Introducing the IDENTITY GUARDIANS

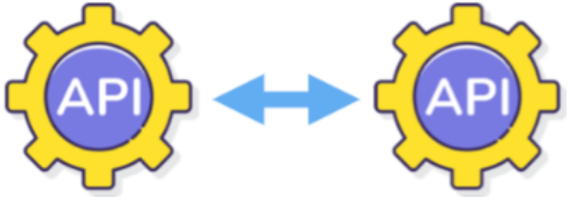
Alisa Duncan

Senior Developer Advocate @ Okta

Use cases

Web applications

Device

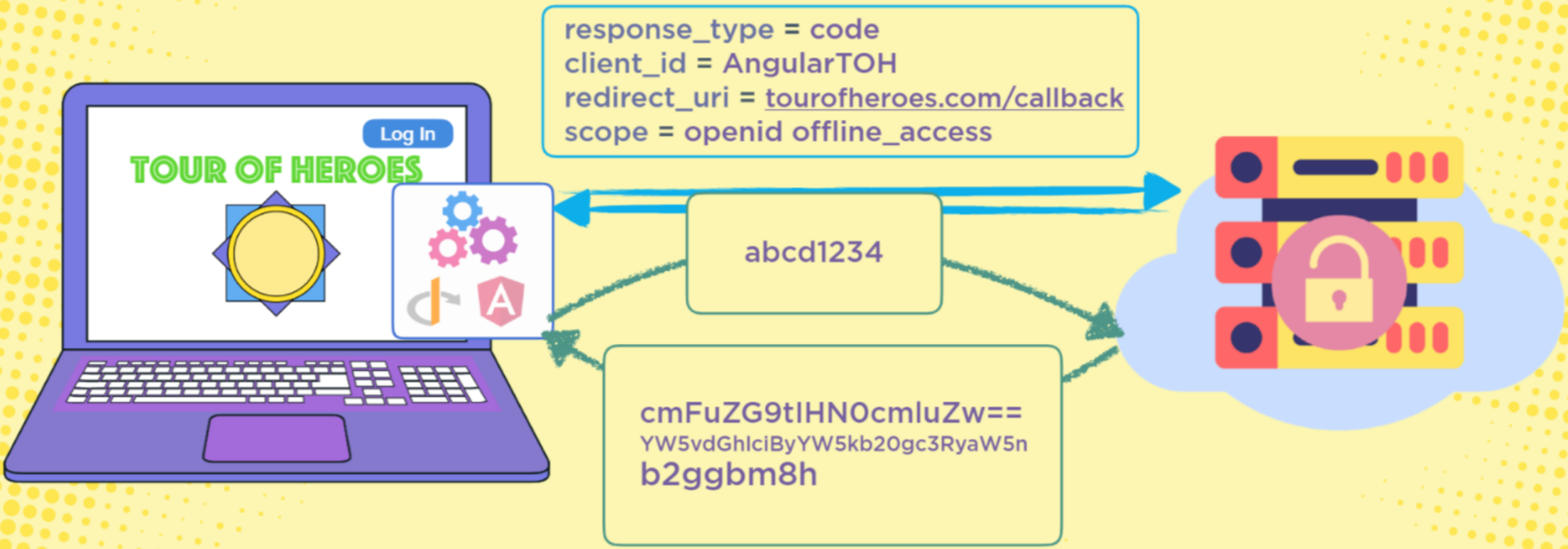


Service interaction

OAuth 2.0

OpenID Connect (OIDC)

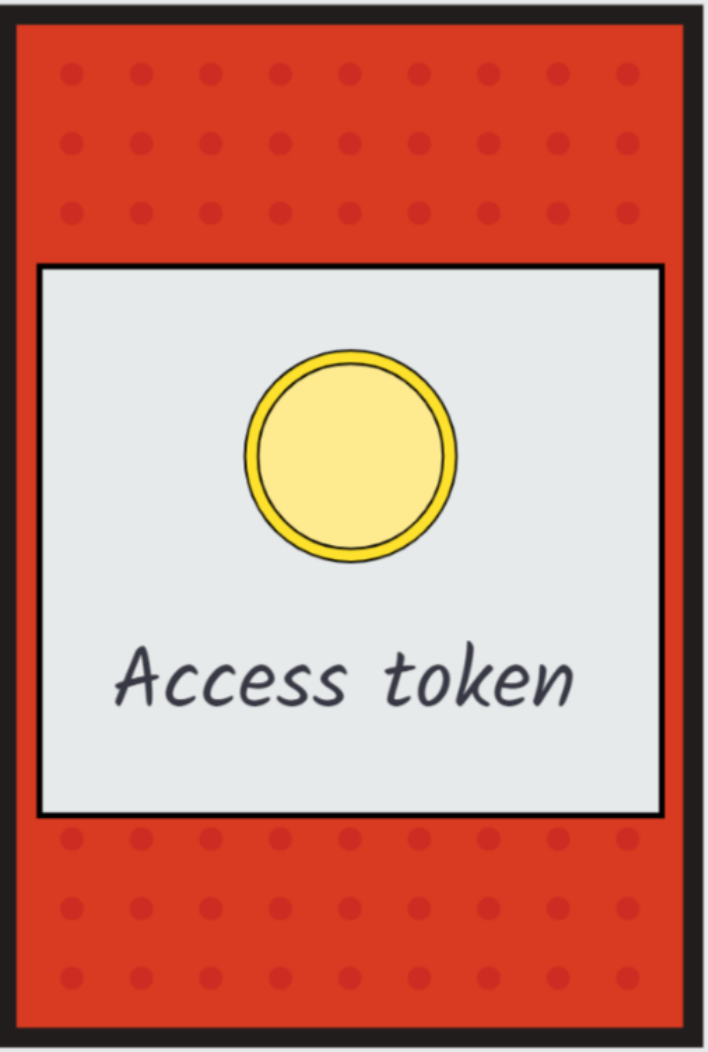
OAuth 2.0 + OpenID Connect



OAuth flows

Web applications

Authorization Code
Proof Key for Code Exchange (PKCE)


Access token


ID token


Refresh token

Access token

★ For authorization, from OAuth

★ Access to data and the actions you want to perform

ID token

Refresh token

Access token

ID token

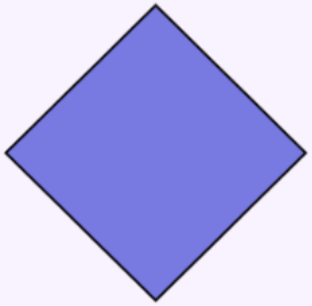
★ For authentication, from OpenID Connect

★ Standardized identity information

Refresh token

Access token

ID token



Refresh token

★ Extends the life of authorization and identity

★ Exchange for new tokens


@AlisaDuncan #identityguardians
How do the guardians protect us and our apps?

ID token

Issuer TOUR OF HEROES LLC

Nonce

One time use ID 123456

Issued at

Issued Today

Expires

Expires Soon

ID valid only for Angular TOH

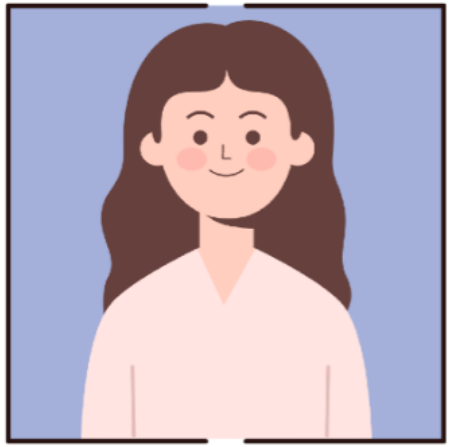
Client ID

Authenticated

ID token

```
@Component({
  selector: 'app-root',
  template: `
    @if (authService.isAuthenticated$ | async) {
      <a>Sign out</a>
    } @else {
      <a>Sign in</a>
    }
  `,
})
export class AppComponent {
  authService = inject(AuthService);
}
```

ID token



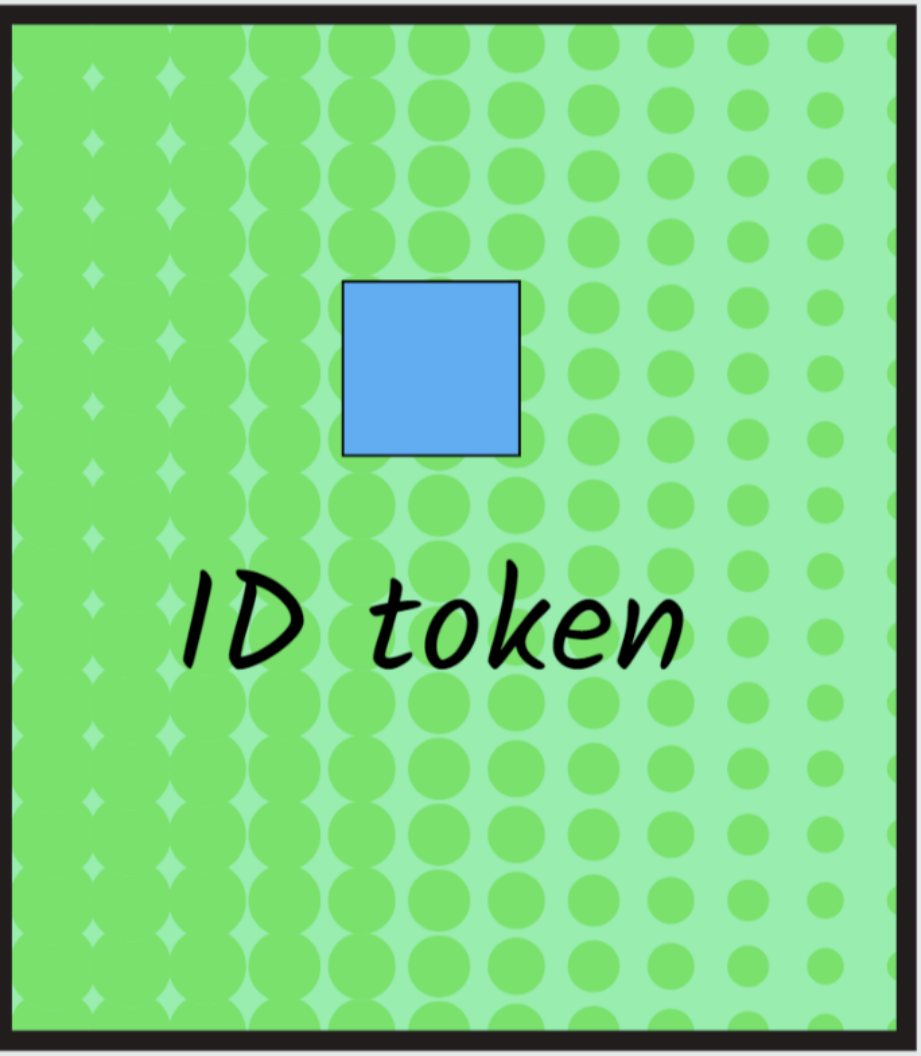
Ima Programmer

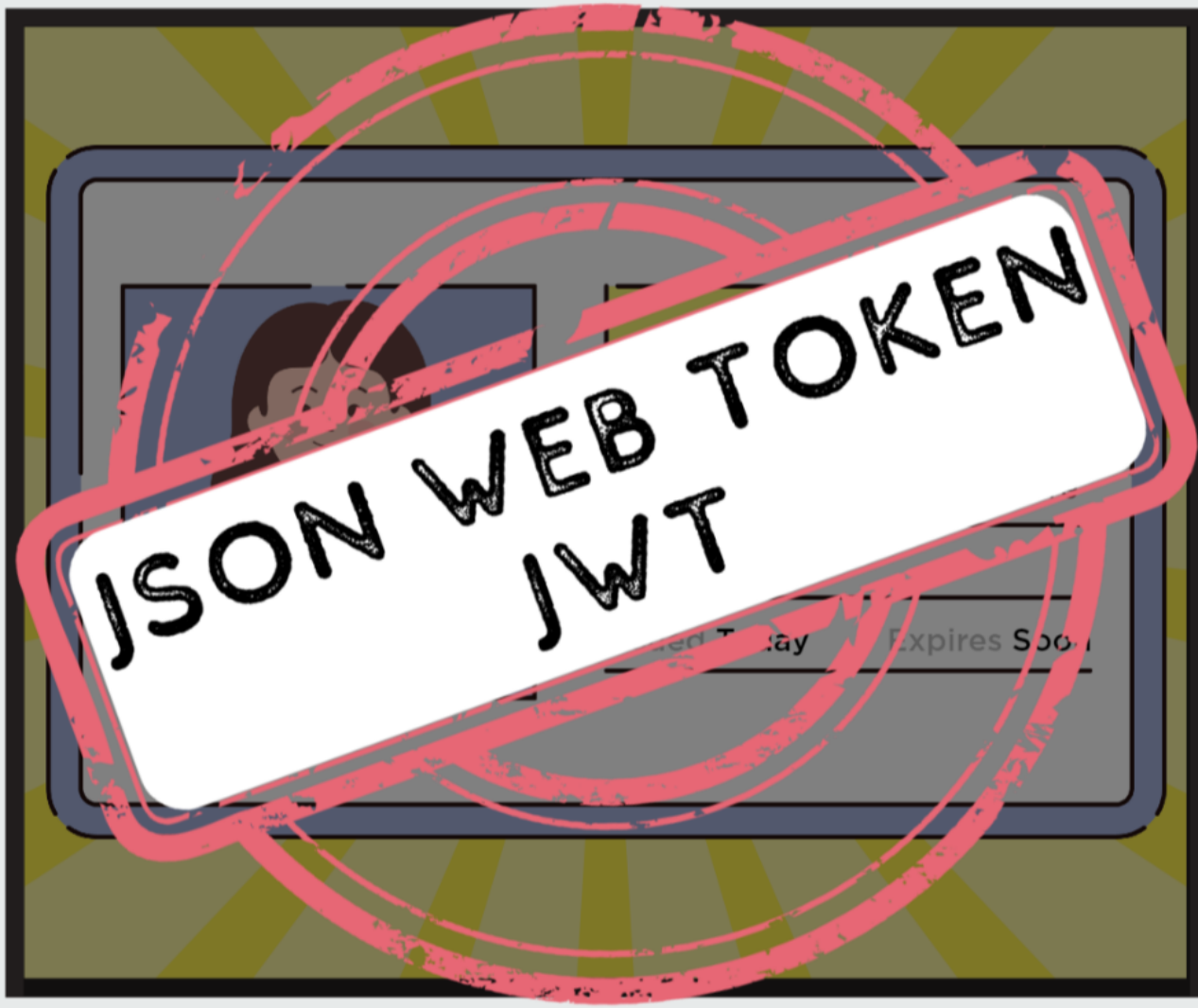
ima.programmer@email.fake

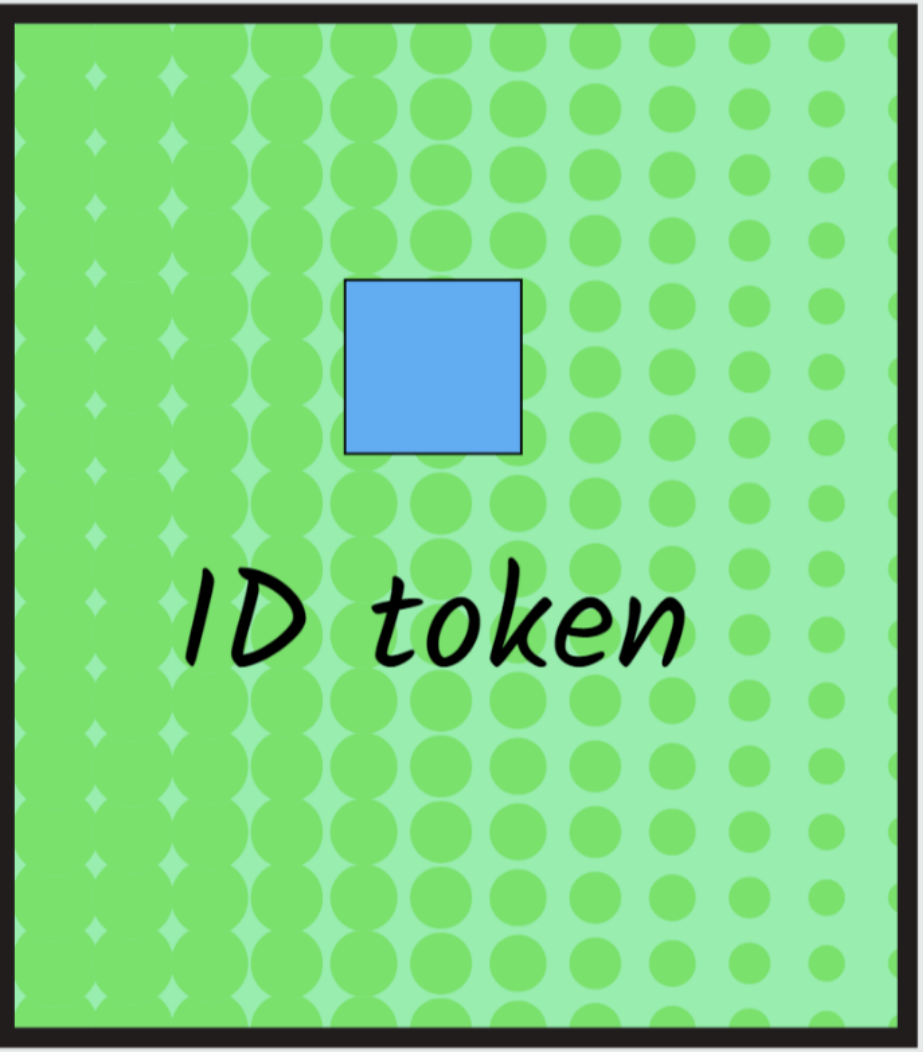
Title Developer Extraordinaire

Role Member

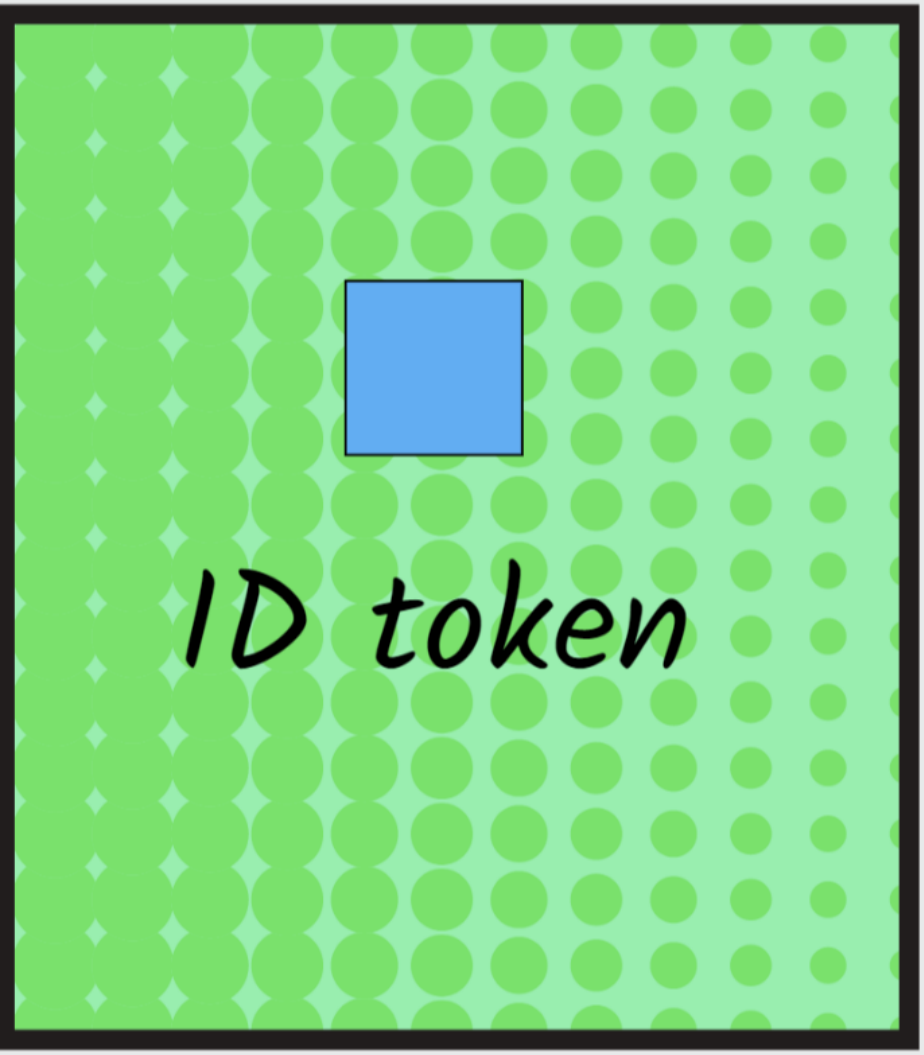
Issued Today Expires Soon


ID token


JSON WEB TOKEN
JWT



eyJraWQiOiJhb1B1U2FKSUlOLXV5UEYzNEpMUHozV3puRzBHOX
N1bjZTYUN3UjBFSHowIiwiYWxnIjoiUlMyNTYifQ.eyJzdWIiO
iIwMHUxanFvZDU2ZnZqMkcxTzFkOCIsIm5hbWUiOiJBbGlzYSB
EdW5jYW4iLCJsb2NhbGUiOiJVUyIsImVtYWlsIjoiYWxpc2EuZ
HVuY2FuQG9rdGEuY29tIiwidmVyIjoxLCJpc3MiOiJodHRwczo
vL29rdGEub2t0YS5jb20iLCJhdWQiOiJva3RhLjJiMTk1OWM4L
WJjYzAtNTZlYi1hNTg5LWNmY2ZiNzQyMmYyNiIsImlhdCI6MTY
4MTQ4MjczMiwiZXhwIjoxNjgxNDg2MzMyLCJqdGkiOiJJRC41V
2NSRk9MYkVidVdjNU1UeUd1VEtJVkZ0YkhlTkRxNUw3S1F0T3N
GM0k5b05JVHh3b2xVWGplRWxkT0FWRG5ZTWtKZnh3QnMiLCJwc
mVmZXJyZWRfdXNlcm5hbWUiOiJhbGlzYS5kdW5jYW5Ab2t0YS5
jb20iLCJnaXZlbl9uYW1lIjoiQWxpc2EiLCJmYW1pbHlfbmFtZ
SI6IkR1bmNhbiIsInpvbmVpbmZvIjoiVVMvUGFjaWZpYyIsImV
tYWlsX3ZlcmlmaWVkIjpmYWxzZSwiYXV0aF90aW1lIjoxNjgxN
Dc4OTY2LCJhdF9oYXNoIjoiWWRYUGdIZ1NpMTFNak5nM0Mza0w
5dyJ9.PcRMtX3moWNWiQ2HQuwbIOOR0zJw0MU8jmY0BDQruqLM
BFKfWKrE4SR3iHLWeGm5gD9B1S4w6iHxOQwWSJJJ_cUwSJnanX
P6akRv3mbfRtFhgVac_OryfOmlQ2FEP3AFwOf5Inu3QSckMMxU
WFOnAqGrBDthZbXmixH_T30mVamt7EEJofrpwj7rL9C4b2XNdA
2P1O_WudAY

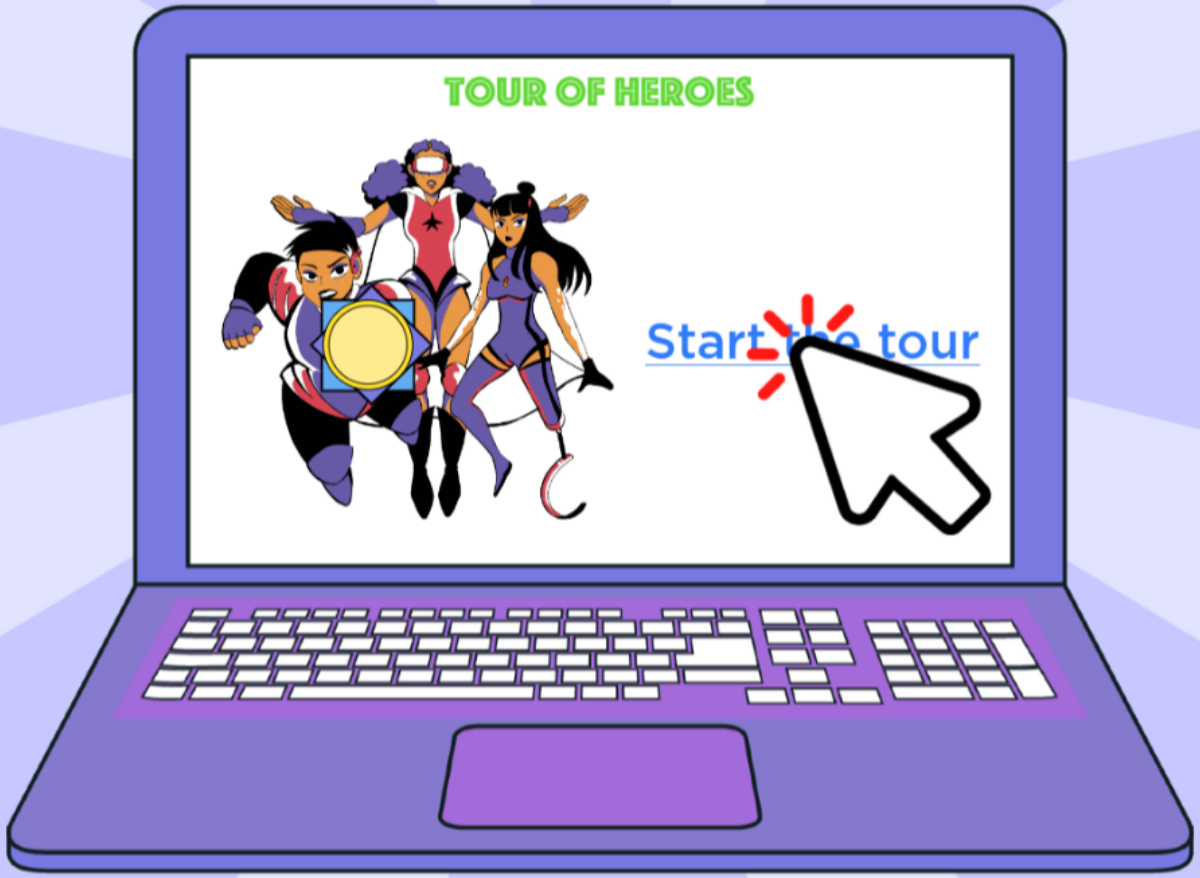



```
{
  "sub": "00u156fvj2G101d8",
  "name": "Ima Programmer",
  "locale": "en-US",
  "email": "ima.programmer@email.fake",
  "iss": "https://tourofheroes",
  "aud": "heroes.123",
  "role": "member",
  "iat": 168148273,
  "exp": 168148633,
  "nonce": "EbuWc5MTyGuTtbHeDnYMkJfxwBs"
}
```

ID token

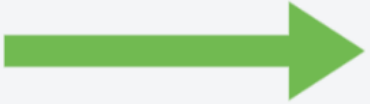
```
@Component({
  selector: 'app-welcome',
  template: `
    @if(name) {
      <span> Welcome, {{name}} </span>
    }
  `,
})
export class WelcomeComponent {
  auth = inject(AuthService);
  name = this.auth.getIdTokenClaim('name');
}
```

ID token



ID token

```
export const isMemberFn: CanMatchFn = (
  route: Route, segments: UrlSegment[], authService = inject(AuthService)) =>
  authService.getIdTokenClaim('role') === 'member')
);

const routes: Routes = [
  {
    path: 'tour',
    loadChildren: () => import('./tour/tour.routes').then(m => m.TOUR_ROUTES),
    canMatch: [isMemberFn]
  }
];
```



Access token

Access token

GET /api/heroes

access_token

Authorized?

200 OK [{hero}]

Not Authorized?

401 Unauthorized

403 Forbidden

Access token

What's in this token?

Access token

Let me introspect it

```
issuer = reputable_idp_url
expires = very_soon
audience = api://toh
```

Access token

Interceptor

```
GET /api/heroes
Authorization: Bearer <access_token>
```

Access token

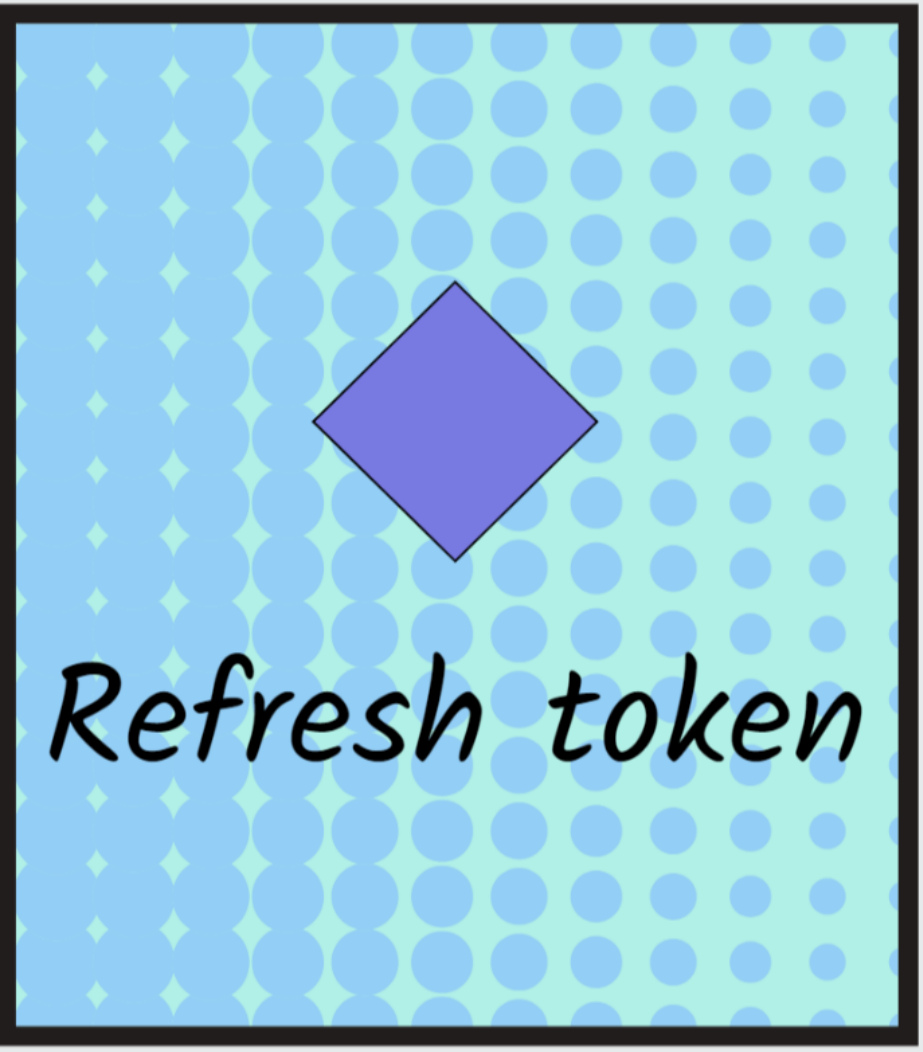
```
export const authInterceptor: HttpInterceptorFn =
  (req, next, authService = inject(AuthService)) => {

    let request = req;
    const allowedOrigins = ['/api/heroes'];
    if (!!allowedOrigins.find(origin => req.url.startsWith(origin))) {
      const token = authService.accessToken;
      const request = req.clone({
        headers: req.headers.set('Authorization', `Bearer ${token}`)
      });
    }

    return next(request);
}
```

Access token

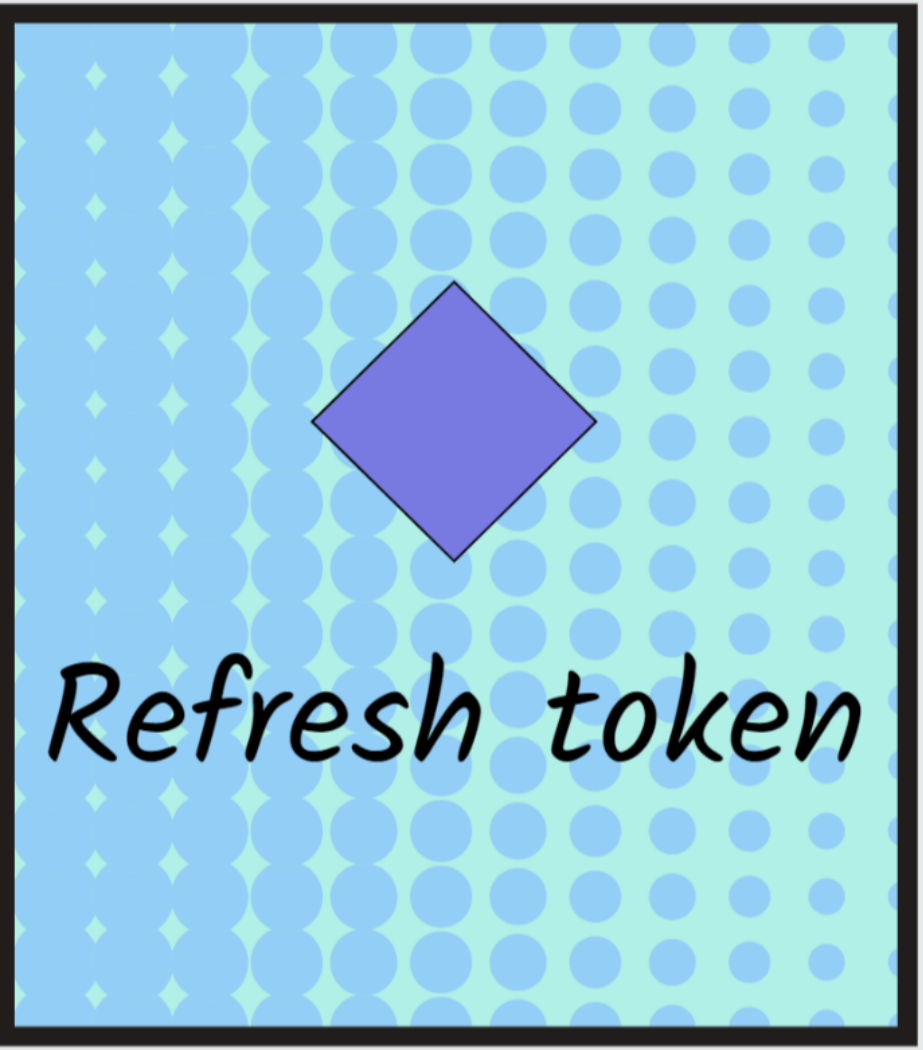
```
export const authInterceptor: HttpInterceptorFn =
  (req, next, authService = inject(AuthService)) => {

    let request = req;
    const allowedOrigins = ['/api/heroes'];
    if (!!allowedOrigins.find(origin => req.url.startsWith(origin))) {
      const token = authService.accessToken;
      const request = req.clone({
        headers: req.headers.set('Authorization', `Bearer ${token}`)
      });
    }

    return next(request);
}
```

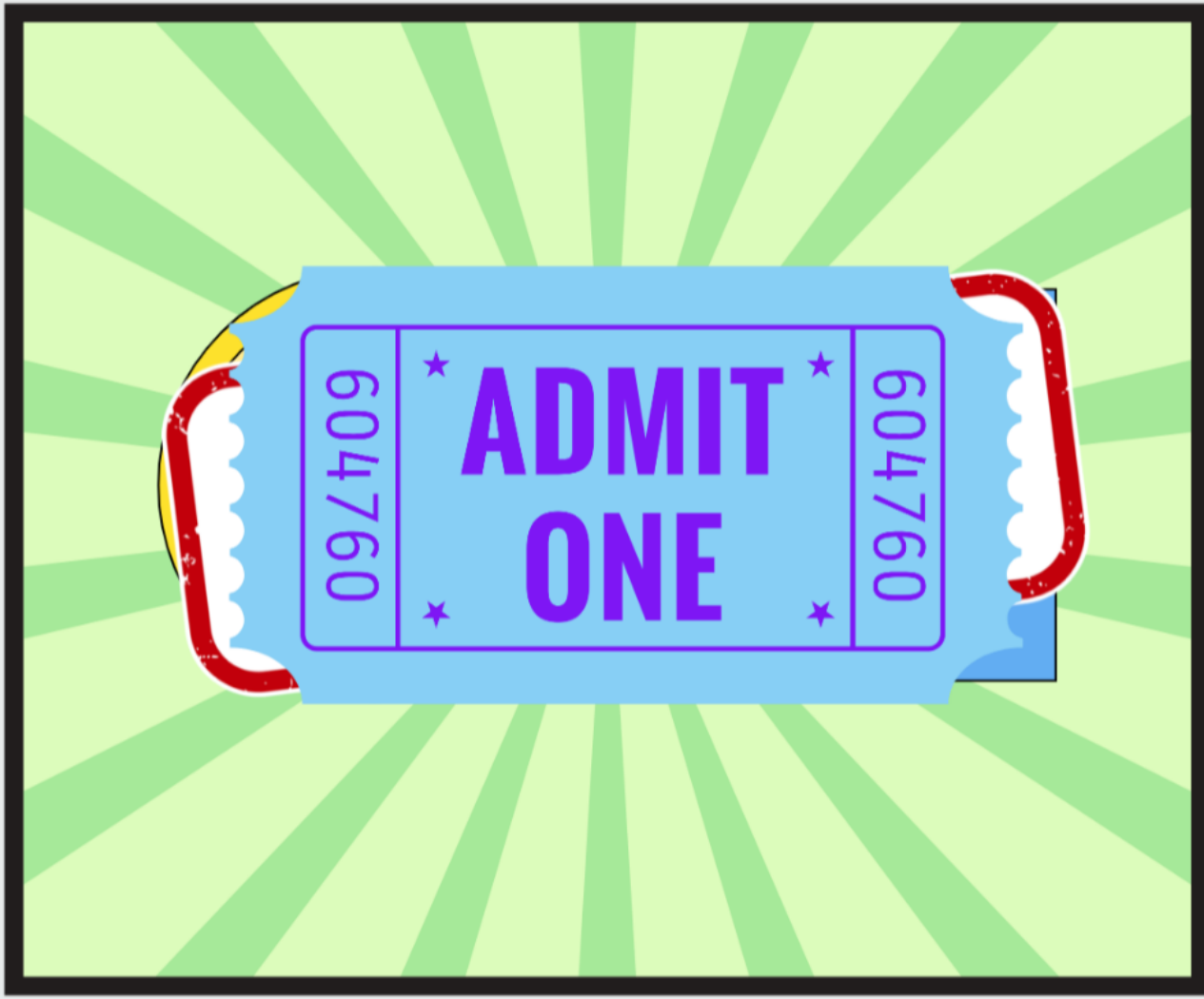
Access token

```
export const authInterceptor: HttpInterceptorFn
  (req, next, authService = inject(AuthService)) {

    let request = req;
    const allowedOrigins = ['/api/h      s']
    if (!!allowedOrigins.find(orig      req
      const token = authService.ac      ssTok
      const request = req.clone({
        headers: req.headers.set('Authoriz
      });
    }

    return next(request);
}
```

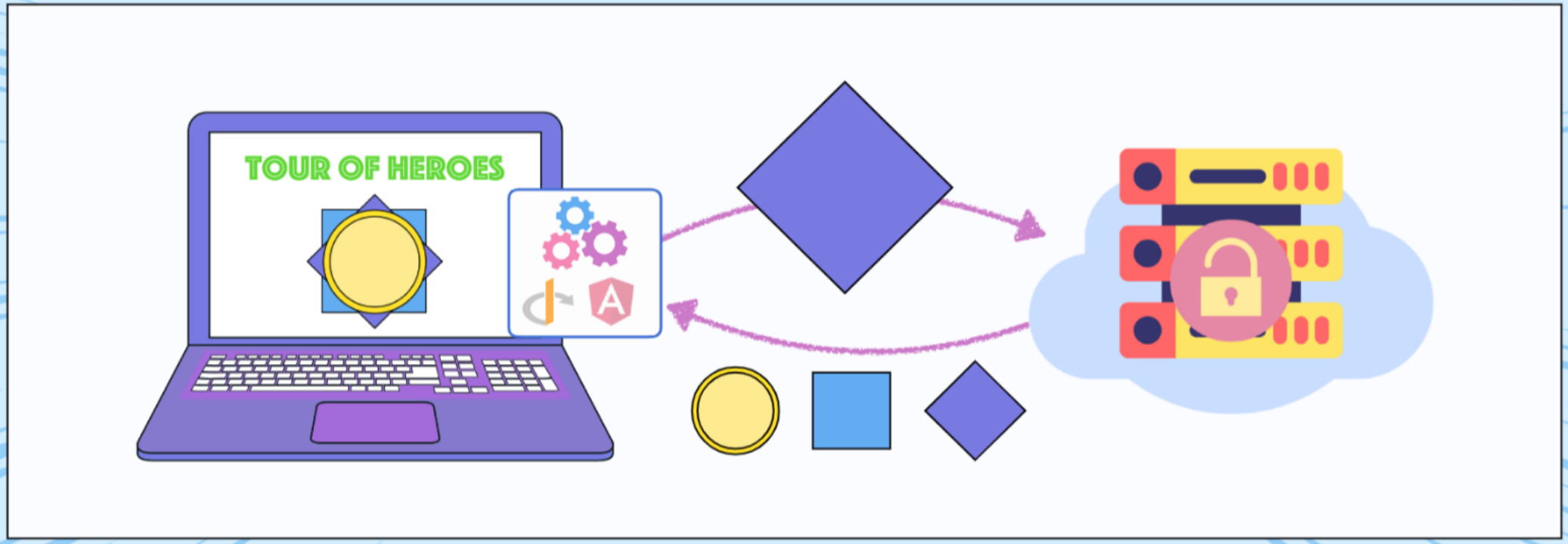

Refresh token


EXPIRED

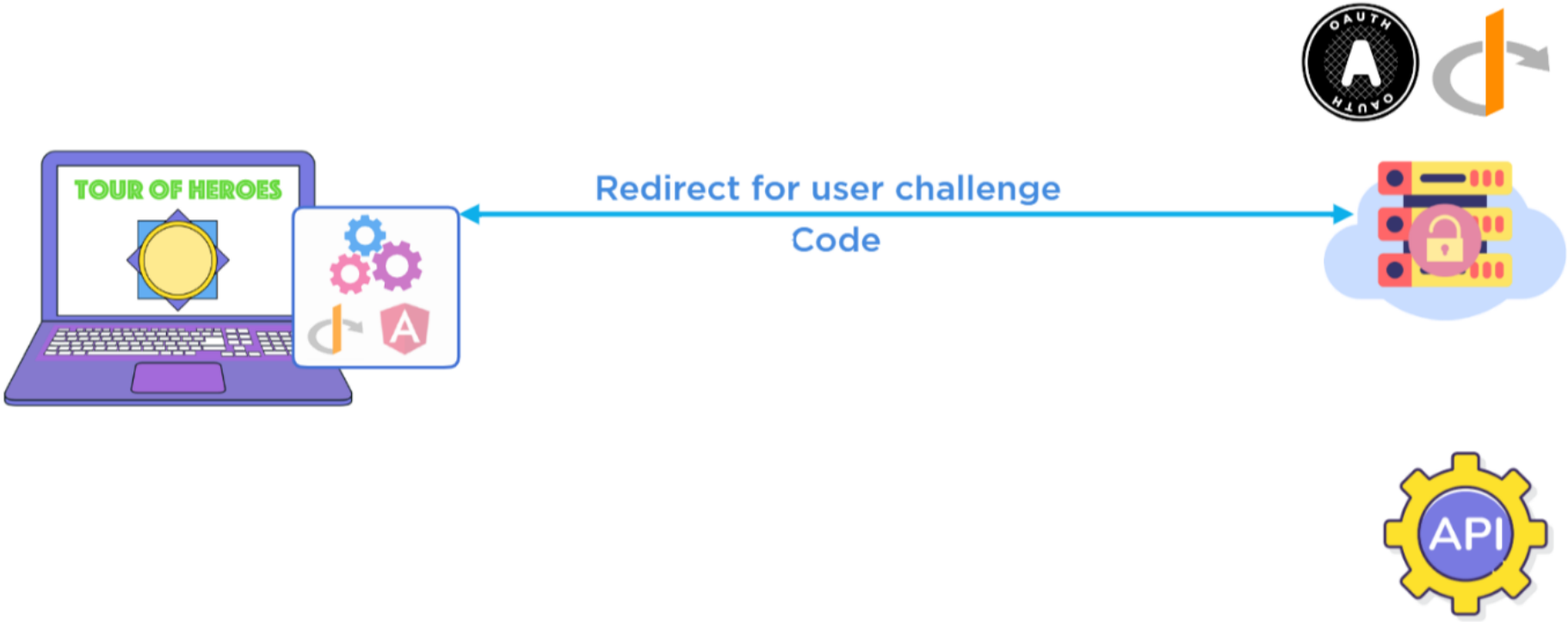

Refresh token


ADMIT
ONE
604760
604760

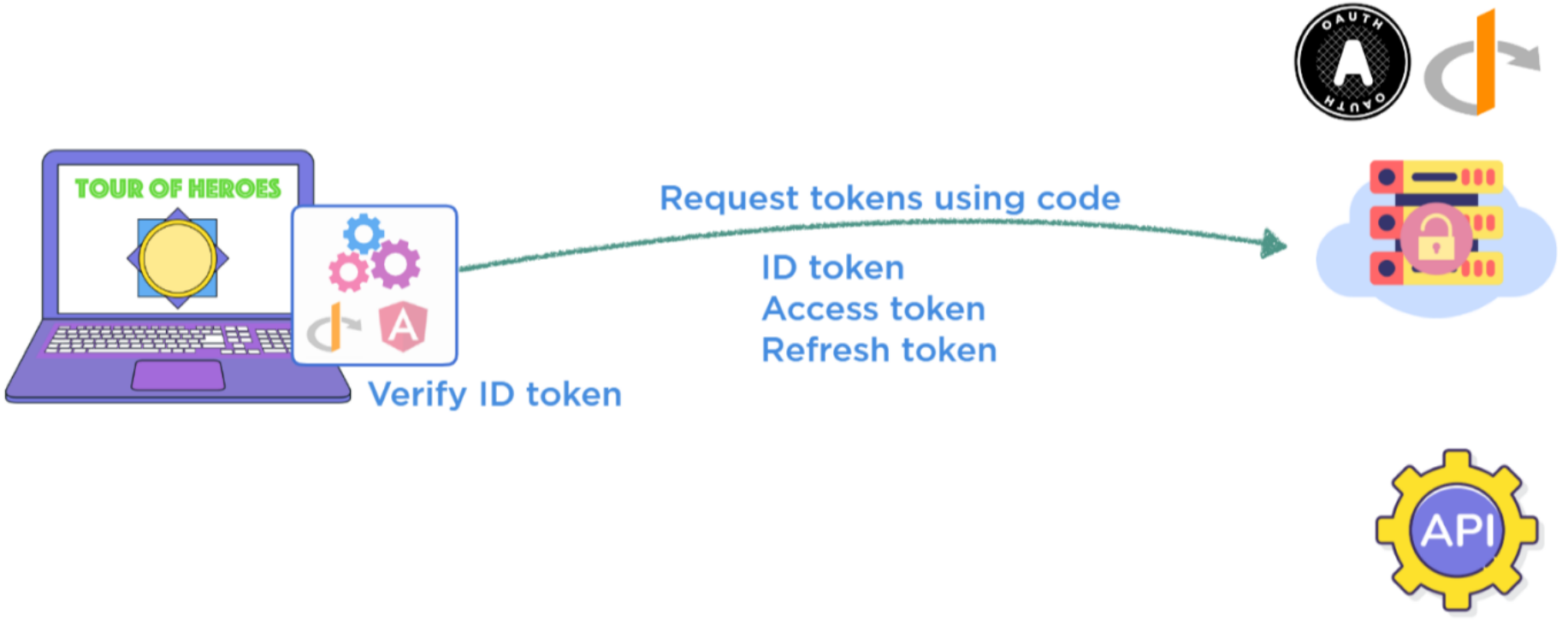
Refresh token



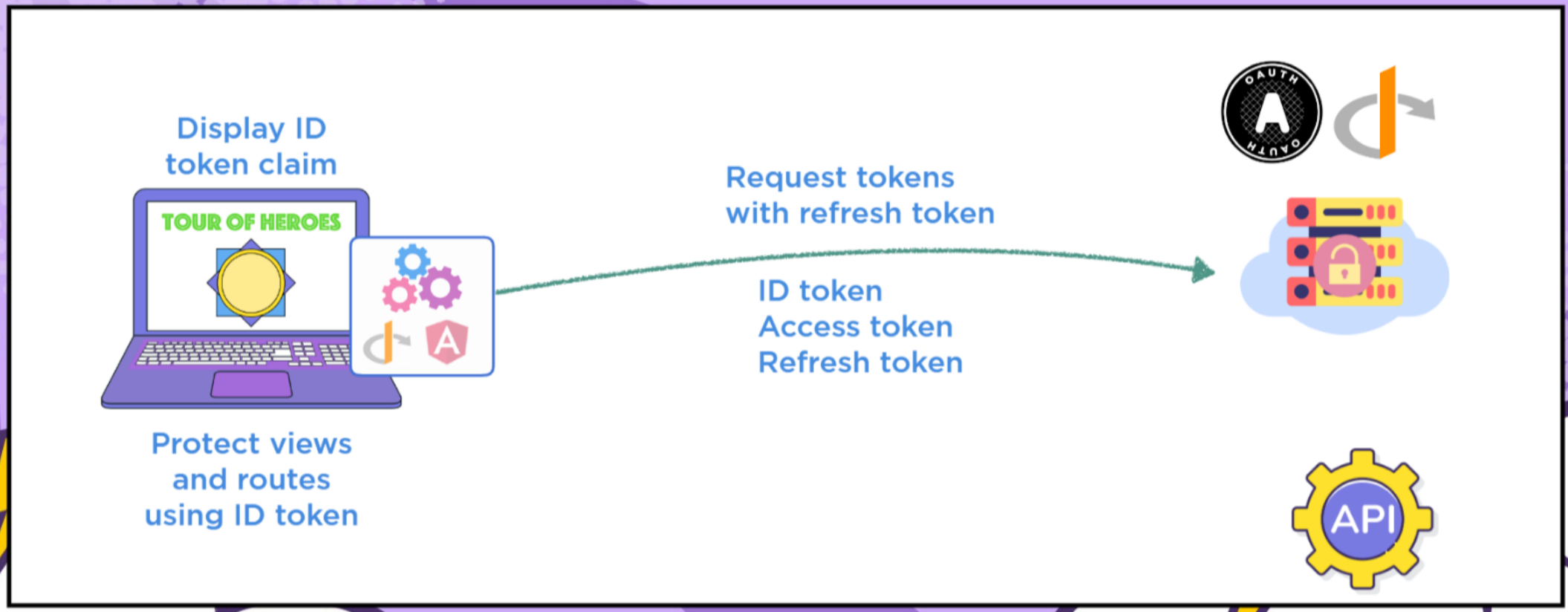
The Identity Guardians are hard at work



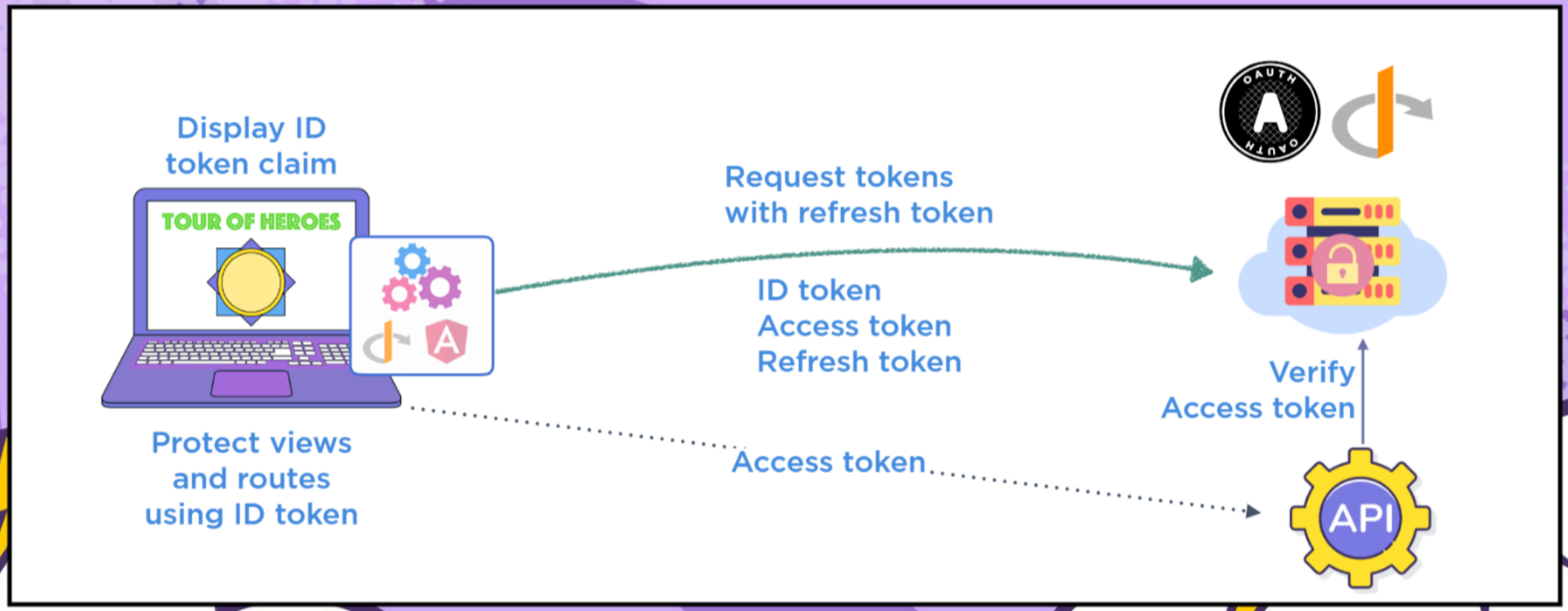
The Identity Guardians are hard at work



The Identity Guardians are hard at work



The Identity Guardians are hard at work



Endpoint discovery

OpenID Connect (OIDC)

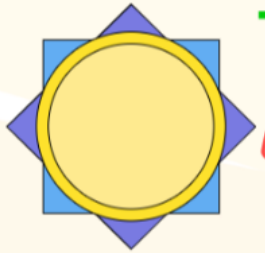
OAuth 2.0

Use OpenID certified products

★ Start with a reputable Identity Provider

openid.net/certification

★ Use your IdP's SDK or an OIDC certified client



The IDENTITY GUARDIANS Work for You

How Authentication and Authorization Work for SPAs

https://developer.okta.com/blog/2023/04/04/spa-auth-tokens

@AlisaDuncan

#identityguardians

Senior Developer Advocate at Okta

Google Developer Expert - Angular

Pluralsight Author