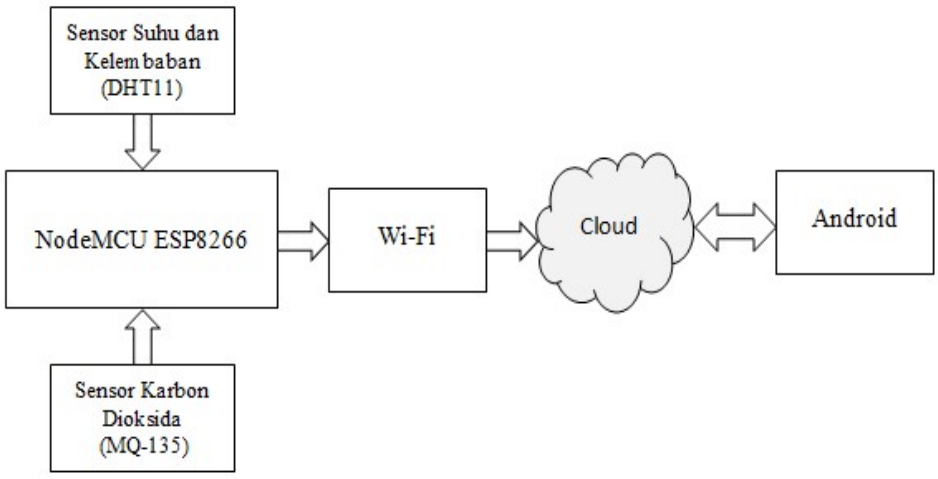
BAB 3
METODE PENELITIAN

3.1.Desain Sistem

Perancangan dan desain sistem diperlukan untuk memudahkan dalam pembuatan sistem pemantau ini. Perancangan sistem terdiri dari desain perangkat keras (*hardware*) dan desain perangkat lunak (*software*). Sistem menggunakan jaringan internet agar dapat mengunggah data sensor ke dalam *cloud* yang digunakan dan selanjutnya akan ditampilkan ke *Graphical User Interface* (GUI). Pada desain perangkat keras akan dijelaskan tentang bagaimana perancangan dan sistem yang akan dibuat, sementara untuk perancangan perangkat lunak akan dijelaskan bagaimana alur diagram pemrograman pada kontroler dan alur diagram pada aplikasi Android untuk menampilkan data dari *cloud*.

Desain sistem dapat dilihat secara menyeluruh pada diagram blok yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Diagram Blok Sistem

Pada perancangan desain sistem, mikrokontroler menggunakan NodeMCU ESP8266 yang terhubung ke semua sensor yang sudah ditentukan seperti sensor suhu, kelembaban, dan gas CO_2. NodeMCU ESP8266 memiliki antena sehingga tidak dipelrukan modul Wi-Fi tambahan untuk bisa terhubung ke internet melalui

Wi-Fi. Jika sudah terhubung ke jaringan internet, maka data sensor akan diunggah ke *cloud* Antares. Data yang ada di *cloud* tersebut diolah agar dapat ditampilkan pada Smartphone dengan platform Android.

Komponen-komponen yang ada pada diagram blok sistem ini terdiri dari :

1) NodeMCU ESP8266

   NodeMCU ESP8266 ini berfungsi sebagai kontroler. Pada blok ini terjadi proses pembacaan dari sensor yang digunakan, dan kemudian data tersebut akan dikirim.

2) Sensor Suhu dan Kelembaban

   Pada blok ini pembacaan suhu dan kelembaban. Kedua parameter ini diukur oleh sebuah modul sensor. Suhu dalam satuan Celcious (°C) dan Kelembaban dalam satuan %RH.

3) Sensor Karbon Dioksida (CO_2)

   Pada blok ini pembacaan kadar gas Karbon Dioksida dengan menggunakan sebuah modul sensor. Karbon Dioksida dalam satuan ppm.

4) Wi-Fi

   Wi-Fi berfungsi sebagai *access point* agar dapat terhubung ke jaringan internet. Sistem ini menggunakan transmisi *wireless* sehingga membutuhkan konfigurasi agar dapat terhubung ke internet melalui Wi-Fi.

5) *Cloud*

   Cloud tersedia jika terhubung ke dalam jaringan internet. Data hasil baca dari sensor akan disimpan di *cloud* dan kemudian diolah untuk ditampilkan ke Smartphone Android sebagai GUI.

6) Android

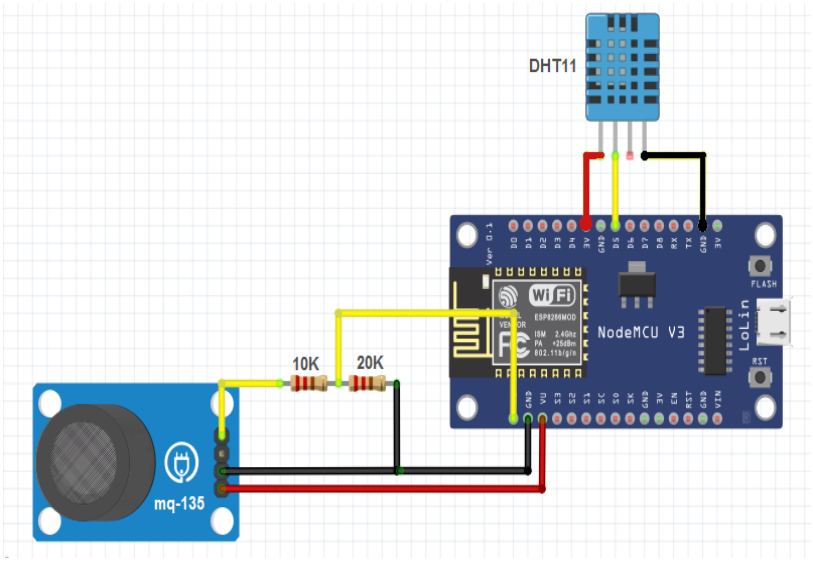
   Blok ini merupakan hasil tampilan GUI pada Smartphone Android dari data sensor yang tersimpan di *cloud* agar dapat dipahami pengguna.

3.2.Perancangan Sistem

3.2.1.Perangat Keras (*Hardware*)

Pada perancangan perangkat keras terjadi proses dimana menghubungkan sensor-sensor ke NodeMCU ESP8266, kemudian dihubungkan ke jaringan

internet melalui Wi-Fi, setelah itu data yang dibaca oleh sensor diunggah ke *cloud* dan kemudian ditampilkan ke Smartphone Android sebagai GUI.



Gambar 3.2 *WiringDiagram* Perangkat Keras

Komponen-komponen yang digunakan dalam perancangan perangkat keras (*hardware*), sebagai berikut :

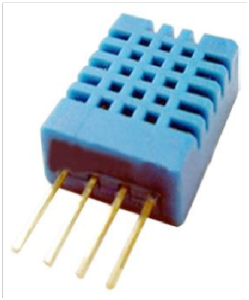
1) NodeMCU ESP8266

NodeMCU adalah sebuah perangkat keras berupa *System on Chip* ESP8266 buatan *Espressif System* yang berbasis modul ESP-12. NodeMCU dapat dianalogikan sebagai board Arduino dengan chip ESP8266 yang memiliki fitur Wi-Fi sehingga dapat dengan mudah terhubung ke jaringan internet tanpa membutuhkan perangkat modul Wi-Fi lagi. NodeMCU memiliki 1 pin analog input, 11 pin GPIO dari D0-D10, fungsionalitas PWM, interface I2C, SPI, dan 1-*wire*.

Gambar 3.3 NodeMCU ESP8266

2) Sensor DHT11

Sensor DHT11 ini berfungsi untuk mengukur suhu dan kelembaban.



Gambar 3.4 Sensor DHT11

Spesifikasi dari sensor DHT11, sebagai berikut :

a.*Power Supply* : 3-5.5V DC
b.*Output Signal* : Digital
c.*Interface* : 1-*wire* (*single bus*)
d.*Measuring Range* : Suhu 0-50 °C
Kelembaban 20-90% RH
e. *Accuracy* : Suhu ±2 °C
Kelembaban ±4% RH
f. *Resolution/Sensitivity* : Suhu 0,1 °C
Kelembaban 1%RH

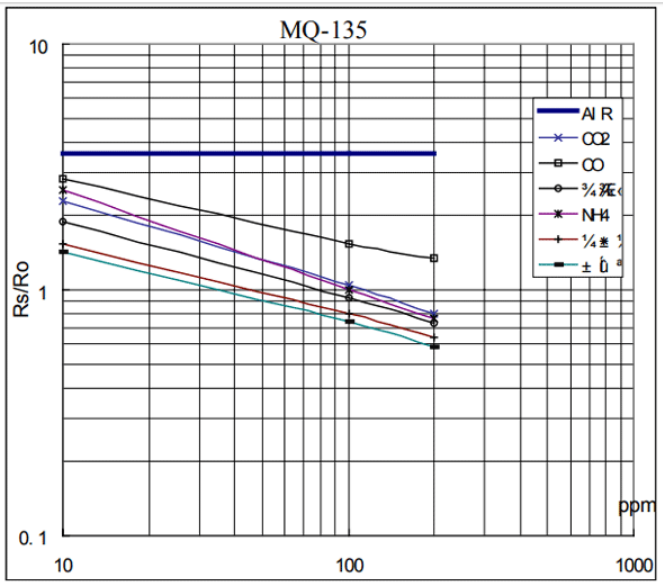
3) Sensor MQ-135

Sensor MQ-135 berfungsi untuk mengukur kadar gas berbahaya yang ada di udara, salah satunya yaitu Karbon Dioksida (CO_2). Sensor ini bekerja dengan cara menerima perubahan nilai resistansi (analog) bila terkena gas. Sensor ini memiliki daya tahan yang baik untuk penggunaan Penanda bahaya polusi karena praktis dan tidak memakan daya yang besar.

Gambar 3.5 Sensor MQ-135

Modul sensor ini membutuhkan *power supply* 5V DC. Pada sensor ini terdapat dua pin untuk keluaran sinyal, yaitu untuk pin keluaran analog (A0) dan keluaran digital (D0) yang berupa TTL. Pin analog dari MQ-135 terhubung dengan pin analog pada nodeMCU ESP8266, kemudian dilakukan *debugmicrocontroller* nodeMCU ESP8266 melalui pemrograman pada Arduino IDE. Setelah ini hasil pembacaan sensor dari sensor dilihat melalui serial monitor.

Nilai yang terbaca pada serial monitor masih berupa nilai ADC dan belum terkalibrasi untuk pendeteksian gas *carbon dioxide* (CO_2). Selanjutnya untuk mengkalibrasi agar nilai pembacaan sensor menjadi nilai ppm (satuan gas CO_2), pertama yang dilakukan harus mengetahui grafik Rs/Ro terhadap ppm dari datasheet MQ-135 untuk pembacaan sensor gas CO_2.Grafik dibawah adalah acuan untuk mengkalibrasi sensor agar bisa menemukan nilai ppm.



Gambar 3.6 Karakteristik Sensitivitas MQ-135

Grafik diatas adalah acuan untuk mengkalibrasi sensor agar dapat menemukan nilai ppm.Untuk mencari nilai Rs/Ro perlu mencari nilai Rs dan nilai Ro. Rs adalah nilai resistansi sensor pada konsentrasi gas dan Ro adalah tahanan sensor pada udara yang bersih.Rs/Ro juga dapat disebut sebagai rasio.Pada saat udara bersih, rasio sebesar 3,6. Diperoleh pada

serial monitor nilai ADC sebesar 25 pada ruang tanpa asap. Nilai ADC dikonversi menjadi nilai tegangan keluaran dengan rumus :

$$VRL = \frac{ADC\ x\ 5}{1024} = \frac{25\ x\ 5}{1024} = 0{,}122\ V$$

Dengan menggunakan RL sebesar 10KΩ, tegangan input 5V, maka dapat diperoleh persamaan nilai Rs sebagai berikut :

$$Rs = \frac{Vc - VRL}{VRL}\ x\ RL = \frac{5 - 0{,}122}{0{,}122}\ x\ 10000 = 399836{,}06\ \Omega$$

Keterangan :

Rs = Hambatan sensor (Ω)

RL = Hambatan beban (Ω)

Vc = Tegangan input sensor (Volt)

VRL = Tegangan output sensor (Volt)

Selanjutnya menghitung nilai Ro, perbandingan Rs dan Ro pada saat udara bersih (0 ppm) sebesar 3,6. Maka, mendapatkan nilai Ro dengan persamaan sebgai berikut :

$$Ro = \frac{Rs}{3{,}6} = \frac{399836{,}06}{3{,}6} = 111065{,}572\Omega$$

Untuk menghubungkan nilai rasio dan nilai ppm, maka menggunakan persamaan logaritmik sebagai berikut :

log (y) = m * log (x) + b

Dimana :

y = nilai Y (rasio Rs/Ro)

x = nilai X (ppm)

m = kemiringan garis pada grafik

b = titik persimpangan

Memerlukan 2 titik dari grafik, yaitu (10, 2.3) dan (197, 0.79) dari garis CO_2. Rumus untuk menghitung m adalah :

m = log [log(y) – log (y0)] / [log(x) – log (x0)]

m = log (y/y0) / log (x/x0)

m = log (0.79/2.3) / log (197/10)

m = -0.358

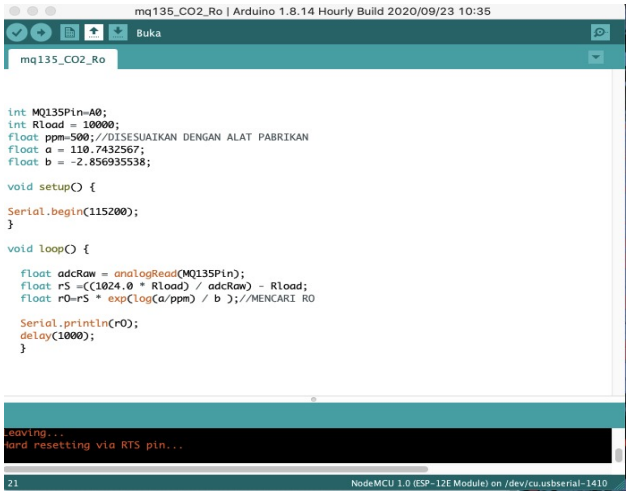
Kemudian menghitung b, maka memerlukan satu titik dari grafik yaitu titik (99, 1.05).

Log (y) = m*log (x) + b

b = log (y) – m*log(x)

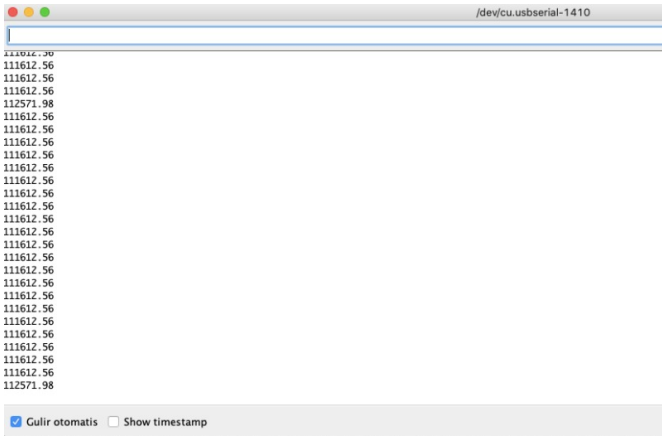
b = log (1.05) – (-0.358) * log (99)

b = 0.735



Gambar 3.7 Pemrograman Arduino IDE

Kemudian debug code program ke microcontroller yang digunakan dan telah terhubung dengan sensor MQ-135.Sehingga diperoleh nilai Ro sebesar 111612.56 Ω. Nilai Ro sudah ditemukan, maka dapat digunakan untuk mencari nilai ppm konsentrasi suatu gas dari keluaran sensor dan nilai Ro bersifat tetap.



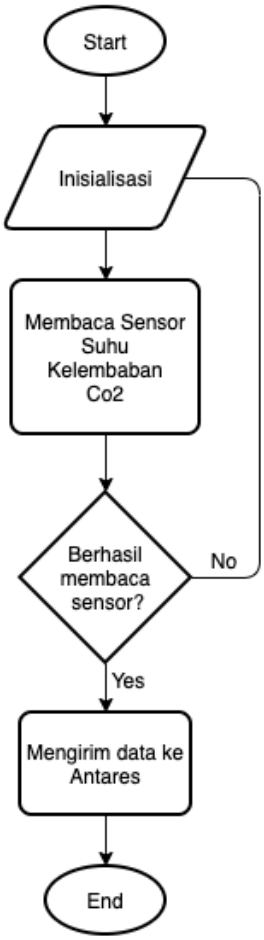
Gambar 3.8 Hasil Nilai Ro

3.2.2.Perangkat Lunak (*Software*)

Pada perancangan perangkat lunak terdapat dua diagram alir, yaitu diagram alir pemrograman pada mikrokontroler dan diagram alir pemrograman pengolahan data dari Antares ke dalam aplikasi di Android. Pemrograman pada mikrokontroler hanya sebatas membaca sensor kemudian dikirim ke *cloud* dan pemrograman pada Android berguna untuk mendapatkan data dari *cloud*.

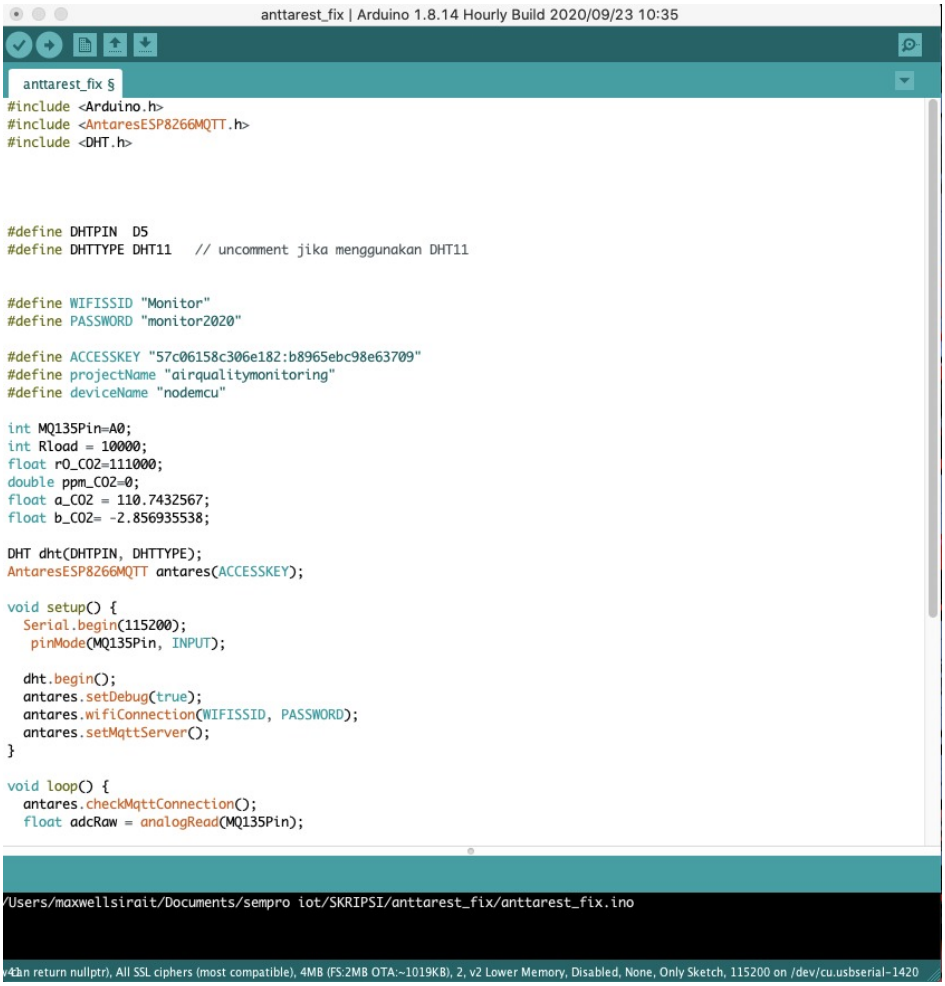
1.Diagram Alir Pada Mikrokontroler

Diagram ini menjelaskan cara kerja program yang ada pada kontroler NodeMCU dimulai dari inisialisasi hingga mengirim data yang dibaca sensor ke *cloud* Antares. Langkah awal adalah inisialisasi *access point* agar terhubung ke internet lalu menentukan pin yang dipakai. Proses selanjutnya adalah data sensor tersebut diunggah ke *cloud*.



Gambar 3.9 Diagram Alir Mikrokontroler

Pemrograman yang dipakai dalam penelitian ini adalah menggunakan Arduino IDE. Arduino merupakan sebuah platform elektronik yang *open source*, berbasis pada *software* dan *hardware* yang fleksibel dan mudah digunakan.

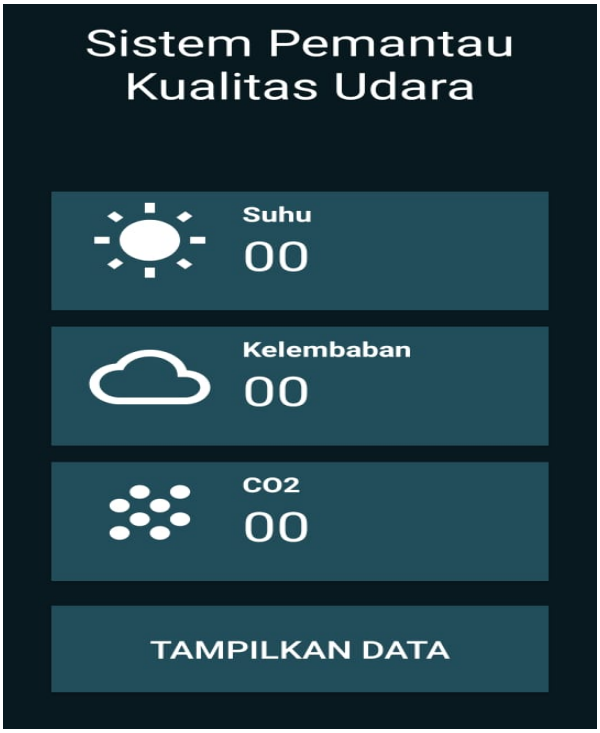


Gambar 3.10 Arduino IDE

2.Diagram Alir Pada Aplikasi Android

Diagram ini menjelaskan cara untuk memperoleh data dari *cloud* Antares agar dapat ditampilkan pada Android. Pemrograman Android terdiri dari dua Bahasa pemrograman, yaitu menggunakan Bahasa Java sebagai Bahasa pemrograman utama untuk menjalankan proses dari aplikasinya dan XML sebagai pemrograman untuk membuat desain tampilan aplikasinya.

Tampilan aplikasi sistem pemantau pada Smartphone Android yang akan dibuat seperti pada gambar 3.11. Pengujian sitem pemantauan ini dilakukan di area perkotaan yang berlokasi di kota Bekasi.

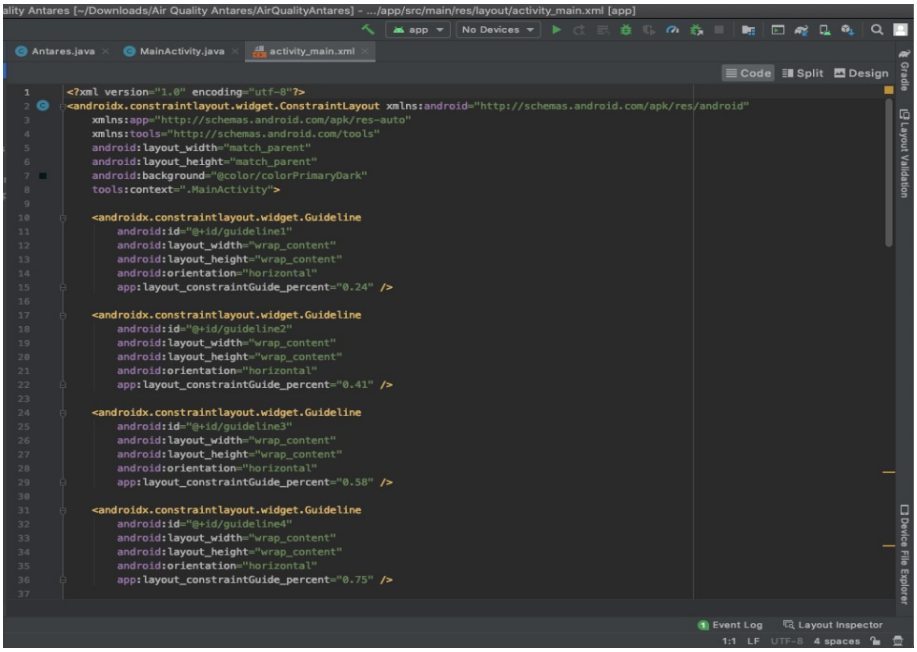


Gambar 3.11 Desain Aplikasi Android

Fitur dan fungsi yang ada pada aplikasi Android, sebagai berikut :

a) Tampilkan Data adalah tombol yang berfungsi untuk menampilkan nilai dari Suhu, Kelembaban, dan CO_2.
b) Suhu (°C) adalah tampilan informasi nilai Suhu yang dibaca sensor dengan satuan (°C).
c) Kelembaban (%RH) adalah tampilan informasi nilai Kelembaban yang dibaca sensor dengan satuan %RH.
d) CO_2 (ppm) adalah tampilan informasi nilai Karbon Dioksida yang dibaca sensor dengan satuan ppm.

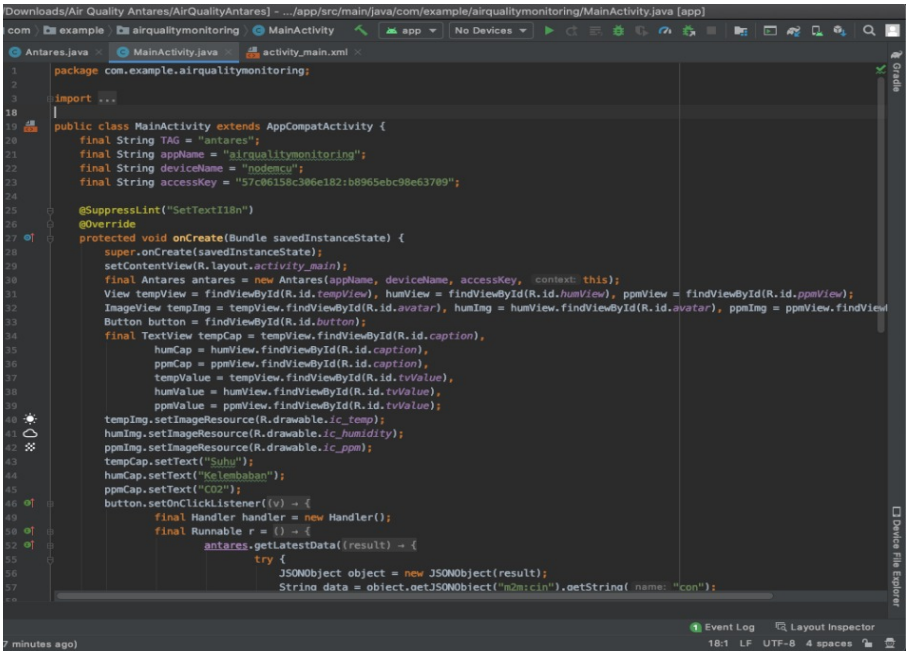
Berikut dibawah ini merupakan tampilan *source code* activity_main.xml sebagai pemrograman untuk desain tampilan halaman utama aplikasinya.



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/colorPrimaryDark"
    tools:context=".MainActivity">

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintGuide_percent="0.24" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintGuide_percent="0.41" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintGuide_percent="0.58" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintGuide_percent="0.75" />
```

Gambar 3.12 Source Code activity_main.xml

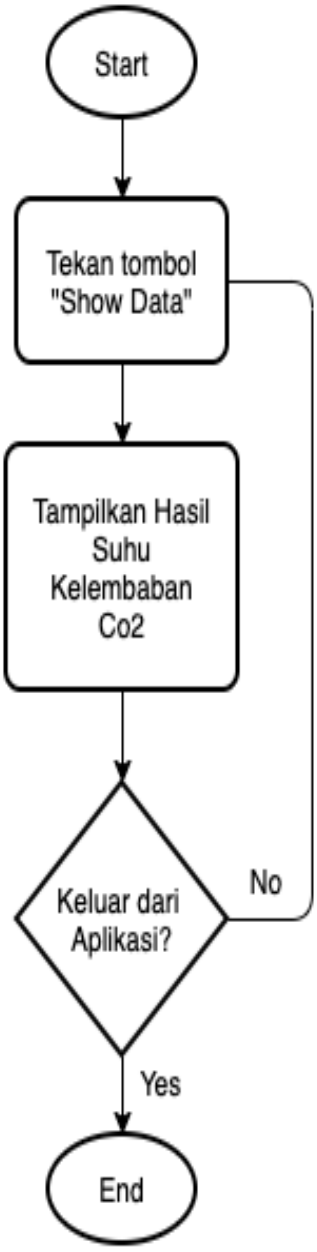
Berikut dibawah ini merupakan tampilan *source code* MainActivity.java sebagai proses aplikasinya untuk tampilan halaman utama pada aplikasi Android yang akan menampilkan data dari Antares.



```
package com.example.airqualitymonitoring;

import ...

public class MainActivity extends AppCompatActivity {
    final String TAG = "antares";
    final String appName = "airqualitymonitoring";
    final String deviceName = "nodemcu";
    final String accessKey = "57c06158c306e182:b8965ebc98e63709";

    @SuppressLint("SetTextI18n")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final Antares antares = new Antares(appName, deviceName, accessKey, context: this);
        View tempView = findViewById(R.id.tempView), humView = findViewById(R.id.humView), ppmView = findViewById(R.id.ppmView);
        ImageView tempImg = tempView.findViewById(R.id.avatar), humImg = humView.findViewById(R.id.avatar), ppmImg = ppmView.findView
        Button button = findViewById(R.id.button);
        final TextView tempCap = tempView.findViewById(R.id.caption),
                humCap = humView.findViewById(R.id.caption),
                ppmCap = ppmView.findViewById(R.id.caption),
                tempValue = tempView.findViewById(R.id.tvValue),
                humValue = humView.findViewById(R.id.tvValue),
                ppmValue = ppmView.findViewById(R.id.tvValue);
        tempImg.setImageResource(R.drawable.ic_temp);
        humImg.setImageResource(R.drawable.ic_humidity);
        ppmImg.setImageResource(R.drawable.ic_ppm);
        tempCap.setText("Suhu");
        humCap.setText("Kelembaban");
        ppmCap.setText("CO2");
        button.setOnClickListener((v) -> {
                final Handler handler = new Handler();
                final Runnable r = () -> {
                        antares.getLatestData((result) -> {
                                try {
                                    JSONObject object = new JSONObject(result);
                                    String data = object.getJSONObject("m2m:cin").getString( name: "con");
```

Gambar 3.13 Source Code MainActivity.java

Berikut dibawah ini merupakan diagram alir dari aplikasi Android yang menjelaskan gambaran cara kerja aplikasi yang dibuat.



Gambar 3.14 Diagram Alir Aplikasi Android