



**PENGEMBANGAN APLIKASI PENENTUAN GRADE MAKANAN DAN  
MINUMAN BERBASIS ANDROID**

**KERJA PRAKTIK**



**Program Studi  
S1 Sistem Informasi**

**UNIVERSITAS  
Dinamika**

**Oleh:**

**HABIB FATKHUL ROHMAN**

**21410100050**

---

**FAKULTAS TEKNOLOGI DAN INFORMATIKA**

**UNIVERSITAS DINAMIKA**

**2024**

**PENGEMBANGAN APLIKASI PENENTUAN GRADE MAKANAN DAN  
MINUMAN BERBASIS ANDROID**

Diajukan sebagai salah satu syarat untuk menyelesaikan  
Program Sarjana



**Disusun Oleh :**

**Nama : Habib Fatkhul Rohman**

**Nim : 21410100050**

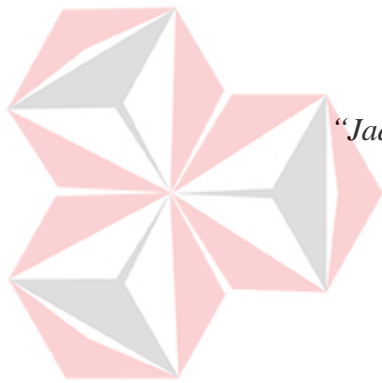
**Program : S1 (Strata Satu)**

**Jurusan : Sistem Informasi**

**FAKULTAS TEKNOLOGI DAN INFORMATIKA**

**UNIVERSITAS DINAMIKA**

**2024**



*“Jadilah baik, berbuatlah baik, dan cobalah yang terbaik”*

Habib Fatkhul Rohman

UNIVERSITAS  
**Dinamika**



*Laporan Kerja Praktik ini  
Saya persembahkan kepada  
semua orang yang terlibat dalam  
membantu penyusunan laporan ini.*

UNIVERSITAS  
Dinamika

**LEMBAR PENGESAHAN**

**PENGEMBANGAN APLIKASI PENENTUAN GRADE MAKANAN DAN  
MINUMAN BERBASIS ANDROID**

Laporan Kerja Praktik oleh

Habib Fatkhul Rohman

NIM : 21410100050

Telah diperiksa, diuji dan disetujui

Surabaya, 11 Juli 2024

Disetujui,

Dosen Pembimbing

Penyelia

Digitally signed by Pradita  
Maulidya Effendi  
DN: cn=Pradita Maulidya  
Effendi, o=Universitas  
Dinamika, ou=S1 Sistem  
Informasi,  
email=pradita@dinamika.ac.id,  
c=ID  
Date: 2024.08.01 14:06:33  
+07'00'

 **bangkit!**

**Pradita Maulidya Effendi, M.Kom.**


**Deti Anggraini Ekawati**

NIDN. 0720089401

NIP. 02022018017

Mengetahui,

Ketua Program Studi S1 Sistem Informasi

Digitally signed 

by Julianto

Date: 2024.08.02

11:58:06 +07'00'

**Julianto Lemantara, S.Kom., M.Eng**

NIDN. 0722108601

**PERNYATAAN**  
**PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH**

Sebagai mahasiswa **Universitas Dinamika**, Saya :

Nama : **Habib Fatkhul Rohman**

NIM : **21410100050**

Program Studi : **S1 Sistem Informasi**

Fakultas : **Fakultas Teknologi dan Informatika**

Jenis Karya : **Laporan Kerja Praktik**

Judul Karya : **PENGEMBANGAN APLIKASI PENENTUAN GRADE MAKANAN DAN MINUMAN BERBASIS ANDROID**

Menyatakan dengan sesungguhnya bahwa :

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, Saya menyetujui memberikan kepada **Universitas Dinamika** Hak Bebas Royalti Non-Fksklusif (*Non-Exclusive Royalty Free Right*) atas seluruh isi/sebagian karya ilmiah Saya tersebut diatas untuk disimpan, dialihmediakan, dan dikelola dalam bentuk pangkalan data (*database*) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama Saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut diatas adalah hasil karya asli Saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya, atau pendapat orang lain yang ada dalam karya ilmiah ini semata-mata hanya sebagai rujukan yang dicantumkan dalam Daftar Pustaka Saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiasi pada karya ilmiah ini, maka Saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada Saya.

Demikian surat pernyataan ini Saya buat dengan sebenar-benarnya.

Surabaya, 01 Juli 2024



Habib Fatkhul Rohman  
NIM : 21410100050

## ABSTRAK

Pola makan yang tidak sehat menjadi salah satu penyebab utama berbagai penyakit di Indonesia. Untuk mengatasi masalah ini, diperlukan solusi yang dapat membantu masyarakat dalam memilih makanan dan minuman yang sehat. Penelitian ini bertujuan untuk mengembangkan aplikasi *mobile* berbasis Android yang dapat menentukan *grade* makanan dan minuman berdasarkan kandungan nutrisinya. Aplikasi dikembangkan menggunakan pendekatan *waterfall* dan bahasa pemrograman yang digunakan adalah bahasa *Kotlin*. Fitur utama aplikasi adalah *scan* nutrisi menggunakan kamera *smartphone* untuk menentukan *grade* suatu produk. Sistem *grading* yang digunakan mengacu pada aturan *Nutri-Grade*, yang mengkategorikan produk ke dalam empat tingkatan yaitu A, B, C, D berdasarkan kandungan gula dan lemak jenuh. Hasil pengembangan menunjukkan bahwa aplikasi dapat diimplementasikan dengan baik, menerapkan arsitektur MVVM (*Model-View-ViewModel*) untuk pemisahan logika bisnis dari antarmuka pengguna. Pengujian menggunakan metode *Blackbox Testing* untuk memastikan bahwa fitur-fitur aplikasi berfungsi sesuai dengan yang diharapkan. Aplikasi ini diharapkan dapat membantu masyarakat dalam membuat keputusan yang lebih baik terkait pemilihan makanan dan minuman, sehingga berkontribusi pada peningkatan kesehatan masyarakat secara umum.

**Kata Kunci:** Android, *Grade* Makanan, MVVM, *Retrofit*, *Scan* Nutrisi

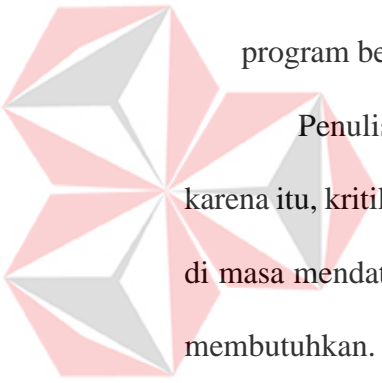
## KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan penulisan laporan Kerja Praktik dengan lancar. Laporan ini disusun berdasarkan pengalaman penulis dalam menyelesaikan Program Magang dan Studi Independen Bersertifikat (MSIB). Kegiatan ini berlangsung selama empat bulan, dimulai pada 19 Februari 2024 dan berakhir pada Juli 2024. Program ini diselenggarakan oleh Yayasan Dicoding Indonesia dalam program *Bangkit Academy 2024*, yang didukung oleh perusahaan teknologi terkemuka seperti Google, GoTo, dan Traveloka. Selama masa program, penulis memperoleh berbagai pengetahuan dan pengalaman yang sangat berharga. Manfaat yang didapat tidak hanya terbatas pada peningkatan kemampuan hard skill, tetapi juga pengembangan soft skills yang penting dalam dunia profesional. Dalam kesempatan ini, saya ingin menyampaikan ucapan terima kasih yang tulus kepada:

1. Orang tua dan keluarga yang selalu memberikan dukungan moral dan material.
2. Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia, yang telah menyelenggarakan program MSIB dan memberikan kesempatan berharga ini kepada saya.
3. Bapak Julianto Lemantara, selaku ketua program studi S1 Sistem Informasi Universitas Dinamika yang telah memberikan persetujuan untuk mengikuti program MSIB.
4. Bapak Tutut Wuriyanto, selaku dosen wali yang telah memberikan bimbingan selama mengikuti program MSIB.



5. Ibu Pradita Maulidya Effendi, selaku dosen pembimbing yang sudah membantu pendampingan selama pelaksanaan dan pelaporan KP dengan tulus dan sabar.
6. Bapak Wigananda Firdaus Putra Aditya, selaku ketua pelaksana program MSIB di Universitas Dinamika yang telah memberikan arahan dari proses pendaftaran hingga akhir pelaksanaan program MSIB.
7. Tim Bangkit yang telah menerima dan memberikan kesempatan kepada saya untuk melaksanakan studi independen di program Bangkit.
8. Teman-teman seperjuangan MSIB, Group MD-19 dan Tim Capstone Nutri-O-Matic yang telah memberikan semangat dan kerja sama yang baik selama program berlangsung.



Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun sangat saya harapkan untuk perbaikan di masa mendatang. Semoga laporan ini dapat bermanfaat bagi semua pihak yang membutuhkan.

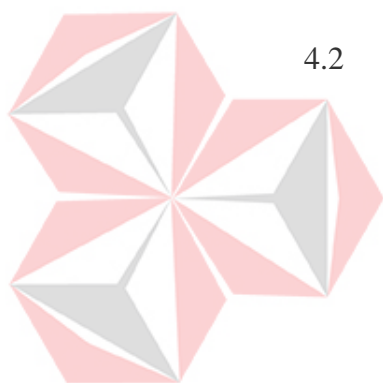
Surabaya, 1 Juli 2024

Penulis

## DAFTAR ISI

	<b>Halaman</b>
ABSTRAK .....	v
KATA PENGANTAR .....	vi
DAFTAR ISI.....	viii
DAFTAR GAMBAR .....	xi
DAFTAR TABEL.....	xiii
DAFTAR LAMPIRAN.....	xiv
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan .....	3
1.5 Manfaat .....	3
BAB II GAMBARAN UMUM.....	4
2.1 Latar Belakang Perusahaan.....	4
2.2 Identitas Perusahaan.....	5
2.3 Visi Dicoding .....	6
2.4 Misi Perusahaan .....	6
BAB III LANDASAN TEORI.....	7
3.1 <i>Software Development Life Cycle</i> .....	7
3.2 <i>Metode Waterfall</i> .....	9
3.3 <i>Grading Makanan dan Minuman</i> .....	9
3.4 Android .....	11

3.5	<i>Kotlin</i>	11
3.6	<i>Application Programming Interface (API)</i>	12
3.7	<i>Retrofit</i>	12
3.8	<i>Blackbox Testing</i>	12
BAB IV DESKRIPSI PEKERJAAN		11
4.1	Alur Sistem	11
4.1.1	<i>Use Case Diagram</i>	11
4.1.2	<i>Activity Diagram</i>	12
4.1.3	<i>Sequence Diagram</i>	19
4.1.4	<i>Class Diagram</i>	21
4.2	<i>Design Wireframe</i>	22
4.2.1	<i>Login Dan Register</i>	22
4.2.2	<i>Home</i>	23
4.2.3	<i>Scan Nutrisi</i>	24
4.2.4	<i>History</i>	25
4.2.5	<i>Tambah Produk</i>	26
4.2.6	<i>Profil</i>	27
4.3	<i>API</i>	28
4.4	Implementasi	32
4.4.1	Halaman Registrasi	32
4.4.2	Halaman <i>Login</i>	33
4.4.3	Halaman <i>Home</i>	35
4.4.4	Halaman Detail Produk	36
4.4.5	Halaman Pencarian Produk	37



UNIVERSITAS  
Dinamika

4.4.6	Halaman Tambah Produk.....	38
4.4.7	Halaman <i>Scan</i> Nutrisi.....	39
4.4.8	Halaman <i>Preview</i> .....	40
4.4.9	Halaman Hasil <i>Scan</i> .....	41
4.4.10	Halaman <i>History</i> .....	42
4.4.11	Halaman Profil .....	43
4.5	Pengujian.....	44
BAB V PENUTUP.....		45
5.1	Kesimpulan .....	45
5.2	Saran .....	45
DAFTAR PUSTAKA .....		46
LAMPIRAN .....		48

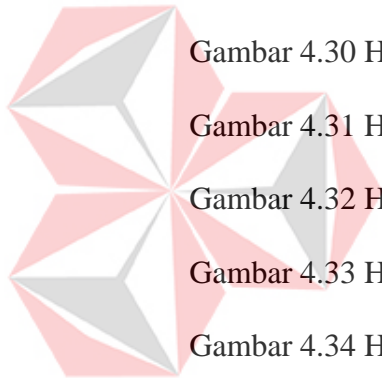


UNIVERSITAS  
**Dinamika**

## DAFTAR GAMBAR

	<b>Halaman</b>
Gambar 2.1 Dicoding Space .....	4
Gambar 2.2 Struktur Organisasi Bangkit <i>Academy</i> .....	5
Gambar 3.1 Tahapdan <i>Software Development Live Cycle</i> .....	8
Gambar 4.1 <i>Use Case Diagram</i> Aplikasi.....	11
Gambar 4.2 <i>Activity Diagram Login dan Register</i> .....	12
Gambar 4.3 Alur Sistem Melihat <i>List</i> Produk dan Detail Produk .....	13
Gambar 4.4 Alur Sistem Cari Produk .....	14
Gambar 4.5 Alur Sistem Tambah Produk .....	15
Gambar 4.6 Alur Sistem <i>Scan</i> Nutrisi.....	16
Gambar 4.7 Alur Sistem Melihat <i>List History</i> dan Detail <i>History</i> .....	17
Gambar 4.8 Alur Sistem Profil .....	18
Gambar 4.9 <i>Sequence Diagram</i> Tambah Produk .....	19
Gambar 4.10 <i>Sequence Diagram Scan</i> Nutrisi .....	20
Gambar 4.11 Class Diagram Aplikasi.....	21
Gambar 4.12 Desain <i>Wireframe Login</i> dan Register .....	22
Gambar 4.13 Desain <i>Wireframe Home</i> .....	23
Gambar 4.14 Desain <i>Wireframe Scan</i> Nutrisi.....	24
Gambar 4.15 Desain <i>Wireframe History</i> .....	25
Gambar 4.16 Desain <i>Wireframe</i> Tambah Produk .....	26
Gambar 4.17 Desain <i>Wireframe</i> Profil .....	27
Gambar 4.18 Alur Implementasi API .....	28
Gambar 4.19 Kode <i>Api Config</i> .....	28

Gambar 4.20 Kode <i>Api Service</i> .....	29
Gambar 4.21 Kode <i>Repository</i> .....	30
Gambar 4.22 Kode <i>Injection</i> .....	31
Gambar 4.23 Kode <i>ViewModel</i> .....	32
Gambar 4.24 Halaman Registrasi .....	33
Gambar 4.25 Halaman <i>Login</i> .....	34
Gambar 4.26 Halaman <i>Home</i> .....	35
Gambar 4.27 Halaman Detail Produk .....	36
Gambar 4.28 Halaman Pencarian Produk .....	37
Gambar 4.29 Halaman Tambah Produk .....	38
Gambar 4.30 Halaman <i>Scan</i> Nutrisi .....	39
Gambar 4.31 Halaman Preview .....	40
Gambar 4.32 Halaman Hasil <i>Scan</i> .....	41
Gambar 4.33 Halaman <i>History</i> .....	42
Gambar 4.34 Halaman Profil .....	43



## DAFTAR TABEL

	<b>Halaman</b>
Tabel 3.1 Aturan Pemberian <i>Grade</i> .....	10
Tabel 3.2 Contoh Perhitungan <i>Grade</i> .....	10
Tabel 4.1 Pengujian <i>Blackbox Testing</i> .....	44



UNIVERSITAS  
**Dinamika**

## DAFTAR LAMPIRAN

	<b>Halaman</b>
Lampiran 1. Surat Balasan Dari Perusahaan.....	48
Lampiran 2. Form KP-5 (MBKM).....	49
Lampiran 3. Form KP-6 dan 7 (MBKM).....	60
Lampiran 4. Kartu Bimbingan KP .....	66
Lampiran 5. Sertifikat MBKM.....	67
Lampiran 6. Biodata Penulis .....	68



UNIVERSITAS  
**Dinamika**



# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Makan dan minum merupakan salah satu kebutuhan primer manusia agar manusia tersebut dapat mempertahankan hidupnya. Namun, di era modern yang semakin mudahnya mencari makanan dan minuman juga mempengaruhi pola hidup menjadi tidak sehat. Tanpa disadari, berbagai penyakit yang dialami manusia sering kali berasal dari makanan yang dikonsumsi sehari-hari. Masyarakat Indonesia umumnya mengonsumsi makanan dengan pola yang tidak teratur atau tidak seimbang, yang semakin memperburuk kondisi kesehatan mereka (Kristiana Pelealu et al., 2021). Kemenkes (2018) mengungkapkan bahwa lebih dari 200 jenis penyakit dapat disebabkan melalui makanan yang dimakan. Menurut Santika dalam databoks (2023), stroke, penyakit jantung, dan diabetes adalah tiga penyakit utama yang menyebabkan kematian tertinggi di Indonesia (Dedy Kasingku & Lumoindong, 2023).

Dalam kehidupan sehari-hari, sering kali bagian yang paling sulit diperhatikan adalah memilih makanan yang sehat untuk dikonsumsi. Mengingat kemajuan teknologi yang sangat pesat saat ini, maka penulis mengembangkan aplikasi yang dapat menentukan *grade* makanan dan minuman berdasarkan tabel nutrisi yang tercantum pada produk tersebut. Aplikasi ini akan menilai makanan dan minuman dengan menggunakan *grade* yang terdiri dari A, B, C, dan D. Jika suatu produk diberi *grade* A, maka produk tersebut dianggap sehat, sementara *grade* D menunjukkan bahwa produk tersebut kurang layak untuk dikonsumsi terus

menerus (Shin et al., 2023). Parameter yang digunakan untuk menentukan *grade* tersebut adalah kandungan gula dan lemak jenuh per gram/100ml.

Tren teknologi modern, khususnya penggunaan aplikasi *mobile* memberikan solusi yang efektif untuk masalah ini. Salah satu sistem operasi *mobile* paling populer saat ini adalah Android, karena sifatnya yang open source, dukungan terhadap berbagai aplikasi melalui Google Play Store, fleksibilitas antarmuka pengguna, kemampuan multitasking, dukungan konektivitas yang luas, integrasi dengan layanan Google, serta pembaruan sistem operasi yang berkelanjutan (Musfiroh et al., 2024). Aplikasi berbasis Android memungkinkan pengguna untuk mengakses informasi secara cepat dan akurat hanya dengan menggunakan perangkat *mobile* mereka. Ini sangat relevan dengan kebutuhan konsumen modern yang cenderung menginginkan informasi instan dan mudah diakses mengenai kualitas dan kandungan nutrisi dari makanan dan minuman yang mereka konsumsi (Adjeng et al., 2023).

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan, maka rumusan masalahnya adalah bagaimana mengembangkan aplikasi Android yang dapat membantu masyarakat dalam memilih makanan dan minuman yang sehat.

## 1.3 Batasan Masalah

Berikut adalah batasan masalah dalam pengembangan aplikasi penentuan *grade* makanan dan minuman ini :

1. Penulis memanfaatkan *algoritma* atau metode *scanning* yang telah ada untuk mengklasifikasikan *grade* makanan dan minuman.

2. Aplikasi dikembangkan hanya untuk perangkat *mobile* dengan sistem operasi Android.

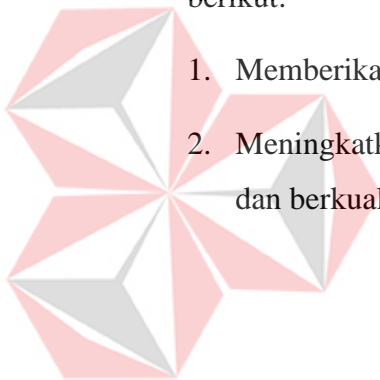
#### **1.4 Tujuan**

Tujuan dari penulis adalah mengembangkan aplikasi *mobile* berbasis Android yang dapat menentukan *grade* makanan dan minuman berdasarkan kandungan nutrisi.

#### **1.5 Manfaat**

Manfaat dari pengembangan aplikasi pada penelitian ini adalah sebagai berikut:

1. Memberikan kemudahan bagi konsumen dalam memilih produk yang sehat
2. Meningkatkan kesadaran akan pentingnya konsumsi makanan yang seimbang dan berkualitas.



UNIVERSITAS  
**Dinamika**

## BAB II

### GAMBARAN UMUM

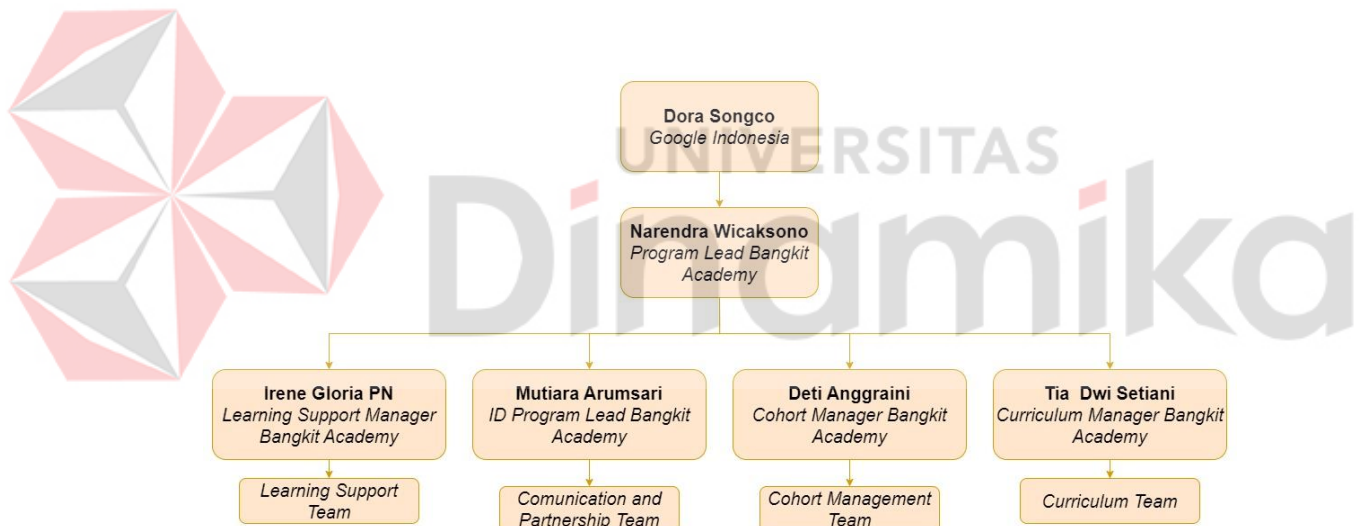
#### 2.1 Latar Belakang Perusahaan



Gambar 2.1 Dicoding Space

Dicoding adalah sebuah platform pendidikan teknologi yang membantu menghasilkan talenta digital berstandar global yang didirikan pada tanggal 5 Januari 2015 oleh Narendra Wicaksono. Dicoding memiliki empat pilar kegiatan utama, yaitu *Academy*, *Event*, *Challenge*, dan *Jobs*. Dicoding menyediakan materi yang selalu diperbarui dan disesuaikan dengan kurikulum industri IT. Keuntungan utama ketika belajar di Dicoding adalah memperoleh *Code Review* dari tugas (*submission*) yang telah dikerjakan, review tersebut dilakukan oleh Tim *Expert* Dicoding. Sebagai bentuk apresiasi, Dicoding memberi hadiah *points* yang bisa ditukarkan dengan berbagai *rewards*, seperti perangkat keras, sesi konsultasi dengan ahli, hingga merchandise. Lulusan *Academy* Dicoding juga dapat peluang lebih besar untuk diterima di perusahaan mitra melalui *Dicoding Jobs*.

Salah satu kegiatan atau program populer di Dicoding adalah program *Bangkit Academy*. *Bangkit* adalah program kesiapan karir yang bertujuan menciptakan talenta untuk perusahaan teknologi dan startup kelas dunia di Indonesia. Program ini didukung penuh oleh Google, GoTo, Tokopedia, dan Traveloka. *Bangkit* menawarkan tiga *track* pembelajaran yaitu: *Mobile Development*, *Machine Learning*, dan *Cloud Computing*. Di program ini, selain belajar tentang teknis teknologi juga terdapat pembelajaran *soft skill* dan bahasa Inggris yang diperlukan untuk bertransisi dari dunia akademis ke dunia kerja dan mencapai kesuksesan di perusahaan terkemuka. Struktur organisasi *Bangkit Academy* dapat dilihat pada Gambar 2.2 dibawah ini :



Gambar 2.2 Struktur Organisasi *Bangkit Academy*

## 2.2 Identitas Perusahaan

Nama Instansi : Yayasan Dicoding Indonesia  
 Alamat : Jl. Batik Kumeli No.50, Sukaluyu, Kec. Cibeunying  
 Kaler, Kota Bandung, Jawa Barat 40123  
 No. Telepon : 0818-485-850

*Website* : <https://dicoding.com/>

*Email* : [info@dicoding.com](mailto:info@dicoding.com)

### **2.3 Visi Dicoding**

Visi Dicoding adalah menjadi platform edukasi teknologi terdepan yang mendorong akses literasi digital yang lebih luas untuk semua. Dicoding memiliki misi untuk mengakselerasi transisi Indonesia menuju dunia digital melalui pendidikan teknologi yang mentransformasi kehidupan.

### **2.4 Misi Perusahaan**

Misi Dicoding adalah mengakselerasi transisi Indonesia menuju dunia digital melalui pendidikan teknologi yang mentransformasi kehidupan. Kini semua bangsa bergerak menuju dunia digital yang bertumpu pada inovasi teknologi di semua sendi kehidupan. Kami yakin pendidikan teknologi adalah fondasi bagi setiap bangsa agar menjadi yang terdepan dalam menghadapi dunia digital. Dicoding hadir sebagai platform pendidikan teknologi yang membantu menghasilkan talenta digital berstandar global. Semua demi mengakselerasi Indonesia agar menjadi yang terdepan.

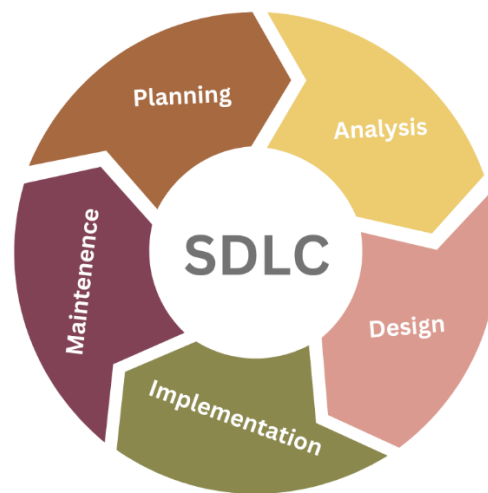
## **BAB III**

### **LANDASAN TEORI**

#### **3.1 *Software Development Life Cycle***

*Software Development Life Cycle* (SDLC) merupakan gambaran tahapan dalam proses pengembangan sistem. SDLC menyajikan metode terstruktur untuk menciptakan sebuah sistem, dengan pendekatan klasik yang sistematis dan berurutan dalam pembuatan perangkat lunak. Menurut Richard Scroggins (2014) tujuan utama SDLC adalah memastikan terpenuhinya kebutuhan pengguna terkait sistem yang sedang dikembangkan. Kebutuhan ini dapat mencakup modifikasi aplikasi yang sudah ada atau pembuatan aplikasi baru, baik secara bertahap maupun melalui instalasi lengkap.

SDLC juga memungkinkan pengembang untuk mengestimasi durasi penggunaan perangkat lunak yang dihasilkan atau diimplementasikan. Dengan demikian, SDLC tidak hanya berfokus pada proses pengembangan, tetapi juga mempertimbangkan aspek keberlanjutan dan masa pakai produk akhir (Silitonga & Purba, 2021). Menurut (Nukman et al., 2022) tahapan *Software Development Live Cycle* (SDLC) ada 5 proses yaitu: *planning, analysis, design, implementation, dan maintenance*.



Gambar 3.1 Tahapan *Software Development Live Cycle*

Sumber : <https://www.dicoding.com/blog/metode-sdlc/>

#### 1. *Planning*

Tahap ini berfokus pada identifikasi sistem yang akan dikembangkan dan penetapan tujuan yang ingin dicapai.

#### 2. *Analysis*

Pada tahap ini, dilakukan penelitian terhadap sistem yang sudah ada dan studi literatur untuk menentukan kasus yang dapat ditangani oleh sistem baru. Juga dilakukan identifikasi terhadap sistem yang telah dikembangkan sebelumnya.

#### 3. *Design*

Tahap ini melibatkan penentuan proses dan teknik yang akan digunakan untuk mengimplementasikan sistem baru berdasarkan pengembangan dari sistem sebelumnya.

#### 4. *Implementasi*

Pada tahap ini, rancangan sistem yang telah dibuat diimplementasikan. Selain itu, dilakukan juga pengujian terhadap sistem yang baru dikembangkan.



## 5. *Maintenance*

Tahap terakhir merupakan proses pemeliharaan sistem untuk memastikan sistem tetap berfungsi dengan baik dan teratur.

### 3.2 **Metode *Waterfall***

Metode *waterfall* adalah metode pengembangan sistem yang melibatkan penyelesaian setiap fase secara berurutan dan terstruktur. Dalam penerapan metode *Waterfall*, setiap langkah harus diselesaikan sepenuhnya mulai dari tahap pertama sebelum melanjutkan ke tahap berikutnya (Fachri & Surbakti, 2021). Metode ini mencakup beberapa tahap penting dalam pengembangan perangkat lunak, yaitu analisis, desain, pengkodean, dan pengujian (Akmal Hidayat et al., 2023).

Keuntungan utama metode *Waterfall* adalah strukturnya yang jelas dan mudah dikelola, serta dokumentasi yang terorganisir dengan baik pada setiap tahap. Namun, metode ini juga memiliki keterbatasan, terutama dalam hal fleksibilitas. Karena sifatnya yang linear, sulit untuk kembali ke tahap sebelumnya jika ditemukan masalah atau perubahan kebutuhan di tengah proses pengembangan. Hal ini dapat menyebabkan peningkatan biaya dan waktu jika perubahan signifikan diperlukan pada tahap lanjut proyek (Suherni, 2023).

### 3.3 ***Grading Makanan dan Minuman***

*Grading* makanan dan minuman atau sistem *Nutri-Grade* adalah sistem penilaian yang diterapkan untuk menilai kandungan gizi pada makanan dan minuman dengan tujuan memberikan informasi yang jelas dan mudah dipahami oleh konsumen. Sistem ini membantu konsumen membuat pilihan yang lebih sehat dengan mengkategorikan empat tingkat dari A (paling sehat) hingga D (paling tidak

sehat), yang terutama didasarkan pada kadar gula dan lemak jenuh per 100 ml minuman. *Nutri-Grade* juga mendorong konsumen untuk meningkatkan kualitas kesehatan dengan mengurangi konsumsi gula berlebih, terutama dari minuman manis (Shin et al., 2023). Aturan dan cara perhitungan *grading* dapat dilihat pada Tabel 3.1 dibawah ini :

Tabel 3.1 Aturan Pemberian *Grade*

<i>Grade</i>	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
Gula (gram/100ml)	$\leq 1$	1 - 5	5 - 10	$> 10$
Lemak Jenuh (gram/100ml)	$\leq 0.7$	0.7 – 1.2	1.2 – 2.8	$> 2.8$
Penggunaan pengganti gula	Tidak boleh mengandung pengganti gula	Mungkin mengandung pengganti gula yang diizinkan		

Tabel 3.2 Contoh Perhitungan *Grade*

No.	<i>Grade</i> Gula	<i>Grade</i> Lemak Jenuh	Pengganti Gula	<i>Grade</i>
1	B	C	No	<b>C</b>
2	C	B	No	<b>C</b>
3	A	A	Yes	<b>B</b>
4	A	A	No	<b>A</b>

Berdasarkan Tabel 3.2 diatas jika suatu produk memiliki gula di *grade* B dan lemak jenuh di *grade* C dan tidak menggunakan gula pengganti maka hasil *grade* nya adalah C (Produk 1), diambil berdasarkan nilai *grade* terendah dalam produk tersebut. Jika suatu produk memiliki nilai *grade* A pada gula dan lemak jenuh tapi menggunakan gula pengganti, maka hasil *grade* nya adalah B (Produk

3). Tapi jika tidak menggunakan gula pengganti maka hasil *grade* nya adalah A (Produk 4).

### 3.4 Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* yang berbasis *Linux*. Sistem ini tersedia secara *open source* artinya dapat digunakan secara bebas oleh siapa saja (Yussandi, 2021). Dengan keunggulan tersebut para pengembang atau *developer* Android dapat menciptakan berbagai aplikasi *mobile* yang inovatif dan beragam. Saat penulisan ini, Android telah merilis versi 15 *beta*, menunjukkan komitmen mereka untuk terus maju dan berkembang dalam menyediakan sistem operasi yang lebih baik.

### 3.5 Kotlin

*Kotlin* adalah bahasa pemrograman *open source* yang mendukung pemrograman berorientasi objek dan fungsional. Bahasa ini memiliki sintaks dan konsep yang mirip dengan bahasa lain seperti *C#*, *Java*, dan *Scala*. Awal terciptanya *Kotlin* dimulai ketika seorang *Lead Developer* bernama Dmitry Jemerov tidak menemukan beberapa fitur yang diinginkan di bahasa *Java*. Sehingga, ia mengembangkan bahasa pemrogramannya sendiri yang disebut *Kotlin*, bahasa pemrograman yang menggabungkan semua fitur modern dan bisa berjalan di *Java Virtual Machine* (JVM), serta memiliki kecepatan kompilasi setara dengan *Java*.

*Kotlin* sekarang dikelola oleh Kotlin Foundation, sebuah organisasi yang didirikan oleh JetBrains dan Google, yang bertanggung jawab atas pengembangan bahasa tersebut. *Kotlin* secara resmi didukung oleh *Google* untuk pengembangan

Android, yang berarti dokumentasi dan fitur-fitur Android dirancang dengan mempertimbangkan *Kotlin*.

### 3.6 *Application Programming Interface (API)*

*Application Programming Interface (API)* adalah antarmuka yang dikembangkan oleh pengembang yang memungkinkan beberapa atau semua fungsi sistem dapat diakses secara terprogram (Hasanuddin et al., 2022). API merupakan sekumpulan teknik yang jelas untuk menciptakan komunikasi antara berbagai komponen perangkat lunak. Fungsi utama API adalah mempermudah penggunaan teknologi tertentu saat membangun perangkat lunak atau aplikasi bagi pengembang (Agustin Muris et al., 2023)

### 3.7 *Retrofit*

*Retrofit* adalah library REST yang aman untuk Android, *Java*, dan *Kotlin* yang dikembangkan oleh *Square*. Dengan *Retrofit* membuat akses ke *framework* menjadi kuat yang membantu dalam mengautentikasi dan berinteraksi dengan API, serta mengirimkan permintaan jaringan menggunakan *OkHttp*. *Library* ini mempermudah pengunduhan data JSON atau XML dari API. Di *Retrofit*, setelah data diunduh, data tersebut diuraikan menjadi *Plain Old Java Objects (POJO)*.

### 3.8 *Blackbox Testing*

Pengujian atau testing adalah langkah penting yang bertujuan untuk mendeteksi kesalahan dalam sistem yang dibangun dan mengurangi potensi kerugian akibat kesalahan-kesalahan tersebut (Praniffa et al., 2023). Salah satu jenis pengujian adalah *Blackbox Testing*. *Blackbox Testing* adalah metode pengujian perangkat lunak yang mengevaluasi fungsi aplikasi (*Functional Testing*) tanpa

mengetahui struktur internal atau kinerja aplikasi tersebut (Fahrezi et al., 2023). Dengan kata lain penguji tidak memerlukan pengetahuan tentang kode program dari sistem yang diuji. Dalam implementasinya *blackbox testing* juga memiliki kelebihan dan kekurangan. Kelebihannya adalah membantu menemukan aspek-aspek yang tidak memenuhi spesifikasi kebutuhan dalam pengembangan perangkat lunak. Sedangkan, kekurangan dari *blackbox testing* adalah pengujian tidak dapat dilakukan secara menyeluruh karena pengetahuan penguji terbatas mengenai perangkat lunak yang diuji(Praniffa et al., 2023)



UNIVERSITAS  
Dinamika

## BAB IV

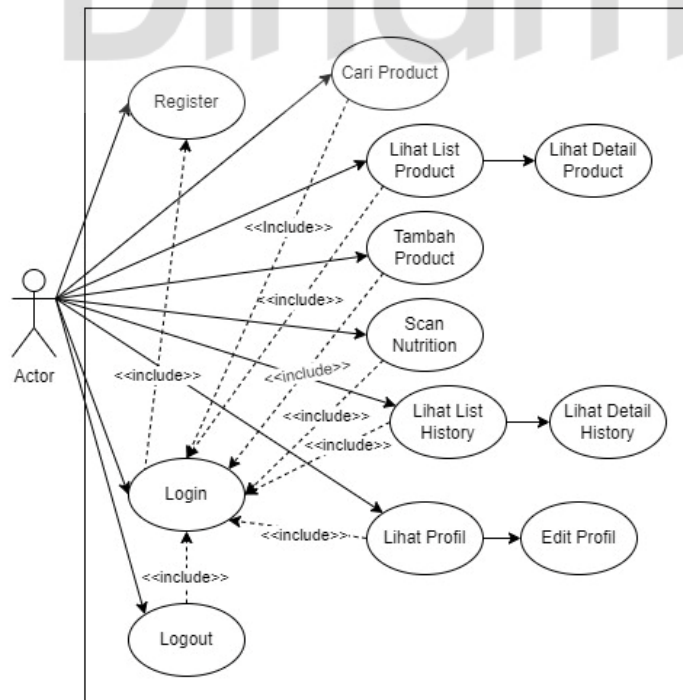
### DESKRIPSI PEKERJAAN

#### 4.1 Alur Sistem

Alur sistem adalah rangkaian langkah-langkah dan proses yang terjadi dalam aplikasi untuk mencapai tujuan tertentu. Berikut adalah alur sistem yang diterapkan dalam aplikasi ini.

##### 4.1.1 Use Case Diagram

*Use Case Diagram* mengilustrasikan bagaimana pengguna (aktor) berinteraksi dengan sistem aplikasi. Diagram ini memvisualisasikan fitur-fitur utama yang ada dalam aplikasi dan cara menggunakannya. Diagram dapat dilihat pada Gambar 4.1 dibawah ini :

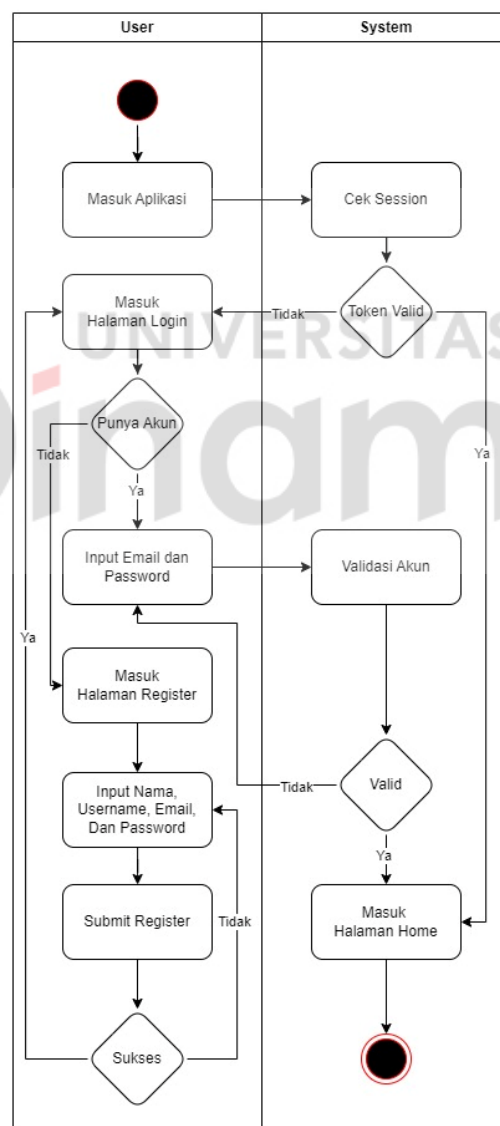


Gambar 4.1 *Use Case Diagram* Aplikasi

### 4.1.2 Activity Diagram

#### A. Login dan Register

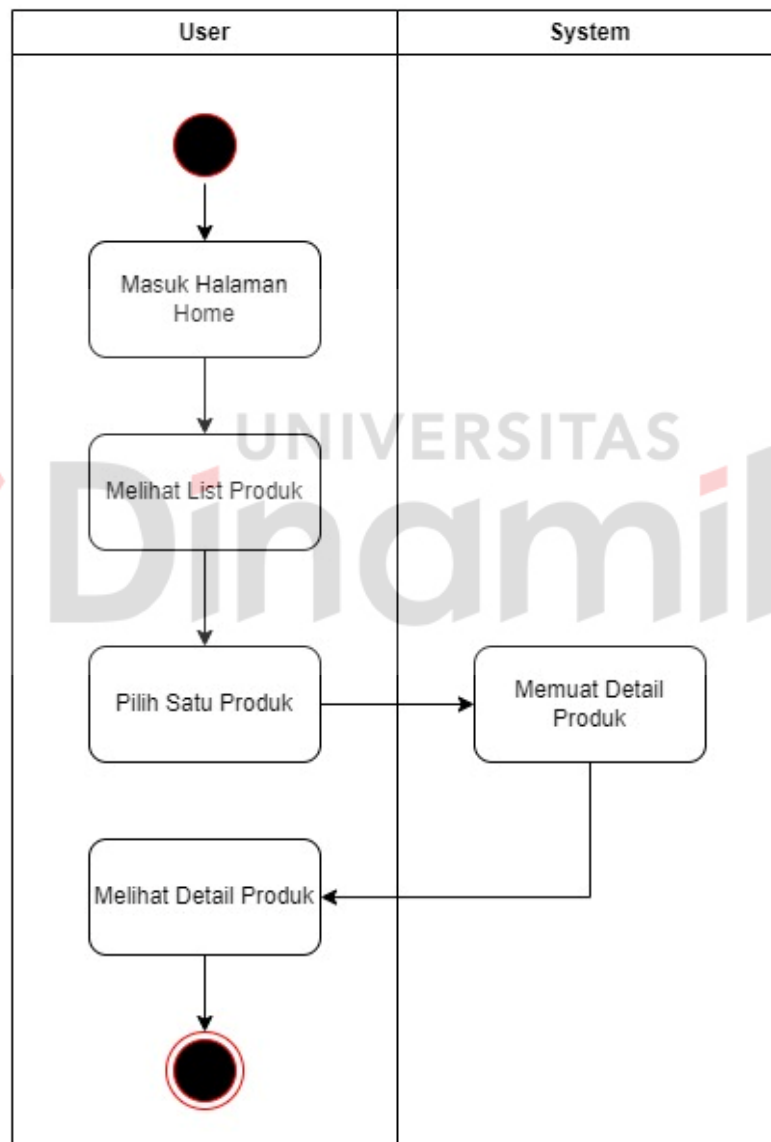
Alur sistem mengecek sesi *login*. Jika ada, pengguna masuk *Home*. Jika tidak, pengguna harus *login*. Jika *login* berhasil akan menuju *Home*. Jika gagal maka harus mendaftar, setelah berhasil pengguna dapat *login*. Gambaran alur sistem *login* dan *register* dapat dilihat pada Gambar 4.2 dibawah ini :



Gambar 4.2 Activity Diagram Login dan Register

## B. Melihat *List* Produk dan Detail Produk

Pada saat setelah *Login* secara otomatis pengguna akan masuk ke halaman *Home*. Pada saat itu juga pengguna akan melihat *list* produk. Dan pengguna juga dapat memilih produk salah satu produk untuk melihat detailnya. Gambaran alur sistemnya dapat dilihat pada Gambar 4.3 dibawah ini :

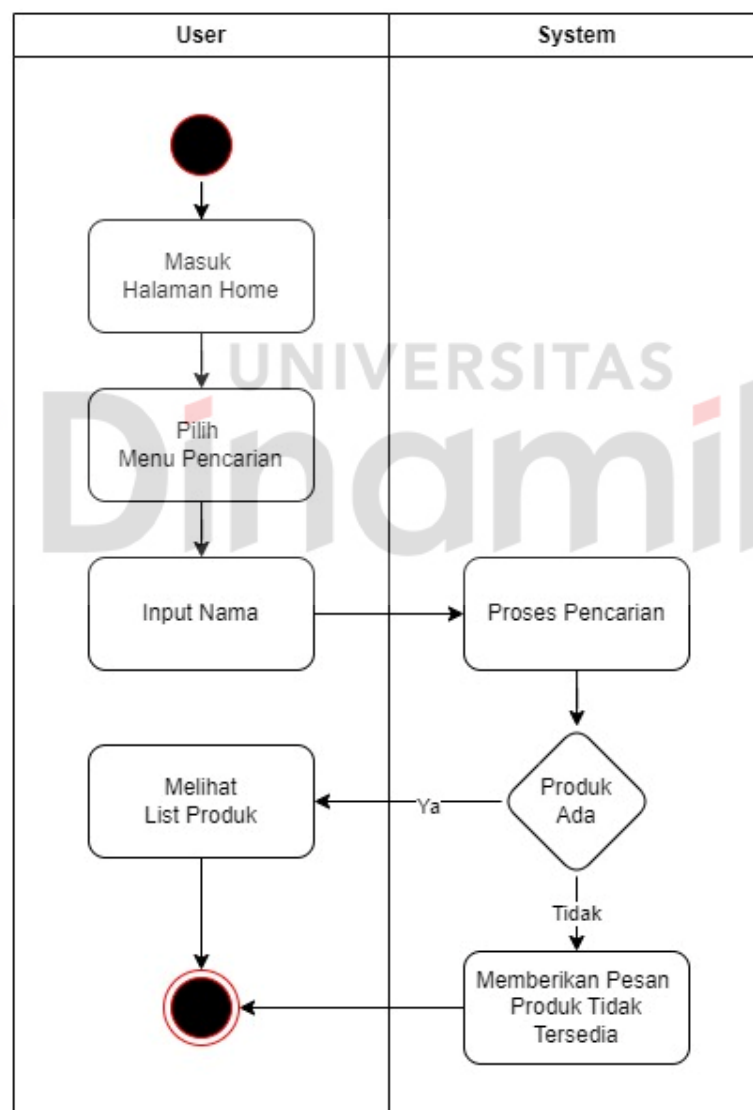


Gambar 4.3 Alur Sistem Melihat *List* Produk dan Detail Produk



### C. Cari Produk

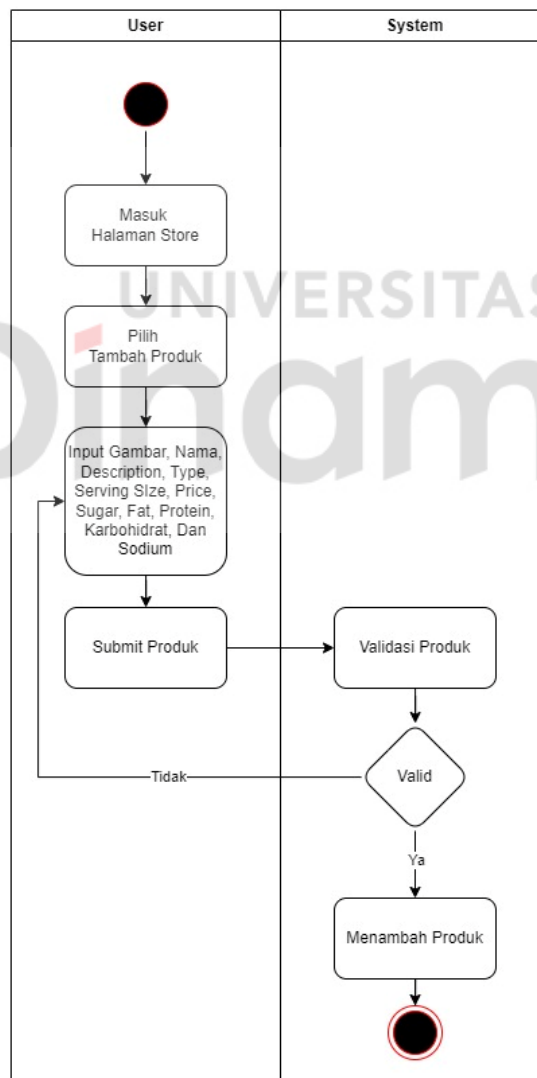
Pertama kali pengguna harus memulai memilih pencarian. Kemudian menginputkan nama produk yang ingin dicari. Setelah itu, sistem akan melakukan cek apakah produk ada, jika ada maka produk akan ditampilkan. Jika tidak, maka akan memberikan pesan produk tidak ada kepada pengguna. Gambaran alur sistem untuk mencari produk dapat dilihat pada Gambar 4.4 dibawah ini :



Gambar 4.4 Alur Sistem Cari Produk

#### D. Tambah Produk

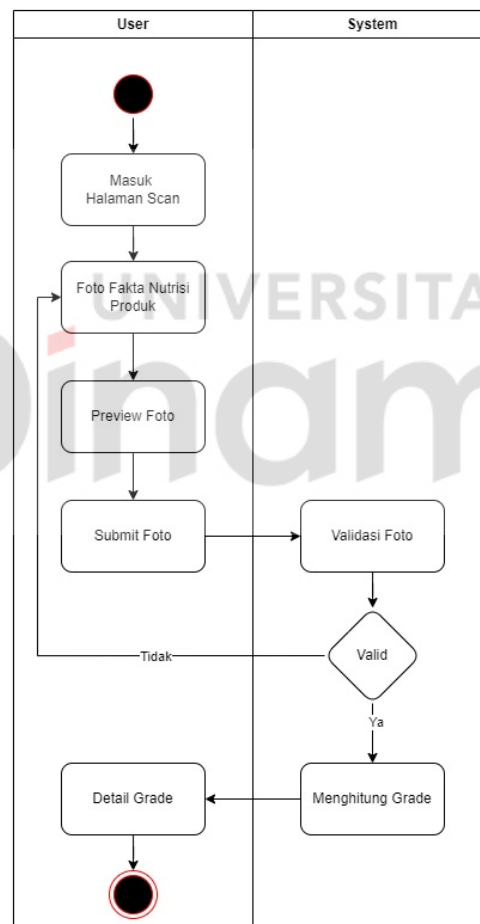
Hal pertama adalah masuk halaman *Store* kemudian menambah produk dengan menginput data seperti gambar, nama, deskripsi, tipe, harga, kandungan gula, garam, *protein* dan *sodium*. Kemudian, sistem akan melakukan cek validasi, jika produk tidak valid maka pengguna harus melengkapi produk. Jika valid, maka produk akan ditambahkan. Alur sistem tambah produk dapat dilihat pada Gambar 4.5 dibawah ini :



Gambar 4.5 Alur Sistem Tambah Produk

### E. *Scan Nutrisi*

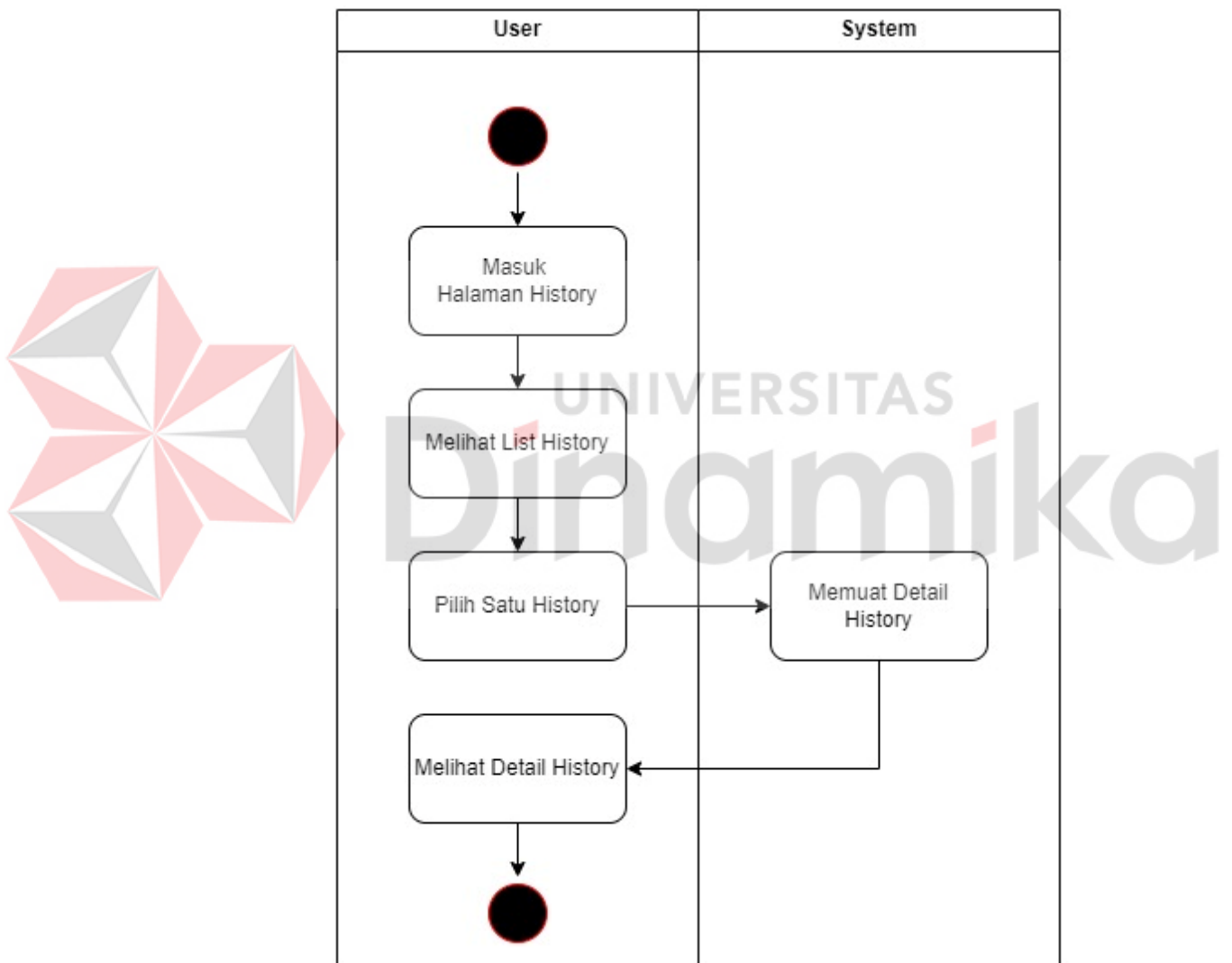
Pertama adalah masuk menu *Scan*, kemudian secara otomatis system akan memulai kamera. Pengguna dapat memfoto fakta nutrisi produk dan kemudian melihat hasil fotonya. Jika foto telah jelas maka dapat dilakukan *submit* proses dan sistem akan melakukan perhitungan *grade*. Setelah perhitungan oleh *system* selesai, maka pengguna akan diarahkan ke halaman detail *grade* untuk melihat hasil *Scan* nya. Alur sistem *scan* nutrisi dapat dilihat pada Gambar 4.6 dibawah :



Gambar 4.6 Alur Sistem *Scan* Nutrisi

#### F. Melihat *List History* dan Detail *History*

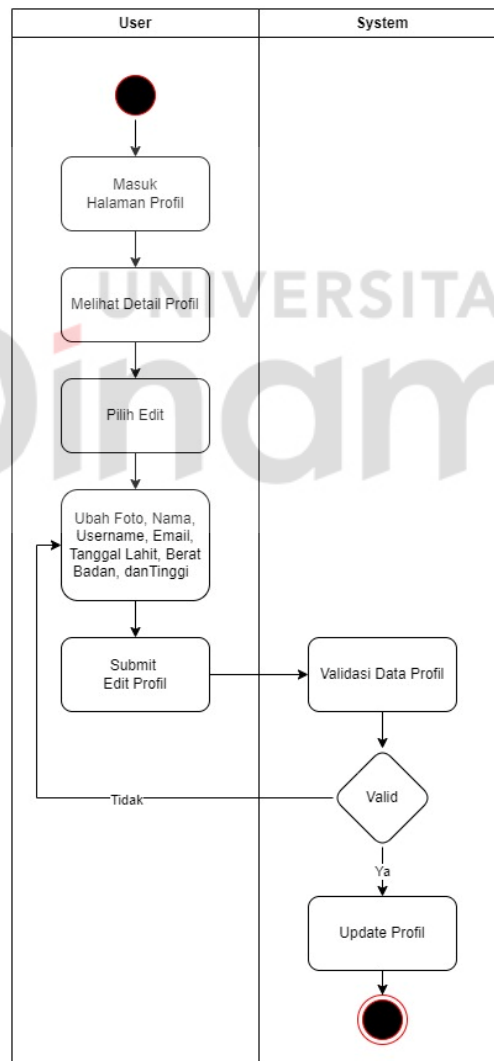
Pertama adalah masuk ke menu *History*, pada *menu* ini pengguna akan melihat *list History Scan*. Pengguna juga dapat memilih salah satu *History* untuk melihat detailnya. Alur Melihat *List History* dan Detail *History* dapat dilihat pada Gambar 4.7 dibawah ini:



Gambar 4.7 Alur Sistem Melihat *List History* dan Detail *History*

## G. Profil

Pertama adalah masuk menu Profil, pada menu ini pengguna akan melihat detail profil. Jika pengguna ingin mengubah profil maka dapat melakukan *action edit* dan memperbarui data seperti foto, nama, email, tinggi, dan berat badan. Maka selanjutnya adalah submit atau simpan perubahan data. Dengan begitu data pengguna akan diperbarui. Alur sistem Profil dapat dilihat pada gambar 4.8 dibawah :

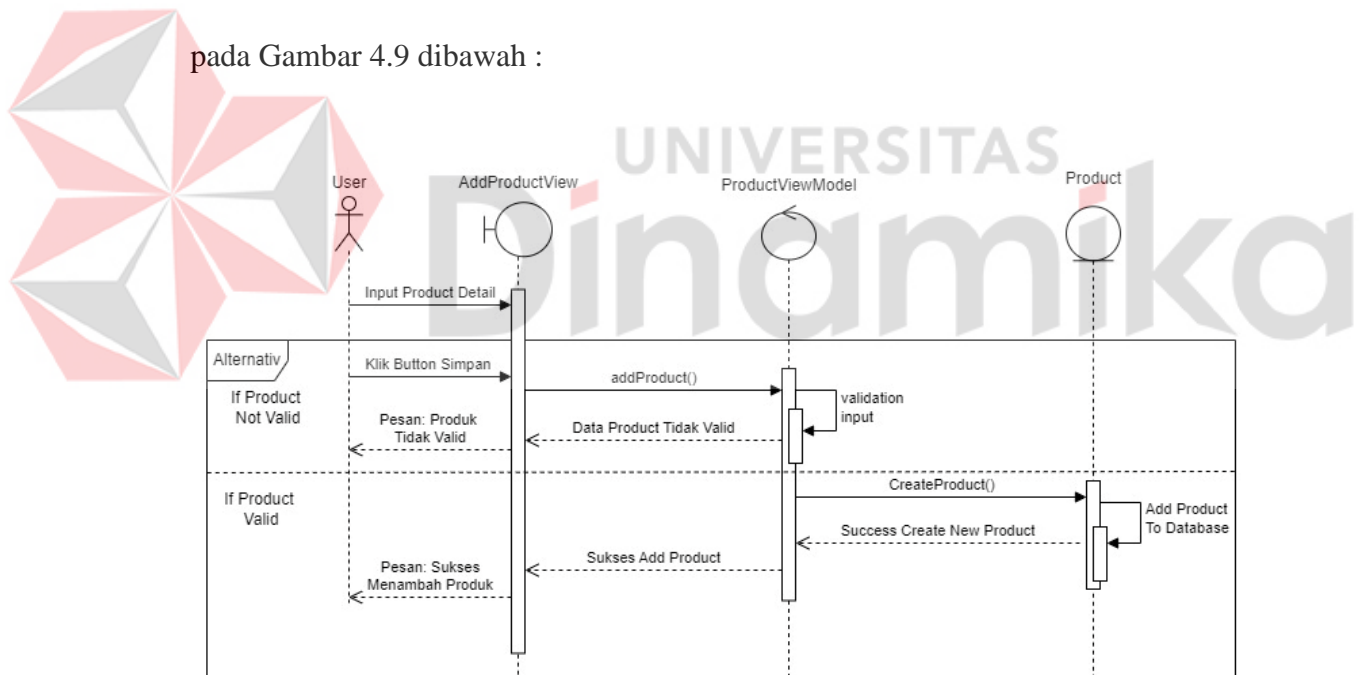


Gambar 4.8 Alur Sistem Profil

### 4.1.3 Sequence Diagram

#### A. Tambah Produk

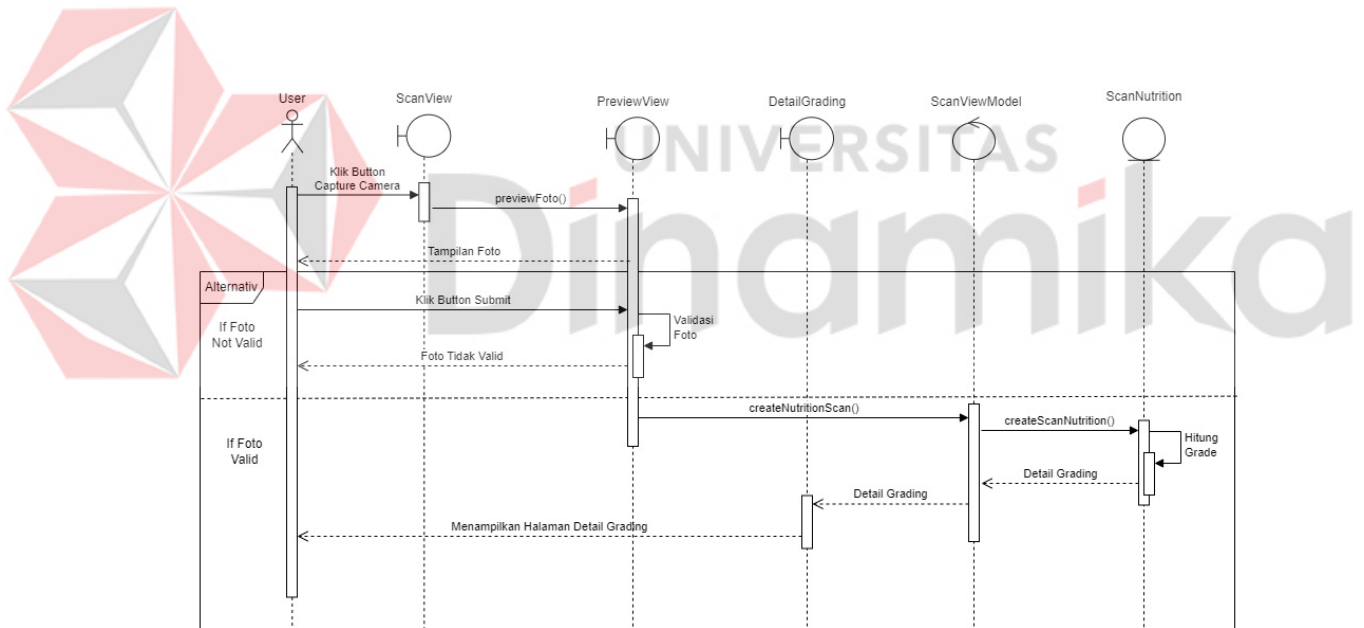
*Sequence Diagram* Tambah Produk digunakan untuk menggambarkan proses pengguna yang sedang melakukan penambahan produk. Dimulai dengan pengguna menginputkan detail produk dan melakukan *submit* tambah produk di *AddProductView*. Kemudian, system akan melakukan validasi *inputan* detail produk dari pengguna di *ProductViewModel*, jika ada yang salah maka system akan memberikan pesan produk tidak valid. Jika product valid maka, *system* akan membuat produk baru sesuai *inputan* pengguna dan memberikan pesan sukses membuat produk baru. Gambaran *Sequence Diagram* Tambah Produk dapat dilihat pada Gambar 4.9 dibawah :



Gambar 4.9 *Sequence Diagram* Tambah Produk

## B. Scan Nutrisi

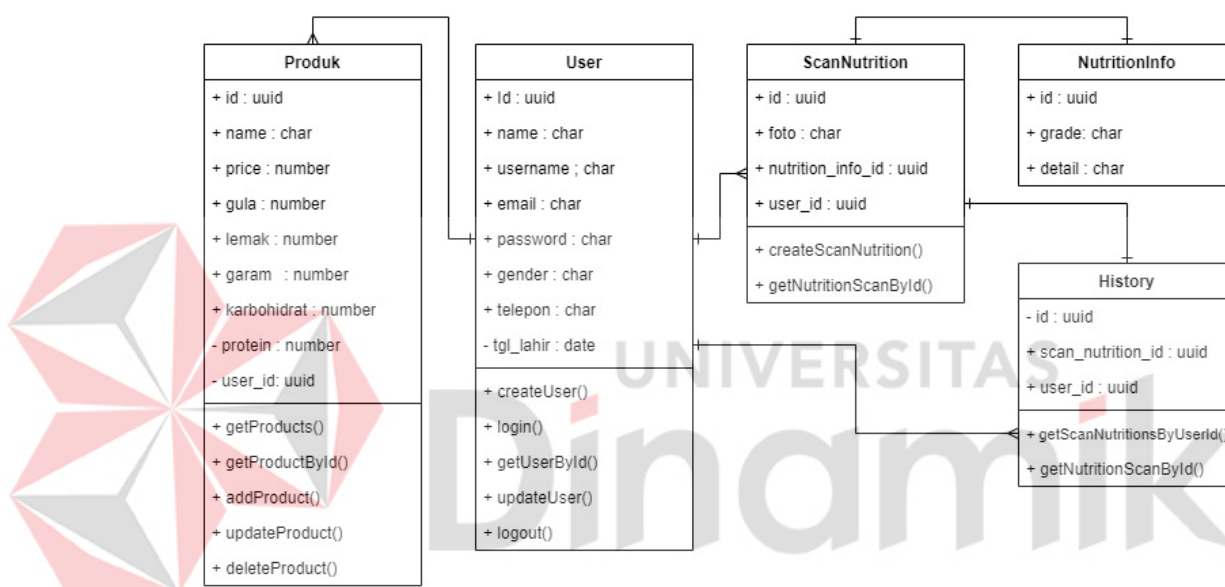
*Sequence Diagram Scan Nutrisi* menggambarkan proses pengguna untuk mengetahui *grade* dari suatu produk. Dimulai dengan pengguna mengambil foto di *ScanView*, kemudian melihat hasil fotonya di *PreviewView*. Jika foto tidak jelas maka dapat melakukan *recapture* foto, jika foto jelas dan valid maka pengguna dapat melakukan proses perhitungan dengan menekan tombol *submit*. *System* akan mulai memproses gambar dan menghitung *grade* di *ScanViewModel*. Setelah proses perhitungan selesai maka pengguna di arahkan ke *DetailGradingView* untuk melihat hasil detail *grade* dari produk yang telah difoto. Gambaran *sequence Scan Nutrisi* dapat dilihat pada Gambar 4.10 dibawah :



Gambar 4.10 *Sequence Diagram Scan Nutrisi*

#### 4.1.4 Class Diagram

*Class* diagram menggambarkan struktur dan hubungan antar class dalam sistem aplikasi. Setiap kelas dalam diagram ini memiliki dua komponen utama yaitu atribut sebagai properti kelas, dan metode sebagai perilaku atau fungsi yang dapat dilakukan oleh *class* tersebut. Dalam pengembangan aplikasi ini terdapat beberapa *class* diagram seperti pada Gambar 4.11 dibawah :



Gambar 4.11 Class Diagram Aplikasi

*Class Product* merupakan representasi dari produk ditambahkan oleh pengguna. *Class User* merepresentasikan pengguna aplikasi. *Class Scan Nutrition* digunakan untuk proses *scan* pada suatu produk untuk mendapatkan hasil *grade*. *Class Nutrition Info* berfungsi untuk menyimpan informasi gizi yang didapatkan dari proses *scan*. *Class History* digunakan untuk mencatat riwayat *scan* yang dilakukan oleh pengguna.

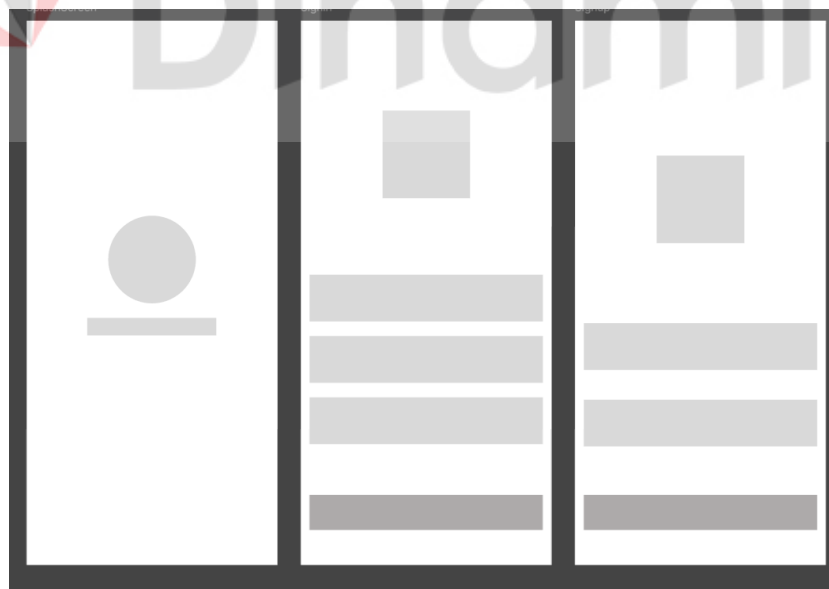


## 4.2 *Design Wireframe*

Sebelum membuat antarmuka pengguna aplikasi Android, tahap awal yang dilakukan adalah merancang *Wireframe*. *Wireframe* merupakan sketsa *visual* sederhana yang menggambarkan tata letak dan alur antarmuka pengguna aplikasi, tujuannya adalah untuk memvisualisasikan konsep desain antarmuka sebelum dilakukan implementasi sebenarnya.

### 4.2.1 *Login Dan Register*

Halaman *Login* dan *Register* digunakan sebagai autentikasi sebelum pengguna masuk dan menggunakan fitur aplikasi. Gambaran halaman *Login* dan *Register* dapat dilihat pada gambar 4.12 dibawah ini. Pada gambar kiri adalah halaman *Splashscreen*, gambar tengah adalah halaman *Registrasi*, dan gambar kanan adalah halaman *Login*.



Gambar 4.12 Desain *Wireframe Login* dan *Register*

#### 4.2.2 *Home*

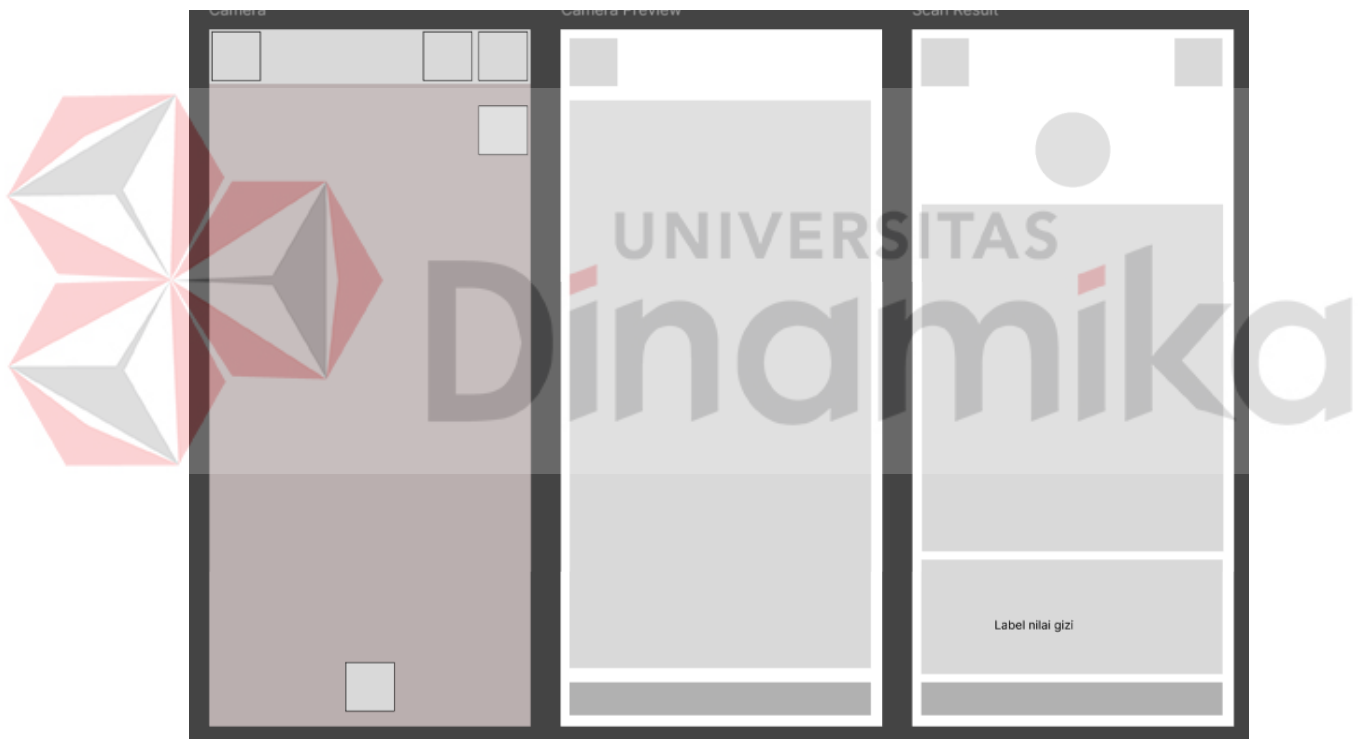
Halaman *Home* merupakan halaman yang menampilkan produk-produk. Gambaran halaman *Home* dapat dilihat pada Gambar 4.13 dibawah ini. Pada gambar kiri merupakan halaman yang berisi *list* produk. Pada gambar kanan merupakan halaman detail dari suatu produk :



Gambar 4.13 Desain *Wireframe Home*

### 4.2.3 Scan Nutrisi

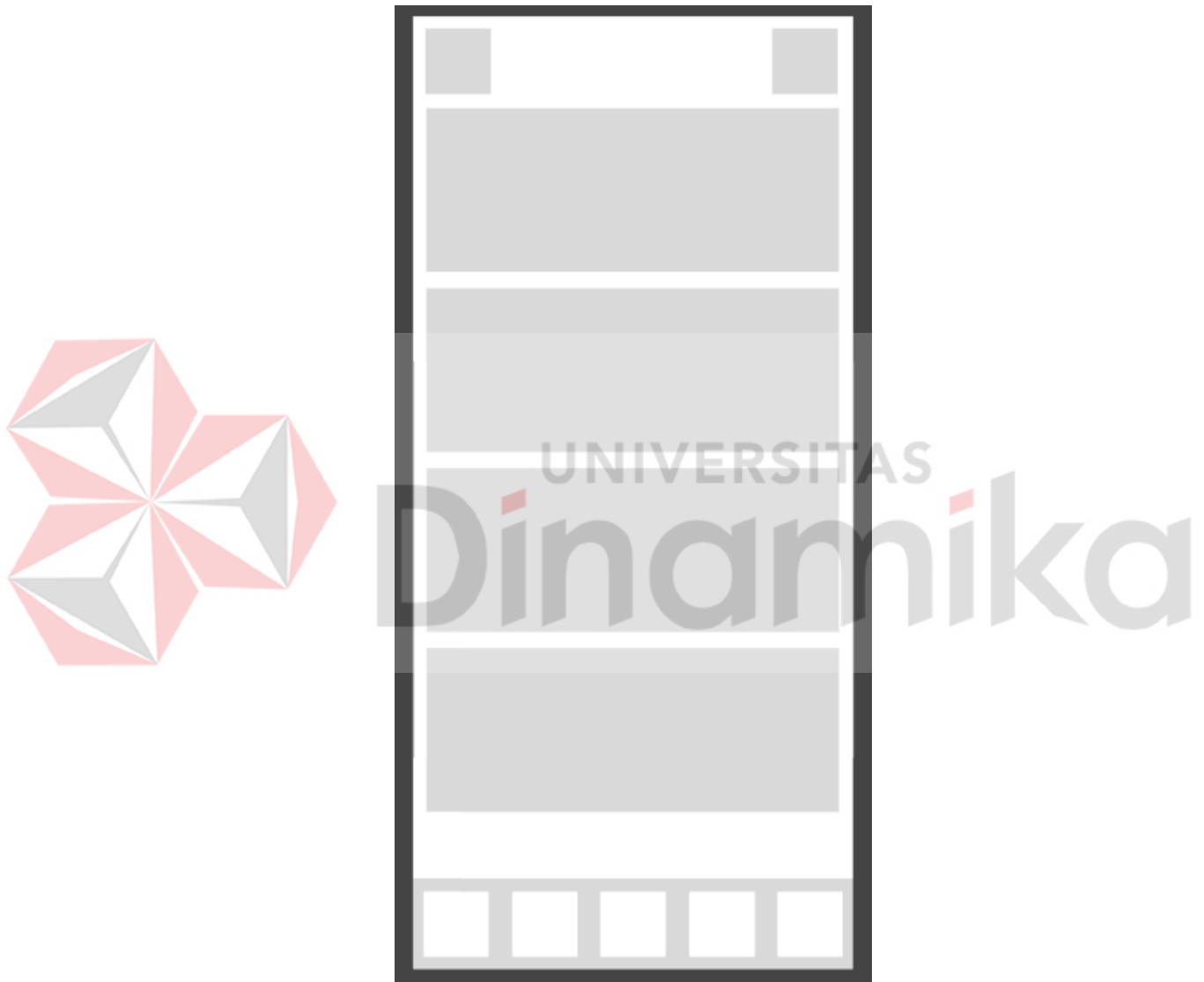
Halaman *Scan Nutrisi* merupakan halaman untuk melakukan cek *grading* pada suatu produk. Gambaran halaman *Scan Nutrisi* dapat dilihat pada Gambar 4.14 dibawah ini. Pada gambar kiri adalah sebuah kamera untuk melakukan foto label nutrisi pada sebuah produk makanan atau minuman. Pada gambar tengah merupakan *preview* hasil foto. Dan gambar paling kanan adalah hasil detail *grading* dari produk makanan atau minuman yang telah dilakukan proses perhitungan oleh *sistem*.



Gambar 4.14 Desain *Wireframe Scan Nutrisi*

#### 4.2.4 *History*


Halaman *History* merupakan halaman yang digunakan untuk melihat hasil *Scan* sebelumnya. Gambaran halaman *History* dapat dilihat pada Gambar 4.15 dibawah ini :



Gambar 4.15 Desain *Wireframe History*

#### 4.2.5 Tambah Produk

Halaman Tambah Produk merupakan halaman yang digunakan untuk menambah sebuah produk. Gambaran halaman Tambah Produk dapat dilihat pada Gambar 4.16 dibawah ini :



UNIVERSITAS  
Dinamika

nama	
type	
deskripsi	
jumlah sajian per kemasan	
Jumlah per sajian	
Lemak	Karbohidrat
Gula	Garam

Gambar 4.16 Desain *Wireframe* Tambah Produk

#### 4.2.6 Profil

Halaman Profil merupakan halaman yang digunakan untuk melihat detail informasi pengguna seperti nama, *email*, foto profil, tanggal lahir dan lain-lain. Selain itu, pengguna juga dapat mengubah data profil dengan cara melakukan edit.

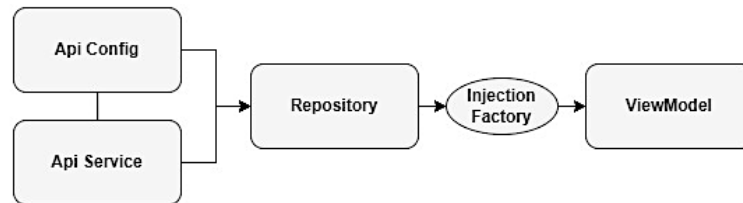
Gambaran halaman Profil dapat dilihat pada Gambar 4.17 dibawah ini :



Gambar 4.17 Desain *Wireframe* Profil

### 4.3 API

Gambaran alur implementasi *Retrofit* dalam pengembangan aplikasi ini dapat dilihat pada gambar 4.18 dibawah ini :



Gambar 4.18 Alur Implementasi API

#### A. *Api Config*

*Api Config* digunakan untuk membuat *instance Retrofit* dan *OkHttp* dan beberapa *interceptor* untuk memodifikasi *header* dan *logging* untuk *debugging*.

Gambaran kode *Api Config* dapat dilihat pada Gambar 4.19 dibawah ini :

```

object ApiConfig {
    fun getApiService(userPreference: UserPreference): ApiService {
        val loggingInterceptor = if (BuildConfig.DEBUG) {
            HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)
        } else {
            HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.NONE)
        }

        val authInterceptor = Interceptor { chain ->
            val req = chain.request()
            val token = runBlocking { userPreference.getToken().first() }
            val requestHeader = req.newBuilder()
                .addHeader( name: "Authorization", value: "Bearer $token" )
                .build()
            chain.proceed(requestHeader) ^ Interceptor
        }

        val client = OkHttpClient.Builder()
            .connectTimeout( timeout: 30, TimeUnit.SECONDS )
            .readTimeout( timeout: 30, TimeUnit.SECONDS )
            .writeTimeout( timeout: 30, TimeUnit.SECONDS )
            .addInterceptor(loggingInterceptor)
            .addInterceptor(authInterceptor)
            .build()

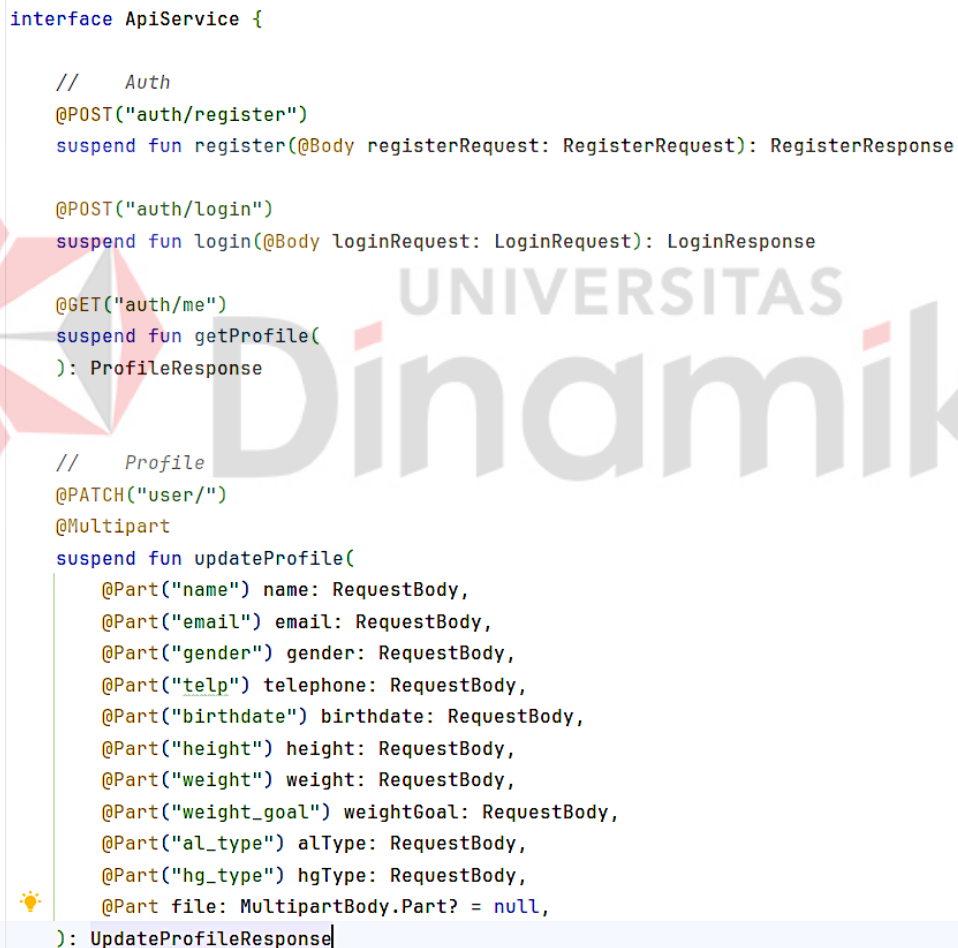
        val retrofit = Retrofit.Builder() Retrofit.Builder
            .baseUrl(BuildConfig.BASE_URL) Retrofit.Builder
            .addConverterFactory(GsonConverterFactory.create())
            .client(client)
            .build()

        return retrofit.create(ApiService::class.java)
    }
}
  
```

Gambar 4.19 Kode *Api Config*

## B. *Api Service*

*Api Service* digunakan untuk mendefinisikan *endpoint-endpoint* atau *url* API. Di dalam *Api service* perlu menentukan tipe metode HTTP yang digunakan dalam pemanggilan API seperti *GET*, *POST*, *DELETE* atau *PUT* serta parameter-parameter yang diperlukan. Selain itu juga harus membuat data *class Response* kembalian dari hasil API yang telah dipanggil. Kode *Api Service* dapat dilihat pada Gambar 4.20 dibawah :



```
interface ApiService {

    // Auth
    @POST("auth/register")
    suspend fun register(@Body registerRequest: RegisterRequest): RegisterResponse

    @POST("auth/login")
    suspend fun login(@Body loginRequest: LoginRequest): LoginResponse

    @GET("auth/me")
    suspend fun getProfile(
    ): ProfileResponse

    // Profile
    @PATCH("user/")
    @Multipart
    suspend fun updateProfile(
        @Part("name") name: RequestBody,
        @Part("email") email: RequestBody,
        @Part("gender") gender: RequestBody,
        @Part("telp") telephone: RequestBody,
        @Part("birthdate") birthdate: RequestBody,
        @Part("height") height: RequestBody,
        @Part("weight") weight: RequestBody,
        @Part("weight_goal") weightGoal: RequestBody,
        @Part("al_type") alType: RequestBody,
        @Part("hg_type") hgType: RequestBody,
        @Part file: MultipartBody.Part? = null,
    ): UpdateProfileResponse
}
```

Gambar 4.20 Kode *Api Service*



### C. Repository

*Repository* merupakan *class* penghubung antara *ViewModel* dan *Api Service*. *Repository* digunakan untuk pemanggilan *function-function* secara *asynchronous* yang telah didefinisikan di *Api Service* yang ditandai dengan kata *suspend* diawal *functionnya*. Kemudian melakukan proses penyimpanan data dari hasil *Response* API ke dalam sebuah variabel *livedata*. Kode *Repository* dapat dilihat pada Gambar 4.21 dibawah :



```
class UserRepository private constructor(
    private val userPreference: UserPreference,
    private val apiService: ApiService,
) {
    private val _registerStatus = MutableLiveData<Result<RegisterResponse>>()
    val registerStatus: LiveData<Result<RegisterResponse>> = _registerStatus

    private val _detailProfile = MutableLiveData<Result<ProfileResponse>>()
    val detailProfile: LiveData<Result<ProfileResponse>> = _detailProfile

    private val _updateProfileResponse = MutableLiveData<Result<UpdateProfileResponse>>()
    val updateProfileResponse: LiveData<Result<UpdateProfileResponse>> = _updateProfileResponse

    suspend fun register(name: String, email: String, password: String) {
        _registerStatus.value = Result.Loading
        try {
            val request = RegisterRequest(name, email, password)
            val response = apiService.register(request)
            _registerStatus.value = Result.Success(response)
        } catch (e: HttpException) {
            val jsonString = e.response()?.errorBody()?.string()
            val errorBody = Gson().fromJson(jsonString, ErrorResponse::class.java)
            val errorMessage = errorBody.message
            _registerStatus.value = Result.Error(error: errorMessage ?: "An error occurred")
        }
    }
}
```

Gambar 4.21 Kode *Repository*

### D. Injection Factory

*Dependency Injection* merupakan teknik desain perangkat lunak yang memungkinkan suatu objek menerima objek lain yang dibutuhkannya dari luar. Sementara itu, *Factory Pattern* digunakan untuk membuat objek tanpa secara

eksplisit menentukan kelas yang tepat dari objek yang akan dibuat. Tujuan penggunaan teknik-teknik ini adalah untuk menghasilkan kode yang lebih bersih, mudah dikelola, dan mudah diuji. Kode *Injection* dapat dilihat pada Gambar 4.21 dibawah :



```
object Injection {
    fun provideRepository(context: Context): UserRepository {
        val pref = UserPreference.getInstance(context.dataStore)
        val apiService = ApiConfig.getApiService(pref)

        return UserRepository.getInstance(pref, apiService)
    }

    fun provideProductRepository(context: Context): ProductRepository {
        val pref = UserPreference.getInstance(context.dataStore)
        val apiService = ApiConfig.getApiService(pref)

        return ProductRepository.getInstance(pref, apiService)
    }

    fun provideNutritionScanRepository(context: Context): NutritionScanRepository {
        val pref = UserPreference.getInstance(context.dataStore)
        val apiService = ApiConfig.getApiService(pref)

        return NutritionScanRepository.getInstance(apiService)
    }
}
```

Gambar 4.22 Kode *Injection*

#### E. *ViewModel*

*ViewModel* adalah kelas yang dirancang untuk menyimpan dan mengelola data terkait antarmuka pengguna dengan siklus hidup *Activity*. Data yang digunakan dalam *ViewModel* adalah *LiveData* dengan tujuan untuk mengamati perubahan data secara reaktif, serta memanfaatkan *coroutines* untuk operasi *asynchronous*. Tujuan penggunaan *ViewModel* adalah untuk memastikan data antarmuka pengguna tetap ada selama perubahan konfigurasi seperti saat HP di rotasi layar. Selain itu,

*ViewModel* dapat digunakan untuk memisahkan logika bisnis dari komponen antarmuka pengguna. Kode *ViewModel* dapat dilihat pada Gambar 4.23 dibawah.

```
class AuthViewModel(private val userRepository: UserRepository) : ViewModel() {
    val loginData: LiveData<Result<List<String>>> = userRepository.loginData
    val registerStatus: LiveData<Result<RegisterResponse>> = userRepository.registerStatus

    fun login(email: String, password: String) {
        viewModelScope.launch { this: CoroutineScope
            userRepository.login(email, password)
        }
    }

    fun register(username: String, email: String, password: String) {
        viewModelScope.launch { this: CoroutineScope
            userRepository.register(username, email, password)
        }
    }

    fun logout() {
        viewModelScope.launch { this: CoroutineScope
            userRepository.logout()
        }
    }
}
```

Gambar 4.23 Kode *ViewModel*

## 4.4 Implementasi

Implementasi merupakan penerapan dari perancangan sistem yang telah dibuat sebelumnya. Berikut ini adalah hasil implementasi ke dalam aplikasi.

### 4.4.1 Halaman Registrasi

Halaman registrasi digunakan pengguna baru untuk membuat akun. Pengguna diharuskan mengisi data pribadi seperti nama, email, dan kata sandi.

Halaman registrasi aplikasi dapat dilihat pada Gambar 4.24 dibawah ini



The image shows a mobile registration form with a green logo at the top, consisting of a stylized 'U' shape with a spiral inside. Below the logo is the word 'Register' in a green font. The form contains four input fields: 'Name' (with a person icon), 'Email' (with an envelope icon), 'Password' (with a lock icon and an eye icon for visibility), and 'Confirmation Password' (with a lock icon and an eye icon). A green button labeled 'REGISTER' is positioned below the fields. At the bottom, there is a link that says 'Have account? Login'. A large, semi-transparent watermark 'UNIVERSITAS Dinamika' is overlaid on the right side of the form. To the left of the form is a red and white geometric logo.

Gambar 4.24 Halaman Registrasi

#### 4.4.2 Halaman *Login*

Halaman *Login* digunakan pengguna yang sudah terdaftar untuk masuk ke aplikasi. Pengguna diharuskan mengisi alamat email dan kata sandi mereka. Untuk tampilan halaman *Login* aplikasi dapat dilihat pada Gambar 4.26 dibawah ini :



UNIVERSITAS  
Dindamika

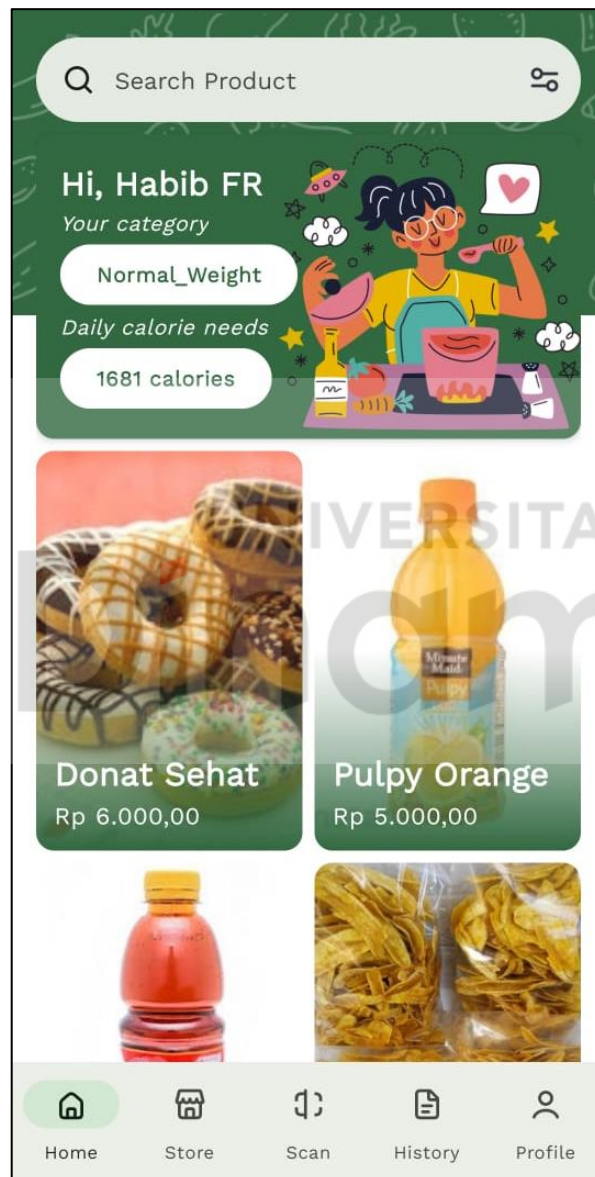
LOGIN

*Don't have account? Register*

Gambar 4.25 Halaman *Login*

#### 4.4.3 Halaman *Home*

Halaman *Home* merupakan halaman awal saat pengguna telah berhasil *Login*. Pada halaman ini pengguna dapat melihat daftar *list* produk. Untuk tampilan halaman *Home* aplikasi dapat dilihat pada Gambar 4.26 dibawah ini :

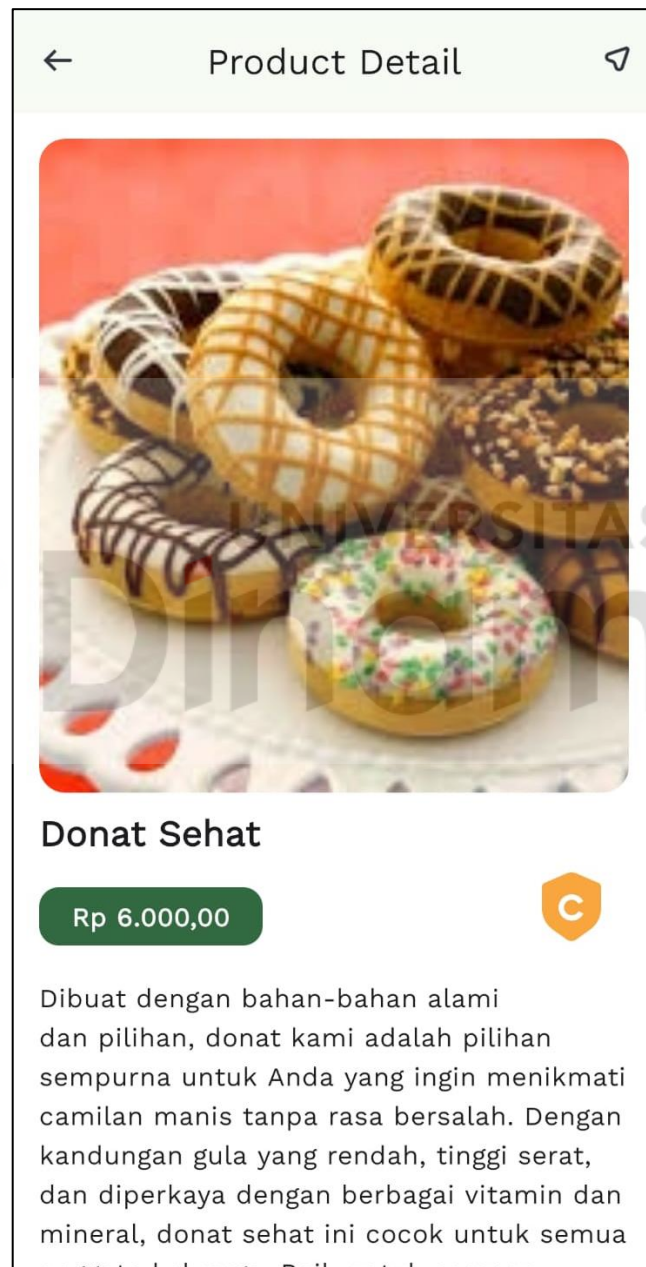


Gambar 4.26 Halaman *Home*

#### 4.4.4 Halaman Detail Produk

Halaman detail produk digunakan untuk memberikan informasi lengkap tentang salah satu produk, yang meliputi gambar, nama, deskripsi, harga dan *grade*.

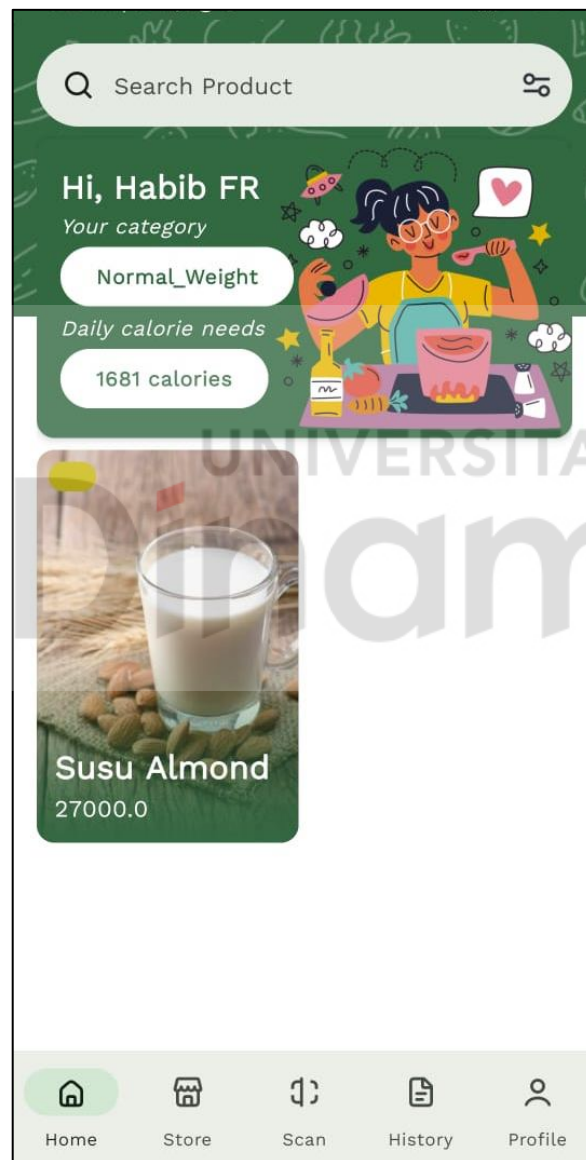
Tampilan halaman *Home* aplikasi dapat dilihat pada Gambar 4.27 dibawah ini :



Gambar 4.27 Halaman Detail Produk

#### 4.4.5 Halaman Pencarian Produk

Halaman pencarian produk digunakan pengguna untuk mencari produk tertentu dengan kata kunci nama produk, kemudian hasil pencarian akan ditampilkan. Tampilan halaman hasil pencarian dapat dilihat pada Gambar 4.28 dibawah ini :

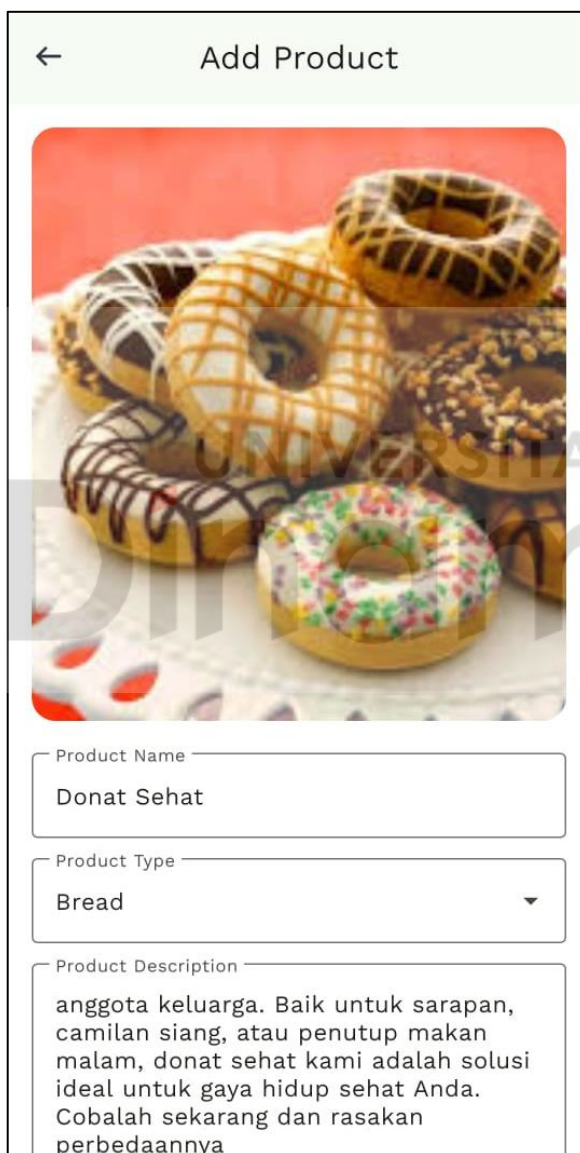


Gambar 4.28 Halaman Pencarian Produk




#### 4.4.6 Halaman Tambah Produk

Halaman tambah produk digunakan pengguna untuk menambahkan produk baru ke dalam aplikasi. Pengguna harus mengisi detail produk seperti gambar, nama, deskripsi, harga dan lain-lain. Untuk tampilan halaman tambah produk dapat dilihat pada Gambar 4.29 dibawah ini :



← Add Product



Product Name  
Donat Sehat

Product Type  
Bread ▼

Product Description  
anggota keluarga. Baik untuk sarapan, camilan siang, atau penutup makan malam, donat sehat kami adalah solusi ideal untuk gaya hidup sehat Anda. Cobalah sekarang dan rasakan perbedaannya

Gambar 4.29 Halaman Tambah Produk

#### 4.4.7 Halaman *Scan* Nutrisi

Halaman *Scan* nutrisi digunakan pengguna untuk memfoto label nutrisi pada makanan atau minuman menggunakan kamera. Untuk tampilan halaman *Scan* nutrisi dapat dilihat pada Gambar 4.30 dibawah ini :



Gambar 4.30 Halaman *Scan* Nutrisi

#### 4.4.8 Halaman *Preview*

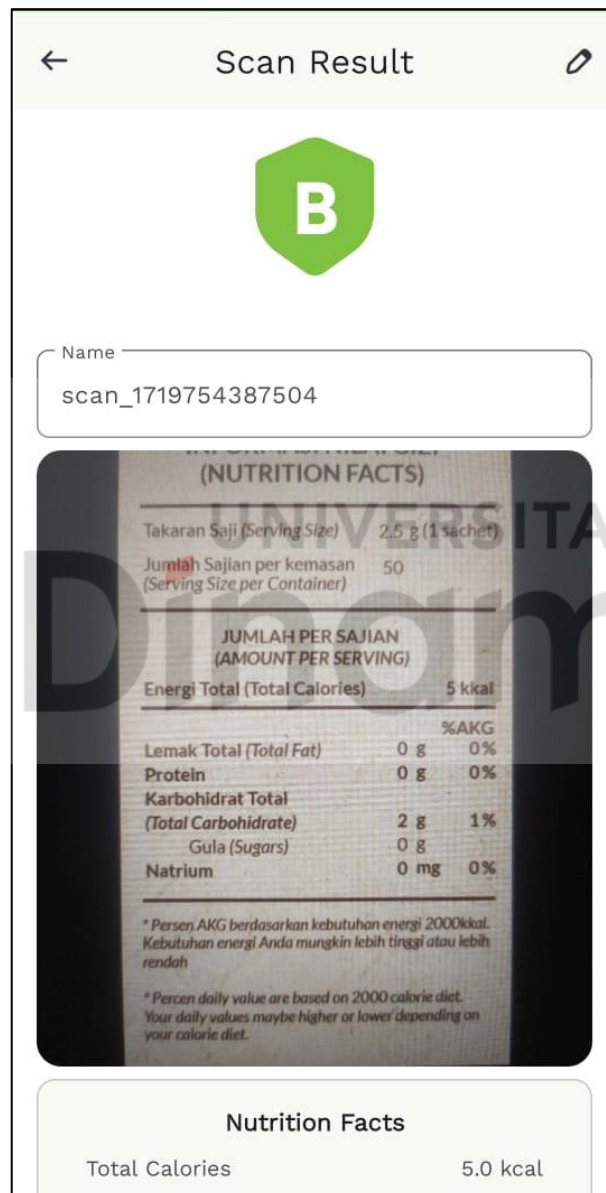
Halaman *preview* digunakan untuk menampilkan foto sebelum diproses perhitungan *grade*. Untuk tampilan halaman *preview* dapat dilihat pada Gambar 4.31 dibawah ini :



Gambar 4.31 Halaman Preview

#### 4.4.9 Halaman Hasil *Scan*

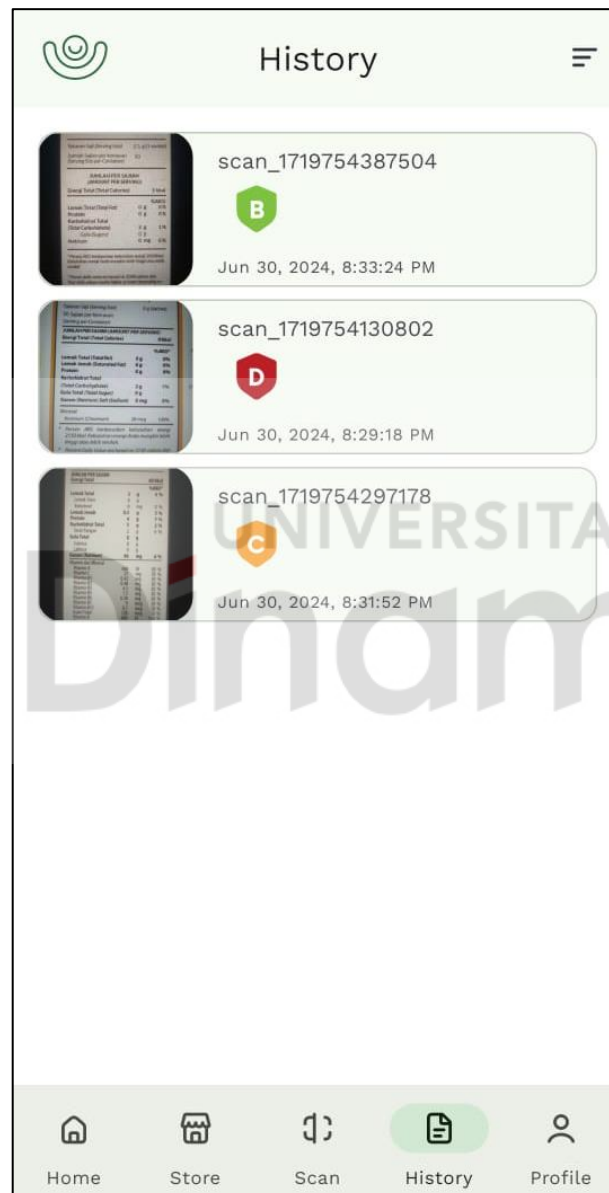
Halaman hasil *scan* digunakan untuk melihat informasi detail dari hasil pemindaian label nutrisi. Pengguna dapat melihat *grade* dan nilai gizi. Untuk tampilan halaman preview dapat dilihat pada Gambar 4.32 dibawah ini :



Gambar 4.32 Halaman Hasil *Scan*

#### 4.4.10 Halaman *History*

Halaman *History* digunakan untuk menyimpan riwayat *scan* nutrisi yang telah dilakukan pengguna. Untuk tampilan halaman *History* dapat dilihat pada Gambar 4.33 dibawah ini :



Gambar 4.33 Halaman *History*

#### 4.4.11 Halaman Profil

Halaman profil digunakan pengguna untuk melihat dan mengedit informasi pribadi pengguna seperti nama, alamat email, dan tanggal lahir. Untuk tampilan halaman *History* dapat dilihat pada Gambar 4.34 dibawah ini



Profile

Name  
Habib FR

Email  
habib@gmail.com

Phone Number  
0854646

Birthdate  
2002-01-10

Gender  
Male

**Measurement**

Height (cm)  
160

Weight (...)  
50

Weight Goal...  
55

Home Store Scan History Profile

Gambar 4.34 Halaman Profil

#### 4.5 Pengujian

Pegujian dilakukan dengan mencoba fitur-fitur yang ada dalam aplikasi untuk memastikan apakah sudah sesuai dengan *output* yang diharapkan. Pengujian menggunakan metode *Blackbox* testing. Berikut ini adalah beberapa pengujian yang dilakukan pada aplikasi ini :

Tabel 4.1 Pengujian *Blackbox* Testing

No	Nama Tes	Proses	Input	Output yang Diharapkan
1	Uji Coba Form Register	Pendaftaran Akun	Nama, Email, dan Kata Sandi	Pengguna sukses mendaftar dan dapat digunakan untuk <i>Login</i>
2	Uji Coba Form <i>Login</i>	<i>Login</i> Aplikasi	Email dan Kata Sandi	Pengguna dapat masuk ke aplikasi halaman <i>Home</i>
3	Uji Coba Pencarian Produk	Pencarian Produk	Nama	Menampilkan produk yang nama nya sesuai dengan input
4	Uji Coba <i>Scan</i> Nutrisi	<i>Scan</i> Nutrisi	Foto	Menampilkan <i>grade</i> dan detail nutrisi
5	Uji Coba Form Tambah Produk	Tambah Produk	Data Produk	Menambah dan menampilkan produk ke halaman <i>Home</i>
6	Uji Form Edit Profil	Edit Profil	Data Pengguna	Mengubah data profil pengguna

## **BAB V**

### **PENUTUP**

#### **5.1 Kesimpulan**

Kesimpulan dalam penelitian ini adalah aplikasi dapat diimplementasikan dan digunakan untuk melakukan *scan* nutrisi pada label makanan atau minuman dengan menerapkan beberapa fitur berikut :

1. Desain *Wireframe* dan desain UI/UX yang telah dibuat dapat diimplementasi ke Android XML dengan menghasilkan tampilan yang responsif dan meningkatkan pengalaman pengguna
2. Penggunaan arsitektur *Model-View-ViewModel* (MVVM) telah berhasil diterapkan untuk memisahkan logika bisnis dari tampilan UI dan meningkatkan *maintainability* kode.
3. *Library Retrofit* juga telah berhasil diimplementasikan untuk menangani komunikasi jaringan antara aplikasi Android dan *server backend* dengan efisien.

#### **5.2 Saran**

Dalam pembuatan aplikasi ini penulis menyadari bahwa hasil akhir ini tidaklah sempurna. Maka saran yang dapat diberikan oleh penulis untuk aplikasi ini dapat dijelaskan sebagai berikut :

1. Mengganti dan menggunakan metode yang lebih akurat dalam melakukan *Scan* gambar label nutrisi pada makanan atau minuman.
2. Menambahkan fitur transaksi pada produk yang memungkinkan pengguna dapat membeli produk langsung melalui aplikasi



## DAFTAR PUSTAKA

- Adjeng, (, Asih, D., Iskandar, K., Peluang, ), Strategi, D., Bandeng, P. T., Elrina, J., Peningkatan, U., Di, B., Modern, E. M., & Dwi Asih, A. (2023). Peluang dan Strategi pada PT Bandeng Juwana Elrina dalam Upaya Peningkatan Bisnis di Era Masyarakat Modern Opportunities and Strategies at PT Bandeng Juwana Elrina in Efforts To Improve Business in The Era Of Modern Society. In *JECMER: Journal of Economic, Management and Entrepreneurship Research* (Vol. 1, Issue 2).
- Agustin Muris, A., Ratu Penghulu No, J., Sari, K., Ogan Komering Ulu, K., & Selatan, S. (2023). INFORMATIKAN DAN TEKNOLOGI (INTECH) Pembuatan Company Profile Rutan Klas II B Baturaja Menggunakan Android Studio. *JURNAL INTECH*, 4(1), 1–6.
- Akmal Hidayat, Nur Qirani Ridhaihi, M Fiqral Ash Shiddiq, Febrianti Tandir Ra'pak, & Andi Ashilla Khaerunnisa. (2023). Pengembangan Aplikasi MySaku Menggunakan Metode *Waterfall*. *Indonesian Technology and Education Journal*, 68–77. <https://doi.org/10.61255/itej.v1i2.178>
- Dedy Kasingku, J., & Lumoindong, B. (2023). Peran Penting Pendidikan Lewat Makanan Bergizi dalam Meningkatkan Kesadaran Masyarakat akan Kesehatan Tubuh dan Pikiran: Studi Literatur. *Journal on Education*, 05(04), 16071–16080.
- Fachri, B., & Surbakti, R. W. (2021). PERANCANGAN SISTEM DAN DESAIN UNDANGAN DIGITAL MENGGUNAKAN METODE *WATERFALL* BERBASIS WEBSITE (STUDI KASUS: ASCO JAYA). *JOURNAL OF SCIENCE AND SOCIAL RESEARCH*, 4(3), 263. <https://doi.org/10.54314/jssr.v4i3.692>
- Fahrezi, A., Salam, F. N., Ibrahim, G. M., Syaiful, R. R., & Saifudin, A. (2023). *Pengujian Black Box Testing pada Aplikasi Inventori Barang Berbasis Web di PT. AINO Indonesia*. <https://journal.mediapublikasi.id/index.php/logic>
- Hasanuddin, Asgar, H., & Hartono, B. (2022). RANCANG BANGUN REST API APLIKASI WESHARE SEBAGAI UPAYA MEMPERMUDAH PELAYANAN DONASI KEMANUSIAAN. *Jurnal Informatika Teknologi Dan Sains*, 4(1), 8–14. <https://doi.org/10.51401/jinteks.v4i1.1474>
- Kristiana Pelealu, S., Moleong, M., Pongoh, L., Studi, P., & Masyarakat, I. K. (2021). HUBUNGAN POLA MAKAN DAN MINUM DENGAN KEJADIAN OBESITAS DI SMA NEGERI 1 TOMOHON. *Jurnal Kesehatan Masyarakat UNIMA*, 2(02), 32–37.

Musfiroh, D. U., Wulandari, D. N., & Muharram, R. R. (2024). *Perancangan Aplikasi Pengenalan Sparepart Sepeda Motor Berbasis Android*. <https://jurnalmahasiswa.com/index.php/jriin>

Nukman, N., Prayudi, Y., & Yudha, F. (2022). Pengembangan Framework Digital Forensics Investigation (FDFI) Pada Sosial Media Dengan Metode System Development Life Cycle (SDLC). *JATISI (Jurnal Teknik Informatika Dan Sistem Informasi)*, 9(3), 1852–1860. <https://doi.org/10.35957/jatisi.v9i3.2151>

Praniffa, A. C., Syahri, A., Sandes, F., Fariha, U., Giansyah, Q. A., & Hamzah, M. L. (2023). PENGUJIAN BLACK BOX DAN WHITE BOX SISTEM INFORMASI PARKIR BERBASIS WEB BLACK BOX AND WHITE BOX TESTING OF WEB-BASED PARKING INFORMATION SYSTEM. In *Jurnal Testing dan Implementasi Sistem Informasi* (Vol. 1, Issue 1).

Shin, S., Puri, J., & Finkelstein, E. (2023). A randomized trial to evaluate the impact of Singapore’s forthcoming Nutri-grade front-of-pack beverage label on food and beverage purchases. *International Journal of Behavioral Nutrition and Physical Activity*, 20(1), 18. <https://doi.org/10.1186/s12966-023-01422-4>

Silitonga, P. D. P., & Purba, D. E. R. (2021). IMPLEMENTASI SYSTEM DEVELOPMENT LIFE CYCLE PADA RANCANG BANGUN SISTEM PENDAFTARAN PASIEN BERBASIS WEB. *Jurnal Sistem Informasi Kaputama (JSIK)*, 5(2), 196–203. <https://doi.org/10.59697/jsik.v5i2.712>

Suherni, P. (2023). Aplikasi Sistem Informasi Transaksi Pelayanan Obat Diapotek Menggunakan Metode Waterfall. *Jurnal SANTI - Sistem Informasi Dan Teknik Informasi*, 1(2), 23–31. <https://doi.org/10.58794/santi.v1i2.323>

Yussandi, R. (2021). ANALISIS DAN PERANCANGAN SISTEM INFORMASI SIMULASI PENGECATAN KENDARAAN BERBASIS ANDROID. *Jurnal Informatika Dan Rekayasa Perangkat Lunak*, 2(3), 382–389. <https://doi.org/10.33365/jatika.v2i3.1240>