

*Proposal Tugas Proyek Perancangan Perangkat Lunak*

# **Aplikasi Pengelolaan Kualitas Udara dan Suhu di Kafe Berbasis IoT**



**Oleh:**

- |                                |                          |
|--------------------------------|--------------------------|
| <b>1. Rihan Hidayat</b>        | <b>(225150301111009)</b> |
| <b>2. Maulana Ihza Ishlahy</b> | <b>(225150300111012)</b> |

**Dosen Pengampu:**

**Achmad Arwan, S.Kom., M.Kom.**

**PROGRAM STUDI S1 TEKNIK KOMPUTER  
DEPARTEMEN TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS BRAWIJAYA  
MALANG  
2024**

## 1.1 Latar Belakang

Kafe adalah tempat yang sering digunakan untuk berkumpul, bekerja, atau bersantai. Namun, kualitas udara dan suhu yang buruk bisa mengganggu kenyamanan pelanggan dan berisiko untuk kesehatan, terutama akibat asap rokok atau suhu yang tidak sesuai. Oleh karena itu, dibutuhkan aplikasi berbasis IoT yang memonitor kualitas udara dan suhu secara real-time di kafe, serta mengontrol perangkat seperti pembersih udara dan AC secara otomatis. Pengguna, dalam hal ini pemilik kafe atau staf, bisa berinteraksi dengan sistem untuk mengatur ambang batas kualitas udara, suhu, dan mengontrol perangkat secara manual.

## 1.2 Pemodelan Kebutuhan Berorientasi Objek

### a. Identifikasi Kebutuhan

Mendefinisikan beberapa fitur dan layanan yang ingin disediakan pada aplikasi.

- **Pemantauan Kualitas Udara Ruangan:** Sistem memantau kualitas udara di dalam kafe dengan membaca data sensor CO<sub>2</sub> dan PM<sub>2.5</sub>. Ketika sensor CO<sub>2</sub> mendeteksi kadar CO<sub>2</sub> terlalu tinggi karena banyaknya orang dalam ruangan tertutup, atau sensor PM<sub>2.5</sub> mendeteksi asap dan polusi dari pengunjung yang merokok, maka sistem akan secara otomatis mengaktifkan alat pembersih dan pengolah udara berupa Air Purifier.
- **Pemantauan Suhu Ruangan:** Sistem memantau suhu udara di dalam kafe dengan membaca data sensor suhu. Ketika sensor suhu mendeteksi suhu yang terlalu panas, maka sistem akan secara otomatis mengatur AC di dalam kafe untuk menyesuaikan suhu agar lebih sejuk.
- **Notifikasi ke Barista:** Barista yang menggunakan aplikasi ini akan mendapatkan notifikasi peringatan secara otomatis apabila terjadi kondisi udara atau suhu yang tidak sesuai standar. Notifikasi juga bisa berupa informasi laporan terkait hasil monitoring dalam jangka waktu tertentu.

- **Pengaturan Suhu Manual:** Memungkinkan barista untuk mengatur tingkat derajat suhu ruangan sesuai kebutuhan melalui antarmuka aplikasi.
- **Riwayat Data Pemantauan:** Barista dapat melihat riwayat grafik historis untuk parameter kualitas udara dan suhu, sehingga dapat menganalisis kondisi lingkungan kafe pada jam-jam tertentu. Data ini disimpan pada suatu database berbasis cloud.
- **Penjadwalan Pengaturan Lingkungan:** Sistem dapat memberikan fitur bagi barista untuk melakukan pengaturan suhu atau pembersih udara berdasarkan waktu operasional kafe, termasuk mempertimbangkan jam sibuk atau non-sibuk, sehingga dapat mengaktifkan mode hemat energi.
- **Sistem Pengontrol:** Merupakan sistem yang menghubungkan seluruh sub-sistem yang ada, sehingga mereka dapat saling terhubung dan terkoordinasi. Sistem ini berada di bawah aplikasi yang juga akan mengolah data yang didapatkan.

#### **b. Identifikasi Kelas**

Mengidentifikasi beberapa kelas yang diperlukan dalam sistem. Setiap kelas mencerminkan objek atau entitas dalam sistem yang memiliki peran tertentu.

- **Kelas Sensor\_CO2:** Sebagai kelas untuk sensor yang memonitoring kadar CO2 di beberapa lokasi dalam sebuah ruangan kafe.
- **Kelas Sensor\_PM2.5:** Sebagai kelas untuk sensor yang memonitoring kadar partikel halus seperti asap di beberapa lokasi dalam sebuah ruangan kafe.
- **Kelas Sensor\_Suhu:** Sebagai kelas untuk sensor yang memonitoring suhu di dalam sebuah ruangan kafe.
- **Kelas Air\_Purifier:** Mewakili perangkat IoT berupa Air Purifier yang mengatur kualitas udara berupa Air Purifier di dalam ruangan kafe.
- **Kelas Air\_Conditioner (AC):** Mewakili perangkat IoT pendingin suhu udara berupa Air Conditioner di dalam ruangan kafe.

- **Kelas Barista (Pengguna Aplikasi):** Sebagai kelas yang mewakili barista / pengelola kafe yang menggunakan aplikasi.
- **Kelas Notifikasi:** Mewakili sistem yang mengirimkan peringatan kualitas udara, suhu, kesalahan error pada sensor atau perangkat IoT, ataupun informasi status kepada barista.
- **Kelas Database:** Sebagai kelas yang melakukan pencatatan historis pemantauan kualitas udara dan suhu dalam ruangan kafe, data akan disimpan pada database berbasis cloud.
- **Kelas Jadwal\_Pengaturan:** Sebagai kelas yang mewakili fitur penjadwalan pengaturan mode daya perangkat IoT berdasarkan waktu operasional kafe.
- **Kelas Controller:** Sebagai kelas yang akan mengontrol berbagai kelas lainnya seperti perangkat IoT, ataupun fitur-fitur yang ada seperti kelas penjadwalan dan notifikasi. Kelas ini bisa dikatakan sebagai otak yang juga akan mengolah berbagai data dan informasi yang diterima.

#### c. Identifikasi Relasi Antar Kelas

- **Kelas Sensor dengan Kelas Air\_Purifier dan Air\_Conditioner:** Data yang dibaca oleh sensor akan menentukan kapan kedua perangkat IoT diaktifkan atau diatur intensitasnya.
- **Kelas Notifikasi dengan Kelas Barista:** Barista menerima notifikasi yang sesuai seperti peringatan ataupun informasi laporan berkala.
- **Kelas Database dengan kelas Sensor\_CO2, Kelas Sensor\_PM2.5, dan Kelas Sensor\_Suhu:** Kelas Database akan menyimpan riwayat dari data yang diperoleh dari ketiga sensor ke dalam database berbasis cloud.
- **Kelas Barista dengan Kelas Jadwal\_Pengaturan dan Database:** Barista dapat melakukan pengaturan preferensi seperti jadwal operasional untuk menentukan mode daya dari perangkat IoT serta melakukan analisis berdasarkan informasi data riwayat yang didapatkan dan disimpan di database.

- **Kelas Barista dengan Kelas Air\_Purifier dan Air\_Conditioner:** Barista dapat melakukan pengaturan preferensi tingkat derajat suhu udara kepada perangkat IoT AC.
- **Kelas Controller dengan Kelas Air\_Purifier dan Air\_Conditioner:** Kelas Controller akan memberikan perintah untuk aktif atau non-aktifnya dari kelas Air\_Purifier dan Air\_Conditioner, termasuk juga dengan mode yang telah ditentukan oleh kelas Barista.
- **Kelas Controller dengan Kelas Notifikasi:** Kelas Controller akan memberikan notifikasi kepada kelas Barista apabila ada informasi yang perlu diberikan.
- **Kelas Controller dengan Kelas Barista:** Kelas Controller akan menerima semua preferensi yang diberikan oleh kelas Barista, seperti pengaturan mode perangkat IoT pengolah udara pada kelas Air\_Purifier dan Air\_Conditioner, ataupun pengaturan penjadwalan.
- **Kelas Controller dengan Kelas Sensor\_CO2, Kelas Sensor\_PM2.5, dan Kelas Sensor\_Suhu:** Kelas Controller akan menerima data yang diberikan oleh kelas Sensor untuk dilakukan pengolahan data serta memberikan perintah apabila perlu dilakukan proses kalibrasi.
- **Kelas Controller dengan Database:** Kelas Controller yang akan menyimpan data yang diterima untuk disimpan ke database. Kelas ini juga yang akan mengambil data dari database apabila diperlukan seperti memberikan informasi kepada kelas Barista.
- **Kelas Controller dengan Kelas Jadwal\_Pengaturan:** Kelas Controller akan menerima preferensi pengaturan jadwal yang telah diatur oleh barista, yang kemudian akan digunakan untuk menentukan kapan perlu memberikan perintah kepada perangkat IoT pengolah udara yang ada pada kelas Air\_Purifier dan Air\_Conditioner.

#### **d. Identifikasi Atribut dan Operasi Setiap Kelas**

- **Kelas Sensor\_CO2**
  - **Atribut**
    - **idSensorCO2** : ID unik untuk setiap sensor CO2.

- **lokasiSensorCO2** : Lokasi penempatan sensor CO2.
  - **kadarCO2** : Nilai kadar CO2 yang terdeteksi.
- **Operasi**
  - **bacaCO2()** : Membaca kadar CO2 di udara.
  - **kalibrasiCO2()** : Mengatur ulang sensor untuk akurasi pembacaan.
- **Kelas Sensor\_PM2.5**
  - **Atribut**
    - **idSensorPM2.5** : ID unik untuk setiap sensor PM2.5.
    - **lokasiSensorPM2.5**: Lokasi penempatan sensor PM2.5.
    - **kadarPM2.5** : Nilai kadar partikel udara halus yang terdeteksi.
  - **Operasi**
    - **bacaPM2.5()** : Membaca kadar CO2 di udara.
    - **kalibrasiPM2.5()** : Mengatur ulang sensor PM2.5 untuk akurasi pembacaan.
- **Kelas Sensor\_Suhu**
  - **Atribut**
    - **idSensorSuhu** : ID unik untuk setiap sensor suhu.
    - **lokasiSensorSuhu**: Lokasi penempatan sensor suhu.
    - **suhuRuangan** : Nilai suhu ruangan yang terdeteksi.
  - **Operasi**
    - **bacaSuhu()** : Membaca suhu udara di ruangan kafe.
    - **kalibrasiSuhu()** : Mengatur ulang sensor suhu untuk akurasi pembacaan.
- **Kelas Air\_Purifier**
  - **Atribut**
    - **idAP** : ID unik untuk setiap Air Purifier.
    - **statusAP** : Status Air Purifier (aktif/nonaktif).
    - **modeAP** : Mode Air Purifier (normal/hemat daya).
  - **Operasi**
    - **aktifkanAP()** : Mengaktifkan Air Purifier.

- **nonaktifkanAP()** : Mematikan Air Purifier.
  - **aturModeAP()** : Mengatur Mode Air Purifier berdasarkan jam operasional.
- **Kelas Air\_Conditioner**
  - **Atribut**
    - **idAC** : ID unik untuk setiap AC.
    - **statusAC** : Status AC (aktif/nonaktif).
    - **prefSuhu** : Suhu preferensi diatur pengguna untuk AC
    - **modeAC** : Mode AC (normal/hemat daya).
  - **Operasi**
    - **aturSuhu()** : Mengatur suhu yang ingin dicapai AC.
    - **aktifkanAC()** : Mengaktifkan AC.
    - **nonaktifkanAC()** : Mematikan AC.
    - **aturModeAC()** : Mengatur Mode AC berdasarkan jam operasional.
- **Kelas Barista**
  - **Atribut**
    - **idBarista** : ID unik barista aplikasi.
    - **nama** : Nama barista aplikasi.
    - **email** : Alamat email barista untuk notifikasi serta identitas login ke aplikasi.
  - **Operasi**
    - **aturSuhuRuangan()** : Mengatur preferensi derajat suhu ruangan.
    - **aturJamOperasional()** : Mengatur jam operasional sibuk atau non-sibuk sebagai patokan perangkat IoT untuk mengubah mode dayanya.
    - **terimaNotifikasi()** : Menerima notifikasi peringatan atau informasi dari sistem.
    - **lihatRiwayat()** : Mengakses riwayat pemantauan ruangan yang diambil dari database cloud aplikasi.

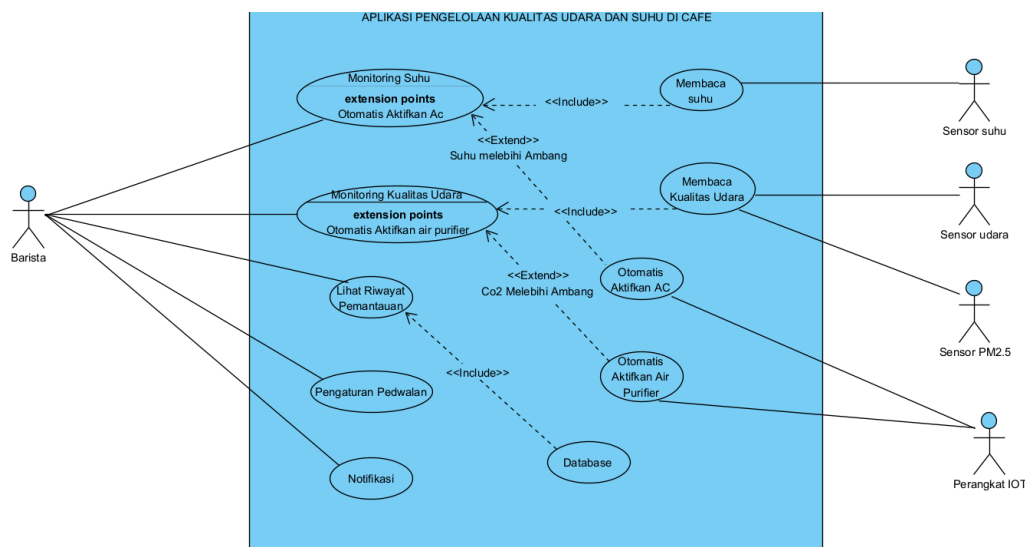
- **Kelas Notifikasi**
  - **Atribut**
    - **idNotifikasi** : ID unik untuk setiap notifikasi.
    - **pesan** : Isi notifikasi yang dikirimkan.
    - **waktuKirim** : Waktu notifikasi dikirim.
  - **Operasi**
    - **kirimNotifikasi()** : Mengirimkan notifikasi kepada barista.
    - **ambilData()** : Mengambil data dari database untuk diolah menjadi notifikasi.
- **Kelas Database**
  - **Atribut**
    - **idRiwayat** : ID unik untuk setiap entri riwayat.
    - **waktuPencatatan**: Waktu pencatatan data yang diterima.
    - **kadarCO2** : Nilai kadar CO2 yang dicatat.
    - **kadarPM2.5** : Nilai kadar PM2.5 yang dicatat.
    - **SuhuRuangan** : Nilai suhu yang dicatat.
    - **dataIDSensor** : Menyimpan ID setiap sensor yang terhubung dengan sistem.
    - **dataIDPerangkat**: Menyimpan ID setiap perangkat IoT pengolah udara yang terhubung ke sistem.
    - **dataStatusAP** : Data status dari Air Purifier ketika data dicatat, seperti status mode aktif/nonaktif serta mode daya.
    - **dataStatusAC** : Data status dari Air Conditioner ketika data dicatat, seperti status mode aktif/nonaktif, preferensi suhu saat itu, serta mode daya.
  - **Operasi**
    - **simpanRiwayat()** : Menyimpan riwayat data pemantauan ke dalam database cloud aplikasi
    - **lihatGrafik()** : Menampilkan grafik data historis kepada barista pada antarmuka aplikasi.
- **Kelas Jadwal\_Pengaturan**
  - **Atribut**

- **idJadwal** : ID unik untuk setiap pembuatan penentuan jadwal baru.
  - **waktuMulai** : Pengaturan waktu mulai suatu jadwal.
  - **waktuSelesai** : Pengaturan waktu selesai suatu jadwal.
  - **mode** : Mode yang diatur untuk setiap jadwal (normal/hemat daya).
- **Operasi**
  - **aturJadwal()** : Menentukan jadwal untuk pengaturan mode pada perangkat IoT.
  - **aktifkanJadwal()** : Mengaktifkan jadwal berdasarkan waktu yang ditentukan.
- **Kelas Controller**
  - **Atribut**
    - **idController** : ID unik untuk controller sistem yang dimiliki oleh Barista.
  - **Operasi**
    - **aktifkanPerangkat()**: Mengaktifkan / menonaktifkan perangkat IoT pengolah udara seperti Air Purifier dan Air Conditioner.
    - **aturPrefSuhuAC()** : Mengatur perangkat IoT Air Conditioner untuk preferensi suhu ruangan dari Barista.
    - **gantiModePerangkat()** : Mengubah mode perangkat IoT pengolah udara sesuai yang ditentukan pada jadwal oleh Barista.
    - **cekStatusPerangkat()** : Mengecek status perangkat IoT pengolah udara.
    - **simpanStatusPerangkat()**: Menyimpan status perangkat IoT pengolah udara.
    - **kirimNotifikasi()** : Mengirimkan notifikasi ke Barista.
    - **prosesDataSensor()** : Memproses data yang diterima dari sensor.

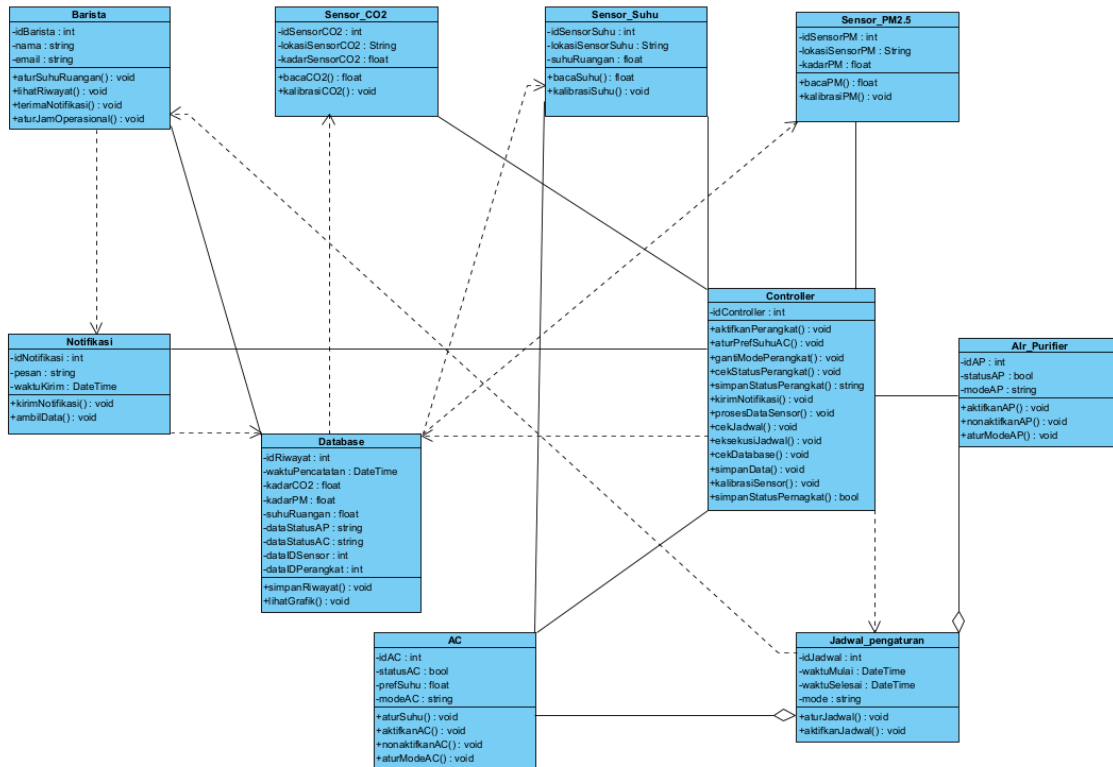
- **cekJadwal()** : Membaca data jadwal yang dibuat oleh Barista untuk mengatur perangkat IoT pengolah udara.
- **eksekusiJadwal()** : Menjalankan perubahan status sesuai jadwal.
- **cekDatabase()** : Mengambil data dari database apabila diperlukan.
- **simpanData()** : Menyimpan data yang diterima serta hasil dari proses data ke database.
- **kalibrasiSensor()** : Memberikan perintah ke sensor untuk melakukan kalibrasi.

**e. Diagram UML (Unified Modeling Language)**

### i. Use-Case Diagram

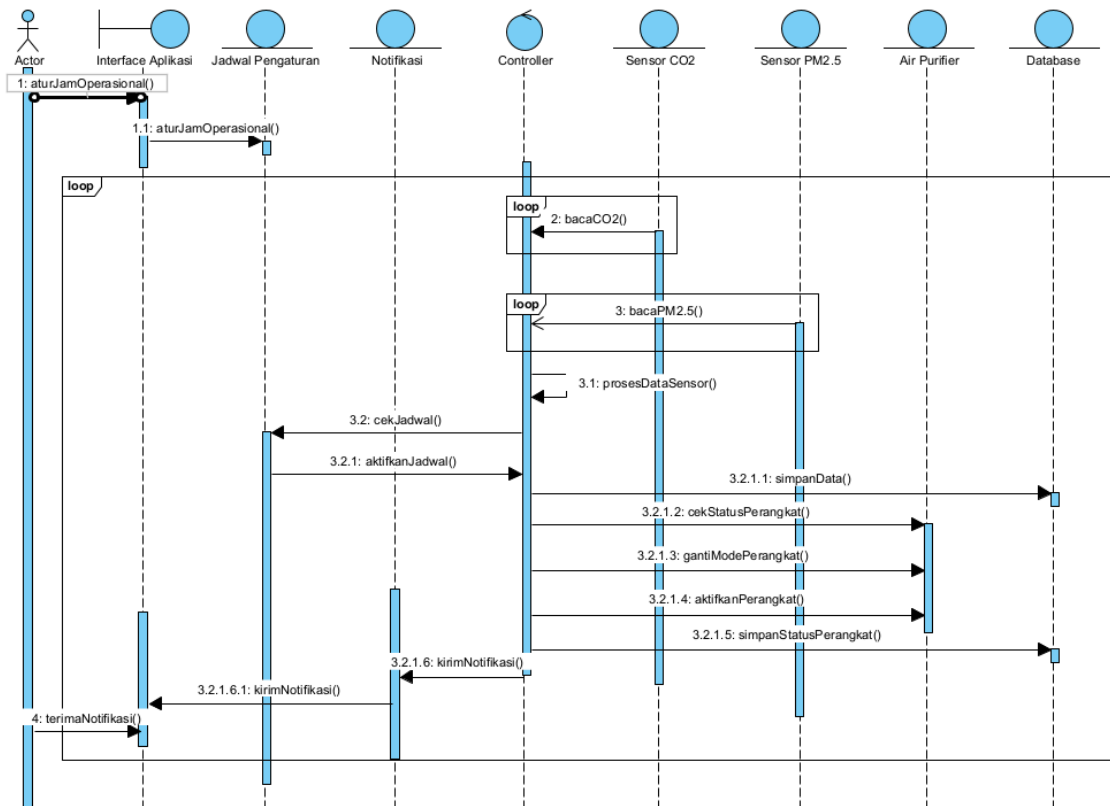


## ii. Class Relationship Diagram

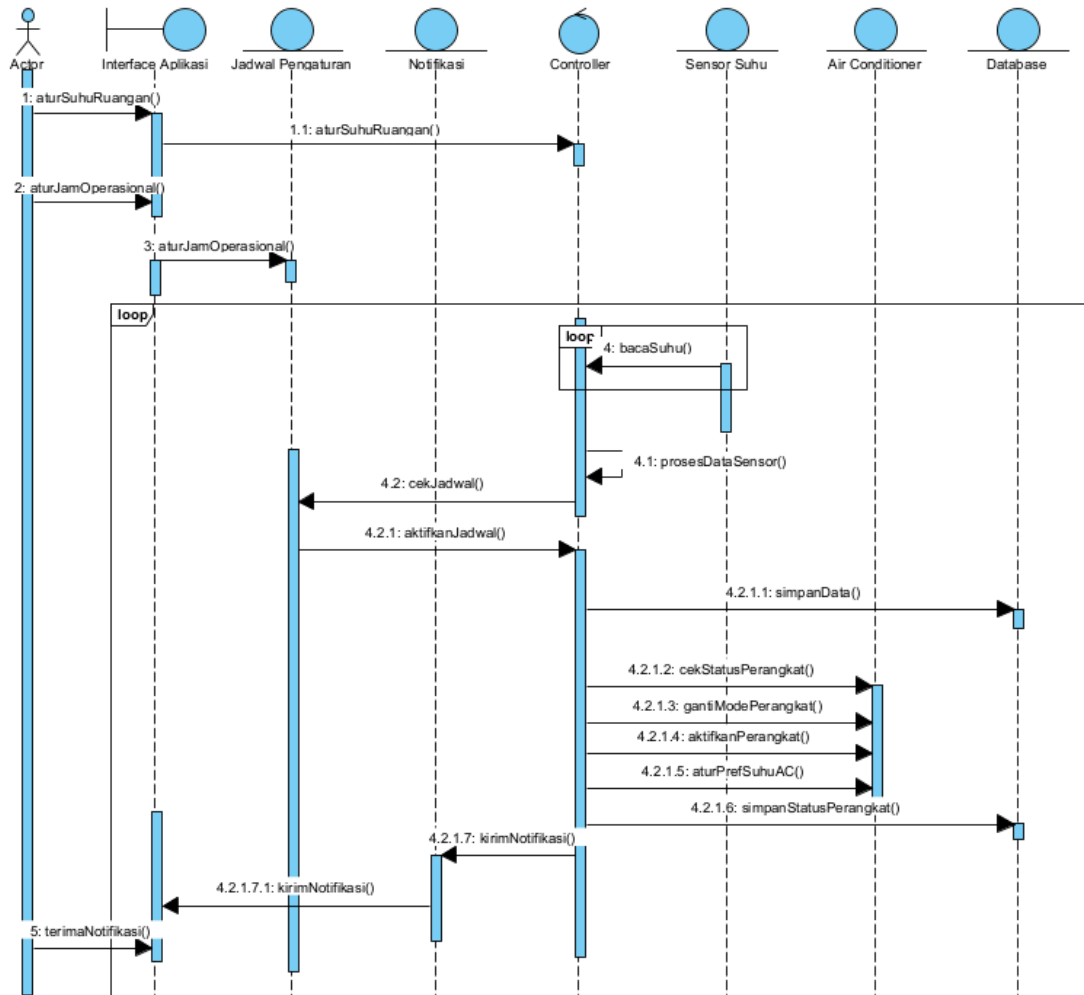


### iii. Sequence Diagram

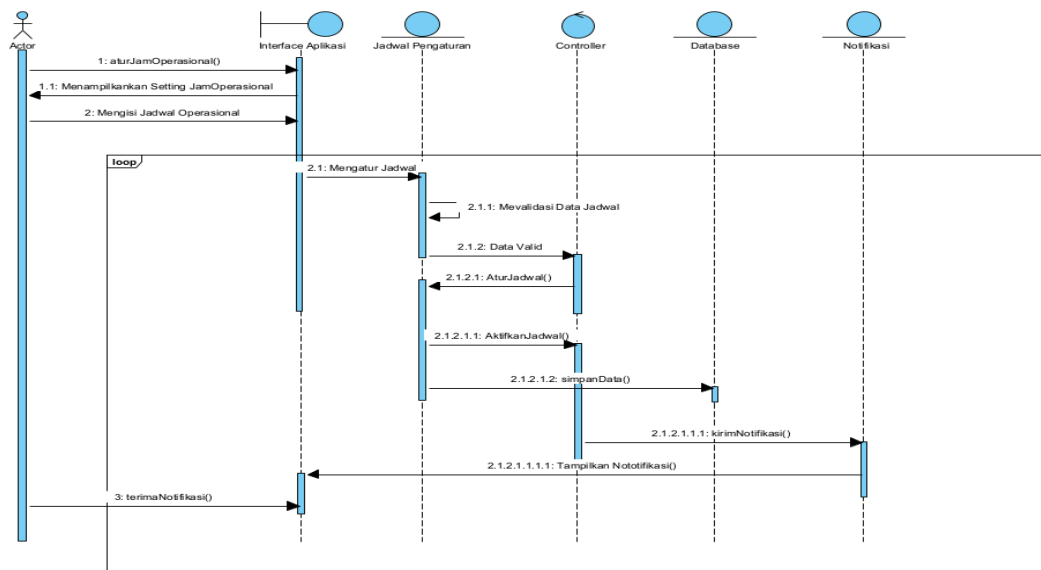
### a. Monitoring Kualitas Udara



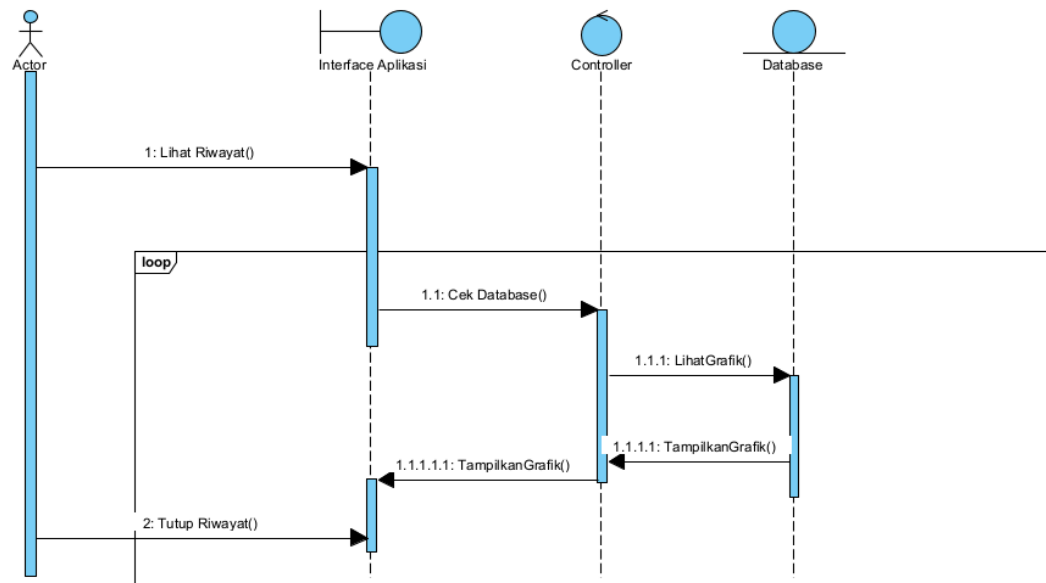
## b. Monitoring Suhu Udara



## c. Pengaturan Penjadwalan



#### d. Lihat Riwayat Pemantauan



#### f. Use-Case Scenario

<b>Objective</b>	Pemantauan Kualitas Udara
<b>Actors</b>	Sistem (SensorCo2, SensorPM2.5, Controller, Database, Air Purifier, Jadwal_Pengaturan Notifikasi), Barista.
<b>Pre-Condition</b>	<ul style="list-style-type: none"> <li>• Sensor CO2 dan PM2.5 aktif dan terhubung dengan sistem.</li> <li>• Barista telah membuat Jadwal_Pengaturan</li> </ul>

	untuk perangkat.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. SensorCO2 dan SensorPM2.5 membaca kualitas udara secara berkala.</li> <li>2. Controller menerima data hasil pemantauan kualitas udara dari kedua sensor.</li> <li>3. Controller melakukan cek apakah merupakan saat untuk melakukan pengaktifan perangkat berdasarkan dari Jadwal_Pengaturan apabila ternyata kualitas udara buruk.</li> <li>4. Controller memvalidasi data: <ol style="list-style-type: none"> <li>a. Apabila data melebihi ambang batas normal, Controller mengaktifkan Air Purifier serta memberi tahu barista melalui notifikasi. Status pengaktifan Air Purifier juga akan disimpan ke Database.</li> <li>b. Apabila kualitas udara baik, data hanya disimpan ke Database.</li> </ol> </li> <li>5. Barista dapat memantau informasi atau melakukan tindakan manual jika diperlukan.</li> </ol>
<b>Alternative Flow</b>	Sensor gagal membaca data : Sistem menampilkan pesan kesalahan dan mengirimkan notifikasi ke barista.
<b>Post-Condition</b>	Data kualitas udara berhasil dicatat ke Database dan (apabila diperlukan) memberikan notifikasi kepada Barista.

<b>Objective</b>	Pemantauan Suhu Ruangan
<b>Actors</b>	Sistem (SensorSuhu, Controller, Database, Air Conditioner, Jadwal_Pengaturan, Notifikasi), Barista.
<b>Pre-Condition</b>	<ul style="list-style-type: none"> <li>• Sensor suhu aktif dan terhubung dengan sistem.</li> <li>• Barista telah mengatur preferensi suhu.</li> <li>• Barista telah membuat Jadwal_Pengaturan untuk perangkat.</li> </ul>
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. SensorSuhu membaca suhu udara secara berkala.</li> <li>2. Controller menerima data hasil pemantauan</li> </ol>

	<p>suhu udara dari SensorSuhu.</p> <ol style="list-style-type: none"> <li>3. Controller melakukan cek apakah merupakan saat untuk melakukan pengaktifan perangkat berdasarkan dari Jadwal_Pengaturan apabila ternyata suhu tidak sesuai preferensi Barista.</li> <li>4. Controller memvalidasi data: <ol style="list-style-type: none"> <li>a. Apabila data melebihi ambang batas normal, Controller mengaktifkan Air Conditioner serta memberi tahu barista melalui notifikasi. Status pengaktifan Air Conditioner juga akan disimpan ke Database.</li> <li>b. Apabila suhu udara sesuai dengan preferensi, data hanya disimpan ke Database.</li> </ol> </li> <li>5. Barista dapat memantau informasi atau melakukan tindakan manual jika diperlukan.</li> </ol>
<b>Alternative Flow</b>	Sensor gagal membaca data : Sistem menampilkan pesan kesalahan dan mengirimkan notifikasi ke barista.
<b>Post-Condition</b>	Data suhu berhasil dicatat dan (apabila diperlukan) memberikan notifikasi kepada barista.

<b>Objective</b>	Notifikasi ke Barista
<b>Actors</b>	Sistem (Notifikasi, SensorCO2, SensorPM2.5, SensorSuhu, Air Purifier, Air Conditioner Controller), Barista
<b>Pre-Condition</b>	<ul style="list-style-type: none"> <li>• Ada kondisi udara atau suhu yang tidak sesuai standar.</li> <li>• Terjadi kesalahan pada sensor ataupun perangkat pengolah udara.</li> <li>• Sudah saatnya jadwal waktu untuk memberikan informasi status kondisi kepada Barista.</li> </ul>
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Controller mendeteksi anomali data kualitas udara maupun suhu dari sensor, atau terjadi kesalahan pada perangkat, ataupun telah waktunya untuk memberikan informasi</li> </ol>

	<p>status kepada Barista.</p> <ol style="list-style-type: none"> <li>2. Controller menyusun pesan notifikasi yang relevan.</li> <li>3. Controller mengirimkan notifikasi ke antarmuka aplikasi yang digunakan Barista.</li> </ol>
<b>Alternative Flow</b>	Barista tidak dapat menerima notifikasi: Sistem menyimpan pesan dan akan mencoba mengirim ulang saat jaringan tersedia.
<b>Post-Condition</b>	Barista menerima notifikasi peringatan kesalahan sistem ataupun informasi status dari kondisi ruangan dan sistem. Dengan begitu Barista dapat mengambil tindakan jika diperlukan.

<b>Objective</b>	Penyimpanan dan Analisis Riwayat
<b>Actors</b>	Sistem (Database, Controller, AntarmukaAplikasi), Barista.
<b>Pre-Condition</b>	<p>Data dari sensor yang telah tercatat tersimpan di database.</p> <p>Data dari status perangkat yang tersimpan di database.</p>
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Barista memilih opsi untuk melihat riwayat pemantauan pada antarmuka aplikasi.</li> <li>2. Sistem mengambil data historis dari Database.</li> <li>3. Sistem menampilkan grafik atau daftar data historis pada tampilan antarmuka aplikasi kepada Barista.</li> </ol>
<b>Alternative Flow</b>	Tidak ada data historis: Sistem menampilkan pesan bahwa data belum tersedia.
<b>Post-Condition</b>	Data riwayat sistem berhasil ditampilkan kepada Barista.

<b>Objective</b>	Penjadwalan Pengaturan Lingkungan
<b>Actors</b>	Sistem (Jadwal Pengaturan, Antarmuka

	Aplikasi, Controller, Database), Barista.
<b>Pre-Condition</b>	Data hasil pemantauan sensor diterima oleh Controller, namun sistem dibuat untuk mengikuti Jadwal Pengaturan agar dapat menyesuaikan kebutuhan.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Barista telah mengatur jadwal pengaturan lingkungan melalui antarmuka aplikasi (menentukan jadwal jam sibuk dan non-sibuk untuk menentukan mode perangkat).</li> <li>2. Jadwal operasional kafe telah dimasukkan oleh Barista.</li> <li>3. Sistem memvalidasi data jadwal, kemudian menyimpan jadwal yang ditentukan.</li> <li>4. Pada waktu dari jadwal yang telah di atur, sistem akan menentukan mode yang sesuai kepada perangkat IoT pengolah udara (Air Purifier dan Air Conditioner) sesuai dengan data dari Jadwal Pengaturan.</li> </ol>
<b>Alternative Flow</b>	Jadwal tidak dapat disimpan: Sistem menampilkan pesan kesalahan kepada Barista.
<b>Post-Condition</b>	Jadwal pengaturan lingkungan berhasil diterapkan kepada perangkat.

#### g. Sensor Use case dan Class

##### 1. Sensor\_CO2

use case : Pemantauan kadar CO2 dalam ruangan kafe.

Example : Mengurangi kadar CO2 yang meningkat akibat banyaknya pengunjung di ruang tertutup.

```

-----
|   Sensor_CO2   |
-----
| - idSensorCO2   |
| - lokasiSensorCO2 |
| - kadarSensorCO2 |

```

```

-----
| +bacaCO2()          |
| +kalibrasiCO2()     |
-----

```

## 2. Sensor\_PM2.5

Use Case : Deteksi polusi udara akibat asap rokok atau debu.

Example : Mengaktifkan Air Purifier ketika sensor mendeteksi PM2.5 tinggi akibat asap rokok.

```

-----
|   Sensor_PM2.5     |
-----
| - idSensorPM        |
| - lokasiSensorPM    |
| - kadarSensorPM     |
-----
| +bacaPM()           |
| +kalibrasiPM()      |
-----

```

## 3. Sensor\_Suhu

Use Case : Pemantauan suhu ruangan untuk kenyamanan pengunjung.

Example : Menurunkan suhu ruangan dengan mengatur AC ketika suhu melebihi ambang batas.

```

-----
|   Sensor_Suhu      |
-----
| - idSensorSuhu      |
| - lokasiSensorSuhu  |
| - kadarSensorSuhu   |

```

```

-----
| +bacaSuhu()          |
| +kalibrasiSuhu()     |
-----

```

#### H. Table Class Completeness Test

Requirement	Class/Method
Pemantauan kualitas udara ruangan	bacaCO2(), bacaPM2.5()
Pemantauan suhu ruangan	bacaSuhu()
Pemberian notifikasi jika parameter buruk	kirimNotifikasi()
Penyimpanan data historis	simpanRiwayat()
Tampilan grafik data historis	lihatGrafik()
Penjadwalan pengaturan perangkat	aturJadwal(), aktifkanJadwal()
Pengaturan manual perangkat IoT	aturSuhu(), aturMode()