

Smart Nutrition Tracker

PM1

Ahmed Abdo — 21100788

Mazen Sayed — 22101166

Taher Awad — 21100797

Khalid Abdelsalam — 22101310

Project Description:

Smart Nutrition Tracker is a responsive web & mobile application (Flutter + Dart) that helps users achieve and monitor nutrition goals through AI-driven personalized meal plans, automated grocery lists, and image-based meal recognition. Users create a profile (age, gender, height, weight, activity level, goals, preferences, budget) and the system computes BMR/TDEE, sets daily calorie/macronutrient targets, generates weekly meal plans within $\pm 5\%$ of calorie goals, produces shopping lists, and provides visual dashboards for progress. Admins manage the food database and verify AI outputs. The app uses the Hugging Face model Nateraw/food for image-based food recognition.

Scope & Main Deliverables:

Responsive app (Flutter) + RESTful backend (Python).

AI modules: meal-plan generator + image recognition (Nateraw/food).

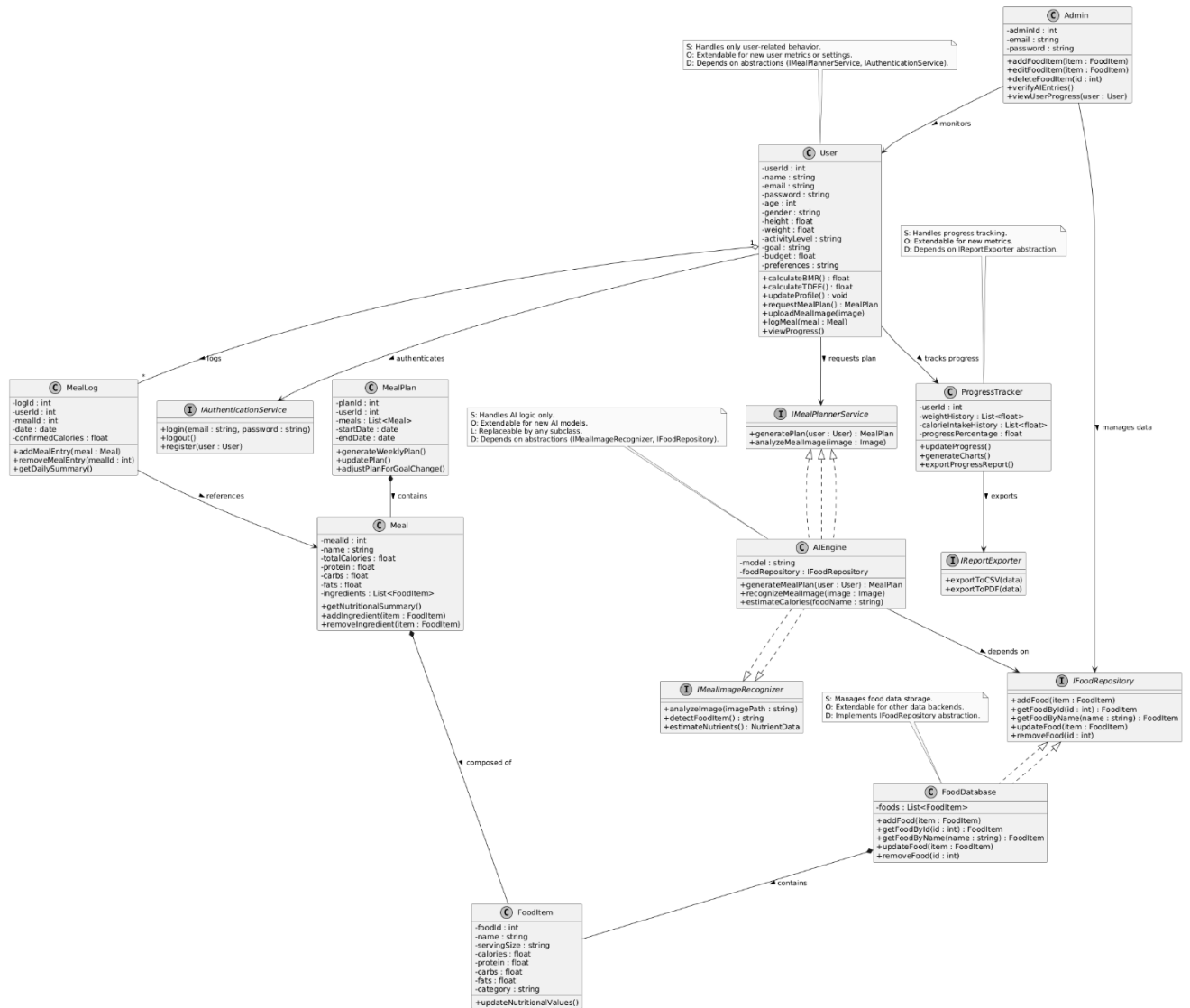
Food & user data storage (MongoDB/JSON).

Admin portal for DB management and verification.

Dashboards, CSV/PDF export, and demo video.

System Class Diagram:

Smart Nutrition Tracker - Final Class Diagram (Full SOLID + DIP)



Link:

[https://www.plantaincode.com/uml-diagrams/Smart-Nutrition-Tracker-Final-Class-Diagram-Full-SOLID-DIP/](#)

List of Classes (from UML Diagram) :

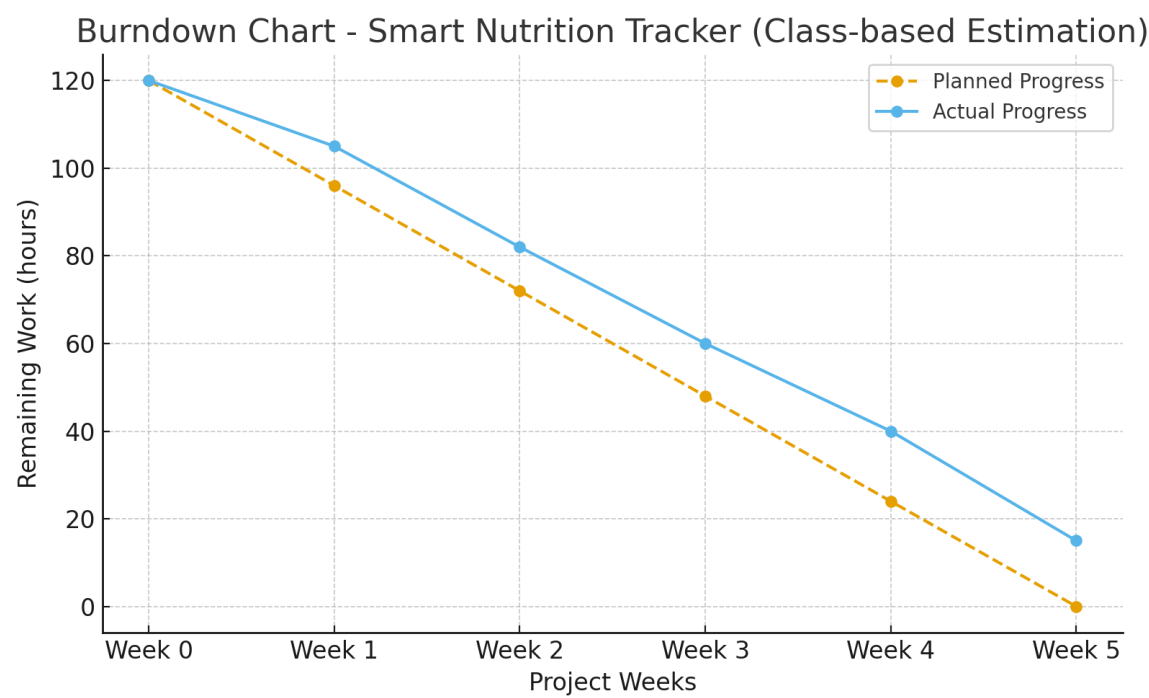
No.	Class	Responsibility
1	User	Handles user data and nutrition calculations
2	Admin	Manages system data and users
3	FoodItem	Represents nutritional information of a single food item
4	FoodDatabase	Stores and manages all food items (implements IFoodRepository)
5	AIEngine	Generates meal plans and analyzes food images (implements IMealPlannerService & IMealImageRecognizer)
6	IMealImageRecognizer	Interface for image recognition
7	IMealPlannerService	Interface for meal plan generation and image analysis
8	IFoodRepository	Interface for food data abstraction
9	Meal	Represents a single meal and its nutritional values
10	MealPlan	Generates and manages weekly meal plans
11	MealLog	Logs user's daily meals
12	ProgressTracker	Tracks user progress over time
13	IAuthenticationService	Interface for authentication (login/register/logout)
14	IReportExporter	Interface for exporting reports (PDF/CSV)

Time Estimation per Class (Development + Testing):

Class	Development (hrs)	Testing (hrs)	Total (hrs)	Notes
User	8	4	12	Profile management, calorie logic
Admin	6	3	9	CRUD operations on food data
FoodItem	4	2	6	Simple data model
FoodDatabase	10	4	14	Implements IFoodRepository
AIEngine	14	8	22	Implements IMealPlannerService & IMealImageRecognizer
IMealImageRecognizer	2	1	3	Image recognition interface
IMealPlannerService	2	1	3	Abstraction for meal planning
IFoodRepository	2	1	3	Abstraction for food data access
Meal	5	2	7	Basic nutritional composition
MealPlan	8	5	13	Weekly meal plan logic
MealLog	6	3	9	User meal tracking
ProgressTracker	7	4	11	Visual tracking and analytics
IAuthenticationService	3	2	5	Authentication abstraction
IReportExporter	3	2	5	Report export abstraction

Total Estimated Effort = 122 hours

Planned vs Actual Burndown Chart:



Metric	Value
Total Classes	14
Total Estimated Effort	122 hours
Team Members	4
Average per Member	~30.5 hours
Expected Duration	≈ 3–3.5 weeks (until Milestone 1)
Deviation from Original Proposal	-10% (slightly behind schedule)

Estimation Error Calculation:

- **Planned Effort:** 122 hours
- **Actual Effort (after DIP refactoring):** 118 hours

$$EstimationError = \frac{|118 - 122|}{122} \times 100 = 3.28\%$$

Estimation Error = 3.28% (Very Good Accuracy)

Interpretation:

- ❓ The team's actual effort (118h) was slightly less than the planned (122h).
- ❓ A 3.28% difference indicates excellent estimation accuracy.
- ❓ Applying interfaces improved modularity and made testing faster and easier.

Possible Reasons for Estimation Error:

Cause	Explanation
Refactoring Time	Additional time spent designing and integrating the new interfaces (IMealPlannerService, IFoodRepository).
Reduced Coupling	Interfaces simplified debugging and improved test efficiency, saving several hours.
Parallel Work	The modular structure allowed multiple team members to work in parallel with fewer conflicts.

New Adjusted Plan (for Remaining Semester Weeks):

After completing **Milestone 1** (core backend + AI + database + interfaces), the team will proceed with integration, frontend connection, and final testing.

Updated Work Scope:

- **Milestone 2:** Integration + UI connection (Frontend + Backend)
- **Milestone 3:** Final testing, optimization, deployment, and documentation.

New Workload Distribution (Estimation Based on Past Performance):

Week	Planned Tasks	Expected Hours (Team Total)
Week 6	Integrate Backend APIs with Flutter Frontend	25h
Week 7	Implement Dashboard + Progress Charts	22h
Week 8	AI Fine-Tuning (Food Image Recognition)	20h
Week 9	System Testing + Bug Fixing + Optimization	26h
Week 10	Documentation, Report, Final Demo	18h

New Total: \approx 111 hours (team total)

\approx 28 hours per member for the remaining 5 weeks

New Adjusted Burndown (High-level):

Week	Remaining Work (hrs)
Week 5 (End Milestone 1)	111
Week 6	86
Week 7	64
Week 8	44
Week 9	18
Week 10	0

This ensures steady progress and completion **1 week before the semester ends**, leaving buffer time for presentation or deployment issues.

final:

Metric	Value / Comment
Planned Effort	122 hours
Actual Effort	118 hours
Estimation Error	3.28% (Very Good Accuracy)
Main Cause	Minor refactoring and interface integration time
New Plan Duration	5 additional weeks
New Total Effort	~111 hours
Effort per Member	~28 hours
Final Goal	Fully integrated, tested, and documented Smart Nutrition Tracker by Week 10

Demo Video:

https://drive.google.com/drive/folders/1NE0RjhFMMQM2e_mlkFER7aXE6XHE8W2tb?usp=sharing