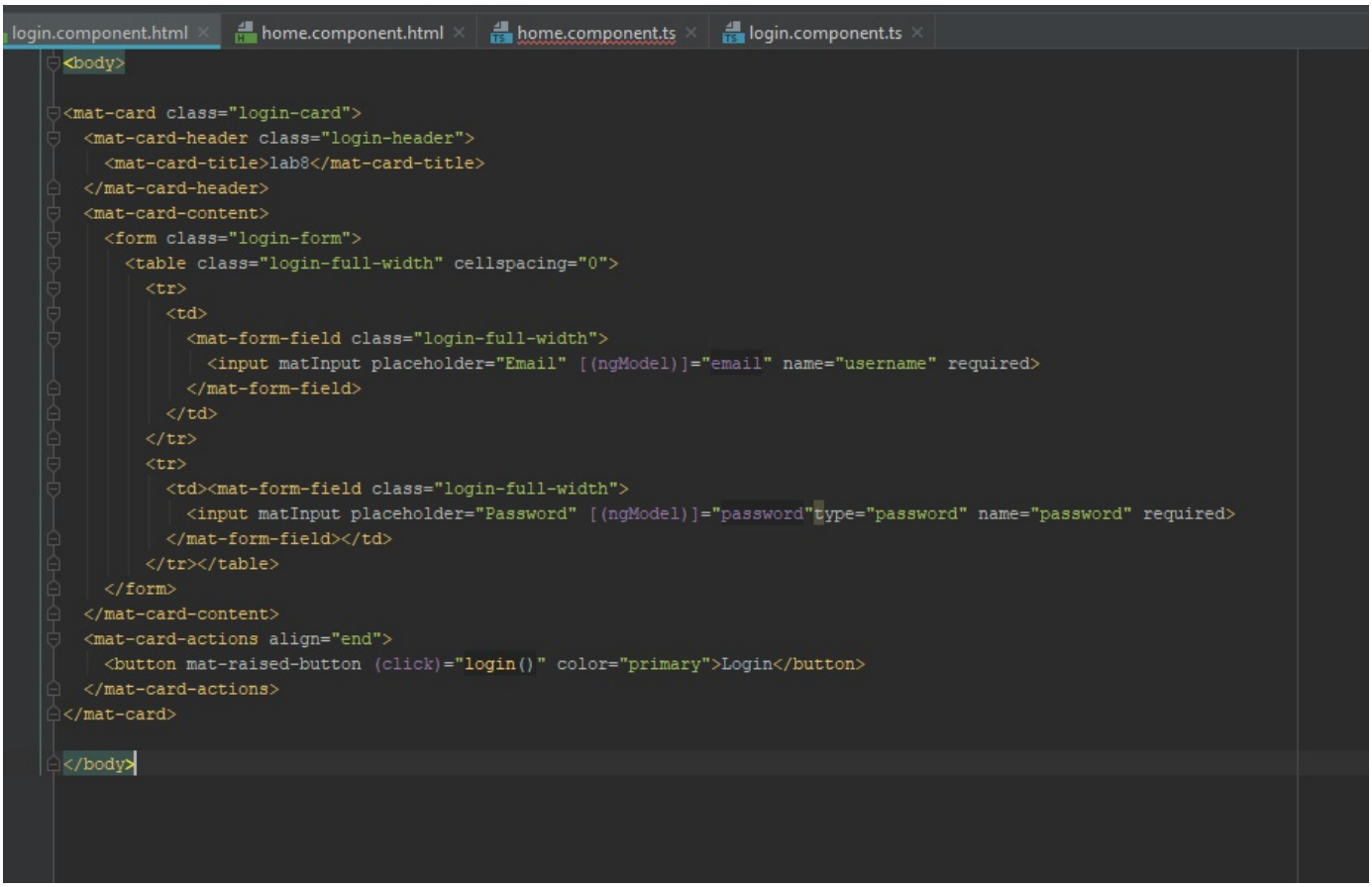
Objective

Understand the lab-8 source code and write an app to perform user authentication (email, password) using JWT token.

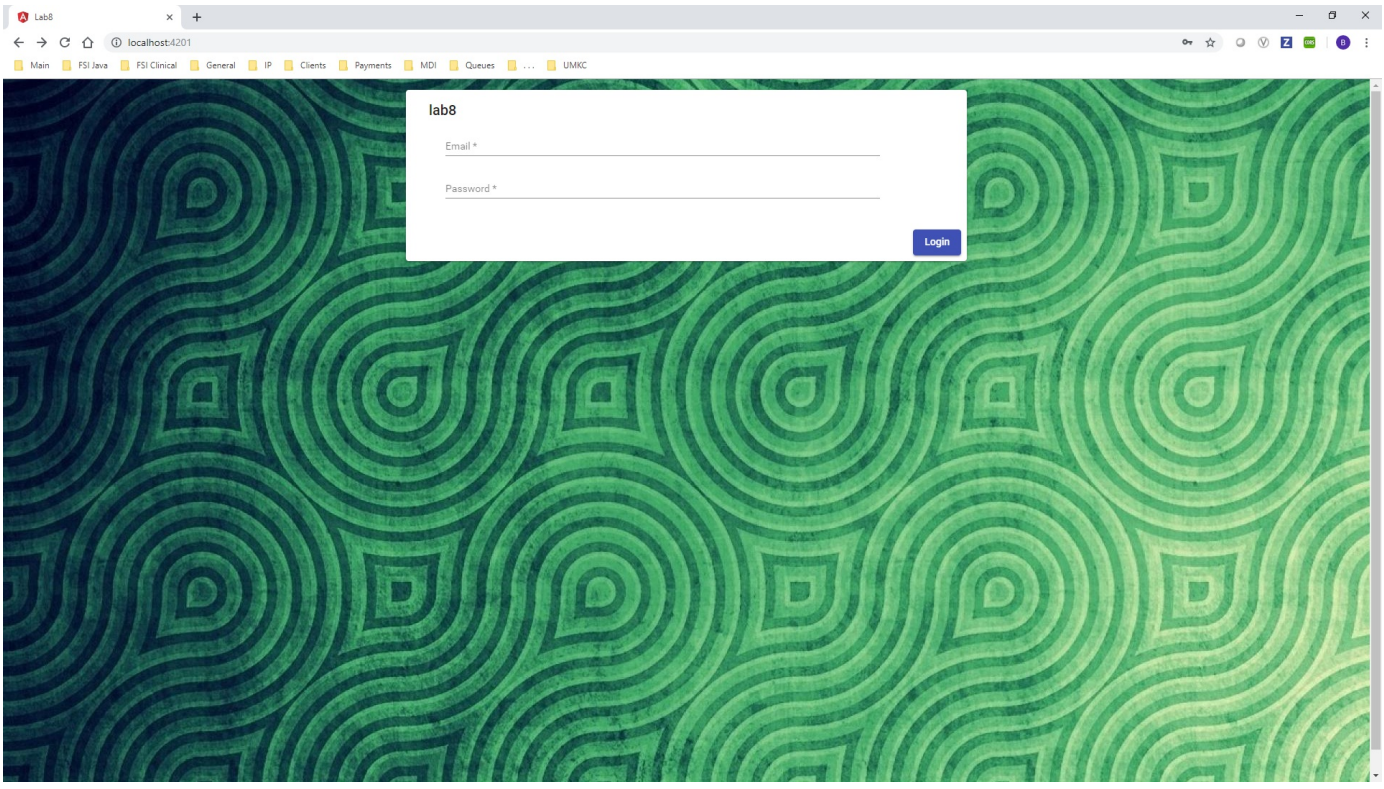
Login

This page features two input boxes for email address and password. Upon login, the JWT token is stored in local storage using a post request to the URL http://localhost:3000/api/login. Once submitted, the app navigates to the home page:



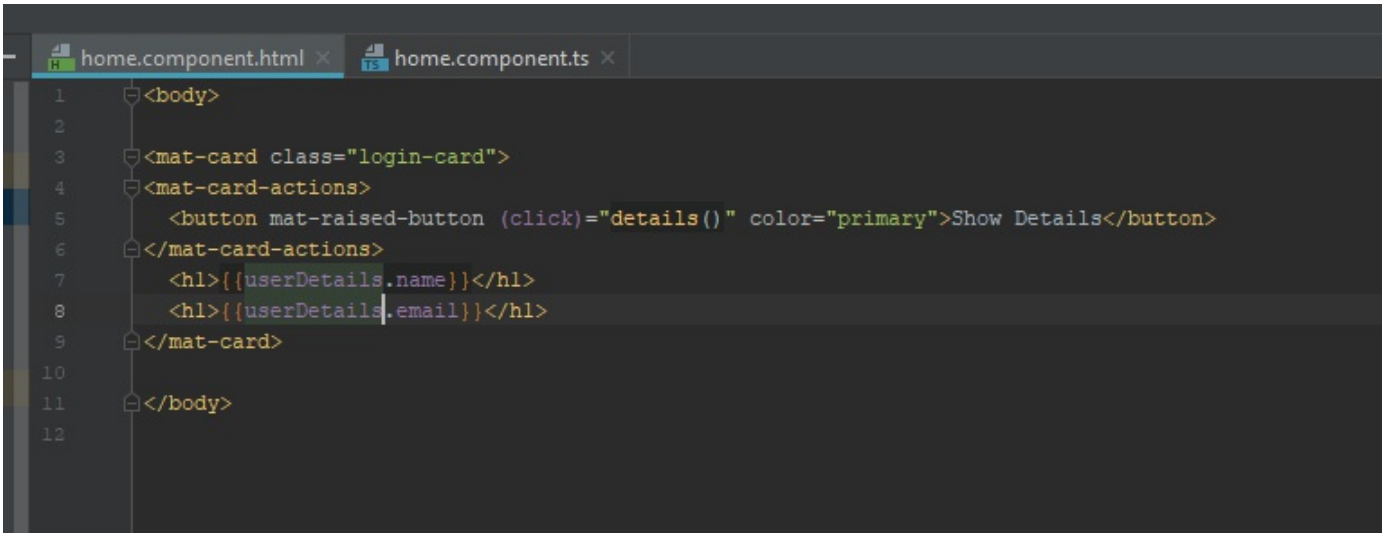
home.component.html | home.component.ts | login.component.ts

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

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

  constructor(private http: HttpClient, private router: Router) { }

  email: string;
  password: string;

  ngOnInit() {
  }

  login() {
    if (this.email !== '' && this.password !== '') {
      this.http.post( url: 'http://localhost:3000/api/login',  body: {name: this.email, email: this.password})
        .subscribe( next: (data: any) => {
          localStorage.setItem('auth_token', data.token);
          this.router.navigate( commands: ['/home']);
          console.log(data.token);
        });
    } else {
      alert('Invalid credentials');
    }
  }
}
```



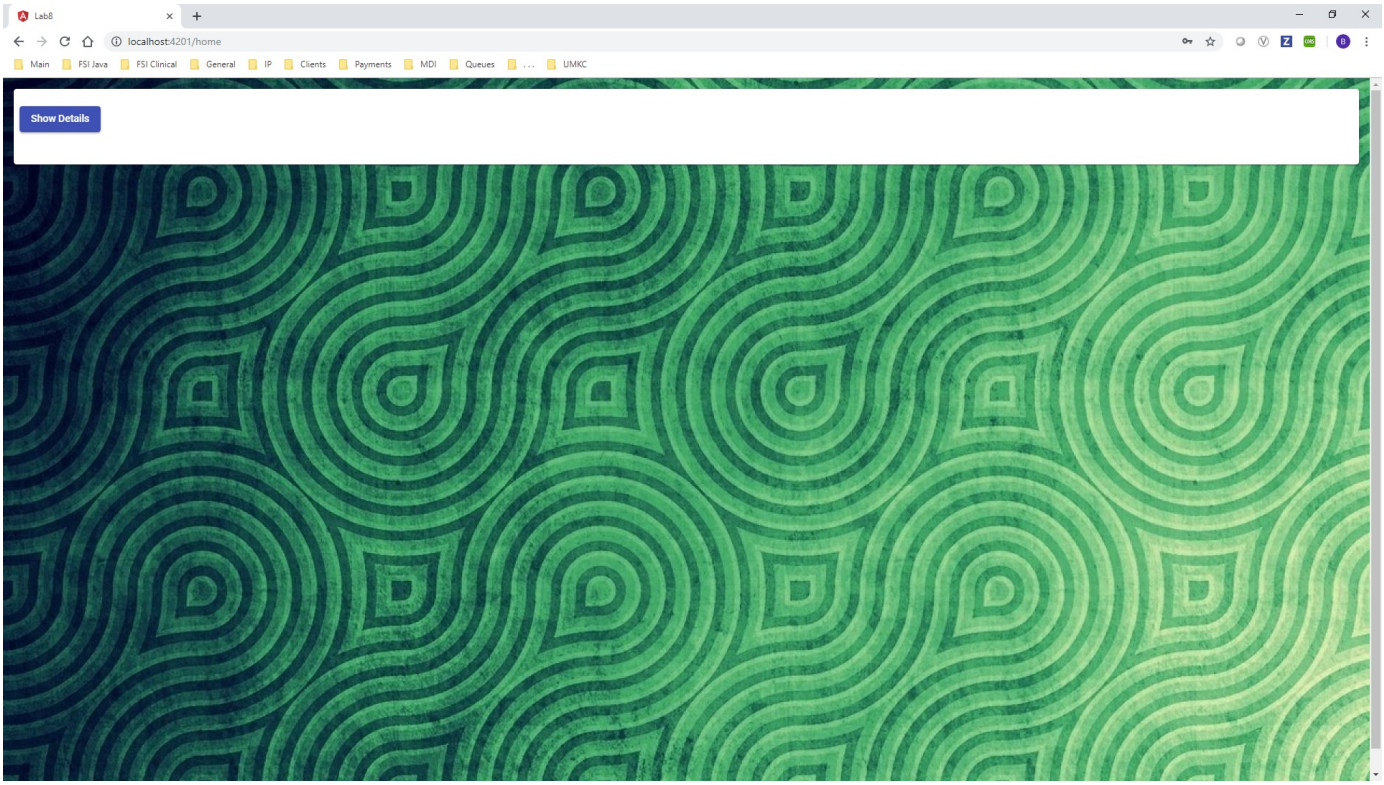
Home

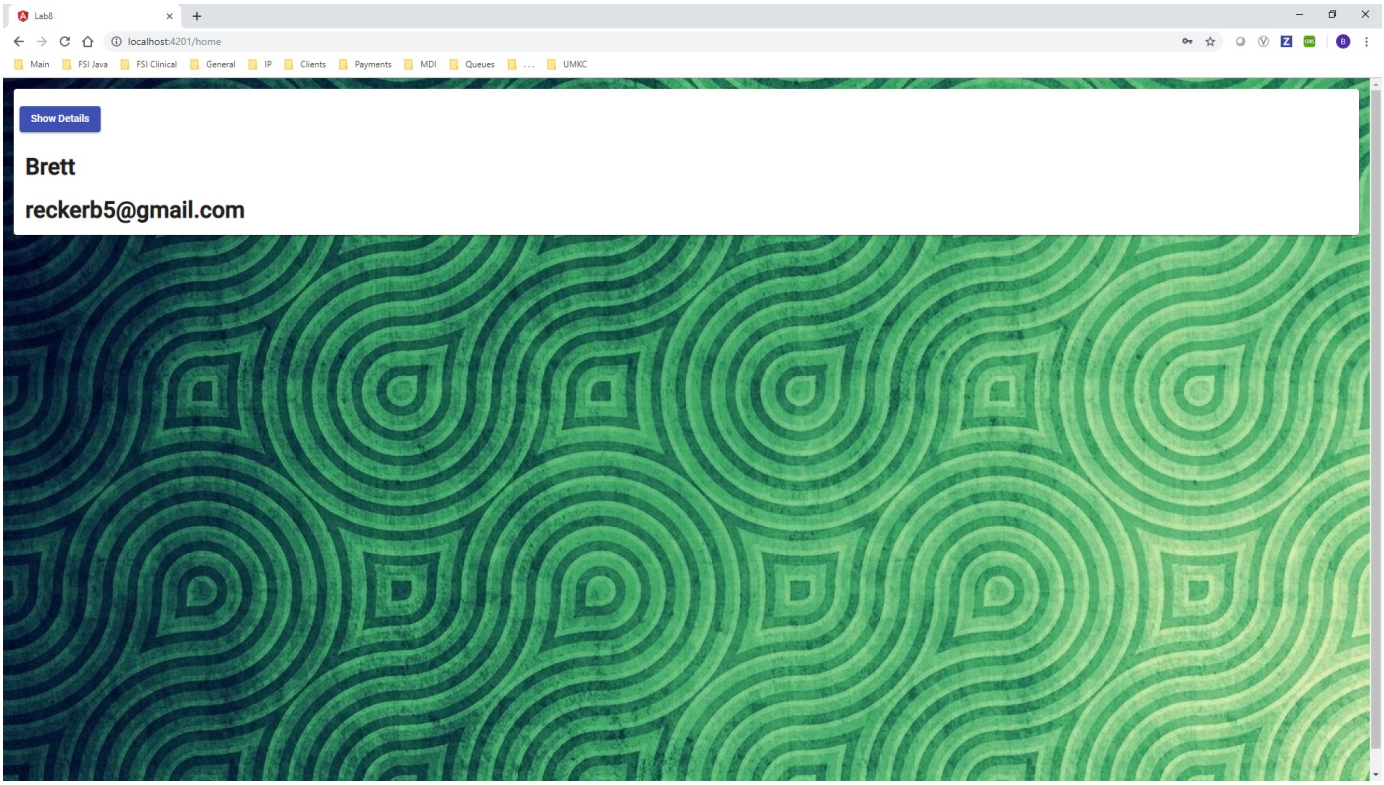
This page features a Show Details button. When clicked, the token is pulled from local storage a get request is sent to URL http://localhost:3000/api/posts. Then, the returned details (name, email) are populated on the page:



home.component.ts

```ts
import { Component, OnInit } from '@angular/core';
import {HttpClient, HttpHeaders} from '@angular/common/http';

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

  userDetails: any;

  constructor(private http: HttpClient) { }

  ngOnInit() {
  }

  details() {
    const token = localStorage.getItem( key: 'auth_token');
    console.log(token);
    const headers = new HttpHeaders().set ('auth_token', token);
    this.http.get( url: 'http://localhost:3000/api/posts',  options: {headers})
      .subscribe( next: response => {
        this.userDetails = response.user;
        console.log(response);
      });
  }
}

```





Outcome

This was completely new to me and I will need to continue working with tokens to fully understand this workflow before implementing in our

project.