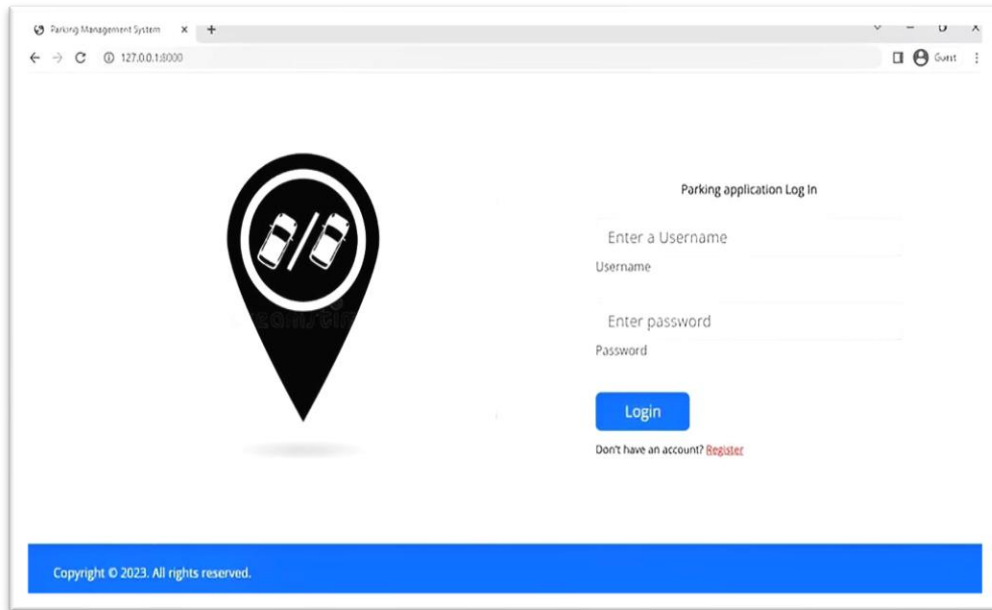
if __name__ == '__main__':

 main().

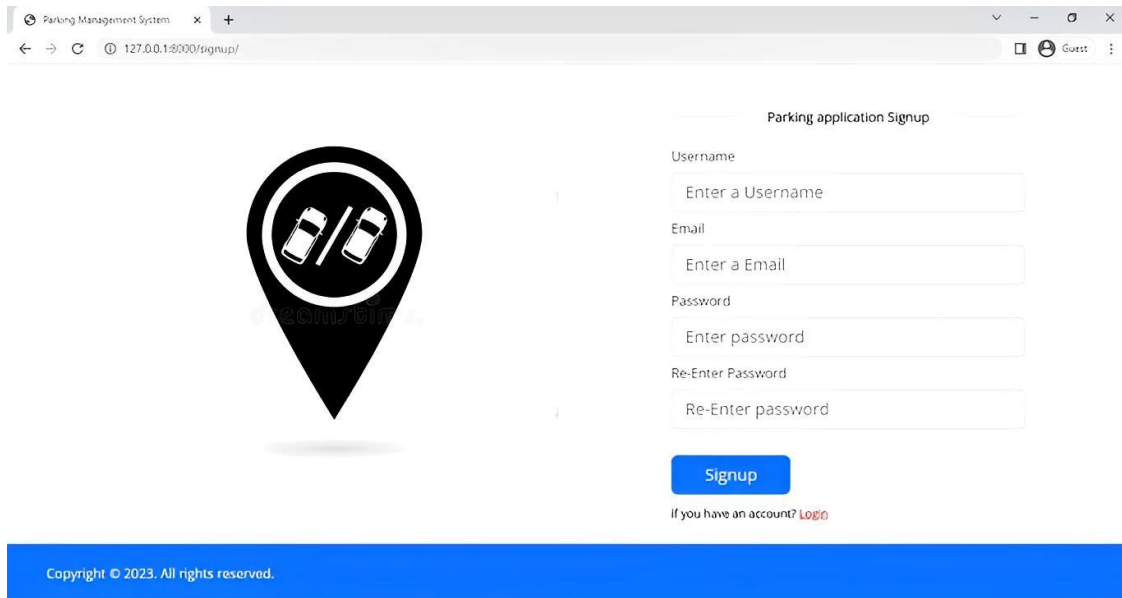
```

## 6.2 SCREENSHOTS

### Client side page



A screenshot of a web browser displaying the 'Parking Management System' login page. The browser's address bar shows '127.0.0.1:8000'. The page features a large black location pin icon with two white car silhouettes inside. To the right of the icon is a login form titled 'Parking application Log In'. The form includes two input fields: 'Enter a Username' (labeled 'Username') and 'Enter password' (labeled 'Password'). Below these fields is a blue 'Login' button. At the bottom of the form, there is a link that says 'Don't have an account? Register'. A blue footer bar at the bottom of the page contains the text 'Copyright © 2023. All rights reserved.'



A screenshot of a web browser displaying the 'Parking Management System' signup page. The browser's address bar shows '127.0.0.1:8000/signup/'. The page features the same large black location pin icon with two white car silhouettes. To the right of the icon is a signup form titled 'Parking application Signup'. The form includes four input fields: 'Enter a Username' (labeled 'Username'), 'Enter a Email' (labeled 'Email'), 'Enter password' (labeled 'Password'), and 'Re-Enter password' (labeled 'Re-Enter Password'). Below these fields is a blue 'Signup' button. At the bottom of the form, there is a link that says 'If you have an account? Login'. A blue footer bar at the bottom of the page contains the text 'Copyright © 2023. All rights reserved.'

Parking Management System x +

← → ↻ 127.0.0.1:8000/signup/

New tab Sign Up google Booked cars - 5- Tata Aft alfa-romeo code pe Admit O Sign Up Sign Up Sign X +

← ↻ 127.0.0.1:8000/slot

### Book Your Slot

Name:

Make:

Model:

License plate:

New tab Sign Up google Booked cars - 5- Tata Aft alfa-romeo code pe Admit O Sign Up Sign Up Sign X +

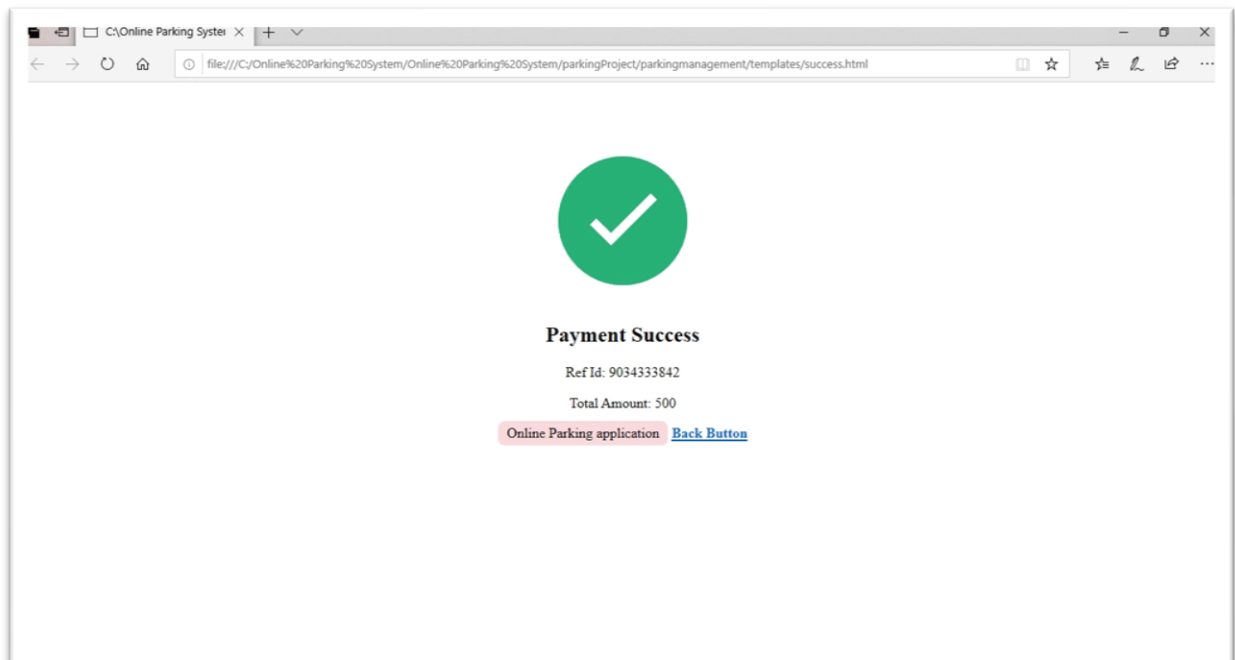
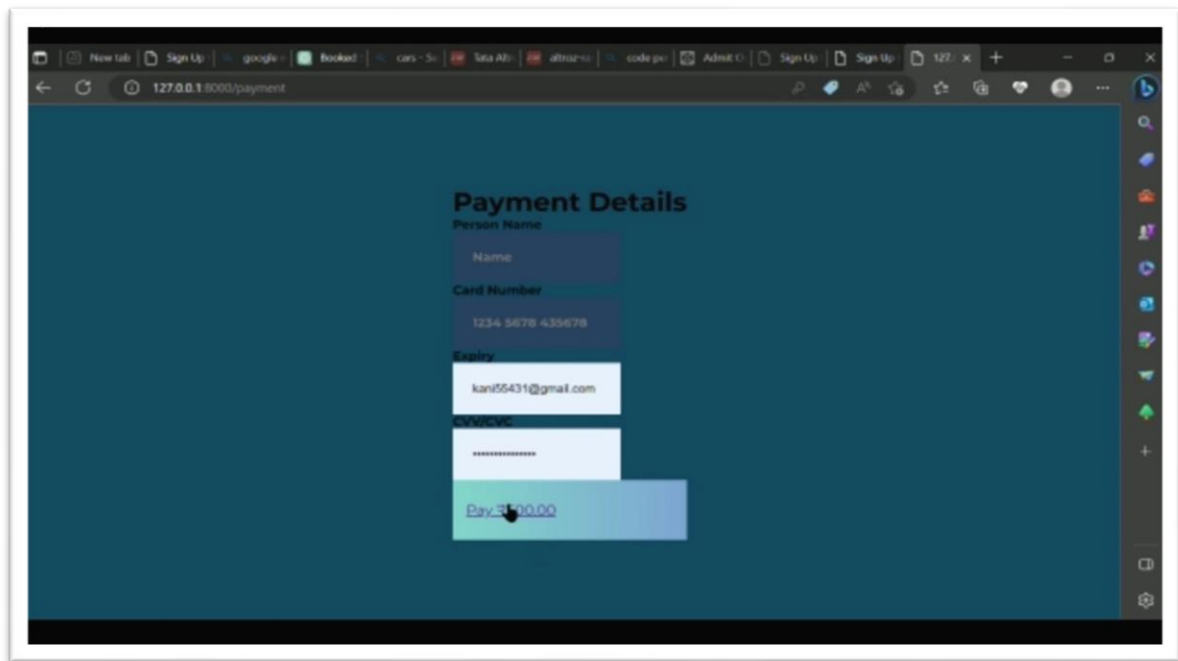
← ↻ 127.0.0.1:8000/slot

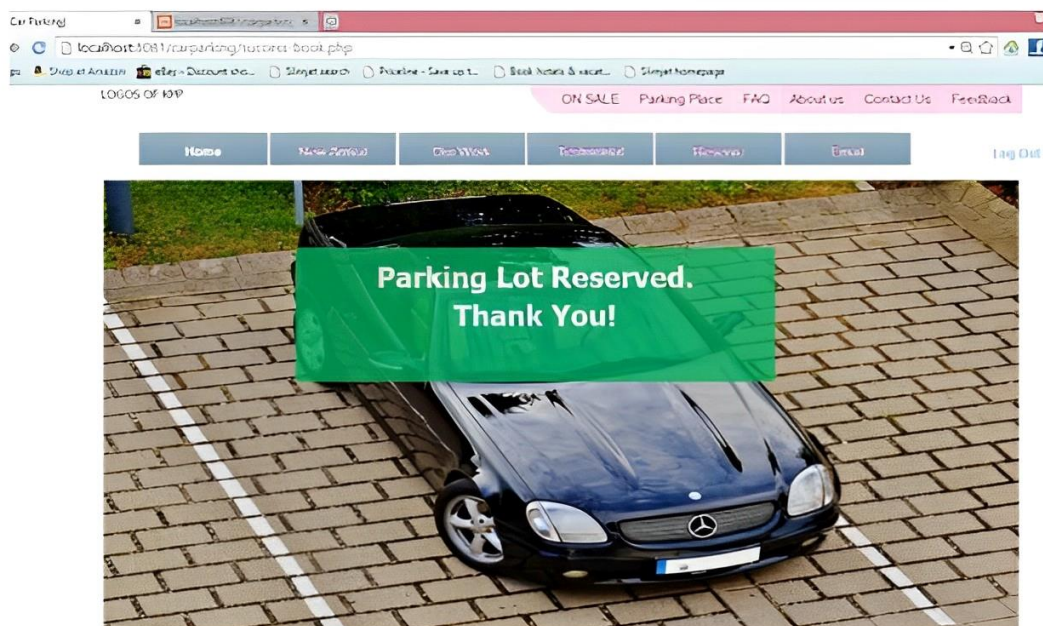
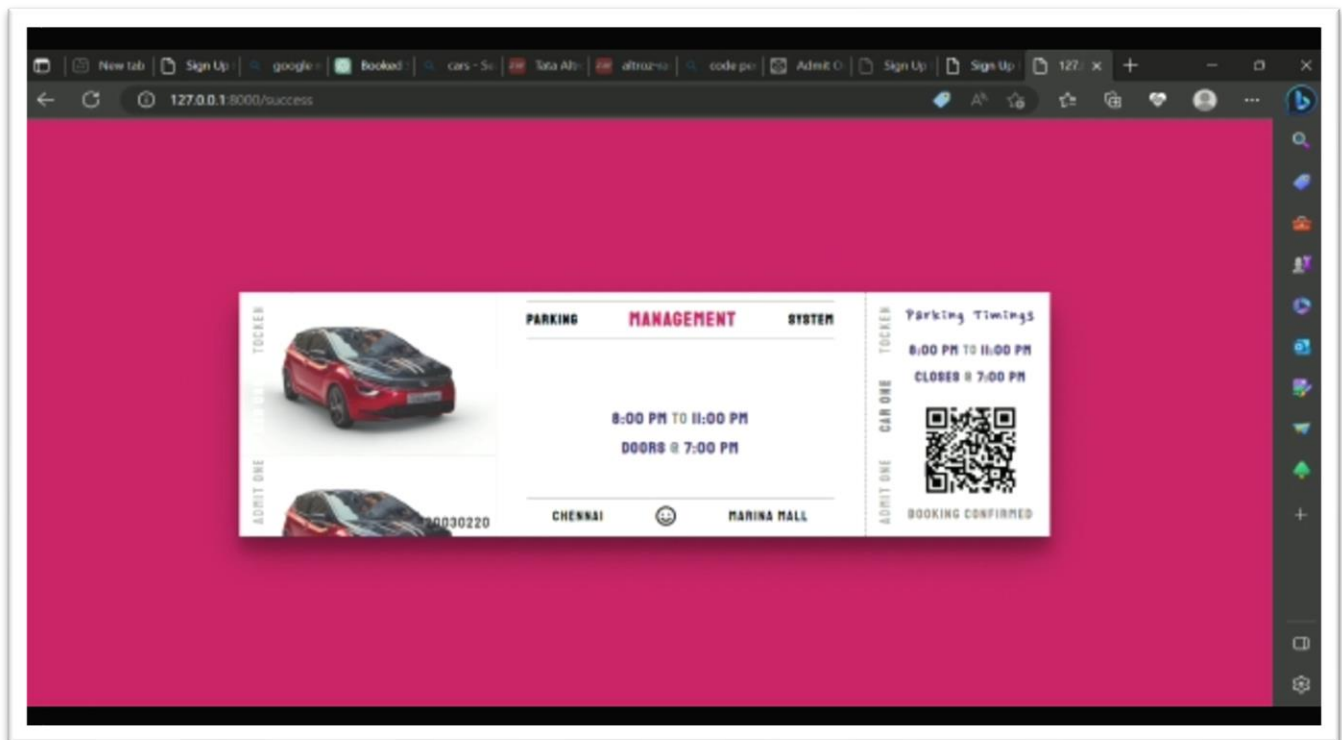
License plate:

Location:

Date:

Time:





Admin side output:



Parking application Log In

antony

Username

.....

Password

Login

Don't have an account? [Register](#)

Copyright © 2023. All rights reserved.



Parking application Signup

Username

anjok

Email

prasanth.k99y@gmail.com

Password

.....

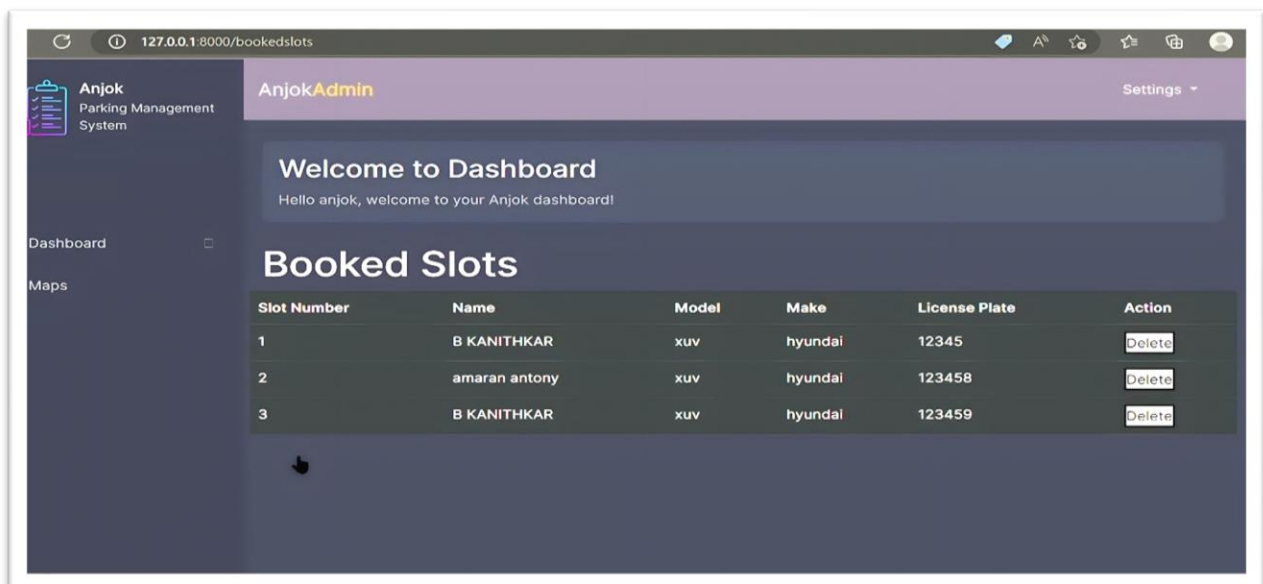
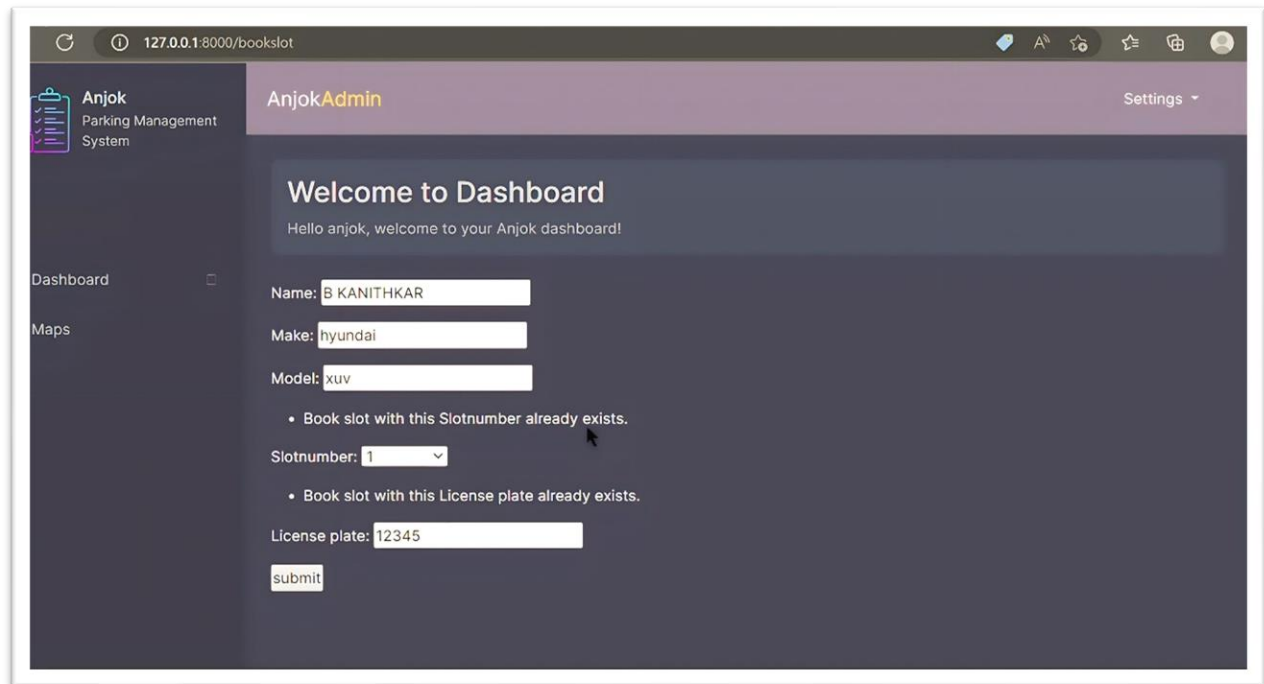
Re-Enter Password

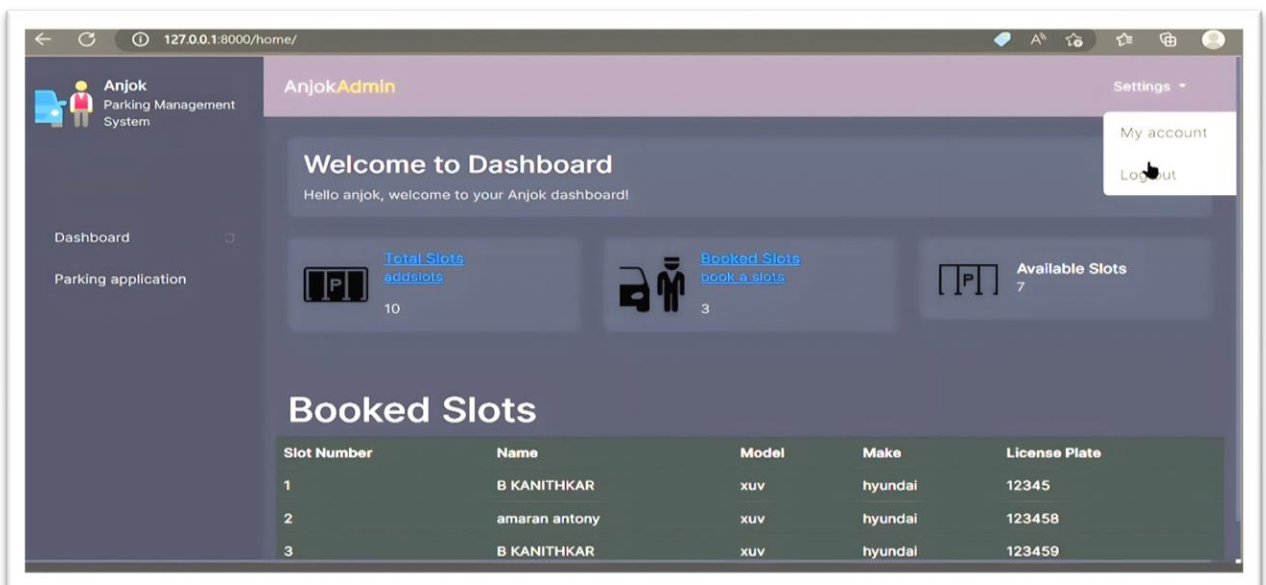
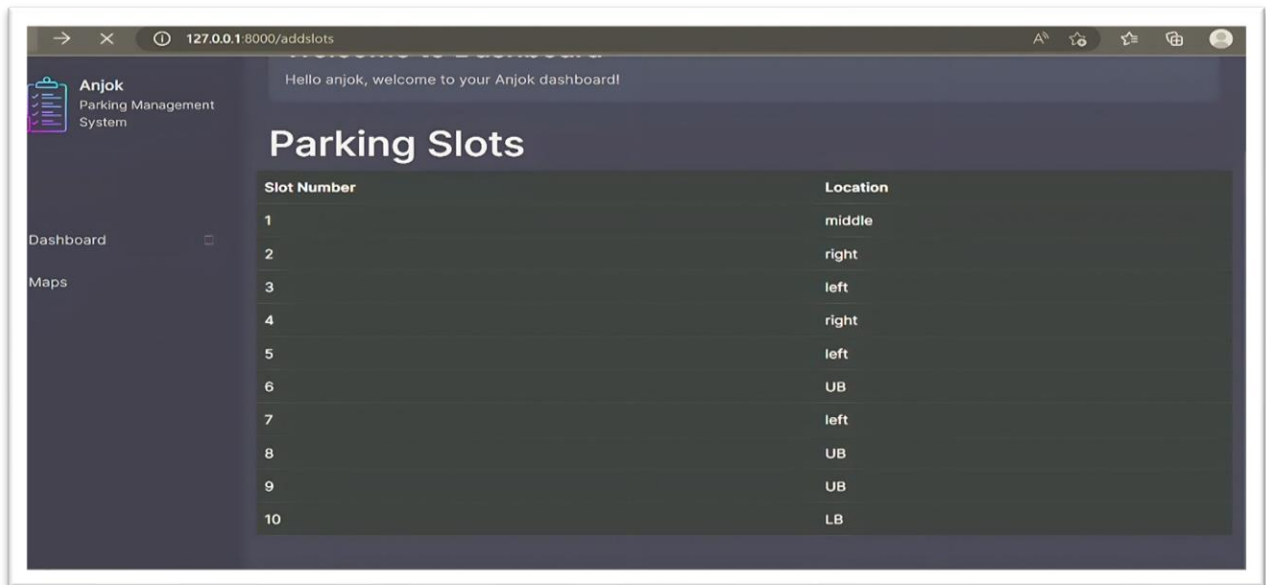
.....

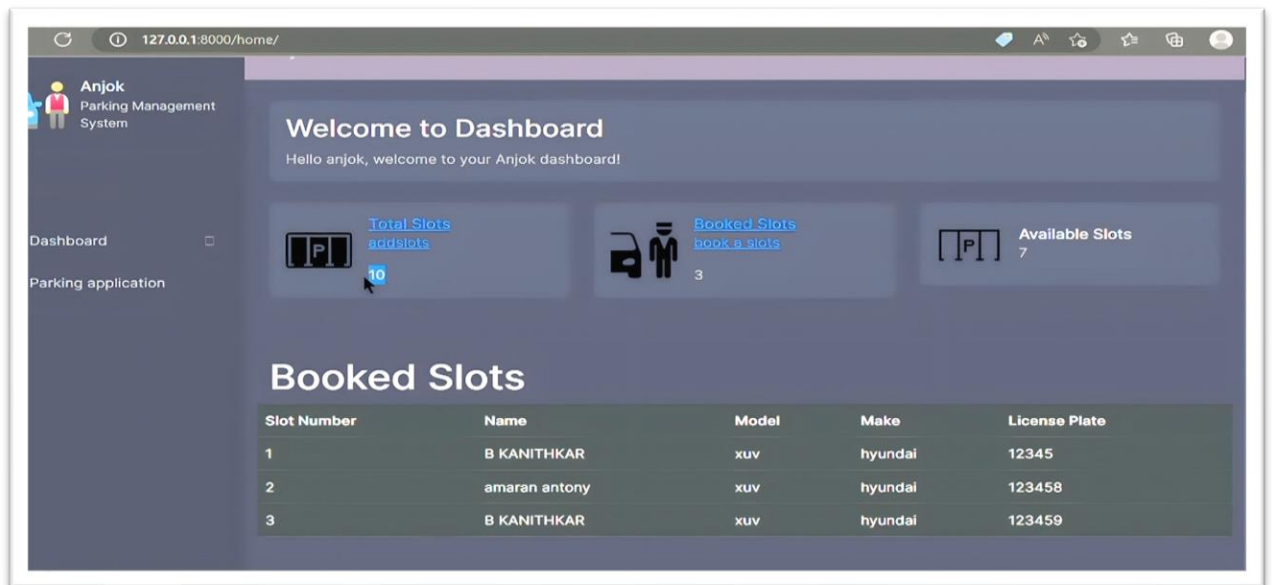
Signup

If you have an account? [Login](#)

Copyright © 2023. All rights reserved.







### 6.3 TABLE OF DESCRIPTION

Table 1:describes the parking table

Column name	Data type	Description	size	Allow null value
Parkid	Int	Primary key for the table	15	no
Park	text	Name of the parking	default	no
Plot	text	Name the parking space number	default	no
Model	text	Name the vehicle model	default	no
Num_plate	int	Indicate the vehicle number plate	9	no
Payment	float	Indicate the amount paid	default	no
Time_A	timestamp	Date and time of the client arrive in parking space	default	no
Time_D	timestamp	Date and time of the client departure from parking lot	default	no

Table 2: describes the user table

Column name	Data Type	Description	Size	Allow null value
User_id	int	Primary key for the table	20	no
Fname	text	The first name and the last name of the user	default	no
Email	text	The email of the user	default	no
Password	text	Password of the user	default	no
Phone_num	int	The phone number of the user	12	no
gender	varchar	The gender of the user	6	no

Table 3:describes the Admin table

Column name	Data Type	Description	size	Allow null value
Admin_id	int	The primary key of the admin	12	no
username	text	Username of the admin	default	no
Password	text	Password of the admin	default	no
Phone_num	int	Phone number of the admin	12	no
DOB	date	The date of birth of the admin	default	no

Table 4: describes the contact table

Column name	Data Type	Description	size	Allow null value
Contact_id	int	The primary key of the contact	12	no
Name	text	The name of client or the user who want to send a feedback	default	no
Email	text	The email of user who want to send a feedback	default	no
Purpose	text	The reason of sending a feedback	default	no
Message	text	The message which needed to be delivered	default	no

## CHAPTER 7

### REFERENCES

1. Kim, J.; Song, J. A Dual Key-Based Activation Scheme for Secure LoRaWAN. *Wirel. Commun. Mob. Comput.* 2017, 2017, 1–12.
2. Hung, M. Leading the IoT; Technical report; Gartner Inc.: Stamford, CT, USA, 2017.
3. Sinha, R.S.; Wei, Y.; Hwang, S.H. A survey on LPWA technology: LoRa and NB-IoT. *ICT Express* 2017, 3, 14–21.
4. Qian, Y.; Jiang, Y.; Chen, J.; Zhang, Y.; Song, J.; Zhou, M.; Pustisek, M. Towards decentralized IoT security enhancement: A block chain approach. *Comput. Electr. Eng.* 2018, 72, 266–273.
5. Endress, C.; Friedrich-Baasner, G.; Heim, D. From Facets to a Universal Definition—An Analysis of IoT Usage in Retail. In *Proceedings of the International Conference on Wirtschaftsinformatik*, Siegen, Germany, 24–27 February 2019.
6. Sha, K.; Yang, T.A.; Wei, W.; Davari, S. A survey of edge computing based designs for IoT security. *Digit. Commun. Netw.* 2019.
7. Lee, M. An Empirical Study of Home IoT Services in South Korea: The Moderating Effect of the Usage Experience. *Int. J. Hum. Comput. Interact.* 2019, 35, 535–547.
8. Mekki, K.; Bajic, E.; Chaxel, F.; Meyer, F. A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express* 2019, 5, 1–7.
9. Aydin, I.; Karakose, M.; Karakose, E. A navigation and reservation based smart parking platform using genetic optimization for smart cities. In *proceeding of the ICSG 2017—5<sup>th</sup> International Istanbul Smart Grids and Cities Congress and Fair*, Istanbul, Turkey, 19–21 April 2017; pp. 120–124.

10. International Telecommunication Union. World Telecommunication Development Conference (WTDC-14): Final Report; International Telecommunication Union: Geneva, Switzerland, 2014.