

Third party APIs and Technologies

Spring Boot

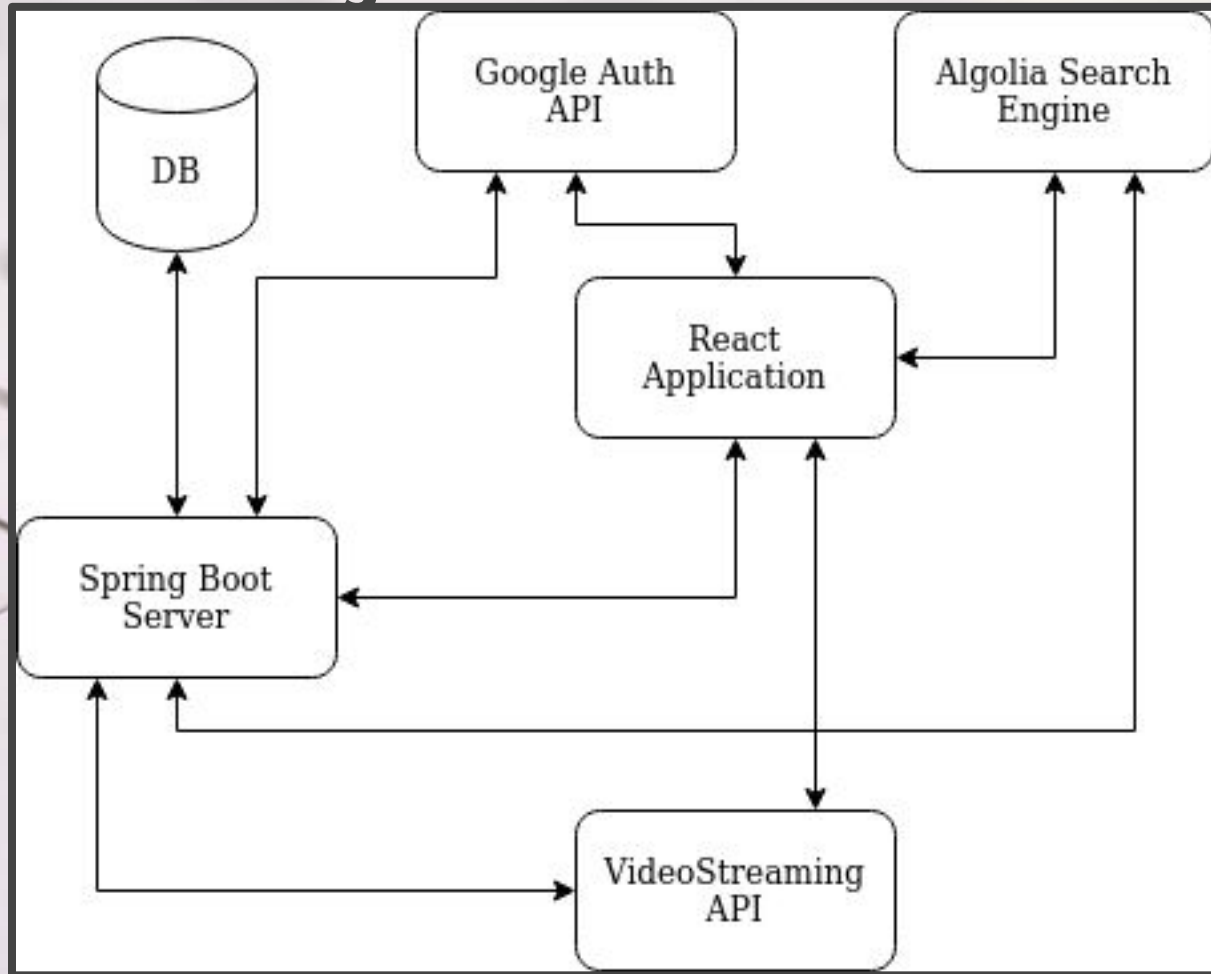
React.js

Algolia Search Engine

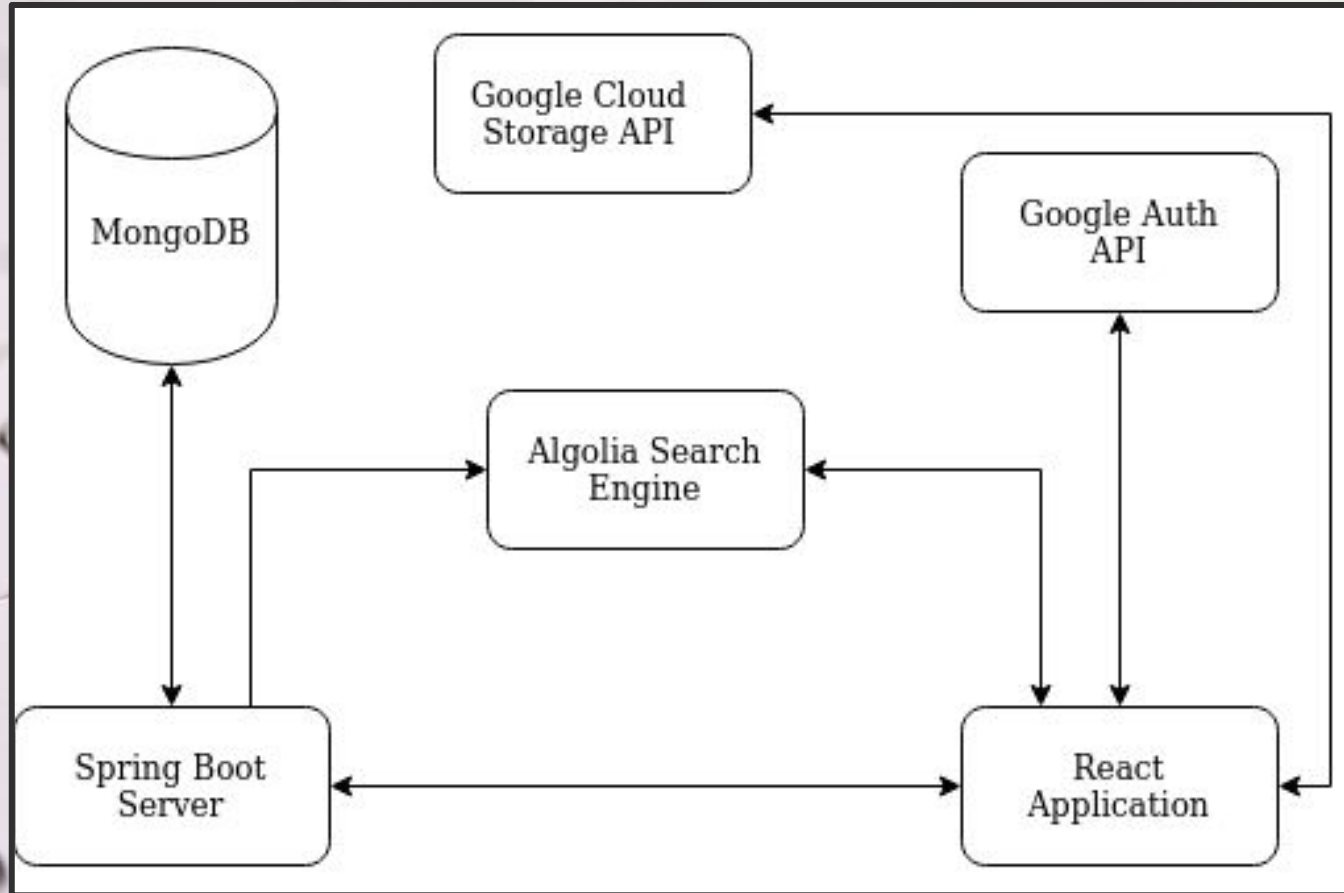
Google Storage API

Google Auth

Original Architecture



Final Architecture - (MVC)



Technically Complex Problems

Real time search

Saving and accessing images with URL links

Solutions

Algolia Search

Google Cloud Storage

Indexing Recipes To Algolia

```
70
71 @RequestMapping(method=RequestMethod.POST, value="app/user/add/recipe")
72 public Recipe saveRecipe(@RequestBody Recipe recipe) {
73
74     recipeRepository.save(recipe);
75
76     // UPDATE INDEX
77     RecipeImage recipeImage = new RecipeImage(
78         recipe.getId(),
79         recipe.getUserName(),
80         recipe.getMealType(),
81         recipe.getDietAndHealth(),
82         recipe.getWorldCuisine(),
83         recipe.getMealName(),
84         recipe.getCreatedAt(),
85         recipe.getImageString(),
86         recipe.getLikesCount()
87     );
88     index.saveObject(recipeImage);
89     return recipe;
90 }
91 //update recipe info
```

Rendering Hits from Algolia

```
376
377 <div className="row">
378
379   {(this.state.hitsFound) &&
380
381     (
382       this.state.Hits.map((hit, i)=>(
383         <div className="col-lg-4 col-md-6 mb-4" key={i}>
384           <div className="card border-0 shadow">
385             <Link
386               to={{
387                 pathname: "/Recipe",
388                 state:{
389                   userId: this.props.location.state.userId,
390                   userName: this.props.location.state.userName,
391                   profileImage: this.props.location.state.imageString,
392                   imageString: this.props.location.state.imageString,
393                   recipeId: hit.objectID
394                 }}
395             >>
396
397             <img src={hit.imageString} className="card-img-top" alt="..."/>
398           </Link>
399
```

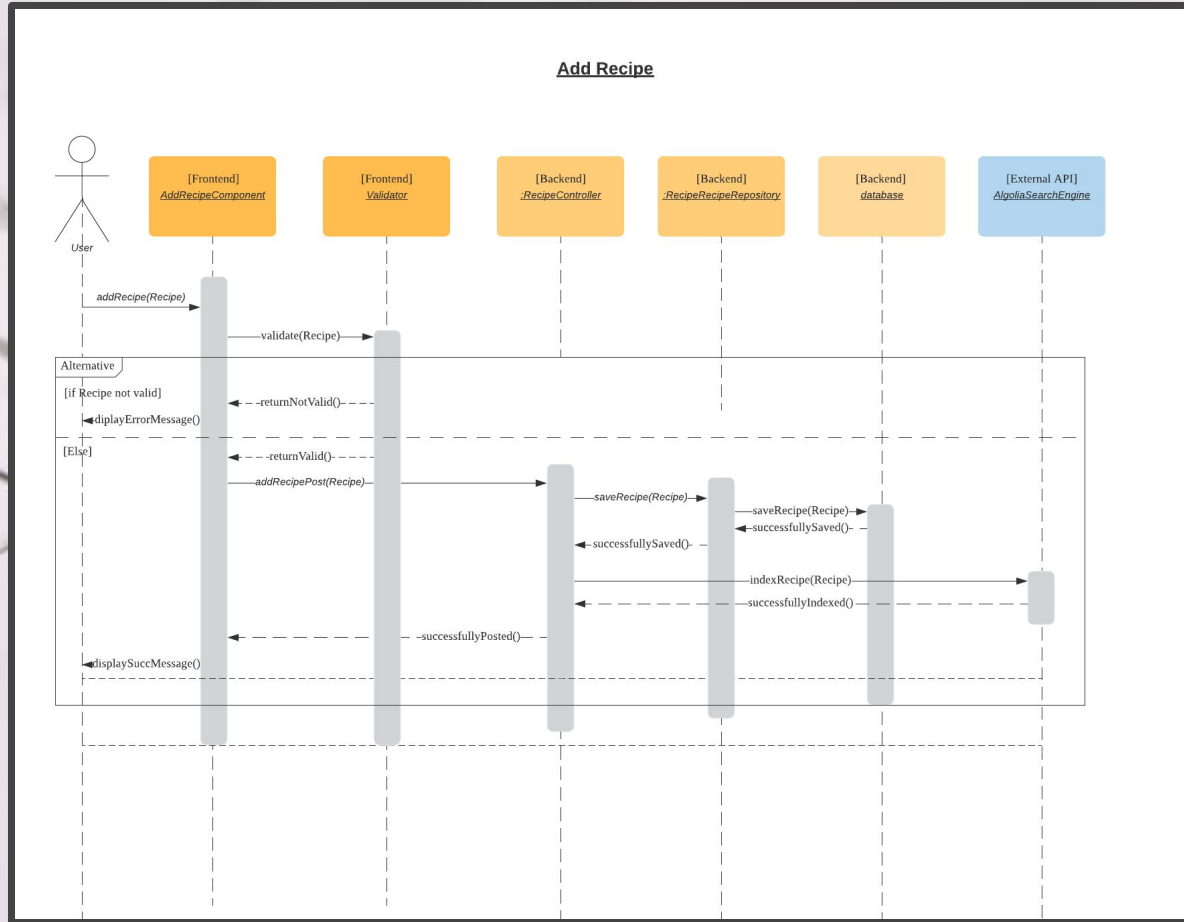

Saving Image To Firebase and Fetching URL

```
376
377 <div className="row">
378
379   {(this.state.hitsFound) &&
380
381     (
382       this.state.Hits.map((hit, i)=>(
383         <div className="col-lg-4 col-md-6 mb-4" key={i}>
384           <div className="card border-0 shadow">
385             <Link
386               to={{
387                 pathname: "/Recipe",
388                 state:{
389                   userId: this.props.location.state.userId,
390                   userName: this.props.location.state.userName,
391                   profileImage: this.props.location.state.imageString,
392                   imageString: this.props.location.state.imageString,
393                   recipeId: hit.objectID
394                 }}
395             >>
396
397             <img src={hit.imageString} className="card-img-top" alt="..."/>
398           </Link>
399
```

Basics of how the system works

Fetch data from React frontend
Make calls to Spring Boot API functions
Use response perform action

Example: Sequence Diagram



Frontend: Add Recipe

```
JS Recipe.js JS Home.js JS AddRecipe.js
foodweb_frontend > src > JS AddRecipe.js > AddRecipe > onSubmit

135 onSubmit = (e) => {
136   e.preventDefault();
137   const { userId, userName, mealType, dietAndHealth, worldCuisine, mealName, description,
138         imageString, videoId, steps, ingredients } = this.state;
139   axios.post('/user/add/recipe/', { userId, userName, mealType, dietAndHealth,
140                                   worldCuisine, mealName, description, imageString, videoId, steps, ingredients })
141     .then((result) => {
142       console.log("After Posting new Contact - returned data: " + result.data);
143       const recipeID = result.data.id;
144       this.props.history.push({
145         pathname: "/Home",
146         state: {userId: this.state.userId,
147                userName: this.state.userName,
148                imageString: this.state.profileImage
149                }
150       });
151     })
152     .catch((err) => {
153       console.log(`=====response.data=====`);
154       console.log(`Errors: {errors}`);
155     })
156     .catch((err) => {
157       console.log(`Errors: {errors}`);
158     });
159   });
160 }
161
162 handleFileChange = e => {
163   if (e.target.files[0]) {
```

Backend: Add Recipe

```
JS Recipe.js JS Home.js JS AddRecipe.js RecipeController.java X
foodweb_backend > src > main > java > com > app > foodweb > controllers > RecipeController.java
70
71 @RequestMapping(method=RequestMethod.POST, value="app/user/add/recipe")
72 public Recipe saveRecipe(@RequestBody Recipe recipe) {
73
74     recipeRepository.save(recipe);
75
76     // UPDATE INDEX
77     RecipeImage recipeImage = new RecipeImage(
78         recipe.getId(),
79         recipe.getUserName(),
80         recipe.getMealType(),
81         recipe.getDietAndHealth(),
82         recipe.getWorldCuisine(),
83         recipe.getMealName(),
84         recipe.getCreatedAt(),
85         recipe.getImageString(),
86         recipe.getLikesCount()
87     );
88     index.saveObject(recipeImage);
89     return recipe;
90 }
91 //update recipe info
92 @RequestMapping(method=RequestMethod.PUT, value="app/edit_recipe/{id}")
93 public Recipe updateRecipe(@PathVariable String id, @RequestBody Recipe recipe){
94     Recipe r = recipeRepository.findById(id).get();
95
96     if(recipe.getMealName() != null){
97         r.setMealName(recipe.getMealName());
98     }
99     if(recipe.getMealType() != null){
```

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

A close-up photograph of a white puzzle piece being placed into a larger white puzzle. The puzzle piece is slightly raised, showing its interlocking edges. The background is a soft-focus view of the rest of the puzzle. Overlaid on the center of the puzzle piece is the text "Thank You !!!" in a bold, orange, sans-serif font.

Thank You !!!