FINAL PROJECT SOFTWARE ENGINEERING

AIR QUALITY

BY GROUP:

Aufa Dhiya Aydan	**1706043260**
Muhamad Fadil	**1706042812**
Muhammad Farhan	**1706042876**

TEKNIK KOMPUTERFAKULTAS TEKNIK

UNIVERSITAS INDONESIA

TABLE OF CONTAINS

FOREWORD

We give our thanks to the Almighty God, Allah SWT for his blessings, taufiq, and inayah which bestowed His grace in the form of an opportunity to make an RPL report so that it could be completed on time.

In this work, do not forget we also thank our supervisor DR. Ruki Harwahyu S.T, M.T, M.Sc and Prof. Ir. Riri Fitri Sari M.M, M.Sc and also friends who helped us contribute by providing input in the form of bright ideas so that this paper can be arranged well and neatly.

We hope that this report can add insight and knowledge to those who read it. We apologize profusely because this report is far from perfect, so we look forward to criticism and suggestions for us that are constructive for the creation of further papers that are even better.

CHAPTER I

Introduction

1.1 Background

Air is a mixture of gases that has various elements in it and has a large impact on the survival of living that exists on the entire surface of the earth. Because of that, we must pay attention to maintain sustainability the air around us with finding out if the air quality is still suitable for us or already in the critical zone.

Why we must know the air quality of an area? Because, the area with bad air quality can cause dangerous diseases such as skin cancer, *asma, penyakit paru obstruktif kronis (PPOK) , infeksi saluran pernapasan atas (ISPA),*and death because heart disease.

Cause of bad air quality is a vehicle emission with petroleum fuel. The petroleum fuel contains hydrocarbon compounds which are then burned into carbon dioxide (CO2) and water (H2O). the incomplete combustion results from the vehicle, producing carbon monoxide (CO) which is a poisonous gas, and then nitrogen oxides and volatile organic compounds will be pollutants, and then the gas will affect air quality.

From that’s a problem, we want to build software that can be used to determine the pros and cons of air for our bodies (all the living things in the world). We want to combine the software with hardware tools to detect air quality. The output from hardware becomes the input for software. We build the software with HTML, CSS, PHP programming language and the hardware is esp32 and MQ135 sensor.

1.2 Formulation of The Problem

- what is air quality?
- What are the cause of bad air quality?
- How to find out air quality?
- What is the impact from bad air quality?
- How to maintain good air quality?

1.3 Purpose

Air quality is measured with the Air Quality Index, or AQI. The AQI works like a thermometer that runs from 0 to 500 degrees. However, instead of showing changes in the temperature, the AQI is a way of showing changes in the amount of pollution in the air.

Because of that, we must pay attention for air quality. In the city, air pollution is caused by fumes from vehicle. This is called ground level ozone (urban smog). Ground-level ozone increases in cities when the air is still, the temperature is warm, and the sun is out. This combination traps pollution in the air. Airplanes also cause air pollution. Other things, such as construction vehicles and tobacco smoke also cause air pollution. In rural areas, outdoor air pollution often is cause by dust from tractors plowing field, trucks and cars driving on dirt or gravel roads, rock quarries, and smoke from wood and crop firs.

Bad effects from bad air quality on humans is can be broken down into short term effects and long term effects. Short-term effects, which are temporary, include illnesses such as pneumonia or bronchitis. They also include discomforts such as irritation to the nose, throat, eyes, or skin. Air pollution can also cause headaches, dizziness, and nausea. Bad smells made by factories, garbage, or sewer systems are considered air pollution, too. These odors are less serious but still unpleasant. Long-term effects of air pollution can last for years or an entire lifetime. They can even lead to a person's death. Long-term health effects from air pollution include heart disease, lung cancer, and respiratory diseases such as emphysema. Air pollution can also cause long-term damage to people's nerves, brain, kidneys, liver, and other organs. Some scientists suspect air pollutants cause birth defects. Nearly 2.5 million people die worldwide each year from the effects of outdoor or indoor air pollution. For effects on the environment, this can kill crops or reduce their yield. It can kill young trees and other plants. That’s 10 easy steps for cleaner air quality, such as:

1. Walk, bike, carpool, or take public transit
2. Reduce your heating needs by making your house more energy efficient
3. Say no to backyard burning
4. Use hand-powered garden tools
5. Say no to gasoline or diesel powered
6. Check your tire pressure to increase fuel efficiency

7. Reduce reuse recycle
8. Be idle free
9. Garden without pesticides
10. Get involved to support national and local efforts to clean up the air

1.4 Benefit

1. The data collected from air quality monitoring helps us assess impacts caused by poor air quality on public health air
2. Air quality data helps us determine if an area is meeting the air quality standards devised by WHO
3. The data collected from air quality monitoring would primarily help us identify polluted areas, the level of pollution and air quality level.
4. Air quality monitoring would assist in determining if air pollution control programmer devised in a locality are working efficiently or not
5. Air quality data helps us understand the mortality rate of any location due to air pollution. We can also assess and compare the short term and long term diseases/disorders which are a result of air pollution
6. Based upon the data collected control measures can be devised for protection of environment and health of all living organisms.
7. Reducing air pollution helps tackle climate change
8. Taking coherent action from local to global
9. Information key to minimize exposure

CHAPTER II

Project Management

2.1 4P

People:
- Developer: 3 members
- Supervisor: 1 Lecturer

Product
- Scope: FTUI Campus
- Objective: FTUI students
- Cost: 0% for using opensource (thingsboard)
- estimated time: 9 weeks

Process:
By using the Waterfall method.
- requairment: required software specifications (device, mobile apps, web)
- implementation: used to detect CO2 levels around the FTUI
- testing: implemented around FTUI whether it suits the needs (custemer is still a developer not yet a FTUI resident)
- maintenance: updating software according to existing needs or changing specifications so that the software can be optimized

Project:
- Projects must be accompanied by a project manager
- Projects that are made can be estimated when the completion will be on time. The group work we want is good student chemistry and regular inter-member discussions are held via social media (line) and with a meating system 2 or 3 times per week.

2.2 Features:

a. Web thinger.io
b. CO2 detection device
c. Mobile apps
d. Data from device is stored at thinger.io

3. Tools:
a. MQ135 sensor
b. ESP 32
c. Thinger.io
d. Breadboard

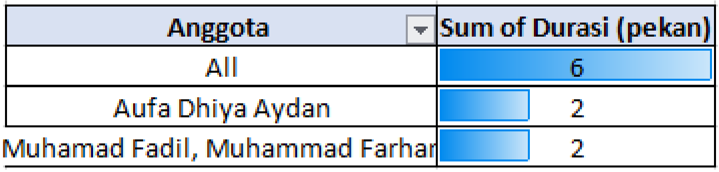
- Software Cost

Cost: 0% for using opensource (thingsboard)

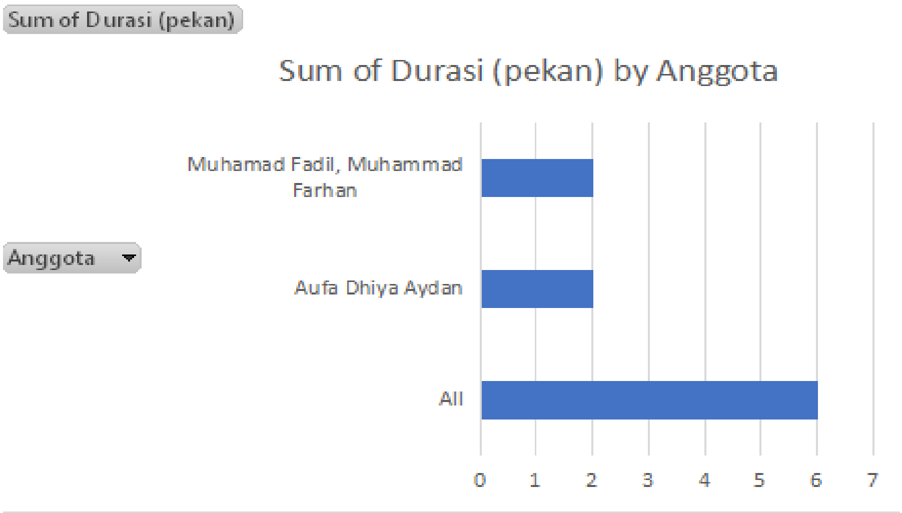
3.4 Project Planning Table

Starting time	Time's up	Duration	Division of tasks	Description	Member
4-Oct-19	18 Oktober	2	Requirment dan analisa	Understand the work process of the project created and the implementation on the web and apk	Aufa Dhiya Aydan
19-Oct-19	2 November	2	Disain program	Creating hardware and software project designs	Muhamad Fadil, Muhammad Farhan
3-Nov-19	16-Nov-19	2	Implementasi dan tes unit	Hardware programming and implementation on the web or apk and testing programs	All
17-Nov-19	30-Nov-19	2	Integrasi dan tes unit	Test programs that have been implemented in a cloud system (AI database)	All
1-Dec-19	14-Dec-19	2	Maintain	Application of the program functionally in places around the UI	All

- Graph



Anggota	Sum of Durasi (pekan)
All	6
Aufa Dhiya Aydan	2
Muhamad Fadil, Muhammad Farhar	2



- Time

  Time/Cost per 100 LoC
  Planning 2 hr / (1 person)
  Overview 1 hr / (1 person)
  Preparation 2 hr / (3 people)
  Inspection meeting 2 hr / (3 people)
  Re-work 2 hr / (3 people)
  Analysis 2 hr / (3 people)
  Total: approx. 11 - 14 person-hours

- Plan Project
  Project Plan is a project plan that uses
  Programming Languages: PHP, HTML, CSS, Arduino
  Repository: github
  Framework: -
  IDEA : -
  Cloud: We use a database from Thingsboard that already provides an API

- CCN Method
  In the CNN method, we have just entered level 2, which is Repentable where there is a simple project management such as determining the cost and determining the work schedule that will be the starting point of success

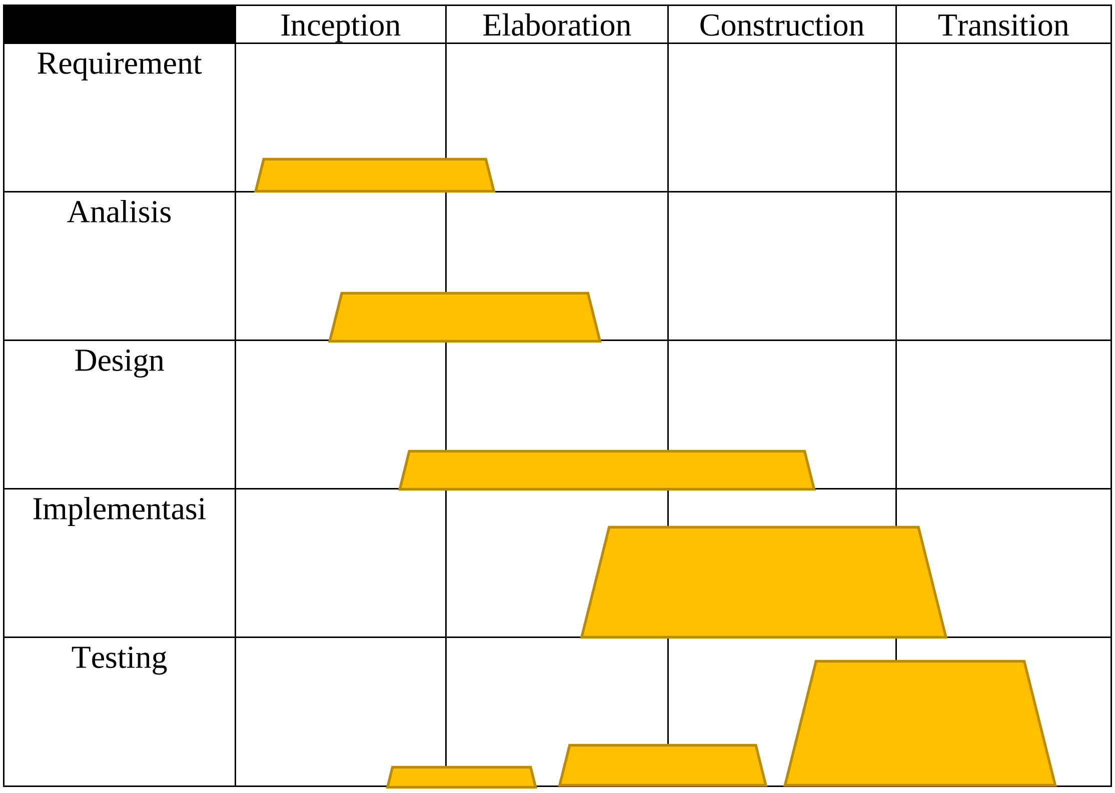
- Focus: Project Management
  OBJECT ORIENTED PROGRAMMING

Gas Sensor (Object)
Air Quality (Class)
Pin (Property or State)
Number of Pin (State)

- Prosedur Dokumentation Project (soon)
  A. Verfication and Validation (SVVP)
  B. Quality Assurance (SQAP)
  C. Configuration (SCMP)
  D. Project Status (SPMP)
  E. Requirement (SRS)
  F. Design (SDD) & Code > Source Code

- Waterfall

  We use the waterfall method in making this project because in this project we take a systematic approach and are carried out sequentially, so there are steps that are carried out sequentially (working on developing the model one by one), such as what should be done (planning), then how the modeling, construction, and how the system to its users.



A. Requirement Analysis

The Air Quality Application Program that we created is designed to determine the condition of the air around the FT environment by detecting CO2. Users only need to open the platform that we provide, then there will be a display of information about CO2 levels in every place that we install the air quality tool, in that platform we will also display information on whether the CO2 level has reached a good or bad stage in the environment.

• User page
A1. The user only sees the available page interface

• Admin page
B1. Admin can update CO2 levels
B2. Admin can update whether the CO2 condition is good enough or bad enough

B. Code
This Air Quality application project was designed using the Web (php, css, html), App (the application's own language), and Thingsboard (Arduino).

C. Support

All pc specifications are all biased

- Case tool

CASE TOOL> tools on computer devices that aim to support one or more software engineering activities in the software development process.

UPPER CASE> Case tools designed to support project planning, identification and selection. For the project, we use php, htlm, css for the web and the thingsboard platform

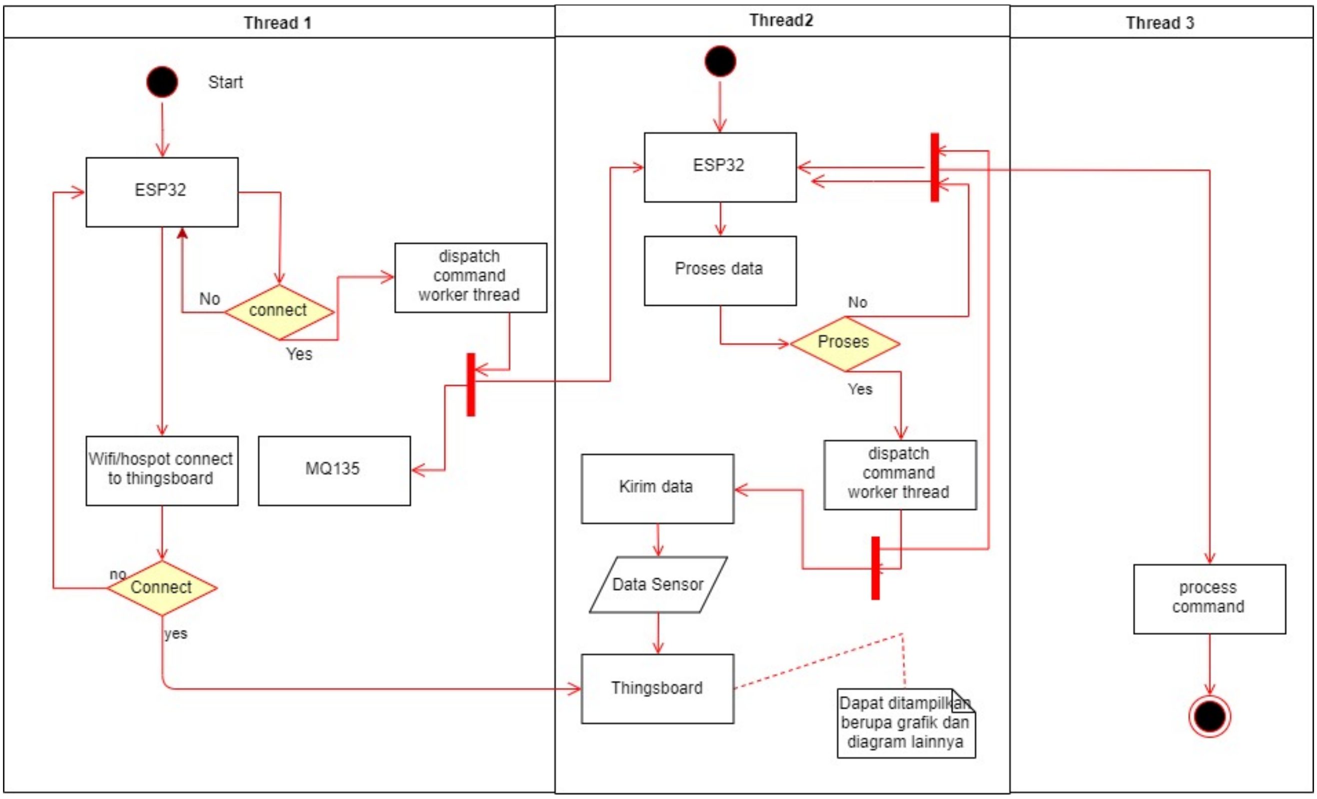
LOWER CASE> CASE tools designed to support the implementation and maintenance stages of SDLC. For the project we don't use the framework because we use thingsboard

CHAPTER III

DESIGN AND ANALYSES

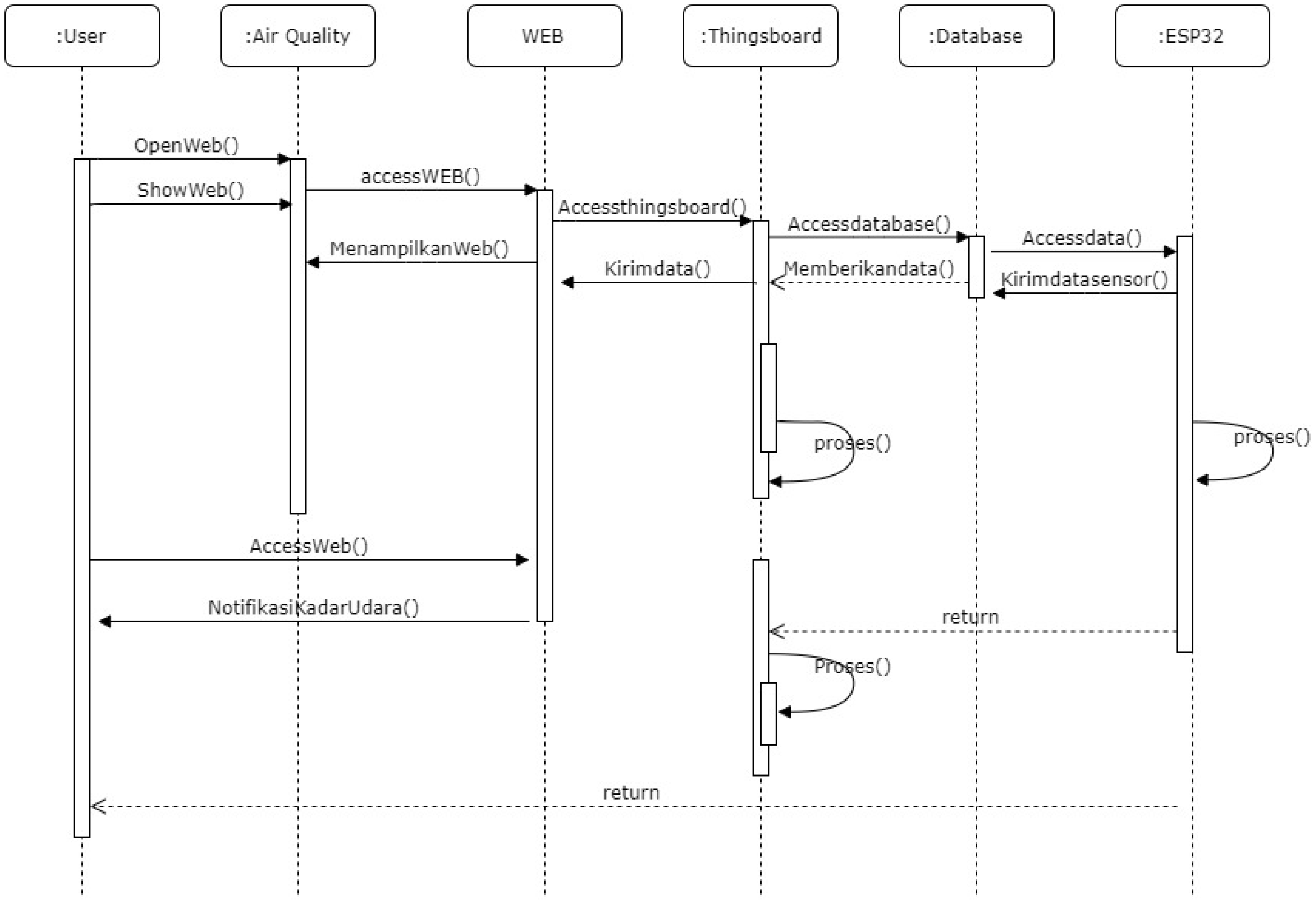
Unified Modeling Language (UML) is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex system. The UML is important part of developing object-oriented software and the software development process. The following is a sample about UML, such as:

a. Active-UML


Thread 1
Thread2
Thread 3
Start
ESP32
connect
No
Yes
dispatch command worker thread
Wifi/hospot connect to thingsboard
MQ135
no
Connect
yes
ESP32
Proses data
No
Proses
Yes
dispatch command worker thread
Kirim data
Data Sensor
Thingsboard
Dapat ditampilkan berupa grafik dan diagram lainnya
process command

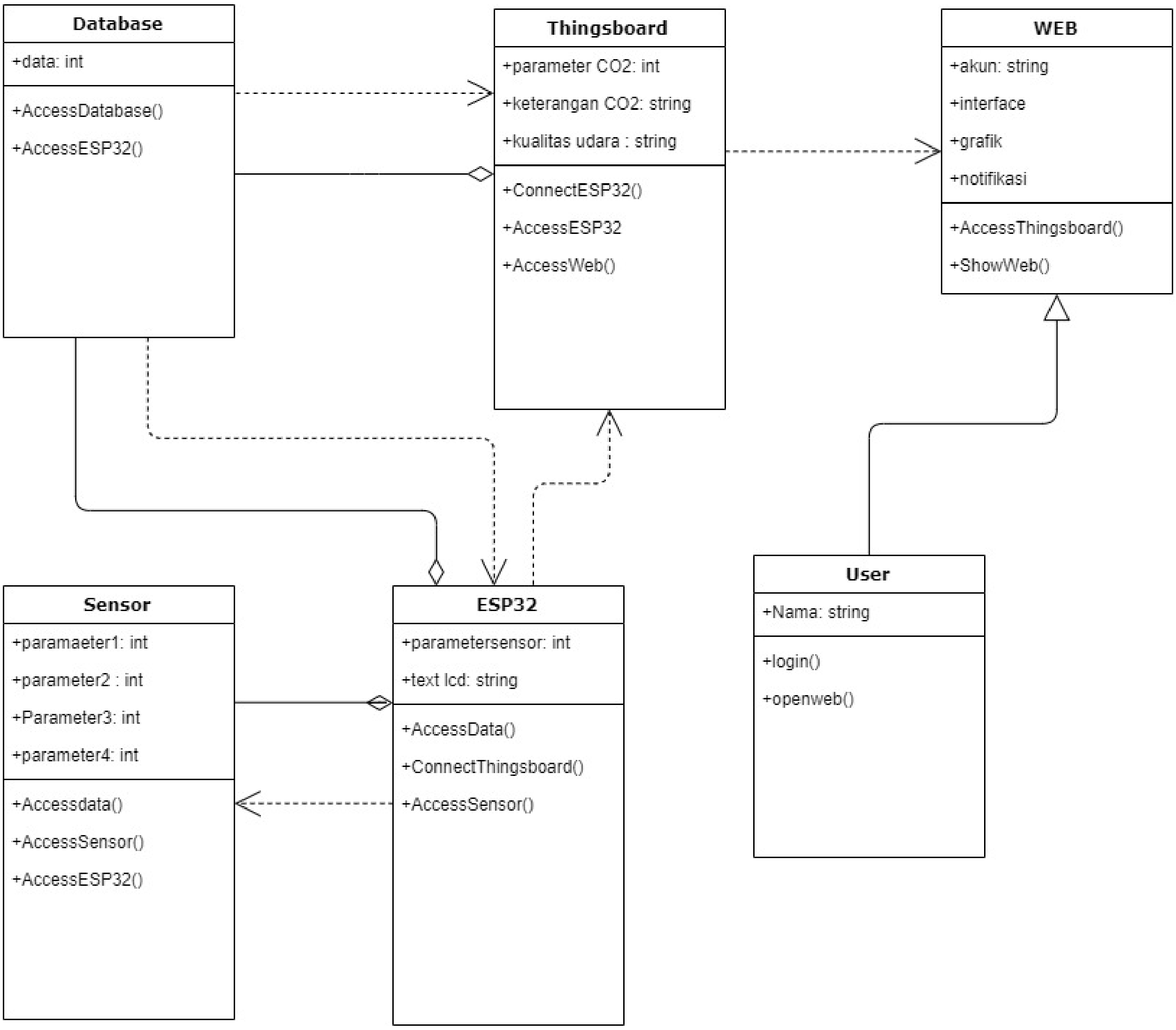
Describes a series of streams of activities, used to describe activities that are formed in an operation so that it can also be used for other activities such as use cases or interactions.

b. Sequence-UML



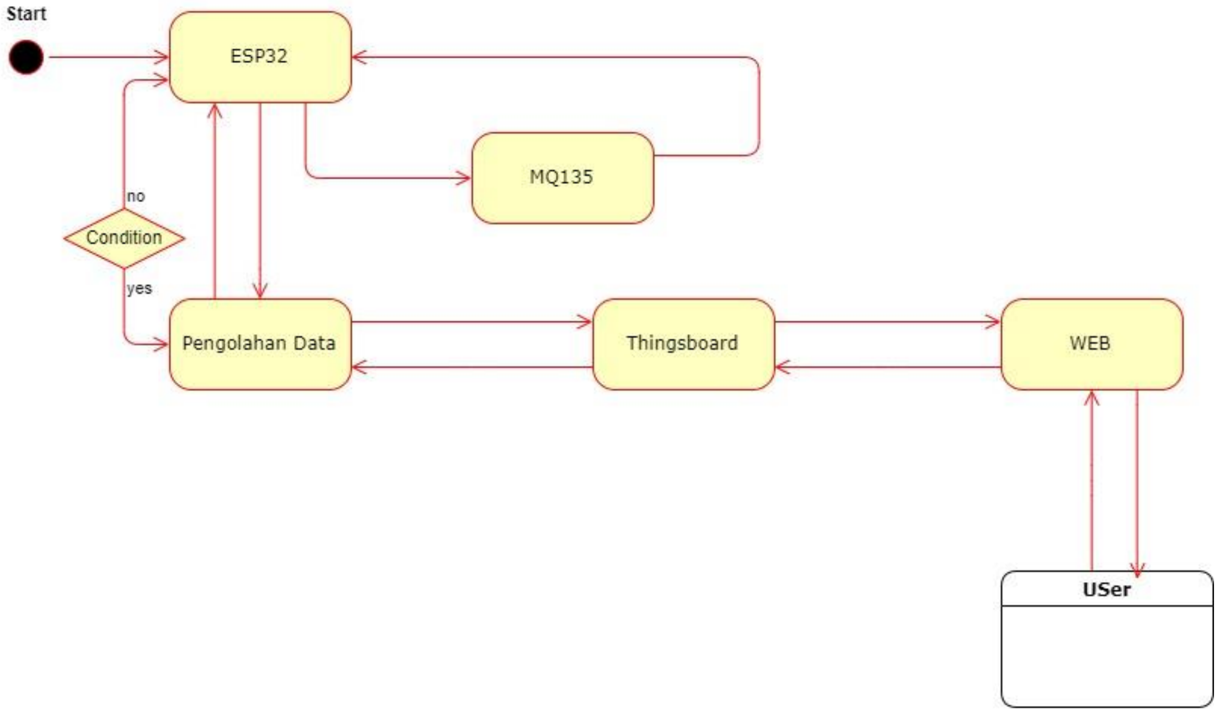
Describes dynamic collaboration between several objects. Its purpose is to show the sequence of messages sent between objects as well as interactions between objects, something that happens at a certain point in the system's execution.

c. Case-UML



Describe the static structure of classes in the system. The class represents something that is handled by the system. Classes can be related to each other in various ways: associated (connected), dependent (one class depends on / uses another class), specialed (one class is a specialization of other classes), or package (groups together as one unit). A system usually has several class diagrams.

d. State-UML



Describes several external actors and their relationship to the use case provided by the system. The use case is a description of the function provided by the system in text form as documentation of the use case symbol but can also be done in the activity diagram. Use cases are described only those seen from outside by the actor (the state of the system environment the user sees) and not how the functions are in the system.

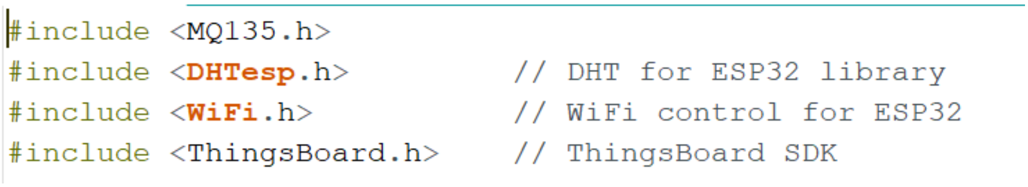
CHAPTER IV

IMPLEMENTATION PROGRAM

4.1 Code Program

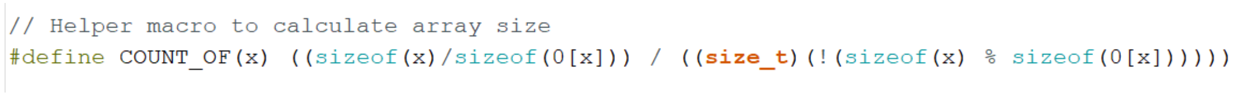
- Arduino code

Library is used



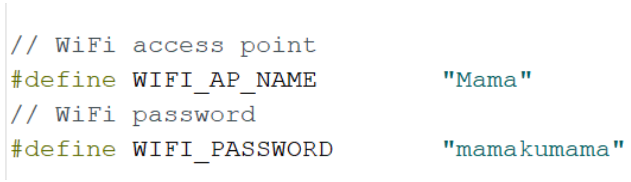
```
#include <MQ135.h>
#include <DHTesp.h>           // DHT for ESP32 library
#include <WiFi.h>             // WiFi control for ESP32
#include <ThingsBoard.h>      // ThingsBoard SDK
```

Counting array



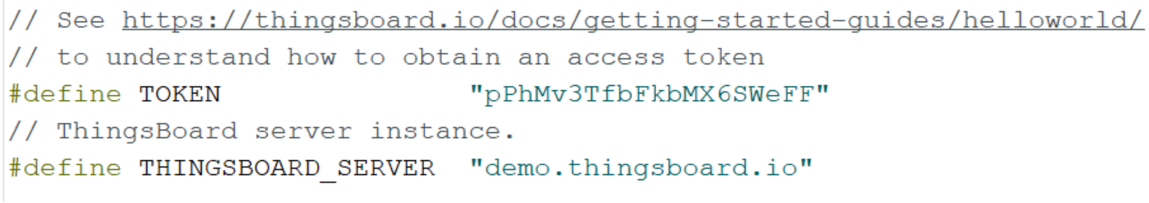
```
// Helper macro to calculate array size
#define COUNT_OF(x) ((sizeof(x)/sizeof(0[x])) / ((size_t)(!(sizeof(x) % sizeof(0[x])))))
```

SSID and SSID’s Password



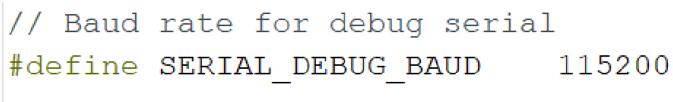
```
// WiFi access point
#define WIFI_AP_NAME        "Mama"
// WiFi password
#define WIFI_PASSWORD       "mamakumama"
```

Connect to Thingsboard.io



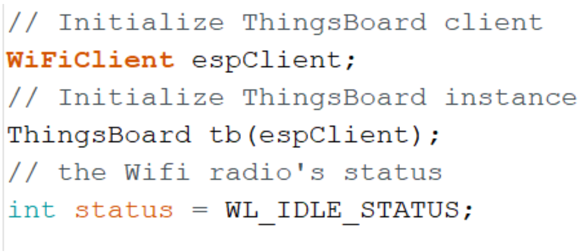
```
// See https://thingsboard.io/docs/getting-started-guides/helloworld/
// to understand how to obtain an access token
#define TOKEN               "pPhMv3TfbFkbMX6SWeFF"
// ThingsBoard server instance.
#define THINGSBOARD_SERVER  "demo.thingsboard.io"
```

Baud Rate for debugging



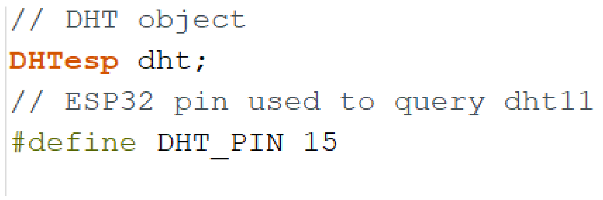
```
// Baud rate for debug serial
#define SERIAL_DEBUG_BAUD    115200
```

Initialize client with thingsoard



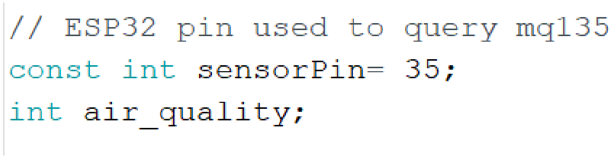
```
// Initialize ThingsBoard client
WiFiClient espClient;
// Initialize ThingsBoard instance
ThingsBoard tb(espClient);
// the Wifi radio's status
int status = WL_IDLE_STATUS;
```

Pin for DHTT11



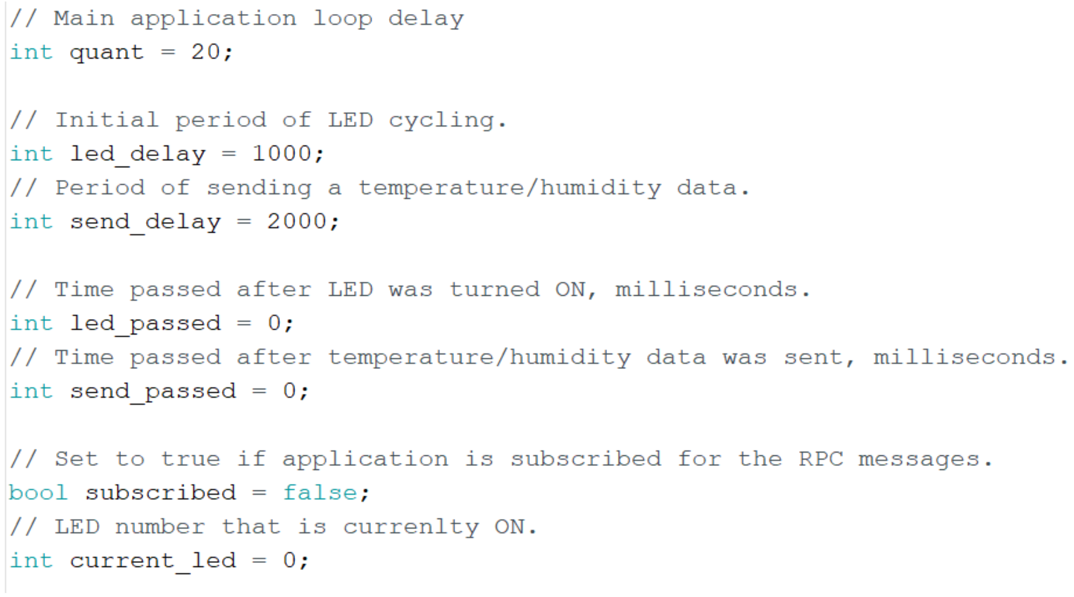
```
// DHT object
DHTesp dht;
// ESP32 pin used to query dht11
#define DHT_PIN 15
```

Pin For MQ135



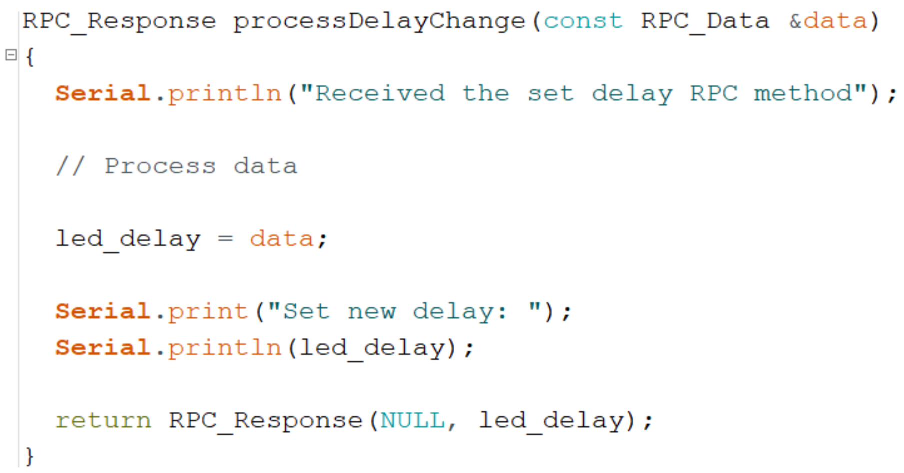
```
// ESP32 pin used to query mq135
const int sensorPin= 35;
int air_quality;
```

Main menu Code

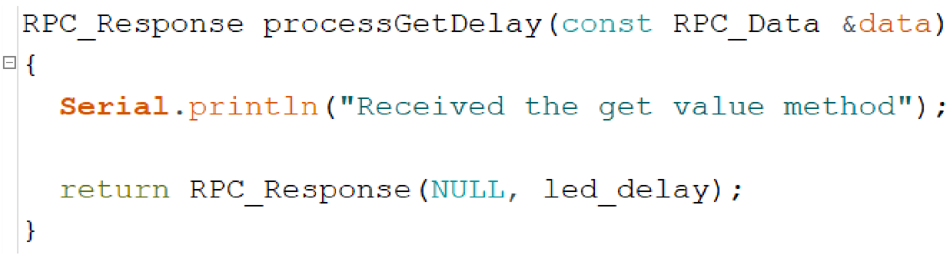


```
// Main application loop delay
int quant = 20;

// Initial period of LED cycling.
int led_delay = 1000;
// Period of sending a temperature/humidity data.
int send_delay = 2000;

// Time passed after LED was turned ON, milliseconds.
int led_passed = 0;
// Time passed after temperature/humidity data was sent, milliseconds.
int send_passed = 0;

// Set to true if application is subscribed for the RPC messages.
bool subscribed = false;
// LED number that is currenlty ON.
int current_led = 0;
```

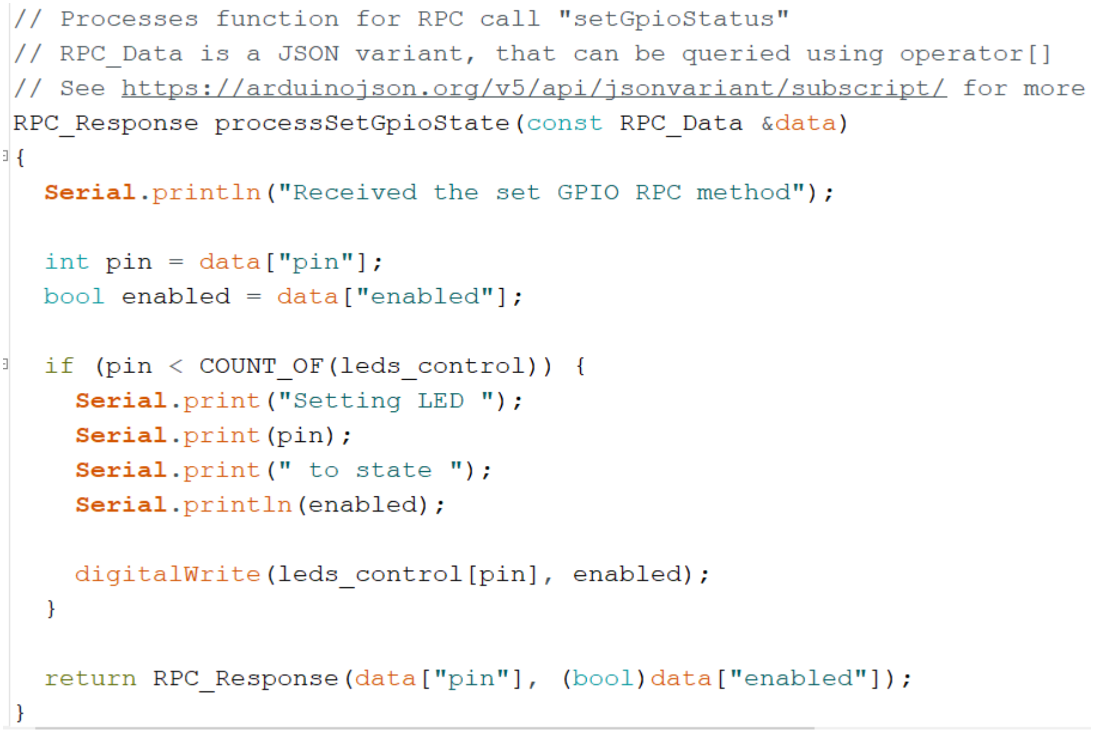
Using RPC for Send the data with Json



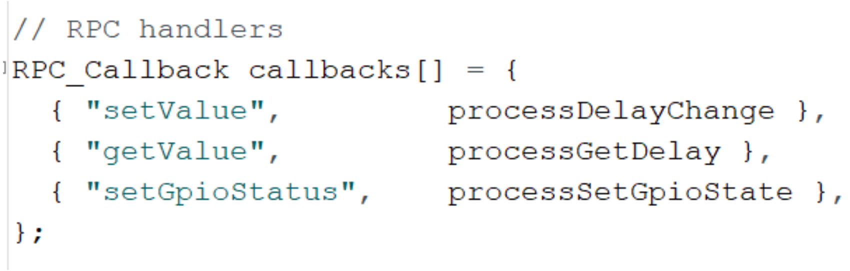
```
RPC_Response processDelayChange(const RPC_Data &data)
{
  Serial.println("Received the set delay RPC method");

  // Process data

  led_delay = data;

  Serial.print("Set new delay: ");
  Serial.println(led_delay);

  return RPC_Response(NULL, led_delay);
}
```



```
RPC_Response processGetDelay(const RPC_Data &data)
{
  Serial.println("Received the get value method");

  return RPC_Response(NULL, led_delay);
}
```

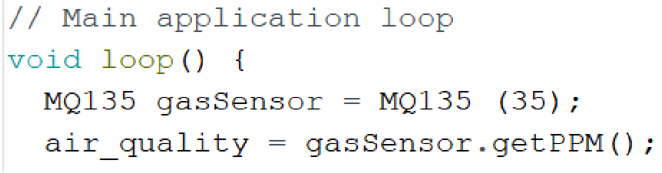


```
// Processes function for RPC call "setGpioStatus"
// RPC_Data is a JSON variant, that can be queried using operator[]
// See https://arduinojson.org/v5/api/jsonvariant/subscript/ for more
RPC_Response processSetGpioState(const RPC_Data &data)
{
  Serial.println("Received the set GPIO RPC method");

  int pin = data["pin"];
  bool enabled = data["enabled"];

  if (pin < COUNT_OF(leds_control)) {
    Serial.print("Setting LED ");
    Serial.print(pin);
    Serial.print(" to state ");
    Serial.println(enabled);

    digitalWrite(leds_control[pin], enabled);
  }

  return RPC_Response(data["pin"], (bool)data["enabled"]);
}
```

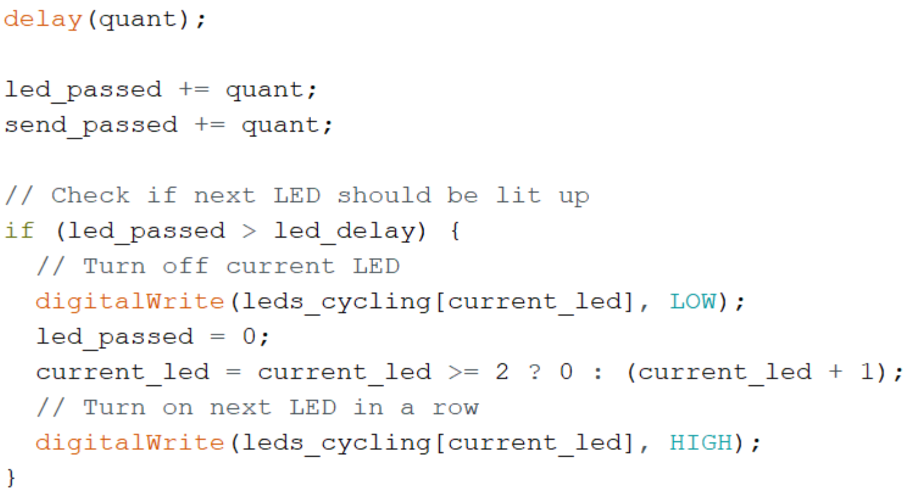


```
// RPC handlers
RPC_Callback callbacks[] = {
  { "setValue",         processDelayChange },
  { "getValue",         processGetDelay },
  { "setGpioStatus",    processSetGpioState },
};
```

Main Aplication for declare the Sensor Pin

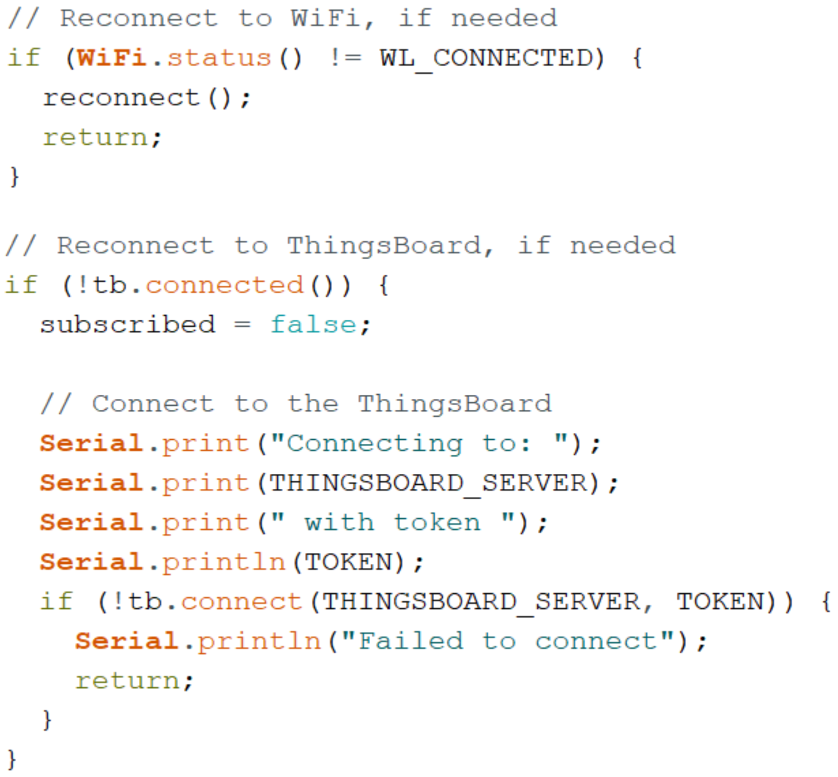


```
// Main application loop
void loop() {
  MQ135 gasSensor = MQ135 (35);
  air_quality = gasSensor.getPPM();
```

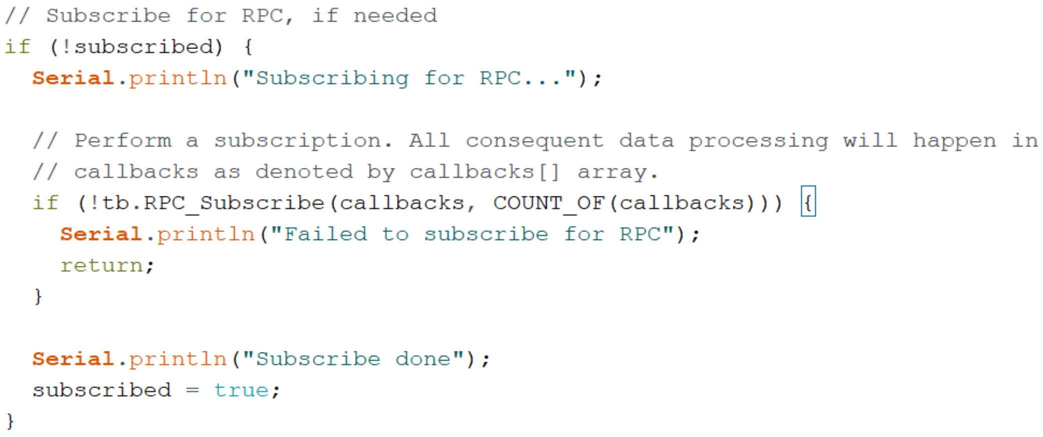


```
delay(quant);

led_passed += quant;
send_passed += quant;

// Check if next LED should be lit up
if (led_passed > led_delay) {
  // Turn off current LED
  digitalWrite(leds_cycling[current_led], LOW);
  led_passed = 0;
  current_led = current_led >= 2 ? 0 : (current_led + 1);
  // Turn on next LED in a row
  digitalWrite(leds_cycling[current_led], HIGH);
}
```

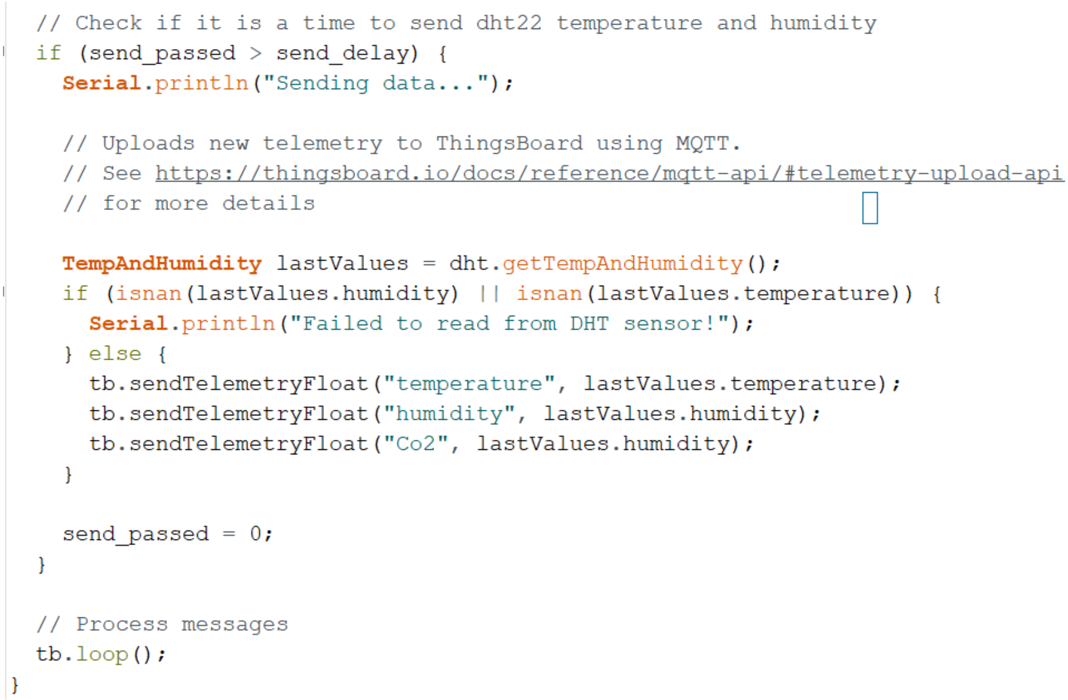
Check Wifi connect



```
// Reconnect to WiFi, if needed
if (WiFi.status() != WL_CONNECTED) {
  reconnect();
  return;
}

// Reconnect to ThingsBoard, if needed
if (!tb.connected()) {
  subscribed = false;

  // Connect to the ThingsBoard
  Serial.print("Connecting to: ");
  Serial.print(THINGSBOARD_SERVER);
  Serial.print(" with token ");
  Serial.println(TOKEN);
  if (!tb.connect(THINGSBOARD_SERVER, TOKEN)) {
    Serial.println("Failed to connect");
    return;
  }
}
```

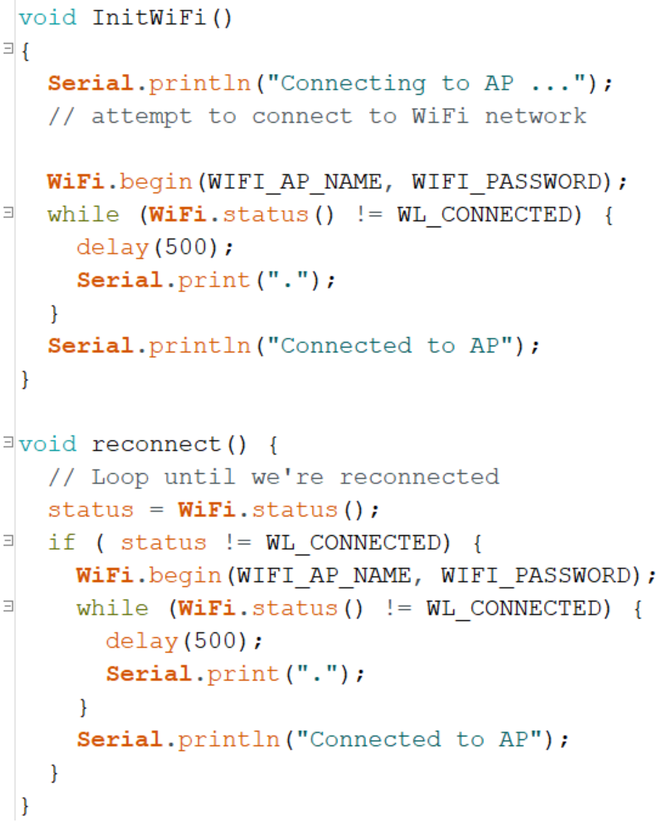


```
// Subscribe for RPC, if needed
if (!subscribed) {
  Serial.println("Subscribing for RPC...");

  // Perform a subscription. All consequent data processing will happen in
  // callbacks as denoted by callbacks[] array.
  if (!tb.RPC_Subscribe(callbacks, COUNT_OF(callbacks))) {
    Serial.println("Failed to subscribe for RPC");
    return;
  }

  Serial.println("Subscribe done");
  subscribed = true;
}
```



```
  // Check if it is a time to send dht22 temperature and humidity
  if (send_passed > send_delay) {
    Serial.println("Sending data...");

    // Uploads new telemetry to ThingsBoard using MQTT.
    // See https://thingsboard.io/docs/reference/mqtt-api/#telemetry-upload-api
    // for more details

    TempAndHumidity lastValues = dht.getTempAndHumidity();
    if (isnan(lastValues.humidity) || isnan(lastValues.temperature)) {
      Serial.println("Failed to read from DHT sensor!");
    } else {
      tb.sendTelemetryFloat("temperature", lastValues.temperature);
      tb.sendTelemetryFloat("humidity", lastValues.humidity);
      tb.sendTelemetryFloat("Co2", lastValues.humidity);
    }

    send_passed = 0;
  }

  // Process messages
  tb.loop();
}
```

Check Sending data to thingsboard

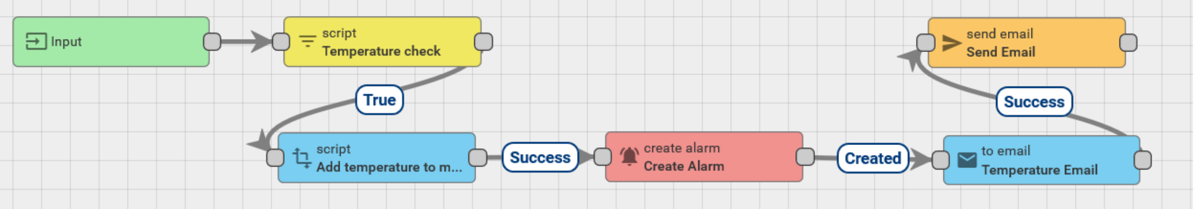


```
void InitWiFi()
{
  Serial.println("Connecting to AP ...");
  // attempt to connect to WiFi network

  WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to AP");
}

void reconnect() {
  // Loop until we're reconnected
  status = WiFi.status();
  if ( status != WL_CONNECTED) {
    WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
    Serial.println("Connected to AP");
  }
}
```

4.2 Json thingsbord

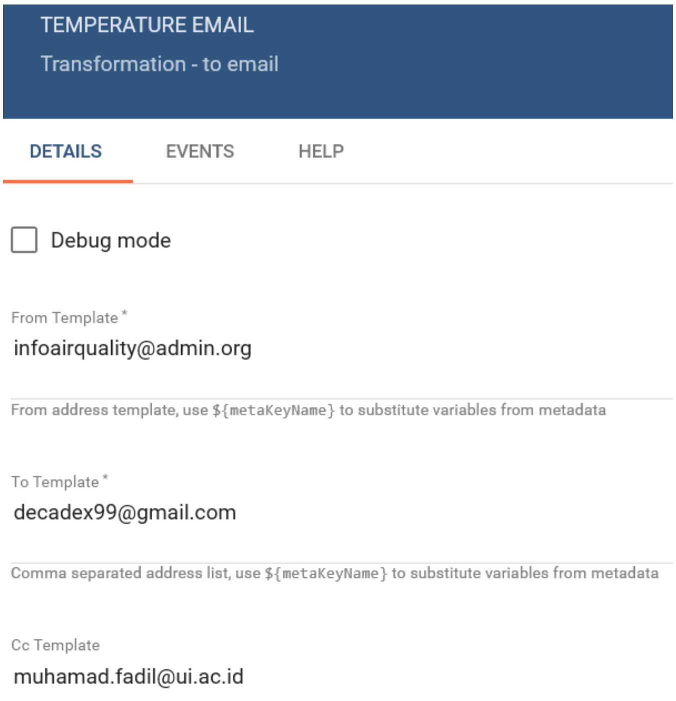
– Alarm

- Alarm for high temperature



Code per nodes

```
//----Filter-script---- //
return msg.temp > 35;
//----Transformation-script----//
metadata.temperature = msg.temperature;
return {msg: msg, metadata: metadata, msgType: msgType};
//-----Action – create alarm---- //
var details = {};
if (metadata.prevAlarmDetails) {
    details = JSON.parse(metadata.prevAlarmDetails);
}
return details;
//-----transformation - to email----//
```



TEMPERATURE EMAIL

Transformation - to email

☐ Debug mode

From Template *

infoairquality@admin.org

From address template, use ${metaKeyName} to substitute variables from metadata

To Template *

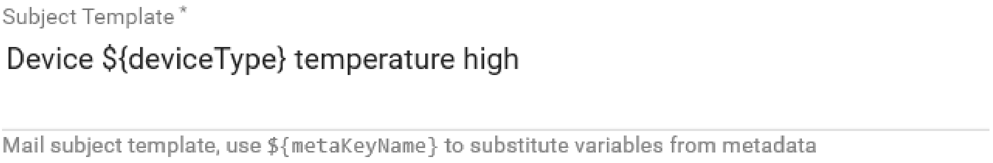
decadex99@gmail.com

Comma separated address list, use ${metaKeyName} to substitute variables from metadata

Cc Template

muhamad.fadil@ui.ac.id

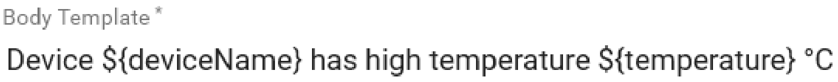
---Massage for email receiver



Subject Template *

Device ${deviceType} temperature high

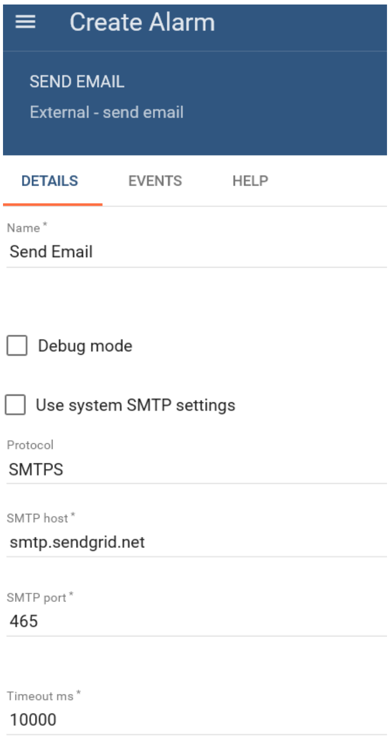
Mail subject template, use `${metaKeyName}` to substitute variables from metadata



Body Template *

Device ${deviceName} has high temperature ${temperature} °C

//----External – send email ----//



Name *
Send Email

□ Debug mode

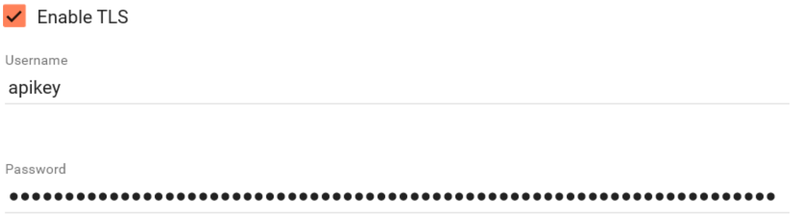
□ Use system SMTP settings

Protocol
SMTPS

SMTP host *
smtp.sendgrid.net

SMTP port *
465

Timeout ms *
10000



☑ Enable TLS

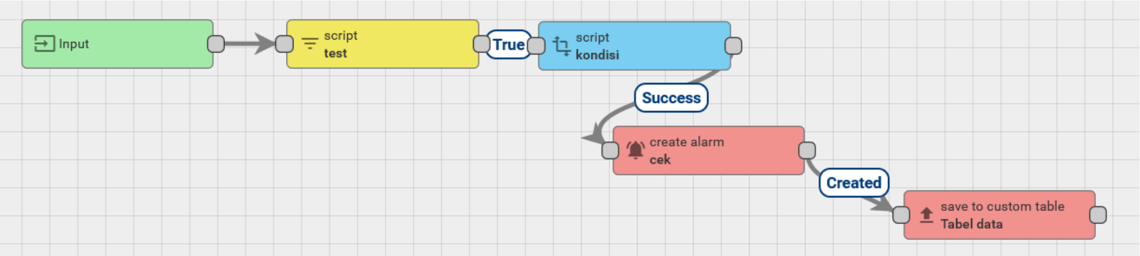
Username
apikey

Password
●●●

Full Code .json for Alarm

```
//.json code
{
  "ruleChain": {
    "additionalInfo": null,
    "name": "Create Alarm",
    "firstRuleNodeId": null,
    "root": false,
    "debugMode": false,
    "configuration": null
  },
  "metadata": {
    "firstNodeIndex": 0,
    "nodes": [
      {
        "additionalInfo": {
          "layoutX": 286,
          "layoutY": 150
        },
        "type": "org.thingsboard.rule.engine.filter.TbJsFilterNode",
        "name": "Temperature check",
        "debugMode": false,
        "configuration": {
          "jsScript": "return msg.temp > 28;"
        }
      },
      {
        "additionalInfo": {
          "layoutX": 566,
          "layoutY": 250
        },
        "type": "org.thingsboard.rule.engine.action.TbCreateAlarmNode",
        "name": "Create Alarm",
        "debugMode": false,
        "configuration": {
          "alarmType": "Critical Temperatur",
          "alarmDetailsBuildJs": "var details = {};\nif (metadata.prevAlarmDetails)
{\n    details = JSON.parse(metadata.prevAlarmDetails);\n}\nreturn details;",
          "severity": "CRITICAL",
          "propagate": true,
          "useMessageAlarmData": false
        }
      },
      {
        "additionalInfo": {
          "layoutX": 281,
          "layoutY": 251
        },
        "type": "org.thingsboard.rule.engine.transform.TbTransformMsgNode",
        "name": "Add temperature to metadata",
        "debugMode": false,
        "configuration": {
          "jsScript": "metadata.temperature = msg.temperature; \nreturn {msg: msg, metadata: metadata, msgType: msgType};"
        }
      },
      {
        "additionalInfo": {
          "layoutX": 860,
          "layoutY": 253
        },
        "type": "org.thingsboard.rule.engine.mail.TbMsgToEmailNode",
        "name": "Temperature Email",
        "debugMode": false,
        "configuration": {
          "fromTemplate": "infoairquality@admin.org",
          "toTemplate": "decadex99@gmail.com",
          "ccTemplate": "muhamad.fadil@ui.ac.id",
          "bccTemplate": null,
          "subjectTemplate": "Device ${deviceType} temperature high",
          "bodyTemplate": "Device ${deviceName} has high temperature ${temperature} °C"
        }
      },
      {
        "additionalInfo": {
          "layoutX": 847,
          "layoutY": 151
        },
      "type": "org.thingsboard.rule.engine.mail.TbSendEmailNode",
      "name": "Send Email",
      "debugMode": false,
      "configuration": {
        "useSystemSmtpSettings": false,
        "smtpHost": "smtp.sendgrid.net",
        "smtpPort": 465,
        "username": "apikey",
        "password": "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",
        "smtpProtocol": "smtps",
        "timeout": 10000,
        "enableTls": true
      }
    }
  ],
  "connections": [
    {
      "fromIndex": 0,
      "toIndex": 2,
      "type": "True"
    },
    {
      "fromIndex": 1,
      "toIndex": 3,
      "type": "Created"
    },
    {
      "fromIndex": 2,
      "toIndex": 1,
      "type": "Success"
    },
    {
      "fromIndex": 3,
        "toIndex": 4,
        "type": "Success"
      }
    ],
    "ruleChainConnections": null
  }
}
```

- HTTP Access

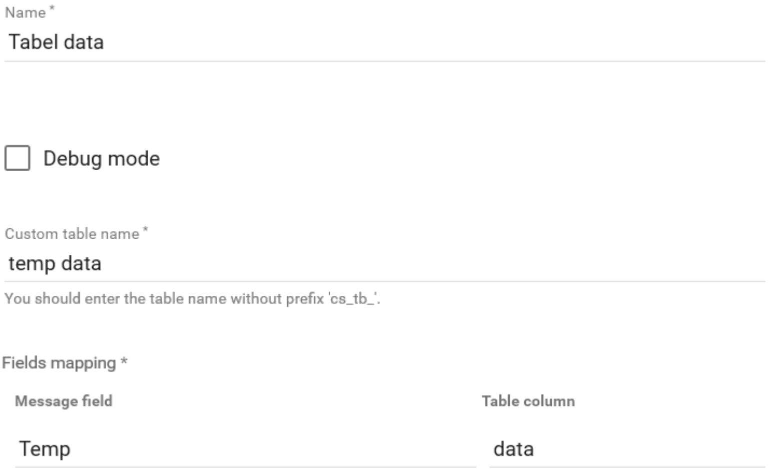


Code for nodes

```
//---Filter-script----//
const fetch1 = require("node-fetch");
var temperature;
const url1=
'https://testcheckclass.000webhostapp.com/L/TB?temp='+msg.temperature;
 const otherPram2 ={   //Parameter lain yang digunakan apda http request

             //body: formdata2,
             mode: 'cors',
             credentials:'omit',
             method:'GET'
             };
 fetch1(url1,otherPram2).then(response => {        //melakukan http request
menggunakan fetch API

     return response.json();

             }).then(json => {
          temperature=json.keterangan;
```

```
            });
 return temperature>50;
```

```
//----Transformation-script----//
return {msg: msg, metadata: metadata, msgType: msgType};
//----Action-create alarm ----//
var details = {};
if (metadata.prevAlarmDetails) {
    details = JSON.parse(metadata.prevAlarmDetails);
}
return details;
//----Action-save to custom table----//
```



Name *
Tabel data

□ Debug mode

Custom table name *
temp data
You should enter the table name without prefix 'cs_tb_'.

Fields mapping *

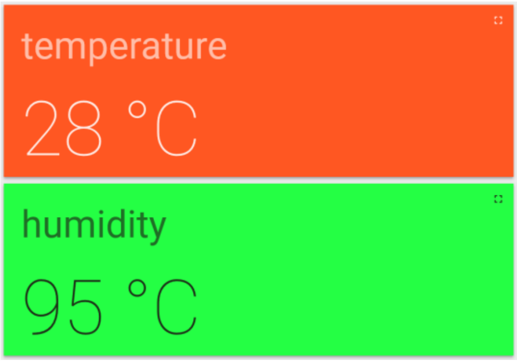
Message field	Table column
Temp	data

Full Code .json for HTTP Access

```
//.sjon code
{
  "ruleChain": {
    "additionalInfo": null,
    "name": "HTTP Access",
    "firstRuleNodeId": null,
    "root": false,
    "debugMode": false,
    "configuration": null
  },
  "metadata": {
    "firstNodeIndex": 0,
    "nodes": [
      {
        "additionalInfo": {
          "layoutX": 287,
          "layoutY": 150
        },
        "type": "org.thingsboard.rule.engine.filter.TbJsFilterNode",
        "name": "test",
        "debugMode": false,
        "configuration": {
          "jsScript": "const fetch1 = require(\"node-fetch\");\nvar
temperature;\nconst url1=
'https://testcheckclass.000webhostapp.com/L/TB?temp='+msg.temperature;\n
const otherPram2 ={\t//Parameter lain yang digunakan apda http
request\n\t\t\n\t\t//body: formdata2,\n\t\tmode:
'cors',\n\t\tcredentials:'omit',\n\t\tmethod:'GET'\t\n\t\t};\n
fetch1(url1,otherPram2).then(response => {\t\t//melakukan http request
menggunakan fetch API\n        \n\t    return response.json();\n\t\n\t\t}).then(json
=> {\n          temperature=json.keterangan;\n\t\t   });\n return temperature>50;\n
\n"
        }
      },
      {
        "additionalInfo": {
          "layoutX": 512,
          "layoutY": 152
        },
        "type": "org.thingsboard.rule.engine.transform.TbTransformMsgNode",
        "name": "kondisi",
        "debugMode": false,
        "configuration": {
          "jsScript": "return {msg: msg, metadata: metadata, msgType: msgType};"
        }
      },
      {
        "additionalInfo": {
          "layoutX": 578,
          "layoutY": 245
        },
        "type": "org.thingsboard.rule.engine.action.TbCreateAlarmNode",
        "name": "cek",
        "debugMode": false,
        "configuration": {
          "alarmType": "General Alarm",
          "alarmDetailsBuildJs": "var details = {};\nif (metadata.prevAlarmDetails)
{\n    details = JSON.parse(metadata.prevAlarmDetails);\n}\nreturn details;",
          "severity": "CRITICAL",
          "propagate": false,
          "useMessageAlarmData": false
        }
      },
      {
        "additionalInfo": {
          "layoutX": 838,
          "layoutY": 303
        },
        "type":
"org.thingsboard.rule.engine.action.TbSaveToCustomCassandraTableNode",
        "name": "Tabel data",
        "debugMode": false,
        "configuration": {
          "tableName": "temp data",
          "fieldsMapping": {
            "Temp": "data"
          }
        }
      }
    ],
    "connections": [
      {
        "fromIndex": 0,
        "toIndex": 1,
        "type": "True"
      },
      {
        "fromIndex": 1,
        "toIndex": 2,
        "type": "Success"
      },
      {
        "fromIndex": 2,
        "toIndex": 3,
        "type": "Created"
      }
    ],
    "ruleChainConnections": null
  }
}
```

- Dashboard

Temperature dashboard



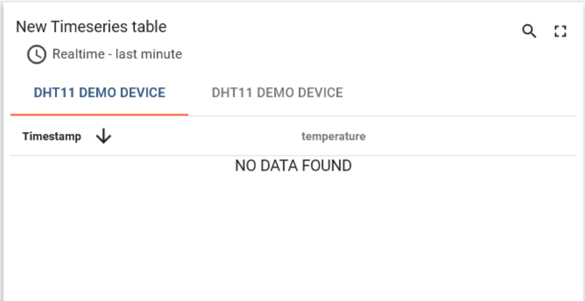
temperature

28 °C

humidity

95 °C

Real-time table



New Timeseries table

Realtime - last minute

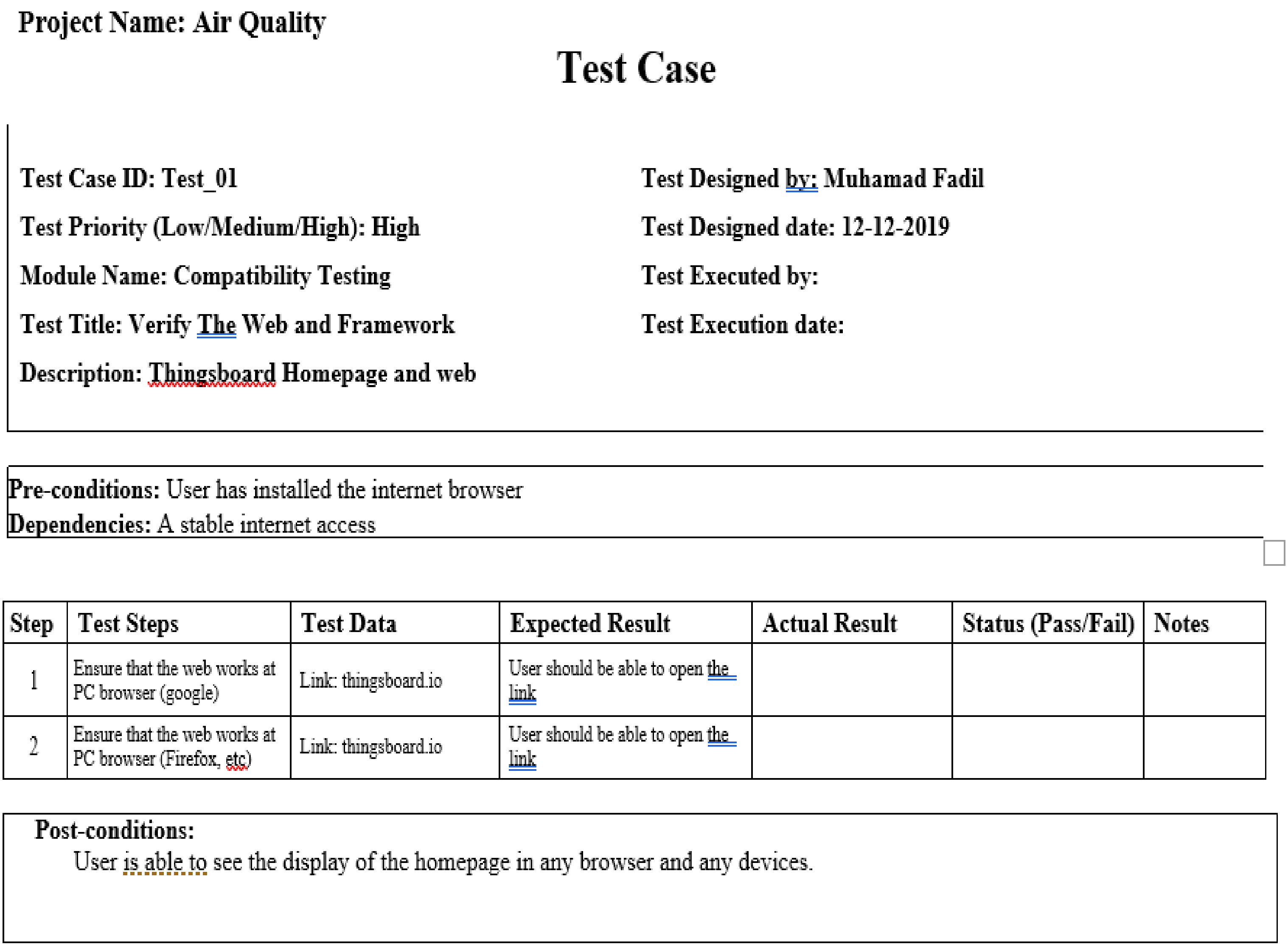
Timestamp	temperature

NO DATA FOUND

CHAPTER V

TESTING DOCUMENT

5.1 Test Case



Project Name: Air Quality

Test Case

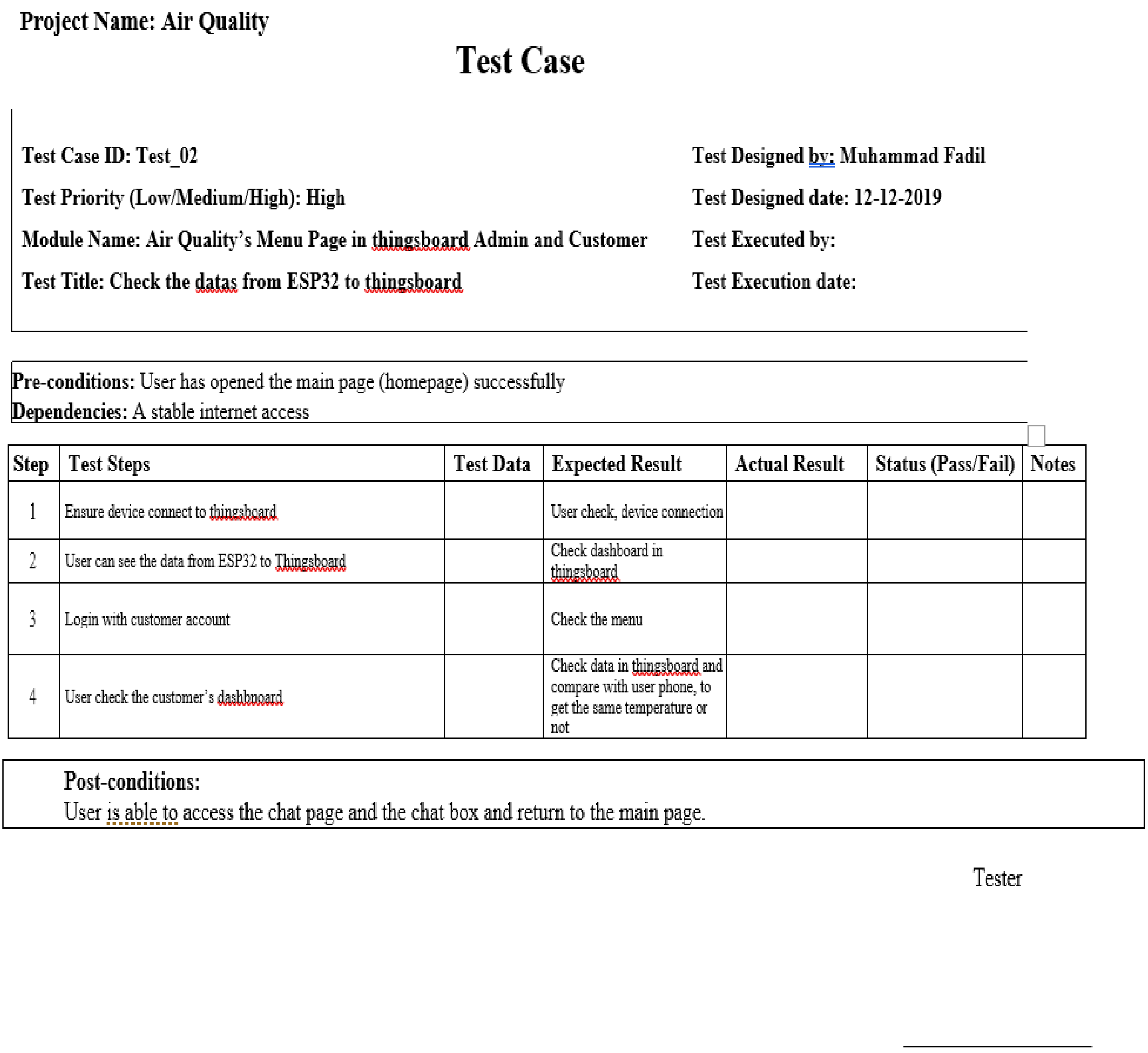
Test Case ID: Test_01	**Test Designed by: Muhamad Fadil**
Test Priority (Low/Medium/High): High	**Test Designed date: 12-12-2019**
Module Name: Compatibility Testing	**Test Executed by:**
Test Title: Verify The Web and Framework	**Test Execution date:**
Description: Thingsboard Homepage and web	

Pre-conditions: User has installed the internet browser
Dependencies: A stable internet access

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Ensure that the web works at PC browser (google)	Link: thingsboard.io	User should be able to open the link			
2	Ensure that the web works at PC browser (Firefox, etc)	Link: thingsboard.io	User should be able to open the link			

Post-conditions:
User is able to see the display of the homepage in any browser and any devices.

Tester



Project Name: Air Quality

Test Case

Test Case ID: Test_02 — **Test Designed by: Muhammad Fadil**

Test Priority (Low/Medium/High): High — **Test Designed date: 12-12-2019**

Module Name: Air Quality's Menu Page in thingsboard Admin and Customer — **Test Executed by:**

Test Title: Check the datas from ESP32 to thingsboard — **Test Execution date:**

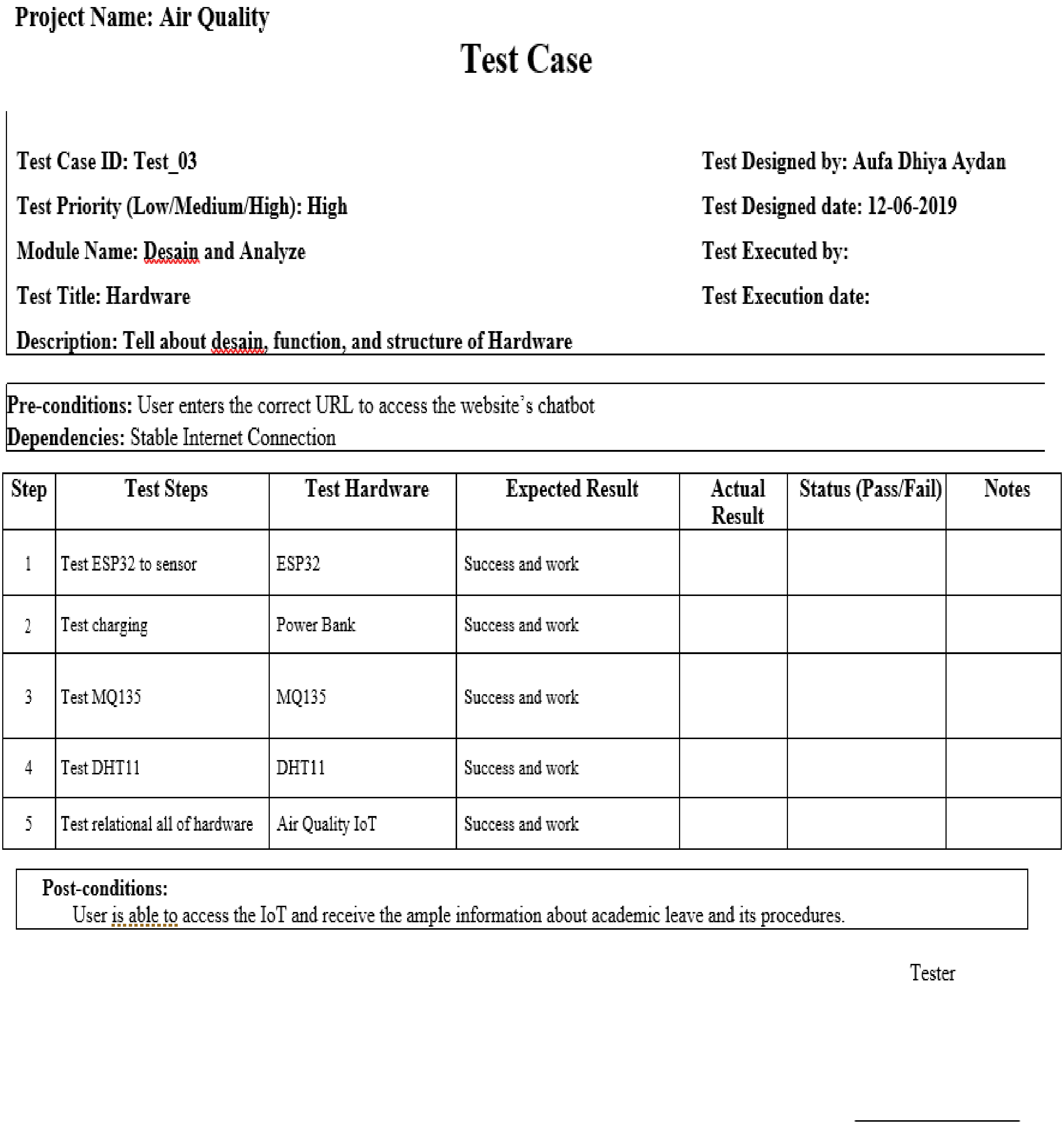
Pre-conditions: User has opened the main page (homepage) successfully
Dependencies: A stable internet access

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Ensure device connect to thingsboard		User check, device connection			
2	User can see the data from ESP32 to Thingsboard		Check dashboard in thingsboard			
3	Login with customer account		Check the menu			
4	User check the customer's dashbnoard		Check data in thingsboard and compare with user phone, to get the same temperature or not			

Post-conditions:
User is able to access the chat page and the chat box and return to the main page.

Tester

5.2 Lampiran Test Plan



Project Name: Air Quality

Test Case

Test Case ID: Test_03	**Test Designed by: Aufa Dhiya Aydan**
Test Priority (Low/Medium/High): High	**Test Designed date: 12-06-2019**
Module Name: Desain and Analyze	**Test Executed by:**
Test Title: Hardware	**Test Execution date:**

Description: Tell about desain, function, and structure of Hardware

Pre-conditions: User enters the correct URL to access the website's chatbot
Dependencies: Stable Internet Connection

Step	Test Steps	Test Hardware	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Test ESP32 to sensor	ESP32	Success and work			
2	Test charging	Power Bank	Success and work			
3	Test MQ135	MQ135	Success and work			
4	Test DHT11	DHT11	Success and work			
5	Test relational all of hardware	Air Quality IoT	Success and work			

Post-conditions:
User is able to access the IoT and receive the ample information about academic leave and its procedures.

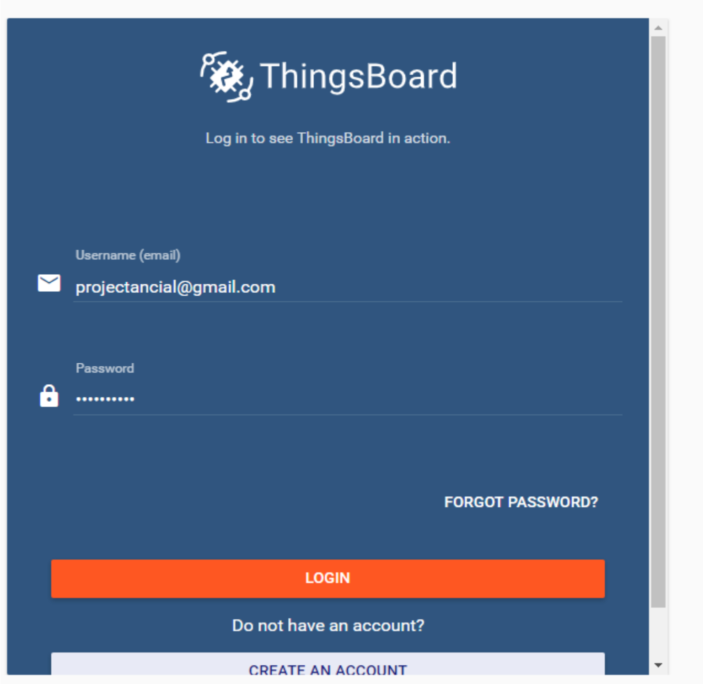
Tester

CHAPTER VI

USER MANUAL

6.1 Manual For User

To see the display of air quality, the first step is to log in to the platform



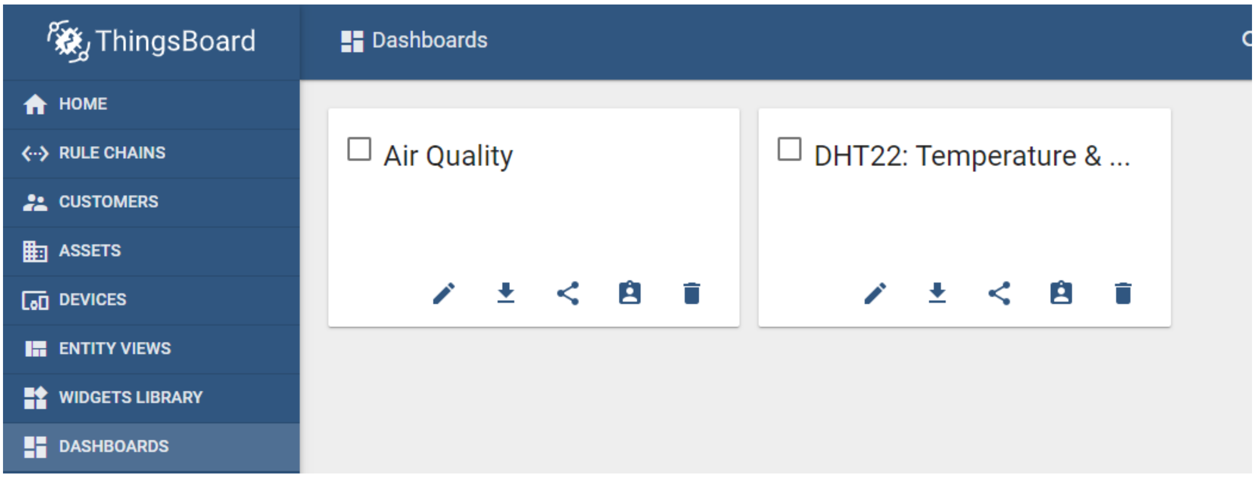
Log in to see ThingsBoard in action.

Username (email)
projectancial@gmail.com

Password
••••••••••

Do not have an account?

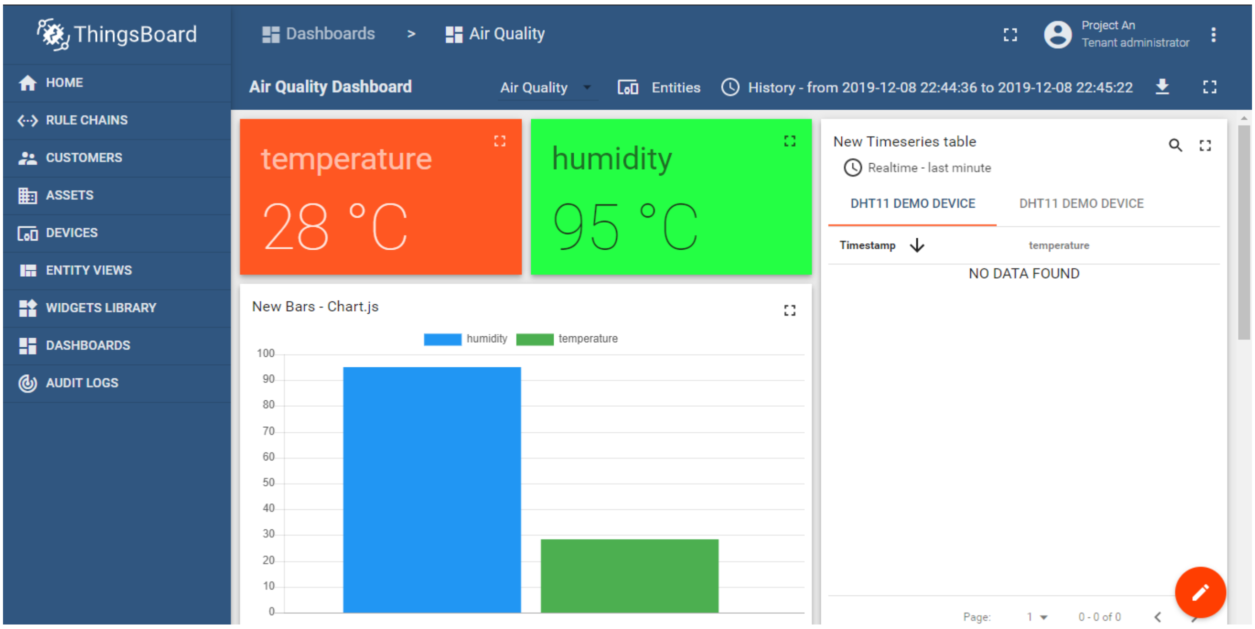
And then the dashboard display will appear like this



☐ Air Quality

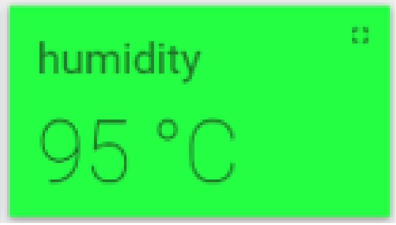
☐ DHT22: Temperature & ...

Select the Air Quality menu, then the display will appear as follows

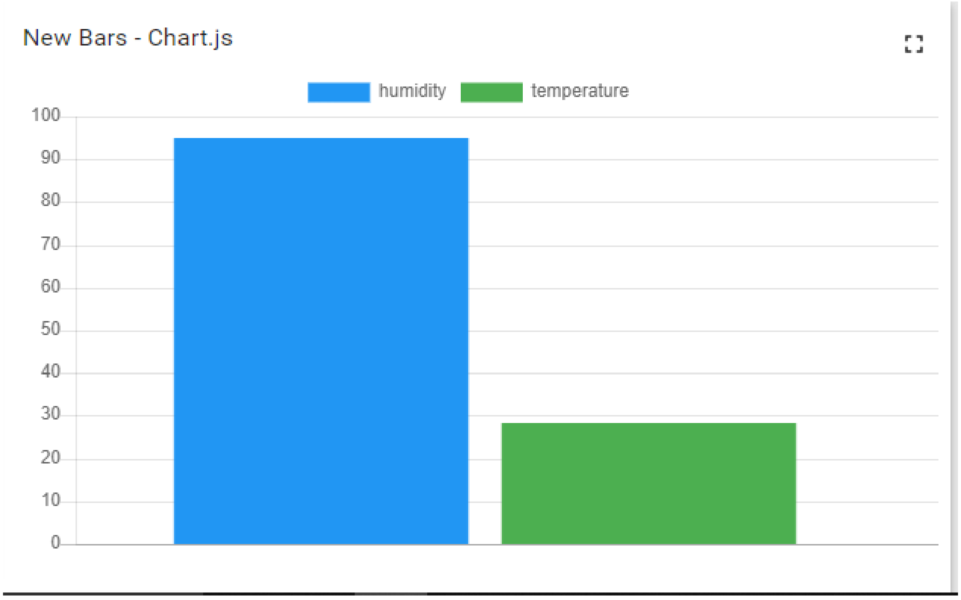


Temperature display is a display of the thingsboard produced from the temperature sensor on the hardware that we have made. Display temperature displays the temperature that is around our hardware.

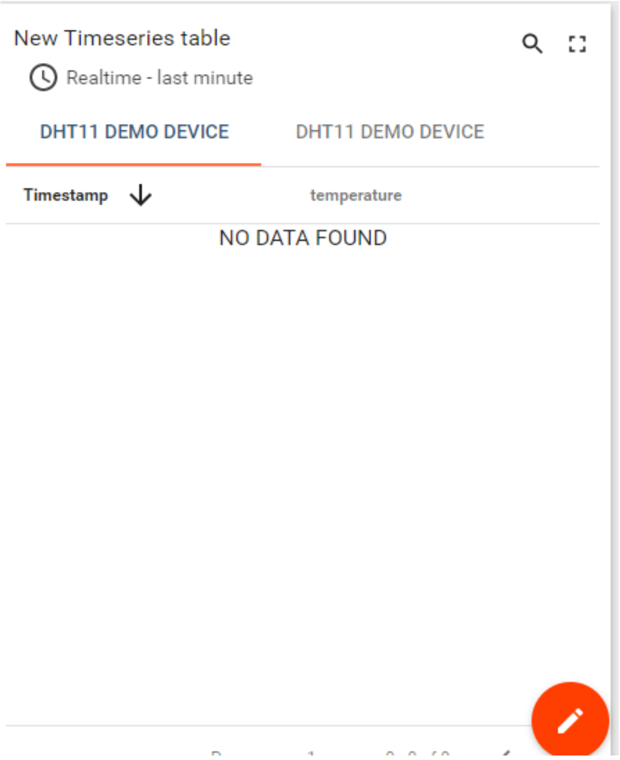
Humidity display is a display of thingsboard generated from the humidity sensor on the hardware that we have made. Display humidity displays the humidity that is around our hardware.



New Bars-Chart display is data from the reading of the humidity level represented in the form of a graph bar



The display below is a realtime display of the device that we connect to this thingsboard, which is the time of the movement of the increase in temperature and humidity.



New Timeseries table

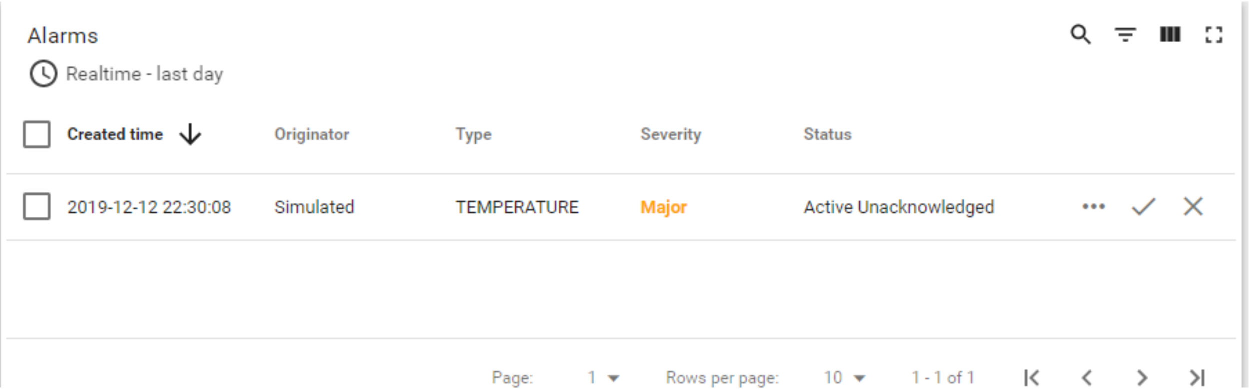
Realtime - last minute

DHT11 DEMO DEVICE | DHT11 DEMO DEVICE

Timestamp ↓	temperature

NO DATA FOUND

An alarm is a warning or notification that includes a create time, originator, type, severity, status column



Alarms

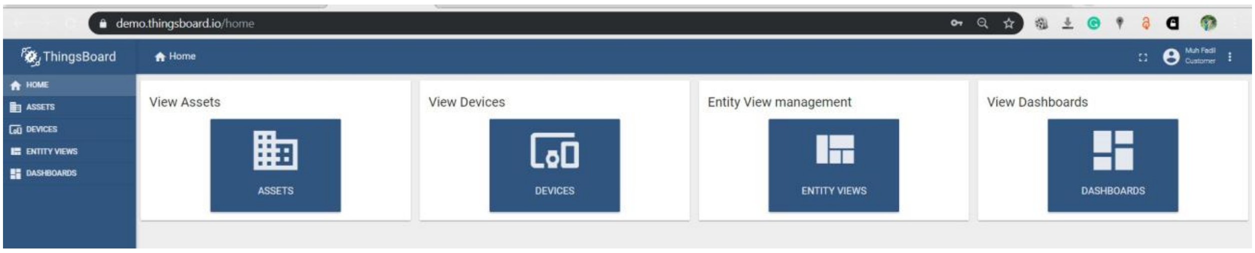
Realtime - last day

□	Created time ↓	Originator	Type	Severity	Status	
□	2019-12-12 22:30:08	Simulated	TEMPERATURE	Major	Active Unacknowledged	••• ✓ ✕

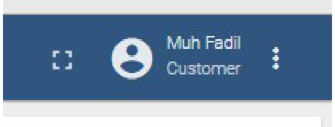
Page: 1 ▾ Rows per page: 10 ▾ 1 - 1 of 1

6.3 User Menu

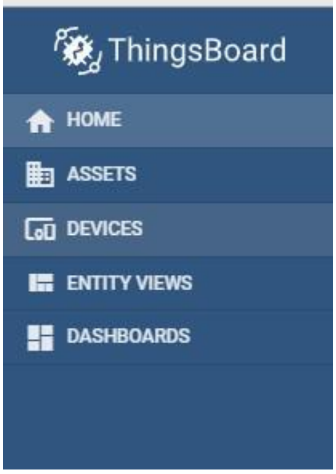
- ❖ Main Menu



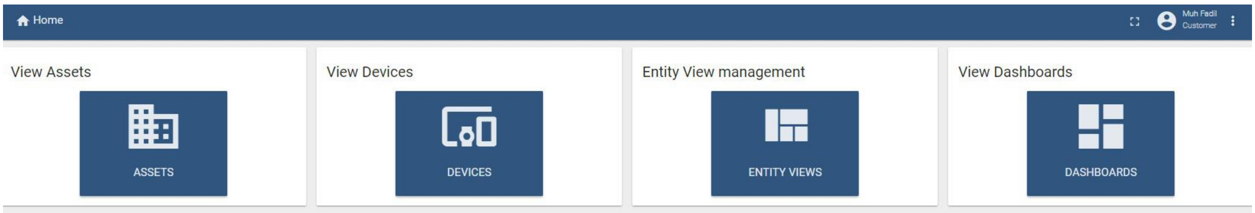
- ❖ Sign that the user is logged in



- ❖ Thingsboard User Menu

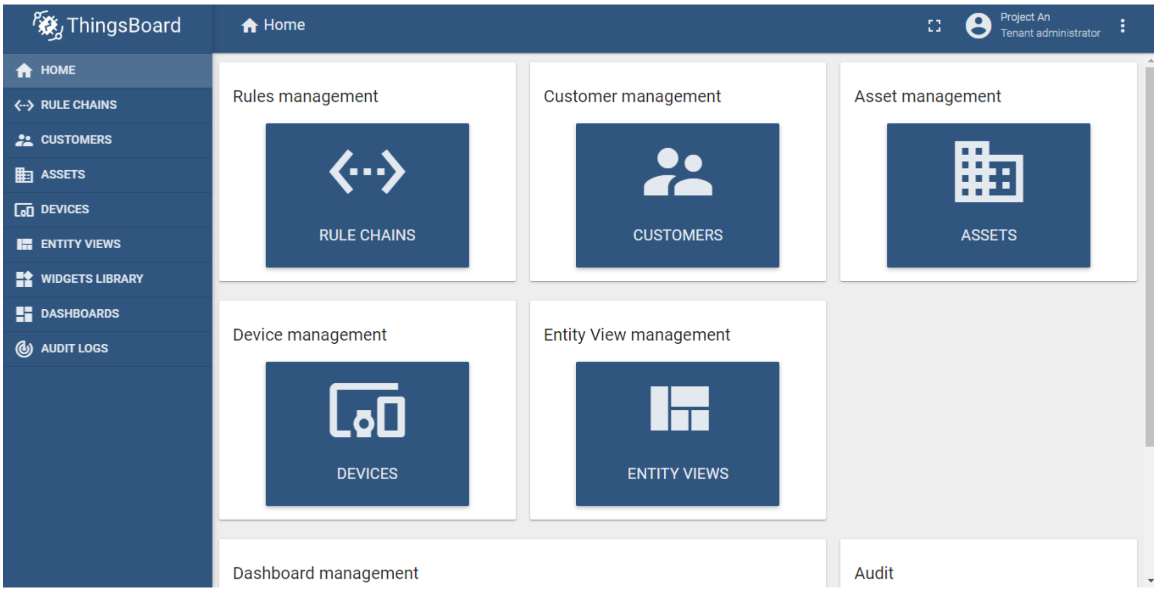


- ❖ Home Display

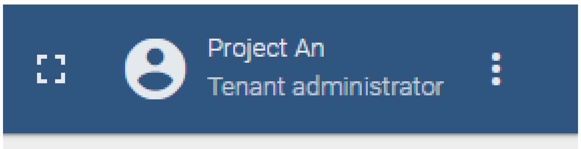


6.4 Admin Menu

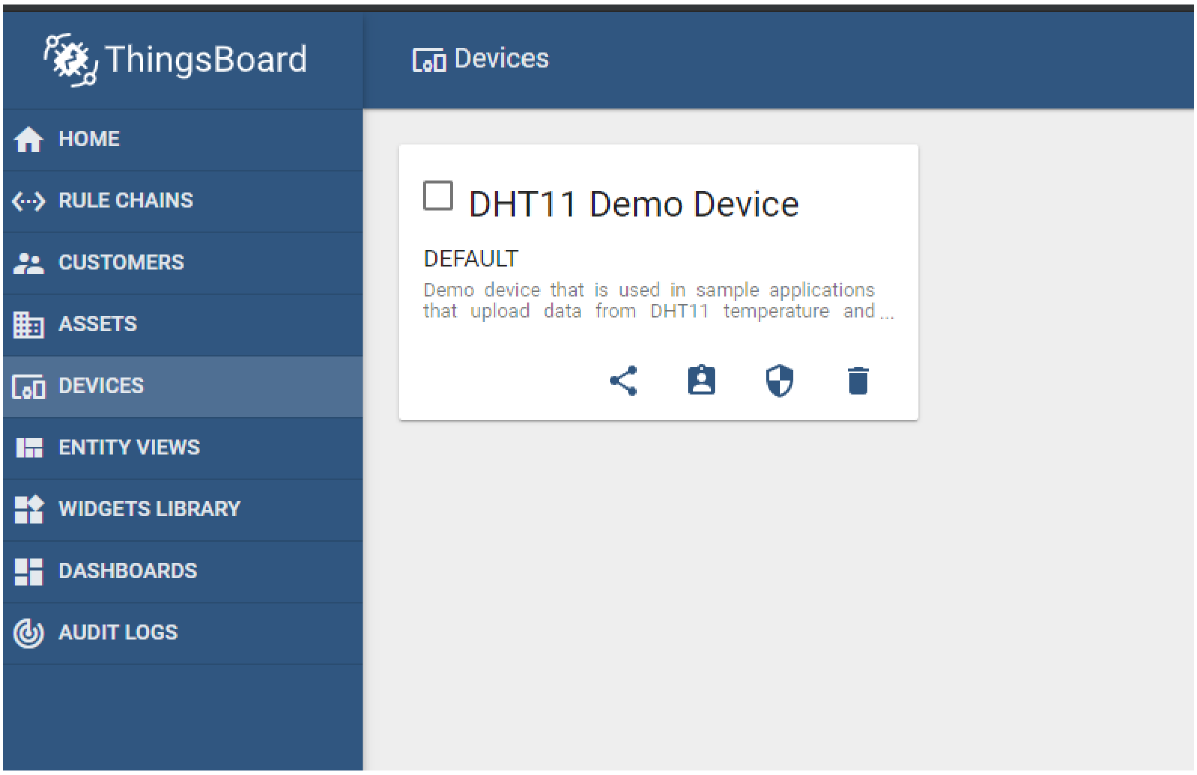
- ❖ Admin Home



- ❖ Sign that the admin login



- ❖ What device is being used

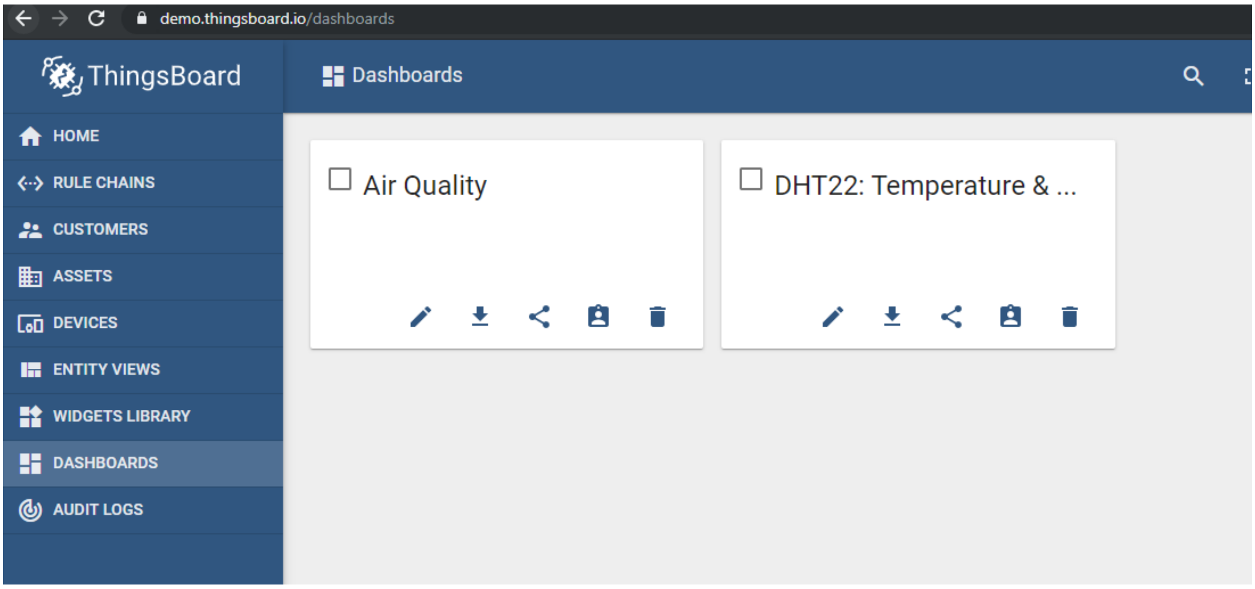


□ DHT11 Demo Device

DEFAULT

Demo device that is used in sample applications that upload data from DHT11 temperature and ...

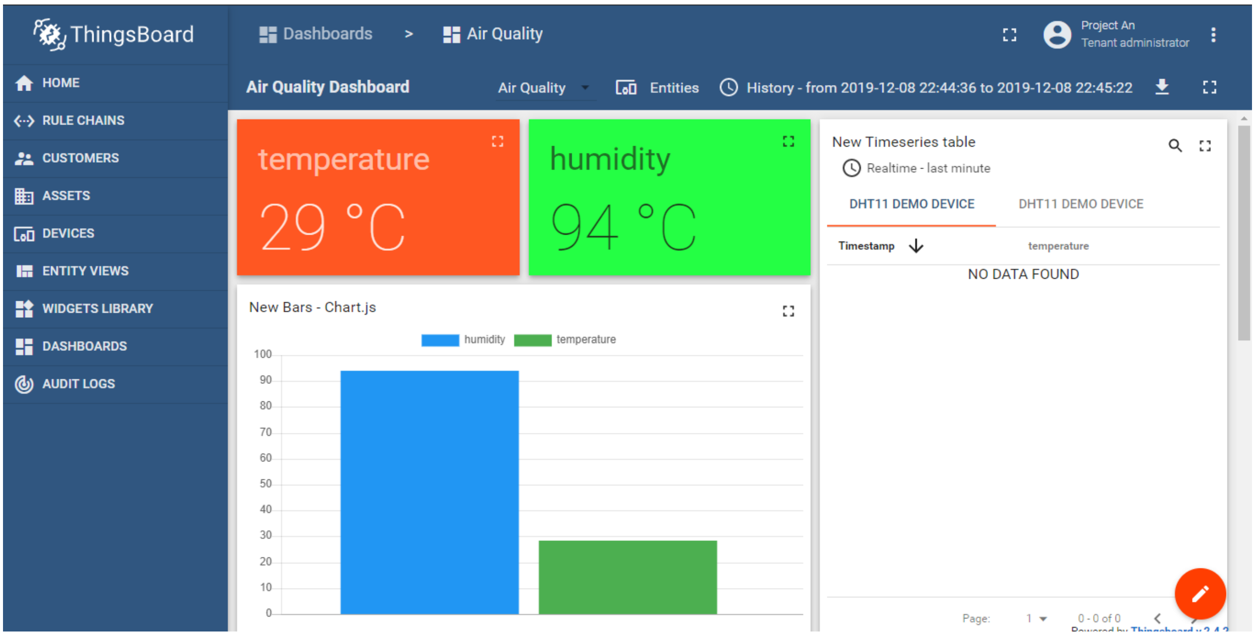
- ❖ Dashboard Display



□ Air Quality

□ DHT22: Temperature & ...

- ❖ Views



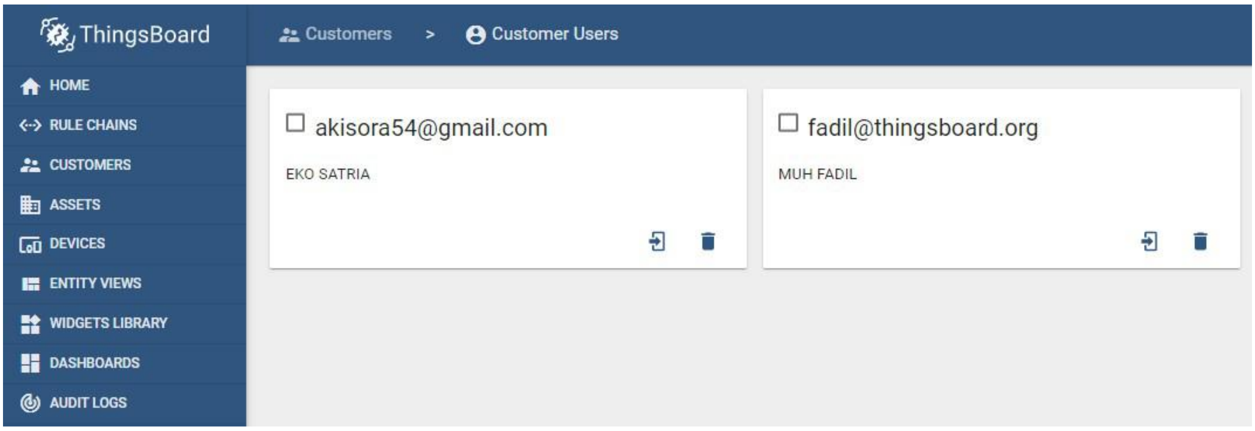
History - from 2019-12-08 22:44:36 to 2019-12-08 22:45:22

temperature 29 °C

humidity 94 °C

NO DATA FOUND

- ❖ Customer Data on Admin



☐ akisora54@gmail.com

EKO SATRIA

☐ fadil@thingsboard.org

MUH FADIL

CHAPTER VII

CONCLUSION

7.1 software requirement specification

7.1.1 functional requirements

In this system, the functional requirement consist of several function, that is:

1. Arduino will process the data from sensor, after that, it can display the data in from of numbers
2. Input data obtained from DHT11: the input input will be processed by the web server, then displayed to the web.
3. Input data obtained from MQ132: the input input will be processed by the web server, then displayed to the web.
4. User able to see the air quality from our website (just around FTUI)

7.1.2 non functional requirement

In this AirQuality system, needs that support the smooth functioning of the main functions can be defined in the table below:

Parameter	Explanation
Avaibility	The IoT system always on to display temperature, and Humidity
Practical	The IoT system must be practical to be placed anywhere and under any circumstances
Safety	When it is broken, it doesn't cause severe damage
Response Time	record every number changing
Aesthetics	Make it easy for users to use

7.1.3 interface requirements

Interface requirements from IoT program are hardware like MQ135, DHT11, Arduino and software like Arduino IDE, Xampp where the internet must be connected. To show the air quality must be provided is a web browser.

7.1.4 Spesification

This lot has been tested in the following specifications:

Functional	Software
Operating System	Windows 10, MacOS Mojave
Programming Language	C++ (Arduino), PHP
Word Processing	Notepad++

7.1.5 Design Limits

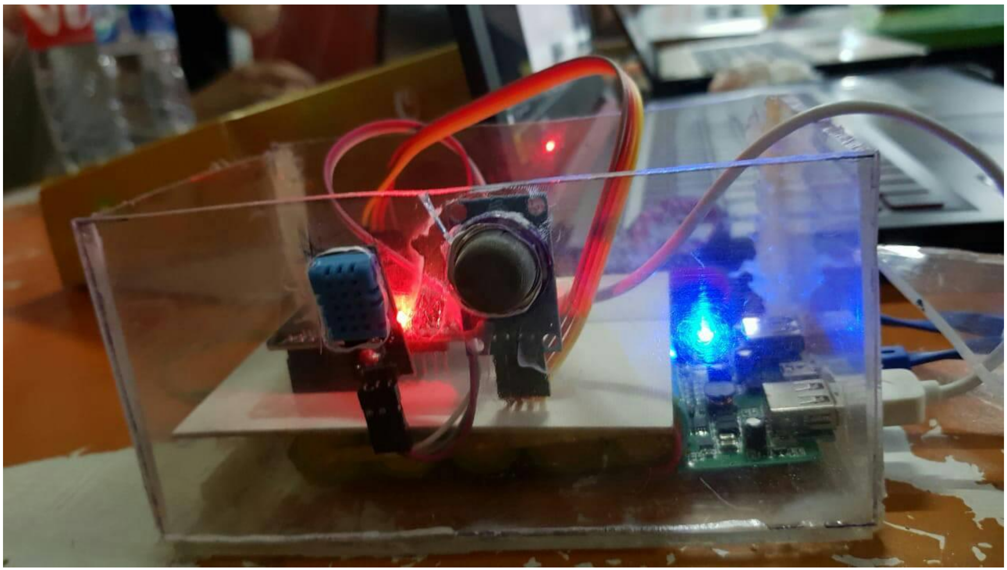
The limitation of designing this program is the IoT Web and can run on each operating system. We have UML diagram and interface designs.

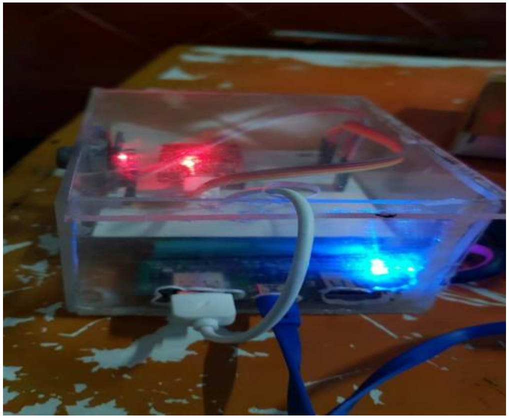
7.2 Software Design

already explained in chapter 4

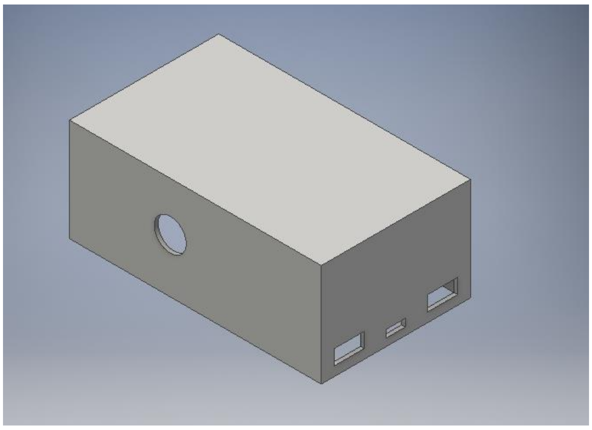
7.3 Hardware Design

that is the IoT Hardware Design

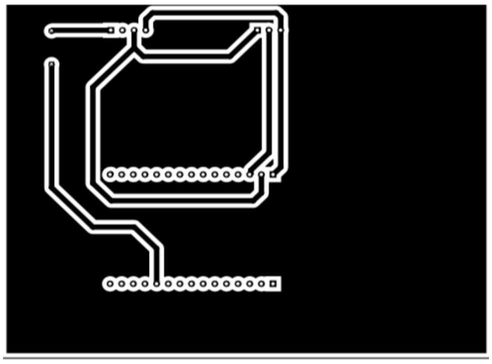




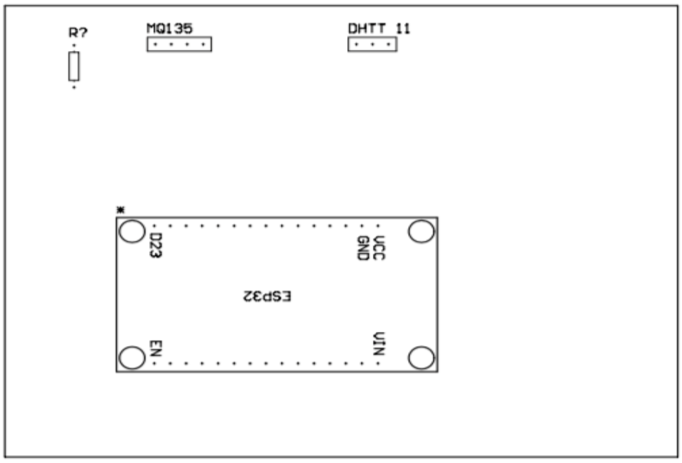
design via inventor



PCB design



Schematic design



Referensi:

- PPT Perkuliahan Rekayasa Perangkat Lunak
- Thingsboard.io. online. Available: https://thingsboard.io/docs/
- https://www.w3schools.com/js/js_json_intro.asp