

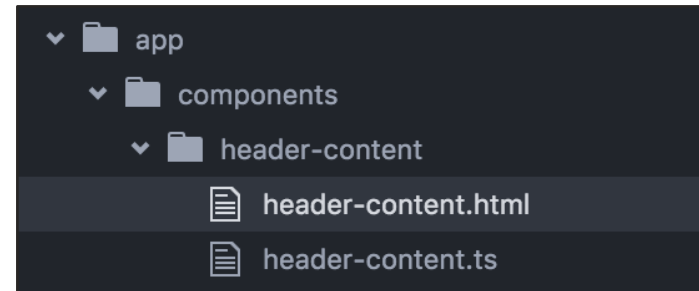
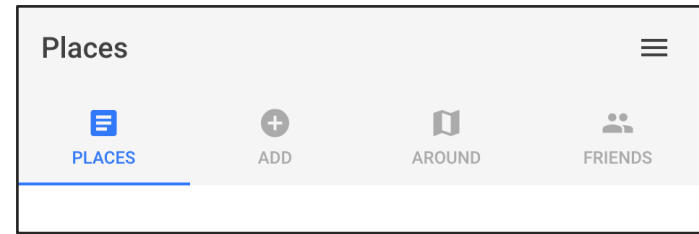
Formation Mobile Hybride Day 2

Cordova, Ionic

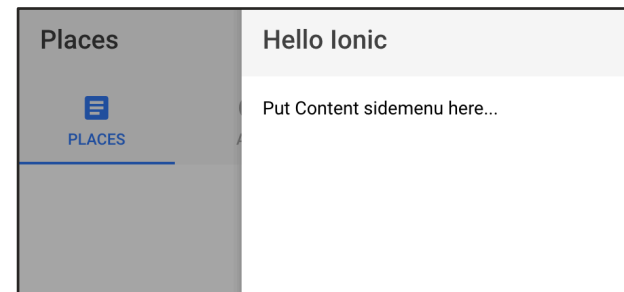
Agenda

1. Correction travaux pratiques
2. Ionic Modal
3. Test avec Ionic View
4. Authentification JWT

2) Mettre en place un composant header

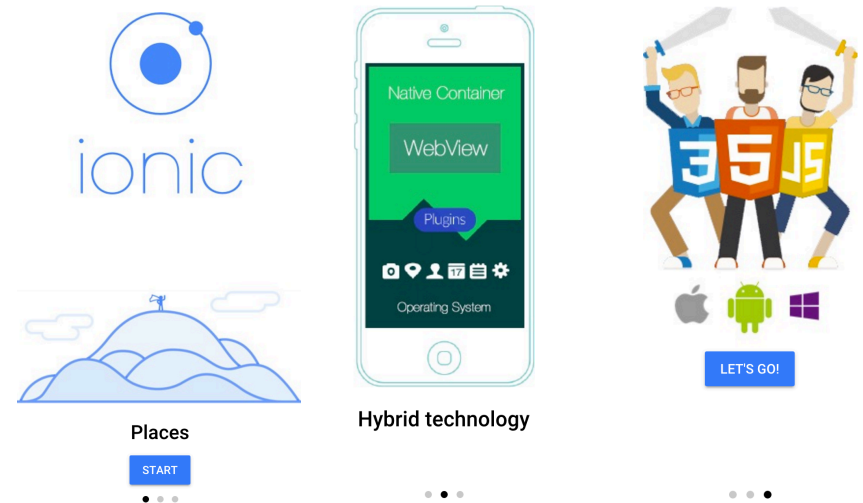


+ Mettre en place un sidemenu



```
<ion-menu [content]="content" side="right">
  <ion-toolbar>
    <ion-title>Hello Ionic</ion-title>
  </ion-toolbar>
  <ion-content padding>
    Put Content sidemenu here...
  </ion-content>
</ion-menu>
```

3) Mettre en place un slider sur la home »



4) Mettre en place une page login

← login

Username

Password

SIGN IN

Ionic Modal

Modal Component

```
import {Component} from '@angular/core';
import {Modal, NavController, ViewController} from 'ionic-angular';

@Component({
  template: `
    <ion-content padding>
      <h2>I'm a modal!</h2>
      <button (click)="close()">Close</button>
    </ion-content>`
})
class MyModal {
  constructor(
    private navCtrl: NavController) {}

  close() {
    this.navCtrl.dismiss();
  }
}
```

Ouvrir une modal

```
import { ModalController, NavParams } from 'ionic-angular';

@Component(...)
class HomePage {

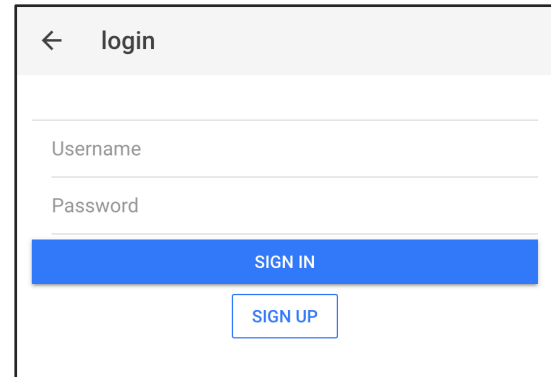
  constructor(public modalCtrl: ModalController) {

  }

  presentProfileModal() {
    let profileModal = this.modalCtrl.create(Profile, { userId: 8675309 });
    profileModal.present();
  }

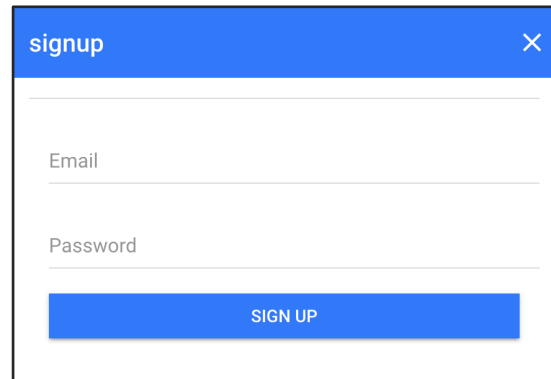
}
```

5) Ajouter un bouton « signup » sur le login



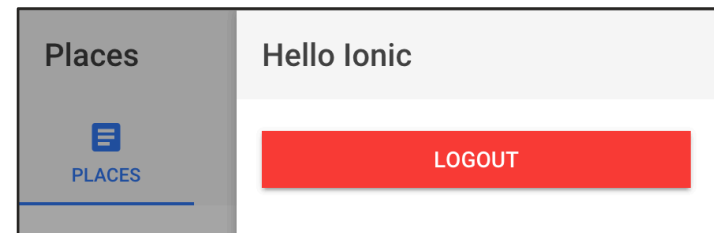
A login form with a header bar containing a back arrow and the text 'login'. Below the header are two input fields labeled 'Username' and 'Password'. At the bottom, there are two buttons: a blue 'SIGN IN' button and a white 'SIGN UP' button with a blue border.

Ouverture de la page en Modal



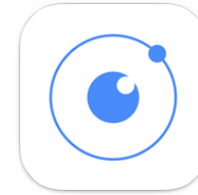
A signup modal form with a blue header bar containing the text 'signup' and a close button (X). Below the header are two input fields labeled 'Email' and 'Password'. At the bottom, there is a blue 'SIGN UP' button.

Redirection sur la Home au clique sur Logout

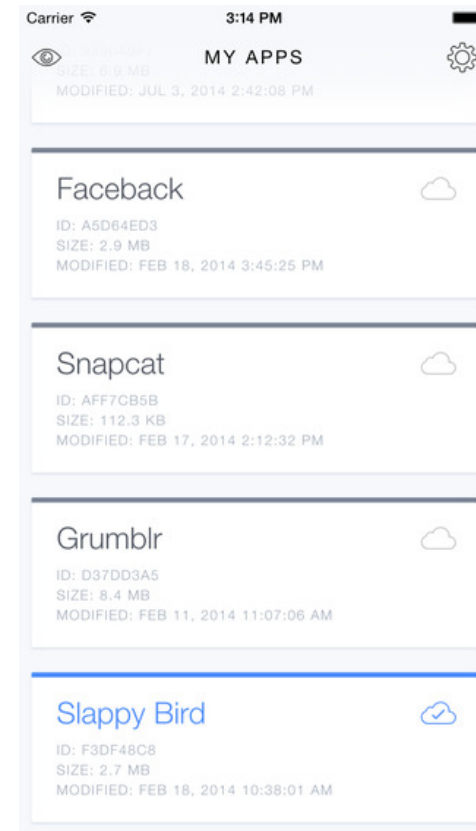


A home page layout with a sidebar on the left containing the text 'Places' and a blue button labeled 'PLACES'. The main content area has a header bar with the text 'Hello Ionic' and a red 'LOGOUT' button below it.

Testons sur nos téléphones



Ionic View App



\$ ionic upload

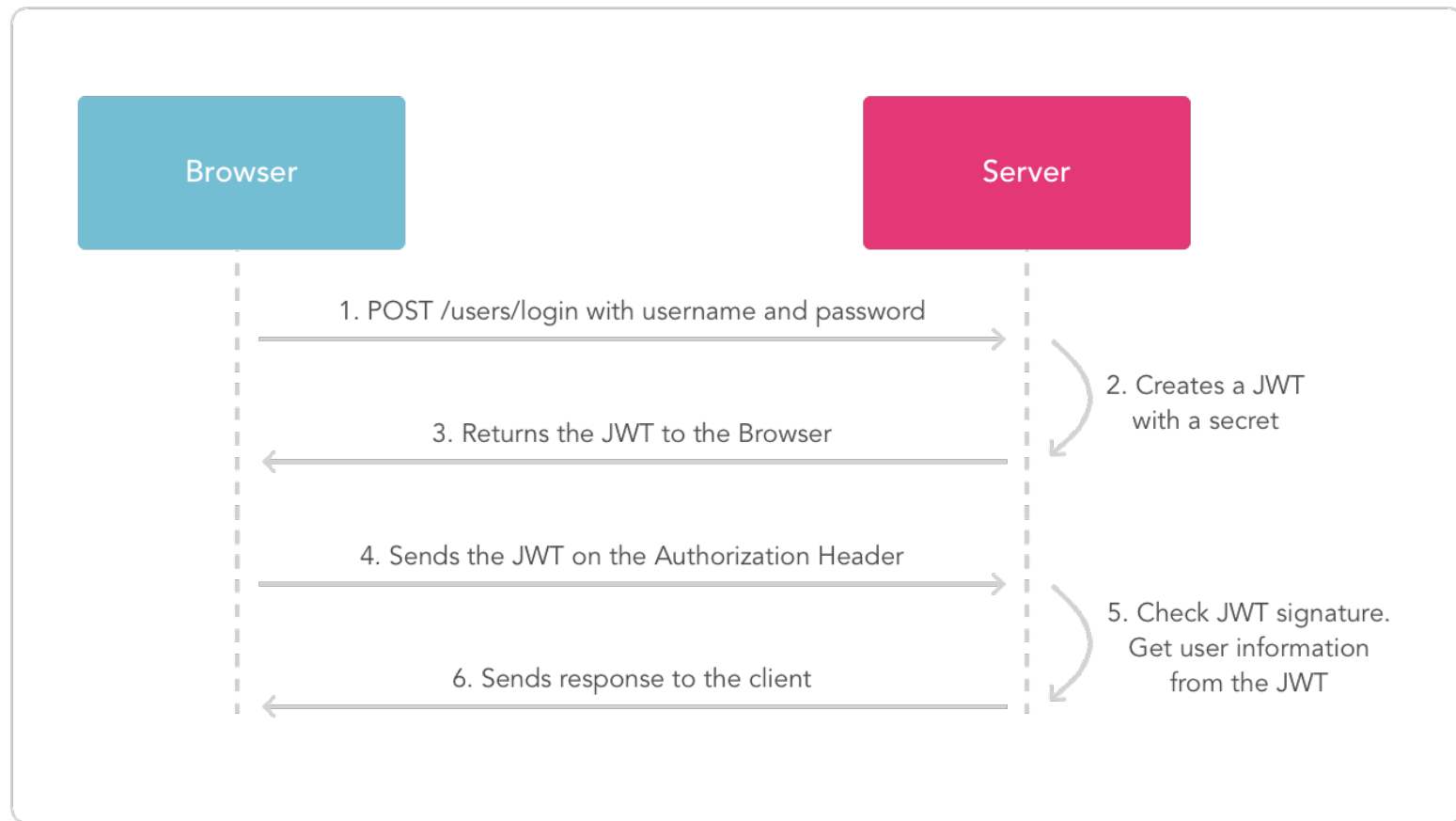
\$ ionic share email@client.io

Authentication avec JWT (Json Web Token)

Json Web Token


JWT.io

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



Example

nodejs-jwt-authentication-sample <https://github.com/kimak/nodejs-jwt-authentication-sample>

 **kimak / nodejs-jwt-authentication-sample**
forked from [auth0-blog/nodejs-jwt-authentication-sample](#)

Unwatch 1

Star 0

Fork 72

<> Code

Pull requests 0

Wiki

Pulse

Graphs

Settings

A NodeJS API that supports username and password authentication with JWTs — Edit

25 commits

1 branch

0 releases

10 contributors

Branch: master

New pull request


Create new file

Upload files

Find file

Clone or download

This branch is even with [auth0-blog:master](#). [Pull request](#) [Compare](#)

 **chenkie** Fixed repo url in package.json. Closes #13

Latest commit 0d7e229 on 25 Feb

.gitignore	First commit	a year ago
LICENSE	Create LICENSE	5 months ago
README.md	Update README.md	5 months ago
anonymous-routes.js	First commit	a year ago
config.json	First commit	a year ago
package.json	Fixed repo url in package.json. Closes #13	4 months ago
protected-routes.js	First commit	a year ago
quoter.js	First commit	a year ago
quotes.json	Updated quotes: typos, duplicate, etc.	5 months ago
server.js	cors module duplicate require	9 months ago
statusError.js	First commit	a year ago
user-routes.js	Merge pull request #16 from juukie/fix-error-message-as-plain-text	4 months ago

[README.md](#)

- **Installer angular2-jwt**

\$ npm install angular2-jwt --save

Configuration for Ionic 2

To configure angular2-jwt in Ionic 2 applications, use the factory pattern in your `@NgModule`. Since Ionic 2 provides its own API for accessing local storage, configure the `tokenGetter` to use it.

```
import { AuthHttp, AuthConfig } from 'angular2-jwt';
import { Http } from '@angular/http';
import { Storage } from '@ionic/storage';

let storage = new Storage();

export function getAuthHttp(http) {
  return new AuthHttp(new AuthConfig({
    headerPrefix: YOUR_HEADER_PREFIX,
    noJwtError: true,
    globalHeaders: [{'Accept': 'application/json'}],
    tokenGetter: (() => storage.get('id_token')),
  }), http);
}

@NgModule({
  imports: [
    IonicModule.forRoot(MyApp),
  ],
  providers: [
    {
      provide: AuthHttp,
      useFactory: getAuthHttp,
      deps: [Http]
    },
  ],
  ...

  bootstrap: [IonicApp],
  ...
})
```

SignUP

- **Créer un nouveau provider « Endpoints »**

\$ **ionic g provider endpoints**

<http://ionic-places-jwt.herokuapp.com>

```
import { Injectable } from '@angular/core';

@Injectable()
export class Endpoints {

  API_PATH: string = "http://ionic-places-jwt.herokuapp.com"

  getLogin(){
    return this.API_PATH + "/sessions/create"
  }

  getSignup(){
    return this.API_PATH + "/users"
  }

}
```

SignUP

- **Créer un nouveau provider « Auth »**

\$ **ionic g provider auth**

```
@Injectable()
export class Auth {

  contentTypeHeader: Headers = new Headers({"Content-Type": "application/json"});
  jwtHelper: JwtHelper = new JwtHelper();
  local: Storage = new Storage();
  user: string;
  error: string;

  constructor(private authHttp: AuthHttp, private endpoints: Endpoints) {
  }

  signup(credentials) {
    let observable = this.authHttp.post(this.endpoints.getSignup(),
      JSON.stringify(credentials), { headers: this.contentTypeHeader })
      .map(res => { return res.json()})
      return observable.toPromise().then((data)=>{
        this.authSuccess(data.id_token);
        return data;
      });
  }

  authSuccess(id_token) {
    this.local.set('id_token', id_token);
  }
}
```

SignUP

- **Créer un nouveau provider « Auth »**

\$ **ionic g provider auth**

```
@Injectable()
export class Auth {

  contentType: Headers = new Headers({"Content-Type": "application/json"});
  jwtHelper: JwtHelper = new JwtHelper();
  local: Storage = new Storage();
  user: string;
  error: string;

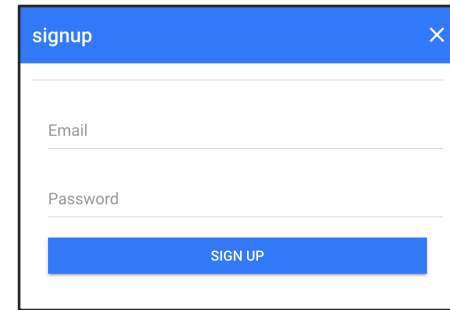
  constructor(private authHttp: AuthHttp, private endpoints: Endpoints) {
    this.local.get('id_token').then(token => {
      this.setUser(token);
    }).catch(error => {
      console.log(error);
    });
  }

  signup(credentials) {
    let observable = this.authHttp.post(this.endpoints.getSignup(),
      JSON.stringify(credentials), { headers: this.contentType })
      .map(res => { return res.json() })
      return observable.toPromise().then((data) => {
        this.authSuccess(data.id_token);
        return data;
      });
  }

  authSuccess(id_token) {
    this.local.set('id_token', id_token);
    this.setUser(id_token);
  }

  setUser(token) {
    this.user = this.jwtHelper.decodeToken(token);
  }
}
```

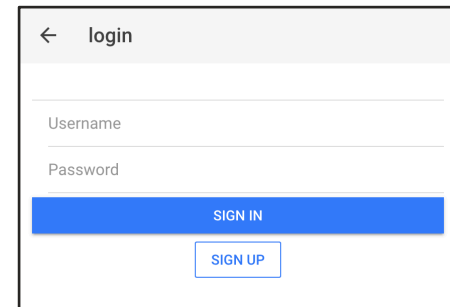
step5



A mockup of a signup form. It has a blue header bar with the text "signup" and a close button (X). Below the header are two input fields labeled "Email" and "Password". At the bottom is a blue button labeled "SIGN UP".

Mise en place du signup

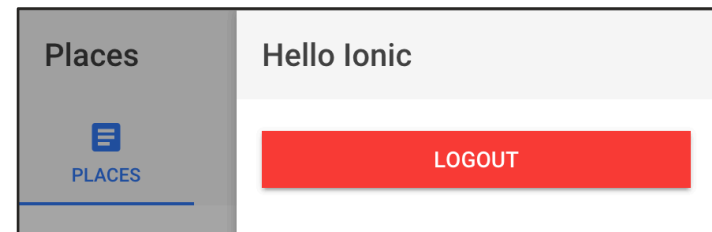
Mise en place du signIn



A mockup of a login form. It has a grey header bar with a back arrow and the text "login". Below the header are two input fields labeled "Username" and "Password". At the bottom are two buttons: a blue "SIGN IN" button and a white "SIGN UP" button with a blue border.

```
this.authHttp.post(this.endpoints.getLogin())
```

Affichage email + Logout



A mockup of a user profile page. It has a grey sidebar on the left with the text "Places" and a blue button labeled "PLACES". The main content area has a grey header bar with the text "Hello Ionic". Below the header is a red button labeled "LOGOUT".