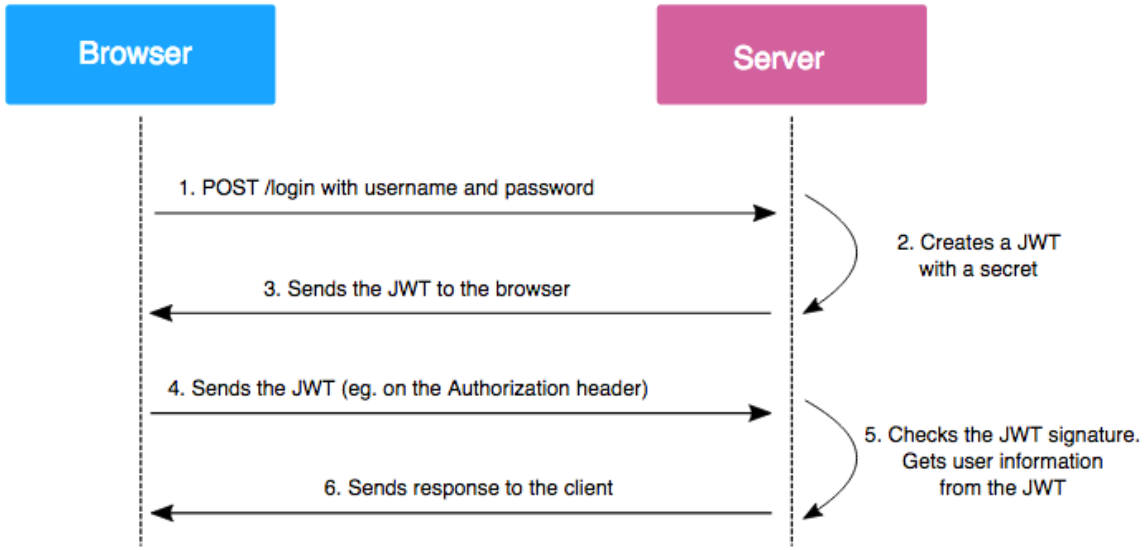
netmind
a bts company

Angular 14 - 19

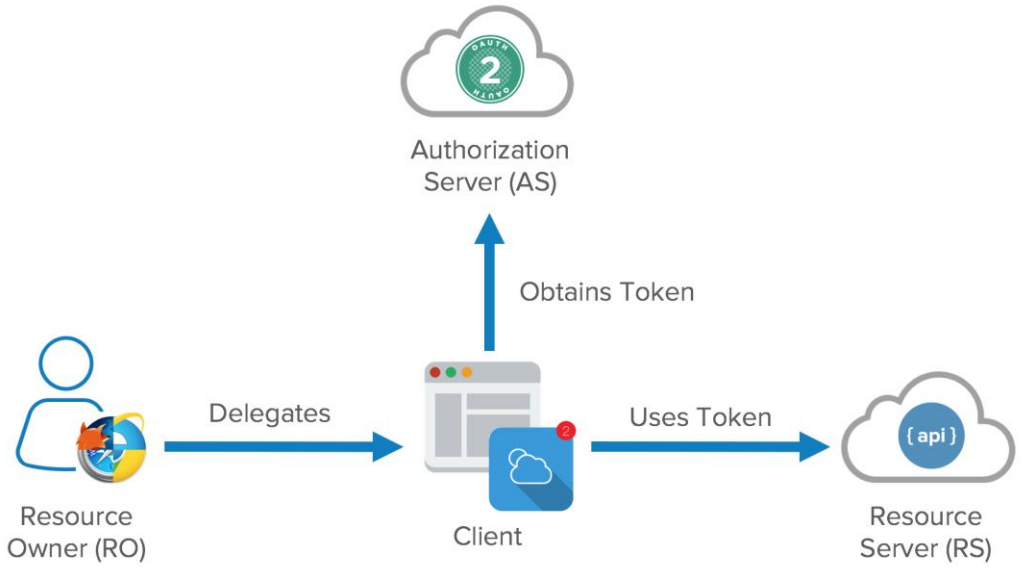
Server Authentication

JWT (JSON Web Token)



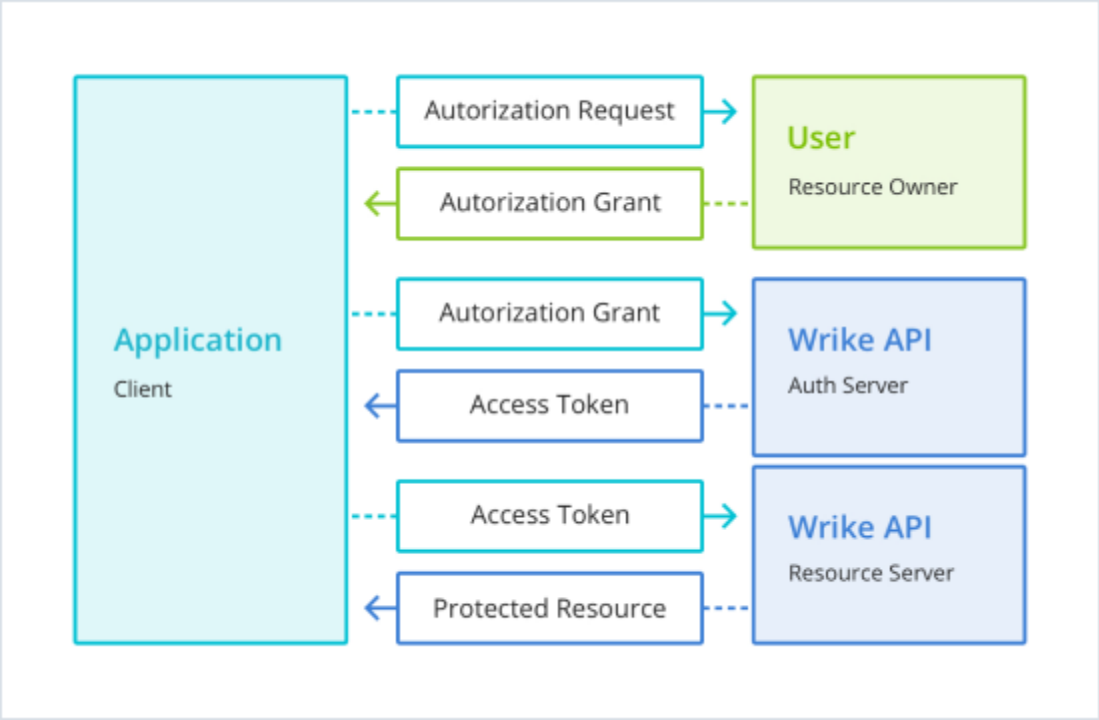
- A JWT technically is a **mechanism to verify the owner of some JSON data**.
- It's an **encoded string**, which is URL safe, that can contain an unlimited amount of data (unlike a cookie), and it's cryptographically signed.
- When a server receives a JWT, it can **guarantee the data** it contains can be **trusted** because it's signed by the source.
  - No middleman can modify a JWT once it's sent.
- JWT **guarantees data ownership** but not encryption; the JSON data into a JWT can be seen by anyone that intercepts the token, as it's just serialized, not encrypted.
  - For this reason, it's highly recommended to use HTTPS with JWTs (and HTTPS in general, by the way).

OAuth



- OAuth is not an API or a service: it's an **open standard for authorization** and anyone can implement it.
- More specifically, OAuth is a standard that apps can use to provide client applications with "**secure delegated access**".
- OAuth works over HTTPS and authorizes devices, APIs, servers, and applications with access tokens rather than credentials.
- Simply put: it's a standard to securely access stuff with randomized tokens.

OAuth2



- OAuth 2.0 is the **industry-standard protocol for authorization**.
  - https://oauth.net/2/
- It focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices.
  - This specification and its extensions are being developed within the IETF OAuth Working Group.
- OAuth2 uses HTTPS for communication between the client and the authorization server.
- There are 2 types of token:
  - **Access Token**: This allows a third-party application to access user data on a resource server.
  - **Refresh Token**: this token is issued with the access token but unlike the latter, it is NOT sent in each request from the client to the resource server. The client application can simply use a refresh token to renew access token when it expires.

JSON Web Tokens vs Oauth 2.0

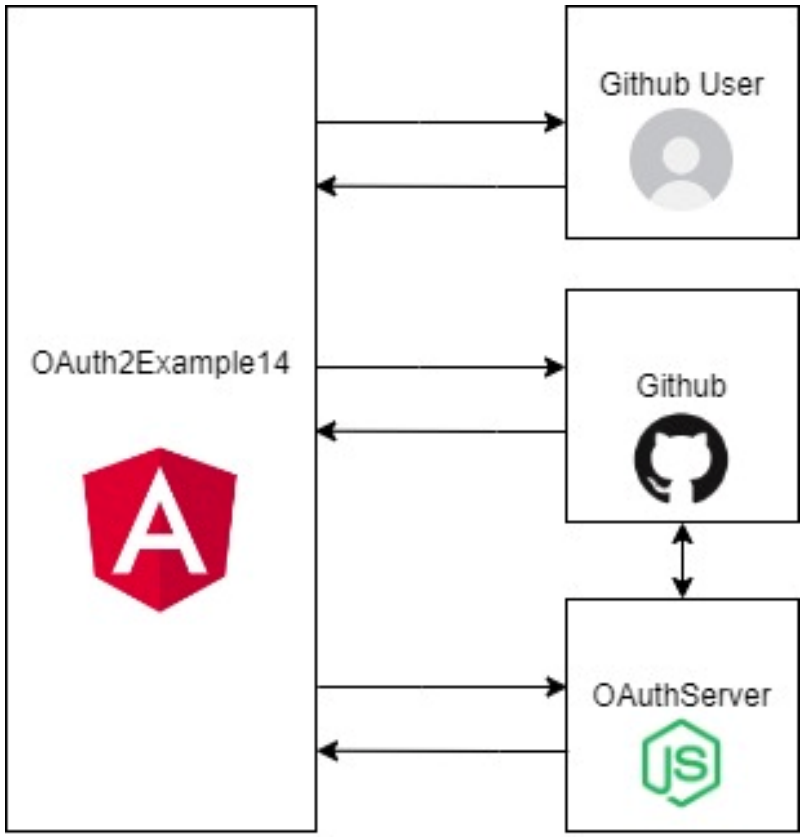
- So the real difference is that JWT is just a token format, OAuth 2.0 is a protocol (that may use a JWT as a token format or access token which is a bearer token.).

Oauth 2.0 actors

OAuth2 has 4 players in operation:

1. **Resource owner**: A resource owner has the capacity to give access to a protected resource. If the resource owner is a person then you call it as end user.
2. **Resource server**: A server storing the protected resource.
3. **Client**:The client is the application that makes a request on behalf of the resource owner to access the protected resource.
4. **Authorisation server**: The authorization server may be the same server as the resource server or a separate entity. It authenticates the resource owner and after obtaining authorization, issues access tokens to the client.

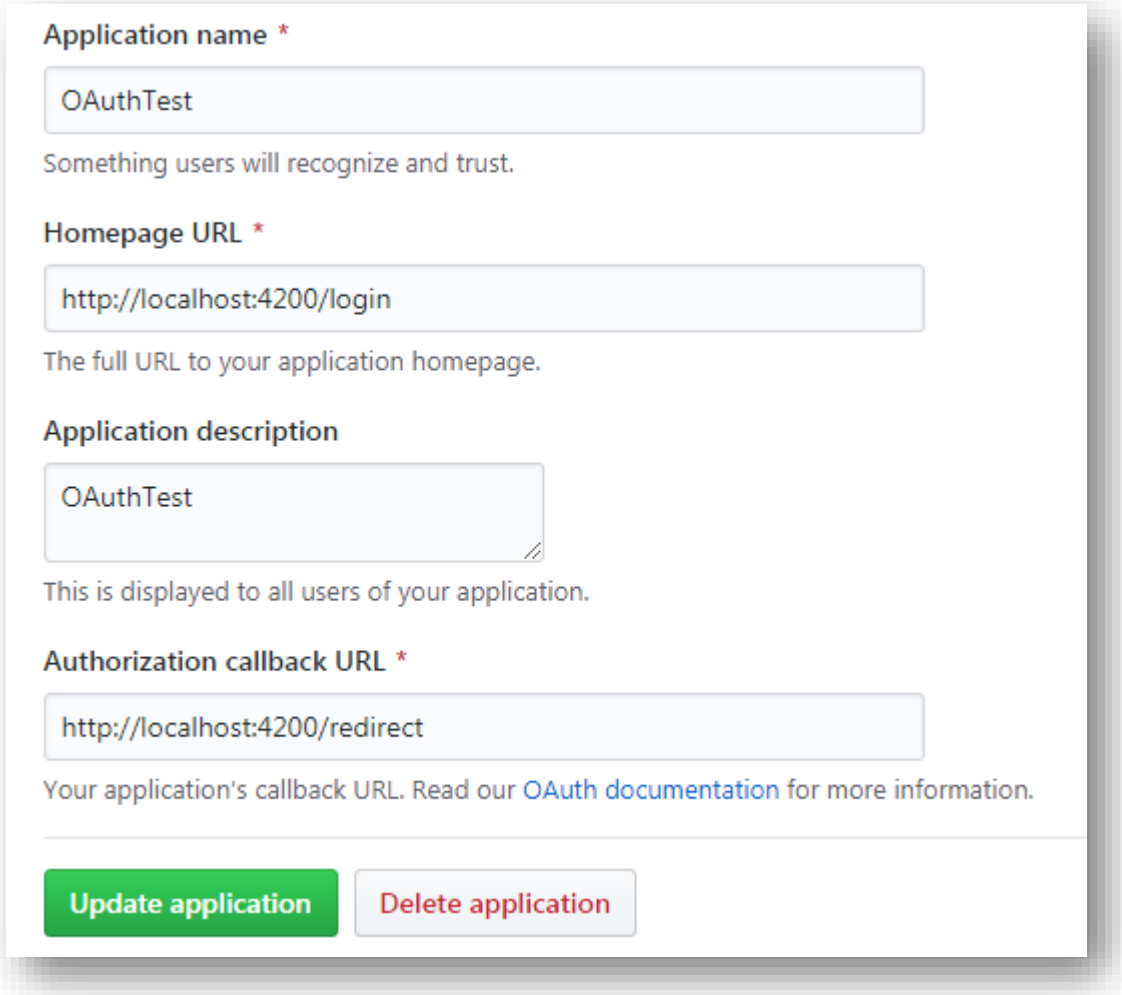
Oath2 with Angular client



- The following example illustrates how Angular app can be used with OAuth2 protocol. For this we are going to use Github as Auth server.
- The architecture is made of:
  - **Resource owner**: the owner of github account (yourself).
  - **Resource server**: A node server project (OAuthServer).
  - **Client**:the Angular application (OAuth2Example14).
  - **Authorisation server**: Github.

Configuring Github

- We need to register the OAuthServer application with the Authorisation server (Github)
  - For that follow this guide: https://docs.github.com/en/developers/apps/building-oauth-apps/creating-an-oauth-app
- Once the application is registered, Git will issue the client credentials in the form of a client identifier and a client secret:
  - The **Client ID** is a publicly exposed string that is used by GitHub to identify the application
  - The **Client Secret** is used to authenticate the identity of the application to Git and must be kept private.
  - These values must be used for configuring the **.env** file in OAuthServer.
- GitHub's authorization endpoints and flow are defined here:
  - https://docs.github.com/en/developers/apps/building-oauth-apps/authorizing-oauth-apps



The node server

It is an express server that is able to receive requests in 4 routes:

- **/AuthPage** returns GitHub's authorisation endpoint to the angular application.
- **/getAccessToken** is a request made to the token endpoint of the authorisation server i.e GitHub to provide an access token upon successful validation of the authorisation code and state.We are storing the token in a session cookie.
- **/getUserDetails** is a request to the resource server i.e GitHub to provide the protected resource.
- **/logout** is a request to log the user out of the OAuth application. The user is not logged out of GitHub.
- Values in **.env** file must be update with github client and secret.

🔗 **start the server with**

```
npm start
```

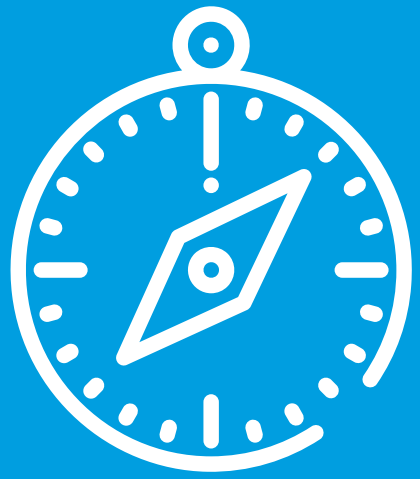
Angular app

We will implement our application in a very similar way as with JWT authentication:

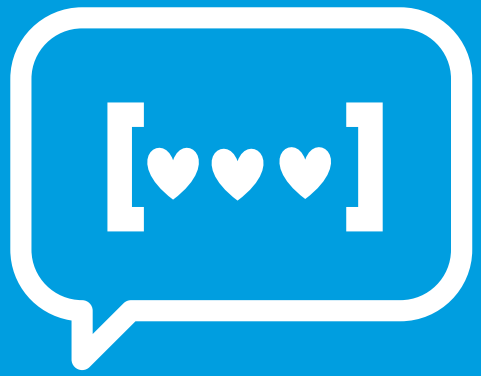
- **Module:** a main module as usual, with the import of routing and HTTP_INTERCEPTORS.
- **Routing:** with the routes 'dashboard', 'login', 'redirect', 'auth', '**'.
  - 'auth' will be associated to a resolver.
  - 'dashboard' will be restricted and needs authorization.
- **Components:** one for each route. DashboardComponent, LoginComponent, RedirectComponent, GitAuthComponent, NoSuchComponent
- **Services:**
  - OAuthService to communicate with the server.
  - ExtUrlResolverService to enable navigation to GitHub's authorisation endpoint.
- **Interceptors:** ErrorHandlingInterceptor
- **Proxy:** proxy.conf.json for allowing shortcutting the server endpoints associated to '/oauth'
- **Environment variables:** define the baseUrl.

🔗 **start the dev server with:**

```
npm start
```



Next steps



We would like to know your opinion!

Please, let us know what you think about the content.
From Netmind we want to say thank you, we appreciate time and effort you have taking in answering all of that is important in order to improve our training plans so that you will always be satisfied with having chosen us
quality@netmind.es

netmind
a bts company

Thanks!

Follow us: