

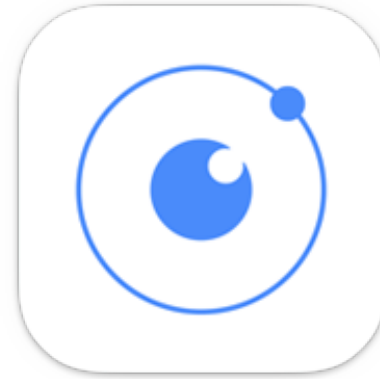
Formation Mobile Hybride

Ionic Cordova

Correction des exercices

```
git checkout step1
```

Ionic View



<https://docs.ionic.io/tools/view/>

Ionic View

Installer et s'enregistrer



App Store



Google play

S'authentifier au niveau du projet

```
ionic login
```


Créer une nouvelle app

```
ionic link
```

 <https://apps.ionic.io/apps/>

Vérifier l'APP ID

```
/ionic.config.json
```

 "app_id"

Envoyer le code vers Ionic View

```
ionic upload
```

Authentication

login

Email

Password

LOGIN

SIGN UP

signup

Email

Password

SIGN UP

Modals

Transition de la page Login => Signin

ModalController

Gestion des erreurs

ToastController

Spinner qui s'affiche à l'authentification

LoadingController

Authentication avec JWT

Web Token

Pas de cookie ni de session serveur.

Applicable pour tout type de
plateforme et tous devices

JSON Web Token

JSON Web Tokens are an open, industry
standard [RFC 7519](#) method for
representing claims securely between two
parties.

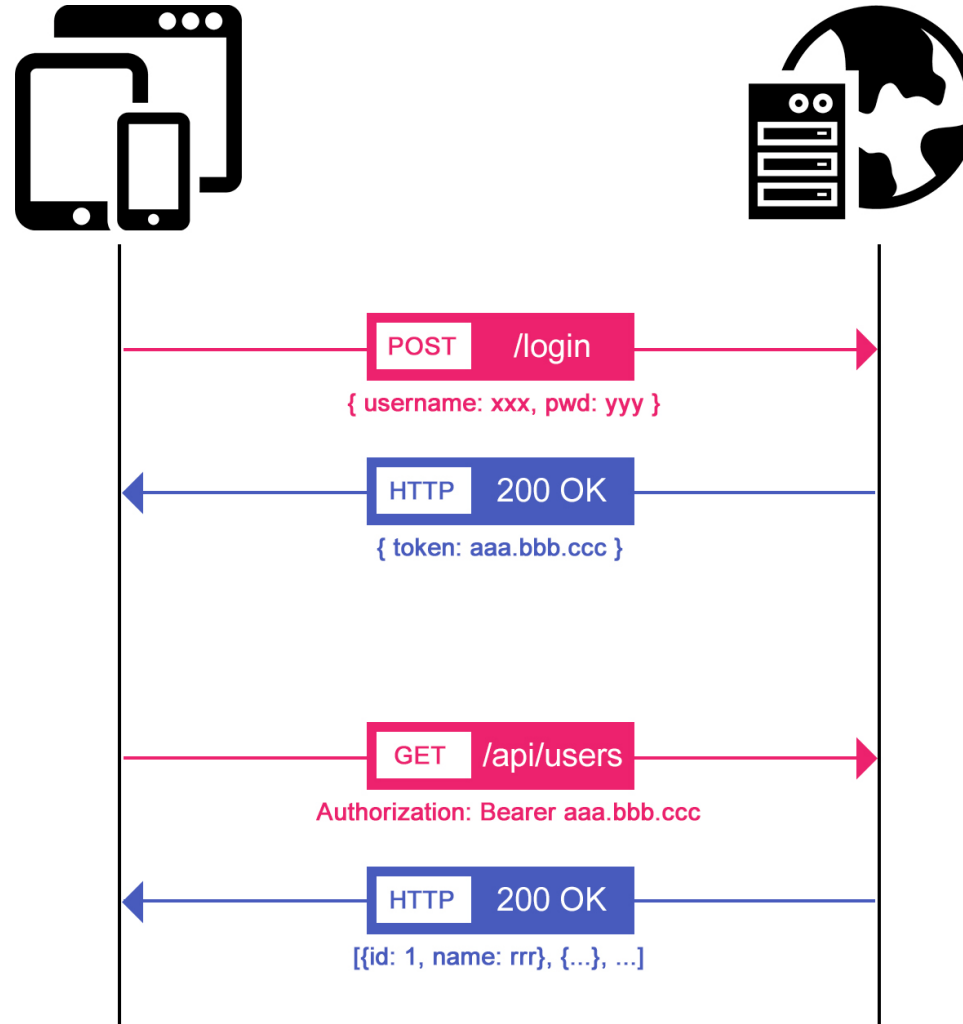


Header encodé en base64

Contenu encodé en base64

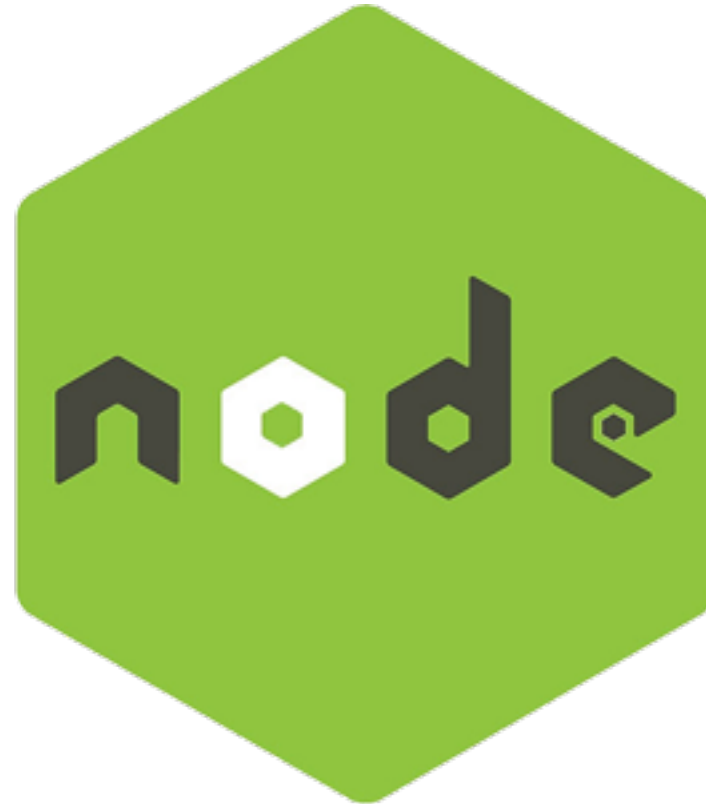
Encodage du header + contenu

Authentication avec JWT

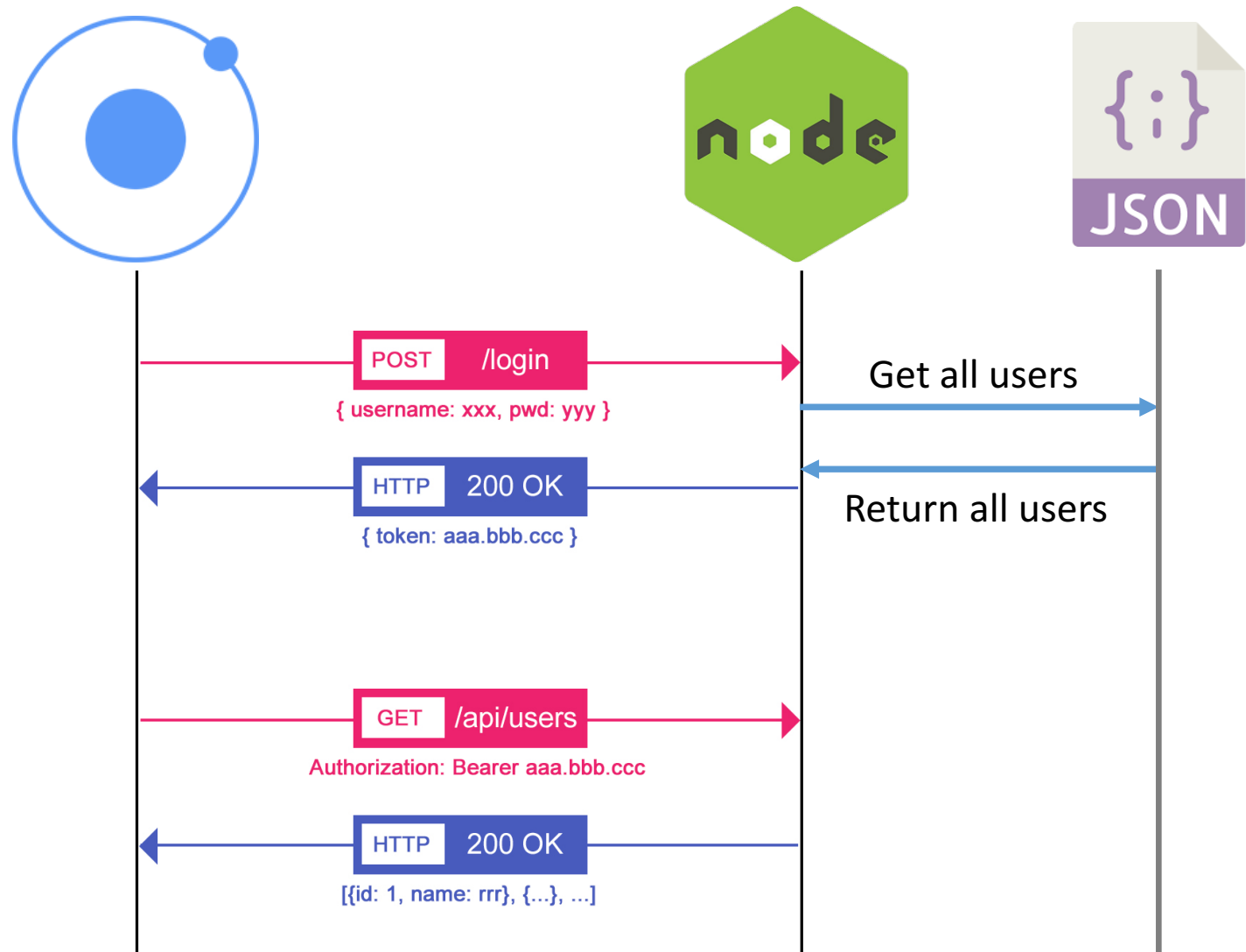


Authentication avec JWT

Rapide survol de NodeJS



Authentication avec JWT



Authentication avec JWT

```
git checkout step2
```

Authentication avec JWT

Installer JWT

```
npm install angular2-jwt --save
```

Installer les paquets pour le serveur node (en local)

```
npm install nodemon -g
```

```
npm install body-parser express fs jsonwebtoken --save
```

Lancer le serveur en local sur le port 4300

```
nodemon
```

Authentication avec JWT

Tester son serveur avec Postman



Authentication avec JWT

Les routes et la sécurité (authguard)



olegwn commented on 4 May

@Barryrowe right to the point. It's our pain to implement route guards with Ionic 3 :(Works fine for small apps, but for production ready - nope



3

Authentication avec JWT

app.module.ts

```
import { JwtHelper, AuthConfig, AuthHttp } from "angular2-jwt";  
import { Http, HttpClientModule, RequestOptions } from "@angular/http";  
import { Storage, IonicStorageModule } from "@ionic/storage";
```

```
// Auth Factory  
export function authHttpServiceFactory(http: Http, options: RequestOptions, storage: Storage) {  
  const authConfig = new AuthConfig({  
    tokenGetter: (() => storage.get('jwt')),  
  });  
  return new AuthHttp(authConfig, http, options);  
}
```

```
imports: [  
  HttpClientModule,  
  IonicStorageModule.forRoot({  
    name: 'myapp',  
    driverOrder: ['sqlite', 'indexeddb', 'websql']  
  })  
],
```

```
providers: [  
  JwtHelper,  
  {  
    provide: AuthHttp,  
    useFactory: authHttpServiceFactory,  
    deps: [Http, RequestOptions, Storage]  
  }  
]
```

REVISION

Exercices

Consignes

Créer 2 providers « endpoints » et « auth ». Le 1^{er} contiendra les 3 différentes urls de connexion au serveur (login, signup, getauth) qui seront appelées par auth afin d'établir la stratégie d'authentification.

Le provider « auth » sera utilisé par loginPage pour s'authentifier, signupPage pour créer l'utilisateur, app.component pour vérifier si l'utilisateur est connecté et tabsPage pour afficher l'e-mail de l'utilisateur courant dans le menu et la gestion du logout.

git checkout step3

Référence

<https://golb.hplar.ch/p/JWT-Authentication-with-Ionic-2-and-Spring-Boot>

Merci de votre attention