



SOFTWARE DESIGN DOCUMENT

**Presentation by Cogitater Sigauke,
Merry Mekonnen,
Redi Negash,
Pyungkang Hong**



MyRecipes

A recipe sharing web application

A recipe search web application

An application develop your cooking skills

An application place to meet new friends

A place to be inspired



Problem

Some cooks are eager to share their recipes
People don't know how to cook

and Technologies

Spring Boot

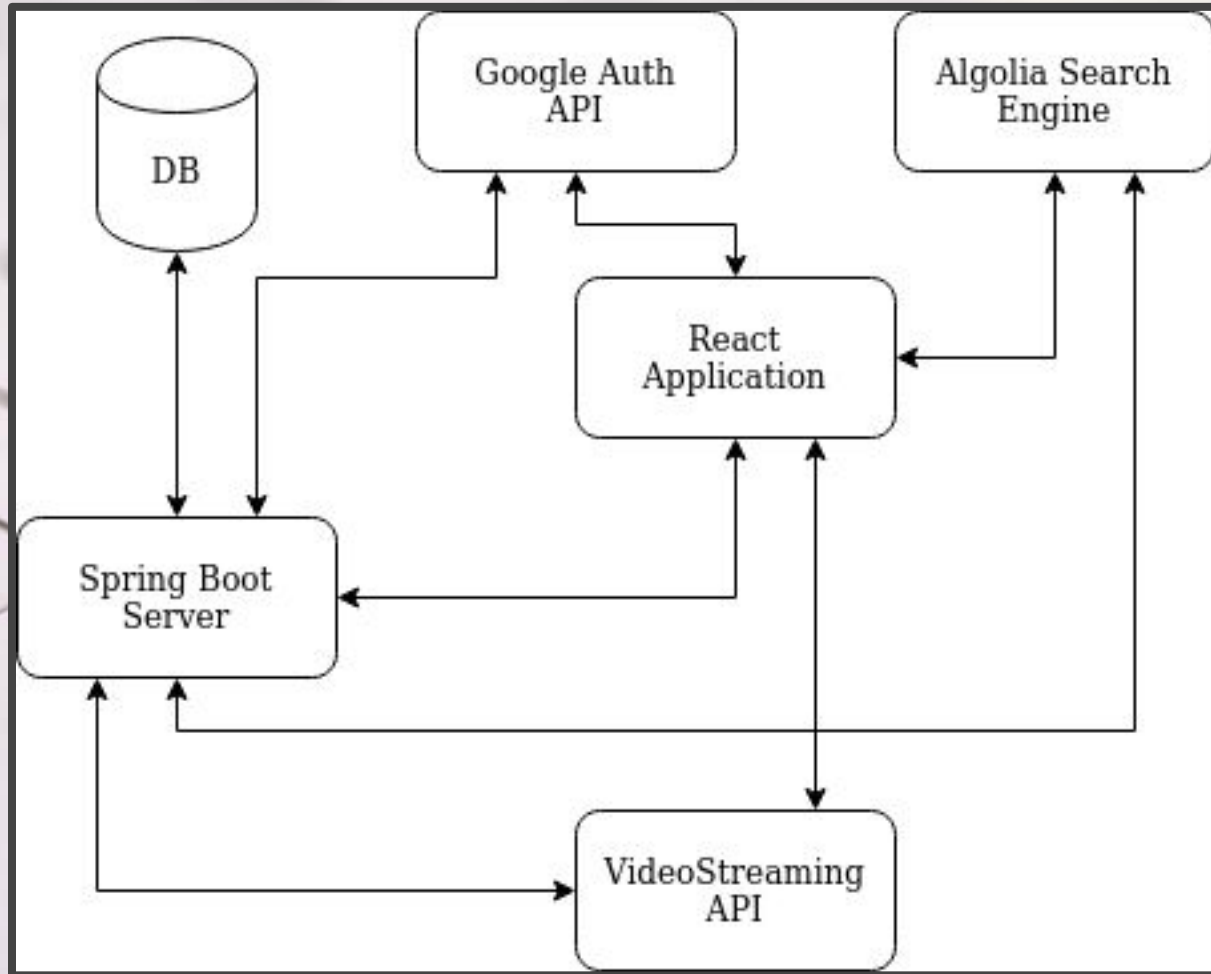
React.js

Algolia Search Engine

WebRCT

Google Auth

General Architecture View





Major Features

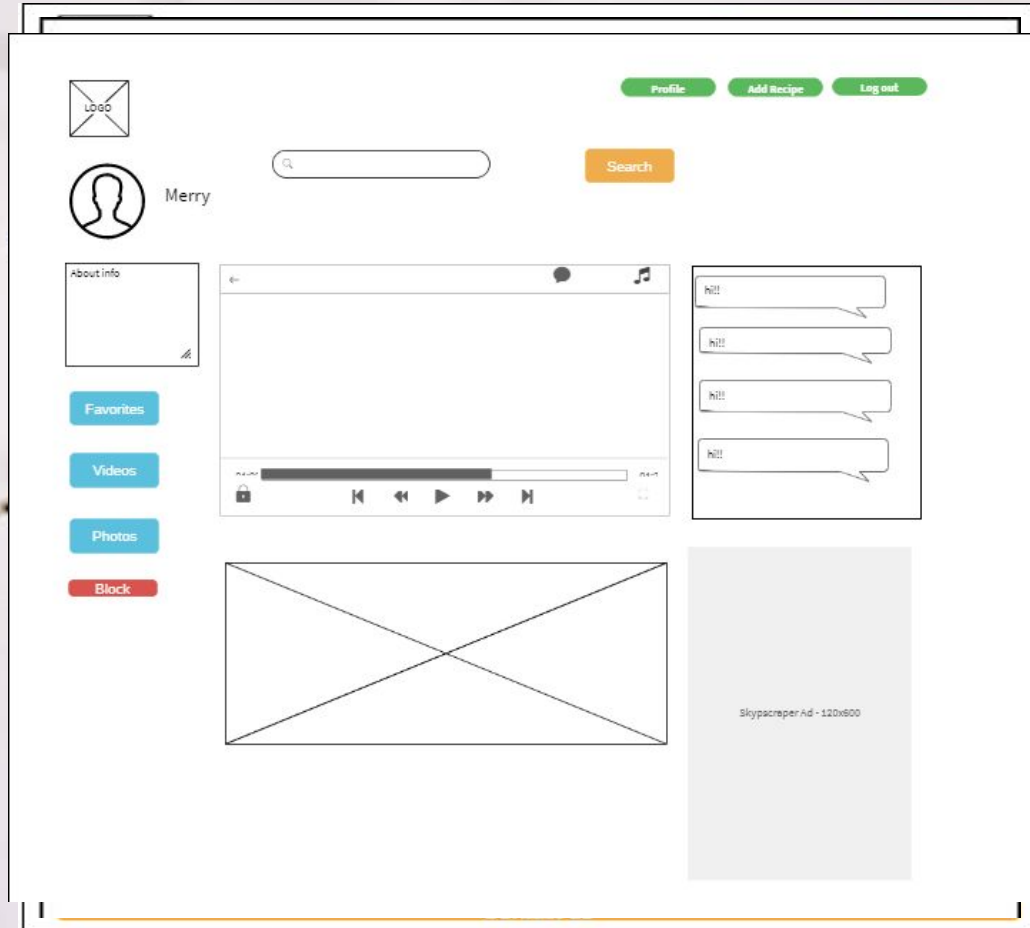
General Text Based Search with different filters

Display of uploaded recipes in different categories

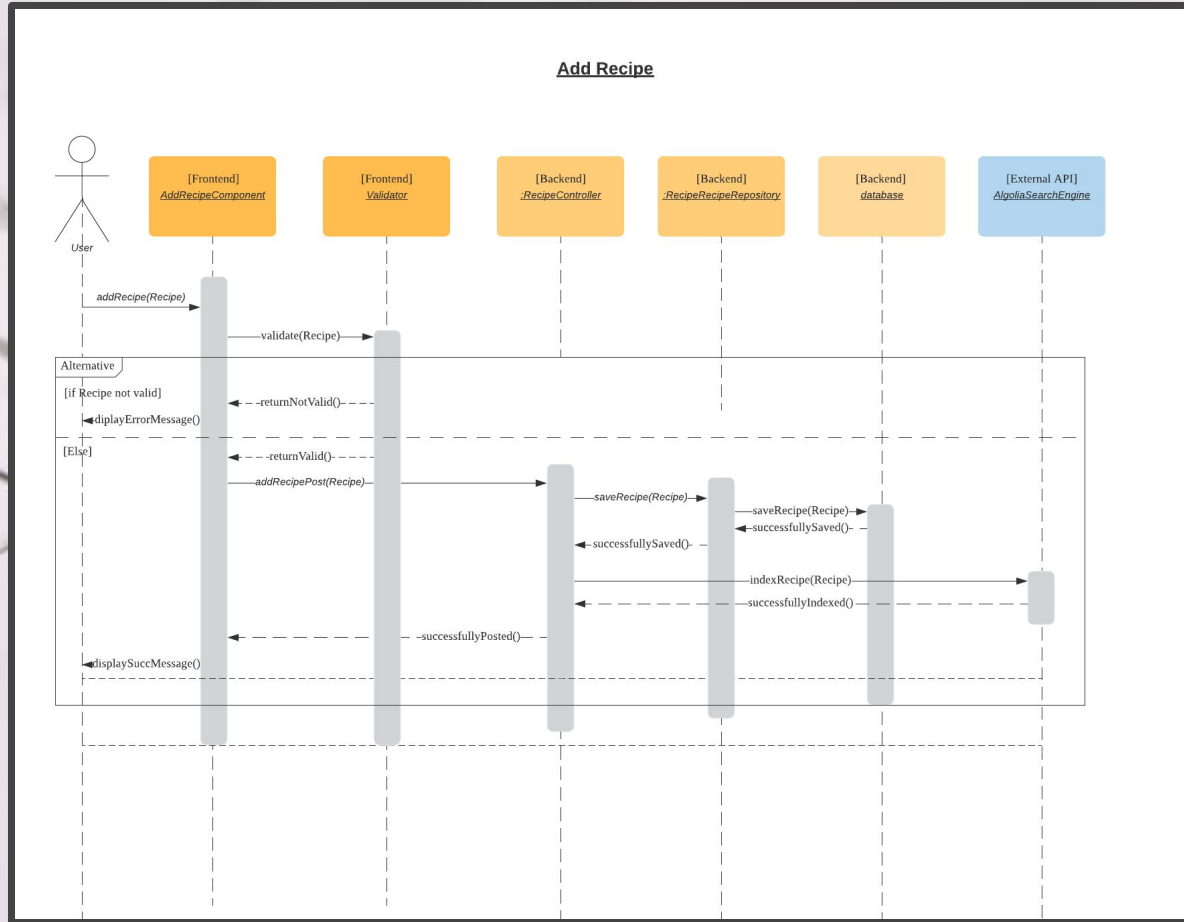
Live streaming

Messages and inbox

User Interface Diagram



Sequence Diagram



CLASS DIAGRAM

```

classDiagram
    class Recipe {
        -ID: String
        -likesCount: Int
        -mealType: String
        -dietAndHealth: String
        -worldCuisine: String
        -mealName: String
        -description: String
        -imageCollection: Image
        -videoCollection: Video
        -reviews: Review
        -comments: Comment
        -recipeStatus: RecipeStatus
        -favouriteBy: User
        +addImage(Image img): void
        +addVideo(Video video): void
        +addReview(Review review): void
        +addComment(Comment Comment): void
    }
    class User {
        -ID: String
        -name: String
        -profilePic: Image
        -aboutMe: String
        -notification: Notification
        -message: Message
        -recipe: Recipe
        -userStatus: AccountStatus
        -followers: ArrayList<User>
        -following: ArrayList<User>
        -blockedBy: ArrayList<User>
        +sendMessage(Message message): void
        +receiveMessage(): Message
        +deleteNotifications(): void
        +addRecipe(): boolean
        +editRecipe(): boolean
        +deleteRecipe(): boolean
        +favoriteRecipe(): void
        +commentOnRecipe(): void
        +giveReviewToRecipe(): void
        +blockUser(): boolean
    }
    class Message {
        -ID: String
        -sender: User
        -receiver: User
        -messageText: String
        -readBoolean: Boolean
        -image: Image
        +deleteMessage(): void
    }
    class Image {
        -ID: String
        -imgCap: String
    }
    class Video {
        -ID: String
        -likesCount: Int
        -videoName: String
        -viewsCount: Int
    }
    class Comment {
        -ID: String
        -likesCount: Int
        -recipe: Recipe
        -commentLikedBy: User
        +addComment(): void
        +deleteComment(): void
        +editComment(): void
    }
    class Review {
        -ID: String
        -recipe: Recipe
        -reviewGiven: Int
    }
    class Notification {
        -ID: String
        -notificationText: String
        -readBoolean: Boolean
    }
    class AdminUser {
        -ID: String
        -name: String
        -profile: Image
        -notification: Notification
    }
    class AccountStatus {
        <<Enum>>
        Deactivated
        Normal
    }
    class RecipeStatus {
        <<Enum>>
        Favorite
        Normal
    }
    class GeneralUser {
        <<Interface>>
        +getID(): String
        +getName(): String
        +getProfilePic(): Image
        +receiveNotification(): Notification
    }

    Recipe "1" -- "*" User
    Recipe "1" -- "*" Message
    Recipe "1" -- "*" Image
    Recipe "1" -- "*" Video
    Recipe "1" -- "*" Comment
    Recipe "1" -- "*" Review
    Recipe "1" -- "*" Notification
    User "1" -- "*" Message
    User "1" -- "*" Image
    User "1" -- "*" Video
    User "1" -- "*" Comment
    User "1" -- "*" Review
    User "1" -- "*" Notification
    Message "1" -- "*" Image
    Message "1" -- "*" Video
    Comment "1" -- "*" Image
    Comment "1" -- "*" Video
    Review "1" -- "*" Image
    Review "1" -- "*" Video
    Notification "1" -- "*" Image
    Notification "1" -- "*" Video
    AdminUser "1" -- "*" Image
    AdminUser "1" -- "*" Video
    AdminUser "1" -- "*" Comment
    AdminUser "1" -- "*" Review
    AdminUser "1" -- "*" Notification
    AccountStatus "1" -- "*" User
    RecipeStatus "1" -- "*" Recipe
    GeneralUser "1" -- "*" AdminUser
    
```

The diagram illustrates the relationships between various components of a recipe application. Key elements include:

- Recipe Class:** Contains attributes like ID, likesCount, mealType, dietAndHealth, worldCuisine, mealName, description, imageCollection, videoCollection, reviews, comments, recipeStatus, and favouriteBy. It has methods for adding images, videos, reviews, and comments.
- User Class:** Contains attributes like ID, name, profilePic, aboutMe, notification, message, recipe, userStatus, followers, following, and blockedBy. It has methods for sending and receiving messages, deleting notifications, adding/editing recipes, deleting recipes, favoriting recipes, commenting on recipes, giving reviews, and blocking users.
- Message Class:** Contains attributes like ID, sender, receiver, messageText, readBoolean, and image. It has a method for deleting a message.
- Image Class:** Contains attributes like ID and imgCap.
- Video Class:** Contains attributes like ID, likesCount, videoName, and viewsCount.
- Comment Class:** Contains attributes like ID, likesCount, recipe, and commentLikedBy. It has methods for adding, deleting, and editing comments.
- Review Class:** Contains attributes like ID, recipe, and reviewGiven.
- Notification Class:** Contains attributes like ID, notificationText, and readBoolean.
- AdminUser Class:** Contains attributes like ID, name, profile, and notification.
- AccountStatus Enum:** Has values Deactivated and Normal.
- RecipeStatus Enum:** Has values Favorite and Normal.
- GeneralUser Interface:** Defines methods like getID(), getName(), getProfilePic(), and receiveNotification().

Relationships are shown using solid lines with multiplicities (e.g., 1 to *), dashed lines with open diamonds for aggregation, and solid lines with hollow triangle heads for generalization.

Design Decisions

Choosing Google Auth over implementing our own login system

Using algolia search engine rather than writing our own

Choosing video streaming API over coding our own from scratch

Choosing mongoDb over MySql

Schedule and Deliverables

The screenshot displays a Trello board titled "Schedule" in a green-themed interface. The board is organized into four columns, each representing a project milestone. The top navigation bar includes icons for home, boards, and search, along with a notification banner that reads: "Hey cogitater.sigauke, we need you to confirm your email address. [Confirm email](#) · [Remind me later](#)".

Below the navigation bar, the board is categorized by "Personal" and "Private" tabs, with a filter set to "C MM A P RY". A "Butler" button and a "Show Menu" option are also visible.

The four project milestones and their associated tasks are as follows:

- Project Milestone1**
 - Create the Models for the DB (Due: Apr 24, Assignees: C, P)
 - Create React Components (Due: Apr 24, Progress: 0/3, Assignees: MM, RY)
 - Prepare the APIs and Libraries we expect to use (Due: Apr 25, Assignees: MM, P)
 - Implement the HomePage (Due: Apr 25, Progress: 0/3, Assignees: C, RY)
- Project Milestone2**
 - Implement Recipe - Backend (Due: May 6, Progress: 0/5, Assignees: C, RY)
 - Implement Recipe - Frontend (Assignees: C, P)
 - Implement Comment - Backend (Due: May 11, Progress: 0/4, Assignees: MM, P)
 - Implement Reviews - Backend (Due: May 8, Progress: 0/1, Assignees: C, P)
 - Implement Comment: Frontend (Assignees: MM, P)
- Project Milestone3**
 - Implement Chat Box - Backend (Assignees: C, MM)
 - Implement Notification - Frontend (Assignees: P, RY)
 - Implement Chat Box - Frontend (Due: May 18, Assignees: MM, RY)
 - Implement Live Stream - Backend (Due: May 20, Progress: 2, Assignees: C, RY)
 - Implement Live Stream - Frontend (Assignees: MM, P)
- Project Milestone4**
 - Testing (Due: May 29, Progress: 2, Assignees: C, MM, RY)
 - Ads (Due: May 28, Progress: 1, Assignees: C, MM, P)

Each milestone column includes a "+ Add another card" button at the bottom. A "+ Add another list" button is located on the right side of the board.

A close-up photograph of a white puzzle piece being placed into a larger white puzzle. The piece is slightly raised, showing its thickness and the interlocking edges. The background is a soft-focus view of the rest of the puzzle, with various other pieces visible. The lighting is bright and even, highlighting the clean, white surfaces of the puzzle.

Thank You !!!