```
/*

*/
#include <WiFiNINA.h>
#include <Arduino_JSON.h>
#include <Arduino_MKRIoTCarrier.h>
MKRIoTCarrier carrier; //Constructor of the carrier maybe we can include it on
 the library itself
bool CARRIER_CASE = false;

const int GREY = 4, RED = 3, BLUE = 2, GREEN = 1, OFF = 0;  //values for LED
 indicator when online

char ssid[] = SECRET_SSID;      // your network SSID (name)
char pass[] = SECRET_PASS;   // your network password
char weather[] = SECRET_WEATHER;
char server[] = "owner-api.teslamotors.com";  // server address
char timeserver[] = "worldtimeapi.org";   // time website
char weatherserver[] = "api.openweathermap.org";   // time website
int port = 443;
float temperature, humidity;
String  now, date;
int loops = 0;
String line = "";

WiFiSSLClient client;
int status = WL_IDLE_STATUS;

void(* resetFunc) (void) = 0; //declare reset function @ address 0

void setup() {
  Serial.begin(9600);
  //Wait to open the Serial monitor to start the program and see details on
   errors
//  while (!Serial);
  delay(1000);

  carrier.begin();
  connectWiFi();
  loops += 1;
}

void loop() {
//    carrier.display.fillScreen(ST77XX_BLACK); //oled clear()
    busyLight(RED); getTime();
    busyLight(BLUE); Battery();
    busyLight(OFF); getTemp();
    busyLight(GREEN); getOutside();
    busyLight(0);
    loops += 1;
  delay(60*1000);   // update every minute
```

```
}

void getTime() {
  if (client.connect(timeserver, port)) {
    client.println("GET /api/ip HTTP/1.1");
    client.println("Host: worldtimeapi.org");
    client.println("Content-type:application/json");
    client.println("Connection: close");
    client.println();
  }
  else {
    resetFunc();  //call reset
    return;
  }
  getReply();
  String reply = "";
  JSONVar myObject = JSON.parse(line);
  reply = myObject["datetime"];
  int start = 1 + reply.indexOf('T');  //finds location of T
  int end = reply.indexOf(':');  //finds location of hours
  end = reply.indexOf(':',end+1);  //finds location of minutes
  date = reply.substring(5, start-1);
  now = reply.substring(start, end);
  Serial.println(now);

  printIt(10, 10, 4, ST77XX_MAGENTA, now);
  printIt(150, 17, 2, ST77XX_MAGENTA, date);
}

void getTemp(){
  temperature = carrier.Env.readTemperature();
  humidity = carrier.Env.readHumidity();

  printIt(20, 150, 3, ST77XX_MAGENTA, " Tmp & Hum");
  printIt(0, 180, 3, ST77XX_WHITE, "I: " + String(temperature,1)+" &
   "+String(humidity,0));
}

void getOutside(){
  if (client.connect(weatherserver, port)) {
    client.print("GET /data/2.5/weather?q=Littleton,NH,US&appid="+
     String(SECRET_WEATHER));
    client.print("&cnt=3");
    client.println("&units=metric");
    client.println("Host: api.openweathermap.org");
    client.println("Content-type:application/json");
    client.println("Connection: close");
    client.println();
  }
  double temp = 0.0;
  double humidity = 0.0;
  String line = "";
  line = client.readStringUntil('\n');
  Serial.println(line);
  JSONVar myObject = JSON.parse(line);
  temp = myObject["main"]["temp"];
  humidity = myObject["main"]["humidity"];
  Serial.println(temp);
  Serial.println(humidity);
  Serial.println("disconnecting from server.");
  client.stop();

  printIt(0, 220, 3, ST77XX_WHITE, "O: "+String(temp,1)+" &
   "+String(humidity,0));
}

void Battery() {
  if (client.connect(server, port)) {
    // Make a HTTPS request:
    client.println("GET /api/1/vehicles/"+ String(SECRET_CAR) +
     "/data_request/charge_state HTTP/1.1");
    client.println("Host: owner-api.teslamotors.com");
    client.println("Content-type:application/json");
    client.println("Authorization:bearer " + String(SECRET_TESLA));
    client.println("Connection: close");
    client.println();
  }
  String buffer = "";
  double level = 0.0;
  double current = 0.0;
  getReply();
  JSONVar myObject = JSON.parse(line);
  level = myObject["response"]["battery_level"];
  current = myObject["response"]["charger_actual_current"];
  Serial.println(level);
  Serial.println(current);

  printIt(60, 60, 2, ST77XX_WHITE,"Tesla     " + String(loops));
  int Color = ST77XX_RED;
  if (level > 79) Color = ST77XX_GREEN;
  if (level < 20) Color = ST77XX_YELLOW;
  printIt(10, 90, 4, Color,String(level,0) + "%  "+String(current,0)+"A");

}

void connectWiFi()  {
  busyLight(GREY);
  // attempt to connect to Wifi network:
  while ( status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    // Connect to WPA/WPA2 network. Change this line if using open or WEP
     network:
    status = WiFi.begin(ssid, pass);
    // wait 10 seconds for connection:
    delay(10000);
  }
  printWifiStatus();
  busyLight(0);    // signal message received
}

void printWifiStatus() {
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your WiFi shield's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
  // print where to go in a browser:
  Serial.print("http://");
  Serial.println(ip);
  carrier.display.fillScreen(ST77XX_BLACK); //oled clear()
  carrier.display.setTextSize(2); //medium sized text
  carrier.display.setCursor(10, 10);
  carrier.display.print("IP:  ");
  carrier.display.setTextColor(ST77XX_MAGENTA);
  carrier.display.print(ip);
}

void printIt(int x, int y, int font, int Color, String text){
  // prints line in a given font - erases the previous line first
  carrier.display.fillRect(x,y,carrier.display.width()-x,
   y+font*4,ST77XX_BLACK);
  carrier.display.setTextSize(font);
  carrier.display.setCursor(x, y);
  carrier.display.setTextColor(Color);
  carrier.display.print(text);
}

void getReply(){
  //reads lines until connction closed or until 10 sec has passed
  int counter = 0;
  while (client.connected() and counter < 1000) {
    line = client.readStringUntil('\n');
    Serial.println(line);
    counter += 1;
    delay(10);
  }
  Serial.println("disconnecting from server.");
  client.stop();
  if (counter >= 1000){
    resetFunc();   // if someone is not responding - reset everything
  }
}

void busyLight(int Color){
  switch (Color){
    case BLUE:
      carrier.leds.setPixelColor(2, 0, 0, 1);
      break;
    case GREEN:
      carrier.leds.setPixelColor(2, 1, 0, 0);
      break;
    case RED:
      carrier.leds.setPixelColor(2, 0, 1, 0);
      break;
    case GREY:
      carrier.leds.setPixelColor(2, 1, 1, 1);
      break;
    case 0:  //off
      carrier.leds.setPixelColor(2, 0, 0, 0);
      break;
  }
  carrier.leds.show();
}Mo
```