Team Members:

R.Deepika Sri-921021104006

T. Muthuvaisali-921021104030

L.Luckymalathi-921021104023

M.Sowmiya-921021104047

**Smart Parking System Using IoT**

**Introduction**:

Smart parking systems are becoming increasingly popular as a way to address the challenges of urban congestion and parking scarcity. By using sensors and IoT connectivity, smart parking systems can provide real-time information on parking availability, guide drivers to open spots, and even automate the payment process.

This will build a simple smart parking system using an Arduino or ESP8266 microcontroller, ultrasonic sensors, and a cloud platform like Firebase. The system will detect the presence of vehicles in parking slots and send this information to the cloud in real time. Users can then access this information through a web or mobile app to see which parking slots are available and get directions to them.

**Components:**

* Arduino or ESP8266 microcontroller
* Ultrasonic sensors (one for each parking slot)
* LED indicator lights (one for each parking slot, optional)
* Breadboard and jumper wires
* Power supply
* Cloud platform

**Hardware Setup:**

* Connect the ultrasonic sensors to the microcontroller according to the sensor’s datasheet.
* Connect the LED indicator lights to the microcontroller (optional).
* Connect the microcontroller to the power supply.

**Software Setup:**

* Install the necessary libraries for the microcontroller and cloud platform.
* Write a program to read the ultrasonic sensors and send the data to the cloud.
* Create a web or mobile app to display the parking availability information from the cloud.

**Coding:**

The following code shows an example of how to read an ultrasonic sensor and send the data to Firebase using an Arduino microcontroller:

#include <FirebaseArduino.h>

#include <NewPing.h>

// Define the ultrasonic sensor pins

#define TRIG\_PIN 12

#define ECHO\_PIN 11

// Define the Firebase project credentials

#define FIREBASE\_HOST https://<your-project-id>.firebaseio.com/

#define FIREBASE\_AUTH “<your-firebase-auth-secret>”

// Create a NewPing object to measure the distance to the nearest object

NewPing sonar(TRIG\_PIN, ECHO\_PIN, 400);

// Create a Firebase object

Firebase firbase(FIREBASE\_HOST, FIREBASE\_AUTH);

Void setup() {

// Initialize the serial port

Serial.begin(9600);

// Initialize Firebase

Firebase.begin();

}

Void loop() {

// Measure the distance to the nearest object

Unsigned int distance = sonar.ping();

// If the distance is less than 1 meter, the parking spot is occupied

If (distance < 100) {

Serial.println(“Parking spot occupied”);

// Update the Firebase database to indicate that the parking spot is occupied

Firebase.update(“parking/spot1”, “occupied”);

} else {

Serial.println(“Parking spot available”);

// Update the Firebase database to indicate that the parking spot is available

Firebase.update(“parking/spot1”, “available”);

}

// Wait 1 second before measuring the distance again

Delay(1000);

}

This code will measure the distance to the nearest object every second and update the Firebase database to indicate whether the parking spot is occupied or available.

**Web or Mobile App:**

Once the hardware and software are set up, you can create a web or mobile app to display the parking availability information from the cloud. The app can use the Firebase REST API to retrieve the latest parking data.

The following code shows an example of how to retrieve the parking availability information using a Python script:

Import requests

# Define the Firebase project credentials

FIREBASE\_HOST = https://<your-project-id>.firebaseio.com/

# Make a GET request to the Firebase database to retrieve the parking availability information

Response = requests.get(FIREBASE\_HOST + “/parking.json”)

# Parse the JSON response to get the parking availability data

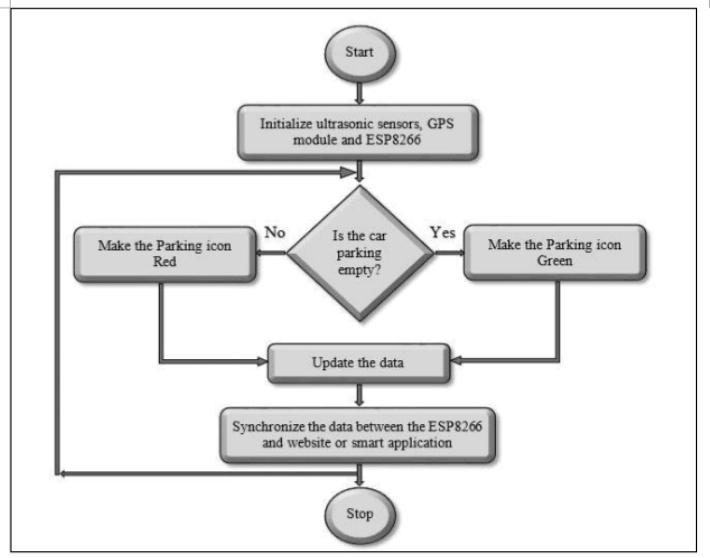
Parking\_availability = response.json()

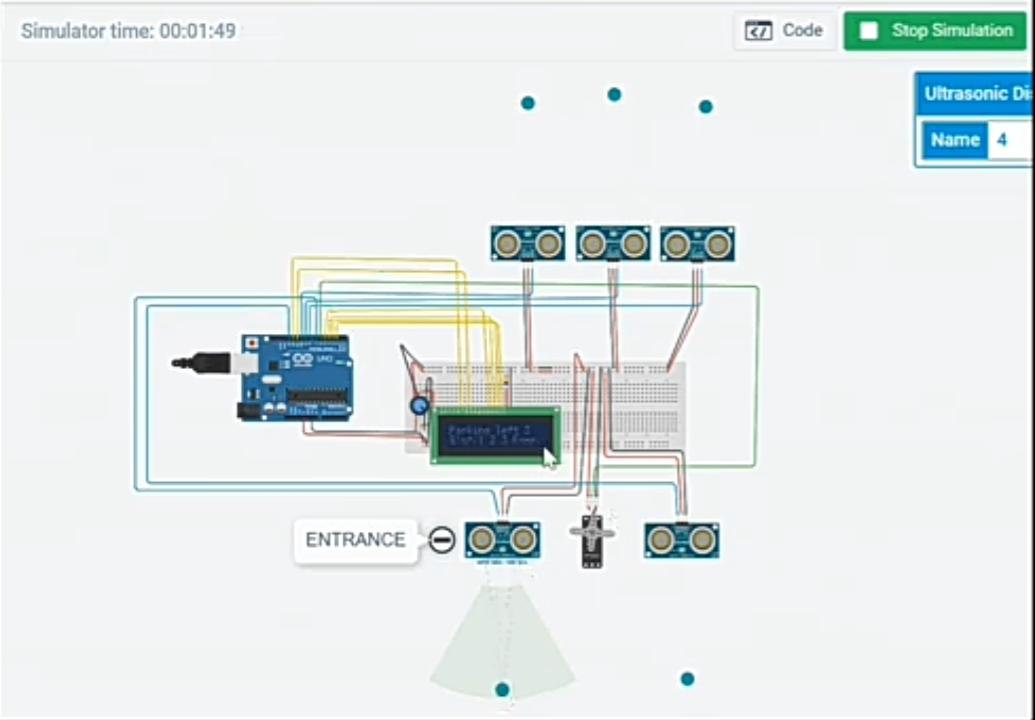
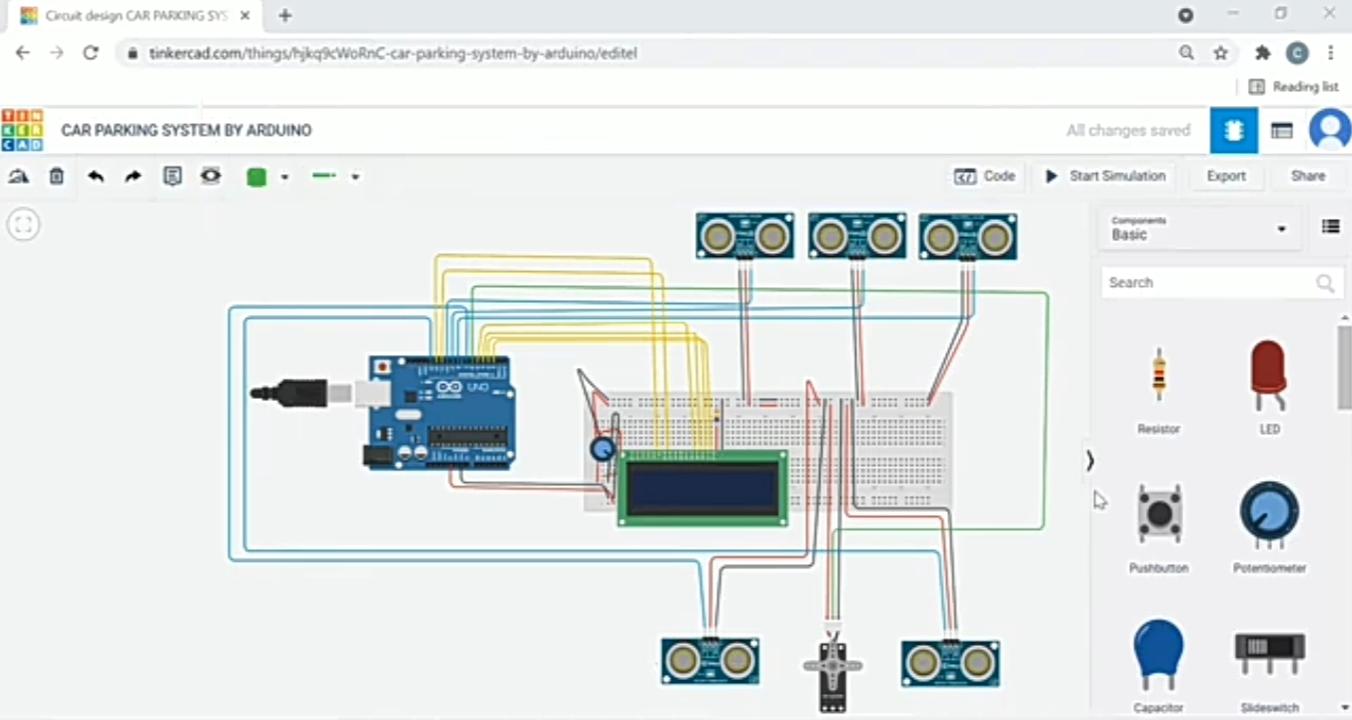
# Iterate over the parking availability data and print the results to the console

For spot in parking\_availability:

Print(spot[“name”], spot[“status”])

This code will make a GET request to the Firebase database to retrieve the parking availability information. The response is a JSON object, which is parsed to get the parking availability data. The data is then iterated over and printed to the console.

**Flowchart:**

**Output:**