Date	19 November 2022
Team Id	PNT2022TMID42771
Project Name	Project - Signs with smart connectivity for Better road safety

Signs with smart connectivity for Better road safety

SOURCE CODE:

weather.py:

```
import requests as reqs


def get(myLocation,APIKEY):
    apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"
    responseJSON = (reqs.get(apiURL)).json()
    returnObject = {
        "temperature" : responseJSON['main']['temp'] - 273.15,
        "weather" : [responseJSON['weather'][_]['main'].lower() for _ in range(len(responseJSON['weather']))],
        "visibility" : responseJSON['visibility']/100, # visibility in percentage where 10km is 100% and 0km is 0%
    }
    if("rain" in responseJSON):
```

returnObject["rain"] = [responseJSON["rain"][key] for key in responseJSON["rain"]]

return(returnObject)

PUBLISH DATA.PY

```
import wiotp.sdk.device # python -m pip install wiotp
import time
# IMPORT SECTION ENDS
#
# API CONFIG SECTION STARTS
myConfig = {
"identity" :{
"orgId" : "gsqz5f",
"typeId" : "NANDY",
"deviceId" : "12345"
},
"auth" : {
"token" : "9876543210"
}
}
# API CONFIG SECTION ENDS
```

```
#
# FUNCTIONS SECTION STARTS
def myCommandCallback(cmd):
 print("recieved cmd : ",cmd)
def logData2Cloud(location,temperature,visibility):
 client =
wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers
=None)
 client.connect()

client.publishEvent(eventId="status",msgFormat="json",da
ta={
    "temperature" : temperature,
    "visibility" : visibility,
    "location" : location
 },qos=0,onPublish=None)
 client.commandCallback = myCommandCallback
 client.disconnect()
 time.sleep(1)
 import requests as reqs
```

```
def get(myLocation,APIKEY):
    apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"
    responseJSON = (reqs.get(apiURL)).json()
    returnObject = {
        "temperature" : responseJSON['main']['temp'] - 273.15,
        "weather" : [responseJSON['weather'][_]['main'].lower() for _ in range(len(responseJSON['weather']))],
        "visibility" : responseJSON['visibility']/100, # visibility in percentage where 10km is 100% and 0km is 0%
    }
    if("rain" in responseJSON):
        returnObject["rain"] = [responseJSON["rain"][key] for key in responseJSON["rain"]]
    return(returnObject)


# Python code


# IMPORT SECTION STARTS
```

```
import weather
from datetime import datetime as dt


# IMPORT SECTION ENDS
# -------------------------------------------
# UTILITY LOGIC SECTION STARTS
def processConditions(myLocation, APIKEY ,localityInfo):
    weatherData = weather.get(myLocation,APIKEY)


    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData else localityInfo["usualSpeedLimit"]/2
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else finalSpeed/2


    if(localityInfo["hospitalsNearby"]):
        # hospital zone
        doNotHonk = True
    else:
        if(localityInfo["schools"]["schoolZone"]==False):
```

```
            # neither school nor hospital zone
            doNotHonk = False
        else:
            # school zone
            now = [dt.now().hour,dt.now().minute]
            activeTime = [list(map(int,_.split(":"))) for _ in
localityInfo["schools"]["activeTime"]]
            doNotHonk =
activeTime[0][0]<=now[0]<=activeTime[1][0] and
activeTime[0][1]<=now[1]<=activeTime[1][1]


    return({
        "speed" : finalSpeed,
        "doNotHonk" : doNotHonk
    })


# UTILITY LOGIC SECTION ENDS
```

brain.py:

```
# Python code
```

```
# IMPORT SECTION STARTS


import weather
from datetime import datetime as dt


# IMPORT SECTION ENDS
# -----------------------------------------
# UTILITY LOGIC SECTION STARTS
def processConditions(myLocation,APIKEY,localityInfo):
    weatherData = weather.get(myLocation,APIKEY)


    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in
weatherData else localityInfo["usualSpeedLimit"]/2
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else
finalSpeed/2


    if(localityInfo["hospitalsNearby"]):
        # hospital zone
        doNotHonk = True
    else:
```

```
        if(localityInfo["schools"]["schoolZone"]==False):
            # neither school nor hospital zone
            doNotHonk = False
        else:
            # school zone
            now = [dt.now().hour,dt.now().minute]
            activeTime = [list(map(int,_.split(":"))) for _ in
localityInfo["schools"]["activeTime"]]
            doNotHonk =
activeTime[0][0]<=now[0]<=activeTime[1][0] and
activeTime[0][1]<=now[1]<=activeTime[1][1]


    return({
        "speed" : finalSpeed,
        "doNotHonk" : doNotHonk
    })


# UTILITY LOGIC SECTION ENDS
```

main.py:

import brain

```
# IMPORT SECTION ENDS
#
# USER INPUT SECTION STARTS
myLocation = "Chennai,IN"
APIKEY = "92eedd4b0b4cd6c543c365f562a59ab3"
localityInfo = { "schools" : {
"schoolZone" : True,
"activeTime" : ["7:00","17:30"] # schools active from 7 AM till 5:30 PM
},
"hospitalsNearby" : False, "usualSpeedLimit" : 40 # in km/hr
}
# USER INPUT SECTION ENDS
#
# MICRO-CONTROLLER CODE STARTS
while True :
print(brain.processConditions(myLocation,APIKEY,localityInfo))
```

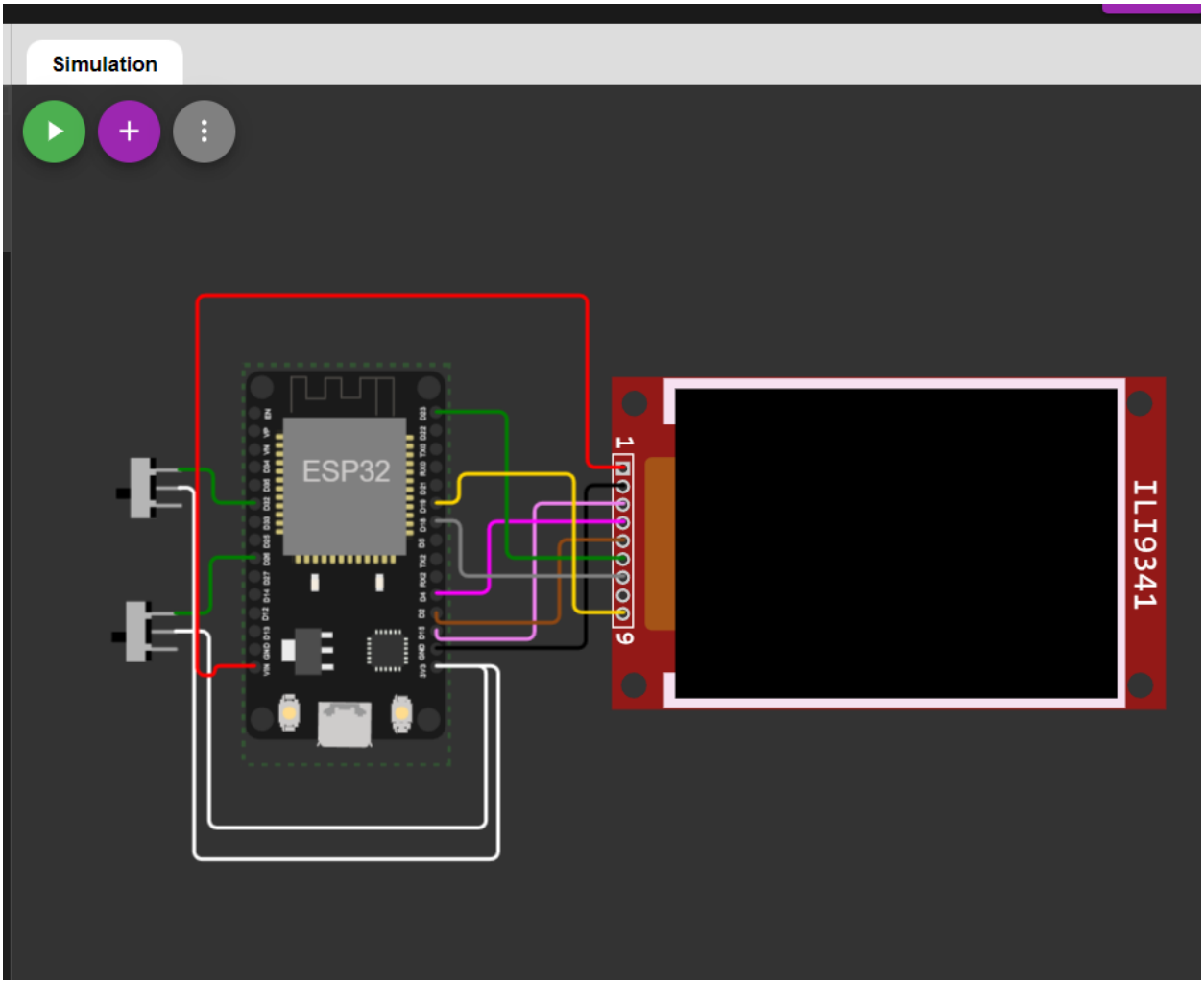
Wokwi Circuit :

Wokwi Code

Wokwi Link

https://wokwi.com/projects/348420917875966547

Circuit Diagram :



ESP 32 CODE :

```
#include <WiFi.h>

#include <HTTPClient.h>
```

```
#include <Adafruit_GFX.h>

#include <Adafruit_ILI9341.h>

#include <string.h>


const char* ssid = "Wokwi-GUEST";

const char* password = "";


#define TFT_DC 2

#define TFT_CS 15

Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);


String myLocation = "Chennai,IN";

String usualSpeedLimit = "70"; // kmph


int schoolZone = 32;

int hospitalZone = 26;


int uid = 2504;


String getString(char x)

{

    String s(1, x);
```

```
    return s;
}

String stringSplitter1(String fullString,char delimiter='$')
{
    String returnString = "";
    for(int i = 0; i<fullString.length();i++) {
        char c = fullString[i];
        if(delimiter==c)
            break;
        returnString+=String(c);
    }
    return(returnString);
}

String stringSplitter2(String fullString,char delimiter='$')
{
    String returnString = "";
    bool flag = false;
    for(int i = 0; i<fullString.length();i++) {
        char c = fullString[i];
        if(flag)
```

```
            returnString+=String(c);
        if(delimiter==c)
            flag = true;
    }
    return(returnString);
}

void rightArrow()
{
  int refX = 50;
  int refY = tft.getCursorY() + 40;

  tft.fillRect(refX,refY,100,20,ILI9341_RED);
  tft.fillTriangle(refX+100,refY-30,refX+100,refY+50,refX+40+100,refY+10,ILI9341_RED);
}

void leftArrow()
{
  int refX = 50;
  int refY = tft.getCursorY() + 40;

  tft.fillRect(refX+40,refY,100,20,ILI9341_RED);
```

```
  tft.fillTriangle(refX+40,refY-30,refX+40,refY+50,refX,refY+10,ILI9341_RED);
}

void upArrow()
{
  int refX = 125;
  int refY = tft.getCursorY() + 30;

  tft.fillTriangle(refX-40,refY+40,refX+40,refY+40,refX,refY,ILI9341_RED);
  tft.fillRect(refX-15,refY+40,30,20,ILI9341_RED);
}

String APICall() {
  HTTPClient http;

  String url = "https://node-red-nwmrt-2022-11-04.eu-gb.mybluemix.net/getSpeed?";
  url += "location="+myLocation+"&";
  url += "schoolZone="+(String)digitalRead(schoolZone)+(String)"&";
  url += "hospitalZone="+(String)digitalRead(hospitalZone)+(String)"&";
  url += "usualSpeedLimit="+(String)usualSpeedLimit+(String)"&";
  url += "uid="+(String)uid;
  http.begin(url.c_str());
```

```
  int httpResponseCode = http.GET();

  if (httpResponseCode>0) {
    String payload = http.getString();
    http.end();
    return(payload);
  }
  else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }
  http.end();
}

void myPrint(String contents) {
  tft.fillScreen(ILI9341_BLACK);
  tft.setCursor(0, 20);
  tft.setTextSize(4);
  tft.setTextColor(ILI9341_RED);
  //tft.println(contents);

  tft.println(stringSplitter1(contents));
```

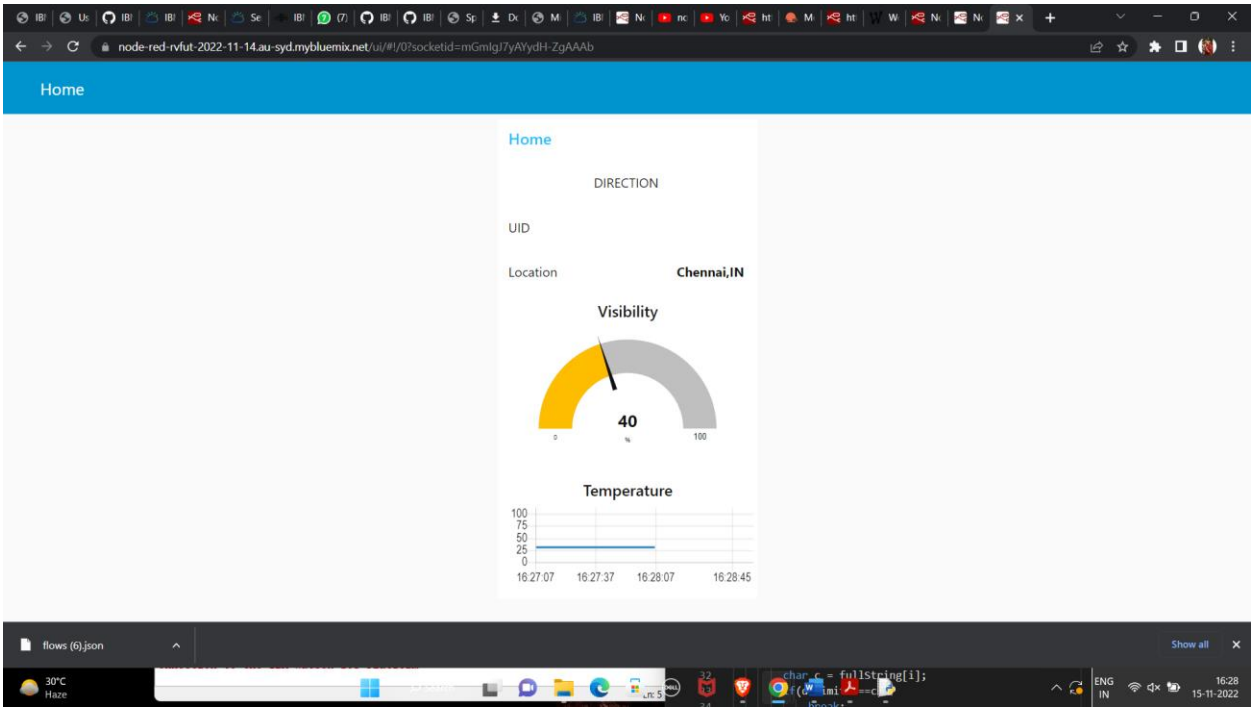
```
  String c2 = stringSplitter2(contents);
  if(c2=="s") // represents Straight
  {
    upArrow();
  }
  if(c2=="l") // represents left
  {
    leftArrow();
  }
  if(c2=="r") // represents right
  {
    rightArrow();
  }
}

void setup() {
  WiFi.begin(ssid, password, 6);

  tft.begin();
  tft.setRotation(1);

  tft.setTextColor(ILI9341_WHITE);
```

```
  tft.setTextSize(2);
  tft.print("Connecting to WiFi");

  while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    tft.print(".");
  }

  tft.print("\nOK! IP=");
  tft.println(WiFi.localIP());
}

void loop() {

  myPrint(APICall());

  delay(100);
}
```

Output :

Node RED Dashboard :

LINK TO NODE RED DASHBOARD



[LINK TO WOKWI PROJECT]