Pengembangan Sistem Backend Menggunakan .Net Pada Aplikasi Konsultasi Makanan Diet Menggunakan Deep Learning

1st Harvan Nurluthfi Irawan
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
whitefall@student.telkomuniversity.ac.id

Casi Setianingsih
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
setiacasie@telkomuniversity.ac.id

Anggunmeka Luhur Prasasti
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
anggunmeka@telkomuniversity.ac.id

***Abstrak* — Penelitian ini membahas implementasi *backend* pada aplikasi konsultasi makanan diet menggunakan *deep learning*. *Backend* dikembangkan menggunakan .NET dan PostgreSQL untuk mengelola data pengguna dan makanan. Model YOLOv8 digunakan untuk mendeteksi makanan pada gambar, dengan informasi nutrisi yang diambil dari FatSecret. Integrasi model dilakukan menggunakan ONNX Runtime, dan aplikasi di-deploy di Google Cloud Run menggunakan Docker. Hasil pengujian beban menunjukkan bahwa backend ini mampu menangani deteksi makanan untuk setidaknya 10 pengguna secara bersamaan, dengan masing-masing pengguna melakukan hingga 100 permintaan, sehingga total mencapai 1.000 permintaan. Implementasi ini diharapkan menjadi dasar bagi pengembangan aplikasi konsultasi diet yang lebih canggih.**

***Kata kunci*— backend, deep learning, yolov8, onnx, .net, docker, gcp, fatsecret, diet**

I. PENDAHULUAN

Kesehatan merupakan salah satu aspek yang sangat penting dalam kehidupan manusia dan berperan penting dalam kesejahteraan individu secara keseluruhan. Pemahaman yang buruk tentang pentingnya makanan seimbang dapat menyebabkan pola makanan yang tidak sehat, meningkatkan angka obesitas, dan menyebabkan masalah kesehatan lainnya. Selain itu, keterbatasan waktu untuk berkonsultasi dengan ahli gizi dapat menjadi hambatan dalam menjalankan program diet.

Konsultasi diet secara tradisional sering kali memerlukan biaya yang tinggi dan memakan banyak waktu, sehingga tidak semua orang dapat mengakses layanan ini secara rutin. Tantangan ini semakin diperparah dengan meningkatnya prevalensi obesitas, hipertensi, dan diabetes tipe 2 di Indonesia, yang semuanya terkait dengan pola makan yang tidak sehat. Dengan semakin banyaknya masyarakat yang membutuhkan panduan diet yang tepat, solusi yang efektif dan efisien menjadi sangat dibutuhkan.

Implementasi teknologi, seperti aplikasi konsultasi makanan diet, menawarkan alternatif yang lebih terjangkau dan praktis, yang dapat membantu mengatasi hambatan biaya dan waktu serta memberikan akses yang lebih luas untuk masyarakat terhadap pengetahuan seputar diet dan pola makan sehat. Aplikasi ini nantinya akan diberi nama "Dietary", yang diambil dari kata "Diet" (aturan makanan khusus dalam menjaga pola makan) dan "Diary" (buku harian). Dengan demikian "Dietary" adalah aplikasi yang dapat memonitor pola makan dan menjadikannya sebagai buku harian.

Untuk mencapai tujuan ini, sistem *backend* dirancang dengan menggunakan Application Programming Interface (API) yang didukung oleh *framework* .NET. Pemilihan .NET didasarkan pada ketersediaan *library* bawaan yang luas dan beragam terutama dukungan untuk integrasi ONNX Runtime yang sangat penting untuk implementasi model *deep learning*. Dengan pembuatan sistem ini diharapkan aplikasi konsultasi makanan diet menggunakan *deep learning* dapat berjalan dengan optimal.

II. KAJIAN TEORI

A. *Deep learning* untuk Konsultasi Makanan Diet

Deep learning adalah cabang dari *machine learning* yang menggunakan jaringan saraf tiruan dengan lapisan berjumlah besar. Dalam konteks konsultasi makanan diet, *deep learning* digunakan untuk mendeteksi dan mengenali berbagai jenis makanan dalam gambar. Algoritma seperti YOLO (You Only Look Once) mampu melakukan deteksi objek secara real-time dengan akurasi tinggi. Informasi nutrisi dari makanan yang terdeteksi kemudian dapat diolah untuk memberikan rekomendasi diet yang tepat. Keuntungan menggunakan *deep learning* adalah kemampuannya untuk belajar dan meningkatkan akurasi seiring dengan bertambahnya data yang diproses.

B. .NET Framework untuk Pengembangan Backend

.NET Framework adalah platform pengembangan yang mendukung pembuatan aplikasi untuk Windows, web, dan perangkat mobile. Dalam proyek ini, .NET Core digunakan untuk mengembangkan backend dari aplikasi konsultasi makanan diet. .NET Core memungkinkan pengembangan aplikasi yang cepat dan efisien, serta mendukung berbagai bahasa pemrograman seperti C#. Keunggulannya adalah cross-platform compatibility, yang memungkinkan aplikasi berjalan di berbagai sistem operasi, termasuk Windows, Linux, dan macOS. Ini memfasilitasi integrasi dengan berbagai layanan dan deployment di lingkungan cloud seperti Google Cloud Platform.

C. Keamanan JWT (JSON Web Token)

Keamanan adalah aspek kritis dalam pengembangan aplikasi, terutama yang berkaitan dengan data sensitif seperti informasi diet pengguna. JSON Web Token (JWT) adalah standar terbuka yang digunakan untuk berbagi informasi keamanan antara klien dan server. JWT memungkinkan autentikasi yang aman dan efisien dengan cara mengkodekan klaim dalam format JSON yang ditandatangani secara digital. Dalam aplikasi ini, JWT digunakan untuk memastikan bahwa hanya pengguna yang terotentikasi yang dapat mengakses layanan backend, sehingga meningkatkan keamanan dan perlindungan data pengguna. Secara umum JWT terbagi menjadi tiga bagian, sebagai berikut:

1. Header: berisi informasj tentang algoritma yang digunakan untuk menghasilkan tanda tangan, seperti HMAC, RSA, dan SHA 256
2. Payload: berisi klaim atau pernyataan tentang entitas dan data tambahan
3. Signature: dihasilkan menggunakan header, payload, dan kunci rahasia. Biasanya digunakan untuk memverifikasi keaslian dan integritas token.

D. Entity Framework Core (EF Core) sebagai ORM

Entity Framework Core (EF Core) adalah Object-Relational Mapper (ORM) untuk .NET Core. EF Core mempermudah pengembangan aplikasi dengan mengizinkan developer untuk bekerja dengan data dalam bentuk objek, tanpa perlu menulis kode SQL secara langsung. Dengan EF Core, pengelolaan database menjadi lebih mudah dan efisien. ORM ini mendukung berbagai database, seperi PostgreSQL, yang digunakan dalam proyek ini. EF Core memungkinkan manipulasi data dengan aman, serta mendukung migrasi database yang mempermudah pengembangan aplikasi.

E. ONNX Runtime

ONNX Runtime adalah mesin inferensi yang dirancang untuk menjalankan model machine learning yang diformat dalam ONNX (Open Neural Network Exchange). ONNX memungkinkan berbagai framework machine learning seperti PyTorch, TensorFlow, dan scikit-learn untuk saling beroperasi. Dalam proyek ini, ONNX Runtime digunakan untuk menjalankan model *deep learning* yang sudah dilatih menggunakan YOLOv8 (*object detection*). ONNX dipilih karena dukungannya yang bagus terhadap C# dan integrasinya dengan .NET. Hal ini memastikan bahwa proses inferensi model berjalan dengan lancar dan cepat di lingkungan deployment.

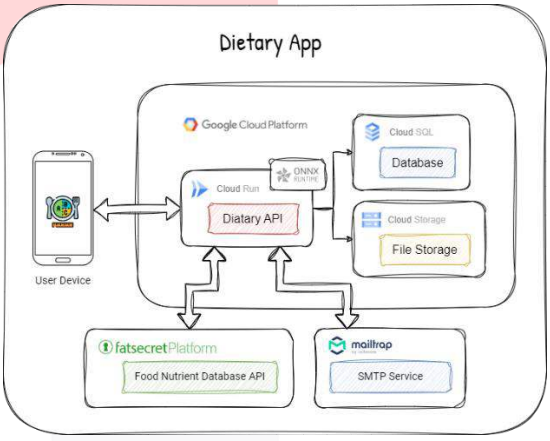
F. Google Cloud Platform untuk Deployment

Google Cloud Platform (GCP) menyediakan serangkaian layanan komputasi awan yang mendukung pengembangan, deployment, dan pengelolaan aplikasi. Dalam proyek ini, GCP digunakan untuk menghosting backend aplikasi konsultasi makanan diet. GCP menawarkan skalabilitas yang tinggi, keandalan, dan berbagai alat yang mendukung integrasi serta automasi deployment. Dengan menggunakan GCP, aplikasi dapat dengan mudah diskalakan sesuai kebutuhan.

III. METODE

Pada bagian *backend*, dibutuhkan rancangan sistem yang efisien untuk mengelola logika aplikasi, basis data, dan integrasi model *machine learning*. *Framework* dan *library* yang digunakan untuk tujuan tersebut antara lain .NET Core (C#) sebagai pondasi utama, serta Entity Framework Core untuk ORM (Object-Relational Mapping), Swashbuckle untuk dokumentasi API, FluentValidation untuk validasi input, Authentication JwtBearer untuk otentikasi, Npgsql untuk PostgreSQL, dan YOLOv8 untuk integrasi model *machine learning*. Aplikasi ini juga menggunakan Application Programming Interface (API) Fat Secret, yang berfungsi untuk menyediakan informasi nutrisi makanan dari gambar yang diunggah oleh pengguna. Algoritma yang digunakan *machine learning* untuk mendeteksi makanan adalah YOLOv8 (*object detection*). Proses pelatihan dataset dilakukan menggunakan Google Colab.

Rancangan dari aplikasi Dietary dapat dilihat pada gambar berikut:



GAMBAR 1
Arsitektur aplikasi Dietary App

Arsitektur aplikasi Dietary tersebut menggambarkan pengguna yang mengakses aplikasi melalui perangkat mobile, kemudian aplikasi akan berkomunikasi dengan server melalui API untuk mengambil dan menyimpan data. Layanan yang digunakan untuk menjalankan API ini adalah (GCP) Google Cloud Platform.

Untuk *database* aplikasi digunakan Cloud SQL untuk menyimpan data seperti informasi pengguna, log makanan, dan data lain yang dibutuhkan aplikasi. Selain itu, digunakan juga Cloud Storage dari GCP sebagai tempat penyimpanan file seperti gambar makanan yang telah di upload oleh pengguna. Untuk memproses model *machine learning,* digunakan ONNX Runtime untuk mendapatkan output nutrisi dari *file* gambar makanan.

Dalam arsitektur ini, terdapat integrasi dengan API pihak ketiga yaitu Fatsecret dan Mailtrap. Fatsecret digunakan untuk mendapatkan informasi detail mengenai kandungan nutrisi dari makanan yang dikonsumsi pengguna. Sedangkan Mailtrap digunakan untuk mengirim email yang dibutuhkan untuk fitur seperti reset password.

A. Konfigurasi Swagger

Swagger merupakan *tools* yang digunakan untuk mendokumentasikan dan menguji API. Terdapat beberapa

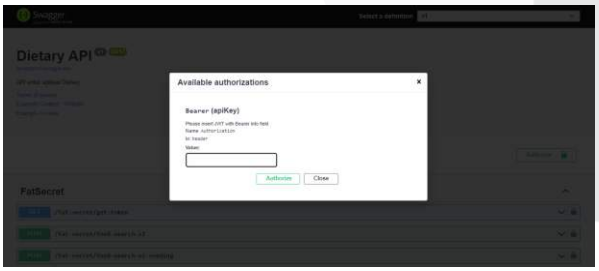
langkah untuk mengintegrasikan swagger pada API. Beberapa diantaranya dapat dilihat pada gambar berikut.

GAMBAR 2
Langkah untuk konfigurasi swagger

langkah pertama yang dilakukan adalah melakukan installasi Swashbuckle.AspNetCore. Ini adalah *library* yang digunakan untuk mengintegrasikan Swagger UI pada proyek ASP.NET Core. Setelah Swashbuckle terinstal, diperlukan informasi dasar tentang API yang akan dibuat seperti judul, versi, dan deskripsi untuk memberikan gambaran umum kepada pengguna API tentang layanan yang disediakan.

Langkah selanjutnya adalah mendefinisikan skema keamanan yang akan digunakan oleh API. Dalam Dietary API, skema yang digunakan adalah "Bearer". Skema ini sering digunakan untuk mengimplementasikan autentifikasi JWT (JSON Web Token). Dalam proses ini ditentukan juga bagaimana API akan mengautentifikasi pengguna menggunakan token Bearer. Hal ini dilakukan untuk memastikan bahwa hanya pengguna swagger terautentikasi yang dapat mengakses endpoint dilindungi. Berikut contoh tampilan endpoint dalam swagger Dietary:

GAMBAR 3
Contoh endpoint `Food`



GAMBAR 4
Implementasi JWT dalam swagger Dietary API

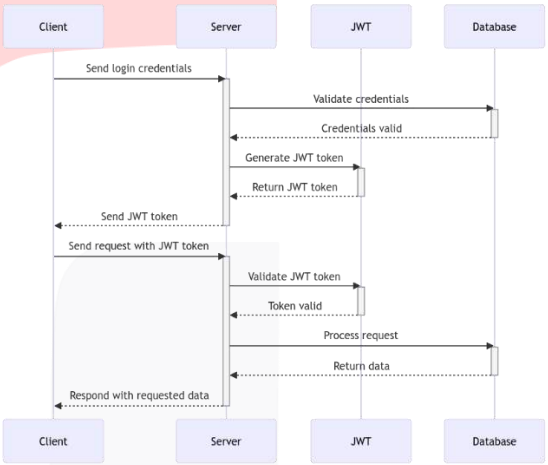
B. Implementasi keamanan JWT

Dalam mengembangkan Dietary API, diimplementasikan JWT untuk melakukan otentikasi dan otorisasi pada setiap request yang diterima API. Cara kerja keamanan pada API ini dimulai dari Pengguna login dengan memasukkan kredensial mereka (nama pengguna dan kata sandi) di sisi client (aplikasi Android), yang kemudian dikirimkan ke server.

Server menerima kredensial dan meminta database untuk memverifikasi apakah nama pengguna dan kata sandi yang diberikan cocok. Jika kredensial valid, server membuat token yang berisi klaim informasi tentang pengguna (misalnya, ID pengguna, nama, email) dan data relevan lainnya seperti waktu kedaluwarsa, kemudian menandatanganinya dengan kunci rahasia untuk memastikan keaslian dan integritasnya.

JWT yang dihasilkan dan ditandatangani dikirim kembali ke klien sebagai bagian dari respons login, dan klien menyimpannya dengan aman dalam local storage pada aplikasi Android. Untuk mengakses *endpoint* yang dilindungi, klien harus melampirkan token di *authorization header* pada *request*. Server yang menerima permintaan tersebut akan mengekstrak token dari *authorization header*, dan memverifikasi *signature* token menggunakan kunci rahasia yang sama untuk memastikan integritasnya seklaigus memeriksa apakah JWT telah kedaluwarsa. Jika token yang diberikan valid, server akan memproses permintaan dari klien dan mengembalikan data sebagai *response*. Namun, jika token tidak valid (seperti kadaluarsa), server mengembalikan respons error 401 tanpa memproses permintaan dari klien.

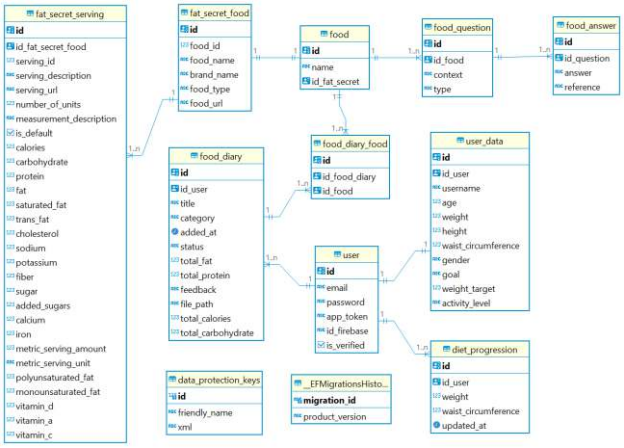
Secara alur proses cara kerja tersebut JWT dapat digambarkan sebagai berikut:



GAMBAR 5
Alur proses cara kerja JWT

C. Integrasi Database

Agar database dapat berinteraksi dengan API secara efisien sekaligus mengurangi kompleksitas penulisan kode SQL secara langsung, maka digunakanlah Object-Relational Mapping (ORM) menggunakan Entity Framework Core (EF-Core). ORM adalah sebuah teknik yang memungkinkan pemrogram untuk bekerja dengan data dalam bentuk objek, tanpa perlu menulis kode SQL. EF-Core adalah salah satu implementasi ORM yang dikembangkan oleh Microsoft untuk platform .NET.



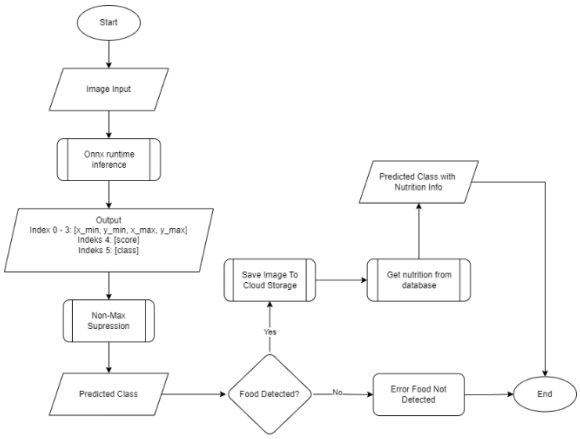
GAMBAR 6
ERD Aplikasi

D. Proses mendapat nutrisi dari gambar

Sistem ini dirancang untuk mendeteksi makanan dari gambar dan memberikan informasi nutrisi terkait. Berikut adalah cara kerjanya:

1. Input Gambar: Pengguna mengunggah gambar makanan yang akan dianalisis.
2. Inference dengan Onnx Runtime: Gambar yang diunggah diproses menggunakan model deteksi objek berbasis Onnx runtime, yang menghasilkan output berupa koordinat bounding box, skor kepercayaan, dan kelas objek.
3. Non-Max Suppression: Proses ini digunakan untuk menyaring bounding box yang berlebihan, mempertahankan bounding box dengan skor tertinggi untuk setiap objek terdeteksi.
4. Prediksi Kelas: Setelah proses Non-Max Suppression, sistem menghasilkan prediksi kelas untuk objek yang terdeteksi.
5. Deteksi Makanan: Sistem memeriksa apakah objek yang terdeteksi adalah makanan. Jika ya, langkah berikutnya dilakukan; jika tidak, sistem memberikan pesan kesalahan bahwa makanan tidak terdeteksi.
6. Simpan Gambar ke Penyimpanan Awan: Jika makanan terdeteksi, gambar akan disimpan ke penyimpanan awan untuk referensi atau analisis lebih lanjut.
7. Ambil Informasi Nutrisi dari Basis Data: Sistem mengambil informasi nutrisi dari basis data berdasarkan kelas objek yang terdeteksi (makanan).
8. Prediksi Kelas dengan Informasi Nutrisi: Sistem menampilkan prediksi kelas beserta informasi nutrisi yang terkait.
9. Selesai: Proses berakhir setelah semua langkah selesai.

Detail lengkapnya dapat dilihat dari gambar berikut:



GAMBAR 7
Alur proses fetching dari FatScret

E. Deploy API

Agar API dapat di akses secara online dari aplikasi android, API perlu di deploy pada penyedia layanan seperti GCP. Pada kasus ini API akan di deploy di Cloud Run menggunakan docker. Proses konfigurasi docker melibatkan beberapa tahap mulai dari menyiapkan dockerfile hingga mengatur pipeline CI/CD.

Tahap pertama yang dilakukan dalam konfigurasi docker yaitu menyiapkan Dockerfile. Dockerfile adalah file teks yang berisi instruksi untuk membangun image Docker. Instruksi ini mencakup langkah-langkah untuk menginstal dependensi, menyalin kode aplikasi, dan mengatur entry point untuk container.

Tahap selanjutnya mengunggah image docker yang telah dibangun ke Google Container Registry (GCR). Kemudian digunakan alat seperti Google Cloud Build, untuk mengatur pipeline CI/CD. Pipeline ini berfungsi untuk otomatisasi proses build dan deployment setiap kali terdapat perubahan kode yang di push ke repository.

IV. HASIL DAN PEMBAHASAN

A. Integration Test

Dalam integration test waktu yang diperlukan untuk menjalankan test akan lebih lama dibandingkan unit test karena membutuhkan koneksi internet dan perangkat Android. Selain itu seperti yang dijelaskan pada bagian skenario integration test, data yang digunakan bukan data tetapi data yang sebenarnya. Berdasarkan test scenario yang telah dibuat sebelumnya, tahap selanjutnya yaitu membuat test script untuk menjalankan testing. Berikut adalah hasil integration test untuk semua skenario yang telah dibuat:

TABEL 1
Hasil Integration Test

No	Skenario	Dietary API *Endpoint*	Hasil
1	Masuk dengan “emailAddress” dan “password” yang benar.	https://api.Dietary.cloud/user/login (POST)	Lulus
2	Masuk dengan “emailAddress” yang benar dan	https://api.Dietary.cloud/user/login (POST)	Lulus

No	Skenario	Dietary API *Endpoint*	Hasil
	"password" yang salah.		
3	Daftar dengan "unregisteredEmailAddress" yang belum terdaftar dan "password" yang valid.	https://api.Dietary.cloud/user/register (POST)	Lulus
4	Daftar dengan "registeredEmail Address" yang sudah terdaftar dan "password" yang valid.	https://api.Dietary.cloud/user/register (POST)	Lulus
5	Menambahkan profil pengguna dengan data-data yang valid seperti "unreigsteredUser name", "age", "weight", "height", "waistCircumfere nce", "gender", "goal", "weightTarget", dan "activityLevel".	https://api.Dietary.cloud/user/user-data (POST) https://api.Dietary.cloud/user/user-data/{userId} (GET)	Lulus
6	Menambahkan profil pengguna dengan "registeredUserna me" yang sudah terdaftar.	https://api.Dietary.cloud/user/user-data (POST)	Lulus
7	Memperbarui profil pengguna dengan data-data yang valid "newUsername", "age", "weight", "height", "waistCircumfere nce", "gender", "goal", "weightTarget", dan "activityLevel"..	https://api.Dietary.cloud/user/user-data (PUT)	Lulus
8	Memperbarui profil pengguna dengan "unavailableUser name".	https://api.Dietary.cloud/user/user-data (PUT)	Lulus
9	Mendeteksi makanan dengan "foodPicture" dan menambahkannya ke dalam diari makanan.	https://api.Dietary.cloud/food/predict (POST) https://api.Dietary.cloud/food-diary (POST)	Lulus
		https://api.Dietary.cloud/food-diary/{foodDiaryId} (GET)	
10	Mendeteksi makanan dengan gambar "nonFoodPicture" .	https://api.Dietary.cloud/food/predict (POST)	Lulus
11	Menghapus diari makanan dengan judul dari "foodDiaryTitle" dari daftar diari makanan.	https://api.Dietary.cloud/food-diary/user/{userId} (GET) https://api.Dietary.cloud/food-diary (DELETE)	Lulus
12	Menampilkan daily nutrition report dan terdapat "caloriesToday"	https://api.Dietary.cloud/food-diary/daily-nutrients (GET)	Lulus

Hasil dari pengujian integrasi yang sudah sesuai antara hasil yang diharapkan dan hasil pengamatan dinyatakan berhasil. Semua fungsi sistem bekerja sesuai dengan apa yang diharapkan. Untuk menghitung akurasi, digunakan persamaan sebagai berikut:

$$Intregration\ Test = \frac{Total\ Berhasil}{Total\ Pengujian} \times 100\%$$

$$Integration\ Test = \frac{12}{12} \times 100\%$$

$$Integration\ Test = 100\%$$

Berdasarkan dari hasil *integration test*, seluruh fungsi dari aplikasi dapat berjalan dengan baik.

V. KESIMPULAN

Berdasarkan penelitian dan pengembangan yang telah dilakukan dapat diambil kesimpulan bahwa sistem sudah dapat terintegrasi dengan baik dengan akurasi sebesar 100%. Selain itu, dari beberapa sub sistem yang dibuat dalam backend sudah terhubung dan dapat bekerja dengan baik sesuai dengan fungsinya. Sistem backend dapat digunakan untuk menyimpan hasil deteksi makanan dan history data makanan. Sehingga dapat membantu orang yang akan melakukan diet dengan melakukan pemantauan makanan.

REFERENSI

[1] C. Rachmi, H. Jusril, I. Ariawan, T. Beal, and A. Sutrisna, "Eating behaviour of Indonesian adolescents: a systematic review of the literature," Public Health Nutrition, vol. 24, pp. s84-s97, 2020. [Online]. Available: https://doi.org/10.1017/S1368980020002876. [Accessed: June 27, 2024].

[2] D. Colozza, "A qualitative exploration of ultra-processed foods consumption and eating out behaviours in an Indonesian urban food environment," Nutrition and Health, vol. 2601060221133897, 2022. [Online]. Available:

https://doi.org/10.1177/02601060221133897. [Accessed: June 27, 2024].

[3] Alodokter, "Konsultasi gizi dan pola makan," Alodokter. [Online]. Available: https://www.alodokter.com/cari-rumah-sakit/gizi-klinik/konsultasi-gizi-dan-pola-makan?page=1. [Accessed: June 27, 2024].

[4] World Health Organization, "Indonesia obesity rates among adults double over past two decades," WHO Indonesia News, Mar. 4, 2021. [Online]. Available: https://www.who.int/indonesia/news/detail/04-03-2021-indonesia-obesity-rates-among-adults-double-over-past-two-decades. [Accessed: June 27, 2024].

[5] R. Ardiyanti, "Obesity and cholesterol factors on hypertension in Indonesia: Data of Indonesian basic health research 2013," Neliti, 2018. [Online]. Available: https://www.neliti.com/publications/275485/obesity-and-cholesterol-factors-on-hypertension-in-indonesia-data-of-indonesian. [Accessed: June 27, 2024].

[6] Goodstats.id, "Kasus diabetes pada usia muda di Indonesia meroket jadi yang terbanyak di ASEAN," Goodstats, Nov. 17, 2022. [Online]. Available: https://goodstats.id/article/kasus-diabetes-pada-usia-muda-di-indonesia-meroket-jadi-yang-terbanyak-di-asean-LVlXV. [Accessed: June 27, 2024].

[7] Goodstats.id, "Tingkat kematian akibat obesitas di Indonesia selama kurun waktu 10 tahun," Goodstats, 2023. [Online]. Available: https://goodstats.id/article/tingkat-kematian-akibat-obesitas-di-indonesia-selama-kurun-waktu-10-tahun-GZ74l. [Accessed: June 29, 2024].

[8] Goodstats.id, "Peringkat 6 di Asia Tenggara, obesitas masih jadi mimpi buruk di Indonesia," Goodstats, 2023. [Online]. Available: https://goodstats.id/article/peringkat-6-di-asia-tenggara-obesitas-masih-jadi-mimpi-buruk-di-indonesia-Gd3Pn. [Accessed: June 29, 2024].

[9] Goodstats.id, "Tingkat obesitas negara Asia Tenggara," Goodstats, 2023. [Online]. Available: https://goodstats.id/infographic/tingkat-obesitas-negara-asia-tenggara-WVgu8. [Accessed: June 29, 2024].

[10] Katadata.co.id, "Ini Penyakit Kronis yang Banyak Diderita Lansia Indonesia," Databoks, May 30, 2022. [Online]. Available: https://databoks.katadata.co.id/datapublish/2022/05/30/ini-penyakit-kronis-yang-banyak-diderita-lansia-indonesia. [Accessed: June 29, 2024].