- Include a high-level description of your classes using CRC Cards: what they are, what their responsibilities are, and what is the interaction between them.

- You can use the following template for the CRC Cards:

  Class name: User
  Parent class (if any): None
  Classname Subclasses (if any): None
  Responsibilities:
  - Create user
  - Edit profiles
  - Get profiles

  Collaborators:
  - Fandom
  - Post
  - Event
  - Search

- The description of system interaction with environment should indicate any dependencies or assumptions made about the operating environment of the system. E.g. OS, programming language compilers and virtual machine, DB's, network conguration, etc.

- Describe the architecture of the system, that is the most abstract view of how your system is divided into components and how those components are interconnected. The architecture should be described with a diagram showing components and how they are related (or equivalent in words). Beware of designs based on large number of components, they may signal a design that is overly complex.

- The system decomposition should relate the system architecture to the detailed design, to identify the role of each component in the higher-level architectural view. Description of strategy for dealing with errors and exceptional cases (e.g. invalid user input, network or external system failure) that might arise in the use of the software. For anticipated errors and exceptions, a summary of how the software will respond in these situations.

CRC Cards

Class name: Fan
Parent class : None
Classname Subclasses: Fan, Fandom Owner
Responsibilities:

- Create user
- Edit profiles
- Get profiles
- Request to create Fandom
- Create posts on a particular Fandom
- Create event

Collaborators:

- Fandom
- Post
- Event
- Administrative System

Class name: Fandom
Parent class :
Classname Subclasses (if any): List all the subclasses separated by a comma.
Responsibilities:

- Create user’s own Fandom
- Create posts on a particular Fandom
-

Collaborators:

- Fandom
- Post
- Event

Class name: Administrative System
Parent class:
Classname Subclasses (if any): List all the subclasses separated by a comma.
Responsibilities:

- Respond to a creating-user request.
- Respond to user’s request to create Fandom

Collaborators:

- Fandom
- User

Class name: Fandom Owner
Parent class : Fan
Classname Subclasses: None
Responsibilities:

- inherit all responsibilities from
- Remove fans from Fandom
- Editing Fandom profile

Collaborators:

- Fandom
- Post
- Event
- Administrative System

Class name: Event
Parent class :
Classname Subclasses: None
Responsibilities:

- Containing Event info
- Add participants

Collaborators:

- User

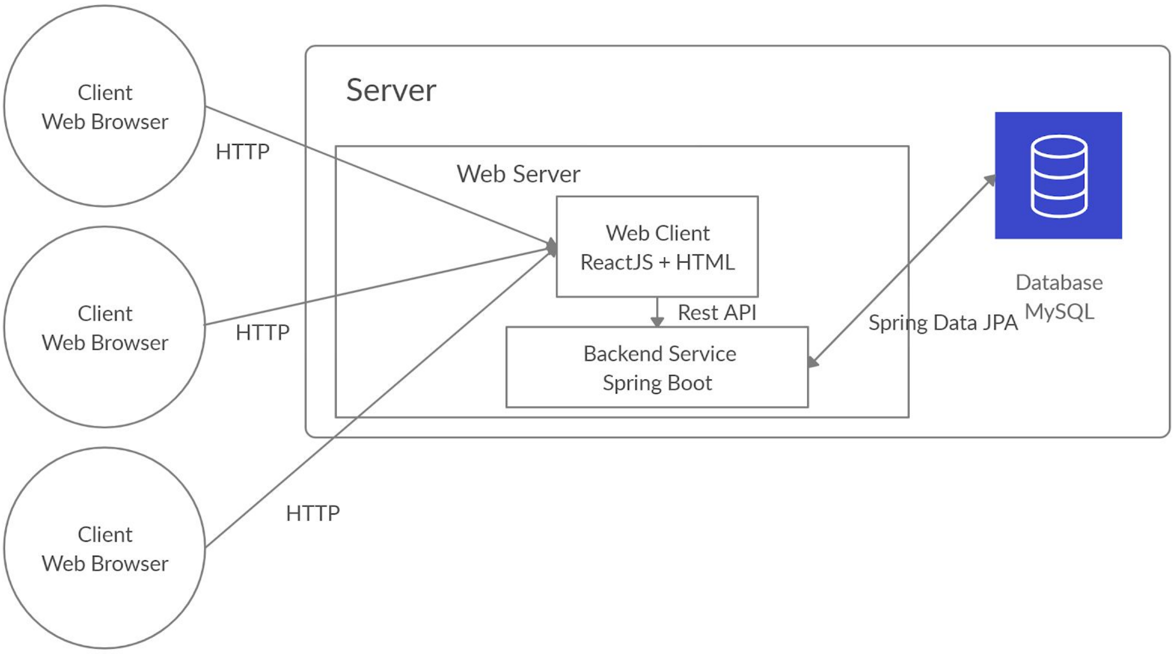
Class name: Post
Parent class :
Classname Subclasses: None
Responsibilities:

- comment on a post
- attach text
- attach picture

Collaborators:

- Fandom
- User

Software Architecture


Client
Web Browser
Client
Web Browser
Client
Web Browser
HTTP
HTTP
HTTP
Server
Web Server
Web Client
ReactJS + HTML
Rest API
Backend Service
Spring Boot
Spring Data JPA
Database
MySQL