# Project Report

## IoT Device Essentials

**Title :** Smart Parking System

**Project ID :** SPS-170-Smart Parking System

**Micro Project at SmartInternz**

**By,**
**Aniket Bordekar**
(aniket.bordekar123@gmail.com)

## Table of Contents

# 1.Introduction:

## 1.1 Overview:

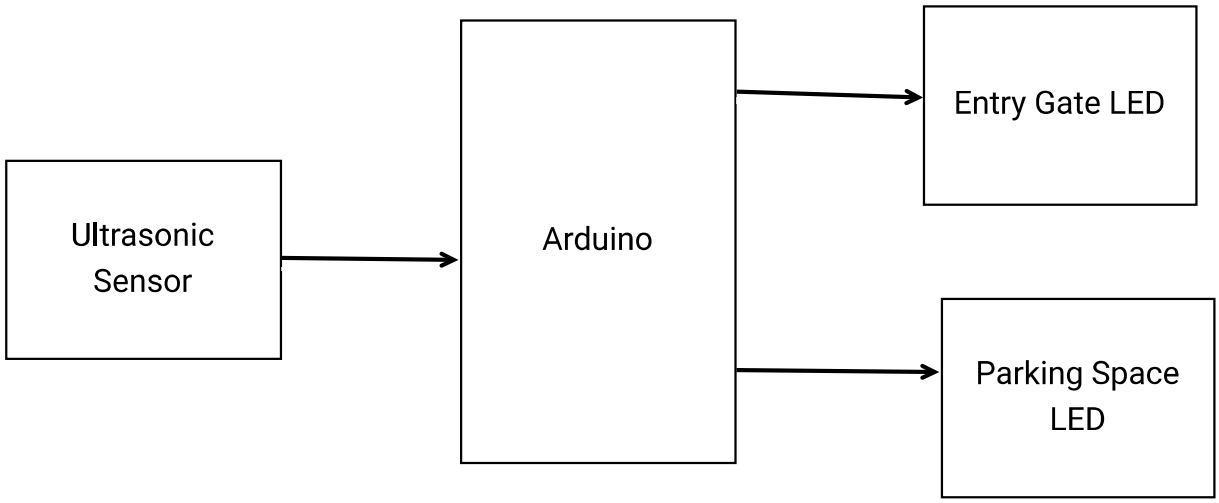
In this project a new parking system called Smart Parking System (SPS) is proposed to assist drivers to find vacant spaces in a car parking in a shorter time and easily. The new system uses ultrasonic sensors to detect either car park occupancy or improper parking actions.

## 1.2 Purpose:

Due to increase in the number of vehicles on road, traffic problems are bound to exist. This is due to the fact that the current transportation infrastructure and car park facility developed are unable to cope with the influx of vehicles on the road. To alleviate the aforementioned problems, the smart parking system has been developed. With the implementation of the smart parking system, patrons can easily locate and secure a vacant parking space at any car park deemed convenient to them.
Features of SPS include vacant parking space detection through the use of specific LEDs.

# 2.Theoritical Analysis

## 2.1 Block Diagram




## 2.2 Hardware / Software Designing:

1. Tinkercad Software.
2. Arduino UNO.
3. Ultrasonic Sensor.
4. RGB LED.

# 3.Experimental Investigations:

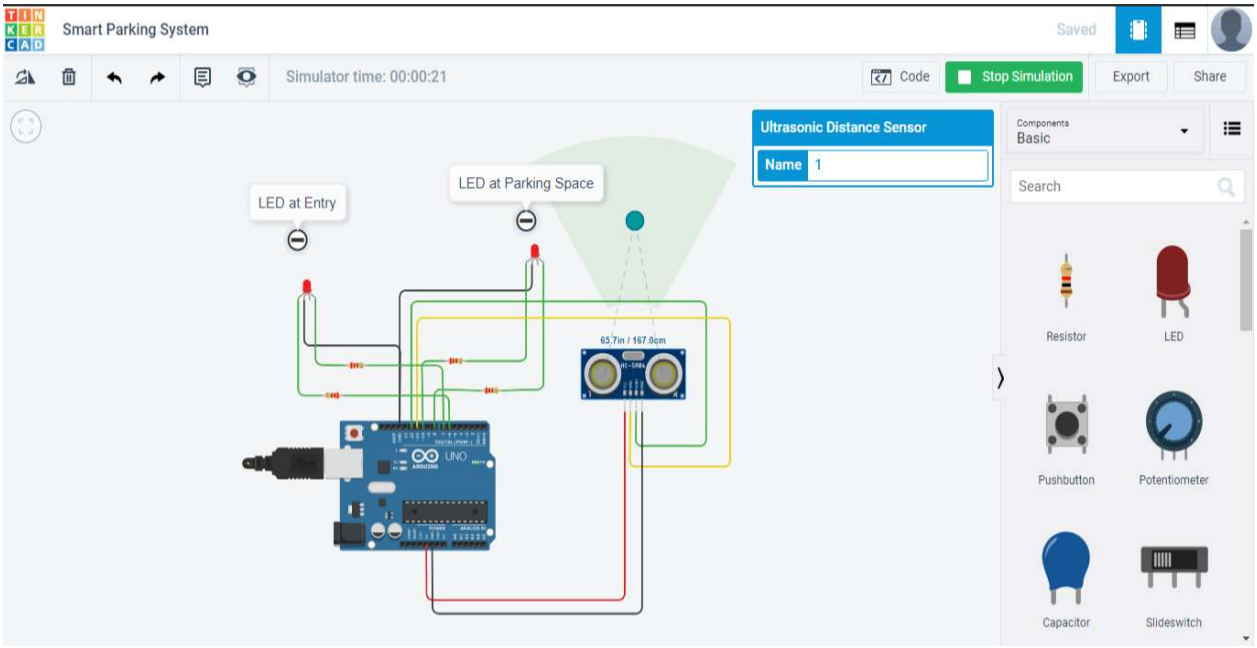



fig. 1

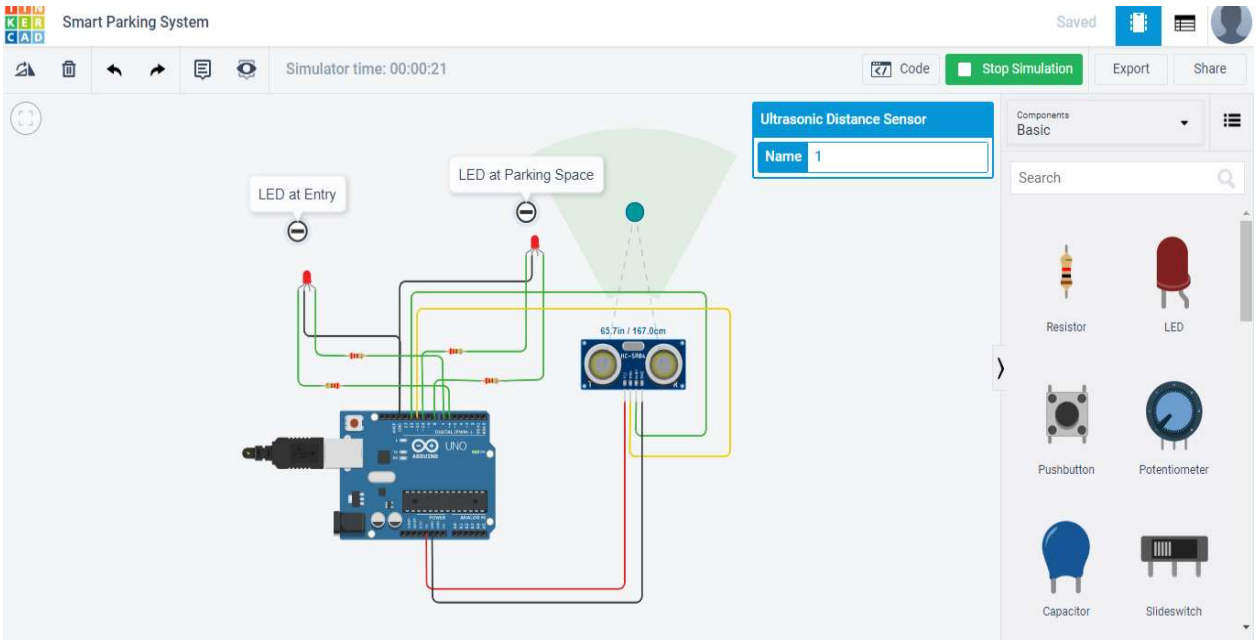
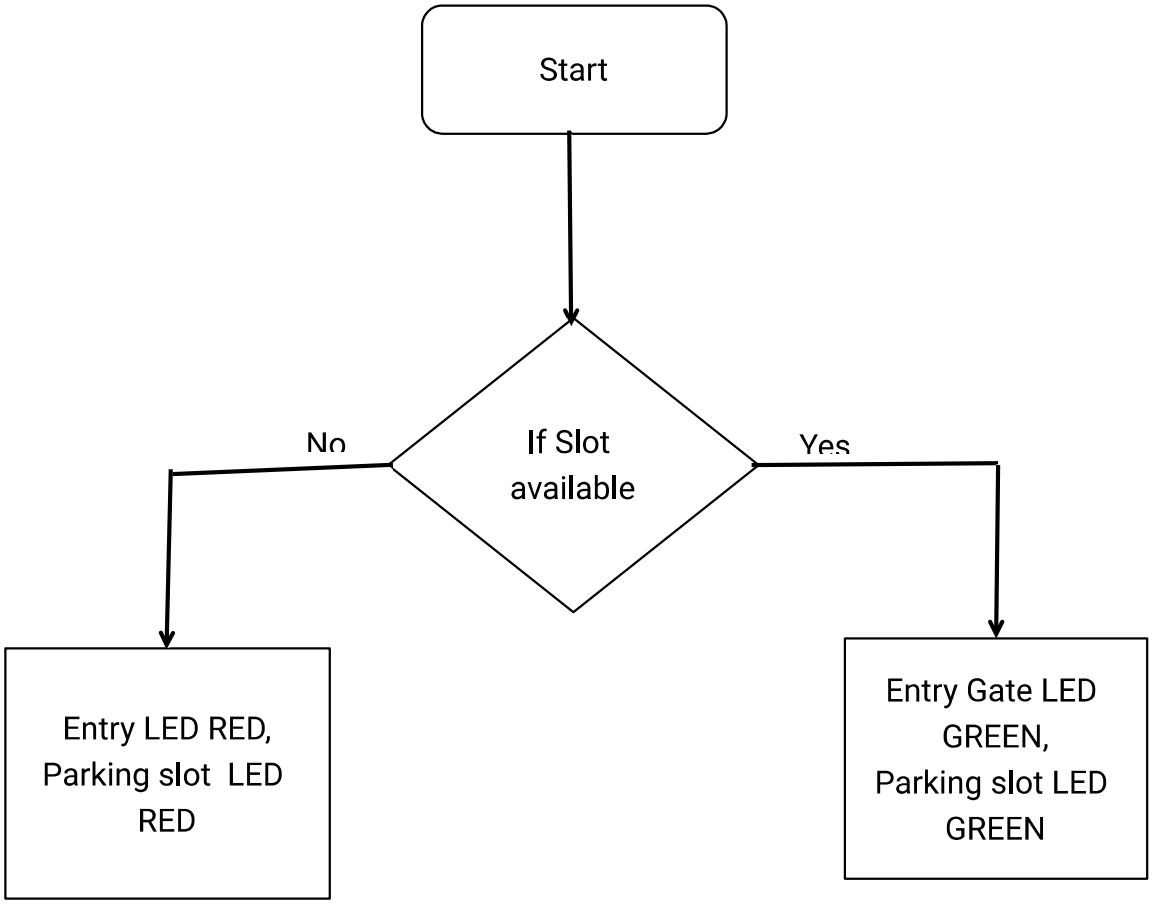



fig. 2

# 4. Flowchart:


Start
If Slot available
No
Yes
Entry LED RED, Parking slot LED RED
Entry Gate LED GREEN, Parking slot LED GREEN

# 5. Result:

We have successfully buld a Smart Parking System device which will indicate weather the parking is available or not.

# 6. Advantages and Dis-advantages:

## 6.1 Advantages :

- Easy indication for availability of parking space
- Less chances for vehicle vandalism.
- Emissions are greatly brought down and reduced.
- There is a minimal staff requirement.
- Can be used in a dark parking lot, facilitating parking in a smooth manner.

## 6.2 Dis-advantages:

- There is a greater construction cost per space.
- It may be a bit confusing for unfamiliar users.
- It requires a maintenance contract with the supplier
- Any obstacle that is present along the sides of the car will go unnoticed.

# 7. Applications:

- Easily indicates about new parking areas available.
- Can be used in Public places like shopping malls, theaters, restaurants, etc.
- Private parking spaces for safe parking.
- Reduces traffic by detecting empty spaces.

# 8. Conclusion:

A model for smart parking system has been created on Tinkercad, using Arduino.

# 9. Future Scope:

A web Application can be created for getting the data on mobile like devices for reducing traffics in public places, roads and more also detecting the parking availability at different locations using GPS.

# 10. Bibliography:


- Tinkercad :

  https://www.tinkercad.com/

- Arduino Uno :

  https://store.arduino.cc/usa/arduino-uno-rev3

- Ultrasonic sensor :

  https://randomnerdtutorials.com/courses/

# 11. Appendix:

## Source Code :

```
int trigPin = 11;
int echoPin = 12;
int redPin = 10;
//int bluePin = 9;
int greenPin = 8;
int entrance_green = 7;
int entrance_red = 6;
long duration, cm;

void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(redPin, OUTPUT);
  //pinMode(bluePin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(entrance_green, OUTPUT);
  pinMode(entrance_red, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  digitalWrite(trigPin, LOW);
  delay(20);
  digitalWrite(trigPin, HIGH);
  delay(1000);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  cm = duration/58.2;  //distance = (traveltime/2) x speed of sound
                                                    //The speed of sound is: 343m/s =
0.0343 cm/uS = 1/29.1 cm/uS
```

```
  if (cm<=270)
  {
    RG_light(255, 0);
    RG_light_entrance(255, 0);
  }
  else if (cm>270 && cm<=318)
  {
    RG_light(0, 255);
    RG_light_entrance(255, 0);
  }
  else
  {
    RG_light(0, 0);
    RG_light_entrance(0, 255);
  }
}

void RG_light(int red_value, int green_value)
{

  analogWrite(redPin, red_value);
  analogWrite(greenPin, green_value);
}

void RG_light_entrance(int Ered_value, int Egreen_value)
{
  analogWrite(entrance_red, Ered_value);
  analogWrite(entrance_green, Egreen_value);
}
```