PENGEMBANGAN *BACKEND* SISTEM PEMANTAUAN CUACA BERBASIS IOT DENGAN NEST.JS PADA *MICROCLIMATE STATION* DI KAWASAN HUTAN MANGROVE PETENGORAN

(Skripsi)

Oleh

Witra Tama Adiansya
2115031037

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
2025

ABSTRAK

PENGEMBANGAN *BACKEND* SISTEM PEMANTAUAN CUACA BERBASIS IOT DENGAN NEST.JS PADA *MICROCLIMATE STATION* DI KAWASAN HUTAN MANGROVE PETENGORAN

OLEH

WITRA TAMA ADIANSYA

Hutan mangrove di Desa Gebang, Kecamatan Teluk Pandan, Kabupaten Pesawaran, Provinsi Lampung merupakan salah satu ekosistem pesisir yang penting dan memerlukan pemantauan lingkungan secara berkelanjutan untuk mendukung upaya pelestarian. Penelitian ini bertujuan mengembangkan *backend* sistem pemantauan cuaca berbasis *Internet of Things (IoT)* menggunakan *Nest.js* pada *microclimate station* yang dipasang di kawasan hutan mangrove Petengoran. Sistem ini dirancang untuk mengintegrasikan protokol *MQTT* dan keamanan *TLS* guna mengirimkan data sensor cuaca secara *real-time*. Parameter yang dipantau meliputi suhu udara, kelembapan relatif, curah hujan, kecepatan dan arah angin, tekanan atmosfer, kadar oksigen dalam air, dan radiasi matahari. Data yang diterima disimpan secara terstruktur dalam basis data *PostgreSQL* dan disajikan melalui *REST-API* untuk mendukung kebutuhan integrasi dan visualisasi data. Pengembangan *backend* mengikuti metode *Kanban*, dengan pembagian tugas pada papan kerja *To Do, In Progress*, dan *Done*. Hasil pengujian menunjukkan sistem mampu mengirimkan data secara stabil melalui *MQTT*, serta menyimpan dan mengelola data secara efisien. Rata-rata waktu kueri pada *PostgreSQL* tercatat antara 0,236–0,683 milidetik. Respons *REST-API* juga menunjukkan performa baik, dengan waktu respons 287 milidetik pada *endpoint* latest/station dan 1100 milidetik pada daily/station. Secara keseluruhan, sistem terbukti dapat beroperasi andal meskipun berada di lokasi dengan konektivitas jaringan yang terbatas. Penelitian ini memberikan kontribusi terhadap pengembangan sistem pemantauan lingkungan berbasis *IoT* untuk kawasan pesisir, sebagai bagian dari dukungan terhadap pelestarian ekosistem mangrove secara digital.

Kata Kunci: Hutan Mangrove, *IoT*, *MQTT*, *Nest.js*, Pemantauan Cuaca, *PostgreSQL*, *REST API*.

ABSTRACT

BACKEND DEVELOPMENT OF AN IOT-BASED WEATHER MONITORING SYSTEM USING NEST.JS FOR A MICROCLIMATE STATION IN THE PETENGORAN MANGROVE FOREST AREA

By

WITRA TAMA ADIANSYA

The Petengoran mangrove forest in Gebang Village, District of Teluk Pandan, Pesawaran Regency, Lampung, is a vital coastal ecosystem that requires continuous environmental monitoring to support conservation efforts. This study aims to develop a backend for an Internet of Things (IoT) based weather monitoring system using Nest.js for a microclimate station installed in the Petengoran mangrove area. The system is designed to integrate the MQTT protocol and TLS security to transmit weather sensor data in real-time. Observed parameters include air temperature, relative humidity, rainfall, wind speed and direction, atmospheric pressure, absorbed oxygen levels in water, and solar radiation. The collected data is stored in a structured format using PostgreSQL database and exposed through a REST API to support data integration and visualization purposes. The backend system was developed using the Kanban method, with task distribution organized on To Do, In Progress, and Done boards. Testing results show that the system reliably transmits data via MQTT and store and manage data effiiciently. The average query time in PostgreSQL ranged from 0.236 to 0.683 milliseconds. The REST API also demonstrated good performance, with response times of 287 milliseconds for the latest/station endpoint and 1.100 milliseconds for daily/station. Overall the system is proven to operate reliably even under limited network connectivity conditions. This research contributes to the development of IoT-based environmental monitoring systems for coastal areas, as part of digital support for mangrove ecosystems conservation.

Keywords: *Mangrove Forest*, IoT, MQTT, *Nest*.js, *Weather Monitoring, PostgreSQL, REST API.*

Judul Skripsi	:	**PENGEMBANGAN *BACKEND* SISTEM PEMANTAUAN CUACA BERBASIS IOT DENGAN NEST.JS PADA *MICROCLIMATE STATION* DI KAWASAN HUTAN MANGROVE PETENGORAN**
Nama Mahasiswa	:	**Witra Tama Adiansya**
Nomor Pokok Mahasiswa	:	**2115031037**
Program Studi	:	**Teknik Elektro**
Fakultas	:	**Teknik**

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Yetti Yuniati, S.T., M.T.	**Dr.-ing. Ardian Ulvan, S.T., M.Sc.**
NIP. 198001132009122002	NIP. 197311281999031005

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Elektro
Herlinawati, S.T., M.T	**Sumadi, S.T., M.T.**
NIP. 197103141999032001	NIP. 197311042000031001

MENGESAHKAN

1. Tim Penguji

Ketua : **Yetti Yuniati, S.T., M.T.** .....

Sekretaris : **Dr.-ing.Ardian Ulvan, S.T., M.Sc., Ph.D.** .....

Penguji : **Misfa Susanto, S.T., M.Sc., Ph.D.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 197509282001121002

Tanggal Lulus Ujian Skripsi: 19 September 2025

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa dalam skripsi yang saya buat dengan judul "Pengembangan *Backend* Sistem Pemantauan Cuaca Berbasis Iot Dengan Nest.Js Pada *Microclimate Station* Di Kawasan Hutan Mangrove Petengoran" dibuat tidak berdasarkan karya yang pernah dilakukan orang lain. Bahwa karya ini tidak terdapat karya lain atau pendapat yang ditulis atau diterbitkan orang lain, kecuali secara tertulis diacu dalam naskah ini sebagaimana yang disebutkan di dalam daftar Pustaka. Selain itu saya menyatakan pula bahwa skripsi ini dibuat oleh saya sendiri. Apabila saya tidak benar, maka saya bersedia dikenai sangsi sesuai dengan hukum yang berlaku.

Bandar Lampung, 5 Desember 2025

Witra Tama Adiansya
NPM. 2155031037

RIWAYAT HIDUP

Penulis, Witra Tama Adiansya, lahir di Lampung Barat pada tanggal 18 Desember 2001. Penulis saat ini berdomisili di Bandar Lampung, Lampung. Riwayat pendidikan penulis dimulai dari Sekolah Dasar di SDS Tunas Bangsa yang diselesaikan pada tahun 2014. Pendidikan Sekolah Menengah Pertama diselesaikan di SMPN 1 Dente Teladas pada tahun 2017, dan Sekolah Menengah Atas di SMA Negeri 1 Way Tenong diselesaikan pada tahun 2020. Pada tahun 2021, penulis terdaftar sebagai mahasiswa di Jurusan Teknik Elektro, Fakultas Teknik, Universitas Lampung.

Selama masa perkuliahan, penulis aktif dalam beberapa kegiatan organisasi dan kepanitiaan, di antaranya menjadi bagian dari Kepengurusan Himpunan Mahasiswa Teknik Elektro (2022-2023) di Departemen Pengembangan Keteknikan. Penulis juga pernah berpartisipasi dalam program Program Penguatan Kapasitas Organisasi Kemahasiswaan (PPK Ormawa) tahun 2024 dengan topik "*Drone Rover*", yang berhasil lolos Pendanaan. Selain itu, penulis mengikuti kegiatan Merdeka Belajar Kampus Merdeka (MBKM) studi independen di *Startup Campus* dengan fokus ilmu data (*Data Science*).

MOTTO

"Tetaplah santai dalam setiap tekanan."
(Penulis)

PERSEMBAHAN

Setiap langkah dalam perjalanan ini adalah jejak cinta, doa, dan pengorbanan. Untuk itu, dengan penuh rasa syukur, kupersembahkan kepada:

Papa dan Mama

Serta

Kakak-kakakku dan Adik-adikku

Terima kasih telah membesarkan dan membimbing dengan penuh kasih sayang, selalu memotivasi dan memberi dukungan moril maupun materi, selalu mendoakan kesuksesan Penulis, serta segala bentuk pengorbanan dan semua hal yang telah diberikan kepada Penulis yang tidak akan pernah bisa terbalas, semoga Allah selalu melindungi kalian.

Dan juga rekan-rekan

Excalto 2021

TELTI 2021

HIMATRO UNILA

Terima kasih atas kebersamaan, motivasi, kerjasama, dan rasa kekeluargaan yang luar biasa. Dukungan kalian telah membuat proses studi ini penuh warna dan makna

SANWACANA

Alhamdulillahirabbil'alamin. Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayat-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi/tugas akhir ini dengan judul "Pengembangan *Backend* Sistem Pemantauan Cuaca Berbasis Iot Dengan Nest.Js Pada *Microclimate Station* Di Kawasan Hutan Mangrove Petengoran". Dalam pelaksanaan dan pembuatan skripsi ini penulis menerima dukungan baik secara moril maupun materil yang sangat berharga dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih ke semua pihak yang telah membantu, khususnya kepada:

1. Kedua orang tua tercinta dan seluruh keluarga penulis yang tidak hentinya mendo'akan serta memberikan dorongan semangat dan materi;
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung;
3. Ibu Herlinawati, S.T., M.T. selaku Ketua Jurusan Teknik Elektro Universitas Lampung;
4. Ibu Yetti Yuniati, S.T., M.T. selaku Dosen Pembimbing Utama yang telah memberikan ilmu, bimbingan, bantuan, masukan, dan pandangan kehidupan kepada penulis disetiap kesempatan dengan baik dan ramah;
5. Bapak Dr.-ing. Ardian Ulvan, S.T., M.Sc., Ph.D. selaku Dosen Pembimbing Pendamping yang telah memberikan ilmu, bimbingan, masukan, damotivasi kepada penulis disetiap kesempatan dengan baik dan ramah;
6. Bapak Misfa Susanto, S.T., M.Sc., Ph.D. selaku Dosen Penguji yang telah memberikan kritik dan saran yang membangun kepada penulis;

7. Bapak Ir. Noer Soedjarwanto, M.T. selaku Pembimbing Akademik yang telah memberikan arahan, nasehat dan bimbingan yang membangun bagi penulis selama menempuh perkuliahan;
8. Segenap dosen di Jurusan Teknik Elektro yang telah memberikan ilmu yang bermanfaat kepada penulis selama menempuh pendidikan perkuliahan;
9. Segenap staff di Jurusan Teknik Elektro yang telah membantu penulis baik dalam hal administrasi dan lain-lain;
10. Keluarga Besar Laboratorium Telekomunikasi yang memberikan banyak ilmu, masukan dan saran kepada penulis yang tidak bisa dibayarkan;
11. Keluarga Besar MCS, Akmal Dwiki Wijaya, Haris Nasution, dan Firmansyah yang telah Bersama-sama memberikan semangat dan gagasan dalam menyelesaikan skripsi ini;
12. Teknik Elektro'21, terimakasih atas semangat dan kebersamaannya selama menempuh pendidikan di Jurusan Teknik Elektro;
13. Keluarga Besar HIMATRO yang telah memberikan ilmu dan pengalaman kepada penulis selama pendidikan baik secara langsung maupun tak langsung;
14. Mahasiswi Bimbingan Konseling dengan NPM 2413052064, terimakasih telah menemani semester akhir penulis sehingga menjadi lebih berwarna;
15. Seluruh pihak yang tidak bisa disebutkan satu persatu yang telah membantu penulis dalam proses pembuatan skripsi baik secara langsung maupun tidak langsung.

Penulis mengakui adanya kekurangan dalam skripsi ini dan dengan tulus menerima kritik serta saran yang membangun dari berbagai pihak demi kemajuan bersama. Semoga skripsi ini bermanfaat bagi penulis dan pembaca.

Bandar Lampung, 5 Desember 2025

Penulis

Witra Tama Adiansya

DAFTAR ISI

ABSTRAK ..... **ii**
ABSTRACT ..... **iii**
SURAT PERNYATAAN ..... **vi**
RIWAYAT HIDUP ..... **vii**
MOTTO ..... **viii**
SANWACANA ..... **x**
DAFTAR ISI ..... **xii**
DAFTAR GAMBAR ..... **xii**
DAFTAR TABEL ..... **xvi**
I. PENDAHULUAN **17**
1.1 Latar Belakang 17
1.2 Tujuan Penelitian 3
1.3 Batasan Masalah 3
1.4 Rumusan Masalah 4
1.5 Manfaat Penelitian 4
1.6 Sistematika Penulisan 5
II. TINJAUAN PUSTAKA **7**
2.1 Penelitian Terkait 7
2.2 *Microclimate Station* (MCS) 13
2.3 *Tranport Layer Security* (TLS) 13
2.4 *Message Queuing Telemetry Transport* (MQTT) 14
2.5 *Mosquitto* MQTT 14
2.6 *Virtual Private Server* (VPS) 15
2.7 *Backend Development* 15
2.8 Node.js 15
2.9 Nest.js 16
2.10 PostgreSQL 17
2.11 *Application Programming Interface* (API) 17
2.12 RESTful API 17
2.13 GitHub 18
2.14 *Visual Studio Code* 18
2.15 *Python* 19
2.18 Zenkit 19

III. METODOLOGI PENELITIAN 20
3.1 Waktu dan Tempat Penelitian 20
3.2 *Capstone Project* 20
3.3 Alat dan Bahan 22
3.3.1 Alat 22
3.3.2 Bahan 24
3.4 Tahapan Penelitian 25
3.5 Studi Literatur dan Analisa Kebutuhan 26
3.6 Pengembangan Sistem 31
3.6.1 Tahap *To Do* 33
3.6.2 Tahap *In Progress* 33
3.6.3 Tahap *Testing* 33
3.6.4 Tahap *Done* 34
3.7 Tahapan Pengujian 36
3.8 Indikator Keberhasilan 37
IV. HASIL DAN PEMBAHASAN 38
4.1 Proses Pengembangan Sistem 38
4.1.1 Pembuatan Kartu Tugas 38
4.1.2 *Flow* Pertama 39
4.1.2 *Flow* Kedua 47
4.1.3 *Flow* Ketiga 51
4.2 *Testing Backend* 62
4.2.1 *Flow* Keempat 62
4.3 *Deployment System* 65
4.3.1 *Flow* Kelima 65
4.3.2 *Flow* Keenam 69
4.4 Analisa Hasil 72
4.4.1 Kecepatan *Respons Endpoint* 72
4.4.2 Analisa Performa *query* dengan *EXPLAIN ANALYZE* 73
4.4.3 Indikator Keberhasilan 74
BAB V KESIMPULAN DAN SARAN 76
5.1 Kesimpulan 76
5.2 Saran 77
DAFTAR PUSTAKA 78
LAMPIRAN 80

DAFTAR GAMBAR

Gambar 3. 1 Alur Penelitian.....27
Gambar 3. 2 *Capstone Project*..... xiv
Gambar 3. 3 Tahapan Penelitian.....26
Gambar 3. 4 Tahapan *Backend* 44
Gambar 3. 5 *Input* dan *output backend*.....47
Gambar 3. 6 *Platform* Zenkit.....51
Gambar 4. 1 Kartu Tugas……39
Gambar 4. 2 *Flow* Pertama.....39
Gambar 4. 3 Desain Arsitektur Sistem.....40
Gambar 4. 4 *Handler* Program.....42
Gambar 4. 5 *Rest* API *Response and Request*.....43
Gambar 4. 6 *Flow* Kedua 47
Gambar 4. 7 *file* .env.....49
Gambar 4. 8 *Subscribe Topic*.....49
Gambar 4. 9 Konfigurasi Database 50
Gambar 4. 10 Tabel *Climate* 51
Gambar 4. 11 Konfigurasi *Handler Backend*.....51
Gambar 4. 12 Folder Konfigurasi *Backend*.....52
Gambar 4. 13 *Main*.ts.....53
Gambar 4. 14 App.*module*.ts.....54
Gambar 4.16 App.*controller*.ts 55
Gambar 4. 17 MQTT.dto.ts.....56
Gambar 4. 18 MQTT.*service*.ts.....56
Gambar 4. 19 MQTT.*controller*.ts.....57
Gambar 4. 20 MQTT.*module*.ts 58
Gambar 4. 21 Petengoran.*entity*.ts 66

Gambar 4. 22 Petengoran.dto.ts 59
Gambar 4. 23 Petengoran.*service*.ts 60
Gambar 4. 24 Petengoran.*controller*.ts 61
Gambar 4. 25 Petengoran.*module*.ts 61
Gambar 4. 26 Flow Pengujian *Backend* 62
Gambar 4. 27 *Running* npm *run start* 63
Gambar 4. 28 Tampilan *Rest API* 64
Gambar 4. 29 Tampilan *Endpoint* 64
Gambar 4. 30 Tahapan *Deployment System* 66
Gambar 4. 31 Tampilan *Github* 66
Gambar 4. 32 PM2 *Start* 69
Gambar 4. 33 PM2 *Logs* 70
Gambar 4. 34 *Testing Rest* API 70
Gambar 4. 35 *Testing Endpoint* 71
Gambar 4. 36 *Query EXPLAIN ANALYZE* 73

DAFTAR TABEL

Tabel 3. 1 Tahapan Penelitian.....20
Tabel 3. 2 Alat berupa hardware dan software yang digunakan.....22
Tabel 3. 3 Data Sensor.....25
Tabel 3. 4 Indikator Keberhasilan.....37
Tabel 4. 1 *Station*.....44
Tabel 4. 2 *First_station*.....45
Tabel 4. 3 *Second_station*.....46
Tabel 4. 4 *Topic* MQTT.....48
Tabel 4. 5 Pengujian dengan *Postman*.....72
Tabel 4. 6 Pengujian dengan *EXPLAIN ANALYZE*.....74

I. PENDAHULUAN

1.1 Latar Belakang

Hutan mangrove adalah sumber daya alam yang sangat penting di daerah tropis, dengan manfaat yang besar baik dari sisi ekologi maupun ekonomi. Indonesia, sebagai negara kepulauan, memiliki hutan mangrove terluas di dunia. Ekosistem mangrove memainkan peran yang krusial dalam menjaga keseimbangan ekosistem pesisir dan laut [1]. Hutan mangrove juga menciptakan mikroiklim unik dengan perbedaan signifikan antara di atas dan di bawah kanopi. Di bawah kanopi, suhu udara lebih stabil (3-4°C lebih hangat pada malam dingin), kelembapan lebih tinggi, kecepatan angin turun hingga 63%, dan curah hujan yang mencapai tanah berkurang 19-20% karena intersepsi kanopi. Cahaya matahari yang terbatas di bawah kanopi dapat menghambat pertumbuhan bibit mangrove, yang memerlukan cahaya penuh dan hujan langsung untuk mengurangi salinitas tanah dan mendukung hidrasi, sehingga bibit sering tumbuh lebih baik di celah kanopi atau area terbuka. Namun, dalam lima dekade terakhir, luas hutan mangrove global menurun drastis akibat konversi lahan untuk pertanian dan akuakultur pesisir [2].

Pelestarian dan pengelolaan mangrove yang berkelanjutan, dibutuhkan pendekatan berbasis teknologi yang mampu memberikan informasi terkini tentang kondisi lingkungan. Penggunaan teknologi *Internet of Things* (IoT) menawarkan solusi modern untuk memantau berbagai parameter cuaca dan lingkungan, seperti suhu udara, kelembapan, curah hujan, serta kecepatan dan arah angin. Data yang diperoleh dari perangkat IoT ini dapat digunakan untuk memahami pola cuaca yang berdampak langsung pada kesehatan ekosistem mangrove [3].

Sistem pemantauan berbasis IoT membutuhkan *backend* yang handal untuk mengintegrasikan data dari sensor, menyimpannya secara terstruktur dan menyediakannya dalam bentuk visualisasi yang mudah dipahami. Pengembangan *backend* ini bukan hanya tentang teknologi, tetapi juga tentang bagaimana sistem tersebut dapat menjawab kebutuhan lokal, seperti keterbatasan jaringan di kawasan terpencil dan tantangan lingkungan yang keras. Selain itu, *backend* harus dirancang untuk memastikan keamanan data, efisiensi komunikasi, dan kemampuan untuk menangani volume data yang terus meningkat, mengingat potensi pengembangan dan integrasi dengan perangkat lain di masa depan. Hal ini menjadikan pengembangan *backend* sebagai elemen kunci yang tidak hanya mendukung fungsi teknis tetapi juga keberlanjutan sistem dalam jangka panjang [4].

Pengembangan *backend* untuk sistem pemantauan cuaca berbasis IoT yang dijelaskan dalam penelitian ini berfokus pada perancangan sistem inti yang mengelola data dari perangkat sensor di lapangan. *Backend* bertugas menerima data cuaca, menyimpannya dalam basis data yang terstruktur, dan memastikan bahwa data tersebut dapat diakses secara *real-time* melalui antarmuka visualisasi. Dalam penelitian yang berjudul "Pengembangan *Backend* untuk Sistem Pemantauan Cuaca Berbasis IoT dengan Nest.js pada *Microclimate Station* di Kawasan Hutan Mangrove Petengoran," pendekatan pengembangan *backend* yang kuat dan efisien menjadi kunci untuk menyediakan informasi cuaca yang akurat dan cepat kepada pengguna. Hal ini mendukung respon cepat terhadap perubahan cuaca yang dapat memengaruhi keberlanjutan ekosistem mangrove.

Selain mengelola data, *backend* juga dirancang untuk menangani tantangan teknis seperti konektivitas yang terbatas di kawasan terpencil dan kebutuhan akan efisiensi pemrosesan data. Melalui protokol komunikasi seperti MQTT dan penyimpanan data berbasis *cloud*, sistem ini memungkinkan data dikirim dan diakses dengan stabil meskipun berada di daerah dengan infrastruktur jaringan yang terbatas. Penelitian ini tidak hanya berkontribusi pada pengembangan teknologi IoT untuk pengelolaan lingkungan, tetapi juga membuka peluang untuk inovasi lebih lanjut di bidang pemantauan berbasis sensor, terutama dalam pengelolaan ekosistem penting seperti hutan mangrove.

1.2 Tujuan Penelitian

Tujuan penelitian ini adalah sebagai berikut :

1. Merancang dan mengimplementasikan *backend* sistem pemantauan cuaca berbasis IoT menggunakan Nest.js yang terintegrasi dengan protokol MQTT berkeamanan TLS pada *microclimate station* di kawasan hutan mangrove Petengoran.
2. Membangun mekanisme pengelolaan data cuaca yang terstruktur menggunakan PostgreSQL dan penyediaan REST-API, sehingga data hasil pengukuran berbagai parameter cuaca (suhu, kelembapan, curah hujan, angin, tekanan, oksigen, dan radiasi matahari) dapat diakses dan diintegrasikan untuk keperluan visualisasi dan analisis.
3. Mengevaluasi kinerja dan keandalan *backend* sistem pemantauan cuaca berdasarkan stabilitas pengiriman data melalui MQTT, waktu kueri basis data, serta waktu *respons REST-API*, khususnya pada kondisi jaringan dengan konektivitas terbatas di kawasan hutan mangrove Petengoran.

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut :

1. Penelitian hanya berfokus pada perancangan dan implementasi *backend* (Nest.js, MQTT, TLS, PostgreSQL, dan REST-API). Pengembangan antarmuka pengguna (*frontend/web dashboard*) dan analisis lanjutan terhadap data cuaca yang dikumpulkan tidak dibahas secara mendalam
2. Sistem diuji dan diimplementasikan hanya pada satu kawasan studi, yaitu hutan mangrove Petengoran di Desa Gebang, Pesawaran. Integrasi dengan *microclimate station* di lokasi lain atau dengan tipe perangkat IoT berbeda berada di luar cakupan penelitian.
3. Pengujian kinerja hanya mencakup stabilitas pengiriman data melalui protokol MQTT, waktu kueri basis data PostgreSQL, dan waktu *respons*

REST-API. Evaluasi aspek keamanan lanjutan, skalabilitas untuk beban pengguna besar, dan tidak menjadi fokus utama penelitian ini.

1.4 Rumusan Masalah

Rumusan masalah pada penelitian ini adalah sebagai berikut :

1. Bagaimana merancang dan mengimplementasikan *backend* sistem pemantauan cuaca berbasis IoT menggunakan Nest.js yang terintegrasi dengan protokol MQTT berkeamanan TLS pada *microclimate station* di kawasan hutan mangrove Petengoran?
2. Bagaimana merancang mekanisme penyimpanan dan pengelolaan data cuaca secara terstruktur menggunakan PostgreSQL serta menyediakan *REST-API* yang memungkinkan data dapat diakses dan diintegrasikan untuk keperluan visualisasi dan analisis?
3. Bagaimana kinerja *backend* sistem pemantauan cuaca yang dikembangkan jika dievaluasi dari aspek stabilitas pengiriman data melalui MQTT, waktu kueri basis data PostgreSQL, dan waktu *respons REST-API*?
4. Sejauh mana *backend* sistem pemantauan cuaca yang dikembangkan mampu beroperasi secara andal pada kondisi jaringan dengan konektivitas terbatas di kawasan hutan mangrove Petengoran, serta kendala teknis apa saja yang ditemui selama pengujian?

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

1. Meningkatkan pemantauan lingkungan dengan memberikan data cuaca *real-time* yang akurat, yang mendukung pengelolaan hutan mangrove.
2. Menerapkan teknologi untuk keberlanjutan dalam pengelolaan sumber daya alam melalui sistem IoT yang efisien dan penyimpanan data terstruktur.
3. Menyediakan akses data yang mudah dan terstruktur untuk mendukung penelitian lanjutan dan pengembangan aplikasi cuaca.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan pada laporan penelitian ini adalah sebagai berikut :

BAB I : PENDAHULUAN

Bab ini menjelaskan secara rinci tentang latar belakang yang mendasari dilakukannya penelitian ini, tujuan yang ingin dicapai, serta rumusan masalah yang menjadi fokus utama penelitian. Selain itu, bab ini juga menguraikan tantangan teknis serta solusi yang dihadapi dalam pengembangan sistem pemantauan cuaca berbasis IoT.

BAB II : TINJAUAN PUSTAKA

Membahas tentang penelitian-penelitian sebelumnya pada tinjauan pustaka, dan dasar-dasar teori dari penelitian Pengembangan *Backend* Untuk Sistem Pemantauan Cuaca Berbasis Iot Dengan Visualisasi *Real-Time* Pada *Microclimate Station* Di Kawasan Hutan Mangrove Petengoran.

BAB III : METODOLOGI PENELITIAN

Menjelaskan waktu, tempat, alat dan bahan, dan metode penelitian beserta tahapannya mengenai Pengembangan *Backend* Untuk Sistem Pemantauan Cuaca Berbasis IoT Dengan Nest.js Pada *Microclimate Station* Di Kawasan Hutan Mangrove Petengoran.

BAB IV : HASIL DAN PEMBAHASAN

Memaparkan hasil penelitian yang telah dilakukan sesuai dengan metodologi. Hasil yang diperoleh meliputi implementasi pengembangan *backend* sistem pemantauan cuaca berbasis IoT dengan Nest.js, integrasi dengan *microclimate station* di Kawasan Hutan Mangrove Petengoran, serta pengujian fungsionalitas sistem.

BAB V : KESIMPULAN

Membahas mengenai hasil kesuluruhan penelitian, Kesimpulan dari hasil Analisa dan fungsionalitas teknologi yang diterapkan, dan saran keberlanjutan penelitian selanjutnya.

DAFTAR PUSTAKA

Bab ini menjabarkan seluruh daftar pustaka yang dipakai dalam mendukung teori yang digunakan serta sumber data-data terverifikasi.

LAMPIRAN

Bab ini menjabarkan terkait lampiran-lampiran tambahan yang mendukung dan memperjelas penelitian.

II. TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Peneliti merujuk pada sejumlah penelitian relevan sebagai dasar dan acuan untuk memperkuat studi yang dilakukan.

Dalam studi yang dilakukannya *A. Karim Mohamed Ibrahim* (2019) mengembangkan *middleware IoT* yang ringan untuk mendukung pengembangan aplikasi dengan cepat dan efisien. *Backend* sistem ini dibangun menggunakan *Node.js* karena kesederhanaan dan fleksibilitasnya dalam menangani banyak koneksi. *Middleware* ini dirancang untuk mendukung aplikasi IoT milik individu maupun organisasi dengan memastikan keamanan data di *server* lokal atau berbasis *cloud*. Dilengkapi dengan *API* terstandarisasi yang adaptif melalui abstraksi tingkat tinggi, sistem ini mempermudah integrasi dengan perangkat IoT dan layanan aplikasi lainnya. Sebagai bukti konsep, sistem mencakup satu *endpoint* dan *database* statis, dengan komunikasi antara *klien* dan *server* dilakukan melalui protokol *HTTP* dan format *JSON*. Pengguna dapat berinteraksi dengan sistem menggunakan metode *HTTP* seperti *GET*, *POST*, *DELETE*, dan *PUT*, sehingga memberikan fleksibilitas dalam pengelolaan data [5].

Perbedaan penelitian yang dilakukan oleh *A. Karim Mohamed Ibrahim* dengan penelitian ini terletak pada teknologi dan fitur yang digunakan. Penelitian *Ibrahim* menggunakan protokol *HTTP* sebagai komunikasi data, dengan dukungan *server* lokal atau berbasis *cloud*, dan berfokus pada pengamanan data serta fleksibilitas aplikasi melalui *API* terstandarisasi. Namun, penelitian tersebut tidak mencakup fitur visualisasi langsung sehingga pengguna hanya dapat berinteraksi dengan data

melalui metode *HTTP*. Sementara itu, penelitian ini menggunakan protokol *MQTT* untuk komunikasi data yang lebih efisien dan ringan, *VPS* sebagai infrastruktur *server* yang lebih *scalable* untuk mendukung peningkatan kapasitas data, serta *framework Nest.js* yang memungkinkan visualisasi data cuaca secara *real-time*. Dengan fitur ini, penelitian ini memberikan pengalaman pengguna yang lebih interaktif, memudahkan pemantauan langsung kondisi cuaca, dan meningkatkan efektivitas pengawasan di kawasan mangrove.

Penelitian yang dilakukan oleh Kittasil Silanon dan Kriddikorn Kaewwongsri pada tahun 2020 membahas tentang sistem pemantauan cuaca yang melibatkan beberapa komponen penting dalam pengumpulan dan pengolahan data. Data yang dikumpulkan dari berbagai sensor, seperti suhu, kelembaban, kecepatan angin, dan parameter cuaca lainnya, dikirimkan ke *server cloud* menggunakan teknologi NB-IoT dan protokol CoAP. Setelah data terkumpul, data tersebut disimpan dalam *database* MySQL untuk pengolahan lebih lanjut. Untuk visualisasi, data yang tersimpan di MySQL diolah dan ditampilkan dengan menggunakan perangkat lunak Grafana, yang memungkinkan pengguna untuk memantau kondisi cuaca secara real-time dengan tampilan yang mudah dipahami. Selain itu, penelitian ini juga menghadapi berbagai tantangan dalam transmisi data, seperti batasan transmisi pada NB-IoT dan komunikasi I2C yang digunakan antara sensor dan mikrokontroler. Tantangan ini diatasi dengan memotong string data agar dapat diproses dan dikirimkan dengan efisien, memastikan transmisi data tetap berjalan lancar meskipun dengan batasan yang ada pada teknologi tersebut [6].

Perbedaan yang dilakukan antara penelitian yang dilakukan oleh Kittasil Silanon dan Kriddikorn Kaewwongsri dengan penelitian penulis terletak pada teknologi komunikasi, *platform server, dan* metode visualisasi. Penelitian Kittasil dkk. menggunakan teknologi NB-IoT untuk transmisi data dan protokol CoAP, dengan penyimpanan data menggunakan MySQL. Untuk visualisasi, data yang tersimpan diolah dan ditampilkan menggunakan perangkat lunak Grafana. Sementara itu, penelitian penulis menggunakan *MQTT* sebagai protokol komunikasi data yang lebih efisien untuk IoT, VPS sebagai infrastruktur server yang *scalable*, dan *Express.js* untuk visualisasi data cuaca secara *real-time*. Penelitian Kittasil dkk.

lebih fokus pada pengiriman data menggunakan NB-IoT dan mengatasi tantangan transmisi data dengan pemotongan *string*, sedangkan penelitian penulis lebih mengutamakan efisiensi komunikasi data melalui MQTT dan fleksibilitas server menggunakan VPS.

Penelitian yang dilakukan oleh Asif Rahman Rumee pada tahun 2021 menjelaskan mengenai *prototipe* sistem *monitoring* cuaca di Hutan Bakau Sudarbans, Bangladesh. Bagian *back end* dari sistem prototipe IoT ini melibatkan penggunaan server jarak jauh yang dioperasikan oleh sebuah komputer papan tunggal (SBC-PT), yang setara dengan *Raspberry Pi.* Server ini menerima data dari *prototipe* yang ditempatkan di hutan bakau, menyimpan, dan memproses data sesuai kebutuhan pengguna. Data yang diterima mencakup delapan parameter lingkungan seperti kelembaban, asap, kualitas air, suara, suhu, curah hujan, angin, dan deteksi kebakaran. Dalam situasi darurat, seperti deteksi kebakaran, sistem mengirimkan notifikasi *email* kepada administrator jarak jauh menggunakan protokol SMTP. Pengujian dilakukan menggunakan alat simulasi *Cisco Packet Tracer*, dan data dikirimkan ke server setiap 2 detik menggunakan TCP/IP [7].

Perbedaan penelitian yang dilakukan oleh Asif Rahman Rumee dengan penelitian penulis terletak pada teknologi dan fitur yang digunakan. Penelitian Rumee menggunakan *TCP/IP* sebagai protokol komunikasi dan server berbasis komputer papan tunggal (*SBC-PT*) seperti Raspberry Pi, yang berfokus pada penyimpanan serta pemrosesan data. Penelitian ini dilengkapi dengan mekanisme notifikasi darurat melalui email untuk memberikan peringatan saat terjadi kejadian penting. Sementara itu, penelitian penulis menggunakan *MQTT* sebagai protokol komunikasi data yang lebih efisien, yang memungkinkan pertukaran data lebih cepat dan ringan. Selain itu, penulis menggunakan *VPS* sebagai infrastruktur server yang lebih scalable, Framework *Nest.js* juga digunakan untuk mendukung pengembangan backend yang lebih terstruktur serta menyediakan fitur visualisasi data cuaca secara *real-time*.

Penelitian selanjutnya dilakukan Feldmann berjudul *Deep Dive into the IoT Backend Ecosystem* dipresentasikan pada konferensi *ACM Internet Measurement*

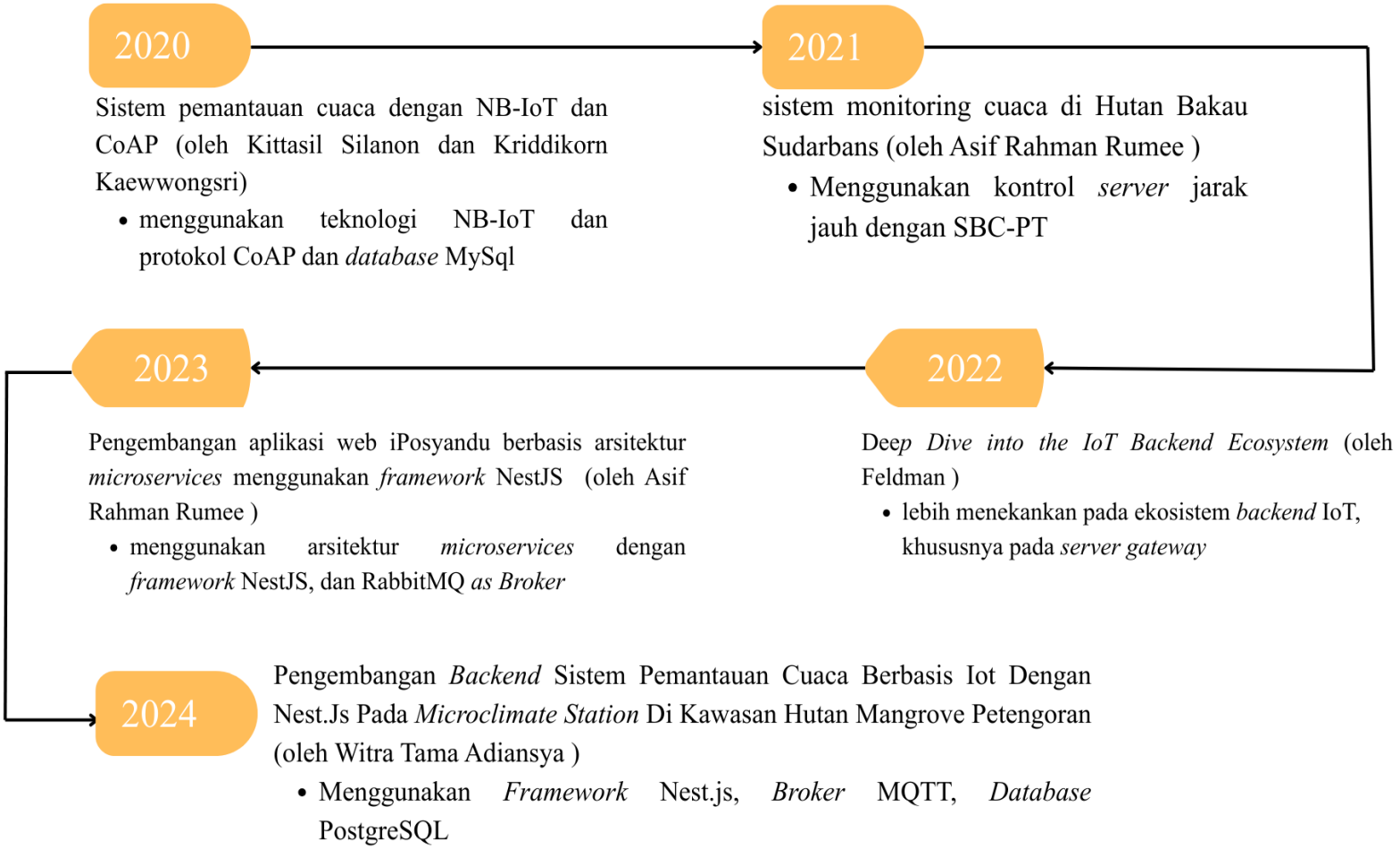
Conference (IMC) pada tanggal 25-27 Oktober 2022 di Nice, Prancis. Penelitian ini berfokus pada ekosistem *backend Internet of Things* (IoT) dan bertujuan untuk memahami bagian publik dari platform IoT, yaitu *server gateway* yang menghadap ke *Internet*, yang disebut sebagai *IoT backend*. Penelitian ini mengidentifikasi dan mengkarakterisasi *IoT backend* yang memfasilitasi pertukaran data antara perangkat IoT, sistem internal, dan mungkin platform lainnya. Fokus utama dari studi ini adalah pada *IP publik* yang memungkinkan pertukaran dan aliran data antara perangkat IoT dan sistem internal platform IoT. Aspek lain seperti perangkat lunak dan perangkat keras pada perangkat IoT, sistem pemrosesan dan penyimpanan internal. Penelitian ini juga mengembangkan metodologi untuk menyimpulkan jaringan dan lokasi fisik dari penyedia *backend* IoT utama, serta menganalisis ekosistem *backend* IoT terkait dengan penyebaran, operasi, dan ketergantungan [8].

Perbedaan penelitian yang dilakukan oleh Feldmann dengan penelitian ini terletak pada fokus dan cakupan sistem yang dianalisis. Penelitian Feldmann lebih menekankan pada ekosistem *backend* IoT, khususnya pada *server gateway* yang menghadap ke *Internet* dan pertukaran data melalui *IP publik* antara perangkat IoT dan sistem internal platform IoT. Penelitian ini berfokus pada aspek jaringan dan lokasi fisik penyedia *backend* IoT serta penyebaran, operasi, dan ketergantungannya. Sementara itu, penelitian penulis lebih fokus pada penggunaan *MQTT* untuk komunikasi data yang efisien, penerapan *VPS* untuk infrastruktur *server* yang *scalable*, serta penggunaan *Nest.js* yang mendukung visualisasi data cuaca secara *real-time*. Penelitian ini juga mengutamakan pemantauan kondisi cuaca di kawasan mangrove dengan pengalaman pengguna yang lebih interaktif.

Penelitian selanjutnya ditulis oleh Chandra Arcychan Azfar pada tahun 2023 sebagai bagian dari tugas akhir untuk Program Studi Teknik Informatika di Universitas Pasundan Bandung. Pengembangan aplikasi web iPosyandu berbasis arsitektur microservices menggunakan framework NestJS berhasil meningkatkan efisiensi dan skalabilitas pengelolaan data posyandu, yang sebelumnya menggunakan arsitektur monolith dan menghadapi tantangan seperti ketidakakuratan data serta kesulitan akses oleh kader kesehatan. NestJS dipilih

karena mendukung struktur modular, *TypeScript* untuk *type safety,* dan pengembangan API yang menjadi tulang punggung komunikasi antar layanan microservices, memungkinkan pembaruan fitur tanpa mengganggu sistem secara keseluruhan. PostgreSQL digunakan sebagai sistem manajemen basis data relasional untuk menyimpan informasi kesehatan ibu dan anak, jadwal kegiatan, serta laporan kader, menawarkan performa tinggi dan keandalan melalui transaksi ACID. Sementara itu, RabbitMQ berperan sebagai *message broker* untuk mengelola komunikasi asinkronus antar layanan, seperti notifikasi dan sinkronisasi data, memastikan ketahanan sistem dengan menyimpan pesan di antrean hingga diproses. Pengembangan dilakukan dengan metodologi Scrum, melibatkan tahapan pengumpulan kebutuhan melalui wawancara dan observasi, perancangan, implementasi, serta pengujian iteratif untuk memastikan performa dan keandalan. Hasilnya, iPosyandu menjadi solusi modern yang mendukung pengelolaan data posyandu secara akurat, efisien, dan mudah diakses, dengan potensi pengembangan fitur baru di masa depan, menjadikannya model untuk aplikasi kesehatan berbasis masyarakat [9].

Perbedaan utama antara penelitian yang dilakukan oleh Chandra Arcychan Azfar dan penelitian penulis terletak pada pendekatan arsitektur sistem serta tujuan implementasi *backend*. Aplikasi iPosyandu dikembangkan menggunakan arsitektur *microservice*s dengan framework NestJS, memanfaatkan RabbitMQ sebagai *message broker* untuk mengelola komunikasi asinkronus antar layanan. Pendekatan ini bertujuan membangun sistem yang skalabel, modular, dan mudah dikelola untuk mendukung pengelolaan data layanan kesehatan masyarakat, seperti pencatatan kesehatan ibu dan anak serta laporan kegiatan posyandu, dengan PostgreSQL sebagai sistem manajemen basis data yang andal. Sebaliknya, penelitian penulis tidak menggunakan arsitektur microservices, melainkan berfokus pada pengolahan data sensor iklim secara *real-time* menggunakan protokol komunikasi MQTT. *Backend* yang juga dibangun dengan NestJS diintegrasikan langsung dengan *broker MQTT* untuk menerima data dari perangkat IoT, menyimpan data ke PostgreSQL, dan berjalan pada VPS untuk menjamin efisiensi dan keandalan. Sistem ini dirancang untuk menyediakan data iklim yang akurat untuk visualisasi di web.



Gambar 2. 1 *State of The Art*

2.2 *Microclimate Station* (MCS)

Microclimate adalah lingkungan iklim lokal yang dikendalikan secara spesifik dalam area terbatas, seperti rumah kaca, fasilitas pertanian, atau zona perkotaan, yang dirancang untuk menciptakan kondisi ideal guna mendukung pertumbuhan tanaman, kenyamanan manusia, atau penyimpanan produk dengan kualitas terjaga. Pengelolaan microclimate melibatkan pengendalian faktor-faktor lingkungan seperti suhu, kelembapan, intensitas cahaya, konsentrasi karbon dioksida, dan sirkulasi udara, yang diatur menggunakan teknologi canggih *seperti Internet of Things* (IoT), *Edge computing*, dan *machine learning* untuk memantau dan mengelola kondisi secara *real-time*. Sistem ini memungkinkan respons cepat terhadap perubahan lingkungan, meningkatkan efisiensi, dan mengurangi ketergantungan pada pengamatan manual. Aplikasi *microclimate* banyak diterapkan dalam pertanian presisi untuk mengoptimalkan hasil panen, penyimpanan produk pertanian untuk menjaga kesegaran, serta perencanaan kota pintar untuk menciptakan lingkungan yang lebih nyaman dan berkelanjutan. Dengan otomatisasi, sistem *microclimate* menghasilkan data yang lebih akurat dan mendukung pengambilan keputusan yang lebih baik, sehingga memberikan manfaat ekonomi dan lingkungan yang signifikan [10].

2.3 *Tranport Layer Security* (TLS)

Transport Layer Security (TLS) adalah protokol keamanan yang digunakan untuk melindungi komunikasi data di jaringan, khususnya melalui internet. TLS merupakan pengembangan dari SSL (*Secure Sockets Layer*) yang lebih dulu digunakan, dengan tujuan memberikan keamanan yang lebih kuat dan fleksibilitas lebih besar.

Protokol ini menyediakan enkripsi, otentikasi, dan integritas data, yang sangat penting untuk melindungi data sensitif selama transmisi, seperti pada transaksi perbankan *online* atau komunikasi *email*. TLS bekerja dengan cara mengenkripsi data yang dikirimkan antara klien dan server untuk mencegah intersepsi dan

modifikasi oleh pihak ketiga. Selain itu, TLS juga memastikan keaslian server dengan menggunakan sertifikat digital yang dikeluarkan oleh otoritas sertifikasi yang terpercaya, sehingga klien dapat memverifikasi identitas server sebelum memulai komunikasi yang aman [11].

2.4 *Message Queuing Telemetry Transport* (MQTT)

MQTT (*Message Queuing Telemetry Transport*) adalah protokol komunikasi yang menggunakan model *publish/subscribe* untuk mengirim pesan. Protokol ini dirancang agar ringan, sehingga cocok untuk sistem M2M (*Machine-to-Machine*) dan IoT (*Internet of Things*). MQTT terdiri dari tiga komponen utama: *Publisher* atau *Producer* (sebagai MQTT *Client*), Broker (sebagai MQTT Server), dan *Consumer atau Subscriber* (sebagai MQTT *Client*).

Broker MQTT bertanggung jawab untuk menerima koneksi jaringan dari klien, menerima pesan aplikasi yang dipublikasikan oleh klien, memproses permintaan berlangganan dan berhenti berlangganan dari klien, mengirim pesan aplikasi ke klien sesuai dengan langganannya, dan menutup koneksi jaringan dari klien. Protokol ini mendukung komunikasi dua arah, yang membantu dalam berbagi data, mengelola, dan mengontrol perangkat.

MQTT juga memiliki beberapa tingkat *Quality of Service* (QoS) yang memastikan pengiriman pesan dengan tingkat keandalan yang berbeda. Protokol ini biasanya memerlukan *header* tetap sebesar 2-*byte* dengan *payload* pesan kecil hingga ukuran maksimum 256 MB [12].

2.5 *Mosquitto* MQTT

Mosquitto MQTT adalah perangkat lunak *broker server* yang digunakan untuk mengimplementasikan sistem pesan berbasis protokol MQTT, yang dirancang untuk komunikasi antar mesin (M2M) dan sangat cocok untuk lingkungan dengan *bandwidth* rendah dan latensi tinggi. *Mosquitto* MQTT memungkinkan pengiriman

pesan secara *real-time* kepada klien dan dapat digunakan untuk membuat sistem notifikasi *push* yang efektif dan mudah dikelola. Dalam konteks Android, *Mosquitto* MQTT dapat diintegrasikan melalui layanan Android untuk menerima dan mengirim pesan, serta menyimpan data menggunakan *SQLite* yang memanfaatkan sumber daya komputasi yang minimal [13].

2.6 *Virtual Private Server* (VPS)

Virtual Private Server (VPS) adalah server virtual yang memberikan pengalaman seperti menggunakan server pribadi meskipun dioperasikan pada mesin fisik yang menjalankan beberapa sistem operasi. VPS menawarkan cara yang hemat biaya untuk menyediakan server serbaguna yang kuat dan fleksibel. Penggunaannya mencakup berbagai kebutuhan, mulai dari *rendering* video hingga *hosting* server *game* dan situs web. VPS digunakan oleh berbagai pengguna untuk memenuhi kebutuhan pemrosesan, komputasi, dan penyimpanan yang semakin meningkat, baik untuk keperluan pribadi maupun professional [14].

2.7 *Backend Development*

Pengembangan *backend* adalah bagian dari pengembangan aplikasi web yang berfokus pada pembuatan logika sisi server, arsitektur, dan pengelolaan interaksi basis data untuk mendukung permintaan dari *front-end*. *Backend* bertanggung jawab menjalankan logika bisnis, mengelola penyimpanan data, serta menangani autentikasi, otorisasi, dan integrasi dengan layanan eksternal. Proses ini menggunakan bahasa pemrograman seperti *Python, Java, atau JavaScript* (*Node.js*) dan kerangka kerja seperti *Django, Laravel,* atau *Spring Boot*, yang menyediakan alat untuk mempercepat pengembangan dan memastikan efisiensi serta keamanan sistem. Kerangka kerja *backend* yang dirancang dengan baik memungkinkan pengembang menciptakan aplikasi yang stabil, *scalable*, dan responsif. *Backend* juga harus mendukung skalabilitas sistem untuk menangani pertumbuhan pengguna dan data tanpa mengorbankan kinerja [15].

2.8 Node.js

Node.js adalah lingkungan *runtime JavaScript* yang beroperasi di sisi server, memungkinkan pengembang untuk membuat aplikasi web yang dapat diskalakan dan memiliki kinerja tinggi. Dengan menggunakan model I/O non-blokir yang digerakkan oleh peristiwa, *Node.js* sangat cocok untuk menangani banyak permintaan secara bersamaan. Ekosistem modul yang luas tersedia melalui NPM, memungkinkan pengembang untuk memperluas fungsionalitas *Node.js*. Dokumentasi resmi *Node.js* mencakup berbagai aspek, termasuk API, modul, dan praktik terbaik, yang berfungsi sebagai referensi komprehensif bagi pengembang yang bekerja dengan *Node.js* [16]. Logo Node Js dapat dilihat pada Gambar 2.1 Node Js.

Gambar 2. 1 Node Js

2.9 Nest.js

NestJS adalah *framework backend* yang dibangun menggunakan *TypeScript* dan *JavaScript*, yang dirancang untuk membangun aplikasi *server-side* yang efisien, dapat diskalakan, dan mudah dipelihara. *NestJS* menggabungkan prinsip-prinsip pemrograman berorientasi objek (*OOP*), pemrograman fungsional (*FP*), dan pemrograman reaktif fungsional (*FRP*), memungkinkan pengembang untuk memilih dan menggabungkan pendekatan yang sesuai dengan kebutuhan aplikasi mereka. Dengan struktur yang modular dan terorganisir, *NestJS* mendukung pembuatan aplikasi yang kompleks secara lebih terstruktur, memudahkan pengembangan dan pemeliharaan [17].

2.10 PostgreSQL

PostgreSQL adalah sistem manajemen basis data relasional yang mendukung perluasan tipe data dan fungsi melalui katalognya. Dengan fitur ini, *PostgreSQL* memungkinkan pemuatan kode atau objek yang telah dikompilasi secara dinamis selama *runtime,* sehingga pengguna dapat menambahkan tipe data dan fungsi kustom tanpa perlu memulai ulang server. Selain itu, *PostgreSQL* menawarkan berbagai jenis indeks, seperti *B-Tree*, GiST, dan SP-GiST, yang mendukung pengelolaan data dan berbagai metode pengindeksan. Fitur lain yang menonjol adalah dukungan untuk pemrosesan paralel kueri, yang merupakan inovasi untuk meningkatkan kinerja [18].

2.11 *Application Programming Interface* (API)

Application Programming Interface (API) adalah sebuah konsep yang memungkinkan aplikasi berinteraksi dengan aplikasi lain melalui fungsi antarmuka pemrograman. API memungkinkan akses dan pemanfaatan aplikasi oleh pihak ketiga tanpa perlu mengubah kode utama atau struktur basis data. Dengan API, sistem yang berbeda platform dapat berkomunikasi secara efisien. API juga berperan sebagai penghubung antara berbagai aplikasi lintas *platform*, sering kali dikenal sebagai *public* API. Salah satu bentuk API adalah *Web Service*, yang memungkinkan interaksi antara dua atau lebih aplikasi berbeda melalui jaringan, menggunakan arsitektur *Representational State Transfer* (ReST) dan beroperasi melalui protokol *Hypertext Transfer Protocol* (HTTP) [19].

2.12 RESTful API

RESTful API adalah antarmuka pemrograman aplikasi yang menerapkan arsitektur REST (*Representational State Transfer*) untuk membangun layanan yang dapat digunakan di berbagai platform dan lingkungan. Tujuan utama dari *RESTful API* adalah mendukung interoperabilitas dan penggunaan di *World Wide Web* (WWW). API ini terdiri dari *endpoint*, yaitu implementasi spesifik dari fungsionalitas proses

bisnis. Biasanya, *RESTful API* diakses melalui protokol HTTP (*Hypertext Transfer Protocol)* dengan memanfaatkan metode standar seperti *GET, POST, PUT*, dan *DELETE*. Untuk mengaksesnya, digunakan alamat yang disebut URI (*Uniform Resource Identifier*) [20].

2.13 GitHub

GitHub adalah *platform* yang mendukung pengelolaan dan pengembangan perangkat lunak secara kolaboratif. Proyek di GitHub dapat dengan mudah digandakan melalui fitur seperti *fork* atau dengan perintah *Git clone* dan *push*. Namun, kemudahan ini dapat menimbulkan tantangan dalam penelitian rekayasa perangkat lunak empiris karena adanya risiko bias data atau pelatihan model pembelajaran mesin yang tidak akurat akibat proyek yang diduplikasi.

Selain pengembangan perangkat lunak, GitHub juga digunakan untuk berbagai tujuan lain, seperti aktivitas pembelajaran dalam kursus dengan banyak peserta, eksperimen menggunakan sistem kontrol versi, serta aktivitas terkait lainnya. Banyaknya proyek yang disalin di GitHub juga dapat memengaruhi keakuratan analisis data dan pengembangan model pembelajaran mesin [21].

2.14 *Visual Studio Code*

Visual Studio Code adalah editor kode sumber terbuka yang populer dan kaya fitur, dirancang untuk pengembang dalam berbagai bahasa pemrograman. Dibuat oleh Microsoft, *VS Code* menawarkan kemampuan lintas *platform*, dapat digunakan di Windows, macOS, dan Linux. Beberapa fitur utama dari *VS Code* meliputi:

1. Antarmuka yang Mudah Digunakan: *VS Code* memiliki panel utama untuk pengeditan kodedan terminal bawaan untuk menjalankan perintah langsung dalam editor.
2. Ekstensi dan *Marketplace:* Tersedia ribuan ekstensi untuk menambahkan fitur seperti dukungan bahasa pemrograman, alat pengembang, *debugger*, dan tema tampilan.

3. Fitur Pintar: *VS Code* menyediakan fitur seperti *IntelliSense* (penyelesaian kode otomatis), pelacakan kesalahan, *refactoring*.
4. *Debugging*: Mendukung *debugging* bawaan untuk banyak bahasa dengan konfigurasi sederhana.
5. Sinkronisasi Pengaturan: Anda dapat menyinkronkan pengaturan, ekstensi, dan tema di berbagai perangkat menggunakan akun Microsoft atau GitHub.
6. Penggunaan untuk Berbagai Keperluan: Mulai dari pengembangan aplikasi web, pengolahan data dengan *Python*, hingga proyek berbasis kontainer *Docker* dan *Kubernetes* [22].

2.15 *Python*

Python adalah bahasa pemrograman tingkat tinggi yang dapat digunakan di berbagai *platform* melalui instalasi atau langsung diakses lewat web. *Python* telah berkembang menjadi bahasa yang sangat populer untuk pengembangan perangkat lunak dan pengolahan data. Fitur-fiturnya mencakup pustaka standar yang kaya, serta kemampuan untuk mengelola, menguji, dan membagikan kode secara efisien. Dengan *Python*, pengguna dapat dengan mudah menulis, menjalankan, dan menguji program, termasuk melakukan pengujian API menggunakan pustaka seperti *requests*, memverifikasi hasil keluaran, serta meninjau respons yang diterima. Bahasa ini sangat berguna pada analisis data, pembelajaran mesin, serta berbagai proyek yang melibatkan pengelolaan data digital dan otomasi proses [23].

2.18 Zenkit

Zenkit adalah platform manajemen proyek yang serbaguna, dirancang untuk membantu individu dan tim dalam mengelola berbagai jenis tugas dan data. Dengan fitur seperti *Kanban Board*, *Mind Map*, Kalender, dan *Gantt Chart*, *Zenkit* memungkinkan pengguna untuk menyesuaikan alur kerja mereka sesuai kebutuhan. Aplikasi ini digunakan untuk berbagai keperluan, mulai dari pengelolaan proyek hingga organisasi kegiatan pendidikan [24].

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Pelaksanaan penelitian ini dilakukan pada:

Waktu : Oktober 2024 – Maret 2025

Tempat : Laboratortium Teknik Telekomunikasi, Teknik Elektro, Universitas Lampung

Tahapan penelitian ini dapat dilihat pada Tabel 3.1 Tahapan Penelitian

Tabel 3. 1 Tahapan Penelitian

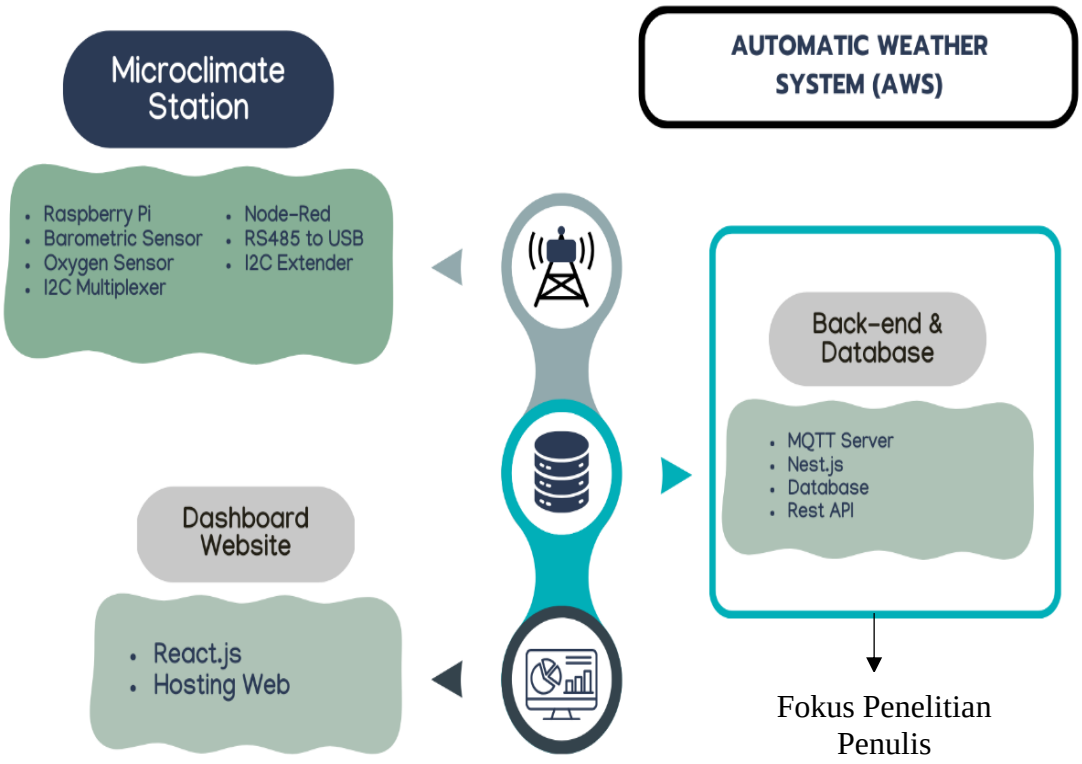
NO	Nama Kegiatan	Bulan 1	Bulan 2	Bulan 3	Bulan 4	Bulan 5	Bulan 6
1	Studi Literatur						
2	Research Gap						
3	Desain Sistem						
4	Implementasi						
5	Pengujian Dan Perbaikan						
6	Pembuatan Laporan						

3.2 *Capstone Project*

Penelitian ini merupakan hasil kerja sama tiga orang yang berkolaborasi dalam membangun sistem *Microclimate Station* (MCS) di Kawasan Hutan Mangrove

Petengoran, dengan pembagian tugas yang terstruktur dan saling melengkapi. Anggota tim pertama bertugas mengembangkan perangkat keras MCS, termasuk merancang dan merakit rangkaian sensor untuk memantau parameter lingkungan. Sementara itu, anggota tim kedua berfokus pada pengembangan *front-end*, yaitu antarmuka pengguna yang menampilkan data sensor secara interaktif dan *real-time* dalam bentuk dasbor pemantauan. Tampilan ini ditujukan agar pengguna dapat dengan mudah mengakses dan memahami kondisi mikroklimat.

Penulis sendiri mengambil peran dalam pengembangan *backend*, yang menjadi komponen inti dalam pengelolaan dan pemrosesan data dari sensor. Backend bertanggung jawab untuk menerima data dari perangkat keras melalui protokol MQTT, menyimpan data tersebut dalam basis data, serta menyediakan layanan API agar data dapat diakses oleh *frontend*. *Backend* juga memastikan komunikasi antar komponen berjalan stabil dan efisien. Diagram keseluruhan proyek, termasuk pembagian tugas, dapat dilihat pada Gambar 3.1 *Capstone Project.*



Gambar 3. 1 *Capstone Project*

3.3 Alat dan Bahan

Penelitian dan pengembangan *system Microclimate Station* (MCS) di kawasan hutan mangrove Petengoran didukung oleh daftar alat dan bahan berikut yang disesuaikan
dengan kebutuhan proyek berbasis sensor, perangkat keras, dan perangkat lunak.

3.3.1 Alat

Alat yang digunakan dalam pengerjaan penelitian dan pengembangan *system* yang dibuat adalah sebagai berikut (Tabel 3.2):

Tabel 3. 2 Alat berupa *hardware* dan *software* yang digunakan

No	Perangkat	Spesifikasi	Kegunaan	Keterangan
1	Laptop	*Processor* IntelI CoreI i5 2430M, RAM 8GB, SSD 256GB, Sistem Operasi Windows	Perangkat pembuatan pengujian *system.*	Perangkat keras yang digunakan sebagai *compiler* dalam pemrograman
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 *system* yang akan dibuat.	Diakses melalui *Secure Shell (SSH)*
	MQTT Explorer	*MQTT Explorer*	*Software* yang digunakan untuk pengujian	Terpasang di Laptop

No	Perangkat	*Spesifikasi*	Kegunaan	Keterangan
4	*Visual Studio Code*	VS Code versi 1.75.0	*Software* yang digunakan dalam penulisan kode program untuk pengambilan dan penyimpanan data dari sensor	Terpasang di Laptop
5	PostgreSQL	PostgreSQL *database* versi 9.2.24	Program *database* yang digunakan dalam penyimpanan data dari sensor	Terpasang di VPS
6	Postman	Postman V.10.19	Program yang digunakan untuk mencoba Endpoint API	Terpasang di Laptop
7	Node.js & NPM	Node.js versi v16.19.0. NPM versi 8.19.3	Aplikasi yang digunakan untuk menjalankan kode program untuk penyimpanan data	Terpasang di VPS

No	Perangkat	Spesifikasi	Fungsi	Keterangan
8	Niaga Hoster	*Website Hosting*	Aplikasi yang digunakan untuk melakukan publikasi *website*	Terpasang di *provider*
9	*Node RED*	*Node RED Online Version*	Aplikasi yang digunakan untuk menghubungkan perangkat ke *backend* via *publisher* dengan MQTT	Terpasang di raspberry pi
10	Github	Membuat *Repository* kode dan kolaborasi Git.	Kolaborasi dan manajemen proyek kode.	*Platform* untuk *hosting* dan kolaborasi kode.

3.3.2 Bahan

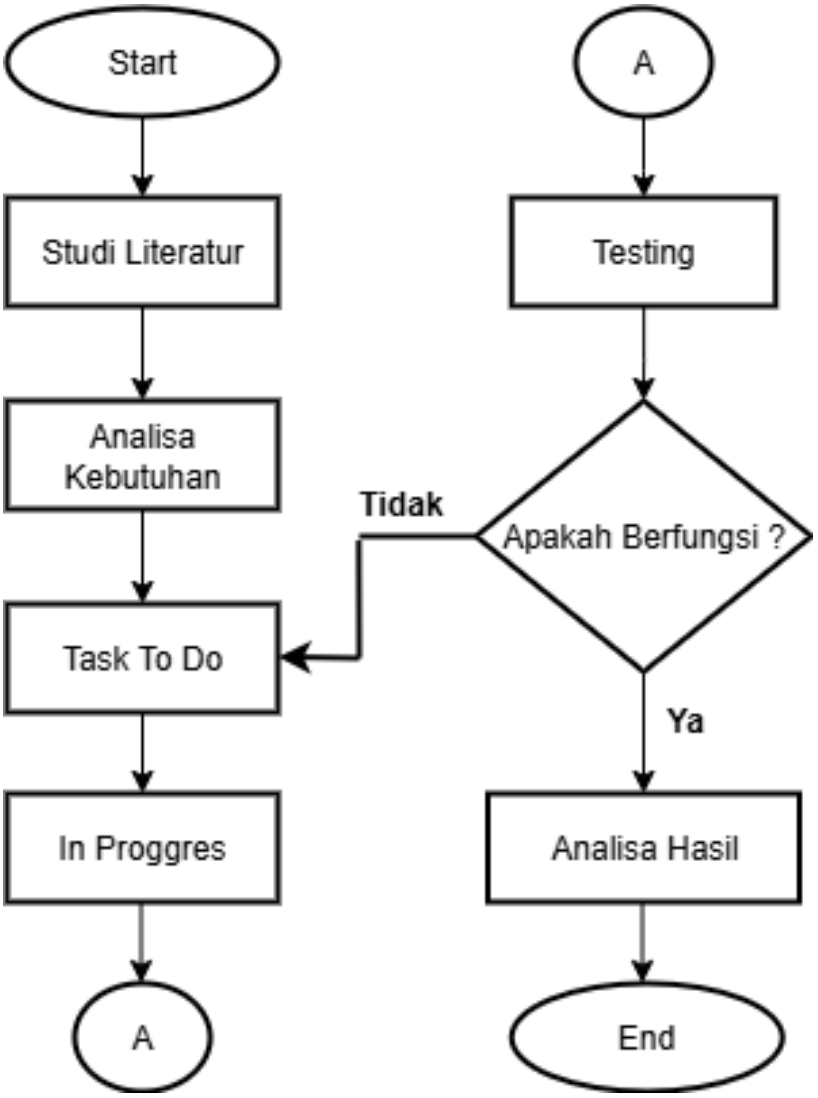
Bahan utama yang digunakan dalam pelaksanaan penelitian ini adalah data yang diperoleh langsung dari berbagai sensor yang telah dipasang di lokasi penelitian. Data yang terkumpul melalui sistem pemantauan ini mencakup berbagai parameter penting yang berkaitan dengan kondisi mikroklimat di Kawasan Hutan Mangrove Petengoran, seperti suhu udara, kelembapan, kecepatan angin, arah angin, serta curah hujan. Setiap sensor bekerja untuk mengukur parameter-parameter tersebut secara *real-time*, menghasilkan informasi yang sangat berguna untuk melakukan analisis lebih lanjut tentang perubahan kondisi lingkungan di kawasan tersebut.. Bahan yang digunakan dapat dilihat pada Tabel 3.2 Data Sensor.

Tabel 3. 3 Data Sensor

No	Komponen dan Perangkat Lunak	Kegunaan
1	Sensor *Rain Gauge*	Sebagai pengukur curah hujan
2	Sensor *Anemometer*	Sebagai pengukur kecepatan angin
3	Sensor *Wind Direction*	Sebagai pengukur arah angin
4	SHT31	Sebagai pengukur suhu, kelembaban dan tekanan udara
5	Sensor Oksigen	Sebagai pengukur kadar oksigen
6	Sensor DS18B20	Sebagai pengukur suhu air
7	Sensor BMP390	Sebagai Pengukur tekanan udara
8	*Pyranometer*	Sebagai pengukur intensitas radiasi matahari

3.4 Tahapan Penelitian

Penelitian ini diawali dengan tahap perencanaan dan analisis kebutuhan untuk menentukan spesifikasi serta fitur utama yang diperlukan dalam pengembangan sistem. Tahap ini bertujuan untuk memastikan sistem dirancang sesuai dengan kebutuhan pengguna dan tujuan penelitian. Proses pengembangan dilakukan menggunakan metode Kanban, yang berfokus pada visualisasi alur kerja menggunakan papan Kanban, pembatasan jumlah tugas yang berjalan (*work in progress*), dan pengelolaan prioritas secara efektif guna menjaga efisiensi serta kelancaran proses. Metode ini juga mendukung fleksibilitas dalam menyelesaikan tugas-tugas yang ada. Setelah sistem selesai dikembangkan, dilakukan tahap analisis hasil untuk memastikan bahwa sistem telah memenuhi seluruh kebutuhan, spesifikasi, dan tujuan yang ditetapkan sebelumnya. Diagram alur dari keseluruhan tahapan penelitian ini dapat dilihat pada Gambar 3.1 Tahapan Penelitian.



Gambar 3. 2 Tahapan Penelitian

3.5 Studi Literatur dan Analisa Kebutuhan

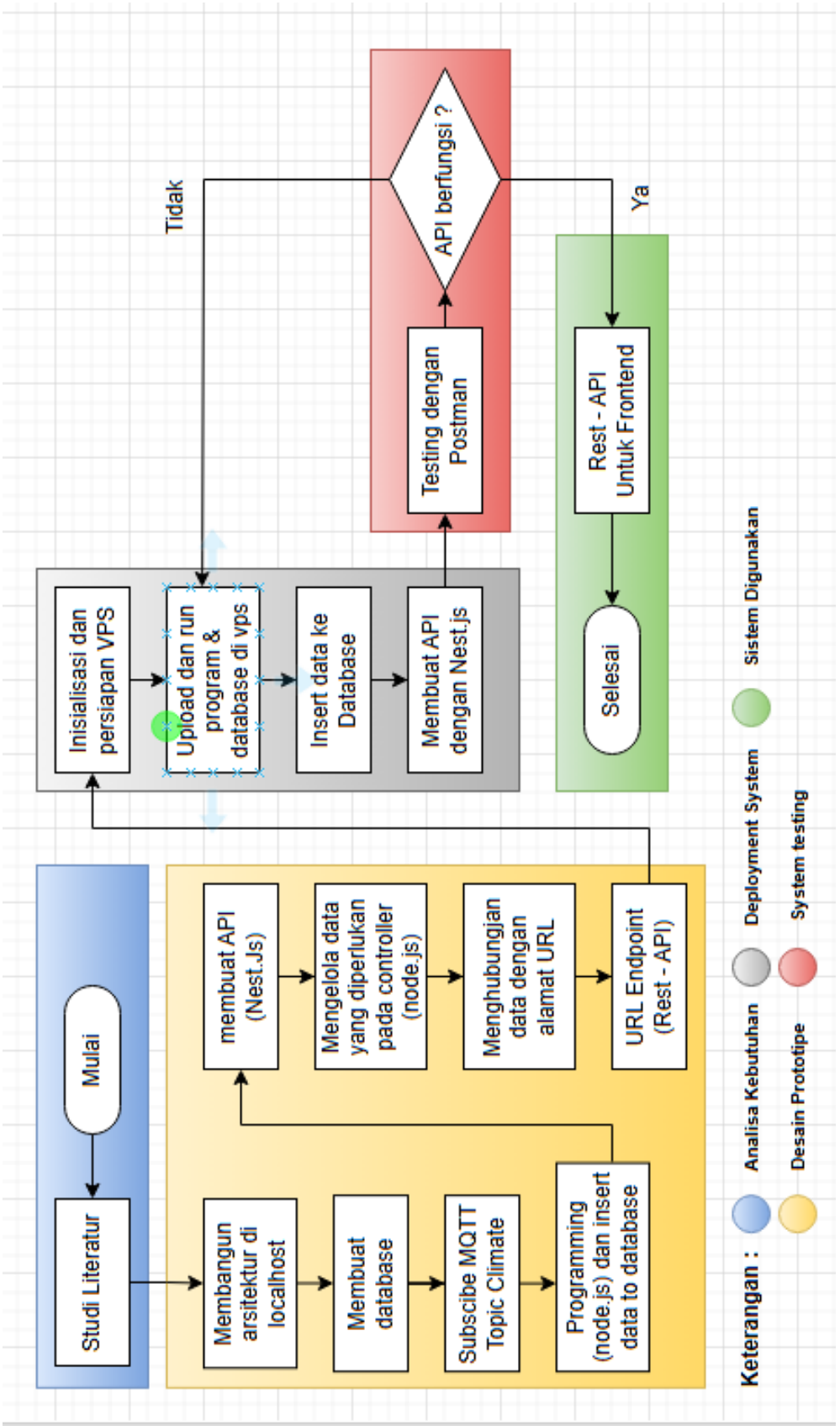
Studi literatur merupakan bagian penting dalam penelitian ini untuk memahami konsep-konsep dasar yang relevan dengan topik yang dibahas, serta untuk mengidentifikasi penelitian terdahulu yang memiliki kaitan langsung. Dalam konteks sistem pemantauan cuaca otomatis, berbagai teknologi, perangkat keras, dan perangkat lunak telah dikaji untuk mengetahui kekuatan dan kelemahan setiap metode yang digunakan. Berdasarkan studi literatur ini, penulis dapat merumuskan kebutuhan dan strategi pengembangan untuk sistem yang lebih baik.

Tahapan analisis kebutuhan dilakukan setelah mempelajari hasil studi literatur yang relevan. Pada tahap ini, penulis mengidentifikasi masalah dan kekurangan yang ada pada sistem yang sudah ada. Analisis kebutuhan untuk pengembangan backend sistem pemantauan cuaca otomatis fokus pada identifikasi komponen-komponen teknis yang diperlukan untuk mendukung pengolahan dan pengiriman data secara efisien. Tahapan ini mencakup penentuan fitur *backend* yang meliputi kemampuan untuk menangani data sensor dalam jumlah besar, sistem komunikasi seperti MQTT untuk pengiriman data *real-time*, serta penyimpanan dan pengolahan data menggunakan basis data yang tepat. *Backend* juga harus dirancang agar dapat memberikan API yang efisien untuk mengakses data cuaca, serta memungkinkan pengelolaan dan analisis data lebih lanjut.

Berdasarkan Studi Literatur dan Hasil Analisa berikut adalah beberapa hal yang dibutuhkan pada penelitian ini:

1. *Virtual Private Server* (VPS) digunakan sebagai tempat untuk mengelola MQTT *Broker* sekaligus sebagai penyimpanan data.
2. MQTT *Broker* berfungsi sebagai system yang memungkinkan pengiriman data dari sensor pemantauan ke *dashboard* serta ke *database* untuk pengolahan lebih lanjut.
3. Data yang diperoleh dari sensor harus disimpan secara permanen dalam database yang ada di VPS agar tidak hilang. Hal ini penting untuk memastikan bahwa data tersebut bisa digunakan untuk menampilkan informasi di *dashboard* serta mendukung penelitian.
4. Data yang tersimpan di *database* dapat diteruskan ke *dashboard* atau antarmuka pengguna menggunakan API, sehingga memudahkan akses dan visualisasi data secara *real-time*.

Berdasarkan kebutuhan yang telah diidentifikasi sebelumnya, maka dirumuskan sejumlah tugas yang harus dilaksanakan dalam proses pengembangan sistem secara menyeluruh. Rincian dari masing-masing tugas tersebut disajikan pada Gambar 3.4, yang menggambarkan tahapan serta alur kerja yang akan dilalui selama proses pengembangan berlangsung.



gambar 3. 3 Tahapan *Backend*

Proses pengembangan sistem dimulai dengan melaksanakan studi literatur yang bertujuan untuk memperoleh pemahaman yang komprehensif mengenai teori dasar serta praktik terbaik yang relevan dengan bidang yang diteliti. Kegiatan ini merupakan tahap awal yang sangat penting untuk membentuk landasan kuat agar seluruh proses pengembangan dapat berjalan dengan lancar dan tepat sasaran. Studi literatur dilakukan dengan mengumpulkan referensi dari berbagai sumber terpercaya, seperti jurnal ilmiah, buku teknis, dan artikel terkait, untuk memastikan pengetahuan yang diperoleh cukup mendalam. Setelah penguasaan materi dasar tercapai dan pemahaman teori sudah cukup kuat, langkah berikutnya adalah menginisialisasi dan merancang arsitektur awal pada lingkungan *localhost*. Pada tahap ini, basis data dibuat menggunakan *PostgreSQL* karena kemampuannya dalam mengelola data secara efisien dan skalabel. Konfigurasi lingkungan pengembangan juga diatur secara cermat, termasuk instalasi perangkat lunak pendukung dan penyesuaian parameter teknis, agar siap digunakan secara optimal. Tahap awal ini sangat menentukan karena memastikan bahwa fondasi teknis dan teoretis yang dibangun mendukung kelancaran proses pengembangan berikutnya dan mengurangi risiko kesalahan.

Infrastruktur dasar telah dibentuk dengan baik, sehingga proses pengembangan dilanjutkan dengan membuat program menggunakan *Node.js* yang berfungsi untuk memasukkan data secara terstruktur ke dalam basis data. *Node.js* dipilih karena kemampuannya menangani operasi *input* dan *output* data dengan cepat dan efisien, mendukung pembuatan aplikasi yang responsif. Pengembangan program ini dirancang untuk memastikan bahwa data yang dimasukkan dapat dikelola dengan baik. Secara bersamaan, sistem disiapkan untuk melakukan *subscribe* terhadap topik *MQTT* yang berhubungan dengan aspek *Climate*. Fitur ini memungkinkan sistem menerima dan mengelola data yang dikirimkan melalui topik-topik tersebut secara *real-time* dari sumber seperti sensor atau perangkat IoT. Proses ini menjaga akurasi dan kelancaran pengambilan data, memastikan bahwa informasi yang diterima selalu mutakhir dan dapat diandalkan untuk analisis lebih lanjut. Integrasi *MQTT* juga memungkinkan sistem untuk merespons perubahan data dengan cepat, mendukung operasional yang dinamis.

Tahap perancangan arsitektur pada lingkungan lokal selesai dilaksanakan sesuai rencana, kemudian proses dilanjutkan dengan persiapan serta inisialisasi pada *Virtual Private Server* (VPS) sebagai lingkungan produksi. Pada tahap ini, seluruh perangkat lunak dan basis data yang diperlukan diunggah serta dikonfigurasikan agar dapat berjalan optimal di VPS. Selanjutnya, dilakukan pemindahan dan penyimpanan data ke dalam basis data yang tersedia pada server tersebut, sehingga sistem dapat beroperasi dengan baik di lingkungan produksi.

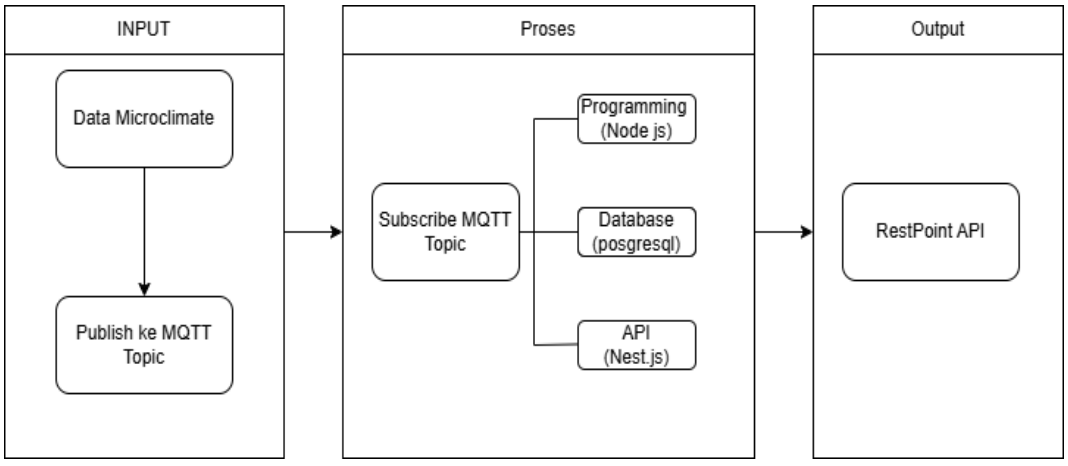
Seluruh komponen sistem diuji awal menggunakan data aktual dari sumber data *Climate* pada lingkungan lokal untuk memastikan fungsi sistem berjalan sesuai harapan. Pengujian ini bertujuan mengevaluasi kinerja dan kestabilan sistem secara menyeluruh serta mengidentifikasi potensi masalah yang mungkin muncul selama operasi. Apabila ditemukan ketidaksesuaian atau kegagalan pada API, perlu dilakukan evaluasi menyeluruh terhadap tahapan sebelumnya guna menentukan penyebab dan langkah perbaikan agar masalah dapat segera diatasi.

Pengujian awal berhasil dilewati dengan hasil yang baik dan sistem dinyatakan stabil, sehingga tahap pengujian akhir dilaksanakan menggunakan aplikasi *Postman*. Pengujian ini dilakukan untuk memastikan keandalan serta performa API secara komprehensif, termasuk uji responsivitas dan ketahanan terhadap beban kerja, sehingga dapat dipastikan bahwa layanan yang disediakan telah memenuhi spesifikasi dan kebutuhan fungsional yang telah dirancang.

Setelah API berhasil diverifikasi dan memenuhi standar kualitas yang ditetapkan, langkah berikutnya adalah pengembangan URL *endpoint* menggunakan *REST API* yang akan diakses oleh aplikasi frontend maupun mobile guna mendukung fungsi aplikasi secara penuh. Seluruh pengujian dinyatakan sukses serta sistem beroperasi dengan stabil tanpa kendala berarti, sehingga proses pengembangan dapat dianggap selesai dan sistem siap untuk diimplementasikan dalam lingkungan *frontend.*

3.6 Pengembangan Sistem

Pengembangan sistem ini penulis lakukan secara bertahap dan terstruktur dengan menggunakan metode Kanban. Langkah awal dimulai dengan perencanaan yang matang, kemudian saya membagi pekerjaan menjadi tugas-tugas kecil yang lebih jelas dan terukur. Setiap tugas tersebut saya susun dan tampilkan pada papan Kanban yang terdiri dari tiga kolom utama, yaitu *To Do*, *In Progress*, dan *Done*, untuk memudahkan visualisasi alur kerja dan memantau perkembangan setiap bagian dari proses. Metode Kanban saya gunakan agar proses pengerjaan menjadi lebih tertata dan berurutan. Dengan membatasi jumlah pekerjaan yang sedang dikerjakan (*Work in Progress/WIP*), penulis dapat lebih fokus menyelesaikan satu tugas hingga tuntas sebelum melanjutkan ke tugas berikutnya. Pendekatan ini membantu menghindari multitasking yang seringkali justru menurunkan efisiensi kerja, serta mengurangi potensi terjadinya kesalahan atau pekerjaan yang tertunda. Setiap kali sebuah tugas selesai, penulis melakukan proses pengujian untuk memastikan bahwa fungsionalitas sistem berjalan dengan baik dan sesuai dengan yang diharapkan. Jika terdapat kekurangan atau potensi perbaikan, saya melakukan iterasi untuk menyempurnakan hasilnya, baik dari segi desain, implementasi, maupun performa. Secara keseluruhan, penggunaan metode Kanban terbukti sangat membantu penulis dalam menjaga alur kerja tetap lancar, fokus, dan efisien sepanjang proses pengembangan sistem. Untuk lebih memahami tahapan pengembangan sistem ini dapat dilihat pada Gambar 3.4 *Input* dan *Output Backend*.



Gambar 3. 4 *Input* dan *output backend*

Pada tahap input, sistem menerima data dari *Microclimate Station* secara *real-time*, di mana data tersebut dipublish ke topik MQTT yang telah ditentukan. MQTT, sebagai protokol komunikasi ringan yang sangat efisien dan cepat, memungkinkan data untuk dikirimkan dengan latensi rendah dan konsumsi *bandwidth* yang minimal. Hal ini sangat penting dalam sistem pemantauan lingkungan seperti MCS, di mana data yang terkumpul dari berbagai sensor harus dapat diproses dan disebarkan secara cepat untuk analisis lebih lanjut. Setelah data *dipublish* ke topik MQTT, sistem *backend* yang dibangun dengan Node.js akan s*ubscribe* ke topik tersebut untuk mengambil data yang dikirimkan. Dengan mekanisme ini, data dapat langsung diterima dan siap untuk diproses lebih lanjut tanpa adanya *delay* yang signifikan.

Setelah data diterima oleh *backend*, Node.js akan memproses informasi yang dikirimkan dari sensor dan kemudian menyimpannya dalam *database PostgreSQL*. *PostgreSQL* dipilih karena kemampuannya dalam menangani transaksi data besar dan tingkat keamanan yang tinggi, sehingga memastikan integritas dan keandalan data yang dikumpulkan selama pemantauan. Selain itu, PostgreSQL menawarkan fitur yang mendukung analisis data lebih lanjut, yang penting untuk evaluasi kondisi mikroklimat di kawasan hutan mangrove. Data yang telah tersimpan dengan aman ini selanjutnya akan dipersiapkan untuk diakses melalui API yang dibangun menggunakan Nests.js, sebuah *framework* yang populer dan efisien dalam ekosistem Node.js.

Dengan menggunakan Nest.js, sistem backend membangun Rest-API yang menyediakan *endpoint* yang dapat diakses oleh *frontend*. *Endpoint* ini berfungsi sebagai jembatan antara database dan aplikasi pengguna, memungkinkan aplikasi untuk menarik data yang dibutuhkan dari sistem secara langsung. *Frontend* dapat menggunakan *endpoint* ini untuk menampilkan informasi terbaru yang dikumpulkan dari sensor, seperti grafik suhu, kelembapan, atau kecepatan angin. Dengan cara ini, data yang diproses dan disimpan dengan aman di *backend* dapat dengan mudah diintegrasikan ke dalam pengalaman pengguna yang interaktif, memungkinkan pengguna untuk memantau kondisi *microclimate* secara *real-time* melalui perangkat mereka.

3.6.1 Tahap *To Do*

Tahapan *To Do* pada Kanban menjadi elemen utama dalam mengorganisasi tugas-tugas yang belum dikerjakan. Kolom ini sebagai tempat awal untuk mencatat pekerjaan yang perlu diselesaikan. Tugas-tugas tersebut diurutkan berdasarkan prioritas agar memudahkan identifikasi langkah berikutnya dalam proses penelitian. Dengan visualisasi ini, dapat mempermudah pengelolaan beban kerja secara efektif, memastikan setiap tahapan penelitian berpindah secara sistematis ke kolom *In Progress*, dan menjaga alur penelitian tetap terstruktur hingga selesai.

3.6.2 Tahap *In Progress*

Tahapan *In Progress* berfungsi untuk mencatat tugas-tugas yang sedang dikerjakan. Setelah tugas dipindahkan dari kolom *To Do*, tugas tersebut ditempatkan di kolom ini untuk menandai bahwa proses penyelesaian telah dimulai. Tahapan ini membantu menjaga fokus pada satu atau beberapa tugas sesuai kapasitas yang ada, sehingga kualitas pengerjaan dapat terjaga dan *multitasking* berlebihan dapat dihindari. Kolom ini juga memungkinkan pemantauan perkembangan pekerjaan secara *real-time*, sehingga hambatan atau kebutuhan tambahan dapat diidentifikasi dan diatasi dengan cepat. Dengan pengelolaan yang baik di tahap ini, alur kerja penelitian tetap terorganisasi dan efisien.

3.6.3 Tahap *Testing*

Tahapan *Testing* dilakukan proses pengujian yang akan dilakukan terhadap API yang telah dikembangkan. Pengujian ini bertujuan untuk memastikan kualitas dan fungsionalitas API sebelum diluncurkan ke lingkungan produksi.
Rencana Pengujian:

1. Fungsionalitas: Setiap *endpoint* utama (*GET*) akan diuji untuk memastikan bahwa API berfungsi sesuai dengan yang diharapkan.

2. Kinerja: Waktu *respons* API akan diukur untuk memastikan bahwa permintaan diproses dalam waktu yang wajar.

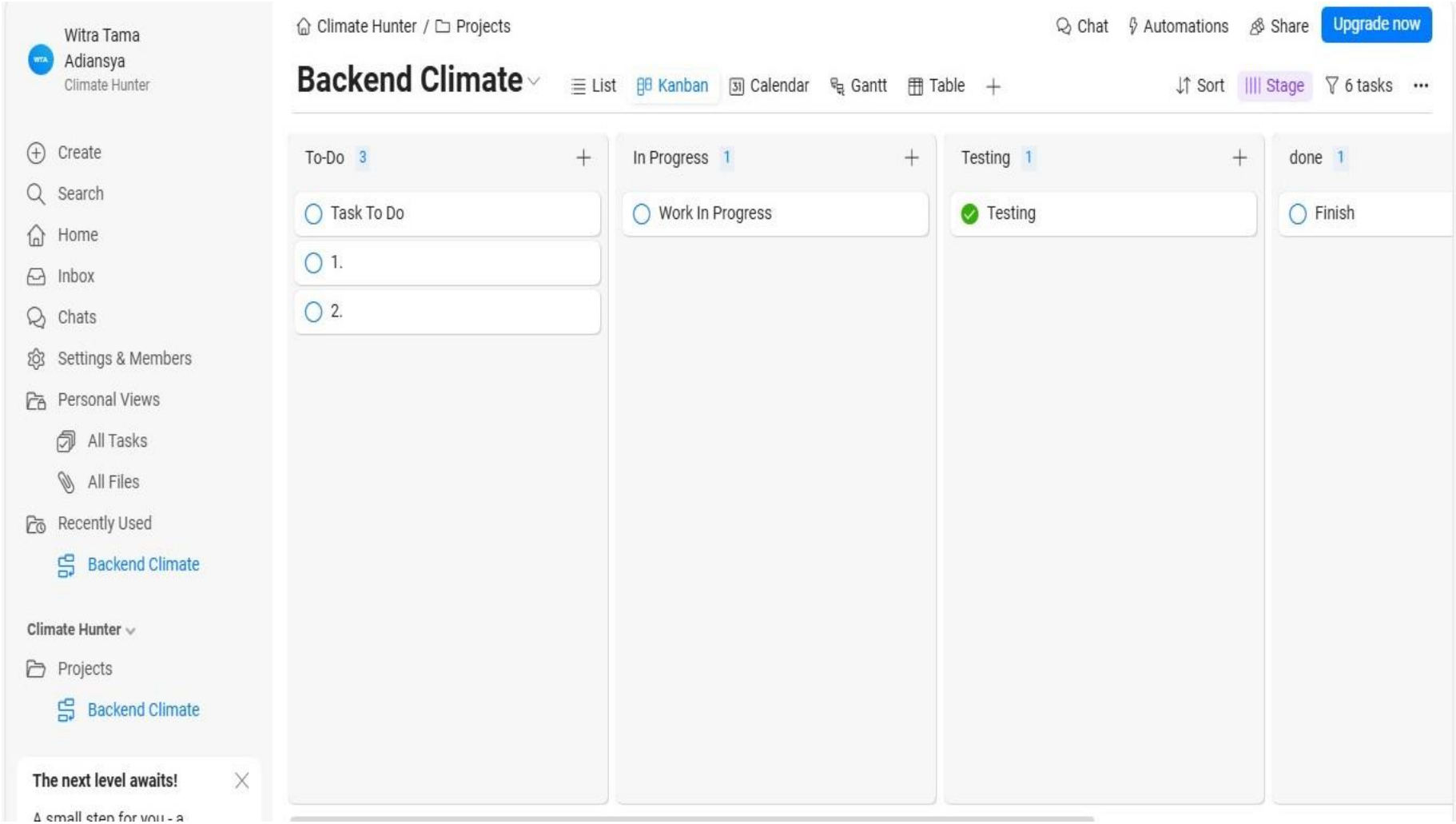
Rencana pengujian ini memastikan bahwa API yang dikembangkan memenuhi standar sebelum diluncurkan ke tahap berikutnya.

3.6.4 Tahap *Done*

Tahapan *Done*, tugas yang telah melewati semua tahapan sebelumnya dan telah berfungsi dengan baik sesuai dengan spesifikasinya akan ditempatkan di bagian ini. Ketika tugas sudah berada di tahap ini, maka tugas tersebut dianggap selesai dan siap untuk digunakan. Dalam penelitian ini, *platform* yang digunakan adalah Zenkit untuk memvisualisasikan tugas-tugas yang berlangsung selama proses pengembangan *system*.

Zenkit adalah *platform* manajemen proyek yang menyediakan berbagai fitur lengkap untuk mendukung kolaborasi dan pengelolaan tugas dalam sebuah tim. Dengan tampilan yang intuitif dan mudah digunakan, Zenkit memungkinkan pengguna untuk memvisualisasikan tugas, menetapkan prioritas, dan melacak progres proyek secara *real-time. Platform* ini menawarkan berbagai tampilan, seperti papan Kanban, daftar tugas, kalender, dan tampilan , yang dapat disesuaikan sesuai dengan kebutuhan tim.

Zenkit mendukung integrasi dengan berbagai aplikasi lain, seperti Google Drive, Slack, dan Zapier, untuk meningkatkan produktivitas dan efisiensi kerja. Zenkit juga menyediakan fitur manajemen *database* yang memungkinkan pengguna untuk menyimpan, mengelola, dan mengakses informasi dengan mudah. Fitur kolaborasi tim seperti komentar, pengingat, dan pemberitahuan memungkinkan komunikasi yang lebih baik antar anggota tim. Dengan fleksibilitas yang tinggi, Zenkit cocok digunakan untuk berbagai jenis proyek, termasuk pengembangan sistem seperti yang dilakukan dalam penelitian ini. Tampilan *platform* zenkit dapat dilihat pada Gambar 3.5 *Platform* Zenkit.



Gambar 3. 5 *Platform* Zenkit

3.7 Tahapan Pengujian

Pengujian dilakukan untuk memastikan bahwa REST API yang dikembangkan benar-benar mampu berfungsi sebagaimana mestinya serta memberikan performa yang optimal dalam proses pengambilan dan penyajian data dari *microclimate station.* Melalui pengujian ini, dapat diketahui sejauh mana sistem *backend* dapat diandalkan dalam mendukung kebutuhan monitoring data cuaca secara *real-time*. Secara garis besar, terdapat dua jenis pengujian utama yang dilakukan, yaitu:

1. **Pengujian Waktu respon API**

   Pengujian ini bertujuan untuk mengevaluasi kecepatan layanan API dalam merespons setiap permintaan (request) yang dikirimkan oleh klien. Proses pengujian dilakukan dengan menggunakan *script Python* yang memanfaatkan pustaka *requests.* Melalui script ini, sejumlah besar permintaan 1000 kali dikirimkan secara berulang ke *server,* kemudian dicatat waktu respons dari setiap permintaan. Hasil pengujian ini menjadi dasar penilaian apakah API dapat memberikan respons dalam rentang waktu yang wajar, stabil, serta sesuai standar kinerja yang diharapkan untuk sebuah sistem monitoring cuaca berbasis IoT yang menuntut kecepatan akses data secara konsisten.

2. **Pengujian *Query Database***

   Pengujian ini berfokus pada sisi efisiensi eksekusi kueri dalam basis data *PostgreSQL* yang digunakan sebagai penyimpanan data sensor. Instrumen utama yang digunakan adalah perintah *EXPLAIN ANALYZE*, yang mampu menampilkan detail proses eksekusi kueri secara lengkap. Analisis difokuskan pada dua parameter penting, yaitu *planning time* (waktu yang diperlukan sistem untuk merencanakan strategi eksekusi kueri) dan *execution time* (waktu yang digunakan sistem dalam menjalankan kueri). Melalui pengujian ini dapat diketahui seberapa optimal struktur basis data dan kueri yang diterapkan dalam mendukung performa *REST API.*

3.8 Indikator Keberhasilan

Indikator keberhasilan merupakan seperangkat ukuran, parameter, atau kriteria terukur yang digunakan untuk menilai tingkat ketercapaian tujuan, program, ataupun kegiatan penelitian. Indikator ini berfungsi sebagai pedoman objektif untuk memastikan bahwa seluruh proses yang direncanakan dapat berjalan sesuai dengan standar yang telah ditetapkan. Dalam konteks penelitian sistem, indikator keberhasilan tidak hanya berfokus pada keluaran akhir, tetapi juga mencakup penilaian terhadap kualitas proses pengembangan, kesesuaian implementasi, serta efektivitas fungsi yang dihasilkan.

Melalui indikator keberhasilan, peneliti dapat melakukan evaluasi secara sistematis mengenai apakah sistem mampu memenuhi kebutuhan pengguna, apakah performanya berada dalam batas yang dapat diterima, serta apakah arsitektur sistem bekerja secara konsisten dan stabil dalam berbagai kondisi. Evaluasi ini dapat meliputi aspek fungsionalitas, kecepatan *respons REST API*, efisiensi dalam pengelolaan basis data, akurasi data yang disimpan dan ditampilkan, serta tingkat keandalan sistem dalam menangani data secara *real time*. Dengan demikian, indikator keberhasilan menjadi elemen penting untuk menilai mutu keseluruhan dari penelitian dan memastikan bahwa hasil yang dicapai benar-benar sesuai dengan tujuan yang telah dirumuskan. Indikator keberhasilan yang digunakan dalam penelitian ini dirangkum pada Tabel 3.3 Indikator Keberhasilan.

Tabel 3. 4 Indikator Keberhasilan

No.	Kriteria Penilaian	Indikator Kinerja	Hasil Aktual (ceklis)
1	MQTT	Berhasil menerima data sensor yang dikirmkan melalui mqtt (berhasil *Subscribe*)	
2	Kinerja Sistem Penyimpanan Data	Berhasil menyimpan data dari mqtt ke *database* (PostgreSQL)	
3	Kinerja API	API berhasil merespons permintaan tanpa *error* (status 200 OK).	

BAB V KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa kesimpulan yang didapat sebagai berikut :

1. *Backend Microclimate* berhasil mengintegrasikan komunikasi berbasis protokol MQTT, serta mengamankan layanan API dengan koneksi HTTPS yang dilindungi TLS.
2. Penyimpanan data sensor pada sistem *Microclimate* menggunakan basis data PostgreSQL terstruktur yang mampu menjamin ketersediaan data. Hasil analisis performa basis data menunjukkan efisiensi kueri yang tinggi, dengan waktu perencanaan dan eksekusi untuk pemindaian indeks pada tabel Station (0,236 ms dan 0,683 ms), First_station (0,354 ms dan 0,375 ms), serta Second_station (0,362 ms dan 0,389 ms), mendukung akses data yang cepat dan stabil.
3. Pengembangan REST-API pada sistem *Microclimate* telah menghasilkan *endpoint* yang andal untuk mengakses data sensor dari basis data dan mendukung kebutuhan visualisasi pada *dashboard* pemantauan. Uji performa API menunjukkan waktu *respons* yang sangat baik pada *endpoint* latest/first-station (2054 ms) dan latest/second-station (844 ms) yang berada jauh di bawah ambang batas 1000 ms. *Endpoint* dengan kompleksitas lebih tinggi, seperti daily/first-station (1985 ms) dan daily/second-station (1832 ms), serta weekly/first-station (3795 ms) dan weekly/second-station (3471 ms) memiliki waktu respons lebih lama. Sementara itu, endpoint monthly/first-station (4366 ms) dan monthly/second-station (3950 ms) menunjukkan waktu respons paling tinggi karena jumlah data yang besar, sedangkan *endpoint* dengan *resampling* seperti resample-15min (1610–

1735 ms) dan resample-30min (1387–1636 ms) berada pada kategori sedang, lebih lambat dibandingkan *endpoint latest*, namun lebih cepat dibandingkan *daily*, *weekly*, maupun *monthly*.

5.2 Saran

Berdasarkan hasil penelitian dan evaluasi yang telah dilakukan, terdapat saran untuk penelitian selanjutnya, antara lain:

1. Optimalisasi Kueri Basis Data: Meningkatkan performa kueri pada basis data PostgreSQL untuk mempercepat akses data pada tabel penyimpanan data *microclimate.*
2. Integrasi *Machine Learning* pada *Backend:* Menambahkan fungsi *machine learning* pada *backend* dengan model tertentu untuk memprediksi kondisi *microclimate* berdasarkan data sensor yang tersimpan di PostgreSQl.

DAFTAR PUSTAKA

[1] G. O. Ramena, C. E. V. Wuisang, dan F. O. P. Siregar, “Pengaruh Aktivitas Masyarakat Terhadap Ekosistem Mangrove di Kecamatan Mananggu,” *Spasial*, vol. 7, no. 3, pp. 343–351, Jul. 2020, doi: 10.35793/sp.v7i3.32124.

[2] S. C. DeVoe et al., "Mangrove Microclimates Alter Seedling Dynamics At The Range Edge," *Ecology*, vol. 98, no. 10, pp. 2607–2618, Oct. 2017, doi: 10.1002/ecy.1979.

[3] M. U. H. Al Rasyid, A. Basofi, S. Sukaridhoto, dan Y. A. Nugraha, “Implementation Of Iot Platform Analytics For Monitoring Coastal Water Conditions,” in *Proc. Int. Conf. Appl. Sci. Technol. Eng. Sci. (iCAST-ES)*, Dordrecht, The Netherlands: Atlantis Press, Jul. 2024, vol. 230, pp. 587–606, doi: 10.2991/978-94-6463-364-1_55.

[4] T. S. Gbadebo,“Optim,Izing Full-Stack Development For Fintech Applications: Balancing User Experience And Backend Performance In High-Stakes Financial Environments,” Sikkim Manipal Univ., Majitar, India, Tech. Rep., Oct. 2024. [Online]. Available: https://www.researchgate.net/publication/384977185

[5] K. M. Ibrahim, R. A. Rashid, A. H. F. A. Hamid, M. A. Sarijari, dan M. A. Baharudin, “Lightweight Iot Middleware For Rapid Application Development,” *TELKOMNIKA Telecommun. Comput. Electron. Control.*, vol. 17, no. 3, pp. 1385–1392, Jun. 2019, doi: 10.12928/TELKOMNIKA. v17i3.11793.

[6] K. Silanon dan K. Kaewwongsri, “Design And Implement Of A Weather Monitoring Station Using Coap On NB-Iot Network,” *in Proc. 17th Int. Conf. Electr. Eng./Electron., Comput., Telecommun. Inf. Technol*. (ECTI-CON), Phuket, Thailand, pp. 230–233, Jun. 2020, doi: 10.1109/ECTI-CON49241.2020.9158290.

[7] A. R. Rumee, “Iot System For Remote Monitoring Of The Mangrove Forests Of Sundarbans,” *J. Comput. Sci. Inst.*, vol. 20, pp. 254–258, 2021, doi: 10.35784/jcsi.2703.

[8] S. J. Saidi, S. Matic, O. Gasser, G. Smaragdakis, dan A. Feldmann, “Deep Dive Into The Iot Backend Ecosystem,” in Proc. 22nd ACM Int. Meas. Conf. (IMC), Nice, France, pp 488–503, Oct. 2022, doi:10.1145/3517745.356143

[9] A. K. Alnaim dan A. M. Alwakeel, “Machine-Learning-Based Iot–Edge Computing Healthcare Solutions,” *Electronics*, vol. 12, no. 4, Feb. 2023, doi: 10.3390/electronics12041027.

[10] T.-H. Chen, M.-H. Lee, I.-W. Hsia, C.-H. Hsu, M.-H. Yao, and F.-J. Chang, “Develop A Smart Microclimate Control System For Greenhouses Through System Dynamics And Machine Learning Techniques,” *Water*, vol. 14, no. 23, p. 3941, Dec. 2022, doi: 10.3390/w14233941.

[11] R. Oppliger, *SSL and TLS: Theory and Practice*, 3rd ed., Norwood, MA: Artech House, 2023.

[12] C. M. Gallardo Paredes, P. D. R. Rodríguez Fiallos, dan F. J. Galora Silva, “Android Service To Interface Mosquitto Messaging Broker (MQTT),” *Rev. ODIGOS*, vol. 3, no. 1, pp. 9–24, Feb. 2022, doi: https://doi.org/10.35290/ro.v3n1.2022.539

[13] C. M. Gallardo Paredes, P. del R. Rodríguez Fiallos, dan F. J. Galora Silva, “Android Service To Interface Mosquitto Messaging Broker (MQTT),” *Rev. ODIGOS*, vol. 3, no. 1, pp. 9–24, Feb. 2022, doi: 10.35290/ro.v3n1.2022.539.

[14] J. Balen, D. Vajak, dan K. Salah, "Comparative Performance Evaluation of Popular Virtual Private Servers," *Journal of Internet Technology*, vol. 21, no. 2, pp. 343–356, Mar. 2020, DOI: 10.3966/160792642020032102003

[15] I. H. Madurapperuma, M. S. Shafana, and M. J. A. Sabani, “State-Of-Art Frameworks For Front-End And Back-End Web Development,” *in Proceedings of the 2nd International Conference on Science and Technology (ICST2022),* Sri Lanka, Aug. 2022. [Online]. Available: http://ir.lib.seu.ak.ac.lk/handle/123456789/6339.

[16] I. P. A. E. Pratama dan I. Made S. Raharja, *“Node.js Performance Benchmarking and Analysis at Virtualbox, Docker, and Podman Environment Using Node-Bench Method”*, JOIV — International Journal on Informatics Visualization, vol. 7, no. 4, pp. 2240–2246, 2023, doi: 10.62527/joiv.7.4.1762

[17] C. Y. Andika and S. Rudiarto, “Rancang Bangun Aplikasi Sosial Media Crawler Menggunakan Nodejs Menerapkan Konsep Non-Blocking I/O,” *Jurnal Ilmiah FIFO*, vol. 9, no. 2, pp. 132–137, 2017, doi: 10.22441/fifo.2017.v9i2.006.

[18] E. Zimányi, M. Sakr, and A. Lesuisse, “Mobilitydb: A Mobility Database Based on Postgresql and Postgis,” *ACM Trans. Database Syst.*, vol. 45, no. 4, pp. 1–42, Dec. 2020, doi: 10.1145/3406534.

.