

SMART PARKING SYSTEM USING IOT

A Project Report Submitted by

Sujay S Shenoy (4NM17CS188)

Stalin C DSouza (4NM17CS187)

Saneeth S (4NM17CS159)

Suraj U Baliga M (4NM17CS191)

UNDER THE GUIDANCE OF

Mr. Pradeep Kanchan

Assistant Professor Gd III

*In partial fulfillment of the requirements for the
award of the Degree of*

***Bachelor of Engineering
(Computer Science & Engineering)***

From

Visvesvaraya Technological University

Department of Computer Science and Engineering

NMAM Institute of Technology, Nitte – 574110

(An Autonomous Institution affiliated to VTU Belgaum)

July 2021



NITTE
EDUCATION TRUST

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC

☎ : 08258 - 281039 - 281263, Fax: 08258 - 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

CERTIFICATE

Certified that the project work entitled

“Smart Parking System”

is a bonafide work carried out by

Sujay S Shenoy
(4NM17CS188)

Stalin C DSouza
(4NM17CS187)

Suraj U Baliga M
(4NM17CS191)

Saneeth S
(4NM17CS159)

in partial fulfilment of the requirements for the award of

Bachelor of Engineering Degree in Computer Science and Engineering

prescribed by Visvesvaraya Technological University, Belgaum

during the year 2020-2021.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of the

project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide

Mr. Pradeep Kanchan,

Asst. Professor Gd III

Signature of HOD

Signature of Principal

Semester End Viva Voce Examination

Name of the Examiners

Signature with Date

1. _____

2. _____

ACKNOWLEDGEMENT

The satisfaction that accompanies the completion of any task would be incomplete without the mention of all the people, without whom this endeavour would have been a difficult one to achieve. Their constant blessings, encouragement, guidance and suggestions have been a constant source of inspiration

First and foremost, my gratitude to my project guide **Mr. Pradeep Kanchan** for the constant guidance throughout the course of this Project Phase-1 and for the valuable suggestions.

I also take this opportunity to express a deep sense of gratitude to the project coordinators for their valuable guidance and support.

My thanks to our beloved HOD, **Dr. Jyothi Shetty**, for extending support in carrying out this project in the department and providing us with all necessary facilities.

My sincere thanks to our beloved principal, **Dr. Niranjan N Chiplunkar** for permitting us to carry out this project at our college and providing us with all needed facilities.

Finally, thanks to staff members of the Department of Computer Science and Engineering and our friends for their honest opinions and suggestions throughout the course of our project Phase-2.

Sujay S Shenoy (4nm17cs188)

Stalin Christopher Dsouza (4nm17cs187)

Saneeth S(4nm17cs159)

Suraj U Baliga M(4nm17cs191)

ABSTRACT

Efficient and smart way to automate the management of the parking system that allocates an efficient parking space using internet of things technology. The IoT provides a wireless access to the system and the user can keep a track of the availability of the parking area. With increase in the population of the vehicles in metropolitan cities, road congestion is the major problem that is being faced. The aim of this project is to resolve this issue. The user usually wastes his time and efforts in search of the availability of the free space in a specified parking area.

Internet-of-things-based technologies have advanced so much and helped public necessities. The use of IoT at a parking lot will help vehicle users to know the availability of a parking location through website. This IoT-based parking system is created by using controllers, sensors, servers and cloud. Controllers and sensors will be placed on the base of each parking slots to detect the presence of a car. Server collect the results of the sensors and store them in Cloud. The website queries with the server to get the real-time status of the parking slot (whether slot is empty or some vehicle is parked in the slot).

Table of Contents

Title page.....	i
Certificate.....	ii
Acknowledgement.....	iii
Abstract.....	iv
Table of Contents.....	v-vi
Chapter 1 Introduction.....	1-3
1.1 Overview.....	1
1.2 Problem Statement.....	1
1.3 Study Area.....	2
1.4 Objectives.....	2
1.5 Organization of Chapters.....	2-3
Chapter 2 Literature Survey.....	4-5
2.1 Existing System.....	4-5
2.2 Proposed System.....	5
Chapter 3 System Analysis and Requirements.....	6-8
3.1 Scope and Boundary.....	6
3.2 Requirement analysis.....	6-8
3.2.1 Software requirements.....	6-7
3.2.2 Hardware requirements.....	7-8
Chapter 4 Software Approach.....	9-10
4.1 NodeJS.....	9
4.2 Flutter.....	9
4.3 Android Studio.....	9-10
4.4 Visual Studio Code.....	10
Chapter 5 System Design.....	11-12
5.1 High Level Design Architecture.....	11-12

Chapter 6 System Implementation	13-15
6.1 Hardware Implementation.....	13
6.1.1 Configuring ESP-32.....	13
6.1.2 Configuring Ultrasonic Sensor.....	13
6.1.3 Configuring Servo motor.....	13
6.2 Server Implementation.....	14
6.2.1 Setting up the NodeJS Server API.....	14
6.2.2 Setting up mongoDB database.....	14
6.3 Android Application Implementation.....	15
 Chapter 7 System Testing	 16-18
7.1 Introduction.....	16
7.2 Unit Testing.....	16-17
7.3 Integration Testing.....	17-18
 Chapter 8 Results and Discussion	 19-22
8.1 Mini parking lot model.....	19
8.2 Server.....	19
8.3 Android Application.....	20-22
 Chapter 9 Conclusion and Future Works	 23
 References	 24

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Alongside with the development of information technology, the Internet of Things (IoT) become more popular at this time. IoT is a concept to connect various devices (smart house, car, mobile phone and household device) and enable communication between them. The IoT device is equipped with electronic components, sensors, actuators and network connectivity. With these components, IoT device can send data or even be controlled remotely by utilizing the internet network infrastructure.

When the concept of IoT is applied in urban life, a new concept of Smart City is introduced. Although there is no clear definition of Smart City yet, the exact goal of Smart City is to create services to facilitate access to public facilities and improve their quality. This aim can be realized by spreading the use of IoT technology on the infrastructure of public facilities in urban. One example of an important public facility in urban areas is public parking for four-wheeled vehicles.

In this part of the IoT concept can be used to improve the accessibility of parking lot for the community. People do not have to go to the location to find out how much parking space is available and where the parking lot is empty instead, they are provided with a website where they can view the vacant spaces available for parking in a parking lot.

1.2 PROBLEM STATEMENT

In order to propose and implement a smart-parking system based on Internet of things and commanded by web application using ultrasonic sensors that helps automatically find free parking space based on the parameters of performance that makes the system cost and time efficient. The parking slots which are available should be sensed and can be updated on Server and so that every user connected with the Server can identify free parking slots in a specific location.

1.3 STUDY AREA

There are several technologies that have been created for parking systems in Indonesia, and some of them have been used in shopping areas. For example, the concept of a smart parking lot by adding sensors, actuators, and microcontrollers. Users only need to put the vehicle at a certain location, then the microcontroller will instruct the actuator to drive the vehicle to an empty parking location. This concept is very interesting, but it needs a considerable cost because it takes a total physical renovation if anyone wants to apply it on an existing public parking lot. There is also a concept of another smart parking lot using sensors, microcontrollers, and Light-Emitting Diode (LED) displays. The sensor will read the amount of vacant parking space and send the data to the microcontroller. Then the microcontroller will display the amount of vacant parking spot and display it to the LED display.

1.4 OBJECTIVE

The aim of this project is to make it cost effective and user friendly. Smart parking system helps the user to sustain the data with 90% of accuracy. Smart parking system provides a comprehensive parking solution for the user as well as admin of the parking area.

1.5 ORGANIZATION OF THE CHAPTERS

Chapter I: Introduces to the main idea of the project. It gives a brief knowledge about the aim and methodology of the same.

Chapter II: It includes literature survey of related works.

Chapter III: Discusses the system requirements that are needed for the project. These include hardware and software requirements.

Chapter IV: Includes the software approach.

Chapter V: Includes the High Level Design architecture and Low Level Design Architecture.

Chapter VI: Includes the implementation part of the Project.

Chapter VII: Includes the testing part.

Chapter VIII: Includes the Results with Screenshots of the project.

Chapter IX: Includes the Conclusions and Future Work.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Pp . No	Paper Name	Author Name	Year	Advantages	Disadvantages
1	Smart parking reservation system using short message services (SMS).	1.Noor HazrinHanyMohamadHanif 2.Mohd Hafiz Badiozaman 3.Hanita Daud	2009	1. Enhanced security due to password requirement. 2. System can be used and applied anywhere due to ease of usage.	1.Cost of implementation is high. 2.GSM feature creates bottlenecks 3.The microcontroller will have to take a lot of load which can crash the system.
2	ZigBee and GSM based secure vehicle parking management and reservation system.	1.Ashwin Sayeeraman 2.P.S. Ramesh	2012	1.Parking lot vacancy module uses ZigBee along with PIC. 2.Security Feature: The exit password must be entered else the user is not allowed to get out of the parking bay as the barrier gate will not get open until correct exit password is entered	1. The GSM and SMS module makes the system expensive. 2. The SMS contains entry/exit password to the parking lot may not be received due to network congestion.
3	Smart Parking Service based on Wireless Sensor Networks.	1.Jihoon Yang 2.Jorge Portilla 3.Teresa Riesgo	2012	1. Use of android application provides ease of usage and better interface. 2. GPS helps in max coverage of available area, displaying various	1. Reservation feature is not available for the user. 2. Multilevel parking inside an infrastructure is not available.

				options for parking.	
4	An Intelligent Parking Guidance and Information System by using image processing technique.	1.P. DharmaReddy 2.A. RajeshwarRao 3.Dr. Syed Musthak Ahmed	2013	1. By using image processing technique it identifies car only but if any object other than car is at parking slot it doesn't considered that slot is booked. 2. Shows real time information.	1.High cost of implementation 2. User will have to inquire for every slot available. 3. GSM system creates bottlenecks.

2.2 PROPOSED SYSTEM

In recent times the concept of smart cities has gained great popularity. Thanks to the evolution of Internet of things the idea of smart city now seems to be achievable. Consistent efforts are being made in the field of IoT in order to maximize the productivity and reliability of urban infrastructure. Problems such as, traffic congestion, limited car parking facilities and road safety are being addressed by IoT. The proposed Smart Parking system consists of an on-site deployment of an IoT module that is used to monitor and signalize the state of availability of each single parking space. A mobile application and website is also provided that allows an end user to check the availability of parking space. This proposed can be used in city malls, offices, public parking places etc.

CHAPTER 3

SYSTEM ANALYSIS AND REQUIREMENTS

3.1 SCOPE AND BOUNDARY

Requirements are during early stages of a system development as a specification of what should be implemented or as a constraint of some kind of on the system. They may be a user level facility description, a detailed specification of expected system behaviour, a general system property, a specific constraint on the system, and information on how to carry out some computation or a constraint on the development of the system. The end product of the requirement analysis phase is a requirement specification. The requirement specification is a reconstruction of the result of this analysis phase. Its purpose is to communicate this result to others. System requirements are more detailed descriptions of the user requirements. They may serve as the basis for a contract to the implementation of the system and should therefore be a complete and consistent specification of the whole system. In principle, the system requirements should state what the system should do and not how it should be implemented. However, at the level of detail required to specify the system completely, it is virtually impossible to exclude all design information.

3.2 REQUIREMENT ANALYSIS

➤ 3.2.1 SOFTWARE REQUIREMENTS

➤ NodeJS Server

A Server application built using NodeJS to gather data from sensors, store them in the database and also send the real-time data to web application.

➤ Android Application

An android application that can fetch data from server and display the real-time data to the users.

➤ **MongoDB**

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL).

➤ **3.2.2 HARDWARE REQUIREMENTS**

➤ **Ultrasonic Sensors (HC-SR04)**

The HC-SR04 uses non-contact ultrasound sonar to measure the distance to an object, and consists of two ultrasonic transmitters (basically speakers), a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce off any nearby solid objects, and the receiver listens for any return echo. That echo is then processed by the control circuit to calculate the time difference between the signal being transmitted and received.

➤ **ESP-32**

ESP32 is a series of low-cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process. It is a successor to the ESP8266 microcontroller.

➤ **Servo motor**

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration.[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

Servomotors are not a specific class of motor, although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system. Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.

CHAPTER 4

SOFTWARE APPROACH

4.1 NodeJS

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts. Here the NodeJS is used to establish connection between hardware (Ultrasonic sensors and Wi-Fi module) and software.

4.2 FLUTTER

Flutter is an open-source UI software development kit created by Google. It is used to develop applications for Android, iOS, Linux, Mac, Windows, Google Fuchsia, and the web from a single codebase. Here it is used to build a mobile application for the user.

4.3 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020.

Android Studio supports all the same programming languages of IntelliJ e.g. Dart, Java, C++, and more with extensions, such as Go and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version.". At least some new language features up to Java 12 are usable in Android.

4.4 Visual Studio Code

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE. Features includes debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

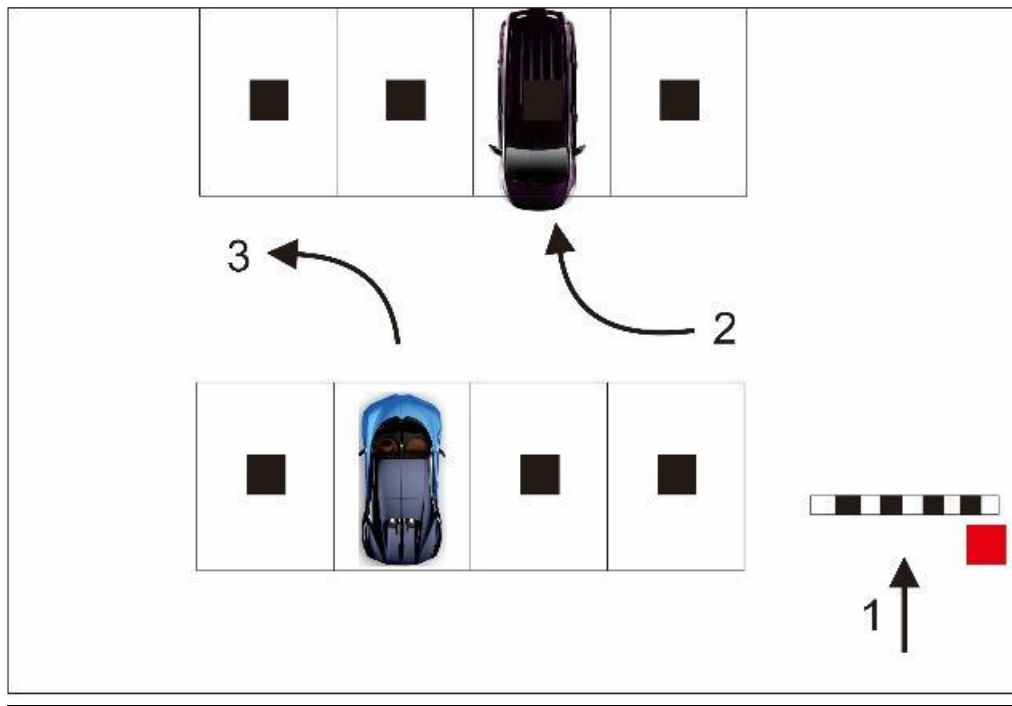
Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software.

Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language

CHAPTER 5

SYSTEM DESIGN

5.1 HIGH LEVEL DESIGN ARCHITECTURE

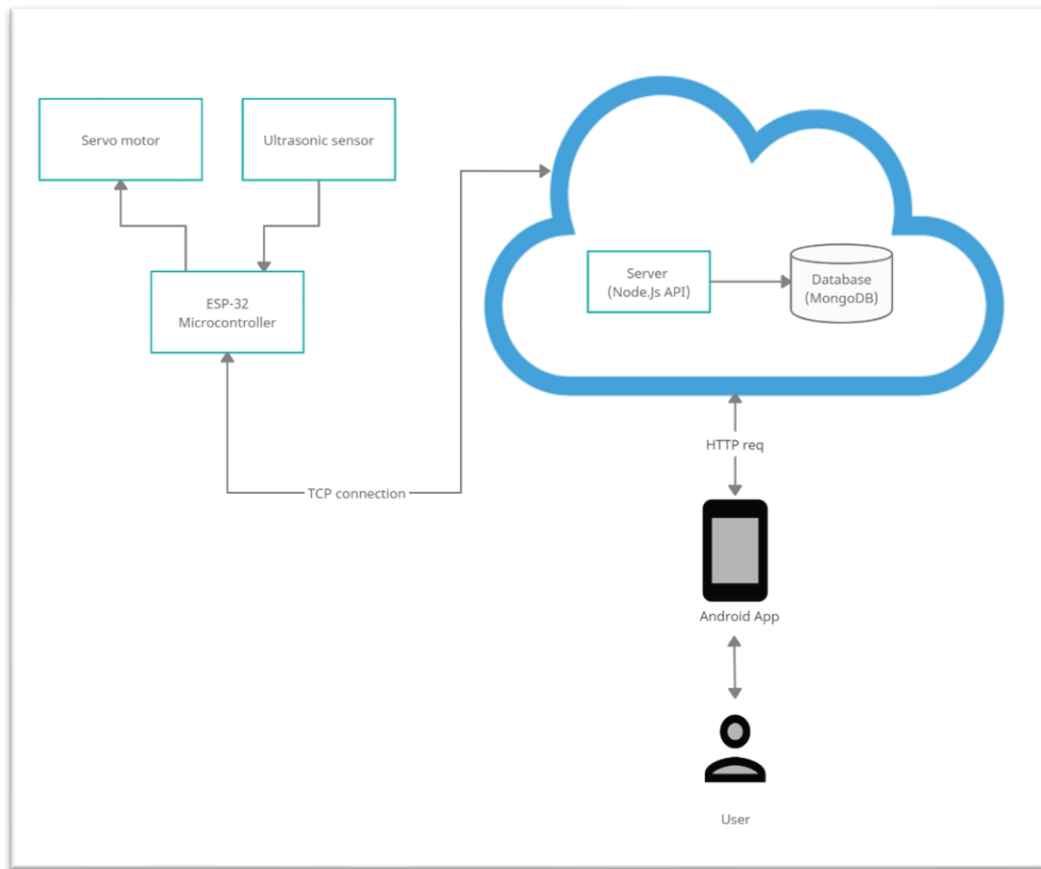


The developed parking system is a system that is able to calculate and detect the presence of cars in the parking area. The data obtained in the parking area will be sent to the cloud database. The data will be used by web and android application to inform the users which parking space they should go.

Every parking spot in the parking lot is fitted with ultrasonic sensors. The microcontroller continuously monitors each and every sensor. Whenever the user views the visualization of the parking lot in the android app or website, the application sends a http request to the backend API. Whenever a http request is received from the application, the API communicates with the microcontroller through the TCP channel. On receiving the request from the API, the microcontroller reads all the sensors and sends the data back to the API through the TCP channel. The TCP connection is maintained throughout the working of the system.

The application continuously sends http requests to update its data, so that the user can view

the status of parking spots in real time.



Users can also reserve a spot using the android application. The reserved spot will be marked unavailable to other users. The gates in the parking spot reserved will be dropped on reservation. Users can open the gate by using the secret code provided by the android application. Reservations are only valid for 15 minutes; gates will be automatically raised after the timeout revoking the reservation.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Hardware Implementation

6.1.1 Configuring ESP-32

ESP-32 is used to control and coordinate all the components used in the project.

- ESP-32 is programmed by connecting to the laptop serial port
- The C++ Program used to control other hardware units is written into the ESP-32 memory using Arduino IDE
- A TCP connection is established between the ESP-32 and the server using network libraries
- This TCP channel is used for receiving commands from the server and also send the data to the server
- The ESP-32 is powered using 3.3V DC.

6.1.2 Configuring Ultrasonic Sensors

Ultrasonic Sensors are used to monitor the status of the parking spots.

- The Ultrasonic Sensor is powered using 5V DC
- The trig pin (sends ultrasonic signal) and echo pin (receives ultrasonic signal) are connected to their respective data pins on ESP-32
- The data from the sensor is sent to ESP-32 using these data pins
- The distance is calculated by measuring time required by the ultrasonic signal to bounce back from an object.

6.1.3 Configuring Servo motors

Servo motors are used to raise/drop barricades in the parking area.

- The Servo motors are powered using 5V DC
- The data pin of Servo motor is connected to data pin of ESP-32
- Upon receiving the commands from the server, the ESP-32 rotates the motors, hence raising/dropping the barricades attached to the motor as required.

6.2 Server Implementation

6.2.1 Setting up the NodeJS Server API

This API acts as a mediator between the hardware and android application.

- A http listener is set up at the port 4000 where other applications can access this API
- A TCP channel is established between the server and the microcontroller at port 8000
- On receiving http requests from the app on port 4000, the NodeJS API communicates with the microcontroller on port 8000
- The command is sent to the microcontroller over the TCP channel requesting data. The microcontroller after reading the command interacts with other components connected to it to gather data. The data is sent to the API using the same TCP channel
- The API on receiving the data from the hardware processes it and converts it to JSON format and send the processed data to the app
- This API uses token-based authentication where the valid users are provided with the token to access its services
- User details and reservation info are stored in mongoDB database
- NodeJS uses mongoose library to connect to the database.

6.2.2 Setting up mongoDB database

A NoSQL database used to store and manage user information required by the application.

- Three collections namely users, reservations and orders are used to manage user information, parking spot reservation status and all reservations made by the users respectively
- The database can be accessed by the API using the connection string.

6.3 Android Application Implementation

An Android application is developed for the users to get real time information from the parking lot.

- The UI is designed using Flutter
- After the user enters the credentials, they are sent to the API for validation. The API sends token if the credentials match
- The token sent by the API is unique to each user and is stored in local storage of the device for future uses
- A parking lot layout is provided by the application where the users can view the parking lot in real time which is dynamically updated by continuously requesting data from the API
- The parked spots, vacant spots and reserved spots are marked in red, green and yellow respectively
- The users can click on vacant parking spots to reserve it
- The users have to pay a small fee for reserving the parking spot. Upon successful payment their request is sent to the API for dropping the gates
- The API returns a 4-digit unique code which can be used to raise the barricades
- User can also view all the previous reservations. All the previous reservations are fetched from database upon request to the API.

CHAPTER 7

SYSTEM TESTING

7.1 Introduction

Software testing is a process used to identify the correctness, completeness and quality of the developed software. Testing is the process of questioning a product in order to evaluate it, where the questions are things the tester tries to do with the product and the product answers with its behavior in reaction to probing of the tester.

Testing phase is performed after coding to detect all the errors and provide quality assurance and ensure reliability of the software. Testing is vital to the success of the system. During testing, the software to be tested is executed with a set of test cases, and the behavior of the system for the test cases is evaluated to determine if the system is performing as expected. Clearly the success of testing in revealing errors depends critically on the test cases.

7.2 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

The benefits of Unit Testing are:

- Unit testing increases confidence in changing/ maintaining code. If good unit tests are written and if they are run every time any code is changed, we will be able to promptly catch any defects introduced due to the change. Also, if codes are already made less interdependent to make unit testing possible, the unintended impact of changes to any code. Codes are more reusable. In order to make unit testing possible, codes need to be modular. This means

that codes are easier to reuse.

- Development is faster. If you do not have unit testing in place, you write your code and perform that fuzzy 'developer test' (You set some breakpoints, fire up the GUI, provide a few inputs that hopefully hit your code and hope that you are all set.) But, if you have unit testing in place, you write the test, write the code and run the test. Writing tests takes time but the time is compensated by the less amount of time it takes to run the tests; You need not fire up the GUI and provide all those inputs. And, of course, unit tests are more reliable than 'developer tests'. Development is faster in the long run too. The effort required to find and fix defects found during unit testing is very less in comparison to the effort required to fix defects found during system testing or acceptance testing.
- The cost of fixing a defect detected during unit testing is lesser in comparison to that of defects detected at higher levels. Compare the cost (time, effort, destruction, humiliation) of a defect detected during acceptance testing or when the software is live.
- Debugging is easy. When a test fails, only the latest changes need to be debugged. With testing at higher levels, changes made over the span of several days/weeks/months need to be scanned.
- Codes are more reliable. I think there is no need to explain this to a same person.

7.3 Integration Testing

Integration Testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

Definitions of Integration Testing are:

- **Integration Testing:**
Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.
- **Component Integration Testing:**

Testing performed to expose defects in the interfaces and interaction between integrated components.

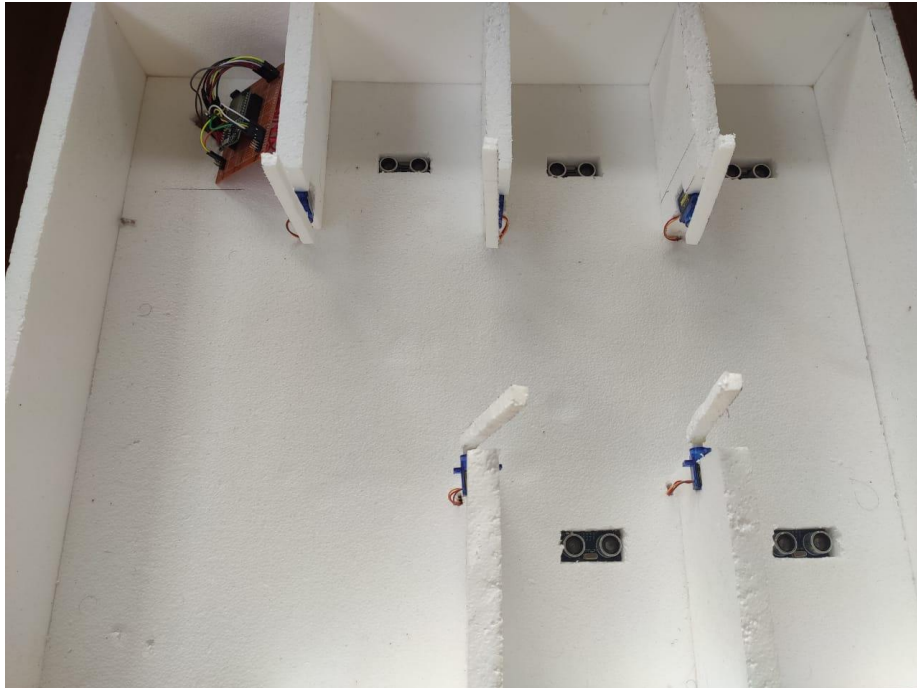
➤ System Integration Testing:

Testing the integration of systems and packages; testing

CHAPTER 8

RESULTS AND DISCUSSION

8.1 Mini parking lot model



A minified version of an parking lot that helps to demonstrate the use of smart parking system in real world applications.

8.2 Server

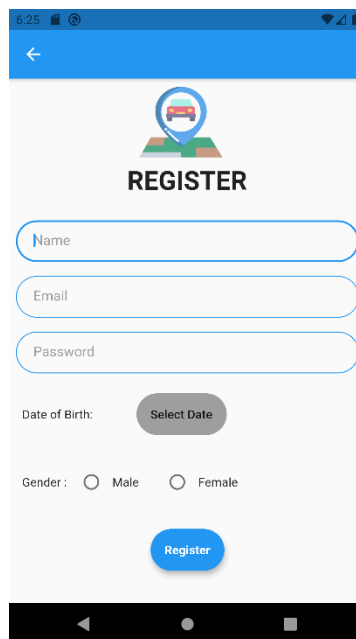
```
TCP Server Started on port 8000

server Listening on port 4000 +0ms
database Database Connected +25ms
server true +42s
POST /login 200 64.583 ms - 54
GET /me? 200 9.812 ms - 130
GET /myorders? 200 12.783 ms - 4305
GET /myorders? 200 10.574 ms - 4305
```

Nodejs server API to serve all the requests from the android application.


8.3 Android Application

Registration



6:25

←



REGISTER

Name

Email

Password

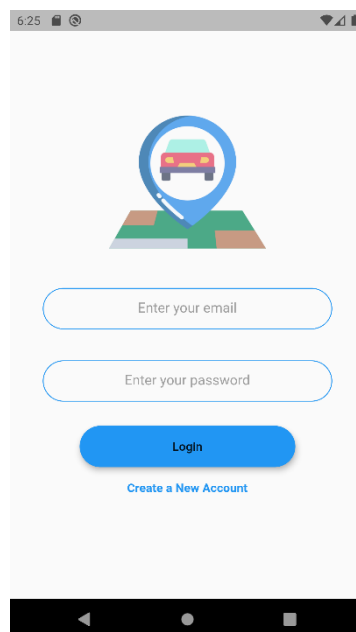
Date of Birth: [Select Date](#)

Gender: ☐ Male ☐ Female


[Register](#)

Users can register themselves for using the smart parking system using registration page.

Login



6:25



Enter your email

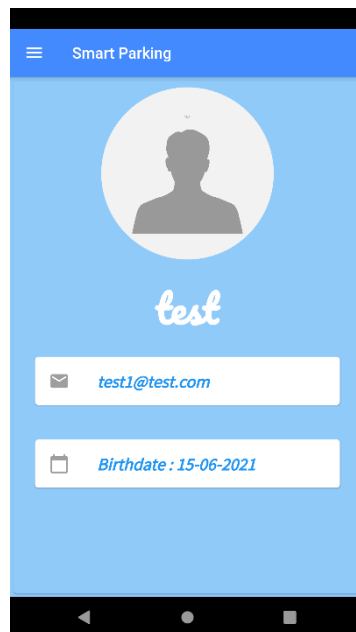
Enter your password

[Login](#)

[Create a New Account](#)

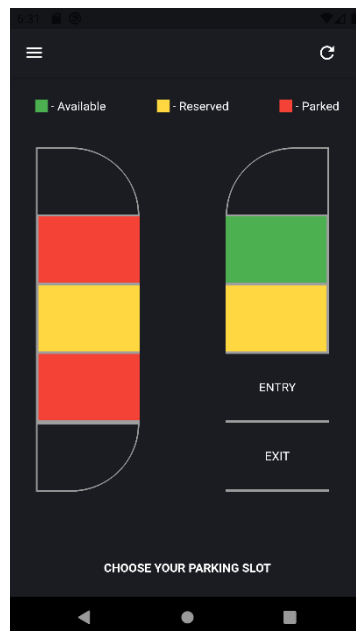
Users can log into the system using credentials provided by them during registration

Profile



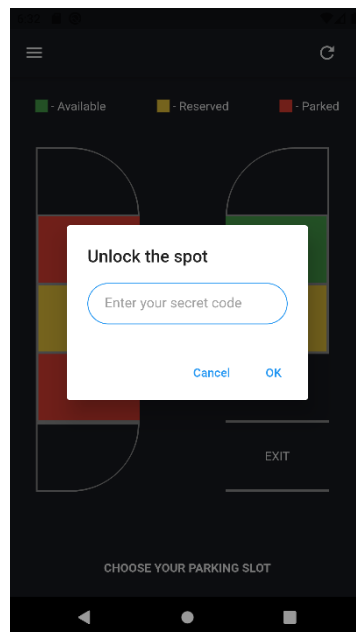
User profile page containing basic user information.

Parking Lot visualization page



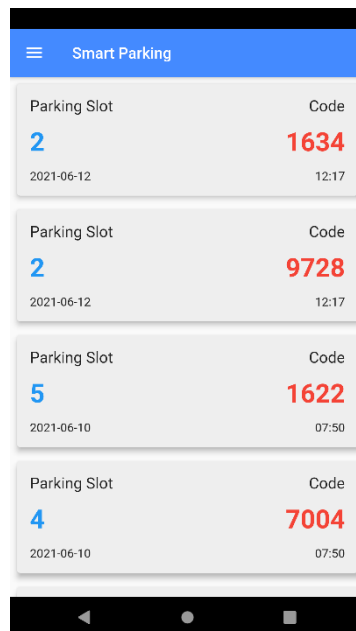
UI to view the parking lot in real time

Unlocking gates



Users can unlock gates using 4 digit secret code provided to them during reservation.

All Orders



In this page users can view all the reservations made by them.

CHAPTER 9

CONCLUSION AND FUTUREWORK

9.1 Conclusion

The concept of Smart Cities has always been a dream for humanity. Since the past couple of years large advancements have been made in making smart cities a reality. The growth of Internet of Things and Cloud technologies have given rise to new possibilities in terms of smart cities. Smart parking facilities and traffic management systems have always been at the core of constructing smart cities. In this paper, we address the issue of parking and present an IoT based Cloud integrated smart parking system. The system that we propose provides real time information regarding availability of parking slots in a parking area. Users from remote locations could book a parking slot for them by the use of our mobile application. The efforts made in this paper are indented to improve the parking facilities of a city and thereby aiming to enhance the quality of life of its people.

9.2 Future Work

- A website for better management of users and parking spots can be implemented.
- Implementation of an entry gate that only opens where there are vacant parking spots available.
- Use of distributed network of microcontrollers to better manage components in the parking lot.
- Image processing to record vehicle number plates and also capture driver's face.

REFERENCES

1. Abhirup Khanna, R. A. (2016). IoT based Smart Parking System. International Conference on Internet of Things and Applications (IOTA) (p. 5). Pune: IEEE.
2. Deng, D. (2015). A Cloud-Based Smart-Parking System Based on Internet-of-Things Technologies. IEEE, 11.
3. O. Orrie, B. S. (2015). A Wireless Smart Parking System. IECON (p.5). Yokohama: IEEE.
4. Khaoula Hassoune, W. D. (2016). Smart parking Systems: A Survey. IEEE, 6.
5. Wael Alsafery, B. A. (2018). Smart Car Parking System Solution for the Internet of Things in Smart Cities. IEEE, 5.
6. Rachapol Lookmuang, K. N. (2018). Smart Parking Using IoT Technology. IEEE, 6.
7. Mohit Patil, R. S. (2014). Smart Parking System Based on Reservation. International Journal of Scientific Engineering and Research (IJSER), 6.
8. Vishwanath Y, A. D. (2016). Survey paper on Smart Parking System based on Internet of Things. International Journal of Recent Trends in Engineering & Research (IJRTER), 5.
9. Dr.V. Kepuska, H. A. (2016). Smart Car Parking System. International Journal of Science and Technology, 7.
10. J. Cynthia, C. B. (2018). IOT based Smart Parking Management System. International Journal of Recent Technology and Engineering (IJRTE), 6.
11. Ahteshamul huq osmani, A. G. (2016). Research paper on Smart City Parking System. IJARIE, 3.
12. Asghar Ali Shah, G. M. (2019). Video Stitching with Localized 360 Model for Intelligent Car Parking Monitoring and Assistance System. IJCSNS International Journal of Computer Science and Network Security, 6.
13. Bachhav, J. D. (2017). Smart Car Parking System. International Research Journal of Engineering and Technology (IRJET), 3.
14. R, M. B. (2015). Automatic Smart Parking System using Internet of Things (IOT). International Journal of Scientific and Research Publications, 4.
15. Sadhukhan, P. (2017). An IoT-based E-Parking System for Smart Cities. Research gate , 6.
16. T. Bhanusri, K. R. (2016). Advanced Car Parking System with GSM Supported Slot Messenger. IOSR Journal of Electronics and Communication Engineering, 6.