

**PENGEMBANGAN APLIKASI MOBILE PADA
PEMANTAUAN LINGKUNGAN LAUT DENGAN
DYNAMIC SYSTEM DEVELOPMENT METHOD
(STUDI KASUS TELUK KILUAN)**

TUGAS AKHIR

Diajukan sebagai syarat menyelesaikan jenjang strata Satu (S-1)
di Program Studi Teknik Informatika, Jurusan Teknologi,
Produksi dan Industri, Institut Teknologi Sumatera

Oleh:

ABDURRACHMAN FARRAS

119140052



**PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI, PRODUKSI DAN INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
LAMPUNG SELATAN**

2023

LEMBAR PENGESAHAN

Tugas Akhir dengan judul “Pengembangan Aplikasi Mobile Pada Pemantauan Lingkungan Laut Dengan *Dynamic System Development Method* (Studi Kasus Teluk Kiluan)” adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya, baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Institut Teknologi Sumatera maupun di institusi pendidikan lainnya.

Lampung Selatan,
Penulis,

PHOTO
BERWARNA

Abdurrachman Farras
NIM. 119140052

Diperiksa dan disetujui oleh,

Pembimbing

Tanda Tangan

1. Andika Setiawan, S.Kom., M.Cs.
NIP. 19911127 2022 03 1 007
2. Eko Dwi Nugroho, S.Kom., M.Cs.
NRK. 1991020 2020 1 279

.....
.....

Penguji

Tanda Tangan

1. Ilham Firman Ashari, S.Kom., M.T.
NIP. 19930314 201903 1 018
2. Winda Yulita, S.Pd., M.Cs.
NIP. 19930727 2022 03 2 022

.....
.....

Disahkan oleh,
Koordinator Program Studi Teknik Informatika
Jurusan Teknologi, Produksi dan Industri
Institut Teknologi Sumatera

Andika Setiawan, S.Kom., M.Cs.
NIP. 19911127 2022 03 1 007

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir dengan judul “Pengembangan Aplikasi Mobile Pada Pemantauan Lingkungan Laut Dengan *Dynamic System Development Method* (Studi Kasus Teluk Kiluan)” adalah karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan benar.

Nama : Abdurrachman Farras

NIM : 119140052

Tanda Tangan :

Tanggal : 05 Juni 2023

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Institut Teknologi Sumatera, saya yang bertanda tangan di bawah ini:

Nama : Abdurrachman Farras
NIM : 119140052
Program Studi : Teknik Informatika
Jurusan : Jurusan Teknologi, Produksi dan Industri
Jenis Karya : Tugas Akhir

demikian demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Sumatera **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty Free Right*) atas karya ilmiah saya yang berjudul:

“Pengembangan Aplikasi Mobile Pada Pemantauan Lingkungan Laut Dengan *Dynamic System Development Method* (Studi Kasus Teluk Kiluan)”

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini, Institut Teknologi Sumatera berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Lampung Selatan

Pada tanggal 05 Juni 2023

Yang menyatakan,

Abdurrachman Farras

KATA PENGANTAR

Puji syukur kehadiran Allah SWT atas limpahan rahmat, karunia, serta petunjuk-Nya sehingga penyusunan tugas akhir ini telah terselesaikan dengan baik. Dalam penyusunan tugas akhir ini penulis telah banyak mendapatkan arahan, bantuan, serta dukungan dari berbagai pihak. Oleh karena itu pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Prof. Dr. I Nyoman Pugeg Aryantha
2. Hadi Teguh Yudistira, S.T., Ph.D.
3. Andika Setiawan S.Kom., M.Cs sebagai dosen pembimbing 1.
4. Eko Dwi Nugroho, S.Kom., M.Cs sebagai dosen pembimbing 2.
5. Ilham Firman Ashari, S.Kom., M.T sebagai dosen penguji 1.
6. Winda Yulita, S.Pd., M.Cs sebagai dosen penguji 2.
7. Terima kasih kepada ibu (Samsuarni) dan ayah (Herman) penulis karena sudah mendukung anakmu ini sampai dengan sarjana dari segi doa, kasih sayang, material, waktu, dan bahkan tenaga. Terima kasih banyak telah mendidik anakmu ini sehingga menjadi pribadi yang sekarang. Setetes keringat yang kalian keluarkan untukku tidak akan bisa aku gantikan dengan material apapun. Tanpa dukungan dari kalian, mungkin anakmu ini hanyalah bisa menjadi seorang pecundang yang tidak tau apa-apa.
8. Dzaki Khothir dan Pathina Annisa sebagai adik sekaligus salah satu alasan untuk abang mu ini berjuang untuk menjadi sukses.
9. Terima kasih kepada Aprilia Purwanto, Muhammad Firdaus sati, Gibran Basyayef, Ridho Liwardana, Dhifaf Athiyah Zhabiyah, M. Ariefuddin Satria Dharma, Mayang Hermanda Anggraini, Geizka Rozilia Ruicosta, Della Salsabila, Eliza Maharani, Apri Kurniawansyah yang telah mendukung dari segi penulisan tugas akhir sampai dengan kuliah sehari-hari
10. Terima kasih kepada Rekan Lomba dan Proyek selama kuliah Ilham Novrianda, Agung, Mega Muli Utami, David Yakob, Dani, Rafi Arya Nugraha, Eko Santoso, Aqsal, Ardian, Asyraf Nur Ramli.
11. Terima kasih kepada dosen-dosen dan rekan-rekan tim SFAD.

12. Terima kasih kepada pengelola wisata Teluk Kiluan, Nelayan Teluk Kiluan dan pemilik *Wi-Fi* gratis selama 3 menit yang sangat berjasa untuk mengumpulkan berkas sidang akhir di menit-menit sebelum tutup.
13. Terima kasih kepada dosen Teknik Biosistem Zunanik Mufidah dan mas Kharis yang telah menjadi mentor sekaligus penasihat, sekaligus contoh baik dalam segi kerja keras, usaha, keyakinan, dan ketekunan.
14. Terima kasih kepada mama Lely, teteh Ratu, kak Verdi, bu Tini, kak puput, Novel Ali Akbar.

Akhir kata penulis berharap semoga tugas akhir ini dapat memberikan manfaat bagi kita semua, amin.

RINGKASAN

Pengembangan Aplikasi Mobile Pada Pemantauan Lingkungan Laut Dengan Dynamic System Development Method (Studi Kasus Teluk Kiluan)

Abdurrachman Farras

Teluk Kiluan memiliki risiko bencana alam dengan tingkat sedang dengan risiko penyusunnya adalah banjir, gempa bumi, badai dan tsunami. Pada tahun 2018, telah terjadi tsunami yang mengakibatkan korban jiwa, korban luka-luka, dan banyaknya properti yang rusak. Berdasarkan survei lapangan yang sudah dilakukan oleh pihak klien penelitian kepada masyarakat lokal, didapatkan permasalahan dimana masyarakat di sana memiliki masalah penangkapan ikan yang terganggu akibat cuaca buruk atau bencana laut seperti angin kencang dan badai laut. Masyarakat di sana tidak mengetahui kondisi laut yang aman untuk menangkap ikan. Di lain sisi terdapat beberapa peneliti yang ingin melakukan penelitian di Teluk Kiluan, akan tetapi dibatasi oleh data dari kondisi laut itu sendiri. sehingga masyarakat Teluk Kiluan dan para peneliti yang ingin meneliti Teluk Kiluan memerlukan sistem yang dapat memberikan informasi pemantauan kondisi lingkungan laut Teluk Kiluan agar para nelayan dapat berjaga-jaga akan adanya bahaya laut seperti badai laut saat penangkapan ikan dan peneliti bisa mendapatkan data lingkungan laut untuk penelitian lebih lanjut.

Tanda-tanda badai sendiri adalah rendahnya suhu udara, tingginya gelombang air laut, kuatnya kecepatan gelombang, serta kuatnya kecepatan angin. Badai laut dapat terjadi diakibatkan karena tingginya gelombang laut, tidak stabilnya panjang gelombang dan periode gelombang serta tingginya kecepatan angin. Untuk itu dibutuhkan sistem yang dapat memberikan informasi kepada nelayan agar mereka dapat mendapatkan informasi keadaan laut saat sebelum berlayar serta memberikan data lingkungan laut kepada para peneliti. Oleh karena itu dibuatlah aplikasi *mobile android* yang dapat memudahkan nelayan dan peneliti untuk melihat kondisi lingkungan laut. Perangkat *mobile* dipilih dikarenakan ringan, sehingga mudah untuk dibawa ke mana-mana.

Aplikasi *mobile* yang dikembangkan sebagai media penyelesaian masalah dikembangkan menggunakan salah satu metode *Software Development Life Cycle* (SDLC) yakni *Dynamic Systems Development Method* (DSDM). DSDM digunakan berdasarkan hasil studi komparasi antara beberapa metode SDLC yang ada dan memiliki kelebihan yang cocok dengan permasalahan yang ada. Selain itu, metode ini memiliki sedikit kekurangan sehingga proses pengembangan dapat berjalan sebagaimana mestinya

Setelah aplikasi berhasil dikembangkan dengan menggunakan metode DSDM, dilakukan pengujian kepada pihak peneliti dan pihak nelayan dengan menggunakan metode kuesioner. Kuesioner dibagikan ke pihak nelayan dengan pertanyaan yang bersifat membantu nelayan dalam berlayar atau mencari ikan di laut, serta dibagikan kepada para peneliti dengan pertanyaan yang bersifat membantu peneliti dalam melihat data lingkungan laut Teluk Kiluan.

Didapatkan hasil rata-rata pada kuesioner kepada pihak peneliti adalah sebesar 79.3%. Dari hasil yang didapat dapat dianalisis bahwa aplikasi *mobile* berbasis android ini membantu peneliti melihat mengumpulkan data kondisi laut seperti yang diharapkan. Pada pengujian kepada nelayan, didapatkan hasil rata-rata skor 84% sehingga dapat disimpulkan bahwa aplikasi dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan.

ABSTRAK

Pengembangan Aplikasi Mobile Pada Pemantauan Lingkungan Laut Dengan
Dynamic System Development Method (Studi Kasus Teluk Kiluan)

Abdurrachman Farras

Berdasarkan survei lapangan yang sudah dilakukan oleh pihak klien penelitian kepada masyarakat lokal, didapatkan permasalahan dimana masyarakat di sana memiliki masalah penangkapan ikan yang terganggu akibat cuaca buruk atau bencana laut seperti angin kencang dan badai laut. Di lain sisi terdapat beberapa peneliti yang ingin melakukan penelitian di Teluk Kiluan, akan tetapi dibatasi oleh data dari kondisi laut itu sendiri. Dibuatkan aplikasi *mobile android* yang dapat memudahkan nelayan dan peneliti untuk melihat kondisi lingkungan laut. Perangkat mobile dipilih dikarenakan ringan, sehingga mudah untuk dibawa ke mana-mana. Aplikasi mobile yang dikembangkan sebagai media penyelesaian masalah dikembangkan menggunakan salah satu metode *Software Development Life Cycle* (SDLC) yakni *Dynamic systems development method* (DSDM). Setelah aplikasi berhasil dikembangkan dengan menggunakan metode DSDM, dilakukan pengujian kepada pihak peneliti dan pihak nelayan dengan menggunakan metode kuesioner. Didapatkan hasil rata-rata kuesioner kepada pihak peneliti adalah sebesar 79.3%. Serta didapatkan hasil pengujian dengan nelayan sebesar 84%.

Kata Kunci: Teluk Kiluan, Cuaca, DSMS, Kuesioner, Peneliti, Nelayan

ABSTRAK

Mobile Application Development for Marine Environment Monitoring Using Dynamic System Development Method (Teluk Kiluan Case Study)

Abdurrachman Farras

Based on a field survey conducted by a research client on the local community, it was found that the community there had fishing problems that were disrupted due to bad weather or sea disasters such as strong winds and sea storms. On the other hand, there are several researchers who wish to conduct research in Kiluan Bay, but are constrained by data from the sea entrepreneurs themselves. An android mobile application was created that can make it easier for fishermen and researchers to see the condition of the marine environment. Mobile devices are chosen because they are lightweight so they are easy to carry anywhere. The mobile application developed as a problem-solving media was developed using the Software Development Life Cycle (SDLC) method, namely the Dynamic system development (DSDM) method. After the application was successfully developed using the DSDM method, researchers and fishermen were tested using the questionnaire method. The average result obtained from questionnaires to researchers is 79.3%. As well as the results of testing with fishermen by 84%.

Keywords: Kiluan Bay, Weather, DSMS, Questionnaire, Researcher, Fishermen

DAFTAR ISI

LEMBAR PENGESAHAN	ii
HALAMAN PERNYATAAN ORISINALITAS	iii
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS	iv
KATA PENGANTAR	v
RINGKASAN	vii
ABSTRAK	ix
ABSTRAK	x
DAFTAR ISI	xi
DAFTAR TABEL	xv
DAFTAR GAMBAR	xvi
DAFTAR RUMUS	xviii
DAFTAR LAMPIRAN	xix
BAB I PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	5
1.3 Tujuan Penelitian	5
1.4 Batasan Masalah	5
1.5 Manfaat Penelitian	6
1.6 Sistematika Penulisan	6
BAB II TINJAUAN PUSTAKA	7
2.1. Tinjauan Pustaka	7
2.2. Dasar Teori	13

2.2.1. Dynamic System Development Method (DSDM).....	13
2.2.2. Bencana Alam Laut	14
2.2.3. Unified Modeling Language (UML)	15
2.2.4. <i>Android</i>	17
2.2.5. React Native.....	18
2.2.6. <i>Black Box Testing</i>	19
2.2.7. <i>Kuesioner</i>	19
2.2.8. <i>Internet of Things</i> (IOT)	21
2.2.9. <i>Firebase</i>	21
2.2.10. <i>Express JS</i>	22
2.2.11. <i>MongoDB</i>	22
2.2.12. Uji Validitas dan Reliabilitas	22
BAB III METODE PENELITIAN	24
3.1. Alur Penelitian	24
3.2. Penjabaran Langkah Penelitian.....	24
3.2.1. Studi Literatur	25
3.2.2. Penerapan <i>Dynamic System Development Method</i> (DSDM).....	25
3.2.3. Pengujian	25
3.3. Alat dan Bahan Tugas Akhir	27
3.3.1. Alat.....	27
3.3.2. Bahan	27
3.4. Metode Pengembangan/Metode Pengukuran	28
3.4.1. Studi Kelayakan (<i>Feasibility Study</i>)	29
3.4.2. Studi Bisnis (Business Study).....	29
3.4.2.1. Karakteristik Pengguna.....	31
3.4.2.2. Kebutuhan Fungsional	31

3.4.2.3. Kebutuhan Non-Fungsional	32
3.4.2.4. Diagram Sistem Arsitektur	33
3.4.1. <i>Functional Model Iteration</i>	33
3.4.3.1. Identify Functional Prototype	34
3.4.3.2. Agree Plan.....	34
3.4.3.3. Create Functional Prototype	34
3.4.3.4. Review functional prototype.....	47
3.4.4. <i>Design and Build</i>	47
3.4.4.1. Identify Design Prototype	47
3.4.4.2. Agree Plan.....	47
3.4.4.3. Create Design Prototype	47
3.4.4.4. Review Design Prototype	48
3.4.5 <i>Implementation</i>	52
3.4.5.1 Review Business Aspects	52
3.4.5.2 Client Approve & Guidelines	53
3.4.5.3 Train Klein.....	53
3.4.5.4 Implement	53
BAB IV	54
HASIL DAN PEMBAHASAN	54
4.1 Hasil Penelitian	54
4.1.1. Hasil <i>Design and Build</i>	54
4.1.1.1. Hasil Identify Design Prototype	54
4.1.1.2. Hasil Agree Plan	56
4.1.1.3. Hasil Create Design Prototype.....	56
4.1.2.4. Hasil Review Design Prototype.....	67
4.1.2. Hasil <i>Implementation</i>	67

4.1.2.1. Hasil Review Business Aspects	68
4.1.2.2. Hasil Client Approval and Guidelines	71
4.1.2.3. Hasil <i>Train Users</i>	72
4.1.2.4. Hasil <i>Implements</i>	73
4.2 Hasil Pengujian	74
4.2.1 Hasil Pengujian Fungsional	74
4.2.2 Pengujian Kuesioner	81
4.2.2.1 Hasil Pengujian Kepada Pihak Peneliti	81
4.2.2.1 Hasil Pengujian Kepada Pihak Nelayan	84
4.3 Pembahasan	87
BAB V	89
KESIMPULAN DAN SARAN	89
5.1. Kesimpulan	89
5.2. Saran	89
DAFTAR PUSTAKA	90
Lampiran	96

DAFTAR TABEL

Tabel 2.1 Tinjauan Pustaka.....	8
Tabel 2.2 Komponen Use Case Diagram.....	15
Tabel 2.3 Conceptual Data Modeling (CDM)	16
Tabel 2.4 Komponen Activity Diagram	17
Tabel 2.5 Skor Skala Likert	19
Tabel 2.6 Bobot Hasil Skala Likert	20
Tabel 2.7 Level Significance	23
Tabel 3.1 Kuesioner Untuk Peneliti.....	26
Tabel 3.2 Kuesioner Untuk Nelayan Teluk Kiluan	26
Tabel 3.3 Hasil Wawancara Studi Bisnis.....	29
Tabel 3.4 Karakteristik Pengguna.....	31
Tabel 3.5 Rancangan Kebutuhan Fungsional	32
Tabel 3.6 Rancangan Kebutuhan Non-Fungsional	32
Tabel 3.7 Rancangan Pengujian <i>Dashboard</i>	48
Tabel 3.8 Rancangan Pengujian Login	49
Tabel 3.9 Rancangan Pengujian Register	50
Tabel 3.10 Rancangan Pengujian Detail Data	50
Tabel 4.1 <i>Timeline</i> Pekerjaan.....	55
Tabel 4.2 Validasi Level Keamanan Laut	64
Tabel 4.3 Pengujian Data Lingkungan Laut	74
Tabel 4.4 Pengujian <i>Form Login</i>	75
Tabel 4.5 Pengujian <i>Form Register</i>	76
Tabel 4.6 Pengujian Tampilan Grafik dan Rata-Rata Lingkungan Laut	77
Tabel 4.7 Pengujian Arah Angin	79
Tabel 4.8 Pengujian Tampilan Profil.....	80
Tabel 4.9 Pengujian Tampilan Perkiraan Cuaca.....	81
Tabel 4.10 Hasil Kuesioner Kepada Peneliti	82
Tabel 4.11 Hasil Validasi Kuesioner Peneliti	83
Tabel 4.11 Hasil Kuesioner Kepada Nelayan	84
Tabel 4.11 Hasil Validasi Kuesioner Peneliti	86

DAFTAR GAMBAR

Gambar 2.1 DSDM Circle [14].....	13
Gambar 3.1 Alur Penelitian	24
Gambar 3.2 Alur DSDM.....	28
Gambar 3.3 Diagram Arsitektur	33
Gambar 3.4 Rancangan CDM Firebase	35
Gambar 3.5 Rancangan CDM Server	35
Gambar 3.6 Rancangan Algoritma	36
Gambar 3.7 Diagram Use Case.....	37
Gambar 3.8 Activity Diagram Dashboard	38
Gambar 3.9 Activity Diagram Login	39
Gambar 3.10 Activity Diagram Register	40
Gambar 3.11 Activity Diagram Detail Lingkungan Laut	41
Gambar 3.12 Rancangan Halaman Login	42
Gambar 3.13 Rancangan Halaman Register	43
Gambar 3.14 Rancangan Halaman Dashboard	44
Gambar 3.15 Rancangan Halaman Detail.....	45
Gambar 3.16 Rancangan Halaman Arah Angin	46
Gambar 4.1 Detail Pengeluaran	55
Gambar 4.2 Pembuatan Database Dengan Firebase	56
Gambar 4.3 Model Database MongoDB	58
Gambar 4.4 Pembuatan API Input IoT	59
Gambar 4.5 Fungsi Pengelolaan Data Rata-Rata.....	60
Gambar 4.6 Fungsi Grafik	61
Gambar 4.7 Contoh Get API.....	62
Gambar 4.8 Tampilan Halaman Beranda	63
Gambar 4.9 Pengambilan Data	64
Gambar 4.10 Fungsi Pengambilan Data Keamanan	64
Gambar 4.11 Halaman Login Dan Register.....	65
Gambar 4.12 Halaman Detail Data Lingkungan Laut	66
Gambar 4.13 Halaman Detail Arah Angin	66
Gambar 4.14 Code Visualisasi Arah Angin.....	67

Gambar 4.15 Halaman Ramalan Cuaca.....	69
Gambar 4.16 Perubahan Tampilan Baranda	69
Gambar 4.17 Tampilan Halaman Informasi	70
Gambar 4.18 Penambahan Icon Informasi	71
Gambar 4.19 Playstore Aplikasi	72
Gambar 4.20 Pelatihan Sederhana Kepada Komunitas Nelayan Teluk Kiluan...	72
Gambar 4.21 Pelatihan Sederhana Kepada Peneliti.....	73
Gambar 4.22 Bukti Aplikasi Sudah Dirilis	73
Gambar 4.23 Correlations Pada Kuesioner Peneliti	83
Gambar 4.24 Relibility Statics Pada Kuesioner Peneliti	84
Gambar 4.25 Correlations Pada Kuesioner Nelayan	86
Gambar 4.24 Relibility Statics Pada Kuesioner Nelayan	87

DAFTAR RUMUS

Rumus (2.1) Skala Likert.....	20
-------------------------------	----

DAFTAR LAMPIRAN

Lampiran I Kode Pada Mobile.....	96
Lampiran II Kode Pada Back-End.....	142
Lampiran III Dokumentasi Dengan Klien	168
Lampiran IV Dokumen Persetujuan Dengan Klien	170
Lampiran V Hasil Kuesioner Kepada Peneliti dan Nelayan.....	177
Lampiran VI Hasil User Guide	199
Lampiran VII Bukti Fuzzy Logic	208
Lampiran VIII Dokumentasi Pengujian.....	209

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Teluk Kiluan objek wisata laut yang terletak pada Kecamatan Kelumbayan, Kabupaten Tanggamus, Provinsi Lampung. Teluk Kiluan terkenal akan wisata lumba-lumba serta memiliki banyak populasi ikan yang menjadikan mayoritas masyarakat disana berprofesi sebagai nelayan [1]. Menurut riset yang telah dilakukan oleh Tiwi [2], Teluk Kiluan memiliki risiko bencana alam dengan tingkat sedang dengan risiko penyusunnya adalah banjir, gempa bumi, badai dan tsunami. Pada tahun 2018, telah terjadi tsunami yang mengakibatkan korban jiwa, korban luka-luka, dan banyaknya properti yang rusak [3]. Berdasarkan Inayah et al. [4] Banyaknya jumlah korban yang terjadi akibat bencana alam laut didasari dari kurangnya pengetahuan tentang bencana alam serta kurangnya informasi yang didapat mengenai laut itu sendiri.

Berdasarkan survei lapangan yang sudah dilakukan oleh pihak klien kepada masyarakat lokal, didapatkan permasalahan dimana masyarakat di sana memiliki masalah penangkapan ikan yang terganggu akibat cuaca buruk atau bencana laut seperti angin kencang dan badai laut. Masyarakat di sana tidak mengetahui kondisi laut yang aman untuk menangkap ikan sehingga masyarakat Teluk Kiluan memerlukan sistem yang dapat memberikan informasi pemantauan kondisi lingkungan laut Teluk Kiluan agar para nelayan dapat berjaga-jaga akan adanya bahaya laut dalam penangkapan ikan. Sementara itu, dari pihak klien atau peneliti yang ingin meneliti Teluk Kiluan merasa kesulitan dalam mendapatkan data lingkungan laut Teluk Kiluan. Mereka membutuhkan sebuah sistem yang dapat merekam kondisi lingkungan laut untuk melakukan penelitian lebih lanjut seperti penelitian perkiraan badai ataupun bencana alam laut lainnya. Tanda-tanda dari badai itu sendiri adalah rendahnya suhu udara, tingginya gelombang air laut, kuatnya kecepatan gelombang, serta kuatnya kecepatan angin [5]. Badai laut dapat terjadi diakibatkan karena tingginya gelombang laut, tidak stabilnya panjang gelombang dan periode gelombang serta tingginya kecepatan angin [6] badai laut juga dapat

mengakibatkan angin topan yang dimana angin topan sendiri merupakan pusaran angin dengan kecepatan 120 Km/jam atau lebih yang dapat menyebabkan gelombang laut yang kuat [5]. Untuk itu dibutuhkan data dari lingkungan laut berupa suhu udara, tinggi gelombang, kecepatan angin, dan kecepatan gelombang laut untuk dapat diolah menjadi informasi keadaan laut yang aman atau tidak. Data-data tersebut juga dapat disimpan dan diolah agar kelak dapat digunakan untuk penelitian lebih lanjut oleh para peneliti yang ingin meneliti lingkungan laut Teluk Kiluan.

Dalam sistem pemantauan, dibutuhkan sensor-sensor yang dapat mengambil data dari laut Teluk Kiluan. *Internet of Things (IoT)* merupakan sebuah objek atau benda yang terkoneksi dengan internet dan dapat terhubung dengan media lain [7]. Oleh karena itu dibutuhkan sistem *Internet of Things (IoT)* yang dapat mengirimkan data kondisi laut. Dibutuhkan media yang cocok dalam penerimaan data agar dapat memudahkan dalam pemantauan. Perangkat *mobile* merupakan perangkat yang ringan sehingga mudah untuk dibawa ke mana-mana serta dapat digunakan dalam berkomunikasi dan kolaborasi serta dapat dimasukkan berbagai aplikasi yang dapat memudahkan pekerjaan manusia [8]. Pada tahun 2018 sampai dengan 2020, telah tercatat sebanyak 74.95% pengguna *smartphone* menggunakan sistem operasi *android* [9]. Oleh karena itu aplikasi *mobile* berbasis *android* cocok digunakan untuk melakukan pemantauan kondisi laut Teluk Kiluan secara berkala.

Dalam pengembangan sistem atau aplikasi perangkat lunak, dibutuhkan sebuah metode pengembangan perangkat lunak atau *Software Development Life Cycle (SDLC)* agar mempermudah pengembang dalam pengembangan sistem tersebut. Berdasarkan penelitian yang Lawal et al. [10] telah dilakukan perbandingan dua metode yaitu metode *waterfall* dan *agile*, dimana pada metode *waterfall* diperlukan kerjasama yang tinggi antar tim serta pengembangan pada metode ini bersifat *one by one* sehingga dapat meminimalisir kesalahan akan tetapi metode ini dianggap tidak stabil dikarenakan tidak dapat diuji pada hampir tahap ketidakmampuan untuk mengidentifikasi tahap pengembangan masalah, rawan kekurangan kapasitas dikarenakan pengkodean sistem harus selesai sebelum kita bisa melakukannya menguji kinerja perangkat lunak serta membutuhkan biaya yang tinggi dikarenakan komponen dan persyaratan hilang ke tidak konsisten dapat diidentifikasi pada

tahap desain dan pengkodean yang meningkatkan kebutuhan, sementara metode *agile* yang bersifat fleksibel akan terjadi perubahan, kemajuan pengembangan ditentukan oleh kinerja perangkat lunak, serta cocok dijadikan pengembangan secara berkelanjutan.

Berdasarkan hasil diskusi yang dilakukan dengan pihak klien, pembuatan sistem membutuhkan biaya yang seminimal mungkin dalam pengembangan aplikasi, dapat terjadi perubahan pada sistem kapan pun, serta terdapat rencana untuk melakukan pengembangan berkelanjutan, serta akan dilakukan sosialisasi mengenai teknologi dan aplikasi kepada para nelayan di Teluk Kiluan. Dari hasil wawancara tersebut dapat disimpulkan bahwa metode dalam pengembangan aplikasi yang cocok dalam penelitian pengembangan aplikasi *mobile* ini adalah *agile*. Islam et al. [11] Telah dilakukan perbandingan *framework* pada metode *agile*, dimana mereka membandingkan metode *Extreme Programming (XP)*, *Scrum*, dan *Dynamic Systems Development Method (DSDM)*. Pada metode XP berfokus pada *performance* dari *code* program yang selalu diuji dengan menggunakan pengujian otomatis dari *code* program, akan tetapi tidak memiliki kemampuan untuk mendukung tim yang didistribusikan karena berfokus pada komunitas dan *co-location* serta praktik keterlibatan pelanggan yang sebenarnya efektif tetapi membuat stres, dan menghabiskan waktu [12]. Pada penelitian ini dirasa kurang cocok dengan metode pengembangan XP, dikarenakan perlunya kerja sama tim antara pengembang aplikasi dan pembuat alat IoT dalam pengelolaan data. Metode *Scrum* merupakan metode yang mengutamakan kecepatan dan kerja sama yang sangat tinggi antar *developer* akan tetapi dalam metode ini memungkinkan terjadinya pelanggaran tanggung jawab dimana *developer* yang satu dapat mengerjakan tugas *developer* lainnya, metode ini juga tidak ada panduan aturan dalam kerja sama dalam pengembangan [13]. Pada penelitian ini dirasa tidak cocok dengan metode *Scrum* dikarenakan *role* pekerjaan dalam tim sudah ditentukan di awal dan memiliki tanggung jawabnya masing-masing untuk menyelesaikan tugasnya. Metode DSDM mengutamakan keterlibatan klien dan pengembang secara berkesinambungan secara berulang, serta tanggap dengan perubahan dalam pembuatan sistem perangkat lunak secara tepat waktu dan tepat anggaran [14] Akan tetapi pada metode ini pihak klien dapat mengganti kebutuhan sistem menjadi apa pun sehingga dapat terjadi

pengulangan desain sistem kapan pun [11]. Metode DSDM merupakan metode yang cocok digunakan dalam penelitian ini dikarenakan pada penelitian ini menekankan komunikasi antar klien dan pengembang, berdasarkan diskusi kepada pihak klien, tampilan sistem atau aplikasi dapat berganti kapan pun akan tetapi tetap dilakukan diskusi dengan pihak pengembang untuk menyesuaikan waktu pengembangan sehingga dapat menutupi kelemahan pada DSDM itu sendiri.

Pada penelitian yang telah dilakukan oleh Dewi et al. dalam pembuatan sistem informasi menggunakan metode *Dynamic System Development Method* (DSMS) [15] metode tersebut dipakai karena aplikasi yang dikembangkan berdasarkan kebutuhan serta komunikasi antar pengguna dan pengembang sehingga mempermudah dalam segi implementasi aplikasi. Pada penelitian ini juga para penulis menggunakan metode *blackbox testing* sebagai metode pengujian dalam pembuatan sistem informasi untuk menguji setiap fitur dalam sistem yang dikembangkan. Dalam penelitian ini, para peneliti berhasil mengembangkan sistem informasi dengan menggunakan DSMS, dimana metode ini sangat membantu peneliti dikarenakan terdapat tahap pengulangan dan pertambahan sehingga terjadinya komunikasi yang interaktif antara pengguna dan pengembang. Para peneliti juga berhasil menguji setiap *test case* atau *feature* dengan menerapkan metode *blackbox testing* dalam pengujian sesuai skenario yang diharapkan oleh pihak pengembang dan pengguna.

Dari penelitian yang telah dilakukan oleh Dewi et al. didapatkan kesimpulan bahwa mereka berhasil mengembangkan sistem dengan menggunakan metode DSDM serta berhasil menguji sistem yang telah mereka kembangkan dengan menggunakan metode *black box testing*. Oleh karena itu pada penelitian ini menggunakan metode DSDM dalam pengembangan aplikasi pemantauan laut di Teluk Kiluan dikarenakan perlunya interaksi antar pengguna dan pengembang untuk memaksimalkan dalam pengembangan aplikasi berbasis *mobile*. Aplikasi ini juga diharapkan dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan, serta memberikan data hasil pemantauan untuk penelitian lebih lanjut.

Untuk mengetahui aplikasi dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan serta membantu peneliti

dalam mendapatkan data laut, dibutuhkan data responden yang dapat membuktikan bahwa aplikasi yang telah dibuat membantu pihak nelayan ataupun peneliti. Kuesioner merupakan sebuah instrumen pengumpulan data dari responden dengan cara memberikan beberapa pertanyaan secara struktural terkait variabel yang diteliti [16]. Oleh karena itu metode kuesioner cocok digunakan dalam pengujian kepada nelayan dan serta pihak peneliti yang memakai aplikasi.

1.2 Rumusan Masalah

Berdasarkan pemaparan latar belakang yang sudah dijelaskan di atas, penulis mengambil rumusan masalah:

1. Bagaimana mengimplementasikan aplikasi berbasis *mobile* untuk memantau keadaan laut Teluk Kiluan Dengan metode *Dynamic System Development Method* (DSDM)?
2. Bagaimana mengetahui aplikasi berbasis *android* ini dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan serta membantu para peneliti yang ingin meneliti lingkungan laut Teluk Kiluan dalam mengumpulkan data kondisi laut.

1.3 Tujuan Penelitian

Penelitian ini dibuat penulis dengan bertujuan untuk:

1. Membuat aplikasi berbasis *mobile* untuk memantau kondisi laut Teluk Kiluan secara berkala dengan metode *Dynamic System Development Method* (DSDM).
2. Membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan, membantu para peneliti dalam mengumpulkan data kondisi laut dengan menggunakan aplikasi berbasis *android*.

1.4 Batasan Masalah

Batasan masalah pada penelitian ini sangat diperlukan, agar penjelasan lebih terarah dan tidak menyimpang. Berikut batasan masalah yang ditetapkan oleh peneliti :

1. Aplikasi hanya bisa berjalan pada sistem operasi *android*.

2. Aplikasi hanya menerima data yang dikirimkan oleh perangkat sensor dan *instalasi internet of things (IOT)* yang telah terintegrasi oleh aplikasi.
3. Penelitian hanya berfokus di daerah Teluk Kiluan dalam memantau suhu udara, tinggi gelombang, kecepatan angin, dan kecepatan gelombang laut, arah angin level keamanan laut.
4. Aplikasi hanya menampilkan perkiraan cuaca berdasarkan data dari BMKG.
5. Penelitian hanya berfokus dalam pembuatan aplikasi berbasis *mobile*.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah dengan diimplementasikannya aplikasi berbasis *mobile android* ini nelayan dapat memantau kondisi lingkungan laut sebelum melakukan pelayaran mencari ikan serta dapat memberikan data kondisi laut untuk penelitian lebih lanjut.

1.6 Sistematika Penulisan

Bab ini membahas mengenai latar belakang masalah, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, dan sistematika penulisan.

1.6.1 BAB I PENDAHULUAN

Bab ini membahas tentang tinjauan pustaka berupa teori-teori dasar untuk mendukung penelitian.

1.6.2 BAB II LANDASAN TEORI

Bab ini membahas mengenai alur dan rancangan dari penelitian.

1.6.3 BAB III METODE PENELITIAN

Bab ini membahas mengenai hasil dari pembuatan aplikasi dan pengujian aplikasi serta pembahasan dari yang telah di kerjakan

1.6.4 BAB IV HASIL DAN PEMBAHASAN

Bab ini membahas hal yang padat disimpulkan dari penelitian yang telah di kerjakan serta memberikan saran untuk penelitian selanjutnya.

1.6.5 BAB V KESIMPULAN DAN SARAN

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Pada penelitian ini, peneliti mendapatkan tinjauan pustaka dari penelitian-penelitian yang berkaitan dengan pengembangan aplikasi *mobile*. Penelitian-penelitian yang berkaitan dengan pengembangan aplikasi *mobile* dapat dilihat pada Tabel 2.1. Dari Penelitian yang dilakukan oleh Dewi Ayu Nur Wulandari, Muhammad Dika Atthariq, Wahyu Dwi Nanda, Lestari Yusuf (2021) dengan judul Implementasi *Dynamic System Development Method* (DSMS) Pada Sistem Informasi Manajemen Bengkel Mobil Berbasis Web [15], penelitian Deny Friyansyah, Nicolaus Pramono Hardosubroto, Wellhard Halomoan Simamora, dan Yunita Sartika Sari (2022) dengan judul Aplikasi Cafe Point Of Sales (CAPOS) dengan *Dynamic System Development Method* (DSDM) (Studi Kasus Ropang LOILO), dan penelitian oleh Tumini, Sugiy[17] dengan judul Penerapan *Dynamic System Development Method* Pada Sistem Monitoring Status Gizi Balita [18] menggunakan *System Development Method* (DSMS) sebagai metode pengembangan perangkat lunak. Dari penelitian yang dilakukan oleh Tumini, Sugiyanti (2020) dengan judul Monitoring Turbin Angin Menggunakan Smartphone Android [19] dan penelitian oleh Bambang, Rindy C., Yudhi, Ocsirendi, Sidhiq Andriyanto (2021) dengan judul Monitoring Aliran Arus Pasang Surut Air Laut Berbasis Arduino [20] buatlah sistem yang dapat memonitoring kecepatan dan arah angin yang dapat dipantau menggunakan *android* serta pembuatan alat monitoring tinggi dan kuat gelombang air. Pada penelitian yang dilakukan oleh Afrizal Bagas Santoso, Agung Budi Prasetyo, Ike Pertiwi Windasari (2022) dengan judul Perancangan Aplikasi Android Konsultasi Kesehatan Menggunakan React Native [21] dengan menggunakan *Rapid Application Development* sebagai metode pengembangan pengembangan sistem aplikasi.

Tabel 2.1 Tinjauan Pustaka

No	Nama Penulis	Judul	Metode	Hasil	Perbedaan
1	Dewi Ayu Nur Wulandari, Muhammad Dika Atthariq, Wahyu Dwi Nanda, Lestari Yusuf (2021)	IMPLEMENTASI DYNAMIC SYSTEM DEVELOPMENT METHOD (DSDM) PADA SISTEM INFORMASI MANAJEMEN BENGKEL MOBIL BERBASIS WEB	<i>Dynamic System Development Method</i>	DSDM sangat membantu dalam pembuatan aplikasi dikarenakan ada tahapan pengulangan penambahan sehingga timbulnya komunikasi yang intensif antara pengguna dan pengembang. Peneliti berhasil mengimplementasikan sistem informasi management bengkel mobil berbasis web dengan menggunakan metode DSDM. hal ini dibuktikan dengan pengujian aplikasi dengan metode <i>black box</i>	<ul style="list-style-type: none"> - Pada penelitian sebelumnya platform yang dibuat berbasis web, sementara platform yang akan dikembangkan pada penelitian ini berbasis <i>mobile</i>. - Penelitian sebelumnya berfokus pada sistem informasi, sementara sistem yang akan dikembangkan di penelitian ini adalah <i>monitoring</i> data keadaan laut. - Penelitian sebelumnya hanya memakai pengujian <i>black box</i>, sementara pada penelitian ini menggunakan pengujian <i>black box</i> dan kuesioner.
2	Deny Friyansyah,	Aplikasi Cafe Point	<i>Dynamic System</i>	Metode DSDM sangat efektif	- Penelitian sebelumnya

No	Nama Penulis	Judul	Metode	Hasil	Perbedaan
	Nicolaus Pramono Hardosubroto, Wellhard Halomoan Simamora, Yunita Sartika Sari (2022)	of Sales (CAPOS) dengan Dynamic System Development Method (DSDM) (Studi Kasus Ropang LOILO)	<i>Development Method</i>	dikarenakan dari walap pembuatan sampai dengan selesai melibatkan <i>stakeholder</i> sehingga mendapatkan hasil yang maksimal dalam pembuatan aplikasi Cafe Point, hal ini dibuktikan dengan berjalan nya setiap fungsi yang diuji dengan menggunakan <i>black box testing</i>	berfokus pada sistem penjualan, sementara sistem yang akan dikembangkan di penelitian ini adalah <i>monitoring</i> data keadaan laut. - Penelitian sebelumnya hanya memakai pengujian <i>black box</i> , sementara pada penelitian ini menggunakan pengujain <i>black box</i> dan kuesioner.
3	Tumini, Sugiyanti (2020)	Penerapan Dynamic System Development Method Pada Sistem Monitoring Status Gizi Balita	<i>Dynamic System Development Method</i>	Penggunaan metode DSDM dalam Sistem Monitoring Status Gizi Balita didapatkan bahwa sistem dapat dikerjakan dengan berkolaborasi antara pengembang dan pengguna serta sistem yang dibuat sesuai yang diinginkan dan dibutuhkan oleh pengguna.	- Pada penelitian sebelumnya platform yang dibuat berbasis web, sementara platform yang akan dikembangkan pada penelitian ini berbasis mobile. - Penelitian sebelumnya tidak mencantumkan metode pengujian

No	Nama Penulis	Judul	Metode	Hasil	Perbedaan
					sementara pada penelitian ini menggunakan pengujain <i>black box</i> dan kuesioner.
4	Randy Yonanda Pratama, Muldi Yuhendri (2020)	Monitoring Turbin Angin Menggunakan Smartphone Android	<i>Direct field oriented control</i>	Sistem monitoring turbin angin berjalan dengan baik, hal ini dibuktikan dari tampilan data kecepatan angin dan arah angin yang ditampilkan pada sistem android	<ul style="list-style-type: none"> - Penelitian sebelumnya tidak mencantumkan metode pengujian, sementara pada penelitian ini menggunakan pengujain <i>black box</i> dan <i>SUS</i>. - Penelitian sebelumnya menggunakan metode <i>Direct field oriented control</i>, sementara pada penelitian ini memakai DSDM sebagai metode pengembangan. - Pada penelitian sebelumnya penelitian berfokus pada alat dan android, sementara pada penelitian ini

No	Nama Penulis	Judul	Metode	Hasil	Perbedaan
					berfokus pada android saja.
5	Bambang, Rindy C., Yudhi, Ocsirendi, Sidhiq Andriyanto (2021)	Monitoring Aliran Arus Pasang Surut Air Laut Berbasis Arduino	<i>Admiralty</i>	Alat monitoring teal teruji dengan perbandingan rata-rata persentase <i>error</i> sebanyak 1.01%.	<ul style="list-style-type: none"> - Pada penelitian sebelumnya penelitian berfokus pada alat, sementara pada penelitian ini berfokus pada android saja. - Pada penelitian sebelumnya melakukan pengujian dengan metode <i>Admiralty</i>, sementara pada penelitian ini melakukan pengujian dengan <i>black box</i> dan kuesioner
6	Afrizal Bagas Santoso, Agung Budi Prasetyo, Ike Pertiwi Windasari (2021)	PERANCANGAN APLIKASI ANDROID KONSULTASI KESEHATAN MENGGUNAKAN REACT NATIVE	<i>Rapid Application Development</i>	Aplikasi android konsultasi memberikan solusi yang mudah dan efisien dalam penggunaan untuk melakukan pelayanan konsultan kesehatan.	<ul style="list-style-type: none"> - Penelitian sebelumnya menggunakan metode <i>Rapid Application Development</i>, sementara pada penelitian ini memakai DSDM sebagai metode pengembangan.

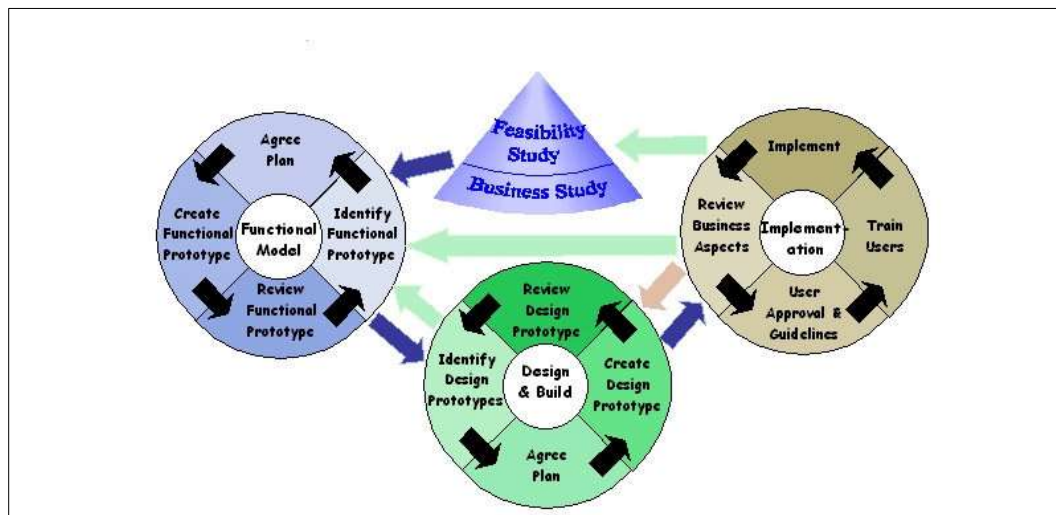
No	Nama Penulis	Judul	Metode	Hasil	Perbedaan
					<ul style="list-style-type: none"> - Penelitian sebelumnya berfokus untuk membuat aplikasi konsultasi, sementara sistem yang ingin dikembangkan di penelitian ini adalah <i>monitoring</i> data keadaan laut. - Penelitian sebelumnya tidak mencantumkan metode pengujian, sementara pada penelitian ini menggunakan pengujain <i>black box</i> dan <i>SUS</i>

2.2. Dasar Teori

Dalam penelitian tugas akhir diperlukan teori pendukung lebih lanjut agar peneliti mendapatkan masukan lebih lanjut. Adapun teori-teori yang berhubungan dengan penelitian tugas akhir ini yaitu:

2.2.1. Dynamic System Development Method (DSDM)

Dynamic System Development Method (DSDM) merupakan pendekatan *Agile Software Development* dalam penyediaan kerangka kerja untuk pembuatan atau pemeliharaan sistem yang menghadapi masalah waktu yang tergolong tidak lama [7]. Metode mengutamakan keterlibatan pengguna dan pengembang secara berkesinambungan secara berulang, serta tanggap dengan perubahan dalam pembuatan sistem perangkat lunak secara tepat waktu dan tepat anggaran [14]. Metode DSDM memiliki siklus perputaran perulangan yang dapat dilihat pada Gambar 2.1.



Gambar 2.1 DSDM Circle [14]

Pada Gambar 2.1 terdapat 5 fase dalam DSDM dalam pengembangan sistem, yaitu:

1. *Feasibility Study*, dalam fase ini dilakukan analisis kepada pihak pemegang proyek atau klien dalam segi kelayakan proyek serta mendiskusikan mengenai jalannya proyek dengan begitu pengembang dapat mengetahui apakah metode DSDM layak untuk dipakai dalam metode ini.

2. *Business Study*, dalam fase ini analisis karakteristik pengguna, kebutuhan fungsional, kebutuhan non-fungsional, serta arsitektur sistem.
3. *Functional Model Iteration*, dalam fase ini dilakukan pembuatan fungsionalitas yang dibutuhkan oleh pengguna menggunakan *prototype*. Siklus pada fase ini adalah *Identify Functional Prototype, Agree Plan, Create functional Prototype, Review functional prototype*.
4. *Design and Build* atau iterasi desain dan pengembangan, dalam fase ini dilakukan penyempurnaan *prototype* dalam bentuk *code* sehingga menjadi sistem yang diinginkan disertai dengan pengujian fungsional. Adapun hasil dari tahap ini adalah *Identify Design Prototype, Agree Plan, Create Design Prototype, Review Design Prototype*.
5. *Implementation* atau implementasi, pada fase ini dilakukan implementasi kepada pengguna yang telah disetujui oleh pihak klien. Adapun hasil dari tahap ini adalah *Review Business Aspects, User Approval and Guideline, Train Users, Implementation*

2.2.2. Bencana Alam Laut

Indonesia merupakan negara dengan banyak pegunungan dan dikelilingi oleh lautan, oleh karena itu Indonesia menjadi negara yang subur dengan sumber daya alam yang melimpah, akan tetapi Indonesia juga tidak luput dari resiko bencana alam. Bencana alam merupakan suatu peristiwa yang merugikan dan mengancam kehidupan makhluk hidup yang terkena dampak bencana. Bencana alam dapat terjadi dimana saja, salah satu tempat yang rawan terjadinya bencana alam adalah laut, seperti bencana alam tsunami, angin topan, gelombang tinggi, serta badai [22]. Tsunami adalah bencana alam laut dengan gelombang air laut yang dapat mencapai 30 meter saat mencapai pantai dengan panjang gelombang 50-200 km yang dapat disebabkan karena letusan gunung berapi atau gesekan lempengan bumi [23]. Bencana alam lainnya yang sering terjadi di laut adalah Badai. Tidak seperti tsunami yang dapat mencapai ketinggian 30 meter, bencana ini memiliki ketinggian gelombang hingga 9 meter, bencana ini dapat mengombang-ambingkan kapal, biasanya bencana alam ini berkaitan dengan kecepatan angin yang tinggi hingga 200 m/s [24].





2.2.3. Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan sekumpulan diagram yang memiliki fungsi sebagai abstraksi sistem perangkat lunak untuk mempermudah pengembangan aplikasi berkelanjutan [25]. UML terdiri dari 10 jenis diagram yang berbeda beda yang mengandung aspek dinamis [26]. Dalam penelitian ini, digunakan *use-case diagram* dan *activity diagram* dikarenakan sudah cukup membantu untuk mengabstraksi sistem yang ingin dikembangkan.

1. Use-case Diagram

Use-case Diagram merupakan suatu diagram yang menggambarkan tingkah laku setiap aktor yang terlibat dalam sistem [27]. Diagram yang dibuat memiliki setiap simbol komponen dan arti yang berbeda dalam pembuatannya. Setiap simbol dan artinya dapat dilihat pada Tabel 2.2.

Tabel 2.2 Komponen Use Case Diagram

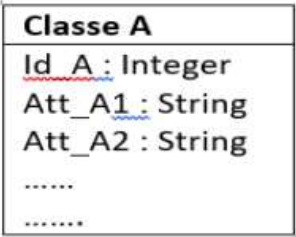
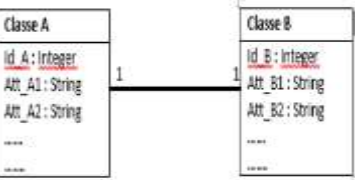
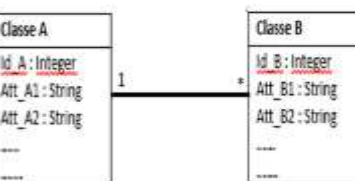
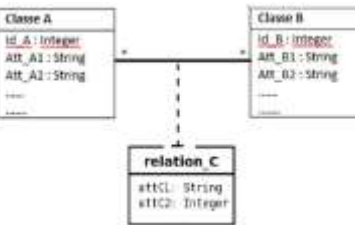
Simbol	Nama	Deskripsi
	<i>Use Case</i>	Merepresentasikan dialog satu dengan dialog lainnya
	<i>Actors</i>	Aktor yang terlibat dalam sistem saat sistem berjalan dengan semestinya
	<i>Extend</i>	Menunjukkan bahwa suatu <i>case</i> dapat tambahan fungsional dan bilamana fungsional tidak berfungsi maka sistem tetap bisa berjalan
	<i>Include</i>	Menunjukkan bahwa suatu <i>case</i> merupakan sebuah fungsionalitas

Simbol	Nama	Deskripsi
		dari <i>case</i> lainnya.

2. Conceptual Data Modeling (CDM)

Conceptual Data Modeling (CDM) merupakan model *database* untuk membangun *database* tanpa relasional yang dapat mendukung dalam penggambaran data *NoSQL* [28]. Sama halnya dengan *use case diagram*, *CDM* juga memiliki simbol komponen dan arti yang berbeda dalam pembuatan nya. Setiap simbol dan artinya dapat dilihat Tabel 2.3.






Tabel 2.3 Conceptual Data Modeling (CDM)

Simbol	Nama	Deskripsi
	<i>Class With Attribute</i>	Setiap <i>class</i> atau tabel dibuat serta atributnya
	<i>Association One to One</i>	Menggambarkan relasi antara 2 tabel yang hanya memiliki 1 relasi sesama
	<i>Association One to Many</i>	Menggambarkan relasi 1 tabel yang bisa digunakan oleh banyak tabel B
	<i>Association Many to Many</i>	Menggambarkan relasi banyak tabel A yang dapat digunakan oleh banyak tabel B

3. Activity Diagram

Activity Diagram adalah sebuah diagram yang berfungsi sebagai pendeskripsian tingkah laku sebuah sistem dan logika pengguna pada sebuah operasi yang kompleks [29]. Sama halnya dengan *use case diagram*, *Activity Diagram* juga memiliki simbol komponen dan arti yang berbeda dalam pembuatannya. Setiap simbol dan artinya dapat dilihat pada Tabel 2.4.

Tabel 2.4 *Komponen Activity Diagram*

Simbol	Nama	Deskripsi
	<i>Start</i>	Melambangkan awal untuk memulai sebuah sistem
	<i>End</i>	Melambangkan akhir untuk mengakhiri sebuah sistem
	<i>Activity</i>	Melambangkan sebuah aktivitas yang dilakukan oleh sebuah sistem.
	<i>Decision</i>	Melambangkan sebuah kondisi yang percabangan
	<i>Swimlane</i>	Melambangkan pemisahan organisasi bisnis yang bertanggung jawab pada sebuah aktivitas.

2.2.4. Android

Android merupakan sebuah sistem operasi perangkat yang ditanamkan di *mobile* berbasis *linux* yang mencakup *middleware* [30]. Pada tahun 2018 - 2020 pengguna *android* menjadi pendominasi sistem operasi *mobile*, dimana terdapat

74.95% tercatat merupakan pengguna *android* [9]. Dari data tersebut dapat ditarik kesimpulan bahwa mayoritas penduduk dunia memakai sistem operasi *android* pada gawai mereka, sehingga sistem operasi *android* cocok digunakan dalam berbagai penelitian yang berkesinambungan dengan perangkat *mobile*.

2.2.5. React Native

React native merupakan *framework* pengembangan *cross platform*, dimana *react native* dapat mengembangkan aplikasi berbasis *android* dan *iOs*. *React native* dirilis untuk pertama kalinya pada tahun 2015 yang dikembangkan oleh *Facebook*, Pada perilisan awalnya *react native* hanya ditujukan untuk sistem operasi *iOS*, akan tetapi dikarenakan *react native* yang bersifat *open source* berhasil membuat komunitas aktif yang dapat mengembangkan dukungan *android* pada *react native* [31]. Pengembangan aplikasi *mobile* dengan *react native* menggunakan bahasa pemrograman *javascript*. *Javascript* dirancang dan diimplementasikan pada Mei 1995 di *Netscape* oleh Brendan Eich, pada awalnya *javascript* dibuat untuk memudahkan *developer* dalam mengembangkan *website*, akan tetapi dari perkembangan zaman, *javascript* dapat berkembang menjadi bahasa dalam pembuatan aplikasi *mobile android* dan *iOS* [32]. Bahasa pemrograman lain yang mendukung dalam pengembangan aplikasi *mobile* dengan *react native* ini adalah *typescript*. *Typescript* merupakan bahasa yang dikembangkan dari *javascript* dimana bahasa ini pertama kali dirilis pada Oktober 2012. Dalam segi performa, *javascript* lebih unggul dibandingkan *typescript*, dikarenakan *typescript* melakukan *compiling* lebih banyak dari *javascript*, dalam segi penulisan *typescript* lebih panjang dikarenakan penulisan tipe data yang ditentukan di awal, oleh karena itu *typescript* lebih sensitif dalam penulisan, akan tetapi akan sangat membantu *developer* dalam mengembangkan aplikasi lebih lanjut [33]. Dalam penelitian ini menggunakan *react native* sebagai *framework* serta menggunakan bahasa pemrograman *typescript* untuk mengembangkan tugas akhir ini. Peneliti mengambil *react native* dikarenakan bersifat *cross platform* sehingga aplikasi dapat berjalan pada sistem operasi *android* dan *iOS* dengan bahasa *typescript* dikarenakan mempermudah pengembang dalam melakukan *maintenance*.

2.2.6. *Black Box Testing*

Black box testing merupakan metode yang berfokus dalam fungsional aplikasi perangkat lunak, pada pengujian ini memungkinkan aplikasi mendapatkan *input* dan *output* sesuai yang diharapkan pengembang dan *customer* [34]. Metode ini memang efektif dalam pengujian terutama dalam segi fungsional, akan tetapi pengembang bisa saja tidak tahu apakah perangkat lunak yang diuji benar-benar berjalan dengan semestinya dikarenakan tidak spesifik dalam mencari kesalahan program. Dalam penelitian ini menggunakan metode pengujian *black box* dalam pengujian aplikasi, dikarenakan metode ini cocok dalam pengembangan aplikasi dengan titik berat fungsionalitas dalam waktu pengembangan yang relatif cepat.

2.2.7. *Kuesioner*

Berdasarkan penelitian yang telah dilakukan oleh Doni et al. [35] Telah dilakukan pengujian aplikasi berbasis *game* dengan menggunakan metode kuesioner. Dimana pada pihak peneliti mendapatkan data dari berbagai responden yang dapat diolah sehingga mendapatkan hasil rata-rata 92% dari nilai kuesioner. Menurut kasnodiharjo [36] sebelum membuat kuesioner, penulis harus mengetahui terlebih dahulu apa tujuan dari penelitian serta menentukan target sampel yang akan mengisi kuesioner, Setelah itu penulis harus membuat pertanyaan yang berkaitan dengan tujuan dari penelitian yang akan dilakukan. Pada penelitian tugas akhir ini, telah didapatkan tujuan yang jelas serta menargetkan 4 responden dari pihak peneliti serta menargetkan sebanyak-banyaknya responden dari pihak nelayan. Pertanyaan yang dibuat akan berkaitan dengan tujuan dari tugas akhir ini sendiri. Di dalam Kuesioner terdapat skala yang dapat mengukur persepsi sikap dari sikap seseorang atau kelompok yang bernama *skala likert* [16]. Skala pengukuran tersebut dapat dilihat pada Tabel 2.

Tabel 2.5 Skor Skala Likert

Tingkat Kepuasan	Skala
Sangat Setuju (SJ)	5
Setuju (S)	4
Cukup Setuju (CS)	3

Kurang setuju (KS)	2
Tidak Setuju (TS)	1

Setelah dilakukan pengukuran maka akan dilakukan analisis dengan menghitung total persentase dari data responden [37] dengan menggunakan rumus berikut.

$$Y = \frac{\sum(N * R)}{Skor Ideal} * 100\%$$

Rumus (2.1) Skala Likert

Keterangan:

Y = Nilai persentase yang dicari

N = Nilai dari setiap Jawaban

R = Frekuensi

Skor Ideal = Nilai likert Tertinggi * Jumlah Responden

Setelah dilakukan pengukuran pada setiap pertanyaan, maka dilakukan pengukuran rata-rata yang dapat menghasilkan nilai akhir serta dapat dianalisis seperti halnya pada Tabel 2.6 berikut.

Tabel 2.6 Bobot Hasil Skala Likert

Rata-Rata Persentase (Y)	Penjelasan
0% - 20%	Sangat tidak setuju / sangat tidak puas
21% - 40%	Tidak setuju / sangat tidak puas
41% - 60%	Cukup / netral
61% - 80%	Setuju/ puas/ baik
81% - 100%	Sangat setuju/ sangat puas/ sangat baik

Dari Tabel 2.6 dapat dilihat bahan analisis untuk menentukan apakah aplikasi *mobile* berbasis android dapat membantu para peneliti dalam mengumpulkan data kondisi laut dengan menggunakan aplikasi serta membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan atau tidak.

2.2.8. *Internet of Things (IOT)*

Internet of Things (IOT) merupakan sebuah teknologi yang hadir di berbagai perangkat elektronik yang dapat memudahkan manusia dalam mengakses sebuah informasi atau alat dengan kebutuhan tertentu [38]. IOT hanyalah sebuah istilah untuk sebuah perangkat keras yang dapat terhubung ke internet dan dapat dikontrol atau dipantau tanpa harus bersentuhan fisik dengan alat tersebut. Salah satu perangkat keras yang biasanya digunakan dalam pembuatan sistem IOT adalah *microcontroller*. Dalam implementasinya, *microcontroller* memperoleh data dari suatu perangkat seperti sensor atau perangkat lain, lalu *microcontroller* mengirimkan data tersebut ke *database* atau *clude* melalui jaringan internet, kemudian data tersebut dapat dipantau atau diimplementasikan lagi ke sistem lainnya [39]

2.2.9. *Firebase*

Firebase merupakan sebuah *platform* layanan dari *google* yang berfungsi untuk memudahkan para pengembang aplikasi dalam membangun *backend* dengan format *JavaScript Object Notation (JSON)* secara *real time* [40]. Layanan-layanan yang tersedia di *Firebase* yaitu Fitur ini bertujuan untuk mengukur keterlibatan pengguna dalam menggunakan aplikasi yang dikembangkan. *Firebase Auth* merupakan layanan yang berfungsi untuk memudahkan pihak *developer* dalam pembuatan sistem masuk dan registrasi memakai *email*, *facebook*, *google*, *github* dan *twitter*. Fitur dalam menyimpan dan mengirimkan berkas data berupa dokumen, gambar, *video*, atau konten lainnya. Dalam pengembangan aplikasi pemantauan berbasis *mobile*, fitur basis data *real-time* sangat berguna dalam membantu pengembang, terutama dalam pergantian data yang cepat [41]. Perlunya keakuratan data yang diterima tiap waktunya menjadikan fitur ini seperti fitur utama dalam pengembangan aplikasi *monitoring*, sekaligus menjadi solusi bagi para pengembang untuk bisa mendapatkan informasi yang cepat secara otomatis disalurkan ke sistem aplikasi tanpa harus melakukan *refresh* halaman [42].

2.2.10. *Express JS*

Express Js merupakan salah satu *framework* hasil pengembangan dari *Node JS* yang mempunyai sifat dalam memberikan kemudahan pembuatan *server-side* dengan bahasa *JavaScript* yang digunakan dalam aplikasi *back-end* [43]. *Back-end* merupakan sebuah program yang berjalan di belakang layar, dimana berfungsi untuk membuat *Application Programming Interface* (API). API merupakan jembatan penghubung antara *database* dan juga tampilan *user* seperti *mobile* ataupun *website* [44]. Dalam penelitian ini peneliti menggunakan *framework Express Js* sebagai aplikasi *back-end* dikarenakan memiliki *base* bahasa yang sama dengan aplikasi *mobile* yang ingin dikembangkan peneliti sehingga tidak berganti-ganti bahasa dalam pembuatan aplikasi.

2.2.11. *MongoDB*

MongoDB merupakan basis data yang bersifat *NoSQL* yang tersusun dari objek dan dokumen [45]. Berbeda dari *database* yang berbasis *SQL*, *MongoDB* dirancang untuk membantu *developer* dalam penulisan *syntax*, dimana *developer* hanya perlu memanggil fungsi yang sudah dirancang oleh pihak pengembang *MongoDB* itu sendiri [46]. Penelitian ini menggunakan *MongoDB* sebagai basis data untuk menyimpan data, dikarenakan terdapat fungsi yang sudah disediakan oleh *MongoDB* sehingga mempermudah pengembang dalam pembuatan aplikasi secara cepat, serta *MongoDB* juga memberikan *server* gratis untuk melakukan data

2.2.12. Uji Validitas dan Reliabilitas

Statistical Product and Service Solution (SPSS) adalah integral proses dari analisis sebuah data [47]. Menurut Ryano et al. [48] validitas merupakan sebuah pengujian untuk mengukur tingkatan keefektifan sebuah kuesioner atau alat ukur lainnya dengan menggunakan membandingkan *significance* 5% sesuai dengan total responden (N) dengan hasil pengujian pada SPSS.

Tabel 2.7 Level *Significance*

N	The Level of Significance	
	5%	1%
3	0.997	0.999
4	0.950	0.990
5	0.878	0.959
6	0.811	0.917
7	0.754	0.874
8	0.707	0.834
9	0.666	0.798
10	0.632	0.765
11	0.602	0.735
12	0.576	0.708
13	0.553	0.684
14	0.532	0.661
15	0.514	0.641

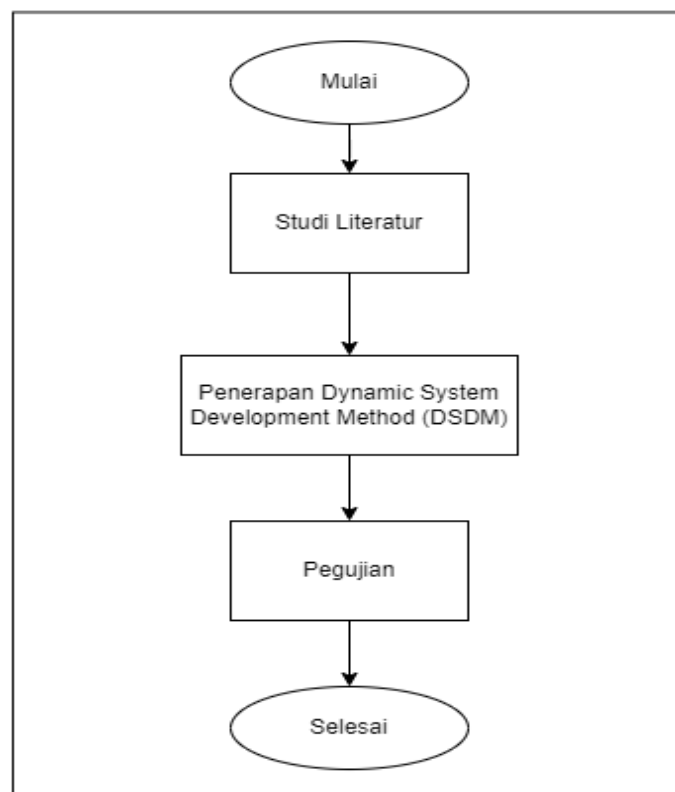
Bilamana significance perhitungan lebih tinggi dari tabel, maka akan bernilai valid. Setelah dilakukan uji validitas, maka dilakukan pula uji coba reliabilitas. Menurut Anggraini et al. [49] reliabilitas merupakan sebuah pengujian yang dapat menunjukkan sejauh apa sebuah kuesioner atau alat ukur lain dapat digunakan dengan baik. Uji reliabilitas sendiri dapat di uji dengan menggunakan Perbandingan dari significance 5% sesuai dengan total responden (N) dengan hasil pengujian pada SPSS.

BAB III

METODE PENELITIAN

3.1. Alur Penelitian

Dalam penelitian ini, digunakan alur penelitian untuk memudahkan peneliti dalam rancang bangun sistem. Alur penelitian dibangun dengan diagram alir yang dapat dilihat pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

Pada Gambar 3.1 dapat dilihat bahwa penelitian ini dimulai dengan studi literatur, penerapan SDLC *Dynamic System Development Method* (DSDM), dan diakhiri dengan pengujian kuesioner.

3.2. Penjabaran Langkah Penelitian

Dari alur penelitian yang sudah digambarkan, berikut merupakan penjelasan setiap langkah dalam alur penelitian tersebut.

3.2.1. Studi Literatur

Perancangan sistem aplikasi *mobile android* ini memerlukan pemahaman secara teoritis terhadap pembuatan aplikasi serta pengambilan data dari sistem IoT dan *database* secara berkala dari berbagai referensi jurnal dan buku. Pada tahap ini juga dilakukan perbandingan metode SDLC berdasarkan jurnal-jurnal yang dibaca sehingga mendapatkan gambaran untuk menentukan metode yang digunakan. Pada tahap ini juga dilakukan pencarian metode yang cocok dalam pengujian aplikasi untuk mengetahui respon pengguna pada aplikasi. Setelah dilakukan pencarian dari berbagai referensi maka dilakukan pemilihan metode pengembangan dan pengujian aplikasi. Dengan terpilihnya metode-metode tersebut, diharapkan dapat membantu dalam proses penelitian yang akan dilakukan secara tim.

3.2.2. Penerapan *Dynamic System Development Method (DSDM)*

Berdasarkan permasalahan yang telah diperoleh akan dibuat sebuah aplikasi berbasis *mobile android* yang diharapkan dapat memberikan solusi bagi pengguna. Aplikasi akan dibuat dengan menggunakan metode SDLC *Dynamic System Development Method (DSDM)* dimana metode ini menekankan koordinasi pengembang dan juga pengguna atau klien. Demikian, pengembangan aplikasi dapat berjalan sesuai kebutuhan dari klien.

Dilihat dari Gambar 3.2, DSDM yang digunakan terdiri dari beberapa tahap perancangan yaitu *Feasibility*, *Business Study*, *Functional Model Iteration*, *Design Prototype Function*, *Implementation*. Terdapat pengulangan atau perbaikan *design* pada tahap *Design Prototype Function* dan *Implementation* bilamana dibutuhkan. Terdapat pengulangan pada fase *Implementation* bilamana terdapat *bug* ataupun melakukan pengembangan lebih lanjut.

3.2.3. Pengujian

Aplikasi berbasis *mobile android* yang telah dikembangkan, dilakukan pengujian dengan menggunakan metode kuesioner. Pada fase ini dilakukan

pembagian kuesioner kepada para target pengguna secara terpisah dimana target pengguna dari aplikasi ini adalah nelayan, dan peneliti. Dalam pengujian ini memilih responden dari pihak peneliti dan nelayan.

Tabel 3.1 Kuesioner Untuk Peneliti

No	Pertanyaan
1	Peneliti merasa mudah dalam menggunakan aplikasi
2	Aplikasi membantu peneliti dalam melihat kuat arus laut Teluk Kiluan
3	Aplikasi membantu peneliti untuk melihat arah angin di laut Teluk Kiluan
4	Aplikasi membantu peneliti dalam melihat tinggi gelombang laut Teluk Kiluan
5	Aplikasi membantu peneliti dalam melihat suhu lingkungan laut Teluk Kiluan
6	Aplikasi membantu peneliti dalam melihat kecepatan arus laut Teluk Kiluan
7	Aplikasi membantu peneliti dalam melihat grafik data arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut Teluk Kiluan pada bulan dan tahun yang dipilih.
8	Aplikasi membantu peneliti dalam mendapatkan data rata-rata arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut Teluk Kiluan pada bulan dan tahun yang dipilih.

Pada tabel di atas dapat dilihat list pertanyaan yang akan diberikan kepada pihak peneliti. Pada Penelitian ini, kuesioner diberikan kepada 4 peneliti yang terlibat langsung pada penelitian Laut Teluk Kiluan untuk mendapatkan data kondisi laut pada waktu-waktu tertentu.

Tabel 3.2 Kuesioner Untuk Nelayan Teluk Kiluan

No	Pertanyaan
1	Nelayan merasa mudah dalam menggunakan aplikasi
2	Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi
3	Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi
4	Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi
5	Nelayan merasa terbantu dalam mengecek arah angin laut dengan menggunakan aplikasi
6	Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi

7	Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi
8	Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi

Pada tabel di atas dapat dilihat list pertanyaan yang akan diberikan kepada pihak nelayan. Pada Penelitian ini, kuesioner diberikan kepada nelayan lokal sebanyak-banyaknya. Setiap responden akan diberikan pertanyaan serta setiap jawaban dari pertanyaan itu akan diolah menjadi hasil akhir seperti yang telah dijelaskan pada sub-bab 2.2.7. Dari data responden tersebut akan didapatkan hasil nilai akhir yang menentukan apakah aplikasi yang dikembangkan membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan dengan menggunakan aplikasi berbasis android.

3.3. Alat dan Bahan Tugas Akhir

3.3.1. Alat

Pada penelitian ini dibutuhkan alat yang dapat mendukung berjalanya penelitian. Alat-alat yang dipakai dalam penelitian ini yaitu:

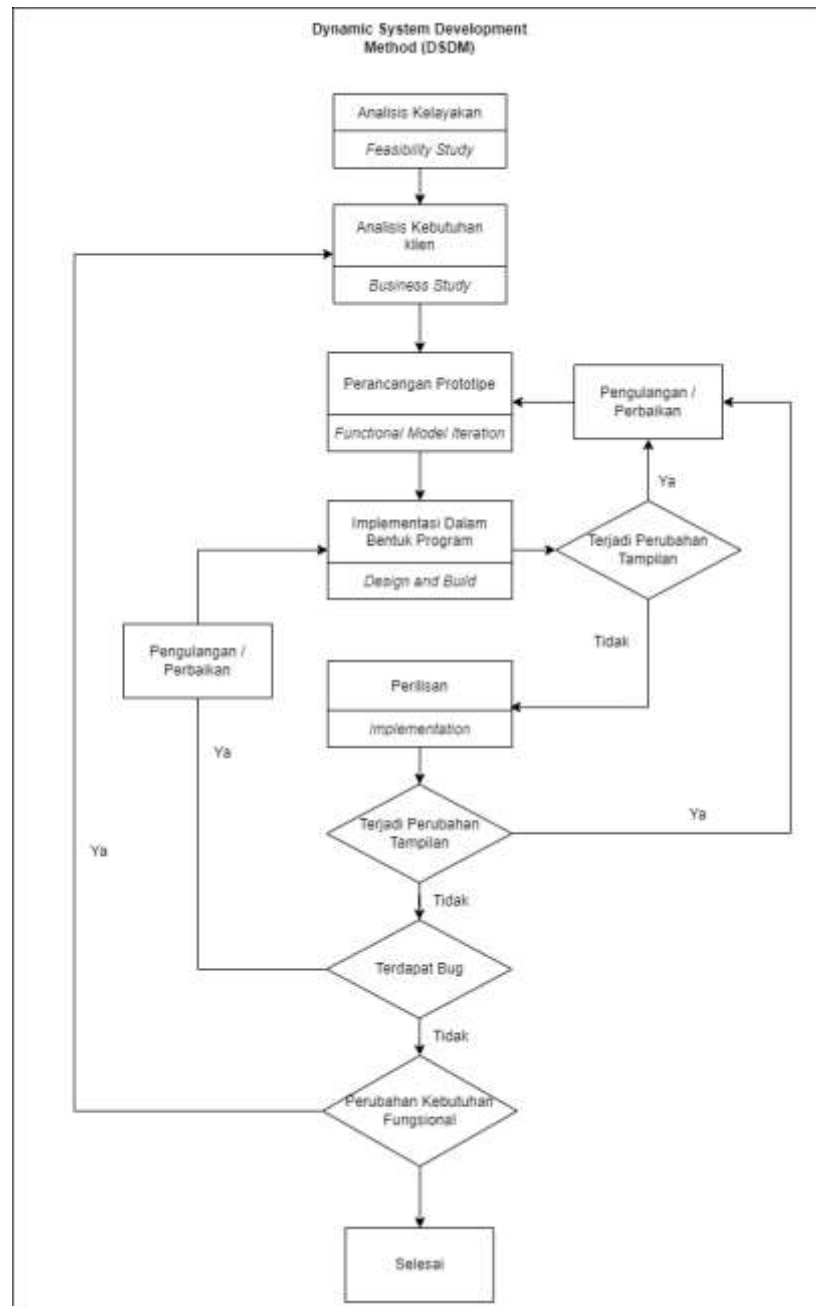
1. *Notebook* dengan spesifikasi Ubuntu OS 22.04 (GNU/Linux), AMD Ryzen 5 4600H @ 4,0 GHz, *memory* 16GB DDR4 3200MHz, SSD 1TB, grafis Nvidia GTX 1050 TI.
2. *Smartphone* dengan spesifikasi OS Android v10 (Q), CPU Octa-core 2 x 2,0 GHz & 6 x 1,8 GHz, GPU Adreno 610, Internal 64 GB, RAM 4GB
3. *Visual Studio Code* sebagai *Code Editor* dengan versi 1.69.2 (dapat berubah seiring adanya pembaruan Visual Studio Code).

3.3.2. Bahan

Pada penelitian ini dibutuhkan bahan yang dibutuhkan dalam penelitian untuk menunjang pengembangan aplikasi. Bahan-bahan yang dibutuhkan yaitu.

1. Hasil wawancara kepada klien
2. Data dari sistem IoT
3. Hasil data kuesioner dari nelayan dan peneliti

3.4. Metode Pengembangan/Metode Pengukuran



Gambar 3.2 Alur DSDM

Pada Gambar 3.2 terlihat Alur *Dynamic System Development Method* (DSDM) yang dimana dipakai untuk pengembangan aplikasi berbasis *mobile android*. Berikut merupakan penjabaran tahap-tahapan dari metode DSDM.

3.4.1. Studi Kelayakan (*Feasibility Study*)

Pada fase ini dilakukan identifikasi alasan yang sah untuk melakukan sebuah proyek ataupun penelitian dengan cara mengajukan beberapa pertanyaan kepada klien. Pada tahap pengajuan pertanyaan, pengembang melakukan *meet* secara *online* dengan Dr. Meezan Ardhanu Asagabaldan, S.Pi, M. Klien menggunakan *google meet*. Berdasarkan hasil wawancara yang mereka lakukan di daerah Teluk Kiluan diperkirakan lebih dari 2 kasus kecelakaan kapal nelayan saat sedang menangkap ikan saat cuaca sedang tidak baik yang dihitung dari bulan Januari 2022 sampai Juli 2022, masyarakat di sana juga tidak mengetahui kapan cuaca yang aman dalam melakukan penangkapan ikan. Dari pihak klien atau peneliti juga membutuhkan data dari lingkungan laut Teluk Kiluan dalam melakukan riset lebih dalam mengenai lingkungan laut Teluk Kiluan.

Kemudian pada tahap ini dilakukan penilaian kelayakan apakah pendekatan DSDM merupakan solusi yang tepat dalam pembuatan aplikasi. Metode DSDM berfokus pada pengerjaan yaitu mengutamakan keterlibatan setiap aktor, serta berfokus pada aktivitas sistem atau proses bisnis yang ada. Pada proyek ini melibatkan peran dari nelayan dan juga klien sebagai peneliti lingkungan laut itu sendiri. Untuk itu DSDM dinilai layak dijadikan sebagai metode pengembangan aplikasi untuk memantau kondisi laut Teluk Kiluan.

3.4.2. Studi Bisnis (*Business Study*)

Pada fase ini dilakukan wawancara kepada klien mengenai karakteristik bisnis dan teknologi. Dengan begitu dapat dilakukan analisis proses bisnis yang terkena dampak secara rinci. Hasil wawancara dapat dilihat dari Tabel 3.3.

Tabel 3.3 Hasil Wawancara Studi Bisnis

No.	Pertanyaan	Jawaban
1	Siapa saja target pengguna dari aplikasi yang	Masyarakat umum seperti nelayan dan peneliti yang ingin meneliti kondisi

No.	Pertanyaan	Jawaban
	dikembangkan?	laut Teluk Kiluan.
2	Apa permasalahan utama yang dapat diselesaikan menggunakan aplikasi yang akan dikembangkan ini?	<ul style="list-style-type: none"> - Nelayan kesulitan memantau kondisi laut sebelum berlayar, - Para peneliti memerlukan data untuk melakukan penelitian lebih lanjut di Teluk Kiluan
4	Apakah aplikasi berjalan di Android dan IOS? Serta apakah aplikasi boleh digunakan oleh orang lain?	Aplikasi hanya berjalan pada sistem android saja dan bisa digunakan oleh semua orang, terkhusus masyarakat atau nelayan Teluk Kiluan dan peneliti.
5	Apa saja kebutuhan fungsional dari aplikasi?	<ul style="list-style-type: none"> - Aplikasi dapat menampilkan kondisi laut Teluk Kiluan seperti arah angin, kuat angin, tinggi gelombang, kuat arus, serta suhu lingkungan laut, serta menampilkan level keamanan laut dari sistem IoT - Aplikasi dapat menampilkan grafik riwayat kondisi laut berdasarkan bulan dan tahun. - Aplikasi dapat menampilkan kompas arah angin laut Teluk Kiluan dari sistem IoT - Aplikasi dapat memunculkan fitur riwayat, grafik, dan Kompas arah angin Teluk Kiluan bilamana pengguna masuk (<i>login</i>) sebagai peneliti.
6	Bagaimana aplikasi berjalan secara garis besar?	Aplikasi hanya mengambil data dari sistem IoT, kemudian data tersebut divisualisasikan ke dalam sistem monitoring, menampilkan level keamanan laut untuk nelayan dan menampilkan grafik dan rata-rata kondisi laut bila mana pengguna masuk sebagai peneliti apabila tidak maka pengguna tidak bisa membuka fitur tersebut.

Dari hasil wawancara yang dilakukan dengan Dr. Meezan Ardhanu Asagabaldan, S.Pi, M.Si selaku klien dari proyek, maka dapat dianalisis

karakteristik pengguna, kebutuhan fungsional, kebutuhan non-fungsional, serta arsitektur sistem.

3.4.2.1. Karakteristik Pengguna

Karakteristik pengguna bertujuan untuk menjelaskan hak akses dan tugasnya dari setiap aktor. Karakteristik pengguna tersebut dapat dilihat pada Tabel 3.4.

Tabel 3.4 Karakteristik Pengguna

Kategori Pengguna	Tugas
Nelayan	<ul style="list-style-type: none"> - Memantau kondisi suhu lingkungan, kuat arus, tinggi gelombang, arah angin, kecepatan angin laut Teluk Kiluan - Mendapatkan informasi aman atau tidaknya kondisi laut Teluk Kiluan
Peneliti	<ul style="list-style-type: none"> - Memantau riwayat kondisi laut Teluk Kiluan per hari dan per bulan - memantau detail arah angin - Semua yang fitur yang didapatkan oleh nelayan dapat didapatkan juga oleh peneliti
Sistem IoT	Mengirimkan data kondisi suhu lingkungan, kuat arus, tinggi gelombang, arah angin, kecepatan angin laut Teluk Kiluan

3.4.2.2. Kebutuhan Fungsional

Kebutuhan fungsional adalah pernyataan layanan pada sistem yang harus tersedia. Sebagai contoh sistem dapat bereaksi dengan *input* tertentu dan akan memberikan keluaran tertentu sesuai dengan masukan. Rancangan kebutuhan fungsional aplikasi pemantauan kondisi laut Teluk Kiluan berbasis *mobile* android dapat dilihat pada Tabel 3.5.

Tabel 3.5 Rancangan Kebutuhan Fungsional

ID	Deskripsi
F-01	Aplikasi dapat menampilkan informasi arah angin
F-02	Aplikasi dapat menampilkan informasi kecepatan angin
F-03	Aplikasi dapat menampilkan informasi kekuatan arus
F-04	Aplikasi dapat menampilkan informasi tinggi gelombang
F-05	Aplikasi dapat menampilkan informasi suhu lingkungan
F-06	Aplikasi dapat menampilkan kondisi aman atau tidaknya lingkungan Teluk Kiluan berdasarkan dari data level keamanan dari sistem IoT
F-07	Aplikasi dapat melakukan <i>login</i> akun sebagai peneliti
F-08	Aplikasi dapat melakukan <i>register</i> akun untuk mendaftar sebagai peneliti
F-09	Aplikasi dapat menampilkan grafik data riwayat dan rata-rata per hari data pemantauan kecepatan angin, tinggi gelombang, suhu udara, dan kecepatan gelombang bila masuk sebagai peneliti.
F-010	Aplikasi dapat menampilkan visualisasi detail arah angin (Seperti Kompas) bila masuk sebagai peneliti.

3.4.2.3. Kebutuhan Non-Fungsional

Kebutuhan non fungsional merupakan batasan dari suatu fungsi yang ditawarkan oleh sistem seperti standarisasi, batasan waktu, dan batasan pengembangan. Rancangan kebutuhan non fungsional aplikasi pemantauan kondisi laut Teluk Kiluan berbasis *mobile* android dapat dilihat pada Tabel 3.6.

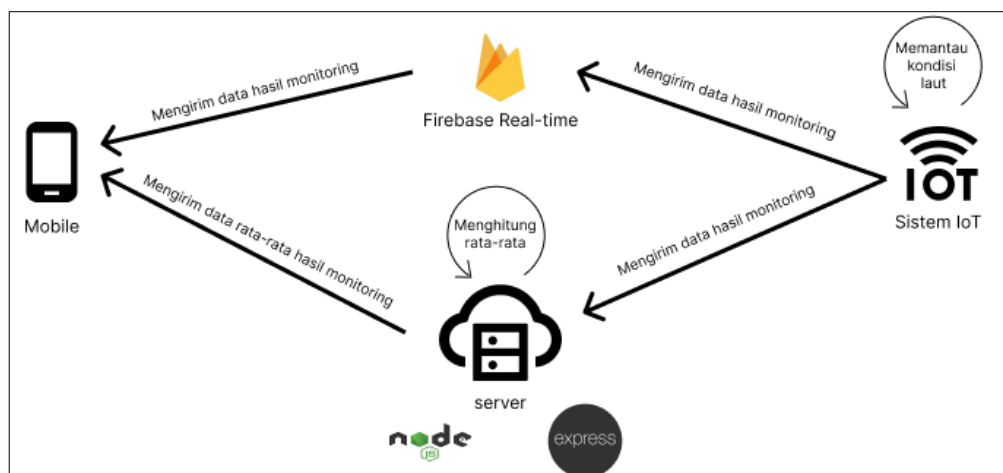
Tabel 3.6 Rancangan Kebutuhan Non-Fungsional

ID	Parameter	Requirement
NF-01	Portabilitas	Sistem dapat dijalankan di perangkat <i>mobile</i> dengan sistem operasi Android
NF-02	Skalabilitas	Sistem dapat terjadi perubahan

ID	Parameter	Requirement
		bilamana dibutuhkan
NF-03	Kompatibilitas	Sistem dapat menampilkan data yang diterima dari sistem IoT

3.4.2.4. Diagram Sistem Arsitektur

Diagram arsitektur merupakan gambaran dari bidang dalam penerapan konsep dan cara kerja sebuah sistem. Diagram sistem arsitektur dapat dilihat pada Gambar 3.3.



Gambar 3.3 Diagram Arsitektur

Dari data arsitektur yang telah dipaparkan, terlihat bahwa aplikasi *mobile* mengambil data dari sistem IoT yang dikirimkan melalui *Firebase*. Sistem IoT juga mengirimkan data ke dalam server yang bertugas untuk mengolah data kecepatan angin, tinggi gelombang, suhu udara, dan kecepatan gelombang menjadi nilai rata-rata setiap hari serta menjadi nilai rata-rata setiap minggu.

3.4.1. Functional Model Iteration

Setelah mengetahui kebutuhan dalam aplikasi, maka selanjutnya dilakukan perancangan. Pada tahap ini dibagi menjadi 4 siklus yaitu *Identify Fungcional Prototype*, *Agree Plan*, *Create functional Prototype*, *Review functional prototype*

3.4.3.1. Identify Functional Prototype

Siklus aktivitas pertama pada tahap *functional model iteration* adalah mengidentifikasi apa saja yang akan dilakukan dalam tahap perulangan model fungsional itu sendiri. Hasil dari siklus ini adalah hal-hal yang akan dilakukan yakni perancangan *prototype* yang meliputi *conceptual data modeling*, rancangan pemodelan algoritma pemantauan lingkungan laut, *use case diagram*, dan rancangan antarmuka aplikasi.

3.4.3.2. Agree Plan

Siklus aktivitas kedua setelah menentukan apa saja hal-hal yang akan dilakukan dalam tahap perulangan model fungsional yaitu menyetujui atas apa yang akan dilakukan atau apa yang sudah diidentifikasi pada siklus sebelumnya. Persetujuan tersebut dilakukan bersama Dr. Meezan Ardhanu Asagabaldan, S.Pi, M.Si selaku klien dari proyek. Hasil dari siklus aktivitas ini adalah persetujuan atas apa yang telah diidentifikasi dari tahap sebelumnya.

3.4.3.3. Create Functional Prototype

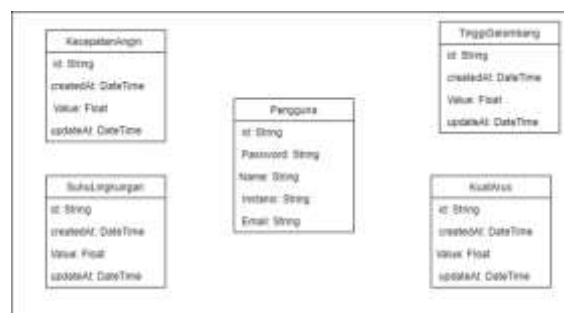
Pada siklus ini melakukan hal-hal yang sudah ditentukan dan disetujui. Hal-hal tersebut meliputi perancangan data, perancangan pemodelan algoritma, *use case diagram*, dan rancangan antarmuka sistem.

a. Perancangan Data

Conceptual Data Modeling (CDM) merupakan model *database* untuk membangun *database* tanpa relasional yang dapat mendukung dalam penggambaran data NoSQL. Dikarenakan masih sebuah rancangan, maka akan ada kemungkinan perubahan dalam implementasi dari pembuatan aplikasi. Dalam pengembangan aplikasi pemantauan kondisi Teluk Kiluan, menggunakan *firebase* dan juga server sebagai media pengolahan data yang dapat dilihat pada Gambar 3.4 dan Gambar 3.5.

Gambar 3.4 Rancangan CDM *Firestore*

Pada Gambar 3.4 dapat dilihat bahwa terdapat tabel pengguna dan sistem IoT. Dari diagram di atas dapat dilihat bahwa pengguna data angin dan laut. Dimana data angin mengirimkan objek arahAngin dan kecepatan_mps. data laut mengirimkan objek suhu_ms5611_c, ketinggian_m, dan kecepatan_mps



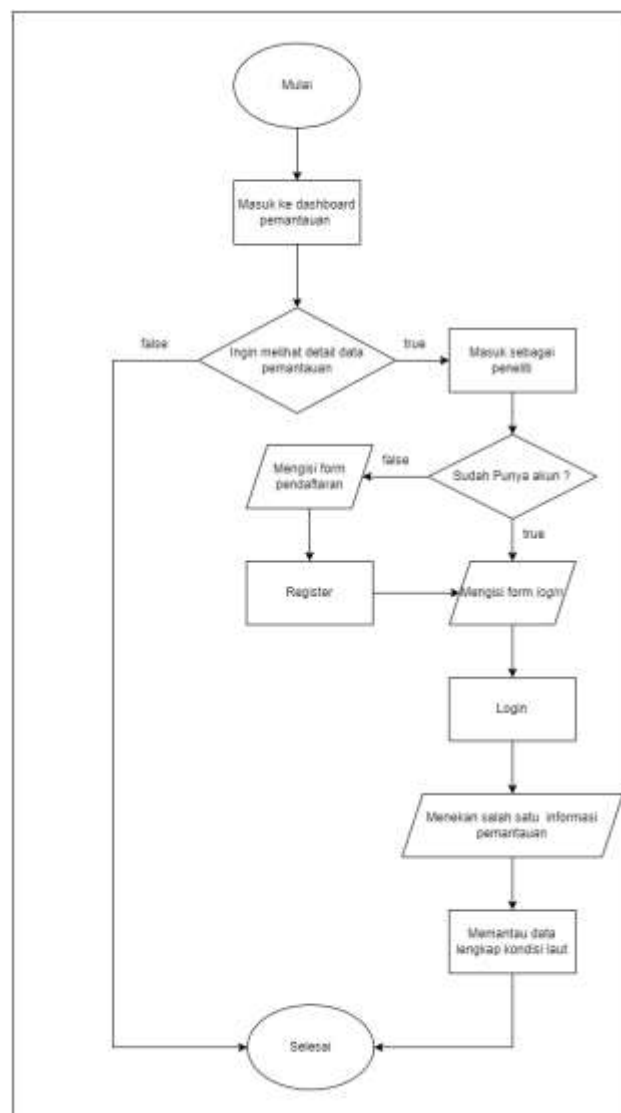
Gambar 3.5 Rancangan CDM Server

Pada Gambar 3.5 dapat dilihat bahwa terdapat tabel pengguna, TinggiGelombang, KuatArus, ArahAngin, KecepatanAngin, SuhuLingkungan dimana setiap tabel menyimpan data updateAt, value, id, dan createAt. Dimana data tersebut dihubungkan dengan tabel pengguna yang menyimpan data *id*, *email*, *password*, *name*, dan *instance*. Dari tabel TinggiGelombang, KuatArus, KecepatanAngin, SuhuLingkungan dimana setiap tabel menyimpan data updateAt, value, id, dan createAt dapat diolah di server menjadi rata-rata tiap data pemantauan dalam *range* harian dan bulanan kemudian diterima oleh

mobile dengan menggunakan *Application Programming Interface* (API).

b. Rancangan Pemodelan Algoritma

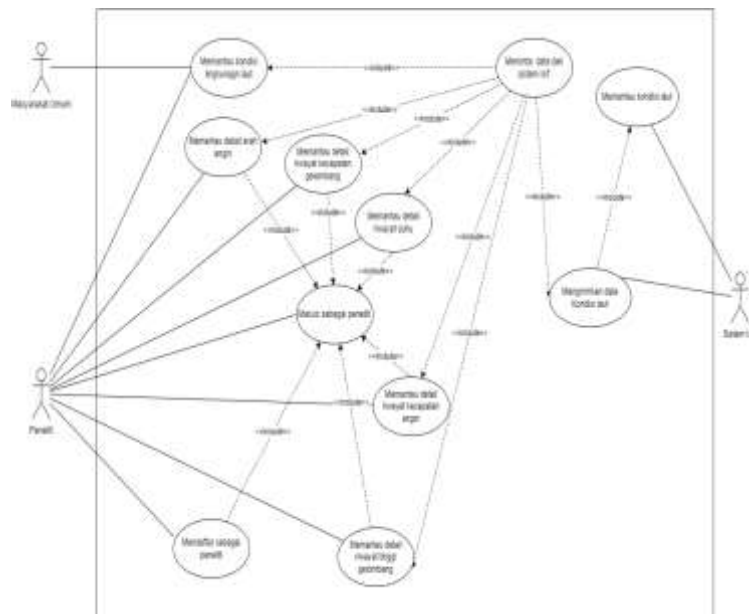
Pada Gambar 3.6 telah dibuat Rancangan pemodelan algoritma aplikasi dengan menggunakan *query* percabangan dengan flowchart. Pengguna akan langsung diarahkan ke pada *dashboard* aplikasi lalu melakukan pengecekan apakah pengguna ingin melihat detail data pemantauan, kondisi *false* mengartikan bahwa pengguna hanya memantau kondisi pada hari itu saja.



Gambar 3.6 Rancangan Algoritma

c. Use Case Diagram

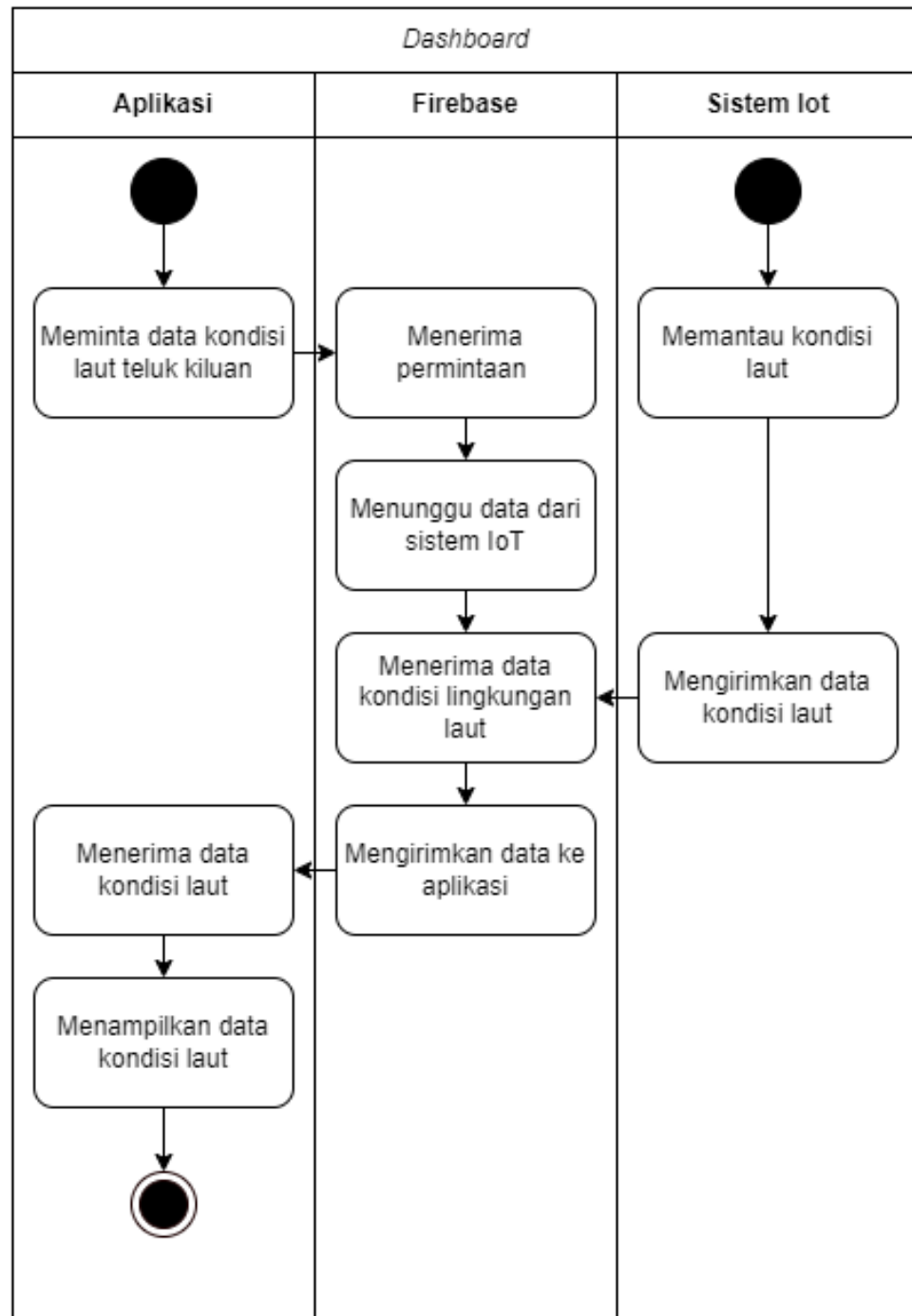
Use Case Diagram mendeskripsikan tentang interaksi pada aplikasi berbasis *mobile*. Pada Gambar 3.7 dapat dilihat bahwa pengguna dapat masuk ke halaman *dashboard* untuk melakukan pemantauan data kondisi laut. Pengguna juga dapat memantau detail riwayat detail arah angin, kecepatan gelombang, suhu lingkungan laut, kecepatan angin serta arah angin. Pengguna juga dapat melakukan *login* atau masuk sebagai peneliti untuk membuka fitur pemantauan detail tiap data. Serta pengguna juga dapat melakukan *registrasi* atau mendaftar sebagai peneliti. Sistem IoT dapat memantau kondisi laut dan mengirimkan data kondisi laut ke dalam server dan *firebase*.



Gambar 3.7 Diagram *Use Case*

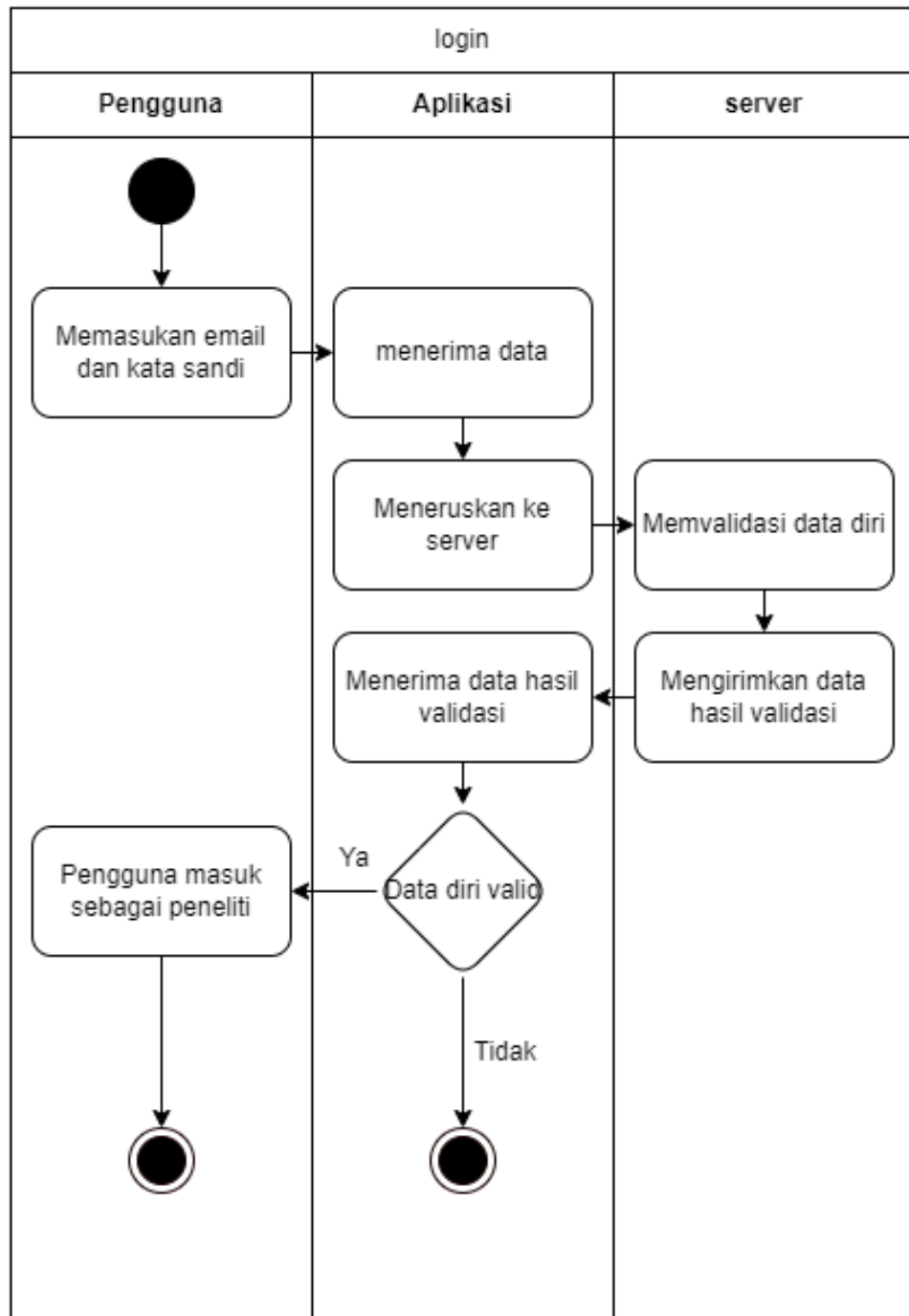
d. Activity Diagram

Activity diagram merupakan sebuah diagram yang bertujuan untuk menggambarkan alur aktivitas dari sebuah sistem. Dalam penelitian ini dipecah menjadi 4 *Activity diagram* yaitu diagram dengan *case* pengguna masuk ke dalam *dashboard*, melakukan *login*, melakukan *register*, dan melakukan pemantauan tiap detail riwayat data kondisi Teluk Kiluan. Berikut gambar dan penjelasan setiap gambar.



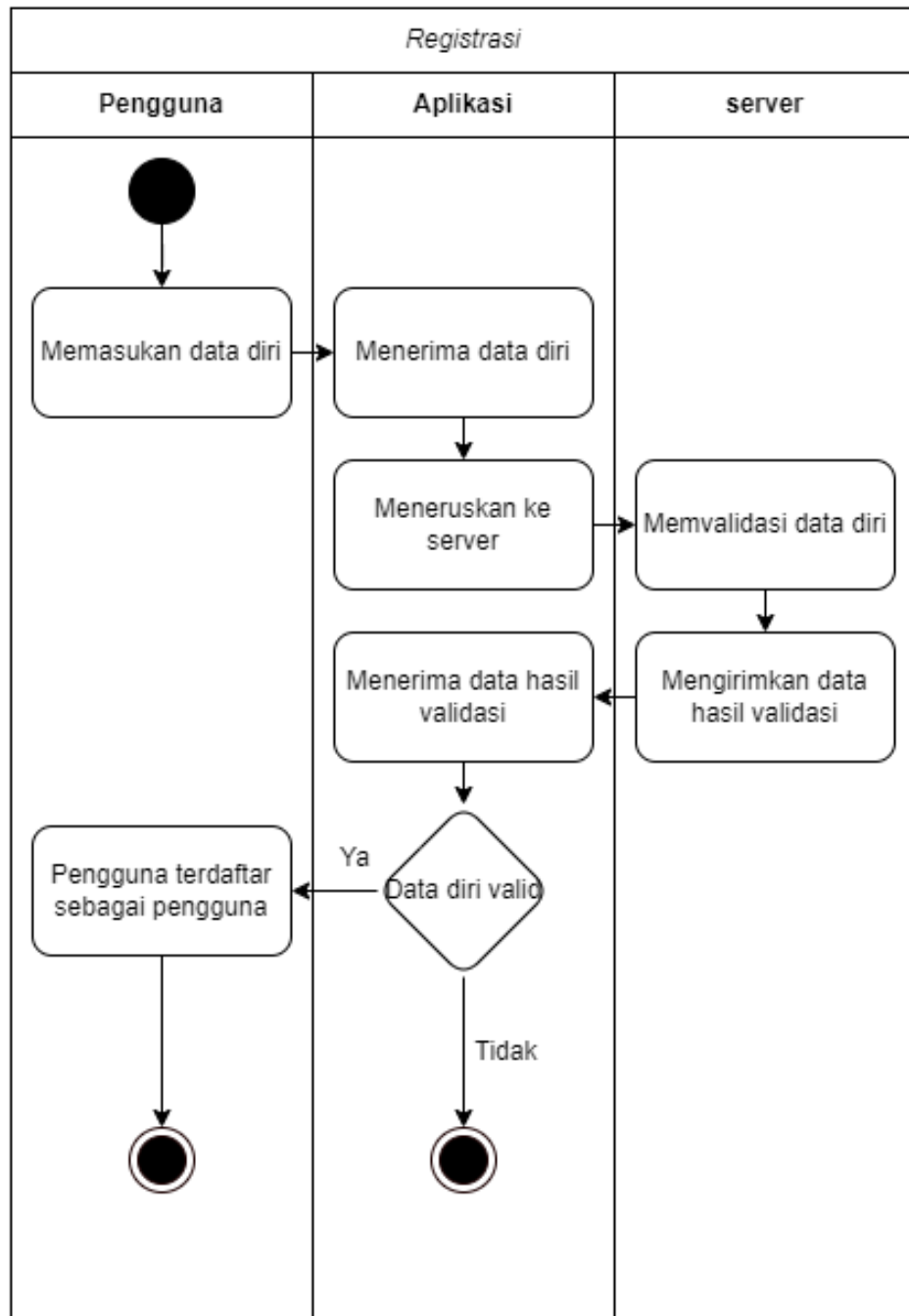
Gambar 3.8 Activity Diagram Dashboard

Pada Gambar 3.8 dapat dilihat bahwa proses diawali dari aplikasi meminta data, kemudian disalurkan kepada *firebase* menunggu data yang dikirimkan oleh sistem *IoT*, kemudian *firebase* menyalurkan data tersebut ke dalam aplikasi sehingga aplikasi dapat menampilkan data kondisi laut.



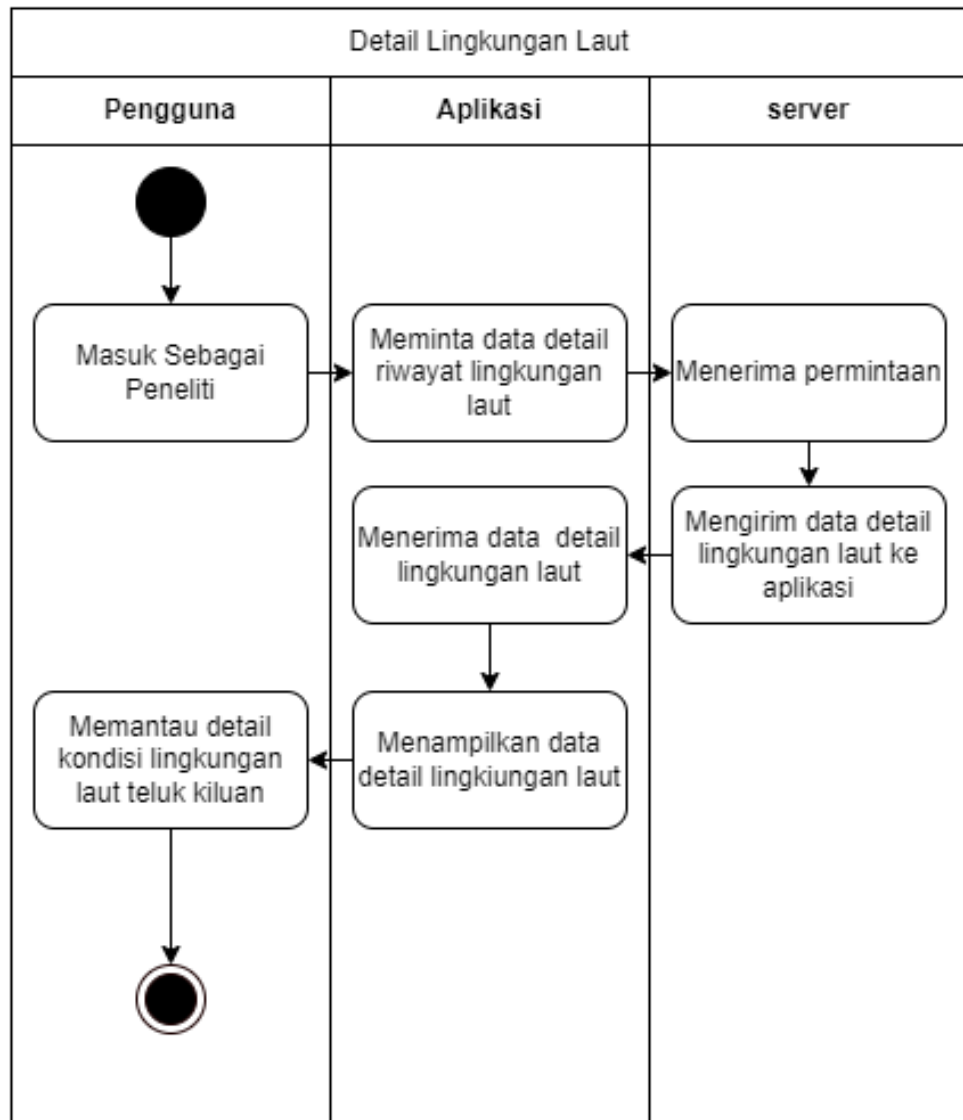
Gambar 3.9 Activity Diagram Login

Pada Gambar 3.9 dapat dilihat bahwa proses diawali dari pengguna memasukkan email dan kata sandi, kemudian aplikasi menerima dan meneruskan data ke server, kemudian server memvalidasi apakah data yang dimasukkan terdaftar, lalu server akan mengirimkan hasil validasi tersebut ke aplikasi, bila data valid maka pengguna bisa masuk sebagai peneliti.



Gambar 3.10 Activity Diagram Register

Pada Gambar 3.10 dapat dilihat bahwa proses diawali dari pengguna memasukkan data diri, kemudian aplikasi menerima dan meneruskan data ke server, kemudian server memvalidasi apakah data yang dimasukkan valid, lalu server akan mengirimkan hasil validasi tersebut ke aplikasi, bila data valid maka pengguna sudah terdaftar sebagai peneliti.



Gambar 3.11 Activity Diagram Detail Lingkungan Laut

Pada Gambar 3.11 dapat dilihat bahwa proses diawali dari pengguna memasukkan masuk sebagai peneliti, kemudian aplikasi meminta detail lingkungan laut ke server, lalu server memberikan data tersebut ke aplikasi yang kemudian dapat dilihat oleh pengguna.

e. Rancangan Antarmuka

Dalam perancangan sebuah aplikasi, dibutuhkan desain rancangan antarmuka yang dapat memudahkan pihak pengguna dalam melakukan

pengembangan aplikasi. Desain antarmuka yang sudah dirancang akan diimplementasikan menjadi aplikasi yang sebenarnya. Rancangan aplikasi yang dibuat dalam penelitian ini yaitu.

The image shows a mobile application login screen wireframe. At the top left, there is a back arrow icon followed by the text "Masuk". In the center, there is a large gray square placeholder labeled "LOGO". Below the logo, there are two input fields: the first is labeled "Email" and the second is labeled "Kata Sandi". Both fields are represented by light gray rounded rectangles. Below the "Kata Sandi" field is a dark gray button with the text "MASUK" in white. At the bottom, there is a link that says "Belum Punya Akun ? DAFTAR".

Gambar 3.12 Rancangan Halaman Login

Gambar 3.12 merupakan tampilan masuk untuk membuka fitur yang hanya bisa dibuka oleh para peneliti. Pada tampilan ini berisikan formulir yang harus dimasukkan.

The image shows a registration form layout. At the top left, there is a back arrow and the text "Daftar". Below this is a placeholder for a logo, labeled "LOGO". The form consists of five input fields, each with a label to its left: "Nama Lengkap", "Instansi", "Email", "Kata Sandi", and "Konfirmasi Kata Sandi". Each field is represented by a light gray rounded rectangle. Below the input fields is a dark gray button with the text "DAFTAR" in white. At the bottom, there is a link that says "Sudah Punya Akun ? Masuk".

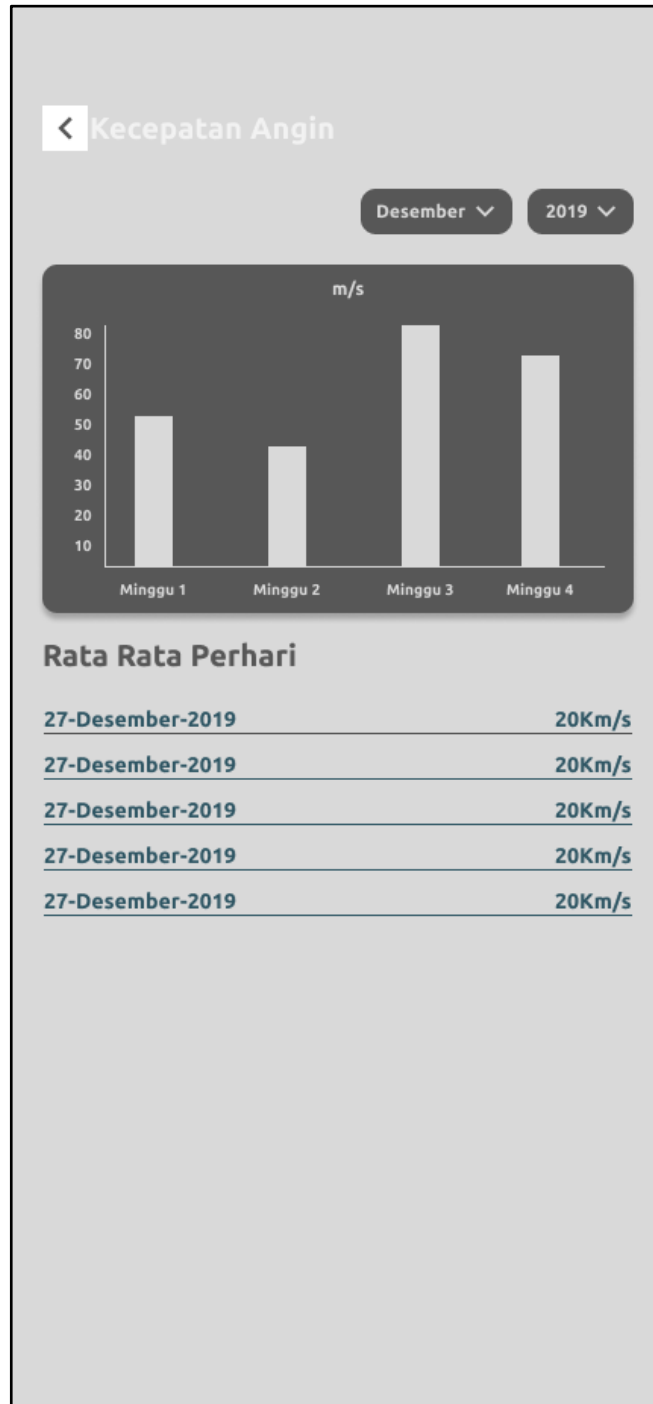
Gambar 3.13 Rancangan Halaman Register

Gambar 3.13 merupakan tampilan pendaftaran untuk mendaftar sebagai peneliti. Terdapat formulir pendaftaran yang wajib diisi oleh calon peneliti.



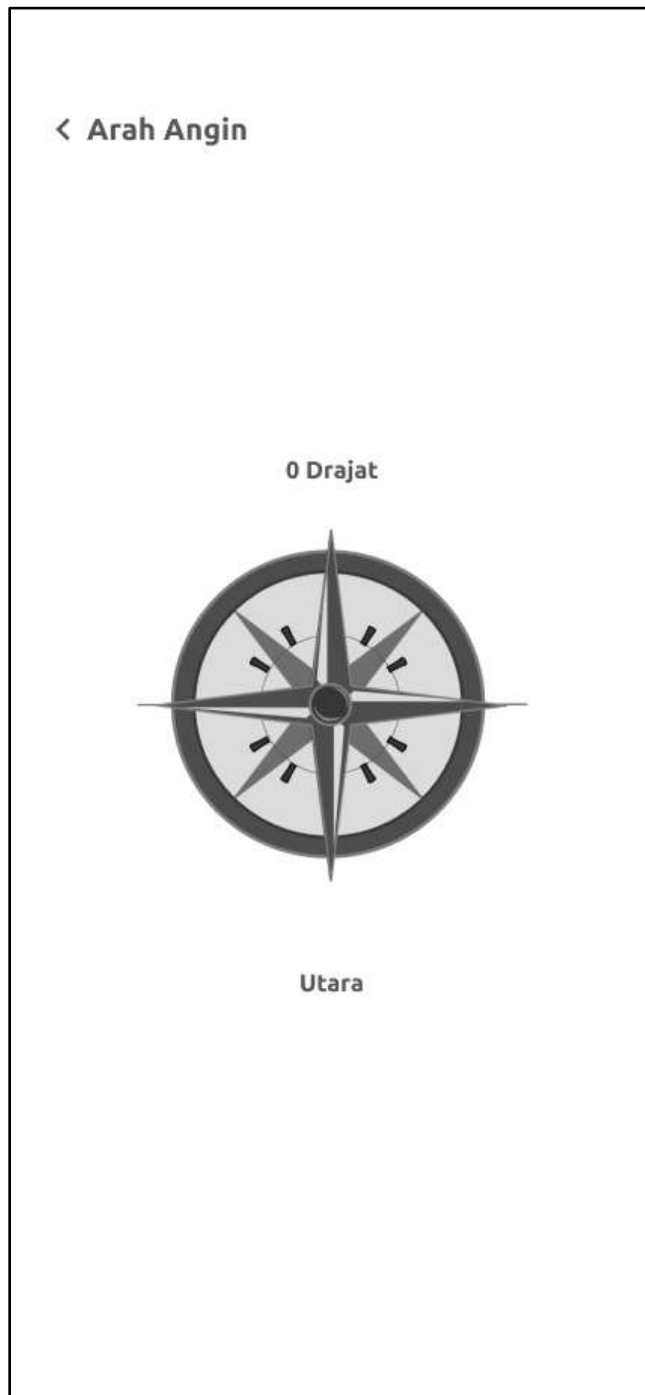
Gambar 3.14 Rancangan Halaman *Dashboard*

Gambar 3.14 merupakan rancangan halaman *dashboard* dimana pengguna dapat memantau kondisi lingkungan laut Teluk Kiluan.



Gambar 3.15 Rancangan Halaman Detail

Gambar 3.15 merupakan rancangan halaman detail yang dapat dibuka ketika pengguna sudah melakukan *login* ke aplikasi.



Gambar 3.16 Rancangan Halaman Arah Angin

Gambar 3.15 merupakan rancangan halaman detail arah angin alaman yang dapat dibuka ketika pengguna sudah melakukan *login* ke aplikasi.

3.4.3.4. Review functional prototype

Pada fase ini melakukan peninjauan kembali apakah hal-hal yang sudah dilakukan dalam siklus *Create Functional Prototype* sudah dilakukan dengan benar. Memeriksa kebenaran *prototype*, *use case*, dan dokumentasi apakah sudah sesuai dengan kebutuhan pengguna dan persetujuan dari klien. Bila mana semua sudah sesuai maka akan melanjutkan ke tahap *design and build*.

3.4.4. Design and Build

Pada tahap *design and build*, melakukan penyempurnaan prototipe dalam bentuk *code* sehingga menjadi sistem yang diinginkan. Bilamana terdapat perubahan dalam aplikasi, maka harus mengulang pada proses *Functional Model*. Jika setiap fungsi sudah teruji dan berjalan sesuai yang diharapkan oleh klien dan pengembang maka dapat melanjutkan ke tahap selanjutnya yaitu *implementasi*. Adapun hasil dari tahap ini adalah *Identify Design Prototype*, *Agree Plan*, *Create Design Prototype*, *Review Design Prototype*.

3.4.4.1. Identify Design Prototype

Pada siklus ini dilakukan identifikasi dari hasil desain dan *flow* aplikasi atau sistem yang telah dibuat sebelumnya sehingga didapatkan *timeline* pengerjaan aplikasi dan perkiraan pengeluaran dana untuk pembuatan aplikasi. Setelah proses identifikasi selesai dilakukan maka akan masuk ke siklus *Agree Plan*.

3.4.4.2. Agree Plan

Pada siklus ini menentukan apa saja hal-hal yang akan dilakukan yaitu menyetujui atas apa yang akan dilakukan atau apa yang sudah diidentifikasi pada siklus aktivitas sebelumnya. Persetujuan tersebut dilakukan bersama Dr. Meezan Ardhanu Asagabaldan, S.Pi, M.Si selaku klien dari proyek. Hasil dari siklus aktivitas ini adalah persetujuan atas apa yang telah diidentifikasi dari tahap sebelumnya.

3.4.4.3. Create Design Prototype

Pada siklus ini dilakukan pembuatan aplikasi dari desain yang telah disetujui sebelumnya oleh klien. Dalam penelitian ini akan dilakukan pembuatan aplikasi *mobile* berbasis android untuk memantau kondisi laut Teluk Kiluan. Bila

aplikasi sudah dibuat maka akan masuk ke proses selanjutnya yaitu *Review Design Prototype*

3.4.4.4. Review Design Prototype

Pada fase ini melakukan peninjauan kembali apakah hal-hal yang sudah dilakukan dalam siklus *Design and Build* sudah dilakukan dengan benar. Pada siklus ini dilakukan pengujian setiap fungsional aplikasi dengan menggunakan metode *black box testing*. Dalam pembuatan aplikasi android pemantauan kondisi laut Teluk Kiluan ini akan diuji fitur-fitur utama. Fitur yang akan diuji yaitu pengujian *form login*, pengujian *form register*, pengujian tampilan dashboard, pengujian tampilan detail setiap *card* data pemantauan.

a. Pengujian *Form Dashboard*

Pada Tabel 3.7 akan dilakukan pengujian pada tampilan *dashboard*. Pada pengujian ini dilakukan pengecekan perubahan data pada setiap kondisi lingkungan laut. Pengujian dilakukan saat pengguna telah melakukan *login* 52 dan sebelum melakukan *login*. Pengguna tidak bisa masuk ke tampilan detail data apabila belum masuk sebagai peneliti.

Tabel 3.7 Rancangan Pengujian *Dashboard*

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
SK-01	Memantau kondisi lingkungan laut	Sistem IoT mengirimkan data arah angin dengan menjadi “Utara”, suhu menjadi “24”, kecepatan angin menjadi “50”, tinggi gelombang menjadi “7”, kecepatan Angin menjadi “40”, kecepatan gelombang “30” ke server	Aplikasi menampilkan kondisi keadaan laut Teluk Kiluan menjadi “Cuaca Tidak Aman”, data arah angin dengan menjadi “Utara”, suhu menjadi “24”, kecepatan angin menjadi “50”, tinggi gelombang menjadi “5”, kecepatan

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
			Angin menjadi “30”, kecepatan gelombang “30”

b. Pegujian *Form Login*

Pada Tabel 3.8 akan dilakukan pengujian fitur *login*. Pengguna harus masuk ke dalam *login page* kemudian mengisi email dan *password*. pengguna bisa masuk ke dalam aplikasi apabila mengisi email dan *password* yang sudah terdaftar. semua *field* dalam formulir harus diisi dengan benar, jika tidak diisi maka pengguna tidak bisa masuk ke dalam aplikasi.

Tabel 3.8 Rancangan Pengujian Login

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
SK-02	Memasukkan email dan <i>password</i> yang salah lalu klik tombol <i>login</i>	Email : contoh@kl.com <i>Password</i> : ContohSandi	Aplikasi akan menolak dan kembali ke halaman <i>login</i> dan muncul pesan “ <i>Email</i> atau <i>Password</i> Salah”
SK-03	Memasukkan email dan <i>password</i> lalu klik tombol <i>login</i>	Memasukkan email dan <i>password</i> lalu klik tombol <i>login</i>	Aplikasi menerima akses <i>login</i> dan pengguna bisa mengakses fitur detail lengkap
SK-04	Tidak memasukkan data pada <i>form login</i> secara lengkap lalu klik Masuk	Mengosongkan <i>form email</i> atau mengosongkan <i>form Password</i>	Memunculkan pesan “Lengkapi data”

c. Pengujian Tampilan Register

Pada Tabel 3.9 akan dilakukan pengujian pendaftaran akun. Pengguna akan memasukkan data diri di register *page*. Pengguna tidak akan bisa mendaftar apabila data diri belum terisi sepenuhnya serta pengguna tidak

bisa mendaftar apabila email sudah terdaftar sebelumnya. Pengguna bisa mendaftar apabila pengguna sudah mengisi data diri dengan benar dan memasukkan email yang belum pernah terdaftar.

Tabel 3.9 Rancangan Pengujian Register

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
SK-05	Tidak memasukkan data pada <i>form</i> register secara lengkap lalu klik Daftar	Mengosongkan salah satu <i>form</i>	Akan ada pesan yang menyuruh pengguna untuk melengkapi data diri.
SK-06	Mengisi <i>form</i> dengan lengkap dan benar lalu klik Daftar	Mengisi nama lengkap, Instansi, Kata sandi, dan kata sandi ulang yang sama dengan sandi	masuk ke menu <i>login</i>
SK-07	Mengisi <i>form</i> Kata sandi berbeda dengan konfirmasi <i>password</i>	Kata sandi : Aku123 Konfirmasi Kata Sandi : aku123	Akan ada pesan yang menyatakan bahwa data yang dimasukkan salah
SK-08	Mengisi <i>form</i> email yang sudah digunakan lalu klik tombol “daftar”	Email : “farras@gmail.com”	Aplikasi akan menampilkan “Email Sudah Terpakai”

d. Pengujian Tampilan Detail Setiap Data Pemantauan

Pada Tabel 3.10 akan dilakukan pengujian pada *detail page*. Pengguna diharuskan *login* terlebih dahulu untuk membuka fitur ini. Dalam *page* ini terdapat grafik dan rata-rata kondisi keadaan yang akan dicatat setiap hari.

Tabel 3.10 Rancangan Pengujian Detail Data

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
SK-09	Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi	Pengguna menekan salah satu dari <i>card</i> kecepatan angin, suhu	Tidak terjadi apa-apa

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
	gelombang, kecepatan gelombang pada halaman beranda saat pengguna belum masuk sebagai peneliti	lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda	
SK-10	Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda saat pengguna setelah masuk sebagai peneliti	Pengguna menekan salah satu dari <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda	Pengguna masuk ke halaman detail yang berisikan grafik mingguan dan rata-rata harian dari Teluk Kiluan
SK-11	Melihat data pada tahun dan bulan tertentu	Pengguna menekan tombol filter tahun dan memilih pilihan “2022” serta menekan tombol filter bulan dan memilih “November”	Aplikasi akan menampilkan grafik dan rata-rata per hari kondisi laut Teluk Kiluan pada bulan dan waktu tersebut.
SK-12	Menekan <i>card</i> arah angin pada halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Tidak terjadi apa-apa
SK-13	Menekan <i>card</i> arah angin pada halaman beranda saat pengguna sudah masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Aplikasi menampilkan kompas arah angin dari Teluk Kiluan, menampilkan derajat arah

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan
			angin tersebut, serta menampilkan dominan arah angin itu sendiri
SK-14	Sistem IoT mengirimkan data arah angin	Sistem IoT mengirimkan data 180 derajat ke <i>firebase</i>	Aplikasi menerima data tersebut dan mengkonvert data tersebut menjadi arah kompas yang menunjukkan ke selatan

Jika setiap fungsi sudah teruji dan berjalan sesuai yang diharapkan oleh klien dan pengembang maka dapat melanjutkan ke tahap selanjutnya yaitu *implementasi*. Dan bilamana terdapat perubahan dalam aplikasi, maka harus mengulang pada proses *Functional Model*.

3.4.5 Implementation

Tahap penerapan merupakan tahap dimana pengguna dapat menggunakan aplikasi yang telah dikembangkan dengan fungsionalitas yang telah sesuai dengan yang sudah dirancang oleh pengembang dan disetujui oleh pihak klien.

3.4.5.1 Review Business Aspects

Pada siklus ini akan dilakukan diskusi antara pengembang dan klien terkait aspek fungsional pada aplikasi. Bilamana aplikasi terjadi perubahan dalam tampilan aplikasi, maka akan mengulang kembali ke tahapan *Functional Model*. Bilamana terjadi *bug* atau kesalahan pada aplikasi maka proses akan mengulang pada tahapan *Design and Build*. Bilamana terjadi perubahan aplikasi seperti penambahan atau pengurangan fitur maka akan kembali lagi ke *Business Study*., bilamana tidak ada perubahan akan bisa dilanjutkan ke siklus selanjutnya

3.4.5.2 Client Approve & Guidelines

Pada siklus ini aplikasi sudah disetujui oleh klien dimana aplikasi sudah tidak ada lagi perubahan fungsional ataupun *bug*. Dalam siklus ini juga akan dilakukan persetujuan antara pengembang dan klien untuk dilakukan peluncuran aplikasi ke *public* serta akan dibuat dokumen *user guide* sebagai pedoman dalam penggunaan aplikasi.

3.4.5.3 Train Klein

Pada siklus ini dilakukan presentasi atau pelatihan kepada peneliti dan nelayan mengenai penggunaan aplikasi.

3.4.5.4 Implement

Pada siklus terakhir ini dilakukan implementasi atau peluncuran aplikasi kepada para pengguna yang akan memakai aplikasi ini untuk nelayan yang ingin memantau kondisi Teluk Kiluan serta para peneliti yang ingin meneliti kondisi laut Teluk Kiluan.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Dynamic System Development Method (DSDM) dibagi menjadi 5 tahap yaitu *Feasibility Study*, *Business Study*, *Functional Model*, *Design and Build*, dan *Implementation*. Hasil dari *Feasibility Study* adalah penentuan kelayakan metode DSDM itu sendiri yang sudah dijelaskan pada sub-bab 3.4.1, hasil dari *Business Study* adalah persetujuan *business* antara pengembang yang sudah dijelaskan pada sub-bab 3.4.2, dan hasil dari *Functional Model Iteration* adalah terciptanya rancangan *prototype* yang meliputi *Conceptual Data Modeling (CDM)*, rancangan pemodelan algoritma pemantauan lingkungan laut, *use case diagram*, dan rancangan antarmuka aplikasi berdasarkan kebutuhan fungsional dan non-fungsional yang telah disetujui oleh pihak pengembang dan klien yang dapat dilihat pada sub-bab 3.4.3. Sedangkan untuk hasil dari *Design and Build*, dan *Implementation* dapat dilihat sebagai berikut.

4.1.1. Hasil *Design and Build*

Pada tahap ini didapatkan hasil berupa penyempurnaan prototipe dalam bentuk *code* sehingga menjadi aplikasi yang diinginkan. Pada tahap ini terjadi pengulangan tahapan yang diakibatkan terjadi perubahan pada desain aplikasi serta kebutuhan dari deployment yang akan dijelaskan pada sub-bab 4.1.2.4. Adapun hasil dari tahap ini adalah *Identify Design Prototype*, *Agree Plan*, *Create Design Prototype*, *Review Design Prototype*.

4.1.1.1. Hasil *Identify Design Prototype*

Pada siklus ini dilakukan identifikasi dari *design* dan *flow* dari rancangan aplikasi. Hasil dari siklus ini adalah didapatkan *timeline* pengerjaan aplikasi seperti yang dapat dilihat pada Tabel 4.1.

Tabel 4.1 *Timeline Pekerjaan*

Tugas	Waktu Mulai	Target Selesai
Pembuatan <i>Backend</i> dan <i>Database</i>	14 September 2022	25 September 2022
Pembuatan Halaman dan Fitur Aplikasi	26 September 2022	28 Oktober 2022
Menyambungkan <i>Database</i> dan Mobile	31 Oktober 2022	04 November 2022
<i>Review</i> dan Pengujian Aplikasi Secara Fungsional	07 November 2022	25 November 2022
<i>Upload</i> Aplikasi ke <i>Playstore</i>	27 November 2022	09 Desember 2022

Pada Tabel 4.1 dilihat bahwa telah dibuat *timeline* yang berguna untuk membantu proses pembuatan aplikasi berbasis *mobile* agar tepat sasaran. Pada siklus ini juga menghasilkan perkiraan daftar pengeluaran untuk pembuatan aplikasi yang dapat dilihat pada Gambar 4.1 berikut.

	B	C	D	E	F
1	DAFTAR KEBUTUHAN MOBILE APP				
2					
3	Nama Alat/Layanan	Tautan (Jika ada)	Harga	Keterangan	Status pembelian
4	Hosting Database	Atlas (mongoDB)	0		<input checked="" type="checkbox"/>
5	Deploy App ke Playstore		400000		<input checked="" type="checkbox"/>
6	Hosting Backend	Fly.io	0		<input checked="" type="checkbox"/>
7					<input type="checkbox"/>
8					<input type="checkbox"/>
9					<input type="checkbox"/>
10					<input type="checkbox"/>
11					<input type="checkbox"/>
12					<input type="checkbox"/>
13					<input type="checkbox"/>
14	Total		400000		
15					
16					

Gambar 4.1 Detail Pengeluaran

Pada Gambar 4.2 dapat dilihat bahwa pengeluaran keuangan dalam pembuatan aplikasi ini adalah sebesar Rp 400000 untuk pembelian *Playstore*, Rp 0 (Gratis) untuk *hosting database*, dan Rp 0 (Gratis) untuk *hosting backend*. Pemilihan list tersebut bertujuan untuk menekan *budget* keuangan dalam pembuatan aplikasi.

4.1.1.2. Hasil Agree Plan

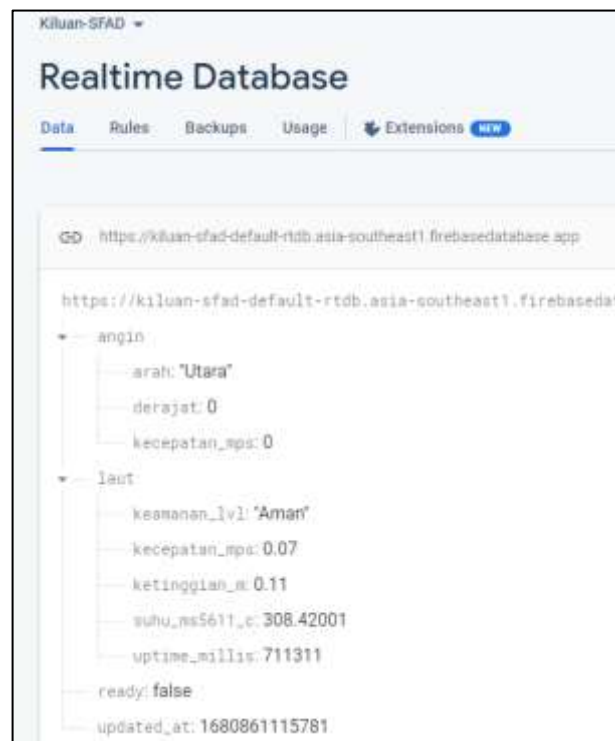
Pada siklus ini menghasilkan persetujuan dari pihak klien dan pengembang tentang apa yang sudah ditentukan pada tahap *identify functional prototype* hal ini dibuktikan dengan dokumen persetujuan perencanaan pengembangan aplikasi yang dapat dilihat pada halaman lampiran. Setelah itu masuk ke siklus *create function prototype*.

4.1.1.3. Hasil Create Design Prototype

Pada tahap ini menghasilkan pembuatan *database*, pengelolaan data, serta pembuatan tampilan aplikasi android. Berikut merupakan jabaran hasil dari siklus ini.

1. Pembuatan *Database* Menggunakan *Firebase*

Gambar 4.2 merupakan hasil dari perancangan dan pembuatan *database* menggunakan *firebase*.



Gambar 4.2 Pembuatan *Database* Dengan *Firebase*

Terdapat *object* “angin” yang dimana didalamnya terdapat objek “arah” yang berfungsi untuk menyimpan arah angin, “derajat” yang berfungsi untuk menyimpan drajat arah angin dan “kecepatan_mps” yang

berfungsi untuk menyimpan data kecepatan angin. Serta terdapat *object* “laut” yang dimana didalamnya terdapat *object* “keamanan_lvl” yang berfungsi untuk menyimpan data kesimpulan keadaan laut secara keseluruhan di Teluk Kiluan, “kecepatan_mps” berfungsi untuk menyimpan data kecepatan ombak laut Teluk Kiluan, “ketinggian_m” berfungsi untuk menyimpan ketinggian gelombang laut Teluk Kiluan, dan “suhu_m5611_c” berfungsi untuk menyimpan suhu lingkungan Teluk Kiluan. Terdapat juga *object* “ready” yang berfungsi untuk menyimpan data kesiapan rilis dimana data tersebut membantu dalam publikasi ke *playsrore*, serta terdapat data *update_at* untuk mempermudah tim IoT dalam pengecekan *post* ke server *firebase*. Data-data dari *object* tersebut didapatkan langsung oleh sistem IoT kemudian di-consume oleh aplikasi *mobile*.

2. Pengelolaan Data

Telah dilakukan pengelolaan data dengan menggunakan *framework express* dengan bahasa *typescript*. Data yang diterima dari sistem IoT kemudian masuk ke dalam *database* yang dimana *database* yang digunakan dibuat dengan menggunakan *mongoDB*.

```
datasource db {
  provider = "mongodb"
  url      = env("DATABASE_URL")
}

generator client {
  provider = "prisma-client-js"
}

model KecepatanAngin {
  id String @id @default(auto()) @map("_id") @db.ObjectId
  value float
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt @map("kecepatan-angin")
}

model TinggiGelombang {
  id String @id @default(auto()) @map("_id") @db.ObjectId
  value float
  createdAt DateTime @default(now())
}
```

```

    updatedAt DateTime @updatedAt
    @@map("tinggi-gelombang")
  }

model KuatArus {
  id String @id@default(auto()) @map("_id") @db.ObjectId
  value float
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
  @@map("kuat-arus")
}

model SuhuLingkungan {
  id String @id @default(auto()) @map("_id")
  @db.ObjectId
  value float
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
  @@map("suhu-lingkungan")
}

model User {
  id String @id @default(auto()) @map("_id")
  @db.ObjectId
  name String
  instansi String
  email String
  password String
}

```

Gambar 4.3 Model Database MongoDB

Pada Gambar 4.3 dapat dilihat pembuatan model *database* dengan *MongoDB*. Terdapat model KecepatanAngin, TinggiGelombang, KuatArus, SuhuLingkungan yang memiliki *object id* yang berfungsi untuk menyimpan *id* data, *value* yang berfungsi untuk menyimpan nilai pada setiap model, *createAt* yang berfungsi untuk menyimpan waktu pembuatan data dari IoT, serta terdapat *updateAt* untuk mengetahui waktu *update* data, terdapat juga model *User* yang menyimpan data nama, instansi, email dan *password* pengguna. Setelah dibuat model database, pembuatan pembuatan API untuk mengirimkan data ke dalam *database*.

```

export const createPostAllDataMw = asyncMw(async (req,
res, next) => {
  req.kecepatanAngin = await
  repository.kecepatanAngin.create({
    value: req.body.kecepatanAngin
  })
})

```

```

req.kuatArus = await repository.kuatArus.create({
  value: req.body.kuatArus
})
req.suhuLingkungan = await
repository.suhuLingkungan.create({
  value: req.body.suhuLingkungan
})
req.tinggiGelombang=await
repository.tinggiGelombang.create({
  value: req.body.tinggiGelombang
})
  return next()
})

export const returnPostAllDataMw = asyncMw(async (req,
res) => {
return res.status(200).json({
  status: 200,
  data: {
    kecepatanAngin: await
    repository.kecepatanAngin.modelToResource(req.kecepa
tanAngin),
    kuatArus: await
    repository.kuatArus.modelToResource(req.kuatArus),
    suhuLingkungan: await
    repository.suhuLingkungan.modelToResource(req.suhuLi
ngkungan),
    tinggiGelombang: await
    repository.tinggiGelombang.modelToResource(req.tingg
iGelombang),
  }
})
})

```

Gambar 4.4 Pembuatan API Input IoT

Pada Gambar 4.4 dapat dilihat bahwa telah dibuatkan API yang mana API tersebut digunakan oleh sistem IoT untuk memasukkan data keadaan laut Teluk Kiluan ke dalam *database*, data dari sistem IoT tersebut kemudian diolah menjadi data rata-rata setiap harinya.

```

export const geDayAverageData = async (query:
Request["query"]) => {
  const { month, year, type } = query;

  const aggregate = getAggregate(type as string);

  if (_.isNil(aggregate)) return;

  const yearMonth = `${year}-${month}`;

  const totalDays = moment(`${yearMonth}-
1`).daysInMonth();

```

```

const results = await seqPromise(
  _.map(_.range(totalDays), async (value) => {
    return {
      date: `${yearMonth}-${value + 1}`,
      value:
        (
          await aggregate(
            {
              //@ts-ignore
              createdAt: {
                gte: moment(`${yearMonth}-${value + 1}`).toDate(),
                lte: moment(`${yearMonth}-${value + 1}`).add(23, "hour").add(59, "minutes").add(59, "second").toDate(),
              },
            ),
            {
              _avg: {
                value: true,
              },
            },
          )
        )?._avg?.value ?? 0,
    };
  })
);

return results;
};

```

Gambar 4.5 Fungsi Pengelolaan Data Rata-Rata

Pada Gambar 4.5 dapat dilihat fungsi pengelolaan data rata-rata setiap *value* yang dikirimkan oleh sistem IoT. Pada fungsi tersebut pula telah dibuat *query* untuk melakukan filter data pada bulan dan tahun tertentu. Data akan di-*query* berdasarkan bulan dan tahun. Server akan mengambil data rata-rata *value* berdasarkan waktu penginputan data sehingga didapatkan hasil rata-rata setiap harinya, bilamana pada waktu tersebut tidak memiliki data maka akan otomatis dideklarasikan bahwa rata-rata pada hari tersebut adalah 0. Setelah mendapatkan data rata-rata harian kondisi laut Teluk Kiluan, maka data diolah kembali untuk memudahkan proses visualisasi data secara grafik.

```

export const getGrafikData = async (query:
Request["query"]) => {
  const { month, year, type } = query;

  const aggregate = getAggregate(type as string);

  if (_.isNil(aggregate)) return;
  if (_.isNil(aggregate)) return;

  const { dates, toDates } = createDateRange({
    month: Number(month),
    year: Number(year),
  });

  const results = await seqPromise(
    _.map(dates, async (date, index) => {
      console.log(moment(date).toDate());

      return {
        index,
        date: moment(date).format("YYYY-MM-D"),
        value:
          (
            await aggregate(
              {
                //@ts-ignore
                createdAt: {
                  gt: date,
                  lte: toDates[index],
                },
              },
              {
                _avg: {
                  value: true,
                },
              }
            )
          )
          //@ts-ignore
          ? _avg?.value ?? 0,
      };
    })
  );

  return results;
};

```

Gambar 4.6 Fungsi Grafik

Pada Gambar 4.6 dapat dilihat fungsi pengelolaan grafik dimana setiap *value* yang sudah dirata-ratakan setiap harinya dilakukan filterisasi dengan memberikan skala setiap minggu dalam kurun waktu 1 bulan, kemudian setiap minggu tersebut dirata-ratakan kembali, sehingga

didapatkanlah data grafik. Telah dibuat *query* untuk melakukan filter data pada bulan dan tahun tertentu. Pada fungsi tersebut juga dibuatkan *handler* untuk membantu dalam memfilter waktu mingguan yang dapat dilihat pada halaman lampiran. Setelah dilakukan pengelolaan data maka dibuatlah API untuk menyambungkan data hasil olahan tersebut pada aplikasi *mobile*. Dibuatkan pula pengelolaan data *user* untuk melakukan *login* dan *register* sebagai peneliti. Untuk fungsi pengelolaan API dan fungsi pengelolaan data *User* dapat dilihat pada halaman lampiran.

```
{
  "status":200,
  "data":
  [
    {"index":0,"date":"2022-11 1","value":0},
    {"index":1,"date":"2022-11-8","value":60.5},
    {"index":2,"date":"2022-11-15","value":79},
    {"index":3,"date":"2022-11-22","value":20.6},
    {"index":4,"date":"2022-11-29","value":0}
  ]
}
```

Gambar 4.7 Contoh Get API

Gambar 4.7 merupakan salah satu hasil data olahan dari *backend* yang ditampilkan API. Data dari API tersebut kemudian digunakan oleh aplikasi *mobile* untuk menampilkan data grafik keadaan laut dan data keadaan laut setiap harinya. Untuk *code* fungsi dan *response* API lainnya dapat dilihat pada halaman lampiran.

3. Pembuatan Aplikasi Berbasis Mobile Android

Telah dilakukan pembuatan aplikasi berbasis android untuk memantau kondisi lingkungan laut Teluk Kiluan. Aplikasi tersebut dibuat berdasarkan hasil diskusi antara pihak pengembang dan pihak klien terkait kebutuhan fungsional yang ada pada aplikasi.



Gambar 4.8 Tampilan Halaman Beranda

Pada Gambar 4.8 terdapat tampilan halaman yang memberikan informasi arah angin, kecepatan angin, kekuatan arus, tinggi gelombang, dan suhu lingkungan dan level keamanan laut. yang dikirimkan oleh Sistem *IoT* ke server *firebase*.

```
const [arahAhngin, setArahAngin] = useState<string>('')

const [kecepatanAngin, setKecepatanAngin] =
  useState<number>(0)

const [KecepatanGelombang, setKecepatanGelombang] =
  useState<number>(0)

const [tinggiGelombang, setTinggiGelombang] =
  useState<number>(0)

const [suhuLaut, setSuhuLaut] = useState<number>(0)

database().ref('/').on('value', snapshot =>
{
  setArahAngin(snapshot.val().angin.arah)

  setKecepatanAngin(snapshot.val().angin.kecepatan_mps)

  setKecepatanGelombang(snapshot.val().laut.kecepatan_mps)
```

```

setTinggiGelombang(snapshot.val().laut.ketinggian_m)

setSuhuLaut(snapshot.val().laut.suhu_ms5611_c)

});

```

Gambar 4.9 Pengambilan Data

Pada Gambar 4.9 dapat dilihat *code* fungsi untuk menampilkan data *firebase* yang dikirimkan oleh sistem *IoT*. Data tersebut ditampilkan dan diolah menjadi informasi kesimpulan kondisi keadaan laut.

```

database().ref('/').on('value', snapshot => {

    setCircumstances(snapshot.val().laut.keamanan_lvl)

});

```

Gambar 4.10 Fungsi Pengambilan Data Keamanan

Pada Gambar 4.10 merupakan fungsi untuk mendapatkan data level keamanan di daerah laut Teluk Kiluan berdasarkan data yang dikirimkan oleh sistem *IoT*. Data level keamanan laut sendiri dikirimkan berdasarkan dari perhitungan fuzzy logic yang diolah oleh sistem *IoT* berdasarkan parameter tinggi gelombang dan kecepatan angin.

Tabel 4.2 Validasi Level Keamanan Laut

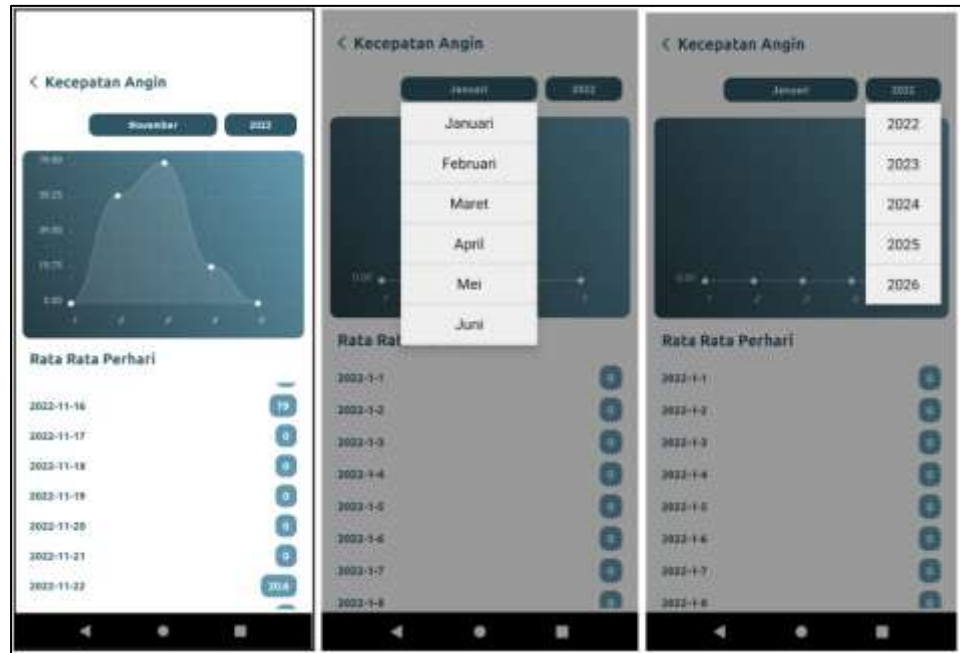
Parameter Yang Digunakan		Hasil Fuzzy Logic	Hasil Yang Ditampilkan Aplikasi
Tinggi Gelombang	Kecepatan Arus		
0.11 m	0.21 m/s	Aman	Aman
1.83 m	1.42 m/s	Waspada	Waspada
5.76 m	3.24 m/s	Bahaya	Bahaya

Dari Tabel 4.2 telah dilakukan validasi juga antara hasil dari fuzzy logic yang diolah oleh sistem *IoT* dan yang ditampilkan pada halaman *dashboard* aplikasi. Pada tampilan halaman ini juga terdapat perbedaan antara pengguna biasa dan pengguna yang telah melakukan *login*

sebagai peneliti, dimana pengguna biasa disapa dengan kalimat “Hallo Pemantau”, sementara untuk pengguna yang sudah *login* sebagai peneliti bisa disapa dengan nama peneliti.

Gambar 4.11 Halaman Login Dan Register

Pada Gambar 4.11 dapat dilihat halaman *login* (Masuk) dan register (Daftar). Pada halaman *login*, sistem meminta email dan *password* untuk masuk sebagai peneliti dan pada halaman register sistem meminta data diri berupa nama lengkap, instansi, *email*, dan *password*. Ketika pengguna sudah masuk sebagai peneliti maka pengguna dapat masuk ke halaman detail grafik beserta rata-rata data kondisi lingkungan laut yang dapat terlihat pada Gambar 4.12.



Gambar 4.12 Halaman Detail Data Lingkungan Laut

Pada tampilan ini dapat dilihat bahwa aplikasi dapat menampilkan data grafik dan rata-rata data setiap harinya pada bulan dan tahun sesuai filter bulan dan tahun yang bisa diubah. Selain melihat data tersebut, pihak peneliti juga dapat melihat arah angin yang divisualisasikan seperti kompas yang dapat dilihat pada Gambar 4.6.



Gambar 4.13 Halaman Detail Arah Angin

Pada Gambar 4.13 dilihat tampilan kompas arah angin yang menunjukkan arah angin Teluk Kiluan dari sistem IoT.

```
<Image source={require('../assets/compass.png')}
style={[styles.image,
{
transform:
[{' rotate: `${(-compassHeader) - (360 -derajat)}deg` }]}
},
]} />
```

Gambar 4.14 Code Visualisasi Arah Angin

Pada Gambar 4.14 dapat dilihat bahwa telah dibuatkan sebuah *image* yang didesain seperti Kompas, kemudian *image* tersebut dibuat dapat menunjukkan arah angin dengan kondisi lapangan Teluk Kiluan. Data lapangan tersebut dikurangi dengan nilai 360 derajat lalu dikurangi lagi dengan arah utara yang telah di-*handle* oleh kompas dari android.

4.1.2.4. Hasil Review Design Prototype

Pada siklus aktivitas *review design prototype* dilakukan pemeriksaan kebenaran sistem yang telah dibuat kepada klien dimana aplikasi yang dibuat, di-*review* dan diuji secara fungsional. Pada tahap ini pengujian aplikasi dilakukan dengan metode *black-box testing* yang dapat dilihat pada sub-bab 4.2.1 sehingga didapatkan hasil bahwa aplikasi sudah berjalan dengan apa yang pengembang dan klien harapkan. Pada tahap ini tidak ada pengulangan siklus dikarenakan tidak adanya *bug* pada aplikasi.

4.1.2. Hasil Implementation

Pada tahap ini didapatkan hasil berupa persetujuan dari klien dimana aplikasi sudah tidak membutuhkan perubahan fungsional serta penambahan fitur. Hasil dari tahap ini juga berupa dibuatnya *user guide* sebagai dokumentasi pemakaian aplikasi dan pelatihan sederhana kepada nelayan di Teluk Kiluan sehingga aplikasi sudah dapat digunakan oleh para nelayan dan klien sebagai peneliti.

4.1.2.1. Hasil Review Business Aspects

Pada siklus ini dilakukan *report* kepada pihak klien mengenai aplikasi yang sudah dibuat. Dari *report* tersebut didapatkan hasil bahwa terdapat penambahan fitur aplikasi yang mengharuskan terjadinya pengulangan ke tahap *Business Study*. Terdapat penambahan fungsional yaitu aplikasi dapat menampilkan ramalan cuaca di daerah Teluk Kiluan dengan rentang waktu hari ini, besok, dan lusa berdasarkan data dari Badan Meteorologi, Klimatologi, dan Geofisika (BMKG), pihak klien juga meminta untuk menambahkan informasi data diri setiap orang yang terlibat dalam penelitian Teluk Kiluan. Berikut merupakan hasil penambahan dan perubahan pada aplikasi.

1. Penambahan Fitur Perkiraan Cuaca

Berdasarkan dari diskusi yang telah dilakukan oleh pihak pengembang dan juga klien, terdapat fitur tambahan untuk memprediksi cuaca di daerah Teluk Kiluan. Fitur tersebut dapat dilihat pada Gambar 4.15.



Gambar 4.15 Halaman Ramalan Cuaca

Pada gambar di atas dapat terlihat halaman aplikasi yang menampilkan fitur ramalan cuaca hari ini, besok hari, dan lusa berdasarkan data dari BMKG. Ditambahkannya fitur tersebut membuat perubahan tampilan pada halaman beranda yang dapat dilihat pada Gambar 4.16 berikut.



Gambar 4.16 Perubahan Tampilan Baranda

Pada gambar di atas dapat terlihat penambahan tombol perkiraan cuaca yang dapat ditekan untuk melihat prediksi cuaca di Teluk Kiluan. Fitur Perkiraan Cuaca tidak memerlukan *login* untuk mengakses fitur tersebut, oleh karena itu nelayan bisa melihat prediksi cuaca di Teluk Kiluan.

2. Penambahan Halaman Informasi

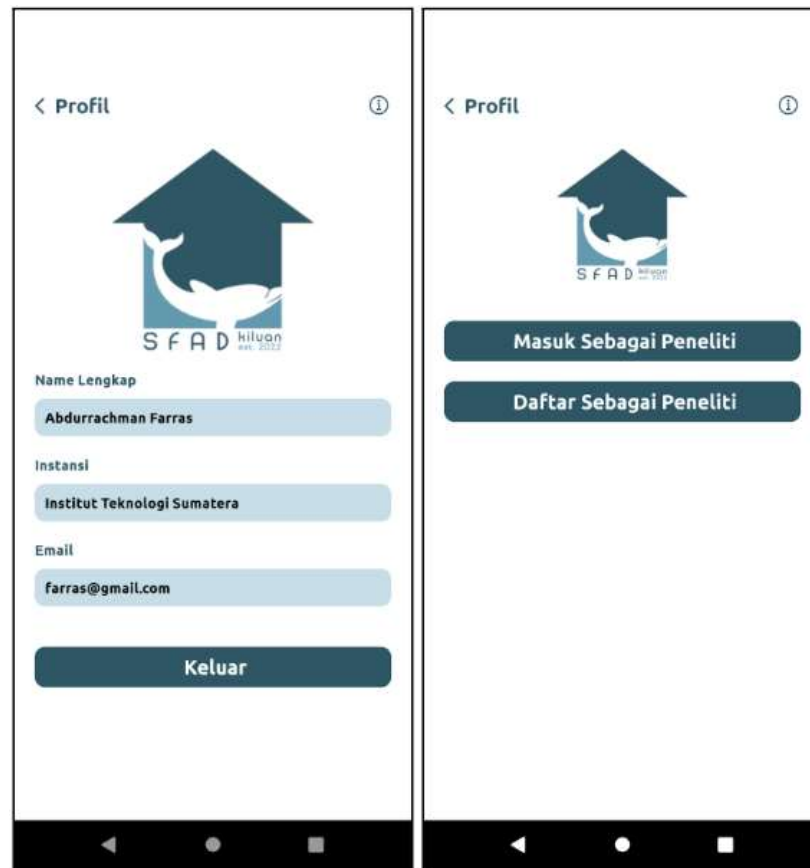
Berdasarkan diskusi yang telah dilakukan oleh pihak nelayan dan pengembang, klien meminta menambahkan informasi mengenai aplikasi dan data diri dari pihak yang berpartisipasi di proyek atau penelitian ini. Halaman informasi dapat dilihat pada Gambar 4.17 berikut.



Gambar 4.17 Tampilan Halaman Informasi

Pada halaman ini bertujuan untuk memberikan informasi kepada pengguna mengenai aplikasi dan memberikan data diri berupa nama, email, dan program studi dosen dan mahasiswa. Dari halaman ini diharapkan pihak nelayan dapat menghubungi pihak peneliti bila terjadi masalah pada alat atau hal urgensi lainnya. Adanya halaman ini

membuat perubahan pada halaman *Profile* yang dimana ditambahkan *icon* informasi yang sebelumnya tidak ada seperti Gambar 4.18 berikut.



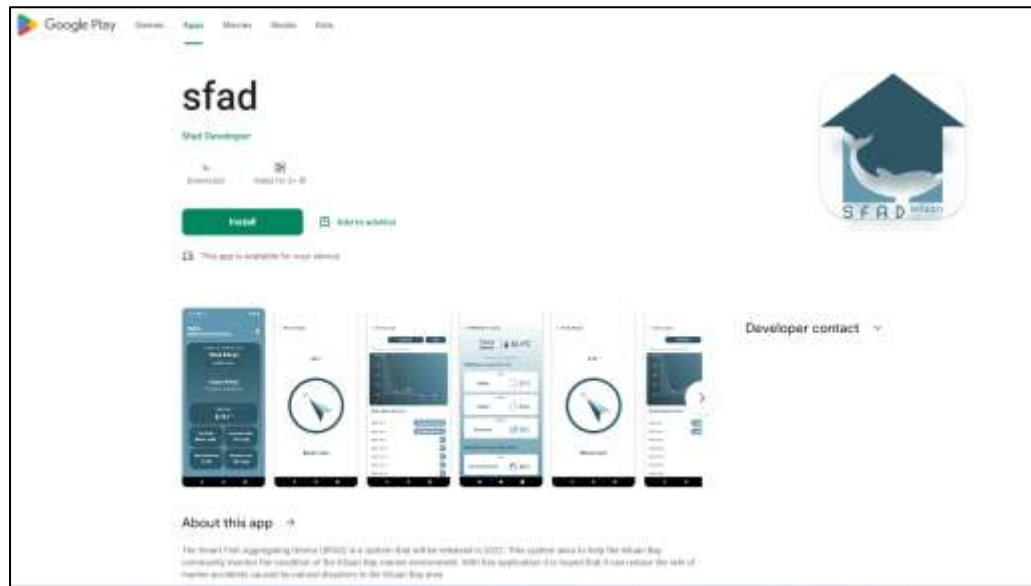
Gambar 4.18 Penambahan Icon Informasi

Setelah dilakukan perubahan pada aplikasi dengan pengulangan siklus tersebut, dilakukan *review* ulang kepada pihak klien. Dari hasil diskusi antara klien dan pengembang didapatkan bahwa tidak ada lagi perubahan pada aplikasi sehingga aplikasi sudah layak dipublikasikan.

4.1.2.2. Hasil Client Approval and Guidelines

Pada siklus ini menghasilkan persetujuan perilsan dari pihak klien mengenai aplikasi yang sudah dikembangkan. Untuk bukti persetujuan sendiri telah dibuatkan dokumen persetujuan. Dokumen persetujuan tersebut dapat dilihat pada lampiran. Melalui persetujuan tersebut maka aplikasi sudah dapat

digunakan oleh para nelayan Teluk Kiluan dan para peneliti yang ingin meneliti kondisi laut Teluk Kiluan.



Gambar 4.19 Playstore Aplikasi

Pada Gambar 4.19 dapat dilihat bahwa aplikasi sudah tersedia di *Playsrore*, dilakukan perilsan tersebut bertujuan untuk membantu para nelayan Teluk peneliti yang ingin meneliti kondisi laut Teluk Kiluan untuk mengunduh aplikasi. Dibuatkan dokumentasi aplikasi berupa dokumen *User Guide* yang dapat dilihat pada halaman lampiran.

4.1.2.3. Hasil *Train Users*

Pada siklus ini telah dilakukan pelatihan sederhana kepada komunitas nelayan di Teluk Kiluan yang dapat dilihat pada Gambar 4.20.



Gambar 4.20 Pelatihan Sederhana Kepada Komunitas Nelayan Teluk Kiluan

Telah dilakukan juga pelatihan sederhana kepada pihak peneliti mengenai cara membaca grafik, hasil olahan data rata-rata keadaan laut, penjelasan detail arah angin Teluk Kiluan, serta penggunaan fitur aplikasi lainnya yang dapat dilihat pada Gambar 4.21.



Gambar 4.21 Pelatihan Sederhana Kepada Peneliti

4.1.2.4. Hasil *Implements*

Pada siklus ini menghasilkan perilisan aplikasi yang dapat diunduh oleh para nelayan Teluk Kiluan untuk memantau keadaan laut Teluk Kiluan sebelum melakukan pelayaran menangkap ikan dan melihat prediksi cuaca serta perilisan aplikasi kepada para pihak peneliti yang membutuhkan data lingkungan laut di Teluk Kiluan.



Gambar 4.22 Bukti Aplikasi Sudah Dirilis

4.2 Hasil Pengujian

Pengujian pada Tugas Akhir ini dibagi menjadi dua yaitu pengujian fungsional dengan menggunakan *black-box testing* dan pengujian lapangan dengan menggunakan kuesioner.

4.2.1 Hasil Pengujian Fungsional

Pada pengujian fungsional, dilakukan uji coba aplikasi dengan beberapa skenario yang sudah disepakati oleh pihak klien dan pengembang. Pengujian fungsional tersebut menggunakan *black-box testing*. Berikut hasil dari pengujian fungsional.

1. Pengujian Data Lingkungan Laut

Pada Tabel 4.2 telah dilakukan data lingkungan laut. Pada pengujian ini melibatkan sistem IoT yang mengirimkan data keadaan laut Teluk Kiluan berupa arah angin, suhu lingkungan, tinggi gelombang, kuat arus dan kecepatan angin ke server kemudian aplikasi mengambil data ke server kemudian mengolah data laut menjadi sebuah kesimpulan yang terdiri dari “Aman”, “Waspada”, dan “Tidak Aman”.

Tabel 4.3 Pengujian Data Lingkungan Laut

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-01	Memantau kondisi lingkungan laut	Sistem IoT mengirimkan data arah angin dengan menjadi “Utara”, suhu menjadi “24”, kecepatan angin menjadi “50”, tinggi gelombang menjadi “7”, kecepatan Angin menjadi “40”,	Aplikasi menampilkan kondisi keadaan laut Teluk Kiluan menjadi “Cuaca Tidak Aman”, data arah angin dengan menjadi “Utara”, suhu menjadi “24”, kecepatan angin menjadi “50”, tinggi gelombang menjadi “5”,	Sesuai

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
		kecepatan gelombang "30" ke server	kecepatan Angin menjadi "30", kecepatan gelombang "30"	

2. Pengujian Login

Pada Tabel 4.3 telah dilakukan pengujian fitur *login*. Pengguna harus masuk ke dalam *login page* kemudian mengisi email dan *password*. Apabila email dan *password* yang dimasukkan sudah benar maka pengguna dapat masuk sebagai peneliti dan bila salah akan ada pesan bahwa pengguna harus memeriksa email dan *password* yang benar. Apabila pengguna mengosongkan salah satu *form* dan menekan tombol *login* maka akan ada pesan bahwa pengguna harus melengkapi data.

Tabel 4.4 Pengujian *Form Login*

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-02	Memasukkan email dan <i>password</i> yang salah lalu klik tombol <i>login</i>	Email : contoh@kl.com Password : ContohSandi	Aplikasi akan menolak dan kembali ke halaman <i>login</i> dan muncul pesan "Email atau Password Salah"	Sesuai
SK-03	Memasukkan email dan <i>password</i> lalu klik tombol <i>login</i>	Memasukkan email dan <i>password</i> lalu klik tombol <i>login</i>	Aplikasi menerima akses <i>login</i> dan pengguna bisa mengakses fitur detail lengkap	Sesuai

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-04	Tidak memasukkan data pada <i>form login</i> secara lengkap lalu klik Masuk	Mengosongkan <i>form</i> email atau mengosongkan <i>form Password</i>	Memunculkan pesan “Lengkapi data”	Sesuai

3. Pengujian Register

Pada Tabel 4.4 Telah dilakukan pengujian pendaftaran akun. Pengguna harus memasukkan data diri beserta alamat email yang belum pernah digunakan sebelumnya untuk bisa mendaftar sebagai peneliti. Apabila pengguna mendaftar dengan email yang sama, tidak melengkapi data diri, kata sandi dan konfirmasi kata sandi berbeda maka pengguna tidak bisa mendaftar sebagai peneliti.

Tabel 4.5 Pengujian *Form* Register

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-05	Tidak memasukkan data pada <i>form register</i> secara lengkap lalu klik Daftar	Mengosongkan salah satu <i>form</i>	Akan ada pesan yang menyuruh pengguna untuk melengkapi data diri.	Sesuai
SK-06	Mengisi <i>form</i> dengan lengkap dan benar lalu klik	Mengisi nama lengkap, Instansi, Kata sandi, dan kata sandi ulang yang sama	Aplikasi akan menerima pendaftaran dan akan langsung masuk ke	Sesuai

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
	Daftar	dengan sandi	menu <i>login</i>	
SK-07	Mengisi <i>form</i> Kata sandi berbeda dengan konfirmasi <i>password</i>	Kata sandi : Aku123 Konfirmasi Kata Sandi : aku123	Akan ada pesan yang menyatakan bahwa data yang di- <i>input</i> salah	Sesuai
SK-08	Mengisi <i>form</i> email yang sudah digunakan lalu klik tombol “daftar”	Email : “farras@gmail.com”	Aplikasi akan menampilkan “Email Sudah Terpakai”	Sesuai

4. Pengujian Tampilan Detail Grafik Dan Rata-Rata Lingkungan Laut

Pada Tabel 4.5 telah dilakukan pengujian pada halaman detail kecepatan angin, suhu lingkungan, tinggi gelombang, dan kecepatan gelombang. Pengguna tidak akan bisa masuk ke halaman detail tersebut apabila tidak masuk sebagai peneliti. Halaman dapat menampilkan grafik dan rata-rata per hari sesuai dengan bulan dan tahun yang dapat diubah dengan menggunakan filter.

Tabel 4.6 Pengujian Tampilan Grafik dan Rata-Rata Lingkungan Laut

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-09	Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang,	Pengguna menekan salah dari <i>card</i> kecepatan angin, suhu lingkungan,	Tidak terjadi apa-apa	Sesuai

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
	kecepatan gelombang pada halaman beranda saat pengguna belum masuk sebagai peneliti	tinggi gelombang, kecepatan gelombang pada halaman beranda		
SK-10	Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda saat pengguna setelah masuk sebagai peneliti	Pengguna menekan salah dari <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda	Pengguna masuk ke halaman detail yang berisikan grafik mingguan dan rata-rata harian dari Teluk Kiluan	Sesuai
SK-11	Melihat data pada tahun dan bulan tertentu	Pengguna menekan tombol <i>filter</i> tahun dan memilih pilihan “2022” serta menekan tombol filter bulan dan memilih “November”	Aplikasi akan menampilkan grafik dan rata-rata per hari kondisi laut Teluk Kiluan pada bulan dan waktu tersebut.	Sesuai

5. Pengujian Tampilan Detail Arah Angin

Pada Tabel 4.6 telah dilakukan pengujian pada halaman detail arah angin, dimana pengguna diharuskan masuk terlebih dahulu sebagai peneliti untuk masuk ke dalam fitur tersebut. Pengguna dapat melihat detail arah angin.

Tabel 4.7 Pengujian Arah Angin

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-12	Menekan <i>card</i> arah angin pada halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Tidak terjadi apa-apa	Sesuai
SK-13	Menekan <i>card</i> arah angin pada halaman beranda saat pengguna sudah masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Aplikasi menampilkan kompas arah angin dari Teluk Kiluan, menampilkan derajat arah angin tersebut, serta menampilkan dominan arah angin itu sendiri	Sesuai
SK-14	Sistem IoT mengirimkan data arah angin	Sistem IoT mengirimkan data 180 derajat ke <i>firebase</i>	Aplikasi menerima data tersebut dan mengkonvert data tersebut menjadi arah kompas yang menunjukan ke selatan	Sesuai

6. Pengujian Tampilan Profil

Pada Tabel 4.7 telah dilakukan pengujian pada halaman tampilan profil dimana terdapat 2 tampilan. Aplikasi akan menampilkan tombol masuk dan daftar sebagai peneliti, akan tetapi aplikasi akan menampilkan data peneliti dan tombol keluar saat sudah masuk sebagai peneliti.

Tabel 4.8 Pengujian Tampilan Profil

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-15	Pengguna masuk ke halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan <i>icon</i> pengguna pada halaman beranda	Pengguna akan masuk ke halaman profil dimana akan ada tombol masuk dan daftar	Sesuai
SK-16	Pengguna masuk ke halaman beranda saat pengguna sudah masuk sebagai peneliti	Pengguna menekan <i>icon</i> pengguna pada halaman beranda	Aplikasi menampilkan data pribadi pengguna berupa nama, email, dan instansi dan terdapat tombol keluar untuk keluar sebagai peneliti	Sesuai
SK-17	Pengguna Keluar dari peneliti	Pengguna menekan tombol keluar kemudian menekan tombol setuju untuk keluar sebagai peneliti	Aplikasi akan masuk ke halaman beranda dan sudah tidak bisa lagi melihat detail dari kondisi lingkungan laut Teluk Kiluan.	Sesuai

7. Pengujian Tampilan Perkiraan Cuaca

Pada Tabel 4.8 telah dilakukan pengujian pada halaman tampilan perkiraan cuaca. Aplikasi akan menampilkan perkiraan cuaca pada hari ini, besok dan juga lusa menurut data dari BMKG.

Tabel 4.9 Pengujian Tampilan Perkiraan Cuaca

ID Skenario	Skenario pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapat
SK-18	Masuk ke halaman perkiraan cuaca	Pengguna menekan tombol perkiraan cuaca pada halaman beranda	Aplikasi akan menampilkan perkiraan cuaca pada hari ini, besok dan lusa menurut data dari BMKG	Sesuai

Berdasarkan hasil pengujian menggunakan *black-box testing*, mendapatkan hasil yang memuaskan, dimana aplikasi berjalan sesuai harapan pihak klien dan pengembang.

4.2.2 Pengujian Kuesioner

Aplikasi *mobile* android yang dikembangkan pada tugas akhir ini bertujuan untuk membantu para peneliti dalam mengumpulkan data kondisi laut dengan menggunakan aplikasi serta membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan. Oleh sebab itu aplikasi yang telah dikembangkan juga diuji dengan menggunakan metode kuesioner seperti yang sudah dijelaskan pada sub-bab 2.2.7. Hal ini bertujuan untuk mengetahui apakah aplikasi yang telah dikembangkan dapat membantu pihak peneliti dalam mendapatkan data laut dan membantu nelayan dalam memantau kondisi laut yang aman atau tidak.

4.2.2.1 Hasil Pengujian Kepada Pihak Peneliti

Telah dilakukan pengujian kuesioner kepada para peneliti yang ingin meneliti lingkungan laut Teluk Kiluan. Terdapat beberapa pertanyaan yang diajukan, pertanyaan tersebut dapat dilihat pada Tabel 4.1 Setelah peneliti mengisi setiap soal maka dilakukan perhitungan untuk menghitung total persen kepuasan setiap jawaban dengan menggunakan rumus yang telah dijelaskan pada Rumus

2.1. Berikut merupakan hasil perhitungan berdasarkan jawaban dari setiap responden.

Tabel 4.10 Hasil Kuesioner Kepada Peneliti

Nomor Pertanyaan	Jumlah Responden yang memilih (R)					$\sum N \times R$	Persentase (Y)
	SJ(5)	S(4)	CS(3)	KS(2)	TS(1)		
1	3	1	0	0	0	19	95%
2	1	3	0	0	0	17	85%
3	0	3	1	0	0	15	75%
4	0	4	0	0	0	16	80%
5	0	4	0	0	0	16	80%
6	0	4	0	0	0	16	80%
7	0	2	2	0	0	14	70%
8	0	2	2	0	0	14	70%
Rata-Rata							79.3%

Pada Tabel 4.10 dapat terdapat 8 pertanyaan. Pada pertanyaan pertama mendapatkan skor 95% dengan 3 responden menjawab sangat setuju dan 1 responden menjawab setuju, dari data tersebut dapat dianalisis bahwa peneliti merasa sangat setuju bahwa aplikasi mudah digunakan. Pada pertanyaan kedua mendapatkan skor 85% dengan 1 responden menjawab sangat setuju dan 3 responden menjawab setuju, dari data tersebut dapat dianalisis bahwa peneliti merasa sangat setuju bahwa aplikasi dapat membantu peneliti dalam melihat kuat arus laut. Pada pertanyaan ke tiga mendapatkan skor 75% dengan 3 responden menjawab setuju dan 1 responden menjawab dengan netral, dari data tersebut dapat dianalisis bahwa peneliti masih merasa terbantu untuk melihat visualisasi data arah angin. Pada pertanyaan keempat, kelima, dan keenam, mendapatkan skor yang sama yaitu 80% dengan 4 responden menjawab setuju, dari data tersebut dapat disimpulkan bahwa peneliti merasa setuju bahwa aplikasi membantu dalam melihat data tinggi gelombang, suhu lingkungan laut, dan kecepatan arus laut. Pada pertanyaan ketujuh dan kedelapan didapatkan skor yang sama yaitu 70% dengan 2 responden menjawab setuju dan 2 menjawab netral, dari data tersebut dapat disimpulkan bahwa aplikasi masih tergolong membantu peneliti dalam mendapatkan data dan melihat grafik laut Teluk Kiluan. Dari semua skor yang ada, dengan menggunakan *skala likert*, didapatkan skor 79.3% yang memiliki arti Setuju/ Puas/ Baik, dari skor tersebut

dapat dianalisis bahwa aplikasi masih tergolong membantu para pihak peneliti untuk mengumpulkan data keadaan laut Teluk Kiluan sehingga dapat disimpulkan bahwa aplikasi membantu pihak peneliti dalam mengumpulkan data lingkungan laut Teluk Kiluan untuk penelitian lebih lanjut mereka.

Setelah melakukan analisis kuesioner dengan *skala likert*, selanjutnya dilakukan uji validasi dan reliabilitas pada kuesioner dengan SPSS.

Correlations										
	P1	P2	P3	P4	P5	P6	P7	P8	Total	
P1	1	.333	1.000 ^{**}	.333	.333	.333	.577	.577	.888	
	Sig. (2-tailed)		<.001	.887	.887	.887	.423	.423	.032	
N	4	4	4	4	4	4	4	4	4	
P2		1	.333	-.333	-.333	-.333	-.577	-.577	.086	
	Sig. (2-tailed)		.887	.887	.887	.887	.423	.423	.914	
N		4	4	4	4	4	4	4	4	
P3			1	.333	.333	.333	.577	.577	.888	
	Sig. (2-tailed)			.887	.887	.887	.423	.423	.032	
N			4	4	4	4	4	4	4	
P4				1	1.000 ^{**}	1.000 ^{**}	.577	.577	.888	
	Sig. (2-tailed)				<.001	<.001	.423	.423	.558	
N				4	4	4	4	4	4	
P5					1	1.000 ^{**}	.577	.577	.888	
	Sig. (2-tailed)					<.001	.423	.423	.558	
N					4	4	4	4	4	
P6						1	.577	.577	.888	
	Sig. (2-tailed)						.423	.423	.558	
N						4	4	4	4	
P7							1	1.000 ^{**}	.764	
	Sig. (2-tailed)							<.001	.236	
N							4	4	4	
P8								1	.764	
	Sig. (2-tailed)								<.001	.236
N								4	4	
Total									1	
	Sig. (2-tailed)									
N										

Gambar 4.23 Correlations Pada Kuesioner Peneliti

Pada gambar di atas terdapat nilai *person correlation* yang akan dibandingkan nilai perhitungan yang di ambil dari data kolom total, sehingga dapat dilihat kevalidan dari kuesioner.

Tabel 4.11 Hasil Validasi Kuesioner Peneliti

Nomor	Perhitungan		(N= 4, a = 0.05)	Keterangan
1	0.968	>	0.950	Valid
2	0.086	<		Tidak Valid
3	0.968	>		Valid
4	0.444	<		Tidak Valid
5	0.444	<		Tidak Valid
6	0.444	<		Tidak Valid
7	0.764	<		Tidak Valid
8	0.764	<		Tidak Valid

Setelah melakukan validasi, maka dilakukan juga reliability untuk mengetahui tingkatan keandalan kuesioner.

Scale: ALL VARIABLES

Case Processing Summary

		N	%
Cases	Valid	4	100.0
	Excluded ^a	0	.0
	Total	4	100.0

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

Cronbach's Alpha	N of Items
.737	8

Gambar 4.24 *Relibilty Statics* Pada Kuesioner Peneliti

Dari hasil validasi dan *relibility* dapat di analisis bahwa masih banyaknya perhitungan kari kuesioner yang tidak valid sehingga dapat di simpulkan bahwa kuesioner yang dilontarkan bersifat tidak dapat di andalkan. Salah satu faktor yang dapat menyebabkan ini terjadi adalah sedikitnya responden dari pihak peneliti.

4.2.2.1 Hasil Pengujian Kepada Pihak Nelayan

Telah dilakukan pengujian kuesioner kepada para nelayan Teluk Kiluan. Terdapat beberapa pertanyaan yang diajukan, pertanyaan tersebut dapat dilihat pada Tabel 4.11 Setelah nelayan mengisi setiap soal maka dilakukan perhitungan untuk menghitung total persen kepuasan setiap jawaban dengan menggunakan rumus yang telah dijelaskan pada Rumus 2.1. Berikut merupakan hasil perhitungan berdasarkan jawaban dari setiap responden.

Tabel 4.12 Hasil Kuesioner Kepada Nelayan

Nomor Pertanyaan	Jumlah Responden yang memilih (R)					$\sum N \times R$	Persentase (Y)
	SJ(5)	S(4)	CS(3)	KS(2)	TS(1)		
1	2	2	3	0	0	27	77%
2	4	3	0	0	0	29	82%
3	4	2	0	1	0	30	85%
4	4	2	0	1	0	30	85%
5	3	3	0	1	0	29	82%
6	4	2	0	1	0	30	85%

7	4	2	0	1	0	30	85%
8	6	1	0	0	0	45	97%
Rata-Rata							84%

Pada Tabel 4.11 dapat dilihat terdapat 8 responden nelayan Teluk Kiluan. Pada pertanyaan pertama mendapatkan skor 77% dengan 2 responden menjawab sangat setuju, 2 responden menjawab setuju dan 3 responden menjawab netral atau cukup, dari data tersebut, berdasarkan analisis, terdapat 4 nelayan yang masih tergolong setuju bahwa aplikasi mudah digunakan akan tetapi masih ada 3 nelayan yang masih ragu akan kemudahan aplikasi sehingga dapat disimpulkan dari hasil para responden tersebut adalah aplikasi masih tergolong mudah digunakan akan tetapi masih memerlukan sedikit perbaikan dalam segi user experience untuk memudahkan dalam pemakaian. Pada pertanyaan kedua mendapatkan skor 82% dengan 4 responden sangat setuju dan 3 responden setuju, dari data tersebut dapat dianalisis bahwa mayoritas nelayan merasa sangat terbantu dalam memantau kondisi laut aman atau tidaknya untuk berlayar menangkap ikan dengan menggunakan aplikasi. Pada pertanyaan ketiga, keempat, keenam dan ketujuh, mendapatkan skor yang sama yaitu 85% dengan 4 responden sangat setuju, 2 responden setuju dan 1 responden kurang setuju, dari data tersebut dapat dianalisis bahwa mayoritas nelayan merasa sangat terbantu dalam mengecek tinggi gelombang, kecepatan angin, kecepatan ombak dan kecepatan angin, akan tetapi perlunya peningkatan pendekatan pengguna seperti penambahan visualisasi untuk memudahkan nelayan dalam memantau data-data tersebut. Pada pertanyaan kelima mendapatkan skor 82% dengan 3 responden sangat setuju, 3 responden setuju, dan 1 responden tidak setuju, dari data tersebut dapat dianalisis bahwa aplikasi masih tergolong membantu pihak nelayan dalam melihat arah angin akan tetapi perlunya penambahan pendekatan user experience kepada nelayan dalam segi visualisasi data arah angin kepada para nelayan. Pada pertanyaan kedelapan mendapatkan skor 97% dengan 6 responden sangat setuju, dan 2 responden setuju, dari data tersebut dapat dianalisis bahwa aplikasi sangat membantu nelayan dalam Menyusun rencana pelayaran untuk menangkap ikan. Dari semua skor yang ada, dengan menggunakan skala likert, didapatkan skor 84% yang memiliki arti

Sangat Setuju/puas/setuju, sehingga dapat dianalisis bahwa aplikasi masih sangat membantu para pihak nelayan untuk memantau kondisi laut Teluk Kiluan sehingga dapat disimpulkan bahwa aplikasi dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan.

Setelah melakukan analisis kuesioner dengan *skala likert*, selanjutnya dilakukan uji validasi dan reliabilitas pada kuesioner dengan SPSS.

		Correlations									
		P1	P2	P3	P4	P5	P6	P7	P8	Total	
P1	Pearson Correlation	1	.336	.336	.385	.545	.371	.371	.271	.559	
	Sig. (2-tailed)		.461	.461	.408	.206	.412	.412	.556	.192	
	N	7	7	7	7	7	7	7	7	7	
P2	Pearson Correlation	.336	1	1.000**	.479**	.566**	.798	.706	-.062	.901**	
	Sig. (2-tailed)	.461		<.001	<.001	.005	.075	.075	.895	.006	
	N	7	7	7	7	7	7	7	7	7	
P3	Pearson Correlation	.336	1.000**	1	.479**	.566**	.798	.706	-.062	.901**	
	Sig. (2-tailed)	.461	<.001		<.001	.005	.075	.075	.895	.006	
	N	7	7	7	7	7	7	7	7	7	
P4	Pearson Correlation	.385	.479**	.479**	1	.845**	.801	.801	-.132	.840	
	Sig. (2-tailed)	.408	<.001	<.001		.002	.031	.031	.779	.002	
	N	7	7	7	7	7	7	7	7	7	
P5	Pearson Correlation	.545	.566**	.566**	.845**	1	.854**	.854**	-.021	.882**	
	Sig. (2-tailed)	.206	.005	.005	.002		.014	.014	.880	<.001	
	N	7	7	7	7	7	7	7	7	7	
P6	Pearson Correlation	.371	.706	.706	.801	.854**	1	1.000**	.091	.904**	
	Sig. (2-tailed)	.412	.075	.075	.031	.014		<.001	.846	.002	
	N	7	7	7	7	7	7	7	7	7	
P7	Pearson Correlation	.371	.706	.706	.801	.854**	1.000**	1	.091	.904**	
	Sig. (2-tailed)	.412	.075	.075	.031	.014	<.001		.846	.002	
	N	7	7	7	7	7	7	7	7	7	
P8	Pearson Correlation	.271	-.062	-.062	-.132	.091	.091	.091	1	.130	
	Sig. (2-tailed)	.556	.895	.895	.779	.846	.846	.846		.781	
	N	7	7	7	7	7	7	7	7	7	
Total	Pearson Correlation	.559	.901**	.901**	.840	.882**	.904**	.904**	.130	1	
	Sig. (2-tailed)	.192	.006	.006	.002	<.001	.006	.006	.781		
	N	7	7	7	7	7	7	7	7	7	

** Correlation is significant at the 0.01 level (2-tailed).

* Correlation is significant at the 0.05 level (2-tailed).

** Correlation is significant at the 0.01 level (2-tailed).
* Correlation is significant at the 0.05 level (2-tailed).

Gambar 4.25 Correlations Pada Kuesioner Nelayan

Pada gambar di atas terdapat nilai *person correlation* yang akan dibandingkan nilai perhitungan yang di ambil dari data kolom total, sehingga dapat dilihat kevalidan dari kuesioner.

Tabel 4.13 Hasil Validasi Kuesioner Peneliti

Nomor	Perhitungan		(N= 4, a = 0.05)	Keterangan
1	0.559	<	0.754	Tidak Valid
2	0.901	>		Valid
3	0.901	>		Valid
4	0.940	>		Valid
5	0.982	>		Valid
6	0.904	>		Valid
7	0.904	>		Valid
8	0.130	<		Tidak Valid

Setelah melakukan validasi, maka dilakukan juga reliability untuk mengetahui tingkatan keandalan kuesioner.

Scale: ALL VARIABLES

Case Processing Summary

		N	%
Cases	Valid	7	100.0
	Excluded ^a	0	.0
	Total	7	100.0

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

Cronbach's Alpha	N of Items
.923	8

Gambar 4.26 *Reliability Statics* Pada Kuesioner Nelayan

Dari hasil validasi dan *reliability* dapat di analisis bahwa masih terdapat kuesioner yang tidak valid, akan tetapi dari *reliability* menghasilkan data yang lebih dari nilai 0.754 sehingga dapat di simpulkan bahwa kuesioner yang dilontarkan dapat di andalkan.

4.3 Pembahasan

Menggunakan metode *Dynamic Systems Development Method* (DSDM) diketahui bahwa aplikasi pemantauan lingkungan laut berbasis *mobile* android telah dikembangkan dengan baik. Hal ini dapat terjadi dikarenakan metode DSDM sendiri memiliki tahapan pengulangan kembali ke siklus awal atau fase sebelumnya bilamana terjadi masalah, penambahan fitur, ataupun perubahan *design* ketika dilakukan *review* antara pengembang dan klien terkait aplikasi, sehingga aplikasi berjalan sesuai yang diharapkan. Selain itu terdapat proses pengulangan siklus yang dimana membantu pihak pengembang untuk mengubah aplikasi di saat ada perubahan tampilan atau fitur pada aplikasi saat sedang di-*review* oleh pihak klien.

Aplikasi yang dikembangkan dengan menggunakan metode DSDM dinilai tepat sasaran dalam segi waktu dan anggaran, aplikasi juga berjalan dengan semestinya dan sesuai dengan yang diharapkan oleh pihak klien dan

pengembang itu sendiri. Telah dilakukan juga pembuktian dengan melakukan pengujian *black-box*. Pada pengujian *black-box* terlihat bahwa aplikasi telah berjalan sesuai dengan skenario yang telah dibuat. Selain itu dari pihak klien telah menyetujui keberhasilan pembuatan aplikasi yang dapat dilihat pada dokumen persetujuan pada lampiran. Serta aplikasi sudah dirilis dan sudah dapat digunakan oleh pihak peneliti dan nelayan Teluk Kiluan.

Pada pengujian kuesioner dibagi menjadi dua yaitu pengujian kepada pihak peneliti dan pihak nelayan. Pada pihak peneliti didapatkan skala skor 79.3% yang berarti Setuju/ Puas/ Baik, dari kuesioner tersebut terdapat pertanyaan-pertanyaan yang berhubungan dengan pengambilan data kondisi laut Teluk Kiluan. Sehingga dapat disimpulkan bahwa aplikasi membantu pihak peneliti dalam mengumpulkan data lingkungan laut Teluk Kiluan untuk penelitian lebih lanjut mereka. Pada pengujian kepada nelayan, didapatkan hasil 84% yang berarti Sangat Puas/ Sangat Setuju,/ Sangat Baik, dari kuesioner tersebut terdapat pertanyaan-pertanyaan yang berhubungan dengan pemantauan keadaan laut yang aman dalam menangkap ikan dengan melihat data-data yang ada, serta membantu dalam melakukan perencanaan penangkapan ikan. Sehingga dapat disimpulkan bahwa aplikasi dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan

Pada aplikasi sendiri memang bertujuan untuk memberikan data rata-rata keadaan laut Teluk Kiluan kepada peneliti, akan tetapi data tersebut belum dapat diunduh menjadi file mentah atau csv, serta aplikasi hanya bisa hanya bisa memunculkan grafik yang update setiap minggu, sehingga kurangnya visualisasi data tambahan berupa gambar pengelolaan dari data yang disimpan di server. Aplikasi juga belum dapat memvalidasi data peneliti untuk mengetahui apakah *user* yang melakukan registrasi merupakan peneliti asli atau masyarakat umum yang melakukan registrasi. Dari beberapa kelemahan dari aplikasi ini sendiri, diharapkan dapat ditambahkan pada penelitian lebih lanjut untuk menutupi setiap kekurangan agar aplikasi dapat berjalan lebih efisien.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan diperoleh beberapa kesimpulan sebagai berikut:

1. Metode *Dynamic Systems Development Method* (DSDM) dapat digunakan dengan baik dalam proses pengembangan aplikasi *mobile* berbasis android untuk memantau kondisi lingkungan laut. Pengulangan yang terjadi pada metode ini memudahkan pihak pengembang dalam melakukan perubahan setiap terjadi penambahan fitur dan perubahan *design* aplikasi. Metode ini juga memudahkan pengembang dalam pembuatan *design* dan fitur aplikasi dikarenakan adanya penekanan proses diskusi atau komunikasi antara pihak pengembang dan klien.
2. Berdasarkan pengujian dengan menggunakan kuesioner, didapatkan hasil rata-rata skor 79.3% dari pihak peneliti, hal ini dapat disimpulkan bahwa peneliti terbantu dalam melihat dan mengumpulkan data kondisi laut. Pada pengujian kepada nelayan, didapatkan hasil rata-rata skor 84% yang sehingga dapat disimpulkan bahwa aplikasi dapat membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan

5.2. Saran

Berdasarkan hasil penelitian yang telah dilakukan, adapun beberapa saran untuk penelitian lebih lanjut sebagai berikut:

1. Sebaiknya dibuatkan fitur visualisasi data kondisi laut dengan berupa gambar, tidak hanya grafik
2. Dibuatkan fitur *download* data menjadi CSV untuk mendapatkan data yang dapat diolah menjadi *model Machine Learning* (ML)
3. Dibuatkan validasi *user* bila registrasi sebagai peneliti.

DAFTAR PUSTAKA

- [1] K. Gilang Widagdyo¹ and S. Bhudiharty², “MODEL PENGEMBANGAN DESTINASI WISATA TELUK KILUAN MELALUI OPTIMALISASI FAKTOR-FAKTOR DAYA TARIK EKOWISATA,” *Jurnal Industri Pariwisata*, vol. 1, no. 1, 2018.
- [2] D. Abad *et al.*, “PEMETAAN MULTI RISIKO BENCANA PADA KAWASAN STRATEGIS DI KABUPATEN TANGGAMUS MAPPING OF DISASTER MULTI-RISK ASSESSMENT FOR STRATEGIC AREAS IN TANGGAMUS DISTRICT,” 2017.
- [3] rvk and rvn, “Pantai Kiluan Lampung Juga Diterjang Tsunami, 1 Balita Tewas,” *detiknews*, Lampung, pp. 1–1, Dec. 23, 2018.
- [4] V. Julianto *et al.*, “PENGARUH SOSIALISASI KESIAPSIAGAAN BENCANA TERHADAP PENGETAHUAN SISWA DALAM MENGHADAPI BENCANA TSUNAMI DI DESA KILUAN NEGERI 1,” Januari-Juni, 2019.
- [5] P. Literasi *et al.*, “Article Info,” *Jurnal Warta LPM*, vol. 23, no. 2, pp. 165–179, 2020, [Online]. Available: <http://journals.ums.ac.id/index.php/warta>
- [6] L. Cutroneo *et al.*, “Near real-time monitoring of significant sea wave height through microseism recordings: Analysis of an exceptional sea storm event,” *J Mar Sci Eng*, vol. 9, no. 3, Mar. 2021, doi: 10.3390/jmse9030319.
- [7] N. Anwar, A. Fansuri, A. M. Widodo, K. K. Juman, and B. Ulum, “Modelling IoT Untuk Monitoring Suhu dan pH Budidaya Ikan Nila Metode Dynamic System Development Method (DSDM).”
- [8] B. G. Jayatilleke, G. R. Ranawaka, C. Wijesekera, and M. C. B. Kumarasinha, “Development of mobile application through design-based research,” *Asian Association of Open Universities Journal*, vol. 13, no. 2, pp. 145–168, Apr. 2019, doi: 10.1108/AAOUJ-02-2018-0013.
- [9] F. Awanda Alviansyah and E. Ramadhani, “Implementasi Dynamic Application Security Testing pada Aplikasi Berbasis Android.”
- [10] A. Lawal and R. C. Ogbu, “A COMPARATIVE ANALYSIS OF AGILE AND WATERFALL SOFTWARE DEVELOPMENT METHODOLOGIES.”

- [11] B. AKM Zahidul Islam, A. Ferworn, A. Zahidul Islam α , and A. Ferworn σ , “A Comparison between Agile and Traditional Software Development Methodologies,” 2020.
- [12] S. Al-Saqqa, S. Sawalha, and H. Abdelnabi, “Agile software development: Methodologies and trends,” *International Journal of Interactive Mobile Technologies*, vol. 14, no. 11, pp. 246–270, 2020, doi: 10.3991/ijim.v14i11.13269.
- [13] A. Przybyłek and M. E. Morales-Trujillo, Eds., *Advances in Agile and User-Centred Software Engineering*, vol. 376. in Lecture Notes in Business Information Processing, vol. 376. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-37534-8.
- [14] P. D. Larasati, “Analisis dan Perancangan Sistem E-Learning Classroom for Academic Menggunakan Dynamic System Development Method (DSDM) Studi Kasus: School of Engineering and Technology Tanri Abeng University,” 2020.
- [15] D. Ayu, N. Wulandari, M. Dika Atthariq, W. D. Nanda, and L. Yusuf, “IMPLEMENTASI DYNAMIC SYSTEM DEVELOPMENT METHOD (DSDM) PADA SISTEM INFORMASI MANAJEMEN BENGKEL MOBIL BERBASIS WEB,” *Sistem Informasi* /, vol. 8, no. 1, pp. 10–17, 2021.
- [16] V. H. Pranatawijaya, W. Widiatry, R. Priskila, and P. B. A. A. Putra, “Penerapan Skala Likert dan Skala Dikotomi Pada Kuesioner Online,” *Jurnal Sains dan Informatika*, vol. 5, no. 2, pp. 128–137, Dec. 2019, doi: 10.34128/jsi.v5i2.185.
- [17] D. Friyansyah, N. P. Hardosubroto, W. H. Simamora, Y. S. Sari, P. Studi, and S. Informasi, “Aplikasi Café Point Of Sales (CAPOS) dengan Dynamic System Development Method(DSDM) (Studi Kasus Ropang LOILO).”
- [18] I. And and D. Expert, “Penerapan Dynamic System Development Method Pada Sistem Monitoring Status Gizi Balita,” 2020. [Online]. Available: <http://index.unper.ac.id>
- [19] R. Y. Pratama and M. Yuhendri, “JTEV (JURNAL TEKNIK ELEKTRO DAN VOKASIONAL) Monitoring Turbin Angin Menggunakan Smartphone Android”, [Online]. Available: <http://ejournal.unp.ac.id/index.php/jtev/index>

- [20] S. Andriyanto, P. Manufaktur Negeri Bangka Belitung, and J. Teknik Elektro dan Informatika, "Monitoring Aliran Arus Pasang Surut Air Laut Berbasis Arduino," *Jurnal ELECTRA: Electrical Engineering Articles*, vol. 2, no. 1, pp. 1–8, 2021.
- [21] A. B. Santoso, A. B. Prasetijo, and I. P. Windasari, "PERANCANGAN APLIKASI ANDROID KONSULTASI KESEHATAN MENGGUNAKAN REACT NATIVE," *Jurnal Ilmu Teknik dan Komputer*, vol. 6, no. 1, 2022.
- [22] M. Murdiaty, A. Angela, and C. Sylvia, "Pengelompokan Data Bencana Alam Berdasarkan Wilayah, Waktu, Jumlah Korban dan Kerusakan Fasilitas Dengan Algoritma K-Means," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 4, no. 3, p. 744, Jul. 2020, doi: 10.30865/mib.v4i3.2213.
- [23] D. Sugawara, "Numerical modeling of tsunami: advances and future challenges after the 2011 Tohoku earthquake and tsunami," *Earth-Science Reviews*, vol. 214. Elsevier B.V., Mar. 01, 2021. doi: 10.1016/j.earscirev.2020.103498.
- [24] P. Siklon Tropis Terhadap Tinggi Gelombang di Wilayah Selatan Jawa, dan Nusa Tenggara, K. Islamiyah, K. Ngurah Suarbawa, and K. Sumaja, "Effect of Tropical Cyclones on High Waves in the Southern Regions of Java, (Kholidatul Islamiyah, dkk)."
- [25] S. Nasiri, Y. Rhazali, M. Lahmer, and A. Adadi, "From User Stories to UML Diagrams Driven by Ontological and Production Model." [Online]. Available: www.ijacsa.thesai.org
- [26] R. Gh Alsarraj, A. M. Altaie, and A. A. Fadhil, "Designing and implementing a tool to transform source code to UML diagrams," vol. 9, no. 2, pp. 430–440, 2021.
- [27] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "A Different Approach on Automated Use Case Diagram Semantic Assessment," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, pp. 496–505, 2021, doi: 10.22266/IJIES2021.0228.46.
- [28] B. A, "NoSQL Implementation of a Conceptual Data Model: UML Class Diagram to a Document Oriented Model," *International Journal of Database*

- Management Systems*, vol. 10, no. 2, pp. 01–10, Apr. 2018, doi: 10.5121/ijdms.2018.10201.
- [29] S. Al-Fedaghi, “Validation: Conceptual versus Activity Diagram Approaches.” [Online]. Available: www.ijacsa.thesai.org
 - [30] “ANALISA DAN PERANCANGAN SISTEM INFORMASI TEXT CHATTING BERBASIS”.
 - [31] “CROSS-PLATFORM MOBILE APPLICATION DEVELOPMENT WITH REACT NATIVE,” 2019.
 - [32] A. Wirfs-Brock and B. Eich, “JavaScript: The first 20 years,” *Proceedings of the ACM on Programming Languages*, vol. 4, no. HOPL, Jun. 2020, doi: 10.1145/3386327.
 - [33] S. Haque, Z. Eberhart, A. Bansal, and C. McMillan, “Semantic Similarity Metrics for Evaluating Source Code Summarization,” in *IEEE International Conference on Program Comprehension*, IEEE Computer Society, 2022, pp. 36–47. doi: 10.1145/nnnnnnn.nnnnnnn.
 - [34] M. T. Fakultas and I. Komputer, “Pengembangan Sistem Stock Opname Berbasis Mobile Application Using SDLC Methode,” 2021. [Online]. Available: <https://doi.org/10.25047/jtit.v8i1.198>
 - [35] P. Black, D. Fahrezi, and F. N. Khasanah, “Pengujian Black Box Dan Kuesioner Pada Game Feed The Animal,” vol. 3, no. 2, pp. 193–202, 2019.
 - [36] “langkah-langkah dalam menyusun kuesioner”.
 - [37] F. N. Khasanah and S. Murdowo, “PENGUJIAN BETA PADA APLIKASI GAME EDUKASI PENGENALAN DASAR ISLAM MELALUI KUESIONER.”
 - [38] S. Villamil, C. Hernández, and G. Tarazona, “An overview of internet of things,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 5, pp. 2320–2327, Oct. 2020, doi: 10.12928/TELKOMNIKA.v18i5.15911.
 - [39] Z. Wu, K. Qiu, and J. Zhang, “A smart microcontroller architecture for the internet of things,” *Sensors (Switzerland)*, vol. 20, no. 7, Apr. 2020, doi: 10.3390/s20071821.

- [40] F. Kurniawan, I. #1, F. Fahurian, and A. Hafiz, “Expert-Jurnal Management Sistem Informasi dan Teknologi RANCANG BANGUN APLIKASI CLOUD STORAGE DENGAN ANGGULAR DAN FIREBASE BERBASIS ANDROID”.
- [41] S. K. Dirjen, P. Riset, D. Pengembangan, R. Dikti, and I. Firman Maulana, “Terakreditasi SINTA Peringkat 2 Penerapan Firebase Realtime Database pada Aplikasi E-Tilang Smartphone berbasis Mobile Android,” *masa berlaku mulai*, vol. 1, no. 3, pp. 854–863, 2017.
- [42] E. A. W. Sanad, “Pemanfaatan Realtime Database di Platform Firebase Pada Aplikasi E-Tourism Kabupaten Nabire,” *Jurnal Penelitian Enjiniring*, vol. 22, no. 1, pp. 20–26, May 2019, doi: 10.25042/jpe.052018.04.
- [43] L. F. D. Alves and A. D. O. Costa Junior, “Be a Maker: Um Aplicativo de Compartilhamento de Materiais Instrucionais Sobre Robótica Educacional,” *Sociedade Brasileira de Computacao - SB*, Nov. 2020, pp. 132–139. doi: 10.5753/cbie.wcbie.2020.132.
- [44] P. Dullabh, L. Hovey, K. Heaney-Huls, N. Rajendran, A. Wright, and D. F. Sittig, “Application Programming Interfaces (APIs) in Health Care: Findings from a Current-State Assessment,” in *Studies in Health Technology and Informatics*, IOS Press, 2019, pp. 201–206. doi: 10.3233/SHTI190164.
- [45] B. Cahyo Santoso, Y. Natasya, S. Willian, and F. Alfando, “Tinjauan Pustaka Sistematis terhadap Basis Data MongoDB.”
- [46] M. M. Eyada, W. Saber, M. M. el Genidy, and F. Amer, “Performance Evaluation of IoT Data Management Using MongoDB Versus MySQL Databases in Different Cloud Environments,” *IEEE Access*, vol. 8, pp. 110656–110668, 2020, doi: 10.1109/ACCESS.2020.3002164.
- [47] N. Miftahul Janna and D. Pembimbing, “KONSEP Uji VALIDITAS DAN RELIABILITAS DENGAN MENGGUNAKAN SPSS.”
- [48] M. Rizqy Riyono, A. Dwi Churniawan Program Studi, and J. Sistem Informasi STIKOM Surabaya STIMIK STIKOM Surabaya Jl Raya Kedung Baruk, “ANALISIS PENGARUH WEBSITE STIKOM INSTITUTIONAL REPOSITORIES (SIR) PADA INSTITUT BISNIS DAN INFORMATIKA STIKOM SURABAYA,” 2016.

- [49] F. D. P. Anggraini, A. Aprianti, V. A. V. Setyawati, and A. A. Hartanto, "Pembelajaran Statistika Menggunakan Software SPSS untuk Uji Validitas dan Reliabilitas," *Jurnal Basicedu*, vol. 6, no. 4, pp. 6491–6504, May 2022, doi: 10.31004/basicedu.v6i4.3206.

Lampiran

Lampiran I Kode Pada Mobile

Adapun Lampiran Untuk Menampilkan Mobile sebagai berikut :

1. app.tsx

```
import React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

// page
import SplashScreen from './src/page/splash_screen';
import BerandaPage from './src/page/beranda_page';
import DetailPage from './src/page/detail_page';
import LoginPage from './src/page/login_page';
import ProfilPage from './src/page/profil_page';
import ProfilAfter from './src/page/profil_after';
import ProfilBefore from './src/page/profil_before';
import RegisterPage from './src/page/register_page';
import Information from './src/page/information';
import ArahAngin from './src/page/arrah_angin';
import RamalanCuaca from './src/page/ramalan_cuaca';
//redux
import { Provider } from 'react-redux';
import { Store } from './src/redux/store';

type RootStackParamList = {
  SplashScreen: undefined,
  BerandaPage: undefined,
  DetailPage: undefined,
  LoginPage: undefined,
  ProfilPage: undefined,
  ProfilAfter: undefined,
  ProfilBefore: undefined,
  RegisterPage: undefined,
  Information: undefined,
  ArahAngin: undefined,
  RamalanCuaca: undefined
}

const Stack = createNativeStackNavigator<RootStackParamList>();

const App = () => {
  return (
    <Provider store={Store}>
      <NavigationContainer>
        <Stack.Navigator
          screenOptions={{ headerShown: false }}>
          <Stack.Screen name="SplashScreen"
            component={SplashScreen} />
          <Stack.Screen name="BerandaPage"
            component={BerandaPage} />
        </Stack.Navigator>
      </NavigationContainer>
    </Provider>
  );
}
```

```

        <Stack.Screen                                name="DetailPage"
        component={DetailPage} />
        <Stack.Screen                                name="LoginPage"
        component={LoginPage} />
        <Stack.Screen                                name="ProfilPage"
        component={ProfilPage} />
        <Stack.Screen                                name="ProfilAfter"
        component={ProfilAfter} />
        <Stack.Screen                                name="ProfilBefore"
        component={ProfilBefore} />
        <Stack.Screen                                name="RegisterPage"
        component={RegisterPage} />
        <Stack.Screen                                name="Information"
        component={Information} />
        <Stack.Screen                                name="ArahAngin"
        component={ArahAngin} />
        <Stack.Screen                                name="RamalanCuaca"
        component={RamalanCuaca} />
      </Stack.Navigator>
    </NavigationContainer>
  </Provider>
);
};

```

```
export default App;
```

2. src/page/arrah_angin.tsx

```

import React, { useEffect, useState } from "react";
import {
  SafeAreaView,
  StatusBar,
  StyleSheet,
  View,
  Text,
  TouchableOpacity,
  Image,
  Platform,
  PermissionsAndroid,
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import { useNavigation } from "@react-navigation/native";
import Geolocation from 'react-native-geolocation-service';
import CompassHeading from 'react-native-compass-heading';
import { useRoute } from "@react-navigation/native";
import { compass, convertCompass } from "../utils/compass";
import database from '@react-native-firebase/database';

const ArahAngin = () => {

```

```

const navigate = useNavigation()

const [compassHeader, setCompassHeader] =
useState<number>(0)

// const [value, setValue] = useState<number>(0)

const [arahAngin, setArahAngin] =
useState<string>('')

// const [derajat, setDerajat] =
useState<string>('')
const [derajat, setDerajat] = useState<number>(0)

//@ts-ignore
const page = useRoute().params.page

async function requestPermissions() {
  if (Platform.OS === 'ios') {
    //@ts-ignore
    Geolocation.requestAuthorization();
    //@ts-ignore
    Geolocation.setRNConfiguration({
      skipPermissionRequests: false,
      authorizationLevel: 'whenInUse',
    });
  }

  if (Platform.OS === 'android') {
    await PermissionsAndroid.request(
PermissionsAndroid.PERMISSIONS.ACCESS_FINE_LOCATION,
    );
  }
}

database()
.ref('/')
.on('value', snapshot => {
  // setValue(compass({ data:
snapshot.val().angin.arah }))
  setArahAngin(snapshot.val().angin.arah)
  setDerajat(snapshot.val().angin.derajat)
});

useEffect(() => {
  requestPermissions().then(() => {
    const degree_update_rate = 3;

    CompassHeading.start(degree_update_rate, ({
heading, accuracy }) => {
      console.log('CompassHeading: ',
heading, accuracy);
      setCompassHeader(heading)
    });
  });
})

```

```

        return () => {
            CompassHeading.stop();
        };

    }, [])

    return (
        <SafeAreaView
            style={[stylesGlobal.backgroundWhite,
                styles.container]}>
            <StatusBar
                animated={true}

                backgroundColor={stylesGlobal.backgroundWhite.backgroundColor}
            />

            <View style={styles.concomponent}>
                <TouchableOpacity
                    style={styles.titleBack}          onPress={() =>
                    navigate.goBack()}>
                    <MaterialIcons      name="arrow-back-
                    ios" size={20} color="#2F5664" />
                    <Text style={[stylesGlobal.header2,
                    stylesGlobal.colorPremier]}>
                        {page}
                    </Text>
                </TouchableOpacity>
                <View style={styles.compass}>
                    <Text style={[stylesGlobal.header1,
                    stylesGlobal.colorPremier]}>
                        {derajat + ' °'}
                    </Text>
                    <Image
                        source={require('../assets/compass.png')} style={[
                            styles.image,
                            { transform: [{ rotate: `${(-
                                compassHeader) - (360 - derajat)}deg` }],
                            } ] />
                    <Text
                        style={[stylesGlobal.colorPremier,
                            stylesGlobal.header1]}>
                            {arahAngin}
                        </Text>
                    </View>

                </View>
            </SafeAreaView>
        )
    }

    const styles = StyleSheet.create({
        container: {
            flex: 1,
            paddingVertical: 30,
        },
    },

```

```

        concomponent: {
            width: '100%',
            height: '100%',
            flexDirection: "column",
        },
        titleBack: {
            flexDirection: "row",
            alignItems: 'center',
            marginBottom: 30,
            paddingHorizontal: 20,
        },
        compass: {
            height: '95%',
            width: "100%",
            justifyContent: 'space-between',
            alignItems: 'center',
            padding: 90
        },
        image: {
            height: 270,
            width: 270
        }
    })

export default ArahAngin

```

3. src/page/beranda.tsx

```

import React, { useEffect, useState } from "react";
import {
    SafeAreaView,
    StatusBar,
    StyleSheet,
    View,
    Text,
    TouchableOpacity
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import Detail from "../component/detail";
import { useRoute } from "@react-navigation/native";
import { useSelector } from "react-redux";
import { getAvarageApi, getGrafikApi } from '../utils/api'
import { getDataGrafik, getDataAverage } from '../utils/axios'
import Loading from "../component/loading/loading";
import { useNavigation } from "@react-navigation/native";

const DetailPage = () => {

    const navigate = useNavigation()

    const { month, year } = useSelector(
        //@ts-ignore
        state => state.userReducer
    )

```

```

    )

    const [dataGrafik, setDataGrafik] =
    useState<number[]>([])

    const [labelGrafik, setLabelGrafik] =
    useState<any>([])

    const [dataAverage, setDataAvefarge] =
    useState<any>(null)

    const [isclear, setClear] =
    useState<boolean>(false)

    const [isLoading, setLoading] =
    useState<boolean>(true)

    //@ts-ignore
    const name = useRoute().params.name
    //@ts-ignore
    const page = useRoute().params.page

    const countries = [2023, 2024, 2025, 2026, 2027]
    const bulan = ['Januari', 'Februari', 'Maret',
    "April", "Mei", 'Juni', "Juli", "Agustus", "September",
    "Oktober", "November", "Desember"]

    const data = {
      datasets: {
        color: (opacity = 255) => `rgba(255, 255,
    255, ${opacity})`, // optional
        strokeWidth: 2 // optional
      }
    },
  };

  useEffect(() => {
    getDataGrafik({
      data: getGrafikApi({
        name: name,
        month: month,
        year: year
      }),
      setData: setDataGrafik,
      setLabel: setLabelGrafik,
      setAfter: setClear
    })

    getDataAverage({
      data: getAvarageApi({
        name: name,
        month: month,
        year: year
      }),
      setData: setDataAvefarge,
      setLabel: setDataAvefarge,
      setAfter: setLoading
    })
  })

```



```

        return () => {
            setDataGrafik([])
            setLabelGrafik([])
            setClear(false)
            setLoading(true)
        }
    }, [month, year])

    return (
        <SafeAreaView
            style={ [stylesGlobal.backgroundWhite,
                styles.container]} >
            <StatusBar
                animated={true}

backgroundColor={stylesGlobal.backgroundWhite.background
Color}
            />
            {
                isclear == true && isLoading == false ?
                <View style={styles.concomponent}>
                    <TouchableOpacity
                        style={styles.titleBack}          onPress={()          =>
navigate.goBack()} >
                        <MaterialIcons name="arrow-
back-ios" size={20} color="#2F5664" />
                        <Text
                            style={ [stylesGlobal.header2,
                                stylesGlobal.colorPremier]} >
                                {page}
                        </Text>
                    </TouchableOpacity>

                    <Detail.FilterDetail
                        year={countries}
                        date={bulan}
                    />
                    <Text
                        style={ [styles.text,
                            stylesGlobal.subtitle, stylesGlobal.colorPremier]} >
                        *Berdasarkan Data Lapangan
                    </Text>
                    <Detail.GrafikDetail
                        label={labelGrafik}
                        data={dataGrafik}
                        color={data.datasets.color}

strokeWidth={data.datasets.strokeWidth}
                    />
                    <Detail.AverageDetail
                        data={dataAverage}
                    />
                </View> : <Loading />
            }
        </SafeAreaView>
    )
}

const styles = StyleSheet.create({

```

```

        container: {
            flex: 1,
            paddingVertical: 30,
        },
        concomponent: {
            width: '100%',
            height: '100%',
            flexDirection: "column",
        },
        titleBack: {
            flexDirection: "row",
            alignItems: 'center',
            marginBottom: 30,
            paddingHorizontal: 20,
        },
        text: {
            paddingHorizontal: 20,
            marginBottom: 10
        }
    })

export default DetailPage

```

4. src/page/detail_page.tsx

```

import React, { useEffect, useState } from "react";
import {
    SafeAreaView,
    StatusBar,
    StyleSheet,
    View,
    Text,
    TouchableOpacity
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import Detail from "../component/detail";
import { useRoute } from "@react-navigation/native";
import { useSelector } from "react-redux";
import { getAvarageApi, getGrafikApi } from '../utils/api'
import { getDataGrafik, getDataAverage } from '../utils/axios'
import Loading from "../component/loading/loading";
import { useNavigation } from "@react-navigation/native";

const DetailPage = () => {

    const navigate = useNavigation()

    const { month, year } = useSelector(
        //@ts-ignore
        state => state.userReducer
    )

    const [dataGrafik, setDataGrafik] =

```

```

useState<number[]>([])

    const      [labelGrafik,      setLabelGrafik]      =
useState<any>([])

    const      [dataAverage,      setDataAvefarge]      =
useState<any>(null)

    const      [isclear,      setClear]      =
useState<boolean>(false)

    const      [isLoading,      setLoading]      =
useState<boolean>(true)

    //@ts-ignore
    const name = useRoute().params.name
    //@ts-ignore
    const page = useRoute().params.page

    const countries = [2023, 2024, 2025, 2026, 2027]
    const bulan = ['Januari', 'Februari', 'Maret',
"April", "Mei", 'Juni', "Juli", "Agustus", "September",
"Oktober", "November", "Desember"]

    const data = {
      datasets: {
        color: (opacity = 255) => `rgba(255, 255,
255, ${opacity})`, // optional
        strokeWidth: 2 // optional
      }
    },
  };

  useEffect(() => {
    getDataGrafik({
      data: getGrafikApi({
        name: name,
        month: month,
        year: year
      }),
      setData: setDataGrafik,
      setLabel: setLabelGrafik,
      setAfter: setClear
    })

    getDataAverage({
      data: getAvarageApi({
        name: name,
        month: month,
        year: year
      }),
      setData: setDataAvefarge,
      setLabel: setDataAvefarge,
      setAfter: setLoading
    })

    return () => {
      setDataGrafik([])
      setLabelGrafik([])
    }
  })

```

```

        setClear(false)
        setLoading(true)
    }, [month, year])

    return (
        <SafeAreaView
            style={[stylesGlobal.backgroundWhite,
                styles.container]}>
            <StatusBar
                animated={true}

backgroundColor={stylesGlobal.backgroundWhite.background
Color}
            />
            {
                isclear == true && isLoading == false ?
                <View style={styles.concomponent}>
                    <TouchableOpacity
                        style={styles.titleBack}          onPress={() =>
navigate.goBack()}>
                        <MaterialIcons name="arrow-
back-ios" size={20} color="#2F5664" />
                        <Text
                            style={[stylesGlobal.header2,
                                stylesGlobal.colorPremier]}>
                                {page}
                        </Text>
                    </TouchableOpacity>

                    <Detail.FilterDetail
                        year={countries}
                        date={bulan}
                    />
                    <Text
                        style={[styles.text,
                            stylesGlobal.subtitle, stylesGlobal.colorPremier]}>
                        *Berdasarkan Data Lapangan
                    </Text>
                    <Detail.GrafikDetail
                        label={labelGrafik}
                        data={dataGrafik}
                        color={data.datasets.color}

strokeWidth={data.datasets.strokeWidth}
                    />
                    <Detail.AverageDetail
                        data={dataAverage}
                    />
                </View> : <Loading />
            }
        </SafeAreaView>
    )
}

const styles = StyleSheet.create({
    container: {
        flex: 1,
        paddingVertical: 30,

```

```

    },
    concomponent: {
      width: '100%',
      height: '100%',
      flexDirection: "column",
    },
    titleBack: {
      flexDirection: "row",
      alignItems: 'center',
      marginBottom: 30,
      paddingHorizontal: 20,
    },
    text: {
      paddingHorizontal: 20,
      marginBottom: 10
    }
  })
}

export default DetailPage

```

5. src/page/information.tsx

```

import {
  StyleSheet,
  Text,
  SafeAreaView,
  StatusBar,
  ScrollView,
  TouchableOpacity,
  View,
} from "react-native"
import stylesGlobal from "../utils/global_style"
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import { useNavigation } from "@react-navigation/native";
import CardInformation from "../component/card_information/card_information";
import { timData, pengenalan } from "../utils/data_tim";

const Information = () => {

  const navigate = useNavigation()

  const goBack = () => navigate.goBack()

  return (
    <SafeAreaView style={[styles.container, stylesGlobal.backgroundWhite]}>
      <StatusBar
        animated={true}
        backgroundColor={stylesGlobal.backgroundWhite.backgroundColor}
      />
      <TouchableOpacity style={styles.titleBack} onPress={goBack}>
        <MaterialIcons name="arrow-back-ios" size={20} color="#2F5664"
      />
      <Text style={[stylesGlobal.header2, stylesGlobal.colorPremier]}>
        Informasi
      </Text>
    </TouchableOpacity>
  )
}

```

```

    <View style={stylesGlobal.enter30} />
    <ScrollView>
      <View style={styles.margin}>
        <Text style={[stylesGlobal.colorPremier, stylesGlobal.header1,
styles.text]}>Pengenalan</Text>
        <View style={[styles.line, stylesGlobal.backgroundPremier]} />
        <Text style={[styles.pengenalan, stylesGlobal.colorPremier,
stylesGlobal.subtitle]}>{pengenalan}</Text>
      </View>
      {
        timData.map((statement) => (
          <View style={styles.margin}>
            <Text style={[stylesGlobal.colorPremier, stylesGlobal.header1,
styles.text]}>{statement.tittle}</Text>
            <View style={[styles.line, stylesGlobal.backgroundPremier]} />
            <View style={stylesGlobal.enter20} />
            {
              statement.data.map((state) => (
                <CardInformation
                  email={state.email}
                  name={state.name}
                  prodi={state.instansi}
                />
              ))
            }
          </View>
        ))
      }
    </ScrollView>
  </SafeAreaView>
)
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  view: {
    justifyContent: 'center',
    alignItems: 'center',
  },
  titleBack: {
    flexDirection: "row",
    alignItems: 'center',
    paddingHorizontal: 20,
  },
  text: {
    marginHorizontal: 20
  },
  line: {
    width: '90%',
    height: 2,
    marginHorizontal: 20,
    marginTop: 5
  }
});

```

```

    },
    margin: {
      marginBottom: 20
    },
    pengenalan: {
      paddingHorizontal: 20,
      marginTop: 10,
      textAlign: "justify",
      // color: "black"
    }
  }
})

```

export default Information

6. src/page/login_page.tsx

```

import React, { useEffect, useState } from "react";
import {
  SafeAreaView,
  StatusBar,
  StyleSheet,
  View,
  Text,
  TouchableOpacity,
  ScrollView,
} from "react-native";
import stylesGlobal from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import LogoSFAD from "../component/image/image";
import TextInputData from "../component/textInput/textInput";
import ButtonInput from "../component/button_input/button_input";
import { useNavigation } from "@react-navigation/native";
import { namePage } from "../utils/namePage";
import { AlertStop } from "../component/alert/alert";
import { postLogin } from "../utils/axios";
import Loading from "../component/loading/loading";
import AsyncStorage from "@react-native-async-storage/async-storage";
import { useDispatch, useSelector } from "react-redux";
import { getUserName, getIdUser } from "../redux/action";
import axios from "axios";
import { getDataUserById } from "../utils/api";

const LoginPage = () => {

  const dispatch = useDispatch()

  const { idUser } = useSelector(
    //@ts-ignore
    state => state.userReducer
  )

  const navigate = useNavigation()

```

```

        const [email, setEmail] = useState<string>('');
        const [password, setPassword] =
        useState<string>('');
        const [isLoading, setLoading] =
        useState<boolean>(false)

        const goback = () => navigate.goBack()
        //@ts-ignore
        const gotoRegister = () =>
        navigate.navigate(namePage.REGISTER_PAGE)
        //@ts-ignore
        const gotoBeranda = () =>
        navigate.navigate(namePage.BERANDA_PAGE)

        const checkUser = async () => {
            await AsyncStorage.getItem('id')
                .then(async response => {
                    console.log(response)

                    if (response !== null) {

                        if (idUser === '') {
                            await
                            axios.get(getDataUserById({ id: response }))
                                .then(data => {

                                    console.log(data.data.data)

                                    dispatch(getUserName(data.data.data.name))

                                    dispatch(getIdUser(data.data.data.id))
                                    setLoading(false)
                                }).finally(() =>
                                    gotoBeranda())
                        }
                    }
                    else {
                        setLoading(false)
                    }
                })
        }

        const input = () => {
            if (email === '' || password === '') {
                AlertStop({
                    title: 'Perikasa Data',
                    message: 'Email dan Password anda belum
                    lengkap',
                })
            }
            else {
                setLoading(true)
                postLogin({
                    email: email,
                    password: password,
                    setData: setLoading,
                    next: checkUser

```



```

    ))
  }
}

useEffect(() => {

}, [isLoading])

return (
  <>
    {
      isLoading ? <Loading /> :
      <SafeAreaView
style={ [stylesGlobal.backgroundWhite,
styles.container]}>
        <StatusBar
          animated={true}

backgroundColor={stylesGlobal.backgroundWhite.background
Color}

        />
        <ScrollView
style={styles.concomponent}>

          <TouchableOpacity
style={styles.titleBack} onPress={goback}>
            <MaterialIcons
name="arrow-back-ios" size={20} color="#2F5664" />
            <Text
style={ [stylesGlobal.header2,
stylesGlobal.colorPremier]}>

              Masuk
            </Text>
          </TouchableOpacity>
          <View
style={styles.imageStyle}>

            <LogoSFAD size={250} />
          </View>

          <TextInputData
            setData={setEmail}
            data={email}
            placeholder={'Maukan
Email'}

            title={'Email'}
            isPasword={false}
          />

          <TextInputData
            setData={setPassword}
            data={password}
            placeholder={'Maukan
Password'}

            title={'Password'}
            isPasword={true}
          />
          <View
style={stylesGlobal.enter20} />

```

```

                                <ButtonInput action={input}
tittle={'Masuk'} />
                                <View
style={styles.daftar}>
                                <Text
style=[stylesGlobal.header3, { color: '#000' }]]>Belum
Punya Akun ? </Text>
                                <TouchableOpacity
onPress={gotoRegister}>
                                <Text
style=[stylesGlobal.colorPremier,
stylesGlobal.header3]]>DAFTAR</Text>
                                </TouchableOpacity>
                                </View>
                                </ScrollView>
                                </SafeAreaView>
                                }
                                </>
                                )
                                }

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  concomponent: {
    width: '100%',
    height: '100%',
    flexDirection: "column",
  },
  titleBack: {
    flexDirection: "row",
    alignItems: 'center',
    marginBottom: 30,
    paddingHorizontal: 20,
  },
  marginContainer: {
    marginHorizontal: 20,
  },
  imageStyle: {
    justifyContent: "center",
    alignItems: 'center',
    marginVertical: 30,
  },
  daftar: {
    justifyContent: 'center',
    alignItems: 'center',
    flexDirection: "row",
    marginTop: 10,
  }
})

export default LoginPage

```

7. src/page/profile_after.tsx

```
import React, { useState, useEffect } from "react";
```

```

import {
  SafeAreaView,
  StatusBar,
  StyleSheet,
  View,
  TouchableOpacity,
  Text,
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import LogoSFAD from "../component/image/image";
import MaterialIcons from "react-native-vector-
icons/MaterialIcons";
import SelfData from
"../component/self_data/self_data";
import ButtonInput from
"../component/button_input/button_input";
import { userData } from "../utils/axios";
import { getUsername, getIdUser } from
"../redux/action";
import { useSelector, useDispatch } from "react-redux";
import Loading from "../component/loading/loading";
import { useNavigation } from "@react-
navigation/native";
import AsyncStorage from "@react-native-async-
storage/async-storage";
import { namePage } from "../utils/namePage";
import { AlertShow } from "../component/alert/alert";
import Ionicon from "react-native-vector-
icons/Ionicons";
import database from '@react-native-firebase/database';

const ProfilAfter = () => {

  const dispatch = useDispatch()

  const navigate = useNavigation()

  const [name, setName] = useState<string>('')
  const [instansi, setInstansi] =
useState<string>('')
  const [email, setEmails] = useState<string>('')
  const [isLoading, setLoadings] =
useState<boolean>(true)
  const [isReady, setReady] =
useState<boolean>(false)

  database()
    .ref('/')
    .on('value', snapshot => {
      setReady(snapshot.val().ready)
    })

  const { idUser } = useSelector(
    //@ts-ignore
    state => state.userReducer
  )

  const goBack = () => navigate.goBack()

```



```

name="arrow-back-ios" size={20} color="#2F5664" />
      <Text
        style={ [stylesGlobal.header2,
          stylesGlobal.colorPremier] }>
          Profil
        </Text>
      </TouchableOpacity>
    <TouchableOpacity
      style={styles.titleBack} onPress={goInformation}>
      <Icon
        name="information-circle-outline" size={24}
        color="#2F5664" />
      </TouchableOpacity>
    </View>
  <View
    style={styles.imageStyle}>
    <LogoSFAD size={200} />
  </View>
  {
    isReady == false ? null :
    <>
      <SelfData
        data={name} title={'Name Lengkap'} />
      <SelfData
        data={instansi} title={'Instansi'} />
      <SelfData
        data={email} title={'Email'} />
      <View
        style={stylesGlobal.enter20} />
      <ButtonInput
        action={alertShow}
        tittle={'Keluar'}
      />
    </>
  }
</SafeAreaView>
}
</>
)
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  imageStyle: {
    justifyContent: "center",
    alignItems: 'center',
    marginVertical: 30,
    marginBottom: 20,
  },
  viewHeader: {
    justifyContent: 'space-between',
    flexDirection: 'row',

```

```

        alignItems: 'center'
      },
      titleBack: {
        flexDirection: "row",
        alignItems: 'center',
        paddingHorizontal: 20,
      },
    },
  ))
export default ProfilAfter

```

8. src/page/profile_before.tsx

```

import React, { useEffect } from "react";
import {
  SafeAreaView,
  StatusBar,
  StyleSheet,
  View,
  TouchableOpacity,
  Text,
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import LogoSFAD from "../component/image/image";
import MaterialIcons from "react-native-vector-
icons/MaterialIcons";
import Ionicon from "react-native-vector-
icons/Ionicons";
import ButtonInput from "../component/button_input/button_input"
import { useNavigation } from "@react-
navigation/native";
import { namePage } from "../utils/namePage";

const ProfilBefore = () => {

  const navigate = useNavigation()

  //@ts-ignore
  const gotoLogin = () =>
navigate.navigate(namePage.LOGIN_PAGE)
  //@ts-ignore
  const gotoRegister = () =>
navigate.navigate(namePage.REGISTER_PAGE)
  //@ts-ignore
  const gotoInformation = () =>
navigate.navigate(namePage.INFORMATION)

  const goBackPage = () => {
    navigate.goBack()
  }

  return (
    <SafeAreaView style={ [styles.container,
stylesGlobal.backgroundWhite]}>
      <StatusBar

```

```

        animated={true}

        backgroundColor={stylesGlobal.backgroundWhite.backgroundColor}
      />
      <View style={styles.viewHeader}>
        <TouchableOpacity
          style={styles.titleBack} onPress={goBackPage}>
          <MaterialIcons name="arrow-back-
            ios" size={20} color="#2F5664" />
          <Text style={[stylesGlobal.header2,
            stylesGlobal.colorPremier]}>
            Profil
          </Text>
        </TouchableOpacity>
        <TouchableOpacity
          style={styles.titleBack} onPress={gotoInformation}>
          <Icon name="information-circle-
            outline" size={24} color="#2F5664" />
        </TouchableOpacity>
      </View>
      <View style={styles.imageStyle}>
        <LogoSFAD size={130} />
      </View>
      <View style={stylesGlobal.enter20} />
      <ButtonInput
        action={gotoLogin}
        tittle={'Masuk Sebagai Peneliti'}
      />
      <View style={stylesGlobal.enter20} />
      <ButtonInput
        action={gotoRegister}
        tittle={'Daftar Sebagai Peneliti'}
      />

    </SafeAreaView>
  )
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  viewHeader: {
    justifyContent: 'space-between',
    flexDirection: 'row',
    alignItems: 'center'
  },
  imageStyle: {
    justifyContent: "center",
    alignItems: 'center',
    marginVertical: 30,
    marginBottom: 20,
  },
  titleBack: {
    flexDirection: "row",
    alignItems: 'center',

```

```

paddingHorizontal: 20,
    },
  })
export default ProfilBefore

```

9. src/page/ramalan_cuaca.tsx

```

import {
  View,
  Text,
  StyleSheet,
  SafeAreaView,
  ScrollView,
  StatusBar,
  TouchableOpacity,
} from "react-native"
import stylesGlobal from "../utils/global_style"
import LinearGradient from "react-native-linear-gradient";
import { COLOR_GRADIEN2 } from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import Cuaca from "../component/ramalan_cuaca";
import { indicator } from "../utils/indicator";
import { useNavigation } from "@react-navigation/native";

const RamalanCuaca = () => {

  const navigate = useNavigation()

  const goBack = () => navigate.goBack()
  return (
    <LinearGradient colors={COLOR_GRADIEN2}
style={styles.container}>
      <StatusBar
        animated={true}

backgroundcolor={stylesGlobal.backgroundWhite.background
Color}
        />
      <TouchableOpacity style={styles.titleBack}
onPress={goBack}>
        <MaterialIcons name="arrow-back-ios"
size={20} color="#2F5664" />
        <Text style={[stylesGlobal.header2,
stylesGlobal.colorPremier]}>
          Perkiraan Cuaca
        </Text>
      </TouchableOpacity>

      <Cuaca.Header />
      <Cuaca.Body />
    </LinearGradient>
  )
}

```



```

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  titleBack: {
    flexDirection: "row",
    alignItems: 'center',
    paddingHorizontal: 20,
  },
})

export default RamalanCuaca

```

10. src/page/registration_page.tsx

```

import React, { useState, useEffect } from "react";
import {
  SafeAreaView,
  StatusBar,
  StyleSheet,
  View,
  TouchableOpacity,
  Text,
  ScrollView,
} from 'react-native';
import stylesGlobal from "../utils/global_style";
import MaterialIcons from "react-native-vector-icons/MaterialIcons";
import ButtonInput from "../component/button_input/button_input";
import LogoSFAD from "../component/image/image";
import TextInputData from "../component/textInput/textInput";
import { useNavigation } from "@react-navigation/native";
import { AlertStop } from "../component/alert/alert";
import { namePage } from "../utils/namePage";
import { postRegister } from "../utils/axios";
import Loading from "../component/loading/loading";

const RegisterPage = () => {

  const navigate = useNavigation()

  const goBack = () => navigate.goBack()

  // @ts-ignore
  const gotoNavigate = () =>
    navigate.navigate(namePage.LOGIN_PAGE)

  const [name, setName] = useState<string>('')
  const [instansi, setInstansi] =
    useState<string>('')
  const [email, setEmail] = useState<string>('')
  const [password, setPassword] =
    useState<string>('')

```

```

        const [repassword, setRepassword] =
        useState<string>('')
        const [isLoading, setLoading] =
        useState<boolean>(false)

        const input = () => {
            if (name == '' || instansi == '' || password ==
            '' || email == '' || repassword == '') {
                AlertStop({
                    title: 'Perikasa Data',
                    message: 'Lengkapi data diri anda',
                })
            }
            else if (password != repassword) {
                AlertStop({
                    title: 'Perikasa Data',
                    message: 'Konfirmasi kata sandi salah',
                })
            }
            else if (email.search('@') == -1 ||
            email.search(' ') > -1 || email.length <= 10) {
                AlertStop({
                    title: 'Perikasa Data',
                    message: 'Masukan email yang benar',
                })
            }
            else {
                setLoading(true)
                postRegister({
                    email: email,
                    instansi: instansi,
                    name: name,
                    password: password,
                    setData: setLoading,
                    next: gotoNavigate
                })
            }
        }

        useEffect(() => {

        }, [isLoading])

        return (

            <>
            {
                isLoading == true ? <Loading /> :

                <SafeAreaView
                style={[styles.container,
                stylesGlobal.backgroundWhite]}>
                    <StatusBar
                        animated={true}

                backgroundColor={stylesGlobal.backgroundWhite.background
                Color}

                />
                <ScrollView>

```

```

<TouchableOpacity
style={styles.titleBack} onPress={goBack}>
    <MaterialIcons
name="arrow-back-ios" size={20} color="#2F5664" />
    <Text
style={ [stylesGlobal.header2,
stylesGlobal.colorPremier]}>
        Daftar
    </Text>
</TouchableOpacity>
<View
style={styles.imageStyle}>
    <LogoSFAD size={100} />
</View>
<View
style={stylesGlobal.enter10} />
    <TextInputData
        setData={setName}
        data={name}
        placeholder={'Nama
Lengkap'}
        title={'Nama Lengkap'}
        isPasword={false}
    />
    <TextInputData
        setData={setInstansi}
        data={instansi}
        placeholder={'Instansi'}
        title={'Instansi'}
        isPasword={false}
    />
    <TextInputData
        setData={setEmail}
        data={email}
        placeholder={'Email'}
        title={'Email'}
        isPasword={false}
    />
    <TextInputData
        setData={setPassword}
        data={password}
        placeholder={'Kata
Sandi'}
        title={'Kata Sandi'}
        isPasword={true}
    />
    <TextInputData
        setData={setRepassword}
        data={repassword}
        placeholder={'Konfirmasi Kata Sandi'}
        title={'Konfirmasi Kata
Sandi'}
        isPasword={true}
    />
    <ButtonInput action={input}

```

```

        tittle={'DAFTAR'} />
      </ScrollView>
    </SafeAreaView>
  }
</>
)
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingVertical: 30,
  },
  imageStyle: {
    justifyContent: "center",
    alignItems: 'center',
    marginVertical: 30,
    marginBottom: 20,
  },
  titleBack: {
    flexDirection: "row",
    alignItems: 'center',
    paddingHorizontal: 20,
  },
});

export default RegisterPage

```

11. src/page/splash_screen.stx

```

import React, { useEffect, useState } from "react";
import {
  StatusBar,
  StyleSheet,
  View,
  Image,
  Text
} from 'react-native';
import LogoSFAD from "../component/image/image";
import stylesGlobal from "../utils/global_style";
import { useNavigation } from "@react-navigation/native";
import { namePage } from "../utils/namePage";
import AsyncStorage from "@react-native-async-storage/async-storage";
import axios from "axios";
import { useDispatch } from "react-redux";
import { getUsername, getIdUser } from "../redux/action";
import { getDataUserById } from "../utils/api";
import database from '@react-native-firebase/database';

const SplashScreen = () => {

  const navigate = useNavigation()

  const dispatch = useDispatch()

```

```

        // @ts-ignore
        const toNavigate = () =>
        navigate.navigate(namePage.BERANDA_PAGE)

        useEffect(() => {
            setTimeout(() => {
                AsyncStorage.getItem('id')
                    .then(async response => {
                        if (response == null) {
                            database()
                                .ref('/')
                                .on('value', snapshot => {
                                    let temp =
                                    snapshot.val().ready

                                    if (temp == false) {
                                        //MUST DELETE

                                    dispatch(getUserName("Nelayan"))

                                    dispatch(getIdUser("63b787d5d527b4a3ee8746a5"))
                                        toNavigate()
                                    }
                                    else {
                                        toNavigate()
                                    }
                                })
                            }
                        else {
                            await
                            axios.get(getDataUserById({ id: response }))
                                .then(data => {

                                    console.log(data.data.data)

                                    dispatch(getUserName(data.data.data.name))

                                    dispatch(getIdUser(data.data.data.id))
                                        }).finally(() =>
                                    toNavigate())
                                }
                            })
                        }, 3000)
                    }, [])

            return (
                <View style={styles.container}>
                    <StatusBar
                        animated={true}

                    backgroundColor={stylesGlobal.backgroundWhite.backgroundColor}

                    />
                    <View />
                    <LogoSFAD size={200} />
                    <View style={styles.view}>
                        <Text
                            style={[stylesGlobal.colorPremier,
                                stylesGlobal.header3]}> Supported by : </Text>

```

```

                                <View style={stylesGlobal.enter10} />
                                <Image
      source={require('../assets/suportby.png')}      style={{
      height: 39, width: 150 }} />
                                </View>
      </View>
    )
  }

  const styles = StyleSheet.create({
    container: {
      flex: 1,
      justifyContent: "space-between",
      alignItems: 'center',
      backgroundColor: '#FFFF',
      padding: 35
    },
    view: {
      justifyContent: 'center',
      alignItems: 'center',
    }
  })

  export default SplashScreen

```

12. src/component/alert/alert.tsx

```

import { Alert } from "react-native";

export const AlertShow = ({ action, title, message }:
Props) => {
  Alert.alert(
    title,
    message,
    [
      {
        text: "Tidak",
        onPress: () => null,
        style: "cancel"
      },
      { text: "Iya", onPress: () => action() }
    ]
  );
}

export const AlertNext = ({ action, title, message }:
Props) => {
  Alert.alert(
    title,
    message,
    [
      { text: "Selanjutnya", onPress: () =>
action() }
    ]
  );
}

export const AlertStop = ({ title, message }:

```

```

JustProps) => {
  Alert.alert(
    title,
    message,
    [
      { text: "Tutup", onPress: () => null }
    ]
  );
}

interface Props {
  action(): any,
  title: string,
  message: string,
}

interface JustProps {
  title: string,
  message: string,
}

```

13. src/component/beranda/body_beranda.tsx

```

import React from "react";
import {
  StyleSheet,
} from 'react-native';
import LinearGradient from 'react-native-linear-gradient';
import { COLOR_GRADIEN } from "../../utils/global_style";
import Body from "../body_beranda";

const BodyBeranda = () => {
  return (
    <LinearGradient colors={COLOR_GRADIEN}
    style={styles.container}>
      <Body.HeaderBody />
      <Body.StatusBody />
      <Body.ContainDataBody />
    </LinearGradient>
  )
}

const styles = StyleSheet.create({
  container: {
    alignItems: 'center',
    padding: 15,
    justifyContent: 'space-between',
    flexDirection: 'column',
    width: '100%',
    height: '90%',
    borderRadius: 30,
    shadowColor: "#000",
    shadowOffset: {
      width: 0,
      height: 1,
    },
  },

```

```

        shadowOpacity: 0.20,
        shadowRadius: 1.41,
        elevation: 2,
      },
    ))

export default BodyBeranda

```

14. src/component/beranda/header_beranda.tsx

```

import React from "react";
import {
  Text,
  StyleSheet,
  View,
  TouchableOpacity
} from 'react-native';
import stylesGlobal from "../../utils/global_style";
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';
import { useNavigation } from "@react-navigation/native";
import { namePage } from "../../utils/namePage";
import { useSelector } from "react-redux";

const HeaderBeranda = () => {

  const navigate = useNavigation()

  const { userName, idUser } = useSelector(
    //@ts-ignore
    state => state.userReducer
  )

  const nextPage = () => {
    if (idUser == '') {
      //@ts-ignore
      navigate.navigate(namePage.PROFIL_BEFORE)
    }
    else {
      //@ts-ignore
      navigate.navigate(namePage.PROFIL_AFTER)
    }
  }

  return (
    <View style={styles.container}>
      <View>
        <Text style={[stylesGlobal.colorWhite, stylesGlobal.header1]}> Hallo, </Text>
        <Text style={[stylesGlobal.colorWhite, stylesGlobal.header2]}> {userName} </Text>
      </View>
      <TouchableOpacity onPress={nextPage}>
        <MaterialCommunityIcons name="account"
size={30} color="#fff" />
      </TouchableOpacity>
    </View>
  )
}

```



```

    )
  }

  const styles = StyleSheet.create({
    container: {
      flexDirection: 'row',
      justifyContent: "space-between",
      alignItems: 'center',
      marginBottom: 30,
    },
  }),
  })

export default HeaderBeranda

```

15. src/component/body_beranda/contain_data_body.tsx

```

import React, { useEffect, useState } from "react";
import {
  View,
  StyleSheet,
  TouchableWithoutFeedback,
} from 'react-native';
import Card from "../card_beranda/card";
import database from '@react-native-firebase/database';

const ContainDataBody = () => {
  const [arahAhngin, setArahAngin] =
    useState<string>('')
  const [kecepatanAngin, setKecepatanAngin] =
    useState<number>(0)
  const [KecepatanGelombang, setKecepatanGelombang] =
    useState<number>(0)
  const [tinggiGelombang, setTinggiGelombang] =
    useState<number>(0)
  const [suhuLaut, setSuhuLaut] = useState<number>(0)

  database()
    .ref('/')
    .on('value', snapshot => {
      setArahAngin(snapshot.val().angin.arah)

      setKecepatanAngin(snapshot.val().angin.kecepatan_mps)

      setKecepatanGelombang(snapshot.val().laut.kecepatan_mps)

      setTinggiGelombang(snapshot.val().laut.ketinggian_m)

      setSuhuLaut(snapshot.val().laut.suhu_ms5611_c)
    });

  return (
    <View style={styles.container}>
      <Card.LongCard
        name={'Suhu Laut'}
        value={suhuLaut}

```

```

        type={'°'}
        link={'suhu-lingkungkans'}
        key={'suhu-lingkungkans'}
      />
    <View style={styles.wrap}>
      <Card.GridCard
        name={'Arah Angin'}
        value={arahAhngin}
        type={' '}
        link={'arah-angins'}
        key={'arah-angins'}
      />
      <Card.GridCard

        name={'Kecepatan Angin'}
        value={kecepatanAngin}
        type={'m/s'}
        link={'kecepatan-angins'}
        key={'kecepatan-angins'}
      />
      <Card.GridCard

        name={'Tinggi Gelombang'}
        value={tinggiGelombang}
        type={'M'}
        link={'tinggi-gelombangs'}
        key={'tinggi-gelombangs'}
      />
      <Card.GridCard

        name={'Kecepatan Arus'}
        value={KecepatanGelombang}
        type={'m/s'}
        link={'kuat-aruss'}
        key={'kuat-aruss'}
      />
    </View>
  </View>
)
}

const styles = StyleSheet.create({
  contain: {
    width: '100%'
  },
  wrap: {
    width: '100%',
    flexDirection: 'row',
    justifyContent: 'space-between',
    flexWrap: 'wrap'
  },
})

export default ContainDataBody

```

```

import React from "react";
import {
  View,
  StyleSheet,
  Text,
  TouchableOpacity
} from 'react-native';
import stylesGlobal from "../../utils/global_style";
import { TimeBeranda } from "../../utils/moment.";
import { useNavigation } from "@react-navigation/native";
import { namePage } from "../../utils/namePage";

const HeaderBody = () => {

  const navigate = useNavigation()

  //@ts-ignore
  const gtoPrediksi = () =>
  navigate.navigate(namePage.RAMALAN_CUACA)

  return (

    <View style={styles.container}>
      <Text style={[stylesGlobal.colorWhite,
stylesGlobal.subtitle]}>
        {TimeBeranda()}
      </Text>
      <View style={stylesGlobal.enter10} />
      <Text style={[stylesGlobal.colorWhite,
stylesGlobal.header1]}>
        Teluk Kiluan
      </Text>
      <View style={stylesGlobal.enter10} />
      <TouchableOpacity style={[styles.box,
stylesGlobal.backgroundSekunder]} onPress={gtoPrediksi}>
        <Text style={[stylesGlobal.colorWhite,
stylesGlobal.header3]}>Prediksi Cuaca</Text>
      </TouchableOpacity>
    </View>

  )

}

const styles = StyleSheet.create({
  container: {
    justifyContent: 'center',
    alignItems: "center",
    width: '100%'
  },
  box: {
    width: '55%',
    height: 30,

    borderRadius: 30,
    justifyContent: 'center',
    alignItems: 'center',
    // shadowColor: "#000",
    shadowOffset: {

```

```

        width: 1,
        height: 3,
      },
      shadowOpacity: 1,
      shadowRadius: 5,
      elevation: 5,
    }
  })

```

```
export default HeaderBody
```

17. src/component/body_beranda/status.tsx

```

import React, { useState } from "react";
import {
  View,
  Text,
} from 'react-native';
import stylesGlobal from "../../utils/global_style";
import database from '@react-native-firebase/database';
import { indicator } from "../../utils/indicator";
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';

const StatusBody = () => {
  const [circumstances, setcircumstances] =
    useState<string>('')

  database()
    .ref('/')
    .on('value', snapshot => {

    setcircumstances(snapshot.val().laut.keamanan_lvl)
    });

  return (
    <View style={{ justifyContent: 'center',
    alignItems: 'center' }}>
      <Text style={[stylesGlobal.colorWhite,
        stylesGlobal.header1]}>
        {circumstances}
      </Text>
      <View style={stylesGlobal.enter10} />
    </View>
  )
}

export default StatusBody

```

18. src/component/ramalan_cuaca/body.tsx

```

import React, { useEffect, useState } from "react"
import {

```



```

<Text                                style={ [stylesGlobal.header2,
stylesGlobal.colorPremier]}>{response.status}</Text>

</View>

<View style={styles.box50}>

<Image source={{

uri: response.gambar,

}} style={styles.img} />

<Text                                style={ [stylesGlobal.header1,
stylesGlobal.colorPremier]}>{response.suhu}</Text>

</View>

</View>

                                                                    </View>
                                                                ))
                                                                    }
                                                                </View>
                                                                ))
                                                                    }
                                                                </ScrollView>
</View>
    }
</>
)
}

const styles = StyleSheet.create({
  container: {
    width: '100%',
    height: '78%',
    paddingHorizontal: 20
  },
  card: {
    width: '100%',
    height: 100,
    backgroundColor: '#fff',
    borderRadius: 10,
    borderWidth: 1,
    borderColor: "#6399B0",
    alignItems: 'center',
    padding: 5,
    marginBottom: 10
  },
  boxInCard: {
    width: '100%',
    height: 60,
    marginTop: 5,
    flexDirection: "row",
    justifyContent: 'space-between',
    alignItems: 'center'
  },
  box50: {
    width: '50%',

```

```

        height: '100%',
        justifyContent: 'center',
        alignItems: "center",
        flexDirection: 'row'
      },
      img: {
        height: 50,
        width: 50,
        marginRight: 5,
      }
    })
  export default Body

```

19. src/component/ramalan_cuaca/header.tsx

```

import React, { useState } from "react";
import {
  View,
  Text,
  StyleSheet,
} from "react-native"
import stylesGlobal from "../../utils/global_style"
import database from '@react-native-firebase/database';
import { indicator } from "../../utils/indicator";
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';

const Header = () => {

  const [kecepatanAngin, setKecepatanAngin] =
    useState<number>(0)
  const [tinggiGelombang, setTinggiGelombang] =
    useState<number>(0)
  const [suhuLaut, setSuhuLaut] = useState<number>(0)

  database()
    .ref('/')
    .on('value', snapshot => {

      setKecepatanAngin(snapshot.val().angin.kecepatan_mps)

      setTinggiGelombang(snapshot.val().laut.ketinggian_m)

      setSuhuLaut(snapshot.val().laut.suhu_ms5611_c)
    });

  return (
    <View style={styles.container}>
      <View style={styles.boxContain}>
        <View style={styles.box50}>
          <Text
            style={[stylesGlobal.colorPremier,
              stylesGlobal.header1, styles.textTittle]}>
            {indicator({
              angin: kecepatanAngin,
              gelombang: tinggiGelombang
            })}

```

```

        </Text>
      </View>

      <View style={{ height: "55%", width:
2,      marginHorizontal:      3      }},
stylesGlobal.backgroundSekunder}} />

      <View style={styles.box50}>
        <View style={styles.suhuCuaca}>
          <MaterialCommunityIcons
name="thermometer" size={42} color='#2F5664' />
          <Text
style={([stylesGlobal.colorPremier, styles.testSuhu])>
            {suhuLaut + '°' + 'C'}
          </Text>
        </View>
      </View>
    </View>
    <Text style={([stylesGlobal.subtitle,
stylesGlobal.colorPremier])>
      * Berdasarkan data BMKG
    </Text>
  </View>
)
}

const styles = StyleSheet.create({
  container: {
    width: '100%',
    height: 150,
    padding: 20,
    justifyContent: 'center',
    alignItems: 'center',
  },
  boxContain: {
    width: "75%",
    height: "100%",
    justifyContent: 'space-between',
    alignItems: 'center',
    flexDirection: "row",
  },
  box50: {
    width: "49.5%",
    justifyContent: 'center',
    alignItems: 'center',
  },
  textTittle: {
    textAlign: 'center'
  },
  testSuhu: {
    fontSize: 32,
    fontFamily: 'Ubuntu-Bold'
  },
  suhuCuaca: {
    flexDirection: 'row',
  },
})

export default Header

```


20. src/redux/action.tsx

```

export const GET_MONTH_VALUE = 'GET_MONTH_VALUE'
export const GET_YEAR_VALUE = 'GET_YEAR_VALUE'
export const GET_USER_NAME = 'GET_USER_NAME'
export const GET_ID_USER = 'GET_ID_USER'

export const setMonthValue = (data : number) => ({
  type : GET_MONTH_VALUE,
  data : data
})

export const setYearValue = (data : string) => ({
  type: GET_YEAR_VALUE,
  data : data
})

export const getUsername = (data : string) => ({
  type : GET_USER_NAME,
  data : data
})

export const getIdUser = (data : string) => ({
  type : GET_ID_USER,
  data : data
})

```

21. src/redux/reducer.tsx

```

import {
  GET_MONTH_VALUE,
  GET_USER_NAME,
  GET_YEAR_VALUE,
  GET_ID_USER
} from "../action";

const initialState = {
  month : 1,
  year : '2023',
  userName : 'Pemantau',
  idUser : ''
}

function userReducer(state = initialState, action :any)
{
  switch (action.type) {
    case GET_MONTH_VALUE :
      return { ...state, month : action.data};
    case GET_YEAR_VALUE :
      return { ...state, year : action.data};
    case GET_USER_NAME:
      return {...state, userName : action.data};
    case GET_ID_USER:
      return {...state, idUser : action.data};
    default :
      return state
  }
}

```

```

    }
  }
  export default userReducer

```

22. src/redux/store.tsx

```

import { combineReducers } from "redux";
import userReducer from "../reducer";
import { configureStore } from "@reduxjs/toolkit";
import logger from "redux-logger";

const rootReducer = combineReducers({ userReducer });

export const Store = configureStore({
  reducer: rootReducer,
  middleware: (getDefaultMiddleware) =>
    getDefaultMiddleware().concat(logger)
});

```

23. src/utlis/axios.ts

```

import axios from "axios";
import { ReactSetter } from "../interface";
import { registerApi, loginApi, getDataUserById } from
  '../utlis/api'
import { AlertStop, AlertNext } from
  "../component/alert/alert";
import AsyncStorage from "@react-native-async-
storage/async-storage";
// import {getUserName, getIdUser} from
  '../redux/action'

export const getDataGrafik = async ({data, setData,
setLabel, setAfter}: Props) => {
  await axios.get(data)
  .then(respons => {
    let length = respons.data.data.length
    for (let i = 0; i < length; i++){
      setLabel((oldArray: any) => [ ...oldArray,
(respons.data.data[i].index + 1 + ' ')])
      setData ((oldArray : any) => [ ...oldArray,
(respons.data.data[i].value + 0)])
    }
    setAfter(true)
  })
}

export const getDataAverage = async ({data, setData,
setLabel, setAfter}: Props) => {
  await axios.get(data)
  .then(response => {
    setData(response.data.data)
    setAfter(false)
  })
}

```

```

    })
  }

  export const postRegister = async ({email, instansi, name, password, setData, next} : Register) => {
    const data = registerApi()
    await axios.post(data, {
      name : name,
      email: email,
      instansi : instansi,
      password : password
    })
    .then(response => {
      console.log(response.data.data == 'email-already')
      if (response.data.data == 'email-already'){
        setData(false)
        AlertStop({
          title: 'Email Sudah Terpakai',
          message: 'Periksa lagi email anda',
        })
      }
      else {
        setData(false)
        AlertNext({
          title : "Registrasi Berhasil",
          message : "Berhasil melakukan registrasi",
          action : next
        })
      }
    })
  })
}

export const postLogin = async ({email, password, setData, next} : Login) => {
  const linkData = loginApi()

  await axios.post(linkData, {
    email : email,
    password : password
  })
  .then( async response => {
    console.log(response.data.data)
    await AsyncStorage.setItem('id', response.data.data.id)
    .then(() => {
      console.log(response.data.data.id)
      setData(false)
      AlertNext({
        title : "Sukses",
        message : "Anda sudah masuk sebagai peneliti",
        action : next
      })
    })
  })
}

```

```

        })
      }
    )

    }).catch(err => {
      setData(false)
      AlertStop({
        title: 'Email Dan Password Salah',
        message: 'Periksa kembali email dan
password anda',
      })
    })
  }

export const userData = async
({data, setEmail, setInstansi, setLoading, setName}: User)
=> {
  const linkData = getUserById({id : data})
  await axios.get(linkData)
  .then(response => {
    console.log(response.data.data)
    setEmail(response.data.data.email)
    setInstansi(response.data.data.instansi)
    setName(response.data.data.name)
    setLoading(false)
  })
}

interface User{
  data : string
  setName: ReactSetter<string>,
  setInstansi: ReactSetter<string>,
  setEmail: ReactSetter<string>,
  setLoading: ReactSetter<boolean>,
}
interface Props {
  data : string,
  setData: ReactSetter<any>,
  setLabel : ReactSetter<any>,
  setAfter : ReactSetter<boolean>
}
interface Register {
  name : string,
  instansi : string,
  email : string,
  password : string,
  setData: ReactSetter<boolean>,
  next() : any
}

interface Login {
  email : string,
  password : string,
  setData: ReactSetter<boolean>,
  next() : any
}

```

```

import moment from "moment"
export const TimeBeranda = () => {
  let day = moment().format('dddd');
  if (day == 'Monday') day = `Senin`
  if (day == 'Tuesday') day = `Selasa`
  if (day == 'Wednesday') day = `Rabu`
  if (day == 'Thursday') day = `Kamis`
  if (day == 'Friday') day = `Jum'at`
  if (day == 'Saturday') day = `Sabtu`
  if (day == 'Sunday') day = `Minggu`
  let date = moment().format('D')
  let month = moment().format("MMMM");
  if (month == 'January') month = 'Januari'
  if (month == 'February') month = 'Februari'
  if (month == 'March') month = 'Maret'
  if (month == 'April') month = 'April'
  if (month == 'May') month = 'Mei'
  if (month == 'June') month = 'Juni'
  if (month == 'July') month = 'Juli'
  if (month == 'August') month = 'Agustus'
  if (month == 'September') month = 'September'
  if (month == 'October') month = 'Oktober'
  if (month == 'November') month = 'Novembar'
  if (month == 'December') month = 'Desember'
  let year = moment().format('YYYY');
  return day + ', ' + date + ' ' + month + ' ' +
year
}

```

25. src/utills/perkiraan_cuaca.tsx

```

import axios from "axios"
import { cuacaApi } from "../api"
import { ReactSetter } from "../interface"
import { convertCuacaImage } from "../convert_image"

const dataPerkiraanCuaca = ({setData, isLoading}:Props)
=> {
  axios.get(cuacaApi())
    .then(response => {

      const perkiraanCuaca = [
        {
          tiitle : "Perkiraan Cuaca Hari Ini",
          data : [
            {
              waktu : 'Pagi',
              status
            },
            {
              waktu : 'Siang',
              status
            }
          ]
        }
      ]

      response.data.data.params[6].times[1].name,
      gambar
      convertCuacaImage({data
      :response.data.data.params[6].times[1].code }),
      suhu
      response.data.data.params[5].times[1].celcius
    }
  )
}

```



```

                                status :
response.data.data.params[6].times[9].name,
                                gambar :
convertCuacaImage({data
:response.data.data.params[6].times[9].code })),
                                suhu :
response.data.data.params[5].times[9].celcius
                                },
                                {
                                    waktu : "Siang",
                                    status :
response.data.data.params[6].times[10].name,
                                    gambar :
convertCuacaImage({data
:response.data.data.params[6].times[10].code })),
                                    suhu :
response.data.data.params[5].times[10].celcius
                                    },
                                    {
                                        waktu : "Malam",
                                        status :
response.data.data.params[6].times[11].name,
                                        gambar :
convertCuacaImage({data
:response.data.data.params[6].times[11].code })),
                                        suhu :
response.data.data.params[5].times[11].celcius
                                        }
                                    ]
                                }
                            ]
                        setData(perkiraanCuaca)
                    })
                }

export default dataPerkiraanCuaca
interface Props {
    setData: ReactSetter<any>,
    isLoading: ReactSetter<boolean>,
}

```

26. src/utlis/backHandler.ts

```

import {
    BackHandler,
    Alert,
} from 'react-native';
import { useNavigation } from '@react-
navigation/native';
const navigation = useNavigation()
export const isExit = () => {
    Alert.alert("Menutup Aplikasi", "Anda yakin ingin
menutup aplikasi ?", [
        {
            text: "Tidak",
            onPress: () => null,
            style: "cancel"
        },
    ],

```

```
        {
            text: "Ya",
            onPress: () => BackHandler.exitApp()
        }
    });
    return true;
}

export const backPressHandler = () => {
    navigation.goBack()
    return true
}

export const backCheckExit = () =>
BackHandler.addEventListener("hardwareBackPress", isExit)
```


Lampiran II Kode Pada Back-End

Adapun Lampiran Untuk Menampilkan back-end sebagai berikut :

1. src/index.ts

```
import { config } from "dotenv";
import { expand } from "dotenv-expand";
import express from "express";
import root from "../utils/root";

const envConfig = config();
expand(envConfig);

const app = express();
const PORT = process.env.PORT || 3000;

root(app);

app.listen(PORT, () => console.log(`Server started at port ${PORT}`));
```

2. src/handler/average.ts

```
import _ from "lodash";
import moment from "moment";
import type { Request } from "express";
import { seqPromise } from "../utils/promise";
import { getAggregate } from "../helper";

export const geDayAverageData = async (query: Request["query"]) => {
  const { month, year, type } = query;

  const aggregate = getAggregate(type as string);

  if (_.isNil(aggregate)) return;

  const yearMonth = `${year}-${month}`;

  const totalDays = moment(`${yearMonth}-1`).daysInMonth();

  const results = await seqPromise(
    _.map(_.range(totalDays), async (value) => {
      return {
        date: `${yearMonth}-${value + 1}`,
        value:
          (
            await aggregate(
              {
                // @ts-ignore
                createdAt: {
                  gte: moment(`${yearMonth}-${value + 1}`).toDate(),
                  lte: moment(`${yearMonth}-${value + 1}`).add(23, "hour")
                }
              )
            )
          )
      };
    })
  );
}
```

```

        .add(59, "minutes")
        .add(59, "second")
        .toDate(),
    },
},
{
    _avg: {
        value: true,
    },
}
)
//@ts-ignore
)?._avg?.value ?? 0,
};
})
);

return results;
};

```

3. src/handler/grafik.ts

```

import _ from "lodash";
import moment from "moment";
import type { Request } from "express";
import { createDateRange } from "../utils/date";
import { seqPromise } from "../utils/promise";
import { getAggregate } from "../helper";

export const getGrafikData = async (query:
Request["query"]) => {
    const { month, year, type } = query;

    const aggregate = getAggregate(type as string);

    if (_.isNil(aggregate)) return;
    if (_.isNil(aggregate)) return;

    const { dates, toDates } = createDateRange({
        month: Number(month),
        year: Number(year),
    });

    const results = await seqPromise(
        _.map(dates, async (date, index) => {
            console.log(moment(date).toDate());

            return {
                index,
                date: moment(date).format("YYYY-MM-D"),
                value:
                    (
                        await aggregate(
                            {
                                //@ts-ignore

```

```

        createdAt: {
          gt: date,
          lte: toDates[index],
        },
      },
      {
        _avg: {
          value: true,
        },
      }
    )
  )
  //@ts-ignore
  ?._avg?.value ?? 0,
};
}))
);

return results;
};

```

4. src/handler/helper.ts

```

import _ from "lodash";
import repository from "../repository";
import { QUERY_TYPE } from "../utils/constant";

export const getAggregate = (type: string) => {
  let aggregate:
    | typeof repository.tinggiGelombang.aggregate
    | typeof repository.kecepatanAngin.aggregate
    | typeof repository.suhuLingkungan.aggregate
    | typeof repository.kuatArus.aggregate
    | undefined;

  switch (_.toLower(type)) {
    case QUERY_TYPE.TINGGI_GELOMBANG:
      aggregate = repository.tinggiGelombang.aggregate;
      break;
    case QUERY_TYPE.KECEPATAN_ANGIN:
      aggregate = repository.kecepatanAngin.aggregate;
      break;
    case QUERY_TYPE.SUHU_LINGKUNGAN:
      aggregate = repository.suhuLingkungan.aggregate;
      break;
    case QUERY_TYPE.KUAT_ARUS:
      aggregate = repository.kuatArus.aggregate;
      break;
    default:
      break;
  }

  return aggregate;
};

```

5. src/middleware/kecepatanAngin.ts

```

import { asyncMw } from "express-asyncmw";
import _ from "lodash";
import repository from "../repository";
import { seqPromise } from "../utils/promise";
import { orderBy } from "../utils/query";
import { getGrafikData } from "../handler/grafik";
import { geDayAverageData } from "../handler/average";
import { QUERY_TYPE } from "../utils/constant";

export const createKecepatanAnginMw = asyncMw(async (
  req, res, next ) => {
  req.kecepatanAngin = await
  repository.kecepatanAngin.create(req.body);

  return next();
});

export const getKecepatanAnginMw = asyncMw(async (req,
  res, next) => {
  const kecepatanAngin = await
  repository.kecepatanAngin.findOne({
    id: req.params.id,
  });

  if (!kecepatanAngin)
    return res.status(404).json({
      status: 404,
      message: "Kecepatan Angin Not Found",
    });

  req.kecepatanAngin = kecepatanAngin;

  return next();
});

export const getKecepatanAnginsMw = asyncMw(async (req,
  res, next) => {
  req.kecepatanAngins = await
  repository.kecepatanAngin.findAll(
    {},
    req.filterQueryParams,
    {},
    {
      orderBy: {
        //@ts-ignore
        createdAt: orderBy(req.query),
      },
    },
  );

  return next();
});

export const returnKecepatanAnginMw = asyncMw(async (
  req, res ) => {
  return res.status(200).json({
    status: 200,

```

```

        data: await
        repository.kecepatanAngin.modelToResource(req.kecepatan
Angin),
    });
});

export const returnKecepatanAnginsMw = asyncMw(async
(req, res) => {
    return res.status(200).json({
        status: 200,
        data: await seqPromise(
            _.map(_.get(req.kecepatanAngins, "rows", []),
(kecepatanAngin) =>

        repository.kecepatanAngin.modelToResource(kecepatanAngi
n)
        )
    ),
    total: _.get(req.kecepatanAngins, "count", 0),
    });
});

export const getGrafikKecepatanAnginsMw = asyncMw(async
(req, res, next) => {
    req.grafikKecepatanAngins = await getGrafikData({
        ...req.query,
        type: QUERY_TYPE.KECEPATAN_ANGIN,
    });

    return next();
});

export const returnGrafikKecepatanAnginsMw =
asyncMw(async (req, res) => {
    return res.status(200).json({
        status: 200,
        data: req.grafikKecepatanAngins,
    });
});

export const getDayAverageKecepatanAnginMw =
asyncMw(async (req, res, next) => {
    req.averageKecepatanAngins = await geDayAverageData({
        ...req.query,
        type: QUERY_TYPE.KECEPATAN_ANGIN,
    });

    return next();
});

export const returnDayAverageKecepatanAnginsMw =
asyncMw(async (req, res) => {
    return res.status(200).json({
        status: 200,
        data: req.averageKecepatanAngins,
    });
});

```

6. src/middleware/kuatArus.ts

```

import { asyncMw } from "express-asyncmw";
import _ from "lodash";
import { geDayAverageData } from "../handler/average";
import { getGrafikData } from "../handler/grafik";
import repository from "../repository";
import { QUERY_TYPE } from "../utils/constant";
import { seqPromise } from "../utils/promise";
import { orderBy } from "../utils/query";

export const createKuatArusMw = asyncMw(async (req,
res, next) => {
  req.kuatArus = await
  repository.kuatArus.create(req.body);

  return next();
});

export const getKuatArusMw = asyncMw(async (req, res,
next) => {
  const kuatArus = await repository.kuatArus.findOne({
    id: req.params.id,
  });
  if (!kuatArus)
    return res.status(404).json({
      status: 404,
      message: "Kecepatan Angin Not Found",
    });

  req.kuatArus = kuatArus;

  return next();
});

export const getKuatArussMw = asyncMw(async (req, res,
next) => {
  req.kuatAruss = await repository.kuatArus.findAll(
    {},
    req.filterQueryParams,
    {},
    {
      orderBy: {
        //@ts-ignore
        createdAt: orderBy(req.query),
      },
    },
  );

  return next();
});

export const returnKuatArusMw = asyncMw(async (req,
res) => {
  return res.status(200).json({
    status: 200,
    data: await
    repository.kuatArus.modelToResource(req.kuatArus),
  });
});

```

```

});

export const returnKuatArussMw = asyncMw(async (req,
res) => {
  return res.status(200).json({
    status: 200,
    data: await seqPromise(
      _.map(_.get(req.kuatAruss, "rows", []),
(kuatArus) =>

repository.kecepatanAngin.modelToResource(kuatArus)
    )
    ),
    total: _.get(req.kuatArus, "count", 0),
  });
});

export const getGrafikKuatArusMw = asyncMw(async (req,
res, next) => {
  req.grafikKuatArus = await getGrafikData({
    ...req.query,
    type: QUERY_TYPE.KUAT_ARUS,
  });

  return next();
});

export const returnGrafikKuatArusMw = asyncMw(async
(req, res) => {
  return res.status(200).json({
    status: 200,
    data: req.grafikKuatArus,
  });
});

export const getDayAverageKuatArusMw = asyncMw(async
(req, res, next) => {
  req.averageKuatArus = await geDayAverageData({
    ...req.query,
    type: QUERY_TYPE.KUAT_ARUS,
  });

  return next();
});

export const returnDayAverageKuatArusMw = asyncMw(async
(req, res) => {
  return res.status(200).json({
    status: 200,
    data: req.averageKuatArus,
  });
});
});

```

7. src/middleware/postAllData.ts

```

import { asyncMw } from 'express-asyncmw';
import _ from 'lodash';
import repository from '../repository';

```

```

export const createPostAllDataMw = asyncMw(async (req,
res, next) => {
  req.kecepatanAngin = await
repository.kecepatanAngin.create({
  //@ts-ignore
  value: req.body.kecepatanAngin
})
  req.kuatArus = await repository.kuatArus.create({
  //@ts-ignore
  value: req.body.kuatArus
})
  req.suhuLingkungan = await
repository.suhuLingkungan.create({
  //@ts-ignore
  value: req.body.suhuLingkungan
})
  req.tinggiGelombang = await
repository.tinggiGelombang.create({
  //@ts-ignore
  value: req.body.tinggiGelombang
})

  return next()
})

export const returnPostAllDataMw = asyncMw(async (req,
res) => {
  return res.status(200).json({
    status: 200,
    data: {
      kecepatanAngin: await
repository.kecepatanAngin.modelToResource(req.kecepatan
Angin),
      kuatArus: await
repository.kuatArus.modelToResource(req.kuatArus),
      suhuLingkungan: await
repository.suhuLingkungan.modelToResource(req.suhuLingk
ungan),
      tinggiGelombang: await
repository.tinggiGelombang.modelToResource(req.tinggiGe
lombang),
    }
  })
})

```

8. src/middleware/queryParses.ts

```

import PrismaFQP from '@krsbx/prisma-fqp';
import { asyncMw } from 'express-asyncmw';

export const queryParserMw = asyncMw(async (req, res,
next) => {
  req.filterQueryParams = req.query.filters ?
PrismaFQP(req.query.filters as string) : {};
  delete req.query.filters;
  return next();
});

```


9. src/middleware/suhuLingkungan.ts

```

import { asyncMw } from "express-asyncmw";
import _ from "lodash";
import { geDayAverageData } from
  "../handler/average";
import { getGrafikData } from "../handler/grafik";
import repository from "../repository";
import { QUERY_TYPE } from "../utils/constant";
import { seqPromise } from "../utils/promise";
import { orderBy } from "../utils/query";

export const createSuhuLingkunganMw =
  asyncMw(async (req, res, next) => {
    req.suhuLingkungan = await
      repository.suhuLingkungan.create(req.body);

    return next();
  });

export const getSuhuLingkunganMw = asyncMw(async
  (req, res, next) => {
    const suhuLingkungan = await
      repository.suhuLingkungan.findOne({
        id: req.params.id,
      });

    if (!suhuLingkungan)
      return res.status(404).json({
        status: 404,
        message: "Kecepatan Angin Not Found",
      });

    req.suhuLingkungan = suhuLingkungan;

    return next();
  });

export const getSuhuLingkungansMw = asyncMw(async
  (req, res, next) => {
    req.suhuLingkungans = await
      repository.suhuLingkungan.findAll(
        {},
        req.filterQueryParams,
        {},
        {
          orderBy: {
            //@ts-ignore
            createdAt: orderBy(req.query),
          },
        },
      );
  });

```

```

        return next();
    });

export const returnSuhuLingkunganMw =
  asyncMw(async (req, res) => {
    return res.status(200).json({
      status: 200,
      data: await
        repository.suhuLingkungan.modelToResource(req.suhu
          Lingkungan),
    });
  });

export const returnSuhuLingkungsMw =
  asyncMw(async (req, res) => {
    return res.status(200).json({
      status: 200,
      data: await seqPromise(
        _.map(_.get(req.suhuLingkungs, "rows",
          []), (suhuLingkungs) =>

        repository.kecepatanAngin.modelToResource(suhuLing
          kungs)
        ),
      total: _.get(req.suhuLingkungs, "count", 0),
    });
  });

export const getGrafikSuhuLingkunganMw =
  asyncMw(async (req, res, next) => {
    req.grafikSuhuLingkungan = await getGrafikData({
      ...req.query,
      type: QUERY_TYPE.SUHU_LINGKUNGAN,
    });

    return next();
  });

export const returnGrafikSuhuLingkunganMw =
  asyncMw(async (req, res) => {
    return res.status(200).json({
      status: 200,
      data: req.grafikSuhuLingkungan,
    });
  });

export const getDayAverageSuhuLingkunganMw =
  asyncMw(async (req, res, next) => {
    req.averageSuhuLingkungan = await
      geDayAverageData({
        ...req.query,

```

```

        type: QUERY_TYPE.SUHU_LINGKUNGAN,
    });

    return next();
});

export const returnDayAverageSuhuLingkunganMw =
asyncMw(async (req, res) => {
    return res.status(200).json({
        status: 200,
        data: req.averageSuhuLingkungan,
    });
});

```

10. src/middleware/tinggiGelombang.ts

```

import { asyncMw } from "express-asyncmw";
import _ from "lodash";
import { geDayAverageData } from "../handler/average";
import { getGrafikData } from "../handler/grafik";
import repository from "../repository";
import { QUERY_TYPE } from "../utils/constant";
import { seqPromise } from "../utils/promise";
import { orderBy } from "../utils/query";

export const createTinggiGelombangMw = asyncMw(async
(req, res, next) => {
    req.tinggiGelombang = await
repository.tinggiGelombang.create(req.body);

    return next();
});

export const getTinggiGelombangMw = asyncMw(async (req,
res, next) => {
    const tinggiGelombang = await
repository.tinggiGelombang.findOne({
        id: req.params.id,
    });

    if (!tinggiGelombang)
        return res.status(404).json({
            status: 404,
            message: "Tinggi Gelombang Not Found",
        });

    req.tinggiGelombang = tinggiGelombang;

    return next();
});

export const getTinggiGelombangMw = asyncMw(async
(req, res, next) => {
    req.tinggiGelombang = await
repository.tinggiGelombang.findAll(
    {},
    req.filterQueryParams,

```

```

    {} ,
    {
      orderBy: {
        //@ts-ignore
        createdAt: orderBy(req.query),
      },
    }
  );

  return next();
});

export const returnTinggiGelombangMw = asyncMw(async
(req, res) => {
  return res.status(200).json({
    status: 200,
    data: await
repository.tinggiGelombang.modelToResource(req.tinggiGe
lombang),
  });
});

export const returnTinggiGelombangMw = asyncMw(async
(req, res) => {
  return res.status(200).json({
    status: 200,
    data: await seqPromise(
      _.map(_.get(req.tinggiGelombang, "rows", []),
(tinggiGelombang) =>

repository.kecepatanAngin.modelToResource(tinggiGelomba
ngs)
    )
  ),
    total: _.get(req.tinggiGelombang, "count", 0),
  });
});

export const getGrafikTinggiGelombangMw = asyncMw(async
(req, res, next) => {
  req.grafikTinggiGelombang = await getGrafikData({
    ...req.query,
    type: QUERY_TYPE.TINGGI_GELOMBANG,
  });

  return next();
});

export const returnGrafikTinggiGelombangMw =
asyncMw(async (req, res) => {
  return res.status(200).json({
    status: 200,
    data: req.grafikTinggiGelombang,
  });
});

export const getDayAverageTinggiGelombangMw = asyncMw(
  async (req, res, next) => {
    req.averageTinggiGelombang = await

```

```

    geDayAverageData({
      ...req.query,
      type: QUERY_TYPE.TINGGI_GELOMBANG,
    });

    return next();
  }
);

export const returnDayAverageTinggiGelombangMw =
  asyncMw(async (req, res) => {
    return res.status(200).json({
      status: 200,
      data: req.averageTinggiGelombang,
    });
  });

```

11. src/middleware/user.ts

```

import { asyncMw } from "express-asyncmw";
import _ from "lodash";
import repository from "../repository";
import bcrypt from "bcrypt";

export const createPostDataUserMw =
  asyncMw(async (req, res, next) => {

    const dataEmail = await
    repository.user.findOne({email : req.body.email})

    if (dataEmail){
      return res.status(200).json({
        status : 200,
        data : "email-already"
      })
    }
    const saltRounds = 10;

    const hash = await bcrypt.hash(req.body.password,
    saltRounds)

    req.user = await repository.user.create({
      name : req.body.name,
      email : req.body.email,
      instansi : req.body.instansi,
      password : hash
    })

    return next()
  })

export const returnPostDataUserMw =
  asyncMw(async (req, res, next) => {
    return res.status(200).json({
      status : 200,
      data : await
      repository.user.modelToResource(req.user)
    })
  })

```

```

    })
  })

  export const checkDataUserMw = asyncMw(async(req, res,
next) => {

    const dataUser = await repository.user.findOne({
      email : req.body.email,
    })

    if (!dataUser) {
      return res.status(404).json({
        status : 404,
        data : 'no-have-data'
      })
    } else{

      const checkpassword = await
bcrypt.compare(req.body.password, dataUser.password)

      if (!checkpassword){
        return res.status(404).json({
          status : 404,
          data : 'wrong-password'
        })
      }

      req.checker = dataUser
      return next()
    }

  })

  export const returnCheckDataUserMw = asyncMw(async(req,
res, next) =>{
    return res.status(200).json({
      status : 200,
      data : await req.checker
    })
  })

  export const getDataUserMw = asyncMw(async(req, res,
next) =>{

    if (req.params.id.length != 24){
      return res.status(404).json({
        status : 404,
        data : "no-have-user"
      })
    }

    const dataUser = await repository.user.findOne({
      id : req.params.id
    })

    console.log('test',dataUser)
  })

```

```

        if (!dataUser) {
            return res.status(404).json({
                status : 404,
                data : "no-have-user"
            })
        }

        req.dataPengguna = dataUser

        return next()
    })

    export const returnDataUserMw =
    asyncMw(async(req, res, next)=>{
        return res.status(200).json({
            status : 200,
            data : await req.dataPengguna
        })
    })
})

```

12. src/repository/baseRepository.ts

```

import { Prisma } from "@prisma/client";
import _ from "lodash";
import {
    Aggregate,
    AnyRecord,
    BaseOption,
    CountArgs,
    Find,
    ModelName,
    models,
    ModelScalarFields,
    ModelStructure,
    ModelTypes,
} from "../prisma-repo";

export const extractCondition = <Cursor, Where>(
    conditions: Cursor | Where | number | string
) => {
    const dbCond = _.isObject(conditions)
        ? conditions
        : { id: _.toNumber(conditions) };

    return dbCond;
};

/**
 * @param modelName - The model name
 */

const BaseRepository = <
    T extends ModelName,
    Where extends ModelTypes[T]["Where"],
    Select extends ModelTypes[T]["Select"],
    Include extends ModelTypes[T]["Include"],
    Create extends ModelTypes[T]["Create"],
    Update extends ModelTypes[T]["Update"],

```

```

Cursor extends ModelTypes[T]["Cursor"],
Order extends ModelTypes[T]["Order"],
Delegate extends ModelTypes[T]["Delegate"],
GroupBy extends ModelTypes[T]["GroupBy"],
Scalar extends ModelScalarFields<T>,
Model extends ModelStructure[T]
>(
  modelName: T
) => {
  abstract class AbstractBaseRepository {
    protected static modelName: T = modelName;

    /**
     * Find zero or more `model` that matches the
    filter.\
     * Note, that providing `undefined` is treated as
    the value not being there.
     */

    public static async findAll(
      conditions: Where | number | string,
      filterQueryParams: AnyRecord = {},
      query: AnyRecord = {},
      option: Find<Select, Include, Cursor, Order,
Scalar> = {}
    ) {
      const limit = +(query.limit === "all" ? 0 :
_.get(query, "limit", 10));
      const offset =
        query.page && query.page > 0 ? limit *
(query.page - 1) : 0;
      const otherOptions = _.omit(query, ["limit",
"offset", "page"]);

      const where = {
        ...extractCondition(conditions),
        ...filterQueryParams,
        ...otherOptions,
      };

      return {
        // @ts-ignore
        rows: (await
AbstractBaseRepository.model.findMany({
          // @ts-ignore
          where,
          ...option,
          skip: offset,
          ...(limit > 0 && { take: limit }),
        })) as Model[],
        /* @ts-ignore */
        count: await this.count(where),
      };
    }

    /**
     * Alternative of `findAll`.\
     * It works same as `findOne` but only have
    different names.\

```



```

    * It exists for anyone who prefer to use prisma
    `functions` original name.
    */

    public static async findMany(...params:
Parameters<typeof this.findAll>) {
        return AbstractBaseRepository.findAll(...params);
    }

    /**
    * Find the first `model` that matches the filter.\
    * Note, that providing `undefined` is treated as
    the value not being there.
    */

    public static async findOne(
        conditions: Where | number | string,
        option: Find<Select, Include, Cursor, Order,
Scalar> = {}
    ) {
        const where = extractCondition(conditions);

        // @ts-ignore
        return AbstractBaseRepository.model.findFirst({
            // @ts-ignore
            where,
            ...option,
        }) as Promise<Model | null>;
    }

    /**
    * Alternative of `findOne`.\
    * It works same as `findOne` but only have
    different names.\
    * It exists for anyone who prefer to use prisma
    `functions` original name.
    */

    public static async findFirst(...params:
Parameters<typeof this.findOne>) {
        return AbstractBaseRepository.findOne(...params);
    }

    /**
    * Find zero or one `model` that matches the
    filter.\
    * Note, that providing `undefined` is treated as
    the value not being there.\
    * It works same as `findOne` or `findFirst` but
    only accept a unique column.
    */

    public static async findUnique(
        conditions: Cursor | number | string,
        option: BaseOption<Include, Select> = {}
    ) {
        const where = extractCondition(conditions);

        return AbstractBaseRepository.model.findUnique({
            // @ts-ignore

```

```

        where,
        ...option,
    }) as Promise<Model | null>;
}

/**
 * Create a `model`.
 */

public static async create(
    data: Create,
    option: BaseOption<Include, Select> = {}
) {
    // @ts-ignore
    return AbstractBaseRepository.model.create({
        data,
        ...option,
    }) as Promise<Model>;
}

/**
 * Update a `model`.
 */

public static async update(
    conditions: Where | number | string,
    data: Update | Create,
    option: BaseOption<Include, Select> = {}
) {
    const where = extractCondition(conditions);

    // @ts-ignore
    return AbstractBaseRepository.model.update({
        data,
        // @ts-ignore
        where,
        ...option,
    }) as Promise<Model>;
}

/**
 * Delete any `model` that match with the
conditions.
 */

public static async delete(conditions: Where |
number | string) {
    const where = extractCondition(conditions);

    // @ts-ignore
    return AbstractBaseRepository.model.deleteMany({
        // @ts-ignore
        where,
    }) as Promise<Prisma.BatchPayload>;
}

/**
 * Delete a `model`.
 */

```

```

        public static async deleteOne(conditions: Where |
number | string) {
            const where = extractCondition(conditions);

            // @ts-ignore
            return AbstractBaseRepository.model.delete({
where }) as Promise<Model>;
        }

        /**
         * Create or update one `model`.
         */

        public static async updateOrCreate(
            conditions: Where | number | string,
            data: Create,
            option: Find<Select, Include, Cursor, Order,
Scalar> = {}
        ) {
            const obj = await
AbstractBaseRepository.findOne(conditions, option);

            if (obj) return
AbstractBaseRepository.update(conditions, data,
option);

            return AbstractBaseRepository.create(data);
        }

        /**
         * Alternative of `updateOrCreate`.
         * It works same as `updateOrCreate` but only have
different names.
         * It exists for anyone who prefer to use prisma
`functions` original name.
         */
        public static async upsert(
            ...params: Parameters<typeof this.updateOrCreate>
        ) {
            return
AbstractBaseRepository.updateOrCreate(...params);
        }

        /**
         * Create many `model`.
         */
        public static async bulkCreate(
            data: Prisma.Enumerable<Create>,
            skipDuplicates = true
        ) {
            // @ts-ignore
            return AbstractBaseRepository.model.createMany({
                data,
                // @ts-ignore
                skipDuplicates,
            }) as Promise<Prisma.BatchPayload>;
        }

```

```

/**
 * Alternative of `bulkCreate`.\
 * It works same as `bulkCreate` but only have
different names.\
 * It exists for anyone who prefer to use prisma
`functions` original name.
 */
public static async createMany(
  ...params: Parameters<typeof this.bulkCreate>
) {
  return
AbstractBaseRepository.bulkCreate(...params);
}

/**
 * Update zero or more `model`.
 * Note, that providing `undefined` is treated as
the value not being there.
 */

public static async bulkUpdate(
  where: Where,
  data: Prisma.Enumerable<Update>
) {
  // @ts-ignore
  return AbstractBaseRepository.model.updateMany({
    data,
    where,
  }) as Promise<Prisma.BatchPayload>;
}

/**
 * Alternative of `bulkUpdate`.\
 * It works same as `bulkUpdate` but only have
different names.\
 * It exists for anyone who prefer to use prisma
`functions` original name.
 */
public static async updateMany(
  ...params: Parameters<typeof this.bulkUpdate>
) {
  return
AbstractBaseRepository.bulkUpdate(...params);
}

/**
 * Count the number of `model`.\
 * Note, that providing `undefined` is treated as
the value not being there.
 */

public static async count(
  conditions: Where | number | string,
  option: CountArgs<Select, Cursor, Order, Scalar>
= {}
) {
  const where = extractCondition(conditions);

  // @ts-ignore

```

```

        return AbstractBaseRepository.model.count({
            // @ts-ignore
            where,
            ...option,
        }) as Promise<number>;
    }

    public static async groupBy(
        conditions: Where | number | string,
        aggregator: Omit<
            // @ts-ignore
            Parameters<typeof this.model.aggregate>[0],
            "where" | "cursor"
        > & {
            groupBy: Scalar[];
        }
    ) {
        const where = extractCondition(conditions);

        // @ts-ignore
        return AbstractBaseRepository.model.groupBy({
            // @ts-ignore
            where,
            ...aggregator,
        }) as GroupBy;
    }

    /**
     * Allows you to perform aggregations operations on
     * a `model`.
     * Note, that providing `undefined` is treated as
     * the value not being there.
     * If no any kind of aggregator provided, will use
     * `count` by default.
     */

    public static aggregate(
        conditions: Where | number | string,
        aggregator: Omit<
            // @ts-ignore
            Parameters<typeof this.model.aggregate>[0],
            "cursor" | "take" | "skip" | "orderBy" |
            "where"
        >,
        option: Aggregate<Cursor, Order, Scalar> = {}
    ) {
        // @ts-ignore
        const aggregate = AbstractBaseRepository.model
            .aggregate as Delegate["aggregate"];
        const where = extractCondition(conditions);

        if (_.isEmpty(aggregator)) {
            // @ts-ignore
            // eslint-disable-next-line no-param-reassign,
            no-underscore-dangle
            aggregator._count = true;
        }

        // @ts-ignore

```

```

        return aggregate({ where, ...aggregator,
...option }) as ReturnType<
        typeof aggregate
    >;
    }

    public static get model(): Delegate {
        // @ts-ignore
        return models[AbstractBaseRepository.modelName];
    }
}

return AbstractBaseRepository;
};

export default BaseRepository;

```

13. prisma/schema.prisma

```

datasource db {
  provider = "mongodb"
  url      = env("DATABASE_URL")
}

generator client {
  provider = "prisma-client-js"
}

model KecepatanAngin {
  id          String    @id @default(auto()) @map("_id")
  @db.ObjectId
  value       Int
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
  @@map("kecepatan-angin")
}

model TinggiGelombang {
  id          String    @id @default(auto()) @map("_id")
  @db.ObjectId
  value       Int
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
  @@map("tinggi-gelombang")
}

model KuatArus {
  id          String    @id @default(auto()) @map("_id")
  @db.ObjectId
  value       Int
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
  @@map("kuat-arus")
}

model SuhuLingkungan {
  id          String    @id @default(auto()) @map("_id")
  @db.ObjectId

```

```

        value      Int
        createdAt  DateTime @default(now())
        updatedAt  DateTime @updatedAt
        @@map("suhu-lingkungan")
    }

    model User {
        id          String @id @default(auto()) @map("_id")
        @db.ObjectId
        name         String
        instansi     String
        email        String
        password     String
    }

```

14. src/route/kecepatan-angin.ts

```

const router = Router();

// POST /kecepatan-angins
router.post(
    "/",
    kecepatanAngin.createKecepatanAnginMw,
    kecepatanAngin.returnKecepatanAnginMw
);

// GET /kecepatan-angins
router.get(
    "/",
    kecepatanAngin.getKecepatanAnginsMw,
    kecepatanAngin.returnKecepatanAnginsMw
);

// GET /kecepatan-angins/grafik?year=?&month=?
router.get(
    "/grafik",
    kecepatanAngin.getGrafikKecepatanAnginsMw,
    kecepatanAngin.returnGrafikKecepatanAnginsMw
);

// GET /kecepatan-angins/average?year=?&month=?
router.get(
    "/average",
    kecepatanAngin.getDayAverageKecepatanAnginMw,
    kecepatanAngin.returnDayAverageKecepatanAnginsMw
);

// GET /kecepatan-angins/:id
router.get(
    "/:id",
    kecepatanAngin.getKecepatanAnginMw,
    kecepatanAngin.returnKecepatanAnginMw
);

export default router;

```

15. src/route/kuat-arus.ts

```

const router = Router();

// POST /kecepatan-angins
router.post("/", kuatArus.createKuatArusMw,
kuatArus.returnKuatArusMw);

// GET /kecepatan-angins
router.get("/", kuatArus.getKuatArussMw,
kuatArus.returnKuatArussMw);

// GET /kecepatan-angins/grafik?year=?&month=?
router.get(
  "/grafik",
  kuatArus.getGrafikKuatArusMw,
  kuatArus.returnGrafikKuatArusMw
);

// GET /kecepatan-angins/average?year=?&month=?
router.get(
  "/average",
  kuatArus.getDayAverageKuatArusMw,
  kuatArus.returnDayAverageKuatArusMw
);

// GET /kecepatan-angins/:id
router.get("/:id", kuatArus.createKuatArusMw,
kuatArus.returnKuatArusMw);

export default router;

```

16. src/route/post-all-data.ts

```

const router = Router();

router.post('/', postAllData.createPostAllDataMw,
postAllData.returnPostAllDataMw);

export default router

```

17. src/route/suhu-lingkungan.ts

```

const router = Router();

// POST /kecepatan-angins
router.post(
  "/",
  suhuLingkungan.createSuhuLingkunganMw,
  suhuLingkungan.returnSuhuLingkunganMw
);

// GET /kecepatan-angins
router.get(
  "/",
  suhuLingkungan.getSuhuLingkungansMw,
  suhuLingkungan.returnSuhuLingkungansMw
);

```



```

router.get(
  "/grafik",
  suhuLingkungan.getGrafikSuhuLingkunganMw,
  suhuLingkungan.returnGrafikSuhuLingkunganMw
);

// GET /kecepatan-angins/average?year=?&month=?
router.get(
  "/average",
  suhuLingkungan.getDayAverageSuhuLingkunganMw,
  suhuLingkungan.returnDayAverageSuhuLingkunganMw
);

// GET /kecepatan-angins/:id
router.get(
  "/:id",
  suhuLingkungan.getSuhuLingkunganMw,
  suhuLingkungan.returnSuhuLingkunganMw
);

export default router;

```

18. src/route/tinggi-gelombang.ts

```

const router = Router();

// POST /kecepatan-angins
router.post(
  "/",
  tinggiGelombang.createTinggiGelombangMw,
  tinggiGelombang.returnTinggiGelombangMw
);

// GET /kecepatan-angins
router.get(
  "/",
  tinggiGelombang.getTinggiGelombangMw,
  tinggiGelombang.returnTinggiGelombangMw
);

// GET /kecepatan-angins/grafik?year=?&month=?
router.get(
  "/grafik",
  tinggiGelombang.getGrafikTinggiGelombangMw,
  tinggiGelombang.returnGrafikTinggiGelombangMw
);

// GET /kecepatan-angins/average?year=?&month=?
router.get(
  "/average",
  tinggiGelombang.getDayAverageTinggiGelombangMw,
  tinggiGelombang.returnDayAverageTinggiGelombangMw
);

// GET /kecepatan-angins/:id
router.get(
  "/:id",
  tinggiGelombang.getTinggiGelombangMw,

```

```
        tinggiGelombang.returnTinggiGelombangMw
    );
export default router;
```

19. src/route/user.ts

```
import { Router } from "express";
import *as user from '../middleware/user'

const router = Router()

//POST /register
router.post(
    '/register',
    user.createPostDataUserMw,
    user.returnPostDataUserMw
)

router.post(
    '/login',
    user.checkDataUserMw,
    user.returnCheckDataUserMw
)

router.get(
    '/:id',
    user.getDataUserMw,
    user.returnDataUserMw
)

export default router
```

Lampiran III Dokumentasi Dengan Klien

Adapun ampiran untuk dokumentasi dengan klien sebagai berikut :

1. Diskusi dan persetujuan design aplikasi (WA)



2. Pengujian Aplikasi Ke Klien (Gedung F ITERA)



3. Persetujuan Penyelesaian project yang telah diuji kepada klien (Gedung F ITERA)



Lampiran IV Dokumen Persetujuan Dengan Klien

Adapun dokumen-dokumen dengan klien sebagai berikut :

1. Dokumen Persetujuan Rancangan Pengembangan Aplikasi

Dokumen Persetujuan Rancangan Pengembangan Aplikasi

Proyek : Pengembangan Aplikasi Mobile Berbasis Android

Spesifikasi Kebutuhan :

Aplikasi Mobile Berbasis Android ini bertujuan untuk membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan dan membantu para peneliti yang ingin meneliti lingkungan laut di Teluk Kiluan, Kabupaten Tanggamus, Provinsi Lampung dalam mengumpulkan data kondisi laut dengan menggunakan aplikasi.

Timeline Planning:

Tugas	Waktu Mulai	Target Selesai
Pembuatan Backend dan Database	14 September 2022	25 September 2022
Pembuatan Halaman dan Fitur Aplikasi	26 September 2022	28 Oktober 2022
Menyambungkan Database dan Mobile	31 Oktober 2022	04 November 2022
Review dan Pengujian Aplikasi Secara Fungsional	07 November 2022	25 November 2022
Upload Aplikasi ke Playstore	27 November 2022	09 Desember 2022

Pada tahap **Review dan Pengujian Aplikasi Secara Fungsional** akan dilakukan pengecekan dan pengujian aplikasi. Bilamana terdapat masalah pada aplikasi maka akan dilakukan perbaikan dan bilamana terdapat penambahan fitur atau perubahan desain aplikasi maka akan dikerjakan serta bilamana telah dikerjakan maka akan di review dan di uji ulang.

Anggaran:

Layanan / Alat / Jasa	Harga (Rp)
Hosting Database dengan menggunakan Atlas (MongoDB)	0
Hosting Backend dengan menggunakan fly.io	0
Pembelian Akun Playstore	400.000

Mengalokasikan anggaran sebesar Rp 400.000,- untuk proyek penelitian pengembangan aplikasi mobile berbasis android dengan selesai.

Dengan adanya Dokumen Persetujuan Rancangan Pengembangan Aplikasi ini, pengembang dan juga klien dapat memiliki pandangan yang lebih jelas tentang kebutuhan, jadwal, dan dana yang dibutuhkan. Dokumen ini juga dapat digunakan sebagai acuan untuk memastikan bahwa proyek penelitian tetap berada pada jalur yang benar dan dapat memberikan nilai bisnis yang diharapkan.

Dengan ini disetujui bahwa proyek penelitian pengembangan aplikasi akan dilaksanakan sesuai dengan dokumen disusun. Dokumen ini ditandatangani oleh pihak ketua proyek penelitian dan pengembang.

Disetujui Oleh:

Ketua Proyek Penelitian



Dr. Meezan Ardhanu Asagabaldan, S.Pt., M.Si.

Pengembang Aplikasi



Abdurrahman Farras

2. Dokumen Persetujuan Penyelesaian Pengembangan Aplikasi

Dokumen Persetujuan Penyelesaian Pengembangan Aplikasi

Telah diselesaikan pengembangan aplikasi mobile berbasis android yang bertujuan untuk membantu para nelayan lokal dalam memantau kondisi laut yang aman untuk berlayar menangkap ikan dan membantu para peneliti yang ingin meneliti lingkungan laut di Teluk Kiluan, Kabupaten Tanggamus, Provinsi Lampung, dalam mengumpulkan data kondisi laut dengan menggunakan aplikasi. Dengan spesifikasi kebutuhan aplikasi yaitu:

No	Kebutuhan
1.	Aplikasi dapat menampilkan informasi arah angin secara <i>real-time</i>
2.	Aplikasi dapat menampilkan informasi kecepatan angin secara <i>real-time</i>
3.	Aplikasi dapat menampilkan informasi kekuatan arus secara <i>real-time</i>
4.	Aplikasi dapat menampilkan informasi tinggi gelombang secara <i>real-time</i>
5.	Aplikasi dapat menampilkan informasi suhu lingkungan secara <i>real-time</i>
6.	Aplikasi dapat menampilkan kondisi aman atau tidak nya lingkungan Teluk Kiluan berdasarkan dari data level keamanan dari sistem IoT
7.	Aplikasi dapat melakukan <i>login</i> akun sebagai peneliti
8.	Aplikasi dapat melakukan <i>register</i> akun untuk mendaftar sebagai peneliti
9.	Aplikasi dapat menampilkan grafik data riwayat dan rata-rata per hari data pemantauan kecepatan angin, tinggi gelombang, suhu udara, dan kecepatan gelombang bila masuk sebagai peneliti.
10.	Aplikasi dapat menampilkan visualisasi detail arah angin (Seperti Kompas) bila masuk sebagai peneliti.
11.	Aplikasi dapat menampilkan ramalan cuaca di daerah Teluk Kiluan dengan rentan waktu hari ini, besok, dan lusa berdasarkan data BMKG

Berdasarkan data kebutuhan fungsional tersebut, telah dilakukan juga pengujian pada aplikasi dengan skenario-skenario yang akan terjadi sebagai berikut.

Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil yang Didapatkan
Memantau kondisi lingkungan laut secara <i>real-time</i>	Sistem IoT mengirimkan data arah angin dengan menjadi "Utara", suhu menjadi "24", kecepatan angin menjadi "50", tinggi gelombang menjadi "7", kecepatan Angin menjadi "30", kecepatan gelombang "40" ke server	Aplikasi menampilkan kondisi keadaan laut teluk kiluan menjadi "Cuaca Tidak Aman", data arah angin dengan menjadi "Utara", suhu menjadi "24", kecepatan angin menjadi "50", tinggi gelombang menjadi "7", kecepatan Angin menjadi "30", kecepatan gelombang "40"	Sesuai
Memasukan email dan <i>password</i> yang salah lalu klik tombol Masuk	Email : contoh@kl.com Password : ContohSandi	Aplikasi akan menolak dan kembali ke halaman login dan muncul pesan "Email atau Password Salah"	Sesuai
Memasukan email dan <i>password</i> lalu klik tombol <i>login</i>	Memasukan email dan <i>password</i> lalu klik tombol <i>login</i>	Aplikasi menerima akses login dan Pengguna bisa mengakses fitur detail lengkap	Sesuai
Tidak memasukan data pada form <i>login</i> secara lengkap lalu klik Masuk	Mengosongkan form email atau mengosongkan form Password	Memunculkan pesan "Lengkapi data"	Sesuai
Tidak memasukan data pada form <i>register</i> secara lengkap lalu klik Daftar	Mengosongkan salah satu form	Akan ada pesan yang menyuruh pengguna untuk melengkapi data diri.	Sesuai
Mengisi form dengan lengkap dan benar lalu klik Daftar	Mengisi nama lengkap, Instansi, Kata sandi, dan kata sandi ulang yang sama dengan sandi	Aplikasi akan menerima pendaftaran dan akan langsung masuk ke menu <i>login</i>	Sesuai
Mengisi form Kata sandi berbeda dengan konfirmasi <i>password</i>	Kata sandi : Aku123 Konfirmasi Kata Sandi : aku123	Akan ada pesan yang menyatakan bahwa data yang di input salah	Sesuai

Menekan <i>card</i> arah angin pada halaman beranda saat pengguna sudah masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Aplikasi menampilkan kompas arah angin dari teluk kiluan, menampilkan derajat arah angin tersebut, serta menampilkan dominan arah angin itu sendiri	Sesuai
Sistem IoT mengirimkan data arah angin secara <i>real-time</i>	Sistem IoT mengirimkan data 180 derajat ke <i>firebase</i>	Aplikasi menerima data tersebut dan mengkonvert data tersebut menjadi arah kompas yang menunjukan ke selatan secara <i>real-time</i>	Sesuai
Pengguna masuk ke halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan <i>icon</i> pengguna pada halaman beranda	Pengguna akan masuk ke halaman profil dimana akan ada tombol masuk dan daftar	Sesuai
Pengguna masuk ke halaman beranda saat pengguna sudah masuk sebagai peneliti	Pengguna menekan <i>icon</i> pengguna pada halaman beranda	Aplikasi menampilkan data pribadi pengguna berupa nama, email, dan instansi dan terdapat tombol keluar untuk keluar sebagai peneliti	Sesuai
Pengguna <i>logout</i> dari peneliti	Pengguna menekan tombol keluar kemudian menekan tombol setuju untuk keluar sebagai peneliti	Aplikasi akan masuk ke halaman beranda dan sudah tidak bisa lagi melihat detail dari kondisi lingkungan laut teluk kiluan.	Sesuai

Mengisi form email yang sudah di gunakan lalu klik tombol "daftar"	Email : "farras@gmail.com"	Aplikasi akan menampilkan "Email Sudah Terpakai"	Sesuai
Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan salah dari <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda	Tidak terjadi apa-apa	Sesuai
Menekan <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda saat pengguna setelah masuk sebagai peneliti	Pengguna menekan salah dari <i>card</i> kecepatan angin, suhu lingkungan, tinggi gelombang, kecepatan gelombang pada halaman beranda	Pengguna masuk ke halaman detail yang berisikan grafik mingguan dan rata-rata harian dari teluk kiluan	Sesuai
Melihat data pada tahun dan bulan tertentu	Pengguna menekan tombol <i>filter</i> tahun dan memilih pilihan "2022" serta menekan tombol <i>filter</i> bulan dan memilih "November"	Aplikasi akan menampilkan grafik dan rata-rata perhari kondisi laut teluk kiluan pada bulan dan waktu tersebut.	Sesuai
Menekan <i>card</i> arah angin pada halaman beranda saat pengguna belum masuk sebagai peneliti	Pengguna menekan <i>card</i> arah angin pada halaman beranda	Tidak terjadi apa-apa	Sesuai

Masuk ke halaman perkiraan cuaca	Pengguna menekan tombol perkiraan cuaca pada halaman beranda	Aplikasi akan menampilkan perkiraan cuaca pada hari ini, besok dan lusa menurut data dari BMKG	Sesuai
----------------------------------	--	--	--------

Dari Pengujian yang telah dilakukan maka membuktikan bahwa aplikasi sudah selesai di kembangkan dan sudah layak dipublikasikan.

Dengan ditandatanganinya **Dokumen Penyelesaian Pengembangan Aplikasi** ini, pengembang dan ketua proyek penelitian sepakat menyatakan bahwa **Aplikasi Mobile Berbasis Android** telah selesai di kembangkan dan sudah boleh dipublikasikan kepada **Nelayan Teluk Kiluan**, Kabupaten Tanggamus, Provinsi Lampung dan **Peneliti** yang akan meneliti lingkungan laut di Teluk Kiluan, Kabupaten Tanggamus, Provinsi Lampung

Disetujui Oleh:

Ketua Proyek Penelitian



Dr. Meezan Ardhana Asagabaldan, S.Pd., M.Si.

Pengembang Aplikasi



Abdurrahman Farras

Lampiran V Hasil Kuesioner Kepada Peneliti dan Nelayan

Adapun Hasil Kuesioner dengan peneliti dan nelayan adalah sebagai berikut:

Kuesioner Penelitian

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : Dr. Mersan Arelhanu Asagabalahan
Pekerjaan : Dosen

Daftar Kuesioner:
Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Peneliti merasa mudah dalam menggunakan aplikasi	✓				
Aplikasi membantu peneliti dalam melihat kuat arus laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti untuk melihat arah angin secara <i>real-time</i> di laut teluk kiluan.		✓			
Aplikasi membantu peneliti dalam melihat tinggi gelombang laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat suhu lingkungan laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat		✓			

kecepatan arus laut teluk kiluan secara <i>real-time</i>					
Aplikasi membantu peneliti dalam melihat grafik data arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih			✓		
Aplikasi membantu peneliti dalam mendapatkan data rata-rata arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih.			✓		

Kuesioner Penelitian

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : Andika Sebiawan
Pekerjaan : Peneliti

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Peneliti merasa mudah dalam menggunakan aplikasi	✓				
Aplikasi membantu peneliti dalam melihat kuat arus laut teluk kiluan secara <i>real-time</i>	✓				
Aplikasi membantu peneliti untuk melihat arah angin secara <i>real-time</i> di laut teluk kiluan.		✓			
Aplikasi membantu peneliti dalam melihat tinggi gelombang laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat suhu lingkungan laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat		✓			

kecepatan arus laut teluk kiluan secara <i>real-time</i>					
Aplikasi membantu peneliti dalam melihat grafik data arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih			✓		
Aplikasi membantu peneliti dalam mendapatkan data rata-rata arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih.			✓		

Kuesioner Penelitian

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : Rizki Dimas Permana, S.Kel, M.Si.
Pekerjaan : Dosen

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Peneliti merasa mudah dalam menggunakan aplikasi		✓			
Aplikasi membantu peneliti dalam melihat kuat arus laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti untuk melihat arah angin secara <i>real-time</i> di laut teluk kiluan.			✓		
Aplikasi membantu peneliti dalam melihat tinggi gelombang laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat suhu lingkungan laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat					

kecepatan arus laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat grafik data arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih			✓		
Aplikasi membantu peneliti dalam mendapatkan data rata-rata arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih.			✓		

Kuesioner Penelitian

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : *Budhi Agung Prasetyo*
Pekerjaan : *Dosen & Peneliti*

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Peneliti merasa mudah dalam menggunakan aplikasi	✓				
Aplikasi membantu peneliti dalam melihat kuat arus laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti untuk melihat arah angin secara <i>real-time</i> di laut teluk kiluan,		✓			
Aplikasi membantu peneliti dalam melihat tinggi gelombang laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat suhu lingkungan laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat		✓			

kecepatan arus laut teluk kiluan secara <i>real-time</i>		✓			
Aplikasi membantu peneliti dalam melihat grafik data arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih		✓			
Aplikasi membantu peneliti dalam mendapatkan data rata-rata arus laut, tinggi gelombang, suhu lingkungan, kecepatan arus, kuat arus laut teluk kiluan pada bulan dan tahun yang dipilih.		✓			

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : Juki
Pekerjaan : nelayan

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi		✓			
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidaknya untuk berlayar menangkap ikan dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek arah angin laut dengan		✓			

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : WILWIT SURI
Pekerjaan :

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi			✓		
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek arah angin laut dengan	✓				

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : *Sadam Laman*
Pekerjaan : *Nelayan*

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek arah angin laut dengan	✓				

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kluang. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : *Saihan*
Pekerjaan : *Nelayan*

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi			✓		
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidaknya untuk berlayar menangkap ikan dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek arah angin laut dengan		✓			

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi		✓			

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : *Wahid Abdur*
Pekerjaan : *Nelayan*

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi		✓			
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi		✓			
Nelayan merasa terbantu dalam mengecek arah angin laut dengan		✓			

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : H Yusy
Pekerjaan : Nelayan

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek arah angin laut dengan	✓				

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Kuesioner Lapangan

Berikut ini adalah kuesioner yang berkaitan tentang kepuasan pengguna dalam penggunaan aplikasi pemantauan kondisi lingkungan laut Teluk Kiluan. Oleh karena itu saya memohon dengan hormat kesediaan dan partisipasi anda dalam mengisi kuesioner yang ada. Saya ucapkan banyak terimakasih.

Identitas Responden

Nama : *Amy Alias*
Pekerjaan : *Itikan*

Daftar Kuesioner:

Mohon untuk memberikan tanda (V) pada setiap pertanyaan yang anda pilih

Pernyataan	Sangat Setuju	Setuju	Cukup Setuju	Kurang Setuju	Tidak Setuju
Nelayan merasa mudah dalam menggunakan aplikasi			✓		
Nelayan merasa terbantu untuk memantau kondisi laut aman atau tidak nya untuk berlayar menangkap ikan dengan menggunakan aplikasi	✓				
Nelayan merasa terbantu dalam mengecek tinggi gelombang laut dengan menggunakan aplikasi				✓	
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi				✓	
Nelayan merasa terbantu dalam mengecek arah angin laut dengan				✓	

menggunakan aplikasi					
Nelayan merasa terbantu dalam mengecek kecepatan ombak laut dengan menggunakan aplikasi				✓	
Nelayan merasa terbantu dalam mengecek kecepatan angin laut dengan menggunakan aplikasi				✓	
Nelayan merasa terbantu untuk Menyusun rencana pelayaran penangkapan ikan dengan melihat ramalan cuaca pada aplikasi	✓				

Lampiran VI Hasil User Guide

Berikut Merupakan user guide yang diberikan kepada klien



Memantau Kondisi Laut Secara Real-Time

Ketika Pengguna sudah menginstall aplikasi maka pengguna dapat masuk ke dalam aplikasi



Gambar di atas merupakan tampilan awal jika membuka aplikasi. Setelah sistem telah selesai mengambil data dan mengecek sinyal maka sistem akan mengirimkan pengguna ke tampilan **Beranda**



Pada gambar diatas merupakan tampilan **Beranda**, dimana pengguna dapat melihat data keadaan laut yang aman ataupun tidak aman untuk berlayar ataupun berwisata. Serata terdapat data real-time kondisi keadaan laut seperti kecepatan angin, tinggi gelombang, kecepatan angin, arah angin dan suhu lingkungan.

Mendaftar Menjadi Peneliti

Pada Tahap awal pengguna harus sudah ada pada tampilan **Beranda**, kemudian pengguna menekan icon profil seperti gambar di bawah ini



Kemudian pengguna akan diarahkan ke halaman profil seperti gambar di bawah ini.



Pengguna diharuskan menekan tombol “Daftar Sebagai Pengguna”. Setelah itu pengguna akan diarahkan ke arah halaman **Daftar**.



Pengguna diwajibkan mengisi identitas diri dengan baik dan benar. Bila terdapat kesalahan maka pengguna akan mendapatkan informasi kesalahan dari sistem seperti dapat terlihat pada tampilan di bawah ini.



Bila Semua data telah berhasil maka akan keluar informasi seperti di bawah ini.



Registrasi sudah berhasil dilakukan, pengguna harus menekan tombol “Selanjutnya” untuk melakukan proses login sebagai peneliti

Masuk Sebagai Peneliti

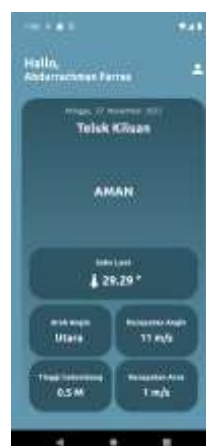
Sebelumnya pengguna harus masuk ke tampilan **Profil**, kemudian pengguna menekan tombol “Masuk sebagai Peneliti”. Setelah menekan tombol maka pengguna akan langsung masuk ke halaman **Masuk**.



Pada tahap ini pengguna harus memasukkan email dan kata sandi yang sudah terdaftar sebagai peneliti.



Jika pengguna sudah masuk sebagai peneliti, maka pengguna harus menekan tombol ‘selanjutnya’. Pengguna akan masuk ke halaman **Beranda** dan pengguna sudah bisa membuka fitur yang dimiliki oleh Peneliti.

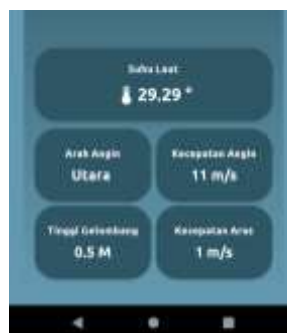


Pada gambar di atas dapat terlihat bahwa sudah ada nama peneliti pada jata setelah “Hallo,”. Pengguna juga sudah bisa melihat detail dan grafik kondisi Teluk Kiluan. Pengguna juga dapat melihat profil diri dengan menekan icon profil. Pengguna akan masuk ke halaman **Profil** dan dapat melihat profil peneliti.



Melihat Detail Kondisi Laut dan Visualisasi keadaan Laut

Pada tahap ini pengguna diwajibkan masuk terlebih dahulu sebagai peneliti, setelah itu pengguna bisa menekan card “Kecepatan Angin”, “Suhu Lingkungan”, “Tinggi Gelombang” dan “Kecepatan Arus”.



Setelah mengklik card tersebut, maka pengguna dapat masuk kedalam halaman **Detail**.

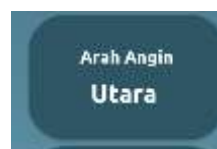


Pada Gambar di atas dapat terlihat visualisasi data dari grafik kondisi laut setiap minggu, serta dapat dilihat rata rata kondisi laut setiap harinya dengan parameter bulan dan tahun yang dapat diganti.



Melihat Arah Angin

Pada tahap ini pengguna harus masuk terlebih dahulu sebagai peneliti. Kemudian pengguna menekan card “Arah Angin “ pada halaman **Beranda**.



Kemudian pengguna akan masuk ke halaman **Arah Angin**. Pada halaman ini pengguna dapat melihat arah angin yang ditunjukkan dengan menggunakan kompas arah angin seperti gambar di bawah ini.



Melihat Informasi Developer

Halaman ini dapat diakses oleh setiap role pengguna (masyarakat umum dan peneliti) dengan menekan icon informasi pada halaman **Profil** seperti gambar di bawah ini.



Setelah pengguna menekan icon tersebut, pengguna akan diarahkan ke halaman **Informasi**. Pada halaman ini pengguna dapat melihat informasi berupa nama, email dan asal studi dari tim peneliti, pengembangan aplikasi dan tim lapangan seperti gambar di bawah ini.



Keluar Sebagai Peneliti

Setelah masuk sebagai peneliti dan ingin keluar sebagai peneliti, maka pengguna harus masuk ke halaman **Profil** terlebih dahulu. Akan ada tombol “Keluar” yang dapat diklik.



Akan ada pilihan “IYA” dan “TIDAK”. Jika pengguna ingin keluar sebagai peneliti maka menekan tombol “IYA”. Ketika pengguna menekan tombol “IYA” maka pengguna akan langsung masuk ke halaman **Beranda** sebagai pengguna biasa.

Lampiran VII Bukti Fuzzy Logic

```

263     sea_safety = get_sea_safety_level(3.24, 5.76);
264     Serial.print("Keamanan laut: ");
265     Serial.println(sea_safety);
266
267     /* ***** */
268

```

Output Serial Monitor x gdb-server

Message (Enter to send message to 'ESP32 Dev Module' on 'dev/ttyUSB0')

Ketinggian gelombang: 15.18 m
 Kecepatan gelombang: 0.16 m/s
 Keamanan laut: Bahaya
 Berhasil Mengirimkan ke Firebase

Rotasi sudut wX: 0.01, wY: 0.00, wZ: 0.00 rad/s
 Percepatan aX: -0.00, aY: 0.00, aZ: -0.00 m/s^2
 Suhu MPU6050: 26.51 Percepatan gabungan: 0.00 m/s^2
 Kecepatan rotasi gabungan: 0.01 rad/s
 Ketinggian gelombang: 18.84 m
 Kecepatan gelombang: 0.21 m/s
 Keamanan laut: Bahaya

```

263     sea_safety = get_sea_safety_level(1.42, 1.83);
264     Serial.print("Keamanan laut: ");
265     Serial.println(sea_safety);
266
267     /* ***** */
268

```

Output Serial Monitor x gdb-server

Message (Enter to send message to 'ESP32 Dev Module' on 'dev/ttyUSB0')

Ketinggian gelombang: 7.12 m
 Kecepatan gelombang: 0.16 m/s
 Keamanan laut: Waspada
 Berhasil Mengirimkan ke Firebase

Rotasi sudut wX: 0.02, wY: 0.00, wZ: 0.00 rad/s
 Percepatan aX: 0.00, aY: 0.00, aZ: -0.00 m/s^2
 Suhu MPU6050: 26.52 Percepatan gabungan: 0.00 m/s^2
 Kecepatan rotasi gabungan: 0.02 rad/s
 Ketinggian gelombang: 2.47 m
 Kecepatan gelombang: 0.06 m/s
 Keamanan laut: Waspada

```

261
262     // Dapatkan kategori keamanan (fuzzy) berdasarkan kondisi laut
263     sea_safety = get_sea_safety_level(0.21, 0.11);
264     Serial.print("Keamanan laut: ");
265     Serial.println(sea_safety);
266
267     /* ***** */
268

```

Output Serial Monitor x gdb-server

Message (Enter to send message to 'ESP32 Dev Module' on 'dev/ttyUSB0')

Ketinggian gelombang: 19.63 m
 Kecepatan gelombang: 0.19 m/s
 Keamanan laut: Aman
 Berhasil Mengirimkan ke Firebase

Rotasi sudut wX: 0.01, wY: 0.00, wZ: 0.00 rad/s
 Percepatan aX: -0.00, aY: 0.00, aZ: -0.00 m/s^2
 Suhu MPU6050: 26.54 Percepatan gabungan: 0.00 m/s^2
 Kecepatan rotasi gabungan: 0.01 rad/s
 Ketinggian gelombang: 8.00 m
 Kecepatan gelombang: 0.00 m/s
 Keamanan laut: Aman

Lampiran VIII Dokumentasi Pengujian

