Our idea for this project is to implement a web application that allows the users to browse and search mangas from a database using also some filters. For example the user can filter mangas through genre, author, year of pubblication and so on. When selecting a certain manga, the user can see more info and reviews.

We want to implement a review system that allows users, both un-registered and registered, to view reviews of a selected manga and allows the registered user to write a review.

The registered user will have a personal page which includes their information so they will be able to manage their profile.

A costumer can create their personal list of mangas and give it a name. For example he can create a list for “Mangas with bad endings”, or “mangas to read in the future”, or “wishlist” or “sci-fy mangas”.

So the costumer can create its own list and add mangas to the list. The costumer can decide to make the list public or private.

Costumers will be able to follow other costumer, and when they do it they can see the public information and lists.

The manager will be able to see analytics of the mangas and of the costumers. For example he can see the trends of the mangas, the most active users, the least read mangas and so on

This dataset source is created by using the official API of MyAnimeList: https://myanimelist.net/apiconfig/references/api/v2

The dataset contains manga id, title, type, score, number of users who scored this manga, publishing status, number of volumes, number of chapters, starting data and ending data.

We don’t have variety because we are only using one source, but we have variabilty/velocity: in fact the mangas are often updated. For example there will be updates about volumes, chapter, status. So after a certain time they may lose or gain importance because the users use the information of the application to take advice on what mangas to read, so if the application gives wrong advises, it can easily lose credibility and reliablity. Since the main feature of the application is to provide correct information, the data must be updated frequently.

About the simplified UML Analysis Class

The costumer can make more than one review, but the review belongs only to a costumer. The costumer …

We decided to use document DB to manage the information regarding users and mangas. The manga dataset was created by an API that collected for each manga different attributes, so using a SQL DB would have given us a lot of null values. Manga, reviews and registered users will be stored using a document DB.

We can have eventual consistency, so we can accept some delays, but it must be always available.

The porpouse is to represent on to many relationhip with an array of nested documents inside the manga collection instead of using different tables like in SQL DB. This can be useful to store the reviews inside a manga document. Using document DB we can create aggregation that can provide simplified complex queries to analyze and retirieve information quickly.

We can create indexes for fast retrieving of mangas based by year of pubblication, name, type and status.

We choose to use a graph db to rerpresent the social features of the web app like the following system, the liked mangas.

Lists will be stored using graph DB.

On graph DB we can make queries like show the list of following and followers of a user, or show all the users that like a manga, or show all the lists of a user.

We use mongo DB for the document DB, neo4j for the graph DB. For the backend we are going to use IntelliJ IDE for using Java. For the frontend we are going to use HTML, CSS and JavaScript.

To manage the project we’ll use the IDE of IntelliJ.

We’ll use Tomcat as a web server.