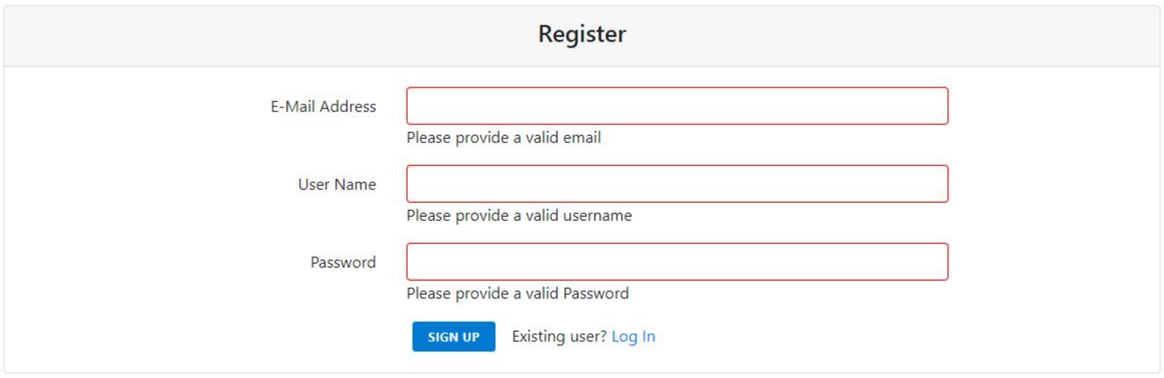
My project is Reddit Clone, it has a few functionalities like creating new post or subreddit, downvoting or upvoting recent posts, you also can comment or read entire post. I'll show all implemented functionalities later on.I created this project using Angular.


Register

E-Mail Address

Please provide a valid email

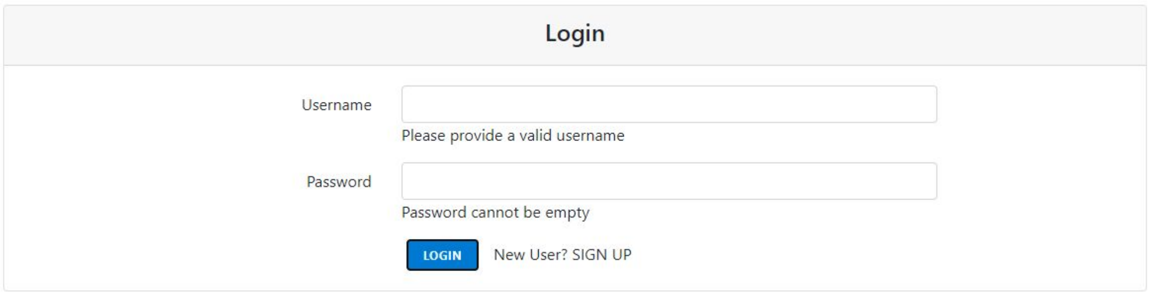
User Name

Please provide a valid username

Password

Please provide a valid Password

Here we have registration screen, with basic validation, when user won't write anything in any field.


Login

Username

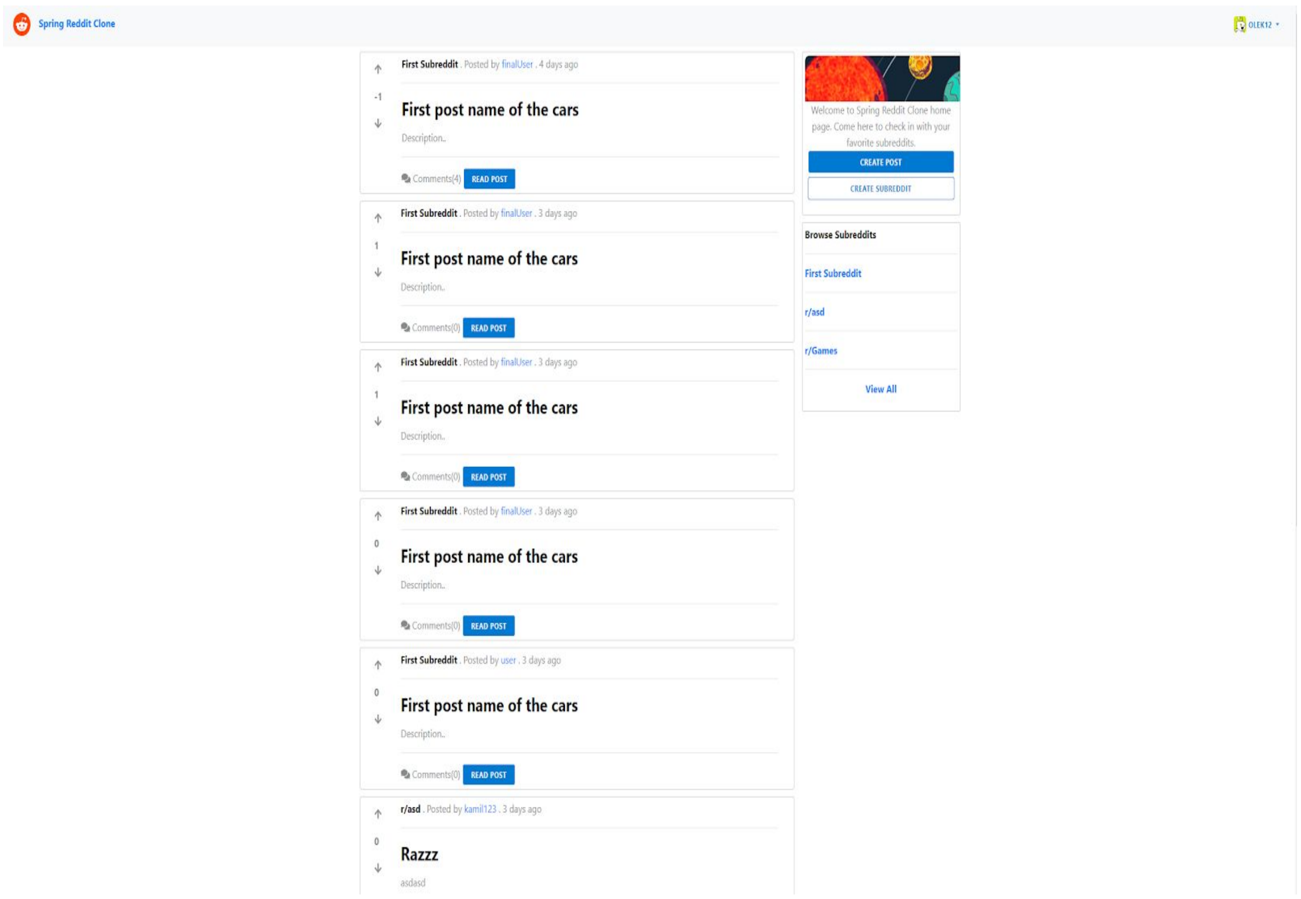
Please provide a valid username

Password

Password cannot be empty

And so is Login screen.

After proper authorization, we are moved to a main page of our application.



Spring Reddit Clone

OLEK12

First Subreddit . Posted by finalUser . 4 days ago

-1

First post name of the cars

Description..

Comments(4) READ POST

First Subreddit . Posted by finalUser . 3 days ago

1

First post name of the cars

Description..

Comments(0) READ POST

First Subreddit . Posted by finalUser . 3 days ago

1

First post name of the cars

Description..

Comments(0) READ POST

First Subreddit . Posted by finalUser . 3 days ago

0

First post name of the cars

Description..

Comments(0) READ POST

First Subreddit . Posted by user . 3 days ago

0

First post name of the cars

Description..

Comments(0) READ POST

r/asd . Posted by kamil123 . 3 days ago

0

Razzz

asdasd

Welcome to Spring Reddit Clone home page. Come here to check in with your favorite subreddits.

CREATE POST

CREATE SUBREDDIT

Browse Subreddits

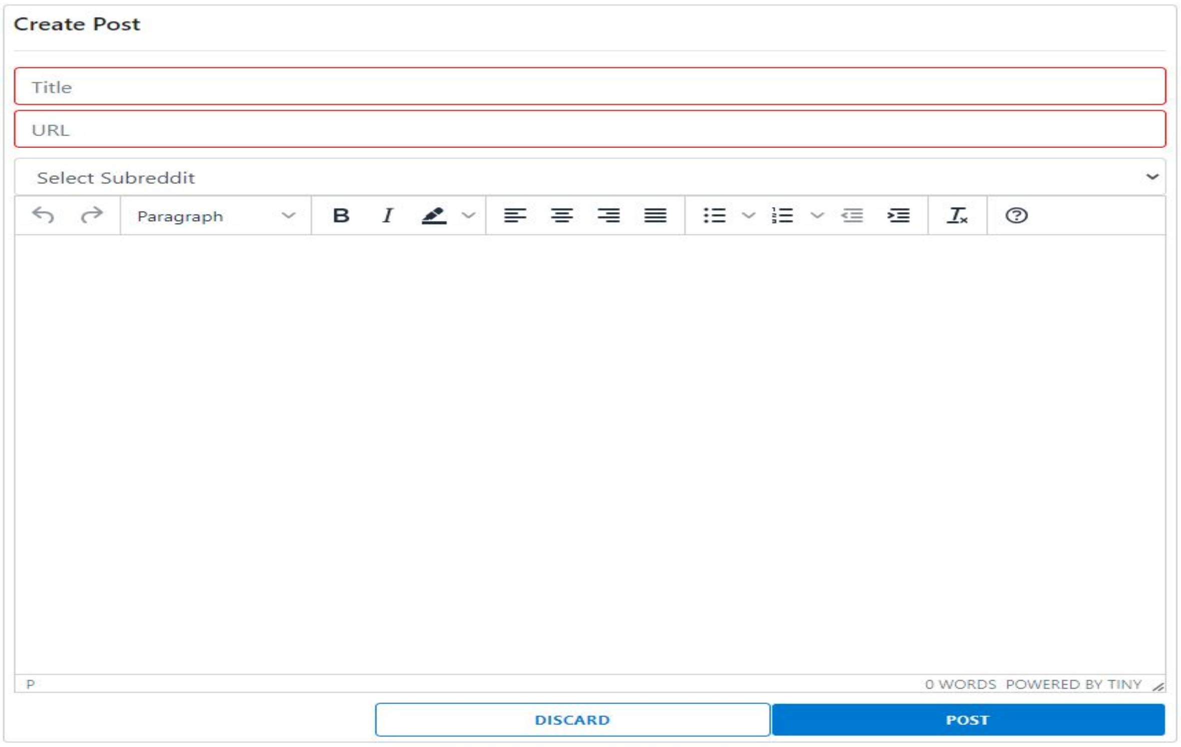
First Subreddit

r/asd

r/Games

View All

Here we can see all created posts all users made, we can sort it by Subreddits, downvote it or upvote using arrows which are on the left side of each post.



Create Post

Title

URL

Select Subreddit

Paragraph

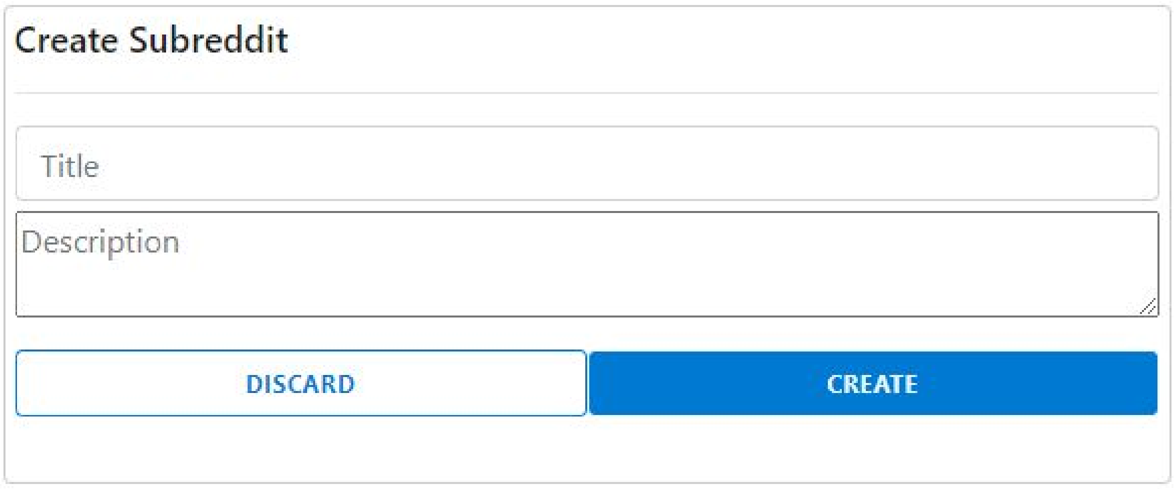
P

0 WORDS POWERED BY TINY

DISCARD

POST

After clicking on “Create Post” button we are transfer to Create Post page, where we can create post with Title, Url on selected Subreddit. It also have a validation.

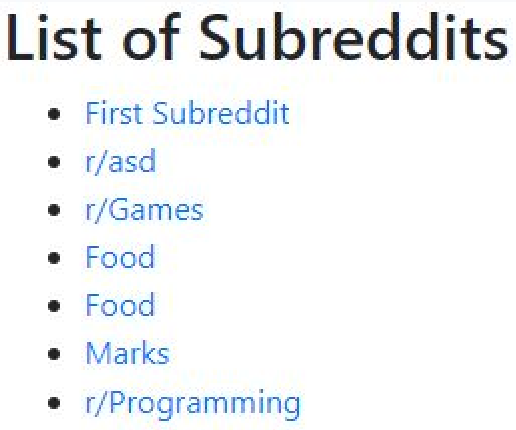

Create Subreddit

Title

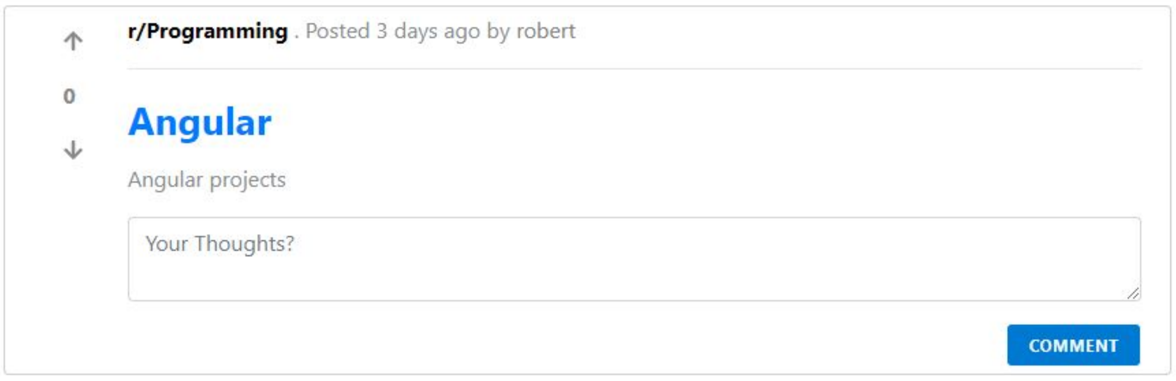
Description

Create Subreddit page looks very similar to previous example.

View All button redirect us to the list of all existing subreddits.


List of Subreddits

- First Subreddit
- r/asd
- r/Games
- Food
- Food
- Marks
- r/Programming

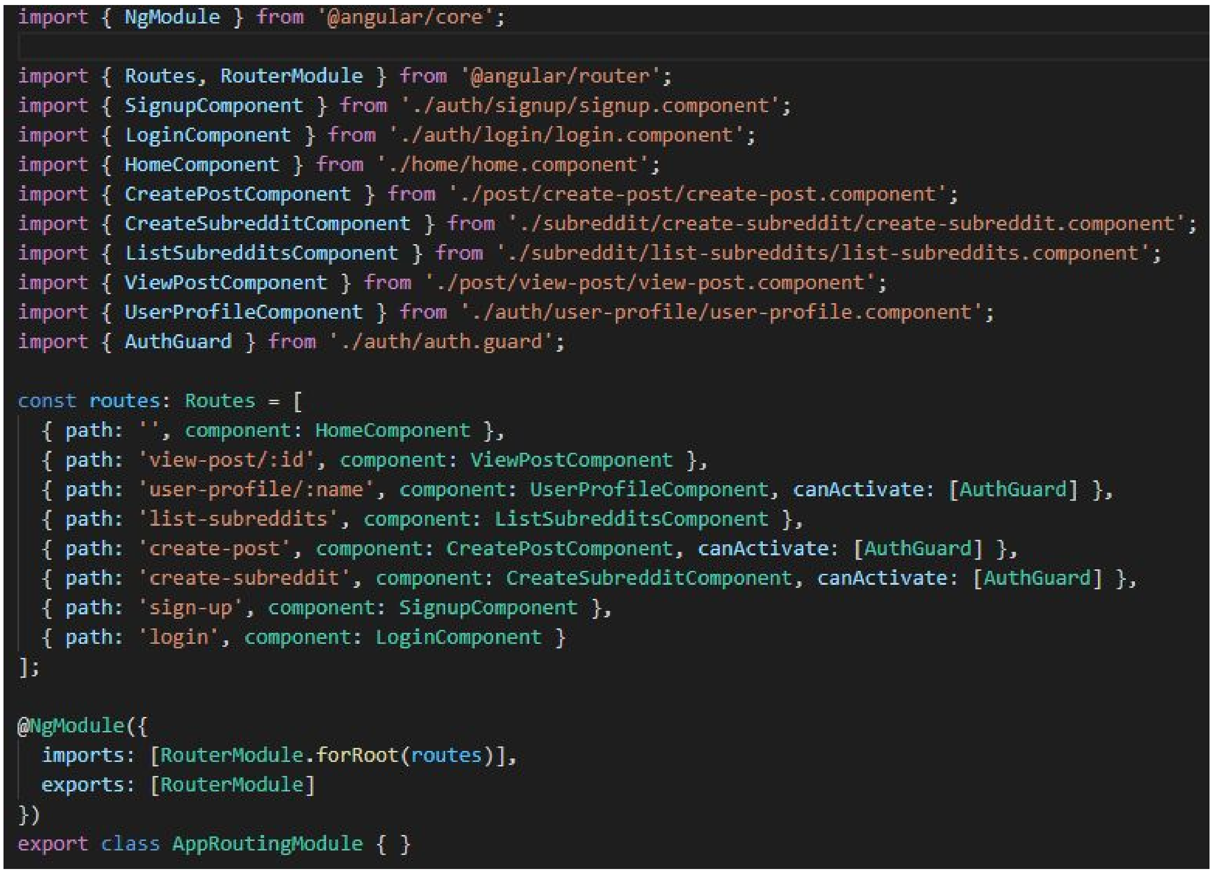

r/Programming . Posted 3 days ago by robert

0

Angular

Angular projects

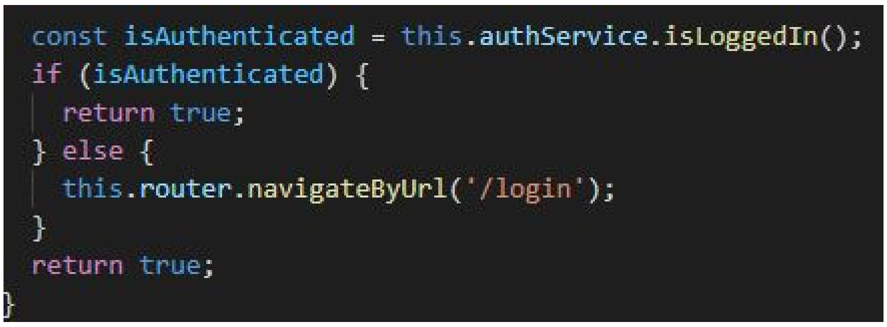
Your Thoughts?



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

import { Routes, RouterModule } from '@angular/router';
import { SignupComponent } from './auth/signup/signup.component';
import { LoginComponent } from './auth/login/login.component';
import { HomeComponent } from './home/home.component';
import { CreatePostComponent } from './post/create-post/create-post.component';
import { CreateSubredditComponent } from './subreddit/create-subreddit/create-subreddit.component';
import { ListSubredditsComponent } from './subreddit/list-subreddits/list-subreddits.component';
import { ViewPostComponent } from './post/view-post/view-post.component';
import { UserProfileComponent } from './auth/user-profile/user-profile.component';
import { AuthGuard } from './auth/auth.guard';

const routes: Routes = [
  { path: '', component: HomeComponent },
  { path: 'view-post/:id', component: ViewPostComponent },
  { path: 'user-profile/:name', component: UserProfileComponent, canActivate: [AuthGuard] },
  { path: 'list-subreddits', component: ListSubredditsComponent },
  { path: 'create-post', component: CreatePostComponent, canActivate: [AuthGuard] },
  { path: 'create-subreddit', component: CreateSubredditComponent, canActivate: [AuthGuard] },
  { path: 'sign-up', component: SignupComponent },
  { path: 'login', component: LoginComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

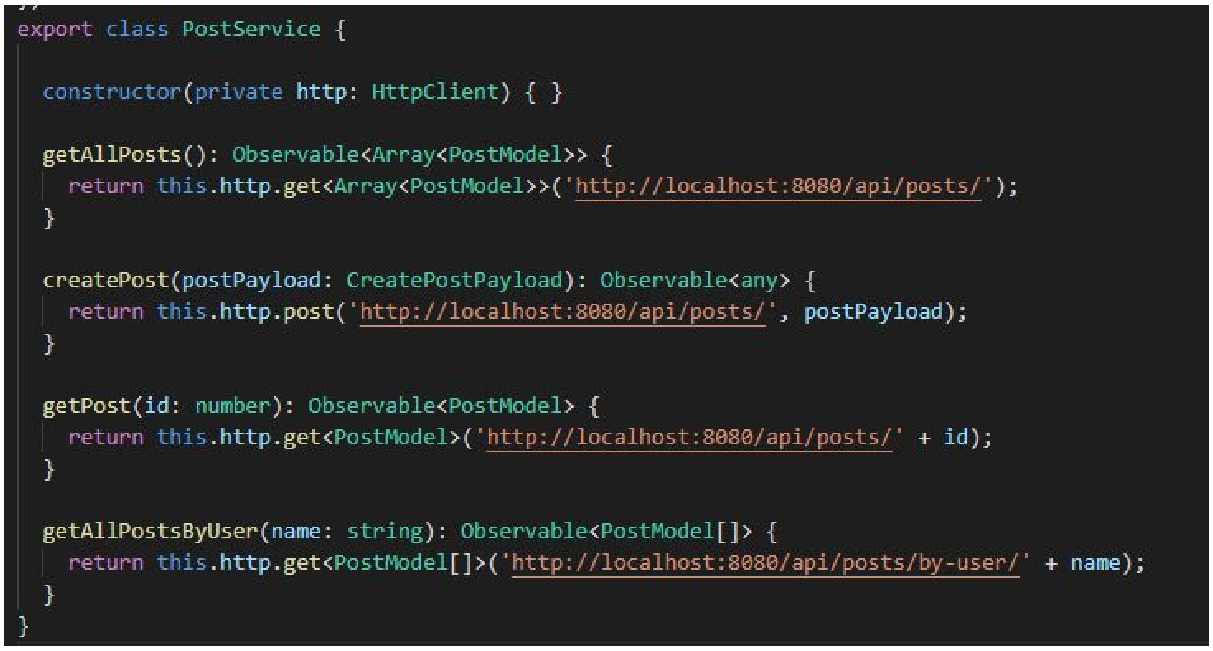
Here we have app-routing module, which shows all Routes i use in my application, as we can see some routes required authorization, such as creating post, or creating subreddit.

AuthGuard is checking for proper authentication and return true if so.



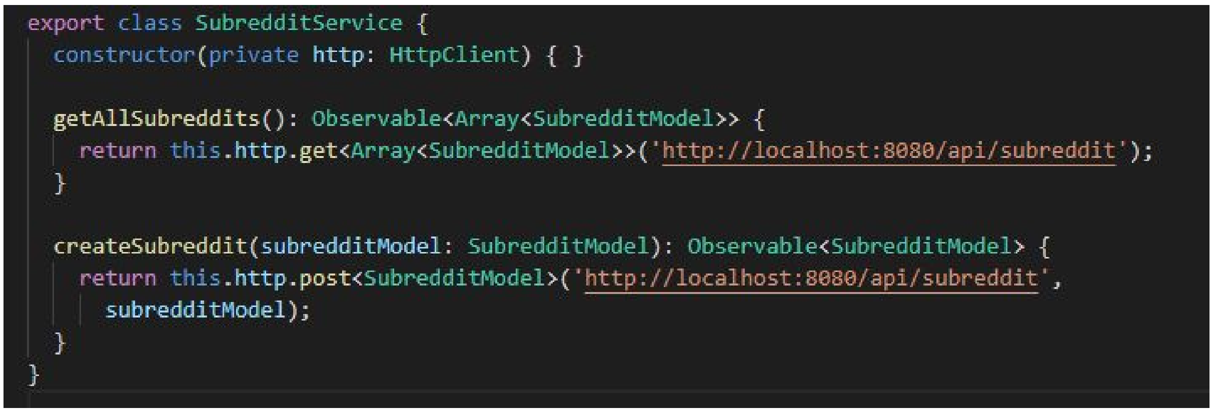
```
 const isAuthenticated = this.authService.isLoggedIn();
 if (isAuthenticated) {
   return true;
 } else {
   this.router.navigateByUrl('/login');
 }
 return true;
}
```

Each of action made has his service.ts file to communicate with API. For example he is a post creating service



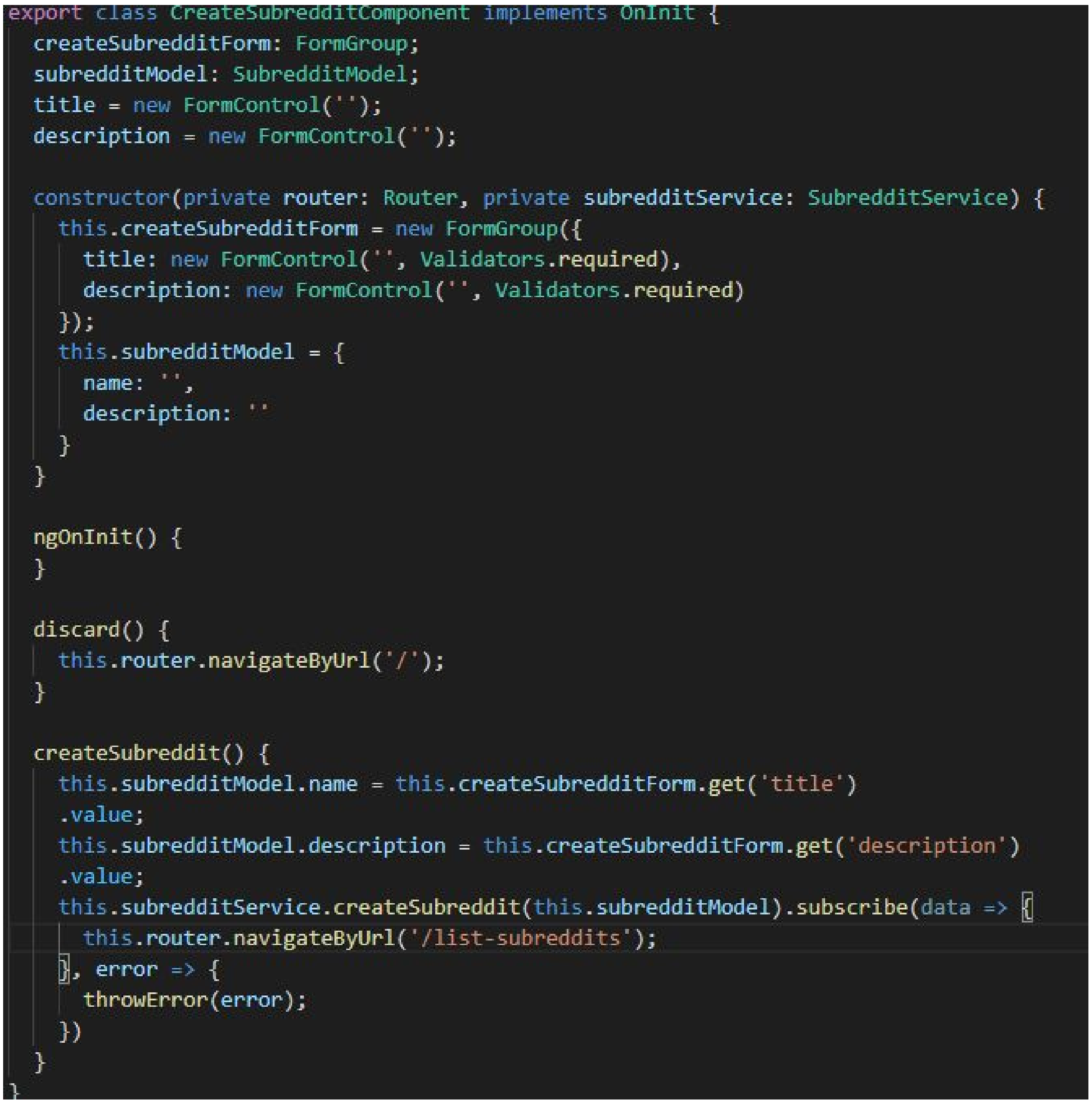
```
export class PostService {

  constructor(private http: HttpClient) { }

  getAllPosts(): Observable<Array<PostModel>> {
    return this.http.get<Array<PostModel>>('http://localhost:8080/api/posts/');
  }

  createPost(postPayload: CreatePostPayload): Observable<any> {
    return this.http.post('http://localhost:8080/api/posts/', postPayload);
  }

  getPost(id: number): Observable<PostModel> {
    return this.http.get<PostModel>('http://localhost:8080/api/posts/' + id);
  }

  getAllPostsByUser(name: string): Observable<PostModel[]> {
    return this.http.get<PostModel[]>('http://localhost:8080/api/posts/by-user/' + name);
  }
}
```

and subreddit service



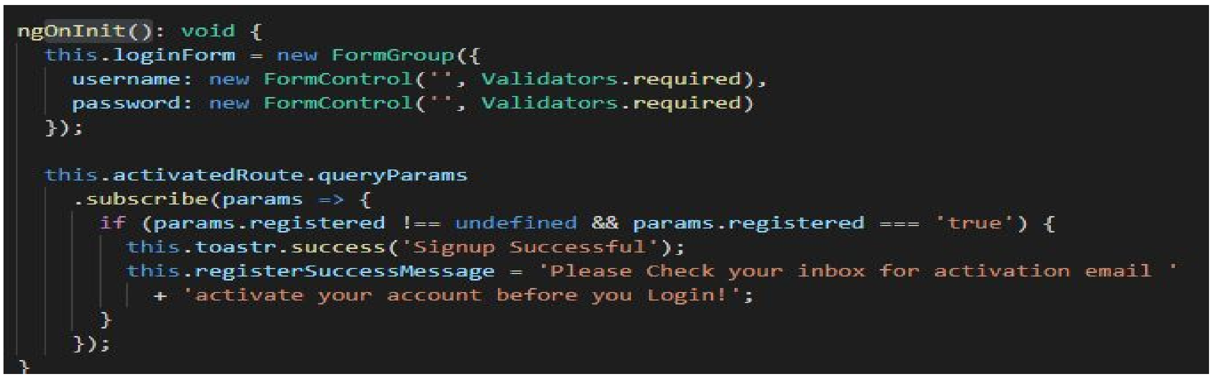
```
export class SubredditService {
  constructor(private http: HttpClient) { }

  getAllSubreddits(): Observable<Array<SubredditModel>> {
    return this.http.get<Array<SubredditModel>>('http://localhost:8080/api/subreddit');
  }

  createSubreddit(subredditModel: SubredditModel): Observable<SubredditModel> {
    return this.http.post<SubredditModel>('http://localhost:8080/api/subreddit',
      subredditModel);
  }
}
```

It get us all of subreddits and store it in an array. So is posting.



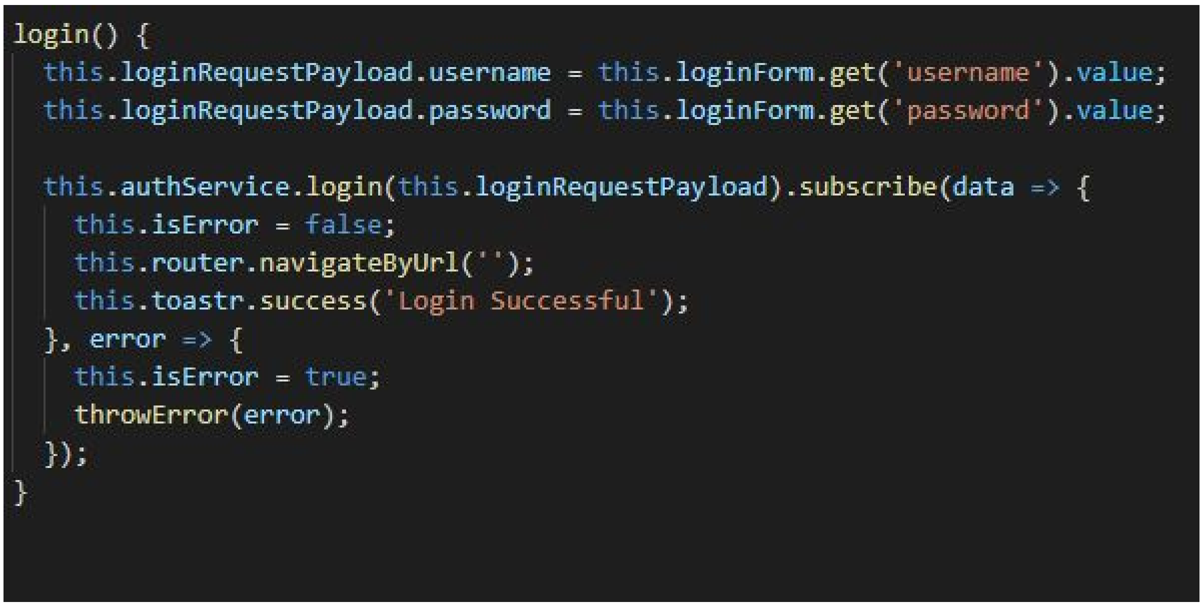
```
export class CreateSubredditComponent implements OnInit {
  createSubredditForm: FormGroup;
  subredditModel: SubredditModel;
  title = new FormControl('');
  description = new FormControl('');

  constructor(private router: Router, private subredditService: SubredditService) {
    this.createSubredditForm = new FormGroup({
      title: new FormControl('', Validators.required),
      description: new FormControl('', Validators.required)
    });
    this.subredditModel = {
      name: '',
      description: ''
    }
  }

  ngOnInit() {
  }

  discard() {
    this.router.navigateByUrl('/');
  }

  createSubreddit() {
    this.subredditModel.name = this.createSubredditForm.get('title')
    .value;
    this.subredditModel.description = this.createSubredditForm.get('description')
    .value;
    this.subredditService.createSubreddit(this.subredditModel).subscribe(data => {
      this.router.navigateByUrl('/list-subreddits');
    }, error => {
      throwError(error);
    })
  }
}
```

CreateSubredditComponent has a constructor, which validates the input, and two options, if we discard it, we are navigated to a main page, or when we createSubreddit, it gets us values from form and navitates it to the /list-subreddits page.

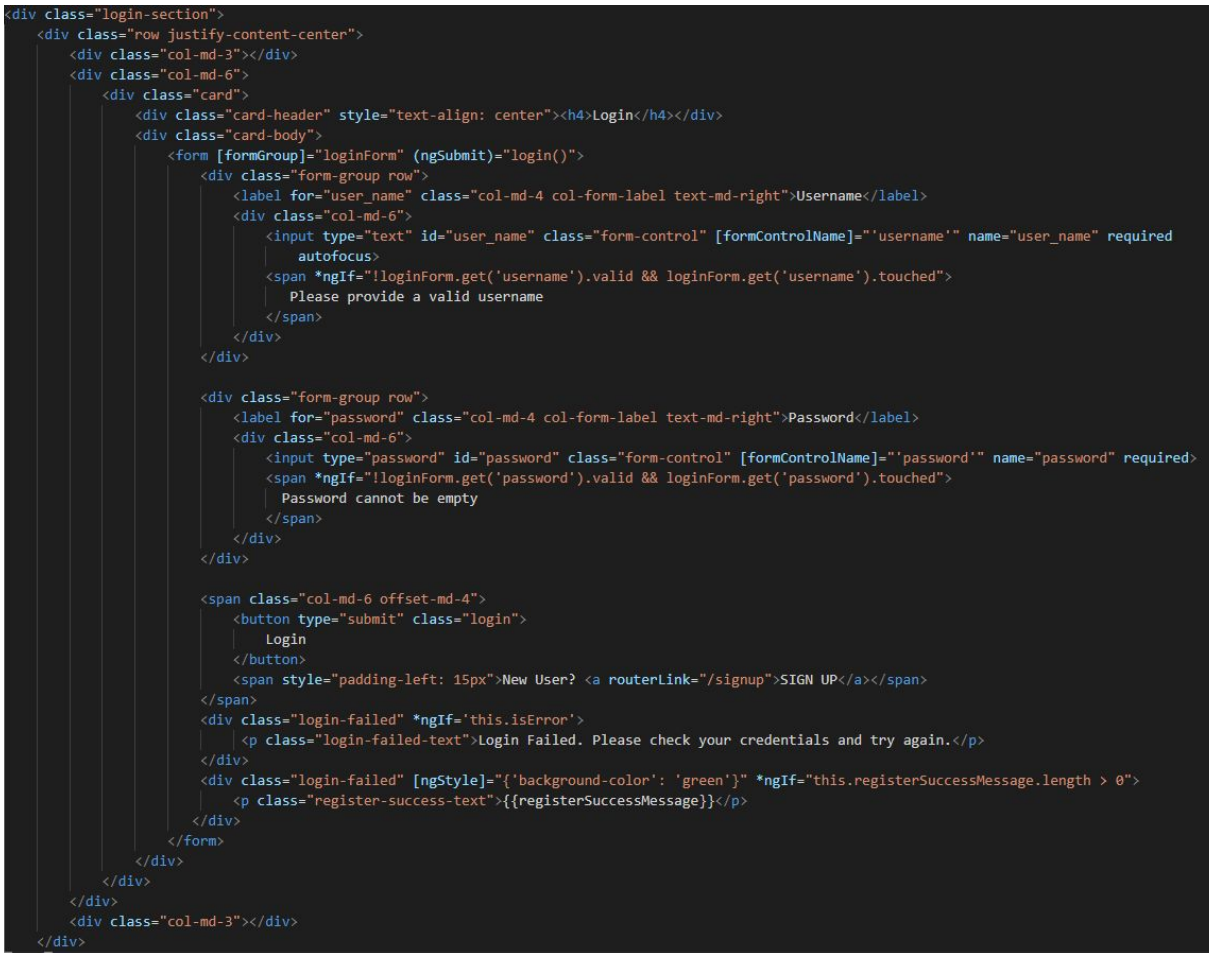


```
ngOnInit(): void {
  this.loginForm = new FormGroup({
    username: new FormControl('', Validators.required),
    password: new FormControl('', Validators.required)
  });

  this.activatedRoute.queryParams
    .subscribe(params => {
      if (params.registered !== undefined && params.registered === 'true') {
        this.toastr.success('Signup Successful');
        this.registerSuccessMessage = 'Please Check your inbox for activation email '
          + 'activate your account before you Login!';
      }
    });
}
```

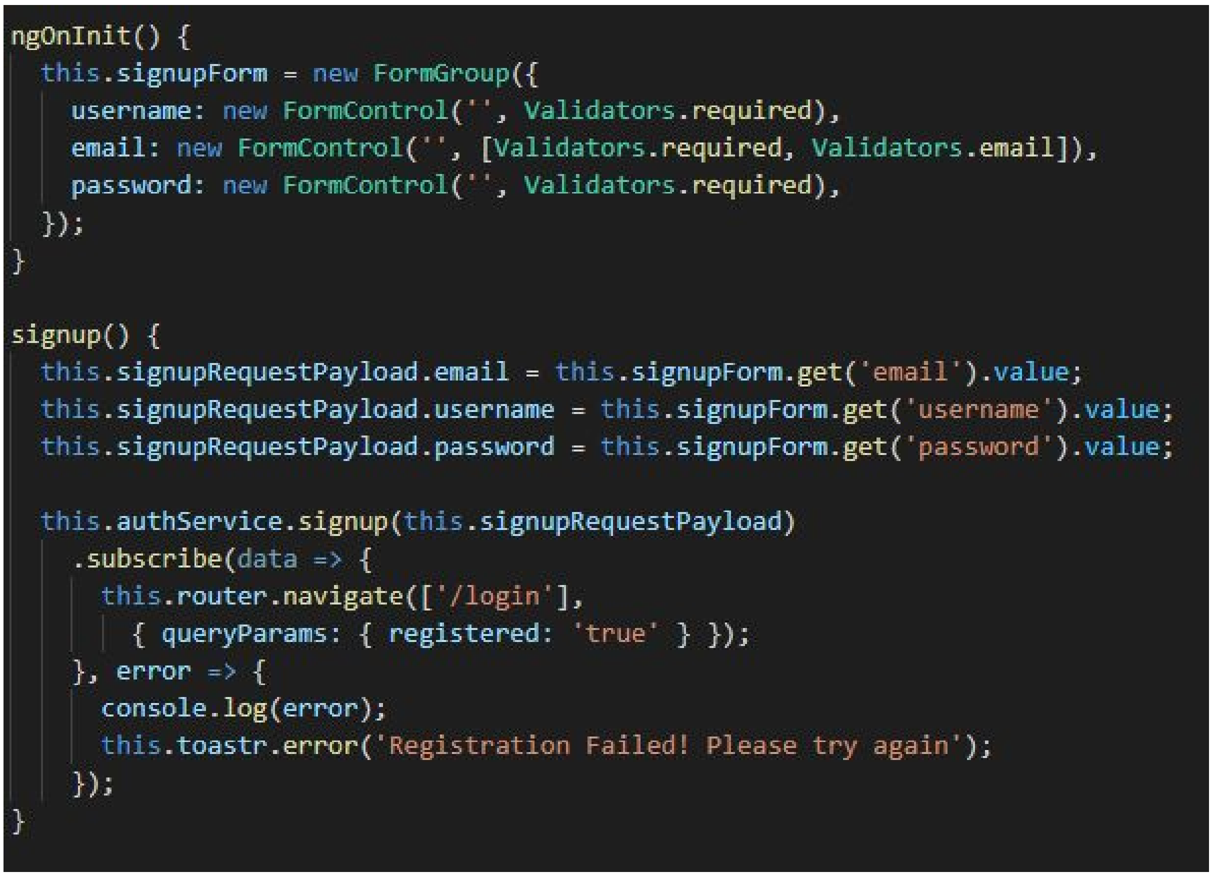
Here is a validation of input and if all went good, we got a notification, that registration went successful.



```
login() {
  this.loginRequestPayload.username = this.loginForm.get('username').value;
  this.loginRequestPayload.password = this.loginForm.get('password').value;

  this.authService.login(this.loginRequestPayload).subscribe(data => {
    this.isError = false;
    this.router.navigateByUrl('');
    this.toastr.success('Login Successful');
  }, error => {
    this.isError = true;
    throwError(error);
  });
}
```

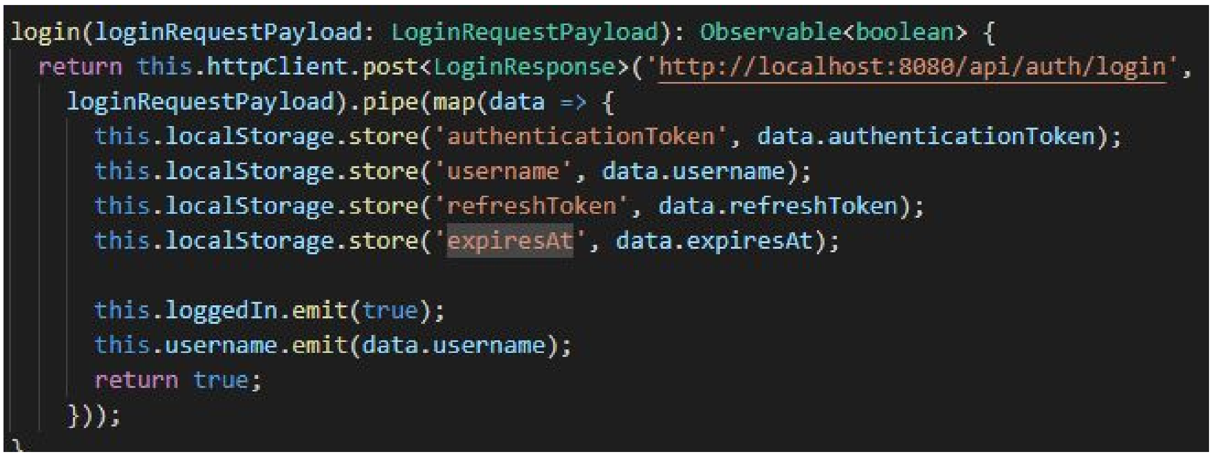


```
<div class="login-section">
    <div class="row justify-content-center">
        <div class="col-md-3"></div>
        <div class="col-md-6">
            <div class="card">
                <div class="card-header" style="text-align: center"><h4>Login</h4></div>
                <div class="card-body">
                    <form [formGroup]="loginForm" (ngSubmit)="login()">
                        <div class="form-group row">
                            <label for="user_name" class="col-md-4 col-form-label text-md-right">Username</label>
                            <div class="col-md-6">
                                <input type="text" id="user_name" class="form-control" [formControlName]="'username'" name="user_name" required
                                    autofocus>
                                <span *ngIf="!loginForm.get('username').valid && loginForm.get('username').touched">
                                   Please provide a valid username
                                </span>
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="password" class="col-md-4 col-form-label text-md-right">Password</label>
                            <div class="col-md-6">
                                <input type="password" id="password" class="form-control" [formControlName]="'password'" name="password" required>
                                <span *ngIf="!loginForm.get('password').valid && loginForm.get('password').touched">
                                  Password cannot be empty
                                </span>
                            </div>
                        </div>

                        <span class="col-md-6 offset-md-4">
                            <button type="submit" class="login">
                                Login
                            </button>
                            <span style="padding-left: 15px">New User? <a routerLink="/signup">SIGN UP</a></span>
                        </span>
                        <div class="login-failed" *ngIf='this.isError'>
                             <p class="login-failed-text">Login Failed. Please check your credentials and try again.</p>
                        </div>
                        <div class="login-failed" [ngStyle]="{'background-color': 'green'}" *ngIf="this.registerSuccessMessage.length > 0">
                            <p class="register-success-text">{{registerSuccessMessage}}</p>
                       </div>
                    </form>
                </div>
            </div>
        </div>
        <div class="col-md-3"></div>
    </div>
```

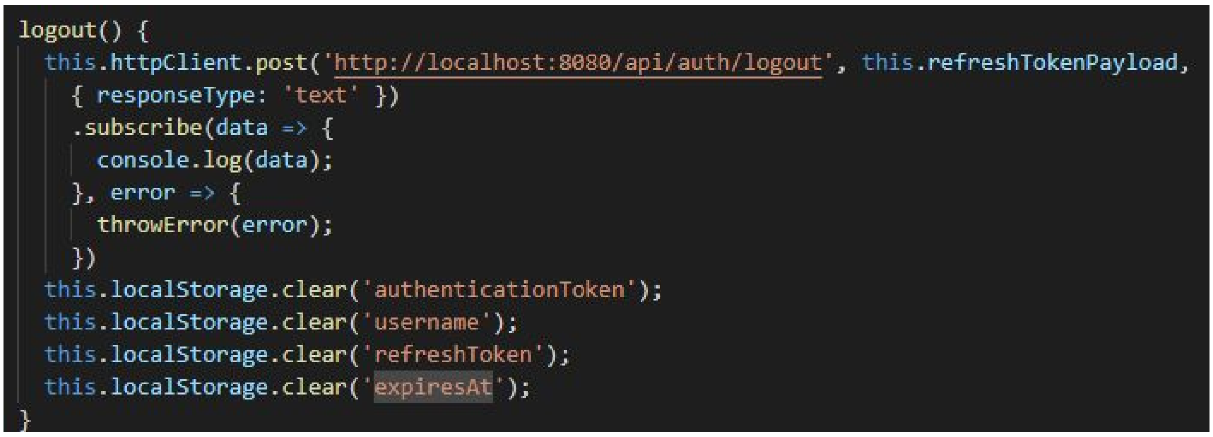


```
ngOnInit() {
  this.signupForm = new FormGroup({
    username: new FormControl('', Validators.required),
    email: new FormControl('', [Validators.required, Validators.email]),
    password: new FormControl('', Validators.required),
  });
}

signup() {
  this.signupRequestPayload.email = this.signupForm.get('email').value;
  this.signupRequestPayload.username = this.signupForm.get('username').value;
  this.signupRequestPayload.password = this.signupForm.get('password').value;

  this.authService.signup(this.signupRequestPayload)
    .subscribe(data => {
      this.router.navigate(['/login'],
        { queryParams: { registered: 'true' } });
    }, error => {
      console.log(error);
      this.toastr.error('Registration Failed! Please try again');
    });
}
```

Login function in auth.service.



```
login(loginRequestPayload: LoginRequestPayload): Observable<boolean> {
  return this.httpClient.post<LoginResponse>('http://localhost:8080/api/auth/login',
    loginRequestPayload).pipe(map(data => {
      this.localStorage.store('authenticationToken', data.authenticationToken);
      this.localStorage.store('username', data.username);
      this.localStorage.store('refreshToken', data.refreshToken);
      this.localStorage.store('expiresAt', data.expiresAt);

      this.loggedIn.emit(true);
      this.username.emit(data.username);
      return true;
    }));
```



```
logout() {
  this.httpClient.post('http://localhost:8080/api/auth/logout', this.refreshTokenPayload,
    { responseType: 'text' })
    .subscribe(data => {
      console.log(data);
    }, error => {
      throwError(error);
    })
  this.localStorage.clear('authenticationToken');
  this.localStorage.clear('username');
  this.localStorage.clear('refreshToken');
  this.localStorage.clear('expiresAt');
}
```

logging out, clearing our localStorage.