Project: Don’t Eat That!

Team: Kenton Ma, Charn Rai, Benny Lo, Alex Macdonald, Adil Kydyrbayev

Requirements, Vision, and Scope

Motivation / Opportunity:

Don’t Eat That! is a web application that helps users choose healthier foods by comparing two food items of a genre to each other and suggesting the healthier of the two foods (eg. two kinds of potato chips). Don’t Eat That! Will be free to use and unlike the book *Eat This, Not That!* and its corresponding mobile application, our solution allows users to compare a greater variety of food choices side by side. Additionally, users will be able to make a profile to have access to specialized algorithms that cater to their health needs. Currently existing food comparison web applications are unable to remember the dietary needs and concerns of regular users.

Problem Statement

The problem of	Choosing the healthiest food from two similar options
Affects	People concerned about their diet
The impact of which is	People may not consume the food options that are the most beneficial to their needs
A successful solution would be	A web application that compares two food choices and recommends the healthiest option for the user

Product Position Statement

For	Anyone who is concerned about their nutritional intake and/or has special dietary concerns/needs
Who	Want to choose healthier foods
Our system	Is a web application (all software)
That	Compares two given inputs and outputs the healthier option
Unlike	Current web applications that do not cater to special needs and current books that have a limited variety and do not compare side by side
Our Product	Allows users to create their own accounts to remember special needs, performs side by side comparisons of a wide variety of foods, and has a friendly, easy to use user interface

User Demographics

Non-registered Users:
Non-registered users are people who use the web application “Don’t Eat That!”, without creating a user account. For these people, the “default” setting will be used to determine the healthiest food choice. It is assumed they do not have any adverse medical condition. Age, weight, height, and sex will not play a major role when comparing two food items. Determining the healthier food choice does not depend heavily on these factors.

Registered Users:
Registered users are people who have created an account. The food selection algorithm will be specially tailored to their dietary restrictions, medical conditions, and/or past food choices.

Developers:
Developers will implement, extend, test, and maintain the webpage. We must be careful to write clear code, and document it properly. It is also very important to keep different subsystems/components completely separate, and loosely coupled. The behaviour of one system cannot depend on the behaviour of another.

Feature List

The product will provide the following features:

- Compare two food items against each other and recommend the healthiest choice
- Provide accurate nutritional information about food products
- Keep track of user preferences
- Recommend similar choices that are healthier

Constraints

The product is designed to work in all operating systems, as it is a web application that operates in browsers. There is a price to buying a web domain, but this is a very low cost. The product will also use MySQL to access databases. To build the website, we will use HTML, CSS, Bootstrap, JavaScript and PHP. The user interface will be made using HTML, CSS, and Bootstrap framework, while PHP will be used to make queries to the databases.

Scope and Limitations

Features that will NOT be in our product:

- Will not provide recipes
- Will not cure health problems, use at your own discretion
- Will not provide fitness plans
- Will not replace a dietician
- Will not count your daily calories
- Will not replace regular visits to a health professional
- Does not replace exercise and other healthy living habits
- Does not guarantee weight loss

Assumptions and Dependencies

We expect that HTML, CSS, Bootstrap, JavaScript, and PHP work as intended. The responsiveness of the web application will depend on Bootstrap, JavaScript, and PHP.

Use Cases:

1: Compare two food items

Primary actor	General user
Stakeholders	General user, database manager
Main Success Scenario	1) User enters two food products into the text fields 2) User submits choices 3) The system displays the healthier food choice, including a comparison of the nutritional info from the original two choices. An image of the healthier food choice will also be shown
Extensions and Alternative Flows	1) The system cannot find the queried food item(s) a) System notifies users that the choices are unavailable b) System returns user to home page 2) The user submits only one food a) The system prompts user to enter the second food choice
Open Issues	What if a user enters a food choice that is in the database but is spelled wrong?

2: Register account

Primary actor	General user
Stakeholders	General user, database manager
Main Success Scenario	1) User enters the following information into the system: username, password, height, age, sex, weight, dietary restrictions, medical conditions 2) User submits account registration information 3) The system notifies user that their account has been created
Extensions and Alternative Flows	1) The user entered a pre-existing username a) The system tells the user to enter another username 2) The user entered a password with less than 6 characters a) The system tells the user to enter a longer password 3) The user has not filled out the form completely a) The system tells the user to fill out the form completely
Open Issues	What if a user with malicious intent creates multiple fake accounts?

3: Delete account:

Primary actor	Registered user
Stakeholders	Registered user, database manager
Main Success Scenario	1) The system prompts user to confirm account deletion process 2) The user confirms they want to delete their account 3) The system notifies the user that their account has been deleted
Extensions and Alternative Flows	1) The user cancels the deletion a) The user is returned to the main page, and their account still exists
Open Issues	N/A

4: Edit account information

Primary actor	Registered user
Stakeholders	Registered user, database manager
Main Success Scenario	1) The user changes height, age, sex, weight, dietary restrictions, and/or medical conditions 2) The system prompts user to confirm account information edits 3) The user confirms edits
Extensions and Alternative Flows	1) The user cancels edits a) The system returns user to edit page 2) The user leaves one or more fields empty a) System indicates to user the fields still required 3) The user enters invalid information e.g. an age of -5 a) The system rejects the edit and indicates to the user the fields that require proper input
Open Issues	N/A

5: Compare two food items with an emphasis on a special dietary need(s)

Primary Actor	User with special dietary need(s)
Stakeholders	User with special dietary need(s), database
Preconditions	1) User must have a registered account
Main Success Scenario	1) The user has entered a dietary concern into his account 2) The user submits choices 3) The system uses a specialized algorithm to produce a recommendation that supports the dietary concern 4) The system displays the healthier food choice, including a comparison of the nutritional info from the original two choices. An image of the healthier food choice will also be shown
Extensions and Alternative Flows	1) The system cannot find the queried food item(s) a) System notifies users that the choices are unavailable b) System returns user to home page 2) The user submits only one food a) The system prompts user to enter the second food choice
Open Issues	What if a user enters a food choice that is in the database but is spelled wrong?

Non-functional Requirements

Performance Requirements:
The web application should process server requests in a relatively fast manner. The user should not have to wait longer than 5 seconds to receive the results of their request(s) as that would be too long for a comfortable user experience. The web application should be able to store up to 100 users and their information.

Safety Requirements:
This web application could accidentally be detrimental to a user's health if the wrong recommendations are made. For example, if a product high in sugar is recommended to a user with diabetes, the user's condition could worsen. To prevent such accidents from occurring, we must ensure that all recommendations are the best choice for the individual user by using specialized algorithms. The user's health could also suffer if their registered account information is lost and recommendations are made without their individual concerns. This risk can be minimized by effective database maintenance.

Security Requirements:
Users can make an account which will keep track of their usage history, dietary preferences, and restrictions. These accounts will be kept in a database. To access their information, users have to log in with a username and password, to prevent unauthorized access to user information. In addition, the database must also defend from SQL injection and similar security threats attempting to steal user data.

Software Quality Attributes:
As this web application is likely to be used by users with health concerns, it is imperative that the application is:

- Correct and reliable: the application should always recommend the healthiest choice so the user does not eat something that could worsen their condition
- Usable: the application should be easy to use so users are not confused and misinterpret results
- Maintainable and flexible: as more food products enter the market, the web application has to be ready for change and easily updated
- Interoperability: the application should work the same on all browsers
- Portability: the application should operate and is responsive on mobile phones

Product Backlog:

Priority	Item #	Description	Estimate (Hours)	Estimated By	Remaining Hours By Sprint Start				
					1	2	3	4	5
Very High	1	Product Vision	5	KM	5	0	0	0	0
	2	Set up a website, buy a domain name and server	2	AM	2	0	0	0	0
	3	UI - Display two boxes on the website, allow user to enter two items, then display a result	30	AK	30	20	5	0	0
	4	Food DB Design - Set up database for food items, with accurate nutritional facts	30	CR	30	20	5	0	0
	5	Query design - Food database has to be able to process queries	30	KM	30	20	5	0	0
	6	Algorithm Design - Create algorithm to calculate healthiest food choice	15	BL	15	15	10	0	0
High	7	User DB Design - Set up database for user accounts to store search history and nutritional preferences	20	CR	20	15	5	0	0
	8	Query design - User database has to be able to handle new accounts, log ins, and log outs	20	AM	20	20	10	0	0
	9	Security - Protect from SQL injection and other security threats (ask Farshid	15	AM	15	15	15	10	0

		or Sathish about possible threats)							
Medium	10	UI - Display list of similar, healthier food items after a single item is entered	20	KM	20	20	20	10	0
	11	User interface - Add drop down menus, loading icon with Javascript	10	AM	10	10	8	0	0
Low	12	DB Design - Set up database for restaurants and entrees	20	BL	20	20	20	15	10
	14	User may log in using other social media accounts	20	AK	20	20	20	20	20
	15	DB Design - Allow users to enter new food choices (how do we check if it's valid?)	10	BL	10	10	10	10	10
				TOTALS	247	205	133	65	40