Project Design

Nutrition and Diet Application

Rohandra Macolm

Dobrian Zaprianov

Diamond Zetty

Dorrell Zimmerman

SDEV 495 6380: Capstone in Computer Science

Professor Hung Dao

3 June 2024

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Editor | Description |
| 1 | 28 May 2024 | Dorrell Zimmerman | Initial Document |
| 2 | 01 June 2024 | Dorrell Zimmerman | Update of pseudocode and Event trace data |
| 3 | 02 June 2024 | Dorrell Zimmerman | Revision and Formatting |

Class Diagram

A diagram of a computer

Description automatically generated

Figure 1: Nutrition and Diet Application Class Diagram

Event Trace Diagrams

Scenario 1:

Description: The user enters their height and weight and the application provides their BMI.

Precondition: The user has required height and weight information to input into application.

Post-Condition: The BMI data is calculated and displayed to the user.

A diagram of a weight and calories

Description automatically generated

Figure 2: Event Trace Scenario 1 UML

Scenario 2:

Description: The user enters their age, gender, height and weight and the application provides their BMR.

Precondition: The user has the required gender, height, and weight data to input into application.

Post-Condition: The BMR data is calculated and displayed to user.

A screenshot of a chat

Description automatically generated

Figure 2: Event Trace Scenario 2 UML

Scenario 3:

Description: The user enters their daily allowed calorie value from their BMR calculation and the food and calorie value of what they have eaten for the day.

Precondition: The user has already calculated their daily BMR data and has calorie value of the food items they have eaten that day.

Post-Condition: The food diary subtracts the food calorie data of the day and provides a total of their daily consumed calories versus their daily allotted calories.

A black text on a white background

Description automatically generated

Figure 3: Event Trace Scenario 3 UML

Class Design

Pseudocode

Flask App

Nutrition\_Application/

app.py # This file contains the Flask application logic

templates/ # Directory for HTML templates

index.html # HTML template for the home page

bmr.html # HTML template for the bmr page

bmi.html # HTML template for the bmi page

diary.html # HTML template for the diary page

# Import modules from Flask

Import Flask and render\_template, request, redirect, url\_for

# Initialize the Flask application

Create a Flask app instance

# Define the route for the home page

When a user navigates to the root URL ('/'):

Render the home.html template

BMR Calculator

Python Code

# Define the route for the BMR page (GET and POST requests)

# Define the route for the BMR page (GET and POST requests)

When a user navigates to the '/bmr' URL:

If the request method is GET:

Try:

Render the bmr.html template without BMR result

Except:

Redirect to the home page with an error message

If the request method is POST:

Try:

Extract height, weight, age, and sex from the form data

Calculate the BMR based on the inputs

Render the bmr.html template with the calculated BMR

Except:

Redirect to the home page with an error message

HTML Code

Set the document type to HTML

In the <head> section:

Set the character encoding to "UTF-8"

Set the title to "BMR Calculator"

Css for styling

In the <body> section:

Add a heading with "BMR Calculator"

Create a form that sends a POST request to the "/bmr" URL

Add a label for the height input field with the text "Height (cm):"

Add a number input field with:

id attribute set to "height"

name attribute set to "height"

required attribute to ensure the field must be filled out

Add a line break

Add a label for the weight input field with the text "Weight (kg):"

Add a number input field with:

id attribute set to "weight"

name attribute set to "weight"

required attribute to ensure the field must be filled out

Add a line break

Add a label for the age input field with the text "Age:"

Add a number input field with:

id attribute set to "age"

name attribute set to "age"

required attribute to ensure the field must be filled out

Add a line break

Add a label for the sex select field with the text "Sex:"

Create a select dropdown with:

id attribute set to "sex"

name attribute set to "sex"

required attribute to ensure the field must be filled out

Two options:

- One with value "male" and text "Male"

- One with value "female" and text "Female"

Add a line break

Add a submit button with the text "Calculate BMR"

Close the form

If BMR result is available:

Add a paragraph with the text "Your Basal Metabolic Rate (BMR) is: " followed by the BMR value and " calories/day"

Add a link back to the home page with the text "Back to Home" and the href attribute set to the URL for the home route

Close the body and html tags

BMI Calculator

Python Code

# Define the route for the BMI page (GET and POST requests)

When a user navigates to the '/bmi' URL:

If the request method is GET:

Try:

Render the bmi.html template without BMI result

Except:

Redirect to the home page with an error message

If the request method is POST:

Try:

Extract height and weight from the form data

Calculate the BMI based on the inputs

Render the bmi.html template with the calculated BMI

Except:

Redirect to the home page with an error message

HTML Code

Set the document type to HTML

In the <head> section:

Set the character encoding to "UTF-8"

Set the title to "BMI Calculator"

Add css for styling

In the <body> section:

Add a heading with "BMI Calculator"

Create a form that sends a POST request to the "/bmi" URL

Add a label for the height input field with the text "Height (cm):"

Add a number input field with:

id attribute set to "height"

name attribute set to "height"

required attribute to ensure the field must be filled out

Add a line break

Add a label for the weight input field with the text "Weight (kg):"

Add a number input field with:

id attribute set to "weight"

name attribute set to "weight"

required attribute to ensure the field must be filled out

Add a line break

Add a submit button with the text "Calculate BMI"

Close the form

If BMI result is available:

Add a paragraph with the text "Your Body Mass Index (BMI) is: " followed by the BMI value

Add a paragraph with the text "This is considered: " followed by the BMI category

Add a link back to the home page with the text "Back to Home" and the href attribute set to the URL for the home route

Close the body and html tags

Food Diary

Python Code

# Define the route for the diary page (GET and POST requests)

When the '/diary' URL is requested:

If the request method is POST:

Try:

# Example daily caloric goal

Set daily\_goal to 2000

Define days as a list containing ['monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday']

Initialize an empty dictionary called diary\_data

For each day in days:

Try:

# Extract calories and foods from the form data

Get the value of '{day}\_calories' from the form and convert it to an integer, store in calories

Get the value of '{day}\_foods' from the form, store in foods

# Calculate the difference from the daily goal

Set difference to calories - daily\_goal

# Store data in diary\_data

Set diary\_data[day] to a dictionary containing 'calories', 'foods', and 'difference'

Except:

Redirect to the diary page with an error message

# Render diary.html template with the diary data

Render the 'diary.html' template, passing diary\_data to it

Except:

Redirect to the diary page with an error message

Else:

Try:

# Render diary.html template without diary data

Render the 'diary.html' template without any additional data

Except:

Redirect to the diary page with an error message

HTML Code

Set the document type to HTML

Set the language attribute to "en" in the <html> tag

In the <head> section:

Set the character encoding to "UTF-8"

Set the title to "Diary"

Css for styling

In the <body> section:

Add a heading with "Weekly Diary"

Create a form that sends a POST request to the "/diary" URL

For each day in the list ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']:

Add a subheading with the day's name

Add a label for the calories input field with the text "Calories:"

Add a number input field with:

id attribute set to the lowercase day name followed by "\_calories"

name attribute set to the lowercase day name followed by "\_calories"

required attribute to ensure the field must be filled out

Add a line break

Add a label for the foods textarea with the text "Foods:"

Add a textarea with:

id attribute set to the lowercase day name followed by "\_foods"

name attribute set to the lowercase day name followed by "\_foods"

rows attribute set to "4"

cols attribute set to "50"

required attribute to ensure the field must be filled out

Add a line break

End the loop

Add a submit button with the text "Submit"

Close the form

If diary\_data is defined:

Add a heading with "Diary Results"

For each day and its corresponding data in diary\_data:

Add a subheading with the capitalized day name

Add a paragraph with the text "Calories: " followed by the day's calories value

Add a paragraph with the text "Foods: " followed by the day's foods value

Add a paragraph with the text "Difference from daily goal: " followed by the difference value and " calories"

End the loop

Add a link back to the home page with the text "Back to Home" and the href attribute set to the URL for the home route

Unresolved risk

|  |  |
| --- | --- |
| Risks | Mitigation |
| BMR calculations does not account for Transgender or Non-Binary users. | Include note on BMR page that states that there is currently no calculation or guidelines that does not use sex assigned at birth to calculate BMR. |
| Users having to manually input calories into daily tracker. | Including instructions set on index page on calculator and tracker. Include calories information on basic food items. |