

DSCI 551

Final Project Report

Team 31

Xingyi Sun

NutriVision: Efficient Food Database Management System

Introduction

In an era where health consciousness is rising and dietary preferences are rapidly evolving, access to reliable and detailed nutritional information is more crucial than ever. NutriVision, a cutting-edge database management system, is designed to meet this need by providing easy access to extensive nutritional data for a diverse range of foods. Motivated by the challenge of making complex nutritional information readily available and comprehensible, NutriVision simplifies the process of retrieving and analyzing food data, supporting better dietary planning and informed nutritional decision-making.

This system is specifically tailored for nutritionists, dieticians, health enthusiasts, and the general public who are proactive about managing their health through diet. By leveraging NutriVision, users can explore nutritional details effortlessly, aiding in the creation of diets that cater to specific nutritional needs and preferences. Furthermore, NutriVision aims to become an indispensable tool for educational purposes, offering valuable insights that enable users to understand and optimize their dietary habits with precision and confidence.

Through NutriVision, we are committed to enhancing nutritional awareness and promoting a healthier society by ensuring that accurate and accessible nutritional data is just a few clicks away.

Planned Implementation

1. Database Design:

I will use MongoDB to structure the nutritional dataset effectively, utilizing strategic partitioning to optimize data management. The database is segmented into two collections: **oddIdFoods** and **evenIdFoods**. This segmentation is determined by the parity of the sum of Unicode values of **ndbNumber**, which acts as a unique identifier for each food item. This method of distributed data management is designed to improve query performance by balancing the workload across two servers, thereby enhancing the system's efficiency and responsiveness.

2. Backend Development:

Use Python serve as the backend to connect and interact with MongoDB. A script named “db.py” defines a “Connection” class that takes a database name as input and sets up a connection using MongoDB’s default settings. There are two instances for the class: “oddIdFoods” and “evenIdFoods” which are connect to the two databases.

3. Data Management UI:

The application's core functionalities are centered around CRUD (Create, Read, Update, Delete) operations, enabling data managers to efficiently manage nutritional information stored in the database. The Create function will distribute data into the two-dataset based on the sum of Unicode values of the ‘id’.

4. User Application

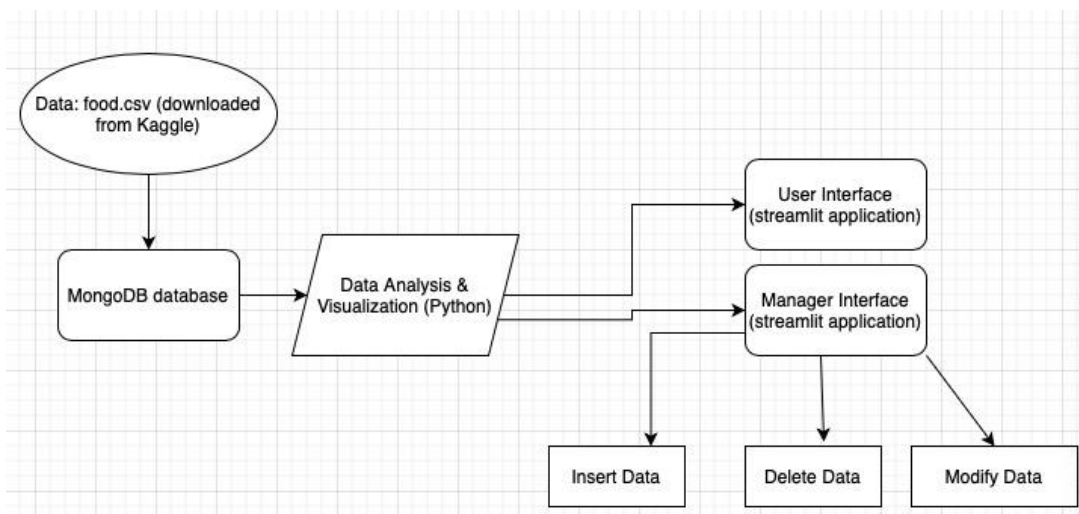
A user friendly interface is planned to be developed. The function will include searching for food items based on various criteria and view detailed notional information. I will be a well-formatted web page using python Steamlit.

Architecture Design

System Architecture Overview

The NutiVision system is designed with a clear separation into three primary layers: the fronted interface, backend server logic, and the MongoDB database layer. Each layer is purpose-built to handle specific aspects of the system’s operations, ensuring a sooth and efficient workflow from user input to data storage and retrieval.

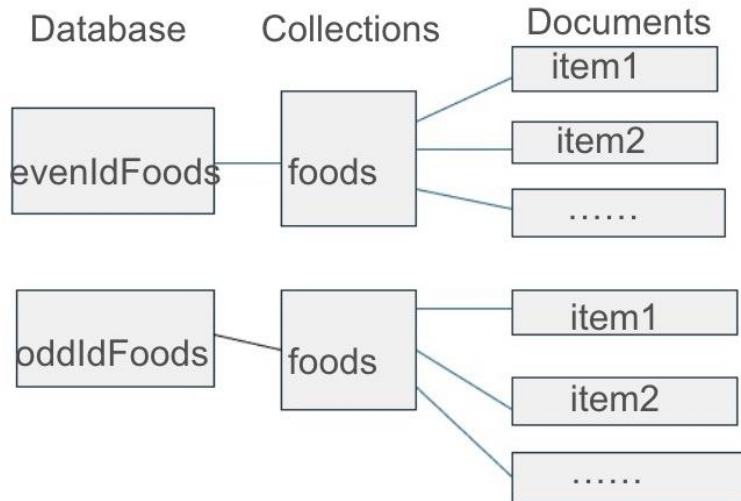
Flow Diagram:



Description of the Flow Diagram:

1. Database Layer (MongoDB)
 - a. Two separate databases: 'oddIdFoods' and 'evenIdFoods'.
 - b. Each database has one collection: 'foods'.
 - c. Stores data based on the sum of Unicode values of the IDs (odd or even).

MongoDB Database Architecture graph:



2. Frontend (Streamlit Web Interface):
 - a. Users interact with this layer directly through a web browser. It provides interactive elements such as tabs, form, and input fields for executing CRUD operations and visualizing nutritional data.
 - b. Streamlit serves both as the fronted framework and part of the backed logic, handling user inputs and rendering the user interface dynamically.
3. Backend (python and Streamlit):
 - a. The backend logic is integrated within the Streamlit application, blurring traditional distinctions between fronted and backend. It processes user requests, interfaces with the database, and manages business logic.
 - b. Python scripts are utilized for connecting the MongoDB, determining the appropriate database based on the sum of Unicode values, and executing database operations such as insert, update, and delete.

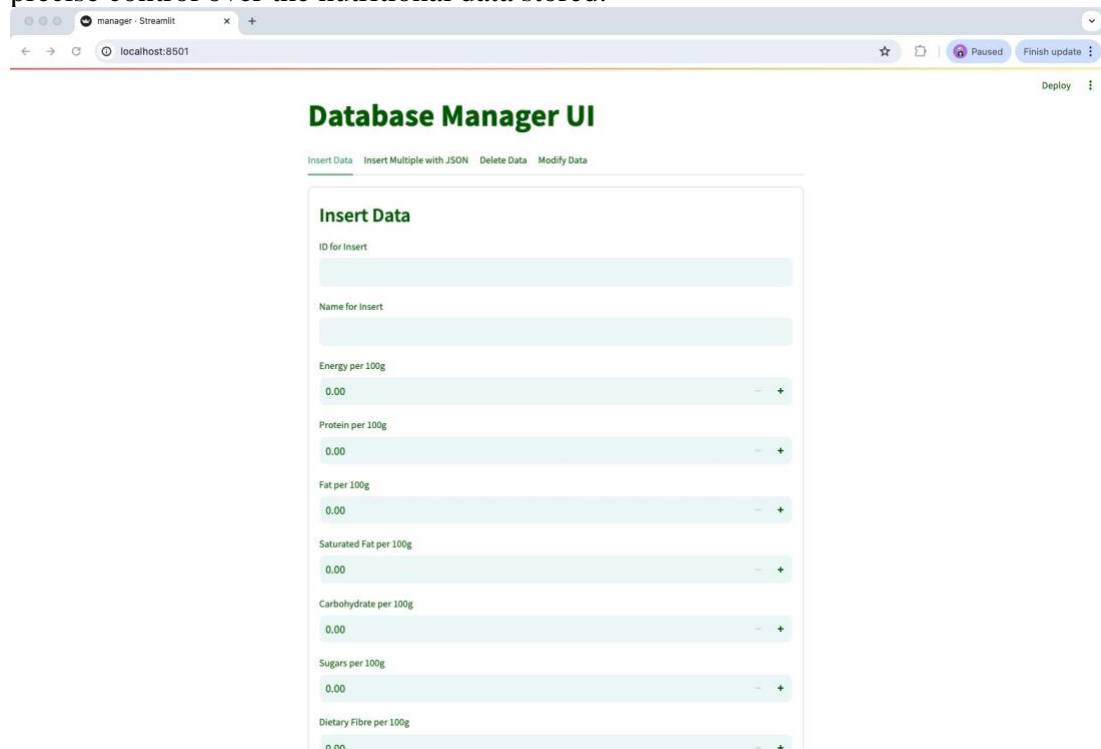
Implementation

Manager interface

1. Functionalities

a. Single Data Insertion

The "Insert Data" functionality within the Database Manager UI of the NutriVision system allows users to add single food entries directly into the database. This is facilitated through a straightforward form interface in Streamlit, where users input unique identifiers and nutritional information such as protein, fat, carbohydrates, and more. Each submission is processed to calculate the sum of Unicode values of the item ID, determining the appropriate database (either `oddIdFoods` or `evenIdFoods`) based on whether the sum is odd or even. This ensures that the data is stored efficiently across two separate collections. Upon submission, the system checks for existing records to prevent duplicate entries and confirms successful data insertion or notifies the user of any errors. This function is crucial for maintaining an up-to-date and accurate database, allowing for precise control over the nutritional data stored.



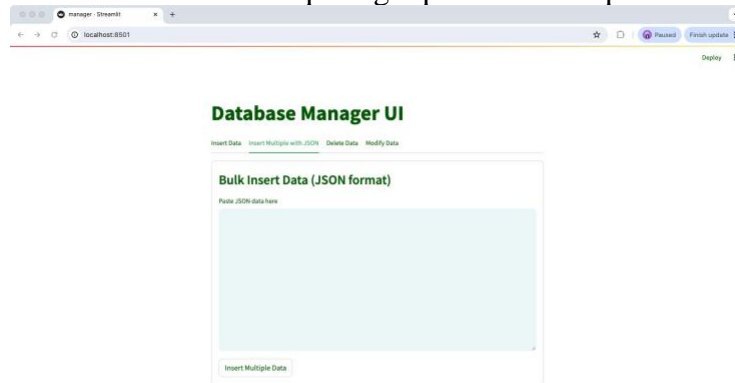
The screenshot displays the 'Database Manager UI' in a web browser. The interface has a green header with the title 'Database Manager UI'. Below the header, there are four tabs: 'Insert Data', 'Insert Multiple with JSON', 'Delete Data', and 'Modify Data'. The 'Insert Data' tab is active. The form contains the following fields:

- ID for insert: A text input field.
- Name for insert: A text input field.
- Energy per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Protein per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Fat per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Saturated Fat per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Carbohydrate per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Sugars per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.
- Dietary Fibre per 100g: A numeric input field with a value of 0.00 and minus/plus buttons.

2. Bulk Data Operations

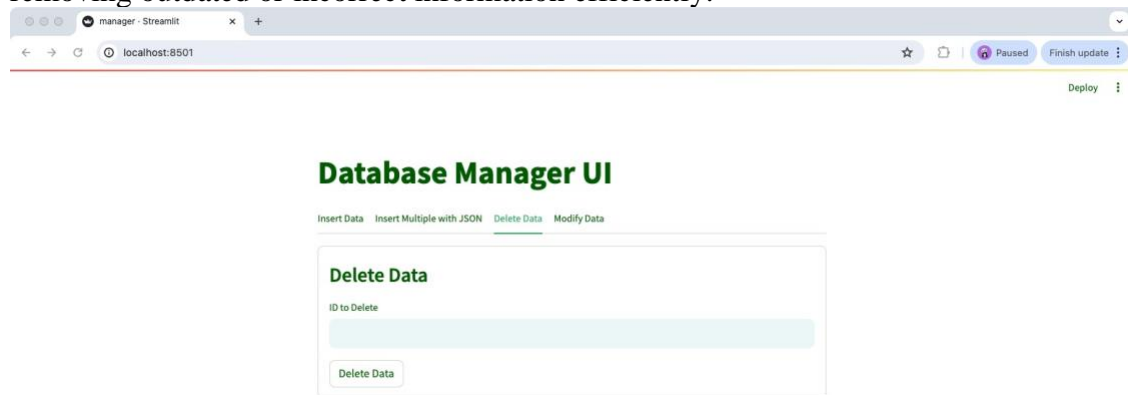
The "Insert Multiple with JSON" feature in the NutriVision system provides a robust mechanism for bulk data entry, enabling users to add multiple food items simultaneously into the database. This functionality is particularly useful for efficiently managing large datasets, as users can input JSON formatted data into a text area within the Streamlit interface. Upon submission, the system parses the JSON to validate its format and structure, ensuring it consists of a list of dictionary entries. Each entry is then routed to either the `oddIdFoods` or `evenIdFoods` database based on the calculated sum of Unicode values of the item ID, facilitating balanced data distribution. Successful insertions are logged, and the user is informed about the number of entries added to each database, enhancing transparency and error tracking. This bulk insertion tool significantly streamlines the process of database population, making it an indispensable feature for

administrative tasks requiring rapid database updates.



3. Data Deletion

The "Delete Data" function in the NutriVision system provides a straightforward and efficient method for users to remove entries from the database. Accessible through the Streamlit interface, this feature allows users to specify the ID of the food item they wish to delete. Once the deletion request is submitted, the system identifies the appropriate database—either `oddIdFoods` or `evenIdFoods`—based on the sum of the Unicode values of the item ID. It then executes the deletion operation, ensuring that all instances of the specified ID are removed. The function returns feedback on the success of the operation, including the number of documents deleted, which is crucial for maintaining data accuracy and integrity. This feature is essential for keeping the database current and removing outdated or incorrect information efficiently.



4. Data Update

The "Modify Data" functionality within the NutriVision system empowers users to update existing entries with new information through a user-friendly interface provided by Streamlit. This feature is critical for maintaining the relevance and accuracy of the nutritional database. Users can specify the ID of the food item they wish to update and provide new values for various attributes such as name, nutritional content (like energy, protein, fats, etc.), and tags. The system calculates the appropriate database based on the ID, checks for the existence of the entry, and then applies the updates, ensuring the information remains current. The update operation confirms the success of changes and notifies users of the modification status, providing immediate feedback if no changes were made due to erroneous inputs or matching data, thereby enhancing data integrity and user interaction.



Database Manager UI

Insert Data Insert Multiple with JSON Delete Data **Modify Data**

Modify Data

ID to Modify

New Name

New Energy per 100g
 - +

New Protein per 100g
 - +

New Fat per 100g
 - +

New Saturated Fat per 100g
 - +

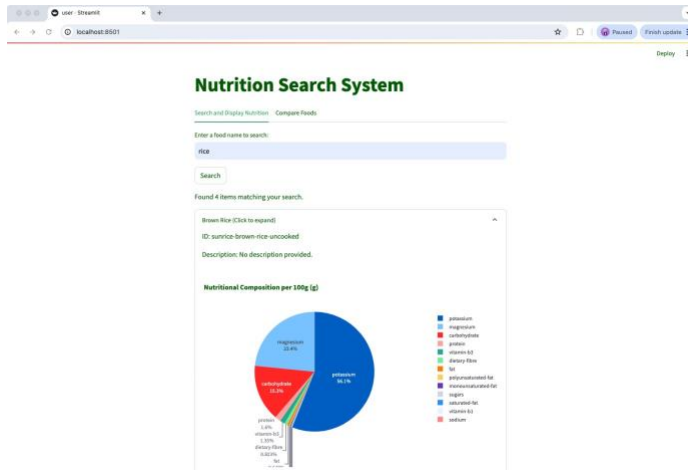
New Carbohydrate per 100g
 - +

New Sugars per 100g
 - +

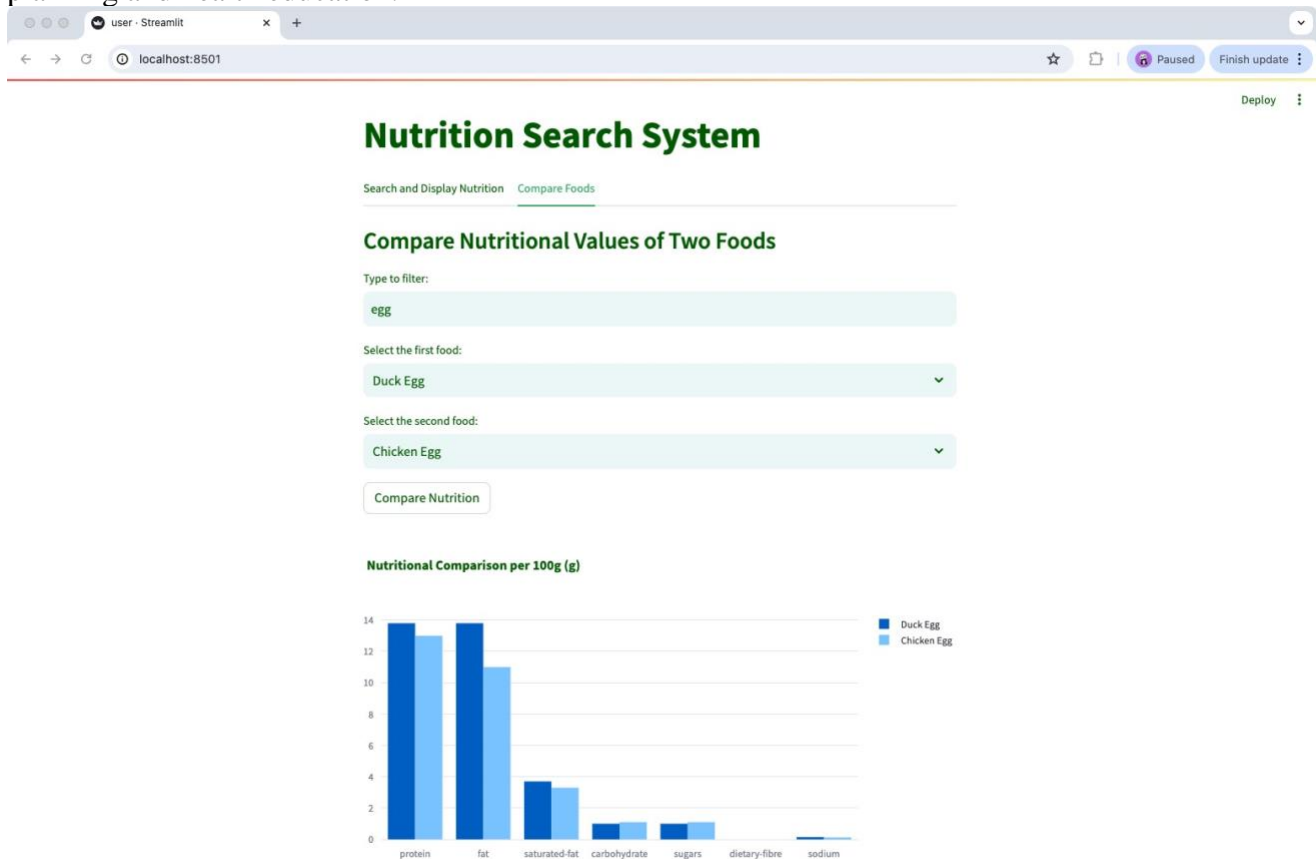
User Interface

1. Search and Display Nutrition

The "Search and Display Nutrition" tab in the NutriVision system enables users to easily search for specific food items by name using a streamlined text input and button interface developed with Streamlit. Upon initiating a search, the application queries both the `oddIdFoods` and `evenIdFoods` MongoDB collections using a regex pattern that ensures precise matching. Results, if found, are displayed under expandable sections where users can view detailed information including nutritional content represented in both text and interactive pie charts, enhancing the user experience by providing a clear and informative breakdown of the food's nutritional values.



- The "Compare Foods" tab in the NutriVision system offers a dynamic functionality where users can select and compare the nutritional profiles of two different food items. After typing to filter and selecting food items from dropdown menus, the system retrieves nutritional data for each from the appropriate MongoDB collections. If data is available, the system generates a comparative bar chart using Plotly, showcasing side-by-side comparisons of key nutrients like protein, fat, and carbohydrates. This visual comparison allows users to easily discern nutritional differences, making it an invaluable tool for diet planning and health education.



Tech Stack

The development of Nutrivision utilized a robust stack of technologies tailored to the needs of efficient data handling and user-friendly interface design:

1. MongoDB
Chose for its flexibility and performance in handling large datasets, MongoDB serves as the backend database, storing food data across two collections for optimized data retrieval.
2. Python
The backend logic was implemented using Python, which interacts with the database and handles all server-side data processing.
3. Streamlit
Used for developing the frontend, Streamlit enabled the creation of an interactive web application that allows users to perform CRUD operations and view data visualizations directly through their web browser.
4. Polt
Integrated within the Streamlit application, Polt provides the tools for creating dynamic and interactive charts that help in visualizing the nutritional data effectively.

Learning Outcomes & Challenges Faced

Learning Outcomes

The development of the NutriVision system provided several key learning outcomes, including a deepened understanding of MongoDB operations, mastery of Streamlit for creating interactive web applications, and enhanced skills in implementing complex data visualization with Plotly. This project also honed my abilities in managing large datasets effectively and designing a user-friendly interface that simplifies complex data interactions.

Challenges Faced

Several challenges were encountered during this project. The first was ensuring efficient data handling and retrieval from a dual-database setup, which required careful planning and testing to balance the load evenly across the **oddIdFoods** and **evenIdFoods** databases. Another significant challenge was implementing robust error handling and validation to prevent data corruption and ensure the reliability of user inputs. Additionally, optimizing the user experience to make the application intuitive and responsive required iterative design adjustments and feedback incorporation.

Conclusion

The NutriVision project has successfully demonstrated the feasibility and effectiveness of a nutrition management system designed to streamline the access and comparison of nutritional data through a user-friendly web interface. By leveraging MongoDB for backend data management and Streamlit for the frontend interface, the project achieved its goals of providing

a reliable, scalable, and easily navigable system. The integration of advanced data visualization tools further enhanced the user experience, allowing for immediate and clear interpretation of complex nutritional information.

Future Scope

Looking forward, the NutriVision system has several avenues for expansion and improvement:

1. **Machine Learning Integration:** Implementing machine learning algorithms to provide personalized nutritional recommendations based on user data and trends observed from the usage patterns.
2. **Mobile Optimization:** Developing a mobile version of the application to increase accessibility and on-the-go usage.
3. **Expansion of Database:** Enriching the database with more diverse food items, including international cuisines and specialty diets to broaden the system's applicability.
4. **Community Features:** Adding social features such as sharing and discussing food items among users, which could foster a community around healthy eating habits.
5. **Real-time Data Updates:** Integrating APIs that fetch real-time data from food databases and nutritional studies to keep the information up-to-date and dynamically expand the database content.

These enhancements will not only improve functionality but also ensure that NutriVision remains at the forefront of nutritional management technology, adapting to new challenges and opportunities in the health and wellness sector.