SiPadi : Sistem Informasi Diagnosis Penyakit Padi Menggunakan Forward Chaining Berbasis Client-Side Web

Anas Aqila[1*], Elyza Putrinika Maharani[2], Surya Dwi Agus Novianto[3], Fajar Suryani[4]

[1]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[1*]250103064@mhs.udb.ac.id
(penulis korespondensi)

[2]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[2]250103071@mhs.udb.ac.id

[3]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[3]250103086@mhs.udb.ac.id

[4]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[4]fajar_suryani@udb.ac.id

***Abstrak*— Serangan hama dan penyakit padi kerap menurunkan produktivitas panen, sementara keterbatasan akses tenaga penyuluh menyulitkan petani melakukan diagnosis dini secara mandiri. Penelitian ini merancang SiPadi, sistem informasi diagnosis penyakit padi berbasis web yang dibangun sepenuhnya sebagai aplikasi *front-end* (HTML5, CSS3, Bootstrap 4, *vanilla JavaScript*) tanpa *backend server* maupun basis data relasional. Basis pengetahuan berisi 10 penyakit, 15 gejala, dan 15 aturan disimpan dalam JSON statis dan diproses oleh mesin inferensi *forward chaining* di sisi klien untuk mencocokkan gejala pengguna dengan aturan IF-THEN. Pengujian *black-box*, UAT, dan *equivalence partitioning* pada 7 skenario menunjukkan seluruh fitur berjalan valid, termasuk peringatan otomatis saat gejala tidak mencukupi. SiPadi terbukti ringan dan mudah diakses tanpa instalasi server, meski basis pengetahuannya masih statis sehingga pembaruan memerlukan perubahan kode secara manual.**
***Kata kunci*—** Sistem Informasi, Forward Chaining, Penyakit Tanaman Padi, JavaScript, JSON, Fetch API.

***Abstract*— Pest and disease outbreaks in rice crops often reduce yield, while limited access to agricultural extension workers hinders farmers from performing early, independent diagnosis. This research develops SiPadi, a web-based information system for rice disease diagnosis built entirely as a front-end application (HTML5, CSS3, Bootstrap 4, vanilla JavaScript) without any backend server or relational database. Its knowledge base of 10 diseases, 15 symptoms, and 15 rules is stored as static JSON and processed by a client-side forward chaining inference engine that matches user symptoms against IF-THEN rules. Black-box testing, UAT, and equivalence partitioning across 7 scenarios confirmed all features functioned validly, including automatic warnings when symptoms were insufficient to trigger a diagnosis. SiPadi proved lightweight and accessible without server installation, though its static knowledge base still requires manual code changes for updates.**
***Keywords*—** Information System, Forward Chaining, Rice Plant Disease, JavaScript, JSON, Fetch API.

I. PENDAHULUAN

Sektor pertanian padi di Indonesia terus menunjukkan capaian yang positif sebagai tulang punggung ketahanan pangan nasional. Seiring dengan perkembangan teknologi digital yang semakin masif, adopsi sistem informasi kini telah merambah secara luas ke berbagai sektor[1]. Penggunaan sistem informasi di bidang pertanian, mulai dari pemantauan lahan hingga platform edukasi berbasis web, telah terbukti mempermudah petani dalam mengakses pengetahuan secara cepat dan mandiri guna meningkatkan efisiensi pengelolaan hasil tani.

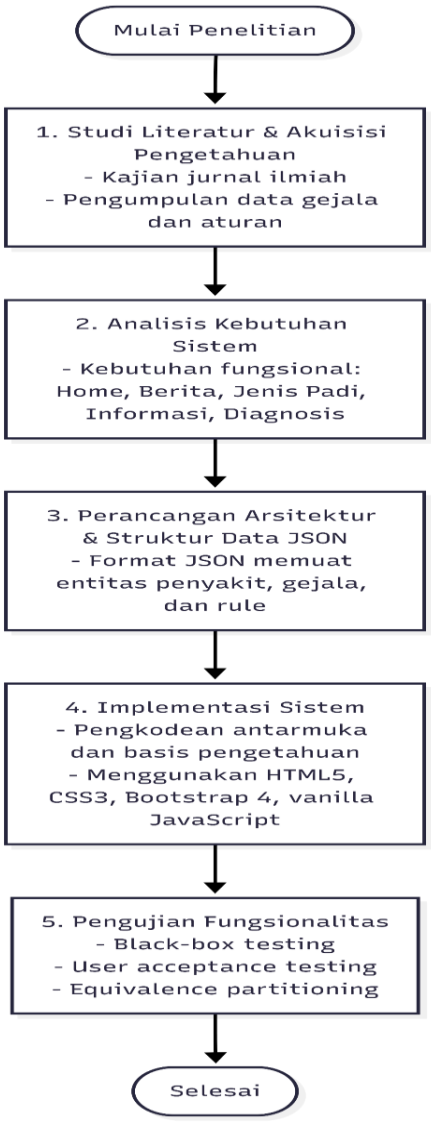
Sebagai negara agraris dengan mayoritas penduduk yang mengkonsumsi beras, menjaga produktivitas tanaman padi (*Oryza sativa L.*) adalah hal yang krusial [2]. Meskipun demikian, masih terdapat berbagai permasalahan teknis di lapangan, khususnya ancaman serangan hama dan penyakit yang dapat menurunkan produktivitas panen secara drastis apabila tidak ditangani dengan tepat waktu [3]. Permasalahan mendasar yang kerap dihadapi petani adalah keterbatasan pengetahuan untuk mengidentifikasi gejala penyakit sejak dini, serta sulitnya akses terhadap tenaga penyuluh pertanian di daerah terpencil [4].

Sebagai solusi atas permasalahan tersebut, diusulkan pemanfaatan sistem informasi diagnostik yang mampu membantu petani mengidentifikasi penyakit secara mandiri dan akurat [4]. Sistem informasi ini dirancang untuk memproses data input berupa gejala-gejala yang terlihat pada tanaman menggunakan metode *forward chaining*, yaitu pendekatan pelacakan ke depan yang mencocokkan fakta-fakta awal (gejala) menuju ke sebuah kesimpulan (diagnosis penyakit) [5].

Berbeda dengan sistem informasi konvensional yang umumnya bergantung pada infrastruktur *server* dan basis data relasional yang berat, penelitian ini berfokus pada pengembangan yang lebih modern dan ringan. Seluruh logika sistem diimplementasikan secara langsung pada sisi klien (*client-side*) menggunakan HTML, CSS, dan JavaScript murni (*vanilla*) dengan basis pengetahuan berformat JSON. Pendekatan ini menjadikan sistem sangat ringan, mudah didistribusikan melalui repositori web, serta dapat diakses kapan saja oleh petani tanpa memerlukan konfigurasi *server* yang kompleks [6]. Tujuan dari penelitian ini adalah merancang dan mengimplementasikan Sistem Informasi Diagnosis Penyakit Padi (SiPadi) berbasis web menggunakan metode *forward chaining*, serta memvalidasi ketepatan logika inferensinya.

II. METODOLOGI PENELITIAN

Penelitian ini menggunakan pendekatan pengembangan *waterfall* yang terdiri atas lima tahapan [6], yaitu: (1) studi literatur dan akuisisi pengetahuan dari pakar pertanian dan referensi ilmiah mengenai penyakit tanaman padi; (2) analisis kebutuhan sistem; (3) perancangan arsitektur sistem dan struktur data JSON sebagai basis pengetahuan; (4) implementasi sistem; serta (5) pengujian fungsionalitas. Gambaran tahapan penelitian dapat dilihat pada Gambar 1.



Gambar 1. Flowchart Tahapan Metode Penelitian

1. Tahap Pertama (Studi Literatur dan Akuisisi Pengetahuan): Mengumpulkan data gejala dan aturan diagnosis dari pakar pertanian serta jurnal ilmiah untuk membentuk basis pengetahuan awal.
2. Tahap Kedua (Analisis Kebutuhan Sistem): Mengidentifikasi kebutuhan fungsional (halaman Home, Berita, Jenis Padi, Informasi, dan Diagnosis) serta kebutuhan non-fungsional berupa aksesibilitas web tanpa instalasi server.

3. Tahap Ketiga (Perancangan Arsitektur Sistem dan Struktur Data JSON): Menetapkan format data JSON yang memuat entitas penyakit, gejala, dan rule (aturan) agar dapat diolah langsung di sisi klien (client-side).
4. Tahap Keempat (Implementasi Sistem): Menerjemahkan rancangan antarmuka (interface) dan basis pengetahuan ke dalam kode pemrograman HTML5, CSS3, Bootstrap 4, dan vanilla JavaScript.
5. Tahap Kelima (Pengujian Fungsionalitas): Memvalidasi seluruh fitur aplikasi web menggunakan pengujian black-box[7],[8].

A. Basis Pengetahuan

Basis pengetahuan SiPadi memuat 10 entitas penyakit padi, 15 gejala, dan 15 aturan (rules) inferensi [9]. Penyusunan daftar gejala dan kesimpulan penyakit mengacu pada hasil akuisisi pengetahuan dari pakar pertanian dan referensi ilmiah terkait hama penyakit tanaman padi **[9]**. Berbeda dengan sistem berbasis bobot, basis pengetahuan pada SiPadi menggunakan pendekatan fakta pasti (ya/tidak) yang dipetakan ke dalam bentuk matriks aturan. Tabel 1 menampilkan daftar gejala, Tabel 2 menunjukkan daftar penyakit, dan Tabel 3 merepresentasikan aturan (rules) basis pengetahuan SiPadi.

Tabel 1. Gejala yang menyebabkan penyakit padi

ID	Gejala
G1	Bercak coklat/hitam pada daun
G2	Bercak berbentuk belah ketupat
G3	Daun menguning
G4	Daun kering seperti terbakar
G5	Tanaman kerdil
G6	Daun kuning-orange
G7	Bercak pada batang atau pelepah
G8	Tanaman mudah rebah
G9	Bulir padi hampa
G10	Pertumbuhan terhambat
G11	Tepi daun bergerigi atau sobek
G12	Tulang daun membengkak
G13	Bulir padi berupa gumpalan spora
G14	Akar membusuk dan berwarna hitam
G15	Tanaman tumbuh memanjang tidak normal

Tabel 2.Penyakit Padi

ID	Penyakit
P1	Penyakit Blas (Blast)
P2	Penyakit Hawar Daun (BLB)
P3	Penyakit Tungro
P4	Penyakit Busuk Pelepah (Sheath Blight)
P5	Penyakit Rumput Kerdil (Grassy Stunt)
P6	Penyakit Kresek (Acute BLB/Bacterial Wilt)
P7	Penyakit Kerdil Hampa (Ragged Stunt)
P8	Penyakit Gosong Palsu (False Smut)
P9	Penyakit Busuk Akar (Root Rot)
P10	Penyakit Bakanae (Bakanae Disease)

Tabel 3. Rules Penentuan Penyakit Padi dengan Gejala

ID	Rules	Kesimpulan Penyakit
R1	Bercak coklat/hitam pada daun dan bercak belah ketupat	Blas (Blast)
R2	Bercak belah ketupat disertai daun kering seperti terbakar	Blas (Blast)
R3	Daun menguning dan mengering seperti terbakar	Hawar Daun Bakteri (BLB)
R4	Daun menguning hingga orange akibat infeksi bakteri	Hawar Daun Bakteri (BLB)
R5	Tanaman kerdil dan daun kuning-orange	Tungro
R6	Bercak pada pelepah dan tanaman rebah	Busuk Pelepah (Sheath Blight)
R7	Tanaman kerdil menyerupai rumput dan bulir hampa	Busuk Pelepah (Sheath Blight)
R8	Tanaman kerdil disertai tepi daun bergerigi atau sobek	Kerdil Hampa (Ragged Stunt)
R9	Bulir padi hampa disertai pembengkakan tulang daun	Kerdil Hampa (Ragged Stunt)
R10	Bulir berubah menjadi gumpalan spora beludru	Gosong Palsu (False Smut)
R11	Daun menguning diiringi akar hitam yang membusuk	Busuk Akar (Root Rot)
R12	Daun menguning pucat dan tanaman tumbuh memanjang abnormal	Bakanae (Bakanae Disease)
R13	Tumbuh tinggi tidak normal dan menghasilkan bulir hampa	Bakanae (Bakanae Disease)
R14	Infeksi akut tanaman muda menyebabkan daun kering terbakar dan layu mendadak	Kresek (Acute BLB/Bacterial Wilt)
R15	Daun menguning lalu kering menggulung disertai hambatan tumbuh drastis	Kresek (Acute BLB/Bacterial Wilt)

Berdasarkan aturan pada basis pengetahuan tersebut, setiap penyakit direpresentasikan oleh kombinasi gejala klinis yang spesifik. Sistem akan memvalidasi kecocokan antara gejala yang

dipilih oleh pengguna di lapangan dengan prasyarat gejala pada setiap *rule* penyakit.

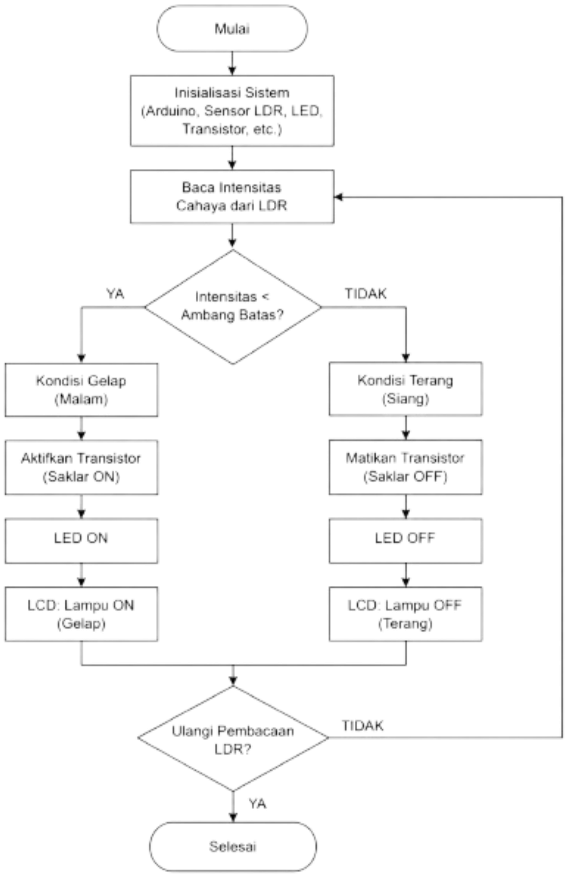
B. Metode Forward Chaining

Metode Forward Chaining digunakan sebagai penggerak utama (mesin inferensi) dalam sistem SiPadi untuk mendiagnosis penyakit. Pendekatan forward chaining merupakan teknik pelacakan beruntun yang memulai penelusuran dari kumpulan fakta yang ada, lalu mencocokkannya dengan aturan-aturan (rules) untuk menarik sebuah kesimpulan akhir [10]. Dalam konteks aplikasi ini, fakta yang dimaksud adalah daftar gejala klinis pada tanaman padi yang dipilih dan diinputkan oleh pengguna.

Secara teknis, basis pengetahuan direpresentasikan dalam bentuk aturan logika bersyarat (IF-THEN). Bagian kondisi atau "IF" (anteseden) berisi kombinasi gejala-gejala spesifik yang menjadi syarat, sedangkan bagian kesimpulan atau "THEN" (konklusi) berisi jenis penyakit padi yang sesuai dengan kombinasi tersebut. Seluruh aturan ini disimpan secara terstruktur di dalam berkas JSON agar mudah dibaca oleh sistem [10].

Ketika pengguna memilih beberapa gejala melalui form di halaman web, sistem akan mencatatnya sebagai sekumpulan fakta awal. Mesin inferensi pada JavaScript kemudian akan melakukan proses iterasi atau perulangan. Sistem akan membandingkan fakta awal dari pengguna dengan bagian "IF" pada setiap aturan di dalam JSON. Apabila semua gejala yang dipersyaratkan oleh sebuah aturan terpenuhi secara persis oleh pilihan pengguna, maka aturan tersebut bernilai benar (aktif). Hasilnya, sistem akan mengeksekusi bagian "THEN" dari aturan tersebut dan menetapkannya sebagai hasil diagnosis final.

Alur kerja dari proses pelacakan algoritma ini, mulai dari penerimaan input gejala, pencocokan aturan, hingga penarikan kesimpulan penyakit, dapat dilihat secara rinci pada Gambar 2.



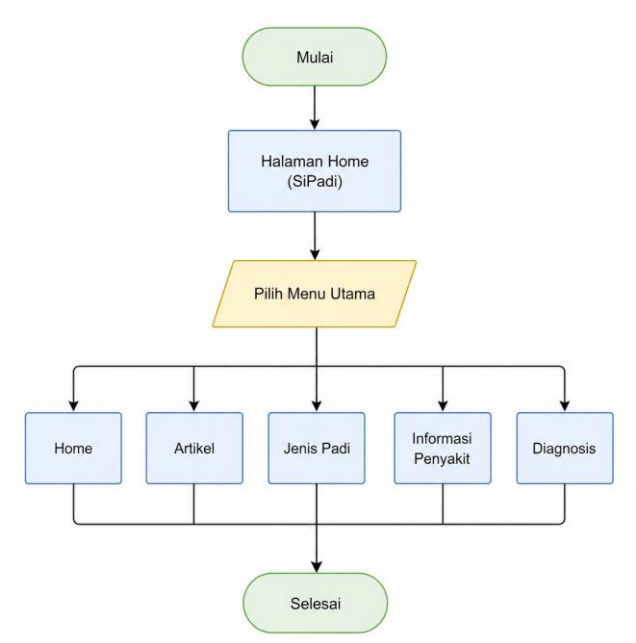
Gambar 2. Flowchart Algoritma Forward Chaining Sistem SiPadi

Berdasarkan Gambar 2, hasil diagnosis yang ditemukan oleh algoritma akan langsung direnderer ke antarmuka pengguna beserta panduan atau rekomendasi penanganannya. Sebaliknya, jika kombinasi gejala yang diinputkan oleh pengguna tidak memenuhi satu pun aturan yang ada di dalam sistem, aplikasi dirancang untuk memberikan pesan peringatan bahwa penyakit tidak teridentifikasi atau input gejala tidak mencukupi untuk menarik kesimpulan.

C. Alur Sistem

Berbeda dengan penelitian sejenis sebelumnya yang umumnya masih bergantung pada arsitektur client-server menggunakan PHP dan basis data relasional MySQL [2], SiPadi diimplementasikan sepenuhnya pada sisi klien (client-side) tanpa memerlukan backend server [6]. Seluruh basis pengetahuan disimpan secara efisien dalam berkas berformat statis (data.json) yang dipanggil secara asinkron menggunakan Fetch API.

Pendekatan arsitektur ini memungkinkan sistem untuk dapat berjalan hanya dengan membuka berkas HTML secara langsung pada peramban (browser). Pengguna maupun petani tidak memerlukan instalasi server lokal (seperti XAMPP) maupun konfigurasi basis data apa pun, sehingga sangat ringan dan mudah didistribusikan. Gambar 3 menunjukkan pemodelan flowchart (yang merepresentasikan alur HIPO) sistem SiPadi secara keseluruhan, mulai dari alur pengguna saat pertama kali mengakses website, melakukan navigasi menu, hingga mengeksekusi proses diagnosis.



Gambar 3. Flowchart Sistem SiPadi

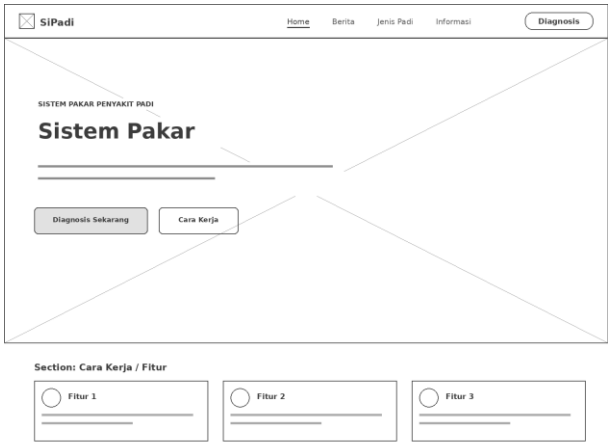
III. HASIL DAN PEMBAHASAN

A. *Desain Interface Sistem*

Sebelum tahapan desain interface, dilakukan perancangan wireframe untuk setiap halaman pada sistem SiPadi guna menentukan tata letak (*layout*), elemen antar muka dan alur navigasi pengguna. Wireframe dirancang dalam bentuk *low-fidelity* yang berfokus pada struktur konten tanpa detail elemen visual. Terdapat 5 wireframe utama yang mewakili halaman Home, Artikel, Jenis Padi, Informasi, dan Diagnosis, sebagaimana ditunjukkan pada Gambar 4 - 8.

1. Wireframe Halaman Home

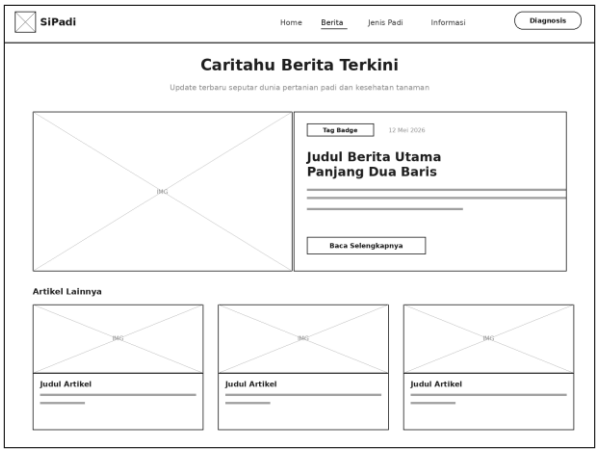
   Wireframe pada Gambar 4 menampilkan struktur navbar dengan 4 menu navigasi (Home, Berita, Jenis Padi, Informasi) beserta tombol akses cepat Diagnosis, bagian *hero section* berisi judul Sistem Pakar dan dua tombol aksi (Diagnosis Sekarang dan Cara Kerja), serta *section* Cara Kerja/Fitur berupa 3 kartu fitur di bagian bawah.



Gambar 4. Wireframe Halaman Home

2. Wireframe Halaman Berita

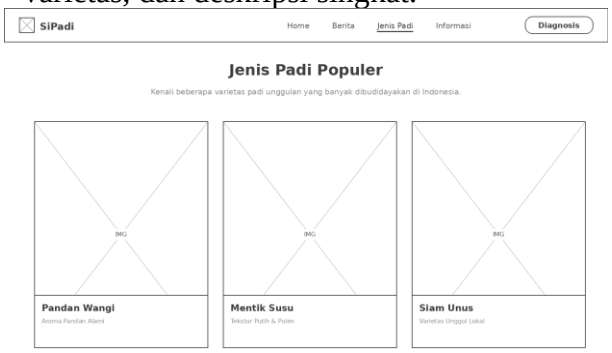
   Wireframe pada Gambar 5 dirancang dengan satu artikel utama berukuran besar di sisi kiri berdampingan dengan tag kategori, tanggal, judul, ringkasan, dan tombol Baca Selengkapnya di sisi kanan, dilengkapi beberapa grid kartu Artikel Lainnya pada bagian bawah halaman.



Gambar 5. Wireframe Halaman Berita

3. Wireframe Halaman Jenis Padi

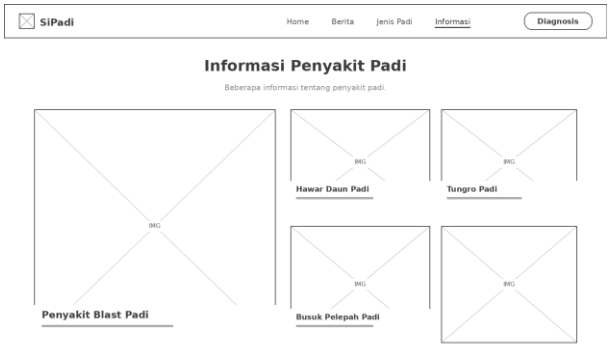
Wireframe pada Gambar 6 menampilkan kartu varietas padi (diantaranya ada : Pandan Wangi, Mentik Susu, Siam Unus, dan masih banyak lagi) yang tersusun sejajar, masing-masing memuat *placeholder* gambar, nama varietas, dan deskripsi singkat.



Gambar 6. Wireframe Halaman Jenis Padi

4. Wireframe Halaman Penyakit Padi

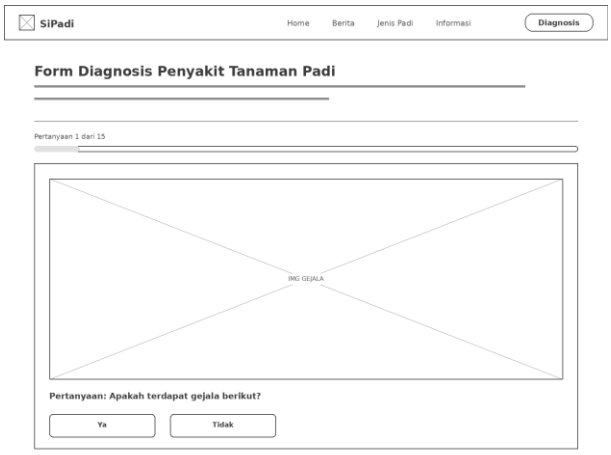
Wireframe pada Gambar 7 menggunakan pola *grid asimetris* dengan satu kartu berukuran besar (Penyakit Blast Padi) di sisi kiri dan beberapa kartu berukuran lebih kecil di sisi kanan yang mewakili penyakit padi, yaitu : Hawar Daun Bakteri, Tungro, dan Busuk Pelepah.



Gambar 7. Wireframe Halaman Informasi

5. Wireframe Halaman Diagnosis

Wireframe pada Gambar 8 merancang form pertanyaan bergaya wizard satu-per-satu, terdiri atas indikator progres berupa teks Pertanyaan 1 dari 15 dan progres bar, area gambar gejala, teks pertanyaan, serta dua tombol pilihan jawaban Ya dan Tidak.



Gambar 8. Wireframe Halaman Diagnosis

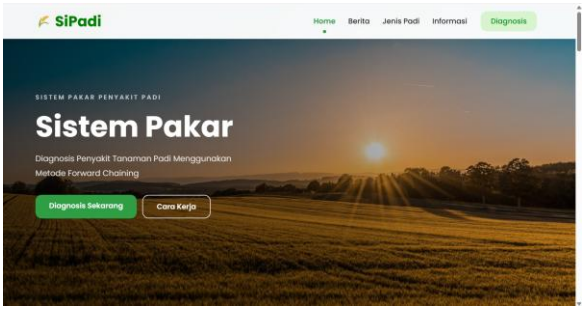
B. Implementasi Halaman

Rancangan wireframe pada Gambar 4-8 selanjutnya diimplementasikan menjadi halaman web yang fungsional menggunakan HTML5, CSS3, Bootstrap 4, dan vanilla JavaScript. Hasil implementasi dapat dilihat pada Gambar 9-13.

1. Halaman Home

Implementasi pada Gambar 9 menghadirkan identitas visual hijau yang merepresentasikan sektor pertanian, dengan *hero section* berupa foto sawah padi saat

matahari terbenam dan judul *Sistem Pakar Diagnosis Penyakit Menggunakan Metode Forward Chaining*, sesuai dengan struktur yang telah dirancang pada wireframe Gambar 4.



Gambar 9. Implementasi Halaman Home

2. Halaman Berita

Implementasi pada Gambar 10 menampilkan artikel utama lengkap dengan label kategori, tanggal publikasi, foto berita, ringkasan berita, dan tombol Baca Selengkapnya, sejajar dengan tata letak pada wireframe Gambar 5. tidak hanya itu di bawah artikel utama terdapat beberapa grid kartu Artikel Lainnya pada bagian bawah halaman.

Gambar 10. Implementasi Halaman Berita

3. Halaman Jenis Padi

Implementasi pada Gambar 11 menampilkan foto asli tiga varietas unggulan yaitu Pandan Wangi, Mentik Susu, dan Siam Unus, beserta ciri khas masing-masing varietas, juga dilengkapi dengan grid kartu jenis-jenis Padi yang lainnya pada bagian bawah halaman, sesuai dengan rancangan kartu pada wireframe Gambar 6.

Gambar 11. Implementasi Halaman Jenis Padi

4. Halaman Penyakit Padi

Implementasi pada Gambar 12 menyajikan katalog penyakit padi meliputi Blast, Hawar Daun Bakteri, Busuk Pelepah, dan masih banyak lagi dalam bentuk kartu bergambar disertai deskripsi singkat penyebab masing-masing penyakit, konsisten dengan pola grid pada wireframe Gambar 7.

Gambar 12. Implementasi Halaman Penyakit Padi

5. Halaman Diagnosis

Implementasi pada Gambar 13 merealisasikan wireframe form diagnosis dengan menambahkan foto gejala nyata pada tanaman padi [9], indikator Pertanyaan 1 dari 15, serta deskripsi instruksi penggunaan

sistem, sebagaimana dirancang pada wireframe Gambar 8.

Gambar 13. Implementasi Halaman Diagnosis

Perbandingan antara wireframe pada Gambar 4 hingga Gambar 8 dengan hasil implementasi pada Gambar 9 hingga Gambar 13 menunjukkan bahwa seluruh elemen struktural yang dirancang pada tahap *wireframing* berhasil direalisasikan secara konsisten pada antarmuka akhir dengan penambahan elemen visual seperti fotografi asli, warna identitas hijau, dan efek interaktif untuk meningkatkan pengalaman pengguna (*user experience*).

C. Hasil Pengujian

Pengujian fungsionalitas sistem dilakukan secara komprehensif menggunakan metode *black-box testing* **[7]**, **[8]** untuk memverifikasi keseluruhan fungsionalitas dan keandalan sistem SiPadi. Tabel 5 menyajikan rekapitulasi hasil pengujian fungsionalitas sistem.

Tabel 5. Hasil Pengujian Fungsionalitas Sistem SiPadi

No	Fitur/Komponen Utama	Prosedur Pengujian	Output yang Diharapkan	Status (Valid/ Tidak)
1	Halaman Home	Klik menu Home pada navbar	Sistem menampilkan halaman beranda dan menu navigasi utama dengan responsif	Valid
2	Fitur Informasi & Berita	Klik menu Berita, Jenis Padi, dan Informasi	Konten artikel mengenai varietas padi dan informasi penyakit terbuka secara utuh	Valid
3	Pengambilan Basis Pengetahuan	Buka halaman Diagnosis	Fungsi Fetch API berhasil memuat data gejala dan aturan dari berkas JSON lokal ke dalam antarmuka	Valid
4	Interaksi Pilihan Gejala	Pilih beberapa checkbox gejala penyakit (G1-G15)	Antarmuka merespons pilihan pengguna secara *real-time* dan menyimpan array gejala terpilih	Valid
5	Mesin Inferensi (Forward Chaining)	Menyelesaikan seluruh (15) pertanyaan gejala hingga akhir tanpa mengeklik tombol submit apapun	Sistem secara otomatis mengolah aturan (*rules*) dan langsung menampilkan hasil keputusan penyakit yang tepat	Valid
6	Penanganan Diagnosis Nihil	Menjawab "TIDAK" pada seluruh pertanyaan sehingga tidak ada kombinasi aturan yang terpenuhi	Sistem menampilkan pesan peringatan (*alert warning*) bahwa penyakit tidak terdiagnosis dan meminta pengguna memeriksa gejala lebih teliti	Valid
7	Tombol Reset/Ulang	Klik tombol "Diagnosis Ulang" setelah proses diagnosis selesai	Sistem membersihkan seluruh checkbox terpilih dam mengembalikan tampilan antarmuka ke kondisi awal	Valid

Berdasarkan Tabel 5, seluruh 7 skenario pengujian menghasilkan output dengan status valid, yang mengindikasikan bahwa kinerja sistem telah memenuhi spesifikasi kebutuhan. Penerapan pengujian *equivalence partitioning* difokuskan pada evaluasi *black-box* terhadap mesin *forward chaining*. Hasilnya membuktikan bahwa algoritma secara konsisten menghasilkan diagnosis yang presisi saat diberikan kombinasi gejala penuh (kelas valid). Sistem juga terbukti tangguh (*robust*) dalam menangani kasus input parsial atau tidak lengkap (kelas tidak valid), di mana sistem secara otomatis memunculkan peringatan apabila gejala yang dipilih pengguna tidak mencukupi untuk mengaktifkan satu pun *rule* pada basis pengetahuan. Sementara itu, pengujian UAT memvalidasi bahwa antarmuka pengguna dapat beroperasi secara optimal, responsif terhadap berbagai ukuran layar (*viewport*), dan seluruh tata letak visual berfungsi dengan baik.

Keunggulan utama sistem informasi SiPadi dibandingkan sistem diagnosis padi konvensional berbasis PHP-MySQL adalah ketiadaan ketergantungan pada infrastruktur *server*. Hal ini menurunkan hambatan teknis secara signifikan, memungkinkan distribusi melalui *hosting* statis atau media penyimpanan

lokal, dan menekan biaya operasional sistem. Penggunaan JSON sebagai basis pengetahuan juga memudahkan pembaruan data secara asinkron oleh pakar pertanian di masa mendatang tanpa memerlukan pemahaman terkait bahasa kueri basis data relasional.

IV. KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan, prototipe Sistem Informasi Diagnosis Penyakit Padi (SiPadi) berhasil dirancang dan diimplementasikan sepenuhnya sebagai aplikasi front-end murni menggunakan teknologi sisi klien (client-side) meliputi HTML5, CSS3, dan vanilla JavaScript. Desain antarmuka dibuat responsif dan ramah pengguna agar memudahkan navigasi petani dalam memilih gejala penyakit. Tanpa ketergantungan pada backend server maupun basis data relasional, mesin inferensi forward chaining dibangun langsung di dalam logika front-end dengan memanfaatkan berkas JSON statis yang diakses secara asinkron sebagai penyimpanan basis pengetahuan. Hasil pengujian black-box memvalidasi bahwa seluruh komponen interaksi antarmuka dan visualisasi alur diagnosis berfungsi dengan valid pada setiap skenario uji.

Meskipun memiliki keunggulan dari segi performa front-end yang sangat ringan, cepat, dan dapat diakses langsung via peramban (browser) tanpa instalasi server, prototipe ini masih memiliki keterbatasan. Kekurangan utama sistem terletak pada arsitektur penyimpanan data yang bersifat statis. Seluruh data gejala dan aturan penyakit tertanam langsung di dalam berkas JSON lokal. Akibatnya, pembaruan konten atau penambahan aturan diagnosis baru belum dapat dilakukan secara dinamis melalui halaman dasbor antarmuka web, melainkan masih harus dilakukan dengan mengubah kode sumber (source code) front-end secara manual.

Untuk penelitian selanjutnya, disarankan agar prototipe sistem ini dikembangkan lebih lanjut menjadi aplikasi web dinamis dengan mengintegrasikan sistem basis data relasional (seperti MySQL) atau NoSQL, serta membangun modul panel admin (dashboard) khusus pada sisi backend. Langkah ini penting agar pakar pertanian dapat memperbarui, menambah, atau memvalidasi basis pengetahuan (data gejala dan aturan penyakit) secara langsung melalui antarmuka visual tanpa harus mengubah kode sumber (source code) front-end secara manual, sehingga sistem menjadi lebih adaptif dan mudah digunakan dalam jangka panjang. Kedua penerapan algoritma *Long Short-Term Memory* (LSTM) untuk mendukung analisis penyakit tanaman padi berbasis pengolahan citra **[11]**, dan ketiga dapat mengintegrasikan perhitungan metode *Certainty Factor* untuk mengkuantifikasi persentase tingkat keyakinan diagnosis secara spesifik **[12]**.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada seluruh anggota tim pengembang SiPadi atas kolaborasi dan dedikasi dalam proyek ini, serta kepada Program Studi Teknik Informatika Universitas Duta Bangsa Surakarta atas dukungan akademik yang diberikan selama proses penelitian dan pengembangan sistem.

REFERENSI

[1] M. Z. Pipiano, Safuan, dan A. Fathurrohman, "Sistem pakar diagnosis penyakit tanaman padi menggunakan metode forward chaining berbasis web," Jurnal Komputer dan Teknologi Informasi, vol. 2, no. 1, hlm. 13–25, 2024.

[2] M. H. Alle, R. Ansar, H. K. Sirajuddin, dan Muharto, "Sistem pakar pendeteksi penyakit pada tanaman padi menggunakan metode forward chaining berbasis web," Indonesian Journal on Information Systems, vol. 6, no. 1, 2021, doi: 10.36549/ijis.v6i1.133.

[3] N. A. Muslimah, R. Y. Bakti, dan T. Wahyuni, "Sistem pakar mendiagnosa penyakit pada tanaman padi menggunakan metode forward chaining," Aurus Jurnal Sains dan Teknologi, vol. 2, no. 2, 2023, doi: 10.57250/ajst.v2i2.657.

[4] S. Sholikhah, D. Kurniadi, dan A. Riansyah, "Sistem pakar menggunakan metode forward chaining untuk diagnosa hama dan penyakit tanaman padi," Sultan Agung Fundamental Research Journal, vol. 2, no. 2, hlm. 103–110, 2021.

[5] S. Holifah, A. M. Saputra, R. T. Saputra, dan U. M. Nurali, "Aplikasi sistem pakar untuk diagnosa penyakit pada padi dengan metode forward chaining," Jurnal Informatika dan Rekayasa Perangkat Lunak, vol. 6, no. 3, hlm. 570–574, 2021.

[6] F. Ramadhan, M. A. Maulana, dan D. Suhendro, "Rancang bangun sistem informasi pertanian berbasis web menggunakan JavaScript dan JSON untuk mendukung ketahanan pangan," Jurnal Teknologi dan Manajemen Informatika, vol. 9, no. 2, hlm. 112–121, 2023.

[7] F. C. Hudi dan C. M. Karyanti, "Pengujian fungsionalitas sistem informasi berbasis web menggunakan metode user acceptance testing," Jurnal Ilmiah Komputasi, vol. 22, no. 4, hlm. 553–560, 2024, doi: 10.32409/jikstik.22.4.3490.

[8] M. Zen, I. Irwan, H. Hafni, dan M. D. P. Ananda, "Implementasi dan pengujian equivalence partitioning pada sistem informasi

berbasis web," Bulletin of Computer Science Research, vol. 4, no. 4, hlm. 327–340, 2024.

[9] G. S. Nasution, "Sistem pakar dalam mendiagnosis hama blas dan kresek pada tanaman padi menggunakan metode forward chaining," Jurnal Sistem Informasi dan Teknologi, vol. 4, no. 4, 2024, doi: 10.37034/jsisfotek.v4i4.144.

[10] Haerudin, Iqbaludin, F. I. Noer, dan P. Rosyani, "Implementasi metode forward chaining dalam sistem pakar berbasis web," OKTAL Jurnal Ilmu Komputer dan Science, vol. 2, no. 6, hlm. 1681–1687, 2023.

[11] A. Wiranda dan M. Sadikin, "Penerapan long short-term memory untuk analisis penyakit tanaman padi," Jurnal Teknologi Informasi dan Ilmu Komputer, vol. 9, no. 1, hlm. 55–64, 2022, doi: 10.25126/jtiik.2022.9112938.

[12] R. Fauzan, Y. Umaidah, dan A. A. Fikri, "Pengembangan sistem pakar diagnosa penyakit tanaman padi menggunakan metode certainty factor dan forward chaining," Jurnal Informatika, vol. 10, no. 1, hlm. 1–8, 2023