<Alpha-SE>

<EBook4U>
Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
28/11/2022	1.0	Write Software Architecture Document	Vu Hoang Phuc

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Document provides an architectural overview of the Reading book web. The Website System is being developed by Alpha-SE to support reading books online.

This Document has been generated directly from the Reading Book Analysis & Design Model implemented in Alpha-SE. The majority of the sections have been extracted from the Thai Model using SoDA and the Software Architecture Document template.

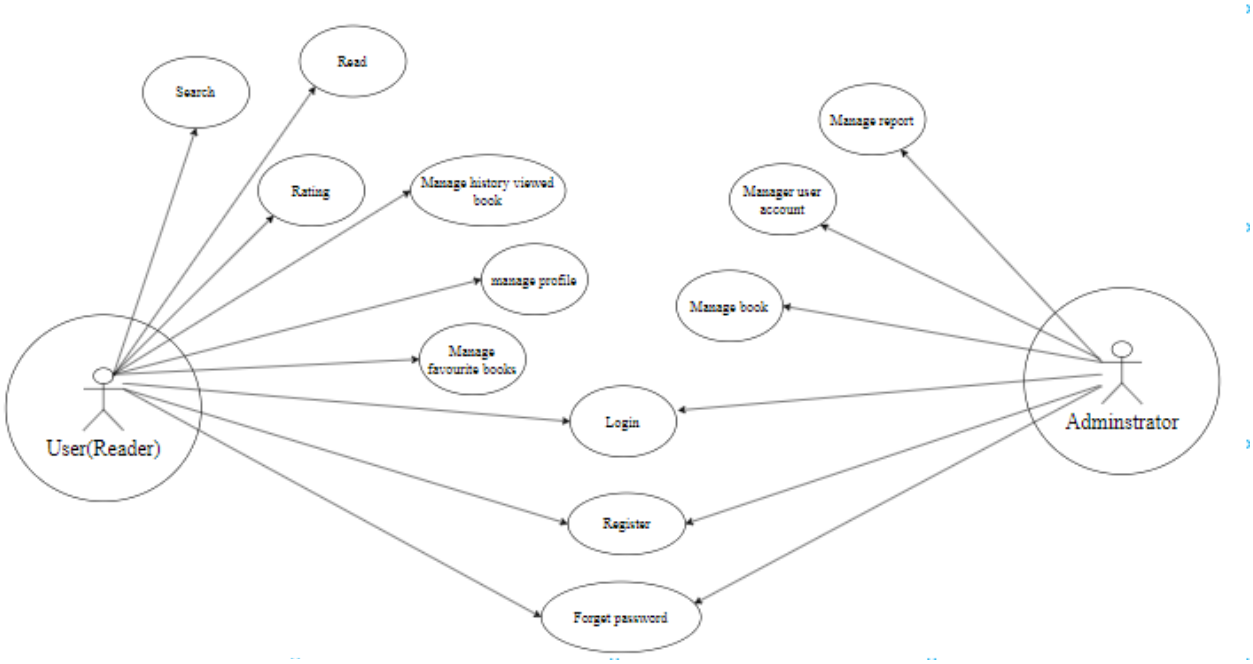
1.3 Acronym list:

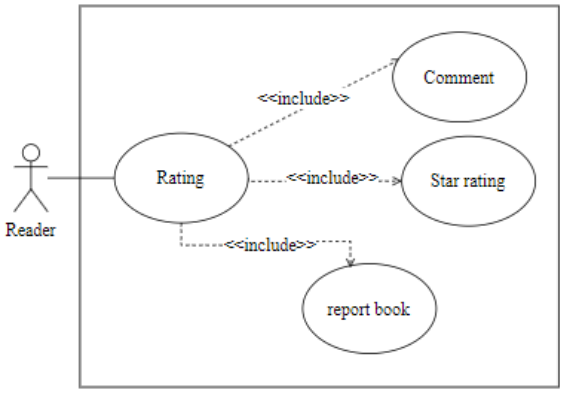
- SAD: Software Architecture Document.
- REST: Representational State Transfer, web API featuring a state-less client-server infrastructure.
- API: Application Programming Interface, a protocol used as an interface to allow communication between different components.
- MJPEG: Motion JPEG, a video format in which each frame is compressed as a JPEG image.
- CSS: Cascading-Style Sheets, document that describes the appearance of web pages.
- JSON: JavaScript Object Notation, a text-based standard for human-readable data exchange.
- MVC: Model-View-Controller, a software architecture pattern that separates the physical way to store data, the business logic and the appearance to the user.

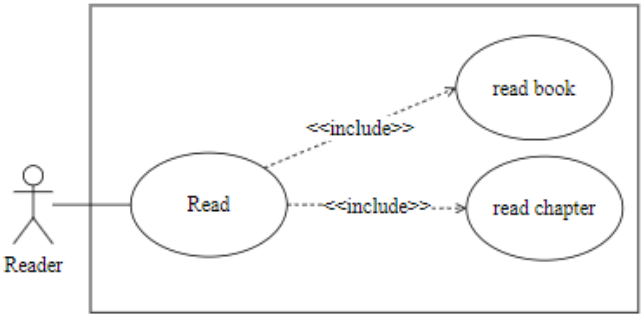
2. Architectural Goals and Constraints

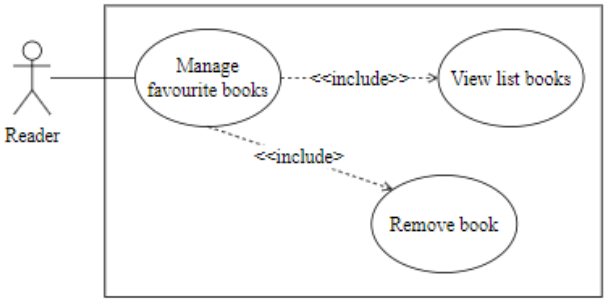
- Security for the account user
    + The password of users will be encrypted and stored in the database. Besides, ensuring the admin of the database doesn't know what password is
    + When the user logins, if the user types the password incorrectly more than five times, the website will lock the account and force the user to input the code to log in again
- To limit the leaking information of users outside, we will apply some method to control access to system because the data of the user will be stored in a cloud database
- Ensure the system still runs if it has an error by using try catch to catch the error
- Maintain system continuously
- Legal aspect, not existing any book or comment which has content 18+, violence,…
    + When book is posted, admin will check it and if it is valid, book will be accepted to display in website
- Loading well when a large number of people access the website at the same time to ensure the website does not lag. This enhances the user experience
- In term of front-end website will be coded by ReactJS, in term of back-end website will be coded by NodeJS and ExpressJS and in term of database website will use mongodb to store data.
- Environment of the application is web.
- Loading well when a large amount of people access to website at the same time.

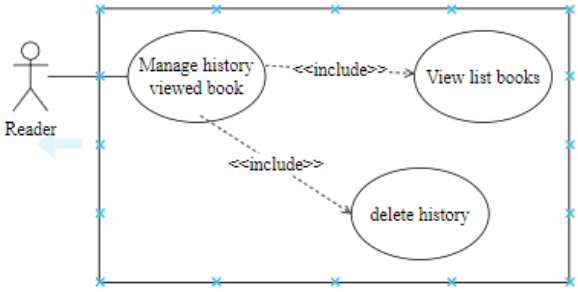
3. Use-Case Model

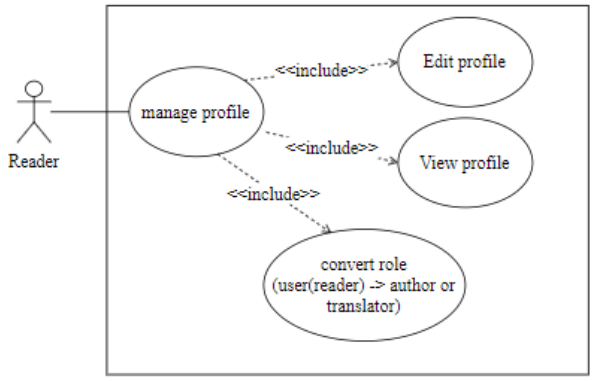

Search
Read
Rating
Manage history viewed book
manage profile
Manage favourite books
User(Reader)
Login
Register
Forget password
Manage report
Manager user account
Manage book
Adminstrator

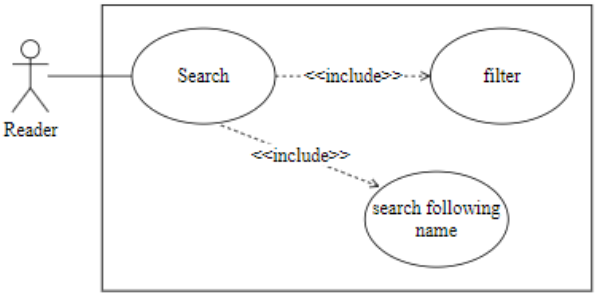

Reader
Rating
<<include>>
Comment
<<include>>
Star rating
<<include>>
report book

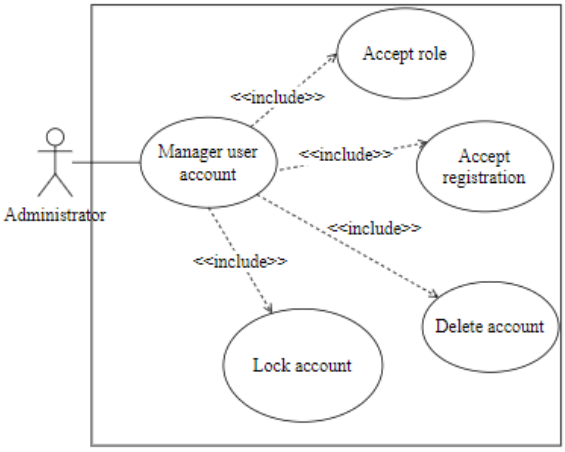

Reader
Read
<<include>>
read book
<<include>>
read chapter

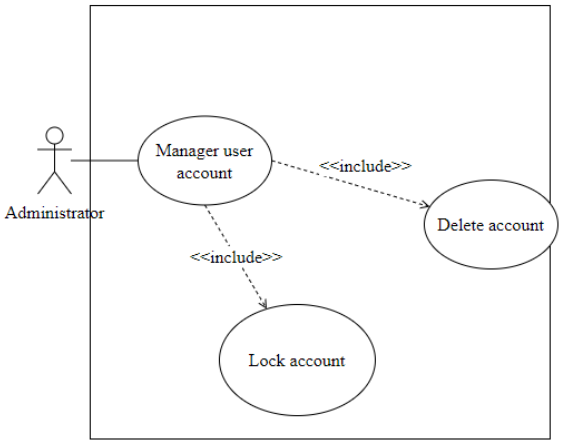

Reader
Manage favourite books
<<include>>
View list books
<<include>
Remove book

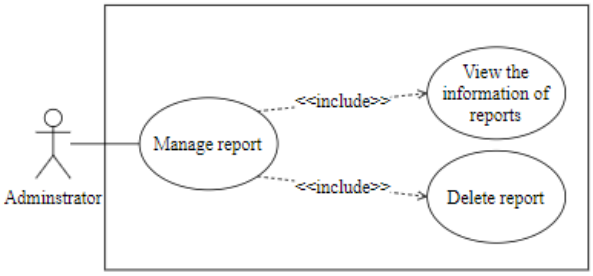

Reader
Manage history viewed book
<<include>>
View list books
<<include>>
delete history


Reader
manage profile
<<include>>
Edit profile
<<include>>
View profile
<<include>>
convert role
(user(reader) -> author or
translator)

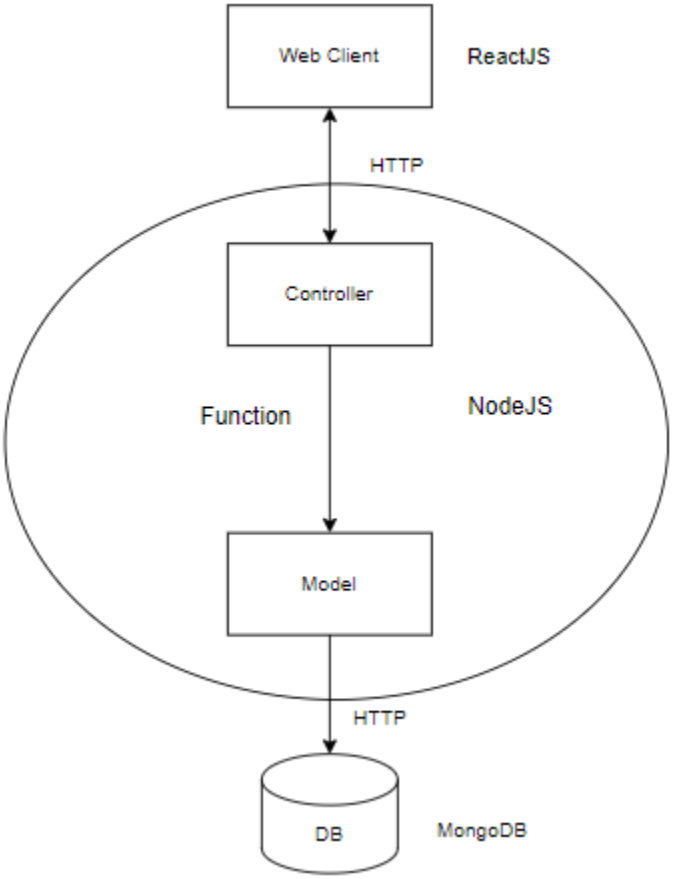

Reader
Search
<<include>>
filter
<<include>>
search following
name


Administrator
Manager user
account
<<include>>
Accept role
<<include>>
Accept
registration
<<include>>
Delete account
<<include>>
Lock account


Administrator
Manager user
account
<<include>>
Delete account
<<include>>
Lock account

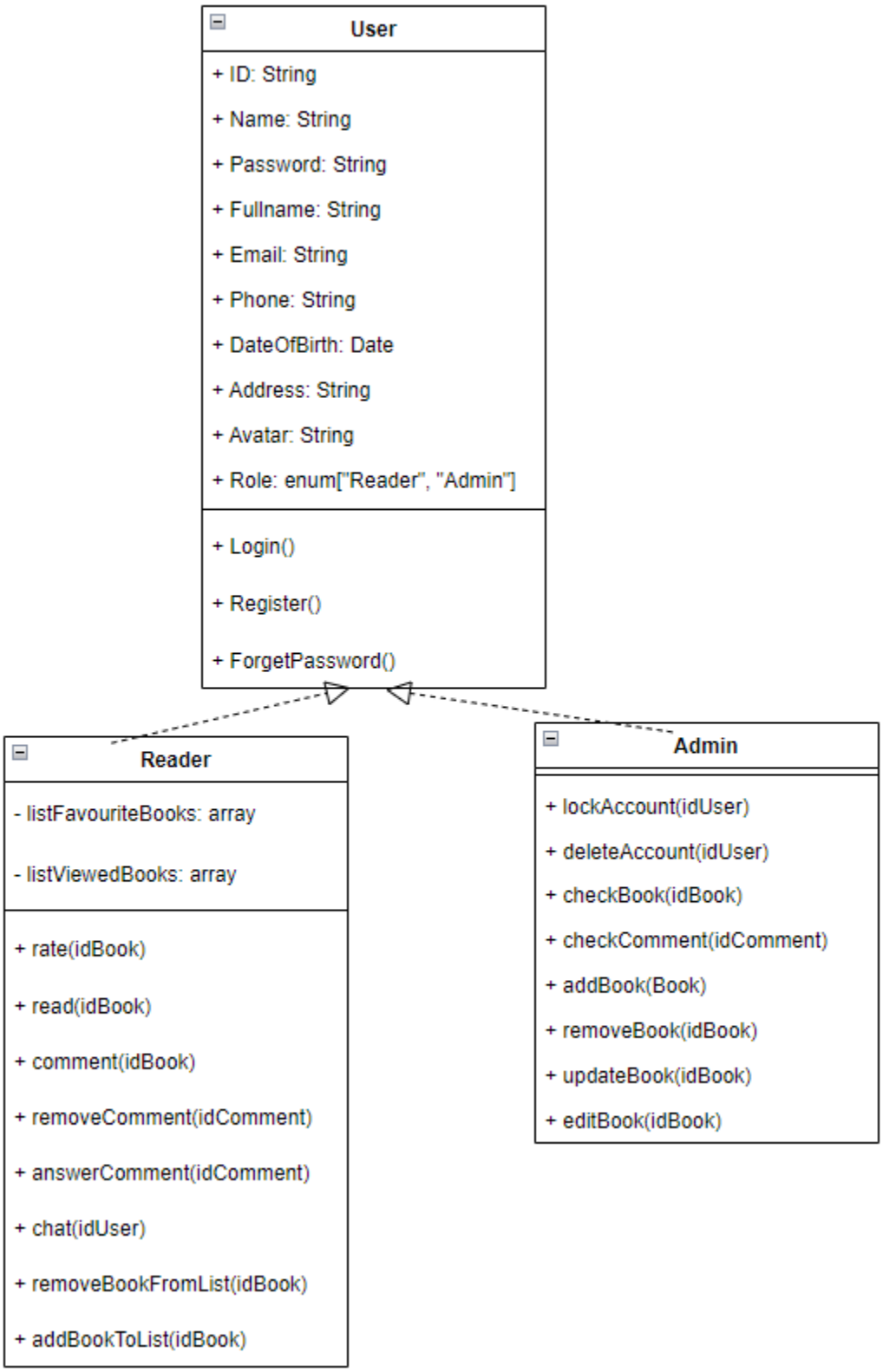

Adminstrator
Manage report
<<include>>
View the
information of
reports
<<include>>
Delete report

4. Logical View



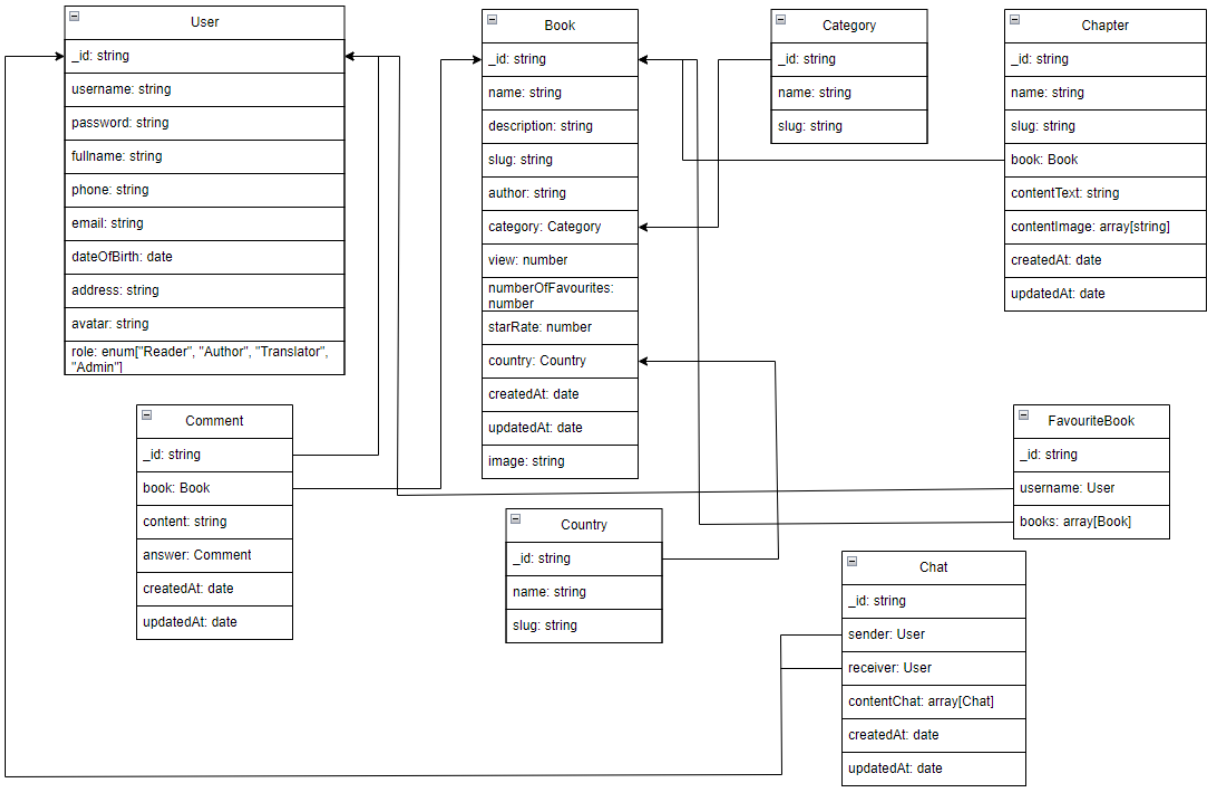
4.1 Client

+ Language: ReactJS
- Display the interface of main website for reading books, user detail page(including list of favorite books, information about the user,...), displaying books,...
- Request api to server in order to offer users required information.

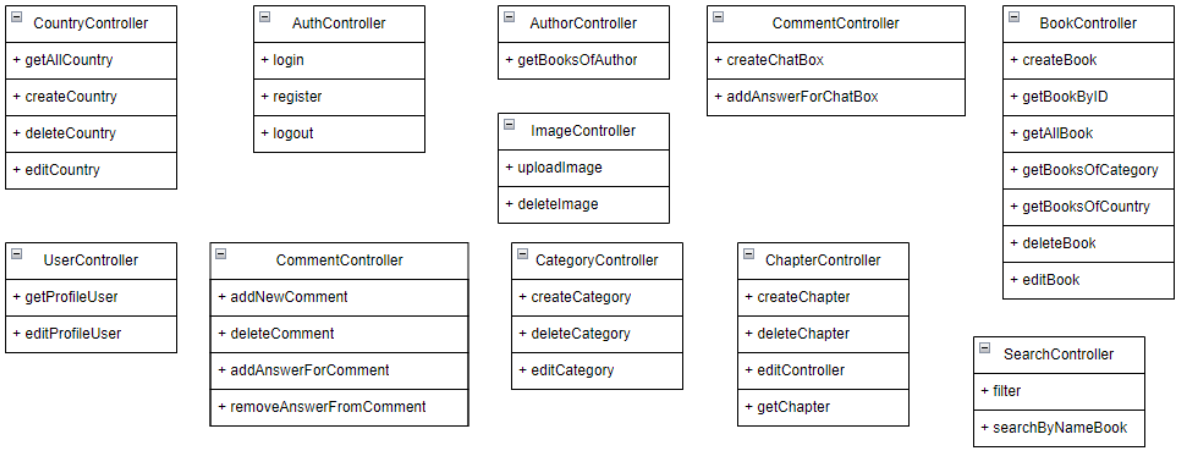

User
+ ID: String
+ Name: String
+ Password: String
+ Fullname: String
+ Email: String
+ Phone: String
+ DateOfBirth: Date
+ Address: String
+ Avatar: String
+ Role: enum["Reader", "Admin"]
+ Login()
+ Register()
+ ForgetPassword()
Reader
- listFavouriteBooks: array
- listViewedBooks: array
+ rate(idBook)
+ read(idBook)
+ comment(idBook)
+ removeComment(idComment)
+ answerComment(idComment)
+ chat(idUser)
+ removeBookFromList(idBook)
+ addBookToList(idBook)
Admin
+ lockAccount(idUser)
+ deleteAccount(idUser)
+ checkBook(idBook)
+ checkComment(idComment)
+ addBook(Book)
+ removeBook(idBook)
+ updateBook(idBook)
+ editBook(idBook)

4.2 Server

- **Language server:** Nodejs
- **Model:** storing data and logic processing calculations to solve problems that software cares about (business logic). The Component Model is usually displayed in Domain Model form. A model where to initialize objects for collections and init collection. The Model will interact with Database directly. In our server, The Model defines every field for every collection in Database.

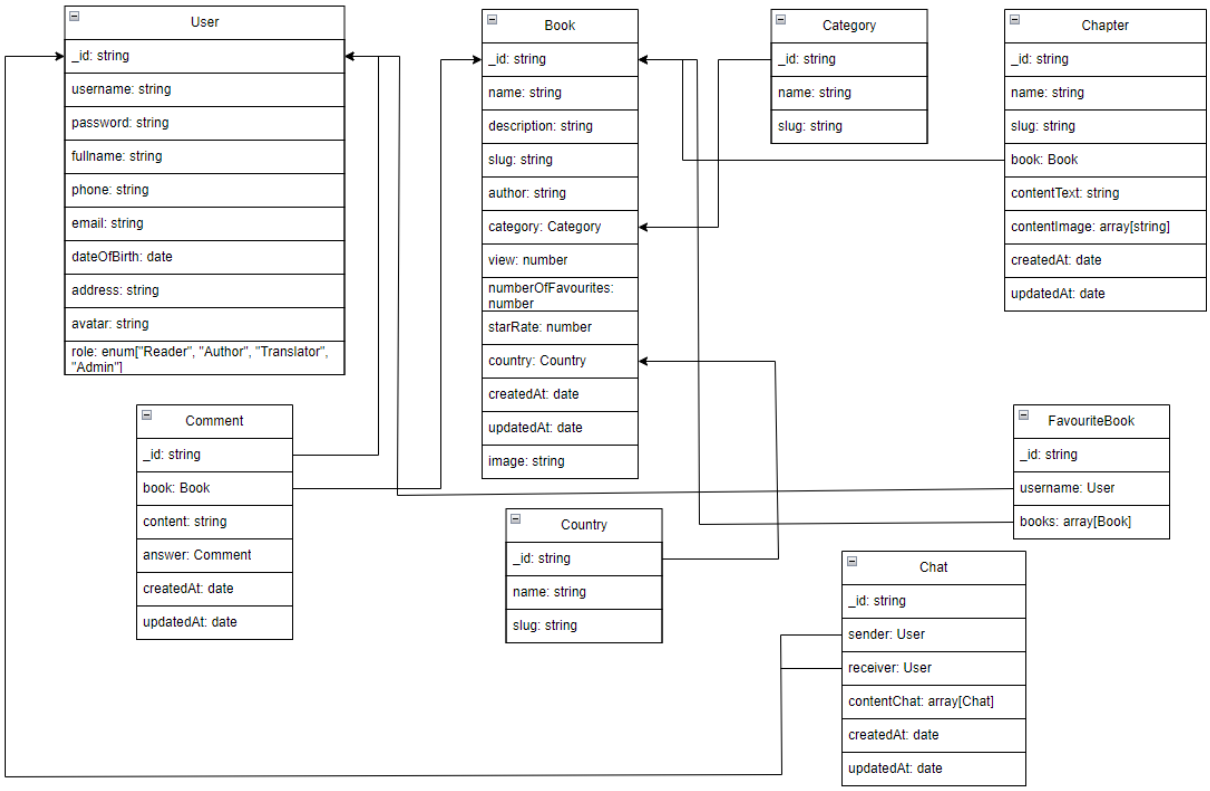


- **Controller:** is the component that takes responsibility for handling request API from Front-End/User to server and data posted by Front-End/User and client Controller is a bridge between users and application in the back-end. Accessing the Model's Data to get, put, delete and post data.



4.3 Database

- Database: MongoDB version cloud
- Data is stored in collection/document form.
- In MongoDB, there are 7 collections and every collection has many documents which has data
- 7 collections include:
    + users
    + books
    + categories
    + chapters
    + comments
    + countries
    + chats
    + favoriteBooks



5. Deployment

[Leave this section blank for PA3 if you are writing this document for PA4.

In this section, describe how the system is deployed by mapping the components in Section 4 to machines running them. For example, your mobile app is running on a mobile device (Android, iOS, etc), your server runs all components on the server side including the database]

6. Implementation View

[Leave this section blank for PA3 if you are writing this document for PA4.

In this section, provide folder structures for your code for all components described in Section 4.]