Republic of the Philippines
BATANGAS STATE UNIVERSITY
The National Engineering University
Lipa Campus
A. Tanco Drive, Brgy. Marawoy, Lipa, Batangas, Philippines 4217
Tel Nos.: (+63 43) 980-0385; 980-0387; 980-0392 to 94 loc. 3130
E-mail Address: cics.lipa@g.batstate-u.edu.ph | Website Address: http://www.batstate-u.edu.ph

College of Informatics and Computing Sciences

NutriAI: Meal Dietary Planner an AI - Generated Management System

A Project Study Presented to the
College of Informatics and Computing Sciences
Batangas State University – Lipa

In partial fulfillment of the requirements of the course
IT 312 - Systems Integration and Architecture

Prepared by:
Arano, Chester Jeff A.
Jardin, Fiel Aerhoze P.
Mendoza, Jose Emmanuel M.

IT - NT - 3101
Oct 2024

TABLE OF CONTENTS

College of Informatics and Computing Sciences

I. PROJECT TITLE

NutriAI: Meal Dietary Planner an AI - Generated Management System

II. OUTCOMES AND GOAL

a. Objective(s)

- To provide personalized meal plans which AI will generate based on the user's health goals, preferences and nutritional needs.
- To foster community for sharing dietary tips, plans, and progress for creating a platform.
- To improve user management and AI - model monitoring for admin to maintain system integrity and to improve user experiences.
- To continuously learn using AI from user interaction, offering increasingly related to dietary suggestions and recommendations.

b. Intended Learning Outcome (ILO)

ILO 1: Users will be able to understand and they can implement personalized meal plans that are in line with their health goals.

ILO 2: Admin will manage the system efficiently, they can ensure proper content moderation, users support and AI model adjustment based on the real - time data.

ILO 3: Users will be able to learn how to monitor and track their dietary progress. Adjusting their habits based on AI - generated suggestions.

ILO 4: Users will actively participate in an interactive community platform, exchanging insights and sharing their dietary experiences.

c. Sustainable Development Goal (SDG)

SDG 3:Good Health and Well - Being. By helping individuals make informed dietary choices, promoting nutritional awareness and to encourage healthy lifestyle. Through personalized meal planning, BMI tracking, and community support, the project aligns with SDG 3 , which contributes to improving health outcomes for users.

III. BACKGROUND / SCENARIO

Nowadays health is a priority, many individuals struggle to find meal plans that meet their specific needs and preferences. The overwhelming amount of dietary information, along with a lack of awareness of their physical and internal needs, makes it challenging to develop effective eating habits. While some people are disciplined in following strict diets, others lack the time or skills to plan meals that

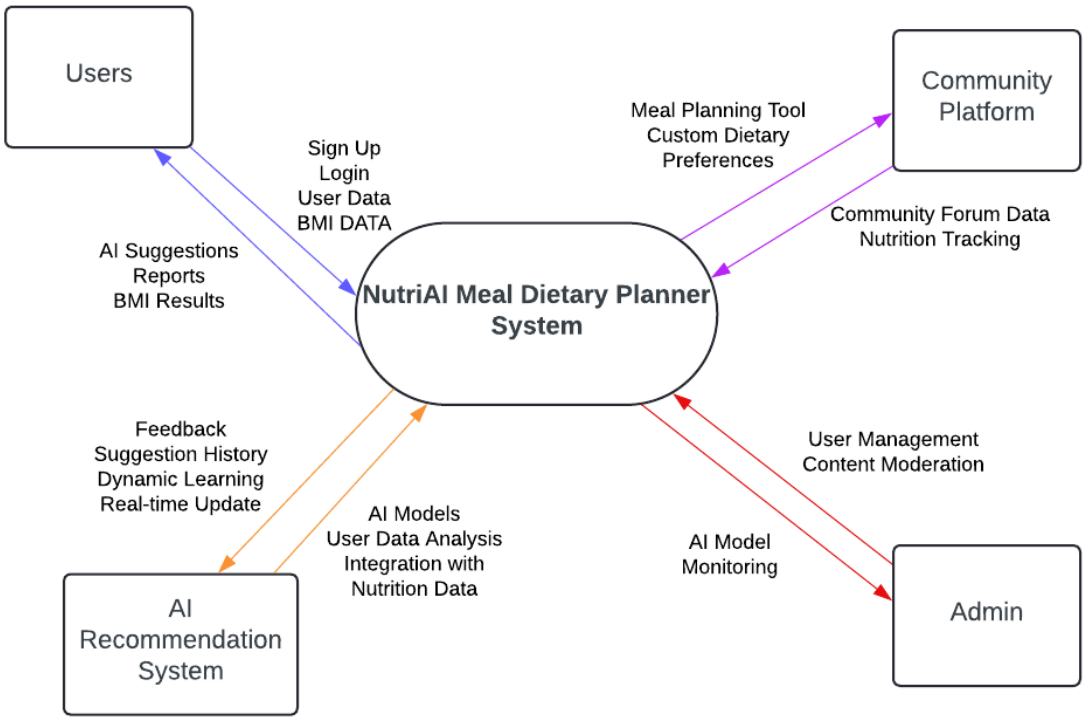
align with their health and fitness goals.

To address these challenges, NutriAI, an AI-generated meal dietary planner management system, is proposed to provide personalized meal suggestions powered by AI. It tracks dietary habits and fosters a supportive community for knowledge exchange. With feedback loops and real-time updates, NutriAI adapts to each user's preferences and progress, ensuring a tailored experience. The system also integrates user management and AI monitoring, enabling administrators to oversee functionality and maintain accurate, beneficial content.

Through this system , it enhances overall health and well-being by equipping users with the knowledge and resources to make informed food choices. It refines its AI algorithm to analyze user data, delivering accurate and relevant recommendations that align with system goals and user needs.

IV. SYSTEM MODELS/DESIGN

A. Context Diagram



External Entities: Users, Community Platform, AI Recommendation System, Admin

- **Process:**
  - **NutriAI Meal Dietary Planner System**
- **Data Stores:**
  - **User Accounts -** CREATE, READ, UPDATE, DELETE
  - **Users Database -** CREATE, READ, UPDATE, DELETE
  - **Admin Database** - CREATE, READ, UPDATE, DELETE
  - **Community Data Store -** CREATE, READ, UPDATE, DELETE
  - **AI Model Store -** CREATE, READ, UPDATE
  - **Admin Activity Logs -** READ, CREATE, UPDATE, DELETE
  - **User Feedback Data Store -** CREATE, READ

- **Data Flows:**
  - **Users:** SignUp, Login, User Data, BMI Data, AI Suggestions, Reports, BMI Results
  - **Admin:** User Management, Content Moderation, AI Model Monitoring
  - **Community Platform:** Meal Planning Tool, Custom Dietary Preferences, Community Forum Data, Nutrition Tracking
  - **AI Recommendation:** AI Models, User Data Analysis, Integration with Nutrition Data.

B. Data Flow Diagram (DFD - Level 0):

Data Flow of Users:
Landing Page, Sign Up / Log In, User Profile, BMI Calculator, Weekly/Monthly Reports, AI Suggestions, Nutrition Guide, Forum.

1. Sign Up / Log In System:

Input: Users enter login credentials or account creation details.
Process: The system verifies credentials or checks account details.
Output:If valid, users can log in or create an account successfully and if invalid it will display an error message.

Input: Users input their Gmail.
Process: The system sends an otp to the Gmail address.
Output: If the user received the otp they can change the password, the change password is successful.

2. User Profile / Dashboard:

Input: Users request access to their profile or dashboard.
Process: System retrieves user data from the database.
Output: Display personalized user profile, current status, weekly report, and monthly report.

3.Weekly/Monthly Report:

Input: Users request a summary of weekly or monthly activity.
Process: The system generates a report based on collected user data.
Output: Display a detailed report showing user activity over time.

4. AI Generated Suggestions:

Input: Users provide preferences or request suggestions.
Process: The AI recommendation system analyzes data and preferences.
Output: The system displays AI-generated meal suggestions tailored to user preferences.

5. Nutrition Guide:

Input: Users search for nutrition information or dietary guidelines.
Process: The system fetches relevant nutrition data based on the user's search query.
Output: Display nutrition guide information tailored to the user's dietary needs.

6. BMI Calculator:

Input: Users enter height and weight.
Process: The system calculates the BMI based on the entered data.
Output: Display BMI results with categorized health suggestions.

7. Forum:

Input: Users post questions or participate in discussions.
Process: The system stores user posts or comments in the forum database.
Output: Display user contributions to other community members.

Data Flows Admin Management System:
Landing Page, User Management, Content Moderation, AI Model Monitoring.

1. User Management:

Input: Admins request to manage user accounts.
Process: The system allows admins to create, update, or delete user data.
Output: User accounts are modified based on admin actions.

2. Content Moderation:

Input: Admins review posts or comments in the community.

Process: The system provides content moderation tools to edit or remove posts.
Output: The community forum is updated based on moderation actions.

3. AI Model Monitoring:
Input: Admins request monitoring of AI models.
Process: The system shows the status and performance of AI models.
Output: Admins receive real-time monitoring data of AI performance metrics.

Data Flows Community Platform:
Community Forum, Meal Planning Tool, Nutrition Tracking, Custom Dietary Preferences.

1. Community Forum:
Input: Users post discussions or questions in the community forum.
Process: The system stores user posts in the community database.
Output: Display posts to other community members.

2. Meal Planning Tool:
Input: Users enter dietary requirements or preferences.
Process: The system generates meal plans based on user inputs.
Output: Display a personalized meal plan.

3. Nutrition Tracking:
Input: Users enter consumed food items.
Process: The system tracks nutritional intake based on user input.
Output: Display nutrition tracking summary with relevant information.

4. Custom Dietary Preferences:
Input: Users provide specific dietary preferences.
Process: The system saves the custom preferences for future recommendations.
Output: Custom preferences are applied to AI-generated suggestions

Data Flows AI Recommendation System:
Feedback Loop, Suggestion History, Integration with Nutrition Data, Dynamic Learning, Real-time Updates.

1.Feedback Loop:
Input: Users provide feedback on meal suggestions.
Process: The system sends the feedback to the AI system for learning.
Output: The AI system updates its models to improve future recommendations.

2. Suggestion History:

Input: The system requests previous suggestions from the AI.
Process: The AI retrieves stored suggestion history.
Output: Suggestion history is provided to the user profile.

3. Integration with Nutrition Data:

Input: Nutrition data is fed into the AI system.
Process: The AI system integrates this data for more accurate suggestions.
Output: Improved AI-generated suggestions are displayed to the users.

4.Dynamic Learning:

Input: User interactions and feedback.
Process: The AI model updates itself based on new inputs to enhance learning.
Output: More dynamic, personalized suggestions in real-time.

Data Flow Diagram (DFD - Level 1):

- **Figure 1:** *User Data Flow Diagram* - The NutriAI: Meal Dietary Planner user flow starts with sign-up or login, followed by access to the dashboard for reports, AI chat, and password reset via OTP if needed. Users can also use the BMI calculator, nutrition guide, and participate in the community forum, concluding the flow after exploring all features.
- **Figure 2.** *Admin Data Flow Diagram* - The admin flow begins with login or sign-up, followed by access to reports and OTP-based password reset if needed, with admin approval required for new sign-ups. Admins manage active logs, AI settings, and forum content, including posts, blogs, and privacy settings, before concluding their actions.

Figure 1: *User Data Flow Diagram*

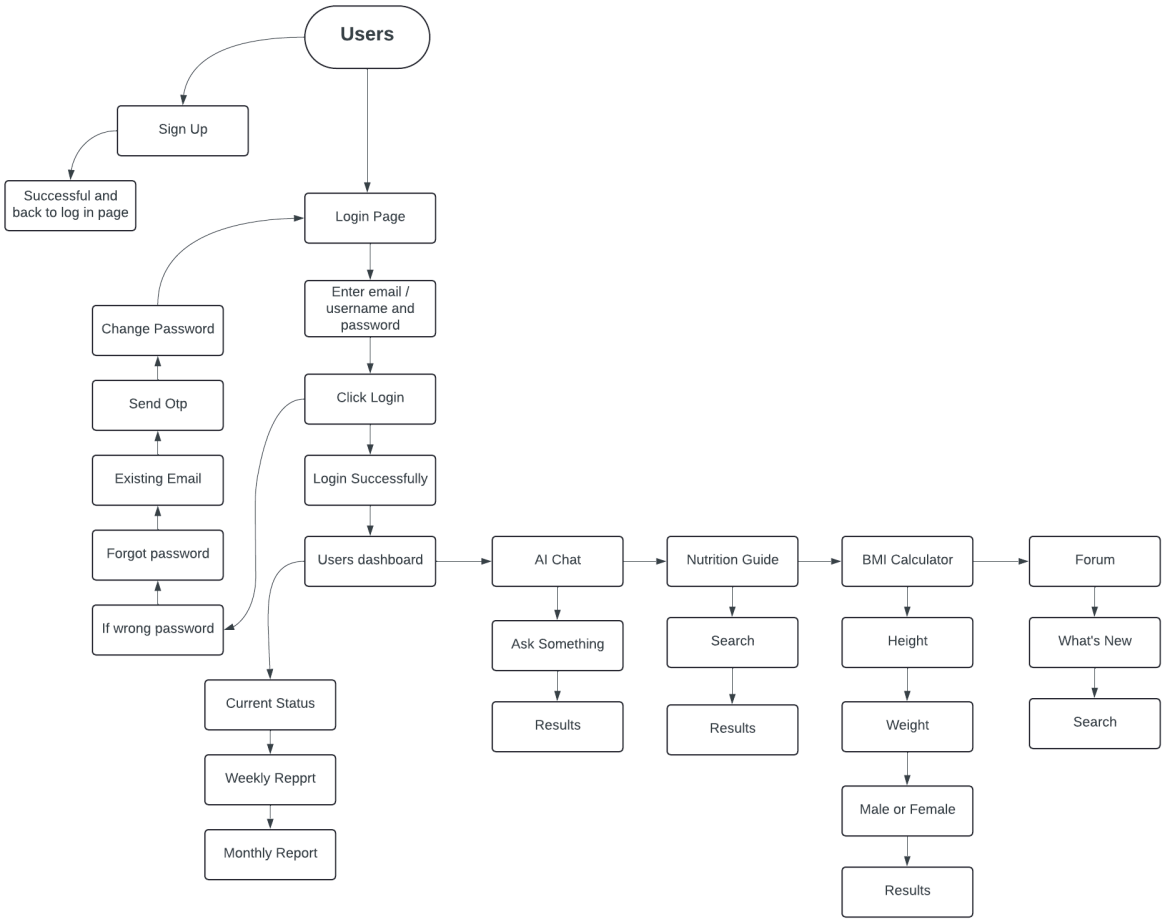
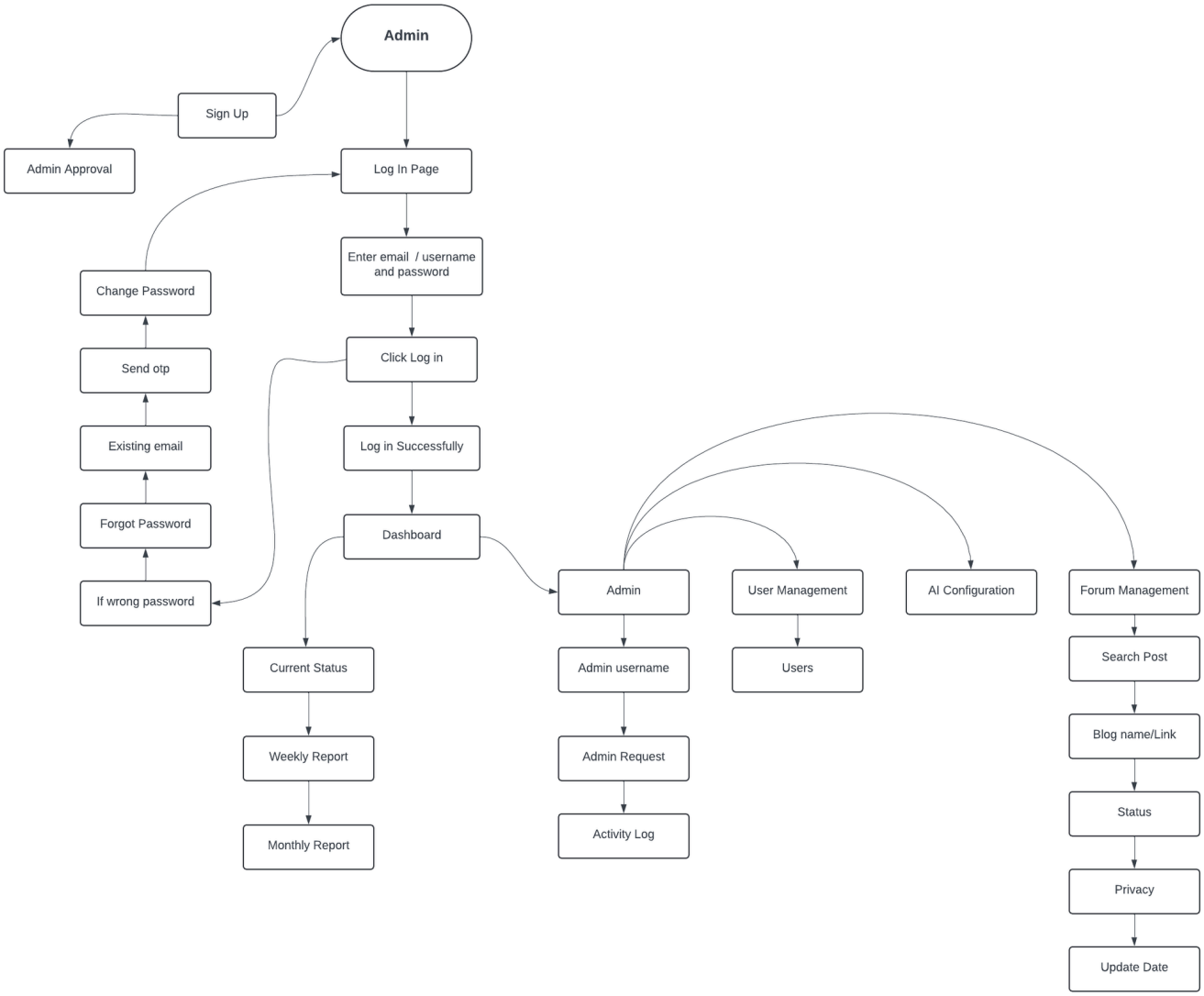


Figure 2. *Admin Data Flow Diagram*



C. Data Structure

Entities and Relationships:

1. **User**: Stores information about users of the system.
2. **UserProfile**: Provides additional profile information for each user.
3. **AISuggestion**: Stores AI-generated dietary suggestions.
4. **SuggestionHistory**: Records the history of AI suggestions viewed by users.

5. **BMICalculator**: Stores BMI calculation data for users.
6. **Report**: Stores weekly or monthly dietary reports for users.
7. **CommunityPost**: Stores information about posts and comments in the community forum.
8. **Admin**: Stores information about administrators of the system.
9. **UserManagement**: Logs administrative actions taken on users.
10. **ActivityLog**: Stores logs of actions performed by users.
11. **AIModelMonitoring**: Stores monitoring information for AI models.

Database Tables:

1. Users

Field Name	Data Type	Description
user_id	INT(PK)	Unique identifier for the user.
First_name	VARCHAR(100)	First name of the user.
Middle_name	VARCHAR(100)	Middle name of the user.
Last_name	VARCHAR(100)	Last name of the user.
Email	VARCHAR(250)	Unique email of the user.
password	VARCHAR(255)	Password of the user.
Birthdate	DATE	Birthdate of the user.
Age	INT	Age of the user
Sex	VARCHAR(20)	Gender of the user.(Male/Female)
Contact_Number	VARCHAR(15)	Contact number of the user.
created_at	TIMESTAMP	The date and time when the user was created.
last_login	TIMESTAMP	The last login time of the user.

Figure 3. *User Database Structure*

2.UserProfile

Field Name	Data Type	Description
profile_id	INT(PK)	Unique identifier for the user profile.
user_id	INT(FK)	The ID of the associated user.
height_cm	FLOAT	Height of the user in centimeters.
weight_kg	FLOAT	Weight of the user in kilograms.
BMI	FLOAT	BMI of the user in decimal.

Figure 4. *UserProfile Database Structure*

3. AISuggestions

Field Name	Data Type	Description
suggestion_id	INT(PK)	Unique identifier of suggestion.
user_id	INT(FK)	The ID of the associated user.
suggestion_text	TEXT	The text of the AI generated..
created_at	TIMESTAMP	The date and time when the suggestion was created.

Figure 5. *AI Suggestion Database Structure*

4. SuggestionHistory

Field Name	Data Type	Description
history_id	INT(PK)	Unique identifier of the history record.
user_id	INT(FK)	The ID of the associated user..
suggestion_id	INT(FK)	The ID of the viewed suggestion.
viewed_at	TIMESTAMP	The date and time when the suggestion was viewed.

Figure 6. *Suggestion History Database Structure*

5. BMICalculator

Field Name	Data Type	Description
bmi_id	INT(PK)	Unique identifier of the BMI record.
user_id	INT(FK)	The ID of the associated user.
bmi_value	FLOAT	The calculated BMI value.
status	VARCHAR(100)	The status of the user's BMI.
calculated_at	TIMESTAMP	The date and time when the BMI was calculated.

Figure 7. *BMI Calculator Database Structure*

6. Reports

Field Name	Data Type	Description
report_id	INT(PK)	Unique identifier for the report.
user_id	INT(FK)	The ID of the associated user.
report_type	ENUM	The type of report('weekly'. 'monthly')
report_content	TEXT	The content of the report.
created_at	TIMESTAMP	The date and time when the report was created.

Figure 8. *Report Database Structure*

7. CommunityPosts

Field Name	Data Type	Description
post_id	INT(PK)	Unique identifier of the community post.
user_id	INT(FK)	The ID of the associated user.

post_title	VARCHAR(255)	The title of the post.
post_content	TEXT	The content of the post.
created_at	TIMESTAMP	The date and time when the post was created.

Figure 9. *Community Post Database Structure*

8. CommunityComments

Field Name	Data Type	Description
comment_id	INT(PK)	Unique identifier of community comment.
post_id	INT(FK)	The ID of the associated post.
user_id	INT(FK)	The ID of the associated user.
comment_text	TEXT	The content of the comment.
created_at	TIMESTAMP	The date and time when the comment was created.

Figure 10. *Community Comments Database Structure*

9. Admins

Field Name	Data Type	Description
AdminUsername	INT(PK)	Unique username of the admin.
FirstName	VARCHAR(200)	First name of the admin.
LastName	VARCHAR(200)	Last name of the admin.
MiddleName	VARCHAR(200)	Middle name of the admin.
Email	VARCHAR(200)	Unique email of admin
Password	VARCHAR(200)	Password of the admin.
RePassword	VARCHAR(200)	Confirmation of the Password.

created_at	TIMESTAMP	The date and time when the admin was created.

Figure 11. *Admins Database Structure*

10. UserManagement

Field Name	Data Type	Description
action_id	INT(PK)	Unique identifier of the action.
AdminUsername	INT(FK)	The username of the associated admin
user_id	INT(FK)	The ID of the associated user
action_type	VARCHAR(100)	The type of action performed.
action_description	TEXT	A description of the action.
action_timestamp	TIMESTAMP	The date and time when the action was performed .

Figure 12. *UserManagement Database Structure*

11. ActivityLogs

Field Name	Data Type	Description
log_id	INT(PK)	Unique identifier of the log
user_id	INT(FK)	The ID of the associated user
action	VARCHAR(255)	The action performed by the user.
action_timestamp	TIMESTAMP	The date and time when the action was performed

Figure 13. *ActivityLogs Database Structure*

12. AIModelMonitoring

Field Name	Data Type	Description
monitor_id	INT(PK)	Unique identifier of the AI model monitoring

		record
AdminUsername	INT(FK)	The Username of the associated admin
model_status	VARCHAR(100)	The status of the AI model
updated_at	TIMESTAMP	The date and time when the status was updated

Figure 14. *AIModelMonitoring Database Structure*

c. Relationship

- **Users and UserProfile:**
  - The **Users** and **UserProfile** tables have a one-to-one relationship. When a user registers, their basic information is stored in the Users table. Additional details about the user, such as their first name, last name, age, height, and weight, are stored in the UserProfile table. The user_id field links these two tables, allowing for detailed user profile management.

- **Users and AISuggestions:**
  - The **Users** and **AISuggestions** tables have a one-to-many relationship. A single user can receive multiple AI-generated suggestions, which are stored in the AISuggestions table. The user_id field connects each suggestion to a specific user, allowing the system to provide personalized recommendations over time.

- **AISuggestions and SuggestionHistory:**
  - The **AISuggestions** and **SuggestionHistory** tables have a one-to-many relationship. Each suggestion generated by the system can be viewed multiple times by the user, and each viewing instance is logged in the SuggestionHistory table. This table also references the Users table, allowing for tracking of which user viewed which suggestion and when.

- **Users and BMICalculator:**
  - The **Users** and **BMICalculator** tables have a one-to-many relationship. A single user can have multiple BMI calculations over time, each recorded in the BMICalculator table. The user_id field

connects each BMI entry to a specific user, helping track changes in a user's health status.

- **Users and Reports:**
  - The **Users** and **Reports** tables have a one-to-many relationship. Each user can generate multiple reports, such as weekly or monthly summaries, which are stored in the Reports table.

D. Wireframe

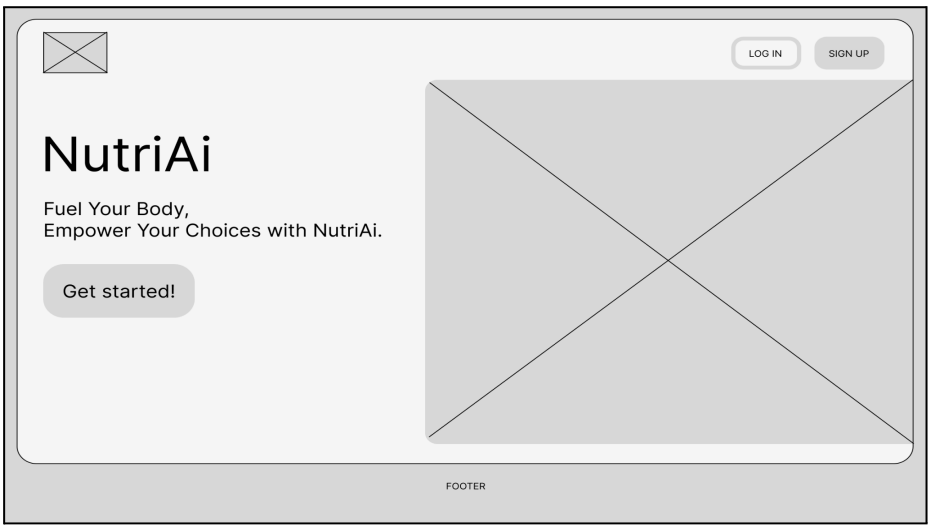


Figure 15. *Landing Page*

This low-fidelity wireframe of NutriAI's landing page features a simple, user-friendly design. It prominently displays the brand name and tagline, "Fuel Your Body, Empower Your Choices with NutriAI," to convey the platform's mission. The "Get started!" button encourages users to take action, while the "Log In" and "Sign Up" buttons are easily accessible in the header. A large visual placeholder suggests future use of engaging imagery, and a minimal footer . The overall focus is on clarity and ease of navigation to guide users toward engaging with NutriAI.

Figure16. *User Login WireFrame*

This low-fidelity wireframe shows a login page with fields for "Username" and "Password," a "Forgot password?" link, and buttons for "Log In" and "Sign Up." A large placeholder on the right is reserved for an image or graphic to enhance visual appeal.

Figure 17. *User Signup Wireframe*

This low-fidelity wireframe displays a sign-up page with fields for personal details, including name, birthdate, age, sex, email, contact number, and password, along with a "Sign Up" button for submission. The layout is simple and organized to efficiently gather user information.

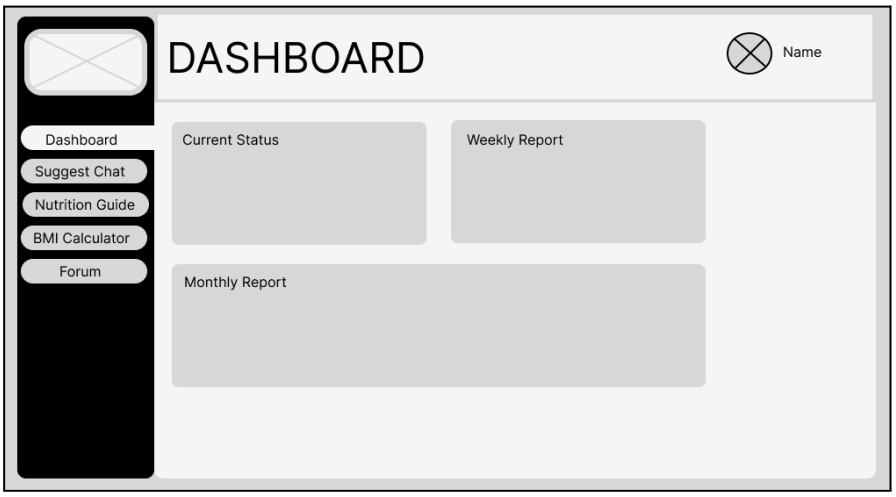


Figure 18. *User Dashboard Wireframe*

This low-fidelity wireframe shows a user dashboard with a sidebar for easy navigation, including options like "Dashboard," "Suggest Chat," "Nutrition Guide," and more. The main area features "Current Status," "Weekly Report," and "Monthly Report" sections, with a profile placeholder in the top right corner.

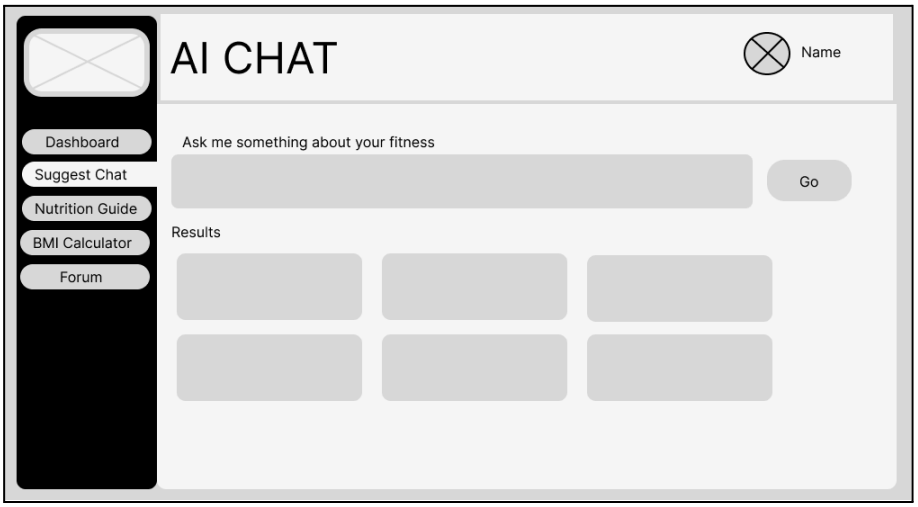


Figure 19. *AI Chat Wireframe*

This low-fidelity wireframe shows the AI Chat section of the user dashboard, with a prompt for fitness-related queries and a "Go" button to initiate the search. Below, placeholders display the AI-generated results.

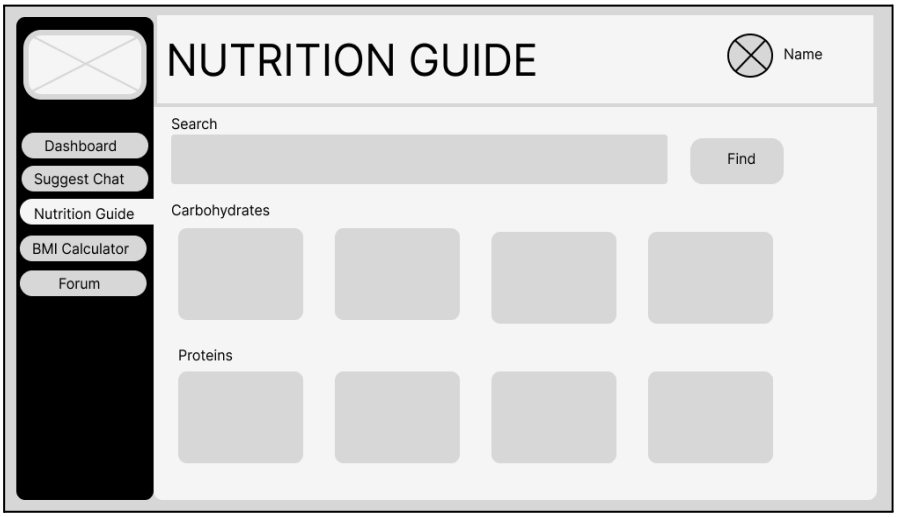


Figure 19. *Nutrition Guide Wireframe*

This low-fidelity wireframe shows the Nutrition Guide section of the dashboard, featuring a search bar for nutrients or food items with a "Find" button. Below, placeholders provide information on "Carbohydrates" and "Proteins" to help users understand their nutrition easily.

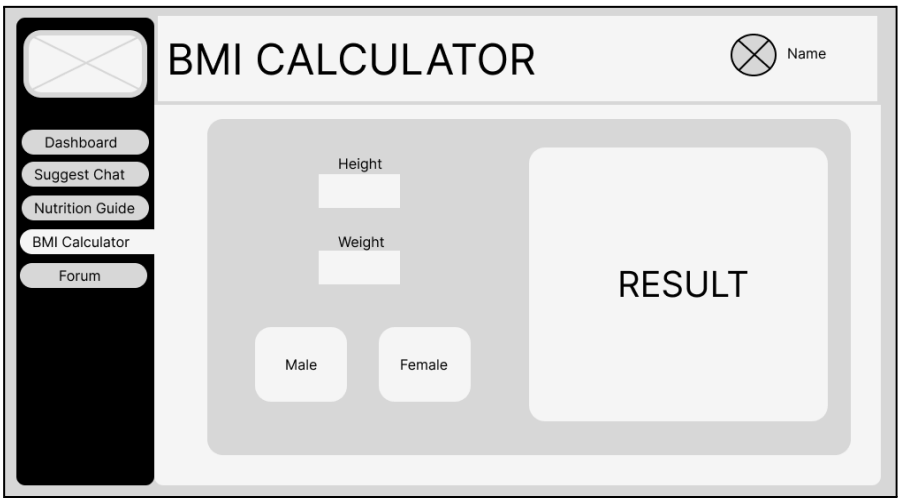


Figure 20. *BMI Calculator Wireframe*

This low-fidelity wireframe shows the BMI Calculator, where users input height, weight, and gender to calculate BMI, with results displayed in the "Result" section. The design is clean and straightforward for quick BMI calculation and health insights.

Figure 21.1. *Forum 1 Dashboard Wireframe*

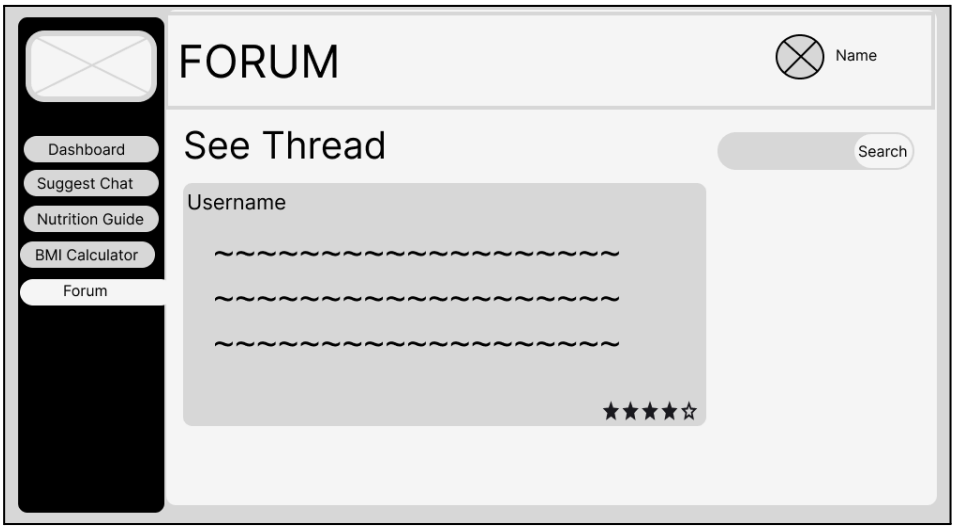


Figure 21.2. *Forum 2 Dashboard Wireframe*

Forum 1: This screen is the main forum page displaying the latest discussions under "What's New?" Users can browse threads or search specific topics using the "Search" bar, enabling them to stay updated on recent conversations.

Forum 2: This screen represents an individual thread view. It displays a username, followed by the user's post or comment content, and includes a rating system at the bottom to allow users to rate the post.

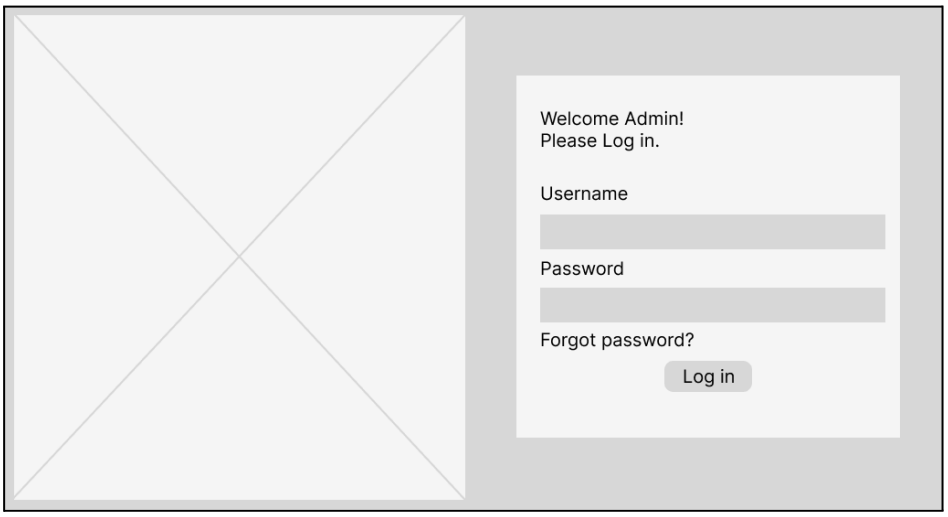


Figure 22. *Admin LogIn Wireframe*

This low-fidelity wireframe depicts the admin login page, featuring a "Welcome Admin!" prompt, fields for "Username" and "Password," a "Forgot password?" link, and a "LogIn" button. A placeholder on the left can be used for an image or brand logo.

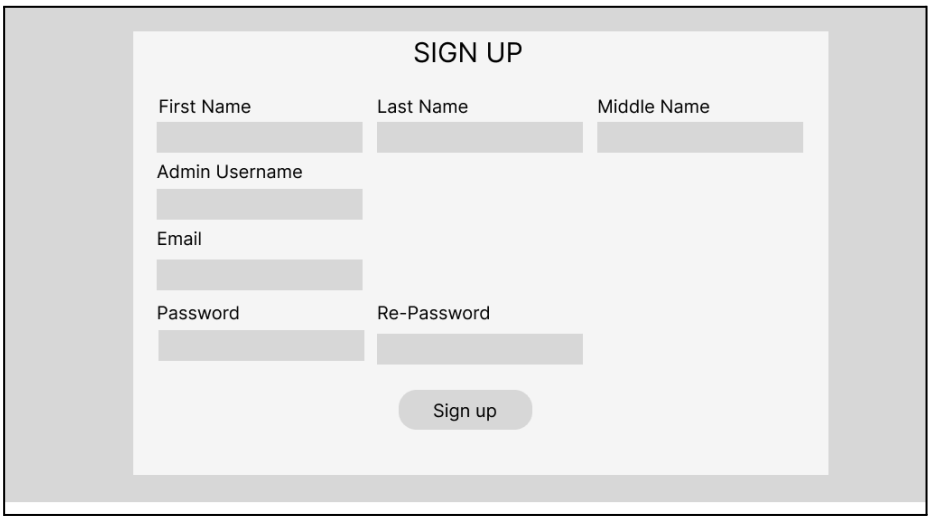


Figure 23. *Admin Signup Wireframe*

This low-fidelity wireframe shows the admin sign-up page, including fields for name, username, email, password, and a "Sign Up" button for registration. The design is simple and organized to gather essential information efficiently.

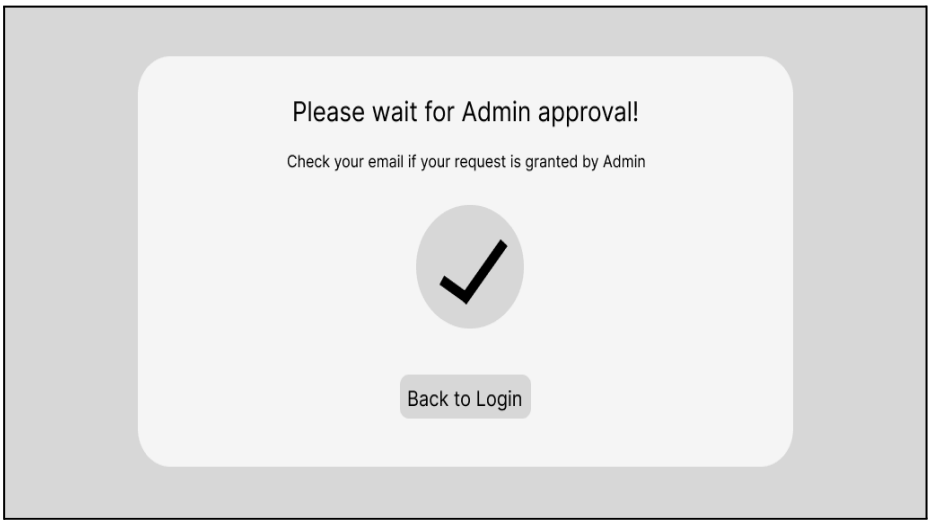


Figure 24. *Admin Signup Request Wireframe*

This low-fidelity wireframe shows the admin sign-up approval page with a message instructing users to await approval and check their email. A check mark symbol indicates successful submission, and a "Back to Login" button lets users return to the login page.

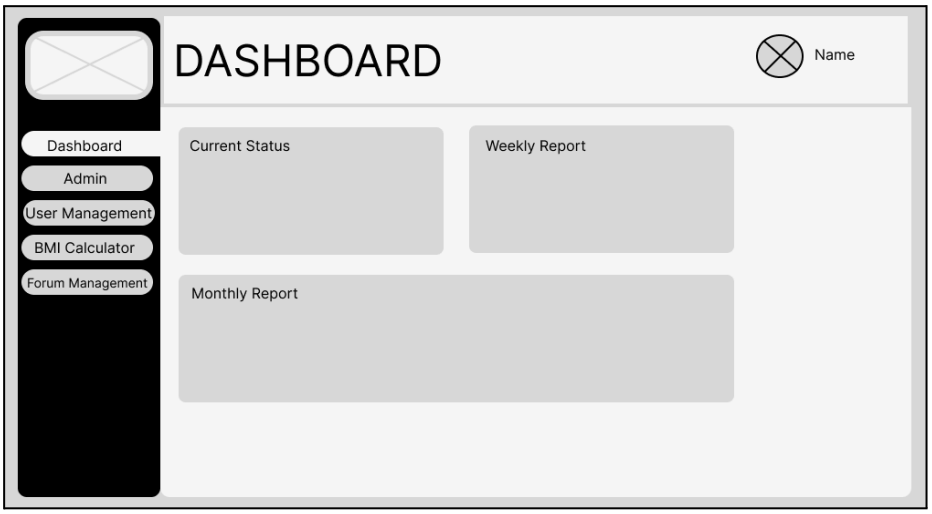


Figure 25. *Admin Dashboard Wireframe*

This low-fidelity wireframe shows the admin dashboard with a sidebar for navigation to sections like "Dashboard," "Admin," "User Management," and more. The main content area features "Current Status," "Weekly Report," and "Monthly

Report" for monitoring, with an admin profile section in the top right corner.

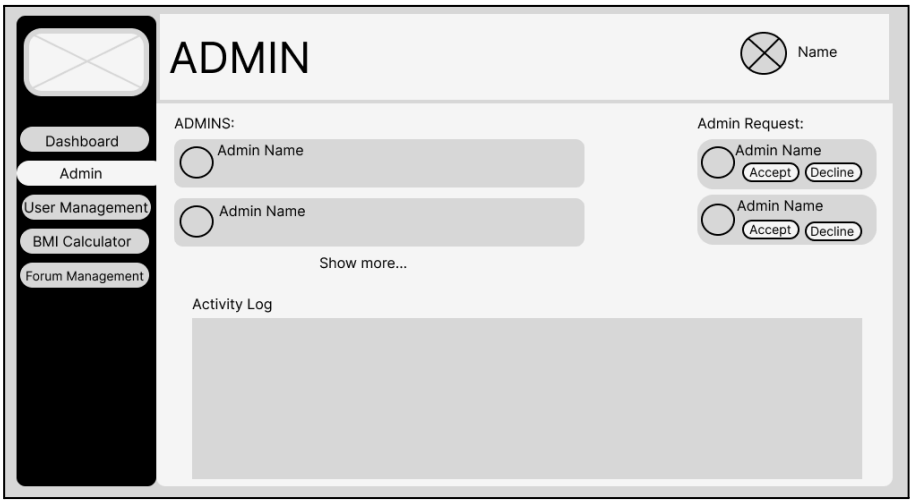


Figure 26. *Admin Wireframe*

This low-fidelity wireframe represents the admin management section of the dashboard.The main area lists current admins and provides options for viewing more. On the right, new admin requests can be accepted or declined. Below is an "Activity Log" to track recent admin actions.

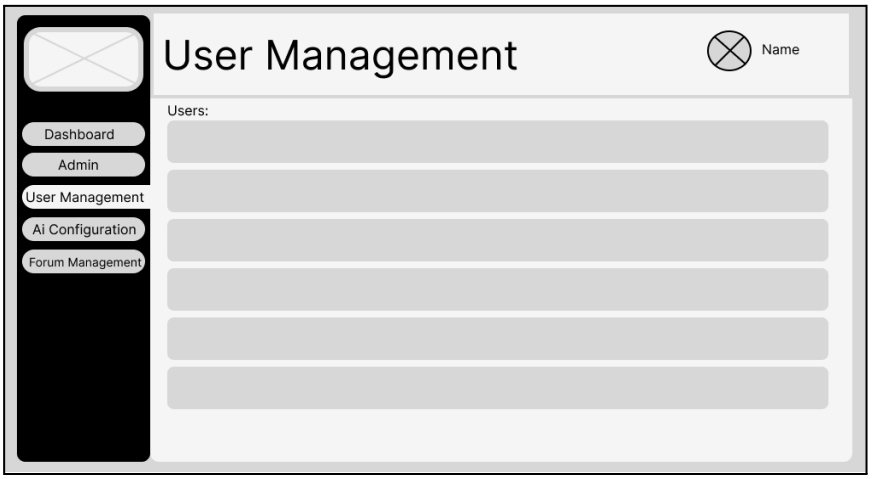


Figure 27. *Admin User Management Wireframe*

This low-fidelity wireframe represents the User Management section of the admin dashboard. The main area lists users, each represented by individual rows, providing a space for viewing and managing user details.

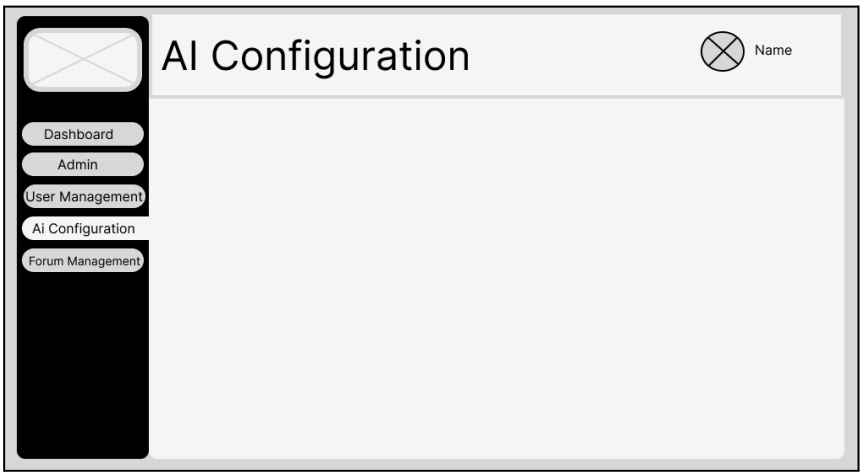


Figure 28. *AI Configuration Wireframe*

This low-fidelity wireframe shows the AI Configuration section of the admin dashboard for managing AI features linked to user AI chat. The main area is reserved for tools or settings to customize AI behaviors, allowing admins to ensure relevant responses that align with user needs.

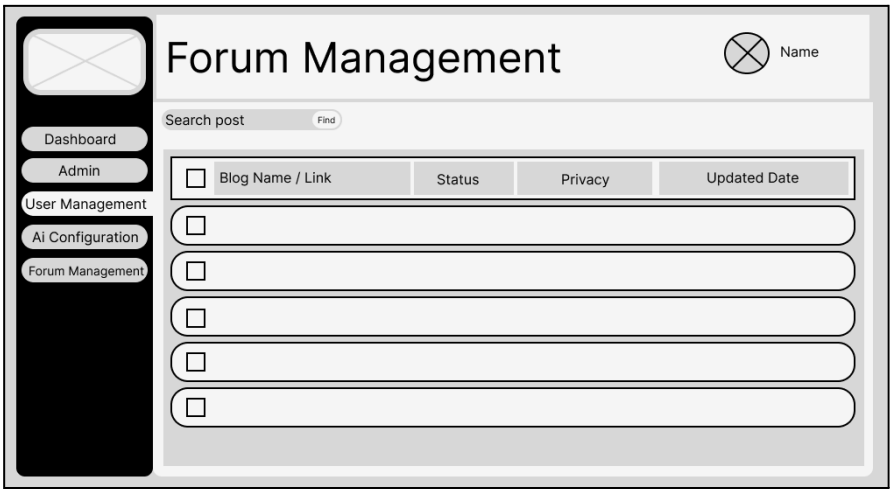


Figure 29. *Forum Management Wireframe*

This low-fidelity wireframe represents the Forum Management section of the admin dashboard. It features a search bar at the top to locate specific posts and a list format for managing forum entries.This design allows admins to easily manage posts, update statuses, adjust privacy settings, and monitor content effectively.

V.PROCEDURE

This procedure will serve as a guideline for the development process for NutriAI: Meal Dietary Planner and AI - Generated Management System.. It includes setting up the directory structure, creating and designing the database, and connecting different system components. The development phase focuses on coding features like the User Dashboard, Admin Dashboard and Login and SignUp User/Admin. This procedure does not yet cover the integration of an Admin Management System, Community Platform and AI - Powered Recommendation System .

1. Install xampp.
2. Install Visual Studio Code.
3. Open Local Disk on your PC/Laptop.
4. Open xampp folder.
5. Open htdocs.
6. Create your own folder
8. Cut the css and js folder from the extracted file then paste it inside the folder you've created on the htdocs.
9. Open VS Code.
10. Hit Ctrl + K Ctrl + O to open a folder or just click the "Open folder" button. \
11. Navigate to find your created folder in the Local Disk/xampp/htdocs.
12. Create a file named index.html which will serve as the landing page.
13. Add the codes for the landing page.
14. Create a folder named "assets".
15. Create a file named “userlogin..php” for the login of user and connect.php” for the configuration of the database connection inside of the "assets" folder

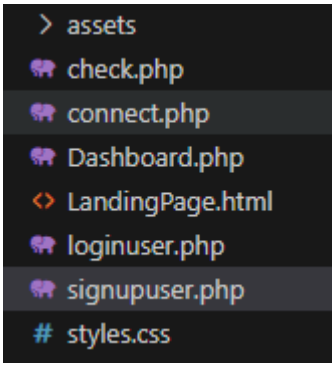
connect.php > ...

```php
<?php
$host = "localhost";
$admin = "root";
$pass = "";
$db = "nutriai"; // Your database name

$conn = new mysqli($host, $admin, $pass, $db);

// Check if the connection was successful
if ($conn->connect_error) {
    die("Connection Failed: " . $conn->connect_error);
}
?>
```

16. Add codes to each file.
17. Link the "userlogin.php" to the files within the "assets" folder using ‘href="assets/yourr_file"’
18.In the “assets” folder create another php file named “usersignup.php” to redirect a designated signup for the user.



19. Connect the “usersignup.php” to the database using “ require_once '../db_config.php'; ”.
20.In your Xampp, create a database named “nutriai”, after that, create a database table (shown on the database structure).
21. Once done, put the necessary codes to validate the credentials inputted.
22. If the credentials are okay (in the signupuser) you will redirect the page to the user login.
23. In the log in you have an option to log in or change password.
24. If you pick log login the user can now login using their created account, otherwise they must sign up first.
25. Once done, use the form to login and don't forget to connect it to the database to authenticate the credentials submitted.

- **CREATE .**This is when the user registers

```
if ($result->num_rows > 0) {
    echo "Email Address Already Exists!";
} else {
    // Insert new record using a prepared statement
    $insertQuery = $conn->prepare("INSERT INTO users (first_name, last_name, middle_name, birthdate, age, sex, contact_number, email, password)
                                   VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
    $insertQuery->bind_param("ssssissss", $first_name, $last_name, $middle_name, $birthdate, $age, $sex, $contact_number, $email, $hashedPassword);

    if ($insertQuery->execute()) {
        // Redirect after successful signup
        header("Location: loginuser.php");
        exit();
    } else {
        echo "Error: " . $insertQuery->error;
    }
```

- **READ**.This is when the admin collect those user, who register.

```
$checkemail = $conn->prepare("SELECT * FROM users WHERE email = ?");
$checkemail->bind_param("s", $email);
$checkemail->execute();
$result = $checkemail->get_result();

if ($result->num_rows > 0) {
    // Fetch user data
    $user = $result->fetch_assoc();

    // Verify the password
    if (password_verify($password, $user['password'])) {
        // Set session variables
        $_SESSION['user_id'] = $user['id'];
        $_SESSION['email'] = $user['email'];
        $_SESSION['first_name'] = $user['first_name'];
```

- **UPDATE.** This is when the users forgot their password or number, they can easily update it.

```
if ($result->num_rows > 0) {
    // Email exists, proceed to update the contact number
    $updateQuery = $conn->prepare("UPDATE users SET contact_number = ? WHERE email = ?");
    $updateQuery->bind_param("ss", $new_contact_number, $email);

    if ($updateQuery->execute()) {
        echo "Contact number updated successfully!";
    } else {
        echo "Error updating contact number: " . $updateQuery->error;
    }
} else {
    echo "Email does not exist!";
}
```

- **DELETE.**This is when the users delete their account, the account will be deleted in the database.

```php
// Prepare a statement to delete the user's account
$deleteQuery = $conn->prepare("DELETE FROM users WHERE email = ?");
$deleteQuery->bind_param("s", $email);

if ($deleteQuery->execute()) {
    echo "Account deleted successfully!";
    // Optionally, redirect to the homepage or a confirmation page
    header("Location: goodbye.php");
    exit();
} else {
    echo "Error: " . $deleteQuery->error;
}
```