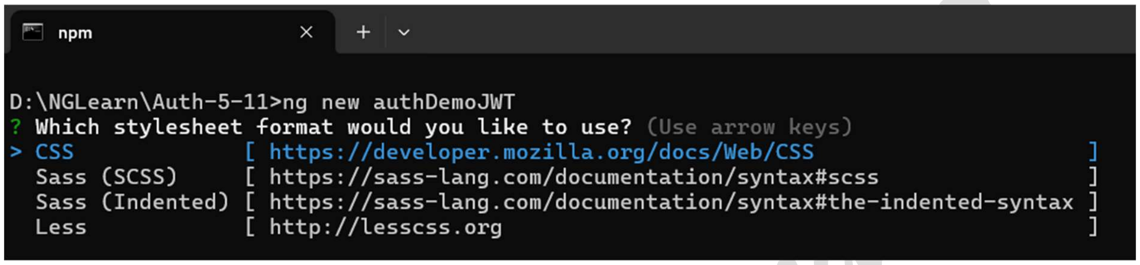
## Angular Login with JWT

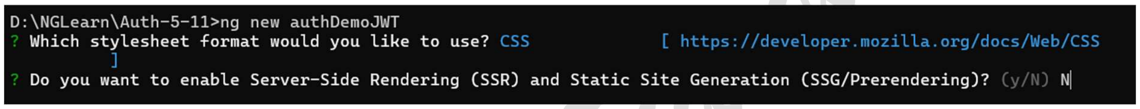
This section is in continuation of the previous section named "Working With Authentication". Here we will be using JWT

Create a new Angular basic application as usual.

**> ng new authDemoJWT**



```
D:\NGLearn\Auth-5-11>ng new authDemoJWT
? Which stylesheet format would you like to use? (Use arrow keys)
> CSS              [ https://developer.mozilla.org/docs/Web/CSS                  ]
  Sass (SCSS)      [ https://sass-lang.com/documentation/syntax#scss             ]
  Sass (Indented)  [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]
  Less             [ http://lesscss.org                                          ]
```



```
D:\NGLearn\Auth-5-11>ng new authDemoJWT
? Which stylesheet format would you like to use? CSS              [ https://developer.mozilla.org/docs/Web/CSS
        ]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? (y/N) N
```

Open application in Visual Studio Code and open terminal into it. Now install bootstrap as usual:

**>...\authDemoJWT> npm i bootstrap**

Go to angular.json and make changes to make bootstrap available to entire application.

```
"styles": ["node_modules/bootstrap/dist/css/bootstrap.css",
              "src/styles.css"
            ],
            "scripts":
["node_modules/bootstrap/dist/js/bootstrap.js"]
          },
```

Go to the app folder add a new folder named pages and create a login component in pages.

**>...authDemoJWT\src\app\pages> ng g c login**

Also create two more components:

**>...authDemoJWT\src\app\pages>ng g c layout**

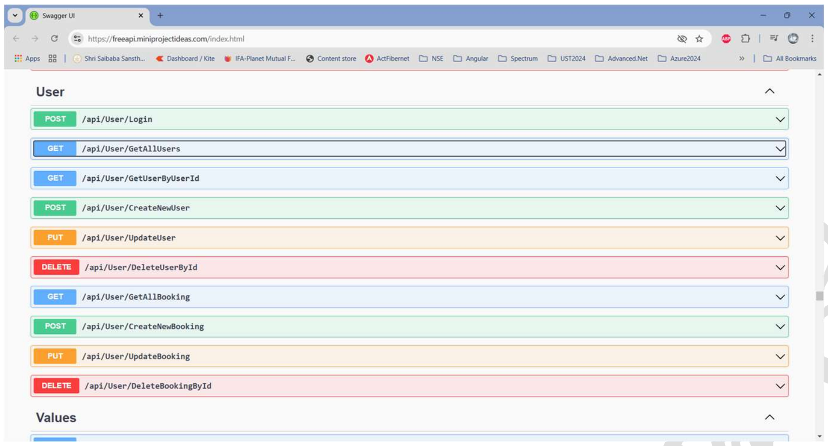
**>...authDemoJWT\src\app\pages>ng g c dashboard**

Now define routes

Design Login Page as we did in Auth example.

Also, navbar to the layout.

Here test and run application on port 4209 as I am using APIs from some free APIs website https://freeapi.miniprojectideas.com/index.html.



User

POST /api/User/Login
GET /api/User/GetAllUsers
GET /api/User/GetUserByUserId
POST /api/User/CreateNewUser
PUT /api/User/UpdateUser
DELETE /api/User/DeleteUserById
GET /api/User/GetAllBooking
POST /api/User/CreateNewBooking
PUT /api/User/UpdateBooking
DELETE /api/User/DeleteBookingById

Values

**>...authDemoJWT> ng serve -o --port 4209**

Now go to login component.ts and create http instance.

Also to use httpclient modify app.config.ts

```
import { provideHttpClient } from '@angular/common/http';

export const appConfig: ApplicationConfig = {
  providers: [provideHttpClient(), provideZoneChangeDetection({
eventCoalescing: true }), provideRouter(routes)]
};
```

Modify login.component.ts

```
http = inject(HttpClient);
router = inject(Router);

  onLogin() {
    this.http.post("https://freeapi.miniprojectideas.com/api/User/Lo
gin", this.userObj).subscribe((res: any) => {
      if (res.result) {
        alert("Login Success");
         this.router.navigateByUrl('dashboard');
      } else {
        alert(res.message);
      }
    })
  }
```

Now go to dashboard component and get all users api data which is a protected route.

```
export class DashboardComponent implements OnInit {
  http = inject(HttpClient);
  userList: any[] = [];
  ngOnInit(): void {
    this.getAllUsers()
  }
  getAllUsers() {
    this.http.get("https://freeapi.miniprojectideas.com/api/User/GetAllUsers").subscribe((res: any) => {
      this.userList = res.data;
    })
  }
}
```

Also, in HTML I am trying to get all users.

```
<p>dashboard works!</p>
<div>
    <p>{{userList | json}}</p>
</div>
```

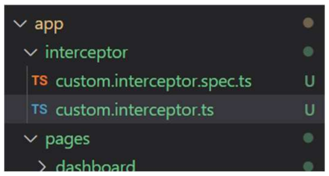
If we inspect in browser I am getting 401 Unauthorized status, because token is not passed in headers. I need to get token from login and pass it in header. For such we need to use interceptors in angular.

**What are Interceptors?**

Interceptors are middleware that allows common patterns around retrying, caching, logging, and authentication to be abstracted away from individual requests.

Interceptors are generally functions which you can run for each request, and have broad capabilities to affect the contents and overall flow of requests and responses. You can install multiple interceptors, which form an interceptor chain where each interceptor processes the request or response before forwarding it to the next interceptor in the chain.

Now go to app folder and create a new folder named interceptor and add an interceptor named custom:



**>...authDemoJWT\src\app\interceptor> ng g interceptor custom**

```
import { HttpInterceptorFn } from '@angular/common/http';

export const customInterceptor: HttpInterceptorFn = (req, next) => {
  return next(req);
};
```

It is an arrow function, previous versions it was available as service but now it is available as an arrow function.

First go to app.config.cs and define that we are going to use interceptors with provideHttpClient.

```
import { customInterceptor } from
'./interceptor/custom.interceptor';

export const appConfig: ApplicationConfig = {
  providers:
[provideHttpClient(withInterceptors([customInterceptor])),
provideZoneChangeDetection({ eventCoalescing: true }),
provideRouter(routes)]
};
```

Now we need to send token received from login to the request of get method and that we will do in this interceptor.

```
import { HttpInterceptorFn } from '@angular/common/http';

export const customInterceptor: HttpInterceptorFn = (req, next) => {
  const token = localStorage.getItem('myLogInToken');

  const clonereq = req.clone({
    setHeaders: {
      Authorization: `Bearer ${token}`
    }
  })

  return next(clonereq);
};
```

Check the dashboard userlist json is available.

☸✠End of Chapter✠☸