Databases, Network and the Web Coursework for Midterm:

CalorieBuddy

Introduction

In the course, we developed a small dynamic web application including routes, forms and database access. In this assignment you are tasked with creating another dynamic web application which will function as a digital calorie counter to help users manage their diet. Essentially the dynamic web application interacts with users to calculate and display nutritional facts including calories for their recipes based on food ingredients in the recipes.

Working Environment

For the purpose of this assignment, we have setup a working directory called “mid-term” inside the “topic7” folder of your labs. At the moment the folder contains a dummy “index.js” file which creates a web server on PORT 8089. Open this lab and take a look at the folder structure.
You can see the root route of your application by visiting the following URL:
“<Your lab url>” + “/topic7/mid-term/”
For the application to work correctly, you should not move the “index.js” file from its current directory. Furthermore the application must run on PORT 8089.

Requirements

The purpose of the web application is to help people manage their diet by displaying nutritional facts including calories, carbs, fat, protein, salt and sugar in a recipe based on food ingredients in the recipe. The web application provides a number of functionalities that should meet the following requirements:

- R1: Home page:
  - R1A: Display the name of the web applications.
  - R1B: Display links to other pages or a navigation bar which contains links to other pages.
- R2: About page:
  - R2A: Display information about the web application including your name as the developer.
  - R2B: Display a link to the home page or a navigation bar which contains links to other pages.

- R3: Add food page:
  - R3A: Display a form to users to add a new food item to the database. The form should consist of the following items: name, unit, calories, carbs, fat, protein, salt and sugar.
  - R3B: Collect form data to be passed to the back-end (database).
  - R3C: Store food item in the database. Each food item consists of the following fields: name, unit, calories, carbs, fat, protein, salt and sugar.
  - R3D: Display a message indicating that add operation has been done.
  - R3E: Display a link to the home page or a navigation bar which contains links to other pages.
- R4: Search food page
  - R4A: Display a form to users to search for a food item in the database. The form should consist of the name of the food.
  - R4B: Collect form data to be passed to the back-end (database).
  - R4C: Search the database based on the food name collected from the form.
  - R4D: If food found, display a template file (ejs, pug or etc) including data related to the food found in the database to users; name, unit, calories, carbs, fat, protein, salt and sugar.
  - R4E: Display a message to the user, if not found.
  - R4F: Display a link to the home page or a navigation bar which contains links to other pages.
- R5: Update food page
  - R5A: Display search food page.
  - R5B: If food found, display data related to the food found in the database to users including name, unit, calories, carbs, fat, protein, salt and sugar in forms so user can update each field.
  - R5C: Display a message to the user if not found.
  - R5D: Collect form data to be passed to the back-end (database)
  - R5E: Store updated food item in the database.
  - R5F: Display a message indicating the update operation has been done.
  - R5G: Display a link to the home page or a navigation bar which contains links to other pages.

- R6: Delete food page
  - R6A: Display search food page
  - R6B: If food found, display data related to the food found in the database to users including name, unit, calories, carbs, fat, protein, salt and sugar in addition to a delete button.
  - R6C: Display a message to the user if not found.
  - R6D: Delete food item from the database.
  - R6E: Display a message indicating the delete has been done.
  - R6F: Display a link to the home page or a navigation bar which contains links to other pages.
- R7: List foods page
  - R7A: Display all foods stored in the database including name, unit, calories, carbs, fat, protein, salt and sugar, sorted by name.
  - R7B: Display a link to the home page or a navigation bar which contains links to other pages.
- R8: Recipe calorie count page
  - R8A: Display all food items with a check box, so users can select items.
  - R8B: Collect the name of all selected foods.
  - R8C: Calculate sum of the nutritional information (calories, carbs, fat, protein, salt and sugar) related to all selected foods for a recipe and display them.
  - R8D: Display a link to the home page or a navigation bar which contains links to other pages.

Code style and technique

Your code should be written according to the following style and technique guidelines:

- C1: Code is organised into java script (.js) files and template files (.html or .ejs or .pug). Java script files contain web server code (index.js) and middleware (main.js in routes folder), template files (.html and .ejs and .pug) are stored in views folder.
- C2: Each route in main.js has comments describing purpose, inputs and outputs
- C3: Code is laid out clearly with consistent indenting
- C4: Each database interaction has comments describing the purpose, inputs and outputs
- C5: Functions and variables have meaningful names, with a consistent naming style

Documentation

You should write a report and submit your source code. The submission should contain the following items and information:

- D1: Link to your application (more on how to include this during the submission process)
- D2: Source code in standard ZIP format (not mandatory but good as a backup)
- D3: Report in PDF format
- D4: Requirements: for each sub requirements (R1A → R8D) state how this was achieved or if it was not achieved. Explain where it can be found in the code. Use focused, short code extracts if they make your explanation clearer.
- D5: Database structure: tables including purpose, field names and data types for each table.

Marking criteria

We will mark your work according to the set of criteria shown below, which consider the requirements, your programming technique and style and the documentation you have provided:

Category	Criterion	Not addressed	Attempted but did not meet requirements	Met requirements well but did not go beyond that	Met requirements well and went significantly beyond them.	Weight
Code style and technique	C1: Code is organised into java script (.js) files and template files (.html or .ejs or .pug). Java script files contain web server code (index.js) and middleware (main.js in routes folder), template files (.html and .ejs and .pug) are stored in views folder					
Code style and technique	C2: Each route in main.js has comments describing purpose, inputs and outputs					

Code style and technique	C3: Code is laid out clearly with consistent indenting					
Code style and technique	C4: Each database interaction has comments describing the purpose, inputs and outputs					
Code style and technique	C5: Functions and variables have meaningful names, with a consistent naming style					
R1: Home page	R1A: Display the name of the web applications.					
	R1B: Display links to other pages or a navigation bar which contains links to other pages.					
R2: About page	R2A: Display information about the web application including your name as the developer.					
	R2B: Display a link to the home page or a navigation bar which contains links to other pages.					
R3: Add food page	R3A: Display a form to users to add a new food item to the database. The form should consist of the following items: name, unit, calories, carbs, fat, protein, salt and sugar.					

	R3B: Display a message indicating that add operation has been done.					
	R3C: Store food item in the database. Each food item consists of the following fields: name, unit, calories, carbs, fat, protein, salt and sugar.					
	R3D: Display a message indicating that add operation has been done.					
	R3E: Display a link to the home page or a navigation bar which contains links to other pages.					
R4: Search food page	R4A: Display a form to users to search for a food item in the database. The form should consist of the name of the food.					
	R4B: Collect form data to be passed to the back-end (database).					
	R4C: Search the database based on the food name collected from the form.					

	R4D: If food found, display a template file (ejs, pug or etc) including data related to the food found in the database to users; name, unit, calories, carbs, fat, protein, salt and sugar.					
	R4E: Display a message to the user, if not found.					
	R4F: Display a link to the home page or a navigation bar which contains links to other pages.					
R5: Update food page	R5A: Display search food page.					
	R5B: If food found, display data related to the food found in the database to users including name, unit, calories, carbs, fat, protein, salt and sugar in forms so user can update each field.					
	R5C: Display a message to the user if not found.					
	R5D: Collect form data to be passed to the back-end (database)					
	R5E: Store updated food item in the database.					

	R5F: Display a message indicating the update operation has been done.					
	R5G: Display a link to the home page or a navigation bar which contains links to other pages.					
R6: Delete food page	R6A: Display search food page					
	R6B: If food found, display data related to the food found in the database to users including name, unit, calories, carbs, fat, protein, salt and sugar in addition to a delete button.					
	R6C: Display a message to the user if not found.					
	R6D: Delete food item from the database					
	R6E: Display a message indicating the delete has been done.					
	R6F: Display a link to the home page or a navigation bar which contains links to other pages.					

R7: List food page	R7A: Display all foods stored in the database including name, unit, calories, carbs, fat, protein, salt and sugar, sorted by name..					
	R7B: Display a link to the home page or a navigation bar which contains links to other pages.					
R8: Recipe calorie count page	R8A: Display all food items with a check box, so users can select items.					
	RA8B: Collect the name of all selected foods					
	R8C: Calculate sum of the nutritional information (calories, carbs, fat, protein, salt and sugar) related to all selected foods for a recipe and display them.					
	R8D: Display a link to the home page or a navigation bar which contains links to other pages.					
Documentatio n	D1: Source code in standard ZIP format					
	D2: Report in PDF format					

	D3: Requirements: for each sub requirements (R1A → R8D) state how this was achieved or if it was not achieved. Explain where it can be found in the code. Use focused, short code extracts if they make your explanation clearer.					
	D4: Database structure: tables including purpose, field names and data types for each table					