TROLLEY BOT

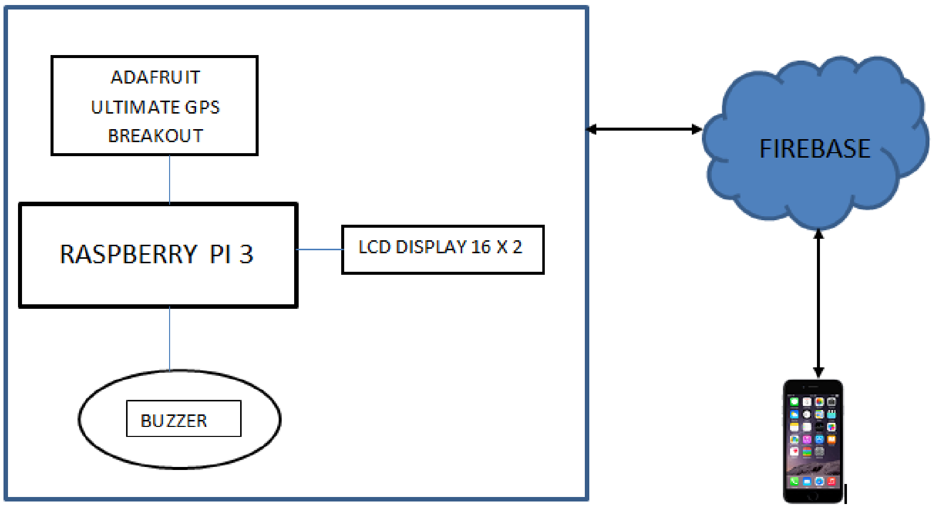
PROJECT OVERVIEW:

Trolleys are one of the most vital components in Supermarkets and Hypermarkets that assist the customer in improving their overall shopping experience. But we find many customers taking the trolleys from the store till their homes and sometimes even abandon them on the roadsides, which would eventually become a nuisance in cleaning up and also cause losses to the retails.

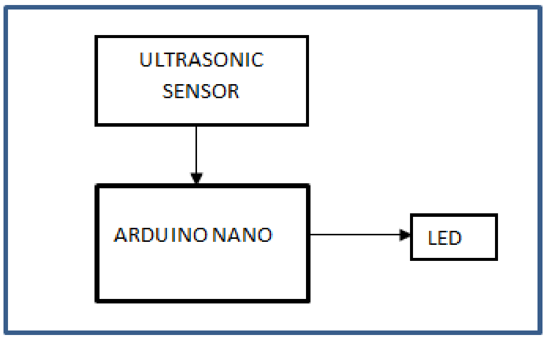
The intention of this project is to solve this problem with the help of a prototype system "TROLLEY BOT". In this system, the trolleys are made smarter by following the Internet of Things paradigm. Each trolley is attached with a GPS sensor that would transmit its location updates continuously to the cloud via a Raspberry PI's internet. We use this data to make sure that the trolleys aren't taken away from the store more than a defined distance. The store's admin will be using an IOS Application to track the trolleys that are in the store and those that are out of the specified zone. As an additional scope, a proximity sensor circuit that is a different unit from the GPS unit (mostly connected on the front of the trolley) is developed to detect any collisions.

SYSTEM ARCHITECTURE:

RASPBERRY PI UNIT:

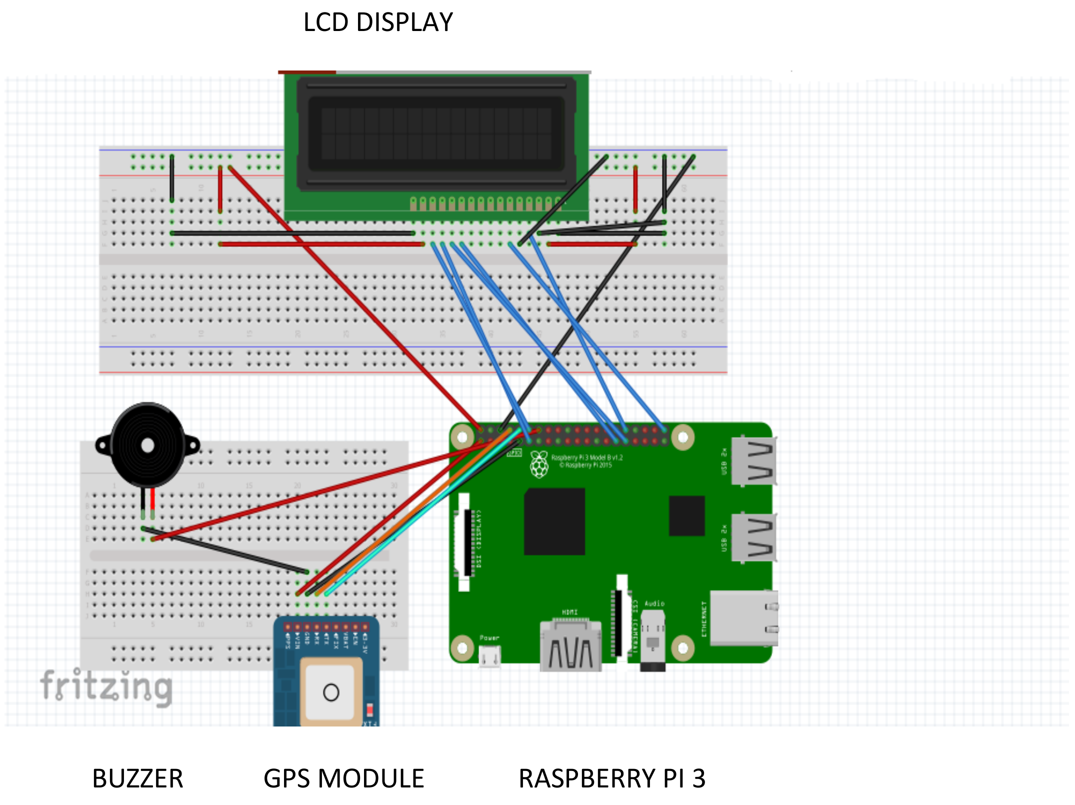


ARDUINO UNIT:

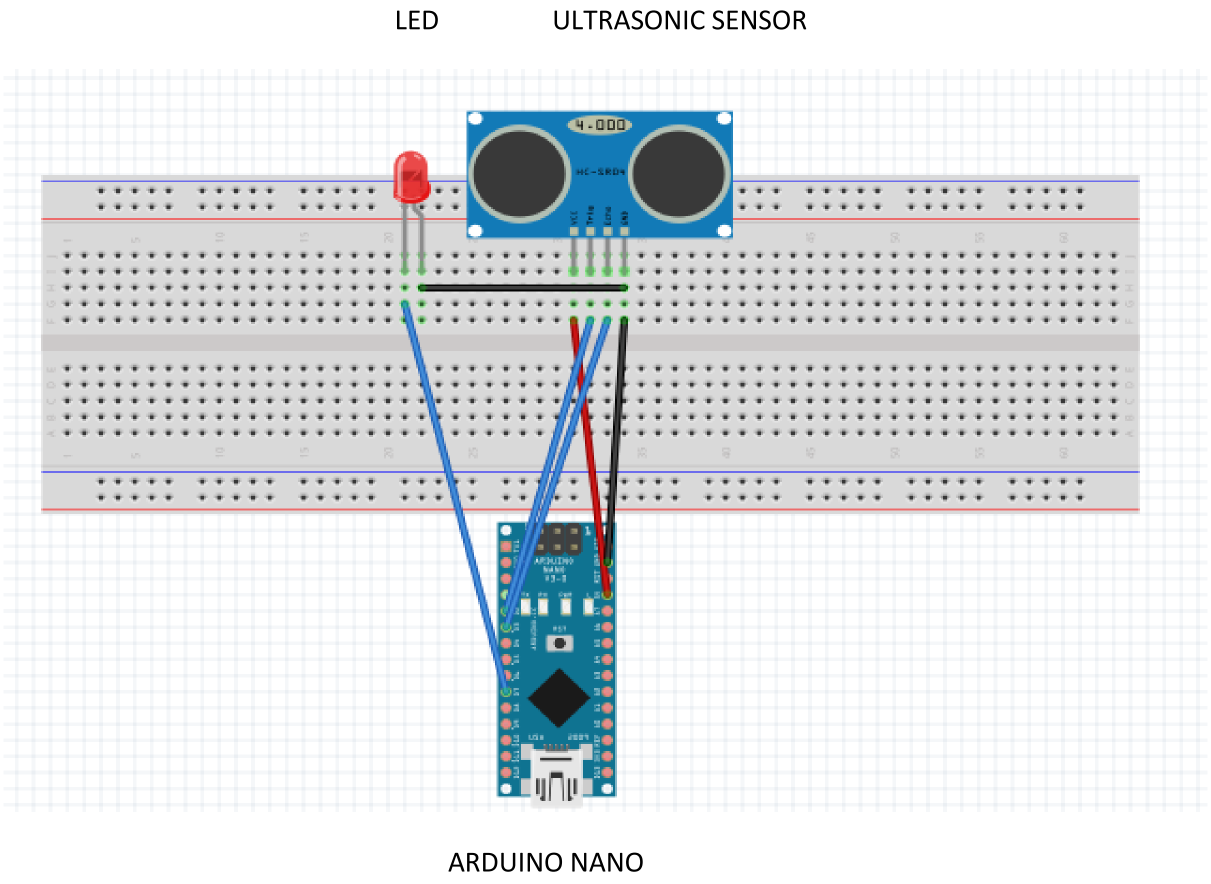


CIRCUIT DIAGRAM:

RASPBERRY PI UNIT:



ARDUINO UNIT:



SYSTEM OPERATION:

The store admin initially registers on the App for his store and enters the details for it like the name of the store and the radius of coverage that we need to track the trolleys in. Once these details are present, we start the Node JS Application that transmits the location of the trolley to Firebase cloud database. It also performs calculations about the status of the trolley and sends it to the cloud along with its location. Whenever the trolley is taken out of the permitted zone, the connected buzzer would ring raising an alarm and also display a message on the connected LCD Display. The IOS Application reads this data, and gives information to the user about the different trolleys present and their status.

DESIGN DECISIONS:

1. Choosing to use Raspberry PI to connect to GPS module:

Initially the plan was to use Arduino to read data from the GPS module and use a special WIFI module (ESP8266) to transmit data to the Raspberry PI which would then transmit the data to Firebase. But this method was not chosen as -

- The circuit with the ESP8266 with an Arduino Board requires a special Voltage divider circuit and a high amount of power.
- We need to use special additional protocols to send and receive data over the WIFI module as it can behave as a client and server.
- Raspberry PI has an inbuilt WIFI adapter that does not require any special setup.
- There was not much difference cost wise between choosing an Arduino with wifi module and voltage divider circuit compared to directly using a Raspberry PI.
- If we had used Arduino, it could be assumed that all the trolleys would send data to Raspberry PI. This would require it to coordinate between all the trolleys and could get heavy loads in real time situations. It could also become a single point of failure for our system.

2. Arduino with Ultrasonic sensor circuit:

A second circuit was built containing a HCSR 04 Ultrasonic sensor that would measure the distance between the trolley and any nearby object. The proximity can be seen by the blinking of a led attached to the circuit. Currently the distance has been hard coded to 30cm.

3. IOS Application listens using a separate listener for each trolley:

We first listen using the addChild Event Type in the 'Trolleys' node of the Firebase structure. This is to monitor addition of any new trolleys. For each trolley, there would be multiple children nodes sending its data. So we use a specific listener to each trolley and update its location on the Map.

4. Internet Connection in Raspberry PI:

If there is no internet connectivity in the raspberry pi, we only display a console message about lack of connectivity. We do not store any historic data, as we do not get any useful information from it, as we always need the latest location.

EXTERNAL LIBRARIES USED:

Some of the new libraries that we used in our project are -

1. SERIAL:

Usually the Adafruit Ultimate GPS Breakout is connected with a USB TTYL to the Raspberry PI. But since we did not have one, we used the other option of using UART serial communication by making /dev/ttyS0 as the terminal where we would get the GPS sensor's data. This serial data had to be parsed to form meaningful NMEA sentences from which we can read GPS latitude and longitude

2. GPS.js:

Once we have NMEA sentences, this library helped us in parsing the sentences to give us the required information and also the distance between any 2 coordinates. We used this information to perform tracking of trolleys within the specified range.

3. LCD:

When a customer violates the rules and takes trolleys out of permitted range, we wanted to display that it is not allowed using a LCD display. The LCD library assisted us to form an lcd object with the required pins in the 4 bit mode, so that we can directly use print statement to print information on the LCD.

4. DNS:

This library helped us in error handling scenario of checking whether the Raspberry PI is connected to the internet. It checks whether DNS lookup is happening on the Google website and returns an error object if unable to do so. We use this to inform the user that the PI's internet connectivity is not there.

5. FIREBASE GOOGLE SIGN IN:

We used Firebase's Google sign in API along with the usual password and email authentication to provide ease of access into the App. If there is a single instance of the app in a store, any staff of the store can access it without knowing the credentials.

6. ARDUINO SERIAL COMMUNICATION:

Since we wanted to connect the ultrasonic sensor to be connected to Arduino board, we used the serial communication at 9600 Baud rate to get the distance from the nearest obstacle.

FURTHER SCOPE:

- Our prototype can be further extended to use the line follower circuit to which the arduino circuit can assist in detecting collisions.
- Any motion detected in the sensors after closing hours, can be used to notify the user of intrusion.
- The prototype can be used along with other trolley locking mechanisms to lock the trolley when a user tries to take it out of the permitted range.