

# RecAPI

"If you have something to steam, roast or peel, we will guide you to your meal!"

By Hanqing, Carla, Danielle, Sakusan



# Agenda



Our Approach




What Does Our API Do?



Postman Request Demo



Our Code



Improvements & Development



Reflection & Conclusion

# Our Approach

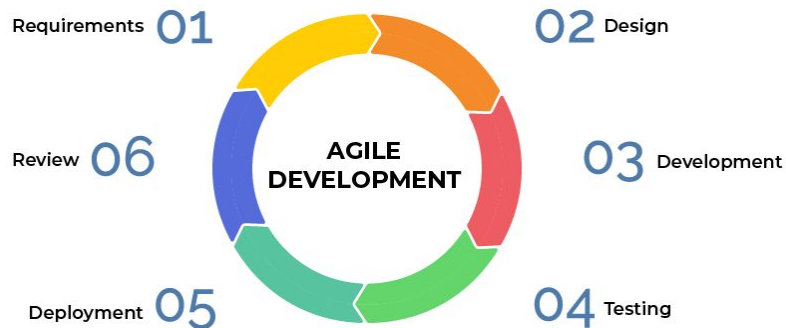
## ❖ What the team did:

- ~ Continuous collaboration
- ~ Member progress updates
- ~ Sort MVP + Extensions
- ~ Regular code testing

## ❖ How the team benefited:

- ~ Improved agility
- ~ Increased alignment
- ~ Reduced risk
- ~ Improved team spirit!

## *Agile Methodology*

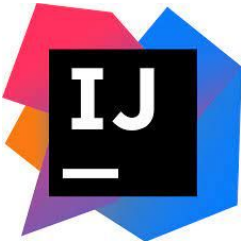


# Our Tools

## *Collaboration*



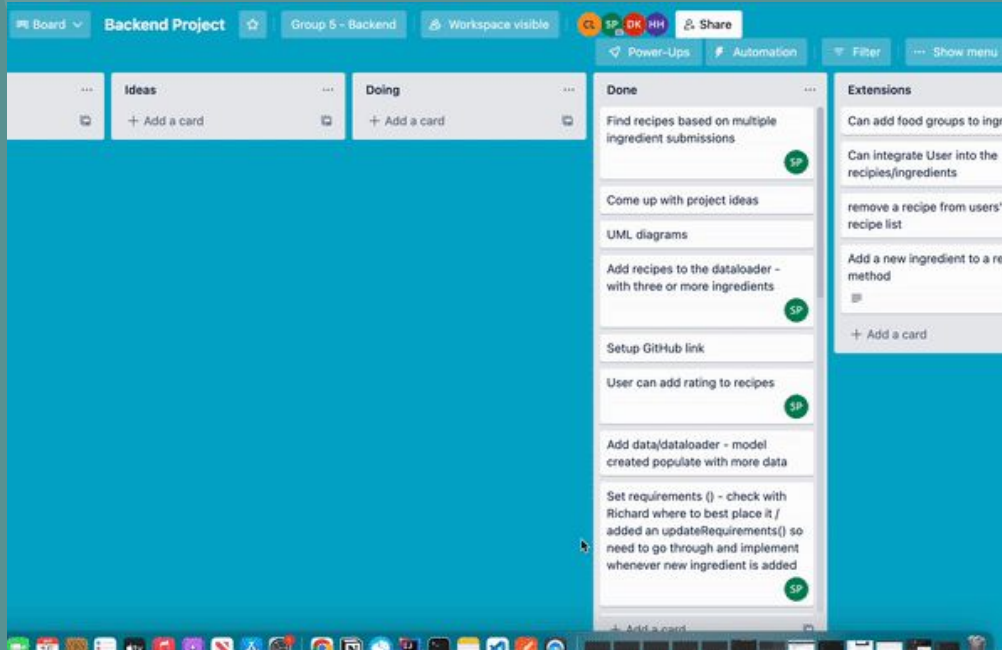
Google Slides



## *Project Management*



# Project Management

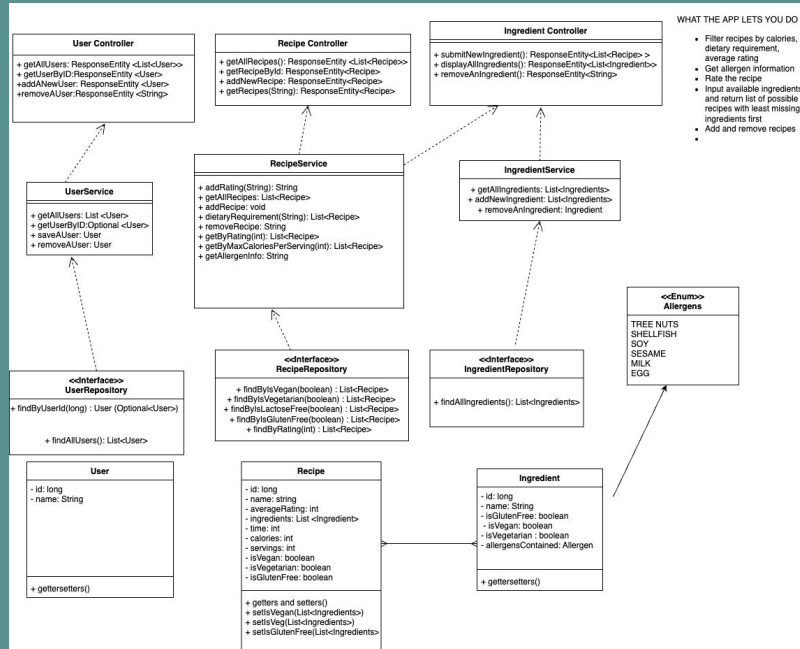


## ❖ How the team used Trello:

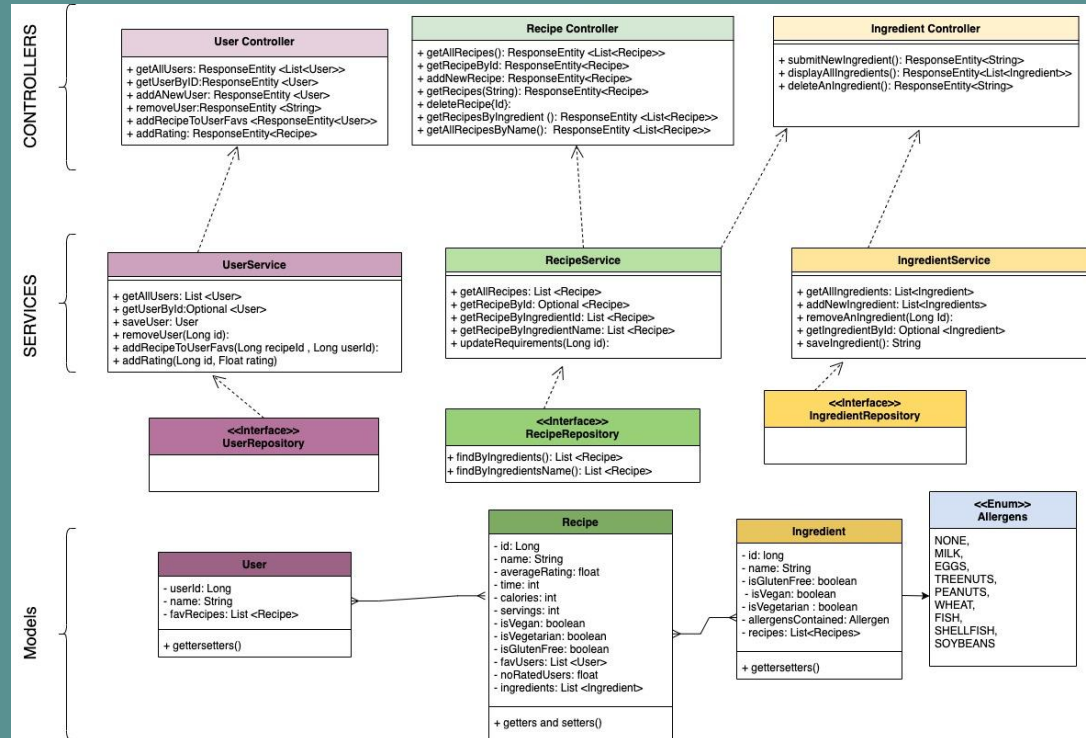
- ~ Create Workspace + Invite Members
- ~ Add 6 board..
- ~ Add tasks
- ~ Assign Tasks
- ~ Move tasks once complete
- ~ Repeat from step 3 when needed

# Unified Modeling Language (UML) Diagram

## Initial UML

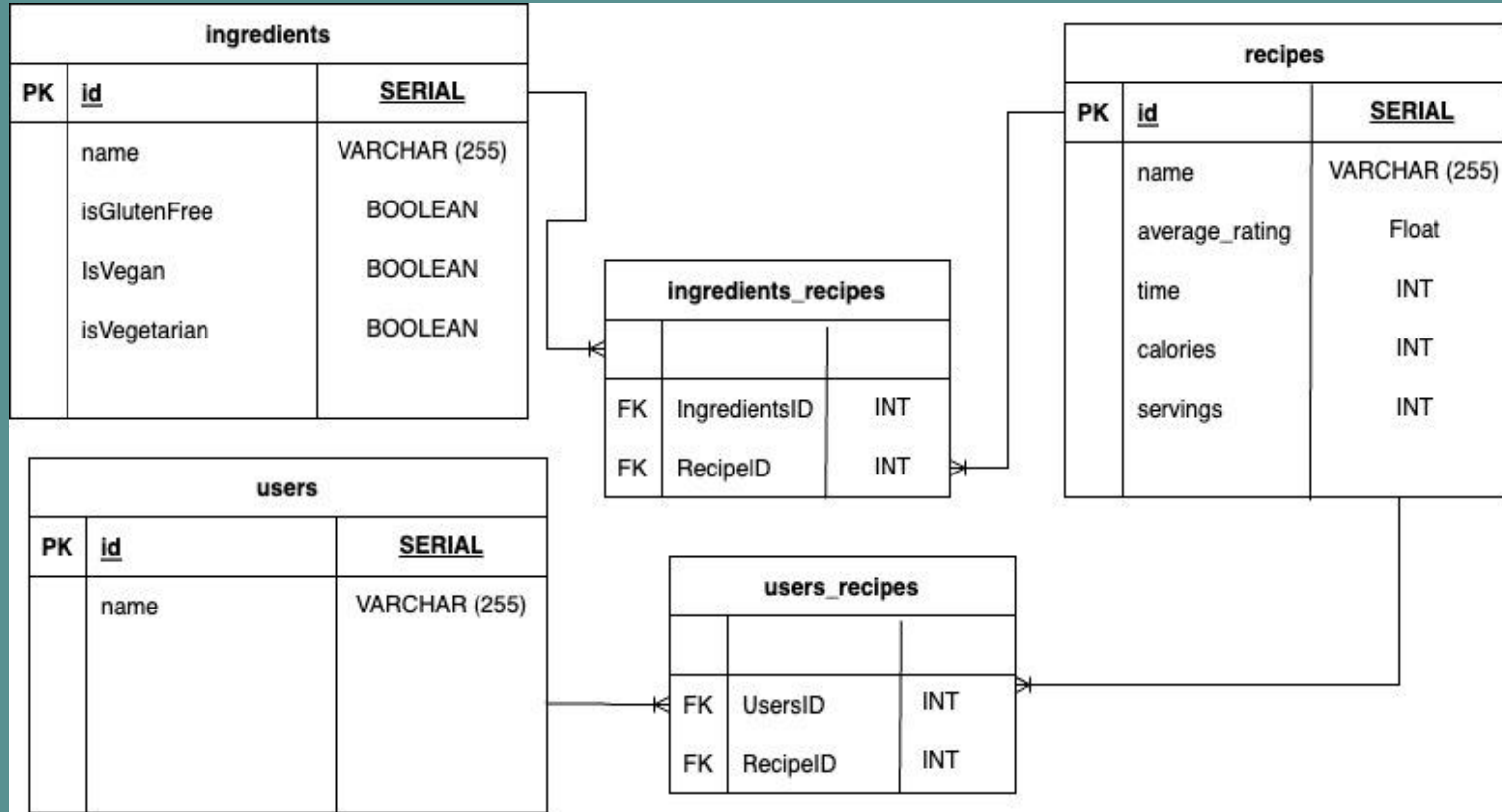


## Revised UML



# Entity Relationship (ERD) Diagram

## Revised ERD



# What does our API do?

Our API is designed to help users find recipes based on their available ingredients.

User Controller	Recipe Controller	Ingredient Controller
<ul style="list-style-type: none"><li>❖ Create and delete profile</li><li>❖ Add recipes to their favourites</li><li>❖ Rate recipes</li></ul>	<ul style="list-style-type: none"><li>❖ Add and delete recipes</li><li>❖ Get recipes containing up to 3 inputted user ingredients</li><li>❖ Get recipes by dietary requirement</li></ul>	<ul style="list-style-type: none"><li>❖ Submit new ingredients</li><li>❖ Delete ingredients</li><li>❖ Display all ingredients</li></ul>



# Demo

The screenshot displays the Postman web application interface. At the top, there's a navigation bar with links for Home, Workspaces, API Network, and Explore, along with a search bar and user options. A notification states, "Your team just unlocked a 14 day trial!". Below this, the main workspace is titled "Group 5 Recipe API".

The left sidebar shows a "Collections" panel with a tree view containing "recipe-api" and several GET endpoints. The main area shows a "POST new user" request configuration. The URL is "http://localhost:8080/users". The "Body" tab is selected, showing a JSON payload: 

```
{  "name": "Colin"}
```

. The "Headers" tab shows "Content-Type" set to "application/json".

Below the request configuration, the "Response" section is visible, showing the JSON response: 

```
{  "userId": 10,  "name": "Colin",  "favRecipes": null}
```

. The response status is "201 Created" with a response time of "800 ms".

# Ingredients : Ingredient Service

## Adding ingredients to the API

- ❖ This method does not allow ingredients to be added in duplicate!

```
public String saveIngredient(Ingredient ingredient) {  
    List ingredientNames = ingredientRepository.findByName(ingredient.getName());  
    if (ingredientRepository.findByName(ingredient.getName()).isEmpty()) {  
        ingredientRepository.save(ingredient);  
        return "You have added your ingredient!";  
    } else if (ingredientNames != null){  
        return "You have already added this ingredient!";  
    } else {  
        ingredientRepository.save(ingredient);  
        return "saved ingredient";  
    }  
}
```

# Recipe Controller - Request

## ◆ GetMapping

Each recipe is given their own identifier + users can get recipes by typing in the Id.

```
(value =("/{id}")  
public ResponseEntity<Recipe> getRecipeById(@PathVariable Long id) {  
    Optional<Recipe> recipe = recipeService.getRecipeById(id);  
    if (recipe.isPresent()) {  
        return new ResponseEntity<>(recipe.get(), HttpStatus.OK);  
    } else {  
        return new ResponseEntity<>(null, HttpStatus.NOT_FOUND);  
    }  
}
```

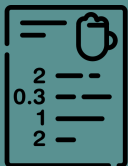
If recipe id is not present



# Recipe Controller



@GET  
all  
recipes



@POST  
new  
recipe



@GET  
recipe by  
ingredients



@GET  
recipe by  
dietary  
req

```
List<  
    @RequestParam Optional<String> ingredient1,  
    @RequestParam Optional<String> ingredient2, @RequestParam Optional<String> ingredient3  
> findByName(ingredient1, ingredient2, ingredient3);
```

```
List<Recipe> findByIsVegan(Boolean isVegan);
```

```
List<Recipe> recipes = recipeService.getRecipeByIngredientName(ingredient1.get());
```

```
List<Recipe> second_recipes = recipeService.getRecipeByIngredientName(ingredient2.get());
```

```
List<Recipe> third_recipes = recipeService.getRecipeByIngredientName(ingredient3.get());
```

```
second_recipes.retainAll(third_recipes);
```

```
recipes.retainAll(second_recipes);
```

# Challenges & Improvements

## Authentication

Add authentication so only user with the correct username and password can access account

## Diet Plan

Let user specify if they aim to gain or lose weight and return tailored recipes

## User favourites

Utilise users list of favourite recipes to recommend similar recipe



## TDD

Use TDD whilst writing logic to eliminate need for exhaustive Postman testing

## Custom Queries

Implement custom queries instead of combining same derived query

## Food groups

Add food groups to ingredients to categorise and choose relevant recipes

**Thank you for  
listening!**

**Do you have  
any questions?**

