# Turnitin LLCC

## 2213030119_Erlangga_Fajar_Ramadhan

SI-02

### Document Details

Submission ID
trn:oid:::2945:394012613

Submission Date
Jun 18, 2026, 11:53 AM GMT+7

Download Date
Jun 18, 2026, 11:56 AM GMT+7

File Name
2213030119_Erlangga_Fajar_Ramadhan.pdf

File Size
2.0 MB

105 Pages

20,831 Words

135,728 Characters

# 8% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

## Filtered from the Report

- Bibliography

## Top Sources

7% Internet sources
4% Publications
0% Submitted works (Student Papers)

## Integrity Flags

**0 Integrity Flags for Review**

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

## Top Sources

7% Internet sources
4% Publications
0% Submitted works (Student Papers)

## Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

| | | |
|---|---|---|
| 1 | Internet | repository.unpkediri.ac.id 1% |
| 2 | Internet | repository.dinamika.ac.id <1% |
| 3 | Internet | elib.pnc.ac.id <1% |
| 4 | Internet | repository.upi.edu <1% |
| 5 | Internet | kc.umn.ac.id <1% |
| 6 | Publication | Anindya Kinarya Yang Esa Riyanto, Trisya Septiana, Resty Annisa. "PENGEMBANG... <1% |
| 7 | Internet | ejurnal.lkpkaryaprima.id <1% |
| 8 | Internet | ejournal.nusacendekia.or.id <1% |
| 9 | Internet | text-id.123dok.com <1% |
| 10 | Internet | repositori.unimma.ac.id <1% |
| 11 | Internet | www.coursehero.com <1% |

12 Internet
www.vinasoftware.com.vn <1%

13 Internet
repository.its.ac.id <1%

14 Publication
Adi Muhammat Maulidin, Munawaroh Munawaroh. "Perancangan Sistem Inform... <1%

15 Internet
eprints.upj.ac.id <1%

16 Internet
www.scribd.com <1%

17 Publication
Dano Fadilah Amelya Rizki Author, Umi Chotijah Author. "PERANCANGAN SISTEM ... <1%

18 Publication
Sakilawati Sakilawati, Donna Boedi Maritasari, Hadiatul Rodiyah, Muhammad Hu... <1%

19 Internet
etheses.iainkediri.ac.id <1%

20 Internet
www.bola.com <1%

21 Publication
Muhammad Fikri Pratama, Teguh Andriyanto, Anita Sari Wardani. "Design and Bu... <1%

22 Internet
docplayer.info <1%

23 Internet
journal.uinsgd.ac.id <1%

24 Publication
Gagah Dwiki Putra Aryono, Ferdy Oka Agung Permana, Sigit Auliana. "PERANCAN... <1%

25 Publication
Radja Ravine Salfriandry. "PERANCANGAN WEBSITE SHOWROOM MOBIL BERBASI... <1%

| | | |
|---|---|---|
| 26 | Internet | |
| ejournal-ibik57.ac.id | | <1% |
| 27 | Internet | |
| repository.ubpkarawang.ac.id | | <1% |
| 28 | Internet | |
| repository.uin-suska.ac.id | | <1% |
| 29 | Publication | |
| Oktavian Putra Kurniawan, Chairul Rizal, Leni Marlina. "Perancangan Sistem Apli... | | <1% |
| 30 | Publication | |
| Rensi Purnama Sari, Indra Sakti, Dedy Hamdani. "PENGEMBANGAN LEMBAR KERJ... | | <1% |
| 31 | Internet | |
| eprints.unipdu.ac.id | | <1% |
| 32 | Internet | |
| repository.nusamandiri.ac.id | | <1% |
| 33 | Internet | |
| core.ac.uk | | <1% |
| 34 | Internet | |
| e-jurnal.lppmunsera.org | | <1% |
| 35 | Internet | |
| journal.eng.unila.ac.id | | <1% |
| 36 | Internet | |
| repository.stiesia.ac.id | | <1% |
| 37 | Internet | |
| repository.umy.ac.id | | <1% |
| 38 | Internet | |
| repository.unsulbar.ac.id | | <1% |
| 39 | Publication | |
| Fery Sofian Efendi, Rahmad Aliy Cagar Wahyu Aji, Toga Aldila Cinderatama, Benni... | | <1% |

40 Publication
Gibran Alfarabi, Puput Budi Wintoro. "PENGEMBANGAN FRONT-END WEB PENCAR... <1%

41 Internet
jom.fti.budiluhur.ac.id <1%

42 Internet
publikasi.hawari.id <1%

43 Publication
Ahmad Azriel Friyansyah, Deni Sutaji. "PENGEMBANGAN FITUR LOGGING DAN DA... <1%

44 Publication
Padang Wardoyo, Komarudin Umar. "Sistem Pakar Diagnosis Penyakit Tanaman ... <1%

45 Internet
www.slideshare.net <1%

46 Publication
Esau Towansiba, Yuliana Sangka, Sofyan Sofyan. "Aplikasi Sistem Pengaduan Mas... <1%

47 Internet
adoc.pub <1%

48 Internet
eprints.poltekkesadisutjipto.ac.id <1%

49 Internet
jurnal.itg.ac.id <1%

50 Internet
proceeding.unpkediri.ac.id <1%

51 Internet
repository.syekhnurjati.ac.id <1%

52 Internet
repository.ummat.ac.id <1%

53 Publication
Isfa Auliyani, Anisah Anisah. "Digitalisasi Absensi Dan Jurnal Harian Guru Berbasi... <1%

| | | |
|---|---|---|
| 54 | Publication | |
| M. Syafiih, M. Ilham Zidni Syauqillah, M. Alif Oleo Pimbiriri. "Peningkatan Pendaft... | | <1% |
| 55 | Internet | |
| cdn.juris.id | | <1% |
| 56 | Internet | |
| docobook.com | | <1% |
| 57 | Internet | |
| repository.iainpurwokerto.ac.id | | <1% |
| 58 | Internet | |
| repository.unika.ac.id | | <1% |
| 59 | Internet | |
| scholar.archive.org | | <1% |
| 60 | Publication | |
| Aferbina Barus, Tiur Malasari Siregar, Fasya Maharani Umri, Jesika Replesia Rajag... | | <1% |
| 61 | Publication | |
| Edi Sudarsono, M Yazed Vebriandi. "IMPLEMENTASI FRAMEWORK LARAVEL FILAM... | | <1% |
| 62 | Publication | |
| Siska Devella, Yohannes Yohannes, Nur Rachmat. "PELATIHAN PEMBUATAN WEBS... | | <1% |
| 63 | Internet | |
| dspace.uii.ac.id | | <1% |
| 64 | Internet | |
| es.scribd.com | | <1% |
| 65 | Internet | |
| gildaarianti19.blogspot.com | | <1% |
| 66 | Internet | |
| idoc.tips | | <1% |
| 67 | Internet | |
| makanarea.com | | <1% |

| | | |
|---|---|---|
| 68 | Internet | |
| muuktest.com | | <1% |
| 69 | Internet | |
| ojs3.lppm-uis.org | | <1% |
| 70 | Internet | |
| 1023tpfm.blogspot.com | | <1% |
| 71 | Internet | |
| 123dok.com | | <1% |
| 72 | Publication | |
| Ahmad Khadafi, Surya Hendra Putra, Jamaludin Jamaludin. "Rancang Bangun Apli... | | <1% |
| 73 | Internet | |
| eprints.unm.ac.id | | <1% |
| 74 | Internet | |
| eprints.utdi.ac.id | | <1% |
| 75 | Internet | |
| fdocumenti.com | | <1% |
| 76 | Internet | |
| fr.scribd.com | | <1% |
| 77 | Internet | |
| fr.slideshare.net | | <1% |
| 78 | Internet | |
| id.scribd.com | | <1% |
| 79 | Internet | |
| idwebhost.com | | <1% |
| 80 | Internet | |
| journal.aptii.or.id | | <1% |
| 81 | Internet | |
| konsultasiskripsi.com | | <1% |

82 Internet
mahasiswateknokrat19.blogspot.com <1%

83 Internet
repository.unissula.ac.id <1%

84 Internet
researchportal.plymouth.ac.uk <1%

85 Internet
si.fst.uinjkt.ac.id <1%

86 Internet
studentjournal.petra.ac.id <1%

87 Internet
tagalayapkm.com <1%

88 Internet
widuri.raharja.info <1%

89 Publication
Muhammad Saiful, Aswian Editri Sutriandi, Yupi Kuspandi Putra, L M Samsu. "In... <1%

90 Publication
Sukri Sukri, Mukhroji Mukhroji, Khairuman Khairuman. "Sistem Informasi Pengel... <1%

Skripsi oleh :

**ERLANGGA FAJAR RAMADHAN**

NPM : 2213030119

Judul

**RANCANG BANGUN *EMERGENCY SYSTEM* BERBASIS WEB**

Telah disetujui untuk diajukan Kepada

Panitia Ujian/Sidang Skripsi Program Studi Sistem Informasi

FTIK UN PGRI Kediri

Tanggal: 9 Juni 2026

| Pembimbing I | Pembimbing II |
| --- | --- |
| **Rini Indriati, M.Kom** | **Dwi Harini, M.M** |
| **NIDN. 0725057003** | **NIDN. 0701037003** |

Skripsi oleh :

**ERLANGGA FAJAR RAMADHAN**

NPM : 2213030119

Judul

**RANCANG BANGUN *EMERGENCY SYSTEM* BERBASIS WEB**

Telah dipertahankan di depan Panitia Ujian/Sidang Skripsi

Program Studi Sistem Informasi FTIK UN PGRI Kediri

**Dan Dinyatakan telah Memenuhi Persyaratan**

Panitia Penguji :

1. Ketua : Isi Nama Ketua Penguji [.....]
2. Penguji 1 : Isi Nama Penguji 1 [.....]
3. Penguji 2 : Isi Nama Ketua Penguji 2 [.....]

Mengetahui,
Dekan Fakultas Teknik dan Ilmu Komputer

**Dr. Sulistiono, M.Si**
**NIDN. 0007076801**

# PERNYATAAN

Yang bertanda tangan di bawah ini saya,

| | |
|---|---|
| Nama | : Erlangga Fajar Ramadhan |
| Jenis Kelamin | : Laki-laki |
| Tempat/tgl. lahir | : Kediri / 12 Desember 2001 |
| NPM | : 2213030119 |
| Fak/Prodi. | : FTIK / S1-Sistem Informasi |

menyatakan dengan sebenarnya, bahwa dalam Skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu perguruan tinggi, dan sepanjang pengetahuan saya tidak terdapat karya tulis atau pendapat yang pernah diterbitkan oleh orang lain, kecuali yang secara sengaja dan tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Kediri, Juni 2026
Yang Menyatakan

[Materai 10.000]

**ERLANGGA FAJAR RAMADHAN**
**NPM: 2213030119**

# MOTTO

*Innamaal usriyusro.* "Sesungguhnya beserta kesulitan ada kemudahan".

***~ Surah Al-Insyirah ayat 5-6***

"Kamu memiliki kendali atas pikiranmu bukan kejadian-kejadian di luar sana. Sadari ini, dan kamu akan menemukan kekuatan."

***~ Marcus Aurelius***

"Inovasi adalah melihat apa yang dilihat semua orang dan memikirkan apa yang tidak dipikirkan orang lain."

***~ Dr. Albert Szent-Györgyi***

# ABSTRAK

**Erlangga Fajar Ramadhan:** Rancang Bangun *Emergency System* Berbasis Web, Skripsi, Sistem Informasi, Fakultas Teknik dan Ilmu Komputer Universitas Nusantara PGRI Kediri, 2026.

**Kata Kunci:** *Emergency System, Waterfall, React.js, CodeIgniter 4, Firebase Cloud Messaging, Geolokasi, RBAC*

Pelaporan insiden darurat di Dinas Pemadam Kebakaran Kabupaten Kediri masih bersifat konvensional melalui *WhatsApp* dan pencatatan manual menggunakan *Microsoft Excel*, sehingga menyebabkan miskomunikasi, ketidakakuratan lokasi, serta tidak adanya dokumentasi terstruktur. Kondisi ini menghambat kecepatan respons petugas dan transparansi informasi kepada masyarakat. Penelitian ini bertujuan merancang dan membangun Sistem Informasi Respon Tanggap Darurat (SIRTADA) berbasis web untuk mengatasi permasalahan tersebut. Metode pengembangan yang digunakan adalah *Waterfall*, dengan analisis kebutuhan melalui wawancara, perancangan menggunakan UML, implementasi *Backend* menggunakan *CodeIgniter* 4 dan MySQL, *Frontend* menggunakan *React*.js serta Tailwind CSS, dan pengujian menggunakan *strategi multi-level* yang mencakup Unit *Testing*, *Integration testing*, Performance *Testing*, Security *Testing*, *Blackbox Testing*, dan *User Acceptance Testing* (UAT). Hasil pengujian menunjukkan seluruh 63 unit test dan 78 dari 80 *assertion* integration test lulus (97,5%). Rata-rata *response time* API adalah 474ms (jauh di bawah batas 15 detik), latensi notifikasi *Firebase* Cloud Messaging (FCM) berkisar 1–3 detik (di bawah batas 8 detik), dan *page load time* sekitar 2 detik. Nilai 474ms merupakan rata-rata pengujian performa *endpoint* CRUD. *Endpoint* POST /api/reports/panic memiliki rata-rata lebih tinggi (3–4 detik) karena melibatkan distribusi notifikasi FCM dan *WhatsApp Gateway* secara sinkron, sehingga rata-rata keseluruhan 41 *endpoint integration testing* adalah 1.032ms. UAT terhadap 5 responden (masyarakat, petugas, admin) menghasilkan persentase rata-rata 89,3% dengan kategori "Sangat Baik". Sistem berhasil mengimplementasikan *panic button* dengan geolokasi otomatis, pelacakan laporan via kode unik, *dashboard* pemantauan *real-time* berbasis peran (RBAC), serta notifikasi push FCM dan *WhatsApp Gateway* berbasis *Baileys*. Kesimpulannya, sistem SIRTADA layak digunakan untuk meningkatkan efisiensi dan akurasi pelaporan darurat di Dinas Pemadam Kebakaran Kabupaten Kediri. Limitasi penelitian ini adalah penyimpanan foto setiap laporan hanya dapat menyertakan satu foto, dan belum ada mekanisme *refresh token*. Pengembangan selanjutnya disarankan menambahkan fitur multiple foto, integrasi dengan instansi lain (BPBD, Dinas Kesehatan), serta konversi ke Progressive Web App (PWA).

# KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah Tuhan Yang Maha Kuasa, karena atas rahmat, taufik, dan hidayah-Nya penyusunan skripsi yang berjudul "Rancang Bangun *Emergency System* Berbasis Web" ini dapat diselesaikan dengan baik. Penyusunan skripsi ini tidak terlepas dari dukungan, arahan, dan bantuan berbagai pihak. Oleh karena itu, pada kesempatan ini penulis menyampaikan terima kasih dan penghargaan yang sebesar-besarnya kepada:

1. Dr. Zainal Afandi, M.Pd., selaku Rektor Universitas Nusantara PGRI Kediri.
2. Dr. Sulistiono, M.Si., selaku Dekan Fakultas Teknik dan Ilmu Komputer Universitas Nusantara PGRI Kediri.
3. Dr. Sucipto, M.Kom., selaku Ketua Program Studi Sistem Informasi Universitas Nusantara PGRI Kediri.
4. Rini Indriati, M.Kom., selaku Dosen Pembimbing Pertama yang telah memberikan arahan, masukan, serta bimbingan selama proses penyusunan skripsi ini.
5. Dwi Harini,S. Si, M.M., selaku Dosen Pembimbing Kedua yang telah memberikan arahan, masukan, serta bimbingan selama proses penyusunan skripsi ini.
6. Dinas Pemadam Kebakaran Kabupaten Kediri selaku instansi yang sudah bersedia menyediakan tempat dan waktu dalam penelitian saya.
7. Kedua orang tua tercinta, yang selalu memberikan doa, dukungan moral, semangat, serta pengorbanan yang tidak ternilai sehingga penulis dapat menempuh pendidikan hingga tahap ini.
8. Orang tersayang yang telah mendukung dan memotivasi saya untuk menyelesaikan skripsi saya ini.

Disadari bahwa skripsi ini masih banyak kekurangan, maka diharapkan tegur sapa, kritik dan saran-saran, dari berbagai pihak sangat diharapkan. Akhirnya, disertai harapan semoga skripsi ini ada manfaatnya bagi kita semua, khususnya bagi dunia pendidikan, meskipun hanya ibarat setitik air bagi samudra yang luas.

Kediri, 29 Mei 2026

**Erlangga Fajar Ramadhan**
**2213030119**

# BAB I
# PENDAHULUAN

## A. Latar Belakang

Pelayanan darurat merupakan bagian penting dari layanan publik karena berkaitan langsung dengan keselamatan jiwa, aset, dan kecepatan penanganan kejadian di lapangan (Pratama et al., 2023). Pada Dinas Pemadam Kebakaran Kabupaten Kediri, kebutuhan terhadap informasi yang cepat, akurat, dan mudah ditindaklanjuti menjadi sangat penting karena petugas harus merespons berbagai laporan insiden secara tepat waktu (Latuconsina et al., 2024).

Berdasarkan hasil wawancara dengan koordinator lapangan Dinas Pemadam Kebakaran Kabupaten Kediri, alur pelaporan yang berjalan saat ini masih dilakukan secara konvensional. Masyarakat menyampaikan laporan melalui *WhatsApp* secara tidak terstruktur, sedangkan pencatatan dan dokumentasi laporan dilakukan secara manual menggunakan *Microsoft Excel* setelah kejadian berlangsung. Kondisi ini menunjukkan bahwa belum tersedia sistem terpusat yang mampu mengelola pelaporan, pelacakan status, dan dokumentasi insiden dalam satu *platform* digital. Kondisi ini sejalan dengan temuan (Karaman et al., 2024) bahwa ketiadaan sistem informasi terintegrasi menghambat efisiensi kerja dan kualitas pengelolaan data pada institusi pelayanan publik.

Kondisi tersebut menimbulkan beberapa dampak. Informasi yang diterima petugas sering tidak tersusun dengan baik, sehingga lokasi kejadian dapat menjadi kurang akurat dan memperlambat respons di lapangan. Di sisi lain, belum tersedianya dokumentasi visual dan deskripsi kejadian yang terstruktur juga menyulitkan petugas dalam memperkirakan kebutuhan personel maupun peralatan sebelum diberangkatkan. Selain itu, masyarakat belum memiliki mekanisme pelaporan darurat yang cepat, sederhana, dan dapat dipantau statusnya secara mandiri.

Melihat kondisi tersebut, terlihat adanya celah penelitian yang jelas, yaitu belum adanya sistem terintegrasi untuk pelaporan, pelacakan, notifikasi darurat, dan pengarsipan insiden dalam satu *platform*. Untuk menjawab kebutuhan tersebut, penelitian ini mengembangkan Sistem Informasi Respon Tanggap Darurat (SIRTADA) berbasis web yang memungkinkan masyarakat melaporkan

insiden darurat secara cepat disertai lokasi otomatis, sementara petugas dan admin dapat memantau laporan, memperbarui status penanganan, dan menerima notifikasi secara *real-time*. Dengan demikian, sistem diharapkan mampu memperbaiki alur komunikasi, mempercepat koordinasi, dan meningkatkan akurasi data penanganan insiden.

73 Berdasarkan seluruh permasalahan dan solusi yang telah dipaparkan, penelitian ini dilakukan dengan judul "Rancang Bangun *Emergency System* Berbasis Web", yang dikembangkan untuk Dinas Pemadam Kebakaran Kabupaten Kediri menggunakan metode *Waterfall*.

## B. Batasan Masalah

Batasan masalah ini dibuat untuk memperjelas aspek-aspek yang hendak diteliti dan memastikan fokus penelitian sesuai dengan tujuan perancangan dan pembangunan *Emergency System* berbasis web, antara lain:

1. Sistem dikembangkan khusus untuk Dinas Pemadam Kebakaran Kabupaten Kediri.
2. Sistem dibangun dalam bentuk aplikasi web responsif dan tidak mencakup pengembangan aplikasi *mobile native*.
3. Pengguna sistem dibatasi pada tiga peran utama, yaitu masyarakat, petugas, dan admin.
4. Fitur yang dikembangkan meliputi pelaporan darurat, pelacakan status laporan, pengelolaan laporan, *dashboard* pemantauan, notifikasi *real-time*, dan pengelolaan pengguna.
5. Sistem menggunakan geolokasi otomatis, tetapi tidak mencakup panggilan suara, komunikasi telepon, atau integrasi dengan call center.
6. Integrasi notifikasi dilakukan melalui FCM dan *WhatsApp Gateway* sebagai kanal yang mendukung sistem, tetapi penelitian ini tidak mencakup integrasi dengan instansi darurat lain di luar Dinas Pemadam Kebakaran Kabupaten Kediri.

## C. Rumusan Masalah

1. Bagaimana merancang dan membangun *Emergency System* berbasis web untuk pelaporan dan pemantauan insiden darurat pada Dinas Pemadam Kebakaran Kabupaten Kediri?
2. Bagaimana mengimplementasikan fitur pelaporan darurat, pelacakan status laporan, geolokasi otomatis, serta notifikasi *real-time* pada sistem?

## D. Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Merancang dan membangun *Emergency System* berbasis web yang dapat digunakan untuk pelaporan dan pemantauan insiden darurat pada Dinas Pemadam Kebakaran Kabupaten Kediri.
2. Mengimplementasikan fitur pelaporan darurat, pelacakan status laporan, geolokasi otomatis, dan notifikasi *real-time* ke dalam sistem.

## E. Manfaat Penelitian

### 1. Manfaat Teoritis

a. Penelitian ini dapat menjadi referensi bagi pengembangan sistem informasi darurat berbasis web, terutama yang memanfaatkan geolokasi, notifikasi *real-time*, dan pengelolaan laporan terstruktur.
b. Penelitian ini menambah kajian tentang penerapan metode *Waterfall* pada pengembangan sistem dengan kebutuhan operasional yang jelas dan terukur.

### 2. Manfaat Praktis

a. Bagi masyarakat, sistem ini mempermudah proses pelaporan insiden darurat agar lebih cepat, terarah, dan dapat dipantau statusnya.
b. Bagi Dinas Pemadam Kebakaran Kabupaten Kediri, sistem ini membantu mempercepat koordinasi, meningkatkan ketepatan informasi lokasi, serta mempermudah dokumentasi dan monitoring laporan.
c. Bagi peneliti atau pengembang selanjutnya, sistem ini dapat menjadi dasar untuk pengembangan sistem lanjutan seperti

integrasi instansi lain, peningkatan *platform* ke arah yang lebih luas dan lain-lain.

# BAB II
# KAJIAN PUSTAKA

## A. Kajian Teori

Kajian teori ini berfungsi sebagai landasan ideal yang mendefinisikan prinsip-prinsip dasar serta justifikasi teknologi dan metode yang relevan dengan perancangan dan pembangunan *Emergency System* berbasis Web. Kajian ini memaparkan konsep inti yang mendukung solusi sistem pada penelitian.

1. Sistem Informasi

Menurut Fithrie Soufitri (2023) Sistem Informasi (SI) didefinisikan sebagai sebuah kumpulan komponen yang memiliki keterkaitan satu sama lain dengan tujuan utama untuk mengumpulkan, mengolah, menyimpan, serta mendistribusikan data agar bertransformasi menjadi informasi yang bermakna. Proses transformasi ini sangat krusial karena memastikan bahwa setiap fakta mentah yang dikelola dapat disajikan kembali dalam bentuk yang lebih berguna bagi pihak-pihak yang membutuhkan. Secara fungsional, keberadaan Sistem Informasi berperan penting dalam mendukung aktivitas organisasi melalui fungsinya yang membantu proses pengambilan keputusan, memperkuat koordinasi antar bagian, hingga menjalankan mekanisme pengawasan terhadap seluruh kegiatan operasional (AAJ Permana et al., 2024). Dengan adanya sistem yang terstruktur, suatu instansi dapat meminimalisir kesalahan manusia dan memastikan bahwa setiap langkah kerja yang diambil didasarkan pada data yang valid serta terdokumentasi dengan baik. Dalam lingkup pelayanan publik, penerapan Sistem Informasi memiliki target strategis untuk meningkatkan efisiensi operasional serta meningkatkan mutu pelayanan yang diberikan kepada masyarakat. Melalui integrasi teknologi yang efektif, sistem ini mampu menyatukan berbagai proses layanan yang sebelumnya terpisah menjadi satu kesatuan yang lebih cepat, transparan, dan responsif terhadap kebutuhan publik (Yusman et al., 2024).

2. *Emergency System*

*Emergency System* merupakan sebuah *platform* digital terintegrasi yang dirancang untuk menjembatani kebutuhan pelaporan situasi kritis dari masyarakat kepada instansi terkait secara *real-time* (Kumar & M. J, 2025). Keberadaan *Emergency System* bertujuan untuk menghilangkan hambatan komunikasi konvensional, sehingga arus informasi mengenai kejadian darurat dapat mengalir dengan cepat dan segera mendapatkan atensi dari pihak berwenang. Penggunaan teknologi informasi dalam skenario ini memastikan bahwa setiap laporan tidak hanya sekadar masuk, tetapi juga memiliki struktur data yang jelas untuk segera ditindaklanjuti. Secara operasional, *Emergency System* menekankan pada akurasi data lokasi dan detail insiden yang dikirimkan melalui teknologi perangkat lunak, seperti aplikasi berbasis web. Hal ini memungkinkan petugas untuk memvalidasi posisi koordinat pelapor secara presisi, sehingga meminimalisir risiko keterlambatan akibat ketidaktahuan medan atau informasi lokasi yang ambigu, dengan adanya data yang akurat sejak tahap awal, tim penyelamat dapat menentukan rute tercepat dan mempersiapkan peralatan yang paling relevan dengan jenis keadaan darurat yang sedang dihadapi di lapangan (Fajar Susanti et al., 2024).

3. Aplikasi Web

29

Ridho Pamungkas (2018) menjelaskan bahwa aplikasi web adalah perangkat lunak yang diakses melalui *browser* standar dengan memanfaatkan jaringan internet tanpa memerlukan instalasi khusus pada perangkat pengguna. Keunggulan utama dari teknologi ini terletak pada fleksibilitas aksesnya yang selalu tersedia 24 jam penuh serta dapat dijangkau kapan saja dan dari lokasi mana pun. Hal ini memungkinkan sistem bekerja secara lintas *platform*, sehingga pengguna dapat berinteraksi dengan aplikasi secara instan menggunakan berbagai perangkat berbeda tanpa membebani ruang penyimpanan lokal. Ditinjau dari sisi operasional, aplikasi web menawarkan efisiensi tinggi dalam pemeliharaan karena seluruh pembaruan fitur dan data dilakukan secara terpusat pada server. Mekanisme ini menjamin pengguna selalu mendapatkan versi sistem terbaru secara otomatis tanpa perlu melakukan pembaruan manual, sehingga meminimalisir kendala teknis pada sisi klien. Dengan dukungan

integrasi data yang kuat, aplikasi berbasis web menjadi solusi ideal untuk manajemen informasi yang cepat, transparan, dan terintegrasi dalam mendukung berbagai kebutuhan layanan digital.

4. Metode *Waterfall*

Metode *Waterfall* merupakan model pengembangan perangkat lunak yang dilakukan secara berurutan, di mana setiap fase harus diselesaikan secara penuh sebelum beralih ke tahap berikutnya. Alur ini dimulai dari analisis kebutuhan untuk menetapkan spesifikasi sistem secara mendalam, diikuti oleh desain sistem untuk merancang arsitektur, hingga tahap implementasi atau pengodean. Karakteristik utamanya yang linier memastikan bahwa setiap langkah memiliki dasar yang kuat, sehingga risiko tumpang tindih antar fase dapat diminimalisir demi menjaga integritas proyek dari awal hingga akhir (Ningki & P, 2023). Setelah proses pembangunan selesai, sistem akan melalui tahap pengujian menyeluruh guna memastikan fungsionalitasnya sesuai rencana sebelum akhirnya masuk ke fase pemeliharaan untuk menjaga performa sistem dalam jangka panjang. Keunggulan model ini terletak pada struktur yang sangat jelas dan dukungan dokumentasi yang detail pada setiap tahapannya, menjadikannya pilihan ideal bagi proyek dengan cakupan yang stabil dan persyaratan yang telah terdefinisi secara matang sejak awal. Dengan pendekatan yang terorganisir ini, manajemen proyek menjadi lebih terukur karena progres setiap tahapan dapat dipantau dengan mudah guna menjamin kualitas hasil akhir (Giandari Maulani et al., 2025).

5. *Role-Based Access Control* (RBAC)

*Role-Based Access Control* (RBAC) adalah model keamanan yang mengatur hak akses pengguna berdasarkan peran (*role*) yang ditetapkan dalam sistem, bukan berdasarkan identitas individu secara langsung. Setiap peran membawa seperangkat izin (*permissions*) yang sudah ditentukan, sehingga pengelolaan akses menjadi terpusat dan konsisten. Model ini banyak diterapkan pada sistem informasi berbasis web karena memudahkan administrasi pengguna dalam skala besar tanpa konfigurasi izin secara satu per satu (Asrianda, 2016).

Dalam *Emergency System* SIRTADA, RBAC diimplementasikan untuk membedakan tiga peran utama: Masyarakat (dapat membuat laporan dan

memantau status laporannya), Petugas (dapat melihat laporan masuk, memperbarui status penanganan, dan mengakses navigasi lokasi), serta Admin (memiliki seluruh akses petugas ditambah manajemen data pengguna dan ekspor laporan). Pembatasan akses antar peran diterapkan melalui *middleware JwtAuthFilter* dan *AdminFilter* pada sisi *Backend*, sehingga setiap *endpoint API* hanya dapat diakses oleh peran yang berwenang.

6. *React JS*

*React JS* merupakan sebuah pustaka *JavaScript* yang menjadi fondasi utama dalam membangun antarmuka pengguna (UI) yang interaktif dan responsif pada pengembangan sistem informasi modern (Dwivedi et al., 2022).

Keunggulan teknisnya terletak pada kemampuan untuk mengelola pembaruan data secara dinamis melalui fitur *Virtual DOM*, yang memungkinkan perubahan informasi terjadi tanpa harus memuat ulang seluruh halaman aplikasi (Isjrem, 2023).

Dalam konteks sistem darurat, efisiensi ini sangat krusial guna memberikan pengalaman pengguna yang cepat, ringan, dan stabil, sehingga pelaporan insiden dapat dilakukan secara *real-time* tanpa hambatan performa yang berarti. Selain keunggulan performa, *React JS* menerapkan arsitektur berbasis komponen yang memungkinkan pengembang untuk membagi antarmuka menjadi bagian-bagian kecil yang bersifat mandiri dan dapat digunakan kembali. Pendekatan modular ini tidak hanya memudahkan proses pemeliharaan kode dan pengembangan fitur baru di masa depan, tetapi juga menjamin konsistensi tampilan aplikasi saat diakses melalui berbagai *platform*, baik perangkat seluler maupun komputer. Dengan skalabilitas yang tinggi dan dukungan ekosistem yang luas, pemilihan teknologi ini memastikan sistem mampu menangani integrasi data yang kompleks dengan tetap menjaga antarmuka yang intuitif bagi pengguna (Rohi Abdulloh, 2023).

7. *Reverse Geocoding*

*(Irfan Bayu & Andini Fifi, 2023)* memaparkan bahwa *Reverse Geocoding* merupakan sebuah proses komputasi yang berfungsi untuk mengonversi koordinat geografis, yang terdiri dari titik lintang (*latitude*) dan titik bujur (*longitude*), menjadi alamat fisik yang lengkap atau deskripsi lokasi yang dapat

dipahami oleh manusia, seperti nama jalan, nomor bangunan, kelurahan, hingga kode pos. Teknologi ini merupakan kebalikan dari proses *geocoding* konvensional yang mengubah alamat tekstual menjadi titik koordinat peta. Dalam implementasi sistem berbasis lokasi, proses ini memungkinkan perangkat untuk mengidentifikasi posisi pengguna secara kontekstual tanpa mengharuskan pengguna memasukkan alamat mereka secara manual.

Mekanisme kerja *Reverse Geocoding* pada umumnya melibatkan pengambilan data koordinat melalui sensor GPS atau API *Geolocation* pada *browser*, yang kemudian dikirimkan melalui permintaan (*request*) ke sebuah basis data spasial atau penyedia layanan pemetaan. Layanan tersebut akan melakukan pencarian pada tabel data geografis untuk menemukan objek alamat terdekat yang merepresentasikan titik koordinat tersebut. Hasil yang dikembalikan biasanya berupa data terstruktur yang berisi rincian administratif wilayah, sehingga memudahkan sistem untuk mendokumentasikan lokasi kejadian atau objek secara akurat dan sistematis (Li, 2018).

Penerapan teknologi ini memberikan keuntungan signifikan dalam meningkatkan efisiensi operasional dan akurasi data lokasi. Dengan adanya fitur ini, risiko kesalahan manusia dalam menentukan titik lokasi dapat diminimalkan, terutama dalam situasi di mana ketepatan alamat menjadi faktor yang sangat krusial. Secara teknis, penggunaan *Reverse Geocoding* juga mendukung terciptanya pangkalan data yang lebih terintegrasi karena setiap entitas informasi yang masuk telah dilengkapi dengan atribut alamat yang telah tervalidasi secara otomatis oleh sistem.

8. *Leaflet.js*

*Leaflet.js* merupakan *Library JavaScript* sumber terbuka (*open-source*) yang dirancang khusus untuk membangun peta interaktif berbasis web yang responsif dan berkinerja tinggi. Dengan ukuran kode yang sangat ringan, yakni hanya sekitar 42KB, *Library* ini menawarkan efisiensi tanpa mengorbankan fungsionalitas inti yang dibutuhkan dalam pemetaan modern. Secara teknis, *Leaflet.js* bekerja dengan mengonsumsi data dari *tile server* seperti *OpenStreetMap* (OSM) sebagai landasan visual petanya. *Library* ini mendukung berbagai fitur interaksi tingkat lanjut, mulai dari penggunaan marker kustom

untuk menandai lokasi spesifik, *popup* informasi yang muncul saat elemen diklik, hingga penggunaan overlay poligon atau garis untuk memvisualisasikan area tertentu di atas peta (Leaflet JS, n.d.).

Fleksibilitasnya memungkinkan pengembang untuk memperluas kapabilitas pemetaan melalui berbagai plugin tambahan yang tersedia di komunitas pengembang secara luas. Pemilihan *Leaflet.js* sebagai komponen pemetaan utama dalam sistem didasari oleh pertimbangan strategis terkait skalabilitas dan pengelolaan anggaran (Bayu Dwi Saputra, 2022).

Berbeda dengan *Google Maps API* yang menerapkan skema penggunaan berbayar dan mewajibkan penggunaan API key dengan batasan kuota tertentu, *Leaflet.js* dapat digunakan sepenuhnya secara gratis tanpa biaya lisensi. Karakteristik ini menjadikannya solusi yang ideal bagi pengembangan sistem informasi di instansi pemerintah atau organisasi dengan anggaran terbatas, namun tetap memerlukan standar keamanan dan reliabilitas yang tinggi. Dengan mengandalkan ekosistem *OpenStreetMap*, sistem dapat beroperasi secara mandiri tanpa ketergantungan pada penyedia layanan pihak ketiga yang bersifat komersial, sehingga menjamin keberlanjutan operasional sistem dalam jangka panjang tanpa beban biaya langganan yang tidak terduga.

9. *CodeIgniter 4*

Randi Adrika Putra (2025) menyatakan bahwa *CodeIgniter 4* merupakan *Framework* pengembangan web berbasis PHP yang dirancang dengan performa tinggi namun tetap memiliki jejak memori (*footprint*) yang sangat kecil. *Framework* ini mengadopsi pola arsitektur *Model-View-Controller* (MVC) yang memungkinkan pemisahan antara logika bisnis, manajemen data, dan presentasi antarmuka, sehingga proses pengembangan menjadi lebih terstruktur dan mudah dikelola.

*CodeIgniter* 4 mendemonstrasikan performa tinggi dalam menangani permintaan REST API dengan latensi rendah, menjadikannya pilihan yang tepat untuk aplikasi yang membutuhkan respons cepat seperti sistem pelaporan kejadian darurat secara *real-time (Bhavesh et al., 2021)*.

Secara teknis, *CodeIgniter 4* berfungsi sebagai penyedia *RESTful API* yang menjembatani komunikasi data antara antarmuka pengguna dan pangkalan

data melalui pertukaran informasi yang efisien dan ringan. Selain mengelola alur data utama, *Framework* ini menyediakan lapisan keamanan yang kuat melalui fitur validasi input otomatis, proteksi terhadap serangan siber seperti *CSRF* dan *XSS*, serta manajemen basis data yang aman (Rahmat & Yung, 2025). Integrasi fitur keamanan dan pengolahan logika yang terpusat ini memastikan bahwa seluruh informasi dalam sistem pelayanan publik tetap terjaga integritasnya serta memiliki reliabilitas tinggi saat dioperasikan dalam skala luas.

*10. MySQL*

Google Cloud (2024) menjelaskan *MySQL* merupakan sistem manajemen basis data relasional *Relational Database Management System* (RDBMS) sumber terbuka (*open-source*) yang menggunakan *Structured Query Language* (SQL) sebagai standar untuk mengelola dan memanipulasi data secara terstruktur.

Dalam pengembangan sistem informasi, MySQL berperan sebagai pusat penyimpanan data yang sangat andal untuk menampung berbagai informasi krusial, mulai dari identitas pelapor, kategori insiden, hingga titik koordinat lokasi yang presisi.

Stabilitas dan performanya yang tinggi dalam menangani transaksi data bervolume besar menjadikannya pilihan utama untuk menjaga integritas informasi melalui model relasional yang menghubungkan berbagai tabel data secara efisien dan sistematis. Secara teknis, keunggulan MySQL terletak pada kemudahan integrasinya dengan berbagai *Framework Backend* serta kemampuannya dalam melakukan pengambilan data secara cepat melalui kueri yang kompleks (Yin & Cheng, 2023).

Karakteristik ini memastikan bahwa setiap laporan yang masuk dapat diakses secara instan oleh petugas terkait, seperti dinas pemadam kebakaran, guna keperluan pemantauan *real-time* maupun pengarsipan dokumentasi riwayat insiden secara akurat. Dengan dukungan fitur keamanan yang mumpuni dan skalabilitas yang baik, MySQL menyediakan fondasi data yang transparan dan akuntabel, menjamin seluruh rekaman kejadian darurat terorganisir dengan standar operasional yang profesional.

34

11. *Firebase Cloud Messaging* (FCM)

*Firebase Cloud Messaging* (FCM) adalah layanan pengiriman pesan dan notifikasi lintas *platform* yang dikembangkan oleh Google sebagai bagian dari ekosistem *Firebase*. FCM memungkinkan pengiriman *Push notification* secara andal kepada perangkat klien, baik Android, iOS, maupun *browser* web modern, tanpa memerlukan koneksi persisten antara server dan perangkat penerima. Mekanisme ini sangat relevan untuk sistem darurat yang menuntut distribusi informasi secara *real-time* kepada pengguna, bahkan saat aplikasi tidak aktif di latar depan *browser* (Google Firebase, 2026).

*12. WhatsApp Gateway*

*WhatsApp* merupakan *platform* pesan instan dengan penetrasi tertinggi di Indonesia dengan 91% pengguna aktif per 2024 menjadikannya kanal komunikasi paling familier bagi seluruh lapisan masyarakat (Maitrayee Dey, 2025). Berbeda dari *Push notification* berbasis *browser*, notifikasi *WhatsApp* muncul di layar kunci perangkat secara instan, tidak bergantung pada status *browser*, dan dapat menjangkau seluruh anggota grup sekaligus tanpa konfigurasi per-perangkat. Karakteristik ini menjadikannya kanal komplementer yang strategis untuk mendistribusikan laporan darurat kepada kelompok petugas Damkar yang terorganisasi dalam satu grup.

Terdapat tiga pendekatan teknis integrasi *WhatsApp* ke sistem berbasis web. Pertama, *WhatsApp Business* API resmi (Meta) menawarkan keandalan tertinggi namun mensyaratkan verifikasi bisnis yang panjang dan biaya berbasis volume pesan, sehingga tidak praktis untuk instansi pemerintah berskala kecil. Kedua, *Library* berbasis otomasi *browser* seperti *WhatsApp*-web.js yang menggunakan *Puppeteer* (*Chromium headless*) lebih mudah diimplementasikan, namun membutuhkan lebih dari 500 MB RAM hanya untuk satu instansi, tidak efisien pada hosting berkapasitas terbatas. Ketiga, *Library* berbasis protokol *WebSocket* langsung yang berkomunikasi ke server *WhatsApp* tanpa melibatkan *browser* sama sekali.

*Baileys* mengimplementasikan pendekatan ketiga tersebut koneksi *WebSocket* langsung ke server *WhatsApp* dengan jejak memori hanya sekitar 50 MB, jauh lebih ringan dibanding pendekatan berbasis *browser* (Baileys

Contributors, 2026). Autentikasi dilakukan sekali melalui pemindaian kode QR, kemudian sesi dipertahankan secara persisten. Dalam sistem SIRTADA, *Baileys* difungsikan sebagai *microservice* independen yang menerima permintaan pengiriman pesan melalui *endpoint* REST API dan meneruskannya ke grup *WhatsApp* Damkar, memisahkan logika notifikasi dari logika bisnis utama sehingga kegagalan pada *Gateway* tidak mempengaruhi alur pelaporan inti.

Pemilihan *Baileys* pada sistem SIRTADA didasarkan pada tiga pertimbangan:

a. Dinas Pemadam Kebakaran Kabupaten Kediri belum memiliki akun *whatsapp* Business terverifikasi sehingga API resmi Meta tidak dapat diterapkan ,
b. Keterbatasan sumber daya hosting membuat pendekatan berbasis *browser* tidak efisien , dan
c. Komunikasi petugas Damkar sudah terorganisasi dalam grup *whatsapp* aktif, sehingga notifikasi grup memungkinkan seluruh petugas menerima informasi darurat secara bersamaan tanpa pendaftaran token FCM per perangkat.

*WhatsApp Gateway* pada sistem ini berperan sebagai kanal notifikasi sekunder yang melengkapi FCM, memastikan informasi laporan darurat tetap tersampaikan bahkan ketika *browser* petugas tidak sedang aktif.

13. *JSON Web Token* (JWT)

*JSON Web Token* (JWT) adalah standar terbuka berbasis RFC 7519 yang digunakan untuk mentransmisikan informasi identitas secara aman antar pihak dalam format JSON yang ringkas. Setiap token terdiri dari tiga bagian yaitu *header*, *payload*, dan *signature* yang dienkripsi menggunakan kunci rahasia sisi server. Struktur ini memastikan integritas data, jika token dimodifikasi tanpa akses ke kunci rahasia, tanda tangan menjadi tidak valid dan permintaan akan ditolak (JWT.io, n.d.).

Keunggulan utama JWT untuk sistem berbasis web adalah sifatnya yang *stateless server* tidak perlu menyimpan data sesi di basis data, karena seluruh konteks identitas pengguna sudah terkandung dalam token itu sendiri. Pendekatan ini sangat sesuai dengan arsitektur *RESTful API* yang memisahkan

*Frontend* dan *Backend*, sekaligus meringankan beban server karena verifikasi hanya dilakukan melalui validasi kriptografis pada setiap permintaan yang masuk (Jones et al., 2015).

Aspek krusial yang melengkapi penggunaan JWT adalah strategi penyimpanan token di sisi klien (*client-side token storage strategy*). Terdapat dua pendekatan utama yang umum digunakan. Pendekatan pertama menyimpan JWT di *localStorage* atau *sessionStorage browser*, yang mudah diimplementasikan namun rentan terhadap serangan *Cross-Site Scripting* (XSS) karena token dapat dibaca oleh JavaScript yang berjalan di halaman yang sama. Pendekatan kedua menyimpan JWT di dalam *httpOnly cookie*, yaitu *cookie* yang secara eksplisit dikonfigurasi agar tidak dapat diakses oleh JavaScript sisi klien sama sekali. Atribut *HttpOnly* pada *cookie* menjadikan token hanya dapat dibaca oleh server melalui header HTTP, sehingga serangan XSS yang berhasil menyuntikkan skrip berbahaya ke halaman pun tidak dapat mencuri token autentikasi (Darmawan et al., 2021). *Browser* akan menyertakan *cookie* ini secara otomatis pada setiap permintaan ke domain yang sama, sehingga dari perspektif developer tidak diperlukan pengelolaan token secara manual di kode JavaScript.

83

Untuk memperkuat perlindungan terhadap serangan *Cross-Site Request Forgery* (CSRF), *httpOnly cookie* yang menyimpan JWT dikombinasikan dengan atribut *SameSite=Strict*. Atribut *SameSite=Strict* menginstruksikan *browser* untuk tidak pernah menyertakan *cookie* pada permintaan lintas situs (*cross-site request*), sehingga situs berbahaya yang memancing pengguna untuk mengirim permintaan ke server target tidak akan mendapatkan token yang valid meskipun korban sedang dalam keadaan terautentikasi (Hande et al., 2024). Kombinasi atribut *HttpOnly* dan *SameSite=Strict* pada *cookie* penyimpan JWT menjadikan lapisan keamanan autentikasi pada sistem SIRTADA memenuhi rekomendasi keamanan aplikasi web modern, khususnya untuk sistem layanan publik yang menangani data darurat sensitif dan menuntut tingkat kepercayaan tinggi dari penggunanya.

14. *Unified Modeling Language* (UML)

Ahmat Adil et al. (n.d.) menjelaskan *Unified Modeling Language* (UML) merupakan bahasa standar pemodelan industri yang digunakan untuk memvisualisasikan, merancang, dan mendokumentasikan spesifikasi dari sebuah sistem perangkat lunak secara grafis.

UML bertindak sebagai cetak biru (*blueprint*) yang memungkinkan pengembang untuk memetakan arsitektur sistem yang kompleks menjadi serangkaian diagram yang lebih sederhana namun tetap terperinci. Keberadaannya sangat krusial dalam menjembatani komunikasi antara berbagai pihak yang terlibat, mulai dari analis hingga pengembang, guna memastikan bahwa setiap kebutuhan fungsionalitas sistem dipahami secara seragam tanpa adanya ambiguitas teknis (Phillips, 2010).

Secara operasional, UML menyediakan berbagai jenis diagram yang merepresentasikan struktur dan perilaku sistem secara menyeluruh, seperti *Use Case Diagram* untuk memetakan interaksi pengguna. Melalui pemodelan visual ini, pengembang dapat mengidentifikasi potensi kesalahan desain sejak tahap awal serta memastikan bahwa setiap komponen sistem saling terintegrasi dengan optimal. Dokumentasi yang dihasilkan melalui UML tidak hanya mempermudah proses konstruksi aplikasi, tetapi juga menjadi instrumen penting dalam fase pemeliharaan dan pengembangan sistem jangka panjang agar tetap adaptif terhadap kebutuhan yang terus berkembang (Debbabi et al., 2010).

15. Strategi Pengujian Perangkat Lunak

Strategi pengujian perangkat lunak merupakan sebuah pendekatan menyeluruh yang mengintegrasikan berbagai metode dan tingkatan pengujian ke dalam suatu perencanaan yang terstruktur, dengan tujuan untuk memastikan bahwa perangkat lunak yang dibangun bebas dari kesalahan serta memenuhi seluruh spesifikasi kebutuhan fungsional maupun non-fungsional (Joshi & Kumari, 2022).

Pengujian perangkat lunak tidak hanya berfokus pada fungsionalitas akhir di tingkat antarmuka, melainkan melibatkan serangkaian tahapan pengujian yang komprehensif mulai dari tingkat komponen terkecil hingga pengujian

karakteristik sistem secara keseluruhan seperti kinerja dan keamanan (Divyani Shivkumar Taley, 2020).

Dalam pengembangan *Emergency System* berbasis web (SIRTADA) ini, strategi pengujian diterapkan yang mencakup pilar-pilar utama, yaitu *Blackbox Testing*, *Unit Testing*, *Integration testing*, *Performance Testing*, dan *Security Testing*.

a. *Blackbox Testing*

*Blackbox testing* ialah metode pengujian perangkat lunak yang berfokus sepenuhnya pada aspek fungsionalitas tanpa memerlukan pemahaman mendalam mengenai struktur kode, desain internal, atau logika pemrosesan di dalam sistem. Dalam pendekatan ini, perangkat lunak diperlakukan sebagai sebuah entitas tertutup di mana penguji hanya perlu mengevaluasi respons sistem berdasarkan input yang diberikan. Tujuannya adalah untuk memastikan bahwa setiap fitur dapat menghasilkan output yang tepat dan konsisten sesuai dengan spesifikasi kebutuhan yang telah didefinisikan sebelumnya dari perspektif pengguna (Ostrand, 2002).

b. *Unit Testing*

*Unit testing* adalah tingkatan pengujian perangkat lunak di mana komponen atau blok kode terkecil dari aplikasi diuji secara terisolasi untuk memastikan bahwa logika internalnya berfungsi dengan benar. Unit terkecil yang dimaksud dapat berupa sebuah fungsi, metode, kelas, atau modul individual. Tujuan utama dari pengujian unit adalah memverifikasi kebenaran algoritma dan penanganan data internal dengan mengisolasi setiap bagian kode guna membuktikan bahwa bagian tersebut benar secara mandiri sebelum digabungkan (Hagos, 2023).

c. *Integration testing*

*Integration testing* adalah tahap pengujian yang dilakukan untuk mengevaluasi interaksi, komunikasi, dan aliran data antar komponen atau subsistem yang berbeda yang telah diintegrasikan. Pengujian ini berfokus pada pembuktian bahwa unit-unit kode yang telah lulus uji mandiri dapat bekerja sama secara harmonis saat saling berhubungan. Alur pengujian integrasi sangat

penting untuk mendeteksi galat yang muncul akibat ketidakcocokan parameter input/output, kegagalan pertukaran data melalui REST API, hingga kendala konektivitas antar subsistem internal maupun subsistem eksternal (Winter, 2013).

*d. Performance Testing*

*Performance testing* merupakan jenis pengujian non-fungsional yang dilaksanakan untuk mengukur karakteristik kecepatan, stabilitas, latensi, dan efisiensi sebuah sistem perangkat lunak di bawah kondisi beban kerja tertentu (Pargaonkar, 2023).

*e. Security Testing*

*Security testing* adalah proses pengujian yang bertujuan untuk mengidentifikasi celah keamanan, kelemahan sistem, dan potensi kerentanan yang dapat dieksploitasi oleh pihak tidak berwenang untuk merusak sistem atau memanipulasi data sensitif. Pada aplikasi pelayanan publik berbasis web, pengujian keamanan wajib dilakukan untuk memastikan aspek kerahasiaan, integritas data, dan ketersediaan layanan (Erdogan G, 2009).

**f.** *User Acceptance Testing* (UAT)

*User Acceptance Testing* adalah metode pengujian sistem yang dilakukan oleh pengguna akhir untuk memverifikasi bahwa sistem yang dikembangkan memenuhi kebutuhan dan persyaratan fungsional yang telah ditetapkan (Cimperman, 2006).

UAT memberikan keyakinan kepada pemangku kepentingan bahwa sistem siap diterapkan secara operasional, karena pengujian dilakukan berdasarkan perspektif pengguna yang akan mengoperasikan sistem sehari-hari.

Dalam penelitian ini, UAT dilaksanakan menggunakan instrumen kuesioner dengan skala Likert lima tingkat, di mana skor 1 mewakili penilaian Sangat Tidak Setuju (STS) dan skor 5 mewakili Sangat Setuju (SS). Persentase penerimaan dihitung dengan rumus: Persentase = (Total Skor / Skor Maksimal) × 100%. Hasil persentase diinterpretasikan menggunakan tabel kategori kelayakan: 81–100% (Sangat Baik), 61–80% (Baik), 41–60% (Cukup), 21–40% (Kurang), dan 0–20% (Sangat Kurang).

## B. Kajian Hasil Penelitian Terdahulu

Berikut ini adalah rangkuman dari berbagai penelitian sebelumnya yang menjadi dasar dan acuan dalam kajian ini, yang akan membantu memahami konteks dan perkembangan topik secara lebih mendalam, melalui kajian hasil penelitian terdahulu yang dilakukan oleh:

(Indriati et al., 2024) dalam penelitiannya yang berjudul "Pengembangan Sistem Absensi Berbasis Website Menggunakan Metode Rapid Application Development" membuktikan bahwa efisiensi dan akurasi administrasi dapat ditingkatkan signifikan melalui transformasi proses manual menjadi digital. Konsep transformasi digital ini relevan dengan penelitian ini karena keduanya mengutamakan integrasi data yang cepat. Namun, jika sistem absensi tersebut ditujukan untuk pengelolaan data rutin sekolah, penelitian ini mengadaptasi kecepatan teknologi tersebut untuk menangani situasi darurat yang membutuhkan respons segera di lapangan.

Rini Indriati et al. (2022) pada penelitiannya yang berjudul "Sistem Informasi Izin Online Berbasis Web Menggunakan *Framework CodeIgniter*" mengembangkan sebuah sistem izin pegawai secara daring dengan memanfaatkan *CodeIgniter*, MySQL, dan metode *Waterfall*. Sistem tersebut juga menggunakan *WhatsApp* API untuk mengirimkan notifikasi secara otomatis. Hasil penelitian menunjukkan bahwa sistem ini mampu meningkatkan kecepatan dan ketepatan proses izin, sehingga cocok sebagai referensi dalam pengembangan *Emergency System* berbasis web yang terintegrasi dan responsif. Namun langkah pengembangan *Emergency System* ini memiliki cakupan yang lebih dinamis karena ditujukan untuk interaksi langsung antara masyarakat dan petugas saat terjadi insiden.

Kerentanan pada sistem pencatatan manual juga menjadi sorotan dalam penelitian (Indriati et al., 2023) Melalui studinya yang berjudul "Aplikasi Web Untuk Pembayaran Biaya Administrasi Sekolah Dasar Putren 1", dijelaskan bahwa proses konvensional sering kali menghambat efisiensi. Penelitian ini membuktikan bahwa migrasi ke *platform* web tidak hanya mempercepat proses, tetapi juga menjamin akurasi melalui validasi data otomatis. Prinsip validasi inilah

yang kemudian diadaptasi dalam sistem ini, namun dengan pergeseran fokus: dari validasi transaksi keuangan menjadi validasi titik koordinat lokasi darurat guna menjamin ketepatan respons petugas di lapangan.".

Pemanfaatan *React*.js sebagai pilar utama antarmuka pengguna juga menjadi fokus dalam studi yang dilakukan oleh (Akhdani, 2022). Melalui penelitiannya yang berjudul "Implementasi *React*.js pada Pengembangan *Frontend* Sistem Informasi Manajemen", ia membuktikan bahwa penggunaan metode Agile yang dipadukan dengan *React*.js mampu menghasilkan antarmuka yang jauh lebih interaktif dan efisien. Meskipun terdapat kesamaan dalam penggunaan *Framework Frontend*, sistem yang tengah dikembangkan saat ini membawa teknologi tersebut ke arah yang lebih spesifik, yakni untuk membangun *dashboard* monitoring yang mampu menyajikan data bencana secara *real-time*.

Hasanah et al. (2024) menjelaskan pada penelitannya yang berjudul "Sistem Informasi Absensi Kehadiran Siswa Berbasis Geolokasi di SMK Nurul Islam Sempolan Jember" merancang sistem absensi berbasis PHP, MySQL, dan *CodeIgniter* 4 dengan metode *Waterfall*. Sistem ini meningkatkan efisiensi dan mengurangi kesalahan absensi melalui validasi lokasi otomatis. Dalam konsep sistem pada penelitian, fungsi geolokasi tersebut akan menjadi instrumen navigasi dan pemetaan titik koordinat bencana guna meminimalkan ketidakteraturan informasi lokasi saat kondisi darurat.

Efektivitas pengelolaan data terpusat juga menjadi perhatian dalam studi yang dilakukan oleh (Ilma Rimbarawa et al., 2022). Melalui penelitiannya berjudul "Pengembangan Database Sistem Informasi Jalur Kereta Berbasis Web Menggunakan MySQL", mereka menunjukkan bahwa integrasi jadwal dan rute kereta api menjadi lebih akurat berkat penggunaan MySQL sebagai pusat basis data. Meskipun penelitian saat ini memiliki kesamaan dalam hal pemanfaatan arsitektur web dan MySQL, arah pengembangannya digeser untuk menangani urgensi layanan publik di situasi darurat. Selain itu, sistem yang sedang dibangun ini dikembangkan lebih jauh dengan kolaborasi *Framework React*.js dan *CodeIgniter* 4 guna menjamin performa yang lebih responsif dan modern.

Dalam penelitiannya Sharyanto et al. (2022) menjelaskan pada studi penelitian berjudul "Rancang Bangun Sistem Tanggap Darurat Berbasis Web Suku

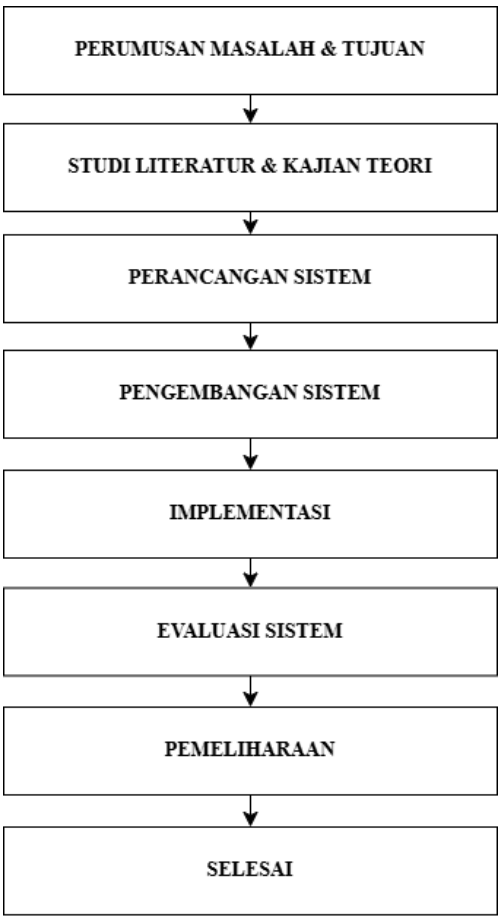
Dinas Penanggulangan Kebakaran dan Penyelamatan Jakarta Utara". Fokus utama penelitian tersebut adalah mengatasi keterlambatan penanganan kebakaran akibat minimnya sarana pengaduan yang efisien dan seringnya terjadi miskomunikasi melalui call center. Perancangan sistem ini menggunakan metode Unified Modelling Language (UML) yang mencakup use case, activity, dan class diagram, serta dibangun menggunakan bahasa pemrograman PHP dan basis data MySQL. Hasil pengujian melalui metode Black Box *Testing* menunjukkan bahwa sistem berbasis web ini mampu mengubah kebiasaan masyarakat dari pelaporan manual menjadi lebih terstruktur, sehingga meningkatkan efisiensi proses pendataan dan pelaporan bagi petugas. Sejalan dengan penelitian yang dilakukan terletak pada pengembangan sistem informasi tanggap darurat (emergency system) berbasis web untuk instansi pemadam kebakaran. Namun, pada penelitian ini, pengembangan ditekankan pada penggunaan tech stack modern seperti *Framework React*.js dan *CodeIgniter* 4 untuk meningkatkan responsivitas antarmuka serta performa sistem.

Lugis Purwanto (2023) dalam penelitiannya yang berjudul "Penerapan *Push notification* pada Sistem Monitoring Belajar Berbasis Web Menggunakan *Firebase* Cloud Messaging" mengkaji urgensi digitalisasi pemantauan progres belajar guna mengatasi hambatan pelaporan konvensional yang sering mengalami kendala keterlambatan informasi. Penelitian tersebut secara teknis mengimplementasikan *Firebase Cloud Messaging* (FCM) sebagai infrastruktur pengiriman *Push notification real-time* ke *browser* pengguna, mencakup proses pendaftaran Service Worker, pengelolaan FCM Registration Token, serta pengiriman pesan notifikasi bertarget dari sisi server. Hasil penelitian membuktikan bahwa FCM mampu mendistribusikan notifikasi secara akurat dan tepat waktu kepada pengguna yang dituju. Penelitian ini menjadi acuan teknis yang paling langsung relevan karena sistem yang dikembangkan dalam penelitian ini juga mengimplementasikan FCM sebagai mekanisme inti distribusi *Push notification*, dengan adaptasi yang lebih kompleks pada konteks manajemen insiden darurat. notifikasi FCM dikirimkan secara bertarget kepada petugas Dinas Pemadam Kebakaran Kabupaten Kediri saat laporan baru masuk, serta kepada masyarakat pelapor saat terjadi pembaruan status penanganan insiden mereka,

sehingga seluruh rantai komunikasi darurat dapat berjalan secara otomatis, cepat, dan terdokumentasi tanpa bergantung pada *platform* komunikasi eksternal.

## C. Kerangka Berpikir

Kerangka berpikir dalam penelitian ini disusun sebagai panduan sistematis untuk menyelesaikan kompleksitas permasalahan pelaporan darurat melalui pendekatan teknologi informasi. Alur pemikiran ini menggambarkan transformasi dari identifikasi kendala di lapangan hingga terciptanya solusi digital yang teruji. Secara visual, tahapan-tahapan tersebut dipetakan dalam gambar berikut:




Gambar 2. 1 Kerangka Berpikir

1. Perumusan Masalah & Tujuan

   Kerangka berpikir penelitian ini diawali dari kondisi Dinas Pemadam Kebakaran Kabupaten Kediri yang masih menggunakan *WhatsApp* untuk pelaporan dan *Microsoft Excel* untuk pencatatan laporan. Proses tersebut menyebabkan laporan tidak terstruktur, lokasi kejadian kurang akurat, serta dokumentasi insiden belum tersimpan dengan baik. Berdasarkan permasalahan tersebut, penelitian ini bertujuan membangun *Emergency System* berbasis web menggunakan *React*.js, *CodeIgniter* 4, dan MySQL

untuk mempercepat pelaporan, meningkatkan akurasi lokasi, serta mempermudah koordinasi antara masyarakat, petugas, dan admin.

2. Studi Literatur dan Kajian Teori

   Tahap ini dilakukan untuk memperkuat dasar penelitian melalui kajian teori dan penelitian terdahulu. Kajian teori meliputi Sistem Informasi, Emergency System, aplikasi web, *React*.js, *CodeIgniter* 4, MySQL, Leaflet.js, JWT, RBAC, *Firebase* Cloud Messaging (FCM), dan *WhatsApp Gateway*. Selain itu, dilakukan analisis beberapa penelitian sebelumnya yang berkaitan dengan sistem geolokasi, sistem tanggap darurat, *React*.js, dan notifikasi *real-time*. Tahap ini bertujuan mendukung pemilihan teknologi, metode pengembangan, serta menemukan perbedaan penelitian yang akan dikembangkan.

3. Perancangan Sistem

   Tahap perancangan dilakukan berdasarkan hasil wawancara dan analisis kebutuhan pengguna. Perancangan meliputi desain arsitektur *Frontend* dan *Backend* berbasis REST API, serta pembuatan Context Diagram, DFD, *Use Case Diagram*, dan ERD. Pada tahap ini juga dirancang fitur utama sistem seperti *panic button*, pelaporan dengan geolokasi otomatis, *dashboard* pemantauan, pengelolaan status laporan, notifikasi otomatis, dan pengaturan hak akses pengguna menggunakan JWT dan RBAC.

4. Pengembangan Sistem

   Tahap pengembangan merupakan proses pembuatan sistem berdasarkan rancangan yang telah dibuat. *Frontend* dikembangkan menggunakan *React*.js dan Tailwind CSS untuk menghasilkan tampilan yang responsif dan mudah digunakan. *Backend* dibangun menggunakan *CodeIgniter* 4 sebagai penyedia REST API dan pengelola data sistem. Basis data MySQL digunakan untuk menyimpan data pengguna, laporan, lokasi, riwayat status, dan token notifikasi. Seluruh proses pengembangan dilakukan mengikuti tahapan metode *Waterfall*.

5. Implementasi

   Tahap implementasi dilakukan dengan menerapkan sistem pada server dan menghubungkan seluruh komponen sistem, termasuk fitur notifikasi otomatis.

Masyarakat dapat melakukan pelaporan melalui perangkat komputer maupun ponsel tanpa instalasi tambahan. Sementara itu, petugas dan admin menggunakan *dashboard* untuk memantau laporan masuk, menerima notifikasi, melihat lokasi kejadian melalui peta interaktif berbasis *Leaflet.js* dan *OpenStreetMap,* serta membuka navigasi rute eksternal via *Google Maps*.

6. Evaluasi Sistem

   Evaluasi dilakukan menggunakan strategi pengujian *multi-level* yang mencakup *Unit Testing, Integration testing, Performance Testing, Security Testing, Blackbox Testing,* dan *User Acceptance Testing* (UAT). Pengujian dilakukan pada fitur pelaporan, geolokasi, *panic button*, notifikasi, perubahan status laporan, dan keamanan akses sistem. Selain itu, dilakukan pengujian performa untuk memastikan proses pengiriman laporan dan notifikasi dapat berjalan dengan cepat sesuai kebutuhan sistem darurat.

7. Pemeliharaan

   Tahap pemeliharaan dilakukan setelah sistem diimplementasikan. Kegiatan pemeliharaan meliputi perbaikan *Bug* dan *Error*.

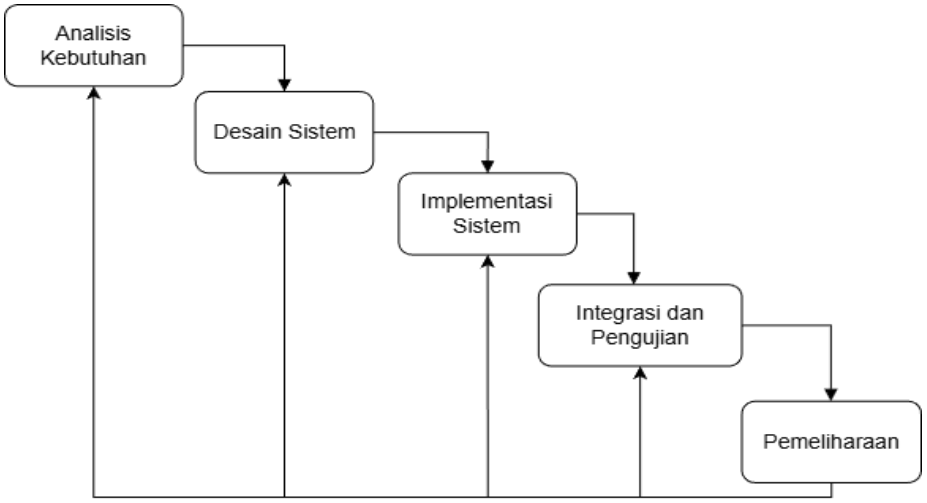
8. Selesai

   Penelitian dinyatakan selesai setelah *Emergency System* berbasis web berhasil dirancang, dikembangkan, diuji, dan diimplementasikan pada Dinas Pemadam Kebakaran Kabupaten Kediri. Sistem yang dihasilkan diharapkan mampu menggantikan proses pelaporan manual menjadi lebih terstruktur, cepat, dan terdokumentasi, serta membantu meningkatkan kecepatan respons dan akurasi informasi lokasi kejadian.

# BAB III
# METODE PENELITIAN

Metode penelitian merupakan langkah-langkah sistematis dan terstruktur yang digunakan oleh peneliti untuk memecahkan masalah atau menjawab pertanyaan penelitian secara ilmiah (Amalia et al., 2024). Dalam konteks perancangan sistem informasi, metode penelitian berfungsi sebagai kerangka kerja yang memandu seluruh proses, mulai dari identifikasi kebutuhan pengguna hingga tahap evaluasi akhir. Tujuan utama dari penggunaan metode ini adalah untuk memastikan bahwa pengembangan *Emergency System* berjalan secara logis, efisien, dan menghasilkan sistem yang valid sesuai dengan spesifikasi yang telah ditetapkan bagi Dinas Pemadam Kebakaran Kabupaten Kediri.

## A. Metode *Waterfall*

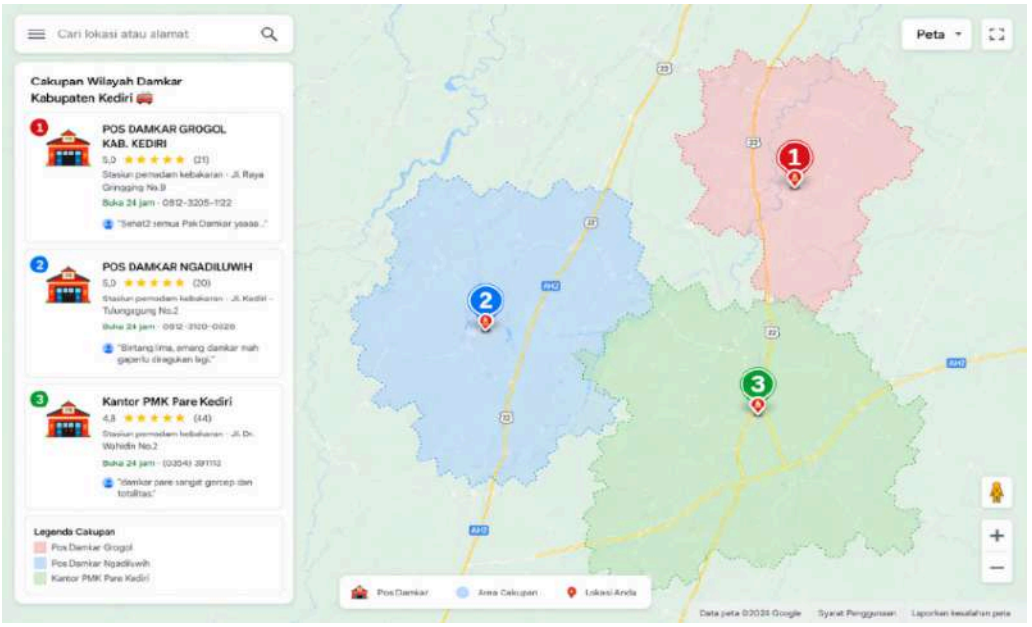



Gambar 3. 1 Metode *Waterfall*

Metode *Waterfall* dipilih sebagai pendekatan pengembangan sistem ini karena ruang lingkup dan kebutuhan fungsional sistem telah dapat didefinisikan secara lengkap sejak awal melalui wawancara langsung dengan koordinator lapangan Dinas Pemadam Kebakaran Kabupaten Kediri. Kondisi ini sesuai dengan karakteristik metode *Waterfall* yang bekerja paling efektif ketika spesifikasi sistem bersifat stabil. Dengan karakteristik tersebut, setiap fase dapat diselesaikan secara tuntas sebelum melangkah ke fase berikutnya, sehingga risiko perubahan kebutuhan di tengah proses dapat diminimalkan.

B. Tempat Penelitian

Penelitian ini dilakukan pada Dinas Pemadam Kebakaran Kabupaten Kediri yang memiliki beberapa pos pelayanan di wilayah Kabupaten Kediri. Pengambilan data dan observasi dilakukan melalui wawancara dengan koordinator lapangan untuk memperoleh informasi mengenai proses pelaporan dan penanganan insiden yang berjalan saat ini.




Gambar 3. 2 Cakupan Wilayah Penelitian

Lokasi penelitian mencakup tiga pos pemadam kebakaran wilayah utama, yaitu:

1. Pos Wilayah Pare
2. Pos Wilayah Grogol
3. Pos Wilayah Ngadiluwih

Ketiga lokasi tersebut dipilih untuk mewakili cakupan pelayanan pemadam kebakaran di wilayah timur, barat, dan selatan Kabupaten Kediri. Penelitian dilakukan untuk memahami proses koordinasi laporan darurat dan kebutuhan sistem dalam mendukung pelayanan di lapangan.

C. Analisis Kebutuhan (*Requirement Analysis*)

1. Studi Kebutuhan Pengguna

Berdasarkan hasil wawancara dengan koordinator lapangan Dinas Pemadam Kebakaran Kabupaten Kediri, proses pelaporan yang berjalan masih menggunakan *WhatsApp* dan pencatatan manual menggunakan *Microsoft Excel*. Proses tersebut dinilai kurang efektif karena laporan sering tidak terstruktur, lokasi kejadian kurang akurat, dan monitoring laporan masih dilakukan secara manual. Berdasarkan hasil analisis, kebutuhan pengguna pada sistem dibagi menjadi tiga aktor utama, yaitu:

a. Masyarakat

Masyarakat membutuhkan sistem yang dapat digunakan untuk:

1) Melakukan pelaporan kejadian darurat secara cepat.
2) Mengirim lokasi kejadian secara otomatis.
3) Mengirim foto kejadian sebagai pendukung laporan.
4) Menggunakan fitur *panic button*.
5) Melihat status penanganan laporan.

b. Petugas

Petugas membutuhkan sistem yang dapat digunakan untuk:

1) Menerima laporan darurat secara *real-time*.
2) Melihat detail laporan dan lokasi kejadian.
3) Memperbarui status penanganan insiden.
4) Melihat riwayat laporan dan dokumentasi kejadian.
5) Menerima notifikasi laporan baru.

c. Admin

Admin membutuhkan sistem yang dapat digunakan untuk:

1) Mengelola data pengguna.
2) Mengelola data laporan dan kategori insiden.
3) Memantau seluruh aktivitas sistem.
4) Mengatur hak akses pengguna.
5) Mengelola data notifikasi dan riwayat laporan.

2. Kebutuhan Fungsional Sistem

Kebutuhan fungsional mendefinisikan layanan yang harus disediakan sistem sebagai respons terhadap masukan pengguna maupun sistem lain yang berinteraksi dengannya (Larasati & Widiono, 2025). Seluruh kebutuhan fungsional dirangkum pada Tabel 3.1.

Tabel 3. 1 Kebutuhan Fungsional

| No | Kebutuhan Fungsional |
|---|---|
| 1 | Sistem dapat melakukan login dan autentikasi pengguna |
| 2 | Sistem dapat membedakan hak akses berdasarkan peran pengguna |
| 3 | Sistem dapat menerima laporan darurat dari masyarakat |
| 4 | Sistem dapat mendeteksi lokasi pengguna |
| 5 | Sistem dapat menampilkan lokasi laporan pada peta |
| 6 | Sistem dapat mengirim notifikasi kepada petugas |
| 7 | Sistem dapat memperbarui status laporan |
| 8 | Sistem dapat menampilkan *dashboard* monitoring laporan |
| 9 | Sistem dapat menyimpan riwayat aktivitas laporan |
| 10 | Sistem dapat mengelola data pengguna |

3. Kebutuhan Non Fungsional Sistem

Kebutuhan non-fungsional mendefinisikan batasan dan standar kualitas yang harus dipenuhi sistem di luar fungsionalitas utamanya, mencakup aspek performa, keamanan, ketersediaan, dan kemudahan penggunaan (Larasati & Widiono, 2025). Seluruh parameter kebutuhan non-fungsional dirangkum pada tabel 3.2

Tabel 3. 2 Kebutuhan Non-Fungsional

| No | Aspek | Parameter | Standar / Target |
|---|---|---|---|
| 1 | *Availability* | Ketersediaan sistem | 24 jam/hari, 7 hari/minggu |
| 2 | *Performance* | *Response time* pengiriman laporan | ≤ 15 detik |
| 3 | *Performance* | Latensi notifikasi *real-time* | ≤ 8 detik |
| 4 | *Performance* | *Page load time* halaman utama | ≤ 3 detik |
| 5 | *Performance* | Kemampuan menerima laporan pengguna secara serentak | 50 pengguna simultan |
| 6 | *Usability* | Antarmuka sistem mudah digunakan oleh masyarakat dan petugas | *Responsive & Mobile-first* |
| 7 | *Security* | Proteksi data API | Bebas dari *SQL Injection* & XSS |
| 8 | *Security* | Hak akses pengguna dibatasi menggunakan RBAC | Bebas dari penyalahgunaan akses |

| | | | |
|---|---|---|---|
| 9 | *Security* | Keamanan unggahan *file* | Bebas dari eksekusi file berbahaya |

4. Spesifikasi Kebutuhan Sistem

Spesifikasi kebutuhan sistem mendefinisikan seluruh komponen perangkat keras dan perangkat lunak yang diperlukan sebagai lingkungan pengembangan dan operasional sistem agar dapat berjalan sesuai dengan standar performa dan keandalan yang telah ditetapkan. Seluruh spesifikasi kebutuhan sistem adalah sebagai berikut:

a. Kebutuhan Perangkat Keras

Tabel 3. 3 Spesifikasi *Hardware*

| No | Perangkat | Spesifikasi |
|---|---|---|
| 1 | Processor | Minimal Dual Core |
| 2 | RAM | Minimal 4 GB |
| 3 | Penyimpanan | SSD Minimal 120 GB |
| 4 | Smartphone | Android dengan GPS aktif |

b. Kebutuhan Perangkat Lunak

Tabel 3. 4 Spesifikasi *Software*

| No | Perangkat Lunak | Keterangan |
|---|---|---|
| 1 | *React*.js | *Frontend* sistem |
| 2 | *CodeIgniter* 4 | *Backend* sistem |
| 3 | MySQL | Basis data |
| 4 | Visual Studio Code | Code editor |
| 5 | Laragon | Web server lokal |
| 6 | Google Chrome | *Browser* pengujian |

c. Kebutuhan Pengguna

Tabel 3. 5 Kebutuhan Pengguna Pada Sistem

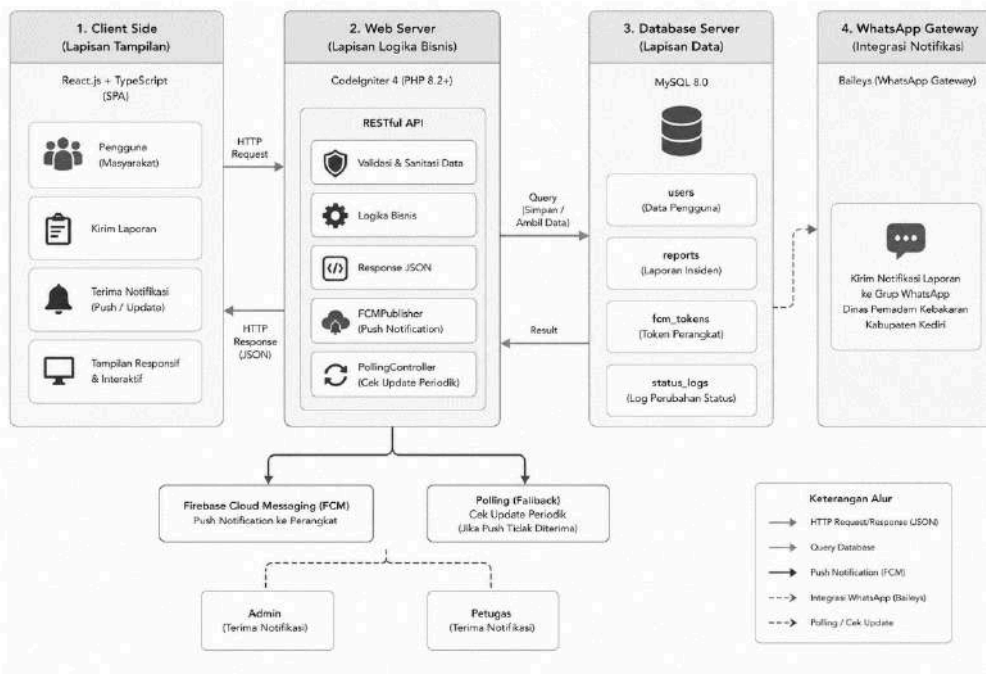
| No | Pengguna | Kebutuhan |
|---|---|---|
| 1 | Masyarakat | Melakukan pelaporan dan melihat status laporan |
| 2 | Petugas | Mengelola dan menangani laporan |
| 3 | Admin | Mengelola sistem dan pengguna |

## D. Desain Sistem (*System Design*)

Tahap desain sistem adalah fase krusial untuk mentransformasikan kebutuhan analisis menjadi spesifikasi teknis yang konkret. Tujuannya adalah memberikan gambaran menyeluruh mengenai arsitektur, alur data, basis data, dan antarmuka agar proses implementasi kode berjalan terarah, terukur, serta minim risiko kesalahan logika.

### 1. Desain Arsitektur Sistem

*Emergency System* dirancang menggunakan pendekatan arsitektur tiga lapisan (*three-tier architecture*) yang memisahkan tanggung jawab sistem ke dalam tiga lapisan independen yaitu lapisan tampilan antarmuka, lapisan logika bisnis, dan lapisan data serta integrasi *WhatsApp Gateway* sebagai notifikasi instan.




Gambar 3. 3 Desain Arsitektur Sistem

Dari gambar 3.3 desain arsitektur sistem terbagi menjadi 3 lapisan yaitu:

a. Lapisan Tampilan (*Client Side*) menggunakan *React.js* & *TypeScript* untuk membangun antarmuka pengguna yang *responsive*. Menggunakan konsep *Single-Page Application* (SPA) sehingga perpindahan halaman terasa cepat dan mulus tanpa perlu memuat ulang seluruh halaman, bahkan saat koneksi internet lambat.

b. Lapisan logika bisnis menggunakan *CodeIgniter 4* sebagai komponen pemrosesan utama yang bertugas menerima permintaan dari lapisan tampilan, memvalidasi data, menjalankan proses bisnis, lalu mengirim balik hasilnya dalam format *JSON* (RESTful API). Lapisan ini juga memiliki dua komponen penting yaitu:
   1) FCMPublisher: berfungsi mengirim notifikasi otomatis ke perangkat pengguna via *Firebase*
   2) PollingController: menyediakan cara alternatif bagi klien untuk mengecek pembaruan data secara berkala

c. Lapisan Data (*Database Server*) menggunakan MySQL 8.0 sebagai tempat penyimpanan seluruh data sistem, seperti data pengguna, laporan insiden, dan riwayat perubahan status.

d. Integrasi *WhatsApp Gateway* menggunakan *Baileys* yang berfungsi sebagai notifikasi laporan darurat dikirimkan ke grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri.

2. Desain Proses

Desain proses bertujuan untuk memvisualisasikan secara logis bagaimana data mengalir di dalam sistem dan bagaimana setiap aktor berinteraksi dengan fitur-fitur yang tersedia pada setiap skenario penggunaan.

Pemodelan ini menggunakan dua pendekatan yang saling melengkapi, *Data Flow Diagram* (DFD) untuk memetakan aliran data antar proses, entitas eksternal, dan penyimpanan data dalam sistem serta *Use Case Diagram* yang merupakan bagian dari standar *Unified Modeling Language* (UML) untuk memetakan interaksi antara setiap aktor dengan fungsi-fungsi yang tersedia.

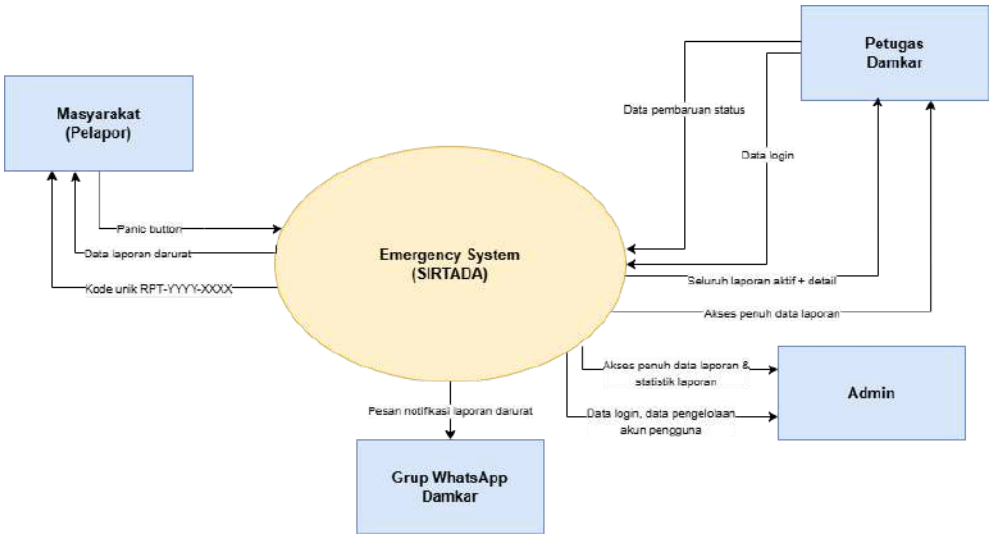
*a. Data Flow Diagram (DFD)*

*Data Flow Diagram* adalah ilustrasi aristektur grafis yang menggambarkan aliran data dari sumber ke tujuan melalui serangkaian proses transformasi, tanpa memperhatikan detail teknis implementasi atau urutan waktu kejadian.

1) DFD Level 0 (*Context Diagram*)

DFD Level 0 atau *Context Diagram* menampilkan *Emergency System* sebagai satu kesatuan sistem tunggal dengan tiga entitas eksternal yang

berinteraksi langsung dengan sistem, yaitu Masyarakat sebagai pelapor, Petugas Damkar, dan Admin.




Gambar 3. 4 Dfd Level 0 *(Context Diagram)*

Masyarakat memberikan masukan data kepada sistem berupa data laporan darurat nama, nomor hp, deskripsi kejadian, koordinat lokasi, dan foto bukti. Sebagai balasan, sistem mengembalikan konfirmasi laporan berupa kode unik RPT-YYYY-XXXX yang berisi informasi status laporan terkini dengan pembaruan status secara *real-time*.

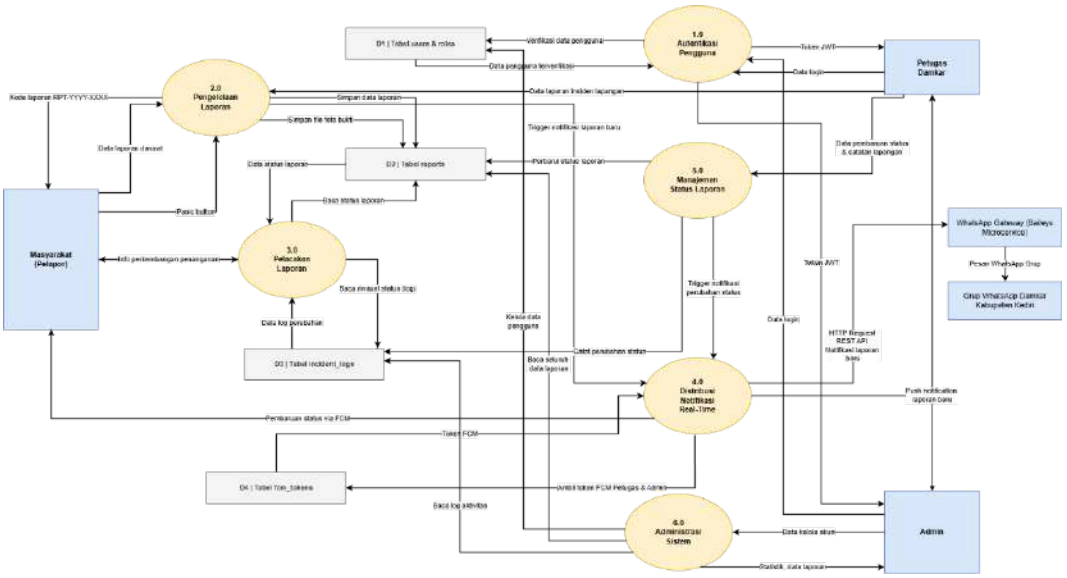
Petugas Damkar memberikan masukan berupa data login dan data pembaruan status penanganan laporan beserta catatannya. Sebagai balasan, sistem mengembalikan informasi seluruh laporan aktif dengan detail, notifikasi laporan baru secara *real-time* via *Firebase Cloud Messaging*, dan rute navigasi menuju lokasi kejadian.

Admin memberikan masukan berupa data login, data pengelolaan akun pengguna, dan parameter ekspor laporan. Sebagai balasan, sistem mengembalikan akses penuh ke seluruh data laporan, statistik laporan, serta berkas ekspor dalam format PDF maupun Excel.

Sistem mengirimkan notifikasi laporan baru secara otomatis kepada Grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri melalui *WhatsApp Gateway* berbasis *Baileys* yang beroperasi sebagai *microservice* independen, memastikan seluruh anggota grup menerima informasi darurat tanpa perlu terdaftar token FCM.

2) DFD Level 1

DFD Level 1 menjabarkan *Emergency System* menjadi enam proses utama yang saling terhubung melalui aliran data dan mengakses empat *data store* yang tersedia.




66

Gambar 3. 5 Dfd Level 1

Proses 1.0 adalah Autentikasi Pengguna, yang menerima data login dari Petugas dan Admin, memverifikasinya terhadap data store D1 (tabel users), lalu menghasilkan token JWT yang dikirimkan kembali kepada pengguna sebagai bukti autentikasi yang valid.

Proses 2.0 adalah Pengelolaan Laporan, yang menerima data laporan darurat dari masyarakat maupun petugas saat dilapangan, Petugas dapat membuat laporan melalui jalur normal dan masyarakat melaporan via *panic button*, memproses validasi data, menyimpan laporan ke data store D2 (tabel reports), menghasilkan kode laporan unik RPT-YYYY-XXXX, dan meneruskan notifikasi laporan baru ke Proses 4.0 untuk didistribusikan secara *real-time*.

Proses 3.0 adalah Pelacakan Laporan, yang menerima kode laporan dari Masyarakat, membaca data status terkini dari data store D2 tabel *reports* dan data log perubahan dari data store D3 tabel *incident_logs*, lalu mengembalikan informasi perkembangan penanganan kepada Masyarakat tanpa memerlukan autentikasi.

Proses 4.0 mendistribusikan notifikasi laporan baru melalui dua kanal secara paralel yaitu *Push notification* kepada Petugas dan Admin melalui *Firebase Cloud Messaging* menggunakan token dari data store D4 tabel *fcm_tokens*, dan pesan teks ke Grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri melalui *WhatsApp Gateway microservice* berbasis *Baileys* via *HTTP request* ke *endpoint* REST-nya.

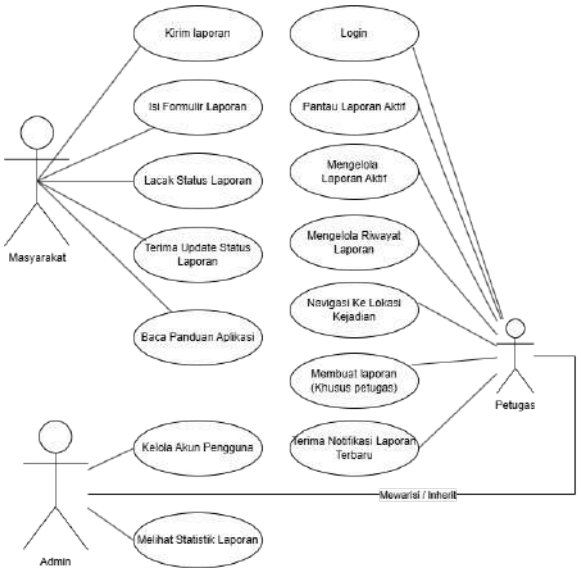
Proses 5.0 adalah Manajemen Status Laporan, yang menerima data pembaruan status dari Petugas, mencatat perubahan ke data store D3 (tabel incident_logs) sebagai jejak audit, memperbarui status di data store D2 (tabel reports), dan meneruskan data perubahan ke Proses 4.0 untuk notifikasi *real-time*.

Proses 6.0 adalah Administrasi Sistem, yang diakses eksklusif oleh Admin dan mencakup pengelolaan data pengguna di data store D1 (tabel users), pembacaan seluruh data laporan dari data store D2 untuk keperluan monitoring dan ekspor, serta pembacaan log aktivitas dari data store D3 untuk keperluan audit.

Keempat *data store* yang diakses oleh proses-proses di atas adalah D1 merepresentasikan dua tabel yang saling berelasi yaitu tabel *users* dan tabel *roles*, yang dikelompokkan dalam satu data store karena keduanya berfungsi secara satu kesatuan dalam manajemen identitas dan otorisasi pengguna, D2 yang merepresentasikan tabel reports, D3 yang merepresentasikan tabel incident_logs, dan data store D4 merepresentasikan tabel fcm_tokens yang menyimpan token FCM per perangkat pengguna sebagai identitas pengiriman *Push notification*.

41 *b. Use Case Diagram*

*Use Case Diagram* mendeskripsikan interaksi antara aktor dengan
15 fungsi-fungsi sistem dari sudut pandang pengguna, tanpa memperlihatkan detail teknis implementasi di dalamnya. Diagram ini memberikan gambaran menyeluruh mengenai apa saja yang dapat dilakukan oleh setiap aktor, sehingga menjadi acuan bersama yang efektif antara pengembang dan pemangku kepentingan untuk memvalidasi kelengkapan fitur sistem. Dalam *Emergency System* terdapat tiga aktor utama yang masing-masing memiliki hak




Gambar 3. 6 *Use Case Diagram*

akses dan cakupan interaksi yang berbeda yaitu Masyarakat Pelapor sebagai aktor dengan hak akses terendah, Petugas Damkar sebagai aktor operasional, dan Admin sebagai aktor dengan hak akses tertinggi. Hubungan antar aktor bersifat hierarkis, Admin mewarisi atau *inherit* seluruh kemampuan Petugas, dan Petugas memiliki akses ke fitur-fitur yang tidak tersedia bagi Masyarakat umum.

Masyarakat sebagai pelapor dapat berinteraksi dengan sistem melalui lima *use case*:

***1)*** Mengirim laporan darurat via *panic button* tanpa perlu login
2) Mengisi formulir dengan data nama, nomor hp, foto kejadian, dan koordinat lokasi yang otomatis diambil dengan sistem geolokasi
3) Melacak status laporan menggunakan kode unik RPT-YYYY-XXXX
4) Menerima pembaruan status laporan secara otomatis, dan
5) Membaca panduan penggunaan aplikasi.

Petugas Damkar dapat berinteraksi dengan sistem melalui tujuh *use case*:

1) Masuk ke sistem / Login
*2)* Memantau seluruh laporan aktif pada *dashboard* secara *real-time*
3) Mengelola laporan aktif, hal ini meliputi ubah status laporan dll.
4) Mengelola riwayat insiden, hal ini juga meliputi peng-export an laporan ke bentuk PDF dan EXCEL

*5)* Membuka navigasi rute menuju lokasi kejadian via *google maps*

6) Membuat laporan insiden yang ditemukan langsung di lapangan

7) Menerima *Push notification* laporan baru secara otomatis via *Firebase Cloud Messaging* (FCM).

Admin memiliki seluruh *use case* Petugas ditambah 2 *use case* eksklusif kecuali buat laporan kejadian darurat:

1) Mengelola akun pengguna dan petugas (tambah, ubah, nonaktifkan, hapus),

2) Melihat statistik laporan darurat dari waktu ke waktu.

3. Desain Basis Data

Pendekatan yang digunakan dalam merancang basis data *Emergency System* berbasis web adalah pemodelan relasional (*relational modeling*) dengan menggunakan *Entity Relationship Diagram* (ERD) sebagai alat bantu visualisasi basis data.


Gambar 3. 7 *Entity Relational Diagram*

Entitas *roles* berfungsi sebagai tabel referensi yang mendefinisikan peran pengguna dalam sistem, yaitu masyarakat, petugas, dan admin. Pemisahan peran ke dalam tabel tersendiri memberikan fleksibilitas apabila di masa mendatang diperlukan penambahan peran baru tanpa harus mengubah struktur tabel utama. Tabel ini dilengkapi dengan *timestamp* created_at dan updated_at yang dikelola otomatis oleh sistem untuk mendukung kebutuhan audit administratif.

Entitas *users* menyimpan data identitas seluruh pengguna terdaftar, mencakup masyarakat pelapor, petugas Damkar, maupun admin. Tabel ini berelasi *many-to-one* terhadap tabel *role*s melalui kolom *role*_id. Selain atribut autentikasi standar seperti *password* (tersimpan dalam bentuk *hash*), tabel ini memuat atribut operasional seperti jabatan, unit, dan status aktif akun. Mekanisme *timestamp* pada tabel ini memastikan administrator memiliki rekaman waktu yang akurat atas setiap perubahan konfigurasi akun untuk keperluan audit keamanan dan penelusuran riwayat pengelolaan personel.

Entitas *reports* merupakan entitas inti sistem yang menyimpan seluruh laporan darurat yang masuk. Tabel ini menerapkan mekanisme *dual-write* bersama tabel incident_logs: setiap pembaruan status laporan secara bersamaan memperbarui kolom status pada tabel reports untuk kebutuhan tampilan *real-time*, sekaligus mencatat rekaman historis ke tabel incident_logs untuk keperluan audit jangka panjang. Beberapa kolom pada tabel ini mencerminkan keputusan desain yang signifikan:

a. Kolom kode menyimpan identifikasi unik berformat RPT-YYYY-XXXX yang di-*generate* otomatis, digunakan pada fitur pelacakan laporan tanpa login.
b. Kolom *type* membedakan laporan normal dan laporan *panic*, di mana laporan *panic* menggunakan kolom nama_pelapor dan no_hp_pelapor sebagai identitas alternatif.
c. Kolom kategori menggunakan tipe VARCHAR dengan validasi diterapkan di lapisan *Backend*, bukan di level struktur database, sehingga penambahan kategori di masa mendatang tidak memerlukan perubahan struktur tabel. Kategori yang tersedia saat ini meliputi: Kebakaran, Kecelakaan, Banjir, Pohon Tumbang, dan Lainnya.
d. Kolom status menggunakan tipe ENUM dengan empat tahapan linear yakni menunggu, diproses, menuju_lokasi, dan selesai, yang memastikan setiap transisi terdokumentasi secara kronologis sekaligus memberikan transparansi informasi kepada masyarakat pelapor.
e. Kolom tingkat menggunakan tipe ENUM bertingkat (rendah, sedang, tinggi) sebagai indikator prioritas penanganan, yang terisi otomatis dengan

nilai *tinggi* pada laporan *panic*. Informasi ini membantu petugas mempertimbangkan alokasi sumber daya sebelum diberangkatkan, sehingga *response time* dapat diminimalkan.

f. Kolom *latitude* dan *longitude* menyimpan koordinat GPS lokasi kejadian yang digunakan sebagai tujuan navigasi pada fitur *"Buka Rute di Google Maps"* di *dashboard* petugas.

Entitas *incident_logs* berperan sebagai tabel jejak audit atau biasa disebut *audit trail* yang mencatat seluruh riwayat perubahan status laporan secara kronologis. Tabel ini berelasi *many-to-one* terhadap tabel *reports* dan tabel *users*. Kombinasi kolom aksi dan catatan membentuk rekaman audit yang lengkap seperti tindakan apa yang dilakukan, dari status apa menuju status apa, disertai keterangan naratif dari petugas yang bertanggung jawab.

Entitas *fcm_tokens* menyimpan token *Firebase Cloud Messaging* (FCM) dari setiap perangkat atau *browser* pengguna yang telah memberikan izin notifikasi. Tabel ini berelasi *many-to-one* terhadap tabel users melalui kolom user_id yang bersifat *nullable*, sehingga satu pengguna dapat memiliki lebih dari satu token apabila mengakses sistem dari perangkat berbeda. Kolom is_active menandai validitas token, yang dapat berubah menjadi tidak aktif apabila pengguna mencabut izin notifikasi atau *browser* melakukan rotasi token secara otomatis. Tabel ini menjadi fondasi teknis seluruh mekanisme *Push notification* dalam sistem.

Relasi antar entitas diimplementasikan melalui mekanisme *foreign key* yang menjamin integritas referensial data satu seperti *roles* dapat dimiliki banyak users, satu users dapat memiliki banyak *reports*, satu *reports* dapat memiliki banyak incident_logs, dan satu users dapat memiliki banyak fcm_tokens. Mekanisme ini memastikan tidak ada laporan yang merujuk pada pengguna yang tidak terdaftar, maupun log insiden yang merujuk pada laporan yang tidak ada dalam sistem.

4. Desain Antarmuka Pengguna

Desain antarmuka pengguna merupakan jembatan komunikasi yang menentukan seberapa efektif dan efisien pengguna dapat berinteraksi dengan seluruh fungsi yang tersedia dalam sistem (Samiha Zaman et al., 2026). Dalam

konteks *Emergency System*, kualitas desain antarmuka memiliki dampak yang jauh lebih signifikan dibandingkan sistem informasi pada umumnya, karena pengguna utama sistem ini, yaitu masyarakat, kemungkinan besar berada dalam kondisi panik, tertekan secara emosional, dan dibatasi oleh waktu saat mengakses aplikasi di tengah situasi darurat. Terdapat 12 halaman utama yang dirancang dalam *Emergency System* SIRTADA, dikelompokkan berdasarkan konteks akses pengguna menjadi tiga bagian: halaman publik, halaman admin, dan halaman petugas.

a. Halaman Publik

1) Halaman Beranda *(Home Page)*

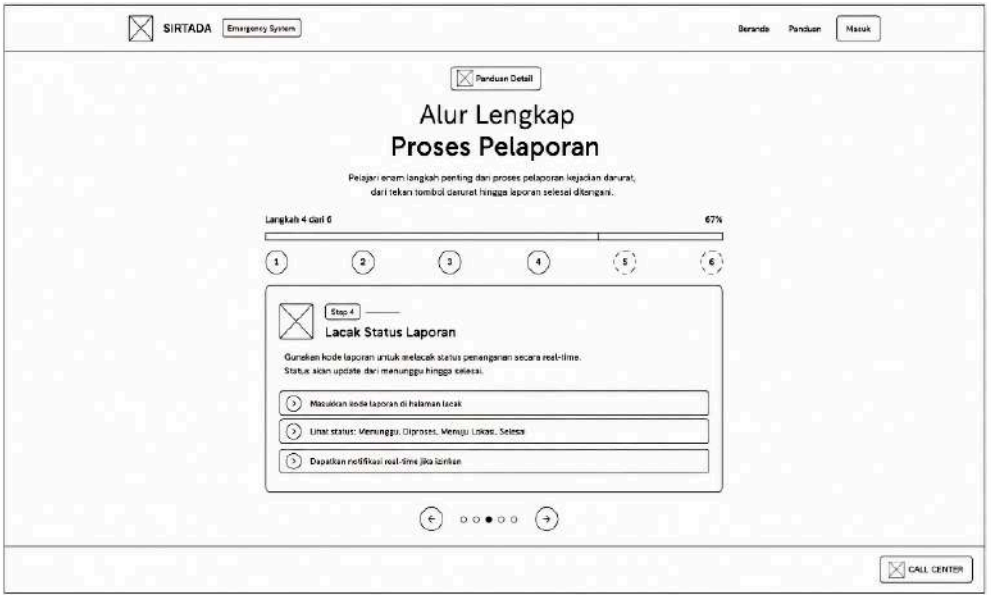
Halaman Beranda dirancang menggunakan pola tata letak *single-page landing* dengan struktur vertikal yang mengalir dari atas ke bawah mengikuti urutan prioritas informasi. Bagian paling atas adalah *persistent navigation bar* yang memuat logo sistem di sisi kiri, tautan menu di tengah, dan tombol Login sebagai *call-to-action* sekunder di sisi kanan. *Navbar* menggunakan *sticky positioning* sehingga tetap terlihat saat pengguna melakukan *scrolling* ke bawah, menjaga aksesibilitas navigasi di setiap titik interaksi.


Gambar 3. 8 *Wireframe Hero Section*

Halaman *Hero Section* berfungsi sebagai titik masuk utama sistem dan menampilkan tombol *panic button* sebagai elemen primer. Fitur utama halaman ini meliputi tombol darurat, informasi status sistem, dan navigasi ke fitur lain. Tombol *panic button* ditempatkan secara mencolok di tengah layar agar dapat diakses dengan cepat dalam kondisi darurat. Penerapan ukuran besar pada

tombol darurat mengikuti prinsip *prominence* dalam hierarki visual elemen dengan prioritas tertinggi mendapatkan bobot visual terbesar.

Di bawah *hero section* terdapat *step carousel* yang memandu pengguna melalui alur pelaporan dalam lima langkah berurutan.




Gambar 3. 9 *Wireframe* Panduan *Section*

Bagian panduan *section* menampilkan *step carousel* yang memandu pengguna melalui lima langkah alur pelaporan secara berurutan. Format *carousel* dipilih agar informasi bertahap dapat disajikan secara kompak tanpa memperpanjang halaman secara vertikal. Setiap langkah dilengkapi ikon, nomor urut, dan keterangan singkat.


Gambar 3. 10 *Wireframe Stats Section*

Di bagian selanjutnya, terdapat *statistics section* yang menampilkan tiga kartu metrik berisi jumlah total laporan masuk, laporan selesai, dan rata-rata respons untuk memberikan transparansi data kepada masyarakat.

2) Halaman Lacak Laporan (*Track Report Page*)

Gambar 3. 11 *Wireframe* Cari Laporan (Lacak laporan)

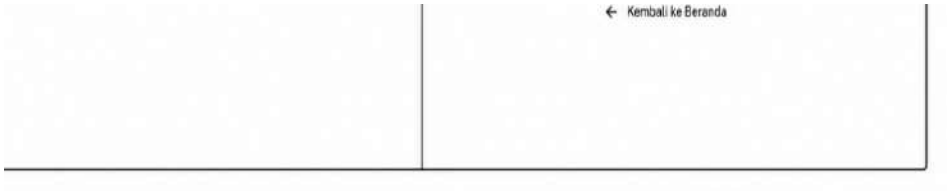
Halaman ini berfungsi untuk memberikan informasi terkini kepada pelapor mengenai status laporan yang telah dikirimkan. Fitur utama terdiri dari *search bar* adalah untuk memasukkan kode laporan.

*Result card* yang menampilkan kode dan status laporan, serta *progress timeline* vertikal yang menampilkan empat tahap penanganan secara kronologis (menunggu, diproses, menuju lokasi, selesai).


Gambar 3. 12 *Wireframe Result Card* (Lacak laporan)

Tahap aktif ditandai dengan warna yang berbeda, sedangkan tahap yang belum tercapai ditampilkan dalam kondisi abu-abu. Pola *timeline* ini mengikuti konvensi sistem pelacakan yang umum ada pada aplikasi *tracking status* sehingga pelapor dapat memahami status atau posisi laporannya dengan mudah tanpa penjelasan yang rumit.

3) Halaman Login (*Login Page*)



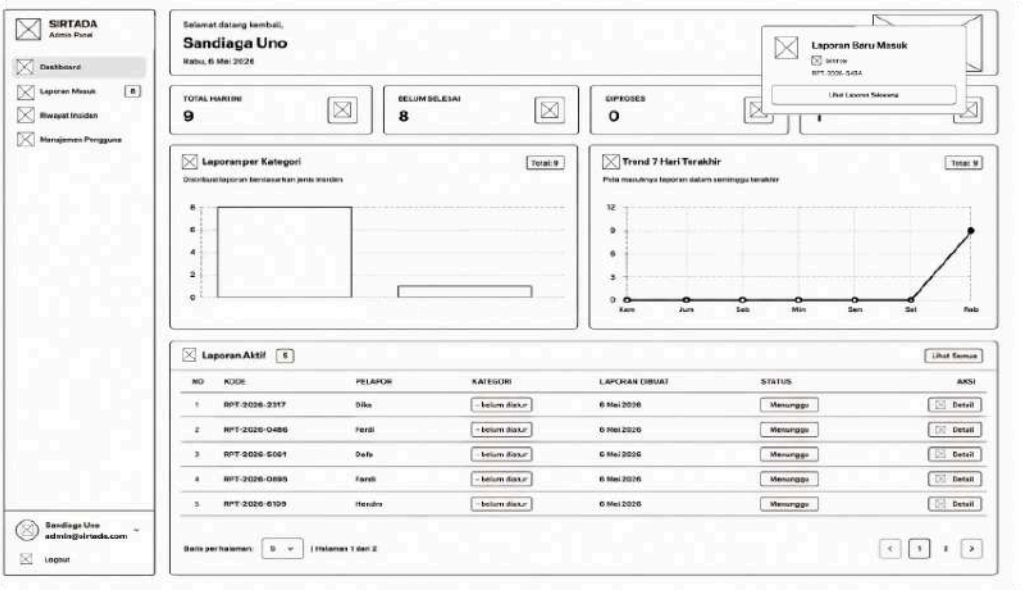

Gambar 3. 13 *Wireframe Login Page*

Halaman Login berfungsi sebagai gerbang autentikasi bagi admin dan petugas untuk mengakses sistem. Formulir ditempatkan di tengah halaman dengan latar polos agar fokus pengguna langsung tertuju pada proses masuk. Formulir terdiri dari dua *field* (email dan kata sandi) yang disusun vertikal dengan label di atas setiap *field*. Tata letak ini mengikuti konvensi formulir login yang sudah familiar sehingga mempercepat proses autentikasi.

b. Halaman Admin

Seluruh halaman admin menggunakan *shell layout* yang konsisten, *sidebar* navigasi tetap (*fixed sidebar*) di sisi kiri dengan lebar tetap memuat seluruh tautan modul, sementara area konten utama menempati sisa lebar halaman di sebelah kanan. Di bagian paling atas area konten terdapat topbar yang menampilkan nama pengguna aktif, avatar, dan ikon lonceng sebagai indikator notifikasi. Pola *sidebar + content area layout* ini mengikuti prinsip antarmuka *back-office* yang sudah sangat familiar di kalangan pengguna profesional, sehingga meminimalkan kurva belajar (*learning curve*) bagi admin yang baru pertama menggunakan sistem.

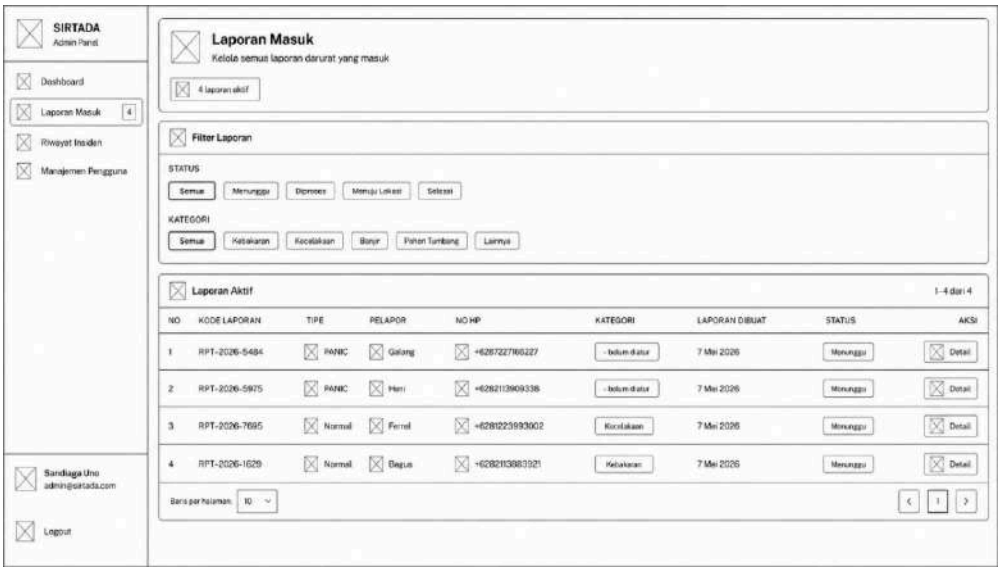
1) *Dashboard* Admin




Gambar 3. 14 *Wireframe Dashboard* Admin

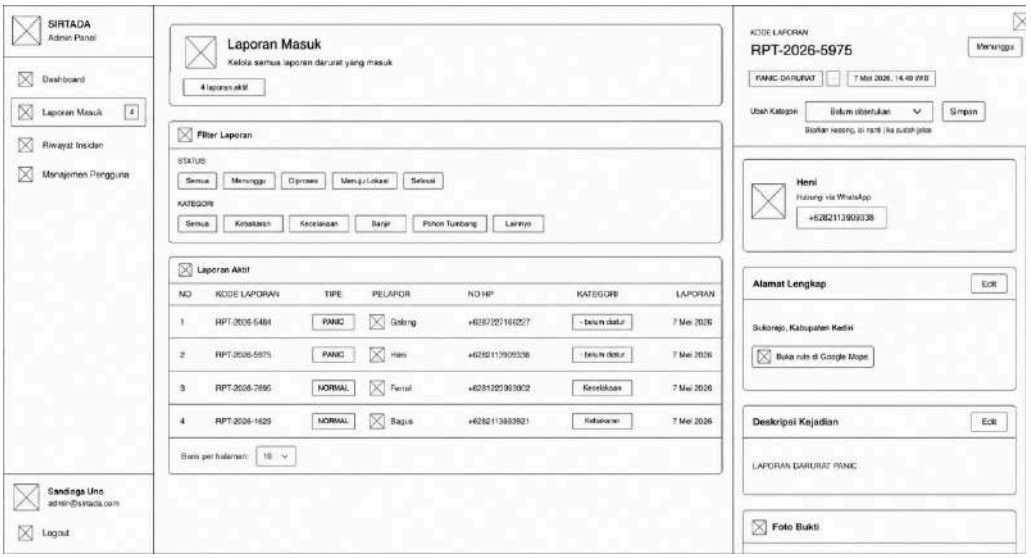
*Dashboard* Admin berfungsi sebagai pusat pemantauan operasional yang menampilkan seluruh informasi kritis sistem dalam satu halaman. Bagian atas menampilkan lima *metric card* berisi angka-angka operasional kunci. Di bawahnya terdapat dua panel grafik berdampingan *line chart* untuk tren laporan tujuh hari terakhir dan *bar/pie chart* untuk distribusi laporan per kategori. Bagian bawah memuat tabel laporan terbaru yang diperbarui secara *real-time* dengan notifikasi saat laporan baru masuk.

2) Halaman Laporan Masuk



Gambar 3. 15 *Wireframe* Halaman Laporan Masuk

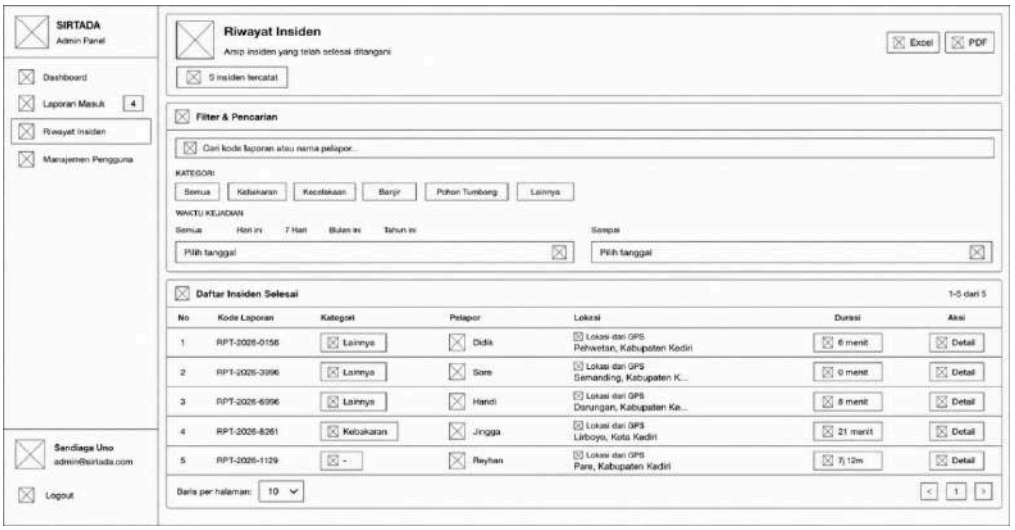
Halaman Laporan Masuk mengadopsi pola *master-detail layout* yang mengintegrasikan daftar laporan dengan side drawer atau modal untuk menampilkan detail informasi dan kontrol aksi secara efisien. Di bagian atas, terdapat *toolbar horizontal* dengan filter status serta kategori yang menggunakan mekanisme instant filter, sehingga data diperbarui secara otomatis tanpa memerlukan tombol konfirmasi tambahan. Selain itu, setiap baris pada tabel dirancang interaktif agar pengguna dapat langsung membuka panel detail dengan satu klik, menciptakan alur kerja yang lebih responsif dan praktis.




Gambar 3. 16 *Wireframe* Detail Laporan Masuk

Panel detail menggunakan *tabbed interface* atau *accordion* yang terbagi menjadi tiga bagian: informasi laporan beserta kontrol status dengan catatan wajib, image uploader untuk manajemen foto, dan *activity log timeline* yang mencatat riwayat perubahan secara kronologis beserta nama pelaksana dan *timestamp*. Pembaruan status menerapkan pola *constrained interaction* pada *dropdown*, yang hanya menampilkan opsi tahapan logis berikutnya untuk mencegah kesalahan input pengguna.

3) Halaman Riwayat Laporan



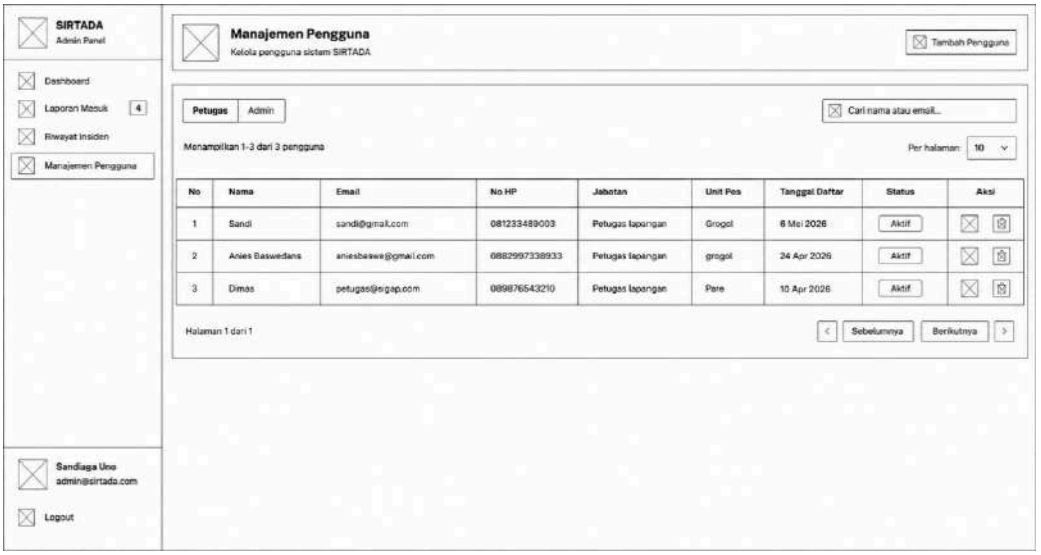

Gambar 3. 17 *Wireframe* Riwayat Insiden

Halaman Riwayat Laporan berfungsi sebagai arsip terpusat untuk laporan berstatus "selesai" dengan fokus pada kemudahan penelusuran data historis. Menggunakan archive layout, halaman ini dilengkapi empat filter simultan (rentang tanggal, kategori, status akhir, dan *search bar*) yang bekerja secara instan, serta fitur ekspor ke format PDF dan Excel untuk pelaporan administratif. Tabel datanya memuat informasi kunci termasuk durasi penanganan otomatis untuk keperluan evaluasi, di mana setiap baris dapat diklik untuk mengakses detail riwayat lengkap.


Gambar 3. 18 *Wireframe* Detail Riwayat Insiden

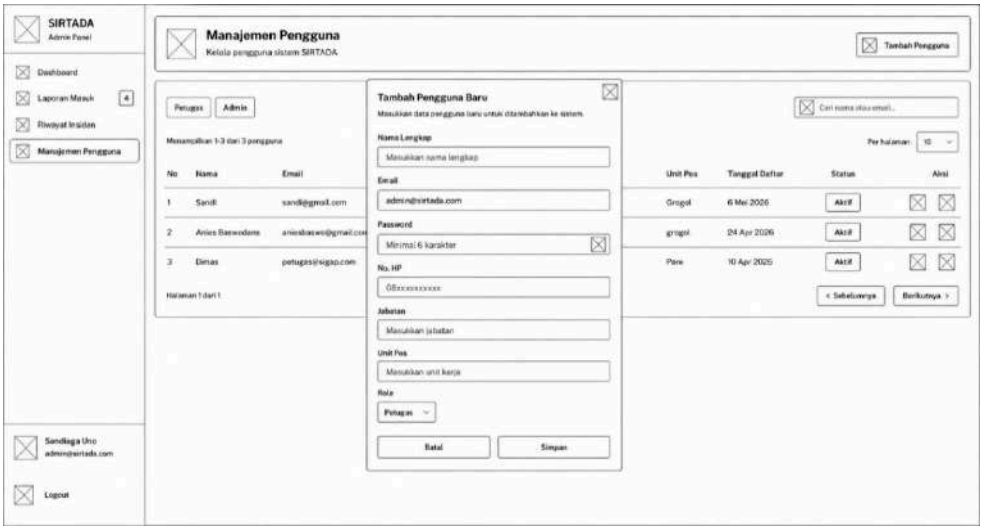
Panel detail riwayat ditampilkan dalam modal dialog *read-only* yang memuat ringkasan identitas (kode, kategori, urgensi, dan badge status). Informasi ini dilengkapi data pelapor, deskripsi, lokasi melalui peta mini Leaflet.js, serta galeri foto dengan fitur *lightbox*. Sebagai bentuk *immutable audit trail*, bagian bawah modal menyajikan *activity log timeline* vertikal yang mencatat seluruh riwayat perubahan, petugas, catatan lapangan, dan *timestamp* secara kronologis untuk kebutuhan evaluasi serta akuntabilitas.

4) Halaman Manajemen Pengguna




Gambar 3. 19 *Wireframe* Halaman Manajemen Pengguna

Halaman Manajemen Pengguna merupakan halaman eksklusif admin yang berfungsi sebagai pusat kendali seluruh akun yang terdaftar dalam sistem SIRTADA. Halaman ini mendukung operasi CRUD (tambah, lihat, ubah, hapus) akun petugas tanpa perlu berpindah halaman. Bagian atas memuat *toolbar* yang terdiri dari *search bar* pencarian berdasarkan nama atau email, filter peran pengguna, dan tombol "Tambah Pengguna" di sisi kanan. Di bawahnya, seluruh data akun ditampilkan dalam tabel dengan kolom foto profil, nama lengkap, email, nomor telepon, peran, jabatan, unit, serta tombol "Edit" dan "Hapus" pada setiap baris agar admin dapat langsung mengakses kontrol pengelolaan tanpa langkah tambahan.




Gambar 3. 20 *Wireframe* Modals Tambah Data Pengguna

Penambahan akun dilakukan melalui *modal dialog* yang muncul di atas halaman tanpa mengalihkan admin ke halaman terpisah. Formulir memuat delapan *field* yang disusun vertikal: nama lengkap, email, kata sandi, konfirmasi kata sandi, nomor telepon, peran (*dropdown* "admin"/"petugas"), jabatan, dan unit penugasan. Setiap *field* dilengkapi validasi *client-side* yang menampilkan pesan kesalahan secara *inline* tanpa menunggu formulir dikirim, serta tombol *toggle visibility* pada *field* kata sandi. Bagian bawah modal memuat tombol "Batal" dan "Simpan" sebagai dua pilihan aksi penyelesaian.


Gambar 3. 21 *Wireframe Modals Edit* Data Pengguna

Modal edit pengguna memiliki struktur yang identik dengan modal tambah pengguna, dengan dua perbedaan utama. Pertama, seluruh *field* terisi

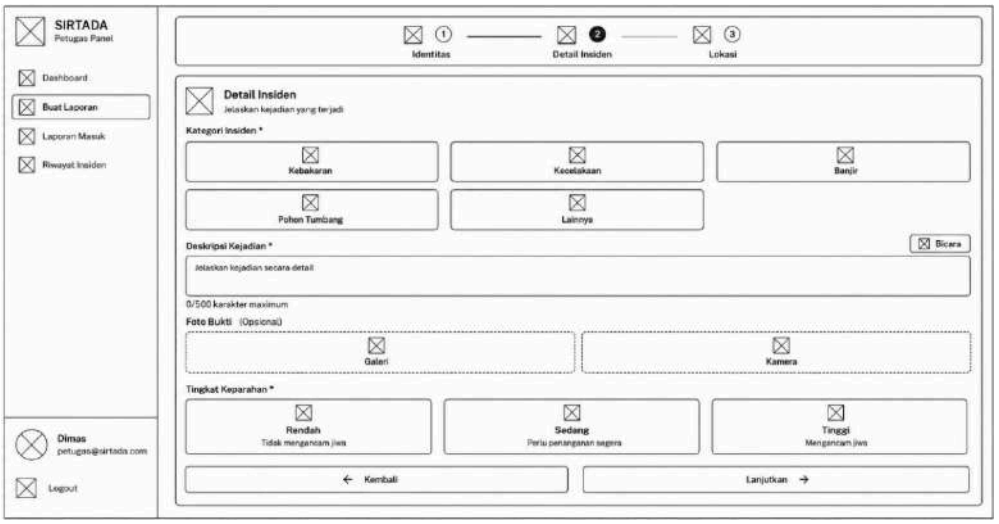
otomatis (*pre-populated*) dengan data akun yang sedang diedit sehingga admin hanya perlu mengubah *field* yang relevan. Kedua, *field* kata sandi bersifat opsional jika dikosongkan, sistem mempertahankan kata sandi lama tanpa perubahan.

c. Halaman Petugas

Seluruh halaman petugas menggunakan *shell layout* yang konsisten dengan halaman admin, namun navigasi *sidebar* hanya menampilkan modul yang relevan dengan tugas operasional petugas: *dashboard*, laporan aktif, riwayat, dan buat laporan. Penyederhanaan ini membatasi akses petugas hanya pada fungsi yang sesuai dengan kewenangannya.


Gambar 3. 22 *Wireframe* Halaman *Dashboard* Utama Petugas

*Dashboard* petugas berfokus pada informasi yang langsung relevan dengan tugas penanganan insiden di lapangan, berbeda dari *dashboard* admin yang menekankan data agregat dan statistik operasional. Bagian atas menampilkan tiga *metric card* berisi jumlah laporan aktif, menunggu respons, dan selesai hari ini. Di bawahnya, tabel laporan aktif menampilkan seluruh laporan berstatus "menunggu" dan "diproses" secara *real-time*, dengan aksen warna merah pada laporan berkegawatan tinggi agar petugas dapat mengidentifikasi prioritas secara langsung. Laporan baru yang masuk muncul di posisi teratas tabel disertai *toast notification* agar tidak terlewat.




Gambar 3. 23 *Wireframe* Halaman Buat Laporan Petugas

Halaman Buat Laporan pada antarmuka petugas menggunakan komponen formulir yang sama (*shared component*) dengan formulir pelaporan publik, sehingga konsistensi antarmuka terjaga dan duplikasi kode diminimalkan. Perbedaan utamanya adalah data pelapor tersinkronisasi otomatis dengan akun petugas yang sedang *login*. Formulir memuat *field* nama pelapor, nomor telepon, kategori insiden, tingkat kegawatan, deskripsi kejadian dengan dukungan *speech-to-text*, deteksi lokasi otomatis via *Geolocation API*, dan *file uploader* untuk foto dokumentasi. Fitur ini memungkinkan petugas mencatat insiden yang ditemukan langsung di lapangan tanpa bergantung pada laporan dari masyarakat.

E. Implementasi Sistem (*Implementation*)

Fase implementasi sistem merupakan tahap di mana seluruh rancangan sistem yang telah didefinisikan pada fase desain ditransformasikan secara konkret ke dalam kode program yang dapat dieksekusi (Gorrod, 2004). Dalam pengembangan *Emergency System* berbasis web, proses implementasi dilakukan dengan memisahkan pengerjaan menjadi dua domain utama yang berjalan secara paralel namun tetap terhubung melalui kontrak antarmuka API yang disepakati sejak awal, yaitu domain *Backend* dan domain *Frontend*.

1. Bahasa dan *Tools* Pemrograman

Pemilihan bahasa pemrograman dan *tools* dalam pengembangan *Emergency System* berbasis web didasarkan pada pertimbangan kesesuaian teknologi dengan kebutuhan sistem. Pada Tabel 3.6 akan disajikan data terkait

penggunaan bahasa pemrograman *Backend*, teknologi atau *Library*, versi dan fungsi yang digunakan dalam sistem.

Tabel 3. 6 Bahasa, *Library* dan *Framework* Pemrograman (*Backend*)

| No | Komponen | Teknologi / *Library* | Versi | Fungsi dalam sistem |
|---|---|---|---|---|
| 1 | Bahasa Pemrograman | PHP | 8.2 + | Bahasa utama *Backend* yang berfungsi sebagai dukungan Fibers untuk operasi konkuren |
| 2 | *Framework Backend* | *CodeIgniter* | 4.7 | Penyedia *RESTful API* berbasis pola MVC (*Routing, Middleware,* dan *Query Builder*) |
| 3 | Autentikasi | *Firebase*/php-jwt | 7.0 | Implementasi JSON Web Token (JWT) *stateless* untuk proteksi *endpoint API* |
| 4 | *Firebase Integration* | kreait/*Firebase*-php | 7.0 | Pengiriman *Push notification* via *Firebase Cloud Messaging* (FCM) menggunakan *Firebase* Admin SDK |
| 5 | *HTTP Client* | Guzzle | 7.0 | Komunikasi ke layanan eksternal termasuk OpenStreetMap Nominatim API untuk *reverse geocoding* |
| 6 | *Logging* | Monolog | 3.0 | Pencatatan log aktivitas dan *Error* sistem di sisi *Backend* |
| 7 | *Package Manager* | Composer | 2.x | Manajemen seluruh dependensi ekosistem PHP |
| 8 | *WhatsApp Gateway Client* | cURL (PHP *native*) | Built-in PHP | Pengiriman *HTTP POST* ke *endpoint REST WhatsApp Gateway microservice* dari dalam *WhatsApp*Service.php |

Pada Tabel 3.7 akan disajikan data terkait penggunaan teknologi *Frontend* yang digunakan untuk membangun antarmuka pengguna pada sistem.

Tabel 3. 7 Bahasa, *Library* dan *Framework* Pemrograman (*Frontend*)

| No | Komponen | Teknologi / *Library* | Versi | Fungsi dalam sistem |
|---|---|---|---|---|
| 1 | Bahasa Pemrograman | *TypeScript* | 5.7 | Bahasa utama *Frontend* dan sistem tipe statis mencegah kesalahan tipe data koordinat |
| 2 | *Framework* UI | *React.js* | 19 | Membangun antarmuka berbasis komponen (SPA) dengan *Virtual DOM* yang efisien |
| 3 | *Firebase* SDK + Service Worker | *Firebase* | 11.0.0 | Registrasi token FCM dan penerimaan *Push notification* di *browser* |
| 4 | *React* Router DOM | *React.js* | 7.14.0 | *Client-side* routing antar halaman (SPA navigation) |
| 5 | Format Waktu | *date-fns* | 4.1.0 | Manipulasi dan format tampilan tanggal/waktu pada laporan |
| 6 | Export Pdf | *jsPDF* + *jspdf-autotable* | 4.2.1 / 5.0.7 | Generate file PDF untuk fitur ekspor data laporan |
| 7 | *Speech to Text* | *Web Speech API* | - | *Hook* untuk *Speech-to-Text,* mengubah input suara menjadi text |

| | | | | |
|---|---|---|---|---|
| 8 | Export Excel | *xlsx* | 0.18.5 | Generate file Excel untuk fitur ekspor data laporan |
| 9 | Build Tool | *Vite* | 8.0.14 | Kompilasi & bundling *Frontend,* HMR (*Hot Module Replacement*) berkecepatan tinggi |
| 10 | CSS *Framework* | *Tailwind CSS* | 4.2 | *Utility-first styling* yang memastikan antarmuka responsif dan *mobile-friendly* |
| 11 | Komponen UI | *shadcn/ui (Radix UI)* | 4.7.0 | Komponen antarmuka yang aksesibel dan konsisten lintas halaman |
| 12 | Validasi Formulir | *React Hook Form + Zod* | 7.54.1 / 3.24.1 | Validasi input terstruktur di sisi client , definisi schema terpusat |
| 13 | Peta Interaktif | *Leaflet.js + React-Leaflet* | 1.9.4 / 5.0.0 | Visualisasi peta interaktif berbasis OpenStreetMap tanpa biaya API key |
| 14 | Grafik & Statistik | *Recharts* | 2.15.0 | Visualisasi data statistik laporan pada *dashboard* petugas dan admin |
| 15 | Animasi UI | *Framer Motion* | 12.38.0 | Animasi antarmuka untuk meningkatkan pengalaman pengguna |
| 16 | HTTP Client | *Axios* | 1.14.0 | Komunikasi terpusat dengan seluruh *endpoint* REST API *Backend* |
| 17 | Package Manager | *npm* | 11.14.0 | Manajemen seluruh dependensi ekosistem JavaScript/TypeScript |

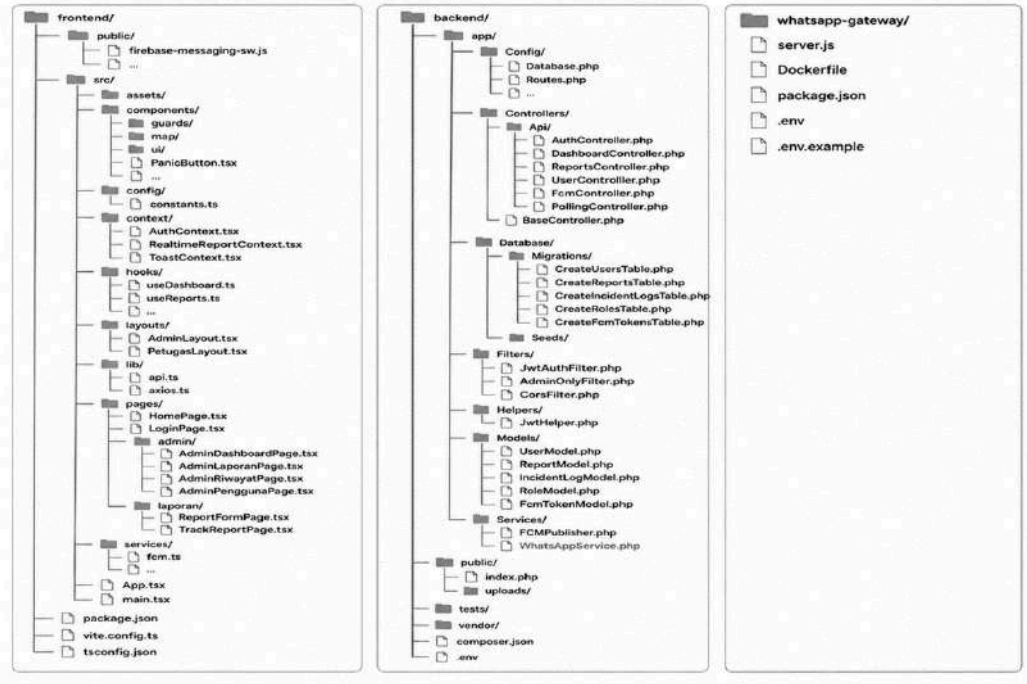
Pada Tabel 3.8 akan disajikan data terkait penggunaan teknologi pengujian pada sistem.

Tabel 3. 8 *Tools* pengujian & pengembangan sistem

| No | Tools | Teknologi / *Library* | Fungsi dalam sistem |
|---|---|---|---|
| 1 | Google Chrome + DevTools | Lingkungan pengujian antarmuka utama | Inspeksi elemen, *network*, dan konsol *JavaScript* |
| 2 | Git | Sistem kontrol versi (version control) | Manajemen riwayat perubahan kode program |
| 3 | Visual Studio Code | Code editor utama | Pengembangan *Backend* dan *Frontend* secara terpadu |
| 4 | Postman + Newman CLI | Alat pengujian Integrasi | Eksekusi dan otomasi pengujian integrasi seluruh *endpoint REST API* secara berurutan |
| 5 | PHPUnit 10.5 dan Vitest 4.1 | PHPUnit 10.5 / Vitest 4.1 | Unit *testing Backend* (PHP) dan *Frontend* (*React* TypeScript) dengan isolasi mock penuh |
| 6 | *Grafana k6* | Alat pengujian performa & beban | Simulasi pengguna serentak dan pengukuran *response time API* di bawah kondisi beban nyata |

2. Struktur Folder dan Modul Sistem

Struktur direktori proyek *Emergency System* dibagi menjadi tiga direktori utama di tingkat teratas: *Backend/* yang berisi seluruh kode *CodeIgniter* 4, *Frontend/* yang berisi seluruh kode *React* dan *WhatsApp-Gateway*/ yang berisi kode *microservice* Node.js berbasis *Baileys* untuk pengiriman notifikasi laporan darurat ke grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri.




Gambar 3. 24 Struktur Folder *Project*

Pada Tabel 3.9 akan disajikan data modul *Backend* yang digunakan pada aplikasi *Emergency System* berbasis web.

Tabel 3. 9 Struktur folder & Modul sistem (*Backend*)

| No | Path / File | Jenis | Fungsi |
|---|---|---|---|
| 1 | Controllers/Api/AuthController.php | Controller | Menangani autentikasi pengguna: login, validasi JWT, dan logout |
| 2 | Controllers/Api/ReportsController.php | Controller | Manajemen laporan darurat: buat, baca, perbarui status, validasi kategori |
| 3 | Controllers/Api/UsersController.php | Controller | Pengelolaan akun pengguna oleh admin (CRUD akun petugas & masyarakat) |
| 4 | Controllers/Api/*Dashboard*Controller.php | | Penyediaan data statistik operasional |

| | | | |
|---|---|---|---|
| | | Controller | dan data marker peta untuk *dashboard* |
| 5 | Controllers/Api/FcmController.php | Controller | Manajemen token FCM: registrasi token baru, penghapusan token tidak aktif |
| 6 | Controllers/Api/PollingController.php | Controller | *Endpoint* polling HTTP sebagai fallback mekanisme update data *real-time* |
| 7 | Models/UserModel.php | Model | Representasi dan query tabel users di basis data |
| 8 | Models/ReportModel.php | Model | Representasi dan query tabel reports di basis data |
| 9 | Models/IncidentLogModel.php | Model | Representasi dan query tabel incident_logs (audit trail) |
| 10 | Models/*Role*Model.php | Model | Representasi dan query tabel *role*s di basis data |
| 11 | Models/FcmTokenModel.php | Model | Representasi dan query tabel fcm_tokens di basis data |
| 12 | Filters/ | Middleware | Implementasi JwtAuthFilter (autentikasi) dan AdminFilter (otorisasi RBAC) |
| 13 | Database/Migrations/ | Migrasi | 5 berkas migrasi yang mendefinisikan skema basis data secara bertahap dan terkontrol |
| 14 | Services/FCMPublisher.php | Service | Pengiriman *Push notification* ke perangkat terdaftar melalui *Firebase* Admin SDK |
| 15 | Services/*WhatsApp*Service.php | Service | Pengiriman notifikasi laporan darurat ke grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri melalui HTTP POST ke *WhatsApp Gateway* microservice berbasis *Baileys* |
| 16 | Helpers/JwtHelper.php | Konfigurasi | Menyediakan fungsi pembuatan dan validasi JSON Web Token (JWT), digunakan oleh AuthController dan JwtAuthFilter. |
| 17 | Config/Routes.php | Konfigurasi | Mendefinisikan seluruh *routing* REST API sistem, termasuk penerapan *filter* autentikasi JWT dan filter otorisasi RBAC pada *endpoint* yang memerlukan perlindungan. Seluruh *controller* terdaftar dan dipetakan ke *endpoint* |

| | | | |
|---|---|---|---|
| | | | HTTP melalui berkas ini. |

Pada Tabel 3.10 akan disajikan data modul *Frontend* yang digunakan pada aplikasi *Emergency System* berbasis web.

Tabel 3. 10 Struktur folder & modul sistem (*Frontend*)

| No | Path / File | Jenis | Fungsi | Akses |
|---|---|---|---|---|
| 1 | pages/admin/Admin*Dashboard* Page.tsx | *Page* | Halaman utama statistik operasional untuk admin | Admin |
| 2 | pages/admin/AdminLaporanPage.tsx | *Page* | Daftar dan manajemen seluruh laporan insiden | Admin |
| 3 | pages/admin/AdminRiwayatPage.tsx | *Page* | Arsip riwayat insiden dengan fitur filter & ekspor PDF/Excel | Admin |
| 4 | pages/admin/AdminPenggunaPage.tsx | *Page* | Manajemen akun pengguna dan petugas | Admin |
| 5 | pages/laporan/ReportFormPage.tsx | *Page* | Formulir pelaporan lengkap dengan geolokasi dan unggah foto | Publik |
| 6 | pages/laporan/TrackReportPage.tsx | *Page* | Pelacakan status laporan via kode unik RPT-YYYY-XXXX | Publik |
| 7 | components/PanicButton.tsx | *Component* | Tombol darurat instan, mengambil geolokasi dan menampilkan form input singkat untuk laporan | Publik |
| 8 | components/NotificationPrompt.tsx | *Component* | Permintaan izin *Push notification* ke *browser* pengguna | Semua |
| 9 | components/map/ | *Component* | Komponen peta Leaflet: marker berwarna, popup info, rute navigasi | Admin / Petugas |
| 10 | components/guards/ | *Component* | Route guard berbasis peran: memblokir akses halaman yang tidak sesuai *role* | Semua |
| 11 | context/AuthContext.tsx | *Context* | Manajemen state autentikasi global (token JWT, data user aktif) | Global |
| 12 | context/ToastContext.tsx | *Context* | Manajemen notifikasi toast UI di seluruh aplikasi | Global |
| 13 | context/RealtimeReportsContext.tsx | *Context* | Manajemen state laporan *real-time* yang dibagikan ke seluruh komponen *dashboard* | Admin/Petugas |
| 14 | services/index.ts | *Service* | Pembungkus (wrapper) terpusat untuk seluruh pemanggilan *endpoint* API | Global |
| 15 | lib/api.ts | *Library* | Konfigurasi instance Axios: base URL, interceptor token, dan *Error* handling | Global |
| 16 | lib/utils.ts | *Library* | Fungsi utilitas umum yang digunakan lintas komponen | Global |
| 17 | hooks/use*Dashboard*.ts | *Hook* | *Custom hook* untuk mengambil dan menyimpan data statistik *dashboard* | Admin/Petugas |

| | | | | |
|---|---|---|---|---|
| 18 | hooks/useReports.ts | *Hook* | *Custom hook* untuk manajemen state daftar laporan | Admin/Petugas |

Pada Tabel 3.11 akan disajikan data modul *WhatsApp Gateway* yang digunakan pada aplikasi *Emergency System* berbasis web.

Tabel 3. 11 Modul *WhatsApp Gateway*

| No | Path / File | Jenis | Fungsi |
|---|---|---|---|
| 1 | server.js | *Entry point* | Inisialisasi server Express.js dan koneksi *Baileys* WebSocket ke server *WhatsApp* yang berfungsi menangani QR scan, *auto-reconnect*, dan *health check* periodik |
| 2 | Dockerfile | *Container* | Image ringan node:22-slim (~100MB) tanpa Chromium , dikonfigurasi untuk deployment di server vps |
| 3 | package.json | Konfigurasi | Dependensi: @whiskeysockets/*Baileys*, express, qrcode |

3. Cuplikan Kode Program Utama

```
export function AuthProvider({ children }: { children: React.ReactNode }) {
  const [user, setUser]   = useState<User | null>(null);
  const [token, setToken] = useState<string | null>(null);
  const navigate = useNavigate();

  // Pulihkan sesi dari httpOnly cookie saat aplikasi dimuat
  useEffect(() => {
    const restoreSession = async () => {
      try {
        const res = await api.get('/auth/me');
        setUser(res.data.user);
        setToken('cookie-auth');
      } catch {
        setUser(null); setToken(null);
      }
    };
    restoreSession();
  }, []);

  const login = async (email: string, password: string) => {
    const res  = await authService.login(email, password);
    const user = res.user; // token disimpan di httpOnly cookie oleh backend
    if (user?.role === 'masyarakat')
      throw new Error('Akses hanya untuk admin dan petugas.');
    setUser(user);
    setToken('cookie-auth');
    return user;
  };

  const logout = useCallback(async () => {
    await api.post('/auth/logout'); // backend hapus cookie
    setUser(null); setToken(null);
    navigate('/login', { replace: true });
  }, [navigate]);

  return (
    <AuthCtx.Provider value={{ user, token, login, logout,
      isAdmin: user?.role === 'admin',
      isPetugas: user?.role === 'petugas' || user?.role === 'admin',
      isLoggedIn: !!token }}>
      {children}
    </AuthCtx.Provider>
  );
}
```

Gambar 3. 25 *Frontend*, Manajemen Sesi Autentikasi

AuthContext.tsx adalah komponen *context React* yang bertanggung jawab mengelola seluruh status autentikasi pengguna secara global di seluruh aplikasi. Komponen ini mendefinisikan tiga fungsi utama. Pertama, fungsi

*restoreSession* yang dijalankan otomatis saat aplikasi pertama kali dimuat, fungsi ini memanggil *endpoint* /auth/me untuk memverifikasi apakah *httpOnly cookie* yang tersimpan di *browser* masih valid, sehingga pengguna tidak perlu login ulang setiap kali membuka halaman. Blok *finally* memastikan status *loading* selalu diperbarui ke nilai *false* meskipun permintaan ke server gagal, mencegah kondisi *infinite loading*. Kedua, fungsi login yang memproses autentikasi pengguna dan memblokir akses bagi pengguna berperan *masyarakat*. Ketiga, fungsi *logout* yang memanggil *endpoint* /auth/logout di *Backend* untuk menghapus *cookie* sekaligus mereset seluruh status autentikasi di sisi *Frontend*. Seluruh status ini dibagikan ke komponen lain melalui mekanisme AuthCtx.Provider.

```
const api = axios.create({
  baseURL: API_URL,
  timeout: 15000,
  withCredentials: true, // wajib agar httpOnly cookie dikirim otomatis
});

// Request: atur Content-Type; biarkan browser atur boundary FormData
api.interceptors.request.use((config) => {
  if (!(config.data instanceof FormData))
    config.headers['Content-Type'] = 'application/json';
  else
    delete config.headers['Content-Type'];
  return config;
});

// Response: logout otomatis jika cookie kedaluwarsa (401)
api.interceptors.response.use(
  (res) => res,
  (err) => {
    if (err.response?.status === 401 && (window as any).__sigapToken)
      (window as any).__sigapLogout?.();
    err.userMessage = err.response?.data?.message || 'Terjadi kesalahan.';
    return Promise.reject(err);
  }
);
```

Gambar 3. 26 *Frontend*, Konfigurasi Axios

axios.ts adalah modul konfigurasi terpusat untuk seluruh komunikasi HTTP antara *Frontend* dan *Backend*. Terdapat dua konfigurasi kritis di dalamnya. Pertama, opsi withCredentials, *true* yang memastikan *browser* menyertakan *httpOnly cookie* secara otomatis pada setiap permintaan API tanpa perlu pengelolaan token secara manual di sisi JavaScript, ini adalah mekanisme utama perlindungan terhadap serangan *Cross-Site Scripting* (XSS). Kedua, dua buah *interceptor*: *request interceptor* yang mengatur *header Content-Type* secara otomatis (JSON untuk data biasa, *multipart* untuk unggahan berkas) dan

*response interceptor* yang menangkap seluruh respons *Error* secara terpusat apabila server mengembalikan status HTTP 401 (token kedaluwarsa atau tidak valid), *interceptor* secara otomatis memanggil fungsi logout untuk mengakhiri sesi pengguna tanpa perlu penanganan di setiap komponen secara terpisah.

```php
// app/Controllers/Api/AuthController.php
public function login(): ResponseInterface
{
    $json     = $this->request->getJSON(true);
    $user     = $this->userModel->findByEmailWithRole($json['email'] ?? '');

    if (!$user || !password_verify($json["password"] ?? '', $user['password']))
        return $this->response->setStatusCode(401)
                    ->setJSON(['success' => false, 'message' => 'Email atau password salah.']);

    if ($user['role'] === 'masyarakat')
        return $this->response->setStatusCode(403)
                    ->setJSON(['success' => false, 'message' => 'Akses ditolak.']);

    $token = JwtHelper::generateToken([
        'id' => $user['id'], 'role' => $user['role'], 'nama' => $user['nama'],
    ]);
    unset($user['password']);

    // Simpan JWT di httpOnly cookie (tidak dapat diakses JavaScript)
    $this->response->setCookie('sigap_token', $token, 86400, '', '/', '', false, true, 'Strict');

    return $this->response->setJSON(["success" => true, 'data' => ['user' => $user]]);
}

public function logout(): ResponseInterface
{
    $this->response->deleteCookie('sigap_token');
    return $this->response->setJSON(['success' => true, 'message' => 'Logout berhasil.']);
}
```

Gambar 3. 27 *Backend*, Autentikasi Pengguna

AuthController.php adalah *controller* yang menangani tiga *endpoint* autentikasi utama sistem. Fungsi login menerima kredensial email dan *password*, memverifikasinya terhadap data di tabel users menggunakan password_verify(), lalu memblokir akses bagi pengguna berperan *masyarakat* dengan mengembalikan respons HTTP 403. Apabila autentikasi berhasil, server menerbitkan token JWT melalui JwtHelper::generateToken() dan menyimpannya sebagai *httpOnly cookie* dengan konfigurasi SameSite=Strict dan masa aktif 24 jam, token ini tidak dapat dibaca oleh JavaScript di sisi klien, sehingga terlindungi dari serangan XSS. Fungsi logout menghapus *cookie* tersebut di sisi server, sementara fungsi me membaca dan memvalidasi *cookie* yang ada untuk keperluan pemulihan sesi.

```php
// app/Filters/JwtAuthFilter.php
public function before(RequestInterface $request, $arguments = null): ?ResponseInterface
{
    // 1. Baca token dari httpOnly cookie (prioritas utama)
    $token = $request->getCookie('sigap_token');

    // 2. Fallback: Authorization header
    if (empty($token)) {
        $header = $request->getHeaderLine('Authorization');
        if (str_starts_with($header, 'Bearer '))
            $token = substr($header, 7);
    }

    // 3. Fallback: query parameter
    if (empty($token))
        $token = $request->getGet('token');

    if (empty($token))
        return service('response')->setStatusCode(401)
                    ->setJSON(['success' => false, 'message' => 'Token tidak ditemukan.']);

    try {
        $request->user = JwtHelper::verifyToken($token);
    } catch (\Exception $e) {
        return service('response')->setStatusCode(401)
                    ->setJSON(['success' => false, 'message' => 'Token tidak valid atau kedaluwarsa.']);
    }
}
```

Gambar 3. 28 *Backend*, Filter Autentikasi JWT

JwtAuthFilter.php adalah *middleware* yang berjalan sebelum setiap permintaan mencapai *controller*, berfungsi sebagai penjaga gerbang seluruh *endpoint* yang memerlukan autentikasi. Filter ini membaca token JWT dari tiga sumber secara berurutan dengan prioritas:

1) *httpOnly cookie* sigap_token sebagai sumber utama
2) *header Authorization*: Bearer sebagai *fallback* untuk kompatibilitas dan
3) *query parameter* token sebagai *fallback* terakhir untuk mendukung koneksi *Server-Sent Events* (SSE).
4) Apabila token tidak ditemukan, filter langsung mengembalikan respons HTTP 401 tanpa meneruskan permintaan ke *controller*.

Apabila token ditemukan namun tidak valid atau sudah kedaluwarsa, filter mengembalikan HTTP 401 dengan pesan yang sesuai. Token yang valid akan di-*decode* dan data *payload*-nya disimpan di objek $request->user sehingga dapat diakses oleh seluruh *controller* di hilir.

```php
public function panic(): ResponseInterface
{
    $formData = $this->request->getPost();
    $file     = $this->request->getFile('foto');
    $lat      = (float) ($formData['latitude']  ?? 0);
    $lng      = (float) ($formData['longitude'] ?? 0);

    $data = [
        'type'          => 'panic',
        'kode'          => $this->reportModel->generateKode(),
        'nama_pelapor'  => $formData['nama_pelapor']  ?? null,
        'no_hp_pelapor' => $formData['no_hp_pelapor'] ?? null,
        'deskripsi'     => $formData['deskripsi']     ?? '',
        'latitude'      => $lat,
        'longitude'     => $lng,
        'alamat'        => $this->getAddressFromCoords($lat, $lng),
        'tingkat'       => 'tinggi',
        'status'        => 'menunggu',
    ];

    if ($file?->isValid()) {
        $fileName      = $file->getRandomName();
        $file->move(WRITEPATH . 'uploads/reports/', $fileName);
        $data['foto']  = 'uploads/reports/' . $fileName;
    }

    $reportId     = $this->reportModel->insert($data);
    $targetTokens = $this->fcmTokenModel->getActiveTokensForRole(['admin', 'petugas']);
    if (!empty($targetTokens))
        $this->fcmPublisher->publishNewReport($this->reportModel->find($reportId), $targetTokens);

    return $this->response->setJSON([
        'success' => true,
        'message' => 'Laporan darurat berhasil dikirim.',
        'data'    => ['kode' => $data['kode']],
    ]);
}
```

Gambar 3. 29 *Backend*, Pembuatan Laporan Darurat

Fungsi *panic( )* pada ReportsController.php menangani alur pembuatan laporan darurat melalui *panic button* dalam satu transaksi pemrosesan yang terintegrasi. Fungsi ini menerima data formulir dari *request*, membangkitkan kode laporan unik berformat RPT-YYYY-XXXX melalui generateKode(), lalu mengonversi koordinat GPS (latitude/longitude) menjadi alamat yang dapat dibaca manusia menggunakan layanan *reverse geocoding* OpenStreetMap Nominatim melalui getAddressFromCoords(). Apabila pengguna menyertakan foto, berkas divalidasi, di-*rename* secara acak menggunakan getRandomName() untuk mencegah eksploitasi nama berkas, lalu disimpan ke direktori writable/uploads/reports/. Setelah laporan tersimpan ke basis data, sistem secara otomatis mengambil seluruh token FCM aktif milik petugas dan admin dari tabel fcm_tokens lalu mendistribusikan *Push notification* melalui FCMPublisher::publishNewReport(). Seluruh proses ini berjalan dalam satu pemanggilan fungsi tanpa interaksi tambahan dari pengguna.

```php
// app/Services/FCMPublisher.php
public function publishNewReport(array $report, array $tokens): bool
{
    if (!$this->initialized || empty($tokens)) return false;

    try {
        $notification = Notification::create(
            "Laporan Baru: " . $report['kode'],
            ($report['kategori'] ?? 'Darurat') . " — " . substr($report['deskripsi'], 0, 80) . '..
        );

        $message = CloudMessage::new()
            ->withNotification($notification)
            ->withData([
                'type'      => 'new_report',
                'report_id' => (string) $report['id'],
                'kode'      => $report['kode'],
                'status'    => $report['status'],
            ])
            ->withDefaultSounds();

        $result = $this->messaging->sendMulticast($message, $tokens);
        return $result->successes()->count() > 0;

    } catch (\Exception $e) {
        log_message('error', '[FCMPublisher] ' . $e->getMessage());
        return false;
    }
}
```

Gambar 3. 30 *Backend*, Layanan *Push notification*

FCMPublisher.php adalah *service class* yang mengenkapsulasi seluruh logika pengiriman *Push notification* ke perangkat pengguna melalui *Firebase Cloud Messaging* (FCM). Pada saat objek diinisialisasi melalui konstruktor, sistem memuat *service account key Firebase* dari berkas konfigurasi *Firebase*-service-account.json dan membuat koneksi ke *Firebase Admin SDK* menggunakan pustaka kreait/*Firebase*-php. Desain ini menggunakan pola *lazy initialization* dengan flag $initialized sehingga apabila berkas konfigurasi tidak tersedia, sistem tetap dapat berjalan tanpa *crash* dan hanya mencatat *Error* ke *log*. Fungsi publishNewReport() membangun objek notifikasi yang memuat judul (kode laporan), isi pesan (kategori dan deskripsi), serta *data payload* tambahan berisi metadata laporan (ID, kategori, status, *timestamp*) yang dibutuhkan *Frontend* untuk memperbarui tampilan secara langsung tanpa perlu melakukan *request* ulang ke server. Notifikasi dikirimkan secara *multicast* ke seluruh token FCM terdaftar dalam satu operasi pengiriman menggunakan sendMulticast().

4. Dokumentasi Implementasi

Proses implementasi *Emergency System* dilaksanakan secara bertahap mengikuti urutan prioritas fungsional yang memastikan setiap lapisan fondasi sistem selesai dan terverifikasi sebelum lapisan fitur di atasnya dibangun.

Tahap pertama difokuskan pada pembangunan fondasi infrastruktur:

*a.* Inisialisasi struktur proyek *Backend* dan *Frontend*
b. Konfigurasi *environment*
c. Pembuatan skema database melalui sistem migrasi
d. Serta implementasi sistem autentikasi jwt secara *end-to-end*.

Fondasi autentikasi dikerjakan pertama karena hampir seluruh fitur lain bergantung padanya, tanpa sistem login dan verifikasi token yang berfungsi dengan benar pengujian fitur-fitur selanjutnya tidak dapat dilakukan dengan valid.

Tahap kedua difokuskan pada implementasi modul inti pelaporan, yang merupakan fungsi paling fundamental dari *emergency system*. Pada tahap ini dikerjakan:

a. *Endpoint* pembuatan laporan (jalur normal dan *panic button*),
b. Logika penanganan unggahan foto dengan validasi format dan ukuran,
c. Serta pengambilan koordinat geolokasi dari *browser*.
d. Setiap *endpoint* diuji sebelum diintegrasikan dengan antarmuka *Frontend*.

Tahap ketiga adalah implementasi sistem komunikasi *real-time* berbasis *Firebase Cloud Messaging* (FCM). Pada tahap ini dikerjakan:

a. Konfigurasi *Firebase Project* dan pengunduhan *service account key* untuk digunakan oleh *Backend* melalui kreait/*Firebase*-php.
b. Implementasi FCMPublisher.php sebagai layanan pengiriman *Push notification* dari *Backend*, yang menggunakan *Firebase* Admin SDK untuk mengirimkan notifikasi secara bertarget berdasarkan token FCM yang tersimpan di tabel fcm_tokens.
c. Implementasi FcmController.php untuk menangani registrasi token FCM dari *browser* klien, termasuk penyimpanan dan pembaruan token di basis data.

d. Implementasi *Firebase*-messaging-sw.js sebagai *Service Worker* di sisi *Frontend* yang memungkinkan *browser* menerima *Push notification* dari FCM server Google.
e. Integrasi *Firebase* JS SDK (*Firebase* v11.0.0) pada *Frontend* melalui services/fcm.ts untuk proses permintaan izin notifikasi kepada pengguna, inisialisasi Service Worker, dan pengiriman token FCM ke *Backend*.
f. Pengujian dilakukan dengan membuka dua sesi *browser* secara bersamaan: satu sesi sebagai pelapor yang mengirim laporan, dan satu sesi sebagai petugas yang terdaftar tokennnya, untuk memverifikasi bahwa *Push notification* muncul secara otomatis di sisi petugas tanpa perlu menyegarkan halaman.

Tahap keempat adalah implementasi pengembangan fitur-fitur pendukung yang melengkapi pengalaman pengguna:

a. Tombol navigasi rute tercepat Google Maps
b. Visualisasi statistik menggunakan *Recharts* pada *dashboard*
c. Fitur pelacakan laporan via kode unik
d. Ekspor data ke format PDF dan Excel
e. Serta notifikasi *browser* menggunakan *Push notification* API.

Pada tahap ini juga dilakukan penyempurnaan antarmuka untuk memastikan seluruh halaman responsif dan nyaman digunakan pada perangkat smartphone maupun desktop.

F. Integrasi dan Pengujian (*Integration & Testing*)

Tahap ini bertujuan memverifikasi bahwa seluruh komponen sistem yang dibangun secara terpisah dapat berfungsi dengan benar sebagai satu kesatuan yang terintegrasi, serta memastikan SIRTADA memenuhi seluruh spesifikasi kebutuhan fungsional dan non-fungsional yang telah ditetapkan. Strategi pengujian dan integrasi sistem diorganisasikan ke dalam empat kategori yang saling melengkapi antara lain:

a. Pertama, *Unit Testing* dilaksanakan menggunakan PHPUnit 10.5 untuk komponen *Backend* (*CodeIgniter* 4) dan Vitest 4.1 bersama *React Testing Library* dan *Mock Service Worker* (MSW) untuk komponen *Frontend* (*React* 19 + TypeScript). Seluruh dependensi eksternal digantikan oleh

objek tiruan (*mock*) sehingga setiap unit diuji secara terisolasi tanpa koneksi basis data maupun layanan eksternal aktif.

b. Kedua, *Integration testing* dilaksanakan menggunakan *Postman Newman CLI* terhadap sistem yang berjalan secara nyata, mencakup 41 *HTTP request* pada 5 kategori skenario yang meliputi *happy path* maupun *Error path*.

c. Ketiga, *Performance Testing* dilaksanakan menggunakan *Grafana k6* dengan simulasi *virtual user* secara bertahap untuk mengukur *response time* dan kapasitas sistem di bawah kondisi beban nyata. Keempat, *Security Testing* dilaksanakan dengan mengirimkan *payload* SQL Injection dan skrip XSS melalui *endpoint* yang tersedia untuk memverifikasi respons sistem terhadap masukan berbahaya.

d. Keempat, *User Acceptance Testing* (UAT) dilaksanakan dengan melibatkan pengguna akhir secara langsung untuk memverifikasi bahwa sistem memenuhi kebutuhan operasional dari perspektif pengguna nyata. Pemilihan responden UAT dilakukan menggunakan metode *purposive sampling*, yaitu teknik pengambilan sampel non-probabilistik yang memilih partisipan secara selektif berdasarkan keterwakilan peran pengguna sistem (Tongco, 2022). Kriteria pemilihan ditetapkan berdasarkan tiga peran utama: dua responden dari kalangan masyarakat pelapor, dua responden dari petugas lapangan Dinas Pemadam Kebakaran Kabupaten Kediri, dan satu responden dari admin sistem. Hasil pengujian unit dan pengujian integrasi disajikan secara lengkap pada Bab IV subbab B.4 dan B.5 sebagai bagian dari data hasil penelitian.

G. Pemeliharaan (*Maintenance*)

Tahap ini mencakup perbaikan *Error* dan *Bug* yang ditemukan pasca-implementasi sebelum sistem dinyatakan siap digunakan.

**1.** Perbaikan *Bug* / *Error*

Sebagaimana telah didokumentasikan secara rinci pada subbagian temuan dan perbaikan selama pengujian iteratif, dua kendala teknis berhasil diidentifikasi selama pelaksanaan pengujian dan diselesaikan dalam fase ini sebelum sistem dinyatakan siap digunakan.

a. Perbaikan *Infinite Loading* pada Fungsi *restoreSession* Masalah i*nfinite loading* diselesaikan dengan menambahkan blok *finally* pada fungsi *restoreSession*, sehingga status loading selalu diperbarui dengan benar meskipun permintaan ke server mengalami kegagalan.
b. Penambahan *Fallback* Manual pada Formulir Pelaporan Ketika pengguna menolak izin akses geolokasi, sistem kini menyediakan opsi pengisian alamat secara manual sebagai alternatif, disertai penguatan validasi skema Zod dan penyesuaian tampilan pesan kesalahan agar lebih informatif.

2. Pembaruan Fitur

Untuk menyesuaikan dengan kebutuhan operasional yang berkembang, direncanakan dua pengembangan fitur lanjutan sebagai berikut:

a. Fitur *Multiple* Foto per Laporan: Mengubah batasan satu foto per laporan menjadi dokumentasi yang lebih komprehensif. Perubahan dilakukan dengan memperbarui skema basis data, yaitu menerapkan tabel report_photos dengan relasi *one-to-many* terhadap tabel reports.
b. Fitur *Refresh token* Otomatis: Mengganti masa berlaku tunggal JWT (24 jam) dengan pola *access* token dan *refresh token*. Pembaruan ini memungkinkan pembaruan sesi secara otomatis tanpa interupsi, sehingga meningkatkan kenyamanan admin dan petugas yang bekerja dalam durasi panjang.

3. Saran Pengembangan Sistem Selanjutnya

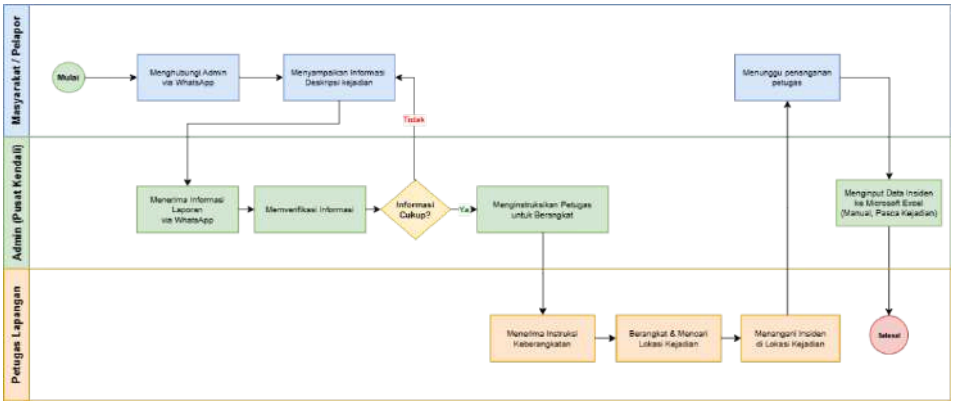
Direkomendasikan empat pengembangan lanjutan untuk *Emergency System* SIRTADA.

a. Pengembangan sistem berbasis web menjadi *Progressive Web App* (PWA)
b. Pengembangan sistem dengan Otomatisasi Alur Kerja menggunakan n8n *Platform open-source*.
c. Pengembangan sistem yang terintegrasi dengan layanan pelaporan darurat instansi lain, seperti instansi Kepolisian, Rumah sakit, BPBD dll.
d. Pengembangan sistem dengan penambahan Peta *Heatmap* interaktif untuk sebaran insiden.

# BAB IV
# HASIL DAN PEMBAHASAN

## A. Data Hasil Penelitian

### 1. Proses Bisnis yang Sedang Berjalan



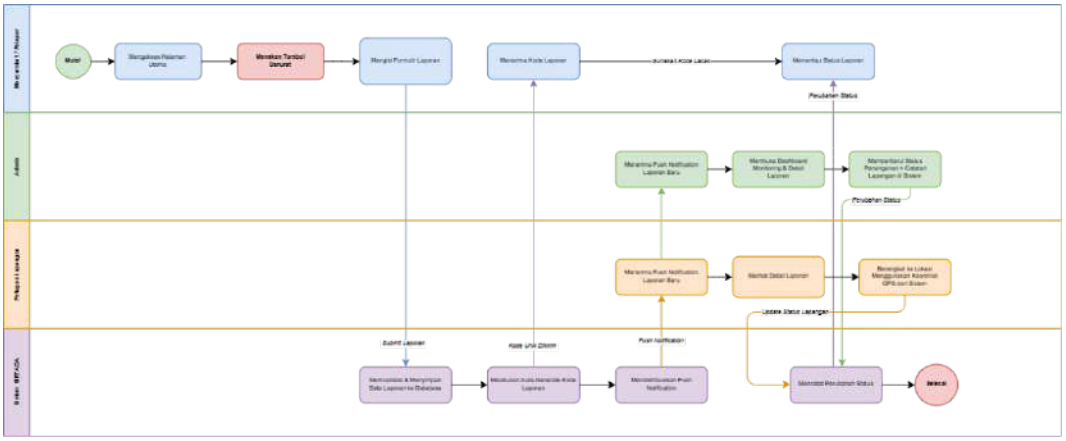

Gambar 4. 1 Proses bisnis saat ini

Berdasarkan observasi dan wawancara dengan Dinas Pemadam Kebakaran Kabupaten Kediri, pelaporan darurat sebelumnya masih dilakukan melalui *WhatsApp* dan pencatatan insiden pasca kejadian melalui *Microsoft Excel*. Masyarakat mengirim laporan lewat pesan pribadi *WhatsApp*, lalu petugas mencatatnya secara manual setelah penanganan selesai. Proses ini menyebabkan informasi sering tidak terstruktur, lokasi kejadian kurang akurat, dan riwayat penanganan sulit ditelusuri karena monitoring belum terintegrasi. Alur proses bisnis lama dapat dijelaskan sebagai berikut:

a. Masyarakat menghubungi petugas melalui *WhatsApp*.
b. Petugas menerima laporan dan menanyakan detail kejadian.
c. Petugas mencatat laporan secara manual.
d. Petugas menuju lokasi kejadian berdasarkan informasi yang diberikan.
e. Setelah penanganan selesai, laporan dicatat kembali ke *Microsoft Excel* sebagai arsip.

Berdasarkan proses tersebut, dapat disimpulkan bahwa sistem lama masih memiliki keterbatasan pada kecepatan pelaporan, akurasi lokasi, monitoring laporan, dan dokumentasi data insiden.

2. Proses Bisnis yang Dikembangkan



Gambar 4. 2 Proses bisnis usulan

Sistem SIRTADA mentransformasi penanganan laporan darurat dari proses manual dan tidak terpusat menjadi alur yang terdigitalisasi, terdokumentasi, dan dapat dipantau secara *real-time* oleh tiga aktor utama: masyarakat, petugas, dan admin.

Masyarakat mengakses SIRTADA melalui *browser* tanpa registrasi, lalu mengirim laporan melalui *panic button* dengan mengisi nama, nomor telepon, deskripsi, foto (opsional), dan koordinat lokasi yang terdeteksi otomatis via *Geolocation API*. Sistem menerbitkan kode unik RPT-YYYY-XXXX sebagai tanda terima sekaligus alat pelacak, dan secara bersamaan mendistribusikan *Push notification* ke seluruh petugas dan admin terdaftar melalui *Firebase Cloud Messaging* dalam hitungan detik.

Di sisi petugas dan admin, laporan langsung tampil di *dashboard* secara *real-time*. Mereka dapat membuka detail laporan dan memperbarui status penanganan secara bertahap yaitu: menunggu, diproses, menuju lokasi dan yang terakhir selesai, disertai catatan lapangan di setiap tahap. Setiap perubahan status otomatis tercatat dalam tabel incident_logs sebagai audit trail yang tidak dapat diubah.

Masyarakat dapat memantau perkembangan penanganan secara mandiri melalui fitur lacak laporan menggunakan kode unik mereka, tanpa perlu menghubungi dinas secara berulang.

3. Implementasi sistem

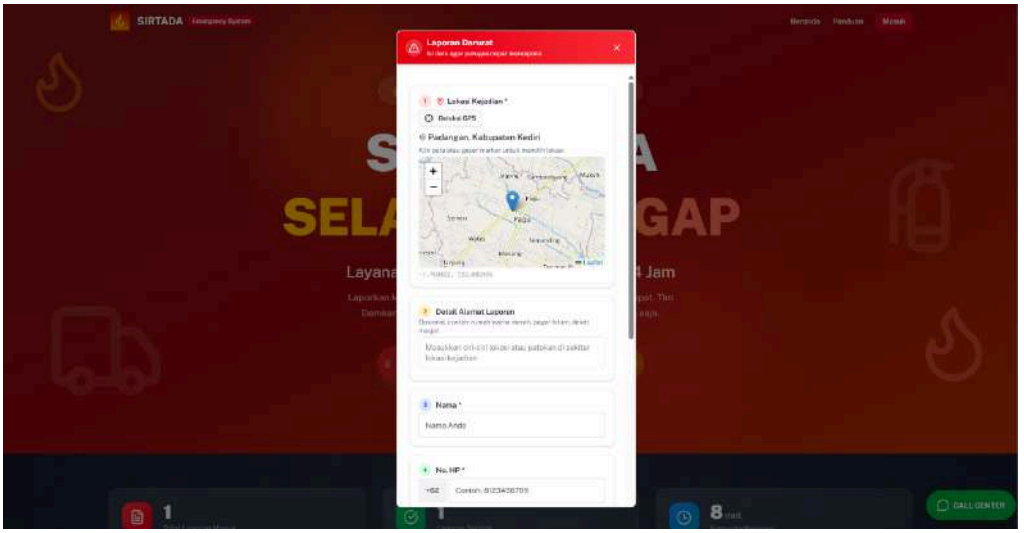
Implementasi SIRTADA dilakukan secara paralel pada empat domain yang terhubung melalui kontrak REST API dengan *Backend* berbasis *CodeIgniter* 4, *Frontend* berbasis *React*.js 19 dengan TypeScript, dan *WhatsApp-Gateway* berbasis Node.js sebagai *microservice* notifikasi *WhatsApp*, ketiganya dikerjakan di lingkungan lokal sebelum dipindahkan ke server produksi.

Pada domain *Backend*, dibangun enam controller REST API, lima model basis data, dua filter autentikasi dan otorisasi (JWT + RBAC), dua service yaitu FCMPublisher untuk distribusi *Push notification* berbasis *Firebase Cloud Messaging* dan *WhatsAppService* untuk pengiriman notifikasi laporan darurat ke grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri melalui *WhatsApp Gateway* berbasis *Baileys*, serta satu helper JWT. Basis data MySQL terdiri dari lima tabel yaitu tabel users, *role*s, reports, incident_logs, dan fcm_tokens, yang dibangun melalui mekanisme migrasi *CodeIgniter* 4 untuk menjamin konsistensi struktur di seluruh lingkungan.

Pada domain *Frontend*, dibangun sepuluh halaman utama yang terbagi menjadi halaman publik, admin, dan petugas, didukung oleh *reusable components*, tiga context untuk manajemen state global, konfigurasi Axios dengan interceptor terpusat untuk autentikasi dan penanganan kesalahan, serta integrasi *Firebase* SDK untuk penerimaan *Push notification* di *browser* melalui *Service Worker*. Berikut adalah tampilan antarmuka sistem SIRTADA yang berhasil diimplementasikan:

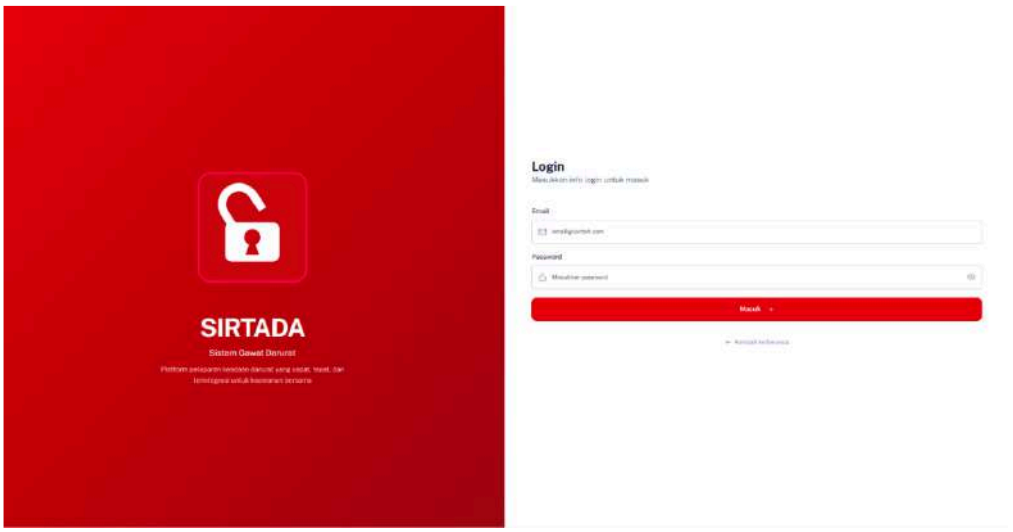
Gambar 4. 3 Tampilan Halaman Utama (Home Page)

Halaman utama sistem SIRTADA dapat diakses oleh publik tanpa perlu melakukan login. Halaman ini menampilkan *navigation bar* di bagian atas, *hero section* dengan tombol *panic button* yang diletakkan secara mencolok di tengah layar sebagai elemen utama, *carousel* yang memandu pengguna melalui lima langkah alur pelaporan, serta *statistics section* yang menampilkan jumlah total laporan masuk, laporan selesai, dan rata-rata respons secara transparan kepada masyarakat.



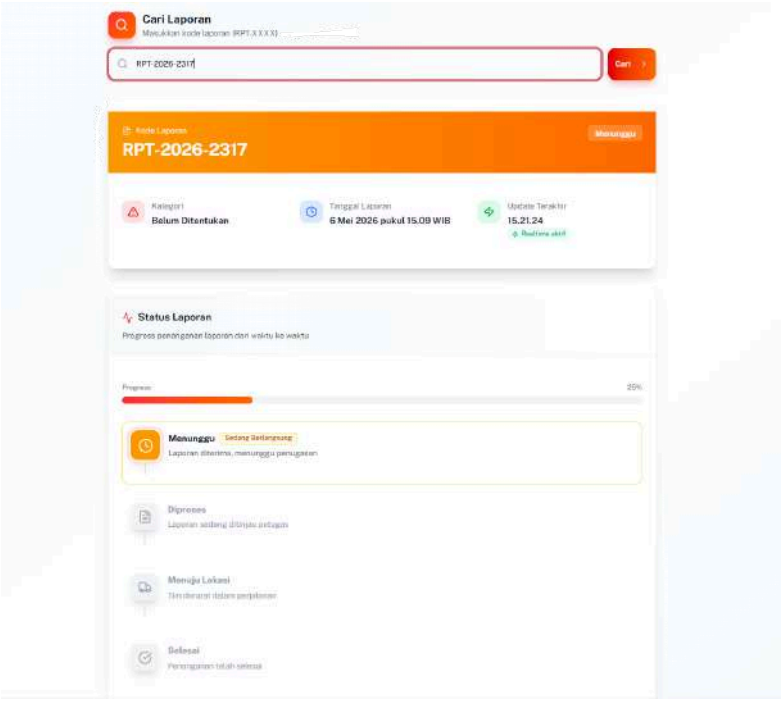

Gambar 4. 4 Tampilan Formulir Laporan Darurat (*Panic button Form*)

Tampilan formulir laporan darurat yang muncul setelah pengguna menekan tombol *panic button*. Formulir ini memuat *field* nama pelapor, nomor telepon, deskripsi kejadian, serta deteksi koordinat lokasi secara otomatis melalui *Geolocation API* yang divisualisasikan pada peta *Leaflet.js*. Terdapat pula opsi unggah foto sebagai dokumentasi visual insiden. Seluruh proses pengisian dapat diselesaikan dalam satu interaksi tanpa memerlukan registrasi akun terlebih dahulu.



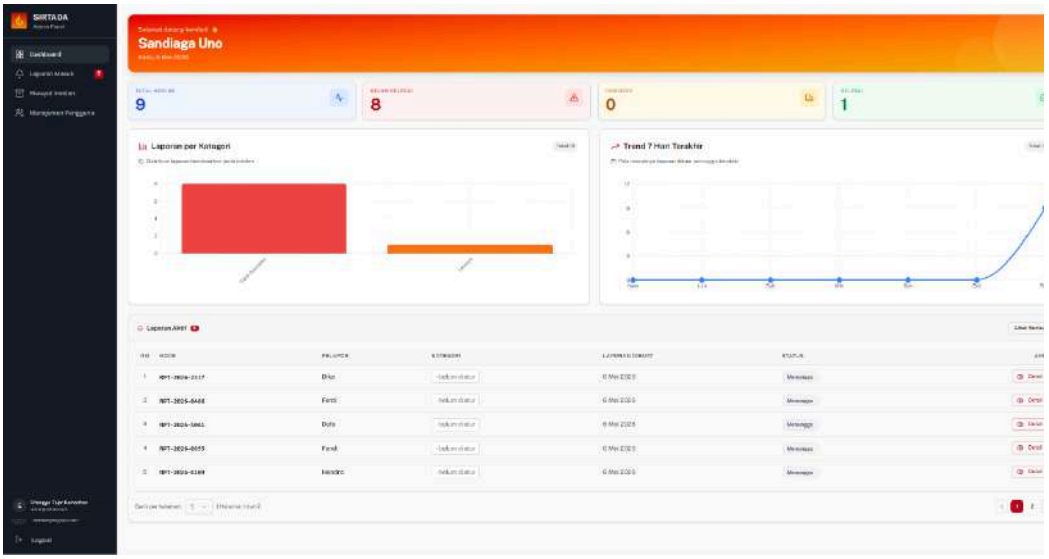

Gambar 4. 5 Tampilan Halaman Login

Halaman login berfungsi sebagai gerbang autentikasi bagi admin dan petugas untuk mengakses sistem. Formulir terdiri dari dua *field* yaitu email dan kata sandi yang disusun secara vertikal di tengah halaman. Sistem menerapkan validasi *client-side* dan menampilkan pesan kesalahan secara *inline* apabila kredensial yang dimasukkan tidak valid. Setelah autentikasi berhasil, pengguna akan diarahkan secara otomatis ke halaman *dashboard* sesuai dengan perannya masing-masing.



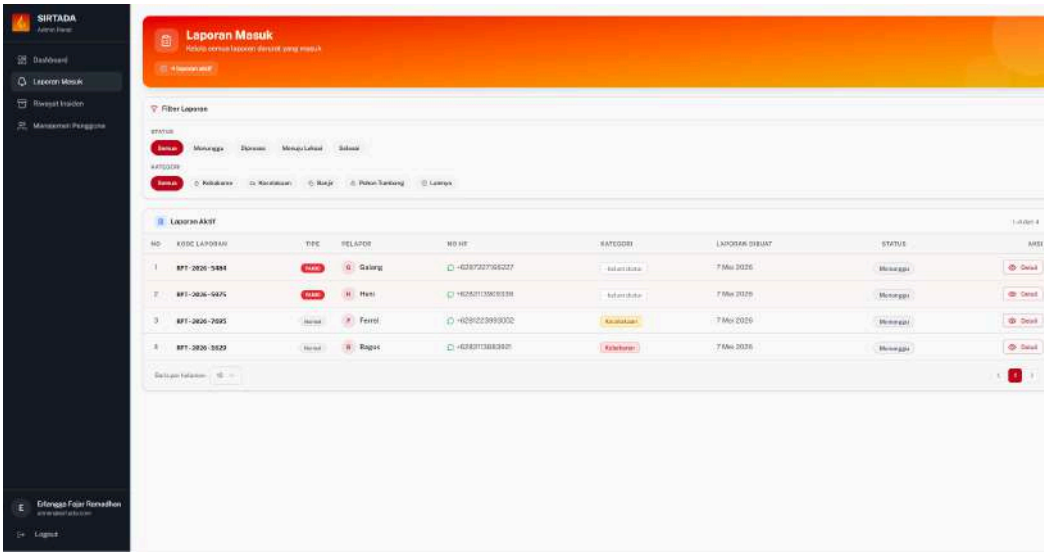

Gambar 4. 6 Tampilan Halaman Lacak Laporan beserta *Timeline Status*

Halaman lacak laporan memungkinkan masyarakat memantau perkembangan penanganan insiden secara mandiri menggunakan kode unik berformat RPT-YYYY-XXXX yang diterima setelah pengiriman laporan. Halaman ini menampilkan *progress timeline* vertikal yang memvisualisasikan empat tahapan penanganan secara kronologis yaitu menunggu, diproses, menuju lokasi, dan selesai. Tahap yang telah dicapai ditandai dengan warna aktif, sementara tahap yang belum tercapai ditampilkan dalam kondisi abu-abu.



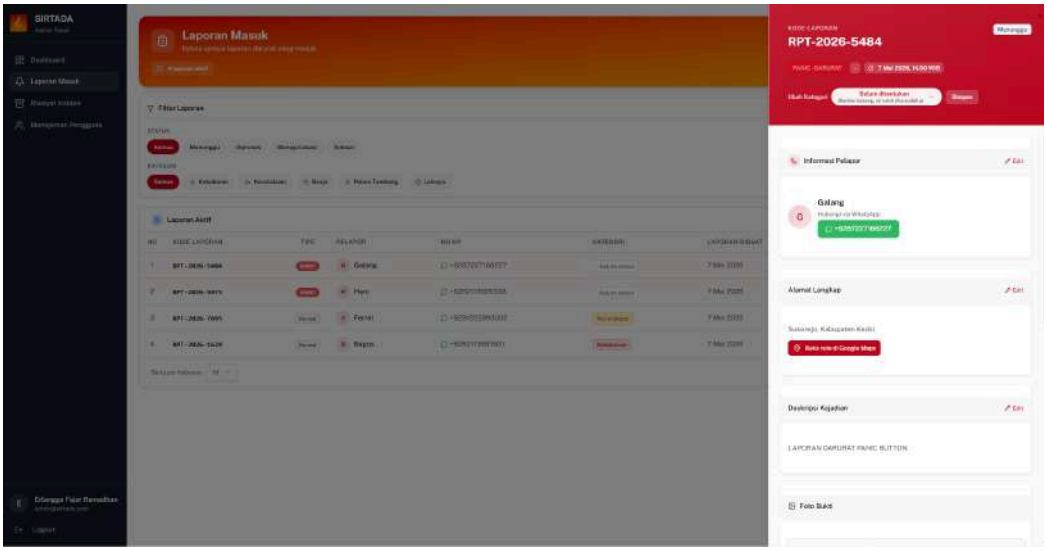

Gambar 4. 7 Tampilan *Dashboard* Admin dengan Statistik

*Dashboard* admin menampilkan seluruh informasi operasional kritis sistem dalam satu halaman terpusat. Bagian atas memuat lima *metric card* yang menampilkan jumlah laporan hari ini, menunggu, diproses, menuju lokasi, dan selesai. Di bawahnya terdapat dua panel grafik yang berdampingan, yaitu *line chart* untuk visualisasi tren laporan tujuh hari terakhir dan *bar chart* untuk distribusi laporan per kategori insiden. Seluruh data diperbarui secara *real-time* saat laporan baru masuk melalui mekanisme *Push notification* berbasis *Firebase Cloud Messaging*.




Gambar 4. 8 Tampilan Halaman Laporan Masuk

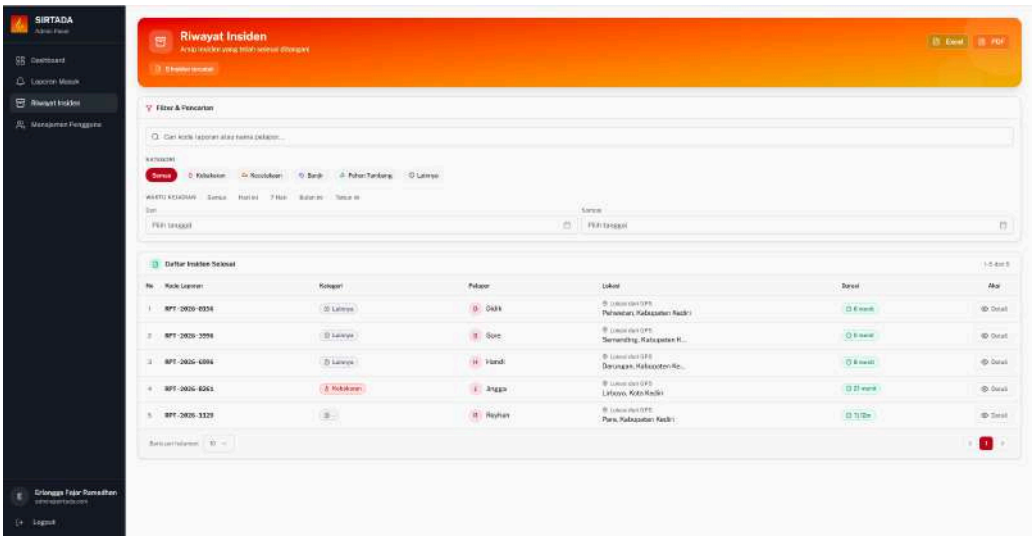
Halaman laporan masuk menampilkan seluruh laporan aktif dalam format tabel interaktif yang dilengkapi dengan *toolbar* filter di bagian atas. Pengguna dapat menyaring data berdasarkan status, kategori, dan rentang tanggal. Setiap baris pada tabel bersifat interaktif sehingga admin atau petugas dapat langsung membuka panel detail laporan dengan satu klik tanpa berpindah halaman.




Gambar 4. 9 Tampilan Halaman Detail Laporan

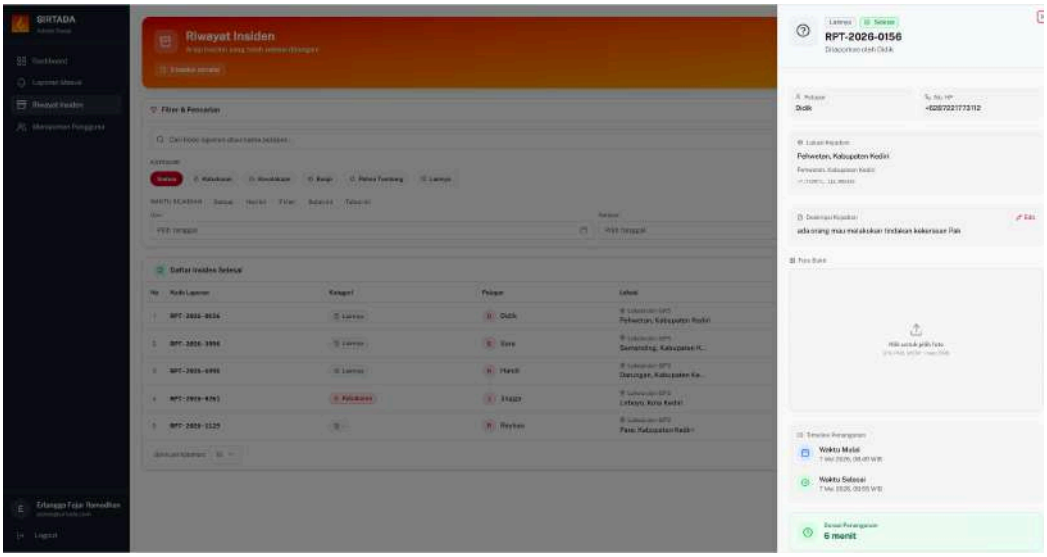
Panel detail laporan menampilkan informasi lengkap mengenai suatu insiden, mencakup identitas pelapor, deskripsi kejadian, tombol navigasi menuju lokasi kejadian, foto dokumentasi, serta kontrol pembaruan status yang hanya menampilkan opsi tahapan logis berikutnya untuk mencegah kesalahan input. Di bagian bawah terdapat *activity log timeline* yang mencatat seluruh riwayat

perubahan status secara kronologis beserta nama petugas yang melakukan tindakan dan *timestamp*-nya.




Gambar 4. 10 Tampilan Halaman Riwayat Insiden

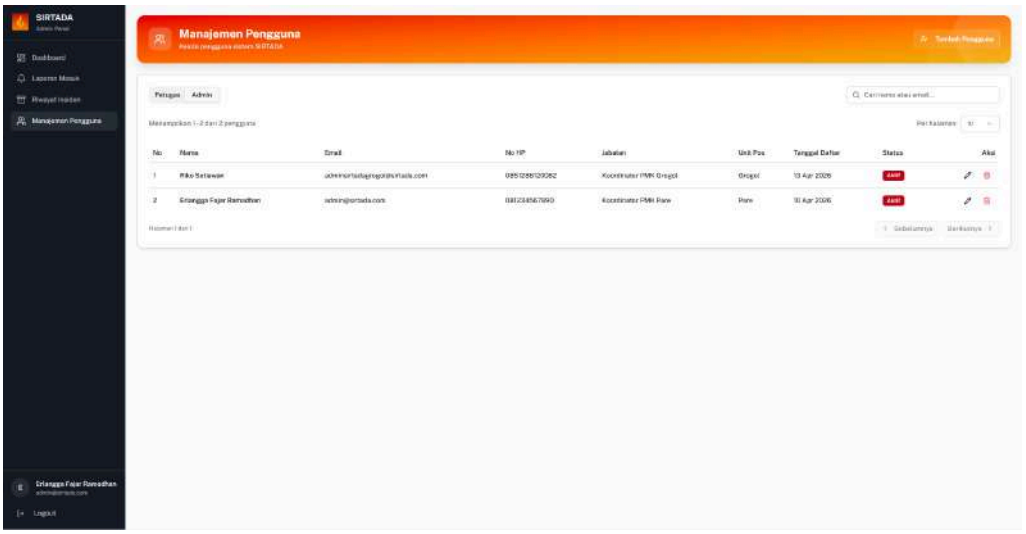
Halaman riwayat insiden berfungsi sebagai arsip terpusat yang menampilkan seluruh laporan berstatus selesai. Halaman ini dilengkapi dengan empat filter simultan yaitu rentang tanggal, kategori, status akhir, dan *search bar*, serta fitur ekspor data ke format PDF dan Excel untuk kebutuhan pelaporan administratif. Kolom durasi penanganan ditampilkan secara otomatis untuk keperluan evaluasi kinerja operasional.




Gambar 4. 11 Tampilan Halaman Detail Riwayat Insiden

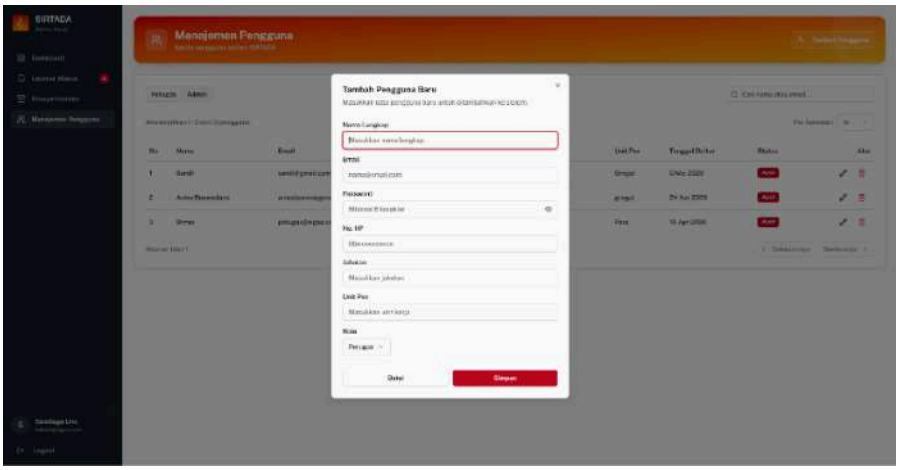
Detail riwayat insiden ditampilkan dalam modal dialog yang bersifat *read-only* sebagai bentuk *immutable audit trail*. Modal ini memuat ringkasan identitas laporan berupa kode, kategori, tingkat urgensi, dan *badge* status,

dilengkapi dengan data pelapor, deskripsi kejadian, peta mini lokasi, serta galeri foto dengan fitur *lightbox*. Bagian bawah modal menyajikan *activity log timeline* vertikal yang mencatat seluruh riwayat perubahan, nama petugas, catatan lapangan, dan *timestamp* secara kronologis.



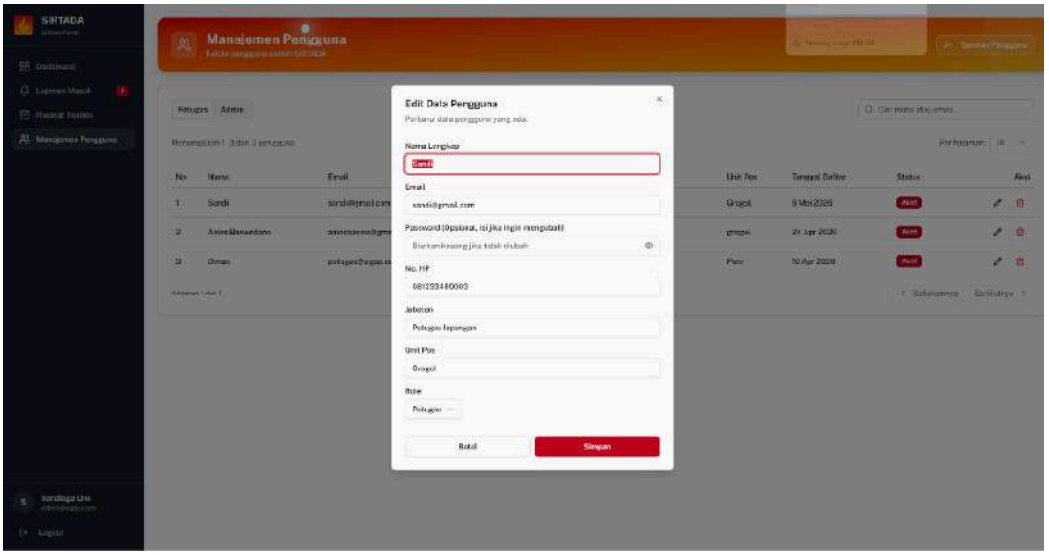

Gambar 4. 12 Tampilan Halaman Manajemen Pengguna

Halaman manajemen pengguna merupakan fitur eksklusif admin yang menampilkan seluruh akun terdaftar dalam sistem. Halaman ini mendukung operasi CRUD yaitu tambah, lihat, ubah, dan hapus akun petugas tanpa perlu berpindah halaman. Bagian atas memuat *toolbar* dengan *search bar* dan filter peran pengguna, sementara data akun ditampilkan dalam tabel dengan informasi foto profil, nama, email, nomor telepon, peran, jabatan, dan unit, disertai tombol aksi pada setiap baris.



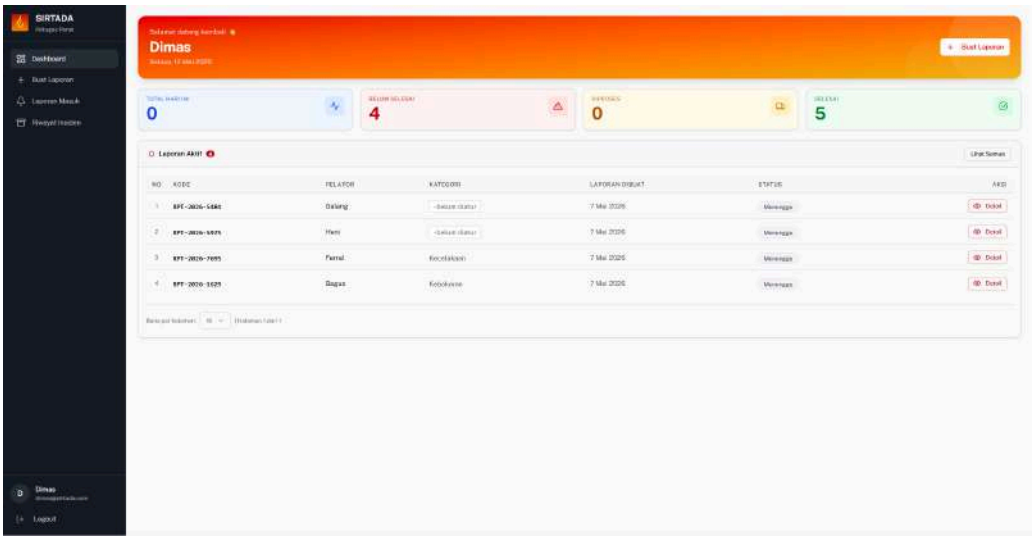

Gambar 4. 13 Tampilan Modal Tambah Pengguna

Modal tambah pengguna menampilkan formulir yang memuat delapan *field* input yaitu nama lengkap, email, kata sandi, konfirmasi kata sandi, nomor telepon, peran, jabatan, dan unit penugasan. Setiap *field* dilengkapi validasi *client-side* yang menampilkan pesan kesalahan secara *inline* tanpa menunggu formulir dikirim. Modal ini muncul di atas halaman tanpa mengalihkan admin ke halaman terpisah sehingga alur kerja tetap efisien.




Gambar 4. 14 Tampilan Modal Edit Data Pengguna

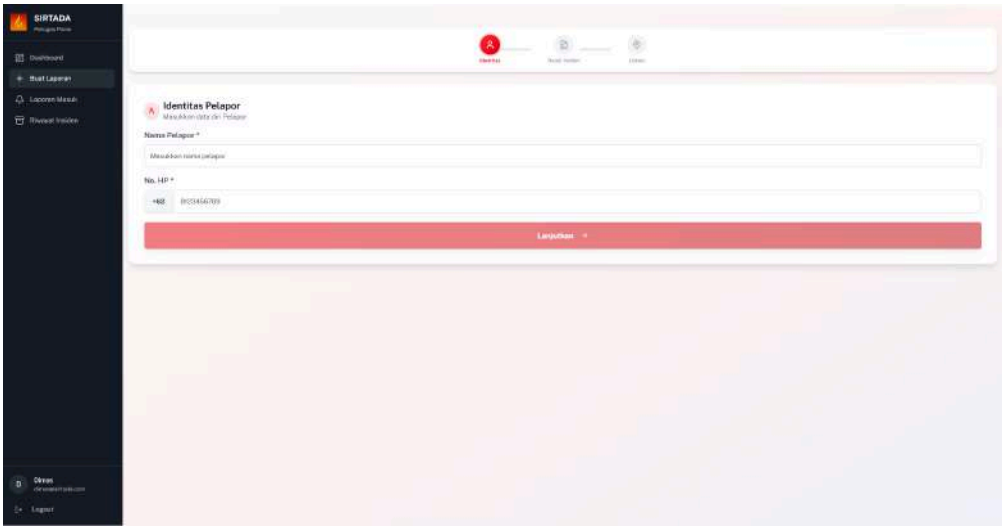
Modal edit pengguna memiliki struktur yang identik dengan modal tambah pengguna, dengan perbedaan utama bahwa seluruh *field* telah terisi otomatis (*pre-populated*) dengan data akun yang sedang diedit.




Gambar 4. 15 Tampilan *Dashboard* Petugas

*Dashboard* petugas berfokus pada informasi yang relevan dengan tugas penanganan insiden di lapangan. Bagian atas menampilkan tiga *metric card*

berisi jumlah laporan aktif, menunggu respons, dan selesai hari ini. Di bawahnya, tabel laporan aktif menampilkan seluruh laporan berstatus menunggu dan diproses secara *real-time*, dengan aksen warna merah pada laporan berkategori darurat tinggi agar petugas dapat mengidentifikasi prioritas penanganan secara langsung.



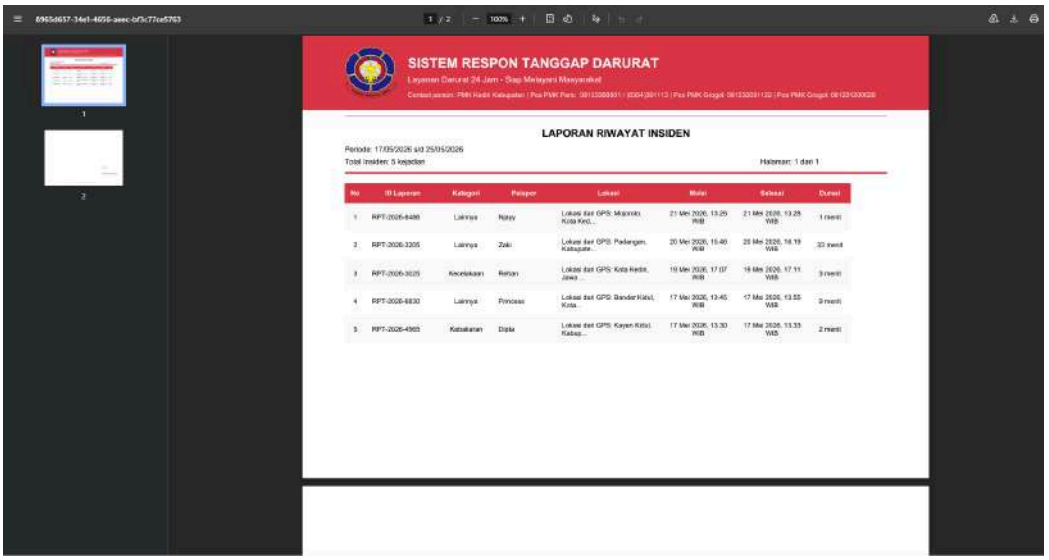

Gambar 4. 16 Tampilan Buat Laporan (Petugas)

Halaman buat laporan pada antarmuka petugas menggunakan komponen formulir yang sama dengan formulir pelaporan publik, sehingga konsistensi antarmuka terjaga. Perbedaan utamanya adalah data pelapor tersinkronisasi secara otomatis dengan akun petugas yang sedang *login*. Fitur ini memungkinkan petugas mendokumentasikan insiden yang ditemukan langsung di lapangan tanpa bergantung pada laporan dari masyarakat, dengan dukungan deteksi lokasi otomatis dan input suara melalui *Web Speech API*.


Gambar 4. 17 Tampilan Notifikasi Push FCM pada *Browser*

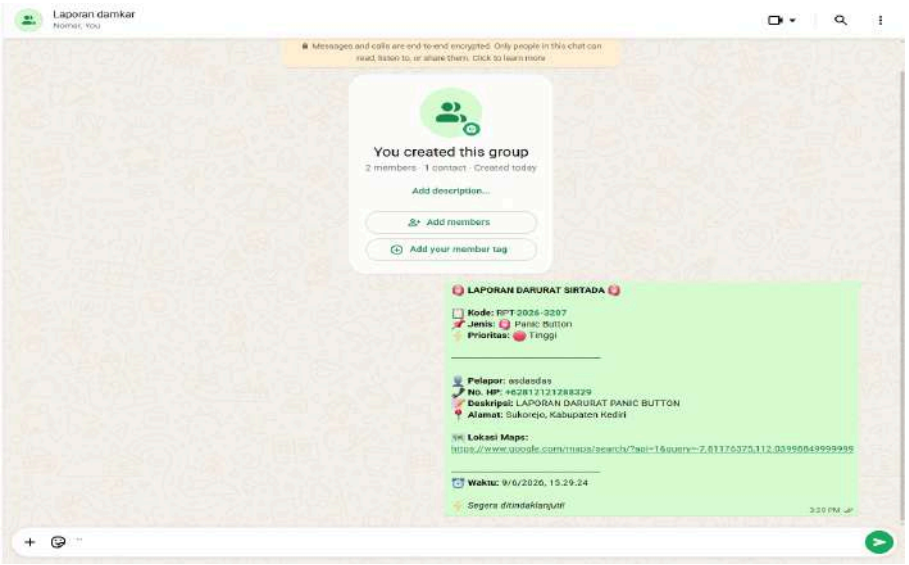
Tampilan *Push notification* yang diterima oleh *browser* admin atau petugas saat laporan darurat baru masuk ke sistem. Notifikasi ini dikirimkan melalui infrastruktur *Firebase Cloud Messaging* (FCM) dengan latensi dalam rentang 1–3 detik sejak laporan dikirimkan oleh pelapor. Notifikasi muncul

secara otomatis meskipun tab aplikasi tidak sedang aktif di latar depan *browser*, memastikan petugas tidak melewatkan satu pun laporan darurat yang masuk.




Gambar 4. 18 Tampilan Hasil Ekspor Data ke Format PDF

Tampilan berkas PDF yang dihasilkan dari fitur ekspor data laporan. Berkas ini memuat data laporan dalam format tabel yang rapi dan terbaca, mencakup informasi kode laporan, kategori, pelapor, lokasi, status, dan waktu penanganan. Fitur ekspor diimplementasikan menggunakan *Library jsPDF* dan *jspdf-autotable* di sisi *Frontend*, sehingga proses pembuatan berkas dilakukan secara langsung di *browser* tanpa memerlukan pemrosesan tambahan dari server.




Gambar 4. 19 Tampilan Notifikasi Pesan Darurat via *Whatsapp Gateway*

Tampilan pesan notifikasi laporan darurat yang dikirimkan secara otomatis oleh sistem ke grup *WhatsApp* Dinas Pemadam Kebakaran Kabupaten Kediri melalui *WhatsApp Gateway* berbasis *Baileys* sesaat setelah laporan berhasil tersimpan di basis data. Notifikasi ini berperan sebagai kanal distribusi sekunder yang melengkapi *Push notification* berbasis FCM, memastikan informasi darurat tetap tersampaikan kepada seluruh anggota grup secara serentak tanpa bergantung pada status aktif *browser* perangkat petugas. Pesan dikemas dalam format terstruktur yang memuat kode laporan unik berformat RPT-YYYY-XXXX, jenis insiden, tingkat prioritas, identitas dan nomor telepon pelapor, deskripsi kejadian, alamat, serta pranala *Google Maps* dengan koordinat GPS yang terdeteksi secara otomatis, dilengkapi *timestamp* pengiriman dan instruksi tindak lanjut. Pendekatan notifikasi berbasis grup ini memungkinkan seluruh petugas yang tergabung menerima informasi darurat secara bersamaan tanpa memerlukan pendaftaran token FCM per perangkat, sekaligus memanfaatkan infrastruktur komunikasi *WhatsApp* yang telah lazim digunakan oleh Dinas Pemadam Kebakaran Kabupaten Kediri.

4. Hasil Pengujian Unit

Pengujian unit bertujuan memastikan setiap unit atau fungsi terkecil dari kode program bekerja secara benar dalam kondisi terisolasi, tanpa ketergantungan pada basis data, layanan eksternal, maupun komponen lain yang aktif. Pengujian mencakup delapan komponen utama: empat komponen *Frontend* (AuthContext, Axios Interceptor, PanicButton, TrackReportPage) yang diuji menggunakan Vitest 4.1 bersama *React Testing Library* dan *Mock Service Worker* (MSW), serta empat komponen *Backend* (AuthController, ReportsController, JwtAuthFilter, ReportModel) yang diuji menggunakan PHPUnit 10.5 dengan isolasi penuh melalui *mock* seluruh dependensi. Total terdapat 63 *test case* yang mencakup skenario *happy path* dan skenario kesalahan (*Error path*). Distribusi kasus uji, hasil eksekusi, serta persentase keberhasilan dari masing-masing komponen disajikan pada Tabel 4.1 berikut.

48

Tabel 4. 1 Rekap Pengujian Unit

| Komponen | Total Kasus | Lulus | Gagal | Tingkat Kelulusan |
|---|---|---|---|---|
| *Frontend AuthContext* | 6 | 6 | 0 | 100% |
| *Frontend Axios Interceptor* | 6 | 6 | 0 | 100% |
| *Frontend PanicButton* | 9 | 9 | 0 | 100% |
| *Frontend TrackReportPage* | 7 | 7 | 0 | 100% |
| *Backend AuthController* | 10 | 10 | 0 | 100% |
| *Backend ReportsController* | 9 | 9 | 0 | 100% |
| *Backend JwtAuthFilter* | 7 | 7 | 0 | 100% |
| *Backend ReportModel* | 9 | 9 | 0 | 100% |
| Total | 63 | 63 | 0 | 100% |

Hasil pengujian unit menunjukkan seluruh 63 *test case* lulus dengan tingkat kelulusan 100%. Pengujian *Backend* dengan PHPUnit menghasilkan 85 *assertion* berhasil dalam waktu 4,5 detik menggunakan memori 16 MB, sedangkan pengujian *Frontend* dengan Vitest menghasilkan 28 *test case* berhasil dalam 73 detik pada 4 berkas uji. Modul autentikasi berhasil memverifikasi kredensial menggunakan JWT yang disimpan sebagai *httpOnly cookie* dan mencegah akses tidak sah berdasarkan peran pengguna. Modul pelaporan darurat berhasil memproses pengiriman laporan secara lengkap mulai dari deteksi geolokasi otomatis, *generate* kode unik berformat RPT-YYYY-XXXX, penyimpanan ke basis data, hingga distribusi *Push notification* via *Firebase* Cloud Messaging. Modul pengelolaan status berhasil memperbarui status penanganan sekaligus mencatat riwayat perubahan pada tabel *incident_logs* sebagai *audit trail*. Modul pelacakan laporan berhasil menampilkan status terkini beserta *timeline* penanganan kepada masyarakat tanpa memerlukan autentikasi.

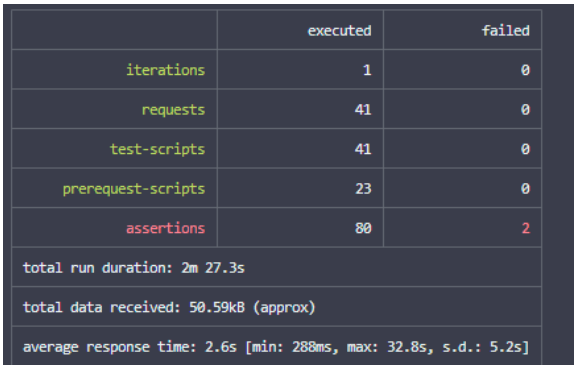
5. Hasil Pengujian Integrasi

Pengujian integrasi bertujuan memverifikasi bahwa seluruh komponen sistem, meliputi *Frontend React*.js, *Backend CodeIgniter* 4, basis data MySQL, dan *Firebase* Cloud Messaging tetap berfungsi dengan benar ketika digabungkan dan berkomunikasi satu sama lain pada sistem yang berjalan secara nyata. Pengujian dilaksanakan menggunakan *Postman Newman CLI*

terhadap 41 *HTTP request* yang diorganisasikan dalam 5 kategori skenario, dengan total 80 *assertion* yang mencakup skenario *happy path* maupun skenario kesalahan (*Error path*). Rincian kategori integrasi, jumlah *request*, *assertion*, dan status kelulusan disajikan pada Tabel 4.2 berikut.

Tabel 4. 2 Rekap Pengujian Integrasi

| Kategori Integrasi | Request | *Assertion* | Lulus | Gagal | Keterangan |
|---|---|---|---|---|---|
| *Authentication Integration* | 8 | 14 | 14 | 0 | Login, logout, /me valid & invalid |
| *Report Creation Integration* | 10 | 18 | 17 | 1 | *Panic button*, track, validasi field |
| *Status Update Integration* | 6 | 12 | 12 | 0 | Siklus penuh menunggu→selesai |
| *Database Integrity* | 10 | 22 | 22 | 0 | Field integrity, CRUD, audit log |
| *Security Integration* | 7 | 14 | 13 | 1 | SQL injection, XSS, JWT tampered |
| **Total** | **41** | **80** | **78** | **2** | **Durasi: 2 menit 27,3 detik** |



| | executed | failed |
|---|---|---|
| iterations | 1 | 0 |
| requests | 41 | 0 |
| test-scripts | 41 | 0 |
| prerequest-scripts | 23 | 0 |
| assertions | 80 | 2 |

total run duration: 2m 27.3s

total data received: 50.59kB (approx)

average response time: 2.6s [min: 288ms, max: 32.8s, s.d.: 5.2s]

Gambar 3. 31 Hasil Uji Postman CLI

Hasil eksekusi *Postman Newman CLI* menunjukkan 41 *request* berhasil dieksekusi dalam total durasi 2 menit 27,3 detik dengan 80 *assertion* yang diperiksa. Dari 80 *assertion* tersebut, 78 dinyatakan lulus. Dari 2 *assertion* yang tidak lulus satu merupakan *false failure* skrip pengujian mengasumsikan celah RBAC yang ternyata tidak terbukti ada sehingga sistem justru terbukti memblokir akses tidak sah secara benar dengan mengembalikan HTTP 400, dan satu lainnya merupakan temuan kerentanan *payload* XSS. Rata-rata *response time* keseluruhan adalah 1.032ms, dengan nilai minimum 192ms pada *endpoint* autentikasi dan maksimum 4.000ms pada *endpoint* POST /api/reports/panic yang melibatkan distribusi notifikasi *Firebase* Cloud Messaging dan *WhatsApp Gateway* secara sinkron setelah data laporan tersimpan. Seluruh alur integrasi utama dari pengiriman laporan darurat oleh masyarakat, pembaruan status oleh petugas, pencatatan *audit trail* ke tabel *incident_logs*, hingga pelacakan laporan

secara publik berhasil diverifikasi dan berfungsi sesuai spesifikasi yang telah ditetapkan.

6. Hasil Pengujian *Blackbox* & *UAT*

a. *Blackbox testing*

Tabel 4. 3 Hasil Pengujian *Blackbox*

| No | Modul | Lulus | Gagal | Keterangan |
|---|---|---|---|---|
| 1 | Autentikasi login admin dan petugas | √ | – | JWT berhasil diterbitkan dan disimpan sebagai *httpOnly cookie* serta *role* masyarakat ditolak (HTTP 403) |
| 2 | Pelaporan darurat melalui *panic button* | √ | – | Geolokasi otomatis, kode laporan unik RPT-YYYY-XXXX, dan notifikasi FCM berfungsi normal |
| 3 | Pengelolaan status laporan oleh petugas | √ | – | Pembaruan status dan pencatatan *audit trail* ke tabel *incident_logs* berjalan normal |
| 4 | Pelacakan status laporan oleh masyarakat | √ | – | Status dan *timeline* penanganan tampil *realtime* tanpa memerlukan autentikasi |
| 5 | Manajemen akun pengguna oleh admin | √ | – | Operasi tambah, ubah, dan hapus akun berhasil |
| 6 | Ekspor data laporan ke format PDF | √ | – | Berkas PDF terunduh dengan format yang benar dan dapat dibaca |
| 7 | Ekspor data laporan ke format EXCEL | √ | – | Berkas EXCEL terunduh dengan format yang benar dan dapat dibaca |
| 8 | Pengujian integrasi *end-to-end* alur pelaporan | √ | – | Dua belas langkah alur penanganan darurat berjalan berkesinambungan tanpa hambatan teknis |
| 9 | Pengujian performa dengan 50 pengguna serentak | √ | – | *Response time* 474ms (rata-rata k6 *endpoint* CRUD, rata-rata keseluruhan 41 *endpoint integration testing* adalah 1.032ms karena *endpoint* panic 3–4 detik), *load time* ~2 detik, latensi FCM dalam rentang 1 - 3 detik |
| 10 | Ketahanan terhadap serangan *SQL Injection* | √ | – | Injeksi berhasil dicegah oleh *Query Builder* dengan *parameterized query CodeIgniter* 4 |
| 11 | Ketahanan terhadap serangan XSS | – | √ | *Payload* <script> tersimpan dan dikembalikan mentah, dicatat sebagai perbaikan |

***b.*** UAT (*User Acceptance Test*)

UAT dilakukan untuk mengetahui tingkat penerimaan pengguna terhadap sistem yang telah dikembangkan. Pengujian dilakukan secara langsung oleh pengguna akhir yang terdiri dari masyarakat, petugas, dan admin Dinas Pemadam Kebakaran Kabupaten Kediri. Aspek yang diuji meliputi kemudahan penggunaan sistem, kesesuaian fungsi, kecepatan respon, dan efektivitas fitur dalam mendukung proses pelaporan darurat secara *real-time*.

Tabel 4. 4 Skenario *User Acceptance Testing*

| No | Aktor | Skenario Pengujian | Hasil yang Diharapkan |
|---|---|---|---|
| 1 | Masyarakat | Mengakses halaman utama sistem | Halaman utama tampil dengan baik |
| 2 | Masyarakat | Mengirim laporan menggunakan *panic button* | Sistem berhasil membuat laporan darurat secara otomatis |
| 3 | Masyarakat | Mengirim laporan dengan formulir lengkap | Laporan berhasil tersimpan ke sistem |
| 4 | Masyarakat | Mengunggah foto kejadian | Foto berhasil diupload |
| 5 | Masyarakat | Sistem mendeteksi lokasi otomatis | Koordinat dan alamat tampil otomatis |
| 6 | Masyarakat | Melacak status laporan | *Timeline Status* laporan tampil |
| 7 | Petugas | Login ke *dashboard* | Sistem mengarahkan ke *dashboard* petugas |
| 8 | Petugas | Menerima notifikasi laporan baru | Notifikasi FCM muncul realtime |
| 9 | Petugas | Membuka rute Google Maps | Navigasi menuju lokasi berhasil |
| 10 | Petugas | Mengubah status laporan | Status laporan berhasil diperbarui |
| 11 | Admin | Mengelola data pengguna | Tambah/edit/hapus pengguna berhasil |
| 12 | Admin | Melihat statistik *dashboard* | Statistik tampil sesuai data |
| 13 | Admin | Mengekspor laporan PDF | File PDF berhasil diunduh |
| 14 | Admin | Mengekspor laporan Excel | File Excel berhasil diunduh |

Seluruh 14 skenario pengujian UAT yang melibatkan tiga peran pengguna (masyarakat, petugas, dan admin) berhasil dijalankan dengan hasil sesuai yang diharapkan. Skenario 1–6 yang diuji oleh responden dari kalangan masyarakat (R1 dan R2) membuktikan bahwa fitur pelaporan publik tanpa login, deteksi geolokasi otomatis, pengunggahan foto, dan pelacakan laporan mandiri berfungsi dengan baik. Skenario 7–10 yang diuji oleh petugas lapangan (R3 dan R4) membuktikan bahwa alur login, penerimaan notifikasi FCM secara *real-time*, akses navigasi Google Maps, serta pembaruan status laporan berjalan lancar. Skenario 11–14 yang diuji oleh admin (R5) membuktikan bahwa fitur manajemen pengguna (CRUD), visualisasi statistik *dashboard*, serta ekspor data ke format PDF dan Excel dapat dioperasikan tanpa hambatan. Secara keseluruhan, tidak ditemukan skenario yang gagal dalam pelaksanaan UAT, sehingga sistem SIRTADA dinyatakan diterima oleh pengguna akhir dari ketiga peran.

| ***Skor*** | ***Keterangan*** |
|---|---|
| *5* | *Sangat Setuju (SS)* |
| *4* | *Setuju (S)* |

Tabel 4. 5 Bobot Skor Skala Linier

| | |
|---|---|
| 3 | Cukup (C) |
| 2 | Tidak Setuju (TS) |
| 1 | Sangat Tidak Setuju (STS) |

Tabel 4.5 menampilkan skala penilaian Likert lima tingkat yang digunakan sebagai instrumen pengukuran UAT. Setiap pernyataan dalam kuesioner dinilai oleh responden menggunakan skala 1 hingga 5, di mana skor 5 (Sangat Setuju) menunjukkan tingkat kepuasan atau penerimaan tertinggi dan skor 1 (Sangat Tidak Setuju) menunjukkan tingkat ketidakpuasan tertinggi (Koo & Yang, 2025).

Tabel 4. 6 Hasil *User Acceptance Testing*

| No | Pernyataan | R1 | R2 | R3 | R4 | R5 |
|---|---|---|---|---|---|---|
| 1 | Tampilan sistem SIRTADA mudah dipahami | 5 | 5 | 5 | 4 | 5 |
| 2 | Sistem mudah digunakan oleh pengguna | 4 | 5 | 4 | 5 | 4 |
| 3 | Proses pengiriman laporan darurat berjalan dengan baik | 5 | 5 | 5 | 5 | 5 |
| 4 | Fitur *panic button* membantu pelaporan darurat dengan cepat | 4 | 3 | 4 | 4 | 4 |
| 5 | Sistem berhasil mendeteksi lokasi kejadian secara otomatis | 5 | 4 | 5 | 5 | 5 |
| 6 | Informasi lokasi pada peta sesuai dengan lokasi pengguna | 5 | 4 | 5 | 4 | 5 |
| 7 | Proses upload foto kejadian berjalan dengan baik | 5 | 4 | 5 | 5 | 5 |
| 8 | *Dashboard* laporan mudah dipahami dan informatif | 4 | 3 | 4 | 4 | 5 |
| 9 | Notifikasi laporan diterima secara *real-time* | 5 | 4 | 5 | 5 | 5 |
| 10 | Fitur pembaruan status laporan berjalan dengan baik | 5 | 3 | 5 | 4 | 4 |
| 11 | Sistem membantu mempercepat proses pelaporan insiden | 5 | 4 | 5 | 5 | 5 |
| 12 | Sistem membantu koordinasi antara masyarakat dan petugas | 4 | 3 | 4 | 4 | 5 |
| 13 | Sistem berjalan dengan responsif tanpa kendala berarti | 4 | 4 | 4 | 4 | 4 |
| 14 | Fitur navigasi menuju lokasi kejadian membantu petugas | 5 | 3 | 5 | 5 | 4 |
| 15 | Secara keseluruhan sistem SIRTADA sudah sesuai kebutuhan pengguna | 4 | 4 | 5 | 5 | 5 |
| Total | | 69 | 58 | 70 | 68 | 70 |

Tabel 4.4 menyajikan skor penilaian yang diberikan oleh kelima responden (R1–R5) terhadap 15 pernyataan pengujian *User Acceptance Testing* (UAT). Selaras dengan metode *purposive sampling* yang dirancang pada bab sebelumnya, kelima responden ini dipilih secara selektif guna merepresentasikan tiga aktor utama sistem secara proporsional, yaitu: elemen Masyarakat (R1 dan R2), Petugas Lapangan (R3 dan R4), serta Admin/Operator (R5). Berdasarkan rekapitulasi data,

responden R1 memberikan total skor 69, R2 memberikan skor 58, R3 memberikan skor 70, R4 memberikan skor 68, dan R5 memberikan skor 70.

Perbedaan skor yang cukup signifikan pada R2 (Masyarakat) mencerminkan penilaian yang lebih kritis, terutama pada aspek fungsionalitas krusial meliputi fitur *panic button* (skor 3), keterbacaan *dashboard* (skor 3), pembaruan status (skor 3), koordinasi antar-pengguna (skor 3), dan akurasi navigasi lokasi (skor 3). Munculnya deviasi skor kritis dari R2 ini secara ilmiah justru memvalidasi ketepatan penggunaan ukuran sampel pengujian. Hal ini sejalan dengan teori *usability* bahwa pengujian dengan 5 responden yang representatif sudah sangat optimal karena terbukti mampu mengidentifikasi hingga 85% permasalahan utama pada interaksi dan fungsionalitas sistem (A. & Kurniawan, 2023). Sikap kritis responden R2 berhasil menyaring celah pembaruan (*feedback* makro) yang sangat berharga bagi penyempurnaan aplikasi tanggap darurat ini. Secara keseluruhan, akumulasi skor dari seluruh responden mencapai 335 dari skor maksimal 375, yang menunjukkan persentase kelayakan yang tinggi sekaligus mengonfirmasi bahwa *Emergency System* berbasis web ini memiliki tingkat penerimaan (*acceptance rate*) yang sangat baik untuk diimplementasikan.

Tabel 4. 7 Interpretasi Nilai

| Persentase | Kategori |
|---|---|
| 81% – 100% | Sangat Baik |
| 61% – 80% | Baik |
| 41% – 60% | Cukup |
| 21% – 40% | Kurang |
| 0% – 20% | Sangat Kurang |

Tabel 4.5 menampilkan kategori interpretasi nilai persentase UAT yang digunakan untuk menentukan kelayakan sistem. Rentang 81%–100% dikategorikan "Sangat Baik" dan menjadi acuan kelulusan kelayakan sistem (Aliyah Aliyah et al., 2024). Kategori ini berfungsi sebagai patokan apakah sistem sudah dapat diterima dan diterapkan secara operasional. Mengacu pada tabel ini, persentase UAT sistem SIRTADA sebesar 89,33% termasuk ke dalam kategori "Sangat Baik", yang berarti sistem layak digunakan. Rumus perhitungan persentase penerimaan diuraikan sebagai berikut:

*c.* Rumus perhitungan persentase *User Acceptance Testing*

$$Persentase = \frac{Total\ Skor}{Skor\ Maksimal} \times 100\%$$

1) Persentase: Merupakan nilai persentase akhir yang merepresentasikan tingkat kelayakan atau penerimaan pengguna terhadap sistem yang diuji. Angka inilah yang nantinya dicocokkan dengan tabel kategori interpretasi nilai (misalnya: Sangat Baik, Baik, Cukup, dst).
2) Total Skor: Merupakan akumulasi atau jumlah keseluruhan nilai riil yang diberikan oleh seluruh responden pada seluruh pertanyaan kuesioner.
3) Skor Maksimal: Merupakan nilai tertinggi secara teoretis yang bisa dicapai apabila seluruh responden memberikan nilai paling maksimal pada semua pertanyaan. Nilai ini didapatkan dari perkalian antara: Jumlah Responden X Jumlah Pertanyaan X Bobot Skor Tertinggi (misalnya, skor 5 untuk jawaban "Sangat Setuju").

Tabel 4. 8 Rekapitulasi Hasil *User Acceptance Testing*

| No | Responden | Nama | *Role* | Total Skor | Skor Maksimal | Persentase % | Kategori |
|---|---|---|---|---|---|---|---|
| 1 | R1 | Ardhi | Masyarakat | 69 | 75 | 92 | Sangat Baik |
| 2 | R2 | Teguh | Masyarakat | 58 | 75 | 77,33 | Baik |
| 3 | R3 | Hartono | Petugas | 70 | 75 | 89,33 | Sangat Baik |
| 4 | R4 | Zola | Petugas | 68 | 75 | 90,67 | Sangat Baik |
| 5 | R5 | Riko | Admin | 70 | 75 | 93,33 | Sangat Baik |
| Total | | | | 335 | 375 | 89,33 | Sangat Baik |

Tabel 4.6 merangkum hasil akhir UAT secara keseluruhan per responden. Dari lima responden yang terlibat, empat di antaranya (R1, R3, R4, dan R5) memperoleh kategori “Sangat Baik” dengan persentase masing-masing 92%, 89,33%, 90,67%, dan 93,33%. Satu responden (R2) memperoleh kategori “Baik” dengan persentase 77,33%, yang mengindikasikan masih terdapat ruang perbaikan pada aspek yang mendapatkan penilaian lebih rendah. Rata-rata persentase seluruh responden adalah 89,33% dan masuk kategori “Sangat Baik”, sehingga sistem SIRTADA dinyatakan layak dan diterima oleh pengguna akhir dari ketiga peran yang diuji.

7. Dokumentasi fitur-fitur yang berhasil diimplementasikan

Seluruh fitur yang dirancang pada tahap desain berhasil diimplementasikan dan berfungsi sesuai dengan spesifikasi kebutuhan. Berikut adalah dokumentasi lengkap fitur-fitur yang berhasil dibangun:

a. Fitur untuk Masyarakat (Publik):
   1) Tombol darurat (*panic button*) dengan deteksi geolokasi otomatis dan *reverse geocoding* berbasis *openstreetmap Nominatim*
   2) Formulir laporan lengkap dengan validasi sisi klien menggunakan Zod dan dukungan *speech-to-text* via *Web Speech API*
   3) Fitur pelacakan laporan mandiri menggunakan kode unik RPT-YYYY-XXXX dengan tampilan timeline kronologis
   4) Serta halaman statistik publik yang menampilkan data agregat laporan secara transparan.

b. Fitur untuk Admin:
   1) *Dashboard* monitoring dengan lima *metric cards*,
   2) Visualisasi line chart tren tujuh hari, dan bar chart distribusi kategori insiden
   3) Halaman laporan masuk dengan filter multi-kriteria dan panel detail lengkap beserta kontrol pembaruan status bertahap
   4) Halaman riwayat insiden dengan fitur pencarian, filter rentang tanggal, dan ekspor data ke format PDF maupun Excel
   5) Halaman manajemen pengguna dengan operasi CRUD lengkap melalui antarmuka modal dialog

c. Fitur untuk Petugas:
   1) *Dashboard* petugas berorientasi tugas dengan indikator prioritas berbasis kategori dan tingkat keparahan
   2) Halaman laporan aktif dengan kontrol pembaruan status dan penambahan catatan lapangan
   3) Halaman riwayat penanganan
   4) Serta fitur buat laporan yang memungkinkan petugas mendokumentasikan insiden yang ditemukan langsung di lapangan.

d. Fitur *Real-time* dan Keamanan:

1) Distribusi *Push notification* berbasis *Firebase Cloud Messaging* dengan latensi dalam rentang 1-3 detik
2) Mekanisme polling sebagai fallback *real-time*
3) Autentikasi JWT dengan penyimpanan token pada *httponly cookie* berklausul *samesite=Strict*
4) Kontrol akses berbasis peran (RBAC) tiga tingkat
5) Proteksi SQL *Injection* melalui *Query Builder CodeIgniter 4*
6) Proteksi *CSRF* diimplementasikan melalui atribut *SameSite=Strict* namun belum diuji secara eksplisit dalam siklus pengujian ini, validasinya direncanakan pada pengembangan lanjutan.

## B. Pembahasan

1. Evaluasi hasil pengujian

Berdasarkan hasil pengujian yang telah dilaksanakan, sistem SIRTADA terbukti mampu memenuhi seluruh kebutuhan fungsional dan non-fungsional yang ditetapkan pada tahap analisis. Berikut adalah hasil evaluasi utama dari pengujian:

a. Sistem berhasil melewati seluruh 10 skenario *blackbox testing* dengan tingkat kelulusan 100%. Pada pengujian unit, seluruh 63 test case lulus (100%) menggunakan PHPUnit 10.5 untuk *Backend* dan Vitest 4.1 untuk *Frontend*. Pada pengujian integrasi, 78 dari 80 *assertion* dinyatakan lulus (97,5%). Dari 2 *assertion* yang tidak lulus, satu merupakan false failure pada kategori *Security Integration*, skrip pengujian mengasumsikan celah RBAC yang ternyata tidak terbukti ada, sehingga sistem justru terbukti memblokir akses tidak sah secara benar, satu lainnya merupakan temuan kerentanan nyata pada kategori *Security Integration*, *payload* XSS pada *field* nama_pelapor tersimpan tanpa *escaping*, yang dicatat sebagai prioritas perbaikan pada pengembangan selanjutnya.

b. Proses pelaporan kejadian darurat yang sebelumnya dilakukan melalui *WhatsApp* dengan keterbatasan informasi lokasi dan pendokumentasian manual menggunakan *Microsoft Excel*, kini dapat dilaksanakan secara digital dengan deteksi geolokasi otomatis dan konfirmasi kode laporan yang dapat dilacak secara mandiri oleh masyarakat.

c. Mekanisme distribusi informasi kepada petugas yang sebelumnya bergantung pada komunikasi manual via *WhatsApp*, kini diotomasi melalui *Push notification* berbasis *Firebase Cloud Messaging* dengan latensi dalam rentang 1-3 detik, jauh di bawah batas toleransi 8 detik yang telah ditetapkan pada spesifikasi kebutuhan non-fungsional.

d. *Response time* API tercatat rata-rata 474ms dengan nilai maksimum 1.386ms, keduanya di bawah ambang batas toleransi 15 detik yang ditetapkan. Nilai 474ms merupakan rata-rata pengujian performa *endpoint* CRUD, rata-rata keseluruhan 41 *endpoint integration testing* adalah 1.032ms karena *endpoint* POST /api/reports/panic memiliki rata-rata 3–4 detik akibat distribusi notifikasi FCM dan *WhatsApp Gateway* secara sinkron. *Page load time* tercatat ~2 detik, dan latensi FCM dalam rentang 1–3 detik, di bawah toleransi 8 detik.

e. Dari sisi keamanan, sistem berhasil mencegah serangan SQL Injection melalui penggunaan *Query Builder* dengan *parameterized query* pada *CodeIgniter* 4, dibuktikan dengan pengiriman *payload* injeksi ' , DROP TABLE reports , -- yang ditolak tanpa memengaruhi integritas data. Namun, pengujian integrasi keamanan mengidentifikasi bahwa *field* teks seperti nama_pelapor belum menerapkan mekanisme *escaping* XSS *payload* <script>alert('xss')</script> tersimpan dan dikembalikan mentah melalui *endpoint* pelacakan laporan. Temuan ini dicatat sebagai keterbatasan sistem dan menjadi prioritas perbaikan pada pengembangan selanjutnya melalui penambahan fungsi *htmlspecialchars*() pada seluruh output teks di *Backend*.

f. Pengujian integrasi *end-to-end* yang melibatkan dua belas langkah alur penanganan laporan darurat secara berkesinambungan membuktikan bahwa seluruh komponen sistem *Frontend*, *Backend*, basis data, dan FCM berinteraksi tanpa hambatan teknis. Hasil ini dikonfirmasi oleh *integration testing* berbasis *Postman Newman CLI* yang mengeksekusi 41 *HTTP request* dengan 80 *assertion*, menghasilkan 78 *assertion* lulus dan durasi total eksekusi 2 menit 27,3 detik.

2. Kelebihan dan kekurangan sistem

Berdasarkan hasil implementasi dan pengujian, berikut adalah kelebihan dan kekurangan sistem SIRTADA yang dapat diidentifikasi:

a. Kelebihan Sistem

1) Sistem menyediakan mekanisme pelaporan *panic button* yang dapat diakses tanpa registrasi sehingga meminimalkan hambatan bagi masyarakat yang berada dalam kondisi darurat.
2) Integrasi *Firebase Cloud Messaging* memastikan informasi laporan baru tersampaikan kepada admin dan petugas dalam hitungan detik.
3) Integrasi *Web Speech API* memungkinkan pengisian deskripsi laporan menggunakan input suara, sangat membantu pelapor yang berada dalam kondisi darurat dan kesulitan mengetik di perangkat *smartphone*.
4) Fitur navigasi langsung ke *Google Maps* dari koordinat laporan memungkinkan petugas menentukan rute tercepat menuju lokasi kejadian secara otomatis tanpa perlu mencari alamat secara manual.
5) Mekanisme *audit trail* otomatis melalui tabel incident_logs memastikan seluruh riwayat perubahan status penanganan terdokumentasi secara sistematis tanpa bergantung pada kedisiplinan pencatatan manual, menggantikan pendokumentasian *Microsoft Excel* yang sebelumnya digunakan.
6) Penggunaan arsitektur *three-tier* yang memisahkan *Frontend React*.js dan *Backend CodeIgniter* 4 secara independen memberikan fleksibilitas pengembangan dan pemeliharaan yang tinggi, serta membuka peluang pengembangan kanal akses baru di masa mendatang tanpa perlu mengubah logika *Backend* yang sudah ada.
7) Penerapan mekanisme keamanan berlapis JWT dengan *httpOnly cookie*, RBAC tiga tingkat, proteksi *SQL Injection*, dan atribut *SameSite Strict* pada *cookie*, menjadikan sistem memenuhi standar keamanan aplikasi web modern untuk layanan publik.

b. Kekurangan Sistem

1) Penyimpanan foto laporan pada direktori lokal server (writable/uploads/) membatasi skalabilitas penyimpanan.

2) Setiap laporan saat ini hanya dapat menyertakan satu foto bukti kejadian, hal ini disebabkan oleh desain kolom foto pada tabel reports yang bertipe VARCHAR (255) tunggal, bukan relasi *one-to-many* ke tabel terpisah. Dalam kondisi insiden nyata, satu foto seringkali tidak mencukupi untuk dokumentasi yang komprehensif.
3) Sistem belum mengimplementasikan mekanisme *refresh token*, sehingga ketika JWT kedaluwarsa setelah 24 jam, pengguna akan otomatis keluar dari sistem tanpa pembaruan sesi secara transparan, yang dapat mengganggu aktivitas admin atau petugas yang sedang bertugas dalam waktu panjang.
4) Penggunaan *Library Baileys* sebagai *WhatsApp Gateway* merupakan pendekatan *WebSocket* tidak resmi yang tidak memiliki jaminan *Service Level Agreement (SLA)* dari *WhatsApp.*

3. Kendala selama proses pengembangan

Terdapat tiga kendala teknis yang dihadapi selama pengembangan SIRTADA dan berhasil diselesaikan, antara lain:

a. Konfigurasi CORS lintas origin: Penggunaan httpOnly cookie untuk autentikasi mengharuskan konfigurasi CORS antara *Frontend* (port 5173) dan *Backend* (port 8080) ditetapkan secara eksplisit, karena pengiriman credentials lintas origin tidak dapat menggunakan wildcard * pada *allowedOrigins*. Diselesaikan dengan memperbarui CorsFilter.php di *Backend* dan menambahkan withCredentials: *true* pada konfigurasi Axios di *Frontend*.
b. Pendaftaran *Service Worker* di lingkungan lokal: *Firebase Cloud Messaging* mensyaratkan Service Worker terdaftar pada secure context (HTTPS), sehingga lingkungan pengembangan lokal tanpa HTTPS menyebabkan *Service Worker* gagal terdaftar dan *Push notification* tidak dapat diterima. Diselesaikan dengan mengonfigurasi *browser* pengembangan agar memperlakukan localhost sebagai *trusted origin* yang dikecualikan dari persyaratan HTTPS.
c. *Race condition* pada pembaruan data *real-time*: Mekanisme dual-channel (FCM dan polling fallback) yang berjalan bersamaan pada

RealtimeReportsContext berpotensi memicu pembaruan data ganda di antarmuka ketika kedua kanal merespons dalam waktu berdekatan. Diselesaikan dengan mengimplementasikan mekanisme *debounce* pada fungsi pembaruan *state* sehingga respons redundan dapat dicegah dan antarmuka hanya memperbarui data satu kali per kejadian.

4. Perbandingan hasil aktual dengan tujuan awal pada tahap analisis

Berikut adalah perbandingan antara tujuan yang ditetapkan pada tahap analisis kebutuhan dengan hasil aktual yang dicapai setelah implementasi dan pengujian sistem:

Tabel 4. 9 Perbandingan hasil aktual

| No | Tujuan Awal | Hasil Aktual |
|---|---|---|
| 1 | Membangun sistem pelaporan darurat yang dapat diakses masyarakat tanpa registrasi | Tercapai, Fitur *panic button* dapat diakses publik tanpa login |
| 2 | Menyediakan deteksi geolokasi otomatis pada formulir pelaporan | Tercapai, *Geolocation API* mendeteksi koordinat dan *reverse geocoding* mengonversi ke alamat secara otomatis |
| 3 | Menyediakan mekanisme notifikasi *real-time* kepada admin dan petugas saat laporan masuk | Tercapai, FCM mendistribusikan *Push notification* dengan latensi dalam rentang 1-3 detik |
| 4 | Mengimplementasikan sistem pelacakan status laporan yang dapat dipantau masyarakat | Tercapai, Fitur lacak laporan menampilkan *Timeline Status* secara kronologis |
| 5 | Membangun *dashboard* monitoring terpusat untuk admin dan petugas | Tercapai, *Dashboard* menampilkan statistik, grafik dan data laporan *real-time* |
| 6 | Menerapkan sistem autentikasi dan otorisasi berbasis peran (RBAC) | Tercapai, JWT dengan *httpOnly cookie* dan tiga tingkat peran berfungsi dengan baik |
| 7 | Memenuhi standar keamanan terhadap *SQL Injection* | Tercapai serangan SQL Injection berhasil dicegah oleh *Query Builder CodeIgniter* 4 melalui *parameterized query* |
| 8 | Memenuhi standar keamanan terhadap XSS | Belum Tercapai, *payload* XSS pada *field* nama_pelapor tersimpan tanpa *escaping* , dicatat sebagai perbaikan |
| 9 | Mencapai *response time* API di bawah 15 detik | Tercapai, *response time* rata-rata 474ms dan nilai tertinggi 1.386ms, keduanya jauh di bawah batas yang ditetapkan. Nilai 474ms adalah rata-rata k6 *endpoint* CRUD, rata-rata 41 *endpoint integration testing* adalah 1.032ms karena *endpoint* panic membutuhkan 3–4 detik untuk distribusi notifikasi FCM dan *WhatsApp Gateway* |
| 10 | Mencapai *page load time* di bawah 3 detik | Tercapai, *Load time* aktual 2 detik (33% di bawah batas) |
| 11 | Mendukung ekspor data laporan ke format PDF dan Excel | Tercapai, Fitur ekspor menggunakan jsPDF dan SheetJS berfungsi normal |
| 12 | Mendukung hingga 100 pengguna mengakses sistem secara bersamaan | Sebagian Tercapai, Diuji dengan 50 *virtual users* menggunakan *Grafana k6*, seluruh request berhasil (0 failed, |

| | | |
|---|---|---|
| | | *response time* rata-rata 474ms pada pengujian performa *endpoint* CRUD). Pengujian hingga kapasitas 100 pengguna serentak memerlukan pengujian lanjutan. |

Sepuluh dari dua belas tujuan awal yang ditetapkan pada tahap analisis berhasil dicapai sepenuhnya pada tahap implementasi. Satu tujuan terkait kapasitas pengguna serentak sebagian tercapai, dan satu tujuan terkait keamanan XSS belum tercapai dan dicatat sebagai prioritas perbaikan. Kondisi ini menunjukkan bahwa proses analisis kebutuhan dan perencanaan yang dilakukan pada tahap awal metode *Waterfall* cukup akurat dan realistis terhadap kapabilitas teknologi yang dipilih.

5. Efektivitas metode *Waterfall*

Metode *Waterfall* yang diterapkan dalam pengembangan sistem SIRTADA terbukti efektif untuk konteks proyek ini. Berikut adalah efektivitas metode yang dapat diidentifikasi berdasarkan hasil pengembangan:

a. Kejelasan dan kelengkapan dokumentasi kebutuhan yang dihasilkan pada fase analisis meliputi:

9 1) Tabel kebutuhan fungsional,

2) Tabel kebutuhan non-fungsional,

3) Spesifikasi sistem,

11 4) *Data flow diagram*,

5) *Use Case Diagram*, dan

6) *Entity Relational Database*

Hal ini memberikan acuan yang solid bagi seluruh tahap pengembangan berikutnya, sehingga meminimalkan perubahan rancangan di tengah proses implementasi.

b. Pemisahan yang jelas antara fase desain dan fase implementasi memungkinkan seluruh keputusan arsitektur ditetapkan sebelum penulisan pengkodean dimulai, pemilihan *three-tier architecture* kontrak rest api dan skema basis data yang terbukti mengurangi terjadinya *refactoring* besar-besaran selama implementasi berlangsung.

c. Tingkat kelulusan unit *testing* sebesar 100% (63 dari 63 test case) dan *integration testing* sebesar 97,5% (78 dari 80 *assertion*), serta tercapainya sepuluh dari dua belas tujuan awal secara penuh, menjadi bukti empiris

bahwa metode *Waterfall* yang diterapkan secara disiplin dengan dokumentasi yang komprehensif mampu menghasilkan sistem perangkat lunak yang sebagian besar memenuhi spesifikasi kebutuhan yang telah ditetapkan sejak awal. Dua tujuan yang belum terpenuhi sepenuhnya, yaitu proteksi XSS dan pengujian kapasitas hingga 100 pengguna serentak dicatat sebagai keterbatasan yang akan ditangani pada pengembangan selanjutnya.

d. Dokumentasi teknis yang dihasilkan pada setiap fase *Waterfall* mulai dari dokumen analisis, desain, hingga hasil pengujian yang berfungsi sebagai panduan pemeliharaan yang dapat digunakan oleh administrator sistem di Dinas Pemadam Kebakaran Kabupaten Kediri untuk pengelolaan sistem jangka panjang.
e. Metode *Waterfall* menunjukkan keterbatasannya pada satu aspek yaitu penetapan mekanisme keamanan autentikasi yang memerlukan penyesuaian konfigurasi *CORS* setelah fase implementasi berjalan merupakan perubahan yang idealnya sudah ditetapkan pada fase desain. Hal ini menunjukkan bahwa untuk aspek keamanan sistem yang bersifat kritis, pendekatan *Waterfall* perlu dilengkapi dengan tinjauan keamanan (*security review*) yang lebih ketat pada fase desain agar keputusan implementasi keamanan tidak berubah di tengah proses.

Secara keseluruhan, metode *Waterfall* dinilai sesuai dan efektif untuk proyek pengembangan sistem informasi dengan ruang lingkup yang terdefinisi dengan jelas, kebutuhan yang relatif stabil, dan target pengguna yang spesifik seperti pada pengembangan sistem SIRTADA untuk Dinas Pemadam Kebakaran Kabupaten Kediri .

# BAB V
# PENUTUP

## A. Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan pada *Emergency System* berbasis web SIRTADA untuk Dinas Pemadam Kebakaran Kabupaten Kediri, dapat ditarik kesimpulan sebagai berikut:

1. Sistem telah berhasil dikembangkan melalui tahap-tahap metode *Waterfall* secara berurutan dan terdokumentasi. Pada tahap analisis, dilakukan wawancara dengan koordinator lapangan Damkar untuk mengidentifikasi kebutuhan 10 fitur fungsional dan 9 parameter non-fungsional. Tahap desain menghasilkan arsitektur three-tier (*React*.js *Frontend*, *CodeIgniter* 4 *Backend*, MySQL database), DFD level 0 dan 1, *Use Case Diagram*, ERD, serta 12 *Wireframe* antarmuka. Tahap implementasi berhasil membangun 6 controller REST API, 5 model, 2 filter autentikasi (JWT & RBAC), 2 layanan notifikasi (FCM dan *WhatsApp Gateway*), serta 10 halaman *Frontend* dengan 63 *unit test* yang seluruhnya lulus menggunakan PHPUnit 10.5 (*Backend*) dan Vitest 4.1 (*Frontend*). Tahap pengujian meliputi *blackbox testing*, integrasi *end-to-end*, dan UAT yang menghasilkan tingkat kelulusan 100% pada pengujian akhir.
2. Hasil pengujian menunjukkan sistem berjalan sesuai kebutuhan fungsional dan non-fungsional yang telah ditetapkan. Secara fungsional, seluruh fitur utama berhasil diimplementasikan yaitu, *panic button* dengan geolokasi otomatis, pelacakan laporan via kode unik RPT-YYYY-XXXX, *dashboard* pemantauan *real-time* berbasis peran (RBAC), pembaruan status dengan audit trail, serta distribusi notifikasi melalui *Firebase* Cloud Messaging (FCM) dan *WhatsApp Gateway Baileys*. Secara non-fungsional, *response time* API tercatat rata-rata 474ms, jauh di bawah batas 15 detik, nilai ini merupakan rata-rata pengujian performa *endpoint* CRUD, sedangkan rata-rata keseluruhan 41 *endpoint integration testing* adalah 1.032ms karena *endpoint* POST /api/reports/panic memiliki rata-rata 3–4 detik akibat distribusi notifikasi FCM dan *WhatsApp Gateway* secara sinkron. Latensi notifikasi FCM dalam rentang 1–3 detik (di bawah batas 8 detik), *page load time* sekitar

2 detik (di bawah batas 3 detik), serta sistem terbukti tahan terhadap serangan *SQL Injection.* Pengujian integrasi keamanan mengidentifikasi kerentanan XSS pada field nama_pelapor yang belum di*escape,* dan dicatat sebagai perbaikan. Hasil *User Acceptance Testing* (UAT) terhadap 5 responden (masyarakat, petugas, admin) menghasilkan persentase rata-rata 89,3% dengan kategori Sangat Baik.

3. Metode *Waterfall* terbukti efektif untuk proyek pengembangan *Emergency System* dengan ruang lingkup yang terdefinisi jelas dan kebutuhan yang relatif stabil. Keberhasilan metode ini ditunjukkan oleh beberapa indikator yaitu:
   a. Dokumentasi yang komprehensif pada setiap fase (analisis, desain, implementasi, pengujian) menjadi acuan yang solid dan meminimalkan *refactoring* besar-besaran.
   b. Tingkat kelulusan unit *testing* mencapai 100% (63 dari 63 test case), dan *integration testing* 97,5% (78 dari 80 *assertion*). Dari sisi pencapaian tujuan penelitian, 10 dari 12 tujuan awal berhasil dipenuhi sepenuhnya, satu tujuan sebagian tercapai (kapasitas 100 pengguna) dan satu belum tercapai pada (proteksi XSS).
   c. Pemisahan yang jelas antara fase desain dan implementasi memungkinkan keputusan arsitektur (rest api, skema basis data) ditetapkan sebelum pengkodean dimulai. Namun, metode ini juga menunjukkan keterbatasan pada aspek keamanan autentikasi yang memerlukan penyesuaian konfigurasi CORS di tengah implementasi, yang idealnya sudah ditetapkan pada fase desain. Hal ini mengindikasikan perlunya *security review* yang lebih ketat pada fase desain untuk proyek serupa di masa mendatang.

## B. Saran

Berdasarkan limitasi dan temuan selama penelitian, berikut adalah saran untuk pengembangan lebih lanjut serta penerapan sistem di masa depan:

1. Pengembangan sistem lebih lanjut
   a. Fitur *multiple* foto per laporan

Pengembangan skema basis data dari kolom foto tunggal menjadi tabel report_photos dengan relasi *one-to-many*, sehingga petugas dapat mendokumentasikan insiden dari berbagai sudut secara lebih komprehensif.

b. Mekanisme *refresh token*

Implementasi pola access token (masa berlaku pendek) dan *refresh token* (masa berlaku lebih panjang) untuk menggantikan JWT tunggal 24 jam, sehingga admin dan petugas tidak perlu login ulang saat sesi berakhir di tengah tugas operasional.

c. Konversi ke *Progressive Web App* (PWA)

Penambahan manifest.json dan konfigurasi service worker agar sistem dapat diinstal di perangkat seluler sebagai aplikasi mandiri, serta mendukung notifikasi push saat perangkat offline.

d. Peta heatmap interaktif

Penambahan visualisasi sebaran insiden berbasis heatmap menggunakan Leaflet.js dengan plugin seperti leaflet-heatmap untuk membantu Dinas Damkar dalam analisis pola kejadian darurat di wilayah Kabupaten Kediri.

*e.* Integrasi n8n sebagai *workflow automation*

n8n berperan sebagai *orchestrator* yang menghubungkan dua kanal pelaporan yakni dari web (*React*.js) yang sudah ada, dan dari *WhatsApp* API Bot. Masyarakat cukup mengirim pesan *WhatsApp* berformat sederhana (misal: #darurat Nama: Andi, Lokasi: Jl. Sudirman, Deskripsi: Kebakaran) ke nomor bot. n8n mendengarkan pesan via *webhook*, mengekstrak informasi menggunakan *regex*, mengonversi alamat ke koordinat via *reverse geocoding* OpenStreetMap, lalu mengirim data terstruktur ke *endpoint* API SIRTADA (POST /api/reports/*WhatsApp*). *Backend* kemudian memproses laporan seperti biasa (membangkitkan kode unik, notifikasi FCM, *WhatsApp Gateway*) dan n8n membalas kode laporan ke pengirim *WhatsApp*. Keuntungannya adalah masyarakat dapat melapor tanpa akses web, data terstruktur, tidak mengubah arsitektur inti, dan workflow mudah diperluas.

f. Migrasi dari Leaflet.js ke Google Maps API

Saat ini sistem menggunakan Leaflet.js dengan OpenStreetMap yang open source tanpa API key. Untuk meningkatkan akurasi navigasi dan pengalaman pengguna, disarankan mengganti atau menambahkan Google Maps API termasuk *Directions API* dan *Distance Matrix API*. Keunggulannya meliputi data lalu lintas *real-time*, estimasi waktu tempuh yang lebih akurat, *street view*, serta integrasi lebih mudah dengan fitur autocomplete alamat. Namun perlu diperhatikan aspek biaya karena Google Maps API berbayar setelah melebihi kuota gratis.

g. Peralihan dari *Baileys* ke *WhatsApp* Business API resmi Meta

*WhatsApp Gateway* saat ini menggunakan *Library Baileys* yaitu pendekatan WebSocket tidak resmi dan tidak memiliki jaminan *service level agreement* (SLA). Untuk penggunaan produksi di instansi pemerintah, direkomendasikan menggunakan *WhatsApp* Business API resmi dari Meta. Keuntungannya mencakup keandalan tinggi, dukungan resmi, kepatuhan terhadap kebijakan privasi, serta kemampuan mengirim notifikasi dalam skala besar. Tantangannya adalah proses verifikasi bisnis yang panjang dan biaya berbasis volume pesan. Implementasi dapat dilakukan dengan mendaftarkan Dinas Pemadam Kebakaran Kabupaten Kediri sebagai *business verified* dan mengganti *microservice Baileys* dengan Cloud API Meta.

2. Perluasan objek penelitian

Penelitian ini hanya mencakup Dinas Pemadam Kebakaran Kabupaten Kediri sebagai objek tunggal. Pengembangan selanjutnya disarankan untuk memperluas cakupan ke instansi kedaruratan lain seperti Badan Penanggulangan Bencana Daerah (BPBD), Dinas Kesehatan, atau Unit Gawat Darurat (UGD) rumah sakit daerah. Perluasan ini akan memungkinkan sistem SIRTADA berkembang menjadi *platform* tanggap darurat terpadu lintas instansi, sehingga koordinasi penanganan insiden dapat dilakukan secara terintegrasi.

3. Penggunaan metode tambahan

Penelitian ini menggunakan metode pengembangan *Waterfall* yang terbukti efektif untuk proyek dengan kebutuhan stabil. Untuk penelitian selanjutnya, disarankan mengombinasikan pendekatan pengujian yang lebih komprehensif seperti pengujian beban penggunaan hingga kapasitas 100 pengguna serentak menggunakan *Grafana k6*, serta pengujian keamanan *penetration testing* yang lebih menyeluruh. Selain itu, dapat dipertimbangkan penerapan metode Agile atau Scrum untuk modul-modul yang memerlukan iterasi cepat berdasarkan umpan balik pengguna, khususnya pada antarmuka masyarakat. Penggunaan metode evaluasi tambahan seperti *System Usability Scale* (SUS) atau *Technology Acceptance Model* (TAM) juga disarankan untuk mengukur tingkat penerimaan dan kemudahan penggunaan sistem secara lebih terstandarisasi dan dapat dibandingkan dengan sistem serupa.

4. Penerapan nyata sistem atau model ke organisasi terkait

   Sistem SIRTADA yang telah berhasil dikembangkan dan diuji secara akademis perlu ditindaklanjuti dengan penerapan nyata di Dinas Pemadam Kebakaran Kabupaten Kediri sebagai langkah validasi operasional. Penerapan nyata ini mencakup serangkaian tahapan, yaitu:

   a. Sosialisasi dan pelatihan kepada seluruh petugas dan admin terkait penggunaan sistem
   b. Penetapan kebijakan resmi instansi mengenai kewajiban penggunaan sirtada sebagai media pelaporan darurat. Kolaborasi dengan dinas komunikasi dan informatika kabupaten kediri juga disarankan untuk mendukung penyediaan infrastruktur server yang andal dan terkelola secara institusional.