Software Requirements Specification (SRS)

Project Name: BiteRight

Prepared By: 1-Hana Amr
2-Nada Mohamed
3-Farah Mostafa
4-Youssef Ahmed

Date: 9/11/2024

---

1. Introduction

1.1 Purpose

The purpose of this document is to outline the functional and non-functional requirements for BiteRight, a personalized nutritional website. This document serves as a detailed guide for developers, testers, and stakeholders to understand the system's features, functionality, and constraints.

1.2 Scope

BiteRight is a website designed to help users achieve health and fitness goals through tailored meal plans. The system will provide functionality for:

- User Registration and Authentication
- Profile Setup
- Calories Calculation
- BMI Calculation
- Personalized Meal Plan Generation
- Daily and Weekly Meals
- User Feedback Mechanism
- Goal Setting and Adjustment
- Tips and Tricks
- User Support and “Contact Us”

1.3 Definitions, Acronyms, and Abbreviations

- **API**: Application Programming Interface
- **UAT**: User Acceptance Testing
- **GDPR**: General Data Protection Regulation

- **BMI**: Body Mass Index

1.4 References

- Krenitsky, J. (2005). Adjusted Body Weight, PrO: Evidence to support the use of adjusted body weight in calculating calorie requirements. Nutrition in Clinical Practice, 20(4), 468–473. https://doi.org/10.1177/0115426505020004468

- Calories

---

2. System Overview

2.1 Product Perspective

BiteRight is a standalone website interacting with users via a frontend interface and backend services. It consists of:

- **Frontend**: User interface for registration, meal formation, and desire goal.
- **Backend**: Meal generation logic and user management.
- **Database**: Kaggle Dataset.

2.2 Product Functions

The key functions of the system include:

1. **User Registration and Authentication:** Allows secure user registration, login, and account management.
2. **Profile Setup**: Collects personal data for customized calories calculation and nutritional recommendations.
3. **Calorie and Calculation**: Determines daily calorie targets based on user information and desire weight.
4. **BMI Calculation**: Calculates BMI with category guidance on health goals.
5. **Personalized Meal Plan Generation**: Generates daily and weekly meal plans tailored to user needs.
6. **Daily and Weekly Meals:** Allows users to log their meals and monitor calorie and nutrient intake over time.

7. **User Feedback Mechanism**: Collects user feedback on meal plans to improve personalization.
8. **Goal Setting and Adjustment**: After a specific period of time, user inputs the current weight to check if goal weight is reached.
9. **Tips and Tricks Section:** Offers users helpful nutritional advice, meal prep tips, and motivational content.
10. **User Support and "Contact Us":** Provides a dedicated channel for user support and feedback.

2.3 User Classes and Characteristics

Different types of users who will interact with the system include:

- **Admin Users**: Access to full system functionality for user management and data maintenance.

- **Regular Users**: Standard access to meal planning, tracking, and feedback features.
- **Guest Users**: Limited access to public content and basic information resources.

2.4 Operating Environment

The system will operate in the following environments:

- **Client Side**: Runs on web browsers (Chrome, Firefox, Safari).

- **Server Side**: Hosted on a cloud server running Windows.
- **Database**: Kaggle Dataset.

---

3. Functional Requirements

3.1 Use Case Diagrams / User Stories

- **Use Case 1: User Registration and Authentication**
  - **Description:** Allows users to register, log in, or authenticate options like Google.
  - **Actors:** Registered User, Admin
  - **Preconditions:** User has a valid email.
  - **Postconditions:** User is logged in and redirected to the profile setup page.
  - **Steps:**
    1. User inputs registration details.
    2. The system validates information.

3. The system creates a user account and sends the user to log in page.

- **Use Case 2: Profile Setup**
    - **Description:** Enables users to input personal details.
    - **Actors:** Registered User
    - **Preconditions:** User is logged in.
    - **Postconditions:** User profile data is saved and ready for calculations and meal planning.

Steps:

1. **Access Profile Setup**: User navigates to the services section of the app.
2. **Enter Personal Details**: User enters required personal information, such as age, weight, height, activity level in the calorie calculator and bmi calculator.
3. **Review Data**: The user reviews the entered information for accuracy.
4. **Calculations**: Calorie and bmi.
5. **Redirect to Meal Plan Setup**: User is redirected to the next step in meal planning, where personalized meal plans can be generated based on the profile.

- Use Case 3: Calorie Calculation

- **Description**: Calculates the user's daily calorie needs based on their desired weight.
- **Actors**: Registered User
- **Preconditions**: User has completed profile setup, including age, weight, height, activity level.
- **Postconditions**: The system calculates and displays the daily caloric needs.
- **Steps**:
    1. User inputs profile details (age, weight, height, activity level).
    2. The system calculates total daily calorie needs based on user data.
    3. Users see their daily calorie intake.

-Use Case 4: BMI Calculation

- **Description**: Calculates the user’s Body Mass Index (BMI) based on their height and weight.
- **Actors**: Registered User
- **Preconditions**: User has entered their height and weight in the profile.
- **Postconditions**: The system displays BMI value and health category (e.g., underweight, normal weight, overweight, obese).
- **Steps**:
    1. User enters height and weight.
    2. System calculates the BMI using the formula: BMI = weight (kg) / height^2 (m^2).
    3. System displays the BMI value along with a health category (underweight, normal weight, overweight, obese).

-Use Case 5: Personalized Meal Plan Generation

- **Description**: Generates personalized meal plans tailored to the user’s desire goal, and calories.
- **Actors**: Registered User
- **Preconditions**: User has a complete profile.
- **Postconditions**: The system generates a personalized meal plan based on user calculations.
- **Steps**:
    1. System generates a daily or weekly meal plan that meets the user’s dietary needs.
    2. User views the generated meal plan.

-Use Case 6: Daily and Weekly Meal Logging

- **Description**: Allows users to log and track their meals daily and weekly, monitoring calorie intake.
- **Actors**: Registered User
- **Preconditions**: User is logged in and has a meal plan set up.
- **Postconditions**: The system tracks daily and weekly meals, calorie intake.
- **Steps**:
    1. System tracks calories for each meal.
    2. System customizes the user’s daily and weekly nutrient intake.
    3. Users can see their progress towards calorie.

-Use Case 7: User Feedback Mechanism

- **Description**: Collects user feedback on meal plans and satisfaction.
- **Actors**: Registered User
- **Preconditions**: User has completed at least one meal plan.
- **Postconditions**: The system updates meal recommendations based on user feedback.
- **Steps**:
  1. Users provide feedback on their meal plan (e.g., satisfaction, taste, ease of preparation).
  2. System collects the feedback and adjusts future meal suggestions.
  3. Users receive updated meal recommendations based on feedback.

-Use Case 8: Goal Setting and Adjustment

- **Description**: Allows users to set the desire weight.
- **Actors**: Registered User
- **Preconditions**: User has a profile set up.
- **Postconditions**: The system customizes meal plans, calorie targets based on the desire weight.
- **Steps**:
  1. Users set the desire weight.
  2. The system adjusts calorie based on the goal.
  3. System generates a new meal plan tailored to the updated goal.

Use Case 9: Tips and Tricks Section

- **Description**: Provides helpful nutritional tips, meal strategies, and motivational content to assist users in reaching their health goal.
- **Actors**: Registered User
- **Preconditions**: User is logged in.
- **Postconditions**: The system provides helpful tips and tricks, which may include nutritional advice, and motivational content.
- **Steps**:
  1. User accesses the "Tips and Tricks" section of the app.
  2. The system displays tips.
  3. User browses through various tips and incorporate them into their health journey.

-Use Case 10: User Support and "Contact Us"

- **Description**: Provides users with a way to contact support for assistance with technical issues or general inquiries.
- **Actors**: Registered User
- **Preconditions**: User is logged in.
- **Postconditions**: The user’s query or issue is submitted to the support team, and the user receives a response.
- **Steps**:
    1. User navigates to the "Contact Us" section of the app.
    2. User submits a support request (e.g., technical issues, questions).
    3. Support team responds to the user's request through email or app notification.

3.2 Feature Requirements

Feature 1: Calorie Calculation

Description: Determines daily calorie target based on user goal.

Inputs: User profile data (age, weight, height, activity level).

Outputs: Daily recommended calorie intake.

Error Handling:

- Prompts user if profile data is incomplete or unrealistic.
- Ensures that caloric intake does not fall below healthy minimum thresholds.

Feature 2: BMI Calculation

Description: Calculates BMI with category guidance on health goal.

Inputs: User profile data (weight, height).

Outputs: BMI value and corresponding category (underweight, normal weight, overweight, obese) with health goal guidance.

Error Handling:

- Warns if height or weight data is outside plausible range.

- Provides health suggestions based on BMI category.

Feature 3: Personalized Meal Plan Generation

Description: Generates daily and weekly meal plans tailored to user needs.

Inputs: User profile data (calories, desire weight).

Output: Customized meal plans with meal recommendations

Error Handling:

- Notifies user if profile data is missing or incomplete.

Feature 4: Daily and Weekly Meals

Description: Allows users to calorie intake over time.

Inputs: calories.

Output: Daily and weekly logs of meals consumed, with total calorie.

Feature 5: User Feedback Mechanism

Description: Collects user feedback on meal plans to improve personalization.

Inputs: feedback on whole services.

Output: Feedback data used to improve the system.

Error Handling:

- Verifies that the feedback is related to a valid meal plan.

Feature 6: Goal Setting

Description: The user logs the desired weight that wants to be reached.

Inputs: User profile data.

Outputs: Adjusted meal plans and recommendations based on goal.

Feature 7: Tips and Tricks Section

Description: Offers users helpful, for example nutritional advice, meal tips, and motivational content.

Outputs: Personalized tips and tricks for nutrition and motivation.

Error Handling:

- Suggests more relevant tips based on user profile data.

Feature 8: User Support and "Contact Us"

Description: Provides a dedicated channel for user support.

Inputs: User inquiry or request for support, contact information.

Outputs: Response from support team, user confirmation of issue resolution.

4. Non-Functional Requirements

4.1 Performance Requirements

- The system should respond to user requests within 2 seconds.
- Supports up to 10,000 concurrent users without significant performance degradation.

4.2 Security Requirements

- User data, including health metrics, must be encrypted in transit and at rest.

- Implement secure authentication and authorization protocols to restrict access.

4.3 Usability Requirements

- Interface should be intuitive and accessible, with user-friendly navigation.

4.4 Reliability and Availability Requirements

- System must maintain 99.9% uptime.
- It should recover from a failure within 5 minutes.

4.5 Scalability

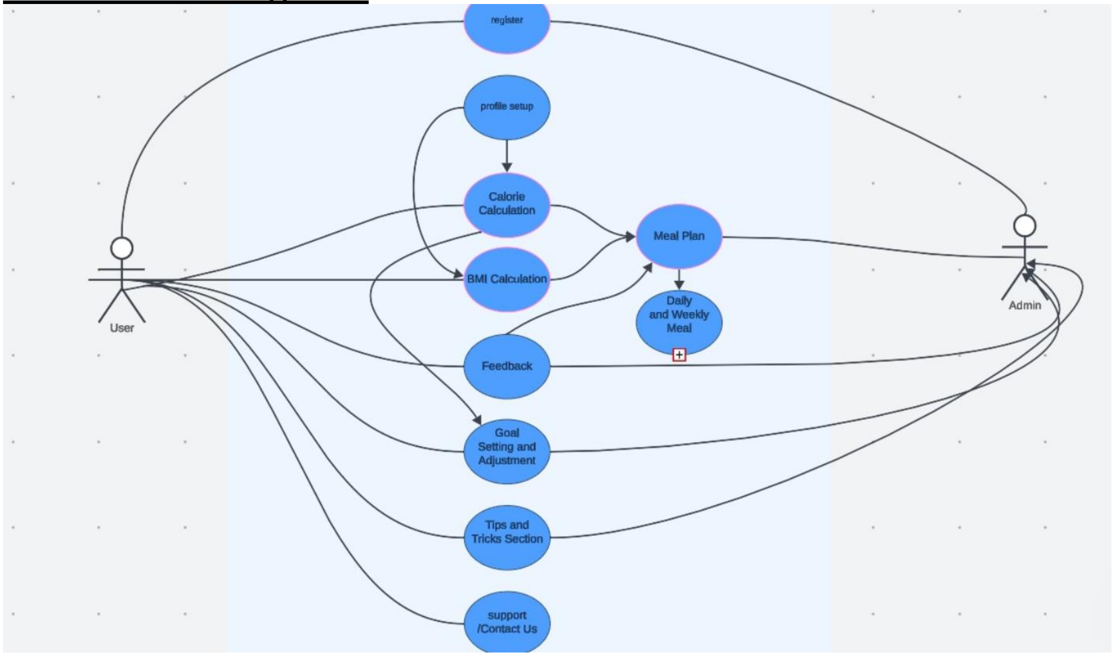
- The system should be able to scale to support 100,000 users and large datasets.

4.6 Compatibility

- The software should work on modern web browsers, including Chrome, Firefox, Edge, and Safari.

5. System Models

5.1 Use Case Diagrams



5.2 Data Flow Diagrams

Use Case 1: User Registration and Authentication

Happy Scenario

1. **User Inputs Registration Details** → User enters details such as email, and password.
2. **System Validates Information** → The system checks if the email is valid and password meets security requirements.
3. **System Creates User Account** → If all validations pass, the account is created, and user is redirected to the login page.
4. **User Login and Authentication** → The user logs in and is directed to the Profile Setup page.

Worst Scenario

1. **Invalid Registration Details** → If any required information is missing, the system prompts the user to enter all fields.
2. **Account Not Created** → If the email is already registered, the system prompts the user to log in instead.

Use Case 2: Profile Setup

Happy Scenario

1. **User Accesses Profile Setup** → Navigates to profile setup section.
2. **User Inputs Personal Details** → User fills in age, weight, height, and activity level.
3. **System Reviews Data** → Validates inputs and saves data.
4. **Redirect to Meal Plan Setup** → User moves to the meal planning section based on profile data.

Worst Scenario

1. **Invalid Input** → If values are out of realistic range, the system prompts for re-entry.
2. **Incomplete Data** → If any field is left blank, the user is notified.
3. **Redirect Fails** → If data save fails, the user is prompted to retry the process.

Use Case 3: Calorie Calculation

Happy Scenario

1. **System Calculates Daily Caloric Needs** → Based on input data, the system computes caloric requirements.
2. **Display Calorie Intake** → Caloric needs are displayed to the user.

Worst Scenario

1. **Invalid Calculation Input** → If profile information is incomplete, the system notifies the user.

Use Case 4: BMI Calculation

Happy Scenario

1. **User Enters Weight and Height** → The system receives user input for BMI calculation.
2. **BMI Calculation** → System calculates BMI and displays health category.

Worst Scenario

1. **Invalid Height/Weight Entry** → If values are out of range, the user receives an error message.

Use Case 5: Personalized Meal Plan Generation

Happy Scenario

1. **System Generates Meal Plan** → Based on the profile, caloric needs, and goal weight.
2. **Display Meal Plan** → The personalized plan is shown to the user.

Worst Scenario

1. **Data Error in Profile** → If required profile data is missing, user is prompted to complete it.

Use Case 6: Daily and Weekly Meal Logging

Happy Scenario

1. **System Tracks Intake** → The System provides daily/weekly meal intake.

Worst Scenario

1. **Inaccurate results** → Daily/Weekly meal plan has inaccurate calories.

Use Case 7: User Feedback Mechanism

Happy Scenario

1. **User Submits Feedback** → Inputs satisfaction, and preferences.

Worst Scenario

1. **Feedback Submission Error** → User receives an error message if feedback cannot be submitted.

Use Case 8: Goal Setting and Adjustment

Happy Scenario

1. **User inputs current weight** → After a specific period of time, user inputs current weight to check if goal weight is reached.
2. **System Adjusts Caloric Needs** → Updates meal plans and caloric targets.

Worst Scenario

1. **Goal not reached** → User follows the meal plan but goal is not reached.

Use Case 9: Tips and Tricks Section

Happy Scenario

1. **User Accesses Tips Section** → User browses available tips.
2. **System Displays Tips** → Helpful content is shown to the user.

Worst Scenario

1. **Content Loading Error** → Tips content fails to load.

Use Case 10: User Support and "Contact Us"

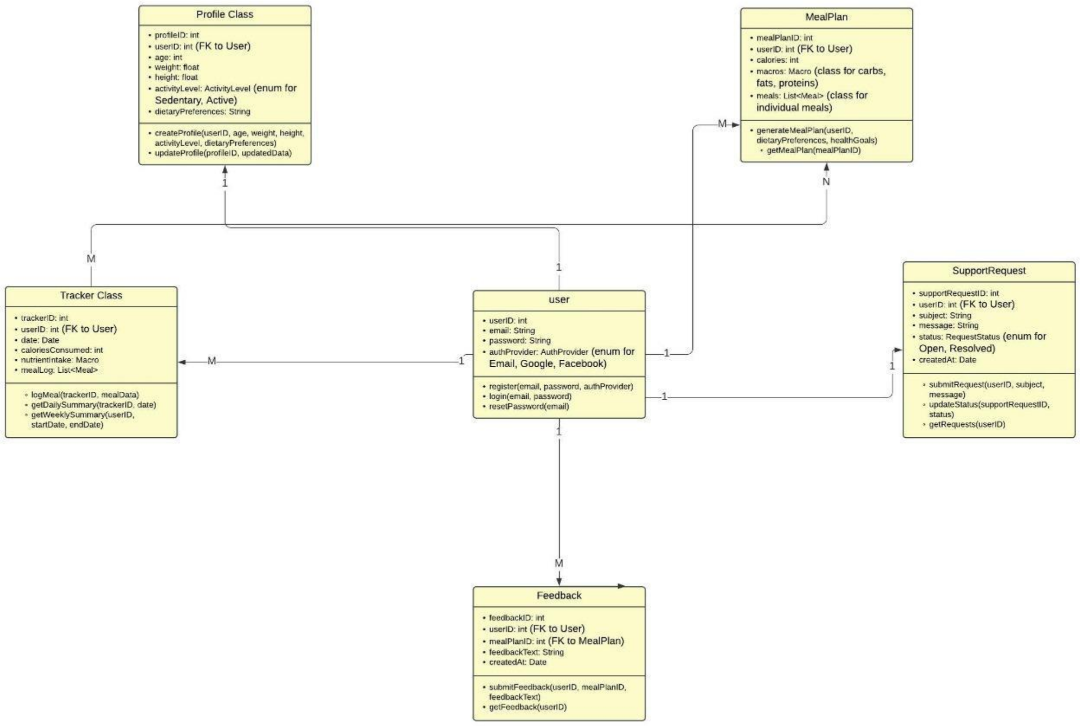
Happy Scenario

1. **User Submits Support Request** → Details the issue or question.
2. **System Receives Request** → Sends notification to support team.
3. **Support Team Responds** → User receives a response through email or app notification.

Worst Scenario

1. **Support Request Submission Fails** → User is prompted to retry if an error occurs.

5.3 Class Diagrams


Profile Class
• profileID: int
• userID: int (FK to User)
• age: int
• weight: float
• height: float
• activityLevel: ActivityLevel (enum for Sedentary, Active)
• dietaryPreferences: String
• createProfile(userID, age, weight, height, activityLevel, dietaryPreferences)
• updateProfile(profileID, updatedData)
MealPlan
• mealPlanID: int
• userID: int (FK to User)
• calories: int
• macros: Macro (class for carbs, fats, proteins)
• meals: List<Meal> (class for individual meals)
• generateMealPlan(userID, dietaryPreferences, healthGoals)
• getMealPlan(mealPlanID)
Tracker Class
• trackerID: int
• userID: int (FK to User)
• date: Date
• caloriesConsumed: int
• nutrientIntake: Macro
• mealLog: List<Meal>
◦ logMeal(trackerID, mealData)
▪ getDailySummary(trackerID, date)
◦ getWeeklySummary(userID, startDate, endDate)
user
• userID: int
• email: String
• password: String
• authProvider: AuthProvider (enum for Email, Google, Facebook)
• register(email, password, authProvider)
• login(email, password)
• resetPassword(email)
SupportRequest
• supportRequestID: int
• userID: int (FK to User)
• subject: String
• message: String
• status: RequestStatus (enum for Open, Resolved)
• createdAt: Date
◦ submitRequest(userID, subject, message)
◦ updateStatus(supportRequestID, status)
◦ getRequests(userID)
Feedback
• feedbackID: int
• userID: int (FK to User)
• mealPlanID: int (FK to MealPlan)
• feedbackText: String
• createdAt: Date
• submitFeedback(userID, mealPlanID, feedbackText)
• getFeedback(userID)
1
M
N
M
1
M
1
1
1
1
1
1
M

6. External Interface Requirements

6.1 User Interfaces

The system should have an intuitive UI with the following major components:

- **Home Screen** Our slogan and the menu that contain all our detailed services.
- **Dashboard**: Allows users to reach to desire weight , track meals, and access health metrics.

6.2 API Interfaces

- Provides APIs for user registration, meal plan generation, and progress tracking.

- Follows REST protocol for easy integration with frontend.

6.3 Hardware Interfaces

- N/A for this project; no external hardware integration required.

7. Other Requirements

7.1 Legal and Regulatory Requirements

- The system must comply with GDPR to ensure user privacy and data protection.

7.2 Documentation Requirements

- Provide user manuals and API documentation for developers.

7.3 Data Backup Requirements

- Data backups should be performed daily and retained for 30 days.

---

8. Conclusion

This SRS document outlines the necessary functional and non-functional requirements for BiteRight. Adhering to these specifications ensures the development team can deliver a product that meets user needs and comply with privacy standards. This document serves as a foundational guide for building a secure, scalable, and user-friendly application that supports users in achieving their health and fitness goals through personalized meal planning.