Perancangan Dan Implementasi Aplikasi Foodit Berbasis Android Untuk Menghitung Zat Gizi Makro Pada Makanan

1st Kurnivan Noer Yusvianto
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
kurnivannoery@student.telkomuniversity.ac.id

2nd Suryo Adhi Wibowo
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
suryoadhiwibowo@telkomuniversity.ac.id

3rd Koredianto Usman
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
korediantousman@telkomuniversity.ac.id

***Abstrak* — Kesehatan tubuh merupakan hal yang sangat penting dalam kehidupan seseorang. Pemerintah telah mengatur Angka Kecukupan Gizi (AKG) sebagai acuan gizi masyarakat Indonesia. Jika kecukupan gizi tidak terpenuhi, maka dapat menimbulkan gangguan kesehatan. Tujuan dari aplikasi ini adalah untuk memberikan rekomendasi jumlah zat gizi makro yang dikonsumsi setiap hari berdasarkan karakteristik pengguna dan menampilkan informasi gizi tentang risiko penyakit yang terkait dengan konsumsi zat gizi makro yang tidak tepat sebagai pengingat bagi pengguna. Aplikasi dirancang menggunakan *use case diagram* dan *activity diagram*, terhubung ke database, dan terintegrasi dengan model *machine learning* dan API Adrees VM Instance GCP. Pengujian aplikasi menggunakan metode *System Usability Scale* (SUS). SUS adalah metode pengukuran yang digunakan untuk menilai kegunaan sistem atau produk berbasis teknologi. SUS dirancang untuk mengumpulkan data subyektif dari pengguna mengenai pengalaman menggunakan suatu sistem. Pada hasil pengujian rata-rata nilai SUS adalah 86,25 oleh 32 responden yang berada pada *Grade* A, *Adjective Excellent*, *Acceptability Acceptable*, dan *Net Promoter Score* (NPS) *Promoter*. Hal ini membuktikan bahwa aplikasi Foodit memiliki rating yang baik dan layak.**

***Kata kunci* — kesehatan tubuh, Angka Kecukupan Gizi (Akg), aplikasi, zat gizi makro, informasi gizi dan risiko penyakit, *System Usability Scale* (SUS).**

I. PENDAHULUAN

Kesehatan tubuh merupakan hal yang sangat penting dalam kehidupan seseorang. Pemerintah telah mengatur Angka Kecukupan Gizi (AKG) sebagai acuan gizi masyarakat Indonesia. AKG terdiri dari zat makro dan mikro yang dibutuhkan oleh tubuh [1]. Jika kecukupan gizi tidak terpenuhi dapat menyebabkan ganggunan kesehatan. Kekurangan gizi dapat menyebabkan penyakit infeksi, penyakit tidak menular seperti jantung dan stroke, diabetes dan kanker [2]. Faktor penyebab masalah gizi antara lain ketidakseimbangan antara asupan makanan yang dikonsumsi dengan kebutuhan yang dibutuhkan.

Berdasarkan latar belakang diatas, terdapat dua rumusan masalah sebagai berikut:

1. Bagaimana merancang aplikasi yang dapat menghitung dan mengontrol asupan zat gizi makro harian?
2. Bagaimana merancang aplikasi yang mengingatkan masyarakat Indonesia untuk mencegah penyakit akibat mengkonsumsi zat gizi makro yang tidak sesuai dengan kebutuhannya?

Berdasarkan rumusan masalah, terdapat tujuan dari aplikasi sebagai berikut:

1. Aplikasi memberikan rekomendasi jumlah zat gizi makro yang dikonsumsi setiap hari berdasarkan karakteristik pengguna. Dengan rekomendasi tersebut, pengguna dapat mengontrol dan memantau asupan makanan hariannya melalui fitur pencatatan dan analisis nutrisi aplikasi.
2. Aplikasi ini juga akan menampilkan informasi risiko penyakit yang berhubungan dengan konsumsi zat gizi makro yang tidak tepat sebagai pengingat bagi pengguna tentang penyakit yang timbul akibat pola makan yang tidak seimbang.

Oleh karena itu, penulis merancang dan mengimplementasikan sebuah aplikasi bernama Foodit. Aplikasi ini membantu pengguna menghitung dan mengontrol zat gizi makro harian mereka. Dengan fitur pencatatan dan analisis zat gizi makro, aplikasi ini merekomendasikan jumlah zat gizi makro sesuai karakteristik pengguna.

II. KAJIAN TEORI

Pada *Capstone Design* ini, digunakan kajian teoritis dalam merancang dan mengimplementasikan aplikasi Foodit. Kajian teori meliputi perancangan aplikasi menggunakan *use case diagram* dan *activity diagram*. Implementasi aplikasi dengan pembuatan *front-end* dan *back-end*. Pengujian aplikasi menggunakan *System Usability Scale* (SUS). Penjelasan tentang *use case diagram, activity diagram*, dan *front-end, back-end,* dan SUS adalah sebagai berikut.

A. *Use Case Diagram*

Use case diagram adalah representasi visual dari sistem fungsional yang berfungsi sebagai dasar untuk pengembangan perangkat lunak. *Use case diagram* memfasilitasi komunikasi yang efektif antara aktor. *Use case diagram* berfungsi untuk mengungkap, mengatur, dan memvisualisasikan kebutuhan pengguna agar pengembangan sistem perangkat lunak sukses [3].

B. *Activity Diagram*

Activity diagram adalah representasi visual dari langkah-langah, tindakan, dan keputusan dalam alur kerja sistem. *Activity diagram* terdiri dari node yang mewakili aktivitas dan panah yang mewakili aliran kontrol di antara aktivitas tersebut. *Activity diagram* untuk mengindentifikasi potensi hambatan, mengoptimalkan alur kerja, dan memastikan sistem memenuhi tujuan yang diinginkan [4].

C. *Front-End*

Front-end melibatkan pengembangan tampilan antarmuka (UI). Pengembangan front-end pada sistem operasi Android menggunakan *Extensible Markup Language* (XML) di Android Studio. Pola desain yang digunakan dalam pengembangan *front-end*, termasuk penggunaan tata letak XML, pengaturan tampilan melalui Kotlin, penggunaan gaya dan tema, pengelolaan interaksi pengguna, dan penangann perubahan oritenasi layar [5].

D. *Back-End*

Back-end melibatkan pembuatan dan pengelolaan komponen sisi server dengan antarmuka klien seperti penyimpanan data, pengolahan permintaan, dan komunikasi dengan sumber eksternal. Pengembangan *back-end* pada sistem operasi Android menggunakan bahasa pemrograman Kotlin. *Back-end* memainkan peran penting dalam menyimpan dan mengelola data menggunakan permintaan HTTP menggunakan Retrofit dengan 2eknik seperti Gson atau JSON dan menggunakan layanan dari Firebase seperti Firebase Realtime Database atau Cloud Firestore [6].

E. *System Usability Scale*

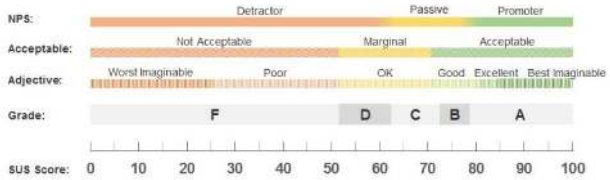
System Usability Scale (SUS) adalah suatu metode pengukuran yang digunakan untuk mengevaluasi kegunaan sistem atau produk yang berbasis teknologi. SUS dirancang untuk mengumpulkan data subjektif dari pengguna mengenai pengalaman penggunaan suatu sistem. SUS menggunakan skala Likert dari satu sampai lima yaitu 1 (sangat tidak setuju, 2 (tidak setuju), 3 (netral), 4 (setuju), dan 5 (sangat setuju) [7]. Pertanyaan pada kuesioner SUS pada Tabel 1 [7] sebagai berikut:

TABEL 1
Kuesioner Sus

No	Pertanyaan Kuesioner SUS
1.	Saya pikir saya akan sering menggunakan sistem ini.
2.	Saya menemukan sistem kompleks yang tidak perlu.
3.	Saya pikir sistemnya mudah digunakan.
4.	Saya pikir saya akan membutuhkan dukungan dari orang teknis untuk dapat menggunakan sistem ini.
5.	Saya menemukan berbagai fungsi dalam sistem ini terintegrasi dengan baik.
6.	Saya pikir ada terlalu banyak inkonsistensi dalam sistem ini.
7.	Saya membayangkan bahwa kebanyakan orang akan belajar menggunakan sistem ini dengan sangat cepat.
8.	Saya menemukan sistem yang sangat rumit untuk digunakan.
9.	Saya merasa sangat yakin menggunakan sistem.
10.	Saya perlu belajar banyak hal sebelum saya bisa menggunakan sistem ini.

Adapun perhitungan hasil pengukuran SUS [7] yaitu:

1. Untuk setiap pertanyaan urutan ganjil, nilai dikurangi dengan satu(nilai -1).
2. Untuk setiap pertanyaan urutan genap, lima dikurangi dengan nilai (5 - nilai).
3. Jumlahkan nilai-nilai dari pertanyaan bernomor ganjil dan genap. Kemudian hasil penjumlahan tersebut dikalikan dengan 2,5.



GAMBAR 1
Grades, Adjectives, Acceptability, Dan Nps

Untuk mengetahui kegunaan dan kelayakan aplikasi dengan mengelompokkan *Grades, Adjectives, Acceptability*, dan *Net Promoter Score* (NPS) dilihat pada Gambar 1 [8][9]. Grades memiliki nilai dari A (kinerja unggul) hingga F (kinerja gagal) dan C (rata-rata). Adjectives berisi enam kata sifat yang menggambarkan pengalaman pengguna dari suatu aplikasi yaitu *Best Imaginable, Excellent, Good, OK, Poor,* dan *Worst Imaginable*. Acceptability menjelaskan dapat atau tidaknya aplikasi diterima oleh pengguna yaitu *Acceptable*, *Marginally Acceptable*, dan *Not Acceptable*. NPS menjelaskan kemungkinan bagi pengguna untuk merekomendasikan aplikasi yaitu *Promoter*, *Passive*, dan *Dectractor* [9].

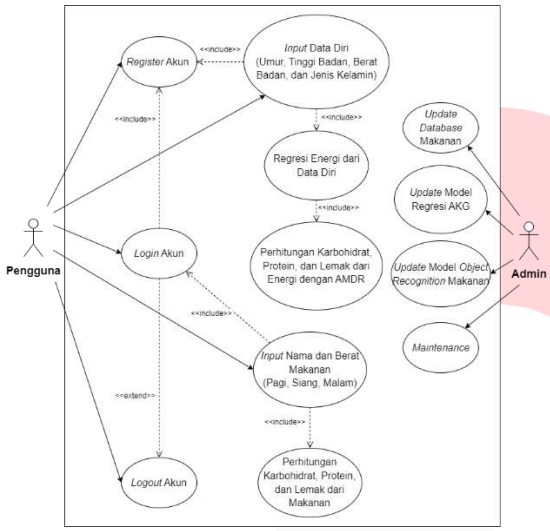
III. PERANCANGAN SISTEM

Pada *Capstone Design*, perancangan sistem atau aplikasi menggunakan *use case diagram* dan *activity diagram*. Kemudian dibuat perancangan *front-end* yang merupakan aplikasi interface dan *back-end* yang merupakan aplikasi manajemen sistem. Kemudian terdapat perancangan implementasi model regresi dan *object recognition* pada Android. Penjelasan tentang *use case diagram*, *activity diagram*, perancangan *front-end* dan *back-end*, perancangan implementasi model regresi di Android, dan perancangan implementasi model *object recognition* pada Android adalah sebagai berikut.

A. *Use Case Diagram*

Pada Gambar 2 terdapat dua aktor yang terlibat dalam aplikasi yaitu, pengguna dan admin. Pengguna harus melakukan registrasi akun terlebih dahulu sebelum masuk ke akun. Ketika pengguna sudah masuk ke akun, pengguna dapat meninggalkan akun tersebut dan masuk kembali. Pada saat pengguna melakukan registrasi akun, pengguna memasukkan data diri seperti umur, tinggi badan, berat badan, dan jenis kelamin untuk menentukan energi yang dibutuhkan pengguna dalam satu hari menggunakan model regresi. Setelah mendapatkan energi menggunakan model

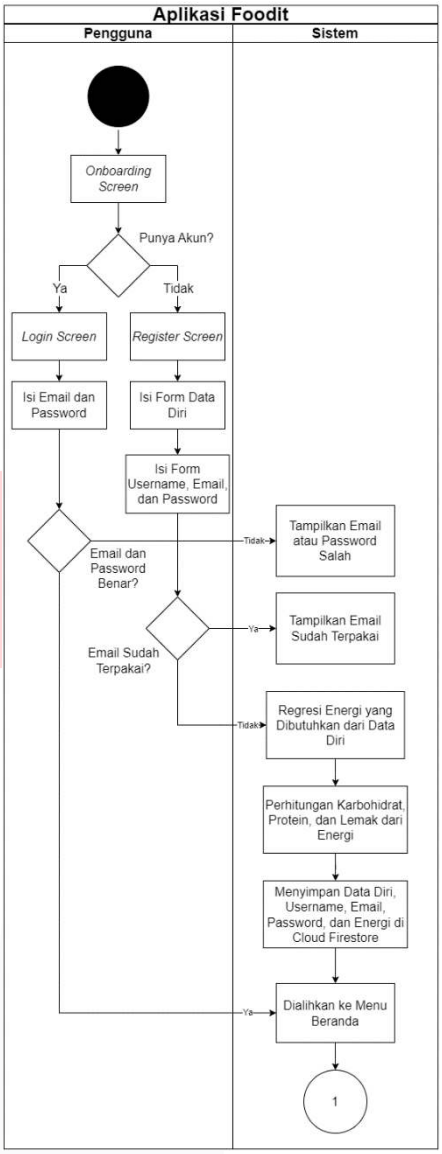
regresi, kebutuhan zat gizi makro berupa karbohidrat, protein, dan lemak akan dihitung menggunakan AMDR. Pengguna juga dapat menggunakan fitur *input* nama dan berat makanan untuk memasukkan makanan yang dikonsumsi pada hari itu sesuai dengan waktunya, yaitu pagi, siang, dan malam. Setelah memasukkan makanan yang dikonsumsi, zat gizi makro berupa karbohidrat, protein, dan lemak yang terkandung dalam makanan tersebut akan dihitung. Sedangkan Admin harus memperbarui *database* makanan, model regresi AKG, model *object recognition* makanan, dan pemeliharaan/*maintenance* aplikasi.



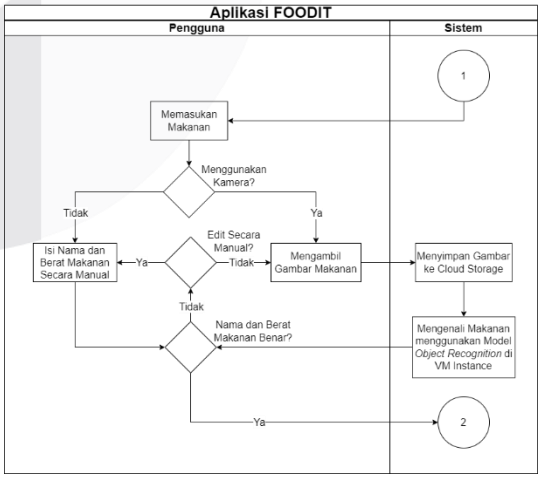
GAMBAR 2
Use Case Diagram

B. Activity Diagram

Pada Gambar 3 terdapat dua *business entity* yang terlibat dalam aplikasi, yaitu pengguna dan sistem. Aktivitas dimulai dengan pengguna masuk ke aplikasi melalui *onboarding screen.* Jika pengguna memiliki akun, pengguna akan mengisi *email* dan *password*. Jika email dan *password* benar, pengguna akan dialihkan ke menu beranda. Jika pengguna tidak memiliki akun, pengguna akan mendaftar akun dengan mengisi formulir data diri (umur, berat badan, tinggi badan, dan jenis kelamin), *username*, *email*, dan *password* di *register screen*. Setelah pengguna mengisi formulir, sistem akan menghitung energi yang diperoleh dari model regresi. Sistem akan menghitung kebutuhan karbohidrat, protein, dan lemak per hari menggunakan AMDR berdasarkan energi. Sistem akan menyimpan data diri, *username*, *email*, *password,* dan energi di Cloud Firestore. Setelah itu, pengguna akan dialihkan ke menu beranda.



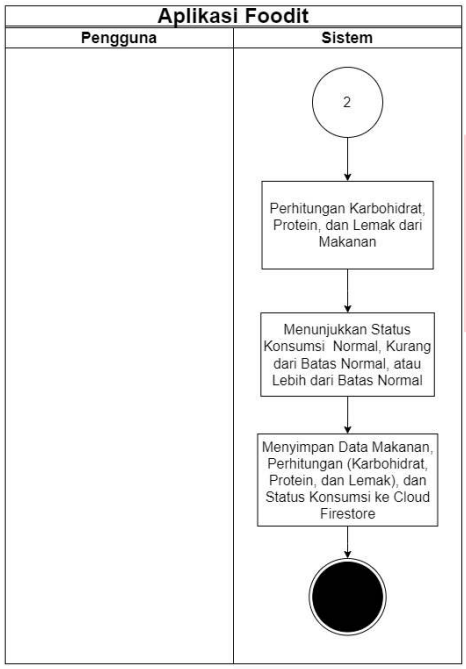
GAMBAR 3
Activity Diagram 1



GAMBAR 4
Activity Diagram 2

Setelah pengguna dialihkan ke menu beranda, pengguna akan memasukkan makanan yang dikonsumsi pada hari tersebut seperti pada Gambar 4. Tersedia dua cara untuk memasukkan makanan, yaitu dengan mengisi nama dan berat

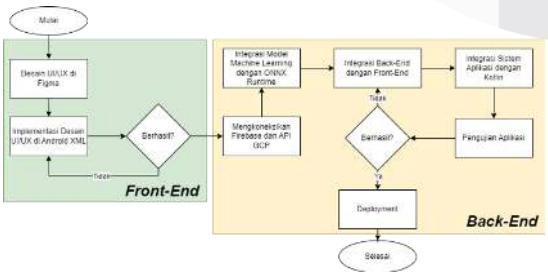
makanan secara manual atau dengan bantuan kamera. Saat pengguna mengambil gambar makanan menggunakan bantuan kamera, sistem akan menyimpan gambar di Cloud Storage. Kemudian mengenali gambar makanan tersebut menggunakan model *object recognition* di VM Instance GCP. Setelah itu, pengguna harus memeriksa kembali apakah nama dan berat makanan yang telah dikenali benar atau salah. Jika nama dan berat makanan salah, maka pengguna akan mengambil gambar ulang atau diarahkan ke pengisian nama dan berat makanan secara manual untuk melakukan koreksi nama dan berat makanan.



GAMBAR *5*
Activity Diagram 3

Jika nama dan berat makanan sudah sesuai, maka sistem akan menghitung karbohidrat, protein, dan lemak berdasarkan nama dan berat makanan tersebut seperti pada Gambar 5. Sistem juga memeriksa apakah status konsumsi karbohidrat, protein, dan lemak pada hari itu normal, kurang dari batas normal, atau lebih dari batas normal. Kemudian sistem menyimpan data makanan berupa nama, berat, nama foto, perhitungan total karbohidrat, protein, dan lemak, dan status konsumsi untuk hari itu. Data tersebut akan disimpan di Cloud Firestore.

C. Perancangan Front-End dan Back-End

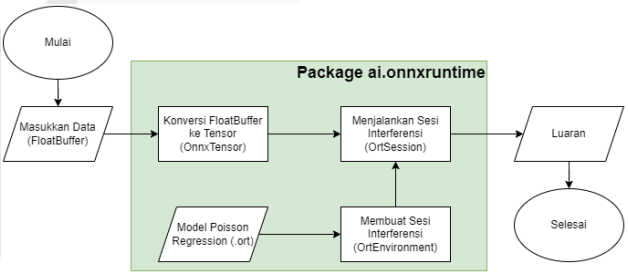


GAMBAR 6
Diagram Alir *Front-End* Dan *Back-End*

Pada Gambar 6 merupakan diagram alir pengembangan aplikasi Android yang terbagi menjadi *front-end* dan *back-end*. *Front-end* bertugas untuk merancang tampilan antarmuka pengguna yang meliputi tampilan visual, tata letak, dan interaksi pengguna. Sebelum membuat tampilan antarmuka pengguna di Android Studio, langkah awal yang perlu dilakukan adalah merancang desain UI/UX menggunakan Figma. *User Interface* (UI) mengacu pada tampilan antarmuka pengguna, sedangkan *User Experience* (UX) mengacu pada pengalaman pengguna. Setelah merancang desain UI/UX, desain UI/UX akan diimplementasikan menggunakan *Extensible Markup Language* (XML) di Android Studio. XML adalah format *markup* yang digunakan untuk menggambarkan struktur dan isi dokumen. Saat aplikasi Android dijalankan, XML akan dibaca dan diinterpretasikan untuk menampilkan desain UI/UX yang telah dirancang sebelumnya di Android Studio. Selain itu, XML juga digunakan untuk menyediakan sumber daya aplikasi seperti *string*, gambar, warna, ukuran, dan kontrol pengguna lainnya yang diperlukan dalam pengembangan aplikasi Android.

Back-end bertugas untuk menyiapkan dan mengelola sistem aplikasi menggunakan bahasa pemrograman Kotlin. Selain itu, *back-end* berperan penting dalam menyimpan dan mengelola data, menjalankan model *machine learning*, dan bertindak sebagai penghubung antara aplikasi dan API. Penggunaan *back-end* menjadi penting dalam mengolah data dari aplikasi dan menyimpannya di Cloud Firestore, sehingga data tersebut dapat diambil kembali untuk ditampilkan di aplikasi. Selain itu, *back-end* juga dapat menjalankan model *machine learning* untuk menghasilkan prediksi kalori berdasarkan data diri pengguna. *Back-end* juga berfungsi sebagai penghubung antara aplikasi dan API, khususnya API yang digunakan untuk mengenali jenis makanan yang berjalan di Google Cloud Platform. Setelah *back-end* selesai, aplikasi akan diuji seacara fungsional untuk menguji apakah aplikasi bekerja seperti yang diharapkan. Setelah itu, aplikasi akan dikompilasi menjadi file *Android App Bundle* (AAB) untuk diunggah di Google Play Store agar aplikasi dapat didistribusikan dan diunduh oleh pengguna.

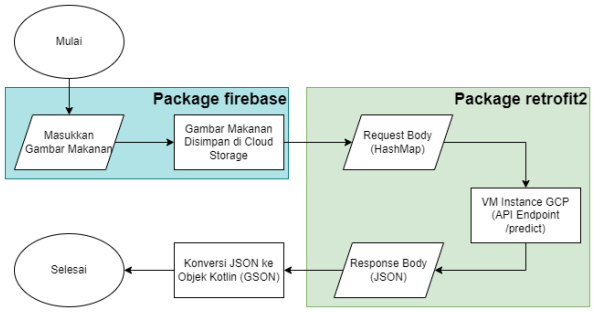
D. Perancangan Implementasi Model Regresi



GAMBAR 7
Diagram Alir Implementasi Model Regresi

Pada Gambar 7 merupakan diagram alir cara kerja model regresi pada Android menggunakan *package* ai.onnxruntime. Data diri pengguna seperti umur, berat badan, tinggi badan, dan jenis kelamin disimpan dalam tipe data FloatBuffer. Kemudian, data tersebut dikonversi menjadi Tensor menggunakan OnnxTensor. Untuk menjalankan sesi interferensi menggunakan OrtSession, sesi interferensi dibuat menggunakan OrtEnvironment dengan model poisson *regression* (.ort) sebagai model regresi. Hasil prediksi yang dihasilkan adalah energi.

E. Perancangan Impementasi Model *Object Recognition*



GAMBAR 8
Diagram Alir *Front-End* Dan *Back-End*

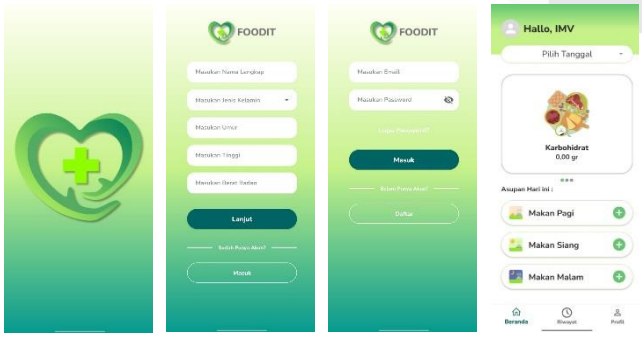
Pada Gambar 8 merupakan diagram alir cara kerja model *object recognition* pada Android. Gambar makanan yang diambil oleh pengguna akan disimpan di Cloud Storage menggunakan *package* firebase. Kemudian sistem aplikasi akan mengirim *request body* dengan data seperti image_url, useruid, bulan_makan, tanggal_makan, dan waktu_makan dan tipe data HashMap. Data tersebut akan dikirim ke VM Instance GCP melalui API *Endpoint* (/predict) menggunakan *package* retrofit2 dengan metode POST. VM Instance GCP akan mengirimikan *response body* dengan data seperti status_predict dan format JSON. Format JSON akan dikonversi menjadi objek Kotlin menggunakan GSON.

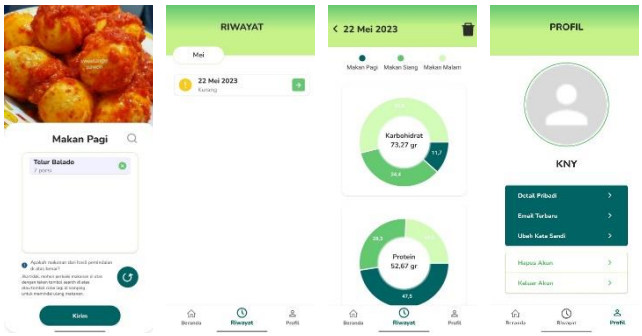
IV. HASIL DAN PEMBAHASAN

Pada Capstone Design, hasil perencanaan sistem adalah aplikasi Foodit. Aplikasi akan diuji menggunakan SUS untuk mengetahui kegunaan dan kelayakan aplikasi. Penjelasan hasil implementasi dan pengujian SUS adalah sebagai berikut.

A. Hasil Implementasi

Hasil akhir dari *android development* adalah sebuah aplikasi yang dapat diunduh melalui Google Play Store. Dalam mengatur dan mengontrol versi aplikasi menggunakan *Version Control System* (VCS). VCS yang digunakan berupa Git yang bekerja secara offline dan GitHub yang merupakan layanan *cloud computing* untuk menyimpan proyek secara *online*. Dengan menggunakan Git, tim penulis dapat dengan mudah mengatur dan menyimpan perubahan pada aplikasi. Struktur kode proyek aplikasi ini dapat diakses melalui GitHub.





GAMBAR 9
Halaman Aplikasi

Halaman aplikasi Foodit dapat dilihat pada Gambar 9. Tampilan antarmuka untuk register atau daftar akun. Untuk mendaftarkan akun, pengguna harus menekan tombol daftar dan memasukkan nama lengkap, jenis kelamin, umur, tinggi badan, dan berat badan di *textfield.* Setelah mengisi informasi tersebut, pengguna menekan tombol lanjut dan memasukkan *username,* email, dan *password* di *textfield*. Sebelum menekan tombol daftar, pengguna harus menyetujui kebijakan privasi. Setelah menyetujui kebijakan privasi, pengguna dapat menekan tombol daftar dan data pengguna akan disimpan di Firebase Authentification dan Cloud Firestore. Tampilan antarmuka untuk *login* atau masuk akun. Untuk memasukan akun, pengguna harus menekan tombol masuk dan memasukkan *email* dan *password* di *textfield*. Kemudian pengguna menekan tombol masuk, sistem autentikasi dari Firebase Authentication akan memeriksa *email* dan *password* yang dimasukkan. Jika *username* dan password benar, maka pengguna dapat masuk ke aplikasi dan sebaliknya. Jika pengguna lupa *password*, maka pengguna dapat menggunakan fitur lupa *password*. Jika pengguna memilih fitur lupa *password*, pengguna dapat memeriksa email untuk mengatur ulang *password*.

Tampilan antarmuka untuk beranda. Pengguna harus memilih tanggal untuk mencatat konsumsi makanan. Setelah memilih tanggal, pengguna dapat melihat informasi mengenai batas konsumsi karbohidrat, protein, dan lemak pada hari tersebut yang dibedakan dengan warna kuning (kurang), hijau (normal), dan merah (lebih) pada tampilan *viewpager*. Pengguna dapat menekan tombol makan pagi, makan siang, atau makan malam untuk memasukkan makanan yang dikonsumsi pada waktu tersebut. Setelah itu, pengguna akan memilih apakah ingin mencatat makanan secara otomatis atau manual. Tampilan antarmuka untuk memasukkan makanan secara otomatis. Jika pengguna memilih ya, maka pengguna akan diberikan pilihan untuk menggunakan kamera atau galeri untuk mendapatkan gambar makanan. Sistem kemudian akan mengenali makanan berdasarkan masukkan gambar. Hasil pengenalan akan ditampilkan bersama dengan gambar makanan. Jika pengenalan tersebut tidak sesuai, maka pengguna dapat mengedit atau menambahkan informasi terkait.

Tampilan antarmuka untuk memasukkan makanan secara manual. Jika pengguna memilih tidak, maka pengguna akan memasukkan makanan secara manual. Pengguna akan memilih makanan yang dikonsumsi, lalu memasukkan berat makanan dalam porsi atau gram. Setelah itu, pengguna dapat menekan tombol tambah jika informasi sudah benar.

Terakhir, pengguna dapat menekan tombol kirim untuk mengirim informasi makanan.

Tampilan antarmuka untuk riwayat. Pengguna akan memilih bulan untuk memfilter data berdasarkan bulan yang diinginkan. Setelah itu, data akan diurutkan berdasarkan tanggal terbaru. Pengguna kemudian dapat memilih tanggal dengan menekannya untuk melihat informasi grafis konsumsi karbohidrat, protein, dan lemak pada pagi, siang dan malam hari. Pengguna juga dapat menekan sekitar informasi grafis untuk melihat informasi makanan yang dikonsumsi saat itu termasuk rincian karbohidrat, protein, atau lemak yang terkandung dalam makanan tersebut.

Tampilan antarmuka untuk profil. Pengguna dapat mengedit informasi mengenai pengguna. Detail pribadi digunakan untuk mengedit informasi mengenai pengguna berupa nama, *username*, jenis kelamin, umur, tinggi badan, dan berat badan. *Email* terbaru digunakan untuk memperbarui *email*. Ubah kata sandi digunakan untuk memperbarui *password*/kata sandi. Hapus akun digunakan untuk menghapus akun. Keluar akun digunakan untuk keluar dari aplikasi.

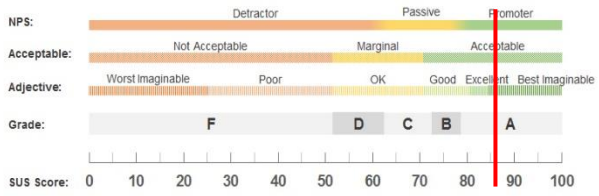
B. Pengujian System Usability Scale

Pengujian SUS akan dilakukan di Google Form dengan pertanyaan yang mengikuti format SUS. Hasil akhir nilai SUS dikelompokkan menggunakan *Grades*, *Adjectives*, *Acceptability*, dan NPS. Pada Tabel 1 merupakan hasil pengujian SUS di Google Form.

TABEL 2
Hasil Pengujian Sus

No	Nama	Nilai SUS
1.	Kurniawan Malik Ibrahim	85
2.	Kurniawan Setiaji Saputra	95
3.	Aini Mulyani Rahman	70
4.	Mahdar Arianto	97,5
5.	Nur Rizki	85
6.	Dinal Gtg	85
7.	Almer Satria Rama	90
8.	Ilmi Aulia Assyifatunisa	90
9.	Maulin Nasari	70
10.	Glenn	72,5
11.	Ilyasa	97,5
12.	Arik	90
13.	Mukhlisin Kurnia Bakti	72,5
14.	Muhamad Al Fitra	95
15.	Fargy Septiari Munir	90
16.	Dhiya Fathiyya	77,5
17.	Reynaldhi	87,5
18.	Jonathan Vito	82,5
19.	Kholis Nur Santosa	87,5
20.	Alvaro Septra Dominggo N	100
21.	Mawar	77,5
22.	Affan Baehaqi	87,5
23.	Laily Nur Amaliah	90
24.	Muhammad Izzah Alfatih	100
25.	Nabila Dwi Pitaloka	85
26.	Winda Maharani Sihotang	97,5
27.	Dea Tsamara	75
28.	Dewi Nurulaeni Achdalina	85
29.	Ferdian Ilham Ramadhan	75
30.	Andi Wahyu Maulana	90
31.	Muhamad Ilham	87,5
32.	Indratama Pangasian Manalu	90
Rata-rata Nilai SUS		86,25

Berdasarkan hasil pengujian SUS, diperoleh rata-rata nilai SUS sebesar 86,25 dari 32 responden. Untuk mengetahui penilaian dan kelayakan aplikasi menggunakan *Grades*, *Adjectives*, *Acceptability*, dan NPS. Garis merah pada Gambar 10 menunjukkan bahwa rata-rata nilai SUS berada pada *Grade* A, *Adjective Best Imaginable*, *Acceptability Acceptable*, dan NPS Promoter. Dapat disimpulkan bahwa aplikasi Foodit memiliki kinerja yang unggul, pengalaman pengguna yang baik, dapat diterima oleh pengguna, dan pengguna akan merekomendasikan aplikasi ini. Dengan demikian, aplikasi Foodit memiliki rating yang baik dan layak untuk digunakan. Selain itu, tim penulis juga meminta saran dari responden terkait masalah spesifik pada sistem atau fitur. Saran-saran ini akan menjadi dipertimbangkan dalam pengembangan aplikasi selanjutnya.



GAMBAR 10
Grades, *Adjectives*, *Acceptability*, Dan Nps Pada Aplikasi Foodit

V. KESIMPULAN

Berdasarkan hasil implementasi, pengujian, dan analisis pengujian yang telah dilakukan terhadap aplikasi Foodit, dapat disimpulkan bahwa aplikasi berhasil dirancang dengan menggunakan *use case diagram*, *activity diagram,* terhubung dengan database, dan terintegrasi dengan model *machine learning* serta API *Adrees* VM Instance GCP. Pada hasil pengujian, nilai rata-rata SUS adalah 86,25 yang berada pada Grade A, Adjective Excellent, Acceptability Acceptable, dan NPS Promoter. Hal ini membuktikan bahwa aplikasi Foodit memiliki rating yang baik dan layak.

REFERENSI

[1] L. A. Furkon, “Ilmu Gizi dan Kesehatan,” J. Chem. Inf. Model., vol. 53, no. 9, pp. 1689–1699, 2016.

[2] K. K. KEMENKES, “Peraturan Menteri Kesehatan Republik Indonesia Nomor 41 Tahun 2014 Tentang Pedoman Gizi Seimbang,” p. 97, 2014.

[3] L. Setiyani, “Desain Sistem: Use Case Diagram,” dalam Prosiding Seminar Nasional Inovasi dan Adopsi Teknologi (INOTEK), 2021, hlm. 246–260.

[4] Satzinger, J. W., Jackson, R. B., & Burd, S. D. (2015). Systems Analysis and Design in a Changing World. Cengage Learning.

[5] Doe, J., & Smith, J. (2022). "Android Front-end Development: Best Practices and Design Patterns." Journal of Mobile App Development, Vol. 10, No. 2, 45-62.

[6] “Documentation: Android Developers,” Android Developers, https://developer.android.com/docs (accessed Jul. 17, 2023).

[7] J. Brooke, “SUS: A quick and dirty usability scale,” Usability Eval. Ind., vol. 189, 1995.

[8] J. Sauro, A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices. Measuring Usability LLC, 2011. [Online]. Available: https://books.google.co.id/books?id=BL0kKQEACAAJ.

[9] J. Sauro, “5 Ways to Interpret,” 2018. https://measuringu.com/interpret-susscore/ (accessed Dec. 30, 2022).