

Order History Frontend

Pass Access Token



View Order History

- The REST API for `/api/orders` is secured
- Need to update Angular App to pass the access token

The screenshot shows the luv2shop website interface. At the top, there is a blue header bar with the luv2shop logo, a search bar, a 'Search' button, a 'Welcome back Demo Darby!' message, a 'Logout' button, a 'Member' button, and an 'Orders' button which is highlighted with a red dashed border. To the right of the header is a shopping cart icon showing '\$0.00' and '0'. Below the header, on the left, is a sidebar with links for Books, Coffee Mugs, Mouse Pads, and Luggage Tags. The main content area is titled 'Your Orders' and displays a table with two rows of order information. The table has columns for Order Tracking Number, Total Price, Total Quantity, and Date.

| Order Tracking Number | Total Price | Total Quantity | Date |
|--------------------------------------|-------------|----------------|---------------------------|
| 51542a04-e16d-4e3e-9549-7e3394558f8d | \$36.98 | 2 | Feb 13, 2021, 10:39:57 AM |
| 0e8014f2-0cd9-4ce4-bb5a-e451154ee9f0 | \$17.99 | 1 | Feb 13, 2021, 11:59:58 AM |

Client sending access token

- Client sends the access token as an HTTP request header

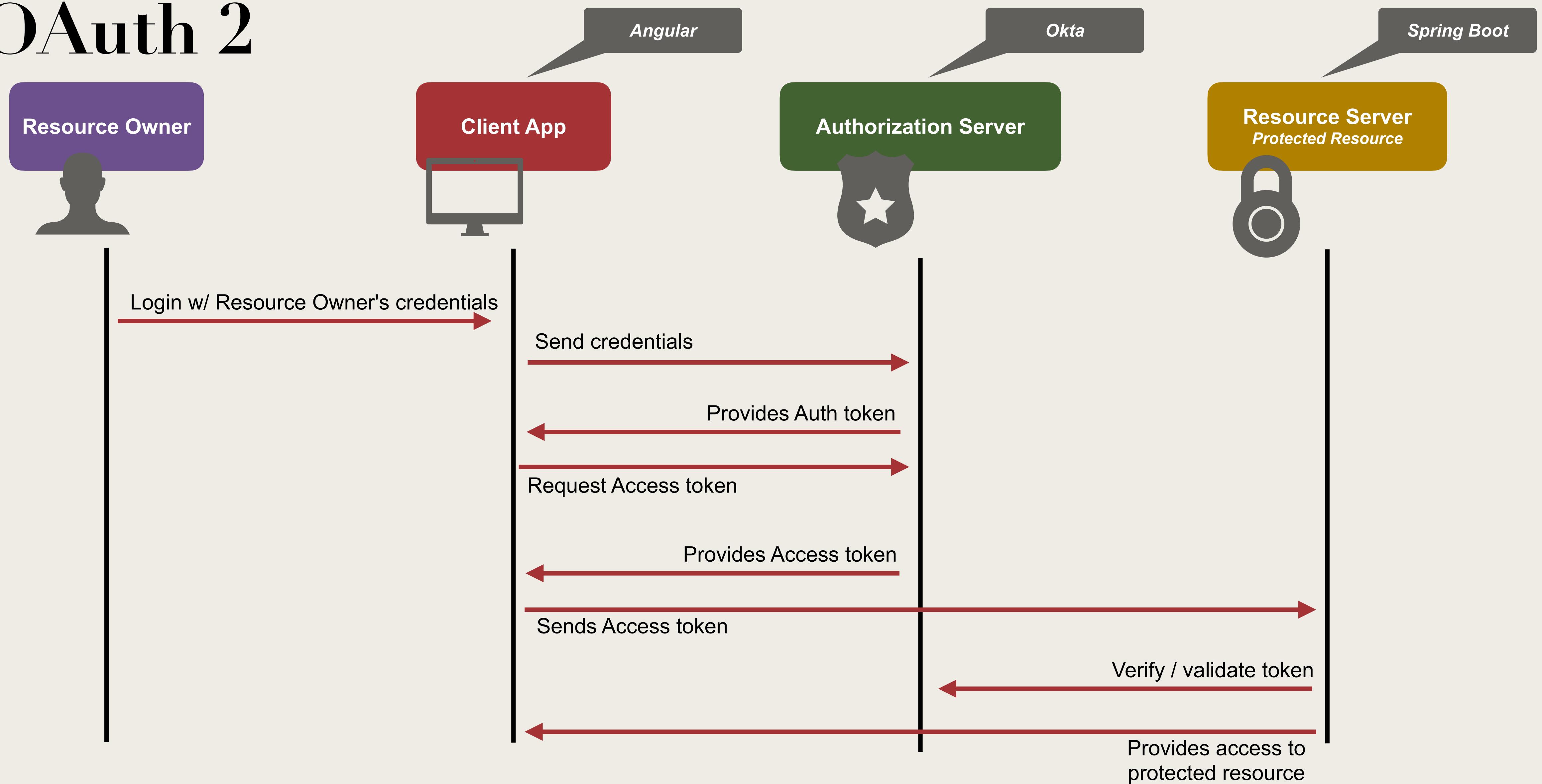
Request header

Authorization: Bearer <token>

Request body

....
....
....

OAuth 2



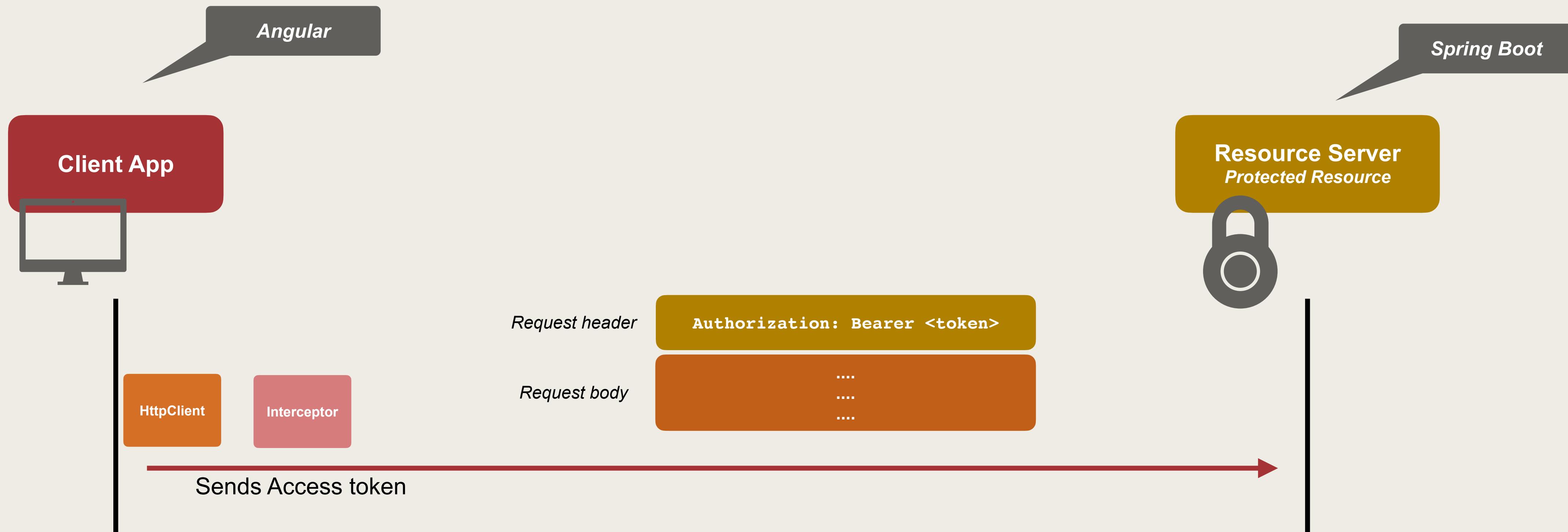
Angular Interceptors

- Angular provides support for interceptors
- An interceptor can intercept HTTP requests / responses
- Useful to perform processing on the HTTP requests / responses



Angular Interceptors

- We will use an interceptor to pass to access token in HTTP request



Development Process

Step-By-Step

1. Create an interceptor
2. Update app.module.ts to reference interceptor

Step 1: Create an interceptor

- Develop the interceptor as a service

```
$ ng generate service services/AuthInterceptor
```

Step 1: Create an interceptor

File: auth-interceptor.service.ts

```
import { HttpEvent, HttpHandler, HttpInterceptor, HttpRequest } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { OktaAuthService } from '@okta(okta-angular)';
import { from, Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class AuthInterceptorService implements HttpInterceptor {

  constructor(private oktaAuth: OktaAuthService) { }

  intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
    return from(this.handleAccess(request, next));
  }
}
```

The diagram shows a flow from a Client App (Angular) to a Resource Server (Spring Boot). The Client App contains an HttpClient and an Interceptor. The Interceptor adds an Authorization header with a token ('Authorization: Bearer <token>') to the request. The request then goes to the Resource Server, which is a protected resource.

```
private async handleAccess(request: HttpRequest<any>, next: HttpHandler): Promise<HttpEvent<any>> {
  // Only add an access token for secured endpoints
  const securedEndpoints = ['http://localhost:8080/api/orders'];

  if (securedEndpoints.some(url => request.urlWithParams.includes(url))) {
    // get access token
    const accessToken = await this.oktaAuth.getAccessToken();

    // clone the request and add new header with access token
    request = request.clone({ headers: request.headers.set('Authorization', `Bearer ${accessToken}`) });
  }

  return next.handle(request).toPromise();
}
```

Annotations on the code:

- Asynchronous function**: Points to the `private async handleAccess` declaration.
- Returns a Promise**: Points to the `Promise<HttpEvent<any>>` return type.
- Waits for the async call to finish**: Points to the `await this.oktaAuth.getAccessToken()` line.
- Async call**: Points to the `next.handle(request).toPromise()` line.

Step 2: Update app.module.ts to reference interceptor

File: app.module.ts

```
...
@NgModule({
  declarations: [ ... ],
  imports: [ ... ],
  providers: [ProductService, { provide: OKTA_CONFIG, useValue: oktaConfig },
    {provide: HTTP_INTERCEPTORS, useClass: AuthInterceptorService, multi: true}],
})
...
```

*Token for
HTTP interceptors*

*Register our
AuthInterceptorService
as an HTTP interceptor*

*Informs Angular that
HTTP_INTERCEPTORS is a token
for injecting an array of values*

Additional Resources

- Angular Interceptors
- Promises
- async / await

www.luv2code.com/angular-additional-resources