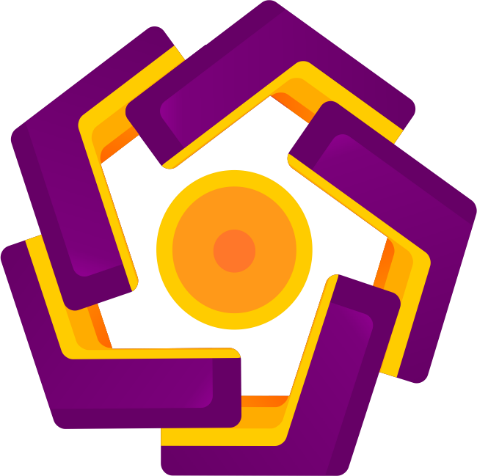
Laporan Final Project Internet of Things

“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**

1. **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*.

1. **Analisis Kebutuhan Non Fungsional**

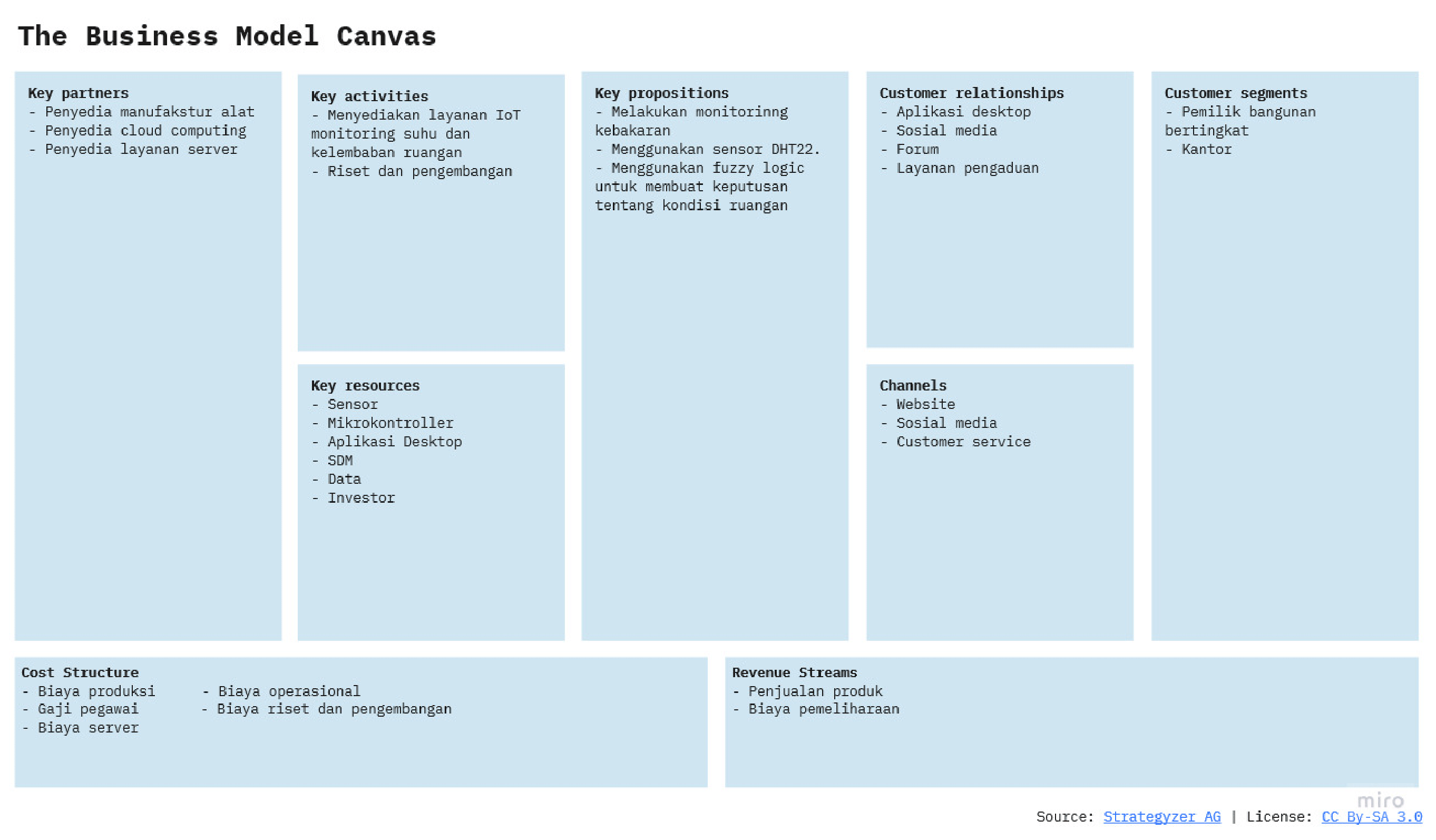
Kebutuhan non fungsional:

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.
6. **Analisis Kebutuhan Pengguna**

Kebutuhan pengguna:

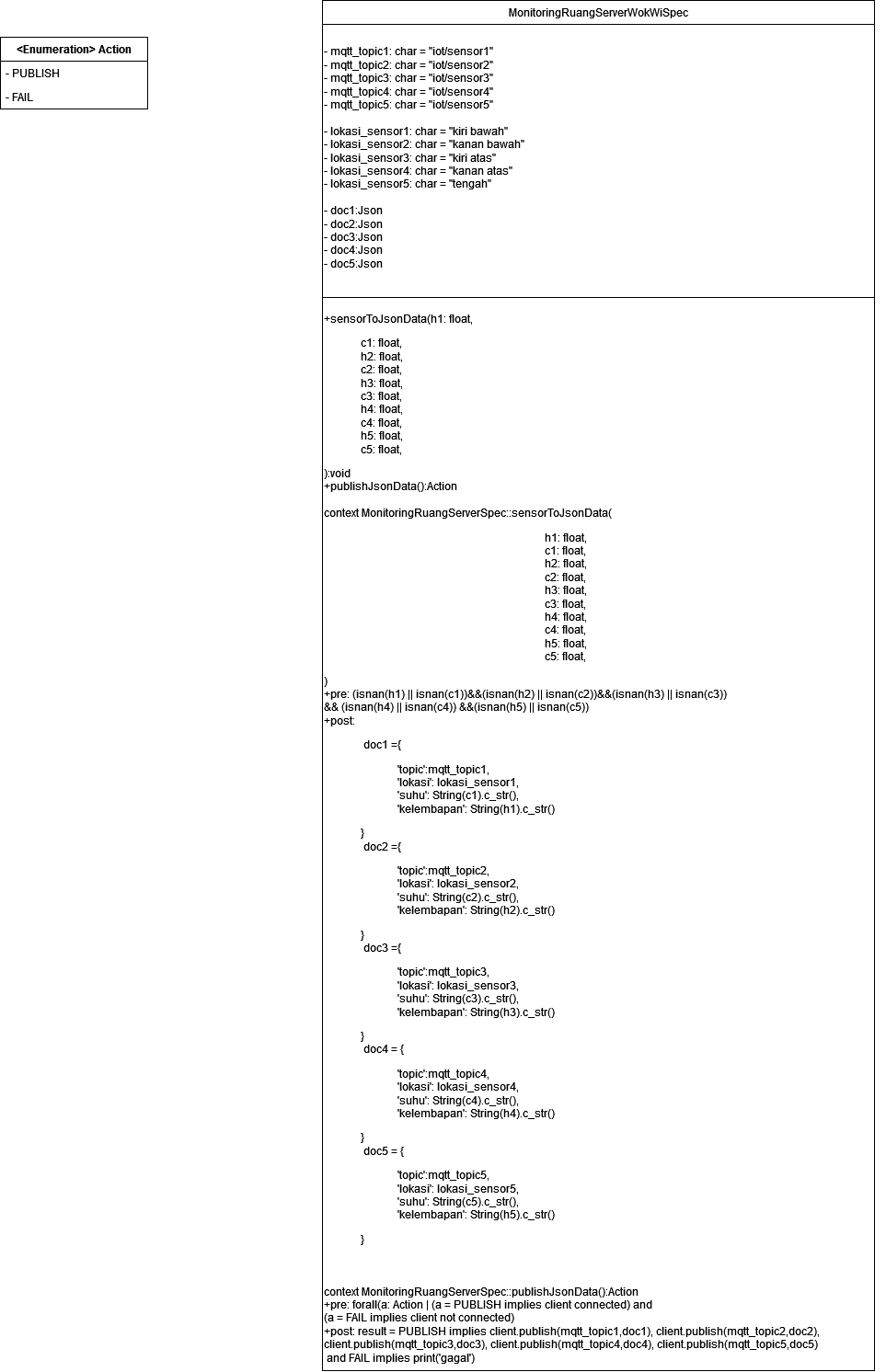
1. Memberikan informasi yang akurat tentang kondisi di dalam ruang server.
2. Pengguna harus dapat mengakses data suhu dan kelembapan secara *real-time.*
3. Pengguna dapat mengakses tampilan sistem melalui sebuah aplikasi.
4. Pengguna dapat melihat data suhu dan kelembapan yang ada dalam database.
5. Sistem harus mudah digunakan.
6. **Analisis Bisnis**

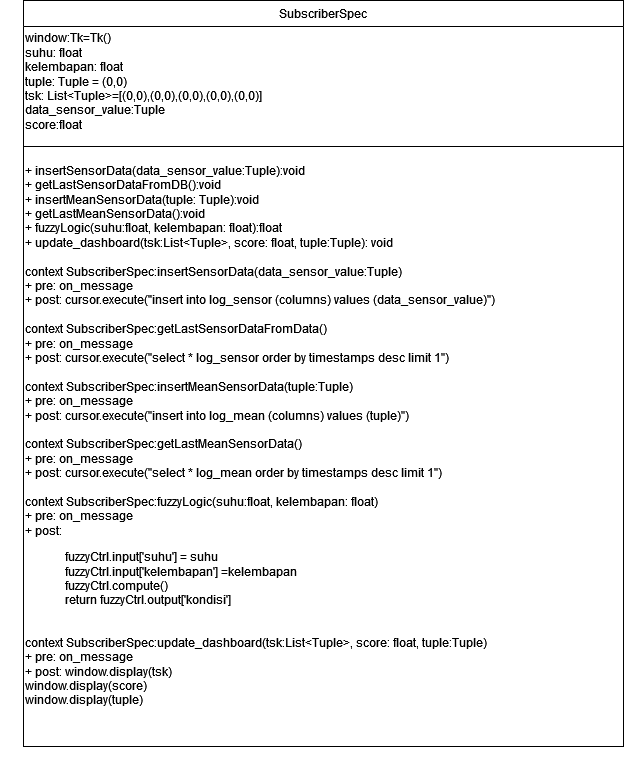
Analisis bisnis dilakukan menggunakan *Business Model canvas* yang ditampilkan sebagai berikut.

****

1. **Analisis Keamanan IoT**

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



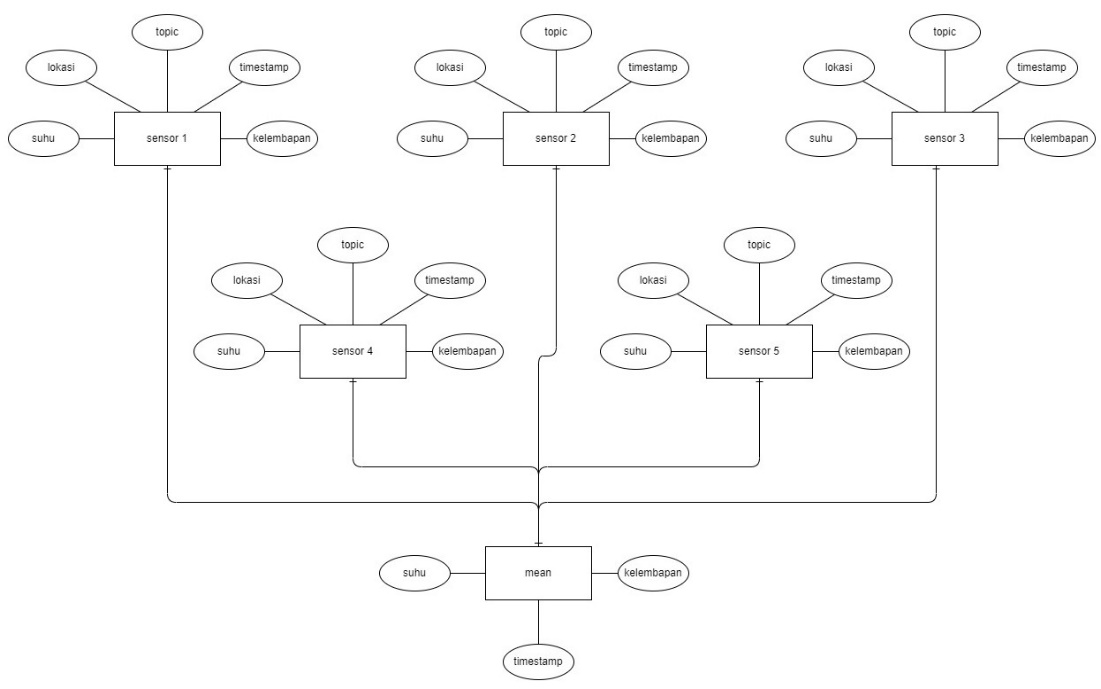


1. **Analisis Kebutuhan Service**

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*.

1. **Analisis Kebutuhan ServerDatabase**

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



**BAB III**

**RANCANGAN SERVER IOT**

1. **Desain Topologi Server**
2. **Rancangan *Node* Sensor**

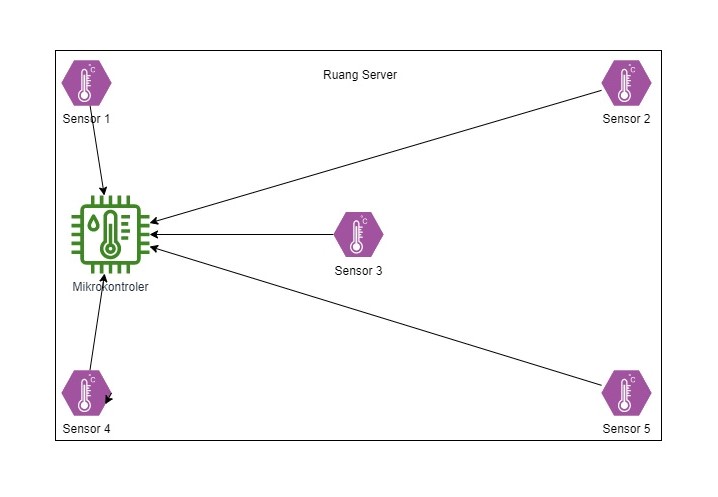
Sensor yang digunakan adalah DHT22. Sensor tersebut dapat membaca suhu dan kelembapan. Sensor dihubungkan dengan mikrokontroler ESP32 menggunakan kabel. Berikut adalah gambar sensor DHT22 yang dihubungkan ke ESP32 pada Wokwi.

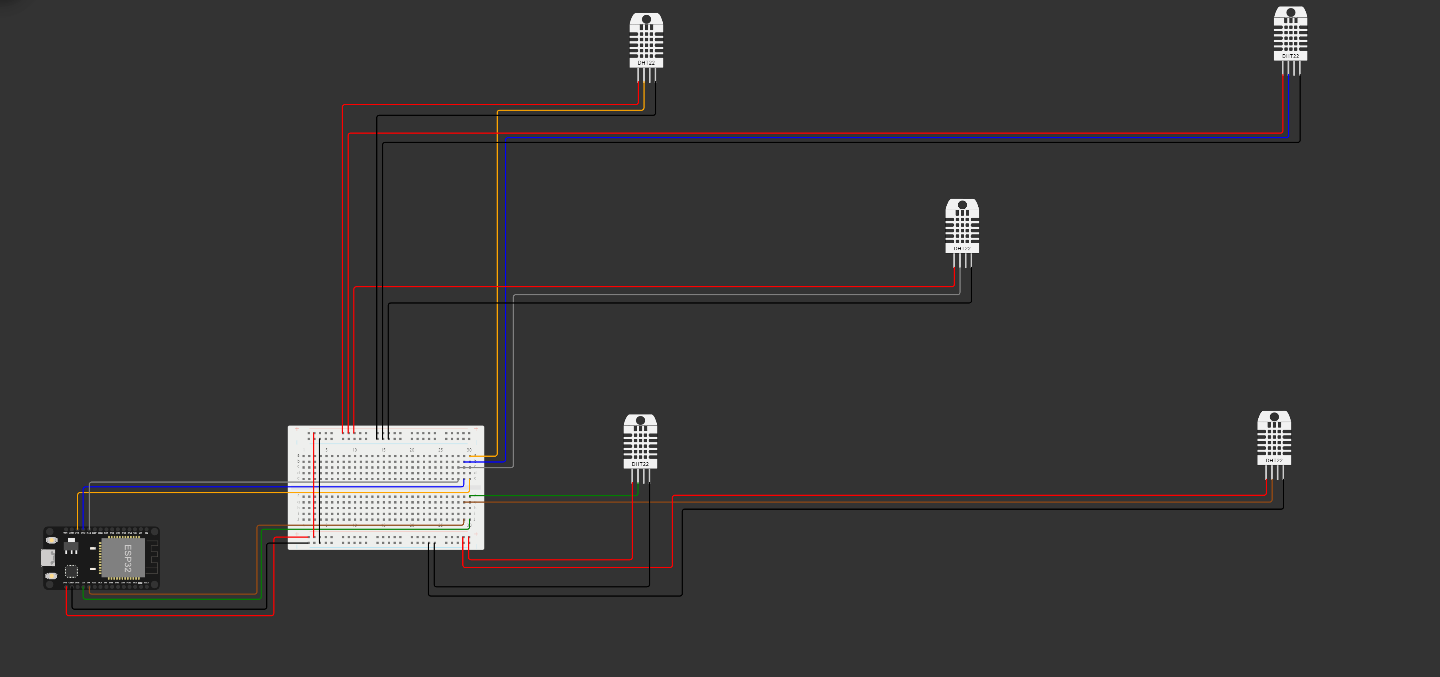
Diagram

Description automatically generated

1. **Rancangan WSN**

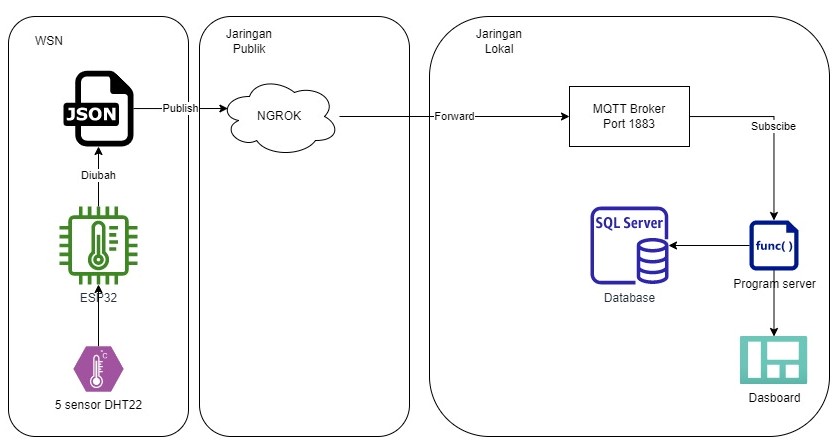
Terdapat lima sensor DHT22 untuk memantau suhu dan kelembapan ruangan server. Sensor diletakkan pada setiap sudut dan pada tengah ruangan. Lima sensor dipakai karena untuk memperbesar cakupan sensor pada seluruh ruang server. Kelima sensor terhubung dengan satu mikrokontroler ESP32 dengan topologi star.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





1. **Rancangan ServerIoT**

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.

****

1. **Kode Program Server**
2. **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()

1. **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**

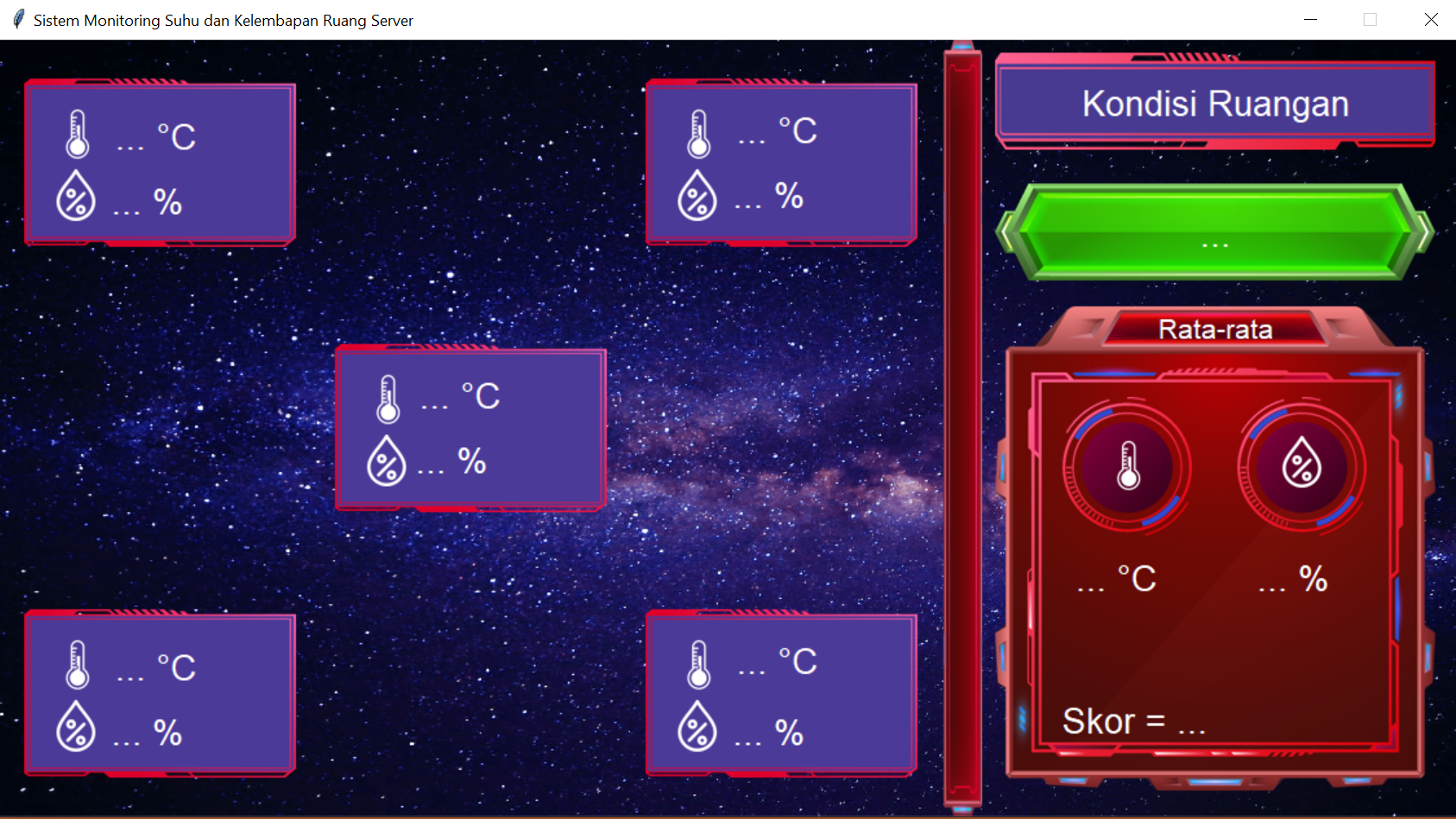
**RANCANGAN DASHBOARD IOT**

1. **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*.

Shape

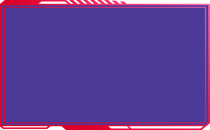
Description automatically generated with medium confidence

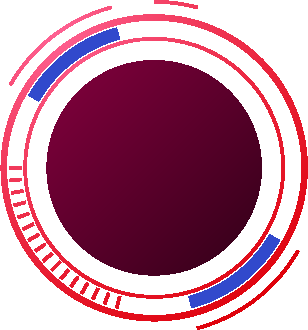


1. **Desain Asset**

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*.

1. **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.

Icon

Description automatically generated A picture containing text, outdoor, sign

Description automatically generated

*Sensor kanan atas*

*Icon

Description automatically generated A picture containing text, sign

Description automatically generated*

*Sensor kiri atas*

*A picture containing text, device

Description automatically generated A picture containing text, outdoor, sign

Description automatically generated*

*Sensor kanan bawah*

*Graphical user interface, icon

Description automatically generated A picture containing text, outdoor, sign

Description automatically generated*

*Sensor kiri bawah*

*Graphical user interface, icon

Description automatically generated A picture containing text, outdoor, sign

Description automatically generated*

*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*.