RANCANG BANGUN MONITORING KELEMBABAN DAN SUHU BERBASIS *BLUETOOTH MESH*

LAPORAN TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan
Pendidikan Diploma Tiga (D-3)
Program Studi Teknik Telekomunikasi
Jurusan Teknik Elektro

ASEP YONO 322 22 009
AI WULANDARI 322 22 024

PROGRAM STUDI D-3 TEKNIK TELEKOMUNIKASI
JURUSAN TEKNIK ELEKTRO
POLITEKNIK NEGERI UJUNG PANDANG
MAKASSAR

2025

HALAMAN PENGESAHAN

Laporan Tugas Akhir dengan judul " **Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh*** " oleh Asep Yono 32222009 dan Ai Wulandari 32222024 dinyatakan layak untuk diujikan.

Makassar, 4 Agustus 2025

Pembimbing I

Dr. Umar Katu, S.T., M.T.
NIP 197308202008011005

Pembimbing II

Sandryones Palinggi
NIP 198806112022031002

Mengetahui,
Ketua Program Studi

Yuliarti, S.ST., M.T.
NIP 197706032002122002

HALAMAN PENERIMAAN

Pada hari ini, Jumat tanggal 8 Agustus 2025, tim penguji ujian sidang laporan Tugas Akhir telah menerima hasil ujian sidang laporan Tugas Akhir oleh mahasiswa Asep Yono NIM 32222009 dan Ai Wulandari NIM 32222024 dengan judul " **Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh***".

Makassar, 08 Agustus 2025

Tim Penguji Ujian Sidang Laporan Tugas Akhir:

1. Prof. Dr. Ir. Hafsah Nirwana, M.T.	Ketua	.....
2. Misnawati, S.T., M.T.	Sekretaris	.....
3. Airin Dewi Utami Thamrin, S.T., M.T.	Anggota	.....
4. Ir. Farchia Ulfiah, M.T.	Anggota	.....
5. Dr. Umar Katu, S.T., M.T.	Anggota	.....
6. Sandryones Palinggi	Anggota	.....

RINGKASAN

Penelitian ini bertujuan untuk membuat dan membangun sistem pemantau kelembaban tanah, kelembaban udara, dan suhu menggunakan teknologi *Bluetooth Mesh* agar bisa membuat jaringan sensor yang bisa diandalkan, bisa diperluas, dan tetap bekerja meski ada *Node* yang rusak atau tidak berfungsi.

Sistem ini dirancang terdiri dari empat *Node* sensor yang masing-masing dilengkapi sensor DHT22 dan sensor kelembaban tanah, serta salah satu *gateway Node* yang bertindak sebagai koordinator. Metode uji terdiri dari tiga bagian, yaitu: (1) menguji kinerja di jarak 2 meter, (2) mensimulasikan kegagalan *Node* dengan menonaktifkan salah satu *Node*, dan (3) menguji tanpa fitur *relay* untuk melihat dampaknya terhadap performa jaringan. Parameter yang diukur meliputi tingkat pengiriman paket, keterlambatan, dan konsistensi data dari sensor.

Hasil pengujian menunjukkan tingkat pengiriman paket mencapai 100% dalam kondisi normal, dengan keterlambatan antara 1,086 hingga 124,211 ms. Saat terjadi kegagalan *Node*, sistem tetap berjalan dengan baik dan keterlambatan menurun menjadi 2,793 hingga 42,592 ms. Ketika fitur *relay* dinonaktifkan, keterlambatan meningkat besar hingga 33,427 ms untuk *Node* terjauh, membuktikan bahwa *relay* sangat penting untuk menjaga kinerja jaringan. Nilai dari sensor yang didapatkan tetap konsisten (suhu: 27,1 hingga 29,1°C, kelembaban udara: 70,4 hingga 85,5%. Kelembaban tanah: 200 hingga 2755), yang menunjukkan sistem ini akurat dalam mengumpulkan data.

KATA PENGANTAR

Puji syukur kehadirat ALLAH SWT atas limpahan rahmat dan karunia-Nya sehingga kami dapat menyelesaikan Tugas Akhir **"Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh*".** Penyusunan Tugas Akhir ini sebagai salah satu syarat akademik yang harus dipenuhi oleh setiap mahasiswa dalam menyelesaikan studi pada Program Studi D3 Teknik Telekomunikasi, Jurusan Teknik Elektro, Politeknik Negeri Ujung Pandang.

Sebagai manusia biasa, kami menyadari bahwa pada laporan Tugas Akhir ini masih banyak terdapat kekurangan dan masih memerlukan perbaikan secara menyeluruh. Hal ini disebabkan keterbatasan pengetahuan dan pengalaman kami dalam menyusun laporan Tugas Akhir. Oleh karena itu, berbagai masukan dan saran yang sifatnya membangun sangatlah diharapkan untuk menyempurnakan laporan Tugas Akhir ini. Dalam penyelesaian laporan Tugas Akhir ini, banyak pihak yang terlibat dan berperan serta untuk mewujudkan selesainya laporan Tugas Akhir ini. Oleh karena itu, kami ingin mengucapkan terima kasih kepada:

1. *Allah Subhana Wa Ta'Ala dan Rasulullah Muhammad Shallallahu 'Alaihi Wassalam* yang senantiasa memberi petunjuk dan membimbing ke jalan yang lurus.
2. Kedua orang tua dan keluarga pemyusun yang telah memberikan dukungan baik berupa materil maupun moril serta doa yang senantiasa diberikan kepada penyusun.
3. Bapak Prof. Dr. Jamal, S.T., M.T., selaku Direktur Politeknik Negeri Ujung Pandang.
4. Bapak Prof. Ahmad Rizal Sultan, S.T., M.T., Ph.D., selaku Ketua Jurusan Teknik Elektro Politeknik Negeri Ujung Pandang.
5. Ibu Yuniarti, S.ST., M.T., selaku Ketua Program Studi D3 Teknik Telekomunikasi Politeknik Negeri Ujung Pandang.

6. Bapak Dr. Umar Katu, S.T., M.T., selaku Pembimbing 1 Tugas Akhir D3 Teknik Telekomunikasi.
7. Bapak Sandryones Palinggi, selaku Pembimbing 2 Tugas Akhir D3 Teknik Telekomunikasi.
8. Bapak Sahbuddin Abdul Kadir, S.T., M.T., selaku wali kelas 3A D3 Tekenik Telekomunikasi.
9. Seluruh Dosen Pengajar D3 Teknik Telekomunikasi yang telah memberikan ilmu yang bermanfaat bagi Penyusun.
10. Rekan sekelas atas dukungan, motivasi, serta kebersamaan selama proses perkuliahan hingga penyusunan Tugas Akhir ini.
11. Rekan-rekan mahasiswa Teknik Elektro, khususnya Program Studi D3 Teknik Telekomunikasi angkatan 2022 yang telah membantu Penyusun dan memberikan dukungan dalam menyusun laporan Tugas Akhir sampai selesai.
12. Teknologi *Artificial Intelligence* (AI) yang telah membantu penyusun dalam mencari referensi, menyusun ide, mempermudah penyusunan Tugas Akhir ini.
13. Semua pihak yang telah membantu terselesainya laporan Tugas Akhir ini yang tidak dapat Penyusun sebut satu persatu.

Demi kesempurnaan laporan Tugas Akhir ini, saran dan kritik yang sifatnya membangun sangat Penyusun harapkan. Semoga laporan ini dapat memberikan manfaat yang berarti bagi pihak yang membutuhkan.

Makassar, 8 Agustus 2025

Penyusun

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR ISTILAH

ADC (*Analog to Digital Converter*)	Komponen pada mikrokontroler yang mengubah sinyal analog menjadi data digital.
Advertising (BLE *Advertising*)	Mekanisme pada BLE untuk menyiarkan data singkat agar dapat diterima perangkat lain.
Arduino IDE	Perangkat lunak untuk menulis, menguji, dan mengunggah kode program ke ESP32.
Bluetooth Classic	Versi awal *Bluetooth* dengan konsumsi daya lebih besar, mendukung BR dan EDR.
Bluetooth Low Energy (BLE)	Versi *Bluetooth* hemat daya, digunakan untuk komunikasi perangkat IoT.
Bluetooth Mesh	Protokol komunikasi nirkabel berbasis BLE untuk komunikasi *many-to-many*.
BR/EDR (*Basic Rate*/*Enhanced Data Rate*)	Mode komunikasi *Bluetooth Classic* dengan kecepatan 1–3 Mbps.
Delay	Waktu tunda antara pengiriman dan penerimaan data dalam jaringan.
DF*Robot* SEN0308	Sensor kapasitif kelembaban tanah analog yang tahan air.
DHT22 (AM2302)	Sensor digital untuk mengukur suhu dan kelembaban udara.
ESP32 (WEMOS D1 R32)	Mikrokontroler *dual-core* dengan Wi-Fi dan *Bluetooth*, digunakan sebagai *node* sensor.

Frequency ISM 2,4 GHz	Pita frekuensi nirkabel tanpa lisensi yang digunakan oleh Wi-Fi dan *Bluetooth*.
Gateway	*Node* pusat yang menerima data dari *node* sensor dan menyalurkannya ke sistem monitoring.
GPIO (*General Purpose Input Output*)	Pin mikrokontroler yang bisa difungsikan sebagai *input* atau *output*.
Hop	Perpindahan data dari satu *node* ke *node* lain dalam jaringan mesh.
Kelembaban	Kandungan uap air di udara atau kadar air dalam tanah.
Komunikasi *Point-to-Point*	Pola komunikasi antar dua perangkat langsung tanpa perantara.
Latency	Keterlambatan transmisi data dalam jaringan (serupa dengan *delay*).
Library	Kumpulan kode siap pakai dalam pemrograman (misalnya BLE, DHT).
Mesh Network	Topologi jaringan di mana setiap perangkat bisa mengirim dan meneruskan data.
MQTT	Protokol komunikasi ringan berbasis *publish*/*subscribe* untuk IoT.
Monitoring	Proses pengamatan, pengumpulan, dan analisis data sensor secara kontinu.
Node	Unit dalam jaringan mesh yang terdiri dari ESP32 + sensor.

Pairing	Proses awal untuk menghubungkan dua perangkat *Bluetooth*.
Packet Loss	Kehilangan paket data yang seharusnya diterima.
Point-to-Multipoint	Jaringan komunikasi di mana setiap *node* berkomunikasi secara langsung dengan beberapa *node* lain secara simultan dalam jangkauannya.
PWM (*Pulse Width Modulation*)	Metode pengendalian sinyal digital dengan variasi lebar pulsa.
Relay (*Message Relay*)	Proses penerusan data dari satu *node* ke *node* lain dalam mesh.
RSSI (*Received Signal Strength Indicator*)	Indikator kekuatan sinyal yang diterima pada komunikasi nirkabel.
Sensor Kelembaban Tanah	Sensor kapasitif untuk mendeteksi kadar air dalam tanah.
Sensor Suhu	Perangkat untuk mengukur tingkat panas/dingin lingkungan.
Serial Monitor	Fitur Arduino IDE untuk menampilkan data sensor dari ESP32.
Throughput	Jumlah data yang berhasil dikirimkan per periode waktu tertentu.
TTL (*Time To Live*)	Batas jumlah *hop* yang boleh dilalui oleh sebuah paket dalam *mesh*.

DAFTAR LAMPIRAN

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Asep Yono

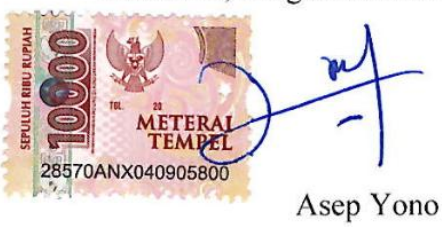
NIM : 32222009

Menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam laporan tugas akhir yang berjudul "Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh*" merupakan gagasan dan hasil karya penyusun sendiri dengan arahan dosen pembimbing dan belum pernah dijadikan dalam bentuk apapun pada perguruan tinggi instansi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang telah diterbitkan dari penulis/penyusun lain telah disebutkan dalam naskah dan dicantumkan dalam laporan Tugas Akhir ini.

Jika pernyataan penyusun tersebut diatas tidak benar, penyusun siap menanggung risiko yang ditetapkan oleh Politeknik Negeri Ujung Pandang.

Makassar, 8 Agustus 2025



Asep Yono

32222009

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Ai Wulandari

NIM : 32222024

Menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam laporan tugas akhir yang berjudul "Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh*" merupakan gagasan dan hasil karya penyusun sendiri dengan arahan dosen pembimbing dan belum pernah dijadikan dalam bentuk apapun pada perguruan tinggi instansi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang telah diterbitkan dari penulis/penyusun lain telah disebutkan dalam naskah dan dicantumkan dalam laporan Tugas Akhir ini.

Jika pernyataan penyusun tersebut diatas tidak benar, penyusun siap menanggung risiko yang ditetapkan oleh Politeknik Negeri Ujung Pandang.

Makassar, 8 Agustus 2025

Ai Wulandari

32222024

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pemantauan kelembaban tanah secara langsung dan tepat waktu sangat penting untuk meningkatkan cara penggunaan air dalam irigasi dan hasil pertanian. Cara lama biasanya hanya mengandalkan pengamatan mata dan jadwal penyiraman manual yang sering tidak tepat. Hal ini bisa menyebabkan air terlalu sedikit atau terlalu banyak yang bisa merusak tanaman dan juga membuang sumber daya (Putra et al., 2021). Beberapa solusi berbasis teknologi internet mulai digunakan, tetapi masih ada masalah karena luasnya lahan pertanian dan kurangnya fasilitas komunikasi yang memadai.

Teknologi Wi-Fi memiliki jangkauan terbatas dan boros daya, sedangkan modul LoRa meskipun jangkauannya lebar, harganya agak mahal untuk penggunaan dalam skala kecil (Santoso et al., 2022). *Bluetooth Low Energy* (BLE) biasa lebih hemat daya, namun struktur pengiriman datanya hanya satu ke satu, sehingga membatasi jangkauannya dan jumlah sensor yang bisa dipasang di satu lahan. Oleh karena itu, dibutuhkan teknologi yang mampu menghubungkan banyak sensor sekaligus dengan jangkauan luas dan daya yang rendah.

Bluetooth Mesh adalah jawaban yang cocok untuk tantangan ini. Berbeda dengan BLE biasa, *Bluetooth Mesh* menggunakan sistem jaringan satu ke banyak, dimana setiap perangkat (*Node*) bisa tidak hanya mengirim data sendiri, tetapi juga meneruskan data dari perangkat lain, seperti dalam sistem lomba lari estafet (*Bluetooth* SIG, 2017). Karakteristik ini memungkinkan jaringan bertambah luas secara alami dengan menambahkan *Node-Node* baru, yang sangat cocok dibuat untuk daerah pertanian.

Penelitian oleh Chen et al. (2020) dalam jurnal "*An Agricultural Monitoring System Based on Bluetooth Mesh*" telah menunjukkan bahwa teknologi ini layak digunakan, membuktikan bahwa jaringan *mesh* bisa memberikan cakupan yang stabil dan luas untuk lahan pertanian dengan penggunaan daya yang hemat. Temuan ini didukung oleh penelitian Aji et al. (2023) yang menunjukkan bahwa *Bluetooth Mesh* lebih bisa diandalkan dibandingkan topologi *star* dalam lingkungan yang banyak penghalang.

Berdasarkan uraian tersebut, penelitian ini mengembangkan prototipe sistem pemantauan kelembaban dan suhu berbasis *Bluetooth Mesh* dengan modul ESP32. Setiap *Node* dilengkapi sensor DHT22 untuk mengukur suhu dan kelembaban udara, serta sensor *Gravity-Analog Waterproof Capacitive Soil Moisture* Sensor V2.0 (DFRobot SEN0308) untuk kelembaban tanah. Data hasil pengukuran dikirimkan melalui jaringan *Bluetooth Mesh* menuju *gateway* dan ditampilkan pada *serial monitor*. Sistem ini diharapkan menjadi solusi sederhana, terjangkau, dan dapat diperluas untuk kebutuhan pemantauan lingkungan secara *real-time* di area luas tanpa ketergantungan pada jaringan kabel atau internet langsung.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, dipaparkan rumusan masalah sebagai berikut:

1. Bagaimana cara merancang jaringan sensor berbasis *Bluetooth Mesh* untuk memonitoring kelembaban dan suhu pada tanah?
2. Bagaimana cara membuat jaringan sensor berbasis *Bluetooth Mesh* untuk memonitoring kelembaban dan suhu pada tanah?

1.3 Ruang Lingkup Kegiatan

Agar lebih terfokus dan mencapai tujuan yang diinginkan, perlu adanya batasan masalah sebagai berikut:

1. Perancangan terdiri dari 4 *Node* ESP32 yang saling terhubung menggunakan *Bluetooth Low Energy* (BLE) dengan prinsip *mesh* sederhana.
2. Perancangan dilengkapi dengan sensor di setiap *Node*, DHT22 untuk membaca suhu lingkungan, kelembaban udara dan *Gravity-Analog Waterproof Capacitive Soil Moisture* Sensor V2.0-DF*Robot* SEN0308 untuk kelembaban tanah.
3. Data dikirim ke serial monitor untuk dicatat dan dianalisis.

1.4 Tujuan

Untuk memberikan fokus pada rumusan masalah maka dibuat tujuan sebagai berikut :

1. Merancang jaringan sensor berbasis *Bluetooth Mesh* untuk memonitoring kelembaban dan suhu pada tanah.
2. Membuat jaringan sensor berbasis *Bluetooth Mesh* untuk memonitoring kelembaban dan suhu pada tanah.

1.5 Manfaat

Adapun manfaat yang ingin dicapai dari perancangan jaringan sensor berbasis *Bluetooth Mesh* ini sebagai berikut:

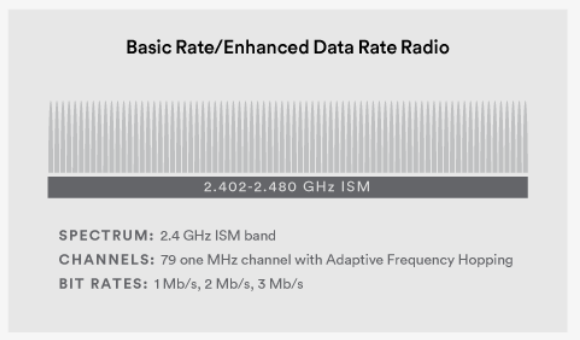
1. Memfasilitasi pengawasan suhu dan kelembaban secara langsung diberbagai tempat tanpa memerlukan kabel atau akses internet secara langsung.
2. Konfigurasi *mesh* memungkinkan penambahan *Node* dengan mudah untuk memperluas area jangkauan dan meningkatkan keandalan komunikasi antar perangkat.

BAB II
TINJAUAN PUSTAKA

2.1 *Bluetooth*

Teknologi nirkabel jarak pendek *Bluetooth* memungkinkan dua perangkat terhubung secara langsung tanpa memerlukan infrastruktur jaringan pendukung seperti router nirkabel atau *access point*. Saat ini, teknologi *Bluetooth* paling umum digunakan oleh orang-orang di seluruh dunia untuk menghubungkan perangkat seperti *headphone* nirkabel, *keyboard*, *mouse*, dan *speaker* ke PC maupun perangkat seluler.

2.1.1 ***Bluetooth Classic***, yang mendukung dua kecepatan data yang berbeda, yaitu *Basic Rate* (BR) dan *Enhanced Data Rate* (EDR). Adapun gambar berikut memperlihatkan karakteristik teknis sistem komunikasi *Bluetooth Classic* yang menggunakan teknologi *Basic Rate* (BR) dan *Enhanced Data Rate* (EDR).

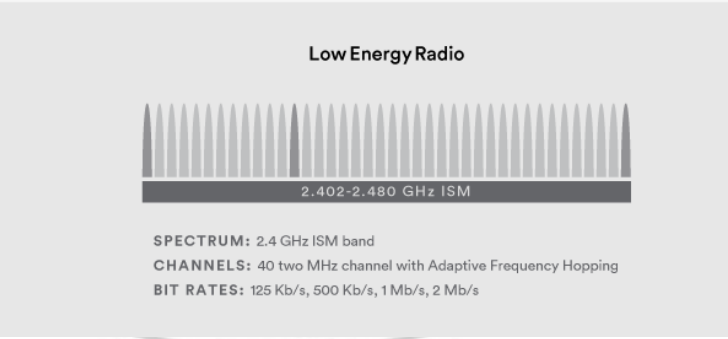


Gambar 2. 1 *Basic Rate/Enhanced Data Rate Radio*

Gambar 2.1 menunjukkan karakteristik teknis dari sistem komunikasi *Bluetooth Classic* yang menggunakan teknologi *Basic Rate/Enhanced Data Rate* (BR/EDR). Komunikasi ini beroperasi dalam pita *frequency* ISM 2,4 GHz, tepatnya antara *frequency* 2,402 GHz hingga 2,480 GHz. Dalam rentang ini, tersedia 79 kanal dengan lebar masing-masing 1 MHz yang dimanfaatkan melalui teknik *Adaptive Frequency Hop*ping, yaitu metode

untuk menghindari interferensi dengan cara berpindah-pindah kanal secara dinamis. Dari sisi kecepatan data, BR/EDR mampu mentransmisikan informasi dengan laju 1 Mb/s, 2 Mb/s, hingga maksimal 3 Mb/s, tergantung pada tingkat optimasi dan kondisi komunikasi. Gambar 2.1 tersebut memberikan gambaran visual bagaimana kanal-kanal tersebut tersebar merata di dalam spektrum, menunjukkan kepadatan kanal yang tinggi untuk mendukung koneksi simultan dengan efisiensi tinggi dalam ruang *frequency* terbatas.

2.1.2 *Bluetooth Low Energy (*BLE***),*** yang dioptimalkan untuk konsumsi daya rendah dan terutama digunakan untuk aplikasi yang dibatasi oleh daya baterai. *Bluetooth Low Energy* tidak umum digunakan untuk mentransfer data dalam jumlah besar, tetapi mendukung kualitas audio yang lebih tinggi dan opsi mendengarkan yang lebih beragam dibandingkan *Bluetooth Classic*. Pada Gambar 2.2 memperlihatkan spektrum *frequency Bluetooth Low Energy* (BLE) yang memperlihatkan rentang pita *frequency* 2,402–2,480 GHz, jumlah kanal, serta kecepatan bitnya.



Gambar 2. 2 *Low Energi Radio*

Karakteristik dari teknologi *Bluetooth Low Energy* (BLE) dirancang untuk komunikasi nirkabel dengan konsumsi daya rendah. Sama seperti *Bluetooth Classic*, BLE juga beroperasi di pita *frequency* ISM 2,4 GHz, mencakup rentang dari *frequency* 2,402 GHz hingga 2,480 GHz. Namun, perbedaannya terletak pada struktur kanalnya, dimana BLE menggunakan

40 kanal dengan lebar masing-masing frekusnsi 2 MHz. Kanal-kanal ini tersebar merata dan memanfaatkan metode *Adaptive Frequency Hopping* untuk mengurangi gangguan sinyal dari perangkat lain yang menggunakan *frequency* yang sama. Kecepatan transmisi datanya bervariasi, mulai dari 125 Kb/s hingga maksimum 2 Mb/s, tergantung pada mode efisiensi dan kebutuhan aplikasi. Melalui pendekatan ini, BLE dapat mengirim data secara efisien sambil tetap menghemat energi, sehingga sangat cocok digunakan untuk perangkat IoT, sensor nirkabel, dan perangkat *wearable* yang membutuhkan koneksi berkelanjutan tanpa menguras baterai dengan cepat. Teknologi *Bluetooth* beroperasi pada *frequency* radio pada rentang *frequency* 2,4 GHz. Dua standar *Bluetooth* yang saat ini digunakan dan spesifikasinya seperti yang ditunjukkan pada Tabel 2.1.

Tabel 2. 1 BLE & *Bluetooth Classic Spesification*

	***Bluetooth Low Eenergy* (BLE)**	***Bluetooth Classic***
Frequency Band	2,4GHz ISM *Band* (2,402 – 2,480 GHz Utilized)	2.4GHz ISM *Band* (2,402 – 2,480 GHz *Utilized*)
Channels	*40 channels with* 2 *MHz spacing (3 advertising channels/37 data channels)*	*79 channels with* 1 *MHz spacing*
Channel Usage	*Frequency-Hopping Spread Spectrum (FHSS)*	*Frequency-Hopping Spread Spectrum (FHSS)*
Modulation	GFSK	GFSK, π/4 DQPSK, 8DPSK

	***Bluetooth Low Eenergy* (BLE)**	***Bluetooth Classic***
Data Rate	LE 2M PHY: 2 Mb/s LE 1M PHY: 1 Mb/s LE Coded PHY (S=2): 500 Kb/s LE Coded PHY (S=8): 125 Kb/s	EDR PHY (8DPSK): 3 Mb/s EDR PHY (π/4 DQPSK): 2 Mb/s BR PHY (GFSK): 1 Mb/s
Data Transports	*Asynchronous Connection-oriented* *Isochronous Connection-oriented* *Asynchronous Connectionless* *Synchronous Connectionless* *Isochronous Connectionless*	*Asynchronous Connection-oriented* *Synchronous Connection-oriented*
Communication Topologies	*Point-to-Point (including piconet)* *Broadcast* *Mesh*	*Point-to-Point (including piconet)*

Organisasi Standar *Bluetooth Special Interest Group* (SIG) bertanggung jawab atas pengembangan dan lisensi teknologi *Bluetooth*. Intel merupakan anggota pendiri *Bluetooth SIG* dan telah memainkan peran penting dalam evolusi serta penyebaran teknologi ini. Bahkan, nama

Bluetooth sendiri diusulkan oleh Jim Kardach dari Intel pada tahun 1997, yang terinspirasi oleh raja Swedia abad ke-10, *Harald Bluetooth*.

Pada masa awalnya, banyak PC yang mengandalkan *dongle* eksternal untuk terhubung ke perangkat *Bluetooth*, sering kali menggunakan port USB. Pendekatan ini memberikan fungsionalitas *Bluetooth* dasar tetapi sering dianggap merepotkan oleh pengguna. Saat ini, sebagian besar PC dilengkapi dengan kartu jaringan tertanam yang memiliki fungsi Wi-Fi dan *Bluetooth*. Hal ini memungkinkan kinerja yang lebih baik dan koordinasi antar kapabilitas radio, membantu mencegah gangguan serta memastikan kecepatan transfer data yang memadai.

Teknologi *Bluetooth* terutama digunakan untuk menghubungkan perangkat periferal secara nirkabel ke ponsel, desktop, dan laptop. Beberapa aksesori *Bluetooth* yang paling umum meliputi *mouse, keyboard, speaker,* dan *headphone*. Banyak pengontrol *game* juga menggunakan teknologi *Bluetooth* untuk konektivitas nirkabel.

Selain itu, teknologi *Bluetooth* semakin umum digunakan dalam elektronik konsumen seperti kamera, televisi, bahkan peralatan rumah tangga seperti kulkas atau oven. Perangkat medis seperti sensor *glukosa* atau *pacemaker* juga bergantung pada konektivitas *Bluetooth*. Sistem infotainment mobil modern menggunakan teknologi *Bluetooth* untuk melakukan *streaming* musik atau navigasi dari ponsel pengemudi.

2.1.3 Dukungan untuk Koneksi Ganda

Untuk mendukung banyak perangkat yang ingin terhubung melalui teknologi *Bluetooth*, banyak perangkat modern dirancang untuk mendukung beberapa koneksi *Bluetooth* secara bersamaan. Contohnya, dalam skenario kerja dari rumah, pengguna dapat menyinkronkan perangkat seperti *keyboard* dan *mouse* dengan beberapa PC secara bersamaan dan dengan mudah beralih hanya dengan menekan satu tombol.

2.1.4 Menghubungkan Perangkat *Bluetooth*

Menghubungkan dua perangkat *Bluetooth* adalah proses yang sederhana, yang sering disebut *pairing*. Banyak perangkat *Bluetooth* modern secara otomatis masuk ke mode *pairing* saat pertama kali dinyalakan. Namun, dalam beberapa kasus, mode *pairing* harus diaktifkan secara manual.

Dalam perangkat *Bluetooth*, proses *pairing* mirip dengan bertukar informasi kontak. Pertama, setiap perangkat mendaftarkan informasi *pairing*, termasuk kunci keamanan, dari perangkat pasangannya. Informasi ini disimpan pada kedua perangkat sehingga dapat dengan mudah terhubung atau secara otomatis terhubung kembali tanpa mengulangi proses *pairing* awal.

Bergantung pada perangkatnya, proses *pairing* awal ini mungkin melibatkan perbandingan angka yang ditampilkan di kedua perangkat untuk memastikan koneksi yang benar, atau memasukkan PIN dari satu perangkat ke perangkat lainnya. Langkah-langkah ini dirancang untuk mencegah pengguna secara tidak sengaja terhubung ke perangkat yang salah dan untuk mencegah pihak yang tidak diinginkan mengakses perangkat *Bluetooth*.

2.2 *Bluetooth Low Energy* (BLE) & BT *Mesh*

2.2.1 *Bluetooth Low Energi* (BLE)

Radio *Bluetooth Low Energy* (BLE) dirancang untuk operasi dengan daya yang sangat rendah. Radio ini mentransmisikan data melalui 40 saluran di pita *frequency* ISM 2,4GHz yang tidak memerlukan lisensi, memberikan fleksibilitas besar bagi pengembang untuk menciptakan produk yang memenuhi kebutuhan konektivitas yang beragam.

Bluetooth LE mendukung berbagai topologi komunikasi, mulai dari *point-to-point* hingga *broadcast*, dan yang terbaru adalah *mesh*. Hal ini memungkinkan teknologi *Bluetooth* mendukung pembuatan jaringan perangkat berskala besar yang andal.

Awalnya dikenal sebagai teknologi komunikasi antar perangkat, *Bluetooth* LE kini juga banyak digunakan sebagai teknologi pemetaan perangkat untuk memenuhi permintaan layanan lokasi dalam ruangan yang akurat. *Bluetooth* LE kini mencakup fitur-fitur yang memungkinkan satu perangkat menentukan keberadaan, jarak, dan arah perangkat lain.

2.2.2 *Bluetooth Mesh* (BLE *Mesh*)

Komunikasi memainkan peran penting dalam banyak aplikasi modern, terutama di bidang *Internet of Things* (IoT). Dibandingkan dengan mekanisme komunikasi kabel tradisional, teknologi komunikasi nirkabel lebih diminati karena keunggulan yang ditawarkan, termasuk efisiensi, ketersediaan, fleksibilitas, pengurangan masalah instalasi, dan penghematan biaya. Namun, teknologi nirkabel seperti ZigBee, Z-*Wave*, *Thread*, 6LoWPAN, dan Wi-Fi memiliki kekurangan seperti cakupan yang terbatas, latensi tinggi, serta tingkat keandalan yang rendah akibat kehilangan paket data, sehingga sering kali tidak mampu memenuhi kebutuhan banyak aplikasi IoT secara memadai.

Bluetooth Mesh (BT *Mesh*) adalah protokol nirkabel baru yang secara resmi dirilis pada tahun 2017. Protokol ini dikembangkan untuk memberikan cakupan yang lebih luas, keandalan yang lebih tinggi, serta biaya yang lebih rendah dan konsumsi daya yang hemat. Versi *Bluetooth* ini dibangun di atas inti *Bluetooth Low Energy* (BLE) dari spesifikasi *Bluetooth* 4.2. BT *Mesh* memungkinkan komunikasi banyak-ke-banyak (*many-to-many*) melalui BLE, memungkinkan terciptanya jaringan *mesh* perangkat *Bluetooth*.

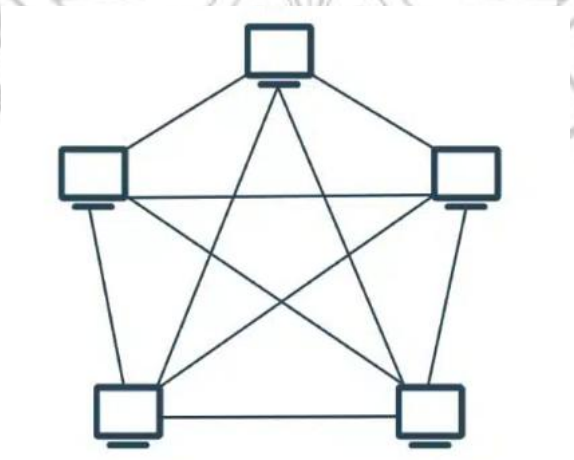
Fitur-fitur BT *Mesh* menjadikannya sangat cocok untuk berbagai aplikasi berbasis IoT, seperti otomasi rumah dan industri. Dibandingkan dengan jaringan *mesh* lainnya, seperti Wi-Fi, ZigBee, *Thread*, dan Z-*Wave*, yang menggunakan teknik perutean (*routing*) dengan penundaan tinggi, BT

Mesh menggunakan teknik *flooding* yang sangat andal dan mudah diterapkan.

2.3 Topologi *Mesh*

Topologi *mesh* adalah salah satu konfigurasi jaringan yang memungkinkan komunikasi antar perangkat secara langsung tanpa perlu bergantung pada satu hub pusat. Keuntungan utama dari topologi *mesh* adalah skalabilitas yang tinggi, karena dapat mendukung ribuan perangkat dalam satu jaringan tanpa menambah infrastruktur yang kompleks. Aisyah Amini A'maari Suhud (2020) dalam penelitiannya tentang penerapan jaringan *mesh* dalam IoT untuk pemantauan lingkungan, menyatakan bahwa sistem ini cocok untuk skenario perkotaan dimana banyak perangkat perlu saling berkomunikasi dalam area yang luas. Dalam topologi *mesh*, setiap perangkat (*Node*) dalam jaringan dapat berfungsi sebagai penghubung bagi perangkat lainnya, meminimalkan potensi kegagalan jaringan dan meningkatkan ketahanan sistem. Dapat dilihat pada Gambar 2.3 bentuk topologi *mesh* dimana setiap perangkat saling terhubung satu sama lain.

Secara keseluruhan, *mesh network* adalah sebuah konsep topologi jaringan yang inovatif dan fleksibel. Dalam dunia yang terus berkembang menuju konektivitas yang lebih luas dan kompleks, pemahaman tentang *mesh network* menjadi penting untuk membangun jaringan yang efisien dan handal.



Gambar 2. 3 Topologi *Mesh*

2.4 Sistem Monitoring

Sistem adalah sekumpulan kegiatan, komponen, unsur, elemen, atau variabel yang terorganisir, saling berinteraksi, dan berhubungan satu sama lain untuk bekerja sama secara harmonis dalam mencapai tujuan yang telah ditetapkan. Secara umum, unsur-unsur yang mendefinisikan suatu sistem meliputi masukan (*input*), pengolahan (*processing*), dan keluaran (*output*). Sistem selalu terhubung dengan lingkungannya, sehingga umpan balik (*feedback*) dapat berasal baik dari keluaran maupun dari lingkungan sistem tersebut. Organisasi dianggap sebagai sebuah sistem yang memiliki semua unsur tersebut, dan sebuah sistem dalam organisasi dapat berfungsi dengan baik jika masukan dapat diproses menjadi keluaran yang bermanfaat bagi pihak yang memerlukannya (Laudon 2021).

Monitoring adalah proses pengumpulan dan analisis informasi berdasarkan indikator yang ditetapkan secara sistematis dan kontinu tentang kegiatan/program sehingga dapat dilakukan tindakan koreksi untuk penyempurnaan kegiatan/program itu selanjutnya. Monitoring adalah pemantauan yang dapat dijelaskan sebagai kesadaran apa yang ingin diketahui, pemantauan berkadar tingkat tinggi dilakukan agar dapat membuat pengukuran melalui waktu yang menunjukkan pergerakan ke arah tujuan atau menjauh dari itu. Monitoring akan memberikan informasi tentang status dan kecenderungan bahwa pengukuran dan evaluasi yang diselesaikan berulang dari waktu ke waktu, pemantauan umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses berikut objek atau untuk mengevaluasi kondisi atau kemajuan menuju tujuan hasil manajemen atas efek tindakan dari beberapa jenis antara lain tindakan untuk mempertahankan manajemen yang sedang berjalan. Pemantauan biasanya dilakukan dengan tujuan tertentu, seperti memeriksa objek terhadap proses selanjutnya atau untuk memberi evaluasi keadaan atau peningkatan. Pengukuran serta evaluasi yang dilaksanakan secara rutin untuk memberi suatu informasi mengenai status serta kecenderungan (Fauzia, E., & Narini, M.2018).

2.5 Suhu

Suhu adalah salah satu parameter penting dalam berbagai aplikasi monitoring lingkungan, industri, dan pertanian. Pengukuran suhu dilakukan untuk memantau kondisi lingkungan secara *real-time* dan membantu pengambilan keputusan yang tepat. Menurut jurnal yang dipublikasikan oleh O'Neill dan Williams (2020), suhu adalah tingkat panas atau dinginnya suatu objek atau lingkungan yang diukur menggunakan sensor suhu. Sensor Suhu sendiri berfungsi mengubah variasi suhu menjadi sinyal listrik yang dapat diinterpretasikan oleh sistem pengukuran. Dalam pengembangannya, sensor suhu berbasis teknologi semikonduktor seperti thermistor dan sensor suhu berbasis resistansi banyak digunakan karena keakuratannya dan kemampuannya untuk integrasi dengan system digital.

2.6 Kelembaban

Kelembaban adalah tingkat kandungan uap air di dalam udara yang mempengaruhi berbagai aspek kehidupan dan proses industri. Menurut sebuah studi yang dipublikasikan oleh Wang et al. (2020) dalam jurnal Sensors, kelembaban sering diukur menggunakan sensor kelembaban yang mampu mendeteksi perubahan kadar air dalam lingkungan secara akurat.

Kelembaban udara diukur dalam bentuk kelembaban relatif dengan satuan persen (%). Angka ini menunjukkan seberapa banyak uap air yang ada di udara dibandingkan dengan jumlah maksimal yang bisa ditampung pada suhu tertentu. Sementara itu, kelembaban tanah biasanya diukur berdasarkan kandungan air volumetrik, yang juga menggunakan satuan persen (%), dan menyatakan volume air yang ada dalam suatu volume tanah.

2.7 WEMOS D1 R32 *Development Board*

Gambar 2. 4 Mikrokontroler WEMOS D1 R32
Development Board

WEMOS D1 R32 adalah papan pengembangan berbasis mikrokontroler ESP32 dengan arsitektur Xtensa. Papan ini memiliki kecepatan CPU maksimal dengan *frequency* 240 MHz dan kapasitas penyimpanan flash sebesar 4 MB. ESP32 adalah mikrokontroler yang dikenalkan oleh *Espressif System* merupakan penerus dari mikrokontroler ESP8266. Pada mikrokontroler ini sudah tersedia modul Wi-Fi dalam chip sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. Terlihat pada Gambar 2.4 merupakan tampilan dari ESP32. Memiliki beberapa Pin yang dijadikan input atau output untuk menyalakan LCD, lampu, bahkan untuk menggerakan motor DC. Spesifikasi ESP32 dapat dilihat pada Tabel 2.2.

Tabel 2. 2 Spesifikasi ESP32

Model	ESP32
Tegangan	3.3/5 Volt
CPU	Xtensa® 32-bit LX6 *DuaL-core processor* 240 MHz
Arsitektur	XTENSA
Flash memory	4 MB

SRAM	520 kb
GPIO Pin (ADC/ DAC/PWM)	40 (16/16/18)
BLE	4.2
WiFi	802.11 b/g/n (2.4 GHz)
Interface	*Micro* USB *Connection*
USB *Bridge* IC	CH340G
Board Dimension	68 x 53 x 3.1 mm

Terlihat pada tabel 2.2 spesifikasi WEMOS D1 R32 dengan berbagai *pin out* nya memori dan terdapat *bluetooth* 4.2 *low energy* serta tersedia WiFi 802.11 b/g/n yang memungkinkan untuk mengaplikasikan *Internet of Things* dengan mikrokontroler ESP32.

2.8 *Gravity-Analog Waterproof Capacitive Soil Moisture* Sensor *V*2.*0*-DF*Robot SEN0*3*08*

Gambar 2. 5 Sensor *Soil Moisture Waterproof Capacitive V*2.*0*

Gravity-Analog Waterproof Capacitive Soil Moisture Sensor V2.0-DF*Robot* SEN0308 tampak pada gambar 2.5 adalah sensor kapasitif yang digunakan untuk mengukur kelembaban tanah dengan mendeteksi perubahan kapasitansi akibat adanya kadar air. Semakin banyak air di tanah, maka kapasitansinya meningkat dan menghasilkan tegangan analog yang bisa dibaca oleh mikrokontroler seperti ESP32 melalui pin ADC. Sensor ini lebih unggul dibanding versi sebelumnya

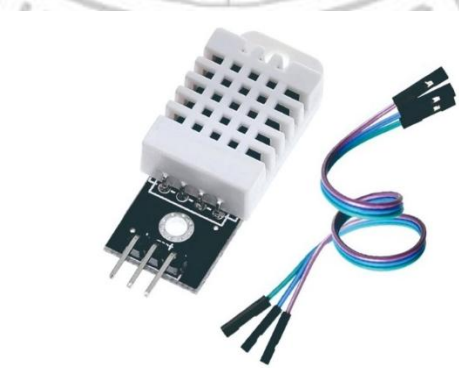
karena bagian sensornya dilapisi bahan antikorosi, sehingga lebih tahan lama dan cocok digunakan untuk pemantauan kelembaban jangka panjang, termasuk dalam proyek berbasis IoT atau sistem monitoring berbasis *Bluetooth Mesh.*

Perangkat ini dirancang untuk mengukur kelembaban tanah dengan deteksi kapasitif. Produk DF*Robot* ini memiliki ketahanan air yang lebih baik, sehingga dapat ditempatkan di tanah untuk waktu yang lama tanpa mengalami korosi. Hal ini menjadikannya jauh lebih efisien daripada sensor berbasis resistansi. Perangkat ini bekerja pada rentang tegangan yang luas, yaitu dari 3.3 V hingga 5.5 V.

Produk ini kompatibel dengan Arduino, ESP32, micro: bit, atau Raspberry Pi. Untuk dapat berfungsi dengan mikrokontroler tersebut, sensor ini hanya perlu dihubungkan ke ADC eksternal (pengubah sinyal analog menjadi digital). Ini menunjukkan bahwa sensor *Gravity-Analog Waterproof Capacitive Soil Moisture* Sensor V2.0-DF*Robot* SEN0308 cukup andal dan layak digunakan dalam penelitian atau aplikasi pertanian berbasis sensor.

2.9 Sensor *DHT22 / AM2302*

DHT22 (juga disebut AM2302) adalah sensor digital yang dapat mengukur suhu dan kelembaban udara di sekitarnya. Memiliki tingkat stabilitas yang sangat baik serta fitur kalibrasi yang sangat akurat. Koefisien kalibrasi disimpan dalam OTP *program memory*, sehingga ketika *internal* sensor mendeteksi sesuatu, maka *module* ini menyertakan koefisien tersebut dalam kalkulasinya (Sardi *et. al.*,2020).



Gambar 2. 6 Sensor Kelembaban dan Suhu (DHT22/ AM2302)

Sensor ini menggunakan komponen kapasitif untuk mendeteksi tingkat kelembaban, perubahan tingkat kelembaban di udara akan memengaruhi kapasitas bahan isolator di dalam sensor, kemudian hasilnya diubah menjadi angka digital. Untuk mengukur suhu, sensor menggunakan termistor yang mengukur perubahan hambatan listrik akibat perubahan suhu, dan hasilnya juga diubah menjadi bentuk data digital. Data hasil pengukuran kelembapan dan suhu ini kemudian diproses oleh *chip* dalam sensor dan dikirim ke mikrokontroler utama. Tampilan dan Spesifikasi sensor DHT22 ditunjukkan pada Gambar 2.6 dan Tabel 2.3 .

Tabel 2. 3 Spesifikasi Sensor DHT22 / AM2302

Pin	VCC	Power Supply 3.5V -5-5 V
	Data	*Temperature & Humidity*
	Ground	*Connected to the Ground of The Circiut*
Operating Voltage	3.5 – 5.5 Volt	
Temperature Range	-40°C – 80°C	
Humidity Range	0% – 100%	
Resolution	T/H 16 bit	
Accurary	±0.5°C and ±1%	

BAB III
METODE KEGIATAN

3.1 Tempat dan Waktu Kegiatan

Perancangan desain dan implementasi jaringan sensor berbasis *Bluetooth Mesh* dilaksanakan di Laboratorium *Microprocessor* dan IoT, Kampus 1 Politeknik Negeri Ujung Pandang. Kegiatan ini berlangsung dari bulan Februari 2025 hingga Juli 2025. Selama periode ini, dilakukan berbagai tahap mulai dari perencanaan, pengembangan, hingga pengujian sistem.

3.2 Alat dan Bahan

Untuk merancang Jaringan sensor berbasis *Bluetooth Mesh* dibutuhkan perlengkapan baik perangkat keras maupun perangkat lunak yang mendukung.

3.2.1 Perangkat Keras

Perangkat keras (*hardware*) yang digunakan dalam penelitian ini dirancang untuk membangun sebuah sistem jaringan sensor nirkabel berbasis *Bluetooth Mesh*. Pemilihan komponen dilakukan dengan mempertimbangkan aspek fungsionalitas, konsumsi daya, dan kompatibilitas dengan protokol *Bluetooth Low Energy* (BLE) yang menjadi dasar dari *Bluetooth Mesh*. Adapun perincian perangkat keras beserta fungsinya disajikan pada Tabel 3.1 di bawah ini.

Tabel 3. 1 Perangkat Keras dan Spesifikasi

No	Nama	Fungsi	Jumlah
1.	Laptop	Digunakan untuk melakukan *coding* yang akan diupload ke mikrokontroler.	1 Buah
2.	Mikrokontroler WEMOS ESP32	Sebagai mikrokontroler untuk melakukan proses perintah yang akan dijalankan.	4 Buah

No	Nama	Fungsi	Jumlah
	(WROOM D1 R32)		
3.	Sensor DHT22	Digunakan untuk mengukur tingkat kelembaban udara dan suhu.	4 Buah
4.	*Gravity-Analog Waterproof Capacitive Soil Moisture* Sensor V2.0-DF*Robot* SEN0308	Digunakan untuk mendeteksi kadar air di dalam tanah.	4 Buah
5.	Wadah Plastik	Digunakan sebagai wadah penyimpanan sampel tanah.	4 Buah
6.	Sampel Tanah	Digunakan sebagai sampel tanah yang akan diobservasi dan diuji.	Secukupnya
7.	Kabel Jumper	Digunakan sebagai penghubung untuk seluruh komponen.	Secukupnya
8.	Kabel USB	Sebagai penghubung dan *power* mikrokontroler utama yang membaca data dari sensor ke gateway (PC/Laptop).	4 Buah

3.2.2 Perangkat Lunak

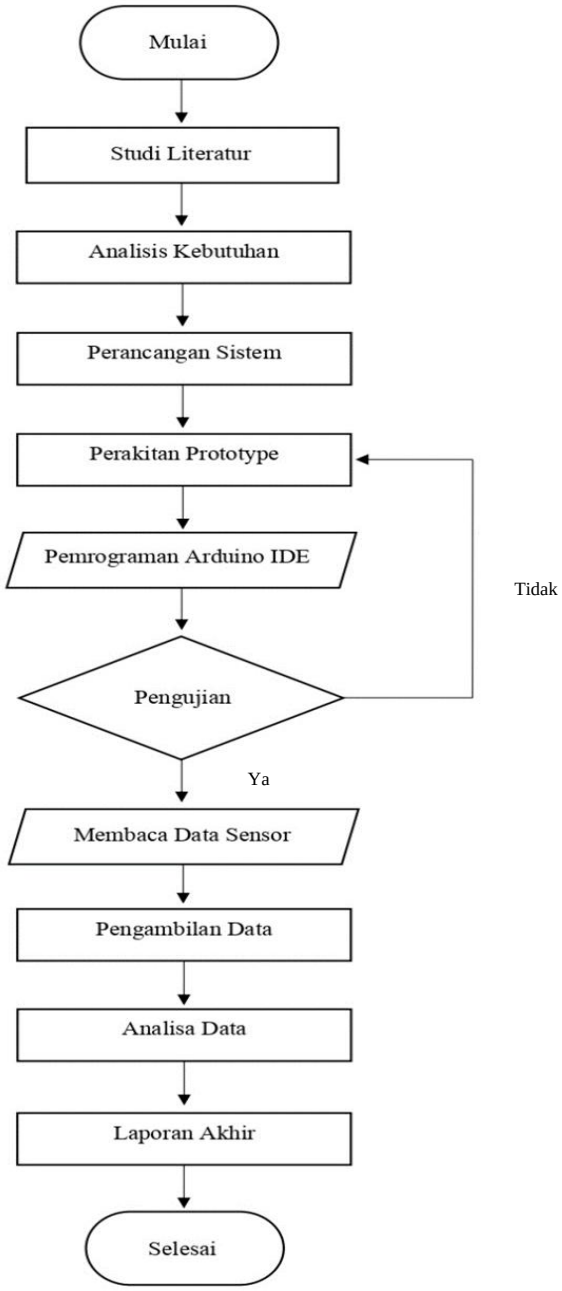
Selain perangkat keras, penelitian ini juga memerlukan perangkat lunak (*software*) untuk pengembangan, pemrograman, dan operasional sistem. Perangkat lunak dipilih berdasarkan kompatibilitasnya dengan mikrokontroler ESP32 dan dukungannya terhadap library *Bluetooth Mesh*. Spesifikasi perangkat lunak yang digunakan secara detail pada Tabel 3.2.

Tabel 3. 2 Perangkat Lunak

No	Nama	Keterangan	Fungsi
1.	Arduino IDE	-	Platform untuk menulis, menguji, dan meng-upload kode ke ESP32. *Software* ini juga berfungsi sebagai sumber data pengujian dengan hasil pembacaan data sensor pada serial monitor.

3.3 Tahapan Perancangan

Langkah kerja Rancang Bangun Monitoring Kelembaban dan Suhu Berbasis *Bluetooth Mesh* diperlihatkan dalam bentuk *flowchart* pada Gambar 3.1 :



Gambar 3. 1 *Flowchart* Perancangan Jaringan Sensor *Bluetooth Mesh*

Keterangan :

Proses membuat alat dimulai dengan tahap penelitian dan pengembangan prototipe sistem. Langkah pertama adalah melakukan studi literatur, yaitu mengumpulkan informasi dan referensi dari berbagai sumber, seperti buku, jurnal ilmiah, skripsi sebelumnya, serta dokumen teknis yang berkaitan dengan ESP32, sensor DHT22, sensor kelembaban tanah, dan struktur jaringan *Bluetooth Mesh*. Tujuan dari studi literatur ini adalah untuk mendapatkan dasar teori yang kuat dan pemahaman yang mendalam sebagai acuan dalam merancang sistem.

Setelah itu dilakukan analisis kebutuhan guna mengidentifikasi seluruh komponen yang diperlukan, baik dari sisi perangkat keras seperti jumlah ESP32, sensor, dan kabel jumper, maupun perangkat lunak seperti Arduino IDE dan *library* BLE. Hasil analisis ini menjadi dasar bagi tahap perancangan sistem, yang meliputi pembuatan diagram blok, perancangan alur komunikasi antar *Node*, pembagian peran tiap *Node* dalam jaringan *mesh*, serta perancangan struktur pemrograman dan alur kerja data untuk mendukung proses monitoring.
Tahap berikutnya adalah perakitan prototipe, dimana komponen yang telah ditentukan dirakit sesuai rancangan. Proses ini melibatkan penyusunan ESP32, sensor DHT22, dan sensor kelembaban tanah. Setelah perakitan selesai, dilakukan pemrograman menggunakan Arduino IDE untuk membuat dan mengunggah kode program ke masing-masing ESP32. Program mencakup pembacaan sensor, pengolahan data, serta komunikasi antar *Node* melalui jaringan *mesh* berbasis *Bluetooth*.

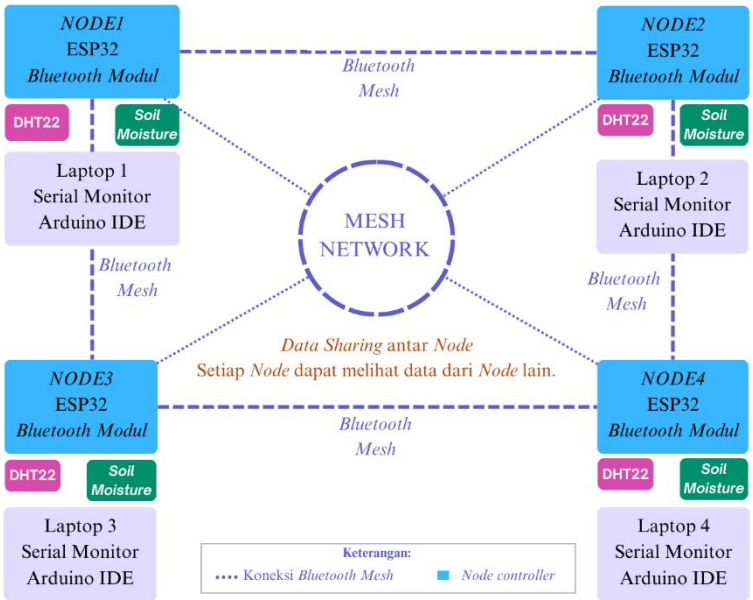
Selanjutnya dilakukan pengujian untuk memastikan semua komponen berfungsi sesuai rancangan. Apabila ditemukan kesalahan, dilakukan perbaikan baik pada rangkaian maupun program. Jika sistem lulus pengujian, dilakukan pembacaan data sensor DHT22 (suhu dan kelembaban udara) serta sensor kelembaban tanah. Data yang diperoleh kemudian masuk ke tahap pengambilan data, dimana sistem dijalankan untuk mengumpulkan data suhu dan kelembaban tanah dari setiap *Node*.

Data yang terkumpul dianalisis pada tahap analisis data untuk menilai performa alat, tingkat akurasi sensor, dan kestabilan jaringan *mesh*. Hasil analisis ini menjadi acuan dalam menilai keberhasilan sistem secara keseluruhan. Tahap akhir adalah penyusunan laporan akhir yang mendokumentasikan seluruh rangkaian kegiatan, mulai dari studi literatur hingga kesimpulan. Setelah laporan selesai, proses perancangan dinyatakan selesai, menandakan bahwa alat telah selesai dirancang dan diuji.

3.3.1 Studi Literatur

Dalam perancangan sensor berbasis *Bluetooth Mesh* ini, langkah pertama yang dilakukan adalah mencari sebanyak mungkin data, informasi serta referensi teori yang relevan dengan desain yang akan dibuat, baik itu melalui media digital dan media cetak yang memiliki data sebelumnya. Studi literatur ini sendiri bertujuan untuk mengumpulkan berbagai data seperti spesifikasi alat dan bahan, proses perancangan, simulasi sampai ke tahap penyelesaian.

3.3.2 Perancangan Sistem



Gambar 3. 2 Sistem *Bluetooth Mesh Network* dengan 4 *Node Gateway*

Pada tahap ini dilakukan perancangan rangkaian komponen, diawali dengan membuat program yang akan digunakan pada Arduino IDE. Berikut adalah *listing* program pada Arduino IDE beserta fungsinya:

```
#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEScan.h>
#include <BLEAdvertisedDevice.h>
#include <DHT.h>
```

Pada bagian awal, program mengimpor *library* yang dibutuhkan. *Library* `BLEDevice`, `BLEUtils`, `BLEScan`, dan `BLEAdvertisedDevice` digunakan untuk mendukung komunikasi *Bluetooth Low Energy* (BLE), sedangkan `DHT.h` digunakan agar ESP32 dapat membaca sensor suhu dan kelembaban bertipe DHT22.

```
#define DHTPIN 4
#define DHTTYPE DHT22
#define SOIL_PIN 34
#define LED_PIN 2
#define NODE_ID "Node1"
```

Bagian ini digunakan untuk mendefinisikan pin dan konfigurasi penting. Sensor DHT22 terhubung ke pin 4, sedangkan jenis sensor diset sebagai DHT22. Sensor kelembaban tanah dihubungkan ke pin analog 34. LED indikator menggunakan pin 2. Identifier atau nama *`Node`* ini adalah "*`Node`*`1`", yang akan digunakan sebagai identitas saat mengirim atau menerima pesan antar *Node*.

```
DHT dht(DHTPIN, DHTTYPE);
BLEAdvertising* pAdvertising;
BLEScan* pBLEScan;
```

Objek dht digunakan untuk mengakses pembacaan suhu dan kelembaban dari sensor DHT22. Pointer `pAdvertising` dan `pBLEScan` masing-masing berfungsi untuk proses pengiklanan (*advertising*) dan pemindaian (*scanning*) pada fitur BLE.

```
String lastSentMessage = "";
String lastRelayedMessage = "";
String pendingRelayMsg = "";
```

Variabel-variabel ini menyimpan pesan terakhir yang sudah dikirim, pesan terakhir yang sudah diteruskan ke *Node* lain, serta pesan yang masih menunggu untuk dikirim ulang atau di*relay*.

```
unsigned long lastSend = 0;
unsigned long lastScanRestart = 0;
unsigned long relayStartTime = 0;
bool isRelaying = false;
```

Variabel-variabel tersebut berfungsi untuk mencatat waktu pengiriman terakhir, waktu *restart scanning* terakhir, waktu mulai proses *relay*, dan status apakah sedang melakukan *relay* atau tidak. Semua nilainya dinyatakan dalam milidetik menggunakan fungsi *millis*().

```
unsigned long lastReceivedTime_Node1 = 0;
unsigned long lastReceivedTime_Node2 = 0;
unsigned long lastReceivedTime_Node3 = 0;
unsigned long lastReceivedTime_Node4 = 0;
```

Variabel-variabel ini menyimpan waktu terakhir saat *Node* menerima pesan dari masing-masing *Node* dalam jaringan BLE *Mesh*. Tujuannya adalah untuk mengukur *delay* atau jeda waktu antar pengiriman.

```
class MyAdvertisedDeviceCallbacks : public BLEAdvertisedDeviceCallbacks {
  void onResult(BLEAdvertisedDevice advertisedDevice) {
```

Kode ini merupakan pembuatan kelas turunan dari `BLEAdvertisedDeviceCallbacks` yang digunakan untuk memproses pesan-pesan BLE yang diterima saat proses pemindaian aktif.

Fungsi `onResult()` akan dipanggil secara otomatis setiap kali ada perangkat BLE yang ditemukan.

```
String msg = advertisedDevice.getManufacturerData();
```

Baris ini mengambil data yang dikirim oleh perangkat BLE lain. Data tersebut diambil dari bagian `manufacturer` data dalam bentuk string.

```
if (msg.startsWith("Node") && !msg.startsWith(NODE_ID)) {
```

Kondisi ini memastikan bahwa hanya pesan dari *Node* lain yang akan diproses. Jika pesan tersebut dimulai dengan kata "*Node*" dan bukan dari dirinya sendiri (`bukan NODE_ID milik Node saat ini`), maka program akan melanjutkan membaca dan merespons pesan tersebut.

```
if (msg != lastRelayedMessage) {
```

Pesan akan diproses jika memang berbeda dari pesan terakhir yang pernah diteruskan. Hal ini berguna untuk mencegah pengulangan pesan yang sama.

```
Serial.print("📥 Pesan diterima: ");
Serial.println(msg);
```

Program akan mencetak pesan yang baru diterima ke serial monitor agar dapat dipantau oleh pengguna melalui komputer.

```
int totalBits = msg.length() * 8;
Serial.print("📊 Total bit: ");
Serial.println(totalBits);
```

Kode ini menghitung jumlah total bit dari pesan yang diterima. Karena setiap karakter dalam string terdiri dari 8 bit, maka panjang pesan dikalikan 8.

```
lastRelayedMessage = msg;
pendingRelayMsg = msg;
relayStartTime = millis();
isRelaying = true;
```

Pesan disimpan sebagai pesan terakhir yang di*relay*, ditandai sebagai pesan yang sedang menunggu untuk dikirim ulang, waktu dimulainya *relay* dicatat, dan status *relay* diaktifkan.

```
String sender = msg.substring(0, 5);
unsigned long now = millis();
```

Dua baris ini digunakan untuk mengetahui pengirim pesan dari *Node* mana dan mencatat waktu sekarang agar bisa dibandingkan dengan waktu sebelumnya.

```
if (sender == "Node1") {
  Serial.print("⏱ Delay dari Node1: ");
  Serial.print(now - lastReceivedTime_Node1);
  Serial.println(" ms");
  lastReceivedTime_Node1 = now;
} else if (sender == "Node2") {
  Serial.print("⏱ Delay dari Node2: ");
  Serial.print(now - lastReceivedTime_Node2);
  Serial.println(" ms");
  lastReceivedTime_Node2 = now;
} else if (sender == "Node3") {
  Serial.print("⏱ Delay dari Node3: ");
  Serial.print(now - lastReceivedTime_Node3);
  Serial.println(" ms");
  lastReceivedTime_Node3 = now;
} else if (sender == "Node4") {
  Serial.print("⏱ Delay dari Node4: ");
  Serial.print(now - lastReceivedTime_Node4);
  Serial.println(" ms");
  lastReceivedTime_Node4 = now;
}
```

Bagian ini mengecek asal pengirim, apakah dari *Node*1, *Node*2, dan seterusnya. *Delay* dihitung dari selisih waktu sekarang dengan waktu terakhir pesan diterima dari *Node* tersebut.

```
for (int i = 0; i < 3; i++) {
  digitalWrite(LED_PIN, HIGH);
  delay(100);
  digitalWrite(LED_PIN, LOW);
  delay(100);
}
```

LED indikator akan berkedip sebanyak tiga kali untuk memberi tanda bahwa pesan dari *Node* lain berhasil diterima dan sedang diproses untuk *relay*.

```
void setup() {
  Serial.begin(115200);
  Serial.println("🚀 Program dimulai...");
 pinMode(LED_PIN, OUTPUT);
 dht.begin();
```

Fungsi setup digunakan untuk mengatur konfigurasi awal. Komunikasi serial dimulai dengan baudrate `115200`, pin LED diatur sebagai `output`, dan sensor DHT22 diinisialisasi agar siap membaca suhu dan kelembaban.

```
BLEDevice::init(NODE_ID);
```

Baris ini menginisialisasi BLE dan mengatur ID *Node* sesuai dengan nilai *`NODE_ID`*.

```
pBLEScan = BLEDevice::getScan();
pBLEScan->setAdvertisedDeviceCallbacks(new MyAdvertisedDeviceCallbacks());
pBLEScan->setActiveScan(true);
pBLEScan->start(0, nullptr);  // terus scan
```

Proses scanning BLE dimulai dengan mode aktif. `Callback` yang dibuat sebelumnya dihubungkan agar setiap pesan BLE yang masuk bisa langsung diproses.

```
pAdvertising = BLEDevice::getAdvertising();
pAdvertising->setScanResponse(false);
pAdvertising->setMinPreferred(0x06);
pAdvertising->setMinPreferred(0x12);
```

Proses iklan BLE (`advertising`) diinisialisasi dengan pengaturan tertentu agar bisa disebarkan ke *Node-Node* lain dalam jaringan.

```
void loop() {
  unsigned long now = millis();
```

Fungsi `loop` akan berjalan terus-menerus selama ESP32 menyala. Pada awal fungsi `loop`, waktu saat ini disimpan ke dalam variabel `now` untuk digunakan dalam perhitungan waktu.

```
if (now - lastScanRestart > 10000) {
  pBLEScan->stop();
  delay(100);
  pBLEScan->start(0, nullptr);
  lastScanRestart = now;
}
```

Bagian ini menjalankan restart proses scanning BLE setiap 10 detik untuk memastikan proses pemindaian tetap berjalan stabil.

```
if (now - lastSend > 5000) {
```

Setiap 5 detik, *Node* akan membaca data sensor suhu, kelembaban udara, dan kelembaban tanah.

```
float temp = dht.readTemperature();
float hum = dht.readHumidity();
int soil = analogRead(SOIL_PIN);
```

Sensor DHT22 digunakan untuk membaca suhu dan kelembaban udara, sedangkan sensor kelembaban tanah dibaca dari pin analog dan menghasilkan nilai `integer`.

```
if (isnan(temp) || isnan(hum)) {
  Serial.println("⚠ Gagal membaca sensor DHT22!");
} else {
```

Jika pembacaan sensor suhu atau kelembaban tidak valid (NaN), maka pesan peringatan akan muncul di serial monitor. Jika pembacaan valid, maka proses pengiriman dilanjutkan.

```
String message = String(NODE_ID) + ":T=" + String(temp, 1) + ",H=" + String(hum, 1) + ",S=" + String(soil);
```

Data yang dibaca dikemas menjadi sebuah string dengan format seperti "*Node*1:T=28.3,H=60.1,S=850" agar bisa dibaca oleh *Node* lain.

```
if (message != lastSentMessage) {
```

Pesan akan dikirim hanya jika isinya berbeda dari pesan sebelumnya, agar tidak terjadi pengiriman duplikat

```
lastSentMessage = message;
Serial.print("📤 Mengirim: ");
Serial.println(message);
```

Pesan tersebut dicatat sebagai pesan terakhir yang dikirim, lalu ditampilkan di serial monitor.

```
BLEAdvertisementData data;
data.setManufacturerData(message);
pAdvertising->setAdvertisementData(data);
```

Data dikonversi menjadi data BLE advertising agar bisa dikirim ke *Node-Node* lain di sekitarnya.

```
pBLEScan->stop();
pAdvertising->start();
delay(200);
pAdvertising->stop();
pBLEScan->start(0, nullptr);
```

Agar pengiriman berjalan lancar, pemindaian BLE dihentikan sementara saat mengirim data, kemudian dilanjutkan kembali setelah pengiriman selesai.

```
digitalWrite(LED_PIN, HIGH);
delay(200);
digitalWrite(LED_PIN, LOW);
```

LED indikator menyala sebentar sebagai tanda bahwa pengiriman data berhasil dilakukan.

```
  lastSend = now;
}
```

Waktu terakhir pengiriman diperbarui untuk referensi pengiriman berikutnya.

```
if (isRelaying && millis() - relayStartTime > 200) {
  Serial.println("🔁 Relay ulang pesan ke node lain");
```

Jika terdapat pesan dari *Node* lain yang perlu diteruskan dan telah melewati 200 milidetik sejak dideteksi, maka proses *relay* dilakukan.

```
BLEAdvertisementData relayData;
relayData.setManufacturerData(pendingRelayMsg);
pAdvertising->setAdvertisementData(relayData);
```

Pesan yang diterima dari *Node* lain dimasukkan kembali ke dalam format BLE `advertising` untuk dikirim ulang.

```
pBLEScan->stop();
pAdvertising->start();
delay(5000);
pAdvertising->stop();
pBLEScan->start(0, nullptr);
```

Proses *scanning* dihentikan sejenak, pesan dikirim, kemudian *scanning* dilanjutkan. Waktu *delay* pengiriman *relay* diperpanjang menjadi 5 detik agar data tersebar dengan baik.

```
    isRelaying = false;
  }
}
```

Setelah *relay* selesai, status *relay* dinonaktifkan untuk mencegah pengiriman ulang yang tidak perlu.

3.3.3 Identifikasi Masalah

Dalam tahapan ini dilakukan penelusuran tentang apa saja yang dilakukan untuk merancang Jaringan Sensor Berbasis *Bluetooth Mesh* menggunakan Mikrokontroler ESP32.

1. **Konsep Sistem Sensor Berbasis *Bluetooth Mesh***

   Bluetooth Mesh adalah sistem komunikasi *many-to-many*, artinya setiap *Node* bisa terhubung dan meneruskan data ke *Node* lain. Tidak ada satu pusat komunikasi, jadi ini cocok untuk jaringan sensor yang luas dan bisa meng-*cover* area yang lebih besar.

2. **Struktur sistem** 4 ***Node*** **ESP**32

Setiap ESP32 berfungsi sebagai *Node* dalam jaringan *mesh* dan memiliki

- 1 Sensor DHT22
- 1 *Soil Moisture* Sensor

Tiap *Node* saling bertukar *relay* data secara *mesh*, masing-masing *Node* berfungsi sebagai *gateway*/penerima utama (bisa dikirim ke *cloud* atau ditampilkan data di *dashboard local*).

3. **Port koneksi per ESP32**

Konfigurasi pin untuk setiap modul sensor pada perangkat ESP32 dapat dilihat pada Tabel 3.3 dan Gambar 3.2. Setiap *Node* ESP32 dalam jaringan dilengkapi dengan dua sensor: sebuah sensor DHT22 untuk mengukur suhu dan kelembaban udara, serta sebuah sensor soil moisture (kelembaban tanah) untuk mengukur kadar air dalam media tanam. Pemetaan pin untuk setiap sensor telah ditetapkan secara konsisten pada semua modul untuk memudahkan proses pembuatan program (*coding*) dan pemeliharaan sistem. Seperti yang terlihat pada tabel, pin data sensor DHT22 terhubung ke GPIO 4, sedangkan pin data sensor soil moisture memanfaatkan pin GPIO 34 yang memiliki kemampuan *Analog-to-Digital Converter* (ADC). Kedua sensor mendapatkan catu daya yang sama, yaitu 5V dan tentu saja berbagi jalur *Ground* (GND) yang sama pada board ESP32.

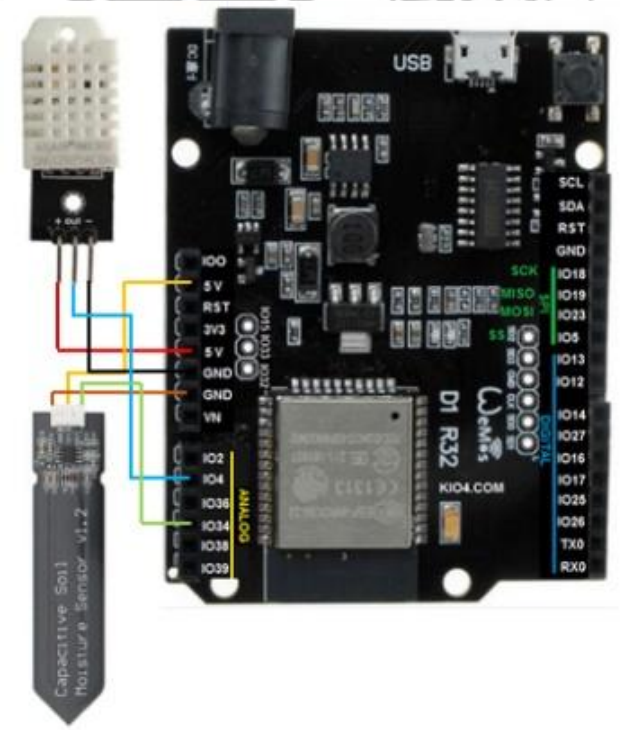
Tabel 3. 3 Port Koneksi

Sensor	Pin Data	Tegangan	*Ground*
DHT22	GPIO 4	5 V	GND
Soil Moisture	GPIO 34	5 V	GND

Masing masing ESP32 memiliki :

- DHT22 pada GPIO 4
- *Soil Moisture* pada GPIO 34

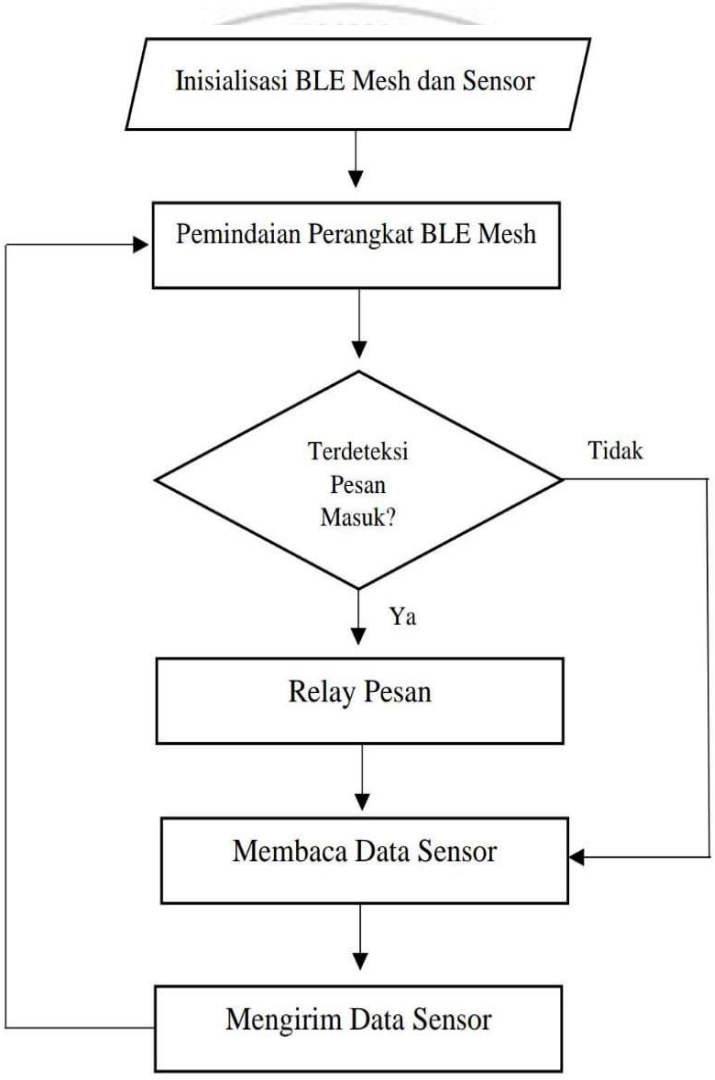
Gambar 3.2 menunjukkan pengkabelan dengan cara menghubungkan *Node* menggunakan board ESP32 yang terpasang dua sensor, yaitu sensor DHT22 untuk mengukur suhu dan kelembaban udara serta sensor kelembaban tanah untuk mengetahui kadar air di tanah. Hasil pembacaan dari setiap *Node* dikirim ke *Node* lain, lalu diteruskan melalui sistem *relay* agar informasi tetap diterima antar *Node*. Dengan metode ini, komunikasi berbasis *Bluetooth* Mesh dapat memastikan data dari semua *Node* dapat terkumpul dan sampai ke ESP32 tujuan secara stabil.



Gambar 3.2 Pengkabelan *Node*

3.4 Pengujian Alat

Setelah alat selesai dibuat dengan Arduino IDE, kemudian dilakukan pengujian alat. Pengujian alat dilakukan pada sampel tanah yang telah disiapkan. Adapun sistem kerja alat sensor berbasis *Bluetooth Mesh* dapat dilihat pada Gambar 3.3.



Gambar 3. 3 *Flowchart* Sistem Kerja Alat

Proses diawali saat perangkat dinyalakan, yang menandai dimulainya seluruh sistem dan program utama mulai berjalan.

Langkah pertama yang dilakukan adalah inisialisasi. Pada tahap ini, ESP32 mempersiapkan semua komponen yang dibutuhkan, seperti komunikasi *Bluetooth Low Energy* (BLE), sensor suhu dan kelembaban DHT22, serta sensor kelembaban tanah. Tujuan dari inisialisasi ini adalah agar semua sensor dan sistem komunikasi dapat berfungsi dengan baik saat proses monitoring dimulai.

Setelah inisialisasi selesai, ESP32 masuk ke tahap pemindaian sinyal BLE. Proses ini dilakukan untuk mendeteksi apakah terdapat *Node* lain yang sedang mengirimkan data di dalam jaringan. Pemindaian ini menjadi kunci dalam komunikasi antar *Node* dalam jaringan *mesh*.

Jika dari hasil pemindaian ditemukan adanya pesan masuk dari *Node* lain, maka perangkat akan memeriksa apakah pesan tersebut baru atau sudah pernah diterima sebelumnya. Jika pesan tersebut belum pernah diterima, maka ESP32 akan menyimpannya dan melakukan pengiriman ulang (*relay*) pesan tersebut agar dapat diteruskan ke *Node* lainnya. Proses ini penting dalam menjaga jangkauan komunikasi dan penyebaran data di dalam jaringan.

Namun, jika tidak ada pesan masuk yang terdeteksi, maka sistem akan beralih ke proses pembacaan sensor. Pada tahap ini, ESP32 membaca data suhu dan kelembaban udara dari sensor DHT22 serta data kelembaban tanah dari sensor analog. Data tersebut kemudian dikemas dalam format iklan data (BLE *advertisement*) dan dikirimkan melalui BLE agar dapat diterima oleh *Node-Node* lain yang sedang melakukan pemindaian.

Setelah pengiriman data selesai atau setelah proses *relay* dilakukan, sistem akan kembali ke tahap pemindaian dan mengulang seluruh proses secara berkelanjutan. Dengan alur ini, ESP32 dapat terus melakukan pembacaan sensor, pertukaran data, serta menjaga kestabilan komunikasi antar *Node* di dalam jaringan *mesh*. Sistem akan berjalan secara *real-time* dan terus aktif selama perangkat dalam keadaan hidup.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil

Hasil dari rancang bangun monitoring kelembaban dan suhu berbasis *Bluetooth Mesh* ini diperoleh setelah perangkat diuji secara langsung. Pengujian dilakukan pada sistem yang terdiri dari empat *Node* ESP32, masing-masing dilengkapi dengan sensor suhu dan kelembaban udara (DHT22) serta sensor kelembaban tanah.

Setiap *Node* berhasil membaca data lingkungan di sekitarnya, lalu mengirimkan data tersebut melalui komunikasi *Bluetooth Mesh*. Hasil pengujian menunjukkan bahwa alat dapat bekerja sesuai dengan fungsi yang dirancang, yaitu membaca data sensor, mengirimkan data antar *Node*, dan menampilkannya di Serial Monitor.

Untuk menunjang sistem, digunakan catu daya berupa adaptor 5V yang terhubung langsung ke port USB ESP32. Selama pengujian, seluruh *Node* dapat menyala dan bekerja dengan stabil tanpa gangguan daya.

Data suhu, kelembaban udara, dan kelembaban tanah dari tiap *Node* berhasil diterima dan ditampilkan pada *Node* pusat. Proses pengiriman dan penerimaan data juga disertai dengan LED indikator yang menyala sebagai penanda aktivitas komunikasi.

1. Data Setiap *Node* (Tanpa Jarak)

Berikut merupakan hasil data pengujian yang diperoleh dari setiap *Node*:

Data *Node* 1

Data hasil pengujian yang tercatat pada serial monitor untuk *Node* 1 selama periode pengujian disajikan secara lengkap dalam Tabel 4.1. Data ini merekam seluruh aktivitas penerimaan paket data oleh *Node* 1.

Tabel 4. 1 Data Serial Monitor *Node* 1

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	22:10:29	*Node* 3	31	77.7	2592	208	757	Ya
2	22:10:49	*Node* 4	31.3	63.2	387	200	21205	Ya
3	22:11:00	*Node* 4	31.4	63.5	383	200	10185	Ya
4	22:11:10	*Node* 4	31.4	63.3	383	200	9976	Ya
5	22:11:15	*Node* 2	31.5	66.7	1998	208	46829	Ya
6	22:11:25	*Node* 2	31.5	68.8	2000	208	9897	Ya
7	22:11:30	*Node* 3	31.1	78.1	2599	208	61487	Ya
8	22:11:36	*Node* 4	31.5	65.1	373	200	26316	Ya
9	22:11:41	*Node* 2	31.5	68	2002	208	16483	Ya
10	22:11:47	*Node* 4	31.5	63.5	383	200	11305	Ya
11	22:11:57	*Node* 4	31.5	62.8	379	200	9918	Ya
12	22:12:03	*Node* 2	31.5	67.7	2000	208	21186	Ya
13	22:12:12	*Node* 2	31.5	67.4	2007	208	9773	Ya
14	22:12:18	*Node* 3	31.1	79.4	2599	208	47906	Ya
15	22:12:28	*Node* 3	31.1	78.3	2597	208	9963	Ya
16	22:12:34	*Node* 2	31.5	65.2	2013	208	21452	Ya

Data *Node* 2

Tabel 4. 2 Data Serial Monitor *Node* 2

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	22:10:29	*Node* 3	31	77.7	2592	208	771	Ya
2	22:10:35	*Node* 1	31.4	64	2431	208	6878	Ya
3	22:10:41	*Node* 3	31	77.7	2592	208	11642	Ya
4	22:10:46	*Node* 1	31.4	63.3	2522	208	10984	Ya
5	22:10:52	*Node* 4	31.3	63.2	387	200	23391	Ya
6	22:10:57	*Node* 1	31.4	63.3	2522	208	11010	Ya
7	22:11:03	*Node* 4	31.4	63.5	383	200	11036	Ya
8	22:11:10	*Node* 4	31.4	63.3	383	200	6953	Ya
9	22:11:35	*Node* 3	31.1	78.5	2599	208	54491	Ya
10	22:11:40	*Node* 4	31.5	65.1	373	200	30969	Ya
11	22:11:47	*Node* 4	31.5	63.5	383	200	6718	Ya
12	22:11:57	*Node* 4	31.5	62.8	379	200	9886	Ya
13	22:12:18	*Node* 3	31.1	79.4	2599	208	42607	Ya
14	22:12:28	*Node* 3	31.1	78.3	2597	208	9997	Ya
15	22:12:33	*Node* 4	31.6	63.8	375	200	36062	Ya

Data hasil pengujian dari serial monitor *Node* 2 selama periode yang sama disajikan dalam Tabel 4.2. Data ini memperlihatkan peran *Node* 2 sebagai *Relay Node* yang aktif dalam jaringan.

Data *Node* 3

Data hasil pengujian yang tercatat pada *serial monitor* untuk *Node* 3 selama periode pengujian disajikan secara lengkap dalam Tabel 4.3. Data ini mencatat seluruh aktivitas penerimaan dan penerusan paket data oleh *Node* 3, yang berperan sebagai *Relay Node* sekaligus Sensor *Node* dalam arsitektur jaringan.

Tabel 4. 3 Data Serial Monitor *Node* 3

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	22:10:34	*Node* 1	31.4	64	2531	208	5241	Ya
2	22:10:44	*Node* 1	31.4	63.3	2522	208	9883	Ya
3	22:10:50	*Node* 4	31.3	63.2	387	200	21198	Ya
4	22:10:55	*Node* 1	31.4	63.3	2522	208	11326	Ya
5	22:11:01	*Node* 4	31.4	63.5	383	200	10809	Ya
6	22:11:10	*Node* 4	31.4	63.3	383	200	9374	Ya
7	22:11:16	*Node* 2	31.5	66.7	1998	208	46985	Ya
8	22:11:25	*Node* 2	31.5	68.8	2000	208	8864	Ya
9	22:11:35	*Node* 2	31.5	68	2002	208	10057	Ya
10	22:11:41	*Node* 4	31.5	65.1	373	200	30494	Ya
11	22:11:46	*Node* 2	31.5	68	2002	208	11401	Ya
12	22:11:52	*Node* 4	31.5	63.5	383	200	10970	Ya
13	22:11:58	*Node* 4	31.5	62.8	379	200	5841	Ya
14	22:12:04	*Node* 2	31.5	67.7	2000	208	17237	Ya
15	22:12:12	*Node* 2	31.5	67.4	2007	208	8748	Ya
16	22:12:33	*Node* 2	31.5	65.2	2013	208	20594	Ya

Data *Node* 4

Tabel 4.4 menunjukkan bahwa *Node* 4 aktif menerima dan meneruskan data dari *Node* lainnya, termasuk *Node* 1, *Node* 2, dan *Node* 3.

Kolom *Relay* yang bernilai "Ya" pada seluruh atau sebagian besar entri mengonfirmasi bahwa *Node* 4 berfungsi sebagai *relay* yang membantu memperluas jangkauan jaringan.

Tabel 4. 4 Data Serial Monitor *Node* 4

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	22:10:31	*Node* 3	31	77.7	2592	208	879	Ya
2	22:10:37	*Node* 1	31.4	64	2531	208	6797	Ya
3	22:10:44	*Node* 1	31.4	63.3	2522	208	7026	Ya
4	22:11:20	*Node* 2	31.5	66.7	1998	208	49299	Ya
5	22:11:26	*Node* 2	31.5	68.8	2000	208	5934	Ya
6	22:11:31	*Node* 3	31.1	78.5	2599	208	59941	Ya
7	22:11:37	*Node* 2	31.5	68	2002	208	11275	Ya
8	22:12:06	*Node* 2	31.5	67.7	2000	208	29658	Ya
9	22:12:13	*Node* 2	31.5	67.4	2007	208	6496	Ya
10	22:12:19	*Node* 3	31.1	79.4	2599	208	47793	Ya
11	22:12:28	*Node* 3	31.1	78.3	2597	208	8881	Ya
12	22:12:34	*Node* 2	31.5	65.2	2013	208	20787	Ya

2. Percobaan Dengan Jarak

Hasil pengujian performa jaringan dengan konfigurasi jarak 2 meter antar *Node* disajikan dalam Tabel 4.5. Pengujian ini dilakukan untuk mengevaluasi kinerja sistem dalam skenario jarak pendek dengan kondisi *line-of-sight* yang optimal.

Tabel 4. 5 Data Uji Coba pada Jarak 2 Meter

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	00:36:24	*Node* 3	27.9	78.6	2525	208	13618	Ya
2	00:37:01	*Node* 4	27.3	83.6	493	200	49827	Ya
3	00:37:11	*Node* 1	28.7	74.1	2683	208	60423	Ya
4	00:38:00	*Node* 4	27.4	83.5	499	200	59020	Ya
5	00:38:29	*Node* 3	27.9	78.7	2525	208	124211	Ya
6	00:38:37	*Node* 3	27.9	78.4	2525	208	8551	Ya
7	00:38:50	*Node* 4	27.4	83.1	476	200	50652	Ya
8	00:39:08	*Node* 1	28.7	73.8	2672	208	116395	Ya

3. Skenario *Node* 1 Mati

Pengujian pada tabel 4.6, skenario dengan *Node* 2 sebagai *gateway* *Node* 1 mati dilakukan untuk mengevaluasi ketahanan (*resilience*) dan kemampuan adaptasi jaringan *Bluetooth Mesh* dalam menghadapi kegagalan salah satu *Node* utama. Dalam skenario ini, *Node* 1 dinonaktifkan untuk mensimulasikan kondisi kegagalan *Node*.

Tabel 4. 6 Data Skenario salah satu *Node* dinonaktifkan

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)	*Relay* (Ya/Tidak)
1	00:50:23	*Node* 3	27.5	80.6	2503	208	2793	Ya
2	00:50:33	*Node* 3	27.6	80.4	2512	208	10469	Ya
3	00:50:43	*Node* 3	27.6	79.7	2507	208	9610	Ya
4	00:51:03	*Node* 4	27	83.5	482	200	42592	Ya
5	00:51:13	*Node* 4	27.1	83.7	485	200	10517	Ya

4. Data *Node* Tanpa *Relay*

Pengujian skenario tanpa fungsi *relay* dilakukan untuk mengevaluasi performa jaringan *Bluetooth Mesh* ketika fungsi penerusan paket (*message relaying*) dinonaktifkan pada seluruh *Node*.

Tabel 4. 7 Uji Coba Tanpa *Relay*

No	Timestamp	Diterima dari	Suhu (°C)	Kelembaban Udara (%)	Kelembaban Tanah	Ukuran Data (bit)	Delay (ms)
1	01:32:56	*Node* 3	27.2	85.4	220	200	1086
2	01:32:57	*Node* 1	28.3	72.9	2675	208	2629
3	01:33:01	*Node* 3	27.1	85.3	211	200	5108
4	01:33:02	*Node* 1	28.3	72.8	2671	208	5002
5	01:33:06	*Node* 3	27.2	85	208	200	5071
6	01:33:07	*Node* 1	28.3	72.8	2673	208	5092
7	01:33:12	*Node* 1	28.3	72.7	2671	208	5002
8	01:33:16	*Node* 3	27.2	84.9	201	200	5041
9	01:33:17	*Node* 1	28.3	72.6	2673	208	5051
10	01:33:26	*Node* 3	27.2	85.5	200	200	9928
11	01:33:26	*Node* 1	28.3	72.6	2672	208	9999
12	01:33:28	*Node* 4	29.1	70.4	2755	208	33427

Dalam skenario ini, setiap *Node* hanya mengirimkan data sensornya sendiri dan tidak meneruskan paket dari *Node* lain, mengubah jaringan *mesh* menjadi komunikasi *point-to-multipoint* dimana semua *Node* berkomunikasi secara langsung dengan setiap node masih berfungsi sebagai gateway seperti pada tabel 4.7.

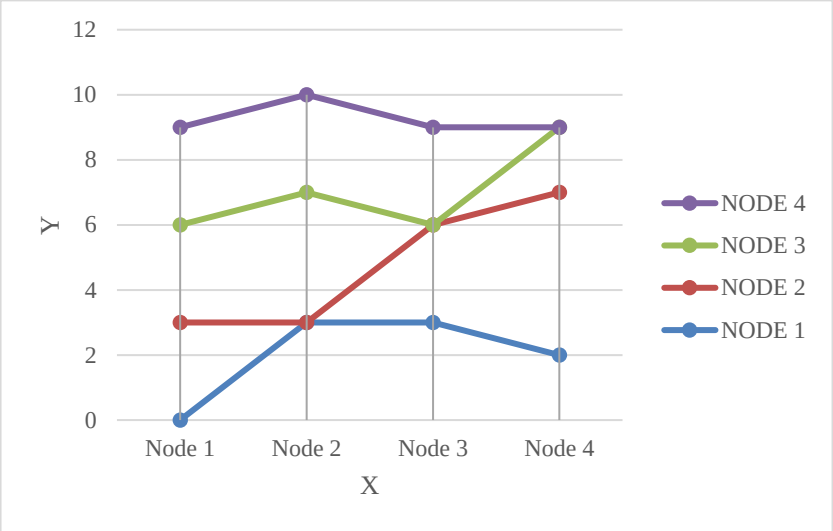
4.2 Analisa Data

1. Jumlah data yang diterima setiap *Node*

Tabel 4.8 menyajikan analisis komprehensif mengenai pola komunikasi dan konektivitas antar *Node* dalam jaringan *Bluetooth Mesh* yang diimplementasikan. Tabel ini merepresentasikan matriks konektivitas yang menunjukkan jumlah paket data yang berhasil dikirimkan dari *Node* sumber (baris) ke *Node* tujuan (kolom) selama periode pengujian.

Tabel 4.8 Jumlah Data ditiap *Node*

Node	***Node*** 1	***Node*** 2	***Node*** 3	***Node*** 4
NODE 1	0	3	3	2
NODE 2	3	0	3	5
NODE 3	3	4	0	2
NODE 4	3	3	3	0



Gambar 4. 1 Grafik Jumlah Data Tiap *Node*

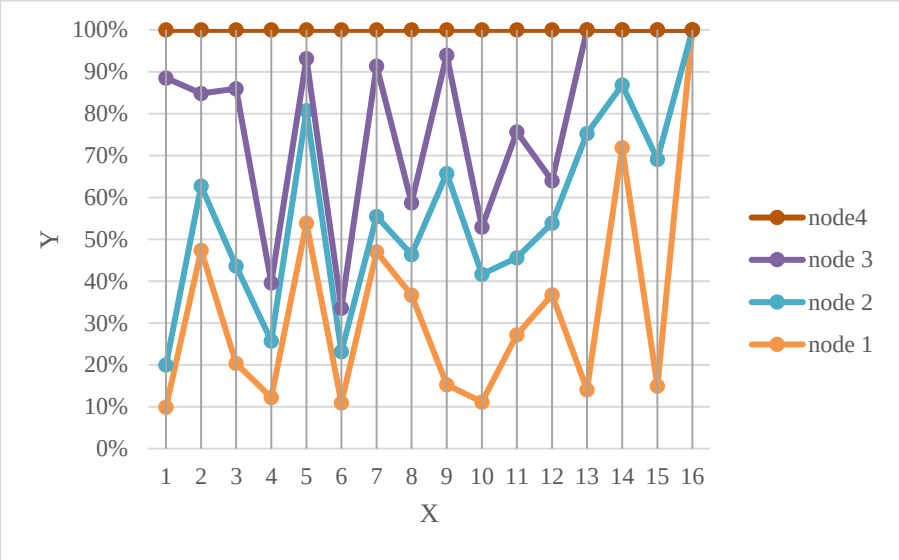
Grafik pada Gambar 4.1 menunjukkan berapa banyak data yang diterima oleh setiap *Node* sesuai dengan Tabel 4.8. Sumbu X mewakili identitas *Node* pengirim (*Node* 1 hingga *Node* 4), sedangkan sumbu Y menunjukkan jumlah data yang diterima oleh *Node* lainnya. Dari grafik tersebut terlihat bahwa *Node* 4 menerima data paling banyak dan aliran datanya konsisten hampir di semua *Node*, sedangkan *Node* 1 cenderung menerima data paling sedikit. Hal ini menunjukkan ada perbedaan dalam distribusi dan tingkat aktivitas komunikasi antar *Node* dalam jaringan. Hal ini mencerminkan inti dari *Bluetooth Mesh*, yakni tiap *Node* bisa berfungsi sebagai pengirim, penerima, atau *relay*. Tampak jelas bahwa seluruh *Node* saling berinteraksi, dengan tingkat komunikasi yang relatif seragam, menandakan keberadaan koneksi *mesh* yang komprehensif dan redundansi jalur yang memadai. *Node* tidak semata bertukar data secara langsung, tetapi juga berperan meneruskan pesan dari *Node* lain, membuktikan bahwa sistem beroperasi selaras dengan prinsip *Bluetooth Mesh* yang mengandalkan *relay* antar perangkat guna menjangkau *Node* yang lokasinya lebih jauh. Visualisasi ini menjadi bukti konkrit bahwa topologi jaringan bersifat adaptif dan dinamis, yang mana memungkinkan komunikasi untuk tetap berjalan meskipun terdapat gangguan pada suatu jalur tertentu. Dengan begitu, grafik tersebut menegaskan bahwa jaringan yang telah dibangun telah mengimplementasikan *Bluetooth Mesh* secara efektif.

2. *Delay* per *Node*

Analisis performa temporal pada Tabel 4.9 menunjukkan variasi *delay* yang signifikan antar *Node*, dengan nilai terendah 757 ms pada *Node* 1 dan nilai tertinggi 69.941 ms pada *Node* 4. *Node* 1 menunjukkan *latency* terbaik secara konsisten, sedangkan *Node* 4 mengalami fluktuasi *delay* paling besar. Variasi ini disebabkan oleh beberapa faktor termasuk perbedaan jumlah *hop* antar *Node*, fluktuasi kualitas sinyal nirkabel, proses retransmisi paket, dan distribusi beban *traffic* yang tidak merata pada setiap *path*.

Tabel 4. 9 Delay tiap *Node*

Delay per *Node* Pengirim			
Node 1	***Node*** 2	***Node*** 3	***Node***4
757	771	5241	879
21205	6878	9883	6797
10185	11642	21198	7026
9976	10984	11326	49299
46829	23391	10809	5934
9897	11010	9374	59941
61487	11036	46985	11275
26316	6953	8864	29658
16483	54491	30494	6496
11305	30969	11401	47793
9918	6718	10970	8881
21186	9886	5841	20787
9773	42607	17237	
47906	9997	8748	
9963	36062	20594	
21452			



Gambar 4. 2 Grafik Variasi Delay Tiap Data Diterima

Grafik pada Gambar 4.2 menunjukkan perbedaan waktu penundaan (*delay*) untuk setiap data yang diterima oleh *Node*, berdasarkan data di Tabel 4.9. Sumbu X menunjukkan urutan pengiriman data (total 16 kali percobaan), sedangkan sumbu Y menunjukkan nilai *delay* yang sudah diubah ke dalam bentuk persentase (%). Dari grafik terlihat bahwa *Node* 1 memiliki *delay* terkecil, dengan rata-rata berkisar antara 9.963 ms hingga 21.452 ms, sedangkan *Node* 2 memiliki *delay* tertinggi dengan rata-rata mencapai 36.062 ms. *Node* 3 berada di tengah dengan rata-rata *delay* sekitar 20.594 ms, sedangkan *Node* 4 relatif konsisten dengan rata-rata *delay* di bawah 15.000 ms. Pola ini menunjukkan bahwa ada perbedaan signifikan antar *Node*, dimana *Node* 2 cenderung paling lambat menerima data, sementara *Node* 1 paling cepat.

Dari data tersebut, terlihat bahwa setiap *Node* mengalami variasi *delay* saat mengirim data ke *Node* lain. *Node* 1 menunjukkan *delay* rendah di awal, namun meningkat tajam pada beberapa titik, kemungkinan karena adanya penambahan *hop* atau kepadatan jaringan. *Node* 2 dan *Node* 3 memiliki *delay* yang fluktuatif namun masih dalam batas wajar, menandakan jalur komunikasi yang cukup stabil. Sementara itu, *Node* 4 menunjukkan *delay* yang lebih tinggi dan bervariasi, terutama saat mengirim ke *Node* 2 dan *Node* 3, yang disebabkan oleh jalur komunikasi yang memerlukan lebih banyak *hop* atau adanya gangguan sinyal. Tidak ada satu *Node* pun yang selalu memiliki *delay* paling rendah atau paling tinggi, yang menunjukkan bahwa sistem *Bluetooth Mesh* bekerja secara dinamis dan menyesuaikan diri dengan kondisi jaringan. Variasi *delay* ini mengindikasikan bahwa komunikasi antar *Node* tidak selalu berlangsung secara langsung, melainkan sering melibatkan *Node* perantara sesuai dengan prinsip kerja jaringan *mesh*.

3. Jumlah *Relay* per *Node*

Data pada tabel 4.10 menunjukkan jumlah *relay* yang dilakukan oleh masing-masing *Node* dalam jaringan *Bluetooth Mesh*. *Node*1 memiliki jumlah *relay* terbanyak, yaitu 16 kali, diikuti oleh *Node*2 dan *Node*3 sebanyak 15 kali,

serta *Node*4 sebanyak 12 kali. Hal ini menunjukkan bahwa *Node*1 lebih aktif dalam meneruskan data, sedangkan *Node*4 cenderung lebih pasif.

Tabel 4. 10 *Relay* pada setiap *Node*

Node	**Jumlah *Relay***
NODE 1	16
NODE 2	15
NODE 3	15
NODE 4	12

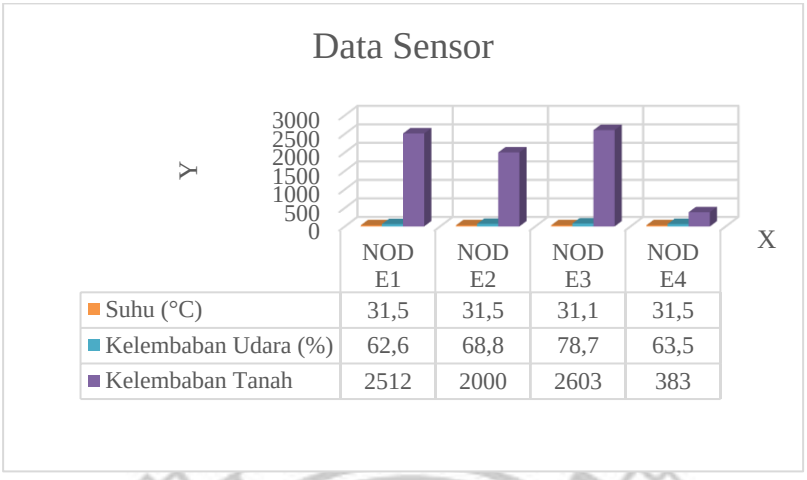
4. Data Sensor Per *Node*

Data sensor dari keempat *Node* pada Tabel 4.11 menunjukkan variasi nilai parameter lingkungan yang signifikan, khususnya pada kelembaban tanah (*Node* 4: 383, *Node* 3: 2603). Variasi ini membuktikan bahwa sistem *Bluetooth Mesh* berhasil mengakuisisi data dari lokasi yang berbeda secara simultan. Konsistensi nilai suhu (31.1-31.5°C) antar *Node* menunjukkan reliabilitas pengukuran, sementara perbedaan kelembaban udara (62.6-78.7%) mencerminkan kondisi mikro lingkungan yang berbeda di setiap titik *Node*.

Grafik pada Gambar 4.3 menampilkan hasil bacaan sensor di setiap *Node* berdasarkan Tabel 4.11. Sumbu X menunjukkan masing-masing *Node* (*Node* 1 sampai *Node* 4), sedangkan sumbu Y menunjukkan nilai hasil pengukuran sensor, yaitu suhu dalam satuan derajat Celsius, kelembaban udara dalam persen, dan kelembaban tanah dalam satuan ADC. Dari data tersebut terlihat suhu di semua *Node* relatif stabil, berkisar antara 31 hingga 31,5°C.

Tabel 4. 11 Data Sensor masing-masing *Node*

Node	**Suhu (°C)**	**Kelembaban Udara (%)**	**Kelembaban Tanah**
*NODE*1	31.5	62.6	2512
*NODE*2	31.5	68.8	2000
*NODE*3	31.1	78.7	2603
*NODE*4	31.5	63.5	383



Gambar 4. 3 Grafik Data Setiap Sensor

Kelembaban udara terbesar ditemukan pada *Node* 3 sebesar 78,7%, sedangkan kelembaban tanah tertinggi juga ada di *Node* 3 dengan nilai 2603, sementara nilai terendah terjadi di *Node* 4 sebesar 383. Hal ini menunjukkan bahwa kondisi tanah di setiap *Node* berbeda, meskipun nilai suhu tetap relatif sama.

5. Percobaan dengan Jarak

Berdasarkan hasil pengujian pada jarak 2 meter, performa *Bluetooth Mesh* menunjukkan keandalan yang sangat tinggi dalam hal pengiriman data, ditandai dengan tingkat keberhasilan 100% untuk semua transmisi dari *Node* 1, 3, dan 4. Seluruh data berhasil diterima dan diproses dengan fungsi *relay* yang aktif, membuktikan bahwa mekanisme dasar jaringan *mesh* telah berjalan sesuai desain.

Meskipun demikian, teramati variasi *latency* yang signifikan, dari 8,5 ms hingga 124,2 ms, yang merupakan ciri khas dari protokol *Bluetooth Mesh* akibat penerapan *managed flooding* dan rute yang berbeda-beda. Hal ini mengonfirmasi bahwa bahkan pada jarak dekat, pertukaran paket melibatkan dinamika perutean yang kompleks.

Dari segi konsistensi, ukuran paket data untuk setiap *Node* cenderung stabil (*Node* 1 dan 3: 208 bit, *Node* 4: 200 bit), menunjukkan integritas struktur pesan yang baik. Nilai-nilai sensor yang dilaporkan juga konsisten, mengindikasikan bahwa tidak ada gangguan selama transmisi.

Secara keseluruhan, pada jarak 2 meter, *Bluetooth Mesh* membuktikan kehandalannya untuk aplikasi monitoring sensor, meskipun tantangan *latency* masih perlu diperhatikan untuk aplikasi yang membutuhkan waktu nyata. Pengujian lebih lanjut pada jarak yang lebih jauh direkomendasikan untuk mengevaluasi kemampuan perluasan jangkauan yang sebenarnya.

6. Skenario *Node* 1 dinonaktifkan

Berdasarkan hasil uji coba dengan kondisi *Node* 1 dimatikan, performa jaringan *Bluetooth Mesh* menunjukkan ketahanan yang tinggi. Data berhasil diterima secara penuh dari *Node* 3 dan *Node* 4, membuktikan bahwa jaringan tetap beroperasi dengan normal meskipun satu *Node* tidak aktif. Hal ini mengkonfirmasi kemampuan *fault tolerance* dan *self-healing* yang menjadi keunggulan desain jaringan *mesh*.

Dari segi kinerja, *latency* yang tercatat mengalami penurunan signifikan dari rentang milidetik sebelumnya yang mencapai 124 ms, kini *latency* tertinggi hanya 42,5 ms dan terendah 2,8 ms. Penurunan ini disebabkan berkurangnya lalu lintas data sehingga beban jaringan lebih ringan dan proses *relay* menjadi lebih efisien.

Fungsi *relay* tetap berjalan pada semua transmisi, dan ukuran data dari masing-masing *Node* stabil (*Node* 3: 208 bit, *Node* 4: 200 bit), menunjukkan bahwa integritas data tidak terganggu meskipun terjadi perubahan topologi jaringan.

Secara keseluruhan, uji coba ini membuktikan bahwa *Bluetooth Mesh* dapat tetap andal dan bahkan mengalami peningkatan kinerja *latency* dalam kondisi kehilangan satu *Node*, sehingga cocok untuk aplikasi yang memerlukan ketahanan dan ketersediaan tinggi.

7. Percobaan Tanpa *Relay*

Berdasarkan data Tabel 4.7 “Uji Coba *Node* Tanpa *Relay*” berikut adalah analisis performa jaringan *Bluetooth Mesh* ketika fungsi *relay* dinonaktifkan:

1. Kinerja latensi tidak stabil cenderung tinggi
   A. Variasi Latensi Signifikan: Delay berkisar dari 1.086 ms hingga 33,427 ms yang menunjukkan ketidakkonsistenan yang besar.
   B. Pola Peningkatan: Terdapat tren peningkatan *latency* seiring waktu, dengan delay tertinggi (33,427 ms) terjadi pada transmisi dari *Node* 4. Hal ini menunjukkan bahwa tanpa *relay*, *Node* yang jaraknya lebih jauh mengalami kesulitan untuk mengirim data secara efisien.
2. Dampak negatif pada *Node* jarak jauh

   Node 4 Mengalami *latency* Tertinggi: Transmisi dari *Node* 4 mencatat delay 33,427 ms, yang secara signifikan lebih tinggi daripada *Node* lainnya. Ini membuktikan bahwa tanpa fungsi *relay*, *Node* yang berada di posisi terjauh menjadi paling terdampak karena tidak dapat memanfaatkan *Node* lain sebagai perantara.
3. Stabilitas jaringan terjaga dengan baik
   A. Ukuran Data Konsisten:
      1) *Node* 1 dan *Node* 4 mengirim 208 bit.
      2) *Node* 3 mengirim 200 bit.

      Kekonsistenan ini menunjukkan bahwa integritas data tetap terjamin meskipun fungsi *relay* dinonaktifkan.
   B. Pembacaan Sensor Stabil: Nilai-nilai sensor (suhu, kelembaban udara, kelembaban tanah) dilaporkan dengan konsisten oleh masing-masing *Node*, mengindikasikan bahwa akurasi pengukuran tidak terpengaruh.

4. Keterbatasan jaringan tanpa *relay*

Rentan terhadap Gangguan: Tanpa *relay*, jaringan menjadi lebih bergantung pada koneksi langsung antar-*Node*, yang rentan terhadap gangguan seperti penghalang fisik atau jarak yang terlalu jauh.

Efisiensi Jaringan Menurun: Tidak adanya mekanisme *relay* menyebabkan beban pengiriman data hanya bergantung pada *Node* itu sendiri, sehingga *Node* yang jaraknya jauh harus mengirim data dengan daya yang lebih tinggi dan *latency* yang lebih lama.

4.3 Pembahasan

Berdasarkan serangkaian pengujian yang dilakukan, *Bluetooth Mesh* secara konsisten membuktikan dirinya sebagai protokol yang andal dan tangguh untuk aplikasi *Internet of Things* (IoT), khususnya dalam sistem monitoring sensor. Pada pengujian dasar dengan jarak 2 meter, jaringan menunjukkan kinerja yang sempurna dalam hal keandalan dengan mencapai *packet delivery rate* 100%, dimana semua data dari *Node* 1, 3, dan 4 berhasil diterima. Keberhasilan ini didukung oleh fungsi *relay* yang bekerja optimal, yang ditandai dengan kolom "*Relay*" yang seluruhnya bernilai Ya, membuktikan bahwa mekanisme fundamental dari jaringan *mesh* telah beroperasi sesuai desain. Namun, di balik keandalan ini, teramati variasi *latency* yang signifikan, berkisar antara 8,5 ms hingga 124,2 ms, yang merupakan karakteristik inherent dari protokol *managed flooding* pada *Bluetooth Mesh*, dimana setiap paket data dapat menempuh jalur dan jumlah *hop* yang berbeda-beda dalam jaringan, bahkan pada jarak yang relatif dekat.

Ketangguhan sistem ini semakin teruji dalam skenario simulasi kegagalan *Node*, dimana ketika *Node* 1 dimatikan, jaringan secara otomatis menyesuaikan diri dan mempertahankan operasi penuh hanya melalui *Node* 3 dan 4, membuktikan kemampuan *fault tolerance* dan *self-healing* yang menjadi keunggulan utama arsitektur *mesh*. Yang menarik, dalam kondisi ini justru terjadi peningkatan kinerja *latency* yang signifikan (2,8-42,6 ms),

mengindikasikan bahwa berkurangnya beban lalu lintas jaringan menyebabkan proses *relay* menjadi lebih efisien. Sebaliknya, ketika fungsi *relay* secara keseluruhan dinonaktifkan, performa jaringan mengalami penurunan yang jelas dengan *latency* yang menjadi tidak stabil (1,1-33,4 ms) dan *Node* yang secara geografis terjauh (*Node* 4) mengalami *latency* tertinggi (33,4 ms), menegaskan peran kritis fungsi *relay* dalam memperluas jangkauan jaringan, mendistribusikan beban komunikasi, dan mempertahankan *latency* yang rendah.

Secara keseluruhan, hasil pengujian menyimpulkan bahwa *Bluetooth Mesh* menawarkan solusi yang sangat efektif untuk aplikasi IoT yang memprioritaskan cakupan area yang luas, ketahanan terhadap kegagalan komponen, dan keandalan pengiriman data, dengan *trade-off* pada aspek *latency* yang masih dalam batas *acceptable* untuk aplikasi-aplikasi *non-real-time* seperti monitoring lingkungan.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan alat sensor berbasis *Bluetooth Mesh*, maka dapat disimpulkan:

1. Perancangan dan implementasi jaringan sensor *Bluetooth Mesh* berhasil dibangun menggunakan 4 *Node* dengan spesifikasi: mampu mengukur suhu (27,1–29,1°C), kelembaban udara (70,4–85,5%), dan kelembaban tanah (200–2755). Sistem dirancang dengan memanfaatkan fungsi *relay* untuk memperluas jangkauan dan menjaga keandalan jaringan.
2. Hasil pengujian membuktikan keandalan sistem dengan capaian 100% *packet delivery rate* pada kondisi normal, *latency* berkisar 1,086–124,211 ms, dan ketahanan terhadap kegagalan *Node* dimana jaringan tetap beroperasi penuh meskipun satu *Node* dimatikan. Namun, *disable* fungsi *relay* menyebabkan peningkatan *latency* signifikan hingga 33,427 ms pada *Node* terjauh.

5.2 Saran

Agar sistem ini dapat dikembangkan dan dimanfaatkan secara maksimal, disarankan:

1. Sistem sebaiknya dilengkapi dengan antarmuka aplikasi agar pengguna dapat memantau data hasil monitoring secara langsung tanpa bergantung pada serial monitor. Selain itu, pengembangan lebih lanjut dapat mencakup fitur pelacakan jalur transmisi data untuk memvalidasi setiap *hop* dalam jaringan.
2. Sistem dirancang sesuai dengan standar dan spesifikasi BLE *Mesh* dari SIG (*Special Interest Group*). Pengujian lanjutan dapat dilakukan dengan menambah jumlah *Node*, mengoptimalkan jarak antar *Node*, dan mengujinya dalam berbagai kondisi lingkungan guna meningkatkan cakupan, keandalan, serta kualitas data yang diperoleh dari sistem monitoring ini.

DAFTAR PUSTAKA

Aji, B. R., Wahyu, R., & Sari, I. P. (2023). Performance Evaluation of *Bluetooth Mesh* for Soil Moisture Monitoring in Obstacle-Dense Environments. *Journal of Agricultural Engineering*, 15(1), 45-56.

Bluetooth SIG. (2017). *Bluetooth Mesh Networking: The Definitive Guide*.

Bluetooth Special Interest Group. *Bluetooth technology overview*. *Bluetooth*. https://www.*bluetooth*.com/learn-about-*bluetooth*/tech-overview/ diakes pada (12.20 Februari 2025)

Botland. *GRAVITY TEMPERATURE* SENSORS. https://botland.store/gravity-temperature-sensors/17377-gravity-analog-waterproof-capacitive-soil-moisture-sensor-v20-dfrobot-sen0308-6959420917068.html. Diakses pada (00.47 1 Agustus 2025)

Chen, L., Wang, Y., & Zhao, F. (2020). An Agricultural Monitoring System Based on *Bluetooth Mesh*. *IEEE* Sensor*s Journal*, 20(18), 10923-10930.

COMPONENTS101. DHT22 – *Temperature and Humidity* Sensor. https://components101.com/sensors/dht22-pinout-specs-datasheet diakses pada (11.35 31 Juli 2025)

ESPBoards. *epsboards.dev*. https://www.espboards.dev/esp32/d1-uno32/ diakses pada (10.48 31 Juli 2025)

Intel. *What is Bluetooth® technology?* Intel. https://www.intel.com/content/www/us/en/products/docs/wireless/what-is-*bluetooth*.html(09.56 2 Februari 2025)

Jiang, X., Zhang, H., Barsallo Yi, E. A., Raghunathan, N., Mousoulis, C., Chaterji, S., Peroulis, D., Shakouri, A., & Bagchi, S. (2021). Hybrid Low-Power WideArea *Mesh* Network for IoT Applications. IEEE Internet of Things Journal, 8(2), 901–915. https://doi.org/10.1109/JIOT.2020.3009228

Kadir, A. A., Bhawiyuga, A., & Basuki, A. (2023). Implementasi perangkat *gateway* mobile berbasis protokol BLE untuk mendukung akuisisi data Internet of Things secara portabel. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 7*(6), 3033-3040. http://j-ptiik.ub.ac.id

Mansyur S., Fatizah R. (2019). Rancang Bangun Alat Ukur Kadar Air Pada Gabah Menggunakan Sensor YL-69 Berbasis Mikrokontroler. Politeknik Negeri Ujung Pandang.

Muliadi, Imran, A., & Rasul, M. (2020). Pengembangan tempat sampah pintar menggunakan ESP32. *Jurnal MEDIA ELEKTRIK*, 17(2), 73–79.

Natgunanathan, I., Fernando, N., Loke, S.W., & Weerasuriya, C. (2023). *Bluetooth* Low Energy *Mesh*: Applications, Considerations and Current State-of-the-Art. Sensors, 23(4), 1826. https://doi.org/10.3390/s23041826

Purnawan A. (2021). Sistem Multi Sensor Nirkabel Untuk Aplikasi IoT Dalam Lingkungan Pertanian. Politeknik Negeri Ujung Pandang.

Putra, A. S., Darmawan, H., & Wijaya, H. (2021). IoT-Based Smart Irrigation System for Sustainable Agriculture: A Review. *Indonesian Journal of Electronics and Instrumentation Systems*, 11(2), 123-135.

Santoso, B., Prasetyo, D., & Handayani, T. (2022). Comparative Analysis of LoRa and NB-IoT for Smart Farming Applications in Rural Areas. *Journal of Telecommunication, Electronic and Computer Engineering*, 14(1), 27-32.

Sardi, R. I. *et al. (*2020) 'Usulan Tugas Akhir Implementasi Algoritme Aes 128bit Pada Mikrokontroler *Node*mcu Mneggunakan Arsitektur Web Service Rest Untuk Keamanan'

Suhud, A. A. A. (2022). Model jaringan *mesh* pada sistem Internet of Things (IoT) untuk pemantauan lingkungan. Politeknik Negeri Ujung Pandang.

LAMPIRAN

Lampiran 1 *Listing* Pemrograman Arduino IDE

```
#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEScan.h>
#include <BLEAdvertisedDevice.h>
#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define SOIL_PIN 34
#define LED_PIN 2
#define NODE_ID "Node1" // Ganti sesuai Node

DHT dht(DHTPIN, DHTTYPE);
BLEAdvertising* pAdvertising;
BLEScan* pBLEScan;

String lastSentMessage = "";
String lastRelayedMessage = "";
String pendingRelayMsg = "";

unsigned long lastSend = 0;
unsigned long lastScanRestart = 0;
```

```
unsigned long relayStartTime = 0;
bool isRelaying = false;

// Waktu terakhir pesan diterima dari masing-masing Node
unsigned long lastReceivedTime_Node1 = 0;
unsigned long lastReceivedTime_Node2 = 0;
unsigned long lastReceivedTime_Node3 = 0;
unsigned long lastReceivedTime_Node4 = 0;

class MyAdvertisedDeviceCallbacks : public BLEAdvertisedDeviceCallbacks {
 void onResult(BLEAdvertisedDevice advertisedDevice) {
 String msg = advertisedDevice.getManufacturerData();

 if (msg.startsWith("Node") && !msg.startsWith(NODE_ID)) {
 if (msg != lastRelayedMessage) {
 Serial.print("📥 Pesan diterima: ");
 Serial.println(msg);

 // Hitung total bit
 int totalBits = msg.length() * 8;
```

```
  Serial.print("📊 Total bit: ");
  Serial.println(totalBits);

  lastRelayedMessage = msg;
  pendingRelayMsg = msg;
  relayStartTime = millis();
  isRelaying = true;

  // Logging delay antar Node
  String sender = msg.substring(0, 5);
  unsigned long now = millis();
  if (sender == "Node1") {
  Serial.print("⏱ Delay dari Node1: ");
  Serial.print(now - lastReceivedTime_Node1);
  Serial.println(" ms");
  lastReceivedTime_Node1 = now;
  } else if (sender == "Node2") {
  Serial.print("⏱ Delay dari Node2: ");
  Serial.print(now - lastReceivedTime_Node2);
  Serial.println(" ms");
  lastReceivedTime_Node2 = now;
  } else if (sender == "Node3") {
```

```
 Serial.print("⏱ Delay dari Node3: ");
 Serial.print(now -
lastReceivedTime_Node3);
 Serial.println(" ms");
 lastReceivedTime_Node3 = now;
 } else if (sender == "Node4") {
 Serial.print("⏱ Delay dari Node4: ");
 Serial.print(now -
lastReceivedTime_Node4);
 Serial.println(" ms");
 lastReceivedTime_Node4 = now;
 }

 // Blink LED saat menerima pesan
 for (int i = 0; i < 3; i++) {
 digitalWrite(LED_PIN, HIGH);
 delay(100);
 digitalWrite(LED_PIN, LOW);
 delay(100);
 }
 }
 }
 }
};
```

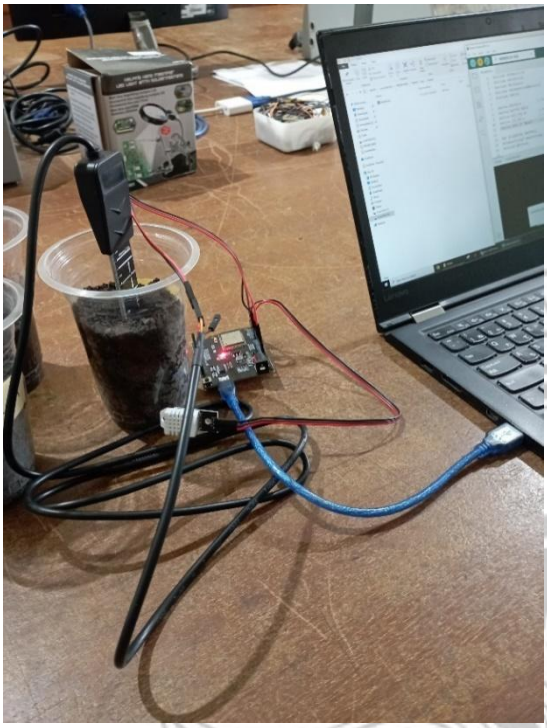
```
void setup() {
 Serial.begin(115200);
 Serial.println("🚀 Program dimulai...");
 pinMode(LED_PIN, OUTPUT);
 dht.begin();

 BLEDevice::init(NODE_ID);

 pBLEScan = BLEDevice::getScan();
 pBLEScan->setAdvertisedDeviceCallbacks(new
MyAdvertisedDeviceCallbacks());
 pBLEScan->setActiveScan(true);
 pBLEScan->start(0, nullptr); // terus scan

 pAdvertising =
BLEDevice::getAdvertising();
 pAdvertising->setScanResponse(false);
 pAdvertising->setMinPreferred(0x06);
 pAdvertising->setMinPreferred(0x12);
}

void loop() {
```

```
unsigned long now = millis();


// Restart scan setiap 10 detik
if (now - lastScanRestart > 10000) {
pBLEScan->stop();
delay(100);
pBLEScan->start(0, nullptr);
lastScanRestart = now;
}


// Kirim data sensor setiap 5 detik
if (now - lastSend > 5000) {
float temp = dht.readTemperature();
float hum = dht.readHumidity();
int soil = analogRead(SOIL_PIN);


if (isnan(temp) || isnan(hum)) {
Serial.println("⚠️ Gagal membaca sensor DHT22!");
} else {
String message = String(NODE_ID) + ":T=" + String(temp, 1) + ",H=" + String(hum, 1) + ",S=" + String(soil);
```

```
if (message != lastSentMessage) {
lastSentMessage = message;
Serial.print("📤 Mengirim: ");
Serial.println(message);

BLEAdvertisementData data;
data.setManufacturerData(message);
pAdvertising->setAdvertisementData(data);

pBLEScan->stop();
pAdvertising->start();
delay(200);
pAdvertising->stop();
pBLEScan->start(0, nullptr);

digitalWrite(LED_PIN, HIGH);
delay(200);
digitalWrite(LED_PIN, LOW);
}
}
lastSend = now;
}
```

```
 // Relay pesan jika diperlukan
 if (isRelaying && millis() - relayStartTime > 200) {
 Serial.println("🔁 Relay ulang pesan ke Node lain");

 BLEAdvertisementData relayData;

relayData.setManufacturerData(pendingRelayMsg);
 pAdvertising->setAdvertisementData(relayData);

 pBLEScan->stop();
 pAdvertising->start();
 delay(5000);
 pAdvertising->stop();
 pBLEScan->start(0, nullptr);

 isRelaying = false;
 }
}
```

Lampiran 2 Foto Kegiatan Perancangan



Lampiran 3 Rancangan Anggaran Biaya

No	Komponen	Jumlah	Harga Satuan	Jumlah Harga
1	Wemos ESP32 UnoD1 R32 WiFi Bluetooth 4MB Module Arduino	4	Rp 59.090	Rp 236.360
2	Gravity Analog Waterproof Capacitive Soil Moisture Sensor	4	Rp 256.500	Rp 1,026.000
3	DHT22 AM2302 Temperature Humidity Sensor Suhu Module For Arduino	4	Rp 22.900	Rp 91.600
4	Kabel Micro USB to USB Wemos Nodemcu ESP32 Arduino	4	Rp 10.500	Rp 42.000
5	Kabel Jumper Male to Male	40 Pcs	Rp 9.900	Rp 9.900
6	Kabel Jumper Famale to Male	40 Pcs	Rp 9.900	Rp 9.900
7	USB HUB LOGON 3.0 4port High Speed Power USB A to USB A , 0.15 Meter	1	Rp 60.000	Rp 60.000
8	Casing Plastic Arduino Uno R3 V3 Box	4	Rp 11.000	Rp 44.000
Total Biaya				**Rp 1,519.760**

Lampiran 4 Kartu Kontrol Pembimbing 1

CATATAN KONSULTASI TUGAS AKHIR
PROGRAM STUDI D3 TEKNIK TELEKOMUNIKASI

NAMA MAHASISWA : 1. AI WULANDARI 2. ASEP YONO	STB :32222024 STB :32222009
JUDUL : RANCANG BANGUN MONITORING KELEMBABAN DAN SUHU BERBASIS BLUETOOTH MESH	NAMA PENGARAH 1. Umar Katu, S.T., M.T.

Tanggal Persetujuan Judul : 12 FEBRUARI 2025

Pengarah :

No.	Tanggal	Catatan/Komentar	Tanda Tangan
1	2	3	4
1.	20/02/25	konsultasi judul	1
2	11/03/25	konsultasi pembuatan	2
3	25/03/25	konsultasi Alat	3
4	30/04/25	konsultasi komponen	4
5	20/04/25	konsultasi data	5
6	5/05/25	konsultasi data	6
7	16/06/25	konsultasi BAB IV/V	7
8	31/07/25	konsultasi presentasi	8
9	4/08/25	ACC.	9

Pengarah Utama,

Umar Katu, S.T., M.T.
NIP. 19730820 200801 1 005

Lampiran 5 Kartu Kontrol Pembimbing 2

CATATAN KONSULTASI TUGAS AKHIR
PROGRAM STUDI D3 TEKNIK TELEKOMUNIKASI

NAMA MAHASISWA : 1. AI WULANDARI 2. ASEP YONO	STB :32222024 STB :32222009
JUDUL : RANCANG BANGUN MONITORING KELEMBABAN DAN SUHU BERBASIS BLUETOOTH MESH	NAMA PENGARAH PENDAMPING 1. Ir. Sandriyones Bunga Palinggi, S.T., M.T.
Tanggal Persetujuan Judul : 12 FEBRUARI 2025	

Pengarah Pendamping

No.	Tanggal	Catatan/Komentar	Tanda Tangan
1	2	3	4
1	21/02/25	Diskusi Desain Alat	
2	12/03/25	Revisi Tata Penulisan	
3	29/04/25	Revisi Tampilan Grafik	
4	6/05/25	Revisi Penulisan Laporan TA (2)	
5	28/05/25	Revisi Penulisan Laporan TA (3)	
6	18/05/25	Demonstrasi Alat	
7	30/07/25	Revisi Blok Diagram	
8	2/8/25	Finalisasi Laporan	

Pengarah Pendamping,

Ir. Sandriyones Bunga Palinggi, S.T., M.T.
NIP.19880611 202203 1 002