SISTEM PEMANTAUAN KUALITAS UDARA BERBASIS *INTERNET OF THINGS* (IOT) MENGGUNAKAN NODEMCU DENGAN *INTERFACE WEBSITE*

SKRIPSI

AHMAD SYAIFULLOH

NPM 20670116

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK DAN INFORMATIKA

UNIVERSITAS PGRI SEMARANG

2024

SISTEM PEMANTAUAN KUALITAS UDARA BERBASIS *INTERNET OF THINGS* (IOT) MENGGUNAKAN NODEMCU DENGAN *INTERFACE WEBSITE*

Diajukan kepada Fakultas Teknik dan Informatika
Universitas PGRI Semarang untuk Penyusunan Skripsi

SKRIPSI

AHMAD SYAIFULLOH

NPM 20670116

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK DAN INFORMATIKA

UNIVERSITAS PGRI SEMARANG

2024

LEMBAR PERSETUJUAN

SISTEM PEMANTAUAN KUALITAS UDARA BERBASIS *INTERNET OF THINGS* (IOT) MENGGUNAKAN NODEMCU DENGAN *INTERFACE WEBSITE*

Disusun dan diajukan oleh

AHMAD SYAIFULLOH

NPM 20670116

telah disetujui oleh pembimbing untuk dilanjutkan

dihadapan Dewan Penguji

Semarang, 7 Agustus 2024

Pembimbing Utama	**Pembimbing Pendamping**
Mega Novita, S.Si., M.Si., M.Nat.Sc., Ph.D.	**Ir. Agung Handayanto, M.Kom.**
NIDN. 0615118801	**NIDN. 0019116202**

SKRIPSI

SISTEM PEMANTAUAN KUALITAS UDARA BERBASIS *INTERNET OF THINGS* (IOT) MENGGUNAKAN NODEMCU DENGAN *INTERFACE WEBSITE*

Disusun dan diajukan oleh

AHMAD SYAIFULLOH

NPM 20670116

Telah dipertahankan dihadapan Dewan Penguji pada tanggal 19 Agustus 2024 dan Dinyatakan telah memenuhi syarat Dewan Penguji

Ketua	Sekretaris
Ibnu Toto Husodo, S.T., M.T.	**Bambang Agus Herlambang, S.Kom., M.Kom.**
NIDN. 0602126902	**NIDN. 0601088201**

Penguji I	Penguji II
Mega Novita, Ph.D.	**Ir. Agung Handayanto, M.Kom.**
NIDN. 0615118801	**NIDN. 0019116202**

Penguji III

Nugroho Dwi Saputro, S.Kom., M.Kom.

NIDN. 0623058802

MOTTO DAN PERSEMBAHAN

Moto:

1. “Lebih baik mencoba lalu gagal daripada tidak melakukan sama sekali. Karena sejatinya kegagalan memungkinkan adanya perubahan yang lebih baik.” (Ahmad Syaifulloh)
2. “Man Jadda Wajada (Barang siapa yang bersungguh-sungguh, maka akan akan berhasil).”
3. **“**Maka, sesungguhnya beserta kesulitan ada kemudahan. Sesungguhnya beserta kesulitan ada kemudahan.” (QS. Al-Insyirah: 5-6)

Persembahan:

Saya persembahkan skripsi ini untuk:

1. Ibu saya tercinta
2. Alamameter saya Univeristas PGRI Semarang
3. Dosen yang telah memberikan ilmu kepada saya

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama : Ahmad Syaifulloh

NPM : 20670116

Progdi : Informatika

Fakultas : Teknik dan Informatika

Menyatakan dengan sebenarnya bahwa skripsi yang saya buat ini benar-benar merupakan hasil karya sendiri, bukan plagiarisme.

Apabila ada kemudian hari skripsi ini terbukti plagiarisme, saya bersedia menerima sanksi atas perbuatan tersebut.

Semarang, 19 Agustus 2024

Yang membuat pernyataan

Ahmad Syaifulloh

NPM 20670116

ABSTRAK

Pencemaran udara di lingkungan perkotaan menjadi masalah serius akibat peningkatan polutan dari aktivitas industri, transportasi, dan domestik. Laporan terbaru IQAir menunjukkan penurunan kualitas udara di Indonesia, terutama di Tangerang Selatan yang mencatat konsentrasi PM2,5 tertinggi di Asia Tenggara. Penurunan ini mendesak perlunya solusi pemantauan kualitas udara secara *real-time*. Sistem pemantauan yang ada saat ini sering kali mahal dan terbatas jangkauannya, sehingga tidak memberikan gambaran yang komprehensif. Penelitian ini mengembangkan sistem pemantauan kualitas udara berbasis *Internet of Things* (IoT) menggunakan NodeMCU, yang dipilih karena kemampuan WiFi ESP8266 terintegrasi, memungkinkan koneksi internet tanpa perangkat tambahan. Dengan sistem ini, data kualitas udara dapat dikirim secara langsung ke *website* dalam bentuk angka, grafik, atau tulisan, memudahkan pemantauan *real-time* dari jarak jauh. Sistem ini bertujuan untuk mempermudah pemantauan di perusahaan, mendukung kepatuhan terhadap regulasi lingkungan, dan meningkatkan kesejahteraan karyawan. Pengujian sistem dilakukan melalui *white box*, *black box*, dan *user acceptance testing* (UAT). Hasil pengujian *white box* menunjukkan bahwa sistem ini baik karena tidak kompleks. Pengujian *black box* berhasil mencapai 100%, tanpa kegagalan. UAT menunjukkan tingkat kepuasan 85,33%, menandakan bahwa sistem ini sangat efektif dalam pemantauan kualitas udara secara *real-time*, memberikan nilai tambah signifikan bagi perusahaan dan karyawan dalam menjaga kualitas udara dan memenuhi standar lingkungan yang lebih tinggi.

Kata Kunci: Kualitas Udara, *Internet of Things* (IoT), NodeMCU, *Real-Time*, Teknologi *Website*.

PRAKATA

Puji syukur kami haturkan kepada Allah SWT atas berkat dan rahmat-Nya, sehingga penulis berhasil menyelesaikan skripsi ini dengan lancar. Skripsi yang berjudul "Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*" ini penulis susun sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer.

Penyusunan skripsi ini tidak lepas dari hambatan dan rintangan serta kesulitan-kesulitan. Namun, berkat bimbingan, bantuan, nasihat, dan dorongan serta saran-saran dari berbagai pihak, khususnya Pembimbing, segala hambatan dan rintangan serta kesulitan tersebut dapat teratasi dengan baik. Oleh karena itu, dalam kesempatan ini dengan tulus hati penulis sampaikan terima kasih kepada:

1. Dr. Sri Suciati, M.Hum., selaku Rektor Universitas PGRI Semarang.
2. Bapak Ibnu Toto Husodo, ST., MT., selaku Dekan Fakultas Teknik dan Informatika Universitas PGRI Semarang, yang telah memberikan izin untuk kami melakukan penelitian.
3. Bapak Bambang Agus Herlambang, S.Kom., M.Kom., selaku Ketua Program Studi Informatika Universitas PGRI Semarang.
4. Ibu Mega Novita, S.Si., M.Si., M.Nat.Sc., Ph.D., selaku Pembimbing I yang telah mengarahkan penulis dengan penuh ketekuna dan kecermatan
5. Bapak Ir. Agung Handayanto, M.Kom., selaku Pembimbing II yang telah membimbing penulis dengan penuh dedikasi yang tinggi.
6. Bapak dan Ibu Dosen Program Studi Informatika yang telah memberikan bekal ilmu kepada penulis selama belajar di Universitas PGRI Semarang.
7. Ibu Kartini selaku ibu kandung penulis yang tak hentinya mendo'akan dan memberikan dukungan atas jalannya studi dari awal sampai dengan penyelesaian skripsi ini.
8. Aliffia Devi Nurlistyanti, A. Md. Kep, orang yang tak kalah istimewa selain orang tua penulis yang telah memberikan dukungan selama jalannya penulisan skripsi ini.

9. Dan teman-teman penulis yang tidak bisa disebutkan satu persatu yang telah memberikan arahan penulis selama jalannya pembuatan skripsi ini.

Akhir kata penulis mengucapkan terima kasih kepada semua pihak yang terlibat dalam penulisan skripsi ini. Penulis berharap agar skripsi ini dapat memberikan manfaat bagi pembaca.

Semarang, 7 Agustus 2024

Penulis

Ahmad Syaifulloh

NPM 20670116

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

A. Latar Belakang Masalah

Udara merupakan suatu gas yang menglilingi bumi mengandung beberapa zat diantaranya oksigen, karbon dioksida, karbon monoksida dan lain sebagainya. Oksigen adalah salah satu unsur zat yang sangat berguna bagi keberlangsungan hidup makhluk yang ada di bumi. Udara memang materi yang tak kasat mata, tetapi dampaknya sangat berpengaruh pada kehidupan. Pencemaran udara diartikan dengan turunnya kualitas udara sehingga udara mengalami penurunan mutu dalam penggunaannya dan akhirnya tidak dapat dipergunakan lagi sebagai mana mestinya sesuai dengan fungsinya. Polusi udara akhir-akhir ini merupakan masalah yang banyak meresahkan masyarakat terutama dilingkungan perkotaan. Indeks Standar Pencemar Udara (ISPU) diatur dalam Peraturan Menteri Lingkungan Hidup dan Kehutanan (Permen LHK) Nomor 14 Tahun 2020. Dalam laporan terbarunya berjudul World Air Quality Report 2023, IQAir menyebutkan rata-rata konsentrasi PM2,5 di Indonesia pada 2023 adalah 37,1 mikrogram per meter kubik. Konsentrasi PM2,5 di Indonesia pada 2023 tersebut mengalami peningkatan 20 persen bila dibandingkan 2022. Dari berbagai wilayah, Tangerang Selatan menjadi kota dengan kualitas udara terburuk di Indonesia bahkan Asia Tenggara dengan konsentrasi PM2,5 sebanyak 71,1 mikrogram per meter kubik. Sedangkan secara global, Indonesia menempati peringkat ke-14 sebagai negara dengan rata-rata kualitas udara terburuk didunia [1]. Penurunan kualitas udara diakibatkan oleh berberapa aktivitas diantaranya aktivitas industri, transportasi, dan domestik telah menyebabkan peningkatan jumlah polutan berbahaya.

Dalam menghadapi masalah pencemaran udara tersebut, pemantauan secara *real-time* menjadi krusial. Maka dari itu dibutuhkan sistem pemantauan udara agar bisa mengetahui kualitas udara disuatu tempat atau wilayah tertentu. Berdasarkan dampak kualitas udara buruk dan manfaat pemantauan kualitas

udara, maka dibutuhkan membangun suatu sistem pemantauan kualitas udara secara *real-time* yang dapat diakses kapanpun dan dimanapun. Sistem pemantauan kualitas udara yang bagus ialah hasil data pemantauan dapat dilihat meskipun dari jarak yang jauh. Sedangkan sistem pemantauan udara konvensional cenderung mahal, canggung, dan terbatas dalam cakupan jangkauannya. Stasiun pemantauan udara biasanya tersebar di lokasi yang terbatas. Sehingga tidak dapat memberikan gambaran yang lengkap tentang kualitas udara di suatu tempat atau wilayah.

Internet of Things atau IoT menawarkan solusi yang potensial dalam pemantauan lingkungan, termasuk kualitas udara. Agar kualitas udara dapat dipantau secara *real-time*. Dalam penelitian ini didalam prototipe IoT menggunakan NodeMCU dibandingkan Arduino. Hal tersebut dikarenakan NodeMCU sudah dilengkapi dengan modul WiFi ESP8266 yang memungkinkan koneksi internet dan jaringan nirkabel langsung tanpa memerlukan tambahan *hardware*. Sedangkan Arduino memerlukan modul tambahan (seperti *Arduino WiFi Shield* atau ESP8266) untuk konektivitas WiFi, yang menambah biaya dan kompleksitas. Perangkat-perangkat IoT yang terkoneksi dapat dipasang secara luas dan relatif terjangkau, sehingga memungkinkan untuk membangun jaringan pemantauan udara yang luas dan terdistribusi. IoT adalah struktur di mana objek, orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke komputer [2]. Teknologi IoT dapat mengkombinasikan internet dengan web, ataupun aplikasi untuk memperoleh data informasi dan menampilkan data informasi dari jauh. IoT juga mampu membuat perangkat keras dapat berkomunikasi, bertukar data, dan saling mengendalikan melalui *website* [3]. Data hasil monitoring dari perangkat IoT ditampilkan ke *website* dalam bentuk angka, grafik, maupun tulisan agar mudah untuk dibaca dan dipahami oleh *user*.

Oleh sebab itu, peneliti membuat sebuah sistem pemantuan kualitas udara berbasis IoT menggunakan NodeMCU dengan *interface website* yang

lebih efektif, akurat dan mudah diakses oleh pengguna. Sistem pemantauan ini tidak hanya meningkatkan efisiensi dalam pemantauan kualitas udara. Sistem pemantauan ini juga mendukung upaya perusahaan dalam menerapkan standar operasional yang lebih tinggi dan tanggung jawab lingkungan yang lebih besar. Dalam jangka panjang, penerapan sistem ini dapat memberikan nilai tambah yang signifikan bagi perusahaan. Baik dari segi kepatuhan terhadap regulasi lingkungan maupun dalam menjaga kesejahteraan karyawan dan masyarakat sekitar.

B. Identifikasi Masalah

Berdasarkan uraian latar belakang diatas, permasalahan yang akan dibahas dalam penulisan ini adalah sebagai berikut:

1. Polusi udara yang meningkat, terutama di area industri dan perkotaan, meningkatkan risiko kesehatan bagi karyawan dan dapat memengaruhi produktivitas serta kesejahteraan di tempat kerja. Perusahaan memerlukan solusi pemantauan yang lebih efektif untuk mengelola dan mengurangi dampak polusi.
2. Sistem yang ada mahal, dan tidak mampu memberikan data kualitas udara secara *real-time* yang menyeluruh. Ini menyulitkan perusahaan dalam memantau kondisi udara secara kontinu dan membuat keputusan yang tepat untuk perbaikan lingkungan kerja.
3. Meskipun teknologi IoT menawarkan solusi yang lebih terjangkau dan luas. Perusahaan menghadapi tantangan dalam memastikan akurasi sensor, menjaga koneksi internet yang stabil. Serta mengintegrasikan dan memvisualisasikan data secara efektif.
4. Data kualitas udara yang dihasilkan harus mudah diakses dan dipahami oleh pihak manajemen perusahaan untuk meningkatkan kesadaran tentang kondisi lingkungan kerja dan mendukung pengambilan keputusan yang lebih baik dalam menjaga kesehatan karyawan dan memenuhi standar lingkungan.

C. Batasan Masalah

Adapun batasan masalah dalam penelitian ini adalah:

1. Menggunakan NodeMCU daripada Arduino karena NodeMCU telah dilengkapi dengan modul WiFi ESP8266, memungkinkan koneksi internet dan nirkabel langsung tanpa perlu perangkat keras tambahan. Sebaliknya, Arduino memerlukan modul tambahan seperti *Arduino WiFi Shield* atau ESP8266 untuk koneksi WiFi, yang menambah biaya dan kompleksitasnya.
2. Menggunakan sensor MQ-135 untuk sensor mendeteksi gas-gas berbahaya, contohnya seperti karbon monoksida, asap, karbon dioksida, dsb.
3. Alat yang dibuat hanya satu buah.
4. Menggunakan *server* web lokal dan dihubungkan dengan satu jaringan wifi yang sama.
5. Pada penelitian ini pengembangan aplikasi menggunakan metode pengembangan *waterfall* yang digunakan hanya sampai tahap *testing*, tanpa *deployment and maintenance*.

D. Perumusan Masalah

Berdasarkan latar belakang dan batasan masalah yang telah utarakan diatas, maka dapat disimpulkan permasalahan dalam penelitian ini adalah bagaimana cara merancang dan membuat sistem pemantauan kualitas udara berbasis IoT (*Internet of Things*) menggunakan NodeMCU dengan *interface website*?

E. Tujuan Penelitian

Tujuan dalam penelitian merupakan jawaban atau sasaran yang ingin dicapai dalam sebuah penelitian. Adapun tujuan penelitian ini yaitu:

1. Merancang sistem pemantauan kualitas udara berbasis IoT untuk melakukan *monitoring* kualitas udara.
2. Mampu mengintegrasikan sistem dengan *web server* sebagai media informasi hasil *monitoring*.

F. Manfaat Penelitian

1. Manfaat Teoritis

Hasil penelitian ini diharapkan dapat memperluas pengetahuan tentang sistem pemantauan kualitas udara menggunakan NodeMCU berbasis *Internet of Things* (IoT), yang dapat diimplementasikan di lingkungan perusahaan. Penelitian ini bertujuan untuk memberikan solusi praktis bagi perusahaan dalam memantau dan mengelola kualitas udara secara *real-time*. Serta sebagai alternatif efektif untuk pengujian emisi gas buang pada kendaraan bermotor, mendukung kepatuhan terhadap regulasi lingkungan, dan meningkatkan kesehatan serta produktivitas karyawan.

2. Manfaat Praktis

Hasil penelitian ini diharapkan bermanfaat bagi pihak-pihak berikut:

a. Bagi Peneliti

Menambah dan memperluas wawasan ilmu pengetahuan penulis dalam bidang penelitian yang berkaitan dengan sistem kualitas udara yang ada disuatu tempat menggunakan NodeMCU berbasis *Internet of Things* serta dapat menerapkan wawasan teori yang diperoleh dari Universitas PGRI Semarang.

b. Bagi Universitas PGRI Semarang

Hasil penelitian ini diharapkan menjadi rujukan atau panduan bagi peneliti selanjutnya yang ingin melakukan penelitian lebih lanjut berhubungan dengan perancangan sistem monitoring karbon monoksida dan gas berbahaya lainnya menggunakan NodeMCU berbasis *Internet of Things*.

c. Bagi Perusahaan

Sistem pemantauan kualitas udara berbasis IoT dan web memberikan manfaat signifikan bagi perusahaan dengan memastikan lingkungan kerja yang sehat dan aman. Sistem ini memungkinkan pemantauan real-time terhadap kualitas udara, memberikan peringatan dini untuk mencegah paparan gas berbahaya, dan membantu perusahaan mematuhi regulasi lingkungan. Selain itu, data yang dikumpulkan

memungkinkan analisis yang lebih baik untuk meningkatkan efisiensi operasional dan produktivitas karyawan. Dengan menjaga kesehatan dan keselamatan karyawan, perusahaan juga dapat meningkatkan reputasi dan mengurangi risiko operasional.

BAB II

KAJIAN TEORI

A. Tinjauan Pustaka

Tinjauan Pustaka adalah proses umum yang dilalui untuk mendapatkan teori yang relevan dengan masalah yang diteliti. Mencari beberapa kumpulan penelitian yang terkait kemudian diangkat untuk mendukung penelitian yang dibuat agar penelitian semakin menguat. Ini membantu peneliti untuk memahami konteks penelitian yang sedang dilakukan, mengidentifikasi kesenjangan pengetahuan yang ada dan membangun dasar teoritis untuk studi peneliti.

Tabel 2. 1. Penelitian Terdahulu Yang Berkaitan Dengan Sistem Pemantauan Kualitas Udara Berbasis IoT Menggunakan NodeMCU Dengan *Interface Website*

No.	Nama dan Judul Penelitian	Metode Penelitian	Kesimpulan
1.	Hendi Budianto dan Budi Sumanto. Perancangan Sistem Monitoring Kualitas Udara dalam Ruangan Berbasis Internet of Things. Departemen Teknik Elektro dan Informatika, Universitas Gadjah Mada. (2024)	*Prototype*	Sistem pemantauan kualitas udara berbasis IoT telah berhasil dikembangkan untuk memantau kualitas udara secara *real-time* dan jarak jauh. Sensor yang digunakan dapat mendeteksi polutan seperti CO dan CO2, serta suhu dan debu, dengan hasil ditampilkan pada layar LCD dan *website*. Data disimpan dengan baik dalam *database* dan ditampilkan melalui *dashboard* yang menyediakan grafik dan peringatan tentang penurunan

			kualitas udara. Sistem ini efektif untuk meningkatkan kesadaran akan pentingnya udara bersih di dalam ruangan.
2.	Reza Ramadhan dan Joko Christian Chandra. "Rancang Bangun Sistem Pemantauan Kualitas Udara Berbasis IoT Dengan NodeMCU". Universitas Budi Luhur (2022)	*Prototype*	Proses dan desain alat ini dapat berfungsi dengan baik, sensor dapat mendeteksi terjadinya perubahan kualitas udara, keluaran yang dihasilkan yaitu LED, buzzer dan LCD juga sesuai dengan apa yang diharapkan. Sensor MQ-135 tidak hanya dapat mendeteksi benzena, alkohol dan asap rokok, tapi juga dapat mendeteksi naptha/butana dari korek gas. NodeMCU dapat terhubung dengan wifi yang dituju dengan baik, sehingga pengiriman data tidak mengalami masalah. Thingspeak sebagai IoT platform juga bekerja dengan baik, sehingga dapat menampilkan nilai dari kualitas udara yang telah dikirimkan dari NodeMCU.
3.	Grace C. Rumampuk, Vecky C. Poekoel dan Arthur M. Rumagit. "Perancangan Sistem	*Prototype*	Berdasarkan hasil penelitian dari tugas akhir ini, maka dapat ditarik beberapa kesimpulan bahwa Rancangan Sistem *monitoring* udara berbasis IoT

	Monitoring Kualitas Udara Dalam Ruangan Berbasis Internet of Things". Jurusan Teknik Elektro, Universitas Sam Ratulangi Manado. (2021)		menggunakan sensor MQ-135 dan berhasil melakukan pengiriman data kualitas udara ke Platform OVoRD dan data yang terkirim ditampilkan dalam bentuk angka dan grafik dalam platform OVoRD.
4.	Sumardi Sadi, Sri Mulyati, Perdana Bagaskara Setiawan. "Internet of Things Pada Sistem Monitoring Kualitas Udara Menggunakan Web Server". Universitas Muhammadiyah Tangerang (2022).	*Prototype*	Pembuatan alat monitoring ini mendapatkan hasil gas CO sebesar 0,25, gas NO2 sebesar 0,83 dan partikulat PM 2.5 sebesar 0.10. Alat yang dibuat dengan konsep IoT menggunakan web server mqtt thingsboard berjalan dengan baik secara realtime berdasarkan hasil pengiriman waktu, tempat dan ketiga parameter yang dipakai serta suhu udara sekitar. Ketiga parameter yang dilakukan penelitian ini dapat dinyatakan hasil gas CO dan Partikulat PM 2.5 dalam keadaan bagus untuk manusia sedangkan untuk gas NO2 dinyatakan tidak sehat bagi manusia.

5.	Muhammad Hasanuddin, Herdianto. "Sistem Monitoring dan Deteksi Dini Pencemaran Udara Berbasis *Internet of Things* (IoT)". Sains dan Teknologi, Sistem Komputer, Universitas Pembangunan Panca Budi (2023).	*Waterfall*	Implementasi Sistem Monitoring dan Deteksi Dini Pencemaran Udara Berbasis *Internet of Things* (IoT) telah membuktikan keefektifannya dalam mendukung pemantauan kualitas udara secara *real-time* dan memberikan deteksi dini terhadap tingkat pencemaran udara yang berbahaya. Dengan Skala ISPU biasanya terdiri dari beberapa kategori yang menggambarkan tingkat kualitas udara, seperti rentang 1-50 dikatakan "Baik", rentang 51-100 dikatakan "Sedang", rentang 101-200 dikatakan "Tidak Sehat", rentang 201-300 dikatakan "Sangat Tidak Sehat", dan rentang 301+ dikatakan "Berbahaya". Setiap kategori memiliki interpretasi dan peringatan tertentu terkait dengan kesehatan masyarakat. Grafik *Thingspeak* menunjukkan pada bahwa di jam 14:20 pencemaran udara yang disebabkan oleh gas naik di atas 250 yang di katagorikan sangat tidak baik untuk makhluk hidup, Dan dengan menggunakan

			sensor - sensor lingkungan yang terhubung ke mikrokontroler dan *gateway* node, data sensor dapat dikumpulkan dan dikirimkan ke *cloud server* untuk pemrosesan dan analisis.

B. Landasan Teori

1. Udara

Udara merupakan faktor terpenting dalam kehidupan, namun dengan meningkatnya pembangunan kota dan pusat-pusat industri, kualitas udara telah mengalami perubahan. Dikarenaakan banyaknya pembangunan pusat-pusat industri dan banyaknya kendaraan bermotor yang membuat udara mencadi tercemar dan kurang sehat untuk kehidupan. Pencemaran udara dapat diartikan dengan menurunnya kualitas udara, sehingga udara mengalami penurunan mutu dalam penggunaannya dan akhirnya tidak dapat dipergunakan lagi sebagai mana mestinya sesuai dengan fungsinya.

Menurut Direktorat Pengendalian Pencemaran Udara KLHK, ISPU merupakan angka tanpa satuan yang digunakan untuk menggambarkan kondisi mutu udara ambien di lokasi tertentu dan didasarkan kepada dampak terhadap kesehatan manusia, nilai estetika, dan makhluk hidup lainnya. Perhitungan ISPU berdasarkan hasil pengukuran tujuh parameter pencemar udara yakni PM10, PM2.5, NO2, SO2, CO, O3, dan HC. Pengukuran parameter pencemar udara tersebar di 72 stasiun di berbagai daerah. Berdasarkan Permen LHK No. 14 Tahun 2020 tentang Indeks Standar Pencemar Udara, ISPU terdapat 5 jenis rentang kualitas udara.

a. Rentang 0-50 memiliki kualitas udara baik
b. Rentang 51-100 berarti kualitas udara sedang
c. Rentang 101-200 kualitas udara tidak sehat yang bersifat merugikan manusia, hewan, dan tumbuhan.

d. Rentang 201-300 kualitas udara sangat tidak sehat pada rentang dapat meningkatkan risiko kesehatan pada kelompok sensitif.
e. Rentang >300 kualitas udara berbahaya dapat merugikan kesehatan secara serius dan perlu penanganan cepat.

Tabel 2.2. Data 10 Wilayah Berpolusi Di Indonesia Pada Sabtu, 1 Juni 2024 [4]

No.	Nama Data	Nilai
1.	Jawa Barat	164
2.	Banten	147
3.	DKI Jakarta	119
4.	Kep. Riau	97
5.	Riau	88
6.	Jawa Timur	84
7.	Bengkulu	78
8.	Papua	77
9.	Jambi	74
10.	Yogyakarta	70

Pada Tabel 2.2 kualitas udara di Jawa Barat sore ini terburuk di Indonesia. Berdasarkan halaman Indeks Standar Pencemar Udara (ISPU) Kementerian Lingkungan Hidup dan Kehutanan (KLHK) pada Sabtu (1/6/2024) pukul 16.00 WIB terungkap bahwa indeks kualitas udara di Jawa Barat sebesar 164. Di bawah Jawa Barat, ada Banten yang menempati posisi kedua terburuk di Indonesia dengan indeks kualitas udara 147. Kemudian, di posisi ketiga ada DKI Jakarta dengan indeks kualitas udara 119. Ini artinya, tidak ada wilayah yang memiliki kualitas udara berbahaya.

2. Karbon Dioksida (CO2)

Karbon dioksida (CO2) adalah produk sampingan yang kita hembuskan dari dalam tubuh saat kita bernapas. Gas ini juga dihasilkan ketika bahan bakar fosil dibakar atau dari vegetasi yang membusuk. Tanpa karbon

dioksida, bumi bisa menjadi sangat dingin. Namun, peningkatan konsentrasi CO2 di atmosfer juga bisa menyebabkan suhu rata-rata global meningkat yang akan mengganggu aspek lain dari iklim bumi. Pada tahun 2023 rata rata indeks global saat ini untuk karbondioksida adalah 419,3 ppm [5]. Paparan CO2 yang tinggi bisa menyebabkan berbagai efek pada tubuh manusia. Hal itu mungkin termasuk sakit kepala, pusing, gelisah, kesemutan, kesulitan bernapas, berkeringat, kelelahan, peningkatan denyut jantung, tekanan darah tinggi, koma, asfiksia, dan kejang-kejang. Berikut tingkat CO2 di udara dan potensi masalah kesehatan yang bisa ditimbulkannya [6]:

a. 400 ppm: Ini tingkat CO2 dalam udara di luar ruangan rata-rata.
b. 400–1.000 ppm: Ini tingkat karbon dioksida yang biasanya ada di ruangan yang memiliki pertukaran udara yang baik.
c. 1.000-2.000 ppm: Tingkat ini bisa menyebabkan keluhan kantuk dan udara yang buruk.
d. 2.000–5.000 ppm: Ini tingkat CO2 yang bisa menyebabkan sakit kepala, kantuk, dan udara yang stagnan, pengap, dan sesak. Konsentrasi yang buruk, kehilangan perhatian, peningkatan denyut jantung dan sedikit mual juga bisa terjadi.
e. ppm: Ini menunjukkan kondisi udara yang tidak biasa di mana tingkat tinggi gas lain juga bisa hadir. Toksisitas atau kekurangan oksigen bisa terjadi. Ini adalah batas paparan yang diizinkan untuk paparan di tempat kerja sehari-hari.
f. 40.000 ppm: Tingkat ini ini sangat berbahaya karena bisa menyebabkan kekurangan oksigen.

Tabel 2.3. Proporsi Kontribusi Emisi CO2 Terkait Enegeri Berdasarkan Sektor (2021) [7]

No.	Nama Data	Nilai
1.	Ketenagalistrikan	43%
2.	Transportasi	25%

3.	Industri	23%
4.	Bangunan	5%
5.	Energi untuk pribadi	3%
6.	Pertanian	1%

Pada Tabel 2.3 diatas ketenagalistrikan menjadi sektor penyumbang emisi karbon dioksida (CO2) terbesar di Indonesia. Proporsinya mencapai 43% dari total emisi di Indonesia. Sektor kedua terbesar yakni transportasi dengan proporsi 25%. Disusul sektor industri di posisi ketiga yang menyumbang 23% emisi. Keempat ada sektor bangunan dengan emisi sebesar 5%. Posisi kelima dan keenam diisi sektor energi pribadi dan pertanian dengan persentase masing-masing 3% dan 1%. Emisi CO2 dari pembakaran bahan bakar menjadi pendorong terbesar emisi gas rumah kaca secara keseluruhan.

3. Karbon Monoksida (CO)

Karbon monoksida (CO) adalah gas yang tidak berwarna, tidak berbau, dan tidak berasa, yang sangat beracun bagi manusia dan hewan jika dihirup dalam jumlah yang signifikan. Karbon monoksida dihasilkan dari pembakaran tidak sempurna bahan bakar fosil seperti bensin, gas alam, minyak, kayu, batu bara dan sumber alami contohnya seperti kebakaran hutan dan aktivitas vulkanik. Menurut Permenaker No.13 tahun 2012 nilai ambang batas (NAB) karbon monoksida adalah 25 ppm, jika lebih dari nilai ambang batas maka dapat menyebabkan gangguan kesehatan pada manusia.

Keracunan karbon monoksida (CO) berbahaya karena dapat mengikat hemoglobin dalam darah lebih kuat dibandingkan oksigen, sehingga menghambat distribusi oksigen ke seluruh tubuh. Ini dapat menyebabkan keracunan dengan gejala seperti sakit kepala, pusing, lemah, mual, kebingungan, dan pada tingkat yang tinggi, dapat menyebabkan kematian. Paparan CO dalam jangka panjang dapat menyebabkan kerusakan pada jantung dan otak, serta komplikasi kesehatan lainnya.

Tabel 2. 4. Sumber Pencemaran Karbon Monoksida (CO) [8]

No.	Sumber Pencemar CO	Jumlah Presentasi (%)
1.	Transportasi	63.8
2.	Pembakaran Stasioner	1.9
3.	Proses Industri	9.6
4.	Pembuangan Limbah Padat	7.8
5.	Lain-lain	16.9

Pada Tabel 2.4 diatas transportasi adalah penyumbang terbesar gas CO yaitu sebesar 63.8%. Pada proses industri sebesar 9.6%, pembuangan limbah padat sebesar 7.8%, pembakaran stasioner sebesar 1.9% dan dari berbagai sumber lain sebesar 16.9%. Sumber-sumber lain bisa berasal dari sumber alami contohnya dari aktivitas manusia, aktivitas vulkanik, oksidasi metana dari atmosfer dan lain sebagainya.

Tabel 2. 5. Faktor Emisi Kendaraan Bermotor [9]

No.	Jenis Kendaraan	Faktor Emisi CO (g/km)
1.	Mobil Bensin	40
2.	Mobil Solar	2.8
3.	Motor	14
4.	Bis	11
5.	Truk	8.4

Pada Tabel 2.5 emisi terbesar kendaraan bermotor ada pada mobil dengan bahan bakar bensin dengan faktor emisi CO sebesar 40 g/km. Diikuti dengan motor sebesar 14 g/km, bis sebesar 11 g/km, truk sebesar 8.4 g/km dan mobil dengan bahan bakar solar sebesar 2.8 g/km. Mesin bensin seringkali tidak membakar bahan bakar dengan sempurna, terutama pada kondisi beban tinggi atau saat akselerasi cepat.

Tabel 2. 6. Dampak Paparan Karbon Moniksida (CO) Terhadap Tubuh [10]

No.	Kadar CO	Waktu Kontak	Dampak Terhadap Tubuh
1.	≤ 100 ppm	Sebentar	Dianggap aman
2.	± 30 ppm	8 Jam	Pusing dan mual
3.	± 1000 ppm	1 Jam	Pusing dan kulit berubah kemerah-merahan
4.	± 1300 ppm	1 Jam	Kulit jadi merah tua dan rasa pusing yang hebat
5.	> 1300 ppm	1 Jam	Kematian

Pemberian Karbon Monoksida (CO) selama 1 sampai 3 minggu pada konsentrasi sampai dengan 100 ppm yang tidak memberi pengaruh yang nyata terhadap tanaman-tanaman tingkat tinggi. Namun kemampuan terhadap fiksasi nitrogen oleh bakteri bebas akan terhambat dengan pemberian Karbon Monoksida (CO) selama 35 jam pada konsentrasi 2000 ppm, sedangkan untuk kemampuan fiksasi nitrogen oleh bakteri yang terdapat pada akar tanaman dapat terhambat dengan pemberian Karbon Monoksida (CO) sebesar 100 ppm selama satu bulan [11].

4. Sistem Monitoring

Monitoring atau pemantuan adalah identifikasi keberhasilan atau kegagalan aktual atau potensial sedini mungkin dan setiap saat selama penyelesaian kegiatan untuk menilai kemajuan dan merekomendasikan tindakan untuk mencapai dan melaksanakan tujuan. Monitoring merupakan pemantauan yang bisa diuraikan sebagai pemahaman tentang apa yang mau diketahui, pemantauan dilakukan supaya bisa menciptakan pengukuran melalui waktu yang menampilkan pergerakan ke arah tujuan [12].

Monitoring adalah proses teratur pengumpulan informasi serta pengukuran kemajuan atas objektif program. Memantau segala perubahan, yang fokus pada proses serta keluaran. Monitoring menyediakan informasi dasar guna menanggapi kasus, sementara itu evaluasi merupakan

memposisikan data informasi tersebut supaya bisa digunakan serta diharapkan memberikan nilai tambah. Evaluasi adalah mempelajari peristiwa, membagikan penyelesaian guna suatu permasalahan, anjuran yang mesti dibuat, menganjurkan revisi. Akan tetapi tanpa monitoring, evaluasi tidak bisa dilakukan sebab tidak mempunyai informasi dasar buat dilakukan analisis, serta dikhawatirkan bakal menyebabkan spekulasi, oleh sebab itu monitoring serta evaluasi mesti berjalan bersamaan.

Sistem monitoring adalah proses pengumpulan data informasi yang kemudian dilakukan analisis terhadap data tersebut yang hasilnya akan ditampilkan. Sistem monitoring dapat membantu dalam proses kegiatan pemantauan, merekap data hasil pemantauan, dan pembuatan laporan. Sistem monitoring akan memberikan hasil yang baik asalkan dirancang serta dicoba secara efisien. Berikut kriteria sistem monitoring yang efisien [13] :

a. *User friendly*. Sistem monitoring harus dirancang dengan sederhana dan mudah dimengerti dan tepat sasaran.
b. Perancangan yang matang. Tujuannya adalah aplikasi teknis yang terarah dan terstruktur. Oleh karena itu, perancangan dilakukan dengan matang dan memenuhi aspek-aspek teknis seperti apa, siapa, kapan, mengapa, di mana, dan bagaimana akan dilaksanakan monitoring.
c. Fokus pada indikator utama. Indikator merupakan titik fokus dari kegiatan monitoring, banyaknya indikator menyebabkan objek dan pelaku monitoring tidak fokus. Yang kemudian akan berdampak pada pelaksanaan sistem.
d. Prosedur pengumpulan data. Data yang didapatkan dalam kegiatan monitoring harus memiliki prosedur yang tepat agar memudahkan proses keluar dan masuknya data.

5. Internet of Things (IoT)

Internet of Things (IoT) merupakan suatu kemampuan interaksi antara manusia dan komputer melalui jaringan internet untuk memindahkan suatu data [14]. *Internet of Things* pertama kali diperkenalkan pada tahun 1999

oleh Kevin Asthon. Penerapan IoT dalam berbagai bidang banyak ditemukan dalam kehidupan sehari-hari pada manusia. Menurut CISCO, telah menargetkan bahwa pada tahun 2020, 50 miliar objek akan terhubung dengan internet [15]. IoT sebagai sebuah infrastruktur jaringan global yang menghubungkan benda-benda fisik maupun virtual melalui eksploitasi data capture dan kemampuan komunikasi. Atau bisa dikatakan IoT adalah suatu teknologi komunikasi antar mesin menggunakan koneksi internet atau *machine to machine* [16].

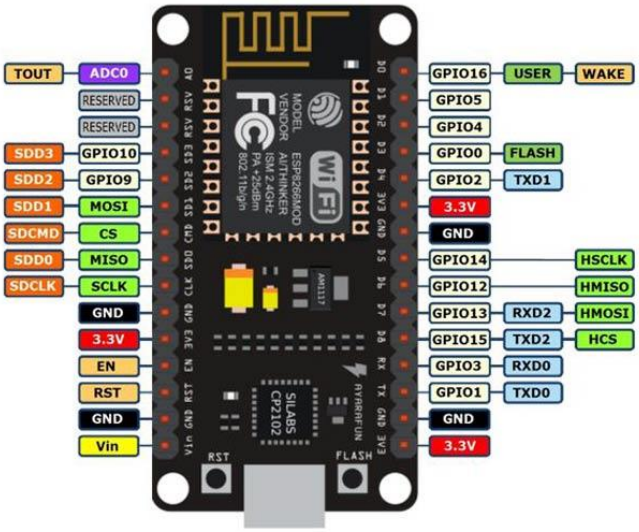
Salah satu contoh sederhana dari IoT adalah thermostat pintar yang dapat menyesuaikan suhu di dalam rumah sesuai dengan pola penggunaan dan preferensi pengguna. Namun, potensi IoT jauh lebih luas dan bisa diterapkan dalam berbagai industri seperti kesehatan, manufaktur, pertanian, transportasi, dan lain sebagainya. Konsep kerja IoT mengacu pada tiga elemen utama, yaitu barang fisik yang dilengkapi modul IoT, perangkat koneksi internet, dan *cloud* data center sebagai database. Semua penggunaan perangkat yang terhubung internet akan menyimpan data dan terkumpul sebagai big data dan kemudian dapat dianalisis lebih lanjut [17].

6. NodeMCU

Pada penelitian ini digunakan board NodeMCU ESP8266. Modul ini merupakan salah satu platform yang terbilang murah untuk ukuran mikrokontroler. Meskipun murah namun nyatanya ESP8266 ini terbilang efektif untuk mengontrol dan berkomunikasi melalui jaringan internet. NodeMCU adalah sebuah platform IoT yang bersifat *open source*. Terdiri dari perangkat keras berupa *System on Chip* ESP8266. dari ESP8266 buatan Espressif *System*, juga *firmware* yang digunakan, yang menggunakan bahasa pemrograman *scripting Lua* [18].

Lahirnya NodeMCU berdekatan dengan rilisnya ESP8266 pada tanggal 30 Desember 2013, Espressif *Systems* selaku pembuat ESP8266 memulai produksi ESP8266 yang menjadi SoC Wi-Fi yang terintegrasi dengan *Processor Tensilica Xtensa* LX106. Kemudian NodeMCU dimulai pada 13 Oktober 2014 saat Hong *meng-commit* file pertama nodemcu-

firmware ke Github. Dua bulan setelah itu *project* itu dikembangkan ke *platform* perangkat keras ketika Huang Rui *meng-commit file* dari *board* ESP8266, yang diberi nama devkit v.0.9. NodeMCU ESP8266 adalah jenis modul pengembangan dari *platform* IoT (*Internet of Things*) sejenis ESP8266 yaitu chip Wi-Fi dengan kemampuan mengatur TCP (*Transmission Control Protocol*) / IP (*Internet Protocol*) dan MCU (unit mikrokontroler) yang diproduksi oleh produsen asal China yang berbasis di Shanghai, *Espressif Systems*. Modul tersebut memiliki fungsi serupa dengan modul arduino pada umumnya, yang membedakan adalah fungsi khususnya yaitu "*Connected to Internet*". Penggunaan NodeMCU sebagai wadah atau base khusus ESP8266 untuk meringkas penggunaan komponen dan fitur-fitur lainnya seperti pembagian catu daya 5 volt dan 3 volt, penyediaan *port external* DC *power supply*, percabangan pin *input output*, penyesuai tegangan DC *power supply* 10 sesuai tegangan kerja ESP8266, dan sebagainya. NodeMCU bisa dikatakan sebagai *board* Arduino-nya ESP8266 yang di mana memiliki kapabilitas akses terhadap wifi. Berikut adalah gambar NodeMCU. Gambar 2.1 menunjukkan gambar NodeMCU tipe ESP8266.



Gambar 2. 1. NodeMCU ESP 8266

Sumber: www.panduancode.com [19]

Dari Gambar 2.1 diatas sekilas kita sudah mengetahui letak dan nama nama pin yang tersedia dalam NodeMCU tipe ESP8266 tersebut. Adapun fungsi setiap pin yang terdapat dalam NodeMCU ESP8266 adalah sebagai berikut.

a. Vin (*Voltage In*) / VU (*USB Voltage*): Pin masukan tegangan yang dapat diberi tegangan antara 4.5V hingga 9V melalui pin ini untuk menyediakan daya ke NodeMCU.
b. GND (*Ground*): Pin tanah atau ground yang digunakan sebagai referensi tegangan nol (0V).
c. 3V3 (3.3V *Output*): Ini adalah keluaran tegangan 3.3V yang dapat digunakan untuk memberi daya ke perangkat lain atau sebagai referensi tegangan.
d. EN (*Enable/Reset*): Pin yang digunakan untuk mengatur ulang (*reset*) modul ESP8266. Jika Anda menghubungkan ke VCC, itu akan menyalakan NodeMCU. Jika diberi GND, itu akan me-*reset* modul.
e. RST (*Reset*): Pin *reset* digunakan untuk menghubungkannya ke GND akan mereset NodeMCU.
f. A0 (*Analog Input* 0): Salah satu pin masukan analog yang dapat digunakan untuk mengukur tegangan analog pada pin ini.
g. D0 - D8 (*Digital Pins* 0-8): Ini adalah pin-pin digital yang dapat digunakan sebagai *input* atau *output*. Mereka juga dapat digunakan sebagai GPIO (*General-Purpose Input/Output*) dan memiliki berbagai fungsi tambahan.
h. TX (*Transmit*): Pin TX yang digunakan untuk mentransmisikan data dari ESP8266 ke perangkat lain.
i. RX (*Receive*): Pin RX digunakan untuk menerima data dari perangkat lain ke ESP8266.
j. SD2 - SD3 (*Special Function Pins*): Pin ini adalah pin khusus yang memiliki fungsi tambahan yang terkait dengan *flash memory* dan komunikasi SPI. Dalam kebanyakan kasus, mereka digunakan dalam mode pemrograman khusus.

k. GPIO16 (*General Purpose Input/Output* 16): Pin GPIO yang dapat digunakan sebagai *input* atau *output*. Biasanya digunakan dalam proyek-proyek tertentu seperti *wake-up* dari mode *deep sleep*.
l. CH_PD (*Chip Enable*): Pin ini yang mengaktifkan chip ESP8266. Menghubungkannya ke VCC adalah langkah yang diperlukan untuk mengaktifkan modul.
m. GPIO15 (*General Purpose Input/Output* 15): Pin GPIO yang digunakan untuk mengatur *mode boot* ESP8266. Biasanya diterapkan dengan resistors eksternal saat memprogram NodeMCU.
n. GPIO2 (*General Purpose Input/Output* 2): Pin GPIO yang biasanya digunakan untuk mengontrol *modus boot* NodeMCU dan untuk menghubungkan LED *on board*.
o. GPIO0 (*General Purpose Input/Output* 0): Pin GPIO yang sering digunakan untuk mengontrol modus pemrograman NodeMCU. Dalam kondisi tertentu, harus dihubungkan ke GND untuk memulai proses pemrograman.

7. Arduino Integrated Development Environment (Arduino IDE)

Bahasa pemrograman Arduino digunakan dalam program pembuatan sistem yang menggunakan papan Arduino, dalam penelitian ini digunakan NodeMCU. Arduino IDE merupakan perangkat lunak pengolah teks program yang ditujukan untuk keluarga arduino dan clonenya. Program ini meliputi *code editor*, *compiler* dan *uploader*. Selain menuliskan kode secara manual, pada Arduino IDE juga telah tersedia contoh program yang bisa digunakan sebagai referensi. Arduino IDE cukup mudah digunakan karena menggunakan bahasa pemrograman C/C++ yang umum digunakan. Selain itu, pada editornya juga terdapat kode warna yang membedakan antara sintak, komentar dan label dengan tulisan lainnya. Pada program ini terdapat menu bar dan *toolbar* yang memudahkan penggunanya mengakses fitur arduino. Beberapa *toolbar* yang umum digunakan beserta fungsinya yakni *Verify* untuk mengecek kode dari kesalahan sintak ataupun nilai. *Upload*

untuk mengunggah kode ke perangkat, *New* untuk membuat *sketch* (sebutan untuk lembar kerja pada Arduino IDE) baru dan Serial Monitor untuk membuka tampilan komunikasi serial arduino. Untuk penelitian ini *software* Arduino IDE digunakan untuk memprogram NodeMCU.

8. Website

Website adalah suatu kumpulan-kumpulan halaman yang menampilkan berbagai macam informasi teks, data, gambar, video maupun gabungan dari semuanya bersifat statis dan dinamis. Sebelum dibahas lebih lanjut, tentunya terlebih dahulu mengetahui pengertian web. Menurut Sibero (2014:11), "Web merupakan suatu sistem yang berkaitan dengan dokumen digunakan sebagai media untuk menampilkan teks, gambar, multimedia dan lainnya pada jaringan internet" [20].

Sedangkan menurut Hidayatullah dan Kawistara (2015:3), "Web adalah suatu sistem yang ditemukan oleh Tim Bernes-Lee untuk menyusun arsip-arsip risetnya, sehingga memudahkan pencarian informasi yang dibutuhkan" [21]. Berdasarkan teori diatas dapat disimpulkan bahwa pengertian web adalah suatu sistem yang menudahkan pencarian informasi untuk menampilkan teks, gambar, multimedia dan lain sebagainya pada jaringan internet. Internet adalah jaringan global menghubungkan komputer satu dengan komputer yang lainnya untuk mengakses sebuah data.

Website adalah sumber dari halaman-halaman situs, yang terangkum dalam sebuah domain, yang ada di dalam WWW (*World Wide Web*) di Internet. Halaman web adalah dokumen yang dituliskan dengan format HTML atau kepanjangan dari (*Hyper Text Markup Language*), yang bisa diakses kapanpun melalui HTTP. HTTP adalah protokol yang dapat menyampaikan informasi melalui server website untuk ditampilkan kepada pemakai melalui *web browser. Web browser* adalah aplikasi perangkat lunak yang digunakan untuk menyajikan sumber informasi dan menampilkan hasil *website* yang telah dibuat. Berdasarkan teori di atas dapat disimpulkan bahwa *website* adalah kumpulan halaman yang berisi informasi baik berupa

gambar, teks, suara, animasi, atau gabungan dari semuanya yang disediakan oleh koneksi internet.

9. Laravel

Didalam penelitian menggunakan *software* aplikasi Laravel 10 untuk membuat *interface website.* Laravel merupakan *framework* berbasis pemrograman PHP yang dapat digunakan dalam proses pengembangan *website* supaya lebih maksimal. Penggunaan Laravel akan membuat *website* yang dihasilkan lebih maksimal dan dinamis. Kehadiran *framework* Laravel membuat pemrograman PHP menjadi maksimal dan lebih *powerfull*. Sebab, kehadiran *framework* Laravel lebih baik dan fitur-fiturnya pun lebih ringan dibandingkan framework lain. *Framework* Laravel ini menggunakan struktur *Model View Controller* atau MVC. MVC sendiri adalah model aplikasi pemisah antara tampilan komponen aplikasi dan data. Dengan menggunakan model MVC, para pengguna Laravel jadi lebih mudah memahami dan mempelajari Laravel. Dan juga proses pembuatan aplikasi website tentu akan menjadi lebih cepat [22].

Framework Laravel juga tersedia fitur bawaan yang cukup lengkap, termasuk fitur otentikasi. Jadi, *framework* ini cenderung fokus pada level end user. Di *mana framework* Laravel ini memang sederhana tapi memiliki banyak keunggulan. Baik dari sisi tampilannya maupun sisi penulisan kodenya. Meski begitu, *framework* Laravel tetap banyak peminatnya untuk membuat aplikasi berbasis web sebab fiturnya yang lengkap. Hal ini karena fleksibilitasnya dan keunggulan dalam proses pengembangan aplikasi web, banyak perusahaan yang menggunakan pengembangan dengan Laravel. Baik perusahaan kecil hingga perusahaan besar [22].

Laravel, Sublime Text, dan Visual Studio Code melayani tujuan yang berbeda dalam pengembangan web. Laravel adalah *framework* yang menyediakan struktur dan alat khusus untuk membangun aplikasi web dengan efisien dan aman. Sementara itu, Sublime Text dan Visual Studio *Code* adalah editor teks/kode yang kuat yang mendukung berbagai bahasa

pemrograman dan alat, tetapi tidak menyediakan fitur-fitur khusus yang ditawarkan oleh sebuah *framework* seperti Laravel. Oleh karena itu, jika tujuannya adalah mengembangkan aplikasi web dengan PHP, Laravel akan memberikan lebih banyak kelebihan dibandingkan dengan hanya menggunakan editor teks seperti Sublime Text atau Visual Studio Code.

10. Sensor MQ-135

Sensor adalah perangkat yang digunakan untuk mendeteksi perubahan besaran fisik. Perubahan fisik tersebut seperti tekanan, gaya, besaran listrik, cahaya, gerakan, kecepatan dan fenomena-fenomena lingkungan lainnya. Setelah mengamati terjadinya perubahan, *Input* yang terdeteksi tersebut akan dikonversi menjadi *Output* yang dapat dimengerti oleh manusia baik melalui perangkat sensor itu sendiri ataupun ditransmisikan secara elektronik melalui jaringan. Kemudian ditampilkan menjadi informasi yang bermanfaat bagi penggunanya. Sensor pada dasarnya dapat digolong sebagai *Transduser Input* karena dapat mengubah energi fisik seperti cahaya, tekanan, gerakan, suhu atau energi fisik lainnya menjadi sinyal listrik ataupun resistansi. Kemudian dikonversikan lagi ke tegangan atau sinyal listrik. Gambar 2.2 menunjukkan gambar sensor MQ-135.

Gambar 2. 2. Sensor MQ-135

Sumber: (https://ecadio.com) [23]

Pada penelitian dibuat menggunakan sebuah sensor berjenis MQ-135. MQ-135 *Air Quality Sensor* adalah sensor yang memonitor kualitas udara untuk mendeteksi gas O2 (Oksigen), gas ammonia (NH3), natrium-(di)oksida (NOx), alcohol / ethanol (C2H5OH), benzena (C6H6), karbondioksida (CO2), gas belerang / sulfurhidroksida (H2S) dan asap atau

gas-gas lainnya di udara. Sensor ini melaporkan hasil deteksi kualitas udara berupa nilai resistensi analog di pin keluarannya. Pin keluaran ini bisa disambungkan dengan pin ADC (*analog-to-digital converter*) di mikrokontroler/pin analog *input* Arduino. Cara kerja sensor MQ-135 dipengaruhi oleh gas yang terdeteksi yang mana akan mempengaruhi tahanan sensor. Ketika tahanan sensor semakin besar maka kepadatan gas yang terdeteksi kecil sehingga tegangan analog juga semakin kecil [24]. Sensor MQ-135 memiliki beberapa kelebihan dibandingkan dengan sensor udara lainnya, terutama dalam konteks deteksi gas dan polusi udara. Berikut adalah kelebihan utama dari sensor MQ-135:

a. MQ-135 sangat sensitif terhadap berbagai gas berbahaya seperti amonia (NH3), nitrogen oksida (NOx), alkohol, benzena, asap, dan karbon dioksida (CO2). Hal ini membuatnya ideal untuk aplikasi monitoring kualitas udara.
b. Sensor ini relatif murah dibandingkan dengan sensor udara lain yang mungkin menawarkan fungsi serupa, sehingga menjadikannya pilihan ekonomis untuk proyek DIY, sistem monitoring rumah, dan aplikasi industri skala kecil.
c. MQ-135 mampu mendeteksi konsentrasi gas dalam rentang yang luas, biasanya dari 10 ppm hingga 1000 ppm, memungkinkan penggunaannya dalam berbagai aplikasi lingkungan dan industri.
d. Sensor ini mudah diintegrasikan dengan mikrokontroler seperti Arduino, Raspberry Pi, dan *platform* IoT lainnya. Modul yang sudah tersedia di pasaran sering kali dilengkapi dengan pinout yang sederhana, sehingga memudahkan proses instalasi dan kalibrasi.
e. MQ-135 memiliki waktu respon yang cepat terhadap perubahan konsentrasi gas di sekitarnya, yang penting untuk aplikasi yang memerlukan monitoring *real-time*.
f. Sensor ini memiliki stabilitas jangka panjang yang baik dan daya tahan yang cukup tinggi, membuatnya andal untuk penggunaan dalam jangka waktu yang lama tanpa sering memerlukan kalibrasi ulang.

g. MQ-135 menyediakan keluaran analog yang proporsional dengan konsentrasi gas, serta keluaran digital yang bisa dikonfigurasi dengan ambang batas tertentu untuk memicu alarm atau tindakan otomatis lainnya.

Meskipun memiliki banyak kelebihan, penting juga untuk mempertimbangkan beberapa keterbatasan dari MQ-135. Seperti ketergantungan pada kondisi lingkungan (suhu dan kelembaban) dan kebutuhan kalibrasi berkala. Mungkin juga kurangnya akurasi yang sangat tinggi dibandingkan dengan sensor kelas atas yang lebih mahal. Namun, secara keseluruhan, MQ-135 adalah pilihan yang sangat baik untuk banyak aplikasi pengukuran kualitas udara berkat kombinasi dari keunggulan-keunggulan tersebut.

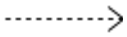
11. Unified Modeling Lengauge (UML)

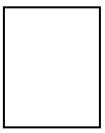
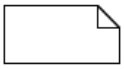
Diagram UML adalah pemodelan grafis standard untuk merancang dan menggambarkan *software system.* UML biasa dipakai untuk membantu pengembang perangkat lunak dalam mengkomunikasikan, merancang dan memahami *system* perangat lunak yang kompleks. UML menggunakan notas grafis untuk mempresentasikan konsep-konsep seperti *class*, *object,* dan relasi antara *object*, fungsi dan alur kerja sistem perangkat lunak. UML juga memungkinkan pengembang untuk memodelkan berbagai aspek sistem perangkat lunak, seperti analisis kebutuhan, perancangan arsitektur dan implementasi [24]. Ada beberapa contoh dari UML yang sering digunakan untuk perancangan system diantaranya:

a. *Use Case* Diagram

Use Case Diagram merupakan urutan interaksi yang memiliki keterkaitan antara sistem dan actor. *Use case* diagram dijalankan dengan cara menggambarkan tipe interaksi yang terjadi diantara user yang terlibat dalam sistem. Sebuah *use case* mempresentasikan sebuah interaksi antara aktor dengan sistem [26].

Tabel 2. 7. Simbol-simbol Pada *Use Case* Diagaram

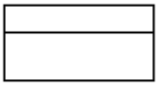
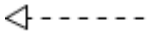
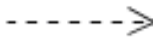
No	Gambar	Nama	Keterangan
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

b. *Class* Diagram

Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML. Menggambarkan dengan jelas struktur serta deskripsi *class, attribute*, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi. Diagram kelas ini sesuai jika diimplementasikan ke proyek yang menggunakan konsep *object-oriented*. Hal tersebut dikarenkan gambaran dari *class* diagram cukup mudah untuk digunakan [27].

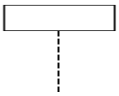
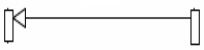
Tabel 2. 8. Simbol-simbol Pada *Class* Diagram

No	Gambar	Nama	Keterangan
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

c. *Squence* Diagram

Squence diagram digunakan untuk menggambarkan urutan dari interaksi antara *object* dalam *system* perangkat lunak. *Sequence* diagram bersifat dinamis. Diagram urutan ini adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu. Ini sangat berguna dalam merancang dan menganalisis sistem yang kompleks

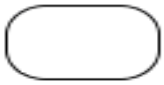
Tabel 2. 9. Simbol-simbol Pada *Squence* Diagram

No	Gambar	Nama	Keterangan
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.

d. *Activity* Diagram

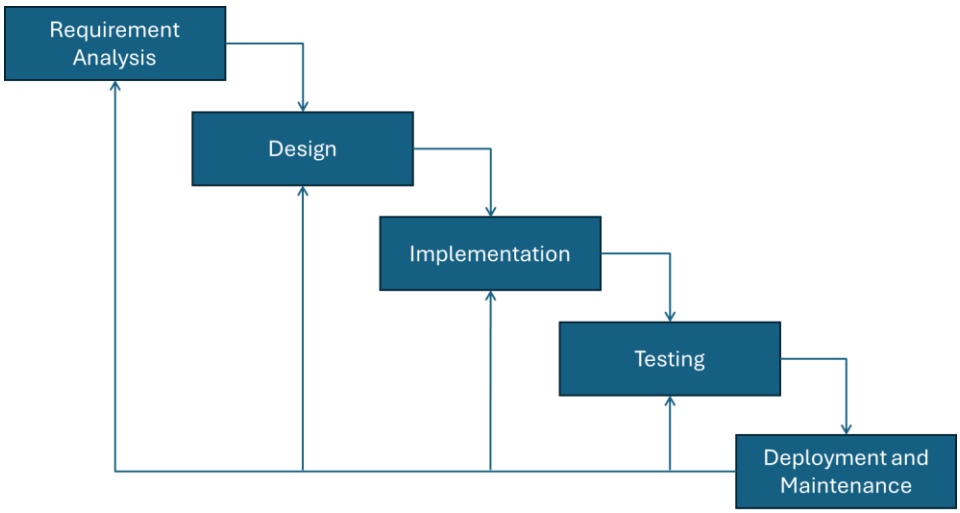
Digunakan untuk menggambarkan alur kerja *system software* atau proses bisnis yang terkait. Menurut Sukamto dan Shalahuddin diagram aktivitas atau *activity* diagram adalah menggambarkan aliran kerja atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktifitas menggambarkan aktifitas sistem bukan apa yang dilakukan oleh aktor. Dengan demikian diagram aktivitas atau *activity* diagram adalah menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem.

Tabel 2. 10. Simbol-simbol Pada Activity Diagram

No	Gambar	Nama	Keterangan
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

12. Metode *Waterfall*

Waterfall adalah salah satu metode pengembangan sistem perangkat lunak. Disebut *waterfall* (berarti air terjun) dikarenakan metode ini tahapan prosesnya mirip dengan air terjun yang bertingkat. Metode *waterfall* setiap tahapannya dikerjakan secara berurutan dari atas kebawah. Para developer perangkat lunak tentunya sering mengaplikasikan Metode *waterfall* dalam pekerjaannya. Alasannya dikarenakan metode ini memiliki risiko kecil, tidak memerlukan perubahan secara terus-menerus, gambaran produk sudah jelas. Istilah metode *waterfall* adalah salah satu pendekatan dalam pengembangan perangkat lunak yang paling umum digunakan. Metode ini memiliki keunggulan yakni proses pengembangan yang terstruktur dan terorganisir dengan baik. Kemudahan dalam pemahaman struktur hingga menghasilkan perangkat lunak dapat terdokumentasi dengan baik. Gambar 2.3 menunjukkan metode *waterfall.*



Gambar 2. 3. Metode *Waterfall*

Pada Gambar 2.3 diatas metode *waterfall* memiliki 5 tahapan diantara *requirement analysis, design, implementation, testing* dan *deployment and maintenance.* Pada penelitian ini metode *waterfall* menggunakan *looping* atau iterasi balik. *Looping* bukan fitur utama dalam model *waterfall* tradisional. Namun, penerapan konsep ini dalam praktik memungkinkan fleksibilitas yang lebih besar dan membantu dalam menangani dinamika pengembangan perangkat lunak yang kompleks. Metode ini juga memungkinkan kontrol yang ketat terhadap jadwal, dan biaya. Nantinya hal ini akan berpengaruh pada kualitas, serta memungkinkan penyelesaian satu tahap sebelum memulai lanjutannya [28].

13. *MySQL*

MySQL adalah salah satu jenis database server yang sangat terkenal. Kepopulerannya disebabkan Mysqlmenggunakan SQL sebagai bahasa dasar untuk mengakses databasenya. MySQL termasuk jenis RDBMS (*Relational Database Management System*). Pada MySQL, sebuah database mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom. Untuk mengelola database MySQL ada beberapa cara yaitu melalui prompt DOS (*tool command line*) [29]. Kelebihan MySQL dalam hal kinerja, keandalan, dan

dukungan komunitas yang kuat menjadikannya pilihan populer di kalangan pengembang dan administrator basis data.

14. CorelDRAW

CorelDRAW adalah aplikasi editor grafik vektor yang digunakan untuk menghasilkan gambar visual. Fokus utama aplikasi ini adalah pengeditan gambar, sehingga sering digunakan dalam bidang periklanan, desain visual, percetakan, dan bidang lain yang memerlukan format visualisasi. Beberapa kelebihan CorelDRAW dibandingkan aplikasi serupa meliputi: kemampuan mengolah garis dan warna dengan akurat, beragam jenis font untuk mendukung kreativitas dan imajinasi dalam pembuatan brosur, pamflet, sampul buku, dan lainnya. CorelDRAW juga menawarkan tingkat kejelasan dan spesifikasi warna yang mendetail, serta akurasi tinggi dalam mendesain, termasuk dalam hal garis, ketebalan garis, lengkungan, sudut, dan kerapatan garis. [30].

15. *White Box Testing*

White box testing atau dapat diartikan menjadi "pengujian kotak putih" adalah pengujian yang dilakukan untuk menguji perangkat lunak dengan cara menganalisa dan meneliti struktur internal dan kode dari perangkat lunak. Lain halnya dengan *blackbox testing* yang hanya melihat hasil input dan output dari perangkat lunak, pengujian *whitebox testing* berfokus pada aliran input dan output dari perangkat lunak [31]. Dalam melakukan pengujian, penguji atau *tester* harus memiliki pemahan *code* dari suatu program. Adapun teknik-teknik dalam pengujian *whitebox* antara lain:

a. *Cyclomatic Complexity*

Cyclomatic complexity adalah metrik pengujian yang digunakan untuk mengukur complexitas suatu program perangkat lunak. Ini adalah ukuran kuantitatif jalur independen dalam kode sumber program perangkat lunak. Rumus: V(G) = E - N + 2. Dimana E adalah jumlah tepi dan N adalah jumlah Node [32].

b. *Basis path testing*

Teknik pertama adalah *basis path testing*. Teknik bertujuan untuk mengukur kompleksitas kode program dan mendefinisikan alur yang dieksekusi.

c. *Branch coverage*

Kemudian ada *branch coverage*. Pengujian ini dirancang agar setiap branch code diuji setidaknya satu kali.

d. *Condition coverage*

Selanjutnya adalah teknik *condition coverage*, tujuannya untuk menguji seluruh kode agar menghasilkan nilai *TRUE* atau *FALSE*. Dengan begitu, tester dapat memastikan perangkat lunak dapat bekerja dan mengeluarkan output sesuai dengan input dari pengguna.

e. *Loop testing*

Ada juga teknik *loop testing*. Pengujian ini yang wajib dilakukan untuk menguji berbagai perulangan/*looping* yang ada dalam program, seperti *do-while*, *for*, dan *while*. Dalam pengujian ini kamu juga dapat memeriksa kondisi dari perulangan, apakah sudah berjalan dengan benar atau tidak.

f. *Multiple condition coverage*

Berikutnya adalah *multiple condition coverage*. Teknik ini dilakukan untuk menguji seluruh kombinasi dari kode yang mungkin digunakan dalam berbagai kondisi. Seluruh kombinasi harus diuji minimal satu kali, tujuannya untuk memastikan perangkat lunak agar berjalan dengan baik.

g. *Statement coverage*

Teknik terakhir adalah *statement coverage*. Teknik ini dilakukan minimal satu kali untuk menguji setiap statement dalam perangkat lunak. Dengan pengujian ini, kamu dapat mengetahui kode-kode yang error sehingga dapat segera memperbaikinya.

16. *Black Box Testing*

Black box testing merupakan metode pengujian perangkat lunak yang menguji fungsionalitas aplikasi. Pengujian dibangun di sekitar spesifikasi

dan persyaratan, yakni, apa yang seharusnya dilakukan aplikasi. Menggunakan deskripsi eksternal perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. pengujian ini dapat menjadi fungsional atau juga nonfungsional, walaupun biasanya fungsional. Perancang uji memilih *input* yang valid dan tidak valid dan menentukan *output* yang benar. Adapun teknik-teknik yang digunakan untuk menguji perangkat lunak dengan metode *blackbox testing* antara lain:

a. *All pair testing*

   Teknik *all pair* testing ini dikenal juga dengan *pairwise testing*. Pengujian ini digunakan untuk menguji semua kemungkinan kombinasi dari seluruh pasangan berdasarkan input parameternya.

b. *Boundary value analysis*

   Teknik ini berfokus pada pencarian error dari luar atau sisi dalam perangkat lunak.

c. *Cause-effect graph*

   Berikutnya adalah teknik *cause-effect graph*. Teknik pengujian ini menggunakan grafik sebagai patokannya. Grafik ini menggambarkan relasi antara efek dan penyebab dari error.

d. *Equivalence partitioning*

   Teknik ini bekerja dengan cara membagi data *input* dari beberapa perangkat lunak menjadi beberapa partisi data.

e. *Fuzzing*

   Fuzzing merupakan teknik pencarian bug dalam perangkat lunak dengan memasukan data yang tidak sempurna.

f. *Orthogonal array testing*

   Selanjutnya adalah *orthogonal array testing*. Teknik ini digunakan jika input berukuran kecil, akan tetapi cukup berat jika digunakan dalam skala yang besar.

g. *State transition*

   Terakhir adalah *state transition*. Teknik ini berguna untuk melakukan pengujian terhadap mesin dan navigasi dari UI dalam bentuk grafik.

17. *User Acceptence Testing* (UAT)

User Acceptance Testing (UAT) adalah tahap testing terakhir dan terpenting dari empat tahapan *testing software* yang umum dilakukan. Dalam tahapan ini, pengujian sistem dilakukan untuk menentukan apakah sistem telah memenuhi kebutuhan pengguna dan dapat mendukung semua skenario bisnis dan pengguna. UAT dilakukan oleh *client* dan *end-user*. Pengujian UAT ini erat kaitannya dengan pembayaran yang dilakukan kepada pengembang *software* [33]. Adapun metode umum yang digunakan dalam UAT:

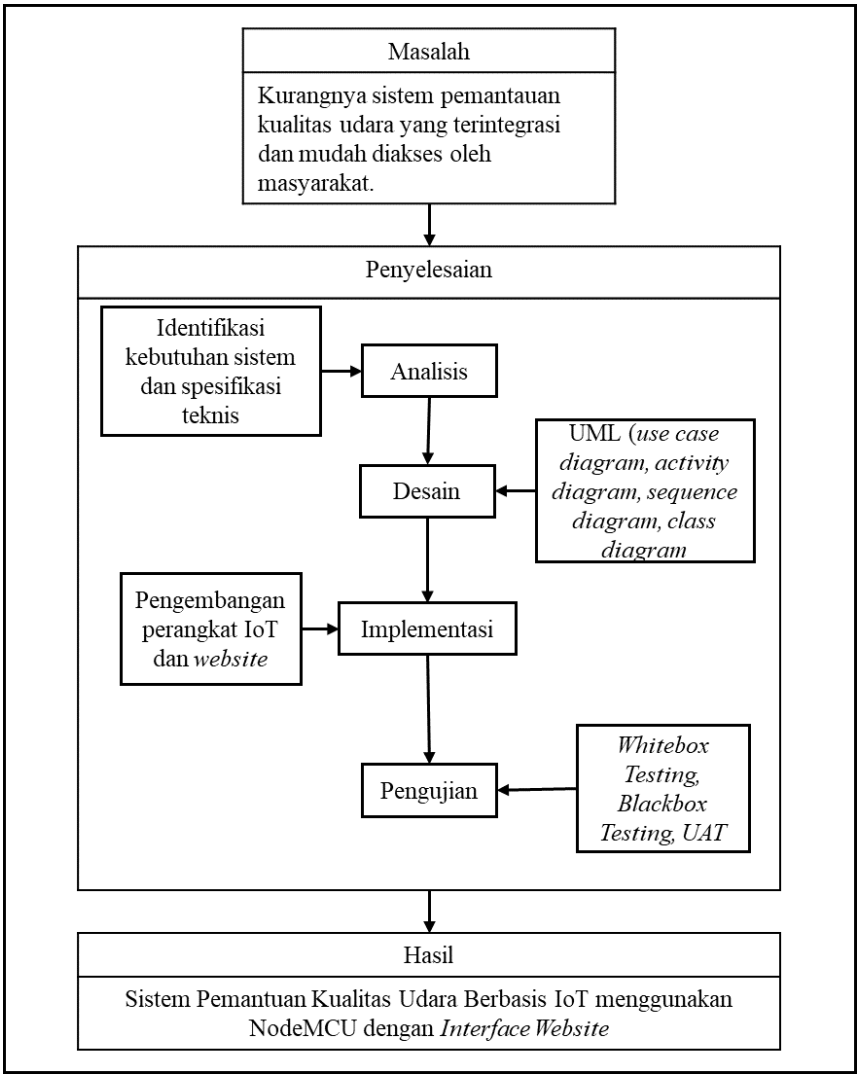
1. Persiapan
2. Pelaksanaan
3. Validasi dan Verifikasi
4. Pengambilan Keputusan
5. Penyelesaian
6. Umpan Balik dan Perbaikan

UAT memiliki memiliki 5 jenis metode dalam pengujiannya, diantaranya:

a. *Alpha Testing*: Dilakukan oleh internal tim sebelum melibatkan pengguna akhir.
b. *Beta Testing*: Dilakukan oleh pengguna akhir di lingkungan nyata.
c. *Black Box Testing*: Fokus pada fungsionalitas tanpa melihat kode internal.
d. *End-to-End Testing*: Menguji keseluruhan alur kerja aplikasi dari awal hingga akhir.

C. Kerangka Berpikir

Kerangka berpikir adalah alat penting dalam penelitian yang berfungsi sebagai panduan untuk mengidentifikasi dan menghubungkan berbagai konsep serta variabel yang relevan, memastikan penelitian berjalan dengan arah yang jelas dan terstruktur. Gambar 2.4 menunjukkan kerangka berpikir dalam membuat penelitian ini.



Gambar 2. 4. Kerangka Berpikir

BAB III

METODE PENELITIAN

A. Pendekatan Penelitian

Penulisan proposal skripsi ini menggunakan pendekatan *research and development. Research and Development* (R & D). Metode atau langkah ini untuk menciptakan produk baru atau mengembangkan dan menyempurnakan produk yang sudah ada dan digunakan untuk menguji keefektifan produk tersebut. Beberapa metode yang digunakan saat melakukan R&D, yaitu metode: deskriptif, evaluatif dan eksperimental. Tujuan dari penelitian ini adalah untuk menghasilkan produk yang akan digunakan dalam pendidikan melalui proses ilmiah yang diakhiri dengan tahap validasi. Perlu diketahui bahwa produk penelitian pengembangan tidak hanya tersedia dalam bentuk buku, film atau bahan pembelajaran lainnya, tetapi juga berupa proses, model pembelajaran atau metode pengajaran. Secara umum, kerja penelitian dan pengembangan juga bersifat siklis, sehingga produk pendidikan yang dihasilkan benar-benar bermanfaat dan memenuhi kebutuhan. Produk pelatihan ditingkatkan selama fase penelitian pengembangan untuk menghasilkan produk yang optimal. [34]

Adapun metode yang digunakan dalam penelitian ini adalah model waterfall, dikarenakan metode tersebut menggunakan pendekatan yang sistematis dan berurutan dalam membangun suatu sistem.

B. Fokus Penelitian

Penelitian ini berfokus pada pengembangan dan implementasi sistem pemantauan kualitas udara berbasis *Internet of Things* (IoT) dan *website*. Sistem ini dirancang khusus untuk kebutuhan perusahaan. Tujuan utamanya adalah menciptakan sistem yang efektif, efisien, dan mudah diakses. Tujuannya memungkinkan perusahaan untuk memantau dan menganalisis kualitas udara secara *real-time*. Sistem ini bertujuan untuk mendukung perusahaan dalam

upaya mitigasi polusi udara dan meningkatkan pemantauan lingkungan kerja, serta memastikan kepatuhan terhadap standar lingkungan yang berlaku.

C. Jenis dan Sumber Data

1. Data Primer

Data yang dikumpulkan langsung dari sumber aslinya melalui perangkat IoT yang digunakan dalam sistem pemantauan kualitas udara. Data yang dikumpulkan dari sensor kualitas udara yang mengukur berbagai parameter seperti mengukur partikel padat atau cair dalam udara yang berukuran kecil dan mengukur berbagai gas polutan seperti karbon monoksida (CO), karbon dioksida, asap dan gas-gas lain sebagainya.

2. Data Sekunder

Data sekunder adalah data yang telah dikumpulkan oleh pihak lain dan digunakan kembali dalam sistem pemantauan kualitas udara. Data pada penelitian ini diperoleh dari berbagai jurnal, buku ataupun *website* yang berkaitan dengan sistem pemantauan kualitas udara, penggunaan IoT mikrokontroler NodeMCU dan pengembangan *interface website*.

D. Teknik Pengumpulan Data

Dalam melakukan penelitian diperlukan metode dalam pengambilan data agar terlaksana dengan baik. Pada penelitian ini ada beberapa metode pengumpulan data antara lain:

1. Studi Literatur

Mengumpulkan informasi dari berbagai sumber tertulis yang relevan seperti buku, jurnal ilmiah, artikel, laporan penelitian. Selain itu juga mengumpulkan dokumentasi teknis yang berkaitan dengan sistem pemantauan kualitas udara, teknologi IoT, dan aplikasi web. Kemudian pada prosesnya dilakukaan telaah pustaka bertujuan untuk memahami konsep dan teknologi yang digunakan dalam sistem pemantauan kualitas udara. Ini juga mencakup studi tentang metode terbaik untuk visualisasi data dan penggunaan teknologi WebSocket untuk *real-time monitoring*.

2. Studi Kasus

Menganalisis studi kasus dari proyek atau sistem pemantauan kualitas udara yang telah ada. Fokus pada bagaimana sistem tersebut dirancang, diimplementasikan, dan dipelihara. Studi kasus ini melibatkan peninjauan dokumentasi proyek, laporan, atau artikel yang menguraikan pengembangan dan operasionalisasi sistem pemantauan kualitas udara di berbagai lokasi. Tujuannya adalah untuk mengidentifikasi praktik terbaik dan tantangan yang mungkin dihadapi.

E. Langkah Penelitian

Langkah penelitian adalah tahapan yang sistematis untui mengidentifikasi masalah, mengumpulkan dan menganalisi data dan juga menarik kesimpulan guna menguji hipotesis. Berikut adalah langkah-langkah yang tedapat dalam penelitian ini.

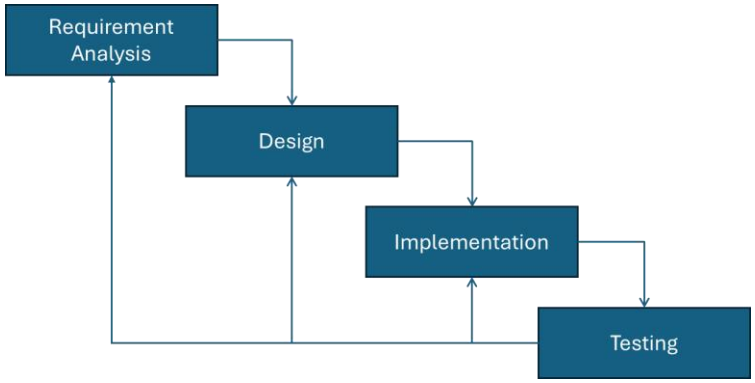
1. Identifikasi Masalah

Tingkat polusi udara saat ini semakin meningkat, terutama di kota-kota besar, akibat faktor-faktor seperti asap kendaraan bermotor dan aktivitas industri. Hal ini mengakibatkan kualitas udara yang buruk dan tidak sehat untuk kehidupan. Untuk mengatasi masalah ini, diperlukan sistem pemantauan kualitas udara yang dapat memberikan data secara *real-time*. Namun, tantangan utama yang dihadapi meliputi biaya pemasangan yang tinggi, cakupan pemantauan yang terbatas, dan kesulitan dalam memperoleh data yang akurat.

Penggunaan teknologi berbasis IoT dengan NodeMCU dan *website* sebagai antarmuka dapat menjadi solusi yang lebih terjangkau dan efisien. Oleh karena itu, dirancanglah sistem pemantauan kualitas udara berbasis IoT menggunakan NodeMCU dengan antarmuka *website* yang lebih efektif, akurat, dan mudah diakses oleh perusahaan. Sistem ini bertujuan untuk meningkatkan kesadaran tentang kualitas udara di lingkungan perusahaan dan membantu pengambilan keputusan yang lebih baik dalam menjaga kesehatan dan lingkungan kerja.

2. Penyelesaian Masalah

Penyelesaian masalah yang digunakan penulis pada penelitian ini menggunakan metode pengembangan *waterfall* bisa disebut juga *sequential linear* atau *classic cycle.* Penyelesaian masalah dilakukan dengan metode pengembangan *waterfall.* Pada penelitian ini metode *waterfall* yang digunakan hanya sampai tahap *testing,* tanpa *deployment and maintenance.* Gambar 3.1 menunjukkan metode *waterfall* tanpa tahap *deployment and maintenance*.



Gambar 3. 1. Metode *Waterfall* tanpa *Deployment and Maintenance*

Pada Gambar 3.1 diatas metode *waterfall* memiliki 4 tahapan, diantaranya:

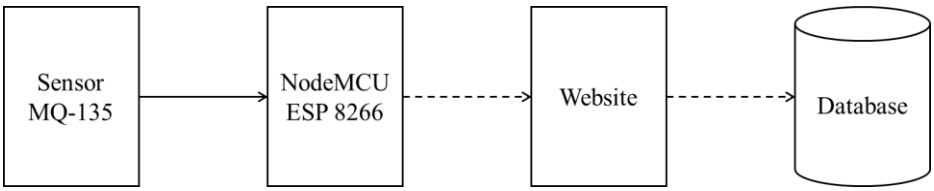
a. *Requirement Analysis*

Pada tahap pertama dalam *Requirement Analysis* (Analisis Kebutuhan) adalah pengumpulan kebutuhan. Pada tahap ini telah dihasilkan berberapa macam antara lain analisis kebutuhan sistem dan teknis. Analisis kebutuhan sistem meliputi kebutuhan perangkat keras (*hardware*) dan perangkat lunak (*software*). Analisis kebutuhan teknis meliputi analisis kebutuhan fungsional dan non-fungsional.

b. *Design*

Desain dilakukan sebelum proses *coding* dan perancangan alat dimulai. Proses ini bertujuan untuk memberikan gambaran tentang sesuatu yang harus dikerjakan dan bagaimana sistem dan alat dapat berjalan sesuai dengan keinginan. Untuk desain pada penelitian ini meliputi perancangan alat dan perancangan sistem. Pada perancangan alat menggunakan *software* CorelDRAW. Pada perancangan sistem

menggunakan *Unified Modelling Language* (UML) sebagai desain sistem yang akan dibuat. Perancangan sistem aplikasi yang penulis lakukan dengan menggunakan *tools* UML yang meliputi perancangan *use case* diagram, perancangan *activity* diagram, perancangan *class* diagram, perancangan *sequence* diagram. Untuk desain UML menggunakan *software* aplikasi Visual Paradigm. Gambar 3.2 menunjukkan desain sistem yang akan dibuat dalam bentuk blok diagram.



Gambar 3. 2. Blok Diagram Sistem

Pada Gambar 3.2 diatas menunjukkan proses sistem dari mulai *input* hingga *output.* Pada diagram ini menunjukkan garis solid dan garis putus-putus. Garis solid menunjukkan menunjukkan terhubung langsung melalui kabel pada hardware. Garis putus-putus menggambarkan hubungan sistem dengan nirkabel atau *wireless*. Sistem pemantauan kualitas udara ini terdiri dari sistem pengukuran, sistem *database*, dan sistem antarmuka data seperti yang terlihat pada blok diagram sistem. Sistem pengukuran dari Sensor MQ-135 dihubungkan dengan mikrokontroler NodeMCU ESP8266. Mikrokontroler ini berfungsi sebagai penerima data, pengelola sistem, dan pemroses data analog ke digital dengan selanjutnya terhubung dengan internet. Hasil pembacaan oleh NodeMCU ESP8266 kemudian dikirimkan ke web *server* untuk kemudian dicatat dan disimpan ke *database* MySQL. Sistem *database* yang telah dibuat digunakan dalam menerima data hasil pengukuran, dengan tabel yang tersusun pada MySQL di komputer *server*. Data yang tersimpan nantinya dapat diakses oleh pengguna melalui situs jaringan, sehingga informasi pada sistem *database* dapat diakses oleh pengguna melalui antarmuka *website*.

c. *Implementation*

Setelah tahap desain maka dilakukan *implementation* atau perancangan sistem sesuai dengan desain yang sudah dibuat. Untuk merancang *prototipe* IoT menggunakan mikrokontroller NodeMCU ESP8266, sensor MQ-135 untuk mengukur kualitas udara, dan dua lampu LED sebagai penunjuk kualitas udara. Setiap lampu LED dilengkapi dengan resistor 220Ω ± 5% untuk melindungi dari arus yang berlebihan. Kemudian perangkat-perangkat tersebut dirangkai diatas *breadboard*. Setelah rangkaian sudah terhubung dengan benar, kemudian perangkat IoT akan diprogram menggunakan *software* Arduino IDE. Untuk membuat antarmuka *website*, penulis memilih Laravel versi 10 dengan pengkodean menggunakan PHP versi 8.1. Antarmuka *website* ini dirancang untuk menampilkan data pemantauan kualitas udara secara *real-time*, yang diimplementasikan dengan *WebSocket* untuk efisiensi tampilan data yang *real-time*. Kemudian pada penampilan datanya dikombinasikan dengan *broadcasting* untuk penampilan data otomatis *refresh*.

d. *Testing*

Setelah implementasi selesai, selanjutnya *prototipe* dan aplikasi akan diuji. Pada penelitian ini pengujian akan dilakukan dalam 3 tahap yaitu *White Box Testing, Black Box Testing* dan *User Acceptance Testing* (UAT). Pengujian *white box* akan terfokus pada fungsi utama sistem pemantaun kualitas udara yaitu dari pengambilan data kualitas udara oleh perangkat *internet of things*, pengiriman data ke *server* sampai dengan penampilan data pada *website* sistem pemantauan. Pengujian *black box* akan dilakukan oleh 3 dosen Program Studi Informatika Universitas PGRI Semarang selain dosen pembimbing atas nama Ibu Noora Qotrun Nada, S.T., M.Eng., Ibu Nur Latifah Dwi MS, M.Kom., Bapak Ramadhan Renaldy, S.Kom., M.Kom. Untuk pengujian UAT akan dilakukan minimal 3 orang umum selain dari dosen program studi informatika dan dosen pembimbing.

3. Hasil

Setelah tahap penyelesaian masalah maka dibuat hasil Sistem Pemantauan Kualiatas Udara Berbasis *Internet of Things* menggunakan NodeMCU dengan *interface website* dengan tujuan untuk memantau atau memonitoring kualitas udara suatu tempat secara *real-time* dan menampilkan data hasil dari pemantaun perangkat IoT melalui *website* interaktif.

BAB IV

HASIL DAN PEMBAHASAN

A. Hasil

Metode yang digunakan dalam perancangan prototipe dan pengembangan sistem dalam penelitian ini yaitu dengan metode *waterfall.* Dalam penelitian ini metode *waterfall* yang digunakan hanya sampai tahap *testing* tanpa *deployment and maintenance*. Adapun tahapan dalam perancangan prototipe dan pembuatan sistem berdasarkan metode *waterfall* dapat dijabarkan sebagai berikut.

1. *Requirement Analysis*

Merupakan proses awal dalam pengembangan sistem dan perancangan prototipe. Kebutuhan dan persyaratan perancangan prototipe maupun pengembangan sistem yang akan dikembangan dikumpulkan, kemudian akan dianalisis secara mendalam. Langkah-langkah dalam *requirement analysis* pada metode *waterfall* meliputi:

a. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem merupakan tahap kritis dalam perancangan dan pengembangan sistem. Untuk memastikan keberhasilan teknis *project* dan kesesuaian sistem dengan kebutuhan pengguna. Adapun perangkat keras (*hardware*) dan perangkat lunak (*software*) yang dibuhkan dalam perancangan prototipe maupun pengembangan sistem ini adalah sebagai berikut.

1) Kebutuhan perangkat keras (*hardware*)

Perangkat keras yang digunakan untuk mengembangkan dan menguji coba terbagi menjadi beberapa bagian antara lain:

a) Laptop HP 14s-dk0127AU dengan spesifikasi *processor* AMD Ryzen 3 3200U, *memory* RAM 8 GB, SSD 256 GB, *hard drive* 1 TB, monitor 14” dengan resolusi 1920 * 1080 px.
b) NodeMCU ESP8266
c) Sensor MQ-135
d) *Breadboard*

e) Kabel *jumperwire*
f) Lampu LED
g) Resistor 220 ohm ± 5%

2) Kebutuhan perangkat lunak (*software*)

Perangkat lunak dalam penelitian ini menggunakan beberapa *software* diantaranya:

a) Arduino IDE
b) Laravel versi 10
c) Websocket
d) Xampp
e) Php MyAdmin
f) Visual Studio Code
g) CorelDRAW 2021 (64-bit)

b. Analisis Kebutuhan Fungsional

1) *User* dapat melihat tampilan *landing page* sistem pemantauan kualitas udara.
2) *User* dapat melihat berita terkait dengan kualitas udara.
3) *User* dapat masuk ke dalam halaman *login monitoring*.
4) Sistem mampu mengumpulkan data dari sensor-sensor IoT secara terus menerus dan tanpa intervensi manusia.
5) Sistem *website* mampu menampilkan data pengukuran kualitas udara dari IoT seperti kualitas udara keseluruhan, karbon dioksida (CO2) dan karbon monoksida (CO) secara *real-time.* Kemudian setiap data yang ditampilkan dapat memberikan keterangan kualitas udara sehat atau tidak sehat.
6) Data kualitas udara yang dikumpulkan dapat disimpan dan di *export* dalam bentuk *file excel* untuk analisis lebih lanjut.
7) Sistem dapat memberikan pemantuan secara langsung terhadap tingkat polutan tertentu yang melibihi batas yang ditetapkan kemudian memberikan peringatan dengan lampu LED.

c. Analisis Kebutuhan Non-Fungsional

1) Performa dalam sistem harus mampu mengakomodasi pengukuran dan pengiriman data secara *real-time* tanpa penundaan yang signifikan.
2) Data yang dikumpulkan dan disimpan harus diamankan untuk mencegah hak akses yang tidak sah atau modifikasi data.
3) *User Interface* (UI) harus dirancang dengan baik untuk memudahkan pengguna dalam mengakses dan memahami kualitas udara.

2. *Design*

Design untuk perancangan alat dan pengembangan sistem pemantauan kualitas udara ini bertujuan untuk memberikan panduan bagi penulis sebelum melakukan implementasi. Tahap desain bertujuan untuk mempermudah merancang alat dan membuat sistem sesuai dengan yang diinginkan. Adapun desain yang dibuat penulis dalam merancang dan mengembangkan sistem adalah sebagai berikut:

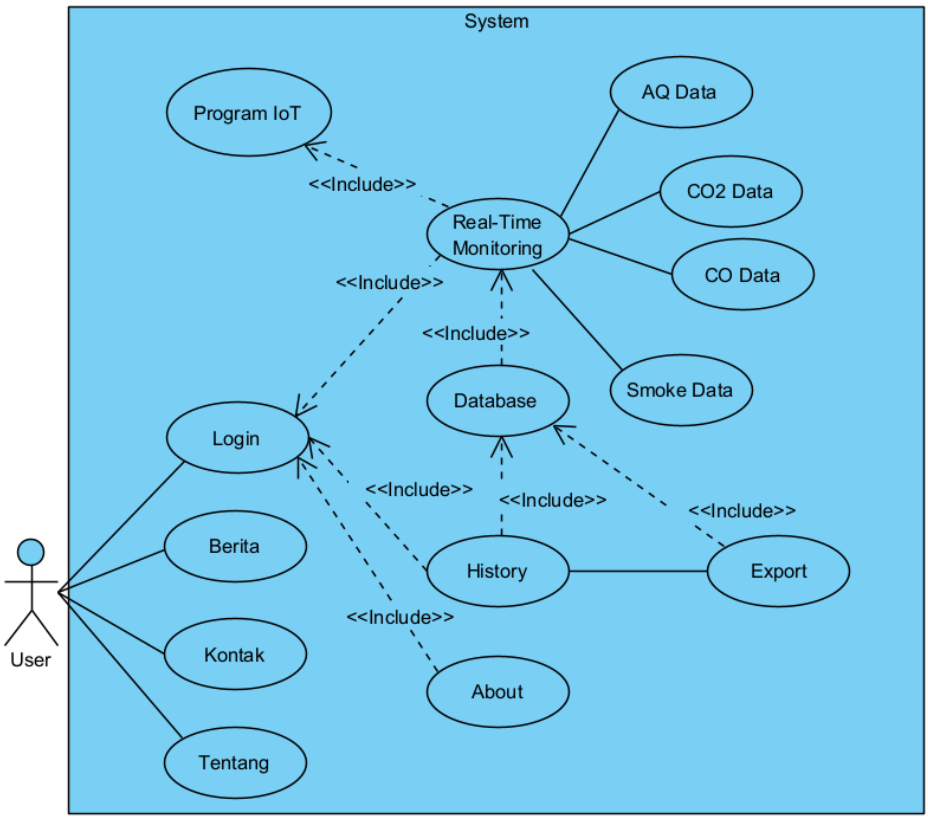
a. *Design System*

Perancangan dan pengembangan sistem aplikasi yang penulis lakukan dengan menggunakan *tools Unified Modelling Language* (UML). Ada beberapa *tools* UML yang digunakan penulis diantanya sebagai berikut.

1) *Use Case* Diagram

Use case diagram merupakan representasi visual dari fungsionalitas sistem yang menunjukkan interaksi antara sistem dengan pengguna atau sistem dengan entitas lain seperti sistem lain, perangkat keras, atau sistem lainnya. Diagram ini menggunakan *use case* (skenario fungsional) untuk menggambarkan berbagai kemungkinan interaksi antara sistem dan aktor yang terlibat. Gambar 4.1 menunjukkan gambar *use case* diagram sistem yang dibuat penulis

untuk merancang dan mengembangkan sistem pemantauan kualitas udara.



Gambar 4. 1. *Use Case* Diagram

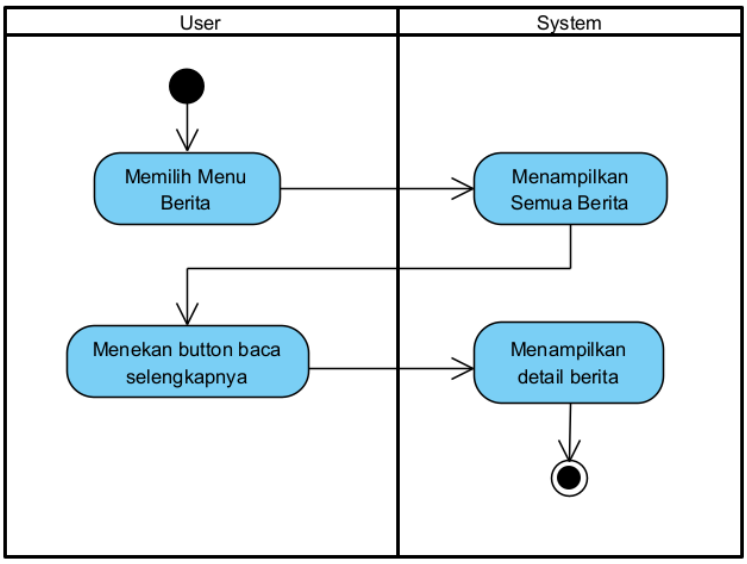
Pada Gambar 4.1 diatas, hanya terdapat 1 aktor yaitu *user*. Pada *user* dapat melakukan aksi diantaranya *login*, berita, kontak dan tentang. *Login* untuk melihat pemantauan kualitas udara. Aksi berita untuk menampilkan halaman berita. Aksi kontak untuk menampilkan halaman kontak. Kemudian ada aksi tentang untuk menampilkan halaman tentang. Pada *User* setelah melakukan aksi *login* maka *sser* dapat masuk pada aksi halaman *real-time monitoring* untuk melihat data kualitas udara saat ini. *User* juga dapat melakukan aksi *history* untuk melihat kualitas udara sebelumnya dan didalam *history user* juga dapat melakukan aksi *export* data. Kemudian aksi *about* untuk melihat halaman *about*. Pada *real-time monitoring* terdapat beberapa aksi diantaranya: AQ data (melihat data kualitas udara), CO2 (melihat data Karbon Dioksida), CO (melihat data Karbon Monoksida) dan

Smoke (melihat data kualitas asap). Pada setiap data kualitas udara yang ada di *real-time monitoring* mengambil dari program IoT.

2) *Activity* Diagram

Activity diagram atau diagram aktivitas dalam konteks sistem pemantauan kualitas udara membantu dalam memvisualisasikan proses yang kompleks menjadi serangkaian langkah yang terstruktur. Diagram aktivitas juga memudahkan pemahaman tentang bagaimana sistem bekerja dan berinteraksi dengan lingkungannya.

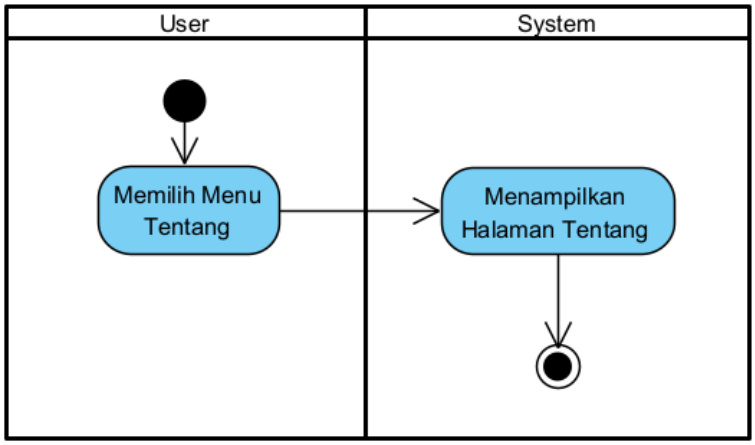
a) *Activity* Diagram Halaman Berita



Gambar 4. 2. *Activity* Diagram Halaman Berita

Pada Gambar 4.2 diatas, menunujukkan diagram aktivitas pada menu berita. Gambar diatas dapat dijelaskan *user* harus memilih menu berita terlebih dahulu. Kemudian *system* akan menampilkan semua berita yang berikaitan dengan kualitas udara. Jika *user* ingin melihat detail berita maka *user* harus menekan *button* baca selengkapnya untuk, maka sistem akan menampilkan halaman detail berita.

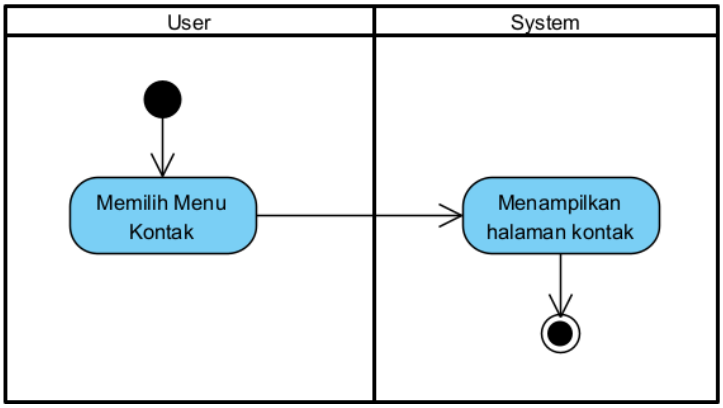
b) *Activity* Diagram Halaman Tentang



Gambar 4. 3. *Activity* Diagram Halaman Tentang

Pada Gambar 4.3 menunjukkan diagram aktivitas untuk melihat halaman tentang. Gambar diatas dapat dijelaskan setelah *user* memilih *button* tentang, maka *system* akan menampilkan halaman tentang.

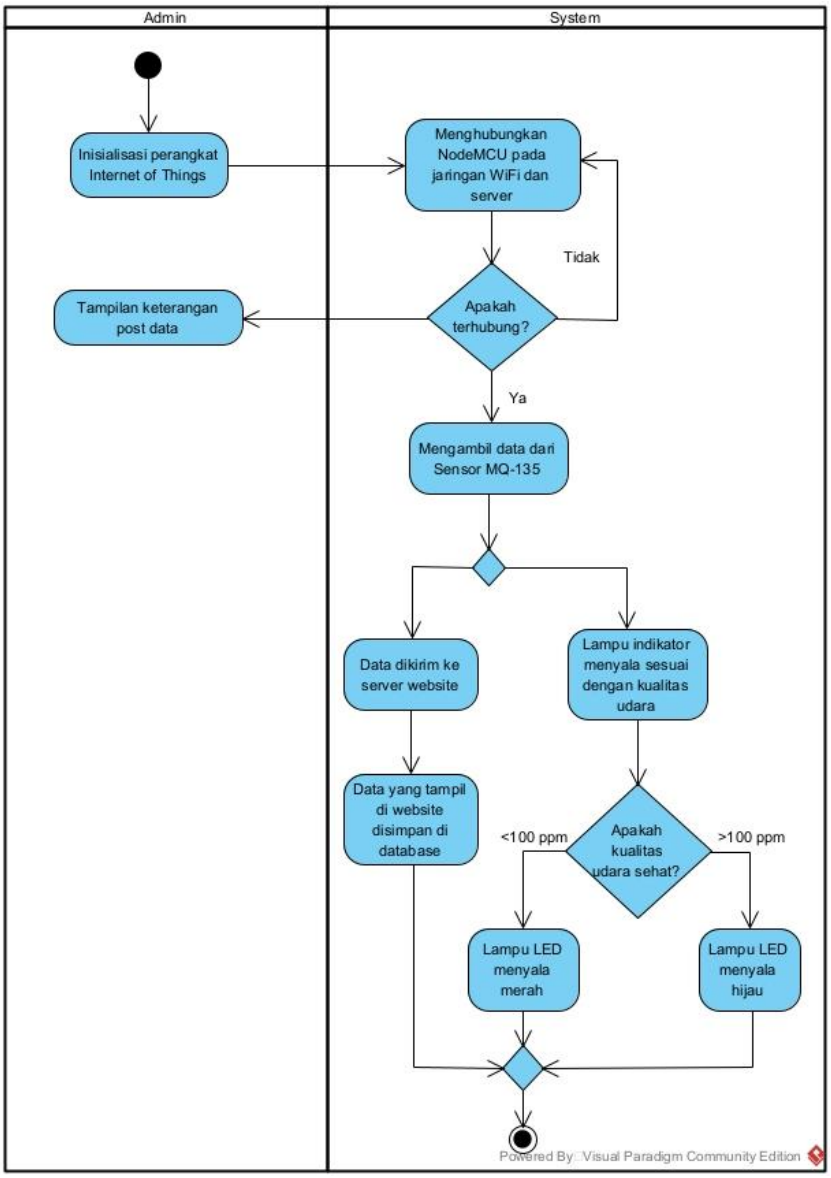
c) *Activity* Diagram Halaman Kontak



Gambar 4. 4. *Activity* Diagram Halaman Kontak

Pada Gambar 4.4 diatas, menunjukkan diagram aktivitas untuk menampilkan halaman kontak. Gambar tersebut dapat dijelaskan bahwa setelah *user* memilih menu kontak, maka *system* akan menampilkan halaman kontak.

d) *Activity* Diagram Pengambilan Nilai Kualitas Udara

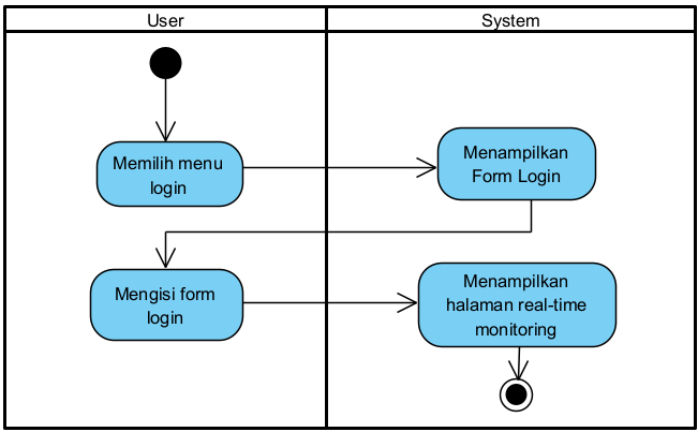


Gambar 4. 5. *Activity* Diagram Pengambilan Nilai Kualitas Udara

Pada Gambar 4.5 diatas, menunjukkan langkah-langkah dalam pengambilan nilai kualitas udara agar bisa tampil kedalam *website.* Langkah awal Admin melakukan inisialisasi terhadap perangkat IoT. Kemudian *System* akan langsung mencoba menghubungkan NodeMCU dengan jaringan WiFi yang sama dengan perangkat laptop. Jika NodeMCU tidak terhubung dengan jaringan WiFi, maka *system* akan mengulangi proses

tersebut sampai terhubung. Kemudian terdapat keterangan *post* data, jika terhubung maka keterangan pada serial monitor yang terdapat di program Arduino akan muncul kalimat *post* data berhasil. Jika muncul kalimat *error sending post* menandakan terdapat *error* yang menyebabkan data tidak bisa dikirimkan ke *server website*. Setelah terhubung maka *system* akan melakukan pengambilan data nilai kualitas udara dari Sensor MQ-135. Kemudian data kualitas udara diolah untuk langsung dikirimkan ke *server website* untuk ditampilkan dan sebagai acuan untuk menyalakan lampu indikator. Setelah data kualitas udara tampil di *website*, data tersebut langsung dicatat untuk disimpan di *database* MySQL. Pada lampu indikator jika data yang diterima menunjukkan kualitas udara baik yaitu <100 ppm, maka lampu LED menyala warna hijau. Jika data kualitas udara yang diterima menunjukkan kualitas udara buruk yaitu >100 ppm, maka lampu LED menyala warna merah. Itu menandakan semakin tinggi nilai kualitas udara maka kualitas udara tersebut buruk. Dan jika semakin rendah nilai kualitas udara maka menandakan kualitas udara tersebut baik.

e) *Activity* Diagram Halaman *Login*

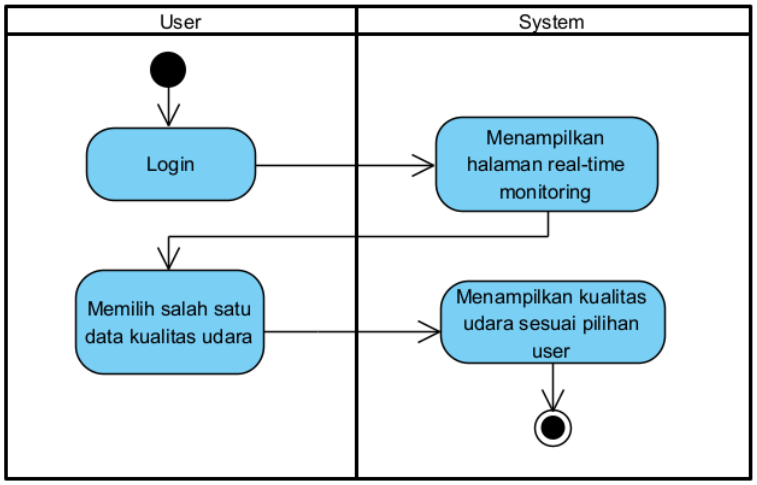


Gambar 4. 6. *Activity* Diagram Halaman *Login*

Pada Gambar 4.6 menunjukkan diagram aktivitas halaman *login.* Gambar diatas dapat dijelaskan setelah *user* memilih menu

login dan selanjutnya *system* menampilkan *form login* Selanjutnya jika *user* mengisi *form* tersebut dengan benar maka sistem akan menampilkan halaman *real-time monitoring*.

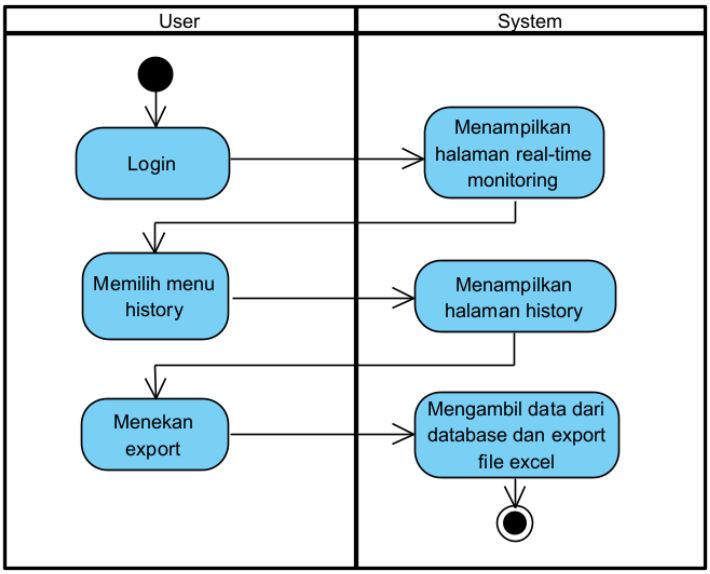
f) *Activity* Diagram Halaman *Real-Time Monitoring*



Gambar 4. 7. *Activity* Diagram Halaman *Real-Time Monitoring*

Pada Gambar 4.7 diatas, menunjukkan langkah-langkah diagram aktivitas pada untuk menampilkan halaman *real-time monitoring* pada *website* pemantauan kualitas udara. Langkah awal *user* melakukan *login* terlebih dahulu, setelah berhasil melakukan *login*, maka *system* akan menampilkan halaman *real-time monitoring.* Kemudian jika ingin melihat kualitas udara lain maka *User* memilih salah satu data kualitas udara, maka sistem akan menampilkan kualitas udara sesuai dengan permintaan *user*.

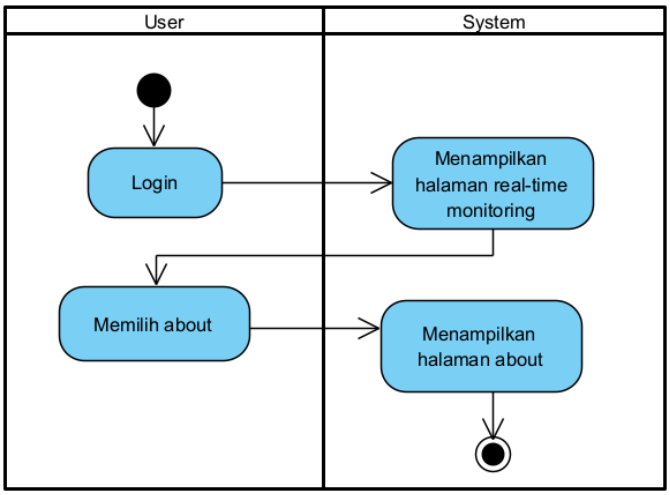
g) *Activity* Diagram Halaman *History*



Gambar 4. 8. *Activity* Diagram Halaman *History*

Pada Gambar 4.8 diatas, menunjukkan diagram aktivitas untuk halaman *history.* Setelah *user* melakukan *login*, maka *system* akan menampilkan halaman *real-time monitoring*. Selanjutnya *user* memilih menu *history*. Setelah *user* memilih menu *history*, maka *system* akan menampilkan halaman *history*. Didalam halaman *history* terdapat *button export*. Jika *user* menekan *button export* maka *system* akan mengambil data dari *database* dan mendownload data dalam bentuk file excel.

h) *Activity* Diagram Halaman *About*



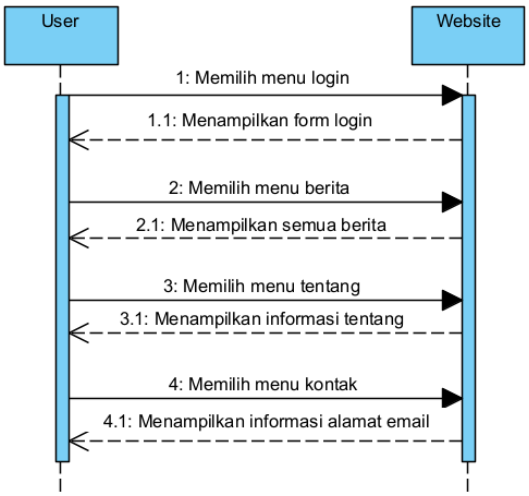
Gambar 4. 9. *Activity* Diagram Halaman *About*

Pada Gambar 4.9 menunjukkan diagram aktivitas halaman *about*. Gambar diatas dapat dijelaskan setelah *user* berhasil melakukan *login* maka *system* akan menampilkan halaman *real-time monitoring*. Kemudian *user* memilih menu *about*, maka *system* akan halaman *about.*

3) *Sequence* Diagram

Sequence diagram atau diagram urutan untuk pemantauan kualitas udara berbasis IoT dan web mengilustrasikan cara komponen-komponen sistem saling berinteraksi. Melalui diagram ini juga dapat dipahami bagaimana data dikumpulkan, diolah, dan dimanfaatkan. Data tersebut digunakan untuk memberikan informasi yang berguna kepada pengguna dalam konteks pemantauan kualitas udara dengan menggunakan teknologi *internet of things* (IoT) dan *website*.

a) *Sequence* Diagram Pada *Landing Page*

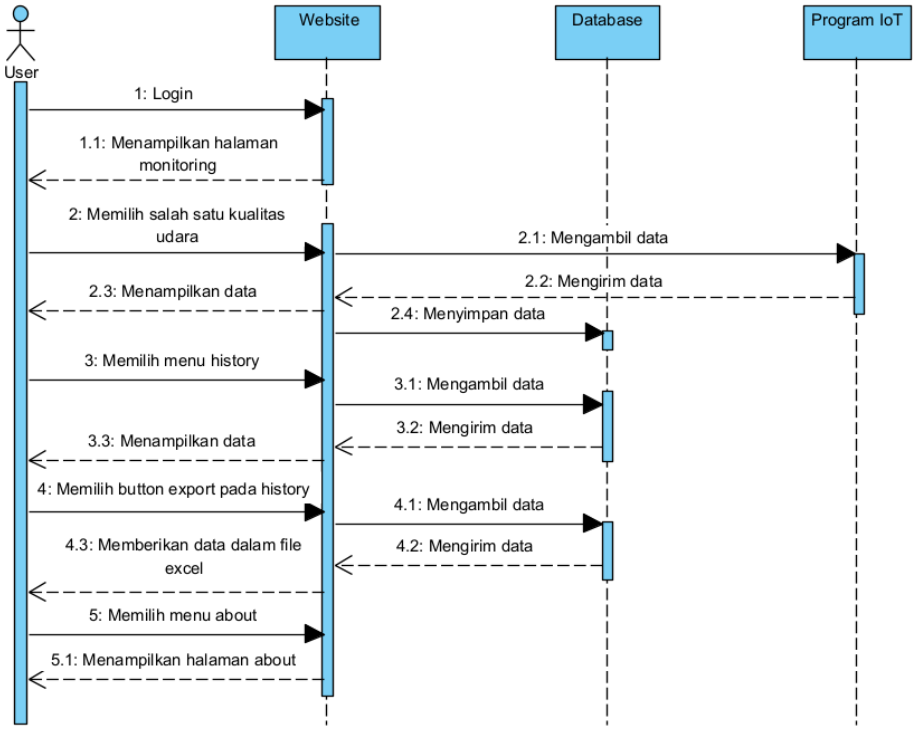


Gambar 4. 10. *Squence* Diagram Pada *Landing Page*

Pada Gambar 4.10 diatas, menunjukkan *sequence* diagram yang ada pada *landing page* di *website.* Pada aksi 1 *user* memilih menu *login*, kemudian *website* akan menampilkan *form login*. Pada aksi 2 *user* memilih menu berita, kemudian sistem akan menampilkan halaman berita. Pada aksi 3 *user* memilih menu tentang, kemudian *website* akan menampilkan halaman tentang.

Pada aksi 4 *user* memilih menu kontak, kemudian *website* akan menampilkan halaman kontak.

b) *Sequence* Diagram Pada *Website Real-Time Monitoring*



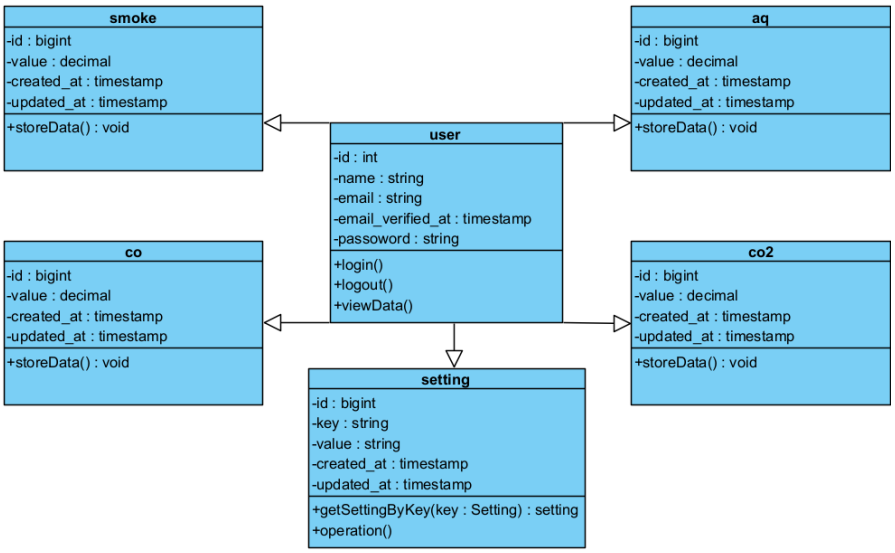
Gambar 4. 11. *Sequence* Diagram Pada *Website Real-Time Monitoring*

Pada Gambar 4.11 diatas, menunjukkan *sequence* diagram pada *website real-time monitoring*. Pada aksi yang ke 1 *user* harus *login* untuk dapat mengakases menu lainnya. Setelah *login* berhasil maka *website* akan menampilkan halaman *real-time monitoring*. Pada aksi 2 *user* memilih salah satu kualitas udara pad *website*, kemudian *website* mengambil data dari program IoT. Setelah itu program IoT mengirimkan data ke *website* dan *website* menampikan data ke *user.* Pada *website* juga menyimpan data ke *database*. Pada aksi 3 *user* memilih menu *history* pada *website*, kemudian *website* mengambil data dari *database* dan mengirimkan ke *website* Setelah itu *website* menampilkan data ke *user.* Pada aksi 4 *user* menekan *button export* didalam menu *history* di *website*. Kemudian *website* mengambil data dari

database dan mengirimkan ke *website*. Setelah itu *website* memberikan data ke *user* dalam bentuk *file excel*. Pada aksi 5 *user* memilih menu *about* pada *website* dan kemudian *website* menampilkan halaman *about* kepada *user*.

4) *Class* Diagram

Class diagram sistem pemantauan kualitas udara berbasis IoT dan web mengilustrasikan struktur serta hubungan antara komponen-komponen inti dalam sistem tersebut. Komponen-komponen yang terdapat dalam diagram tersebut memberikan gambaran tentang bagaimana sistem berinteraksi untuk mengumpulkan, menyimpan, dan menampilkan data kualitas udara berbasis IoT dan web. Gambar 4.13 menampilkan bagaimana NodeMCU digunakan dalam sistem untuk berkomunikasi dengan sensor-sensor dan antarmuka *website*.

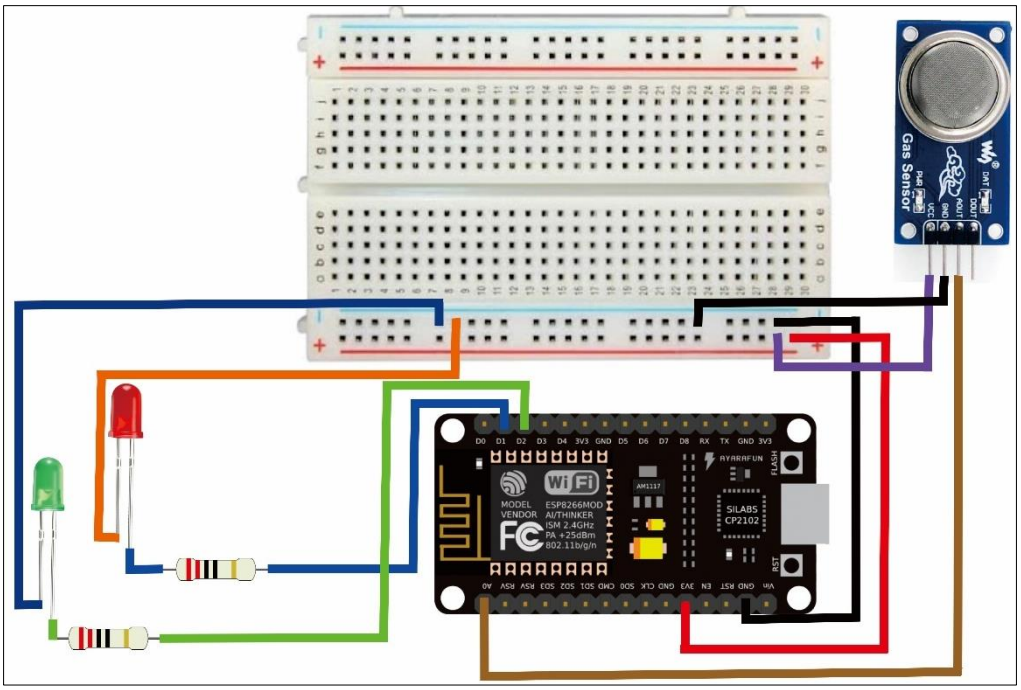


Gambar 4. 12. *Class* Diagram

Diagram ini menghubungkan berbagai entitas (kelas) yang terlibat dalam sistem pemantauan kualitas udara berbasis IoT dan website. User berinteraksi dengan berbagai jenis data kualitas udara (AQ, CO2, CO, Smoke), serta memiliki kemampuan untuk mengonfigurasi pengaturan melalui Setting.

b. *Design* Alat

Design alat menggunakan *software* CorelDRAW untuk membuat rancangan alat yang akan dibuat. Penelitian ini merancang sebuah alat yang dapat memantau kualitas udara seperti karbon dioksida, karbon moniksida dan asap untuk mengukur kadar polusi udara dengan menggunakan *Internet of Things.* Pada perangkaian alat ini menggunakan mikrokontroler NodeMCU ESP8266 yang dirangkai dengan Sensor MQ-135. Sensor MQ-135 berfungsi mendeteksi gas karbon dioksida (CO^2), gas karbon monoksida, asap dan kualitas udara keseluruhan. Alat ini akan mengirimkan informasi tentang data kualitas udara kepada *user* melalui *website*. Gambar 4.13 menunjukkan rancangan alat yang dibuat.



Gambar 4. 13. Rancangan Alat

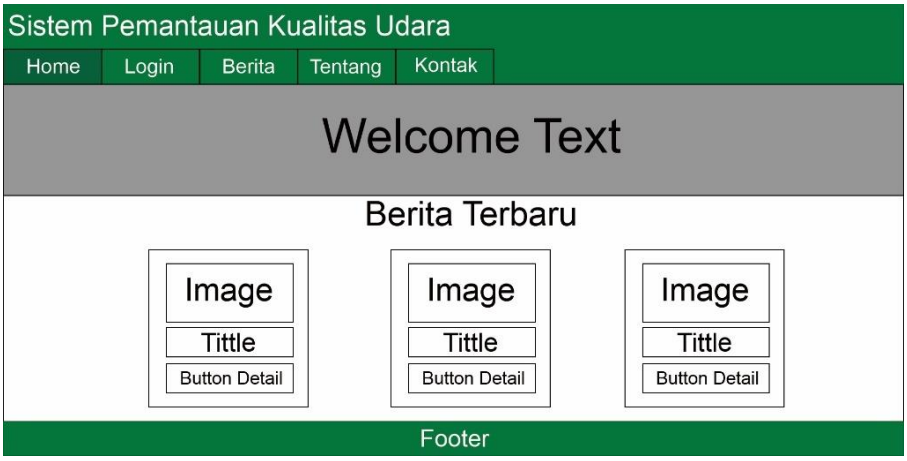
Pada Gambar 4.13 diatas terdapat instalasi kabel untuk menghubungkan satu komponen dengan komponen lainnya. Sensor MQ-135 dihubungkan dengan NodeMCU ESP8266 menggunakan pin A0 (analog *input* 0). NodeMCU ESP8266 adalah mikrokontroler berbasis WiFi yang berfungsi untuk mengirimkan data. Sensor MQ-135 juga biasanya memerlukan pemanasan sebelum penggunaan, jadi pastikan

untuk memberikan waktu pemanasan yang cukup sebelum membaca data. Menghubungkan kedua LED (merah dan hijau) ke pin digital NodeMCU (misalnya D1 dan D2). Lampu LED merah berfungsi sebagai indikator jika kondisi udara buruk. Sedangkan lampu LED hijau berfungsi sebagai indikator jika udara baik. Resistor 220 Ω(*ohm*) ke masing-masing kaki anoda LED. Kaki katoda LED dihubungkan ke *ground* NodeMCU. Resistor berfungsi untuk melindungi LED dari arus yang berlebihan.

c. *Design User Interface* (UI)

Desain UI untuk sistem monitoring kualitas udara berbasis IoT dan web harus fokus pada kemudahan penggunaan. Visualisasi data dibuat efektif dan mampu memberikan informasi yang relevan serta tepat waktu kepada pengguna. Dengan menggunakan prinsip-prinsip desain yang baik, pengguna akan dapat dengan mudah memantau dan mengelola kualitas udara di sekitar mereka secara efisien. Berikut adalah desain UI tampilan *website* yang dibuat.

1) *Design Landing Page*



Gambar 4. 14. *Design User Interface Landing Page*

Pada Gambar 4.14 diatas menunjukkan desain UI pada *landing page*. Gambar diatas dapat dijalaskan tampilan yang akan dibuat untuk *landing page* terdapat beberapa tampilan, diantaranya *header* dan *footer*. Pada *header* terdapat nama *website* sistem pemantauan kualitas udara dan berberapa *button.* Jika salah satu menu aktif maka

button akan berubah warna lebih gelap mengikuti menu yang aktif. Contohnya pada desain UI diatas aktif menu *Home*, maka *button* berubah warna menjadi lebih gelap dibandingkan *button* lainnya. Nama *website* dan *button* pada *header* nantinya akan di fix, supaya jika *user* melakukan pengguliran *button* dan nama *website* akan tetap tampil. Untuk *footer* nantinya tidak akan di fix. Desain UI diatas juga menjelaskan bahwa nanti *website* yang dibuat dapat menampilkan *welcome text* dan dibawahnya terdapat tampilan 3 (tiga) berita terbaru terkait dengan kualitas udara. Kemudian dibawah tampilan berita terbaru nantinya akan ditampilan tentang sistem aplikasi yang dibuat dan kontak jika terjadi kendala ataupun ingin memberikan masukan untuk sistem pada *website.*

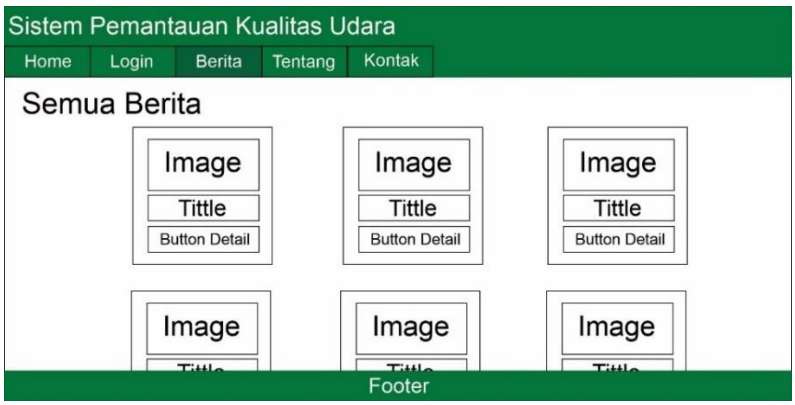
2) *Design User Interface* (UI) *Login*

Gambar 4. 15. *Design User Interface Login*

Pada Gambar 4.15 menunjukkan desain UI halaman *login website monitoring air quality.* Gambar desain UI diatas dapat dijelaskan terdapat *header* dan *footer* yang sama seperti pada tampilan *home* / *landing page* yang terdapat nama dan beberapa *button*. *Button* yang berubah warna menjadi gelap mengikuti menu yang aktif. Pada desain UI *login* juga terdapat nama *website* Sistem Pemantauan Kualitas Udara. *Website* yang digambarkan pada desain UI diatas mempunyai fitur diantaranya kolom untuk mengisi username dan

password untuk masuk ke dalam *website*. Terdapat juga tombol *login* untuk melakukan validasi *username* dan *password* yang sudah diisi.

3) *Design User Interface* (UI) Menu Berita Pada *Landing Page*



Gambar 4. 16. *Design User Interface* Menu Berita Pada *Landing Page*

Pada Gambar 4.16 diatas adalah desain UI pada menu berita. Gambar desain UI diatas dapat dijelaskan terdapat tampilan *header* dan *footer* yang sama seperti pada *landing page.* Untuk *header* di fix (tidak hilang jika *user* melakukan pengguliran), sedangkan pada *footer* tidak di fix. Terdapat juga *button* yang berubah menjadi gelap menyesuaikan dengan menu yang aktif. Pada menu berita nantinya akan tampil semua berita yang ada dan semuanya berkaitan dengan kualitas udara. Berita ditampilkan 3 (tiga) sejajar. Setiap berita terdapat gambar, judul dan juga *button* detail untuk melihat informasi berita secara lengkap.

4) *Design User Interface* (UI) Menu Detail Berita

Gambar 4. 17. *Design User Interface* Menu Detail Berita

Pada Gambar 4.17 menunjukkan desain UI menu untuk melihat detail berita. Gambar diatas dapat dijelaskan bahwa tampilan pada detail berita terdapat *header* dan *footer* yang sama seperti pada menu *landing page* / *home*. Untuk *header* di fix (tidak hilang jika *user* melakukan pengguliran), sedangkan pada *footer* tidak di fix. Pada halaman detail berita nantinya akan diberikan informasi terkait berita dan terdapat juga gambar yang berkaitan dengan berita. Disamping informasi detail berita juga menampilkan berita lainnya untuk memudahkan *user* jika ingin melihat berita lainnya tanpa harus kembali ke halaman semua berita.

5) *Design User Interface* (UI) Menu Tentang Pada *Landing Page*

Gambar 4. 18. *Design User Interface* Menu Tentang Pada *Landing Page*

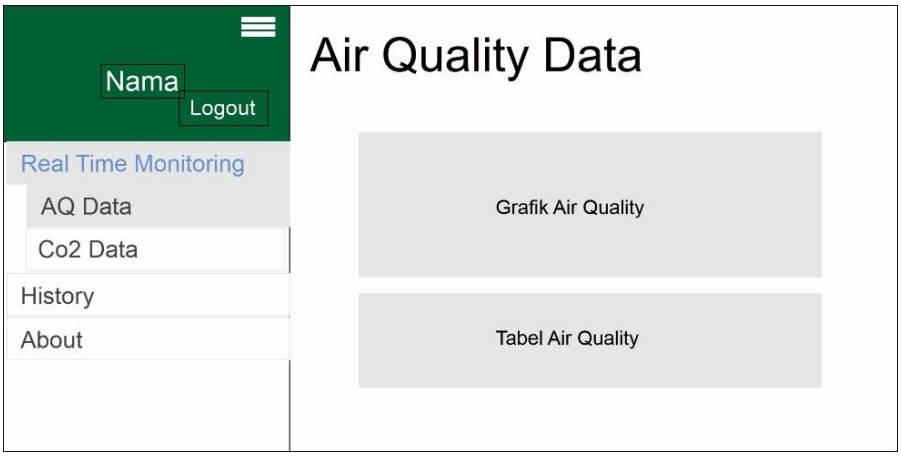
Pada Gambar 4.18 diatas menunjukkan desain UI yang ada pada menu tentang. Gambar diatas dapat dijelaskan pada menu tentang terdapat *header* dan *footer* yang sama seperti pada halaman *landing page*. Untuk *header* di fix (tidak hilang jika *user* melakukan pengguliran), sedangkan pada *footer* tidak di fix. Tampilan *button* tentang berubah warna menjadi lebih gelap dibandingkan *button* lain, menyesuaikan dengan menu yang aktif. Pada halaman tentang terdapat informasi terkait dengan *website* sistem pemantauan kualitas udara.

6) *Design User Interface* (UI) Menu Kontak Pada *Landing Page*

Gambar 4. 19. *Design User Interface* Menu Kontak Pada *Landing Page*

Pada Gambar 4.19 diatas menunjukkan desain UI menu kontak. Gambar diatas dapat dijelaskan terdapat *header* dan *footer* yang sama seperti pada halaman awal / *landing page*. Untuk *header* di fix (tidak hilang jika *user* melakukan pengguliran), sedangkan pada *footer* tidak di fix. Tampilan *button* kontak juga berubah warna menjadi lebih gelap dibandingkan *button* lain, menyesuaikan dengan menu yang aktif. Pada halaman kontak terdapat informasi alamat email dari *developer* guna mengirimkan masukan ataupun kendala terkait dengan sistem aplikasi.

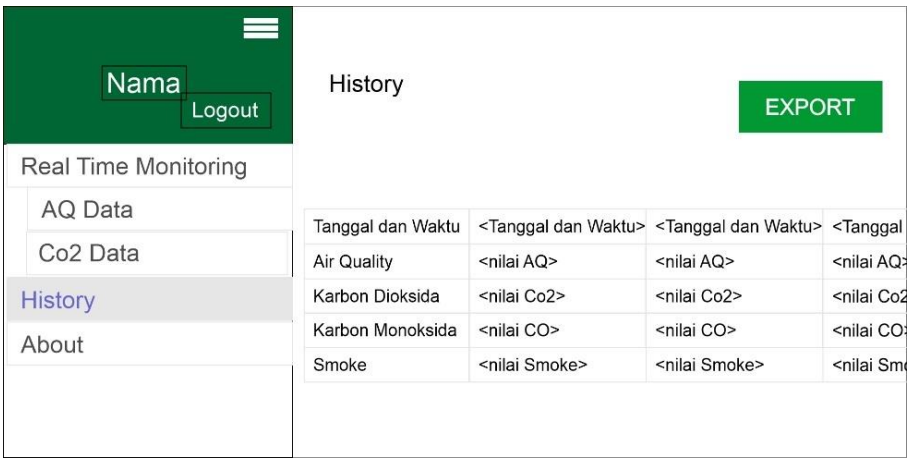
7) *Design User Interface* (UI) Menu *Real-Time Monitoring*



Gambar 4. 20. *Design User Interface Real-Time Monitoring*

Pada Gambar 4.20 menunjukkan desan UI pada menu *real-time monitoring.* Gambar diatas dapat dijelaskan *button* pada menu *real-time monitoring* dibuat dengan gaya *drop-down*. Jika ditekan maka akan muncul beberapa menu terkait dengan data kualias udara. Apabila ditekan salah satu menu data kualitas udara, maka akan tampil data kualitas udara berbentuk grafik, angka dan tabel sesuai dengan menu yang ditekan. Misalnya jika ditekan menu AQ Data, maka akan tampil data *air quality* dan *button* AQ Data berubah warna menjadi lebih gelap. Pada *drop-down* menu *real-time monitoring* terdapat beberapa menu diantaranya AQ Data, CO^2 Data, CO Data, dan *Smoke* Data. Pada tampilan desain UI diatas jika nama ditekan maka akan muncuul *button* keluar dari *website* sistem pemantaun.

8) *Design User Interface* (UI) Menu *History*



Gambar 4. 21. *Design User Interface* Menu *History*

Pada Gambar 4.21 diatas menunjukkan desain UI pada menu *history*. Desain UI diatas dapat dilihat *button* pada menu *History* berubah warna menjadi lebih gelap dibandingkan *button* menu lain. Didalam menu *history* terdapat tabel data kualitas udara sebelumnya yang tersimpan di *database.* Data yang tampil di menu *History* sebanyak 200 data, diambil dari data yang tebaru. Dalam tabel terdapat keterangan waktu dan semua data kualitas udara yang sebelumnya tampil di menu *real-time monitoring.* Pada tampilan

menu *history* juga terdapat tombol *export* untuk mencetak data kualitas udara dalam bentuk file. Nantinya data kualitas udara yang dicetak dalam bentuk *file excel*. Data kualitas udara yang dicetak bisa mencapai 8000 data pada masing-masing pengukuran kualitas udara.

9) *Design User Interface* (UI) Menu *About*

Gambar 4. 22. *Design User Interface* (UI) Menu *About*

Pada Gambar 4.22 diatas menunjukkan desain UI menu *about*. *Button* menu *about* berubah warna menjadi lebih gelap dibandingkan *button* pada menu lain. Didalam halaman *about* nantinya akan diisi dengan informasi terkait dengan sistem pemantauan kualitas udara. Isi didalamnya meliputi latar belakang, teknologi yang digunakan, manfaat dan lain sebagianya terkait dengan pembuatan sistem pemantauan kualitas udara berbasis IoT dan web.

3. *Implementation*

Implementasi ini melibatkan dua bagian utama. Diantaranya pemrograman NodeMCU dengan Arduino IDE untuk mengambil data dari sensor MQ-135 dan mengontrol lampu LED. Dan juga pengembangan *interface website* menggunakan Laravel 10 dengan *WebSocket* untuk menampilkan data pemantauan kualitas udara secara *real-time* dan menyimpan data ke *database* MySQL. Adapun langkah-langkah implementasi dalam membuat sistem pemantauan kualitas udara berbasis IoT menggunakan NodeMCU dengan *interface website*.

a. Langkah-langkah Pembuatan Sistem

1) Pemrograman Arduino untuk Inisialisasi Perangkat IoT

Berikut adalah kode untuk NodeMCU yang menggunakan sensor MQ-135 dan mengontrol lampu LED. Kode ini mengirimkan data ke server melalui HTTP POST:

```
#include <ESP8266WiFi.h>       // Library untuk
mengontrol modul WiFi ESP8266
#include "MQ135.h"             // Library untuk sensor
MQ135
#include <ESP8266HTTPClient.h> // Library untuk
mengirimkan HTTP request
#include <NTPClient.h> // Library untuk mengakses
waktu melalui protokol NTP.
#include <WiFiUdp.h> // Library untuk komunikasi UDP
yang digunakan oleh NTPClient.

// Mendeklarasikan API key dan informasi jaringan WiFi
String apiKey = "9nZmjxB83vwynwr"; // API key untuk
server ThingSpeak
const char *ssid = "MyHotspot";     // SSID dari
jaringan WiFi
const char *pass = "ahmad45077";    // Password untuk
jaringan WiFi
String server = "http://192.168.137.33:8005"; //
Alamat server untuk mengirim data

WiFiUDP ntpUDP; NTPClient timeClient(ntpUDP,
"pool.ntp.org", 25200, 6000); // NTP client untuk
mendapatkan waktu WiFiClient client;
WiFiClient client;

// Mendefinisikan pin LED
const int greenLED = D2;
const int redLED = D1;

void setup() {
  Serial.begin(115200); // Inisialisasi komunikasi
serial dengan baud rate 115200

  // Mengatur pin LED sebagai output
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);

  // Menghubungkan ke jaringan WiFi
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
```

```
  Serial.println("WiFi connected");
  Serial.print("IP: ");
  Serial.println(WiFi.localIP()); // Menampilkan
alamat IP yang diperoleh
}

void loop() {

timeClient.update(); // Memperbarui waktu dari NTP
server Serial.print("Current time: ");
Serial.println(timeClient.getFormattedTime());

  MQ135 gasSensor = MQ135(A0); // Membuat objek sensor
MQ135 yang terhubung ke pin A0

  // Mendapatkan nilai PPM CO2
  float air_quality = gasSensor.getPPM();

  // Mendapatkan nilai tingkat asap (contoh
implementasi)
  float smokeLevel = gasSensor.getResistance(); // Ini
hanyalah contoh kasar

  // Mendapatkan nilai PPM CO2 yang dikoreksi untuk
suhu dan kelembaban tertentu
  float ppm = gasSensor.getCorrectedPPM(40, 75); //
40°C dan 75% RH sebagai contoh

  float co_ppm = air_quality * 0.4; // Estimasi PPM CO
berdasarkan kualitas udara

  // Mengatur status LED berdasarkan kualitas udara
  if (air_quality < 100) { // Threshold untuk kualitas
udara yang baik
    digitalWrite(greenLED, HIGH);
    digitalWrite(redLED, LOW);
  } else {
    digitalWrite(greenLED, LOW);
    digitalWrite(redLED, HIGH);
  }

  // Menampilkan hasil di Serial Monitor
  Serial.print("Air Quality: ");
  Serial.print(air_quality);
  Serial.println("  PPM");
  Serial.print("Smoke: ");
  Serial.print(smokeLevel);
  Serial.println("  (Nilai Resistansi)"); // Ini hanya
kasar, Anda memerlukan kalibrasi lebih lanjut
  Serial.print("CO2: ");
  Serial.print(ppm);
  Serial.println("  PPM");
  Serial.print("CO: ");
  Serial.print(co_ppm);
  Serial.println("  PPM (Estimasi)");
  Serial.println();
```

```
  if (WiFi.status() == WL_CONNECTED) { // Memastikan
WiFi terhubung
    HTTPClient http;
    http.begin(server + "/api/post_data"); // Memulai
HTTP request ke server

    // Menyusun payload JSON
    String payload = "{";
    payload += "\"air_quality\":" +
String(air_quality) + ",";
    payload += "\"smoke_level\":" + String(smokeLevel)
+ ",";
    payload += "\"co2\":" + String(ppm) + ",";
    payload += "\"carbon_monoksida\":" +
String(co_ppm);
    payload += "}";

    // Menambahkan header
    http.addHeader("Content-Type",
"application/json");
    http.addHeader("API-KEY", apiKey);

    // Mengirim request POST
    int httpResponseCode = http.POST(payload);
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
      Serial.print("Response: ");
      Serial.println(response);
    } else {
      Serial.print("Error on sending POST: ");
      Serial.println(httpResponseCode);
    }

    // Mengakhiri koneksi
    http.end();
  } else {
    Serial.println("Wifi Disconnected!"); //
Menampilkan pesan jika WiFi terputus
  }
  delay(1000); // Menunggu 1 detik sebelum iterasi
loop berikutnya
}
```

Penjelasan Fungsi-fungsi Utama:

- *Setup*(): Fungsi yang dijalankan sekali saat board dinyalakan atau di-reset. Menginisialisasi komunikasi serial, mengatur pin mode LED, dan menghubungkan ke jaringan WiFi.
- Loop(): Fungsi yang dijalankan berulang kali setelah *setup*(). Membaca data dari sensor MQ135, mengatur LED berdasarkan

kualitas udara, menampilkan data di Serial Monitor, dan mengirimkan data ke *server* melalui *HTTP POST request*.

Penjelasan detail lainnya:

- *Library Inclusions*: Menggunakan *library* ESP8266WiFi untuk konektivitas WiFi, MQ135 untuk membaca data dari sensor, dan *ESP8266HTTPClient* untuk mengirimkan *HTTP request*.
- *WiFi Connection*: Menggunakan *WiFi.begin*(*ssid, pass*) untuk memulai koneksi ke jaringan WiFi dan *WiFi.status*() untuk mengecek status koneksi.
- *HTTP Request*: Menggunakan *HTTPClient* untuk membuat dan mengirimkan *HTTP POST request* ke *server* dengan data JSON.

2) *Backend Webserver* dengan Laravel 10

Langkah-langkah dalam pengembangan *interface website* adalah sebagai berikut:

1. Install Laravel 10 dan Dependensi:

```
composer create-project --prefer-dist laravel/laravel
aq "10*"
cd aq
composer require pusher/pusher-php-server
composer require beyondcode/laravel-websockets
```

Kode `composer create-project --prefer-dist laravel/laravel aq "10*"` berarti membuat project baru dengan dengan nama fil aq menggunakan Laravel versi 10. Perintah `composer require` digunakan di dalam lingkungan Composer untuk menambahkan paket baru ke dalam proyek PHP. Diikuti oleh nama paket yang ingin Anda instal. Biasanya terdiri dari nama vendor (seperti `pusher` atau `beyondcode`) diikuti oleh nama paket itu sendiri (`pusher-php-server` atau `laravel-websockets`). Perintah `pusher-php-server` menyediakan klien PHP untuk menggunakan layanan *Pusher*, yang merupakan layanan *real-time messaging* untuk aplikasi web dan *mobile*. Dengan menjalankan perintah ini, Composer akan mengunduh dan menginstal paket ini beserta dependensinya ke dalam proyek PHP yang sedang dikerjakan. Sedangkan untuk

perintah `laravel-websockets` adalah implementasi server *WebSocket* untuk aplikasi Laravel. WebSocket adalah protokol komunikasi dua arah yang memungkinkan klien dan *server* untuk saling berkomunikasi dengan cara yang efisien dan *real-time*. Laravel *WebSockets* adalah solusi yang kuat untuk membangun fitur *real-time* dalam aplikasi Laravel, seperti *chat* atau *update real-time*.

2. Konfigurasi *WebSockets*:

   - Publikasikan konfigurasi *WebSockets* dengan perintah:

     ```
     php artisan vendor:publish --
     provider="BeyondCode\LaravelWebSockets\WebSocketsS
     erviceProvider" --tag="config"
     ```

     Perintah `php artisan vendor:publish` dengan konfigurasi yang diberikan bertujuan untuk memungkinkan mengubah atau menyesuaikan pengaturan dari paket `laravel-websockets` yang telah di*install* sebelumnya. Dengan mempublikasikan *file* konfigurasi dan dapat mengedit nilai-nilai konfigurasi. Dengan mempublikasi *file* konfigurasi juga dapat mengaktifkan atau menonaktifkan fitur-fitur tertentu. Dapat juga menyesuaikan perilaku paket sesuai kebutuhan aplikasi.

   - Edit *file* konfigurasi `config/websockets.php` dan `config/broadcasting.php`:

     ```
     'pusher' => [
         'driver' => 'pusher',
         'key' => env('PUSHER_APP_KEY'),
         'secret' => env('PUSHER_APP_SECRET'),
         'app_id' => env('PUSHER_APP_ID'),
         'options' => [
             'cluster' => env('PUSHER_APP_CLUSTER'),
             'useTLS' => true,
             'encrypted' => true,
             'host' => '127.0.0.1',
             'port' => 6001,
             'scheme' => 'http',
         ],
     ],
     ```

     Konfigurasi ini memungkinkan aplikasi Laravel Anda untuk terhubung dan berkomunikasi dengan layanan *Pusher*

menggunakan kunci, rahasia, ID aplikasi, dan opsi-opsi yang telah diatur di *file* `.env`. Hal ini memfasilitasi pengiriman pesan *real-time* dan integrasi dengan fitur-fitur lain dalam aplikasi web.

- Tambahkan ke `.env`:

```
PUSHER_APP_ID=12345678
PUSHER_APP_KEY=qwerty
PUSHER_APP_SECRET=qaz
PUSHER_HOST=127.0.0.1
PUSHER_PORT=6002
PUSHER_SCHEME=http
PUSHER_APP_CLUSTER=mt1
```

Variabel lingkungan seperti *PUSHER_APP_ID*, *PUSHER_APP_KEY*, *PUSHER_APP_SECRET*, dan *PUSHER_APP_CLUSTER* digunakan untuk mengatur integrasi aplikasi dengan layanan *Pusher*. *PUSHER_APP_ID* adalah ID unik yang mengidentifikasi aplikasi di *Pusher* API. *PUSHER_APP_KEY* adalah kunci API yang diperlukan untuk otentikasi dan mengizinkan akses ke layanan Pusher. *PUSHER_APP_SECRET* adalah rahasia yang digunakan untuk menghasilkan token otentikasi dan melindungi komunikasi antara aplikasi dan *Pusher*. Sedangkan *PUSHER_APP_CLUSTER* menentukan *cluster server Pusher* yang digunakan untuk koneksi, yang tersebar di berbagai lokasi untuk meningkatkan kinerja dan reliabilitas aplikasi.

Perlu mengganti setiap nilai seperti *your-app-id*, *your-app-key*, *your-app-secret*, dan *mt1* dengan nilai yang diberikan oleh *Pusher* saat awal mendaftar aplikasi di *dashboard* mereka. Dengan menggunakan variabel lingkungan ini dalam aplikasi Laravel (atau *framework* lain yang mendukung *file .env*), dapat mengatur konfigurasi aplikasi dengan aman dan mudah disesuaikan antara lingkungan pengembangan dan produksi.

3. *Migrate Database*

Migrasi *database* adalah proses pemindahan data dari satu sistem basis data ke sistem lain atau dari satu versi ke versi lainnya. Proses ini dilakukan untuk meningkatkan kinerja, memperbaiki keamanan, menambahkan fitur baru, memastikan kompatibilitas, mengurangi biaya, dan mempermudah pemeliharaan serta dukungan. Dalam pengembangan aplikasi dengan Laravel, migrasi membantu mengelola skema basis data secara otomatis dengan skrip migrasi, memungkinkan perubahan skema dicatat dan diterapkan dengan konsisten di berbagai lingkungan. Berikut langkah-langkah dalam migrasi *database*:

- Buat migrasi contoh untuk tabel aq:

```
php artisan make:migration create_air_quality_table
--create=aq
```

- Edit file migrasi di *database/migrations* contohnya pada tabel data aq:

```
public function up(): void
    {
        Schema::create('aq', function (Blueprint
$table) {
            $table->id();
            $table->decimal("value",10,2)-
>nullable();
            $table->timestamps();
        });
    }
```

- Jalankan migrasi dengan perintah:

```
php artisan migrate
```

4. Buat model dan controller pada setiap data kualitas udara.
   - Contoh perintah untuk membuat pada data aq:

```
php artisan make:model aq
```

   - Contoh perintah controller pada sensor:

```
php artisan make:controller SensorController
```

5. Tambahkan *Route* dan *Method*

- Tambahkan route untuk menerima data dari NodeMCU di `routes/api.php`:

```
Route::group(['middleware' => [
'api_middleware']], function () {
    Route::post('/post_data',
[App\Http\Controllers\Api\SensorController::class,
'post']);
});
```

- Membuat kode php pada SensorController agar dapat menerima dan menyimpan data.

```
<?php
namespace App\Http\Controllers\Api;
use App\Http\Controllers\Controller;
use App\Models\Setting;
use Illuminate\Http\Request;
class SensorController extends Controller
{
function   responseSuccess($data,$status=true,$code
=200)
    {
        $response = [
            'status'  =>  $status,
            'response_code'  =>  $code,
            'data'  =>  $data,
        ];
        return response()->json($response,200);
    }
    public function post(Request $request)
    {
        $data  =  json_decode(json_encode($request-
>all()));
        $aq = Setting::where('key','aq')->first();
        $aq->value = $data->air_quality;
        $aq->save();
        $co = Setting::where('key','co')->first();
        $co->value = $data->co2;
```

```
        $co->save();
        $smoke   =   Setting::where('key','smoke')-
>first();
        $smoke->value = $data->smoke_level;
        $smoke->save();
        $carbon  =  Setting::where('key','carbon')-
>first();
        $carbon->value = $data->carbon_monoksida;
        $carbon->save();
        return  $this->responseSuccess("Post  data
sukses!");

    }
}
```

Pada kode php diatas dapat dijelaskan bahwa kode pada SensorController dapat menerima data dari sensor melalui permintaan POST. Kode diatas juga dapat memperbarui data dalam basis data dan mengirimkan respons yang menunjukkan bahwa operasi berhasil. Setiap nilai sensor disimpan dalam entri terpisah di tabel *settings*.

6. Membuat *event* untuk *broadcast*:
   - Membuat *event* dengan perintah:

     ```
     php artisan make:event BroadcastChartEvent
     php artisan make:event BroadcastDataEvent
     ```

     Pada perintah diatas dapat dijelaskan bahwa nantinya terdapat dua *event* yaitu *charts event* untuk membuat grafik naik turun nilai kualitas udara. Kemudian untuk data *event* nantinya digunakan menampilkan nilai dari kulaitas udara beserta keterangan sehat dan tidak sehat.
   - Edit masing-masing *event* sesuai dengan yang diinginkan. Contohnya mengedit BroadcastChartevent untuk menampilkan grafik pada tiap nilai kualitas udara. Berikut adalah kode php nya:

```
<?php
namespace App\Events;
use App\Models\Aq;
use App\Models\Co;
use App\Models\Setting;
use App\Models\Smoke;
use Carbon\Carbon;
use Illuminate\Broadcasting\Channel;
use Illuminate\Broadcasting\InteractsWithSockets;
use Illuminate\Broadcasting\PresenceChannel;
use Illuminate\Broadcasting\PrivateChannel;
use Illuminate\Contracts\Broadcasting\ShouldBroadcast;
use Illuminate\Foundation\Events\Dispatchable;
use Illuminate\Queue\SerializesModels;
use App\Models\Carbon as CarbonMonoksida;
class BroadcastChartEvent implements ShouldBroadcast
{
    use Dispatchable, InteractsWithSockets, SerializesModels;
    public function __construct()
    {
    }
    public function broadcastOn(): array
    {
        return [
            new Channel('data-chart'),
        ];
    }
    public function broadcastWith()
    {
        $createdAtDates_aq = Aq::orderBy('id','DESC')-
>limit(20)->pluck('created_at');
```

```
        $formattedDates_aq = $createdAtDates_aq->map(function
($date) {
            return Carbon::parse($date)->format('d-m-Y H:i:s');
        });
        $createdAtDates_co = Co::orderBy('id','DESC')-
>limit(20)->pluck('created_at');
        $formattedDates_co = $createdAtDates_co->map(function
($date) {
            return Carbon::parse($date)->format('d-m-Y H:i:s');
        });
        $createdAtDates_smoke = Smoke::orderBy('id','DESC')-
>limit(20)->pluck('created_at');
        $formattedDates_smoke = $createdAtDates_smoke-
>map(function ($date) {
            return Carbon::parse($date)->format('d-m-Y H:i:s');
        });
        $createdAtDates_carbon =
CarbonMonoksida::orderBy('id','DESC')->limit(20)-
>pluck('created_at');
        $formattedDates_carbon = $createdAtDates_carbon-
>map(function ($date) {
            return Carbon::parse($date)->format('d-m-Y H:i:s');
        });
        $data['aq']['data'] = Aq::orderBy('id','DESC')->limit(20)-
>get()->pluck('value');
        $data['aq']['tanggal'] = $formattedDates_aq;

        $data['co']['data'] = Co::orderBy('id','DESC')->limit(20)-
>get()->pluck('value');
        $data['co']['tanggal'] = $formattedDates_co;
```

```
        $data['smoke']['data'] = Smoke::orderBy('id','DESC')-
    >limit(20)->get()->pluck('value');
        $data['smoke']['tanggal'] = $formattedDates_smoke;
        $data['carbon']['data'] =
    CarbonMonoksida::orderBy('id','DESC')->limit(20)->get()-
    >pluck('value');
        $data['carbon']['tanggal'] = $formattedDates_carbon;

        return $data;
      }
    }
```

Kode diatas mendefinisikan sebuah *event* yang mengumpulkan data dari berbagai sensor seperti aq, co2, smoke dan co. Kemudian memformat tanggalnya dalam format `d-m-Y H:i:s` menggunakan `Carbon::parse`. Pada *event* ini juga dapat menyiarkannya secara *real-time* melalui saluran *data-chart*. *Event* ini penting dikarenakan *event* ini memungkinkan aplikasi untuk mengirimkan data sensor secara *real-time* ke *frontend*. Sehingga data dapat ditampilkan dalam bentuk *chart* atau grafik secara langsung tanpa perlu menyegarkan halaman.

3) Menampilkan Data di Web *Interface*

1. Tambahkan route untuk menampilkan data di routes/web.php untuk menampilkan data dengan kode sebagai berikut:

```
Route::get('/', function () {
   return redirect('/login');
});
Auth::routes();
Route::group(['middleware' => [ 'auth']], function () {
   Route::get('/dashboard',
[App\Http\Controllers\DashboardController::class,          'index'])-
>name('dashboard.index');
```

```
Route::get('/history',
[App\Http\Controllers\HistoryController::class, 'index'])-
>name('history.index');
Route::get('/history/export',
[App\Http\Controllers\HistoryController::class, 'export'])-
>name('history.export');
});
```

Pada kode php diatas dapat dijelaskan kode ini mendefinisikan beberapa rute yang dilindungi oleh middleware `auth`. Ini berarti pengguna harus login terlebih dahulu untuk mengakses rute-rute tersebut. Rute-rute yang didefinisikan adalah untuk mengakses *dashboard, history,* dan *export history*. Kode php ini penting dikarenakan dapat melindungi rute dengan *middleware* `auth` memastikan bahwa hanya pengguna yang terautentikasi yang bisa mengakses bagian-bagian tertentu dari sistem web. Sehingga meningkatkan keamanan. Selain itu, memberikan nama pada rute memudahkan dalam pengelolaan dan referensi rute di berbagai bagian sistem web.

2. Kemudian setelah itu baru membuat *views* pada beberapa menu yang diinginkan seperti *view* pada menu *login*, *dashboard* dan *history*. Pada setiap nama *file* menu diberi nama `blade.php` sebagai pembeda, contohnya `dashboard.blade.php` dan begitupun *file view* yang lain.

4) Menjalankan Sistem

1. Inisialisasi perangkat IoT menggunakan Arduino yang sudah diprogram.
2. Nyalakan Xampp, dikarenakan masih menggunakan *server* local.
3. Buka *powershell* atau *command protompt* dan buka folder aq kemudian jalankan *server* Laravel dengan perintah:
   `php artisan serve --host=0.0.0.0 --port=8005`

Perintah digunakan untuk menjalankan *server* lokal untuk aplikasi Laravel. Dengan demikian dapat mengakses aplikasi melalui browser di alamat http://localhost:8005 atau dari perangkat lain di jaringan yang sama dengan menggunakan alamat IP komputer dalam jaringan tersebut. Contohnya http://127.0.0.1:8005/.

4. Jalankan *websocket server* dengan perintah:

   ` php artisan websockets:serve --port=6002`

   Perintah diatas digunakan untuk menjalankan *server WebSocket* untuk aplikasi Laravel pada *port* 6002. Ini memungkinkan aplikasi untuk mendukung komunikasi real-time antara server dan klien melalui *WebSocket,* menggunakan *port* yang ditentukan. Anda dapat mengakses fitur-fitur *real-time* dari aplikasi Laravel melalui *port* 6002 setelah menjalankan perintah ini.

5. Jalankan perintah sensor update:

   `php artisan app:sensor-update`

   Perintah diatasa digunakan untuk menjalankan perintah artisan khusus yang ditentukan dalam aplikasi Laravel untuk memperbarui data sensor. Perintah ini bisa digunakan untuk mengambil data terbaru dari sensor yang terhubung dan memprosesnya. Kemudian menyimpan atau memperbarui informasi tersebut di dalam basis data aplikasi. Ini adalah cara terstruktur untuk mengotomatisasi dan menjalankan tugas-tugas pemeliharaan atau pembaruan data sensor dalam aplikasi Laravel.

6. Menjalankan broadcast data:

   `php artisan app:boardcast-data`

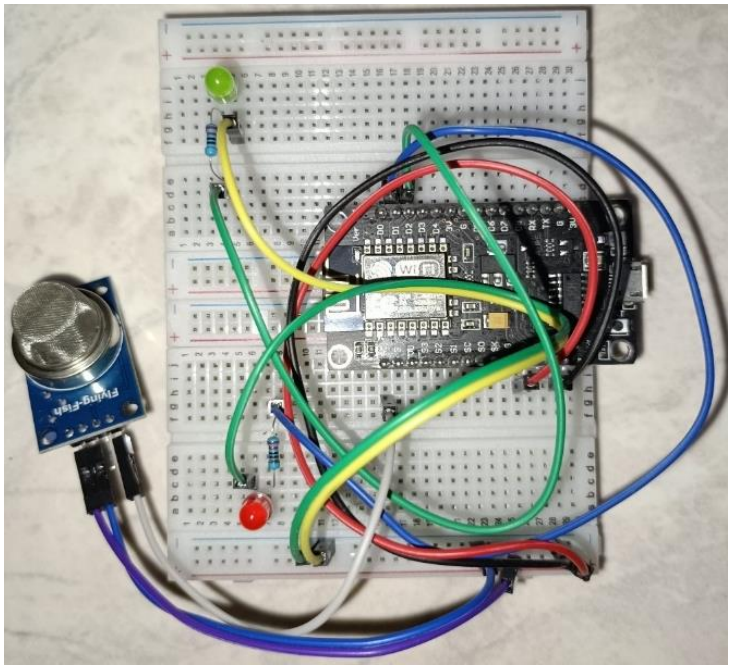
   Perintah ini digunakan untuk mengirimkan data tertentu, mungkin dari sensor atau sumber lain ke berbagai klien secara *real-time* melalui mekanisme *broadcasting* Laravel. Ini memungkinkan aplikasi untuk memberikan informasi terbaru secara langsung

kepada pengguna yang terhubung, memastikan mereka selalu menerima data terkini tanpa perlu menyegarkan halaman.

7. Buka browser dan akses halaman http://127.0.0.1:8005/ untuk melihat *interface website* dan memantau kualitas udara.

Dengan langkah-langkah di atas, Anda telah membangun sistem pemantauan kualitas udara berbasis IoT dan web yang menggunakan NodeMCU dan sensor MQ-135 untuk mengirim data ke *server* Laravel. Kemudian menampilkan data tersebut di antarmuka web dengan *update real-time* menggunakan *WebSockets* dan *broadcast* data.

b. Hasil Perancangan Alat *Internet of Things* (IoT)



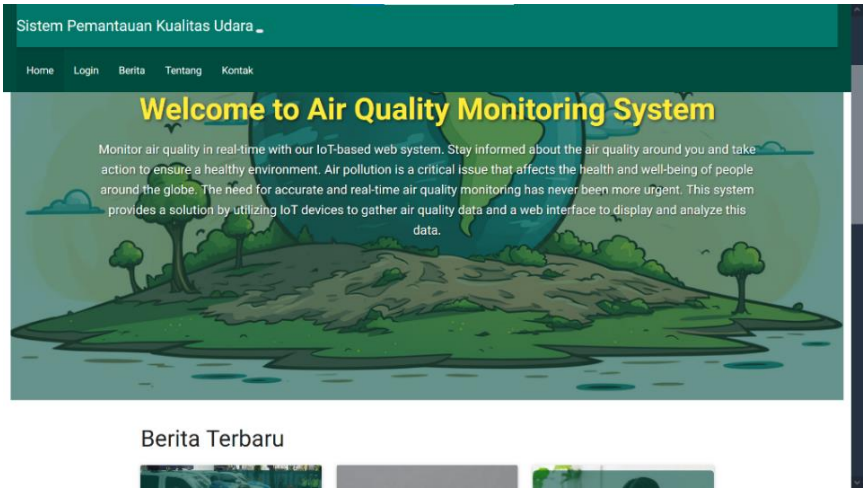
Gambar 4. 23. Hasil Perancangan Alat *Internet of Things* (IoT)

Pada Gambar 4.23 diatas, menunjukkan rangkaian alat sistem pemantauan kualitas udara. Terdapat instalasi kabel untuk menghubungkan satu komponen dengan komponen lainnya. Sensor MQ-135 dihubungkan dengan NodeMCU ESP8266 menggunakan pin A0 (analog *input* 0). NodeMCU ESP8266 adalah mikrokontroler berbasis WiFi yang berfungsi untuk mengirimkan data. Sensor MQ-135 juga biasanya memerlukan pemanasan sebelum penggunaan,

jadi pastikan untuk memberikan waktu pemanasan yang cukup sebelum membaca data. Menghubungkan kedua LED (merah dan hijau) ke pin digital NodeMCU (misalnya D1 dan D2). Lampu LED merah berfungsi sebagai indikator jika kondisi udara buruk. Sedangkan lampu LED hijau berfungsi sebagai indikator jika udara baik. Resistor 220 Ω(*ohm*) ke masing-masing kaki anoda LED. Kaki katoda LED dihubungkan ke *ground* NodeMCU. Resistor berfungsi untuk melindungi LED dari arus yang berlebihan.

c. Hasil Pembuatan *Website* Sistem Pemantauan Kualitas Udara

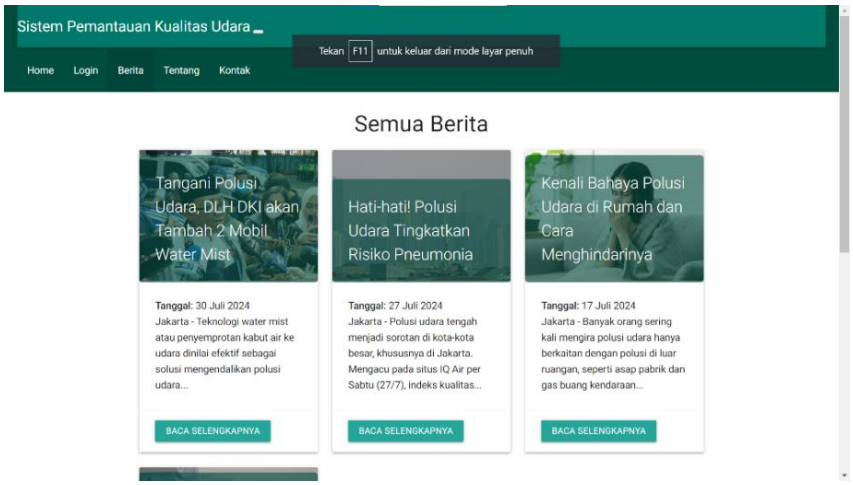
1. Tampilan *Landing Page* atau *Home*



Gambar 4. 24. Tampilan *Landing Page*

Gambar 4.24 diatas, menunjukkan tampilan pada *landing page* ataua halaman *home*. Pada *landing page* terdapat *header* yang di fix. Pada *header* terdapat juga *button home*, *login*, berita, tentang dan kontak. Pada tampilan awal ini juga terdapat *welcome text*, berita terbaru, tentang sistem, dan kontak. Pada *website* sistem pemantauan kualitas udara menggunakan tema berwarna hijau yang selaras.

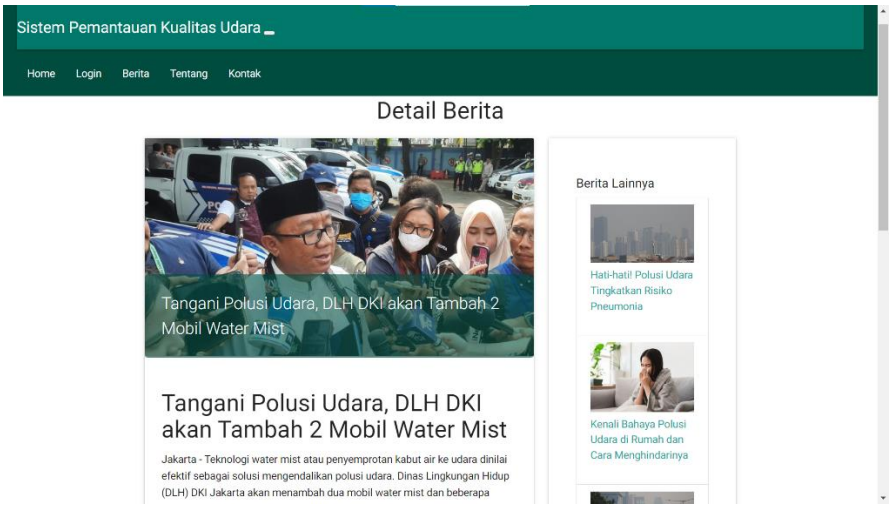
2. Tampilan Menu Berita



Gambar 4. 25. Tampilan Menu Berita

Gambar 4.25 diatas, menunjukkan tampilan pada menu berita. Tampilan ini bertujuan untuk menyajikan berita terkait dengan kualitas udara. Pada menu berita terdapat *header* yang difix (tidak hilang jika dilakukan pengguliran) dan terdapat tampilan berita yang ditampilan 3 sejajar. Disetiap berita berdapat keterangan tanggal berita dan ringkasan berita terkait dengan berita tersebut. Terdapat juga *button* baca selengkapnya untuk melihat detail dari berita.

3. Tampilan Detail Berita

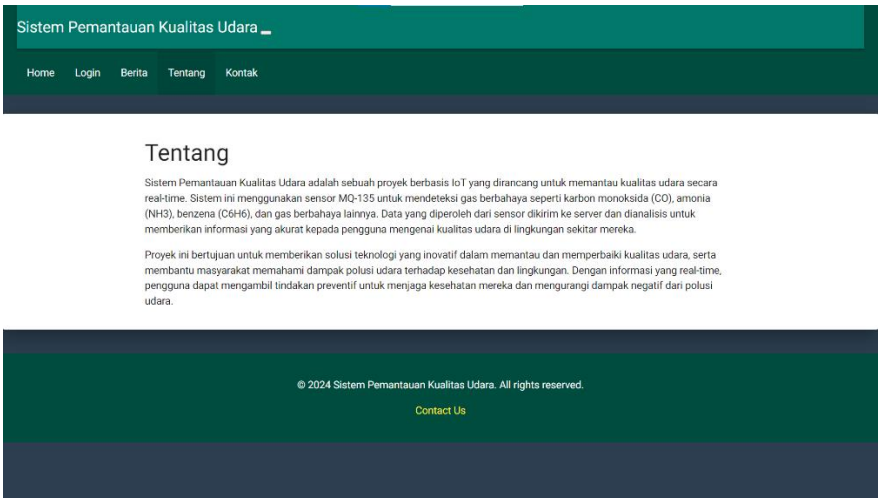


Gambar 4. 26. Tampilan Detail Berita

Gambar 4.26 diatas, menunjukkan tampilan pada detail berita. Tampilan ini ada ada jika *user* menekan baca selengkapnya, maka

akan muncul halaman detail berita. Pada halaman detail berita terdapat informasi lengkap tentang berita. Disamping informasi berita terdapat menu berita selanjutnya yang memudahkan *user* untuk melihat berita lain dengan cepat tanpa harus kembali ke halaman semua berita.

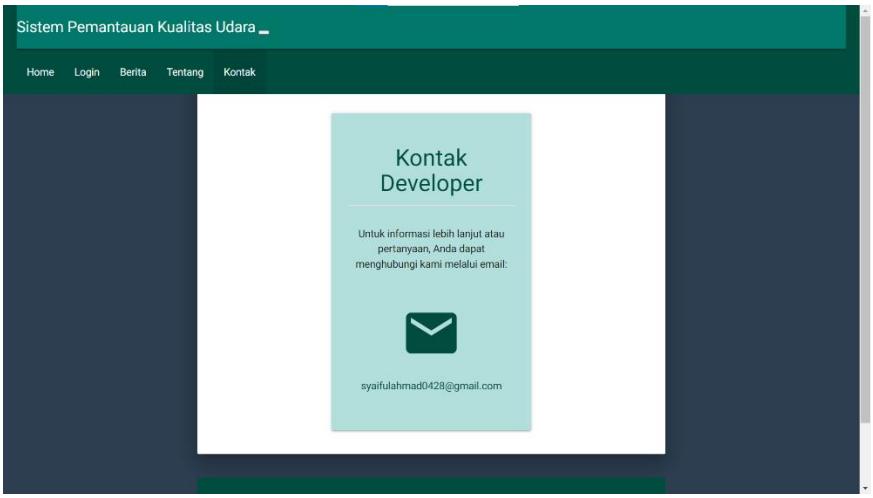
4. Tampilan Menu Tentang



Gambar 4. 27. Tampilan Menu Tentang

Gambar 4.27 diatas, menunjukkan tampilan pada menu tentang. Pada tampilan menu tentang terdapat *header* yang sama seperti pada halaman awal. Pada menu tentang berisi informasi singkat terkait dengan sistem pemantauan kualitas udara yang dibuat. Terdapat juga tampilan *footer* yang berada dibawah.

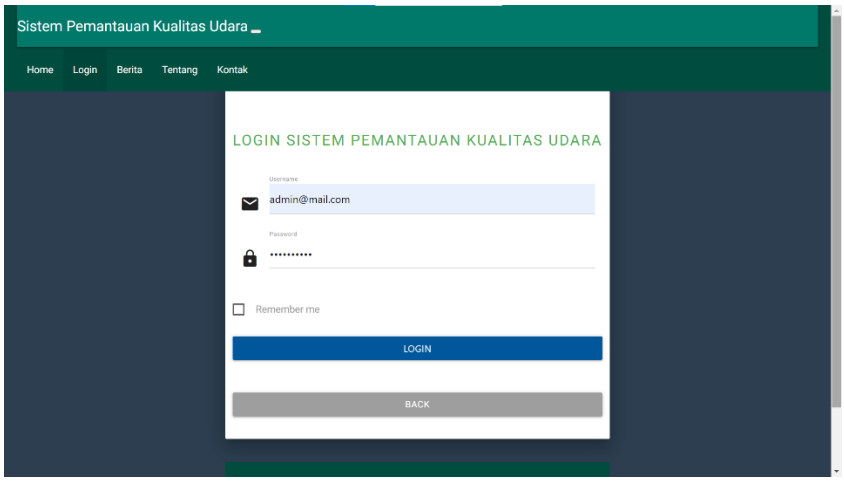
5. Tampilan Menu Kontak



Gambar 4. 28. Tampilan Menu Kontak

Gambar 4.28 diatas, menunjukkan tampilan pada menu kontak. Pada menu kontak terdapat *header* sama seperti yang ada pada tampilan awal yang bertujuan untuk memudahkan *user* berpindah-pindah menu lain. Pada halaman ini juga terdapat informasi alamat email *developer* jika terdapat kendala dan masukan terkait dengan sistem pemantauan kualitas udara. Pada alamat email jika ditekan akan langsung menuju halaman google mail.

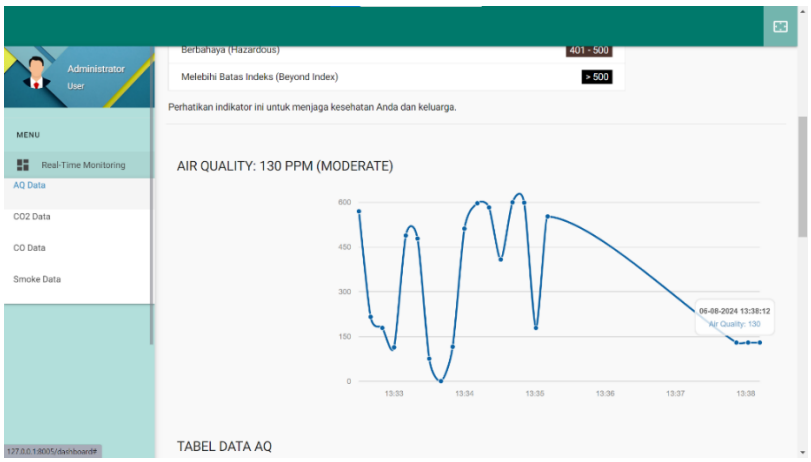
6. Tampilan Menu *Login*



Gambar 4. 29. Tampilan Menu *Login*

Gambar 4.29 diatas menunjukkan tampilan yang ada pada menu *login*. *User* harus mengisi *username* dan juga *password* untuk dapat masuk ke dalam halaman *real-time monitoring*.

7. Tampilan Menu *Real-Time Monitoring*

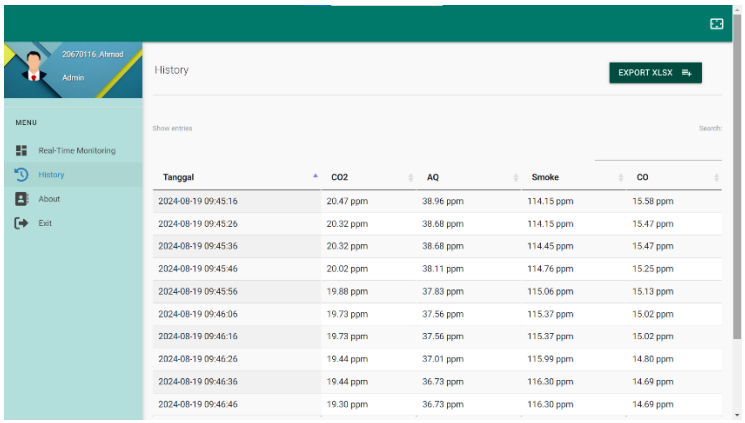


Gambar 4. 30. Tampilan Menu *Real-Time Monitoring*

Pada Gambar 4.30 diatas, menunjukkan tampilan pada menu *real-time monitoring*. Pada *button real-time* monitoring memiliki tampilan *drop-down* untuk menampilkan beberapa menu diantaranya ada menu AQ Data (data kualitas udara), CO^2 Data (data kualitas karbon dioksida), CO Data (data kualitas karbon monoksida), dan *Smoke* Data (data kualitas semua asap). Pada setiap menu kualitas udara memiliki tampilan yang sama kualitas udara satu dengan yang lain. Pada tampilan kualitas udara terdapat informasi terkait dengan rentang nilai kualitas udara dan keterangan. Dibawah informasi rentang nilai kualitas udara terdapat nilai kualitas udara dan juga keterangan kualitas udara tersebut. Kemudian terdapat juga grafik naik turun kualitas udara yang pada setiap titik terdapat keterangan tanggal, waktu dan nilai dari kualitas udara hasil pengukuran dari perangkat IoT. Pada nilai kualitas udara dan grafik yang tampil di *website* bisa me-*refresh* secara otomatis tanpa harus memperbarui halaman. Dibawah grafik kualitas udara terdapat juga tabel dari kualitas udara tersebut yang menampilkan

informasi terkait nilai kualitas udara, waktu pengukuran dan keterangan pada setiap kualitas udara yang tampil. Grafik dan tabel kualitas udara menampilkan maksimal 20 data kualitas udara.

8. Tampilan Menu *History*



Gambar 4. 31. Tampilan Menu *History*

Pada Gambar 4.31 diatas, menunjukan tampilan *website* pemantauan kualitas udara pada menu *history*. Pada menu *history* menampilkan tabel 4 data kualitas udara dari *air quality* (AQ), karbon dioksida (CO^2), karbon monoksida (CO) dan *smoke* (asap.) Tabel yang tampil menu *history* ada 200 data pada masing-masing kualitas udara. Pada menu ini juga terdapat *button* untuk melakukan aksi *export* data dalam bentuk excel. Data yang dapat di *export* bisa mencapai maksimal 8000 data pada masing-masing kualitas udara.

9. Tampilan Menu *About*

Gambar 4. 32. Tampilan Menu *About*

Pada Gambar 4.32 diatas, menunjukkan tampilan *website* pemantauan kualitas udara pada menu *about*. Pada menu *about* menampilkan informasi tentang sistem yang dibuat. Informasi tersebut berupa informasi latar belakang pembuatan sistem, teknologi yang digunakan, fitur, dan lain sebagainya.

4. *Testing*

Terdapat 3 (tiga) *testing* atau pengujian dalam penelitian ini, diantaranya *white box testing*, *black box testing* dan *user acceptance test* (UAT). *White box testing*, *black box testing*, dan UAT memiliki tujuan yang berbeda tetapi saling melengkapi untuk memastikan kualitas, keandalan dan kesiapan sistem untuk diimplementasikan. Pengujian *white box* fokus pada validasi internal kode dan logika. Pengujian *black box* fokus pada validasi fungsionalitas dan kinerja dari perspektif pengguna. Sementara UAT memastikan bahwa sistem memenuhi kebutuhan bisnis dan siap digunakan oleh pengguna akhir.

a. *White Box Testing*

White box testing adalah teknik pengujian perangkat lunak yang melibatkan pemeriksaan kode sumber dari program untuk mendeteksi adanya kesalahan. Tujuan dari pengujian ini adalah memastikan setiap bagian kode berfungsi sesuai dengan spesifikasi yang telah ditetapkan. Hasil pengujian *white box* dapat dilihat pada Tabel 4.1.

Tabel 4. 1. *White Box Testing*

Node	Code	Keterangan
1	public function post(Request $request) { $data = json_decode(json_encode($request->all())); $aq = Setting::where('key','aq')->first();$aq->value = $data->air_quality; $aq->save(); $co = Setting::where('key','co')->first();$co->value = $data->co2; $co->save();	Penyimpanan data ke server dan respons untuk serial monitor pada Arduino.

	$smoke = Setting::where('key','smoke')->first();$smoke->value = $data->smoke_level; $smoke->save(); $carbon = Setting::where('key','carbon')-first();$carbon->value = $data->carbon_monoksida; $carbon->save(); return $this->responseSuccess("Post data sukses!"); }	
2	class LandingPageController extends Controller{ public function index() { return view('landing'); }}	Controller untuk menampilkan landingpage
3	protected $redirectTo = '/dashboard'; public function __construct(){$this->middleware('guest')-except('logout'); $this->middleware('auth')->only('logout'); }	Login ini memastikan bahwa middleware guest dan auth.
4	public function index() { $createdAtDates_aq =Aq::orderBy('id','DESC')->limit(20)->pluck('created_at'); $formattedDates_aq = $createdAtDates_aq->map(function ($date) { return Carbon::parse($date)->format('d-m-Y H:i:s'); }); $data['aq']['data'] = Aq::orderBy('id','DESC')->limit(20)->get()->pluck('value'); $data['aq']['tanggal'] = $formattedDates_aq;}	Controller untuk mengambil data.
5	Auth::routes(); Route::group(['middleware' => ['auth']], function () {	Route untuk mengkases data halaman dashboard/aq

	Route::get('/dashboard', [App\Http\Controllers\DashboardController:: class, 'index'])->name('dashboard.index'); });	data. Harus login.
6	'password_timeout' => 10800,	Password time out pada auth
7	@foreach($aq['data'] as $index => $value) init_data_aq.push({"date":"{{$aq['tanggal'][$ index]}}", "ppm": {{$aq['data'][$index]}} }); @endforeach	View untuk menampilkan data kualitas udara aq.

1) Basis *Path Test*

Basis *path test* adalah teknik yang digunakan untuk memastikan bahwa semua jalur eksekusi yang mungkin dalam kode telah diuji setidaknya sekali. Berikut adalah langkah-langkah dalam basis *path test* untuk sistem pemantauan kualitas udara.

a) Identifikasi Jalur Dasar

Identifikasi semua jalur eksekusi dalam kode yang relevan dengan sistem penampilan kualitas udara. Setiap node yang telah kita tetapkan sebelumnya akan menjadi bagian dari jalur eksekusi.

b) Menghitung *Complexity Cyclometic*

Complexity cyclometic atau kompleksitas siklomatik adalah metrik yang digunakan untuk mengukur jumlah jalur linier independen dalam program. Berikut rumus untuk menghitung kompleksitas siklomatik (V(G)).

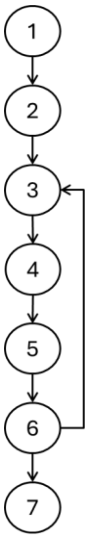
(V(G)) = E – N + 2

Keterangan:

E : jumlah edge (garis alur dalam *flow graph*)

N : jumlah node (titik keputusan dalam *flow graph*)

2) *Flow Graph*



Gambar 4. 33 *Flow Graph Basis Path.*

Gambar 4.33 menunjukkan *flow graph basis path* sistem pemantauan kualitas udara. Dari representasi *flow graph* diatas, maka didapatkan *node* dan *edge* sebagai berikut.

Nodes: 7

Edges: 7

Perhitungan menggunakan rumus *complexity cyclometic*:

(V(G)) = E – N + 2 = 7 – 7 + 2 = 2

Ini menunjukkan bahwa ada 2 *path* jalur independen dalam sistem pemantauan kualitas udara.

Jalur Independen:

Path 1 : 1, 2, 3, 5, 6, 7

Path 2 : 1, 2, 4, 2, 3, 5, 6, 3, 4, 5, 6, 7

Keterangan:

Path 1 : Penampilan data kualitas udara.

Path 2 : Penampilan data kualitas udara jika waktu berakhir.

b. *Black Box Testing*

Black box testing atau pengujian *black box* digunakan untuk memastikan aplikasi berjalan sesuai dengan harapan pengguna.

Pengujian *black box* ini penulis melakukan pengujian sistem kepada 3 (tiga) dosen program studi Informatika Universitas PGRI Semarang. Hasil *black box testing* dilihat pada Tabel 4.2.

Tabel 4. 2. Hasil *Black Box Testing*

Nama Pengujian	**Hasil yang Diharapkan**	**Hasil yang Didapatkan**	**Keterangan**					
			Diterima			**Ditolak**		
			1	**2**	**3**	**1**	**2**	**3**
Program di Arduino	*Admin* inisialisasi perangkat IoT	Data kualitas udara dapat dikirimkan ke *server website* secara *real-time*.	v	v	v			
	Melihat lampu led pada rangkaian IoT.	Lampu LED nyala sesuai dengan kualitas udara. Jika nilai kualitas udara kurang dari 100 ppm maka lampu LED nyala warna hijau. Jika nilai kualitas udara melebihi 100 ppm maka	v	v	v			

		lampu LED nyala warna merah.						
Halaman *LandingPage* / Tampilan Awal Saat Menjalankan *Server*	*User* melihat tampilan *landingpage* yang terdapat beberapa tampilan seperti berita, kontak dan tentang sistem aplikasi.	Sistem menampilkan beberapa button (*login,* berita, tentang dan kontak), berita tentang sistem dan kontak *developer*.	v	v	v			
	User dapat melakukan *login.*	Sistem menampilkan halaman *login* untuk masuk ke sistem pemantauan.	v	v	v			
	User menekan *button* berita.	Sistem menampilkan semua berita.	v	v	v			
	User menekan link	Sistem menampilkan detail berita.	v	v	v			

	baca selengkapnya							
	User menekan email pada menu kontak.	Sistem menampilkan alamat email *developer.*	v	v	v			
	User menekan *button* Tentang.	Sistem menampilkan gambaran singkat mengenai sistem yang dibuat.	v	v	v			
Halaman Setelah *Login*	*User* menekan menu *Real-Time Monitoring.*	Sistem menampilkan *drop down* beberapa menu data kualitas udara seperti *air quality,* CO, CO^2, dan *smoke.*	v	v	v			

	User menekan data kualitas udara.	Sistem menampilkan grafik dan tabel data kualitas udara yang dikirimkan dari perangkat IoT.	v	v	v			
	User melihat grafik dan nilai kualitas udara.	Sistem menampilkan grafik, nilai kualitas udara dan terdapat keterangan kualitas udara yang *auto refresh* tanpa harus memperbarui *website.*	v	v	v			
	User menekan nama *user.*	Sistem menampilkan *button* untuk keluar dari *website* pemantaun kualitas udara dan kembali	v	v	v			

		ke *landing page* / halaman awal.						
Halaman *History* Setelah *Login.*	*User* menekan menu *history.*	Sistem menampilkan 200 data kualitas udara terbaru yang tersimpan di *database* dan menampilkan *button export* data excel.	v	v	v			
	User menekan *button export* excel.	Sistem mengambil maksimal 8000 data kualitas udara terbaru yang tersimpan dan menjadikan *file* excel.	v	v	v			
Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi,	v	v	v			

		spesifikasi sistem dan manfaat dari sistem yang dibuat.						

1. Kesimpulan Hasil *Black Box Testing*

   Berdasarkan *black box testing* dari 14 pengujian pada sistem pemantauan kualitas udara yang didapat dari 3 responden. Berikut hasil dari pengujian *black box testing.*

   a) Pengujian pertama

      Nama Penguji : Noora Qotrun Nada, S.T., M.Eng.

      Diterima : $\frac{14}{14} \times 100\% = 100\%$

      Ditolak : $\frac{0}{14} \times 100\% = 0\%$

   b) Pengujian Kedua

      Nama Penguji : Ramadhan Renaldy, S.Kom., M.Kom.

      Diterima : $\frac{14}{14} \times 100\% = 100\%$

      Ditolak : $\frac{0}{14} \times 100\% = 0\%$

   c) Pengujian Ketiga

      Nama Penguji : Nur Latifah Dwi MS, M.Kom

      Diterima : $\frac{14}{14} \times 100\% = 100\%$

      Ditolak : $\frac{0}{14} \times 100\% = 0\%$

   Jumlah presentase rata-rata diterima = $\frac{300\%}{3} = 100\%$

   Jumlah presentase rata-rata ditolak = $\frac{0\%}{3} = 0\%$

Hasil dari perhitungan diatas yang diperoleh dari 14 jenis pengujian dan dilakukan oleh 3 responden. Maka dapat disimpulkan pengujian *black box* berhasil mencapai tingkat keberhasilan sempurna sebesar 100% tanpa kegagalan. Dengan demikian sistem pemantauan kualitas udara berfungsi sesuai dengan fungsionalitas yang diharapkan.

c. *User Acceptence Testing* (UAT)

UAT dilakukan untuk memverifikasi bahwa sistem pemantauan kualitas udara memenuhi persyaratan dan ekspektasi pengguna. Proses pengujian ini memberikan kesempatan bagi pengguna untuk memberikan umpan balik dan memastikan bahwa sistem pemantauan kualitas udara siap untuk digunakan dilingkungan masyarakat ataupun bisnis. Pengujian ini melibatkan 3 responden yang diminta untuk mengisi kuisioner skala likert dari 1 hingga 5. Berikut adalah hasil kuisioner pengujian UAT yang telah diisi oleh 3 responden. Hasil pengujian UAT sistem pemantauan kualitas udara dapat dilihat pada Tabel 4.3.

Tabel 4. 3. Hasil *User Acceptance Testing* (UAT)

Pertanyaan	**Hasil Pengujian**		
	Responden 1	**Responden 2**	**Responden 3**
No.1	4	5	4
No.2	5	5	5
No.3	4	4	3
No.4	4	4	4
No.5	5	5	4
No.6	5	3	5
No.7	4	4	3
No.8	4	4	5
No.9	4	5	4
No.10	5	3	5
Jumlah skor	44	42	42

Presentase	88%	84%	84%
Total	256%		

Keterangan:

Sangat Setuju = 5

Setuju = 4

Cukup Setuju = 3

Kurang Setuju = 2

Tidak Setuju = 1

Dari hasil presentase yang diberikan oleh 3 responden untuk setiap pertanyaan yang menacakup asepek kegunaan, aspek kemudahan pengguna dan *user interface* (UI) atau tampilan. Kemudian mencari nilai rata-ratanya. Tujuannya adalah untuk mengetahui Tingkat penerimaan sistem pemantauan kualitas udara yang dikembangkan oleh responden. Nilai rata-rata ini dapat dihitung menggunakan persamaan berikut.

$$\text{Presentase rata-rata} = \frac{Jumlah\ total\ presentase}{Jumlah\ responden}$$

$$\text{Presentase rata-rata} = \frac{256\%}{3} = 85{,}33\%$$

Keterangan kategori presentase:

0% – 20% = Sangat Kurang

21% - 40% = Kurang

41% - 60% = Cukup

61% - 80% = Baik

81% - 100% = Sangat Baik

Dari perhitungan diatas, diperoleh presentase dari ketiga aspek yaitu sebesar 85,33%. Dengan demikian, dapat disimpulkan bahwa pengujian UAT pada sistem pemantauan kualitas udara ini mendapat kategori "Sangat Baik". Namun pada pengujian UAT ini terdapat hasil poin bernilai poin 3 (cukup), diantaranya pengujian no.3, 6, 7, dan 10.

B. Pembahasan

Pembahasan dalam penelitian sistem pemantauan kualitas udara berbasis IoT menggunakan NodeMCU dengan *interface website* dan penggunaan metode *waterfall* adalah bagian yang esensial untuk menganalisis dan menginterpretasikan hasil dari tiap tahapan pengembangan sistem. Pada bagian ini, peneliti menjelaskan bagaimana setiap tahap dalam model waterfall. Mulai dari *requirement analysis*, *design*, *implementation* dan *testing* terhadap tercapainya tujuan penelitian.

1. *Requirement Analysis*

Pada tahap pertama yaitu tahap *requirement analysis* atau analisis kebutuhan, dilakukan pengumpulan, pemahaman, dan penjelasan secara rinci tentang kebutuhan yang diperlukan untuk pengembangan sistem pemantauan kualitas udara. Beberapa jenis analisis kebutuhan termasuk analisis kebutuhan perangkat keras, analisis kebutuhan perangkat lunak, analisis kebutuhan fungsional dan analisis kebutuhan non fungsional. Kebutuhan perangkat keras yang dibutuhkan antara lain : laptop, NodeMCU ESP8266, sensor MQ-135, *breadboard*, 2 lampu LED beda warna (merah dan hijau), resistor 220Ω ± 5% dan kabel jumper. Pada perangkat lunak membutuhkan beberapa *software* diantaranya : Arduino IDE, Laravel versi 10 dengan php versi 8.1, Websockets, Xampp, *Database* MySQL, dan Visual Studio Code untuk mengolah kode *website*. Kebutuhan fungsional berfokus pada fitur spesifik yang harus disediakan sistem, seperti tampilan *real-time* data kualitas udara yang mudah dipahami. Kebutuhan non-fungsional mencakup aspek seperti performa, keamanan, skalabilitas, dan keandalan, yang menentukan kualitas operasi sistem. Contohnya, pengguna harus *login* untuk memastikan akses data terbatas pada pihak yang berwenang, melindungi privasi dan integritas informasi. Langkah-langkah ini bertujuan untuk mengidentifikasi dan merinci kebutuhan sistem pemantauan kualitas udara secara menyeluruh.

2. *Design*

Setelah menyelesaikan tahap pengumpulan kebutuhan sistem, langkah selanjutnya adalah tahap desain. Dalam tahap ini, terdapat 3 jenis desain diantaranya: *design* alat, *design system* dan *design user interface* (UI). Desain alat ini adalah gambaran rangkaian komponen *internet of things* yang akan dibuat untuk untuk pengampilan data kualitas udara. Pada desain alat terdapat komponen NodeMCU ESP8266, sensor MQ-135, *breadboard*, resistor dan lampu LED yang dihubungkan menggunakan kabel *jumper*.

Pada desain sistem yang menggunakan model *Unified Modeling Language* (UML) seperti *use case* diagram, *activity* diagram, *sequence* diagram, dan *class* diagram menjadi penting. Melalui diagram-diagram tersebut, penulis dapat mengidentifikasi fungsi-fungsi aplikasi serta alur kerja yang terjadi di dalamnya. *Use case* diagram membantu dalam memahami interaksi pengguna dengan sistem. *Activity* diagram menjelaskan langkah-langkah proses secara rinci. *Sequence* diagram menggambarkan interaksi antara objek dalam waktu tertentu. Kemudian *class* diagram menunjukkan struktur sistem dalam hal kelas-kelas dan hubungan antar kelas tersebut.

Pada desain *user interface* (UI) dibuat antarmuka yang menarik, intuitif, dan mudah digunakan oleh pengguna. Desain UI berfokus pada tata letak visual, interaksi pengguna, dan navigasi dalam sebuah aplikasi atau situs web. Desain UI ini dibuat untuk memastikan bahwa pengguna dapat dengan mudah berinteraksi dengan aplikasi atau situs web, sehingga meningkatkan kepuasan dan keterlibatan mereka. Dengan menggunakan hirarki visual dan tata letak yang efektif, desain UI membantu pengguna memahami dan mengakses informasi dengan cepat dan efisien. Desain UI ini meningkatkan konversi dengan memandu pengguna melalui proses seperti *login* sistem pemantauan, melihat berita, atau tindakan lain yang diinginkan.

3. *Implementation*

Implementasi sistem pemantauan kualitas udara berbasis *Internet of Things* menggunakan NodeMCU dengan *interface website* melibatkan perancangan perangkat keras dan perangkat lunak. Perangkat keras meliputi sensor MQ-135 yang dihubungkan ke pin A0 (analog input 0) pada NodeCMU untuk mendeteksi gas berbahaya di udara. NodeMCU sebagai mikrokontroler untuk mengolah dan mengirim data kualitas udara ke *server website*. Serta 2 buah resistor yang dihubungkan ke masing-masing kaki anoda LED dan untuk kaki katoda LED dihubungkan ke G (*ground*) NodeMCU. Lampu LED ini digunakan sebagai indikator kualitas udara berdasarkan ambang batas tertentu, yaitu lampu LED hijau jika kualitas udara <100 ppm dan lampu LED merah jika kualitas udara >100 ppm. Resistor yang digunakan resistensi sebesar 220 Ω ± 5% yang berfungsi untuk melindungi lampu LED dari arus yang berlebih.

Di sisi perangkat lunak, API dikembangkan menggunakan Laravel untuk menerima data dari NodeMCU dan menyimpannya di basis data MySQL, serta menampilkan data secara *real-time*. WebSocket digunakan untuk memastikan pembaruan data otomatis di antarmuka web, sehingga pengguna dapat melihat perubahan kualitas udara secara langsung tanpa perlu me-*refresh* halaman. Dengan fitur broadcast data dan sensor update, setiap kali ada data baru dari sensor, *serve*r akan mengirim notifikasi ke semua klien yang terhubung dan memperbarui tampilan data pada halaman web secara otomatis. Sistem ini tidak hanya menyediakan informasi yang akurat dan *real-time* mengenai kualitas udara, tetapi juga memungkinkan pengguna untuk memantau dan merespons perubahan kualitas udara dengan cepat, menjadikannya alat yang penting untuk menjaga kesehatan dan keselamatan lingkungan sekitar.

Pembuatan *website* sistem pemantauan kualitas udara yang interaktif dilakukan untuk memberikan akses yang mudah dan *real-time* kepada pengguna dalam memantau kondisi kualitas udara di lingkungan mereka.

Dengan adanya antarmuka yang interaktif, pengguna dapat dengan cepat memahami informasi yang disajikan, seperti tingkat polusi udara, melalui visualisasi data yang intuitif dan responsif. Hal ini sangat penting dalam konteks kesehatan masyarakat, di mana informasi yang akurat dan terkini dapat membantu individu dan komunitas dalam mengambil tindakan yang tepat untuk melindungi diri mereka dari dampak buruk polusi udara. Selain itu, interaktivitas pada *website* memungkinkan pengguna untuk mendapatkan notifikasi atau pembaruan otomatis tanpa perlu melakukan penyegaran halaman secara manual, sehingga meningkatkan kenyamanan dan efisiensi dalam penggunaan. Dengan fitur-fitur tersebut, *website* ini tidak hanya berfungsi sebagai alat *monitoring*, tetapi juga sebagai *platform* edukasi dan pemberdayaan bagi pengguna untuk lebih sadar akan pentingnya menjaga kualitas udara dan lingkungan sekitar mereka.

4. *Testing*

Tahap terakhir adalah *testing* atau pengujian. Bertujuan untuk memastikan kualitas sistem yang berfungsi dengan baik. Penulis melakukan 3 jenis pengujian *White Box Testing, Black Box Testing* dan *User Acceptance Testing* (UAT). Pengujian *white box* menggunakan teknik *path testing*. Pengujian *white box* ini hanya terfokus pada pengujian dari pengambilan kualitas udara yang dilakukan oleh perangkat *internet of things* (IoT) sampai dengan penampilan data kualitas udara di *website* sistem pemantauan kualitas udara. Hasil dari pengujian ini tercapai dengan melakukan pengujian pada 2 *independent path* dengan rentang nilai 1 sampai 10 yang dikatakan baik karena bukan termasuk kompleks, hal ini menunjukkan bahwa aplikasi ini mudah dalam perawatan. Dalam pengujian *Black Box* dilakukan oleh 3 dosen Program Studi Informatika Universitas PGRI Semarang. Pengujian *black box* berhasil mencapai presentase 100% keberhasilan, sedangkan kegagalan mendapat presentase 0% dari 3 responden dan 14 pengujian. Pengujian *user acceptance test* (UAT) dilakukan oleh 3 orang umum dan menghasilkan presentase sebesar 85,33% dari 3 responden dengan 10 pertanyaan. Dengan demikian, dapat

disimpulkan bahwa pengujian UAT pada sistem pemantauan kualitas udara ini mendapat kategori "Sangat Baik". Namun pada pengujian UAT ini terdapat pengujian yang menghasilkan poin 3 (cukup) yang terdapat pada pengujian no. 3, 6, 7 dan 10. Pada poin no.3 pengujian UAT terdapat pertanyaan apakah sistem pemantauan kualitas udara mudah dioperasikan?. Pertanyaan tersebut bernilai 3 (cukup) dikarenakan untuk dari segi *backend* sudah berfungsi dengan baik, namun *frontend* pada halaman detail berita belum tersesun secara rapi. Pada pertanyaan no.6 apakah sistem pemantauan kualitas udara sesuai dengan kebutuhan anda?, pengujian tersebut diberikan nilai 3 (cukup) dikarenakan sistem tersebut lebih cocok digunakan untuk suatu perusahaan atau instansi dikarenakan pada *website real-time monitoring* tombol keluar terdapat dibalik nama *user*, jika di klik nama tersebut maka tombol keluar tidak akan muncul. Pada pertanyaan no.7 yaitu apakah tampilan kualitas udara mudah untuk dipahami?, pengujian tersebut diberikan poin 3 (cukup) dikarenakan terdapat menu data kulaitas udara yang belum diberikan informasi keterangan setiap kualitas udara. Pada pengujian UAT no.10 yaitu apakah sistem pemantauan kualitas udara perlu dikembangkan lagi?, pengujian tersebut ada salah satu diberi nilai 3 (cukup) dikarenakan orang tersebut menganggap sistem tersebut sudah cukup baik jika tidak dikembangkan lagi. Tetapi jika ingin dikembangkan maka akan lebih baik lagi. Pada setiap masukan dari Masyarakat sudah saya pertimbangkan dan untuk sistem sudah saya perbaiki sesuai dengan hasil pengujian UAT.

BAB V
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian yang telah dilakukan oleh penulis, maka dapat ditarik beberapa kesimpulan sebagai berikut.

1. Sistem pemantauan kualitas udara berbasis *Internet of Things* (IoT) dengan *interface website* yang dikembangkan menawarkan solusi yang canggih dan efisien untuk memantau kualitas udara secara *real-time*. Menggunakan NodeMCU dan sensor MQ-135, sistem ini mampu mendeteksi berbagai jenis gas berbahaya dan mengirim data tersebut secara terus-menerus ke *server*. Kemudian jika nilai kualitas udara berbahaya atau >100 ppm maka akan terdapat indikator lampu LED berwarna merah. Teknologi Laravel 10 dan PHP 8.1 digunakan untuk mengembangkan web *interface*, sementara WebSocket memungkinkan pembaruan data yang cepat dan *seamless*.
2. Fitur utama dalam *website* sistem pemantauan kualitas udara yang dibuat penulis adalah *websocket* dan *broadcasting*. *Websocket* memungkinkan pembaruan data secara *real-time*, sehingga pengguna dapat melihat data kualitas udara yang terkini tanpa perlu me-*refresh* halaman. *Broadcast* yang digunakan dibagi menjadi 2 macam, yaitu *broadcast chart* untuk data kualitas udara dalam bentuk grafik yang diperbarui secara otomatis dan *broadcast* data kualitas udara untuk informasi kualitas udara disiarkan secara *real-time* ke semua pengguna yang sedang mengakses web *interface*. Kemudian data kualitas udara yang tampil terdapat keterangan dan bisa berubah sesuai dengan kualitas udara tanpa harus me-*refresh* halaman.
3. Pengembangan *website* sistem pemantauan kualitas udara yang interaktif merupakan langkah krusial dalam memberikan informasi yang akurat dan *real-time* kepada pengguna. Dengan fitur-fitur interaktif, seperti pemantauan langsung, informasi berita, dan visualisasi data yang

responsif, pengguna dapat dengan mudah memahami kondisi kualitas udara di sekitarnya dan mengambil tindakan yang tepat. Sistem ini tidak hanya meningkatkan kesadaran publik tentang pentingnya kualitas udara, tetapi juga mendukung pengambilan keputusan yang lebih baik untuk kesehatan dan lingkungan. Interaktivitas dalam sistem ini memastikan bahwa informasi yang disajikan selalu relevan dan dapat diakses dengan mudah, memberikan manfaat yang signifikan bagi masyarakat dan lingkungan.

4. Penulis melakukan 3 jenis pengujian untuk memastikan kualitas sistem pemantauan kualitas udara. Pengujian tersebut meliputi *white box testing*, *black box testing*, dan *user acceptance testing* (UAT). Pengujian *white box* juga tercapai dengan melakukan pengujian pada 2 *independent path* itu dikatakan baik karena bukan termasuk kompleks. Hasil pengujian *black box* menunjukkan keberhasilan 100%, sedangkan kegagalan mencapai 0% yang dihasilkan dari 3 responden dalam 14 pengujian. Pengujian *user acceptance test* (UAT) berhasil mencapai 85,33% yang menandakan sistem pemantauan kualitas udara sangat baik. Tetapi dalam pengujian UAT terdapat poin 3 (cukup) pada pengujian no.3, 6, 7, 10. Namun penulis sudah memperbaiki sesuai dengan hasil pengujian UAT.

B. Saran

Untuk meningkatkan kinerja dan fungsionalitas sistem pemantauan kualitas udara berbasis IoT dan web ini, beberapa saran berikut dapat dipertimbangkan:

1. Implementasikan notifikasi *real-time* melalui email, SMS, atau *push notification* ketika kualitas udara mencapai tingkat yang berbahaya atau terjadi perubahan signifikan.
2. Tambahkan sensor lain seperti DHT22 untuk mengukur suhu dan kelembaban, atau sensor PM2.5 untuk mengukur partikel debu. Hal tersebut dapat memberikan data yang lebih komprehensif tentang kondisi lingkungan.

DAFTAR PUSTAKA

[1] D. L. Pristiandaru, "Laporan IQAir: Kualitas Udara Indonesia Terburuk se-Asia Tenggara," Kompas.com, 2024.

[2] W. Wilianto and A. Kurniawan, "SEJARAH, CARA KERJA DAN MANFAAT INTERNET OF THINGS," *Matrix - Jurnal Manajemen Teknologi dan Informatika,* vol. 8, no. 2, 2018.

[3] B. Satria, "IoT Monitoring Suhu dan Kelembaban Udara dengan Node MCU ESP8266," *Sudo Jurnal Teknik,* vol. 1, no. 3, 2022.

[4] I. Fadlurrahman, "10 Provinsi dengan Kualitas Udara Terburuk di Indonesia per 1 Juni 2024 Pukul 16.00 WIB," Kadata Media Network, 2024.

[5] R. Lindsey, "Climate Change: Atmospheric Carbon Dioxide," Climate.gov, 2024.

[6] d. Fadli, "halodoc," 2 November 2022. [Online]. Available: https://www.halodoc.com/artikel/tak-hanya-banyak-fungsinya-ini-bahaya-co2-pada-tubuh-manusia. [Accessed 23 May 2024].

[7] E. F. Santika, "Proporsi Kontribusi Emisi CO2 Terkait Enegeri Berdasarkan Sektor (2021)," Kadata Media Network, 2023.

[8] J. Hadihardaja, Rekayasa Lingkungan, Jakarta: Gunadarma, 1997.

[9] R. P. Bachtera, H. S. Huboyo and B. P. Samadikun, "UJI COBA ESTIMASI EMISI KENDARAAN BERMOTOR YANG BEROPERASI DI KOTA SEMARANG BERDASARKAN UMUR DAN JENIS KENDARAAN DENGAN MENGGUNAKAN PERANGKAT LUNAK LEAP," *Jurnal Teknik Lingkungan,* vol. 6, no. 3, 2017.

[10] W. A. Wardhana, Dampak Pencemaran Lingkungan, Yogyakarta: Andi, 2004.

[11] Fardiaz, Polusi Air dan Udara, Yogyakarta: Kanisius, 2008.

[12] A. A. Bachtiar, "Rancang Bangun Sistem Informasi Monitoring dan Evaluasi Pelayanan Pelanggan Pada Paramuda Tour & Transport," 2016. [Online]. Available: http://repository.dinamika.ac.id/id/eprint/2044.

[13] Mercy, Design, Monitoring and Evaluation Guidebook, Portland, USA: Mercy Corps, 2005.

[14] R. R. Abdullah and A. Wibowo, "Monitoring Suhu Ruangan Server Dengan Fuzzy Logic Metode Sugeno Menggunakan Arduino dan SMS," *SWABUMI,* vol. 1, no. 1, 2014.

[15] E. D. Meutia, "Internet of Things – Keamanan dan Privasi," *Seminar Nasional dan Expo Teknik Elektro,* no. 85-89, 2015.

[16] D. Chandra, "PENGENALAN INTERNET OF THINGS (IOT) DAN MANFAATNYA DI MASA SEKARANG," 17 October 2021.

[17] M. V. Overbeek, "Internet of Things (IoT) dalam Bidang Informatika," 14 June 2019.

[18] S. Salmon, A. Y. Rangan and B. A. Ramadhan, "RANCANG BANGUN SISTEM KEMANAN RUMAH DENGAN MENGGUNAKAN MODULE NODEMCU BERBASIS IOT (INTERNET OF THINGS)," *Jurnal Informatika Wicida,* vol. 12, no. 2, pp. 48-54, 2022.

[19] P. Code, "PANDUAN CODE," [Online]. Available: https://www.panduancode.com/2023/10/esp8266-pinout.html#gsc.tab=0. [Accessed 2 June 2024].

[20] A. F. Sibero, Web Programming Power Pack, Yogyakarta: Mediakom, 2014.

[21] P. Hidayatullah and J. K. Kawistara, Pemrograman Web, Bandung: Informatika Bandung, 2015.

[22] V. Novi, "Gramedia Page," [Online]. Available: https://www.gramedia.com/literasi/laravel/. [Accessed 2 June 2024].

[23] Ecadio. [Online]. Available: https://ecadio.com/jual-sensor-gas-mq-135. [Accessed 20 May 2024].

[24] K. Pratama and E. B. Setiawan, "Menggunakan Peramalan Exponential Smoothing dan NodeMCU Berbasis Mobile Android," *ULTIMA COMPUTING : JURNAL SISTEM KOMPUTER ,* vol. 9, no. 2, 2017.

[25] D. W. T. Putra and R. Andriani, "Unified Modelling Language (UML) dalam Perancangan SistemInformasi Permohonan Pembayaran Restitusi SPPD," *Jurnal TEKNOIF,* vol. 7, no. 1, 2019.

[26] I. Zufria, "Pemodelan Berbasis UML (Unified Modeling Language) dengan," *J. Sains Teknol,* pp. 1-16, 2013.

[27] C. D. Ariyani and B. A. Herlambang, "Aplikasi E-office Berbasis Mobile pada Balai DIKLAT Keagamaan," *Science And Engineering National Seminar 7 (SENS 7),* 2022.

[28] Meilinaeka, "Telkom University," [Online]. Available: https://it.telkomuniversity.ac.id/metode-waterfall-dalam-pengembangan-perangkat-lunak/. [Accessed 21 May 2024].

[29] T. Butler and K. Yank, PHP & MySQL: Novice to Ninja, 6th Edition, SiitePoing, 2016.

[30] I. G. M. Budiarta and I. N. Sila, "PEMANFAATAN APLIKASI CORELDRAW SEBAGAI MEDIA PEMBELAJARAN PADA KULIAH DESAIN KOMUNIKASI VISUAL PRODI PENDIDIKAN SENI RUPA UNDIKSHA," *Jurnal Pendidikan Seni Rupa Undiksha,* vol. 12, no. 2, pp. 115-128, 2022.

[31] Dicoding. [Online]. Available: https://www.dicoding.com/blog/white-box-testing/. [Accessed 2 June 2024].

[32] T. Hamilton, "Guru99," [Online]. Available: https://www.guru99.com/id/cyclomatic-complexity.html. [Accessed 20 08 2024].

[33] M. Andriana, "User Acceptance Test," *Binus Popular Articles,* 28 October 2020.

[34] Okpatrioka, "Research And Development (R&D) Penelitian Yang Inovatif Dalam Pendidikan," *DHARMA ACARIYA NUSANTARA : Jurnal Pendidikan, Bahasa dan Budaya,* vol. 1, no. 1, pp. 86-100, 2023.

LAMPIRAN

Lampiran 1 Lembar Bimbingan Dosen Pembimbing 1

UNIVERSITAS PGRI SEMARANG
FAKULTAS TEKNIK DAN INFORMATIKA
Kampus : Jalan Sidodadi Timur Nomor 24 Dr. Cipto, Semarang – Indonesia 50125
Telp. (024) 8316377, **Faks.** (024) 8448217, **E-mail** : upgrismg@gmail.com, **Homepage :** www.upgrismg.ac.id

LEMBAR PEMBIMBINGAN SKRIPSI

Nama Mahasiswa	:	Ahmad Syaifulloh
NPM	:	20670116
Program Studi	:	Informatika
Judul Skripsi	:	Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*
Dosen Pembimbing I	:	Mega Novita, S.Si., M.Si., M.Nat.Sc., Ph.D.
Dosen Pembimbing II	:	Ir. Agung Handayanto, M.Kom.

No.	Hari Tanggal	Uraian Bimbingan	Paraf
1.	3-5-2029	Bimbingan Judul	
2.	7-5-2029	Acc Judul	
3.	20-5-2029	Revisi Bab I - Bab III	
4.	1-6-2029	Revisi Bab I & Bab II	
5.	10-6-2029	Acc Bab I & Bab II, revisi Bab III	
6.	13-6-2029	Acc proposal	
7.	29-6-2029	penambahan frontend website	
8.	9-7-2029	Revisi frontend website	
9.	30-7-2029	Acc Sistem aplikasi	
10.	5-8-2029	Acc Bab IV	
11.	6-8-2029	Acc Bab V & Acc Skripsi	

Dosen Pembimbing I,

Mega Novita, S.Si., M.Si., M.Nat.Sc., Ph.D.
NIDN. 0615118801

Mahasiswa,

Ahmad Syaifulloh
NPM. 20670116

Lampiran 2 Lembar Bimbingan Dosen Pembimbing 2

UNIVERSITAS PGRI SEMARANG
FAKULTAS TEKNIK DAN INFORMATIKA
Kampus : Jalan Sidodadi Timur Nomor 24 Dr. Cipto, Semarang – Indonesia 50125
Telp. (024) 8316377, **Faks.** (024) 8448217, **E-mail** : upgrismg@gmail.com, **Homepage :** www.upgrismg.ac.id

LEMBAR PEMBIMBINGAN SKRIPSI

Nama Mahasiswa	:	Ahmad Syaifulloh
NPM	:	20670116
Program Studi	:	Informatika
Judul Skripsi	:	Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*
Dosen Pembimbing I	:	Mega Novita, S.Si., M.Si., M.Nat.Sc., Ph.D.
Dosen Pembimbing II	:	Ir. Agung Handayanto, M.Kom.

No.	Hari Tanggal	Uraian Bimbingan	Paraf
1.	7-5-2024	Acc Judul	
2.	11-6-2024	Revisi Bab I & Bab II	
3.	10-7-2024	Revisi Bab III & penambahan fitur history pada website.	
4.	15-7-2024	Acc proposal	
5.	29-7-2024	Revisi tampilan website	
6.	31-7-2024	Acc Sistem website	
7.	9-8-2024	Revisi Bab 4 & Bab 5	
8.	12-8-2024	Acc Bab 4 & Bab 5	

Dosen Pembimbing II,

Ir. Agung Handayando, M.Kom.
NIDN. 0019116202

Mahasiswa,

Ahmad Syaifulloh
NPM. 206701116

Lampiran 3 Form Pengujian Black Box

Nama Pengujian	Hasil yang Diharapkan	Hasil yang Didapatkan	Keterangan	
			Diterima	Ditolak
Program di Arduino	*Admin* inisialisasi perangkat IoT	Data kualitas udara dapat dikirimkan ke *server website* secara *real-time*.		
	Melihat lampu led pada rangkaian IoT.	Lampu LED nyala sesuai dengan kualitas udara. Jika nilai kualitas udara kurang dari 100 ppm maka lampu LED nyala warna hijau. Jika nilai kualitas udara melebihi 100 ppm maka lampu LED nyala warna merah.		
Halaman *LandingPage* / Tampilan Awal Saat Menjalankan *Server*	*User* melihat tampilan *landingpage* yang terdapat beberapa tampilan	Sistem menampilkan beberapa button (*login,* berita, tentang dan kontak), berita		

	seperti berita, kontak dan tentang sistem aplikasi.	tentang sistem dan kontak *developer*.		
	User dapat melakukan *login.*	Sistem menampilkan halaman *login* untuk masuk ke sistem pemantauan.		
	User menekan *button* berita.	Sistem menampilkan semua berita.		
	User menekan link baca selengkapnya.	Sistem menampilkan detail berita.		
	User menekan email pada menu kontak.	Sistem menampilkan alamat email *developer.*		
	User menekan *button* Tentang.	Sistem menampilkan gambaran singkat mengenai sistem yang dibuat.		

Halaman Setelah *Login*	*User* menekan menu *Real-Time Monitoring.*	Sistem menampilkan *drop down* beberapa menu data kualitas udara seperti *air quality,* CO, CO^2, dan *smoke.*		
	User menekan data kualitas udara.	Sistem menampilkan grafik dan tabel data kualitas udara yang dikirimkan dari perangkat IoT.		
	User melihat grafik dan nilai kualitas udara.	Sistem menampilkan grafik, nilai kualitas udara dan terdapat keterangan kualitas udara yang *auto refresh* tanpa harus memperbarui *website.*		

	User menekan nama *user.*	Sistem menampilkan *button* untuk keluar dari *website* pemantaun kualitas udara dan kembali ke *landing page* / halaman awal.		
Halaman *History* Setelah *Login.*	*User* menekan menu *history.*	Sistem menampilkan 200 data kualitas udara terbaru yang tersimpan di *database* dan menampilkan *button export* data excel.		
	User menekan *button export* excel.	Sistem mengambil maksimal 8000 data kualitas udara terbaru yang tersimpan dan menjadikan *file* excel.		

Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi, spesifikasi sistem dan manfaat dari sistem yang dibuat.		

Lampiran 4 Hasil Pengujian Black Box

Nama :
NPM :

Kuisioner *Black Box* Pada Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*

Nama Penguji : Noora Qotrun Nada.

Tanggal Pengujian : 31 ~~Agustus~~ Juli 2024.

Nama Pengujian	Hasil yang Diharapkan	Hasil yang Didapatkan	Keterangan	
			Diterima	Ditolak
Program di Arduino	*Admin* inisialisasi perangkat IoT	Data kualitas udara dapat dikirimkan ke *server website* secara *real-time*.	✓	
	Melihat lampu led pada rangkaian IoT.	Lampu LED nyala sesuai dengan kualitas udara. Jika nilai kualitas udara kurang dari 100 ppm maka lampu LED nyala warna hijau. Jika nilai kualitas udara melebihi 100 ppm maka lampu LED nyala warna merah.	✓	

Halaman *LandingPage* / Tampilan Awal Saat Menjalankan *Server*	*User* melihat tampilan *landingpage* yang terdapat beberapa tampilan seperti berita, kontak dan tentang sistem aplikasi.	Sistem menampilkan beberapa button (*login*, berita, tentang dan kontak), berita tentang sistem dan kontak *developer*.	✓	
	User dapat melakukan *login*.	Sistem menampilkan halaman *login* untuk masuk ke sistem pemantauan.	✓	
	User menekan *button* berita.	Sistem menampilkan semua berita.	✓	
	User menekan link baca selengkapnya.	Sistem menampilkan detail berita.	✓	
	User menekan email pada menu kontak.	Sistem menampilkan alamat email *developer*.	✓	
	User menekan *button* Tentang.	Sistem menampilkan gambaran singkat mengenai	✓	

		sistem yang dibuat.		
Halaman Setelah *Login*	*User* menekan menu *Real-Time Monitoring.*	Sistem menampilkan *drop down* beberapa menu data kualitas udara seperti *air quality,* CO, CO^2, dan *smoke.*	✓	
	User menekan data kualitas udara.	Sistem menampilkan grafik dan tabel data kualitas udara yang dikirimkan dari perangkat IoT.	✓	
	User melihat grafik dan nilai kualitas udara.	Sistem menampilkan grafik, nilai kualitas udara dan terdapat keterangan kualitas udara yang *auto refresh* tanpa harus memperbarui *website.*	✓	

	User menekan nama *user*.	Sistem menampilkan *button* untuk keluar dari *website* pemantaun kualitas udara dan kembali ke *landing page* / halaman awal.	✓	
Halaman *History* Setelah *Login*.	*User* menekan menu *history*.	Sistem menampilkan 200 data kualitas udara terbaru yang tersimpan di *database* dan menampilkan *button export* data excel.	✓	
	User menekan *button export* excel.	Sistem mengambil maksimal 8000 data kualitas udara terbaru yang tersimpan dan menjadikan *file* excel.	✓	

Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi, spesifikasi sistem dan manfaat dari sistem yang dibuat.	✓	

Kritik dan saran

- Beri beda tupilan. Saat · kondisi air (udara) Unhealthy.
- Bisa query lap. y. per bulan. dan. ditambah dgn warna. y. yg unhealthy.

Semarang,

Noora Stk.

NPP/NIDN.

Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi, spesifikasi sistem dan manfaat dari sistem yang dibuat.	✓	

Kritik dan saran

Frontend nya diperbagus, UInya ditata kembali, Uxnya diperbaiki agar user nyaman.

Semarang, 31 Juli 2024

Ramadhan Renaldy

NPP/NIDN. 24990165 9

Kuisioner *Black Box* Pada Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*

Nama Penguji : Ramadhan Renaldy, S.Kom, M.Kom

Tanggal Pengujian : 31 Juli 2024

Nama Pengujian	Hasil yang Diharapkan	Hasil yang Didapatkan	Keterangan	
			Diterima	Ditolak
Program di Arduino	*Admin* inisialisasi perangkat IoT	Data kualitas udara dapat dikirimkan ke *server website* secara *real-time*.	✓	
	Melihat lampu led pada rangkaian IoT.	Lampu LED nyala sesuai dengan kualitas udara. Jika nilai kualitas udara kurang dari 100 ppm maka lampu LED nyala warna hijau. Jika nilai kualitas udara melebihi 100 ppm maka lampu LED nyala warna merah.	✓	

Halaman *LandingPage* Tampilan Awal Saat Menjalankan *Server*	*User* melihat tampilan *landingpage* yang terdapat beberapa tampilan seperti berita, kontak dan tentang sistem aplikasi.	Sistem menampilkan beberapa button (*login*, berita, tentang dan kontak), berita tentang sistem dan kontak *developer*.	✓	
	User dapat melakukan *login*.	Sistem menampilkan halaman *login* untuk masuk ke sistem pemantauan.	✓	
	User menekan *button* berita.	Sistem menampilkan semua berita.	✓	
	User menekan link baca selengkapnya.	Sistem menampilkan detail berita.	✓	
	User menekan email pada menu kontak.	Sistem menampilkan alamat email *developer*.	✓	
	User menekan *button* Tentang.	Sistem menampilkan gambaran singkat mengenai	✓	

		sistem yang dibuat.		
Halaman Setelah *Login*	*User* menekan menu *Real-Time Monitoring*.	Sistem menampilkan *drop down* beberapa menu data kualitas udara seperti *air quality*, CO, CO^2, dan *smoke*.	✓	
	User menekan data kualitas udara.	Sistem menampilkan grafik dan tabel data kualitas udara yang dikirimkan dari perangkat IoT.	✓	
	User melihat grafik dan nilai kualitas udara.	Sistem menampilkan grafik, nilai kualitas udara dan terdapat keterangan kualitas udara yang *auto refresh* tanpa harus memperbarui *website*.	✓	

	User menekan nama *user.*	Sistem menampilkan *button* untuk keluar dari *website* pemantaun kualitas udara dan kembali ke *landing page* / halaman awal.	✓	
Halaman *History* Setelah *Login.*	*User* menekan menu *history.*	Sistem menampilkan 200 data kualitas udara terbaru yang tersimpan di *database* dan menampilkan *button export* data excel.	✓	
	User menekan *button export* excel.	Sistem mengambil maksimal 8000 data kualitas udara terbaru yang tersimpan dan menjadikan *file* excel.	✓	

Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi, spesifikasi sistem dan manfaat dari sistem yang dibuat.	✓	

Kritik dan saran

Frontend aya diperbagus, UInya ditata kembali, Uxnya diperbaiki agar user nyaman.

Semarang, 31 Juli 2024

Ramadhan Renaldy

NPP/NIDN. 24990165 9

Kuisioner *Black Box* Pada Sistem Pemantauan Kualitas Udara Berbasis *Internet of Things* (IoT) Menggunakan NodeMCU Dengan *Interface Website*

Nama Penguji : Nur Latifah Dwi MS, M.Kom

Tanggal Pengujian : 31 Juli 2024

Nama Pengujian	Hasil yang Diharapkan	Hasil yang Didapatkan	Keterangan	
			Diterima	Ditolak
Program di Arduino	*Admin* inisialisasi perangkat IoT	Data kualitas udara dapat dikirimkan ke *server website* secara *real-time*.	✓	
	Melihat lampu led pada rangkaian IoT.	Lampu LED nyala sesuai dengan kualitas udara. Jika nilai kualitas udara kurang dari 100 ppm maka lampu LED nyala warna hijau. Jika nilai kualitas udara melebihi 100 ppm maka lampu LED nyala warna merah.	✓	

Halaman *LandingPage* / Tampilan Awal Saat Menjalankan *Server*	*User* melihat tampilan *landingpage* yang terdapat beberapa tampilan seperti berita, kontak dan tentang sistem aplikasi.	Sistem menampilkan beberapa button (*login*, berita, tentang dan kontak), berita tentang sistem dan kontak *developer*.	✓	
	User dapat melakukan *login*.	Sistem menampilkan halaman *login* untuk masuk ke sistem pemantauan.	✓	
	User menekan *button* berita.	Sistem menampilkan semua berita.	✓	
	User menekan link baca selengkapnya.	Sistem menampilkan detail berita.	✓	
	User menekan email pada menu kontak.	Sistem menampilkan alamat email *developer*.	✓	
	User menekan *button* Tentang.	Sistem menampilkan gambaran singkat mengenai	✓	

		sistem yang dibuat.		
Halaman Setelah *Login*	*User* menekan menu *Real-Time Monitoring*.	Sistem menampilkan *drop down* beberapa menu data kualitas udara seperti *air quality*, CO, CO^2, dan *smoke*.	✓	
	User menekan data kualitas udara.	Sistem menampilkan grafik dan tabel data kualitas udara yang dikirimkan dari perangkat IoT.	✓	
	User melihat grafik dan nilai kualitas udara.	Sistem menampilkan grafik, nilai kualitas udara dan terdapat keterangan kualitas udara yang *auto refresh* tanpa harus memperbarui *website*.	✓	

	User menekan nama *user*.	Sistem menampilkan *button* untuk keluar dari *website* pemantaun kualitas udara dan kembali ke *landing page* / halaman awal.	✓	
Halaman *History* Setelah *Login*.	*User* menekan menu *history*.	Sistem menampilkan 200 data kualitas udara terbaru yang tersimpan di *database* dan menampilkan *button export* data excel.	✓	
	User menekan *button export* excel.	Sistem mengambil maksimal 8000 data kualitas udara terbaru yang tersimpan dan menjadikan *file* excel.	✓	

Halaman *About* Setelah *Login*	*User* menekan menu *about*.	Sistem menampilkan tentang aplikasi dari latar belakang pembuatan aplikasi, spesifikasi sistem dan manfaat dari sistem yang dibuat.	✓	

Kritik dan saran

Diperbaiki lagi tampilan biar lebih menarik.

Semarang, 31 Juli 2024

Nur Lutfian, ST, MS, M.Kom.

NPP/NIDN. 0623089002

Lampiran 5 Form Pengujian UAT

No.	Pertanyaan	Keterangan				
		Sangat Setuju	**Setuju**	**Cukup Setuju**	**Kurang Setuju**	**Tidak Setuju**
Aspek Kegunaan						
1.	Apakah sistem pemantauan kualitas udara dapat bermanfaat bagi pengguna?					
2.	Apakah sistem pemantauan kualitas udara memberikan informasi tentang kondisi kualitas udara saat ini?					
Aspek Kemudahan Pengguna						
3.	Apakah sistem pemantauan kualitas udara mudah dioperasikan?					
4.	Apakah sistem pemantauan kualitas udara sesuai yang diharapkan?					
5.	Apakah sistem pemantauan kualitas udara berisi informasi yang dibutuhakan?					
6.	Apakah sistem pemantauan kualitas udara sesuai dengan keperluan anda?					
Aspek *User Interface* (UI) atau Tampilan						

7.	Apakah tampilan sistem pemantauan kualitas udara mudah untuk dipahami?					
8.	Apakah sistem pemantauan kualitas udara memiliki tampilan yang menarik?					
9.	Apakah sistem pemantauan kualitas udara memiliki tema yang sesuai?					
10.	Apakah sistem pemantauan kualitas udara perlu dilakukan pengembangan lagi?					

Lampiran 6 Hasil Pengujian UAT

Form User Acceptence Testing (UAT) Sistem Pemanatauan Kualitas Udara

Nama Penguji : Alifia Devi . M.

Tanggal : 9 Agustus 2024

No.	Pertanyaan	Keterangan				
		Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Aspek Kegunaan						
1.	Apakah sistem pemantauan kualitas udara dapat bermanfaat bagi pengguna?		✓			
2.	Apakah sistem pemantauan kualitas udara memberikan informasi tentang kondisi kualitas udara saat ini?	✓				
Aspek Kemudahan Pengguna						
3.	Apakah sistem pemantauan kualitas udara mudah dioperasikan?		✓			
4.	Apakah sistem pemantauan kualitas udara sesuai yang diharapkan?		✓			
5.	Apakah sistem pemantauan kualitas udara berisi informasi yang dibutuhakan?	✓				

6.	Apakah sistem pemantauan kualitas udara sesuai dengan keperluan anda?	✓				
Aspek *User Interface* (UI) atau Tampilan						
7.	Apakah tampilan sistem pemantauan kualitas udara mudah untuk dipahami?		✓			
8.	Apakah sistem pemantauan kualitas udara memiliki tampilan yang menarik?		✓			
9.	Apakah sistem pemantauan kualitas udara memiliki tema yang sesuai?		✓			
10.	Apakah sistem pemantauan kualitas udara perlu dilakukan pengembangan lagi?	✓				

Kritik dan Saran

...Tampilan...sudah...bagus...tetapi...kurang...tampilan...
...pie...chart...pada...menu...real-time monitoring.....
.....

Semarang...9...Agustus...2024

...Aliffia...Devi...N.....

Form User Acceptence Testing (UAT) Sistem Pemanatauan Kualitas Udara

Nama Penguji : Muhammad Prada Hiresando

Tanggal : 4 Agustus 2024

No.	Pertanyaan	Keterangan				
		Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Aspek Kegunaan						
1.	Apakah sistem pemantauan kualitas udara dapat bermanfaat bagi pengguna?	✓				
2.	Apakah sistem pemantauan kualitas udara memberikan informasi tentang kondisi kualitas udara saat ini?	✓				
Aspek Kemudahan Pengguna						
3.	Apakah sistem pemantauan kualitas udara mudah dioperasikan?		✓			
4.	Apakah sistem pemantauan kualitas udara sesuai yang diharapkan?		✓			
5.	Apakah sistem pemantauan kualitas udara berisi informasi yang dibutuhakan?	✓				

6.	Apakah sistem pemantauan kualitas udara sesuai dengan keperluan anda?			✓		
Aspek *User Interface* (UI) atau Tampilan						
7.	Apakah tampilan sistem pemantauan kualitas udara mudah untuk dipahami?		✓			
8.	Apakah sistem pemantauan kualitas udara memiliki tampilan yang menarik?		✓			
9.	Apakah sistem pemantauan kualitas udara memiliki tema yang sesuai?	✓				
10.	Apakah sistem pemantauan kualitas udara perlu dilakukan pengembangan lagi?			✓		

Kritik dan Saran

Sudah bagus.

Semarang, 4 Agustus 2024

M. PRADA M

Form User Acceptence Testing (UAT) Sistem Pemanatauan Kualitas Udara

Nama Penguji : Ilham abu salam

Tanggal : 04 · agustus 2024

No.	Pertanyaan	Keterangan				
		Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Aspek Kegunaan						
1.	Apakah sistem pemantauan kualitas udara dapat bermanfaat bagi pengguna?		✓			
2.	Apakah sistem pemantauan kualitas udara memberikan informasi tentang kondisi kualitas udara saat ini?	✓				
Aspek Kemudahan Pengguna						
3.	Apakah sistem pemantauan kualitas udara mudah dioperasikan?			✓		
4.	Apakah sistem pemantauan kualitas udara sesuai yang diharapkan?		✓			
5.	Apakah sistem pemantauan kualitas udara berisi informasi yang dibutuhakan?		✓			

6.	Apakah sistem pemantauan kualitas udara sesuai dengan keperluan anda?	✓				
Aspek *User Interface* (UI) atau Tampilan						
7.	Apakah tampilan sistem pemantauan kualitas udara mudah untuk dipahami?			✓		
8.	Apakah sistem pemantauan kualitas udara memiliki tampilan yang menarik?	✓				
9.	Apakah sistem pemantauan kualitas udara memiliki tema yang sesuai?		✓			
10.	Apakah sistem pemantauan kualitas udara perlu dilakukan pengembangan lagi?	✓				

Kritik dan Saran

Cukup Baik

Semarang 9 agustus 2019

Nia