ISSN (e): 2250 – 3005 || Volume, 09 || Issue, 6 || June– 2019 ||
International Journal of Computational Engineering Research (IJCER)

Smart Parking System Implementation using Internet of Things (IoT) and Cloud Computing

Knvs Sridevi
Assistant Professor, Department of Information Technology, CBIT, Hyderabad, India
Bellamkonda Urvashi
Department of Information Technology, CBIT, Hyderabad, India
Yenigalla Satvika
Department of Information Technology, CBIT, Hyderabad, India

ABSTRACT:*Finding an empty parking spot is one of the toughest and most time-consuming job in the places with very high vehicle density. This project focuses on simulating a cloud based smart parking system using Internet of Things (IoT) technologies which helps the user to find an available parking spot hence minimizing the user waiting time. This application allows the user to reserve a parking slot minutes earlier before they reach the vehicle parking space.*
Users are uniquely identified using RFID tags and are allotted an empty space for parking. Users can keep track of all the parking slots, their parked vehicles, their parked time and the parking cost through an online cloud-based web application or a mobile application. Check-ins and check-outs will be handled in a fast manner without having to stop the cars so that traffic jam problem will be avoided during these processes. Drivers will not have to stop near the parking ticket machine to collect his ticket thereby reducing traffic jams and the usage of parking tickets during check-ins and check-outs. Vehicle owners will not have to make any payments at each check-out thus a faster traffic flow will be possible. Since there won't be any waiting during check-ins and check-outs process.

Date of Submission: 21-06-2019 Date of acceptance:05-07-2019

I. INTRODUCTION

To implement traffic management system, a smart parking system was created to reduce the cost of human resource and efficient use of resources for car-park owners. Currently, the common technique of finding a parking lot is manual. Finding parking slot with high vehicle density is most difficult and consumes lots of time and effort.

1.1 Objective

This system enables users to check the parking space before they reach the parking lot and it also provides users with a feature of reserving the space beforehand. The bill is also generated. The main aim of the project is to build a system, which involves as less human force as possible, and speed up the process of parking.

1.2 Existing Parking System

The existing parking system is partially automated and involves a lot of human force. The existing system involves a ticket vending machine from which a ticket is generated at the entrance which should be produced to a person at exit who will collect the money from you after checking the duration for which you parked your vehicle. After entering the lot, a person may or may not be present to direct you to the available space. If there is no one to guide, it is on the driver to find a parking space to park his vehicle.

1.3 Problems in Existing System

The present parking system is luck based and the driver may or may not find the parking space at the end. The issuing of tickets and payment of bills involve a lot of human force. The ticketing hassle leads to traffic jams. There is also no guarantee that the user may find an empty parking spot.

1.4 Proposed system

The proposed system is a smart parking system in which the registered user can check the available spaces in the android app. He is also given an option to reserve the slot in advance if he wants to. The billing process is also automated. There is no hassle and overhead of waiting to collect tokens and to pay the bills. All the information about the lot is updated in the database through a cloud server which gets the information from the sensors in the lot.

1.5 Advantages of the Proposed System

The system that is proposed is an automated system which enables users to find a parking space within seconds without any difficulty. The user can reserve and check the availability before reaching the lot thereby ruling out any possibilities of not finding a parking space in the end. The billing is done based on the information sent from the sensors to the cloud server. Hence bills are generated online resulting in free flow of vehicles.

II. LITERATURE REVIEW

2.1 Internet of things (IoT)

The inter-networking of vehicles (also referred as "connected devices" or "smart devices"), physical devices, high-rise buildings embedded with software, actuators, electronics and Network connectivity that allow these objects to collect and interchange data.

The IoT allows objects to be recognized or controlled remotely through current network organization, building opportunities for more direct integration of the physical world into computer-based systems, and resulting enhanced efficiency, correctness and economic benefit in addition to minimizing human involvement. When IoT is associated with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, virtual power plants, smart homes, smart transportation and smart cities.

2.2 Cloud Computing

Cloud computing is Internet-based computing that offers shared computer processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., computer networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in either privately owned, or third-party data centers that may be located far from the user–ranging in distance from across a city to across the world.

2.3 RFID

Radio Frequency Identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. The tags contain electronically stored information. Passive tags collect energy from a nearby RFID reader’s interrogating radio waves. Active tags have a local power source such as a battery and may operate at hundreds of meters from the RFID reader. Unlike a barcode, the tag need not be within the line of sight of the reader, so it may be embedded in the tracked object.

RFID is one method for Automatic Identification and Data Capture (AIDC). RFID belongs to a group of technologies referred to as Automatic Identification and Data Capture (AIDC). AIDC methods automatically identify objects, collect data about them, and enter those data directly into computer systems with little or no human intervention. RFID methods utilize radio waves to accomplish this. At a simple level, RFID systems consist of three components: an RFID tag or smart label, an RFID reader, and an antenna. RFID tags contain an integrated circuit and an antenna, which are used to transmit data to the RFID reader (also called an interrogator). The reader then converts the radio waves to a more usable form of data. Information collected from the tags is then transferred through a communications interface to a host computer system, where the data can be stored in a database and analyzed later.

2.4 NoSQL Database

A NoSQL database provides a mechanism for storage and retrieval of data which is modeled other than the tabular relations used in relational databases. NoSQL databases are increasingly used in big data and real-time web applications. NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support SQL-like query languages. Motivations for this approach include: simplicity of design, simpler "horizontal" scaling to clusters of machines (which is a problem for relational databases), and finer control over availability.

The data structures used by NoSQL databases (e.g. key-value, wide column, graph, or document) are different from those used by default in relational databases, making some operations faster in NoSQL. The suitability of a given NoSQL database depends on the problem it must solve. Sometimes the data structures used by NoSQL databases are also viewed as "more flexible" than relational database tables.

2.5 ARDUINO

Arduino is an open source, computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone.
Arduino boards are available commercially in preassembled form, or as do-it-yourself kits. Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

2.6 Android Studio IDE

Android Studio is the official Integrated Development Environment (IDE) for the Android platform. It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0. Android Studio was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.
Based on JetBrains IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, MacOS and Linux and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

2.6.1 Features provided are

i. Gradle-based build support
ii. Android-specific refactoring and quick fixes
iii. Lint tools to catch performance, usability, version compatibility and other problems
iv. ProGuard integration and app-signing capabilities
v. Template-based wizards to create common Android designs and components
vi. A rich layout editor allows users to drag-and- drop UI components, option to preview layouts on multiple screens
vii. Support for building Android apps
viii. Built-in support for Google Cloud Platform, enabling integration with Google Cloud Messaging and App Engine
ix. An Android Virtual Device that is used to run and debug apps.

2.7 Firebase Console

Firebase is a mobile and web application platform with tools and infrastructure designed to help developers build high-quality apps. Firebase is made up of complementary features that developers can mix-and- match to fit their needs. The team is based in San Francisco and Mountain View, California. The company was founded in 2011 by Andrew Lee and James Tamplin. Firebase's initial product was a real-time database, which provides an API that allows developers to store and sync data across multiple clients. Over time, it has expanded its product line to become a full suite for app development. The company was acquired by Google in October 2014and a significant number of new features were featured in May 2016 at Google I/O.

III. METHODOLOGY

3.1 Proposed Algorithm

3.1.1 Steps

1. The user logs into the mobile app.
2. The parking map displays the free slots in the parking area.
3. If the user wishes to reserve a slot in the parking area
a. Reserve button is clicked, proceed to step 5 onwards.
b. Exit the system.
4. Notification sent via cloud to Arduino board.
5. Status of the reserved area will be updated in the application.
6. The user enters the parking lot and his RFID is scanned and authenticated.

7. If the user is authentic and there are free spaces available then
a. Allow the user, proceed to step 8.
b. Don't allow the user and end the system.
8. If the user has made a reservation then
a. The user is redirected to his reserved spot, proceed to step 9.
b. The user is allotted the nearest available free slot and the timer starts.
9. When the user leaves the lot the duration of his stay is calculated and respective bill will be generated.

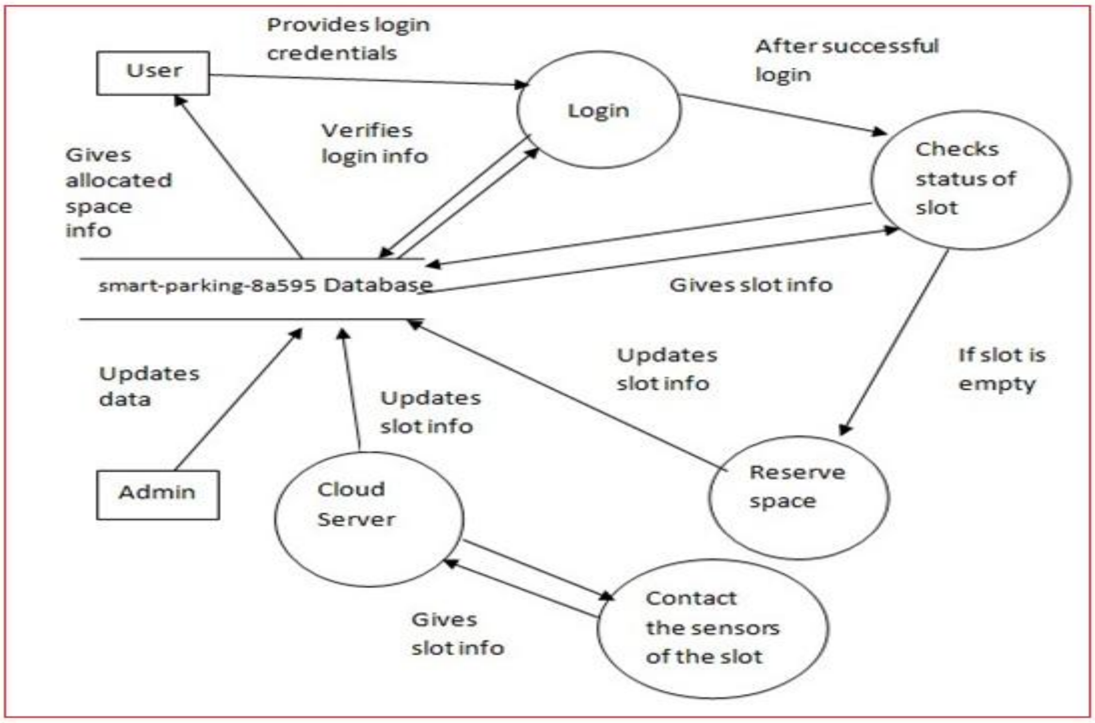


Figure 1: Flow Diagram

As shown in figure 1 the user can login to the android application which is connected to the cloud server and fetches the data from the parking lot. The user has to login to check the status of the available parking spaces and reserve them beforehand. The cloud server is directly connected to the micro controller which gives the RFID value and status of the parking lot with the help of sensors connected to it. If the slot is empty, the user can reserve it and the information is updated in the database.

IV. IMPLEMENTATION

4.1 IoT (Internet of Things)

The parking lot is equipped with various hardware devices such as Arduino Mega

4.1.1 Arduino MEGA

Figure 2: Arduino Mega 2560

The Arduino Mega in Figure 2 is a microcontroller board based on the ATmega1280 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.
The microcontroller is programmed in such a way that whenever the RFID readers detect a RFID card, the data is sent to the cloud server and the output is displayed using the leads placed at the entrance and exit.

4.1.2 ETHERNET SHIELD

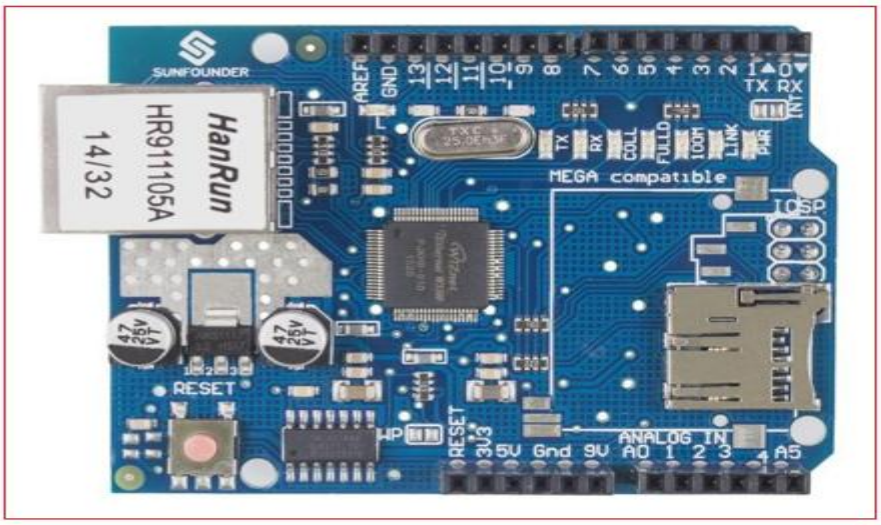


Figure 3: Ethernet W5100 Shield

The Arduino Wiznet Ethernet W5100 Shield shown in figure 3. Allows an Arduino board to connect to the internet. It is based on the Wiznet W5100 ethernet chip providing a network (IP) stack capable of both TCP and UDP. The Arduino Ethernet Shield supports up to four simultaneous socket connections.
Using the ethernet shield, the Arduino mega board is connected to the internet, with the help of which it communicates with cloud server.

4.1.3 RFID Reader

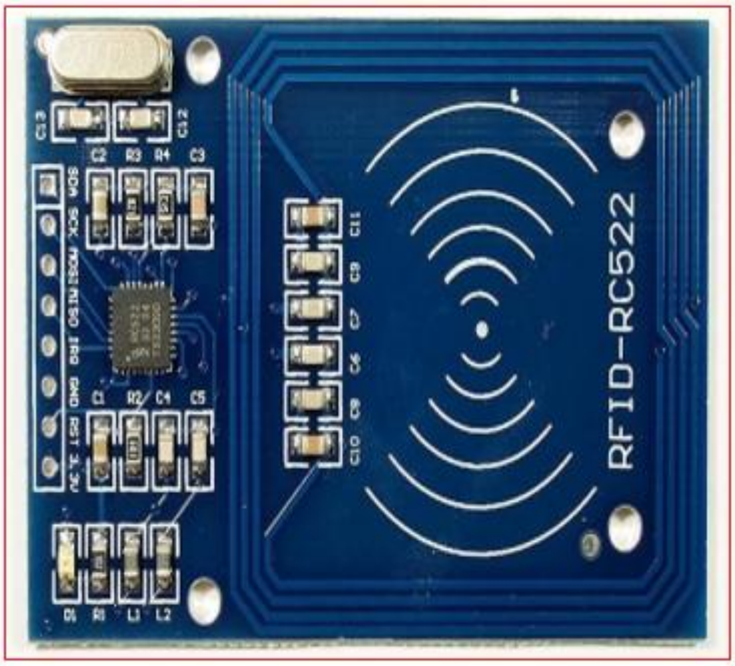


Figure 4: MFRC522 RFID Reader

The MFRC522 (Figure 4) is a highly integrated reader/writer IC for contactless communication at 13.56 MHz. The parking lot is equipped with two such readers, one at the entry and the other at the exit.
This RFID reader is used to read the RFID cards installed in the cars. Every car (user) is given a unique RFID using which the car is identified and that user will be billed. The RFID value read by the reader is sent to the cloud with the help of the microcontroller and the ethernet shield.

4.1.4 Circuit Diagram

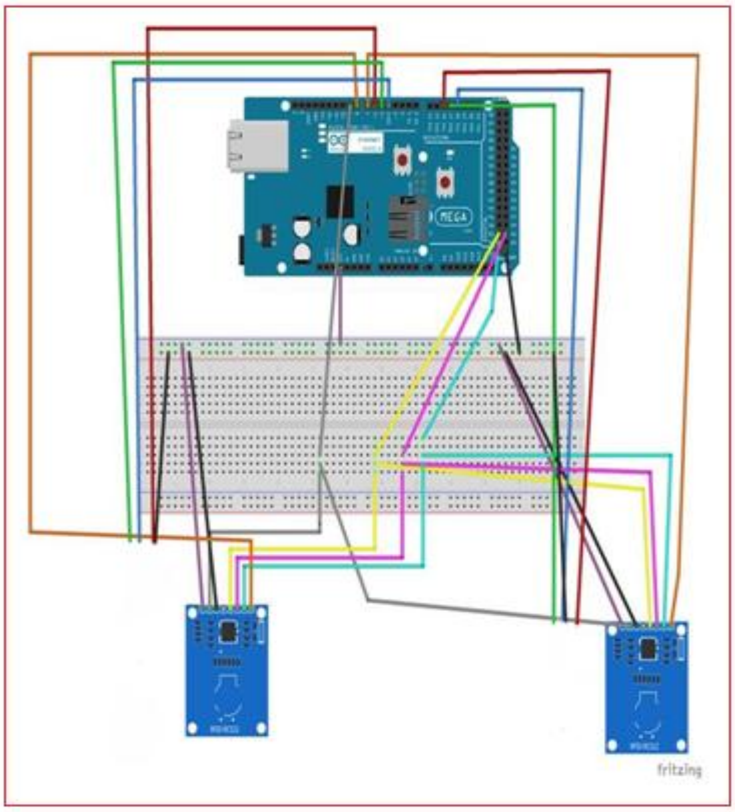


Figure 5: Circuit Diagram of all the components

All the components are connected as shown in the Figure 5 for proper working of the system. Two RFID readers are placed, one at the entrance and one at the exit. The reader at the entrance reads the RFID and starts the timer of the checked in user. The reader at the exit reads the RFID and stops the timer to generate a bill.

4.1.5 Architecture

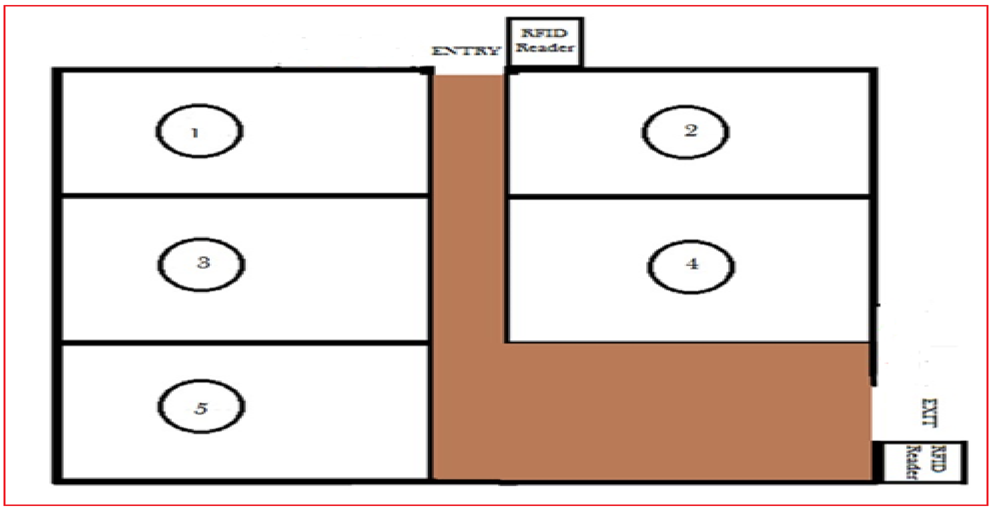


Figure 6: Architecture of the Parking Lot

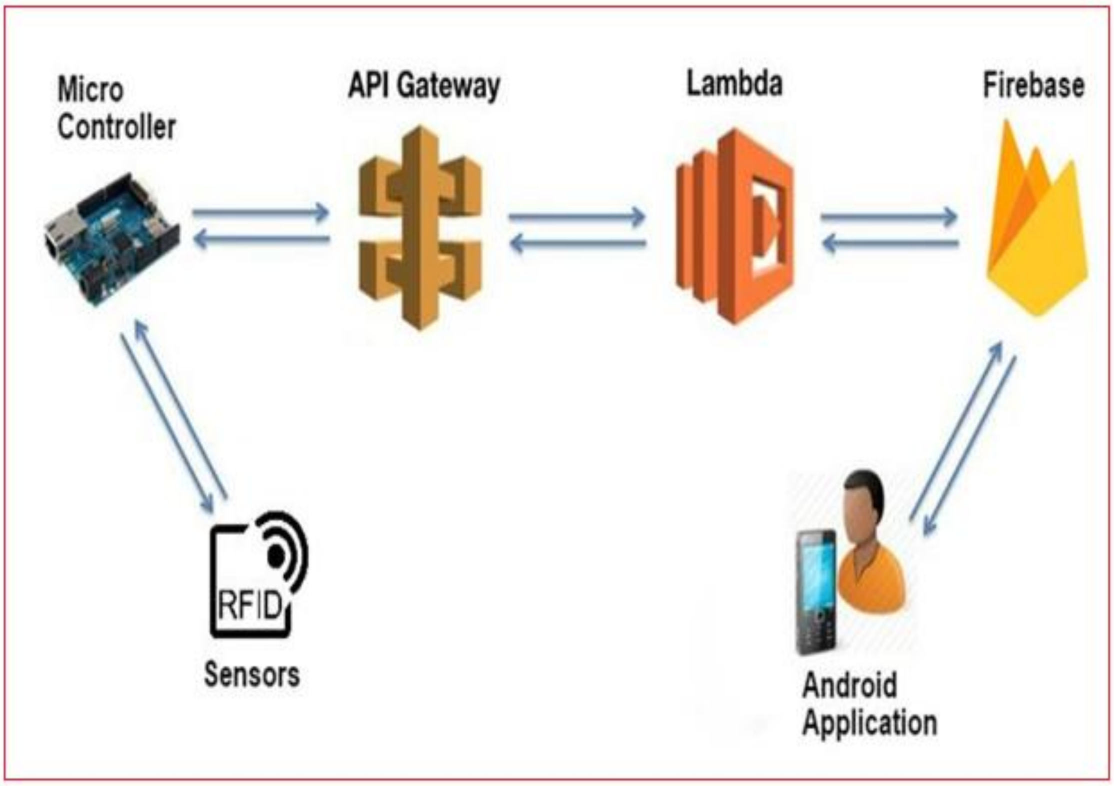


Figure 7: Working and Flow of Data

As shown in Figure 7 the sensors are connected to the microcontroller and the data received from these sensors is sent to the Cloud Server (AWS Lambda) through the API Gateway, which in turn makes changes in the database (Firebase).The changes in the database are reflected in the android application in real time. The changes made by the user in the Android application are also reflected in the Firebase in real time.

4.2 Cloud Server

4.2.1 AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second. You pay only for the compute time you consume - there is no charge when your code is not running. With AWS Lambda, you can run code for virtually any type of application or backend service - all with zero administration. AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the computer resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. All you need to do is supply your code in one of the languages that AWS Lambda supports (currently Node.js, Java, C# and Python).
RFID details are scanned by the microcontroller and sent to the cloud server using REST calls. The RFID details sent by the microcontroller is checked and authenticated with the real time database and checks and allot slots to the user. The Lambda micro server acts as the core cloud server to the application and handles all the interaction between the hardware and the database. The server returns success or failure to the microcontroller telling it whether the user should be allowed inside or not.

4.2.2 API Gateway

Amazon API Gateway supports the following two major functionalities:

i. It lets you create, manage and host a RESTful API to expose AWS Lambda functions, HTTP endpoints as well as other services from the AWS family including, but not limited to, Amazon DynamoDB, Amazon S3 and Amazon Kinesis. You can use this feature through the API Gateway REST API requests and responses, the API Gateway console, AWS Command-Line Interface (CLI), or an API Gateway SDK of supported platforms/languages. This feature is sometimes referred to as the API Gateway control service.

ii. It lets you or 3rd-party app developer call a deployed API to access the integrated backend features, using standard HTTP protocols or a platform- or language-specific SDK generated by API Gateway for the API. This feature is sometimes known as the API Gateway execution service.

The API you create in API Gateway consists of a set of resources and methods. A resource is a logical entity that can be accessed through a resource path using the API. A resource can have one or more operations that are defined by appropriate HTTP verbs such as GET, POST, and DELETE. A combination of a resource path and an operation identify a method in the API.
Each method corresponds to a REST API request submitted by the user of your API and the corresponding response returned to the user. API Gateway integrates the method with a targeted back end by mapping the method request to an integration request acceptable by the back end and then mapping the integration response from the back end to the method response returned to the user. As an API developer, you can configure how methods are mapped to integrations and vice versa by stipulating what parameters to use and specifying mapping templates to transform payloads of given data models.

4.3 Cloud Database

4.3.1 Firebase

Firebase provides a real-time 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 cloud. The company provides client libraries that enable integration with Android, IOS, JavaScript, Java, Objective-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.
The database holds all the information of the registered users and details of the slots in the parking lot. It is connected to the cloud server and the Android application, accepts requests, and responds to them. The database being real time, any change happening in the database is reflected in the application on the spot.

4.3.2 Firebase Cloud Messaging

Formerly known as Google Cloud Messaging (GCM), Firebase Cloud Messaging (FCM) is a cross-platform solution for messages and notifications for Android, IOS, and web applications. This is used to send notifications to the user for every activity he performs allowing him to stay updated with every occurring action.

V. RESULTS

5.1 Results Screenshots

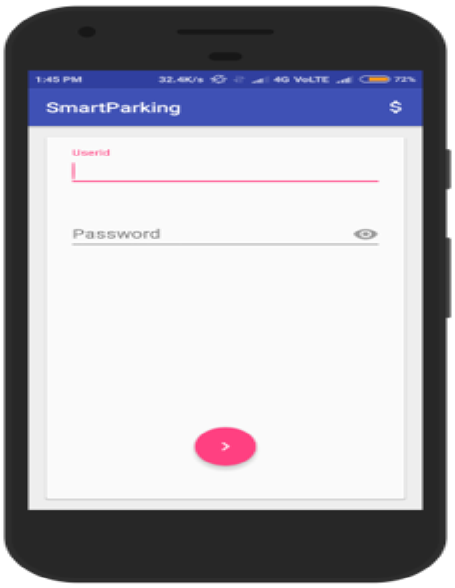
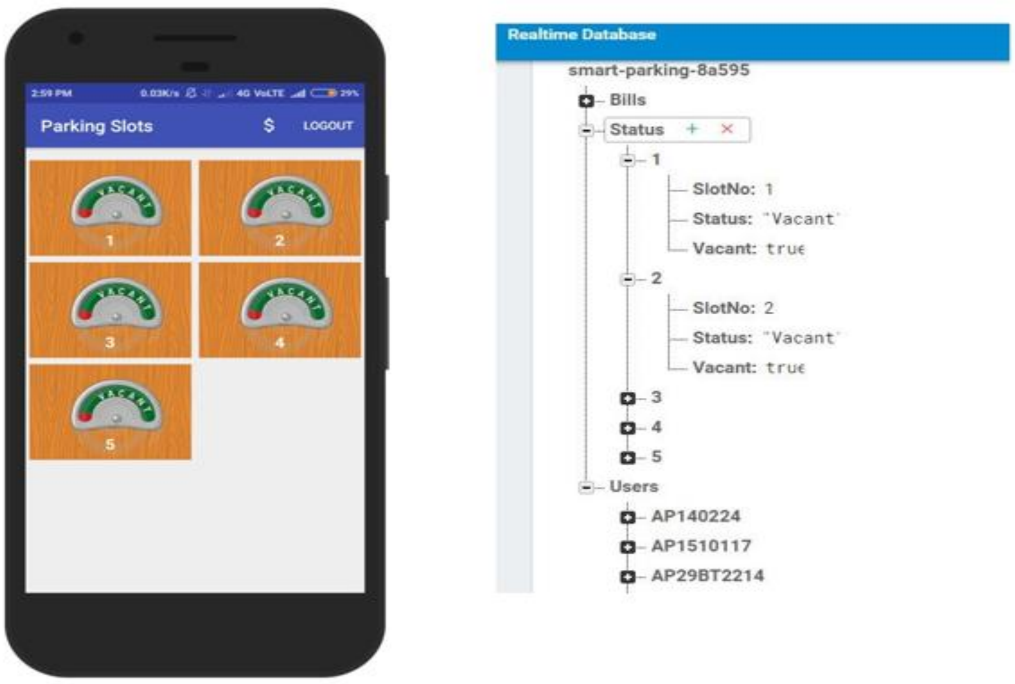


Figure 8: Login Screen

Figure 8 shows the login screen that appears on opening the app. The user logs in using the details given to the user by the admin at the time of issue of RFID card to the user. Every user is identified by a unique RFID. After the user enters the details, the details are verified in the database before showing him the home screen.



```
Realtime Database
smart-parking-8a595
  Bills
  Status  + ×
    1
      SlotNo: 1
      Status: "Vacant"
      Vacant: true
    2
      SlotNo: 2
      Status: "Vacant"
      Vacant: true
    3
    4
    5
  Users
    AP140224
    AP1510117
    AP29BT2214
```

Figure 9: Home Screen and Database structure

Figure 9 shows the home screen that is displayed to the user after he logs in to the app successfully using the Username and password given to the user at the time of issue of RFID card by the admin and the corresponding database screenshot where all the users and spaces details are stored.

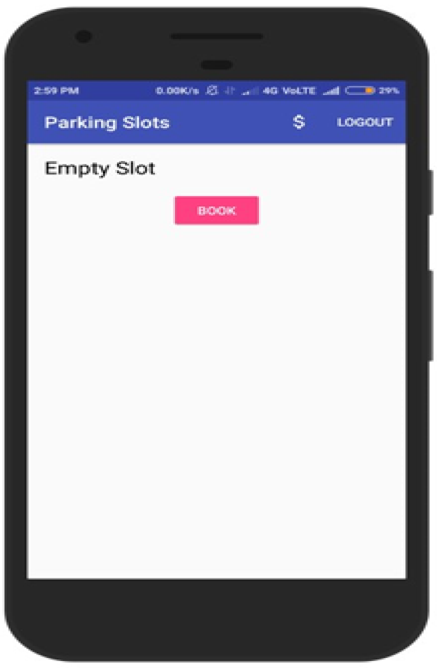
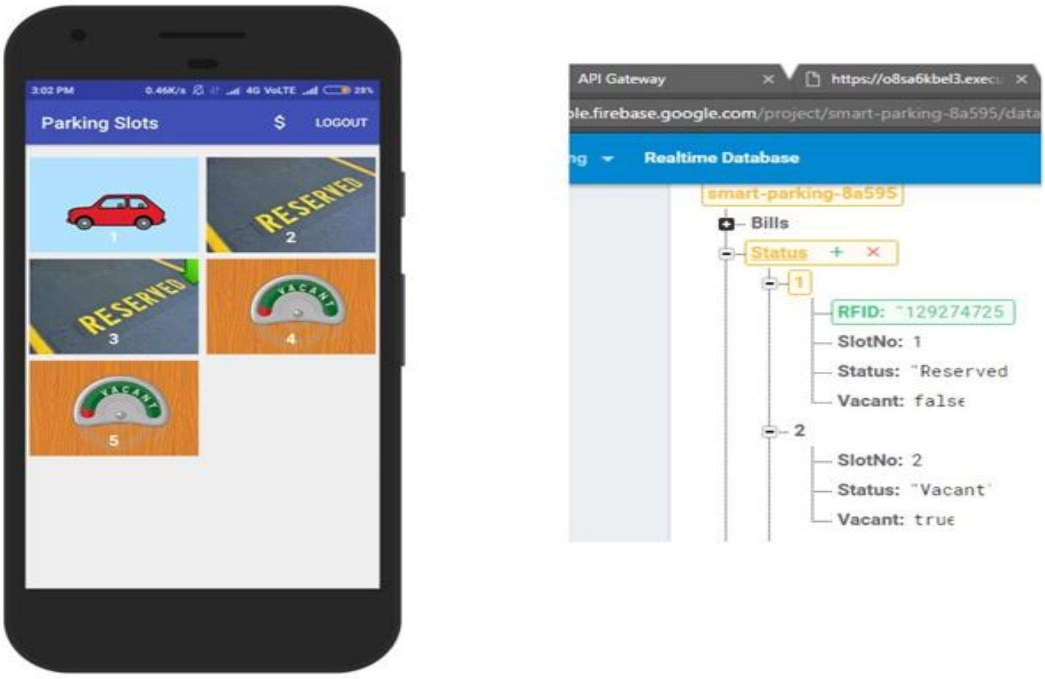


Figure 10: Reserving a Spot

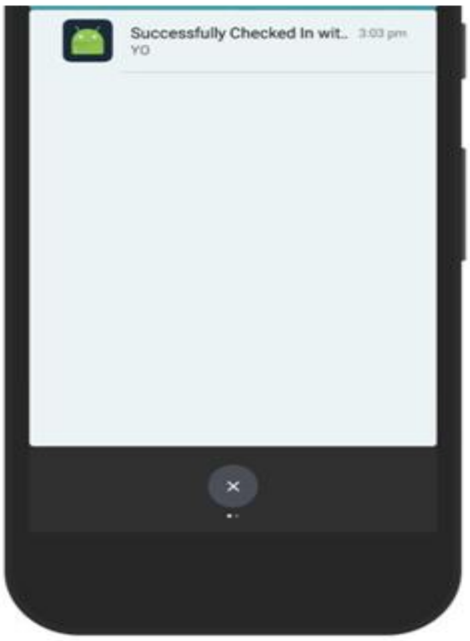
Figure 10 shows the screen that is displayed when the user selects a space to reserve. User can reserve a space only when it is empty. User is allotted this space to park his vehicle when he arrives at the parking lot. User can tap on the book button to confirm his reservation of the space.



```
smart-parking-8a595
 ⊞- Bills
 ⊟- Status  +  ×
    ⊟- 1
       RFID: "129274725
       SlotNo: 1
       Status: "Reserved
       Vacant: false
    ⊟- 2
       SlotNo: 2
       Status: "Vacant"
       Vacant: true
```

Figure 11: Slot reserved

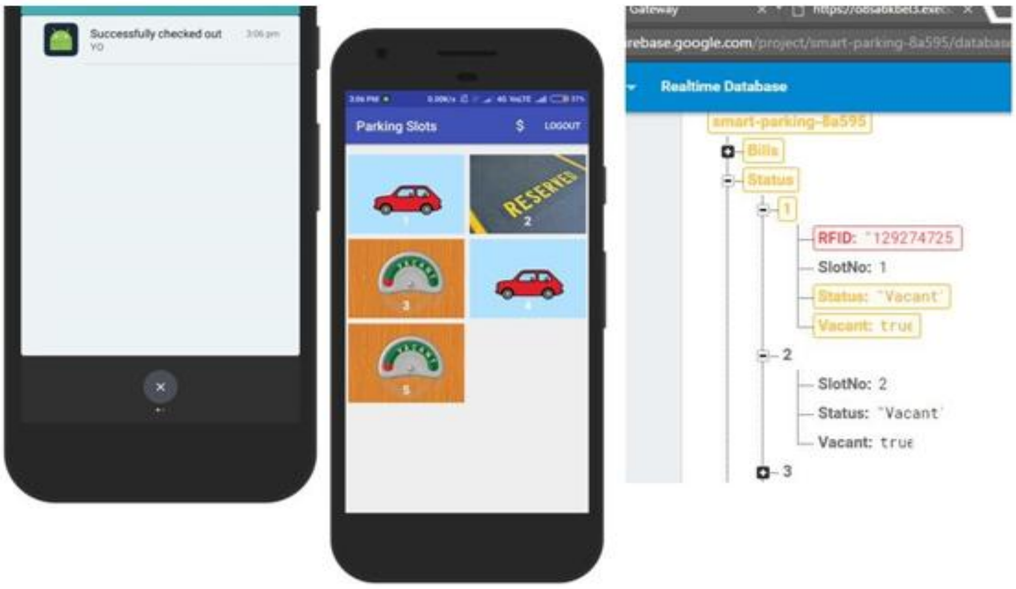
Figure 11 shows the screen that is displayed upon parking the vehicle, it shows the spots that are vacant, reserved and occupied. This screen is shown when the user arrives at the parking lot to shown to the user where the user's car is to be parked or is parked. A snapshot of the database after the vehicle is parked is also shown.



Successfully Checked In wit.. 3:03 pm
YO

Figure 12: Real-time Notification after entering the lot

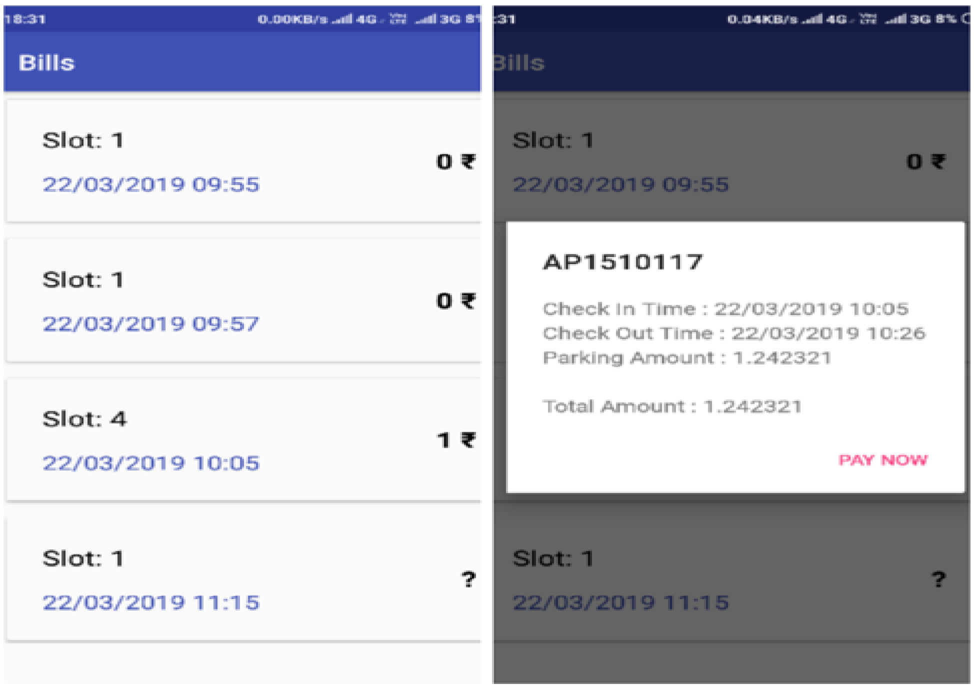
Figure 12 shows the notification sent to user after entering the lot. This notification is sent to the user when he arrives at the parking lot and after his RFID is read and verified in the database. Only users whose RFID is registered in the database will be allowed to enter the parking lot. After this notification is sent to the user, the user can see where the car should be parked in the app. The timer starts immediately after the user checks in.


Successfully checked out 3:06 pm
YO

1
RFID: "129274725"
SlotNo: 1
Status: "Vacant"
Vacant: true
2
SlotNo: 2
Status: "Vacant"
Vacant: true
3

Figure 13: Exiting the lot

Figure 13 shows the notification sent to the user after checking out of the lot, changes in the app after he checks out and the database snapshot after checking out. The user will be checked out only after the user's RFID is read by the reader at the exit of the parking lot. The timer is also stopped immediately after the user checks out and a bill is generated.


Bills

Slot: 1	22/03/2019 09:55	0 ₹
Slot: 1	22/03/2019 09:57	0 ₹
Slot: 4	22/03/2019 10:05	1 ₹
Slot: 1	22/03/2019 11:15	?

AP1510117
Check In Time : 22/03/2019 10:05
Check Out Time : 22/03/2019 10:26
Parking Amount : 1.242321
Total Amount : 1.242321
PAY NOW

Figure 14: Bill history and Bills

Figure 14 shows the bill history and the details of a bill along with an option to pay the bill. A bill is generated after the user checks out from the parking lot depending on the time for which the user's vehicle was parked for. The user is shown different bills based on the spaces where the user parked.

VI. CONCLUSION

The proposed system is a smart parking system in which the user can check the available spaces in the parking lot through an Android application thereby overcoming the problem of not finding the parking space in the end. The user can also reserve the parking space before he/she reaches the parking lot. The system does not involve any human force at all.

REFERENCES

[1]. Y. Geng and C. G. Cassandras, ``A new `smart parking' system based on optimal resource allocation and reservations," in Proc. 14th Int. IEEE Conf. Intell. Transp. Syst. (ITSC), Oct. 2011, pp. 979-984.

[2]. R. Buyya, "Cloud computing: The next revolution in information technology", Parallel Distributed and Grid Computing (PDGC) 2010 1st International Conference on, pp. 2-3, Oct 2010.

[3]. I. Wigmore, Internet of Things (IoT). Newton, MA, USA: TechTarget, Jun. 2014.

[4]. Fariza Norbaya R. Yusnita and Norazwinawati Basharuddin. Intelligent parking space detection system based on image processing. International Journal of Innovation, Management and Technology, 3(3), June 2012.

[5]. C.Laugier and F.Thierry, "Sensor-based control architecture for a car-like vehicle", International Conference on Intelligent Robots and Systems, 1998.

[6]. J. Li Lin, Changwei Zou, Research on Cloud Computing Based on Android Platform, 2010.

[7]. L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, R. Vergallo, "Integration of RFID and WSN in technologies in a smart parking system", Proc. 22nd International Conference Software Telecommunication Computer Network (SoftCOM), pp. 104-110, 2014.

[8]. X.L. Jia, Q.Y. Feng, C.Z. Ma, "An efficient anti-collision protocol for RFID tag identification," IEEE Communications Letters, vol.14, no.11, Pp.1014-1016, 2010.

[9]. MandeepKaur,ManjeetSandhu,NeerajMohanandParvinderS.Sandhu,"RFID technology Principles, Advantages, Limitations & Its Applications "International Journal of Computer and Electrical Engineering, Vol.3, No.1, February, 2011.

[10]. Prof.GaatriBhandari, Mrinal Bari, ShwetaBorse, AshwiniGaikwad, ReshmaKadam, "Parking Navigation System Based on RFID and IR Sensor", International Journal of Computer Science and Information Technologies, Volume 6,Issue 2, June2015.

[11]. Mr Basavaraju S R," Automatic Smart Parking System using Internet of Things (IOT)", International Journal of Scientific and Research Publications, Volume 5, Issue 12, December 2015.

[12]. NdayambajeMoses1 , Y. D. Chincholkar2," Smart Parking System for Monitoring Vacant Parking", International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 6, June 2016.

[13]. HinaKousar ,Kavitha Kumar, Shoney Sebastian, "Reservation Based Parking System with Dynamic Slot Allocation" International Journal of Scientific and Research Publications, Volume 5, Issue 3, March 2015.

[14]. MohdFairus Bin Abdollah, Syed Najib Bin Syed Salim,"Development Of An Automatic Parallel Parking System for Nonholonomic Mobile Robot", International Conference on Electrical, Control and Computer Engineering, Volume 2, Issue 2, April 2009.

[15]. Viral MV, Choksi V, Potdar MB. Car Safety System Enhancements using the Internet of Things (IoT). International Research Journal of Engineering and Technology (IRJET). 2017; 4(12):1-4.

[16]. Rahman MA, Rashid MM, Farahana N, Musa A, Farhana A. Automatic vehicle Parking Management System and Fee Collection Based on Number Plate Recognition. International Journal of Machine Learning and Computing. 2012; 2(2):1-6.

[17]. Clarke S, Razzaque MA. Middleware for Internet of Things: A Survey. IEEE Internet of Things Journal. 2016; 3(1):70-95.

[18]. Gandhi BMK, Kameswara RM. A Prototype for IoT based Car Parking Management system for Smart Cities. Indian Journal of Science and Technology. 2016; 9(17):1-6.

Knvs Sridevi" Smart Parking System Implementation using Internet of Things (IoT) and Cloud Computing" International Journal of Computational Engineering Research (IJCER), vol. 09, no. 6, 2019, pp 18-29