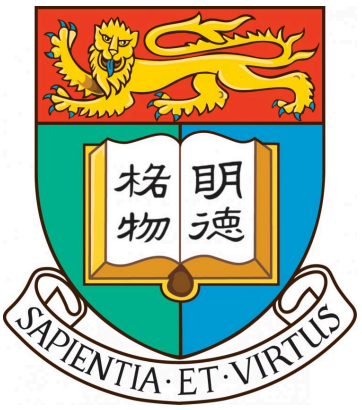

COMP 4801 Final Year Project

Final Report

AI-Driven Meal Evaluation and Real-Time Assistance Application

Tracy Chan

UID: 3036184175

Group: fyp24115

Supervisor:

Zhao Heng Shuang

21 April 2025

Abstract

Micronutrient deficiencies are a pervasive yet often unrecognized global health issue, affecting an estimated one-third of the world's population without obvious clinical symptoms. Conventional dietary assessment methods, such as manual food logging or sensor-dependent imaging, both impose a substantial user burden or require specialized hardware, limiting widespread adoption. This report describes the design, implementation, and evaluation of a mobile platform that integrates lightweight computer vision preprocessing with large language model services to deliver real-time, personalized nutrient feedback from standard smartphone images. We employ YOLOv11 for preliminary food-group classification to minimize token usage in GPT-4o API calls, coupled with native iOS (Swift) and Android (Kotlin) frontends, a Python-based Docker-containerized backend, local persistence via Swift Data and Room, and Firebase Authentication for secure user management. User interfaces including diary, report, and conversational chatbot pages, provide intuitive workflows for capturing meal data, visualizing intake against individualized targets, and receiving tailored dietary insights. Preliminary system deployment demonstrates the feasibility of a cost-effective, hardware-agnostic approach, although the current classification model's limited taxonomy constrains content coverage. Future enhancements will target comprehensive food recognition, depth-based portion estimation, and integration with external health data sources.

Acknowledgements

I would like to express my sincere gratitude to my project supervisor, Dr. Zhao Heng Shuang, for their invaluable guidance, support, and encouragement throughout this project. I am also grateful to the Department of Computer Science, the University of Hong Kong, for providing this wonderful opportunity.

Table of Contents

List of Figures

List of Tables

Abbreviations

AI – Artificial Intelligence

API – Application Programming Interface

APK – Android Package

GPU – Graphics Processing Unit

HTTP – Hypertext Transfer Protocol

IDE – Integrated Development Environment

iOS – iPhone Operating System

JSON – JavaScript Object Notation

SQL – Structured Query Language

UI – User Interface

URL – Uniform Resource Locator

VPS – Virtual Private Server

1. Introduction

1.1. Background

More than half of the world's population consumes inadequate levels of critical micronutrients, such as calcium, iron, and vitamins C and E, yet most are unaware of these deficiencies until significant health issues emerge [1]. Globally, one in three people suffers from at least one micronutrient deficiency, often without obvious symptoms in early stages [2]. In the United States, deficiency rates can reach up to 10% for vitamin B6, vitamin D, and iron among certain demographic groups, highlighting gaps in public awareness and routine screening [3]. Despite this prevalence, traditional nutrient monitoring relies on periodic blood tests and self-report surveys, which many individuals neglect due to cost, access barriers, or low perceived risk [4].

1.2. Existing Approach

Most nutrition tracking apps require manual food logging, where users search for items, estimate portion sizes, and enter details, an error-prone process that can underreport calories by up to 20% and overlook micronutrient intake complexity [5], [6]. Image-based solutions like SnapCalorie use LiDAR depth sensors (available only on select devices) and human reviewers to estimate portions and verify AI outputs, introducing latency, cost, and device dependence [7], [8]. Using SnapCalorie as an example, SnapCalorie's target audiences are mainly health-conscious people focusing on macronutrients (Figure 1). While early trials of AI-powered food recognition demonstrate promising accuracy for macros, few consumer apps deliver real-time, end-to-end analysis of vitamins and minerals without specialized hardware [9], [10].

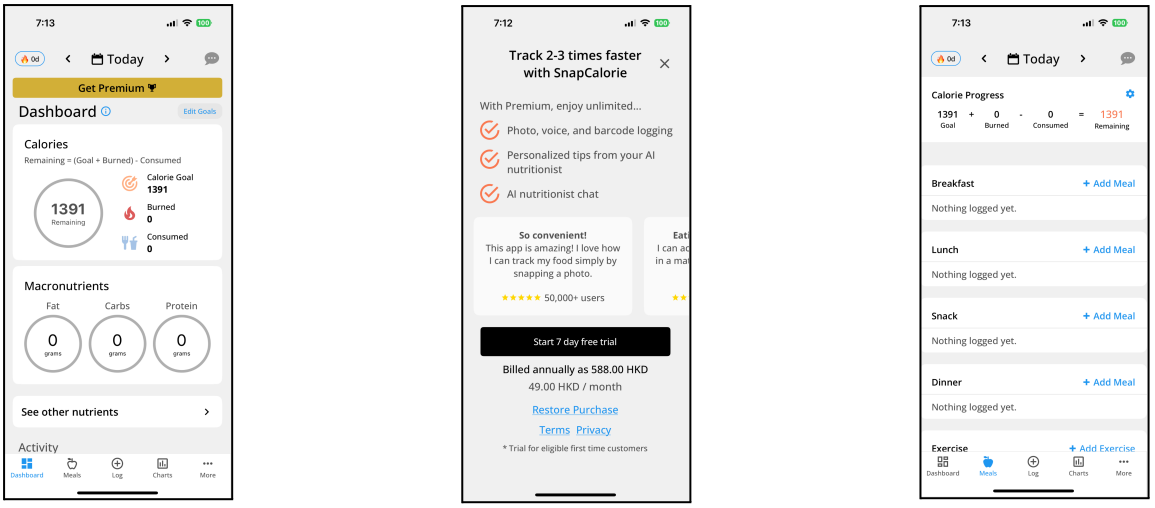

Figure 1. SnapCalorie application targeting health conscious people

1.3 Motivation

A significant portion of the population remains unaware of their dietary deficiencies until advanced stages, underscoring the need for continuous, low-barrier nutrient monitoring [1], [4]. By providing instant AI-driven feedback on both macro- and micronutrient intake from simple food photos, we can raise awareness and prompt corrective action sooner than traditional methods allow [9], [11]. Eliminating reliance on extra sensors or manual review reduces friction, encouraging sustained use and empowering users to recognize and address gaps in their diet before symptoms arise [12], [13].

1.4. Objectives

The objective of this project is to create a mobile application with a unified, intuitive interface tailored for users unfamiliar with nutrient terminology. The application will integrate GPT-4o to identify multiple foods per image, to analyze the user's food intake, and to power a personalized chatbot. Consequently, the app will then produce daily and up to seven-day summaries that flag any nutrient shortfalls, benchmark intake against

recommended ranges, and present alerts and explanations in straightforward, easy-to-understand formats.

1.5. Outline of the report

The remainder of this document is organized into three main sections. Section 2 details the methodology, outlining the development tools, the client–server architecture, frontend and backend implementations, local data persistence, and user authentication flows. Section 3 presents the project results, describing the design and functionality of each user interface page—Authorization, Welcome, Home/Diary, Report, Chat, Settings, and Dark Mode—followed by an examination of challenges encountered and the limitations of the current implementation. Finally, Section 4 offers a conclusion that summarizes the system's achievements and feasibility, and proposes future work to enhance classification accuracy, portion estimation, user experience, and integration with external health data sources.

2. Methodology

In this section, we will explain the development environment, collaboration tools, and the architectural design of the mobile nutrition platform. We first discuss the selection and purpose of each development tool used. Next, we present an overview of the end-to-end system architecture, followed by focused descriptions of the frontend, backend, database, and authorization subsystems.

2.1. Development Tools

The project leveraged four primary development tools to facilitate coding, debugging, version control, and deployment.

XCode

Xcode served as the integrated development environment (IDE) for the iOS client, offering a suite of editors and simulators that streamline Swift code writing, interface design, and performance profiling.

Android Studio

For the Android client, Android Studio was the IDE of choice, providing advanced Kotlin support, real-time layout previews, and built‑in tools for code linting and APK packaging.

Git

To manage source code, coordinate team contributions, and maintain a clear history of changes, we used Git in conjunction with a remote hosting service. Git enabled distributed version control, branch management, and pull‑request workflows, ensuring reproducible builds and simplified integration.

Docker

Docker was utilised to containerise the backend services, encapsulating runtime dependencies and configurations into portable images that could be consistently deployed on development machines and the VPS server without environmental discrepancies.

2.2. Project Architecture

The system follows a client–server model in which the mobile applications act as thin clients, forwarding user requests to a centralized backend. The backend processes incoming requests by interacting with the ChatGPT API, then returns structured responses that the clients render in their native interfaces. This separation of concerns promotes modularity, facilitates independent scaling, and simplifies maintenance.

2.2.1. Frontend

The frontend comprises two native mobile applications: an iOS client written in Swift and an Android client written in Kotlin. On iOS, we used SwiftUI for declarative UI layouts, URLSession for asynchronous network calls, and SwiftImage for initial image handling and preprocessing. On Android, the app leverages declarative XML files for seamless UI composition, Retrofit for HTTP communication, and Coil for image loading and caching. Both codebases implement a repository pattern to abstract data sources, enabling consistent request formatting, response parsing, and local caching of recent analysis results.

2.2.2. Backend

The backend is implemented in Python, using a lightweight web framework to define RESTful endpoints that correspond to app actions (e.g., scan food, fetch report, chat). Each endpoint validates incoming JSON payloads, forwards the relevant data to the GPT API, and applies minimal business logic before returning the API's response. To ensure environmental consistency and simplify deployment, the backend code and its dependencies are packaged into a Docker container, which is then deployed to a Virtual Private Server (VPS). This containerized approach eliminates configuration drift between local development and production environments.

2.2.3. Database

For offline support and to minimize network calls, the application uses each platform's native local persistence layer. On iOS, Swift Data provides an object graph and persistence framework that stores user profiles, meal logs, and cached nutrient analyses in a SQLite store under the hood. On Android, we employ Room, which offers a type-safe abstraction over SQLite, enabling compile-time verification of SQL queries and seamless mapping between entities and database tables. Both solutions support migrations and encrypted storage to protect sensitive user data.

2.2.4. Authorization

User identity and access control are handled via Firebase Authentication, which offers secure, managed sign-up and sign-in flows. We configured Firebase to support Google Sign-In, which enabled users to authenticate using their existing Google accounts. Firebase handles token issuance, refresh, and revocation, and the mobile clients attach the issued ID tokens to each backend request, ensuring that only authenticated sessions can access user-specific endpoints.

2.3. Timetable

Date	**Milestones**	**Status**
September 2024	Detailed Project Plan	Completed
October 2024	UI/UX Design Basic Prototype	Completed
November 2024	Frontend Development, API Setup	Completed
December 2024	Frontend Development, Backend Development	Completed
January 2025	Frontend Development, Backend Development, First Presentation, Interim report	Completed
February 2025	Frontend Development, Backend Development	Completed
March 2025	Overall System Testing	Completed
April 2025	Final Report	Completed

Table 1. Project Schedule

3. Project Results

Following the methodology discussed in section 2, this section showcases the results that have been produced, with challenges faced and limitations of the project.

3.1 Platform Design

3.1.1 Authorization

The Authorization page appears upon first launch or after a user logs out. It offers options to sign in or register using Firebase Authentication, with a Google Sign-In button for streamlined access. Users can log in through their Google email account, and successful authentication transitions users to the Welcome page. All authentication tokens are securely stored in the client's key chain (iOS) or encrypted shared preferences (Android) to maintain session state across app restarts.

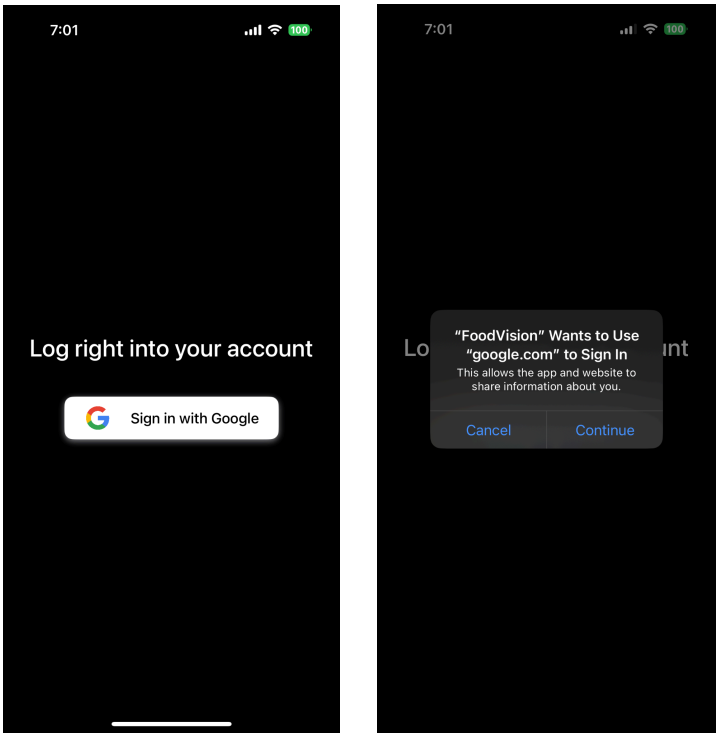

Figure 2. Authorization Page

3.1.2 Welcome Page

The Welcome page serves as the app's landing screen for authenticated users who sign into the app for the first time. It features a brief scanning workflow that takes the user's basic data and stores it in the database. A central "Continue" button guides users to the Home page. The design employs a clean layout with illustrative pictures and concise text.

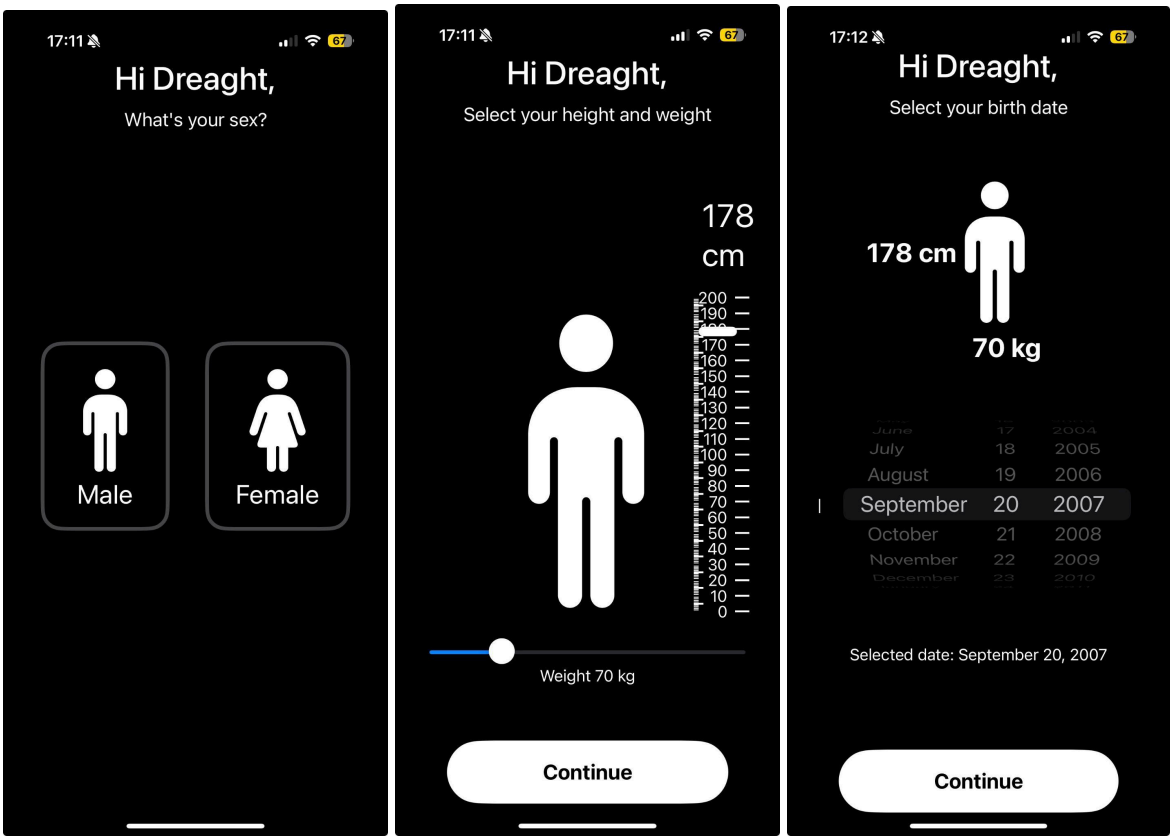

Figure 3. Welcome Page

3.1.3 Home Page

Also referred to as the Diary page, the Home page allows users to capture and review food entries. Daily nutrition details are summed up and are displayed at the top below the date of the page. Below this, three scrollable lists display the meals logged, with each section having its own “add” button. The add button opens up the camera and gives access to the gallery. Users are also able to swipe to delete the meals logged under each category of breakfast, lunch, and dinner.

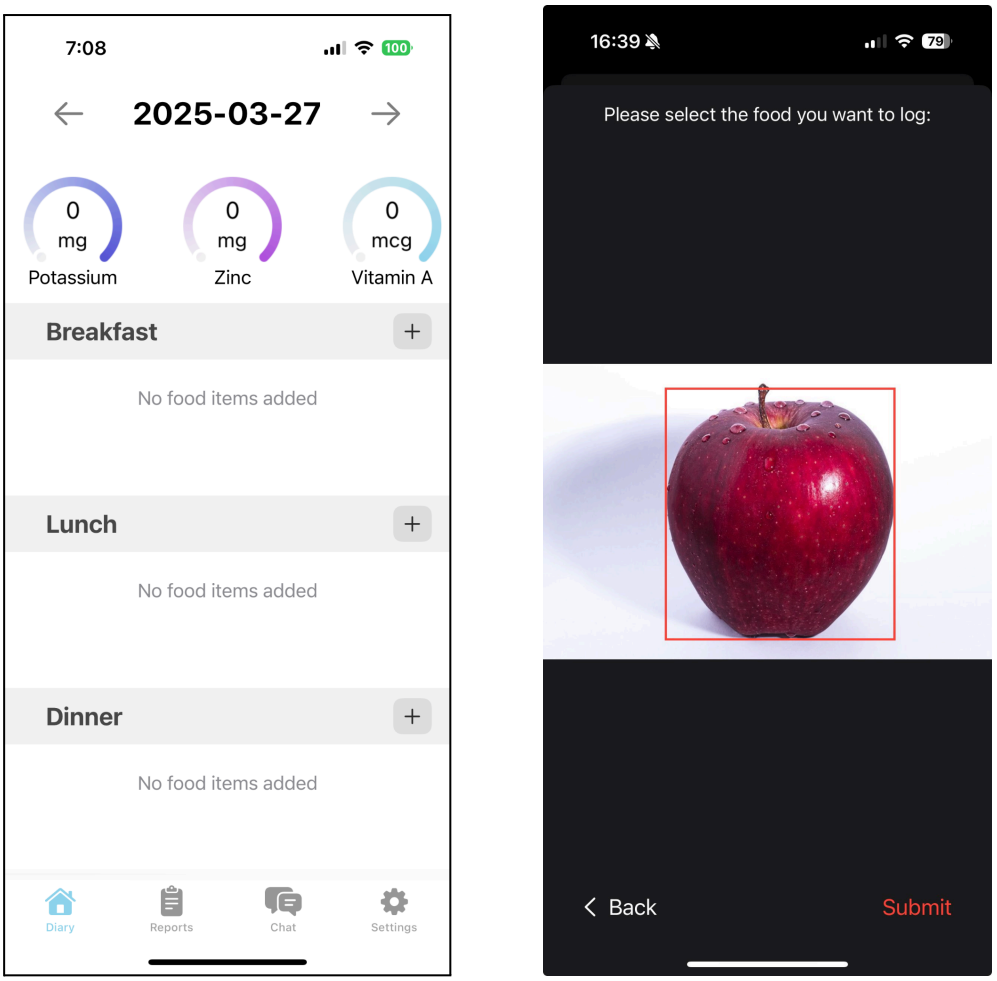

Figure 4. Home Page

3.1.4 Report Page

The report page lets users choose up to seven consecutive days to generate a report for. Users are then able to visualize AI's analysis of their daily intake of nutrients based on their personal information. An overall score would be given to the user, along with a comment from AI on their diet. All thirty nutrients are displayed in the report; deficiencies below 80% and over 120% are highlighted in red, while those within the range are highlighted in green. The report will be available to be download as an image to the user's phone.

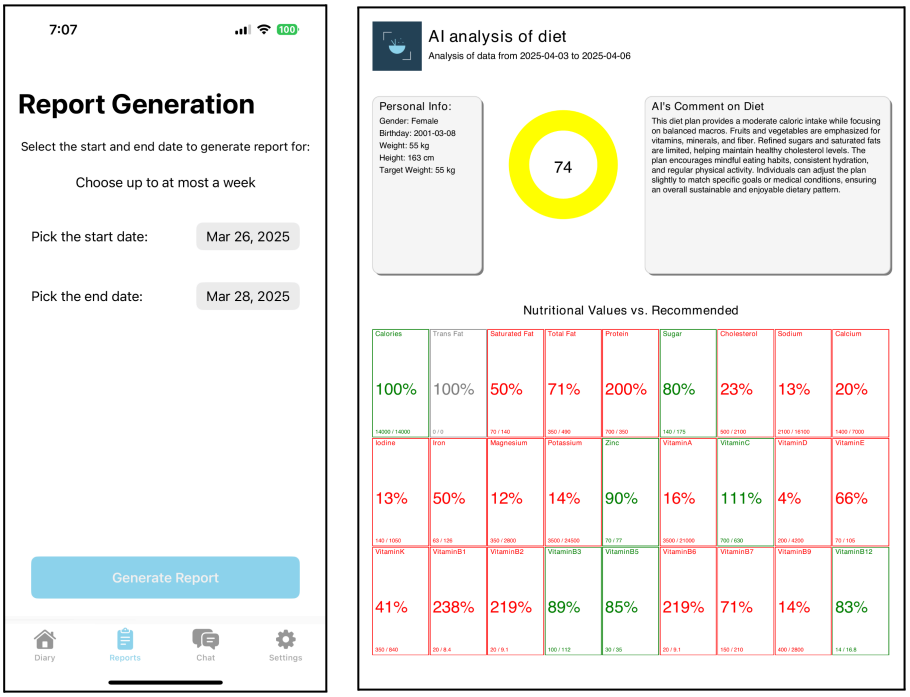

Figure 5. Report Page

3.1.5 Chat Page

The Chat page hosts the GPT-4o-powered chatbot, presented as a conversational interface. Previous messages appear in a vertically scrollable view, with user queries and bot responses to be viewed by their profile picture. The input field at the bottom supports text entry. The assistant takes information for the past week of the user's logs and the user's personal information.

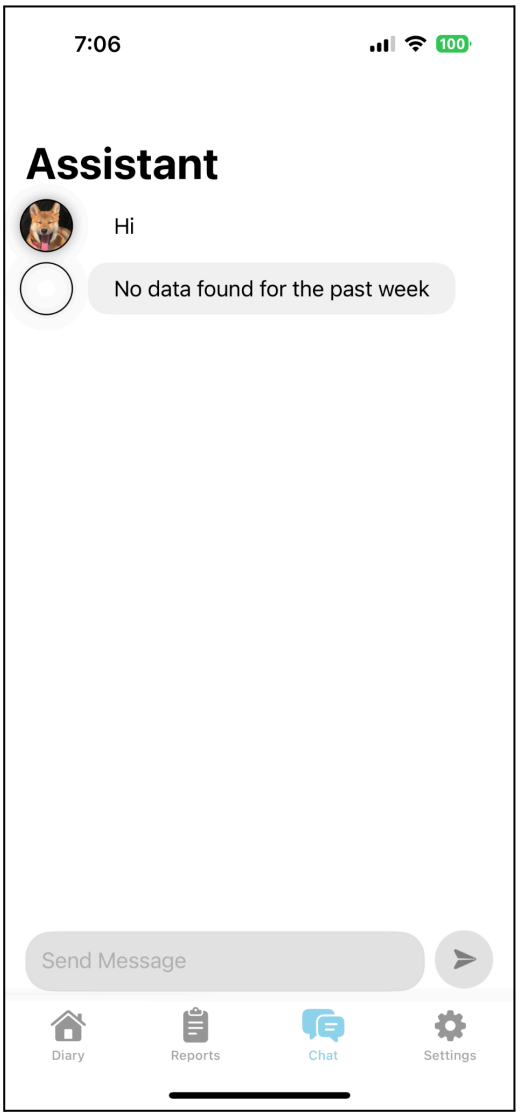

Figure 6. Chat Page

3.1.6 Settings Page

The Settings page allows users to manage their personal health data and account settings. At the top, the user's profile picture and display name, retrieved from their Google account via Firebase Authentication, are prominently displayed, offering a sense of personalization. Below this, editable fields allow users to input or update their current weight, goal weight, height, and birthdate. These metrics are used by the app to tailor nutrient targets, report and chatbot feedback. The logout function is located at the bottom of the page, accompanied by the user's authenticated email address for clarity. When selected, it signs the user out securely and clears session tokens from the device.

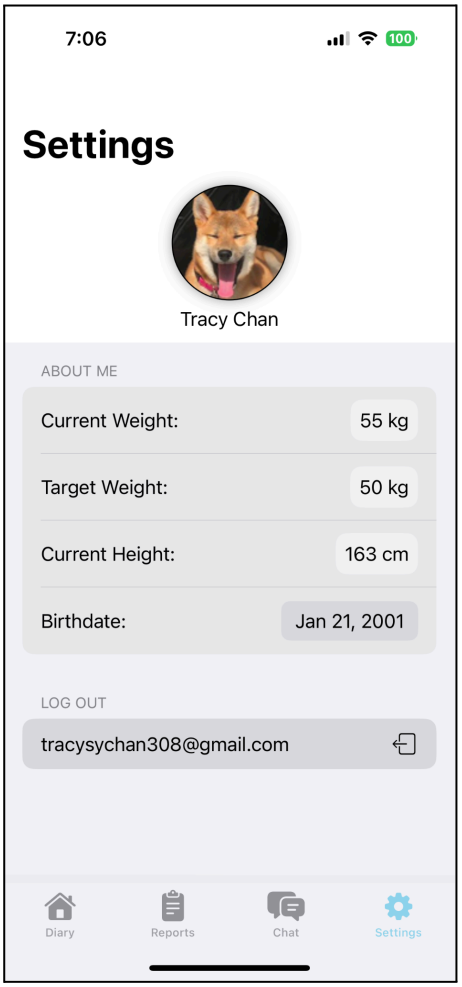

Figure 7. Settings Page

3.1.7 Dark Mode

To accommodate user preferences and reduce eye strain, the app detects system-level colour scheme settings and automatically switches between light and dark themes. All UI components, including backgrounds, text, icons, and charts, adjust their colour palettes to maintain contrast and readability.

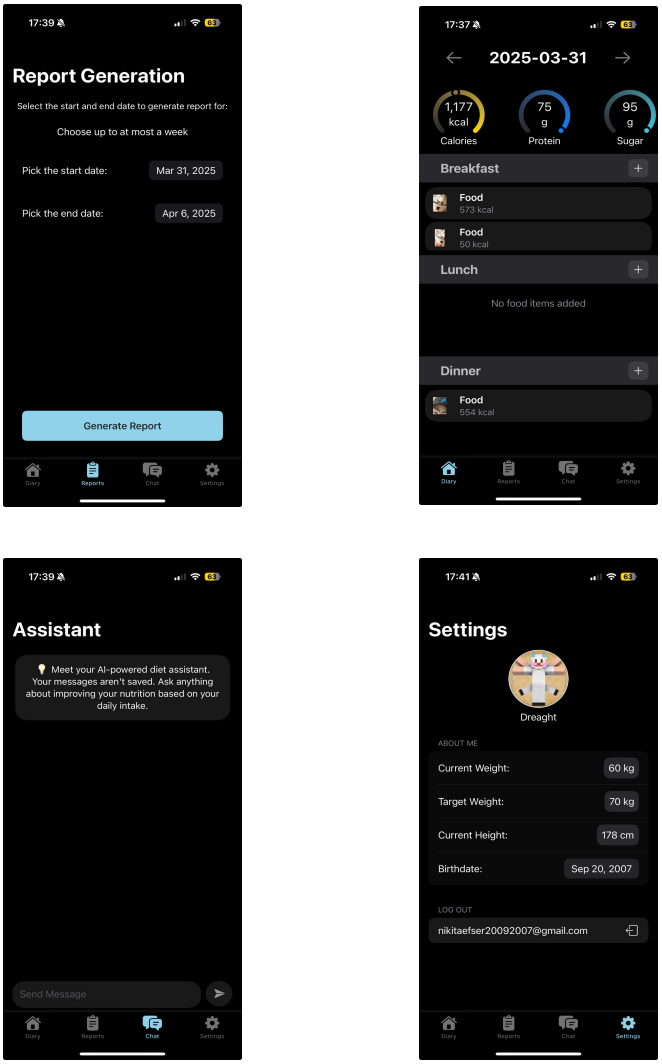

Figure 8. App in Dark mode

3.2 Challenges

One of the primary challenges encountered during development involved the handling of food images for nutritional analysis using GPT-4o. Directly uploading user images to the GPT model was found to be highly inefficient, as it consumed a large number of tokens per request. This significantly impacted both performance and cost. To avoid this issue, we introduced a preprocessing step using the YOLOv11 (You Only Look Once version 11) object detection model. This model is employed to classify the image into general food groups before passing the classified result to GPT-4o for further nutritional analysis. This approach substantially reduced the number of tokens required, thereby optimizing both latency and budget constraints. Additionally, ensuring visual and functional consistency between the iOS and Android applications presented difficulties due to native UI conventions and component differences, necessitating extra design iterations and conditional styling to maintain a uniform look and feel.

3.3 Limitations

While the YOLOv11-based classification step proved to be an effective solution, it introduced its own limitations. Specifically, the model is only trained to recognize a limited set of food groups, which restricts the range and accuracy of the nutritional analysis for diverse and complex meals. A more comprehensive classification model would ideally be used; however, implementing such a model requires a high-performance backend infrastructure equipped with GPU acceleration. Given the project's constrained budget and timeline, it was not feasible to either acquire the necessary hardware or allocate time for training a more advanced model capable of recognizing a broader spectrum of foods. As a result, the application currently provides an informative but not exhaustive analysis of dietary intake.

4. Conclusion and Future Works

4.1. Conclusion

The development and deployment of this mobile nutrition platform demonstrate the feasibility and value of leveraging modern AI techniques to bridge the awareness gap in dietary intake. By integrating GPT-4o for natural language understanding and employing YOLOv11 for initial image classification, the application is able to provide users—many of whom previously lacked insight into their micronutrient status—with rapid, on-device feedback after photographing their meals. The native Swift and Kotlin frontends deliver a responsive, intuitive user experience, while the Python-based, containerized backend ensures consistent, secure communication with the OpenAI API. Local persistence via Core Data and Room underpins offline support and efficient caching, and Firebase Authentication facilitates streamlined, secure user onboarding. User interface components—from the Diary page to the AI chatbot—work in concert to present nutrient summaries, identify deficiencies, and offer personalized insights in straightforward language. Although constrained by budget and token consumption challenges, the project successfully illustrates a cost-effective, end-to-end solution for dietary monitoring that requires no specialized hardware beyond a standard smartphone camera. Overall, the platform validates the potential of combining lightweight computer vision with large-language-model services to enhance public understanding of nutrition and to encourage healthier eating behaviors among a population segment that may be unaware of their own dietary shortfalls.

4.2. Future Works

Looking ahead, several avenues exist to expand the platform's capabilities and accuracy. First, migrating classification to a more comprehensive food-recognition model—trained on an extensive, diverse dataset—would broaden the range of detectable foods beyond the limited categories currently supported by YOLOv11. Achieving this will likely require GPU-accelerated infrastructure or partnerships that can underwrite the associated computational costs. Second, integrating real-time portion-size estimation using monocular depth estimation techniques could replace heuristic volume approximations and further reduce dependency on coarse group classifications. Third, incorporating longitudinal user studies and A/B testing will provide empirical data on engagement patterns, usability, and behavior change efficacy, informing iterative improvements to both UI design and chatbot dialogue strategies. Additionally, expanding the nutrient database to include specialized dietary regimes (e.g., ketogenic, vegan, low-FODMAP) and localizing content for different cultural cuisines will increase relevance for a broader user base. Finally, enabling interoperability with wearable health devices and third-party fitness platforms could enrich the context for nutrient recommendations and support a more holistic view of user health metrics. Collectively, these enhancements will advance the platform toward a robust, scalable solution for personalized nutrition monitoring and guidance.

5. References

[1] “Billions worldwide consume inadequate levels of micronutrients critical to human health,” *Harvard T.H. Chan School of Public Health*, Aug. 29, 2024. https://hsph.harvard.edu/news/billions-worldwide-consume-inadequate-levels-of-micronutrients-critical-to-human-health

[2] X. Han, S. Ding, J. Lu, and Y. Li, “Global, regional, and national burdens of common micronutrient deficiencies from 1990 to 2019: A secondary trend analysis based on the Global Burden of Disease 2019 study,” *eClinicalMedicine*, vol. 44, p. 101299, Feb. 2022, doi: https://doi.org/10.1016/j.eclinm.2022.101299.

[3] CDC, “About the Second Nutrition Report,” *CDC’s Second Nutrition Report*, May 14, 2024. https://www.cdc.gov/nutrition-report/about/index.html

[4] J. Bird, R. Murphy, E. Ciappio, and M. McBurney, “Risk of Deficiency in Multiple Concurrent Micronutrients in Children and Adults in the United States,” *Nutrients*, vol. 9, no. 7, p. 655, Jun. 2017, doi: https://doi.org/10.3390/nu9070655.

[5] J. H. West, L. M. Belvedere, R. Andreasen, C. Frandsen, P Cougar Hall, and B. T. Crookston, “Controlling Your ‘App’etite: How Diet and Nutrition-Related Mobile Apps Lead to Behavior Change,” *JMIR mHealth and uHealth*, vol. 5, no. 7, p. e95, Jul. 2017, doi: https://doi.org/10.2196/mhealth.7410.

[6] S. Volz, A. Ward, and T. Mann, “Eating up cognitive resources: Does attentional consumption lead to food consumption?,” *Appetite*, vol. 162, p. 105165, Jul. 2021, doi: https://doi.org/10.1016/j.appet.2021.105165.

[7] J. Gilliland, R. Sadler, A. Clark, C. O’Connor, M. Milczarek, and S. Doherty, “Using a Smartphone Application to Promote Healthy Dietary Behaviours and Local Food Consumption,” *BioMed Research International*, vol. 2015, pp. 1–11, 2015, doi: https://doi.org/10.1155/2015/841368.

[8] V. Guan *et al.*, “A Novel Mobile App for Personalized Dietary Advice Leveraging Persuasive Technology, Computer Vision, and Cloud Computing: Development and Usability Study,” *JMIR Formative Research*, vol. 7, no. 1, p. e46839, Aug. 2023, doi: https://doi.org/10.2196/46839.

[9] S. S. Coughlin, M. Whitehead, J. Q. Sheats, J. Mastromonico, D. Hardy, and S. A. Smith, “Smartphone Applications for Promoting Healthy Diet and Nutrition: A Literature Review,” *Jacobs journal of food and nutrition*, vol. 2, no. 3, p. 021, 2015, Accessed: Apr. 20, 2025. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC4725321

[10] J. Choi, C. Chung, and H. Woo, “Diet-Related Mobile Apps to Promote Healthy Eating and Proper Nutrition: A Content Analysis and Quality Assessment,” *International Journal of Environmental Research and Public Health*, vol. 18, no. 7, p. 3496, Mar. 2021, doi: https://doi.org/10.3390/ijerph18073496.

[11] “PATIENT SAFETY e Cognitive Load Theory and Its Impact on Diagnostic Accuracy,” May 2024. Accessed: Apr. 20, 2025. [Online]. Available: https://www.ahrq.gov/sites/default/files/wysiwyg/diagnostic/resources/issue-briefs/dxsafety-cognitive-load-theory.pdf

[12] A. M. Kappattanavar, P. Hecker, S. Moontaha, N. Steckhan, and B. Arnrich, “Food Choices after Cognitive Load: An Affective Computing Approach,” *Sensors*, vol. 23, no. 14, p. 6597, Jan. 2023, doi: https://doi.org/10.3390/s23146597.

[13] V. Drake, “Micronutrient Inadequacies in the US Population: an Overview,” *Linus Pauling Institute*, Apr. 20, 2018. https://lpi.oregonstate.edu/mic/micronutrient-inadequacies/overview