A logo for a college

AI-generated content may be incorrect.A logo for a university

AI-generated content may be incorrect.

|  |  |  |  |
| --- | --- | --- | --- |
| **Academic Year** | **Module** | **Assessment Type** | **Type** |
| 2025 | Project and Professionalism | A2 | Artefact Design and Test Plan |

Student ID: 240707

Student Name: Luja Ratna Manandhar

Group: L5CG5

Supervisor: Johan Tandukar

Submitted On: 2025-12-26

Introduction:

NutriLift is a mobile application designed to help people maintain a healthier lifestyle by combining fitness tracking, nutrition logging, gamification and a small online community in a single platform. This report presents the design artefacts produced during the current phase of development, including sprint planning, product backlog, data models and UML diagrams for each subsystem of the application. Together, these artefacts show how the initial proposal has been translated into a structured, testable system that is ready to move into implementation and formal evaluation.

Table of Contents

[1. FDD Diagram: 1](#_Toc217674002)

[2. Planning: 1](#_Toc217674003)

[3. Wire Frame: 2](#_Toc217674004)

[4. User Management: 5](#_Toc217674005)

[4.1. SRS: 5](#_Toc217674006)

[4.2. Data Dictionary: 6](#_Toc217674007)

[4.3. Diagrams: 9](#_Toc217674008)

[5. Nutrition Tracking 14](#_Toc217674009)

[5.1. SRS: 14](#_Toc217674010)

[5.2. Data Dictionary: 15](#_Toc217674011)

[5.3. Diagrams: 19](#_Toc217674012)

[6. Workout Tracking 23](#_Toc217674013)

[6.1. SRS: 23](#_Toc217674014)

[6.2. Data Dictionary: 24](#_Toc217674015)

[6.3. Diagrams: 27](#_Toc217674016)

[7. Rep Count Module 31](#_Toc217674017)

[7.1. SRS: 31](#_Toc217674018)

[7.2. Data Dictionary: 32](#_Toc217674019)

[7.3. Diagrams: 33](#_Toc217674020)

[8. Challenge and Gamification 37](#_Toc217674021)

[8.1. SRS: 37](#_Toc217674022)

[8.2. Data Dictionary: 38](#_Toc217674023)

[8.3. Diagrams: 41](#_Toc217674024)

[9. Community Module 46](#_Toc217674025)

[9.1. SRS: 46](#_Toc217674026)

[9.2. Data Dictionary: 47](#_Toc217674027)

[9.3. Diagrams: 50](#_Toc217674028)

[10. Progress Reports and Analytics 55](#_Toc217674029)

[10.1. SRS: 55](#_Toc217674030)

[10.2. Data Dictionary: 56](#_Toc217674031)

[10.3. Wireframe:Diagrams: 59](#_Toc217674032)

[11. Payment Integration 64](#_Toc217674033)

[11.1. SRS: 64](#_Toc217674034)

[11.2. Data Dictionary: 65](#_Toc217674035)

[11.3. Diagrams: 68](#_Toc217674036)

[12. Gym Discovery and Membership 72](#_Toc217674037)

[12.1. SRS: 72](#_Toc217674038)

[12.2. Data Dictionary: 73](#_Toc217674039)

[12.3. Diagrams: 74](#_Toc217674040)

[13. ERD Diagram of whole system: 78](#_Toc217674041)

[14. Test Plan: 79](#_Toc217674042)

[**14.1.** **Purpose of Testing** **Error! Bookmark not defined.**](#_Toc217674043)

[14.2. Scope of Testing **Error! Bookmark not defined.**](#_Toc217674044)

[14.3. Test Approach **Error! Bookmark not defined.**](#_Toc217674045)

[14.4. Test Environment **Error! Bookmark not defined.**](#_Toc217674046)

[14.5. Roles and Responsibilities **Error! Bookmark not defined.**](#_Toc217674047)

[14.6. Summary of Test Cases 79](#_Toc217674048)

Table of Figures:

[FDD 1 1](#_Toc217672343)

[Planning 1 1](#_Toc217672347)

[Wireframe 1 2](#_Toc217672368)

[Wireframe 2 2](#_Toc217672369)

[Wireframe 3 3](#_Toc217672370)

[Wireframe 4 3](#_Toc217672371)

[Wireframe 5 4](#_Toc217672372)

[Wireframe 6 4](#_Toc217672373)

[User Management 2: Use Case Diagram 9](#_Toc217672380)

[User Management 3: Activity Diagram 10](#_Toc217672381)

[User Management 4: Class Diagram 11](#_Toc217672382)

[User Management 5: Collaboration Diagram 12](#_Toc217672383)

[User Management 6: Sequence Diagram 13](#_Toc217672384)

[Nutrition Tracking 2: Use Case Diagram 19](#_Toc217672399)

[Nutrition Tracking 3: Activity Diagram 20](#_Toc217672400)

[Nutrition Tracking 4: Class Diagram 21](#_Toc217672401)

[Nutrition Tracking 5: Collaboration Diagram 22](#_Toc217672402)

[Nutrition Tracking 6: Sequence Diagram 22](#_Toc217672403)

[Workout Tracking 2: Use Case Diagram 27](#_Toc217672406)

[Workout Tracking 3: Activity Diagram 28](#_Toc217672407)

[Workout Tracking 4: Class Diagram 29](#_Toc217672408)

[Workout Tracking 5: Collaboration Diagram 29](#_Toc217672409)

[Workout Tracking 6: Sequence Diagram 30](#_Toc217672410)

[Rep Count 1: Use Case Diagram 33](#_Toc217672439)

[Rep Count 2: Activity Diagram 34](#_Toc217672440)

[Rep Count 3: Class Diagram 35](#_Toc217672441)

[Rep Count 4: Collaboaration Diagram 35](#_Toc217672442)

[Rep Count 5: Sequence Diagram 36](#_Toc217672443)

[Challenge&Game 2:Use Case Diagram 41](#_Toc217672446)

[Challenge&Game 3: Activity Diagram 42](#_Toc217672447)

[Challenge&Game 4: Class Diagram 43](#_Toc217672448)

[Challenge&Game 5: Collaboration Diagram 44](#_Toc217672449)

[Challenge&Game 6: Sequence Diagram 45](#_Toc217672450)

[Community Page 2: Use Case Diagram 50](#_Toc217672464)

[Community Page 3: Activity Diagram 51](#_Toc217672465)

[Community Page 4: Class Diagram 52](#_Toc217672466)

[Community Page 5: Collaboration Diagram 53](#_Toc217672467)

[Community Page 6: Sequence Diagram 54](#_Toc217672468)

[Progress Report 1: Use Case Diagram 59](#_Toc217672476)

[Progress Report 2: Activity Diagram 60](#_Toc217672477)

[Progress Report 3: Class Diagram 61](#_Toc217672478)

[Progress Report 4: Collaboration Diagram 62](#_Toc217672479)

[Progress Report 5: Sequence Diagra 63](#_Toc217672480)

[Payment Integration 1: Use Case Diagram 68](#_Toc217672483)

[Payment Integration 2: Activity Diagram 69](#_Toc217672484)

[Payment Integration 3: Class Diagram 70](#_Toc217672485)

[Payment Integration 4: Collaboration Diagram 71](#_Toc217672486)

[Payment Integration 5: Sequence Diagram 71](#_Toc217672487)

[Gym Discovery 1: Use Case Diagram 74](#_Toc217672493)

[Gym Discovery 2: Activity Diagram 75](#_Toc217672494)

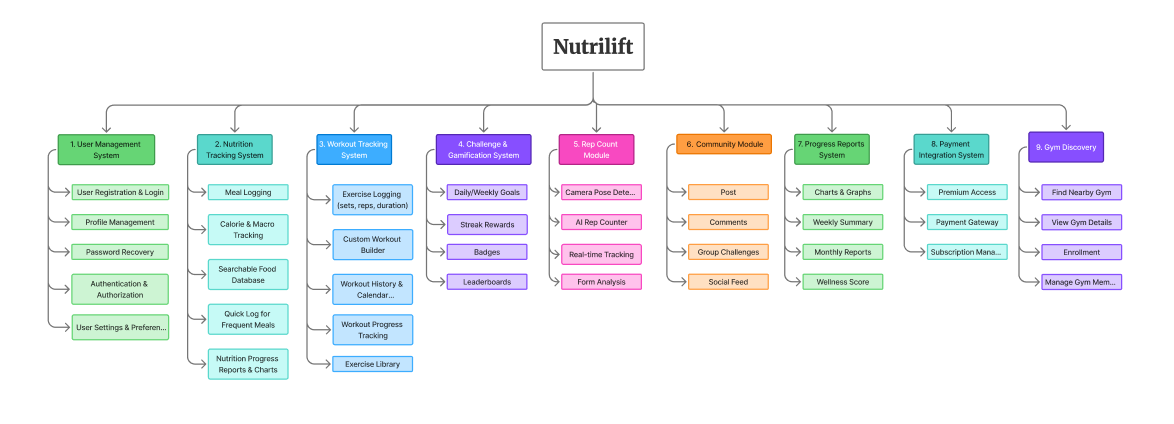
[Gym Discovery 3: Class Diagram 76](#_Toc217672495)

[Gym Discovery 4: Collaboration Diagram 77](#_Toc217672496)

[Gym Discovery 5: Sequence Diagram 77](#_Toc217672497)

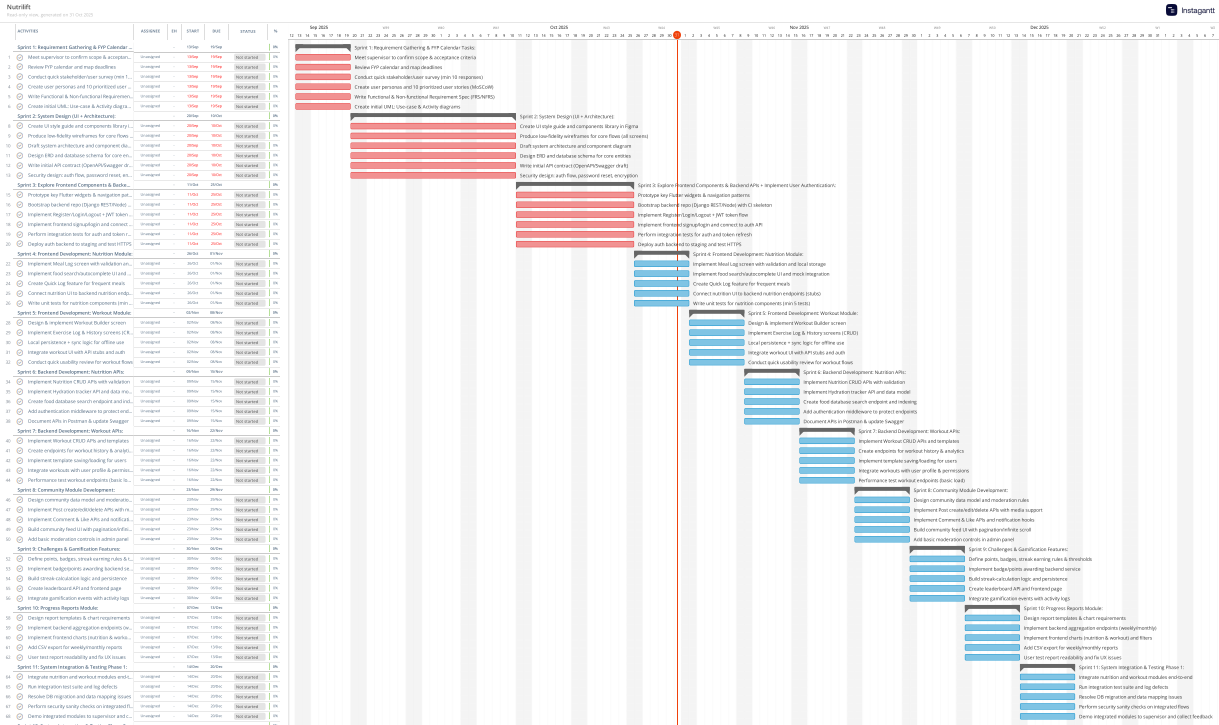
[ERD Diagram 1 78](#_Toc217672508)

# FDD Diagram:



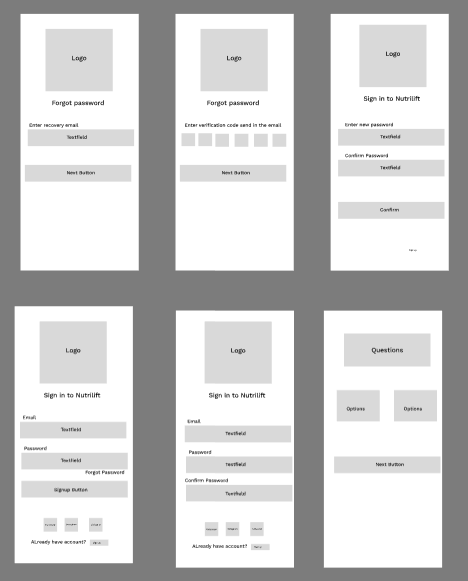
FDD 1

# Planning:

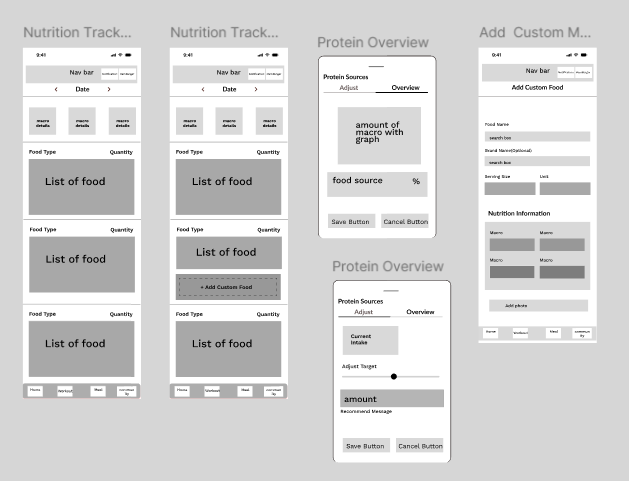


Planning 1

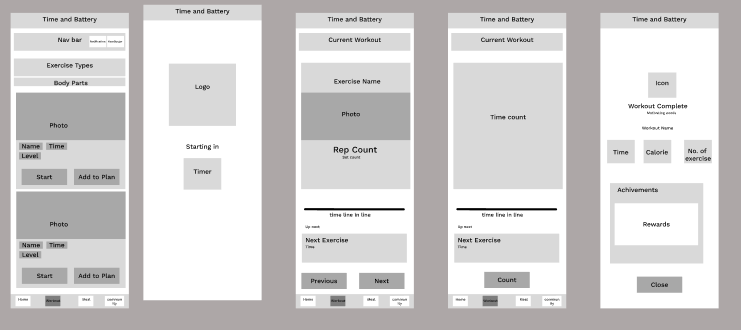
# Wire Frame:



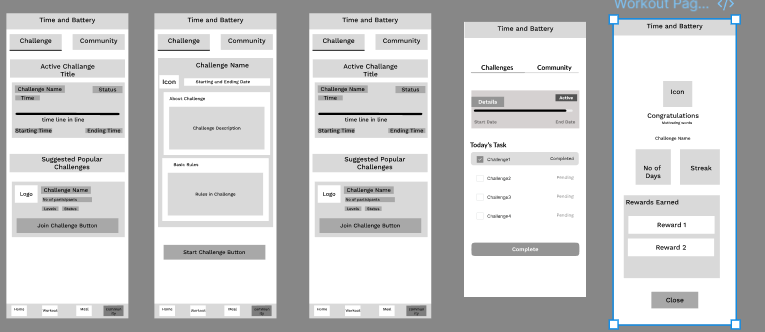
Wireframe 1



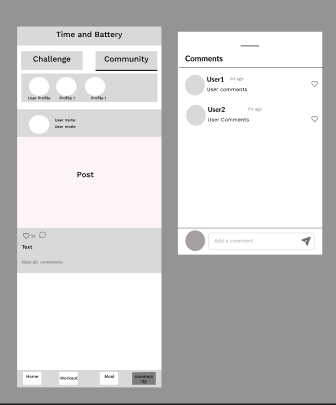
Wireframe 2



Wireframe 3



Wireframe 4



Wireframe 5



Wireframe 6

# User Management:

This subsystem looks after user accounts, including registration, login, logout and password reset. It also stores profile details and preferences so other parts of the system can tailor calculations and behaviour to each user.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **UM-F-1.0** | As a user, I want to register with email and password so that I can create an account. | User Management | Registration works with valid email, unique records created, invalid inputs rejected. |
| **UM-F-2.0** | As a user, I want to log in and log out securely. | User Management | Valid credentials open session, invalid ones denied, logout ends session safely. |
| **UM-F-3.0** | As a user, I want to reset my password via email. | User Management | Token email sent, reset link updates password successfully. |
| **UM-F-4.0** | As a user, I want to edit my profile (age, weight, goal). | User Management | Saved changes reappear correctly after reopening profile. |
| **UM-F-5.0** | As a user, I want to set dietary preferences. | User Management | Preferences saved and available for food filtering. |
| **UM-NF-1.1** | Registration data must be transmitted securely. | User Management | All credentials transferred via HTTPS only. |
| **UM-UR-1.1** | Password fields should hide/unhide easily on mobiles. | User Management | Toggle icon works intuitively on touch devices. |

## Data Dictionary:

Entity: USER

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Unique identifier for each user. |
| email | String | Login email address. |
| password\_hash | String | Hashed password. |
| name | String | Full name. |
| age | Integer | Age in years. |
| weight | Float | Current weight. |
| height | Float | Height. |
| gender | String | Gender value. |
| fitness\_goal | String | Main fitness goal. |
| dietary\_preferences | JSON | Dietary preferences/allergies. |
| is\_premium | Boolean | Whether user has active premium. |
| subscription\_expires\_at | Datetime | Premium expiry timestamp. |
| is\_active | Boolean | Account active or deactivated. |
| created\_at | Datetime | Record creation time. |
| updated\_at | Datetime | Last update time. |

Entity: USER\_SESSION

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Session identifier. |
| user\_id | UUID | References USER(id). |
| jwt\_token | String | JSON Web Token used for authentication. |
| expires\_at | Datetime | Token expiry datetime. |
| device\_info | String | Device or platform information. |
| ip\_address | String | IP address used. |
| created\_at | Datetime | Session creation time. |

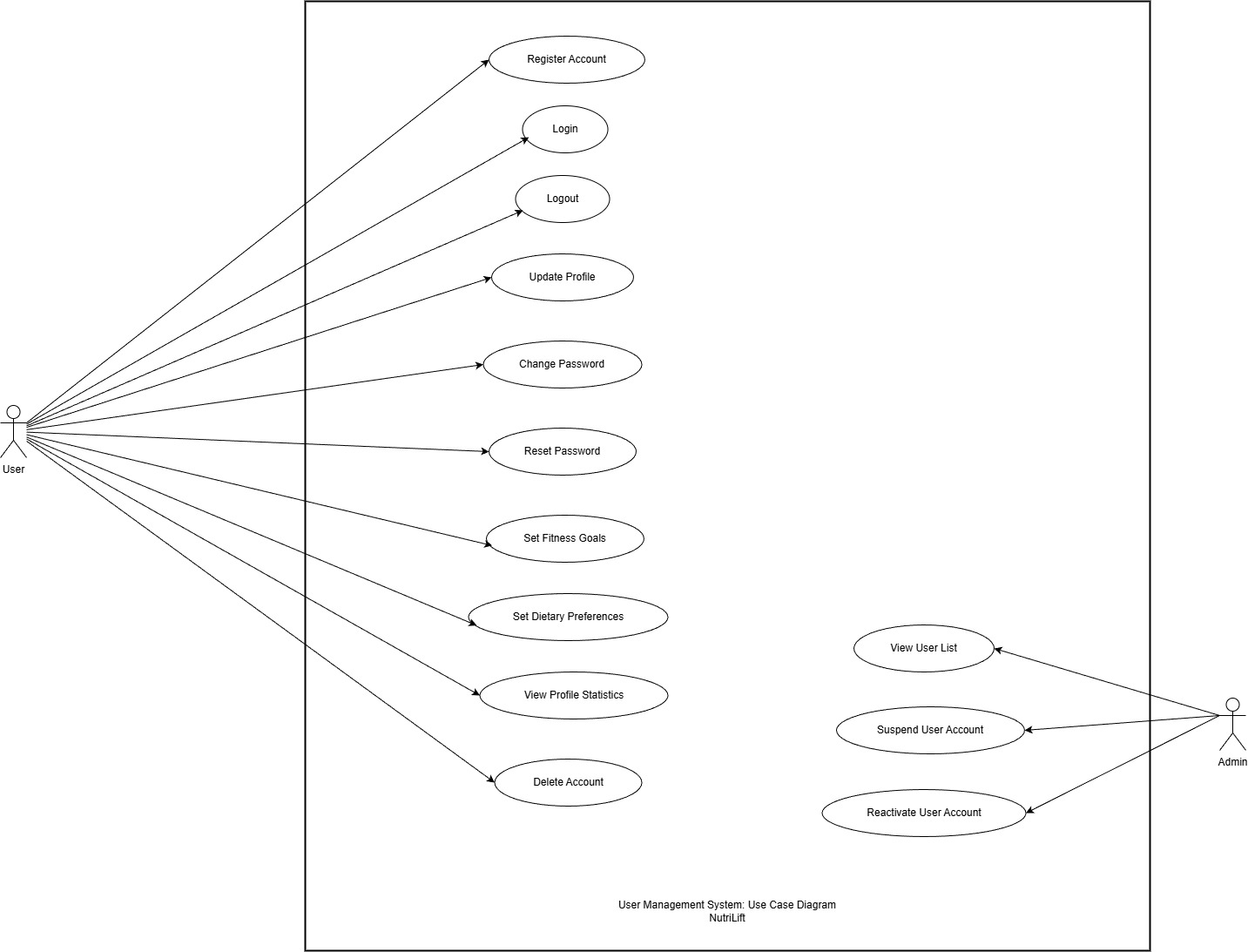
Entity: PASSWORD\_RESET

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Password reset request id. |
| user\_id | UUID | References USER(id). |
| reset\_token | String | Secure reset token. |
| expires\_at | Datetime | Token expiry time. |
| is\_used | Boolean | Whether token has been used. |
| created\_at | Datetime | Request creation time. |

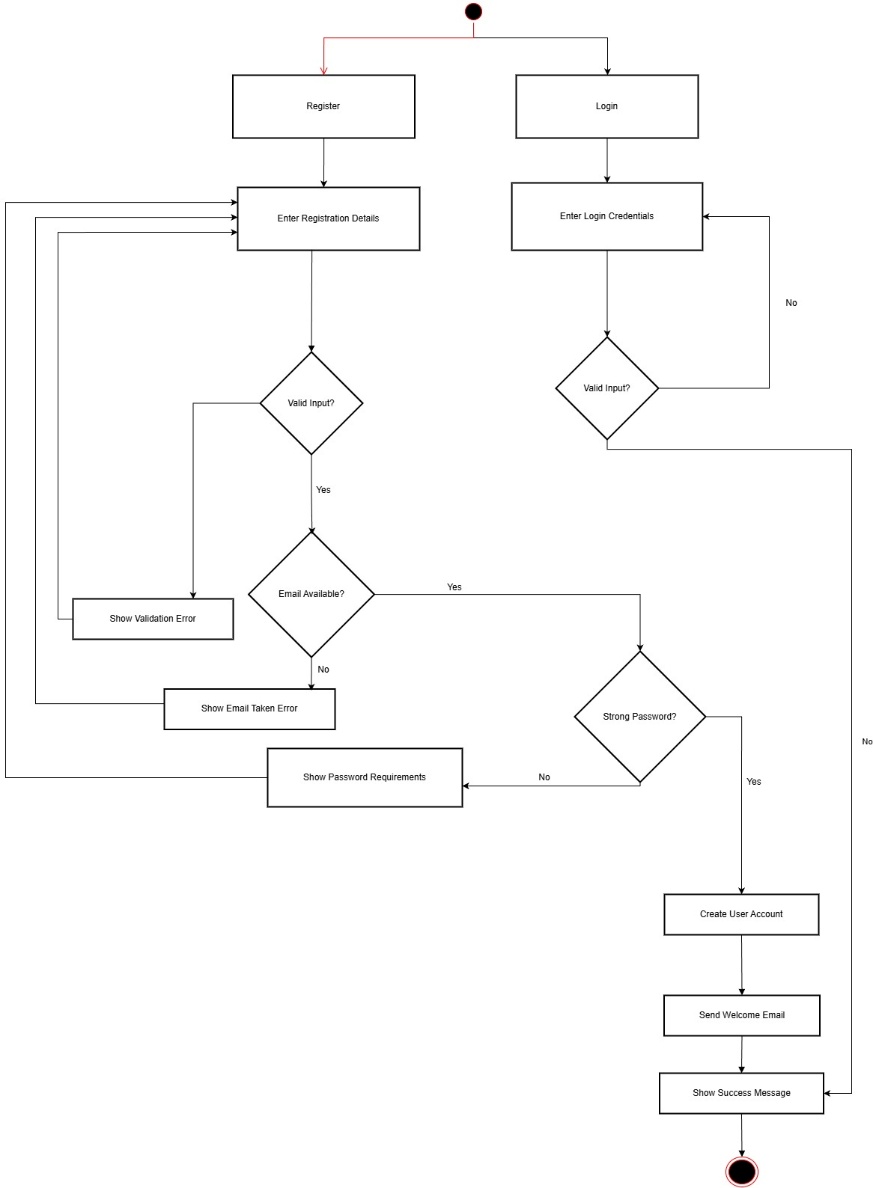
Entity; USER\_PREFERENCES

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Preferences id. |
| user\_id | UUID | References USER(id). |
| notification\_settings | JSON | Notification configuration settings. |
| language | String | Preferred language code. |
| timezone | String | Time zone of the user. |
| privacy\_settings | JSON | Privacy options. |
| updated\_at | Datetime | Last update time. |

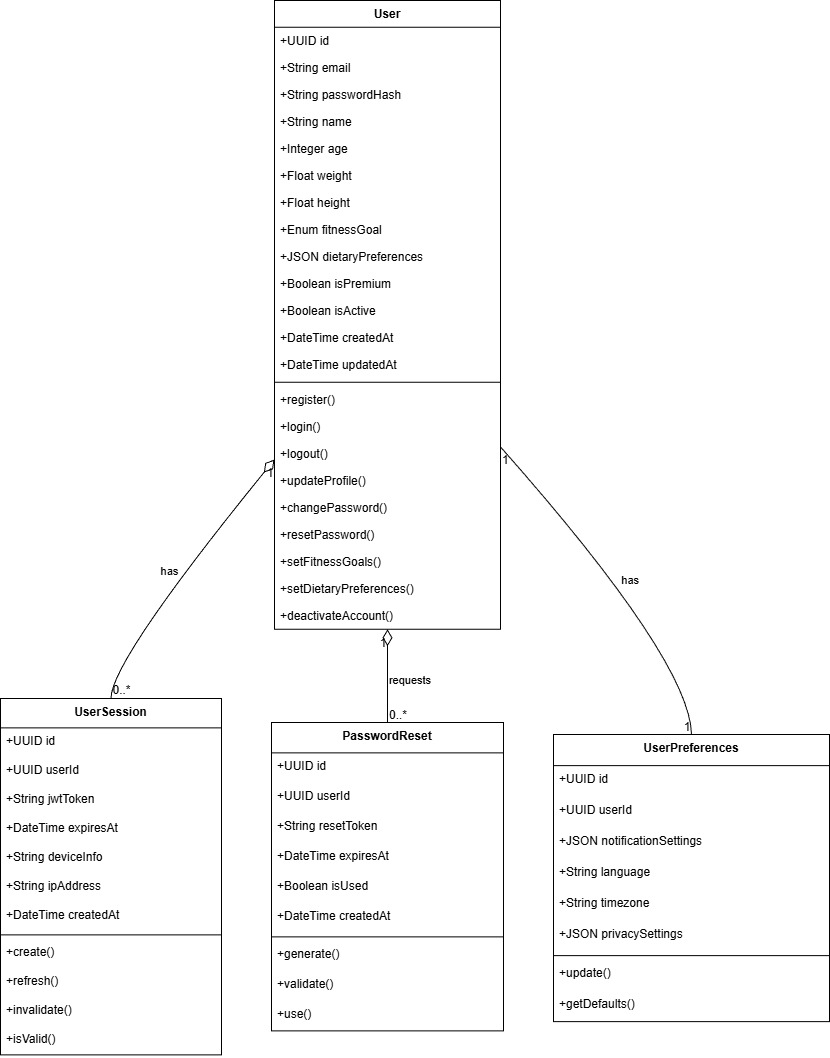
## Diagrams:



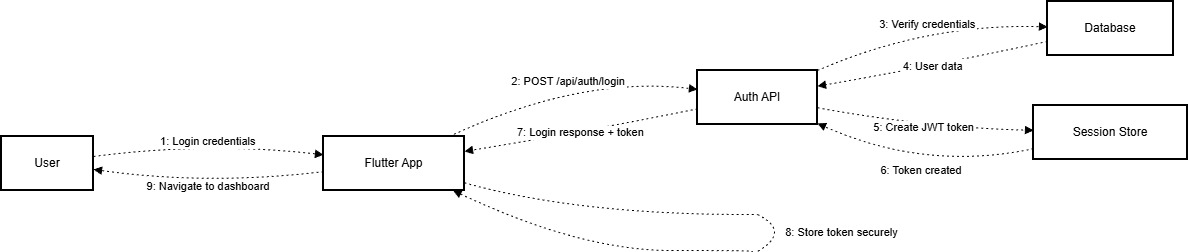
User Management 1: Use Case Diagram



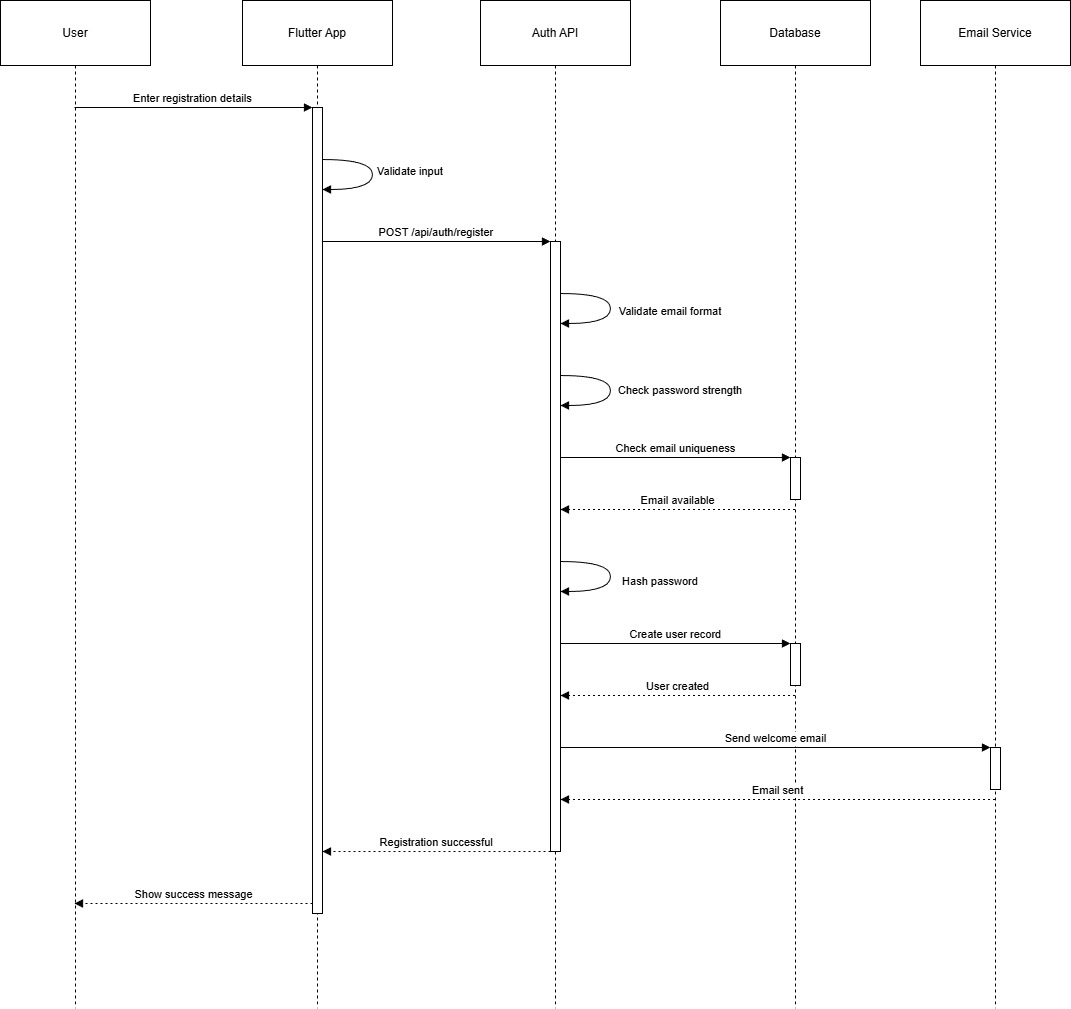
User Management 2: Activity Diagram



User Management 3: Class Diagram



User Management 4: Collaboration Diagram



User Management 5: Sequence Diagram

# Nutrition Tracking

This subsystem records what the user eats and drinks in a day and calculates calories and macros. It helps the user stay on target by showing daily totals, goals and frequently used meals for quick logging.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **NT-F-1.0** | As a user, I want to log meals with calories and macros. | Nutrition Tracking | Meal saved, totals update instantly. |
| **NT-F-2.0** | As a user, I want to search foods from a database. | Nutrition Tracking | Search returns matches; custom foods allowed. |
| **NT-F-3.0** | As a user, I want to add custom food entries. | Nutrition Tracking | Custom item created and reusable in logs. |
| **NT-F-4.0** | As a user, I want to log water intake. | Nutrition Tracking | Water entry updates total and progress bar. |
| **NT-F-5.0** | As a user, I want to set daily nutrition goals. | Nutrition Tracking | Targets saved and shown beside daily totals. |
| **NT-F-6.0** | As a user, I want to view a daily nutrition summary. | Nutrition Tracking | Summary card displays calorie and macro totals. |
| **NT-F-7.0** | As a user, I want quick access to frequent meals. | Nutrition Tracking | Recent meals list appears for faster logging. |
| **NT-F-8.0** | As a user, I want to edit or delete logged meals. | Nutrition Tracking | Edits recalculate totals; deletes remove entries. |

## Data Dictionary:

Entity: FOOD\_ITEM

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Food item id. |
| name | String | Food name. |
| barcode | String | Barcode string. |
| calories\_per\_100g | Float | Energy per 100g of food. |
| protein\_per\_100g | Float | Protein per 100g. |
| carbs\_per\_100g | Float | Carbohydrates per 100g. |
| fats\_per\_100g | Float | Fats per 100g. |
| fiber\_per\_100g | Float | Fiber per 100g. |
| sugar\_per\_100g | Float | Sugar per 100g. |
| is\_custom | Boolean | Indicates if created by user or from global DB. |
| created\_by | UUID | References USER(id) for custom foods. |
| created\_at | Datetime | Food creation time. |

Entity: INTAKE\_LOG

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Intake log id. |
| user\_id | UUID | References USER(id). |
| entry\_type | String | Type of entry: meal, snack, drink, water. |
| food\_item\_id | UUID | References FOOD\_ITEM(id) if a defined food is selected. |
| description | String | Free-text description of intake. |
| quantity | Float | Amount consumed. |
| unit | String | Unit such as g, ml, piece. |
| logged\_at | Datetime | When intake was logged. |
| calories | Float | Calculated calories for this entry. |
| protein | Float | Calculated protein. |
| carbs | Float | Calculated carbs. |
| fats | Float | Calculated fats. |
| created\_at | Datetime | Record creation time. |

Entity: HYDRATION\_LOG8

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Hydration entry id. |
| user\_id | UUID | References USER(id). |
| amount | Float | Volume of water consumed. |
| unit | String | Unit such as ml or L. |
| logged\_at | Datetime | Time of hydration entry. |

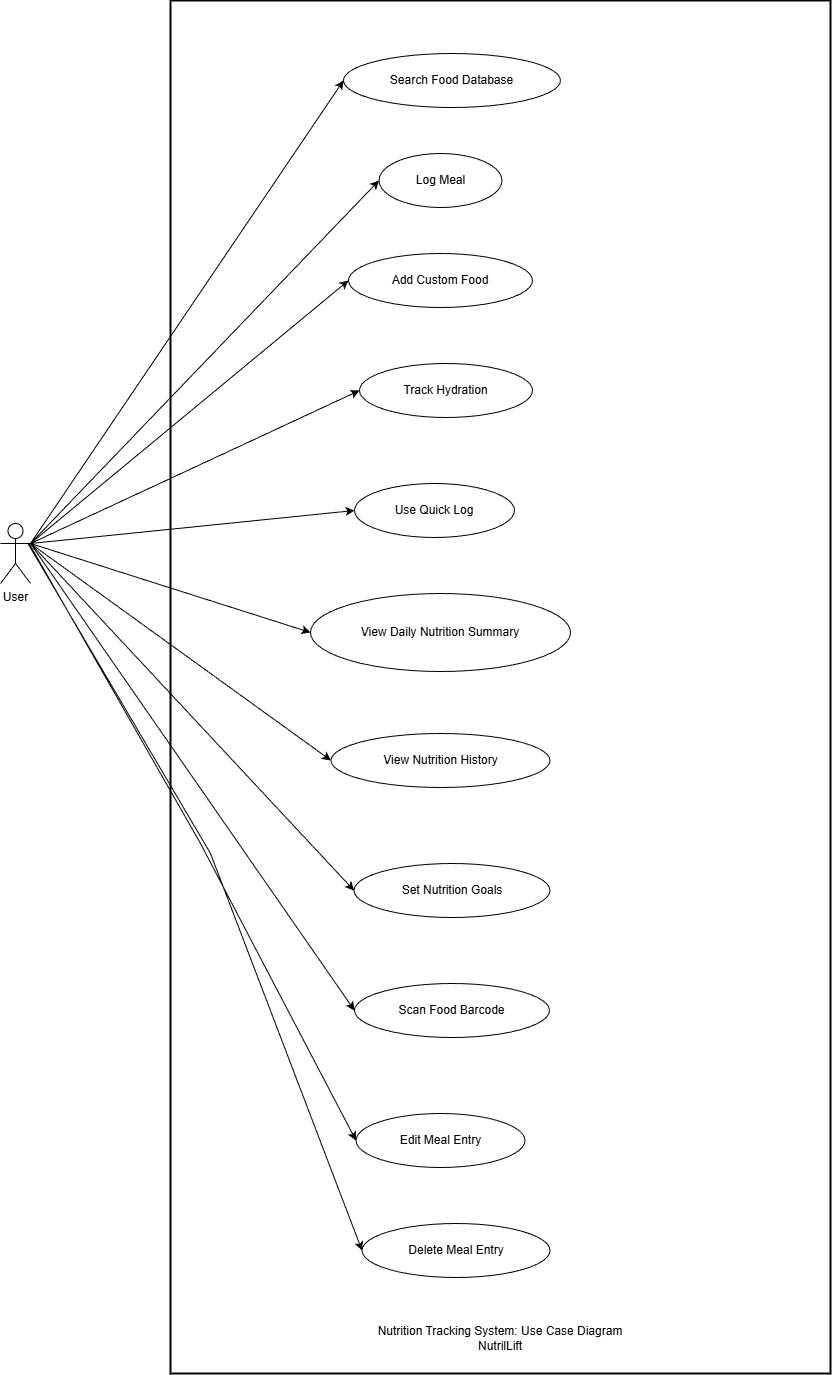
Entity: NUTRITION\_GOALS

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Nutrition goals id. |
| user\_id | UUID | References USER(id). |
| daily\_calories | Float | Daily target calories. |
| daily\_protein | Float | Daily target protein. |
| daily\_carbs | Float | Daily target carbs. |
| daily\_fats | Float | Daily target fats. |
| daily\_water | Float | Daily target water intake. |
| updated\_at | Datetime | Last update time. |

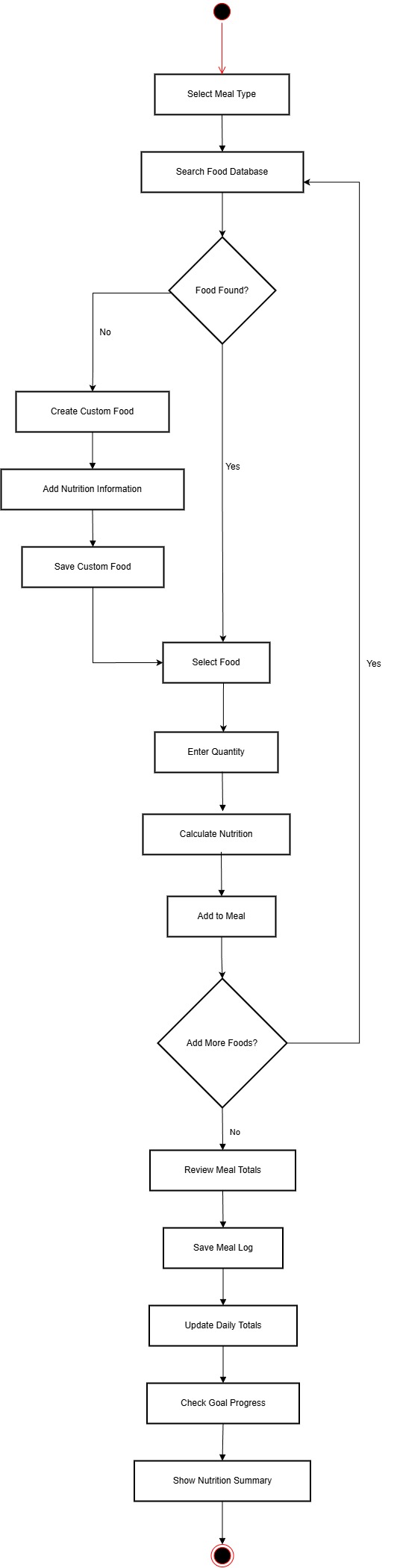
Entity: QUICK\_LOG

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Quick log id. |
| user\_id | UUID | References USER(id). |
| frequent\_meals | JSON | Cached frequent meals and templates. |
| updated\_at | Datetime | Last update time. |

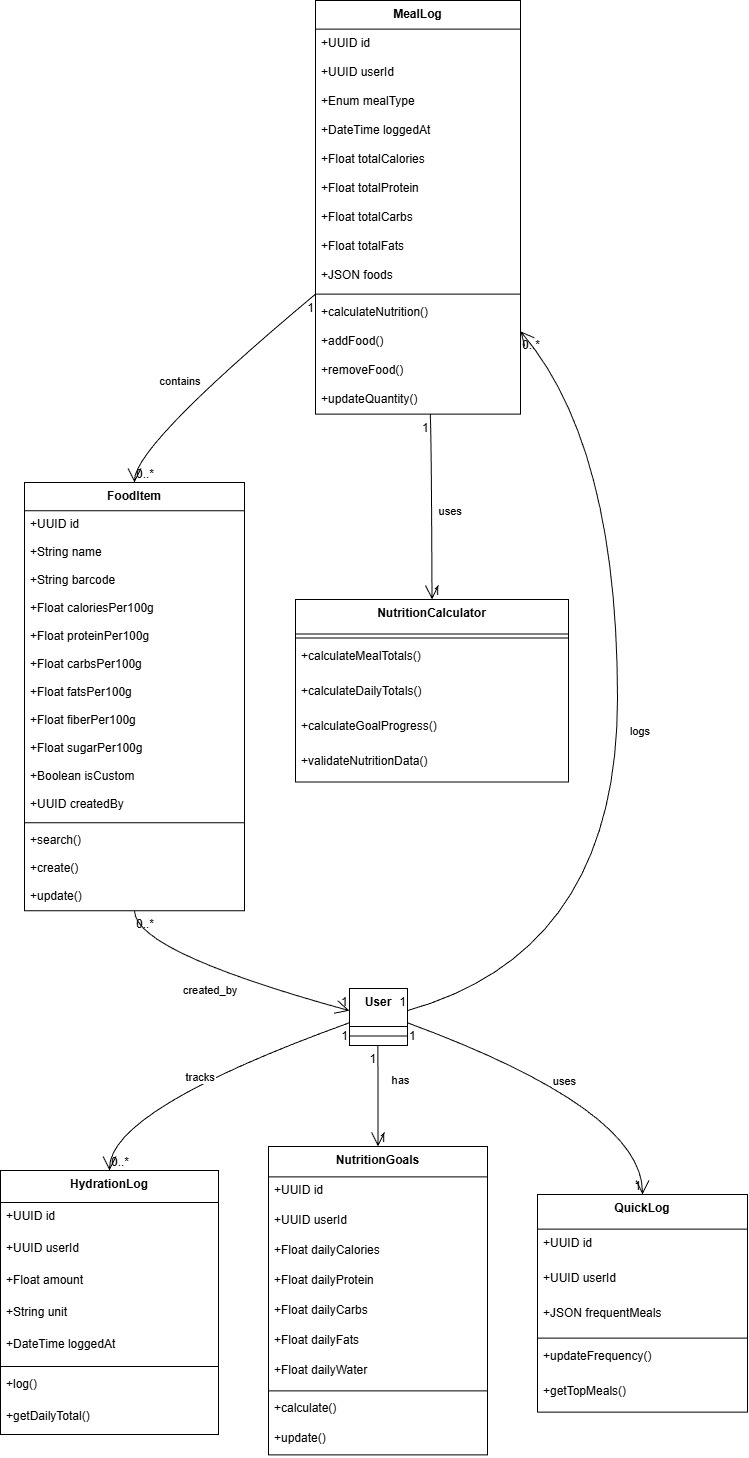
## Diagrams:



Nutrition Tracking 1: Use Case Diagram



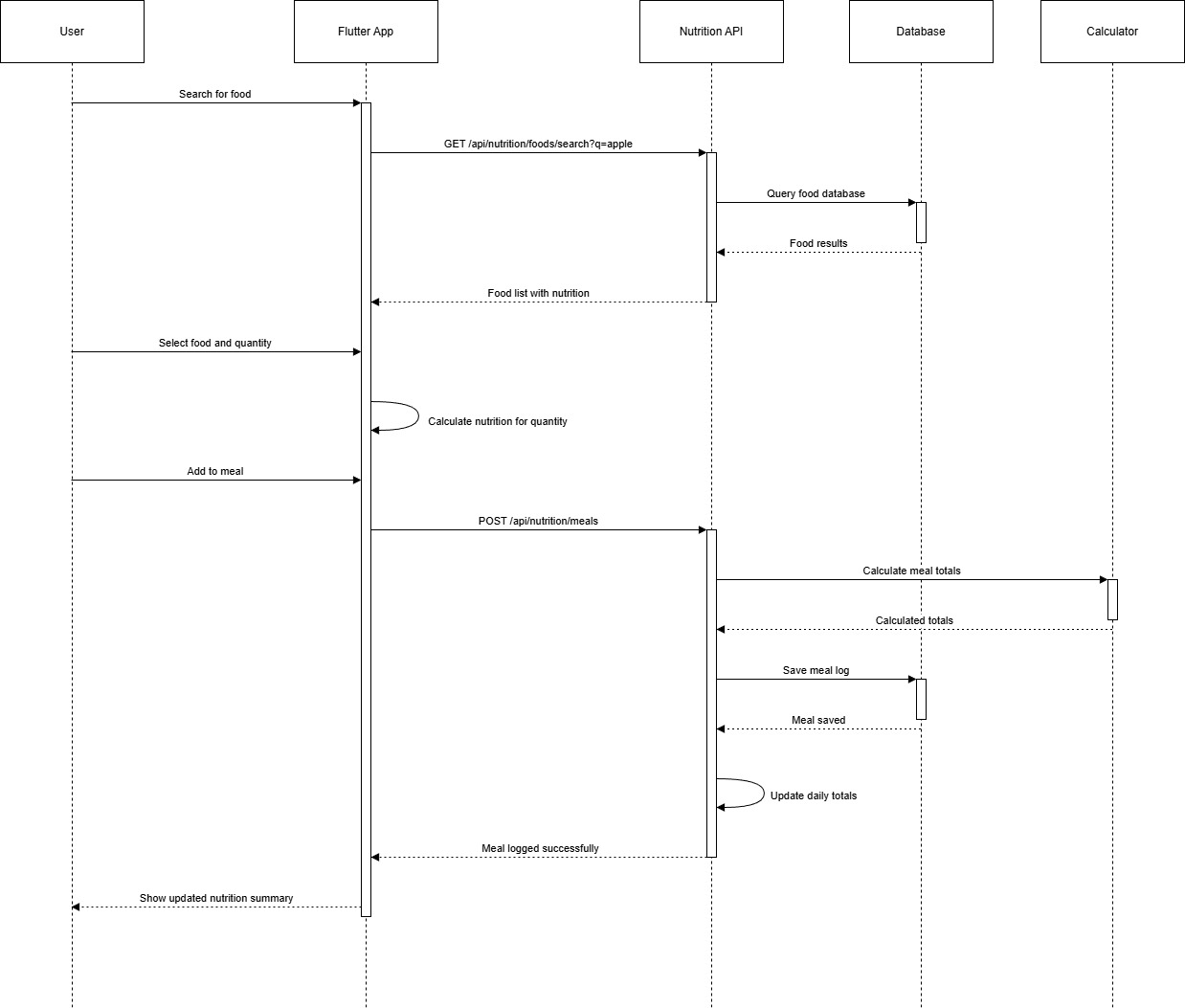
Nutrition Tracking 2: Activity Diagram

**

Nutrition Tracking 3: Class Diagram



Nutrition Tracking 4: Collaboration Diagram



Nutrition Tracking 5: Sequence Diagram

# Workout Tracking

This subsystem captures gym and home workouts with exercises, sets, reps and weights. It keeps a history of past sessions and supports templates so users can repeat routines and see how their training changes over time.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **WT-F-1.0** | As a user, I want to log workouts with exercises and reps. | Workout Tracking | Records save correctly and visible in history. |
| **WT-F-2.0** | As a user, I want to browse the exercise library. | Workout Tracking | List shows exercises with category and muscles. |
| **WT-F-3.0** | As a user, I want to view my workout history. | Workout Tracking | History lists previous workouts in date order. |
| **WT-F-4.0** | As a user, I want reusable workout templates. | Workout Tracking | Templates save, edit, and load into new sessions. |
| **WT-F-5.0** | As a user, I want calorie estimates for workouts. | Workout Tracking | Workout record includes estimated calories burned. |
| **WT-F-6.0** | As a user, I want to see personal bests per exercise. | Workout Tracking | Best weight/reps updated automatically. |
| **WT-F-7.0** | As a user, I want to link workouts to a gym. | Workout Tracking | Gym field optional; saved location shown in history. |

## Data Dictionary:

Entity: EXERCISE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Exercise id. |
| name | String | Exercise name. |
| category | String | Category such as strength or cardio. |
| muscle\_groups | JSON | Target muscle groups. |
| instructions | Text | How to perform the exercise. |
| video\_url | String | Tutorial video link. |
| image\_url | String | Exercise image link. |
| difficulty | String | Difficulty level. |
| equipment | String | Required equipment. |
| created\_at | Datetime | Creation time. |

Entity: WORKOUT\_LOG

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Workout log id. |
| user\_id | UUID | References USER(id). |
| workout\_name | String | User-visible workout name. |
| logged\_at | Datetime | When workout was completed. |
| duration\_minutes | Integer | Length of workout in minutes. |
| calories\_burned | Float | Estimated calories burned. |
| exercises | JSON | Per-exercise sets, reps and weights. |
| notes | Text | User notes about the workout. |
| gym\_id | UUID | References GYM(id) if workout done at a gym. |
| created\_at | Datetime | Record creation time. |

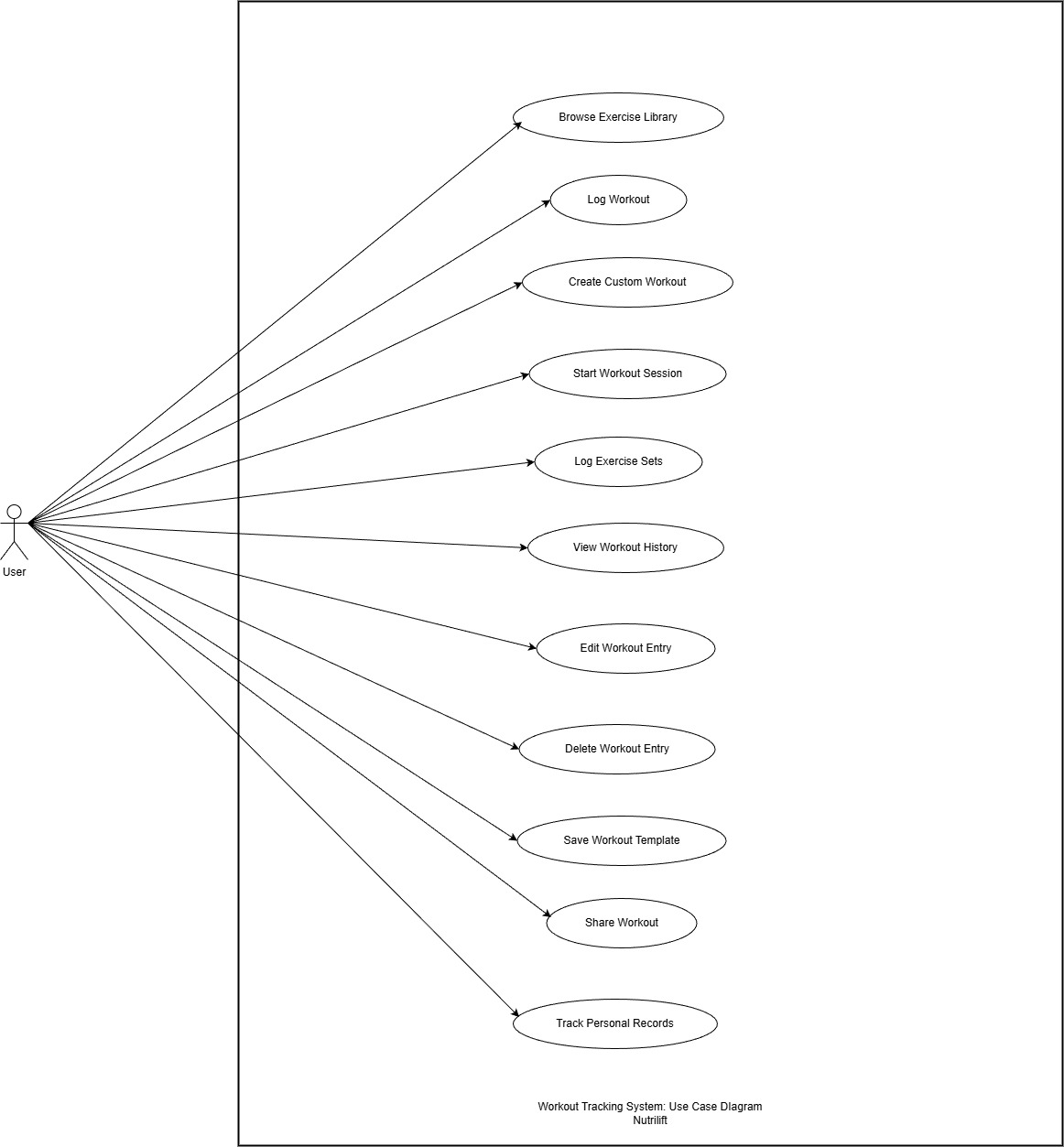
Entity: CUSTOM\_WORKOUT

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Custom workout id. |
| user\_id | UUID | References USER(id). |
| name | String | Custom workout name. |
| description | Text | Description of the routine. |
| exercises | JSON | List of exercise references with structure. |
| estimated\_duration | Integer | Estimated duration in minutes. |
| is\_public | Boolean | Indicates if workout template is shareable. |
| created\_at | Datetime | Creation time. |
| updated\_at | Datetime | Last update time. |

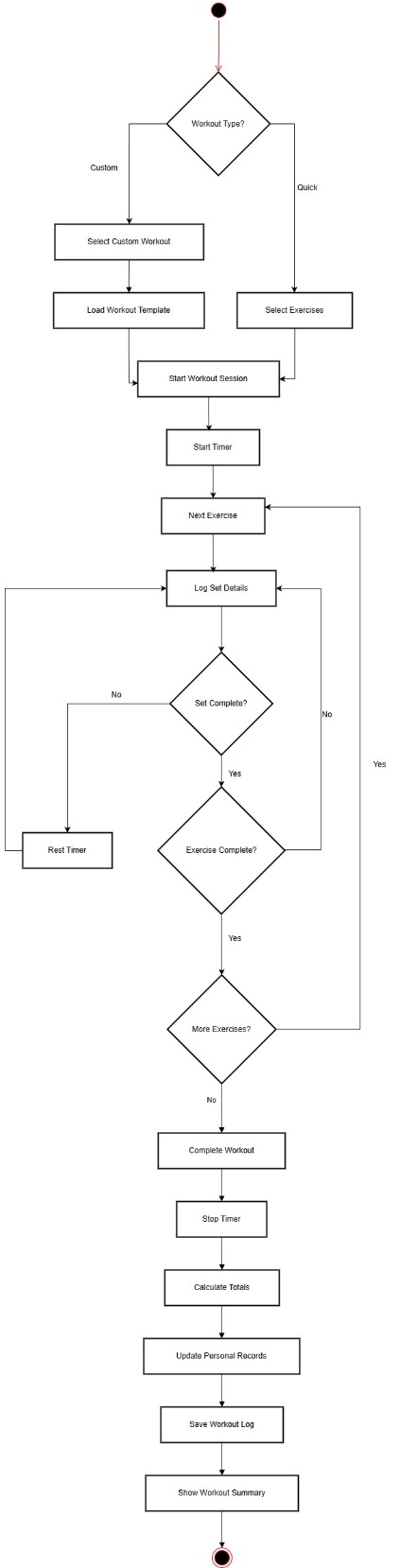
Entity: PERSONAL\_RECORD

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Personal record id. |
| user\_id | UUID | References USER(id). |
| exercise\_id | UUID | References EXERCISE(id). |
| max\_weight | Float | Maximum weight lifted. |
| max\_reps | Integer | Maximum repetitions performed. |
| max\_volume | Float | Total volume lifted for best set or session. |
| achieved\_at | Datetime | Time when record was achieved. |

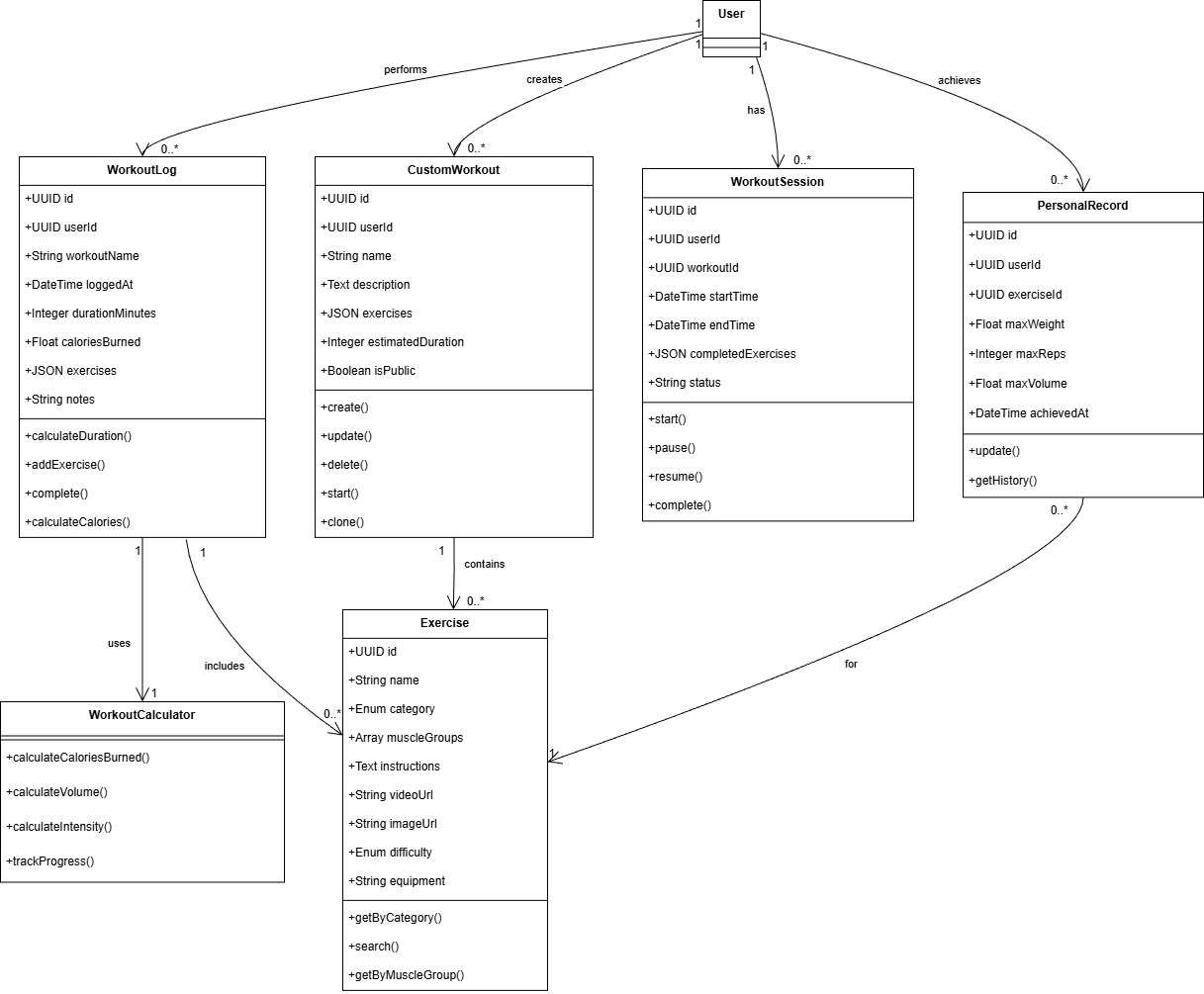
## Diagrams:



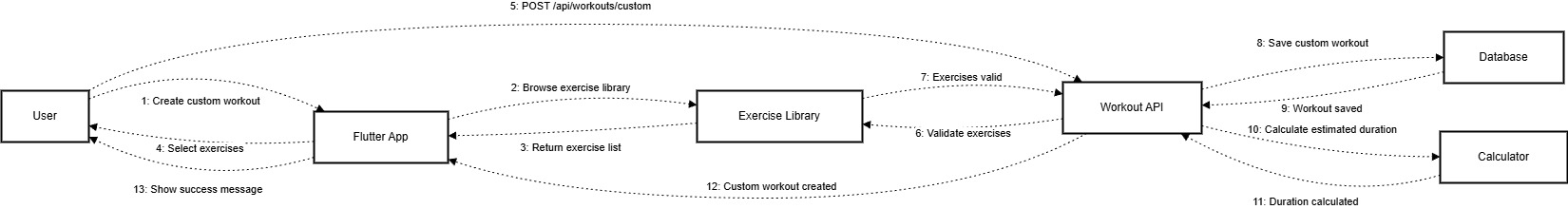
Workout Tracking 1: Use Case Diagram



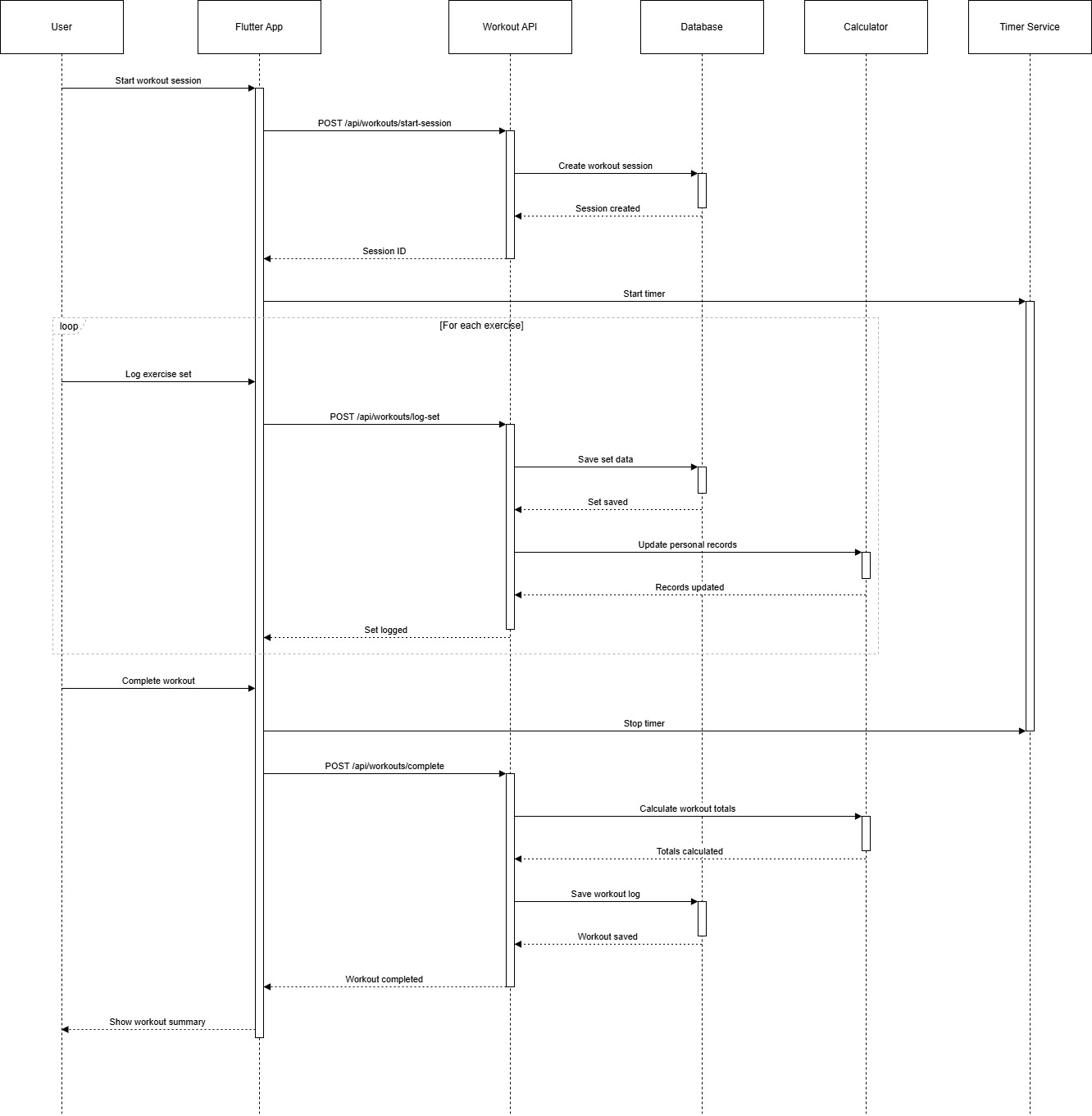
Workout Tracking 2: Activity Diagram



Workout Tracking 3: Class Diagram



Workout Tracking 4: Collaboration Diagram



Workout Tracking 5: Sequence Diagram

# Rep Count Module

This subsystem uses the device camera to count repetitions for supported exercises automatically. It shows a live rep counter during the session and saves basic session details for later review.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **RC-F-1.0** | As a user, I want the app to automatically count reps using the camera so that I do not have to count manually. | Rep Count Module | Rep session uses camera and counts about 10 reps with small error. SRS\_NutriLift.docx​ |
| **RC-F-2.0** | As a user, I want to see a live rep counter during the exercise so that I know the system is working. | Rep Count Module | Current rep number is visible and increases as reps are detected. SRS\_NutriLift.docx​ |
| **RC-F-3.0** | As a user, I want rep sessions to be saved so that I can review them later. | Rep Count Module | Finished sessions appear in a history with exercise, reps and time. SRS\_NutriLift.docx​ |
| **RC-F-4.0** | As a user, I want to know when tracking quality is low so that I can adjust my position. | Rep Count Module | App shows a simple warning when angle or light is bad. SRS\_NutriLift.docx​ |

## Data Dictionary:

Entity: REP\_SESSION

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Rep counting session id. |
| user\_id | UUID | References USER(id). |
| exercise\_id | UUID | References EXERCISE(id). |
| start\_time | Datetime | Session start time. |
| end\_time | Datetime | Session end time. |
| total\_reps | Integer | Total repetitions detected by the model. |
| source\_video | String | Optional video file path or URL. |

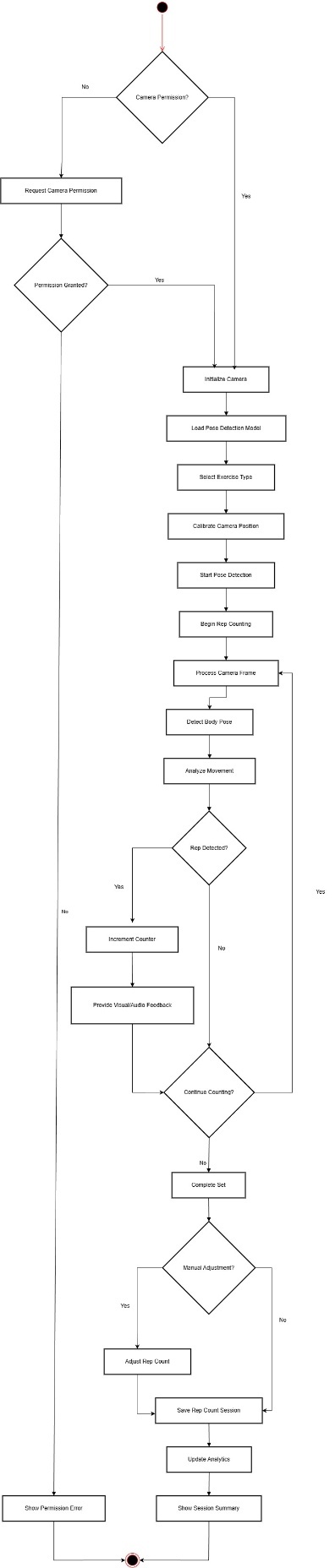
Entity: REP\_EVENT

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Rep event id. |
| session\_id | UUID | References REP\_SESSION(id). |
| rep\_number | Integer | Sequential rep count within the session. |
| timestamp | Datetime | Time when rep was detected. |
| confidence | Float | Model confidence score for this rep. |

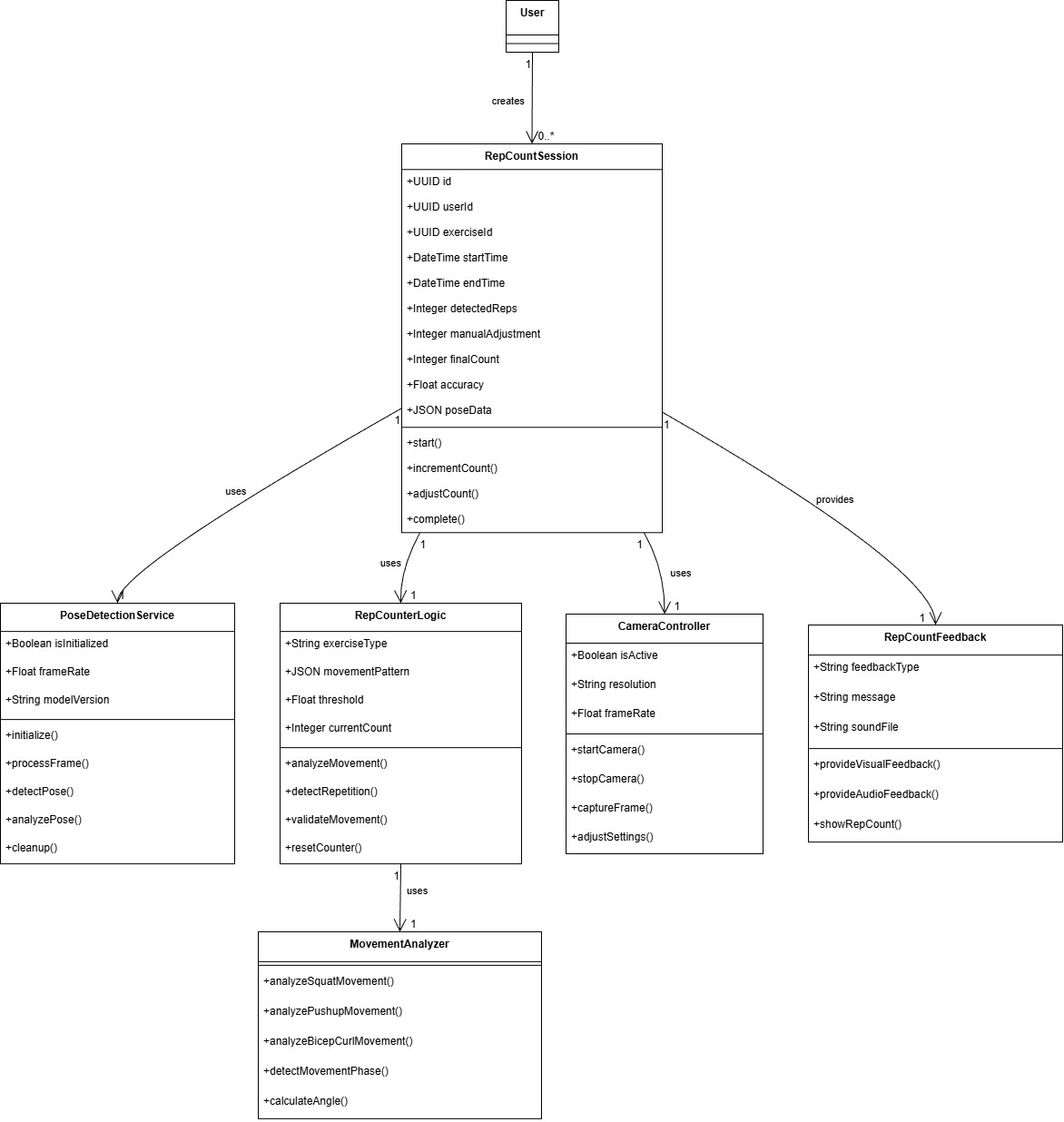
## Diagrams:



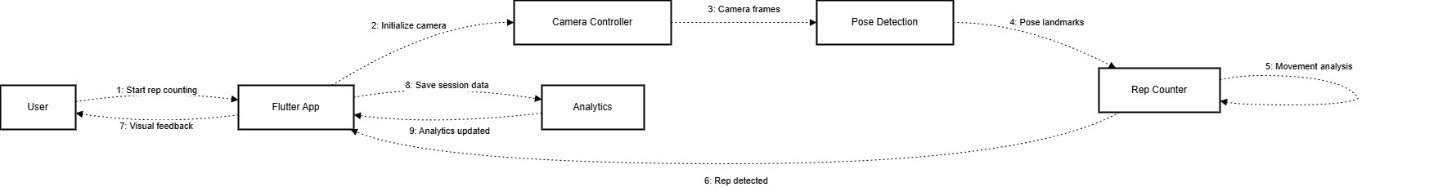
Rep Count 1: Use Case Diagram



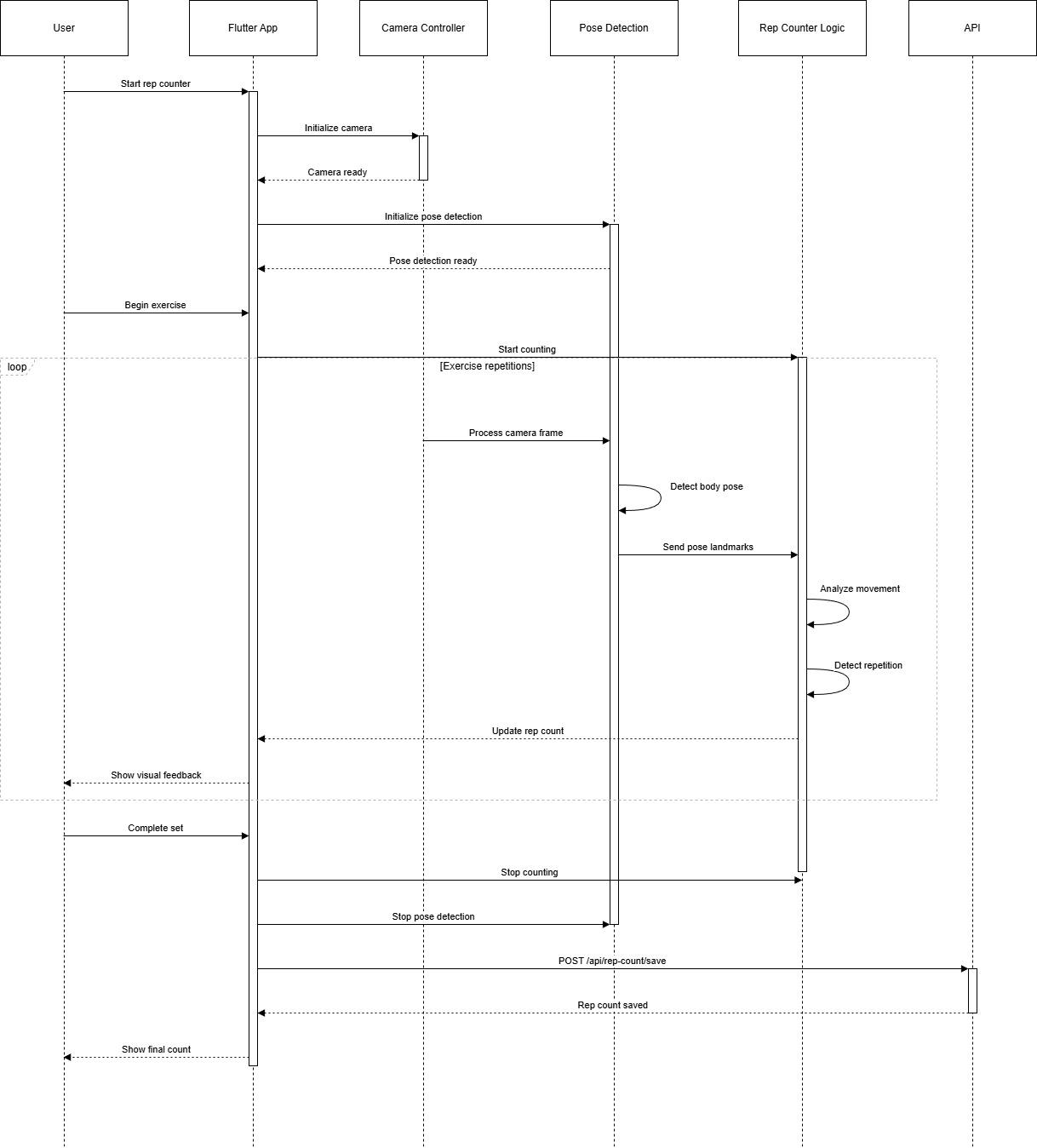
Rep Count 2: Activity Diagram



Rep Count 3: Class Diagram



Rep Count 4: Collaboaration Diagram



Rep Count 5: Sequence Diagram

# Challenge and Gamification

This subsystem provides challenges, streaks, badges and leaderboards to make consistent activity more rewarding. It updates progress using existing nutrition and workout data so users do not have to enter anything twice.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **CG-F-1.0** | As a user, I want to browse and join active challenges so that I stay motivated. | Challenges & Gamification | Active challenges list shows name, metric, goal and dates; join status is stored. SRS\_NutriLift.docx​ |
| **CG-F-2.0** | As a user, I want my challenge progress to update automatically from my nutrition and workout logs so that there is no double entry. | Challenges & Gamification | Logging food or workouts updates joined challenge progress. SRS\_NutriLift.docx​ |
| **CG-F-3.0** | As a user, I want to see a clear start and end date for each challenge so that I know how long I have. | Challenges & Gamification | Detail page shows start, end and remaining time; challenge closes after end date. SRS\_NutriLift.docx​ |
| **CG-F-4.0** | As a user, I want streaks for consecutive active days so that I am rewarded for consistency. | Challenges & Gamification | Current and longest streak values change when days are active or missed. SRS\_NutriLift.docx​ |
| **CG-F-5.0** | As a user, I want badges for specific milestones so that my achievements are visible. | Challenges & Gamification | Hitting a milestone adds a badge that appears in a badge list on profile. SRS\_NutriLift.docx​ |
| **CG-F-6.0** | As a user, I want a simple leaderboard so that I can compare my performance with others. | Challenges & Gamification | Leaderboard shows top users and the logged‑in user’s rank. SRS\_NutriLift.docx​ |

## Data Dictionary:

Entity: CHALLENGE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Challenge id. |
| name | String | Challenge name. |
| description | Text | Description and rules. |
| challenge\_type | String | Type such as nutrition, workout, mixed. |
| goal\_value | Float | Target metric to achieve. |
| unit | String | Unit such as kcal, reps, days. |
| start\_date | Datetime | Challenge start date and time. |
| end\_date | Datetime | Challenge end date and time. |
| created\_by | UUID | References USER(id) as creator. |
| is\_active | Boolean | Indicates if challenge is currently active. |
| created\_at | Datetime | Creation time. |

Entity: CHALLENGE\_PARTICIPANT

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Challenge participant id. |
| challenge\_id | UUID | References CHALLENGE(id). |
| user\_id | UUID | References USER(id). |
| progress | Float | Current progress towards goal. |
| completed | Boolean | Completion status of the challenge. |
| joined\_at | Datetime | Time user joined the challenge. |
| completed\_at | Datetime | Time challenge was completed by the user. |
| rank | Integer | User rank on the challenge leaderboard. |

Entity: BADGE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Badge id. |
| name | String | Badge name. |
| description | Text | Explanation of the badge. |
| icon\_url | String | Badge icon path or URL. |
| criteria | JSON | Rules or thresholds for awarding the badge. |
| points\_reward | Integer | Points awarded when badge is earned. |
| is\_active | Boolean | Indicates if badge is active. |
| created\_at | Datetime | Creation time. |

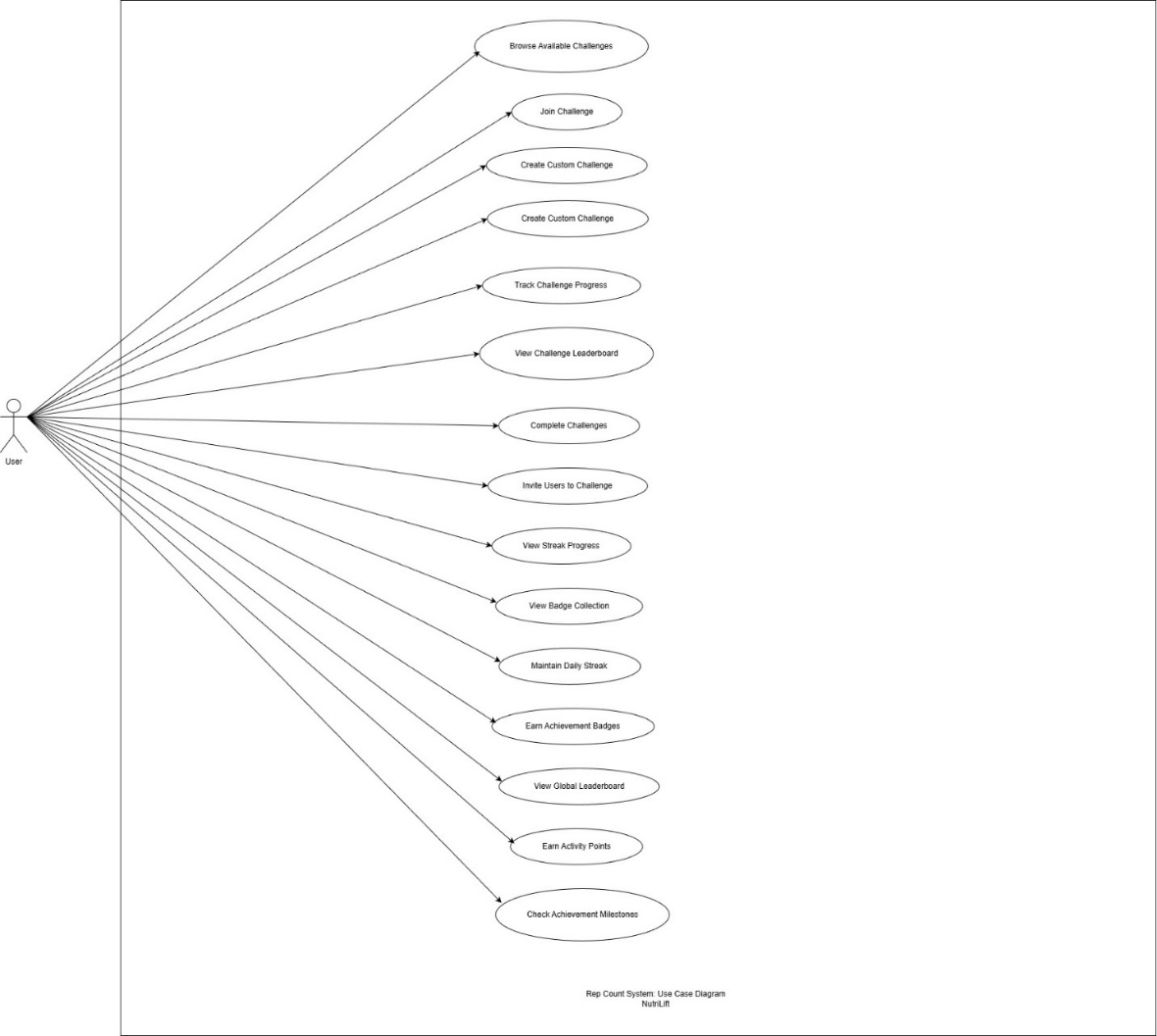
Entity: USER\_BADGE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | User badge id. |
| user\_id | UUID | References USER(id). |
| badge\_id | UUID | References BADGE(id). |
| earned\_at | Datetime | Time when user earned the badge. |

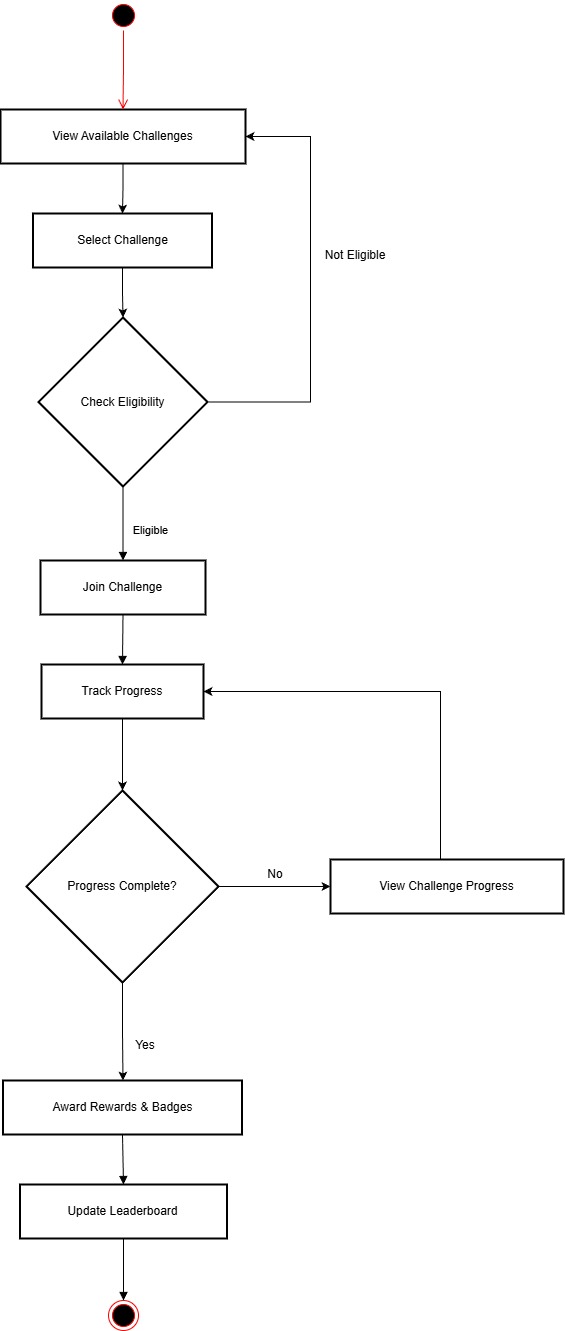
Entity: STREAK

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Streak id. |
| user\_id | UUID | References USER(id). |
| current\_streak | Integer | Current consecutive active days. |
| longest\_streak | Integer | Longest streak achieved. |
| last\_active\_date | Date | Last date user was active. |
| updated\_at | Datetime | Last update time. |

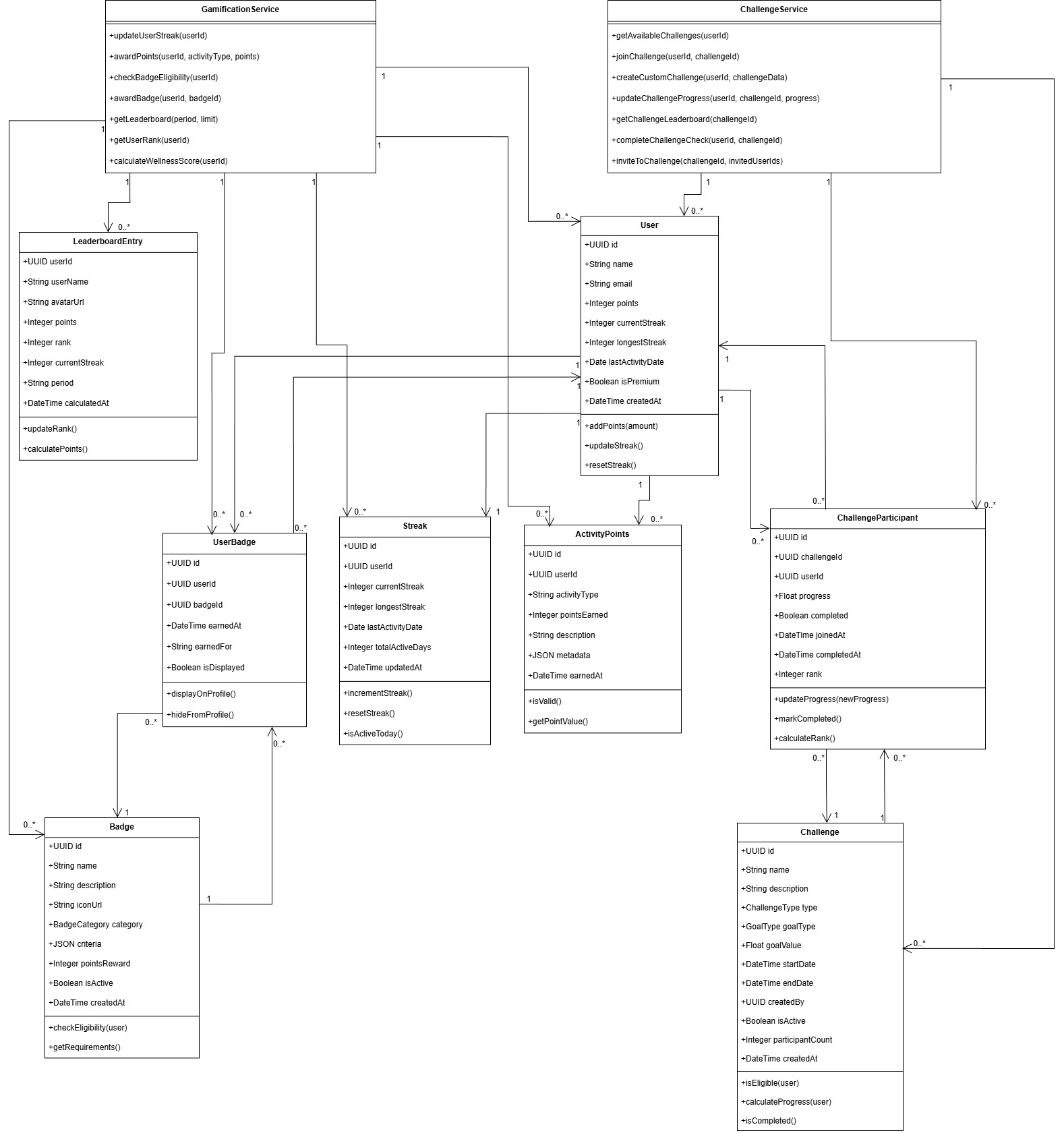
## Diagrams:



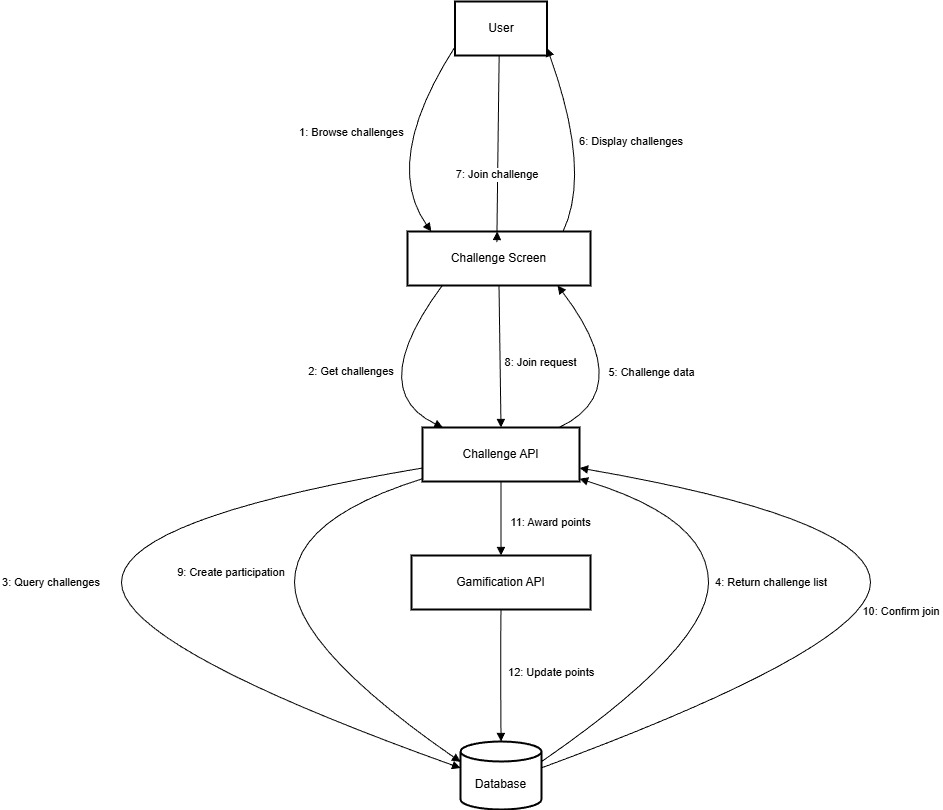
Challenge&Game 1:Use Case Diagram



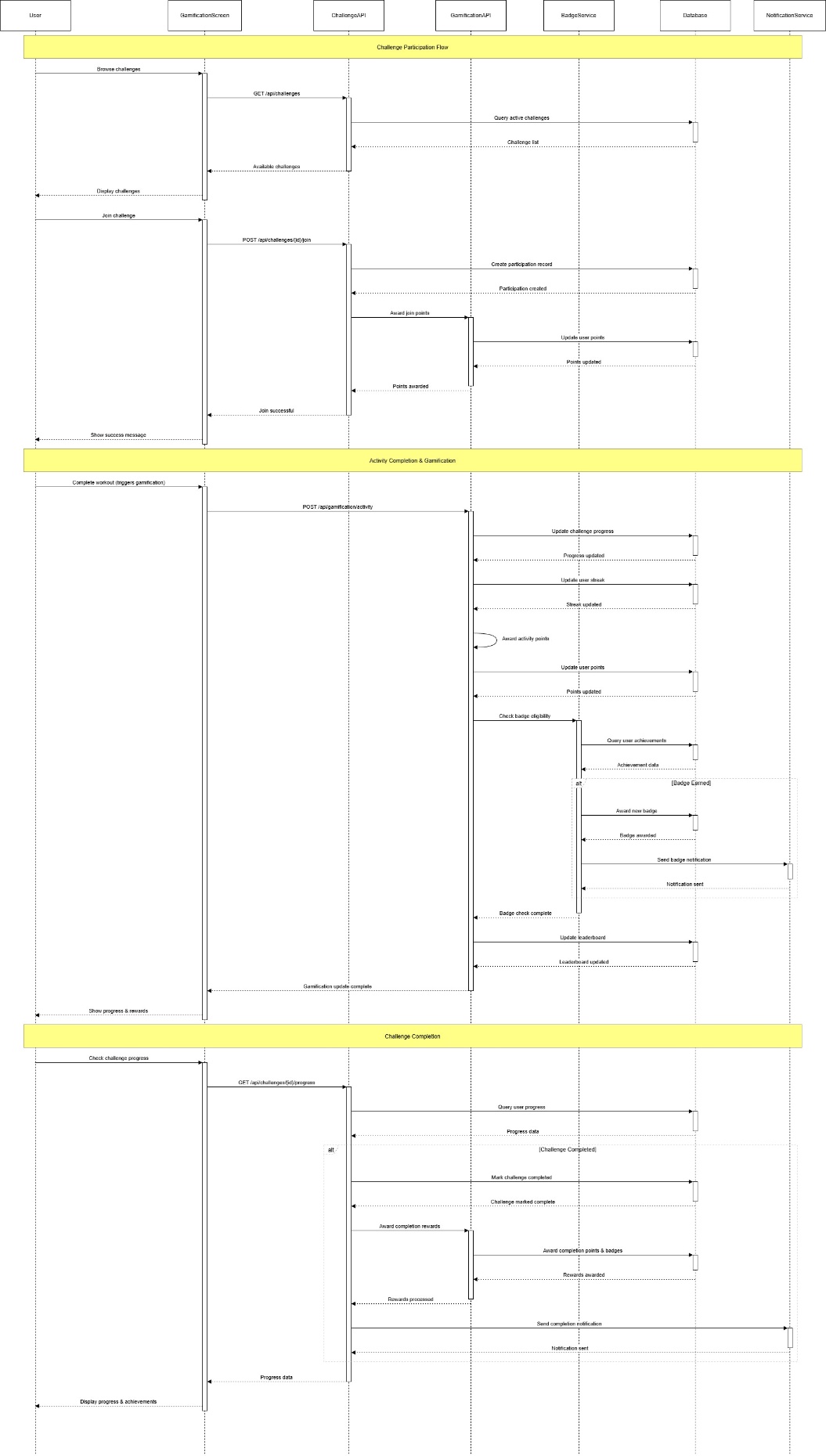
Challenge&Game 2: Activity Diagram



Challenge&Game 3: Class Diagram



Challenge&Game 4: Collaboration Diagram



Challenge&Game 5: Sequence Diagram

# Community Module

This subsystem offers a simple social space where users can post updates, like and comment on content, and follow others. It also includes reporting and moderation so that inappropriate posts can be reviewed and removed.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **CM-F-1.0** | As a user, I want to post updates with text and images so that I can share my progress. | Community Module | New post appears in feed with correct text, image and time. SRS\_NutriLift.docx​ |
| **CM-F-2.0** | As a user, I want to like and unlike posts so that I can support others. | Community Module | Like button toggles and like count updates without duplicates. SRS\_NutriLift.docx​ |
| **CM-F-3.0** | As a user, I want to comment on posts so that I can interact with the community. | Community Module | Added comment shows under the post with author and time. SRS\_NutriLift.docx​ |
| **CM-F-4.0** | As a user, I want to follow and unfollow other users so that my feed feels relevant. | Community Module | Follow button switches between Follow / Following; followed users appear more in feed. SRS\_NutriLift.docx​ |
| **CM-F-5.0** | As a user, I want to report inappropriate posts so that the feed stays safe. | Community Module | Reporting saves a record with reason; post is flagged for review. SRS\_NutriLift.docx​ |
| **CM-F-6.0** | As an admin, I want to moderate reported posts so that I can remove harmful content. | Community Module | Admin can mark a post removed; it disappears from normal feeds. SRS\_NutriLift.docx​ |

## Data Dictionary:

Entity: POST

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Post id. |
| user\_id | UUID | References USER(id). |
| content | Text | Text content of the post. |
| image\_urls | String | Comma-separated or JSON list of image URLs. |
| like\_count | Integer | Cached count of likes. |
| comment\_count | Integer | Cached count of comments. |
| is\_reported | Boolean | Flag indicating reported content. |
| is\_removed | Boolean | Flag indicating moderation removal. |
| created\_at | Datetime | Post creation time. |
| updated\_at | Datetime | Last update time. |

Entity: Comment

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Comment id. |
| post\_id | UUID | References POST(id). |
| user\_id | UUID | References USER(id). |
| content | Text | Comment text content. |
| created\_at | Datetime | Comment creation time. |

Entity: LIKE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Like id. |
| post\_id | UUID | References POST(id). |
| user\_id | UUID | References USER(id). |
| created\_at | Datetime | Time when like was created. |

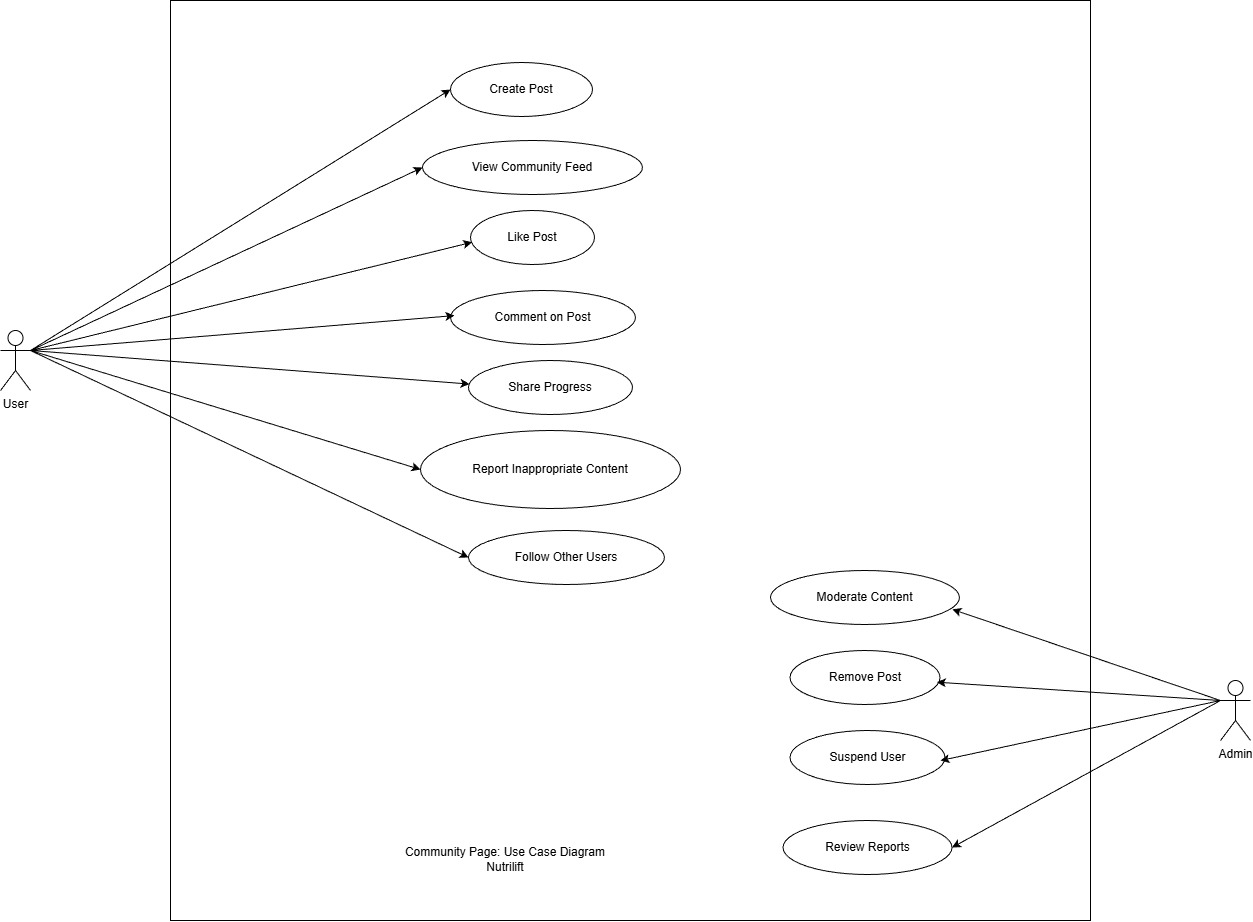
Entity: REPORT

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Report id. |
| post\_id | UUID | References POST(id). |
| reported\_by | UUID | References USER(id) who reported. |
| reason | Text | Reason for reporting the post. |
| status | String | Status such as pending, reviewed, actioned. |
| created\_at | Datetime | Report creation time. |
| reviewed\_at | Datetime | Time report was reviewed. |

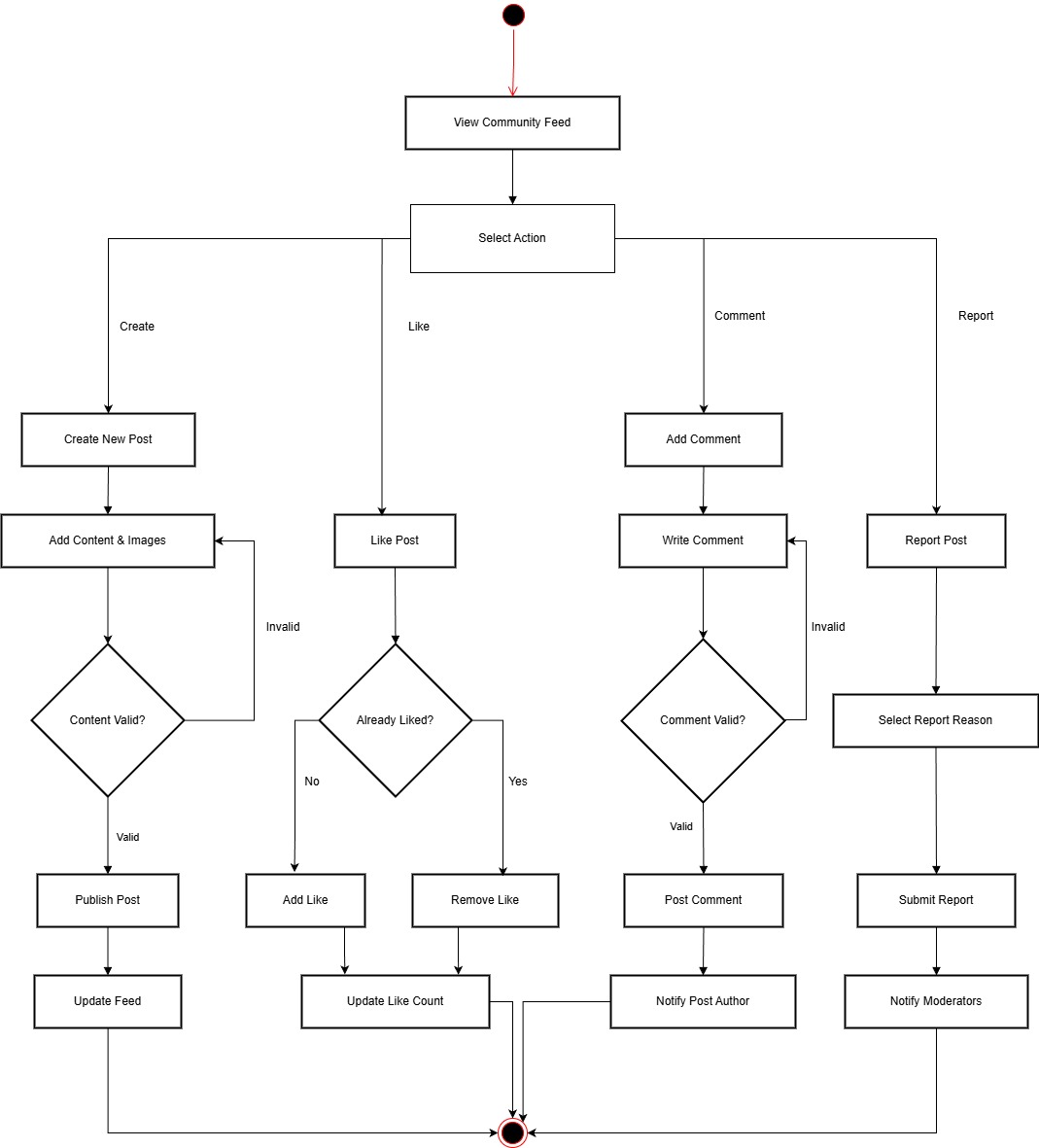
Entity: FOLLOW

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Follow relationship id. |
| follower\_id | UUID | User who follows another user. |
| following\_id | UUID | User being followed. |
| created\_at | Datetime | Time when follow started. |

## Diagrams:



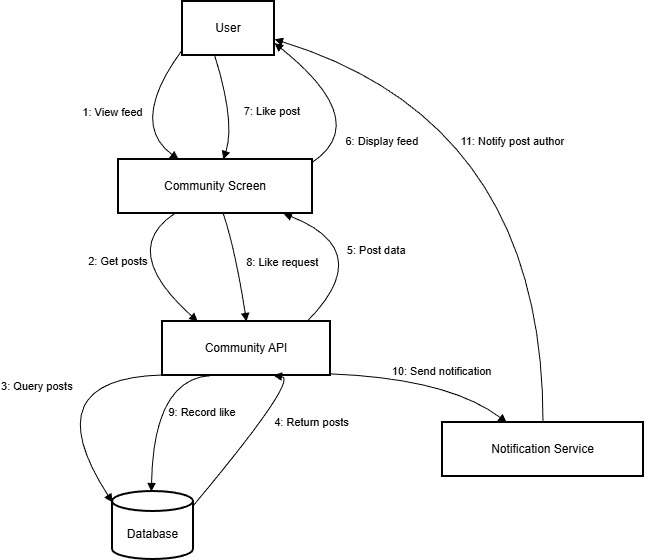
Community Page 1: Use Case Diagram



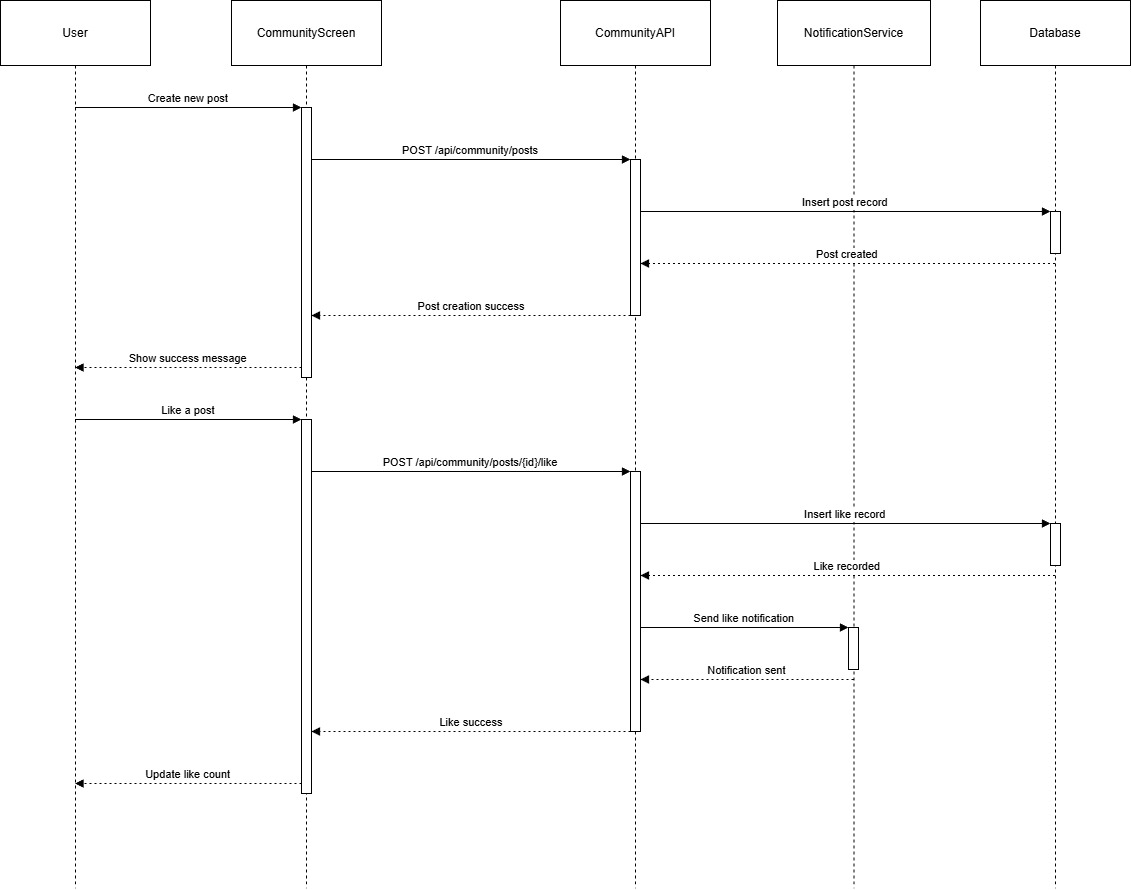
Community Page 2: Activity Diagram



Community Page 3: Class Diagram



Community Page 4: Collaboration Diagram



Community Page 5: Sequence Diagram

# Progress Reports and Analytics

This subsystem turns raw logs into weekly and period‑based summaries with charts and a wellness score. It lets users quickly see trends in their eating and training instead of interpreting numbers on their own.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **PR-F-1.0** | As a user, I want weekly graphs of calories and workouts so that I can see trends over time. | Progress Reports & Analytics | Weekly charts show daily calories and workouts matching stored logs. SRS\_NutriLift.docx​ |
| **PR-F-2.0** | As a user, I want a wellness score for a chosen period so that I get a simple summary of my habits. | Progress Reports & Analytics | Selected period shows a score and grade that change when data changes. SRS\_NutriLift.docx​ |
| **PR-F-3.0** | As a user, I want to filter reports by date range so that I can review specific periods. | Progress Reports & Analytics | Choosing start and end dates reloads summaries and charts for that range. SRS\_NutriLift.docx​ |
| **PR-F-4.0** | As a user, I want to see both nutrition and workout summaries in the same place so that I can compare them. | Progress Reports & Analytics | Report screen shows nutrition and workout cards side by side. SRS\_NutriLift.docx​ |
| **PR-NF-1.1** | Reports should load quickly for normal data sizes. | Progress Reports & Analytics | Weekly reports appear within a few seconds and UI does not freeze. SRS\_NutriLift.docx​ |

## Data Dictionary:

Entity: NUTRITION\_PROGRESS

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Nutrition progress id. |
| user\_id | UUID | References USER(id). |
| progress\_date | Date | Date for which data is summarized. |
| total\_calories | Float | Total calories consumed that day. |
| total\_protein | Float | Total protein consumed. |
| total\_carbs | Float | Total carbohydrates consumed. |
| total\_fats | Float | Total fats consumed. |
| adherence\_percentage | Float | Percentage of adherence to nutrition goals. |
| calculated\_at | Datetime | Time when summary was calculated. |

Entity: WORKOUT\_PROGRESS

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Workout progress id. |
| user\_id | UUID | References USER(id). |
| progress\_date | Date | Date for which workout is summarized. |
| workout\_count | Integer | Number of workouts performed. |
| total\_duration | Integer | Total workout duration in minutes. |
| calories\_burned | Float | Total calories burned. |
| exercise\_count | Integer | Number of exercises performed. |
| consistency\_score | Float | Consistency metric for workouts. |
| calculated\_at | Datetime | Time when summary was calculated. |

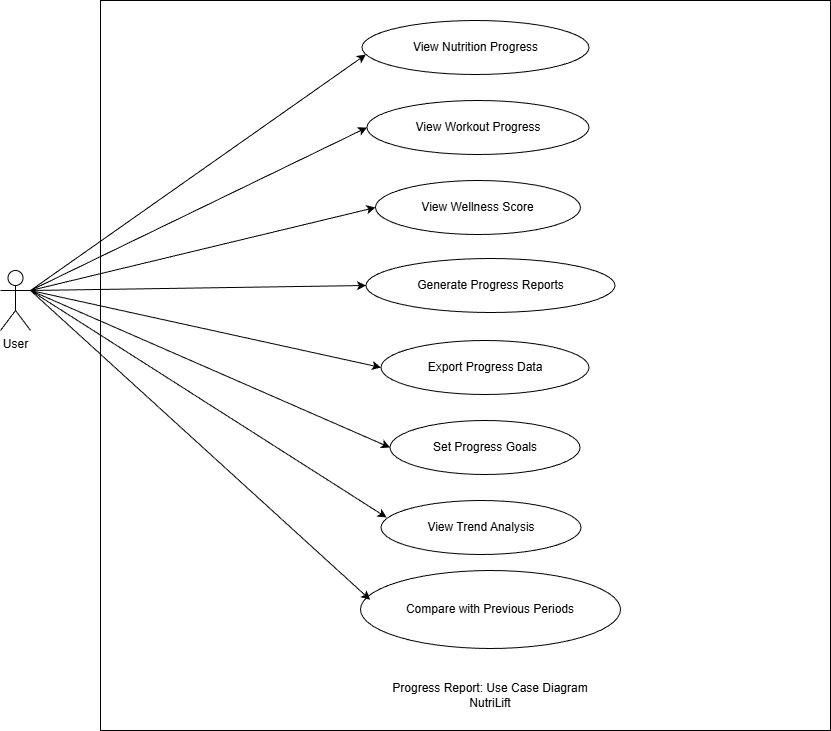
Entity: WELLNESS\_SCORE

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Wellness score id. |
| user\_id | UUID | References USER(id). |
| score\_date | Date | Date of wellness score. |
| nutrition\_score | Float | Nutrition component of wellness. |
| fitness\_score | Float | Fitness component of wellness. |
| consistency\_score | Float | Consistency component of wellness. |
| overall\_score | Float | Combined overall score. |
| score\_grade | String | Grade such as A, B and so on. |
| calculated\_at | Datetime | Time when score was calculated. |

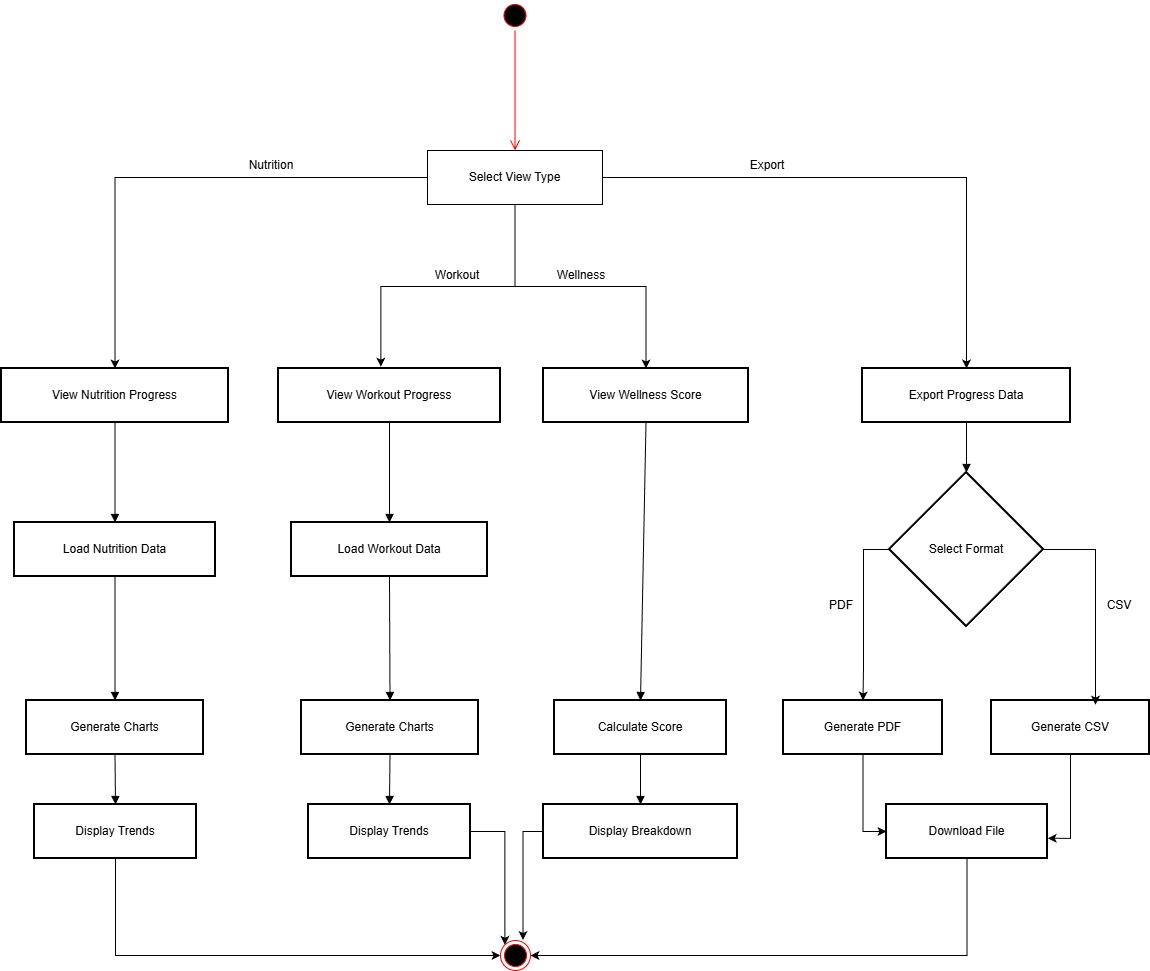
Entity: PROGRESS\_REPORT

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Progress report id. |
| user\_id | UUID | References USER(id). |
| report\_type | String | Type such as weekly, monthly, custom. |
| start\_date | Date | Start date of report period. |
| end\_date | Date | End date of report period. |
| report\_data | JSON | Aggregated report data. |
| format | String | Export format such as pdf, csv, json. |
| generated\_at | Datetime | Time when report was generated. |

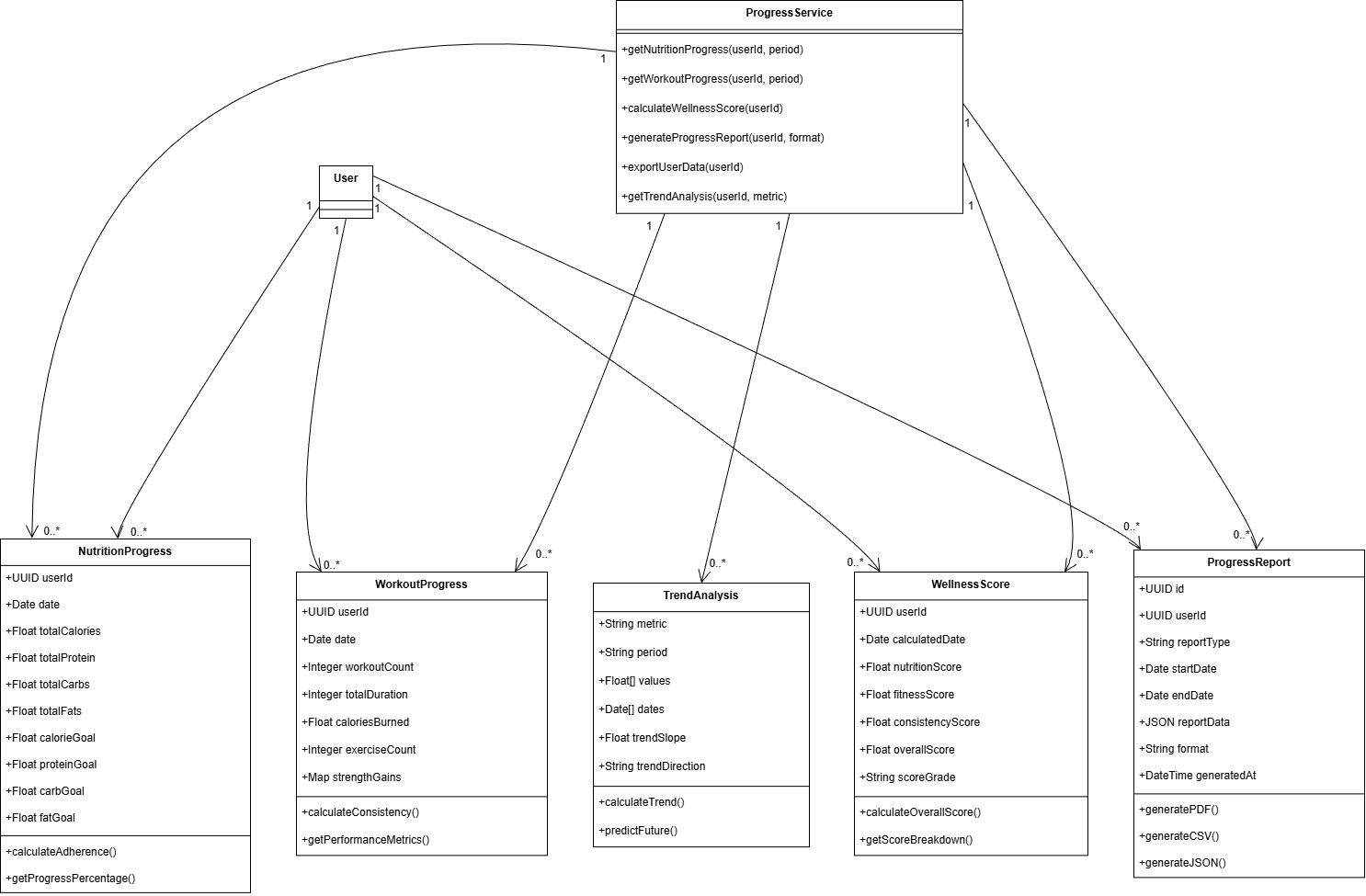
## Wireframe:Diagrams:



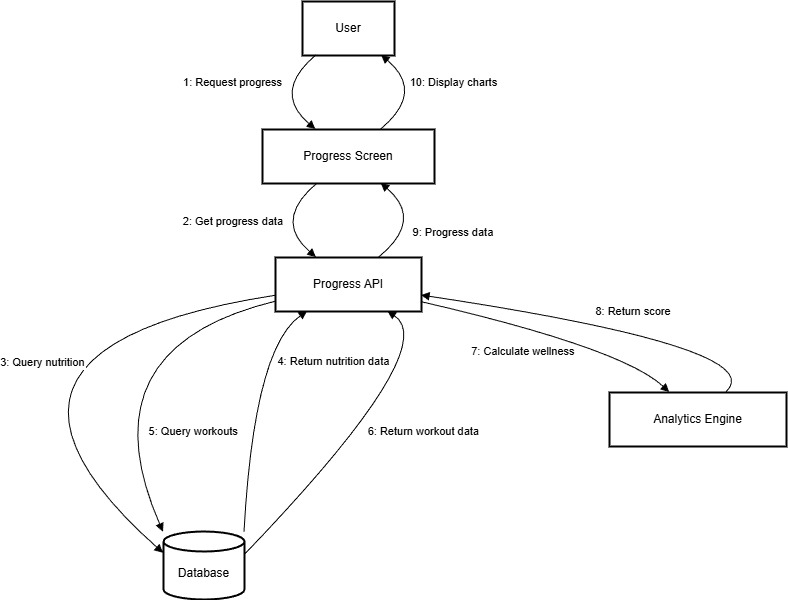
Progress Report 1: Use Case Diagram



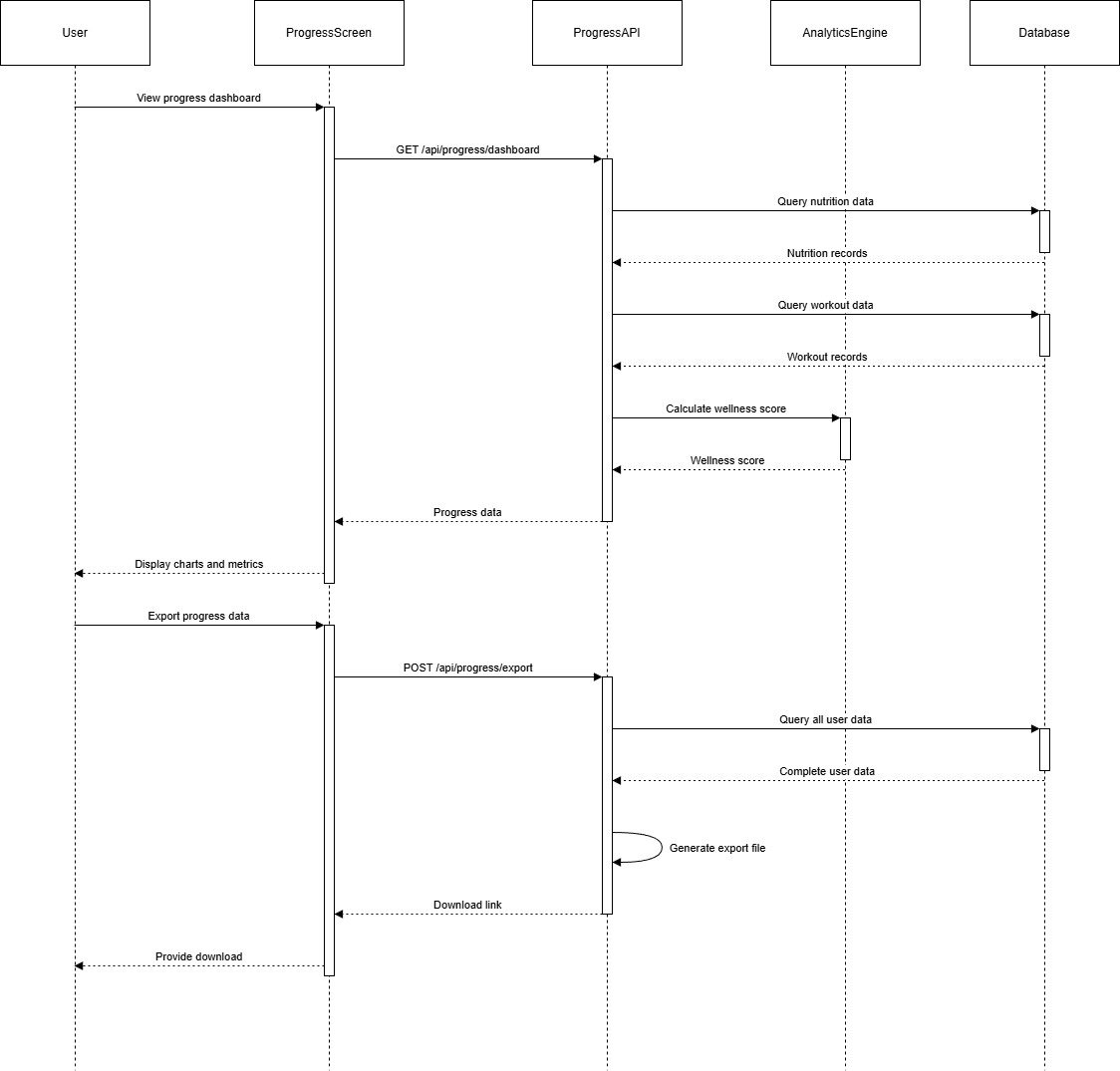
Progress Report 2: Activity Diagram



Progress Report 3: Class Diagram



Progress Report 4: Collaboration Diagram



Progress Report 5: Sequence Diagram

# Payment Integration

This subsystem manages premium plans and payments through a secure gateway. It keeps track of transactions and subscription status so the app knows when to unlock or remove premium features.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **PY-F-1.0** | As a user, I want to see available premium plans so that I can choose one that suits me. | Payment Integration | Premium screen lists plans with name, duration and price. SRS\_NutriLift.docx​ |
| **PY-F-2.0** | As a user, I want to pay securely for premium so that my card details are safe. | Payment Integration | Payment uses gateway flow; app never stores raw card data; success activates premium. SRS\_NutriLift.docx​ |
| **PY-F-3.0** | As a user, I want the app to remember my premium status so that I do not pay again unnecessarily. | Payment Integration | After restart or re‑login, premium features stay unlocked while active. SRS\_NutriLift.docx​ |
| **PY-F-4.0** | As a user, I want to view my payment history so that I can keep track of charges. | Payment Integration | History lists transactions with date, plan, amount and status. SRS\_NutriLift.docx​ |
| **PY-NF-1.1** | Payment processing must be secure and consistent with the gateway. | Payment Integration | All payment calls use HTTPS and handle success/failure without duplicate charges. SRS\_NutriLift.docx​[ppl-ai-file-upload.s3.amazonaws](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/68869185/e07b1379-2525-457b-92bc-53ff251353fc/SRSSample_1bae6977-7664-459f-af46-5f53c26c0877_90184.pdf)​ |

## Data Dictionary:

Entity: SUBSCRIPTION\_PLAN

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| plan\_id | String | Subscription plan code. |
| name | String | Plan name. |
| description | Text | Description of the plan. |
| monthly\_price | Float | Monthly subscription price. |
| yearly\_price | Float | Yearly subscription price. |
| features | JSON | Included features for this plan. |
| is\_active | Boolean | Indicates if plan is active. |
| created\_at | Datetime | Creation time. |

Entity: SUBSCRIPTION

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Subscription id. |
| user\_id | UUID | References USER(id). |
| plan\_id | String | References SUBSCRIPTION\_PLAN(plan\_id). |
| subscription\_type | String | app\_premium or gym\_membership. |
| gym\_id | UUID | References GYM(id) for gym membership. |
| payment\_gateway | String | Payment gateway name such as stripe or khalti. |
| transaction\_id | String | Last payment gateway transaction id. |
| amount | Float | Recurring subscription amount. |
| currency | String | Currency code. |
| status | String | Status such as active, cancelled, expired. |
| starts\_at | Datetime | Subscription start time. |
| expires\_at | Datetime | Subscription expiry time. |
| created\_at | Datetime | Creation time. |

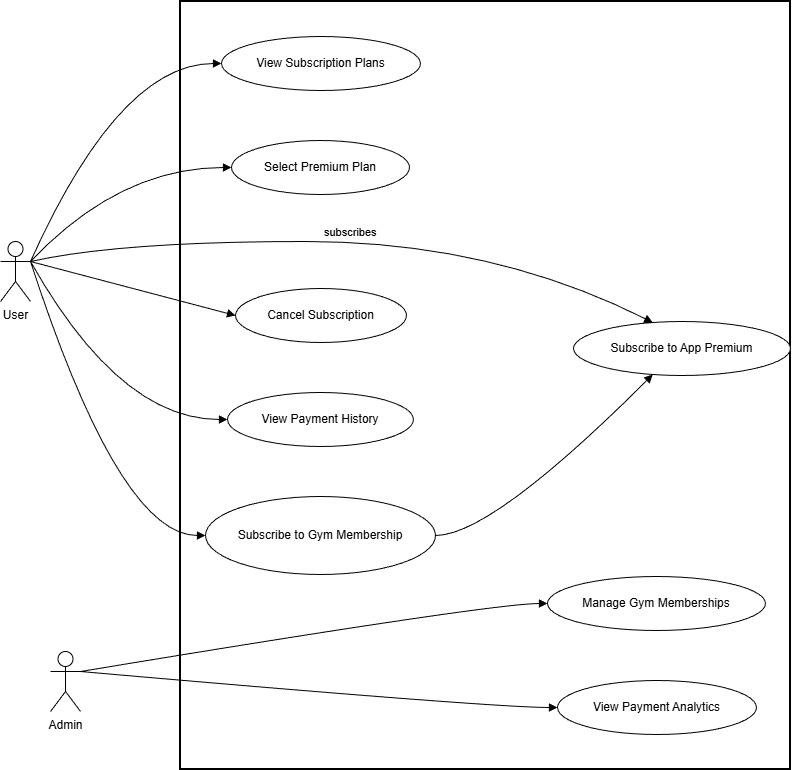
Entity: PAYMENT\_TRANSACTION

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Payment transaction id. |
| subscription\_id | UUID | References SUBSCRIPTION(id). |
| gateway\_transaction\_id | String | Transaction id from payment gateway. |
| amount | Float | Amount charged. |
| currency | String | Currency of transaction. |
| status | String | pending, success or failed. |
| failure\_reason | Text | Explanation if transaction failed. |
| processed\_at | Datetime | Time when transaction was processed. |
| created\_at | Datetime | Creation time. |

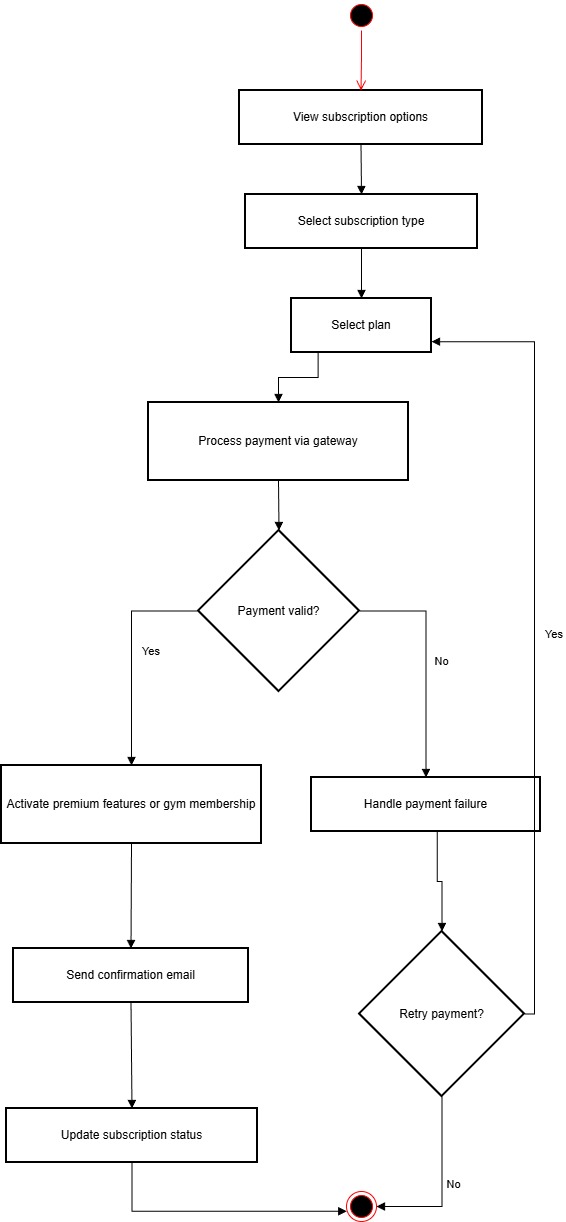
Entity: PAYMENT\_WEBHOOK

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Webhook event id. |
| gateway | String | Source payment gateway. |
| event\_type | String | Webhook event type. |
| payload | JSON | Raw event payload content. |
| processed | Boolean | Indicates whether event was handled. |
| received\_at | Datetime | Time when webhook was received. |

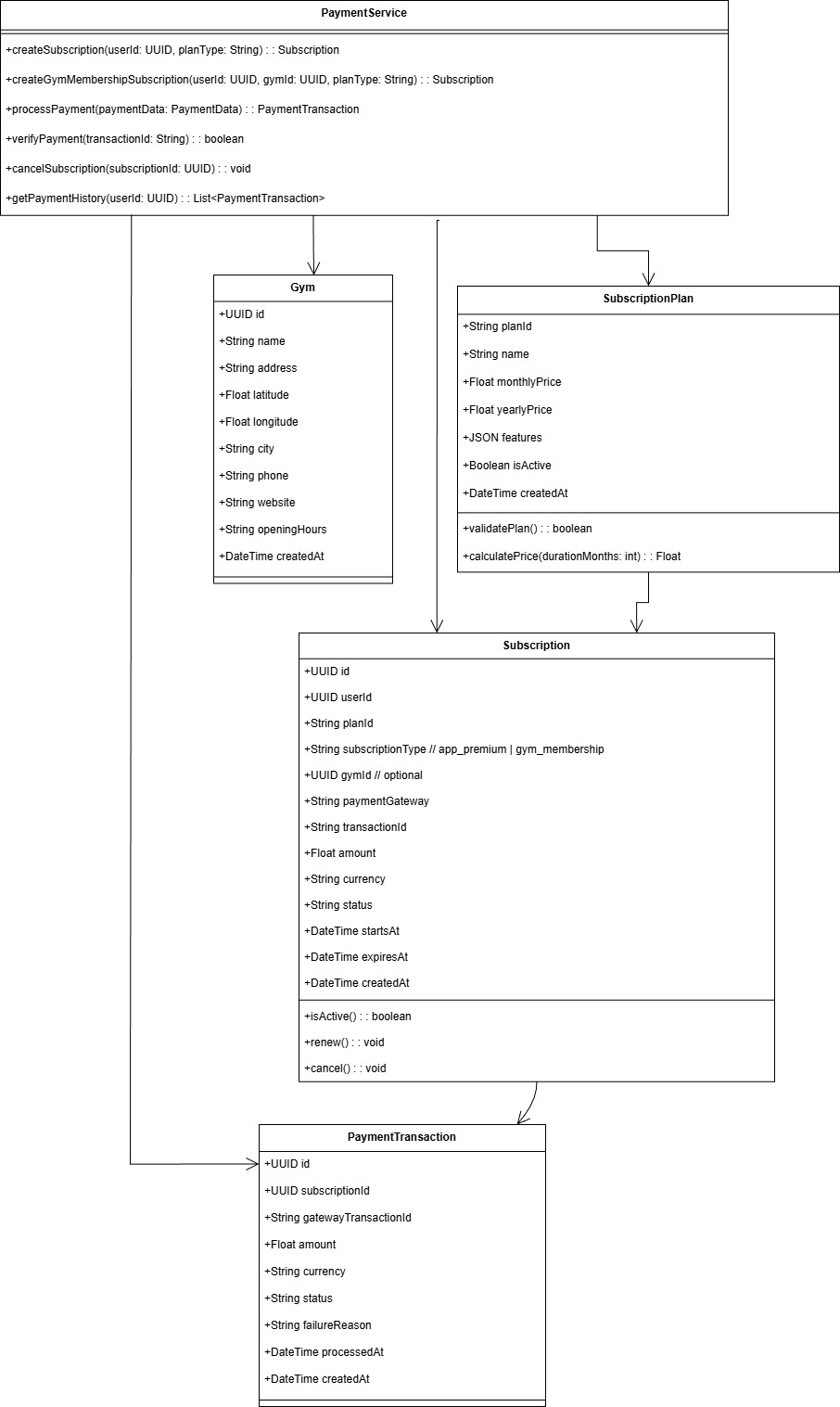
## Diagrams:



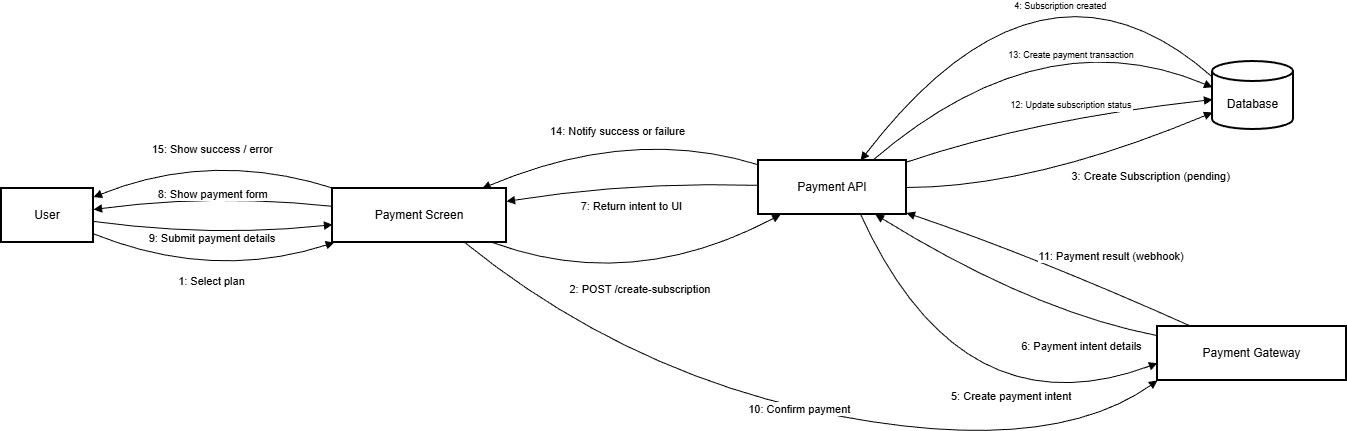
Payment Integration 1: Use Case Diagram



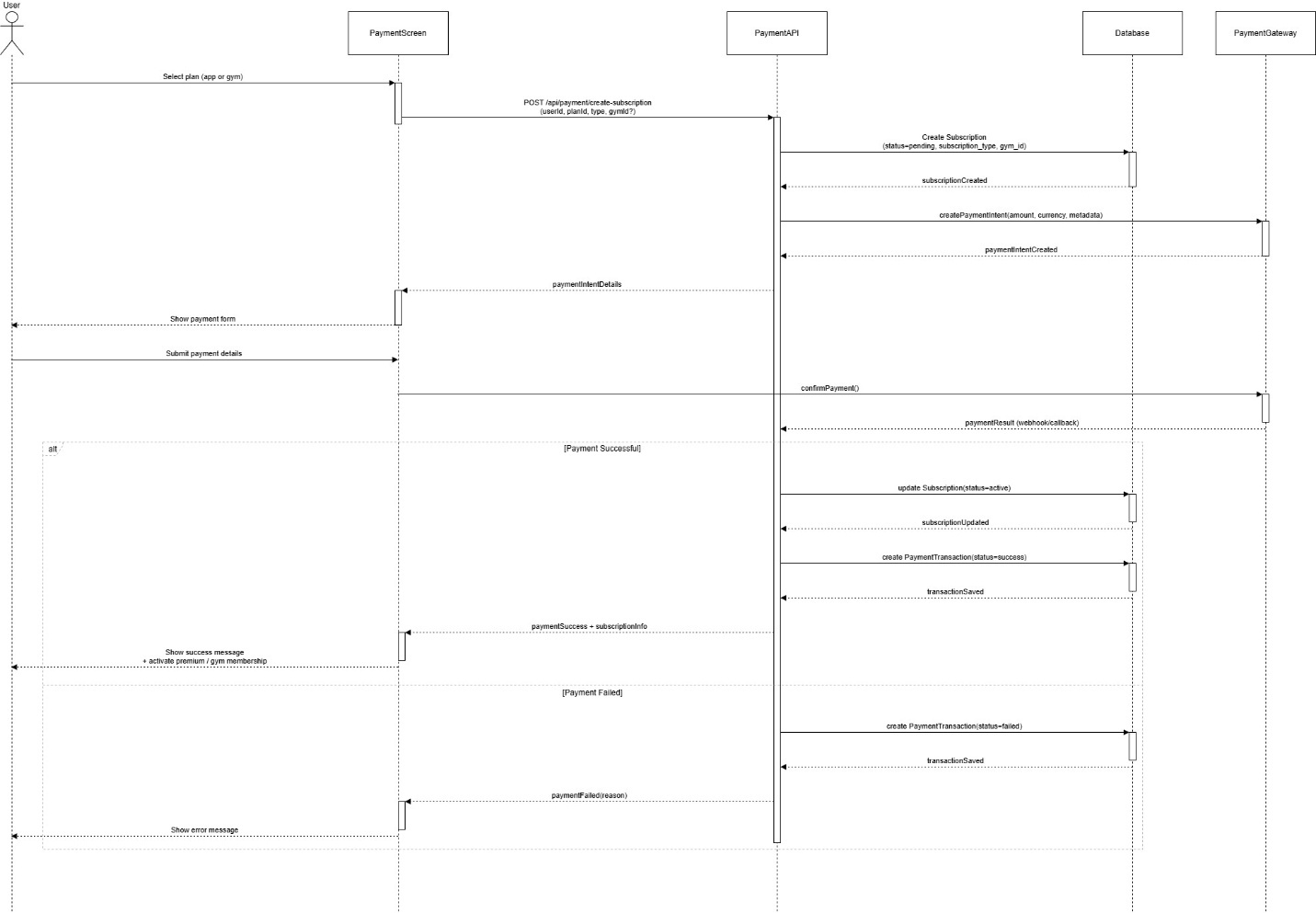
Payment Integration 2: Activity Diagram



Payment Integration 3: Class Diagram



Payment Integration 4: Collaboration Diagram



Payment Integration 5: Sequence Diagram

# Gym Discovery and Membership

This subsystem helps users find gyms, check their details and join partner gyms from within the app. It also stores active and past memberships so users can see where they are currently enrolled.

## SRS:

| **ID** | **User Story / Requirement** | **Subsystem** | **Acceptance Criteria** |
| --- | --- | --- | --- |
| **GYM-F-1.0** | As a user, I want to find gyms near me so that I can pick a convenient one. | Gym Discovery & Membership | With location allowed, list shows nearby gyms sorted by distance. SRS\_NutriLift.docx​ |
| **GYM-F-2.0** | As a user, I want to search gyms by city or name so that I can explore options. | Gym Discovery & Membership | Search bar filters gyms by typed city or name. SRS\_NutriLift.docx​ |
| **GYM-F-3.0** | As a user, I want to view detailed information about a gym so that I can decide if it fits my needs. | Gym Discovery & Membership | Detail screen shows address, hours, contact, website and memberships. SRS\_NutriLift.docx​ |
| **GYM-F-4.0** | As a user, I want to enroll in a partner gym from the app so that my membership is digital and linked to NutriLift. | Gym Discovery & Membership | After payment, a membership record is created and shown as active. SRS\_NutriLift.docx​ |
| **GYM-F-5.0** | As a user, I want to see my active and past gym memberships so that I can manage them. | Gym Discovery & Membership | Membership list shows all memberships with status (active, cancelled, expired). SRS\_NutriLift.docx​ |

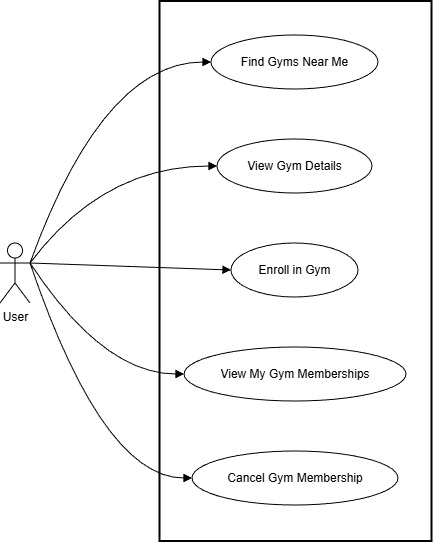
## Data Dictionary:

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Gym id. |
| name | String | Gym name. |
| address | String | Street address of the gym. |
| latitude | Float | GPS latitude. |
| longitude | Float | GPS longitude. |
| city | String | City of the gym. |
| phone | String | Contact phone number. |
| website | String | Website URL of the gym. |
| opening\_hours | String | Opening hours information. |
| created\_at | Datetime | Creation time. |

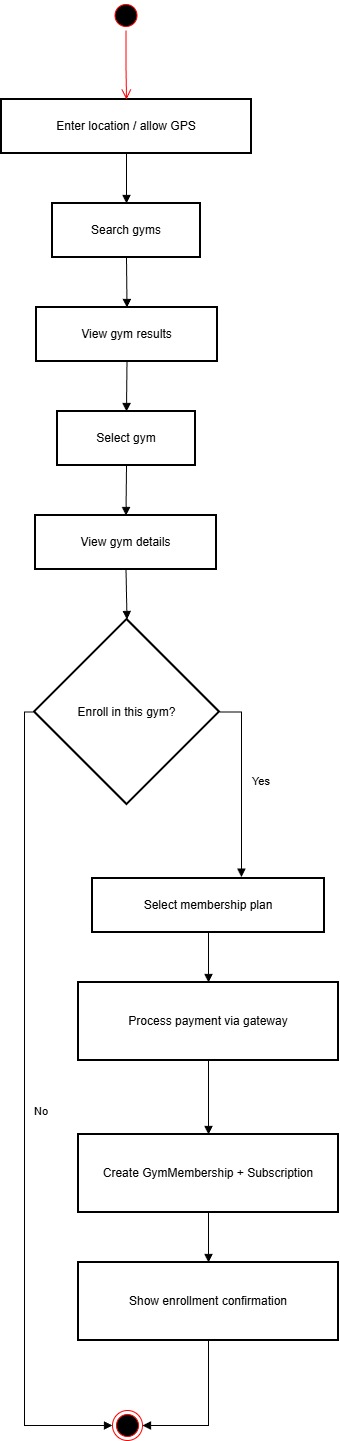
Entity: GYM\_MEMBERSHIP

| **Attribute Name** | **Data Type** | **Description** |
| --- | --- | --- |
| id | UUID | Gym membership id. |
| user\_id | UUID | References USER(id). |
| gym\_id | UUID | References GYM(id). |
| membership\_type | String | Type such as monthly, yearly, day\_pass. |
| start\_date | Date | Membership start date. |
| end\_date | Date | Membership end date. |
| status | String | active, cancelled or expired. |
| created\_at | Datetime | Creation time. |

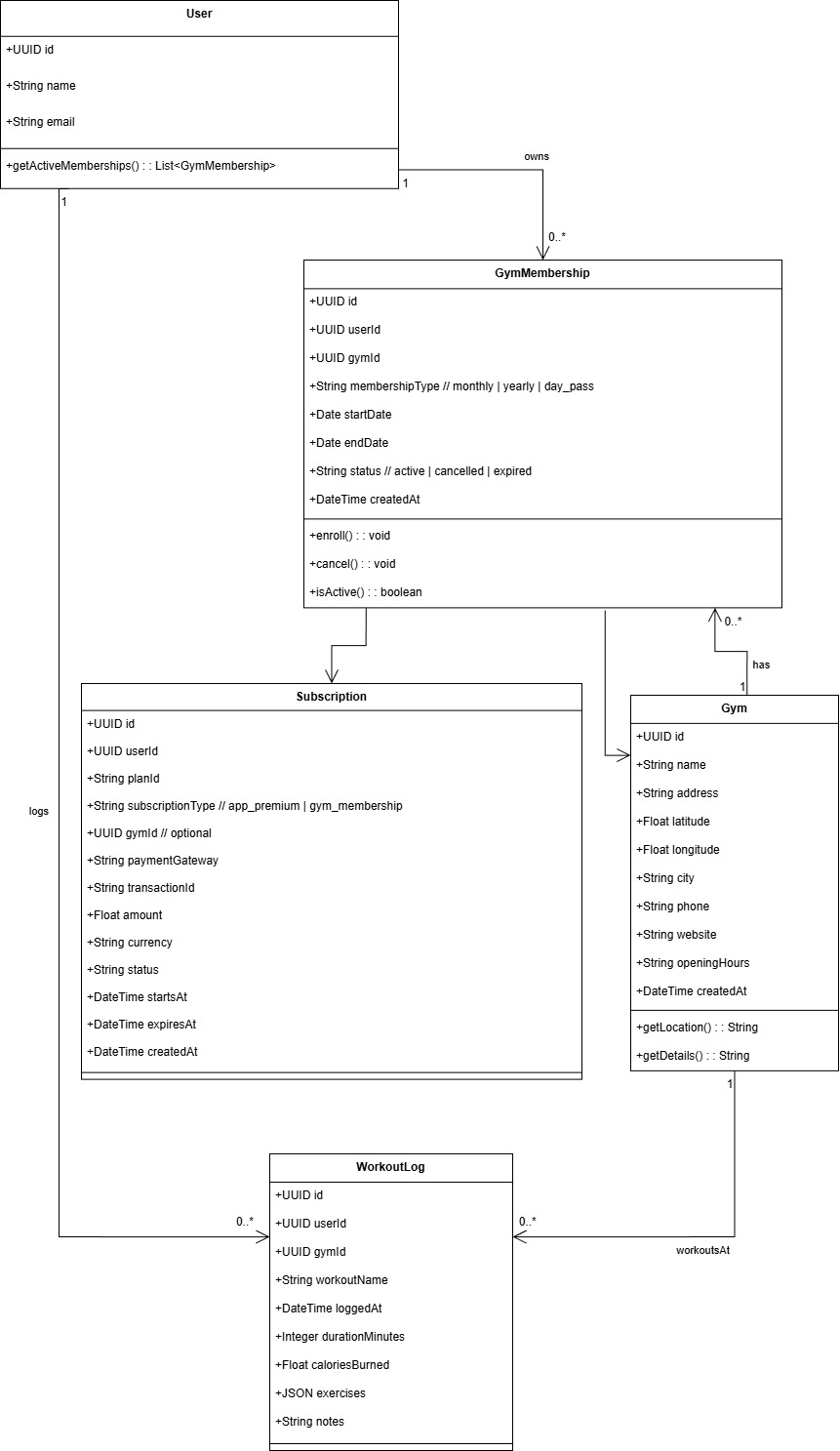
## Diagrams:



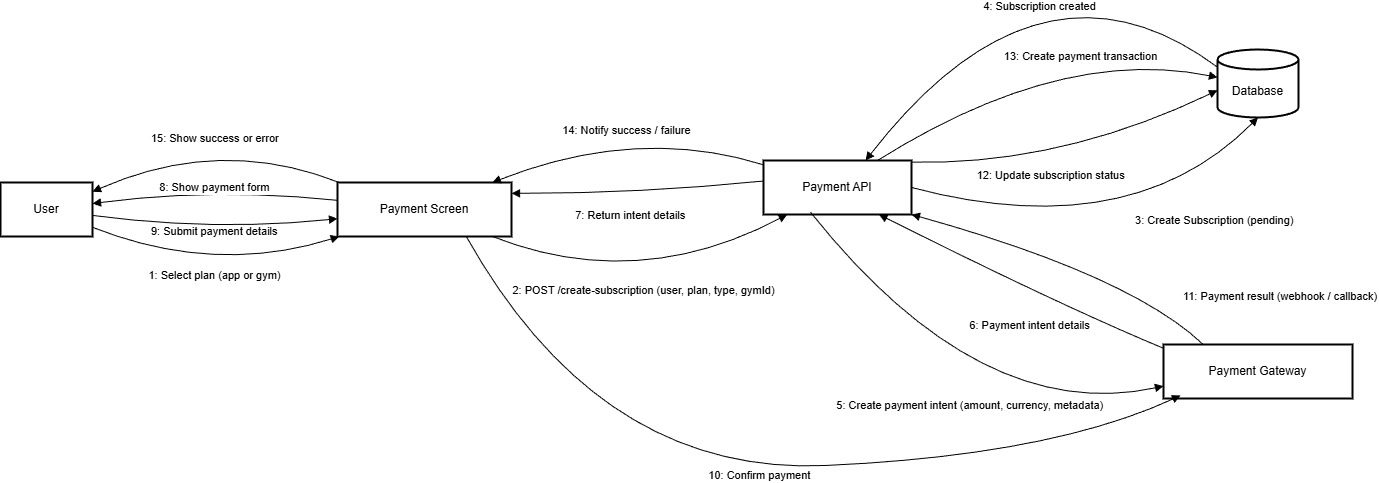
Gym Discovery 1: Use Case Diagram



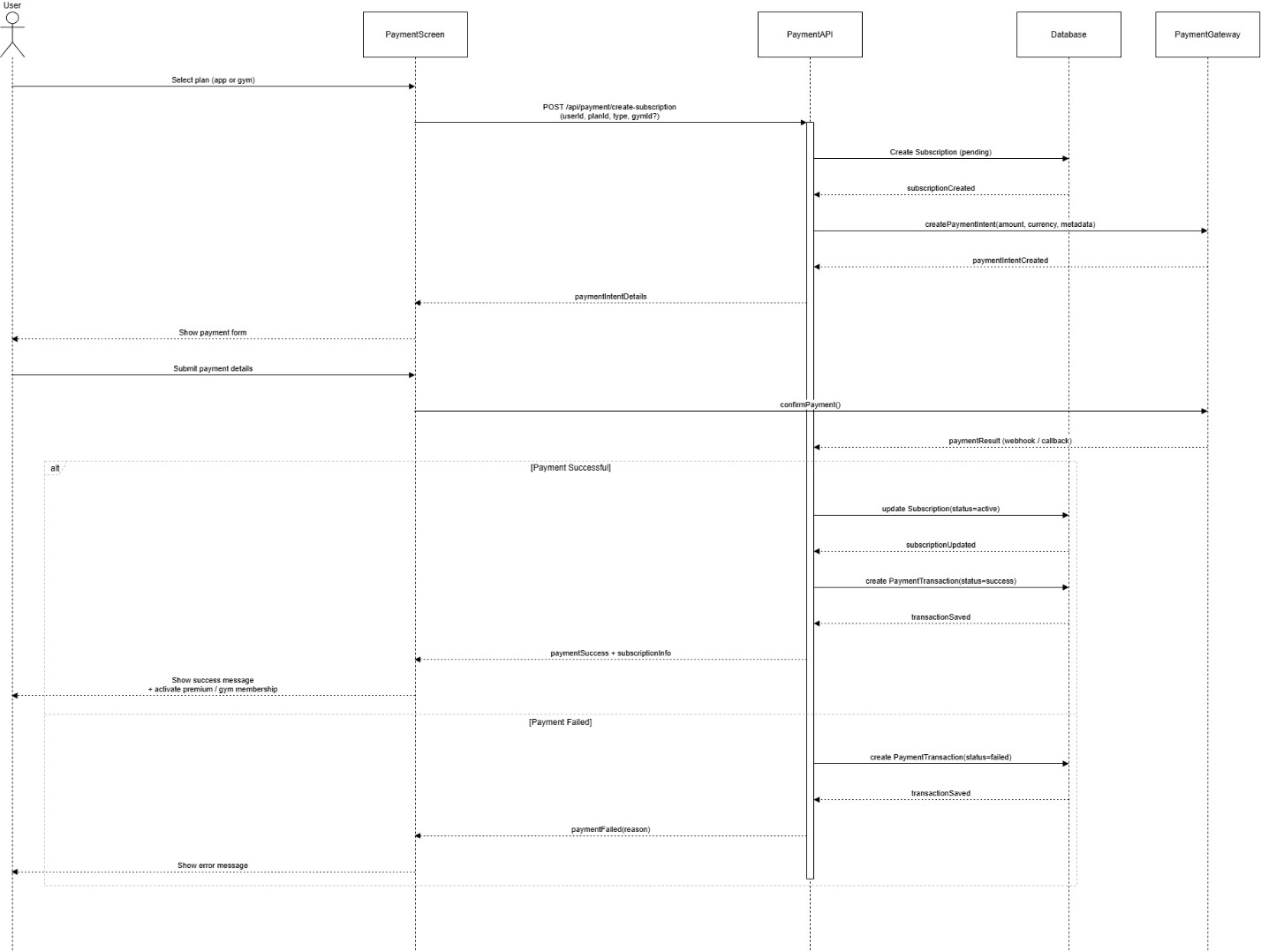
Gym Discovery 2: Activity Diagram



Gym Discovery 3: Class Diagram

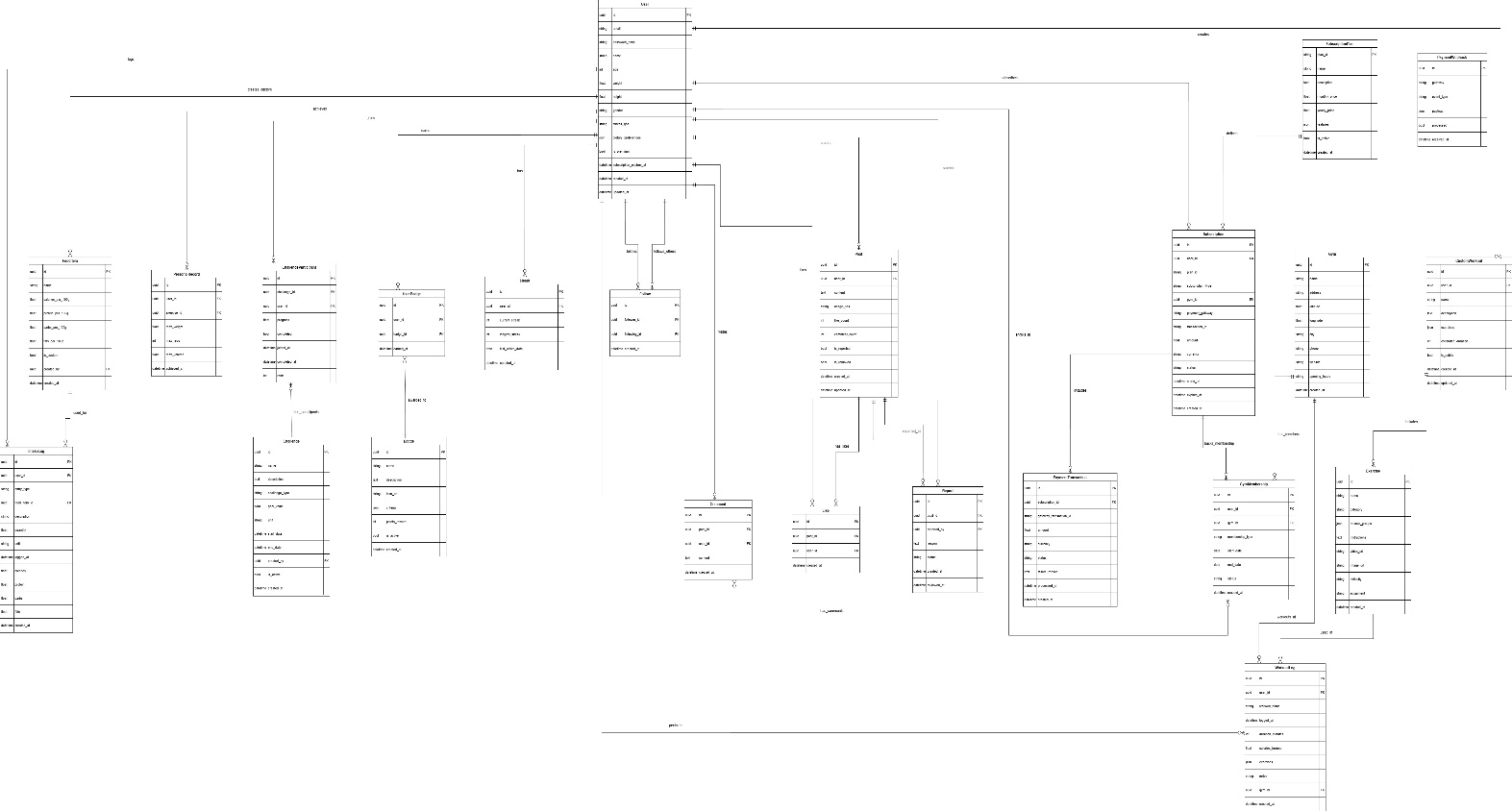


Gym Discovery 4: Collaboration Diagram



Gym Discovery 5: Sequence Diagram

# ERD Diagram of whole system:



ERD Diagram 1

# Test Plan:

This section outlines how the main features of NutriLift will be tested once the implementation is in place. The focus is on manual functional testing of core user journeys on an Android device connected to a test backend. The table below lists representative test cases for each subsystem that will be used as a starting point during the development and integration phase.

## Summary of Test Plan

The given table summarises the main test cases that will be applied across all subsystems. Detailed steps for each case are documented in the testing spreadsheet and can be expanded during implementation.

| **TC ID** | **Subsystem** | **Test Description** | **Preconditions** | **Main Steps** | **Expected Result** |
| --- | --- | --- | --- | --- | --- |
| TC‑UM‑01 | User Management | Create a new account using email and password. | App installed; backend running. | Open register screen, enter valid details, submit. | New user record is created and user is taken to the main dashboard. |
| TC‑UM‑02 | User Management | Prevent a second account with the same email. | One account already exists for that email. | Try to register again with same email. | Registration is rejected and an appropriate error message is shown. |
| TC‑UM‑03 | User Management | Log in and update profile data. | Existing user account. | Log in, open profile, change weight and goal, save. | Login is successful and new profile values are stored and displayed. |
| TC‑NT‑01 | Nutrition Tracking | Log a meal using a food from the database. | Logged‑in user; at least one food item available. | Add a meal with selected food and quantity. | Intake entry is stored and daily calorie and macro totals increase correctly. |
| TC‑NT‑02 | Nutrition Tracking | Add a custom food and use it in a meal log. | Logged‑in user. | Create custom food, then log a meal with it. | Custom food appears in search and behaves like other items when logged. |
| TC‑NT‑03 | Nutrition Tracking | Record daily water intake. | Logged‑in user. | Add one or more water entries. | Hydration total increases and progress towards the daily target is updated. |
| TC‑WT‑01 | Workout Tracking | Log a workout containing multiple exercises. | Logged‑in user; exercises present. | Add two exercises with sets and reps, save workout. | Workout is stored in history with all details. |
| TC‑WT‑02 | Workout Tracking | Use a custom workout template. | Logged‑in user. | Create a template and start a workout from it. | Template can be reused and generates a correct workout entry. |
| TC‑RC‑01 | Rep Count | Count repetitions for a supported exercise. | Camera permission granted; good lighting. | Start a rep‑count session, perform around ten clear reps. | Counter tracks repetitions with small error and saves the session. |
| TC‑CG‑01 | Challenges & Gamification | Join a challenge and see progress update. | Active challenge exists. | Join challenge, then log relevant workouts/meals. | User appears as participant and progress bar increases. |
| TC‑CM‑01 | Community | Create a post and interact with it. | Logged‑in user; network available. | Publish a text+image post, like it and add a comment. | Post is visible in feed; like count and comments behave as expected. |
| TC‑PR‑01 | Reports & Analytics | View a weekly summary and wellness score. | User has data for several days. | Open reports for the last week. | Charts and wellness score reflect the underlying logs. |
| TC‑PY‑01 | Payment Integration | Complete a premium purchase in sandbox mode. | Logged‑in user; gateway sandbox configured. | Select a plan, follow payment flow, return to app. | Subscription is marked active and premium features unlock. |
| TC‑GYM‑01 | Gym Discovery & Membership | Find nearby gyms and enroll in one. | Location permission granted; gyms with coordinates exist. | Use “find gyms near me”, open a gym, choose membership, pay in sandbox. | List shows nearby gyms; after payment a membership record is created and visible in the membership screen. |