Smart Car Parking System

***Mini*** ***Project*** ***Report*** ***submitted*** ***in*** ***partial*** ***fulfillment.***

***of*** ***the*** ***requirement*** ***for*** ***the*** ***degree*** ***of***

**B.** **E.** **(Information** **Technology)**

Submitted By

**Siddhesh Shivgan (18101A2007)**

**Aakash Manjrekar (17101A0071)**

**Atharvan Chavan (18101A2001)**

Under the Guidance of

Prof. Vinita Bhandiwad

Department of Information Technology





Vidyalankar Institute of Technology

Wadala(E), Mumbai 400 037

University of Mumbai

2020-21

CERTIFICATE OF APPROVAL

**For**

**Mini** **Project** **Report**

**On**

**Internet of Everything Lab**

This is to Certify that

**Siddhesh Shivgan (18101A2007)**

**Aakash Manjrekar (17101A0071)**

**Atharvan Chavan (18101A2001)**

Have successfully carried out Mini Project entitled

“**Smart Car Parking System**”

In partial fulfillment of degree course in

Information Technology

As laid down by University of Mumbai during the academic year 2020-21

Under the Guidance of

“Prof. Vinita Bhandiwad”

Signature of Guide Head of Department

Examiner 1 Examiner 2 Principal

Dr. S. A. Patekar

**ACKNOWLEGEMENT**

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. We express our profound gratitude we give to our **Prof. Vinita Bhandiwad** Ma’am, our respectable project guide, for her gigantic support and guidance. Without her counseling our project would not have seen the light of the day.

We extend our sincere thanks to **Dr. Deepali Vora**, Head of the Department of

Information Technology for offering valuable advice at every stage of this undertaking.

We would like to thank all the staff members who willingly helped us. We are grateful to VIDYALANKAR INSTITUTE OF TECHNOLOGY for giving us this opportunity.

The days we have spent in the institute will always be remembered and also be

reckoned as guiding in our career.

1. **Siddhesh Shivgan**
2. **Aakash Manjrekar**

**3. Atharvan Chavan**

**Table of Contents**

| Sr. No. | Topic | Page No. |
| --- | --- | --- |
| 1 | Introduction | 6 |
| 2 | Aim & Objects | 7 |
| 3 | Problem Definition | 8 |
| 4 | Proposed System  4.1 Block Diagram  4.2 Flow Chart | 9 |
| 5 | Components  5.1 Hardware 5.2 Software | 11 |
| 6 | Logic | 13 |
| 7 | Code | 14 |
| 8 | Implementation  8.1 Working  8.2 Circuit Diagram | 18 |
| 9 | Deployment of Testing | 20 |
| 10 | Conclusion & Future Scope | 21 |
|  | IEEE technical paper |  |
|  | Technical poster on Project topic |  |
|  | Link of the project video |  |

**ABSTRACT**

The main objective is to avoid the cramming in the car parking area by implementing an efficient car parking system along with a user-friendly application for an ease of use. Normally at public places such as multiplex theatres, market areas, hospitals, function-halls, offices and shopping malls, one experiences the discomfort in looking out for a vacant parking slot, though it’s a paid facility with an attendant/ security guard. The parking management system is proposed to demonstrate hazel free parking. The proposed system consist of 4 IR Sensors , Arduino UNO , NodeMCU ESP8266 Module , RFID Reader and Tags , Buzzer. Implementation involves minimal human interaction and provides a seamless parking experience thereby reducing a lot of time wasted by the user in parking his/her vehicle.

**1.INTRODUCTION**

Nowadays in many public places such as malls, multiplex systems, hospitals, offices, market areas there is a crucial problem of car parking. The car-parking [1-4] area has many lanes/slots for car parking. So to park a car one has to look for all the lanes. Moreover, this involves a lot of manual labor and investment. So, there is a need to develop an automated parking system that indicates directly the availability of vacant parking slots in any lane right at the entrance.

It involves a system including IR Sensors- receiver pair in each lane .So the person desirous to park his vehicle is well informed about the status of availability of parking slot. Conventional parking systems do not have any intelligent monitoring system and the parking lots are monitored by security guards.

A lot of time is wasted in searching vacant slot for parking and many a times it creates jams. Conditions become worse when there are multiple parking lanes and each lane with multiple parking slots. Use of parking management system would reduce the human efforts and time with additional comfort. In the proposed system, the display unit displays a visual representation of the parking and it shows the empty and occupied slots which help the user to decide where to park their car. The system would not only save time but the software and hardware would also manage the Check-in and check-outs of the cars under the control of RFID readers/ tags, Entry exit data logging.

**2.AIM & OBJECTS**

The main objective is to avoid the cramming in the car parking area by implementing an efficient car parking system along with a user-friendly application for an ease of use.

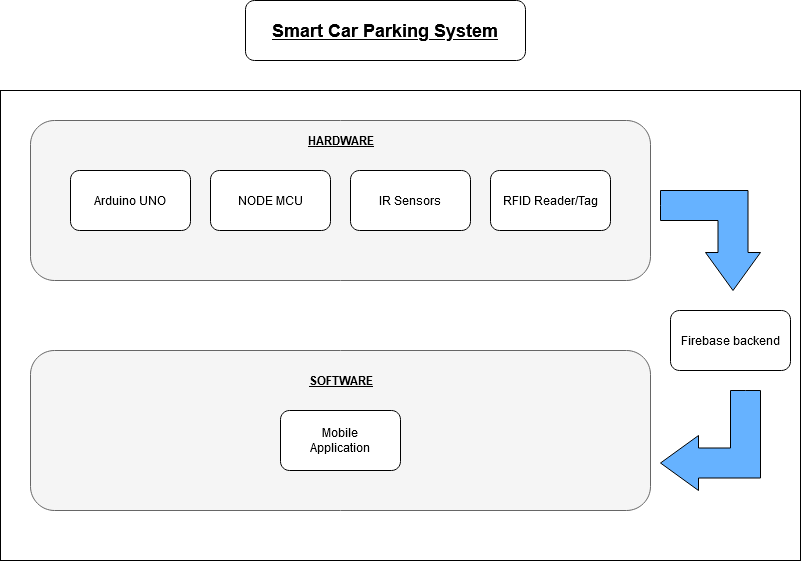
Now days in many public places such as malls, multiplex systems, hospitals, offices, market areas there is a crucial problem of car parking. The car-parking area has many lanes/slots for car parking. So to park a car one has to look for all the lanes. Moreover, this involves a lot of manual labor and investment. So, there is a need to develop an automated parking system that indicates directly the availability of vacant parking slots in any lane right at the entrance.

**3.PROBLEM DEFINITION**

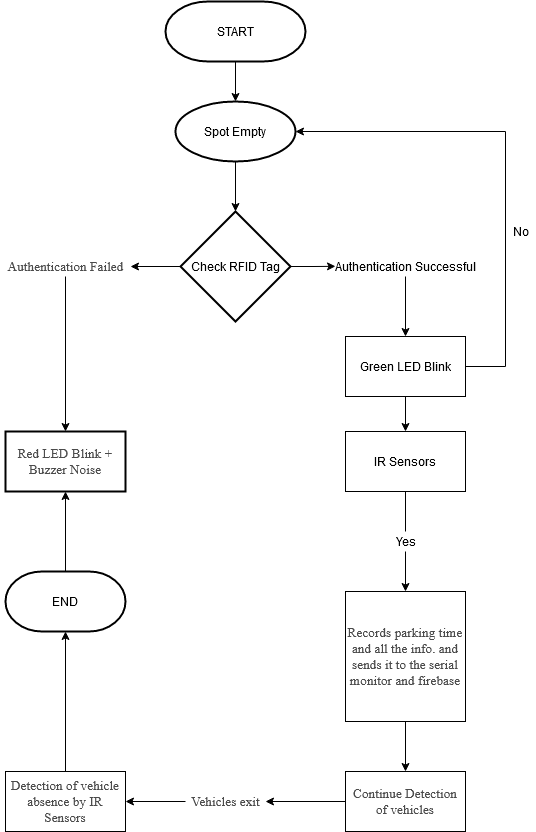
Nowadays in many public places such as malls, multiplex systems, hospitals, offices, market areas there is a crucial problem of car parking. So to park a car one has to look for all the lanes. Moreover, this involves a lot of manual labor and investment.

**4.PROPOSED SYSTEM**

**4.1 Block Diagram :**



**4.2 Flow Chart :**



**COMPONENTS**

**5.1 Hardware :**

* **Arduino UNO R3.**
* **NodeMCU ESP8266.**
* **RFID RC522 Reader / Tags.**
* **Breadboard.**
* **Jumper Wires.**
* **4 IR Sensors.**
* **2 LED Lights (Red and Green).**
* **Buzzer.**

**5.2 Software :**

* **Arduino IDE.**
* **Flutters Frameworks.**
* **Dart.**
* **Firebase.**
* **IBM Watson IOT platform.**

**LOGIC**

Step 1: Check RFID Card / Tag.

Step 2: Give permission to enter on the basis of RFID authentication.

Step 3: Check for empty parking slots.

Step 4: Detect new vehicle entry.

Step 5: Count the parking time.

Step 6: Send this data to firebase.

Step 7: Display real time data on application.

**CODE**

**RFID Reader / Tags Code using Arduino UNO:**

#include <SPI.h>

#include <MFRC522.h>

#define RST\_PIN 9

#define SS\_PIN 10

String myrfidcard = "";

MFRC522 mfrc522(SS\_PIN, RST\_PIN); // Create MFRC522 instance.

// Number of known default keys (hard-coded)

#define NR\_KNOWN\_KEYS 8

byte knownKeys[NR\_KNOWN\_KEYS][MFRC522::MF\_KEY\_SIZE] = {

{0xff, 0xff, 0xff, 0xff, 0xff, 0xff}, // FF FF FF FF FF FF = factory default

{0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5}, // A0 A1 A2 A3 A4 A5

{0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5}, // B0 B1 B2 B3 B4 B5

{0x4d, 0x3a, 0x99, 0xc3, 0x51, 0xdd}, // 4D 3A 99 C3 51 DD

{0x1a, 0x98, 0x2c, 0x7e, 0x45, 0x9a}, // 1A 98 2C 7E 45 9A

{0xd3, 0xf7, 0xd3, 0xf7, 0xd3, 0xf7}, // D3 F7 D3 F7 D3 F7

{0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff}, // AA BB CC DD EE FF

{0x00, 0x00, 0x00, 0x00, 0x00, 0x00} // 00 00 00 00 00 00

};

int buzzer = 5;

void setup() {

Serial.begin(9600);

pinMode(7,OUTPUT);

pinMode(8,OUTPUT);

while (!Serial);

SPI.begin(); // Init SPI bus

mfrc522.PCD\_Init(); // Init MFRC522 card

}

void dump\_byte\_array(byte \*buffer, byte bufferSize) {

for (byte i = 0; i < bufferSize; i++) {

Serial.print(buffer[i] < 0x10 ? " 0" : " ");

Serial.print(buffer[i], HEX);

myrfidcard += buffer[i];

}

}

bool try\_key(MFRC522::MIFARE\_Key \*key)

{

bool result = false;

byte buffer[18];

byte block = 0;

MFRC522::StatusCode status;

status = mfrc522.PCD\_Authenticate(MFRC522::PICC\_CMD\_MF\_AUTH\_KEY\_A, block, key, &(mfrc522.uid));

if (status != MFRC522::STATUS\_OK) {

// Serial.print(F("PCD\_Authenticate() failed: "));

// Serial.println(mfrc522.GetStatusCodeName(status));

return false;

}

// Read block

byte byteCount = sizeof(buffer);

status = mfrc522.MIFARE\_Read(block, buffer, &byteCount);

if (status != MFRC522::STATUS\_OK) {

// Serial.print(F("MIFARE\_Read() failed: "));

// Serial.println(mfrc522.GetStatusCodeName(status));

}

else {

// Successful read

result = true;

Serial.print(F("Success with key:"));

dump\_byte\_array((\*key).keyByte, MFRC522::MF\_KEY\_SIZE);

Serial.println();

// Dump block data

Serial.print(F("Block ")); Serial.print(block); Serial.print(F(":"));

dump\_byte\_array(buffer, 16);

Serial.println();

}

Serial.println();

mfrc522.PICC\_HaltA(); // Halt PICC

mfrc522.PCD\_StopCrypto1(); // Stop encryption on PCD

return result;

}

void loop(){

if (mfrc522.PICC\_IsNewCardPresent()){

// Select one of the cards

if (mfrc522.PICC\_ReadCardSerial()){

// Show some details of the PICC (that is: the tag/card)

Serial.print(F("Card UID:"));

dump\_byte\_array(mfrc522.uid.uidByte, mfrc522.uid.size);

if(myrfidcard == "9961812")

{

digitalWrite(7,HIGH);

Serial.print(" Card authentication successfull");

delay(3000);

digitalWrite(7,LOW);

}

if(myrfidcard != "9961812"){

digitalWrite(8,HIGH);

tone(buzzer,450);

Serial.print(" Card authentication failed");

delay(3000);

noTone(buzzer);

digitalWrite(8,LOW);

}

Serial.println();

}

}

myrfidcard = "";

}

**NodeMCU Code:**

#include <ESP8266WiFi.h>

#include <FirebaseArduino.h>

// Set these to run example.

#define FIREBASE\_HOST "smart-car-parking-7df03-default-rtdb.firebaseio.com"

#define FIREBASE\_AUTH "SMX1gHNByVqUyWwwT53mvV9RHWlx2UgjynyMCWvd"

#define WIFI\_SSID "iot"

#define WIFI\_PASSWORD "intel2365"

int inputPin1 = 4;

int inputPin2 = 5;

int inputPin3 = 13;

int inputPin4 = 12;

int led = 16;

int val1 = 0;

int val2 = 0;

int val3 = 0;

int val4 = 0;

unsigned long StartTime1 = 0;

unsigned long StartTime2 = 0;

unsigned long StartTime3 = 0;

unsigned long StartTime4 = 0;

unsigned long CurrentTime1 = 0;

unsigned long CurrentTime2 = 0;

unsigned long CurrentTime3 = 0;

unsigned long CurrentTime4 = 0;

unsigned long ElapsedTime1 = 0;

unsigned long ElapsedTime2 = 0;

unsigned long ElapsedTime3 = 0;

unsigned long ElapsedTime4 = 0;

void setup() {

Serial.begin(115200);

Serial.println();

pinMode(inputPin1,INPUT);

pinMode(inputPin2,INPUT);

pinMode(inputPin3,INPUT);

pinMode(inputPin4,INPUT);

pinMode(led,OUTPUT);

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

Serial.print("connecting");

while (WiFi.status() != WL\_CONNECTED) {

Serial.print(".");

delay(500);

}

Serial.println();

Serial.print("connected: ");

Serial.println(WiFi.localIP());

Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH);

}

void loop() {

val1 = digitalRead(inputPin1); // read input1 value

val2 = digitalRead(inputPin2); // read input2 value

val3 = digitalRead(inputPin3); // read input3 value

val4 = digitalRead(inputPin4); // read input4 value

if (val1 == HIGH)

{

// check if the input is HIGH

StartTime1 = millis();

digitalWrite(led, LOW); // turn LED OFF

}

if (val1 == LOW)

{

CurrentTime1 = millis();

ElapsedTime1 = CurrentTime1 - StartTime1;

Serial.print("Slot1 Elapsed Time(s): ");

Serial.println(ElapsedTime1/1000);

Firebase.setFloat("Slot1 Elapsed Time(s): ", ElapsedTime1/1000);

// handle error

if (Firebase.failed()) {

Serial.print("setting /number failed:");

Serial.println(Firebase.error());

return;

}

delay(1000);

digitalWrite(led, HIGH); // turn LED ON

}

if (val2 == HIGH)

{

// check if the input is HIGH

StartTime2 = millis();

digitalWrite(led, LOW); // turn LED OFF

}

if (val2 == LOW)

{

CurrentTime2 = millis();

ElapsedTime2 = CurrentTime2 - StartTime2;

Serial.print("Slot2 Elapsed Time(s): ");

Serial.println(ElapsedTime2/1000);

Firebase.setFloat("Slot2 Elapsed Time(s): ", ElapsedTime2/1000);

// handle error

if (Firebase.failed()) {

Serial.print("setting /number failed:");

Serial.println(Firebase.error());

return;

}

delay(1000);

digitalWrite(led, HIGH); // turn LED ON

}

if (val3 == HIGH)

{

// check if the input is HIGH

StartTime3 = millis();

digitalWrite(led, LOW); // turn LED OFF

}

if (val3 == LOW)

{

CurrentTime3 = millis();

ElapsedTime3 = CurrentTime3 - StartTime3;

Serial.print("Slot3 Elapsed Time(s): ");

Serial.println(ElapsedTime3/1000);

if(ElapsedTime3 ==0){

Serial.print("Slot3 Elapsed Time(s): ");

Serial.println(val);

}

val = ElapsedTime3;

Firebase.setFloat("Slot3 Elapsed Time(s): ", ElapsedTime3/1000);

// handle error

if (Firebase.failed()) {

Serial.print("setting /number failed:");

Serial.println(Firebase.error());

return;

}

delay(1000);

digitalWrite(led, HIGH); // turn LED ON

}if (val4 == HIGH)

{

// check if the input is HIGH

StartTime4 = millis();

digitalWrite(led, LOW); // turn LED OFF

}

if (val4 == LOW)

{

CurrentTime4 = millis();

ElapsedTime4 = CurrentTime4 - StartTime4;

Serial.print("Slot4 Elapsed Time(s): ");

Serial.println(ElapsedTime4/1000);

Firebase.setFloat("Slot4 Elapsed Time(s): ", ElapsedTime4/1000);

// handle error

if (Firebase.failed()) {

Serial.print("setting /number failed:");

Serial.println(Firebase.error());

return;

}

delay(1000);

digitalWrite(led, HIGH); // turn LED ON

}

}

**IMPLEMENTATION**

**8.1 Working:**

It proposes a prototype of Smart Car Parking System. There will be an RFID Tag with every user. As soon as the vehicle passes through the entrance the user puts RFID tag over the RFID Reader and gets the Unique ID and then gets entry inside the parking lot.The sensor used in this project is an IR sensors which determines whether the slot is occupied or unoccupied. These sensors are connected to the NodeMCU ESP8266. The output of these sensors is sent to the database through the NodeMCU ESP8266. Each second Firebase database is updated with latest values from IR Sensors. This result is displayed using the Mobile Application.

**8.2 Circuit Diagram:**

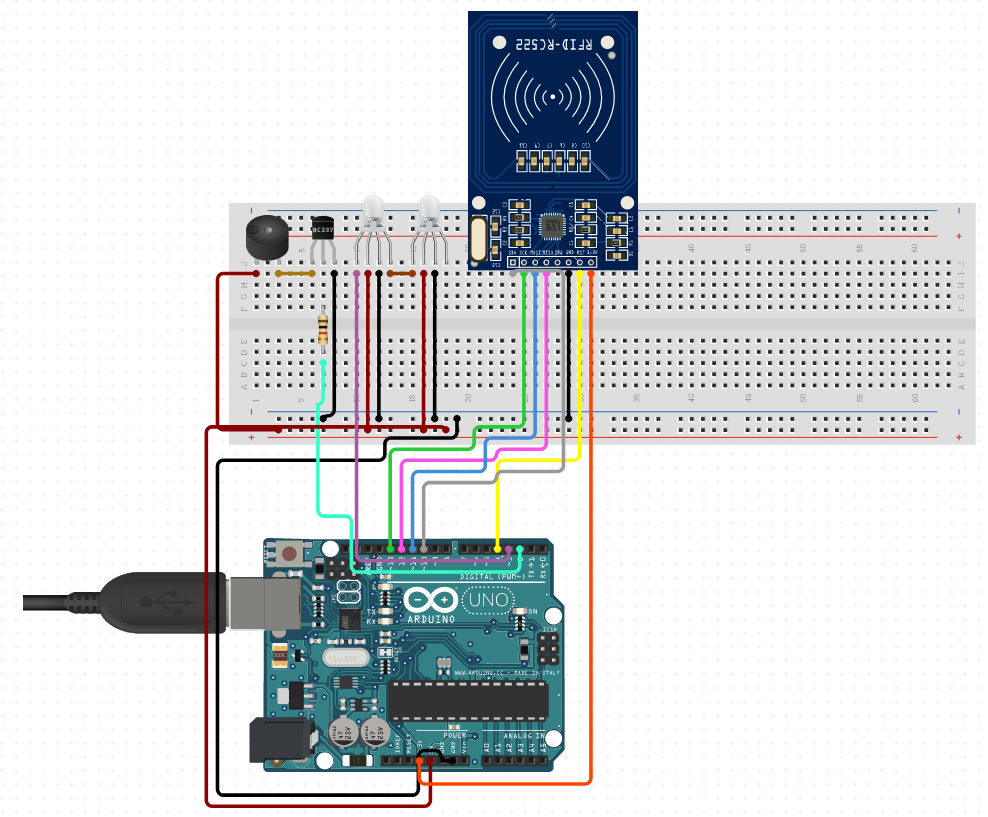


Fig: RFID Module

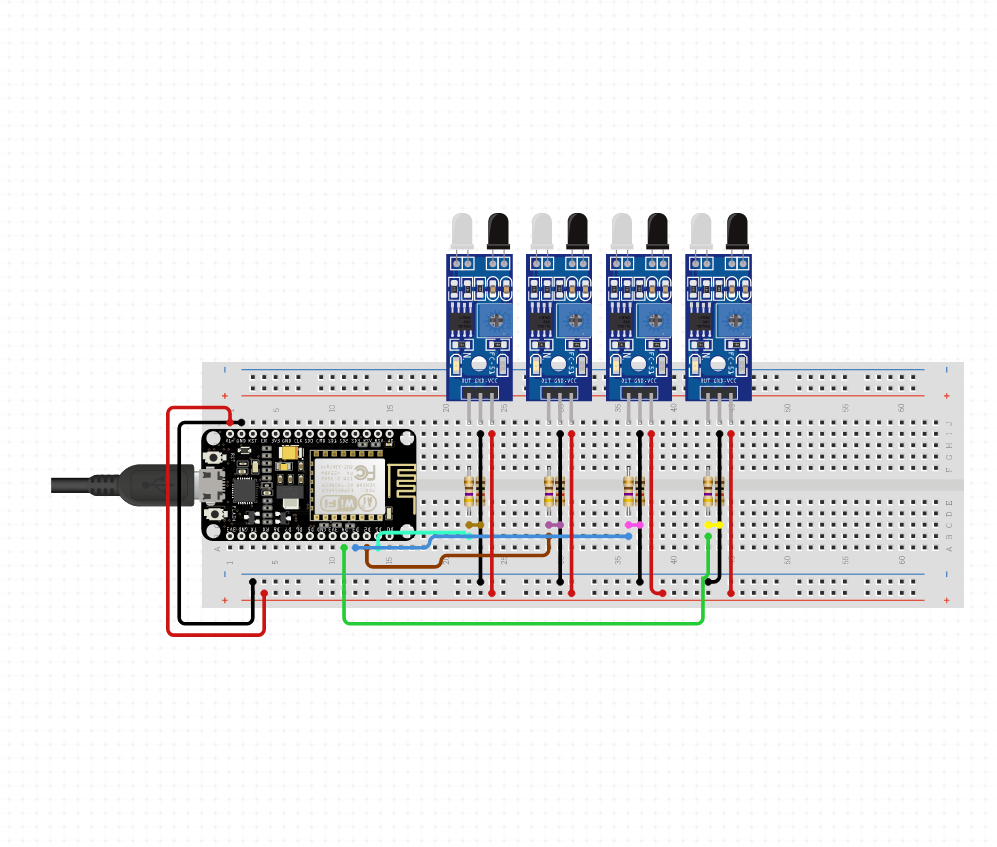
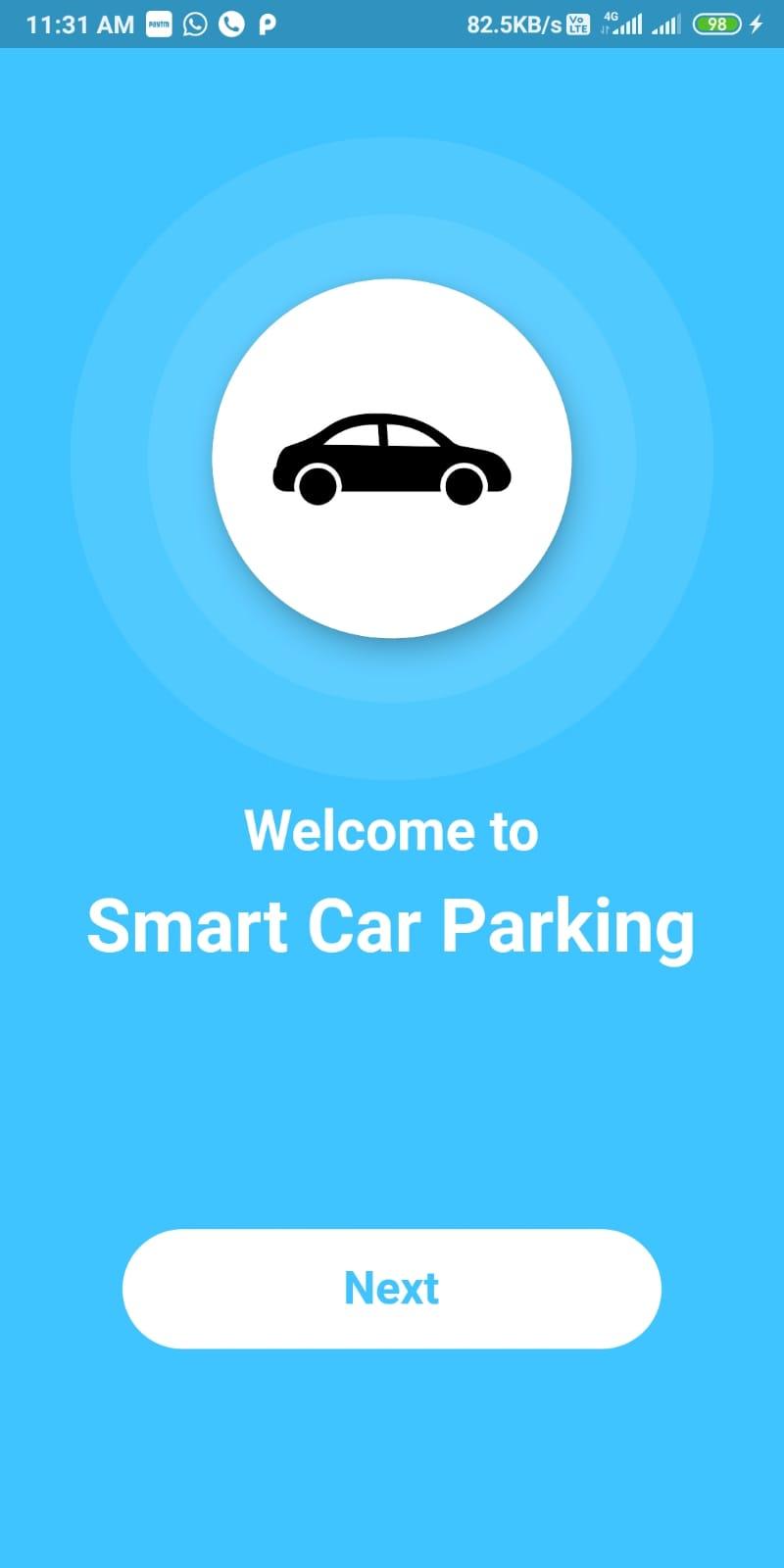
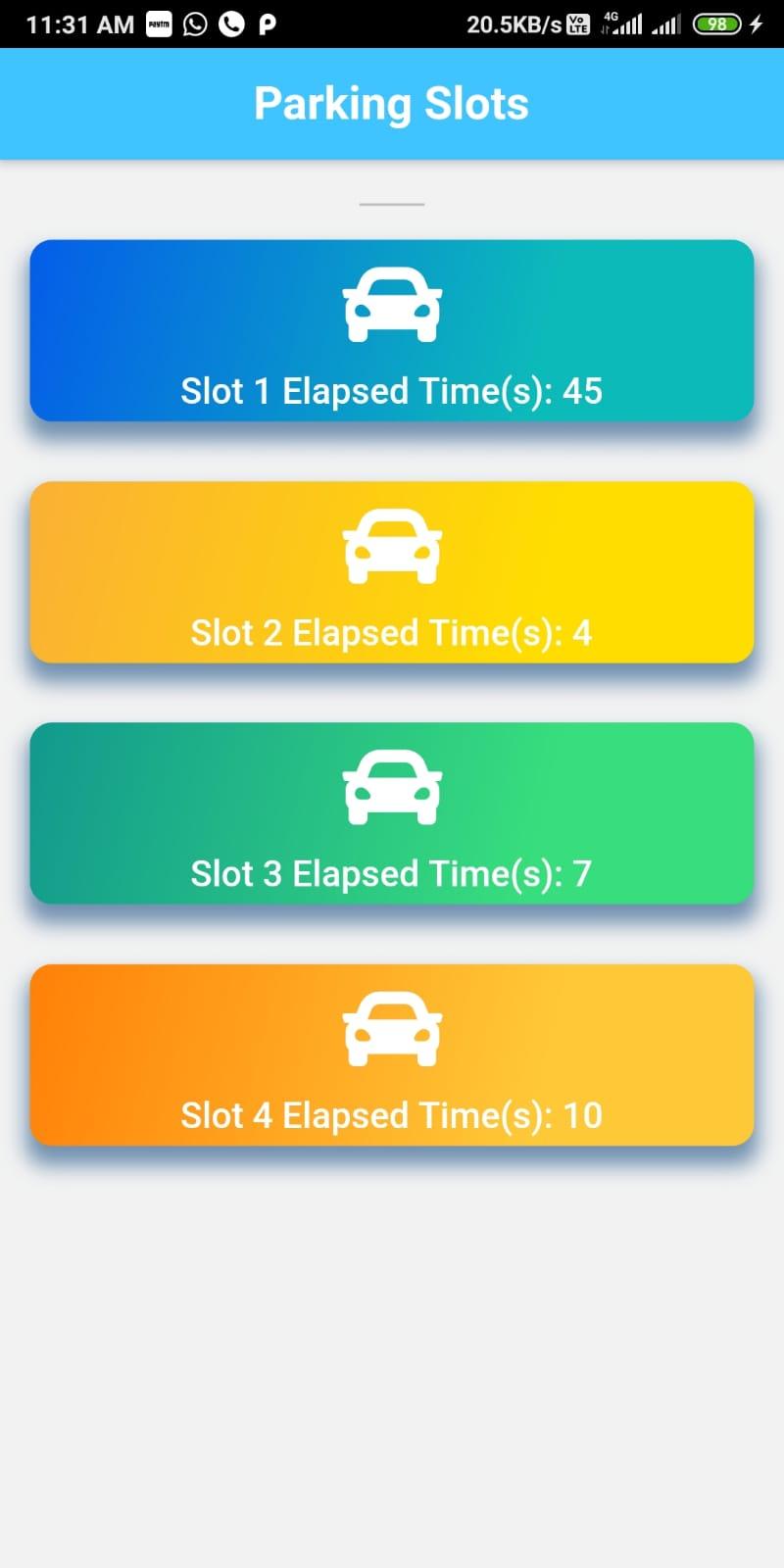


Fig: IR Sensor Module

**DEPLOYMENT OF TESTING** 

**CONCLUSION & FUTURE SCOPE**

The main aim is to design an integrated system which involves two components namely Parking Allocation and Seamless Parking. The Parking Allocation component consists of sensors in front each slot and when a vehicle enters into the slot, the database is updated and the changes are reflected immediately on the Mobile Application. The seamless parking component consists of a RFID Tag with user. It will save the time of human intervention and saving an ample amount of time. We have interfaced 4 IR Sensors and an RFID Reader module (RC-522) using a Arduino UNO and NodeMCU. The IR senses the presence of a vehicle in the parking slot and updates the database. We have made an android application.

We are working on some UI improvements and managing database properly.

**IEEE Paper Link:**

<https://www.ripublication.com/ijaer17/ijaerv12n17_35.pdf>