LAPORAN PRAKTIKUM MINGGU KE-9

Socket Server dan Client

INTERNET OF THINGS

Disusun oleh:

Omar Abdul-Raoof Taha Ghaleb Al-Maktary

1941720237

TI-3H

D4 TEKNIK INFORMATIKA

TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2022

LAPORAN

A. PRAKTIKUM

Practicum 1:

The first practicum is to make a local server using python language utilizing the socket library.

server.py

```python
import socket
from threading import Thread


# Multithreaded Python server
class ClientThread(Thread):

    def __init__(self, ip, port):
        Thread.__init__(self)
        self.ip = ip
        self.port = port
        print("Incoming connection from " + ip + ":" + str(port))

    def run(self):
        while True:
            try:
                data = conn.recv(2048)
                if len(data) == 0:
                    break

                print("length: " + str(len(data)))
                print("Server received data:", data)
                # MESSAGE = input("Input response:")
                MESSAGE = "OK"
                conn.send(MESSAGE.encode("utf8"))  # echo
            except Exception as e:
                print(e)
                break

```

```
PROBLEMS 2  OUTPUT  TERMINAL  DEBUG CONSOLE

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\IoT\PlatformIO IDE\vs-program6> & C:/Python310/python.exe "d:/IoT/PlatformIO IDE/vs-program6/server.py"
Server started on 192.168.56.1 port 3000
```

We can see the permissions that the server file has in the machine:

```
OMAR@LAPTOP-N1SUC5AB MINGW64 /d/IoT/PlatformIO IDE/vs-program6
$ ls -lh
total 5.0K
drwxr-xr-x 1 OMAR 197609    0 May 17 07:51 include/
drwxr-xr-x 1 OMAR 197609    0 May 17 07:51 lib/
-rw-r--r-- 1 OMAR 197609  452 May 17 07:51 platformio.ini
-rw-r--r-- 1 OMAR 197609 1.3K May 17 08:25 server.py
drwxr-xr-x 1 OMAR 197609    0 May 17 07:51 src/
drwxr-xr-x 1 OMAR 197609    0 May 17 07:51 test/
```

This is the telnet test on the IP address and port of the socket

```
Telnet 192.168.56.1
OKoOK
```

```
PS D:\IoT\PlatformIO IDE\vs-program6> & C:/Python310/python.exe "d:/IoT/PlatformIO IDE/vs-program6/server.py"
Server started on 192.168.56.1 port 3000
Incoming connection from 192.168.56.1:65396
Server started on 192.168.56.1 port 3000
length: 1
Server received data: b'h'
length: 1
Server received data: b'o'
```

Build the ESP8266 project so it can send message to the socket server.

```
const uint16_t port = 3000;
const char *host = "192.168.56.1";

void connect_wifi()
{
  Serial.printf("Connecting to %s ", ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println(" connected");
}

void connect_server()
{
  WiFiClient client;

  Serial.printf("\n[Connecting to %s ... ", host);
  if (client.connect(host, port))
  {
```

```
PROBLEMS 7  OUTPUT  TERMINAL  DEBUG CONSOLE

Writing at 0x0000c000... (30 %)
Writing at 0x00010000... (38 %)
Writing at 0x00014000... (46 %)
Writing at 0x00018000... (53 %)
Writing at 0x0001c000... (61 %)
Writing at 0x00020000... (69 %)
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 279104 bytes (204472 compressed) at 0x00000000 in 18.1 seconds (effective 123.6 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
========================================================================== [SUCCESS] Took 23.67 seconds ==========================================================================

Terminal will be reused by tasks, press any key to close it.
```

ESP8266 connected successfully to the WIFI and the Server

```
Connecting to P.TRIS ..... connected

[Connecting to 192.168.100.12 ... connected]
[Sending a request]
[Response:]
OK
```

```
PS D:\IoT\PlatformIO IDE\vs-program6> & C:/Python310/python.exe "d:/IoT/PlatformIO IDE/vs-program6/server.py"
Server started on 192.168.100.12 port 3000
Incoming connection from 192.168.100.70:50416
Server started on 192.168.100.12 port 3000
length: 16
Server received data: b'Hai from ESP8266'
Incoming connection from 192.168.100.70:63850
Server started on 192.168.100.12 port 3000
length: 16
```

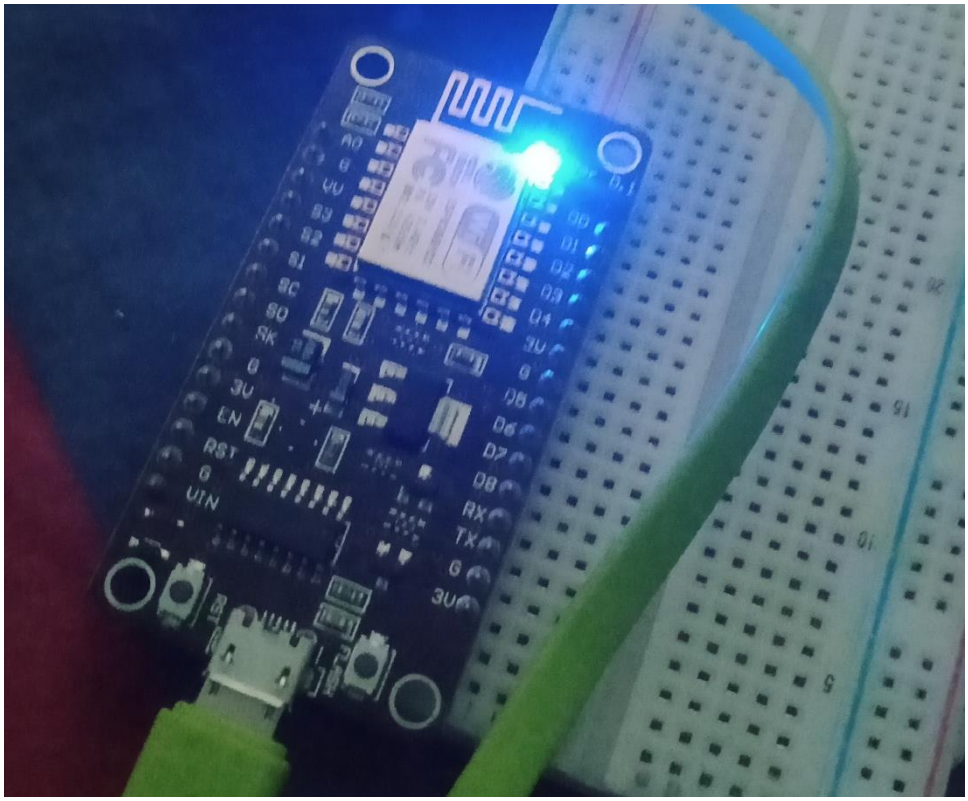
Practicum 2:

In the second practicum, we learn how we can send information form the server to ESP8266 so it can perform some tasks. As shown in the following pictures, the server sent a string contains “led-on” which will be received by ESP8266 and the light the built-in led light.

```python
import socket
from threading import Thread
from time import sleep


# Multithreaded Python server
class ClientThread(Thread):

    def __init__(self, ip, port):
        Thread.__init__(self)
        self.ip = ip
        self.port = port
        print("Incoming connection from " + ip + ":" + str(port))

    def run(self):
        while True:
            try:
                MESSAGE = input("Input response:")
                conn.send(MESSAGE.encode("utf8"))  # echo
            except Exception as e:
                print(e)
                break
            sleep(0.25)
```

```
[Connecting to 192.168.100.12 ... connected]
[Response:]led-on
```

```
PS D:\IoT\PlatformIO IDE\vs-program6> & C:/Python310/python.exe "d:/IoT/PlatformIO IDE/vs-program6/server.py"
Server started on 192.168.100.12 port 3000
Incoming connection from 192.168.100.70:53430
Server started on 192.168.100.12 port 3000
Input response:led-on
```



TUGAS

In the assignment, I made the code so it can connect to the WIFI, in order to get the current time. The system allows the server user to send message contains instruction to be performed on the ESP8266.

Code:

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

#define RED_LED D5
#define GREEN_LED D6
#define BLUE_LED D7

#define sensorLDR D0
int lightIntensity;

int lcdColumns = 16;
int lcdRows = 2;
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

#define DHTType DHT11
DHT dht(D1, DHTType);
String tempNow = "temp";

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");
String timeNow = "time";
String weekDays[7] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"};
String months[12] = {"January", "February", "March", "April", "May", "June",
"July", "August", "September", "October", "November", "December"};

const char *ssid = "P.TRIS";
const char *password = "mahmuditri";
const uint16_t port = 3000;
const char *host = "192.168.100.12";

WiFiClient client;

void connect_wifi()
{
  Serial.printf("Connecting to %s ", ssid);
```

```
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println(" connected");
  delay(250);
}

void connect_server()
{
  while (!client.connect(host, port))
  {
    Serial.printf("\n[Connecting to %s ... ", host);
    delay(1000);
    return;
  }
  Serial.println("connected]");
  delay(1000);
}

void setup()
{
  Serial.begin(9600);

  Wire.begin(D2, D3);
  lcd.init();
  lcd.backlight();

  pinMode(RED_LED, OUTPUT);
  pinMode(GREEN_LED, OUTPUT);
  pinMode(BLUE_LED, OUTPUT);

  connect_wifi();
  connect_server();

  timeClient.begin();
  timeClient.setTimeOffset(25200);

  delay(3000);
}

void getTemp()
{
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);
  if (isnan(t) || isnan(f))
```

```
  {
    lcd.setCursor(0, 1);
    lcd.print("Err Temp");
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  tempNow = String(t) + char(0b11011111) + "C " + String(f) + char(0b11011111)
+ "F";
  Serial.println(tempNow);
  lcd.setCursor(0, 1);
  lcd.print(tempNow);
}

void getTime()
{
  timeClient.update();
  time_t epochTime = timeClient.getEpochTime();
  String formattedTime = timeClient.getFormattedTime();
  int currentHour = timeClient.getHours();
  int currentMinute = timeClient.getMinutes();
  // int currentSecond = timeClient.getSeconds();
  String weekDay = weekDays[timeClient.getDay()];
  struct tm *ptm = gmtime((time_t *)&epochTime);
  int monthDay = ptm->tm_mday;
  int currentMonth = ptm->tm_mon + 1;
  String currentMonthName = months[currentMonth - 1];
  int currentYear = ptm->tm_year + 1900;
  String currentDate = String(monthDay) + "-" + String(currentMonth) + "-" +
String(currentYear);

  timeNow = currentDate + " " + String(currentHour) + ":" +
String(currentMinute);
  Serial.println(timeNow);
  lcd.setCursor(0, 1);
  lcd.print(timeNow);
}

void getLightIntensity()
{
  lightIntensity = analogRead(sensorLDR);
  lcd.setCursor(0, 1);
  if (lightIntensity == 0)
  {
    lcd.print("Bright Light:Day");
  }
  else
  {
    lcd.print("Dim Light:Night");
```

```
  }
}
void allLedOff(String message)
{
  digitalWrite(RED_LED, HIGH);
  digitalWrite(GREEN_LED, HIGH);
  digitalWrite(BLUE_LED, HIGH);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void allLedOn(String message)
{
  digitalWrite(RED_LED, LOW);
  digitalWrite(GREEN_LED, LOW);
  digitalWrite(BLUE_LED, LOW);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void redLightOn(String message)
{
  digitalWrite(RED_LED, LOW);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void redLightOff(String message)
{
  digitalWrite(RED_LED, HIGH);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void greenLightOn(String message)
{
  digitalWrite(GREEN_LED, LOW);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void greenLightOff(String message)
{
  digitalWrite(GREEN_LED, HIGH);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void blueLightOn(String message)
{
  digitalWrite(BLUE_LED, LOW);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
```

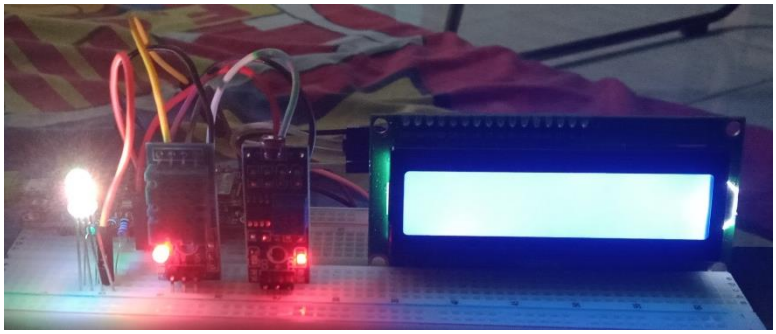
```
void blueLightOff(String message)
{
  digitalWrite(BLUE_LED, HIGH);
  lcd.setCursor(0, 1);
  lcd.print(message);
}
void loop()
{
  delay(1000);
  if (client.connected())
  {
    Serial.print("[Response:]");
    String line = client.readStringUntil('\n');
    Serial.println(line);
    if (line.equalsIgnoreCase("temp"))
    {
      lcd.clear();
      getTemp();
    }
    else if (line.equalsIgnoreCase("time"))
    {
      lcd.clear();
      getTime();
    }
    else if (line.equalsIgnoreCase("light"))
    {
      lcd.clear();
      getLightIntensity();
    }
    else if (line.equalsIgnoreCase("led off"))
    {
      lcd.clear();
      allLedOff("Shut down LEDs");
    }
    else if (line.equalsIgnoreCase("led on"))
    {
      lcd.clear();
      allLedOn("Turn on LEDs");
    }
    else if (line.equalsIgnoreCase("red on"))
    {
      lcd.clear();
      redLightOn("Red LED On");
    }
    else if (line.equalsIgnoreCase("red off"))
    {
      lcd.clear();
      redLightOff("Red LED Off");
```

```
    }
    else if (line.equalsIgnoreCase("green on"))
    {
      lcd.clear();
      greenLightOn("Green LED On");
    }
    else if (line.equalsIgnoreCase("green off"))
    {
      lcd.clear();
      greenLightOff("Green LED Off");
    }
    else if (line.equalsIgnoreCase("blue on"))
    {
      lcd.clear();
      blueLightOn("Blue LED On");
    }
    else if (line.equalsIgnoreCase("blue off"))
    {
      lcd.clear();
      blueLightOff("Blue LED Off");
    }
    else if (line.equalsIgnoreCase("clear screen"))
    {
      lcd.clear();
    }
  }
  else
  {
    connect_server();
  }
  delay(250);
}
```

Results

```
PS D:\IoT\PlatformIO IDE\vs-program6> & C:/Python310/python.exe "d:/IoT/PlatformIO IDE/vs-program6/server.py"
Server started on 192.168.100.12 port 3000
Incoming connection from 192.168.100.70:50852
Server started on 192.168.100.12 port 3000
Input response:time
Input response:led off
Input response:light
Input response:light
Input response:blue onn
Input response:blue on
Input response:temp
```

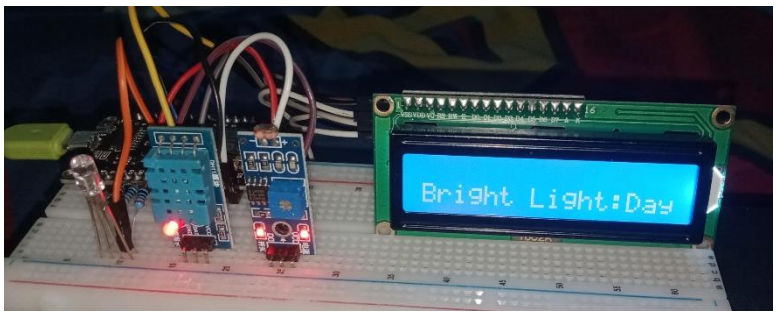
Initial state:



Get time function:

Shutdown all LEDs:

Check light intensity:



Turn blue light on:

Get temperature: as usual, there are some errors regarding DHT11.