TRƯỜNG ĐẠI HỌC FPT

CAPSTONE PROJECT REPORT

Report 4 – Software Design Document

– Danang, January 2022 –

Table of Contents

I. Software Design Document

1. Overall Description

1.1. Assumptions

This application will work on the web platform and support users to use this app on pc, and mobile

1.2. Design Constraints

- Use Axios to get data from back-end to front-end.
- Support login with Google, Facebook, email - password.
- Verify registered email by sending a verified URL via email.
- User interface should be clear and easy to use.
- User's sensitive data must be encrypted before being stored in the database.
- Latency of back-end systems can handle lower than 2s.
- Can handle 200 requests/s.

1.3. Technology Suggestion

- We can use VueJS or AngularJS instead of ReactJS for the front-end.
- Using Docker, we can easily deploy to a VM's server and it is very fast and light. You can create and run docker containers in seconds.
- Firebase allows us to easily log in by email or authenticate with Google or Facebook. Besides, it can sync the real-time data across the web without refreshing the screen.
- MongoDB is a document-based database in which a collection holds different documents. The field number, content, and size of one document may differ from another. MongoDB is easy to extend.

2. System Architecture Design

2.1. Overall Architecture

2.1.1. Mern stack

- MERN stands for MongoDB, Express, React, Node, after the four key technologies that make up the stack.

    - MongoDB - document database
    - Express(.js) - Node.js web framework
    - React(.js) - a client-side JavaScript framework
    - Node(.js) - the premier JavaScript web server

- Express and Node make up the middle (application) tier. Express.js is a server-side web framework and Node.js is the popular and powerful JavaScript server platform. Regardless of which variant you choose, ME(RVA)N is the ideal approach to working with JavaScript and JSON, all the way through. The MERN architecture allows you to easily construct a 3-tier architecture (frontend, backend, database) entirely using JavaScript and JSON.

- Reasons for choosing MERN stack:
    - The speed of design and development of websites and web applications.
    - Reducing server costs.
    - The performance of greatly optimized web applications and software.
    - The ease of transposing a web application to a mobile application or software, thanks in particular to React Native.
    - The luxury of designing a website using a single HTML document.
    - The development of a computer application using a single language, JavaScript.

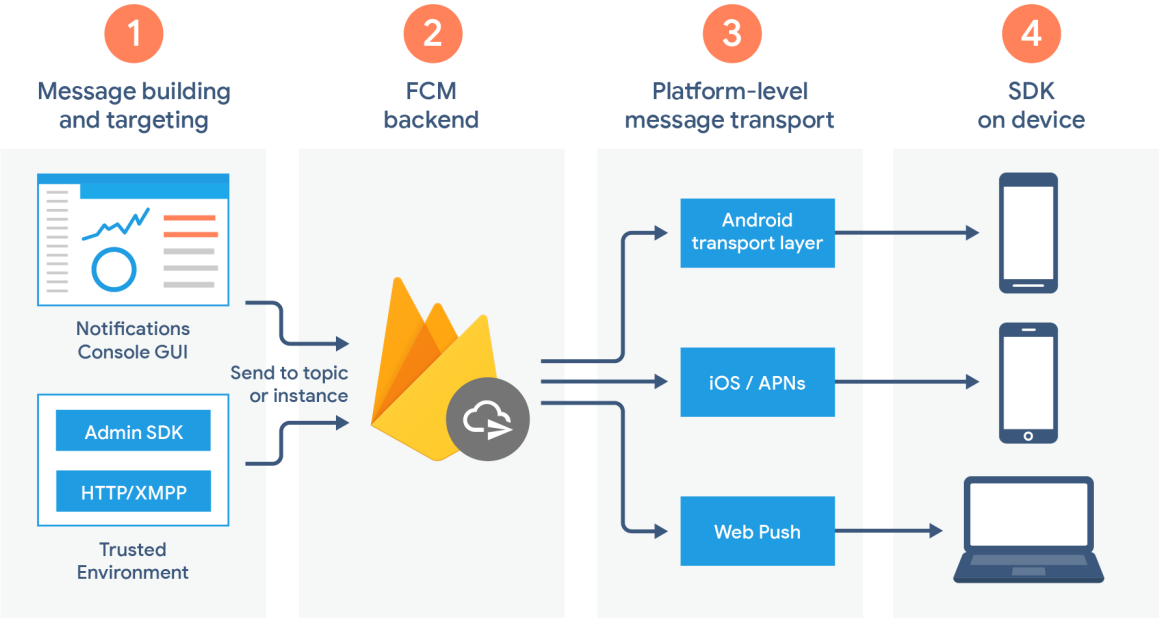
2.1.2. Authentication

- Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.
- In the application, we are using Firebase Authentication to handle sign-in, sign up with Email - Password, or identity providers like Google, and Facebook.
- We also use Firebase Authentication for email actions like verifying email and resetting passwords.

- Reasons for choosing Firebase Authentication:

    - Firebase Authentication aims to make building secure authentication systems easy while improving the sign-in and onboarding experience for end-users. It provides an end-to-end identity solution, supporting email and password accounts, phone auth, Google, Twitter, Facebook, GitHub login, and more.
    - Advantages of Firebase Authentication: Flexible, drop-in UI, Comprehensive security, Fast implementation

2.1.3. Notification Flow



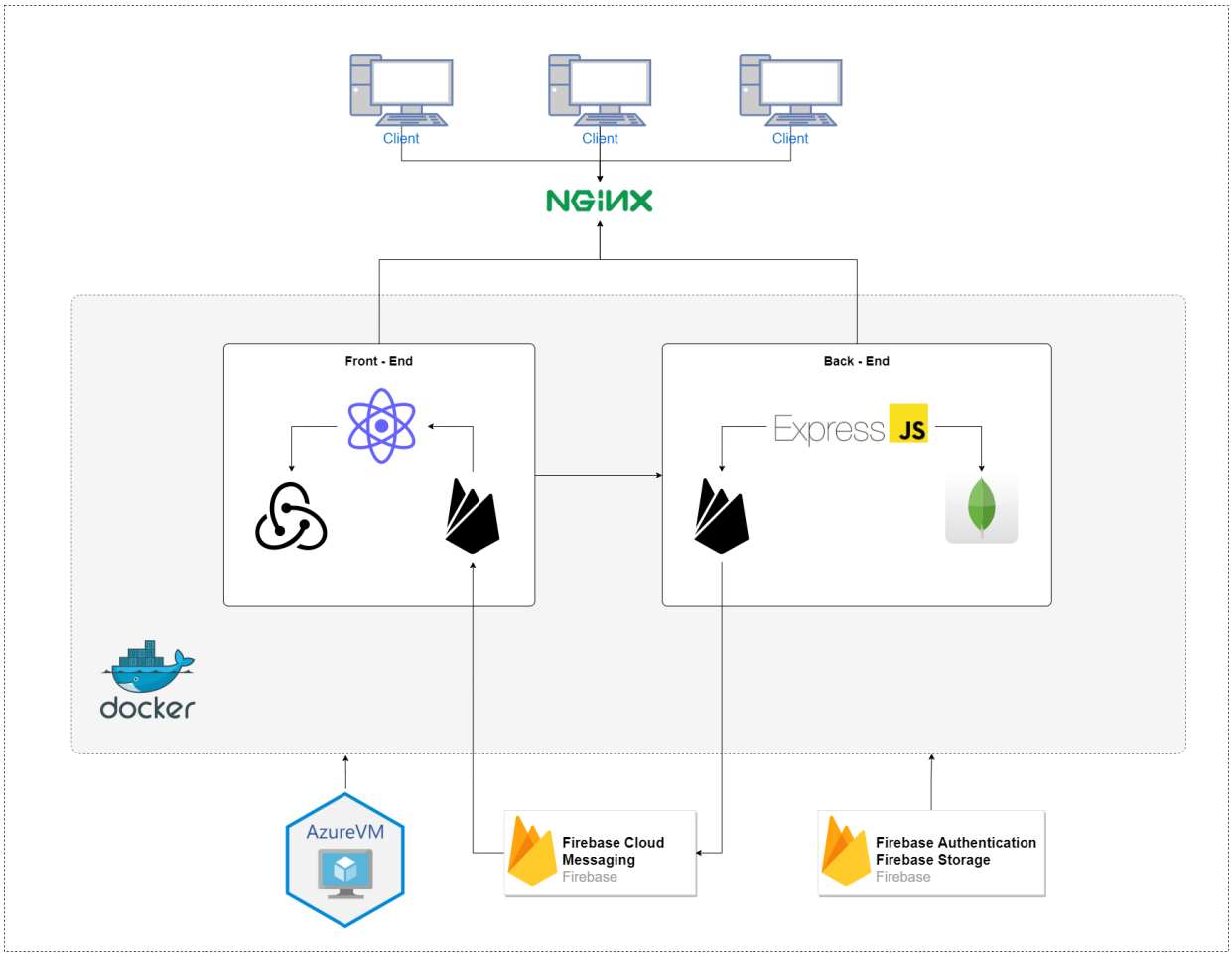
- Firebase Cloud Messaging (FCM) is a cross-platform messaging solution that lets you reliably send messages at no cost.
- In our project, notification can be sent when these events listed in UC38_Type of Notification are triggered.

- Reasons choosing Firebase Cloud Messaging (FCM):

    - Firebase Cloud Messaging (FCM) provides a reliable and battery-efficient connection between your server and devices that allows you to deliver and receive messages and notifications on iOS, Android, and the web at no cost.
    - Firebase Cloud Messaging (FCM) provides a feature that allows you to develop in a notification form without much coding effort. A graphical console is supported by it for dispatching messages, thus removing creating a server

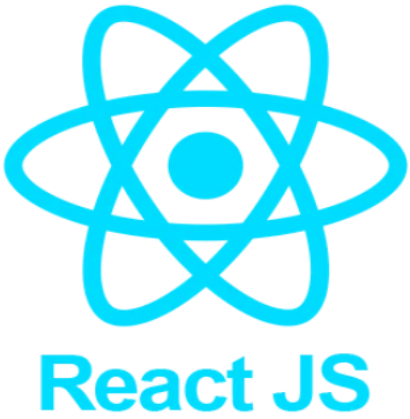
2.2. System Architecture

2.2.1. Architecture Diagram



2.2.2. System Architecture Explanation

2.2.2.1. ReactJS



- ReactJS is a JavaScript library created by the collaboration between Facebook and Instagram. It allows web developers to create user interfaces quickly. The Views section of Reactjs is usually displayed using mainly components that contain specific components or HTML tags. One of the unique features of ReactJS is that the rendering of data can be done not only at the server layer but also at the client layer.

- Reasons for choosing React JS:

    - Helps to build rich user interfaces: React supports frontend libraries such as Bootstrap, Material UI, Chakra, Tailwind, etc to build rich UI.
    - Offers fast rendering: the DOM model is tree-structured. So, a minor modification at a higher level layer can impact the user interface of an application. To solve this, Facebook has introduced a virtual DOM feature. Virtual DOM is the virtual representation of DOM that allows testing all changes to the virtual DOM first to calculate risks with each modification.
    - Strong community support: Currently, React JS has attained 136,079 stars on Github and 1,331 regular contributors. Not only that, but experts are also regularly uploading free React tutorials on Youtube and writing in-depth React tutorial articles & blogs on the internet. For instance, a simple “free React tutorial” search on Google gives 13,00,00,000 results. The latest survey by Stack Overflow also revealed that React is the most loved web framework, while Angular ranked 9th on the same list.
    - Reusable elements: Technology used to be far more complicated than that, but ReactJS provides us with the ability to do just the same. Each React project is constructed using the so-called reusable elements. This means that by calling from other components,

each element of the interface that you have already built can be used anywhere in your project.

2.2.2.2. Redux

- Redux is an open-source JavaScript library for managing the application state. It is most commonly used with libraries like ReactJS or AngularJS to build user interfaces, and easy to manage states.
- Redux helps you write applications that work consistently, run in different environments (client, server, and native), and are easy to test.
- Centralizing your application's state and logic enables powerful capabilities like undo/redo, state persistence, and more.

- Reasons for choosing Redux Toolkit:

    - Redux Toolkit makes it easier to write good Redux applications and speeds up development, by baking in our recommended best practices, providing good default behaviors, catching mistakes, and allowing you to write simpler code. Redux Toolkit is beneficial to all Redux users regardless of skill level or experience. It can be added at the start of a new project or used as part of an incremental migration in an existing project. using Redux Toolkit will make your code better and more maintainable.

2.2.2.3. Firebase

- Firebase is a cloud-based database service - cloud, along with an extremely powerful server system of Google. Firebase is a platform brought by Google, to support the creation of a high-quality Web App. It helps to authenticate login with Google, Facebook, or log in with Email and it also has a real-time database, which allows users to update when refreshed.
- Reasons for choosing Firebase:
    - We use Firebase to use Firebase built-in functions like Authentication, Push Notification, Storage. Helps reduce project development time, and is easy to use, and easy to maintain. Reduce project cost because firebase functions are free

2.2.2.4. Docker

- Docker is a platform to provides an easier way to build, deploy and run applications using containers (on a virtualization platform).
- We are using Docker for the sync environment for memopen-source platform and using docker-compose to deploy.
- Reasons for choosing Docker:
    - Caching a cluster of containers
    - Flexible resource sharing

- Scalability - many containers can be placed in a single host
- Running your service on hardware that is much cheaper than standard servers
- Fast deployment, ease of creating new instances, and faster migrations.
- Ease of moving and maintaining your applications
- Better security, less access needed to work with the code running inside containers, and fewer software dependencies

2.2.2.5. Azure

- Microsoft Azure is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through data centers managed by Microsoft.
- Azure includes infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). It can be used for services like analytics, desktop "virtualization", storage, networking... Azure can be used to replace or complement your on-premises servers.

- Reason for choosing Azure:

    - Container-based solutions ultimately provide cost savings benefits. Containers are a solution to deployment problems because they remove the friction caused by an absence of dependencies in production environments. Removing those issues, improves Dev/Test, DevOps, and production operations significantly.
    - A Docker container is becoming the standard unit of deployment for any server-based application or service.
    - Azure VMs hosting containers are a fast and simple way to create small Dev/Test environments in the cloud.
    - When placing containerized applications in production you will always create or adopt a DevOps culture and DevOps tools for CI/CD pipelines, like Azure DevOps Services or Jenkins.

- Microsoft Azure provides the most comprehensive and complete environment to modernize your existing .NET Framework applications with Windows Containers, cloud infrastructure, and PaaS services.

2.2.2.6. NGINX

- NGINX is a powerful open-source web server. NGINX uses single-threaded, event-driven architecture so it is more efficient than the Apache server. It can also do other important things, such as load balancing, HTTP caching, or use as a reverse proxy.

- Reasons for choosing NGINX:

  - It provides a more friendly configuration format and has a more modern design than any other web server alternatives. It is event-based and allows you to handle multiple connections without having overhead due to context switching. It uses less memory and resources.

2.2.2.7. MongoDB

- MongoDB is a NoSQL database, a type of document-oriented database. They are often used to store large volumes of data. MongoDB does not use a tabular structure like a relational database. Instead, MongoDB will store the data as a Document JSON. So each collection will have different sizes and documents. Besides, the fact that the data is stored in JSON - style documents leads to them being queried very quickly.

- Reasons for choosing MongoDB:

    - Using MongoDB can provide many benefits to a software development team. Its flexible schema makes it easy to evolve and store data in a way that is easy for programmers to work with. MongoDB is also built to scale up quickly and supports all the main features of modern databases such as transactions.

2.2.2.8. NodeJS

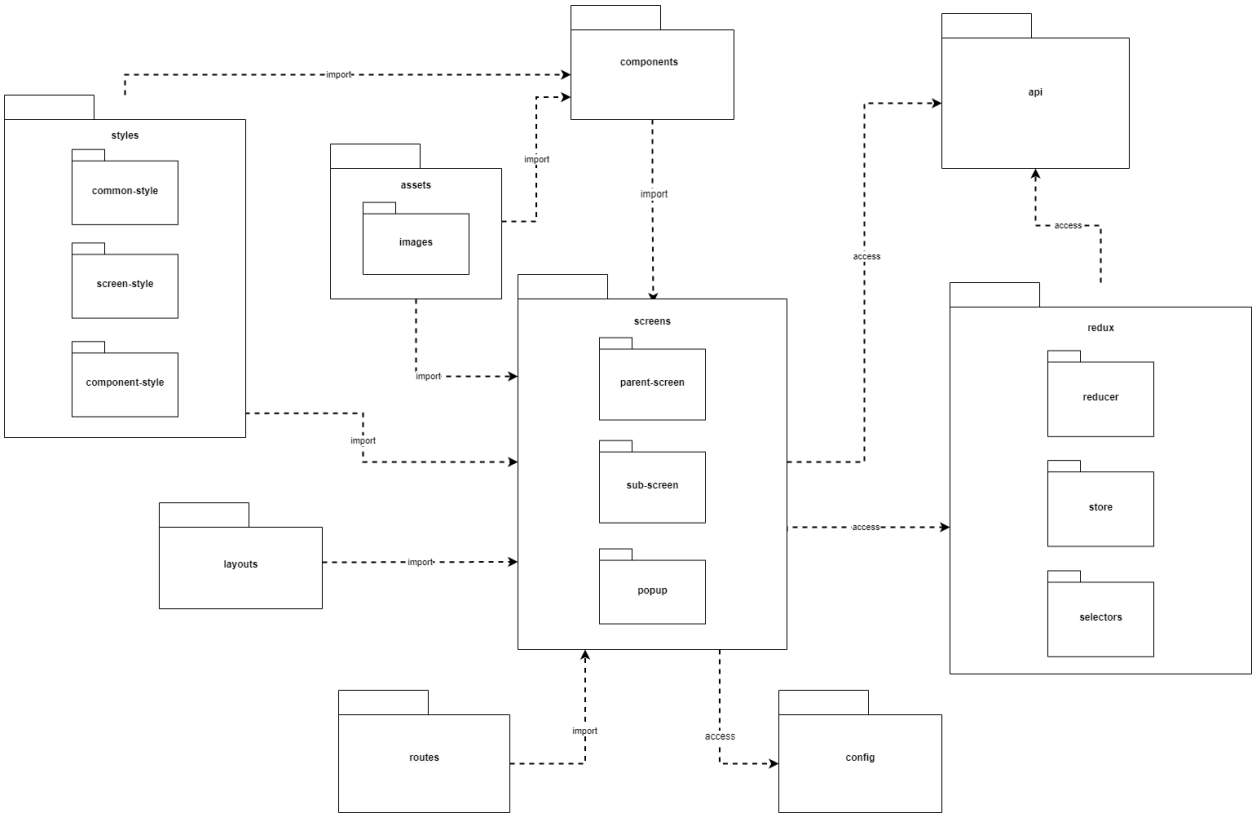
- NodeJS is a software system designed for writing scalable Internet applications, especially web servers. The program is written in JavaScript, using event-driven, asynchronous input/output to minimize total overhead and maximize scalability. NodeJS includes Google's V8 JavaScript engine, libUV, and several other libraries.
- Reasons for choosing Node JS:
    - Node js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js is the greatest tool for building real-time web applications. It provides cross-platform applications which run easily on any web. So you don’t need anything extra for running up a node application. You only need to make one. It’s a light, scalable, and open-source language platform that makes it very easy to build apps even at the enterprise level also. Overall it increases the efficiency of the

development process as it fills the gap between frontend and backend applications. It uses the approach of non-blocking I/O. In the Non-blocking I/O approach, you can initiate a request in parallel for user 2 without waiting for the response to the request for user 1. The requests in Node.js can initiate in parallel. This non-blocking I/O eliminates the need for multi-threading.

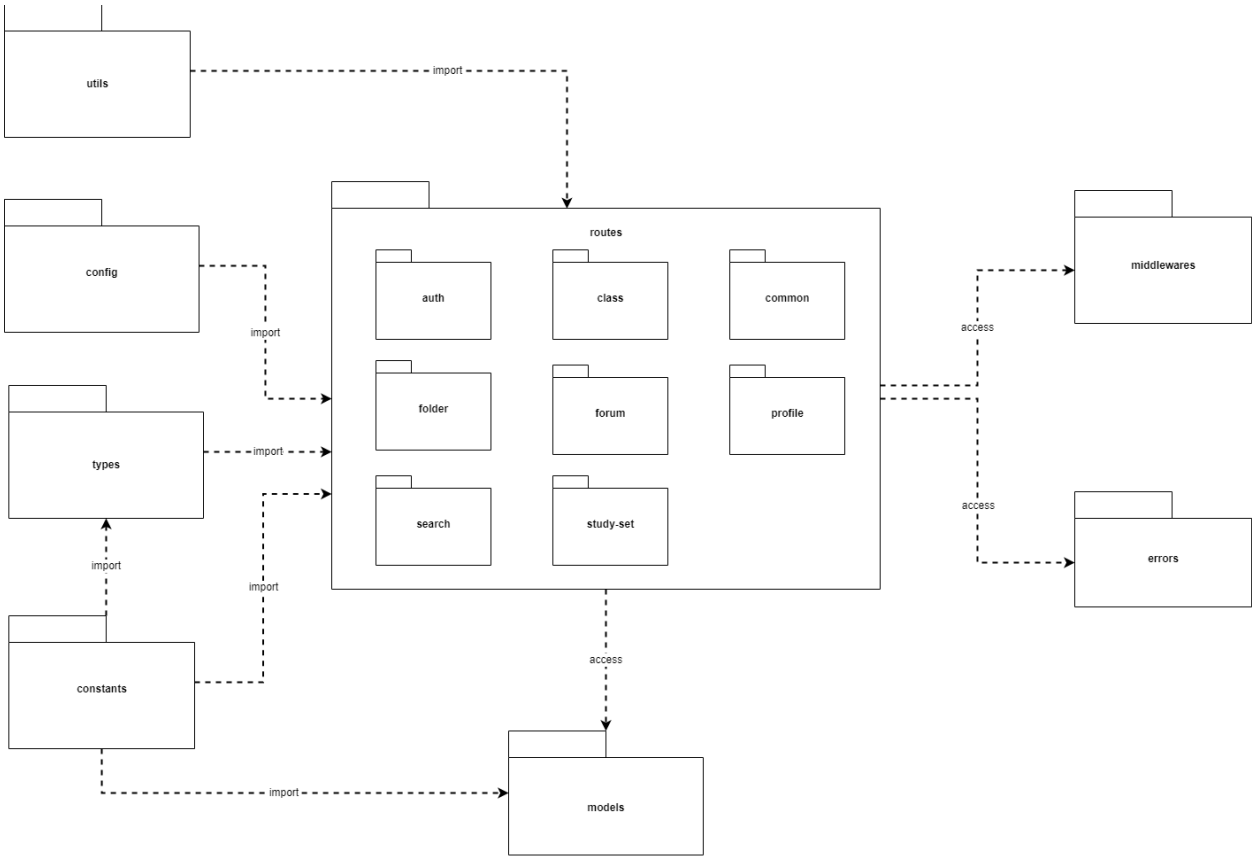
2.3. Package Diagram

- Front_end package diagram
- Back_end package diagram

2.3.1. Package diagram of Front-end



2.3.2. Package diagram of Back-end



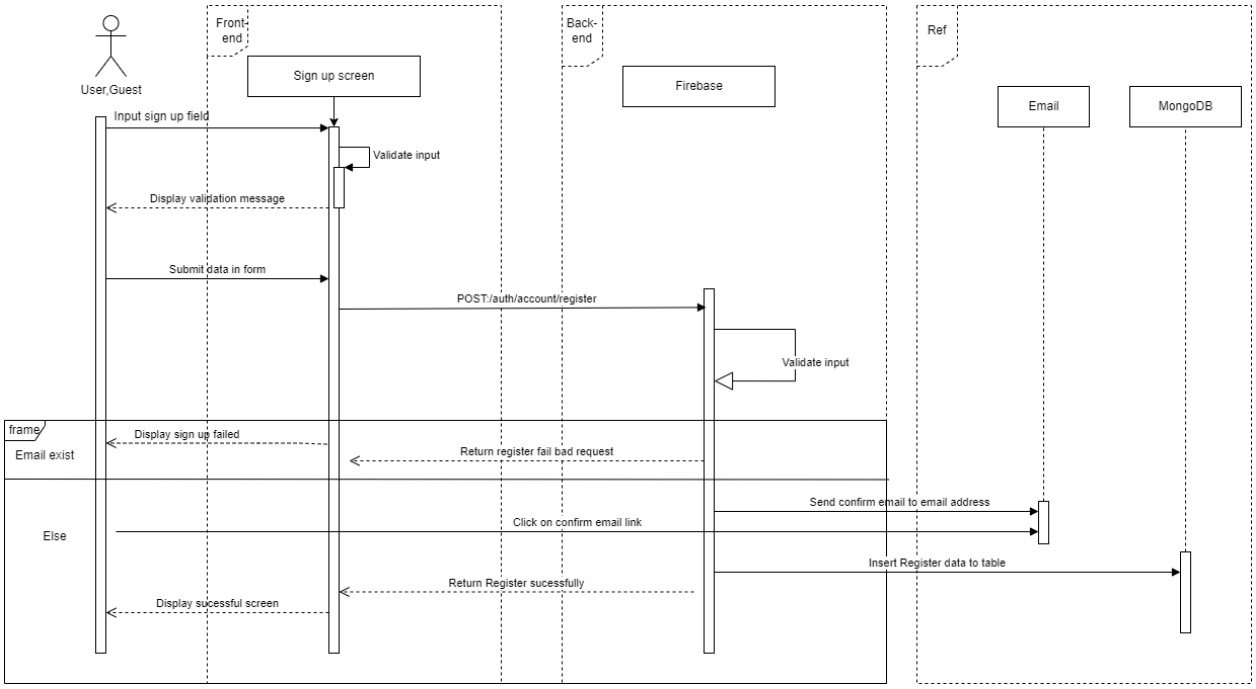
3. System Detailed Design

3.1. Sequence Diagram

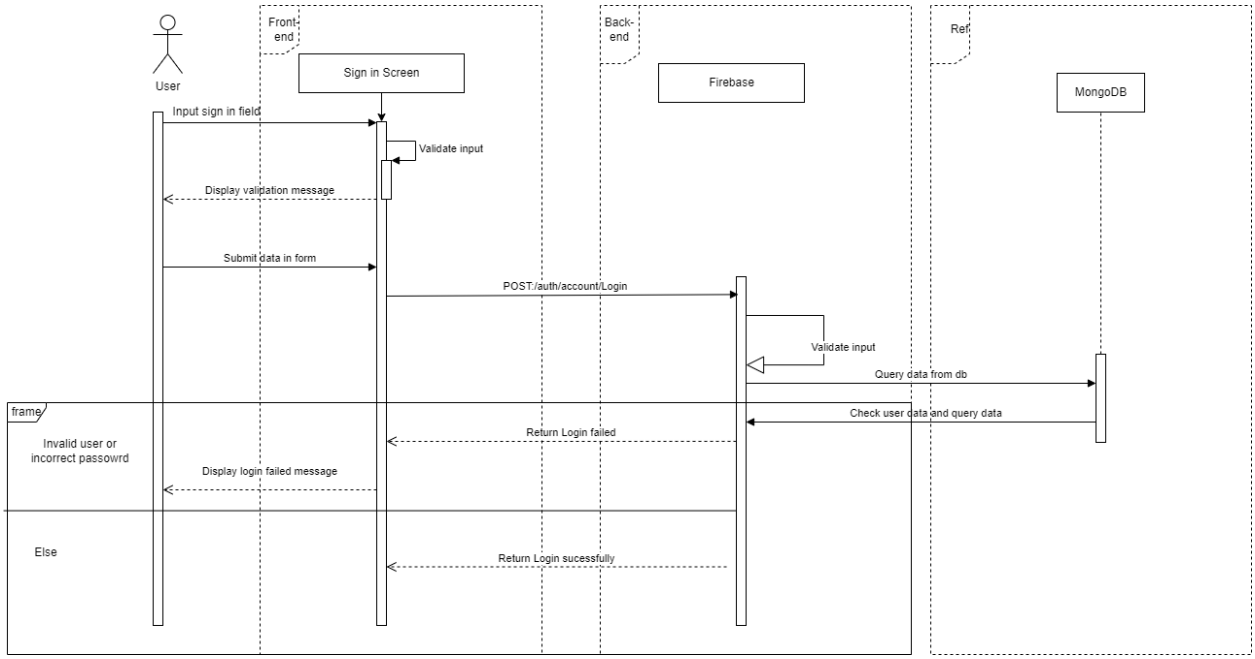
For all of the sequence diagrams' raw images, view them here.

We use the sequence diagram to define the interactions between groups of objects, the messages sent and received between the objects, and the chronology between those messages. We also use sequence diagrams to describe the flow process of all features. For instance, in the sign-up screen, we apply a sequence diagram so the client will know the path of this screen

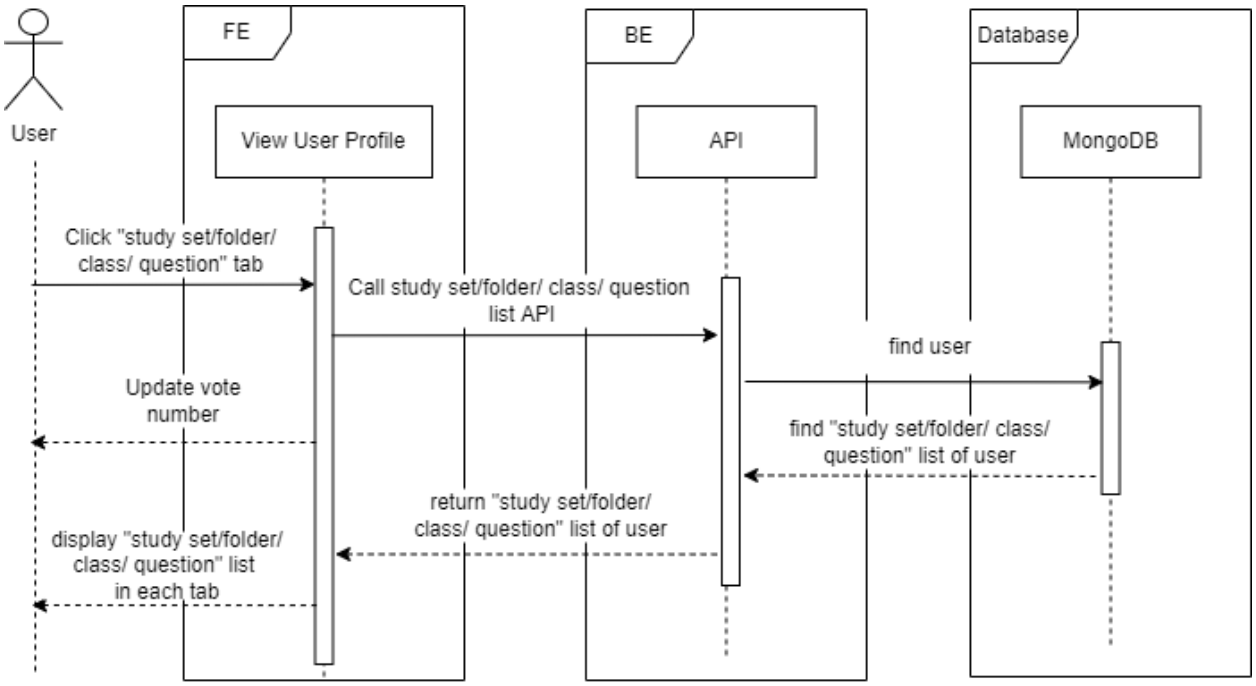
3.1.1. Sign up



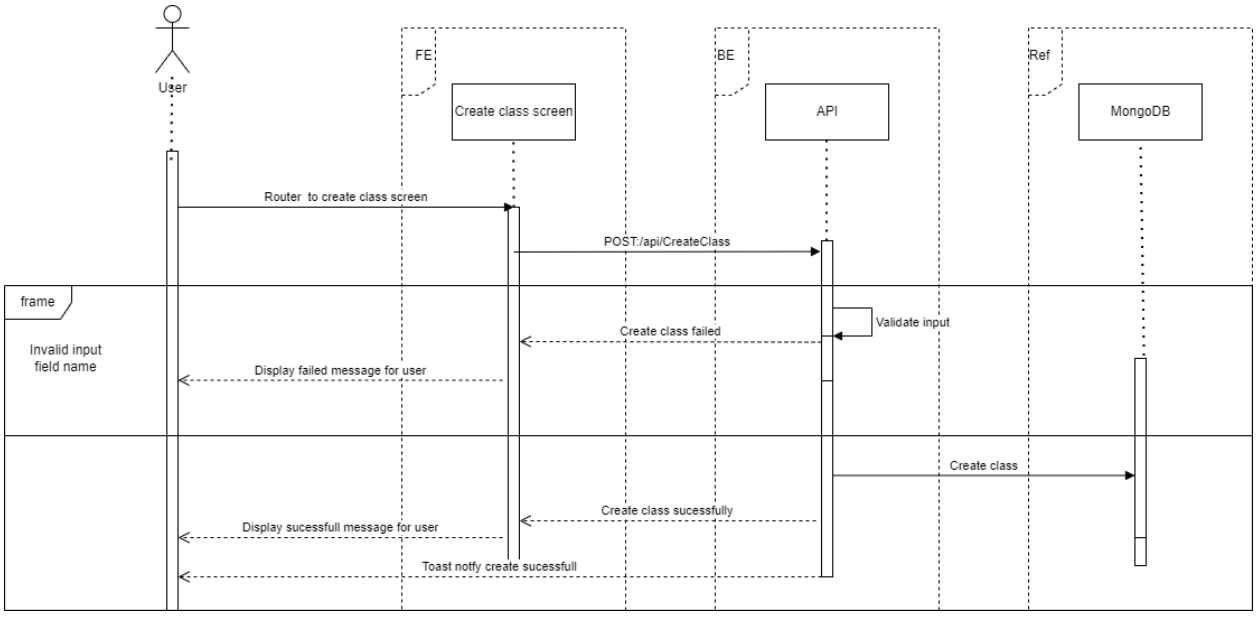
3.1.2. Sign in



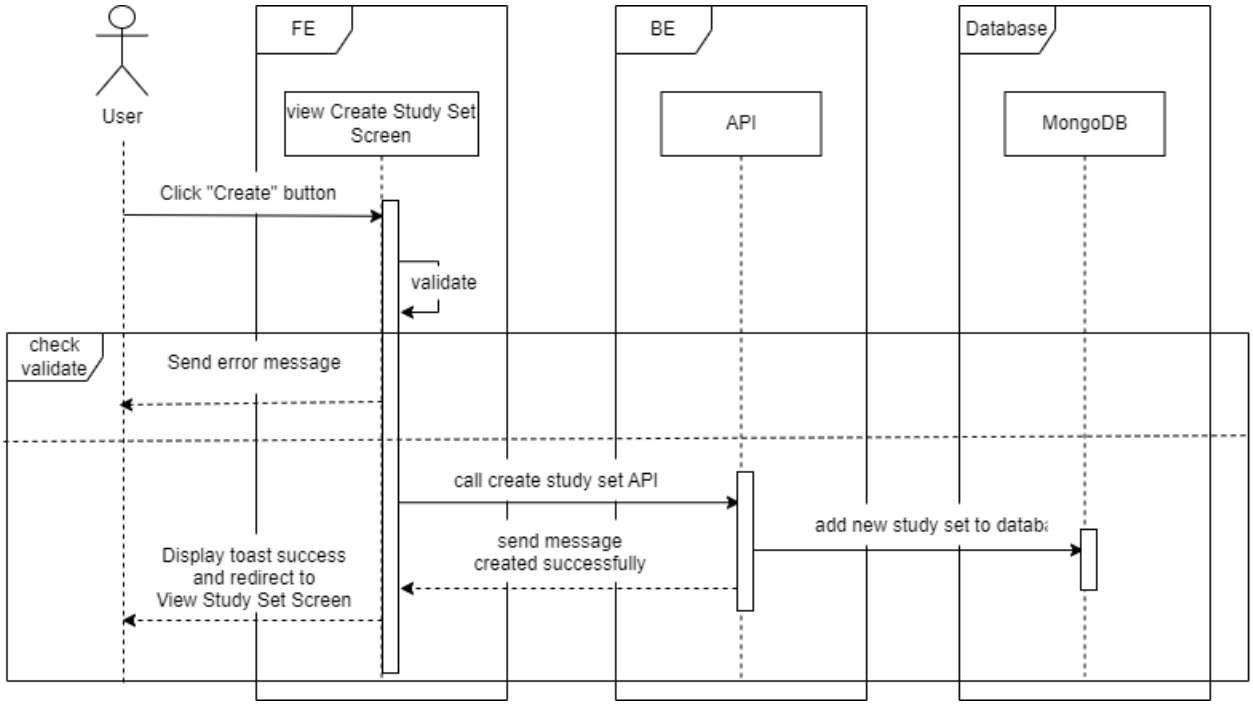
3.1.3. View user profile



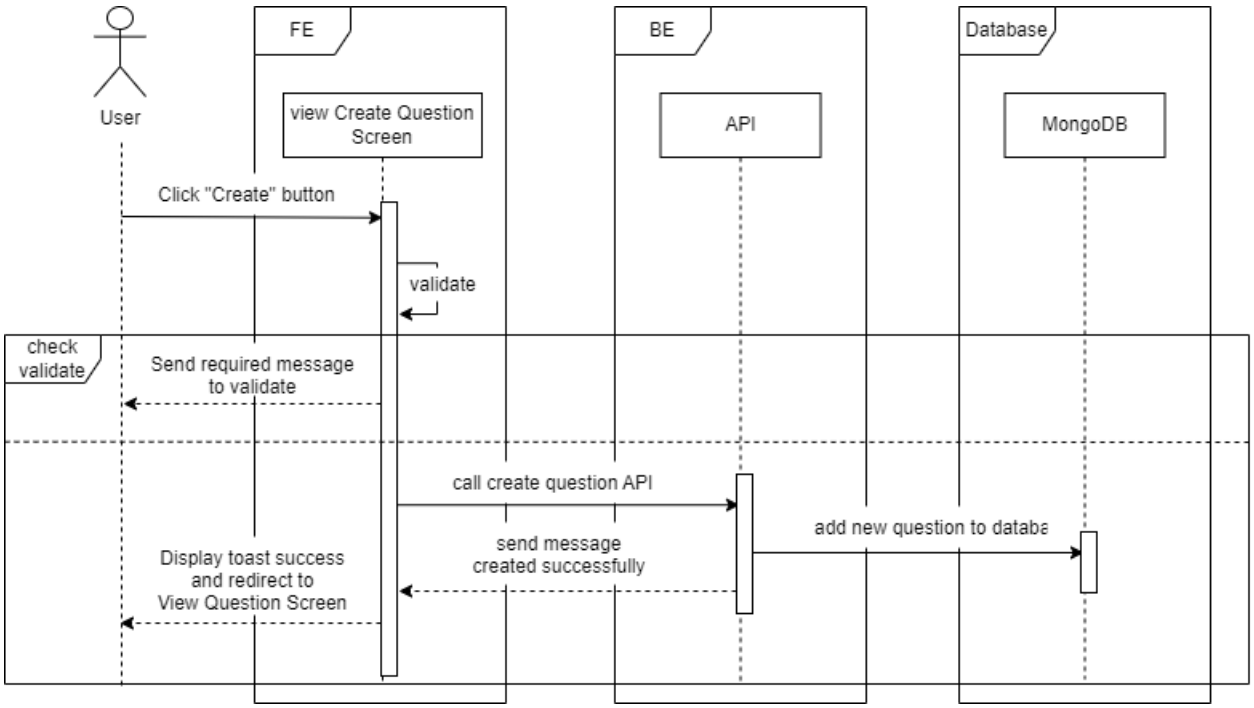
3.1.4. Create class



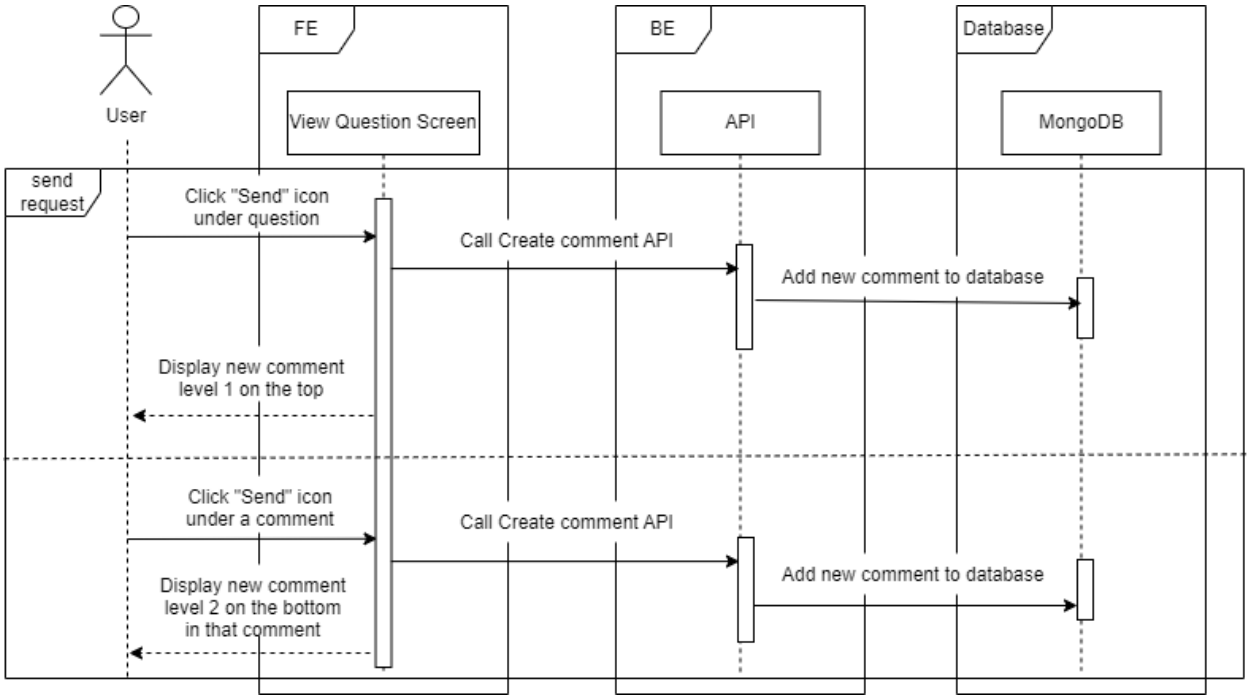
3.1.5. Create study set



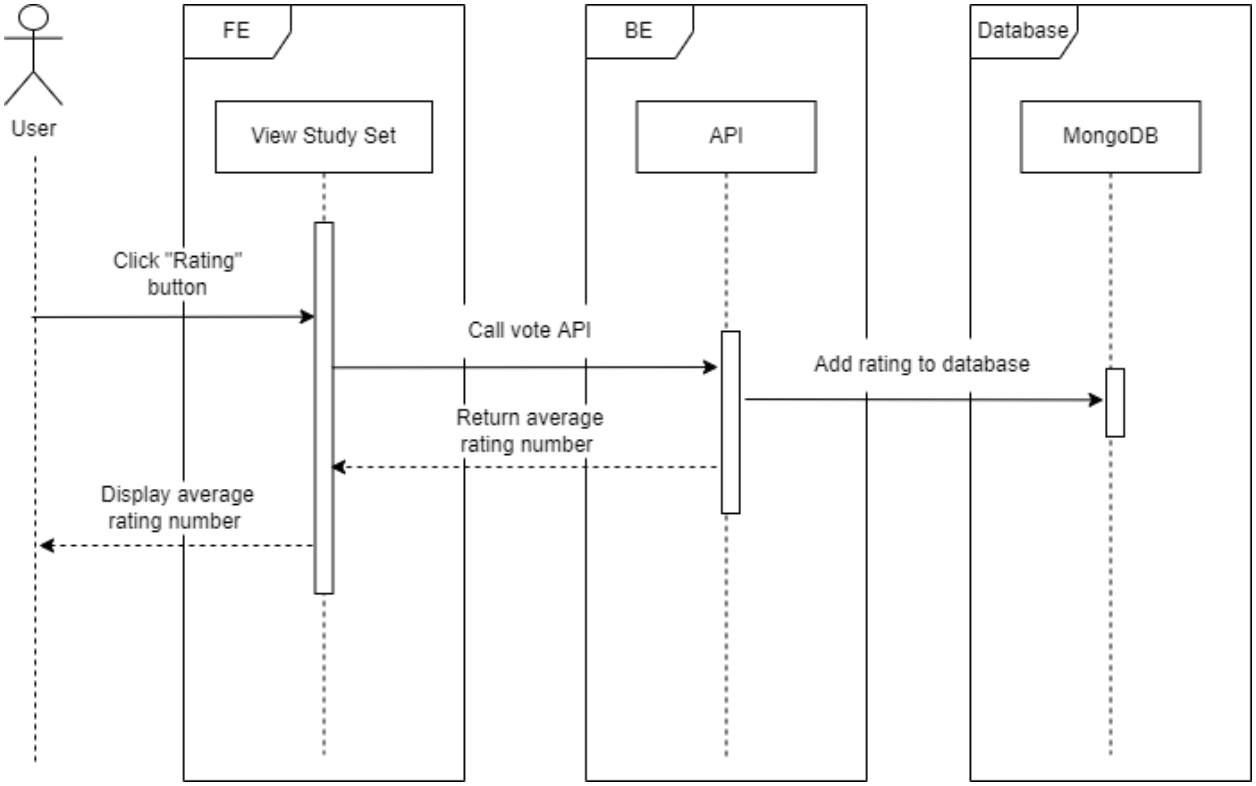
3.1.6. Create question



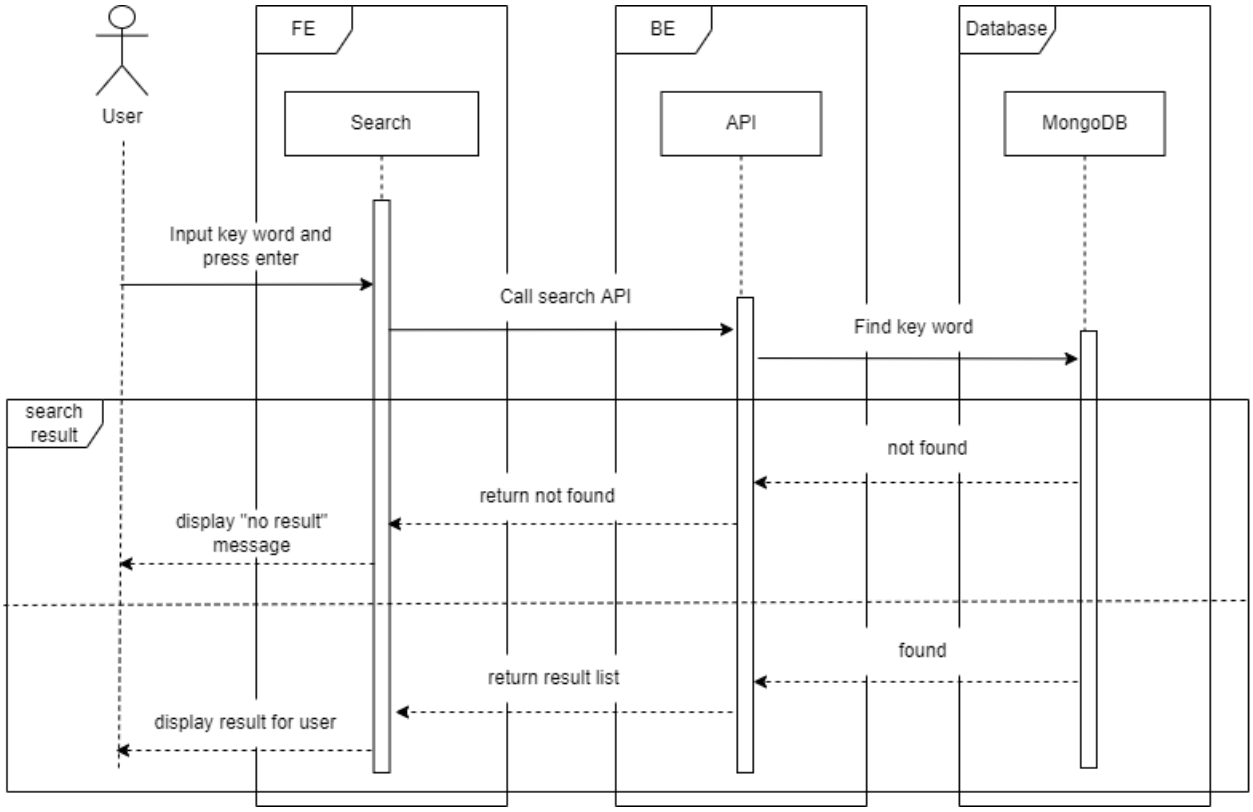
3.1.7. Create comment



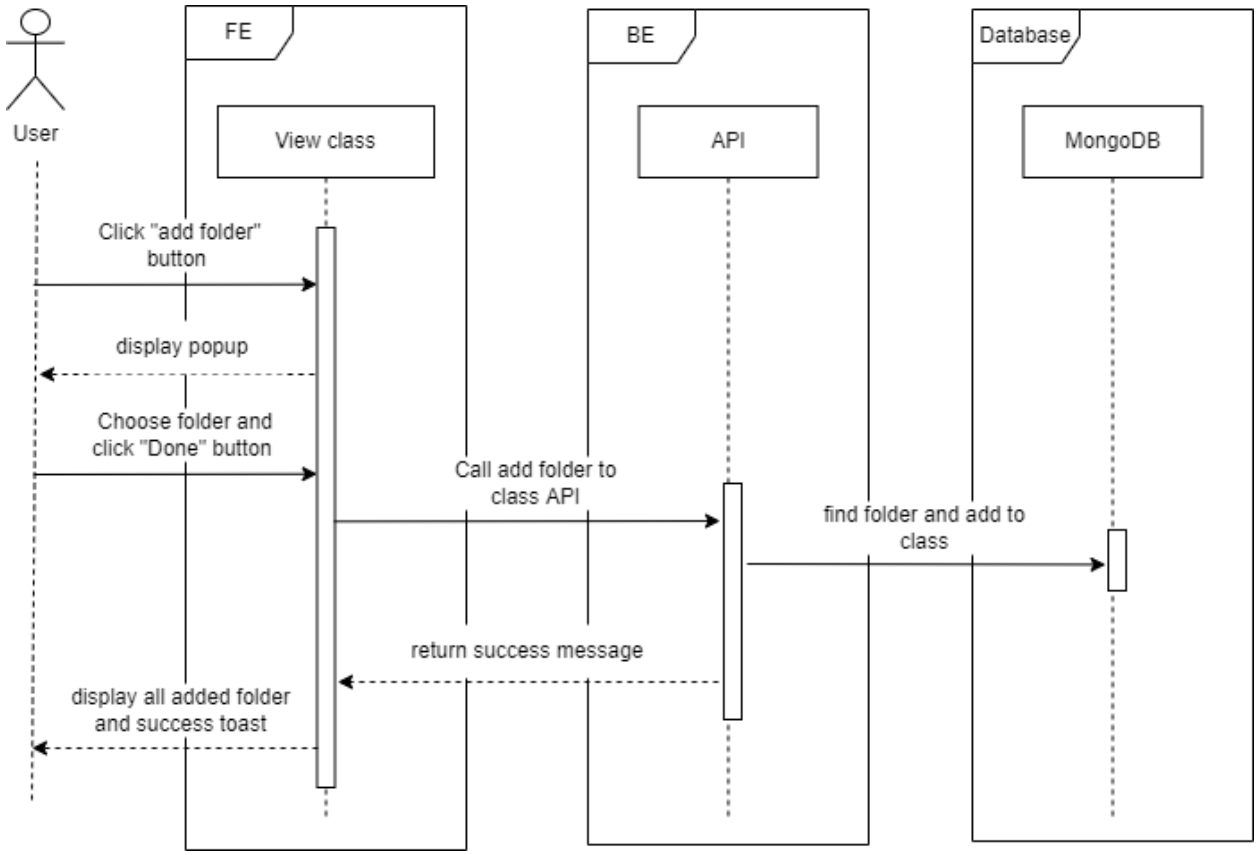
3.1.8. Rate study set



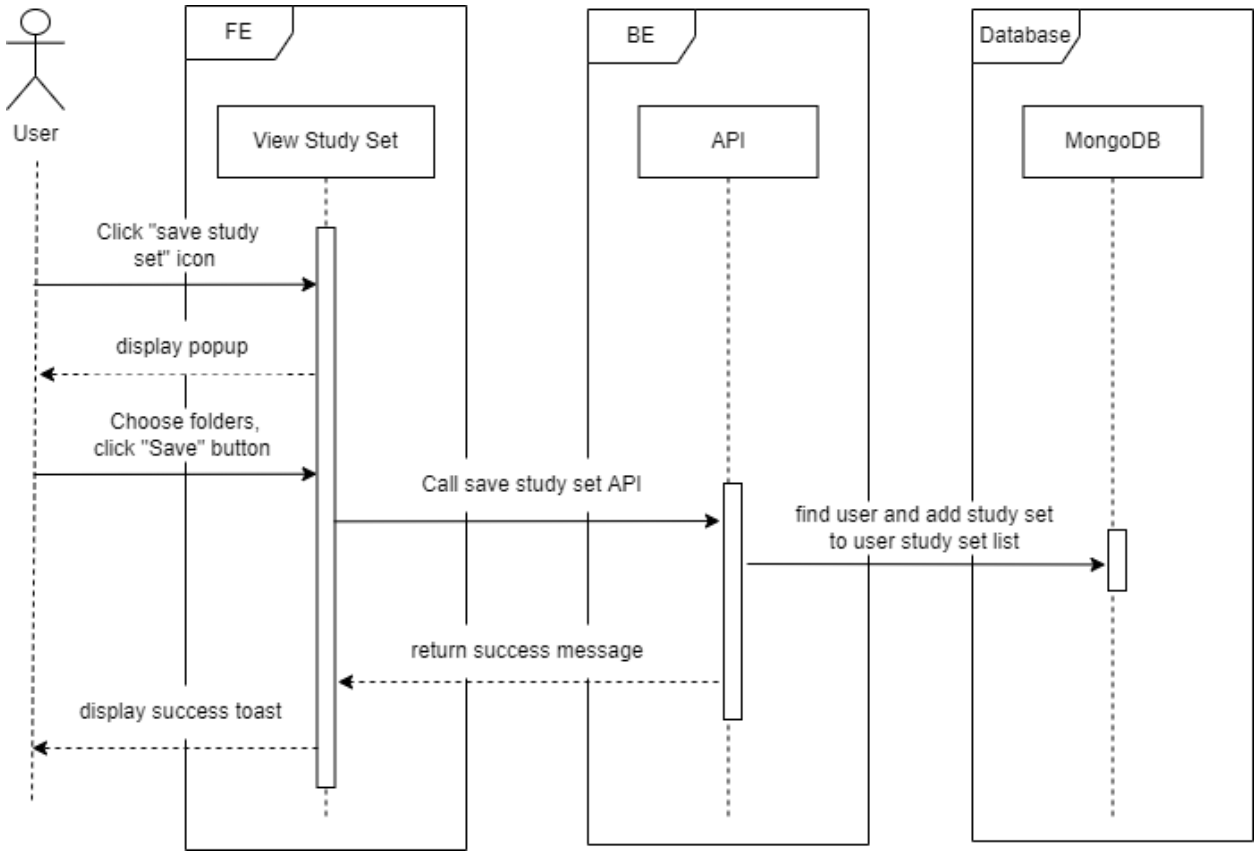
3.1.9. Search


User
FE
Search
BE
API
Database
MongoDB
Input key word and press enter
Call search API
Find key word
search result
not found
return not found
display "no result" message
found
return result list
display result for user

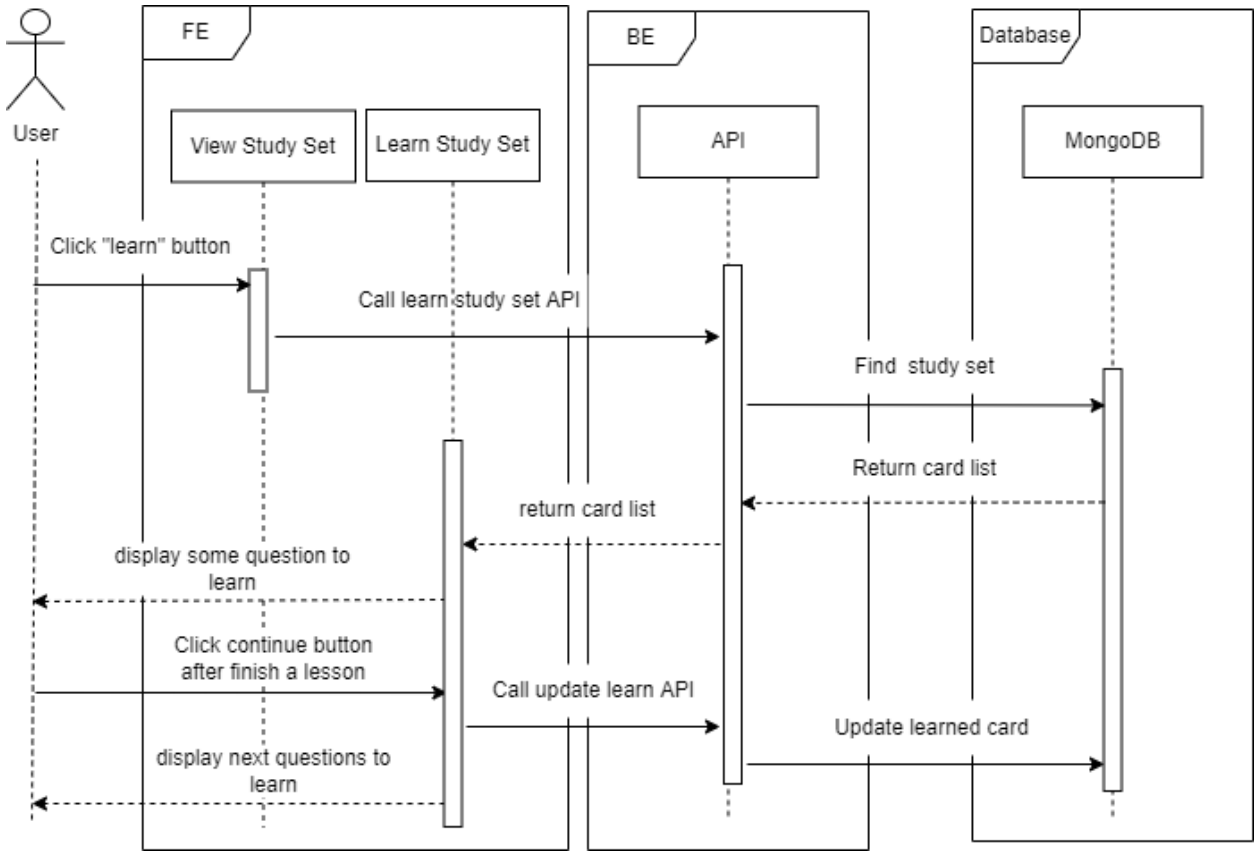
3.1.10. Add personal folder to class



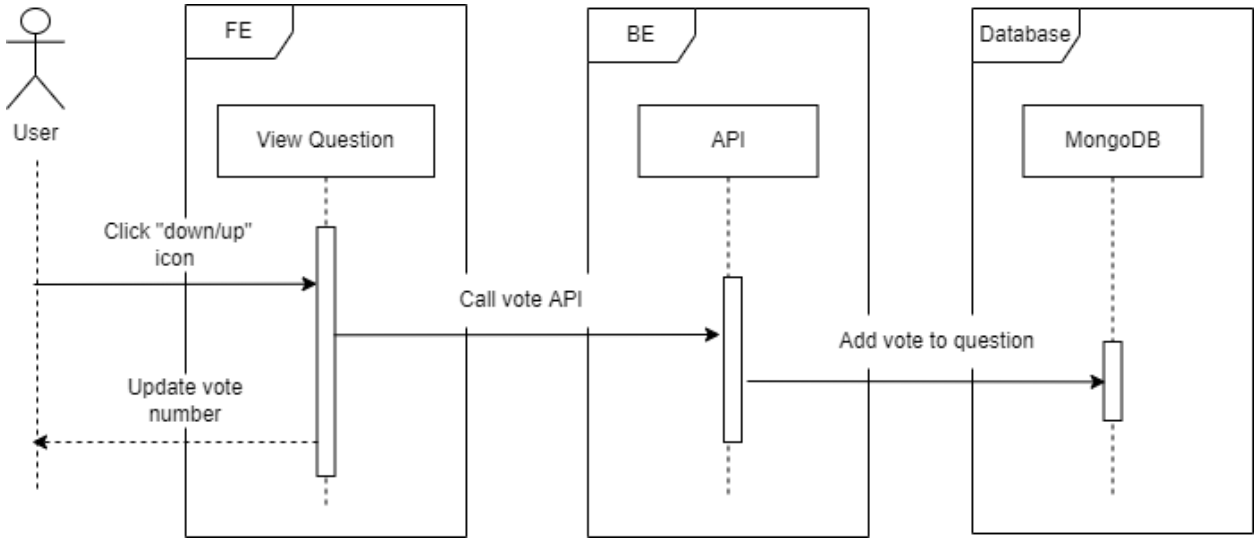
3.1.11. Save study set



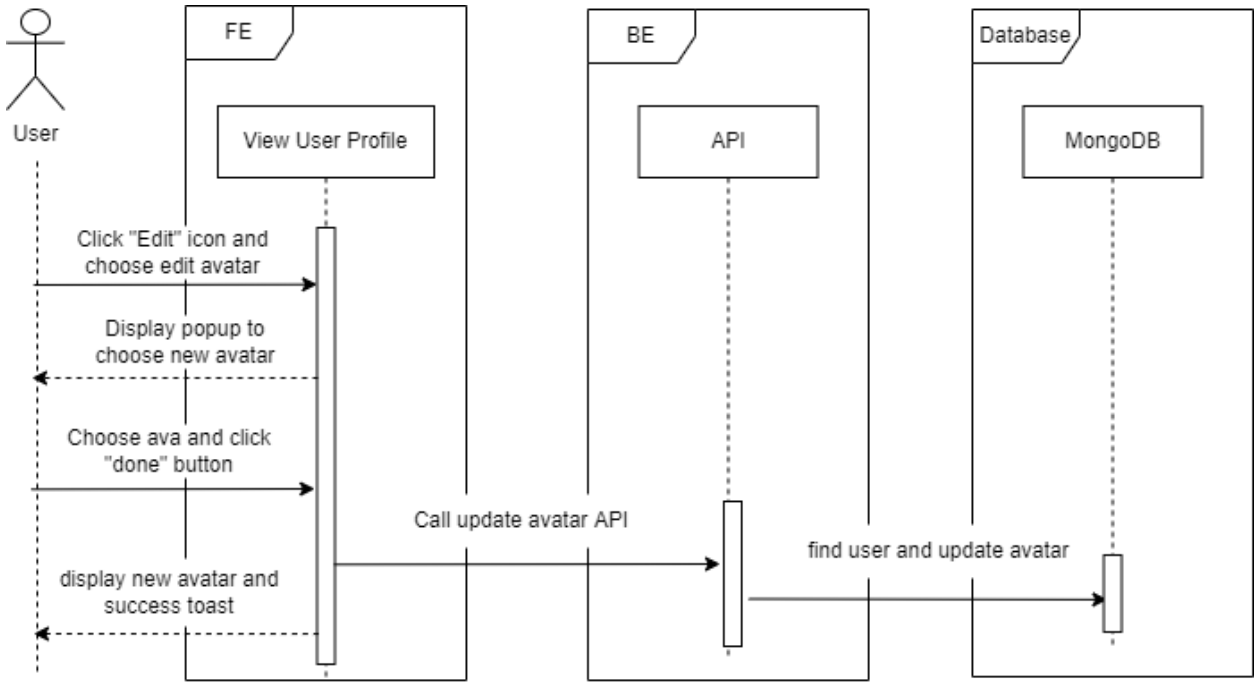
3.1.12. Learn study set



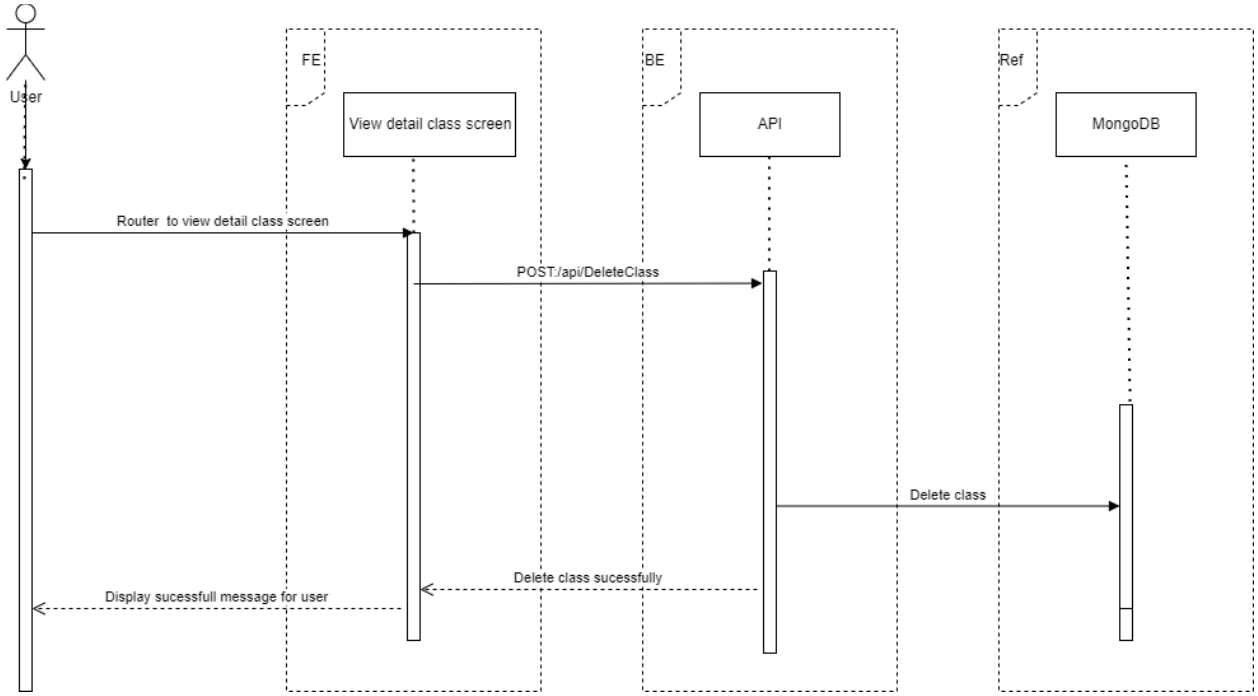
3.1.13. Up/down vote



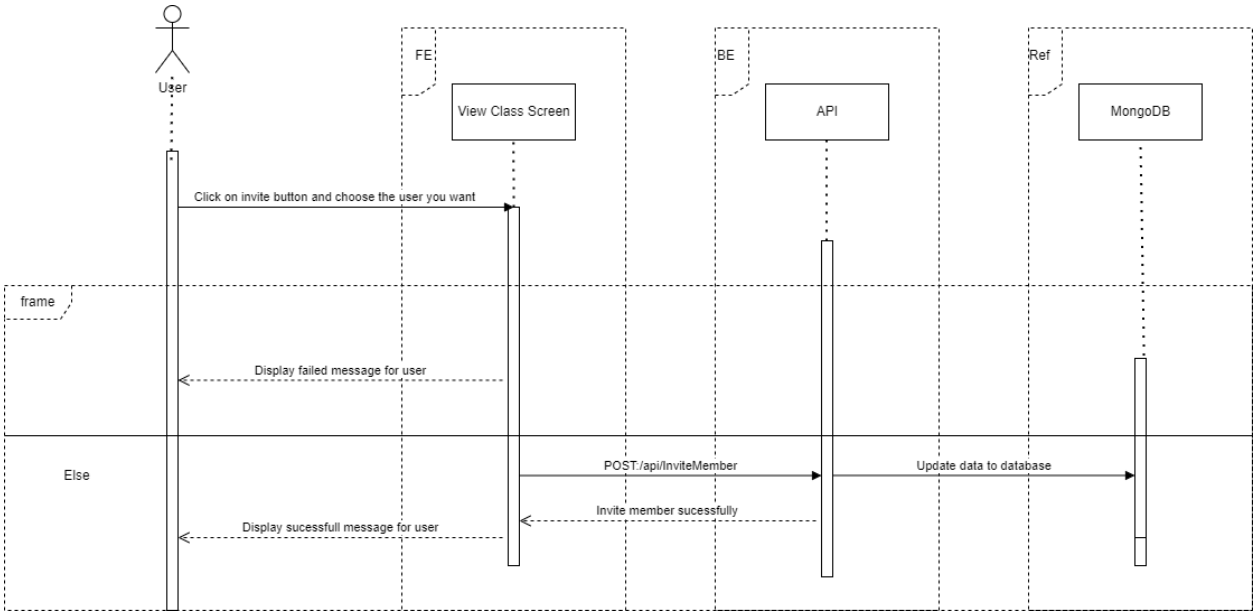
3.1.14. Update profile avatar



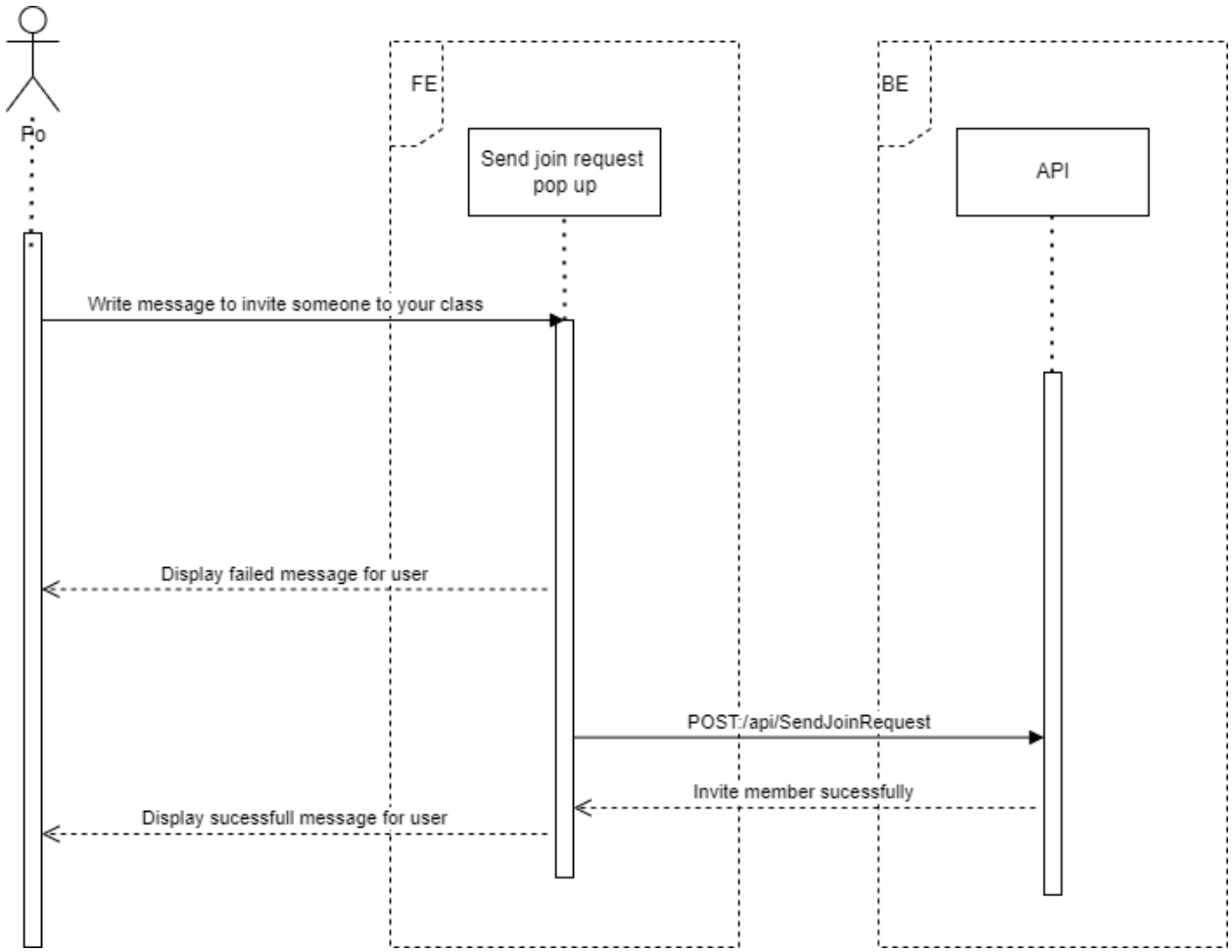
3.1.15. Delete class



3.1.16. Invite member



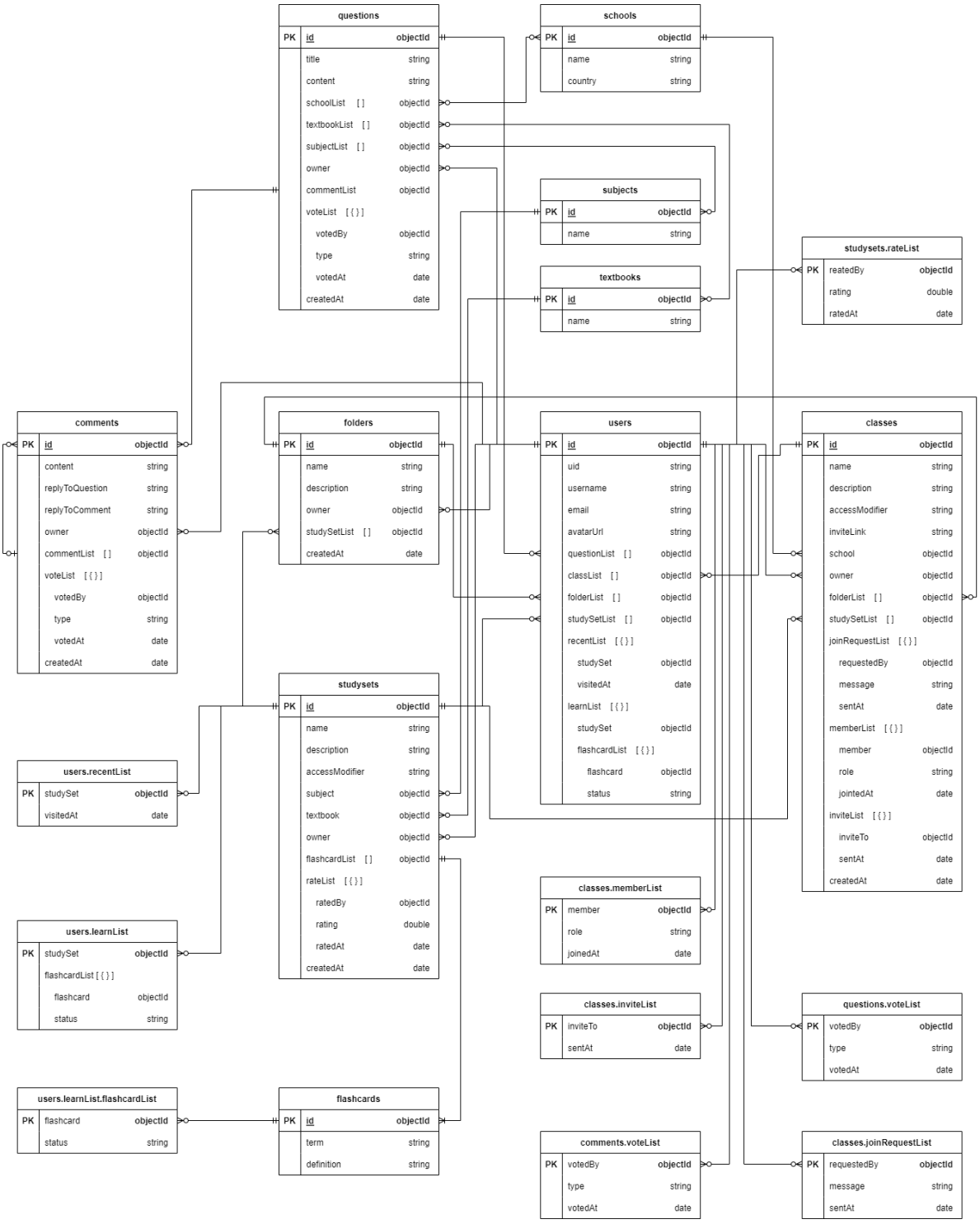
3.1.17. Send join request



4. Data & Database Design

4.1. Database Design

For all of the sequence diagrams' raw images, view them here.



4.1.1. Users Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of user (MongoDB)
2	uid	string	Yes	No		Id of user (firebase)
3	username	string	No	No		Name of username
4	email	string	Yes	No		Email of user
5	avatarUrl	string	No	No		Avatar of user
6	questionList	array of objectId	No	Yes		List question of user
7	classList	array of objectId	No	Yes		List class of user
8	folderList	array of objectId	No	Yes		List folder of user
9	studySetList	array of objectId	No	Yes		List study set of user
10	recentList	array of recentListType	No	Yes		List recent learn list of user
11	learnList	array of learnListType	No	Yes		List learn of user

4.1.2. Studysets Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of study set
2	name	string	No	No		Name of study set
3	description	string	No	Yes		Description of study set
4	accessModifier	string	No	No		Private or public this study set
5	subject	objectId	No	Yes		Name subject of study set
6	textbook	objectId	No	Yes		Name textbook of study set
7	owner	objectId	No	No		Owner of this study set
8	flashcardList	array of objectId	No	Yes		List flash card of study set
9	rateList	array of rateListType	No	Yes		Rate this study set
10	createdAt	date	No	No		Date time of this study set

4.1.3. Flashcards Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of flash card
2	term	string	No	No		Term of this flash card
3	definition	string	No	No		Definition of this flash card

4.1.4. Folders Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of folder
2	name	string	No	No		Name of this folder
3	description	string	No	Yes		Description of this folder
4	owner	objectId	No	No		Owner of this folder
5	studySetList	array of objectId	No	Yes		List study set of folder
6	createdAt	date	No	No		Date time of this folder

4.1.5. Classes Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of class
2	name	string	No	No		Name of class
3	description	string	No	Yes		Description of class
4	accessModifier	string	No	No		Private or public class
5	inviteLink	string	Yes	No		Share link class for member to view class
6	school	objectId	No	No		Choose school of class
7	owner	objectId	No	No		Owner of this class
8	folderList	array of objectId	No	Yes		List folder of class
9	studySetList	array of objectId	No	Yes		List study set of class
10	joinRequestList	array of joinRequestListType	No	Yes		List join request of class
11	memberList	array of memberListType	No	Yes		List member of class
12	inviteList	array of inviteListType	No	Yes		List invite to class
13	createdAt	date	No	No		Data time of this class

4.1.6. Schools Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of school
2	name	string	No	No		Name of school
3	country	string	No	No		Country of this school

4.1.7. Subjects Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of this subject
2	name	string	No	No		Name of this subject

4.1.8. Textbooks Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of textbook
2	name	string	No	No		Name of textbook

4.1.9. Questions Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of question
2	tittle	string	No	No		Title of question
3	content	string	No	No		Content of this question
4	schoolList	array of objectId	No	Yes		List school of question
5	textBookList	array of objectId	No	Yes		List textbook of question
6	subjectList	array of objectId	No	Yes		List subject of question
7	owner	objectId	No	No		Owner of this question
8	commentList	array of objectId	No	Yes		List comment of question
9	voteList	array of voteListType	No	Yes		List vote of question
10	createAt	date	No	No		Date time of this question

4.1.10. Comments Collection

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	id	objectId	Yes	No	PK	Id of comment
2	content	string	No	No		Content of comment
3	replyToQuestion	string	No	No		User reply to other question

4	replyToComment	string	No	Yes		User reply to other comment
5	owner	objectId	No	No		Owner of this comment
6	commentList	array of objectId	No	Yes		List comment of user
7	voteList	array of voteListType	No	Yes		List vote of user
8	createAt	date	No	No		Date time of comment

4.1.11. RecentListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	studySet	objectId	Yes	No	FK	Id of Study set
2	visitedAt	date	No	No		Time when user access to study set

4.1.12. LearnListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	studySet	objectId	Yes	No	FK	Id of this study set
2	flashcardList	array of learnFlashCardListType	No	No		List flash card of user

4.1.13. LearnFlashCardListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	flashcard	objectId	Yes	No	FK	Id of flash card
2	status	string	No	No		Status of flash card

4.1.14. MemberListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	member	objectId	Yes	No	FK	Id of member
2	role	string	No	No		Role of user
3	joinedAt	date	No	No		Time when user join to class

4.1.15. InviteListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	inviteTo	objectId	Yes	No	FK	Id
2	sentAt	date	No	No		Time when user is invited

4.1.16. VoteListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	votedBy	objectId	Yes	No	FK	Id of this vote
2	type	string	No	No		Up/down vote
3	votedAt	date	No	No		Time when user voted

4.1.17. JoinRequestListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	requestedBy	objectId	Yes	No	FK	Id
2	message	string	No	No		Write something to persuade user to join your class
3	sentAt	date	No	No		Time when user sent request

4.1.18. RateListType

No	Field name	Type	Unique	Nullable	FK/ PK	Note
1	ratedBy	objectId	Yes	No	FK	Id
2	rating	double	No	No		Rate
3	ratedAt	date	No	No		Time when user rated