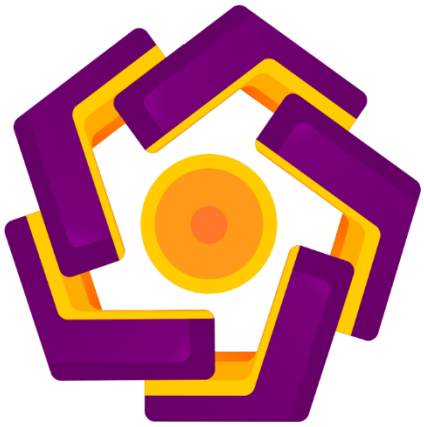
Laporan Final Project Internet of Things

“Sistem Monitoring Suhu dan Kelembapan Ruang Server”



Anggota kelompok:

1. Affan Ardana 20.11.3636
2. Darussalaam Nur Rasyidu 20.11.3637
3. Tonik Purwanto 20.11.3652

Program Studi Informatika

Universitas Amikom Yogyakarta

2022/2023

BAB I

LATAR BELAKANG

Server merupakan salah satu bagian terpenting dari sebuah jaringan komputer. Banyak perusahaan dan organisasi menyimpan data-data penting mereka dalam bentuk digital ke dalam server. Hal ini tidak lain karena data dalam bentuk digital lebih mudah diorganisir dan menghemat sumber daya dibandingkan dengan menyimpan data dalam bentuk fisik. Ditambah lagi data di dalam server mudah diakses oleh seluruh anggota organisasi dimana pun dan kapan pun.

Namun, server yang berisi kumpulan *hard disk* yang merupakan perangkat elektronik akan menghasilkan panas. Cara yang digunakan untuk menangani hal tersebut biasanya adalah dengan menggunakan pendingin ruangan yang digabungkan dengan sistem pendingin lainnya seperti *water cooling*. Kondisi ruangan yang tidak sesuai dapat mengakibatkan kerusakan perangkat sehingga dapat mengakibatkan *downtime* pada jaringan yang pastinya akan merugikan organisasi tersebut. Oleh karena itu, dibutuhkan sebuah sistem yang bisa memantau suhu dan kelembapan ruang server secara *real-time* dan bisa memutuskan kondisi ruang server.

Salah satu solusi yang dapat digunakan adalah menggunakan teknologi *Internet of Things* (IoT). IoT memungkinkan berbagai perangkat elektronik untuk saling berkomunikasi menggunakan internet. Hal ini memungkinkan untuk melakukan pertukaran data antar perangkat secara *real-time*. Data suhu dan kelembapan dapat di simpan secara berkala menggunakan teknologi IoT.

Solusi lainnya adalah penggunaan kecerdasan buatan untuk memutuskan kondisi ruang server. Algoritma yang dapat digunakan untuk menentukan kondisi ruang server yang tidak pasti adalah menggunakan *Fuzzy Logic*. Dengan menggunakan *Fuzzy Logic* sistem dapat memutuskan kondisi ruang server berdasarkan data yang diterima.

BAB II

ANALISIS PROYEK IOT

A. Analisis Kebutuhan Fungsional

Kebutuhan fungsional:

1. Sistem harus dapat terhubung ke internet.
2. Sensor harus dapat mengukur suhu dan kelembapan.
3. Sensor harus bisa menjangkau semua sudut ruangan.
4. Sistem harus bisa mengirimkan data sensor.
5. Sistem harus bisa menyimpan data yang dikirim sensor pada database.
6. Sistem harus bisa menentukan keadaan ruang server.
7. Sistem harus memiliki *user interface*.
8. *User interface* harus menampilkan semua data dari sensor.
9. *User interface* harus menampilkan keadaan ruang server.
10. Sistem harus menampilkan data secara *real-time*.

B. Analisis Kebutuhan Non Fungsional

Kebutuhan non fungsional:

1. *User interface* harus bisa berjalan pada desktop.
2. Sistem harus dapat dikembangkan seperti penambahan sensor.
3. Tampilan *user interface* harus mudah dipahami.
4. Sistem harus bisa menyimpan data pada perangkat lokal.
5. Sistem harus bisa memproses data dengan cepat dan akurat.

C. Analisis Kebutuhan Pengguna

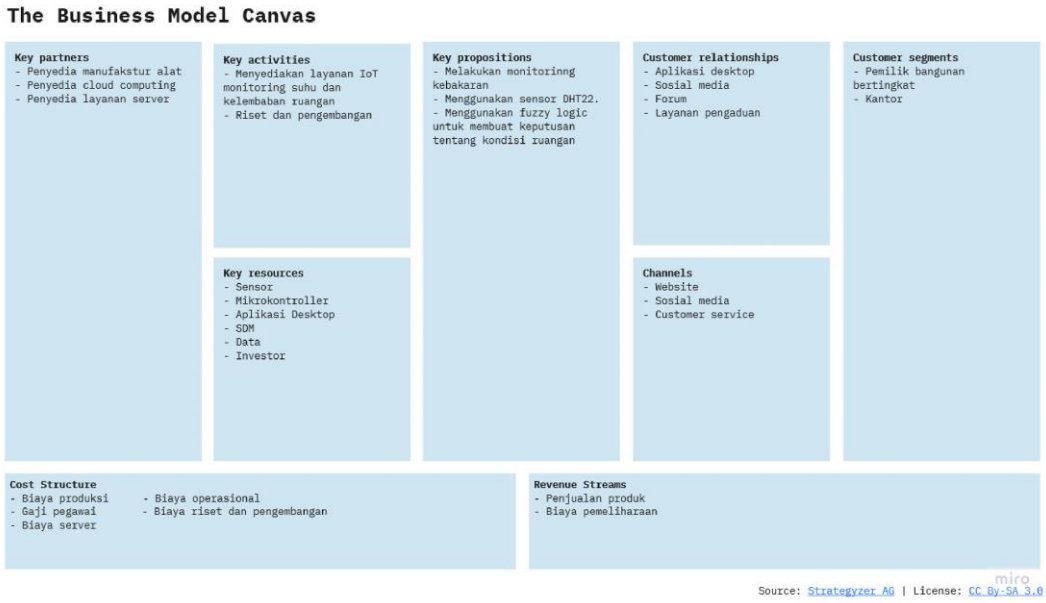
Kebutuhan pengguna:

1. Memberikan informasi yang akurat tentang kondisi di dalam ruang server.
2. Pengguna harus dapat mengakses data suhu dan kelembapan secara *real-time.*

3. Pengguna dapat mengakses tampilan sistem melalui sebuah aplikasi.
4. Pengguna dapat melihat data suhu dan kelembapan yang ada dalam database.
5. Sistem harus mudah digunakan.

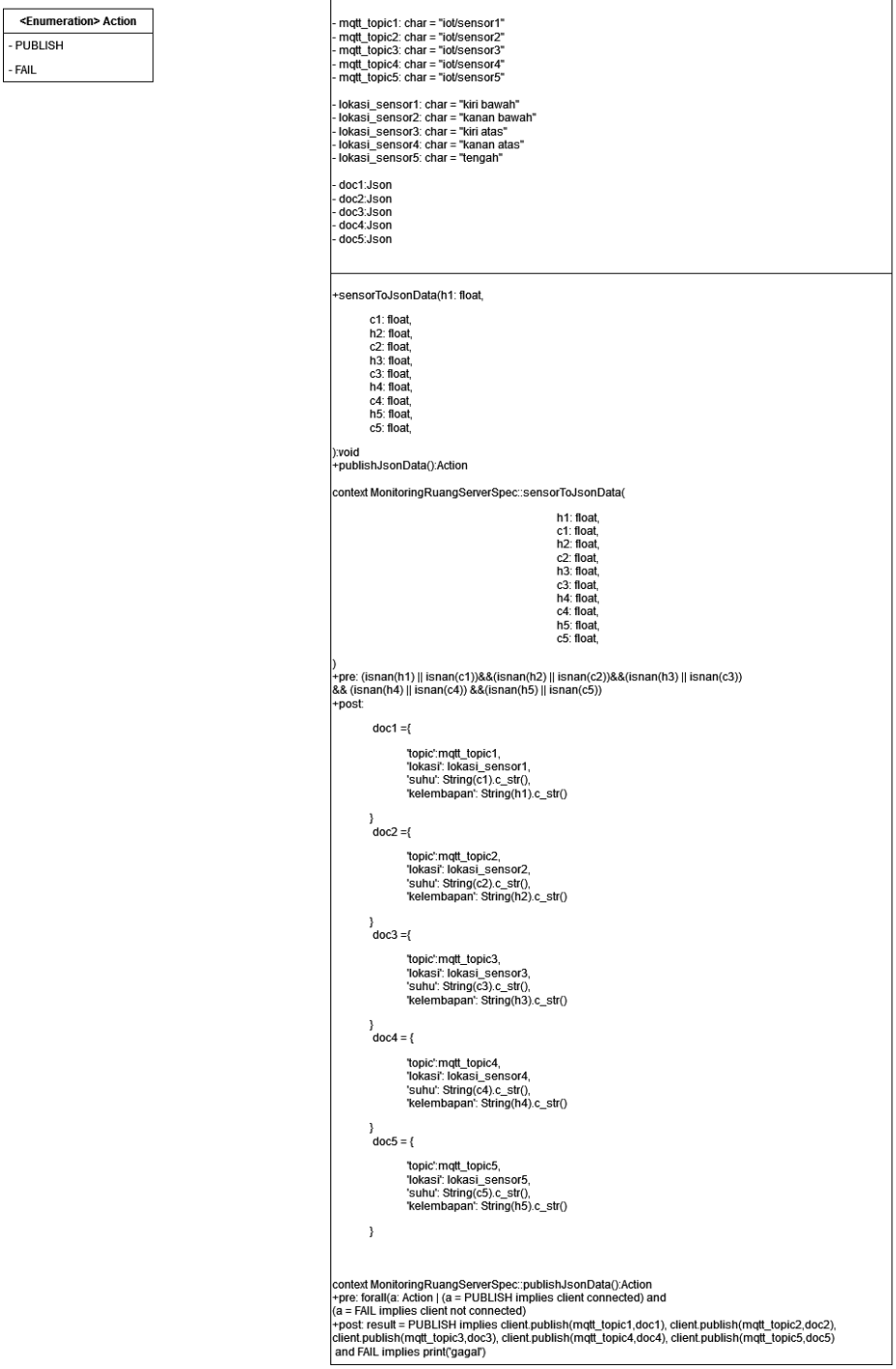
D. Analisis Bisnis

Analisis bisnis dilakukan menggunakan *Business Model canvas* yang ditampilkan sebagai berikut.



E. Analisis Keamanan IoT

Analisis keamanan dilakukan menggunakan Framework ARMET. Berikut adalah penggambaran spesifikasi menggunakan UML.


<Enumeration> Action
- PUBLISH
- FAIL
MonitoringRuangServerWokWiSpec
- mqtt_topic1: char = "iot/sensor1"
- mqtt_topic2: char = "iot/sensor2"
- mqtt_topic3: char = "iot/sensor3"
- mqtt_topic4: char = "iot/sensor4"
- mqtt_topic5: char = "iot/sensor5"
- lokasi_sensor1: char = "kiri bawah"
- lokasi_sensor2: char = "kanan bawah"
- lokasi_sensor3: char = "kiri atas"
- lokasi_sensor4: char = "kanan atas"
- lokasi_sensor5: char = "tengah"
- doc1:Json
- doc2:Json
- doc3:Json
- doc4:Json
- doc5:Json
+sensorToJsonData(h1: float,
c1: float,
h2: float,
c2: float,
h3: float,
c3: float,
h4: float,
c4: float,
h5: float,
c5: float,
):void
+publishJsonData():Action
context MonitoringRuangServerSpec::sensorToJsonData(
h1: float,
c1: float,
h2: float,
c2: float,
h3: float,
c3: float,
h4: float,
c4: float,
h5: float,
c5: float,
)
+pre: (isnan(h1) || isnan(c1))&&(isnan(h2) || isnan(c2))&&(isnan(h3) || isnan(c3))
&& (isnan(h4) || isnan(c4)) &&(isnan(h5) || isnan(c5))
+post:
doc1 ={
'topic':mqtt_topic1,
'lokasi': lokasi_sensor1,
'suhu': String(c1).c_str(),
'kelembapan': String(h1).c_str()
}
doc2 ={
'topic':mqtt_topic2,
'lokasi': lokasi_sensor2,
'suhu': String(c2).c_str(),
'kelembapan': String(h2).c_str()
}
doc3 ={
'topic':mqtt_topic3,
'lokasi': lokasi_sensor3,
'suhu': String(c3).c_str(),
'kelembapan': String(h3).c_str()
}
doc4 = {
'topic':mqtt_topic4,
'lokasi': lokasi_sensor4,
'suhu': String(c4).c_str(),
'kelembapan': String(h4).c_str()
}
doc5 = {
'topic':mqtt_topic5,
'lokasi': lokasi_sensor5,
'suhu': String(c5).c_str(),
'kelembapan': String(h5).c_str()
}
context MonitoringRuangServerSpec::publishJsonData():Action
+pre: forall(a: Action | (a = PUBLISH implies client connected) and
(a = FAIL implies client not connected)
+post: result = PUBLISH implies client.publish(mqtt_topic1,doc1), client.publish(mqtt_topic2,doc2),
client.publish(mqtt_topic3,doc3), client.publish(mqtt_topic4,doc4), client.publish(mqtt_topic5,doc5)
and FAIL implies print('gagal')

SubscriberSpec

```
window:Tk=Tk()
suhu: float
kelembapan: float
tuple: Tuple = (0,0)
tsk: List<Tuple>=[(0,0),(0,0),(0,0),(0,0),(0,0)]
data_sensor_value:Tuple
score:float

+ insertSensorData(data_sensor_value:Tuple):void
+ getLastSensorDataFromDB():void
+ insertMeanSensorData(tuple: Tuple):void
+ getLastMeanSensorData():void
+ fuzzyLogic(suhu:float, kelembapan: float):float
+ update_dashboard(tsk:List<Tuple>, score: float, tuple:Tuple): void

context SubscriberSpec:insertSensorData(data_sensor_value:Tuple)
+ pre: on_message
+ post: cursor.execute("insert into log_sensor (columns) values (data_sensor_value)")

context SubscriberSpec:getLastSensorDataFromData()
+ pre: on_message
+ post: cursor.execute("select * log_sensor order by timestamps desc limit 1")

context SubscriberSpec:insertMeanSensorData(tuple:Tuple)
+ pre: on_message
+ post: cursor.execute("insert into log_mean (columns) values (tuple)")

context SubscriberSpec:getLastMeanSensorData()
+ pre: on_message
+ post: cursor.execute("select * log_mean order by timestamps desc limit 1")

context SubscriberSpec:fuzzyLogic(suhu:float, kelembapan: float)
+ pre: on_message
+ post:

        fuzzyCtrl.input['suhu'] = suhu
        fuzzyCtrl.input['kelembapan'] =kelembapan
        fuzzyCtrl.compute()
        return fuzzyCtrl.output['kondisi']


context SubscriberSpec:update_dashboard(tsk:List<Tuple>, score: float, tuple:Tuple)
+ pre: on_message
+ post: window.display(tsk)
window.display(score)
window.display(tuple)
```

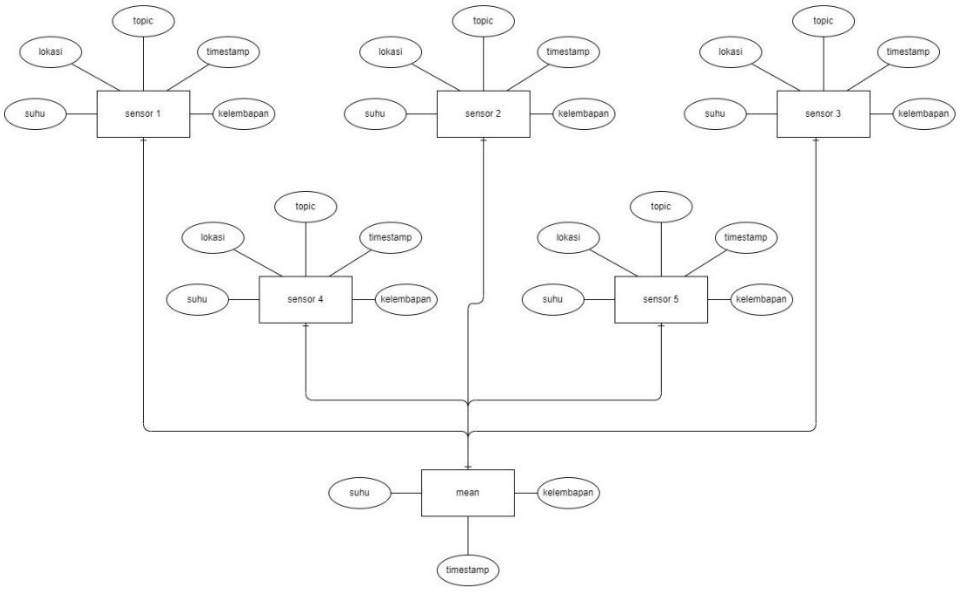
F. Analisis Kebutuhan Service

Protokol yang digunakan pada sistem adalah dengan MQTT. Dengan menggunakan MQTT sistem mendapat beberapa keuntungan dibandingkan dengan menggunakan protokol lain. Keuntungan yang didapat antara lain ukuran pesan yang dikirim kecil sehingga dapat digunakan pada perangkat dengan kapasitas memori terbatas, dapat terhubung dengan berbagai jenis perangkat serta mekanisme komunikasi

data yang memungkinkan perangkat dengan mudah mengirim dan menerima data melalui *broker*.

G. Analisis Kebutuhan Server Database

Analisis kebutuhan server databas dilakukan menggunakan ERD. Berikut adalah rancangan ERD dari database yang menyimpan data dari sensor.



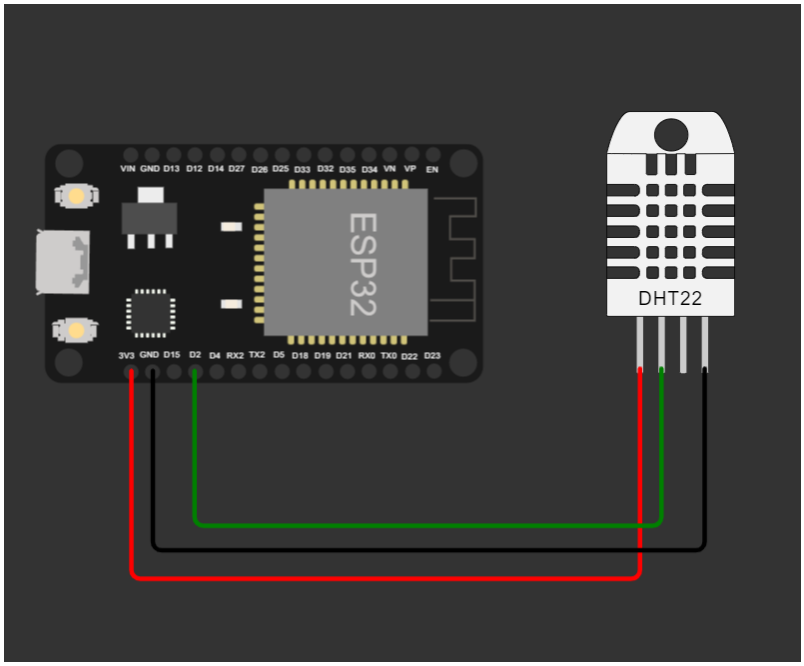
BAB III

RANCANGAN SERVER IOT

A. Desain Topologi Server

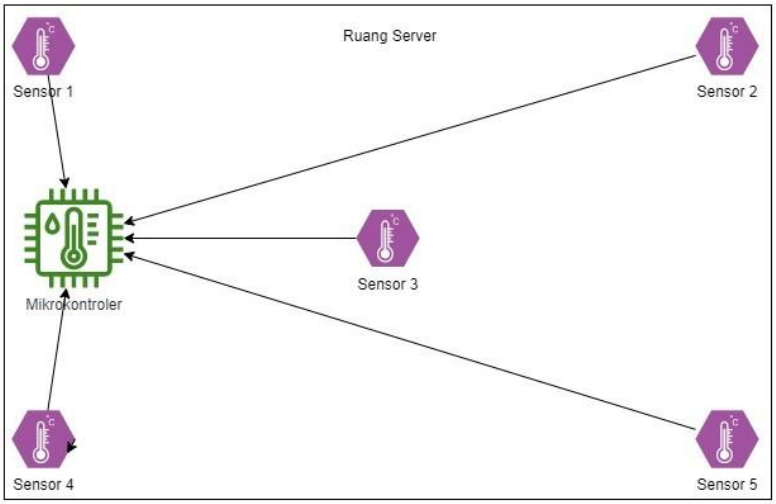
a. Rancangan *Node* Sensor

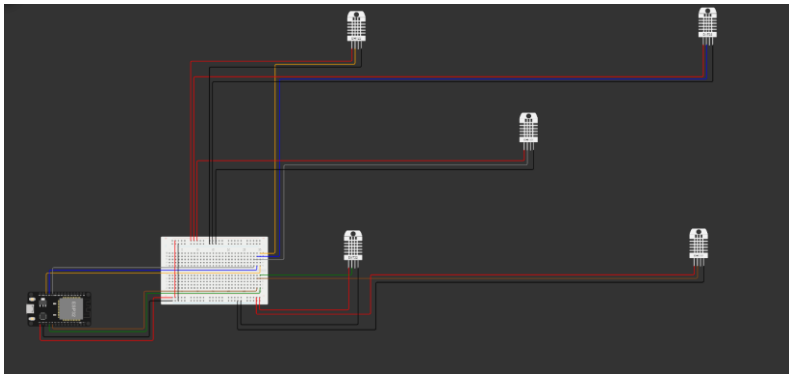
Sensor yang digunakan adalah DHT22. Sensor tersebut dapat membaca suhu dan kelembapan. Sensor dihubungkan dengan mikrokontroler ESP32 menggunakan kabel. Berikut adalah gambar sensor DHT22 yang dihubungkan ke ESP32 pada Wokwi.



b. Rancangan WSN

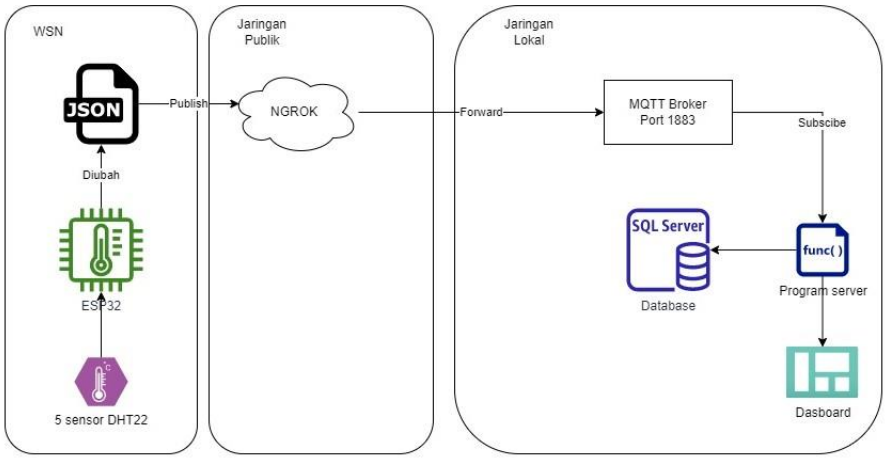
Terdapat lima sensor DHT22 untuk memantau suhu dan kelembapan ruangan server. Sensor diletakkan pada setiap sudut dan pada tengah ruangan. Lima sensor dipakai karena untuk memperbesar cakupan sensor pada seluruh ruang server. Kelima sensor terhubung dengan satu mikrokontroler ESP32 dengan topologi star. Berikut adalah rancangan WSN dan penerapannya pada Wokwi.





c. Rancangan Server IoT

Sensor DHT22 akan membaca keadaan suhu dan kelembapan ruang server. Data dari lima sensor akan diterima oleh ESP32. Pada ESP32 data dari seluruh sensor akan diubah menjadi JSON. Selanjutnya JSON akan dikirim ke jaringan publik melalui WIFI menuju ke NGROK. Data akan dilakukan *forwarding* oleh NGROK menuju port 1883 yang pada jaringan privat. Data yang berbentuk JSON akan di-*subscribe* oleh program server. Berikut adalah rancangan server IoT.



B. Kode Program Server

a. Kode Program Server

Berikut adalah kode pada program server.

```
from paho.mqtt import client as mqtt_client
import skfuzzy as fuzz
from skfuzzy import control as ctrl
import numpy as np
import random
import time
import sqlite3
import json
import matplotlib.pyplot as plt
from tkinter import *
from PIL import Image


broker = "localhost"
port = 1883
topic1 = "iot/sensor1"
topic2 = "iot/sensor2"
topic3 = "iot/sensor3"
topic4 = "iot/sensor4"
topic5 = "iot/sensor5"
client_id = f'python-mqtt-{random.randint(0, 100)}'
```

```
# fuzzy logic
# antecedent -> consequent
suhu = ctrl.Antecedent(np.arange(0, 34, 1), 'suhu')
kelembapan = ctrl.Antecedent(np.arange(0, 101, 1),
'kelembapan')
kondisi = ctrl.Consequent(np.arange(0, 11, 1), 'kondisi')

# variabel fuzzy untuk suhu
suhu['rendah'] = fuzz.trimf(suhu.universe, [0, 0, 18])
suhu['sedang'] = fuzz.trimf(suhu.universe, [13, 18, 23])
suhu['tinggi'] = fuzz.trimf(suhu.universe, [18, 33, 33])

# variabel fuzzy untuk kelembapan
kelembapan['rendah'] = fuzz.trimf(kelembapan.universe, [0,
0, 50])
kelembapan['sedang'] = fuzz.trimf(kelembapan.universe, [30,
50, 70])
kelembapan['tinggi'] = fuzz.trimf(kelembapan.universe, [50,
100, 100])

# variabel fuzzy untuk kondisi
kondisi['buruk'] = fuzz.trimf(kondisi.universe, [0, 0, 5])
kondisi['lumayan baik'] = fuzz.trimf(kondisi.universe, [2,
5, 8])
kondisi['baik'] = fuzz.trimf(kondisi.universe, [5, 10, 10])

# rules
rule1 = ctrl.Rule(suhu['rendah'] & kelembapan['rendah'],
kondisi['buruk'])
rule2 = ctrl.Rule(suhu['rendah'] & kelembapan['sedang'],
                  kondisi['lumayan baik'])
rule3 = ctrl.Rule(suhu['rendah'] & kelembapan['tinggi'],
kondisi['buruk'])
rule4 = ctrl.Rule(suhu['sedang'] & kelembapan['rendah'],
                  kondisi['lumayan baik'])
rule5 = ctrl.Rule(suhu['sedang'] & kelembapan['sedang'],
kondisi['baik'])
rule6 = ctrl.Rule(suhu['sedang'] & kelembapan['tinggi'],
```

```
                  kondisi['lumayan baik'])
rule7 = ctrl.Rule(suhu['tinggi'] & kelembapan['rendah'],
kondisi['buruk'])
rule8 = ctrl.Rule(suhu['tinggi'] & kelembapan['sedang'],
                  kondisi['lumayan baik'])
rule9 = ctrl.Rule(suhu['tinggi'] & kelembapan['tinggi'],
kondisi['buruk'])

kondisi_ctrl = ctrl.ControlSystem([
    rule1, rule2, rule3,
    rule4, rule5, rule6,
    rule7, rule8, rule9
])

kondisi_fuzzy = ctrl.ControlSystemSimulation(kondisi_ctrl)


# dashboard
# window
window = Tk()
window.title("MQTT Dashboard")
window.geometry('1130x600')  # Width, Height
window.resizable(False, False)  # Width, Height
window.configure(bg="white")

# suhu
# gambar suhu
rezizer = Image.open("suhu_img.png")
rezizer = rezizer.resize((50, 50))
rezizer.save("suhu_img_resized.png")

# kelembapan
# gambar kelembapan
rezizer = Image.open("hum_img.png")
rezizer = rezizer.resize((40, 40))
rezizer.save("hum_img_resized.png")

# gambar pemisah
```

```
rezizer = Image.open("pembatas.png")
rezizer = rezizer.resize((30, 600))
rezizer.save("pembatas_resized.png")

# kelembapan
# gambar kelembapan
rezizer = Image.open("hum_img.png")
rezizer = rezizer.resize((40, 40))
rezizer.save("hum_img_resized.png")

# box
# gambar box
rezizer = Image.open("box.png")
rezizer = rezizer.resize((210, 130))
rezizer.save("box_resized.png")

# textBox
# gambar textBox
rezizer = Image.open("textBox.png")
rezizer = rezizer.resize((340, 75))
rezizer.save("textBox_resized.png")

# boxInformation
# gambar boxInformation
rezizer = Image.open("boxInformation.png")
rezizer = rezizer.resize((340, 374))
rezizer.save("boxInformation_resized.png")


# baik
# gambar baik
rezizer = Image.open("baik.png")
rezizer = rezizer.resize((340, 75))
rezizer.save("baik_resized.png")


# lumayan_baik
# gambar lumayan_baik
```

```
rezizer = Image.open("lumayan_baik.png")
rezizer = rezizer.resize((340, 75))
rezizer.save("lumayan_baik_resized.png")


# buruk
# gambar buruk
rezizer = Image.open("buruk.png")
rezizer = rezizer.resize((340, 75))
rezizer.save("buruk_resized.png")

# circle
# gambar circle
rezizer = Image.open("circle.png")
rezizer = rezizer.resize((100, 107))
rezizer.save("circle_resized.png")

# background
# gambar background
rezizer = Image.open("background.png")
rezizer = rezizer.resize((1130, 600))
rezizer.save("background_resized.png")


# resize gambar
img_suhu = PhotoImage(file="suhu_img_resized.png")
img_kelembaban = PhotoImage(file="hum_img_resized.png")
img_pembatas = PhotoImage(file='pembatas_resized.png')
img_box = PhotoImage(file='box_resized.png')
img_textBox = PhotoImage(file='textBox_resized.png')
img_background = PhotoImage(file='background_resized.png')
img_boxInformation =
PhotoImage(file='boxInformation_resized.png')
img_baik = PhotoImage(file='baik_resized.png')
img_lumayan_baik =
PhotoImage(file='lumayan_baik_resized.png')
img_buruk = PhotoImage(file='buruk_resized.png')
img_circle = PhotoImage(file='circle_resized.png')
```

```
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


# buat database
con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_sensor1 = '''CREATE TABLE IF NOT EXISTS
log_sensor1 (
topic TEXT NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
lokasi TEXT NOT NULL,
suhu REAL NOT NULL,
kelembapan REAL NOT NULL);'''
cur.execute(buat_tabel_log_sensor1)
con.commit()

con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_sensor2 = '''CREATE TABLE IF NOT EXISTS
log_sensor2 (
topic TEXT NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
lokasi TEXT NOT NULL,
suhu REAL NOT NULL,
kelembapan REAL NOT NULL);'''
cur.execute(buat_tabel_log_sensor2)
```

```
con.commit()

con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_sensor3 = '''CREATE TABLE IF NOT EXISTS
log_sensor3 (
topic TEXT NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
lokasi TEXT NOT NULL,
suhu REAL NOT NULL,
kelembapan REAL NOT NULL);'''
cur.execute(buat_tabel_log_sensor3)
con.commit()

con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_sensor4 = '''CREATE TABLE IF NOT EXISTS
log_sensor4 (
topic TEXT NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
lokasi TEXT NOT NULL,
suhu REAL NOT NULL,
kelembapan REAL NOT NULL);'''
cur.execute(buat_tabel_log_sensor4)
con.commit()

con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_sensor5 = '''CREATE TABLE IF NOT EXISTS
log_sensor5 (
topic TEXT NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
lokasi TEXT NOT NULL,
suhu REAL NOT NULL,
kelembapan REAL NOT NULL);'''
cur.execute(buat_tabel_log_sensor5)
con.commit()
```

```
con = sqlite3.connect("log_sensor.sqlite")
cur = con.cursor()
buat_tabel_log_gabungan = '''CREATE TABLE IF NOT EXISTS
log_mean (
mean_suhu bawah REAL NOT NULL,
mean_kelembapan REAL NOT NULL,
timestamp DATETIME DEFAULT CURRENT_TIMESTAMP);'''
cur.execute(buat_tabel_log_gabungan)
con.commit()


def subscribe(client: mqtt_client):
    def on_message(client, userdata, msg):

        print(f"Received {msg.payload.decode()} from
{msg.topic} topic")
        data = json.loads(msg.payload.decode())
        topic = data['topic']
        timestamp = time.strftime('%Y-%m-%d %H:%M:%S')
        lokasi = data['lokasi']
        suhu = data['suhu']
        kelembapan = data['kelembapan']

        con = sqlite3.connect("log_sensor.sqlite")
        cur = con.cursor()

        data_sensor_val = (topic, timestamp, lokasi, suhu,
kelembapan)
        if (topic == topic1):
            cur.execute(
                "INSERT INTO log_sensor1 (topic, timestamp,
lokasi, suhu, kelembapan) VALUES (?, ?, ?, ?, ?);",
data_sensor_val)
            con.commit()
        elif (topic == topic2):
            cur.execute(
```

```
                "INSERT INTO log_sensor2 (topic, timestamp,
lokasi, suhu, kelembapan) VALUES (?, ?, ?, ?, ?);",
data_sensor_val)
            con.commit()
        elif (topic == topic3):
            cur.execute(
                "INSERT INTO log_sensor3 (topic, timestamp,
lokasi, suhu, kelembapan) VALUES (?, ?, ?, ?, ?);",
data_sensor_val)
            con.commit()
        elif (topic == topic4):
            cur.execute(
                "INSERT INTO log_sensor4 (topic, timestamp,
lokasi, suhu, kelembapan) VALUES (?, ?, ?, ?, ?);",
data_sensor_val)
            con.commit()
        elif (topic == topic5):
            cur.execute(
                "INSERT INTO log_sensor5 (topic, timestamp,
lokasi, suhu, kelembapan) VALUES (?, ?, ?, ?, ?);",
data_sensor_val)
            con.commit()
            cur.execute(
                """
                INSERT INTO log_mean (mean_suhu,
mean_kelembapan)
                VALUES (
                    ((select suhu from log_sensor1 order by
timestamp DESC limit 1)+
                    (select suhu from log_sensor3 order by
timestamp DESC limit 1)+
                    (select suhu from log_sensor4 order by
timestamp DESC limit 1)+
                    (select suhu from log_sensor5 order by
timestamp DESC limit 1)+
                    (select suhu from log_sensor2 order by
timestamp DESC limit 1))/5,
```

```
                    ((select kelembapan from log_sensor1
order by timestamp DESC limit 1)+
                    (select kelembapan from log_sensor2
order by timestamp DESC limit 1)+
                    (select kelembapan from log_sensor3
order by timestamp DESC limit 1)+
                    (select kelembapan from log_sensor4
order by timestamp DESC limit 1)+
                    (select kelembapan from log_sensor5
order by timestamp DESC limit 1))/5
                );""")
            con.commit()
            cur.execute("""
                select mean_suhu, mean_kelembapan from
log_mean order by timestamp DESC limit 1
            """)
            con.commit()

            tuple = cur.fetchone()
            kondisi_fuzzy.input['suhu'] = tuple[0]
            kondisi_fuzzy.input['kelembapan'] = tuple[1]
            tsk = [(0, 0), (0, 0), (0, 0), (0, 0), (0, 0)]
            for i in range(5):
                cur.execute("select suhu, kelembapan from
log_sensor" +
                            str(i+1)+" order by timestamp
DESC limit 1")
                con.commit()
                tsk[i] = cur.fetchone()

            kondisi_fuzzy.compute()
            score = kondisi_fuzzy.output['kondisi']
            print(score)
            update_dashboard(tsk, score, tuple)

            if (score < 3.33):
                print('Kondisi ruangan buruk')
            elif (score < 6.67):
```

```
                print('kondisi ruangan lumayan baik')
            else:
                print('kondisi ruangan baik')

    client.subscribe(topic1)
    client.subscribe(topic2)
    client.subscribe(topic3)
    client.subscribe(topic4)
    client.subscribe(topic5)
    client.on_message = on_message


def create_dashboard():
    canvas_b = Canvas(window, bg='#aaaaaa',
                      highlightthickness=0, width=1130,
height=600)
    canvas_b.place(x=0, y=0)
    canvas_b.create_image(0, 0, anchor=NW,
image=img_background)

    # sensor 1
    canvas_b.create_image(20, 440, anchor=NW, image=img_box)
    canvas_b.create_image(35, 460, anchor=NW,
image=img_suhu)
    canvas_b.create_image(40, 510, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(90, 485, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(87, 535, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 2
    canvas_b.create_image(500, 440, anchor=NW,
image=img_box)
    canvas_b.create_image(515, 460, anchor=NW,
image=img_suhu)
```

```
    canvas_b.create_image(520, 510, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(570, 480, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(567, 535, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 3
    canvas_b.create_image(20, 30, anchor=NW, image=img_box)
    canvas_b.create_image(35, 50, anchor=NW, image=img_suhu)
    canvas_b.create_image(40, 100, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(90, 75, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(87, 125, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 4
    canvas_b.create_image(500, 30, anchor=NW, image=img_box)
    canvas_b.create_image(515, 50, anchor=NW,
image=img_suhu)
    canvas_b.create_image(520, 100, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(570, 70, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(567, 120, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 5
    canvas_b.create_image(260, 235, anchor=NW,
image=img_box)
```

```
    canvas_b.create_image(275, 255, anchor=NW,
image=img_suhu)
    canvas_b.create_image(280, 305, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(325, 275, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(322, 325, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    canvas_b.create_image(730, 0, anchor=NW,
image=img_pembatas)

    canvas_b.create_image(770, 10, anchor=NW,
image=img_textBox)

    canvas_b.create_text(940, 65, text="Kondisi Ruangan",
font=(
        "Helvetica", 20), fill="white", anchor="s")

    canvas_b.create_image(770, 111, anchor=NW,
image=img_baik)
    canvas_b.create_text(940, 166, text="...", font=(
        "Helvetica", 20), fill="white", anchor="s")

    canvas_b.create_image(770, 206, anchor=NW,
image=img_boxInformation)
    canvas_b.create_text(940, 236, text="Rata-rata",
                         font=("Helvetica", 16),
fill="white", anchor="s")

    canvas_b.create_image(822, 276, anchor=NW,
image=img_circle)
    canvas_b.create_image(957, 276, anchor=NW,
image=img_circle)
    canvas_b.create_image(847, 306, anchor=NW,
image=img_suhu)
```

```
    canvas_b.create_image(987, 306, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(832, 416, text='...'+" °C",
                         font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(972, 416, text='...'+" %",
                         font=("Helvetica", 20),
fill="white", anchor="w")

    canvas_b.create_text(822, 526, text="Skor = "+'...',
                         font=("Helvetica", 20),
fill="white", anchor="w")


def update_dashboard(tsk, score, tskr):
    canvas_b = Canvas(window, bg='#aaaaaa',
                      highlightthickness=0, width=1130,
height=600)
    canvas_b.place(x=0, y=0)
    canvas_b.create_image(0, 0, anchor=NW,
image=img_background)

    # sensor 1
    canvas_b.create_image(20, 440, anchor=NW, image=img_box)
    canvas_b.create_image(35, 460, anchor=NW,
image=img_suhu)
    canvas_b.create_image(40, 510, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(90, 485, text=str(
        tsk[0][0])+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(87, 535, text=str(
        tsk[0][1])+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 2
    canvas_b.create_image(500, 440, anchor=NW,
image=img_box)
```

```
    canvas_b.create_image(515, 460, anchor=NW,
image=img_suhu)
    canvas_b.create_image(520, 510, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(570, 480, text=str(
        tsk[1][0])+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(567, 535, text=str(
        tsk[1][1])+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 3
    canvas_b.create_image(20, 30, anchor=NW, image=img_box)
    canvas_b.create_image(35, 50, anchor=NW, image=img_suhu)
    canvas_b.create_image(40, 100, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(90, 75, text=str(
        tsk[2][0])+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(87, 125, text=str(
        tsk[2][1])+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 4
    canvas_b.create_image(500, 30, anchor=NW, image=img_box)
    canvas_b.create_image(515, 50, anchor=NW,
image=img_suhu)
    canvas_b.create_image(520, 100, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(570, 70, text=str(
        tsk[3][0])+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(567, 120, text=str(
        tsk[3][1])+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    # sensor 5
```

```
    canvas_b.create_image(260, 235, anchor=NW,
image=img_box)
    canvas_b.create_image(275, 255, anchor=NW,
image=img_suhu)
    canvas_b.create_image(280, 305, anchor=NW,
image=img_kelembaban)
    canvas_b.create_text(325, 275, text=str(
        tsk[4][0])+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(322, 325, text=str(
        tsk[4][1])+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    canvas_b.create_image(730, 0, anchor=NW,
image=img_pembatas)

    canvas_b.create_image(770, 10, anchor=NW,
image=img_textBox)

    canvas_b.create_text(940, 65, text="Kondisi Ruangan",
font=(
        "Helvetica", 20), fill="white", anchor="s")

    if (score < 3.33):
        canvas_b.create_image(770, 111, anchor=NW,
image=img_buruk)
        canvas_b.create_text(940, 166, text="Buruk", font=(
            "Helvetica", 20), fill="white", anchor="s")
    elif (score < 6.67):
        canvas_b.create_image(770, 111, anchor=NW,
image=img_lumayan_baik)
        canvas_b.create_text(940, 166, text="Lumayan Baik",
font=(
            "Helvetica", 20), fill="white", anchor="s")
    else:
        canvas_b.create_image(770, 111, anchor=NW,
image=img_baik)
        canvas_b.create_text(940, 166, text="Baik", font=(
```

```
            "Helvetica", 20), fill="white", anchor="s")

    canvas_b.create_image(770, 206, anchor=NW,
image=img_boxInformation)
    canvas_b.create_text(940, 236, text="Rata-rata",
                         font=("Helvetica", 16),
fill="white", anchor="s")

    canvas_b.create_image(822, 276, anchor=NW,
image=img_circle)
    canvas_b.create_image(957, 276, anchor=NW,
image=img_circle)
    canvas_b.create_image(847, 306, anchor=NW,
image=img_suhu)
    canvas_b.create_image(987, 306, anchor=NW,
image=img_kelembaban)

    canvas_b.create_text(832, 416, text=str(
        round(tskr[0], 2))+" °C", font=("Helvetica", 20),
fill="white", anchor="w")
    canvas_b.create_text(972, 416, text=str(
        round(tskr[1], 2))+" %", font=("Helvetica", 20),
fill="white", anchor="w")

    canvas_b.create_text(822, 526, text="Skor =
"+str(round(score, 2)),
                         font=("Helvetica", 20),
fill="white", anchor="w")


def run():
    client = connect_mqtt()
    subscribe(client)
    create_dashboard()
    client.loop_start()
    window.mainloop()
    client.loop_stop()
```

```
if __name__ == '__main__':
    # suhu.view()
    # kelembapan.view()
    # kondisi.view()
    # plt.show()

    run()
```

b. Kode Program Node Sensor

Berikut adalah kode mikrokontroler pada Wokwi.

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <PubSubClient.h>
#include <DHT.h>
#include <ArduinoJson.h>

#define MAX_CHARACTER 50
#define MQTT_SERVER "0.tcp.ap.ngrok.io"

char ssid[] = "Wokwi-GUEST";
char pass[] = "";

int mqtt_port = 19688;

char mqtt_topic1[] = "iot/sensor1";
char mqtt_topic2[] = "iot/sensor2";
char mqtt_topic3[] = "iot/sensor3";
char mqtt_topic4[] = "iot/sensor4";
char mqtt_topic5[] = "iot/sensor5";

DHT dht1(2, DHT22); // kiri bawah
DHT dht2(4, DHT22); // kanan bawah
DHT dht3(13, DHT22); // kiri atas
DHT dht4(12, DHT22); // kanan atas
DHT dht5(14, DHT22); // tengah
```

```
char lokasi_sensor1[] = "kiri bawah";
char lokasi_sensor2[] = "kanan bawah";
char lokasi_sensor3[] = "kiri atas";
char lokasi_sensor4[] = "kanan atas";
char lokasi_sensor5[] = "tengah";

WiFiClient espClient;
PubSubClient client(espClient);

void setupWifi(){
  Serial.print("Menghubungkan ke ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED){
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.print("Terhubung ke ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println("");
}

void setupMqtt(){
  while (!client.connected()){
    Serial.println("Menghubungkan ke MQTT...");

    String idClient = "client-";
    idClient += String(random(0xffff), HEX);
```

```
    if (client.connect(idClient.c_str())){
      Serial.println("MQTT terhubung");
      Serial.println();
    }
    else{
      Serial.print("Error: ");
      Serial.print(client.state());
      Serial.println("Mencoba lagi...");
      delay(5000);
    }
  }
}

void setup() {
  pinMode(2, INPUT);

  Serial.begin(9600);

  setupWifi();

  client.setServer(MQTT_SERVER, mqtt_port);

  dht1.begin();
  dht2.begin();
  dht3.begin();
  dht4.begin();
  dht5.begin();
}

void loop() {
  if (!client.connected()){
    setupMqtt();
  }

  client.loop();

  float h1 = dht1.readHumidity();
  float c1 = dht1.readTemperature();
```

```
  float h2 = dht2.readHumidity();
  float c2 = dht2.readTemperature();
  float h3 = dht3.readHumidity();
  float c3 = dht3.readTemperature();
  float h4 = dht4.readHumidity();
  float c4 = dht4.readTemperature();
  float h5 = dht5.readHumidity();
  float c5 = dht5.readTemperature();

  if (isnan(h1) || isnan(c1)) {
    Serial.println(F("Sensor 1 tidak terbaca!"));
    return;
  }

  if (isnan(h2) || isnan(c2)) {
    Serial.println(F("Sensor 2 tidak terbaca!"));
    return;
  }

  if (isnan(h3) || isnan(c3)) {
    Serial.println(F("Sensor 3 tidak terbaca!"));
    return;
  }

  if (isnan(h4) || isnan(c4)) {
    Serial.println(F("Sensor 4 tidak terbaca!"));
    return;
  }

  if (isnan(h5) || isnan(c5)) {
    Serial.println(F("Sensor 5 tidak terbaca!"));
    return;
  }

// publish sensor 1
String celcius1 = String(c1).c_str();
String humidity1 = String(h1).c_str();
StaticJsonDocument<200> doc1;
```

```
doc1["topic"] = mqtt_topic1;
doc1["lokasi"] = lokasi_sensor1;
doc1["suhu"] = celcius1;
doc1["kelembapan"] = humidity1;

String jsonString1;
serializeJson(doc1, jsonString1);
char json1[jsonString1.length() + 1];
jsonString1.toCharArray(json1, jsonString1.length() + 1);

if (client.publish(mqtt_topic1, json1)) {
  Serial.println("Data berhasil dikirim ke topic " +
String(mqtt_topic1));
} else {
  Serial.println("Gagal mengirim data ke topic " +
String(mqtt_topic1));
}

// publish sensor 2
String celcius2 = String(c2).c_str();
String humidity2 = String(h2).c_str();
StaticJsonDocument<200> doc2;

doc2["topic"] = mqtt_topic2;
doc2["lokasi"] = lokasi_sensor2;
doc2["suhu"] = celcius2;
doc2["kelembapan"] = humidity2;

String jsonString2;
serializeJson(doc2, jsonString2);
char json2[jsonString2.length() + 1];
jsonString2.toCharArray(json2, jsonString2.length() + 1);

if (client.publish(mqtt_topic2, json2)) {
  Serial.println("Data berhasil dikirim ke topic " +
String(mqtt_topic2));
} else {
```

```
  Serial.println("Gagal mengirim data ke topic " +
String(mqtt_topic2));
}

// publish sensor 3
String celcius3 = String(c3).c_str();
String humidity3 = String(h3).c_str();
StaticJsonDocument<200> doc3;

doc3["topic"] = mqtt_topic3;
doc3["lokasi"] = lokasi_sensor3;
doc3["suhu"] = celcius3;
doc3["kelembapan"] = humidity3;

String jsonString3;
serializeJson(doc3, jsonString3);
char json3[jsonString3.length() + 1];
jsonString3.toCharArray(json3, jsonString3.length() + 1);

if (client.publish(mqtt_topic3, json3)) {
  Serial.println("Data berhasil dikirim ke topic " +
String(mqtt_topic3));
} else {
  Serial.println("Gagal mengirim data ke topic " +
String(mqtt_topic3));
}

// publish sensor 4
String celcius4 = String(c4).c_str();
String humidity4 = String(h4).c_str();
StaticJsonDocument<200> doc4;

doc4["topic"] = mqtt_topic4;
doc4["lokasi"] = lokasi_sensor4;
doc4["suhu"] = celcius4;
doc4["kelembapan"] = humidity4;

String jsonString4;
```

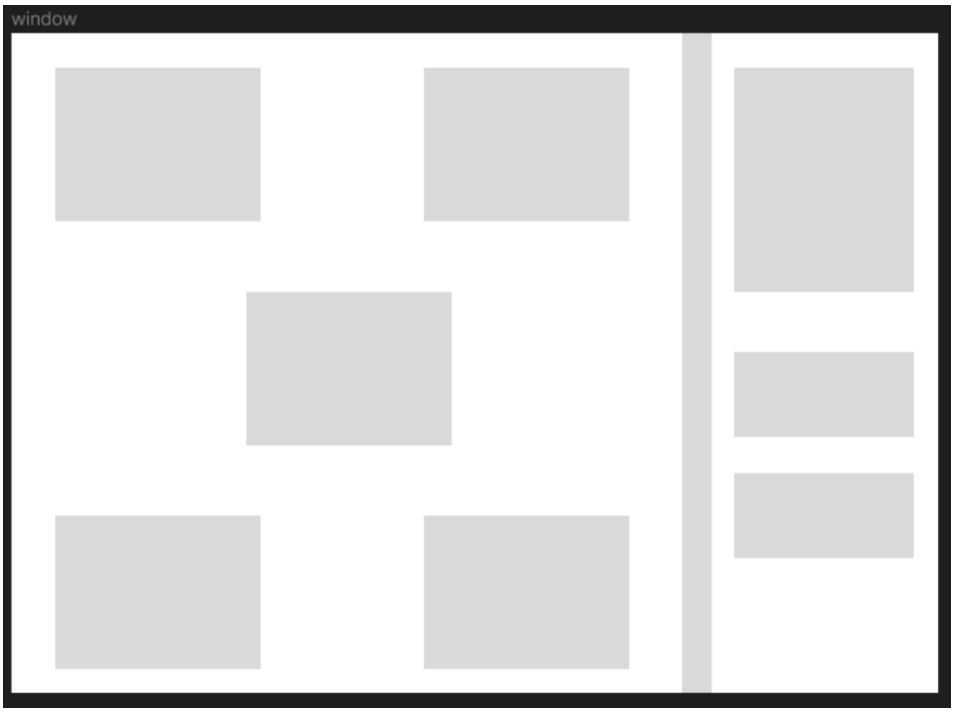
```
serializeJson(doc4, jsonString4);
char json4[jsonString4.length() + 1];
jsonString4.toCharArray(json4, jsonString4.length() + 1);

if (client.publish(mqtt_topic4, json4)) {
  Serial.println("Data berhasil dikirim ke topic " +
String(mqtt_topic4));
} else {
  Serial.println("Gagal mengirim data ke topic " +
String(mqtt_topic4));
}

// publish sensor 5
String celcius5 = String(c5).c_str();
String humidity5 = String(h5).c_str();
StaticJsonDocument<200> doc5;

doc5["topic"] = mqtt_topic5;
doc5["lokasi"] = lokasi_sensor5;
doc5["suhu"] = celcius5;
doc5["kelembapan"] = humidity5;

String jsonString5;
serializeJson(doc5, jsonString5);
char json5[jsonString5.length() + 1];
jsonString5.toCharArray(json5, jsonString5.length() + 1);

if (client.publish(mqtt_topic5, json5)) {
  Serial.println("Data berhasil dikirim ke topic " +
String(mqtt_topic5));
} else {
  Serial.println("Gagal mengirim data ke topic " +
String(mqtt_topic5));
}

delay(5000);
}
```

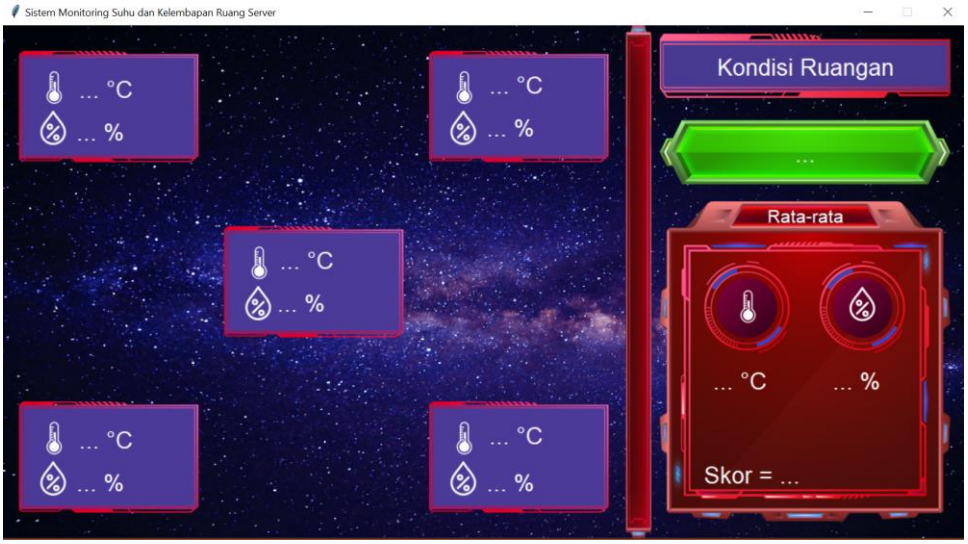
BAB IV

RANCANGAN DASHBOARD IOT

A. Wireframe Dashboard IoT

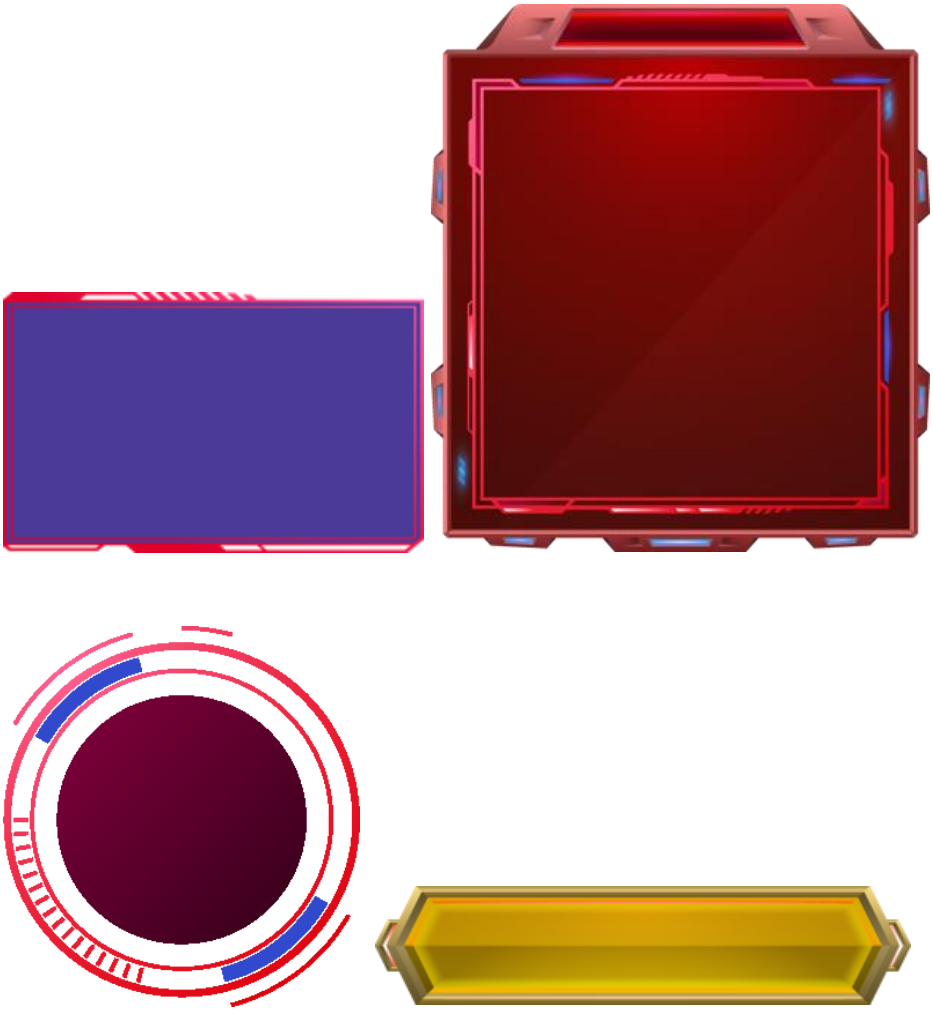
Berdasarkan analisis kebutuhan, ada beberapa komponen yang harus ada pada *dashboard*. Kebutuhan yang harus ada antara lain yaitu semua data suhu dan kelembapan dari semua sensor, rata-rata suhu dan kelembapan yang dihitung dari semua sensor, dan kondisi ruangan yang didapat dari hasil proses menggunakan Logika Fuzzy. *Dashboard* menampilkan semua data suhu dan kelembapan yang dikirim dari semua sensor secara *real-time*. Hal ini sejalan dengan kebutuhan fungsional dan non fungsional yang telah dijabarkan sebelumnya. Berikut merupakan *layout* dan tampilan *dashboard*.





B. Desain Asset

Semua aset yang ditampilkan pada *dashboard* didapat dari internet. Gambar aset yang ukurannya belum sesuai dengan yang diharapkan akan diubah ukurannya menggunakan *library* PIL pada Python dan jika diperlukan akan dilakukan pengeditan menggunakan aplikasi Adobe Photoshop. Berikut merupakan beberapa aset yang ditampilkan pada *dashboard*.



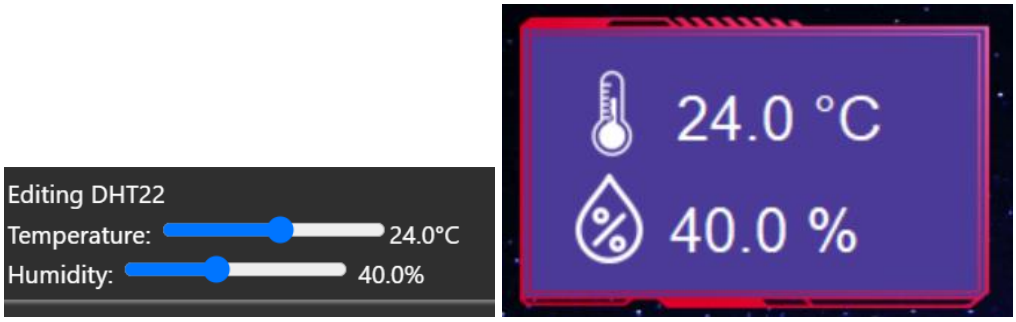
C. High Fidelity Dashboard IoT

Pada *dashboard* data yang ditampilkan adalah data yang di-*subscribe* pada saat itu juga. Hal ini dilakukan agar suhu dan kelembapan yang muncul adalah suhu dan kelembapan pada ruang server pada waktu yang bersamaan atau biasa disebut *real-time*. Selain data suhu dan kelembapan, *dashboard* menampilkan kondisi ruangan dari hasil proses Logika Fuzzy dengan input rata-rata suhu dan kelembapan.

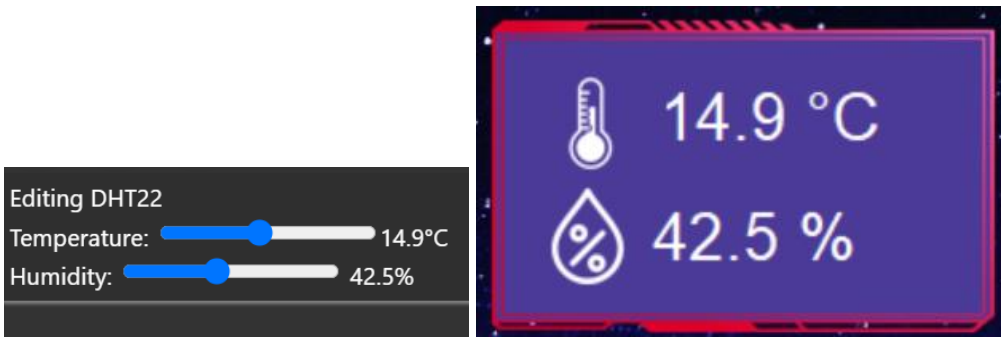
BAB V

UNIT TESTING

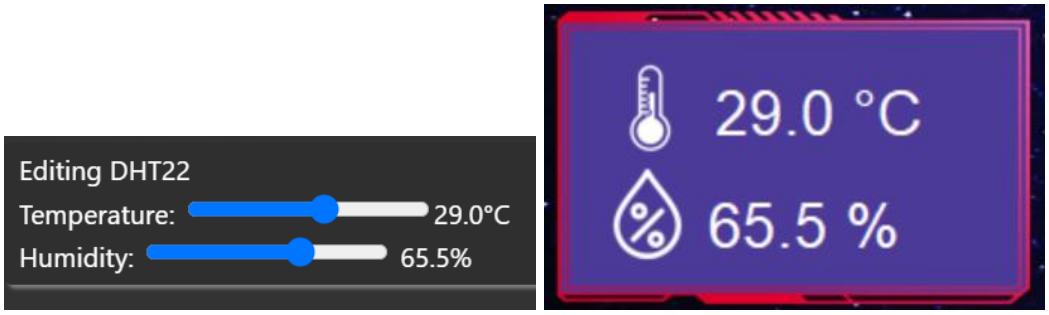
Unit testing dilakukan dengan menggunakan Blackbox Testing yang merupakan metode pengujian suatu sistem tanpa mengetahui bagaimana suatu sistem itu dibuat atau komputasi apa yang ada pada dalam sistem. Pengujian dilakukan dengan mencocokkan input sensor DHT22 pada Wokwi dengan output pada *Dashboard*. Berikut adalah hasil pencocokan sensor dengan output.



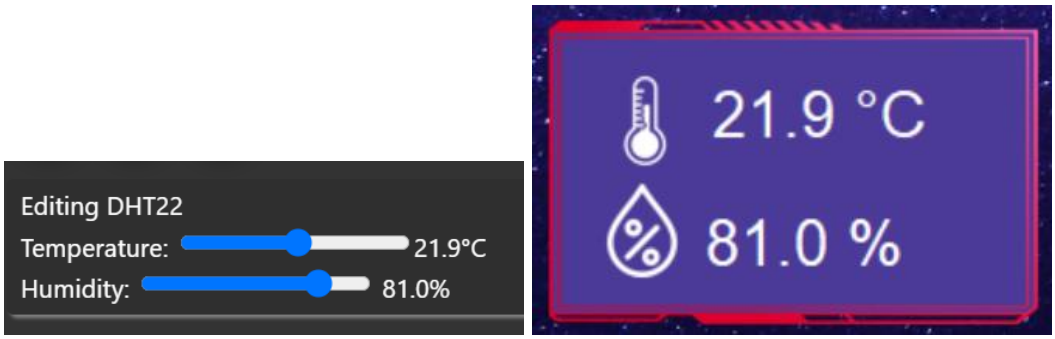
Sensor kanan atas



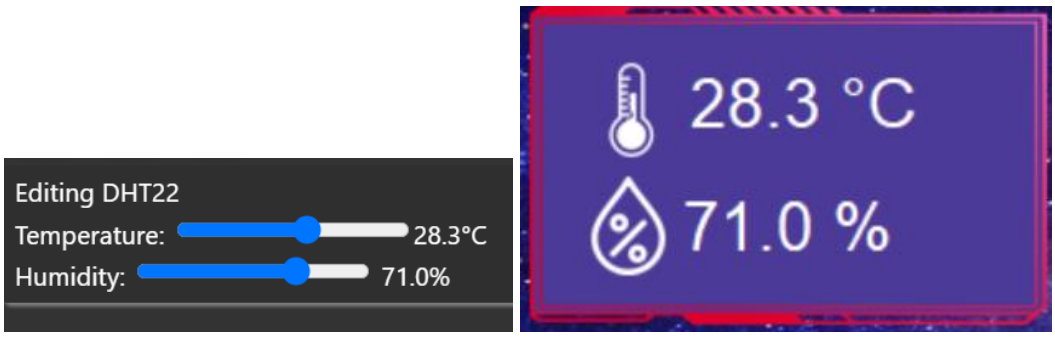
Sensor kiri atas



Sensor kanan bawah



Sensor kiri bawah



Sensor tengah

BAB VI

KESIMPULAN

Salah satu solusi yang bisa digunakan dalam melakukan pemantauan suhu dan kelembapan ruang server adalah dengan menggunakan Sistem Monitoring Ruang Server yang berbasiskan IoT. Dengan menggunakan teknologi IoT sensor-sensor pada ruang server dapat mengirimkan data ke internet melalui jaringan publik menuju jaringan privat menggunakan NGROK. Terdapat lima sensor pada WSN untuk memperluas cakupan sensor pada seluruh ruang server. Sistem dapat menyimpan data sensor ke dalam database dan menampilkannya pada *dashboard.* Penentuan kondisi ruang server dilakukan menggunakan Logika Fuzzy yang hasilnya juga ditampilkan pada *dashboard*.