LAPORAN PROJECT III

SMART AIR FRESHENER

Dosen Pembimbing: **Muhammad Ilham Kurniawan, M. Kom**

Disusun Oleh: **KIMAS (Koalisi Indonesia Emas)**

Anggota:

1. **Ananda Syahrul Ramadhan** **1122140054**
2. **Muhammad Fauzan Ardiansyah** **1122460012**
3. **Muhamad Fiqri Haikal** **1122140104**
4. **Raja Agil Husein Harahap** **1122140078**

INSTITUT TEKNOLOGI DAN BISNIS BINA SARANA GLOBAL

FAKULTAS TEKNIK INFORMASI DAN KOMUNIKASI

PROGRAM STUDI TEKNIK INFORMATIKA

KONSENTRASI COMPUTER NETWORKING

2025

LEMBAR PENGESAHAN

Laporan Proyek dengan judul: “Smart Air Freshener”

Disusun untuk memenuhi salah satu syarat dalam menyelesaikan Tugas Akhir semester VI (Enam) pada Program Studi Teknik Informatika.

Disusun oleh:

1. **Ananda Syahrul Ramadhan** **1122140054**
2. **Muhammad Fauzan Ardiansyah** **1122460012**
3. **Muhamad Fiqri Haikal** **1122140104**
4. **Raja Agil Husein Harahap** **1122140078**

Program Studi : Teknik Informatika

Konsentrasi : Computer Networking

Disahkan Oleh,

Tangerang, 2025

Ka. Prodi Teknik Informatika,	Dosen Penguji
Nunung Nurmaesah, M.Kom **NUPTK. 8744766667230332**	**Muhammad Ilham Kurniawan, M. Kom**

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan proyek yang berjudul **"Smart Air Freshener"** dengan baik dan tepat waktu.

Laporan ini disusun sebagai salah satu bentuk dokumentasi dari kegiatan proyek 3 yang bertujuan untuk merancang sebuah alat penyemprot ruangan otomatis berbasis mikrokontroler ESP32. Alat ini dirancang agar dapat bekerja secara otomatis maupun manual, serta dapat dikendalikan melalui aplikasi Blynk sebagai bentuk penerapan teknologi "**Internet of Things (IoT)"** dalam kehidupan sehari-hari. Proyek ini diharapkan dapat menjadi solusi inovatif dalam menjaga kesegaran udara ruangan secara efisien dan praktis.

Dalam penyusunan laporan ini, penulis menyadari bahwa hasil yang dicapai tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada:

- Dosen pembimbing yang telah memberikan arahan, bimbingan, dan masukan selama proses pengerjaan proyek ini.
- Teman-teman yang telah membantu baik secara teknis maupun non-teknis dalam pelaksanaan proyek.
- Serta semua pihak yang turut memberikan dukungan secara langsung maupun tidak langsung.

Kami menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan untuk perbaikan di masa mendatang. Semoga laporan ini dapat memberikan manfaat dan menjadi referensi bagi pembaca yang tertarik pada bidang teknologi IoT.

Tangerang, 23 Juni 2025

Penulis

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kenyamanan dalam ruangan sangat dipengaruhi oleh kualitas udara, termasuk aroma yang tercium oleh penghuni. Udara yang segar dan wangi dapat menciptakan suasana yang lebih menyenangkan dan meningkatkan produktivitas, baik di rumah, kantor, maupun ruang publik. Oleh karena itu, penggunaan air freshener sudah menjadi hal yang umum dalam kehidupan sehari-hari.

Namun, kebanyakan air freshener yang beredar di pasaran masih bersifat manual atau hanya menyemprot secara berkala tanpa mempertimbangkan kondisi aktual di lingkungan sekitar. Hal ini bisa menyebabkan pemborosan cairan pewangi, kurangnya fleksibilitas, dan ketergantungan pada interaksi pengguna.

Melalui proyek ini, dirancang sebuah perangkat berbasis mikrokontroler ESP32 yang bernama “Smart Air Freshener” yang dapat bekerja secara otomatis dan juga dikendalikan secara jarak jauh melalui koneksi internet menggunakan aplikasi Blynk. Dengan fitur ini, pengguna dapat:

- Menyetel jadwal penyemprotan otomatis.
- Melakukan penyemprotan manual melalui tombol fisik atau aplikasi.
- Melakukan penyemprotan Ketika mode PIR diaktifkan.
- Menerima notifikasi ketika menyemprot melalui tombol fisik, jadwal, dan sensor PIR.

Proyek ini tidak hanya bertujuan untuk meningkatkan kenyamanan dan efisiensi penggunaan perangkat, tetapi juga sebagai implementasi nyata dari teknologi Internet of Things (IoT) dalam kehidupan sehari-hari. Dengan memadukan aspek fungsional dan teknologi, Smart Air Freshener menjadi solusi inovatif untuk mengelola kualitas udara secara cerdas dan efisien.

1.2 Rumusan Masalah

1. Bagaimana merancang sistem penyemprot ruangan otomatis yang dapat bekerja secara terjadwal tanpa perlu intervensi manual dan dapat diatur oleh pengguna?
2. Bagaimana merancang sistem air freshener otomatis yang dapat dikontrol secara jarak jauh menggunakan aplikasi Blynk melalui koneksi internet?
3. Bagaimana merancang sistem yang memungkinkan pengguna mengaktifkan semprotan secara manual menggunakan tombol fisik maupun tombol virtual pada aplikasi?
4. Bagaimana mengintegrasikan sensor PIR untuk memungkinkan sistem menyemprot hanya ketika mendeteksi pergerakan di sekitar area tertentu?
5. Bagaimana memastikan sistem dapat berfungsi dalam kondisi online maupun offline?

1.3 Batasan Masalah

1. Sistem Smart Air Freshener ini dirancang dengan menggunakan mikrokontroler ESP32 sebagai pusat kendali utama yang mengatur seluruh fungsi perangkat.
2. Aplikasi kontrol jarak jauh hanya menggunakan platform Blynk, tidak mencakup integrasi dengan platform IoT lain seperti MQTT atau Home Assistant.
3. Sensor gerak yang digunakan adalah sensor PIR, yang hanya mendeteksi keberadaan gerakan tanpa membedakan jenis objek atau arah pergerakan.
4. Notifikasi dan kontrol hanya dapat dilakukan ketika perangkat terhubung ke jaringan WiFi yang stabil.
5. Tombol fisik hanya berfungsi untuk mengaktifkan semprotan secara manual, tidak untuk mengubah mode atau jadwal.
6. Sistem tidak dilengkapi dengan sistem pengisian ulang otomatis atau pendeteksi level cairan pengharum ruangan, hanya mengandalkan waktu/jadwal dan input pengguna.
7. Karena sistem bergantung pada jaringan Wi-Fi yang stabil, fitur kontrol melalui aplikasi tidak dapat digunakan saat perangkat tidak terhubung WiFi Perangkat ini ditujukan untuk penggunaan skala kecil, seperti ruang pribadi/kamar, atau ruang tamu, dan belum dioptimalkan untuk penggunaan komersial berskala besar. Pengujian dilakukan dalam ruang tertutup dengan jangkauan WiFi yang stabil, sehingga tidak mencakup kondisi ekstrem seperti area luar ruangan, atau sinyal WiFi lemah.

1.4 Tujuan Penulisan Laporan

1. Menjelaskan implementasi aplikasi Blynk sebagai media kontrol jarak jauh, termasuk pengaturan jadwal penyemprotan dan mengirim notifikasi ke perangkat smartphone.
2. Menjelaskan proses perancangan dan pembuatan sistem Smart Air Freshener yang dapat bekerja secara otomatis maupun manual.
3. Menguji dan mengevaluasi kinerja sistem Smart Air Freshener dalam kondisi nyata untuk memastikan sistem bekerja sesuai dengan fungsi dan tujuan yang diharapkan.
4. Mengimplementasikan fitur penjadwalan semprotan otomatis berdasarkan waktu yang dapat diatur pengguna sesuai kebutuhan.
5. Menyediakan kontrol manual semprotan melalui tombol fisik dan tombol virtual pada aplikasi sebagai alternatif pengendalian.
6. Menerapkan sensor PIR untuk mengaktifkan semprotan secara otomatis saat mendeteksi pergerakan di sekitar perangkat.
7. Menguji dan mengevaluasi kinerja sistem dalam berbagai kondisi untuk memastikan keandalan fungsi otomatis maupun manual.

1.5 Metode Pengumpulan Data

1. Studi Literatur

   Pengumpulan informasi dilakukan dengan membaca dan menganalisis berbagai sumber seperti buku, jurnal ilmiah, artikel online, dan dokumentasi teknis terkait mikrokontroler ESP32 dan aplikasi Internet of Things (IoT) menggunakan Blynk. Studi ini bertujuan untuk memahami konsep dasar dan teknologi yang akan digunakan dalam sistem.

2. Observasi

   Observasi dilakukan selama proses perancangan dan pengujian alat untuk mengamati bagaimana sistem bekerja dalam kondisi nyata. Data yang diamati meliputi waktu respon sistem, keefektifan penyemprotan, dan kestabilan koneksi dengan aplikasi Blynk.

3. Eksperimen/Pengujian Langsung

   Pengujian dilakukan dengan menjalankan alat dalam berbagai skenario, seperti penyemprotan otomatis berdasarkan jadwal, penyemprotan manual dari aplikasi, serta kondisi saat cairan hampir habis. Data hasil pengujian ini digunakan untuk mengevaluasi performa dan fungsionalitas sistem secara keseluruhan.

4. Diskusi

Diskusi dilakukan dengan dosen pembimbing, teman satu tim, atau pihak yang memahami teknologi IoT dan sistem otomasi untuk mendapatkan masukan serta saran perbaikan selama proses pengembangan.

BAB II
LANDASAN TEORI

2.1 Pengertian Internet Of Things (IoT) Dan Cara Kerjanya

Internet of Things (IoT) adalah konsep di mana berbagai perangkat, seperti sensor, perangkat elektronik, dan objek lainnya, terhubung dan berkomunikasi melalui jaringan internet. Dengan IoT, pengguna dapat terkoneksi untuk melakukan berbagai aktivitas, mulai dari pencarian informasi hingga pengolahan data, tanpa perlu campur tangan manusia.

Konsep IoT sepintas hampir serupa dengan Machine-to-Machine (M2M), namun sebenarmya kedua konsep ini memiliki perbedaan signifikan dalam skala dan lingkup penggunaannya. M2M fokus pada komunikasi antara mesin tanpa intervensi manusia, seperti mesin pabrik yang berkoordinasi secara otomatis untuk meningkatkan efisiensi produksi.

Meskipun berbeda, kedua konsep ini sering digunakan bersamaan. Hal ini disebabkan karena tujuan dari IoT dan M2M adalah sama-sama membangun sebuah komunikasi yang terhubung secara otomatis untuk meningkatkan efisiensi.

Dalam proyek ini, IoT digunakan untuk memungkinkan kendali dan pemantauan air freshener secara online melalui aplikasi Blynk.

2.1.1 Cara Kerja Internet Of Things (IoT)

Internet of Things (IoT) bekerja dengan menghubungkan perangkat keras dan lunak ke internet melalui alamat IP unik, sehingga tiap perangkat dapat saling berkomunikasi. Perangkat seperti sensor dan kamera mengumpulkan data, lalu mengirimkannya ke cloud untuk dianalisis. Data ini kemudian digunakan untuk mengambil tindakan otomatis, seperti menyalakan AC saat suhu ruangan terlalu panas.

2.2 Sejarah Singkat Internet of Things (IoT)

Perkembangan IoT (Internet of Things) dimulai dari kemajuan teknologi nirkabel, sensor, dan *microelectromechanical systems* (MEMS) pada akhir 1980-an. Salah satu implementasi awal adalah toaster yang dikendalikan lewat internet oleh John Romkey dan Simon Hackett pada 1989.

Meski istilah “IoT” baru dikenalkan oleh “Kevin Ashton” pada 1999, konsep perangkat yang saling terhubung sudah ada sejak “1832” melalui telegraf elektromagnetik, dan berkembang pesat setelah penemuan internet di akhir 1960-an. **J.C.R. Licklider** membayangkan jaringan komputer global (ARPANET) yang menjadi dasar internet pada 1969.

Contoh awal implementasi IoT dalam kehidupan sehari-hari adalah “WearCam” oleh Steve Mann (1994), teknologi wearable pertama. Pada 1997, “Paul Saffo” memperkenalkan teknologi sensor yang memungkinkan perangkat mendeteksi dan merespons lingkungan fisik.

Pada 1999, Ashton juga mempopulerkan teknologi “RFID” yang memungkinkan pelacakan otomatis tanpa interaksi manusia, menjadi salah satu fondasi penting IoT modern.

2.2.1 Linimasa sejarah IoT

Perangkat IoT pertama di dunia ditemukan pada awal 1980-an di Universitas Carnegie Melon, USA. Sekelompok mahasiswa di universitas tersebut menciptakan cara agar mesin penjual otomatis Coca-Cola di kampus mereka dapat melaporkan isinya melalui sebuah jaringan sehingga mereka dapat mengetahui jika Coca-Cola di mesin tersebut sudah habis atau belum, dan mereka tidak perlu untuk menghabiskan energi mereka untuk datang ke mesin tersebut. Mereka memasang sakelar mikro ke dalam mesin tersebut untuk melaporkan berapa banyak kaleng Coca-Cola yang tersedia.

2.3 Pengertian Aplikasi Blynk

Blynk adalah platform IoT yang memungkinkan pengguna mengontrol dan memantau perangkat dari jarak jauh melalui antarmuka aplikasi smartphone. Dalam proyek ini, Blynk digunakan sebagai penghubung antara pengguna dan Smart Air Freshener.

Blynk adalah platform Internet of Things (IoT) yang memungkinkan pengembang membuat aplikasi IoT dengan mudah tanpa perlu pengetahuan pemrograman yang mendalam. Dengan menggunakan aplikasi atau platform Blynk Anda dapat dengan cepat membuat antarmuka pengguna yang interaktif untuk mengontrol dan memantau perangkat IoT Anda melalui smartphone atau tablet Anda.

2.4 Sejarah Aplikasi Blynk

Blynk merupkan platform untuk aplikasi OS Mobile (iOS dan Android) yang digunakan untuk melakukan pengendalian pada modul Arduino, Raspberry Pi, ESP8266, dan modul sejenisnya melalui koneksi internet. Sejak diluncurkan pada tahun 2014, aplikasi ini menjadi aplikasi yang banyak digunakan karena menawarkan kemudahan dan memberikan kesempatan bagi pengguna berkreasi membuat tampilan grafis antarmuka dengan metode drag and drop widget yang disediakan.

Pada awal kemunculannya, Blynk hadir dengan aplikasi bernama Blynk Legacy. Selanjutnya, pada bulan Mei 2021 pihak pengembang memberikan pengumuman resmi bahwa aplikasi tersebut tidak lagi dikembangkan. Sebagai gantinya, pengembang aplikasi meluncurkan Blynk 2.0 atau Blynk IOT.

2.5 Pengertian Microcontroller

Mikrokontroler adalah komputer kecil dalam bentuk chip (IC) yang dirancang untuk menjalankan tugas tertentu, seperti menerima input (misalnya dari sensor), mengolahnya, dan menghasilkan output (misalnya ke aktuator). Mikrokontroler terdiri dari CPU, RAM, ROM, serta port input/output yang bisa diprogram.

Meski kemampuannya lebih rendah dibanding komputer (kecepatan 1–16 MHz dan memori dalam KByte), mikrokontroler cukup efisien untuk digunakan pada perangkat sederhana dan tidak memerlukan komputasi tinggi. Contoh penggunaannya meliputi berbagai sistem otomatisasi dan perangkat elektronik, seperti Arduino dan Atmega328.

2.6 Fungsi Mikrokontroler

1. Sebagai timer atau pewaktu
2. Sebagai pembangkit osilasi
3. Sebagai Flip-flop
4. Sebagai ADC (Analog to Digital Converter)
5. Sebagai counter atau penghitung
6. Sebagai decoder dan encoder

2.7 Sejarah Singkat Microcontroller

- Awal Mula (1970-an):

  Mikrokontroler pertama, Intel 4004 (1971), dirancang untuk kalkulator. Kemudian muncul Intel 8048 (1976) dan Motorola MC6801 yang mulai digunakan di industri.

- 1980-an:

  Mikrokontroler mulai digunakan dalam otomotif, telekomunikasi, dan rumah tangga. Muncul produk seperti PIC dari Microchip dan AVR dari Atmel.

- 1990-an:

  Teknologi semikonduktor berkembang, memungkinkan mikrokontroler 16-bit dan 32-bit. ARM muncul sebagai arsitektur dominan berkat efisiensinya.

- Abad 21 (Era IoT):

  Mikrokontroler dilengkapi fitur seperti Wi-Fi, Bluetooth, dan sensor. Espressif (ESP8266/ESP32) dan STMicroelectronics (STM32) menjadi pemain utama.

- Kemajuan Terkini:
  - AI & Machine Learning: Mikrokontroler kini mampu menjalankan algoritma cerdas.
  - Efisiensi Energi: Fokus pada daya rendah untuk perangkat IoT (contoh: ARM Cortex-M0+, TI MSP430).
  - Keamanan: Mikrokontroler modern dilengkapi enkripsi dan fitur keamanan tinggi.

- Aplikasi Luas:
  Digunakan dalam otomasi, perangkat medis, kendaraan otonom, dan perangkat pintar lainnya.

- Tantangan dan Masa Depan:
  Fokus pada keamanan, efisiensi daya, dan integrasi lebih tinggi. Mikrokontroler akan semakin penting dalam dunia digital dan perangkat cerdas.

2.8 Jenis – Jenis Mikrokontroler

1. Arduino

   Arduino adalah salah satu microcontroller yang paling populer di dunia. Dirancang untuk memudahkan pengembangan prototipe perangkat elektronik, Arduino memiliki komunitas yang besar dan beragam. Salah satu keunggulan utama Arduino adalah kemudahan penggunaannya. Dengan bahasa pemrograman yang sederhana dan banyaknya tutorial yang tersedia, bahkan pemula sekalipun dapat dengan cepat memulai proyek-proyek elektronik mereka. Selain itu, Arduino memiliki berbagai jenis board yang sesuai dengan kebutuhan, mulai dari yang sederhana hingga yang lebih canggih

2. Raspberry Pi

   Meskipun bukan microcontroller dalam arti sebenarnya, Raspberry Pi layak disebut dalam konteks ini karena kemampuannya yang mirip dengan microcontroller. Raspberry Pi adalah sebuah komputer kecil yang dapat dihubungkan dengan berbagai sensor dan perangkat lainnya. Kelebihan utama Raspberry Pi adalah fleksibilitasnya yang tinggi. Selain dapat digunakan untuk proyek-proyek elektronik, Raspberry Pi juga dapat dijadikan sebagai media center, server, atau bahkan komputer desktop alternatif. Dengan harga yang terjangkau, Raspberry Pi menjadi pilihan yang populer bagi para pengembang dan hobiis.

3. ESP8266 dan ESP32

   ESP8266 dan ESP32 merupakan microcontroller yang didesain khusus untuk aplikasi IoT (Internet of Things). Kedua microcontroller ini memiliki kemampuan WiFi yang memungkinkan mereka terhubung dengan internet secara langsung. Hal ini membuat mereka sangat cocok digunakan untuk proyek-proyek IoT,

seperti monitoring lingkungan, kontrol perangkat rumah tangga, dan lain sebagainya. Keunggulan utama dari ESP8266 dan ESP32 adalah kemampuan konektivitasnya yang tinggi dan konsumsi daya yang rendah, sehingga cocok untuk digunakan dalam perangkat bertenaga baterai.

4. PIC Microcontroller

   PIC (Peripheral Interface Controller) merupakan salah satu jenis microcontroller yang banyak digunakan dalam industri. Microcontroller ini dikenal akan kehandalannya dan tersedia dalam berbagai varian yang sesuai dengan kebutuhan aplikasi tertentu. Kelebihan dari PIC Microcontroller adalah kemampuannya dalam mengendalikan perangkat eksternal secara efisien. Dengan arsitektur yang canggih dan banyaknya fitur yang tersedia, PIC Microcontroller sering digunakan dalam berbagai sistem kontrol industri, otomotif, dan elektronik konsumen.

5. STM32

   STM32 adalah keluarga microcontroller yang dikembangkan oleh STMicroelectronics. Microcontroller ini sangat populer di kalangan pengembang karena performa yang tinggi dan beragamnya fitur yang ditawarkan. Kelebihan utama dari STM32 adalah kecepatan prosesnya yang tinggi dan konsumsi daya yang rendah. Selain itu, STM32 juga dilengkapi dengan berbagai fitur koneksi seperti USB, Ethernet, dan Bluetooth, membuatnya sangat cocok untuk aplikasi-aplikasi yang membutuhkan koneksi yang stabil dan cepat.

6. Mikrokontroler AVR

   Mikrokontroler AVR (Alf and Vegard’s Risc Processor) merupakan salah satu komponen yang digunakan pada bidang instrumentasi dan elektronika. AVR diambil dari nama Alf Egil Bogen dan Vegard Wollan yang merupakan penemu berkebangsaan Norwegia. AVR dilengkapi dengan arsitektur RISC (Reduce Instruction Set Computing) yang memungkinkan AVR bisa menjalankan berbagai instruksi hanya 1 siklus, kecuali instruksi percabangan yang memerlukan 2 siklus. AVR diproduksi oleh perusahaan bernama Atmel yang kini sudah ada 10 kelas,

Yaitu:

- AVR LCD
- AVR CAN
- AVR Otomotif
- AVR USB
- TinyAVR
- Mega AVR
- XMEGA
- AVR Z-Link
- AVR Manajemen Batere
- AVR Pencahayaan

Pada dasarnya, masing-masing kelas di atas memiliki arsitektur dan instruksi yang hampir sama. Hanya saja berbeda dalam fungsinya, kapasitas memori dan peripheral. Seri AVR yang paling banyak digunakan yaitu Attiny2313, mikrokontroler Atmega8535 dan mikrokontroler Arduino.

7. Mikrokontroler MCS 51

   MCS 51 merupakan produksi dari ATMEL, sama seperti AVR. MCS-51 dibuat dalam dua versi yaitu 40 kaki dan 20 kaki. Secara umum, keduanya memiliki arsitektur yang sama. Perbedaan utamanya pada bagian kapasitas memori-data, memori-program dan jumlah pewaktu 16 bit.

8. Mikrokobtroler ARM

   ARM (Advanced RISC Machine, sebelumnya Acorn RISC Machine) merupakan keluaran dari ARM Holding sebagai prosesor yang memiliki arsitektur RISC dengan set instruksi 32 bit. Awalnya, ARM dikembangkan oleh Acorn Computers. Saat itu, pengembangan ARM difungsikan untuk Personal Computer (PC).

2.9 Pengertian Mikrokontroler ESP32

Mikrokontroler ESP32 dibuat oleh perusahaan bernama Espressif Systems. Salah satu kelebihan yang dimiliki oleh ESP32 yaitu sudah terdapat Wi-Fi dan Bluetooth di dalamnya, sehingga akan sangat memudahkan ketika kita belajar membuat sistem IoT yang memerlukan koneksi wireless.Mikrokontroler ESP32 memiliki keunggulan yaitu sistem berbiaya rendah, dan juga berdaya rendah dengan modul WiFi yang terintegrasi dengan chip mikrokontroler serta memiliki bluetooth dengan mode ganda dan fitur hemat daya menjadikannya lebih fleksibel.

ESP32 adalah mikrokontroler yang dikenalkan oleh Espressif System merupakan penerus dari mikrokontroler ESP8266. Pada mikrokontroler ini sudah tersedia modul WiFi dalam chip sehingga sangat mendukung untuk membuat sistem aplikasi Internet of Things. ESP32 sendiri tidak jauh berbeda dengan ESP8266 yang familiar di pasaran, hanya saja ESP32 lebih komplek dibandingkan ESP8266, cocok untuk sobat dengan proyek yang besar.

Pada project ini Mikrokontroler ESP32 digunakan sebagai otak dari sistem Smart Air Freshener untuk mengontrol sensor, motor, tombol, indikator led, serta komunikasi dengan aplikasi.

2.10 Cara Kerja Microcontroller ESP32

ESP32 diprogram menggunakan bahasa C/C++ melalui Arduino IDE atau lingkungan pengembangan lainnya. Kode program (disebut juga sketch) berisi instruksi dan algoritma yang akan diunggah ke ESP32 untuk mengontrol berbagai fungsi seperti mengaktifkan Wi-Fi, mengirim data, membaca sensor, atau mengendalikan perangkat lain.

ESP32 memiliki banyak pin GPIO (General Purpose Input/Output) yang dapat dikonfigurasi untuk berinteraksi dengan berbagai sensor dan aktuator. Pin ini memungkinkan ESP32 membaca data dari sensor (seperti suhu, kelembaban, tekanan) atau mengontrol aktuator (seperti LED, motor, atau relay). Selain itu, ESP32 dapat berkomunikasi dengan perangkat lain secara nirkabel melalui Wi-Fi atau Bluetooth, serta terhubung melalui antarmuka kabel seperti SPI, I2C, SDIO, dan UART.

Dalam komunikasi nirkabel, ESP32 dapat bekerja dalam dua mode Wi-Fi: mode Station (terhubung ke jaringan yang ada) dan mode Access Point (membuat jaringan sendiri). Selain itu, ESP32 juga mendukung Bluetooth klasik dan Bluetooth Low Energy (BLE), yang memungkinkan komunikasi dengan berbagai perangkat Bluetooth.

ESP32 memiliki dua mode operasi utama, yaitu Normal Mode, di mana mikrokontroler bekerja penuh untuk menjalankan program dan berinteraksi dengan perangkat lain, dan Deep Sleep Mode, yaitu mode hemat daya yang mematikan sebagian besar fungsi ESP32 untuk menghemat energi, sangat cocok untuk aplikasi berbasis baterai.

ESP32 banyak digunakan dalam berbagai proyek IoT, seperti pemantauan lingkungan, rumah pintar, dan otomatisasi industri. Salah satu varian populernya, ESP32-CAM, bahkan dilengkapi kamera untuk aplikasi pemantauan dan pengawasan. Secara keseluruhan, ESP32 bekerja dengan menjalankan kode yang mengontrol fungsi perangkat kerasnya serta berinteraksi dengan perangkat lain melalui koneksi nirkabel (Wi-Fi/Bluetooth) maupun kabel, menjadikannya pilihan tepat untuk aplikasi IoT yang fleksibel dan efisien.

2.11 Pengertian Air Freshener

Pengharum udara atau penyegar ruangan adalah produk konsumen yang biasanya mengeluarkan wewangian dan digunakan di dalam rumah atau ruangan komersial seperti toilet, serambi, lorong, vestibula, dan area dalam ruangan kecil lainnya, serta area yang lebih luas seperti lobi hotel, dealer mobil, fasilitas medis, arena publik, dan ruangan besar lainnya. Pengharum mobil digunakan di mobil. Sebagai sumber aroma, pengharum udara dibuat dari bahan penghilang bau khusus untuk toilet dan urinoir.

Ada beragam metode dan merek pengharum udara. Beberapa jenis pengharum udara di antaranya adalah pengharum udara kipas listrik, pengharum udara semprot, penyebar aroma evaporasi mekanis, dispenser pengharum berwaktu, semprotan, lilin, minyak, gel, manik-manik, dan colokan.

Beberapa pengharum udara mengandung bahan kimia yang bisa memicu gejala alergi dan asma atau bersifat racun. Pengharum udara tidak hanya terbatas pada jenis semprotan, tetapi juga mencakup penggunaan bahan organik dan barang-barang rumah tangga sehari-hari. Meskipun pengharum udara umumnya digunakan untuk menghilangkan bau tidak sedap, sejumlah orang juga menggunakan pengharum udara karena bau harum yang dikeluarkannya.

BAB III
TINJAUAN OBJEK YANG DITELITI

3.1 Gambaran Umum Proyek

Proyek ini bertujuan untuk merancang dan mengimplementasikan sebuah alat penyemprot ruangan pintar (Smart Air Freshener) berbasis mikrokontroler ESP32 yang dapat bekerja secara otomatis dan dikendalikan melalui internet menggunakan aplikasi Blynk. Perangkat ini dirancang untuk meningkatkan efisiensi dan kenyamanan dalam menjaga aroma ruangan agar tetap segar dan menyenangkan.

3.2 Lokasi dan Kondisi Pengujian

Pengujian sistem dilakukan di ruangan tertutup seperti kamar tidur atau ruang kerja dengan ukuran standar (sekitar 3x3 meter). Pemilihan lokasi ini didasarkan pada ruang lingkup penggunaan alat yang ditujukan untuk kebutuhan rumah tangga atau personal space. Ruangan memiliki koneksi Wi-Fi stabil sebagai syarat utama untuk fungsi kendali jarak jauh melalui aplikasi Blynk.

3.3 Fitur Yang Diteliti

Fitur-fitur utama yang menjadi fokus penelitian dan pengembangan dalam proyek ini antara lain:

- Penyemprotan otomatis berdasarkan jadwal melalui aplikasi.
- Penyemprotan manual melalui aplikasi atau tombol fisik.
- Penyemprotan jika sensor PIR mendeteksi adanya pergerakan.
- Notifikasi saat perangkat melakukan penyemprotan melalui jadwal, tombol fisik, dan sensor PIR.
- Indikator WiFi jika perangkat terhubung atau tidak terhubung WiFi.
- Buzzer sebagai indicator bahwa perangkat akan melakukan penyemprotan.
- Kontrol jarak jauh melalui aplikasi Blynk melalui konektivitas WiFi yang stabil.
- Memiliki dua mode, yaitu : Mode manual, dan Mode PIR
- Perangkat dapat menyemprot melalui tombol fisik Ketika perangkat offline.

3.4 Tujuan Penelitian Objek

Tujuan dari penelitian terhadap objek-objek tersebut adalah untuk memastikan bahwa setiap komponen sistem bekerja secara optimal dan saling terintegrasi dengan baik, sehingga menghasilkan sebuah alat yang fungsional, efisien, dan mudah digunakan oleh pengguna.

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi Sistem

Implementasi sistem merupakan tahap di mana semua rancangan dan perencanaan mulai direalisasikan dalam bentuk perangkat keras (hardware) dan perangkat lunak (software) yang terintegrasi. Smart Air Freshener ini dibangun menggunakan mikrokontroler ESP32 sebagai otak pengendali, motor DC untuk aktuasi penyemprot, tombol fisik, buzzer, LCD 16x2 untuk menampilkan informasi, sensor PIR, LED untuk indikator WiFi dan aplikasi Blynk untuk kontrol jarak jauh.

4.2 Perangkat Keras Yang Digunakan

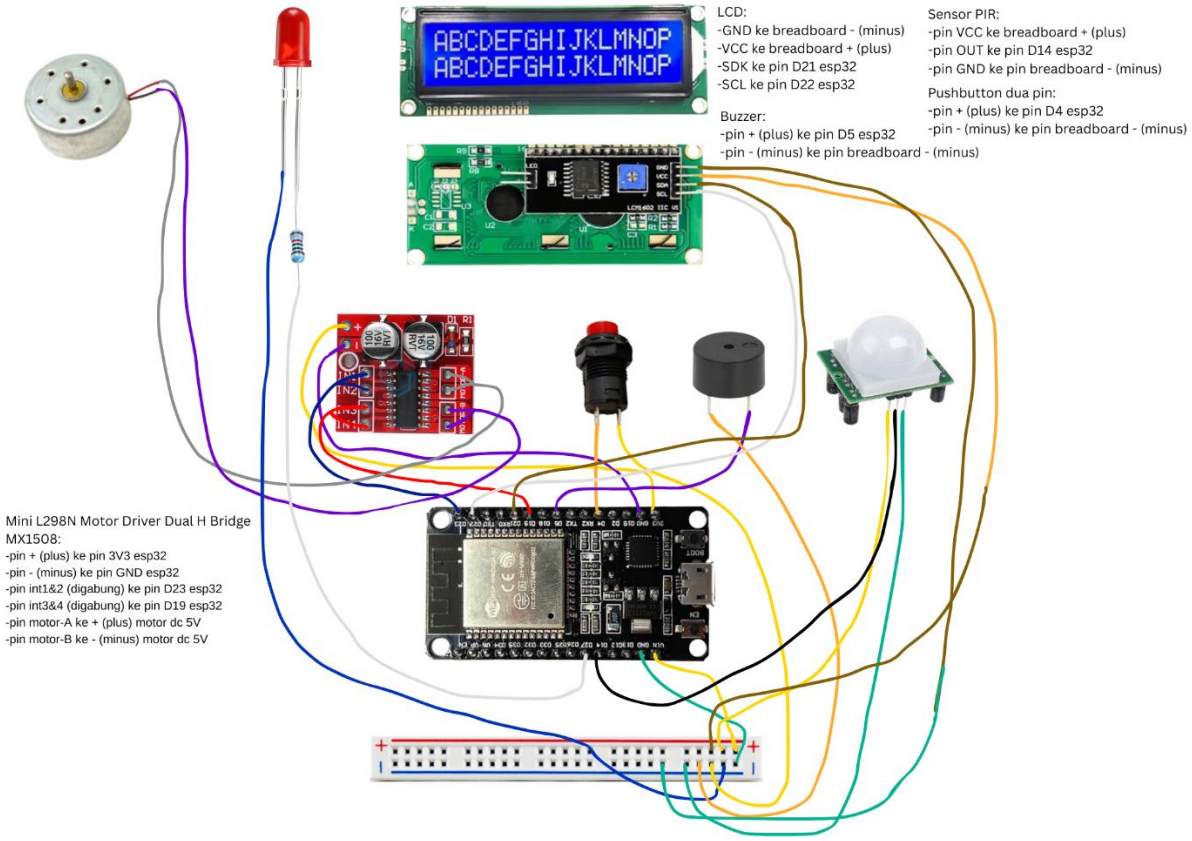
Berikut adalah komponen utama dalam perangkat:

- ESP32 : Sebagai pusat kendali system/otak perangkat, mengatur input-output dari seluruh sensor, LED, LCD, Buzzer, Button dan aktuator.
- Motor DC : Menggerakkan mekanisme penyemprot untuk menyemprotkan cairan ke udara.
- Tombol Fisik : Digunakan untuk menyemprot manual secara langsung tanpa aplikasi.
- Power Supply/Adaptor 5V : Memberikan daya ke sistem.
- Breadboard: Untuk menyambung kabel jumper.
- Kabel Jumper Male to Female, Female to Female : Sebagai penghantar atau konduktor listrik untuk menghubungkan dua titik atau komponen dalam suatu rangkaian
- Mini L298N Motor Driver Dual H Bridge MX11508 : Berfungsi untuk mengontrol dua motor DC atau satu motor stepper.
- Sensor PIR : Mendeteksi pergerakan, jika terdeteksi ada pergerakan mode PIR akan melakukan penyemprotan.
- LCD 16x2 : Menampilkan informasi seperti,
- Buzzer : Sebagai penanda bahwa perangkat akan melakukan penyemprotan.
- LED : Sebagai penanda bahwa perangkat telah terhubung oleh WiFi.

4.3 Perangkat Lunak Yang Digunakan

- Platform IO : Digunakan untuk menulis dan mengunggah kode program ke ESP32.
- Blynk : Aplikasi berbasis Android/iOS yang memungkinkan pengguna untuk mengontrol dan memantau perangkat secara real-time.
- Library : Seperti WiFiClient.h, WiFi.h, BlynkSimpleEsp32.h, TimeLib.h, WidgetRTC.h, HTTPClient.h, Wire.h, dan LiquidCrystal_I2C.h untuk mendukung komunikasi, kendali, display dan fungsi sensor

4.4 Wiring Pada Perangkat



LCD:

- GND ke breadboard - (minus)
- VCC ke breadboard + (plus)
- SDK ke pin D21 esp32
- SCL ke pin D22 esp32

Sensor PIR:

- pin VCC ke breadboard + (plus)
- pin OUT ke pin D14 esp32
- pin GND ke pin breadboard - (minus)

Buzzer:

- pin + (plus) ke pin D5 esp32
- pin - (minus) ke pin breadboard - (minus)

Pushbutton dua pin:

- pin + (plus) ke pin D4 esp32
- -pin - (minus) ke pin breadboard - (minus)

Mini L298N Motor Driver Dual H Bridge MX1508:

- pin + (plus) ke pin 3V3 esp32
- pin - (minus) ke pin GND esp32
- pin int1&2 (digabung) ke pin D23 esp32
- pin int3&4 (digabung) ke pin D19 esp32
- pin motor-A ke + (plus) motor dc 5V
- pin motor-B ke - (minus) motor dc 5V

4.5 Kode Program PlatformIO

4.5.1 main.cpp :

```
#define BLYNK_TEMPLATE_ID "TMPL6yypVJXVs"
#define BLYNK_TEMPLATE_NAME "SmartSpray"
#define BLYNK_AUTH_TOKEN "D6b_dLWhhMktuDoAl9Vm5tA2NZZN1BGZ"
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <TimeLib.h>
#include <WidgetRTC.h>
#include <HTTPClient.h>
#include <Wire.h>
#include <EEPROM.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

String botToken = "7877132934:AAGYGir2xtz7ACkkZg6jh_ZS40DfviuEjkA"; // Token dari BotFather
String chatID = "8070797176";

char auth[] = "D6b_dLWhhMktuDoAl9Vm5tA2NZZN1BGZ";
char ssid[] = "Fnetwork";
char pass[] = "passwordnyalupa";

// Pin Konfigurasi
#define EEPROM_SIZE 32
#define MOTOR_FORWARD_PIN 23
#define MOTOR_REVERSE_PIN 19
#define BUTTON_PIN 4
#define BUZZER_PIN 5
#define WIFI_LED_PIN 27
#define PIR_PIN 14
```

```
#define SPRAY_DURATION 300 // ms

#define VPIN_SPRAY_BUTTON V0
#define VPIN_SPRAY_STATUS V1
#define VPIN_TIMER1 V2
#define VPIN_TIMER2 V3
#define VPIN_TIMER3 V4
#define VPIN_PIR_MODE V5

BlynkTimer timer;
WidgetRTC rtc;

unsigned long lcdTimeout = 0;
bool lcdNeedsReset = false;
bool countdownDisplayed[3] = {false, false, false};

bool pirModeEnabled = false;
bool motionDetected = false;
unsigned long lastMotionTime = 0;

bool lastButtonState = HIGH;
unsigned long lastDebounceTime = 0;
const unsigned long debounceDelay = 300; // ms
volatile bool buttonPressed = false;
unsigned long lastButtonPress = 0;

bool isSpraying = false;

bool triggeredByButton = false;
bool triggeredByTimer = false;

int scheduleHour[3] = {-1, -1, -1};
int scheduleMinute[3] = {-1, -1, -1};
bool alreadySprayed[3] = {false, false, false};

bool isOnline = false;

void updateLCD(String line1, String line2, bool autoReset = false)
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(line1);
  lcd.setCursor(0, 1);
  lcd.print(line2);
```

```
  if (autoReset)
  {
    lcdTimeout = millis();
    lcdNeedsReset = true;
  }
}

void tampilkanDefaultLCD()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("SmartSpray Ready");

  // Misalnya tampilkan mode di baris ke-2:
  lcd.setCursor(0, 1);
  if (!isOnline)
    lcd.print("Mode: Offline");
  else if (pirModeEnabled)
    lcd.print("Mode: PIR Aktif");
  else
    lcd.print("Mode: Manual");
}

void sendTelegramMessage(String message)
{
  if (!isOnline)
    return;
  if (WiFi.status() == WL_CONNECTED)
  {
    HTTPClient http;
    String url = "https://api.telegram.org/bot" + botToken + "/sendMessage?chat_id=" +
chatID + "&text=" + message;

    http.begin(url);
    int httpCode = http.GET();
    http.end();
    Serial.println(httpCode > 0 ? "Pesan Telegram terkirim." : "Gagal kirim Telegram.");
  }
}

void tampilkanCountdownLCD(int index)
{
  char msg[17];
```

```
 sprintf(msg, "Akan Semprot %02d:%02d", scheduleHour[index], scheduleMinute[index]);
 updateLCD("SmartSpray", msg);
}

void buzzerBeepTwice()
{
 for (int i = 0; i < 2; i++)
 {
  digitalWrite(BUZZER_PIN, HIGH); // bunyi
  delay(200);
  digitalWrite(BUZZER_PIN, LOW); // Bunyi OFF
  delay(200);                    // Jeda antar beep (200ms)
 }
}

void simpanJadwalKeEEPROM()
{
 for (int i = 0; i < 3; i++)
 {
  EEPROM.write(i * 2, scheduleHour[i]);
  EEPROM.write(i * 2 + 1, scheduleMinute[i]);
 }
 EEPROM.commit();
 Serial.println("✅ Jadwal disimpan ke EEPROM");
}

void muatJadwalDariEEPROM()
{
 for (int i = 0; i < 3; i++)
 {
  scheduleHour[i] = EEPROM.read(i * 2);
  scheduleMinute[i] = EEPROM.read(i * 2 + 1);

  // Validasi
  if (scheduleHour[i] > 23 || scheduleMinute[i] > 59)
  {
   scheduleHour[i] = -1;
   scheduleMinute[i] = -1;
  }
 }
 Serial.println("📥 Jadwal dimuat dari EEPROM");
}
```

```
void spray()
{
  if (isSpraying)
    return;

  isSpraying = true;

  if (triggeredByButton)
  {
    if (isOnline)
      sendTelegramMessage("Penyemprotan dilakukan secara MANUAL melalui tombol.");
    updateLCD("Manual Button", "Spraying...");
    triggeredByButton = false;
  }

  else if (triggeredByTimer)
  {
    if (isOnline)
      sendTelegramMessage("Penyemprotan dilakukan sesuai JADWAL TIMER.");
    updateLCD("Timer Spray", "Spraying...");
    triggeredByTimer = false;
  }
  else if (pirModeEnabled)
  {
    updateLCD("PIR Sensor", "Spraying...");
  }

  else
  {
    updateLCD("Blynk Manual", "Spraying...");
  }

  buzzerBeepTwice();

  Serial.println("Melakukan Spraying...");
  if (isOnline)
  {
    Blynk.virtualWrite(VPIN_SPRAY_STATUS, "Spraying...");
    Blynk.logEvent("spray_event", "Penyemprotan dilakukan!");
  }

  // digitalWrite(LED_PIN, HIGH);
```

```
  digitalWrite(MOTOR_FORWARD_PIN, HIGH);

  delay(SPRAY_DURATION);

  digitalWrite(MOTOR_FORWARD_PIN, LOW);
  digitalWrite(BUZZER_PIN, LOW);
  delay(100);

  Serial.println("Spray done.");
  if (isOnline)
    Blynk.virtualWrite(VPIN_SPRAY_STATUS, "Selesai Menyemprot");
  delay(1500);
  tampilkanDefaultLCD();
  isSpraying = false;

  delay(1500); // tampilkan status selama 1,5 detik

  tampilkanDefaultLCD();
  isSpraying = false;
}

void checkButton()
{
  static bool lastState = HIGH;
  bool currentState = digitalRead(BUTTON_PIN);

  if (lastButtonState == HIGH && currentState == LOW)
  {
    unsigned long currentTime = millis();
    if (currentTime - lastDebounceTime > debounceDelay)
    {
      Serial.println("Tombol fisik ditekan!");
      triggeredByButton = true;
      spray();
      lastDebounceTime = currentTime;
    }
  }

  lastButtonState = currentState;
}

void IRAM_ATTR handleButtonInterrupt()
{
```

```
  static unsigned long lastInterruptTime = 0;
  unsigned long currentTime = millis();

  // Debounce: abaikan jika <300ms dari tombol sebelumnya
  if (currentTime - lastInterruptTime > debounceDelay)
  {
    buttonPressed = true;
    lastButtonPress = currentTime;
    lastInterruptTime = currentTime;
  }
}

BLYNK_WRITE(VPIN_SPRAY_BUTTON)
{
  if (!isOnline)
    return;
  int value = param.asInt();
  if (value == 1)
  {
    if (!pirModeEnabled)
    {
      spray();
    }
    else
    {
      Serial.println("PIR mode aktif - penyemprotan manual Blynk diabaikan");
    }
    Blynk.virtualWrite(VPIN_SPRAY_BUTTON, 0);
  }
}

BLYNK_WRITE(VPIN_TIMER1)
{
  if (!isOnline)
    return;
  TimeInputParam t(param);
  if (t.hasStartTime())
  {
    scheduleHour[0] = t.getStartHour();
    scheduleMinute[0] = t.getStartMinute();
    alreadySprayed[0] = false;
    simpanJadwalKeEEPROM();
    Serial.printf("Timer 1 diset: %02d:%02d\n", scheduleHour[0], scheduleMinute[0]);
```

```
    char schedule[17];
    sprintf(schedule, "Timer: %02d:%02d", scheduleHour[0], scheduleMinute[0]);
    updateLCD(schedule, "", true);
  }
}

BLYNK_WRITE(VPIN_TIMER2)
{
  if (!isOnline)
    return;
  TimeInputParam t(param);
  if (t.hasStartTime())
  {
    scheduleHour[1] = t.getStartHour();
    scheduleMinute[1] = t.getStartMinute();
    alreadySprayed[1] = false;
    simpanJadwalKeEEPROM();
    Serial.printf("Timer 2 diset: %02d:%02d\n", scheduleHour[1], scheduleMinute[1]);

    char schedule[17];
    sprintf(schedule, "Timer: %02d:%02d", scheduleHour[1], scheduleMinute[1]);
    updateLCD(schedule, "", true);
  }
}

BLYNK_WRITE(VPIN_TIMER3)
{
  if (!isOnline)
    return;
  TimeInputParam t(param);
  if (t.hasStartTime())
  {
    scheduleHour[2] = t.getStartHour();
    scheduleMinute[2] = t.getStartMinute();
    alreadySprayed[2] = false;
    simpanJadwalKeEEPROM();
    Serial.printf("Timer 3 diset: %02d:%02d\n", scheduleHour[2], scheduleMinute[2]);

    char schedule[17];
    sprintf(schedule, "Timer: %02d:%02d", scheduleHour[2], scheduleMinute[2]);
    updateLCD(schedule, "", true);
  }
}
```

```
BLYNK_WRITE(VPIN_PIR_MODE)
{
  if (isOnline)
    pirModeEnabled = param.asInt();
  if (pirModeEnabled)
    Serial.println("Mode PIR AKTIF — hanya semprot jika sensor aktif");
  else
    Serial.println("Mode PIR NONAKTIF — semprot bisa manual dan timer");
}

void checkSchedules()
{
  int nowHour = hour();
  int nowMinute = minute();
  int nowSecond = second();
  Serial.printf("RTC sekarang: %02d:%02d (%d)\n", nowHour, nowMinute, year());

  for (int i = 0; i < 3; i++)
  {
    if (scheduleHour[i] == -1)
      continue;

    // Jika 30 detik sebelum waktu semprot
    if (nowHour == scheduleHour[i] && nowMinute == scheduleMinute[i] &&
nowSecond >= 30 && !countdownDisplayed[i])
      Serial.printf("▶ Countdown LCD Tampil: Jadwal %d %02d:%02d\n", i, nowHour,
nowMinute);

    {
      tampilkanCountdownLCD(i); // Tampilkan ke LCD
      lcdTimeout = millis();    // Mulai timer 5 detik
      lcdNeedsReset = true;
      countdownDisplayed[i] = true;
      char msg[17];
      sprintf(msg, "Timer: %02d:%02d", nowHour, nowMinute, nowSecond);
      Serial.println(msg);
    }

    if (nowHour == scheduleHour[i] && nowMinute == scheduleMinute[i] &&
nowSecond == 0)
    {
      if (!alreadySprayed[i])
      {
        Serial.printf("JADWAL %d COCOK — menyemprot!\n", i);
```

```
        triggeredByTimer = true;

        if (!pirModeEnabled)
        {
          spray();
        }
        else
        {
          Serial.println("PIR mode aktif - penyemprotan via timer diabaikan");
        }
        alreadySprayed[i] = true;
        countdownDisplayed[i] = false; // Reset setelah menyemprot
      }
    }

    if (nowMinute != scheduleMinute[i] || nowHour != scheduleHour[i])
    {
      countdownDisplayed[i] = false;
      alreadySprayed[i] = false;
    }
  }

  Serial.printf("Sekarang %02d:%02d\n", nowHour, nowMinute, nowSecond);
}

void setup()
{
  Serial.begin(115200);
  Serial.println("Scanning I2C devices...");

  EEPROM.begin(EEPROM_SIZE);
  muatJadwalDariEEPROM();

  for (byte address = 1; address < 127; address++)
  {
    Wire.beginTransmission(address);
    if (Wire.endTransmission() == 0)
    {
      Serial.print("I2C device found at address 0x");
      Serial.println(address, HEX);
    }
  }
  Serial.println("Scan done.");
```

```
  lcd.init();      // Inisialisasi LCD
  delay(100);      // Tambahan delay agar LCD siap
  lcd.backlight(); // Nyalakan backlight
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("SmartSpray Ready");
  lcd.setCursor(0, 1);
  lcd.print("Menunggu input...");
  Serial.println("ESP32 Ready. Tekan tombol untuk menyemprot.");

  Wire.begin();

  pinMode(WIFI_LED_PIN, OUTPUT);
  digitalWrite(WIFI_LED_PIN, LOW);

  pinMode(PIR_PIN, INPUT);

  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);

  pinMode(MOTOR_FORWARD_PIN, OUTPUT);
  pinMode(MOTOR_REVERSE_PIN, OUTPUT);
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  //      attachInterrupt(digitalPinToInterrupt(BUTTON_PIN),    handleButtonInterrupt,
FALLING);

  digitalWrite(MOTOR_FORWARD_PIN, LOW);
  digitalWrite(MOTOR_REVERSE_PIN, LOW);

  WiFi.begin(ssid, pass);
  Serial.println("Menyambungkan WiFi...");
  updateLCD("WiFi", "Menyambung...");

  const unsigned long wifiTimeout = 15000;
  unsigned long startAttempt = millis();

  while (WiFi.status() != WL_CONNECTED && millis() - startAttempt <
wifiTimeout)
  {
    delay(200);
    Serial.print(".");
  }
```

```
 if (WiFi.status() == WL_CONNECTED)
 {
  Serial.println("\n✅ WiFi Tersambung");
  digitalWrite(WIFI_LED_PIN, HIGH);
  isOnline = true;
  updateLCD("WiFi", "Tersambung");
  delay(5000);
  tampilkanDefaultLCD();
 }
 else
 {
  Serial.println("\n❌ Gagal WiFi");
  isOnline = false;
  digitalWrite(WIFI_LED_PIN, LOW);
  updateLCD("WiFi", "Gagal");
 }

 if (isOnline)
 {
  Blynk.config(auth, "blynk.cloud", 80);
  Blynk.connect(15000); // timeout 3 detik
  if (Blynk.connected())
  {
   Serial.println("✅ Blynk Terhubung");
   sendTelegramMessage("ESP32 Terhubung!");
  }
  else
  {
   Serial.println("❌ Gagal konek Blynk");
   isOnline = false;
  }
 }

 // timer.setInterval(100, checkButton);
 rtc.begin();
 time_t t = now();
 Serial.printf("Waktu awal RTC: %02d:%02d:%02d %02d/%02d/%04d\n", hour(t),
minute(t), second(t), day(t), month(t), year(t));
 setSyncInterval(60); // update waktu tiap 60 detik

 if (year() < 2025)
 {
  Serial.println("RTC belum tersinkronisasi!");
```

```
    return;
  }
  timer.setInterval(250, checkSchedules);
  Serial.println("✅ Sistem siap. Gunakan tombol fisik atau aplikasi.");

  char scheduleMsg[17];
  sprintf(scheduleMsg, "Timer: %02d:%02d", scheduleHour[0], scheduleMinute[0]);
  updateLCD("SmartSpray", scheduleMsg);

  delay(1500);          // tampilkan info jadwal sejenak
  tampilkanDefaultLCD(); // kembali ke tampilan default
}

void loop()
{
  checkButton();

  if (buttonPressed)
  {
    buttonPressed = false; // Reset
    Serial.println("Tombol ditekan!");

    if (!pirModeEnabled && !isSpraying)
    {
      triggeredByButton = true;
      spray();
    }
  }

  if (isOnline && pirModeEnabled && digitalRead(PIR_PIN) == HIGH && !isSpraying)
  {
    unsigned long now = millis();
    if (now - lastMotionTime > 10000) // Hindari penyemprotan berulang tiap gerakan
    {
      Serial.println("Gerakan terdeteksi oleh PIR!");
      triggeredByButton = false;
      triggeredByTimer = false;
      spray();
      sendTelegramMessage("Penyemprotan dilakukan karena GERAKAN terdeteksi oleh sensor PIR.");
      lastMotionTime = now;
    }
```

```
  }

  if (lcdNeedsReset && millis() - lcdTimeout >= 5000)
  {
    tampilkanDefaultLCD();
    lcdNeedsReset = false;
    Serial.println("🔁 LCD kembali ke tampilan default.");
  }

  if (isOnline)
  {
    Blynk.run();
  }
  timer.run();

  static int lastCheckedMinute = -1;
  time_t t = now();
  int currentHour = hour(t);
  int currentMinute = minute(t);

  if (currentMinute != lastCheckedMinute)
  {
    for (int i = 0; i < 3; i++)
    {
      if (scheduleHour[i] == currentHour && scheduleMinute[i] == currentMinute)
      {
        Serial.printf("Timer %d cocok! Menyemprot sekarang!\n", i + 1);
        triggeredByTimer = true;
        spray();
      }
    }
    lastCheckedMinute = currentMinute;
  }

  if (WiFi.status() == WL_CONNECTED)
  {
    digitalWrite(WIFI_LED_PIN, HIGH);
  }
  else
  {
    digitalWrite(WIFI_LED_PIN, LOW);
  }
}
```

```
// end
```

4.5.2 platformio.ini :

```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

[env:esp32dev]
platform = espressif32
board = esp32dev
framework = arduino
monitor_speed = 115200
upload_port = COM3
monitor_filters = direct
monitor_rts = 0
monitor_dtr = 0
lib_deps =
        blynkkk/Blynk@^1.3.2
        tzapu/WiFiManager@^2.0.17
        paulstoffregen/Time@^1.6.1
        adafruit/Adafruit Unified Sensor@^1.1.15
        adafruit/DHT sensor library@^1.4.6
marcoschwartz/LiquidCrystal_I2C@^1.1.4
```

Link Project :

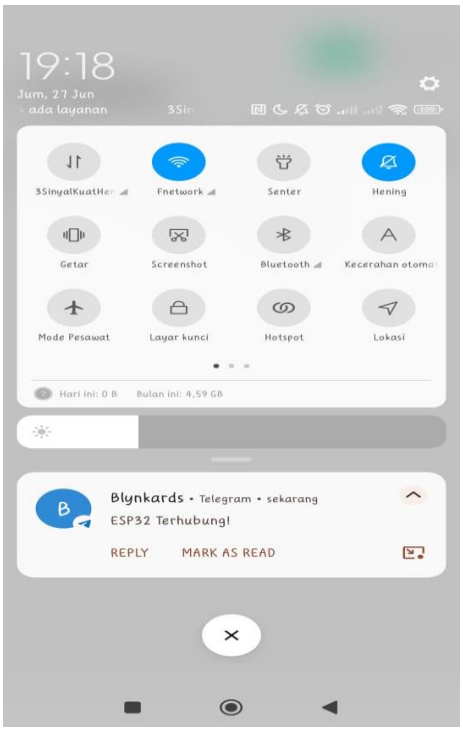
https://github.com/piknyeng/Smart-Air-Freshener

4.4 Pengujian Sistem

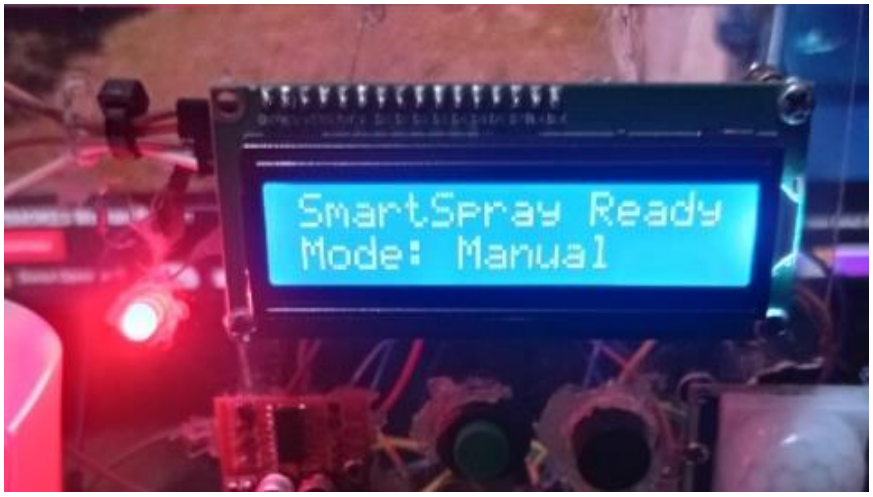
Pengujian dilakukan dalam beberapa skenario:

- Pengujian penyemprotan manual atau dengan aplikasi: Memastikan tombol fisik maupun tombol di aplikasi Blynk berfungsi dengan baik.
- Pengujian penyemprotan dengan jadwal : Memastikan penyemprotan dengan penjadwalan berfungsi dengan baik.
- Pengujian Indikator : Memastikan indikator seperti LED dan Buzzer berfungsi dengan baik.
- Pengujian penampilan informasi melalui LCD 16x2 : Memastikan LCD menampilkan informasi sesuai dengan apa yang telah diprogramkan dengan baik.
- Pengujian notifikasi ketika perangkat melakukan penyemprotan : Perangkat mengirim notifikasi saat melakukan penyemprotan.
- Stabilitas koneksi dengan Blynk : Sistem tetap terhubung selama koneksi Wi-Fi stabil.
- Pengujian mode manual dan mode PIR : Memastikan bahwa mode berfungsi dengan baik tidak ada kendala (asalkan koneksi WiFi stabil).

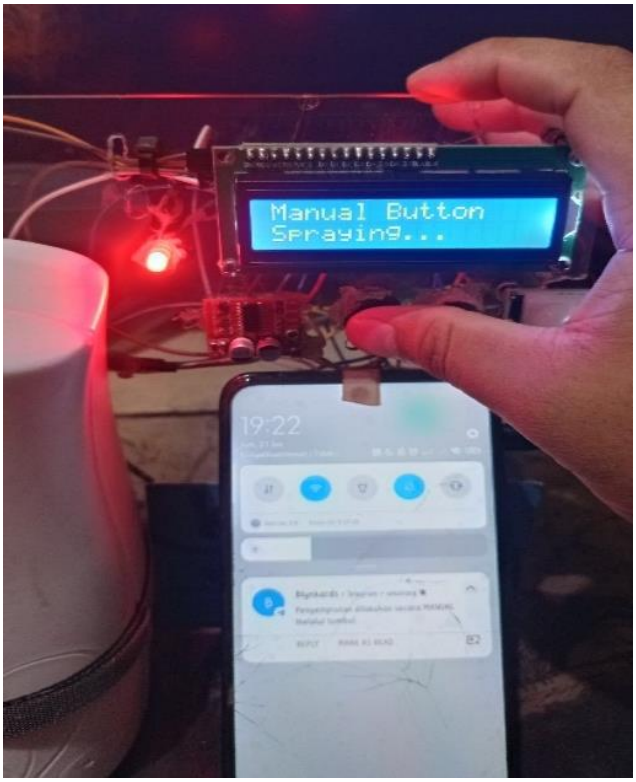
4.5 Hasil Pengujian



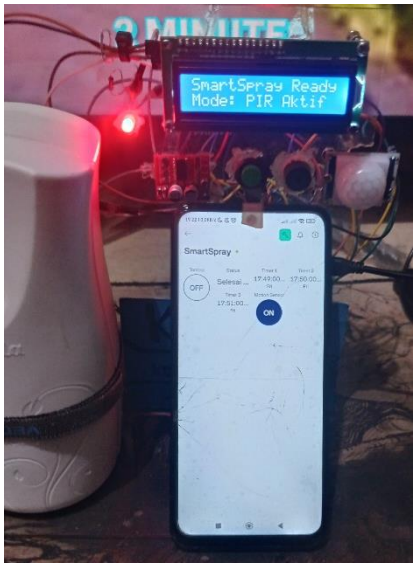
Jika perangkat tersambung lcd akan menampilkan “WiFi Tersambung” dan mengirim notifikasi ke smartphone melalui Bot Telegram (sebenarnya melalui aplikasi blynk bisa mengirim notifikasi, namun harus member PRO).

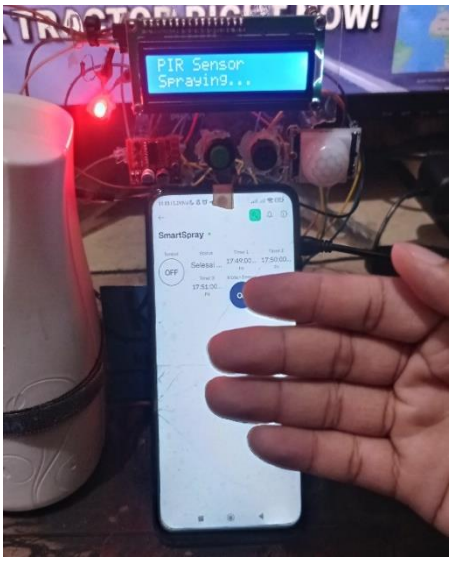


Setelah lcd menampilkan informasi “WiFi Tersambung” lcd menampilkan tampilan default.

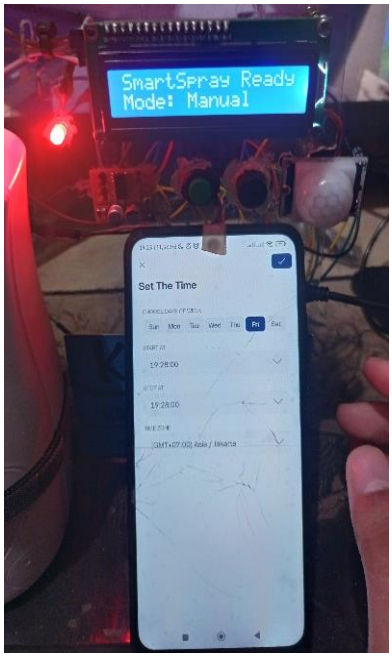


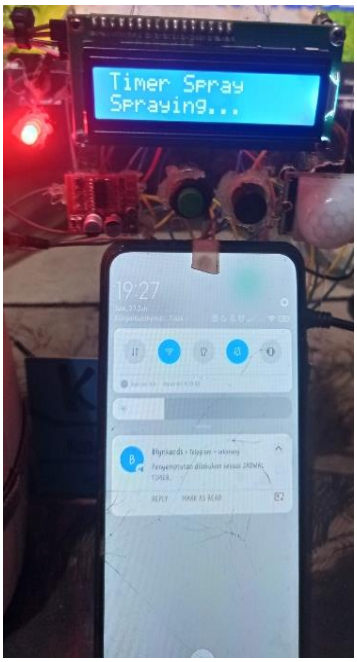
Ketika user memencet tombol manual fisik dan tombol manual pada aplikasi untuk menyemprot lcd menampilkan informasi “Blynk Manual Spraying…” sebagai indikator bahwa perangkat akan menyemprot.





Ketika user mengaktifkan mode PIR pada aplikasi lcd akan menampilkan “SmartSpray Ready Mode: PIR Aktif”, lalu perangkat akan menyemprot ketika sensor PIR mendeteksi adanya gerakan, lalu perangkat akan mengirim notifikasi ke smartphone melalui Bot telegram.





User dapat mengatur jadwal/timer, setelah user mengatur jadwal/timer lcd akan menampilkan waktu yang telah diatur lalu kembali ke tampilan default setelah 5 detik, dan lcd akan menampilkan informasi “Timer Spray Spraying…” dan perangkat akan mengirim notifikasi ke smartphone melalui Bot telegram.

Berikut adalah link Project Smart Air Freshener, tutorial setup aplikasi Blynk agar smartphone dapat mengontrol perangkat Smart Air Freshener, dan video hasil pengujian perangkat.

Project Smart Air Freshener :
https://github.com/piknyeng/Smart-Air-Freshener

Tutorial Setup Blynk :
https://youtu.be/E_pQ1cMHW9M?si=30_eru-77Ga6vp2C

Hasil Pengujian :
https://youtu.be/SJ0pVnbAHfc?si=prsmf9JQ05-fzHgO

4.6 Pembahasan

Berdasarkan hasil implementasi dan pengujian, sistem Smart Air Freshener bekerja sesuai dengan keinginan yaitu perangkat dapat berfungsi dalam keadaan offline dan online, dapat mengirim notifikasi ke smartphone. Alat mampu melakukan penyemprotan secara otomatis berdasarkan jadwal, dengan sensor PIR, dan tombol manual/tombol dari aplikasi, fitur yang ada memberikan kemudahan pengguna mengontrol perangkat melalui aplikasi Blynk, dan mengirimkan notifikasi saat perangkat melakukan penyemprotan melalui sensor PIR, tombol fisik, dan penjadwalan.

Namun, terdapat beberapa kendala seperti:

- Perangkat membutuhkan koneksi Wi-Fi yang stabil jika pengguna ingin mengontrol jarak jauh.
- Penjadwalan masih harus melalui aplikasi, jika perangkat dalam keadaan offline/tidak terhubung WiFi pengguna tidak bisa mengatur jadwal semprot, dan jadwal tidak tersimpan jika perangkat tidak terhubung oleh WiFi.
- Penyemprotan belum dapat diatur volume atau durasinya secara fleksibel.
- Perangkat tidak memiliki sensor suhu DHT22 untuk monitoring suhu dan kelembapan ruangan.
- Perangkat tidak memiliki sensor level cairan.
- Perangkat tidak memiliki refill auto reminder untuk mengingatkan pengguna jika cairan pengharum sudah habis.
- Kendala ini menjadi bahan evaluasi untuk pengembangan sistem lebih lanjut agar lebih akurat dan fleksibel dalam penggunaannya.

BAB V
PENUTUP

5.1 Kesimpulan

Proyek Smart Air Freshener berbasis IoT berhasil diimplementasikan menggunakan mikrokontroler ESP32 yang terhubung dengan sensor PIR, tombol fisik, buzzer, LED, serta motor penyemprot.

Sistem ini mampu melakukan penyemprotan secara manual melalui tombol fisik dan aplikasi Blynk, secara otomatis dengan penjadwalan dan sensor PIR. Selain itu, alat juga dapat mengirim notifikasi ke aplikasi saat perangkat melakukan penyemprotan sehingga memudahkan pengguna dalam pemantauan.

Dengan dukungan teknologi Internet of Things, alat ini mampu memberikan kontrol dan monitoring jarak jauh secara real-time, dan secara keseluruhan telah berjalan sesuai tujuan sebagai alat penyemprot ruangan yang efisien, fleksibel, dan modern.

5.2 Saran

Untuk pengembangan lebih lanjut, sistem Smart Air Freshener masih memiliki potensi peningkatan dari segi akurasi, fungsionalitas, dan kenyamanan pengguna.

Beberapa saran pengembangan menambah sensor DHT22 untuk memantau suhu dan kelembapan ruangan, penambahan fitur volume semprot agar cairan tidak cepat habis, notifikasi ketika cairan sudah habis.

Selain itu, antarmuka aplikasi Blynk juga disarankan untuk ditingkatkan agar lebih interaktif, misalnya dengan menampilkan grafik dan log aktivitas.

Pengendalian alat juga dapat diperluas melalui fitur perintah suara atau integrasi dengan asisten digital seperti Google Assistant atau Alexa guna meningkatkan kemudahan penggunaan.

DAFTAR PUSTAKA

https://github.com/piknyeng/Smart-Air-Freshener

https://youtu.be/E_pQ1cMHW9M?si=30_eru-77Ga6vp2C

https://youtu.be/SJ0pVnbAHfc?si=prsmf9JQ05-fzHgO

IoT basics: A guide for beginners. (n.d). techtarget.com.

https://www.techtarget.com/whatis/feature/IoT-basics-A-guide-for-beginners

IoT Air Freshener (with NodeMCU, Arduino, IFTTT and Adafruit.io). (n.d). instructables.com.

https://www.instructables.com/IoT-Air-Freshner-with-NodeMCU-Arduino-IFTTT-and-Ad/

IoT Air Freshener. (n.d). hackaday.io.
https://hackaday.io/project/28284-iot-air-freshener

IOFresh. (n.d). github.com/SupreethRao99.
https://github.com/SupreethRao99/IOFresh

SMART AFRESH IO: Integrated Smart Air Freshener Dispenser. (n.d). researchgate.net.
https://www.researchgate.net/publication/388463983_SMART_AFRESH_IO_Integrated_Smart_Air_Freshener_Dispenser

Pengharum udara. (20 Maret 2024). wikipedia.org.

https://id.wikipedia.org/wiki/Pengharum_udara

apa itu esp32 salah satu modul wi-fi popular. (n.d). anakteknik.co.id.

https://www.anakteknik.co.id/krysnayudhamaulana/articles/apa-itu-esp32-salah-satu-modul-wi-fi-poppuler

PENGENALAN ESP32 BOARD. (n.d). pens.ac.id.

https://zenhadi.lecturer.pens.ac.id/kuliah/InternetOfThings/Praktek/Praktek%20Modul%201%20Pengenalan%20ESP32.pdf

ESP32. (n.d). binus.ac.id.

https://student-activity.binus.ac.id/himtek/2022/07/27/esp32/

Apa Itu Mikrokontroler : Jenis, Komponen dan Cara Kerjanya. (n.d). kelasplc.com.

https://www.kelasplc.com/mikrokontroler-adalah/

Mengenal Berbagai Jenis Microcontroller dan Kelebihannya. (n.d). uma.ac.id.

https://baraka.uma.ac.id/mengenal-berbagai-jenis-microcontroller-dan-kelebihannya/

Nini Firmawati. (2024, Juni 3). *Menyimak Sejarah, Perkembangan, dan Kemajuan Mikrokontroler*. kumparan.com.

https://kumparan.com/firmawatinini/menyimak-sejarah-perkembangan-dan-kemajuan-mikrokontroler-22raZYDqrUl/full

Mengenal Sejarah dan Perkembangan Microcontroller dari Masa ke Masa. (n.d). uma.ac.id.

https://baraka.uma.ac.id/mengenal-sejarah-dan-perkembangan-microcontroller-dari-masa-ke-masa/

Mikrokontroler pengertian fungsi dan jenis-jenisnya. (n.d). itbmg.ac.id.

https://mediacenter.itbmg.ac.id/mikrokontroler-pengertian-fungsi-dan-jenis-jenisnya/

Internet of Things (IoT): Pengertian, Cara Kerja dan Contohnya. (n.d). cloudcomputing.id.

https://www.cloudcomputing.id/pengetahuan-dasar/iot-pengertian-contohnya

Mengenal Blynk Platform IoT, Instalasi dan Penerapannya. (n.d). anakkendali.com.

https://www.anakkendali.com/mengenal-blynk-platform-iot-instalasi-dan-penerapannya/

Arduino dan Blynk: Membuat Aplikasi IoT Tanpa Coding Berlebih. (n.d). arduinoindonesia.id.

https://www.arduinoindonesia.id/2024/04/arduino-dan-blynk-membuat-aplikasi-iot-tanpa-coding-berlebih.html

Hary Tri. (2023, November 9). *Menggunakan Aplikasi BLYNK untuk fungsi IOT.* jejakmedia.link.

https://blog.jejakmedia.link/menggunakan-aplikasi-blynk-untuk-fungsi-iot/

Berkenalan Singkat dengan IoT. (n.d). uii.ac.id.

https://informatics.uii.ac.id/2023/02/13/berkenalan-singkat-dengan-iot/