Formation Mobile Hybride Day 2

Cordova, Ionic

Agenda

Pratique

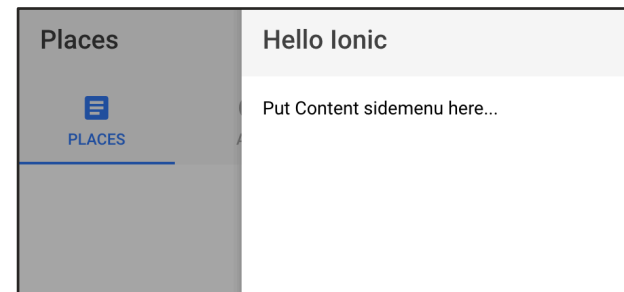## 2) Mettre en place un composant header
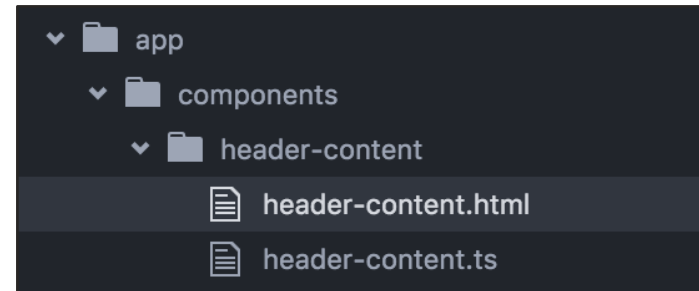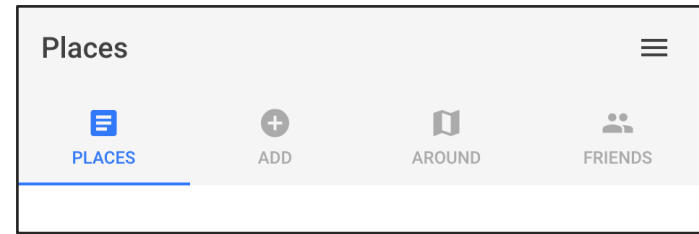
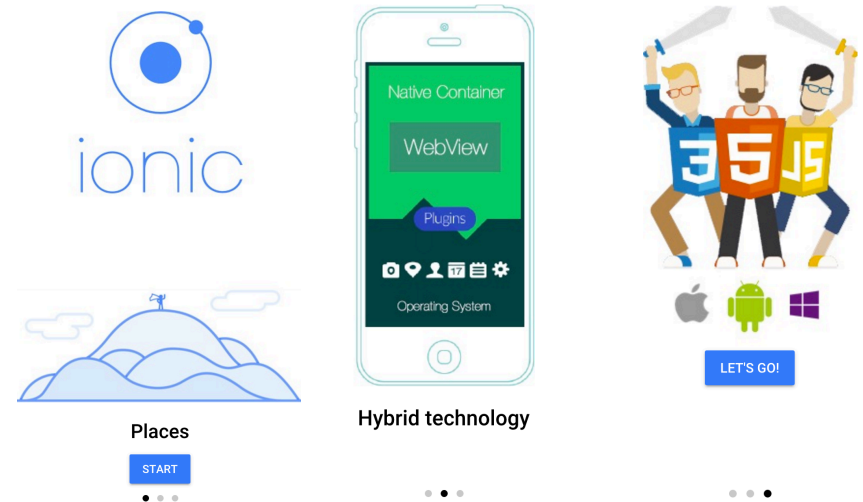## + Mettre en place un sidemenu


```
<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>
```

Pratique

3) Mettre en place un slider sur la home »

Places

START

Hybrid technology

LET'S GO!

4) Mettre en place une page login

← login

Username

Password

SIGN IN

Ionic Modal

Modal Component

```typescript
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 viewCtrl: ViewController) {}

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

Ouvrir une modal

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


@Component(...)
class HomePage {

  constructor(public modalCtrl: ModalController) {

  }


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


}
```
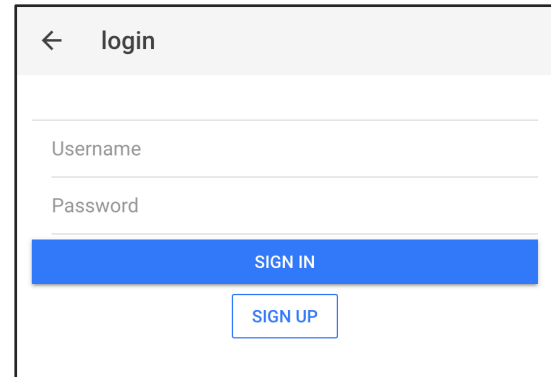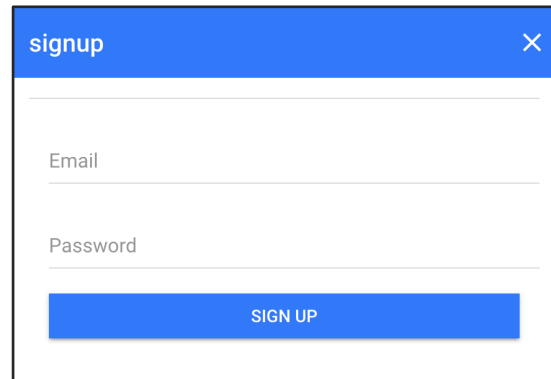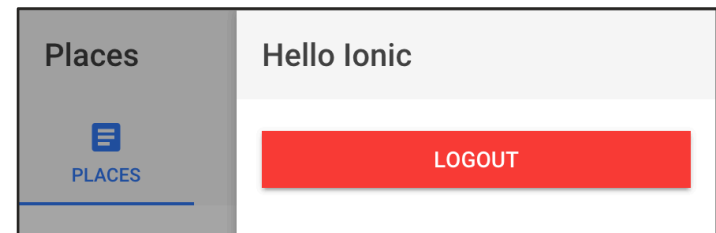
Pratique

5) Ajouter un bouton « signup » sur le login

Ouverture de la page en Modal

Redirection sur la Home au clique sur Logout

Testons sur nos téléphones
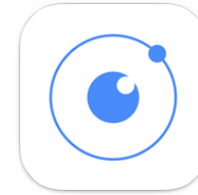
Ionic View App
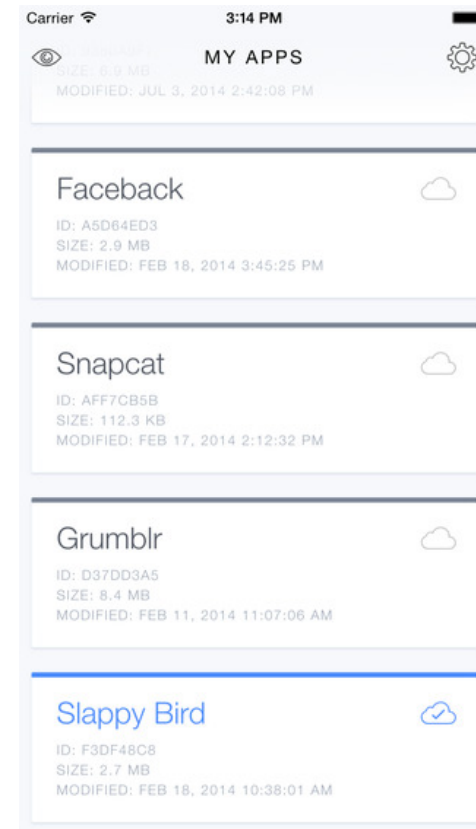
$ ionic upload

$ ionic share email@client.io
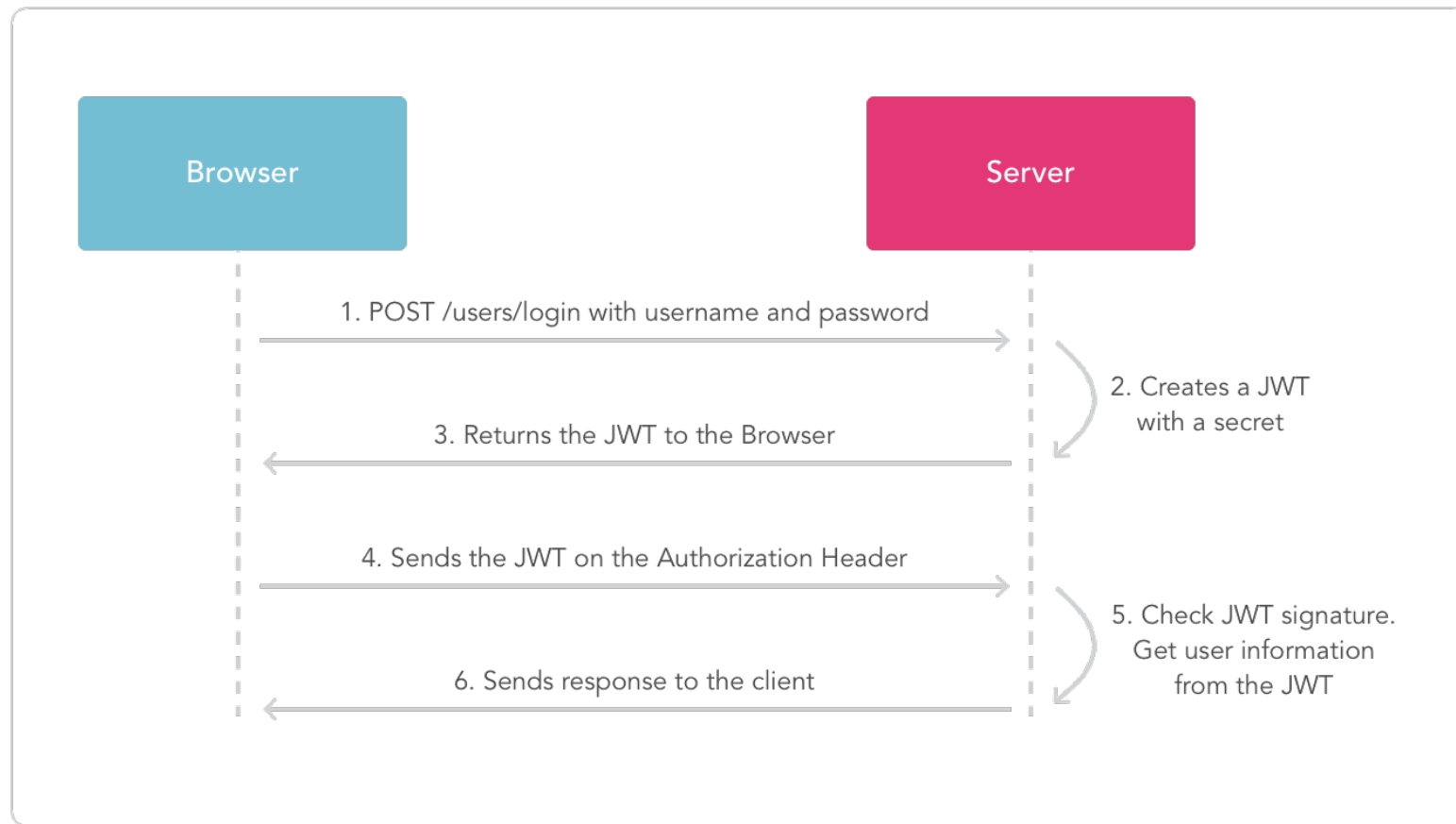
Authentification avec JWT (Json Web Token)

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

SignUP

- **Installer angular2-jwt**

$ npm install angular2-jwt --save

Configuation 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

```typescript
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 {

  contentHeader: 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.contentHeader })
      .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

```typescript
@Injectable()
export class Auth {

  contentHeader: 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.contentHeader })
      .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);
  }
}
```
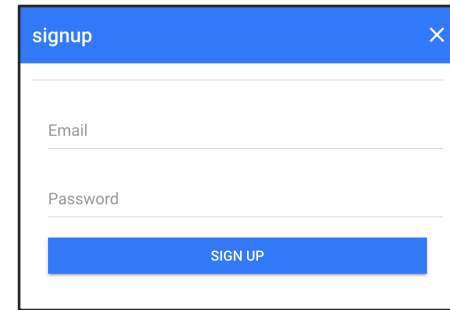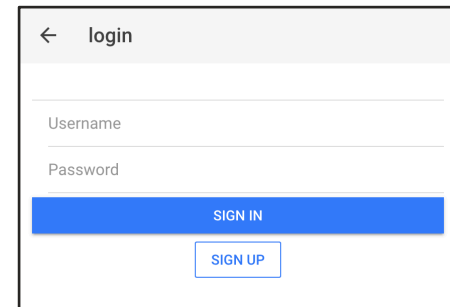
Pratique

step5

Mise en place du signup
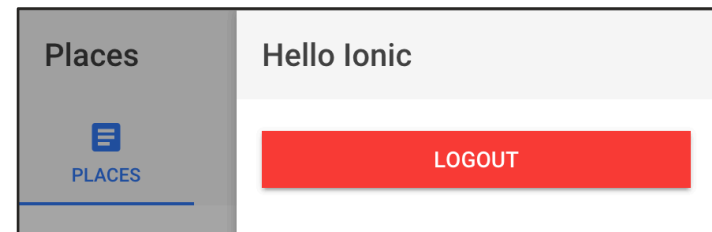
Mise en place du signIn

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

Affichage email + Logout