FINAL REPORT

Project Title

Indoor Mapping and Air Quality System

Project Members

Member #1	Member #2
Osman Çiçek	Mehmet Çağlar Çınkır
2015513013	2016513404

Abstract:

In this project, we made an IoT system that does indoor mapping of a closed environment (mostly rooms) and measurements of elements such as air-quality, humidity and temperature. We used machine learning to make predictions of numbers of people can inhabit in these environments. For sketching maps, we used turtle module in python and to handle measured data, we used Firebase servers.

In addition, we had problems with Raspberry pi booting as we mentioned before and we couldn't solve the problem. So, we had to switch from Raspberry pi to some other hardware. We used NodeMCU to connect to Wi-Fi and continued from there.

To see the system operation, there will be links for videos that are the car operation and the drawing of the measurements on turtle screen in the Appendix.

Problem Definition:

Collecting data from an environment remotely can be quite important in some applications. The air might be poisonous or the temperature might be at a risky level and we would also want to know how much space there is in an unknown environment. So, this project is for exploring purposes and make predictions on how many people can live in a certain area.

Measurements are recorded flawlessly to the server and can be easily retrieved from the server to a computer that has the python code running. With this code, we are able to sketch the obtained distance measurements according to the car position and we can also monitor other types of measurements on the screen.

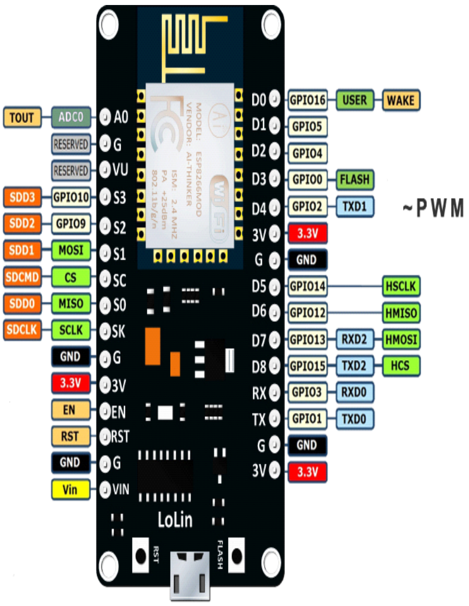
One weakness we can name is power. If the power is obtained from the grid, we can't find any errors in the operation. If the power is obtained from a power bank while the car is roaming the area, we sometimes face some problems with the operation but that can be solved with a better power source.

System Design and Features:

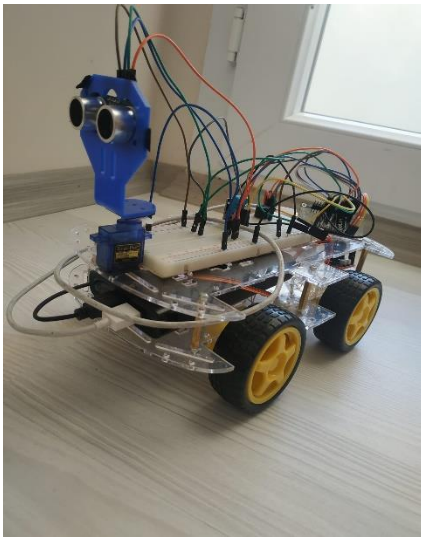
Hardware Components

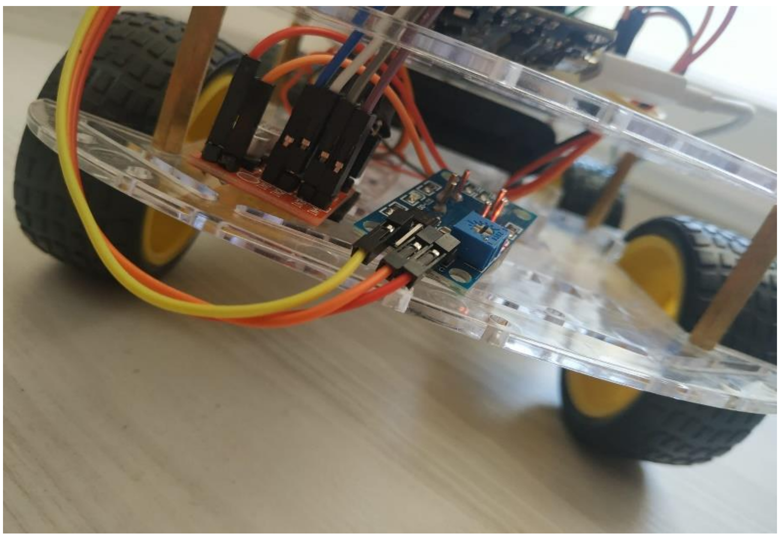
1. 4WD Car Kit
2. Lolin NodeMCU V4
3. Buzzer
4. HCSR-04
5. Breadboard
6. MX1508 Motor Driver
7. SG90 Servo Motor (x1)
8. DHT-11 Temperature-Humidity Sensor
9. MQ-135 Air Quality Sensor
10. 1.5V-2.1A Power Bank

NodeMCU Pin Configuration and Pin Connection Information



MX1508:	**HCSR-04:**	**DHT-11:**
IN1 => D4 IN2 => D3 IN3 => D2 IN4 => D1	TRIG => D5 ECHO => D0	IN => D8
SERVO	**BUZZER:**	**MQ-135:**
IN => D6	IN => D7	IN => A0 VCC => VU





Procedure

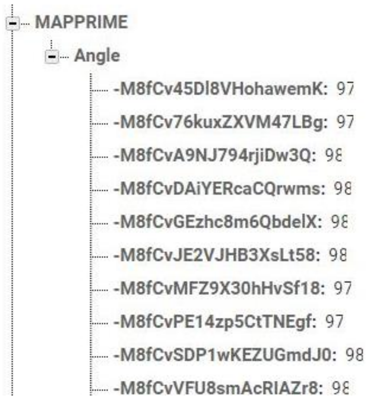
For starters, in order to make use of NodeMCU, we first adjusted the Arduino IDE to recognize ESP8266 by referencing its URL. Then we added certain libraries to make communication between the ESP8266 module and a Firebase server. We used HTTP protocol for this overall communication. Codes for this purpose can be seen below.

```
#include <Servo.h>
#include "DHT.h"
#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
Servo sg90;
```

```
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
Serial.print("Connecting");
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  delay(500);
}
```

After this setup there will be measurements taken in the room. Since air-quality, humidity and temperature can be measured and transferred almost instantly, it works different for measuring distance while staying on the center point in the room. To achieve that, we built an algorithm for the car to find the center point. In this algorithm, the car basically roams the room until it finds a wall. Then changes its direction and search for another wall. With these repetitive processes, the car can finally find the center and take its position on this point.

When the car is positioned on the center, it starts to take distance measurements while rotating from 0 to 359 degrees and it takes one measurement for each degree. Then it transfers these “angle” and “distance” values along with other measurements to the Firebase Database. For this database, the images below can be observed.



The values in this database is accessible from any device since we set the authentication to “None”. This could be changed if we would like access these data via ID and password but we didn’t apply this function because privacy wasn’t our main focus in this project.

For the mapping of this data, we created an application in Spyder on a different computer that is connected to the Internet. This application sends a HTTP request to the database to retrieve the measured values. The code for this purpose is below.

This line is to set URL reference and authentication.

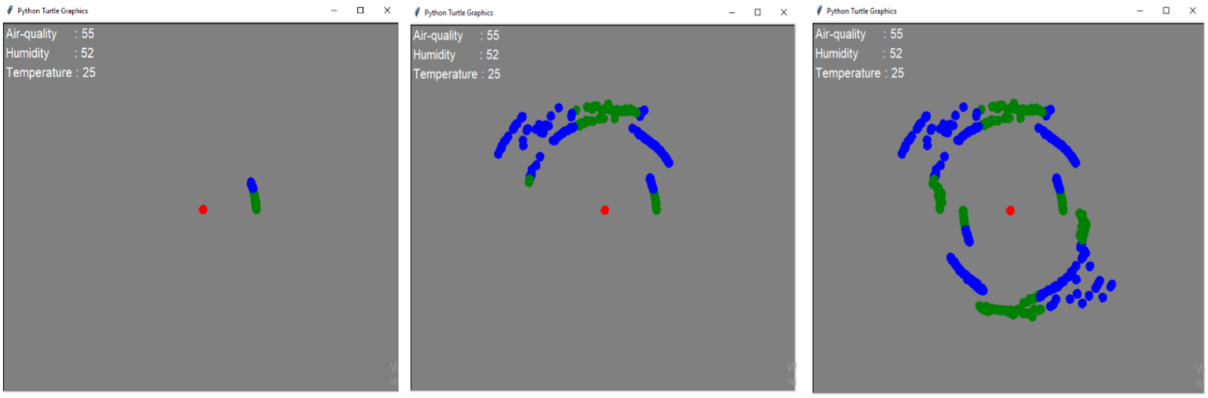
```
firebase = firebase.FirebaseApplication("https://fir-arduino-269f4.firebaseio.com/",None)
```

Code below is for getting all required data from the server.

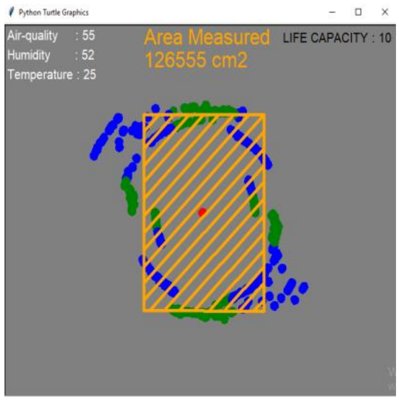
```python
while True:
    result = firebase.get('/MAPPRIME','')        ## measurements are obtained from server
    if result != None:
        result = result["Angle"].values()
        if len(result) == 360:
            result = list(enumerate(result))
            result1 = [list(item) for item in result]

            air_quality = firebase.get("/DATAPRIME/Air-quality", "")
            humidity = firebase.get("/DATAPRIME/Humidity", "")
            temperature = firebase.get("/DATAPRIME/Temperature", "")
            print("DATA Obtained")
            break
        else:
            print("More Data Required")
    else:
        print("No Data")
        time.sleep(1)
```

After all the measurements are obtained, program start drawing the map. This is done using Turtle Graphics module in python. These drawings are not 100% accurate since there will be some portion of wrong measurement data from HCSR-04 but still, there is a good amount of precision. We can view the drawing process on the turtle screen below.



As observed on the images above, there are measurements for each angle and they are drawn by blue or green dots. Since, the distance data around the corners can have more errors then the ones around the angles 0, 90, 180 and 270. We used the measurements around these angles to have better estimation of the area. At this drawing, only the areas on right, left, up and down directions with a range of 40 degrees are considered when calculating the area of the space. So, only the green dots are used to sketch the shape. In this measurement, area is calculated around 12.65 m^2.



To calculate the life capacity of the environment observed, we used the following linear regression machine learning code. For the training data to make the prediction, we used the numerical values written in the Results and Conclusion part.

The code mentioned above for linear regression machine learning.

```
data = pd.read_csv("veri.csv") # TRAIN DATA  veriyi csv den çektim

result2 = firebase.get('/DATAPRIME','')
result2 = list(result2.values())
firebase.put("/DATAPRIME", "Area", area)
result2[1] = result2[1]/10000.0 # Convert cm unit area to m unit area
result2 = np.array(result2)
result2 = result2.reshape(1,-1)  # Air-quality, Area, Humudity, Temperature veriyi tek satıra çevirmek için şekillendirdim

# DATA SPLİT
x_train = data.iloc[:,:-1].values  # veri kümem öğrenme verisini ve sonucu beraber tutuyordu onları ayırdım
y_train = data.iloc[:,-1:].values

# DATA NEW DESIGN ayırdığım verilerin parametre isimlerini verdim tekrardan
X_train = pd.DataFrame(data = x_train, index = range(34), columns = ["Air-q","Area","Humidity","Temperature"])
Y_train = pd.DataFrame(data = y_train, index = range(34), columns= ["NoL"])
X_test = pd.DataFrame(data = result2, index = range(1), columns = ["Air-q","Area","Humidity","Temperature"])

# REGRESSION
regression = LinearRegression()  # modelin nesnesini oluşturdum

regression.fit(X_train,Y_train)  # öğrenme verisini ve sonucu verdim . böylece algoritma parametreler arası ilişki kurdu
Y_pred = regression.predict(X_test) # tahmin için internetten çektiğim veriyi girdim

# FLOOR NUMBER
human_number = floor(Y_pred)  # insan sayısı, virgüllü olamaz tabana yuvarladım

# UPLOAD DATA TO MY CLOUD SERVER
firebase.put("","NOFPRIME", human_number) # Upload new data
```

Results and Conclusion

To train to make a prediction we used the data below that are air-quality, area, humidity, temperature and number of people can inhabit.

```
70,10.0,30,25,4
50,25.2,25,22,20
90,8.3,36,28,2
150,25.23,15,30,0
50,20.01,60,40,10
77,0.98,32,28,1
65,12.07,40,25,8
100,30.56,50,50,10
50,15.01,5,0,9
80,10.10,25,13,7
40,3.0,15,20,2
```

```
100,24.23,56,45,12
75,0.99,26,25,1
80,1.0,36,40,0
125,25.25,30,22,3
82,1.26,26,22,1
105,19.98,40,24,10
125,23.56,42,25,2
140,15.37,32,30,0
59,5.15,16,22,4
100,13.62,60,35,6
123,17.16,60,20,3
159,26.10,36,32,0
122,10.13,38,10,1
```

```
145,29.55,21,36,0
120,2,33,38,0
160,25.23,30,25,0
56,11.33,23,36,8
89,19.30,25,25,14
75,20.05,30,32,16
60,13.06,26,23,10
66,15.02,25,27,13
75,0.56,31,26,0
55,1.01,22,15,1
```

In this project, we first planned to use a Raspberry pi. Unfortunately, our pi was damaged and we had to go along with what we have at the moment which was a NodeMCU and it was quite enough to finish the project.

Appendix

Codes will be attached to the mail as pdf files.

The video of the car sending data and the video of drawing these data with turtle can be downloaded with the link below (these are the videos we filmed)
https://wetransfer.com/downloads/c9163271b1d4705a09eb2d26dcb2f73920200531205614/892aaae20550e791d361a4f38713df1a20200531205614/591ec9