# PENGEMBANGAN DAN IMPLEMENTASI *BACKEND* & REST-API

## *DASHBOARD* *IOT* PADA *INTELLIGENCE CONTROLLING* *SYSTEM* (*ICS*) *SMART* *GREENHOUSE* MENGGUNAKAN PROTOKOL MQTT

**(Proposal)**

### Oleh

**M. Affan Siddiqie**

### Asmara 2015031048



# JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK UNIVERSITAS LAMPUNG

**2023**

# PENGEMBANGAN DAN IMPLEMENTASI *BACKEND* & REST-API

## *DASHBOARD* *IOT* PADA *INTELLIGENCE* *CONTROLLING* *SYSTEM* (*ICS*) *SMART* *GREENHOUSE* MENGGUNAKAN PROTOKOL MQTT

### Oleh

**M. Affan Siddiqie Asmara**

**Proposal Penelitian**

**Sebagai Salah Satu Syarat untuk Mencapai Gelar**

# SARJANA TEKNIK

**Pada**

## Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung



# FAKULTAS TEKNIK UNIVERSITAS LAMPUNG BANDAR LAMPUNG

**2023**

# DAFTAR ISI

**Halaman**

|  |  |
| --- | --- |
| **DAFTAR ISI** |  |
| **DAFTAR GAMBAR** |  |
| **DAFTAR TABEL** | **i** |
| **BAB I PENDAHULUAN** |  |
| 1.1. Latar Belakang |  |
| 1.2. Rumusan Masalah |  |
| 1.3 Batasan Masalah |  |
| 1.4 Tujuan Penelitian |  |
| 1.5 Manfaat Penelitian |  |
| 1.6 Sistematika Penulisan |  |
| **BAB II TINJAUAN PUSTAKA** |  |
| 2.1 Penelitian Terkait |  |
| *2.2* *Smart GreenHouse* |  |
| *2.3* *Intelligence Controlling System (ICS)* |  |
| *2.4* *Internet of Things (IoT)* |  |
| *2.5* *Message Queuing Telemetry Transport (MQTT)* |  |
| 2.6 *Mosquitto* |  |
| 2.7 *Virtual Private Server (VPS)* |  |
| *2.8* *Backend Development* |  |
| 2.9 *Node.Js* |  |
| 2.10 *Express*.*js* |  |
| *2.11* *PostgreeSQL* |  |
| *2.12* *Application Programming Interface (API)* |  |
| *2.13* *REST-API*  *2.14* *GitHub* |  |
| *2.15* *VSCode* |  |
| *2.16* *Postman* |  |
| **BAB III METODOLOGI PENELITIAN** |  |
| 3.1 Waktu dan Tempat Penelitian |  |
| *3.2* *Capstone Project* |  |

3.3 Alat dan Bahan

3.3.1 Alat

3.3.2 Bahan

3.4 Tahapan Penelitian

3.5 Analisa Kebutuhan

3.6 Pengembangan Sistem

3.7 Tahap *Task To Do*

*3.8* Tahap *Work in Progress*

3.9 Tahap *Testing*

3.10 Tahap *Done*

3.11 Analisa Hasil

DAFTAR PUSTAKA **DAFTAR TABEL**

**DAFTAR GAMBAR**

# BAB I PENDAHULUAN

## 1.1. Latar belakang

Indonesia, sebagai negara agraris, mengandalkan pertanian sebagai tulang punggung kehidupan, terutama di kalangan masyarakat kecil. Saat ini, kegiatan pertanian merajalela di berbagai komunitas di Indonesia, terutama di lingkungan masyarakat kecil. Namun, masyarakat kecil yang berada di daerah terpencil masih mengalami kendala akibat minimnya pemanfaatan dan pengembangan teknologi yang dapat mendukung pengelolaan lahan pertanian dan hasilnya. Sebagian besar petani masih sangat tergantung pada kondisi cuaca alam, sehingga hasil pertanian tidak selalu memuaskan ketika cuaca tidak sesuai harapan. Aktivitas manusia untuk mempertahankan hidupnya juga berdampak signifikan pada peningkatan kebutuhan lahan. Akibatnya, lahan pertanian menjadi unsur krusial dalam mendukung kehidupan manusia. Pemanfaatan lahan ini semakin meningkat karena banyak orang menggunakan lahan untuk tempat tinggal, usaha, pembangunan akses umum, dan fasilitas lainnya, yang pada akhirnya mengakibatkan keterbatasan luas lahan. Penggunaan lahan yang tidak terkendali ini dapat mengakibatkan gangguan pada keseimbangan ekosistem. Oleh karena itu, penelitian ini bertujuan untuk merancang sistem Smart Greenhouse dengan *Intelligence Controlling System* (ICS) yang dapat memantau perkembangan tanaman dengan mengatur suhu, kelembapan, dan intensitas cahaya berbasis Internet of Things (IoT).

Agar pemantauan data pada smart greenhouse dapat dilakukan dengan baik maka diperlukan sebuah dashboard untuk memonitoring data-data yang telah diterima dari sensor yang ada pada smart greenhouse. Dashboard adalah alat yang menyediakan antarmuka visual, yang menggabungkan dan menyajikan informasi penting untuk mencapai tujuan tertentu secara sekilas. Tampilan visual dashboard yang mampu mengkomunikasikan informasi dengan jelas, cepat, dan memberikan persepsi benar-benar menjadi kunci keberhasilan dashboard. Konsep visualisasi data dan informasi akan digunakan saat merancang antarmuka dashboard.

Visualisasi data dan informasi terkait hal-hal mengenai persepsi visual dan media penyajian data, penyampaian komponen dashboard harus mengutamakan efektifitas penyampaian informasi untuk memudahkan pengguna melihat, memantau dan membantu dalam mengambil keputusan yang tepat. dalam waktu nyata.

Dalam pengembangan dashboard Dalam pengembangan dashboard, keberadaan backend sangat penting karena berperan sebagai fondasi yang mendukung pengolahan dan penyimpanan data secara efisien. Backend merupakan bagian dari sistem yang bertanggung jawab untuk mengelola basis data, menyimpan informasi dari sensor, dan mengolah data mentah menjadi informasi yang dapat ditampilkan dengan jelas pada dashboard.

Implementasi backband pada dashboard ini menggunakan Penelitian ini mengadopsi protokol MQTT (Message Queuing Telemetry Transport) sebagai salah satu komponen utama dalam sistem backend. Protokol MQTT dipilih karena kemampuannya untuk menyampaikan data secara ringan dan efisien melalui jaringan, yang sangat relevan untuk aplikasi Internet of Things (IoT) seperti Smart Greenhouse. MQTT memungkinkan perangkat di dalam greenhouse untuk mengirimkan data ke backend secara real-time, memastikan bahwa informasi yang disajikan pada dashboard selalu terkini. Virtual Private Server (VPS) digunakan sebagai lingkungan hosting untuk backend, dan Node.js, sebagai platform pengembangan server-side yang berbasis JavaScript, menjadi pilihan dalam mengimplementasikan backend pada penelitian ini.

## 1.2. Rumusan Masalah

Rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana data yang dikirimkan oleh sensor-sensor yang digunakan pada ICS dapat dikirimkan menggunakan protokol MQTT (Message Queuing Telemetry Transport)
2. Bagaimana menjalankan program backend pada Virtual Private Server (VPS)
3. Bagaimana data yang telah diterima dari sensor dapat disimpan dan diolah kedalam database postgreeSQL
4. Bagaimana cara data yang telah disimpan dapat diambil dan diolah sebagai bahan penelitian lain

## 1.3. Batasan Masalah

Adapun batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penelitian berfokus pada mengimplementasikan protokol komunikasi *Message Queuing Telemetry Transport (MQTT)* pada *Virtual Private Server (VPS)* yang terhubung dengan ICS Smart Greenhouse dan mengirimkan data ke *database* PostgreeSQL, dan perancangan API yang akan digunakan oleh frontend dan mobile apps.
2. Penelitian ini tidak mencakup proses pembuatan *dashboard* dan alat sensor Intelligence Controlling System pada Smart Greenhouse.

## 1.4. Tujuan Penelitian

Adapun tujuan penelitian dalam penelitian ini adalah sebagai berikut:

1. Mengimplementasikan sebuah protokol komunikasi *Message Queue Telemetry Transport (MQTT)* dalam proses pengiriman dan menerima data sensor*.*
2. Menyimpan data yang diterima oleh sensor pada ICS Smart Greenhouse kedalam *database PostgreeSQL*.
3. Membangun API yang dapat digunakan untuk mengambil data yang tersimpan di dalam *database*.

## 1.5. Manfaat Penelitian

Adapun manfaat dari penelitian adalah

1. Optimalisasi kegiatan pertanian. Petani dan peneliti dapat memantau kondisi tanaman secara real-time, melalui data sensor yang dikirimkan oleh smart ICS
2. Memudahkan para peneliti terkait dan petani untuk mengolah data yang diperlukan.
3. Menyimpan yang dikirimkan oleh Intelligence Controlling System kedalam sebuah database yang dapat diambnil dan digunakan pada penelitian lebih lanjut

**1.7. Sistematika Penulisan**

Sistematika penulisan dari penulisan laporan kerja praktik ini adalah:

## I. PENDAHULUAN

Bab ini menjelaskan tentang latar belakang, tujuan penelitian, rumusan masalah, batasan masalah, manfaat penelitian, dan sistematika penulisan pada penelitian ini.

## II. TINJAUAN PUSTAKA

Membahas tentang penelitian-penelitian sebelumnya pada tinjauan pustaka, dan dasar-dasar teori dari penelitian pengembangan dan implementasi backend dashboard iot pada integrated monitoring system (ics) smart greenhouse menggunakan protokol mqtt

## III METODOLOGI PENELITIAN

Menjelaskan waktu, tempat, alat dan bahan, dan metode penelitian beserta tahapannya mengenai pengembangan dan implementasi backend dashboard iot pada integrated monitoring system (ics) smart greenhouse menggunakan protokol mqtt

**DAFTAR PUSTAKA**

Bab ini berisikan referensi dari penulisan dan pelaksanaan riset.

# BAB II TINJAUAN PUSTAKA

## 2.1 Penelitian Terdahulu

Penelitian sebelumnya mengambil beberapa penelitian terkait sebagai referensi ataupun acuan untuk penelitian yang dilakukan.

Pada penelitian yang dilakukan oleh Inung Bagus Prasetyo pada tahun 2020 menjelaskan tentang perkembangan sistem kontrol dan *monitoring* dari *wireless sensor network (wsn)* dengan memanfaatkan teknologi *MQTT* sebagai pengiriman datanya. Pada penelitian ini terdapat sebuah sistem *WSN* yang dapat menerima data dari sensor suhu DHT-11 dan memiliki *sink node* sebagai perantara *server* kontrol dan *monitoring*. *Sink node* tersebut dihubungkan dengan *broker MQTT* dalam jaringan lokal. *MQTT* digunakan sebagai jalur komunikasi dikarenakan kelebihan dari protokol komunikasi ini yaitu sederhana, ringan, hemat energi, dan mudah diaplikasikan [1]. *nomor referensi ditempatkan sebelum tanda baca.*

Terdapat juga penelitian lain yang membahas tentang *MQTT* yang dilakukan oleh Sri Mulyono pada tahun 2018. Sistem ini menggunakan *mikrokontroller* ESP32 yang sudah terintegrasi dengan *wifi* untuk menerima data berupa suhu dan kelembaban udara dari sensor DHT11, kemudian data tersebut dikirim menggunakan protokol *MQTT* ke *Esp* dimana *Esp* disini berfungsi sebagai *server* *MQTT* [2].

Protokol komunikasi MQTT juga digunakan dalam penelitian yang dilakukan oleh Sudha Senthilkumar pada tahun 2020 mengenai sistem manajemen dan pengendalian kecelakaan yang menggunakan protokol MQTT sebagai pengiriman komunikasinya. Penelitian ini meneliti bagaimana membangun sebuah Vehichle Tracking System (VTS) yang dapat mengirimkan data secara realtime. Penelitian ini menggunakan nodemcu yang dihubungkan dengan tilt sensor dan juga GPS pada kendaraan, yang kemudian dikirimkan ke server [3].

Tidak jauh berbeda dengan penelitian yang telah disebutkan sebelumnya, pada penelitian yang dilakukan oleh Rifqi Zumadilla Pratama dan Heru Nurwasito pada tahun 2019 meneliti tentang penggunaan MQTT pada sistem pemantauan daya listrik rumah. Sistem ini memanfaatkan sensor PZEM-004T yang digunakan sebagai sensor pengukur tegangan listrik AC yang dihubungkan dengan mikrokontroler nodemcu. Sistem yang dibangun ini menggunakan MQTT Broker dari cloudMQTT yang digunakan sebagai pengiriman data yang telah diambil dari sensor ke broker. Data yang telah di publish ke broker, kemudian di subscribe ke sebuah dashboard interface yang dapat diakses user untuk menampilkan data yang telah diambil . [4]

Terdapat juga sebuah penelitian yang dilakukan oleh Arif Hidayat pada tahun 2020. Penelitian ini membahas tentang proses membangun sebuah Virtual Private Network (VPN) dengan menggunakan Ocean Cloud Server yang akan digunakan pada BMT. Mentari East Lampung. Pada penelitian ini VPS di install sistem operasi Windows 10 dengan akses Remote Desktop Protocol (RDP). VPS ini juga dapat digunakan sebagai layanan web server . [5]

Pada penelitian yang dilakukan oleh Ardian Nurrahman pada tahun 2020, terdapat sebuah sistem yang juga memanfaatkan virtual private server dalam pengembangannya. Penelitian ini membangun sebuah sistem yang dapat melakukan pemantauan dan kontrol suhu dengan memanfaatkan photovoltaic. Sistem ini menggunakan Esp sebagai pusat kontrol dan juga tempat penyimpanan data sementara yang nantinya akan dikirimkan ke virtual private sever untuk penyimpanan monitoring data dalam jumlah besar . [6]

Penelitian lain yang digunakan sebagai referensi dalam membangun sistem ini adalah penelitian yang dilakukan oleh Irfan Kurniawan dkk pada tahun 2020 mengenai pembuatan REST API yang digunakan dalam membangun mobile apps yang digunakan sebagai pencatatan transaksi elektronik yang dibagun menggunakan Node.js. Pada penelitian ini dibuat sebuah API yang menghubungkan front-end yang merupakan antarmuka pengguna dengan back-end dimana proses pengambilan data dari database dilakuka. [7]

## 2.2. *Smart Greenhouse*

Smart Greenhouse merupakan evolusi dari rumah kaca tradisional, di mana teknologi sensor dan otomatisasi digunakan untuk meningkatkan efisiensi dan kontrol lingkungan pertanian. Dengan memanfaatkan sensor suhu, kelembaban, intensitas cahaya, dan CO2, serta sistem otomatisasi seperti aktuator dan kontrol komputer, Smart Greenhouse memberikan pendekatan yang terintegrasi untuk meningkatkan produktivitas dan keberlanjutan pertanian. Sistem ini memungkinkan client untuk memantau dan mengontrol kondisi tumbuh tanaman secara real-time, membuat penyesuaian berdasarkan data, dan bahkan melakukan tindakan otomatis untuk menjaga kondisi optimal.

## 2.3. *Intelligence Controlling System ( ICS )*

Intelligence Controlling System (ICS) adalah sistem terpadu yang dirancang untuk mengoptimalkan dan mengontrol berbagai aspek dalam pertanian, termasuk pemantauan lingkungan, irigasi, pemupukan, dan manajemen sumber daya secara menyeluruh. ICS menggunakan teknologi sensor, kecerdasan buatan, dan konektivitas berbasis Internet of Things (IoT) untuk memberikan solusi terpadu yang memungkinkan petani untuk mengambil keputusan berdasarkan data secara akurat dan efisien. Salah satu elemen kunci dari ICS adalah integrasi data dari berbagai sensor yang terdistribusi di seluruh lahan pertanian. Sensor ini dapat mencakup sensor suhu, kelembaban tanah, tingkat nutrisi, dan lainnya. Data yang dikumpulkan oleh sensor-sensor ini kemudian diolah oleh sistem kecerdasan buatan untuk memberikan pemahaman yang lebih mendalam tentang kondisi pertanian.

## 2.4. *Internet Of Things (IoT)*

*Internet of Things* (IoT) merupakan sebuah konsep yang sangat populer di masa ini dimana sebuah perangkat ataupun objek dapat ditanamkan sebuah teknologi seperti sensor yang dapat berkomunikasi, mengendalikan, menghubungkan dan bertukar data dengan perangkat lain dengan memanfaatkan internet.

Pada dasarnya IoT adalah sebuah yang dilakukan secara berkala dalam pengambilan data dari sebuah perangkat ataupun sensor yang kemudian data tersebut dikirimkan menggunakan jaringan internet sebagai jalur komunikasi antara perangkat dengan *server* IoT, dimana *server* dapat berperan sebagai penyimpanan data. Manfaat yang didapat dalam penggunaan *Internet of Things* (IoT) ialah membuat pekerjaan yang dilakukan dapat menjadi lebih cepat, mudah dan efisien sehingga hal ini dapat meningkatkan kinerja penggunanya .

Pada penelitian ini, pengemembangan sistem yang dilakukan dalam penelitian ini menerapkan konsep IoT, dimana terdapat sebuah sensor pemantauan yang terhubung dengan internet, dan sensor tersebut dapat diakses datanya secara jarak jauh menggunakan jaringan internet.

## 2.5. *Message Queue Telemetry Transport (MQTT)*

*Message Queue Telemetry Transport (MQTT)* adalah sebuah protokol standar baru yang banyak digunakan dalam pengembangan sebuah sistem yang berbasis *Internet of Things (IoT).* Sistem ini sangat cocok dalam penggunaannya pada perangkat IoT dikarenakan protokol ini sangat ringan, mudah digunakan, dan hemat energi. *MQTT* dalam pengembangannya saat ini sudah banyak digunakan dalam berbagai industry seperti otomotif, manufaktur, telekomunikasi, minyak dan gas.

*MQTT* saat ini menjadi pilihan sebagai jalur komunikasi IoT dikarenakan beberapa kelebihan seperti :

1. Ringan dan Efisien.

*MQTT-clients* yang sangat ringan, hanya membutuhkan sedikit sumberdaya dalam penggunaan nya sehingga dapat digunakan dalam mikrokontroller dengan memori berukuran kecil. Dalam pengiriman datanya, *MQTT Message Header* yang berfungsi sebagai identitas alamat komunikasi juga sangat ringan sehingga penggunaan *network bandwidth* juga sangat optimal.

1. Pengiriman pesan yang reliabel.

Reliabilitas dalam pengiriman pesan sangat penting dalam berbagai perangkat IoT. *MQTT* memiliki 3 level *Quality of Service (QoS)* yang dapat disesuaikan.

1. Komunikasi dua arah.

*MQTT* dapat melakukan pengiriman pesan antara *device* ke *cloud* dan juga *cloud* ke *device*. Hal ini memudahkan membagikan pesan dalam group dengan *broadcasting*.

1. Dukungan untuk jaringan yang tidak dapat diandalkan.

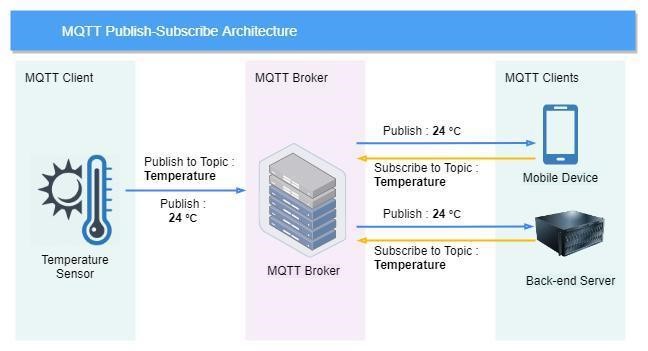
Banyak perangkat IoT terhubung melalui jaringan seluler yang tidak dapat diandalkan. *MQTT* dapat dijalankan secara persisten sehingga mengurangi waktu untuk menghubungkan kembali antara klien dengan *broker*.

1. Skala besar.

*MQTT* dapat terhubung dengan jutaan perangkat IoT.

1. Aman.

*MQTT* memudahkan untuk mengenkripsi pesan menggunakan TLS dan mengautentikasi klien menggunakan protokol autentikasi modern.



Gambar 2. Cara Kerja MQTT *…jika bukan gambar sendiri, sebutkan sumbernya/nomor referensinya.*

Cara kerja dari *MQTT* adalah dengan menggunakan metode *publish / subsribe.* Metode ini bekerja dengan cara satu perangkat mengirimkan sebuah pesan (*publish*), kemudian perangkat lain yang bertindak sebagai penerima (*subscribe*) dapat menerima pesan yang telah dikirimkan tadi sesuai dengan topic yang sesuai. Seluruh pengiriman dan penerimaan pesan ini diatur oleh *MQTT* Broker [14].

Pada sistem yang akan dibangun dalam penelitian ini, *MQTT* yang digunakan sebagai broker adalah Mosquitto. Program *MQTT* tersebut akan diinstall ke sebuah *Virtual Private Server* *(VPS)* dan dapat diakses dari luar jaringan *VPS* sebagai protokol komunikasi dan pengiriman data. Pemilihan menggunakan Broker *MQTT* Mosquitto adalah karena kelebihannya yang ringan, mudah untuk digunakan, gratis, *Open-source*, dan tersedia di berbagai *platform*.

## 2.6. Mosquitto MQTT



## Gambar 3. MQTT Broker Mosquitto [25]

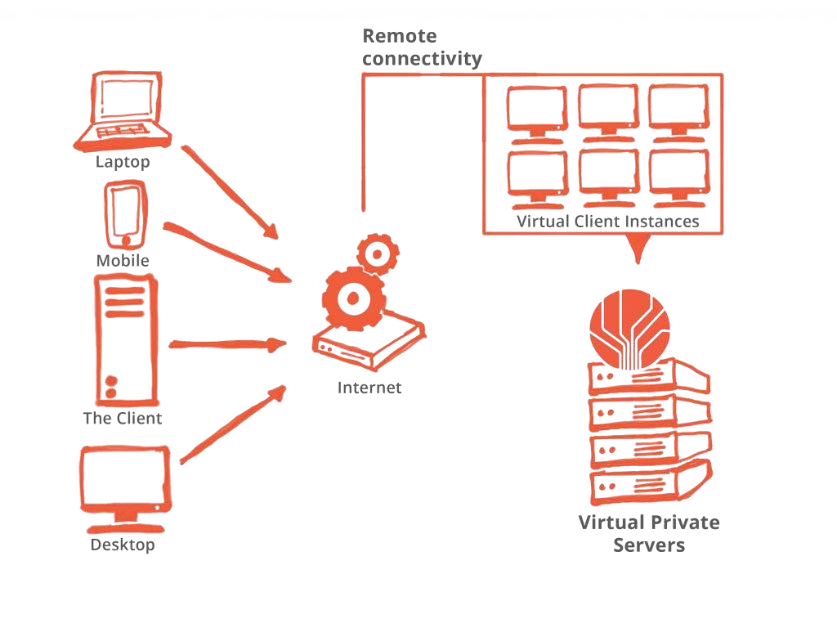
Mosquitto adalah sebuah perangkat lunak open-source yang berfungsi sebagai message broker untuk protokol *MQTT (Message Queuing Telemetry Transport)*. Mosquitto dikembangkan oleh *Eclipse Foundation* dan tersedia untuk digunakan secara bebas dengan lisensi EPL/EDL.

Mosquitto dirancang untuk menjadi sebuah message broker yang ringan dan sederhana, sehingga cocok digunakan pada berbagai jenis perangkat, dari komputer papan tunggal hingga *server* penuh. Mosquitto juga menyediakan perpustakaan C untuk mengimplementasikan klien *MQTT*, serta klien *MQTT* pada baris perintah seperti mosquitto\_pub dan mosquitto\_sub yang sangat populer.

Mosquitto mendukung beberapa versi protokol *MQTT*, yaitu versi 5.0, 3.1.1, dan 3.1, sehingga dapat digunakan pada berbagai jenis aplikasi. Protokol *MQTT* yang digunakan oleh Mosquitto menggunakan model *publish*/*subscribe* yang sangat efisien untuk mengirim dan menerima pesan, dan dapat digunakan untuk berbagai jenis aplikasi, termasuk pada *Internet of Things (IoT)* dan perangkat berdaya rendah.

Beberapa fitur penting dari Mosquitto antara lain adanya dukungan untuk autentikasi dan enkripsi, adanya dukungan untuk multiple broker, serta kemampuan untuk mengatur kualitas layanan (QoS) pada pengiriman pesan. Mosquitto juga dapat diintegrasikan dengan berbagai jenis perangkat dan *platform*, sehingga menjadi solusi yang fleksibel dan dapat disesuaikan dengan kebutuhan pengguna .

### 2.7. *Virtual Private Server (VPS)*



Gambar 4. *Virtual Private* *Server* [15]

*Virtual Private Server (VPS)* merupakan sebuah teknologi virtualisasi dari sebuah *physical server*. beberapa *Virtual Private Server* dapat dibuat dalam sebuah *physical server*. *VPS* dapat berjalan seperti sebuah *server* mandiri yang memiliki *full root access*, sistem operasi, konfigurasi, user, pemrosesan, *CPU*, dan *RAM* nya masing-masing sehingga sebuah *VPS* tidak akan mempengaruhi *VPS* lain dalam satu *physical server* [15].

Pada penelitian ini digunakan sebuah *VPS* dikarenakan dalam membangun sebuah protokol komunikasi pengiriman data dibutuhkan sebuah layanan yang dapat beroperasi terus menerus. Apabila tidak menggunakan *VPS* melainkan menggunakan komputer/laptop secara lokal, terdapat kemungkinan bahwa perangkat tersebut tidak dapat dinyalakan secara terus menerus sehingga protocol komunikasi data juga dapat terganggu. Penggunaan *VPS* pada penelitian ini juga sangat penting dikarenakan seluruh proses pengiriman dan penyimpanan data akan dilayani oleh *VPS* secara terus menerus.

### 2.8. Backend Development

Pengembangan Backend pada Dashboard IoT Smart Greenhouse mencakup pembangunan infrastruktur yang mendukung fungsi dan integrasi semua komponen dalam sistem. Dashboard ini menjadi jantung operasional yang mengelola data dari sensor, memproses informasi, dan menyajikan hasilnya dalam bentuk yang dapat dimengerti oleh pengguna. Fungsi utama dari Backend pada Dashboard IoT Smart Greenhouse melibatkan:

1. Manajemen Data Sensor: Backend mengatur penerimaan dan penyimpanan data dari sensor-sensor yang terpasang di dalam greenhouse.
2. Integrasi dan Pemrosesan Data: Data yang dikumpulkan dari sensor diolah dan diintegrasikan oleh Backend untuk memberikan gambaran keseluruhan tentang kondisi lingkungan dalam greenhouse.
3. Interaksi dengan Database: Backend berkomunikasi dengan database untuk menyimpan dan mengelola data secara efisien. Ini memastikan keberlanjutan dan ketersediaan data untuk dianalisis atau ditampilkan pada dashboard.
4. API untuk Koneksi Frontend: Backend menyediakan API untuk menghubungkan Dashboard dengan Frontend (antarmuka pengguna). **2.9. Node.js**



## Gambar 5. Node.js

Node.js adalah sebuah *platform runtime JavaScript* yang dibangun di atas mesin JavaScript V8 milik Google yang berguna untuk proses development aplikasi secara cepat dan efisien serta mempermudah pembangunan aplikasi berbasis jaringan yang memiliki *scalability* ( daya pengembangan ) yang tinggi, Node Js menggunakan *event-driven non-blocking* I/O model yang membuat Node Js ringan dan efisen, cocok untuk aplikasi data-intensive realtime yang berjalan pada cross *platform*.

Dengan menggunakan Node.js, pengembang dapat menulis kode JavaScript yang dapat dijalankan di sisi *server*, mengakses sistem file, dan melakukan tugas lainnya yang memerlukan akses ke sumber daya sistem.

Beberapa manfaat yang ditawarkan oleh Node.js adalah kecepatan eksekusi, skalabilitas, kemampuan untuk mengatasi banyak permintaan (concurrency), dan kemudahan penggunaan .

### 2.10. Express.js



## Gambar 6. Express.js

Express.js adalah kerangka kerja web yang dirancang untuk mempermudah pengembangan aplikasi web dengan menggunakan bahasa pemrograman

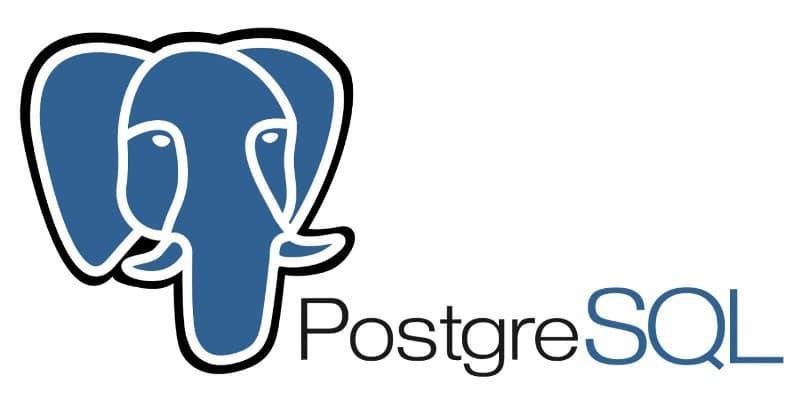
JavaScript di sisi server. Dibangun di atas Node.js, Express.js menyediakan cara yang cepat dan minimalis untuk membuat server HTTP dan mengelola rute aplikasi. Salah satu kekuatan Express.js adalah pendekatannya yang ringkas, memberikan fleksibilitas kepada pengembang untuk membangun aplikasi dengan struktur yang mudah dimengerti.

Express.js memudahkan penanganan permintaan HTTP dengan memungkinkan definisi rute-rute yang menentukan cara server menanggapi permintaan tertentu. Ini memungkinkan pengembang untuk membuat aplikasi web dengan mudah, termasuk pembuatan endpoint API, pengaturan middleware, dan manajemen sesi.

Selain itu, Express.js memiliki sistem templat yang dapat diintegrasikan dengan berbagai mesin templat, memfasilitasi pembuatan tampilan dinamis.

Kerangka kerja ini juga mendukung pengembangan aplikasi RESTful dengan menyediakan alat untuk mengelola metode HTTP seperti GET, POST, PUT, dan DELETE. Express.js juga mendukung penggunaan middleware, yang dapat digunakan untuk menangani berbagai aspek seperti otentikasi, log, dan manipulasi permintaan sebelum mencapai penanganan utama.

### 2.11. PostgreSQL



## Gambar 7. PostgreeSQL

PostgreSQL merupakan salah satu sistem basis data yang tergolong kedalam DBMS atau *Database Management System* yaitu suatu sistem perangkat lunak yang kegunaanya didesain untuk membantu pekerjaan penggunanya seperti mengelola suatu basis data dan menjalankan suatu perintah operasi terhadap data yang diminta oleh banyak penggunanya. PostgreSQL digunakan sebagai manajemen basis data yang menampung dan mengolah data aplikasi dan dapat dijalankan pada sistem operasi seperti *Windows, Linux, Mac,* dan lain sebagainya.

PostgreSQL dapat digunakan sebagai aplikasi manajemen *database* dengan aplikasi bawaannya pgAdmin. PostgreSQL bekerja dengan menangani proses pengolahan data dengan sintaks query SQL.

PostgreSQL dikenal sebagai sistem pengolahan *database* open source termaju di dunia yang umum digunakan dalam pengembangan perangkat lunak dengan kemampuannya yang mampu memproses banyak sintaks query dalam satu waktu .

PostgreSQL dipilih dalam penelitian ini sebagai *database* penyimpanan data dikarenakan beberapa kelebihan dari PostgreSQL yaitu *Open-source*, skalabilitas nya yang baik, mudah dikonfigurasi, dapat menangani volume data yang besar, dan lebih cepat dibandingkan beberapa *Database* *Management System* (DBMS) yang lain seperti MySQL atau MongoDB .

### 2.12. Application Programming Interface (API)

*Application Programming Interface (API)* adalah sebuah antarmuka yang memungkinkan aplikasi untuk berinteraksi dengan aplikasi atau layanan lainnya secara programatik. API menyediakan kumpulan instruksi dan protokol yang dapat digunakan oleh pengembang untuk membangun aplikasi yang dapat berkomunikasi dengan sistem atau layanan yang telah ada .

API memungkinkan pengembang untuk mengakses fungsi dan layanan yang disediakan oleh sistem atau layanan lainnya tanpa harus mengetahui seluruh detail implementasi di belakangnya. Dengan menggunakan API, pengembang dapat mempercepat proses pengembangan, mengurangi biaya, dan meningkatkan efisiensi dalam pengembangan aplikasi karena tidak perlu membangun seluruh fitur dari awal.

Contoh penggunaan API adalah ketika sebuah aplikasi mengakses data dari layanan seperti Facebook, Twitter, atau Google Maps. API yang disediakan oleh layanan tersebut memungkinkan aplikasi untuk mengambil data dari layanan tersebut dan menggunakannya dalam aplikasi mereka sendiri .

### 2.13. REST-API

REST API adalah salah satu jenis dari API yang didasarkan pada arsitektur REST *(Representational State Transfer).* *REST* sendiri adalah sebuah arsitektur perangkat lunak yang terdiri dari aturan dan konvensi yang digunakan untuk membuat web service.

Dalam REST API, setiap sumber daya (*resource*) diidentifikasi dengan URI (*Uniform Resource Identifier*) dan diakses melalui metode HTTP seperti GET, POST, PUT, DELETE, dan sebagainya.

Selain itu, REST API juga menggunakan format data yang ringan dan terstruktur, seperti JSON (*JavaScript Object Notation*) atau XML (*eXtensible Markup Language*) untuk pertukaran data antara aplikasi dan *server*. Format data ini memudahkan pengolahan data oleh aplikasi klien (*client*) yang mengakses API. Dalam penggunaannya, REST API sering digunakan oleh pengembang aplikasi untuk mengintegrasikan berbagai aplikasi atau layanan secara terdistribusi dan scalable. Selain itu, REST API juga sering digunakan dalam pembuatan aplikasi mobile, *Internet of Things* (IoT), dan integrasi antar sistem yang berbeda *platform*. **2.14. GitHub**

*GitHub* adalah *platform hosting* kode sumber berbasis *web* yang sangat populer di kalangan pengembang perangkat lunak. *Platform* ini menyediakan repositori *git* publik dan pribadi yang memungkinkan pengguna untuk menyimpan, mengelola, dan berkolaborasi dalam pengembangan perangkat lunak secara terdistribusi. *GitHub* memiliki fitur-fitur kolaborasi yang kuat, yang memungkinkan tim pengembang bekerja bersama dalam proyek perangkat lunak.

### 2.15. Visual Studio Code

Visual Studio Code adalah editor teks gratis dan *open-source* yang dikembangkan oleh Microsoft. Visual Studio Code memungkinkan pengguna untuk menulis kode pada berbagai bahasa pemrograman, termasuk HTML, CSS, JavaScript, PHP, Python, dan lain-lain. Visual Studio Code berjalan pada sistem operasi Windows, Mac, dan Linux. Editor teks ini dilengkapi dengan fitur-fitur seperti fitur debugging, Git integration, autocomplete, snippet, syntax highlighting, dan lainlain .

Visual Studio Code memiliki beberapa kelebihan yang membuatnya populer di kalangan pengembang perangkat lunak. Kelebihan-kelebihan tersebut antara lain adalah :

* Gratis dan *open-source*
* Ringan dan cepat
* Mudah digunakan
* Mendukung banyak bahasa pemrograman
* Memiliki banyak ekstensi dan plugin

### 2.16. Postman

Postman adalah sebuah alat yang digunakan oleh para pengembang perangkat lunak untuk berinteraksi dengan API (Application Programming Interface). API seperti pelayan di restoran, yang mengambil pesanan Anda (permintaan data atau fungsi) dan memberikan balasan dari dapur (sistem atau aplikasi). Dengan Postman, pengembang bisa menguji API dengan mengirimkan permintaan-nya dan melihat apa yang dikembalikan. Misalnya, jika Anda ingin melihat data pengguna dari sebuah layanan, Anda bisa "meminta" data tersebut melalui Postman, dan layanan akan "memberikan" data yang diminta .

Postman sendiri kemudian digunakan untuk mengirim permintaan ke API.

“Permintaan dapat mengambil (GET), menambah (POST), menghapus (DELETE), atau memperbarui data (PUT)”. Permintaan dapat digunakan untuk mengirim parameter, detail login, informasi otorisasi, atau data tubuh lain yang diperlukan. Perangkat lunak Postman dapat diunduh ke komputer pengguna (sistem operasi Windows, Mac, dan Linux didukung) atau digunakan dari peramban web dengan versi Web Postman .

# BAB III METODE PENELITIAN

## 3.1. Waktu dan Tempat Penelitian

Penelitian dan pembuatan Tugas Akhir ini dilakukan pada :

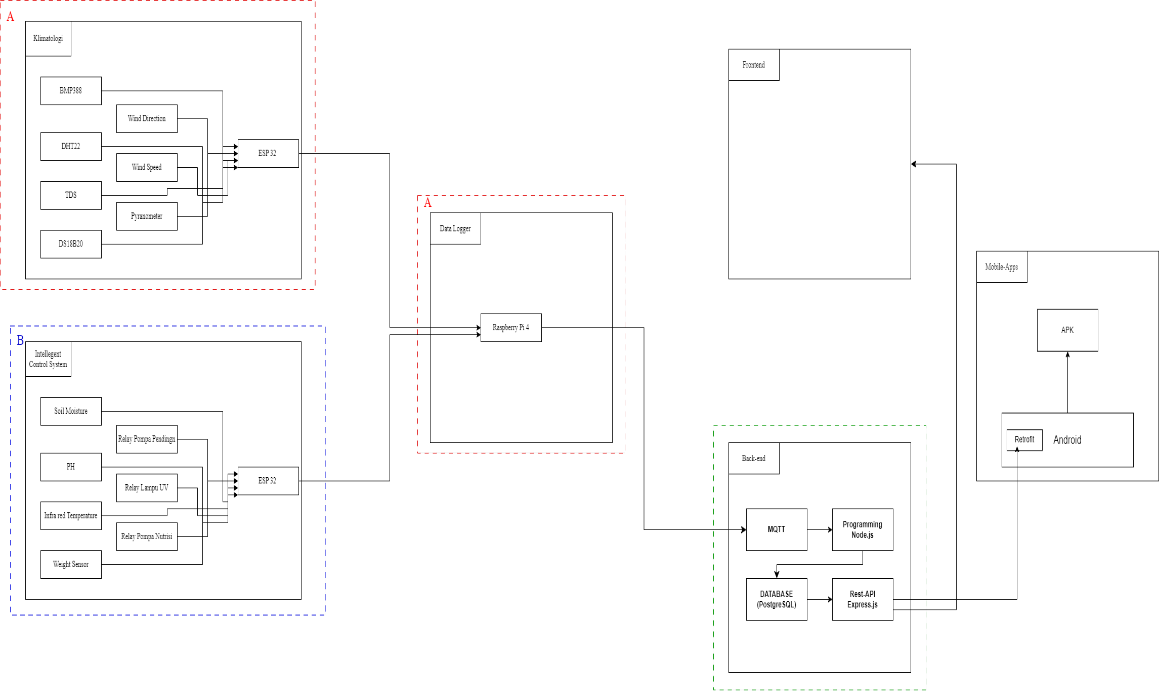
Waktu : Agustus 2023 – Januari 2023

Tempat :

● Laboratorium Teknik Komputer, Teknik Elektro, Universitas Lampung ● Laboratorium terpadu fakultas pertanian universitas lampung

## 3.2. Capstone Project

Pengerjaan penelitian ini berkaitan dengan beberapa penelitian lain yang berkolaborasi untuk membangun sistem Intelligence Controlling System ( ICS ) pada smart greenhouse. Diagram keseluruhan project yang dikerjakan adalah sebagai berikut :



## Gambar 8. Diagram keseluruhan pengembangan sistem

Dalam project yang akan dikembangkan tersebut terdapat beberapa bagian yang akan dikerjakan secara terpisah. Beberapa bagian tersebut adalah sebagai berikut :

Tabel 1. Tim pengembangan project

|  |  |  |
| --- | --- | --- |
| No | Nama | Bagian |
| 1 | M. Herly Pratama | Hardware |
| 2 | Auliya Syahputra Siregar | Hardware |
| 3 | M. Affan Siddiqie Asmara | Back-end Developer |
| 4 | Alprealian Renando | Front-end Developer  Web |
| 5 | K.G.S Ade Iqbal | Mobile App Dev |

Dalam pengerjaan proyek ini, peneliti hanya akan berfokus pada bagian *server*, sementara bagian lain seperti perangkat ICS yang merupakan rangkaian hardware sensor pemantauan dan juga bagian *front-end* sebagai dashboard monitoring akan dikerjakan dalam penelitian lain.

*Server* dalam proyek ini berfungsi sebagai penghubung antara perangkat sensor dengan *front-end*, dimana didalam *server* itu sendiri akan menjalankan berbagai service yang dibutuhkan dalam proyek ini seperti *database*, *MQTT Broker*, dan *API*.

### 3.3. Alat dan Bahan

#### 3.3.1. Alat

Alat yang digunakan dalam pengerjaan penelitian dan pengembangan sistem yang dibuat adalah sebagai berikut :

Tabel 2. Alat berupa Hardware dan Software yang digunakan.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Perangkat | Spesifikasi | Kegunaan | Keterangan |
| 1 | Laptop | Processor Intel Core i5 10300H, RAM 16GB, SSD 512GB,  Sistem Operasi  Hackintosh | Perangkat pembuatan dan pengujian sistem. |  |
| 2 | Virtual Private  *Server* (VPS) | Penyedia :  Niagahoster.co.id,  Lokasi: Singapore, 3  CPU Core, RAM  3GB, Storage 60GB. | *Virtual private* *server* yang digunakan sebagai pusat layanan sistem yang akan dibuat. | Diakses melalui *Secure Shell*  *(SSH)* |
| 3 | MQTTX | MQTTX Versi 1.9.6 | Software yang digunakan untuk pengujian broker mqtt dalam mengirim dan menerima data | Terinstall di  Laptop |
| 4 | Visual Studio  Code | VS Code versi 1.84.2 | Software yang digunakan dalam penulisan kode program untuk pengambilan dan | Terinstall di  Laptop |
|  |  |  | penyimpanan data dari sensor |  |
| 3 | PostgreSQL | PostgreSQL *database* versi 7.4 | Program *database* yang digunakan dalam penyimpanan data dari sensor | Terinstall di  VPS dan laptop |
| 4 | Mosquitto  MQTT  Broker | Mosquitto versi  1.6.10 | Program yang di install di dalam  VPS sebagai broker  MQTT | Terinstall di  VPS dan laptop |
| 5 | Node.js &  NPM | Node.js versi v16.19.0. NPM versi  8.19.3 | Aplikasi yang digunakan untuk menjalankan kode program untuk penyimpanan data | Terinstall di  VPS dan laptop |
| 6 | Postman | Postman V.10.19 | Program yang digunakan untuk checking Endpoint  API | Terinstall di  Laptop |

#### 3.3.2. Bahan

Bahan yang digunakan dalam pengerjaan penelitian dan pengembangan sistem ini berupa data yang didapatkan dari sensor ICS adalah sebagai berikut :

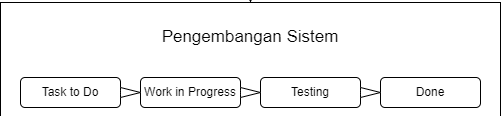
Tabel 3. Bahan berupa data sensor dari *esp*.

|  |  |  |
| --- | --- | --- |
| **NO** | **Data yang diterima** | **Keterangan** |
| 1 | Timestamp | Data yang menunjukan waktu |
| 2 | Ph | Data yang menunjukan tingkat keasaman atau kebasaan air |

|  |  |  |
| --- | --- | --- |
| 3 | Tds | Data kualitas air |
| 4 | Suhu air | Data yang menunjukan suhu air |
| 5 | Arah mata angin | Data yang menunjukan arah mata angin pada greenhouse |
| 6 | Kecepatan angin | Data yang menunjukan kecepatan angin dalam greenhouse |
| 7 | Waterflow 1*…water flow tidak usah*  *dibahas di skripsi* | Data yang mengukur kecepatan arus air yang mengalir pada meja 1 |
| 8 | Waterflow 2 | Data yang mengukur kecepatan arus air yang mengalir pada meja 2 |
| 9 | Waterflow 3 | Data yang mengukur kecepatan arus air yang mengalir pada meja 3 |
| 10 | Waterflow 4 | Data yang mengukur kecepatan arus air yang mengalir pada meja 4 |
| 11 | Soilmoisture  1*…soilmoisture cukup 1 saja yang*  *dibahas di skripsi* | Data yang menunjukan kelembapan tanah pada tumbuhan di meja 1 |
| 12 | Soilmoisture 2 | Data yang menunjukan kelembapan tanah pada tumbuhan di meja 2 |
| 13 | Soilmoisture 3 | Data yang menunjukan kelembapan tanah pada tumbuhan di meja 3 |
| 14 | Soilmoisture 4 | Data yang menunjukan kelembapan tanah pada tumbuhan di meja 4 |
| 15 | Suhu ruangan | Data yang menunjukan suhu greenhouse |
| 16 | Tekanan udara | Data yang menunjukan tekanan udara di dalam greenhouse |
| 17 | Pompa nutrisi | Data yang menunjukan apakah sistem pompa nutrisi |
|  |  | menyala tau tidak |
| 18 | Pompa air | Data yang menunjukan apakah sistem pompa air menyala atau tidak |
| 19 | Kelembapan | Data yang menunjukan kelembapan didalam greenhouse |

### 3.4. Tahapan Penelitian

Pengerjaan dalam penelitian ini dilakukan secara bertahap mulai dari studi literatur lalu dilanjutkan dengan tahapan perancangan sistem menggunakan metode Kanban lalu di akhiri dengan tahapan Analisa hasil. Berikut ini adalah diagram alir dari tahapan yang dilakukan.



## Gambar 9. Diagram alir tahapan penelitian

### 3.5. Analisa Kebutuhan

Penentuan kebutuhan didapatkan setelah mendapatkan beberapa data berupa literasi yang berkaitan dengan pengembangan dan implementasi *backend* *dashboard* *iot* pada *integrated monitoring system (ics) smart greenhouse* menggunakan protokol *mqtt* yang selanjutnya akan dicari informasi yang berhubungan dengan aktivitas yang akan dilakukan dalam pengembangan sistem.

Berdasarkan studi pustaka dan melihat kondisi ICS pada smart greenhouse ini telah berjalan, maka dapat ditentukan apa saja yang dibutuhkan dalam penelitian ini untuk mengatasi beberapa kelemahan yang ada pada sistem yang saat ini sedang berjalan.

Adapun setelah dilakukannya identifikasi terhadap tujuan pengembangan aplikasi dan kebutuhan, didapat beberapa kebutuhan yang dapat diselesaikan dengan penenilian ini antara lain :

1. *Virtual Private Server* yang dapat berfungsi sebagai *MQTT Broker* dan penyimpanan data.
2. *MQTT Broker* yang dapat digunakan sebagai protokol pengiriman data antara sensor pemantauan dengan dashboard dan *database*.
3. Data yang didapat dari sensor harus dapat disimpan dalam *database* VPS dan tidak boleh hilang dikarenakan data tersebut dapat digunakan sebagai data yang akan digunakan dalam dashboard dan sebagai bahan penelitian lebih lanjut.
4. Data yang tersimpan di *database* bisa dikirimkan ke *dashboard* atau *front-end* menggunakan *API.*

Berdasarkan beberapa kebutuhan tersebut, maka didapatkan beberapa *task* yang harus dilakukan. *Task* tersebut adalah sebagai berikut :

Tabel 4. *Task* yang dilakukan dalam pengembangan sistem.

|  |  |
| --- | --- |
| **No** | **Tugas *(Task)* yang dilakukan** |
| 1. | Desain arsitektur sistem |
| 2. | Inisialisai dan membangun arsitektur awal di local |
| 3. | Persiapan dan konfigurasi MQTT *Broker* di VPS |
| 4. | Tahap testing menggunakan dummy data |
| 5. | Konfigurasi *ESP* untuk mengirimkan data sensor |
| 6. | Inisialisasi dan persiapan Virtual Private *Server* (VPS) |
| 7. | Membuat *database* untuk menyimpan data dari *Esp* |
| 8. | Menerima data dari *ESP* dengan protokol MQTT |
| 9. | Pengkodean untuk penyimpanan data sensor dari *ESP* ke *Database* menggunakan node.js |
| 10. | Mengambil data pada database dan membuat API MQTT untuk mengirim data ke *front-end* menggunakan node.js |

### 3.6. Pengembangan Sistem

Dalam penelitian ini, metode yang digunakan dalam pengembangan sistem akan mengikuti model Kanban Board. Berikut ini adalah tahapan yang akan dilaksanakan dalam pengembangan menggunakan metode Kanban.



## Gambar 10. Tahapan pengembangan sistem dengan Metode Kanban

Pada gambar 3.2 dapat dilihat bahwa pada penelitian yang dilakukan memiliki beberapa tahapan sesuai dengan metode Kanban yang digunakan. Tahap yang dilakukan adalah sebagai berikut :

### 3.7. Tahap *Task To Do*

Pada tahap *task to do, task* tugas yang akan diklasifikasikan dengan skala tingkat prioritas. Pada tahap ini terdapat 3 skala prioritas:

1. Prioritas Tinggi

Pada tugas yang prioritas tinggi, menunjukan tugas tersebut berprioritas tinggi untuk dikerjakan.Prioritas tinggi menandakan bahwa tugas tersebut adalah tugas yang penting dan krusial untuk membuat sistem dapat berjalan dengan baik. Apabila tugas tersebut tidak dilakukan, maka akan mengganggu jalan nya sistem atau sistem bisa tidak berjalan sama sekali.

1. Prioritas Sedang

Tugas dengan prioritas menengah menandakan bahwa tugas tersebut adalah tugas yang penting untuk dikerjakan namun tugas tersebut tidak akan mengganggu kinerja sistem apabila belum dikerjakan.

1. Prioritas Rendah

Prioritas rendah dalam pengembangan sistem ini menandakan bahwa tugas tersebut merupakan tugas opsional yang apabila belum dikerjakan tidak akan mengganggu kinerja sistem

Penentuan prioritas dari setiap tugas yang dilakukan adalah sebagai berikut :

Tabel . Penentuan prioritas dari tugas yang dilakukan.

### 3.8. Tahap *Work in Progress*

Bagian ini merupakan bagian dari *Task to Do yang sedang dikerjakan* akan dipindahkan ke bagian ini. Sesuai dengan prinsip dari Kanban yaitu membatasi tugas yang sedang berlangsung, maka tugas yang diletakan pada bagian ini akan dibatasi jumlah tugasnya pada setiap *flow*nya. Pembatasan tugas pada tahap ini berfungsi untuk menghindari terjadinya *overproduction* atau pemborosan dari pengerjaan banyak tugas sekaligus, sehingga fokus dapat lebih terarahkan pada menyelesaikan pekerjaan yang sedang dalam proses sebelum memulai pekerjaan baru.

Dalam penelitian ini ditentukan batasan dari tahapan ini adalah 2 buah *task* yang dapat dikerjakan secara bersamaan. Hal ini dikarenakan dalam pengembangan sistem ini dilakukan secara individu, sehingga membatasi dua buah pekerjaan sudah cukup efektif dalam penyelesaiannya.

**3.9. Tahap *Testing***

**3.10. Tahap *Done***

**3.11. Analisa Hasil**

# DAFTAR PUSTAKA

1. I. B. Prasetyo, M. Faiqurahman, dan Z. Sari, “Rancang Bangun Control and Monitoring Sensor Node WSN Menggunakan Protokol Message Queue Telemetry Transport,” *Repositor* , vol. 2, no. 1, hlm. 15–26, 2020.
2. S. Mulyono dan S. F. C. Haviana, “Implementasi MQTT untuk Pemantauan Suhu dan Kelembaban pada Laboratorium,” *Jurnal Transistor Elektro dan Informatika (TRANSISTOR EI*, vol. 3, no. 3, hlm. 140–144, 2018.
3. R. Z. Pratama dan H. Nurwarsito, “Monitoring Penggunaan Daya Listrik menggunakan Protokol MQTT berbasis Web,” *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 11, hlm. 10820–10826, 2019, [Daring]. Tersedia pada: http://j-ptiik.ub.ac.id
4. S. Senthilkumar, K. Brindha, dan S. Bhandari, “Vehicle Accident Management and Control System Using MQTT,” *International Journal of Advances in Applied Sciences*, vol. 9, no. 1, hlm. 1, Mar 2020, doi: 10.11591/ijaas.v9.i1.pp1-11.
5. A. Hidayat dan D. Prabowo, “Implementation of Virtual Private Server (VPS)

Using Digital Ocean Cloud Server on BMT. Mentari East Lampung,” *Jurnal JTKSI*, vol. 03, no. 03, hlm. 116–121, 2020.

1. A. Nurrahman dan N. A. Z, “Sistem Kontrol Suhu dengan PID dan Monitoring

Daya Output Pada Panel Photovoltaic Portable dengan Virtual Private Server,” *Jurnal Maestro*, vol. 3, no. 2, 2020.

1. I. Kurniawan, Humairah, dan F. Rozi, “REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android,” *Jurnal Ilmiah Teknologi Sistem Informasi*, vol. 1, no. 4, hlm. 127–132, 2020, [Daring]. Tersedia pada: http://jurnal-itsi.org
2. S. Mulyono dan S. F. C. Haviana, “Implementasi MQTT untuk Pemantauan Suhu dan Kelembaban pada Laboratorium,” *Jurnal Transistor Elektro dan Informatika (TRANSISTOR EI*, vol. 3, no. 3, hlm. 140–144, 2018.
3. R. Z. Pratama dan H. Nurwarsito, “Monitoring Penggunaan Daya Listrik menggunakan Protokol MQTT berbasis Web,” *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 11, hlm. 10820–10826, 2019, [Daring]. Tersedia pada: http://j-ptiik.ub.ac.id
4. S. Senthilkumar, K. Brindha, dan S. Bhandari, “Vehicle Accident Management and Control System Using MQTT,” *International Journal of Advances in Applied Sciences*, vol. 9, no. 1, hlm. 1, Mar 2020, doi: 10.11591/ijaas.v9.i1.pp1-11.
5. A. Hidayat dan D. Prabowo, “Implementation of Virtual Private Server (VPS)

Using Digital Ocean Cloud Server on BMT. Mentari East Lampung,” *Jurnal JTKSI*, vol. 03, no. 03, hlm. 116–121, 2020.

1. A. Nurrahman dan N. A. Z, “Sistem Kontrol Suhu dengan PID dan Monitoring

Daya Output Pada Panel Photovoltaic Portable dengan Virtual Private Server,” *Jurnal Maestro*, vol. 3, no. 2, 2020.