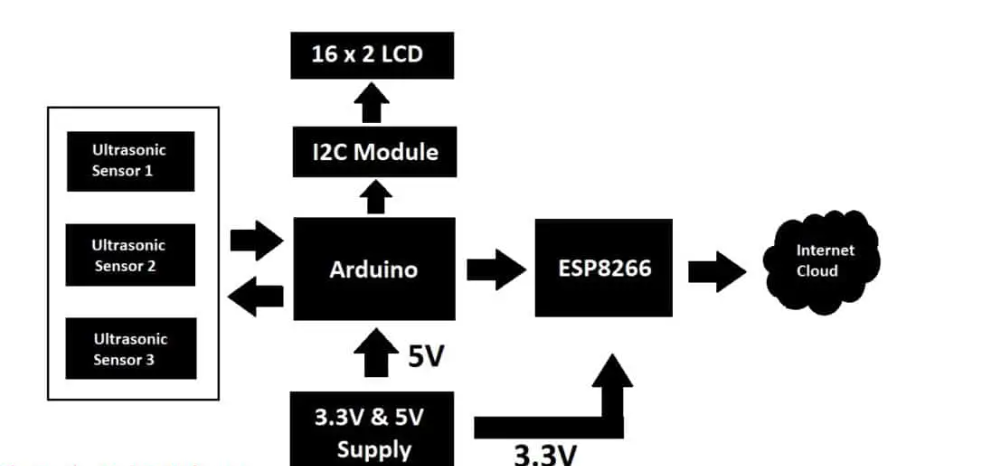
**INTRODUCTION**

What is an IoT Based Parking System?

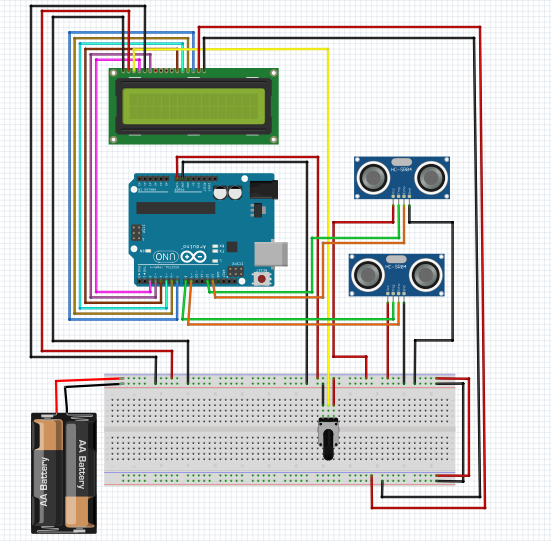
An IoT based parking system is a vehicle parking management system to ease the search for a vacant parking spot in a parking lot through a smartphone. The system utilizes various sensors and microcontrollers with internet capability for detecting parked vehicles and to update the data in real-time on internet.

The IoT-based car parking management system is an innovative solution designed to optimize the utilization of parking spaces and enhance the overall parking experience. This report presents the development of a smart parking system using Arduino and various components. The system incorporates IoT technology to monitor and manage parking spaces in real-time, providing users with accurate information on parking availability.

**BLOCK DIAGRAM AND ARCHITECTURE:**



The circuit we are going to build will be based on the above architecture. An inexpensive Arduino board is going to be the brain of the project.



A **16 x 2 LCD** is utilized for displaying the number of vacant spots locally (without internet). An I2C module is utilized for driving the LCD with just four wires so that GPIO pins can be saved for interfacing the sensors and other modules.

There are three **ultrasonic sensors** for detecting 3 cars / vehicles on the parking spot, we are using ultrasonic sensors instead of IR based sensors because if the parking lot is situated outdoors, infrared light from sunlight may interfere with IR sensors and may give incorrect detection of the vehicle, whereas ultrasonic sensor acts like a mini radar and environmental factors affecting its functionality is minimal.

An ESP8266 Wi-Fi module is used for internet connectivity which sends the parking lot’s data to a cloud server where general public can view the data in real time. A power supply module is utilized which provides 5V and 3.3V for Arduino, ultrasonic sensors and ESP8266 Wi-Fi Module.

**Generic ESP8266 Wi-Fi module:**

This project utilizes a generic ESP8266 Wi-Fi module for internet connectivity. The ESP8266 is actually a miniature microcontroller board and just like Arduino the ESP8266 need a program code to perform its intended function.

It uses UART protocol to communicate with Arduino board; the baud rate we are going to set for UART is 115200 bits per second.

**PIN DIAGRAM of ESP8266:**



**Ultrasonic Sensor HC – SR04:**

The sensor we are going to use for detecting a parked vehicle on its parking spot is called HC – SR04 which is an ultrasonic sensor module.

The ultrasonic sensor module generates ultrasonic sound at around 40 KHz, these sound waves are inaudible to human beings and propagate through air and if the ultrasonic sound wave hits an obstacle, it reflects back to sensor just like radars.

If a car or any vehicle is parked, the ultrasonic sound waves hit the parked vehicle and the sensor module detects the reflection and thus existence of a vehicle on a parking spot is detected.

The ultrasonic sensor module has four pins, Vcc, GND, trigger and echo. The Vcc is connected to 5V supply and GND is connected to GND of the supply. When we apply “HIGH” signal to trigger pin for 10 microseconds, the module generates ultrasonic sound from one of the transducers, when the sound wave hit back the other transducer, the echo pin gets “HIGH” and this signal is detected by Arduino.

The time taken between generating and detecting the sound wave is calculated and thus a parked vehicle is detected.

**LCD display module 16 x 2:**



In this project we are using a 16 x 2 LCD display for displaying parking lot’s data locally without the need for internet. The LCD is driven by an I2C adapter module to reduce the number of wires to four; otherwise you need to connect up to 16 wires to Arduino just to drive the display. If the LCD occupies most of the I/O pins, then there won’t be any pins left for the sensors.

The I2C module has 16 pins at the output and just four at the input: Vcc, GND, SDA and SCL. The SDA and SCL are I2C bus pins which are connected to A4 and A5 pins of Arduino respectively and it operates on 5V.

You can control the contrast of the display by adjusting the trim pot on the I2C adapter module