Food Management System

Final Year Project

Session 2023-2027

A project submitted in partial fulfillment of the degree of

BS in Computer Science

Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior University, Lahore

Fall 2024

Type (Nature of project)	[..] Development [ ] Research [ ] R&D
Area of specialization on	
FYP ID	

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	S23-079	Yusra Noor	SU92-BSCSM-S23-079@ gmail.com	
(ii)	S23-132	Hafsa Irshad	SU92-BSCSM-S23-132@ gmail.com	
(iii)	S23-127	Maheen Rehman	SU92-BSCSM-S23-127@ gmail.com	
(iv)	S24-011	Saira Arshad	SU92-BSCSM-S24-011@ gmail.com	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I _____ S/D of Muhammad Siddique, group leader of FYP under registration no _____at Software Engineering Department, The Superior College, Lahore. I declare that my FYP report is checked by my supervisor.

Date: Name of Group Leader: _____ Signature: _____

Name of Supervisor: Mr. ABC Co-Supervisor: Dr. XYZ

Designation: Lecturer Designation: Associate Professor

Signature: _____ Signature: _____

HoD: Dr. Muhammad Azam

Signature: _____

Project Report

[Title of Project]

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
	1.0		<Original Draft>	
			<Changes Based on Feedback from Supervisor>	
			<Changes Based on Feedback From Faculty>	
			<Added Project Plan>	
			<Changes Based on Feedback from Supervisor>	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

This work is dedicated to my beloved family and friends who have always supported me throughout my academic journey, and to all individuals striving to lead healthier lives through technology-driven solutions.

Acknowledgements

I am really thankful to my Madam who has consistently guided me through each phase of this project with patience and expertise. Her valuable insights and constructive feedback helped
shape the project into a practical and innovative solution. I also extend my gratitude to my team members for their dedication, as well as to my university for providing the necessary resources and platform to bring this idea to life.

Executive Summary

The Food Management System (Calorie Detection System) is an intelligent application developed to simplify and enhance the way users manage their daily food intake. Leveraging image processing and machine learning, the system allows users to capture photos of their meals, identify food items, and receive accurate calorie and nutrient estimations. This solution eliminates the need for manual logging and addresses the growing demand for automated and reliable dietary monitoring tools.

The system is designed to cater to a diverse user base including caregivers of infants, Health-conscious teenagers, busy adults, elderly individuals, and patients with dietary restrictions such as diabetes. By integrating a trained Convolutional Neural Network (CNN) for food detection and connecting it with trusted nutritional databases like USDA Food-Data Central, the system ensures high accuracy and real-time feedback. It also supports multilingual functionality and accessibility features to promote inclusive usage.

This project involved phases such as requirement analysis, UI/UX design, model training, backend API development, frontend implementation, testing, and final deployment. Built using modern technologies like Flutter, Python, TensorFlow, and Firebase, the system is cross-platform compatible and designed for scalability.

The result is an efficient and user-friendly application that empowers individuals to take control of their nutrition in a smarter and simpler way. The Food Management System serves not just as a project but as a step toward healthier living through innovative technology.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1: Introduction

This chapter introduces the concept, motivation, and need for developing a **Food Management System with Calorie Detection**. The project is designed to help users manage their food intake efficiently by identifying the calorie count in food items through either image recognition or manual input. It emphasizes the importance of health awareness and how technology can support individuals in leading a healthier lifestyle.

1.1. Background

In the modern era, where fast food and unhealthy eating habits have become a norm, many individuals struggle to monitor their calorie consumption. While awareness about diet and fitness is growing, most people do not have the time or knowledge to track their meals manually. Manual tracking can be tedious and inaccurate, especially for those unfamiliar with nutritional values. To address this problem, a smart and accessible solution is needed—one that automates calorie detection and provides users with insights into their food choices. This project aims to create such a system using a combination of image processing, a food database, and interactive user interfaces.

1.2. Motivations and Challenges

The motivation for this system arises from the increasing concern over poor eating habits and their link to health issues such as obesity, diabetes, and heart problems. Many mobile applications for calorie tracking exist, but they often rely solely on manual entry, which is inconvenient and prone to user error.

Key challenges include:

- Ensuring accurate calorie detection through image recognition.
- Building a comprehensive and localized food database.
- Creating a user-friendly and responsive interface suitable for all age groups.
- Maintaining fast system performance despite image processing complexity.

1.3. Goals and Objectives

The main goal of this project is to assist users in monitoring their calorie intake through a smart and simple application. The specific objectives are:

- To develop a food management application that allows both image-based and manual food logging.
- To integrate a backend food database with calorie values.
- To track and display daily, weekly, and monthly intake summaries.
- To provide real-time alerts and healthy food suggestions.
- To ensure the system is accessible via both web and mobile platforms.

1.4. Literature Review/Existing Solutions

Several applications currently exist for tracking calories, such as:

- MyFitnessPal
- HealthifyMe
- FatSecret
- CalorieMama

While these apps are useful, they primarily rely on manual input. Some of them offer barcode scanning or image detection, but often lack accuracy or support for local foods. Our project aims to fill these gaps by offering localized food detection with high accuracy and simplicity.

1.5. Gap Analysis

Feature	Existing Apps	Our Proposed System
Calorie Detection	Manual/Barcode-based	Image + Manual options
Support for Local/Regional Foods	Limited or None	Included
Personalized Suggestions	Basic	Based on history and goals
Reminders and Alerts	Not consistent	Real-time alert system
Reporting and Visualization	Basic charts	Advanced summaries and insights

1.6. Proposed Solution

We propose a **smart web and mobile system** that allows users to:

- Upload or capture a food image to detect calories.
- Search and log food items manually if needed.
- View summaries of their calorie intake over time.
- Receive health tips and recommendations.
- Be notified when exceeding daily calorie limits.

This solution will combine machine learning (for image detection) with a structured nutritional database and a user-friendly interface. It will be developed using modern web technologies and optionally hosted on cloud platforms for availability.

1.7. Project Plan

The project will follow the **Incremental Process Model** as described in Sommerville's Software Engineering (Chapter 2). The system will be developed in stages:

1. Planning and Requirement Gathering
2. UI/UX Design
3. Backend and Database Setup
4. Calorie Detection Module
5. Integration and Testing
6. Final Reporting and Deployment

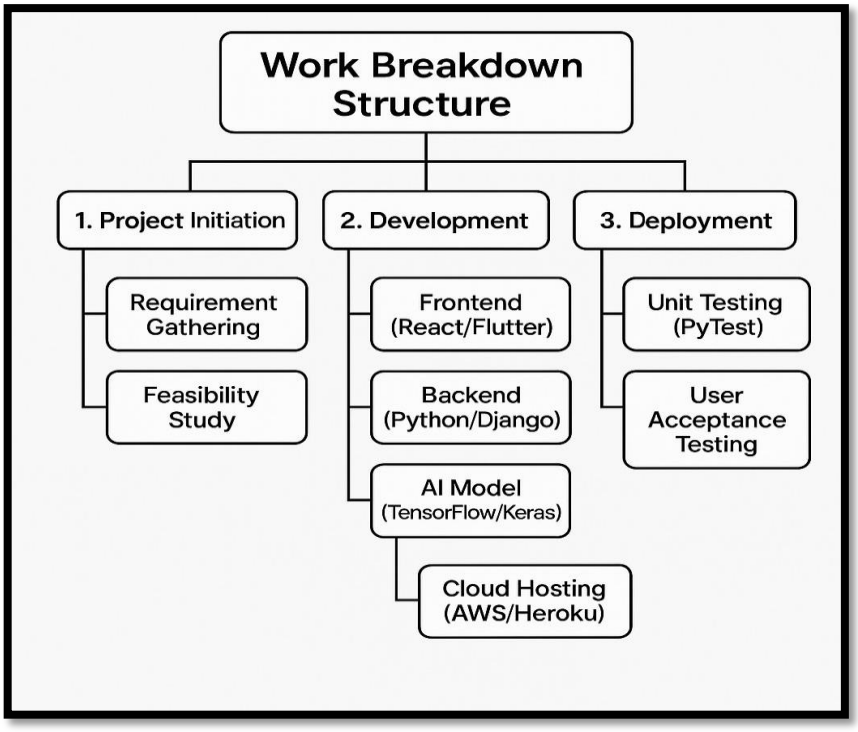
1.7.1. Work Breakdown Structure

1. Project Initiation
- Requirement Gathering
- Feasibility Study

2. Development
- Frontend (React/Flutter)
- Backend (Python/Django)
- AI Model (TensorFlow/Keras)

3. Testing
- Unit Testing (PyTest)
- User Acceptance Testing

4. Deployment
- Cloud Hosting (AWS/Heroku)

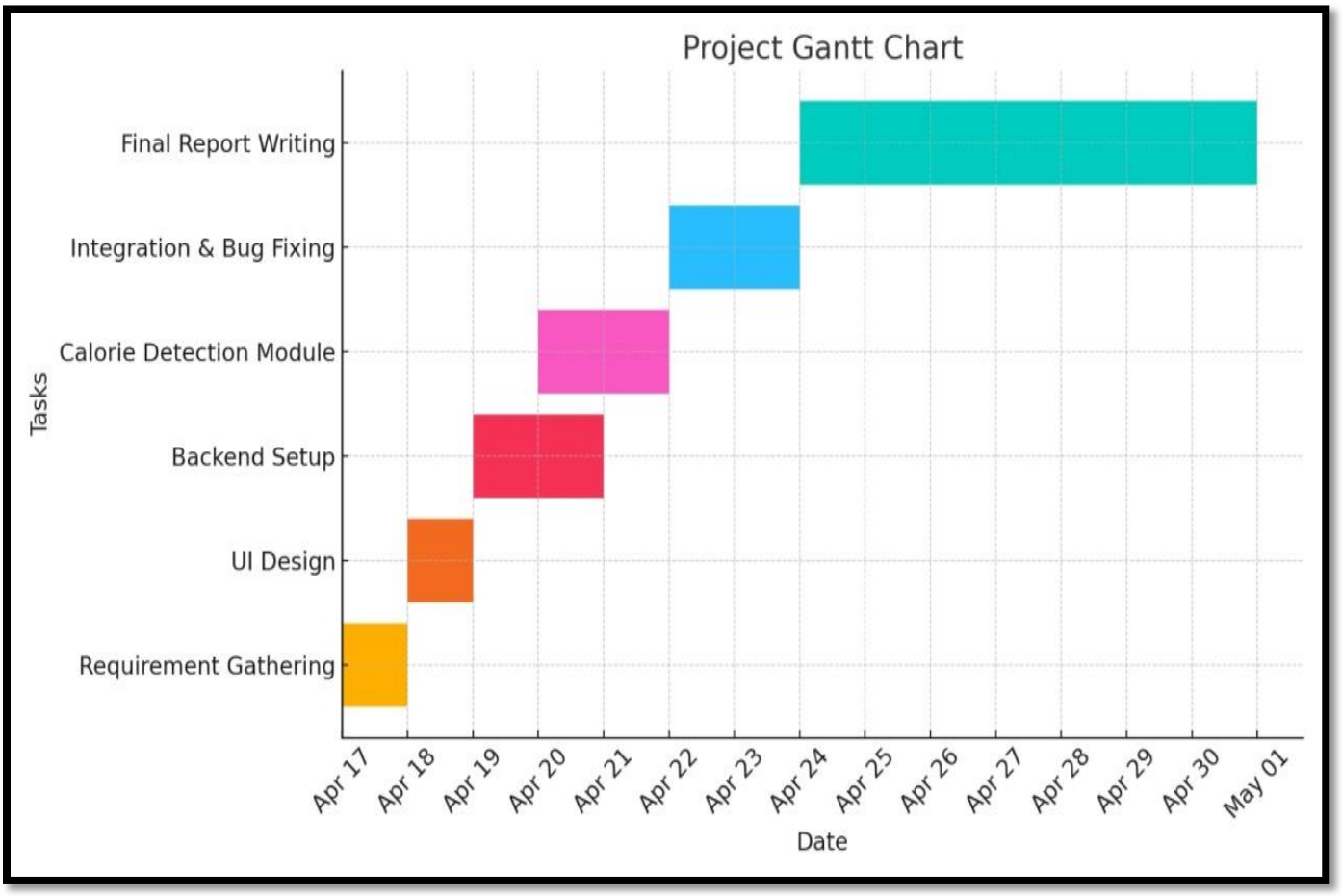


1.7.2. Roles & Responsibility Matrix

Team Member	**Role**	**Responsibilities**
Yusra Noor	Team Leader	Coordination, Integration, Planning
Maheen Rehman	Front-End Developer	UI/UX Design, Interface Implementation
Hafsa Irshad	Back-End Developer	API Development, Database Integration
Saira Ashraf	ML Developer	Calorie Detection Model, Testing

1.7.3. Gantt Chart(Timeline: April 17 – April 30, 2025)

Task	Start Date	End Date	Assigned Member
Requirement Gathering	Apr 17, 2025	Apr 17, 2025	Yusra Noor
UI Design	Apr 18, 2025	Apr 18, 2025	Maheen Rehman
Backend Setup	Apr 19, 2025	Apr 20, 2025	Hafsa Irshad
Calorie Detection Module	Apr 20, 2025	Apr 21, 2025	Saira Ashraf
Integration & Bug Fixing	Apr 22, 2025	Apr 23, 2025	Yusra Noor
Final Report Writing	Apr 24, 2025	Apr 30, 2025	Saira Ashraf



Chapter 2

Software Requirement Specifications

Chapter 2: Software Requirement Specifications

2.1. Introduction

2.1.1. Purpose

This SRS defines the software requirements for the Food Management System (Calorie Detection), version 1.0. It is a new standalone system designed to help users track their food intake and calories using image recognition or manual logging. This document is intended to be the definitive source of all software functionality and constraints.

2.1.2. Document Conventions

- Headings follow section numbers (e.g., 5.1.1)
- Requirements are listed with identifiers (e.g., REQ-SF1-1)
- Bold text is used for important terms and system features
- Monospace font is used for identifiers and input/output terms

2.1.3. Intended Audience and Reading Suggestions

This document is intended for:

- Developers: to understand system functionality and implementation needs
- Testers: to develop and execute test cases
- Project Managers: to monitor scope and ensure delivery
- Users and Stakeholders: to confirm feature requirements

Start by reviewing the Introduction, then refer to Functional and Non-Functional requirements. Developers and testers should focus on Section 5.4 (System Features) and Section 5.5 (Quality Attributes).

2.1.4. Product Scope

The system will allow users to:

- Upload food images for calorie detection
- Enter food details manually
- Track calorie intake history

- View nutritional summaries and recommendations

It supports health-focused users who want a user-friendly, intelligent assistant for managing food habits. This system aligns with digital healthcare goals.

2.1.5. References

- Sommerville, Ian. Software Engineering (10th Edition)
- MyFitnessPal – https://www.myfitnesspal.com/
- HealthifyMe – https://www.healthifyme.com/
- IEEE SRS Template Guidelines – IEEE Std 830-1998

2.2. Overall Description

2.2.1. Product Perspective

The system is a new, independent web/mobile application. It integrates an AI model or API to detect food items and calories. It contains UI, server, database, and optional ML backend components.

2.2.2. Product Functions

- User login and registration
- Upload food images for calorie detection
- Manual food entry
- View calorie reports
- Admin panel for food database management

2.2.3. User Classes and Characteristics

User Class	Characteristics
End User	Tech-savvy or non-technical health-conscious individuals
Admin	Nutrition expert managing food entries

2.2.4. Operating Environment

- Devices: Mobile (Android), Web (Desktop)
- Backend: Django/Python
- ML Model: TensorFlow/Keras

- Database: MySQL or Firebase
- Deployment: AWS or Heroku

2.2.5. Design and Implementation Constraints

- Must use Python/Django backend
- AI model should be image-based or API-driven
- Web responsive design (React preferred)
- Mobile app with Flutter (optional)

2.2.6. User Documentation

- User Manual (PDF)
- Online Help (FAQs)
- Onboarding Tutorial within app

2.2.7. Assumptions and Dependencies

- Users have internet connectivity
- Food image database/API is reliable
- Food database will be maintained manually by admin

2.3. External Interface Requirements

2.3.1. User Interfaces

- Clean UI with login, upload, and report pages
- Use of standard buttons like Submit, Back, Cancel
- Real-time alerts and confirmations

2.3.2. Hardware Interfaces

- Camera access for image uploads (Mobile only)
- Compatible with Android 9.0+ and modern web browsers

2.3.3. Software Interfaces

- AI engine (TensorFlow or external API)
- Database (MySQL/Firebase)
- RESTful API endpoints for data handling

2.3.4. Communications Interfaces

- HTTP/HTTPS protocols
- JSON data exchange format
- Secure image upload using multipart encoding

2.4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

2.4.1. System Feature 1

Upload Food Image

2.4.1.1. Description and Priority

Allow users to upload images for calorie estimation. Priority: High

2.4.1.2. Stimulus/Response Sequences

User uploads image → System detects food → Displays calories

2.4.1.3. Functional Requirements

- REQ-SF1-1: The system shall allow image upload (JPG/PNG)
- REQ-SF1-2: The system shall return estimated calorie value
- REQ-SF1-3: The system shall save the food log into the database

2.4.2. System Feature 2

Manual Food Entry

2.4.2.1. Description and Priority

Allow users to enter food name and quantity manually. Priority: High

2.4.2.2. Stimulus/Response Sequences

User selects item from list → Enters quantity → Views calories

2.4.2.3. Functional Requirements

- REQ-SF2-1: User can select food from a searchable list
- REQ-SF2-2: System calculates calories based on quantity
- REQ-SF2-3: System logs the entry for reporting

2.4.3. System Feature 3

View Reports

2.4.3.1 Description and Priority

Show weekly/monthly calorie reports. Priority: Medium

2.4.3.2 Stimulus/Response Sequences

User selects date range → System fetches logs → Shows chart

2.4.3.3 Functional Requirements

- REQ-SF3-1: System shall support date filtering
- REQ-SF3-2: Reports shall include calories and items summary
- REQ-SF3-3: Reports shall be exportable (PDF/CSV)

2.5. Other Nonfunctional Requirements

2.5.1. Performance Requirements

- The system should return image results in under 5 seconds
- UI should load in under 2 seconds on 4G

2.5.2. Safety Requirements

- If calorie data is incomplete or uncertain, users should be alerted
- No food will be saved without user confirmation

2.5.3. Security Requirements

- All passwords must be hashed
- Data must be encrypted in transit (HTTPS)
- Authentication is required to log or view meals

2.5.4. Software Quality Attributes

- Usability: Easy-to-use interfaces for all user types
- Availability: 95% uptime
- Reliability: System should not crash during image upload or report generation
- Scalability: Must support multiple concurrent users

2.5.5. Business Rules

- Only registered users can access meal logging
- Admin can modify food database
- System will not allow duplicate entries per day for the same food item

2.6. Other Requirements

- Database must support future expansion
- Must support multiple units (grams, ml, pieces)
- Internationalization support (future scope)
- Compliance with basic data privacy guidelines (GDPR if required)

Chapter 3

Use Case Analysis

Chapter 3: System Analysis

This chapter provides a detailed analysis of how users interact with the Food Management System (Calorie Detection). It introduces the system's core actors, their roles, and the specific use cases they engage in. By outlining the use case model and describing primary interactions, this chapter lays the foundation for design and implementation.

3.1 Use Case Model

The system has two primary actors: User and Admin.

Actors:

User: Interacts with the system to upload food images, log meals, and track calories.

Admin: Manages the food database and monitors user logs.

Main Use Cases:

Actor	Use Case	Description
User	Register/Login	Authenticate into the system
User	Upload Food Image	Upload image for calorie detection
User	Enter Food Manually	Add a food item manually
User	View Calorie Result	See estimated calories from uploaded food
User	Track Meal History	View list of logged food entries
User	View Calorie Reports	Weekly/monthly summaries and charts
Admin	Add/Edit/Delete Food Items	Manage nutritional database
Admin	Manage Users	Monitor and manage registered users

3.2 Use Case Descriptions

□ Use Case: Upload Food Image

Actor: User

Precondition: User must be logged in

Description: The user uploads a food image. The system detects the food using an ML model/API and returns an estimated calorie count.

Postcondition: The food is logged with estimated calories.

Exceptions: If the image is unclear or unidentifiable, the system prompts for manual input.

Use Case: View Calorie Report

Actor: User

Precondition: At least one food item must be logged

Description: The user selects a time frame (daily/weekly/monthly) to view total calorie consumption with visual summaries.

Postcondition: A report is displayed on the screen.

Exceptions: No data for selected period results in a message: "No records available."

Use Case: Manage Food Items

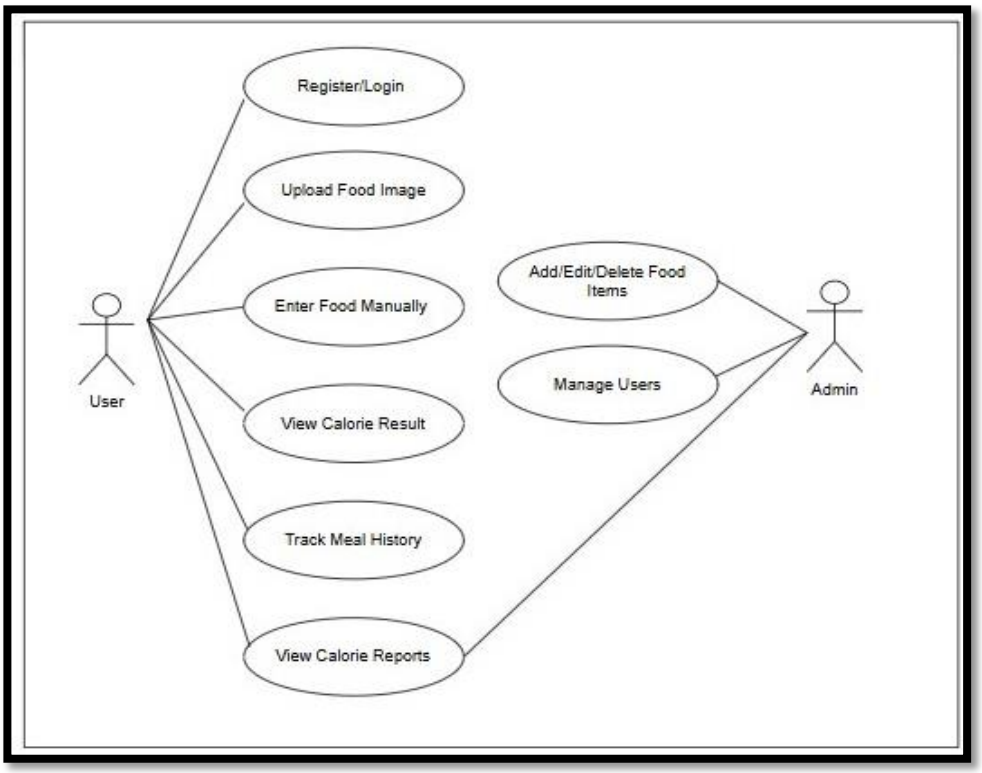
Actor: Admin

Precondition: Admin must be authenticated

Description: Admin can add new food items, update existing values, or remove outdated entries.

Postcondition: The food database is updated accordingly.

Exceptions: Database errors or invalid data formats trigger an error alert.



Chapter 4

System Design

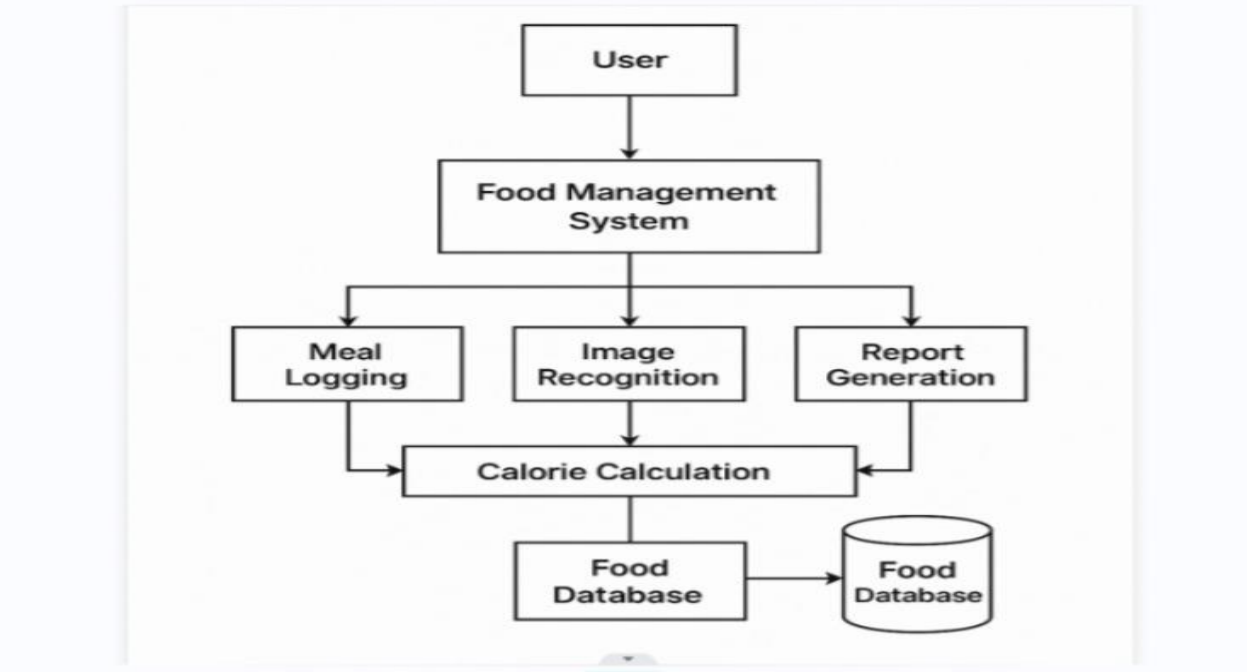
Chapter 4: System Design

This chapter presents the complete system design for the Food Management System (Calorie Detection). It includes the system architecture, domain model, ERD, class diagram, sequence diagram, operation contracts, and various UML diagrams that illustrate how the system functions. Each diagram and design component helps to understand the internal structure and data flow of the system. This structured approach ensures clarity and maintainability in implementation.

4.1. Architecture Diagram

The system uses a layered architecture that separates concerns into distinct layers:

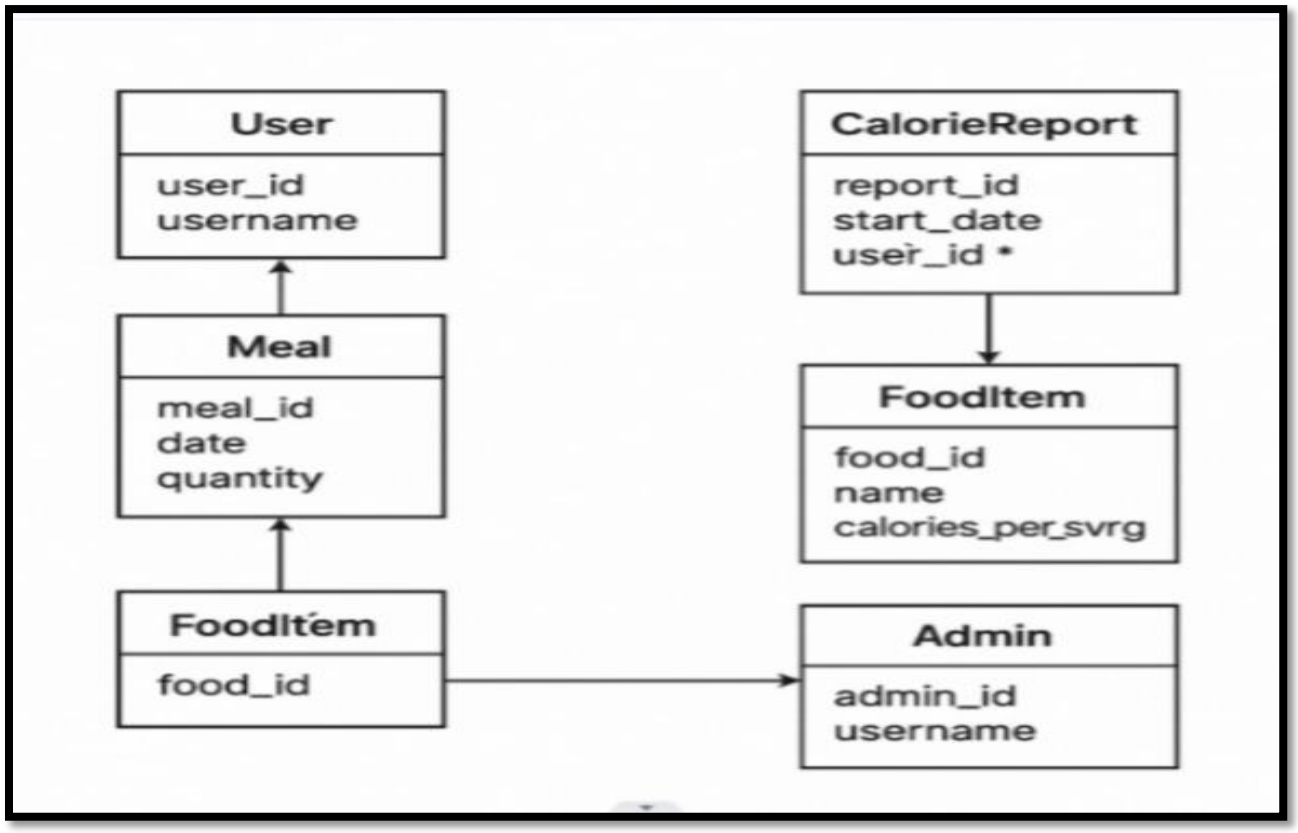
- Presentation Layer: Handles user interaction via web/mobile UI.
- Application Layer: Processes requests, connects UI with backend logic.
- AI/Model Layer: Calorie detection using image analysis (ML or API).
- Data Layer: Manages persistent data storage such as user logs and food entries.



4.2. Domain Model

The domain model defines the core concepts of the system:

- User: A person using the system to log and monitor meals.
- FoodItem: Represents an entry with name, calories, and nutritional details.
- LogEntry: Record of food consumed by the user.
- Admin: Manages food entries in the system.
- AI Engine: Used to process food image and return detected results.



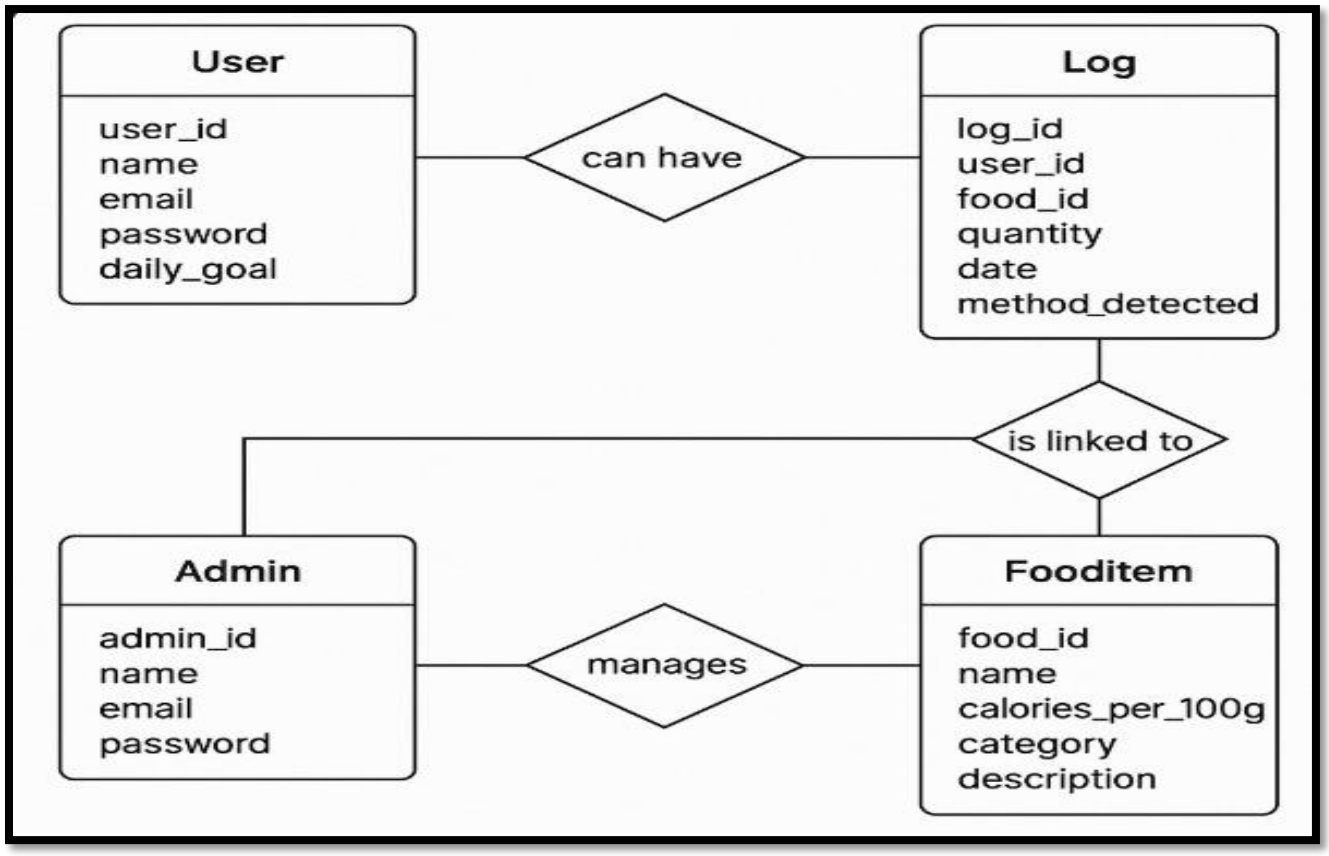
4.3. Entity Relationship Diagram with data dictionary

Entities and Attributes:

- User (user_id, name, email, password, daily_goal)
- FoodItem (food_id, name, calories_per_100g, category, description)
- Log (log_id, user_id, food_id, quantity, date, method_detected)
- Admin (admin_id, name, email, password)

Relationships:

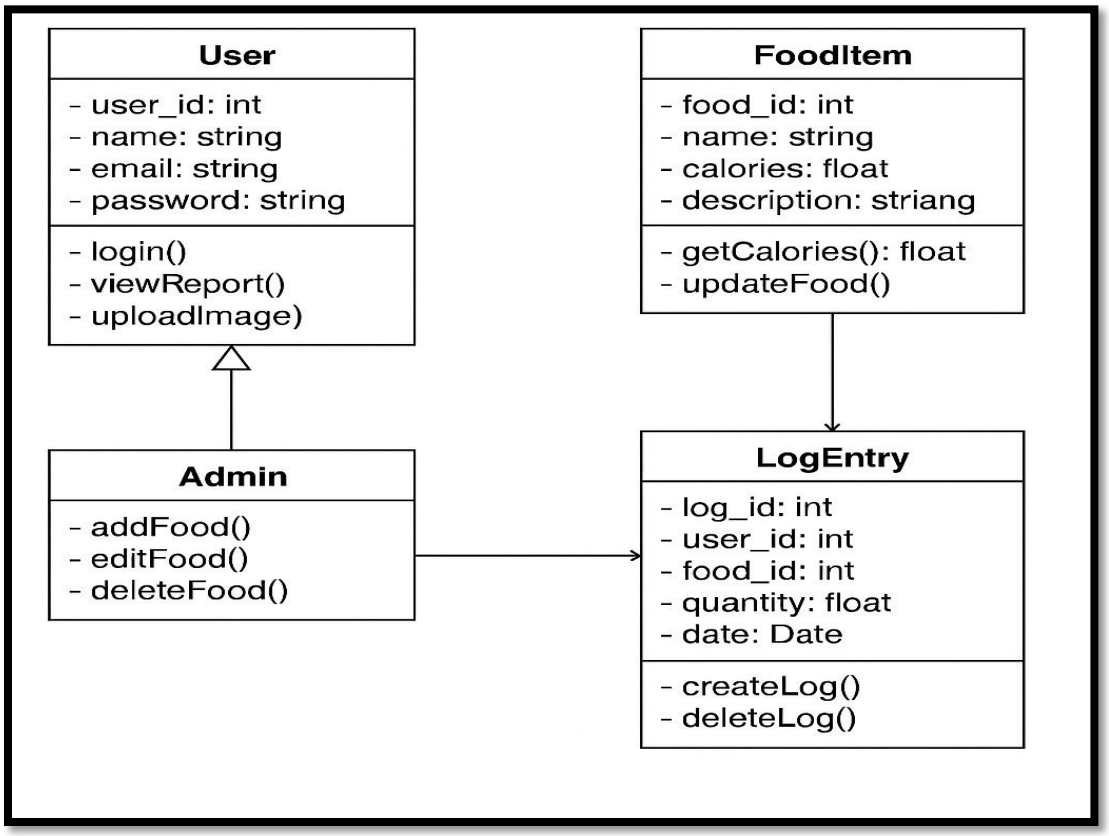
- One User can have many LogEntries
- Each LogEntry is linked to one FoodItem
- Admin manages many FoodItems



4.4. Class Diagram

Main Classes:

- User
  - Attributes: user_id, name, email, password
  - Methods: login(), viewReport(), uploadImage()
- FoodItem
  - Attributes: food_id, name, calories, description
  - Methods: getCalories(), updateFood()
- LogEntry
  - Attributes: log_id, user_id, food_id, quantity, date
  - Methods: createLog(), deleteLog()
- Admin
  - Methods: addFood(), editFood(), deleteFood()

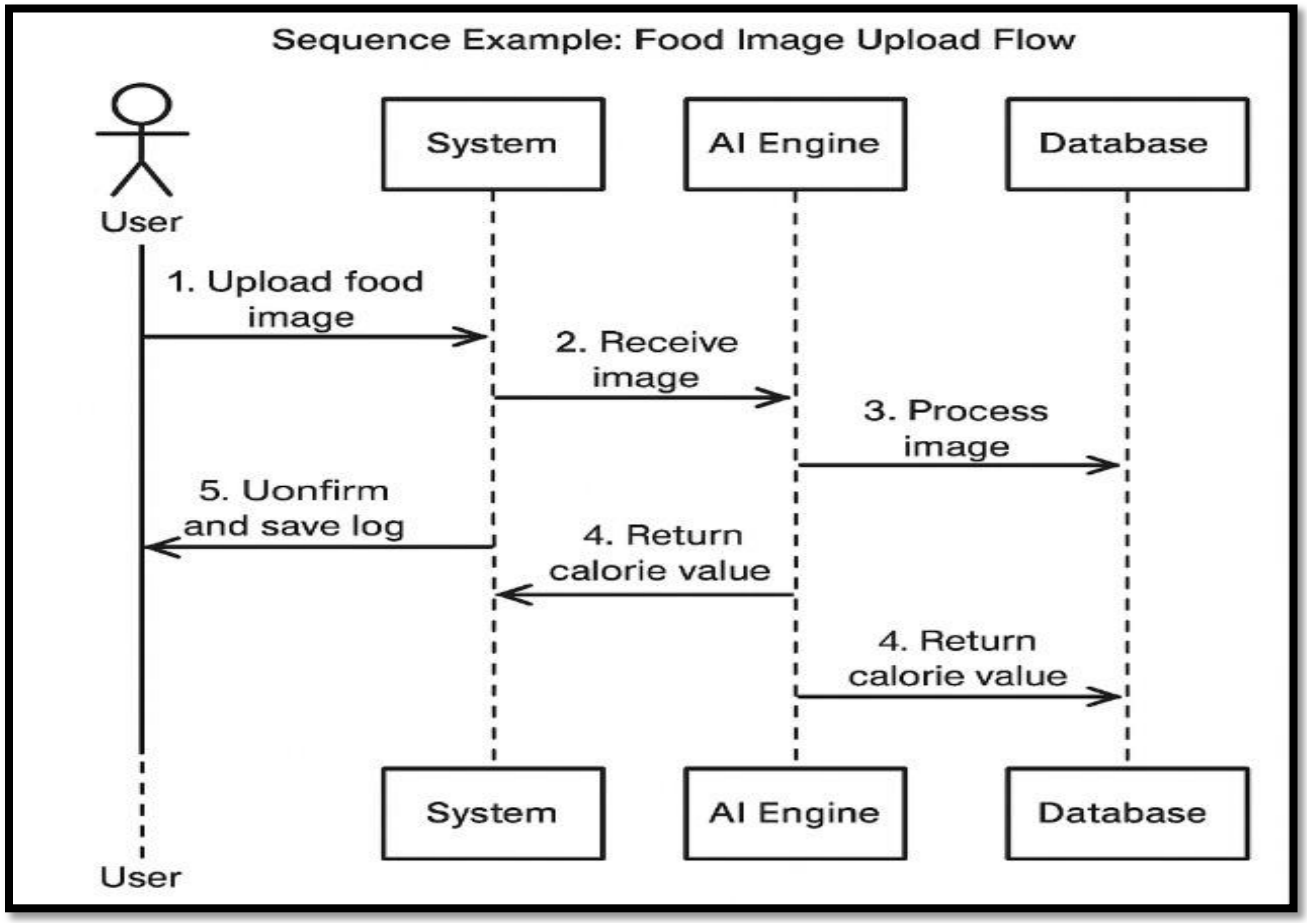


4.5. Sequence / Collaboration Diagram

Sequence Example: Food Image Upload Flow

1. User uploads food image
2. System receives image
3. AI Engine processes image
4. System returns calorie value
5. User confirms and saves log

Each step involves an interaction between User → System → AI Engine → Database.



4.6. Operation contracts

Operation: Upload Image and Detect Calories

- Precondition: User is logged in
- Postcondition: Calorie value is calculated and displayed
- Inputs: Image file, timestamp
- Outputs: Food name, estimated calories

Operation: View Weekly Report

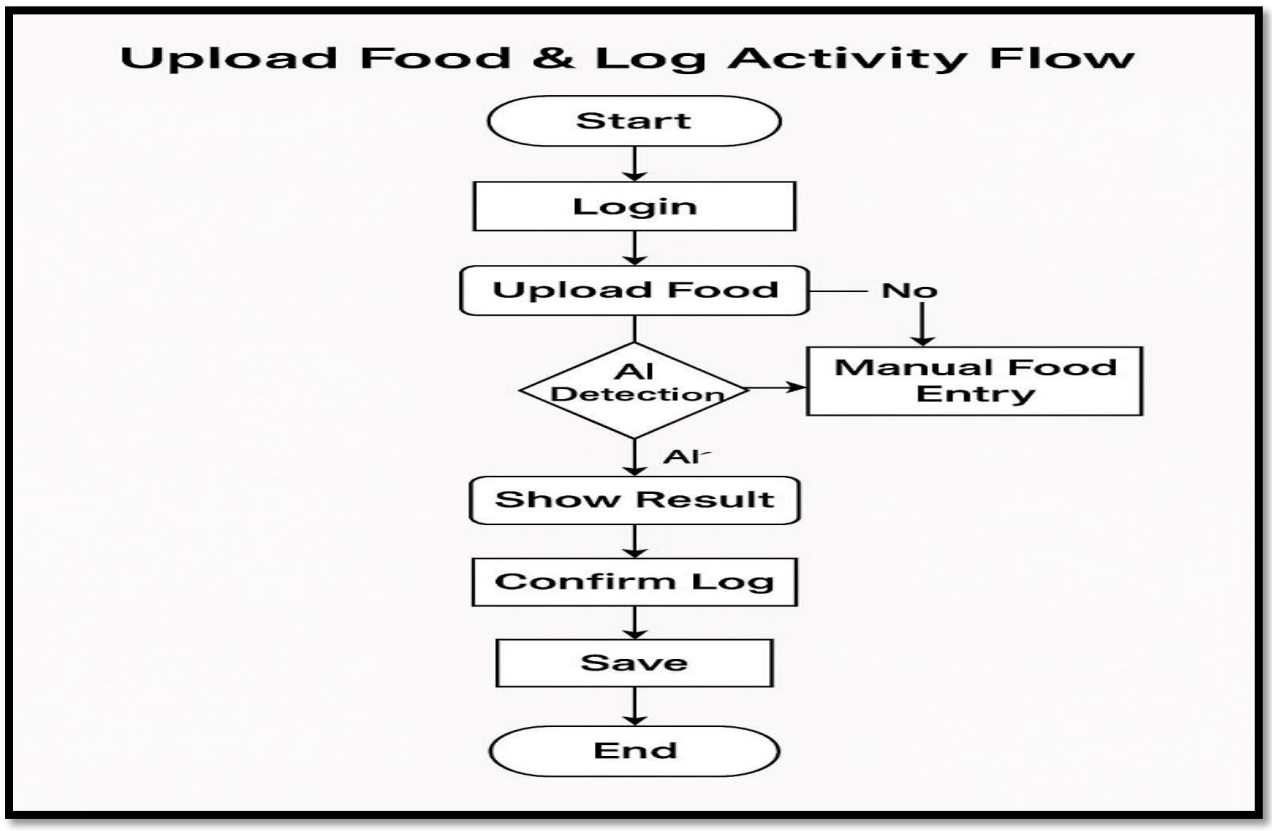
- Precondition: Food logs must exist
- Postcondition: Weekly summary shown with calorie totals

4.7. Activity Diagram

Upload Food & Log Activity Flow:

- Start
- Login → Upload food → AI detection → Show result → Confirm log → Save → End

Alternate path: Manual food entry instead of AI detection.

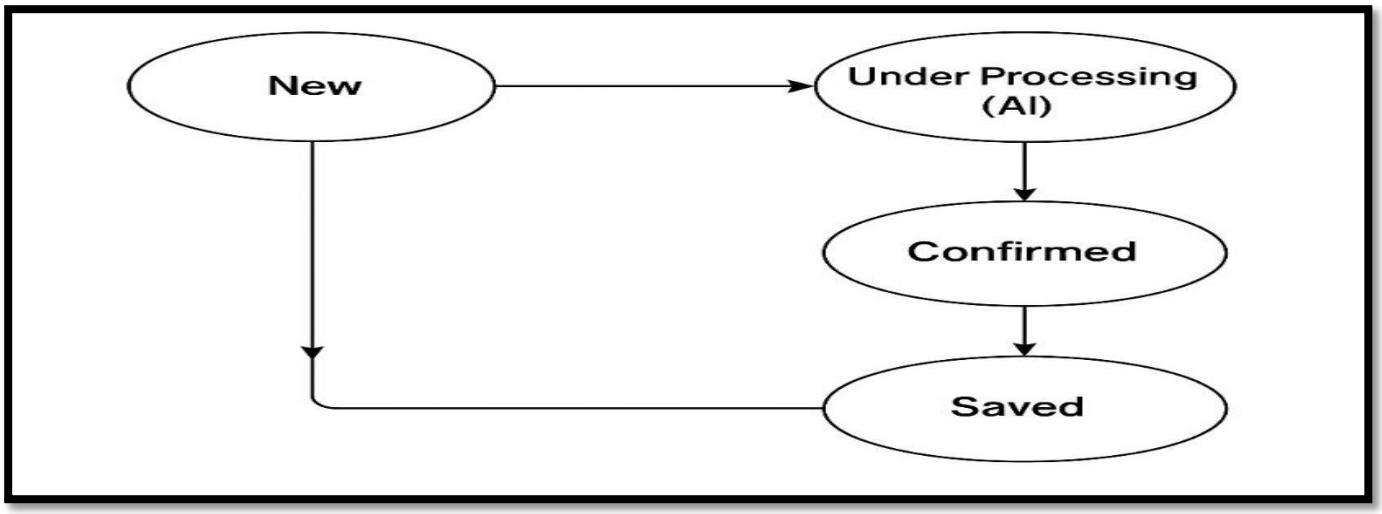


4.8. State Transition Diagram

States of a Log Entry:

- New → Under Processing (AI) → Confirmed → Saved
- Manual Entry skips "Processing" state.

Triggers include user actions like upload, confirm, or cancel.

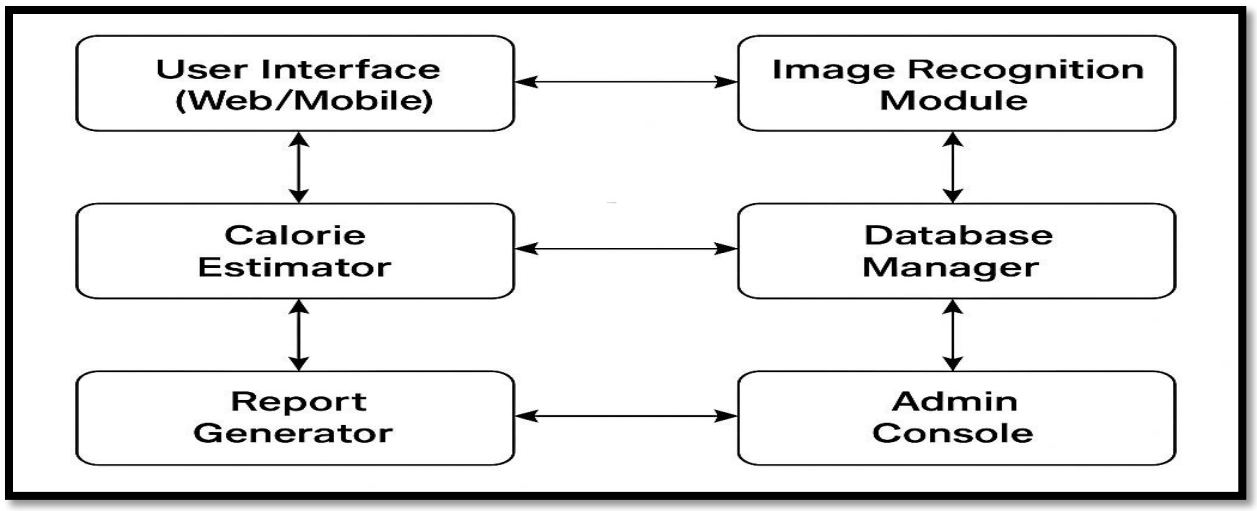


4.9. Component Diagram

Components:

- User Interface (Web/Mobile)
- Image Recognition Module
- Calorie Estimator
- Database Manager
- Report Generator
- Admin Console

Each component is loosely coupled and communicates via REST APIs or internal function calls.

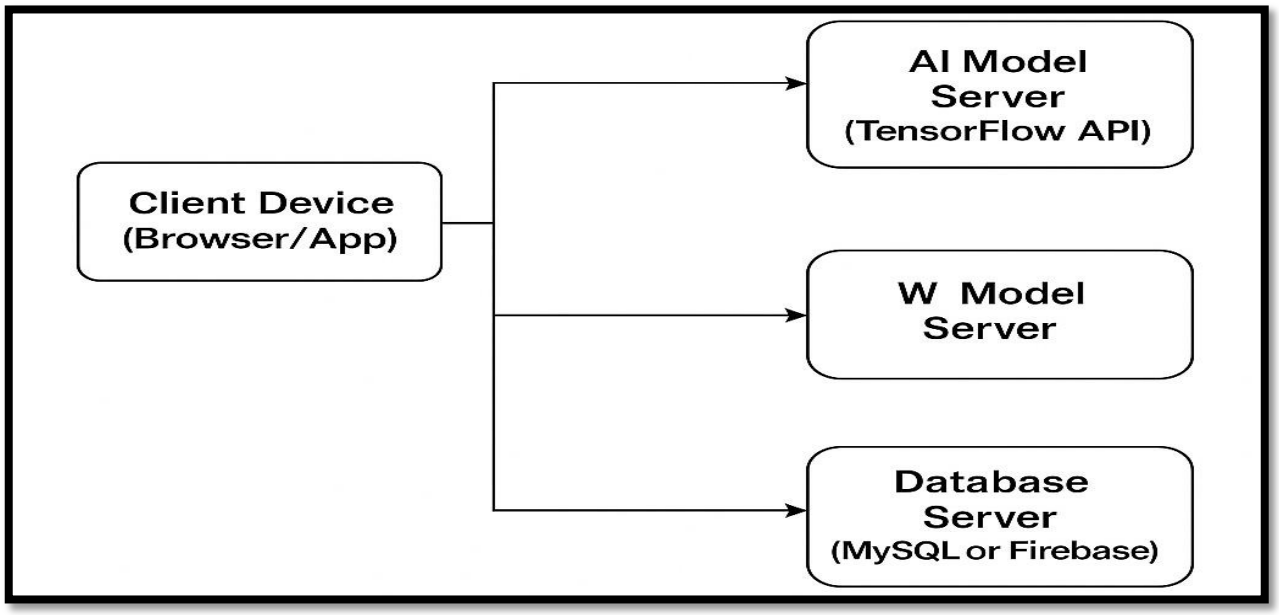


4.10. Deployment Diagram

Nodes:

- Client Device (Browser/App)
- Web Server (Django backend)
- AI Model Server (TensorFlow API)
- Database Server (MySQL or Firebase)

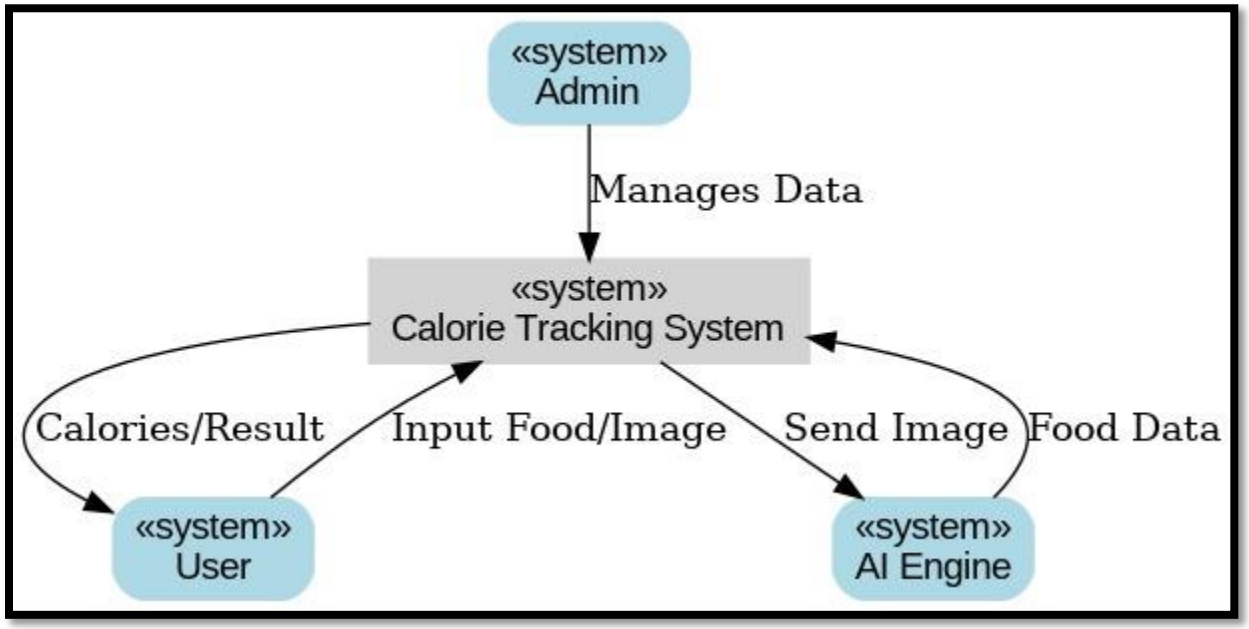
Deployment Environment: Cloud-based (AWS/Heroku)



4.11. Data Flow diagram

Level 0: Context Diagram

- User inputs → System → Returns calories/result
- Admin manages → System
- AI engine receives image → returns food data



Level 1: Process Breakdown

- 1.0 Login & Authentication
- 2.0 Food Logging (Image or Manual)
- 3.0 Calorie Detection & Analysis
- 4.0 Reporting & History
- 5.0 Admin Management