Software Design Document

MyRecipe webApp

Submitted to: Professor Alex Kuhn
April 23, 2020

Group Members

1, Rediet Negash: rediet.negash@stonybrook.edu
2, Cogitater Sigauke: cogitater.sigauke@stonybrook.edu
3, Merry Mekonnen: merry.mekonnen@stonybrook.edu
4, Pyungkang Hong: pyungkang.hong@stonybrook.edu

Content

Software Design Document

1. System Architecture

1.1 Overview

We are using Spring Boot 2.2, Azure Cosmos DB for MongoDB API, React 16.8, Bootstrap 3, Java as a programming tool, WebRCT(API), and libraries such as, Messaging libraries(JMS), Logging libraries, Unit-testing libraries, JSON parsing libraries. We will be using Spring Boot Framework on the backend and follow the MVC(Model View Controller) pattern.

The front end of our website will be purely built on React 16.8. React supports many core features which will be of great use for our website, such as component-based approach,backward compatibility, flexibility. At this point, we have a clear picture of the components we need for our website, and using React will allow us to design the UI components with a minimal effort as it follows a component-based approach. React also provides backward compatibility and flexibility which we believe our website will greatly benefit from. We have decided to use the latest version of React since many useful APIs are included such as Hooks. It is highly likely that we use the newly included APIs in React. In addition to React, we are using Bootstrap for the front end of our website for styling the pages. As Bootstrap has excellent documentation, it will save our time on styling the front-end. We will be using Bootstrap 3 in order for us to be able to use Glyphicon library.

The back end of our website will use Spring Boot 2.2 and Microsoft Azure Cosmos MongoDB. We decided to use Spring Boot because it is auto configured and does not need any manual configuration. It also manages Rest points easily. Similarly,MongoDB is simple to install and implement. It also provides high performance and automatic scaling.The libraries we decided to use may change later in the process, depending on the implementation of our code.

1.2 Components and interfaces

Major React Components for the frontend

App - root component

Home - extends from app and is parent for the following components

searchBar - it allows user to search for a meal either by meal name or Ingredient that it contains- (through ingredients filter). It will also have an auto-complete list to assist the user searching by giving suggestions.

NavBar - It has logo-link to home-, search bar and some important buttons like- Login, Logout, profile and addRecipe

Categories: it has the meal categories including the following
meal type:- it could be either breakfast, lunch, dinner or supper
dish type:- starter, main dish, sweets, salad, fruits, juice
diet and health:- healthy foods, diet based dishes
world Cuisine:- it has cultural foods from different countries

ContactUs - is a footer that has contact information including aboutUs, social media links, phone numbers, email address and other contact information.

Recipe- it lets the user add their own recipes

addRecipe:

Categories: lets the user add a recipe to a specific category by providing an option where the user can select from list of categories

Ingredients: this has a list of ingredients where the user can add or remove from the recipe that they are adding. It has division under it to make it easier for the user to quickly select the ingredients. Some of the divisions are dairy, meat, vegetables, fruits, spices, seasonings etc…

descriptionForm: It contains meal name, description box, steps with an option to upload images for each steps, uploadVideo button to upload video and submit button to complete the form and register the recipe

editRecipe: lets the owner of the recipe edit the recipe they already have.
favoriteRecipe: gives an option of tagging the recipe as a favorite recipe.
deleteRecipe: lets the owner delete the recipe that he/she added.

profile - this refers to the personal pages of a specific user

myRecipes - it has the recipes that are uploaded by the user so far with their ratings

myCategories: inherited from Recipe component to let the user filter their recipes by different categories

myReviews: it has lists of reviews given to each recipes with their ratings

myFavorites: it will have the list of favorite recipes that the user liked or added as favorite

Account:

Login - lets the user login with their google account using google auth.

Logout - lets the user logout from their account

Followers - lets the user see his/her followers and also the user can click on a block button to block the users following him/her

Following - let the user look at the list of the people that he/she is following.

editProfile- lets the user change password, username, or aboutMe description including their profile picture

Notification- it tracks news feeds and notifications and lets the user look at the list of notifications he/she has.

Settings: allows the user to deactivate their account or delete their account

chatBox - root component

Title- contains the user name

messagesList- it contains the list of messages the people chatted

sendMessageForm- this refers to the text box where the user types the message

We will have one index.html file where it is used to link to the generated JS components, and has that one single parent node for the React content to attach itself to.

1. 3 Third party Libraries and APIs

Spring Boot- this would basically serve as our backend server

React.js - we will use react for developing our frontend application

WebRCT - this is the API we will use for live streaming

Algolia Search Engine - This is a search engine that we will use for searching our ingredients and meals. It allows human language sentence searches as the algorithm so the user doesn't need to worry about entering a specific key for searching.

Google Auth- this is what we will use to authenticate a user when logging in. This would avoid any problem with account insecurity as google does it for us.

1.4 Detailed description for object oriented classes

(1) Recipe

- Attributes

Name	Type	Description
ID	String	The ID value will be the key that defines a single recipe.
LikeCount	Int	The number of likes that the recipe received
MealType	String	Is a category that describes the type of the meal which could be breakfast, lunch, dinner or supper
DietAndHealth	String	It is another category of a meal that contains specific type of meals that are healthy and diet specific
WorldCuisine	String	This category refers to cultural foods from different parts of the world
MealName	String	Refers to the name of the meal that is entered when the Recipe is added to the database
Description	String	Refers to the description entered by the user when the recipe is created and iis added to the database
favoritedBy	ArrayList<User>	This contains a list of users that liked or added this recipe to their favorite list.

-RecipeRepository: An interface to store,edit,and delete a recipe.

Name	Parameters	Return values	Description
delete	Recipe	void	Deletes Recipe from the Database
save	Recipe	void	Saves Recipe to the Database
find	recipeId	Recipe	Fetches Recipe with the given ID
update	recipeId	Recipe	Updates the Recipe

(2) GeneralUser<Interface>

- Methods

Name	Type	Description
getID	String	Describes the user’s Id
getName	String	Has user's username
getProfilePic	Image	Has user's profile image which is an Image object
receiveNotification	Notification	Updates the user with new information regarding the user's account.

(3) AdminUser

- Attributes

Name	Type	Description
ID	String	Describes the user’s Id
name	String	Has user's username
profilePic	Image	Has user's profile image which is an Image object

(4) User

- Attributes

Name	Type	Description
ID	String	Describes the user’s Id
name	String	Has user's username
profilePic	Image	Has user's profile image which is an Image object
aboutMe	String	Describes user's 'about me' information
userStatus	AccountStatus	Shows the account status either deactivated or normal
Followers	ArrayList<User>	Has a list of users who are following the user
Following	ArrayList<User>	Has a list of users whom the user follow
BlockedByUsers	ArrayList<User>	Has a list of users who are blocked by the user

- The User Repository - An interface for the interaction with the user model and performing database.

Name	Parameters	Return values	Description
delete	user	void	Deletes user from the Database
save	user	void	Saves user to the Database
find	userId	User	Fetches user with the given ID
update	userId	User	Updates the user

(5) Message

- Attributes

Name	Type	Description
ID	String	ID value that uniquely identifies the message object.
sender	User	the user by whom the message is sent
receiver	User	the user to whom the message is sent
messageText	String	has the message text to be sent to the receiver
readBoolean	Boolean	shows if the message is read by the receiver or not
image	Image	has the image to be sent to the receiver

- The Message Repository

Name	Parameters	Return	Description
editMessage	User	void	is used to edit the text message to user after the message being sent
deleteMessage	User	void	is used to delete a text message which is already sent
sendMessage	User	void	used to send a message to the receiver

(6) Notification

- Attributes

Name	Type	Description
ID	String	An id that uniquely identifies a notification object
notificationText	String	The attribute holds the notification text description
readBoolean	Boolean	This boolean would help to track notifications that are already read and the ones that are not visited yet

-Notification Rository

Name	Parameters	Return values	Description
delete	notification	void	Deletes notification from the Database
save	notification	void	Saves notification to the Database
find	notificationId	User	Fetches notification with the given ID
update	notificationId	User	Updates the notification in the database

(7) Comment

- Attributes

Name	Type	Description
ID	String	The comment id helps to uniquely identify comments
likesCount	Int	The number of likes that the comment received
recipe	Recipe	Refers to the recipe that the comment is given to
commentLikedBy	User	Refers to the user who liked the comment

-The Comment Repository

Name	Parameters	Return	Description
save	Comment	void	addComment adds the user's comment given to a specific recipe
delete	Comment	void	Delete method is used to delete a text comment which is already sent
update	Comment	void	editComment method is used to edit the text Comment to user after the comment is posted

8) Review

- Attributes

Name	Type	Description
ID	String	The Id would uniquely identify the review object
recipe	Recipe	This is the recipe that is being reviewed or rated
reviewGiven	Int	This refers to the review that the recipe has received. It is the number of starts that the commenter gave while rating

-The Review Repository

Name	Parameters	Return	Description
save	Review	void	Save review in the database
find	reviewId	void	Fetches a review from the database given the ID
update	Review	void	Updates a review in database

9) Image

- Attributes

Name	Type	Description
ID	String	The id uniquely identifies the image object
imgCap	String	This has a caption of the image

-Image Repository

Name	Parameters	Return values	Description
delete	image	void	Deletes image from the Database
save	image	void	Saves image to the Database
find	imageId	Image	Fetches image with the given ID
update	imageId	Image	Updates the image in the database

10) Video

- Attributes

Name	Type	Description
ID	String	The ID value will be the key that defines a user
likesCount	String	The number of like that the video received
videoName	String	The user input of the videoName from the User use case
videoCount	Int	The number of views that the video has

-Video Repository

Name	Parameters	Return values	Description
delete	video	void	Deletes a video from the Database
save	video	void	Saves video to the Database
find	videoId	Video	Fetches video with the given ID
update	videoId	Video	Updates the video in the database

1.3 UML Class Diagram

https://app.diagrams.net/#G1r1xh5914B5bmROgwju_OvgGK0r-5If2y

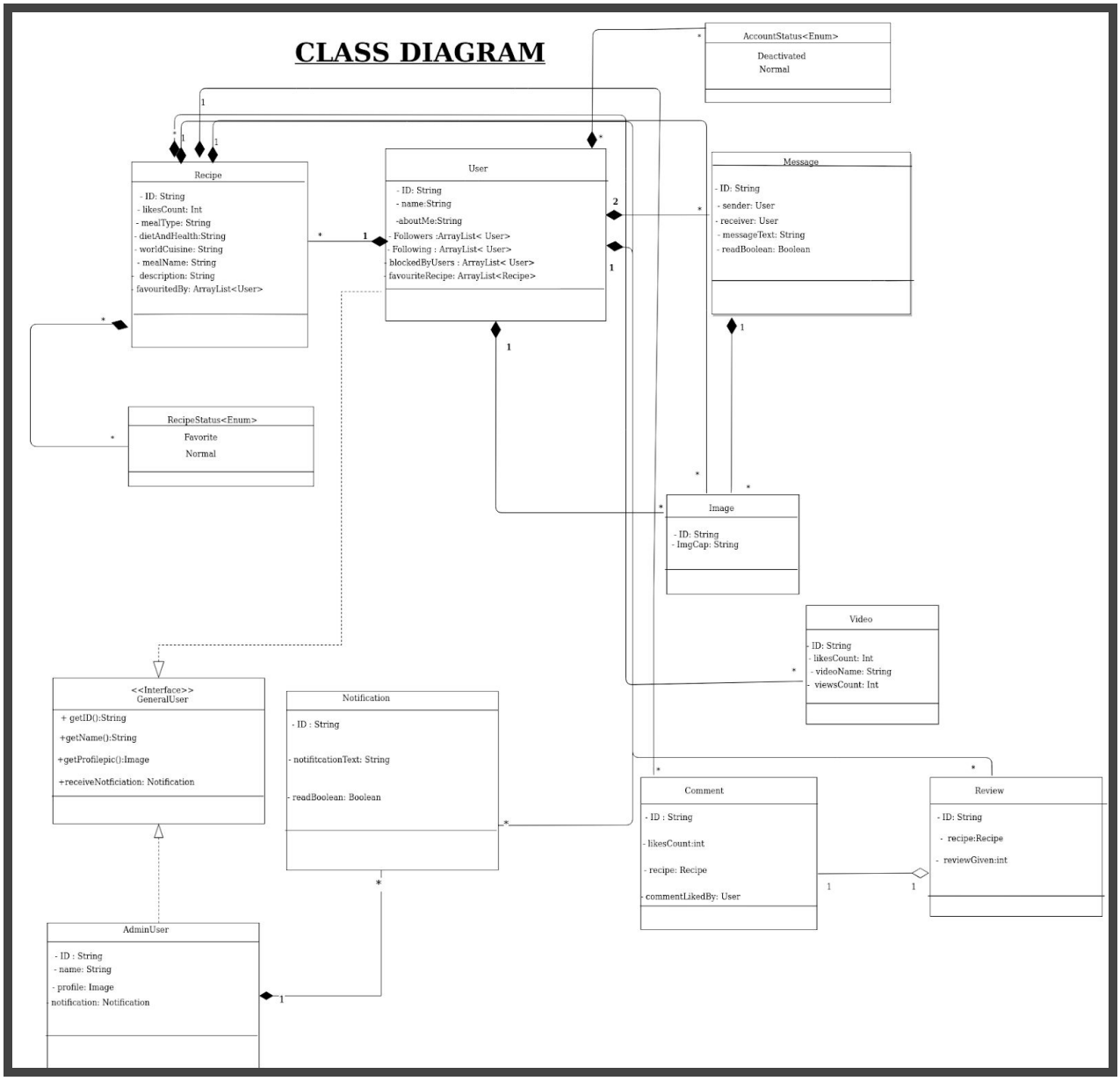


Figure 1.2 User and Recipe class

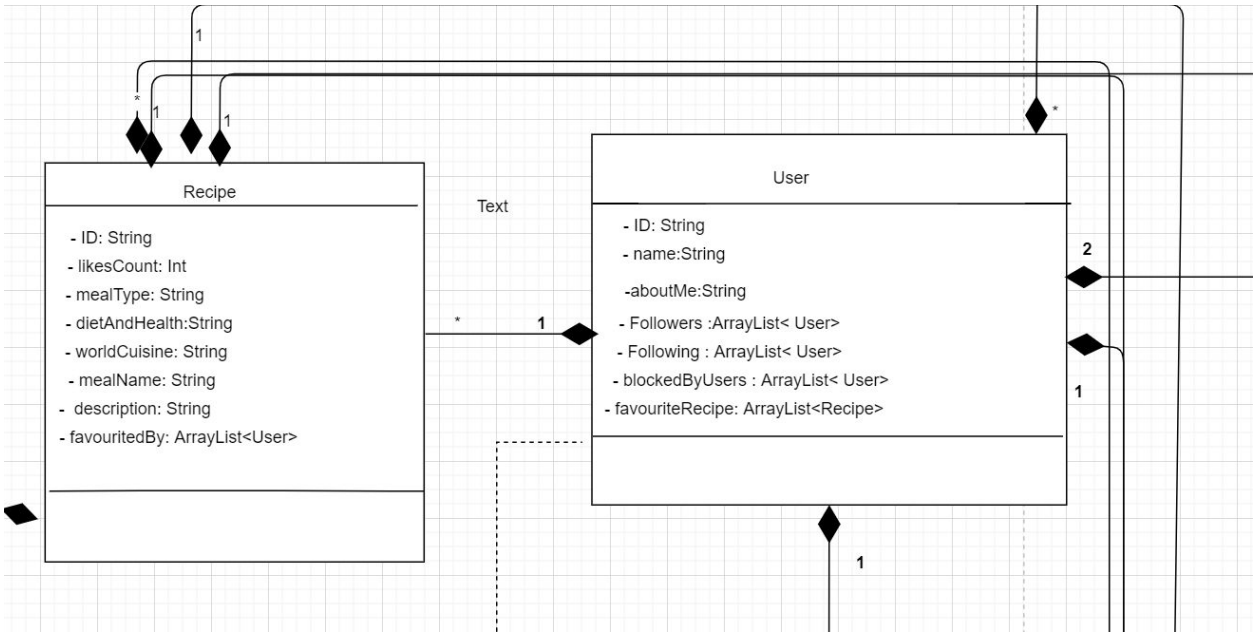


Figure 1.2 AccountStatus Enum and Message classes

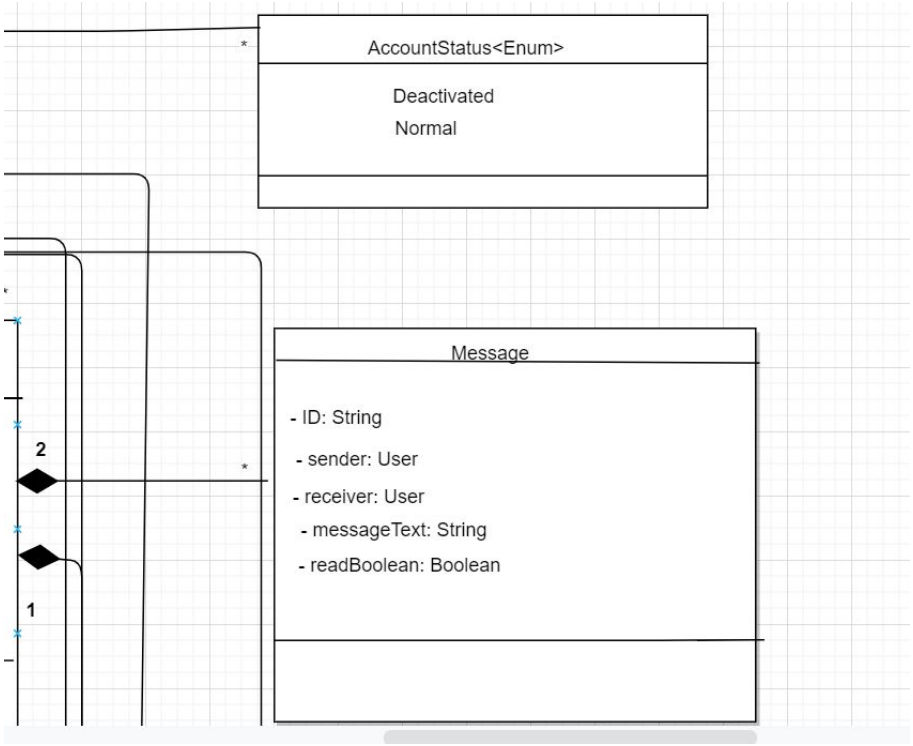
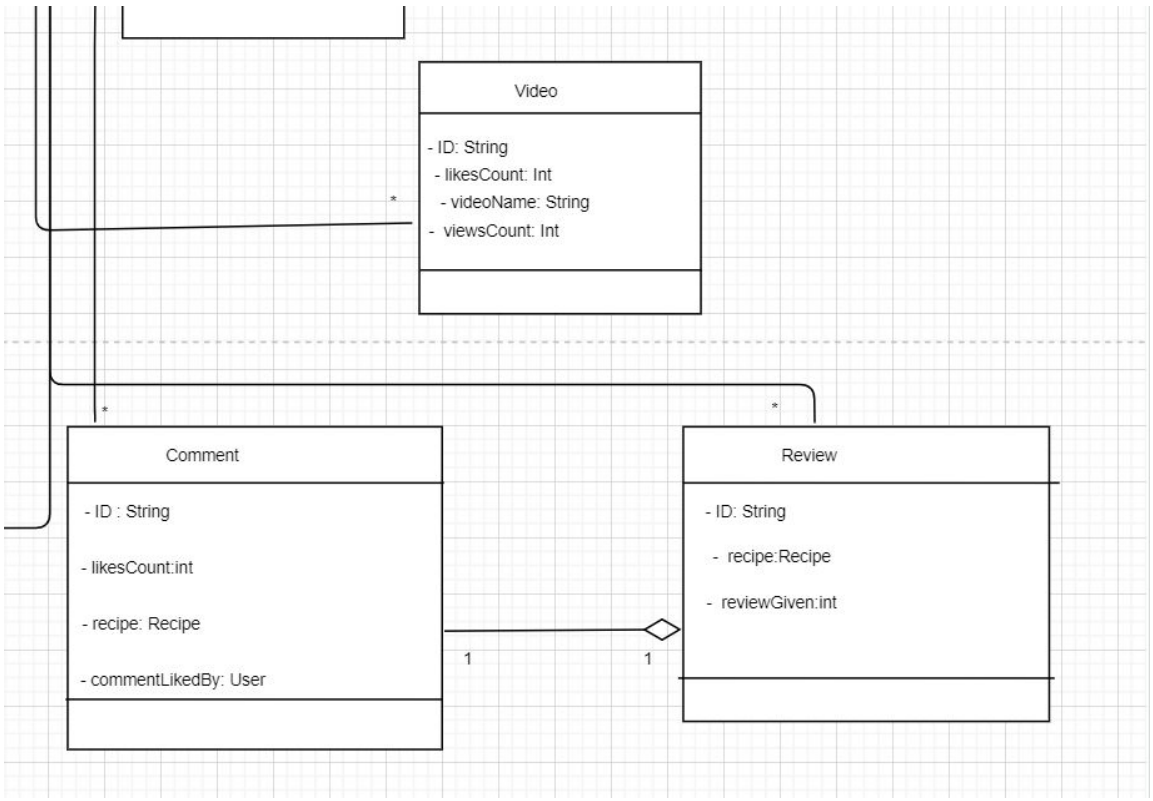


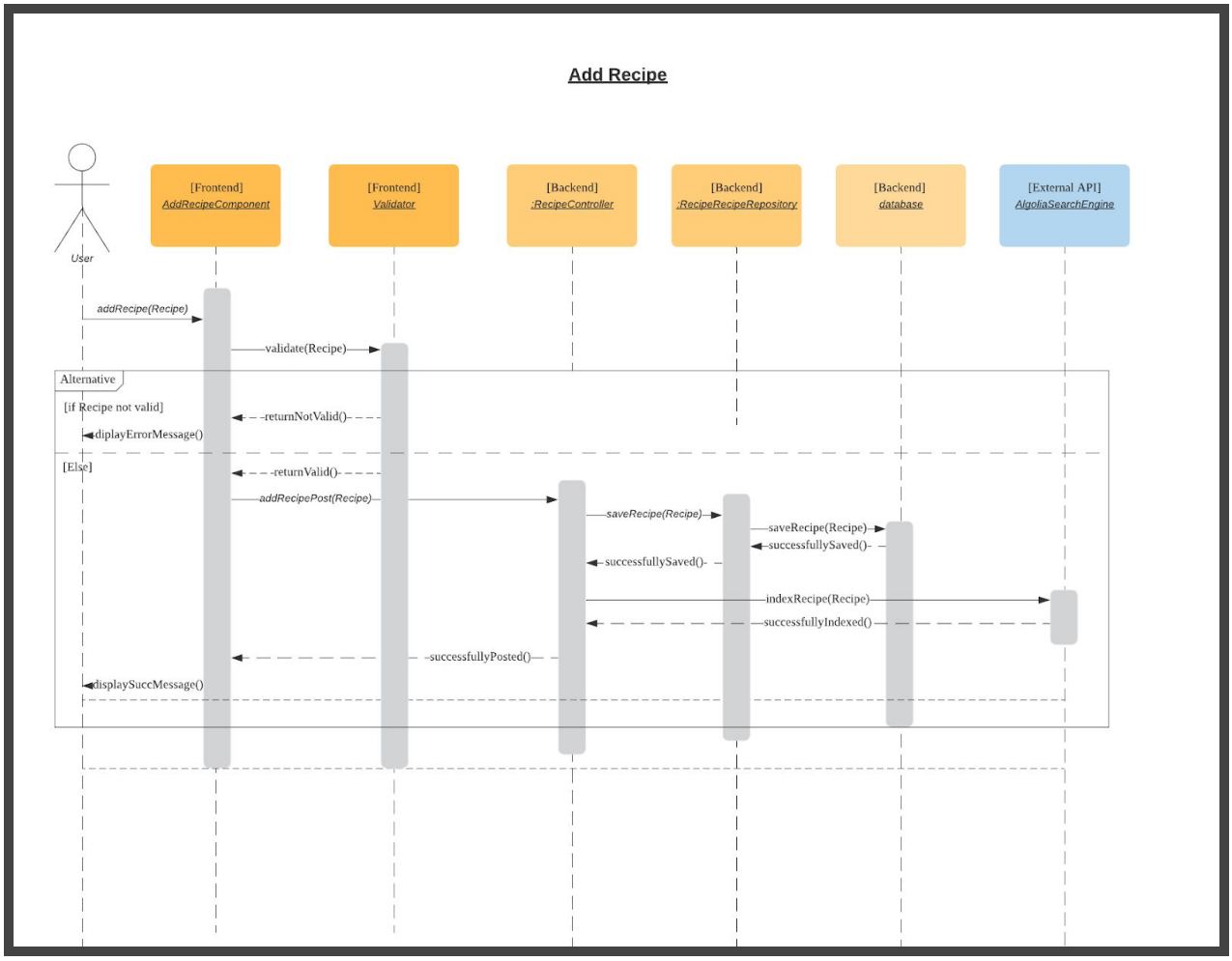
Figure 1.3 video, comment and Review classes



1.4 UML Sequence Diagrams

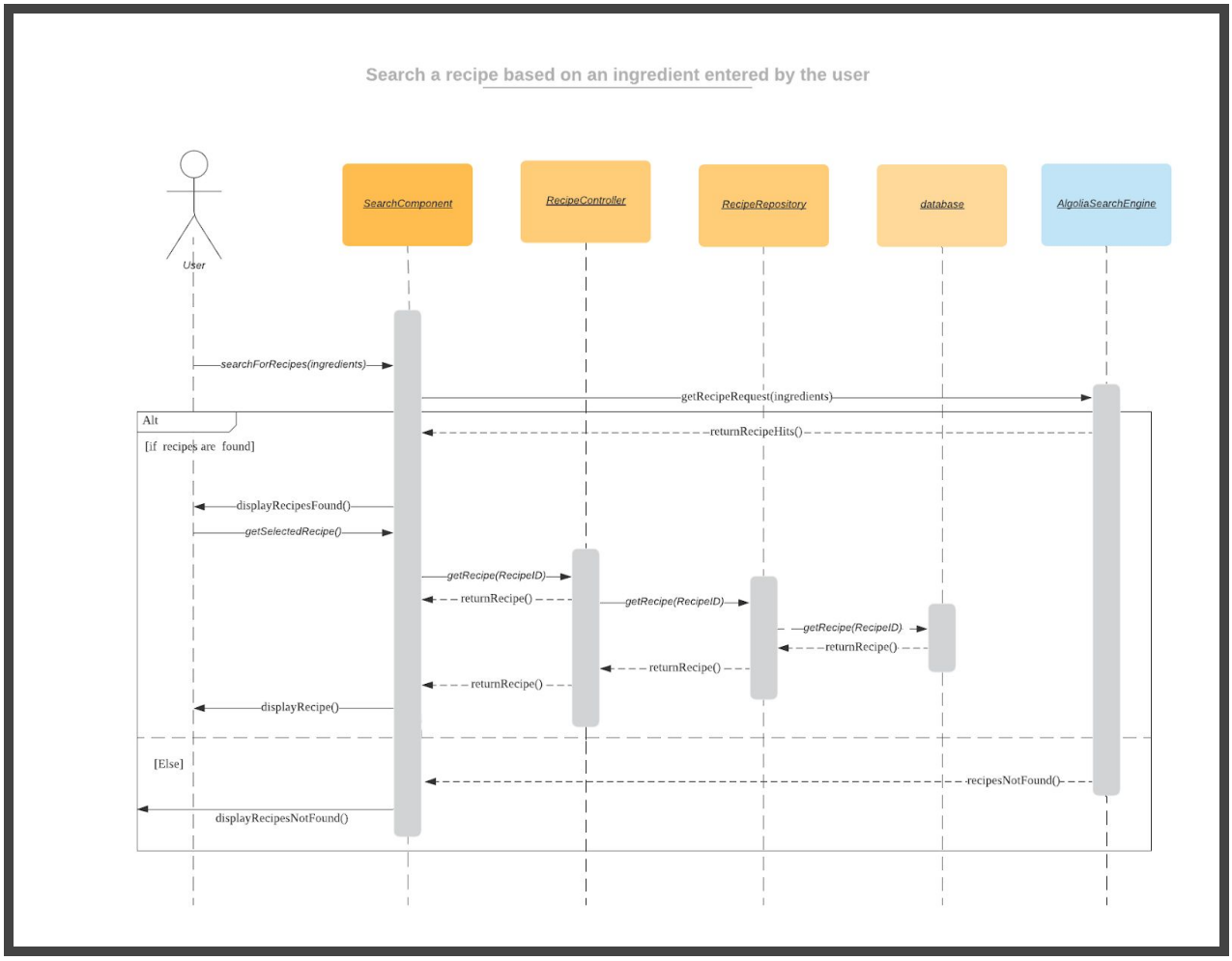
1.4.1 ADD Recipe

https://www.lucidchart.com/documents/edit/daa98cb7-cfb7-4c9d-8ab7-2d4912b9c1ca/0_0



1.4.2 Search a recipe based on an ingredient entered by the user

https://www.lucidchart.com/documents/edit/ef294250-de7b-4e97-9052-439695208515/0_0


Search a recipe based on an ingredient entered by the user
User
SearchComponent
RecipeController
RecipeRepository
database
AlgoliaSearchEngine
searchForRecipes(ingredients)
getRecipeRequest(ingredients)
returnRecipeHits()
Alt
[if recipes are found]
displayRecipesFound()
getSelectedRecipe()
getRecipe(RecipeID)
returnRecipe()
getRecipe(RecipeID)
getRecipe(RecipeID)
returnRecipe()
returnRecipe()
returnRecipe()
displayRecipe()
[Else]
recipesNotFound()
displayRecipesNotFound()

1.5 Deployment:

We will be using Microsoft Azure for the full development of our application. For this, we will be using the Azure Cosmos DB for MongoDB API as our database. In addition, both the backend spring boot application and the react front end of our application will be deployed to Microsoft Azure.

1.6 Alternatives:

We will be using MongoDB for our database with MySQL as a second option. Similarly, We plan to use Google Auth for signing in(authentication and verification). However, depending upon the users' experience, we might include account creation features rather than simply using Google Auth.

2. Schedule

https://trello.com/b/OUJ2p6AO/schedule