BAB III
PELAKSANAAN KERJA PROFESI

3.1 Bidang Kerja

Selama masa magang, praktikan ditempatkan di Divisi Kesejahteraan Sosial di bawah bimbingan Kepala Seksi Kesejahteraan Sosial, Ibu Nurul Hayati, serta Ibu Tini sebagai Penanggung Jawab (PIC) untuk Komplek Kasuari. Praktikan bertanggung jawab sebagai *Frontend Developer* dalam pengembangan aplikasi keamanan lingkungan berbasis *mobile* yang dinamakan ***Kasuari Secure***. Aplikasi ini dirancang untuk meningkatkan keamanan di lingkungan Perumahan Kasuari, dengan fokus pada penggunaannya oleh warga dan petugas keamanan.

Pada tahap implementasi, praktikan menggunakan bahasa pemrograman *TypeScript* dan *Angular* dengan *Ionic Framework* untuk pengembangan sisi *frontend* aplikasi. Penerapan teknologi ini bertujuan untuk menciptakan tampilan antarmuka yang interaktif dan *user-friendly*, sekaligus memastikan aplikasi dapat berjalan dengan lancar di perangkat mobile. Dalam prosesnya, praktikan mengerjakan berbagai aspek pengembangan tampilan, seperti merancang tampilan *dashboard* keamanan, halaman laporan kejadian, dan notifikasi *real-time*.

Setelah menyelesaikan setiap tahap implementasi, praktikan secara rutin melaporkan perkembangan hasil kerja kepada tim terkait. Komunikasi dilakukan terutama dengan *Project Manager*, PIC dari Komplek Kasuari, serta pembimbing kerja profesi untuk memastikan bahwa aplikasi yang dikembangkan sudah sesuai dengan kebutuhan dan spesifikasi yang telah disepakati. Proses pelaporan ini juga digunakan untuk mendapatkan *feedback* dan melakukan perbaikan atau pengembangan lebih lanjut, sehingga kualitas aplikasi terus meningkat seiring berjalannya proyek.

3.2 Pelaksanaan Kerja

Praktikan melaksanakan program Kerja Profesi (KP) di Kelurahan Pondok Pucung selama periode 1 April 2024 hingga 3 Juli 2024, dengan total durasi tiga bulan. Jam kerja praktikan disesuaikan dengan jam operasional kelurahan, yaitu dari pukul 08.00 hingga 16.00 WIB. Selama menjalankan KP, praktikan berperan aktif dalam berbagai tugas dan tanggung jawab yang berkaitan dengan pengembangan aplikasi keamanan lingkungan ***Kasuari Secure***. Tahapan kerja praktikan dimulai dengan melakukan analisis desain dan *mockup* yang telah

disiapkan oleh tim desain. Dari hasil analisis tersebut, praktikan kemudian menerjemahkan desain menjadi tampilan antarmuka aplikasi *mobile* yang fungsional.

Selain itu, praktikan juga bertanggung jawab dalam proses implementasi, memastikan bahwa setiap elemen desain terintegrasi dengan baik dalam aplikasi. Setelah tahap implementasi, praktikan secara berkesinambungan melakukan evaluasi terhadap hasil pengembangan, baik melalui uji coba internal maupun dengan masukan dari *stakeholder*. Evaluasi ini penting untuk memastikan bahwa aplikasi berjalan sesuai dengan spesifikasi dan kebutuhan pengguna.

3.2.1. Pengenalan Lingkungan Kerja dan Perencanaan

Pada awal pelaksanaan Kerja Profesi (KP), praktikan diperkenalkan dengan lingkungan kerja di Kelurahan Pondok Pucung, khususnya dalam Divisi Kesejahteraan Sosial, tempat di mana praktikan ditempatkan. Dalam divisi ini, praktikan berkolaborasi dengan tim yang terdiri dari beberapa anggota, termasuk Kepala Seksi Kesejahteraan Sosial dan Penanggung Jawab (PIC) dari Komplek Kasuari, serta *stakeholders* lainnya yang terlibat dalam pengembangan aplikasi keamanan lingkungan ***Kasuari Secure***.

Lingkungan kerja di kelurahan ini memiliki suasana yang kolaboratif dan berbasis pelayanan publik. Praktikan juga mendapat kesempatan untuk belajar langsung dari para pembimbing dan tim, baik dari aspek teknis maupun manajerial. Salah satu poin penting dari pengenalan ini adalah memahami bagaimana kelurahan berfungsi sebagai Lembaga pemerintahan yang berperan dalam menjaga kesejahteraan sosial masyarakat, termasuk memastikan keamanan lingkungan di sekitar perumahan.

Setelah mengenal lingkungan kerja, tahapan berikutnya adalah melakukan perencanaan pengembangan aplikasi. Perencanaan ini mencakup diskusi dengan tim mengenai fitur-fitur yang akan diimplementasikan dalam aplikasi, menyusun *timeline* proyek, dan membagi tugas serta tanggung jawab masing-masing anggota tim. Praktikan juga ikut serta dalam pertemuan dengan pihak-pihak terkait untuk mengumpulkan *requirements* dan memastikan semua pihak memiliki pemahaman yang sama terkait tujuan dan manfaat dari aplikasi yang dikembangkan. Pada tahap ini, *mockup* dan desain awal aplikasi dianalisis oleh praktikan untuk mendapatkan gambaran lebih jelas mengenai alur kerja dan tampilan yang akan diterapkan dalam pengembangan.

3.2.2. Penyusunan Dokumen Kebutuhan Sistem

Setelah pembagian tugas, pihak *developer* melakukan wawancara dan diskusi dengan pihak komplek Kasuari yang terlibat dalam pengelolaan proyek Aplikasi Keamanan Lingkungan ***KASUARI SECURE*** berbasis mobile di Kelurahan Pondok Pucung. Data yang diperoleh dari wawancara dan diskusi ini dianalisis untuk menghasilkan informasi tentang kebutuhan pengguna dan sistem, serta merumuskan alur penggunaan aplikasi sesuai dengan proses bisnis yang ada. Hasil analisis kemudian dituangkan dalam bentuk tabel yang merinci kebutuhan fungsional dan non-fungsional, yang mencakup fitur, kinerja, dan batasan yang diperlukan. Dengan langkah ini, tim pengembang diharapkan dapat menciptakan solusi yang efektif dan efisien, meningkatkan keamanan lingkungan di kawasan tersebut.

Tabel 3.1 Analisis Kebutuhan

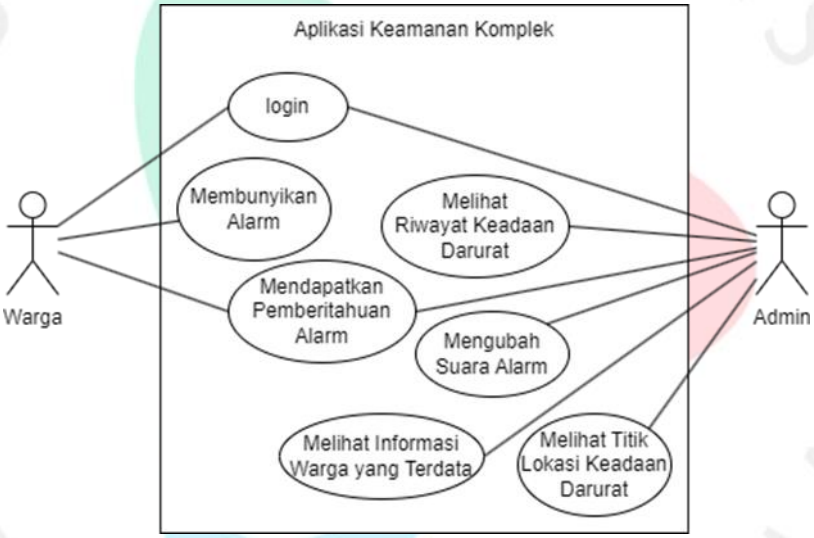
FUNGSIONAL
ANALISA KEBUTUHAN
Saya ingin sistem dapat :
User dapat login sekali untuk selamanya/waktu yang sangat lama
User dapat membunyikan alarm darurat dari aplikasi
User mendapatkan pemberitahuan setiap keadaan darurat melalui notifikasi aplikasi
Admin dapat melihat *history* keadaan darurat
Admin dapat mengubah suara alarm keadaan darurat
Admin dapat melihat informasi warga yang terdata di aplikasi
Admin dapat melihat titik lokasi keadaan darurat
NON FUNGSIONAL
ANALISA KEBUTUHAN
Saya ingin sistem dapat :
User mengakses aplikasi menggunakan perangkat mobile (Android).
User mengakses aplikasi secara Online
Desain Antarmuka harus mudah dipahami
Terdapat Map pada aplikasi untuk menampilkan titik keadaan darurat
Terproteksi dengan baik

3.2.3. Desain Sistem dan UI/UX

Pada tahap desain, Praktikan merancang sistem dengan memanfaatkan *Unified Modelling Language* (UML). Dalam tahapan perancangan dan pengembangan aplikasi *Frontend*, Praktikan menggunakan UML untuk membuat *Use Case Diagram* dan *Activity Diagram*.

A. *Use Case Diagram*

Use Case Diagram merupakan pemodelan perilaku dari sistem yang akan dikembangkan. Diagram ini digunakan untuk menggambarkan interaksi antara aktor dan sistem, yang divisualisasikan melalui aktor dan *use case* (Fitriani, 2018).



Gambar 3.3 *Use Case Diagram* Aplikasi *kasuari Secure*

B. *Use Case Description*

Tabel 3.2 *Use Case Description Login*

Use Case Name	***Login***
Actors	*User*
Secondary Actor	Aplikasi *Android Mobile*
Trigger	Pengguna mengakses aplikasi

Preconditions	• Akun pengguna terdaftar di sistem. • Pengguna memiliki perangkat dengan aplikasi terinstal.	
Postcondition	• Pengguna berhasil masuk ke aplikasi dan dapat mengakses fungsinya. • Sistem mencatat perangkat dan sesi login pengguna.	
Succes Scenario	**Actor**	**System**
	1. *User* membuka aplikasi. 2. *User* memasukkan *username* dan *password* yang benar. 3. Aplikasi *Android Mobile* mengirimkan kredensial *user* ke *server*. 6. Aplikasi *Android Mobile* menampilkan menu utama aplikasi kepada *user*.	3. Sistem memvalidasi kredensial pengguna terhadap *Database*. 4. Sistem mengirimkan respon sukses ke Aplikasi *Android Mobile*.
Alternatives flows	• Akun pengguna tidak ditemukan. Sistem menampilkan pesan kesalahan dan meminta pengguna untuk melakukan pendaftaran akun ke pihak keamanan komplek. • Kata sandi salah. Sistem menampilkan pesan kesalahan. • Koneksi tidak tersedia. Sistem akan menampilkan pesan pemberitahuan	

Tabel 3.3 Membunyikan Alarm

Use Case Name	**Membunyikan Alarm**
Actors	Warga
Secondary Actor	Aplikasi Android Mobile
Trigger	Warga menekan alarm keadaan darurat

Preconditions	• Pengguna telah masuk ke aplikasi. • Konektivitas internet atau jaringan seluler tersedia.	
Postcondition	• Alarm keadaan darurat diaktifkan dan notifikasi dikirim ke Pusat Keamanan. • Aplikasi menampilkan konfirmasi kepada pengguna bahwa alarm telah dibunyikan.	
Succes Scenario	**Actor**	**System**
	1. Warga memicu alarm keadaan darurat melalui tombol yang tersedia di aplikasi. 2. Aplikasi Android Mobile mengirimkan informasi alarm ke server. 5. Aplikasi Android Mobile menampilkan konfirmasi kepada Warga yang menekan alarm bahwa alarm telah dibunyikan. 6. *Admin* dan Warga menerima notifikasi alarm.	3. Sistem menerima informas alarm dan memvalidasi pengguna. 4. Sistem meneruskan informasialarm ke *Admin* dan Warga lain.
Alternatives flows	• Koneksi terputus. Sistem mencoba mengirim notifikasi kembali saat koneksi tersedia. • Tombol alarm tidak berfungsi karena kesalahan aplikasi. Sistem menampilkan pesan kesalahan dan menyarankan pengguna untuk menutup dan membuka kembali aplikasi.	

Tabel 3.4 Use Case Description Mendapatkan Pemberitahuan Alarm

Use Case Name	**Mendapatkan Pemberitahuan Alarm**
Actors	Pengguna (Satpam Komplek, Warga) *Admin*(Satpam Komplek)
Secondary Actor	-

Trigger	Pengguna menerima alarm keadaan darurat	
Preconditions	• *Admin* telah masuk ke aplikasi. • Alarm keadaan darurat diterima dari aplikasi. • Sistem dan Server pada aplikasi beroperasi normal.	
Postcondition	• *Admin* mencatat informasi alarm dan tindakan yang diambil. • Aplikasi mengirimkan konfirmasi ke pengguna bahwa alarm telah diterima dan ditangani.	
Succes Scenario	**Actor**	**Systems**
	1. Pengguna menerima notifikasi alarm dari aplikasi. 2. Pengguna menekan notifikasi. 4. *Admin* Mengonfirmasi notifikasi keadaan darurat 6. *Admin* mengambil tindakan yang sesuai berdasarkan jenisalarm jenisalarm	3. Sistem menampilkan detailalarm, termasuk lokasi pengguna dan jenis alarm.
Alternatives flows	• Aplikasi tidak dapat memproses alarm. Sistem mencoba memproses ulang alarm nanti atau mengirimkan pemberitahuan. • Koneksi tidak tersedia. Sistem tidak akan menampilkan alarm keadaan darurat sampai koneksi tersedia.	

Tabel 3.5 Use Case Melihat Riwayat Keadaan Darurat

Use Case Name	**Melihat Riwayat Keadaan Darurat**
Actors	*Admin* (Satpam Komplek)
Secondary Actor	Aplikasi Android Mobile
Trigger	*Admin* mengakses aplikasi pada menu riwayat alarm keadaan darurat
Preconditions	• *Admin* telah masuk ke aplikasi. • Sistem dan Server pada aplikasi beroperasi normal.
Postcondition	• *Admin* dapat melihat informasi detail setiap alarm dalam riwayat.

	Actor	Systems
Succes Scenario	1. *Admin* membuka menu riwayat keadaan darurat pada aplikasi. 2. Aplikasi Android Mobile mengirimkan permintaan riwayat alarm ke server. 3. Aplikasi Android Mobile menampilkan daftar riwayat alarm kepada Satpam, termasuk tanggal, waktu dan jenis alarm.	4. Sistem mengambil data riwayat alarm pengguna daridatabase. 5. Sistem mengirimkan datariwayat alarm ke Aplikasi Android Mobile.
Alternatives flows	• Koneksi terputus. Sistem mencoba memuat ulang data riwayat saat koneksi tersedia. • Data riwayat alarm tidak ditemukan jika tidak ada Riwayat alarm. Sistem menampilkan pesan kepada pengguna.	

Tabel 3.6 Use Case Description Mengubah Suara Alarm Keadaan Darurat

Use Case Name	**Mengubah Suara Alarm Keadaan Darurat**	
Actors	*Admin* (Satpam Komplek)	
Secondary Actor	-	
Trigger	*Admin* ingin mengubah suara alarm untuk situasi keadaan darurat.	
Preconditions	• *Admin* harus dalam kondisi login ke aplikasi. • Aplikasi harus dalam keadaan online.	
Postcondition	• Suara alarm keadaan darurat diperbarui sesuai dengan pilihan *Admin*.	
Succes	**Actor**	**Actor**

Scenario	1. *Admin* mengakses menu pengaturan alarm. 3. *Admin* memilih jenis alarm yang ingin diubah suaranya. 5. *Admin* memilih suara alarm baru. 6. *Admin* menekan tombol simpan untuk mengubah suara alarm. 9. *Admin* menerima notifikasi bahwa suara alarm berhasil diperbarui.	2. Sistem menampilkan opsi untuk mengubah suara alarm. 4. Sistem menampilkan daftar suara alarm yang tersedia. 7. Sistem memperbarui suara alarm yang akan digunakan untuk keadaan darurat dan menyimpan perubahan ke database. 8. Sistem memberikan notifikasibahwa suara alarm berhasil diperbarui.
Alternatives flows	• Koneksi terputus. Sistem menampilkan pesan kesalahan yang meminta *Admin* untuk mencoba lagi setelah koneksi stabil. • *Admin* membatalkan perubahan. Sistem membatalkan operasi perubahan suara alarm dan kembali ke menu pengaturan alarm tanpa ada perubahan yang disimpan. • Terjadi kesalahan saat memproses data riwayat alarm. Sistem mengirimkan pesan pemberitahuan.	

Tabel 3.7 Use Case Description Melihat Informasi Warga yang Terdata

Use Case Name	Melihat Informasi Warga yang Terdata	
Actors	*Admin* (Satpam Komplek)	
Secondary Actor	-	
Trigger	*Admin* ingin melihat informasi warga yang terdata di aplikasi.	
Preconditions	• *Admin* harus dalam kondisi login ke aplikasi.	
Postcondition	• Aplikasi harus dalam keadaan online.	
Succes	**Actor**	**System**

Scenario	1. *Admin* mengakses menu data warga. 2. *Admin* memilih warga dari daftar untuk melihat informasi detail. 3. *Admin* mendapatkan detail informasi warga.	2. Sistem menampilkan daftar warga yang terdata. 4. Sistem menampilkan informasi detail dari warga yang dipilih.
Alternatives flows	• Data warga tidak ditemukan. Sistem menampilkan pesan bahwa data warga tidak ditemukan. • Koneksi internet terputus. Sistem menampilkan pesan kesalahan dan meminta *Admin* untuk mencoba lagi setelah koneksi stabil. • Terjadi kesalahan saat memproses data warga. Sistem mengirimkan pesan pemberitahuan.	

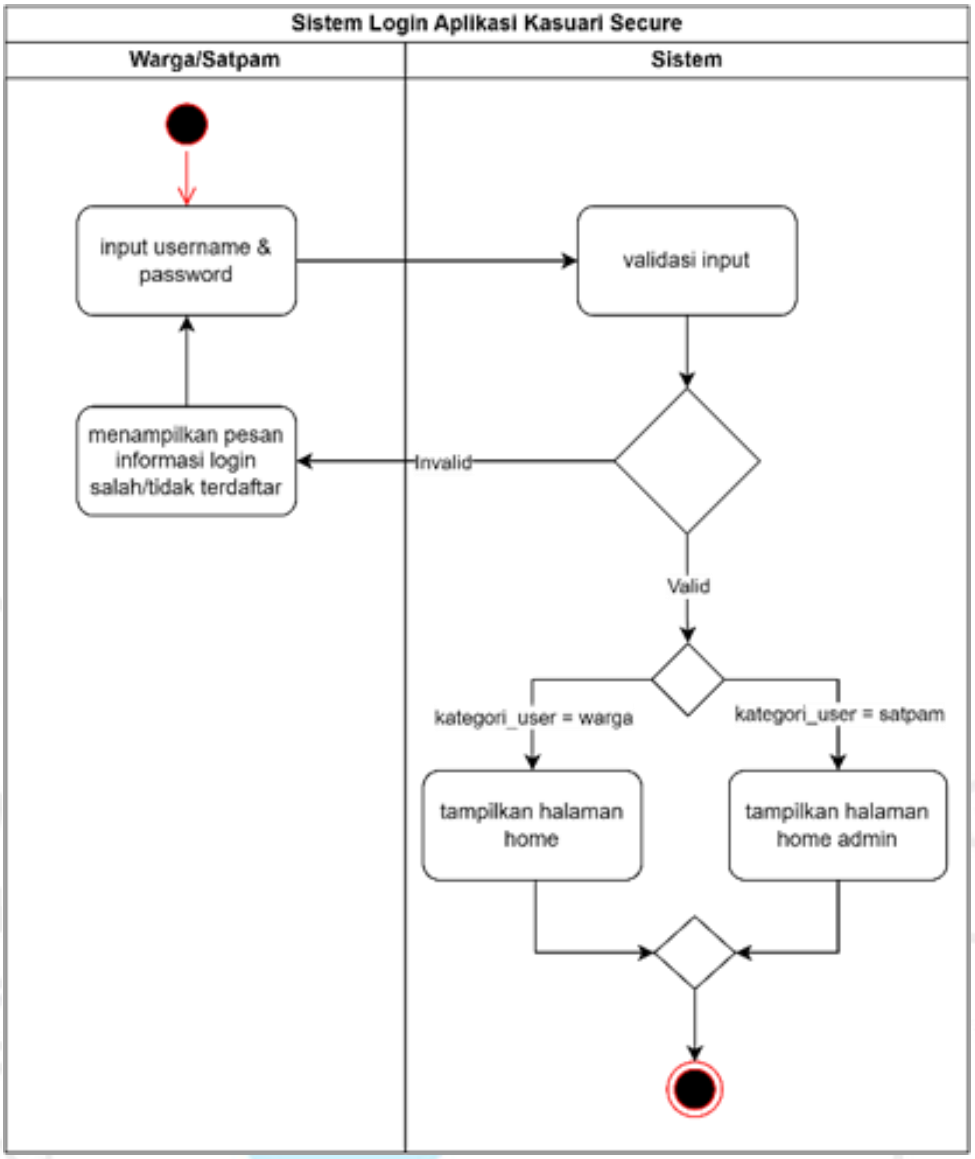
Tabel 3.8 Use Case Description Melihat Titik Lokasi Keadaan Darurat

Use Case Name	Melihat Titik Lokasi Keadaan Darurat	
Actors	*Admin* (Satpam Komplek)	
Secondary Actor	-	
Trigger	*Admin* ingin melihat titik lokasi keadaan darurat yang terdeteksi di aplikasi.	
Preconditions	• *Admin* harus dalam kondisi login ke aplikasi. • Aplikasi harus dalam keadaan online.	
Postcondition	• *Admin* dapat melihat titik lokasi keadaan darurat.	
Succes	**Actor**	**System**

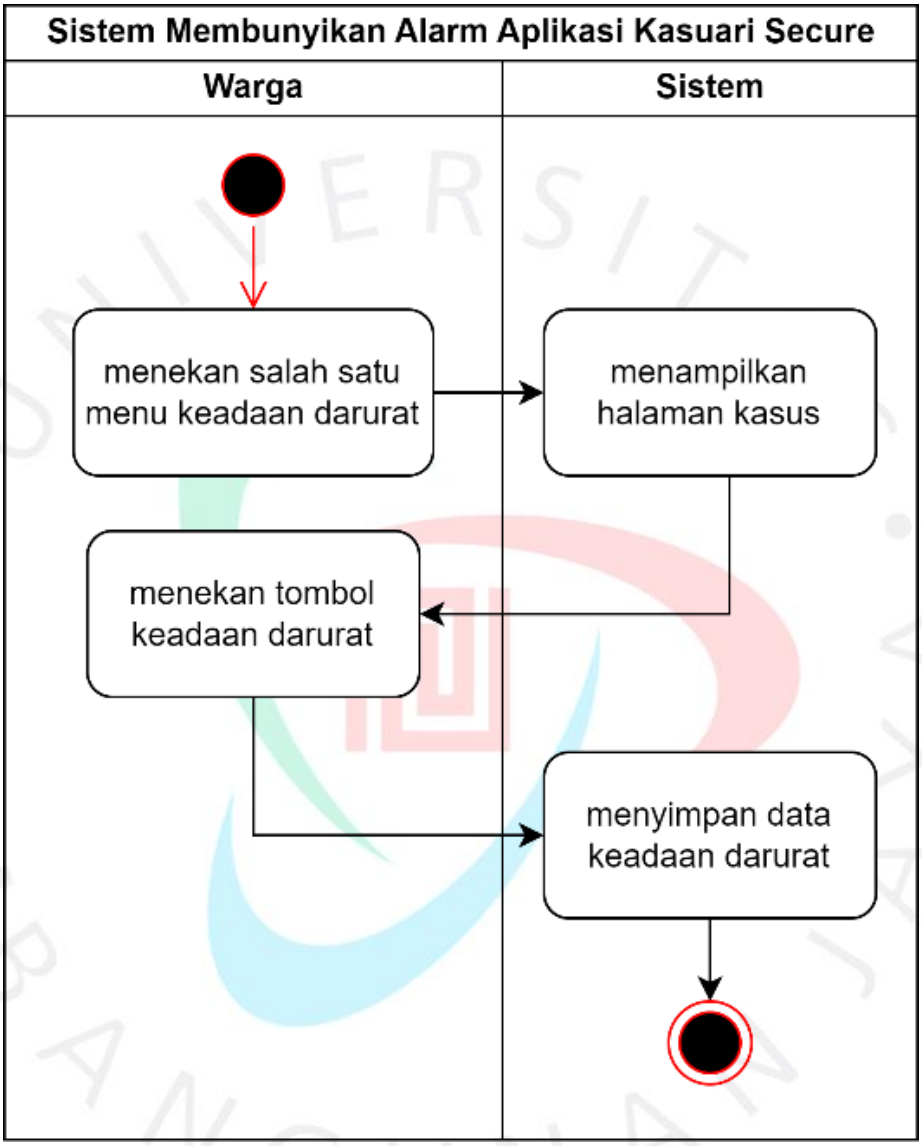
Scenario	1. *Admin* mengakses menu peta. 3. *Admin* memilih titik lokasi untuk melihat detailnya. 5. *Admin* mendapatkan detail informasi keadaan darurat pada titik tersebut.	2. Sistem menampilkan peta dengan titik lokasi keadaan darurat yang terdeteksi. 4. Sistem menampilkan informasi detail mengenai keadaan darurat di lokasi tersebut.
Alternatives flows	• Tidak ada data keadaan darurat. Sistem menampilkan pesan bahwa tidak ada keadaan darurat yang terdeteksi saat ini dan peta tetap kosong atau hanya menampilkan data sebelumnya. • Koneksi internet terputus. Sistem menampilkan pesan kesalahan dan meminta *Admin* untuk mencoba lagi setelah koneksi stabil. • Terjadi kesalahan saat memproses data titik Lokasi keadaan darurat. Sistem mengirimkan pesan pemberitahuan.	

C. *Activity Diagram*

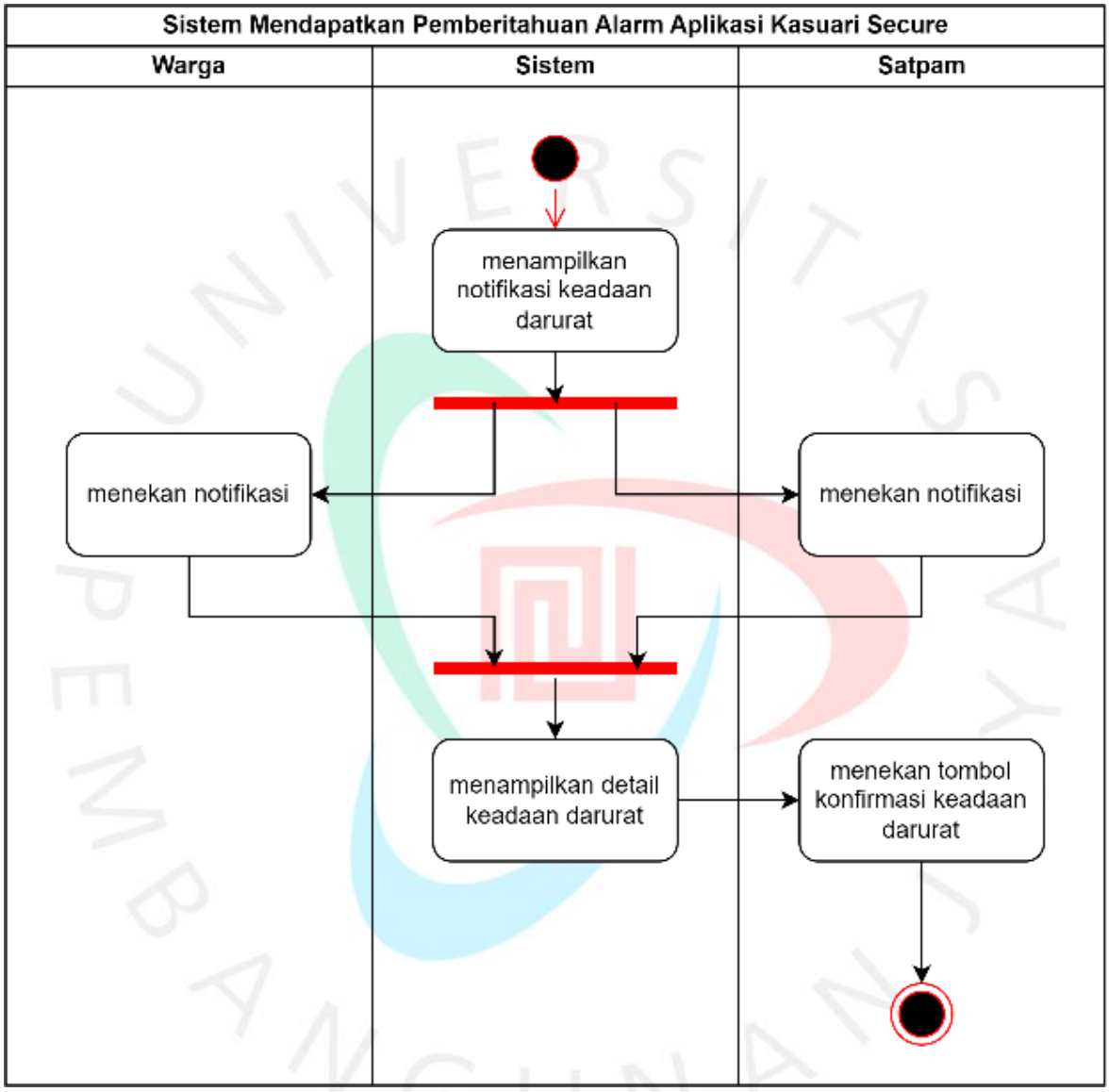
Activity Diagram adalah salah satu jenis diagram UML yang digunakan untuk menggambarkan aliran proses bisnis atau aktivitas yang terjadi dalam suatu sistem. *Activity Diagram* dapat membantu pengembang perangkat lunak untuk memodelkan proses bisnis atau alurkerja sistem dengan lebih sistematis dan terstruktur. (Pranoto, S., Sutiono, S., & Nasution, 2024).



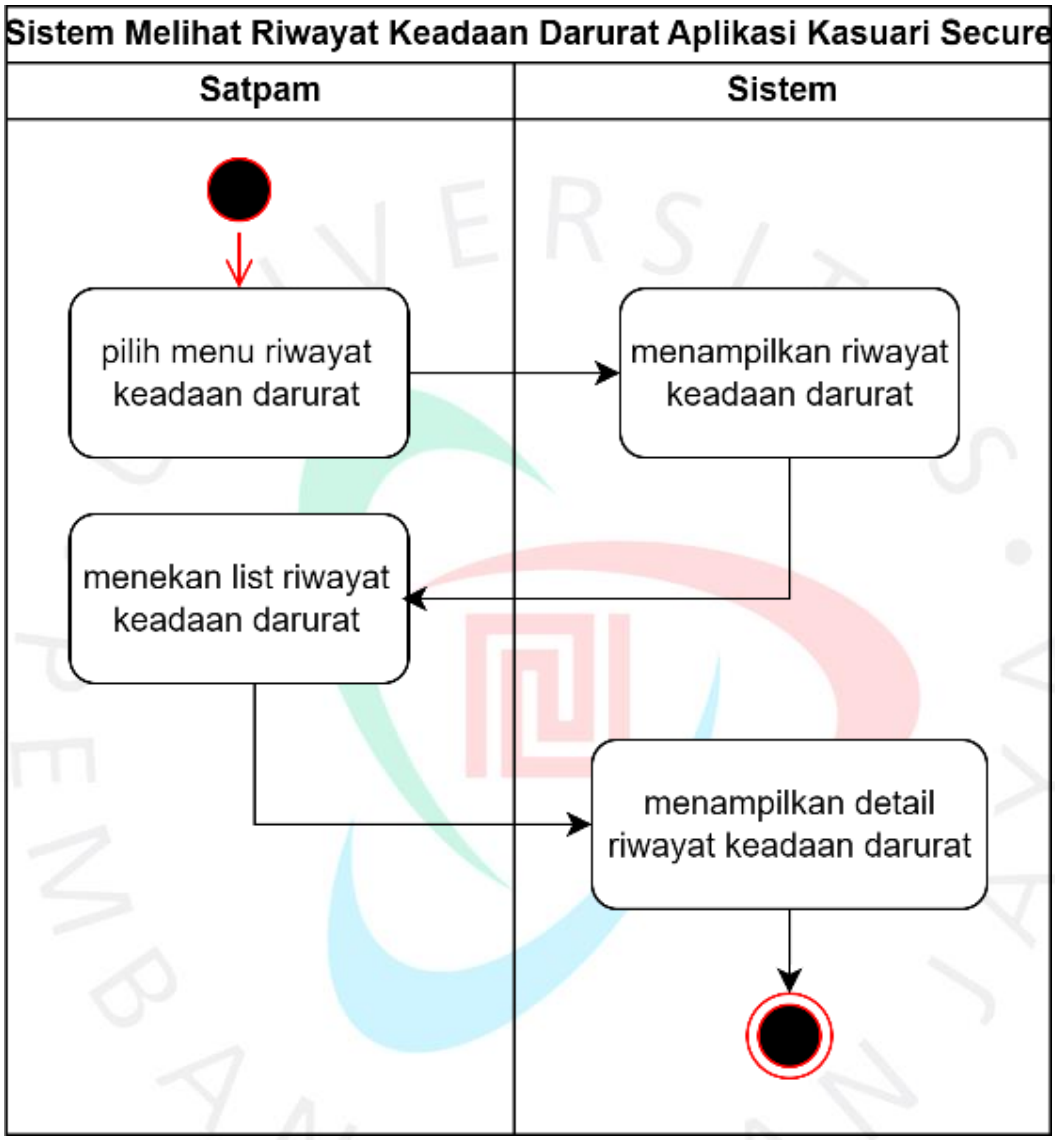
Gambar 3.4 *Activity Diagram Login*



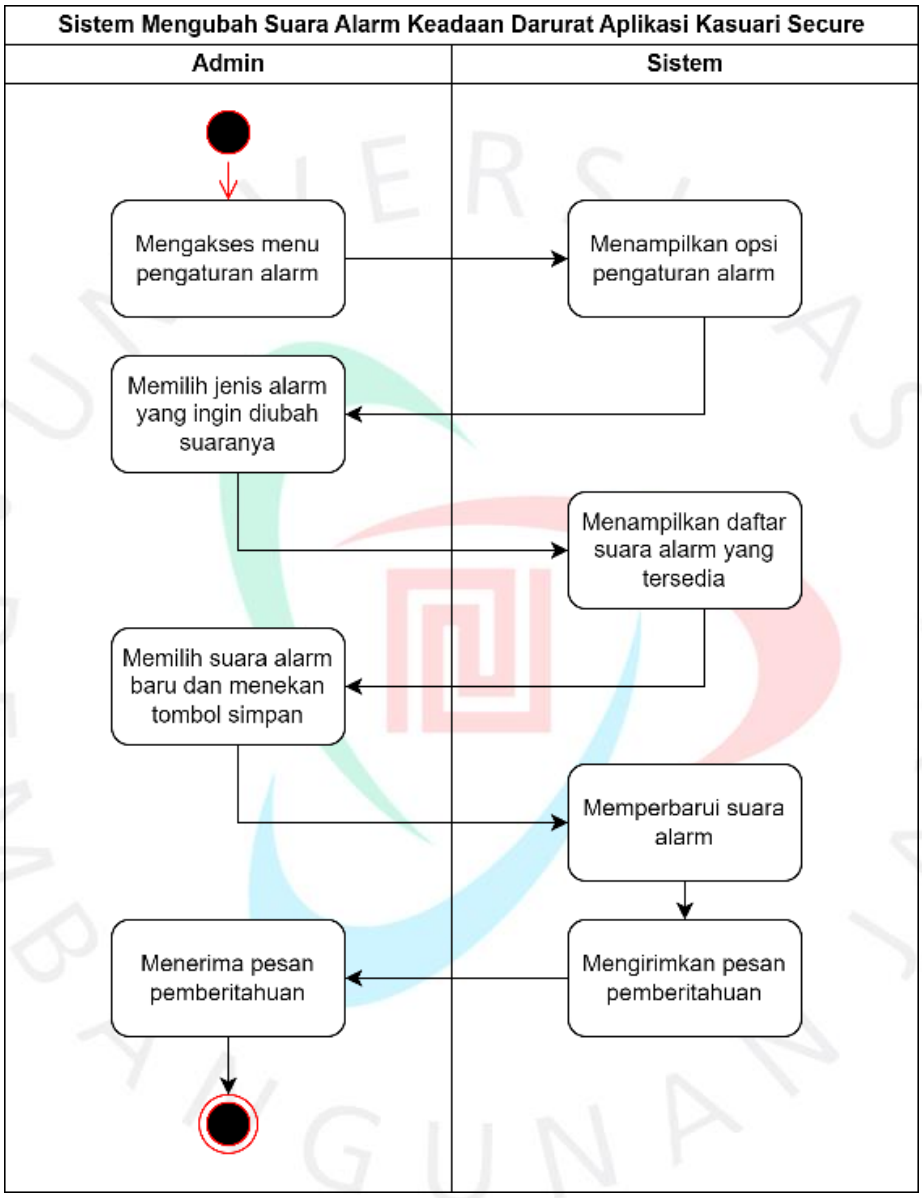
Gambar 3.5 *Activity Diagram* Membunyikan Alarm



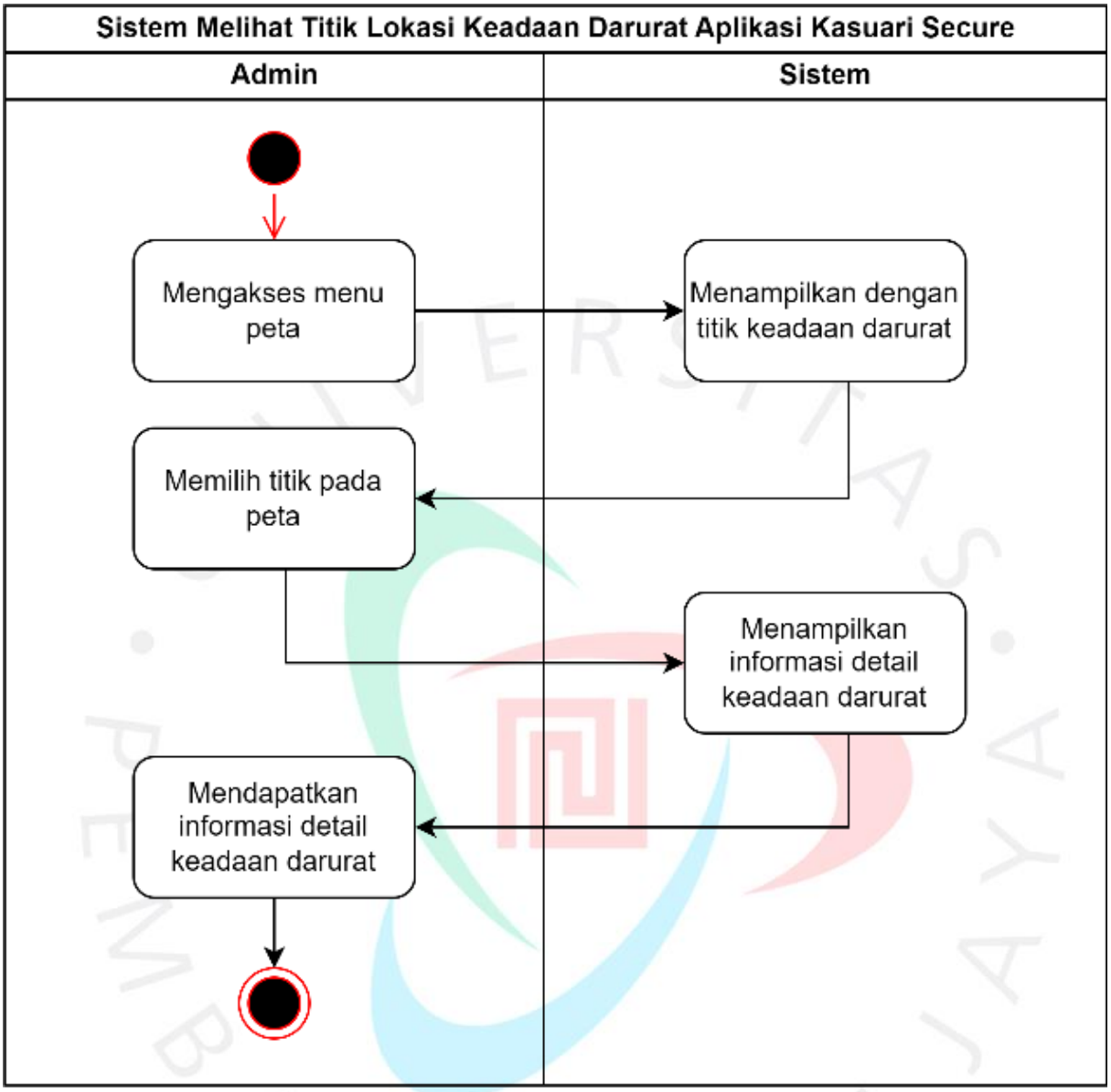
Gambar 3.6 *Activity Diagram* Mendapatkan Pemberitahuan *Alarm*



Gambar 3.7 *Activity Diagram* Melihat Riwayat Keadaan Darurat



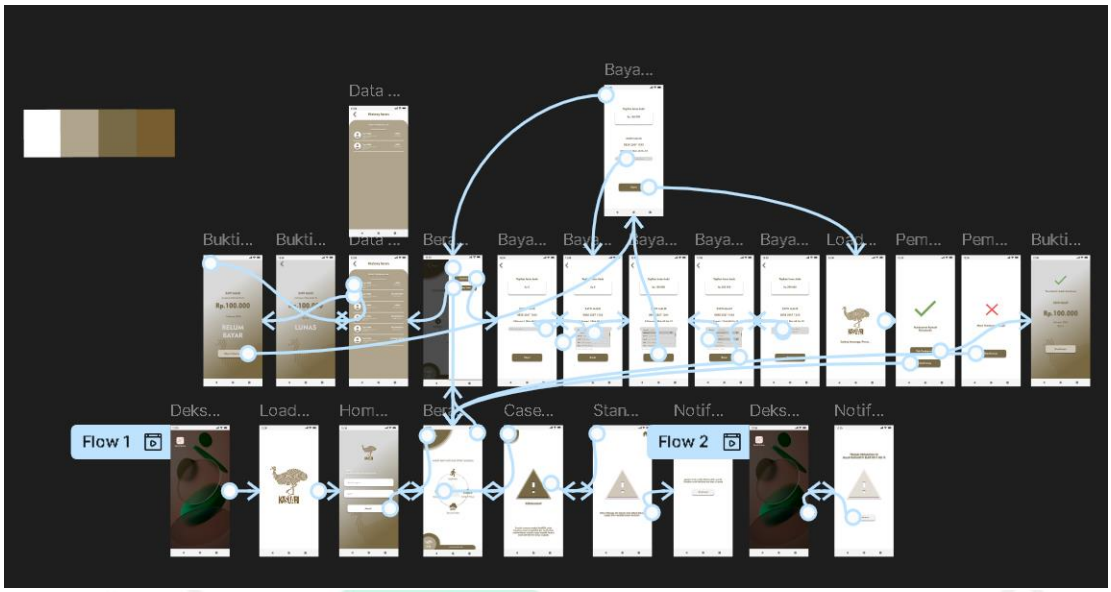
Gambar 3.8 *Activity Diagram* Mengubah Suara *Alarm*



Gambar 3.9 Activity Diagram Melihat Titik Lokasi Keadaan Darurat

C. Prototyping

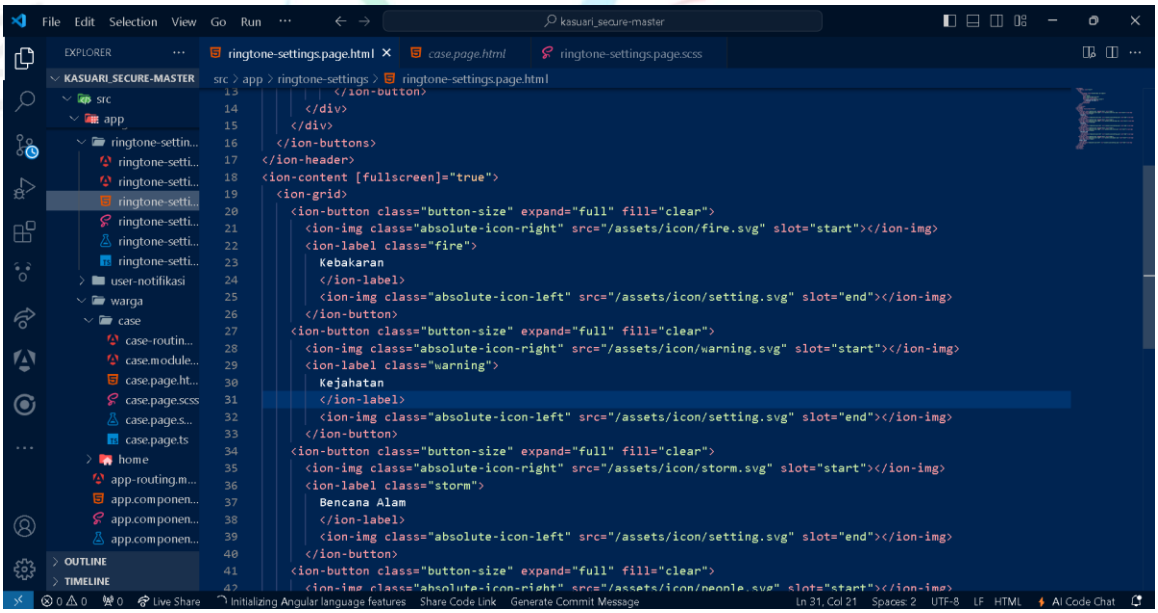
Setelah mempelajari alur proses yang diberikan, praktikan kemudian menganalisis kembali dengan menyusun prototipe. Prototipe aplikasi ***Kasuari Secure*** dapat dilihat pada gambar berikut.



Gambar 3.10 Prototyping Pada Aplikasi *Kasuari Secure*

3.2.4. *Frontend* (Implementasi)

Setelah memahami dan mempelajari desain sistem serta prototipenya, praktikan melanjutkan dengan mengimplementasikan hasil desain tersebut ke dalam bentuk perangkat lunak. Untuk pengembangan, praktikan menggunakan *Visual Studio Code* sebagai aplikasi utama.

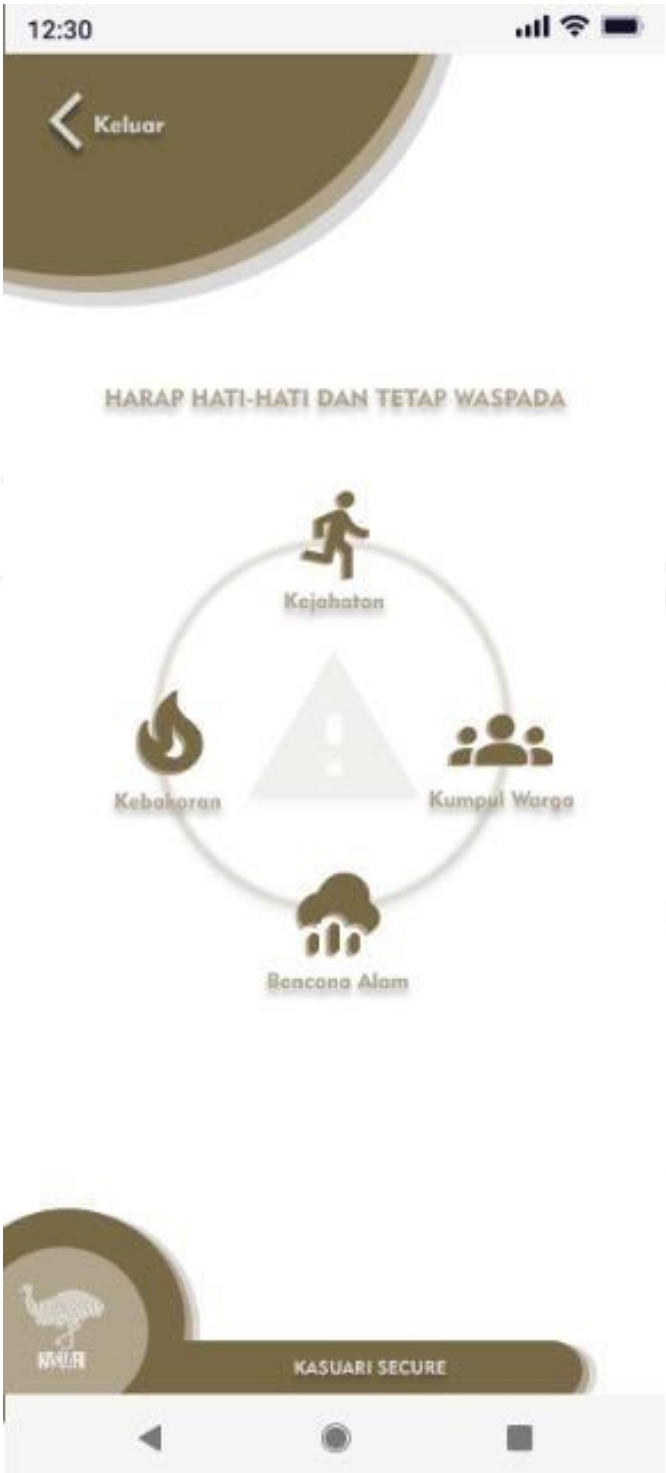


Gambar 3.11 Implementasi Menggunakan Visual Studio Code

Pada sisi *frontend*, praktikan memanfaatkan bahasa pemrograman *TypeScript* yang dikombinasikan dengan *Angular*, serta menggunakan *Ionic Framework* untuk memperkuat antarmuka aplikasi. Kombinasi teknologi ini memungkinkan pengembangan yang lebih efisien dan responsif, terutama dalam membangun aplikasi yang dapat berjalan di berbagai *platform* dengan tampilan yang modern dan interaktif.

3.2.5. Tampilan Pada Aplikasi *(Mockup)*

Gambar 3.12 Implementasi Halaman *Login* (Warga)



Gambar 3.13 Implementasi Halaman *Home* (Warga)

Gambar 3.14 Implementasi Halaman Membunyikan *Alarm* (Warga)

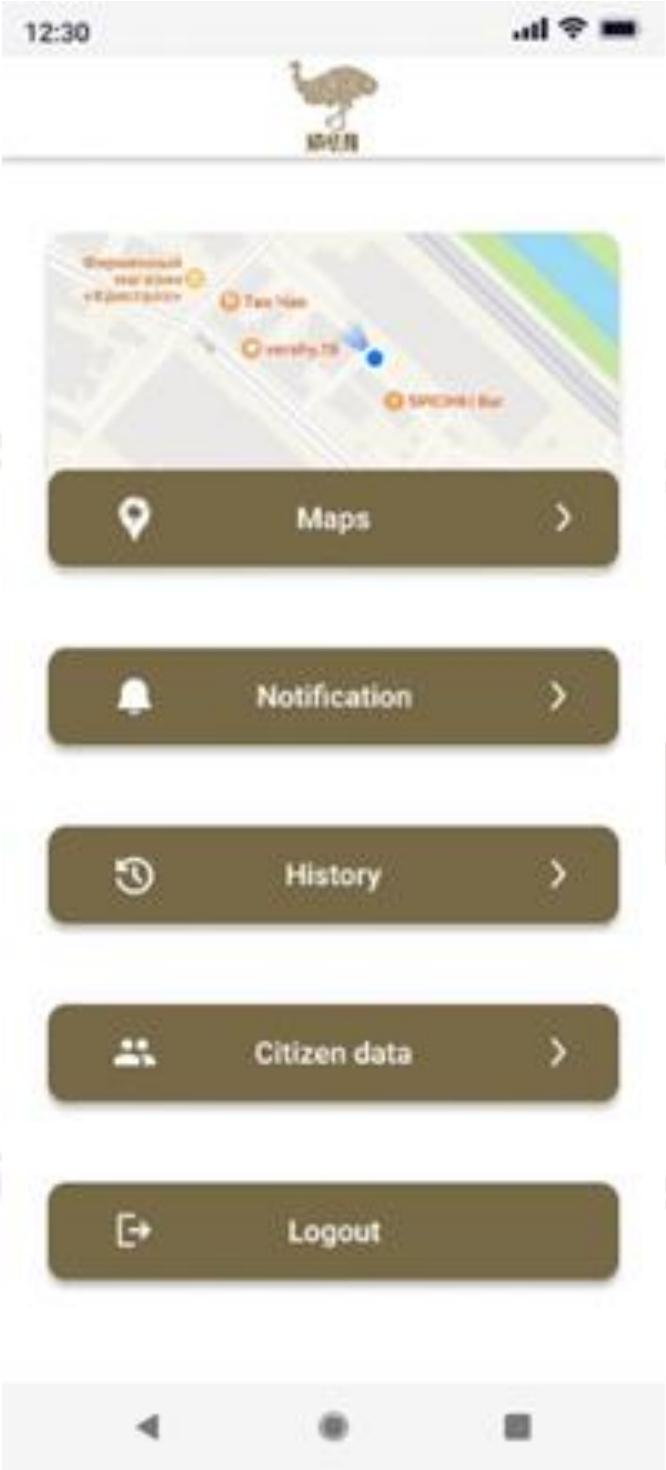
Pada **Gambar 3.12, 3.13, 3.14** ditunjukkan implementasi dari halaman *login*, *home*, dan halaman untuk membunyikan *alarm* pada aplikasi ***Kasuari Secure*** yang dirancang untuk warga. **Halaman *Login*** menampilkan form sederhana untuk nama pengguna dan sandi, serta tombol "Masuk" yang mengarahkan pengguna ke halaman utama. Desainnya minimalis dengan dominasi warna coklat muda dan logo Kasuari di bagian atas. **Halaman *Home*** memberikan informasi terkait keadaan darurat seperti kebakaran, kejahatan, dan bencana alam, lengkap dengan ilustrasi dan opsi tindakan bagi warga, seperti berkumpul di tempat aman. Tombol navigasi terletak di bagian bawah, dan tombol "Keluar" di bagian atas. **Halaman Membunyikan *Alarm*** digunakan dalam keadaan darurat seperti kebakaran, di mana setelah pengguna menekan tombol alarm, aplikasi akan memberitahu pihak keamanan dan memberi peringatan kepada warga.

Gambar 3.15 Implementasi Halaman Notifikasi (Warga)

Gambar 3.16 Implementasi Halaman Konfirmasi (Warga)

Pada **Gambar 3.15, Gambar 3.16** ditunjukkan implementasi dari halaman notifikasi dan konfirmasi pada aplikasi ***Kasuari Secure*** untuk warga. **Halaman Notifikasi** menampilkan pesan bahwa laporan darurat warga telah diterima oleh pihak keamanan, disertai instruksi untuk tetap waspada. Pengguna diberikan opsi untuk mengonfirmasi laporan tersebut dengan menekan tombol "Konfirmasi". **Halaman Konfirmasi** menunjukkan pemberitahuan spesifik mengenai kejadian darurat, misalnya kebakaran di lokasi tertentu, dengan ikon peringatan besar untuk menarik perhatian pengguna. Tombol "Konfirmasi" kembali disertakan agar warga dapat memverifikasi bahwa mereka telah menerima dan memahami notifikasi tersebut. Desainnya sederhana dan fokus pada komunikasi informasi penting dengan jelas dan cepat.

Gambar 3.17 Implementasi Halaman *Login* (*Security*/*Admin*)



Gambar 3.18 Implementasi Halaman Home (Security/*Admin*)

Gambar 3.19 Implementasi Halaman Konfirmasi (Security/*Admin*)

Pada **Gambar 3.17, 3.18, 3.19** ditampilkan implementasi halaman *login*, *home*, dan konfirmasi pada aplikasi ***Kasuari Secure*** yang digunakan oleh pihak *security* atau *Admin*. **Halaman *Login*** memiliki desain yang mirip dengan halaman *login* untuk warga, dengan *form* untuk memasukkan nama pengguna dan sandi, serta tombol "Masuk". **Halaman *Home*** (gambar kedua) menyajikan beberapa opsi penting, seperti peta lokasi (*Maps*), notifikasi, riwayat laporan (*History*), data warga (*Citizen data*), dan tombol untuk keluar (*Logout*). Menu ini memudahkan pihak keamanan atau *Admin* dalam mengelola dan memantau situasi secara langsung. **Halaman Konfirmasi** menampilkan informasi darurat terkait kebakaran di lokasi spesifik, disertai dengan ikon api besar dan opsi untuk "Menyalakan Alarm" atau langsung "Menghubungi Pemadam Kebakaran". Desain ini dibuat untuk memudahkan *Admin* atau security mengambil tindakan cepat dalam situasi darurat, dengan fokus pada informasi dan akses cepat ke layanan darurat.

Gambar 3.20 Implementasi Riwayat (*Security*/*Admin*)

Gambar 3.21 Implementasi *Ringtone* (*Security*/*Admin*)

Pada **Gambar 3.20**, **3.21** bagian kiri menampilkan **Riwayat Notifikasi**, di mana pengguna dapat melihat daftar notifikasi terkait kejadian tertentu, seperti kebakaran yang terjadi pada "Jalan Kasuari II Blok HB 5 No. 15" pada tanggal 12 Februari 2022. Terdapat opsi untuk memfilter riwayat berdasarkan tanggal dengan tombol *dropdown* di atas.

Bagian kanan adalah tampilan ***Ringtone Setting*** atau pengaturan nada dering. Di sini, pengguna dapat memilih dan mengatur nada dering untuk berbagai jenis kejadian, seperti kebakaran, kejahatan, darurat/bencana, dan kumpul warga, dengan setiap kategori memiliki ikon dan tombol pengaturan terpisah.

3.2.6. Evaluasi

Evaluasi mingguan dalam pengembangan aplikasi ***Kasuari Secure*** dilakukan secara rutin untuk memastikan bahwa proyek berjalan sesuai rencana dan memenuhi kebutuhan pengguna. Evaluasi ini dipimpin oleh kepala proyek yang berperan sebagai fasilitator, menghubungkan antara tim *developer*, tim analisis, dan para *stakeholder*. Tim *developer* melaporkan progres teknis terkait fitur-fitur yang telah dikembangkan, kendala teknis yang dihadapi, serta rencana pengembangan berikutnya. Tim analisis memberikan masukan berdasarkan data penggunaan dan kebutuhan fungsional yang relevan, memastikan bahwa setiap pembaruan memenuhi ekspektasi pengguna akhir.

Dalam evaluasi ini, *stakeholder* dari kelurahan dan pengurus kompleks Kasuari turut hadir untuk memberikan feedback langsung terkait implementasi di lapangan. Mereka berperan dalam memberikan wawasan terkait kebutuhan keamanan lingkungan yang spesifik, serta memastikan aplikasi ***Kasuari Secure*** dapat diintegrasikan dengan sistem keamanan yang sudah ada. Masukan dari *stakeholder* sangat penting untuk memastikan bahwa fitur yang dikembangkan sesuai dengan kebutuhan komunitas. Setiap pertemuan diakhiri dengan pembaruan rencana aksi untuk minggu berikutnya, termasuk penyesuaian prioritas fitur dan solusi atas kendala yang dihadapi.

3.3 Kendala Yang Dihadapi

1. Kerja dari Rumah

Salah satu kendala yang dihadapi dalam pengembangan aplikasi ***Kasuari Secure*** adalah kerja dari rumah (*work from home*). Koordinasi antar anggota tim seringkali terhambat karena keterbatasan komunikasi langsung. Diskusi yang biasanya dapat

diselesaikan secara cepat dalam rapat tatap muka kini harus dilakukan melalui *platform online*, yang kadang mengalami gangguan teknis seperti koneksi internet yang tidak stabil. Selain itu, perbedaan lokasi dan jam kerja anggota tim juga mempengaruhi efektivitas kolaborasi dan produktivitas.

2. Kebutuhan Berubah Secara Mendadak

Kebutuhan aplikasi sering kali berubah secara mendadak, terutama karena masukan baru dari *stakeholder* kelurahan atau pengurus kompleks yang menginginkan penyesuaian fitur berdasarkan perkembangan di lapangan. Perubahan yang tiba-tiba ini membuat tim harus melakukan penyesuaian cepat, yang dapat berdampak pada *timeline* proyek dan kualitas fitur yang sudah dikembangkan. Selain itu, penyesuaian terhadap perubahan mendadak seringkali memerlukan pengembangan ulang yang tidak diantisipasi sebelumnya, menambah beban kerja tim.

3.4 Cara Mengatasi Kendala

1. Kerja dari Rumah

Untuk mengatasi kendala kerja dari rumah, tim menggunakan alat kolaborasi (*Video Conference*) yang efisien seperti *Zoom* atau *Google Meet* untuk memastikan komunikasi tetap lancar. Jadwal rapat rutin diatur untuk sinkronisasi pekerjaan dan pembaruan status proyek. Selain itu, tim mengadopsi metode kerja *agile* dengan sprint mingguan untuk menjaga alur kerja tetap terstruktur meski dilakukan secara *remote*. Penggunaan alat manajemen proyek seperti Jira juga membantu dalam melacak tugas-tugas yang sedang dikerjakan dan memastikan tanggung jawab jelas di setiap anggota tim.

2. Kebutuhan Berubah Secara Mendadak

Dalam menghadapi perubahan kebutuhan yang mendadak, tim mengimplementasikan pendekatan *agile development* dengan fleksibilitas tinggi. Tim beradaptasi dengan perubahan dengan membuat sprint yang lebih pendek dan rapat evaluasi mingguan untuk cepat merespons perubahan. Setiap perubahan yang diusulkan oleh *stakeholder* dievaluasi secara prioritas dan dimasukkan ke *backlog* untuk dipertimbangkan dalam iterasi berikutnya. Dengan demikian, tim dapat tetap menjaga kecepatan tanpa mengorbankan kualitas dan tetap sesuai dengan *timeline* yang telah ditentukan.

3.5 Pembelajaran Yang Diperoleh dari Kerja Profesi

1. **Pendalaman *TypeScript*, *Angular*, dan *Ionic***

   Dalam kerja profesi ini, saya memperdalam pemahaman tentang *TypeScript*, *Angular*, dan *Ionic*, yang sangat berguna dalam pengembangan aplikasi mobile lintas *platform*. *TypeScript* membantu meningkatkan stabilitas kode dengan tipe yang kuat, sementara *Angular* memungkinkan saya membangun antarmuka pengguna yang dinamis dan responsif. Selain itu, pengalaman praktis menggunakan *Ionic* memberikan kemampuan untuk mengembangkan aplikasi yang dapat berjalan lancar di *Android* maupun *iOS*, sehingga lebih efisien dalam menciptakan aplikasi lintas *platform*.

2. **Kerja Sama Tim**

   Saya belajar bahwa kerja sama dan komunikasi efektif sangat penting, terutama dalam proyek yang melibatkan banyak pihak seperti tim *developer*, analis, dan *stakeholder* eksternal. Saya berlatih membagi informasi dengan jelas, mendengarkan dan mempertimbangkan berbagai sudut pandang, serta menghargai masukan dari rekan-rekan demi tercapainya tujuan proyek. Kolaborasi ini memperkuat kemampuan saya dalam memahami peran setiap anggota tim dan memaksimalkan sinergi di dalam proyek.

3. **Manajemen Waktu**

   Dalam menghadapi tugas yang beragam, saya belajar pentingnya manajemen waktu untuk menjaga efisiensi kerja. Dengan membagi waktu antara tugas teknis, rapat, dan dokumentasi, saya bisa menetapkan prioritas dan mencapai target proyek tepat waktu. Pengalaman ini membantu saya lebih disiplin dan terfokus pada tugas yang paling penting, tanpa mengabaikan aspek lainnya, sehingga produktivitas dan kualitas kerja dapat terjaga.