RecipeDB Frontend

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app.scss

```scss
// widths
$breakpoint-xs: 400px;
$breakpoint-sm: 576px;
$breakpoint-md: 768px;
$breakpoint-lg: 992px;
$breakpoint-xl: 1480px;
$breakpoint-xxl: 1920px;

// colors
$brand: #5248d7;
$white: #ffffff;
$black: #000000;
$border: #cccccc;
$dark: lighten($black, 10%);
$error: #cc3333;

// transition timings
$t-fast: 250ms;
$t-reg: 500ms;

body {
    overflow: hidden;
}

// wrap
.wrap {
    width: 100%;
    padding-left: 2rem;
    padding-right: 2rem;
    transition: padding $t-reg;

    @media screen {
        @media (min-width: $breakpoint-sm) {
            padding-left: 4rem;
            padding-right: 4rem;
        }
        @media (min-width: $breakpoint-md) {
            padding-left: 4rem;
            padding-right: 4rem;
        }
        @media (min-width: $breakpoint-lg) {
            padding-left: 8rem;
            padding-right: 8rem;
        }
        @media (min-width: $breakpoint-xl) {
            padding-left: 16rem;
            padding-right: 16rem;
        }
        @media (min-width: $breakpoint-xxl) {
            padding-left: 24rem;
            padding-right: 24rem;
        }
    }
}

.vertical-wrap {
    width: 100%;
    padding-top: 2rem;
    padding-bottom: 2rem;
    transition: padding $t-reg;

    @media screen and (min-width: $breakpoint-md) {
        padding-top: 4rem;
        padding-bottom: 4rem;
    }
}

a {
    text-decoration: none;
    color: $brand;
    transition: color $t-fast;

    &:hover {
        color: lighten($brand, 10%);
    }
}

.btn {
    transition: background-color $t-fast;

    svg {
        fill: $white;
        width: 1rem;
        height: 1rem;
    }

    &.btn-rdb {
        background-color: $brand;
        color: $white;

        &:hover {
            background-color: darken($brand, 10%);
        }
    }
}

.page-header {
    background-color: darken($white, 10%);
    user-select: none;

    &.dark {
        background-color: darken($white, 15%);
    }

    h1, h4, p {
        margin-bottom: unset;
    }

    h4 {
        font-weight: normal;
    }
}

.recipe-container {
    display: flex;
    flex-wrap: wrap;
    margin: -1rem;

    .recipe {
        width: 100%;
        padding: 1rem;

        .card {
            height: 100%;

            h5 {
                font-weight: bold;
            }
        }

        @media screen {
            @media (min-width: $breakpoint-md) {
                width: 50%;
            }
            @media (min-width: $breakpoint-xl) {
                width: calc(100% / 3);
            }
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\config.ts

```
export const API_URL = 'http://localhost:5000';
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\index.html

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>RecipeDB</title>
    <base href="/">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/x-icon" href="./assets/favicon/favicon.ico">
    <link rel="apple-touch-icon" sizes="180x180" href="./assets/favicon/apple-
touch-icon.png">
    <link rel="icon" type="image/png" sizes="32x32"
href="./assets/favicon/favicon-32x32.png">
    <link rel="icon" type="image/png" sizes="16x16"
href="./assets/favicon/favicon-16x16.png">
    <link rel="manifest" href="./assets/favicon/site.webmanifest">
    <link rel="mask-icon" href="./assets/favicon/safari-pinned-tab.svg"
color="#5248d7">
```

```
    <meta name="apple-mobile-web-app-title" content="RecipeDB">
    <meta name="application-name" content="RecipeDB">
    <meta name="msapplication-TileColor" content="#5248d7">
    <meta name="theme-color" content="#5248d7">
</head>
<body>
<app-root></app-root>
</body>
</html>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\main.ts

```
import {enableProdMode} from '@angular/core';
import {platformBrowserDynamic} from '@angular/platform-browser-dynamic';

import {AppModule} from './app/app.module';
import {environment} from './environments/environment';

if (environment.production) {
    enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
    .catch(err => console.error(err));
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\app.component.html

```
<div class="app">
    <app-nav></app-nav>
    <div class="app-body">
        <router-outlet></router-outlet>
    </div>
    <app-notifications></app-notifications>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\app.component.scss

```
@import '../app';

.app {
    display: flex;
    flex-direction: column;
    width: 100vw;
    height: 100vh;
    overflow: hidden;

    .app-body {
        flex: 1;
        height: 100%;
        overflow-y: auto;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\app.component.ts

```
import {Component} from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'RecipeDB';
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\app.module.ts

```
import {BrowserModule} from '@angular/platform-browser';
import {NgModule} from '@angular/core';
import {HttpClientModule} from '@angular/common/http';
import {RouterModule} from '@angular/router';
import {ReactiveFormsModule} from '@angular/forms';

import {NgbModule} from '@ng-bootstrap/ng-bootstrap';
import {ClickOutsideModule} from 'ng-click-outside';
import {NgxPaginationModule} from 'ngx-pagination';

import {AuthService} from './auth/auth.service';
import {RecipeService} from './recipes.service';

import {AppComponent} from './app.component';
import {NavComponent} from './nav/nav.component';

import {HomeComponent} from './home/home.component';
import {LoginComponent} from './auth/login/login.component';
import {RegisterComponent} from './auth/register/register.component';
import {RecipeComponent} from './recipe/recipe.component';
import {SearchComponent} from './search/search.component';
import {BookmarksComponent} from './bookmarks/bookmarks.component';
import {AdminComponent} from './admin/admin.component';
import {BarComponent} from './search/bar/bar.component';
import {NotificationsComponent} from './notifications/notifications.component';
import {NotificationService} from './notification.service';

const routes = [
    {path: '', component: HomeComponent},
    {path: 'login', component: LoginComponent},
    {path: 'register', component: RegisterComponent},
    {path: 'recipes/search/:criteria', component: SearchComponent},
    {path: 'recipes/search/:criteria/:page', component: SearchComponent},
    {path: 'recipe/:id', component: RecipeComponent},
    {path: 'bookmarks', component: BookmarksComponent},
    {path: 'admin', component: AdminComponent}
];

@NgModule({
    declarations: [
        AppComponent,
        NavComponent,
        HomeComponent,
        RecipeComponent,
        LoginComponent,
        RegisterComponent,
        SearchComponent,
        BookmarksComponent,
        AdminComponent,
        BarComponent,
        NotificationsComponent
    ],
    imports: [
        BrowserModule,
        HttpClientModule,
        RouterModule.forRoot(routes),
        ReactiveFormsModule,
        NgbModule,
        ClickOutsideModule,
        NgxPaginationModule
    ],
    providers: [AuthService, NotificationService, RecipeService],
    bootstrap: [AppComponent]
})
export class AppModule {
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\notification.service.ts

```
import {Injectable} from '@angular/core';
import {HttpClient} from '@angular/common/http';
import {BehaviorSubject} from 'rxjs';
import uuid from 'uuid';


@Injectable()
export class NotificationService {
    constructor(private http: HttpClient) {
        this._notifications = new BehaviorSubject({});
        this.notifications = this._notifications.asObservable();
    }

    private _notifications;
    public notifications;

    notify(title: string, message: string) {
        const data = this._notifications.getValue();
        data[uuid.v4()] = {
            title,
            message,
            date: Date.now(),
        };
        this._notifications.next(data);
    }

    removeNotification(id) {
        const data = this._notifications.getValue();
        delete data[id];
        this._notifications.next(data);
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\recipes.service.ts

```
import {HttpClient, HttpHeaders} from '@angular/common/http';
import {Injectable} from '@angular/core';
import {BehaviorSubject, Subject} from 'rxjs';
import {API_URL} from '../config';
import {isEqual} from 'lodash';

@Injectable()
export class RecipeService {
    constructor(private http: HttpClient) {
        this._top = new BehaviorSubject([]);
        this.top = this._top.asObservable();

        this._data = new BehaviorSubject({});
        this.data = this._data.asObservable();

        this._comments = new BehaviorSubject([]);
        this.comments = this._comments.asObservable();

        this._search = new Subject();
        this.search = this._search.asObservable();

        this._bookmarks = new BehaviorSubject([]);
        this.bookmarks = this._bookmarks.asObservable();
    }

    private _top;
    public top;

    private _data;
    public data;

    private _comments;
    public comments;

    private _search;
    public search;

    private _bookmarks;
    public bookmarks;

    getRecipe(id: string, token?: string) {
        let headers = null;
        if (token) {
            headers = {
                headers: new HttpHeaders({'x-access-token': token})
            };
            this.http.get(`${API_URL}/recipe/${id}`,
headers).toPromise().then(response => {
                if (!isEqual(response, this._data.getValue())) {
                    this._data.next(response);
                }
            }).catch(error => {
                console.warn(error);
                this._data.next(null);
            });
        } else {
            this.http.get(`${API_URL}/recipe/${id}`).toPromise().then(response =>
{
                this._data.next(response);
            }).catch(error => {
                console.warn(error);
                this._data.next(null);
            });
        }
    }

    getRecipeComments(id: string) {

this.http.get(`${API_URL}/recipe/${id}/comments`).toPromise().then(response => {
            this._comments.next(response);
        }).catch(error => {
            console.warn(error);
        });
    }

    comment(id: string, body: string, token: string) {
        let formData = new FormData();
        formData.append('body', body);

        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };

        this.http.post(`${API_URL}/recipe/${id}/comments`, formData,
headers).toPromise().then(response => {
            this._comments.next(response);
        });
    }

    updateComment(id: string, body: string, token: string) {
        let formData = new FormData();
        formData.append('body', body);

        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };

        this.http.put(`${API_URL}/recipe/${id}/comments`, formData,
headers).toPromise().then(response => {
            this._comments.next(response);
        });
    }

    deleteComment(id: string, token: string) {
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        return this.http.delete(`${API_URL}/recipe/${id}/comments`,
headers).toPromise().then((response) => {
            this._comments.next(response);
            return {message: 'Your comment has been deleted successfully.'};
        }).catch(error => {
            return {error: 'An error occurred deleting this comment.'};
        });
    }

    bookmarkRecipe(id: string, token: string) {
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        return this.http.post(`${API_URL}/recipe/${id}/bookmark`, null,
headers).toPromise().then(() => {
            const recipe = this._data.getValue();
            if (!recipe.bookmarked) {
                recipe.bookmarked = true;
                this._data.next(recipe);
            }
            return {message: `"${recipe.title}" has been added to your
bookmarks.`};
        }).catch(error => {
            return {error: 'An error occurred bookmarking this recipe.'};
        });
    }

    unbookmarkRecipe(id: string, token: string) {
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        return this.http.delete(`${API_URL}/recipe/${id}/bookmark`,
headers).toPromise().then(() => {
            const recipe = this._data.getValue();
            if (recipe.bookmarked) {
                recipe.bookmarked = false;
                this._data.next(recipe);
            }
            return {message: `"${recipe.title}" has been removed from your
bookmarks.`};
        }).catch(error => {
            return {error: 'An error occurred removing this recipe from your
bookmarks.'};
        });
    }

    searchRecipes(criteria: string, page?: number) {
        if (criteria.length > 0) {
            this.http.get(`${API_URL}/recipes/search?criteria=${criteria}${page >
1 ? `&p=${page}` : ''}`).toPromise().then(response => {
                this._search.next(response);
            }).catch(error => {
            });
        }
    }

    topRecipes() {
        if (this._top.getValue().length === 0) {
            this.http.get(`${API_URL}/recipes/top`).toPromise().then(response => {
                if (!isEqual(response, this._top.getValue())) {
                    this._top.next(response);
                }
            });
        }
    }

    getBookmarks(token: string) {
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        this.http.get(`${API_URL}/bookmarks`, headers).toPromise().then(response
=> {
            if (!isEqual(response, this._bookmarks.getValue())) {
                this._bookmarks.next(response);
            }
        }).catch(error => {
        });
    }

    scrapeBbc(url: string, token: string, navigate: Function) {
        let formData = new FormData();
        formData.append('url', url);
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        this.http.post(`${API_URL}/recipes`, formData,
headers).toPromise().then(response => {
            if (response.hasOwnProperty('inserted')) {
                navigate(response['inserted']);
            }
        });
    }

    deleteRecipe(id: string, token: string) {
        const headers = {
            headers: new HttpHeaders({'x-access-token': token})
        };
        return this.http.delete(`${API_URL}/recipe/${id}`,
headers).toPromise().then(() => {
            this.getRecipe(id, token);
            return {message: `Recipe has been deleted successfully.`};
        }).catch(error => {
            return {error: 'An error occurred deleting this recipe.'};
        });
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\admin\admin.component.html

```
<div class="page-header wrap vertical-wrap">
    <h1>Admin</h1>
</div>

<div class="page-header dark wrap admin-menu">
    <div>Scrape New Recipe</div>
</div>

<div class="wrap vertical-wrap content">
    <div class="scrape" *ngIf="activePage === 'scrape'">
```

```
        <form [formGroup]="scrapeForm">
            <div class="form-group">
                <label for="url">BBC Recipe URL</label>
                <input
                    id="url"
                    type="text"
                    name="url"
                    class="form-control"
                    formControlName="url"
                    placeholder="BBC Recipe URL"
                />
            </div>
            <button class="btn btn-rdb" (click)="scrape()">Scrape</button>
        </form>
    </div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\admin\admin.component.scss

```
@import '../../app';

.admin-menu {
    display: flex;

}

.content {
    .scrape {
        form {
            display: flex;
            flex-direction: column;
            max-width: 30rem;

            button {
                margin-left: auto;
            }
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\admin\admin.component.ts

```
import {Component, OnInit} from '@angular/core';
import {AuthService} from '../auth/auth.service';
import {Router} from '@angular/router';
import {FormBuilder} from '@angular/forms';
import {RecipeService} from '../recipes.service';

@Component({
    selector: 'app-admin',
    templateUrl: './admin.component.html',
    styleUrls: ['./admin.component.scss']
})
export class AdminComponent implements OnInit {
    constructor(private authService: AuthService, private recipeService:
RecipeService, private router: Router, private formBuilder: FormBuilder) {
        this.activePage = 'scrape';
        this.scrapeForm = this.formBuilder.group({
            url: ''
        });
    }

    activePage;
    scrapeForm;
    token: string = null;

    ngOnInit() {
        this.authService.user.subscribe(user => {
            if (user.admin) {
                this.token = user.token;
            } else {
                this.router.navigate(['login']);
            }
        });
    }

    changePage(page: string) {
        this.activePage = page;
    }

    scrape() {
        const {url} = this.scrapeForm.value;
        const navigate = (id) => {
            this.router.navigate(['recipe', id]);
        };
        this.recipeService.scrapeBbc(url, this.token, navigate);
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\auth.scss

```
@import '../../app';

.auth-form {
    max-width: 30rem;
    margin: 0 auto;

    h3 {
        margin-bottom: 2rem;
    }

    .form-group{
        &:last-of-type{
            margin-bottom: 2rem;
        }
    }

    .btn-success {
        color: $white;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\auth.service.ts

```
import {Injectable} from '@angular/core';
import {HttpClient, HttpHeaders} from '@angular/common/http';
import {BehaviorSubject} from 'rxjs';
import {API_URL} from '../../config';

@Injectable()
export class AuthService {
    constructor(private http: HttpClient) {
        this._user = new BehaviorSubject(getDefaultUserObject());
        this.user = this._user.asObservable();
        this.getStoredUser();
    }

    private _user;
    public user;

    register(name: string, username: string, email: string, password: string) {
        let data = new FormData();
        data.append('name', name);
        data.append('email', email);
        data.append('username', username);
        data.append('password', password);

        return this.http.post(`${API_URL}/register`, data).toPromise().then(() =>
{
            this.login(username, password);
        }).catch(error => {
            if (error.error) {
                return error.error;
            } else {
                return null;
            }
        });
    }

    login(username: string, password: string) {
        let data = new FormData();
        data.append('username', username);
        data.append('password', password);

        return this.http.post(`${API_URL}/login`, data).toPromise().then(response
=> {
            localStorage.setItem('user', JSON.stringify(response));
            this._user.next(response);
            return null;
        }).catch(error => {
            if (error.error) {
                return error.error;
            } else {
                return null;
            }
        });
    }
    logout() {
        const token = this._user.getValue().token;
        if (token && token.length > 0) {
            const headers = {
                headers: new HttpHeaders({'x-access-token': token})
            };
            return this.http.post(`${API_URL}/logout`, null,
headers).toPromise().then(response => {
                this._user.next(getDefaultUserObject());
                localStorage.removeItem('user');
                // returm
            }).catch(error => {
                return {error: 'An error occurred logging out, please try
again.'};
            });
        } else {
            return new Promise(resolve => {
                resolve(null);
            });
        }
    }

    getStoredUser() {
        try {
            const user = localStorage.getItem('user');
            if (user && user.length > 0) {
                const userObj = JSON.parse(user);
                if (new Date(userObj.exp) > new Date()) {
                    this._user.next(userObj);
                } else {
                    this._user.next(getDefaultUserObject());
                }
            }
        } catch {
            this._user.next(getDefaultUserObject());
        }
    }
}

export const getDefaultUserObject = () => {
    return {
        name: null,
        username: null,
        token: null,
        exp: null
    };
};
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\login\login.component.html

```
<div class="page-header wrap vertical-wrap">
    <h1>Login</h1>
    <h4>Don't have an account? <a [routerLink]="'/register'">Register
here</a></h4>
</div>
<div class="wrap vertical-wrap">
    <form class="auth-form" [formGroup]="loginForm" (ngSubmit)="login()">
```

```
        <div class="form-group">
            <label for="username">Username</label>
            <input type="text" id="username" name="username" class="form-control" 
formControlName="username"/>
            <small *ngIf="isInvalid('username')" class="form-text text-muted">
                The username field is required.
            </small>
        </div>
        <div class="form-group">
            <label for="password">Password</label>
            <input type="password" id="password" name="password" class="form-
control" formControlName="password"/>
            <small *ngIf="isInvalid('password')" class="form-text text-muted">
                The password field is required.
            </small>
        </div>
        <button type="submit" class="btn btn-block btn-success">Login</button>
    </form>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\login\login.component.ts

```
import {Component, OnInit} from '@angular/core';
import {Router} from '@angular/router';
import {FormBuilder, Validators} from '@angular/forms';

import {AuthService} from '../auth.service';
import {NotificationService} from '../../notification.service';

@Component({
    selector: 'app-login',
    templateUrl: './login.component.html',
    styleUrls: ['../auth.scss']
})
export class LoginComponent implements OnInit {

    constructor(private authService: AuthService, private notificationService: 
NotificationService, private router: Router, private formBuilder: FormBuilder) {
    }

    loginForm;

    ngOnInit() {
        this.authService.user.subscribe(user => {
            if (user.token) {
                this.router.navigate(['']);
                this.notificationService.notify('Login success', `Welcome to 
RecipeDB, ${user.name}.`);
            }
        });

        this.loginForm = this.formBuilder.group({
            username: ['', Validators.required],
            password: ['', Validators.required]
        });
    }
    login() {
        if (this.loginForm.valid) {
            const {username, password} = this.loginForm.value;
            this.authService.login(username, password).then(res => {
                if (res) {
                    this.notificationService.notify('Login error', res.error);
                }
            });
        }
    }

    isInvalid(control) {
        const {controls} = this.loginForm;
        return controls[control].invalid && controls[control].touched;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\register\register.component
.html

```
<div class="page-header wrap vertical-wrap">
    <h1>Register</h1>
    <h4>Already have an account? <a [routerLink]="'/login'">Login now</a></h4>
</div>
<div class="wrap vertical-wrap">
    <form class="auth-form" [formGroup]="registerForm" (ngSubmit)="register()">
        <div class="form-group">
            <label for="name">Name</label>
            <input type="text" id="name" name="name" class="form-control"
formControlName="name"/>
            <small *ngIf="isInvalid('name')" class="form-text text-muted">
                The name field is required.
            </small>
        </div>
        <div class="form-group">
            <label for="username">Username</label>
            <input type="text" id="username" name="username" class="form-control"
formControlName="username"/>
            <small *ngIf="isInvalid('username')" class="form-text text-muted">
                The username field is required.
            </small>
        </div>
        <div class="form-group">
            <label for="email">Email Address</label>
            <input type="text" id="email" name="email" class="form-control"
formControlName="email"/>
            <small *ngIf="isInvalid('email')" class="form-text text-muted">
                The email field is required.
            </small>
        </div>
        <div class="form-group">
            <label for="password">Password</label>
            <input type="password" id="password" name="password" class="form-
control" formControlName="password"/>
            <small *ngIf="isInvalid('password')" class="form-text text-muted">
               Passwords must be at least 8 characters long.
            </small>
        </div>
        <div class="form-group">
            <label for="passwordConfirm">Confirm Password</label>
            <input type="password" id="passwordConfirm" name="passwordConfirm"
class="form-control"
                   formControlName="passwordConfirm"/>
            <small *ngIf="isInvalid('passwordConfirm')" class="form-text text-
muted">
                Passwords entered do not match.
            </small>
        </div>
        <button type="submit" class="btn btn-block btn-success">Register</button>
    </form>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\auth\register\register.component
.ts

```
import {Component, OnInit} from '@angular/core';
import {Router} from '@angular/router';
import {FormBuilder, FormGroup, ValidationErrors, ValidatorFn, Validators} from
'@angular/forms';

import {AuthService} from '../auth.service';
import {NotificationService} from '../../notification.service';

@Component({
    selector: 'app-register',
    templateUrl: './register.component.html',
    styleUrls: ['../auth.scss']
})
export class RegisterComponent implements OnInit {

    constructor(private authService: AuthService, private notificationService:
NotificationService, private router: Router, private formBuilder: FormBuilder) {
    }

    registerForm: FormGroup;

    ngOnInit() {
        this.authService.user.subscribe(user => {
            if (user.token) {
                this.router.navigate(['']);
            }
        });

        this.registerForm = this.formBuilder.group({
            name: ['', [Validators.required]],
            username: ['', [Validators.required]],
            email: ['', [Validators.required]],
            password: ['', [Validators.required, Validators.minLength(8)]],
            passwordConfirm: ''
        });
        this.registerForm.setValidators(this.passwordMatchValidator());
    }

    register() {
        if (this.registerForm.valid) {
            const {name, username, email, password} = this.registerForm.value;
            this.authService.register(name, username, email, password).then(res =>
{
                if (res) {
                    this.notificationService.notify('Login error', res.error);
                }
            });
        }else{

        }
    }

    passwordMatchValidator(): ValidatorFn {
        return (group: FormGroup): ValidationErrors => {
            const password = group.controls['password'];
            const passwordConfirm = group.controls['passwordConfirm'];
            if (password.value !== passwordConfirm.value) {
                passwordConfirm.setErrors({passwordMismatch: true});
            } else {
                passwordConfirm.setErrors(null);
            }
            return;
        };
    }

    isInvalid(control) {
        const {controls} = this.registerForm;
        return controls[control].invalid && controls[control].touched;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\bookmarks\bookmarks.component.ht
ml

```
<div class="page-header wrap vertical-wrap">
    <h1>Bookmarks</h1>
</div>

<div class="wrap vertical-wrap">
    <div class="recipe-container">
        <div class="text-center" *ngIf="bookmarks.length === 0">
            When you bookmark a recipe it will appear here.
        </div>
        <div *ngFor="let recipe of recipeService.bookmarks | async"
class="recipe">
            <div class="card" [routerLink]="['/recipe', recipe._id]">
                <h5 class="card-header">{{recipe.title}}</h5>
                <div class="card-body">
                    {{recipe.desc}}
                </div>
                <div class="card-footer">
                    {{recipe.calories}} calories
                </div>
            </div>
        </div>
    </div>
```

```
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\bookmarks\bookmarks.component.sc
ss

```
@import '../../app';
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\bookmarks\bookmarks.component.ts

```
import {Component, OnInit} from '@angular/core';
import {Router} from '@angular/router';

import {RecipeService} from '../recipes.service';
import {AuthService} from '../auth/auth.service';

@Component({
    selector: 'app-bookmarks',
    templateUrl: './bookmarks.component.html',
    styleUrls: ['./bookmarks.component.scss']
})
export class BookmarksComponent implements OnInit {

    constructor(private recipeService: RecipeService, private authService:
AuthService, private router: Router) {
    }

    token: string = null;
    bookmarks: [] = [];

    ngOnInit() {
        this.authService.user.subscribe(user => {
            if (user.token) {
                this.token = user.token;
            } else {
                this.router.navigate(['login']);
            }
        });
        this.recipeService.bookmarks.subscribe(bookmarks => {
            this.bookmarks = bookmarks;
        });
        if (this.token) {
            this.recipeService.getBookmarks(this.token);
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\home\home.component.html

```
<div class="hero">
    <div class="wrap">
        <h1>Find your next meal.</h1>
        <app-search-bar></app-search-bar>
    </div>
```

```
</div>
<div class="wrap vertical-wrap top-recipes">
    <h1>This Week's Top Recipes</h1>
    <div class="recipe-container">
        <div *ngFor="let recipe of recipeService.top | async" class="recipe">
            <div class="card" [routerLink]="['/recipe', recipe._id]">
                <h5 class="card-header">{{recipe.title}}</h5>
                <div class="card-body">
                    {{recipe.desc}}
                </div>
                <div class="card-footer">
                    {{recipe.calories}} calories
                </div>
            </div>
        </div>
    </div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\home\home.component.scss

```
@import '../../app';

.hero {
    background: linear-gradient(rgba(0, 0, 0, 0.25), rgba(0, 0, 0, 0.25)),
url('../../assets/hero.jpg');
    background-position: center;
    background-size: cover;
    padding: 8rem 0;
    user-select: none;
    transition: padding $t-reg;

    @media screen {
        @media (min-width: $breakpoint-md) {
            padding: 12rem 0;
        }
        @media (min-width: $breakpoint-lg) {
            padding: 16rem 0;
        }
    }

    h1 {
        color: $white;
        //margin-bottom: 2rem;
    }

    .wrap /deep/ {
        .search-invalid {
            color: $white !important;
        }
    }
}


.top-recipes {
    h1 {
        margin-bottom: 2rem;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\home\home.component.ts

```
import {Component, OnInit} from '@angular/core';

import {RecipeService} from '../recipes.service';

@Component({
    selector: 'app-home',
    templateUrl: './home.component.html',
    styleUrls: ['./home.component.scss']
})
export class HomeComponent implements OnInit {

    constructor(private recipeService: RecipeService) {
    }

    searchBox;

    ngOnInit() {
        this.recipeService.topRecipes();
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\nav\nav.component.html

```
<nav class="navbar navbar-expand-md navbar-light bg-light wrap"
     (clickOutside)="navbarCollapsed === false ? navbarCollapsed = true : null">
    <a class="navbar-brand" [routerLink]="''">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 512 512">
            <path
                d="M416 32a95.17 95.17 0 0 0-57.73 19.74C334.93 20.5 298 0 256 0s-
78.93 20.5-102.27 51.74A95.56 95.56 0 0 0 0 128c0 41.74 64 192 64 192h60.09L112
169.25a8 8 0 0 1 7.33-8.61l16-1.28a8 8 0 0 1 8.61 7.34L156.2 320h83.14V168a8 8 0 0
1 8-8h16a8 8 0 0 1 8 8v152h84.46l12.27-153.3a8 8 0 0 1 8.61-7.34l16 1.28a8 8 0 0 1
7.33 8.61L387.91 320H448s64-150.26 64-192a96 96 0 0 0-96-96zM64 480a32 32 0 0 0 32
32h320a32 32 0 0 0 32-32V352H64z"/>
        </svg>
    </a>
    <button class="navbar-toggler" type="button" (click)="navbarCollapsed =
!navbarCollapsed">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 448 512">
            <path
                d="M442 114H6a6 6 0 0 1-6-6V84a6 6 0 0 1 6-6h436a6 6 0 0 1 6
6v24a6 6 0 0 1-6 6zm0 160H6a6 6 0 0 1-6-6v-24a6 6 0 0 1 6-6h436a6 6 0 0 1 6 6v24a6
6 0 0 1-6 6zm0 160H6a6 6 0 0 1-6-6v-24a6 6 0 0 1 6-6h436a6 6 0 0 1 6 6v24a6 6 0 0
1-6 6z"/>
        </svg>
    </button>
    <div class="collapse navbar-collapse" [ngClass]="navbarCollapsed ? '' :
'show'">
        <ul class="navbar-nav ml-auto">
            <li class="nav-item" *ngIf="user['token'] === null">
```

```
                <a class="nav-link" [routerLink]="'login'"
(click)="navbarCollapsed = true">Login</a>
            </li>
            <li class="nav-item" *ngIf="user['token'] === null">
                <a class="nav-link" [routerLink]="'register'"
(click)="navbarCollapsed = true">Register</a>
            </li>
            <li class="nav-item dropdown" *ngIf="user['token'] !== null"
                (clickOutside)="dropdownCollapsed === false ? dropdownCollapsed =
true : null">
                <a class="nav-link dropdown-toggle" role="button"
(click)="dropdownCollapsed = !dropdownCollapsed">
                    {{user['username']}}
                </a>
                <div class="dropdown-menu dropdown-menu-right"
[ngClass]="dropdownCollapsed ? '' : 'show'">
                    <a class="dropdown-item" [routerLink]="'bookmarks'"
                       (click)="dropdownCollapsed =
!dropdownCollapsed">Bookmarks</a>
                    <a class="dropdown-item" *ngIf="user.admin === true"
[routerLink]="'admin'" (click)="dropdownCollapsed = !dropdownCollapsed">Admin</a>
                    <div class="dropdown-divider"></div>
                    <a class="dropdown-item" (click)="logout()">Log out</a>
                </div>
            </li>
        </ul>
    </div>
</nav>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\nav\nav.component.scss

```
@import '../../app';

.navbar {
    padding-top: 1rem;
    padding-bottom: 1rem;
    user-select: none;

    .navbar-brand {
        background-color: $brand;
        width: 2.5rem;
        height: 2.5rem;
        display: flex;
        align-items: center;
        justify-content: center;
        border-radius: 0.25rem;
        transition: background-color $t-fast;

        svg {
            width: 1.15rem;
            height: 1.15rem;
            fill: #fff;
        }

        &:hover {
            background-color: darken($brand, 10%);
        }
    }

    .navbar-toggler {
        display: flex;
        align-items: center;
        justify-content: center;

        @media screen and (min-width: $breakpoint-md) {
            display: none;
        }

        svg {
            width: 1.25rem;
            height: 1.25rem;
        }
    }

    .navbar-nav {
        margin-top: 1rem;

        .nav-item {
            cursor: pointer;
            font-size: 1.1rem;

            .nav-link {
                color: $dark;
            }

            .dropdown-menu {
                .dropdown-item {
                    color: $dark;

                    &:hover {
                        background-color: darken($white, 5%);
                    }

                    &:active {
                        background-color: darken($white, 10%);
                    }
                }
            }
        }

        @media screen and (min-width: $breakpoint-md) {
            margin-top: unset;
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\nav\nav.component.ts

```
import {Component, OnInit} from '@angular/core';
import {AuthService} from '../auth/auth.service';
import {NotificationService} from '../notification.service';

@Component({
    selector: 'app-nav',
    templateUrl: './nav.component.html',
    styleUrls: ['./nav.component.scss']
})
export class NavComponent implements OnInit {

    constructor(private authService: AuthService, private notificationService:
NotificationService) {
    }

    navbarCollapsed = true;
    dropdownCollapsed = true;
    user;

    ngOnInit() {
        this.authService.user.subscribe(user => {
            this.user = user;
        });
    }

    logout() {
        this.authService.logout().then(res => {
            this.notificationService.notify('Logged out', 'You have successfully
logged out.');
        });
        this.dropdownCollapsed = true;
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\notifications\notifications.comp
onent.html

```
<div class="notifications" *ngIf="notifications !== null">
    <div class="toast show" role="alert" *ngFor="let notification of
notifications">
        <div class="toast-header">
            <strong class="mr-auto">{{notification.title}}</strong>
            <small>{{notification.date}}</small>
            <button type="button" class="ml-2 mb-1 close"
(click)="notificationService.removeNotification(notification.id)">
                <span>&times;</span>
            </button>
        </div>
        <div class="toast-body">
            {{notification.message}}
        </div>
    </div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\notifications\notifications.comp
onent.scss

```
@import '../../app';

.notifications {
    position: absolute;
    right: 2rem;
    bottom: 2rem;
    width: 20rem;
    z-index: 1000;

    @media screen {
        @media (min-width: $breakpoint-sm) {
            right: 4rem;
            bottom: 4rem;
        }
        @media (min-width: $breakpoint-md) {
            right: 4rem;
            bottom: 4rem;
        }
        @media (min-width: $breakpoint-lg) {
            right: 8rem
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\notifications\notifications.comp
onent.ts

```
import {Component, OnInit} from '@angular/core';
import {NotificationService} from '../notification.service';
import * as moment from 'moment';

@Component({
    selector: 'app-notifications',
    templateUrl: './notifications.component.html',
    styleUrls: ['./notifications.component.scss']
})
export class NotificationsComponent implements OnInit {

    constructor(private notificationService: NotificationService) {
    }

    notifications = null;

    ngOnInit() {
        this.notificationService.notifications.subscribe(data => {
            if (JSON.stringify(data) !== JSON.stringify({})) {
                this.notifications = Object.entries(data).map(n => {
                    // @ts-ignore
                    const date = moment(n[1].date).fromNow();
                    // @ts-ignore
                    delete n[1].date;
                    return {id: n[0], ...n[1], date};
                });
            } else {
                this.notifications = null;
            }
        });
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\recipe\recipe.component.html

```
<div class="page-header recipe-header wrap vertical-wrap">
    <div class="top-content">
        <h1>{{recipe.title}}</h1>
        <div class="buttons">
            <button class="btn btn-rdb" (click)="copyLink()" ngbTooltip="Share
link to this recipe">
                <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 512 512">
                    <path
                        d="M503.691 189.836L327.687 37.851C312.281 24.546 288
35.347 288 56.015v80.053C127.371 137.907 0 170.1 0 322.326c0 61.441 39.581 122.309
83.333 154.132 13.653 9.931 33.111-2.533 28.077-18.631C66.066 312.814 132.917
274.316 288 272.085V360c0 20.7 24.3 31.453 39.687 18.164l176.004-152c11.071-9.562
11.086-26.753 0-36.328z"/>
                </svg>
            </button>
            <button class="btn btn-rdb" (click)="bookmark()"
                    *ngIf="token !== null && recipe.hasOwnProperty('bookmarked')
&& recipe.bookmarked === false"
                    ngbTooltip="Bookmark this recipe">
                <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 384 512">
                    <path d="M0 512V48C0 21.49 21.49 0 48 0h288c26.51 0 48 21.49
48 48v464L192 400 0 512z"/>
                </svg>
            </button>
            <button class="btn btn-danger" (click)="unbookmark()"
                    *ngIf="token !== null && recipe.hasOwnProperty('bookmarked')
&& recipe.bookmarked === true"
                    ngbTooltip="Unbookmark this recipe">
                <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 384 512">
                    <path d="M0 512V48C0 21.49 21.49 0 48 0h288c26.51 0 48 21.49
48 48v464L192 400 0 512z"/>
                </svg>
            </button>
            <button class="btn btn-danger" (click)="delete()" *ngIf="token !==
null && admin === true"
                    ngbTooltip="Delete this recipe">
                <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 448 512">
                    <path
                        d="M432 32H312l-9.4-18.7A24 24 0 0 0 281.1 0H166.8a23.72
23.72 0 0 0-21.4 13.3L136 32H16A16 16 0 0 0 0 48v32a16 16 0 0 0 16 16h416a16 16 0
0 0 16-16V48a16 16 0 0 0-16-16zM53.2 467a48 48 0 0 0 47.9 45h245.8a48 48 0 0 0
47.9-45L416 128H32z"/>
                </svg>
            </button>
        </div>
    </div>
    <p class="text-muted">{{recipe.desc}}</p>
    <ngb-rating class="rating" [readonly]="true" [(rate)]="recipe.rating"
[max]="5"></ngb-rating>
</div>
<div class="page-header dark recipe-nutrition wrap">
    <div class="nutrition-item" *ngIf="recipe.calories !== undefined">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 448 512">
            <path
                d="M323.56 51.2c-20.8 19.3-39.58 39.59-56.22 59.97C240.08 73.62
206.28 35.53 168 0 69.74 91.17 0 209.96 0 281.6 0 408.85 100.29 512 224 512s224-
```

```
103.15 224-230.4c0-53.27-51.98-163.14-124.44-230.4zm-19.47 340.65C282.43 407.01
255.72 416 226.86 416 154.71 416 96 368.26 96 290.75c0-38.61 24.31-72.63 72.79-
130.75 6.93 7.98 98.83 125.34 98.83 125.34l58.63-66.88c4.14 6.85 7.91 13.55 11.27
19.97 27.35 52.19 15.81 118.97-33.43 153.42z"/>
        </svg>
        <div class="nutrition-title">Calories</div>
        <div class="nutrition-value">{{recipe.calories}} kcal</div>
    </div>
    <div class="nutrition-item" *ngIf="recipe.protein !== undefined">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 640 512">
            <path
                d="M576 192c35.3 0 64-28.7 64-64s-28.7-64-64-64-64 28.7-64 64 28.7
64 64 64zM64 240c-35.3 0-64 28.7-64 64s28.7 64 64 64 64-28.7 64-64-28.7-64-64-
64zm449.6-37.2l-19.2-25.6-48 36 19.2 25.6 48-36zM576 384c-14.4 0-27.6 5-38.3 13l-
96-57.6c3.8-11.2 6.3-23 6.3-35.5 0-61.9-50.1-112-112-112-8.4 0-16.6 1.1-24.4 2.9l-
40.8-87.4C281.4 96 288 80.8 288 64c0-35.3-28.7-64-64-64s-64 28.7-64 64 28.7 64 64
64c1.1 0 2.1-.3 3.2-.3l41 87.8C241.5 235.9 224 267.8 224 304c0 61.9 50.1 112 112
112 32.1 0 60.8-13.7 81.2-35.3l95.8 57.5c-.5 3.2-1 6.5-1 9.8 0 35.3 28.7 64 64
64s64-28.7 64-64-28.7-64-64-64zm-240-32c-26.5 0-48-21.5-48-48s21.5-48 48-48 48
21.5 48 48-21.5 48-48 48zm-184-32h48v-32h-48v32z"/>
        </svg>
        <div class="nutrition-title">Protein</div>
        <div class="nutrition-value">{{recipe.protein}}g</div>
    </div>
    <div class="nutrition-item" *ngIf="recipe.fat !== undefined">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 352 512">
            <path
                d="M205.22 22.09c-7.94-28.78-49.44-30.12-58.44 0C100.01 179.85 0
222.72 0 333.91 0 432.35 78.72 512 176 512s176-79.65 176-178.09c0-111.75-99.79-
153.34-146.78-311.82zM176 448c-61.75 0-112-50.25-112-112 0-8.84 7.16-16 16-16s16
7.16 16 16c0 44.11 35.89 80 80 80 8.84 0 16 7.16 16 16s-7.16 16-16 16z"/>
        </svg>
        <div class="nutrition-title">Fat</div>
        <div class="nutrition-value">{{recipe.fat}}g</div>
    </div>
    <div class="nutrition-item" *ngIf="recipe.sodium !== undefined">
        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 229.997 229.997">
            <path
                d="M99.291,0.012  c-5.102,0.241-9.73,3.061-12.283,7.484c-
0.457,0.797-0.839,1.633-1.143,2.5H58.649c-0.875-1.519-2.134-2.781-3.65-3.66  c-
4.783-2.761-10.899-1.123-13.66,3.66c-1.749,3.041-1.777,6.775-
0.074,9.842L20.161,55.01c-3.628-0.061-7.004,1.847-8.822,4.986  c-2.761,4.783-
1.123,10.899,3.66,13.66c1.569,0.902,3.353,1.361,5.162,1.328l18.293,30.486c-
2.153,2.599-3.455,5.922-3.455,9.525  c0,3.603,1.301,6.929,3.453,9.527l-
15.465,25.775c-0.966-0.199-1.967-0.303-2.988-0.303c-8.225,0-15,6.775-15,15
c0,8.225,6.775,15,15,15c1.021,0,2.021-0.104,2.986-0.303l15.469,25.777c-
2.153,2.599-3.455,5.922-3.455,9.526  c0,8.225,6.775,15,15,15c6.473,0,12.034-
4.203,14.115-10h21.771c2.081,5.797,7.64,10,14.113,10c8.225,0,15-6.775,15-15  c0-
3.604-1.3-6.928-3.453-9.528l15.467-
25.775c0.966,0.199,1.965,0.303,2.986,0.303c6.473,0,12.034-4.203,14.115-10h27.24
c1.784,3.089,5.078,4.994,8.645,5c5.523,0,10-4.477,10-10c-0.004-1.696-0.439-3.362-
1.264-4.844l18.275-30.457  c0.984,0.2,1.985,0.301,2.988,0.301c8.284,0,15-6.716,15-
15s-6.716-15-15-15c-1.002,0.006-2,0.112-2.98,0.316l-18.277-30.465  c0.824-
1.484,1.257-3.154,1.258-4.852c0-5.523-4.477-10-10-10c-3.568,0.004-6.864,1.91-
8.648,5h-32.705  c-1.784-3.09-5.079-4.995-8.646-5c-0.055,0.003-0.111,0.006-
0.166,0.01l-18.279-30.465c0.534-0.639,1.015-1.322,1.436-2.041  c4.142-7.174,1.684-
16.348-5.49-20.49c-2.488-1.437-5.337-2.129-8.207-1.994L99.291,0.012z
M58.658,19.997h27.205
c1.19,3.368,3.542,6.202,6.635,7.99c3.17,1.822,6.896,2.424,10.479,1.691l18.281,30.4
```

```
67c-0.825,1.484-1.258,3.154-1.26,4.852  c0.004,1.695,0.439,3.361,1.264,4.842l-
18.275,30.461c-0.966-0.199-1.967-0.303-2.988-0.303c-6.473,0-12.032,4.204-
14.113,10H64.113  c-2.082-5.796-7.643-10-14.115-10c-1.021,0-2.02,0.106-
2.986,0.305L28.738,69.848c1.671-3.016,1.671-6.68,0-9.695l21.102-35.17
c3.628,0.061,7.004-1.847,8.822-4.986L58.658,19.997z
M138.649,69.997h32.705c1.784,3.089,5.078,4.994,8.645,5  c0.055-0.003,0.111-
0.006,0.166-0.01l18.273,30.457c-2.221,2.687-3.437,6.063-
3.439,9.549c0.004,3.485,1.222,6.86,3.443,9.545  L180.162,155c-0.055-0.004-0.109-
0.007-0.164-0.01c-3.568,0.004-6.864,1.91-8.648,5h-27.236c-2.082-5.796-7.643-10-
14.115-10  c-1.022,0-2.021,0.106-2.988,0.305l-15.465-25.775c2.154-2.599,3.453-
5.925,3.453-9.529c0-3.604-1.301-6.928-3.455-9.527  l18.291-
30.48c0.055,0.004,0.109,0.007,0.164,0.01c3.569-0.004,6.866-1.909,8.65-
5L138.649,69.997z M49.998,109.997  c2.821,0,5,2.179,5,5c0,2.821-2.179,5-5,5c-
2.821,0-5-2.179-5-5C44.998,112.176,47.178,109.997,49.998,109.997z M99.998,109.997
c2.821,0,5,2.179,5,5c0,2.821-2.179,5-5,5c-2.821,0-5-2.179-5-
5C94.998,112.176,97.178,109.997,99.998,109.997z M64.113,119.997
h21.771c2.081,5.797,7.64,10,14.113,10c1.021,0,2.022-0.104,2.988-
0.303l15.467,25.777c-2.153,2.599-3.455,5.922-3.455,9.525
c0,3.603,1.301,6.929,3.453,9.527l-15.465,25.775c-0.966-0.199-1.967-0.303-2.988-
0.303c-6.473,0-12.032,4.204-14.113,10H64.113  c-2.082-5.796-7.643-10-14.115-10c-
1.022,0-2.021,0.106-2.988,0.305l-15.465-25.775c2.154-2.599,3.453-5.925,3.453-9.529
s-1.3-6.928-3.453-9.527l15.467-
25.775c0.966,0.199,1.965,0.303,2.986,0.303C56.472,129.997,62.032,125.794,64.113,11
9.997  L64.113,119.997z M19.998,159.997c2.821,0,5,2.179,5,5c0,2.821-2.179,5-5,5c-
2.821,0-5-2.179-5-5  C14.998,162.176,17.178,159.997,19.998,159.997z
M129.998,159.997c2.821,0,5,2.179,5,5c0,2.821-2.179,5-5,5c-2.821,0-5-2.179-5-5
C124.998,162.176,127.178,159.997,129.998,159.997z
M49.998,209.997c2.821,0,5,2.179,5,5c0,2.821-2.179,5-5,5c-2.821,0-5-2.179-5-5
S47.178,209.997,49.998,209.997z M99.998,209.997c2.821,0,5,2.179,5,5c0,2.821-
2.179,5-5,5c-2.821,0-5-2.179-5-5  S97.178,209.997,99.998,209.997z"/>
        </svg>
        <div class="nutrition-title">Sodium</div>
        <div class="nutrition-value">{{recipe.sodium}}mg</div>
    </div>
</div>
<div class="wrap vertical-wrap recipe-content" *ngIf="recipe !== null">
    <div class="recipe-ingredients">
        <h2>Ingredients</h2>
        <ul *ngIf="recipe.ingredients" class="ingredients">
            <li *ngFor="let ingredient of recipe.ingredients; let i = index"
class="custom-control custom-checkbox">
                <input type="checkbox" class="custom-control-input"
[id]="['ingredient-',i]">
                <label class="custom-control-label" [htmlFor]="['ingredient-',
i]">{{ingredient}}</label>
            </li>
        </ul>
    </div>
    <div class="recipe-directions">
        <h2>Directions</h2>
        <ol *ngIf="recipe.directions" class="directions">
            <li *ngFor="let direction of recipe.directions; let i = index">
                <span class="number">{{i + 1}}.</span>
                <p>{{direction}}</p>
            </li>
        </ol>
    </div>
    <div class="recipe-comments">
        <h2>Comments</h2>
        <ul class="list-group">
            <form class="list-group-item comment-box" *ngIf="token !== null"
[formGroup]="commentBox"
                  (ngSubmit)="comment()">
                <textarea formControlName="body"></textarea>
                <button class="btn btn-rdb">Comment</button>
            </form>
            <div class="list-group-item" *ngIf="comments.length === 0">No comments
yet, <span *ngIf="token === null"><a
                [routerLink]="'/login'">log in</a> to </span>be the first to add
one.
            </div>
            <div *ngFor="let comment of comments" class="list-group-item comment">
                <div class="comment-header">
                    <div class="name">{{comment.user_name}}</div>
                    <div class="right">
                        <div class="date">{{timeAgo(comment.date)}}</div>
                        <div
                            *ngIf="token !== null && comment.user_id === userId"
                            class="btn"
                            (click)="commentUpdate === comment._id ? commentUpdate
= null : commentUpdate = comment._id; commentUpdateValue = comment.body"
                            ngbTooltip="Update this comment"
                        >
                            <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0
576 512">
                                <path
                                    d="M402.6 83.2l90.2 90.2c3.8 3.8 3.8 10 0
13.8L274.4 405.6l-92.8 10.3c-12.4 1.4-22.9-9.1-21.5-21.5l10.3-92.8L388.8 83.2c3.8-
3.8 10-3.8 13.8 0zm162-22.9l-48.8-48.8c-15.2-15.2-39.9-15.2-55.2 0l-35.4 35.4c-3.8
3.8-3.8 10 0 13.8l90.2 90.2c3.8 3.8 10 3.8 13.8 0l35.4-35.4c15.2-15.3 15.2-40 0-
55.2zM384 346.2V448H64V128h229.8c3.2 0 6.2-1.3 8.5-3.5l40-40c7.6-7.6 2.2-20.5-8.5-
20.5H48C21.5 64 0 85.5 0 112v352c0 26.5 21.5 48 48 48h352c26.5 0 48-21.5 48-
48V306.2c0-10.7-12.9-16-20.5-8.5l-40 40c-2.2 2.3-3.5 5.3-3.5 8.5z"/>
                            </svg>
                        </div>
                        <div
                            *ngIf="token !== null && comment.user_id === userId"
                            class="btn"
                            (click)="deleteComment(comment._id)"
                            ngbTooltip="Delete this comment"
                        >
                            <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0
448 512">
                                <path
                                    d="M432 32H312l-9.4-18.7A24 24 0 0 0 281.1
0H166.8a23.72 23.72 0 0 0-21.4 13.3L136 32H16A16 16 0 0 0 0 48v32a16 16 0 0 0 16
16h416a16 16 0 0 0 16-16V48a16 16 0 0 0-16-16zM53.2 467a48 48 0 0 0 47.9
45h245.8a48 48 0 0 0 47.9-45L416 128H32z"/>
                            </svg>
                        </div>
                    </div>
                </div>
                <div>{{comment.body}}</div>
            </div>
            <form class="list-group-item comment-box" *ngIf="token !== null &&
commentUpdate !== null" [formGroup]="updateCommentBox"
                  (ngSubmit)="updateComment()">
                <h4 class="mb-3">Update Comment</h4>
                <textarea formControlName="body"
[value]="commentUpdateValue"></textarea>
                <button class="btn btn-rdb">Update</button>
            </form>
        </ul>
    </div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\recipe\recipe.component.scss

```
@import '../../app';

.recipe-header {
    .top-content {
        display: flex;
        flex-direction: row;
        align-items: center;
        justify-content: space-between;
        margin-bottom: 1rem;

        h1 {
            margin-bottom: unset;
        }

        .buttons {
            display: flex;
            flex-direction: row;

            button {
                margin-left: 1rem;
            }
        }
    }

    p {
        margin-bottom: unset;
    }

    .rating {
        user-select: none;
        pointer-events: none;
        font-size: 1.5rem;
    }
}

.recipe-nutrition {
    display: flex;
    flex-direction: column;
    flex-wrap: wrap;

    .nutrition-item {
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: center;
        width: 100%;
        padding: 2rem;
```

```scss
        transition: width $t-reg ease-out;

        svg {
            fill: darken($white, 75%);
            width: 3rem;
            height: 3rem;
        }

        .nutrition-title {
            text-transform: uppercase;
            letter-spacing: 0.35rem;
            color: darken($white, 75%);
            opacity: 0.75;
            padding: 0.5rem 0 0;
        }

        .nutrition-value {
            color: darken($white, 75%);
            font-weight: bold;
            font-size: 1.1rem;
        }
    }

    @media screen {
        @media (min-width: $breakpoint-sm) {
            flex-direction: row;

            .nutrition-item {
                width: 50%;
            }
        }
        @media (min-width: $breakpoint-md) {
            .nutrition-item {
                width: 25%;
            }
        }
    }
}

.recipe-content {
    display: flex;
    flex-direction: column;
    flex-wrap: wrap;

    .recipe-ingredients, .recipe-directions {
        width: 100%;

        h2 {
            margin-bottom: 1rem;
            transition: margin-bottom $t-fast;
        }
    }

    .recipe-ingredients {
        margin-bottom: 2rem;
    }

    @media screen and(min-width: $breakpoint-lg) {
        flex-direction: row;
        .recipe-ingredients, .recipe-directions {
            width: 50%;

            h2 {
                margin-bottom: 2rem;
            }
        }
        .recipe-ingredients {
            padding-right: 1rem;
            margin-bottom: unset;
        }
        .recipe-directions {
            padding-left: 1rem;
        }
    }

    ul.ingredients {
        padding-left: 0;
        margin: 0;

        .custom-control-input:checked ~ .custom-control-label::before {
            border-color: $brand;
            background-color: $brand;
        }

        li {
            padding-top: 0.15rem;
            padding-bottom: 0.15rem;
        }
    }

    ol.directions {
        list-style-type: none;
        padding-left: 0;

        li {
            display: flex;
            flex-direction: row;
            margin-bottom: 1rem;

            .number {
                width: 4rem;
                font-size: 3rem;
                font-weight: 300;
                line-height: 3rem;
                padding-right: 1rem;
                user-select: none;
            }

            p, &:last-of-type {
                margin-bottom: 0;
            }
        }
    }

    .recipe-comments {
        width: 100%;
```

```
h2 {
    margin: 2rem 0;
}

.comment-box {
    display: flex;
    flex-direction: column;
    padding: 1rem;

    textarea {
        width: 100%;
        height: 5rem;
        min-height: 5rem;
        max-height: 10rem;
        padding: 1rem;
        border: 1px solid $border;
        border-radius: 0.25rem;
        margin-bottom: 1rem;
    }

    button {
        margin-left: auto;
    }
}

.comment {
    display: flex;
    flex-direction: column;

    .comment-header {
        display: flex;
        align-items: center;
        justify-content: space-between;

        .name {
            font-weight: bold;
        }

        .right {
            display: flex;
            align-items: center;
            justify-content: center;

            .date {
                opacity: 0.5;
                user-select: none;
            }

            .btn {
                margin-left: 1rem;
                padding: 0.25rem 0.5rem;
                border: 1px solid lighten($error, 10%);
                transition: background-color $t-reg, border-color $t-reg;

                svg {
                    fill: lighten($error, 10%);
                    transition: fill $t-reg;
                }
```

```
                        &:hover {
                            border-color: $error;
                            background-color: $error;

                            svg {
                                fill: $white;
                            }
                        }
                    }
                }
            }

            .body {

            }
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\recipe\recipe.component.ts

```
import {Component, OnInit} from '@angular/core';
import {ActivatedRoute, Router} from '@angular/router';
import {FormBuilder, Validators} from '@angular/forms';
import {isEqual} from 'lodash';
import * as moment from 'moment';

import {AuthService, getDefaultUserObject} from '../auth/auth.service';
import {RecipeService} from '../recipes.service';
import {NotificationService} from '../notification.service';

@Component({
    selector: 'app-recipe',
    templateUrl: './recipe.component.html',
    styleUrls: ['./recipe.component.scss']
})
export class RecipeComponent implements OnInit {

    constructor(private recipeService: RecipeService,
                private authService: AuthService,
                private notificationService: NotificationService,
                private router: Router,
                private route: ActivatedRoute,
                private formBuilder: FormBuilder) {
        this.recipe = null;
        this.comments = [];
    }

    recipe: any = null;
    comments: [] = null;
    commentBox;
    updateCommentBox;
    token: string = null;
    userId: string = null;
    admin: boolean = false;
    commentUpdate: string = null;
    commentUpdateValue: string = null;
    ngOnInit() {
        const {id} = this.route.snapshot.params;

        this.commentBox = this.formBuilder.group({
            body: ''
        });
        this.updateCommentBox = this.formBuilder.group({
            body: ['', Validators.required]
        });

        this.authService.user.subscribe(user => {
            if (!isEqual(user, getDefaultUserObject())) {
                this.token = user.token;
                this.userId = user.id;
                if (user.admin) {
                    this.admin = true;
                }
            } else {
                this.token = null;
            }
        });
        this.recipeService.data.subscribe(data => {
            if (data === null) {
                this.router.navigate(['']);
            } else {
                this.recipe = data;
            }
        });
        this.recipeService.comments.subscribe(comments => {
            this.comments = comments;
        });

        this.recipeService.getRecipe(id, this.token);
        this.recipeService.getRecipeComments(id);
    }

    comment() {
        const {id} = this.route.snapshot.params;
        const {body} = this.commentBox.value;
        this.recipeService.comment(id, body, this.token);
        this.commentBox = this.formBuilder.group({
            body: ''
        });
    }

    updateComment() {
        if (this.updateCommentBox.valid) {
            const {body} = this.updateCommentBox.value;
            this.recipeService.updateComment(this.commentUpdate, body,
this.token);
            this.commentUpdate = null;
            this.commentUpdateValue = null;
        }
    }

    deleteComment(id: string) {
        this.recipeService.deleteComment(id, this.token).then(res => {
            if (res) {
                if ((res as any).message) {
                    this.notificationService.notify('Comment deleted', (res as 
any).message);
                }
                if ((res as any).error) {
                    this.notificationService.notify('Error', (res as any).error);
                }
            }
        });
    }

    bookmark() {
        const {id} = this.route.snapshot.params;
        this.recipeService.bookmarkRecipe(id, this.token).then(res => {
            if (res) {
                if ((res as any).message) {
                    this.notificationService.notify('Bookmark added', (res as 
any).message);
                }
                if ((res as any).error) {
                    this.notificationService.notify('Error', (res as any).error);
                }
            }
        });
    }

    unbookmark() {
        const {id} = this.route.snapshot.params;
        this.recipeService.unbookmarkRecipe(id, this.token).then(res => {
            if (res) {
                if ((res as any).message) {
                    this.notificationService.notify('Bookmark removed', (res as 
any).message);
                }
                if ((res as any).error) {
                    this.notificationService.notify('Error', (res as any).error);
                }
            }
        });
    }

    copyLink() {
        const link = document.createElement('textarea');
        link.style.position = 'fixed';
        link.style.left = '0';
        link.style.top = '0';
        link.style.opacity = '0';
        link.value = 
`http://localhost:4200/recipe/${this.route.snapshot.params.id}`;
        document.body.appendChild(link);
        link.focus();
        link.select();
        document.execCommand('copy');
        document.body.removeChild(link);
        this.notificationService.notify('Link copied', 'A link to this recipe has 
been copied to your clipboard.');
    }

    delete() {
        const {id} = this.route.snapshot.params;
        this.recipeService.deleteRecipe(id, this.token).then(res => {
            if (res) {
                if ((res as any).message) {
                    this.notificationService.notify('Recipe deleted', (res as
any).message);
                }
                if ((res as any).error) {
                    this.notificationService.notify('Error', (res as any).error);
                }
            }
        });
    }

    timeAgo(date) {
        return moment(date).fromNow();
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\search.component.html

```
<div class="page-header wrap vertical-wrap">
    <h1>Search "{{route.snapshot.params.criteria}}"</h1>
</div>
<div class="page-header dark wrap">
    <app-search-bar></app-search-bar>
</div>
<div class="wrap vertical-wrap">
    <div class="text-center" *ngIf="results.data && results.data.length === 0">
        No results found for "{{route.snapshot.params.criteria}}", try searching
for something else.
    </div>
    <div class="recipe-container" *ngIf="results.data && results.data.length > 0">
        <div *ngFor="let recipe of results.data" class="recipe">
            <div class="card" [routerLink]="['/recipe', recipe._id]">
                <h5 class="card-header">{{recipe.title}}</h5>
                <div class="card-body">
                    {{recipe.desc}}
                </div>
                <div class="card-footer">
                    {{recipe.calories}} calories
                </div>
            </div>
        </div>
    </div>
    <div class="pagination" *ngIf="results.data && results.data.length > 0">
        <pagination-controls
(pageChange)="goToPage(route.snapshot.params.criteria, $event)"
                             class="pagination"></pagination-controls>
        <div>
            <div class="page-item" *ngFor="let item of results.data | paginate: {
              itemsPerPage:results.perPage,
              currentPage: results.page,
              totalItems: results.total
        }">

            </div>
        </div>
    </div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\search.component.scss

```
@import '../../app';

$border-radius: 0.25rem;
$padding: 0.5rem 1rem;

.pagination /deep/ {
    pagination-controls {
        ul.ngx-pagination {
            margin: 0;
            padding: 0;

            li {
                border: 1px solid $border;
                border-left: 0;
                margin: 0;

                a {
                    padding: $padding;
                }

                &:first-of-type {
                    border-left: 1px solid $border;
                    border-top-left-radius: $border-radius;
                    border-bottom-left-radius: $border-radius;
                }

                &:last-of-type {
                    border-top-right-radius: $border-radius;
                    border-bottom-right-radius: $border-radius;
                }

                &.disabled {
                    padding: $padding;
                    cursor: not-allowed;
                }

                &.current {
                    padding: $padding;
                    background-color: $brand;
                }
            }
        }
    }
}

.pagination {
    display: flex;
    flex-direction: row;
    align-items: center;
    justify-content: center;
    margin-top: 2rem;
    user-select: none;

    .page-item {
        cursor: pointer;

        .page-link {
            color: $dark;

            svg {
                width: 1rem;
                height: 1rem;
                fill: $dark;
            }
        }

        &.disabled {
            cursor: not-allowed;

            svg {
                fill: lighten($dark, 30%);
            }
        }

        &.active {
            .page-link {
                background-color: $brand;
                color: $white;
                cursor: default;
            }
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\search.component.ts

```
import {Component, OnInit} from '@angular/core';
import {ActivatedRoute, Router} from '@angular/router';
import {FormBuilder} from '@angular/forms';
import {range} from 'lodash';

import {RecipeService} from '../recipes.service';

@Component({
    selector: 'app-search',
    templateUrl: './search.component.html',
    styleUrls: ['./search.component.scss']
})
export class SearchComponent implements OnInit {

    constructor(private recipeService: RecipeService, private router: Router,
private route: ActivatedRoute, private formBuilder: FormBuilder) {
        this.results = {
            data: null,
            total: 0,
            page: 0,
        };
        this.pagination = [];
    }

    searchBox;
    results;
    pagination;

    ngOnInit() {
        const {criteria, page} = this.route.snapshot.params;
        if (page) {
            this.recipeService.searchRecipes(criteria, page);
        } else {
            this.recipeService.searchRecipes(criteria);
        }

        this.recipeService.search.subscribe(search => {
            this.results = search;
            this.pagination = range(search.pageCount);
        });

        this.searchBox = this.formBuilder.group({criteria});
    }

    goToPage(criteria, page) {
        if (page > 1) {
            this.recipeService.searchRecipes(criteria, page);
            this.router.navigate(['recipes', 'search', criteria, page]);

        } else {
            this.recipeService.searchRecipes(criteria);
            this.router.navigate(['recipes', 'search', criteria]);
        }
        window.scroll(0, 0);
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\bar\bar.component.html

```
<div class="search-container" [ngClass]="{'invalid': isInvalid('criteria')}">
    <form [formGroup]="searchForm" (ngSubmit)="search()" class="search">
        <input
            type="search"
            name="criteria"
            formControlName="criteria"
            placeholder="Search recipes.."
            id="input-criteria"
        />
        <button type="submit">
            <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 512 512">
                <path
                    d="M505 442.7L405.3 343c-4.5-4.5-10.6-7-17-7H372c27.6-35.3 44-
79.7 44-128C416 93.1 322.9 0 208 0S0 93.1 0 208s93.1 208 208 208c48.3 0 92.7-16.4
128-44v16.3c0 6.4 2.5 12.5 7 17l99.7 99.7c9.4 9.4 24.6 9.4 33.9 0l28.3-28.3c9.4-
9.4 9.4-24.6.1-34zM208 336c-70.7 0-128-57.2-128-128 0-70.7 57.2-128 128-128 70.7 0
128 57.2 128 128 0 70.7-57.2 128-128 128z"/>
            </svg>
        </button>
    </form>
    <div id="search-invalid" class="search-invalid" *ngIf="isInvalid('criteria')">
Please enter three or more characters</div>
</div>
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\bar\bar.component.scss

```
@import '../../../app';

.search-container {
    position: relative;
    padding-top: 2rem;
    padding-bottom: 2rem;

    .search {
        display: flex;
        border-radius: 0.5rem;
        border: 1px solid $border;
        box-shadow: none;
        transition: border-color $t-fast, box-shadow $t-fast;

        input {
            width: 100%;
            padding: 1rem;
            border: 0;
            border-top-left-radius: 0.5rem;
            border-bottom-left-radius: 0.5rem;
        }

        button {
            display: flex;
            align-items: center;
            justify-content: center;
            background-color: $white;
            padding: 0 1.5rem;
            border: 0;
            border-left: 1px solid $border;
            border-top-right-radius: 0.5rem;
            border-bottom-right-radius: 0.5rem;
            cursor: pointer;
            transition: background-color $t-fast;

            svg {
                width: 1.5rem;
                height: 1.5rem;
            }

            &:hover {
                background-color: darken($white, 10%);
            }
        }
    }

    .search-invalid {
        position: absolute;
        bottom: 0;
        display: none;
        align-items: center;
        height: 2rem;
        color: $error;
    }

    &.invalid {
        .search {
            border: 1px solid $error;
            box-shadow: 0 0 10px $error;

            button {
                border-left: 1px solid $error;
                box-shadow: 0 0 10px $error;
            }
        }

        .search-invalid {
            display: flex;
        }
    }
}
```

C:\Users\rob\Documents\GitHub\RecipeDB\ui\src\app\search\bar\bar.component.ts

```
import {Component, OnInit} from '@angular/core';
import {ActivatedRoute, Router} from '@angular/router';
import {FormBuilder, FormGroup, Validators} from '@angular/forms';
import {RecipeService} from '../../recipes.service';

@Component({
    selector: 'app-search-bar',
    templateUrl: './bar.component.html',
    styleUrls: ['./bar.component.scss']
})
export class BarComponent implements OnInit {

    constructor(private recipeService: RecipeService, private route:
ActivatedRoute, private router: Router, private formBuilder: FormBuilder) {
        this.searchForm = this.formBuilder.group({
            criteria: ['', [Validators.required, Validators.minLength(3)]]
        });
    }

    searchForm: FormGroup;
    criteria: string = null;

    ngOnInit() {
        try {
            const urlIndex = (index) => {
                if (url[index] && url[index].path) {
                    return url[index].path;
                } else {
                    return null;
                }
            };
            const {url, params} = this.route.snapshot;
            if (urlIndex(0) === 'recipes' && urlIndex(1) === 'search') {
                this.searchForm.controls.criteria.setValue(params.criteria);
            }
        } catch (e) {
            console.warn(e);
        }
    }

    search() {
        document.getElementById('input-criteria').focus();
        if (this.searchForm.valid) {
            const {criteria} = this.searchForm.value;
            if (criteria !== this.criteria) {
                this.router.navigate(['recipes', 'search', criteria]).then(() => {
                    this.recipeService.searchRecipes(criteria);
                });
            }
        }
    }

    isInvalid(control) {
        const {controls} = this.searchForm;
        if (controls[control].value.length > 0) {
            return controls[control].invalid && controls[control].touched;
        } else {
            return false;
        }
    }
}
```