Jurnal Rekayasa Sistem Informasi dan Teknologi
Volume 3, No 4 – Mei 2026
e-ISSN : 3025-888X

IMPLEMENTASI *PROGRESSIVE WEB APP* (PWA) UNTUK MONITORING KUALITAS NUTRISI HIDROPONIK BERBASIS IOT

M. Rizki Rianto[1*], La Ode Amril[2], Susi Maulidiah[3]
[1,2,3]Universitas Djuanda
mrizkirianto116@gmail.com[1*], laodeamril@unida.ac.id[2], susi.maulidiah@unida.ac.id[3]

Received: 20-04-2026	Revised: 14-05-2026	Approved: 27-05-2026

ABSTRAK

Penelitian ini bertujuan untuk mengimplementasikan Progressive Web App (PWA) pada sistem monitoring kualitas nutrisi hidroponik berbasis Internet of Things (IoT) guna meningkatkan efisiensi pemantauan kondisi tanaman secara real-time dan mendukung pengambilan keputusan yang cepat berdasarkan data aktual. Metode penelitian yang digunakan adalah metode Waterfall yang meliputi tahapan analisis kebutuhan, perancangan arsitektur sistem, implementasi, dan pengujian. Sistem dikembangkan menggunakan mikrokontroler ESP32 sebagai perangkat IoT, Node.js dengan framework HAPI sebagai backend, MongoDB sebagai basis data, serta teknologi Progressive Web App yang memanfaatkan service worker dan web app manifest. Hasil penelitian menunjukkan bahwa sistem berhasil menampilkan data pH, suhu, dan Total Dissolved Solids (TDS) secara real-time melalui dashboard yang responsif dan dapat diinstal pada perangkat pengguna melalui fitur Add to Home Screen (A2HS). Sistem juga mampu mengelola data historis dalam jumlah besar melalui mekanisme paginasi yang ringan, menyediakan fitur ekspor laporan PDF, serta mengirimkan notifikasi peringatan dini melalui email ketika nilai nutrisi berada di luar batas toleransi yang telah ditentukan. Berdasarkan hasil pengujian menggunakan metode Black Box Testing, seluruh fitur utama sistem yang meliputi autentikasi pengguna, monitoring real-time, riwayat data, unduh laporan PDF, dan notifikasi peringatan dini memperoleh status valid. Simpulan penelitian ini adalah implementasi PWA pada sistem monitoring kualitas nutrisi hidroponik berbasis IoT berhasil menghasilkan aplikasi yang efektif, responsif, andal, dan mudah diakses, sehingga layak digunakan untuk mendukung pengelolaan nutrisi hidroponik secara berkelanjutan.

Kata Kunci: *Progressive Web App (PWA), Internet of Things (IoT), Monitoring Nutrisi, Black Box Testing*

PENDAHULUAN

Pesatnya perkembangan teknologi informasi di era digital telah membawa transformasi signifikan pada berbagai sektor kehidupan melalui inovasi *Internet of Things* (IoT). Sebagai konsep yang menghubungkan perangkat fisik ke jaringan internet, IoT memungkinkan pertukaran data, pemantauan kondisi, serta kontrol perangkat secara otomatis maupun manual. Kemampuannya dalam mengumpulkan dan menyampaikan data secara *real-time* memberikan dampak luas, mulai dari bidang industri, kesehatan, hingga optimalisasi pada sektor pertanian [1] . Keberhasilan sistem pertanian hidroponik sangat bergantung pada stabilitas kondisi lingkungan, terutama pada pH air, suhu, dan kadar nutrisi yang diberikan pada tanaman [2].

Pada praktik konvensional, pemantauan kondisi lingkungan hidroponik sering dilakukan secara manual. Pemantauan manual ini membutuhkan banyak waktu dan tenaga, yang dapat menyebabkan efisiensi kerja menurun serta berdampak negatif pada kualitas hasil panen [3]. Selain itu, keterbatasan akses fisik pada sistem manual menghalangi kemampuan pengguna untuk melakukan monitoring secara langsung dan kontrol jarak jauh, sehingga diperlukan sistem berbasis teknologi yang menawarkan kemudahan serta akurasi tinggi [4]. Permasalahan ini nyata terjadi pada Kelompok Wanita Tani (KWT) Mekarwangi di Bogor. Tingginya mobilitas anggota di luar area menyebabkan kegiatan monitoring seringkali terlewat, sehingga parameter krusial seperti nutrisi tidak terpantau secara konsisten dan pertumbuhan tanaman menjadi kurang optimal. Integrasi *Internet of Things* (IoT) menjadi solusi untuk menjembatani ranah fisik dan digital. Dengan memanfaatkan sensor suhu, pH, dan *Total Dissolved Solids* (TDS), data kondisi lingkungan dapat dikumpulkan secara *real-*

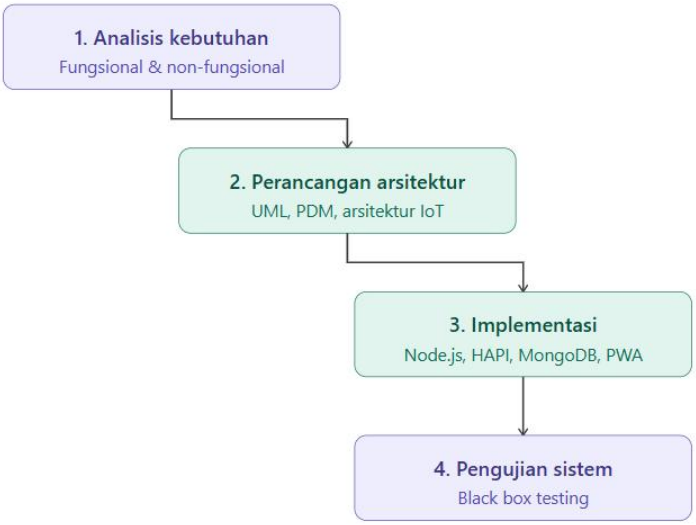
time [2]. Meskipun berbagai platform IoT komersial telah tersedia untuk memfasilitasi hal ini, sebagian besar masih bersifat berbayar dan memiliki keterbatasan fleksibilitas bagi pengguna skala kecil [5] . Oleh karena itu, pengembangan sistem menggunakan teknologi sumber terbuka (*open-source*) seperti bahasa pemrograman JavaScript, *framework backend* HAPI, dan basis data NoSQL MongoDB menjadi alternatif yang efisien dan memiliki skalabilitas tinggi untuk menangani data sensor dalam jumlah besar. Untuk memastikan inovasi teknologi ini dibangun secara sistematis, penggunaan metode Waterfall sangat relevan karena menekankan pada perencanaan yang matang dan dokumentasi yang rapi sejak awal pengembangan [6].

Selain aspek sensor, perkembangan teknologi web memegang peran vital dalam mendukung aksesibilitas informasi. Web berfungsi sebagai media kontrol yang fleksibel, memungkinkan pengguna mengakses data dari berbagai perangkat tanpa batasan waktu dan lokasi. Integrasi antara IoT dan web menjadi solusi efektif untuk menciptakan sistem yang responsif, di mana pengguna dapat memantau perangkat secara langsung melalui antarmuka yang interaktif [7] . Namun, antarmuka web konvensional seringkali menghadapi kendala performa dan aksesibilitas saat digunakan di area pertanian dengan kondisi jaringan internet yang tidak stabil. Untuk mengatasi kelemahan terkait aksesibilitas dan stabilitas jaringan tersebut, penelitian ini mengusulkan implementasi *Progressive Web App* (PWA) pada antarmuka sistem pemantauan. PWA sendiri merupakan teknologi yang dapat menggabungkan fitur dari *mobile native* dengan aplikasi web dengan memanfaatkan teknologi *service workers*, *web app manifests* dan *responsive design* [8] . Konsep PWA diterapkan untuk meningkatkan pengalaman pengguna dengan memungkinkan aplikasi diakses melalui layar utama perangkat layaknya aplikasi *native*, namun tetap ringan dan stabil meskipun kondisi jaringan kurang optimal [9] . PWA juga menawarkan efisiensi konsumsi ruang penyimpanan yang signifikan; berbeda dengan aplikasi *native* yang memerlukan instalasi besar, PWA mampu beroperasi dengan ukuran *cache* yang minimal namun tetap memberikan performa yang responsif, sehingga sangat ideal untuk pengguna dengan keterbatasan spesifikasi perangkat keras [10].

Penelitian ini dibatasi pada perancangan antarmuka web untuk visualisasi data pemantauan lingkungan hidroponik tanpa menyertakan modul aktuator untuk kontrol fisik jarak jauh. Adapun tujuan dari penelitian ini adalah menghasilkan aplikasi berbasis PWA yang mampu menyajikan kondisi hidroponik secara *real-time*, sehingga meningkatkan efisiensi proses pemantauan dan mendukung pengambilan keputusan yang cepat bagi pengelola berdasarkan data aktual.

METODE PENELITIAN

Pengembangan sistem *monitoring* kualitas nutrisi hidroponik ini dilakukan menggunakan metode *Waterfall*, yang berfokus pada pendekatan sekuensial mulai dari analisis kebutuhan, perancangan arsitektur, implementasi (*code generation*), hingga pengujian sistem. Berdasarkan [11] , model *Waterfall* bekerja secara berurutan. Metode ini menekankan pada perencanaan yang matang dan dokumentasi yang rapi sejak awal pengembangan. Namun, inti dari penelitian ini bertumpu pada perancangan arsitektur sistem berbasis *Internet of Things* (IoT) dan integrasinya dengan aplikasi web.



Gambar 1. Tahapan Metode Penelitian

1) Analisis Kebutuhan
   Tahap analisis kebutuhan berfungsi untuk mengidentifikasi seluruh kebutuhan sistem dan pengguna di awal pengembangan [12] . Kebutuhan sistem pada penelitian ini dikategorikan menjadi dua bagian, yaitu kebutuhan fungsional yang berfokus pada fitur operasional dan kebutuhan non-fungsional yang berfokus pada kualitas sistem.
2) Perancangan Arsitektur
   Pada tahap ini, kebutuhan sistem yang telah dikumpulkan diterjemahkan menjadi rancangan teknis, seperti struktur database, alur sistem, serta arsitektur aplikasi. Tujuan dari tahap perancangan adalah memberikan gambaran yang jelas mengenai bagaimana sistem akan dibangun sehingga proses pengembangan dapat berjalan lebih terarah dan terstruktur [13]. Pada pengembangan ini, perancangan dilakukan pada arsitektur menggunakan UML. *Unified Modeling Language* (UML) adalah bahasa pemodelan yang disajikan dalam bentuk diagram atau visual grafis untuk membantu menggambarkan, merancang, serta mendokumentasikan proses pengembangan sistem berbasis objek (*object oriented*) [14] . Jenis UML yang digunakan pada pengembangna ini adalah *Sequence Diagram* dan *Component Diagram*. Selain itu, perancangan struktur penyimpanan dilakukan pada basis data NoSQL dalam bentuk *Physical Data Model* (PDM).

Secara struktural, sistem ini dibangun menggunakan arsitektur *Client-Server* yang memisahkan alur komunikasi data ke dalam tiga lapisan utama:

1) Perangkat IoT (*Node Sensor*)
   Mikrokontroler ESP32 membaca data lingkungan (suhu, pH, TDS) dan bertindak sebagai *client* yang mengirimkan data tersebut secara periodik ke *server backend* melalui permintaan (*request*) API menggunakan protokol HTTP metode POST.
2) *Server Backend* & Basis Data
   Dibangun menggunakan lingkungan Node.js dengan *framework* HAPI. *Backend* bertugas memvalidasi *secret key* dari IoT, mengevaluasi data untuk sistem peringatan, dan menyimpannya secara persisten ke dalam basis data NoSQL MongoDB.
3) *Frontend* (PWA)

Antarmuka web menerapkan pola arsitektur *Model-View-Presenter* (MVP). *Frontend* akan melakukan *request* data secara asinkron (*polling*) ke *server* menggunakan HTTP metode GET untuk memutakhirkan visualisasi data di layar pengguna secara *real-time*.

Tahap implementasi merupakan proses tranformasi atau penerjemahan rancangan yang telah dibuat kedalam bentuk kode sistem [15]. Pada penelitian ini, setelah dilakukan proses perancangan, selanjutnya rancangan tersebut diimplementasikan kedalam kode atau program menggunakan Node.JS, HAPI, MongoDB dan JavaScript untuk membangun web. Implementasi ini juga mencakup penggunaan teknologi PWA seperti *service worker* untuk kapabilitas *offline* dan *caching* serta *web app manifest* untuk fitur instalasi atau menerapkan *homescreen.*

Tahapan pengujian sistem berfungsi untuk memastikan bahwa sistem telah berjalan baik dan sesuai dengan kebutuhan sehingga dapat dilakukan perbaikan jika terdapat kesalahan atau galat [16]. Pada penelitian ini, metode uji yang digunakan adalah *Black Box Testing*, dimana pengujiannya hanya berfokus terhadap fungsional sistem tanpa perlu memeriksa internal kode program [17].

HASIL PENELITIAN DAN PEMBAHASAN

Analisis Kebutuhan

Kebutuhan dalam mengimplementasikan sistem didapat melalui observasi serta wawancara diskusi dengan ketua Kelompok Wanita Tani. Hasil kebutuhan terbagi menjadi dua jenis, yaitu Fungsional dan Non-Fungsional. Rincian hasil kebutuhan terdapat pada tabel berikut.

Tabel 1.
Hasil Analisis Kebutuhan

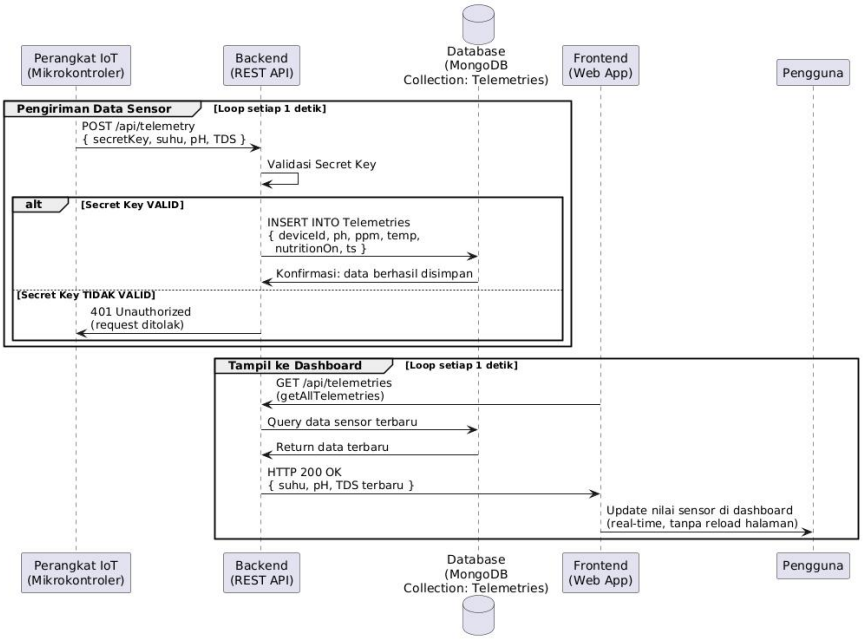
Kategori	Nama Fitur / Kebutuhan	Deskripsi Singkat
Fungsional	Monitoring Real-time	Visualisasi data sensor pH, suhu, dan TDS dari perangkat IoT secara berkala.
	Manajemen Tanaman	Pemilihan jenis tanaman dan pengaturan rentang ppm yang sesuai.
	Riwayat Data	Penyajian data historis sensor dalam bentuk tabel dengan sistem paginasi.
	Early Warning System	Notifikasi otomatis via *email* jika kadar TDS melewati batas toleransi ± 200 ppm.
	Manajemen Data	Fitur untuk mengunduh arsip data sensor dan penghapusan data secara manual.
Non-Fungsional	Keamanan (Autentikasi)	Mekanisme *login* untuk membatasi akses *dashboard* hanya bagi pengguna sah.
	Implementasi PWA	Penggunaan *Progressive Web App* untuk akses cepat dan instalasi pada layar utama.
	Secret Key	Keamanan integrasi API antara perangkat IoT dan *backend server*.

Perancangan Arsitektur

Sebelum tahapan implementasi kode program dilakukan, struktur arsitektur dan alur komunikasi data dimodelkan terlebih dahulu untuk memastikan integrasi yang presisi antara perangkat IoT, *server backend*, dan aplikasi klien. Pemodelan logika perangkat lunak divisualisasikan menggunakan *Unified Modeling Language* (UML), sementara perancangan media penyimpanan persisten direpresentasikan melalui *Physical Data Model* (PDM).

a. *Sequence Diagram*

Sequence Diagram digunakan untuk memvisualisasikan interaksi dinamis dan pertukaran pesan antar entitas sistem berdasarkan urutan waktu [18]. Diagram ini secara khusus mendeskripsikan alur pemantauan (*monitoring*) data sensor secara terus-menerus dari perangkat keras hingga divisualisasikan ke layar pengguna.



Gambar 2 *Sequence Diagram Monitoring* Sistem

Berdasarkan Gambar 1, alur komunikasi sistem terbagi menjadi dua tahapan utama yang masing-masing berjalan secara iteratif dalam siklus (*loop*) setiap satu detik:

1) Siklus Pengiriman Data Sensor

Mikrokontroler (Perangkat IoT) secara periodik mengirimkan hasil pembacaan parameter lingkungan (suhu, pH, TDS) menuju *Backend* menggunakan metode POST ke *endpoint* /api/telemetry. *Backend* akan melakukan validasi *Secret Key*. Apabila valid, data akan disisipkan (*insert*) ke dalam koleksi *Telemetries* di basis data MongoDB, dan server memberikan konfirmasi keberhasilan penyimpanan.

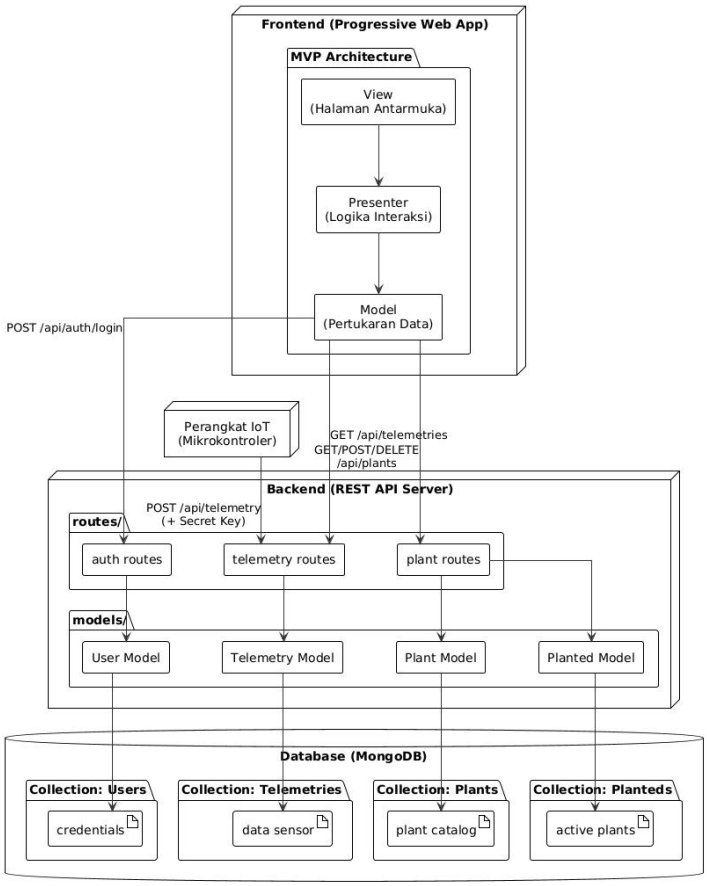
2) Siklus Tampil ke Dashboard (PWA)

Berjalan secara paralel di sisi klien, antarmuka *Frontend* melakukan mekanisme *polling* dengan mengirimkan permintaan GET secara berkala ke *endpoint* /api/telemetries. *Backend* kemudian menarik rekaman data sensor paling mutakhir dari basis data dan mengembalikannya ke *Frontend* dengan status HTTP 200 OK. Aplikasi web kemudian secara dinamis memutakhirkan (*update*) angka visual pada *dashboard* pengguna secara *real-time* tanpa mengharuskan pemuatan ulang (*reload*) halaman peramban.

b. *Component Diagram*

Component Diagram memetakan struktur modular sistem dan hubungan ketergantungan (*dependency*) antarkomponen [19] . Arsitektur ini memisahkan layanan

menjadi antarmuka PWA berbasis *Model-View-Presenter* (MVP) di sisi klien, layanan REST API Node.js di sisi *server*, serta infrastruktur basis data dan perangkat keras IoT yang saling berkomunikasi melalui protokol HTTP (REST API).



Gambar 3 Rancangan *Component Diagram*

Berdasarkan Gambar 2, arsitektur sistem dibangun atas komponen-komponen berikut:

1) *Frontend* (*Progressive Web App*)

   Komponen sisi klien yang dirancang menggunakan pola *Model-View-Presenter* (MVP). Komponen *View* bertanggung jawab atas presentasi halaman antarmuka pengguna, *Presenter* mengoordinasikan logika interaksi, sementara *Model* bertugas mengeksekusi pertukaran data secara asinkron dengan *backend*. Pertukaran data ini difokuskan pada permintaan autentikasi (POST /api/auth/login), penarikan data metrik sensor (GET /api/telemetries), hingga manajemen katalog tanaman (GET/POST/DELETE /api/plants).

2) Perangkat IoT (Mikrokontroler)

   Bertindak sebagai klien perangkat keras independen di lapangan yang mengirimkan data pembacaan sensor (telemetri) secara periodik ke *backend* menggunakan rute POST /api/telemetry. Komunikasi ini diamankan dengan menyertakan *Secret Key* sebagai instrumen validasi autentisitas perangkat.

3) *Backend* (REST API *Server*)

   Berfungsi sebagai pusat kendali logika (API *Gateway*) yang memfasilitasi integrasi antara sisi klien dan basis data. *Backend* mendistribusikan lalu lintas jaringan ke dalam beberapa pemetaan rute spesifik (*auth routes, telemetry routes, plant routes*), yang selanjutnya diteruskan ke lapisan *Models* (*User, Telemetry, Plant,*

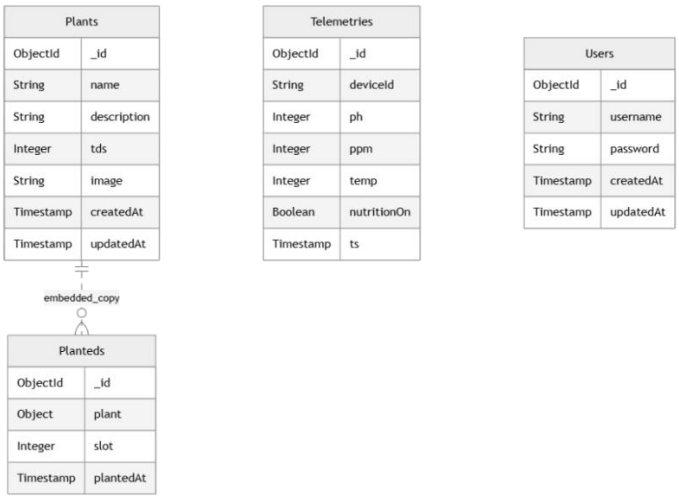
Planted) untuk diproses menjadi kueri basis data.

4) *Database* (MongoDB)

Infrastruktur penyimpanan persisten berbasis NoSQL yang menampung seluruh koleksi data operasional. Komponen ini mengatur pemisahan data ke dalam dokumen spesifik, dengan koleksi Telemetries sebagai pusat penyimpanan data sensor, Plants sebagai katalog referensi, Planteds sebagai penanda komoditas aktif, dan Users untuk menyimpan data kredensial akses

c. *Physical Data Model (PDM)*

Physical Data Model (PDM) merupakan rancangan basis data yang digunakan untuk menggambarkan hubungan antar entitas dalam sistem informasi. Model ini menjadi dasar dalam pembuatan struktur database yang berfungsi sebagai media penyimpanan data pada sistem [20]. Pada sistem ini, basis data diimplementasikan menggunakan MongoDB yang berorientasi dokumen (*document-oriented*). Data disimpan dalam format BSON (*Binary JSON*), sehingga setiap atribut pada koleksi (*collection*) memiliki penentuan tipe data yang tegas seperti *ObjectId, String, Integer, Boolean*, dan *Timestamp.*



Gambar 4 Rancangan Physical Data Model (PDM)

Berdasarkan Gambar 3, struktur basis data dipetakan ke dalam empat koleksi utama yang dirancang untuk mendukung efisiensi pemantauan secara *real-time*:

1) Koleksi Telemetries

Merupakan pusat penyimpanan data historis dengan volume tinggi. Koleksi ini menyimpan rekaman sensor secara periodik dengan struktur data meliputi identitas dokumen (_id), identitas mikrokontroler (deviceId), tingkat keasaman air (ph), konsentrasi nutrisi (ppm), suhu lingkungan (temp), dan penanda waktu pencatatan presisi (ts).

2) Koleksi Plants

Berfungsi sebagai katalog referensi statis komoditas hidroponik. Koleksi ini menyimpan informasi spesifik tanaman seperti nama, deskripsi visual (*image*), serta nilai target kebutuhan nutrisi (tds) yang menjadi acuan logika peringatan dini pada sistem.

3) Koleksi Planteds

Menyimpan informasi komoditas tanaman yang sedang aktif dibudidayakan pada setiap *slot* di fasilitas hidroponik. Relasi antara koleksi Plants dan Planteds diimplementasikan menggunakan pendekatan denormalisasi (*embedded copy*), di

mana dokumen detail tanaman disalin dan disematkan langsung ke dalam atribut plant. Pendekatan ini dipilih secara strategis untuk menghilangkan kebutuhan operasi *join* (*lookup*) antar koleksi, sehingga proses penarikan data oleh antarmuka *frontend* menjadi jauh lebih cepat dan ringan.

4) Koleksi Users

Tempat penyimpanan mandiri untuk data kredensial akses pengguna (username dan password terenkripsi) guna memastikan hanya administrator sah yang dapat mengakses *dashboard* PWA.

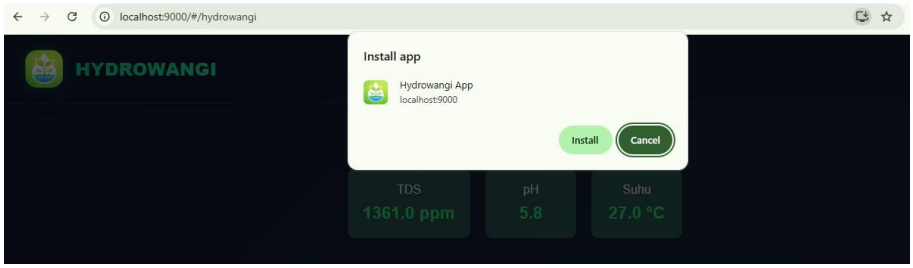
2. Implementasi

Implementasi ini difokuskan pada dua aspek utama: kapabilitas instalasi mandiri layaknya aplikasi *native* dan visualisasi pemantauan (*monitoring*).

a. Konfigurasi *Web App Manifest* dan Instalasi (*Add to Home Screen)*

Karakteristik utama PWA pada sistem ini dibuktikan melalui penerapan *Service Worker* untuk strategi *caching* dan berkas konfigurasi manifest.json. Berkas *manifest* ini mendefinisikan identitas aplikasi, seperti ikon *homescreen*, tema warna (*theme_color*), dan mode tampilan mandiri (*standalone*).

Dengan terpenuhinya kriteria PWA, peramban (*browser*) akan secara otomatis mendeteksi aplikasi dan memicu *prompt* instalasi atau fitur *Add to Home Screen* (A2HS). Fitur ini memungkinkan pengelola KWT Mekarwangi untuk menginstal aplikasi pemantauan langsung ke layar utama perangkat (*smartphone* atau laptop) mereka tanpa harus melalui *App Store* atau *Play Store*.



Gambar 5 *Prompt* Instalasi PWA pada Perambang Pengguna

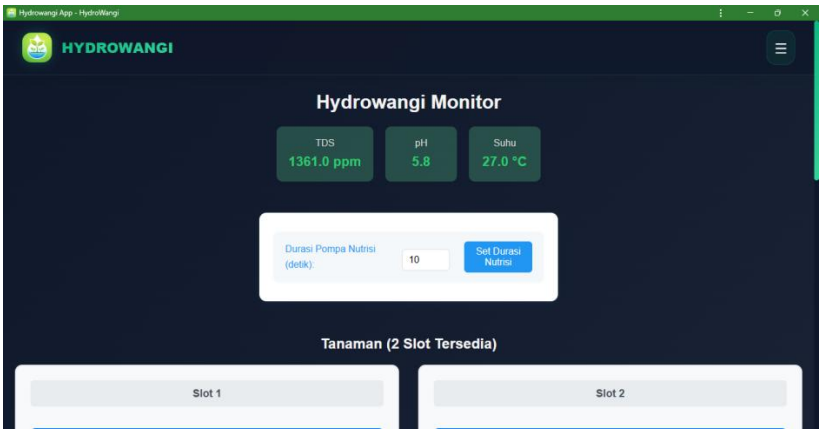
b. Visualisasi Dashboard Monitoring Real-Time

Setelah aplikasi diakses atau diinstal, fungsionalitas pemantauan dipusatkan pada halaman *Dashboard Utama* (Monitor Telemetri). Antarmuka ini dirancang menggunakan desain bernuansa gelap (*dark theme*) untuk memberikan fokus pada elemen metrik yang krusial.

Berdasarkan Gambar 5, dapat dilihat bahwa aplikasi PWA berhasil berjalan dalam mode *standalone* layaknya aplikasi *native* desktop, di mana elemen peramban seperti bilah pencarian (*address bar*) telah dihilangkan secara otomatis. Pada panel monitor, sistem menyajikan tiga indikator lingkungan utama secara akurat, yaitu nilai *Total Dissolved Solids* (TDS) sebesar 1361.0 ppm, tingkat keasaman (pH) sebesar 5.8, dan suhu air pada angka 27.0 °C.

Berkat penerapan arsitektur *Model-View-Presenter* (MVP) di sisi *frontend*, pemutakhiran angka pada panel indikator ini dieksekusi secara asinkron. Aplikasi secara dinamis mengambil rekaman data telemetri terbaru dari *backend* setiap satu detik memanfaatkan fungsi setInterval(). Melalui mekanisme *polling* data yang berkelanjutan ini, pengguna dapat memantau perubahan nilai sensor secara *real-time* tanpa mengharuskan pengguna memuat ulang (*reload*) halaman aplikasi secara manual.

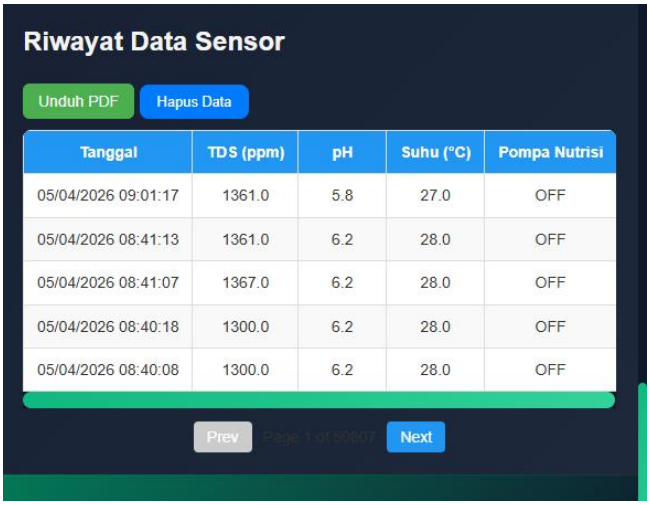
Selain itu, antarmuka ini juga dirancang dengan pewarnaan teks indikator yang dinamis. Warna angka akan memberikan penanda visual responsif kepada pengguna (misalnya berubah menjadi merah) apabila nilai konsentrasi nutrisi terdeteksi berada di luar batas ideal tanaman yang sedang dibudidayakan.



Gambar 6 Implementasi Halaman *Dashboard* Admin

c. Visualisasi Riwayat Data Sensor dan Mekanisme Paginasi

Selain menyajikan pemantauan *real-time*, antarmuka PWA juga mengakomodasi penyajian data historis telemetri yang diimplementasikan dalam bentuk tabel komprehensif. Fitur ini berfungsi krusial untuk merekam jejak kondisi lingkungan hidroponik dari waktu ke waktu sebagai bahan evaluasi panen pengelola.



Gambar 7 Implementasi Tabel Riwayat Data Sensor

Berdasarkan Gambar 7, tabel riwayat menyajikan lima parameter metrik utama secara rapi, yang meliputi catatan waktu perekaman data (Tanggal), nilai konsentrasi nutrisi (TDS), tingkat keasaman (pH), suhu air, dan riwayat status operasional aktuator (Pompa Nutrisi). Nilai kebaruan dan efisiensi pada implementasi antarmuka ini terletak pada penerapan mekanisme paginasi (*pagination*) dinamis. Dalam ekosistem IoT, mikrokontroler mengirimkan data secara terus-menerus sehingga menghasilkan volume data yang masif. Seperti yang terlihat pada indikator navigasi di Gambar 6, sistem terbukti mampu menangani dan memetakan puluhan ribu baris rekaman data (mencapai lebih dari 50.800 halaman).

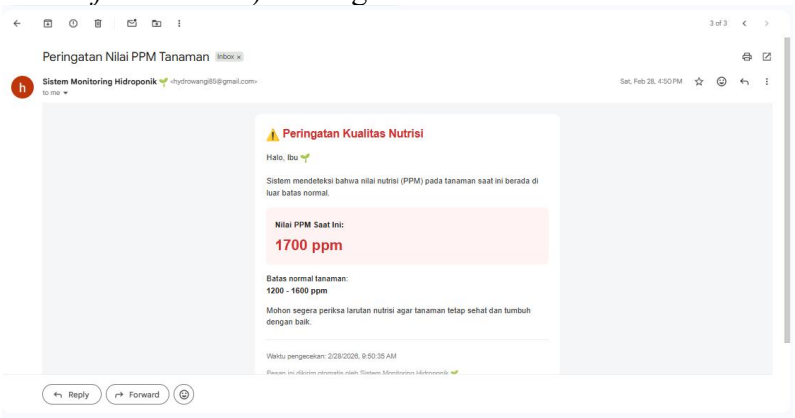
Dengan membatasi jumlah *render* maksimal 5 baris data per halaman, antarmuka PWA berhasil mencegah terjadinya beban memori berlebih (*memory overload*) pada peramban pengguna. Hal ini memastikan aplikasi tetap berjalan ringan, responsif, dan stabil. Selain itu, antarmuka ini terintegrasi dengan fitur manajemen pelaporan melalui tombol "Unduh PDF" untuk mengekspor rekaman menjadi dokumen portabel, serta tombol "Hapus Data" untuk mengakomodasi pembersihan kapasitas penyimpanan basis data

d. Implementasi Fitur Notifikasi Peringatan Dini (*Early Warning System*)

Selain visualisasi pada *dashboard*, sistem juga dilengkapi dengan fitur peringatan dini otomatis (*alerting*) untuk mitigasi risiko penurunan kualitas larutan nutrisi secara cepat. Logika evaluasi peringatan ini dieksekusi murni pada sisi *server backend* untuk mengurangi beban komputasi pada perangkat pengguna.

Setiap kali mikrokontroler IoT mentransmisikan data terbaru, *server* akan membandingkan nilai pembacaan sensor aktual dengan batas toleransi deviasi sebesar

200 ppm dari target ideal tanaman yang terdaftar pada koleksi Planteds. Apabila terdeteksi anomali nilai (seperti konsentrasi nutrisi yang terlalu pekat atau terlalu encer), sistem secara otomatis memicu layanan pengiriman pesan menggunakan protokol SMTP (*Simple Mail Transfer Protocol*) terintegrasi.



Gambar 8 Antarmuka *Email* Notifikasi Peringatan Dini pada Pengguna

Berdasarkan Gambar 8, sistem terbukti berhasil mengirimkan *email* peringatan secara seketika (*real-time*) ke kotak masuk pengelola. Pesan tersebut secara dinamis memuat rincian informasi kritis, seperti status peringatan kualitas nutrisi dan angka metrik aktual yang memicu anomali.

Guna menjaga kenyamanan pengguna dan mencegah terjadinya penumpukan pesan ganda (*spamming*) akibat transmisi data IoT yang terus-menerus, *backend* menerapkan algoritma *state tracking*. Mekanisme ini memberlakukan waktu jeda (*cooldown*) secara ketat selama 30 menit sebelum sistem diizinkan untuk mengirimkan notifikasi peringatan berikutnya.

4. Pengujian Sistem

Pengujian perangkat lunak dilakukan menggunakan metode *Black Box Testing*. Evaluasi ini bertujuan untuk memverifikasi fungsionalitas antarmuka sistem berdasarkan parameter masukan (*input*) dan keluaran (*output*) tanpa perlu meninjau struktur logika kode program di baliknya. Pengujian ini memastikan bahwa setiap fitur antarmuka web PWA—mulai dari proses autentikasi, penyajian data sensor secara *real-time*, rekam jejak historis, hingga integrasi dengan layanan *email* peringatan dini—merespons tindakan pengguna sesuai

dengan skenario spesifikasi kebutuhan.

Tabel 2. Hasil Pengujian Fungsional Sistem Menggunakan Metode Black Box Testing

Kode	Fitur yang Diuji	Skenario Pengujian	Hasil Aktual	Status
TC-01	Autentikasi Pengguna (Login)	Pengguna memasukkan *username* dan *password* terdaftar pada form login.	Sistem memvalidasi data, mencatat sesi, dan mengarahkan pengguna ke halaman *Dashboard*.	Valid
TC-02	Monitoring Sensor *Real-Time*	Mengamati panel indikator pada *dashboard* saat perangkat IoT mengirim data metrik baru.	Angka indikator (TDS, pH, Suhu) berubah secara otomatis tanpa mengharuskan muat ulang halaman.	Valid
TC-03	Visualisasi & Paginasi Riwayat Data	Membuka halaman riwayat sensor dan menavigasi halaman tabel paginasi.	Sistem berhasil memetakan puluhan ribu baris rekaman data ke dalam tabel yang ringan dan responsif.	Valid
TC-04	Unduh Laporan PDF	Menekan tombol ekspor "Unduh PDF" pada antarmuka riwayat sensor.	Peramban berhasil membuat (*generate*) dan mengunduh dokumen laporan ke penyimpanan lokal.	Valid
TC-05	Peringatan Dini (Email)	Mikrokontroler (atau API Client) mengirimkan data TDS anomali sebesar 1700 ppm.	*Server* memicu modul SMTP, dan *email* peringatan berisi rincian nilai aktual berhasil diterima di kotak masuk pengelola.	Valid

KESIMPULAN

Berdasarkan hasil penelitian, implementasi Progressive Web App (PWA) untuk monitoring kualitas nutrisi hidroponik berbasis Internet of Things (IoT) berhasil dikembangkan dan berfungsi sesuai dengan kebutuhan pengguna. Sistem mampu menampilkan data parameter utama hidroponik berupa pH, suhu, dan Total Dissolved Solids (TDS) secara real-time melalui dashboard yang responsif serta dapat diakses layaknya aplikasi native melalui fitur Add to Home Screen (A2HS). Integrasi antara perangkat IoT, backend berbasis Node.js dan MongoDB, serta antarmuka PWA berbasis Model-View-Presenter (MVP) memungkinkan proses pemantauan berlangsung secara efisien, termasuk dalam pengelolaan data historis melalui mekanisme paginasi yang tetap ringan meskipun menangani puluhan ribu rekaman data. Selain itu, fitur early warning system berhasil mengirimkan notifikasi email secara otomatis ketika terjadi penyimpangan kadar nutrisi dari batas yang ditentukan sehingga mendukung pengambilan keputusan yang lebih cepat dan tepat. Hasil pengujian menggunakan metode Black Box Testing menunjukkan seluruh fitur utama sistem, meliputi autentikasi pengguna, monitoring real-time, riwayat data, ekspor laporan PDF, dan notifikasi peringatan dini, berjalan dengan status valid, sehingga sistem yang dikembangkan dinilai efektif, andal, dan layak diterapkan untuk mendukung pengelolaan nutrisi pada budidaya hidroponik.

DAFTAR PUSTAKA

[1] C. W. Kojansow, P. D. K. Manembu, and A. M. Rumagit, "Pengembangan Platform Internet Of Things (IoT) menggunakan komunikasi WebSocket," *Jurnal Teknik Informatika*, vol. 19, no. 3, pp. 259–269, Aug. 2024, doi: 10.35793/jti.v19i3.53680.

[2] Md. A. Awal, A. S. Pio, M. J. Mim, P. K. P. Partha, Md. A. Al Kafi, and S. Farha, "A smart IoT-based hydroponics system for small-scale household in Bangladesh," *Smart Agricultural Technology*, vol. 12, p. 101163, Dec. 2025, doi: 10.1016/j.atech.2025.101163.

[3] S. Karim, I. M. Khamidah, and Yulianto, "Sistem Monitoring Pada Tanaman Hidroponik Menggunakan Arduino UNO dan NodeMCU," *Buletin Poltanesa*, vol. 22, no. 1, Jun. 2021, doi: 10.51967/tanesa.v22i1.331.

[4] F. B. Assa, A. M. Rumagit, and M. E. L. Najoan, "Internet of Things-Based Hydroponic System Monitoring Design Perancangan Monitoring Sistem Hidroponik Berbasis Internet of Things," *Jurnal Teknik Informatika*, vol. 17, no. 1, pp. 129–138, 2022.

[5] A. Aurasopon, T. Thongleam, and S. Kuankid, "Integration of IoT Technology in Hydroponic Systems for Enhanced Efficiency and Productivity in Small-Scale Farming," *Acta Technologica Agriculturae*, vol. 27, no. 4, pp. 203–211, Dec. 2024, doi: 10.2478/ata-2024-0027.

[6] R. D. Rusdiyan Yusron and M. M. Huda, "Analisis Perancangan Sistem Informasi Perpustakaan Menggunakan Model Waterfall Dalam Peningkatan Inovasi Teknologi," *Journal Automation Computer Information System*, vol. 1, no. 1, May 2021, doi: 10.47134/jacis.v1i1.4.

[7] Sigit Umar Anggono, Edy Siswanto, Laksamana Rajendra Haidar Azani Fajri, and Munifah, "User Interface Berbasis Web Pada Perangkat Internet Of Things," *Teknik: Jurnal Ilmu Teknik dan Informatika*, vol. 3, no. 1, pp. 35–54, May 2023, doi: 10.51903/teknik.v3i1.326.

[8] M. S. S. Lingolu and M. K. Dobbala, "A Comprehensive Review of Progressive Web Apps: Bridging the Gap Between Web and Native Experiences," *International Journal of Science and Research (IJSR)*, vol. 11, no. 2, pp. 1326–1334, Feb. 2022, doi: 10.21275/SR24517172948.

[9] Simanshi, S. Pandey, and S. Sinha, "Progressive Web App: An Adventure into Faster Web Experiences," *International Journal of Research Publication and Reviews*, vol. 5, no. 4, pp. 3065–3070, Apr. 2024, [Online]. Available: www.ijrpr.com

[10] Ahyar Muawwal, "The Implementation of PWA (Progressive Web App) Technology in Enhancing Website Performance & Mobile Accessibility," *Buletin Pos dan Telekomunikasi*, vol. 22, no. 1, Jun. 2024, doi: 10.17933/bpostel.v22i1.395.

[11] D. B. J. Rao, P. Kalpana, G. S. N. Kumar, and N. M. Atcha, "A Comparative Analysis of Software Development Models: Waterfall, Agile and DevOps," 2025, pp. 589–599. doi: 10.1007/978-981-97-8355-7_51.

[12] R. Abdul *et al.*, "The Waterfall Model-Software Engineering," *International Journal of Research Publication and Reviews Journal homepage: www.ijrpr.com*, vol. 5, no. 4, pp. 1–6, 2024, [Online]. Available: www.ijrpr.com

[13] A. Saravanos and M. X. Curinga, "Simulating the Software Development Lifecycle: The Waterfall Model," *Applied System Innovation*, vol. 6, no. 6, p. 108, 2023, doi: 10.3390/asi6060108.

[14] Siska Narulita, Ahmad Nugroho, and M. Zakki Abdillah, "Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS)," *Bridge : Jurnal publikasi Sistem Informasi dan Telekomunikasi*, vol. 2, no. 3, pp. 244–256, Aug. 2024, doi:

10.62951/bridge.v2i3.174.

[15] F. I. Maulana, V. Susanto, P. Shilo, J. R. Gunawan, G. Pangestu, and D. R. B. Raharja, "Design and Development of Website Dr.Changkitchen Diet Catering Using SDLC Waterfall Model," in *Proceedings of the 6th International Conference on Sustainable Information Engineering and Technology (SIET)*, 2021, pp. 75–79. doi: 10.1145/3479645.3479652.

[16] J. Halim, Wilson, and R. Tanwijaya, "Design of a web-based company profile information system for Mestika Abadi School (Chong Ren School) using the waterfall method," *Journal of Artificial Intelligence and Engineering Applications (JAIEA)*, vol. 5, no. 1, pp. 1294–1300, Oct. 2025, doi: 10.59934/jaiea.v5i1.1604.

[17] P. C. Jorgensen, *Software Testing: A Craftsman's Approach*, 4th ed. Boca Raton, FL: CRC Press, 2014.

[18] H. Koç, A. M. Erdoğan, Y. Barjakly, and S. Peker, "UML Diagrams in Software Engineering Research: A Systematic Literature Review," in *The 7th International Management Information Systems Conference*, Basel Switzerland: MDPI, Mar. 2021, p. 13. doi: 10.3390/proceedings2021074013.

[19] R. deshmukh and S. Kumar, "Unified Modeling Language (UML) and Its Contribution to Software Maintenance Efficiency," *Advances in Software Engineering*, Nov. 2023.

[20] Amiruddin, Fajriyanto, and F. Lazim, "Sistem Informasi Akademik Pada MTs Salafiyah Syafi'iyah Menggunakan Framework Codeigniter Dan MYSQL," *JUSTIFY : Jurnal Sistem Informasi Ibrahimy*, vol. 1, no. 1, pp. 51–57, Jul. 2022, doi: 10.35316/justify.v1i1.2103.