Pengembangan Aplikasi Si-eMOO Berbasis Android Untuk Klasifikasi Penyakit Sapi Menggunakan *Rapid Application Development*

1st Fatkhurrohman Purnomo
Fakultas Informatika
Direktorat Kampus Purwokerto Universitas Telkom
Purwokerto, Indonesia
fathpurn@student.telkomuniversity.ac.id

2st Nicolaus Euclides Wahyu Nugroho
Fakultas Informatika
Direktorat Kampus Purwokerto Universitas Telkom
Purwokerto, Indonesia
nicolausn@telkomuniversity.ac.id

Abstrak – **Peternakan sapi perah di Desa Tumiyang masih menghadapi kendala dalam pemantauan kesehatan ternak secara akurat dan cepat akibat kurangnya pemanfaatan teknologi, yang berdampak pada produktivitas hewan yang kurang optimal. Oleh karena itu, dibutuhkan sebuah solusi teknologi yang mampu membantu peternak melakukan deteksi dini penyakit ternak secara mandiri. Penelitian ini bertujuan untuk mengembangkan fitur Deteksi Sakit pada aplikasi Si-eMOO berbasis Android dengan pendekatan *Rapid Application Development* (RAD). Fitur ini memanfaatkan teknologi *computer vision* untuk mengklasifikasikan penyakit sapi melalui analisis citra yang diambil menggunakan kamera ponsel pengguna. Proses pengembangan aplikasi melibatkan tahapan perancangan antarmuka, integrasi model klasifikasi berbasis CNN, serta pengujian sistem. Pengujian fungsional menggunakan metode *blackbox* menunjukkan bahwa seluruh skenario uji berhasil 100% tanpa error. Sementara itu, pengujian kegunaan dengan kuesioner *System Usability Scale* (SUS) kepada 15 responden menghasilkan skor rata-rata 76,245 yang tergolong dalam kategori "Baik" dan "*Acceptable*". Berdasarkan hasil tersebut, dapat disimpulkan bahwa fitur yang dikembangkan terbukti fungsional, efektif, dan mudah digunakan oleh peternak dalam mendeteksi penyakit sapi secara mandiri.**

***Kata Kunci* – deteksi penyakit, *computer vision*, peternakan sapi perah, RAD, aplikasi Android, SUS**

I. PENDAHULUAN

Peternakan sapi perah di Desa Tumiyang merupakan komoditas penting dengan potensi ekonomi yang signifikan, namun pengelolaannya masih bersifat tradisional sehingga menimbulkan berbagai tantangan, terutama dalam pemantauan kesehatan ternak secara akurat dan cepat [1]. Minimnya pemanfaatan teknologi dalam proses pengelolaan menyebabkan produktivitas hewan menjadi tidak optimal serta kesejahteraan ternak yang kurang terjamin [2]. Kesenjangan ini menuntut adanya solusi berbasis teknologi informasi yang dapat membantu peternak dalam mendeteksi penyakit ternak secara mandiri dan efisien.

Sebagai upaya menjawab permasalahan tersebut, dikembangkan aplikasi Si-eMOO berbasis Android sebagai solusi digital yang ditujukan untuk meningkatkan efisiensi manajemen peternakan. Aplikasi ini menawarkan berbagai fitur, salah satunya adalah fitur Deteksi Sakit yang memanfaatkan teknologi visi komputer *(computer vision)* untuk melakukan klasifikasi penyakit sapi berdasarkan citra yang diambil menggunakan kamera ponsel pengguna. Dengan memanfaatkan model klasifikasi dari sumber terbuka tanpa pelatihan ulang, fitur ini memberikan kemudahan bagi peternak dalam mengenali penyakit yang mungkin diderita oleh sapi mereka, sekaligus mendorong penanganan dini yang lebih efektif [3].

Hasil wawancara dengan peternak menunjukkan bahwa meskipun aplikasi Si-eMOO telah dikembangkan, pemanfaatannya masih belum maksimal dan informasi terkait penyakit sapi masih terbatas. Peternak menyatakan kebutuhan yang tinggi terhadap fitur deteksi penyakit yang akurat, praktis, dan mudah digunakan. Hal ini mendorong perlunya pengembangan fitur Deteksi Sakit secara lebih fokus agar mampu memenuhi kebutuhan tersebut, sekaligus meningkatkan kesejahteraan hewan dan mengurangi risiko kerugian akibat keterlambatan penanganan penyakit [4].

Dalam pengembangan fitur ini, metode *Rapid Application Development* (RAD) dipilih karena mampu mempercepat proses pengembangan, memberikan fleksibilitas terhadap perubahan kebutuhan, serta memungkinkan keterlibatan pengguna dalam setiap tahapan iterasi [5] [6]. Bahasa pemrograman yang digunakan meliputi Javascript dengan *framework* React Native untuk pengembangan aplikasi Android, serta Python untuk pengolahan model klasifikasi [7]. Pengujian fungsional dilakukan dengan metode *blackbox testing*, sedangkan evaluasi kegunaan dilakukan dengan pendekatan *System Usability Scale* (SUS) [8] [9].

Tujuan dari penelitian ini adalah untuk mengevaluasi efektivitas metode RAD dalam pengembangan fitur Deteksi Sakit pada aplikasi Si-eMOO, serta memastikan fungsionalitas sistem berjalan sesuai dengan yang direncanakan [6]. Selain itu, penelitian ini bertujuan mengukur sejauh mana fitur ini dapat diterima dan digunakan dengan baik oleh peternak dalam konteks penggunaan nyata di lapangan.

II. KAJIAN TEORI

A. Pengembangan

Dalam pengembangan perangkat lunak, terdapat berbagai metode yang dapat digunakan, masing-masing memiliki kelebihan dan kekurangan. Metode pengembangan sendiri merupakan salah satu pendekatan dalam penelitian yang bertujuan untuk menghasilkan suatu produk atau mengukur keefektifan dari produk tersebut [10]. Secara umum, pengembangan dapat dipahami sebagai proses perancangan atau redesain terhadap sistem atau produk yang sudah ada sebelumnya, dengan tujuan untuk meningkatkan efisiensi dan efektivitasnya [11].

B. Aplikasi

Aplikasi merupakan program atau perangkat lunak yang dirancang untuk menyelesaikan tugas tertentu melalui proses dan prosedur aliran data dalam suatu infrastruktur teknologi informasi, yang disesuaikan dengan tingkat dan kebutuhan pengguna [12]. Keberadaan aplikasi sangat penting dalam mewujudkan digitalisasi, karena mampu mempermudah berbagai aktivitas serta meningkatkan efisiensi dalam kehidupan sehari-hari.

C. Aplikasi Si-eMOO

Aplikasi Si-eMOO adalah solusi inovatif untuk mengatasi berbagai kesulitan yang dihadapi peternak sapi perah di Desa Tumiyang. Dengan menggunakan teknologi berbasis Android, aplikasi ini tidak hanya membantu dalam manajemen peternakan sapi perah, tetapi juga menjadi platform edukasi bagi para peternak untuk mengelola dan mengolah susu sapi perah serta limbah kotoran sapi dengan lebih efisien [13].

D. *Convolutional Neural Network* (CNN)

CNN adalah singkatan dari *Convolutional Neural Network* (CNN), yang merupakan jenis struktur jaringan saraf tiruan yang banyak digunakan dalam pengenalan pola dan pengolahan citra [14]. Lapisan konvolusi CNN memungkinkan jaringan untuk secara otomatis mengekstrak elemen penting dari data gambar. CNN telah terbukti sangat efektif dalam melakukan berbagai fungsi penglihatan komputer, seperti deteksi objek, segmentasi gambar, dan klasifikasi gambar.

E. Python

Python adalah bahasa pemrograman yang berfokus pada keterbacaan kode dan serbaguna. Python terkenal karena kemampuannya yang luas, sintaksisnya yang mudah dipahami, dan pustaka standarnya yang besar dan luas. Komunitas yang besar dan aktif mendukung popularitas bahasa ini. Python mendukung banyak paradigma pemrograman, terutama pemrograman berorientasi objek, imperatif, dan fungsional [15].

F. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) dibuat untuk memvisualisasikan dan mendokumentasikan hasil analisis dan desain secara visual. UML menggunakan *sintaks* pemodelan untuk menggambarkan dan menentukan sistem perangkat lunak, yang mempermudah proses pengembangannya. Dalam proses pengembangan aplikasi, tujuan utama UML adalah untuk memberi pengembang kemampuan untuk berkomunikasi dengan lebih efektif, mengeksplorasi potensi desain, dan merancang arsitektur perangkat lunak [16].

Pada pengembangan aplikasi ini, digunakan beberapa jenis diagram UML untuk mendukung proses analisis dan perancangan, antara lain *Use Case* Diagram untuk menggambarkan interaksi antara pengguna dan sistem, *Activity* Diagram untuk menunjukkan alur aktivitas sistem, *Class* Diagram untuk memodelkan struktur data dan relasi antar objek, serta *Sequence* Diagram untuk menggambarkan urutan proses dalam skenario tertentu. Penggunaan berbagai diagram ini mempermudah pemahaman sistem secara menyeluruh dan mendukung perancangan aplikasi yang lebih sistematis dan efisien.

G. React Native

React Native adalah *framework* berbasis javascript yang dirilis oleh Facebook pada tahun 2015. Ini memungkinkan pembuat aplikasi *mobile* yang berjalan pada sistem operasi Android dan IOS secara bersamaan. Banyak alasan mengapa React Native menjadi populer, salah satunya karena *framework* yang menggunakan *library* JavaScript yang sudah populer, sehingga pengembang hanya perlu menulis satu kode untuk membuat aplikasi yang berjalan pada sistem operasi Android dan IOS, yang mengurangi waktu dan biaya pembuatan [17].

H. *Rapid Application Development* (RAD)

Metode pengembangan perangkat lunak cepat (RAD) menggunakan metode *prototyping* dan orientasi objek. Tujuan utamanya adalah membuat sistem yang baik sambil mengurangi waktu dan biaya dalam proses pengembangan. Metode ini dimaksudkan untuk mempercepat siklus pengembangan dan memungkinkan perubahan kebutuhan diatasi dengan cepat [18].



GAMBAR 1. *RAPID APPLICATION DEVELOPMENT* [RAD] Proses RAD terdiri dari tiga tahap utama, seperti yang ditunjukkan pada gambar di atas [19]:

1. Tahap Perencanaan Syarat

   Ini adalah tahap awal, di mana *analyst* dan pengguna bekerja sama untuk menentukan kebutuhan sistem secara menyeluruh. Pada tahap ini, pengembangan sistem dimulai.

2. Tahap *Workshop Design*

   Tahap di mana proses desain dan pengembangan sistem secara aktif dilakukan Dimungkinkan bagi pengguna untuk memberikan umpan balik langsung terhadap desain yang dibuat, yang akan memungkinkan perubahan sebelum langkah implementasi. Dalam tahap ini, kerja sama tim pengembang dan pengguna sangat penting.

3. Tahap Implementasi

   Tahap di mana tim pengembang membuat program yang diperlukan berdasarkan desain yang telah disepakati sebelumnya. Untuk menjamin keberhasilan dan kualitas sistem yang digunakan, *output* yang dihasilkan dari langkah ini akan diuji [40]. Tahap ini merupakan titik tertinggi dari *Rapid Application Development* (RAD), di mana solusi yang telah dirancang akan diimplementasikan.

I. Blackbox Testing

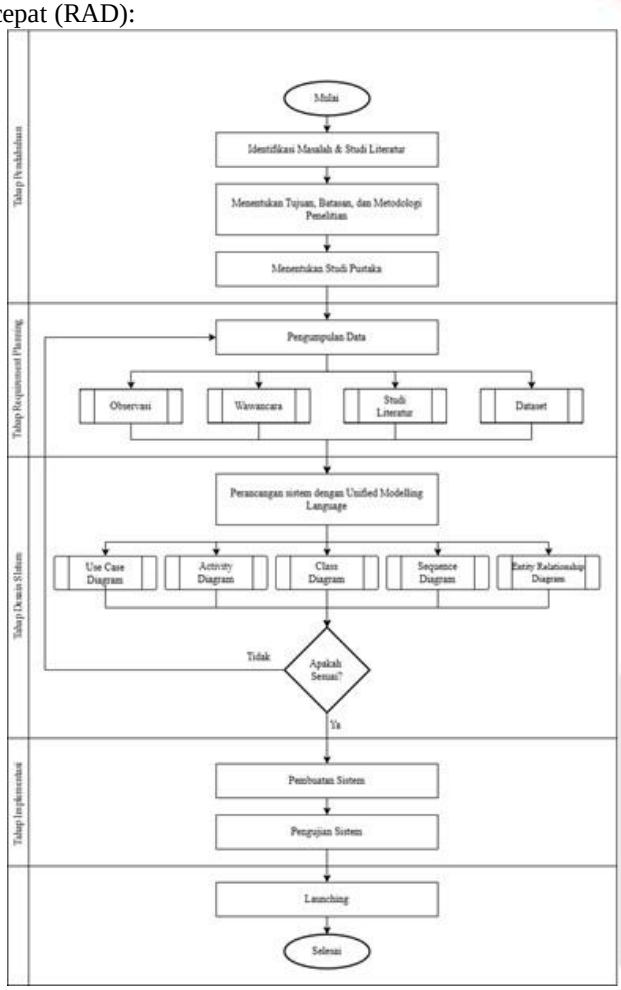
Metode pengujian perangkat lunak yang dikenal sebagai pengujian *blackbox* berkonsentrasi pada input dan output aplikasi dan memeriksa struktur kode, rincian implementasi, dan jalurnya [8]. Proses pengujian ini dilakukan untuk memastikan bahwa aplikasi sudah beroperasi seperti yang diharapkan.

J. System Usability Scale (SUS)

System Usability Scale (SUS) dapat digunakan untuk mengevaluasi *usability*. Digunakan untuk mengevaluasi kemudahan penggunaan produk, sistem, atau layanan [9]. Tujuannya adalah untuk menyelesaikan masalah yang mungkin dihadapi pengguna serta membuat desain yang lebih efisien dan mudah digunakan.

III. SISTEM YANG DIBANGUN

Gambar berikut menunjukkan langkah-langkah penelitian yang akan dilakukan peneliti berdasarkan metode pengembangan sistem pengembangan aplikasi cepat (RAD):



GAMBAR 2.
ALUR PENELITIAN

Penelitian ini mengikuti metodologi pengembangan *Rapid Application Development* (RAD), yang dirancang untuk mempercepat siklus pengembangan sistem dan memungkinkan penyesuaian cepat terhadap perubahan kebutuhan. Prosesnya melibatkan beberapa tahapan utama yang saling berurutan untuk menghasilkan aplikasi yang fungsional dan sesuai kebutuhan peternak.

Tahap Pendahuluan dimulai dengan identifikasi masalah, yaitu belum adanya fitur klasifikasi penyakit pada aplikasi Si-eMOO yang sudah ada, serta kesulitan peternak di Desa Tumiyang dalam mengelola kesehatan sapi. Tahap ini juga mencakup studi literatur mendalam dan penentuan tujuan, batasan, serta metodologi penelitian yang akan digunakan, yaitu RAD.

Selanjutnya, Tahap *Requirement Planning* berfokus pada analisis kebutuhan sistem secara menyeluruh. Pengumpulan data dilakukan melalui observasi langsung di peternakan Bapak Sutarman dari Kelompok Tani Ternak Lestari 1 pada 20 Februari 2025, wawancara dengan beliau untuk memahami kebutuhan deteksi penyakit, studi literatur untuk teori terkait klasifikasi, serta pengumpulan dataset penyakit sapi (FMD, IBK, LSD) dari Kaggle dalam format HDF5. Hasil dari tahap ini merumuskan fitur-fitur yang akan dikembangkan, seperti klasifikasi jenis penyakit, penanganan pertama, data penyakit sapi, dan pendataan riwayat penyakit.

Perancangan Desain Sistem dilakukan berdasarkan kebutuhan yang telah disepakati. Pada tahap ini, peneliti merancang sistem menggunakan *Unified Modeling Language* (UML), yang mencakup *Use Case* Diagram untuk memetakan aktor dan fungsionalitas, *Activity* Diagram untuk menggambarkan alur kerja proses seperti login, deteksi sakit, dan unggah gambar, *Sequence* Diagram untuk mengilustrasikan interaksi objek, dan *Class* Diagram untuk menunjukkan struktur data sistem. Selain itu, dibuat juga Desain *High Fidelity* yang merupakan prototipe tampilan antarmuka pengguna (UI) yang mendetail dan mendekati aplikasi sesungguhnya, menggunakan Figma.

Tahap Implementasi melibatkan pembuatan dan pengujian sistem. Sistem dibangun menggunakan JavaScript dengan *framework* React Native untuk *frontend* aplikasi Android, serta Python dengan metode *Convolutional Neural Network* (CNN) untuk *backend* pemrosesan citra dan klasifikasi penyakit sapi. Model CNN dilatih menggunakan dataset dari Kaggle. Setelah pembangunan, dilakukan pengujian fungsionalitas menggunakan *blackbox testing* dan pengujian kegunaan (*usability testing*) dengan *System Usability Scale* (SUS) untuk memastikan sistem berjalan sesuai harapan dan mudah digunakan.

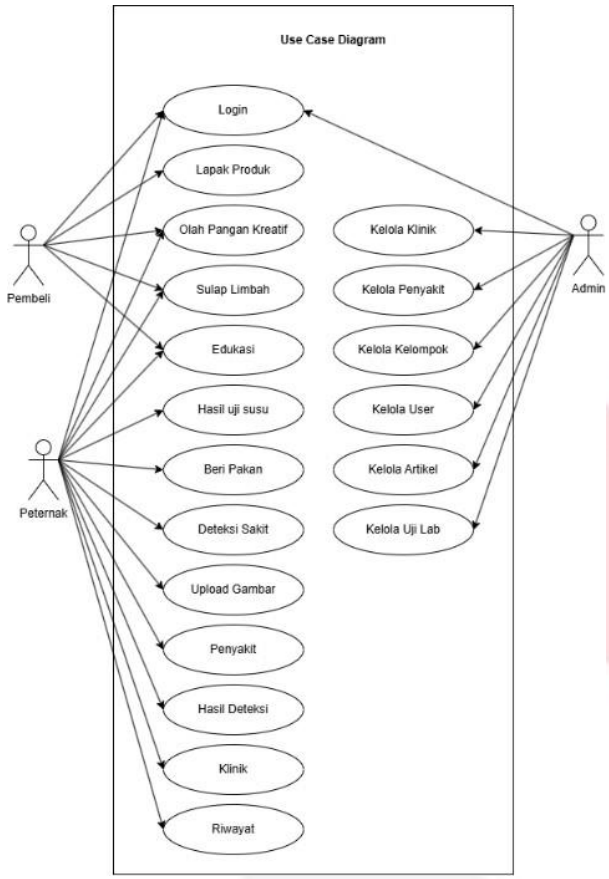
Terakhir, tahap Launching dilakukan setelah aplikasi berhasil dikembangkan dan lolos uji. Aplikasi Si-eMOO kemudian dapat diunduh dan diinstal dalam format .apk pada perangkat Android pengguna. Arsitektur *deployment* sistem ini memisahkan *backend* utama (Express.js di Vercel), *backend* klasifikasi (Flask di Azure VPS), dan penyimpanan data (PostgreSQL dan ImageKit), dengan aplikasi *client* diinstal langsung di perangkat pengguna

IV. HASIL DAN PEMBAHASAN

A. Desain Sistem

1. *Use Case* Diagram

Menggambarkan fungsionalitas sistem dari sudut pandang pengguna.



GAMBAR 3. *USE CASE* DIAGRAM

Gambar diatas menjelaskan *use case* diagram dari aplikasi Si-eMOO yang memiliki aktor pengguna, dan harus login terdahulu jika ingin mengakses semua fitur yang tersedia pada aplikasi.

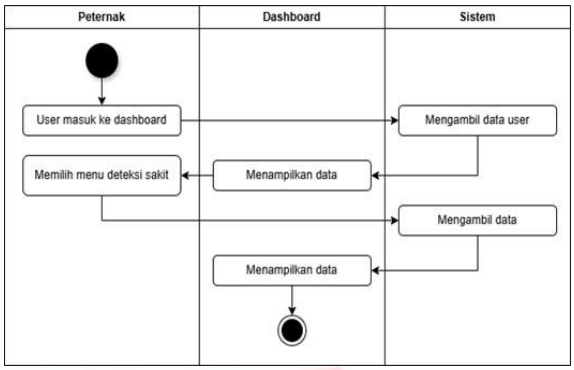
TABEL 1
DESKRIPSI AKTOR

No	Aktor	Deskripsi
1	Admin	Mengelola data dan konten dalam sistem
2	Pembeli	Mengakses informasi dan fitur publik
3	Peternak	Pengguna utama aplikasi untuk kegiatan peternakan

2. *Activity* Diagram

Menunjukkan alur proses atau alur kerja sistem untuk berbagai fitur, seperti Login, Deteksi Sakit, Unggah Gambar, Riwayat Deteksi Sakit, Hasil Deteksi, Klinik, dan Penyakit. Diagram ini mengilustrasikan aktivitas, objek, status, transisi status, dan peristiwa yang terjadi dalam sistem.

Berikut merupakan *activity* diagram yang menggambarkan alur proses deteksi sakit dalam sistem yang menjadi fitur utama dalam pengembangan ini.



GAMBAR 4 *ACTIVITY* DIAGRAM DETEKSI SAKIT

Gambar diatas menunjukan *activity* diagram dari deteksi sakit yang dilakukan oleh role pengaju. Pengaju masuk ke dashboard lalu sistem akan mengambil data yang diperlukan dan menampilkan di dashboard, kemudian pengaju dapat memilih menu deteksi sakit. Saat menu deteksi sakit dipilih, sistem maka akan mengambil data dan menampilkan deteksi sakit.

3. *Sequence* Diagram

Menggambarkan interaksi antar objek dalam urutan waktu untuk proses-proses spesifik, seperti Login, Deteksi Sakit, Unggah Gambar, Riwayat Deteksi Sakit, Hasil Deteksi, Klinik, dan Penyakit. Diagram ini menyajikan langkah-langkah secara bertahap yang menggambarkan perubahan logis untuk mencapai tujuan tertentu.

Berikut merupakan *sequence* diagram yang menggambarkan alur proses deteksi sakit dalam sistem yang menjadi fitur utama dalam pengembangan ini.



GAMBAR 5 *SEQUENCE* DIAGRAM DETEKSI SAKIT

Sequence diagram pada gambar diatas, menggambarkan interaksi antara pengguna dan sistem dalam proses melihat deteksi sakit. Proses ini dimulai ketika pengguna mengakses halaman deteksi sakit pada aplikasi. Pengguna melakukan permintaan untuk membuka halaman deteksi sakit, yang kemudian diteruskan ke sistem.

Sistem merespons permintaan tersebut dengan memproses data hasil uji yang diperlukan dan mengirimkan tampilan halaman deteksi sakit kembali ke pengguna. Setelah sistem mengirimkan data, pengguna akhirnya dapat melihat tampilan halaman deteksi sakit pada antarmuka aplikasi.

4. *Class* Diagram

Mengilustrasikan struktur statis sistem, meliputi kelaskelas utama (seperti Users, Kelompok, Pengujian, Artikel, Warung, Sakit, Klinik, Kota, Jadwal), atribut-atributnya, serta hubungan antar kelas (asosiasi, agregasi,

inheritance). Diagram ini memberikan gambaran tentang bagaimana data diatur dan saling terkait dalam aplikasi.

5. Desain *High Fidelity*

Menampilkan prototipe tampilan antarmuka pengguna yang mendetail dan mendekati aplikasi sesungguhnya, termasuk elemen antarmuka, warna, ikon, dan *layout* untuk fitur-fitur utama seperti Login, Deteksi Sakit, Unggah Gambar, Riwayat Deteksi Sakit, Hasil Deteksi, Klinik, dan Penyakit. Desain ini dibuat menggunakan Figma untuk memberikan gambaran yang jelas mengenai antarmuka pengguna.

B. Tahapan Iterasi dan Evaluasi

Dalam pengembangan aplikasi Si-eMOO, metodologi *Rapid Application Development* (RAD) diterapkan dengan fokus pada pendekatan *prototyping*. Hanya satu kali iterasi desain antarmuka yang dilakukan, di mana prototipe awal diuji langsung oleh calon pengguna. Berdasarkan evaluasi ini, prototipe aplikasi dianggap sudah sesuai dengan kebutuhan pengguna, termasuk alur navigasi, tata letak menu, kemudahan penggunaan, serta fungsionalitas deteksi dan informasi klinik terdekat. Validasi desain dilakukan dengan menunjukkan hasilnya kepada peternak lokal dan dosen pembimbing, yang menyatakan bahwa aplikasi mudah digunakan, informasinya jelas, dan alurnya sesuai dengan kebutuhan lapangan. Oleh karena itu, tidak diperlukan revisi atau iterasi tambahan, dan desain langsung dilanjutkan ke tahap implementasi.

C. PEMBUATAN SISTEM

1. BACKEND

Dalam pengembangan aplikasi Si-eMOO, sistem backend terdiri dari dua komponen terintegrasi. *Backend* utama menggunakan Express.js untuk menangani autentikasi, manajemen data pengguna dan riwayat deteksi pada PostgreSQL, serta mengelola unggahan gambar ke ImageKit. *Backend* kedua, berbasis Flask (Python), digunakan khusus untuk klasifikasi penyakit sapi menggunakan model CNN. Gambar yang diunggah akan diproses oleh Flask melalui URL, dan hasil prediksi dikembalikan ke Express.js untuk disimpan dan ditampilkan.

2. Frontend

Untuk sisi *frontend*, aplikasi dibangun menggunakan React Native (JavaScript) untuk platform Android. Navigasi utama mengandalkan *drawer* menu dengan kombinasi *stack navigation*.

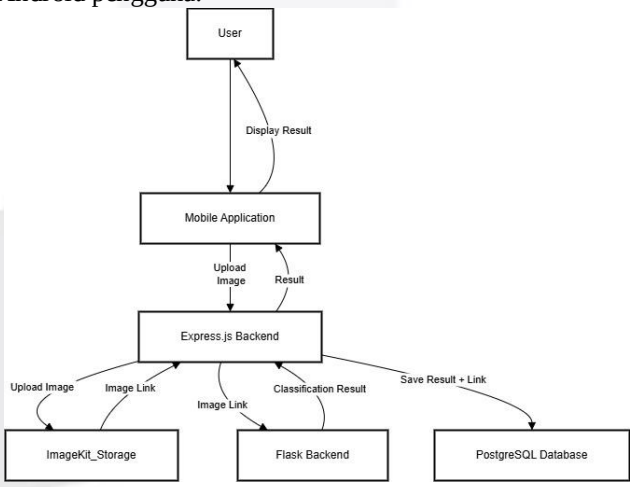


GAMBAR 6.
TAMPILAN HALAMAN UTAMA

Halaman utama "Deteksi Sakit" memungkinkan pengguna mengunggah gambar, melihat hasil deteksi, serta mengakses menu Klinik, Riwayat, dan Penyakit. Pengambilan data dari *backend* dilakukan dengan Axios dan dikelola melalui React Query. Desain antarmuka dibuat responsif menggunakan Tailwind CSS dengan skema warna hijau dan putih.

D. Arsitektur Deployment Sistem

Arsitektur sistem bersifat terdistribusi, dengan Express.js di *hosting* di Vercel, Flask di VPS Azure, dan aplikasi *frontend* React Native berjalan di perangkat Android pengguna.



GAMBAR 7
ALUR KERJA SISTEM

Pengguna mengunggah gambar sapi melalui aplikasi android. Gambar ini kemudian dikirim ke *backend* Express.js untuk diteruskan ke layanan penyimpanan gambar (ImageKit). Setelah mendapatkan tautan gambar, Express.js memanggil *endpoint* Flask untuk melakukan klasifikasi penyakit. Hasil klasifikasi yang diperoleh akan disimpan ke database PostgreSQL melalui *backend*

Express.js, dan selanjutnya dikirim kembali ke aplikasi untuk ditampilkan kepada pengguna.

E. HASIL PENGUJIAN

1. *SYSTEM USABILITY SCALE* (SUS)

Hasil dari prosees implementasi metode *System Usability Scale*, yang dilakukan oleh 15 responden dengan melihat *protoype frontend* aplikasi yang disediakan. Tabel 2. Hasil Pengujian SUS

Responden	Skor Hasil Hitung SUS										Jumlah	Skor SUS (Jumlah x 2.5)
	1	2	3	4	5	6	7	8	9	10		
R1	3	2	4	3	3	4	3	4	4	3	33	82,5
R2	4	2	4	2	4	3	3	2	3	3	30	75
R3	3	2	4	3	4	2	5	1	4	3	31	77,5
R4	5	2	4	2	4	3	4	1	4	3	32	80
R5	3	2	4	3	4	2	5	1	4	3	31	77,5
R6	3	2	5	1	5	2	3	3	4	1	29	72,5
R7	3	1	4	1	4	3	3	2	4	2	27	67,5
R8	5	2	5	2	4	2	4	2	4	2	32	80
R9	4	2	4	2	4	2	4	2	4	2	30	75
R10	4	2	5	2	4	3	4	2	4	1	31	77,5
R11	4	2	4	2	4	3	4	2	4	1	30	75
R12	4	1	4	2	4	3	4	2	4	2	30	75
R13	3	1	4	2	4	2	4	2	4	1	27	67,5
R14	4	2	5	3	5	3	3	3	4	2	34	85
R15	4	2	4	2	4	2	4	2	4	2	30	75

Pengujian *usability* dilakukan menggunakan *System Usability Scale* (SUS) dengan 10 item pernyataan berskala Likert (1–5). Penyesuaian skor dilakukan untuk menjaga konsistensi arah penilaian, kemudian dijumlahkan dan dikalikan 2,5 untuk mendapatkan nilai akhir tiap responden. Dari 15 responden, diperoleh rata-rata skor SUS sebesar 76,2, yang termasuk dalam kategori B (*Good*).

2. BlackBox

Pengujian *Blackbox* dilakukan untuk memastikan semua fitur dalam aplikasi bisa digunakan dengan baik sesuai fungsinya. Metode pengujian dilakukan dengan memberikan 10 skenario uji pada fitur deteksi penyakit, upload gambar, riwayat deteksi, daftar penyakit, dan daftar klinik. Pengujian dilakukan oleh 2 orang, yaitu dari ahli dan dari mitra.

Pengujian fungsional dilakukan melalui 10 skenario uji, dengan sistem penilaian *biner* (1 untuk berhasil, 0 untuk gagal). Hasil pengujian menunjukkan tingkat kelayakan 100%, yang diperoleh dari total jumlah nilai pengujian dibagi nilai maksimal dan dikalikan 100%. Hasil ini menunjukkan bahwa aplikasi telah berfungsi dengan baik.

Meskipun seluruh fitur berjalan sesuai harapan, penguji memberikan masukan untuk pengembangan lebih lanjut. Penguji ahli menyarankan penambahan fitur panduan penanganan awal setelah hasil deteksi muncul, agar peternak mengetahui langkah awal yang dapat dilakukan. Sementara itu, dari pihak mitra mengusulkan penambahan daftar klinik terdekat agar lebih sesuai dengan kebutuhan di lapangan.

F. Analisis Hasil Pengujian

Berdasarkan hasil pengujian, aplikasi deteksi penyakit sapi telah berjalan dengan baik, baik secara fungsional maupun dari segi kenyamanan pengguna. Pengukuran menggunakan *System Usability Scale* (SUS) menghasilkan skor rata-rata 76,2 dari 15 responden, yang masuk kategori "baik" (*Good*) dan "dapat diterima" (*Acceptable*).

Pengujian *Blackbox* oleh dua dosen ahli dan satu peternak menunjukkan bahwa seluruh fitur utama unggah gambar, deteksi, hasil, riwayat, informasi penyakit, dan daftar klinik berfungsi tanpa kendala.

Namun, terdapat masukan penting seperti penambahan panduan penanganan awal setelah deteksi, serta perluasan daftar klinik agar lebih sesuai dengan kebutuhan peternak. Secara keseluruhan, aplikasi dinilai stabil dan layak digunakan, dengan peluang pengembangan lebih lanjut.

G. Launching

Setelah proses pengembangan dan pengujian selesai, aplikasi diluncurkan dan telah berhasil dijalankan pada perangkat Android. Aplikasi deteksi penyakit sapi ini kini dapat diakses langsung oleh pengguna melalui *handphone* mereka.

V. KESIMPULAN

Berdasarkan hasil penelitian dan pengembangan yang telah dilakukan, dapat disimpulkan bahwa aplikasi deteksi penyakit sapi berbasis Android berhasil dibangun dan menunjukkan performa yang baik, baik dari sisi fungsionalitas maupun kegunaan. Aplikasi ini menyediakan fitur utama seperti unggah gambar, deteksi penyakit berbasis citra, tampilan hasil deteksi, riwayat pemeriksaan, informasi penyakit, serta daftar klinik hewan yang dirancang untuk mendukung peternak dalam mengenali dan menangani penyakit ternak secara mandiri. Pengujian fungsional dengan metode *blackbox* terhadap 10 skenario menunjukkan bahwa seluruh fitur berjalan sesuai harapan tanpa adanya kesalahan sistem. Sementara itu, pengujian kegunaan menggunakan instrumen *System Usability Scale* (SUS) menghasilkan skor rata-rata 76,2 dari 15 responden, yang termasuk dalam kategori “Baik” dan “Dapat Diterima”. Namun demikian, masih terdapat keterbatasan pada sistem, terutama hasil deteksi yang bersifat statis dan belum dapat menyesuaikan dengan kompleksitas kondisi riil di lapangan. Oleh karena itu, pengembangan lebih lanjut diperlukan untuk meningkatkan akurasi dan fleksibilitas sistem, seperti melalui pengembangan model klasifikasi citra yang lebih canggih menggunakan teknik *deep learning* serta perluasan dataset pelatihan yang lebih representatif. Selain itu, informasi klinik hewan perlu diperkaya, khususnya yang berada di sekitar wilayah pengguna, guna meningkatkan akses terhadap layanan kesehatan ternak. Peningkatan kualitas sistem juga perlu diarahkan pada pemberian informasi yang lebih dinamis dan kontekstual, tidak sekadar berbasis

template statis. Terakhir, untuk memperoleh masukan yang lebih komprehensif dan realistis, disarankan dilakukan uji coba lapangan dalam skala lebih luas agar sistem dapat terus disesuaikan dengan
kebutuhan dan kondisi pengguna sebenarnya
.

REFERENSI

[1] B. P. Utama, "Analisis Kelayakan Finansial Usaha Peternakan Sapi Potong," Stock Peternakan, vol. 2, no. 1, pp. 10–15, 2020.

[2] T. Wiranda and M. Adri, "Rancang Bangun Aplikasi Modul Pembelajaran Teknologi Wan Berbasis Android," Jurnal Vokasional Teknik Elektronika dan Informatika, vol. 7, no. 1, pp. 1–8, 2019.

[3] G. A. W. Satia, E. Firmansyah, and A. Umami, "Perancangan sistem identifikasi penyakit pada daun kelapa sawit (Elaeis guineensis Jacq.) dengan algoritma deep learning convolutional neural networks," Jurnal Ilmiah Pertanian, vol. 19, no. 1, pp. 1–10, Mar. 2022, doi: 10.31849/jip.v19i1.9556.

[4] M. I. Rosadi, M. Lutfi, and S. Artikel, "Identifikasi Jenis Penyakit Daun Jagung Menggunakan Deep Learning Pre-Trained Model," Jurnal Keilmuan dan Aplikasi Teknik Informatika, pp. 1–8, 2019, doi: 10.35891/explorit.

[5] E. Hutabri, "Penerapan Metode Rapid Application Development (RAD) Dalam Perancangan Media Pembelajaran Multimedia," vol. 1, no. 2, pp. 57–62, 2019.

[6] N. Hidayat and K. Hati, "Penerapan Metode Rapid Application Development (RAD) dalam Rancang Bangun Sistem Informasi Rapor Online (SIRALINE)," Jurnal Sistem Informasi STMIK Antar Bangsa, vol. X, no. 1, pp. 1–10, 2021.

[7] A. Munawir, N. Nugroho, P. Studi, and I. Komputer, "Penerapan Metode Rapid Application Development Pada Sistem Informasi Monitoring Pelanggaran Siswa," 2023.

[8] T. Hamilton, "What is Black Box Testing? Techniques, Types & Example," www.guru99.com. Accessed: Apr. 28, 2024. [Online]. Available: https://www.guru99.com/black-box-testing.html

[9] J. Brooke, "SUS-a quick and dirty usability scale." [Online]. Available: https://www.researchgate.net/publication/319394819

[10] L. Sholihatin and M. Pd, "Pengembangan
Media Pembelajaran Bahasa Arab Berbasis Aplikasi Plotagon Pada Siswa Ma Nu Petung Panceng Gresik," Prosiding Konferensi Nasional Bahasa Arab, pp. 320–326, 2020.

[11] A. Majid, Perencanaan pembelajaran mengembangkan standar kompetensi guru. 2019. [12]A. M. Lukman and O. Rahmanto, "Aplikasi Panduan Pola Hidup Sehat," IJSE-Indonesian Journal on Software Engineering, vol. 6, no. 1, pp. 64–70, 2020.

[13] E. Dunia, "Tumiyang, Pekuncen, Banyumas." Accessed: May 21, 2024. [Online]. Available: https://p2k.stekom.ac.id/ensiklopedia/Tumiyang,_Pekuncen,_Banyumas

[14] M. Nur et al., "Klasifikasi Penyakit Mata Berdasarkan Citra Fundus Menggunakan Yolo V8," 2023.

[15] I. K. S. Buana, "Aplikasi Untuk Pengoprasian Komputer Dengan Mendeteksi Gerakan Menggunakan Opencv Python," 2018.

[16] M. Musihudin and Oktafianto, Analisis dan Perancangan Sistem Informasi Menggunakan Model Terstruktur dan UML. Yogyakarta: Penerbit Andi, 2016.

[17] R. Setiawan, "Apa Itu React Native? Apa Kelebihan dan Kekurangannya?," www.dicoding.com. Accessed: Apr. 28, 2024. [Online]. Available: https://www.dicoding.com/blog/apa-itu-reactnative/

[18]Supriyanta, E. Rahmawati, and M. A. R. Kurnia, "Pengembangan Aplikasi Repository Karya Siswa Dengan Metode Waterfall," Indonesian Journal on Software Engineering (IJSE), vol. 9, no. 2, pp. 92–100, Dec. 2023.

[19]F. N. Hasanah and R. T. Untari, Rekayasa Perangkat Lunak. 2020.