BAB II

LANDASAN TEORI

A. Kajian Teoritis

1. Tanaman Padi

Tanaman padi (Oryza sativa L.) merupakan komoditas pangan paling strategis di dunia, khususnya di Indonesia, di mana padi menempati posisi vital sebagai sumber utama pangan bagi sebagian besar penduduk dan berimplikasi langsung terhadap ketahanan pangan nasional Dwisusilo & Yuadi (2026). Sebagai tanaman serealia dari famili Poaceae, padi menyediakan sumber energi bagi lebih dari separuh populasi global. Namun, dalam proses budidayanya, produktivitas padi sangat bergantung pada kesehatan tanaman selama siklus hidupnya, yang meliputi fase vegetatif, generatif, hingga fase pemasakan.

Ketidakstabilan kondisi lingkungan dan serangan patogen seringkali menjadi hambatan utama dalam mencapai potensi hasil maksimal. Penyakit tanaman padi, seperti blas (blast), bercak coklat, dan tungro, merupakan ancaman biotik yang dapat menurunkan produktivitas padi secara signifikan dan menimbulkan kerugian besar apabila tidak ditangani dengan tepat (Sandy et al., 2024). Klasifikasi penyakit tanaman padi berdasarkan citra visual daun, seperti bercak (Brown Spot), hawar, dan perubahan warna daun, dapat dilakukan secara otomatis menggunakan model Convolutional Neural Network (Habib Hawari et al., 2022)

Mengingat kompleksitas identifikasi gejala secara manual yang seringkali subjektif dan lambat, integrasi teknologi digital menjadi krusial. Tren terkini menunjukkan bahwa penggunaan kecerdasan buatan, khususnya melalui model pembelajaran mendalam (deep learning), telah menjadi standar baru dalam memantau kesehatan tanaman secara presisi dan efisien (Kondaveeti & Simhadri, 2025). Dengan demikian, pemahaman terhadap karakteristik tanaman padi dan ancaman penyakitnya tidak hanya penting secara agronomis, tetapi juga menjadi dasar dalam perancangan sistem deteksi otomatis yang adaptif terhadap tantangan pertanian modern.

2. Penyakit Daun Padi

Penyakit tanaman padi merupakan gangguan fisiologis yang disebabkan oleh interaksi kompleks antara tanaman inang, faktor lingkungan, dan serangan patogen seperti cendawan, bakteri, virus, maupun nematoda. Infeksi ini mengakibatkan penyimpangan fungsi normal tanaman yang berujung pada penurunan kualitas serta kuantitas hasil panen secara signifikan. Di Indonesia, fluktuasi produksi padi turut dipengaruhi oleh kondisi iklim atau cuaca, sistem pengairan, serta intensitas serangan hama dan penyakit yang tidak menentu (Rustanto et al., 2024).

Secara visual, penyakit tanaman padi memicu gejala-gejala spesifik yang dapat diamati pada berbagai organ tanaman, terutama pada helai daun, batang, dan malai. Patogen seperti Magnaporthe oryzae (penyebab penyakit blas) dan Xanthomonas oryzae (penyebab hawar daun) meninggalkan pola lesi atau perubahan warna yang unik, sehingga klasifikasi berdasarkan

informasi warna dan tekstur daun menjadi kunci utama dalam membedakan jenis infeksi agar langkah penanganan yang diambil tepat sasaran (Rahmadani et al., 2023). Kerusakan yang tidak terdeteksi sejak dini pada area daun dapat menghambat proses fotosintesis, yang pada akhirnya menurunkan berat gabah per malai.

Keterlambatan dalam mendeteksi gejala penyakit tidak hanya berdampak pada kegagalan panen, tetapi juga memicu penggunaan pestisida kimia secara berlebihan oleh petani yang justru merusak ekosistem persawahan. Oleh karena itu, diperlukan pendekatan deteksi penyakit padi yang cepat, otomatis, dan andal, di mana pemanfaatan model deep learning ringan terbukti mampu mengenali karakteristik visual penyakit secara presisi dibandingkan model konvensional, sehingga memungkinkan pengambilan keputusan mitigasi yang lebih cerdas dan berkelanjutan di sektor pertanian digital (Rathore et al., 2023).

3. Penyakit Blas (Magnaporthe oryzae)

Penyakit blas, yang disebabkan oleh jamur Magnaporthe oryzae, merupakan kendala biotik destruktif di berbagai ekosistem lahan padi Indonesia. Penyebaran dan keparahan penyakit ini sangat dipengaruhi oleh faktor iklim makro dan mikro, termasuk musim, kelembapan, dan suhu, di mana kondisi optimal pertumbuhan patogen berada pada kisaran suhu 24 – 28°C dengan kelembapan udara mencapai 90% (Kurrata et al., 2021).

Secara morfologis, infeksi ini ditandai dengan gejala awal berupa bercak kecil menyerupai ujung jarum berwarna cokelat, yang kemudian

berkembang menjadi lesi berbentuk belah ketupat (diamond-shaped) dengan pusat abu-abu keputihan dan tepi berwarna cokelat. Jika menyerang bagian leher malai dan tidak segera dimitigasi, penyakit ini dapat memicu kehilangan hasil panen hingga 90% (Salimah et al., 2021).

Mengingat sifat penyebarannya yang dinamis melalui udara, implementasi sistem deteksi dini berbasis citra digital menjadi langkah krusial dalam mendukung upaya pengendalian penyakit secara presisi dan efisien, karena deteksi lesi yang akurat dan tepat waktu memungkinkan petani maupun peneliti melakukan intervensi cepat serta meminimalkan penyebaran infeksi (Uysal et al., 2023).

4. Hawar Daun Bakteri (Bacterial Leaf Blight)

Hawar daun bakteri (HDB) merupakan penyakit vaskular sistemik yang disebabkan oleh bakteri Xanthomonas oryzae pv. oryzae (Xoo). Menurut Laraswati et al. (2021), HDB merupakan salah satu penyakit utama pada tanaman padi dengan potensi kehilangan hasil panen mencapai 15–80%, di mana penyakit ini umumnya berkembang pesat pada musim penghujan dan menginfeksi tanaman melalui luka daun atau lubang alami berupa stomata.

Af'idzatuttama et al. (2025) menjelaskan bahwa gejala khas HDB diawali dengan bercak keabu-abuan pada ujung daun yang kemudian menyebar ke arah pangkal daun dan meluas hingga membentuk hawar (blight), di mana pada varietas yang rentan, gejala ini dapat berkembang hingga seluruh helai daun mengering.

Marsevin et al. (2025) menunjukkan bahwa model klasifikasi penyakit tanaman berbasis transfer learning terbukti efektif dan berpotensi diterapkan pada sistem deteksi otomatis berbasis citra untuk mendukung pertanian presisi secara real-time, sehingga pendekatan serupa relevan diadaptasi dalam mendeteksi gejala HDB secara otomatis guna mencegah penyebaran patogen ke area lahan yang lebih luas.

5. Bercak Coklat (Brown Spot)

Bercak coklat merupakan penyakit tanaman padi yang disebabkan oleh infeksi jamur Bipolaris oryzae (sinonim Helminthosporium oryzae). Menurut Norjamilah et al. (2021), penyakit ini dapat menyebabkan kehilangan hasil produksi mencapai 50–91%, dengan gejala awal berupa bintik-bintik kecil yang muncul pada daun.

Prasetyani (2023) menemukan bahwa penerapan teknik budidaya seperti pemupukan yang berimbang dan pengaturan jarak tanam yang optimal dapat meningkatkan daya tahan tanaman terhadap serangan bercak coklat, mengindikasikan bahwa pengelolaan pemupukan yang tidak seimbang menjadi salah satu faktor pendorong berkembangnya penyakit ini.

Sofyaningrum et al. (2024) pada penyakit bercak daun sejenis menemukan bahwa penambahan bahan organik berpengaruh nyata dalam menekan intensitas penyakit, sedangkan sistem pengairan tidak menunjukkan pengaruh signifikan, sehingga pengelolaan bahan organik menjadi faktor budidaya yang perlu diperhatikan dalam upaya pengendalian bercak coklat pada padi.

6. Tungro

Penyakit tungro merupakan penyakit virus pada tanaman padi yang disebabkan oleh infeksi ganda dua jenis virus, yaitu Rice Tungro Bacilliform Virus (RTBV) dan Rice Tungro Spherical Virus (RTSV), yang ditularkan melalui vektor wereng hijau (Nephotettix virescens). Menurut Ruimassa et al. (2022), tungro tergolong salah satu penyakit penting pada tanaman padi karena menimbulkan gejala tanaman kerdil dan dapat menyebabkan kehilangan hasil yang nyata, sehingga penanaman varietas tahan menjadi strategi pengendalian yang direkomendasikan.

Gunawan et al. (2024) menjelaskan bahwa gejala khas tungro ditandai dengan perubahan warna daun muda menjadi kuning hingga jingga yang dimulai dari ujung daun, disertai berkurangnya jumlah anakan dan terhambatnya pertumbuhan tanaman (kerdil). Penelitian yang sama juga menunjukkan bahwa preferensi wereng hijau sebagai vektor penyebaran turut dipengaruhi oleh ketahanan varietas padi, sehingga pengendalian tungro memerlukan pendekatan terpadu antara pemantauan populasi vektor dan pemilihan varietas tahan.

Mengingat gejala visual tungro yang cukup khas dan kontras dibandingkan penyakit lain, penerapan metode Convolutional Neural Network (CNN) berbasis citra daun terbukti mampu mengklasifikasikan penyakit tungro bersama blas dan bercak coklat dengan akurasi validasi mencapai 100%, sehingga berpotensi besar sebagai alat bantu diagnosis dini penyakit tanaman padi (Hia et al., 2025).

7. Deep Learning

Deep learning merupakan sub-bidang dari machine learning yang berbasis pada arsitektur Jaringan Saraf Tiruan (Artificial Neural Networks) dengan banyak lapisan tersembunyi (deep neural networks). Sirmorya, Sirmorya et al. (2022) menjelaskan bahwa deep learning bersifat robust terhadap perubahan data masukan karena mampu mempelajari fitur secara otomatis tanpa perlu merancang fitur secara presisi terlebih dahulu, berbeda dengan metode konvensional yang memerlukan ekstraksi fitur secara manual.

Noor & Ige (2024) menambahkan bahwa deep learning menggunakan beberapa lapisan unit yang saling terhubung untuk mempelajari pola dan representasi kompleks secara langsung dari data mentah, menjadikannya alat yang ampuh untuk memecahkan berbagai permasalahan kompleks di berbagai domain.

Ning et al. (2023) menunjukkan bahwa arsitektur Convolutional Neural Network (CNN) sangat fleksibel dan efektif untuk berbagai tugas tingkat tinggi seperti klasifikasi citra, deteksi objek, dan segmentasi semantik, dengan performa yang jauh melampaui metode visi komputer tradisional. Penelitian yang sama juga menekankan bahwa perkembangan model CNN kini mengarah pada arsitektur yang lebih ringan (lightweight), sehingga memungkinkan integrasi langsung pada perangkat mobile untuk deteksi penyakit tanaman secara real-time di lapangan.

8. CNN (Convolutional Neural Network)

Convolutional Neural Network (CNN) merupakan salah satu arsitektur deep learning yang paling representatif dan telah mencapai berbagai pencapaian signifikan, terutama pada bidang computer vision. Li et al. (2022) menjelaskan bahwa CNN merupakan jaringan saraf tiruan feedforward yang mampu mengekstraksi fitur dari data secara otomatis melalui struktur konvolusi tanpa memerlukan ekstraksi fitur manual seperti pada metode konvensional, dengan tiga karakteristik utama yaitu koneksi lokal (local connections), berbagi bobot (weight sharing), dan reduksi dimensi melalui down-sampling.

Secara arsitektural, Harahap et al. (2022) menjabarkan bahwa CNN umumnya tersusun atas beberapa lapisan berurutan, yaitu input layer, convolutional layer, fungsi aktivasi ReLU, pooling layer, flatten layer, hingga fully connected layer, di mana masing-masing lapisan memiliki peran spesifik: convolutional layer bertugas mengekstraksi fitur melalui operasi konvolusi, sedangkan pooling layer mereduksi dimensi data untuk menekan kompleksitas komputasi sebelum diproses lebih lanjut pada tahap klasifikasi.

Raj & Kos (2025) menambahkan bahwa convolutional layer tersusun atas kernel-kernel yang dapat dipelajari (learnable kernels) dan bergerak menyusuri lebar serta tinggi citra untuk menghasilkan activation map, di mana lapisan-lapisan awal menangkap fitur tingkat rendah seperti tepi dan tekstur, sedangkan lapisan yang lebih dalam menangkap fitur yang lebih

abstrak, sementara itu, fully connected layer memproses hasil flatten dari pooling layer melalui kombinasi bobot dan fungsi aktivasi untuk menghasilkan keputusan klasifikasi akhir. Dengan kemampuan ekstraksi fitur otomatis dan struktur berlapis tersebut, CNN menjadi fondasi utama bagi berbagai pengembangan arsitektur deep learning modern, termasuk model-model ringan yang diadaptasi untuk deteksi penyakit tanaman berbasis citra.

9. Android

Android merupakan sistem operasi berbasis Linux yang dikembangkan oleh Google khusus untuk perangkat bergerak, seperti smartphone dan tablet, dan telah menjadi platform yang dominan dalam pengembangan aplikasi mobile berkat keterbukaan sistem serta beragamnya fitur yang ditawarkan. Diantoni et al. (2024) menjelaskan bahwa dalam pengembangan aplikasi Android, pemilihan pola arsitektur perangkat lunak menjadi aspek penting, di mana Model-View-ViewModel (MVVM) diterapkan untuk memisahkan aturan bisnis inti, logika aplikasi, dan detail teknis implementasi, sementara framework Jetpack Compose memungkinkan pengembang mendefinisikan antarmuka pengguna menggunakan bahasa pemrograman Kotlin secara dinamis.

Dalam konteks penerapannya pada sektor pertanian, Majid et al. (2023) merancang aplikasi media penyuluhan pertanian berbasis Android menggunakan metode pengembangan perangkat lunak waterfall, yang terdiri atas tahapan requirements analysis, system design, implementation,

verification, dan maintenance, sebagai upaya mendukung digitalisasi layanan penyuluhan bagi petani.

Alhafiz & Sela (2025) mengembangkan aplikasi mobile konsultasi pertanian berbasis Android dengan pendekatan pengembangan agile, yang menyediakan layanan konsultasi teknik pertanian, pemilihan pupuk, pengendalian hama, hingga pemantauan cuaca secara real-time; hasil pengujian menunjukkan bahwa aplikasi tersebut mampu meningkatkan produktivitas pertanian hingga 25%, sehingga menegaskan potensi besar platform Android sebagai media penyampaian solusi teknologi digital di sektor pertanian, termasuk untuk sistem deteksi penyakit tanaman berbasis citra yang dikembangkan dalam penelitian ini.

10. Laravel

Laravel merupakan framework PHP open-source yang dibangun oleh Taylor Otwell pada tahun 2011 dan menganut prinsip pengkodean yang elegan serta efisien, menjadikannya pilihan populer dalam pengembangan aplikasi web modern. Alma Christie C. Reyna (2023) menjelaskan bahwa komponen inti Laravel, seperti Eloquent ORM (Object-Relational Mapping) dan Blade templating engine, memungkinkan pengembang berinteraksi dengan basis data menggunakan pendekatan berorientasi objek yang ekspresif, di mana Eloquent menerapkan Active Record Pattern sehingga setiap tabel basis data direpresentasikan sebagai kelas dan setiap baris data sebagai objek, sejalan dengan penerapan arsitektur Model-View-Controller (MVC) pada Laravel.

Akhdani & Wijayanto (2022) membandingkan performa Eloquent ORM dengan Query Builder pada Laravel dalam studi kasus sistem manajemen kerja, dan menemukan bahwa Query Builder mampu mengeksekusi kueri basis data lebih cepat dan lebih efisien dari segi penggunaan memori dibandingkan Eloquent, meskipun Eloquent menawarkan sintaksis kode yang lebih sederhana bagi pengembang. Dari sisi keamanan, Lubis & Zulherry (2026) menunjukkan bahwa penerapan strategi pertahanan berlapis (defense-in-depth) menggunakan algoritma HMAC-SHA256 sebagai lapisan tambahan mampu memperkuat proteksi bawaan Laravel terhadap serangan Cross-Site Request Forgery (CSRF), terbukti efektif menahan serangan bahkan ketika proteksi bawaan dinonaktifkan.

Sementara itu, Aldino Dewa Ndaru et al. (2025) menerapkan Laravel untuk membangun REST API dengan autentikasi Sanctum guna mendukung pertukaran data yang aman dan efisien antar sistem, di mana API yang dihasilkan mendukung format JSON serta fitur pagination, sehingga menegaskan fleksibilitas Laravel sebagai framework backend yang andal untuk pengembangan sistem berbasis web maupun mobile.

11. MobileNetV2

MobileNetV2 merupakan salah satu arsitektur Convolutional Neural Network (CNN) ringan (lightweight) yang dirancang khusus untuk lingkungan dengan keterbatasan sumber daya komputasi, seperti perangkat mobile dan embedded. Gulzar (2023) menjelaskan bahwa MobileNetV2

pada dasarnya dikembangkan untuk mendukung perangkat seluler dan lingkungan dengan sumber daya terbatas, sehingga model ini banyak diadopsi sebagai arsitektur dasar (base model) pada berbagai penelitian klasifikasi citra berbasis transfer learning karena efisiensi komputasinya yang tinggi.

Secara arsitektural, Yong et al. (2023) menjelaskan bahwa MobileNetV2 dikembangkan berdasarkan MobileNetV1 dengan penyesuaian struktur melalui penerapan Inverted Residuals dan Linear Bottlenecks, di mana struktur inverted residual ini bekerja berkebalikan dari struktur residual konvensional, dimulai dari convolutional upscaling, dilanjutkan depthwise convolution, dan diakhiri dengan convolutional downscaling; penerapan depthwise separable convolution pada inti arsitektur ini secara signifikan mampu mengurangi jumlah komputasi dan parameter model tanpa mengorbankan akurasi secara drastis.

Husnurrizqi et al. (2026) menambahkan bahwa penerapan MobileNetV2 dengan pendekatan transfer learning, di mana model yang telah dipra-latih (pre-trained) digunakan sebagai feature extractor kemudian dilakukan fine-tuning pada lapisan klasifikasi, terbukti mampu menghasilkan kinerja klasifikasi yang andal dan stabil sekaligus tetap efisien secara komputasi, sehingga pendekatan ini relevan diadaptasi untuk berbagai kasus klasifikasi citra dengan keterbatasan data dan sumber daya, termasuk deteksi penyakit tanaman berbasis citra daun.

12. Flask Pyhton

Flask merupakan web framework yang ditulis menggunakan bahasa pemrograman Python dan tergolong sebagai microframework karena tidak memerlukan library tertentu secara wajib dalam penggunaannya. Novindri & Saian (2022) menjelaskan bahwa Flask memanfaatkan dua pustaka eksternal utama, yaitu WSGI toolkit dan Jinja2 template engine, serta dapat dikembangkan lebih lanjut menggunakan berbagai ekstensi sesuai kebutuhan sistem yang dibangun.

Rendi Yusuf Azhari (2022) membandingkan Flask dengan FastAPI sebagai framework web service, dan menjelaskan bahwa meskipun keduanya memiliki struktur program yang serupa, Flask dirancang untuk kebutuhan pengembangan web secara general purpose, sedangkan FastAPI dirancang khusus sebagai web service, sehingga pemilihan framework perlu disesuaikan dengan tujuan interoperabilitas sistem yang ingin dicapai.

Dalam penerapannya, Ratnasari et al. (2024) mengimplementasikan Flask sebagai web server yang dikombinasikan dengan MySQL sebagai basis data dan protokol TCP/IP melalui HTTP untuk membangun sistem client-server pengelolaan data, namun menemukan bahwa sistem berbasis Flask tersebut memiliki keterbatasan skala, di mana performa dapat menurun seiring meningkatnya jumlah pengguna atau volume data tanpa konfigurasi atau migrasi tambahan ke platform yang lebih kuat.

13. Mysql

MySQL merupakan salah satu sistem manajemen basis data relasional (Relational Database Management System/RDBMS) open-source yang banyak digunakan dalam pengembangan sistem informasi berbasis web karena kehandalan, kecepatan, dan kemudahan penggunaannya. Ramadhani et al. (2026) menjelaskan bahwa efektivitas integrasi data pada basis data MySQL sangat bergantung pada ketepatan perancangan Entity Relationship Diagram (ERD) serta proses normalisasi hingga tingkat 3NF untuk menjaga integritas data, di mana pengujian berdasarkan standar ISO 25010 membuktikan bahwa sistem berbasis MySQL memiliki reliabilitas tinggi dengan waktu respons yang stabil.

Dalam upaya meningkatkan performa pencarian data, Thoib et al. (2024) menunjukkan bahwa penerapan full-text index pada basis data MySQL secara signifikan mempercepat proses pencarian data berbasis teks dibandingkan tabel tanpa full-text index, khususnya pada dataset berskala besar, sebagaimana dibuktikan melalui pengujian statistik dengan nilai $p<0,05$.

Selain itu, Mokodaser et al. (2022) menerapkan metode Btree Indexing yang dikombinasikan dengan penggunaan subquery Exist dan IN pada basis data MySQL untuk optimalisasi database rawat jalan rumah sakit, dan menemukan bahwa tabel yang telah diindeks menghasilkan waktu akses rata-rata yang jauh lebih cepat dibandingkan tabel tanpa indeks,

sehingga menegaskan pentingnya strategi indexing dan optimalisasi kueri dalam mendukung kinerja sistem berbasis MySQL secara keseluruhan.

B. Teori Perancangan Sistem

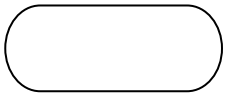
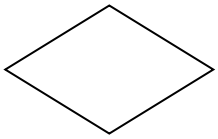
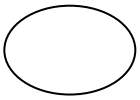
14. Flowchart

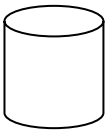
Flowchart atau diagram alir merupakan representasi grafis yang digunakan untuk menggambarkan secara rinci tahapan penyelesaian suatu masalah atau algoritma menggunakan simbol-simbol standar yang dihubungkan dengan panah untuk menunjukkan urutan langkah dan alur logika program. Smrti et al. (2023) menjelaskan bahwa flowchart menjadi salah satu standar yang umum digunakan untuk menggambarkan algoritma, sehingga tahapan penyelesaian masalah dapat lebih mudah dipahami melalui representasi visual dibandingkan uraian tertulis.

Salah satu perangkat lunak yang memudahkan pembuatan flowchart adalah Flowgorithm, yang memungkinkan penggunanya membuat diagram alur sekaligus menjalankannya secara langsung untuk memverifikasi logika program yang dirancang. *Hisamuddin & Siregar (2024)* menunjukkan bahwa penerapan Flowgorithm bersama dengan editor kode seperti Visual Studio Code terbukti membantu mahasiswa pemula memahami konsep perulangan (looping) dan percabangan (conditional) secara lebih terstruktur dan intuitif, sehingga penggunaan flowchart sebagai tahap perancangan sebelum implementasi kode program dapat meningkatkan pemahaman alur logika sistem yang dibangun,

termasuk dalam perancangan sistem deteksi penyakit tanaman padi pada penelitian ini.

Tabel 2.1 Simbol-Simbol Flowchart

Simbol	Nama	Keterangan
	Terminal	Digunakan untuk menandai awal dan akhir program
	Input / Output	Digunakan untuk menyatakan proses masukkan dan keluaran data
	Process	Digunakan untuk menunjukkan proses atau pengolahan data
	Decision	Digunakan untuk menunjukkan percabangan kondisi ya atau tidak
	Flow Line	Digunakan untuk menunjukkan arah alur proses antar simbol
	Preparation	Digunakan untuk menunjukkan persiapan atau inisialisasi proses
	On Page Reference	Digunakan sebagai penghubung alur dalam satu halaman yang sama
	Off Page Reference	Digunakan sebagai penghubung alur pada halaman yang berbeda
	Document	Digunakan untuk menunjukkan dokumen atau laporan yang di hasilkan sistem

	Database	Digunakan untuk menunjukkan penyimpanan data dalam database sistem

(Sumber: Smrti et al., 2023)

15. UML (Unified Modelling Language)

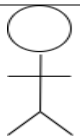
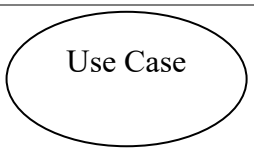
Unified Modeling Language (UML) merupakan bahasa pemodelan standar yang digunakan untuk merancang, mendokumentasikan, dan memvisualisasikan struktur maupun perilaku sistem perangkat lunak. Sumiati M et al. (2021) menjelaskan bahwa UML pertama kali dikembangkan pada pertengahan tahun 1990-an oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson berdasarkan masukan dari komunitas pengembang perangkat lunak, dan pada tahun 1997 distandarkan melalui Object Management Group (OMG), dengan empat jenis diagram yang paling umum digunakan yaitu Use Case Diagram, Class Diagram, Activity Diagram, dan Sequence Diagram. Aryani et al. (2025) menerapkan UML sebagai bahasa grafis untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak dalam proses digitalisasi sistem informasi perpustakaan, dan menunjukkan bahwa pemodelan berbasis UML mampu mengatasi permasalahan pendataan yang tidak terdokumentasi dengan baik serta meningkatkan akurasi pencatatan data, sehingga pendekatan serupa relevan diterapkan dalam merancang sistem deteksi otomatis berbasis citra pada penelitian ini agar alur proses sistem terdokumentasi secara jelas dan terstruktur.

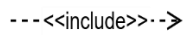
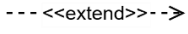
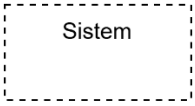
Jenis-jenis diagram UML yang digunakan dalam perancangan sistem ini meliputi:

a. Use Case Diagram

Menurut Nur Sa'adah & Voutama (2023), use case diagram merupakan pemodelan perilaku untuk memetakan fungsi-fungsi yang tersedia di dalam sistem, aktor yang terlibat, serta hak akses pengguna di dalamnya. Senada dengan hal tersebut, Khairunnisa & Voutama (2024) menyatakan bahwa use case diagram menggambarkan fungsionalitas sistem serta interaksi antara aktor dan sistem, di mana identitas aktor dan fungsi-fungsi yang tersedia dalam sistem informasi yang dikembangkan dijelaskan secara rinci di dalam use case tersebut. Adapun komponen standar yang digunakan dalam penyusunan diagram ini dapat dilihat pada Tabel 2.2 Simbol-Simbol Use Case Diagram berikut.

Tabel 2.2 Simbol-Simbol Use Case Diagram

Simbol	Nama	Keterangan
	Actor	Pengguna yang berinteraksi dengan sistem
	Use Case	Fungsi atau layanan yang di sediakan sistem untuk digunakan oleh actor
	Association	Menghubungkan actor dengan use case

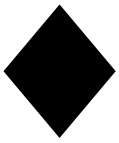
	Include	Use case yang selalu di panggil
	Extend	Use case yang kadang dipanggil atau digunakan oleh use case lain
	System Boundary	Batas yang memisahkan sistem dengan lingkungan luar sistem
	Generalization	Hubungan pewarisan antara aktor atau use case

(Sumber: Nur Sa'adah & Voutama 2023)

b. Activity Diagram

Simare Mare et al. (2022) menjelaskan bahwa activity diagram merupakan rancangan aliran aktivitas atau alur kerja dalam sebuah sistem, dengan komponen berbentuk tertentu yang dihubungkan tanda panah menuju urutan aktivitas dari awal hingga akhir. Di samping itu, Anardani et al. (2023) menunjukkan bahwa activity diagram dirancang setelah use case diagram untuk menguraikan alur aktivitas data pada tiap fungsi sistem yang telah didefinisikan, sehingga membantu menggambarkan interaksi aktor dengan tiap proses secara berurutan sebelum tahap perancangan sequence diagram dilakukan. Adapun komponen standar yang digunakan dalam penyusunan diagram ini dapat dilihat pada Tabel 2.3 Simbol-Simbol Activity Diagram berikut.

Tabel 2.3 Simbol-Simbol Activity Diagram

Symbol	Nama	Keterangan
	Initial Node	Titik awal yang menandakan dimulainya aktivitas dalam diagram
	Action/Activity	Menggambarkan sebuah aksi atau aktivitas yang dilakukan dalam proses
	Decision Node	Titik percabangan untuk menentukan kondisi Ya atau Tidak
	Fork / Join	Percabangan paralel (fork) atau penggabungan aktivitas (join)
	Transition	Menunjukkan alur perpindahan dari satu aktivitas ke aktivitas lain
	Activity Final Node	Titik akhir yang menandakan seluruh aktivitas dalam diagram telah selesai

(Sumber: Simare Mare et al. 2022)

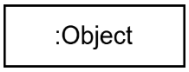
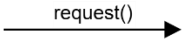
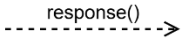
c. Squence Diagram

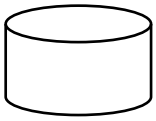
Daus & Utami (2024) menjelaskan bahwa sequence diagram menggambarkan interaksi antar objek di dalam maupun di luar sistem melalui rangkaian pesan yang disusun berdasarkan urutan waktu terjadinya. Diagram ini berasosiasi khusus dengan use case untuk

memperlihatkan tahapan proses secara mendetail demi menghasilkan output tertentu sebagai respons dari suatu aksi.

Nistrina & Sahidah (2022) menjelaskan bahwa sequence diagram menjelaskan dan memodelkan use case, yang berfungsi memodelkan logika dari suatu method, operasi, fungsi, ataupun prosedur.. Adapun komponen standar yang digunakan dalam penyusunan diagram ini dapat dilihat pada Tabel 2.4 Simbol-Simbol Sequence Diagram berikut.

Tabel 2.4 Simbol-Simbol Sequence Diagram

Symbol	Nama	Keterangan
	Object / Actor	Entitas yang terlibat dalam interaksi. Contoh: Flutter, Laravel, Flask, Database
	Lifeline	Garis putus-putus vertikal yang menunjukkan keberadaan objek sepanjang waktu
	Activation Bar	Menunjukkan periode waktu objek sedang aktif memproses pesan
	Synchronous Message	Pesan yang dikirim dan pengirim menunggu balasan dari penerima
	Return Message	Pesan balasan yang dikirimkan dari objek penerima ke pengirim

	Self Message	Pesan yang dikirim objek kepada dirinya sendiri untuk proses internal
	Combined Fragment	Menggambarkan kondisi percabangan dalam interaksi (alt/loop/opt)
	Database / Entity	Menggambarkan tempat penyimpanan data persisten yang digunakan untuk menyimpan atau memanggil data dalam interaksi sistem.

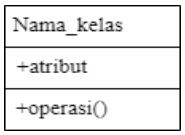
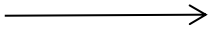
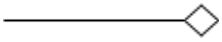
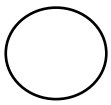
(Sumber: Daus & Utami 2024)

d. Class Diagram

Class diagram merupakan diagram yang menggambarkan struktur statis sistem dengan menunjukkan kelas, atribut, metode, serta hubungan antar-kelas sebagai kerangka dasar sebelum diimplementasikan ke dalam kode program Ramdany et al. (2024). Khasani & Subrata (2025) menambahkan bahwa class diagram menampilkan berbagai elemen penting dari kelas termasuk nama kelas, atribut, dan metode yang saling berhubungan melalui berbagai jenis relasi seperti asosiasi, agregasi, komposisi, pewarisan, atau dependensi, sehingga memberikan gambaran visual yang jelas tentang struktur internal sistem perangkat lunak dan bagaimana komponen-komponen utamanya saling berhubungan. Adapun komponen standar yang

digunakan dalam penyusunan diagram ini dapat dilihat pada Tabel 2.5 Simbol-Simbol Class Diagram berikut.

Tabel 2.5 Simbol-Simbol Class Diagram

Symbol	Nama	Keterangan
	*Class*	Kelas pada sistem dan memiliki atribut dan Operasi
	Association	Merupakan relasi antarkelas, yang dilengkapi dengan multiplicity
	Generalisasi	Merupakan relasi kelas dengan bermakna generalisai
	Dependency	Merupakan relasi keberuntungan antar kelas.
	*Aggregation*	Merupakan relasi kelas bermakna semua bagian (*whole-part*)
 **Name_Interface**	Interface	Merupakan bagian yang ditampilkan namun tidak ada isi dan atribut
	Directed Association	Merupakan satu kelas Digunakan oleh kelas lain dan memiliki multiplicity yang spesifik.

(Sumber: Khasani & Subrata 2025)

C. Kajian Empiris

Roseno et al. (2024) dalam penelitiannya yang berjudul "Comparing CNN Models for Rice Disease Detection: ResNet50, VGG16, and MobileNetV3-Small" membandingkan performa tiga arsitektur CNN pretrained untuk klasifikasi penyakit tanaman padi (Brown Spot, Leaf Scald, Rice Blast, Rice Tungro, dan Sheath Blight). Hasil penelitian menunjukkan bahwa MobileNetV3-Small memberikan akurasi terbaik pada data uji sebesar 79%. Perbedaan dengan penelitian ini terletak pada penggunaan arsitektur MobileNetV2 yang lebih ringan dan cocok untuk perangkat mobile, serta implementasinya dalam bentuk aplikasi Android berbasis Flutter.

Rahmadani et al. (2023) dalam penelitiannya yang berjudul "Klasifikasi Jenis Penyakit Pada Daun Padi berdasarkan pada Warna dan Tekstur menggunakan CNN" menggunakan arsitektur CNN dengan memanfaatkan informasi warna dan tekstur daun untuk mengklasifikasikan empat jenis kondisi daun padi (Blast, Bacteria, Fungi, dan Tanpa Penyakit), dan menerapkan teknik augmentasi data yang mampu meningkatkan akurasi model secara signifikan hingga 92,5%. Perbedaan dengan penelitian ini terletak pada penggunaan arsitektur MobileNetV2 dengan pendekatan transfer learning serta objek penelitian yang difokuskan pada penyakit blas, hawar daun bakteri, dan bercak coklat.

Setyawan et al. (2026) dalam penelitiannya yang berjudul "Leafy AI: Integrating MobileNetV2 and TensorFlow Lite into a Flutter-Based Application for Real-Time Ornamental Plant Recognition" mengintegrasikan

model MobileNetV2 yang dikonversi ke format TensorFlow Lite ke dalam aplikasi berbasis Flutter untuk pengenalan tanaman hias secara real-time sepenuhnya pada perangkat (on-device), dan mencapai akurasi top-one sebesar 0,89. Penelitian ini menjadi referensi dalam pemanfaatan MobileNetV2 pada arsitektur aplikasi berbasis Flutter. Perbedaan dengan penelitian ini terletak pada objek klasifikasi yang berfokus pada penyakit tanaman padi serta pendekatan sistem yang sepenuhnya online, di mana proses inferensi dilakukan pada server Flask dan hasilnya disinkronkan secara real-time melalui backend Laravel.

Saif & Arbaiy (2023) dalam penelitiannya yang berjudul "Plant Disease Detection Application Using Deep Learning (PLANTSCARE)" mengimplementasikan aplikasi deteksi penyakit tanaman menggunakan Flutter framework dan bahasa pemrograman Dart, dengan model deep learning yang dilatih menggunakan Google Teachable Machine serta layanan hosting dan basis data dari Firebase. Hasil penelitian menunjukkan bahwa Flutter merupakan pilihan yang tepat untuk pengembangan aplikasi deteksi penyakit tanaman karena kemudahan pengembangan dan performa yang baik. Perbedaan dengan penelitian ini terletak pada penambahan backend Laravel sebagai API Gateway yang menangani autentikasi, penyimpanan data, dan penerusan ke server inferensi Flask.

AL-atraqchi (2022) dalam penelitiannya yang berjudul "A Proposed Model for Building a Secure Restful API to Connect between Server-side and Mobile Application Using Laravel Framework with Flutter Toolkits"

membahas perancangan model REST API yang aman menggunakan Laravel dan Flutter dengan memanfaatkan Laravel Sanctum untuk autentikasi dan otorisasi pengguna. Hasil penelitian menunjukkan bahwa kombinasi Laravel dan Flutter dengan Laravel Sanctum menghasilkan sistem yang aman, efisien, dan mudah dikembangkan. Penelitian ini menjadi acuan dalam perancangan arsitektur backend Laravel dan autentikasi API pada penelitian yang dikembangkan.

Berdasarkan kelima penelitian terdahulu tersebut, dapat disimpulkan bahwa pemanfaatan metode Deep Learning dengan arsitektur MobileNetV2 sangat efektif untuk klasifikasi penyakit tanaman, sementara kombinasi framework Flutter dan Laravel menawarkan performa aplikasi serta keamanan REST API yang optimal. Persamaan utama penelitian ini dengan literatur acuan terletak pada penggunaan MobileNetV2 sebagai model pengenal citra daun padi serta adopsi Flutter dan Laravel sebagai fondasi utama pembangunan perangkat lunak. Sementara itu, celah penelitian (research gap) sekaligus pembeda utama yang diajukan dalam penelitian ini adalah integrasi sistem diagnosis penyakit padi secara terpusat, yang menghubungkan aplikasi mobile Flutter milik petani dengan dashboard admin Laravel secara real-time untuk mempermudah pemantauan, pembaruan data penyakit, serta penyimpanan riwayat diagnosis yang aman.

D. Kerangka Berpikir

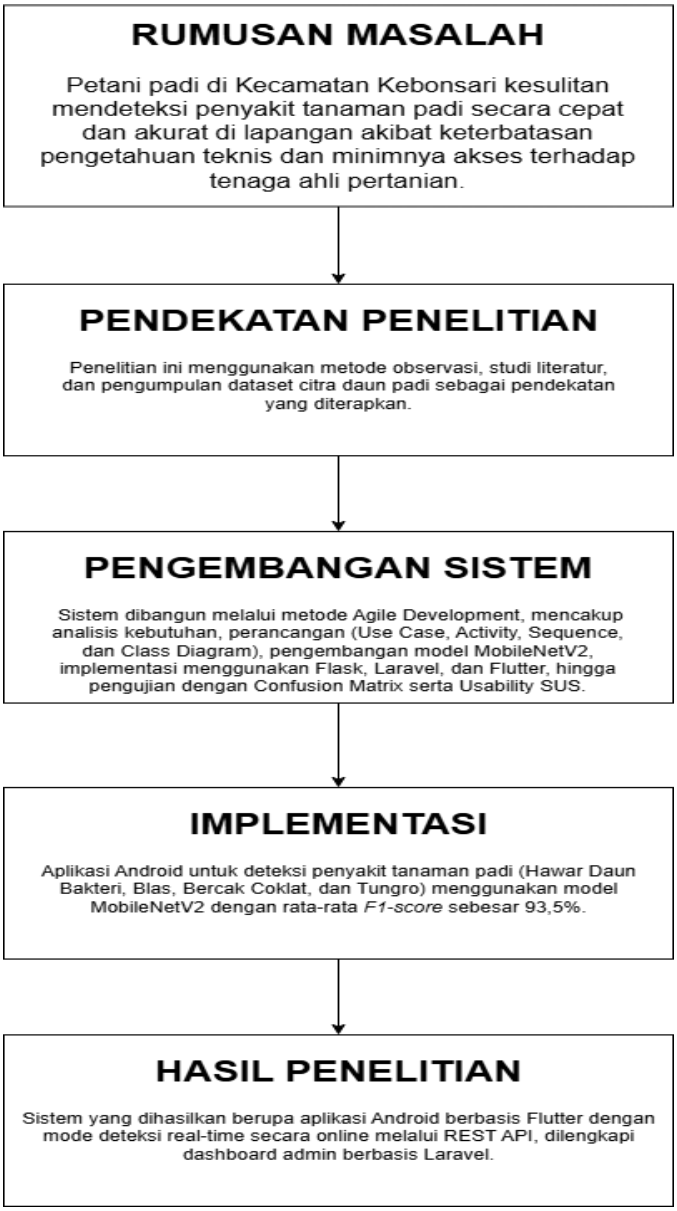
Kerangka berpikir merupakan gambaran alur logis yang menghubungkan antara identifikasi masalah, solusi yang diusulkan, dan hasil yang diharapkan

dalam penelitian ini. Permasalahan utama yang melatarbelakangi penelitian ini adalah keterbatasan petani padi di Kabupaten Madiun dalam mendeteksi penyakit tanaman padi secara cepat dan akurat di lapangan. Keterbatasan pengetahuan teknis petani, minimnya akses terhadap tenaga ahli pertanian, serta metode pengamatan konvensional yang tidak efisien menjadi faktor-faktor yang menyebabkan keterlambatan penanganan penyakit dan berujung pada kerugian hasil panen yang signifikan.

Sebagai solusi atas permasalahan tersebut, penelitian ini mengusulkan pengembangan sistem deteksi penyakit tanaman padi berbasis mobile menggunakan arsitektur MobileNetV2 dengan pendekatan transfer learning. Model MobileNetV2 dipilih karena kemampuannya yang terbukti efektif dalam klasifikasi citra dengan ukuran model yang ringan dan efisiensi komputasi yang tinggi, sehingga cocok untuk diimplementasikan pada perangkat Android. Dataset yang digunakan merupakan hasil kurasi mandiri berjumlah 320 citra, terdiri atas empat kelas penyakit, yaitu Hawar Daun Bakteri, Blas, Tungro, dan Bercak Coklat, yang diambil dari tanaman padi varietas IR64, Ciherang, dan Inpari 32.

Sistem yang dikembangkan terdiri dari tiga lapisan utama yang saling terintegrasi. Lapisan pertama adalah model MobileNetV2 yang bertugas melakukan inferensi untuk mengenali penyakit dari citra daun padi. Lapisan kedua adalah server inferensi Flask Python yang menjalankan model dan mengembalikan hasil prediksi. Lapisan ketiga adalah backend Laravel yang berperan sebagai API Gateway untuk menangani autentikasi, validasi,

penyimpanan data, dan komunikasi dengan server Flask (AL-atraqchi, 2022). Lapisan antarmuka pengguna diimplementasikan menggunakan Flutter yang berkomunikasi dengan backend melalui REST API (AL-atraqchi, 2022) untuk mengirim citra daun padi ke server dan menerima hasil deteksi secara real-time. Hasil yang diharapkan dari penelitian ini adalah sebuah aplikasi Android yang mampu mendeteksi penyakit tanaman padi secara akurat dan mudah digunakan oleh petani di lapangan, sehingga dapat membantu mengurangi kerugian akibat serangan penyakit dan meningkatkan produktivitas pertanian padi di Kabupaten Madiun. Adapun alur kerangka berpikir penelitian ini dapat dilihat pada gambar berikut.



Gambar 2.1 Kerangka Berpikir Penelitian.