



**REAL TIME SYSTEM AND INTERNET OF THINGS FINAL PROJECT REPORT  
DEPARTMENT OF ELECTRICAL ENGINEERING  
UNIVERSITAS INDONESIA**

**BLOOM – Basic Local Observation & Optimized Moisture-control**

**GROUP 16**

<b>Isyana Trevia Pohaci</b>	<b>2306250592</b>
<b>Arsinta Kirana Nisa</b>	<b>2306215980</b>
<b>Jonathan Frederick Kosasih</b>	<b>2306225981</b>
<b>Laura Fawzia Sambowo</b>	<b>2306260145</b>

## PREFACE

Puji dan syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa karena berkat rahmat-Nya, kami dapat menyelesaikan proyek akhir *Real Time System and Internet of Things* (IoT) ini yang berjudul "BLOOM (*Basic Local Observation & Optimized Moisture-control*)" dengan baik.

Proyek ini dirancang untuk memodernisasi perawatan tanaman dengan memanfaatkan teknologi *Real-Time Operating System* dan komunikasi *wireless* berbasis MQTT. Laporan ini disusun untuk mendokumentasikan konsep, desain, implementasi, serta hasil dari pengembangan sistem penyiraman otomatis BLOOM. Sistem ini mengintegrasikan dua node ESP32 yang berkomunikasi secara terpisah (Sensor Node dan *Coordinator Node*), sensor kelembapan tanah, DHT22, serta *platform* Blynk untuk pengendalian jarak jauh.

Kami menyadari bahwa laporan ini masih memiliki kekurangan. Oleh karena itu, kami membuka diri untuk menerima kritik dan saran yang membangun agar proyek ini dapat terus dikembangkan lebih baik lagi.

Akhir kata, kami mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan dan bimbingan selama proses pengembangan proyek ini. Semoga laporan ini dapat memberikan manfaat bagi pembaca, khususnya dalam memahami implementasi teknologi IoT untuk *distributed system* sederhana yang kami terapkan untuk efisiensi daya dan pemrosesan data.

Depok, 8 Desember, 2025

Group 16

## TABLE OF CONTENTS

<b>PREFACE.....</b>	<b>2</b>
<b>TABLE OF CONTENTS.....</b>	<b>3</b>
<b>CHAPTER 1</b>	
<b>INTRODUCTION.....</b>	<b>4</b>
1.1 PROBLEM STATEMENT.....	4
1.2 PROPOSED SOLUTION.....	5
1.3 ACCEPTANCE CRITERIA.....	6
1.4 ROLES AND RESPONSIBILITIES.....	7
1.5 TIMELINE AND MILESTONES.....	7
<b>CHAPTER 2</b>	
<b>IMPLEMENTATION.....</b>	<b>9</b>
2.1 HARDWARE DESIGN AND SCHEMATIC.....	9
2.2 SOFTWARE DEVELOPMENT.....	10
2.3 HARDWARE AND SOFTWARE INTEGRATION.....	15
<b>CHAPTER 3</b>	
<b>TESTING AND EVALUATION.....</b>	<b>17</b>
3.1 TESTING.....	17
3.2 RESULT.....	18
3.2.1 Konektivitas Jaringan (Infrastructure).....	18
3.2.2 Transmisi dan Visualisasi Data (Software).....	19
3.2.3 Fungsionalitas Automatic Control (Hardware Logic).....	19
3.2.4 Peran Coordinator (Node 2).....	21
3.3 EVALUATION.....	22
<b>CHAPTER 4</b>	
<b>CONCLUSION.....</b>	<b>23</b>

# CHAPTER 1

## INTRODUCTION

### 1.1 PROBLEM STATEMENT

Bercocok tanam dan memelihara tanaman hias merupakan kegiatan yang populer di lingkungan perumahan maupun perkotaan atau yang biasa disebut dengan *urban farming*. Namun, keberhasilan dalam merawat tanaman sangat bergantung pada konsistensi perawatan yang dilakukan, terutama dalam hal penyiraman tanaman. Padatnya aktivitas dan mobilitas dari masyarakat modern seringkali menjadi sebuah penghalang utama dalam menjaga rutinitas untuk pemeliharaan tanaman. Akibatnya, banyak tanaman yang mengalami kerusakan bahkan kematian karena kelalaian dari pemilik tanaman, baik karena kekurangan air atau kelebihan air yang bisa menyebabkan pembusukan akar.

Metode perawatan konvensional yang mengandalkan pengecekan manual sering kali tidak efektif dan subjektif. Pemilik tanaman sering kali hanya mengira-ngira ketika melakukan penyiraman tanaman dari tampilan fisik permukaan tanah, padahal kondisi kelembapan di area akar berbeda. Selain itu, penggunaan sistem penyiraman otomatis secara sederhana berbasis *time* juga memiliki kelemahan, yaitu tidak bisa beradaptasi dengan kondisi lingkungan nyata. Sistem timer akan tetap menyiram tanaman, meskipun tanah masih basah yang justru berpotensi untuk merusak tanaman.

Tantangan lainnya adalah keterbatasan akses pemantauan ketika pemilik sedang tidak berada di rumah. Tanpa adanya sistem pemantauan dari jarak jauh, pemilik tidak dapat mengetahui kondisi tanaman, seperti suhu dan kelembapan udara yang dapat mempengaruhi kesehatan tanaman. Kurangnya data *real-time* ini membuat penanganan masalah menjadi lambat dan risiko kematian tanaman semakin tinggi.

Oleh karena itu, diperlukan sebuah sistem cerdas yang mampu melakukan proses penyiraman berdasarkan data kondisi tanah yang akurat, bukan sekadar jadwal waktu secara otomatis. Solusi ini juga harus memiliki kemampuan konektivitas yang memungkinkan pemilik untuk memantau dan mengendalikan perawatan tanaman dari jarak jauh, sehingga dapat memberikan efisiensi, ketenangan pikiran, dan pertumbuhan tanaman yang optimal.

## 1.2 PROPOSED SOLUTION

Untuk mengatasi permasalahan dari tidak konsistennya perawatan tanaman yang diidentifikasi, kami mengusulkan sebuah sistem irigasi cerdas berbasis *Internet of Things* yang diberi nama **BLOOM**, yaitu ***Basic Local Observation and Optimized Moisture Control***. Sistem ini dirancang untuk memberikan perawatan tanaman yang presisi dengan memanfaatkan arsitektur sistem terdistribusi, sensor lingkungan, dan konektivitas jarak jauh melalui aplikasi Blynk.

Solusi ini memiliki fitur utama berupa mekanisme penyiraman otomatis yang dikendalikan oleh data *real-time* dari sensor kelembapan tanah. Sensor ini akan terus-menerus memantau kadar air dalam tanah. Ketika kelembapan tanah terdeteksi berada di bawah ambang batas yang ditentukan, sistem akan mengaktifkan pompa air melalui MOSFET *driver* untuk menyiram tanaman hingga kondisi tanah kembali optimal. Hal ini memastikan tanaman mendapatkan air sesuai kebutuhan fisiologisnya tanpa intervensi manual yang berlebihan.

**BLOOM** menerapkan arsitektur *dual node* menggunakan dua buah mikrokontroler ESP32 yang berkomunikasi melalui MQTT:

- **Node 1**

Bertugas membaca data fisik dari sensor tanah dan sensor suhu atau kelembapan udara (DHT11), serta mengeksekusi penyiraman. Node ini menggunakan sistem operasi *real-time* untuk mengelola multitasking antara pembacaan sensor, kontrol pompa, dan komunikasi data tanpa hambatan (*non-blocking*).

- **Node 2**

Berfungsi sebagai *gateway* yang menerima data dari Node 1 dan meneruskannya ke *interface user*.

Aplikasi Blynk menjadi pusat pemantauan dan pengendalian sistem ini, memungkinkan pengguna untuk:

- Menerima data kondisi tanaman, yaitu kelembapan tanah, suhu udara, kelembapan udara secara *real-time*.
- Mendapatkan status aktif atau non aktif pompa air.

- Melakukan penyiraman manual jarak jauh jika diperlukan, di luar jadwal otomatisasi.

Dengan solusi ini, pemilik tanaman tidak hanya mendapatkan kemudahan dalam perawatan, tetapi juga efisiensi penggunaan air dan jaminan kesehatan tanaman yang lebih baik. Penerapan teknologi *FreeRTOS* dan MQTT pada **BLOOM** juga menjamin sistem berjalan dengan responsif, stabil, dan dapat diandalkan untuk penggunaan jangka panjang.

### 1.3 ACCEPTANCE CRITERIA

Kriteria diterimanya proyek **BLOOM** adalah sebagai berikut:

1. Node 1 harus mampu membaca data kelembapan tanah serta suhu dan kelembapan udara menggunakan DHT11 dengan stabil, lalu mengkonversi nilai analog sensor tanah menjadi persentase 0 sampai 100%.
2. Sistem harus dapat mengaktifkan pompa air secara otomatis ketika sensor mendeteksi kelembapan tanah berada di bawah ambang batas yang ditentukan, yaitu 30% dan mematikannya setelah durasi penyiraman selesai selama 5 detik.
3. Node 1 dan Node 2 harus berhasil terhubung ke broker MQTT publik, di mana Node 1 mengirimkan data sensor, yaitu *publish* dan Node 2 menerima data tersebut, yaitu *subscribe* tanpa kegagalan koneksi yang berulang.
4. Data yang diterima oleh Node 2 harus dapat ditampilkan secara *real-time* pada *interface* aplikasi Blynk, mencakup nilai kelembapan tanah, suhu, kelembapan udara, dan status pompa.
5. Pengguna harus dapat melakukan penyiraman manual melalui tombol virtual pada aplikasi Blynk, di mana perintah tersebut diteruskan dari Node 2 ke Node 1 untuk mengaktifkan pompa.

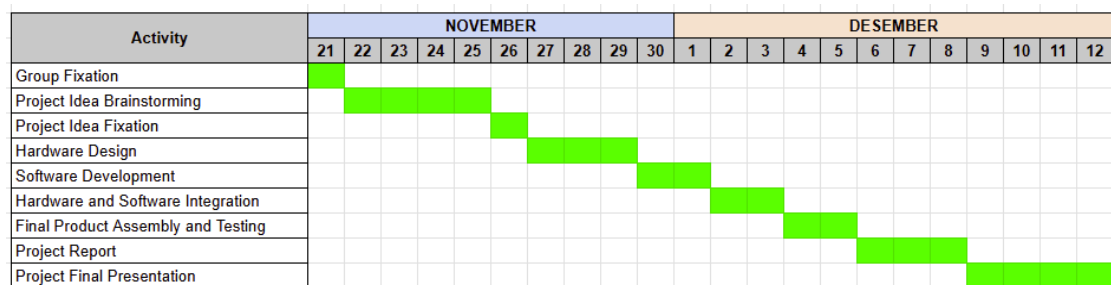
## 1.4 ROLES AND RESPONSIBILITIES

Peran dan tanggung jawab dari anggota tim adalah sebagai berikut:

Roles	Responsibilities	Person
Ketua Kelompok	Laporan Akhir, <i>Hardware assembly</i> , dan Node 1 <i>implementation</i>	Isyana Trevia Pohaci
Anggota	Laporan Akhir, <i>Hardware assembly</i> , <i>debugging hardware</i> .	Arsinta Kirana Nisa
Anggota	Laporan Akhir, <i>Debugging Software</i> dan Integrasi <i>Blynk</i> pada node 2	Jonathan Frederick Kosasih
Anggota	Laporan Akhir, <i>Hardware assembly</i> , <i>Debugging node 1</i> dan node 2, dan Node 2 <i>implementation</i>	Laura Fawzia Sambowo

**Table 1.** Roles and Responsibilities

## 1.5 TIMELINE AND MILESTONES



**Fig 1.** Timeline and Milestones

## CHAPTER 2

### IMPLEMENTATION

#### 2.1 HARDWARE DESIGN AND SCHEMATIC

Pada tahap desain *hardware design*, pengembangan sistem **BLOOM** difokuskan pada penciptaan arsitektur sistem terdistribusi yang memisahkan fungsi pembacaan sensor dan fungsi komunikasi pengguna ke dalam dua unit mikrokontroler yang berbeda. Design ini dipilih untuk meningkatkan efisiensi pemrosesan data dan stabilitas sistem. Desain perangkat keras ini mencakup pemilihan mikrokontroler yang mendukung *multitasking*, sensor yang presisi, serta modul penggerak aktuator yang aman.

Komponen utama yang menjadi otak dari sistem ini adalah ESP32. ESP32 dipilih karena memiliki prosesor *dual core* yang mumpuni untuk menjalankan sistem *FreeRTOS*, serta memiliki fitur konektivitas Wi-Fi terintegrasi yang krusial untuk komunikasi protokol MQTT. Dalam implementasinya, perangkat keras dibagi menjadi dua node utama:

1. Node 1 bertugas sebagai unit pemroses utama di lapangan. Komponen yang terhubung meliputi:
  - *Capacitive Soil Moisture Sensor* dipilih karena lebih tahan terhadap korosi. Sensor ini terhubung ke pin ADC, yaitu GPIO 34 untuk membaca kadar air tanah secara analog.
  - Sensor DHT11 digunakan untuk memantau suhu dan kelembapan udara di sekitar tanaman terhubung ke GPIO 4 atau 15, memberikan data lingkungan yang lebih komprehensif dibandingkan hanya menyiram berdasarkan kondisi tanah.
  - Pompa Air DC dan MOSFET *Driver* karena ESP32 hanya bekerja pada tegangan 3.3V dengan arus rendah, pompa air tidak dihubungkan langsung ke mikrokontroler. BLOOM menggunakan modul *driver* MOSFET D4184 sebagai saklar elektronik. Pin kontrol ESP32, yaitu GPIO 27 mengirim sinyal trigger ke *driver*, yang kemudian mengalirkan daya eksternal ke pompa. Agar melindungi mikrokontroler dari *back EMF* dan kelebihan beban arus.
2. Node 2 berfungsi sebagai *gateway* informasi.



Desain *hardware* BLOOM menerapkan pemisahan jalur daya antara mikrokontroler dan aktuator pompa untuk menjaga stabilitas sistem, di mana penggunaan sumber daya eksternal untuk beban tinggi yang dipadukan dengan teknik *common ground* efektif meminimalkan risiko kerusakan akibat lonjakan arus. Selain itu, konfigurasi *pull-up resistor* internal diterapkan untuk menjamin akurasi pembacaan input tombol, sementara diagram skematik memperlihatkan integrasi menyeluruh sensor dan aktuator pada Node 1 serta fungsi pemantauan independen pada Node 2, menghasilkan arsitektur nirkabel yang baik dan fleksibel untuk operasional jangka panjang.

## 2.2 SOFTWARE DEVELOPMENT

Sistem **BLOOM** mengimplementasikan arsitektur terdistribusi berbasis *IoT* dengan dua node ESP32 yang memiliki peran berbeda namun saling terintegrasi. Arsitektur ini dirancang dengan prinsip *separation of concerns*, di mana Node 1 berfokus pada pembacaan sensor dan kontrol aktuator, sedangkan Node 2 berperan sebagai *gateway* yang menjembatani komunikasi antara sistem lokal dengan *cloud platform* Blynk. Kedua node menggunakan *FreeRTOS* sebagai fondasi sistem operasi yang memungkinkan eksekusi *multitasking* dengan *scheduling* deterministik dan memastikan bahwa setiap task dapat berjalan secara konkuren dengan prioritas yang telah ditentukan.

Node 1 mengimplementasikan enam *task FreeRTOS* yang berjalan secara konkuren dengan fungsi spesifik masing-masing, antara lain:

- **Task 1:** *taskReadSoil* membaca sensor kelembapan tanah secara analog setiap 5 detik dan hasilnya akan dikirimkan ke task lain melalui *queue*.
- **Task 2:** *taskReadDHT* membaca sensor DHT11 setiap 5 detik untuk memperoleh data suhu dan kelembapan udara. Task ini juga melakukan validasi data untuk memastikan pembacaan berhasil sebelum disimpan ke variabel global *latestData*.
- **Task 3:** *taskPumpController* mengimplementasikan logika kontrol otomatis untuk penyiraman tanaman berdasarkan data kelembapan tanah dari *queue*.
- **Task 4:** *taskMQTTPublish* bertanggung jawab untuk mempublikasikan data sensor ke *broker MQTT* setiap 10 detik.

- **Task 5:** `taskButtonHandler` menangani input dari dua *push button* pada rangkaian dengan implementasi *debouncing* untuk menghindari *false trigger*.
- **Task 6:** `taskLEDIndicator` mengontrol dua LED indikator berdasarkan status kelembapan tanah, hijau ketika tanah basah dan kuning ketika tanah kering. Saat pompa aktif, kedua LED berkedip bergantian untuk memberikan indikasi visual yang jelas.

Node 1 menggunakan berbagai metode sinkronisasi untuk mencegah *race condition*. *Queue* digunakan untuk mentransfer data sensor dari task pembaca ke task *controller*. *Mutex* digunakan untuk melindungi akses ke pin pompa untuk memastikan hanya satu task yang dapat mengontrol pompa pada satu waktu. *Binary Semaphore* digunakan untuk melindungi akses ke MQTT client. Selain itu, Node 1 *ter-subscribe* ke topik *plant/control/pump* untuk menerima perintah *remote* dari Node 2. *Callback* `mqttCallback()` dipanggil saat *message* diterima, kemudian memparsing *payload* untuk menentukan aksi (ON/OFF). Saat menerima perintah ON, flag `manualOverride` diaktifkan dan pompa dinyalakan. Flag ini mencegah logika *auto-watering* mengganggu operasi manual.

Node 2 mengimplementasikan dua task utama dengan fungsi sebagai *bridge* antara MQTT dan Blynk:

- **Task 1:** `taskMQTTHandler` berjalan dalam *infinite loop* menunggu *message* dari *FreeRTOS queue* yang diisi oleh MQTT *callback*. Task ini menggunakan *pattern producer-consumer* di mana MQTT *callback* bertindak sebagai *producer* dan task ini sebagai *consumer*.
- **Task 2:** `taskBlynkUpdate` bertanggung jawab untuk mengirimkan data ke Blynk *Cloud* setiap 2 detik. Task ini juga melakukan *auto-reconnection* jika koneksi Blynk terputus.

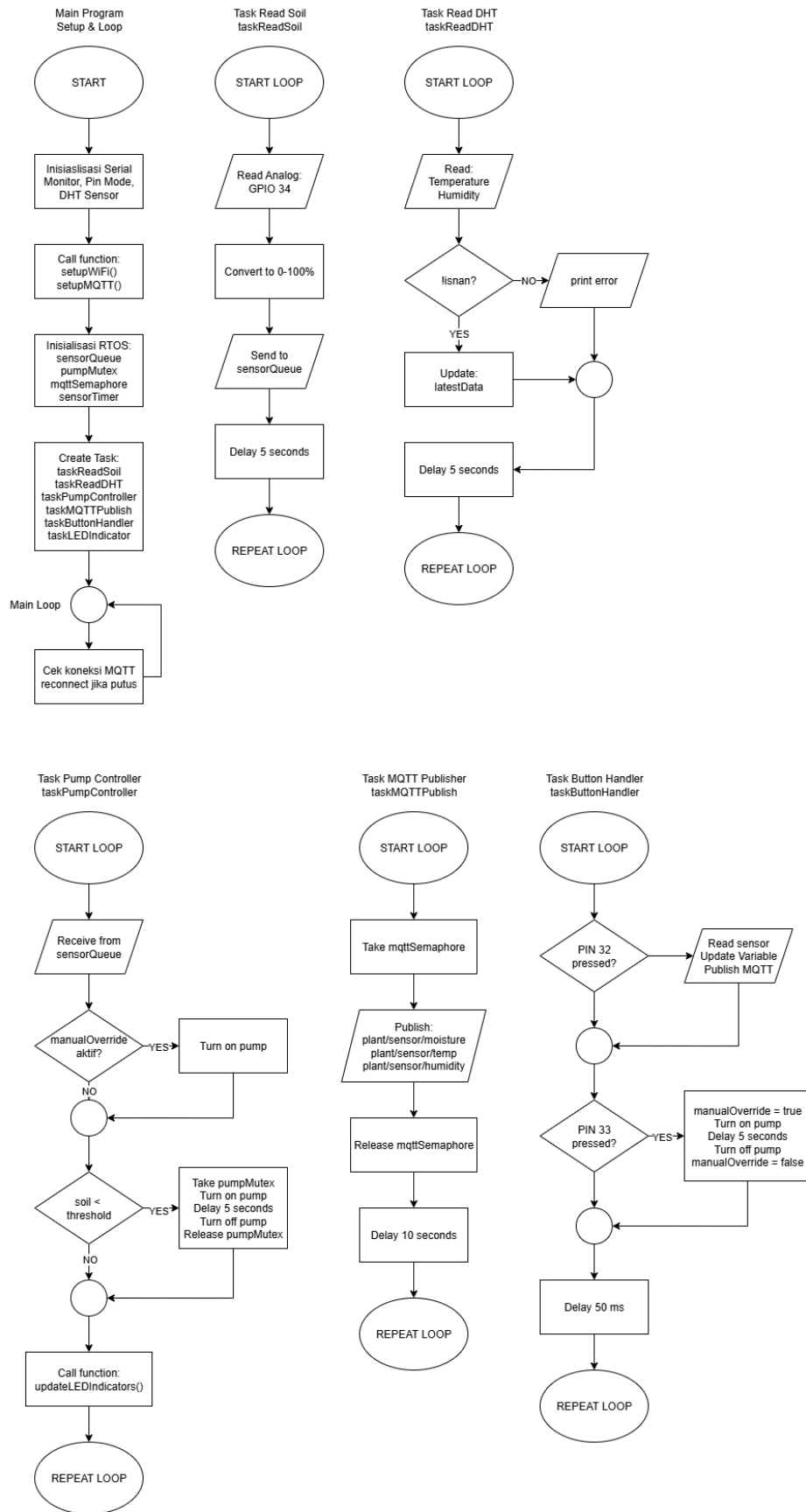
Node 2 mengimplementasikan mekanisme *auto-reconnection* untuk koneksi WiFi, MQTT, dan Blynk. Koneksi WiFi melakukan *auto-reconnect* melalui *timer callback* setiap 30 detik. MQTT melakukan *auto-reconnection* dengan melakukan *subscribe* ulang ke semua *topic* dilakukan setiap kali berhasil *reconnect*. Blynk menggunakan fungsi `Blynk.connect()` dengan *timeout built-in*. *Reconnection* dilakukan di dalam `taskBlynkUpdate` untuk memisahkan dari *main loop*.

Node 2 terintegrasi dengan Blynk untuk melakukan *monitoring* dan kontrol terhadap sistem melalui *library* BlynkSimpleEsp32. Blynk telah di-*setup* dengan *datastream* berikut:

Pin	Name	Data Type	Value Range	Units
V0	Soil Moisture	Integer	0 – 100	%
V1	Temperate	Double	-10 – 50	°C
V2	Humidity	Double	0 – 100	%
V3	Pump Status	Integer	0 – 1	
V4	Status Text	String		
V5	Remote Water Control	Integer	0 – 1	
V6	Last Update	String		

Untuk memperjelas alur kerja sistem, cara kerja sistem digambarkan melalui flowchart berikut:

- Node 1



- Node 2



## 2.3 HARDWARE AND SOFTWARE INTEGRATION

Integrasi *hardware* dan *software* pada sistem BLOOM menggunakan pendekatan iteratif dengan strategi *bottom-up testing*. Proses dimulai dengan testing komponen individual (unit testing), dilanjutkan dengan integrasi subsistem (integration testing), dan diakhiri dengan system testing untuk memverifikasi end-to-end functionality. Pendekatan ini dipilih untuk meminimalkan risiko dan memudahkan *troubleshooting*, karena setiap layer dapat diverifikasi secara independen sebelum diintegrasikan dengan layer yang lebih tinggi. Hal ini sangat penting dalam sistem *embedded* di mana *debugging* dapat menjadi challenging karena keterbatasan visibility dan tools.

### 2.3.1 Integrasi Node 1 - Komponen Hardware

Sensor kelembapan tanah kapasitif dihubungkan ke GPIO 34 dengan pertimbangan bahwa pin ini *dedicated ADC-only (input-only)* sehingga tidak ada *conflict* dengan fungsi GPIO lain. Integrasi dimulai dengan kalibrasi sensor untuk menentukan nilai ADC yang merepresentasikan kondisi kering dan basah. Nilai-nilai ini kemudian digunakan dalam fungsi `map()` untuk konversi linear ke skala 0-100%.

DHT11 menggunakan protokol komunikasi *one-wire proprietary* yang *timing-critical*. Library Adafruit DHT menangani kompleksitas protokol ini dan menyediakan *interface* sederhana melalui fungsi `readTemperature()` dan `readHumidity()`. Integrasi DHT11 memerlukan perhatian khusus pada *timing initialization* karena sensor memerlukan waktu 1-2 detik setelah *power-on* untuk stabilisasi internal.

Pompa air DC 5V tidak dapat langsung dikendalikan oleh GPIO ESP32 karena keterbatasan arus maksimum, sehingga digunakan MOSFET module D4184 sebagai switch driver yang dapat menangani arus tinggi. MOSFET gate dihubungkan ke GPIO 27 yang merupakan output-capable pin dengan dukungan PWM.

Dua LED (hijau dan kuning) dihubungkan ke GPIO 18 dan 19 masing-masing dengan resistor 220 $\Omega$  sebagai current limiter. Pemilihan GPIO ini berdasarkan ketersediaan pin dan tidak ada conflict dengan peripheral lain. Software mengontrol LED melalui `digitalWrite()` dengan logic HIGH untuk menyalakan dan LOW untuk mematikan. Blinking effect pada saat pompa aktif diimplementasikan dengan toggle state setiap 200ms di `taskLEDIndicator`.

Dua push button dihubungkan ke GPIO 32 dan 33 dengan konfigurasi active-low (button press = LOW). Internal pull-up resistor ESP32 diaktifkan melalui `pinMode(pin, INPUT_PULLUP)` sehingga tidak diperlukan resistor eksternal.

### 2.3.2 Integrasi Node 1 - Software Gateway

Node 2 tidak memiliki hardware peripheral yang terhubung ke GPIO, sehingga integrasi fokus pada layer software dan networking. Koneksi WiFi pada ESP32 berjalan pada core 0 dan memiliki task internal dengan prioritas tinggi. Koneksi awal menggunakan blocking call dengan timeout 20 detik, sedangkan sistem

memantau status melalui `connectionTimer` yang check setiap 30 detik dan trigger reconnection jika detect disconnect.

Library `PubSubClient` menyediakan MQTT client implementation dengan dukungan QoS 0 dan 1. Callback `mqttCallback()` dipanggil dari interrupt context, sehingga processing harus minimal. Implementasi menggunakan queue untuk defer processing ke task context di mana blocking operation aman dilakukan. Topic subscription dilakukan dengan multiple `mqtt.subscribe()` calls untuk setiap topic yang diperlukan dan dilakukan setiap kali reconnect untuk memastikan subscription tetap aktif.

Komunikasi antara Node 1 dan Node 2 menggunakan MQTT sebagai message bus. Topic structure dirancang hierarkis dengan prefix `plant/` diikuti category (`sensor/` atau `control/`) dan data type specific.

Blynk menggunakan `Blynk.config()` untuk set credentials dan `Blynk.connect()` untuk establish connection dengan kredensial yang tersedia pada Blynk web. Blynk menggunakan `Blynk.run()` secara berkala dan Virtual pin write menggunakan `Blynk.virtualWrite(pin, value)`.

## CHAPTER 3

### TESTING AND EVALUATION

#### 3.1 TESTING

Pengujian sistem BLOOM (*Basic Local Observation & Optimized Moisture-control*) dilakukan untuk memverifikasi integrasi antara *hardware*, logika *embedded system* FreeRTOS, dan protokol MQTT yang terhubung ke platform Blynk. Skenario *testing* dibagi menjadi tahap konektivitas jaringan dan fungsionalitas logika *automatic control*.

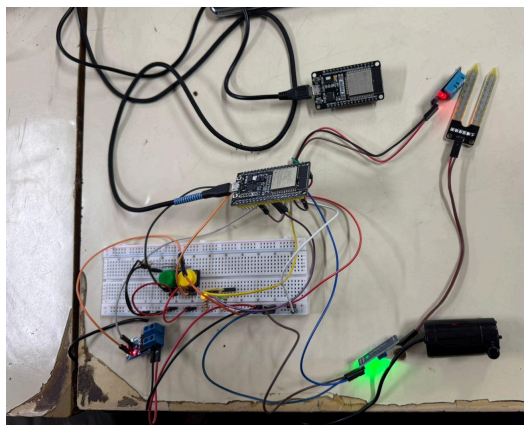
Perangkat keras yang diuji terdiri dari dua node ESP32 yang dirangkai seperti terlihat pada **Fig. 3.1**, yaitu:

- **Node 1 (Sensor dan Aktuator)**

Terhubung dengan sensor DHT11, *Soil Moisture Sensor*, pompa air DC yang dikendalikan MOSFET *Driver*, indikator LED, dan tombol manual.

- **Node 2 (Coordinator)**

Berfungsi sebagai gateway komunikasi data dari Node 1 ke server Blynk melalui koneksi WiFi dan MQTT.



**Fig. 3.1.** *Hardware Implementation Setup*

Pada tahap pengujian ini, sensor Soil Moisture diuji dalam kondisi *open air* (tanpa media tanah) untuk mensimulasikan kondisi kekeringan ekstrim dimana nilai kelembapan tanah 0%. Metode ini digunakan untuk mengamati respons sistem *auto-watering* pada kondisi kritis, serta memastikan bahwa data sensor dapat dikirimkan secara *real-time* dari Node Sensor ke Node *Coordinator* dan divisualisasikan pada *dashboard* Blynk.



Seluruh pengujian dipantau melalui Serial Monitor Arduino IDE untuk mengamati log sistem pada masing-masing node, serta melalui *dashboard* Blynk untuk memverifikasi transmisi dan visualisasi data.

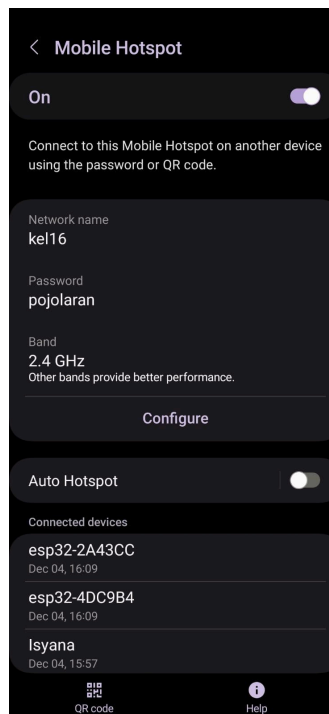
## 3.2 RESULT

Berdasarkan pengujian yang telah dilakukan, diperoleh beberapa hasil observasi terhadap kinerja sistem sebagai berikut:

### 3.2.1 Konektivitas Jaringan (*Infrastructure*)

Sebelum dilakukan pengujian fungsional sistem, stabilitas koneksi jaringan terlebih dahulu diverifikasi. Seperti ditunjukkan pada **Fig. 3.2**, *Mobile Hotspot* pada perangkat *gateway* berhasil mendeteksi dua perangkat ESP32 (Node 1 dan Node 2) yang terhubung secara bersamaan, serta satu perangkat *user* sebagai pemantau.

Hasil ini mengonfirmasi bahwa seluruh node dalam sistem **BLOOM** telah berhasil terkoneksi pada segmen jaringan lokal (*Local Area Network*) yang sama dengan SSID “kel16”, sehingga memungkinkan komunikasi data antar node berjalan dengan baik.

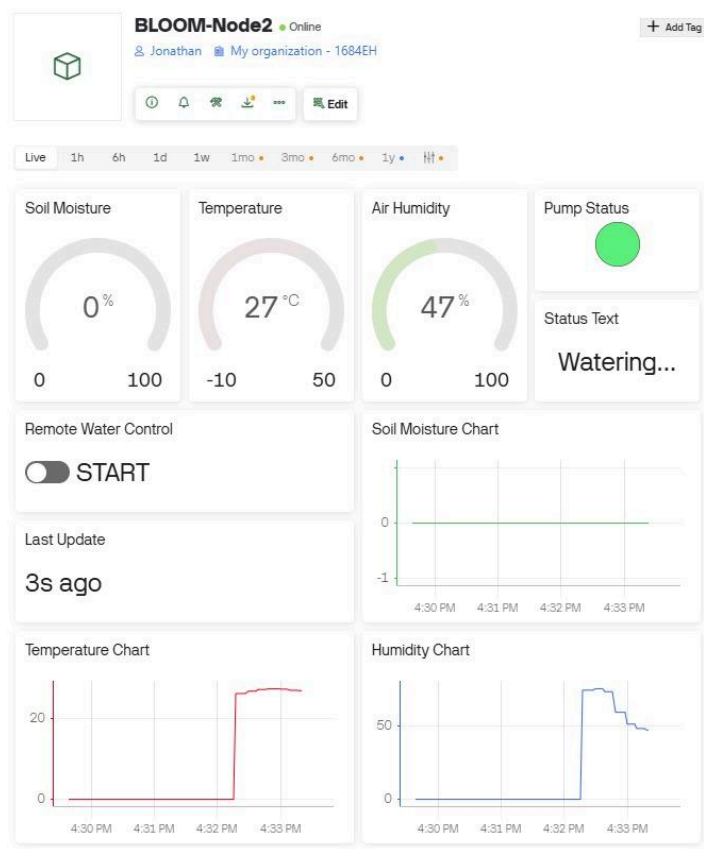


**Fig. 3.2.** *Mobile Hotspot Connected Devices*

### 3.2.2 Transmisi dan Visualisasi Data (*Software*)

Dengan koneksi jaringan yang stabil, Node 1 mampu mengirimkan data sensor ke Node 2 untuk selanjutnya diteruskan ke *platform* Blynk. Indikator keberhasilan proses ini dapat diamati pada **Fig. 3.3**, di mana dashboard Blynk menampilkan data suhu, kelembapan udara, dan kelembapan tanah secara *real-time*.

Grafik yang ditampilkan pada *dashboard* menunjukkan perubahan data yang responsif dan berkelanjutan, menandakan protokol MQTT berjalan stabil.



**Fig. 3.3.** *Real-time Blynk Dashboard*

### 3.2.3 Fungsionalitas *Automatic Control* (*Hardware Logic*)

Pada pengujian logika kontrol otomatis, ditemukan perilaku repetitif pada sistem akibat pengujian sensor *Soil Moisture* dalam kondisi *open air*, yang menyebabkan nilai sensor terbaca konstan pada 0%.

Saat sistem pertama kali dinyalakan, Node 1 langsung mendeteksi kondisi tanah sangat kering dan mengaktifkan pompa air secara otomatis, sebagaimana ditunjukkan pada **Fig. 3.4**.

```
=== ESP32-Node1 Smart Plant System ===
Mode: Sensor + Control (MQTT Only)
Connecting to WiFi.....
WiFi Connected!
IP: 10.208.244.62
Connecting to MQTT...Connected!
Subscribed to: plant/control/pump
Soil Moisture: 0%
AUTO: Soil too dry! Starting pump...
Published Soil: 0%
Pump Status Published: ON
Published Temp: 0.0°C
Published Humidity: 0.0%
--- Data sent to MQTT ---
```

**Fig. 3.4.** Node 1 Initialization Log

Karena tidak terdapat media tanah yang dapat meningkatkan nilai kelembapan setelah penyiraman, sensor tetap membaca nilai 0%. Kondisi ini menyebabkan sistem terjebak dalam siklus penyiraman berulang (*auto-watering loop*), di mana status pompa terus berganti antara ON dan OFF, seperti terlihat pada **Fig. 3.5**. Akibatnya, indikator LED pada perangkat juga berkedip mengikuti perubahan status pompa tersebut.

```
DHT: Temp=26.8°C, Hum=47.0%
Pump Status Published: OFF
AUTO: Pump stopped
Soil Moisture: 0%
AUTO: Soil too dry! Starting pump...
Pump Status Published: ON
DHT: Temp=26.6°C, Hum=47.0%
Pump Status Published: OFF
AUTO: Pump stopped
Soil Moisture: 0%
AUTO: Soil too dry! Starting pump...
Pump Status Published: ON
Published Soil: 0%
Published Temp: 26.6°C
Published Humidity: 47.0%
--- Data sent to MQTT ---
```

**Fig. 3.5.** Node 1 Auto-Watering Loop Log

### 3.2.4 Peran *Coordinator* (Node 2)

Meskipun terjadi *looping* pada Node 1, Node 2 terbukti mampu menangani beban komunikasi dengan baik. Pada **Fig. 3.6**, terlihat Node 2 tetap stabil menerima pembaruan status dari Node 1 dan menyinkronkannya ke Blynk tanpa *crash* atau *disconnect*.

[illegible]

**Fig. 3.6.** Node 2 Coordinator Log

### 3.3 EVALUATION

Berdasarkan hasil pengujian yang telah dilakukan, sistem BLOOM secara umum telah memenuhi kriteria penerimaan yang ditetapkan. Kedua node ESP32 berhasil terhubung ke jaringan WiFi dan broker MQTT dengan stabil, sehingga komunikasi data antar perangkat dapat berjalan dengan baik tanpa gangguan yang berarti.

Data sensor berupa kelembapan tanah, suhu, dan kelembapan udara berhasil dikirim dari Node 1 ke Node 2 dan ditampilkan secara real-time pada aplikasi Blynk. Hal ini menunjukkan bahwa integrasi antara MQTT dan Blynk telah berhasil diimplementasikan dengan baik.

Fungsi penyiraman otomatis juga berjalan sesuai dengan rancangan. Saat nilai kelembapan tanah berada di bawah ambang batas, pompa air akan aktif secara otomatis. Pada saat pengujian dengan kondisi sensor di udara terbuka, terjadi penyiraman berulang karena nilai kelembapan tanah tetap berada pada 0%. Hal ini bukan merupakan kesalahan sistem, melainkan disebabkan oleh kondisi pengujian yang tidak menggunakan media tanah sebenarnya.

Node 2 sebagai coordinator juga menunjukkan kinerja yang stabil, tetap mampu menerima dan meneruskan data ke Blynk meskipun Node 1 berada dalam kondisi penyiraman berulang. Hal ini membuktikan bahwa sistem multitasking dan komunikasi berbasis FreeRTOS berjalan dengan baik.

Namun, sistem masih memiliki beberapa keterbatasan, seperti penggunaan ambang batas dan durasi penyiraman yang masih bersifat tetap, serta belum dilakukan pengujian untuk penggunaan jangka panjang dan kondisi lingkungan luar ruangan.

## **CHAPTER 4**

### **CONCLUSION**

Sistem BLOOM menggunakan dua node ESP32, FreeRTOS sebagai pengelola tugas real-time, MQTT sebagai media komunikasi data, dan Blynk sebagai antarmuka pemantauan jarak jauh. Hasil pengujian menunjukkan bahwa sistem mampu membaca data sensor dengan baik, melakukan penyiraman otomatis berdasarkan kondisi kelembapan tanah, serta menampilkan data secara real-time melalui aplikasi Blynk. Selain itu, pengguna juga dapat melakukan penyiraman manual dari jarak jauh.

Arsitektur sistem terdistribusi antara Node 1 dan Node 2 terbukti mampu meningkatkan stabilitas sistem dan memudahkan proses pemantauan serta pengendalian. Sistem ini juga menunjukkan kinerja yang andal selama pengujian berlangsung.

Meskipun demikian, sistem masih dapat dikembangkan lebih lanjut dengan menambahkan fitur seperti pengaturan otomatis berdasarkan jenis tanaman, pencatatan data historis, serta optimalisasi penggunaan daya untuk implementasi jangka panjang. Dengan demikian, sistem BLOOM dapat menjadi solusi yang efektif untuk membantu perawatan tanaman secara otomatis, efisien, dan berbasis teknologi IoT.

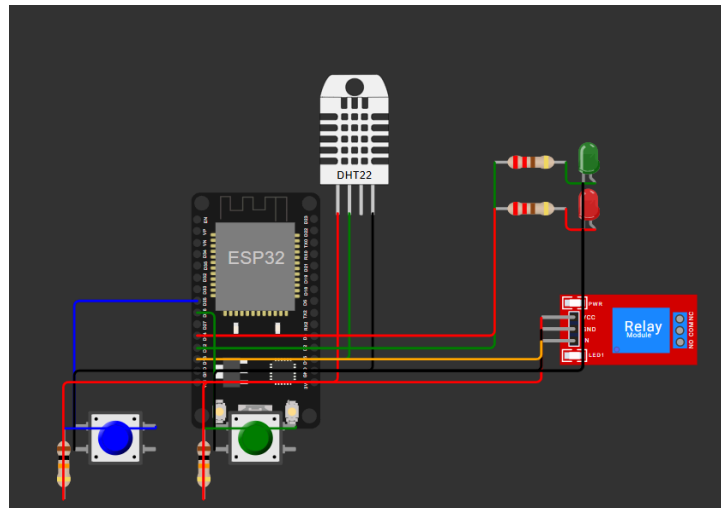
## REFERENCES

- [1] Espressif Systems, “ESP32 Series Datasheet,” Espressif. [Online]. Available: [https://www.espressif.com/sites/default/files/documentation/esp32\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf). (Accessed: Dec. 06, 2025)
- [2] FreeRTOS, “The FreeRTOS™ Kernel,” FreeRTOS.org. [Online]. Available: <https://www.freertos.org/RTOS.html>. (Accessed: Dec. 06, 2025)
- [3] MQTT.org, “MQTT: The Standard for IoT Messaging,” MQTT.org. [Online]. Available: <https://mqtt.org/>. (Accessed: Dec. 06, 2025)
- [4] Blynk Inc., “Blynk Documentation: Getting Started,” Blynk.io. [Online]. Available: <https://docs.blynk.io/en/>. (Accessed: Dec. 06, 2025)
- [5] Components101, “DHT11 Temperature and Humidity Sensor Datasheet,” Components101. [Online]. Available: <https://components101.com/sensors/dht11-temperature-sensor>. (Accessed: Dec. 06, 2025)
- [6] DFRobot, “Capacitive Soil Moisture Sensor V1.2 Wiki,” DFRobot. [Online]. Available: [https://wiki.dfrobot.com/Capacitive\\_Soil\\_Moisture\\_Sensor\\_SKU\\_SEN0193](https://wiki.dfrobot.com/Capacitive_Soil_Moisture_Sensor_SKU_SEN0193). (Accessed: Dec. 06, 2025)
- [7] HiveMQ, “MQTT Essentials Part 1: Introducing MQTT,” HiveMQ. [Online]. Available: <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>. (Accessed: Dec. 06, 2025)

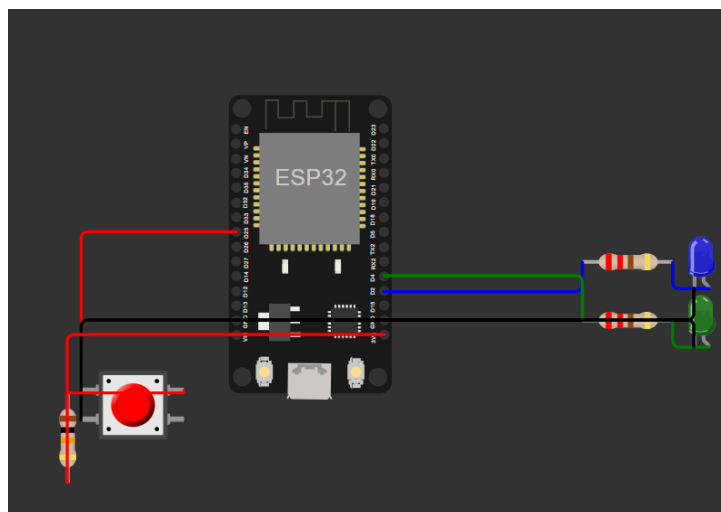
## APPENDICES

### Appendix A: Project Schematic

#### Node 1



#### Node 2





## Appendix B: Documentation

