

Nutrieye

Kussay Talal Alkindy, Grace Luo, Haku Altanpurev

Alkindy@wisc.edu, gpluo@wisc.edu, altanpurev@wisc.edu

Git usernames: usaiKodes, gracepluo, nene0219


[Link to the Nutrieye demo video](#)

There is permission to use this video to market the class to a broad audience in future years.

[backend](#) & [main](#) github links

Nutrieye

Kussay Talal Alkindy, Grace Luo, Haku Altanpurev
Alkindy@wisc.edu, gpluo@wisc.edu, altanpurev@wisc.edu
Git usernames: usaiKodes, gracepluo, nene0219



CS407 Fall 2024 - Group Project

Introduction

NutriEye is a mobile nutrition tracker that allows users to quickly analyze the nutritional value of their meals by simply taking a photo or scanning a barcode.

NutriEye makes it simpler to keep track of caloric intake quickly.

natural users: College students, travelers

Design: Key Features

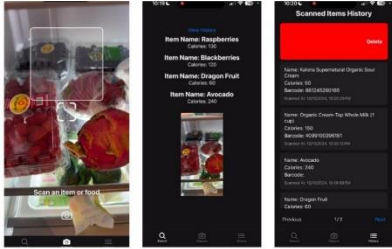
- Image Recognition
- Capture meal photos to analyze nutritional information.
- Automated product lookup via Google search
- Expo-Camera Integration
- Access to camera function for photos and barcode scanning.
- Server-Side Processing

Design: User Interface and Technology Stack

- Design is simple and user friendly
- Minimal effort to use application

Features beyond those in the Labs:

- Advanced Image Recognition
- Dynamic Database Integration
- Barcode Scanning with Custom Processing



Development Team

Kussay: Set up the API for photo recognition and result screen, worked on the http server, Fixed bugs relating to the back end process, Surveyed people for feedback, worked on the poster.

Grace: Set up nodejs, http server set up, implement the new server api to detect items, worked on the poster, gathered resources, gained survey feedback for the most recent feedbacks online & in person

Haku: Created the initial version of the app with the automated barcode detection scanning and Expo camera. Added custom ChatGPT tool to lookup barcodes in Google. Productionized the backend and deployed it on the cloud along with logging.

Challenges and Solutions

- Struggled in history feature, but managed to make it

Not included:

- Bigger database for foods from all over the world.
- More colorful/exciting app design (based on feedback).
- This is due to lack of a 4th member leading to more time constraints

User Feedback/ Future Work

- More minorUI improvements like alignment and feature prioritization

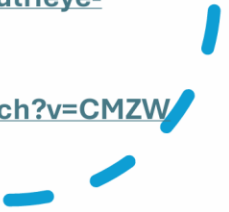
Useful Links

- [backend](#) & [main](#)
- [Link to the Nutrieye demo video](#)

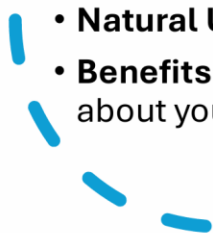
Poster ^^



NutriEye

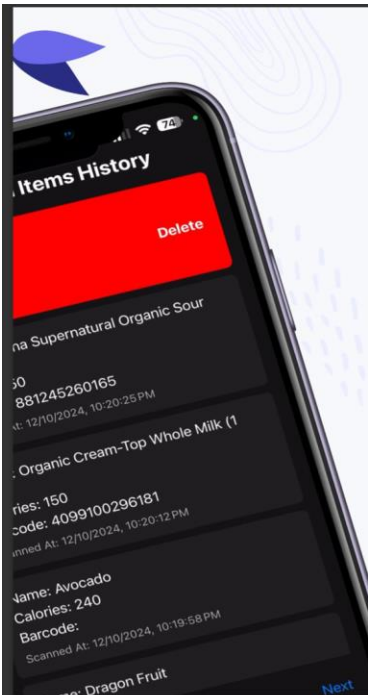
- **Kussay Talal Alkindy**
 - Email: Alkindy@wisc.edu
 - GitHub ID: usaiKodes
 - **Grace Luo**
 - Email: gpluo@wisc.edu
 - GitHub ID: gracepluo
 - **Haku Altanpurev**
 - Email: altanpurev@wisc.edu
 - GitHub ID: nene0219
 - **GitHub Repository:**
 - <https://github.com/cs407uw/nutrieye>
 - <https://github.com/cs407uw/nutrieye-backend>
 - **Project Videos:**
 - <https://www.youtube.com/watch?v=CMZWNOHLwPA>
- 

What is your app?

- **NutriEye** is a mobile app that helps people quickly check the nutritional value of their meals. All you need to do is take a picture or scan a barcode!
 - **Why It's Needed:** It can be hard to keep track of what you're eating. NutriEye makes it simple.
 - **Natural Users:** College students, travelers.
 - **Benefits:** Instant meal analysis, and gives helpful information about your food
- 

Design & Implementation

- **Design Overview:**
- **Frontend:** Built with Expo React Native for cross-platform compatibility focusing on IOS. Utilizes SQLite for local scanning history.
- **Backend:** Utilizes a Node.js server for processing and data retrieval.
- **APIs Integrated:**
 - OpenAI GPT API for nutritional estimation.
 - Google Search API for product information.



Design & Implementation

- **Unique Mobile Characteristics:**
- **Image Recognition:**
 - Capture meal photos to analyze nutritional information.
 - Used advanced AI models for accurate information.
- **Barcode Scanning:**
 - Scan packaged food barcodes for instant results.
 - Automated product lookup via Google search.**Implementation Details:**
- **Expo-Camera Integration:**
 - Access to camera function for photos and barcode scanning.
- **Local SQLite Database**
 - Efficient local history of scanned items.
 - Ability to delete individual history entries.
- **Server-Side Processing:**
 - Offloads heavy computations to the backend.
 - Enhances app performance and accuracy.

Features Beyond the Lab

- **Advanced Image Recognition:**
- NutriEye's ability to process images of meals and extract nutritional details extends beyond **Lab 9 (Cloud Storage and ML Kit)**. The ML Kit in Lab 9 may involve simple machine learning tasks, but NutriEye implements sophisticated backend AI processing for detailed nutritional analysis.
- **Lower-level SQLite Operations:**
- While **Lab 5 (Persistent Storage)** uses RoomDB, we interact directly with the SQLite database for cross platform compatibility across IOS and Android.
- **Dynamic Database Integration:**
- While **Lab 6 (APIs and Maps)** introduces API usage, NutriEye's backend integration with OpenAI's API for meal analysis is a more complex and tailored implementation for specific use cases.
- **Barcode Scanning with Custom Processing:**
- NutriEye's barcode scanning feature goes beyond **Lab 4 (Non-UI Thread Processing)** by implementing automated lookups and integration with nutritional databases.



Features Different From Original Design

- **Features Left Out Due to Constraints:**
- Favorites Feature:
 - Ability to save commonly scanned foods.
 - Deferred to prioritize core functionalities.
- A bigger database for foods from all over the world.
- A more colorful and exciting app design (based on feedback).
- An offline mode so it works without the internet.
- Why Not?: We ran out of time to add everything. And the lack of a 4th member

Reflections and Next Steps

- **Interesting Insights:**
- **User Feedback Impact:**
 - Directly influenced UI improvements and feature prioritization.
- **Backend Advantages:**
 - Improved accuracy and allowed for complex computations not feasible on mobile devices.
- **Next Steps:**
- **Implement Favorites Feature:**
 - Enhance user convenience and app personalization.
- **Expand Food Database:**
 - Include more international cuisines and dietary information.
- **Optimize Performance:**
 - Speed up server responses and streamline processing.
- **User Testing and Feedback:**
 - Conduct broader testing to refine UI/UX further.



^^slides/poster components