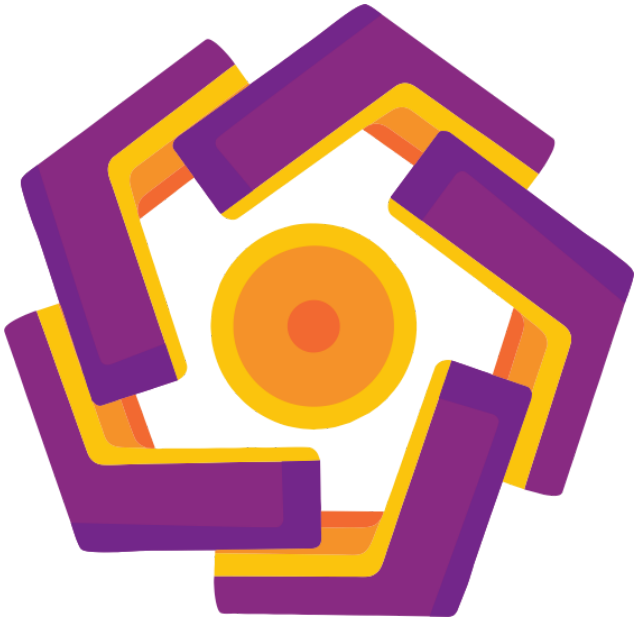
LAPORAN FINAL PROJECT
ALAT PENGUKUR KUALITAS AIR
MATA KULIAH INTERNET OF THINGS



Riefkyanov Surya A. P.	**20.11.3606**
Ahmad Faqih Hidayat	**20.11.3603**
Aditya Yoga Pratama	**20.11.3568**

PRODI S1 INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM YOGYAKARTA

2022/2023

A. Latar Belakang

Air merupakan salah satu elemen penting dalam hidup makhluk hidup terutama manusia. Kurang lebih 70 hingga 80 persen tubuh manusia terdiri dari air. Setidaknya, dalam sehari manusia mengkonsumsi 1,5 - 2 liter air untuk menjaga keseimbangan tubuhnya.

Batas aman air untuk dikonsumsi adalah yang memiliki TDS sekitar 300-500 ppm (parts per million) atau 300 – 500 mg/L. Jika TDS melebihi batas aman tersebut akan mengancam kesehatan tubuh apabila dikonsumsi dan dapat menyebabkan berbagai gangguan kulit jika digunakan untuk mandi dan kebutuhan sehari-hari. Alat ini dapat digunakan untuk monitoring kualitas air secara berkala, sepertihalnya pada sumur rumah dan penampungan air.

B. Analisa Proyek IOT

a. Analisa Kebutuhan Fungsional

Analisa kebutuhan fungsional dari Alat Pengukur Kualitas Air yang kami bangun adalah sebagai berikut.

No	Aktor	Deskripsi	Prioritas
1	Pengguna	Sistem dan perangkat dapat memonitor kualitas air.	Tinggi
		Sistem dan perangkat dapat memproses data mentah (raw data) menjadi data yang dapat dibaca dengan mudah oleh manusia.	Tinggi
		Sistem dan perangkat dapat mengirimkan data kualitas air yang telah diproses ke device pengguna melalui internet dan UI dashboard	Tinggi
		Sistem dan perangkat dapat mengirimkan notifikasi ke device pengguna jika ada anomali atau indeks kualitas air lebih dari batas aman.	Sedang
		Sistem dan perangkat dapat menyarankan tindakan apa yang	Rendah

		perlu dilakukan untuk menanggulangi jika terdapat peringatan terkait kualitas air.	

b. Analisa Kebutuhan Non Fungsional

Operasional :

1. Menggunakan sensor sebagai alat pembaca data.
2. Menggunakan mikrokontroler sebagai otak dari alat.
3. Menggunakan edge-devices (mini_pc) sebagai media pengolahan data.
4. Membutuhkan koneksi internet dan sistem operasi untuk beroperasi.
5. Seseorang yang mengawasi operasional alat.
6. Dokumentasi dan user manual untuk menuntun pengguna.

Kinerja :

1. Data kualitas air dibaca setiap 10 detik sekali.
2. Data pada dashboard akan diupdate setiap 10 detik sekali.

Keamanan :

1. Menggunakan cover pembungkus yang tahan terhadap air.

c. Analisa Kebutuhan Pengguna

Pada penggunaan alat pengukur kualitas air ini pengguna membutuhkan informasi berupa nilai hasil pembacaan kualitas air, kategori kualitas air, dan deskripsi atau saran dari kualitas air yang dibaca. Selain itu, pada dashboard antarmuka juga ditampilkan waktu pembacaan kualitas air.

Untuk menerima informasi tersebut, pengguna juga membutuhkan antarmuka yang mudah diterima dan dipahami.

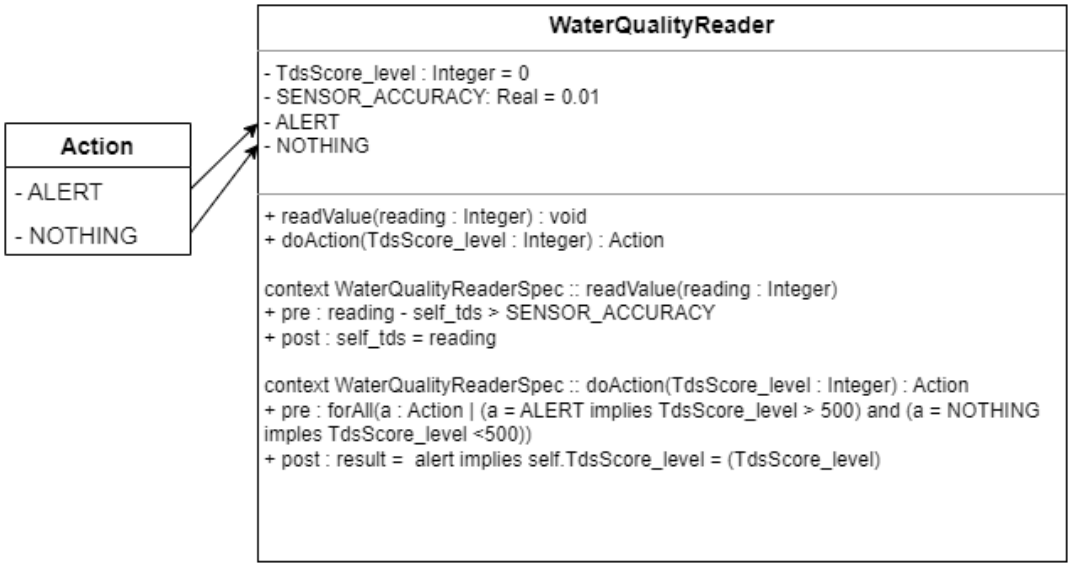
d. Analisa Bisnis (Business Model Canvas)

A. Key Partners

1. Investor
2. Supplier peralatan
3. Database provider
4. Cloud service

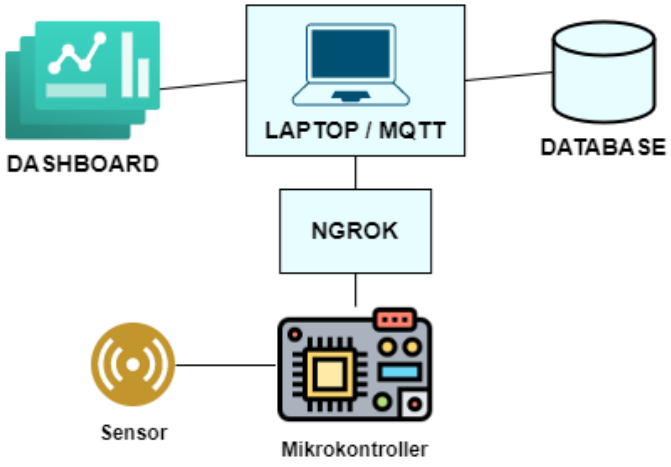
B. Key Activites
    1. Marketing
    2. Pemasangan alat
    3. Maintenance alat
    4. Pengadaan software
C. Value Propositions
    1. Menjaga kualitas air untuk digunakan sehari-hari
    2. Alat monitoring lebih terjangkau dibandingkan produk dari kompetitor
    3. Menjaga kesehatan
D. Customer Relationship
    1. Social media
    2. Review dan rating di halaman e-commerce dan marketplace
    3. Tech support
E. Customer Segments
    1. Keluarga
    2. Kost-kostan
    3. Rumah makan
F. Key Resource
    1. Sensor TDS
    2. ESP3266
    3. Teknisi
G. Channels
    1. UI Dashboard
H. Cost Structure
    1. Infrastruktur Teknologi
    2. Gaji karyawan
    3. Biaya iklan dan marketing
I. Revenue Streams
    1. Pembelian produk
    2. Biaya perawatan
    3. Upgrade fitur

e. Analisa Keamanan IOT (Framework ARMET)



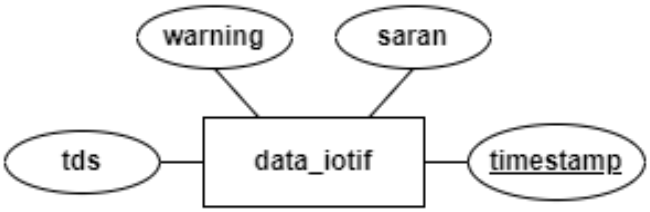
Alat akan melakukan pembacaan angka tds yang ada dalam air, apabila angka tds berada di atas 500 maka alat akan memberikan Alert kepada pengguna, dan apabila angka tds di bawah 500 alat tidak akan memberikan Alert kepada pengguna.

f. Analisa Kebutuhan Service IOT (HTTP atau MQTT)



Mosquitto digunakan sebagai broker MQTT. Ngrok digunakan untuk port forwarding agar service local dapat dijalankan secara publik. Laptop digunakan sebagai broker publisher dan subscriber. Mikrokontroller digunakan sebagai publisher. Koneksi internet dibutuhkan agar alat sensor dan mikrokontroller dapat bekerja dengan baik.

g. Analisa Kebutuhan Server Database IOT (ERD)

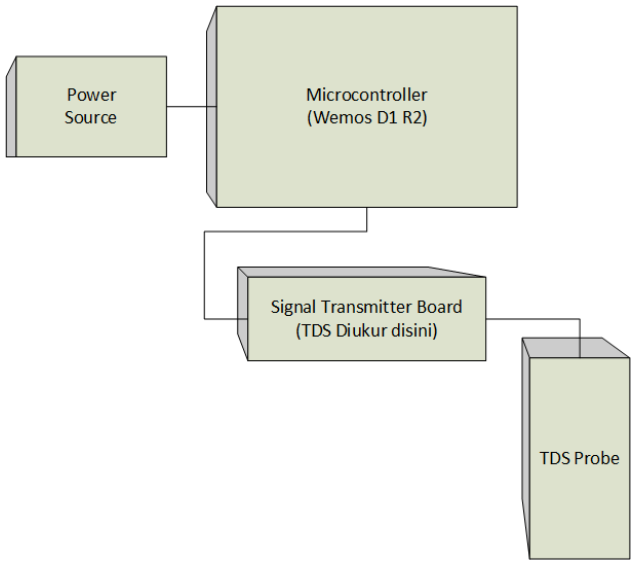


Pada penelitian ini terdapat satu tabel untuk menyimpan data pembacaan kualitas air. Pada tabel data_iotif memiliki terdapat kolom timestamp sebagai primay key, tds, warning, dan saran. Kolom timestamp berisikan waktu pembacaan data. Kolom tds berisikan nilai tds hasil pembacaan alat. Kolom warning merupakan klasifikasi apakah air aman, kurang baik, atau bahaya. Kolom saran berisikan saran berdasarkan klasifikasi yang ada pada kolom saran.

C. Rancangan Server IOT

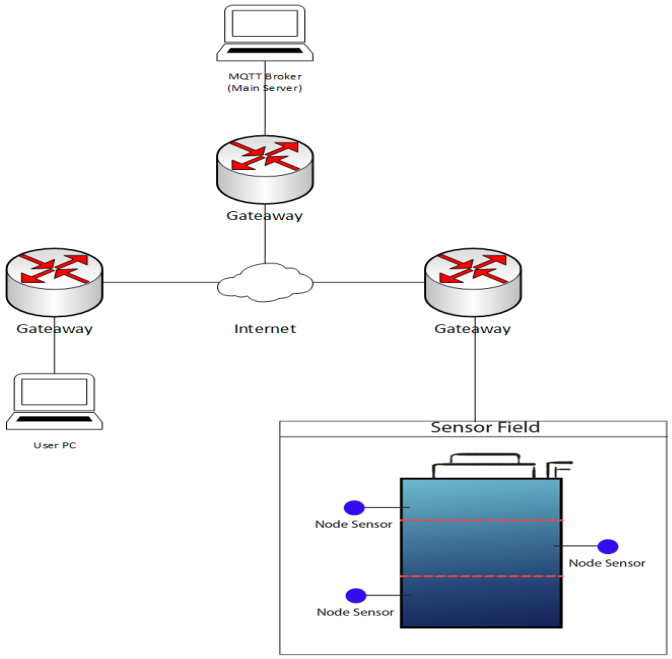
a. Desain Topologi Server

1. Rancangan Node Sensor (mikrokontroler)



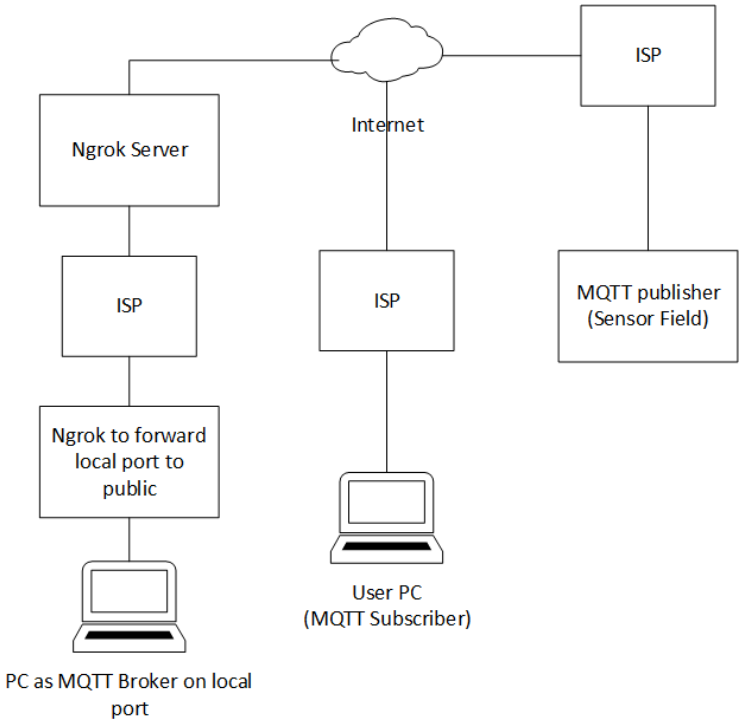
Sensor tds probe bekerja dengan menggunakan prinsip kerja dua elektroda yang terpisah untuk mengukur konduktivitas listrik dalam air. Nilai konduktivitas tersebut kemudian akan diolah dalam signal transmitter board untuk kemudian menjadi nilai tds air. Kemudian nilai tds akan dibaca oleh mikrokontroller.

2. Rancangan WSN



Sensor memiliki batasan jarak (range) dalam membaca kualitas air, sehingga diperlukan jaringan sensor agar dapat mengcover pembacaan kualitas air di keseluruhan penampungan air.

3. Rancangan Server IOT (Private ke Public)



b. Kode Program Server

1. Kode Program Server

```
from tkinter import *
import random
import json
from paho.mqtt import client as mqtt_client
from tkinter import Tk, Canvas, ttk
import sqlite3
from paho.mqtt import client as mqtt_client
import random
from datetime import datetime
import tkinter as tk

broker = '0.tcp.ap.ngrok.io'
port = 12107
topic = "tes/ngrok"
client_id =  f'python-mqtt-{random.randint(0, 100)}'
```

```
# connect to the database
conn = sqlite3.connect('db.sqlite')

# create a cursor
cursor = conn.cursor()

buat_tabel = '''CREATE TABLE IF NOT EXISTS data_iotif (
                        tds INTEGER NOT NULL,
                        warning TEXT NOT NULL,
                        saran TEXT NOT NULL,
                        timestamp NOT NULL
                        );'''

cursor.execute(buat_tabel)
conn.commit()

def connect_mqtt() -> mqtt_client:
        def on_connect(client, userdata, flags, rc):
            if rc == 0:
                print("Connected to MQTT Broker!")
            else:
                print("Failed to connect, return code
%d\n", rc)
        client = mqtt_client.Client(client_id)
        client.on_connect = on_connect
        client.connect(broker, port)
        return client

def subscribe(client: mqtt_client):
    def on_message(client, userdata, msg):
        print(f"Received `{msg.payload.decode()}` from
`{msg.topic}` topic")
        data_sensor_val = int(msg.payload.decode())
        warning=''
        saran=''
        timestamp=datetime.now()
        if data_sensor_val > 650:
            warning='bahaya'
            saran='periksa sumber air anda'
        elif data_sensor_val > 500:
```

```
            warning='kurang baik'
            saran='mungkin ada kontaminasi'
        else:
            warning='aman'
            saran='air aman'

        cursor.execute("INSERT INTO data_iotif (tds,
warning, saran, timestamp) VALUES(?, ? , ?, ?);",
(data_sensor_val, warning, saran, timestamp))
        conn.commit()

    client.subscribe(topic)
    client.on_message = on_message

def run():
    client = connect_mqtt()
    subscribe(client)
    client.loop_forever()

if __name__ == '__main__':
 run()
```

2. Kode Program Node Sensor (mikrokontroller)

```
#include <ESP8266WiFi.h>
#include <Ticker.h>
#include <AsyncMqttClient.h>
#include <Wire.h>
#include <SPI.h>
#define MQTT_PUB_TEMP "tes/ngrok"

#define TdsSensorPin A0
#define VREF 5.0      // analog reference voltage(Volt) of the ADC
#define SCOUNT  30           // sum of sample point

#define WIFI_SSID "AYEPEE"
```

```
#define WIFI_PASSWORD "IndomieSeleraku"

// Raspberry Pi Mosquitto MQTT Broker
#define MQTT_HOST "0.tcp.ap.ngrok.io"
// For a cloud MQTT broker, type the domain name
//#define MQTT_HOST "example.com"
#define MQTT_PORT 12107

int analogBuffer[SCOUNT];    // store the analog value in the array, read from ADC
int analogBufferTemp[SCOUNT];
int analogBufferIndex = 0, copyIndex = 0, tdsValue = 0;
float averageVoltage = 0, temperature = 25;

AsyncMqttClient mqttClient;
Ticker mqttReconnectTimer;

WiFiEventHandler wifiConnectHandler;
WiFiEventHandler wifiDisconnectHandler;
Ticker wifiReconnectTimer;

unsigned long previousMillis = 0;   // Stores last time temperature was published
const long interval = 10000;        // Interval at which to publish sensor readings

void connectToWifi() {
  Serial.println("Connecting to Wi-Fi...");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
}
```

```
void onWifiConnect(const WiFiEventStationModeGotIP& event) {
  Serial.println("Connected to Wi-Fi.");
  connectToMqtt();
}

void onWifiDisconnect(const WiFiEventStationModeDisconnected&
event) {
  Serial.println("Disconnected from Wi-Fi.");
  mqttReconnectTimer.detach(); // ensure we don't reconnect to MQTT
while reconnecting to Wi-Fi
  wifiReconnectTimer.once(2, connectToWifi);
}

void connectToMqtt() {
  Serial.println("Connecting to MQTT...");
  mqttClient.connect();
}

void onMqttConnect(bool sessionPresent) {
  Serial.println("Connected to MQTT.");
  Serial.print("Session present: ");
  Serial.println(sessionPresent);
}

void onMqttDisconnect(AsyncMqttClientDisconnectReason reason) {
  Serial.println("Disconnected from MQTT.");

  if (WiFi.isConnected()) {
    mqttReconnectTimer.once(2, connectToMqtt);
  }
}
```

```
void onMqttPublish(uint16_t packetId) {
  Serial.print("Publish acknowledged.");
  Serial.print("  packetId: ");
  Serial.println(packetId);
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(TdsSensorPin, INPUT);

  wifiConnectHandler = WiFi.onStationModeGotIP(onWifiConnect);
  wifiDisconnectHandler =
WiFi.onStationModeDisconnected(onWifiDisconnect);

  mqttClient.onConnect(onMqttConnect);
  mqttClient.onDisconnect(onMqttDisconnect);
  //mqttClient.onSubscribe(onMqttSubscribe);
  //mqttClient.onUnsubscribe(onMqttUnsubscribe);
  mqttClient.onPublish(onMqttPublish);
  mqttClient.setServer(MQTT_HOST, MQTT_PORT);
  // If your broker requires authentication (username and password), set
them below
  //  mqttClient.setCredentials("iotif", "aditsuryafaqih123");
  connectToWifi();

}

void loop() {
  static unsigned long analogSampleTimepoint = millis();
```

```
  if (millis() - analogSampleTimepoint > 40U)  //every 40
milliseconds,read the analog value from the ADC
  {
    analogSampleTimepoint = millis();
    analogBuffer[analogBufferIndex] = analogRead(TdsSensorPin);
//read the analog value and store into the buffer
    analogBufferIndex++;
    if (analogBufferIndex == SCOUNT)
      analogBufferIndex = 0;
  }
  static unsigned long printTimepoint = millis();
  if (millis() - printTimepoint > 10000U)
  {
    printTimepoint = millis();
    for (copyIndex = 0; copyIndex < SCOUNT; copyIndex++)
      analogBufferTemp[copyIndex] = analogBuffer[copyIndex];
    averageVoltage = getMedianNum(analogBufferTemp, SCOUNT) *
(float)VREF / 1024.0; // read the analog value more stable by the median
filtering algorithm, and convert to voltage value
    float compensationCoefficient = 1.0 + 0.02 * (temperature - 25.0);
//temperature compensation formula: fFinalResult(25^C) =
fFinalResult(current)/(1.0+0.02*(fTP-25.0));
    float compensationVolatge = averageVoltage / compensationCoefficient;
//temperature compensation
    tdsValue = (133.42 * compensationVolatge * compensationVolatge *
compensationVolatge - 255.86 * compensationVolatge *
compensationVolatge + 857.39 * compensationVolatge) * 0.5; //convert
voltage value to tds value
    String tdsValueString = "";
    tdsValueString.concat(tdsValue);
```

```
    uint16_t packetIdPub1 = mqttClient.publish(MQTT_PUB_TEMP, 1,
true, String(tdsValue).c_str());
    Serial.printf("Publishing on topic %s at QoS 1, packetId: %i",
MQTT_PUB_TEMP, packetIdPub1);
  }
}

int getMedianNum(int bArray[], int iFilterLen)
{
  int bTab[iFilterLen];
  for (byte i = 0; i < iFilterLen; i++)
    bTab[i] = bArray[i];
  int i, j, bTemp;
  for (j = 0; j < iFilterLen - 1; j++)
  {
    for (i = 0; i < iFilterLen - j - 1; i++)
    {
      if (bTab[i] > bTab[i + 1])
      {
        bTemp = bTab[i];
        bTab[i] = bTab[i + 1];
        bTab[i + 1] = bTemp;
      }
    }
  }
  if ((iFilterLen & 1) > 0)
    bTemp = bTab[(iFilterLen - 1) / 2];
  else
    bTemp = (bTab[iFilterLen / 2] + bTab[iFilterLen / 2 - 1]) / 2;
  return bTemp;
}
```

D. Rancangan Dashboard IOT

a. Wireframe Dashboard IOT (jelaskan berdasarkan analisa kebutuhan)

Penjelasan :

Berdasarkan analisa kebutuhan, didapatkan kebutuhan informasi yang perlu diketahui oleh pengguna alat pendeteksi kualitas air adalah sebagai berikut.

1. Nilai TDS

   Nilai *Total Dissolve Solids* yang dibaca oleh alat akan ditampilkan pada dashboard alat sebagai informasi utama kepada pengguna, agar pengguna tahu berapa nilai tds yang ada dalam air.

2. Status kualitas air

   Status kualitas air adalah informasi turunan berdasarkan nilai tds yang dibaca oleh alat. Pada alat ini, status kualitas air diklasifikasikan menjadi 3 yaitu, aman, kurang baik, dan bahaya.

3. Saran tindakan

   Saran tindakan juga merupakan informasi turunan berdasarkan nilai tds. Apabila nilai tds di bawah 500 atau pada status aman, saran tindakan akan menampilkan air aman. Apabila nilai tds berada antara 500 hingga 650, saran akan menampilkan mungkin ada kontaminasi. Dan ketika nilai tds

berada di atas 650 atau berada di status bahaya, saran akan menampilkan periksa sumber air anda.

4. Waktu pengambilan data

   Waktu pengambilan data merupakan waktu ketika alat mencatat dan kemudian menyimpannya pada database. Waktu pengambilan data menggunakan format YYYY-MM-DD HH:MM:SS. Waktu pengambilan data perlu ditampilkan pada dashboard agar pengguna mengetahui kualitas air yang ditampilkan tersebut ada pada waktu kapan.

b. Desain Asset

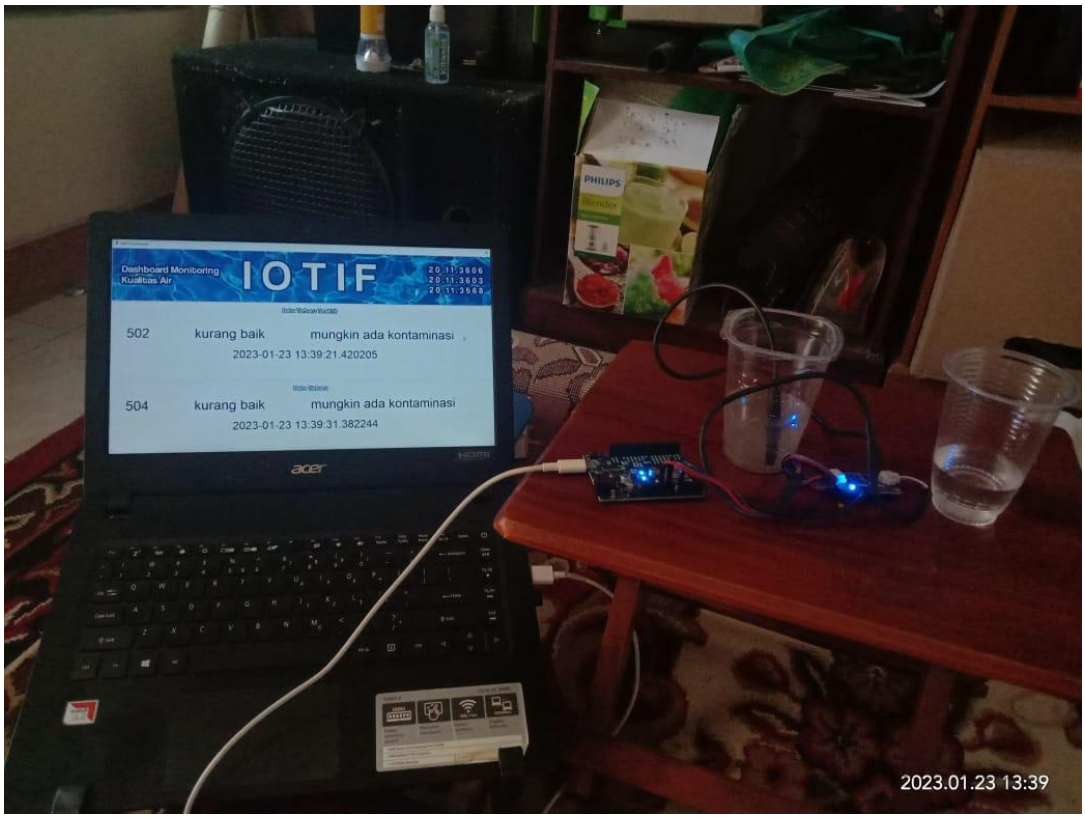
Header:

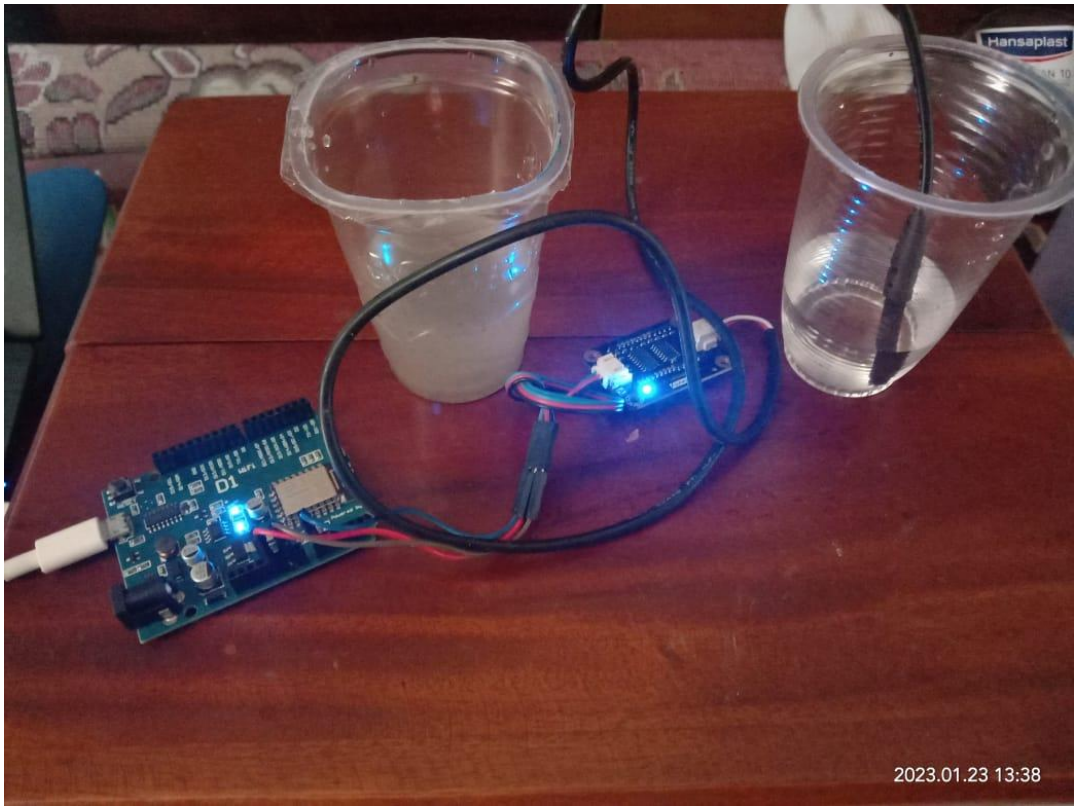
Gambar canvas data terbaru atau data ke n :

Gambar canvas data ke n-1 :

c. High Fidelity Dashboard IOT

E. Unit Testing (Blackbox Testing)





Pada blackbox testing ini kami melakukan pengujian dengan fokus pada fungsi input dan output dari alat sistem monitoring kualitas air. Pada pengujian ini sensor yang digunakan dapat dengan baik membaca nilai tds pada air dan kemudian dashboard yang dibuat juga dapat menampilkan nilai tds beserta keterangan status dan keterangan waktunya.

F. Kesimpulan

Berdasarkan analisa-analisa dari infrastruktur iot sistem monitoring kualitas air yang dibuat, dapat dilihat kapabilitas dan batasan-batasan sistem dalam melakukan monitoring kualitas air berdasarkan index tds yang terdeteksi oleh sensor. Dengan penggunaan protokol mqtt, data yang berasal dari node(publisher) dapat diteruskan ke broker mqtt yang telah terhubung ke ngrok sehingga data dapat terdistribusi ke banyak device(subscriber) sekaligus.