CS 5500 Sprint 1 Proposal

Team Eureka: Yi Hu, Yitian Huang, Ziwei Wang

Github: https://github.com/users/zwwang98/projects/2

Project Description

This project is about developing a fitness application that allows users to keep track of their daily activities, performance ranking and receive personal recommendations from this application. These recommendations are based on users' own activity patterns, favorite activities which were recorded when they registered their profiles, activities that people who are nearby currently do and the differences between their current BMI and target.

Project Scope

The scope of this project is based on the past data provided by our professor.
What is included in this project:

- Analyze past data which includes
    - each user's summary
        - activity
        - duration
        - distance
        - steps
        - calories
    - each user's segment
        - place
        - move
- Generate useful information
    - Ranking
    - Personal Recommendation

What is the limit in this project:

- No new data for now

User Stories

Nowadays, there are many people who are busy but at the same time would like to maintain a healthy lifestyle. Our team aims to develop a fitness app that could help users keep track of their workout, monitor their process and enlarge their social circle. With this application, they could record their daily exercises, check out other people's workout nearby and receive fitness recommendations based on their exercise pattern and favorite activities. By doing so, they could easily keep track of their workout performance, not only themselves, but with a group of friends who share similar interests, and together maintain a healthy lifestyle.

Use Cases

- Ranking
    - A user could check their rank based on
        - calories per day, per week
        - activity time per day, per week
    - A user could see location ranking
        - popular places that have been visited the most, per week
- Activity Recommendation
    - A user could be recommended activities based on
        - user's activity patterns (in a specific timeframe, for example in this week, in this month)
        - user profile's favorite activities
        - what most people do recently (in a specific timeframe, for example in this week, in this month)
        - what most people are doing nearby (current time, current location)
        - how much calories is needed this week or month based on current user's BMI and target

Technology

1. Programming Language: Java
2. External Libraries
    - Spring Boot framework: for building and deploying a standalone, production-ready application.
    - Spring Data JPA: for database integration and management.
    - A relational database like MySQL or PostgreSQL for storing the app's data.
    - Maven or Gradle: for build automation and dependency management.
    - AWS or Heroku: for cloud deployment and hosting.

X Initial Design

X.1 - Ranking

Functionality	Input	Output	Design	Assignee	Sprint
Calories Rank	User ID *	User's Rank	Go to the database, calculate all people's calories sum and rank.	Yitian	2
Activity time Rank	User ID	User's Rank	Go to the database, calculate all people's calories sum and rank.	Ziwei	2
Location Rank	N/A	Location Rank	Go to the database, calculate all visited locations and rank by frequency.	Yi	2

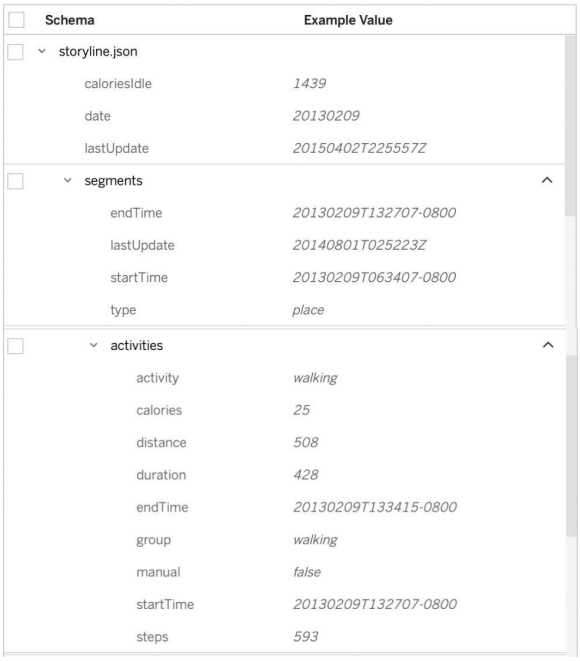
X.2 - Activity Recommendation

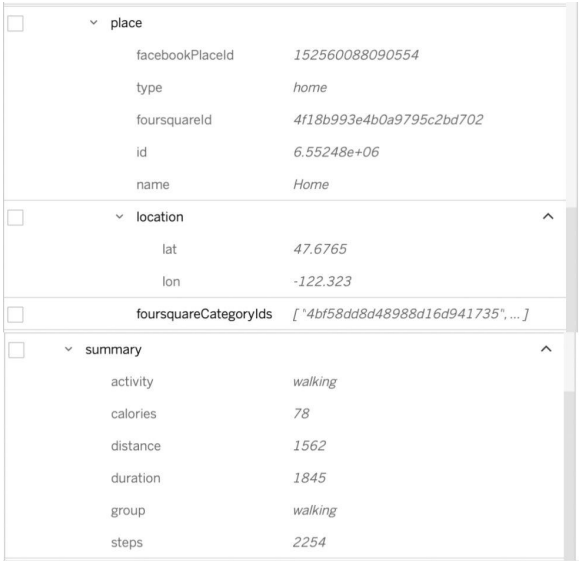
Functionality	Input	Output	Design	Assignee	Sprint
Recommend Activities based on patterns	User ID	Recommend Activities	Check user's pattern, and give recommendations	Yitian	2
Recommend Activities based on preference	User ID	Recommend Activities	Check user's preference, and give recommendations	Ziwei	2
Recommend Activities based on others' recent activities	User ID	Recommend Activities	Check other users' history, and give recommendations	Yi	2
Recommend Activities based on location	User ID	Recommend Activities	Check user's location and activities nearby, and give recommendations	Yitian	3
Recommend Activities based on diet	User ID	Recommend Activities	Check user's profile using the user id and do computing based on user's diet plan to	Ziwei	3

plan			give out appropriate activities		

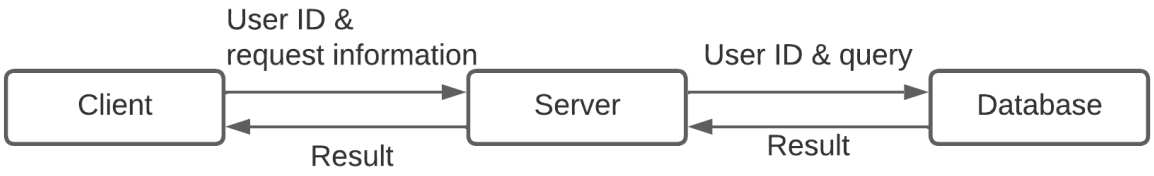
* User ID: Primary Key of user info table, which includes user's activities history, locations, weight, height, diet plan, calorie target, etc.

Data Schema

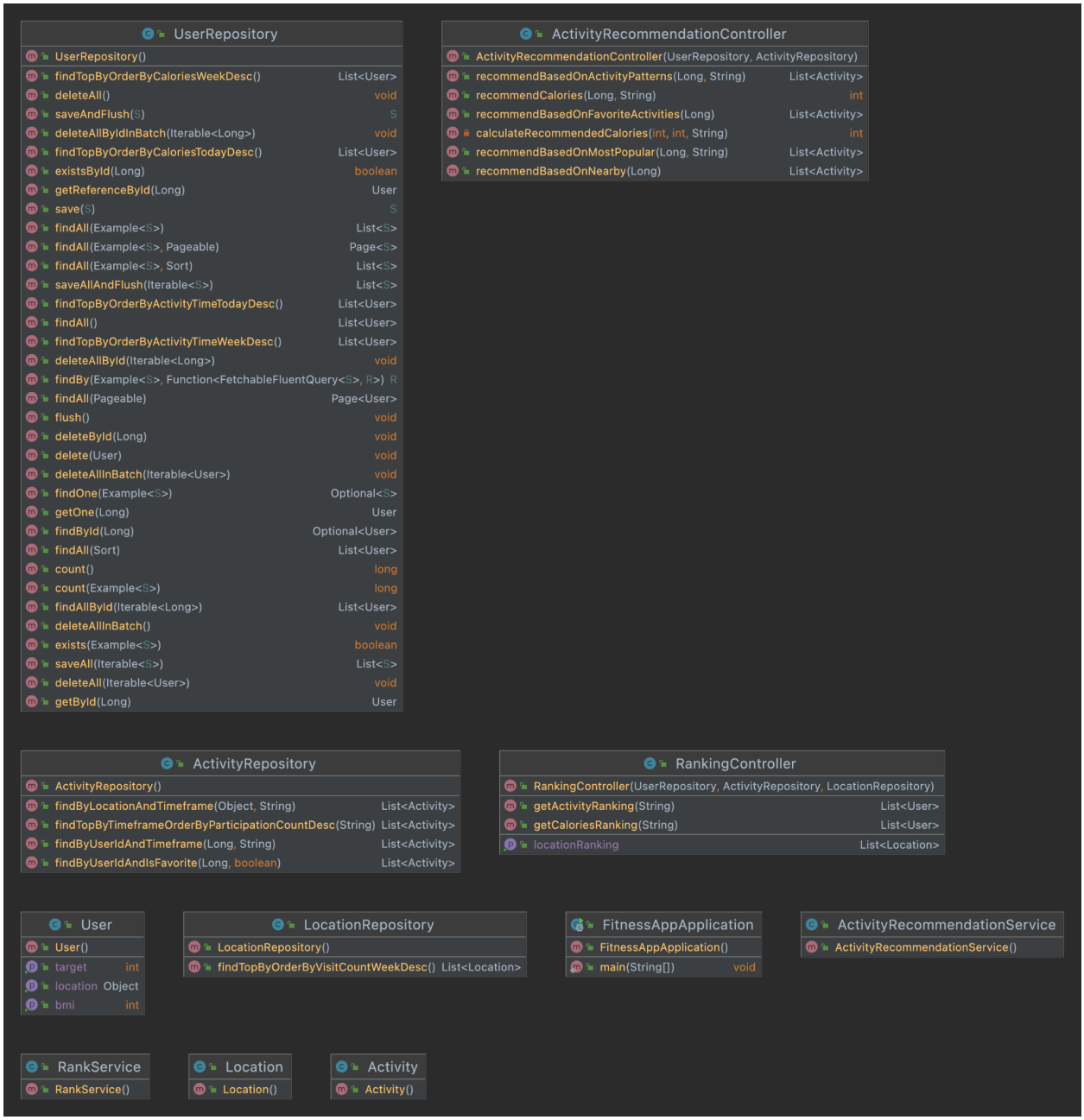




Sequence Diagram



UML


UserRepository
UserRepository()
findTopByOrderByCaloriesWeekDesc() List<User>
deleteAll() void
saveAndFlush(S) S
deleteAllByIdInBatch(Iterable<Long>) void
findTopByOrderByCaloriesTodayDesc() List<User>
existsById(Long) boolean
getReferenceById(Long) User
save(S) S
findAll(Example<S>) List<S>
findAll(Example<S>, Pageable) Page<S>
findAll(Example<S>, Sort) List<S>
saveAllAndFlush(Iterable<S>) List<S>
findTopByOrderByActivityTimeTodayDesc() List<User>
findAll() List<User>
findTopByOrderByActivityTimeWeekDesc() List<User>
deleteAllById(Iterable<Long>) void
findBy(Example<S>, Function<FetchableFluentQuery<S>, R>) R
findAll(Pageable) Page<User>
flush() void
deleteById(Long) void
delete(User) void
deleteAllInBatch(Iterable<User>) void
findOne(Example<S>) Optional<S>
getOne(Long) User
findById(Long) Optional<User>
findAll(Sort) List<User>
count() long
count(Example<S>) long
findAllById(Iterable<Long>) List<User>
deleteAllInBatch() void
exists(Example<S>) boolean
saveAll(Iterable<S>) List<S>
deleteAll(Iterable<User>) void
getById(Long) User
ActivityRecommendationController
ActivityRecommendationController(UserRepository, ActivityRepository)
recommendBasedOnActivityPatterns(Long, String) List<Activity>
recommendCalories(Long, String) int
recommendBasedOnFavoriteActivities(Long) List<Activity>
calculateRecommendedCalories(int, int, String) int
recommendBasedOnMostPopular(Long, String) List<Activity>
recommendBasedOnNearby(Long) List<Activity>
ActivityRepository
ActivityRepository()
findByLocationAndTimeframe(Object, String) List<Activity>
findTopByTimeframeOrderByParticipationCountDesc(String) List<Activity>
findByUserIdAndTimeframe(Long, String) List<Activity>
findByUserIdAndIsFavorite(Long, boolean) List<Activity>
RankingController
RankingController(UserRepository, ActivityRepository, LocationRepository)
getActivityRanking(String) List<User>
getCaloriesRanking(String) List<User>
locationRanking List<Location>
User
User()
target int
location Object
bmi int
LocationRepository
LocationRepository()
findTopByOrderByVisitCountWeekDesc() List<Location>
FitnessAppApplication
FitnessAppApplication()
main(String[]) void
ActivityRecommendationService
ActivityRecommendationService()
RankService
RankService()
Location
Location()
Activity
Activity()