Food Journal Sprint Plan

Sprint 1—2.11-3.4 (21 days)**:**

1. sign up/sign in, providing personal information & health condition & set fitness goal
    - read registration forms from UI, instantiate classes then pass the information to different databases such as (Account, Health information)--User Profile, (Food History)--Local Activity History, (Exercise History)--Local Food History;

2. view/edit personal profile
    - read requested data from database, offer edit function to update the information; if edit is confirmed in UI, then update the database

3. record/view daily calories consumption
    - audio/manually input the daily food and water intake, adding them to the Local Food History
4. record/view daily exercise
    - read daily activity consumption from android activity tracker API, and add it to the database

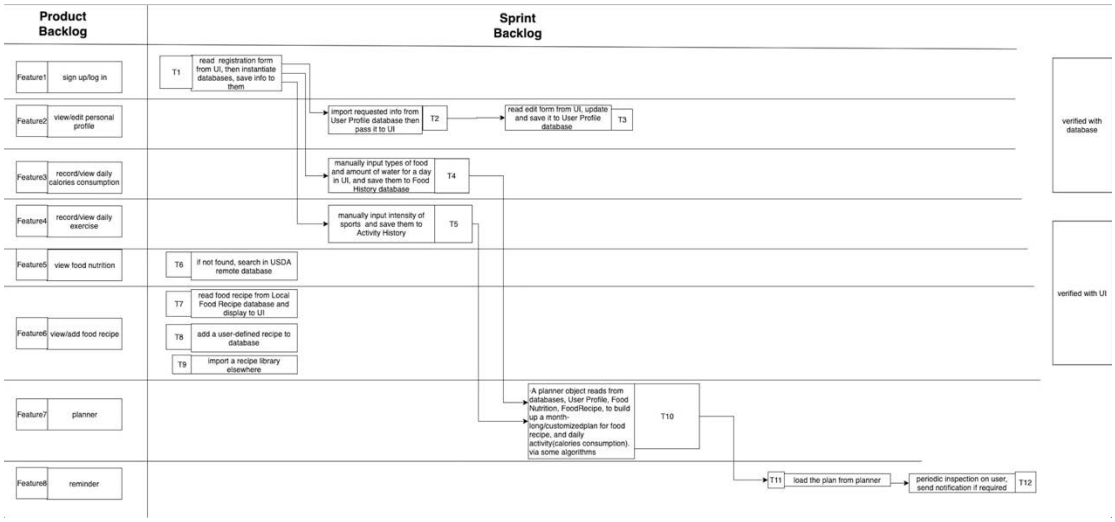
Sprint 2—3.25-4.15 (21 days)**:**

5. search & tag food
    - type food name in the search bar, retrieve nutrition information from local database first; if not found, reads from remote database USDA.

6. view/add local food recipe
    - read food recipe from Local Food Recipe database and display to UI. (optional) import other food recipe to the Local Food Recipe database

7. Planner
    - A planner object reads from databases, User Profile, Food Nutrition, Food Recipe, to build up a month-long/customized plan for food recipe, and daily activity(calories consumption).
8. reminder
    - a reminder object in the planner class, saves the plan set up by planner, and execute a periodic inspect on daily food consumption & activity consumption, in guarantee that the user's daily consumption is on the right track; otherwise, give a notification.

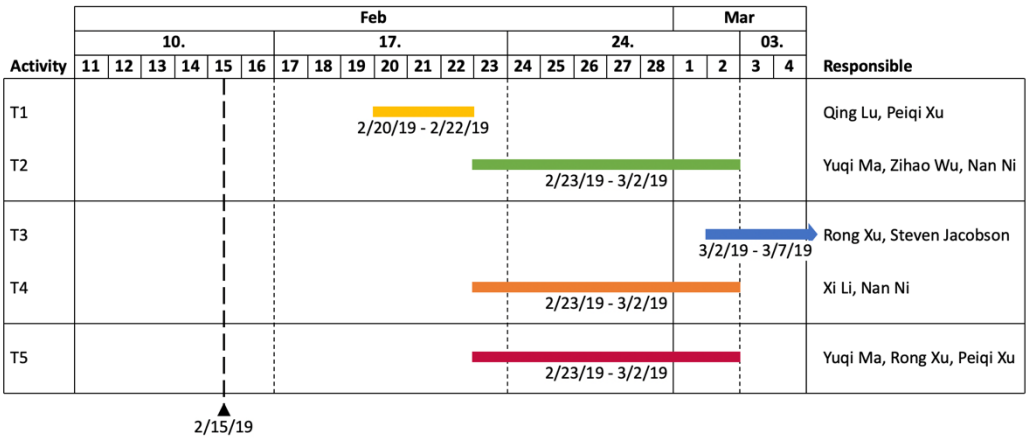
Product and Sprint Backlog:



Task plan:

Task	Effort/days	Development Duration/days	Test Duration/days	Dependency
T1	28	3.5	1	
T2	24	8	2	T1
T3	32	4	1	T2
T4	24	8	2	T1
T5	24	8	2	T1
T6	N/A	N/A	N/A	
T7	N/A	N/A	N/A	T6
T8	N/A	N/A	N/A	
T9	N/A	N/A	N/A	
T10	N/A	N/A	N/A	T4, T5
T11	N/A	N/A	N/A	T10
T12	N/A	N/A	N/A	T11

Gantt chart:



Risk Management:

Risk	Description
Specification changes	The functionality requirement or product specification is modified by stakeholders
Tools/builds underperformance	Tools that supports the project do not perform as anticipated
Underestimated time to develop software	Inexperienced plan on development of the system, may result in underestimation of progress of components
Unexpected difficulties to combine work from each team	A combination of the development teams implementing their functions in a wide range of frameworks with the nature of the project (very large, numerous components), may result in small errors for the whole system.
Team member daily schedule	Every member has a busy daily routine (class, interviews, etc.) sometimes making it difficult to efficiently communicate and track progress.
Member illness	Flu, unstable weather change, epidemic, etc.
Financial problems	APIs, which support the project, may have licenses that require additional budgets, or that do not allow commercial products, etc.
Underestimated time to test & repair components and system	Test cases identified are not broad enough to capture each corner case. Extra test cases may emerge, once every component comes together.

Risk	Probability	Effects
Specification delays	Low	Serious
CASE tool underperformance	Low	Catastrophic
Underestimated time to develop software	High	Serious
Unexpected difficulties to combine work from each team	High	Catastrophic
Team member daily schedule	Moderate	Serious
Member illness	Moderate	Tolerable
Financial problems	Low	Tolerable
Underestimated time to test & repair components and system	High	Serious

Risk	Strategy
Specification delays	Say “no” to client
CASE tool underperformance	Before the project, research is done to establish the correct tools
Underestimated time to develop software	Turn to some experienced or software-proficient people for advice
Unexpected difficulties to combine work from each team	To ensure the efficiency of work combination, every team member is supposed to provide clear and intuitive interface/abstraction of their components
Team member daily schedule	Leader has responsibility to check every team’s progress, and Scrum master will organize meetings every week.
Member illness	Vaccine, exercises
Financial problems	Choose free and easy-to-use APIs
Underestimated time to test & repair components and system	Ensure testing is implemented at a regular interval. Additionally, test with components developed by other teams as often as possible