Introduction

The aim of this report is to detail the steps, methods and techniques for designing and implementing wave sensor. a wave sensor is in internet of things device that use the raspberry 3+ connected to a light dependent resistor via MCP300, the MCP300 is an analog digital converter chip that use serial peripheral interface to communicate with the raspberry pi3+.

The wave sensor uses the LDR to sense changes in light caused by a waving hand that frequently blocks the light when a person waves his/her hand, it then stores the data about changes in light caused by the waving. Data captured is send to a central sever via
Transmission Control Protocol/Internet Protocol python sockets. the data sent to the server via internet contains information about how fast the hand is waving, the number of times the hand waves, the speed at which the hand is waving and the value of luminosity of the light.

The server that is connected to the wave sensor via the internet also controls the waves by sending control commands that can make the wave sensor to stop or start sensing waves, the commands are also send over TCP/IP and when the waves receives the command it uses the python code "os.system("command from server ")" to run the command

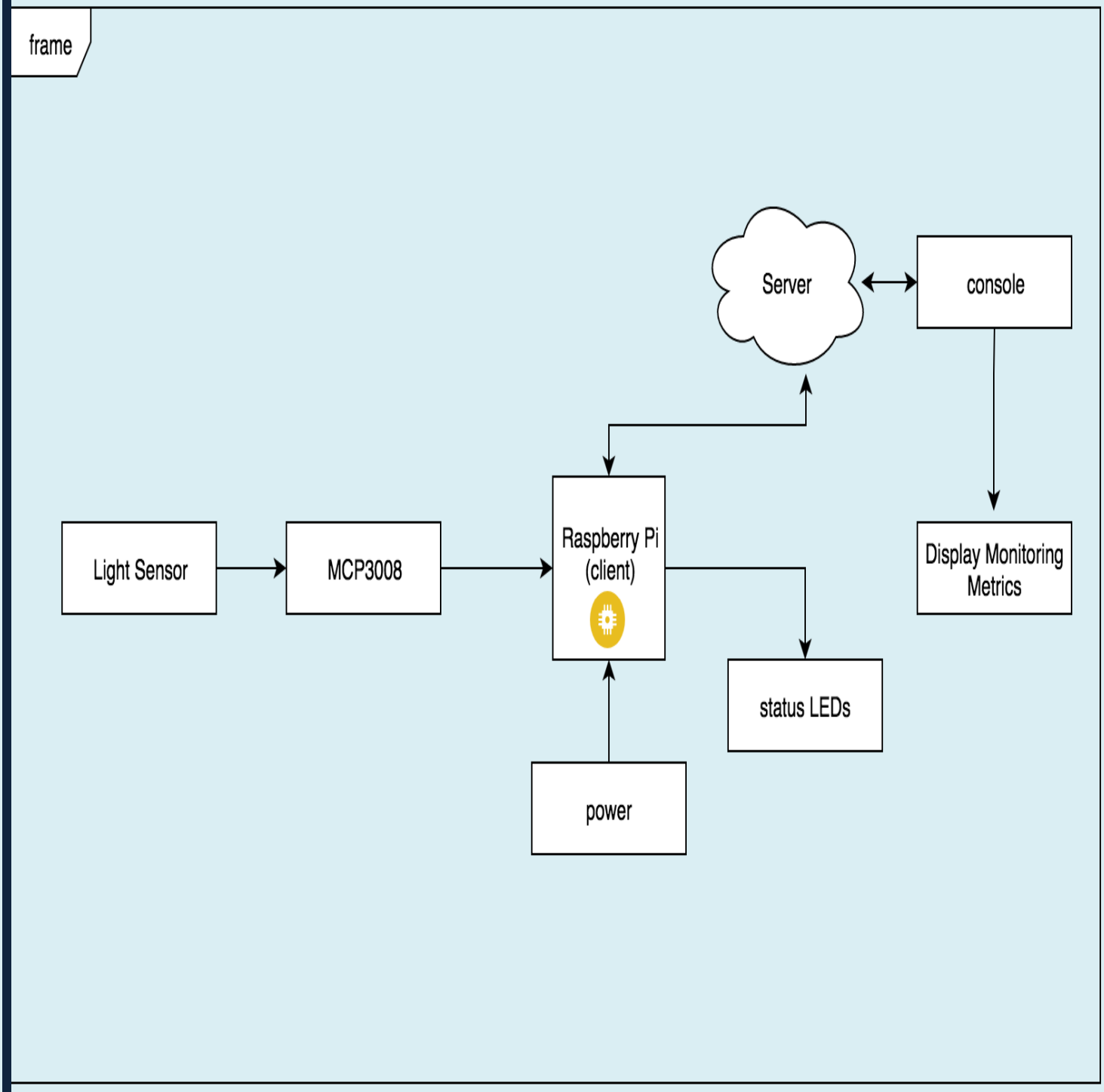
When the sever the gets the data from the waver sensor via TCP/IP is displays it on the console.

.

Design

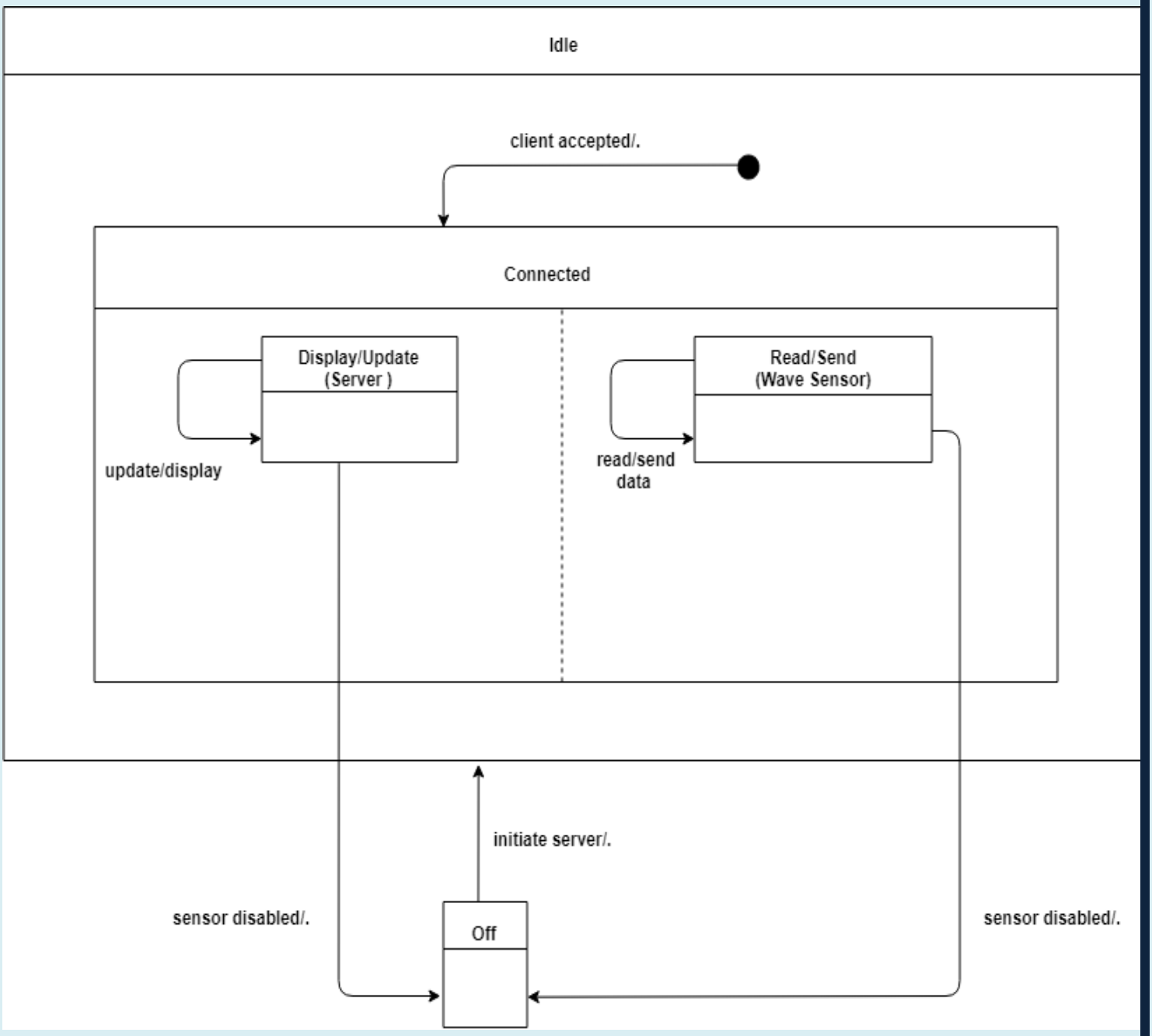
Block diagram

The block diagram below shows how the components and how eco system of the wave sensor works , this is explained fully in detail the introduction.



State chart

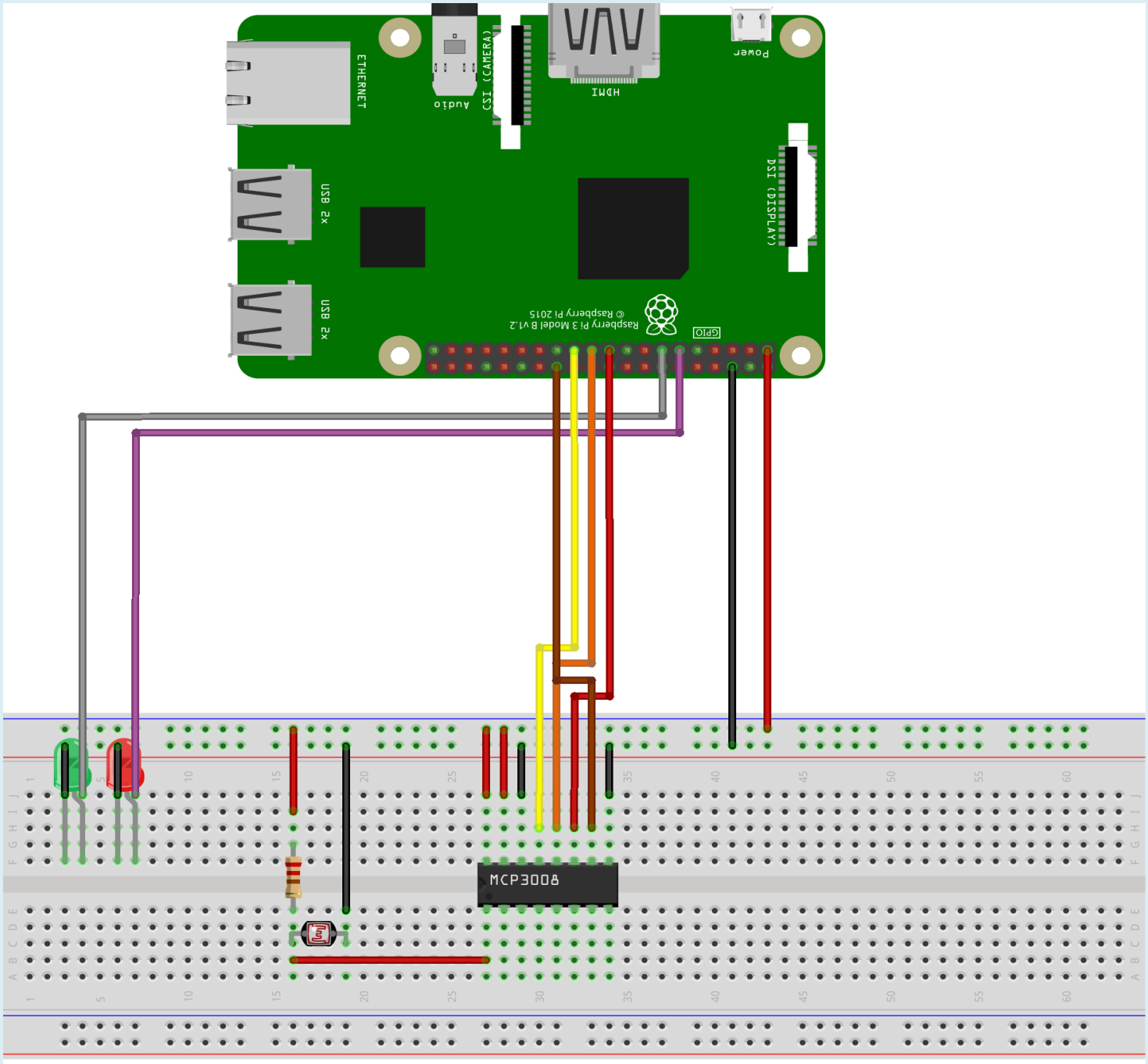
The state chart below shows how the raspberry and interacts with the central server (computer that initiates connection) , the server waits for a TCP connection from the wave sensor once the wave requests a TCP connection it the server acknowledges the request and accept the connection , then the wave sensor starts sending the data to the client the server.



Hardware Circuit Design or a wave sensor

Parts used :

1. **Raspberry Pi Model 3B**
2. **MCP3008 SPI**
3. **Light Dependent Resistor**
4. **100 ohm Resistor**
5. **1x Green LED, 1x Red LED**



Implementation

there are two software codes implemented the first is for the sever that receives data from the wave sensor and the second one is for the wave sensor .

for the ***server***:

you first need to import python socket modules that will help accept TCP connections from the client.

```
import socket

port =1693
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
```

Then then to make sure server waits for connection I used the this:

```
server.bind(('',port))
server.listen(2)
client,address= server.accept()
```

The sever also to has accept data from the from the wave sensor and also to know if the connection is lost or not .

```
client,address= server.accept()
while True:
    data =client.recv(1024)
    if not data:
        break
    print(data.decode('utf-8'))
```

For the ***wave sensor***:

You first need to know that static Internet Protocol version 4 address (Ipv4) and the of the server that the wave sensor is trying to connect to then try to establish a TCP connection with it.

```
port= 1546
server='196.46.9.86'
wave_sensor= socket.socket(socket.AF_INET,socket.SOCK_STREAM)
wave_sensor.connect((server,port))
```

Then you must sample the wave sensor data and send it to the sever this is done in parallel you sing threads to ensure that there's minimal delay when I comes to sending data over internet.

```
def Count():
    global Lightwave
    global Period
    global total
    time.sleep(0.05)
    now= Lightwave
    Period=""
    duration=datetime.now()
    while True:
        time.sleep(0.1)
        value=Lightwave
        if(value>1):
            total+=1
            Period=str(datetime.now()-duration)[3:-4]
            duration=datetime.now()
            time.sleep(0.1)


Thread2=threading.Thread(target=Count)

def Sample():
    global Lightwave
    global total
    global wave_sensor
    global Period
    while True:

        Light_value = ADC(light)
        Lightwave = LightReadding(Light_value)
        time.sleep(0.3)
        print(Lightwave," ",Period)
        wave_sensor.send(str.encode("number of waves: "+str(total)+str(Lightwave)+" wave speed"+str(Period)))

Thread1=threading.Thread(target=Sample)
```

How run code

1. Before trying to run the codes please ensure that you know the Ipv4 address of the server you are trying to connect to, you can change it in the "wave sensor.py" file you also need to know the port at which the program will listen to connection on the server's side. Both the port number and the Ipv4 can be changed in the "wave sensor.py"

```
15  port= 1369
16  server='137.158.63.88'
```

2. You check the firewall settings on your windows /linux machine make sure it allows incoming requests

If you have completed the steps above, then you do the steps below.

First you must run the sever, after that you run the client:

The sever:
It is very important that you include the "\" in the command, if you don't include it the operating system operating system will complain
For example, you must type "python3 wave\ sensor.py"

```
shRia002@DESKTOP-AQLA2MG:/mnt/c/Users/OFENTSE TSHEPE$ cd Desktop/
shRia002@DESKTOP-AQLA2MG:/mnt/c/Users/OFENTSE TSHEPE/Desktop$ python3 wave\ sensor.py
```

The wave sensor:
it is straight forward, it needs nothing special

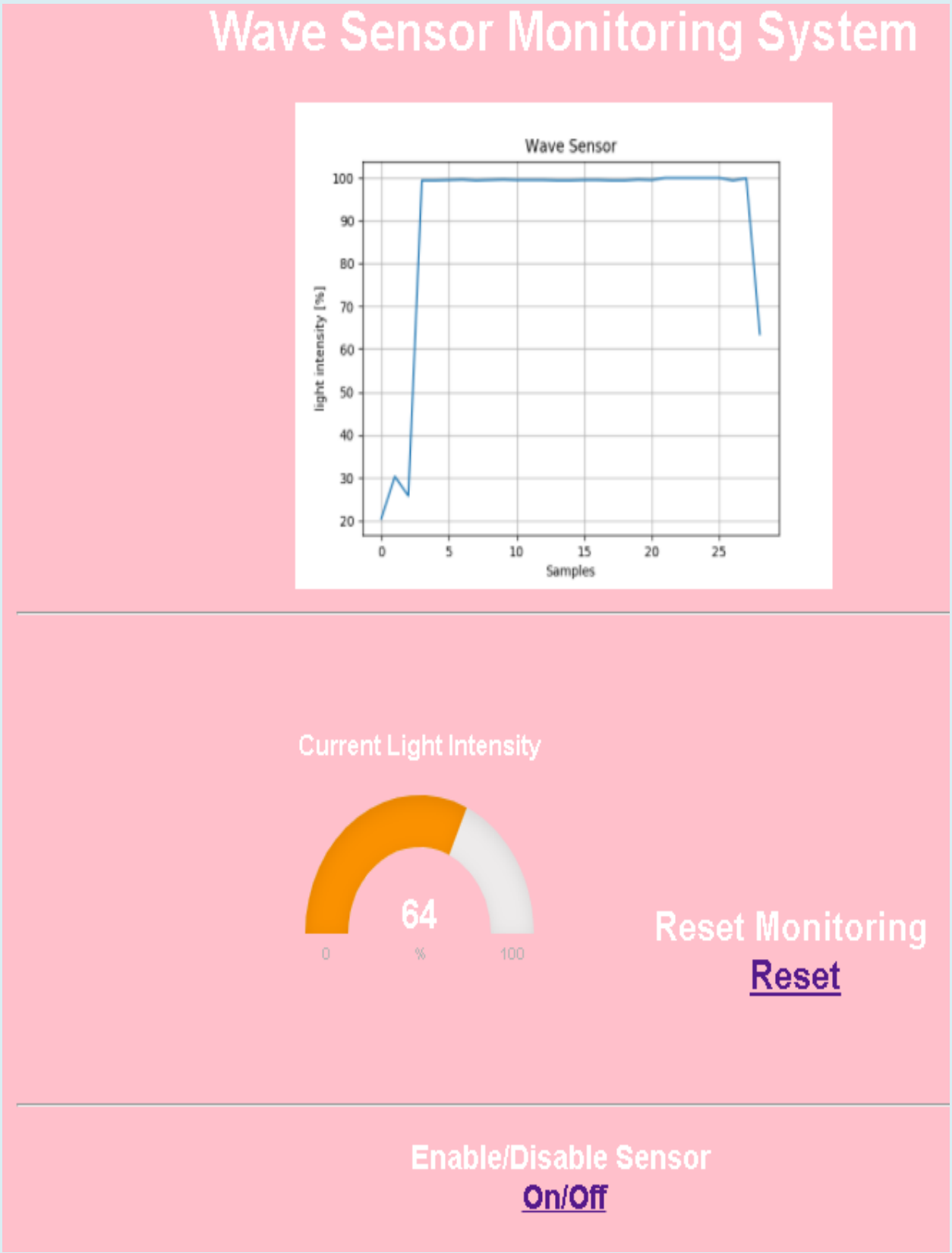
```
shRia002@DESKTOP-AQLA2MG:/mnt/c/Users/OFENTSE TSHEPE/Desktop$ python3 server.py
```

Test and Results

I got the following results after running the codes, this result were displayed on the console on the server side.

```
number of waves: 25 light percentage: 30.3 wave speed0:00.40
number of waves: 26 light percentage: 42.42 wave speed0:00.80
number of waves: 26 light percentage: 30.3 wave speed0:00.80
number of waves: 27 light percentage: 39.39 wave speed0:00.60
number of waves: 28 light percentage: 33.33 wave speed0:00.40
number of waves: 28 light percentage: 48.48 wave speed0:00.40
number of waves: 28 light percentage: 30.3 wave speed0:00.40
number of waves: 29 light percentage: 48.48 wave speed0:00.80
number of waves: 30 light percentage: 33.33 wave speed0:00.40
```

the results show information about the number waves ,the percentage of light seen by the light sensor and the waves speed in seconds .



this shows the plotted results in the graphical user interface.

Conclusion

The system worked well when the raspberry pi model 3 was connected to the internet over ethernet but it did not work well when I t was connected to WIFI it kept losing connection over and over again, so it was very hard determine if the code was working well or if was error with the network. the project could be more useful if was used more environment variables like temperature, humidity ,barometric pressure and other related variables , it has huge capabilities

References

1. **https://docs.python.org/3/library/socket.html**
2. **https://docs.python.org/3/howto/sockets.html**
3. **ttps://www.raspberrypi.org/documentation/configuration/wireless/wireless-cli.md**