

BAB II

TINJAUAN PUSTAKA

Bab tinjauan pustaka berisi penjelasan mengenai teknik, metode, dan konsep yang digunakan dalam penelitian dalam pengembangan prototipe sistem akuaponik. Penjelasan untuk membandingkan penelitian terkait juga terdapat dalam bab ini.

2.1 Penelitian Terkait

Subbab ini membahas penelitian-penelitian terdahulu yang berfokus pada pengembangan sistem akuaponik dengan berbagai konsep dan hasil penelitian. Daftar dari penelitian terkait dapat dilihat pada Tabel 2.1.

Tabel 2.1 Daftar Penelitian Terkait

No	Judul	Penulis	Metode (Software Stack dan Hardware)	Keterangan
1	<i>Real-time Monitoring: The Effect of Using LED Lights on The Growth of Pak Choy (Brassica rapa L. var. chinensis) in Indoor Hydroponic System</i>	Zain dkk. (2024)	Hardware: Arduino Mega 2560, ESP8266 Wi-Fi chip, Sensor LDR (intensitas cahaya), Sensor TDS, Sensor Suhu dan Kelembapan (DHT22), Berbagai variasi warna lampu LED. Software Stack: ThingSpeak (Cloud Database), Aplikasi Mobile berbasis MIT App Inventor.	Menguji pengaruh warna dan intensitas cahaya lampu LED terhadap pertumbuhan tanaman Pakcoy dalam ruangan (indoor) menggunakan metode irigasi <i>Nutrient Film Technique</i> (NFT) dan <i>Deep-Water Culture</i> (DWC) secara <i>real-time</i> .
2	<i>Development of an IoT-based Aquaponics Monitoring System with Automated Feeder</i>	Naputol dkk. (2024)	Hardware: Mikrokontroler ESP32 (TTGO T-Call V1.4), GPRS Network Module (mendukung Nano SIM), Sensor pH (Model 4502C), Sensor Suhu (DS18B20), Sensor Water Level, Water Pump, Automated Feeder, Water Valve Grow Light, , Air Pump. Software Stack: ThingSpeak (Cloud Platform), Web-based Graphical User Interface (GUI).	Mengembangkan sistem pemantauan akuaponik berbasis IoT dengan pemberi pakan otomatis pintar. Penjadwalan pakan disesuaikan berdasarkan jumlah ikan, waktu, dan pembacaan parameter lingkungan.

No	Judul	Penulis	Metode (Software Stack dan Hardware)	Keterangan
3	<i>Prototype NFT/DFT Hydroponic Data Collection Using IoT System</i>	Wijaya Adh dkk. (2022)	Hardware: Mikrokontroler ESP32, Sensor TDS, Sensor pH, Sensor Suhu air, Water Flow Sensor, Water Level Sensor, LCD 16x2, LCD 20x4, Modul I2C, RTC DS3231. Software Stack: Protokol komunikasi MQTT, Database SQL, Web-based Interface (JSON format untuk pengolahan data).	Membangun prototipe sistem pengumpulan data parameter lingkungan hidroponik model NFT dan DFT untuk menjaga konsistensi kualitas nutrisi air secara otomatis tanpa perlu pengecekan manual.
4	<i>A Smart Monitoring And Observation Framework For Aquaponics Ecosystem</i>	Ali dan Hafeez (2024)	Hardware: Mikrokontroler ESP8266, Sensor pH, Kelembapan, Suhu, Sensor Intensitas Cahaya (Sunlight), Sensor Oksigen Terlarut (Oxygen level), Proximity (Water level), LED Light, Water Pump, Fan, Oxygen Pump. Software Stack: C# .NET Core 8.0, React, MySQL, C, Arduino IDE, HTTPS.	Merancang kerangka kerja monitoring akuaponik komprehensif berskala komersial dengan arsitektur tiga lapis (<i>Perception, Network, dan Application Layer</i>) untuk memantau sirkulasi kolam ikan dan bed tanaman secara otonom via web.
5	<i>Iot-Driven Sustainability: Advancing Urban Aquaponics With Smart Water Quality Monitoring</i>	Shukri dkk. (2024)	Hardware: NodeMCU ESP32, Sensor pH Probe, Sensor Gas Amonia (MQ-137), Sensor Suhu air. Software Stack: Blynk IoT Cloud Platform (Aplikasi Android), Analisis Data Statistika dengan metode <i>Principal Component Analysis</i> (PCA).	Mengembangkan sistem monitoring kualitas air akuaponik perkotaan secara <i>real-time</i> . Menggunakan analisis PCA untuk menyimpulkan bahwa kondisi motor pompa air adalah variabel mekanis paling kritis bagi stabilitas ekosistem.
6	<i>Pengembangan Prototipe Sistem Monitoring Akuaponik Berbasis IoT Menggunakan Metodologi Scrum</i>	Penelitian ini	Hardware: Mikrokontroler ESP32 (Arsitektur Master-Slave), Sensor pH (Model 4502C), Sensor Suhu, Kelembapan Udara (DHT22), Sensor Kekeruhan Air (Turbidity), Pompa Irigasi, Pompa Aerator, Lampu Grow Light, Feeder (Pemberi Pakan) Ikan. Software Stack: Framework Laravel, Tailwind CSS, Database MySQL, Protokol	Mengembangkan platform monitoring <i>multi-tenant</i> akuaponik dengan kebaruan Mendukung penambahan lokasi (<i>site</i>), jenis sensor, maupun aktuator baru secara fleksibel di masa depan tanpa harus memodifikasi tabel utama database. Manajemen proyek dikelola secara adaptif menggunakan Scrum.

No	Judul	Penulis	Metode (<i>Software Stack dan Hardware</i>)	Keterangan
			MQTT (HiveMQ Broker), Fonnte (WhatsApp Gateway Notification), <i>Email Gateway</i> , Metodologi Agile Scrum, <i>Black Box Testing</i> .	

Zain dkk. (2024) mengembangkan sistem hidroponik indoor dengan membandingkan dua metode yaitu *Deep Water Culture* (DWC) dan *Nutrient Film Technique* (NFT). Parameter yang diukur meliputi intensitas cahaya, suhu lingkungan, kelembapan lingkungan, serta konsentrasi zat terlarut (TDS) yang diterima tanaman. Data sensor disimpan menggunakan *platform cloud* ThingSpeak, dan hasil monitoring ditampilkan melalui aplikasi mobile secara *real-time*.

Naputol dkk. (2024), mengembangkan sistem monitoring akuaponik berbasis IoT yang dilengkapi dengan pemberi pakan otomatis. Sistem ini membangun jaringan sensor untuk memonitor pH dan suhu air secara *real-time* menggunakan *microcontroller* ESP32 yang mendukung Nano SIM. Sama seperti penelitian sebelumnya, ThingSpeak digunakan sebagai *platform cloud* untuk menyimpan dan menganalisis data, namun tampilan monitoring dilakukan melalui *website*.

Pengembangan sistem yang dilakukan oleh Wijaya Adh dkk. (2022) dalam penelitiannya menyebutkan tiga teknik irigasi hidroponik yaitu NFT, aeroponik, dan DFT, namun hanya teknik DFT yang diimplementasikan dalam sistem yang dibangun. Sistem ini menggunakan ESP32 sebagai *microcontroller* serta beberapa modul pendukung lainnya seperti *Real-time Clock* (RTC), sensor TDS, sensor suhu, dan sensor debit air.

Ali dan Hafeez (2024) mengembangkan sistem yang lebih terintegrasi bernama AEMOS. Dibangun dengan arsitektur tiga lapis, sistem ini menawarkan solusi komprehensif untuk pemantauan, otomatisasi, dan kontrol melalui aplikasi *web*. Dengan fitur manajemen multi-lokasi dan teknologi modern, AEMOS dirancang sebagai platform yang stabil dan siap untuk operasi komersial.

Berbeda dengan penelitian sebelumnya yang berfokus pada arsitektur web dan manajemen multi-lokasi, penelitian yang dilakukan oleh Shukri dkk. (2024) memperkuat aspek keberlanjutan akuaponik melalui integrasi sensor gas amonia (NH₃), pH, dan suhu

berbasis ESP32 yang terhubung ke platform Blynk. Melalui penerapan metode *Principal Component Analysis* (PCA), sistem ini tidak sekadar menyajikan data mentah, tetapi mampu mengidentifikasi bahwa kondisi motor pompa merupakan variabel paling kritis yang memengaruhi stabilitas ekosistem secara keseluruhan. Temuan ini menunjukkan bahwa gangguan pada sirkulasi air berdampak lebih luas terhadap parameter kimia dan fisik dibandingkan faktor sisa makanan atau filter, sehingga pengembangan algoritma pemeliharaan pada komponen mekanis menjadi hal penting dalam meminimalkan risiko kegagalan sistem.

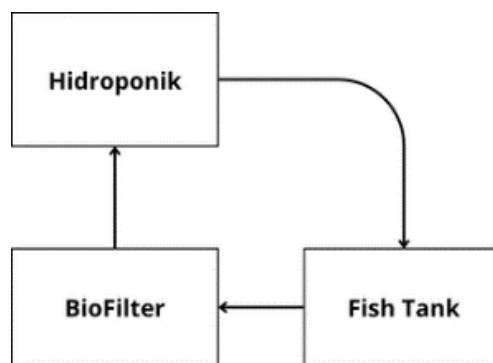
Dari adanya penelitian-penelitian terkait tentang pengembangan sistem *akuaponik* berbasis IoT, sistem perancangan IoT yang dibuat akan mempertimbangkan penggunaan fitur-fitur yang telah dipaparkan pada penelitian-penelitian sebelumnya. Fitur-fitur yang dipilih disesuaikan dengan keadaan dan masalah yang terjadi pada lingkungan penelitian sehingga hasil penelitian ini menjadi sesuai dengan objektif yang direncanakan di awal.

Dari berbagai penelitian terdahulu mengenai sistem hidroponik dan akuaponik berbasis IoT, perancangan sistem yang dibuat dalam penelitian ini akan mengadopsi serta menyempurnakan fitur-fitur tersebut agar sesuai dengan kebutuhan nyata di lapangan, termasuk penggunaan teknik *Nutrient Film Technique* (NFT) yang optimal untuk pertumbuhan tanaman pakcoy. Skripsi ini menawarkan kebaruan berupa arsitektur *website dashboard* berbasis *framework* Laravel yang dirancang dinamis. Melalui implementasi hubungan terstruktur tersebut, aplikasi web ini mampu mendukung penambahan jenis sensor maupun aktuator baru di masa depan secara fleksibel tanpa perlu memodifikasi struktur basis data utama. Ditambah dengan penerapan metodologi Agile Scrum, pengerjaan sistem ini menjamin kolaborasi tim antara seluruh peran dalam Scrum dapat berjalan secara adaptif.

2.2 Sistem Akuaponik

Akuaponik adalah sebuah sistem agrikultur inovatif yang mengintegrasikan akuakultur dengan hidroponik dalam sebuah lingkungan simbiotik (Treeratanaporn dkk., 2025). Sistem ini memanfaatkan sinergi alami antara ikan dan tanaman dimana air yang kaya nutrisi dari wadah ikan digunakan untuk membantu pertumbuhan tanaman. Sebagai timbal baliknya, air akan kembali menjadi jernih setelahnya. Gambar 2.1 adalah gambaran mengenai *waterflow* dalam implementasi sistem akuaponik.

Menurut Robert dkk. (2024), sistem akuaponik sangat efisien dalam penggunaan air karena air terus berputar dari wadah ikan ke wadah tanaman. Hal ini dapat mengurangi kebutuhan sistem akan air secara signifikan jika dibandingkan dengan sistem perkebunan tradisional. Selain itu, air yang berasal dari kolam ikan juga membawa nutrisi untuk tanaman sehingga mengurangi kebutuhan pada pupuk tambahan. Adopsi sistem akuaponik juga memiliki pendekatan yang lebih aman dalam *pest management* sehingga mengurangi kebutuhan pestisida kimia yang berbahaya. Hal ini mendukung konsep pertanian yang efisien, aman, dan hemat sumber daya.



Gambar 2.1 Alur *Waterflow* dalam Sistem Akuaponik

2.3 Budidaya Pakcoy

Pakcoy (*Brassica chinensis L.*) merupakan salah satu jenis sayuran yang cukup populer di masyarakat. Tanaman ini dapat tumbuh secara optimal pada kondisi lingkungan yang stabil dan sesuai dengan kebutuhan pertumbuhannya. Beberapa faktor eksternal seperti suhu udara, kelembapan, serta intensitas cahaya menjadi faktor penting yang sangat mempengaruhi keberhasilan budidaya, terutama pada sistem hidroponik yang menggunakan air sebagai media tanam pengganti tanah (Antanita dkk., 2025).

Dalam sistem akuaponik, kotoran ikan menjadi sumber nutrisi yang dibutuhkan bagi tanaman. Melalui aktivitas mikroba, kotoran ikan dapat diubah menjadi nutrisi yang dibutuhkan bagi tanaman dan secara terus-menerus memperbaiki kualitas air bagi ikan (Hutagalung dkk., 2023). Tanaman pakcoy dapat tumbuh dengan baik pada rentang suhu antara 20°C hingga 27°C, dengan kelembapan udara pada berkisar antara 60% - 80%. Di samping itu, tanaman ini juga membutuhkan intensitas cahaya minimal sebesar 2500 lux untuk mendukung proses fotosintesis dan pertumbuhannya (Antanita dkk., 2025). Selain itu, menurut penelitian yang dilakukan oleh Putra dkk., (2024) menunjukkan bahwa durasi

fotoperiode yang optimal bagi pertumbuhan tanaman pakcoy adalah 20 jam terang dan 4 jam gelap, yang merupakan kombinasi dari 12 jam cahaya matahari alami dan 8 jam tambahan cahaya buatan. Oleh karena itu, pengendalian otomatis terhadap faktor-faktor iklim di dalam *greenhouse* sangat disarankan guna meminimalkan fluktuasi lingkungan yang tidak stabil, sehingga kualitas hasil panen dapat lebih terjamin dan konsisten (Antanita dkk., 2025).

2.4 *Internet of Things*

Internet of Things (IoT) adalah sebuah jaringan objek-objek fisik (*things*) tertanam dengan perangkat elektronik, perangkat lunak, sensor, dan konektivitas yang memungkinkan pengumpulan dan pertukaran data dengan perangkat lain dan sistem melalui internet (Bathla dan Kaur, 2023). Dalam penerapan IoT, objek yang dimaksud bisa mengarah pada hal-hal apapun seperti peralatan rumah tangga, alat-alat industri, bahkan makhluk hidup seperti hewan dan manusia. Hal ini menunjukkan bahwa IoT memiliki banyak manfaat dan membuka peluang untuk merealisasikan potensi penuhnya.

IoT memiliki komponen-komponen utama. Berikut ini adalah komponen-komponen utama dalam IoT.

1. Sensor

Sensor adalah sebuah alat yang digunakan untuk mengumpulkan data dari dunia fisik, mengukur kondisi *environment*, dan memonitor aktivitas seperti deteksi gerakan, dan *facial recognition*.

2. Aktuator

Aktuator adalah perangkat yang dapat melakukan aksi berdasarkan data yang diterima dari sensor, seperti penyesuaian suhu thermostat atau menyalakan lampu.

3. Protokol komunikasi

Perangkat IoT umumnya menggunakan beberapa protokol komunikasi untuk transmisi data. Beberapa diantaranya adalah MQTT, CoAP, dan WebSocket yang sesuai pada perangkat dengan *resources* terbatas.

4. UI (*User Interface*)

Pengguna berinteraksi dengan sistem IoT menggunakan beragam *interface* yang dapat berupa aplikasi pada *smartphone*, portal *website*, dan beragam *digital interface* lainnya (Dudeja dkk., 2022). Dengan adanya *interface* ini, pengguna dapat memantau, mengatur dan menerima notifikasi dari sistem IoT.

2.5 *Microcontroller* Dalam Lingkungan *Internet Of Things*

Dalam sistem otomasi berbasis *Internet of Things* (IoT), *microcontroller* bertindak sebagai unit pemrosesan pusat yang mengelola integrasi seluruh perangkat keras. Sebagaimana dijelaskan oleh Ramkumar dkk. (2023), *microcontroller* digunakan sebagai *data relay* yang berfungsi untuk mengakses, mengelola, serta mengendalikan operasional perangkat yang terhubung dalam lingkungan sistem. Dalam implementasi otomasi, *microcontroller* sangat penting untuk memastikan aliran data dari berbagai sensor dapat diproses secara akurat dan konsisten sebelum sistem memberikan respons melalui aktuator.

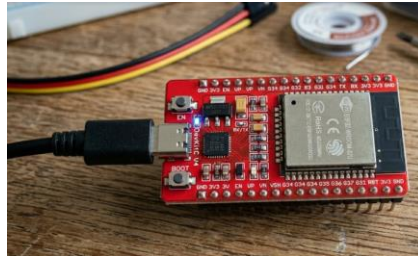
Untuk mendukung arsitektur sistem yang efisien, penggunaan ESP32 menjadi pilihan yang tepat. ESP32 merupakan mikrokontroler yang serbaguna dan ekonomis, yang banyak digunakan dalam berbagai aplikasi *Internet of Things* (IoT). Perangkat ini dilengkapi dengan konektivitas nirkabel, termasuk Wi-Fi dan Bluetooth, sehingga ideal untuk sistem IoT yang memerlukan komunikasi dan pertukaran data. Selain itu, ESP32 memiliki desain yang ringkas, efisien dalam penggunaan daya, serta mendukung berbagai bahasa pemrograman, salah satunya adalah C++ (Zulfiqar, 2024). ESP32 memungkinkan sistem untuk membaca variabel lingkungan secara *real-time* dan melakukan aksi secara otomatis. Menurut Ramkumar dkk. (2023), kapabilitas komunikasi ini memastikan bahwa perangkat dapat bekerja secara otonom dalam mengelola variabel sistem, sehingga meningkatkan reliabilitas dan efisiensi sistem.

2.6 Modul ESP32

ESP32 adalah *microcontroller* yang andal dan multifungsi, yang melayani berbagai macam aplikasi, terutama dalam bidang IoT dan sistem tertanam (*embedded systems*). Gabungan dari *Low Cost*, *Powerful Processing*, serta konektivitas nirkabel terintegrasi menjadikannya pilihan populer di kalangan pengembang dan peneliti (Fadelallah dkk., 2023).

ESP32 memiliki beberapa kelebihan yang membuatnya populer. Menurut (Fadelallah dkk. (2023), ESP32 dikenal memiliki harga yang relatif murah dengan kemampuan yang andal dalam berbagai aplikasi. Selain itu, perangkat ini juga mudah diprogram sehingga memudahkan pengembang dalam proses implementasi sistem. ESP32 mendukung komunikasi jarak jauh serta mampu melakukan akuisisi data dari berbagai sensor secara

bersamaan. Dari segi performa, ESP32 juga memiliki stabilitas yang tinggi dan mendukung sinkronisasi waktu dengan jam universal internet, sehingga cocok digunakan dalam berbagai sistem berbasis IoT. Gambar 2.2 adalah contoh bentuk fisik dari ESP32.



Gambar 2.2 Modul ESP32

2.7 Sensor Dalam Lingkungan *Internet Of Things*

Sensor adalah sebuah alat yang digunakan untuk mengukur kuantitas fisik seperti tekanan, temperatur, intensitas cahaya, gas, kelembapan, dan lain-lain sebagai input kemudian mengubahnya menjadi data yang dapat diukur oleh manusia atau mesin, keluarannya umumnya adalah data elektronik seperti tegangan atau arus listrik (Taghavi, 2024). Saat ini, sensor telah banyak digunakan dalam beragam bidang seperti otomatisasi industri, perangkat medis, hingga perlengkapan sehari-hari.

Menurut Rajora dkk. (2024), tren masa depan sensor IoT mengarah pada penggunaan *smart sensors* yang memiliki kemampuan pemrosesan awal di tingkat perangkat (edge computing) serta integrasi dengan teknologi AI dan konektivitas 5G. Hal ini memungkinkan sistem untuk melakukan analisis data yang lebih prediktif dan responsif. Hal ini memungkinkan sistem melakukan analisis data yang lebih prediktif dan responsif.

Dalam konteks pemantauan lingkungan seperti pada sistem akuaponik, sensor memainkan peran krusial dalam menjaga stabilitas ekosistem secara otomatis. Penelitian oleh Shukri dkk. (2024) menunjukkan bahwa penggunaan sensor kualitas air yang terintegrasi dengan microcontroller ESP32 memungkinkan pemantauan parameter kritis seperti pH, suhu, dan kadar amonia secara *real-time* melalui platform berbasis cloud. Penelitian ini melakukan analisis PCA (*principal component analysis*) untuk mengubah data mentah dari sensor menjadi informasi yang dapat diinterpretasikan untuk menentukan stabilitas sistem.

2.8 Sensor Suhu dan Kelembapan (DHT22)

DHT22 adalah sensor yang digunakan untuk mengukur suhu dan kelembapan udara di lingkungannya. Sensor ini dapat membaca data dari lingkungan dengan akurat sehingga menjadi hal yang krusial dalam mempertahankan kondisi lingkungan penanaman untuk tanaman (Zain dkk., 2024).

2.9 Sensor pH (Model 4502C)

Sensor pH 4502C merupakan elektroda pH kombinasi yang dirancang untuk mengukur tingkat keasaman (pH) dalam larutan berair. Dengan rentang pengukuran antara 0 hingga 14 pH, sensor ini mampu memberikan pengukuran yang komprehensif. Selain itu, sensor ini memiliki tingkat akurasi yang tinggi, dengan kemampuan menghasilkan pembacaan yang presisi hingga $\pm 0,01$ pH (Naputol dkk., 2024).

2.10 Definisi Aktuator

Aktuator dapat merubah sinyal listrik menjadi gerakan mekanik dan kontrol dalam sebuah sistem. aktuator telah banyak digunakan dalam bidang *aerospace*, otomotif, robotik, perangkat medis, dan manufaktur. Aktuator dapat diklasifikasikan ke dalam beberapa jenis diantaranya adalah *hydraulic*, *pneumatic*, *electric* ((G. Roy dkk., 2020). Aktuator-aktuator ini terus digunakan secara luas karena keandalan dan kinerjanya dalam berbagai aplikasi, termasuk aerospace, otomotif, dan otomasi industri. Selain itu, masih ada jenis aktuator lain.

2.11 Pompa Air

Dalam pengembangan sistem akuaponik, pompa air adalah salah satu komponen terpenting dalam menjaga lingkungan akuaponik. Penelitian yang dilakukan oleh (Shukri dkk., 2024) menunjukkan bahwa kondisi pompa air adalah sangat berpengaruh besar terhadap keseimbangan sistem sebagaimana yang ditunjukkan pada tabel 2.2. Dalam penelitiannya, disebutkan bahwa sirkulasi air berpengaruh pada kandungan ammonia dan suhu air.

Dari Tabel 2.2 tersebut dapat terlihat bahwa kondisi masalah yang terjadi pada pompa mempengaruhi paling banyak hal yang diukur dengan sensor. Keadaan pompa yang abnormal mempengaruhi suhu dan kadar NH_3 dalam sistem akuaponik yang mana 2 hal

tersebut tidak terjadi pada dua kondisi lainnya. Oleh karena itu, kebutuhan sistem akuaponik akan pompa menjadi sangat dibutuhkan.

Tabel 2.2 Hubungan Kondisi Akuaponik dengan Dampak pada Lingkungan Akuaponik (Shukri dkk., 2024)

Kondisi	Dampak		
	NH3	Suhu	pH
Boros makanan	-	-	v
Masalah Pompa Air	v	v	-
Masalah Filter	-	-	v

2.12 Pompa Aerator

Menurut S. M. Roy dkk. (2024) , aerator memegang peranan krusial sebagai penopang kehidupan dengan memastikan ketersediaan *Dissolved Oxygen* (DO) yang optimal, yang sering kali tidak dapat dipenuhi melalui proses difusi atmosferik alami dalam sistem *aquaculture*. Secara teoritis, aerator bekerja berdasarkan prinsip pemindahan massa gas-cair untuk meningkatkan konsentrasi oksigen dalam air, yang secara langsung berkorelasi dengan laju pertumbuhan, kesehatan, dan produktivitas organisme akuatik. Dengan mempertahankan parameter DO yang memadai melalui sistem pengudaraan, aerator berfungsi mencegah stres hipoksia pada biota budidaya dan memastikan keberlangsungan sistem produksi yang padat tebar.

2.13 Cahaya Buatan (*Artificial Light*)

Artificial light atau cahaya buatan merupakan komponen penting dalam sistem akuaponik, khususnya ketika digunakan dalam lingkungan *greenhouse*. Meskipun *greenhouse* berfungsi melindungi tanaman dari kondisi eksternal seperti cuaca ekstrem dan hama, intensitas cahaya alami yang masuk sering kali tidak stabil, terutama pada musim hujan. Padahal, cahaya merupakan faktor utama dalam proses fotosintesis yang sangat menentukan pertumbuhan, hasil, dan kualitas tanaman. Oleh karena itu, ketika intensitas cahaya alami tidak mencukupi, penggunaan cahaya buatan menjadi solusi alternatif untuk memastikan tanaman tetap mendapatkan pencahayaan yang optimal (Putra dkk., 2024).

Penggunaan lampu *Light Emitting Diode* (LED) dalam sistem akuaponik menjadi pilihan yang efektif karena memiliki efisiensi energi yang tinggi serta kemampuannya

menghasilkan spektrum cahaya tertentu. Cahaya merah dan biru diketahui paling efektif diserap oleh klorofil untuk mendukung proses fotosintesis. Cahaya merah berperan dalam meningkatkan pertumbuhan biomassa tanaman, sedangkan cahaya biru berpengaruh pada perkembangan vegetatif seperti daun dan batang. Kombinasi kedua jenis cahaya tersebut dengan pengaturan durasi penyinaran (*fotoperiode*) yang tepat terbukti mampu meningkatkan pertumbuhan tanaman secara optimal, sehingga penerapan *light artificial* dapat meningkatkan produktivitas dalam sistem akuaponik (Putra dkk., 2024).

2.14 Model Pengembangan Perangkat Lunak, (*Software Development Lifecycle*)

Pengembangan perangkat lunak adalah sebuah struktur yang digunakan dalam mengembangkan *software*. *Software development life cycle* (SDLC) memiliki beberapa model dalam pengembangannya untuk memastikan bahwa *software* sesuai dengan kebutuhan *customer* (Alazzawi dkk., 2023).

Beberapa metode yang digunakan dalam pengembangan *software* adalah sebagai berikut.

1. Model tradisional

Model tradisional adalah *plan-driven model*. Model-model yang berada dalam kelompok ini umumnya berjalan mengikuti proses-proses urut yang harus dikerjakan untuk memulai proses selanjutnya. Ciri-ciri dari model-model kelompok ini adalah terstruktur, cocok untuk proyek dengan *requirement* yang jelas dan jarang terjadi perubahan, dan pengecekan kualitas pada setiap fasenya. Model yang ada dalam kelompok model tradisional adalah sebagai berikut.

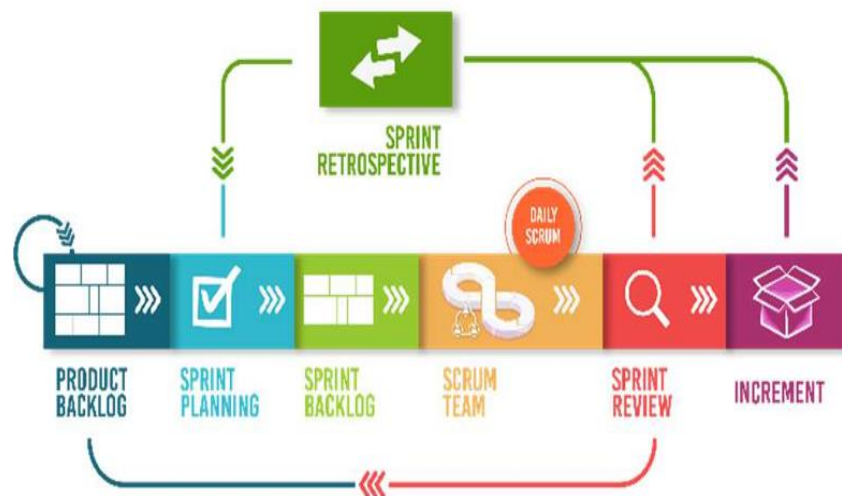
- a. *Waterfall Method* : Model pengembangan *software* yang mengharuskan sebuah fase diselesaikan sebelum fase lainnya dimulai.
- b. *Spiral Method* : Model ini menerapkan iterasi, berorientasi pada Risiko dan lebih fleksibel dibandingkan dengan *waterfall method*.
- c. *V-Method* : Bentuk modifikasi dari model *waterfall*. V-method mengaplikasikan verifikasi dan validasi pada setiap fase pengembangan.

2. Model Agile

Model *agile* menekankan pada fleksibilitas, kolaborasi antara developer dan *stakeholder*, dan proses yang iteratif. Model-model pengembangan yang ada dalam kelompok ini adalah Scrum, *Extreme Programming* (XP), kanban, dan Crystal.

2.15 Model Pengembangan Scrum

Scrum adalah kerangka kerja yang dirancang untuk menghasilkan nilai dalam pengembangan teknologi melalui solusi adaptif guna mengatasi permasalahan yang kompleks (Sastrohartono dkk., 2022). Metodologi ini sering dipilih dalam pengembangan sistem karena kemampuannya untuk mendukung proses bisnis yang rumit dan memerlukan penyesuaian berkelanjutan. Gambar 2.3 adalah alur metodologi Scrum menurut Penelitian Sastrohartono dkk. (2022).



Gambar 2.3 Tabel *Scrum Framework* (Sastrohartono dkk. , 2022)

Scrum memiliki serangkaian kegiatan (*events*) yang digunakan untuk mendukung proses inspeksi dan adaptasi selama pengembangan produk berlangsung. Seluruh kegiatan tersebut berada di dalam Sprint sebagai wadah utama pengembangan produk (Schwaber dan Sutherland, 2020).

1. *Sprint*

Sprint merupakan periode waktu tetap dengan durasi maksimal satu bulan yang digunakan untuk menghasilkan Increment yang bernilai. Seluruh aktivitas Scrum, seperti Sprint Planning, Daily Scrum, Sprint Review, dan Sprint Retrospective, dilaksanakan di dalam Sprint. Setelah satu Sprint selesai, Sprint berikutnya dimulai secara langsung.

2. *Sprint Planning*

Sprint Planning merupakan kegiatan awal yang dilakukan untuk merencanakan pekerjaan yang akan diselesaikan selama Sprint. Pada kegiatan ini, Scrum Team

menentukan tujuan Sprint (Sprint Goal), memilih item dari Product Backlog yang akan dikerjakan, serta menyusun rencana pelaksanaan pekerjaan tersebut.

3. *Daily Scrum*

Daily Scrum adalah pertemuan harian berdurasi maksimal 15 menit yang bertujuan untuk memantau kemajuan menuju Sprint Goal dan menyesuaikan Sprint Backlog apabila diperlukan. Kegiatan ini membantu meningkatkan komunikasi dan koordinasi antaranggota tim.

4. *Sprint Review*

Sprint Review dilakukan pada akhir Sprint untuk mengevaluasi hasil pekerjaan yang telah diselesaikan. Scrum Team bersama stakeholder meninjau Increment yang dihasilkan serta mendiskusikan kemungkinan perubahan yang perlu dilakukan pada Product Backlog berdasarkan kondisi terbaru.

5. *Sprint Retrospective*

Sprint Retrospective merupakan kegiatan refleksi yang dilakukan setelah Sprint Review. Pada tahap ini, Scrum Team mengevaluasi proses kerja yang telah dilakukan, mengidentifikasi kendala yang muncul, serta menentukan langkah-langkah perbaikan yang dapat diterapkan pada Sprint berikutnya untuk meningkatkan efektivitas tim.

Scrum memiliki tiga artefak utama, yaitu *Product Backlog*, *Sprint Backlog*, dan *Increment* (Schwaber dan Sutherland, 2020).

1. *Product Backlog* adalah daftar kebutuhan produk yang tersusun berdasarkan prioritas dan menjadi sumber utama pekerjaan bagi Scrum Team. Daftar ini terus berkembang sesuai dengan kebutuhan dan perubahan yang terjadi selama pengembangan produk.
2. *Sprint Backlog* merupakan kumpulan *Product Backlog Item* yang dipilih untuk dikerjakan dalam Sprint tertentu, beserta rencana pelaksanaannya. Sprint Backlog digunakan sebagai panduan kerja tim selama Sprint berlangsung.
3. *Increment* adalah hasil pekerjaan yang telah memenuhi Definition of Done dan dapat digunakan oleh pengguna. Setiap Sprint menghasilkan satu atau lebih Increment yang memberikan nilai tambah terhadap produk yang sedang dikembangkan.

Scrum Team merupakan unit dasar dalam Scrum yang terdiri atas tiga peran utama, yaitu *Product Owner*, *Scrum Master*, dan *Developers*. Tim Scrum bersifat *cross-functional* dan *self-managing*, yang berarti anggota tim memiliki seluruh keterampilan yang diperlukan

untuk menghasilkan produk serta mampu mengatur pekerjaannya sendiri tanpa bergantung pada pihak eksternal (Schwaber dan Sutherland, 2020).

1. *Product Owner* bertanggung jawab untuk memaksimalkan nilai produk yang dihasilkan oleh Scrum Team. Tugas utama *Product Owner* meliputi penyusunan *Product Goal*, pengelolaan *Product Backlog*, penentuan prioritas pekerjaan, serta memastikan seluruh *backlog* dapat dipahami oleh tim pengembang. *Product Owner* juga berperan sebagai perwakilan *stakeholder* dalam menentukan kebutuhan produk yang akan dikembangkan.
2. *Scrum Master* bertanggung jawab memastikan Scrum diterapkan sesuai dengan aturan dan prinsip yang telah ditetapkan dalam *Scrum Guide*. *Scrum Master* berperan sebagai fasilitator yang membantu tim memahami Scrum, menghilangkan hambatan yang mengganggu pekerjaan tim, serta meningkatkan efektivitas *Scrum Team*. Selain itu, *Scrum Master* juga membantu organisasi dalam mengadopsi dan menerapkan Scrum secara lebih efektif.
3. *Developers* merupakan anggota tim yang bertanggung jawab untuk menghasilkan Increment yang dapat digunakan pada setiap Sprint. *Developers* memiliki tanggung jawab dalam menyusun Sprint Backlog, menjaga kualitas pekerjaan sesuai Definition of Done, menyesuaikan rencana kerja berdasarkan perkembangan proyek, serta bekerja sama untuk mencapai Sprint Goal yang telah ditetapkan.

2.16 Unified Modeling Language (UML)

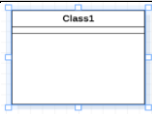
UML (*unified modeling language*) adalah sebuah bahasa modeling yang terstandarisasi dan digunakan secara luas dalam *software engineering* untuk melakukan visualisasi, spesifikasi, membangun, dan mendokumentasikan *artifacts* dalam sistem *software* (Apostol dkk., 2024). Diagram UML bermanfaat untuk pengembangan perangkat lunak yaitu untuk desain, analisis, dokumentasi, komunikasi (Apostol dkk., 2024), dan edukasi (Metzner, 2024) Oleh karena itu, penggunaan UML adalah hal yang penting dalam pengembangan perangkat lunak.

Terdapat beragam jenis UML yang digunakan dalam pengembangan perangkat lunak. Beberapa UML yang umum digunakan adalah sebagai berikut.

1. *Class diagram*

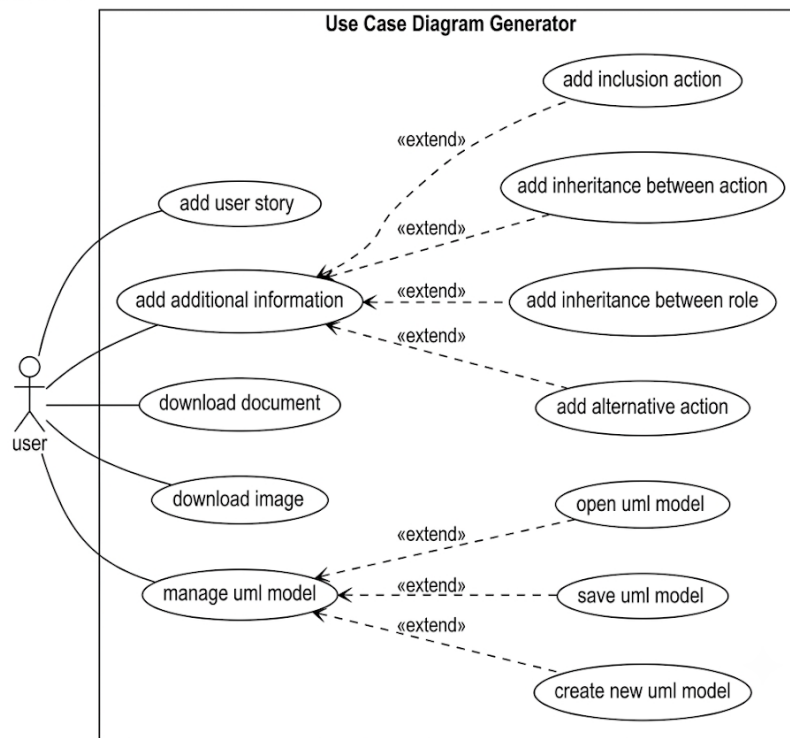
Menurut (Apostol dkk., 2024), *Class diagram* adalah hal fundamental dalam sistem *software* berorientasi objek, memodelkan struktur statis dengan mendefinisikan *class*, atributnya, operasinya, dan hubungannya dengan struktur lainnya (*relationship*). Tabel 2.3 menunjukkan komponen-komponen dalam *class diagram* menurut (Ramdany dkk., 2024).

Tabel 2.3 Komponen *Class Diagram* (Ramdany dkk. , 2024)

Simbol	Nama	Deskripsi
	<i>Class</i>	Blok pembangunan utama dalam pemrograman berorientasi objek yang berisi nama class, atribut, dan metode.
	<i>Association</i>	Hubungan yang menunjukkan adanya interaksi antar class.
	<i>Aggregation</i>	Hubungan keseluruhan-bagian (<i>whole-part relationship</i>) antara dua class.
	<i>Composition</i>	Hubungan ketergantungan kuat dimana suatu class tidak dapat berdiri sendiri tanpa class induknya.
+, -	<i>Visibility</i>	Tingkat akses terhadap atribut atau fungsi suatu class.
0..*, 1	<i>Multiplicity/ Cardinality</i>	Menunjukkan jumlah objek yang dapat berpartisipasi dalam suatu relasi antar class.

2. *Use case diagram*

Use case diagram memodelkan kebutuhan fungsional sistem dengan membuat ilustrasi interaksi antara pengguna (aktor) dan sistem untuk mencapai tujuan tertentu (Fathiyah dan Widyani, 2024). Gambar 2.4 adalah contoh *use case diagram*.



Gambar 2.4 *Use Case Diagram* (Fathiyah dan Widyani, 2024)

Tabel 2.4 menunjukkan komponen-komponen yang digunakan dalam pembentukan *use case diagram* menurut (Ramdany dkk., 2024).

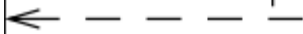
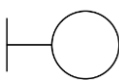
Tabel 2.4 Komponen *Use Case Diagram* (Ramdany dkk. , 2024)

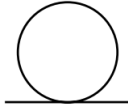
Simbol	Nama	Deskripsi
	<i>Actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem dari luar sistem. Aktor merepresentasikan peran, bukan individu tertentu.
	<i>Use Case</i>	Fungsionalitas yang disediakan sistem dan digunakan oleh aktor untuk mencapai tujuan tertentu. Biasanya dinyatakan dengan kata kerja.
	<i>Association</i>	Hubungan komunikasi antara aktor dan use case yang menunjukkan adanya interaksi.
	<i>Extend</i>	Relasi yang menunjukkan bahwa suatu use case dapat memperluas fungsi use case lain pada kondisi tertentu. Use case tambahan dapat berdiri sendiri.
	<i>Include (Uses)</i>	Relasi yang menunjukkan bahwa suatu use case memerlukan use case lain untuk menjalankan fungsinya.

3. Sequence diagram

Menurut Rumbaugh dkk. (2000), *Sequence diagram* adalah diagram dalam UML yang memodelkan interaksi antar objek dalam sebuah sistem dengan menekankan pada urutan kronologis pengiriman pesan, di mana dimensi vertikal merepresentasikan alur waktu dan dimensi horizontal menunjukkan partisipan atau objek yang terlibat. Dengan memvisualisasikan bagaimana objek-objek saling berkomunikasi melalui pesan baik sinkron maupun asinkron diagram ini membantu pengembang memahami aliran kendali (*flow of control*) dan urutan perilaku sistem untuk menjalankan skenario atau *use case* tertentu. Tabel 2.5 ini adalah komponen-komponen yang digunakan dalam membangun *sequence diagram* menurut Rumbaugh dkk (2000).

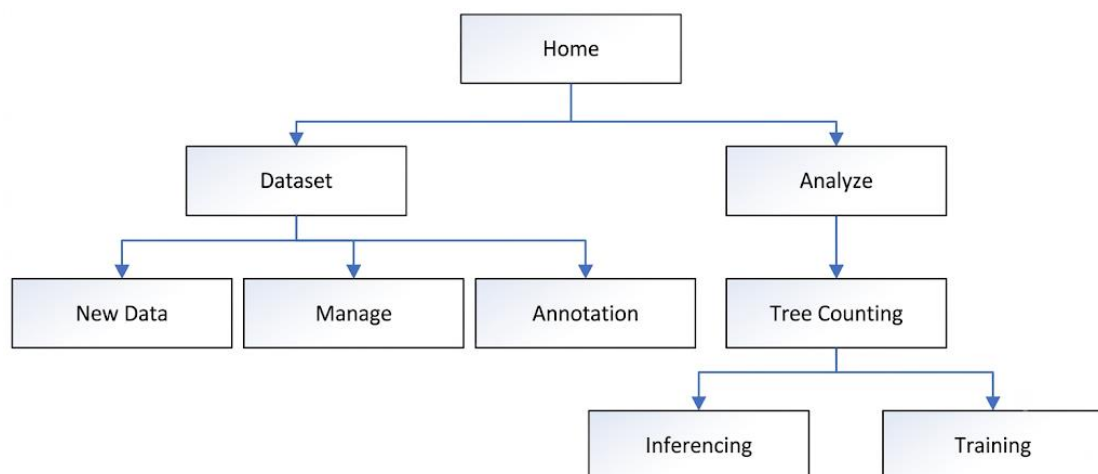
Tabel 2.5 Komponen Sequence Diagram (Rumbaugh dkk, (2000))

Komponen	Deskripsi	Simbol
<i>Lifeline</i>	Merepresentasikan partisipan (objek atau aktor) dalam interaksi dan durasi keberadaannya.	
Aktor	Entitas eksternal yang memulai atau berinteraksi dengan sistem.	
<i>Synchronous Message</i>	Pesan di mana pengirim menunggu respon (balasan) dari penerima sebelum melanjutkan proses.	
<i>Return Message</i>	Pesan balasan dari penerima pesan kepada pengirim setelah proses selesai.	
<i>Self-Message</i>	Pesan yang dikirimkan oleh objek kepada dirinya sendiri.	
<i>Active Lifeline</i>	Menunjukkan periode waktu di mana sebuah objek aktif melakukan operasi/proses.	
<i>Control Class</i>	Menggambarkan objek yang melakukan kontrol terhadap interaksi	
<i>Boundary Class</i>	Menggambarkan objek yang berinteraksi dengan aktor eksternal	

Komponen	Deskripsi	Simbol
<i>Entity Class</i>	Menggambarkan objek pasif yang tidak menginisiasi interaksi. <i>Class</i> ini juga merepresentasikan data utama dalam sistem	

4. *Navigation Diagram*

Dalam pengembangan sistem, diagram navigasi merupakan komponen penting yang digunakan untuk menggambarkan skema keterhubungan antar layar pada *user interface* dari sistem yang akan dikembangkan. Pembuatan diagram navigasi ini dirancang dengan tujuan untuk memberikan kemudahan bagi pengguna dalam menemukan berbagai fungsi yang diperlukan guna mendukung aktivitas mereka saat mengoperasikan sistem (Sastrohartono dkk., 2022).



Gambar 2.5 *Navigation Diagram* (Sastrohartono dkk. , 2022)

2.17 ESP-NOW

ESP-NOW merupakan protokol komunikasi nirkabel *peer-to-peer* yang dikembangkan oleh Espressif Systems, dirancang untuk mendukung kebutuhan perangkat *Internet of Things* (IoT) yang memerlukan efisiensi daya tinggi dan biaya operasional rendah. Protokol ini memfasilitasi komunikasi dua arah baik secara unicast maupun broadcast tanpa memerlukan koneksi Wi-Fi yang kompleks, sehingga sangat fleksibel untuk aplikasi *in-situ* (Magzym dkk., 2023).

2.18 *Model-View-Controller*

Menurut penelitian yang dilakukan oleh Necula (2024) *Model-View-Controller* (MVC) merupakan pola desain yang membagi logika aplikasi menjadi tiga komponen saling terhubung, yaitu *Model*, *View*, dan *Controller* guna menerapkan prinsip *separation of concerns*. Signifikansi arsitektur ini dalam meningkatkan pengelolaan aplikasi (*manageability*), penggunaan kembali kode, dan pendorongan pengembangan paralel terbukti secara ilmiah melalui metodologi *Systematic Literature Review* (SLR) terhadap 123 dokumen riset yang diekstraksi dari berbagai basis data akademik terkemuka. Berdasarkan analisis, pentingnya MVC sebagai praktik yang baik dalam rekayasa perangkat lunak dibuktikan oleh tingginya sentralitas kata kunci "*design patterns*" dan "*software engineering*", yang menegaskan bahwa arsitektur ini menjadi fokus utama dalam diskusi standarisasi desain sistem yang kokoh dan efisien. Kekuatan dan fleksibilitas arsitektur ini dibuktikan oleh luasnya spektrum implementasi lintas bahasa pemrograman dan sektor pasar yang terekam dalam data penelitian.

2.19 *Framework Laravel*

Laravel adalah salah satu *framework* PHP yang *open-source* yang paling populer dan banyak digunakan untuk membangun aplikasi web modern yang kompleks. Menurut Rahmouni dkk. (2023), dunia pengembangan web telah bergeser dari sekadar situs web statis menjadi aplikasi web yang dinamis. Dalam hal ini, Laravel menyediakan arsitektur yang kuat berbasis *Model-View-Controller* (MVC) yang memisahkan logika bisnis, data, dan antarmuka pengguna. Laravel menyediakan aturan pengembangan yang ketat untuk mempercepat proses pembuatan kode program yang bersih, aman, dan mudah dipelihara. Fitur-fitur unggulan seperti *Object-Relational Mapping* (ORM) Eloquent, sistem migrasi basis data, dan pengelolaan routing yang fleksibel membuat Laravel menjadi standar tinggi dalam pengembangan aplikasi web tingkat perusahaan (*enterprise*) maupun sistem manajemen sumber daya (Rahmouni dkk., 2023).

Meskipun awalnya dirancang khusus untuk aplikasi web konvensional, arsitektur Laravel kini telah dieksplorasi lebih luas untuk mendukung ekosistem *Internet of Things* (IoT). Berdasarkan penelitian komparatif yang dilakukan oleh Popa dan Buciu (2025), pemanfaatan Laravel sebagai platform pemantauan (*monitoring*) dan pengendalian

perangkat IoT memiliki signifikansi besar karena memungkinkan para pengembang *web* menggunakan *tools* yang sudah familiar tanpa harus mempelajari lingkungan pemrograman baru. Namun, tantangan utama dari integrasi ini terletak pada sifat protokol HTTP bawaan Laravel yang bersifat *stateless*, sehingga menyulitkan pembentukan koneksi kontinu dengan perangkat keras IoT. Riset tersebut menguji empat metode komunikasi utama: REST API HTTP, *Database Polling*, WebSockets (menggunakan Laravel Echo dan Laravel Reverb), serta protokol MQTT (Message Queuing Telemetry Transport) melalui bantuan broker Mosquitto. Hasil pengujian menunjukkan bahwa metode REST API dan *Database Polling* sangat ideal untuk pembuatan prototipe cepat karena kemudahan implementasinya, sedangkan WebSockets dan MQTT terbukti unggul sebagai standar industri karena mampu menyediakan komunikasi dua arah secara real-time dengan efisiensi dan skalabilitas tinggi untuk menangani ribuan perangkat IoT.

2.20 MySQL

Menurut Noviana (2022), MySQL merupakan *Database Management System* (DBMS) *open-source* yang populer karena sifatnya yang gratis serta memiliki kapabilitas *multiuser* dan *multithreaded*. Noviana (2020) juga menjelaskan bahwa SQL berfungsi sebagai bahasa kueri standar dalam konsep *Relational Database Management System* (RDBMS) yang digunakan untuk menjalankan berbagai tugas, mulai dari pembuatan struktur hingga manipulasi dan pembaruan data.

2.21 Black Box Testing

Menurut (Sasmito dan Mutasodirin, 2023), *black box testing* merupakan metode pengujian perangkat lunak yang dilakukan dengan mengamati fungsi sistem dari sisi pengguna tanpa memperhatikan struktur internal maupun kode program yang digunakan. Pengujian ini berfokus pada validasi antarmuka pengguna dan fungsionalitas setiap halaman aplikasi untuk memastikan bahwa seluruh fitur dapat berjalan sesuai dengan kebutuhan yang telah ditentukan. Dengan demikian, pengujian dilakukan berdasarkan masukan dan keluaran sistem, sehingga dapat diketahui apakah setiap fungsi dan form pada aplikasi telah memberikan hasil yang sesuai dengan yang diharapkan.