PERANCANGAN ANTARMUKA PENGGUNA UNTUK *MONITORING* SENSOR PADA SISTEM PERINGATAN DINI KEBAKARAN HUTAN BERBASIS *WEBSITE*

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI REKAYASA KOMPUTER

Ditujukan untuk memenuhi persyaratan

memperoleh gelar Sarjana Teknik

MERRY TRI NIA

NIM. 165060301111072

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2021

LEMBAR PENGESAHAN

PERANCANGAN ANTARMUKA PENGGUNA UNTUK *MONITORING* SENSOR PADA SISTEM PERINGATAN DINI KEBAKARAN HUTAN BERBASIS *WEBSITE*

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI REKAYASA KOMPUTER

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

MERRY TRI NIA

NIM. 165060301111072

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing
pada tanggal 21 Agustus 2021

Dosen Pembimbing I	Dosen Pembimbing II
Rudy Yuwono, S.T., M.Sc.	Ir. Ali Mustofa, S.T., M.T., IPM.
NIP. 19710615 199802 1 003	NIP. 19710601 200003 1 001

Mengetahui,
Plt. Ketua Jurusan Teknik Elektro

Muhammad Aziz Muslim, ST., MT., Ph.D.
NIP. 19741203 200012 1 001

JUDUL SKRIPSI:

PERANCANGAN ANTARMUKA PENGGUNA UNTUK *MONITORING* SENSOR PADA SISTEM PERINGATAN DINI KEBAKARAN HUTAN BERBASIS *WEBSITE*

Nama Mahasiswa	: Merry Tri Nia
NIM	: 165060301111072
Program Studi	: Teknik Elektro
Konsentrasi	: Rekayasa Komputer

Tim Dosen Pembimbing:

Dosen Pembimbing 1 : Rudy Yuwono, S.T., M.Sc. .....

Dosen Pembimbing 2 : Ir. Ali Mustofa, S.T., M.T., IPM. .....

Tim Dosen Penguji:

Dosen Penguji 1 : Adharul Muttaqin, S.T., M.T. disetujui tanggal 21 Agustus 2021

Dosen Penguji 2 : Dr. Ir. Muhammad Aswin, M.T. disetujui tanggal 21 Agustus 2021

Tanggal Ujian : 23 Juli 2021

SK Penguji : Nomer 1257 Tahun 2021

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Skripsi ini adalah asli dari pemikiran saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam Naska Skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia Skripsi dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, pasal 25 ayat 2 dan pasal 70).

Malang, Juli 2021

Mahasiswa,

MERRY TRI NIA

NIM. 165060301111072

Teriring Ucapan Terimakasih kepada:
Ayah, Mama, Ibuk, dan Bapak tersayang

RINGKASAN

Merry Tri Nia, Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya, Juni 2021 *Perancangan Antarmuka Pengguna untuk Monitoring Sensor pada Sistem Peringatan Dini Kebakaran Hutan*, Dosen Pembimbing: Rudy Yuwono, ST., M.Sc. dan Ir. Ali Mustofa, S.T., M.T., IPM.

Kebakaran pada hutan dan lahan merupakan suatu keadaan di mana hutan atau lahan dilanda api. Akibat dari kebakaran hutan yang meluas akan menyebabkan kerugian lingkungan dan kerugian ekonomi. Oleh karena itu untuk mengurangi kerugian yang disebabkan kebakaran diperlukan penanganan yang cepat untuk menghindari meluasnya kebakaran. Namun, permasalahan dalam penanganan kebakaran hutan adalah lambatnya pemberitahuan mengenai informasi terjadinya kebakaran. Oleh karena itu dibuatlah sebuah Sistem Peringatan Dini Kebakaran Hutan untuk mempercepat proses tanggap bencana. Sistem Peringatan Dini Kebakaran Hutan Menggunakan yang telah dibuat terdiri dari dua subsistem yaitu subsistem pendeteksi yang akan dipasang di hutan dan subsistem *main station* yang akan dipasang di pos pengawas yang memiliki akses *Wi-Fi*. Sistem bekerja dengan mengirimkan data kondisi lingkungan yang didapat dari subsistem pendeteksi kemudian data dikirim ke subsistem *main station* melalui jaringan *LoRa* 923 MHz. Data yang telah diterima di subsistem *main station* kemudian dikirim ke basis data dengan memanfaatkan jaringan *Wi-Fi* untuk kemudian data ditampilkan pada *user interface* yang berbasis *website*.

Pada skripsi ini akan dibahas tentang perancangan antarmuka pengguna untuk *monitoring* sensor berbasis *website* dengan harapan sistem antarmuka pengguna dapat menampilkan data dari sistem peringatan dini kebakaran hutan secara *real-time*, sehingga pengguna dapat dengan mudah memahami informasi yang ingin disampaikan oleh sistem dan proses tanggap bencana kebakaran dapat segera dilaksanakan. Untuk menilai apakah perancangan antarmuka pengguna dapat diimplementasikan dan dapat berjalan sesuai kebutuhan maka akan dilakukan pengujian fungsionalitas. Selain itu akan dilakukan analisis kompleksitas algoritma untuk mengukur kompleksitas waktu dari algoritma yang dibuat.

Pengujian fungsionalitas dilakukan dengan simulasi pengiriman data secara berkala ke basis data dengan memanfaatkan ESP32, kemudian mengamati *website* yang telah dibuat untuk melihat apakah data yang dikirim dapat terbaharui sesuai dengan data terbaru. Hasil pengujian fungsionalitas menunjukkan bahwa *website* telah dapat menampilkan seluruh informasi yang diinginkan dan dapat terbaharui sesuai data terbaru dengan rata-rata perbedaan waktu antara data terkirim dengan data ditampilkan yaitu 0,9 s. Sedangkan hasil analisis kompleksitas waktu algoritma adalah sebagai berikut, untuk algoritma menampilkan nilai dari basis data didapatkan $T(n) = n + 7$, untuk algoritma membuat data *JSON* didapatkan $T(n) = 9$, untuk algoritma menampilkan peta didapatkan $T(n) = 6$, dan untuk algoritma menentukan kategori kondisi didapatkan $T(n) = 12$.

Kata Kunci: antarmuka pengguna, peringatan kebakaran, *website*

SUMMARY

Merry Tri Nia, *Department of Electrical Engineering, Faculty of Engineering, Brawijaya University, June 2021 Web-based User Interface Design for Sensor Monitoring on Forest Fire Early Warning System, Academic Supervisor:* Rudy Yuwono, ST., M.Sc. *and* Ir. Ali Mustofa, S.T., M.T., IPM.

Fire in forest and land is a condition where the forest or land is stricken by fire. The consequences of widespread forest fires will cause environmental and economic losses. Therefore, to reduce losses caused by fires, it is necessary to quickly handle them to avoid the spread of fires. However, the problem in handling forest fires is the delay in notification of information about the occurrence of fires. Therefore, a Forest Fire Early Warning System was created to speed up the disaster response process. Forest Fire Early Warning System consists of two subsystems, namely the detection subsystem that will be installed in the forest and the main station subsystem that will be installed at the control post that has Wi-Fi access. The system works by sending data on environmental conditions obtained from the detection subsystem then the data is sent to the main station subsystem via the LoRa 923 MHz network. The data that has been received in the main station subsystem is then sent to the database using a Wi-Fi network and then the data is displayed on a website-based user interface.

In this final project, we will discuss the design of a user interface for website-based sensors monitoring with the hope that the user interface system can display data from the Forest Fire Early Warning System in real-time so that users can easily understand the information to be conveyed by the system and the fire disaster response process can be implemented immediately. To assess whether the user interface design can be implemented and can run as needed, functionality testing will be carried out. In addition, algorithm complexity analysis will be carried out to measure the time complexity of the algorithm created.

Functionality testing is carried out by simulating sending data periodically to the database by utilizing ESP32, then observing the website that has been created to see if the data sent can be updated according to the latest data. The results of the functionality test show that the website has been able to display all the desired information and can be updated according to the latest data with an average time difference between the data sent and the data displayed, which is 0.9 s. While the results of the time complexity analysis of the algorithm are as follows, for the algorithm to display the value from the database, it is obtained $T(n) = n + 7$, for the algorithm to create JSON data, it is obtained $T(n) = 9$, for the algorithm to display the map, it is obtained $T(n) = 6$, and for the algorithm to determine the condition category, we get $T(n) = 12$.

Keywords: fire warning, user interface, website

DAFTAR ISI

Halaman

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kebakaran pada hutan dan lahan merupakan suatu keadaan dimana hutan atau lahan dilanda api. Faktor yang dapat menyebabkan terjadinya kebakaran terbagi menjadi dua yaitu faktor alam dan faktor manusia. Faktor alam berarti kebakaran hutan disebabkan oleh peristiwa alam yang memicu terjadinya kebakaran seperti sambaran petir. Sedangkan faktor manusia disebabkan oleh aktivitas manusia yang memicu terjadinya kebakaran hutan, seperti membuang puntung rokok, tidak memadamkan api unggun, atau sengaja melakukan pembakaran lahan untuk keperluan tertentu. Akibat dari kebakaran hutan yang meluas akan mengakibatkan kerugian lingkungan dan kerugian ekonomi. Oleh karena itu untuk mengurangi kerugian yang disebabkan kebakaran diperlukan penanganan yang cepat untuk menghindari meluasnya kebakaran. Namun, permasalahan dalam penanganan kebakaran hutan adalah lambatnya pemberitahuan mengenai informasi terjadinya kebakaran. Umumnya penggunaan cara konvensional dalam menyampaikan informasi kebakaran yaitu dengan menelpon pihak pemadam kebakaran kemudian memberikan informasi lokasi tempat kebakaran, ini cenderung memakan waktu ditambah waktu yang dibutuhkan pihak pemadam kebakaran untuk menuju lokasi kejadian membuat waktu tanggap menjadi semakin lama.

Berdasarkan Data Kementerian Lingkungan Hidup dan Kehutanan RI, pada tahun 2020 luas kebakaran hutan dan lahan mencapai 292.922 Ha sedangkan pada tahun sebelumnya yaitu pada tahun 2019 mencapai 1.649.258 Ha dan pada tahun 2018 mencapai 529.266,64 Ha. Hal ini menunjukkan bahwa tingkat kebakaran hutan di Indonesia cukup tinggi. Untuk mengatasi permasalahan dalam penanganan kebakaran seperti yang dijelaskan sebelumnya, maka dibuatlah sebuah Sistem Peringatan Dini Kebakaran Hutan Menggunakan *IoT LoRa* 923 MHz.

Sistem Peringatan Dini Kebakaran Hutan Menggunakan *IoT LoRa* 923 MHz terdiri dari dua bagian yaitu subsistem pendeteksi dan subsistem mainstation. Subsistem pendeteksi dilengkapi *GPS* dan memanfaatkan empat sensor yaitu sensor suhu dan kelembaban, sensor asap, dan sensor api. Sedangkan subsistem main station akan ditempatkan pada pos pengawas yang memiliki akses *Wi-Fi*. Untuk memudahkan pengguna dalam memperoleh informasi dari sistem maka diperlukan

sebuah antarmuka pengguna. Dari antarmuka pengguna akan didapatkan informasi apakah kondisi di hutan aman atau terdeteksi adanya kebakaran.

Pada penelitian sebelumnya Arie Mahendra (2017) membuat sebuah rancang bangun sistem pendeteksi kebakaran berbasis *IoT* dan SMS *gateway*. Dalam sistem tersebut menggunakan tiga sensor yaitu sensor asap, sensor api, dan sensor suhu. Sistem akan menampilkan hasil pembacaan sensor secara *real-time* pada halaman *website* dan akan mengirimkan SMS secara otomatis apabila terdeteksi kebakaran. Penelitian Alpon Sepriando, Hartono, dan Retnadi Jatmiko (2020) membuat sistem deteksi kebakaran dengan memanfaatkan citra satelit Himawari-8. Citra satelit yang didapat kemudian diolah menjadi titik panas. Penelitian Budhi Fatanza Wiratama dan Muarrikh Prasidana (2020) memanfaatkan big data yang berasal dari pengguna *twitter* untuk membuat sebuah sistem peringatan *real-time* untuk bencana kebakaran pada Kota Jakarta. Dari unggahan yang dibuat oleh pengguna *twitter* akan dibuat sistem peringatan bencana kebakaran berupa peta geografis indikasi kebakaran yang ditampilkan secara *real-time*.

Pada penelitian ini akan dibuat sebuah antarmuka pengguna yang dapat memberikan informasi dari subsistem pendeteksi mengenai data hasil pembacaan sensor, status kondisi lingkungan, dan lokasi penempatan alat pendeteksi secara *real-time* pada halaman *website*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka didapatkan rumusan masalah yaitu bagaimana membuat antarmuka pengguna berbasis *website* untuk sistem peringatan dini kebakaran hutan.

1.3 Batasan Masalah

Dengan mengacu pada rumusan masalah, maka hal-hal yang berkaitan dengan perancangan antarmuka pengguna akan diberi batasan masalah agar pembahasan pada penelitian ini lebih fokus pada poin permasalahan. Adapun batasan masalah dapat diuraikan sebagai berikut:

1. Sistem Peringatan Dini Kebakaran Hutan Menggunakan *IoT Lora* 923 MHz telah tersedia.
2. Penelitian ini belum memperhatikan tingkat keamanan sistem.
3. Pengujian antarmuka pengguna menggunakan ESP32 untuk mengirimkan data secara berkala pada basis data.

1.4 Tujuan

Tujuan dari perancangan ini adalah untuk membuat sebuah antarmuka pengguna berbasis *website* yang dapat menampilkan informasi secara *real-time* dari sistem peringatan dini kebakaran hutan dan memberikan pesan apabila terdeteksi terjadinya kebakaran guna mempercepat proses tanggap bencana kebakaran.

1.5 Manfaat

Manfaat maupun luaran yang diharapkan pada penelitian ini dapat diuraikan sebagai berikut:

a. Bagi Pengembang

   Dengan adanya perancangan antarmuka pengguna berbasis *website* pada sistem peringatan dini kebakaran hutan dapat dikembangkan menjadi sistem informasi yang menyediakan data dari sistem pendeteksi kebakaran dengan cakupan yang lebih luas.

b. Bagi Pengguna

   Perancangan antarmuka pengguna pada sistem peringatan dini kebakaran hutan akan memudahkan pengguna dalam memahami informasi yang disampaikan sistem sehingga proses tanggap bencana kebakaran dapat segera dilakukan.

1.6 Sistematika Penulisan

Adapun sistematika penulisan dalam penyusunan skripsi ini adalah sebagai berikut:

BAB I PENDAHULUAN

Membahas latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat serta sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Menjelaskan dasar teori yang menunjang penelitian khususnya yang terkait dengan perancangan dan pembuatan sistem.

BAB III METODE PENELITIAN

Membahas metode yang digunakan pada penelitian dan perancangan sistem. Pada bab ini juga dijelaskan mengenai deskripsi sistem/gambaran sistem, perancangan dan pembuatan program, pengujian, analisis, serta penjelasan mengenai penerapan sistem secara keseluruhan.

BAB IV HASIL DAN PEMBAHASAN

Membahas tentang implementasi perancangan antarmuka pengguna berbasis *website* untuk sistem peringatan dini kebakaran hutan.

BAB V PENUTUP

Menjelaskan tentang pengambilan kesimpulan berdasarkan hasil dan pembahasan yang telah dilakukan pada penelitian ini. Serta memberikan saran berdasarkan pengembangan yang perlu dilakukan agar penelitian selanjutnya dapat berkembang menjadi lebih baik

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan diuraikan mengenai beberapa teori untuk menunjang implementasi perancangan antarmuka pengguna untuk *monitoring* sensor pada sistem deteksi dini kebakaran hutan berbasis *website* yang meliputi:

- Apache HTTP Server
- MySQL
- PHP
- *Model Waterfall*
- Analisis Kompleksitas Algoritma
- ESP32

2.1 Apache HTTP Server

Apache merupakan perangkat lunak yang berfungsi untuk menerima permintaan yang dikirimkan oleh browser kemudian memberikan tanggapan permintaan tersebut dalam bentuk halaman *website*. Apache yang memiliki nama resmi Apache HTTP Server dikembangkan oleh Apache Software Foundation dan bersifat *open source*. Berikut adalah beberapa kelebihan dari Apache:

- *Open source* dan gratis
- Dapat berfungsi di server Unix maupun Windows
- Konfigurasi yang tidak sulit
- Secara otomatis akan menjalankan file index.html

Apache bekerja ketika pengunjung mengakses sebuah halaman *website* dan *browser* akan mengirimkan permintaan ke server dan Apache akan mengirimkan respon dengan memuat file-file yang dibutuhkan. Komunikasi antara server dengan klien menggunakan protokol HTTP, Apache akan memastikan kelancaran dan keamanan komunikasi tersebut. (Yasin, 2019)

2.2 MySQL

Yasin (2019) menyatakan bahwa MySQL merupakan *database management system* yang menggunakan perintah-perintah dasar SQL *(Structured Query Language)*. Pengertian dari SQL sendiri adalah sebuah bahasa yang digunakan untuk pengaksesan data pada relational basis data atau basis data terstruktur. MySQL bersifat open source dengan lisensi GNU *General Public License* (GPL) sehingga

dalam penggunaannya baik untuk pribadi maupun komersial tidak diperlukan membayar lisensi. Sebuah basis data dalam MySQL dapat berupa satu atau lebih tabel, sehingga dalam MySQL dikenal baris, kolom dan tabel. Berikut adalah beberapa kelebihan MySQL:

- Dapat diintegrasikan dengan bahasa pemrograman lain
- Kebutuhan RAM yang tidak terlalu besar
- Dapat digunakan bersamaan (*multi user*) dalam satu waktu
- Bersifat open source
- Struktur tabel tergolong fleksibel

Pada SQL terdapat tiga jenis *query* atau perintah, yaitu DDL (*Data Definition Language*), DML (*Data Manipulation Language*), dan DCL (*Data Control Language*). Masing-masing penjelasannya dapat dilihat dibawah ini:

a. DDL (*Data Definition Language*)

Query SQL ini digunakan untuk mendefinisikan data pada basis data. Selain itu dengan menggunakan *query* ini juga dapat digunakan untuk membuat tabel baru, mengubah tabel, membuat indeks, menentukan struktur penyimpanan tabel dan sebagainya. Berikut adalah *query* yang dimiliki DDL:

- CREATE, untuk membuat basis data dan tabel.
- DROP, untuk menghapus basis data dan tabel.
- ALTER, untuk melakukan perubahan struktur tabel yang telah dibuat, misal menambahkan *field* menggunakan *query* ADD, mengganti nama field menggunakan *query* CHANGE ataupun menamakannya kembali menggunakan *query* RENAME, bisa juga menghapus *field* menggunakan *query* DROP.

b. DML *(Data Manipulation Language)*

Query DML dapat digunakan apabila DDL telah dibuat. DDL digunakan untuk memanipulasi basis data. Berikut adalah *query* yang dimiliki DML:

- INSERT, untuk memasukkan data pada tabel basis data.
- UPDATE, untuk mengubah data yang sudah ada pada tabel basis data.
- DELETE, untuk menghapus data pada tabel basis data.

c. DCL *(Data Control Language)*

Query DCL digunakan untuk memberikan hak otorisasi akses pada basis data, auditan penggunaan basis data, alokasi *space*, dan definisi *space*. Berikut adalah *query* yang dimiliki DCL:

- GRANT, untuk mengizinkan *user* mengakses tabel dalam basis data.
- REVOKE, untuk membatalkan izin hak *user*.
- COMMIT, untuk menetapkan penyimpanan basis data.
- ROLLBACK, untuk membatalkan penyimpanan basis data.

2.3 PHP

PHP atau *Hypertext Preprocessor* (sebelumnya *Personal Home Page*) merupakan bahasa pemrograman untuk pengembangan web. PHP merupakan bahasa pemrograman *script server-side* yang berarti program PHP di eksekusi oleh server. Ini berbeda dengan pemrograman *client-side* yang di proses di web *browser* seperti Javascript. Dalam penulisannya, file PHP dapat disematkan dalam *tag* HTML. PHP memiliki *syntax* dasar yaitu pembatas, variabel, komentar, dan fungsi. (Jannah, Sarwandi, & Creative, 2019)

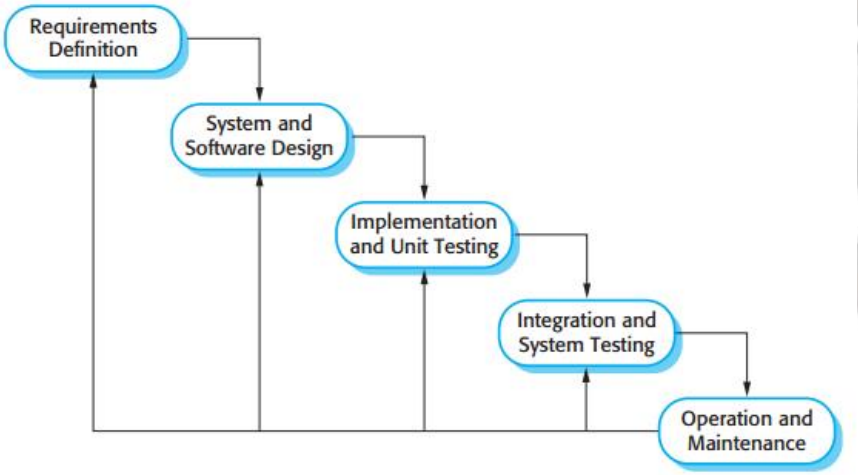
2.4 *Model Waterfall*

Model Waterfall atau *Classic Life Cycle* adalah model yang paling banyak digunakan dalam *software engineering*. Dengan pendekatan yang sistematis dan urut mulai dari level kebutuhan sistem dilanjutkan tahap analisis, desain, *coding*, *testing/verification*, dan *maintenance*. Sesuai dengan namanya *Model Waterfall* harus menyelesaikan satu tahap sebelum melanjutkan ke tahap berikutnya meski dalam prakteknya setiap tahapan bisa jadi saling tumpang tindih dan saling memberikan informasi satu sama lain sehingga memungkinkan adanya pengulangan tahap sebelumnya. (Rinjani, 2013)

Gambar 2.1 menunjukkan tahap-tahap utama dalam *Model Waterfall* menurut Sommerville (2011)

1. *Requirements analysis and definition*, tahap ini akan dibuat rincian mengenai spesifikasi sistem dengan konsultasi bersama pengguna sistem. Sehingga diketahui tujuan, kebutuhan, maupun kendala sistem secara keseluruhan yang nantinya dapat didokumentasikan dengan baik untuk mempermudah tahap berikutnya.

2. *System and software design*, pada tahapan ini akan dilakukan proses desain sistem yang mengalokasikan kebutuhan untuk sistem perangkat keras atau perangkat lunak dengan membangun sistem keseluruhan.
3. *Implementation and unit testing*, ketika sampai pada tahap ini maka desain perangkat lunak telah diimplementasikan dan setiap fungsi telah dapat bekerja sesuai kebutuhan.
4. *Integration and system testing*, setelah semua program terintegrasi dan akan diuji coba sebagai sebuah kesatuan sistem yang lengkap untuk memastikan bahwa semua kebutuhan sistem telah terpenuhi.
5. *Operation and maintenance*, pada tahap ini sistem mulai digunakan secara praktis dan pemeliharaan sistem melibatkan perbaikan kesalahan yang sebelumnya tidak ditemukan. Selain itu peningkatan layanan sistem dapat dilakukan ketika didapatkan kebutuhan baru.



Gambar 2.1 *Model Waterfall*

Sumber: (Sommerville, 2011)

2.5 Analisis Kompleksitas Algoritma

Algoritma merupakan urutan perintah dalam menyelesaikan masalah untuk memperoleh hasil keluaran dari setiap masukan yang diberikan dalam kurun waktu tertentu. Dalam membuat sebuah algoritma selain keberhasilan sebuah algoritma dalam menyelesaikan masalah, terdapat faktor lain yang layak mendapat perhatian selama proses perancangan algoritma yaitu efisiensi. Terdapat dua jenis efisiensi pada algoritma yaitu efisiensi waktu dan efisiensi ruang. Efisiensi waktu, juga disebut kompleksitas waktu, T(n), menunjukkan seberapa cepat algoritma yang bersangkutan

berjalan. Efisiensi ruang, juga disebut kompleksitas ruang, S(n), mengacu pada jumlah unit memori yang dibutuhkan oleh algoritma selain ruang yang dibutuhkan untuk masukkan dan keluarannya. Dalam menentukan waktu yang dibutuhkan suatu algoritma untuk menyelesaikan perintah-perintah didalamnya tentunya akan sulit jika menggunakan satuan waktu seperti detik atau milidetik. Ini dikarenakan pendekatan semacam itu dipengaruhi oleh kecepatan komputer dan kompiler yang digunakan, selain itu juga akan ditemui kesulitan dalam mencatat waktu yang sebenarnya dibutuhkan oleh program untuk menjalankan setiap operasinya. (Levitin, 2012)

Salah satu pendekatan yang mungkin digunakan dalam menghitung kompleksitas waktu adalah dengan menghitung berapa kali masing-masing operasi dasar yang khas atau tipikal dalam sebuah algoritma dijalankan. (Bahri & Maliki, 2012)

2.6 ESP32

Modul ESP32 merupakan mikrokontroler yang menyediakan modul *Wi-Fi* dan *Bluetooth* dan merupakan peningkatan dari versi sebelumnya yaitu ESP8266. Dengan *in-built antenna switches, RF balun, power amplifier, low-noise receive amplifier, filter,* dan *power management module*, maka chip pada ESP32 akan menambahkan fungsionalitas dan keserbagunaannya untuk pengembangan aplikasi pengguna. Berikut adalah beberapa seri ESP32 yang diproduksi oleh Espressif System pabrikan asal Cina.

1. Seri ESP32-SOLO

   Modul ESP32-SOLO-1 didasarkan pada ESP32-S0WD dan memiliki memori flash terintegrasi, sehingga memberikan solusi hemat biaya untuk aplikasi konektivitas berbasis *Wi-Fi* dan *Bluetooth/Bluetooth LE* yang sederhana.

2. Seri ESP32-WROOM

   ESP32-WROOM merupakan modul berbasis ESP32-D0WD dengan flash terintegrasi. Modul ini sangat sesuai untuk aplikasi konektivitas berbasis *Wi-Fi* dan *Bluetooth/Bluetooth LE* dan memberikan kinerja *dual-core* yang solid.

3. Seri ESP32-WROVER

   Seri ESP32-WROVER merupakan modul berbasis ESP32-D0WD SoC, yang juga memiliki memori *flash* terintegrasi dan SPIRAM. Modul ini memiliki kinerja dual-core yang bagus, dan sangat cocok untuk aplikasi yang membutuhkan lebih banyak memori, seperti *AIoT* dan aplikasi *gateway*. (ESP32 *Wi-Fi & Bluetooth Modules*)

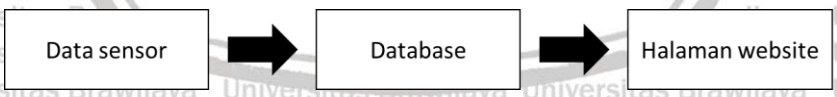
BAB III
METODE PENELITIAN

Penelitian ini didasarkan pada masalah yang bersifat aplikatif, yaitu perencanaan dan implementasi antarmuka pengguna pada sistem pendeteksi dini kebakaran hutan agar dapat bekerja sesuai dengan yang direncanakan dengan mengacu dalam rumusan masalah. Adapun metodologi yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Membuat gambaran sistem
2. Studi literatur
3. Perancangan sistem antarmuka pengguna
4. Pembuatan program sistem antarmuka pengguna
5. Pengujian sistem antarmuka pengguna
6. Analisis hasil pengujian
7. Pengambilan kesimpulan dan saran

3.1 Deskripsi Sistem

Pada Sistem Peringatan Dini Kebakaran Hutan Menggunakan *IoT LoRa* 923 MHz, bagian antarmuka pengguna masuk ke dalam subsistem *main station* dimana skema antarmuka pengguna dimulai dengan data kondisi lingkungan yang didapatkan subsistem *main station* dari subsistem pendeteksi telah berhasil masuk ke basis data. Data yang telah berhasil dikirim ke basis data akan ditampilkan pada halaman *website* dan akan diidentifikasi apakah data tersebut masuk dalam kriteria terdeteksi kebakaran atau masih dalam kriteria aman. Secara lebih jelas skema antarmuka pengguna dapat dilihat pada gambar 3.1.



Gambar 3.1. Diagram Blok Umum dari Antarmuka Pengguna.

Antarmuka pengguna yang dirancang diharapkan mampu menampilkan data hasil pembacaan kondisi lingkungan dari subsistem pendeteksi dan akan memberikan pesan kepada pengguna apabila sistem mendeteksi adanya kebakaran. Untuk dapat membuat sebuah sistem antarmuka pengguna berbasis *website* seperti yang diharapkan, maka dibutuhkan beberapa perangkat pendukung baik perangkat lunak maupun perangkat keras.

1. Perangkat Lunak

- Penyedia layanan web *hosting* untuk menyimpan semua file program yang dibutuhkan dalam membuat *website*. Sehingga *website* dapat diakses secara online.
- Sublime Text 2 sebagai *text editor* yang mendukung penulisan kode HTML, CSS, Java Script, dan PHP.
- Google Chrome sebagai web *browser* untuk menampilkan hasil pemrograman.

2. Perangkat Keras

Laptop yang menggunakan sistem operasi windows 8.1 dengan prosesor Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz digunakan untuk menjalankan semua perangkat lunak yang dibutuhkan dan ESP32-WROOM-32 sebagai pengirim data ke basis data.

3.2 Perancangan dan Pembuatan Sistem

Perancangan dan pembuatan sistem dalam penelitian ini terdiri dari perancangan basis data dan perancangan halaman *website*.

3.2.1 Perancangan Basis data

Dalam pembuatan antarmuka pengguna berbasis *website* untuk sistem peringatan dini kebakaran dibutuhkan sebuah basis data dengan satu tabel untuk menyimpan data yang akan ditampilkan pada halaman *website*. Data yang disimpan dalam tabel tersebut terdiri dari sebelas atribut yaitu:

1. Atribut ID sebagai *primary key*, atribut ini berupa angka bilangan bulat yang akan otomatis bertambah atau *auto increment*. Tujuan dari *auto increment* disini adalah untuk memudahkan proses penentuan data mana yang terakhir masuk pada basis data.
2. Atribut LAT berupa bilangan desimal yang merupakan nilai koordinat *latitude* dari alat pendeteksi kebakaran. Nantinya nilai dari atribut ini akan digunakan bersama dengan nilai dari atribut LNG untuk menampilkan lokasi alat pendeteksi kebakaran dalam bentuk peta.
3. Atribut LNG berupa bilangan desimal yang merupakan nilai koordinat *longitude* dari alat pendeteksi kebakaran. Nantinya nilai dari atribut ini akan digunakan bersama dengan nilai dari atribut LAT untuk menampilkan lokasi alat pendeteksi kebakaran dalam bentuk peta.

4. Atribut DAYA berupa bilangan desimal yang nilainya merepresentasikan nilai tegangan catu daya dari alat pendeteksi kebakaran hutan.
5. Atribut RSSI berupa bilangan bulat yang menunjukkan indikator kekuatan sinyal terima dari jaringan *LoRa*.
6. Atribut SUHU berupa bilangan desimal yang menunjukkan nilai dari hasil pembacaan sensor suhu pada lingkungan sekitar alat pendeteksi kebakaran hutan.
7. Atribut ASAP berupa bilangan desimal yang menunjukkan nilai dari hasil pembacaan sensor asap pada lingkungan sekitar alat pendeteksi kebakaran hutan.
8. Atribut API berupa bilangan bulat yang hanya memiliki dua nilai yaitu bernilai satu apabila sensor api tidak mendeteksi adanya api dan bernilai nol apabila sensor api mendeteksi adanya api.
9. Atribut KELEMBABAN berupa bilangan desimal yang menunjukkan nilai dari hasil pembacaan sensor kelembaban pada lingkungan sekitar alat pendeteksi kebakaran hutan.
10. Atribut COUNTER berupa bilangan bulat yang menunjukkan data keberapa yang dikirim dari alat pendeteksi kebakaran hutan. Dari data ini akan diketahui apabila terdapat data yang gagal terkirim.
11. Atribut WAKTU berupa tanggal dan jam yang menunjukkan waktu ketika data yang dikirim dari alat pendeteksi kebakaran hutan telah berhasil disimpan dalam basis data. Dari data ini, pengguna dapat mengetahui berapa selisih waktu antara data berhasil disimpan dalam basis data dengan waktu ketika mengakses sistem antarmuka pengguna.

Tabel 3.1 menunjukkan struktur dari tabel pada basis data yang akan dibuat sebagai berikut:

Tabel 3.1 Struktur Tabel pada Basis data

Field	Tipe Data
ID (primary key, auto increment)	Integer
LAT	Float
LNG	Float

DAYA	Float
RSSI	Integer
SUHU	Float
ASAP	Float
API	Integer
KELEMBABAN	Float
COUNTER	Integer
WAKTU	Timestamp

3.2.2 Perancangan Halaman *Website*

Untuk merancang bagaimana halaman web yang akan ditampilkan pada pengguna, terlebih dahulu perlu ditentukan apa saja informasi yang ingin disampaikan kepada pengguna. Mengingat bahwa Sistem Peringatan Dini Kebakaran Hutan Menggunakan *IoT LoRa* 923 MHz terdiri dari dua bagian subsistem yaitu subsistem pendeteksi dan subsistem *main station*. Subsistem pendeteksi akan mengambil data kondisi lingkungan yang akan dikirimkan ke subsistem mainstation dengan empat buah parameter sensor yaitu sensor suhu, sensor asap, sensor api, dan sensor kelembaban. Selain itu subsistem pendeteksi juga akan mengirimkan data koordinat lokasi kepada subsistem *main station*. Dari penjelasan sebelumnya maka dapat dirangkum apa saja yang harus ditampilkan pada halaman *website* sebagai antarmuka pengguna, berikut diantaranya:

- Data kondisi lingkungan berupa nilai sensor suhu, sensor asap, sensor api, sensor kelembaban, tegangan catu daya, dan nomor paket data. Selain kondisi lingkungan, antarmuka pengguna juga menampilkan informasi berupa waktu saat ini dan waktu ketika data telah berhasil masuk ke basis data, nilai RSSI, dan pesan singkat mengenai kondisi pada subsistem pendeteksi secara keseluruhan apakah masih masuk dalam kategori aman atau sudah termasuk bahaya. Semua informasi tersebut dapat terbaharui secara sesuai kondisi terbaru. Dengan kata lain dapat menampilkan data pembacaan sensor secara *real-time*.
- Data lokasi terkini dari alat yang bertindak sebagai subsistem pendeteksi.

Gambar berikut menunjukkan rancangan *layout* secara sederhana dari tampilan *website* yang akan dibuat.

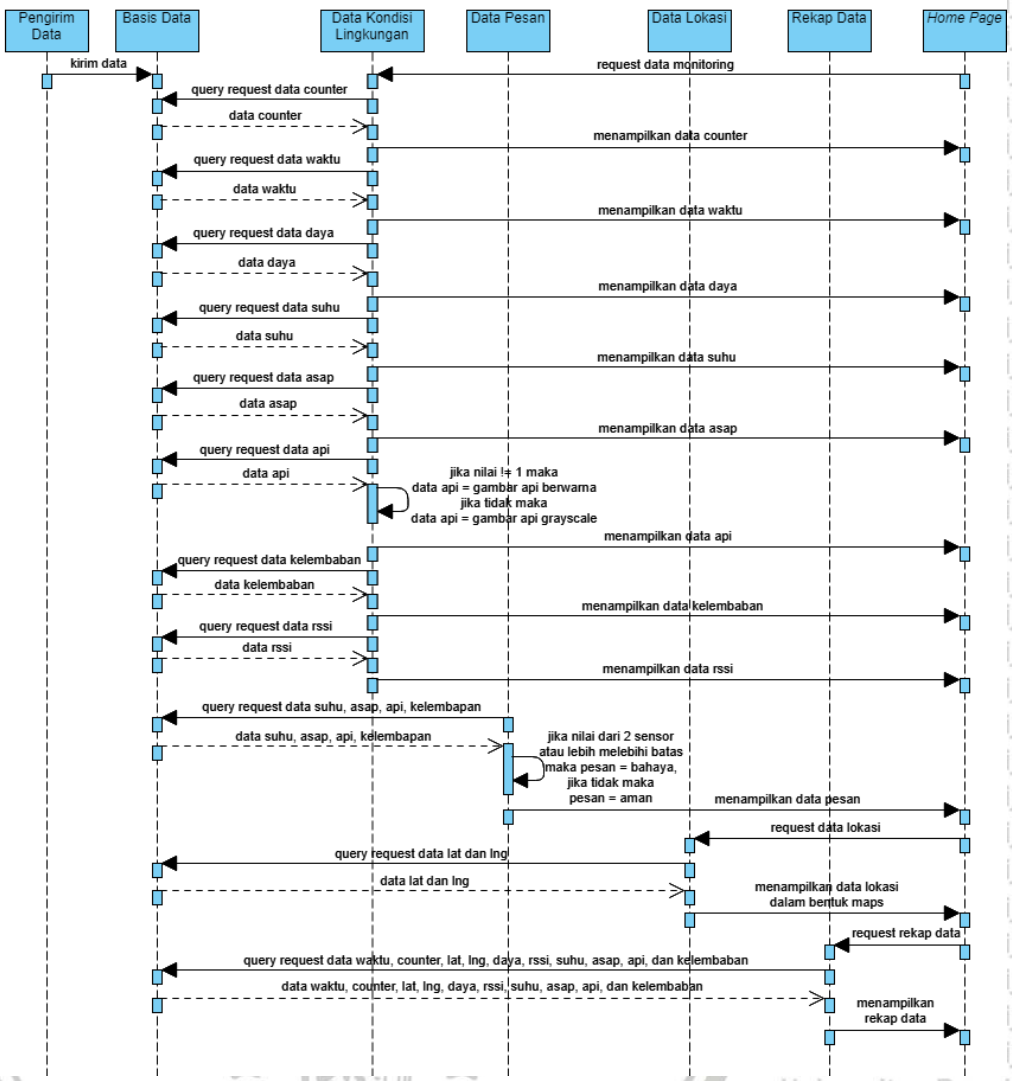
Gambar 3.2 *Layout Website*

3.2.3 Pembuatan Program

Setelah mengetahui informasi apa saja yang harus diketahui oleh pengguna, maka berikut akan dibuat sebuah diagram alir yang akan menunjukkan bagaimana membuat informasi tersebut sampai kepada pengguna. Pertama, untuk menampung semua data yang didapat perlu dibuat sebuah basis data, kemudian data yang akan ditampilkan akan disiapkan terlebih dahulu. Data yang akan ditampilkan dikelompokkan menjadi dua yaitu data lokasi dan data pembacaan sensor yang meliputi data nomor paket yang diterima, waktu ketika data berhasil masuk ke basis data, nilai tegangan catu daya, nilai sensor suhu, sensor asap, sensor api, sensor kelembaban, nilai RSSI, dan data pesan berupa satus kondisi lingkungan secara keseluruhan. Untuk menentukan apakah status kondisi lingkungan aman atau bahaya maka semua parameter sensor akan diidentifikasi nilainya, ketika dua atau lebih nilai sensor melebihi ambang batas maka sistem akan menganggap kondisi bahaya sedangkan selain kondisi tersebut maka sistem akan menganggap kondisi aman.

Data akan masuk pada *data-base* setiap lima detik sehingga agar data bersifat *real-time* maka, *website* harus menampilkan data setiap kurang dari lima detik. Sesuai kebutuhan yaitu hanya data yang terakhir masuk ke basis data atau data terbaru yang akan ditampilkan secara *real-time*. Sedangkan semua data-data sebelumnya akan ditampilkan dalam rekap data, dimana data pada rekap data tidak ditampilkan secara *real-time* atau dengan kata lain hanya menampilkan data pertama masuk sampai dengan data terakhir sebelum rekap

data diakses. Gambaran keseluruhan sistem antarmuka pengguna dapat dilihat pada diagram *sequence* berikut yang ditunjukkan gambar 3.3.

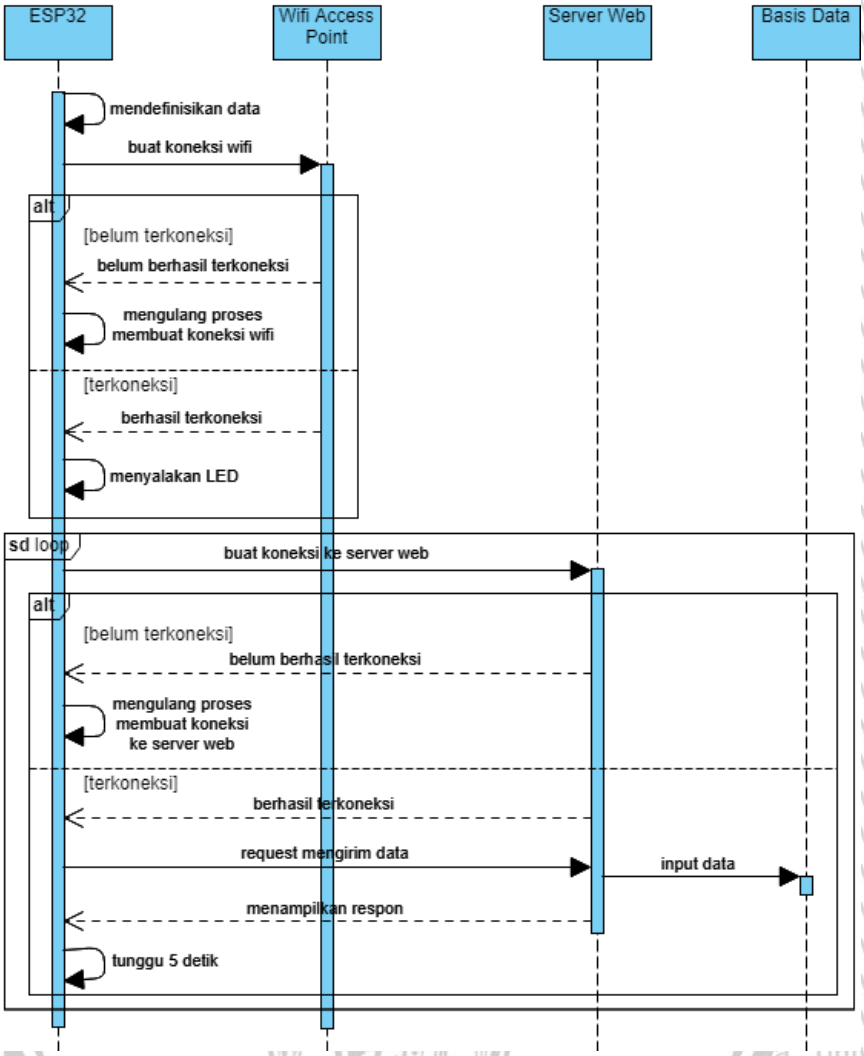


Gambar 3.3 Diagram *Sequence* Keseluruhan Sistem Antarmuka Pengguna

3.3 Pengujian Fungsionalitas Sistem

Pada pengujian fungsionalitas ini akan diketahui apakah *website* sebagai antarmuka pengguna untuk *monitoring* sensor pada sistem peringatan dini kebakaran hutan yang telah selesai dibuat dapat memberikan informasi kepada pengguna sesuai dengan yang diharapkan pada tahap perancangan. Pengujian dilakukan dengan mengamati tampilan *website* ketika adanya data yang dikirim ke basis data apakah data tersebut dapat ditampilkan serta dapat terbaharui sesuai dengan data terbaru dan *real-time*. Sebagai simulasi ketika antarmuka pengguna diintegrasikan dengan sistem deteksi dini kebakaran hutan secara keseluruhan maka pengujian fungsionalitas memanfaatkan modul ESP32. Modul ESP32 akan diprogram sedemikian rupa untuk mengirimkan data ke basis data secara berkala setiap lima detik untuk melihat apakah

setiap data yang dikirim tersebut dapat ditampilkan pada antarmuka pengguna. Berikut adalah diagram *sequence* pemrograman ESP32 yang ditunjukkan gambar 3.4.



Gambar 3.4 Diagram *Sequence* Pemrograman ESP32

3.4 Analisis Kompleksitas Algoritma

Analisis kompleksitas algoritma yang digunakan pada tahap ini adalah kompleksitas waktu atau T(n). Kompleksitas waktu atau T(n) didapatkan dari menentukan jumlah operasi yang dilakukan dalam suatu algoritma berdasarkan ukuran masukan n. Analisis ini dilakukan dengan mengidentifikasi setiap langkah pada *pseudocode* yang dibuat. Secara garis besar pada perancangan antarmuka pengguna berbasis *website* untuk *monitoring* sensor pada sistem peringatan dini kebakaran hutan akan memiliki tiga fitur utama yaitu yang pertama dapat menampilkan nilai dari basis data, yang kedua dapat menampilkan nilai dari basis data namun dalam bentuk titik koordinat pada peta, dan yang ketiga dapat mengkategorikan suatu kondisi. Dari ketiga fitur

tersebut akan dilakukan analisis kompleksitas waktu untuk masing-masing algoritma yang digunakan pada setiap fitur.

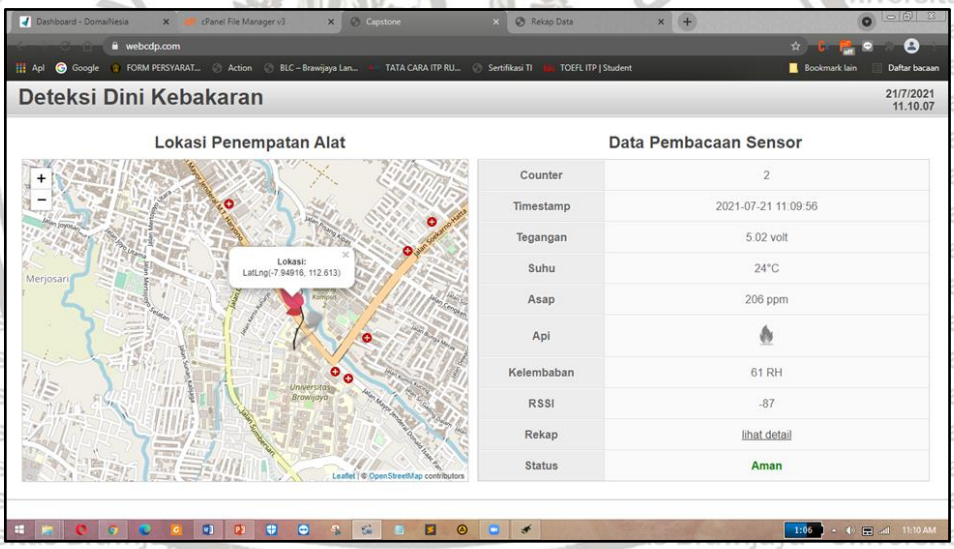
3.5 Penarikan Kesimpulan

Penarikan kesimpulan dapat dilakukan ketika semua hasil pengujian telah dapat dirangkum. Dari data hasil pengujian akan disimpulkan apakah perancangan yang dilakukan telah dapat diimplementasikan dan dapat bekerja dengan baik sesuai dengan yang diharapkan.

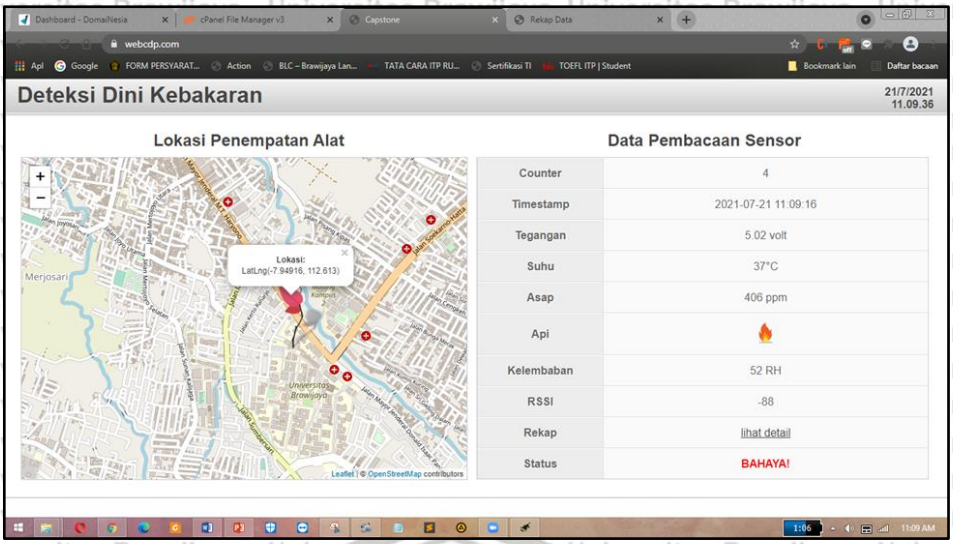
BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengujian Fungsionalitas

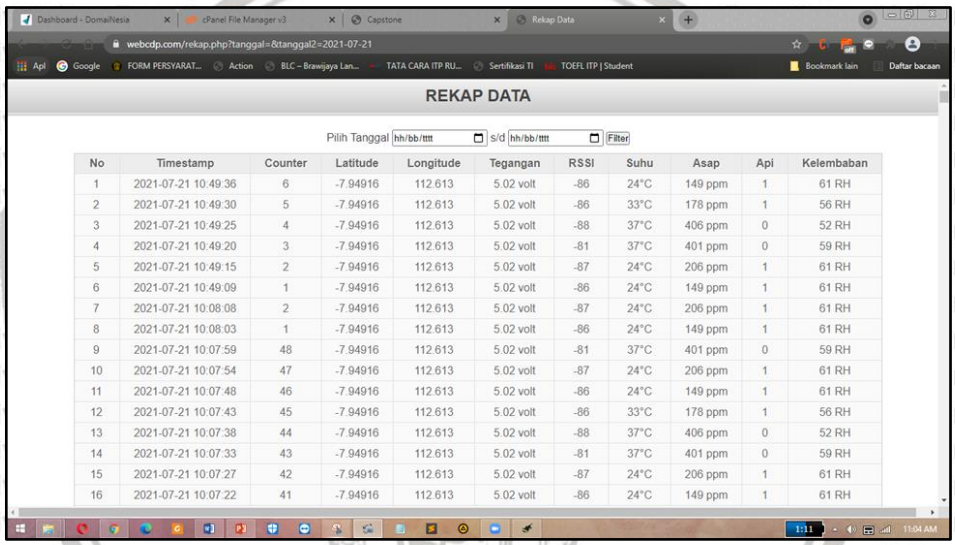
Pada pengujian fungsionalitas akan diketahui apakah *website* sebagai antarmuka pengguna untuk *monitoring* sensor pada sistem peringatan dini kebakaran hutan yang telah selesai dibuat dapat memberikan informasi kepada pengguna sesuai dengan yang diharapkan pada tahap perancangan. Pengujian dilakukan dengan mengakses halaman web yang telah dibuat sesuai domain yang digunakan yaitu www.webcdp.com. Untuk melihat apakah data yang ditampilkan telah dapat terbaharui sesuai dengan data terbaru dan *real-time* maka pengujian dilakukan dengan mengirimkan data secara berkala ke dalam basis data. Dalam hal ini penulis memanfaatkan modul ESP32 untuk mengirim data ke basis data sebagai simulasi saat *website* diintegrasikan dengan sistem deteksi dini kebakaran hutan. Berikut adalah tampilan *website* yang didapatkan.



Gambar 4.1 Tampilan *Website* Ketika Kondisi Terdeteksi Aman



Gambar 4.2 Tampilan *Website* Ketika Kondisi Terdeteksi Bahaya



No	Timestamp	Counter	Latitude	Longitude	Tegangan	RSSI	Suhu	Asap	Api	Kelembaban
1	2021-07-21 10:49:36	6	-7.94916	112.613	5.02 volt	-86	24°C	149 ppm	1	61 RH
2	2021-07-21 10:49:30	5	-7.94916	112.613	5.02 volt	-86	33°C	178 ppm	1	56 RH
3	2021-07-21 10:49:25	4	-7.94916	112.613	5.02 volt	-88	37°C	406 ppm	0	52 RH
4	2021-07-21 10:49:20	3	-7.94916	112.613	5.02 volt	-81	37°C	401 ppm	0	59 RH
5	2021-07-21 10:49:15	2	-7.94916	112.613	5.02 volt	-87	24°C	206 ppm	1	61 RH
6	2021-07-21 10:49:09	1	-7.94916	112.613	5.02 volt	-86	24°C	149 ppm	1	61 RH
7	2021-07-21 10:08:08	2	-7.94916	112.613	5.02 volt	-87	24°C	206 ppm	1	61 RH
8	2021-07-21 10:08:03	1	-7.94916	112.613	5.02 volt	-86	24°C	149 ppm	1	61 RH
9	2021-07-21 10:07:59	48	-7.94916	112.613	5.02 volt	-81	37°C	401 ppm	0	59 RH
10	2021-07-21 10:07:54	47	-7.94916	112.613	5.02 volt	-87	24°C	206 ppm	1	61 RH
11	2021-07-21 10:07:48	46	-7.94916	112.613	5.02 volt	-86	24°C	149 ppm	1	61 RH
12	2021-07-21 10:07:43	45	-7.94916	112.613	5.02 volt	-86	33°C	178 ppm	1	56 RH
13	2021-07-21 10:07:38	44	-7.94916	112.613	5.02 volt	-88	37°C	406 ppm	0	52 RH
14	2021-07-21 10:07:33	43	-7.94916	112.613	5.02 volt	-81	37°C	401 ppm	0	59 RH
15	2021-07-21 10:07:27	42	-7.94916	112.613	5.02 volt	-87	24°C	206 ppm	1	61 RH
16	2021-07-21 10:07:22	41	-7.94916	112.613	5.02 volt	-86	24°C	149 ppm	1	61 RH

Gambar 4.3 Tampilan Hasil Rekap Data

Dari pengamatan yang telah dilakukan selama uji coba ini didapatkan data sebagai berikut.

Tabel 4.1. Tabel Pengujian Fungsionalitas

No	Keterangan	Proses	Output	☑/☒
1	Tanggal dan jam sekarang	Menampilkan tanggal dan jam digital sesuai waktu saat mengakses web.	Menampilkan Jam saat ini.	☑
2	Lokasi	Mengambil data terbaru kolom LAT dan LNG pada basis data,	Menampilkan marker sesuai data *latitude* dan *longitude* terbaru.	☑

		mengkodekan kedalam format json.		
3	Counter	Mengambil data terbaru kolom COUNTER pada basis data, menampilkan data dengan ajax.	Data *counter* terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
4	Timestamp	Mengambil data terbaru kolom WAKTU pada basis data, menampilkan data dengan ajax.	Data *timestamp* terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
5	Tegangan	Mengambil data terbaru kolom DAYA pada basis data, menampilkan data dengan ajax.	Data tegangan terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
6	Suhu	Mengambil data terbaru kolom SUHU pada basis data, menampilkan data dengan ajax.	Data suhu terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
7	Asap	Mengambil data terbaru kolom ASAP pada basis data, menampilkan data dengan ajax.	Data asap terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
8	Api	Mengambil data terbaru kolom API pada basis data, menampilkan data dengan ajax.	Data api terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
9	Kelembaban	Mengambil data terbaru kolom KELEMBABAN pada basis data, menampilkan data dengan ajax.	Data kelembaban terbaharui sesuai dengan data terbaru tanpa *refresh* halaman.	☑
10	RSSI	Mengambil data terbaru kolom RSSI pada basis	Data RSSI terbaharui sesuai dengan data	☑

		data, menampilkan data dengan ajax.	terbaru tanpa *refresh* halaman.	
11	Rekap	Mengambil semua data dari tabel alat, menampilkan pada halaman baru.	Semua data ditampilkan pada halaman baru.	☑
12	Status	Mengambil data terbaru kolom SUHU, ASAP, API, dan KELEMBABAN pada basis data, melakukan seleksi kondisi menampilkan data pesan dengan ajax.	Menampilkan pesan bahaya ketika nilai dari dua atau lebih kolom melebihi batas yang telah ditentukan, menampilkan pesan aman ketika tidak ada atau hanya satu nilai dari kolom yang melebihi batas.	☑

Untuk melihat apakah data yang ditampilkan bersifat *real-time* dapat dilihat perbandingan waktu ketika data berhasil dikirim dari ESP32 dengan data waktu ketika data ditampilkan pada halaman *website*. Data dikatakan *real-time* apabila selisih waktu antara data berhasil dikirim dari ESP32 dengan data waktu ketika data ditampilkan pada halaman *website* kurang dari lima detik. Waktu ketika data berhasil dikirim dari ESP32 dapat dilihat pada serial monitor pada Arduino IDE, sedangkan waktu ketika data ditampilkan pada halaman *website* dapat diketahui dengan mengamati jam yang berada pada halaman *website* saat data ditampilkan. Berikut adalah tabel yang menunjukkan rata-rata untuk sepuluh data dari perbandingan waktu tersebut.

Tabel 4.2 Selisih Waktu Data Terkirim dengan Data Ditampilkan

No	Counter	Data Terkirim	Data Ditampilkan	Selisih Waktu (s)
1	1	19:28:07	19:28:08	1
2	2	19:28:12	19:28:13	1
3	3	19:28:17	19:28:18	1
4	4	19:28:23	19:28:23	0

5	5	19:28:28	19:28:29	1
6	6	19:28:33	19:28:34	1
7	7	19:28:38	19:28:39	1
8	8	19:28:44	19:28:45	1
9	9	19:28:49	19:28:50	1
10	10	19:28:54	19:28:55	1
Total Selisih Waktu (s)				9
Rata-rata Selisih Waktu (s)				0,9

4.2 Analisis Kompleksitas Algoritma

Pada dasarnya *website* yang dibuat memiliki tiga fitur utama yaitu yang pertama dapat menampilkan nilai dari basis data, yang kedua dapat menampilkan nilai dari basis data namun dalam bentuk titik koordinat pada peta, dan yang ketiga dapat mengkategorikan suatu kondisi. Fitur yang pertama dapat dilakukan dengan membuat koneksi pada basis data. Untuk membuat koneksi pada basis data dibutuhkan inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi. Kemudian memilih kolom pada tabel basis data yang ingin ditampilkan nilainya. Berikut adalah *pseudocode* untuk menampilkan nilai dari basis data.

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi.
2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*.
4. Maka tampilkan pesan *error*.
5. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan.
6. Akses data pada tabel basis data.
7. Jika jumlah baris data yang ada > 0
8. Maka tampilkan isi setiap baris.

Berdasarkan *pseudocode* untuk menampilkan nilai dari basis data didapatkan nilai kompleksitas algoritma T(n) = n + 7. Nilai kompleksitas algoritma didapatkan dari operasi dasar yang dilakukan pada *pseudocode*. Operasi dasar yang dilakukan untuk menampilkan nilai dari basis data adalah sebagai berikut:

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi. Inisialisasi ini dilakukan satu kali yang nantinya digunakan untuk membuat koneksi ke basis data.
2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi. Proses ini dilakukan satu kali untuk membuat koneksi ke basis data.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*. Pengecekkan ini dilakukan sebanyak satu kali untuk mengetahui apakah proses membuat koneksi ke basis data berhasil atau gagal.
4. Maka tampilkan pesan *error*. Proses menampilkan pesan ini dilakukan satu kali apabila kondisi pada langkah sebelumnya terpenuhi.
5. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan. Inisalisasi ini dilakukan satu kali untuk menentukan data mana yang akan dipilih untuk ditampilkan.
6. Akses data pada tabel basis data. Proses ini dilakukan satu kali untuk mengakses data pada tabel basis data sesuai dengan inisialisasi sebelumnya.
7. Jika jumlah baris data yang ada > 0. Pengecekkan kondisi ini dilakukan satu kali untuk mengetahui apakah data yang diinginkan tersedia atau tidak.
8. Maka tampilkan isi setiap baris. Proses ini dilakukan sebanyak jumlah baris data yang ada atau dapat dinotasikan dengan n.

Secara lebih ringkas dapat dilihat pada tabel 4.3 yang menunjukkan jumlah iterasi untuk setiap operasi dasar pada *pseudocode* yang telah dibuat.

Tabel 4.3 Jumlah Iterasi untuk Menampilkan Nilai dari Basis Data

No	Operasi Dasar	Jumlah Iterasi
1.	Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi	1
2.	Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi	1
3.	Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*	1
4.	Maka tampilkan pesan *error*	1

5.	Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan	1
6.	Akses data pada tabel basis data	1
7.	Jika jumlah baris data yang ada > 0	1
8.	Maka tampilkan isi setiap baris	n
	Total	n + 7

Fitur yang kedua yaitu dapat menampilkan nilai dari basis data namun dalam bentuk titik koordinat pada peta. Untuk membuat fitur ini, pertama perlu dibuat data *JSON* yang berisi data *latitude* dan *longitude* kemudian setelah data *JSON* telah tersedia selanjutnya yaitu menampilkan data *JSON* dalam bentuk peta dengan membuat sebuah variabel yang akan menampung peta yang akan dibuat. Kemudian menentukan titik koordinat awal atau titik tengah dan nilai *zoom* yang akan ditampilkan pertama kali. Setelah itu menambahkan *layer* peta dan membuat *marker* sesuai dengan koordinat yang didapatkan pada data *JSON*. Berikut adalah *pseudocode* untuk membuat data *JSON*:

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi.
2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*.
4. Maka tampilkan pesan *error*.
5. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan.
6. Akses data pada tabel basis data.
7. Jika jumlah baris data yang ada > 0
8. Maka simpan data dalam sebuah variabel.
9. *Encode* data pada variabel menjadi format *JSON*.

Dari *pseudocode* diatas didapatkan nilai kompleksitas algoritma T(n) = 9. Nilai T(n) tersebut didapatkan dari operasi dasar berikut:

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi.

   Inisialisasi ini dilakukan satu kali yang nantinya digunakan untuk membuat koneksi ke basis data.

2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi. Proses ini dilakukan satu kali untuk membuat koneksi ke basis data.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*. Pengecekkan ini dilakukan sebanyak satu kali untuk mengetahui apakah proses membuat koneksi ke basis data berhasil atau gagal.
4. Maka tampilkan pesan *error*. Proses menampilkan pesan ini dilakukan satu kali apabila kondisi pada langkah sebelumnya terpenuhi.
5. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan. Inisialisasi ini dilakukan satu kali untuk menentukan data mana yang akan dipilih untuk ditampilkan.
6. Akses data pada tabel basis data. Proses ini dilakukan satu kali untuk mengakses data pada tabel basis data sesuai dengan inisialisasi sebelumnya.
7. Jika jumlah baris data yang ada > 0. Pengecekkan kondisi ini dilakukan satu kali untuk mengetahui apakah data yang diinginkan tersedia atau tidak.
8. Maka simpan data dalam sebuah variabel. Proses ini dilakukan sebanyak satu kali apabila kondisi sebelumnya terpenuhi.
9. *Encode* data pada variabel menjadi format *JSON*. Proses ini dilakukan sebanyak satu kali.

Secara lebih ringkas dapat dilihat pada tabel 4.4 yang menunjukkan jumlah iterasi untuk setiap operasi dasar pada *pseudocode* yang telah dibuat.

Tabel 4.4 Jumlah Iterasi untuk Membuat Data *JSON*

No	Operasi Dasar	Jumlah Iterasi
1.	Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi	1
2.	Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi	1
3.	Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*	1
4.	Maka ampilkan pesan *error*	1
5.	Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan	1

6.	Akses data pada tabel basis data	1
7.	Jika jumlah baris data yang ada > 0	1
8.	Maka simpan data dalam sebuah variabel	1
9.	*Encode* data pada variabel menjadi format *JSON*	1
Total		9

Setelah membuat data *JSON* langkah berikutnya untuk membuat fitur kedua adalah menampilkan peta dengan *marker* sesuai dengan koordinat pada data *JSON*. Berikut adalah *pseudocode* untuk menampilkan peta:

1. Buat variabel untuk menampung peta dan atur titik tengah serta nilai *zoom* peta.
2. Tambahkan *layer* peta.
3. Buat variabel untuk menampung gambar yang akan digunakan sebagai *marker*.
4. Atur koordinat awal untuk *marker*.
5. Ambil nilai pada data *JSON*.
6. Ubah koordinat *marker* dengan koordinat yang didapat dari data *JSON*.

Dari *pseudocode* diatas didapatkan nilai kompleksitas algoritma T(n) = 6. Nilai T(n) tersebut didapatkan dari operasi dasar berikut:

1. Buat variabel untuk menampung peta dan atur titik tengah serta nilai *zoom* peta. Proses ini dilakukan satu kali pada awal membuat peta.
2. Tambahkan *layer* peta. Proses ini dilakukan satu kali setelah tempat untuk menampilkan peta tersedia.
3. Buat variabel untuk menampung gambar yang akan digunakan sebagai *marker*. *Marker* yang dibutuhkan hanya satu sehingga proses ini dilakukan sebanyak satu kali.
4. Atur koordinat awal untuk *marker*. Proses ini dilakukan satu kali untuk menentukan posisi *marker* ketika belum ada data koordinat dari data *JSON*.
5. Ambil nilai pada data *JSON*. Proses ini dilakukan satu kali.
6. Ubah koordinat *marker* dengan koordinat yang didapat dari data *JSON*. Proses ini dilakukan satu kali setelah nilai pada data *JSON* diperoleh.

Secara lebih ringkas dapat dilihat pada tabel 4.5 yang menunjukkan jumlah iterasi untuk setiap operasi dasar pada *pseudocode* yang telah dibuat.

Tabel 4.5 Jumlah Iterasi untuk Menampilkan Peta

No	Operasi Dasar	Jumlah Iterasi
1.	Buat variabel untuk menampung peta dan atur titik tengah serta nilai *zoom* peta	1
2.	Tambahkan *layer* peta	1
3.	Buat variabel untuk menampung gambar yang akan digunakan sebagai *marker*	1
4.	Atur koordinat awal untuk *marker*	1
5.	Ambil nilai pada data *JSON*	1
6.	Ubah koordinat *marker* dengan koordinat yang didapat dari data *JSON*	1
Total		6

Fitur yang ketiga yaitu dapat menentukan kategori suatu kondisi apakah termasuk kondisi aman atau bahaya. Kategori aman atau bahaya bergantung pada nilai dari data suhu, asap, api, dan kelembaban. Dari keempat nilai tersebut apabila dua atau lebih nilai melebihi batas yang ditentukan maka akan masuk kategori bahaya namun jika kondisi tersebut tidak terpenuhi maka termasuk kategori kondisi aman. Berikut adalah *pseudocode* untuk menentukan kategori kondisi:

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi.
2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*.
4. Maka tampilkan pesan *error*.
5. Buat empat variabel.
6. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan.
7. Jika jumlah baris data yang ada > 0
8. Maka simpan setiap data pada variabel.
9. Jika dua atau lebih variabel memiliki nilai yang melebihi batas.
10. Maka kategori kondisi adalah bahaya.
11. Jika kondisi sebelumnya tidak terpenuhi.
12. Maka kategori kondisi adalah aman.

Dari *pseudocode* diatas didapatkan nilai kompleksitas algoritma T(n) = 12. Nilai T(n) tersebut didapatkan dari operasi dasar berikut:

1. Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi. Inisialisasi ini dilakukan satu kali yang nantinya digunakan untuk membuat koneksi ke basis data.
2. Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi. Proses ini dilakukan satu kali untuk membuat koneksi ke basis data.
3. Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*. Pengecekkan ini dilakukan sebanyak satu kali untuk mengetahui apakah proses membuat koneksi ke basis data berhasil atau gagal.
4. Maka tampilkan pesan *error*. Proses menampilkan pesan ini dilakukan satu kali apabila kondisi pada langkah sebelumnya terpenuhi.
5. Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan. Inisialisasi ini dilakukan satu kali untuk menentukan data mana yang akan dipilih untuk ditampilkan.
6. Akses data pada tabel basis data. Proses ini dilakukan satu kali untuk mengakses data pada tabel basis data sesuai dengan inisialisasi sebelumnya.
7. Jika jumlah baris data yang ada > 0. Pengecekkan kondisi ini dilakukan satu kali untuk mengetahui apakah data yang diinginkan tersedia atau tidak
8. Maka simpan setiap data pada variabel. Proses ini dilakukan satu kali apabila kondisi sebelumnya terpenuhi.
9. Jika dua atau lebih variabel memiliki nilai yang melebihi batas. Proses pengecekkan ini dilakukan satu kali dengan cara membandingkan setiap nilai variabel dengan nilai batas kemudian mencari apakah ada dua nilai atau lebih yang melebihi nilai batas.
10. Maka kategori kondisi adalah bahaya. Proses ini dilakukan satu kali apabila kondisi sebelumnya terpenuhi.
11. Jika kondisi sebelumnya tidak terpenuhi. Proses ini dilakukan satu kali ketika kondisi pada langkah kesembilan tidak terpenuhi.
12. Maka kategori kondisi adalah aman. Proses ini dilakukan satu kali apabila kondisi sebelumnya terpenuhi.

Secara lebih ringkas dapat dilihat pada tabel 4.6 yang menunjukkan jumlah iterasi untuk setiap operasi dasar pada *pseudocode* yang telah dibuat.

Tabel 4.6 Jumlah Iterasi untuk Menentukan Kategori Kondisi

No	Operasi Dasar	Jumlah Iterasi
1.	Inisialisasi nama *server*, nama basis data, nama pengguna, dan kata sandi	1
2.	Akses basis data sesuai dengan nama *server*, nama basis data, nama pengguna, dan kata sandi	1
3.	Jika nama *server* == *false* || nama basis data == *false* || nama pengguna == *false* || kata sandi == *false*	1
4.	Maka tampilkan pesan *error*	1
5.	Inisialisasi nama kolom, nama tabel, jumlah baris yang ditampilkan	1
6.	Akses data pada tabel basis data	1
7.	Jika jumlah baris data yang ada > 0	1
8.	Maka simpan setiap data pada variabel	1
9.	Jika dua atau lebih variabel memiliki nilai yang melebihi batas	1
10.	Maka kategori kondisi adalah bahaya	1
11.	Jika kondisi sebelumnya tidak terpenuhi	1
12.	Maka kategori kondisi adalah aman	1
	Total	12

BAB V
PENUTUP

Pada bagian bab terakhir ini akan memuat kesimpulan dari hasil perancangan dan implementasi sistem yang telah dibuat. Selain itu akan diuraikan saran untuk pengembangan sistem agar menjadi lebih baik.

5.1 Kesimpulan

Dari hasil implementasi perancangan antarmuka pengguna untuk sistem deteksi dini kebakaran hutan berbasis *website* dapat diambil kesimpulan bahwa implementasi dapat dilaksanakan dan sistem dapat bekerja sesuai harapan. Ini dapat dilihat dari hasil pengujian fungsionalitas pada *website* dimana semua kebutuhan telah dapat terpenuhi dan rata-rata waktu yang dibutuhkan untuk menampilkan data yang telah dikirim adalah 0,9 s yang berarti data dapat ditampilkan kurang dari lima detik atau dengan kata lain data dapat ditampilkan secara *real-time*. Sedangkan hasil analisis kompleksitas waktu algoritma adalah sebagai berikut, untuk algoritma menampilkan nilai dari basis data didapatkan T(n) = n + 7, untuk algoritma membuat data *JSON* didapatkan T(n) = 9, untuk algoritma menampilkan peta didapatkan T(n) = 6, dan untuk algoritma menentukan kategori kondisi didapatkan T(n) = 12.

5.2 Saran

Dalam implementasi perancangan sistem tentunya masih diperlukan pengembangan lebih lanjut guna didapatkan sistem yang lebih baik lagi. Berikut adalah beberapa saran yang dapat dijadikan pertimbangan untuk pengembangan sistem selanjutnya:

- Menambahkan fitur lain yang akan mempermudah proses tanggap bencana kebakaran seperti menambahkan rute terdekat untuk menuju lokasi kebakaran hutan.
- Sistem dapat digunakan secara luas dengan membuat akun bagi pengguna dan menyediakan sistem yang memudahkan pengguna untuk saling berkoordinasi.
- Menambahkan mekanisme keamanan selama proses pengiriman data.

DAFTAR PUSTAKA

Bahri, Raden Sofian., & Maliki, Irfan. (2012). Perbandingan Algoritma Template Matching dan Feature Extraction pada Optical Character Recognition. *Jurnal Komputer dan Informatika (KOMPUTA)*. I (1): 29-35.

Fatanza Wiratama, Budhi & Prasidana, Muarrikh. 2020. Sistem Peringatan Real-Time Berbasis Twitter untuk Bencana Kebakaran di Kota Jakarta. *Jurnal Riset Jakarta*.

Jannah, M., Sarwandi, & Creative, C. (2019). *Mahir Bahasa Pemrograman PHP*. Jakarta: Elex Media Komputindo.

K., Yasin. (2019). Apa Itu Apache? Kelebihan dan Kekurangannya. https://www.niagahoster.co.id/blog/apache-adalah/. (diakses 5 Februari 2021).

K., Yasin. (2019). Pengertian MySQL, Fungsi, dan Cara Kerjanya (Lengkap). https://www.niagahoster.co.id/blog/mysql-adalah/. (diakses 5 Februari 2021).

Levitin, Anany. (2012). *Introduction to the Design & Analysis of Algorithms*. USA, Pearson Education.

Rinjani, Muhammad Angga. 2013). 4 Metodologi Pengembangan Software berbasis SDLC (Software Development Life Cycle). https://id.scribd.com/doc/247917950/4-Metodologi-Pengembangan-Software-Berbasis-SDLC-Software-Development-Life-Cycle-Andgaa-web. (diakses 28 April 2021).

Sasmoko, D., & Mahendra, A. (2017). Rancang Bangun Sistem Pendeteksi Kebakaran Berbasis IoT dan SMS Gateway Menggunakan Arduino. *Jurnal SIMETRIS*. VIII (2):469-476.

Sepriando, Alpon & Hartono, Hartono & Jatmiko, Retnadi. 2019. Deteksi Kebakaran Hutan dan Lahan menggunakan Citra Satelit Himawari-8 di Kalimantan Tengah. *Jurnal Sains & Teknologi Modifikasi Cuaca*. XX (2):79-89.

SiPongi Karhutla *Monitoring* System. (2020). Rekapitulasi Luas Kebakaran Hutan dan Lahan (Ha) Per Provinsi Di Indonesia Tahun 2015-2020 (Data s/d 30 September 2020). Indonesia: Direktorat PKHL Kementerian Lingkungan Hidup dan Kehutanan RI.

Sommerville, Ian. (2011). *Software Engineering (9th Edition)*. USA, Pearson Education.

www.espressif.com. ESP32 Wi-Fi & Bluetooth Modules | Espressif. https://www.espressif.com/en/products/modules/esp32. (diakses 07 Mei 2021)

LAMPIRAN

Lampiran 1

File HTML

```
<!DOCTYPE HTML>
<html>
<head>
<title>Capstone</title>
<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="index.css">
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
<style type="text/css">
#map {
width: 645px;
height: 466px;
margin-top: 10px;
border: 1px solid #ccc;
}
</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
//setInterval memberikan delay pada saat data akan ditampilkan
setInterval(function(){
//Metode load digunakan untuk menampilkan data
    $('#COUNTER').load("COUNTER.php")
    $('#WAKTU').load("WAKTU.php")
        $('#DAYA').load("DAYA.php")
        $('#SUHU').load("SUHU.php")
        $('#ASAP').load("ASAP.php")
        $('#API').load("API.php")
        $('#KELEMBABAN').load("KELEMBABAN.php")
        $('#RSSI').load("RSSI.php")
```

```
        $('#PESAN').load("PESAN.php")
}, 1000);
});
</script>
</head>
<body>
<div id="wrapper">
<div id="header">
        <div id="kiri">
                <h1 id="logo">Deteksi Dini Kebakaran</h1>
        </div>
        <div id="kanan">
        <h4 id="waktu">
                <span id="tanggal"></span><br>
                <span id="jam"></span>
                <script src="waktu.js"></script>
        </h4>
        </div><!--end of kanan-->
</div><!--end of header-->
<div id="info">
        <div class="container">
                <div id="left_section">
                <div class="content">
                        <div class="content_title">
                                <h2>Lokasi Penempatan Alat</h2>
                        </div><!--end of content title-->
                        <div id="map"></div>
                        <script
src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
                <script src="map.js"></script>
                </div><!--end of content-->
                </div><!--end of left section-->
                <div id="right_section">
                <div class="content">
```

```
<div class="content_title">
        <h2>Data Pembacaan Sensor</h2>
</div><!--end of content title-->
<!--ISI DATA PEMBACAAN SENSOR-->
<table cellspacing='0'>
<thead></thead>
<tbody>
<tr>
        <td><b>Counter<b></td>
        <td id="COUNTER"></td>
</tr>
<tr>
        <td><b>Timestamp</b></td>
        <td id="WAKTU"></td>
</tr>
<tr>
        <td><b>Tegangan</b></td>
        <td id="DAYA"></td>
        </tr>
<tr>
        <td><b>Suhu</b></td>
        <td id="SUHU"></td>
</tr>
<tr>
        <td><b>Asap</b></td>
        <td id="ASAP"></td>
</tr>
<tr>
        <td><b>Api</b></td>
        <td id="API"></td>
</tr>
<tr>
        <td><b>Kelembaban</b></td>
        <td id="KELEMBABAN"></td>
```

```
                </tr>
                <tr>
                    <td><b>RSSI</b></td>
                    <td id="RSSI"></td>
                </tr>
                <tr>
                    <td><b>Rekap</b></td>
                    <td><a href="rekap.php" target='_blank'>lihat
detail</a></td>
                </tr>
                <tr>
                    <td><b>Status</b></td>
                    <td id="PESAN"></td>
                </tr>
                </tbody>
                </table>
            </div><!--end of content-->
            </div><!--end of right section-->
        </div><!--end of container-->
    </div><!--end of info-->
    <div id="footer">
    </div><!--end of footer-->
    </div><!--end of wrapper-->
</body>
</html>
```

Lampiran 2

File CSS

A. File index.css

```
*{margin:0; padding: 0; font-family: Arial, Helvetica, sans-serif;}

body{background-color: #ffffff;}

#wrapper{width: 1365px;  margin: 0 auto;  overflow: hidden; clear: both;}

#header{background: url(header.jpg) repeat-x; width: auto; height: 55px;}

#kiri{width: 50%;float: left;}

#kanan{  width: 50%;  float: right;}

#logo{text-align: left; color: #444; padding: 5px; padding-left: 17px;}

#waktu{margin-top: 10px; text-align: right; margin-right: 10px; color: #444; padding-right: 10px;}

#info{width: 100%; clear: both; overflow:hidden;}

#info{width: 100%; clear: both; overflow:hidden;}

.container{width: 1320px; margin: auto;}

#left_section{margin-top: 20px; width: 50%; float: left;}

#right_section{margin-top: 20px; width: 50%; float: right;}

#left_section h2, #right_section h2{color: #444;text-align: center;}
```

```
table{font-family: Arial, Helvetica, sans-serif; color: #666; text-shadow:1px 1px 0px #fff; background: #eaebec; border: #ccc 1px solid; margin-top: 10px;}

table tr { text-align: center; padding-left: 10px; padding-top: 3px; padding-bottom: 3px;}

table td:first-child {width: 175px; padding: 3px; border-top: 1px solid #ffffff; border-left: 1px solid #e0e0e0; border-bottom: 1px solid #e0e0e0; background: #ededed;}

table td {width: 470px; padding-top: 13px; padding-bottom: 12px; border-top: 1px solid #ffffff; border-bottom: 1px solid #e0e0e0; border-left: 1px solid #e0e0e0; background: #fafafa; background: -webkit-gradient(linear, left top, left bottom, from(#fbfbfb), to(#fafafa)); background: -moz-linear-gradient(top, #fbfbfb, #fafafa);}

table tr:last-child td {border-bottom: 0;}

table tr:last-child td:first-child {-moz-border-radius-bottomleft: 3px; -webkit-border-bottom-left-radius: 3px; border-bottom-left-radius: 3px;}

table tr:last-child td:last-child {-moz-border-radius-bottomright: 3px; -webkit-border-bottom-right-radius: 3px; border-bottom-right-radius: 3px;}

table tr:hover td {background: #f2f2f2; background: -webkit-gradient(linear, left top, left bottom, from(#f2f2f2), to(#f0f0f0)); background: -moz-linear-gradient(top, #f2f2f2, #f0f0f0);}

table a{color: #444;}

#footer{margin-top: 25px; border-top: 1px solid #aeb59c; padding: 10px;}
```

B. File rekap.css

```
*{margin:0; padding: 0; font-family: Arial, Helvetica, sans-serif;}

body{background-color: #ffffff;}
```

```
#wrapper{ width: 1365px; margin: 0 auto; overflow: hidden; clear: both;}

#header{background: url(header.jpg) repeat-x; width: auto; height: 55px;}

#logo{text-align: center; color: #444; padding: 10px; }

#info{width: 100%; clear: both; overflow:hidden; color: #444;}

table{font-family: Arial, Helvetica, sans-serif; color: #666; text-shadow:1px 1px 0px #fff; background: #eaebec; border: #ccc 1px solid; margin-top: 10px;}

table th{padding: 5px 20px; border-left: 1px solid #e0e0e0; border-bottom: 1px solid #e0e0e0; background: #ededed;}

table th:first-child{border-left:none;}

table tr {text-align: center; padding-left: 10px;}

table td:first-child {text-align: center; padding-left: 18px; border-left: 0;}

table td {padding: 5px 20px; border-top: 1px solid #ffffff; border-bottom: 1px solid #e0e0e0; border-left: 1px solid #e0e0e0; background: #fafafa; background: -webkit-gradient(linear, left top, left bottom, from(#fbfbfb), to(#fafafa)); background: -moz-linear-gradient(top, #fbfbfb, #fafafa); font-size: 12px;}

table tr:last-child td {border-bottom: 0;}

table tr:last-child td:first-child {-moz-border-radius-bottomleft: 3px; -webkit-border-bottom-left-radius: 3px; border-bottom-left-radius: 3px;}

table tr:last-child td:last-child {-moz-border-radius-bottomright: 3px; -webkit-border-bottom-right-radius: 3px; border-bottom-right-radius: 3px;}
```

```
table tr:hover td {background: #f2f2f2; background: -webkit-gradient(linear, left top, left bottom, from(#f2f2f2), to(#f0f0f0)); background: -moz-linear-gradient(top, #f2f2f2, #f0f0f0);}

#footer{margin-top: 25px; border-top: 1px solid #aeb59c; padding: 10px;}
```

Lampiran 3

Program Javascript

A. File waktu.js

```
var myVar = setInterval(myTimer, 1000);

function myTimer() {
 var d = new Date();
  var tanggal = d.toLocaleDateString();
  var waktu = d.toLocaleTimeString();

  document.getElementById("tanggal").innerHTML = tanggal;
  document.getElementById("jam").innerHTML = waktu;}
```

B. File map.js

```
var mymap = L.map('map').setView([ 112.57067084127601, -7.849639633549004], 15);


const tileUrl = 'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png';
const attribution =
      '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors';
const tiles = L.tileLayer(tileUrl, { attribution });


tiles.addTo(mymap);


var iconA = L.icon({
        iconUrl: 'leaf-red.png',
        shadowUrl: 'leaf-shadow.png',
        iconSize:     [38, 95],
    shadowSize:   [50, 64],
     iconAnchor:   [22, 94],
    shadowAnchor: [4, 62],
     popupAnchor:  [-3, -76]
});
```

```
var markerA = new L.marker([-7.85, 112.57], {icon: iconA}).addTo(mymap);
async function lokasia() {
        const response = await fetch(url1);
const data = await response.json();
const { latitude, longitude } = data;
markerA.setLatLng([latitude, longitude]);
markerA.bindPopup("<center><b>Lokasi:</b><br></center>" +
markerA.getLatLng().toString()).openPopup();
}

lokasia();
setInterval(lokasia, 5000);
```

Lampiran 4

Program PHP

A. File connect.php

```
<?php
        $connect = mysqli_connect('localhost', 'webcdpco_merry', '***', 'webcdpco_alat');
        if(mysqli_connect_error()){
                echo 'Gagal melakukan koneksi ke Basis data : '.mysqli_connect_error();
        }else{}?>
```

B. File COUNTER.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT COUNTER FROM alata ORDER BY ID DESC
LIMIT 1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["COUNTER"]."";}}?>
```

C. File WAKTU.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT WAKTU FROM alata ORDER BY ID DESC
LIMIT 1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["WAKTU"]."<br>";}}?>
```

D. File DAYA.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT DAYA FROM alata ORDER BY ID DESC LIMIT
1");
```

```
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["DAYA"]." volt";}}?>
```

E. File SUHU.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT SUHU FROM alata ORDER BY ID DESC LIMIT
1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["SUHU"]."&degC<br>";}}?>
```

F. File ASAP.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT ASAP FROM alata ORDER BY ID DESC LIMIT
1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["ASAP"]." ppm";}}?>
```

G. File API.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT API FROM alata ORDER BY ID DESC LIMIT
1");
$api;
if ($result->num_rows > 0){
while($row = $result->fetch_assoc()){
extract($row);
foreach ($row as $key => $value) {
$api=$value;
if ($api!=1) {echo "<img src='apinyala.png' width='20' height='30' />";}
```

```
else{echo "<img src='apipadam.png' width='20' height='30' />";}}}}?>
```

H. File KELEMBABAN.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT KELEMBABAN FROM alata ORDER BY ID
DESC LIMIT 1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["KELEMBABAN"]." RH";}}?>
```

I. File RSSI.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT RSSI FROM alata ORDER BY ID DESC LIMIT
1");
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                echo $row["RSSI"]."<br>";}}?>
```

J. File Rekap.php

```
<!DOCTYPE html>
<html>
<head>
<title>Rekap Data</title>
<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="rekap.css">
</head>
<body>
<div id="wrapper">
        <div id="header">
        <h2 id="logo">REKAP DATA</h2>
        </div>
<div id="info">
<center>
```

```
<?php include 'connect.php'; ?>
<form method="get">
<br>
<label>Pilih Tanggal</label>
<input type="date" name="tanggal">
<span>s/d</span>
<input type="date" name="tanggal2">
<input type="submit" value="Filter">
</form>
<table cellspacing='0'>
<tr>
<th>No</th>
<th>Timestamp</th>
<th>Counter</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Tegangan</th>
<th>RSSI</th>
<th>Suhu</th>
<th>Asap</th>
<th>Api</th>
<th>Kelembaban</th>
</tr>
<?php
$no = 1;
if(isset($_GET['tanggal'], $_GET['tanggal2'])){
$tgl = $_GET['tanggal'];
$tgl2 = $_GET['tanggal2'];
$sql = mysqli_query($connect,"select * from alata where WAKTU>='$tgl 00:00:00' and
WAKTU<='$tgl2 23:59:59' order by ID desc");
}else{
$sql = mysqli_query($connect,"select * from alata order by ID desc");}
while($data = mysqli_fetch_array($sql)){
?>
```

```
<tr>
<td><?php echo $no++; ?></td>
<td><?php echo $data['WAKTU']; ?></td>
<td><?php echo $data['COUNTER']; ?></td>
<td><?php echo $data['LAT']; ?></td>
<td><?php echo $data['LNG']; ?></td>
<td><?php echo $data['DAYA']." volt";  ?></td>
<td><?php echo $data['RSSI']; ?></td>
<td><?php echo $data['SUHU']."&degC"; ?></td>
<td><?php echo $data['ASAP']." ppm"; ?></td>
<td><?php echo $data['API']; ?></td>
<td><?php echo $data['KELEMBABAN']." RH"; ?></td>
</tr>
<?php } ?>
</table>
</center>
</div><!-- end of id info-->
<div id="footer">
</div>
</div><!-- end of id wrapper-->
</body>
</html>
```

K. File PESAN.php

```
<?php
include "connect.php";
$result = $connect->query("SELECT SUHU FROM alata ORDER BY ID DESC LIMIT
1");
$result2 = $connect->query("SELECT ASAP FROM alata ORDER BY ID DESC LIMIT
1");
$result3 = $connect->query("SELECT API FROM alata ORDER BY ID DESC LIMIT
1");
$result4 = $connect->query("SELECT KELEMBABAN FROM alata ORDER BY ID
DESC LIMIT 1");
```

```
$suhu;
$asap;
$api;
$kelembaban;
if ($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
                extract($row);
                foreach ($row as $key => $value) {
                        $suhu=$value;}}}
if ($result2->num_rows > 0){
        while($row = $result2->fetch_assoc()){
                extract($row);
                foreach ($row as $key => $value) {
                        $asap=$value;}}}

if ($result3->num_rows > 0){
        while($row = $result3->fetch_assoc()){
                extract($row);
                foreach ($row as $key => $value) {
                        $api=$value;}}}

if ($result4->num_rows > 0){
        while($row = $result4->fetch_assoc()){
                extract($row);
                foreach ($row as $key => $value) {
                        $kelembaban=$value;}}}
if (($suhu>35) && ($asap>500)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}
elseif (($suhu>35) && ($api!=1)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}
```

```
elseif (($suhu>35) && ($kelembaban<38)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}
elseif (($asap>500) && ($api!=1)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}
elseif (($asap>500) && ($kelembaban<38)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}
elseif (($api!=0) && ($kelembaban<38)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}

elseif (($asap>500) && ($api!=1) && ($suhu>35)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}

elseif (($asap>500) && ($kelembaban<38) && ($suhu>35)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}

elseif (($asap>500) && ($kelembaban<38) && ($$api!=1)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}

elseif (($suhu>35) && ($kelembaban<38) && ($$api!=1)) {
        echo "<h4 style='color:red;'>BAHAYA!</h4>";}

else {
        echo "<span style='color:green;'><b>Aman</b></span>";}
?>
```

L. File kirimdata.php

```
<?php
 include "connect.php";
If(isset($_GET["parametersatu"])){
 $suhu=$_GET["parametersatu"];
}else{ $suhu = "404";}
```

```
If(isset($_GET["parameterdua"])){
$asap=$_GET["parameterdua"]; //tingkatasap
}else{ $asap = "404";}

If(isset($_GET["parametertiga"])){
$api=$_GET["parametertiga"]; //tingkatapi
}else{ $api = "404";}

If(isset($_GET["parameterempat"])){
$kelembaban=$_GET["parameterempat"]; //kelembaban
}else{ $kelembaban = "404";}

If(isset($_GET["parameterlima"])){
$rssi=$_GET["parameterlima"]; //rssi
}else{ $rssi = "404";}

If(isset($_GET["parameterenam"])){
$latitude=$_GET["parameterenam"]; //latitude
}else{ $latitude = "404";}

If(isset($_GET["parametertujuh"])){
$longitude=$_GET["parametertujuh"];//longitude
}else{ $longitude = "404";}

If(isset($_GET["parametersembilan"])){
$daya=$_GET["parametersembilan"]; //tegangan
}else{ $daya = "404";}
If(isset($_GET["parametersepuluh"])){
$counter=$_GET["parametersepuluh"]; //tegangan
}else{ $counter = "404";}
```

```
//auto increment = 1 / mengembalikan ID menjadi 1 apabila dikosongkan
mysqli_query($connect, "ALTER TABLE alata AUTO_INCREMENT=1");
//$simpan = mysqli_query($connect, "INSERT INTO alata(SUHUA, ASAPA, APIA,
KELEMBABANA, RSSIA, GPSALa, GPSALo, DAYAA, COUNTERA)
VALUES('$suhu', '$asap', '$api', '$kelembaban', '$rssi', '$latitude', '$longitude', '$daya',
'$counter')");
$simpan = mysqli_query($connect, "INSERT INTO alata(LAT, LNG, DAYA, RSSI,
SUHU, ASAP, API, KELEMBABAN, COUNTER) VALUES('$latitude', '$longitude',
'$daya', '$rssi', '$suhu', '$asap', '$api', '$kelembaban', '$counter')");
//uji simpan untuk memberi respon
if($simpan)
        echo "Berhasil dikirim";
else
        echo "Gagal dikirim";
?>
```

M. File urlgps.php

```
<?php
include'connect.php';

$sql = "SELECT LAT, LNG FROM alata ORDER BY ID DESC LIMIT 1";
$result = $connect->query($sql);
$json = [];
if ($result->num_rows > 0) {
  // output data of each row
  while($row = $result->fetch_assoc()) {
  $json = [
    'geometry'  => [
            'type'       => 'Point',
          'coordinates' => array(doubleval($row['LNG']), doubleval($row['LAT']))
        ],
    'type'      => 'Feature',
     'properties'=> ['nama' => 'Lokasi A'],
    'latitude' => doubleval($row['LAT']),
```

```
        'longitude' => doubleval($row['LNG'])
        ]; }}
    header('Content-Type: application/json');
    echo json_encode($json);?>
```

Lampiran 5

Program ESP32

```
#include <WiFi.h>
#include <HTTPClient.h>
const char* ssid = "997025";
const char* password = "997025123";
const char* host = "webcdp.com";
#define PIN_LED 2
int suhu = 24.20;
int asap = 149;
int api = 1;
int kelembaban = 61.74;
int rssi = -86;
float latitude = -7.9491631153966935 ;
float longitude = 112.61305699908989 ;
float daya = 5.02;
int Counter = 1;

int suhu1 = 24.40;
int asap1 = 206;
int api1 = 1;
int kelembaban1 = 61.70;
int rssi1 = -87;
float latitude1 = -7.9491631153966935;
float longitude1 = 112.61305699908989;
float daya1 = 5.02;
int suhu2 = 37;
int asap2 = 401;
```

```
int api2 = 0;
int kelembaban2 = 59.20;
int rssi2 = -81;
float latitude2 = -7.9491631153966935;
float longitude2 = 112.61305699908989;
float daya2 = 5.02;
int suhu3 = 37;
int asap3 = 406;
int api3 = 0;
int kelembaban3 = 52;
int rssi3 = -88;
float latitude3 = -7.9491631153966935;
float longitude3 = 112.61305699908989;
float daya3 = 5.02;

int suhu4 = 33;
int asap4 = 178;
int api4 = 1;
int kelembaban4 = 56;
int rssi4 = -86;
float latitude4 = -7.9491631153966935;
float longitude4 = 112.61305699908989;
float daya4 = 5.02;

void setup(){
 Serial.begin(9600);
 pinMode (PIN_LED, OUTPUT);
 WiFi.begin(ssid, password);
 while(WiFi.status() !=WL_CONNECTED)
 {
  //led mati
  digitalWrite(PIN_LED, LOW);
```

```
  //terus mencoba koneksi
  Serial.print(".");
  delay(500);
 }
 //jika berhasil koneksi, tampilkan pesan
 digitalWrite(PIN_LED, HIGH);
 Serial.println("WiFi Connected");
}
void loop(){
 String lat = String (latitude, 8);
 String lng = String (longitude, 8);
 String lat1 = String (latitude1, 8);
 String lng1 = String (longitude1, 8);
 String lat2 = String (latitude2, 8);
 String lng2 = String (longitude2, 8);
 String lat3 = String (latitude3, 8);
 String lng3 = String (longitude3, 8);
 String lat4 = String (latitude4, 8);
 String lng4 = String (longitude4, 8);

 Serial.println("Counter : "  + String(Counter));
 Serial.println("Suhu : "  + String(suhu));
 Serial.println("Asap : " + String(asap));
 Serial.println("Api : "+ String(api));
 Serial.println("Kelembaban : " + String(kelembaban));
 Serial.println("RSSI : " + String(rssi));
 Serial.println("Latitude : " + String(lat));
 Serial.println("Longitude : " + String(lng));
 Serial.println("Daya : " + String(daya));

 //cek koneksi ke server
 WiFiClient client;
```

```
const int httpPort = 80;
if(!client.connect(host, httpPort))
{
  Serial.println("Connection Failed");
  return;
}
//ketika berhasil terkoneksi
String Link;
HTTPClient http;
Link = "http://" + String(host) + "/kirimdata.php?parametersatu=" + String(suhu) +
"&parameterdua=" + String(asap) + "&parametertiga=" + String(api) +
"&parameterempat=" + String(kelembaban) + "&parameterlima=" + String(rssi) +
"&parameterenam=" + String(lat) + "&parametertujuh=" + String(lng) +
"&parametersembilan=" + String(daya) + "&parametersepuluh=" + String(Counter);
http.begin(Link);
http.GET(); //mode GET
String respon = http.getString();
Serial.println(respon);
Serial.println("----------------------");
http.end();
delay(5000);
Counter++;

Serial.println("Counter : "  + String(Counter));
Serial.println("Suhu : "  + String(suhu1));
Serial.println("Asap : " + String(asap1));
Serial.println("Api : "+ String(api1));
Serial.println("Kelembaban : " + String(kelembaban1));
Serial.println("RSSI : " + String(rssi1));
Serial.println("Latitude : " + String(lat1));
Serial.println("Longitude : " + String(lng1));
Serial.println("Daya : " + String(daya1));
```

```
 if(!client.connect(host, httpPort))
 {
  Serial.println("Connection Failed");
  return; //coba koneksi lagi
 }
 String Link1;
 Link1 = "http://" + String(host) + "/kirimdata.php?parametersatu=" + String(suhu1) +
"&parameterdua=" + String(asap1) + "&parametertiga=" + String(api1) +
"&parameterempat=" + String(kelembaban1) + "&parameterlima=" + String(rssi1) +
"&parameterenam=" + String(lat1) + "&parametertujuh=" + String(lng1) +
"&parametersembilan=" + String(daya1) + "&parametersepuluh=" + String(Counter);
 http.begin(Link1);
 http.GET(); //mode GET
 Serial.println(respon);
 Serial.println("----------------------");
 http.end();
 delay(5000);
 Counter++;

 Serial.println("Counter : "  + String(Counter));
 Serial.println("Suhu : "  + String(suhu2));
 Serial.println("Asap : " + String(asap2));
 Serial.println("Api : "+ String(api2));
 Serial.println("Kelembaban : " + String(kelembaban2));
 Serial.println("RSSI : " + String(rssi2));
 Serial.println("Latitude : " + String(lat2));
 Serial.println("Longitude : " + String(lng2));
 Serial.println("Daya : " + String(daya2));
 if(!client.connect(host, httpPort))
 {
  Serial.println("Connection Failed");
  return; //coba koneksi lagi
```

```
}
String Link2;
Link2 = "http://" + String(host) + "/kirimdata.php?parametersatu=" + String(suhu2) +
"&parameterdua=" + String(asap2) + "&parametertiga=" + String(api2) +
"&parameterempat=" + String(kelembaban2) + "&parameterlima=" + String(rssi2) +
"&parameterenam=" + String(lat2) + "&parametertujuh=" + String(lng2) +
"&parametersembilan=" + String(daya2) + "&parametersepuluh=" + String(Counter);
http.begin(Link2);
http.GET(); //mode GET
Serial.println(respon);
Serial.println("-----------------------");
http.end();
delay(5000);
Counter++;

Serial.println("Counter : "  + String(Counter));
Serial.println("Suhu : "  + String(suhu3));
Serial.println("Asap : " + String(asap3));
Serial.println("Api : "+ String(api3));
Serial.println("Kelembaban : " + String(kelembaban3));
Serial.println("RSSI : " + String(rssi3));
Serial.println("Latitude : " + String(lat3));
Serial.println("Longitude : " + String(lng3));
Serial.println("Daya : " + String(daya3));

if(!client.connect(host, httpPort))
{
  Serial.println("Connection Failed");
  return; //coba koneksi lagi
}
String Link3;
```

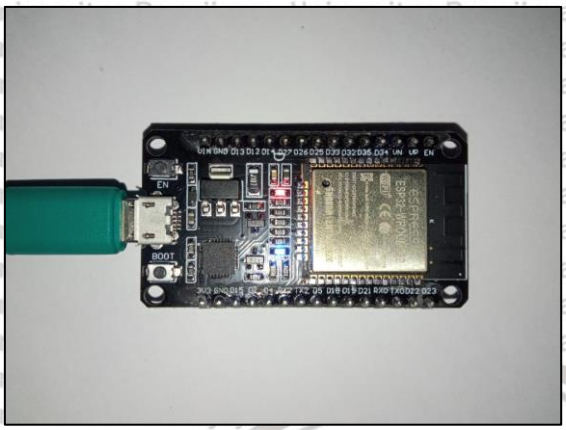
```
  Link3 = "http://" + String(host) + "/kirimdata.php?parametersatu=" + String(suhu3) +
"&parameterdua=" + String(asap3) + "&parametertiga=" + String(api3) +
"&parameterempat=" + String(kelembaban3) + "&parameterlima=" + String(rssi3) +
"&parameterenam=" + String(lat3) + "&parametertujuh=" + String(lng3) +
"&parametersembilan=" + String(daya3) + "&parametersepuluh=" + String(Counter);
  http.begin(Link3);
  http.GET(); //mode GET
  Serial.println(respon);
  Serial.println("------------------------");
  http.end();
  delay(5000);
  Counter++;

  Serial.println("Counter : "  + String(Counter));
  Serial.println("Suhu : "  + String(suhu4));
  Serial.println("Asap : " + String(asap4));
  Serial.println("Api : "+ String(api4));
  Serial.println("Kelembaban : " + String(kelembaban4));
  Serial.println("RSSI : " + String(rssi4));
  Serial.println("Latitude : " + String(lat4));
  Serial.println("Longitude : " + String(lng4));
  Serial.println("Daya : " + String(daya4));

  if(!client.connect(host, httpPort))
  {
   Serial.println("Connection Failed");
   return; //coba koneksi lagi
  }
  String Link4;
  Link4 = "http://" + String(host) + "/kirimdata.php?parametersatu=" + String(suhu4) +
"&parameterdua=" + String(asap4) + "&parametertiga=" + String(api4) +
"&parameterempat=" + String(kelembaban4) + "&parameterlima=" + String(rssi4) +
```

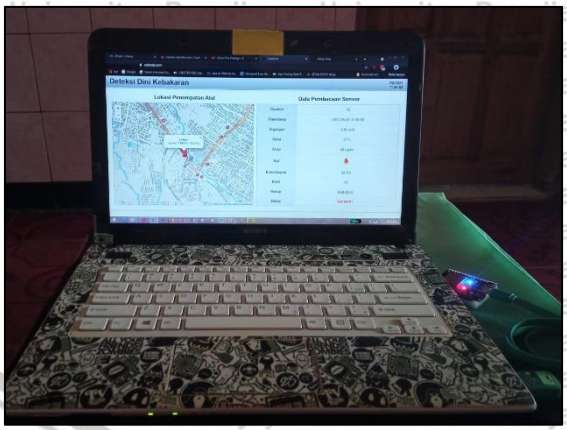
```
"&parameterenam=" + String(lat4) + "&parametertujuh=" + String(lng4) +
"&parametersembilan=" + String(daya4) + "&parametersepuluh=" + String(Counter);
http.begin(Link4);
http.GET(); //mode GET
Serial.println(respon);
Serial.println("-----------------------");
http.end();
delay(5000);
Counter++;
}
```

Lampiran 6

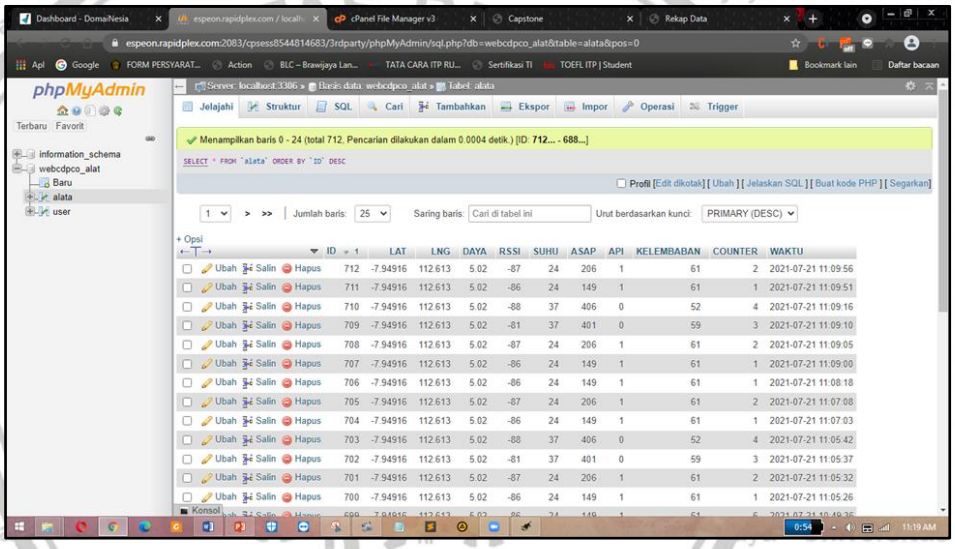
Foto Dokumentasi

A. Simulasi





B. Basis data



Lampiran 7

***Datasheet* ESP32**

1 Overview

ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC ultra-low-power 40 nm technology. It is designed to achieve the best power and RF performance, showing robustness, versatility and reliability in a wide variety of applications and power scenarios.

The ESP32 series of chips includes ESP32-D0WD-V3, ESP32-D0WDQ6-V3, ESP32-D0WD, ESP32-D0WDQ6, ESP32-D2WD, ESP32-S0WD, and ESP32-U4WDH, among which, ESP32-D0WD-V3, ESP32-D0WDQ6-V3, and ESP32-U4WDH are based on ECO V3 wafer.

For details on part numbers and ordering information, please refer to Section 7.

For details on ECO V3 instructions, please refer to *ESP32 ECO V3 User Guide*.

1.1 Featured Solutions

1.1.1 Ultra-Low-Power Solution

ESP32 is designed for mobile, wearable electronics, and Internet-of-Things (IoT) applications. It features all the state-of-the-art characteristics of low-power chips, including fine-grained clock gating, multiple power modes, and dynamic power scaling. For instance, in a low-power IoT sensor hub application scenario, ESP32 is woken up periodically and only when a specified condition is detected. Low-duty cycle is used to minimize the amount of energy that the chip expends. The output of the power amplifier is also adjustable, thus contributing to an optimal trade-off between communication range, data rate and power consumption.

Note:
For more information, refer to Section 3.7 *RTC and Low-Power Management*.

1.1.2 Complete Integration Solution

ESP32 is a highly-integrated solution for Wi-Fi-and-Bluetooth IoT applications, with around 20 external components. ESP32 integrates an antenna switch, RF balun, power amplifier, low-noise receive amplifier, filters, and power management modules. As such, the entire solution occupies minimal Printed Circuit Board (PCB) area.

ESP32 uses CMOS for single-chip fully-integrated radio and baseband, while also integrating advanced calibration circuitries that allow the solution to remove external circuit imperfections or adjust to changes in external conditions. As such, the mass production of ESP32 solutions does not require expensive and specialized Wi-Fi testing equipment.

1.2 Wi-Fi Key Features

- 802.11 b/g/n
- 802.11 n (2.4 GHz), up to 150 Mbps
- WMM
- TX/RX A-MPDU, RX A-MSDU
- Immediate Block ACK

- Defragmentation
- Automatic Beacon monitoring (hardware TSF)
- 4 × virtual Wi-Fi interfaces
- Simultaneous support for Infrastructure Station, SoftAP, and Promiscuous modes
  Note that when ESP32 is in Station mode, performing a scan, the SoftAP channel will be changed.
- Antenna diversity

Note:
For more information, please refer to Section 3.5 *Wi-Fi*.

1.3 BT Key Features

- Compliant with Bluetooth v4.2 BR/EDR and BLE specifications
- Class-1, class-2 and class-3 transmitter without external power amplifier
- Enhanced Power Control
- +12 dBm transmitting power
- NZIF receiver with –94 dBm BLE sensitivity
- Adaptive Frequency Hopping (AFH)
- Standard HCI based on SDIO/SPI/UART
- High-speed UART HCI, up to 4 Mbps
- Bluetooth 4.2 BR/EDR BLE dual mode controller
- Synchronous Connection-Oriented/Extended (SCO/eSCO)
- CVSD and SBC for audio codec
- Bluetooth Piconet and Scatternet
- Multi-connections in Classic BT and BLE
- Simultaneous advertising and scanning

1.4 MCU and Advanced Features

1.4.1 CPU and Memory

- Xtensa® single-/dual-core 32-bit LX6 microprocessor(s), up to 600 MIPS (200 MIPS for ESP32-S0WD/ESP32-U4WDH, 400 MIPS for ESP32-D2WD)
- 448 KB ROM
- 520 KB SRAM
- 16 KB SRAM in RTC
- QSPI supports multiple flash/SRAM chips

1.4.2 Clocks and Timers

- Internal 8 MHz oscillator with calibration
- Internal RC oscillator with calibration
- External 2 MHz ~ 60 MHz crystal oscillator (40 MHz only for Wi-Fi/BT functionality)
- External 32 kHz crystal oscillator for RTC with calibration
- Two timer groups, including 2 × 64-bit timers and 1 × main watchdog in each group
- One RTC timer
- RTC watchdog

1.4.3 Advanced Peripheral Interfaces

- 34 × programmable GPIOs
- 12-bit SAR ADC up to 18 channels
- 2 × 8-bit DAC
- 10 × touch sensors
- 4 × SPI
- 2 × I²S
- 2 × I²C
- 3 × UART
- 1 host (SD/eMMC/SDIO)
- 1 slave (SDIO/SPI)
- Ethernet MAC interface with dedicated DMA and IEEE 1588 support
- Two-Wire Automotive Interface (TWAI®, compatible with ISO11898-1)
- IR (TX/RX)
- Motor PWM
- LED PWM up to 16 channels
- Hall sensor

1.4.4 Security

- Secure boot
- Flash encryption
- 1024-bit OTP, up to 768-bit for customers
- Cryptographic hardware acceleration:
  - AES
  - Hash (SHA-2)

  - RSA
  - ECC
  - Random Number Generator (RNG)

1.5 Applications (A Non-exhaustive List)

- Generic Low-power IoT Sensor Hub
- Generic Low-power IoT Data Loggers
- Cameras for Video Streaming
- Over-the-top (OTT) Devices
- Speech Recognition
- Image Recognition
- Mesh Network
- Home Automation
  - Light control
  - Smart plugs
  - Smart door locks
- Smart Building
  - Smart lighting
  - Energy monitoring
- Industrial Automation
  - Industrial wireless control
  - Industrial robotics
- Smart Agriculture
  - Smart greenhouses
  - Smart irrigation
  - Agriculture robotics
- Audio Applications
  - Internet music players
  - Live streaming devices
  - Internet radio players
  - Audio headsets
- Health Care Applications
  - Health monitoring
  - Baby monitors
- Wi-Fi-enabled Toys
  - Remote control toys
  - Proximity sensing toys
  - Educational toys
- Wearable Electronics
  - Smart watches
  - Smart bracelets
- Retail & Catering Applications
  - POS machines
  - Service robots

1.6 Block Diagram

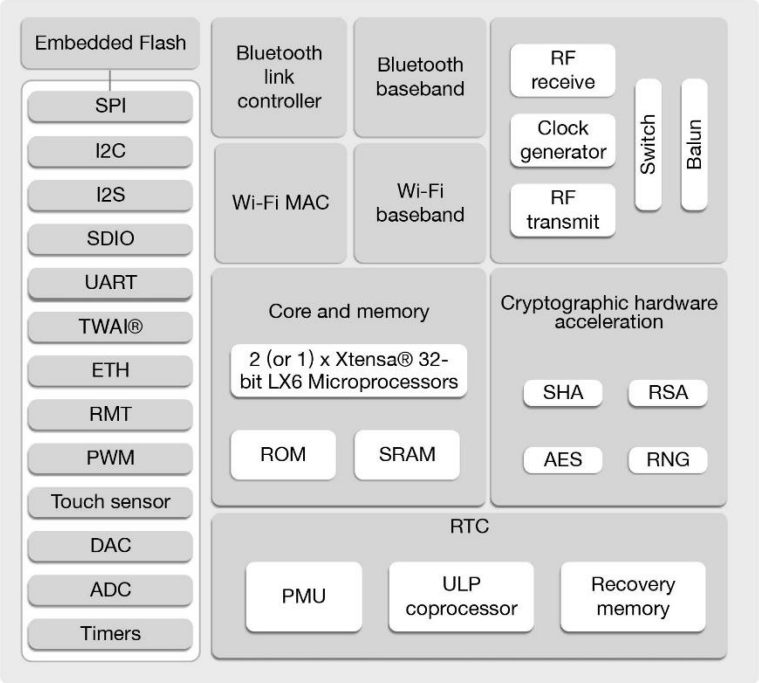


Figure 1: Functional Block Diagram

Note:
Products in the ESP32 series differ from each other in terms of their support for embedded flash and the number of CPUs they have. For details, please refer to Section 7 *Part Number and Ordering Information*.

2 Pin Definitions

2.1 Pin Layout

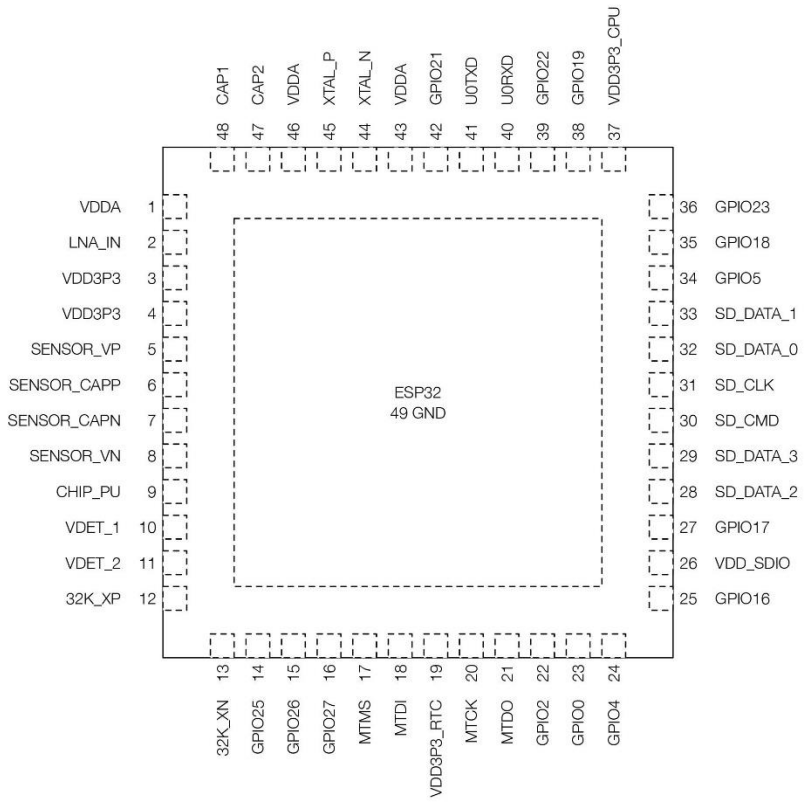


Figure 2: ESP32 Pin Layout (QFN 6*6, Top View)

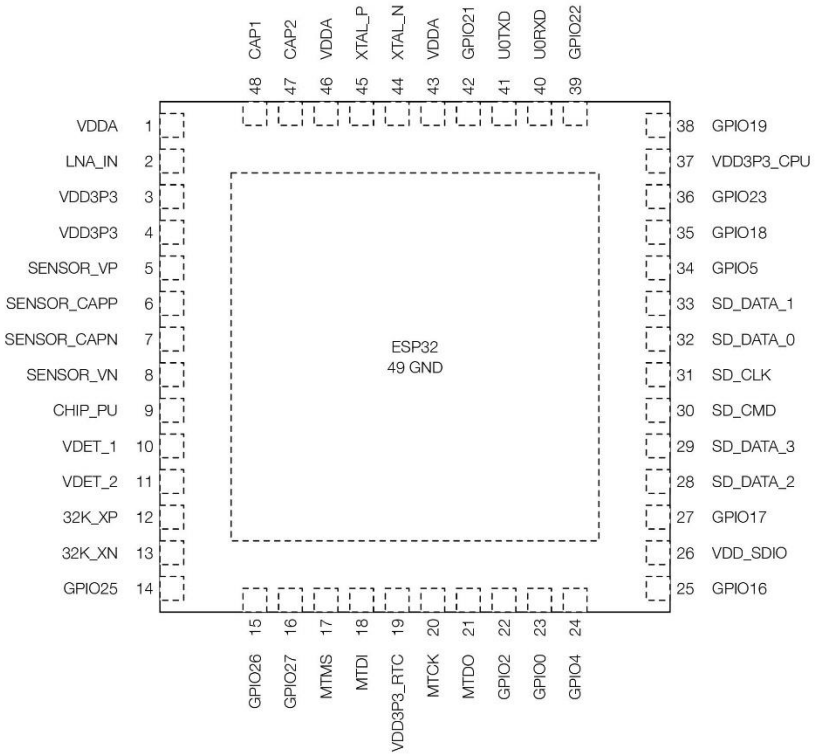


Figure 3: ESP32 Pin Layout (QFN 5*5, Top View)

Note:
For details on ESP32's part numbers and the corresponding packaging, please refer to Section 7 *Part Number and Ordering Information*.

2.2 Pin Description

Table 1: Pin Description

Name	No.	Type	Function
Analog			
VDDA	1	P	Analog power supply (2.3 V ~ 3.6 V)
LNA_IN	2	I/O	RF input and output
VDD3P3	3	P	Analog power supply (2.3 V ~ 3.6 V)
VDD3P3	4	P	Analog power supply (2.3 V ~ 3.6 V)
VDD3P3_RTC			
SENSOR_VP	5	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_CAPP	6	I	GPIO37, ADC1_CH1, RTC_GPIO1
SENSOR_CAPN	7	I	GPIO38, ADC1_CH2, RTC_GPIO2
SENSOR_VN	8	I	GPIO39, ADC1_CH3, RTC_GPIO3
CHIP_PU	9	I	High: On; enables the chip Low: Off; the chip powers off Note: Do not leave the CHIP_PU pin floating.
VDET_1	10	I	GPIO34, ADC1_CH6, RTC_GPIO4
VDET_2	11	I	GPIO35, ADC1_CH7, RTC_GPIO5
32K_XP	12	I/O	GPIO32, ADC1_CH4, RTC_GPIO9, TOUCH9, 32K_XP (32.768 kHz crystal oscillator input)
32K_XN	13	I/O	GPIO33, ADC1_CH5, RTC_GPIO8, TOUCH8, 32K_XN (32.768 kHz crystal oscillator output)
GPIO25	14	I/O	GPIO25, ADC2_CH8, RTC_GPIO6, DAC_1, EMAC_RXD0
GPIO26	15	I/O	GPIO26, ADC2_CH9, RTC_GPIO7, DAC_2, EMAC_RXD1
GPIO27	16	I/O	GPIO27, ADC2_CH7, RTC_GPIO17, TOUCH7, EMAC_RX_DV
MTMS	17	I/O	GPIO14, ADC2_CH6, RTC_GPIO16, TOUCH6, EMAC_TXD2, HSPICLK, HS2_CLK, SD_CLK, MTMS
MTDI	18	I/O	GPIO12, ADC2_CH5, RTC_GPIO15, TOUCH5, EMAC_TXD3, HSPIQ, HS2_DATA2, SD_DATA2, MTDI
VDD3P3_RTC	19	P	Input power supply for RTC IO (2.3 V ~ 3.6 V)
MTCK	20	I/O	GPIO13, ADC2_CH4, RTC_GPIO14, TOUCH4, EMAC_RX_ER, HSPID, HS2_DATA3, SD_DATA3, MTCK
MTDO	21	I/O	GPIO15, ADC2_CH3, RTC_GPIO13, TOUCH3, EMAC_RXD3, HSPICS0, HS2_CMD, SD_CMD, MTDO

Name	No.	Type	Function
GPIO2	22	I/O	GPIO2, ADC2_CH2, RTC_GPIO12, TOUCH2, HSPIWP, HS2_DATA0, SD_DATA0
GPIO0	23	I/O	GPIO0, ADC2_CH1, RTC_GPIO11, TOUCH1, EMAC_TX_CLK, CLK_OUT1,
GPIO4	24	I/O	GPIO4, ADC2_CH0, RTC_GPIO10, TOUCH0, EMAC_TX_ER, HSPIHD, HS2_DATA1, SD_DATA1
VDD_SDIO			
GPIO16	25	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
VDD_SDIO	26	P	Output power supply: 1.8 V or the same voltage as VDD3P3_RTC
GPIO17	27	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
SD_DATA_2	28	I/O	GPIO9, HS1_DATA2, U1RXD, SD_DATA2, SPIHD
SD_DATA_3	29	I/O	GPIO10, HS1_DATA3, U1TXD, SD_DATA3, SPIWP
SD_CMD	30	I/O	GPIO11, HS1_CMD, U1RTS, SD_CMD, SPICS0
SD_CLK	31	I/O	GPIO6, HS1_CLK, U1CTS, SD_CLK, SPICLK
SD_DATA_0	32	I/O	GPIO7, HS1_DATA0, U2RTS, SD_DATA0, SPIQ
SD_DATA_1	33	I/O	GPIO8, HS1_DATA1, U2CTS, SD_DATA1, SPID
VDD3P3_CPU			
GPIO5	34	I/O	GPIO5, HS1_DATA6, VSPICS0, EMAC_RX_CLK
GPIO18	35	I/O	GPIO18, HS1_DATA7, VSPICLK
GPIO23	36	I/O	GPIO23, HS1_STROBE, VSPID
VDD3P3_CPU	37	P	Input power supply for CPU IO (1.8 V ~ 3.6 V)
GPIO19	38	I/O	GPIO19, U0CTS, VSPIQ, EMAC_TXD0
GPIO22	39	I/O	GPIO22, U0RTS, VSPIWP, EMAC_TXD1
U0RXD	40	I/O	GPIO3, U0RXD, CLK_OUT2
U0TXD	41	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
GPIO21	42	I/O	GPIO21, VSPIHD, EMAC_TX_EN
Analog			
VDDA	43	P	Analog power supply (2.3 V ~ 3.6 V)
XTAL_N	44	O	External crystal output
XTAL_P	45	I	External crystal input
VDDA	46	P	Analog power supply (2.3 V ~ 3.6 V)
CAP2	47	I	Connects to a 3.3 nF (10%) capacitor and 20 kΩ resistor in parallel to CAP1

Name	No.	Type	Function
CAP1	48	I	Connects to a 10 nF series capacitor to ground
GND	49	P	Ground

Note:

- The pin-pin mapping between ESP32-D2WD/ESP32-U4WDH and the embedded flash is as follows: GPIO16 = CS#, GPIO17 = IO1/DO, SD_CMD = IO3/HOLD#, SD_CLK = CLK, SD_DATA_0 = IO2/WP#, SD_DATA_1 = IO0/DI. The pins used for embedded flash are not recommended for other uses.
- In most cases, the data port connection between ESP32 series of chips other than ESP32-D2WD/ESP32-U4WDH and external flash is as follows: SD_DATA0/SPIQ = IO1/DO, SD_DATA1/SPID = IO0/DI, SD_DATA2/SPIHD = IO3/HOLD#, SD_DATA3/SPIWP = IO2/WP#.
- For a quick reference guide to using the IO_MUX, Ethernet MAC, and GIPO Matrix pins of ESP32, please refer to Appendix *ESP32 Pin Lists*.