# Smart Parking System

**Madhumita Manish Kumar, Geetanjali B. Yatnalkar**

**Abstract:** Smart Parking System is based on Internet of Things technology and Android Application which enables the customer to view the available parking spaces and also book the parking space. Therefore, the project aimed at designing a system that will enable the customers to book the parking slot at their convenience. The data for functional, non-functional and system requirements were collected. This data was used to prepare the UML diagrams such as the Entity Relationship Diagram. This system was implemented using different IOT tools like IR sensors and also includes mobile application development for creating interfaces for booking parking slots.

**Keywords:** Arduino, Infrared sensors, Node Mcu.

## I. INTRODUCTION

Smart parking is based on the technology of Internet of Things (IOT) and Android application. The problem of traffic congestion is increasing day by day globally. Parking problem is one of the biggest contributor to this issue and with the increasing luxurious vehicle sizes this situation is getting worse in the urban cities. Hunting for parking spaces is the most hectic work for every driver which has given rise to the requirement for proficient smart parking frameworks which will provide ease and consume less time in searching for parking spaces. Therefore, we have built IOT and android application based framework. To exhibit this idea, we use IR sensors for detecting the vehicles which arrives at the parking slot. We use NODE MCU8266 as wifi module and Arduino UNO as the main unit for receiving and sending data between the mobile application and IR sensors. This nodemcu8266 has an inbuilt Wi-Fi module for web connectivity and other services. We use Firebase as a cloud based server space for maintaining an online database and for handling the hardware signals coming to and from the MCU 8266 and the user application. The IR sensors identifies whether the parking slot is occupied or not, it uses the IR innovation to detect if a vehicle has arrived at the parking space. This framework reads the quantity of parking spaces accessible and refreshes information with the cloud server. Accordingly, this information is also updated in the mobile application and this allows customers to check free parking spaces available from anywhere at anytime. Thence, this system reduces the time consuming parking issue and gives the client a productive IOT based parking system framework.

## II. SYSTEM DESIGN

### A. NodeMcu V1.0(version 2)

Fig.1. NodeMcu esp8266 is a LUA based IoT development board developed by ESP8266 WiFi. One of the unique feature of this is that it has an inbuilt support for WiFi connectivity.

For Programming NodeMCU I used the Node MCU, Arduino and USB cable. Connect the USB cable to the NodeMCU.

Install the Arduino IDE version 1.6.9 once this is installed, use the USB cable to connect the Node MCU board with the computer. Open the Arduino IDE and select the correct board and Port Tools > Boards > NodeMCU1.0 (ESP-12E Module) Tools > Port.

Now load the code onto the IDE and click on the ‘upload’ button given on the top bar. Once code is uploaded the built-in LED starts blinking. The code is successfully uploaded.

### B. Arduino UNO

Fig. 2. Arduino Uno 8-bit ATmega328P microcontroller. Uno has 14 digital input/output pins. By using pinMode(), digitalRead() and digitalWrite() we can program the boards. Serial Pins 0 (Rx) and 1 (Tx) are used to receive and transmit Transistor-Transistor Logic (TTL) serial data.

Arduino IDE (Integrated Development Environment) version 1.6.9 needs to be installed. Once this gets installed on the computer, use the USB cable to connect the board with the computer. Open the arduino IDE and select the correct board and correct Port Tools > Boards > Arduino Uno Tools > Port.

Now load the code and click on the ‘upload’ button given on the top bar. The code is uploaded successfully.
C. IR Sensors

Fig. 3. IR sensor is a device which senses objects which are present in the surrounding by emitting infrared light.

D. Connecting wire’s

Fig. 4. They provide a medium to an electrical current.

III. CIRCUIT DIAGRAM

Fig. 5. Represents the Circuit Diagram of the system and Fig. 6. Represents the actual implemented circuit.

IV. SOFTWARE REQUIREMENTS

1. Arduino IDE
2. Android Studio
3. Firebase

V. DATA DICTIONARY

Table I: Parking System Application

<table>
<thead>
<tr>
<th>Serial No.</th>
<th>Column Name</th>
<th>Data Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>slot_no</td>
<td>int</td>
<td>Primary key</td>
<td>Primary key for the table</td>
</tr>
<tr>
<td>2</td>
<td>Car_plate_no</td>
<td>varchar</td>
<td>Not null</td>
<td>Indicate the vehicle number plate</td>
</tr>
<tr>
<td>3</td>
<td>TimeIn</td>
<td>int</td>
<td>Not null</td>
<td>Time at which the car arrived at parking slot.</td>
</tr>
<tr>
<td>4</td>
<td>TimeOut</td>
<td>int</td>
<td>Not null</td>
<td>Time at which the car left from parking slot.</td>
</tr>
<tr>
<td>5</td>
<td>payment</td>
<td>float</td>
<td>Not null</td>
<td>Indicate the amount to pay</td>
</tr>
<tr>
<td>6</td>
<td>feedback</td>
<td>varchar</td>
<td>Not null</td>
<td>User’s feedback</td>
</tr>
</tbody>
</table>
Table-II: User Registration

<table>
<thead>
<tr>
<th>Serial No.</th>
<th>Column Name</th>
<th>Data type</th>
<th>Constraints</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Char</td>
<td>Not null</td>
<td>The name of the user</td>
</tr>
<tr>
<td>2</td>
<td>Email_id</td>
<td>Nvarchar</td>
<td>Primary key</td>
<td>Primary key for the table</td>
</tr>
<tr>
<td>3</td>
<td>Password</td>
<td>Varchar(10)</td>
<td>Not null</td>
<td>Password of user</td>
</tr>
<tr>
<td>4</td>
<td>Phone_no</td>
<td>Int(10)</td>
<td>Not null</td>
<td>Users contact number</td>
</tr>
<tr>
<td>5</td>
<td>Car_plate_no</td>
<td>Varchar(10)</td>
<td>Not null</td>
<td>User’s car plate no</td>
</tr>
</tbody>
</table>

Table-III: User Login

<table>
<thead>
<tr>
<th>Serial No.</th>
<th>Column Name</th>
<th>Data type</th>
<th>Constraints</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Email_id</td>
<td>Char</td>
<td>Primary key</td>
<td>User’s Email-id</td>
</tr>
<tr>
<td>2</td>
<td>password</td>
<td>varchar</td>
<td>Not null</td>
<td>Password of the user</td>
</tr>
</tbody>
</table>

VI. UML DIAGRAMS

A. Entity Relationship Diagram

Fig. 7. ER Diagram

B. State Transition Diagram

Fig. 8. State Transition Diagram

C. Data Flow Diagram

Fig. 9. DFD Diagram

VII. USER INTERFACE

User Interface (UI) is an important part of any development, because it is what the user sees first no matter how complicated the backend can be. The UI needs to be appealing and easy to use for the user. Taking that into consideration I designed and implemented the interface of the mobile apps.
VIII. PARKING MODEL DESIGN

Fig. 10. Mobile Application Interface

Fig. 11. Parking Slot Model

Fig. 12. Green LED glows when slot is empty

Fig. 13. Yellow LED glows when slot is occupied

Fig. 14. Red LED glows when slot is booked by user through app

IX. TESTS AND RESULTS

Table-I: Testing hardware

<table>
<thead>
<tr>
<th>Test case</th>
<th>Test Data</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Testing IR Sensor</td>
<td>Car is placed near the IR sensor</td>
</tr>
<tr>
<td>2</td>
<td>Testing IR Sensor</td>
<td>Car is placed away from the IR sensor</td>
</tr>
<tr>
<td>3</td>
<td>Testing Red Led</td>
<td>User books the slot from mobile app</td>
</tr>
<tr>
<td>4</td>
<td>Testing Red Led</td>
<td>User cancels booking from mobile app</td>
</tr>
</tbody>
</table>

Expected Results
- The Serial Monitor of Arduino IDE Displays output 1 and Yellow LED glows
- The Serial Monitor of Arduino IDE Displays output 0 and Green LED glows
- The Red Led turns ON
- The Red Led turns OFF

Actual Result
- Serial monitor displays 1 and Yellow Led glows
- Serial monitor displays 0 and Green Led glows
- The Red Led turned ON
- The Red Led turned OFF

Table-II: Testing mobile application
Expected Results
User data is registered successfully and data is stored to firebase
User should be authenticated and enter the homepage
User should be sent a password reset email
Slot is booked and booking data should be sent to firebase
User Booking Details should be displayed
Booking is canceled and booking data should be deleted from firebase
User data should be displayed
Feedback should be sent to firebase
Actual Result
User data is registered successfully and data is stored to firebase
User enters homepage
User is sent a password reset email
Slot is booked and booking data is sent to firebase
User Booking Details should be displayed
Booking is canceled and booking data is deleted from firebase
User data is displayed
Feedback is sent to firebase

The different tests on the different modules of the system were done for enhancements of the system. While testing the hardware, the focus was on time and accuracy. The sensors had no idea if what they are sensing is a car or not. Thus, it was observed that the hardware alone does not ensure accuracy and compatibility. The different tests on the different modules of the system were done for enhancements of the system. While testing the hardware, the focus was on time and accuracy. The sensors had no idea if what they are sensing is a car or not. Thus, it was observed that the hardware alone does not ensure accuracy and compatibility.

The Smart Parking System worked convincingly by connecting the Arduino, NodeMCU, Firebase and mobile application together. All modules of this system were tested and are working successfully, thus proving its accuracy and compatibility. I also examined about the security concerns, since the real-time data along with user details and vehicle details gets stored on the Firebase it can help to keep a track of everything. The driver can easily find a parking slot on the mobile app and book the slot without wasting his/her time in searching a parking. It also helps in preserving the environment, fuel and energy. Emissions are greatly brought down and reduced.

The Smart Parking System worked convincingly by connecting the Arduino, NodeMCU, Firebase and mobile application together. All modules of this system were tested and are working successfully, thus proving its accuracy and compatibility. I also examined about the security concerns, since the real-time data along with user details and vehicle details gets stored on the Firebase it can help to keep a track of everything. The driver can easily find a parking slot on the mobile app and book the slot without wasting his/her time in searching a parking. It also helps in preserving the environment, fuel and energy. Emissions are greatly brought down and reduced.

If the hardware alone does not ensure accuracy and compatibility. The different modules of the application were also tested successfully.

X. CONCLUSION

The Smart Parking System worked convincingly by connecting the Arduino, NodeMCU, Firebase and mobile application together. All modules of this system were tested and are working successfully, thus proving its accuracy and compatibility. I also examined about the security concerns, since the real-time data along with user details and vehicle details gets stored on the Firebase it can help to keep a track of everything. The driver can easily find a parking slot on the mobile app and book the slot without wasting his/her time in searching a parking. It also helps in preserving the environment, fuel and energy. Emissions are greatly brought down and reduced. If the limiting factors such as better components are supported with proper funding, we can further develop it to a fully functional system that has commercial value. High convenience and flexibility along with mobile payment options are projected to increase growth in the commercial segment. Thus an efficient and effective Smart Parking System is successfully designed, implemented and tested.