BITE BRIGHT PHASE 2

**Introduction**

**1.1 Purpose of the System**

The purpose of **Bite Bright** is to provide a modern diet management solution that goes beyond traditional calorie counting. Unlike systems that often foster food-related anxiety and unsustainable eating habits, **Bite Bright** focuses on promoting emotional well-being and encouraging healthier, sustainable eating patterns. The system is designed to track users' food intake while offering personalized recommendations tailored to their unique needs, preferences, and goals. By incorporating emotional health into dietary planning, **Bite Bright** seeks to empower users to build a positive and lasting relationship with food.

The system will:

* Track meals, nutrients, and eating habits effortlessly.
* Provide personalized, sustainable dietary recommendations.
* Support users' emotional well-being by addressing mental health aspects of dieting.
* Encourage long-term healthy eating habits over short-term restrictive diets.

**1.2 Design Goals**

1. **Holistic User Wellness**:
   * Promote a balanced approach to diet management by integrating emotional health and physical well-being.
   * Minimize stress and anxiety related to food tracking.
2. **Personalization**:
   * Use advanced algorithms to generate recommendations based on users’ preferences, dietary restrictions, goals, and emotional status.
   * Adapt to users’ progress over time to ensure relevancy and effectiveness.

**High-Level Software Architecture**

**2.1 Subsystem Decomposition**

The system is divided into several subsystems to ensure modularity and scalability. Each subsystem is responsible for specific functionalities:

1. **User Management Subsystem**:
   * Handles user registration, login, and profile management.
   * Tracks user preferences, emotional status, and goals.
2. **Meal Tracking Subsystem**:
   * Allows users to log meals and nutritional intake.
   * Tracks calories, nutrients, and meal types (e.g., breakfast, lunch).
3. **Recommendation Engine**:
   * Generates personalized meal suggestions based on user data, goals, and emotional status.
   * Adapts recommendations dynamically based on user feedback and progress.
4. **Analytics and Reporting Subsystem**:
   * Generates reports on user compliance rates, progress toward goals, and overall well-being.
   * Provides insights into trends and areas of improvement.
5. **Integration Subsystem**:
   * Connects with wearable devices and external APIs for data such as activity levels and nutritional information.
   * Facilitates communication with third-party systems like fitness apps.
6. **Security and Access Control Subsystem**:
   * Ensures data protection and enforces secure authentication.
   * Manages access control for different user roles (e.g., admin vs. regular users).

**2.2 Hardware/Software Mapping**

The system runs on a client-server architecture with the following components:

* **Client Side**:
  + Platform: Mobile and web applications.
  + Technology Stack: HTML, CSS, JavaScript (React.js for web), Flutter for mobile apps.
  + Functionality: User interface, meal logging, and report viewing.
* **Server Side**:
  + Platform: Cloud-based backend (e.g., AWS, Azure, or Google Cloud).
  + Technology Stack: Node.js or Django (backend framework), PostgreSQL or MongoDB (database).
  + Functionality: Core business logic, data processing, and API handling.
* **Wearable Devices**:
  + Devices like Fitbit or Apple Watch for activity and health data collection.
  + Communicates with the backend via APIs.

Use Cases

Use cases

**1. Track Food Intake**

**Actor(s):** User  
**Precondition:** The user is logged into the system and has access to the food intake logging interface.  
**Main Flow:**

1. The user inputs meal details, including food items and portion sizes.
2. The system validates the input for completeness and correct formatting.
3. The system logs the entered data.
4. The system confirms the successful entry of the meal.

**Alternative Flow:**

* **Invalid Input Data:**
  + If the input data is missing or incorrectly formatted, the system displays an error message.
  + The user is prompted to re-enter the data.

**Postcondition:** The meal details are successfully stored in the system.

**2. Monitor Health Data**

**Actor(s):** Wearable Device  
**Precondition:** The wearable device is connected and has data available for synchronization.  
**Main Flow:**

1. The wearable device syncs health data (e.g., heart rate, activity levels) with the system.
2. The system processes and stores the synced data.
3. The user's health dashboard is updated with the latest information.

**Alternative Flow:**

* **Sync Failure:**
  + If the wearable device fails to sync, the system notifies the user of the issue.
  + The system retries the synchronization process after a defined interval.

**Postcondition:** The health data is successfully synced and reflected in the user's health dashboard.

**3. Categorize Meals**

**Actor(s):** User  
**Precondition:** The user has logged at least one meal and has access to the meal categorization interface.  
**Main Flow:**

1. The user categorizes meals as "healthy" or "unhealthy."
2. The system validates the categorization using predefined rules.
3. The categorized data is stored for future analysis.

**Alternative Flow:**

* **Uncategorized Meal:**
  + If a meal cannot be categorized automatically, the system suggests likely categories.
  + The user confirms or overrides the suggested categorization.

**Postcondition:** Meal categorizations are successfully saved in the system.

**4. Provide Personalized Diet Recommendations**

**Actor(s):** User  
**Precondition:** The user has logged food intake and health data, and sufficient data is available for analysis.  
**Main Flow:**

1. The system analyzes the user's health data and food intake.
2. The system generates personalized diet recommendations using the 80/20 principle.
3. The system displays the recommendations to the user.

**Alternative Flow:**

* **Insufficient Data:**
  + If insufficient data is available, the system notifies the user and prompts them to provide the missing information (e.g., logging meals or syncing health data).

**Postcondition:** Personalized diet recommendations are successfully delivered to the user.

**5. Send Real-Time Feedback**

**Actor(s):** User  
**Precondition:** The system is actively monitoring the user's adherence to the 80/20 principle.  
**Main Flow:**

1. The system monitors the user's adherence to the 80/20 balance in real-time.
2. If deviations are detected, the system sends alerts or feedback to the user.

**Alternative Flow:**

* **Feedback Delivery Failure:**
  + If feedback cannot be delivered due to connectivity issues, the system queues the notification.
  + The system retries delivering the feedback once connectivity is restored.

**Postcondition:** Feedback is successfully delivered, or it is queued for future delivery.

**6. Monitor Emotional Health**

**Actor(s):** User  
**Precondition:** The user has access to the emotional health logging interface.  
**Main Flow:**

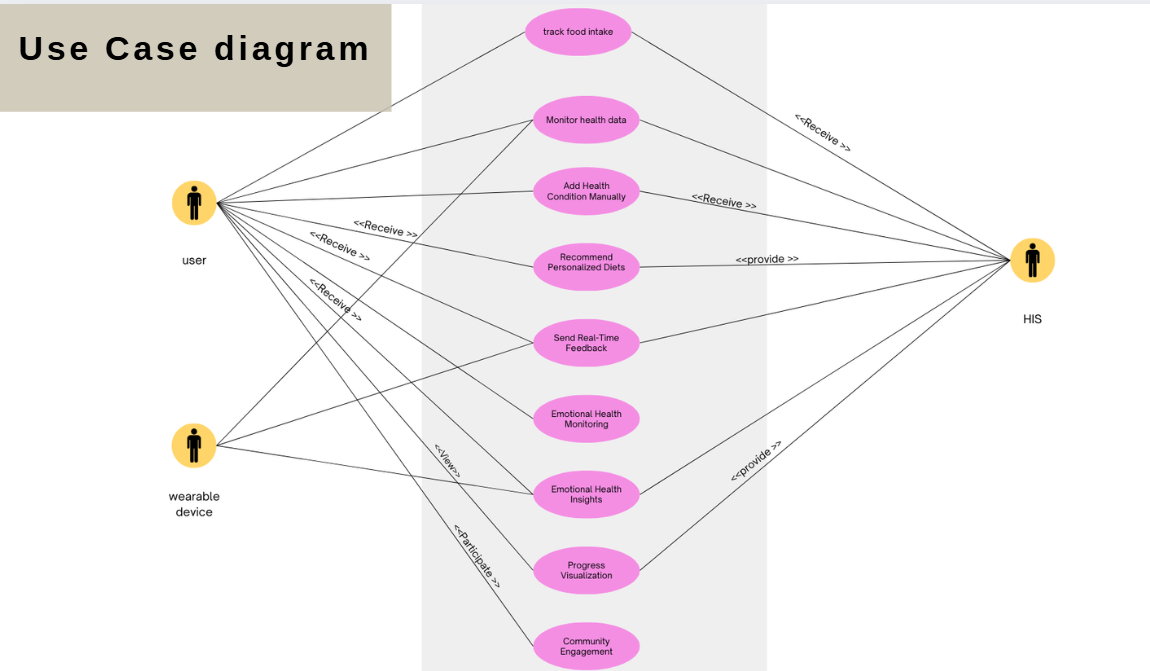
1. The user logs emotional health data, such as mood and stress levels.
2. The system tracks trends and correlates them with the user's eating patterns.
3. Insights are displayed to the user based on the analysis.

**Alternative Flow:**

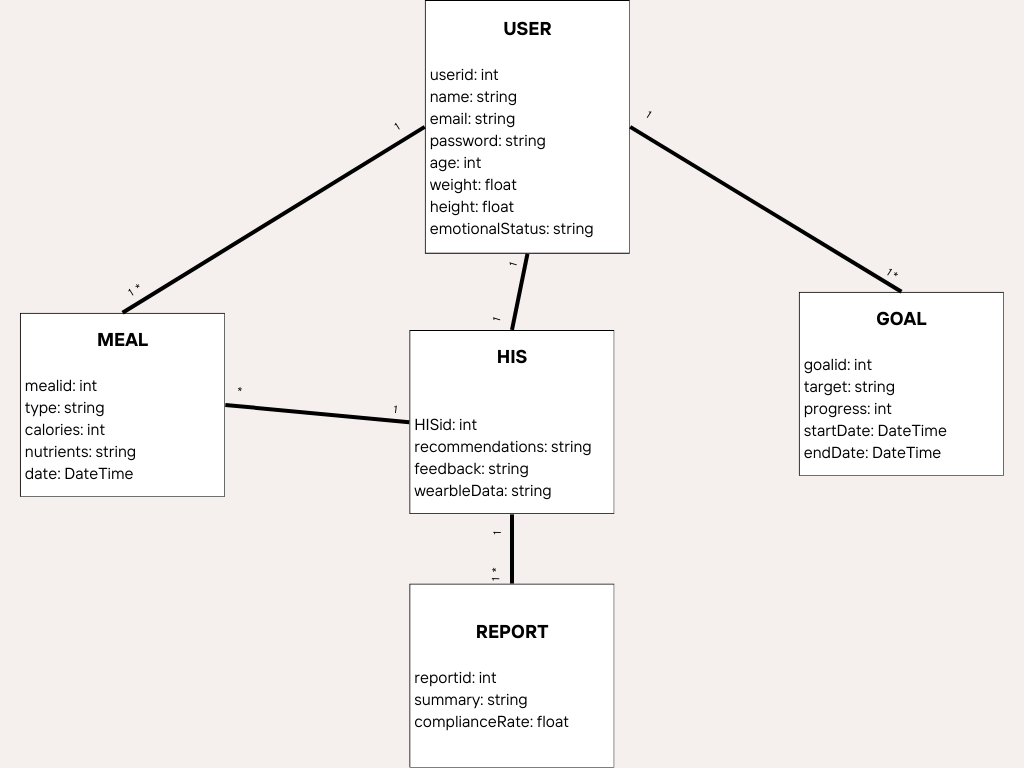
* **No Emotional Health Data Logged:**
  + If no emotional health data is logged, the system generates and provides generic insights based on general patterns and existing eating data.

**Postcondition:** Insights are successfully displayed to the user.

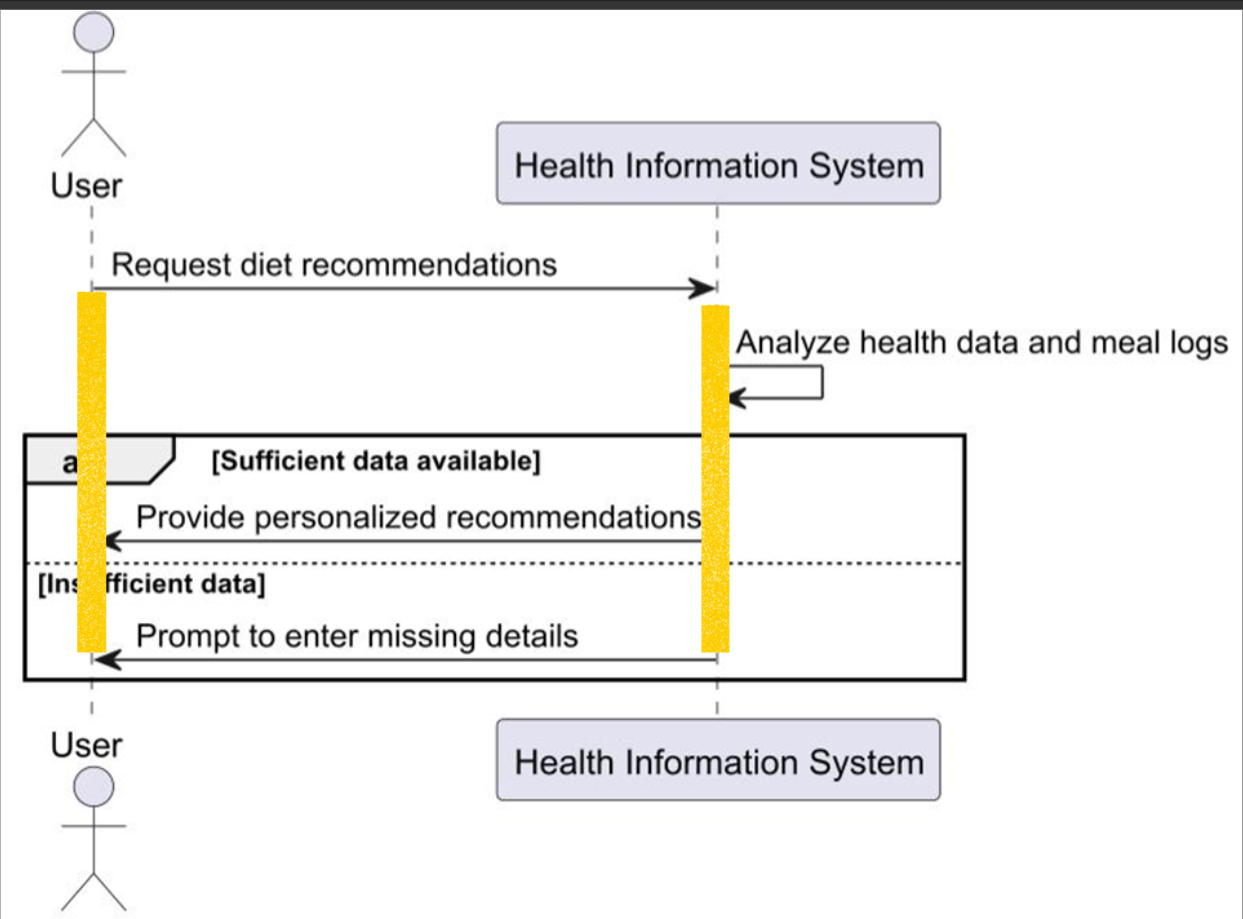
Use Case Model

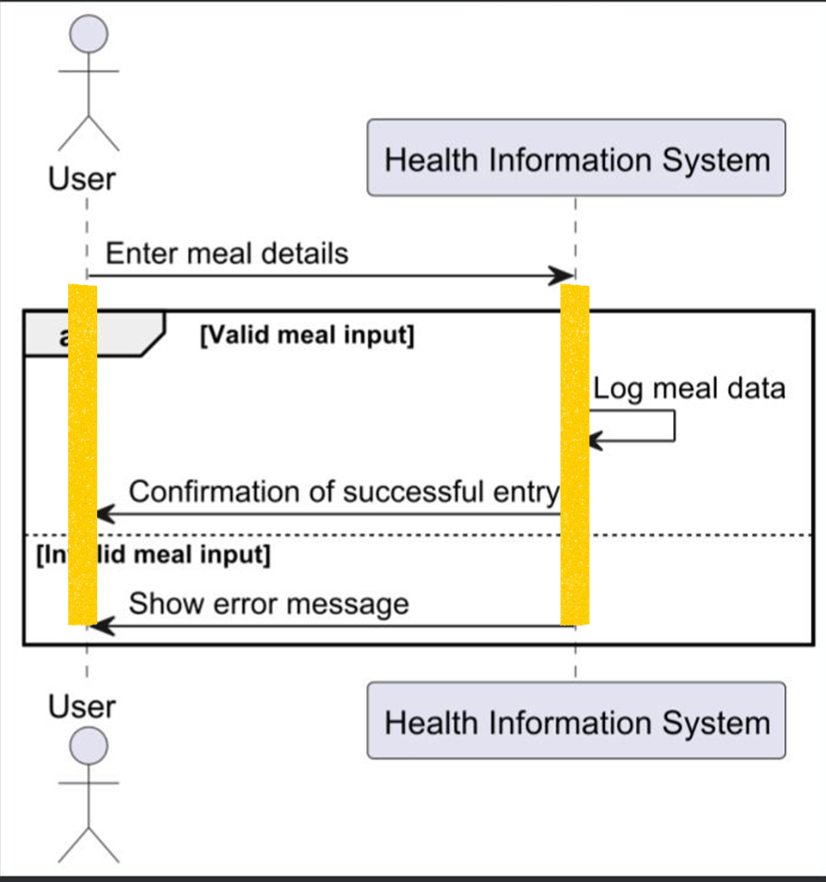


Domain Model

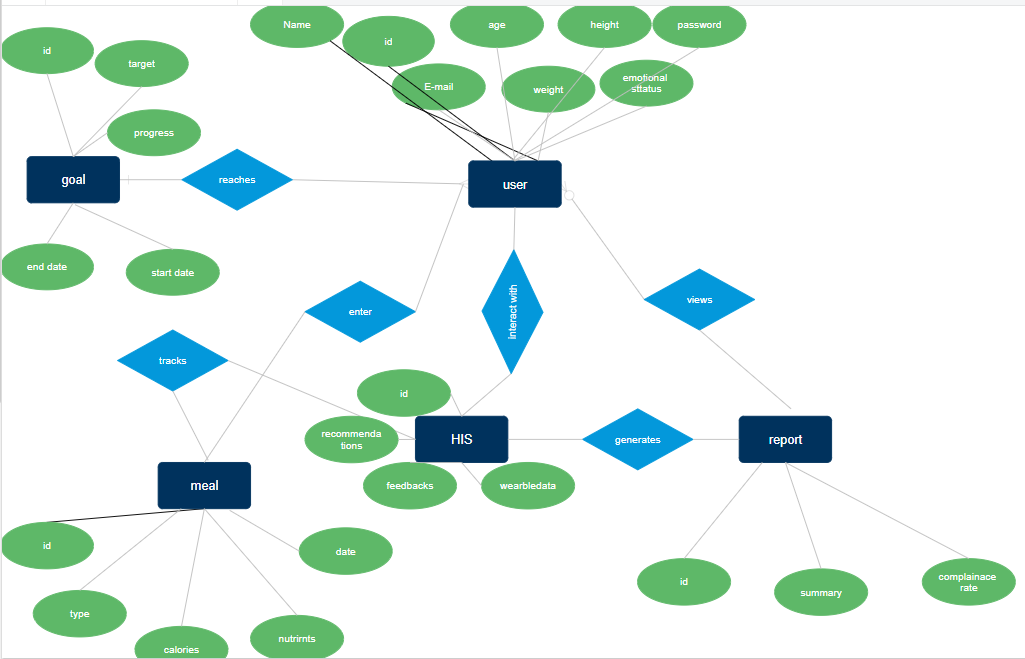


Interaction Models

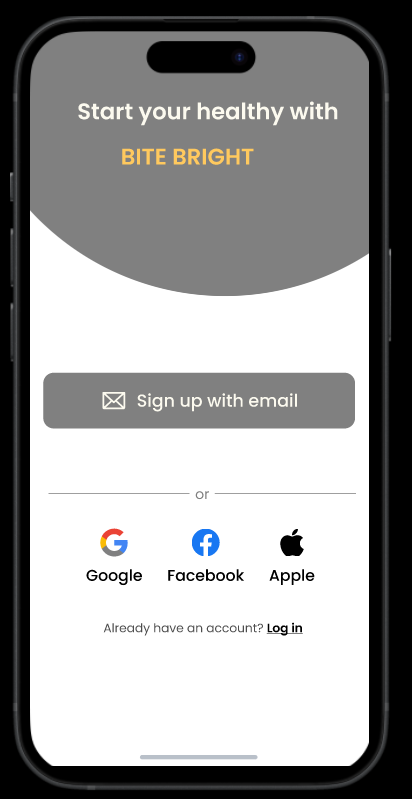
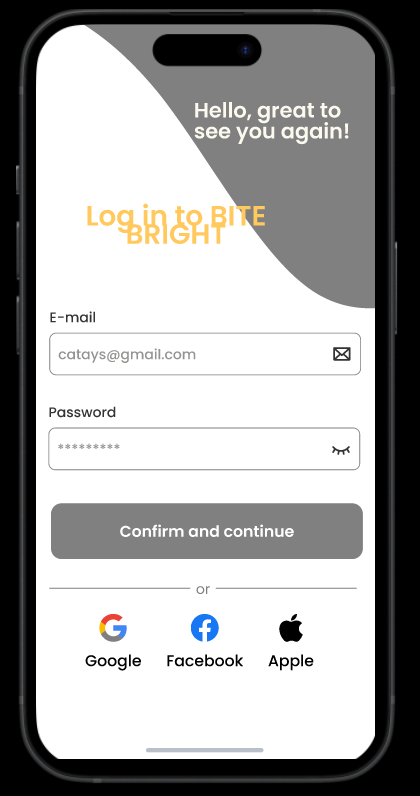


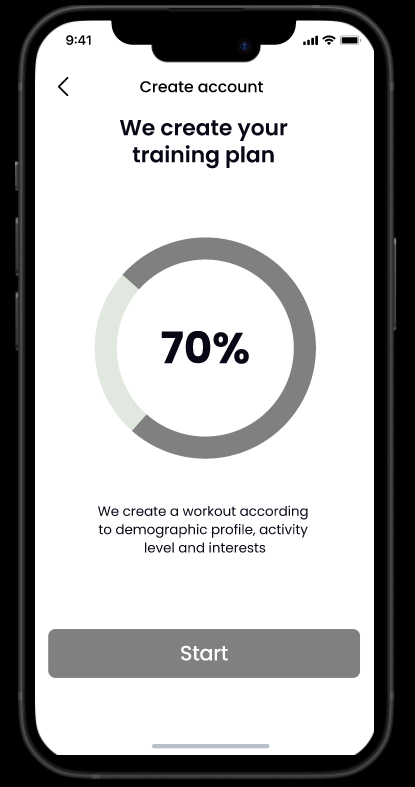
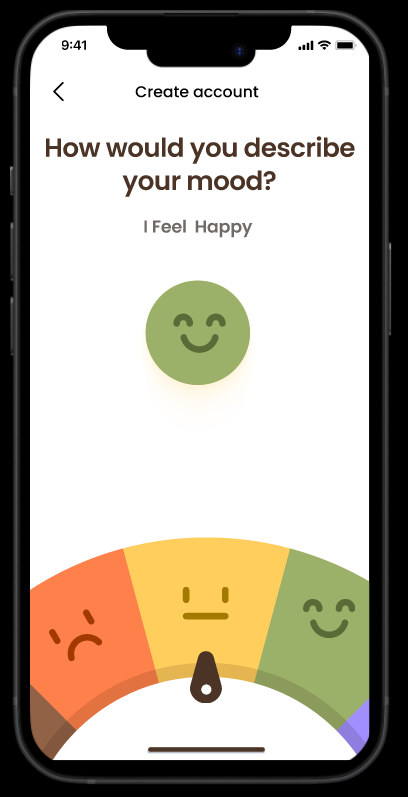
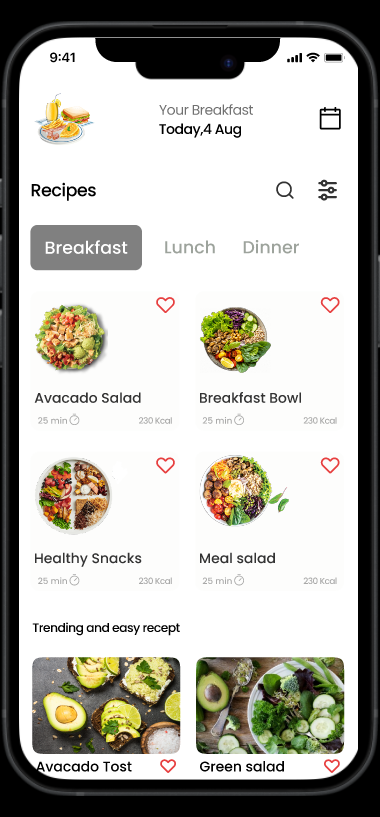


Erd Model



UI Wireframes/Mockups.



Activity Diagram:

A diagram with text and words

Description automatically generated with medium confidence

State Machine Diagram:

A screenshot of a diagram

Description automatically generated

Glossary:

**User**: An individual who interacts with the Bite Bright system to track meals, manage dietary habits, and receive personalized recommendations.

**Meal Details**: Information logged by the user, including food items, portion sizes, and meal types (e.g., breakfast, lunch).

**Nutrients**: Components of food intake, tracked by the system, such as calories, proteins, fats, and carbohydrates.

**Wearable Device**: External hardware (e.g., Fitbit, Apple Watch) integrated with the system to provide health and activity data.

**Health Data**: Information synchronized from wearable devices, including metrics like heart rate and activity levels.

**Meal Categorization**: The process of classifying logged meals as "healthy" or "unhealthy" based on predefined rules.

**Personalized Recommendations**: Diet suggestions generated by the system using user data, goals, and emotional status.

**Emotional Health Data**: Data logged by users regarding their emotional well-being, such as mood and stress levels, used to provide insights.

**80/20 Principle**: A dietary guideline applied by the system, encouraging 80% adherence to healthy eating and 20% flexibility.

**Insights**: Analytical results generated by the system, correlating trends in eating patterns, health data, and emotional well-being.

**Sync**: The process of transferring data from wearable devices to the Bite Bright system.

**Validation**: System checks to ensure user-inputted data or synchronized data is complete and correctly formatted.

**Reports**: Summarized outputs generated by the system, providing information on user compliance, progress, and areas for improvement.

**Security and Access Control**: Mechanisms to ensure data protection and enforce role-based permissions for different user types.

**Integration Subsystem**: A module in the system responsible for connecting with external APIs and wearable devices.

**Supplementary Specs:**

**Introduction**: The system, Bite Bright, provides a modern diet management solution, emphasizing emotional well-being alongside traditional dietary tracking. It promotes a sustainable, balanced approach to eating habits​.

**Common Functionality**: Bite Bright tracks meals, nutrients, and eating habits effortlessly. It also provides personalized, sustainable dietary recommendations based on user input and progress​.

**Logging**: Users log meals and health data, including food details, nutrient information, and emotional health data. Logs are validated for completeness and correctness before being stored​​.

**Error Handling**: When input data is invalid or synchronization fails, the system notifies the user and provides corrective actions. It retries failed operations such as syncing after a predefined interval​​.

**Security**: The system ensures secure authentication and role-based access control to protect user data. It enforces access restrictions between admin and regular users​.

**Usability**: The platform offers mobile and web applications with intuitive user interfaces for meal logging, health tracking, and viewing reports​.

**Reliability**: The system includes retry mechanisms for tasks like syncing health data and storing inputs to ensure reliable performance​.

**Recoverability**: Recovery is handled by ensuring that user and system data are stored securely on a cloud-based backend​.

**Performance**: The cloud backend processes data efficiently and dynamically generates recommendations based on user activity. Performance is further supported by integration with external APIs​​.

**Supportability**: The system’s modular architecture, including separate subsystems for user management, meal tracking, analytics, and security, enhances maintainability and scalability​.

**Adaptability**: Recommendations are dynamically updated based on user progress, preferences, and feedback​​.

**Interfaces**: The system integrates with wearable devices (e.g., Fitbit, Apple Watch) to sync health data. It also communicates with third-party fitness apps via APIs​.

**Reports**: Users can access reports on compliance rates, progress toward goals, and overall well-being​.

**Legal Issues**: The system must adhere to data protection laws, such as GDPR, to ensure user privacy and compliance​.

**Pricing and Payment Handling**: These details are not explicitly provided in the document. You may need to supplement this based on system-specific requirements.

**Operating Systems**: The system supports mobile platforms using Flutter and web platforms with React.js​.

**Networking Systems**: Networking is managed through secure API communication between wearable devices, third-party apps, and the cloud backend​.

**Internationalization**: While not explicitly mentioned, the use of modular design and dynamic recommendations suggests the potential for adaptation to various user demographics​.