

**PENGEMBANGAN APLIKASI *MOBILE SOCIAL
CROWDSOURCING DAN EARLY WARNING BENCANA ALAM
DENGAN MENGGUNAKAN PUSH NOTIFICATION***

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rizqi Aryansa
NIM: 175150209111009



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENGEMBANGAN APLIKASI MOBILE SOCIAL CROWDSOURCING DAN EARLY WARNING BENCANA ALAM DENGAN MENGGUNAKAN PUSH NOTIFICATION

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Rizqi Aryansa
NIM: 175150209111009

Skripsi ini telah diuji dan dinyatakan lulus pada
24 April 2019

Telah diperiksa dan disetujui oleh:

Pembimbing I

Pembimbing II

Adam Hendra Brata, S.Kom., M.T., M.Sc
NIK: 2016079001051001

Dr. Eng. Herman Tolle, S.T, M.T.
NIP: 19740823 200012 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 8 April 2019

Rizqi Aryansa

NIM: 17515020911109

PRAKATA

Puji syukur kehadirat Allah SWT, atas limpahan berkah dan rahmat-Nya sehingga penulis dapat menyelesaikan Penulisan Skripsi ini dengan judul “Pengembangan Aplikasi *Mobile Social Crowdsourcing* dan *Early Warning* Bencana Alam Dengan Menggunakan *Push Notification*”.

Skripsi ini disusun untuk memenuhi syarat kelulusan dalam menyelesaikan studi pada jenjang Sarjana 1 (S1) Fakultas Ilmu Komputer, Jurusan Teknik Informatika, Program Studi Teknik Informatika Universitas Brawijaya. Terselesaiannya tugas akhir ini tidak lepas dari bantuan dan dukungan berbagai pihak. Oleh karena itu, penulis menyampaikan ucapan terima kasih sebanyak-banyaknya kepada:

1. Allah SWT, yang telah memberikan segala rahmat dan hidayah serta rezeki-Nya termasuk kesehatan yang tidak ternilai harganya sehingga saya mampu menyelesaikan tugas akhir ini dengan tepat waktu.
2. Kedua orang tua saya, Saiful Mufid dan Ermawati, atas dukungan dan perhatiannya dalam mengerjakan tugas akhir ini, dan kakak kandung penulis Achmad Nizam serta adik kandung penulis Moch Ikbal yang juga memberikan begitu banyak dukungan moral sehingga tugas akhir ini dapat terselesaikan tepat pada waktunya.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya
4. Agus Wahyu Widodo, S.T, M.Cs, selaku Ketua Program Studi Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya
5. Bapak Adam Hendra Brata, S.Kom., M.T., M.Sc dan Dr. Eng. Herman Tolle, S.T, M.T selaku pembimbing yang selalu meluangkan banyak waktu dan juga tenaganya untuk memberikan ilmu, saran dan juga bimbingan dalam menyelesaikan skripsi ini.
6. Bapak dan ibu dosen pengampu yang telah memberikan ilmunya dalam proses perkuliahan selama 2 tahun ini. Sehingga ilmu tersebut dapat menjadikan bekal pendukung dalam terselesaiannya skripsi ini.
7. Adik perempuan saya satu – satunya dan rekan skripsi saya, Labina Kirby yang telah membantu proses penelitian skripsi saya sampai selesai.
8. Seluruh keluarga SAP Teknik Informatika 2017, Student Employee MGM 2018 dan teman teman Teknik Informatika Fakultas Ilmu Komputer yang

telah memberikan pelajaran serta pengalaman yang tidak terlupakan dalam kebersamaan 2 tahun di Fakultas Ilmu Komputer.

9. Pihak-pihak lain yang belum penulis sebutkan yang ikut serta dalam membantu dan memberi dukungan serta masukan yang sangat berarti dalam menyelesaikan skripsi ini.

Penulis menyadari bahwa skripsi yang dibuat masih jauh dari kata sempurna, oleh sebab itu penulis mengharapkan kritik dan saran yang bersifat membangun dari semua pihak demi kesempurnaan skripsi ini. Akhir kata penulis mengucapkan terima kasih sebanyak-banyaknya kepada semua pihak yang telah membantu dalam penyusunan skripsi ini, dan saya berharap semoga tugas akhir ini dapat berguna dan bermanfaat bagi penulis maupun pembaca.

Malang, 8 April 2019

Penulis

rizqi.aryansa@gmail.com

ABSTRAK

Rizqi Aryansa, Pengembangan Aplikasi *Mobile Social Crowdsourcing dan Early Warning* Bencana Alam dengan Menggunakan *Push Notification*

Pembimbing: Adam Hendra Brata, S.Kom., M.T., M.Sc dan Dr. Eng. Herman Tolle, S.T, M.T

Bencana merupakan suatu peristiwa yang dapat mengakibatkan menganggu kehidupan dan penghidupan masyarakat yang disebabkan oleh faktor alam maupun faktor manusia, sehingga dapat menyebabkan dampak kerugian seperti kerusakan lingkungan, korban jiwa, harta benda dan dampak psikologis. Tingginya jumlah kejadian bencana alam di Indonesia menjadi permasalahan serius yang harus segera ditangani untuk mengurangi resiko bencana. Dengan perkembangan teknologi *mobile* yang kian pesat, pemanfaatan teknologi yang dapat digunakan oleh masyarakat untuk berbagi mengenai informasi kebencanaan, adalah berbagi lokasi bencana dan informasi geospasial dengan memanfaatkan fitur geotagging dan *push notification* secara *real time*. Selain itu aplikasi berbasis Android yang akan dikembangkan dalam penelitian ini (Kitana) memanfaatkan fitur *crowdsourcing* dimana informasi kebencanaan dapat langsung diperoleh dari pengguna yang lain. Dan menggunakan data API (*Application Programming Interface*) BMKG untuk informasi kebencanaan seperti gempa bumi. Metode yang digunakan dalam penelitian ini adalah *prototyping*. Selanjutnya pengujian yang dilakukan adalah pengujian *black box*, *white box*, pengujian *usability* yang menggunakan SUS (*System Usability Scale*) sebagai instrumen, dan pengujian *performance* fitur *push notification*. Hasil dari pengujian fungsional adalah status valid serta aplikasi berjalan sesuai kebutuhan pengguna. Pengujian *usability* yang menghasilkan nilai 73,5 dimana nilai tersebut termasuk kategori baik dan berhasil. Sedangkan untuk pengujian *performance* dari 10 kasus uji menghasilkan nilai rata-rata 1,6103 detik, dimana berarti sistem sudah memenuhi kriteria *real time*.

Kata kunci: Android, *prototyping*, *crowdsourcing*, *performance*, *usability*, *push notification*, API

ABSTRACT

Rizqi Aryansa, Pengembangan Aplikasi *Mobile Social Crowdsourcing* dan *Early Warning* Bencana Alam dengan Menggunakan *Push Notification*

Supervisors: Adam Hendra Brata, S.Kom., M.T., M.Sc dan Dr. Eng. Herman Tolle, S.T, M.T

Disaster is an event that can result in disrupting people's lives and livelihoods caused by natural factors and human factors, so that it can cause losses such as environmental damage, loss of life, property and psychological impacts. The high number of natural disasters in Indonesia is a serious problem that must be addressed immediately to reduce disaster risk. With the rapid development of mobile technology, the use of technology that can be used by the public to share disaster information is to share disaster locations and geospatial information by utilizing the geotagging and push notification features in real time. In addition, Android-based applications will be developed in this study (Kitana) utilizing the crowdsourcing feature where disaster information can be obtained directly from other users. And use the BMKG API (Application Programming Interface) for disaster information such as earthquakes. The method used in this study is prototyping. Furthermore, the tests carried out were black box testing, white box, usability testing using SUS (System Usability Scale) as an instrument, and testing the performance of the push notification feature. The results of functional testing are valid status and the application runs according to user needs. Usability testing that produces a value of 73,5 where the value includes both good and successful categories. While for testing the performance of 10 test cases it produces an average value of 1.6103 seconds, which means that the system meets the real time criteria.

Keywords: *Android, prototyping, crowdsourcing, performance, usability, push notification, API*

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
PRAKATA	iv
ABSTRAK	vi
ABSTRACT	vii
DAFTAR ISI	viii
DAFTAR TABEL	xiv
BAB 1 PENDAHULUAN	1
1.1 Latar belakang	1
1.2 Rumusan masalah	4
1.3 Tujuan	4
1.4 Manfaat.....	5
1.5 Batasan masalah.....	5
1.6 Sistematika pembahasan.....	5
BAB 2 LANDASAN KEPUSTAKAAN	7
2.1 Kajian Pustaka	7
2.2 Aplikasi <i>Mobile</i>	7
2.3 Bencana Alam.....	8
2.4 <i>Crowdsourcing</i>	8
2.5 <i>Early Warning</i>	9
2.6 Android	9
2.7 Kotlin.....	9
2.8 Android Studio	10
2.9 JSON (<i>Java Script Object Notation</i>)	11
2.10 <i>Push Notification</i>	12
2.10 Metode Prototyping	12

2.10.1 <i>Throw-away Prototyping</i>	13
2.10.2 <i>Evolutionary Prototyping</i>	13
2.11 <i>Geotagging</i>	14
2.12 One Signal	14
2.13 SQLite.....	15
2.14 Google App Script.....	15
2.15 <i>Use Case Diagram</i>	15
2.16 <i>Sequence Diagram</i>	16
2.17 <i>Class Diagram</i>	17
2.18 Pengujian Perangkat Lunak.....	18
2.18.1 Pengujian Unit.....	18
2.18.2 Pengujian Validasi	18
2.19 Pengujian <i>Usability</i>	18
2.20 Pengujian <i>Performance</i>	19
BAB 3 METODOLOGI PENELITIAN	20
3.1 Studi Literatur	21
3.2 Pengumpulan Data	21
3.3 Analisis Kebutuhan	23
3.4 Perancangan.....	24
3.5 Implementasi <i>Prototyping</i>	24
3.6 Implementasi dan Pengujian Sistem	25
3.7 Penarikan Kesimpulan	28
BAB 4 ANALISIS KEBUTUHAN.....	30
4.1 Analisis Kebutuhan	30
4.1.1 Hasil <i>interview</i>	30
4.1.2 Hasil analisis kebutuhan.....	31
4.1.3 Gambaran Umum.....	32
4.2 Identifikasi Aktor	33
4.3 Kebutuhan Fungsional	33

4.4 Kebutuhan Non Fungsional.....	35
4.5 Pemodelan <i>Use case Diagram</i>	35
4.5.1 Pemodelan <i>Use case Scenario</i>	36
4.5.1.1 Use case scenario Login.....	36
4.5.1.2 Use case scenario Register	37
4.5.1.3 Use case scenario Melihat informasi bencana alam	38
4.5.1.4 Use case scenario Membuat posting bencana alam	39
4.5.1.5 Melihat notifikasi early warning	40
4.5.1.6 Mengakses informasi cuaca	40
4.5.1.7 Mengakses informasi cuaca maritim	41
4.5.1.8 Mengubah profil	42
4.5.1.9 Upload foto profil.....	43
4.5.1.10 Mengubah Password akun	44
4.5.1.11 Logout.....	45
4.5.1.12 Upvote posting.....	45
4.5.1.13 Downvote posting.....	46
4.5.1.14 Filter posting	47
4.5.1.15 Melihat detail informasi posting bencana	48
4.6 Analisis Data.....	48
BAB 5 PERANCANGAN	50
5.1 Perancangan Arsitektur Sistem.....	50
5.2 Perancangan <i>Activity Diagram</i>	51
5.2.1.1 Activity diagram login.....	51
5.2.1.2 Activity diagram register	52
5.2.1.3 Activity diagram Melihat informasi bencana alam.....	53
5.2.1.4 Activity diagram membuat posting bencana alam.....	54
5.2.1.5 Activity diagram melihat notifikasi early warning	55
5.2.1.6 Activity diagram melihat informasi cuaca.....	56
5.2.1.7 Activity diagram melihat informasi cuaca maritim.....	57
5.2.2 Perancangan <i>Sequence Diagram</i>	58

5.2.2.1 Sequence diagram melihat informasi bencana alam	58
5.2.2.2 Sequence diagram membuat posting bencana alam	59
5.2.2.3 Sequence diagram melihat notifikasi early warning.....	60
5.2.3 Perancangan Class Diagram.....	61
5.2.4 Perancangan Basis Data	65
5.2.4.1 Perancangan Tabel.....	65
5.2.5 <i>Physical Data Model</i>	68
5.2.6 Perancangan <i>web service</i>	69
5.2.6.1 Menampilkan informasi Bencana Alam	69
5.2.6.2 Membuat Posting Bencana Alam	71
5.2.6.3 Menampilkan notifikasi early warning.....	71
5.2.7 Perancangan Antarmuka Pengguna (<i>Wireframe</i>).....	72
5.2.7.1 Wireframe Login	73
5.2.7.2 Wireframe Register.....	73
5.2.7.3 Wireframe Beranda.....	74
5.2.7.4 Wireframe Tambah Posting.....	75
5.2.7.5 Wireframe Cuaca maritim	76
5.2.7.6 Wireframe Detil cuaca informasi maritim.....	76
5.2.7.7 Wireframe Menerima notifikasi crowdsourcing	77
5.2.7.8 Wireframe Melihat notifikasi early warning	77
5.2.8 Perancangan Antarmuka Pengguna (<i>Wireframe</i>) iterasi 1 ...	78
5.2.8.1 Wireframe perbaikan ukuran button Post	79
5.2.8.2 Wireframe perbaikan ukuran icon downvote dan upvote	80
5.2.8.3 Wireframe perbaikan warna icon downvote dan upvote.....	80
5.2.8.4 Wireframe perbaikan ukuran font informasi cuaca maritim ..	81
5.2.8.5 Wireframe perbaikan ukuran font dan icon sharing	82
5.2.9 Perancangan Algoritme.....	82
5.2.10 Perancangan komponen membuat <i>posting</i> bencana.....	82
5.2.11 Perancangan komponen melihat informasi bencana	83
5.2.12 Perancangan komponen melihat notifikasi <i>early warning</i> .	84

BAB 6 IMPLEMENTASI	86
6.1 Spesifikasi sistem.....	86
6.1.1 Spesifikasi perangkat keras	86
6.1.2 Spesifikasi perangkat lunak	87
6.2 Batasan-batasan implementasi.....	88
6.3 Implementasi basis data	88
6.4 Implementasi algoritme.....	93
6.4.1 Algoritme menambahkan <i>posting</i> bencana	93
6.4.2 Algoritme menampilkan informasi bencana alam.....	96
6.4.3 Algoritme menampilkan notifikasi <i>early warning</i>	97
6.5 Implementasi <i>user interface</i>	98
6.5.1 Implementasi <i>user interface</i> membuat <i>posting</i> bencana alam	98
6.5.2 Implementasi <i>user interface</i> melihat informasi bencana alam	99
6.5.3 Implementasi <i>user interface</i> melihat notifikasi <i>early warning</i>	100
6.5.4 Implementasi <i>user interface</i> mengubah <i>profil</i>	100
6.5.5 Implementasi <i>user interface</i> melihat informasi cuaca.....	101
6.5.6 Implementasi <i>user interface</i> cuaca maritim.....	102
6.5.7 Implementasi <i>user interface</i> detail informasi cuaca maritim	103
6.5.8 Implementasi <i>user interface register</i>	103
6.5.9 Implementasi <i>user interface login</i>	104
BAB 7 PENGUJIAN DAN ANALISIS	106
7.1 Pengujian unit	106
7.1.2 Pengujian <i>method</i> postCrowd() <i>Class</i> : AddPostActivity	106
7.1.3 Pengujian <i>method</i> callReqBencana() <i>Class</i> : BerandaFrgment	110

7.1.4 Pengujian <i>method</i> <i>onListNotifChange()</i> <i>Class</i> : <i>NotifikasiFragmant</i>	112
7.2 Pengujian validasi	114
7.2.1 Pengujian validasi <i>login</i>	114
7.2.2 Pengujian validasi <i>registrasi</i>	115
7.2.3 Pengujian validasi melihat <i>posting</i> bencana	115
7.2.4 Pengujian validasi membuat <i>posting</i> bencana alam	116
7.2.5 Pengujian validasi melihat notifikasi <i>early warning</i>	117
7.2.6 Pengujian validasi mengakses informasi cuaca	117
7.2.7 Pengujian validasi mengakses informasi cuaca maritim....	117
7.2.8 Pengujian validasi mengubah profil	118
7.2.9 Pengujian validasi <i>upload</i> foto profil	119
7.2.10 Pengujian validasi mengubah <i>password</i> akun.....	119
7.2.11 Pengujian validasi mengubah <i>logout</i>	120
7.2.12 Pengujian validasi <i>Upvote posting</i>	120
7.2.13 Pengujian validasi <i>Downvote posting</i>	121
7.2.14 Pengujian validasi <i>Filter posting</i>	121
7.2.15 Pengujian validasi Melihat detail informasi bencana	122
7.3 Pengujian <i>Usability</i>	122
7.3.1 Hasil pengujian dan analisis data SUS.....	122
\BAB 8 PENUTUP	129
8.1 Kesimpulan.....	129
8.2 Saran	130
DAFTAR REFERENSI	131

DAFTAR TABEL

Tabel 2.1 Elemen pada <i>usecase diagram</i>	15
Tabel 2.2 Elemen pada <i>sequence diagram</i>	16
Tabel 2.3 Elemen pada <i>class diagram</i>	17
Tabel 3.1 Daftar pertanyaan <i>user interview</i>	22
Tabel 3.2 Daftar pertanyaan <i>expert interview</i>	22
Tabel 3.3 Daftar responden <i>interview</i>	24
Tabel 3.4 Instrumen kuisioner SUS	25
Tabel 3.5 Kualifikasi hasil skor SUS	28
Tabel 4.1 Hasil <i>user interview</i>	30
Tabel 4.2 Daftar kebutuhan fungsional aplikasi Kitana	31
Tabel 4.3 Identifikasi Aktor	33
Tabel 4.4 Analisis kebutuhan fungsional <i>Guest</i>	33
Tabel 4.5 Analisis kebutuhan fungsional Pengguna	34
Tabel 4.6 Daftar kebutuhan non fungsional	35
Tabel 4.7 <i>Use case Scenario Login</i>	37
Tabel 4.8 <i>Use case Scenario Register</i>	37
Tabel 4.9 <i>Use case Scenario Melihat informasi bencana alam</i>	38
Tabel 4.10 <i>Use case Scenario Membuat posting bencana alam</i>	39
Tabel 4.11 <i>Use case Scenario Melihat notifikasi early warning</i>	40
Tabel 4.12 <i>Use case Scenario Melihat informasi cuaca</i>	41
Tabel 4.13 <i>Use case Scenario Melihat informasi cuaca maritim</i>	41
Tabel 4.14 <i>Use case Scenario Mengubah profil</i>	42
Tabel 4.15 <i>Use case Scenario Upload foto profil</i>	43
Tabel 4.16 <i>Use case Scenario Mengubah Password Akun</i>	44
Tabel 4.17 <i>Use case Scenario Logout</i>	45
Tabel 4.18 <i>Use case Scenario Upvote Posting</i>	46
Tabel 4.19 <i>Use case Scenario Downvote Posting</i>	46

Tabel 4.20 <i>Use case Scenario Filter Posting</i>	47
Tabel 4.21 <i>Use case scenario</i> Melihat detail informasi <i>posting</i> bencana	48
Tabel 5.1 Rancangan tabel basis data Pengguna.....	66
Tabel 5.2 Rancangan tabel basis data bencana.....	66
Tabel 5.3 Rancangan tabel basis data <i>early warning (server)</i>	67
Tabel 5.4 Rancangan tabel basis data <i>early warning (local storage)</i>	68
Tabel 5.5 Perancangan <i>web service</i> menampilkan informasi bencana alam..	70
Tabel 5.6 Perancangan <i>web service</i> membuat <i>posting</i> bencana alam	71
Tabel 5.7 Perancangan <i>web service</i> Menampilkan notifikasi <i>early warning</i> ..	72
Tabel 5.8 Algoritme perancangan komponen membuat <i>posting</i> bencana	83
Tabel 5.9 Algoritme perancangan komponen melihat informasi bencana.....	84
Tabel 5.10 Algoritme perancangan komponen melihat notifikasi <i>early warning</i>	85
 Tabel 6.1 Spesifikasi perangkat keras komputer	86
Tabel 6.2 Spesifikasi perangkat keras <i>smartphone mobile</i>	86
Tabel 6.3 Spesifikasi perangkat keras <i>server</i>	87
Tabel 6.4 Spesifikasi perangkat lunak komputer	87
Tabel 6.5 Spesifikasi perangkat lunak <i>smartphone</i>	87
Tabel 6.6 Spesifikasi perangkat lunak <i>server</i>	87
Tabel 6.7 Implementasi Tabel Pengguna	89
Tabel 6.8 Implementasi Tabel Bencana	89
Tabel 6.8 Implementasi Tabel Bencana (lanjutan)	90
Tabel 6.9 Implementasi Tabel Vote	90
Tabel 6.10 Implementasi Tabel Early Warning	91
Tabel 6.11 Implementasi Tabel Notifikasi (<i>server</i>)	92
Tabel 6.12 Implementasi Tabel Notifikasi (<i>local storage</i>).....	92
Tabel 6.13 <i>Source code Method</i> <code>poxstCrowd () Class: AddPostActivity</code>	94

Tabel 6.14 Penjelasan <i>Source code Method postCrowd()</i> Class : AddPostActivity	95
Tabel 6.15 <i>Source code Method callReqBencana ()</i> Class : BerandaFragment	96
Tabel 6.16 Penjelasan <i>Source code Method callReqBencana ()</i> Class : BerandaFragment	97
Tabel 6.17 <i>Source code Method reqAllNotif()</i> Class : NotifikasiFragment	97
Tabel 6.18 Penjelasan <i>Source code Method reqAllNotif()</i> Class : NotifikasiFragment	98
Tabel 7.1 <i>Pseudocode Method postCrowd()</i> Class : AddPostActivity	107
Tabel 7.2 Hasil Pengujian Unit <i>Method postCrowd()</i> Class : AddPostActivity	109
Tabel 7.3 <i>Pseudocode Method callReqBencana()</i> Class : BerandaFragment	110
Tabel 7.4 Hasil Pengujian Unit <i>Method callReqBencana()</i> Class : BerandaFragment	112
Tabel 7.5 <i>Pseudocode Method onListNotifChange()</i> Class : NotifikasiFragment	112
Tabel 7.6 Hasil Pengujian Unit Method <i>onListNotifChange()</i> Class : NotifikasiFragment	114
Tabel 7.7 Pengujian Validasi <i>Login</i>	114
Tabel 7.7 Pengujian Validasi <i>Login</i> (lanjutan).....	115
Tabel 7.8 Pengujian Validasi <i>register</i>	115
Tabel 7.9 Pengujian Validasi melihat <i>posting</i> bencana.....	116
Tabel 7.10 Pengujian Validasi membuat <i>posting</i> bencana alam	116
Tabel 7.11 Pengujian Validasi Melihat notifikasi <i>early warning</i>	117
Tabel 7.12 Pengujian Validasi Melihat informasi cuaca.....	117
Tabel 7.13 Pengujian Validasi Melihat informasi cuaca maritim	118
Tabel 7.14 Pengujian Validasi Mengubah profil	118

Tabel 7.15 Pengujian Validasi <i>Upload foto profil</i>	119
Tabel 7.16 Pengujian Validasi Mengubah <i>password</i> akun	119
Tabel 7.17 Pengujian Validasi <i>logout</i>	120
Tabel 7.18 Pengujian Validasi <i>Upvote posting</i>	120
Tabel 7.19 Pengujian Validasi <i>Downvote posting</i>	121
Tabel 7.20 Pengujian Validasi <i>Filter posting</i>	121
Tabel 7.20 Pengujian Validasi <i>Filter posting</i> (lanjutan).....	122
Tabel 7.21 Pengujian Validasi Melihat detail informasi bencana.....	122
Tabel 7.22 Hasil pengisian kuisioner SUS	123
Tabel 7.23 Hasil konversi kuisioner SUS	124
Tabel 7.24 Selisih waktu penerimaan notifikasi antara <i>server</i> dan <i>client</i>	127

DAFTAR GAMBAR

Gambar 2.1 <i>Object</i> dalam format JSON (JSON Team, 2018).....	11
Gambar 2.2 <i>Array</i> dalam format JSON (JSON Team, 2018)	11
Gambar 2.3 <i>Value</i> dalam format JSON (JSON Team, 2018)	12
Gambar 2.4 <i>Prototyping Model</i>	13
Gambar 3.1 Diagram Alir Penelitian.....	20
Gambar 3.2 Format respon kuisioner SUS (Sauro, 2011)	27
Gambar 3.3 <i>Grade ranking</i> dari hasil skoring SUS	28
Gambar 4.1 <i>Use case diagram</i>	36
Gambar 5.1 Rancangan Arsitektur Sistem.....	50
Gambar 5.2 <i>Activity diagram Login</i>	52
Gambar 5.3 <i>Activity diagram Register</i>	53
Gambar 5.4 <i>Activity diagram</i> Melihat Informasi Bencana alam	54
Gambar 5.5 <i>Activity diagram</i> Membuat <i>Posting</i> bencana alam	55
Gambar 5.6 <i>Activity diagram</i> Melihat notifikasi <i>early warning</i>	56
Gambar 5.7 <i>Activity diagram</i> Melihat informasi cuaca	57
Gambar 5.8 <i>Activity diagram</i> Melihat informasi cuaca maritim	58
Gambar 5.9 <i>Sequence diagram</i> melihat informasi bencana alam.....	59
Gambar 5.10 <i>Sequence diagram</i> membuat <i>posting</i> bencana alam	60
Gambar 5.11 <i>Sequence diagram</i> melihat notifikasi <i>early warning</i>	61
Gambar 5.12 <i>Class diagram</i> Kitana.....	62
Gambar 5.13 <i>Class diagram</i> <i>Package View</i>	63
Gambar 5.14 <i>Class diagram</i> <i>Package ViewModel</i>	64
Gambar 5.15 <i>Class diagram</i> <i>Package Model</i>	65
Gambar 5.16 <i>Physical Data Model (server)</i>	69
Gambar 5.17 <i>Wireframe Login</i>	73
Gambar 5.18 <i>Wireframe Register</i>	74

Gambar 5.19 <i>Wireframe</i> Beranda.....	75
Gambar 5.20 <i>Wireframe</i> Tambah <i>posting</i>	76
Gambar 5.21 <i>Wireframe</i> Cuaca maritim	77
Gambar 5.22 <i>Wireframe</i> Detil informasi cuaca maritim.....	77
Gambar 5.23 <i>Wireframe</i> Menerima notifikasi <i>crowdsourcing</i>	78
Gambar 5.24 <i>Wireframe</i> Melihat notifikasi <i>early warning</i>	78
Gambar 5.25 <i>Wireframe</i> perbaikan ukuran <i>button Post</i>	80
Gambar 5.26 <i>Wireframe</i> perbaikan ukuran <i>icon downvote</i> dan <i>upvote</i>	81
Gambar 5.27 <i>Wireframe</i> perbaikan warna <i>icon downvote</i> dan <i>upvote</i>	81
Gambar 5.28 <i>Wireframe</i> perbaikan ukuran <i>font</i> informasi cuaca maritim	82
Gambar 5.29 <i>Wireframe</i> perbaikan ukuran <i>font</i> dan <i>icon sharing</i>	82
Gambar 6.1 Implementasi <i>user interface</i> Membuat <i>posting</i> bencana alam ..	99
Gambar 6.2 Implementasi <i>user interface</i> melihat informasi bencana alam.	100
Gambar 6.3 Implementasi <i>user interface</i> Melihat notifikasi <i>early warning</i> .	101
Gambar 6.4 Implementasi <i>user interface</i> mengubah profil	101
Gambar 6.5 Implementasi <i>user interface</i> melihat informasi cuaca.....	102
Gambar 6.6 Implementasi <i>user interface</i> cuaca maritim.....	103
Gambar 6.7 Implementasi <i>user interface</i> detail informasi cuaca maritim ...	104
Gambar 6.8 Implementasi <i>user interface register</i>	104
Gambar 6.9 Implementasi <i>user interface login</i>	105
Gambar 7.1 <i>Flow Graph</i> Pengujian <i>method postCrowd</i>	108
Gambar 7.2 <i>Flow Graph</i> Pengujian <i>method callReqBencana</i>	111
Gambar 7.3 <i>Flow Graph</i> Pengujian <i>method onListNotifChange</i>	113
Gambar 7.4 Daftar fungsi eksekusi dari sisi pengirim (<i>server</i>)	126
Gambar 7.5 Hasil waktu eksekusi penerima (<i>client</i>).....	126

DAFTAR LAMPIRAN

LAMPIRAN 1 HASIL KUISIONER SUS	134
LAMPIRAN 2 HASIL WAWANCARA <i>EXPERT</i>	134
LAMPIRAN 3 HASIL WAWANCARA PENGGUNA.....	139
LAMPIRAN 4 SKENARIO <i>USABILITY TESTING</i>	141

BAB 1 PENDAHULUAN

1.1 Latar belakang

Indonesia merupakan negara kepulauan terbesar di dunia dengan memiliki pulau sebanyak 16.056 yang terletak diantara dua benua yaitu Asia dan Australia serta diantara dua samudera yaitu Samudera Hindia dan Samudera Pasifik. Negara Indonesia terletak diantara pertemuan 3 lempeng dunia yaitu lempeng Indo-Australia, Eurasia dan Pasifik, yang berpotensi terjadinya gempa bumi apabila lempeng-lempeng tersebut bertumbukan. Selain itu, Indonesia juga mempunyai 127 Gunung Api Aktif, sekitar total 76 Gunung yang berbahaya, bencana alam lainnya yang seringkali melanda meliputi tsunami, tanah longsor, gempa bumi dan banjir. Dampak dari bencana alam tersebut mengakibatkan kerugian baik dari segi moril dan materil yang di alami masyarakat Indonesia (BNPB, 2016).

Bencana merupakan suatu peristiwa yang dapat mengakibatkan menganggu kehidupan dan penghidupan masyarakat yang disebabkan oleh faktor alam maupun faktor manusia, Sehingga dapat menyebabkan dampak kerugian seperti kerusakan lingkungan, korban jiwa, harta benda dan dampak psikologis pada masyarakat (BNPB, 2018). Pada periode tahun 2005 hingga 2015, di Indonesia lebih dari 78% (11.648) bencana yang terjadi merupakan bencana hidrometeorologi, sedangkan bencana geologi terjadi sebanyak 22% (3.810). Bencana hidrometerologi antara lain meliputi bencana banjir, gelombang ekstrim, kebakaran lahan dan hutan, kekeringan, dan cuaca esktrim. Sedangkan bencana geologi merupakan bencana seperti halnya gempa bumi, tsunami, letusan gunungapi, dan tanah longsor (DIBI-BNPB, 2018). Meskipun data menunjukkan bahwa persentase bencana geologi lebih kecil dibandingkan dengan bencana hidrometeorologi, namun pada kenyataannya kerugian yang dialami pasca bencana menimbulkan dampak kerugian ekonomi yang cukup besar dan juga di sisi korban.

Berdasarkan data dan informasi dari Badan Nasional Penanggulangan Bencana (BNPB) Indonesia, jumlah Bencana Alam yang terjadi di Indonesia pada tahun 2010 – 2018 terakhir sebanyak 16.966 dan menyebabkan banyak korban jiwa meninggal dunia dan kerusakan tempat tinggal. Jumlah korban jiwa bencana gempa bumi dari tahun 2014 hingga 2019 adalah sebanyak 7.796 jiwa (BNPB 2019). Tingginya jumlah kejadian bencana alam di Indonesia menjadi permasalahan serius yang harus segera ditangani untuk mengurangi resiko bencana. Maka dari itu dapat dilakukan upaya penanggulangan dengan pembuatan peta bencana yang akan digunakan sebagai penanganan pra bencana ataupun meningkatkan tingkat kewaspadaan di daerah rawan bencana.

Berkembangnya teknologi yang kian cepat tidak lepas dari kemajuan perangkat *mobile* kini yang semakin pesat dengan jumlah pengguna setiap tahunnya meningkat. Negara Indonesia merupakan salah satu jumlah pengguna terbanyak menggunakan *smartphone* saat ini. Hal ini dibuktikan berdasarkan Data *Global Mobile Market Report* pada September 2018 di situs Newzoo yang menunjukkan bahwa Negara Indonesia menduduki peringkat ke 6 dengan total jumlah pengguna kurang lebih 73 juta. Pengguna *smartphone* di Indonesia yang menggunakan *operating system* Android mengalami peningkatan sebesar 90% yang dibuktikan dari data *Mobile Operating System Market*. Berdasarkan data tersebut, aplikasi *mobile crowdsourcing* bencana alam yang dikembangkan akan menggunakan platform Android.

Pada penelitian yang sebelumnya yang berjudul “Rancang Bangun Aplikasi Geotagging Social Report Bencana Banjir” menjelaskan tentang sistem yang dapat dijadikan wadah sumber informasi tentang pelaporan peristiwa bencana banjir yang dilakukan oleh masyarakat di daerah sekitarnya dalam bentuk gambar ataupun peta geografis, harapan dari aplikasi ini dapat memudahkan pengguna mencari informasi bencana banjir (Wijaya, Tolle dan Kharisma, 2018).

Dengan perkembangan *mobile* yang semakin pesat, hal itu dapat dimanfaatkan oleh pengguna untuk saling berinteraksi dengan pengguna lain, misalnya untuk melakukan pertukaran informasi dengan berbagai lokasi bencana dengan memanfaatkan fitur GPS. Maka dari itu, informasi yang didapatkan dari pengguna lain dapat dijadikan data untuk pemetaan bencana yang akan datang agar terhindar dari resiko yang dapat mengancam jiwa dan kerugian harta.

Salah satu aplikasi bencana yang tersedia saat ini adalah Info BMKG. Saat ini, informasi bencana yang disampaikan hanya mencakup wilayah-wilayah tertentu dan tidak adanya perhimpunan data (*crowdsourcing*). Kepala BMKG Prof. Ir. Dwikorita Karnawati, M.Sc, Ph.D menyampaikan tentang perlu adanya pengembangan *crowdsourcing*, supaya informasi dapat diakses lebih cepat. Selain itu juga terdapat aplikasi yang dikembangkan oleh BNPB yaitu PetaBencana.id. Aplikasi PetaBencana.id merupakan *website* yang digunakan untuk *crowdsourcing* bencana banjir. PetaBencana.id menghimpun data pelaporan banjir melalui berbagai sumber diantaranya Qlue, Z-Alert, PasangMata dan Twitter dengan memanfaatkan *hashtag* banjir. Saat ini, PetaBencana.id masih menjangkau beberapa kota besar seperti Jakarta, Bogor, Depok, Tangerang, Bekasi, Bandung, Semarang dan Surabaya.

Crowdsourcing merupakan suatu metode yang dilakukan oleh suatu kelompok atau individu, berkaborasi untuk melakukan suatu tugas; misalnya berbagi informasi maupun mengerjakan sesuatu yang telah ditentukan (Estellés-

arolas dan González-ladrón-de-guevara, 2012). Aplikasi yang akan dikembangkan dalam penelitian ini juga menggunakan teknologi *geotagging*. *Geotagging* membantu pengguna untuk mengetahui lokasi kejadian bencana. Selain itu untuk membantu supaya informasi bencana dapat tersampaikan ke pengguna secara *real time*, maka diperlukan teknologi *push notification*. Teknologi *geotagging* merupakan bagian dari fitur kamera yang dapat melakukan integrasi langsung dengan GPS. *Geotagging* juga dapat digunakan untuk menambahkan metadata geospasial media digital seperti foto, video, pesan teks, *tweets*, dan halaman web (Holdener, 2011). Informasi geospasial didapatkan dari teknik *geocoding* dengan menerjemahkan koordinat lintang dan bujur, akurasi, ketinggian dan posisi arah yang didapatkan dari teknologi GPS (*Global Positioning System*). Dengan adanya teknologi *geotagging*, pengguna dapat melakukan pelaporan data bencana yang berisi informasi geospasial di lokasi bencana alam yang sedang terjadi. Agar informasi bencana dapat disampaikan ke pengguna secara *real time*, maka diperlukan teknologi *push notification* pada perangkat *mobile*. Teknologi *push notification* merupakan salah satu bagian dari perangkat *mobile* yang mendapatkan informasi berupa notifikasi secara *real time* dari penyedia informasi (*server*) secara otomatis.

Aplikasi yang akan dikembangkan dalam penelitian ini menyediakan fitur *push notification* yang di dalamnya terdapat informasi *early warning* bencana alam yang efektif dan *real time* bagi pengguna. Untuk mendukung aspek *real time* tersebut, pada penelitian ini dilakukan pengujian *performance*. Pengujian *performance* dilakukan untuk menguji performansi kebutuhan non fungsional. Pengujian *performance* dilakukan untuk memvalidasi: kecepatan, kapasitas, skalabilitas, stabilitas, ekspektasi dan *environment* (EPAM, 2008). Hal tersebut bermanfaat ketika dalam kondisi bahaya, pengguna dapat mengambil tindakan seperti menghindari atau mengurangi risiko dari dampak bahaya tersebut. Informasi yang berupa *early warning* memungkinkan individu dan masyarakat untuk melindungi kehidupan dan properti mereka, selain itu juga dapat membantu mengurangi kerugian ekonomi atau mengurangi jumlah cedera atau kematian akibat bencana alam. Informasi peringatan dini memberdayakan masyarakat untuk mengambil tindakan sebelum terjadi bencana.

Dalam mengembangkan wadah pertukaran informasi bencana, perlu pemanfaatan teknologi *mobile* yang menekankan *geotagging*. Pelaporan bencana alam akan dilakukan oleh pengguna aplikasi dengan mengambil foto pada tempat kejadian dengan mengisi beberapa informasi bencana alam yang akan dilaporkan. Setelah pengambilan objek foto kejadian dan informasi terkait bencana alam dikirim ke *server*. Selanjutnya data pelaporan bencana alam berhasil dikirim ke

server, sistem akan mengirim notifikasi secara otomatis ke perangkat pengguna yang menggunakan aplikasi tersebut.

Agar manfaat diatas dapat dicapai, perlu dilakukan penelitian yang dapat membantu mengatasi permasalahan yang telah diulas sebelumnya yakni dengan mengembangkan aplikasi *crowdsourcing* dan *early warning* bencana alam yang juga memanfaatkan *push notification*. Metode yang digunakan dalam penelitian ini adalah *prototyping*. Metode *prototyping* digunakan karena dapat mendemonstrasikan konsep, percobaan rancangan, dan berfungsi untuk menemukan lebih banyak masalah dan menciptakan solusi (Sommerville, 2011). Hal tersebut berpengaruh dalam pengembangan aplikasi bencana alam ini mengingat kebutuhan pengguna masih belum terdefinisikan dengan jelas, oleh karena itu iterasi dalam metode tersebut perlu dilakukan supaya aplikasi dapat memenuhi kebutuhan pengguna. Terdapat beberapa pengujian yang dilakukan setelah dilakukannya implementasi yakni pengujian unit, pengujian validasi, pengujian *performance* dan pengujian *usability*. Pengujian *usability* yang bertujuan untuk mengukur seberapa mudah antarmuka digunakan (Nielsen, 2012). Pengujian yang dilakukan dengan menggunakan teknik *usability testing*. Pengujian *usability* perlu dilakukan mengingat aplikasi ini akan digunakan di kalangan masyarakat dengan latar belakang yang berbeda, selain itu juga bertujuan supaya aplikasi ini dapat digunakan dengan mudah.

1.2 Rumusan masalah

Dari latar belakang yang telah dipaparkan maka rumusan masalah menjadi dasar penelitian yaitu:

1. Bagaimana hasil analisis kebutuhan dalam pengembangan aplikasi *crowdsourcing* bencana alam?
2. Bagaimana rancangan dan implementasi aplikasi *crowdsourcing* dan *early warning* bencana alam dengan menggunakan metode *prototyping*?
3. Bagaimana hasil pengujian fungsional dan non fungsional aplikasi *crowdsourcing* bencana alam?

1.3 Tujuan

Mengacu pada rumusan masalah yang telah dijabarkan, maka dapat dirumuskan tujuan dilakukannya penelitian ini yaitu:

1. Mengetahui kebutuhan pengguna dalam pengembangan aplikasi *crowdsourcing* bencana alam.

2. Mengetahui hasil perancangan dan implementasi aplikasi *crowdsourcing* yang dikembangkan dengan metode *prototyping*.
3. Mengetahui hasil pengujian fungsional dan non fungsional aplikasi *crowdsourcing* bencana alam.

1.4 Manfaat

Berdasarkan dari tujuan-tujuan yang telah dipaparkan, maka penelitian ini juga memperoleh manfaat yaitu:

1. Menyediakan media informasi kepada masyarakat terkait kejadian bencana alam dengan aplikasi *mobile* guna memberikan peringatan dini yang akan terjadi kedepan di lokasi yang ditentukan.
2. Membantu masyarakat dalam melakukan pelaporan bencana alam secara langsung melalui aplikasi *mobile*.

1.5 Batasan masalah

Pada penelitian ini, batasan–batasan masalah yang dikemukakan oleh penulis sebagai berikut:

1. Perancangan dan implementasi aplikasi *mobile* bencana alam dengan menggunakan *geotagging* dan *push notification* hanya berjalan di perangkat bergerak berbasis Android.
2. Penelitian ditujukan pada objek bencana alam yang terdapat di wilayah Indonesia.
3. Penelitian pada aplikasi *mobile* bencana alam ini hanya membahas informasi mengenai cuaca dan keadaan ombak pantai, sedangkan untuk fitur *early warning* tentang bencana banjir, gempa bumi dan tanah longsor.

1.6 Sistematika pembahasan

BAB 1 Pendahuluan

Bab ini membahas tentang gambaran umum isi penelitian seperti latar belakang, identifikasi masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian serta sistematika pembahasan.

BAB 2 Landasan Kepustakaan

Bab ini berisikan mengenai kajian terhadap beberapa teori yang berkaitan mendukung penelitian.

BAB 3 Metodologi Penelitian

Bab ini menjelaskan tentang beberapa hal mengenai metode dari penelitian, metode pendekatan, metode pengumpulan data.

BAB 4 Analisis Kebutuhan

Bab ini menjelaskan tentang identifikasi pengguna, kebutuhan fungsional, kebutuhan non fungsional, pemodelan *use case diagram* dan *use case scenario*.

BAB 5 Perancangan

Bab ini menjelaskan tentang perancangan aplikasi bencana alam yang memuat identifikasi pengguna, *activity diagram*, *sequence diagram*, *class diagram*, basis data, rancang *web service*, perancangan antarmuka pengguna dan rancang algoritma.

BAB 6 Implementasi

Pada bab implementasi menjelaskan tahapan – tahapan implementasi dari rancangan sistem yang telah dilakukan sebelumnya.

BAB 7 Pengujian dan Analisis

Pada bab pengujian menjelaskan mengenai metode pengujian dan analisis terhadap implementasi yang telah diuji.

BAB 8 Penutup

Bab ini berisikan mengenai penarikan kesimpulan dari apa yang telah didapatkan pada saat penelitian serta saran yang bermanfaat bagi pihak terkait.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Sub bab ini mendeskripsikan tentang penelitian sebelumnya yang meneliti mengenai bencana alam. Penelitian yang dilakukan oleh Wijaya, Tolle dan Kharisma (2018), pada skripsi yang berjudul “Rancang Bangun Aplikasi Geotagging Social Report Bencana Banjir” menjelaskan tentang sistem yang menyediakan informasi mengenai bencana banjir secara akurat di lokasi tertentu, serta disediakannya fitur pelaporan masyarakat dengan mengambil gambar dan objek lokasi yang ditentukan dengan memanfaatkan teknologi *geotagging*. Selain itu, pada penelitian yang berjudul “Rancang Bangun Aplikasi Perangkat bergerak Pemantau Prakiraan Cuaca Maritim di Perairan Indonesia” memaparkan bahwa aplikasi dapat digunakan untuk memantau prakiraan cuaca maritim di perairan Indonesia dan juga mendapatkan notifikasi peringatan dini ketika terjadi prakiraan cuaca di lokasi yang telah ditentukan sebelumnya oleh pengguna (Permana, Kharisma dan Tolle, 2015).

Pada penelitian pertama membahas mengenai *social report* untuk bencana banjir. Sedangkan pada penelitian kedua menjelaskan tentang mengenai prakiraan cuaca maritim di perairan Indonesia dengan memanfaatkan data *third party* seperti *World Weather Online* (API). Sedangkan dalam penelitian ini, peneliti menggabungkan fitur dari kedua penelitian tersebut serta mengembangkan dan menambahkan fitur berupa *push notification* secara *real time* dan menggunakan data dari API BMKG.

2.2 Aplikasi *Mobile*

Aplikasi *mobile* (*mobile application*) merupakan suatu perangkat lunak yang berjalan pada *mobile device* seperti di *smartphone*, *Tablet*, *iPod*, *e-reader*, personal digital assistant (PDA) yang memiliki *operating system* (Android, iOS, Blackberry, Windows Phone) dan dapat terkoneksi ke jaringan internet melalui nirkabel ataupun *wifi* (Gahran, 2011).

Aplikasi *mobile* memiliki beberapa karakteristik yakni mudah digunakan, *user friendly*, murah, dapat diunduh secara fleksibel dan dapat berjalan di perangkat *mobile* kelas bawah. *Mobile application* atau aplikasi *mobile* mempunyai banyak fungsi seperti melakukan panggilan, mengirimkan pesan, *browsing*, *chatting*, komunikasi jejaring sosial, audio, video, *game* dan sebagainya.

2.3 Bencana Alam

Bencana alam geologis terjadi dalam permukaan bumi meliputi gempa bumi, tanah longsor, tsunami, gunung meletus dan lain sebagainya. Sedangkan untuk kategori bencana alam klimatologis terjadi dikarenakan perubahan iklim yang ekstrem yang meliputi seperti banjir, angin puting beliung dan lain sebagainya (BNPB, 2016). Menurut Undang – Undang Nomor 24 Tahun 2007 Tentang Penanggulangan Bencana menyebutkan bahwa bencana merupakan suatu peristiwa atau rangkaian peristiwa yang mengancam dan mengganggu kehidupan dan penghidupan masyarakat yang disebabkan oleh faktor alam atau non alam maupun faktor manusia yang mengakibatkan timbulnya kerugian seperti korban jiwa manusia, kerugian harta benda, kerusakan lingkungan dan dampak psikologi.

Bencana alam merupakan bencana yang diakibatkan oleh peristiwa atau rangkaian peristiwa yang disebabkan oleh alam yang terjadi karena adanya perubahan pada kondisi alam baik secara perlahan maupun secara ekstrem yang meliputi berupa gempa bumi, tsunami, gunung meletus, banjir, angin topan dan tanah longsor atau juga karena faktor campur tangan manusia yang tidak bertanggung jawab (BNPB, 2016).

2.4 Crowdsourcing

Crowdsourcing terdiri dari dua kata yaitu *crowd* dan *source*. “*Crowd*” diartikan sebagai kerumunan orang, sedangkan “*sourcing*” dari kata kerja *source* yang artinya sumber daya. Sehingga dapat diartikan secara keseluruhan bahwa *crowdsourcing* merupakan suatu aktifitas atau tindakan yang dilakukan oleh masyarakat dan dibagikan secara bebas, terbuka ke banyak orang yang terhubung dalam suatu jaringan atau terkoneksi dengan internet (Howe, 2006). Jadi konsep di dalam *crowdsourcing* dapat disimpulkan bahwa adanya keterlibatan yang tidak terbatas, juga tidak memandang latar belakang; baik amatir atau professional, kewarganegaraan, pendidikan, agama. Konsep ini berlaku bagi setiap orang yang ingin memberikan kontribusinya atau solusinya atas suatu permasalahan yang dibagikan oleh individu, perusahaan atau institusi, baik dibayar secara royalti ataupun dengan sukarela.

Crowdsourcing merupakan jenis aktifitas *online* yang dilakukan oleh individu, lembaga, organisasi atau perusahaan dengan memberikan tugas kepada kelompok atau individu melalui panggilan terbuka yang sifatnya fleksibel. Setiap orang yang melakukan *crowdsourcing* mendapatkan keuntungan kepuasan bersama dengan jenis kebutuhan yang diberikan baik dari segi ekonomi, pengakuan sosial dan mengembangkan keterampilan individu (Estellés-arolas dan González-ladrón-de-guevara, 2012).

2.5 Early Warning

Peringatan dini atau *early warning* adalah elemen utama dari pengurangan risiko bencana. *Early warning* dapat mengurangi korban jiwa dan dampak ekonomi akibat bencana alam. Agar efektif, sistem peringatan dini perlu melibatkan masyarakat yang berisiko terkena bencana alam seperti mengedukasi masyarakat mengenai kesadaran terkait risiko bencana alam, dan memberikan peringatan dini dengan memastikan kondisi atau tempat yang berisiko siaga terhadap bencana alam (Anon., 2006).

Early warning merupakan penyedia informasi yang efektif dan tepat waktu bagi pengguna dimana ketika dalam kondisi bahaya pengguna dapat mengambil tindakan seperti menghindari, mengurangi resiko dari dampak bahaya tersebut agar lebih siap siaga jika terjadi respon yang membahayakan. Konsep dasar dibalik *early warning* adalah semakin awal dan akurat dalam memprediksi jangka pendek dan jangka panjang terkait potensi risiko bahaya pada alam dan manusia, semakin besar kemungkinan dapat mengelola dan mengurangi dampak bencana pada masyarakat, ekonomi, dan lingkungan (UNSDR, 2010).

2.6 Android

Platform Android disebut sebagai sistem operasi Google yang berupa *mobile* dan berbasis open source Linux bersama atau Open Handset Allience (OHA). Android menyediakan *Standart Development Kit* (SDK) berfungsi sebagai alat-alat perangkat dan *Application Programming Interface* (API) juga digunakan buat membangun aplikasi pada *platform* Android memakai bahasa pemrograman Java dan terakhir Android Studio versi 3.0 sudah support dengan bahasa pemrograman baru yaitu Kotlin.

Android mempunyai total pengguna cukup banyak di Indonesia dibandingkan dengan sistem operasi iOS yaitu dengan jumlah 95%. Di regional kawasan Asia Pasifik, negara Indonesia mempunyai total *user platform* Android terbanyak, yaitu dengan jumlah 34% (Toro dan Rohman, 2018). Terdapat sebanyak 900 juta perangkat diaktifkan di seluruh dunia hingga Mei 2013, dan telah dipasang pada Google Play. Maka sistem operasi ini dapat untuk digunakan sebagai pengembangan aplikasi *mobile crowdsourcing* bencana alam banyaknya pengguna sistem operasi Android di Indonesia.

2.7 Kotlin

Pada tanggal 17 Mei 2017 Google mengumumkan Kotlin sebagai bahasa pemrograman *official* Android. Google menerapkan Kotlin sebagai bahasa kelas satu bagi Android bersama dengan Java dan C++. Bahasa pemrograman Kotlin

nantinya Google memastikan sudah terintegrasi dengan baik seperti fitur baru di Android, IDE, *framework* dan keseluruhan *library*. Kotlin merupakan sebuah bahasa pemrograman dengan pengetikan statis yang berjalan pada Mesin *Virtual Java* ataupun menggunakan kompiler LLVM yang dapat pula dikompilasikan ke dalam bentuk kode sumber Javascript.

Kotlin dikembangkan oleh tim *programmer* JetBrains yang terletak di Saint Petersburg, Rusia. Bahasa pemrograman Kotlin diambil dari nama sebuah pulau di Rusia, sama halnya dengan bahasa pemrograman Java diambil dari nama pulau Jawa (atau dalam bahasa Inggris disebut dengan sebutan Java) di Indonesia. JetBrains mendesain bahasa pemrograman Kotlin dengan pemahaman dari *Java Developer* dan *IntelliJ* sebagai IDE utamanya. Kotlin sebagaimana besar struktur kodennya mirip dengan apa yang ada pada Java dan bergantung kepada kode bahasa Java, seperti berbagai *framework* Java yang ada. Kelebihan Kotlin dibanding Java yaitu mempelajari waktu pemahaman konsep dasar dengan waktu yang singkat. Selain itu, Kotlin juga bahasa pemrograman yang modern yang mudah untuk dipelajari, sederhana, lebih manusiawi, aman (*safe*) dari *null pointer exception* dan efisien (Toro dan Rohman, 2018).

2.8 Android Studio

Aplikasi Android dibangun dengan sebuah *tool* Android Studio yang merupakan *Integrated Development Environment* (IDE) official dengan memiliki sifat *open source* untuk mengembangkan sebuah aplikasi pada *platform* Android. Rilisnya Android Studio ini diinformasikan oleh Google pada 16 Mei 2013 pada acara Google I/O Conference yang dibawakan Manager Produk dari Google yaitu Ellie Powers. Mulai saat itu, sebuah *tool* Android Studio menggantikan sebuah perangkat lunak Eclipse yang mempersiapkan *tool* pengembang Android yang berintegrasi untuk *develop* dan *debugging* aplikasi.

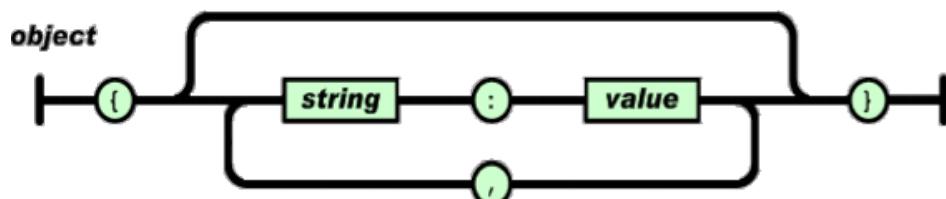
Android Studio merupakan *platform* editor yang memiliki proses *refactoring* dan memperbaiki secara langsung dengan tangkas, hal itu dikarenakan terdapat *tool* bernama Lint digunakan sebagai merespon kemampuan, kompatibilitas pada versi Android dan masalah lainnya. Android Lab juga sebagai *tool editor* tata letak yang memiliki fitur yang dapat digunakan untuk *drag and drop* elemen *user interface*. Android Studio tersedia dibawah lisensi Apache. IDE tersebut berada di tahap awal *preview* mulai dari versi 0.1, lalu tahap selanjutnya dari versi 0.8 (beta) yang diluncurkan pada Juni 2014. *Tools* Android Studio dengan versi final 1.0 diluncurkan pada Desember 2014, IDE ini mulai diluncurkan untuk diunduh di sistem operasi Windows, Linux dan MacOS (Hermawan, 2011).

2.9 JSON (*Java Script Object Notation*)

Javascripts Objects Notation (JSON) merupakan sebuah format pertukaran data komputer. Format JSON yaitu berbentuk teks, mudah dipahami dengan manusia, digunakan sebagai menampilkan struktur data dengan mudah dan bersifat independen tidak terpengaruh dari bahasa pemrograman manapun (Kurniaji, 2015). Salah satu contoh komponen dalam JSON ditunjukkan dalam Gambar 2.1, Gambar 2.2, dan Gambar 2.3.

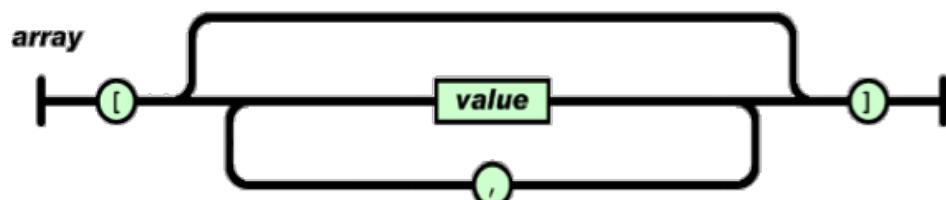
Dalam penelitian ini format data JSON digunakan untuk mendapatkan informasi lokasi dari *Geolocation API*. Format data JSON memiliki dua struktur yaitu *object* dan *array*. *Object* adalah sebuah set nilai yang tidak berurut dengan tanda kurung buka kurawal dan terakhir dengan tanda penutup kurung kurawal. Setiap nilai dibarengi tanda (:) titik dua dan masing masing pasangan di tanda koma (,). Sedangkan *lарik* merupakan nilai - nilai yang berurutan dengan simbol kurung pembuka kotak ([]) dan terakhir dengan tanda kurung kotak penututup (]). Setiap nilai dibedakan dengan tanda koma (,). Berikut ini merupakan representatif dari *format JSON* (JSON, 2019).

1. *Object*



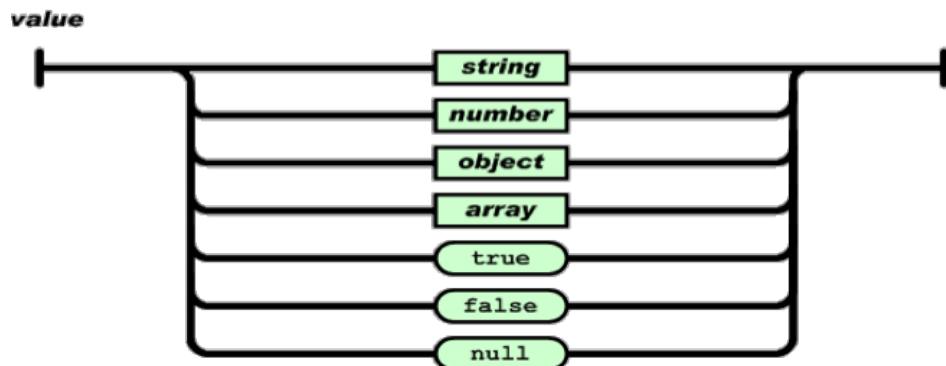
Gambar 2.1 *Object* dalam format JSON (JSON Team, 2018)

2. *Array*



Gambar 2.2 *Array* dalam format JSON (JSON Team, 2018)

3. *Value*



Gambar 2.3 *Value* dalam format JSON (JSON Team, 2018)

2.10 *Push Notification*

Layanan *push notification* merupakan fitur yang disediakan *platform smartphone* untuk menerima informasi dari perangkat lain. Aplikasi *mobile* yang mengirim *push notification* membutuhkan sebuah koneksi internet. Cara kerja untuk melakukan *push notification* adalah dengan mengirim informasi ke pengguna yang telah menginstal aplikasi tersebut ataupun ketika aplikasi tidak sedang dijalankan (Edmunds dan Morris, 2000).

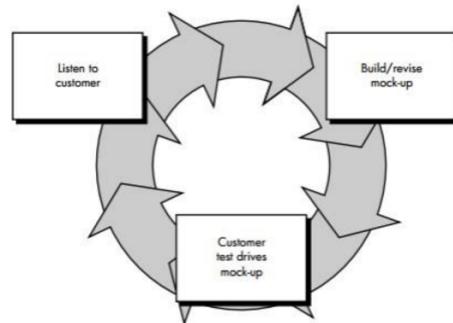
Untuk mengirim sebuah notifikasi, aplikasi menyiapkan objek notifikasi dan mendaftarkan *event* untuk memicu notifikasi terjadi. Penanganan notifikasi tergantung pengaturan *user* pada aplikasi yang telah *di-install*, notifikasi yang berisi informasi bisa ditampilkan langsung di layar *smartphone* atau *user* tidak perlu mengaktifkan notifikasi (Xu dan Zhu, n.d.).

2.10 Metode Prototyping

Prototyping merupakan salah satu jenis *lifecycle* dalam pengembangan perangkat lunak. Metode prototyping bertujuan untuk mengurangi resiko yang signifikan seperti kelalaian dan kesalahan pada proses analisis kebutuhan perangkat lunak. *Prototyping* juga dapat mengurangi masalah proses kebutuhan perangkat lunak, serta memiliki biaya pengembangan yang lebih rendah. Sebuah *prototype* sendiri merupakan versi awal dari sebuah sistem atau perangkat lunak yang digunakan untuk mendemonstrasikan konsep-konsep, percobaan rancangan, dan berfungsi untuk menemukan lebih banyak masalah dan menciptakan solusi (Sommerville, 2011).

Konsep *prototyping* merupakan mekanisme untuk mencapai validasi kebutuhan sebelum melakukan tahap implementasi desain. *Prototyping* dapat digunakan dalam pengembangan perangkat lunak jika kebutuhan pengguna belum terdefinisi dengan jelas. Proses pengembangan perangkat lunak dengan metode *prototyping* ini melalui tahap memvalidasi spesifikasi kebutuhan

pengguna bersamaan dengan proses implementasi desain. Dari gambar model 2 dibawah ini terdapat 3 (tiga) proses utama dalam *prototyping* model, yakni tahap pertama adalah *listen to customer* yang pertama pengembang bertemu dengan calon pengguna, berkomunikasi dengan pengguna untuk sama-sama menentukan tujuan, persyaratan dan kebutuhan pengguna (Pradipta, Yuli Adam Prasetyo, ST. dan Nia Ambarsari, S.Si., 2015). Tahap selanjutnya adalah *build/revise mockup*, yakni pengembang mulai membangun sebuah *mockup (prototype)* dan melakukan revisi jika diperlukan. Berikutnya pengembang mengujikan *mockup* atau *prototype* tersebut ke pengguna.



Gambar 2 *Prototyping Model*

(Sumber: Khosrow-Pour, 2005)

Gambar 2.4 *Prototyping Model*

Pada metode prototyping memiliki 2 (dua) jenis pemodelan, yaitu *throw-away prototyping* dan *evolutionary prototyping*.

2.10.1 *Throw-away Prototyping*

Throw-away prototyping merupakan salah satu jenis pemodelan pada metode *prototyping* yang digunakan untuk mengembangkan sistem yang tidak memiliki pemahaman analisis kebutuhan yang jelas tentang perangkat lunak yang akan dikembangkan. Model ini berfungsi untuk menyempurnakan dan memperjelas spesifikasi kebutuhan perangkat lunak. *Prototype* yang dibangun akan terus dievaluasi sampai kebutuhan pengguna dapat terpenuhi. Setelah proses evaluasi, *prototype* akan dibuang dan tidak dipakai sebagai dasar pengembangan perangkat lunak lebih lanjut (Sommerville, 2011).

2.10.2 *Evolutionary Prototyping*

Evolutionary prototyping merupakan jenis model dari metode *prototyping* yang dimana sistem dikembangkan secara bertahap sehingga mudah untuk dimodifikasi dalam menanggapi masukan atau saran dari pengguna. Tujuan dari *Evolutionary Prototyping* untuk membangun *prototype* yang sangat kuat dengan

cara yang terstruktur dan terus menyempurnakan dengan mengembangkan menjadi kebutuhan fungsional yang dibutuhkan *user*. Penyempurnaan *prototype* terus dilakukan untuk mendapatkan fungsionalitas yang dibutuhkan pengguna dari sistem yang akan dibuat. *Prototype* ini kemudian dilanjutkan dengan produksi, yang nantinya *prototype* evolusioner akan menjadi sistem aktual (McConnell, 1996).

2.11 Geotagging

Teknologi *geotagging* mengarah pada proses penambahan identifikasi metadata geospasial pada media yang beragam, seperti *e-book*, dokumen naratif, konten *website*, gambar, video dan aplikasi sosial media seperti Facebook, Twitter dan sebagainya (Hunter, 2012). Secara otomatis, *Global Positioning System* (GPS) yang terintegrasi di *hardware*, dan beragam *tools* web atau aplikasi akan membantu pengguna untuk menambahkan informasi geospasial kedalam konten web, fotografi, audio, dan video (Peterson, 2008).

Implementasi *geotagging* dapat ditemukan dalam domain tekstual seperti pada halaman web, blog, artikel ensiklopedia, *spreadsheets*, dan sebagainya. Berbagai macam pendekatan dilakukan, seperti dengan menggunakan konteks informasi geospasial yang dapat disimpan dalam konten web. Berbagai pendekatan menggunakan konteks informasi geospasial dilakukan oleh penulis secara manual, perangkat merekam langsung lokasi, menentukan lokasi dari server, sedangkan teknologi *geotagging* dapat merekam lokasi (geoparsing dan geocoding) yang ada secara otomatis (Scharl, 2007).

2.12 One Signal

Salah satu layanan *push notification* untuk *website* dan aplikasi *mobile* (*cross-platform*) adalah OneSignal. OneSignal menyediakan RESTful *server API*, *online dashboard* untuk memonitor performa, operasi *push notification* langsung dari *web service* dan juga untuk melihat statistic pengguna (OneSignal, 2018). Beberapa kelebihan yang dipunyai oleh OneSignal adalah (Gumelar, 2017):

1. Implementasi *interface* yang melalui GCM/FCM milik Google, dan APNS Apple telah didukung oleh OneSignal.
2. *Tools* tambahan yang disediakan oleh OneSignal: A/B *Testing*, *segment targeting*, *variable-substitution*, *localization*, koversi *tracking* dan *drop marketing*.
3. SDK *cross-platform mobile development* seperti Unity, PhoneGap, Cordova, Ionic, ReactNative, Intel XDK, Corona, Xamarin, Marmalade, Adobe Air, dan Web Push telah disediakan oleh OneSignal.

2.13 SQLite

Relational Database Management System (RDBMS) pada sistem operasi Android disebut dengan SQLite (Hendry, 2015). Terdapat kelebihan penggunaan SQLite dalam pengembangan aplikasi Android:

1. Tidak perlu dilakukannya konfigurasi ulang dalam penggunaan SQLite.
2. Bersifat *open source*.
3. Disediakannya fungsi *database* dalam satu set *library*.

2.14 Google App Script

Google Apps Script merupakan *platform* pengembangan aplikasi yang cepat dan mudah dalam pembuatan aplikasi bisnis yang terintegrasi dengan G Suite. Untuk menggunakannya pengembang perlu menulis kode dalam JavaScript dan memiliki akses ke *library* bawaan untuk aplikasi G Suite seperti Gmail, Kalender, Drive, dan lainnya. *Google App Script* berjalan di server Google (Google, 2019).

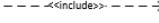
2.15 Use Case Diagram

Diagram yang merepresentasikan *behavior* sistem dan dibuat dengan menggambarkan berbagai aktifitas yang dilakukan oleh aktor dalam sebuah sistem. Di dalam *usecase* ini akan diketahui fungsi apa saja yang terdapat pada sistem, lingkungan sistem, selain itu juga mendeskripsikan kegiatan aktor dan juga relasi aktor pada sistem. Komponen diagram *use case* terdiri dari aktor, *usecase*, asosiasi, *extend*, *include*, dan generalisasi (S dan Shalahuddin, 2018) seperti yang ditunjukkan Tabel 2.1.

Tabel 2.1 Elemen pada *usecase diagram*

Elemen	Notasi	Deskripsi
Aktor	 Actor	Elemen atau komponen aktor digambarkan seperti gambar orang. Namun tidak semua aktor merupakan orang, bisa juga merupakan sistem lain.
UseCase		UseCase merupakan kumpulan fungsi sistem.
Association	———	Asosiasi merupakan penghubung aktor dengan usecase.

Tabel 2.1 Elemen pada *usecase diagram* (lanjutan)

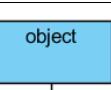
<i>Extend</i>		<i>Extend</i> berelasi dengan <i>usecase</i> yang memaparkan proses lanjutan dari <i>usecase</i> sebelumnya.
<i>Include</i>		Komponen <i>include</i> merupakan relasi <i>usecase</i> yang menginterpretasikan bahwa suatu <i>usecase</i> harus digambarkan dikarenakan terdapat dependensi dari <i>usecase</i> lainnya.
Generalisasi		Merupakan hubungan antara dua buah <i>usecase</i> dimana salah satu fungsi <i>usecase</i> tersebut merupakan fungsi yang lebih umum.

Sumber : Rossa, 2015

2.16 Sequence Diagram

Diagram yang menggambarkan interaksi antar obyek merupakan *sequence diagram*. Diagram ini dibangun untuk mengetahui interaksi setiap obyek di dalam sistem yang menjalankan fungsi kebutuhan yang akan dibangun atau diimplementasi (S and Shalahuddin, 2018). Beberapa komponen yang perlu diketahui dalam pembuatan diagram *sequence* yang ditunjukkan pada Tabel 2.2.

Tabel 2.2 Elemen pada *sequence diagram*

Elemen	Notasi	Deskripsi
Aktor		Elemen atau komponen aktor digambarkan seperti gambar orang. Namun tidak semua aktor merupakan orang, bisa juga merupakan sistem lain. Elemen ini berinteraksi pada sistem yang lain.
Lifeline		<i>Lifeline</i> merupakan kehidupan suatu objek dalam sistem.
Object		<i>Object</i> menyatakan interaksi dengan menggunakan sebuah pesan.
Activation		<i>Activation</i> atau biasa disebut dengan waktu aktif menyatakan obyek dalam keadaan aktif.
Message		Merupakan hubungan yang menyatakan interaksi antar obyek yang saling berhubungan.

Tabel 2.2 Elemen pada *sequence diagram*

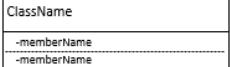
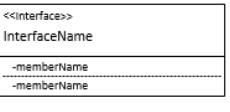
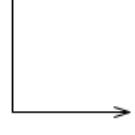
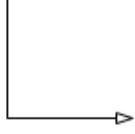
<i>Self message</i>		Hubungan yang dibuat ketika obyek memanggil dirinya sendiri.
<i>Asynchronous Message</i>		Sebuah hubungan yang menyatakan bahwa obyek mengirim data ke dalam obyek yang lain.
<i>Return message</i>		Hubungan yang dibuat ketika obyek telah mengeksekusi suatu operasi dan memberi <i>return</i> pada obyek lain.

Sumber : Rossa, 2015

2.17 Class Diagram

Diagram yang menggambarkan struktur *class-class* di dalam sistem merupakan *class diagram*. Kelas di dalam sistem yang akan dirancang harus berdasarkan kebutuhan yang ada pada sistem (S and Shalahuddin, 2018). Beberapa elemen yang terdapat pada *class diagram* ditunjukkan pada Tabel 2.3.

Tabel 2.3 Elemen pada *class diagram*

Elemen	Notasi	Deskripsi
<i>Class</i>		Salah satu struktur kelas (<i>class</i>) yang berada di dalam sistem.
<i>Interface</i>		Merupakan kelas abstrak dan memiliki definisi yang sama dalam OOP.
<i>Association</i>		Merupakan elemen penghubung antar <i>class</i> .
<i>Directed Association</i>		Merupakan elemen penghubung yang digunakan atau dihubungkan pada beberapa kelas.
<i>Generalisasi</i>		Merupakan elemen penghubung, memiliki makna yang sama dengan generalisasi dan menjelaskan hubungan umum ke khusus.

Tabel 2.3 Elemen pada *class diagram* (lanjutan)

<i>Dependency</i>		Hubungan yang memiliki makna saling ketergantungan. Hubungan ini terdapat pada antar kelas.
<i>Aggregation</i>		Hubungan antar kelas yang bermaksa agregat atau ke semua bagian.

Sumber : Rossa, 2015

2.18 Pengujian Perangkat Lunak

Pengujian perangkat lunak dalam penelitian ini dibagi menjadi 2 (dua) bagian, yakni pengujian unit (*whitebox*) dan pengujian validasi (*blackbox*).

2.18.1 Pengujian Unit

Pengujian unit memfokuskan upaya verifikasi pada unit terkecil dari desain perangkat lunak dan komponen atau modul perangkat lunak. Pengujian ini menggunakan deskripsi desain tingkat komponen sebagai panduan, jalur kontrol penting diuji untuk mengungkap kesalahan dalam batas modul. Kompleksitas relatif dari pengujian dan kesalahan yang tidak terbatasi dibatasi oleh ruang lingkup terbatas yang ditetapkan untuk pengujian unit. Uji unit berorientasi pada *white box* (Sommerville, 2011).

2.18.2 Pengujian Validasi

Sedangkan pengujian validasi dapat didefinisikan dalam banyak cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil ketika fungsi perangkat lunak dengan cara yang dapat diharapkan secara wajar oleh pelanggan (Sommerville, 2011).

2.19 Pengujian *Usability*

Pengujian *usability* dilakukan untuk menilai seberapa mudah dan berguna sebuah sistem atau aplikasi digunakan oleh pengguna (Nielsen, 2012). Tujuan dilakukannya pengujian *usability* adalah untuk mengetahui` bagaimana respon pengguna dalam menggunakan aplikasi. Selain itu juga berkaitan dengan *survival* seorang pengguna dalam menggunakan sebuah sistem. Dengan kata lain, pengguna akan tetap menggunakan aplikasi atau sistem tersebut jika sistem mudah atau memenuhi tujuan dan kebutuhannya. Dan sebaliknya jika sistem atau

aplikasi tersebut sukar digunakan maka pengguna akan meninggalkannya. Penelitian ini berfokus pada *learnability* yang fokus pada seberapa mudah bagi pengguna untuk menyelesaikan tugas-tugas dasar saat pertama kali mereka menggunakan desain aplikasi.

Instrumen yang digunakan dalam pengujian *usability* adalah *System Usability Scale* (SUS). *System Usability Scale* (SUS) merupakan skala yang berisi sepuluh item sederhana yang memberikan pandangan global tentang penilaian subyektif terhadap kegunaan sebuah sistem. SUS adalah skala Likert (Brooke, 2013).

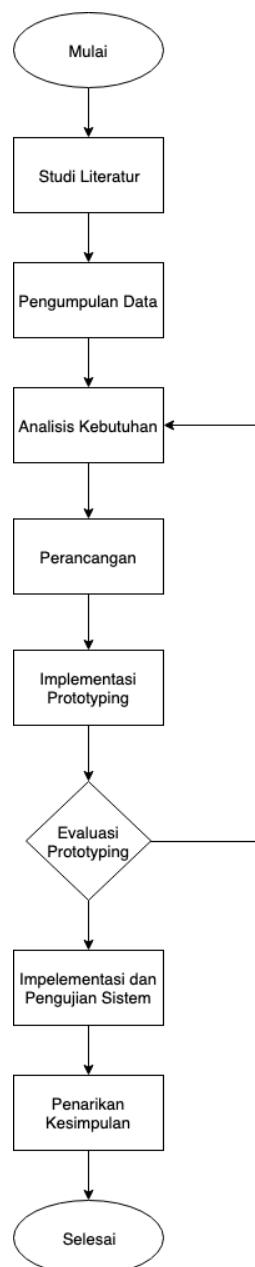
2.20 Pengujian Performance

Pengujian *performance* dilakukan untuk menguji perfomansi kebutuhan non fungsional. Pengujian *performance* dilakukan untuk memvalidasi: kecepatan, kapasitas, skalabilitas, stabilitas, ekspektasi dan *environment* (EPAM, 2008). Pengujian *performance* merupakan proses dimana perangkat lunak diuji untuk menentukan kinerja sistem saat ini. Proses ini bertujuan untuk mengumpulkan informasi tentang kinerja saat sistem saat ini (PerfTestPlus, 2006).

Pengujian *performance* utamanya digunakan untuk menentukan kapasitas sistem yang ada, membuat *benchmark* untuk sistem kedepannya, dan mengevaluasi degradasi dengan berbagai muatan dan / atau konfigurasi. Salah satu referensi mengatakan bahwa dalam sistem peringatan dini, pesan alarm dapat dikirim ke tempat atau kejadian yang kritis di Kota Naples, Italia 20 detik atau lebih sebelum guncangan gempa bumi dimulai (Satriano, Lomax dan Zollo, 2007) sehingga sistem tersebut dapat dikatakan memenuhi kriteria *real time*.

BAB 3 METODOLOGI PENELITIAN

Pada bab ini membahas mengenai metode penelitian yang merupakan langkah-langkah yang akan dikerjakan untuk memecahkan permasalahan atau studi kasus. Tahapan yang terdapat dalam metode penelitian ini adalah studi literatur, analisis sistem, pengumpulan data, perancangan sistem, implementasi, pengujian dan hasil analisa, serta penarikan kesimpulan. Diagram alir penelitian ditunjukkan oleh Gambar 3.1.



Gambar 3.1 Diagram Alir Penelitian

3.1 Studi Literatur

Studi literatur merupakan sebuah tahap dimana terdapat kumpulan teori yang dibutuhkan untuk penelitian ini. Sumber informasi tersebut dapat berasal dari buku, jurnal, laporan penelitian, dan bantuan mesin pencari internet. Studi literatur juga merupakan proses mengumpulkan data-data dari sumber yang terpercaya dan dibuat dengan landasan berfikir teoritis. Studi literatur mempunyai tujuan untuk menegaskan permasalahan serta sebagai landasan teori untuk penelitian lebih lanjut. Teori-teori yang merupakan berkaitan dengan penelitian ini yang berikut terdiri:

1. Bencana alam
2. *Crowdsourcing*
3. Android
4. JSON
5. Android Studio
6. Bahasa pemrograman Kotlin
7. Metode *prototyping*
8. *Push notification*
9. *Geotagging*

3.2 Pengumpulan Data

Tahap pengumpulan data dilaksanakan supaya dapat mendukung penelitian dan dapat memvalidasi kebutuhan sistem. Pada penelitian ini dilakukan proses pengumpulan data dengan metode wawancara. Proses pengumpulan data dengan wawancara langsung yang meliputi beberapa responden yaitu terdiri dari 5 *user* dan 1 *expert* yang pernah mengalami kejadian atau pakar dalam bidang bencana alam. Data yang dijadikan penelitian yaitu jenis bencana alam, pelaporan bencana alam dan *early warning* tempat yang sering terjadi bencana alam. *Interview* yang dilakukan terhadap pengguna menggunakan teknik *open general first, then go deep* oleh IDEO.org (2015). Dimana pertanyaan yang diajukan adalah pertanyaan dari umum ke khusus. Berikut daftar pertanyaan *user interview* yang ditunjukkan oleh Tabel 3.1.

Tabel 3.1 Daftar pertanyaan *user interview*

<i>Open General (What are some broad questions you can ask to open the conversation and warm people up)</i>	<i>Then Go Deep (What are some questions that can help you start to understand this person's hopes, fears, and ambitions?)</i>
Seberapa sering Anda mengalami/merasakan bencana tanah longsor/gempa bumi/banjir/tsunami?	Apa yang Anda rasakan ketika Anda mengalami/melihat tanda-tanda akan terjadi bencana tersebut? Atau adakah tanda-tanda tertentu?
Saat ini, terdapat banyak informasi terkait dengan hal bencana yang telah atau tengah terjadi. Seberapa sering Anda mendapatkan <i>broadcast</i> atau informasi dari sosial media mengenai bencana tersebut?	Bagaimana kondisi sekitar pada saat bencana tersebut terjadi atau akan terjadi? Bisakah Anda menceritakan bagaimana kejadian tersebut?
Dari mana Anda mendapatkan informasi mengenai hal tersebut?	Bagaimana menurut Anda informasi bencana yang disebarluaskan melalui jejaring sosial saat ini?
Informasi apa yang Anda butuhkan ketika sedang berada di musim kemarau atau musim hujan?	Apakah Anda sering berpartisipasi dalam hal penyebarluasan hal tersebut? Bagaimana salah satu bentuk partisipasi tersebut?
Bagaimana peran notifikasi atau pengingat tertentu atau informasi mengenai bencana yang akan terjadi dapat membuat Anda lebih <i>aware</i> terhadap bencana tersebut?	Bagaimana Anda dapat menemukan bahwa informasi terkait dengan kebencanaan tersebut tidak benar? Seberapa sering Anda menemukan hal tersebut terjadi?

Sedangkan untuk pertanyaan *expert interview* dapat dilihat pada Tabel 3.2. Pertanyaan yang akan diajukan pada *expert* adalah mengenai bagaimana idealnya informasi bencana alam yang dapat diakses oleh masyarakat. Selain itu juga peran sosial media dalam menyebarkan informasi mengenai bencana alam.

Tabel 3.2 Daftar pertanyaan *expert interview*

No	Daftar Pertanyaan
1	Bagaimana menurut Bapak/Ibu terkait dengan kejadian bencana alam seperti banjir, tanah longsor, gempa bumi dan tsunami yang saat ini sering terjadi?
2	Bagaimana menurut Bapak/Ibu mengenai informasi kebencanaan yang seringkali disebarluaskan melalui jejaring atau media sosial?

Tabel 3.2 Daftar pertanyaan *expert interview* (lanjutan)

3	Apakah Bapak/Ibu mengetahui mengenai <i>crowdsourcing</i> (penghimpunan data dari berbagai sumber) yang kemudian dimanfaatkan untuk dijadikan informasi yang berguna dan penting?
4	Apakah Bapak/Ibu mengetahui atau pernah menggunakan mengenai aplikasi Info BMKG yang dikembangkan oleh BMKG? Bagaimana peran aplikasi tersebut dalam hal kesiapsiagaan dalam menghadapi bencana alam?
5	Bagaimana <i>early warning</i> dapat bekerja dalam mendukung kesiapsiagaan bencana? Informasi apa yang diperlukan dalam penyebaran informasi <i>early warning</i> (berupa notifikasi) melalui <i>smartphone</i> ?
6	Bagaimana adanya fitur <i>early warning notification</i> penting dalam perancangan dan pembuatan aplikasi ini?
7	Bagaimana menyajikan konten atau informasi yang baik terkait bencana alam gempa bumi dan tsunami? Informasi apa saja yang harus ada di dalamnya?
8	Bagaimana menyajikan konten atau informasi yang baik terkait bencana alam banjir dan tanah longsor? Informasi apa saja yang harus ada di dalamnya?

3.3 Analisis Kebutuhan

Tahap analisis kebutuhan dilakukan bertujuan untuk mendapatkan kebutuhan pengguna yang digunakan untuk mengembangkan perangkat lunak. Daftar kebutuhan tersebut diperoleh dari tahapan pengumpulan data yang dilakukan sebelumnya. Instrumen analisis kebutuhan yang digunakan yaitu *object-oriented analysis* dengan menggunakan UML (*Unified Modelling Language*). Dimana *use case diagram* yang merupakan salah satu jenis diagram UML digunakan untuk mendeskripsikan dan mengidentifikasi kebutuhan serta fungsionalitas yang dibutuhkan oleh pengguna. Aktor yang terlibat dalam *use case diagram* yaitu pengguna yang menggunakan sistem tersebut.

Agar sistem dapat bekerja dengan benar dan memenuhi kebutuhan pengguna, maka penulis mengelompokkan beberapa kebutuhan yang terbagi menjadi dua yaitu kebutuhan fungsional serta non fungsional. Berikut daftar responden *interview* yang ditunjukkan pada Tabel 3.3. Responden *interview* diambil berdasarkan kesamaan latar belakang mereka yakni sama-sama pernah mengalami kejadian kebencanaan.

Tabel 3.3 Daftar responden *interview*

No.	Nama	Profesi	Domisili/Tempat Bencana	Jenis responden	Bencana yang pernah dialami
1	Tripim Apriliyanto	Kepala Bidang Pencegahan dan Kesiapsiagaan, BPBD Malang	-	Expert	-
2	Fajri Fernanda	Mahasiswa Universitas Brawijaya	Kota Malang/Riau	User	Banjir
3	Fathur Rohim	QA Engineer PT. DOT Indonesia	Kota Malang/Bengkulu	User	Banjir
4	Putri Ayu	Mahasiswa Universitas Brawijaya	Kota Malang/Kabupaten Malang	User	Gempa bumi
5	Waroka Beami	Mahasiswa Universitas Brawijaya	Kota Malang/Aceh	User	Tsunami
6	Nida Khoiriyyah	Mahasiswa Universitas Brawijaya	Kota Malang/Balikpapan	User	Banjir

3.4 Perancangan

Tahap perancangan dilakukan setelah tahap analisis kebutuhan selesai dilakukan. Dalam tahap perancangan di dalamnya mengidentifikasi *class–class* apa saja yang dibutuhkan. Dari hasil identifikasi tersebut, kemudian diterjemahkan ke dalam bentuk *class diagram*. Setelah *class diagram* dibuat, tahap selanjutnya mengidentifikasi interaksi antar objek yang kemudian dimodelkan menggunakan *sequence diagram* dengan menggambarkan urutan proses yang dilakukan.

3.5 Implementasi *Prototyping*

Tahap implementasi *prototyping* dilakukan dengan mengacu pada perancangan yang telah dikerjakan pada tahap sebelumnya. Tahap implementasi *prototyping* nantinya akan melalui beberapa proses iterasi dari evaluasi pengguna

atau *client*. Ketika proses iterasi implementasi *prototyping* dilakukan dan sesuai dengan kebutuhan pengguna, maka *prototype* tersebut dijadikan acuan untuk implementasi sistem versi final.

3.6 Implementasi dan Pengujian Sistem

Tahap implementasi dan pengujian sistem dilakukan ketika *evaluasi prototyping* sudah sesuai dengan kebutuhan pengguna. Hal selanjutnya adalah melakukan integrasi data dan fungsionalitas sistem sesuai dengan tahap sebelumnya, yaitu tahapan Analisis dan Perancangan. Pada tahap ini aplikasi sudah siap dijalankan dengan fitur-fitur fungsional di dalamnya. Pengujian dilakukan sebelum aplikasi atau sistem sampai pada *final* atau *release version* ke pengguna. Tujuan dilakukannya pengujian antara lain mengecek apakah fungsionalitas pada sistem sudah berjalan dengan baik dan benar. Metode pengujian yang digunakan pada sistem aplikasi *mobile crowdsourcing* bencana alam yaitu *black box testing*, *white box testing*, *usability testing*, dan *performance testing*.

Usability testing dilakukan untuk menguji seberapa mudah aplikasi digunakan dan bagaimana pengguna dapat berinteraksi dengan mudah pada saat pertama kali menggunakan aplikasi. Atau dengan kata lain *usability testing* dilakukan berdasarkan aspek *learnability* dan *satisfaction*. Instrumen yang digunakan dalam pengujian ini adalah *System Usability Scale* (SUS). Pengujian *usability* dilakukan dengan menjalankan skenario penggunaan aplikasi Kitana yang akan diberikan pada responden uji. Setelah melakukan hal tersebut, maka instrument yang digunakan untuk mengukur *usability* adalah dengan SUS (*System Usability Scale*), dimana responden uji akan mengisi kuisioner tersebut. Tabel 3.4 menunjukkan daftar pertanyaan di dalam kuisioner SUS. Kuisioner SUS memuat 10 (sepuluh) pertanyaan yang menggambarkan bagaimana kesan aplikasi yang telah digunakan (Sauro, 2011).

Tabel 3.4 Instrumen kuisioner SUS

No.	Pertanyaan	Sangat tidak setuju				Sangat Setuju
		1	2	3	4	5
1	Saya merasa bahwa saya akan sering menggunakan aplikasi “Kitana”.					

No.	Pertanyaan	Sangat tidak setuju				Sangat Setuju
		1	2	3	4	5
2	Saya merasa bahwa aplikasi “Kitana” rumit.					
3	Saya merasa bahwa aplikasi “Kitana” mudah digunakan.					
4	Saya merasa saya butuh <i>support</i> atau dukungan dari orang teknis untuk bisa menggunakan aplikasi “Kitana” ini.					
5	Saya merasa bahwa berbagai macam fungsi di aplikasi “Kitana” ini terintegrasi dengan baik.					
6	Saya merasa bahwa terlalu banyak ketidakkonsistenan di dalam aplikasi “Kitana”.					

Tabel 3.4 Instrumen kuisioner SUS (lanjutan)

No.	Pertanyaan	Sangat tidak setuju				Sangat Setuju
		1	2	3	4	5
7	Saya akan membayangkan bahwa sebagian besar orang menggunakan aplikasi “Kitana” ini dengan sangat cepat.					
8	Saya merasa bahwa aplikasi “Kitana” sangat rumit untuk digunakan.					
9	Saya sangat percaya diri menggunakan aplikasi “Kitana” ini.					
10	Saya perlu untuk belajar banyak hal sebelum saya menggunakan aplikasi “Kitana” ini.					

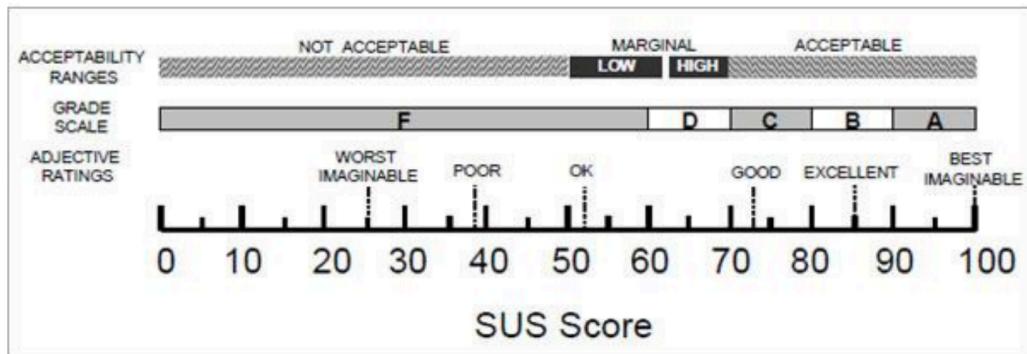
Strongly Disagree 1	2	3	4	Strongly Agree 5
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Gambar 3.2 Format respon kuisioner SUS (Sauro, 2011)

Format respon kuisioner SUS ditunjukkan dalam Gambar 3.2. Langkah selanjutnya adalah menilai hasil kuisioner SUS, yakni hasil isian kusioner SUS diberikan penilaian atau *scoring* dengan ketentuan sebagai berikut:

1. Untuk *item* pertanyaan bermnomor ganjil ditambah nilai +1 dari respon pengguna.

- Untuk *item* pertanyaan bennomor genap dikurangi nilai -1 dari respon pengguna.
- Untuk mengonversi respon pengguna, maka tambahkan semua respon pengguna dan kalikan totalnya dengan 2,5. Konversi ini mengubah rentang nilai menjadi 0 hingga 100.



Gambar 3.3 Grade ranking dari hasil skoring SUS

Grade ranking dari hasil skoring SUS ditunjukkan dalam Gambar 3.3. Atau jika diinterpretasikan dapat dilihat pada Tabel 3.5.

Tabel 3.5 Kualifikasi hasil skor SUS

Skor	Kualifikasi	Hasil
>80	Sangat Baik	Berhasil
68-80	Baik	Berhasil
68	Cukup	Berhasil
51-68	Buruk	Tidak Berhasil
<51	Kurang	Tidak Berhasil

Pengujian *performance* dilakukan untuk mengukur seberapa *real time* performa *push notification* aplikasi diterima oleh pengguna. Pengujian *performance* dilakukan dengan melakukan *debugging* dan *logging* di Android Studio. Pengujian *performance* dilakukan dengan membandingkan waktu sisi *server* dan *client*. Google App Script digunakan sebagai *trigger* untuk mengirim data notifikasi dari sisi *server* ke sisi *client*. Daftar fungsi eksekusi dari sisi pengirim atau *server* dieksekusi dari Google App Script atau *scheduler*.

3.7 Penarikan Kesimpulan

Penarikan Kesimpulan dilaksanakan ketika semua tahap selesai dilakukan. Hasil dari implementasi dan pengujian sistem digunakan untuk pengambilan

kesimpulan terhadap penelitian sistem yang telah dibangun. Dari hasil kesimpulan tersebut peneliti dapat membuat saran untuk memberikan masukan mengenai penelitian, serta untuk perbaikan dan pengembangan lebih lanjut mengenai pembuatan sistem yang lebih baik dan dapat memberikan manfaat untuk kedepannya.

BAB 4 ANALISIS KEBUTUHAN

Pada bab ini membahas mengenai analisis kebutuhan aplikasi *crowdsourcing* dan *early warning* bencana alam berbasis *Android*. Di dalam bab ini terdapat beberapa hal seperti analisis kebutuhan yang terdiri dari identifikasi pengguna, kebutuhan fungsional dan non fungsional serta beberapa *use case diagram* dan *use case scenario*.

4.1 Analisis Kebutuhan

Tahap analisis kebutuhan dilakukan untuk mendapatkan kebutuhan yang sesuai dengan kebutuhan pengguna pada aplikasi *crowdsourcing* dan *early warning* bencana alam. Tahap analisis kebutuhan ini dilakukan dengan cara wawancara ke beberapa pengguna yang representatif. Pengguna yang representative merupakan pengguna yang pernah mengalami salah satu dari bencana alam banjir, tanah longsor, gempa bumi, dan tsunami.

4.1.1 Hasil *interview*

Berdasarkan tahap yang telah dilakukan sebelumnya yakni pengumpulan data dari pengguna dan *expert* maka terdapat beberapa daftar kebutuhan fungsional yang dapat diimplementasi pada aplikasi. hasil *user interview* yang didapatkan dari 5 (lima) pengguna dapat disimpulkan menjadi beberapa poin seperti yang ditunjukkan pada Tabel 4.1. Sebagian dari hasil *interview* dapat dilihat pada Lampiran 2 dan Lampiran 3.

Tabel 4.1 Hasil *user interview*

No.	Hasil <i>user interview</i>
1.	Pengguna biasa mendapatkan informasi kebencanaan dari sosial media seperti Twitter (infoBMKG), WhatsApp, Instagram dan Line.
2.	Sebagian pengguna merasa bahwa informasi kebencanaan yang didapatkan dari sosial media kurang begitu akurat (kecuali informasi resmi yang didapatkan dari akun Twitter infoBMKG).
3.	Belum terdapat pengaturan notifikasi yang terdapat dalam aplikasi infoBMKG sehingga terdapat pengguna yang merasakan hal tersebut sedikit menganggu, karena notifikasi yang diterima beberapa kali dalam satu waktu.
4.	Pengguna perlu membagikan informasi kebencanaan tersebut ke sosial media yang lain.

Tabel 4.1 Hasil *user interview* (lanjutan)

No.	Hasil <i>user interview</i>
5.	Informasi kebencanaan yang tidak terpusat membuat masyarakat yang berada di wilayah-wilayah terpencil tertinggal informasi.
6.	Informasi cuaca yang dibutuhkan pengguna antara lain suhu, <i>weather forecast</i> .
7.	Diperlukannya pengaturan notifikasi aplikasi.

Sedangkan hasil *interview* yang dilakukan terhadap *expert* yang merupakan seseorang yang bertugas di bidang Pencegahan dan Kesiapsiagaan (perencanaan, koordinasi, sosialisasi) adalah bahwasannya informasi seperti *early warning* akan cukup membantu mengingat bencana alam seperti tanah longsot, banjir, pohon tumbang dan puting beliung yang sering terjadi di Malang seringkali pelaporannya dilakukan dengan cara menghubungi langsung pihak BPBD Malang atau melalui WhatsApp *Group* di setiap kelurahan. Di dalam Kelurahan Bareng, Kecamatan Kasin, Malang sebenarnya sudah tersedia EWS atau *Early Warning System* (oleh USAID) untuk mencegah bencana banjir. Mengingat di wilayah tersebut ketika turun hujan dengan curah tinggi dan durasi hujan tersebut melebihi 1 (satu) jam, maka sungai di kelurahan tersebut pasti akan banjir. Informasi semacam *early warning* tentu saja akan sangat membantu. Hal tersebut dapat meningkatkan kewaspadaan masyarakat sekitar terhadap bencana alam yang akan terjadi.

4.1.2 Hasil analisis kebutuhan

Proses analisis kebutuhan dimulai dari proses mengidentifikasi pengguna yang terlibat pada sistem. Selanjutnya menganalisis kebutuhan fungsional yang digambarkan dalam *use case diagram*. Kemudian dilakukan analisis kebutuhan non fungsional untuk mendukung kualitas dari sebuah sistem. Alasan dilakukannya analisis kebutuhan pada aplikasi *crowdsourcing* dan *early warning* bencana alam adalah supaya dapat memenuhi dan sesuai kebutuhan pengguna. Berikut daftar kebutuhan fungsional aplikasi yang ditunjukkan dalam Tabel 4.2.

Tabel 4.2 Daftar kebutuhan fungsional aplikasi Kitana

Kode kebutuhan fungsional	Nama kebutuhan fungsional
F-KT-01	<i>Register</i>

Tabel 4.2 Daftar kebutuhan fungsional aplikasi Kitana (lanjutan)

F-KT-02	<i>Login</i>
F-KT-03	Melihat informasi <i>posting</i> bencana
F-KT-04	Membuat <i>posting</i> bencana alam
F-KT-05	Melihat notifikasi <i>early warning</i>
F-KT-06	Melihat informasi cuaca
F-KT-07	Melihat informasi cuaca maritim
F-KT-08	Mengubah profil
F-KT-09	<i>Upload</i> foto profil
F-KT-10	Mengubah <i>password</i> akun
F-KT-11	<i>Logout</i>
F-KT-12	<i>Filter posting</i>
F-KT-13	<i>Upvote posting</i>
F-KT-14	<i>Downvote posting</i>
F-KT-15	Melihat detail informasi <i>posting</i> bencana

4.1.3 Gambaran Umum

Aplikasi yang dikembangkan pada penelitian ini yaitu aplikasi *crowdsourcing* dan *early warning* bencana alam. Nama lain dari aplikasi ini yakni Kitana (**K**ita **T**Ahu **benc****NA**). Aplikasi Kitana merupakan aplikasi yang bertujuan untuk berbagi informasi mengenai bencana alam dengan menggunakan konsep *social crowdsourcing* dan *early warning*. Konsep *crowdsourcing* pada aplikasi ini melibatkan semua elemen masyarakat untuk berbagi informasi tentang pelaporan kejadian bencana alam di wilayah sekitarnya. Di samping itu pengguna lain juga akan menerima *push notification* jika terjadi kejadian bencana alam yang telah dilaporkan oleh pihak lain atau dari data pihak ketiga (Data API BMKG). Bencana alam yang ter-cover dalam aplikasi ini adalah banjir, gempa bumi, tanah longsor, dan tsunami. Konsep *crowdsourcing* tersebut diterapkan pengguna dengan melaporkan bencana alam yang sedang terjadi. Pelaporan tersebut berisi detil seperti nama kejadian, deskripsi, kategori bencana, lokasi dan bukti gambar atau foto.

4.2 Identifikasi Aktor

Pada tahap ini dilakukan identifikasi aktor yang bertujuan untuk mengidentifikasi aktor yang akan berinteraksi langsung dengan aplikasi Kitana. Hasil identifikasi aktor dipaparkan dalam Tabel 4.3.

Tabel 4.3 Identifikasi Aktor

Aktor	Deskripsi
Pengguna	Pengguna yaitu masyarakat umum yang menggunakan aplikasi untuk mengirimkan pelaporan bencana alam.
<i>Guest</i>	<i>Guest</i> yaitu pengunjung yang menggunakan aplikasi yang hanya dapat mengakses <i>register</i> , melihat <i>postingan</i> bencana, melihat informasi cuaca, melihat informasi cuaca maritim, melihat notifikasi <i>early warning</i> .

4.3 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan utama yang harus dipenuhi oleh sistem supaya aplikasi dapat berjalan dengan baik dan dapat digunakan oleh pengguna. Hasil kebutuhan fungsional didapatkan dari proses analisis kebutuhan. Tabel analisis kebutuhan fungsional ditunjukkan pada Tabel 4.4, dan Tabel 4.5.

Tabel 4.4 Analisis kebutuhan fungsional *Guest*

Kode Fungsional	Deskripsi	Use case
F-KT-01	Sistem dapat menyediakan fungsional <i>register</i> .	<i>Register</i>
F-KT-02	Sistem dapat menyediakan fungsionalitas <i>login</i> .	<i>Login</i>
F-KT-03	Sistem dapat menyediakan tampilan yang berisi <i>postingan</i> bencana dari pengguna.	Melihat informasi bencana alam
F-KT-05	Sistem dapat menyediakan fungsionalitas untuk melihat notifikasi <i>early warning</i> bencana alam.	Melihat notifikasi <i>early warning</i>
F-KT-06	Sistem dapat menyediakan tampilan yang berisi informasi cuaca meliputi suhu, <i>weather forecast</i> .	Melihat informasi cuaca

Tabel 4.4 Analisis kebutuhan fungsional Guest (lanjutan)

F-KT-07	Sistem dapat menyediakan tampilan yang berisi informasi cuaca maritim meliputi informasi keadaan ombak (waktu, jam, tinggi dan tipe ombak).	Melihat informasi cuaca maritim
F-KT-14	Sistem dapat menyediakan fungsionalitas <i>filter posting</i> bencana.	<i>Filter posting</i>
F-KT-15	Sistem dapat menyediakan tampilan yang berisi detail <i>postingan</i> bencana dari pengguna	Melihat detail <i>posting</i> bencana

Tabel 4.5 menunjukkan kebutuhan fungsional untuk aktor Pengguna. Dimana di dalamnya terdapat deskripsi kebutuhan seperti menambah *posting*, mengubah *profil*, mengubah *password* akun dan sebagainya.

Tabel 4.5 Analisis kebutuhan fungsional Pengguna

Kode Fungsional	Deskripsi	Use case
F-KT-04	Sistem dapat menyediakan fungsionalitas membuat <i>posting</i> bencana alam dengan <i>field</i> judul, kategori, deskripsi, gambar dan lokasi bencana.	Membuat <i>posting</i> bencana alam
F-KT-08	Sistem dapat menyediakan fungsionalitas edit profil dengan <i>field</i> nama lengkap, emai, kota dan provinsi.	Mengubah profil
F-KT-09	Sistem dapat menyediakan fungsionalitas <i>upload</i> foto profil pengguna.	<i>Upload</i> foto profil
F-KT-10	Sistem dapat menyediakan fungsionalitas mengubah <i>password</i> akun pengguna.	Mengubah <i>password</i> akun
F-KT-11	Sistem dapat menyediakan fungsionalitas <i>logout</i> yang dilakukan oleh pengguna.	<i>Logout</i>
F-KT-12	Sistem dapat menyediakan fungsionalitas <i>upvote posting</i> yang dilakukan oleh pengguna.	<i>Upvote posting</i>
F-KT-13	Sistem dapat menyediakan fungsionalitas <i>downvote posting</i> yang dilakukan oleh pengguna.	<i>Downvote posting</i>

4.4 Kebutuhan Non Fungsional

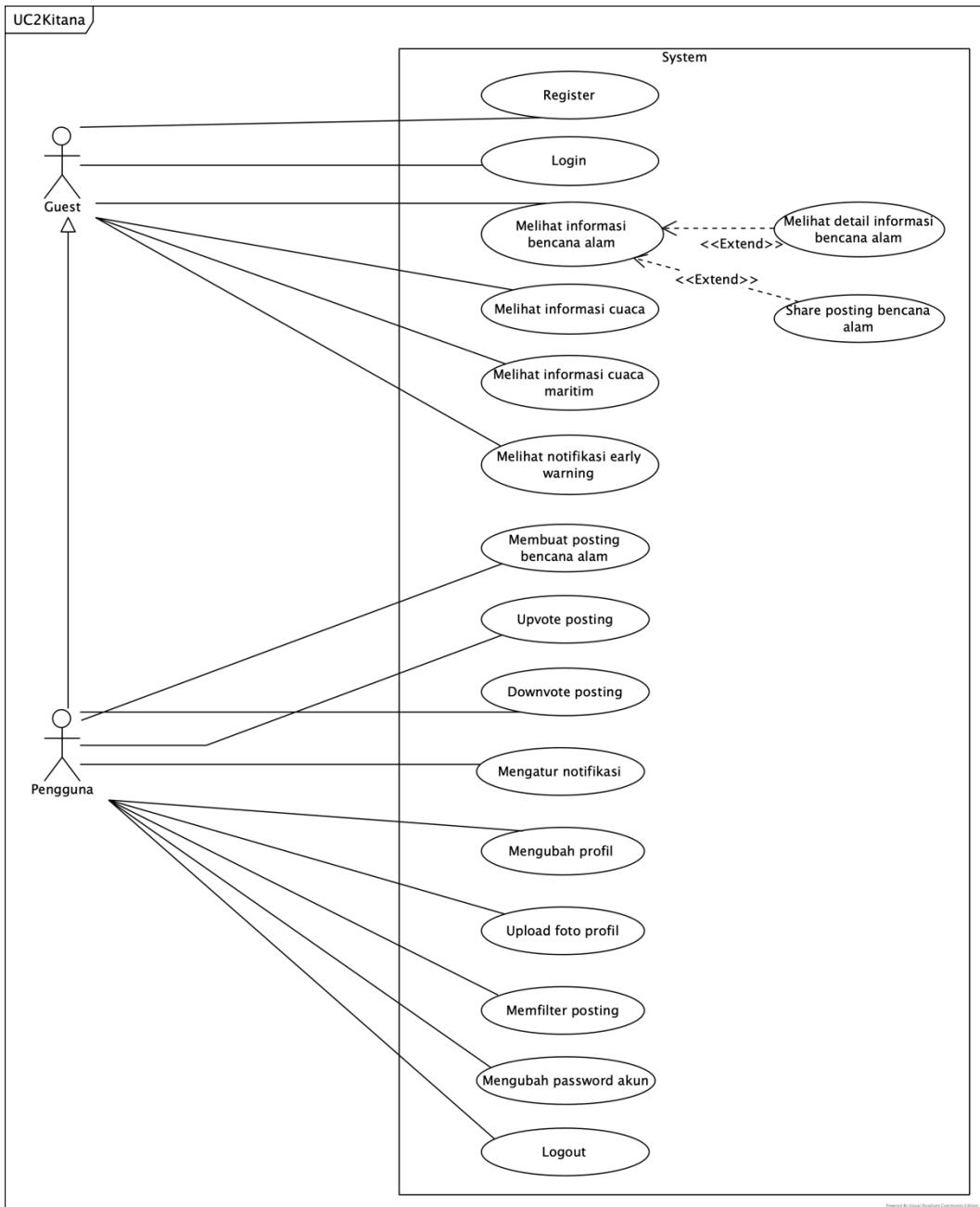
Kebutuhan Non Fungsional merupakan kebutuhan yang tidak harus dipenuhi tetapi dibutuhkan oleh pengguna yang bertujuan untuk mendukung kualitas dari aplikasi. Daftar kebutuhan non fungsional ditunjukkan pada Tabel 4.6.

Tabel 4.6 Daftar kebutuhan non fungsional

Kode Kebutuhan	Nama	Deskripsi
NF-KT-1	<i>Performance</i>	Aplikasi dapat mengirimkan <i>push notifikasi postingan bencana dengan real time</i> ke pengguna lain.
NF-KT-2	<i>Usability</i>	Aplikasi dapat digunakan dengan mudah dan berguna oleh pengguna.

4.5 Pemodelan *Use case Diagram*

Pemodelan *use case diagram* ditunjukkan pada Gambar 4.1 dimana di dalam *use case diagram* tersebut terdapat 2 (dua) aktor yang meliputi *Guest*, dan Pengguna. Aktor *Guest* yang merupakan generalisasi dari aktor Pengguna, dimana *Guest* dapat melakukan *login*, *register*, melihat informasi bencana alam, melihat informasi cuaca dan maritim, melihat notifikasi *early warning*. Aktor kedua, yakni Pengguna yang dapat melakukan membuat *posting* bencana alam, *upvote posting*, *downvote posting*, mengatur notifikasi, mengubah profil, *upload* foto profil, dan *mem-filter posting*. Pada pemodelan *use case* ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah.



Gambar 4.1 Use case diagram

4.5.1 Pemodelan *Use case Scenario*

Pemodelan *use case scenario* dibuat setelah perancangan *use case diagram* selesai. Pemodelan *use case scenario* dilakukan untuk menjelaskan detail proses dari setiap *use case*.

4.5.1.1 *Use case scenario Login*

Pada Tabel 4.7 terdapat *use case scenario* kebutuhan fungsional *Login*. Dimana terdapat *Guest* sebagai aktor. Tindakan yang perlu dilakukan aktor *guest* adalah

mengisikan *field email* dan *password* di *form Login* dan menekan tombol *login* untuk masuk kedalam sistem atau aplikasi. Selanjutnya setelah berhasil aplikasi akan menampilkan halaman Beranda.

Tabel 4.7 Use case Scenario Login

Item	Deskripsi
Kode	F-KT-02
Nama <i>Use case</i>	<i>Login</i>
Aktor	<i>Guest</i>
Deskripsi	<i>Use case login</i> menjelaskan bagaimana aktor dapat melakukan <i>login</i> dengan menggunakan akun yang terdaftar
Pra-kondisi	Aplikasi menampilkan halaman <i>login</i> dan mengisi <i>field username</i> dan <i>password</i>
Tindakan	<ol style="list-style-type: none"> 1. Pengguna mengisi <i>field email</i> dan <i>password</i> di <i>form login</i> 2. Pengguna Tekan tombol <i>login</i>
Post-kondisi	Aplikasi menampilkan halaman Beranda
Alternatif	<ol style="list-style-type: none"> 1. <i>Email</i> dan <i>Password</i> yang dimasukkan salah

4.5.1.2 Use case scenario Register

Tabel 4.8 menunjukkan maritim dari *use case Register*. Aktor dalam *use case* ini adalah *Guest*. Pra-kondisi *use case* ini adalah aplikasi akan menampilkan halaman *Register* lengkap dengan *field* seperti *email*, nama lengkap, kabupaten/kota, provinsi, *password*, dan ketik ulang *password*. Tindakan yang aktor perlu lakukan adalah mengisi *field* dalam *form Register* dan menekan tombol *Register*.

Tabel 4.8 Use case Scenario Register

Item	Deskripsi
Kode	F-KT-01
Nama <i>Use case</i>	<i>Register</i>
Aktor	<i>Guest</i>

Tabel 4.8 Use case Scenario Register (lanjutan)

Item	Deskripsi
Deskripsi	<i>Use case register</i> menjelaskan bagaimana aktor dapat melakukan <i>register</i> dengan mengisi form <i>register</i>
Pra-kondisi	Aplikasi menampilkan halaman <i>register</i> dan mengisi <i>field email</i> , nama lengkap, kabupaten/kota, provinsi, <i>password</i> , ulang <i>password</i>
Tindakan	<ol style="list-style-type: none">1. Pengguna mengisi <i>field email</i>, nama lengkap, kabupaten/kota, provinsi, <i>password</i>, ulang <i>password</i> di form <i>register</i>2. Pengguna Tekan tombol <i>register</i>
Post-kondisi	Aplikasi menampilkan halaman <i>login</i>
Alternatif	<ol style="list-style-type: none">1. Field form <i>register</i> tidak diisi semua2. <i>Password</i> dan ulang <i>password</i> tidak sama

4.5.1.3 Use case scenario Melihat informasi bencana alam

Pada Tabel 4.9 terdapat *use case scenario* melihat informasi bencana alam. Aktor yang terlibat dalam *use case* ini adalah Pengguna. Pengguna dapat menuju halaman atau menu Beranda untuk melihat informasi bencana alam. *Post-kondisi* yang dimunculkan oleh aplikasi adalah ditampilkannya halaman Beranda.

Tabel 4.9 Use case Scenario Melihat informasi bencana alam

Item	Deskripsi
Kode	F-KT-03
Nama <i>Use case</i>	Melihat informasi bencana alam
Aktor	<i>Guest</i>
Deskripsi	<i>Use case</i> melihat informasi bencana alam menjelaskan bagaimana aktor melihat informasi bencana alam dari pengguna lain
Pra-kondisi	Menuju halaman beranda

Tabel 4.9 Use case Scenario Melihat informasi bencana alam (lanjutan)

Tindakan	1. Pengguna membuka aplikasi menuju menu beranda
Post-kondisi	Aplikasi menampilkan halaman beranda
Alternatif	Jika aplikasi dijalankan tidak tersedia koneksi internet, maka sistem menampilkan informasi “Tidak dapat tersambung. Periksa koneksi internet anda.”

4.5.1.4 Use case scenario Membuat posting bencana alam

Pada Tabel 4.10 menunjukkan *use case scenario* tambah *posting* yang dilakukan oleh aktor Pengguna. Dimana terdapat pra kondisi yakni Pengguna melakukan *login* terlebih dahulu dan menekan tombol *plus* (+).

Tabel 4.10 Use case Scenario Membuat posting bencana alam

Item	Deskripsi
Kode	F-KT-04
Nama <i>Use case</i>	Membuat <i>posting</i> bencana alam
Aktor	Pengguna
Deskripsi	<i>Use case</i> membuat <i>posting</i> bencana alam menjelaskan bagaimana aktor dapat melakukan tambah <i>posting</i> mengenai bencana alam.
Pra-kondisi	<i>Login</i> dan menekan tombol <i>plus</i> .
Tindakan	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan tombol <i>plus</i> di bagian tengah navigasi bawah3. Mengisi <i>form</i> tambah <i>posting</i> dengan melengkapi <i>field</i> judul, kategori, deskripsi, gambar, dan lokasi <i>posting</i>.4. Menekan tombol “Post”
Post-kondisi	<i>Posting</i> bencana alam pengguna tampil di menu Beranda
Alternatif flow	<ol style="list-style-type: none">1. Jika Aktor tidak mengisi <i>form</i> tambah <i>posting</i> lalu tekan tombol <i>post</i> maka menampilkan informasi pesan <i>field</i> harus diisi.

4.5.1.5 Melihat notifikasi *early warning*

Tabel 4.11 menunjukkan *use case scenario* pada kebutuhan fungsional melihat notifikasi *early warning*. Aktor yang terlibat dalam scenario ini adalah Pengguna dan *Guest*. Pengguna dan *Guest* menuju halaman Notifikasi untuk melihat notifikasi *early warning*. Post-kondisi yang diterima adalah informasi *early warning* dapat dilihat oleh pengguna. Sumber data notifikasi *early warning* telah berasal dari *crawling* BMKG yang merupakan hasil *request scheduler* setiap satu menit sekali. Setelah proses tersebut, data notifikasi disimpan SQLite *mobile* Android dan setelah itu pengguna memuat ulang data tersebut di dalam SQLite.

Tabel 4.11 Use case Scenario Melihat notifikasi *early warning*

Item	Deskripsi
Kode	F-KT-05
Nama <i>Use case</i>	Melihat notifikasi <i>early warning</i>
Aktor	<i>Guest</i>
Deskripsi	<i>Use case</i> melihat notifikasi bencana menjelaskan bagaimana aktor dapat melihat notifikasi <i>early warning</i> bencana alam melalui menu tertentu yang telah disediakan.
Pra-kondisi	Notifikasi <i>early warning</i> telah diterima dan Menuju halaman Notifikasi
Tindakan	1. Pengguna Menekan <i>icon</i> Notifikasi untuk melihat notifikasi
Post-kondisi	Informasi <i>early warning</i> dapat dilihat oleh pengguna.
Alternatif	Jika data <i>early warning</i> kosong maka menampilkan pesan informasi "Informasi Early Warning tidak tersedia"

4.5.1.6 Mengakses informasi cuaca

Tabel 4.12 menunjukkan *use case scenario* melihat informasi cuaca. Aktor yang terlibat dalam informasi cuaca ini adalah Pengguna dan *Guest*. Tindakan yang perlu dilakukan oleh aktor adalah dengan menekan menu Cuaca sehingga aplikasi dapat menampilkan informasi cuaca.

Tabel 4.12 Use case Scenario Melihat informasi cuaca

Item	Deskripsi
Kode	F-KT-06
Nama <i>Use case</i>	Melihat informasi cuaca
Aktor	<i>Guest</i>
Deskripsi	<i>Use case</i> melihat informasi cuaca menjelaskan bagaimana aktor melihat kondisi cuaca berupa suhu saat ini dan beberapa hari kedepan, dan kota.
Pra-kondisi	Menuju halaman informasi cuaca
Tindakan	1. Pengguna menekan navigasi menu cuaca
Post-kondisi	Aplikasi menampilkan halaman informasi cuaca
Alternatif	Jika aplikasi dijalankan tidak tersedia koneksi internet, maka sistem menampilkan informasi "Tidak dapat tersambung. Periksa koneksi internet anda."

4.5.1.7 Mengakses informasi cuaca maritim

Tabel 4.13 menunjukkan *use case scenario* melihat informasi cuaca maritim. Aktor yang terlibat dalam maritim ini adalah Pengguna dan *Guest*. Tindakan yang perlu dilakukan oleh aktor adalah dengan menekan menu Cuaca sehingga aplikasi dapat menampilkan informasi cuaca maritim.

Tabel 4.13 Use case Scenario Melihat informasi cuaca maritim

Item	Deskripsi
Kode	F-KT-07
Nama <i>Use case</i>	Melihat informasi cuaca maritim
Aktor	Pengguna
Deskripsi	<i>Use case</i> melihat informasi cuaca maritim menjelaskan bagaimana aktor melihat informasi keadaan ombak seperti waktu, jam, tinggi, dan tipe ombak.
Pra-kondisi	Menuju halaman informasi cuaca

Tabel 4.13 Use case Scenario Melihat informasi cuaca maritime (lanjutan)

Tindakan	<ol style="list-style-type: none">1. Pengguna menekan navigasi menu cuaca2. Pengguna menekan <i>tab menu Cuaca Maritim</i>3. Pengguna memilih lokasi maritim pada <i>maps</i>4. Pengguna menekan tombol “Pilih lokasi perairan”
Post-kondisi	Aplikasi menampilkan halaman informasi cuaca maritim
Alternatif	Jika aplikasi dijalankan tidak tersedia koneksi internet, maka sistem menampilkan pesan “Tidak dapat tersambung. Periksa koneksi internet anda.”

4.5.1.8 Mengubah profil

Tabel 4.14 menunjukkan *use case scenario* mengubah profil. Aktor yang terlibat dalam *use case scenario* mengubah profil ini adalah Pengguna. Tindakan yang perlu dilakukan aktor pengguna adalah mengubah *field* di *form setting* profil dan menekan tombol Simpan Profil untuk mengubah data profil. Selanjutnya setelah berhasil aplikasi akan menampilkan pesan “Anda telah berhasil *update* profil”.

Tabel 4.14 Use case Scenario Mengubah profil

Item	Deskripsi
Kode	F-KT-08
Nama Use Case	Mengubah Profil
Aktor	Pengguna
Deskripsi	<i>Use case</i> mengedit profil <i>menjelaskan</i> bagaimana aktor bekerja melakukan mengubah informasi akun pengguna yang meliputi nama, email, alamat, kota dan password.
Pra-kondisi	Login, menuju menu <i>setting</i>

Tabel 4.14 Use case Scenario Mengubah profil (lanjutan)

Tindakan	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna mengubah nama, email, alamat pada form ubah profil3. Pengguna menekan tombol "Simpan profil"
Post-kondisi	Aplikasi melakukan <i>update</i> informasi profil pengguna.
Alternatif	-

4.5.1.9 Upload foto profil

Tabel 4.15 menunjukkan *use case scenario Upload foto profil*. Aktor yang terlibat dalam *use case scenario* mengubah profil ini adalah Pengguna. Pengguna dapat menuju menu *Setting* untuk mengubah foto profilnya. Tindakan yang perlu dilakukan aktor pengguna adalah mengambil foto profil melalui kamera atau *gallery*. Selanjutnya setelah berhasil aplikasi akan memperbarui foto profil pengguna.

Tabel 4.15 Use case Scenario Upload foto profil

Item	Deskripsi
Kode	F-KT-09
Nama Use Case	<i>Upload foto profil</i>
Aktor	Pengguna
Deskripsi	<i>Use case upload foto profil menjelaskan bagaimana aktor bekerja melakukan upload foto profil di menu setting.</i>
Pra-kondisi	Login, menuju menu <i>setting</i>
Tindakan	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna tekan <i>icon photo profil</i>3. Pengguna <i>upload photo profil</i>

Tabel 4.15 Use case Scenario Upload foto profil (lanjutan)

Post-kondisi	Aplikasi melakukan <i>update photo profil</i> pengguna.
Alternatif	-

4.5.1.10 Mengubah Password akun

Tabel 4.16 menunjukkan *use case scenario* mengubah *password* akun. Aktor yang terlibat dalam *use case scenario* mengubah *password* akun ini adalah Pengguna. Tindakan yang perlu dilakukan aktor Pengguna adalah mengisi *field password* baru dan ulang *password* baru pada *form ubah password* akun. Selanjutnya adalah menekan tombol *Ubah Password* untuk mengubah data *password* akun. Selanjutnya setelah berhasil aplikasi akan menampilkan pesan “Anda telah berhasil ubah *password*”.

Tabel 4.16 Use case Scenario Mengubah Password Akun

Item	Deskripsi
Kode	F-KT-10
Nama Use Case	Mengubah <i>Password akun</i>
Aktor	Pengguna
Deskripsi	<i>Use case</i> Mengubah <i>password</i> akun menjelaskan bagaimana pengguna mengubah <i>password</i> dengan mengisi <i>form ubah password</i> .
Pra-kondisi	Login, menuju halaman Setting
Tindakan	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna mengubah <i>password</i> baru pada <i>form ubah password</i>3. Pengguna menekan tombol “<i>Ubah password</i>”
Post-kondisi	Aplikasi melakukan <i>update password</i> baru pengguna
Alternatif	Jika <i>field password</i> dan ulang <i>password</i> baru tidak sama maka menampilkan pesan “ <i>Password</i> dan ulang <i>password</i> tidak sama”

4.5.1.11 Logout

Tabel 4.17 menunjukkan *use case scenario Logout*. Aktor yang terlibat dalam *use case scenario logout* ini adalah Pengguna. Tindakan yang perlu dilakukan aktor pengguna adalah menuju halaman Setting dan menekan *icon logout*. Selanjutnya setelah berhasil aplikasi akan menampilkan pesan “Anda telah berhasil *logout*”.

Tabel 4.17 Use case Scenario Logout

Item	Deskripsi
Kode	F-KT-11
Nama Use Case	<i>Logout</i>
Aktor	Pengguna
Deskripsi	<i>Use Case Logout</i> menjelaskan bagaimana aktor melakukan <i>logout</i> di aplikasi.
Pra-kondisi	Login, menuju menu <i>setting</i>
Tindakan	<ul style="list-style-type: none">- Pengguna menekan <i>tab menu Setting</i>- Pengguna menekan <i>icon logout</i>
Post-kondisi	Pengguna telah <i>logout</i> dan menuju halaman <i>login</i> .
Alternatif	-

4.5.1.12 Upvote posting

Tabel 4.18 menunjukkan *use case scenario Upvote posting*. Aktor yang terlibat dalam *use case scenario upvote posting* ini adalah Pengguna. Tindakan yang perlu dilakukan oleh pengguna adalah menekan *icon upvote* dari salah satu *posting* bencana yang dibuat oleh pengguna. Selanjutnya setelah berhasil melakukan *upvote* maka aplikasi akan menampilkan jumlah *upvote posting* bencana.

Tabel 4.18 Use case Scenario Upvote Posting

Item	Deskripsi
Kode	F-KT-12
Nama Use Case	<i>Upvote posting</i>
Aktor	Pengguna
Deskripsi	<i>Use case upvote posting</i> bencana menjelaskan bagaimana aktor dapat mendukung (<i>upvote</i>) postingan bencana alam yang dibuat oleh pengguna.
Pra-kondisi	Login, menuju halaman Beranda
Tindakan	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan <i>icon Beranda</i> untuk masuk ke menu Beranda3. Menekan <i>icon upvote</i> untuk mendukung posting bencana alam
Post-kondisi	Posting bencana alam yang telah di- <i>upvote</i> akan menambah jumlah <i>upvote posting</i> .
Alternatif	-

4.5.1.13 Downvote posting

Tabel 4.19 menunjukkan *use case scenario Downvote posting*. Aktor yang terlibat dalam *use case scenario* ini adalah Pengguna. Tindakan yang perlu dilakukan oleh pengguna adalah menekan *icon downvote* dari salah satu *posting* bencana yang dibuat pengguna. Selanjutnya setelah berhasil aplikasi akan menampilkan jumlah *downvote posting* bencana.

Tabel 4.19 Use case Scenario Downvote Posting

Item	Deskripsi
Kode	F-KT-13
Nama Use Case	<i>Downvote posting</i>
Aktor	Pengguna
Deskripsi	<i>Use case downvote posting</i> bencana menjelaskan bagaimana aktor untuk tidak mendukung (<i>downvote</i>) postingan bencana alam yang dibuat oleh pengguna.
Pra-kondisi	Login, menuju halaman Beranda

Tabel 4.19 Use case Scenario Downvote Posting (lanjutan)

Tindakan	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan <i>icon Beranda</i> untuk masuk ke menu Beranda3. Menekan <i>icon downvote</i> untuk tidak mendukung posting bencana alam
Post-kondisi	Posting bencana alam yang telah di- <i>downvote</i> akan menambah jumlah <i>downvote</i> posting.
Alternatif	-

4.5.1.14 Filter posting

Tabel 4.20 menunjukkan *use case scenario Filter posting*. Aktor yang terlibat dalam *use case scenario logout* ini adalah Pengguna. Tindakan yang perlu dilakukan adalah menekan *dropdown filter posting* lalu memilih jenis *filter* dan menakan tombol “Terapkan *filter*”. Selanjutnya setelah berhasil aplikasi akan menampilkan halaman informasi bencana alam berdasarkan hasil *filter posting*.

Tabel 4.20 Use case Scenario Filter Posting

Item	Deskripsi
Kode	F-KT-14
Nama Use Case	<i>Filter Posting</i>
Aktor	Pengguna
Deskripsi	<i>Use Case Filter Posting</i> menjelaskan bagaimana aktor melihat informasi <i>posting</i> bencana dengan melakukan <i>filter</i> berdasarkan waktu, <i>upvote</i> , <i>downvote</i> , gempa bumi, banjir, tsunami dan tanah longsor.
Pra-kondisi	Menuju halaman beranda
Tindakan	<ol style="list-style-type: none">1. Pengguna menekan <i>dropdown filter posting</i>2. Pengguna memilih jenis <i>filter posting</i>3. Pengguna menekan tombol “Terapkan <i>filter</i>”
Post-kondisi	Aplikasi menampilkan informasi bencana alam berdasarkan <i>filter posting</i> .
Alternatif	-

4.5.1.15 Melihat detail informasi *posting* bencana

Pada Tabel 4.21 terdapat *use case scenario* melihat detail informasi *posting* bencana. Aktor yang terlibat dalam *use case* ini adalah *Guest* atau Pengguna. Tindakan yang dapat dilakukan aktor adalah menuju halaman Beranda, kemudian memilih salah satu dari daftar *posting* bencana. *Post-kondisi* yang dimunculkan oleh aplikasi adalah ditampilkannya halaman detail informasi *posting* bencana.

Tabel 4.21 Use case scenario Melihat detail informasi *posting* bencana

Item	Deskripsi
Kode	F-KT-15
Nama Use Case	Melihat detail <i>posting</i> bencana
Aktor	Guest
Deskripsi	<i>Use case</i> melihat detail <i>posting</i> bencana menjelaskan bagaimana aktor melihat detail <i>posting</i> bencana dari pengguna.
Pra-kondisi	Menuju halaman Beranda
Tindakan	<ol style="list-style-type: none">1. Pengguna menuju menu Beranda2. Pengguna menekan salah satu <i>cardview</i> pada daftar <i>posting</i> bencana alam
Post-kondisi	Aplikasi dapat melihat detail <i>posting</i> ke sosial media
Alternatif	-

4.6 Analisis Data

Penelitian ini membutuhkan beberapa data yang digunakan untuk pengembangan aplikasi Kitana. Data tersebut seperti data pengguna, bencana, *vote*, *early warning* gempa dari *server*, dan *early warning* gempa dari *client*.

1. Data pengguna digunakan untuk melakukan *login*, *register* dan *setting* akun. *Email*, *password*, nama lengkap, kota, provinsi, dan foto yang merupakan atribut data pengguna.
2. Data bencana digunakan untuk melakukan *posting* dan melihat informasi bencana alam. Judul, kategori, deskripsi, gambar, *latitude*, *longitude*, dan tanggal yang merupakan atribut data bencana.

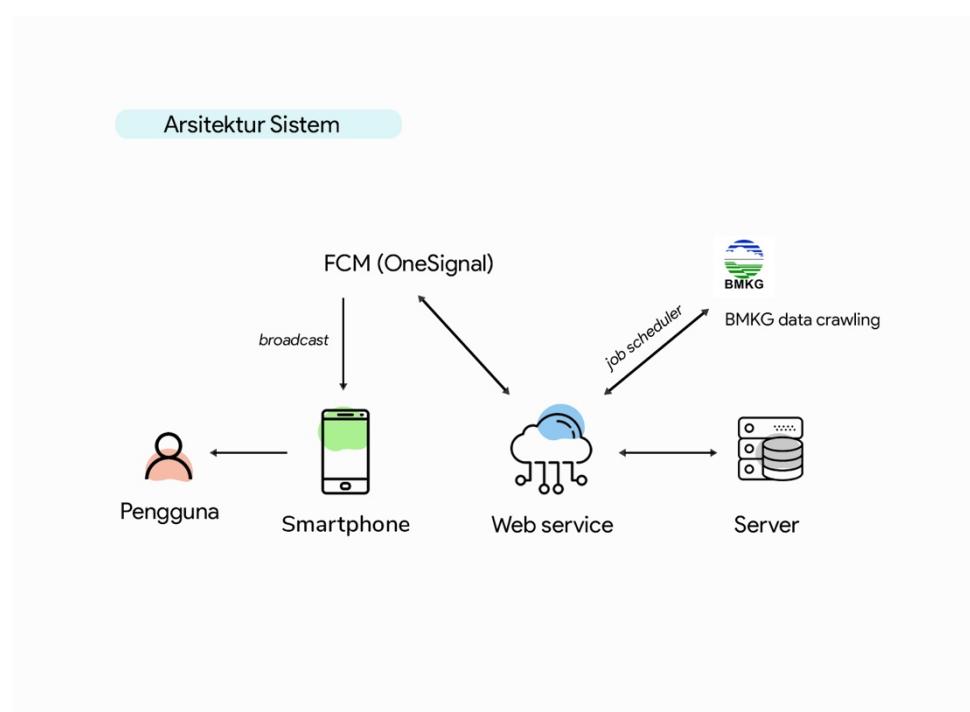
3. Data *vote* digunakan untuk melakukan *upvote* (fungsi untuk mendukung *posting* bencana) dan *downvote* (fungsi untuk tidak mendukung *posting* bencana). *Upvote* dan *downvote* yang merupakan atribut data *vote*.
4. Data *early warning* gempa dari *server* digunakan untuk menyimpan data gempa bumi ke dalam *database*. Lalu dilakukan pengecekan dsri *scheduler* Google App Script, jika ada data yang baru maka *push notification* dikirimkan. Jika tidak ada data baru maka tidak dikirimkan *push notification*. Tanggal, koordinat, posisi, *magnitude*, kedalaman dan keterangan yang merupakan atribut data *early warning* gempa dari *server*.
5. Data *early warning* gempa dari *client* digunakan untuk menyimpan data gempa bumi ke dalam *local storage* (SQLite) perangkat *mobile* di pengguna. *Date*, *title*, dan deksripsi yang merupakan atribut data *early warning* gempa dari *client*.

BAB 5 PERANCANGAN

Pada bab ini membahas mengenai perancangan dari aplikasi *crowdsourcing* dan *early warning* bencana alam berbasis *Android*. Perancangan tersebut meliputi perancangan arsitektur sistem, *activity diagram*, *sequence diagram*, *class diagram*, perancangan basis data, *Physical Data Model* (PDM), perancangan API, perancangan antarmuka pengguna dan perancangan algoritma.

5.1 Perancangan Arsitektur Sistem

Perancangan arsitektur sistem dapat digambarkan seperti pada Gambar 5.1. Pengguna mengakses aplikasi *crowdsourcing* dan *early warning* bencana alam melalui perangkat bergerak.



Gambar 5.1 Rancangan Arsitektur Sistem

Pada proses rancangan arsitektur sistem menggambarkan bahwa terdapat dua proses yaitu *crowdsourcing* dan *early warning*. Untuk proses *early warning* pertama kali dilakukan oleh *web service* yang telah di *deploy* ke *server* serta melakukan *set timer scheduler* dengan *request* ke *BMKG API* untuk mengecek apakah terdapat data terbaru. Setelah itu jika terdapat data terbaru maka melakukan *crawling* dan disimpan ke dalam *web service*, data telah disimpan lalu *web service* melakukan pemanggilan ke *firebase cloud messaging* untuk melakukan *push notification* secara *real time* kejadian bencana alam dari

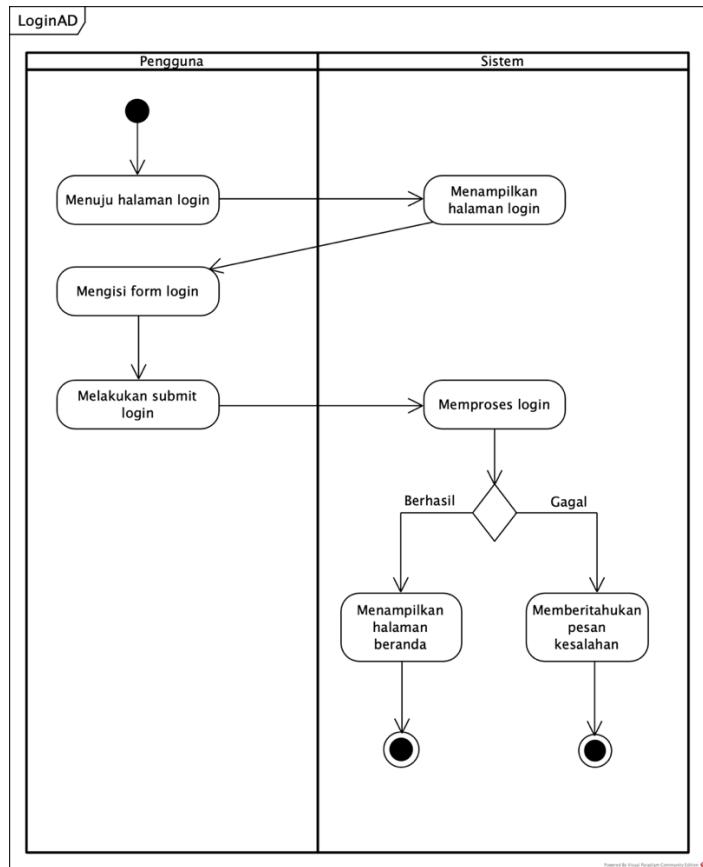
pengguna lain, setelah itu akun pengguna telah terdaftar di aplikasi crowdsourcing bencana alam akan menerima notifikasi ke semua perangkat bergerak.

5.2 Perancangan *Activity Diagram*

Pada sub bab poin *Activity Diagram* ini membahas tentang alur kerja dari *use case* yang telah dijelaskan sebelumnya. Pada bagian perancangan ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah yang dapat dilihat pada Tabel 5.8.

5.2.1.1 *Activity diagram login*

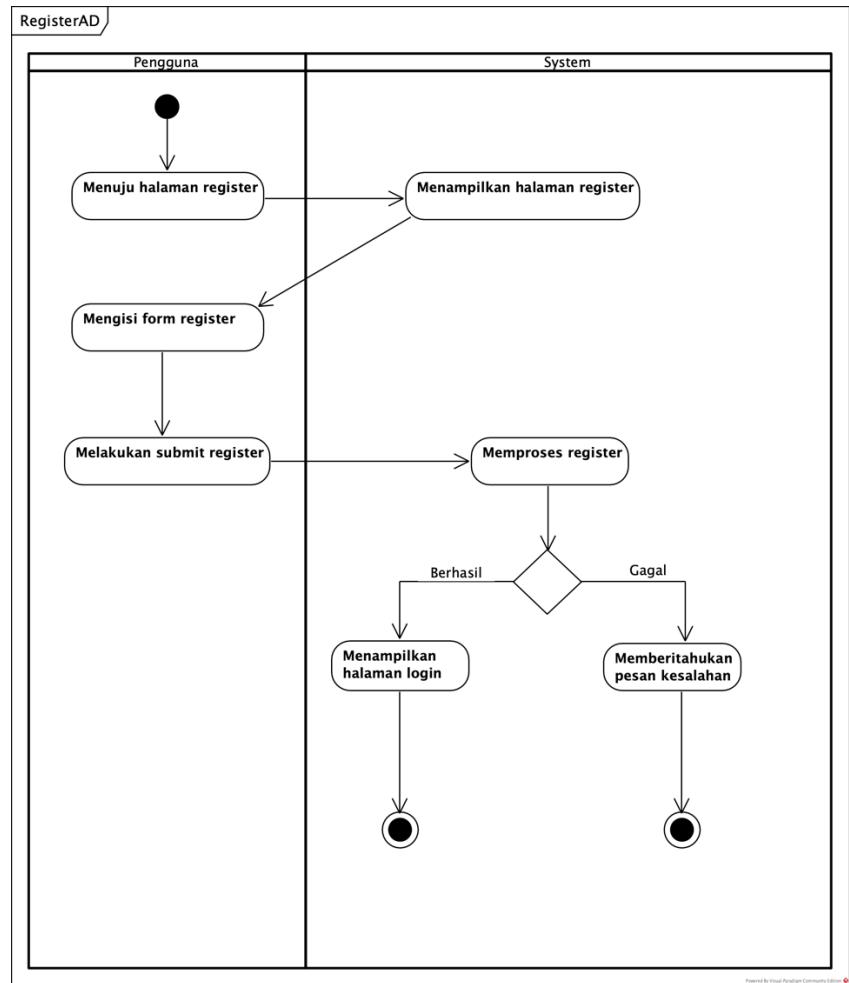
Pada Gambar 5.2 yang merupakan gambar *activity diagram Login* berisi alur *login* dari kedua aktor yakni Pengguna dan Sistem. Alur dari diagram tersebut adalah Pengguna menuju halaman *login*, setelah itu Sistem menampilkan halaman *login*, selanjutnya Pengguna mengisi *form login* dan melakukan *submit form login*. Selanjutnya sistem memproses *login*, jika berhasil maka sistem akan menampilkan halaman Beranda, sedangkan jika gagal maka sistem akan memberitahukan pesan kesalahan.



Gambar 5.2 Activity diagram Login

5.2.1.2 Activity diagram register

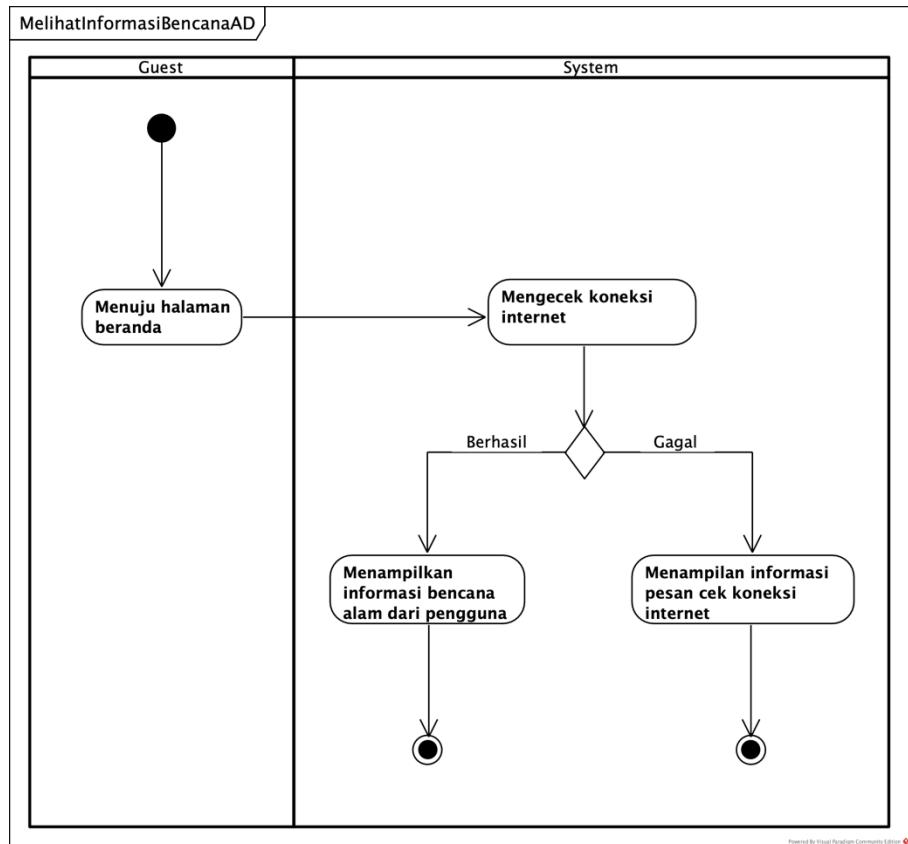
Pada Gambar 5.3 terdapat *activity diagram register* yang berisi alur Pengguna melakukan *register* dan sistem merespon kegiatan pengguna. Aktifitas register dimulai dari Pengguna yang menuju halaman *register*, selanjutnya sistem menampilkan halaman *register*. Selanjutnya Pengguna mengisi *form register* dan melakukan submit *form register*. Setelah itu sistem memproses *form* hasil submit Pengguna, jika berhasil maka sistem menampilkan halaman *login*, dan jika gagal sistem akan memberitahukan pesan kesalahan pada Pengguna.



Gambar 5.3 Activity diagram Register

5.2.1.3 Activity diagram Melihat informasi bencana alam

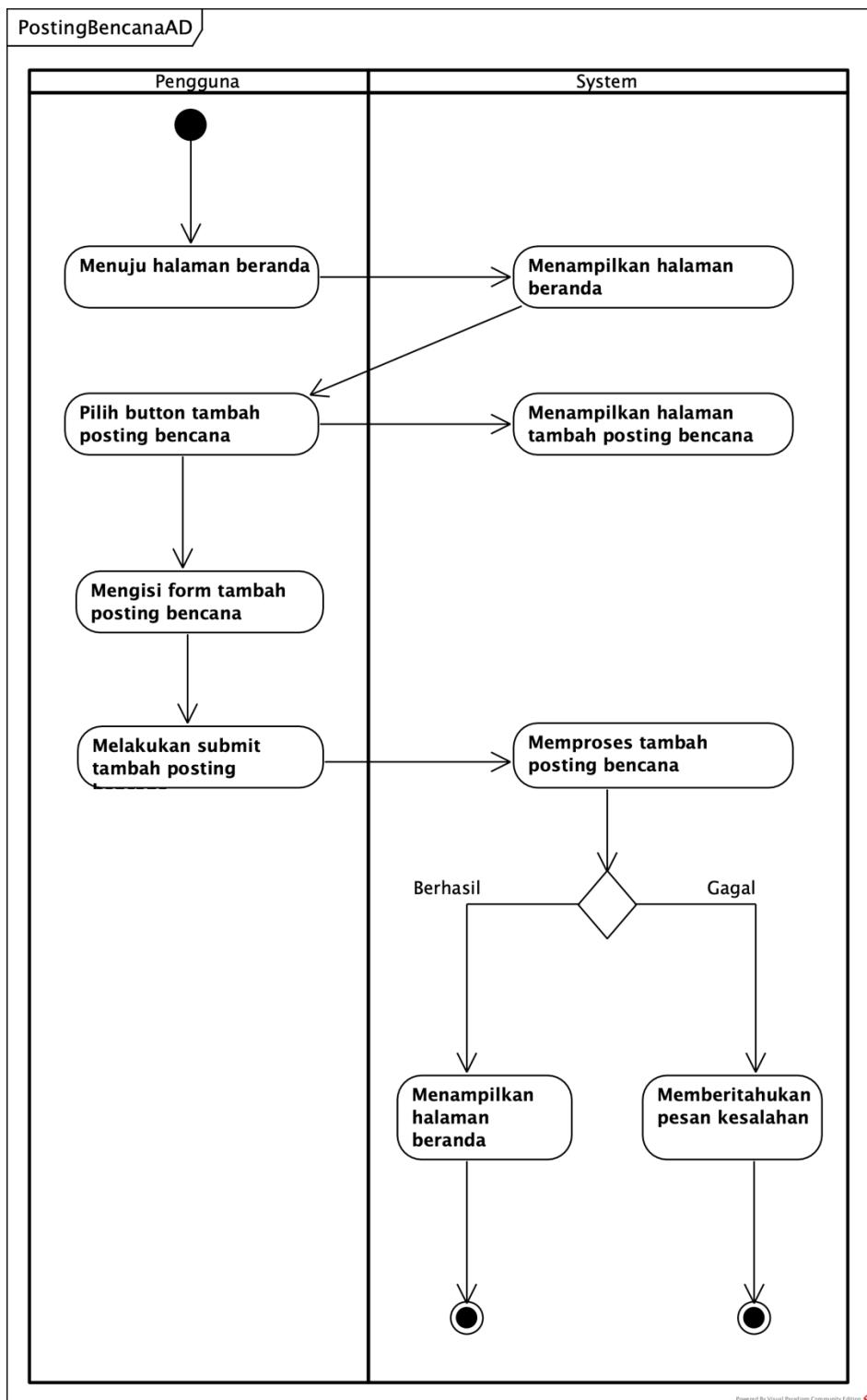
Di dalam Gambar 5.4 terdapat *activity diagram* dimana di dalamnya terdapat Guest dan Sistem. Aktifitas diawali dengan Guest yang menuju halaman Beranda, selanjutnya Sistem akan mengecek koneksi internet. Jika berhasil maka Sistem akan menampilkan informasi bencana alam dari pengguna (*posting*), dan jika gagal maka Sistem akan menampilkan informasi pesan “Cek koneksi internet”.



Gambar 5.4 Activity diagram Melihat Informasi Bencana alam

5.2.1.4 Activity diagram membuat *posting* bencana alam

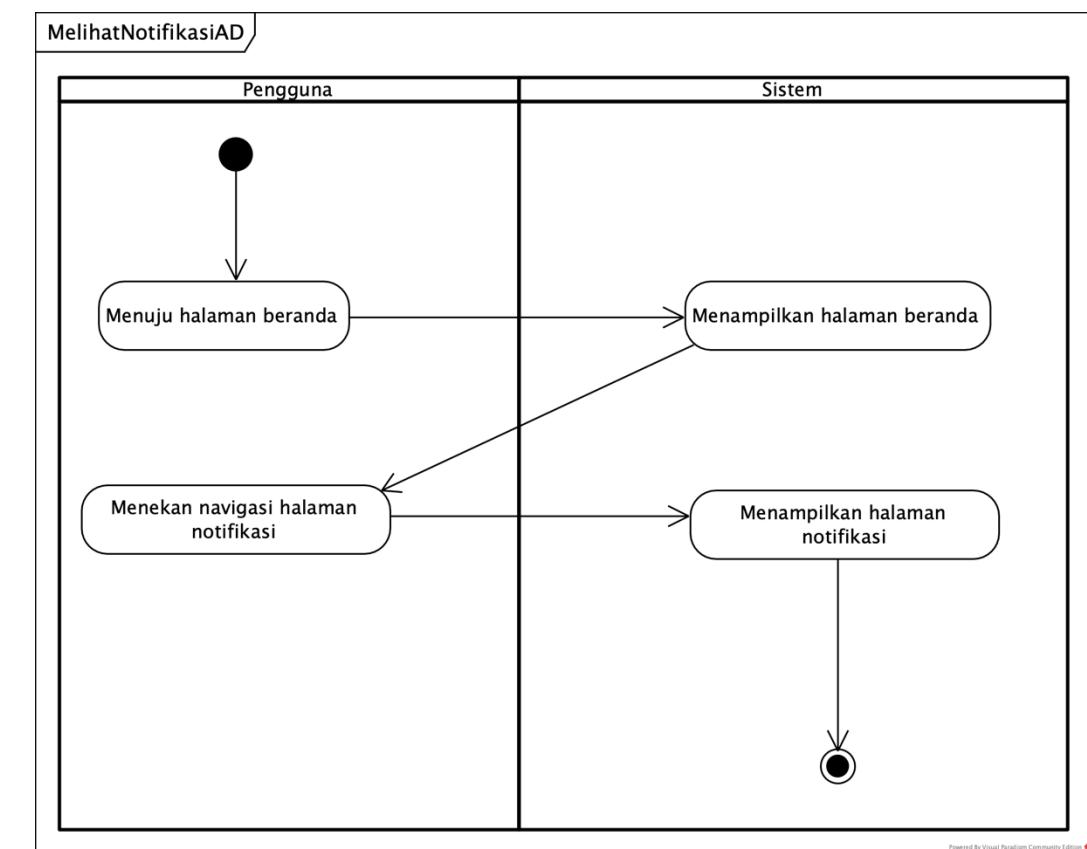
Dalam Gambar 5.5 terdapat *activity diagram* yang berisi bagaimana aktifitas Pengguna membuat *posting* bencana alam dan bagaimana Sistem merespon aktifitas Pengguna. Aktifitas pertama kali dilakukan oleh Pengguna yang *login* ke sistem. Selanjutnya sistem akan menampilkan halaman beranda ke Pengguna, dan di dalam Beranda terdapat *button* tambah *posting* bencana. Pengguna menekan *button* tersebut dan sistem akan menampilkan halaman tambah *posting* bencana yang berisi *form* *posting* bencana. Selanjutnya pengguna mengisi *form tambah posting* bencana dan melakukan *submit* *form* *posting* bencana. Kemudian Sistem akan memproses *form* tersebut, jika berhasil maka sistem akan menampilkan halaman beranda, jika gagal sistem akan memberitahukan pesan kesalahan.



Gambar 5.5 *Activity diagram* Membuat *Posting* bencana alam

5.2.1.5 *Activity diagram* melihat notifikasi *early warning*

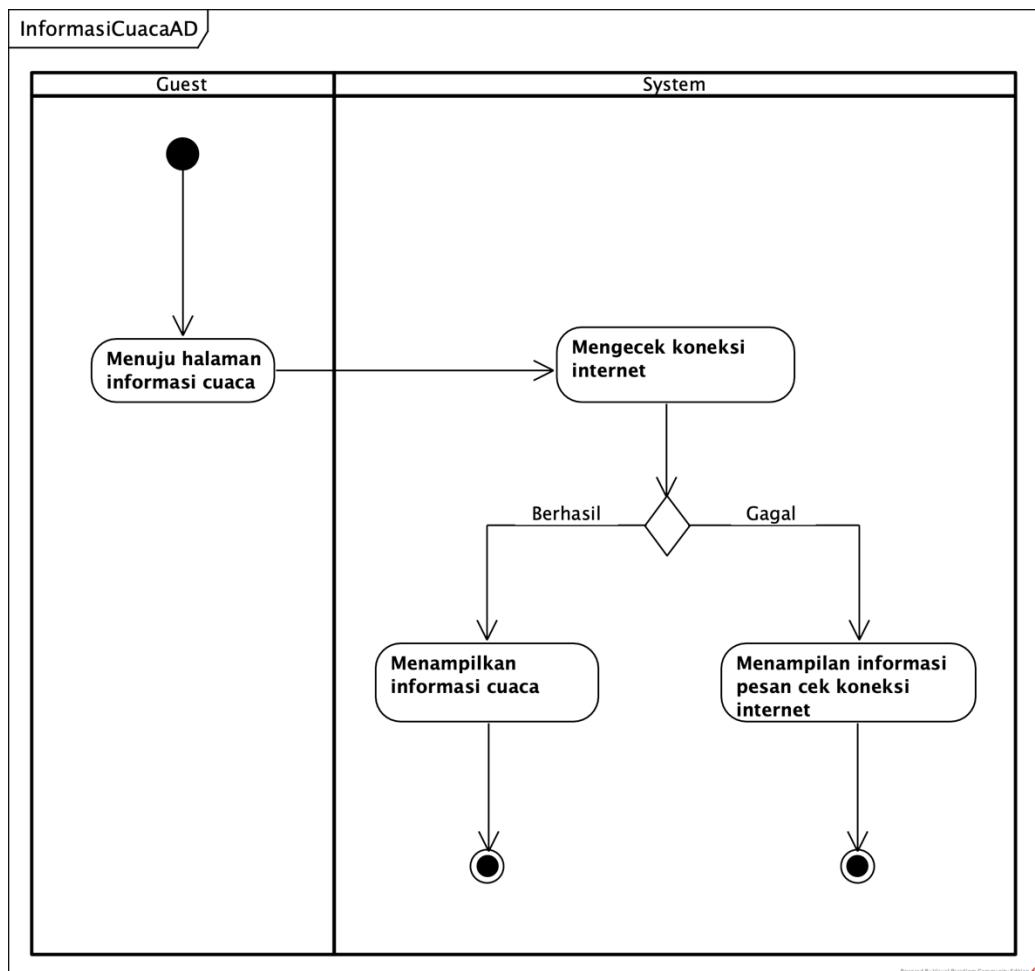
Pada Gambar 5.6 terdapat *activity diagram* mengatur notifikasi dimana Di dalamnya terdapat aktifitas yang dilakukan oleh Pengguna dan Sistem. Aktifitas pertama dilakukan oleh Pengguna, menuju halaman Beranda. Selanjutnya sistem menampilkan halaman Beranda pada Pengguna. Setelah itu Pengguna menekan halaman Notifikasi. Pengguna dapat melihat notifikasi *early warning* bencana alam yang terdapat dalam menu atau halaman tersebut. Selain itu sumber data notifikasi *early warning* sebelumnya telah berasal dari *crawling* BMKG yang merupakan hasil *request scheduler* setiap satu menit sekali, selanjutnya data notifikasi disimpan SQLite *mobile* Android dan setelah itu pengguna memuat ulang data tersebut di dalam SQLite.



Gambar 5.6 Activity diagram Melihat notifikasi *early warning*

5.2.1.6 Activity diagram melihat informasi cuaca

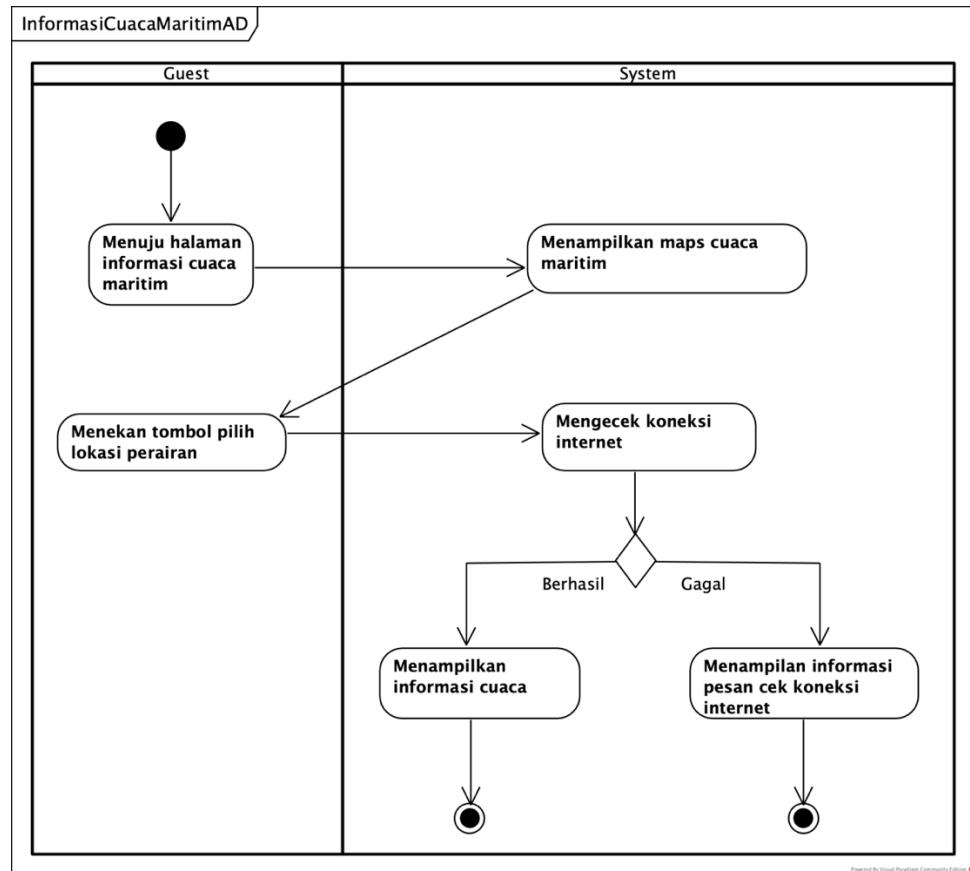
Pada Gambar 5.7 terdapat *activity diagram* melihat informasi cuaca dimana terdapat aktifitas dari 3 (tiga) actor yakni *Guest*, Pengguna dan Sistem. *Guest* menuju halaman informasi cuaca, pengguna melewati aktifitas *login*, selanjutnya menuju halaman informasi cuaca. Setelah itu respon dari sistem adalah dengan menampilkan halaman informasi cuaca.



Gambar 5.7 Activity diagram Melihat informasi cuaca

5.2.1.7 Activity diagram melihat informasi cuaca maritim

Pada Gambar 5.8 terdapat *activity diagram* melihat informasi cuaca maritim dimana terdapat aktifitas dari 3 (tiga) aktor yakni *Guest*, Pengguna dan Sistem. *Guest* menuju halaman informasi cuaca maritim, pengguna melewati aktifitas *login*, selanjutnya menuju halaman informasi cuaca maritim. Setelah itu respon dari sistem adalah dengan menampilkan halaman informasi cuaca maritim.



Gambar 5.8 Activity diagram Melihat informasi cuaca maritim

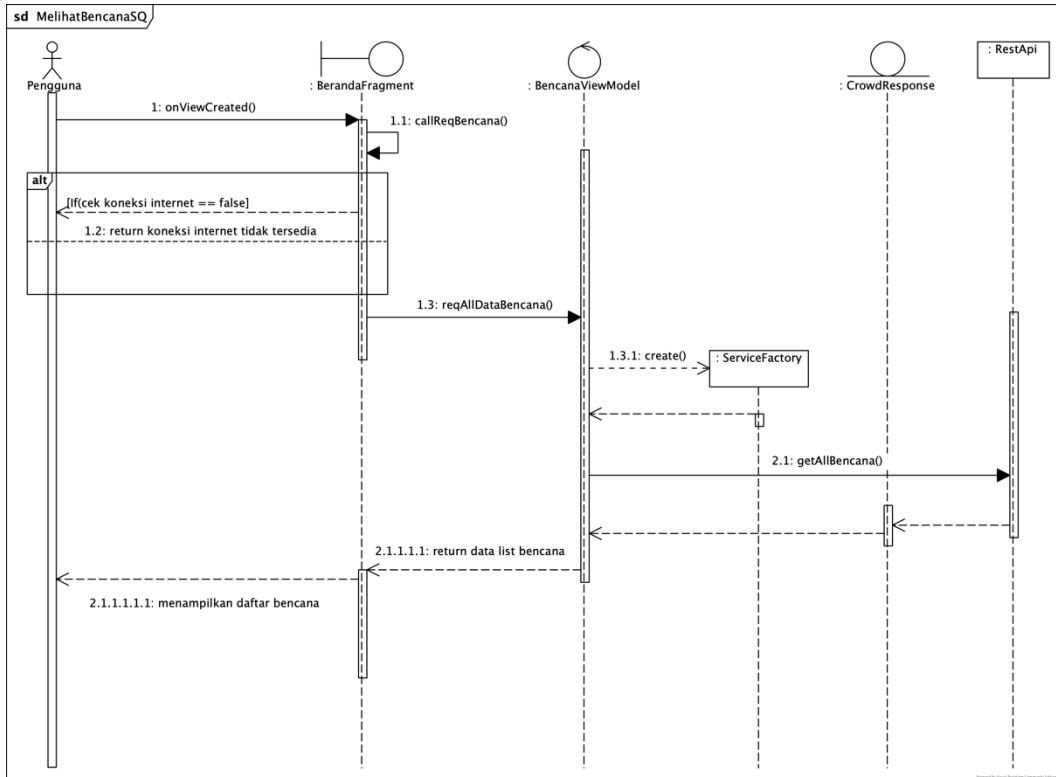
5.2.2 Perancangan *Sequence Diagram*

Pada bagian perancangan *sequence diagram* berisi gambaran interaksi sistem yang memodelkan sebuah pengeksekusian skenario antara pengguna dengan sistem. *Sequence diagram* yang digambarkan tidak secara keseluruhan ditampilkan, hanya beberapa fitur utama atau *major* yang akan dijelaskan di sub bab selanjutnya. Pada bagian perancangan ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah yang dapat dilihat pada Tabel 5.8.

5.2.2.1 *Sequence diagram* melihat informasi bencana alam

Gambar 5.9 menunjukkan *sequence diagram* melihat informasi bencana alam. Awalnya pengguna membuka aplikasi menuju kelas `BerandaFragment` dengan memanggil fungsi `onViewCreated()`, Setelah itu memanggil *method* di dalamnya *method* `callReqBencana()`. Kemudian di cek apakah koneksi tersedia, jika tidak tersedia `BencanaFragment` me-return koneksi internet tidak tersedia. Jika tersedia koneksi internet, selanjutnya *control* `BencanaViewModel` menjalankan *method* `reqAllBencana()` dan

ServiceFactory me-return objek yang dibuat. Selanjutnya method getAllBencana() dijalankan dan me-return objek yang dibuat ke entitas CrowdResponse. Setelah itu control BencanaViewModel me-return data list bencana, boundary BerandaFragment menampilkan daftar bencana pada pengguna.

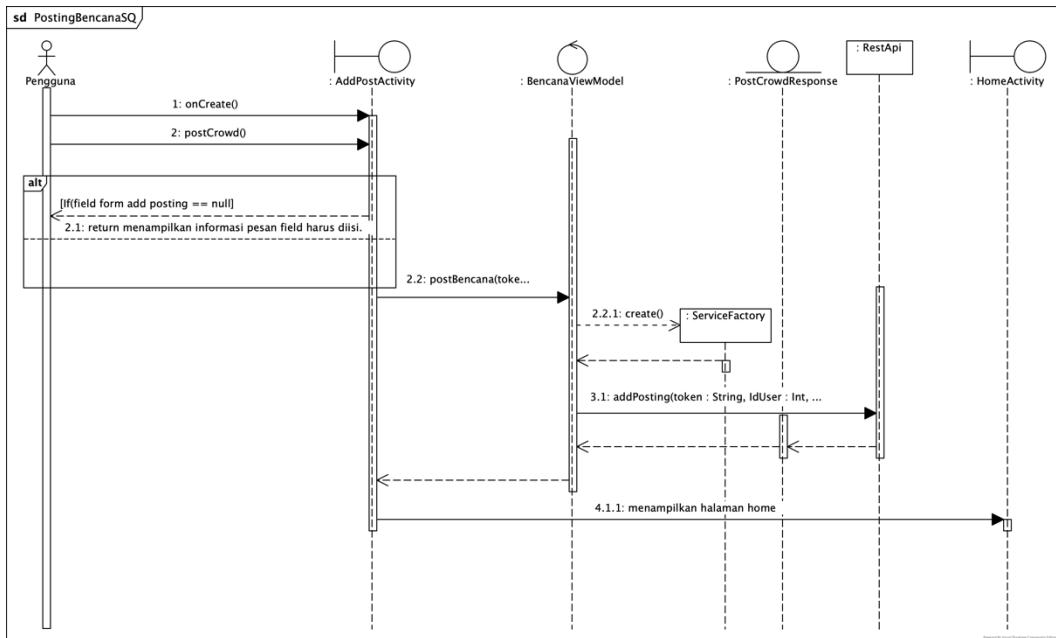


Gambar 5.9 Sequence diagram melihat informasi bencana alam

5.2.2.2 Sequence diagram membuat posting bencana alam

Gambar 5.10 menunjukkan gambar sequence diagram membuat *posting* bencana alam dimana pengguna membuka aplikasi dan mengisi *field* yang terdapat dalam *form* *posting* bencana dengan memanggil method *postCrowd()*. Selanjutnya sistem mengecek apakah ada *field* *form* *posting* bencana. Jika ada yang kosong maka AddPostingActivity me-return menampilkan informasi pesan *field* harus di isi. Jika *field* *form* *posting* bencana tidak ada yang kosong, Setelah itu control BencanaViewModel menjalankan method *postBencana()* dengan parameter hasil isian *form* *posting*, seperti token, idUser, judul, kategori, deskripsi, lokasi, *latitude*, *longitude*, tanggal dan *file image* bencana. Selanjutnya dibuat objek di ServiceFactory dan di-return ke control BencanaViewModel. Setelah itu RestAPI menjalankan method *addPosting()* dengan parameter seperti pada gambar, mengembalikan nilai *return* ke entitas PostCrowdReponse, control BencanaViewModel, dan

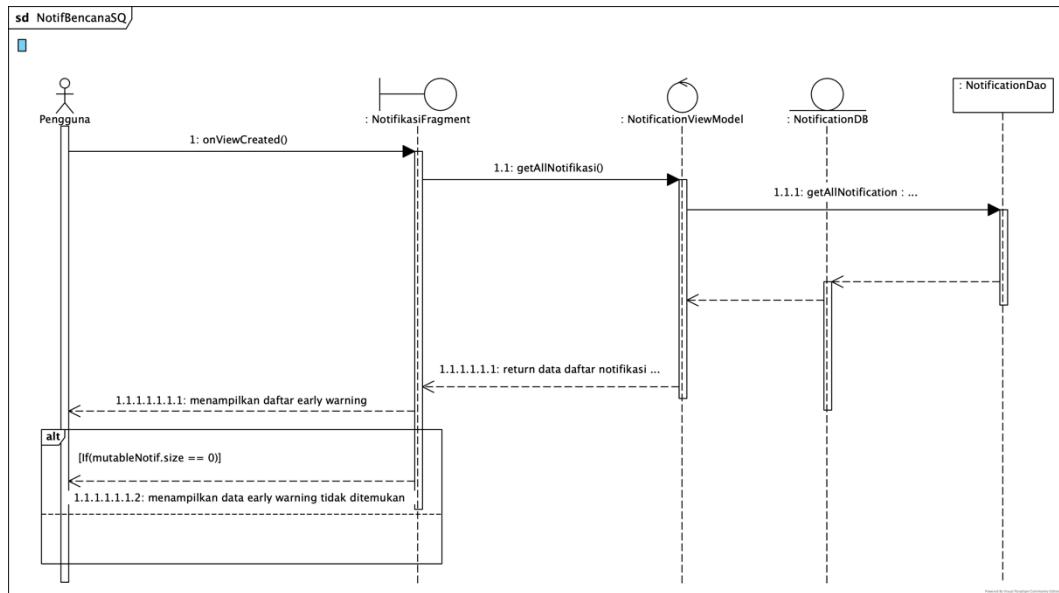
boundary AddPostActivity. Setelah itu *boundary* HomeActivity menampilkan halaman Home.



Gambar 5.10 Sequence diagram membuat posting bencana alam

5.2.2.3 Sequence diagram melihat notifikasi early warning

Gambar 5.8 menunjukkan sequence diagram melihat notifikasi *early warning*. Pengguna membuka aplikasi, dan kelas NotifikasiFragment dijalankan dengan memanggil fungsi onViewCreated(). Selanjutnya *control* NotificationViewModel menjalankan *method* getAllNotifikasi() tanpa adanya parameter. Setelah itu melakukan pemanggilan *variable* getAllNotification untuk melakukan *query* ke *local storage (SQLite)*. Selanjutnya *variable* getAllNotifikasi mengatur nilai ke dalam kelas *entity* NotificationDb. Setelah itu *control* NotificationViewModel me-return data *list early warning* bencana. *Boundary* NotifikasiFragment melakukan pengecekan ukuran pada *variable* mutableNotif, Jika *variable* mutableNotif sama dengan 0 maka menampilkan informasi pesan Data Early Warning tidak ditemukan. Jika *variable* mutableNotif lebih dari 0, *boundary* NotifikasiFragment menampilkan daftar notifikasi *early warning* bencana.



Gambar 5.11 Sequence diagram melihat notifikasi early warning

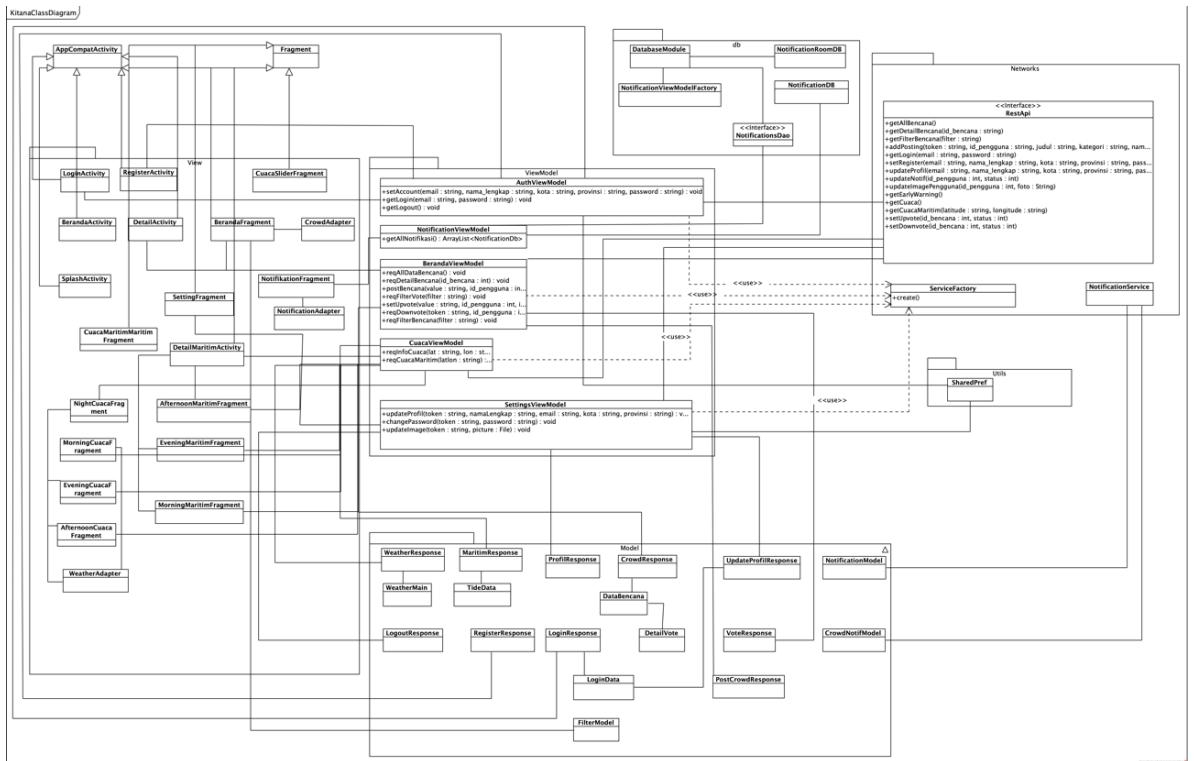
5.2.3 Perancangan Class Diagram

Pada bagian ini akan dijelaskan mengenai *class diagram* yang dijadikan struktur dalam pembentukan kelas pengembangan aplikasi bencana alam. Di dalam kelas terdapat daftar *method* beserta parameternya yang akan digunakan dalam tahap implementasi. Pada dasarnya terdapat beberapa *package* seperti *View*, *ViewModel*, *Model* dan *Utils* yang masing-masing memuat kelas. Kelas-kelas tersebut akan saling berhubungan dengan kelas di dalam *package* yang lain. Perancangan kelas diagram aplikasi Kitana ditunjukkan dalam Gambar 5.12. Pada bagian perancangan ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah yang dapat dilihat pada Tabel 5.8.

Misalnya pada *package View* (ditunjukkan dalam Gambar 5.13) terdapat beberapa kelas seperti *LoginActivity*, *RegisterActivity*, *HomeActivity*, *DetailActivity*, *BerandaFrgment*, *CuacaFrgament*, *InfoCuacaFrgament*, *MaritimFrgament*, *ProfilFrgament*, *AddPostActivity*, dan *EarlyWarningFrgament*. Dari sekian kelas tersebut kelas-kelas *BerandaFrgament*, *CuacaFrgament*, *InfoCuacaFrgament* *MaritimFrgament* dan *ProfilFrgament* yang merupakan *extend* dari kelas *Frgament*.

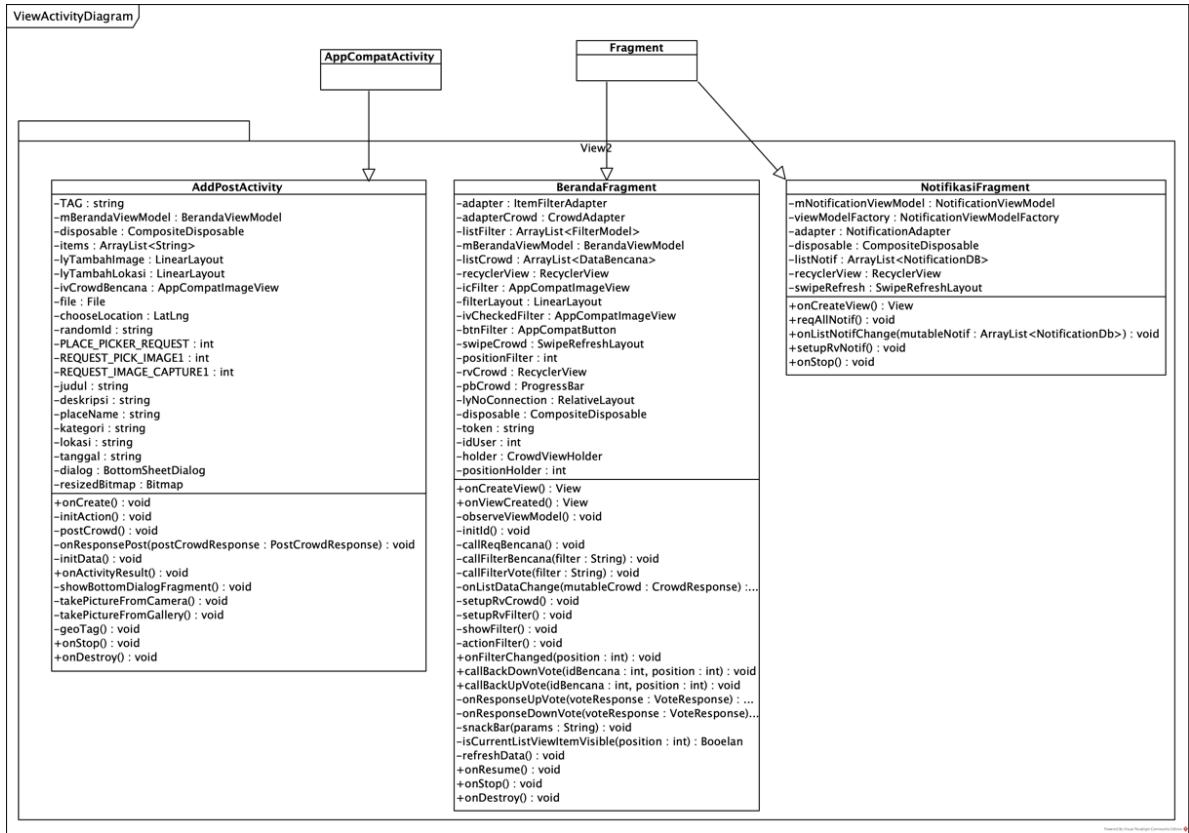
Gambar 5.14 menunjukkan gambar *class diagram* *Package ViewModel*. Diagram tersebut berisi kelas antara lain *AuthViewModel*,

NotificationViewModel, SettingsViewModel, CuacaViewModel, dan BerandaViewModel. Sedangkan dalam Gambar 5.15 menunjukkan *class diagram package* Model yang berisi kelas TideData, WeatherMain, CrowdResponse, DataBencana, DetailVote, ProfilResponse, LogoutResponse, VoteResponse, LoginResponse, UpdateProfilResponse, RegisterResponse, FilterModel, LoginData, PostCrowdResponse, NotificationModel dan CrowdNotifModel.



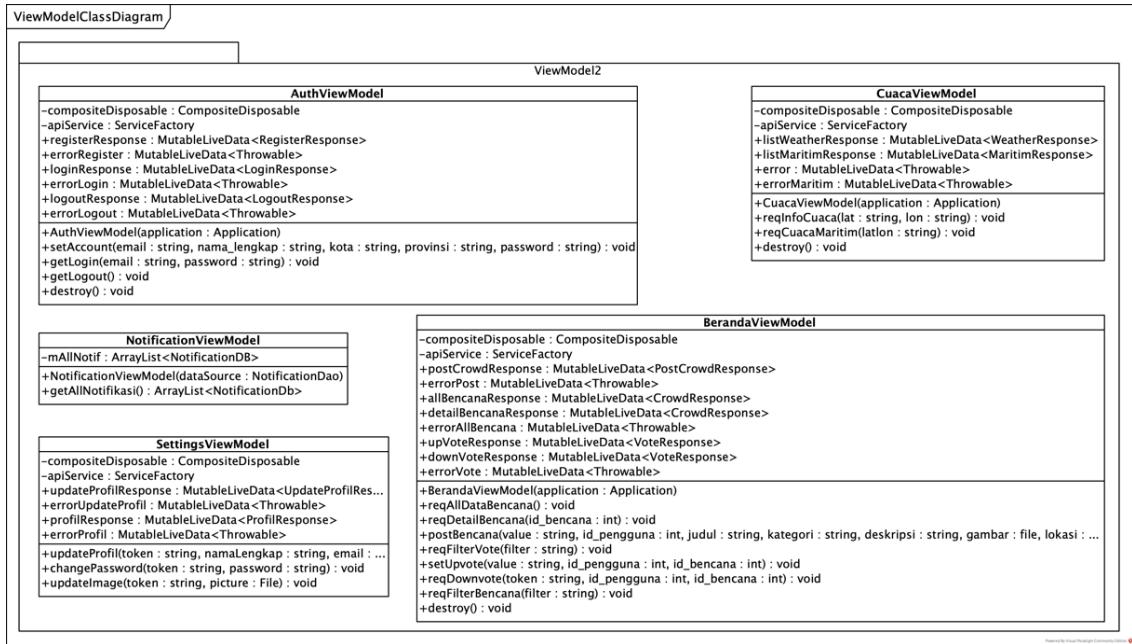
Gambar 5.12 *Class diagram* Kitana

Gambar 5.12 menunjukkan *package View* digunakan untuk menampilkan informasi ke pengguna. Kelas AddPostActivity yang merupakan *extend* dari kelas AppCompatActivity digunakan untuk membuat *posting* bencana alam. Kelas BerandaFragment yang merupakan *extend* dari kelas Fragment digunakan untuk menampilkan informasi bencana alam, melakukan *upvote* dan *downvote* *posting* bencana alam, serta *filter* *posting* bencana alam. Kelas NotifikasiFragment yang merupakan *extend* dari kelas Fragment digunakan untuk menampilkan informasi notifikasi *early warning*.



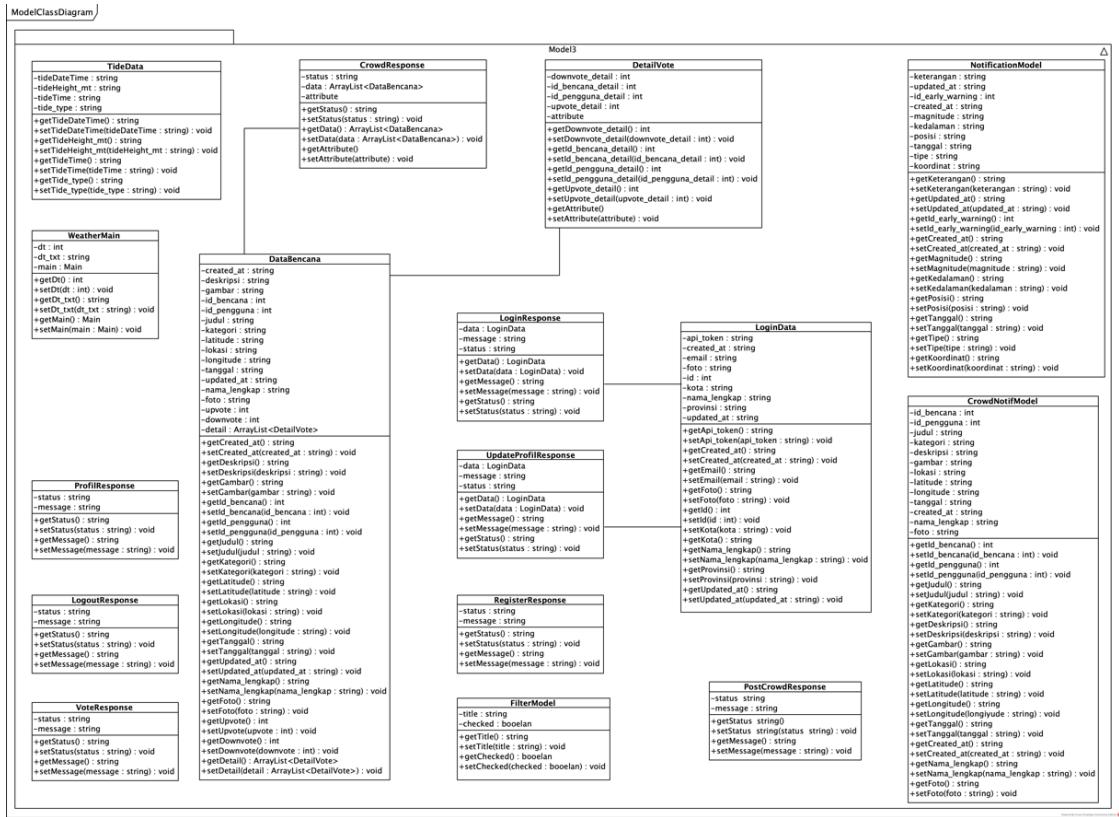
Gambar 5.13 Class diagram Package View

Gambar 5.13 menunjukkan package *ViewModel* digunakan untuk melakukan *request* data ke *server* atau *local storage*. Kelas *AuthViewModel* yang bertanggung jawab melakukan *register* dan *login* ke *server*. Kelas *NotificationViewModel* yang bertanggung jawab memuat data notifikasi *early warning* dari *local storage* (*SQLite*). Kelas *SettingsViewModel* yang bertanggung jawab melakukan *upload* foto profil, ubah *password*, dan mengubah akun profil ke *server*. Kelas *BerandaViewModel* yang bertanggung jawab melakukan *request* data informasi bencana alam, *filter posting*, tambah *posting* bencana alam, *upvote* dan *downvote* bencana alam ke *server*.



Gambar 5.14 Class diagram Package ViewModel

Gambar 5.15 menunjukkan *package Model* digunakan untuk menyimpan dan membaca data dari hasil *request* server yang nantinya informasi tersebut akan diterima oleh pengguna. Kelas WeatherMain dan TideData berfungsi untuk menyimpan data cuaca dan cuaca maritim. Kelas CrowdResponse, DataBencana, DetailVote berfungsi untuk menyimpan informasi bencana alam hingga detail jumlah upvote dan downvote. Kelas ProfileReponse dan LogoutResponse berfungsi untuk menyimpan informasi pesan sukses atau gagal dalam melakukan ubah foto profil dan *user logout*. Kelas LoginResponse, UpdateProfilResponse dan LoginData berfungsi sebagai menyimpan data sementara ke SharedPreferences yang dapat digunakan kembali seperti fungsi *session* di web. Kelas RegisterResponse dan PostCrowdResponse berfungsi untuk menyimpan informasi pesan sukses atau gagal dalam melakukan *register* dan tambah *posting* bencana alam. Kelas FilterModel untuk menyimpan kondisi melakukan perpindahan posisi atribut *filter posting*. Kelas NotificationModel dan CrowdNotifModel untuk menyimpan data *push notification* dari *early warning* dan *crowdsourcing*.



Gambar 5.15 Class diagram Package Model

5.2.4 Perancangan Basis Data

Pada bagian ini membahas mengenai perancangan basis data yang digunakan dalam proses implementasi aplikasi bencana alam. Perancangan basis data terdiri dari 2 (dua) perancangan, yakni perancangan tabel, dan *physical data model*. Pada bagian perancangan ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah yang dapat dilihat pada Tabel 5.8.

5.2.4.1 Perancangan Tabel

1. Tabel Pengguna

Pada Tabel 5.1 berisi rancangan tabel basis data pengguna yang memiliki kolom sebanyak 9 (sembilan) kolom dan memiliki tipe data yang beragam; seperti integer, varchar dan timestamp. Panjang dari tipe data tersebut 11 hingga 255.

Nama tabel: pengguna

Nama kelas model: Pengguna

Fungsi: Menyimpan data pengguna

Tabel 5.1 Rancangan tabel basis data Pengguna

No	Nama Kolom	Tipe Data	Panjang	Keterangan
1.	id_pengguna	integer	11	NOT NULL
2.	nama_lengkap	varchar	255	NULL
3.	kota	varchar	255	NULL
4.	provinsi	varchar	255	NULL
5.	<i>email</i>	varchar	255	NOT NULL
6.	<i>password</i>	varchar	255	NOT NULL
7.	api_token	varchar	255	NOT NULL
8	foto	varchar	255	NULL
8.	<i>created_at</i>	timestamp	255	NULL
9.	<i>updated_at</i>	timestamp	255	NULL

2. Tabel Bencana

Tabel selanjutnya adalah bencana. Detail rancangan tabel bencana ditunjukkan pada Tabel 5.2. Tabel bencana memiliki 12 (dua belas) kolom yang terdiri dari id_bencana, id_pengguna, hingga kolom *updated_at*. Masing-masing kolom memiliki tipe data yang beragam seperti integer, varchar dan timestamp. Dan memiliki Panjang tipe data yang beragam pula. Yakni 11 hingga 255.

Nama tabel: bencana

Nama kelas model: Bencana

Fungsi: Menyimpan data *posting* dan informasi terkait bencana

Tabel 5.2 Rancangan tabel basis data bencana

No	Nama Kolom	Tipe Data	Panjang	Keterangan
1.	id_bencana	integer	11	NOT NULL
2.	id_pengguna	integer	11	NOT NULL
3.	judul	varchar	255	NULL
4.	kategori	varchar	255	NULL

Tabel 5.2 Rancangan tabel basis data bencana (lanjutan)

5.	deskripsi	varchar	255	NULL
6.	gambar	varchar	255	NOT NULL
7.	lokasi	varchar	255	NULL
8.	<i>latitude</i>	varchar	255	NULL
9.	<i>longitude</i>	varchar	255	NULL
10.	tanggal	varchar	255	NOT NULL
11.	<i>created_at</i>	timestamp	255	NULL
12.	<i>updated_at</i>	timestamp	255	NULL

3. Tabel *Early Warning (Server)*

Detail tabel *early_warning (server)* ditunjukkan pada Tabel 5.3. Dimana tabel ini memiliki kolom sebanyak 9 (sembilan) seperti *id_early_warning*, *tanggal*, *koordinat*, *posisi*, *magnitude*, *kedalaman* hingga *updated_at*. Masing-masing kolom memiliki tipe data yang beragam seperti integer, varchar dan timestamp. Dan memiliki Panjang tipe data yang beragam pula. Yakni 11 hingga 255.

Nama tabel : *early_warning*

Nama kelas model : *NotificationModel*

Fungsi : Menyimpan data notifikasi *early warning*

Tabel 5.3 Rancangan tabel basis data *early warning (server)*

No	Nama Kolom	Tipe Data	Panjang	Keterangan
1.	<i>id_early_warning</i>	integer	11	NOT NULL
2.	<i>tanggal</i>	varchar	255	NULL
3.	<i>koordinat</i>	varchar	255	NULL
4.	<i>posisi</i>	varchar	255	NULL
5.	<i>magnitude</i>	varchar	255	NULL
6.	<i>kedalaman</i>	varchar	255	NULL
7	<i>keterangan</i>	varchar	255	NULL
8	<i>created_at</i>	timestamp	255	NULL
9	<i>updated_at</i>	timestamp	255	NULL

4. Tabel Early Warning (Local Storage)

Detail tabel early_warning (*local storage*) ditunjukkan pada Tabel 5.4. Dimana tabel ini memiliki kolom sebanyak 4 (empat) seperti *id*, *date*, *title*, dan deskripsi. Masing-masing kolom memiliki tipe data yang beragam seperti integer dan varchar. Dan memiliki Panjang tipe data yang beragam pula. Yakni 11 hingga 255.

Nama tabel : notification_table

Nama kelas model : NotificationModel

Fungsi : Menyimpan data notifikasi *early warning* di *client*

Tabel 5.4 Rancangan tabel basis data *early warning* (*local storage*)

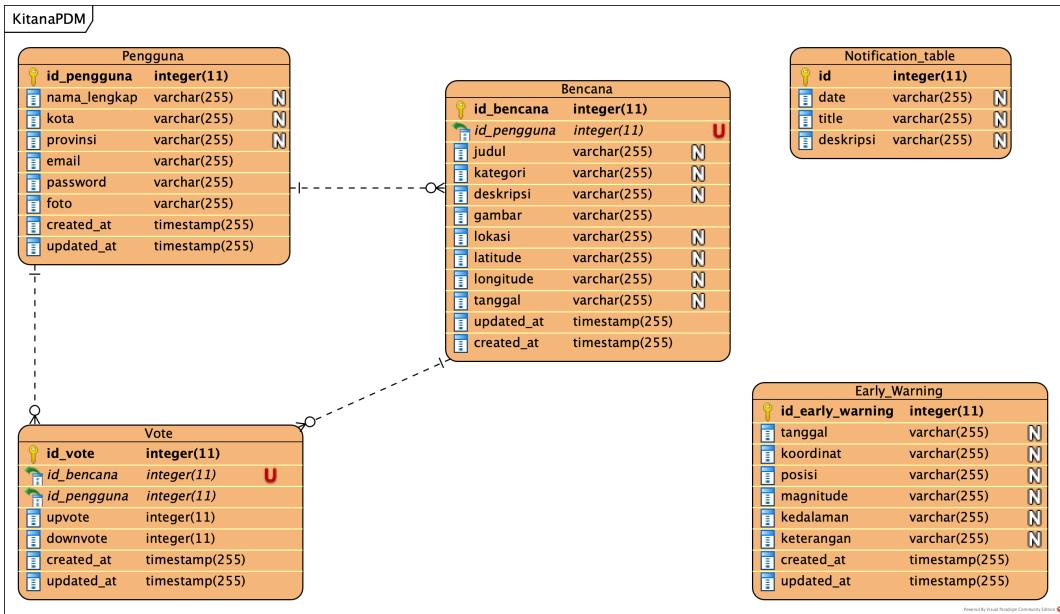
No	Nama Kolom	Tipe Data	Panjang	Keterangan
1.	<i>id</i>	integer	11	<i>NOT NULL</i>
2.	<i>date</i>	varchar	255	<i>NULL</i>
3.	<i>title</i>	varchar	255	<i>NULL</i>
4.	deskripsi	varchar	255	<i>NULL</i>

5.2.5 Physical Data Model

Physical data model ditunjukkan oleh Gambar 5.16 merupakan representasi dari salah satu bentuk rancangan basis data yang digunakan dalam pengembangan aplikasi bencana alam. Terdapat dua tabel seperti Pengguna, dan Bencana dimana masing-masing di dalamnya terdapat kolom beserta tipe data. Di dalam tabel Pengguna terdapat kolom *id_pengguna* yang merupakan *primary key*, *nama_lengkap*, *kota*, *provinsi*, *email*, *password* dan foto yang berelasi dengan tabel Bencana. Tabel Pengguna berelasi dengan tabel Bencana dikarenakan *posting* yang dibuat oleh pengguna memiliki keterkaitan data dengan Tabel Bencana sendiri memiliki 8 (delapan) kolom antara lain, *id_bencana* sebagai *primary key*, dan *id_pengguna* sebagai *unique foreign key* dari tabel Pengguna.

Selain itu tabel Bencana juga berelasi *one-to-many* dengan tabel *Vote* yang memiliki kolom *id_vote* sebagai *primary key*, *id_bencana* sebagai *foreign key* dari tabel Bencana, *upvote* dan kolom *downvote*. Tabel *Vote* berfungsi untuk menyimpan data *vote posting* yang didapatkan dari data tabel bencana. Selain itu terdapat Tabel Early_Warning. Tabel tersebut memiliki 9 (Sembilan) atribut yang terdiri dari 6 (enam) atribut yang bertipe data varchar dengan panjang karakter 255, satu atribut *primary key*, dan 2 (dua) atribut yang bertipe data *timestamp*. Selanjutnya juga terdapat tabel *notification_table* yang memiliki 4 (empat) atribut

yang terdiri dari 3 (tiga) atribut yang bertipe data varchar dengan panjang karakter 22, satu atribut id bertipe *integer* dan *primary key*.



Gambar 5.16 Physical Data Model (*server*)

5.2.6 Perancangan *web service*

Perancangan *web service* merupakan teknik yang digunakan untuk melakukan pertukaran data bersifat dinamis dari *client* (pengguna) ke *server*. Pada bagian ini menjelaskan tentang *endpoint*, *parameter*, *method* dan data apa saja yang akan digunakan aplikasi untuk melakukan pertukaran data ke *web service*. Perancangan *web service* ini hanya menjelaskan 3 (tiga) fitur utama yang terdapat aplikasi Kitana yaitu melihat informasi, membuat *posting* dan menerima notifikasi *early warning* bencana alam. Pada bagian perancangan ini tidak terdapat iterasi dikarenakan fungsionalitas telah memenuhi kebutuhan pengguna, tetapi terdapat iterasi di dalam perancangan antarmuka pengguna berdasarkan temuan masalah yang dapat dilihat pada Tabel 5.8.

5.2.6.1 Menampilkan informasi Bencana Alam

Perancangan *web service* ini dibuat untuk menampilkan informasi bencana yang akan diakses melalui nama *endpoint* `api/showAllBencana` tanpa perlu autentikasi, dikarenakan *Guest* juga dapat melihat informasi bencana alam. *Method* dengan bentuk GET digunakan untuk melakukan pemanggilan *endpoint* `api/showAllBencana`. Setelah sukses mendapatkan respon JSON, objek akan diterima dalam kelas model aplikasi Kitana.

Tabel 5.5 Perancangan web service menampilkan informasi bencana alam

Nama Endpoint	api/showAllBencana
Autentikasi	-
Method	GET
Respond	<pre>{ "status": "200", "message": "success", "data": [{ "id_bencana": 1, "id_pengguna": 1, "judul": "Jalan Klojen Malang Banjir" "kategori": "Banjir", "deskripsi": "Banjir dengan ketinggian 1 meter" "gambar": "image1.png" "lokasi": "Jl Klojen 11 Malang", "tanggal": "11-11-2018 18:10:10" }, { "id_bencana": 2, "id_pengguna": 1, "judul": "Pantai Malang Selatan Gempa" "kategori": "Bencana", "deskripsi": "Gempa dengan kedalaman 1 kilometer" "gambar": "image2.png" "lokasi": "Jl Malang Selatan", "tanggal": "11-11-2018 18:10:10" }] }</pre>

5.2.6.2 Membuat *Posting* Bencana Alam

Perancangan *web service* ini dibuat untuk melihat *posting* bencana yang akan diakses melalui nama *endpoint* api/saveCrowd data menggunakan autentikasi *token user login*. Untuk melakukan pemanggilan *endpoint* api/saveCrowd, digunakan *method* bentuk POST. Tipe *parameter* yang digunakan dalam perancangan API ini adalah RAW.

Tabel 5.6 Perancangan *web service* membuat *posting* bencana alam

Nama Endpoint	api/saveCrowd
Autentikasi	Token User Login
Parameter	{ "id_bencana": 1, "id_pengguna": 1, "judul": "Jalan Klojen Malang Banjir" "kategori": "Banjir", "deskripsi": "Banjir dengan ketinggian 1 meter" "gambar": "image1.png" "lokasi": "Jl Klojen 11 Malang", "tanggal": "11-11- 2018 18:10:10" }
Tipe Parameter	RAW
Method	POST
Respond	{ "status": "200", "message": "Posting bencana berhasil disimpan", }

5.2.6.3 Menampilkan notifikasi *early warning*

Pada perancangan *web service* ini untuk menampilkan notifikasi *early warning* bencana alam yang akan diakses melalui nama *endpoint* scheduleDirasakanBMKG.

Method GET akan digunakan untuk melakukan pemanggilan *endpoint* scheduleDirasakanBMKG. Tipe parameter yang digunakan dalam perancangan API ini adalah RAW.

Tabel 5.7 Perancangan web service Menampilkan notifikasi *early warning*

Nama Endpoint	api/scheduleDirasakanBMKG
Autentikasi	-
Tipe Parameter	RAW
Method	GET
Respond	<pre>{ "id_early_warning": 1, "tanggal": "02/12/2018 01:52:38", "tipe": "Dirasakan", "koordinat": "-2.87, 19.46", "posisi": "2.87 LS, 19.46 BT", "magnitude": "3.9", "kedalaman": "10 Km", "keterangan": "Pusat gempa berada di darat 14 km timur laut Mamasa", "created_at": "2018-12-02 19:26:37", "updated_at": "2018-12-02 19:26:37" }</pre>

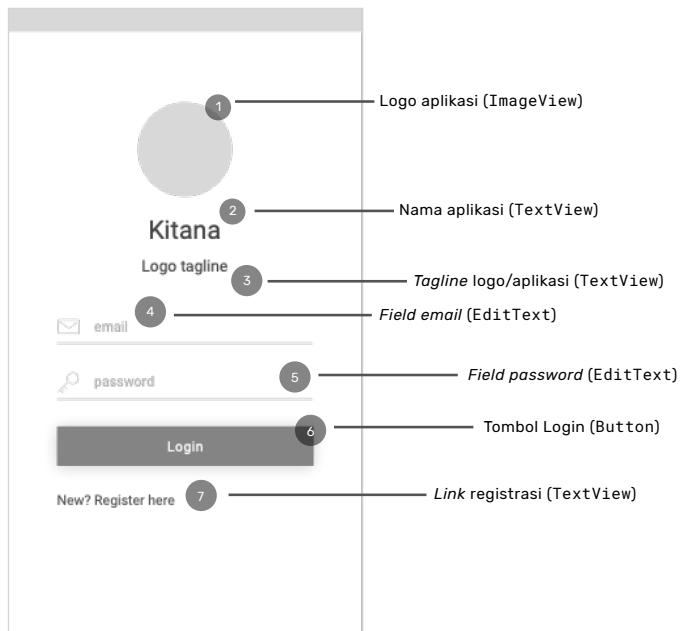
5.2.7 Perancangan Antarmuka Pengguna (*Wireframe*)

Perancangan antarmuka pengguna merupakan salah satu tahap dimana peneliti melakukan rancangan antarmuka aplikasi Kitana yang berupa *wireframe*. Perancangan antarmuka pengguna digunakan untuk memvisualisasikan tampilan antarmuka aplikasi dimana nantinya akan digunakan sebagai acuan dalam mengembangkan aplikasi Kitana. Di dalam perancangan antarmuka pengguna, dibuat *wireframe* yang berisi rancangan *screen* atau halaman yang

merepresentasikan bagaimana aplikasi Kitana dibuat. *Wireframe* tersebut berisi rancangan antarmuka seperti halaman *login*, *register*, tambah *posting*, beranda, informasi cuaca, informasi cuaca maritim, detil informasi cuaca maritim, dan *setting*.

5.2.7.1 *Wireframe Login*

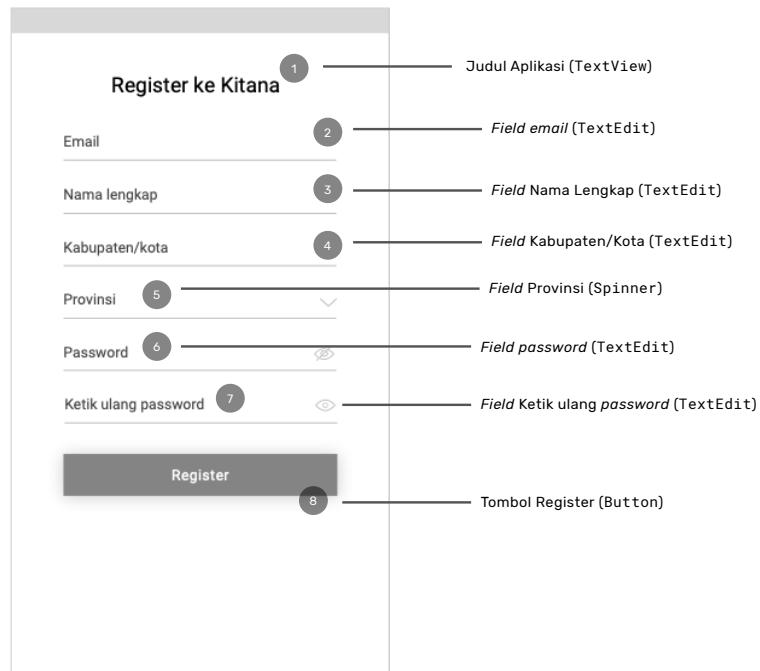
Pada *wireframe login* terdapat informasi seperti pengguna harus memasukkan *email* dan *password*, dan *button* untuk *Login*. Terdapat pula bantuan lupa *password* jika pengguna lupa kredensial akunnya. Selain itu jika pengguna belum mendaftarkan dirinya ke aplikasi, maka terdapat *link* yang mengarahkan ke halaman *register*. Di bagian atas terdapat logo Kitana, dan *tagline* logo Kitana. *Wireframe login* ditunjukkan dalam Gambar 5.17.



Gambar 5.17 *Wireframe Login*

5.2.7.2 *Wireframe Register*

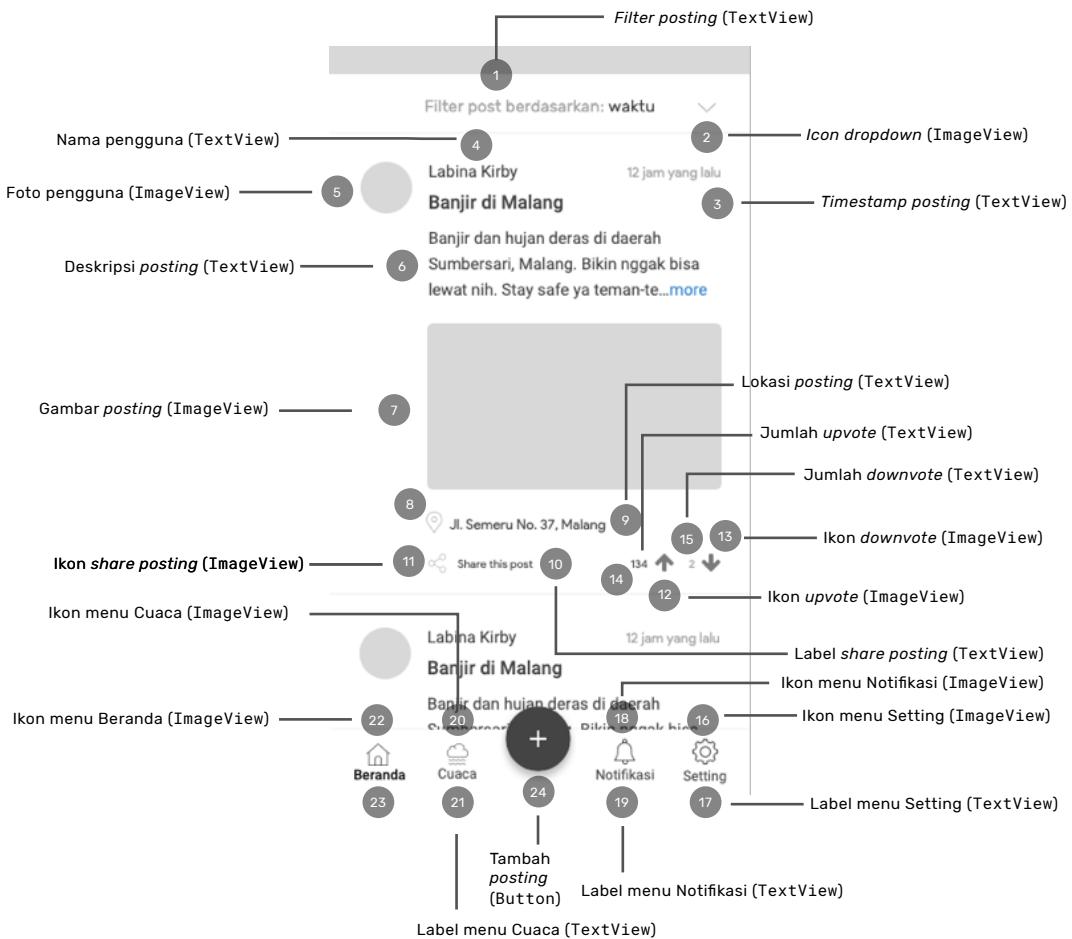
Pada *wireframe register* terdapat *field* yang meliputi *email*, nama lengkap, Kabupaten atau Kota, Provinsi, *password*, dan *field* untuk mengetik ulang *password*. Dalam *field* untuk mengisikan *password* terdapat *icon* untuk memperlihatkan *password* ketika diketik atau membuat *password* tersebut tidak terlihat (*invisible*). *Wireframe register* ditunjukkan dalam Gambar 5.18.



Gambar 5.18 **Wireframe Register**

5.2.7.3 **Wireframe Beranda**

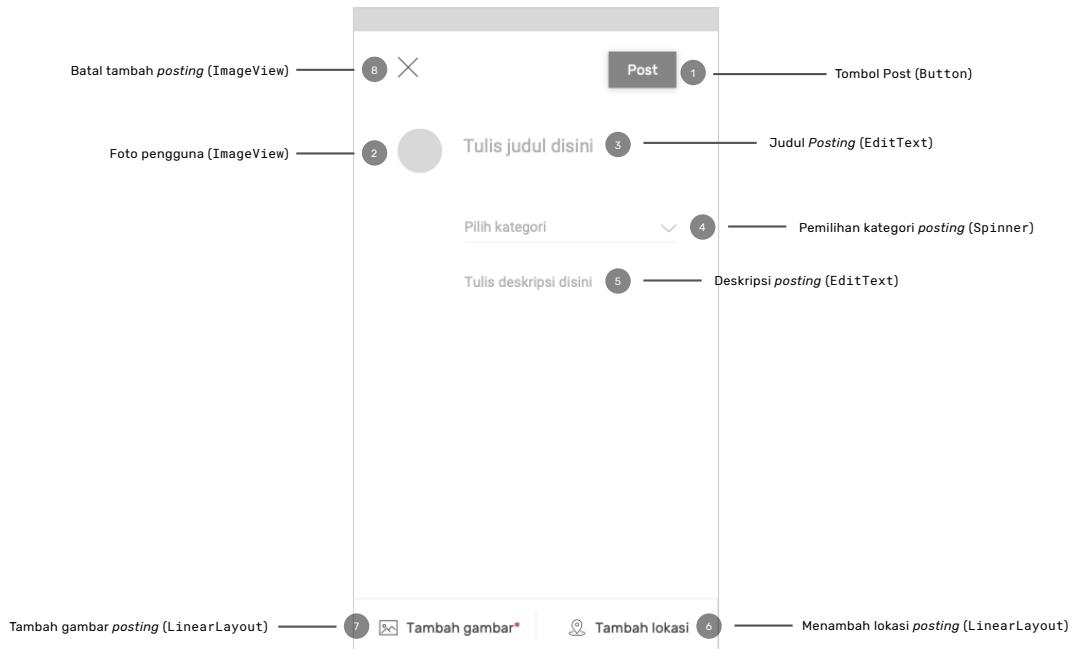
Di dalam *wireframe* Beranda terdapat beberapa informasi seperti rancangan antarmuka untuk *filter post* berdasarkan kategori tertentu, kemudian bagian utama *wireframe* ini terdapat satu *section* untuk *posting* pengguna. Di dalam *section* tersebut terdapat foto profil pengguna yang membuat *posting*, nama, judul, deskripsi, lokasi, dan gambar *posting*. Di bagian sebelah kanan atas terdapat informasi *relative timestamp* kapan *posting* tersebut dibuat. Pada *section* bawah terdapat *icon* untuk membagikan *posting* ke sosial media lain, *upvote* dan *downvote posting*. Disebelah masing-masing *icon downvote* dan *upvote* terdapat label berupa jumlah dari *downvote* atau *upvote* tersebut. *Wireframe* beranda ditunjukkan dalam Gambar 5.19.



Gambar 5.19 Wireframe Beranda

5.2.7.4 Wireframe Tambah Posting

Di dalam wireframe tambah *posting*, terdapat beberapa *field* yang dapat digunakan pengguna untuk membuat *posting* baru yang meliputi judul, kategori, dan deskripsi *posting*. Kategori *posting* tersebut berdasarkan jenis bencana alam yang akan di-post. Di sebelah kiri terdapat foto profil pengguna. Di bagian atas terdapat *button Post* untuk mem-publish *posting*, dan juga *icon cancel* untuk membatalkan pembuatan *posting* baru. Selain itu, di bagian bawah terdapat fungsi untuk menambah gambar *posting* (*required*) dan juga tambah lokasi *posting* bencana alam. Wireframe tambah *posting* ditunjukkan dalam Gambar 5.20.



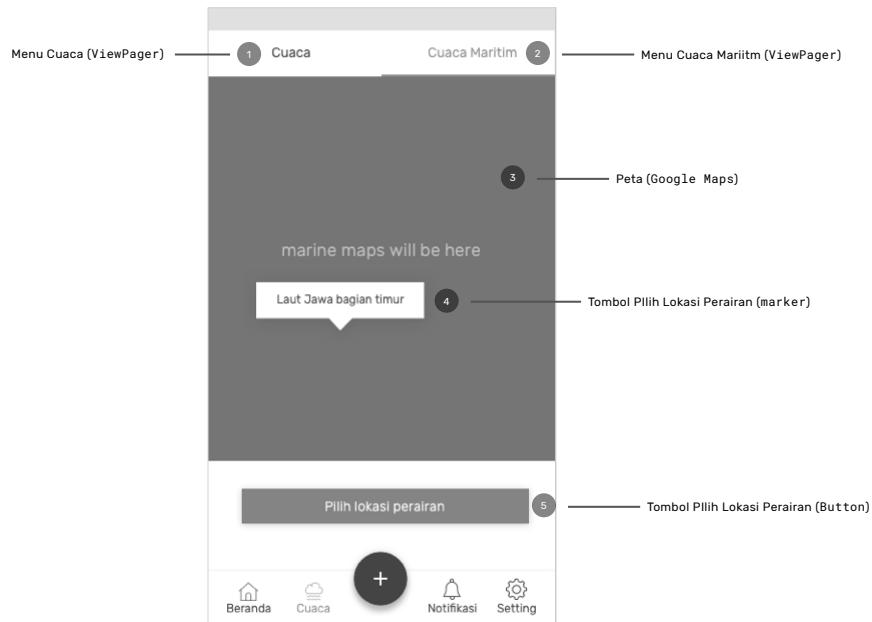
Gambar 5.20 Wireframe Tambah posting

5.2.7.5 Wireframe Cuaca maritim

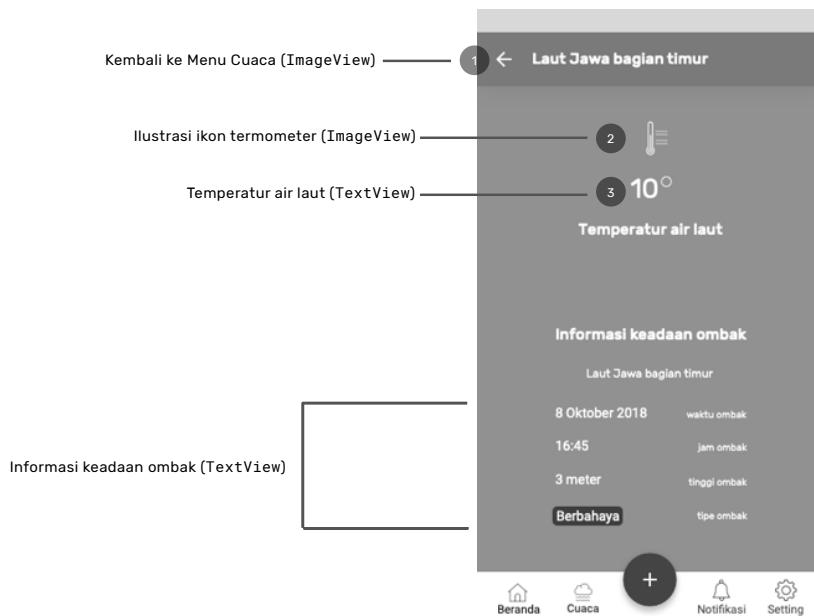
Dalam *wireframe* cuaca maritim terdapat bagian *screen* yang akan memuat *maps*. Pengguna dapat memilih wilayah laut atau maritim untuk dilihat bagaimana informasi cuaca maritimnya. Di dalam *maps* terdapat *marker* yang menunjukkan lokasi perairan atau maritim yang dipilih pengguna. Setelah itu pengguna dapat melanjutkan untuk melihat informasi cuaca maritim tersebut dengan menekan *button* “Pilih lokasi perairan”. *Wireframe* cuaca maritim dapat dilihat pada Gambar 5.18.

5.2.7.6 Wireframe Detil cuaca informasi maritim

Di dalam *wireframe* detil cuaca informasi maritim memuat informasi cuaca maritim seperti temperatur air laut, informasi keadaan ombak yang meliputi waktu, jam, tinggi ombak dalam meter, dan tipe ombak. Gambar 5.21 menunjukkan *wireframe* detil cuaca informasi maritim.



Gambar 5.21 Wireframe Cuaca maritim



Gambar 5.22 Wireframe Detil informasi cuaca maritim

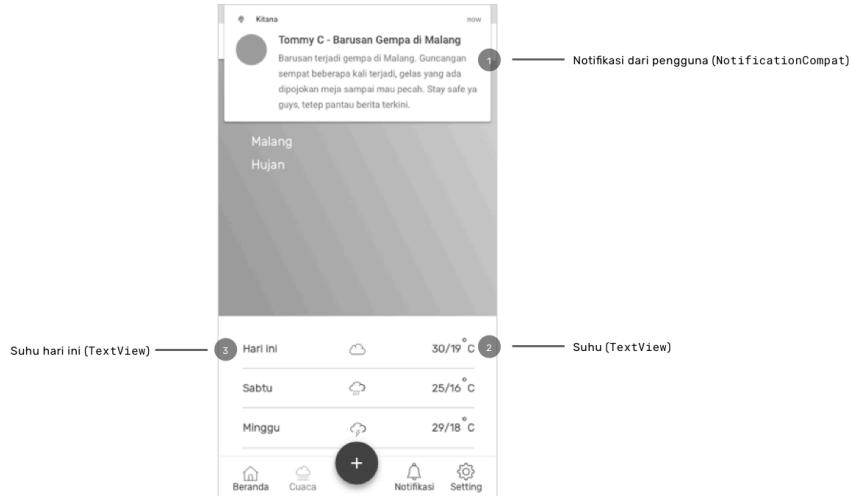
5.2.7.7 Wireframe Menerima notifikasi *crowdsourcing*

Wireframe menerima notifikasi *crowdsourcing* berisi *floating notification* yang memuat informasi seperti foto pengguna yang mem-posting, nama pengguna, judul dan deskripsi *posting*. Wireframe ini ditunjukkan dalam Gambar 5.23.

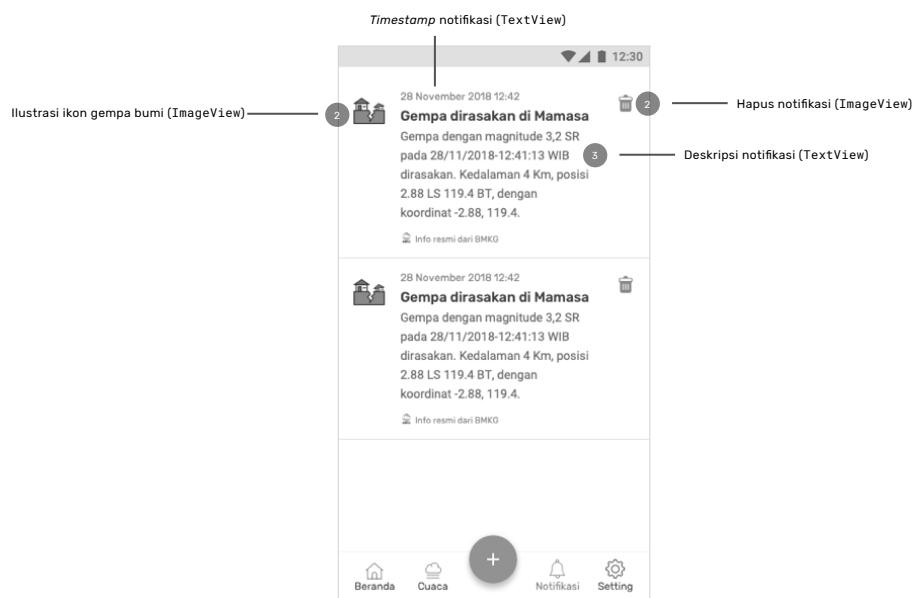
5.2.7.8 Wireframe Melihat notifikasi *early warning*

Wireframe melihat notifikasi *early warning* berisi *list* yang memuat informasi seperti *hashtag* BMKG mengenai bencana, besar skala bencana (dalam contoh

wireframe ini adalah gempa bumi), waktu bencana, dan lokasi gempa. *Wireframe early warning* ini ditunjukkan dalam Gambar 5.24.



Gambar 5.23 *Wireframe* Menerima notifikasi *crowdsourcing*



Gambar 5.24 *Wireframe* Melihat notifikasi *early warning*

5.2.8 Perancangan Antarmuka Pengguna (*Wireframe*) iterasi 1

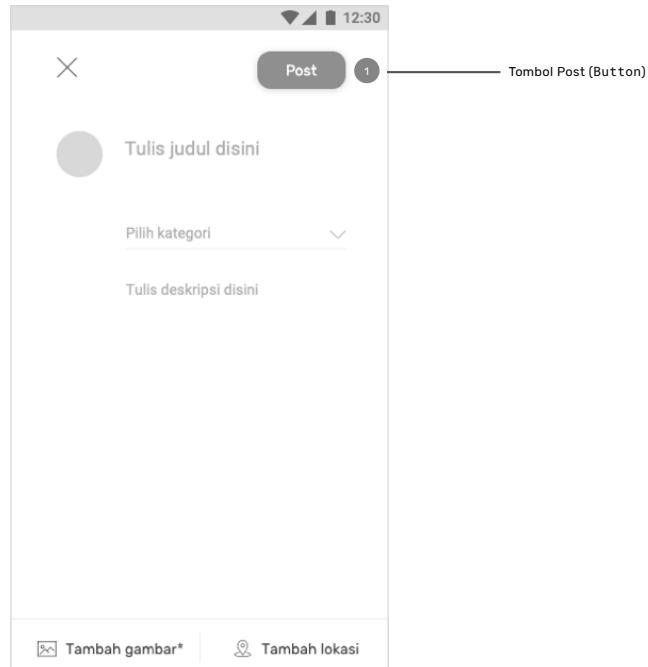
Pada pengembangan aplikasi Kitana ini terdapat iterasi dalam hal perancangan antarmuka pengguna. Hasil iterasi diperoleh dari *walkthrough* aplikasi yang dilakukan oleh pengguna. Perancangan antarmuka pengguna iterasi 1 ini akan direpresentasikan oleh *wireframe*. Evaluasi dilakukan dengan melakukan *walkthrough* terhadap aplikasi Kitana. *Walkthrough* tersebut dilakukan oleh 5 (lima) pengguna. Berikut daftar temuan masalah yang ditunjukkan pada Tabel 5.8.

Tabel 5.8 Temuan masalah aplikasi Kitana

Kode Masalah	Deskripsi masalah	Saran perbaikan
M1	Button “Post” kurang besar sehingga susah untuk diklik.	Menambah <i>width button</i> dari 67dp menjadi 84dp.
M2	Ukuran <i>icon</i> terlalu kecil sehingga susah untuk diklik.	Memperbesar ukuran icon menjadi 20dp x 20dp yang semula awalnya hanya 17dp x 17dp.
M3	Warna <i>icon</i> dapat diganti dengan warna lain yang lebih <i>visible</i> ketika posting di- <i>upvote</i> .	Warna <i>icon downvote</i> dan <i>upvote</i> diganti dengan warna masing-masing #fe5d5d (merah) dan #3881f4 (biru).
M4	Ukuran <i>icon</i> terlalu kecil sehingga susah untuk diklik.	Memperbesar ukuran icon menjadi 20dp x 20dp yang semula awalnya hanya 17dp x 17dp.
M5	Warna <i>icon</i> dapat diganti dengan warna lain yang lebih <i>visible</i> ketika posting di- <i>downvote</i> .	Warna <i>icon downvote</i> dan <i>upvote</i> diganti dengan warna masing-masing #fe5d5d (merah) dan #3881f4 (biru).
M6	Ukuran <i>font</i> terlalu kecil pada informasi cuaca maritim.	Menambah ukuran <i>font</i> di bagian waktu hingga tipe ombak menjadi 12sp.
M7	Ukuran <i>font</i> dan ikon kurang besar.	Mengubah ukuran <i>font</i> dari 10sp menjadi 12sp dan menambah ukuran <i>icon</i> menjadi 20dp.
M8	Tulisan <i>logout</i> tidak <i>clickable</i> , apalagi dengan warna yang berbeda seperti <i>text link</i> .	Mengganti warna Logout yang awalnya warna merah (#E82C2C) dengan warna hitam keabuan #444.

5.2.8.1 Wireframe perbaikan ukuran *button Post*

Permasalahan yang muncul adalah pengguna merasa bahwa *button Post* terlalu kecil untuk diklik. Maka dilakukan perbaikan dengan manambah lebar atau *width button* yang semula berukuran 67dp menjadi 84dp. *Wireframe* perbaikan tersebut ditunjukkan dalam Gambar 5.25.



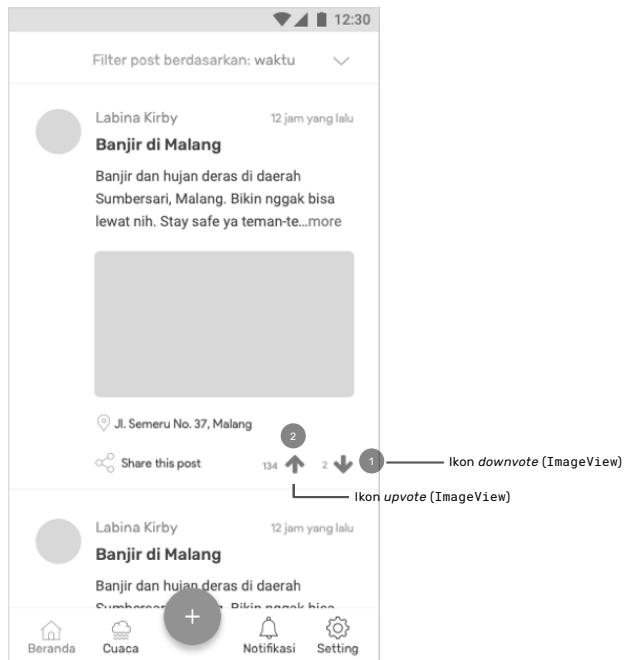
Gambar 5.25 *Wireframe* perbaikan ukuran **button Post**

5.2.8.2 *Wireframe* perbaikan ukuran **icon downvote** dan **upvote**

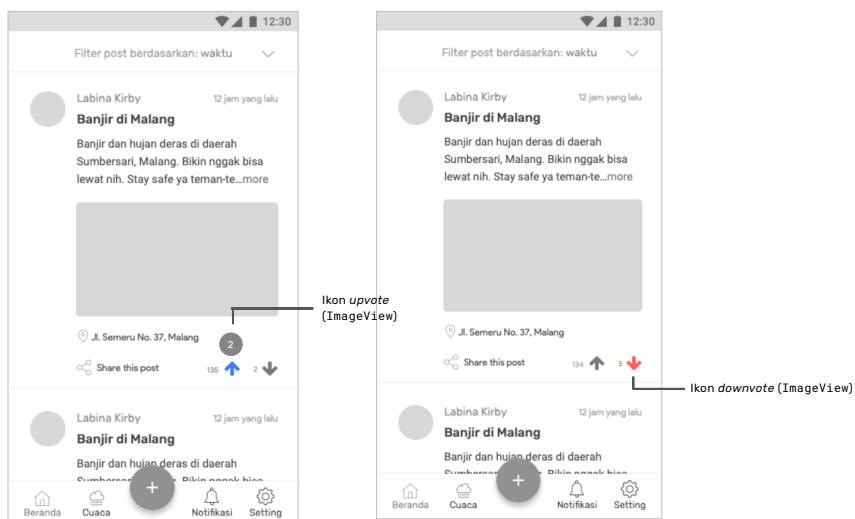
Permasalahan selanjutnya yang dialami oleh pengguna adalah ukuran *icon downvote* dan *upvote* terlalu kecil sehingga agak susah untuk diklik. Maka dilakukan perbaikan dengan menambah ukuran *icon* menjadi berukuran 20dp x 20dp yang semula awalnya hanya 17dp x 17dp. Hasil perbaikan ditunjukkan dalam Gambar *wireframe* 5.26.

5.2.8.3 *Wireframe* perbaikan warna **icon downvote** dan **upvote**

Permasalahan lain terjadi pada fitur *downvote* dan *upvote* dimana pengguna merasa bahwa warna aktif untuk kedua *icon* tersebut belum representatif. Maka perbaikan yang dapat dilakukan adalah dengan mengganti warna *icon downvote* menjadi #fe5d5d (merah) dan *upvote* diganti dengan warna #3881f4 (biru). Kedua *icon* tersebut awalnya memiliki warna yang sama yakni #cc7368 (merah ke oranye). Hasil perbaikan ditunjukkan dalam Gambar *wireframe* 5.27.



Gambar 5.26 Wireframe perbaikan ukuran icon downvote dan upvote



Gambar 5.27 Wireframe perbaikan warna icon downvote dan upvote

5.2.8.4 Wireframe perbaikan ukuran font informasi cuaca maritim

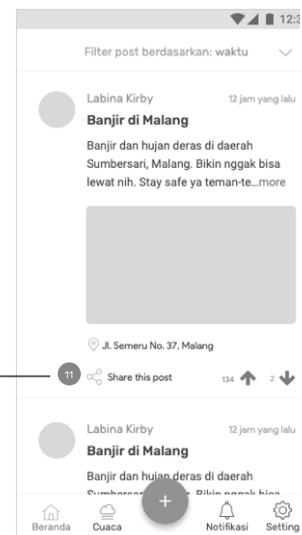
Permasalahan lain ditunjukkan pada menu detail informasi cuaca maritime. Dimana pengguna merasa bahwa ukuran *font* terlalu kecil, membuat *visibility* menjadi rendah. Perbaikan yang dapat dilakukan adalah dengan menambah ukuran *font* dari waktu, jam, tinggi, dan tipe ombak yang awalnya 10sp menjadi 12sp. Hasil perbaikan ditunjukkan dalam Gambar wireframe 5.28.

5.2.8.5 Wireframe perbaikan ukuran font dan icon sharing

Permasalahan selanjutnya terletak pada menu Beranda, fitur *share post*. Pengguna merasa bahwa ukuran *icon* dan label *icon* terlalu kecil. Maka berdasarkan permasalahan tersebut dilakukan perbaikan dengan mengubah ukuran *font* label *icon* dari 10sp menjadi 12sp dan menambah ukuran *icon* menjadi 20dp. Hasil perbaikan ditunjukkan dalam Gambar wireframe 5.29.



Gambar 5.28 Wireframe perbaikan ukuran *font* informasi cuaca maritim



Gambar 5.29 Wireframe perbaikan ukuran *font* dan *icon sharing*

5.2.9 Perancangan Algoritme

Perancangan komponen algoritma menjelaskan tentang proses subsistem dari setiap komponen perangkat lunak. Dari tiap komponen merupakan kumpulan beberapa prosedur yang sistematis berupa proses *algoritma* dan operasi fungsi – fungsi yang terdapat pada aplikasi yang akan dikembangkan. Dalam perancangan algoritma ini hanya menjelaskan 3 (tiga) fitur utama pada aplikasi Kitana yaitu membuat *posting* bencana, melihat informasi bencana dan melihat notifikasi *early warning*.

5.2.10 Perancangan komponen membuat *posting* bencana

Perancangan komponen membuat *posting* bencana dapat dilihat pada Tabel 5.9. Di dalamnya terdapat serangkaian *pseudocode* dimana *pseudocode* dimulai dengan ditekannya tombol *posting* bencana. Terdapat seleksi kondisi apakah semua *field* di dalam *form posting* telah terisi. Jika salah satu *field* belum terisi, maka aplikasi akan menampilkan pesan informasi. Setelah itu terdapat

pengecekan koneksi internet. Jika terdapat koneksi maka *posting* akan di-post. Jika gagal maka akan ditampilkan *toast* “Cek koneksi internet Anda.”.

Nama *class* : AddPostActivity

Nama *method* : postCrowd()

Tabel 5.9 Algoritme perancangan komponen membuat *posting* bencana

No	Pseudocode
1	START
2	Tekan tombol <i>posting</i> bencana
3	IF(cek judul bencana apakah ada yang kosong)
4	Menampilkan pesan informasi “Anda belum <i>input</i> judul <i>posting</i> bencana”
5	ELSE IF(cek spinner kategori bencana apakah di posisi 0)
6	Menampilkan pesan informasi “Anda belum memilih kategori bencana”
7	ELSE IF(cek deskripsi bencana apakah ada yang kosong)
8	Menampilkan pesan informasi “Anda belum <i>input</i> deskripsi bencana”
9	ELSE IF(cek nama lokasi bencana apakah ada yang kosong)
10	Menampilkan pesan informasi “Anda belum memilih lokasi bencana”
11	ELSE IF(cek gambar bencana apakah ada yang kosong)
12	Menampilkan pesan informasi “Anda belum memilih gambar bencana”
13	ELSE
14	IF(cek koneksi internet)
15	IF(idUser != null && token != null)
16	Objek mBerandaViewModel memanggil fungsi <i>postBencana</i>
17	ENDIF
18	ENDIF
19	Menampilkan toast cek koneksi internet anda
20	ENDIF
21	END

5.2.11 Perancangan komponen melihat informasi bencana

Tabel 5.10 menunjukkan algoritme perancangan komponen melihat informasi bencana. Di dalamnya terdapat *pseudocode* pengecekan koneksi internet. Jika

terdapat koneksi internet maka objek mBerandaViewModel memanggil fungsi untuk menampilkan semua data *posting* bencana. Jika tidak maka akan menampilkan halaman “Cek koneksi internet”, dan RecyclerView tidak ditampilkan.

Nama *class* : BencanaFragment
 Nama *method* : callReqBencana()

Tabel 5.10 Algoritme perancangan komponen melihat informasi bencana

No	Pseudocode
1	START
2	IF(cek koneksi internet)
3	objek mBerandaViewModel memanggil fungsi reqAllDataBencana
4	ELSE
5	swipeRefresh berjalan telah berhenti
6	Menampilkan halaman cek koneksi internet
7	recylerview tidak ditampilkan
8	ENDIF
9	
10	END

5.2.12 Perancangan komponen melihat notifikasi *early warning*

Tabel 5.11 menunjukkan algoritme perancangan komponen melihat notifikasi *early warning*. Di dalamnya terdapat *pseudocode* seleksi kondisi dimana jika terdapat nilai atau ukuran lebih 0 maka akan di set nilainya ke adapter list RecyclerView. Jika tidak terdapat nilai maka akan menampilkan toast yang berisi informasi “Data *early warning* tidak ditemukan”.

Nama *class* : NotifikasiFragment
 Nama operasi : onListNotifChange()

Tabel 5.11 Algoritme perancangan komponen melihat notifikasi *early warning*

No	Pseudocode
1	START
2	Inisiasi variabel mutableNotif:
3	MutableList<NotificationDB>
4	IF (mutableNotif.size > 0)
5	listNotif memanggil method clear
6	listNotif menambahkan ke dalam array dari parameter mutableNotif
7	recyclerView melakukan atur nilai adapter
8	ELSE
9	Toast menampilkan informasi "Data Early Warning tidak ditemukan"
10	ENDIF
11	END
12	

BAB 6 IMPLEMENTASI

Bab ini menjelaskan mengenai beberapa hal terkait implementasi sistem atau aplikasi seperti adanya spesifikasi sistem yang meliputi spesifikasi perangkat keras dan lunak, batasan-batasan implementasi, implementasi basis data, algoritme, dan implementasi *user interface*.

6.1 Spesifikasi sistem

Spesifikasi sistem diperoleh berdasarkan Bab 4 (Analisis Kebutuhan) dan Bab 5 (Perancangan). Spesifikasi sistem dilakukan pada 2 (dua) *environment* yang berbeda; terdapat spesifikasi perangkat keras dan perangkat lunak.

6.1.1 Spesifikasi perangkat keras

Pengembangan aplikasi bencana alam *early warning* dan *crowdsourcing* dilakukan dengan menggunakan spesifikasi komputer seperti pada Tabel 6.1.

Tabel 6.1 Spesifikasi perangkat keras komputer

Nama Komponen	Spesifikasi
<i>System Model</i>	MacBook Pro Retina, 13-inch, Early 2015
<i>Processor</i>	2,7 GHz Intel Core i5
<i>Storage</i>	250 GB
<i>Memory</i>	8 GB 1867 MHz DDR3
<i>Display</i>	13,3-inch (2560 x 1600)
Grafis	Intel Iris Graphics 6100 1536 MB

Selanjutnya terdapat spesifikasi perangkat keras *smartphone mobile* untuk dilakukannya proses pengujian, implementasi yang menggunakan sistem operasi Android. Spesifikasi minimal dapat ditunjukkan pada Tabel 6.2.

Tabel 6.2 Spesifikasi perangkat keras *smartphone mobile*

Nama Komponen	Spesifikasi
<i>System Model</i>	Emulator Pixel 2 API 27 (Nougat)
<i>Processor</i>	Octa-core (4 x 2.35 GHz, 4 x 1.9 GHz)
<i>Memory</i>	2,7 GB
<i>Display</i>	5.0-inch (1080x1920 xxhdpi)

Selain itu terdapat spesifikasi perangkat keras *server* yang digunakan untuk proses penyimpanan data ke *database*. Spesifikasi perangkat keras *server* dapat ditunjukkan pada Tabel 6.3.

Tabel 6.3 Spesifikasi perangkat keras *server*

Nama Komponen	Spesifikasi
<i>Server name</i>	srv46
<i>Architecture</i>	x86_64
<i>Kernel version</i>	3.10.0-962.3.2.1ve1.5.24.8.el7.x86_64

6.1.2 Spesifikasi perangkat lunak

Dalam mengembangkan aplikasi *early warning* dan *crowdsourcing* bencana alam dibutuhkan spesifikasi perangkat lunak yang mendukung proses pengembangan tersebut. Adapun spesifikasi perangkat lunak dapat dilihat pada Tabel 6.4. Sedangkan Tabel 6.5 menunjukkan spesifikasi perangkat lunak *smartphone*. Selain itu Tabel 6.6 menunjukkan spesifikasi perangkat lunak *server*.

Tabel 6.4 Spesifikasi perangkat lunak komputer

Nama Komponen	Spesifikasi
<i>Operating System</i>	macOS Mojave
<i>Programming Language</i>	Kotlin
<i>IDE (Integrated Development Environment)</i>	Android Studio 3.2
<i>Editor Dokumentasi</i>	Microsoft Office Word 2016
<i>Editor Perancangan</i>	Visual Paradigm Community Edition

Tabel 6.5 Spesifikasi perangkat lunak *smartphone*

Nama Komponen	Spesifikasi
<i>Operating System</i>	Android versi 7.0 (Nougat)

Tabel 6.6 Spesifikasi perangkat lunak *server*

Nama Komponen	Spesifikasi
<i>Operating System</i>	Linux
<i>Cpanel Version</i>	76.0 (build 15)

Nama Komponen	Spesifikasi
Apache Version	2.4.37
PHP Version	7.2.14
Jenis server basis data	MariaDB
MySQL Version	10.2.21-MariaDB
phpMyAdmin Version	4.8.3

6.2 Batasan-batasan implementasi

Aplikasi *early warning* dan *crowdsourcing* bencana alam Kitana memiliki beberapa batasan dalam pengembangannya. Batasan-batasan implementasi tersebut adalah:

1. Aplikasi Kitana hanya dapat berjalan *smartphone* dengan sistem operasi *Android* minimal versi 5.0 (Lollipop).
2. Untuk menjalankan aplikasi Kitana membutuhkan koneksi internet.
3. Aplikasi ini melakukan pertukaran data membutuhkan pihak ketiga yaitu *web service* dengan format JSON.
4. Aplikasi ini menggunakan data pihak ketiga yaitu *OpenWeatherMap*, *WorldWeatherOnline* dan data BMKG.

6.3 Implementasi basis data

Basis data yang digunakan dalam pengembangan aplikasi *early warning* dan *crowdsourcing* bencana alam Kitana ini diimplementasikan berdasarkan Bab 5 (Perancangan) bagian perancangan basis data. Terdapat 5 (lima) tabel yang akan dibuat dalam implementasi ini. Pada Tabel 6.7 terdapat implementasi basis data untuk Tabel Pengguna. Terdapat 10 (sepuluh) atribut seperti *id_pengguna*, *nama_lengkap*, *kota*, *provinsi*, *email*, *password*, *foto*, *remember_token*, *created_at* dan *updated_at*. Dimana atribut *kota*, *provinsi*, *email*, *password*, *foto*, dan *remember_token* memiliki tipe data varchar dengan panjang karakter yang beragam. Sedangkan atribut *id_pengguna* memiliki tipe data int.

Tabel 6.7 Implementasi Tabel Pengguna

No	Tabel Pengguna
1	CREATE TABLE `pengguna` (
2	`id_pengguna` int(10) UNSIGNED NOT NULL,
3	`nama_lengkap` varchar(255) COLLATE
4	utf8mb4_unicode_ci DEFAULT NULL,
5	`kota` varchar(255) COLLATE utf8mb4_unicode_ci
6	DEFAULT NULL,
7	`provinsi` varchar(255) COLLATE
8	utf8mb4_unicode_ci DEFAULT NULL,
9	`email` varchar(191) COLLATE utf8mb4_unicode_ci
10	NOT NULL,
11	`password` varchar(255) COLLATE
12	utf8mb4_unicode_ci NOT NULL,
13	`foto` varchar(255) COLLATE utf8mb4_unicode_ci
14	NOT NULL,
15	`remember_token` varchar(100) COLLATE
16	utf8mb4_unicode_ci DEFAULT NULL,
17	`created_at` timestamp NULL DEFAULT NULL,
	`updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
	COLLATE=utf8mb4_unicode_ci;

Pada Tabel 6.8 terdapat implementasi basis data untuk tabel Bencana, terdapat 10 (sepuluh) atribut seperti atribut id_bencana, id_pengguna, judul, hingga atribut terakhir yakni created_at. Untuk atribut id_bencana dan id_pengguna terdapat *value* NOT NULL dimana kedua atribut tersebut tidak boleh null.

Tabel 6.8 Implementasi Tabel Bencana

No	Tabel Bencana
1	CREATE TABLE `bencana` (
2	`id_bencana` int(10) UNSIGNED NOT NULL,
3	`id_pengguna` int(10) UNSIGNED NOT NULL,
4	`judul` varchar(255) COLLATE utf8mb4_unicode_ci
	DEFAULT NULL,
5	`kategori` varchar(255) COLLATE utf8mb4_unicode_ci
	DEFAULT NULL,

Tabel 6.9 Implementasi Tabel Bencana (lanjutan)

6	`deskripsi`	varchar(255)	COLLATE
	utf8mb4_unicode_ci DEFAULT NULL,		
7	`gambar`	varchar(255) COLLATE utf8mb4_unicode_ci	
	NOT NULL,		
8	`lokasi`	varchar(255) COLLATE utf8mb4_unicode_ci	
	DEFAULT NULL,		
9	`latitude`	varchar(255) COLLATE utf8mb4_unicode_ci	COLLATE
	utf8mb4_unicode_ci DEFAULT NULL,		
10	`longitude`	varchar(255) COLLATE utf8mb4_unicode_ci	COLLATE
	utf8mb4_unicode_ci DEFAULT NULL,		
11	`tanggal`	varchar(255) COLLATE utf8mb4_unicode_ci	
	NOT NULL,		
12	`created_at`	timestamp NULL DEFAULT NULL,	
13	`updated_at`	timestamp NULL DEFAULT NULL,	
14	CONSTRAINT `bencana_id_pengguna_foreign` FOREIGN KEY		
	(`id_pengguna`) REFERENCES `pengguna`(`id_pengguna`)		
15) ENGINE=InnoDB	DEFAULT	CHARSET=utf8mb4
	COLLATE=utf8mb4_unicode_ci;		

Pada Tabel 6.9 menunjukkan implementasi basis data untuk tabel *Vote*. Terdapat 4 (empat) atribut di dalamnya, seperti *id_vote*, *id_pengguna*, *id_bencana*, *upvote*, *downvote*, terdapat constraint *vote_id_bencana_foreign* yang berisi atribut *id_bencana* yang merupakan *foreign key* terhadap atribut *id_bencana* di tabel *Bencana* dan constraint *vote_id_pengguna_foreign* yang berisi atribut *id_pengguna* yang merupakan *foreign key* terhadap atribut *id_pengguna* di tabel *Pengguna*.

Tabel 6.10 Implementasi Tabel Vote

No	Tabel Vote
1	CREATE TABLE `vote` (
2	`id_vote` int(10) UNSIGNED NOT NULL,
3	`id_pengguna` int(10) UNSIGNED NOT NULL,
4	`id_bencana` int(10) UNSIGNED NOT NULL,
5	`upvote` int(11) NOT NULL,

Tabel 6.9 Implementasi Tabel Vote (lanjutan)

```
6     `downvote` int(11) NOT NULL,  
7     `created_at` timestamp NULL DEFAULT NULL,  
8     `updated_at` timestamp NULL DEFAULT NULL  
9     CONSTRAINT `vote_id_bencana_foreign` FOREIGN KEY  
(`id_bencana`) REFERENCES `bencana`(`id_bencana`)  
10    CONSTRAINT `vote_id_pengguna_foreign` FOREIGN KEY  
(`id_pengguna`) REFERENCES `pengguna`(`id_pengguna`)  
11    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_unicode_ci;
```

Implementasi Tabel Early Warning ditunjukkan pada Tabel 6.10 dimana Didalamnya terdapat 10 (sepuluh) atribut, seperti id_early_warning, tanggal, tipe, koordinat, posisi, magnitude, kedalaman, keterangan, created_at dan updated_at. Atribut yang memiliki tipe data varchar adalah tanggal, tipe, koordinat, posisi, magnitude, kedalaman, dan keterangan dengan panjang maksimal 255 karakter.

Tabel 6.11 Implementasi Tabel Early Warning

No	Tabel Early Warning
1	CREATE TABLE `early_warning` (
2	`id_early_warning` int(10) UNSIGNED NOT NULL,
3	`tanggal` varchar(255) COLLATE
4	`utf8mb4_unicode_ci` DEFAULT NULL,
5	`tipe` varchar(255) COLLATE utf8mb4_unicode_ci
6	DEFAULT NULL,
7	`koordinat` varchar(255) COLLATE
8	`utf8mb4_unicode_ci` DEFAULT NULL,
9	`posisi` varchar(255) COLLATE
10	`utf8mb4_unicode_ci` DEFAULT NULL,
11	`magnitude` varchar(255) COLLATE
12	`utf8mb4_unicode_ci` DEFAULT NULL,
13	`kedalaman` varchar(255) COLLATE
14	`utf8mb4_unicode_ci` DEFAULT NULL,
15	`keterangan` varchar(255) COLLATE
16	`utf8mb4_unicode_ci` DEFAULT NULL,
	`created_at` timestamp NULL DEFAULT NULL,
	`updated_at` timestamp NULL DEFAULT NULL

Tabel 6.10 Implementasi Tabel Early Warning (lanjutan)

17) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
----	--

Implementasi basis data untuk tabel Notifikasi (*server*) ditunjukkan Tabel 6.11. terdapat 5 (lima) atribut seperti id_notifikasi, id_pengguna, status, created_at dan updated_at. Masing-masing dari atribut tersebut memiliki tipe data int.

Tabel 6.12 Implementasi Tabel Notifikasi (*server*)

No	Tabel Notifikasi
1	CREATE TABLE `notifikasi` (`id_notifikasi` int(10) UNSIGNED NOT NULL, `id_pengguna` int(10) UNSIGNED NOT NULL, `status` int(11) NOT NULL, `created_at` timestamp NULL DEFAULT NULL, `updated_at` timestamp NULL DEFAULT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

Implementasi basis data untuk tabel Notifikasi (*local storage*) ditunjukkan Tabel 6.12 terdapat 4 (empat) atribut seperti id, date, title, dan deskripsi. Atribut yang memiliki tipe data varchar yaitu date, title, dan deskripsi. Selain itu atribut id memiliki tipe data int dan primary key.

Tabel 6.13 Implementasi Tabel Notifikasi (*local storage*)

No	Tabel Notifikasi
1	@Entity(tableName = "notification_table")
2	data class NotificationDB(
3	@ColumnInfo(name = "date")

Tabel 6.12 Implementasi Tabel Notifikasi (*local storage*) (lanjutan)

4	val date: String,
5	@field:ColumnInfo(name = "title")
6	val title: String,
7	@field:ColumnInfo(name = "deskripsi")
8	val deskripsi: String) {
9	@PrimaryKey(autoGenerate = true)
10	var id: Int = 0
11	}

6.4 Implementasi algoritme

Algoritme yang akan diimplementasi berasal dari hasil analisis kebutuhan yang dilakukan sebelumnya. Algoritme yang akan diimplementasi merupakan fungsi-fungsi utama dalam aplikasi *crowdsourcing* dan *early warning* bencana alam Kitana. Fungsi tersebut meliputi 3 (tiga) fitur utama pada aplikasi Kitana yaitu menambahkan *posting* bencana, menampilkan informasi *early warning* dan menampilkan informasi bencana yang berupa *timeline*.

6.4.1 Algoritme menambahkan *posting* bencana

Implementasi kode program pada method `postCrowd()` berada pada class `AddPostActivity`, dimana kode program ini digunakan untuk mengisi data *posting* bencana dengan cara melakukan pengecekan `editText` dan `spinner` sesuai yang ada di database pada *form posting* bencana. Setelah `editText` dan `spinner` terdapat nilai yang akan menjadi input, maka dilakukan pengecekan koneksi internet, jika terdapat koneksi internet, maka akan dilakukan *request* pengiriman data dengan objek `mBerandaViewModel` serta *method* `postBencana` dengan *parameter* yang diperoleh dari `editText` dan `spinner`. Jika tidak terdapat koneksi internet, maka dilakukan pemanggilan *toast* “Periksa koneksi internet Anda”. *Source code* dari algoritme ini dapat dilihat pada Tabel 6.13.

Nama Class : `AddPostActivity`

Nama Method : `postCrowd()`

**Tabel 6.14 Source code Method `postCrowd()` Class:
AddPostActivity**

No	Source Code
1	private fun postCrowd() {
2	if(isEmpty(edtTitlePost.text.toString())) {
3	setError(edtTitlePost, "Anda belum input judul posting bencana")
4	edtTitlePost.requestFocus()
5	} else if(spinnerPost.selectedItemPosition ==
6	0) {
7	setSpinnerError(spinnerPost,"Anda belum memilih kategori bencana")
8	spinnerPost.requestFocus()
9	}
10	else if(isEmpty(edtDeskripsiPost.text.toString())) {
11	setError(edtDeskripsiPost, "Anda belum input deskripsi bencana")
12	edtDeskripsiPost.requestFocus()
13	}
14	else if(isEmpty(tvShowNamePlace.text.toString())) {
15	toast(this, "Anda belum memilih lokasi bencana")
16	lyTambahLokasi?.requestFocus()
17	} else {
18	judul = edtTitlePost.text.toString()
19	kategori = spinnerPost.selectedItem.toString()
20	deskripsi = edtDeskripsiPost.text.toString()
21	placeName = tvShowNamePlace.text.toString()

Tabel 6.13 Source code Method `poxstCrowd()` Class: AddPostActivity (lanjutan)

```
22                     lokasi
checkLocationMaritim(chosenlocation.latitude,
chosenlocation.longitude, this)

23                     tanggal = getCurrentDate()

24                     geoTag(file.getAbsolutePath,
chosenlocation.latitude, chosenlocation.longitude, judul)

25

26                     if(hasInternetAccess(this)) {

27                         showProgressDialog(this)

28                         mBerandaViewModel?.postBencana(3,
judul, kategori, deskripsi,
29                                         file,                      lokasi,
chosenlocation.latitude.toString(),
30                                         chosenlocation.longitude.toString(), tanggal)
31                     } else {
32                         toast(this, "Periksa koneksi internet
anda")
33                     }
34                 }
35             }
```

Penjelasan dari source code method `postCrowd()` Class `AddPostActivity` ditunjukkan pada Tabel 6.14.

Tabel 6.15Penjelasan Source code Method postCrowd() Class: AddPostActivity

Baris	Penjelasan
1-22	Melakukan pengecekan nilai dari <code>editText</code> dan <code>spinner</code> pada form <i>posting</i> Bencana alam
23-33	Mengambil nilai dari <code>editText</code> dan <code>spinner</code> disimpan dalam bentuk variabel

Tabel 6.14 Penjelasan *Source code Method postCrowd () Class: AddPostActivity* (lanjutan)

35-42	Melakukan pengecekan koneksi internet, jika terdapat koneksi maka melakukan pemanggilan <i>progressDialog</i> dan mengirim <i>posting</i> bencana dengan parameter nilai yang akan dikirim ke <i>web service</i>
43-47	Apabila tidak terdapat koneksi internet maka melakukan pemanggilan <i>toast</i> "Periksa koneksi internet Anda".

6.4.2 Algoritme menampilkan informasi bencana alam

Implementasi kode program pada method *callReqBencana()* berada pada class *BerandaFragment*, dimana kode program ini digunakan untuk melakukan pemanggilan objek pada *mBerandaViewModel* dengan memanggil *method reqAllDataBencana()*. Sebelumnya akan dilakukan pengecekan koneksi internet, jika terdapat koneksi internet melakukan *request* mengambil data bencana yang akan disimpan dalam *List* setelah itu dilakukan pemanggilan adapter untuk ditampilkan *field* ke dalam *layout*. Jika tidak terdapat koneksi internet melakukan pemanggilan *layout* "Tidak ada koneksi internet". *Source code* algoritme ini ditunjukkan dalam Tabel 6.15.

Nama *Class* : *BerandaFragment*
 Nama *Method* : *callReqBencana()*

Tabel 6.16 *Source code Method callReqBencana () Class: BerandaFragment*

No	Sourcecode
1 2 3 4 5 6 7 8 9 10	private fun callReqBencana() { if(hasInternetAccess(context!!)) { mBerandaViewModel?.reqAllDataBencana() } else { swipeCrowd?.isRefreshing = false lyNoConnectionCrowd?.visible() pbCrowd?.gone() rvCrowd?.gone() } }

Penjelasan dari *source code Method callReqBencana () Class : BerandaFragment* akan dijelaskan pada Tabel 6.16.

Tabel 6.17 Penjelasan Source code Method callReqBencana () Class: BerandaFragment

Baris	Penjelasan
1-3	Melakukan pengecekan koneksi internet, jika terdapat koneksi maka melakukan pemanggilan <i>request</i> data semua bencana yang akan ditampilkan bentuk <i>list adapter</i>
4-10	Apabila tidak terdapat koneksi internet maka melakukan pemanggilan <i>layout</i> tidak ada koneksi

6.4.3 Algoritme menampilkan notifikasi *early warning*

Implementasi kode program pada method `reqAllNotif()` berada pada *class* NotifikasiFragment, dimana kode program ini digunakan untuk melakukan pemanggilan objek pada `mNotificationViewModel` dengan memanggil *method* `getAllNotifikasi()`. Pada pemanggilan tersebut ditujukan untuk memperoleh data notifikasi *early warning* dalam bentuk *List* yang kemudian dilakukan pemanggilan *adapter* untuk ditampilkan *field* ke dalam *layout*, seperti yang tertera pada Tabel 6.17.

Nama *Class* : NotifikasiFragment

Nama *Method* : `reqAllNotif()`

Tabel 6.18 Source code Method reqAllNotif() Class : NotifikasiFragment

No	Source code
1	Private fun onListNotifChange(mutableNotif: MutableList<NotificationDB>) {

Penjelasan dari *source code Method* `reqAllNotif()` *Class* : NotifikasiFragment akan dijelaskan pada tabel 6.18.

Tabel 6.19 Penjelasan Source code Method reqAllNotif() Class : NotifikasiFragment

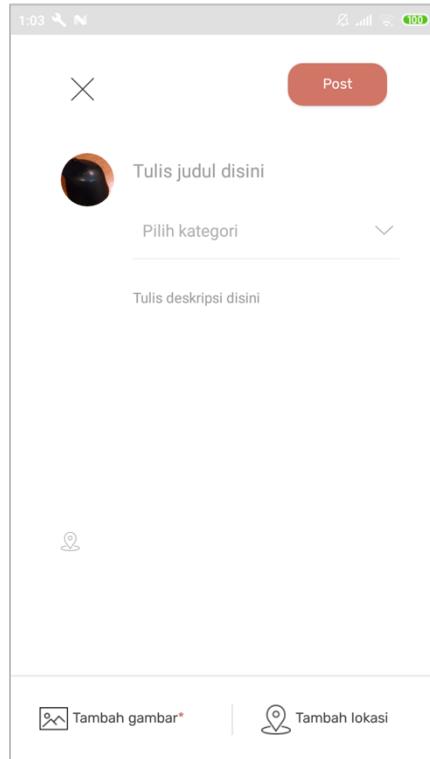
Baris	Penjelasan
1-5	Melakukan pemanggilan objek mNotificationViewModel dengan method getAllNotifikasi() setelah itu dilakukan observe data yang akan di simpan dalam parameter di method onListNotifChange().
7-14	Pada SwipeRefreshLayout digunakan untuk refresh memuat data dan melakukan pemanggilan sama seperti baris 1-5

6.5 Implementasi user interface

Implementasi *user interface* mengacu pada *wireframe* yang telah dibuat pada bab perancangan.

6.5.1 Implementasi user interface membuat posting bencana alam

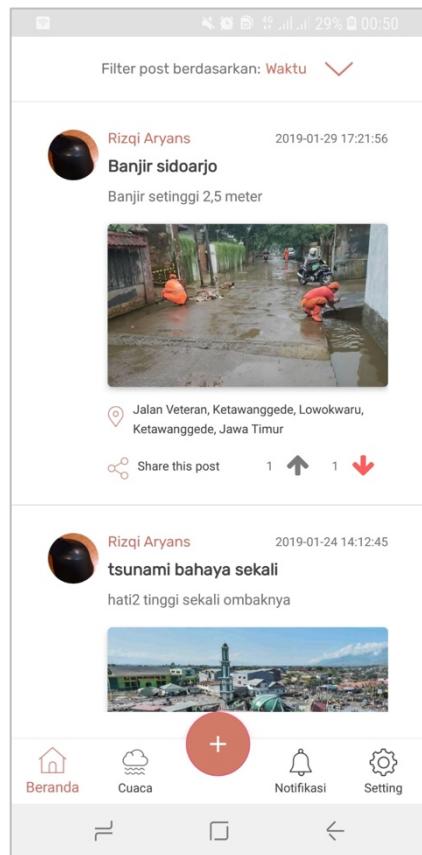
Di dalam implementasi *user interface* tambah *posting*, terdapat beberapa *field* yang dapat digunakan pengguna untuk membuat *posting* baru yang meliputi judul, kategori, dan deskripsi *posting*. Kategori *posting* tersebut berdasarkan jenis bencana alam yang akan di-post. Di sebelah kiri terdapat foto profil pengguna. Di bagian atas terdapat *button Post* untuk mem-publish *posting*, dan juga *icon cancel* untuk membatalkan pembuatan *posting* baru. Selain itu, di bagian bawah terdapat fungsi untuk menambah gambar *posting* (*required*) dan juga tambah lokasi *posting* bencana alam. Implementasi *user interface* tambah *posting* ditunjukkan dalam Gambar 6.1.



Gambar 6.1 Implementasi *user interface* Membuat *posting* bencana alam

6.5.2 Implementasi *user interface* melihat informasi bencana alam

Di dalam implementasi *user interface* melihat informasi bencana alam terdapat beberapa informasi seperti rancangan antarmuka untuk *filter post* berdasarkan kategori tertentu, kemudian bagian utamanya terdapat satu *section* untuk *posting* pengguna. Di dalam *section* tersebut terdapat foto profil pengguna yang membuat *posting*, nama, judul, deskripsi, lokasi, dan gambar *posting*. Di bagian sebelah kanan atas terdapat informasi *relative timestamp* kapan *posting* tersebut dibuat. Pada *section* bawah terdapat *icon* untuk membagikan *posting* ke sosial media lain, *upvote* dan *downvote posting*. Disebelah masing-masing *icon downvote* dan *upvote* terdapat label berupa jumlah dari *downvote* atau *upvote* tersebut. Implementasi *user interface* beranda ditunjukkan dalam Gambar 6.2.



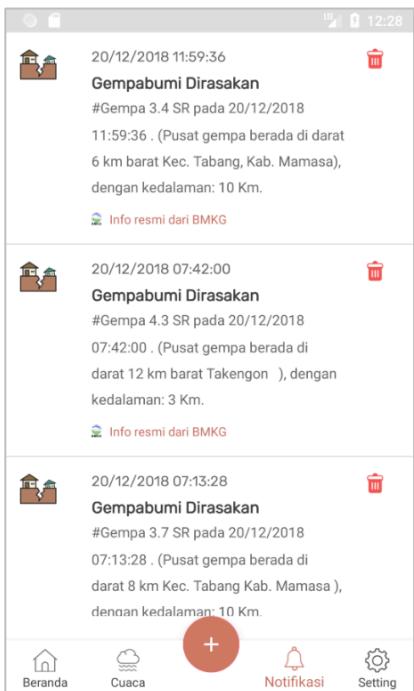
Gambar 6.2 Implementasi *user interface* melihat informasi bencana alam

6.5.3 Implementasi *user interface* melihat notifikasi *early warning*

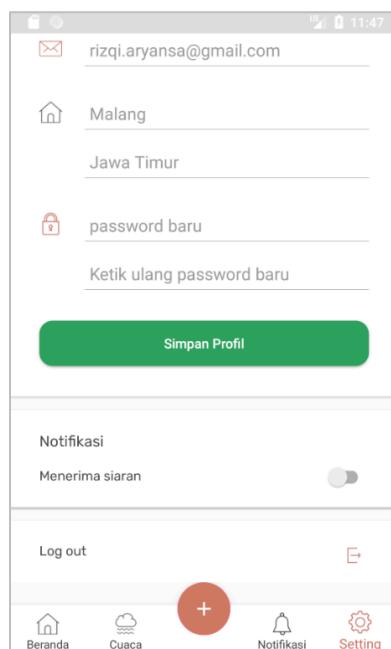
implementasi *user interface* melihat notifikasi *early warning* berisi *list* yang memuat informasi seperti *hashtag* BMKG mengenai bencana, besar skala bencana (dalam contoh *wireframe* ini adalah gempa bumi), waktu bencana, dan lokasi gempa. Implementasi *user interface* *early warning* ini ditunjukkan dalam Gambar 6.3.

6.5.4 Implementasi *user interface* mengubah *profil*

Di dalam implementasi *user interface* yang ditunjukkan pada Gambar 6.4 ini terdapat informasi profil pengguna. Informasi tersebut berupa foto profil, lalu *field* seperti nama, *email*, kota, provinsi, dan *password* pengguna. Kemudian dibawah *form field* tersebut terdapat *button* untuk mengubah *profil*. *Section* selanjutnya terdapat pengaturan untuk menyalakan atau mematikan notifikasi berupa *toggle*. Yang terakhir terdapat *icon log out* untuk keluar dari aplikasi.



Gambar 6.3 Implementasi *user interface* Melihat notifikasi *early warning*

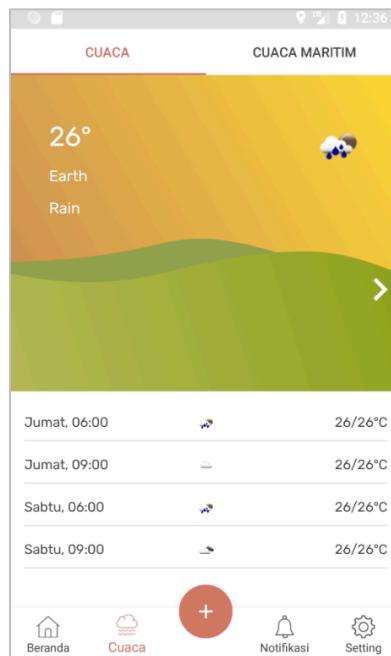


Gambar 6.4 Implementasi *user interface* mengubah profil

6.5.5 Implementasi *user interface* melihat informasi cuaca

Di dalam implementasi *user interface* informasi cuaca menampilkan informasi seperti halnya pada bagian utama terdapat suhu saat ini, cuaca, dan kota pengguna. Selain itu juga terdapat prakiraan cuaca untuk 3 (tiga) hari kedepan yang berisi kondisi cuaca, suhu tertinggi dan terendah dalam *celcius*. Di bagian

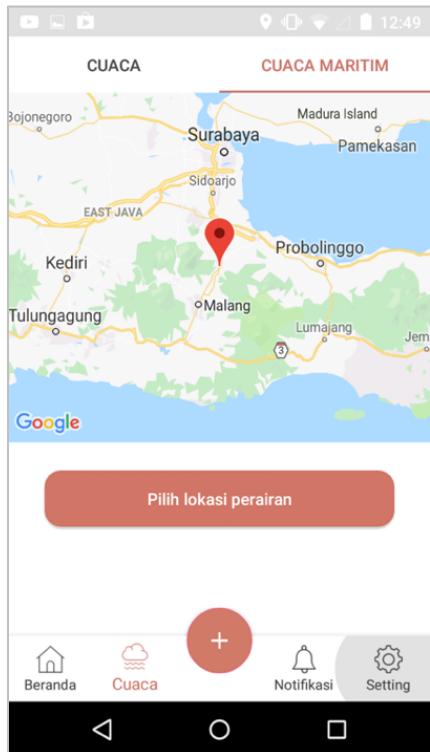
paling atas terdapat 2 menu *tab*; cuaca dan cuaca maritim. Implementasi *user interface* informasi cuaca dapat dilihat pada Gambar 6.5.



Gambar 6.5 Implementasi *user interface* melihat informasi cuaca

6.5.6 Implementasi *user interface* cuaca maritim

Dalam implementasi *user interface* cuaca maritim terdapat bagian *screen* yang akan memuat *maps*. Pengguna dapat memilih wilayah laut atau maritim untuk dilihat bagaimana informasi cuaca maritimnya. Di dalam *maps* terdapat *marker* yang menunjukkan lokasi perairan atau maritim yang dipilih pengguna. Setelah itu pengguna dapat melanjutkan untuk melihat informasi cuaca maritim tersebut dengan menekan *button* "Pilih lokasi perairan". implementasi *user interface* cuaca maritim dapat dilihat pada Gambar 6.6.



Gambar 6.6 Implementasi *user interface* cuaca maritim

6.5.7 Implementasi *user interface* detail informasi cuaca maritim

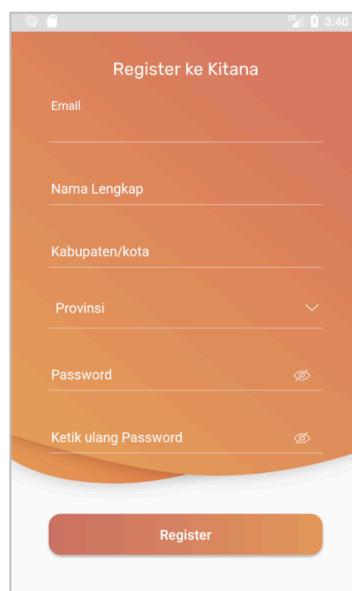
Di dalam implementasi *user interface* detail cuaca informasi maritim memuat informasi cuaca maritim seperti temperatur air laut, informasi keadaan ombak yang meliputi waktu, jam, tinggi ombak dalam meter, dan tipe ombak. Gambar 6.7 menunjukkan implementasi *user interface* detail cuaca informasi maritim.

6.5.8 Implementasi *user interface register*

Pada implementasi *user interface register* terdapat *field* yang meliputi *email*, nama lengkap, Kabupaten atau Kota, Provinsi, *password*, dan *field* untuk mengetik ulang *password*. Dalam *field* untuk mengisikan *password* terdapat *icon* untuk memperlihatkan *password* ketika diketik atau membuat *password* tersebut tidak terlihat (*invisible*). Implementasi *user interface register* ditunjukkan dalam Gambar 6.8.



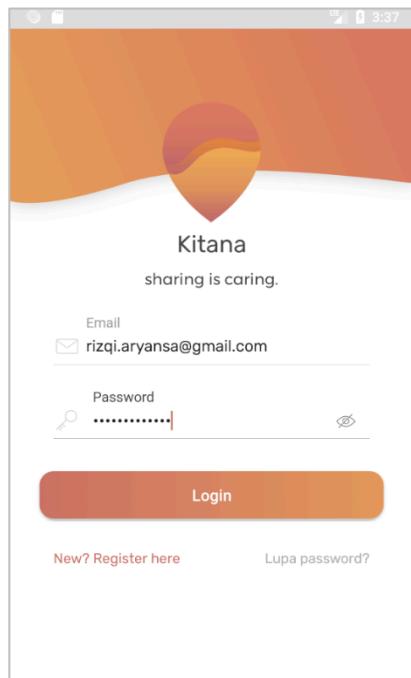
Gambar 6.7 Implementasi *user interface* detail informasi cuaca maritim



Gambar 6.8 Implementasi *user interface register*

6.5.9 Implementasi *user interface login*

Pada implementasi *user interface login* terdapat informasi seperti pengguna harus memasukkan *email* dan *password*, dan *button* untuk *Login*. Terdapat pula bantuan lupa *password* jika pengguna lupa kredensial akunnya. Selain itu jika pengguna belum mendaftarkan dirinya ke aplikasi, maka terdapat *link* yang mengarahkan ke halaman *register*. Di bagian atas terdapat logo Kitana, dan *tagline* logo Kitana. Implementasi *user interface login* ditunjukkan dalam Gambar 6.9.



Gambar 6.9 Implementasi *user interface login*

BAB 7 PENGUJIAN DAN ANALISIS

Pengujian sistem merupakan tahap yang dilakukan setelah selesai dilakukannya implementasi sistem, dimana sistem yang sudah diimplementasikan diujikan dengan berbagai macam pengujian perangkat lunak, seperti pengujian validasi dan pengujian unit. Selain itu, untuk pengujian non fungsional menggunakan pengujian *usability* dan *performance*.

7.1 Pengujian unit

Pengujian unit merupakan pengujian perangkat lunak yang berfokus pada setiap komponen atau unit pada suatu sistem perangkat lunak yang telah diimplementasi sebelumnya. Pengujian unit ini dilakukan bertujuan untuk memastikan perancangan tiap komponen atau unit sudah berjalan sesuai dengan fungsionalitasnya. Metode yang digunakan pengujian unit ini adalah teknik *basis path testing* yang merupakan salah satu teknik dari *whitebox testing*. Metode *basis path testing* dilakukan untuk menentukan setiap jalur yang telah dibuat dapat diujikan. Berikut langkah – langkah yang harus dilakukan dalam membuat kasus uji pada metode *basis path testing*.

1. Pembuatan *flow graph* berdasarkan perancangan algoritma yang telah dilakukan pada tahap sebelumnya.
2. Menentukan *cyclomatic complexity* dari *flow graph* yang telah dibuat.
3. Menentukan *independent path* dari *flow graph* yang akan digunakan sebagai kasus uji.

7.1.2 Pengujian *method postCrowd()* Class : AddPostActivity

Pengujian *method postCrowd()* dilakukan sesuai dengan perancangan algoritma dari implementasi kode program pada sistem. *Pseudocode* pada *method postCrowd()* ditunjukkan pada Tabel 7.1. Setelah itu berdasarkan hasil *pseudocode* tersebut, maka digambarkan menjadi sebuah *flow graph* yang digunakan untuk menghitung *basis path testing* seperti yang ditunjukkan dalam Gambar 7.1. Kemudian dilakukan hasil pengujian tiap jalur yang didapatkan dari perhitungan *cyclomatic complexity* yang telah dijelaskan pada Tabel 7.2.

1. *Pseudocode*

Tabel 7.1 Pseudocode Method postCrowd () Class : AddPostActivity

No	Pseudocode
1	IF(input judul bencana == null)
2	Menampilkan pesan informasi "Anda belum input judul posting bencana"
3	ELSE IF(input spinner kategori bencana == 0)
4	Menampilkan pesan informasi "Anda belum memilih kategori bencana"
5	ELSE IF(input deskripsi bencana == null)
6	Menampilkan pesan informasi "Anda belum input deskripsi bencana"
7	ELSE IF(input nama lokasi bencana == null)
8	Menampilkan pesan informasi "Anda belum memilih lokasi bencana"
9	ELSE IF(input cek gambar bencana == null)
10	Menampilkan pesan informasi "Anda belum memilih gambar bencana"
11	ELSE
11	IF(cek koneksi internet == true)
11	IF(idUser != null && token != null)
12	Objek mBerandaViewModel memanggil fungsi
12	postBencana
13	ENDIF
13	ELSE
14	Menampilkan toast cek koneksi internet Anda
15	ENDIF
15	ENDIF

3. Basis Path Testing

a. Flow Graph

Pemodelan *flow graph* didapatkan dari *perancangan algoritma* pada tahap sebelumnya yaitu *method postCrowd ()*. Berikut *flow graph* *method postCrowd ()* yang ditunjukkan dalam Gambar 7.1.

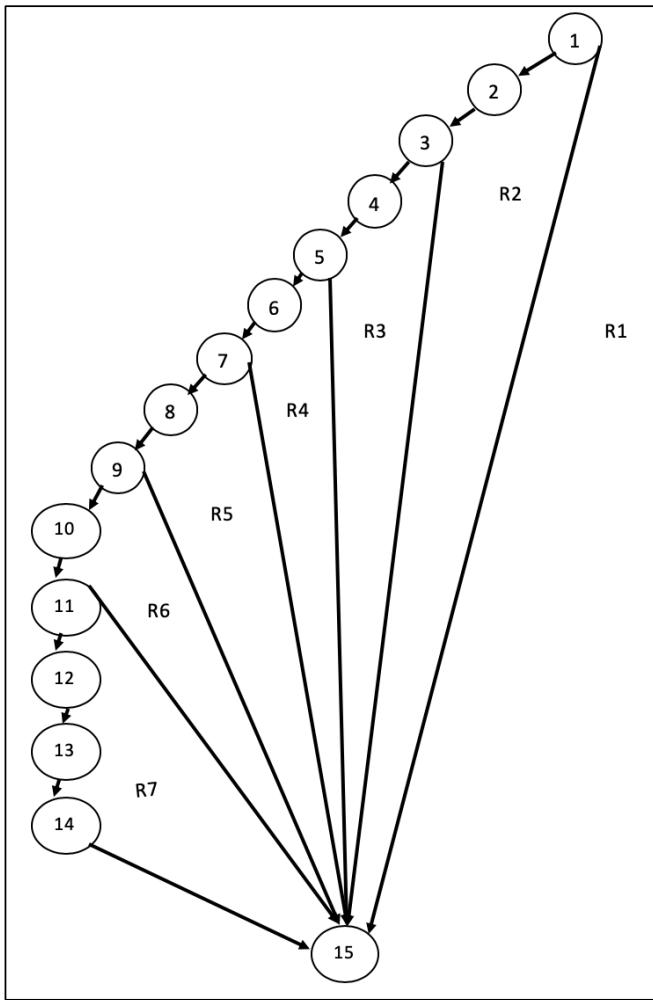
b. Cyclomatic Complexity

Berdasarkan dari *flow graph* *method postCrowd ()* menghasilkan *cyclomatic complexity* terdapat 7 (tujuh) *region* melalui persamaan $V(G)$ yang merupakan jumlah *cyclomatic complexity*, dan E merupakan sebuah garis penghubung antar *node* dan N merupakan jumlah simpul atau *node*.

a) $V(G) = \text{Jumlah Region} = 7$

b) $V(G) = \text{Jumlah Edge} - \text{Jumlah Node} + 2 = 20 - 15 + 2 = 7$

c) $V(G) = 6 \text{ Predicate} + 1 = 7$



Gambar 7.1 Flow Graph Pengujian *method postCrowd*

c. *Independent Path*

Berdasarkan dari perhitungan *cyclomatic complexity* yang telah dilakukan, terdapat 7 (tujuh) basis set dari jalur *independent* sebagai berikut.

- Jalur 1 = 1 – 16
- Jalur 2 = 1 – 2 – 3 – 16
- Jalur 3 = 1 – 2 – 3 – 4 – 5 – 15
- Jalur 4 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 15
- Jalur 5 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 15
- Jalur 6 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 15
- Jalur 7 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15

Hasil pengujian pada *method* `postCrowd()` disetiap jalurnya dijelaskan pada Tabel 7.2

Tabel 7.2 Hasil Pengujian Unit Method `postCrowd()` Class : AddPostActivity

No. Jalur	Prosedur Uji	Expected Result	Result	Status
1	- <i>Input</i> judul bencana sama dengan <i>null</i>	Sistem menampilkan informasi pesan “Anda belum <i>input</i> judul <i>posting</i> bencana”	Sistem menampilkan informasi pesan “Anda belum <i>input</i> judul <i>posting</i> bencana”	Valid
2	- <i>Input spinner</i> kategori bencana sama dengan 0	Sistem menampilkan jenis kategori bencana yang harus dipilih	Sistem menampilkan jenis kategori bencana yang harus dipilih	Valid
3	- <i>Input</i> deskripsi bencana sama dengan <i>null</i>	Sistem menampilkan informasi pesan “Anda belum <i>input</i> deskripsi bencana”	Sistem menampilkan informasi pesan “Anda belum <i>input</i> deskripsi bencana”	Valid
4	- <i>Input</i> deskripsi bencana sama dengan <i>null</i>	Sistem menampilkan informasi pesan “Anda belum memilih lokasi bencana”	Sistem menampilkan informasi pesan “Anda belum memilih lokasi bencana”	Valid
5	- <i>Input</i> gambar bencana sama dengan <i>null</i>	Sistem menampilkan informasi pesan “Anda belum memilih gambar bencana”	Sistem menampilkan informasi pesan “Anda belum memilih gambar bencana”	Valid

Tabel 7.2 Hasil Pengujian Unit Method *postCrowd () Class : AddPostActivity* (lanjutan)

6	- Mengisi semua <i>field</i> form tambah bencana - Koneksi internet tersedia	Sistem berhasil melakukan penyimpanan data tambah bencana ke database	Sistem berhasil melakukan penyimpanan data tambah bencana ke database	Valid
7	- Mengisi semua <i>field</i> form tambah bencana - Koneksi internet tidak tersedia	Sistem menampilkan informasi pesan “Periksa koneksi internet anda”	Sistem menampilkan informasi pesan “Periksa koneksi internet anda”	Valid

7.1.3 Pengujian method *callReqBencana () Class : BerandaFragment*

Pengujian *method callReqBencana ()* dilakukan sesuai dengan perancangan algoritma dari implementasi kode program pada sistem. *Pseudocode* pada *method callReqBencana ()* ditunjukkan pada Tabel 7.3. Selanjutnya dari hasil *pseudocode* digambarkan menjadi sebuah *flow graph* yang digunakan untuk menghitung *basis path testing* seperti yang ditunjukkan dalam Gambar 7.2. Kemudian dilakukan hasil pengujian tiap jalur yang di dapatkan dari perhitungan *cyclomatic complexity* yang telah di jelaskan pada Tabel 7.4.

1. Pseudocode

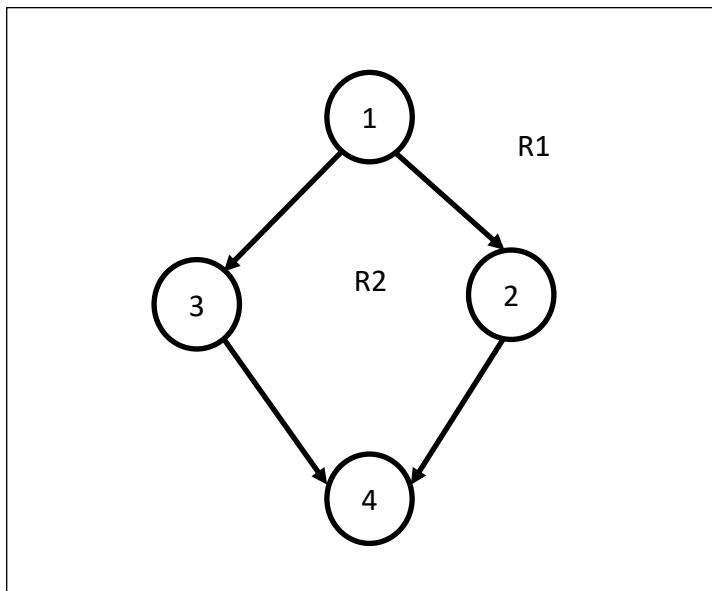
Tabel 7.3 Pseudocode Method *callReqBencana () Class : BerandaFragment*

No	Pseudocode
1	IF(cek koneksi internet)
2	objek mBerandaViewModel memanggil fungsi
2	reqAllDataBencana
3	ELSE
3	swipeRefresh berjalan telah berhenti
3	Menampilkan halaman cek koneksi internet
3	recyclerview tidak ditampilkan
4	ENDIF

2. Basis Path Testing

a. Flow Graph

Pemodelan *flow graph* didapatkan dari perancangan algoritma pada tahap sebelumnya yaitu method `callReqBencana()`. Berikut *flow graph method* `callReqBencana()` yang ditunjukkan dalam Gambar 7.2.



Gambar 7.2 Flow Graph Pengujian method `callReqBencana`

b. *Cyclomatic Complexity*

Berdasarkan dari *flow graph method* `callReqBencana()` yang menghasilkan *cyclomatic complexity*, terdapat 2 (dua) *region* melalui persamaan $V(G)$ yang merupakan jumlah dari *cyclomatic complexity*, E merupakan sebuah garis penghubung antar *node* dan N merupakan jumlah simpul atau *node*.

b) $V(G) = \text{Jumlah Region} = 2$

c) $V(G) = \text{Jumlah Edge} - \text{Jumlah Node} + 2 = 4 - 4 + 2 = 2$

d) $V(G) = 1 \text{ Predicate} + 1 = 2$

c. *Independent Path*

Berdasarkan dari perhitungan *cyclomatic complexity* yang telah dilakukan, terdapat 2 (dua) basis set dari jalur *independent* sebagai berikut.

a) Jalur 1 = 1 – 2 – 4

b) Jalur 2 = 1 – 3 – 4

Hasil pengujian pada *method* callReqBencana () disetiap jalurnya dijelaskan pada Tabel 7.4.

Tabel 7.4 Hasil Pengujian Unit Method callReqBencana () Class: BerandaFragment

No. Jalur	Prosedur Uji	Expected Result	Result	Status
1	- Koneksi internet tersedia	Sistem menampilkan informasi bencana dalam halaman beranda	Sistem menampilkan informasi bencana dalam halaman beranda	Valid
2	- Koneksi internet tidak tersedia	Sistem menampilkan informasi pesan “Periksa koneksi internet anda”	Sistem menampilkan pesan “Periksa koneksi internet anda”	Valid

7.1.4 Pengujian *method* onListNotifChange () Class : NotifikasiFragment

1. Pseudocode

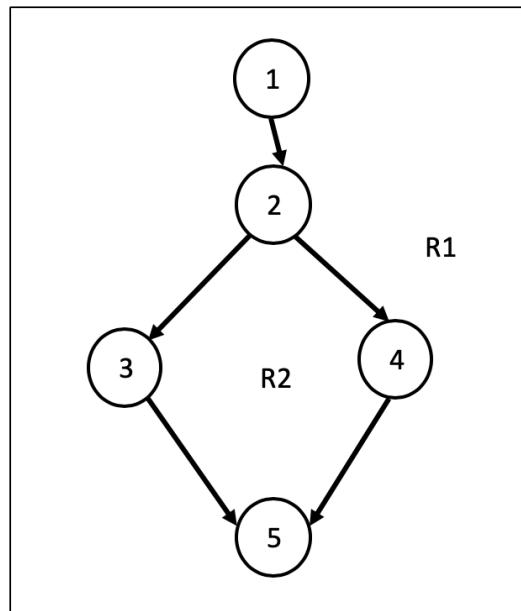
Tabel 7.5 Pseudocode Method onListNotifChange () Class : NotifikasiFragment

No	Pseudocode
1 1 2 3 3 3 4 4 4 5	<pre> Inisiasi variabel mutableNotif: MutableList<Notification DB> IF (mutableNotif.size > 0) listNotif memanggil method clear listNotif menambahkan ke dalam array dari parameter mutableNotif recyclerView melakukan atur nilai adapter ELSE Toast menampilkan informasi "Data Early Warning tidak ditemukan" ENDIF </pre>

3. Basis Path Testing

a. Flow Graph

Pemodelan *flow graph* didapatkan dari perancangan algoritma pada tahap sebelumnya yaitu *method onListNotifChange*. Berikut *flow graph* method *onListNotifChange* yang ditunjukkan dalam Gambar 7.3.



Gambar 7.3 Flow Graph Pengujian *method onListNotifChange*

b. *Cyclomatic Complexity*

Flow graph method onListNotifChange menghasilkan *cyclomatic complexity* terdapat 2 (dua) *region* melalui persamaan $V(G)$ yang merupakan jumlah *cyclomatic complexity*, E merupakan sebuah garis penghubung antar *node* dan N merupakan jumlah simpul atau *node*.

- a) $V(G) = \text{Jumlah Region} = 2$
- b) $V(G) = \text{Jumlah Edge} - \text{Jumlah Node} + 2 = 4 - 4 + 2 = 2$
- c) $V(G) = 1 \text{ Predicate} + 1 = 2$

c. *Independent Path*

Berdasarkan dari perhitungan *cyclomatic complexity* yang telah dilakukan, terdapat 2 (dua) basis set dari jalur *independent* sebagai berikut.

Jalur 1 = 1 – 2 – 4 – 5

Jalur 2 = 1 – 2 – 3 – 5

Hasil pengujian pada *method onListNotifChange()* disetiap jalurnya dijelaskan pada Tabel 7.6.

Tabel 7.6 Hasil Pengujian Unit Method `onListNotifChange ()` Class : `NotifikasiFragment`

No. Jalur	Prosedur Uji	Expected Result	Result	Status
1	- Hasil respon variable <code>mutableNotif</code> lebih dari 0	Sistem menampilkan informasi <i>early warning</i> dalam halaman notifikasi	Sistem menampilkan informasi <i>early warning</i> dalam halaman notifikasi	Valid
2	- Hasil respon variable <code>mutableNotif</code> kurang dari 0	Sistem menampilkan informasi pesan “Data <i>Early Warning</i> tidak ditemukan”	Sistem menampilkan informasi pesan “Data <i>Early Warning</i> tidak ditemukan”	Valid

7.2 Pengujian validasi

Pengujian validasi dilakukan dengan tujuan untuk memvalidasi atau memastikan kembali apakah kebutuhan yang telah diimplementasikan sesuai dengan pemodelan skenario *use case*. Pengujian validasi dapat dikatakan valid apabila aplikasi yang dikembangkan memenuhi kebutuhan yang telah didefinisikan sebelumnya (tahap analisis kebutuhan). Pengujian validasi termasuk dalam pengujian *blackbox* dimana hasil pengujian didapatkan dengan cara menjalankan semua kebutuhan pada aplikasi yang dikembangkan.

7.2.1 Pengujian validasi *login*

Pada Tabel 7.7 terdapat hasil pengujian *login*. Pengujian validasi *login* dilakukan dengan melakukan prosedur seperti pada Tabel 7.7. Hasil yang diharapkan atau *expected result* adalah berhasil untuk *login* jika pengguna memasukkan *email* dan *password* yang benar dan sistem menampilkan halaman beranda dan sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.7 Pengujian Validasi Login

Kode Kebutuhan	F-KT-02
Nama Kasus Uji	<i>Login</i>

Tabel 7.7 Pengujian Validasi *Login* (lanjutan)

Prosedur	<ol style="list-style-type: none">1. Pengguna mengisi <i>email</i> yang telah terdaftar2. Pengguna memasukkan <i>email</i> dan <i>password</i> yang telah di daftarkan3. Pengguna tekan tombol <i>login</i>
Expected Result	Proses <i>login</i> berhasil, dan sistem akan menampilkan halaman beranda
Result	Proses <i>login</i> berhasil, dan sistem akan menampilkan halaman beranda
Status	Valid

7.2.2 Pengujian validasi *registrasi*

Pada Tabel 7.8 terdapat hasil pengujian *registrasi*. Pengujian validasi *login* dilakukan dengan melakukan prosedur seperti pada tabel 7.8. Hasil yang diharapkan atau *expected result* adalah data yang tambahan tersimpan pada sistem dengan menampilkan halaman *login* dan sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.8 Pengujian Validasi *register*

Kode Kebutuhan	F-KT-01
Nama Kasus Uji	<i>Register</i>
Prosedur	<ol style="list-style-type: none">1. Pengguna mengisi field <i>email</i>, nama lengkap, kabupaten/kota, provinsi, <i>password</i>, ulang <i>password</i> di form <i>register</i>2. Pengguna Tekan tombol <i>register</i>
Expected Result	Data yang ditambahkan tersimpan pada sistem dan sistem menampilkan halaman <i>login</i>
Result	Data yang ditambahkan tersimpan pada sistem dan sistem menampilkan halaman <i>login</i>
Status	Valid

7.2.3 Pengujian validasi melihat *posting* bencana

Pada Tabel 7.9 terdapat hasil pengujian melihat *posting* bencana. Pengujian validasi melihat *posting* bencana dilakukan dengan melakukan prosedur seperti pada Tabel 7.9. Hasil yang diharapkan atau *expected result* adalah ditampilkannya

halaman Beranda dan sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid. RRI

Tabel 7.9 Pengujian Validasi melihat *posting* bencana

Kode Kebutuhan	F-KT-03
Nama Kasus Uji	Melihat <i>posting</i> bencana
Prosedur	1. Pengguna membuka aplikasi menuju menu beranda
Expected Result	Menampilkan halaman Beranda
Result	Menampilkan halaman Beranda
Status	<i>Valid</i>

7.2.4 Pengujian validasi membuat *posting* bencana alam

Pada Tabel 7.10 terdapat hasil pengujian membuat *posting* bencana alam. Pengujian validasi membuat *posting* bencana alam dilakukan dengan melakukan prosedur seperti pada tabel 7.10. Hasil yang diharapkan atau *expected result* adalah data tersimpan di *database* dengan menampilkan halaman beranda, hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan *valid*.

Tabel 7.10 Pengujian Validasi membuat *posting* bencana alam

Kode Kebutuhan	F-KT-04
Nama Kasus Uji	Tambah <i>posting</i>
Prosedur	<ol style="list-style-type: none"> 1. Melakukan <i>login</i> 2. Menekan tombol <i>plus</i> di bagian tengah navigasi bawah 3. Mengisi <i>form</i> tambah <i>posting</i> dengan melengkapi <i>field</i> judul, kategori, deskripsi, gambar, dan lokasi <i>posting</i>. 4. Menekan tombol “Post”
Expected Result	Data tersimpan di <i>database</i> dan menampilkan halaman beranda
Result	Data tersimpan di <i>database</i> dan menampilkan halaman beranda
Status	<i>Valid</i>

7.2.5 Pengujian validasi melihat notifikasi *early warning*

Pada Tabel 7.11 terdapat hasil pengujian melihat notifikasi *early warning*. Pengujian validasi notifikasi *early warning* dilakukan dengan melakukan prosedur seperti pada Tabel 7.11. Hasil yang diharapkan atau *expected result* adalah berhasil menampilkan halaman Notifikasi dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan *valid*.

Tabel 7.11 Pengujian Validasi Melihat notifikasi *early warning*

Kode Kebutuhan	F-KT-05
Nama Kasus Uji	Melihat notifikasi <i>early warning</i>
Prosedur	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan <i>icon</i> Notifikasi untuk melihat notifikasi
Expected Result	Menampilkan halaman Notifikasi <i>Early Warning</i>
Result	Menampilkan halaman Notifikasi <i>Early Warning</i>
Status	<i>Valid</i>

7.2.6 Pengujian validasi mengakses informasi cuaca

Pada Tabel 7.12 terdapat hasil pengujian mengakses informasi cuaca. Pengujian validasi mengakses informasi cuaca dilakukan dengan melakukan prosedur seperti pada Tabel 7.12. Hasil yang diharapkan atau *expected result* adalah berhasil menampilkan halaman Cuaca dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan *valid*.

Tabel 7.12 Pengujian Validasi Melihat informasi cuaca

Kode Kebutuhan	F-KT-06
Nama Kasus Uji	Mengakses informasi cuaca
Prosedur	<ol style="list-style-type: none">1. Pengguna Tekan navigasi menu cuaca
Expected Result	Menampilkan halaman Cuaca
Result	Menampilkan halaman Cuaca
Status	<i>Valid</i>

7.2.7 Pengujian validasi mengakses informasi cuaca maritim

Pada Tabel 7.13 terdapat hasil pengujian mengakses informasi cuaca maritim. Pengujian validasi mengakses informasi cuaca maritim dilakukan dengan melakukan prosedur seperti pada Tabel 7.13. Hasil yang diharapkan atau *expected*

result adalah berhasil menampilkan halaman cuaca maritim dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.13 Pengujian Validasi Melihat informasi cuaca maritim

Kode Kebutuhan	F-KT-07
Nama Kasus Uji	Mengakses informasi cuaca maritim
Prosedur	<ol style="list-style-type: none"> 1. Pengguna menekan navigasi menu cuaca 2. Pengguna menekan <i>tab menu Cuaca Maritim</i> 3. Pengguna memilih lokasi maritim pada <i>maps</i> 4. Pengguna menekan tombol “Pilih lokasi perairan”
Expected Result	Menampilkan halaman cuaca maritim
Result	Menampilkan halaman cuaca maritim
Status	<i>Valid</i>

7.2.8 Pengujian validasi mengubah profil

Pada Tabel 7.14 terdapat hasil pengujian mengubah profil. Pengujian validasi mengubah profil dilakukan dengan melakukan prosedur seperti pada Tabel 7.14. Hasil yang diharapkan atau *expected result* adalah berhasil menyimpan data profil ke *database* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.14 Pengujian Validasi Mengubah profil

Kode Kebutuhan	F-KT-08
Nama Kasus Uji	Mengubah profil
Prosedur	<ol style="list-style-type: none"> 1. Pengguna menekan tab menu <i>Setting</i> 2. Pengguna mengubah nama, email, alamat pada form ubah profil 3. Pengguna menekan tombol “Simpan profil”
Expected Result	Data profil yang diubah tersimpan di <i>database</i>
Result	Data profil yang diubah tersimpan di <i>database</i>
Status	<i>Valid</i>

7.2.9 Pengujian validasi *upload foto profil*

Pada Tabel 7.15 terdapat hasil pengujian *upload foto profil*. Pengujian validasi *upload foto profil* dilakukan dengan melakukan prosedur seperti pada Tabel 7.15. Hasil yang diharapkan atau *expected result* adalah berhasil menyimpan foto profil ke *database* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.15 Pengujian Validasi *Upload foto profil*

Kode Kebutuhan	F-KT-09
Nama Kasus Uji	<i>Upload foto profil</i>
Prosedur	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna tekan <i>icon photo profil</i>3. Pengguna upload photo profil
Expected Result	Foto profil yang diubah tersimpan di <i>database</i>
Result	Foto profil yang diubah tersimpan di <i>database</i>
Status	Valid

7.2.10 Pengujian validasi mengubah *password akun*

Pada Tabel 7.16 terdapat hasil pengujian mengubah *password akun*. Pengujian validasi mengubah *password akun* dilakukan dengan melakukan prosedur seperti pada Tabel 7.16. Hasil yang diharapkan atau *expected result* adalah berhasil menyimpan *password akun* ke *database* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.16 Pengujian Validasi Mengubah *password akun*

Kode Kebutuhan	F-KT-10
Nama Kasus Uji	Mengubah <i>password akun</i>
Prosedur	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna mengubah <i>password baru</i> pada <i>form ubah password</i>3. Pengguna menekan tombol "Ubah <i>password</i>"
Expected Result	<i>Password akun</i> yang diubah tersimpan di <i>database</i>
Result	<i>Password akun</i> yang diubah tersimpan di <i>database</i>
Status	Valid

7.2.11 Pengujian validasi mengubah *logout*

Pada Tabel 7.17 terdapat hasil pengujian mengubah *logout*. Pengujian validasi *logout* dilakukan dengan melakukan prosedur seperti pada Tabel 7.17. Hasil yang diharapkan atau *expected result* adalah pengguna telah berhasil *logout* dari sistem dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.17 Pengujian Validasi *logout*

Kode Kebutuhan	F-KT-11
Nama Kasus Uji	<i>Logout</i>
Prosedur	<ol style="list-style-type: none">1. Pengguna menekan <i>tab menu Setting</i>2. Pengguna menekan <i>icon logout</i>
Expected Result	Pengguna telah <i>logout</i> dari sistem
Result	Pengguna telah <i>logout</i> dari sistem
Status	<i>Valid</i>

7.2.12 Pengujian validasi *Upvote posting*

Pada Tabel 7.18 terdapat hasil pengujian *upvote posting*. Pengujian validasi *upvote posting* dilakukan dengan melakukan prosedur seperti pada Tabel 7.18. Hasil yang diharapkan atau *expected result* adalah berhasil menambah jumlah *upvote posting* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.18 Pengujian Validasi *Upvote posting*

Kode Kebutuhan	F-KT-12
Nama Kasus Uji	<i>Upvote posting</i>
Prosedur	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan <i>icon Beranda</i> untuk masuk ke menu Beranda3. Menekan <i>icon upvote</i> untuk mendukung <i>posting</i> bencana alam
Expected Result	Menambah jumlah <i>upvote posting</i> bencana alam yang telah di-upvote
Result	Menambah jumlah <i>upvote posting</i> bencana alam yang telah di-upvote
Status	<i>Valid</i>

7.2.13 Pengujian validasi *Downvote posting*

Pada Tabel 7.19 terdapat hasil pengujian *downvote posting*. Pengujian validasi *downvote posting* dilakukan dengan melakukan prosedur seperti pada Tabel 7.19. Hasil yang diharapkan atau *expected result* adalah berhasil menambah jumlah *downvote posting* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.19 Pengujian Validasi *Downvote posting*

Kode Kebutuhan	F-KT-13
Nama Kasus Uji	<i>Downvote posting</i>
Prosedur	<ol style="list-style-type: none">1. Melakukan <i>login</i>2. Menekan <i>icon Beranda</i> untuk masuk ke menu Beranda3. Menekan <i>icon downvote</i> untuk mendukung <i>posting</i> bencana alam
Expected Result	Menambah jumlah <i>downvote posting</i> bencana alam yang telah di- <i>downvote</i>
Result	Menambah jumlah <i>downvote posting</i> bencana alam yang telah di- <i>downvote</i>
Status	<i>Valid</i>

7.2.14 Pengujian validasi *Filter posting*

Pada Tabel 7.20 terdapat hasil pengujian *filter posting*. Pengujian validasi *filter posting* dilakukan dengan melakukan prosedur seperti pada tabel 7.20. Hasil yang diharapkan atau *expected result* adalah menampilkan informasi bencana berdasarkan jenis *filter* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.20 Pengujian Validasi *Filter posting*

Kode Kebutuhan	F-KT-14
Nama Kasus Uji	<i>Filter posting</i>
Prosedur	<ol style="list-style-type: none">1. Pengguna menekan dropdown filter posting2. Pengguna memilih jenis filter posting3. Pengguna menekan tombol “Terapkan filter”

Tabel 7.20 Pengujian Validasi *Filter posting* (lanjutan)

Expected Result	Menampilkan informasi bencana berdasarkan jenis <i>filter</i>
Result	Menampilkan informasi bencana berdasarkan jenis <i>filter</i>
Status	<i>Valid</i>

7.2.15 Pengujian validasi Melihat detail informasi bencana

Pada Tabel 7.21 terdapat hasil pengujian melihat detail informasi bencana. Pengujian validasi *filter posting* dilakukan dengan melakukan prosedur seperti pada Tabel 7.21. Hasil yang diharapkan atau *expected result* adalah menampilkan informasi bencana berdasarkan jenis *filter* dan hal tersebut sesuai dengan hasil yang telah diujikan. Maka status pengujian ini dapat dikatakan valid.

Tabel 7.21 Pengujian Validasi Melihat detail informasi bencana

Kode Kebutuhan	F-KT-15
Nama Kasus Uji	Melihat detail informasi bencana
Prosedur	1. Pengguna menuju menu Beranda 2. Pengguna menekan cardview dalam daftar posting bencana alam
Expected Result	Menampilkan detail informasi bencana
Result	Menampilkan detail informasi bencana
Status	<i>Valid</i>

7.3 Pengujian *Usability*

Pengujian *usability* dilakukan untuk mengetahui seberapa mudah dan berguna aplikasi yang telah dikembangkan. Pengujian *usability* dilakukan pada 5 (lima) orang responden pengguna, dikarenakan responden uji yang berjumlah 5 (lima) orang akan dapat menemukan 85% masalah *usability* pada desain dari sebuah sistem (Nielsen, 2000).

7.3.1 Hasil pengujian dan analisis data SUS

Setelah dilakukan pengujian *usability* dan mendapatkan hasil skor dari kuisioner SUS, maka selanjutnya dilakukan analisis data SUS dari kelima responden uji. Berikut Tabel 7.24 yang memuat hasil dari masing-masing kuisioner SUS yang

diisi oleh 5 (lima) responden uji. Salah satu hasil dari pengisian kuisioner SUS dapat dilihat pada Lampiran 1.

Tabel 7.22 Hasil pengisian kuisioner SUS

Nomor pertanyaan	Nama Responden Uji				
	Fajri	Roka	Fathur	Nida	Putri
1	4	3	4	4	4
2	2	2	1	2	2
3	5	4	5	4	4
4	1	2	1	1	2
5	5	5	4	4	5
6	1	1	2	1	2
7	4	5	3	5	4
8	2	2	1	1	2
9	4	4	5	4	4
10	2	1	3	1	2

Setelah dilakukan rekap terhadap 5 (lima) kuisioner responden uji, tahap selanjutnya adalah melakukan skoring dan konversi terhadap hasil tersebut dan berakhir dengan mengalikan setiap skor pertanyaan dengan 2,5. Hasil konversi dan skoring ditunjukkan pada Tabel 7.23.

Tabel 7.23 Hasil konversi kuioner SUS

Nomor pertanyaan	Nama Responden Uji				
	Fajri	Roka	Fathur	Nida	Putri
1	5	3	5	5	5
2	1	2	0	1	1
3	6	4	6	5	5
4	0	2	0	0	1
5	6	5	5	5	6
6	0	1	1	0	1
7	5	5	4	6	5
8	1	2	0	0	1
9	5	4	6	5	5
10	1	1	2	1	1
Jumlah	30	29	29	28	31
Dikali 2,5 (x 2,5)	75	72,5	72,5	70	77,5
Rata-rata nilai					73,5

Dari hasil rekap konversi kuisioner SUS, dapat disimpulkan bahwa aplikasi Kitana memiliki rata-rata nilai sebesar 73,5, dengan kualifikasi “Baik” dan dengan hasil “Berhasil”.

7.4 Pengujian *Performance*

Pengujian *performance* dilakukan untuk mengukur seberapa *real time* performa *push notification* aplikasi Kitana diterima oleh pengguna. Pengujian *performance* dilakukan dengan melakukan *debugging* dan *logging* di Android Studio. Pengujian *performance* dilakukan dengan membandingkan waktu sisi

server dan *client*. Google App Script digunakan sebagai *trigger* untuk mengirim data notifikasi dari sisi *server* ke sisi *client*. Daftar fungsi eksekusi dari sisi pengirim atau *server* dieksekusi dari Google App Script atau *scheduler*. Dapat ditunjukkan dalam Gambar 7.6. Sedangkan hasil waktu eksekusi penerima ditunjukkan dalam Gambar 7.7. Untuk detail dan prosedur pengujian performance pada fitur notifikasi *early warning* (BMKG) ditunjukkan dalam Tabel 7.24.

Tabel 7.24 Prosedur pengujian *performance* Notifikasi *Early Warning*

Nama Kasus Uji	Notifikasi <i>Early Warning</i> (BMKG)
Pra-Kondisi	Aplikasi telah diakses minimal sekali
Lingkungan sistem (<i>client</i>)	<ul style="list-style-type: none"> - Menggunakan koneksi internet di <i>smartphone</i> Samsung A8 (Oreo) - Menggunakan <i>internet provider</i> Telkomsel 4G
Prosedur	<ol style="list-style-type: none"> 1. Membuat kode untuk <i>logging</i> waktu notifikasi yang diterima di Android Studio. 2. Memastikan koneksi internet berjalan di <i>smartphone</i>. 3. Melakukan set <i>timer scheduler endpoint server</i> tiap 1 (satu) menit di Google App Script. 4. Mencatat waktu notifikasi yang diterima oleh pengguna dari hasil <i>logging</i>. 5. Mencatat waktu notifikasi dari <i>timer</i> yang dikirim oleh Google App Script. 6. Menghitung selisih waktu antara notifikasi penerima (<i>client</i>) dan pengirim (<i>server</i>).
Expected Result	Waktu notifikasi yang diterima tidak kurang dari standar <i>real time</i> (20 detik atau lebih).
Result	Waktu notifikasi yang diterima tidak kurang dari standar <i>real time</i> (20 detik atau lebih).

Project	Fungsi	Jenis	Waktu Mulai	Durasi	Status
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.10.54	1.033 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.09.54	0.998 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.08.54	0.967 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.07.54	1.025 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.06.54	1.502 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.05.54	1.218 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.04.54	1.028 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.03.54	0.826 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.02.54	1.02 dtk	Selesai
TestScheduler	cronExecute	Dipicu oleh Waktu	31 Mar 2019 19.01.54	1.017 dtk	Selesai

Gambar 7.4 Daftar fungsi eksekusi dari sisi pengirim (*server*)

```
2019-03-31 19:01:55.428 24763-25046/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:02:55.684 24763-25222/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:03:54.841 24763-25244/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:04:55.718 24763-25265/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:05:56.182 24763-25295/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:06:56.303 24763-25315/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:07:55.652 24763-25336/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:08:55.293 24763-25377/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:09:55.598 24763-25397/com.aryansa.rizqi.kitana D/testRealTime: Early warning
2019-03-31 19:10:55.405 24763-25422/com.aryansa.rizqi.kitana D/testRealTime: Early warning
```

Gambar 7.5 Hasil waktu eksekusi penerima (*client*)

Tabel 7.26 menunjukkan selisih waktu penerimaan notifikasi antara *server* dan *client* dari 10 (sepuluh) kasus uji dan memiliki rata-rata 1,61034 detik. Dari hasil rata-rata tersebut dapat disimpulkan bahwa aplikasi dapat bekerja dengan baik dalam penggunaan *push notification service*, dan dapat dikatakan aplikasi Kitana telah memenuhi kriteria *real time* mengingat salah satu referensi mengatakan dengan sistem peringatan dini, pesan alarm dapat dikirim ke tempat atau kejadian yang kritis di Kota Naples, Italia 20 atau lebih detik sebelum guncangan gempa bumi dimulai (Satriano, Lomax dan Zollo, 2007).

Tabel 7.25 Selisih waktu penerimaan notifikasi antara *server* dan *client*

No.	Waktu pengirim (<i>server</i>)	Waktu penerima (<i>client</i>)	Selisih waktu <i>push notification</i> sampai (detik)
1.	19:01:54	19:01:55.428	1,4279
2.	19:02:54	19:02:55.684	1,6839
3.	19:03:54	19:03:54.841	0,841
4.	19:04:54	19:04:55.718	1,7179
5.	19:05:54	19:05:56.182	2,182
6.	19:06:54	19:06:56.303	2,3029
7.	19:07:54	19:07:55.652	1,652
8.	19:08:54	19:08:55.293	1,2930
9.	19:09:54	19:09:55.598	1,5979
10.	19:10:54	19:10:55.405	1,4049
Rata-rata (detik)			1,6103

7.5 Analisis Pengujian

Setelah dilakukan pengujian yang terdiri dari pengujian unit, validasi, *usability* maka dapat disimpulkan dari keseluruhan pengujian menghasilkan status valid. Status valid artinya keseluruhan analisis kebutuhan sudah memenuhi kebutuhan pengguna dan berjalan dengan semestinya.

7.5.1 Analisis Pengujian Unit

Pengujian *unit* dilakukan untuk menguji tiap komponen atau unit pada suatu sistem untuk memastikan apakah perancangan algoritma sudah berjalan sesuai dengan fungsionalitasnya. Pengujian unit ini yang telah dilakukan pada tiga fungsi yaitu *method postCrowd* kelas *AddPostActivity*, *method callReqBencana* kelas *BerandaFragment* dan *method onListNotifChange* kelas *NotifikasiFragment*. Pengujian unit yang dilakukan pada *method postCrowd* terdapat 7 (tujuh) kasus uji dengan status *valid*, *method callReqBencana* terdapat 2 (dua) kasus uji dengan status *valid* dan *method onListNotifChange* terdapat 2 kasus uji dengan status *valid*.

7.5.2 Analisis Pengujian Validasi

Pengujian validasi termasuk salah satu pengujian *blackbox* yang dilakukan untuk menguji fungsionalitas sistem. Pengujian validasi yang dilakukan untuk menguji semua *scenario* kebutuhan *fungsional* yang dimana pada sistem ini menghasilkan 15 kasus uji. Dari keseluruhan 15 kasus uji yang dilakukan pengujian menghasilkan nilai status *valid* yang dapat diartikan bahwa sistem dapat berjalan sesuai kebutuhan.

7.5.3 Analisis Pengujian *Usability*

Pengujian *usability* merupakan pengujian untuk mengetahui sistem seberapa mudah digunakan dan kepuasaan dari pengguna. Pengujian *usability* dilakukan dengan penyebaran *form* kulanjutanisioner ke 5 (lima) responden yang pernah mengalami kejadian bencana dengan instrumen SUS (*System Usability Scale*). Dari setiap kuisioner terdapat 10 pertanyaan yang dijawab 5 *responden*. Dari tiap pertanyaan memiliki skor untuk mengetahui nilai kepuasan pengguna. Teknik menghitung skor dengan rumus SUS. Pada pengujian *usability* ini menghasilkan rata – rata nilai 73,5 dari 5 responden. Nilai 73,5 merupakan termasuk kualifikasi baik dan dapat dikatakan berhasil. Sehingga aplikasi ini sudah sesuai kebutuhan dan dapat digunakan oleh pengguna.

7.5.4 Analisis Pengujian *Performance*

Pengujian *performance* merupakan pengujian kebutuhan non fungsional. Pengujian *performance* yang dilakukan untuk menguji dan menghitung selisih waktu pengirim (*server*) dengan waktu penerima (*client*) menggunakan 10 (sepuluh) kasus uji. Dari keseluruhan 10 (sepuluh) kasus uji yang dilakukan pengujian *performance* menghasilkan nilai rata – rata 1,6103 detik, dimana dari hasil tersebut dapat diartikan bahwa sistem sudah memenuhi kriteria *real time*.

BAB 8 PENUTUP

8.1 Kesimpulan

Berdasarkan hasil penelitian yang dilakukan dari proses analisis kebutuhan, perancangan, implementasi dan pengujian, maka kesimpulan yang di dapatkan adalah sebagai berikut:

1. Berdasarkan hasil analisis kebutuhan dalam pengembangan aplikasi Kitana, terdapat 15 kebutuhan fungsional yakni *register*, *login*, melihat informasi bencana alam, membuat *posting* bencana alam, melihat notifikasi *early warning*, melihat informasi cuaca, melihat informasi cuaca maritim, mengubah profil, upload foto profil, mengubah *password* akun, *logout*, *filter posting*, *upvote posting*, *downvote posting*, melihat detail informasi bencana alam dan 2 kebutuhan non fungsional yaitu *performance* dan *usability*. Daftar kebutuhan tersebut didapatkan dari wawancara atau interview ke 5 (lima) pengguna dan 1 (satu) *expert* di bidang bencana alam.
2. Perancangan aplikasi Kitana yang dilakukan berdasarkan hasil analisis kebutuhan menghasilkan perancangan arsitektur sistem, *activity diagram*, *sequence diagram*, *class diagram*, perancangan basis data yang meliputi *physical data model*, perancangan *web service*, perancangan antarmuka pengguna berupa *wireframe*, dan perancangan algoritme (komponen). Implementasi dalam pengembangan aplikasi Kitana berdasarkan perancangan berupa algoritme dan *user interface*. Selain itu aplikasi Kitana dikembangkan dengan menggunakan *web service* (API), *push notification service* yang mendukung berjalannya kebutuhan fungsional aplikasi.
3. Terdapat 3 (tiga) pengujian yang terdiri dari 2 (dua) pengujian fungsional dan 1 (satu) pengujian non-fungsional. Pengujian fungsional meliputi pengujian unit (*whitebox*) dan pengujian validasi (*blackbox*). Pengujian unit yang dilakukan pada 11 kasus uji dari 3 *method* dan menghasilkan status valid. Sedangkan pengujian validasi terhadap 15 kasus uji yang sesuai skenario kebutuhan fungsional menghasilkan status valid. Pengujian *usability* dilakukan terhadap 5 pengguna menggunakan instrumen SUS (*System Usability Scale*) dan menghasilkan skor 73,5 dengan kualifikasi “Baik” dan dengan hasil “Berhasil”. Pada pengujian *performance* yang dilakukan pada 10 (sepuluh) kasus uji yang dilakukan pengujian *performance* menghasilkan nilai rata – rata 1,61034 detik, dimana dari hasil tersebut dapat diartikan bahwa sistem sudah memenuhi kriteria *real time*.

8.2 Saran

Saran yang diberikan pada aplikasi “Kitana” untuk proses pengembangan selanjutnya yaitu antara lain:

1. Sistem dapat menambahkan fitur mengubah konten *posting* bencana dari aktor pengguna agar dapat memperbaiki konten *posting* bencana jika ada kesalahan penulisan atau *typo*.
2. Sistem dapat menambahkan fitur *zoom full screen* gambar bencana dari *posting* pengguna agar pengguna lain dapat mengetahui detail foto atau gambar kejadian bencana.

DAFTAR REFERENSI

- BNPB, 2016. Risiko Bencana Indonesia (RBI).
- BNPB, 2018. *Definisi dan Jenis Bencana*. [online] Available at: <<http://bnpb.go.id/definisi-bencana/>> [Accessed 18 Aug. 2018].
- Brooke, J., 2013. SUS: A Retrospective. *Journal of Usability Studies*, [online] 8(2), pp.29–40. Available at: <[http://www.usabilityprofessionals.org/upa_publications/jus/2013feb
ruary/brooke1.html](http://www.usabilityprofessionals.org/upa_publications/jus/2013february/brooke1.html)> <http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html>.
- DIBI-BNPB, 2018. *Data Informasi Bencana Informasi*. [online] Available at: <<http://dibi.bnbp.go.id/dibi/>> [Accessed 18 Aug. 2018].
- Edmunds, A. and Morris, A., 2000. The Problem of Information Overload in Business Organizations : A review of The Literature. [online] Available at: <<https://dl.acm.org/citation.cfm?id=2303775>>.
- EPAM, 2008. Performance Testing. [online] Available at: <<http://compalg.inf.elte.hu/~attila/materials/PerformanceTesting.pdf>>.
- Estellés-arolas, E. and González-ladrón-de-guevara, F., 2012. Towards an Integrated Crowdsourcing Definition Towards an integrated crowdsourcing definition. (May 2014).
- Gahran, A., 2011. *WHAT'S A MOBILE APP?* [online] Available at: <<http://www.contentious.com/2011/03/02/whats-a-mobile-app/>> [Accessed 11 Dec. 2018].
- Google, 2019. *Google App Script*. [online] Available at: <<https://developers.google.com/apps-script/>> [Accessed 9 Apr. 2019].
- Gumelar, S., 2017. *PHP Indonesia Chapter Jakarta*. [online] Available at: <<https://phpid-jakarta.github.io/kulwap/2017/09/30/onesignal-push-notification-sasono-gumelar.html>> [Accessed 9 Apr. 2018].
- Hendry, S., 2015. *Cepat Mahir MySQL dan SQLite*. Elex Media Komputindo.
- Hermawan, S., 2011. *Mudah Membuat Aplikasi Android*. Yogyakarta: ANDI.
- Holdener, A.T., 2011. *HTML5 Geolocation*. O'Reilly.
- Howe, J., 2006. *The Rise of Crowdsourcing*. [online] Available at: <<https://www.wired.com/2006/06/crowds/?pg=1&topic=crowds&topi>>

- c_set=> [Accessed 31 Dec. 2018].
- Hunter, D.S., 2012. eBook Geotagging: Linking Literature and Location. [online] 14, p.14. Available at: <<http://ir.obihiro.ac.jp/dspace/handle/10322/3933>>.
- IDEO.org, 2015. The Field Guide to Human-Centered Design.
- JSON, 2019. *Introducing JSON*. [online] Available at: <<https://www.json.org>> [Accessed 28 Mar. 2019].
- KURNIAJI, R., 2015. *PENERAPAN WEB SERVICE MENGGUNAKAN JSON UNTUK MENDUKUNG RANCANGAN APLIKASI PADA PERGURUAN TINGGI*. AKADEMI MANAJEMEN INFORMATIKA DAN KOMPUTER AMIK RAHARJA INFORMATIKA.
- McConnell, S., 1996. *Rapid Development: Taming Wild Software Schedules*. [online] Redmon, Washington: MicrosoftPress. Available at: <<https://ptgmedia.pearsoncmg.com/images/9781556159008/samplepages/9781556159008.pdf>>.
- Nielsen, J., 2000. *Why You Only Need to Test with 5 Users*. [online] Available at: i<<https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/>> [Accessed 12 Dec. 2018].
- Nielsen, J., 2012. *Usability 101: Introduction to Usability*. [online] Available at: <<https://www.nngroup.com/articles/usability-101-introduction-to-usability/>> [Accessed 31 Dec. 2018].
- OneSignal, 2018. *OneSignal*. [online] Available at: <<https://onesignal.com>> [Accessed 9 Apr. 2019].
- PerfTestPlus, 2006. Introduction to Performance Testing Page 9.
- Permana, I.P.Y., Kharisma, P.A. and Tolle, H., 2015. Rancang Bangun Aplikasi Perangkat Bergerak Pemantau Prakiraan Cuaca Maritim di Perairan Indonesia. [online] 6. Available at: <<http://filkom.ub.ac.id/doro/archives/detail/DR00098201512#>>.
- Peterson, M.P., 2008. International perspectives on maps and the Internet: An introduction. *Lecture Notes in Geoinformation and Cartography*, (9783540720287), pp.3–10.
- Pradipta, A.A., Yuli Adam Prasetyo, ST., M. and Nia Ambarsari, S.Si., M., 2015. Pengembangan Web E-Commerce Bojana Sari Menggunakan Metode Prototype. *eProceedings of Engineering*, [online] 2(1), pp.1042–1056. Available at:

- <<http://libraryproceeding.telkomuniversity.ac.id/index.php/engineering/article/view/2726>>.
- S, R.A. and Shalahuddin, M., 2018. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Obyek*. Bandung, Jawa Barat: Informatika Bandung.
- Satriano, C., Lomax, A. and Zollo, A., 2007. Optimal, real-time earthquake location for early warning. *Earthquake Early Warning Systems*, 8, pp.85–96.
- Sauro, J., 2011. *MEASURING USABILITY WITH THE SYSTEM USABILITY SCALE (SUS)*. [online] Available at: <<https://measuringu.com/sus/>> [Accessed 28 Mar. 2019].
- Sommerville, I., 2011. *SOFTWARE ENGINEERING*. Pearson.
- Toro, R. and Rohman, N., 2018. *Kotlin Android Developer Expert*. Bandung, Jawa Barat: PT. Presentologics.
- UNSDR, 2010. Early Warning Practices can Save Many Lives: Good Practices and Lessons Learned. [online] (August), p.67. Available at: <http://www.unisdr.org/files/15254_EWSBLLfinalweb.pdf>.
- Wijaya, W., Tolle, H. and Kharisma, A.P., 2018. Rancang Bangun Aplikasi Geotagging Social Report Bencana Banjir. 2(7).
- Xu, Z. and Zhu, S., n.d. Abusing Notification Services on Smartphones for Phishing and Spamming.

LAMPIRAN 1 HASIL KUISIONER SUS

1. Nama : Nida Khoiriyyah
Umur : 20
Pekerjaan : Mahasiswi Fakultas Kedokteran Universitas Brawijaya

Tanda Tangan : 

No.	Pertanyaan	Sangat tidak setuju				Sangat Setuju
		1	2	3	4	5
1	Saya merasa bahwa saya akan sering menggunakan aplikasi "Kitana".				✓	
2	Saya merasa bahwa aplikasi "Kitana" rumit.		✓			
3	Saya merasa bahwa aplikasi "Kitana" mudah digunakan.				✓	
4	Saya merasa saya butuh <i>support</i> atau dukungan dari orang teknis untuk bisa menggunakan aplikasi "Kitana" ini.	✓				
5	Saya merasa bahwa berbagai macam fungsi di aplikasi "Kitana" ini				✓	

No.	Pertanyaan	Sangat tidak setuju				Sangat Setuju
		1	2	3	4	5
	terintegrasi dengan baik.					
6	Saya merasa bahwa terlalu banyak ketidakconsistenan di dalam aplikasi "Kitana".	✓				
7	Saya akan membayangkan bahwa sebagian besar orang menggunakan aplikasi "Kitana" ini dengan sangat cepat.					✓
8	Saya merasa bahwa aplikasi "Kitana" sangat rumit untuk digunakan.	✓				
9	Saya sangat percaya diri menggunakan aplikasi "Kitana" ini.				✓	
10	Saya perlu untuk belajar banyak hal sebelum saya menggunakan aplikasi "Kitana" ini.	✓				

LAMPIRAN 2 HASIL WAWANCARA EXPERT

Nama	Drs. Tripim Apriliyanto
Profesi/Jabatan	Kabid Pencegahan dan Kesiapsiagaan (PK) BPBD Kota Malang

Pertanyaan	Jawaban
Tugas Bidang Pencegahan dan Kesiapsiagaan adalah meliputi perencanaan, koordinasi, sosialisasi pada masyarakat wilayah Malang. Bencana yang menjadi ruang lingkup BPBD Malang adalah tanah longsor, banjir, pohon tumbang, puting beliung.	
Bagaimana menurut Bapak/Ibu terkait dengan kejadian bencana alam seperti banjir, tanah longsor, gempa bumi dan tsunami yang saat ini sering terjadi?	Biasanya BPBD melakukan tindakan pencegahan dengan cara mengkomunikasikan kepada masyarakat secara langsung (dengan mengadakan sosialisasi, misalnya). Dalam hal bencana tanah longsor, BPBD juga melakukan sinkronisasi program kerja mereka dengan kelurahan seperti mengadakan perencanaan dan pembangunan pra maupun pasca longsor.
Bagaimana menurut Bapak/Ibu mengenai informasi kebencanaan yang seringkali disebarluaskan melalui jejaring atau media sosial?	Informasi yang didapat dari masyarakat sangatlah penting. Namun juga harus mempertimbangkan beberapa hal dalam menyampaikan informasi, seperti tingkat keaktualitasan informasi tersebut, dan dapat dipertanggung jawabkan. BPBD memiliki 15 orang relawan yang bertugas untuk memantau setiap kelurahan di seluruh wilayah

Pertanyaan	Jawaban
	Malang jika terdapat tanda-tanda atau terjadi bencana.
Apakah Bapak/Ibu mengetahui mengenai crowdsourcing (penghimpunan data dari berbagai sumber) yang kemudian dimanfaatkan untuk dijadikan informasi yang berguna dan penting?	Jawaban yang diutarakan oleh Pak Tripim masih berhubungan dengan pertanyaan sebelumnya. Yakni <i>crowdsourcing</i> menjadi berguna dan penting ketika informasi yang disampaikan sifatnya akuntabel, bisa dipertanggung jawabkan, otentik, dan aktual.
Apakah Bapak/Ibu mengetahui atau pernah menggunakan mengenai aplikasi Info BMKG yang dikembangkan oleh BMKG? Bagaimana peran aplikasi tersebut dalam hal kesiapsiagaan dalam menghadapi bencana alam?	Info BMKG umumnya menyediakan informasi seperti cuaca, curah hujan, kecepatan angin, ketinggian air laut. Pak Tripim mengutarakan bahwa info BMKG juga merupakan aplikasi yang cukup berkontribusi dalam pencegahan bencana alam. Misalnya informasi curah hujan yang dapat mendukung informasi apakah terjadi banjir di titik atau wilayah tertentu di Malang. Salah satu kerjasama BPBD dengan BMKG adalah pihak BMKG memasang alat pengukur ketinggian air sungai untuk mengetahui potensi banjir di daerah Jodipan.
Bagaimana early warning dapat bekerja dalam mendukung kesiapsiagaan bencana? Informasi apa yang diperlukan dalam penyebaran informasi early warning (berupa notifikasi) melalui smartphone?	Sebenarnya di Kelurahan Bareng, Kecamatan Kasin sudah terdapat EWS atau Early Warning System dari USAID yang merupakan agensi independen dari Amerika untuk dilakukannya pencegahan bencana banjir. Karena di wilayah tersebut ketika curah hujan tinggi dan hujan

Pertanyaan	Jawaban
	terjadi lebih dari satu jam, sungai di kelurahan tersebut pasti akan banjir.
Bagaimana adanya fitur early warning notification penting dalam perancangan dan pembuatan aplikasi ini?	Tentu saja akan sangat membantu. Hal tersebut dapat meningkatkan kewaspadaan masyarakat sekitar terhadap bencana alam yang akan terjadi. Informasi existing EWS biasanya disampaikan melalui WhatsApp Group di setiap kelurahan.
Bagaimana menyajikan konten atau informasi yang baik terkait bencana alam gempa bumi dan tsunami? Informasi apa saja yang harus ada didalamnya?	(<i>Karena gempa bumi dengan skala belum masuk ruang lingkup BPBD, maka pertanyaan ini tidak terjawab.</i>)
Bagaimana menyajikan konten atau informasi yang baik terkait bencana alam banjir dan tanah longsor? Informasi apa saja yang harus ada didalamnya?	Berdasarkan laporan bencana alam dari masyarakat sekitar, biasanya informasi yang disampaikan idealnya adalah lokasi, dan kondisi atau situasi wilayah yang sedang terdampak. Masyarakat paling sering melapor melalui Hotline BPBD (telepon). Selain itu juga BPBD membuat Forum PRB atau Pengurangan Risiko Bencana.

LAMPIRAN 3 HASIL WAWANCARA PENGGUNA

Nama	Fajri Fernanda
Umur	20 tahun
Profesi/Instansi	Mahasiswa FILKOM/Universitas Brawijaya
Asal	Padang, Sumatera Barat
Rumah tinggal	Pekanbaru, Riau
Domisili saat ini	Kota Malang
Saran atau hal lain yang disampaikan di luar pertanyaan interview	<p>Fajri tidak hanya menceritakan pengalaman ketika mengalami bencana banjir, namun juga gempa bumi dan isu tsunami. Fajri sering memanfaatkan Twitter BMKG sebagai sarana informasi terkait dengan hal kebencanaan. Namun karena ia mengakses Twitter lewat browser <i>smartphone</i>, notifikasi terkini mengenai informasi tersebut tidak didapatkan. Ia juga menyatakan bahwa mengapa tidak memakai aplikasi info BMKG saja, karena memori internal <i>smartphone</i> nya tidak memenuhi untuk instalasi aplikasi tersebut.</p> <p>Ia juga menceritakan sebaiknya informasi tanah longsor juga dapat diakses informasinya. Karena Fajri pernah mendapatkan informasi bahwa karena bencana tanah longsor, salah satu jalan/rute ke suatu tempat di daerahnya ditutup. Informasi semacam itu sangat berguna bagi orang-orang yang ingin melakukan perjalanan ke suatu tempat namun ternyata jalan atau rutenya tidak dapat diakses, jadi mereka dapat menggunakan rute lainnya.</p>

	Fajri menyatakan bahwa biasanya ia merasakan atau mengalami bencana terlebih dahulu, lalu ia melihat atau menerima informasi mengenai bencana tersebut (melalui sosial media atau media yang lain).
--	---

LAMPIRAN 4 SKENARIO USABILITY TESTING

BACA INSTRUKSI DENGAN BAIK	
Pada bagian ini Anda diminta untuk menjalankan aplikasi berdasarkan instruksi yang disediakan.	
Anda diminta untuk melakukan <i>walkthrough</i> aplikasi secara langsung dan memberikan <i>feedback</i> (jika ada) setelah selesai melakukan <i>walkthrough</i> aplikasi “Kitana”.	

Kode tugas	Nama tugas
T1	Melakukan <i>registrasi</i> akun baru
T2	Melakukan <i>login</i>
T3	Melihat informasi bencana alam
T4	Membuat <i>posting</i> mengenai bencana alam yang sedang terjadi
T5	<i>Upvote posting</i> pengguna lain
T6	<i>Downvote posting</i> pengguna lain
T7	Melaporkan <i>posting</i> pengguna lain
T8	Mengakses informasi cuaca
T9	Mengakses informasi cuaca maritim
T10	Mengakses menu notifikasi
T11	Membagikan <i>post</i> ke sosial media lain
T12	Mengubah <i>profile</i>
T13	<i>Logout</i>

SELESAI