**Smart Parking System**

A PROJECT REPORT

*Submitted by*

Yash Jain 14BCE1138

Gauri Dhawan 14BCE1248

Divyang Duhan 14BCE1061

*in partial fulfillment for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE ENGINEERING**



MARCH 2018



**School of Computing Sciences and Engineering**

**DECLARATION**

We hereby declare that the project entitled “Smart Parking System” submitted by us to the School of Computing Science and Engineering, Vellore Institute of Technology, Chennai Campus, Chennai 600127 in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a record of bonafide work carried out by us under the supervision of Dr. G. Bharadwaja Kumar. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

Chennai: Signature of the Candidate

Date:

Yash Jain (14BCE1138)

Gauri Dhawan (14BCE1248)

Divyang Duhan(14BCE1061)



**School of Computing Sciences and Engineering**

**CERTIFICATE**

The project report entitled “Smart Parking System” is prepared and submitted by Yash Jain (14BCE1138), Gauri Dhawan (14BCE1248) and Divyang Duhan (14BCE1061). It has been found satisfactory in terms of scope, quality and presentation as partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering in Vellore Institute of Technology, Chennai Campus, Chennai, India.

Name

**Project Co-Ordinator (B. Tech CSE)**

Date:

Examiner(s) Signatures: 1. 2.

**Acknowledgement**

We would like to express our sincere gratitude to all those who have helped us with their invaluable suggestions during the course of the project. We express our profound gratitude to our faculty guide, Dr. G. Bharadwaja Kumar, for guiding us throughout the project in the best way possible and encouraging us to be on the right path.

We would like to thank our Program Chair, Dr. B. Rajesh Kanna, The Dean of the Department, SCSE, Dr. Vaidehi Vijayakumar and The Associate Dean of the Department, Dr. V. Vijayakumar for their constant support and for being a continuous encouraging force all along the project.

We are grateful for their cooperation during the period of the project and providing us with all the help from within the university during our project. We would also take this opportunity to thank and show our gratitude to our Project Coordinator, Dr. B V A N S S Prabhakar Rao, whose timely mails, and proper communication, facilitated smooth operation and completion of project in stipulated time.

**Table of Contents**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr No** | **Title** | | | **Page No** |
|  | Title Page | | | I |
|  | Declaration | | |  |
|  | Certificate | | |  |
|  | Acknowledgement | | |  |
|  | Table of Contents | | |  |
|  | List of Abbreviations | | |  |
|  | Abstract | | |  |
| 1 | Introduction | | |  |
|  | 1.1 | Objective | |  |
|  | 1.2 | Background | |  |
|  | 1.3 | Motivation | |  |
| 2 | Project Description & Goals | | |  |
|  | 2.1 | Description | |  |
|  | 2.2 | Goals | |  |
| 3 | Technical Specifications | | |  |
|  | 3.1 | Hardware Specifications | |  |
|  | 3.2 | Software Specifications | |  |
| 4 | Engineering Design | | |  |
|  | 4.1 | Design Approach | |  |
|  |  | 4.1.1 | Hardware Design |  |
|  |  | 4.1.2 | Software Design |  |
|  |  | 4.1.3 | Data Interference |  |
|  | 4.2 | Code and Standards | |  |
|  |  | 4.2.1 | Object Oriented |  |
|  |  | 4.2.2 | Camel Casing |  |
|  | 4.3 | Constraints and Alternatives | |  |
|  |  | 4.3.1 | Design Constraints |  |
|  |  | 4.3.2 | Component Alternatives |  |
| 5 | Schedules, Task & Milestones | | |  |
|  | 5.1 | Task Scheme | |  |
|  | 5.2 | Gantt Chart | |  |
| 6 | Project Demonstration | | |  |
|  | 6.1 | Hardware Device | |  |
|  | 6.2 | Android App | |  |
|  | 6.3 | Firebase API | |  |
| 7 | Cost Analysis | | |  |
| 8 | Conclusion | | |  |
| 9 | References | | |  |
| 10 | Appendix A: Source Codes | | |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| S.NO. | Table Name | Page No. |
|  |  |  |
|  |  |  |

**List of Figures**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

**ABSTRACT**

With an increase in the world’s population, there has been a steep increase in the amount of traffic, pollution, more fuel consumption which produced every day. With a majority of population growth occurring in developing the country like India, it is imperative to keep a check on the fuel consumed, pollution created by vehicles, increase in traffic in order to maintain basic levels of traffic control and standard-of-living. The municipal corporation of any district should have a clear idea of the pollution level of the city, so that they can effectively take measures for reducing traffic and fuel. A semi-autonomous parking system is proposed to enable a clear, transparent and effective management of traffic and fuel for both, the municipality as well as the public. Our system interacts with the user and books a parking slot which results in reducing of time and fuel. It sends and retrieves detailed report to and from the cloud at certain intervals which helps the user to book a parking slot in well advance of the journey. The information across all the parking areas is accumulated and is visible to the user app. The user can pre-book a parking slot which generates a unique OTP and QR code so as to park the vehicle when he arrives at the parking slot. There would be two types of user’s gold members and normal members, the gold members would be having extra benefits for less waiting time in queues and other services such as car wash and valet parking. Further, the system helps the corporation to map out the best effective ways to collect all the increase the parking slots in more traffic areas.

**1. INTRODUCTION**

**1.1 Objective**

The primary objective of this system is to create an awareness among the general public to reduce fuel wastage and reduce traffic especially metro cities by using the latest technologies in real life applications. The system captures important data such as the nearby empty parking areas, empty parking slots in parking areas, pre-booking a parking area, car ash services etc. The system can further consolidate all the information and used for further analytics on multiple factors. Along with pre-booking feature, it also generates QR codes and OTP to ensure vehicle safety. The whole system is connected with Google Cloud API which helps us to deliver the real-time data.

**1.2 Background**

The concept of smart homes and smart cities has been around for almost a decade. With the use of a handheld device a user can essentially control each and every part of his home, and much more. Further, internet of things has revolutionized the entire world. Different gadgets and objects autonomously interact with each other to provide a seamless experience to the user.

Based on these concepts, various organizations focused on ways to improve the quality of life in developing and under-developed countries. The Royal Society of Open Sciences identified the need to have better waste management protocols, especially in the SAARC nations. The prime area of attention was human hygiene, and a lot of research was done in developing methods to alleviate the pollution conditions of people dwelling in populated areas.

A few organizations, such as IBM, proposed SaaS platforms for governments to use. These platforms could effectively help in analyzing data collected by the user and help chart out ways to mitigate waste generated. Other research work done primarily focused on tackling the problem by deploying hardware-based tools throughout any area. ‘Fybr’ being one such proposal, suggested to use sensors used by SFPark, a federally funded parking management program in San Francisco run by local transit authority. This allowed for better in-depth analysis of pollution-levels throughout the city and allowed the government to segregate zones based on number of cars parked in a parking slot.

**1.2 Motivation**

Parking management plays a very crucial part in the well-being of the society. Poor control over traffic and pollution management directly leads to pathetic hygiene conditions and a low standard-of-living. Pollution management is especially important in developing countries such as India or Pakistan, which simultaneously need to tackle the problem of rapidly growing population. Countless efforts have been taken by developed countries in educating the people about air pollution and causes, but in our country where a huge population is uneducated, people can’t even distinguish between pollution injected and fresh air to breath. It has been observed that if the air pollution is not segregated at source it becomes almost impossible to segregate it at a later stage, eventually ending up in worse air conditions. Hence, we lose a chance of using the latest technology which would help us to reduce the effort and make life easy. Air Pollution control and fuel consumption are 2 major issues that need to be tackled for a greener and sustainable earth.

**2. PROJECT SUBSCRIPTION AND GOALS**

**2.1 Description**

This project is a system that monitors the amount of time invested by a user to park his vehicle which leads to different problems in the society. It uses Ultrasonic sensor to receive the cars parked and leaved at every parking stations, which helps us to count the parking spaces in the lt. The sensors are connected with the hardware-based system and sends data to the system.

The Arduino further sends the information of the parking slots to the server, where it is used to display out the number of available parking spaces and number of allotted parking spaces in the slot, which helps user to pre-book a parking slot based on his preference. Further, the system also ensures the vehicle safety as it generates text OTP and QR code in the app which is used to take vehicle from parking slot. All the data is stored on Google Cloud server FIREBASE. We chose firebase over other cloud services as the delay time is very less and Google provides the security to it’s servers in a very great way and all the data is access is also fast and easy from the servers.

This project has been divided into 4 modules, as follows:

* Hardware Module: This is the most tangible part of the project, consisting of the Arduino, the sensors. The Arduino collects data from the Ultrasonic sensor and processes the data from the sensors and sends the appropriate ate information to the Python module.
* Backend Module: This module is responsible for receiving data from the Arduino and creating entries in the FIREBASE API.
* Software Module: This module is responsible for creating user friendly app for a GUI model of the product which helps clients to interact with the system.
* Reservation Module: This module works in conjunction with the Hardware Module and the Backend Module, providing the user with real-time information about free parking slots. The user must have the Android App downloaded on his device to use this functionality. This module is also used to book the parking slots from the app.

**2.2 Goals**

The Project will visually display the number of parking slots available and booking a parking slot based on user’s activity. The main aim of the project is to contribution of something to society to reduce pollution and give an economic solution for humans which is beneficial.

**3. TECHNICAL SPECIFICATIONS**

**3.1 Hardware Specification**

1. Arduino

|  |  |
| --- | --- |
| Processor | ATmega328 |
| Operating Voltage | 5V |
| Flash Memory | 32KB |
| I/O Ports | 14 |
| EPROM | 1KB |
| Clock Speed | 16Mhz |

1. IR Sensor

|  |  |
| --- | --- |
| Supply Voltage | 5V |
| Current Consumption | 15mA |
| Max Range | 400cm |
| Min Range | 3cm |
| Resolution | 1cm |
| Frequency | 40K Hz |

1. Breadboard & Others.

|  |  |
| --- | --- |
| Breadboard Rows | 5 |
| Jumper Wires | 7.9mm |
| LED | Voltage: 5V |

**3.2 Software Specifications**

* **Python**

Python is an open-source, high-level language used for general-purpose computing developed by Guido Van Rossum in 1991. An interpreted language, it has a design philosophy which emphasized code readability by incorporating english keywords, indents and elimination of semicolons. Python allows for developers to include a myriad of third-party libraries, making it easier to focus on the workﬂow rather than reinventing the wheel.

* **Arduino Object-C**

In fact, you already are; the Arduino language is merely a set of C/C++ functions that can be called from your code. Your sketch undergoes minor changes (e.g. automatic generation of function prototypes) and then is passed directly to a C/C++ compiler (avr-g++). All standard C and C++ constructs supported by avr-g++ should work in Arduino. For more details, see the page on the Arduino build process.

* **Google Charts API**

The Google Chart API is an interactive Web service (now deprecated) that creates graphical charts from user-supplied data. Google servers create a PNG image of a chart from data and formatting parameters specified by a user's HTTP request. The service supports a wide variety of chart information and formatting. Users may conveniently embed these charts in a Web page by using a simple image tag. Originally the API was Google's internal tool to support rapid embedding of charts within Google's own applications (like Google Finance for example). Google figured it would be a useful tool to open up to web developers. It officially launched on December 6, 2007.

* **Firebase API**

Firebase provides a Realtime database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. The company provides client libraries that enable integration with Android, iOS, JavaScript, Java, Objective-C, swift and Node.js applications. The database is also accessible through a REST API and bindings for several JavaScript frameworks such as AngularJS, React, Ember.js and Backbone.js. The REST API uses the Server-Sent Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the Realtime database can secure their data by using the company's server-side-enforced security rules. Cloud Firestore which is Firebase's next generation of the Realtime Database was released for beta use.

* **Way2SMS API**

Way2sms offers unlimited free SMS to Indian numbers. It was founded by Raju Vanapala in 2006. It is first of its kind in India which allowed person to person free communication by SMS; later features like group SMS, email alerts, social media integration were added. Way2sms promises to deliver more than 95% of messages in less than even 10 seconds.

* **SendGrid Email API**

SendGrid provides a cloud-based email delivery service that assists businesses with email delivery. The service manages various types of email including shipping notifications, friend requests, sign-up confirmations, and email newsletters. It also handles internet service provider (ISP) monitoring, domain keys, sender policy framework (SPF), and feedback loops. Additionally, the company provides link tracking, open rate reporting. It also allows companies to track email opens, unsubscribes, bounces, and spam reports. Beginning in 2012, the company integrated SMS, voice, and push notification abilities to its service through a partnership with Twilio.

* **Arduino IDE**

Arduino is open-source hardware. The hardware reference designs are distributed under a Creative Commons Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available. The source code for the IDE is released under the GNU General Public License, version 2. Nevertheless, an official Bill of Materials of Arduino boards has never been released by Arduino staff.

* **Github**

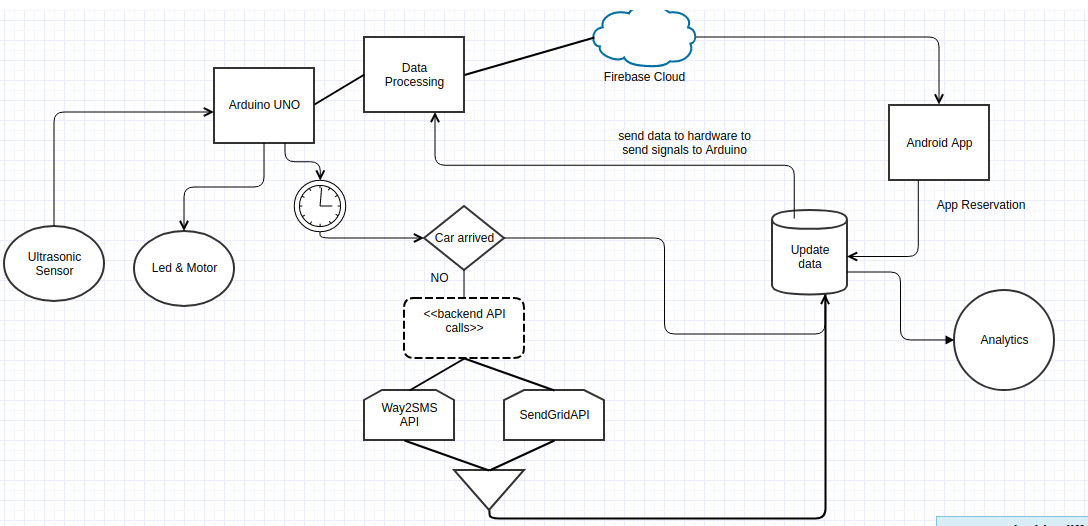
GitHub (originally known as Logical Awesome LLC) is a web-based hosting service for version control using git. It is mostly used for computer code. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers plans for both private repositories and free accounts which are commonly used to host open-source software projects] As of April 2017, GitHub reports having almost 20 million users and 57 million repositories, making it the largest host of source code in the world.

**4. ENGINEERING DESIGN**

**4.1 Design Approach**

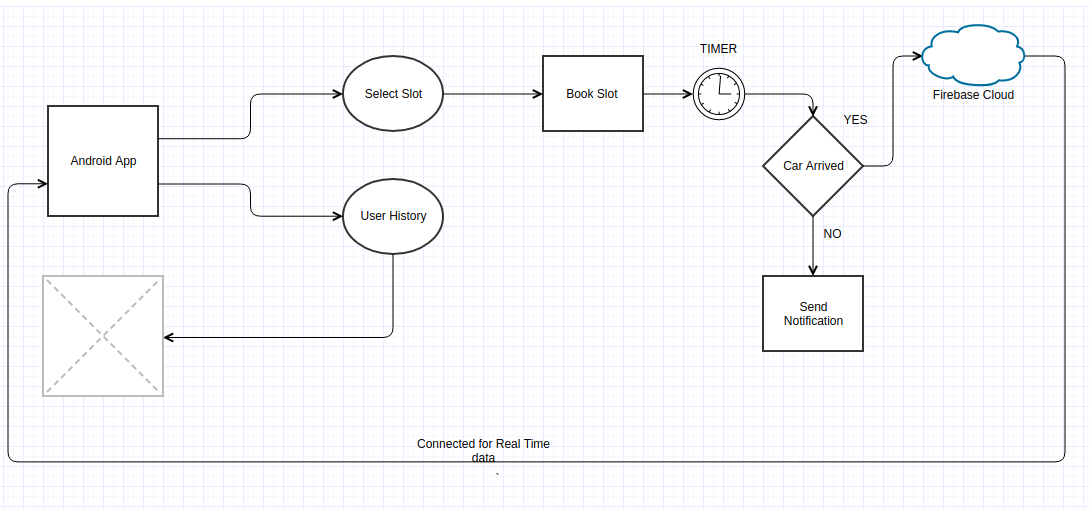
DIAGRAM



*Figure 1: Engineering Design*

**4.2 Flow Diagram**

DIAGRAM

*Figure 2: Flow Diagram*

**4.1.1 Hardware**

The hardware is the most tangible part of the system, consisting of a Arduino UNO, a Ultrasonic Sensor. The Arduino runs a Python script that loops over a set of procedures to be performed. It performs the following functions every iteration:

• Check for number of available parking slots in the area.

• Listen continuously from Firebase.

• Retrieve parking information, if the car is being parked or moving out of parking.

• Use Python to communicate with Firebase API

• Use software module to book a parking space in the area.

• Collect the data in and send it to the server.

**4.1.1.1 Setup**

* The Sensors are placed in the parking area and per parking there would be 1 sensor required which would detect the data for car park.
* The Arduino is the central body through which all the sensors and Python module are connected, depending on sensors number of Arduino’s are decided.
* The Arduino is connected to Python Script, as the Python script continuously reads data from FirebaseAPI.

**4.1.2 Software Interface**

The Application is the central system allowing us to obtain a bird’s eye view of the entire system. It contains main buzz area that shows us real-time parking numbers and visual statistics of every parking area connected to the system. The central system is constituted by the APIs, graphing libraries, dashboard interface and other supporting backend code. The dashboard, a Firebase API is deployed on the cloud. It is connected to a live API endpoint which all the parking slots hit with a HTTP request in JSON format hit this endpoint with a JSON request, passing an Authorization token in the request header, allowing the dashboard data to be updated in real-time. The data is the passed on to the dashboard which displays them using tooling such as the Google Charts API graphing library.

**4.1.3 Analytics**

This stage is the most important stage of the workﬂow as the data collected and managed in the earlier stages will be analyzed and put to use in this stage. The following is done:

* A smart way for efﬁcient and economic parking of vehicles from the sources.
* Real time visual representations of the garbage statistics on real life maps
* Analysis and Education of users regarding proper utilization of land and parking areas so as to control pollution and fuel wastage.

**4.2 Codes and Standards**

**4.2.1 Object Oriented Paradigm**

Object-oriented analysis, design, and programming are proven powerful means to master the complexity inherent in the development process of large, distributed applications. The ingredients and benefits of object orientation are well described by Booch in [Boo94, S. 2,]: the object-oriented conceptual framework, called the object model, is based on the combination of the principles of abstraction, encapsulation, modularity, and hierarchy, often supplemented by the principles of typing, concurrency, and persistence. According to Booch, the usage of the object model for the design process of an application offers several advantages over traditional techniques:

* The application of the object model helps substantially to develop complex programs.
* Only object-oriented analysis and design permit to fully exploit the expressive power of modern object-oriented programming languages.
* The object model facilitates the reuse of both code and designs.
* Systems designed according to object-oriented principles can evolve over time and can be adapted easily to accommodate future demands, without necessitating the abandon or complete restructuring of the original design.
* Object orientation reduces the risks of software development.
* The object model resembles human cognition more closely than traditional design paradigms

**4.3 Constraints and Alternatives**

**4.3.1 Design Constraints**

* The subject (Car) should be parked in from the front side of the sensor, so as to detect it.
* The sensor assumes that car is parked when the distance between the object and sensor is approx. to 4cm.
* The sensor assumes that car is un parked when the distance between car and sensor is greater than a particular limit, it’s set to 10cms.
* The timer waits for approx. 5 mins to allow a user to take his car to parking slot, else the reservation is cancelled.
* Internet is required as to call Firebase Cloud API, if the internet is down due to some reason, then the system goes down.

**4.3.2 Component Alternatives**

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Alternative | Advantages over existing component | Disadvantages over existing compo. |
| Arduino | Rasp. Pi | More powerful | Costly and high maintenance needed |
| Ultrasonic Sensor | IR Sensor | Senses |  |
| Firebase | Aws | Fast and accurate | Expensive |

*Figure 4.1: Component Alternatives*

**5. SCHEDULE, TASKS & MILESTONES**

**5.1 Task Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| Tasks | Start date | End date | Description |
| Project Brainstorming | 30/11/17 | 1/12/17 | Searching and exploring more on the topic, what is currently done and what can be done. |
| Literature Survey | 1/12/17 | 2/12/17 | Survey for existing products related to this and how they’re currently implemented, how we can increase productivity. |
| Break Points into modules | 2/12/17 | 4/12/17 | Division of whole project into separate modules. |
| Design phase for each module | 5/12/17 | 8/12/17 | Design for Every module such as Firebase DB, Android App, Arduino Design. |
| Module separation and assignment | 9/12/17 | 15/12/17 | Separation and assignment of module between us. |
| Hardware Module | 16/12/17 | 30/12/17 | Hardware module, implementation of hardware |
| UX Module | 3/01/18 | 16/01/18 | UX Module Sub Division |
| Testing Phase 1 | 17/01/18 | 19/01/18 | Basic Testing of Hardware |
| Review 1 | 20/01/18 | 25/01/18 | Preparation for Review 1 |
| Firebase Module | 26/01/18 | 08/02/18 | Learning about Firebase and start creating db to insert data and fetch data. |
| App Module | 15/01/18 | 24/02/18 | User Interface, what can be done on client side and implementation of Android App. |
| Review 2 | 20/02/18 | Till date | Preparation of review 2 |
| Sending SMS / Email | 20/03/18 | 22/03/18 | Different API’s to interact with for SMS and email. |
| Integration with API | 23/03/18 | 24/03/18 | Integration of all API into our system. |
| Integration of Android with Firebase | 25/02/18 | 30/03/18 | Integration of Android and Firebase Cloud. |
| Testing Phase 2 | 20/02/18 | 22/02/18 | Testing whether all things are working |
| Integration of Android with hardware module | 20/03/18 | 27/03/18 | Integration of App and other modules. |
| Testing Phase 3 | 27/03/18 | 31/03/18 | Testing with all Integrations |
| Review 3 | 5/04/18 | Till date | Preparation for review 3 |
| Report Draft | 20/03/18 | 5/04/18 | Preparation of Report |

*Table 5.1 Project Schedule*

**5.2 Gannt Chart**

**6. PROJECT DEMONSTRATION**

**Photos.**

**7. COST ANALYSIS**

|  |  |  |
| --- | --- | --- |
| S.NO | Particular | Cost (INR) |
| 1 | Arduino UNO | 550 |
| 2 | Ultrasonic Sensors | 225\*2 = 450 |
| 3 | Jumper Wires (M2M, F2F, M2F) | 140 |
| 4 | Breadboard | 120\*2 = 220 |
| 5 | Firebase API (upto 100 Connections) | N.A. |
| 6 | SendGrid API (40K Emails per Month) | N.A. |
| 7 | Way2SMS (100 SMS per day) | N.A. |
|  | Total | 1360 Approx. |

*Table 7.1 Cost Analysis*

**8. CONCLUSION**

**8.1 Project Summary**

The aim of this project was to Identify and help the eco system over the problem of over air pollution and decrease the fuel use as it’s an important resource which we need to save it for future purpose. The Project aims to reduce the traffic time, avg. fuel usage and reduce air pollution by some manner. There have been many steps by the govt. as to prevent all this over usage and misuse of resources, but none of them have been much affected the public.

The sensors continuously sense the data (object) and send the data to Arduino which then sends the data to the Firebase API so as to keep the data in real time, so that the counter increments/ decrements in real time on the user app. Now, here the user has this choice of booking the parking slot so that he can arrive at his parking within short duration of time and thus reduces his contribution to traffic indirectly.

**8.2 Project Conclusion**

This has been a great experience in learning about the real-world implementation of a product. Thus, the project accomplishes the aim of reducing the air pollution and fuel usage as about 10% of the fuel per day goes in wastage for just only searching parking spaces, about 5-8% of time of user goes in searching parking spaces. Thus, the project comes up with the solution that if you’re nearby to the parking space, you can pre-book a parking slot and go there and park your vehicle at your desired time. This, reduces the user effort to search for parking space, his time to search for parking space and etc. Additionally, this could be also seen as a business model, as we can Integrate a way of Premium users, for which the users have to pay for some extra amount so as to attain privileges.

The Android app was chosen as User Interface as because in today’s world almost every one has a smart phone, and he can install the app, so this was the best way of communication with user, as to attain more attraction and viewership from users. The system also sends text messages and emails if anything wrong goes with the car or anything for notifications.

**9. REFERENCES**

* Way2SMS API: *https://github.com/kingster/Way2SMS-API*
* SendgridAPI*: https://sendgrid.com/docs/API\_Reference/index.html*
* Firebase API: *https://firebase.google.com/*
* Arduino Community
* Stack overflow
* Android Community
* Google Charts API: [*https://developers.google.com/chart/*](https://developers.google.com/chart/)
* Smart Parking Research Paper:
  + [*http://ijariie.com/AdminUploadPdf/Reasearch\_paper\_on\_SmartCity\_Parking\_System\_ijariie2098.pdf*](http://ijariie.com/AdminUploadPdf/Reasearch_paper_on_SmartCity_Parking_System_ijariie2098.pdf)
  + [*http://utc.ices.cmu.edu/utc/CMU%20Reports%202013%202/UTC\_FinalReport\_Smart\_Parking\_Hampshire.pdf*](http://utc.ices.cmu.edu/utc/CMU%20Reports%202013%202/UTC_FinalReport_Smart_Parking_Hampshire.pdf)

**APPENDIX A – SOURCE CODE**

**APPENDIX B – COURSES INVOLVED**

|  |  |  |
| --- | --- | --- |
| S.NO | Course | Module |
| 1 | Computer Architecture | Hardware Module (Arduino) |
| 2 | DBMS | DB Design for Firebase |
| 3 | Internet and Web Programming | Charts for analytics in JS |
| 4 | Computer Networks | Communication via HTTP Protocol (API) |
| 5 | Software Engineering | Agile based methods for developing project |
| 6 | Object Oriented Programming | Android App, Firebase Objects in DB |
| 7 | Data Structures and Algorithms | Android Application (use of lists etc.) |
| 8 | Operating Systems | Threads in Android Applications |
| 9 | Embedded Systems | Hardware Module |
| 10 | Digital Logic | Hardware Module |

**SOFTWARES NEEDED**

* Arduino IDE
* Python IDE
* Android Studio
* Postman for testing API’s
* OS (Linux, Windows)