

Développement Mobile Hybride

IONIC 4



HTTP
JSON

Serveur HTTP

API REST
PHP
JEE
.Net
NodeJS

HTTP
JSON



HTTP
JSON



Mohamed Youssfi

Laboratoire Signaux Systèmes Distribués et Intelligence Artificielle (SSDIA)

ENSET, Université Hassan II Casablanca, Maroc

Email : med@youssfi.net

Supports de cours : <http://fr.slideshare.net/mohamedyoussfi9>

Chaîne vidéo : <http://youtube.com/mohamedYoussfi>

Recherche : http://www.researchgate.net/profile/Youssfi_Mohamed/publications



Développement Mobile

- Trois manières pour développer des applications mobiles :
 - Développement mobile natif
 - Développement mobile Web
 - Développement mobile hybride

Application mobile native ?

- Une application native est une application mobile développée pour un des systèmes d'exploitation utilisés par les smartphones et tablettes (iOS, Android, Windows Phone etc.).
- Elle est développée avec un langage spécifique à son système d'exploitation (App Store, Google Play, Windows Store, etc.)
- Elle est distribuée uniquement par l'intermédiaire des plateformes d'applications qui contrôlent sa nature et ses contenus.
- Lorsque l'application est payante ces plateformes prélèvent une part du prix de vente.
- Les applications natives utilisent toutes les fonctionnalités offertes par le téléphone (GPS, Local Storage, Push Notifications, SMS, Appels Téléphoniques, Caméra, Accéléromètre, Mode connecté et mode non connecté, Gestion de l'énergie, Mémoires embarquées, etc...)

Société	Système d'exploitation	Langage de développement	Plateforme
Apple	iOS	Objective-C	AppStore
Microsoft	Windows Phone	C#	MarketPlace
Google	Android	Java	GooglePlay

Mobile Web Application

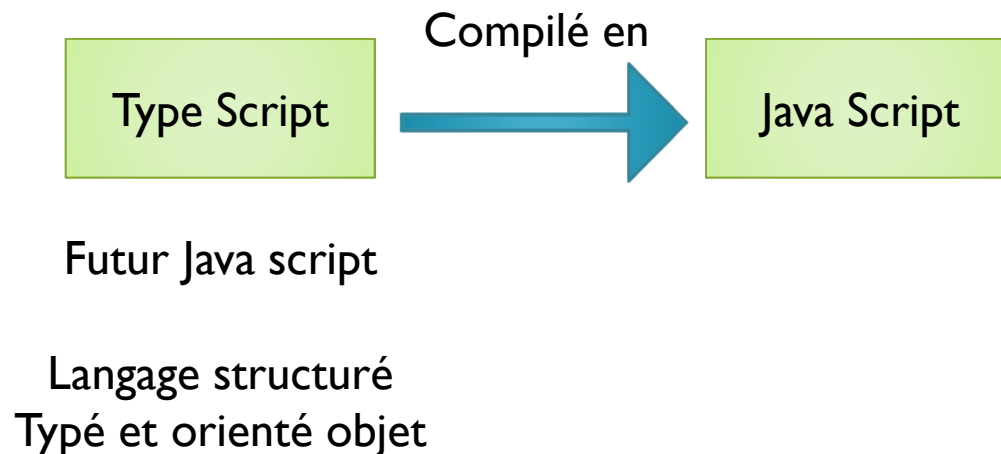
- Une web application est une application mobile développée en HTML, CSS, Java Script accessible et exécutable par le biais d'un navigateur Internet pour téléphone mobile.
- Elle utilise le navigateur du smartphone et ne nécessite pas forcément de télécharger l'application. Elle est accessible par tous les smartphones quelques soient leur marque et système d'exploitation.
- Cependant une web application ne prend pas en considération les différents modèles de smartphones et les différents systèmes d'exploitation.
- Une Application mobile Web n'utilise pas les fonctionnalités offertes par le système du téléphone

IOINC 1, 2 et 3

- IONIC 1 est basé sur Angular 1 (**Angular JS**) :
 - Première version de IONIC qui est la plus populaire.
 - Angular JS est basé sur une architecture MVC coté client. Les applications IONIC 1 sont écrites en Java Script.
- IONIC 2 est basé Angular 2 (**Angular**) :
 - Angular 2 est une réécriture de Angular 1 qui est plus performante, mieux structurée et représente le futur de Angular.
 - Les applications de Angular2 sont écrites en Type Script qui est compilé et traduit en Java Script avant d'être exécuté par les Browsers Web.
 - Angular 2 est basée sur une programmation basée sur les Composants Web (Web Components)
- IONIC 3 est basé sur Angular 4 est une simple mise à jour de IONIC 2 (La version Angular 3 a été abandonnée)

Angular avec Type Script

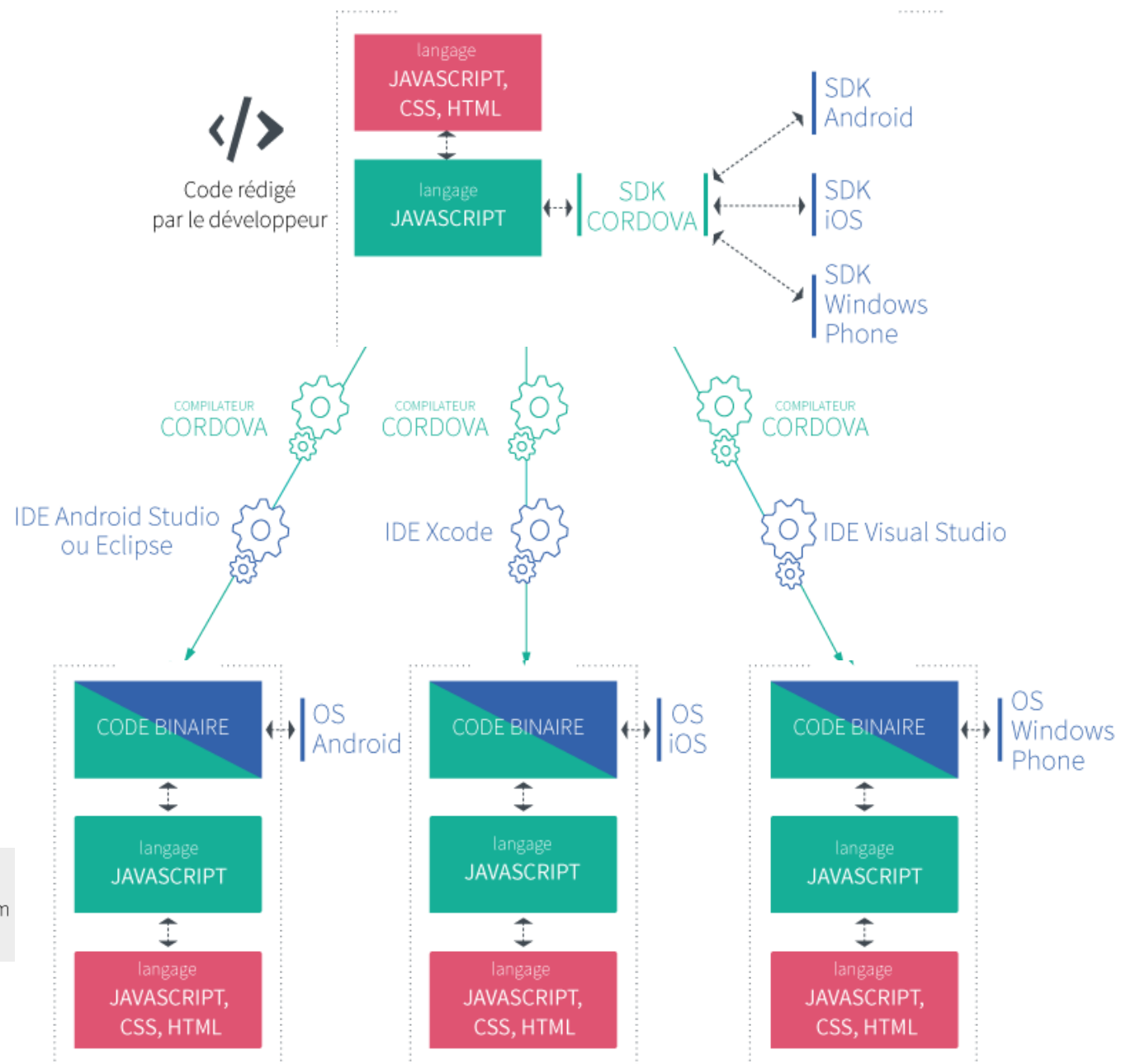
- Pour développer une application Angular il est recommandé d'utiliser Type Script qui sera compilé et traduit en Java Script.
- Type Script est un langage de script structuré et orienté objet qui permet de simplifier le développement d'applications Java Script



Application mobile hybride

- Une application hybride combine
 - Des éléments d'une application web HTML5, CSS, Java Script
 - Pour la partie Présentation des éléments de l'interface on utilise
 - Des Framework CSS créées pour les applications mobiles (Boot Strap)
 - Des Framework Java Script (Angular JS, JQuery Mobile, Dojo, Sencha Touch)
 - Pour la partie Interaction avec le serveur, on utilise des Frameworks Ajax (**Angular**, JQuery Mobile, ...)
 - Des éléments d'une application native permettent d'utiliser toutes les fonctionnalités natives des smartphones ((**GPS, Local Storage, Push Notifications, SMS, Appels Téléphoniques, Caméra, Accéléromètre, Mode connecté et mode non connecté, Gestion de l'énergie, Mémoires embarquées, etc.**)
 - Pour utiliser interagir avec les fonctionnalités natives, l'application mobile native utilise les plugins du Framework **CORDOVA** (Apache) ou PhoneGap (Adobe).
 - En plus Cordova offre des outils qui permettent de :
 - Générer des applications mobiles natives pour les différentes plateformes
 - Tester ces applications sur des émulateurs appropriés
 - De plus elle pourra être distribuée en tant qu'application sur les plateformes d'applications (App Store, Google Play Store, Windows Store)

Développement Mobile Hybrides



Plateforme de développement mobile hybride

- Appcelerator Titanium (gratuit)
- Win D v mobile (payant)
- IBM Worklight (MobileFirst) (gratuit dans sa version d veloppeur et payant la partie serveur)
- Adobe Phone Gap (Payant   partir d'une application de plus de 50 MB)
- **IONIC (GRATUIT, OPENSOURCE)**

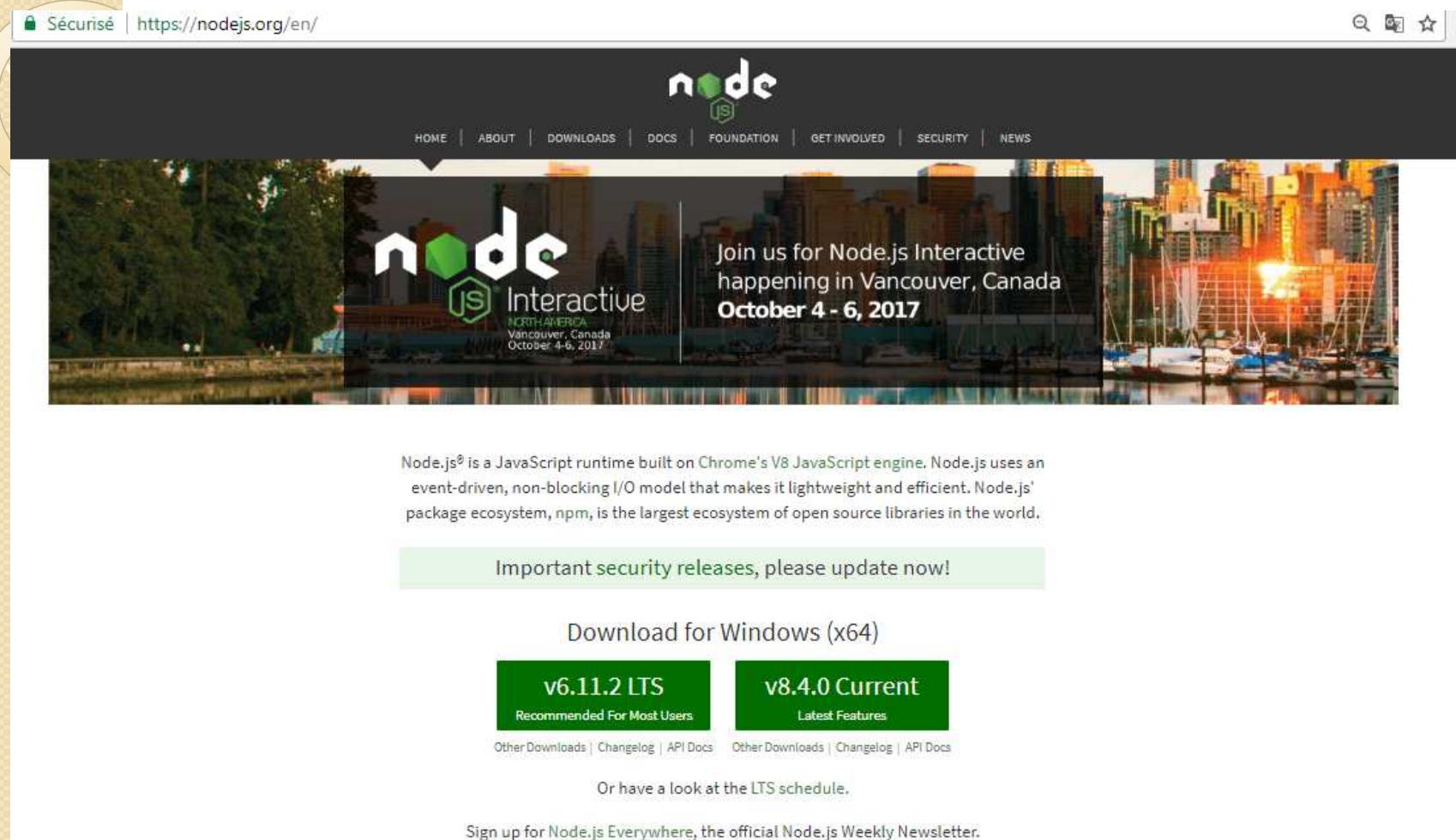
Qu'est-ce que IONIC Framework ?

- Ionic Framework est un mélange d'outils et de technologies pour développer des applications mobiles hybrides rapidement et facilement.
- Il s'appuie sur :
 - **Angular** pour la partie application web du framework
 - **Cordova** pour la partie construction des applications natives.
- Il s'appuie aussi sur la plateforme **NodeJS** :
 - Plus précisément **NPM** (Node Package Manager)
 - Et pour démarrer un serveur web local basé sur NodeJS pour tester la partie web de l'application en local.
- Ce Framework open source permet de développer une application déployable sur plusieurs environnements tel que :
 - Application mobile Web
 - Une application mobile pour des systèmes tel que :
 - Androïde
 - iOS
 - Windows Phone...

Installation

- Installer d'abord [Node.js](#) 6.
- Ensuite, installer Cordova et Ionic.
- Pour les applications Android, il faut installer le SDK de Android
- Pour les application IOS, il faut installer XCode.

Premier outil à installer : NodeJS



The screenshot shows the Node.js website with a dark header containing the Node.js logo and navigation links: HOME, ABOUT, DOWNLOADS, DOCS, FOUNDATION, GET INVOLVED, SECURITY, and NEWS. Below the header is a large banner for Node.js Interactive, featuring the Node.js logo and text: "Join us for Node.js Interactive happening in Vancouver, Canada October 4 - 6, 2017". Below the banner, a paragraph describes Node.js as a JavaScript runtime built on Chrome's V8 JavaScript engine, using an event-driven, non-blocking I/O model, and mentions npm as the largest ecosystem of open source libraries. A green box highlights "Important security releases, please update now!". Below this, a section for "Download for Windows (x64)" features two green buttons: "v6.11.2 LTS Recommended For Most Users" and "v8.4.0 Current Latest Features". Links for "Other Downloads", "Changelog", and "API Docs" are provided for both versions. Below the download section, a link says "Or have a look at the LTS schedule.", and at the bottom, a link says "Sign up for Node.js Everywhere, the official Node.js Weekly Newsletter."

Sécurisé | <https://nodejs.org/en/>

node

HOME | ABOUT | DOWNLOADS | DOCS | FOUNDATION | GET INVOLVED | SECURITY | NEWS

node Interactive
NORTH AMERICA
Vancouver, Canada
October 4-6, 2017

Join us for Node.js Interactive
happening in Vancouver, Canada
October 4 - 6, 2017

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

Important security releases, please update now!

Download for Windows (x64)

v6.11.2 LTS
Recommended For Most Users

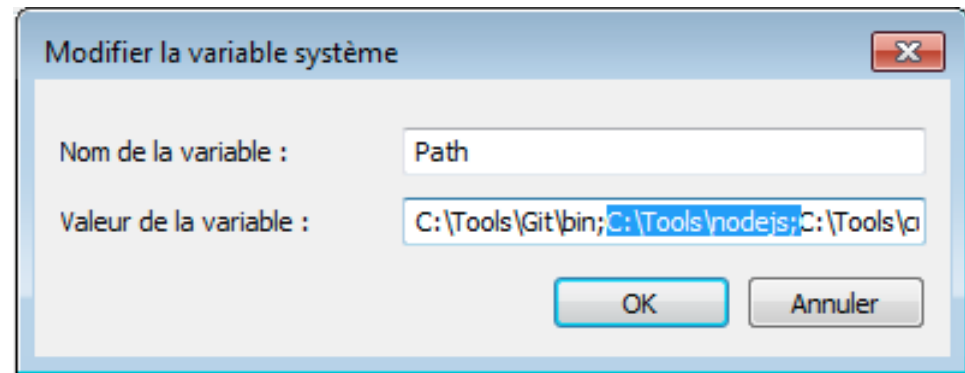
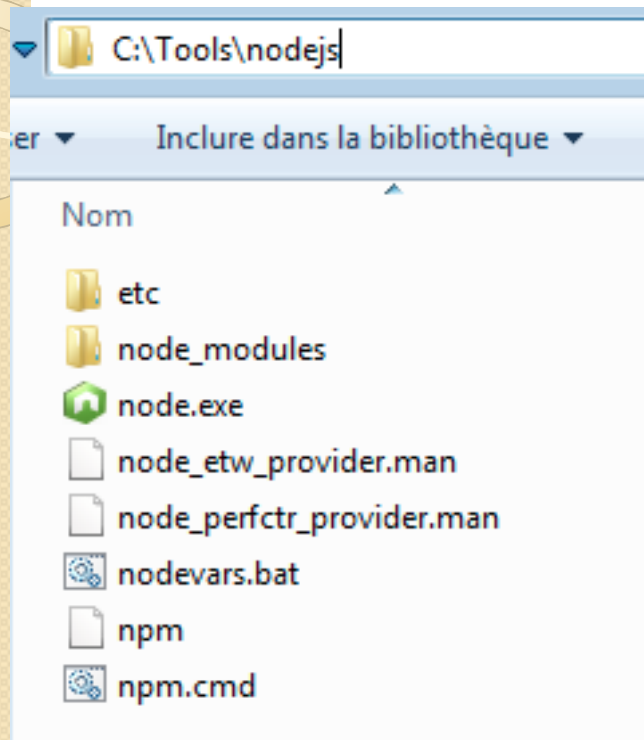
v8.4.0 Current
Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#) [Other Downloads](#) | [Changelog](#) | [API Docs](#)

[Or have a look at the LTS schedule.](#)

[Sign up for Node.js Everywhere, the official Node.js Weekly Newsletter.](#)

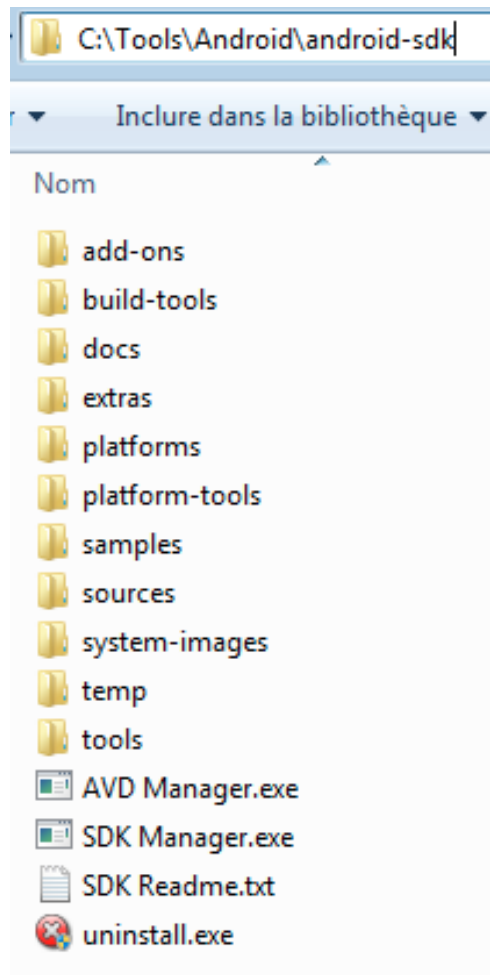
Installer Node JS



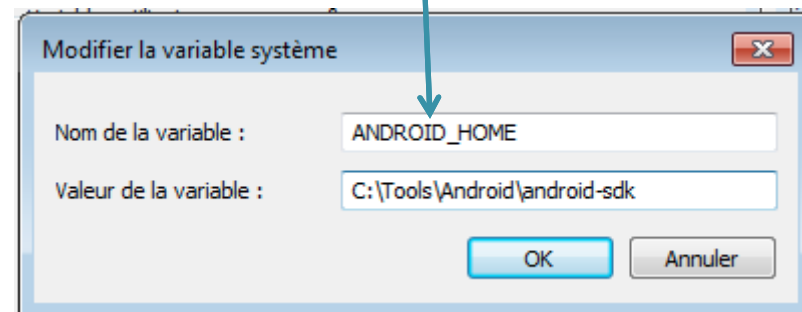
Le chemin de NodeJS devrait être déclaré dans la variable d'environnement Path

Installer Androïde SDK

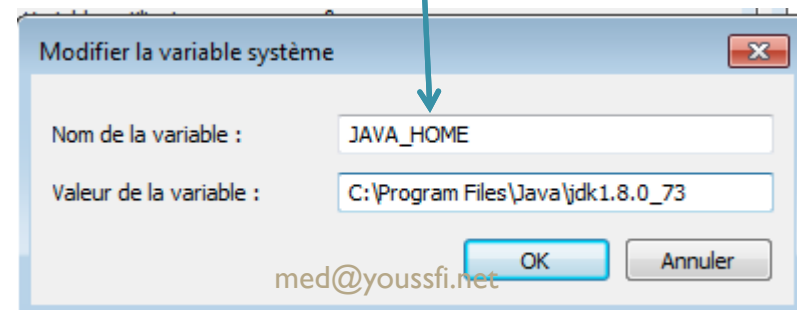
- Pour générer une application androïde, il faudrait installer Androïde SDK



Créer une variable d'environnement ANDROID_HOME Qui indique le chemin de android-sdk

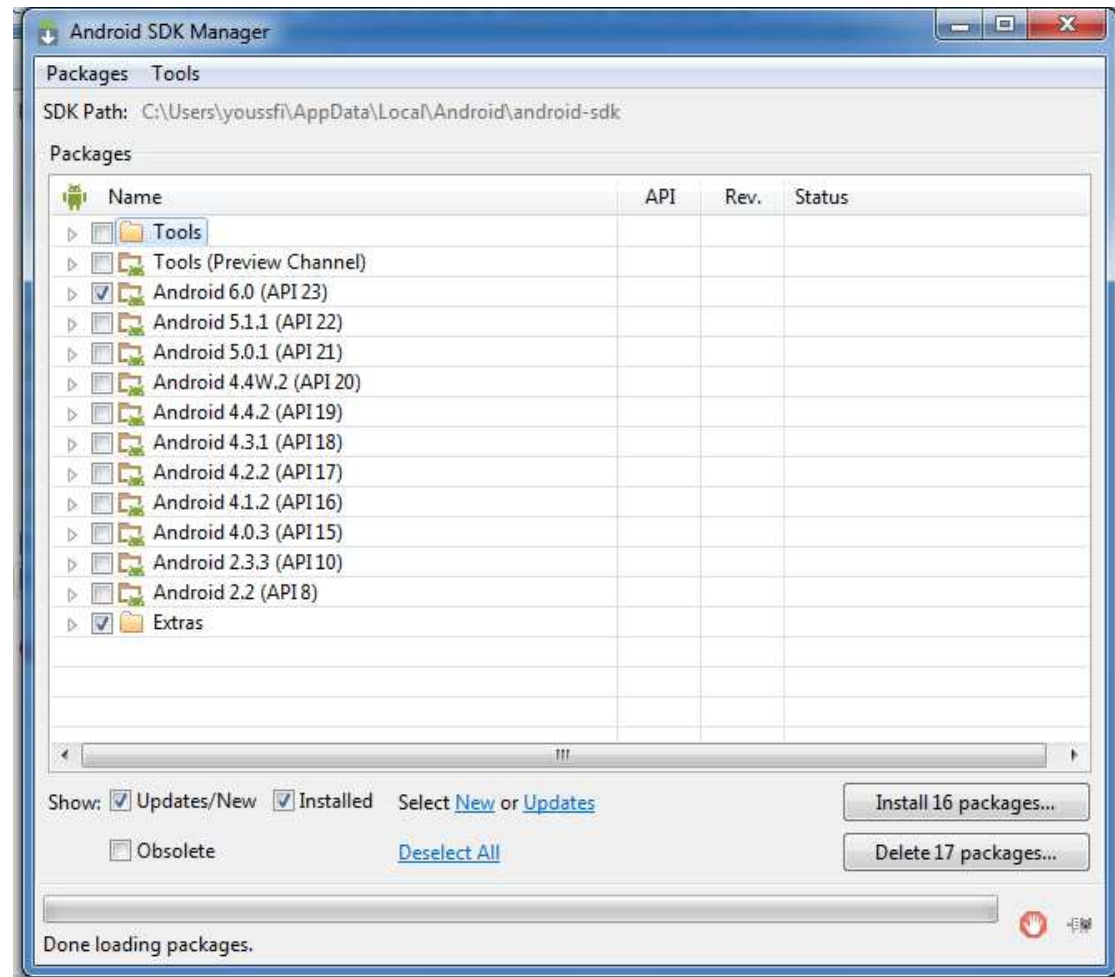


Créer une variable d'environnement JAVA_HOME Qui indique le chemin de du kit de développement java



SDK Manager

- Mettre à jour les API du SDK Android

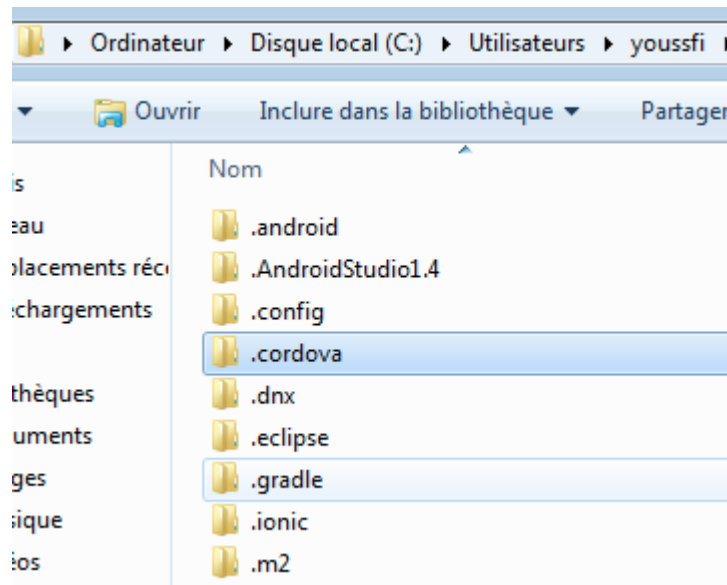


Cordova

- Apache Cordova est une plateforme pour construire des applications natives mobiles en utilisant HTML5, CSS et Java Script
- Suite au rachat de PhoneGap par **Adobe**, l'ensemble du **SDK cross-platform PhoneGap** a été rebaptisé Cordova et a été introduit dans l'incubateur de la fondation Apache.
- **Cordova** représente aujourd'hui **la meilleure solution de développement cross-platform** du marché.
- Elle permet avec peu d'efforts de développer une application mobile une fois et de la faire fonctionner sur toutes les plateformes mobiles du marché.
- **Les avantages de Cordova sont nombreux :**
 - Cordova est **OpenSource** (Licence Apache)
 - Cordova est basée sur les **standards du Web**.
 - Cordova supporte **la plupart des plateformes mobiles** du marché (Android, iOS, Blackberry, Windows Phone 7 ...).

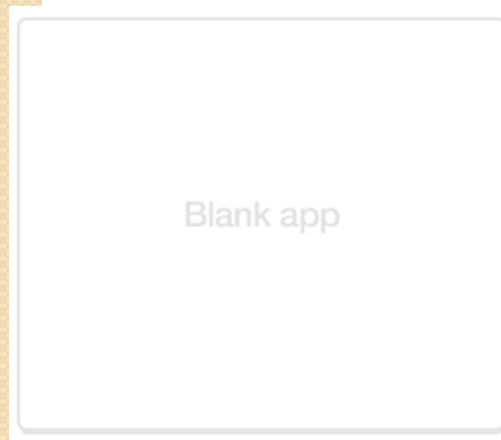
Installation de IONIC et cordova

- Les structures des applications de IONIC sont créées grâce à des commandes IONIC (Ionic CLI)
- Pour installer d'une manière globale IONIC et Cordova, exécuter la commande :
 - **npm install -g ionic cordova**



Première application IONIC

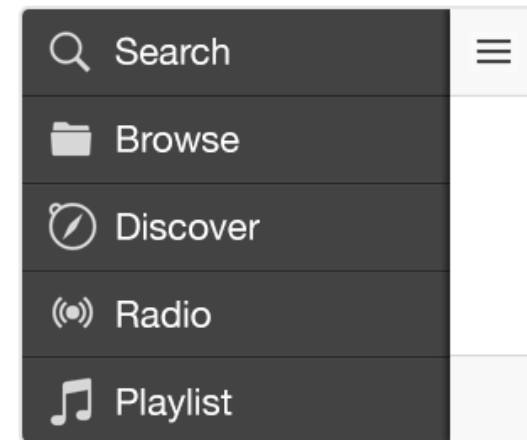
- Maintenant que l'on a un environnement de développement opérationnel, on va créer une application avec un menu sur le côté que l'on va appeler MySideMenuApp
- Ensuite, pour créer un nouveau projet, on saisit la commande :
 - `ionic start FirstApp sidemenu`



\$ ionic start myApp blank



\$ ionic start myApp tabs



\$ ionic start myApp sidemenu

```
C:\Windows\system32\cmd.exe

C:\WS\IONIC2>ionic start FirstApp sidemenu
V Creating directory .\FirstApp - done!
[INFO] Fetching app base
      (https://github.com/ionic-team/ionic2-app-base/archive/master.tar.gz)
V Downloading - done!
[INFO] Fetching starter template sidemenu
      (https://github.com/ionic-team/ionic2-starter-sidemenu/archive/master.tar
.gz)
V Downloading - done!
V Updating package.json with app details - done!
V Creating configuration file ionic.config.json - done!
[INFO] Installing dependencies may take several minutes!
V npm install - done!
V git init - done!
V git add -A - done!
V git commit -m "Initial commit" --no-gpg-sign - done!

? ♪ ? ♪ Your Ionic app is ready to go! ? ♪ ? ♪

Run your app in the browser (great for initial development):
  ionic serve

Run on a device or simulator:
  ionic cordova run ios

Test and share your app on a device with the Ionic View app:
  http://view.ionic.io

Next Steps:
Go to your newly created project: cd .\FirstApp

C:\WS\IONIC2>
```

Structure du projet IONIC

Dépendances externes du projets (Librairies Java Script et CSS)

Code source de votre applications : Composants web, services, modèle, etc..

Composant racine de l'application web

Ressources statiques de l'application mobile (images, icônes, etc.)

Pages de l'application mobile : chaque page est un composant web

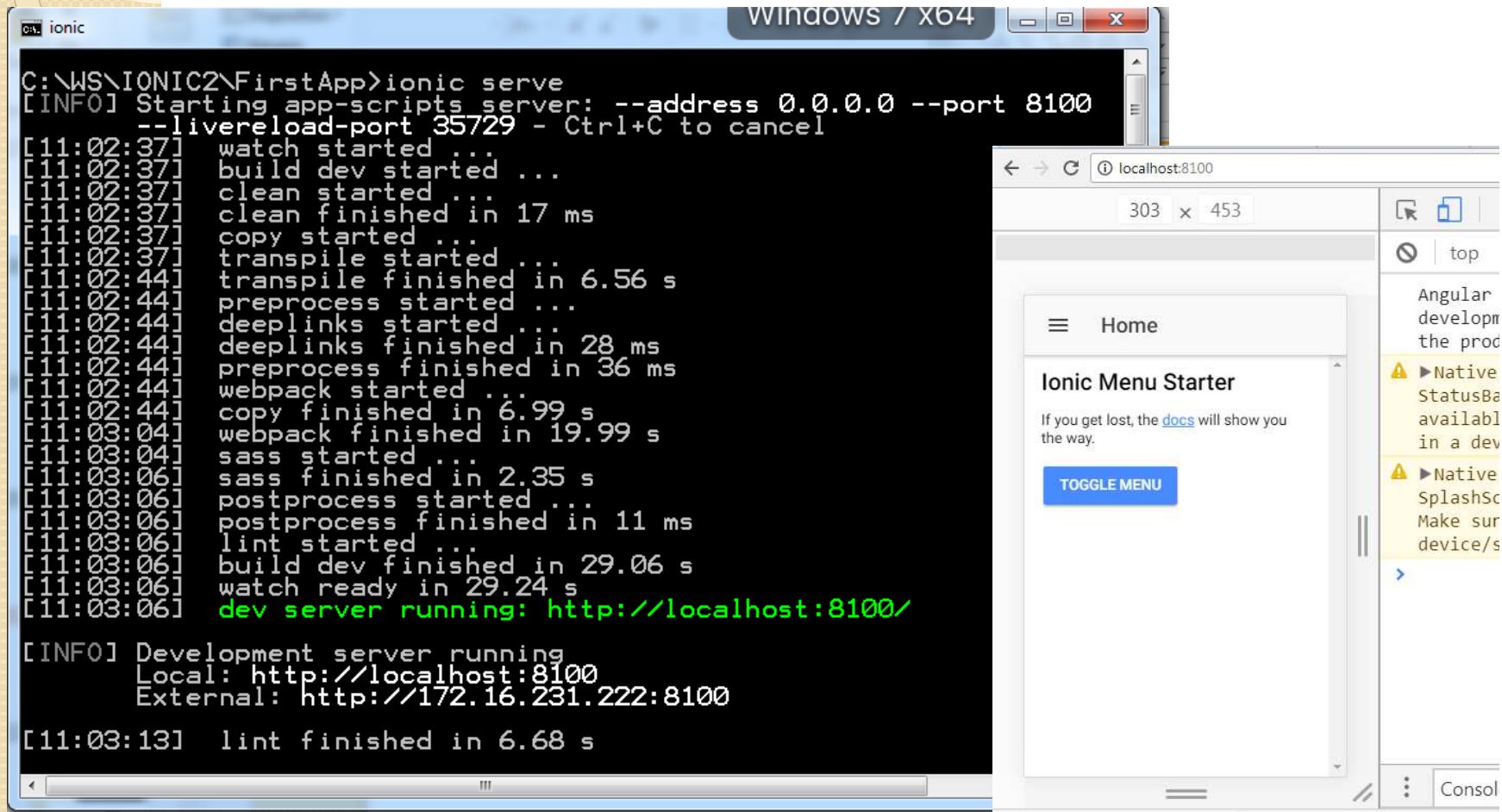
Page index.html : Page principale de l'application mobile

Fichier de configuration de l'application

fichier déclaratif des dépendances du projet

Visualiser l'application en utilisant un serveur local

- Pour lancer un serveur local pour tester l'application web, il faut lancer la commande :
 - **ionic** serve

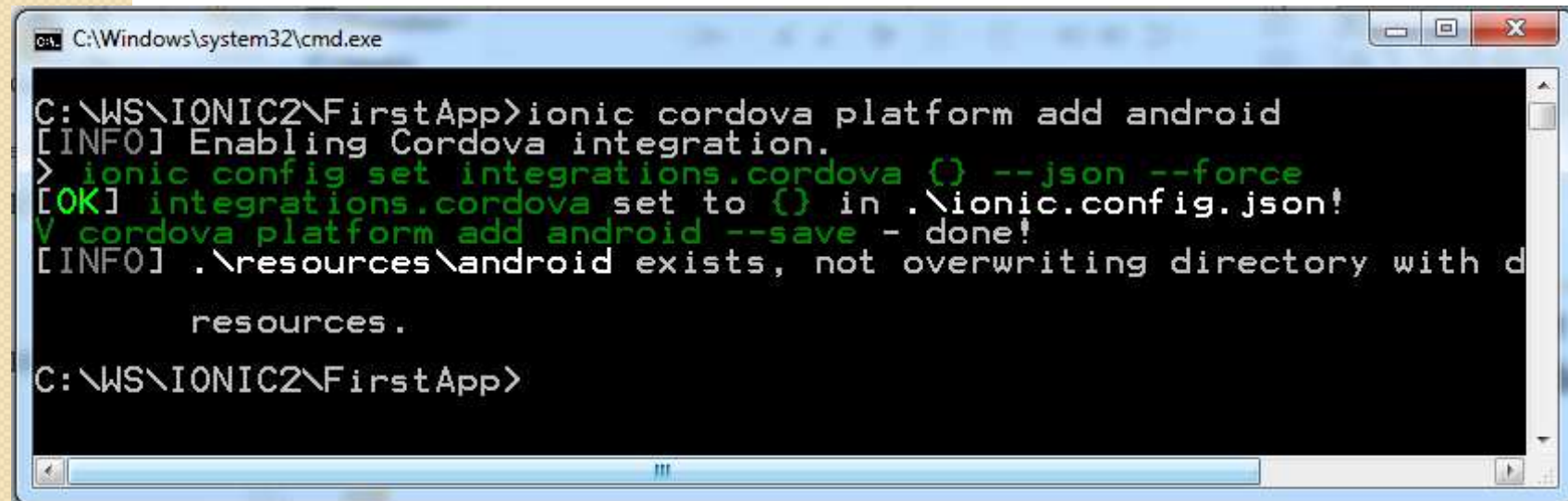


The image shows a Windows terminal window and a web browser side-by-side. The terminal window, titled 'ionic', shows the command 'ionic serve' being executed. It displays a series of status messages indicating the start of the development server, including 'Starting app-scripts server: --address 0.0.0.0 --port 8100', 'livereload-port 35729 - Ctrl+C to cancel', and various build steps like 'watch started', 'build dev started', 'clean started', 'copy started', 'transpile started', 'preprocess started', 'deeplinks started', 'webpack started', 'sass started', 'postprocess started', and 'lint started'. The terminal concludes with 'dev server running: http://localhost:8100/' in green text and 'Development server running' with local and external URLs.

The web browser window, titled 'localhost:8100', displays the application. It has a header 'Home' and a main section titled 'Ionic Menu Starter'. Below the title, it says 'If you get lost, the [docs](#) will show you the way.' and features a blue button labeled 'TOGGLE MENU'. The browser's address bar shows 'localhost:8100' and the page dimensions are 303 x 453. The browser's developer tools are open on the right, showing the 'Console' tab.

Générer une application android

- Pour pouvoir générer une application de type Android, il ne nous reste plus qu'à ajouter la plateforme avec la commande:
 - `ionic cordova platform add android`

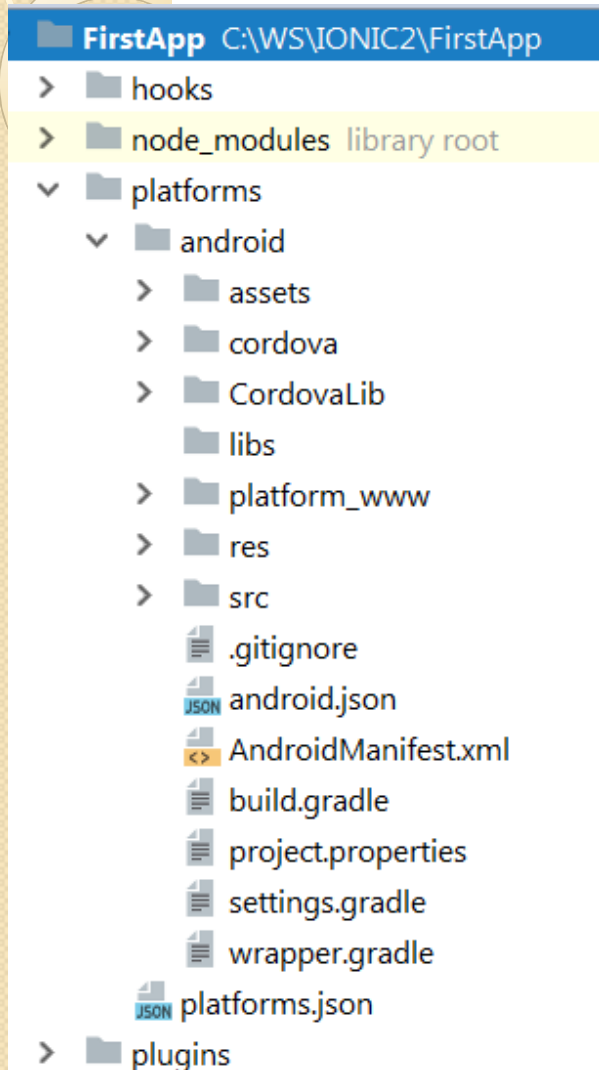


```
C:\Windows\system32\cmd.exe

C:\WS\IONIC2\FirstApp>ionic cordova platform add android
[INFO] Enabling Cordova integration.
> ionic config set integrations.cordova {} --json --force
[OK] integrations.cordova set to {} in .\ionic.config.json!
V cordova platform add android --save - done!
[INFO] .\resources\android exists, not overwriting directory with d
resources.

C:\WS\IONIC2\FirstApp>
```

Structure du projet Android généré



Fichier project.properties

target=android-25
android.library.reference.1=CordovaLib

Fichier AndroidManifest.xml

```
<?xml version='1.0' encoding='utf-8'?>
<manifest android:hardwareAccelerated="true" android:versionCode="1"
  android:versionName="0.0.1" package="io.ionic.starter"
  xmlns:android="http://schemas.android.com/apk/res/android">
  <supports-screens android:anyDensity="true" android:largeScreens="true"
    android:normalScreens="true" android:resizeable="true"
    android:smallScreens="true" android:xlargeScreens="true" />
  <uses-permission android:name="android.permission.INTERNET" />
  <application android:hardwareAccelerated="true"
    android:icon="@mipmap/icon" android:label="@string/app_name"
    android:supportsRtl="true">
    <activity
      android:configChanges="orientation|keyboardHidden|keyboard|screenSize|locale"
      android:label="@string/activity_name" android:launchMode="singleTop"
      android:name="MainActivity"
      android:theme="@android:style/Theme.DeviceDefault.NoActionBar"
      android:windowSoftInputMode="adjustResize">
      <intent-filter android:label="@string/launcher_name">
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
  <uses-sdk android:minSdkVersion="16" android:targetSdkVersion="25" />
</manifest>
```

Construire l'application android : .APK

- Il faudrait s'assurer si la version de l'api du android-sdk spécifiée dans le fichier `project.properties` est bien installée dans votre machine. Sinon, vous devriez spécifier votre version dans les deux fichiers:
 - `project.properties`
 - `AndroidManifest.xml`
- Pour construire l'application on utilise la commande :
 - **`ionic cordova build android`**

```
C:\WS\IONIC2\FirstApp>ionic cordova build android
```

```
Running app-scripts build:
```

```
[11:18:07] build dev started ...
```

```
[11:18:07] clean started ...
```

```
...
```

```
[11:18:35] build dev finished in 28.23 s
```

```
> ionic cordova prepare
```

```
V cordova prepare - done!
```

```
\ cordova build android [11:18:44] lint finished in 8.85 s
```

```
V cordova build android - done!
```

```
ANDROID_HOME=C:\Users\youssfi\AppData\Local\Android\sdk1
```

```
JAVA_HOME=C:\Tools\Java\jdk1.8.0_101
```

```
Starting a Gradle Daemon (subsequent builds will be faster)
```

```
:wrapper
```

```
...
```

```
BUILD SUCCESSFUL
```

```
...
```

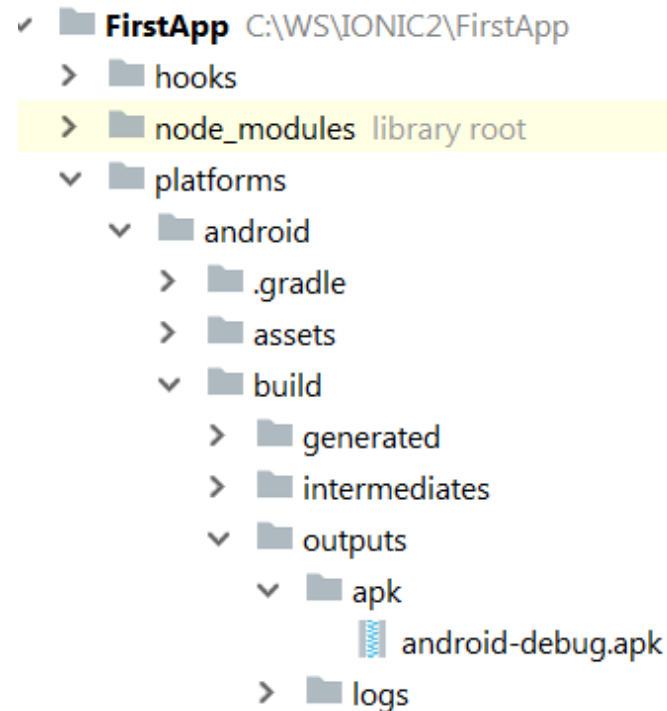
```
Total time: 20.243 secs
```

```
Built the following apk(s):
```

```
C:/WS/IONIC2/FirstApp/platforms/android/build/outputs/  
apk/android-debug.apk
```

Application android générée

- Grace à Cordova, ionic a générée l'application android-debug.apk
- Vous pouvez copier ce fichier dans votre téléphone Android en utilisant un câble USB ou encore envoyer le fichier par mail. puis l'installer.



Tester l'applications avec un émulateur.

- Vous avez besoin d'utiliser AVD Manager de android-sdk pour créer un modèle de téléphone android virtuel.
- Pour lancer l'application dans l'Android Virtual Device, vous pouvez utiliser la commande suivante :
 - **ionic cordova emulate android**

Edition du projet

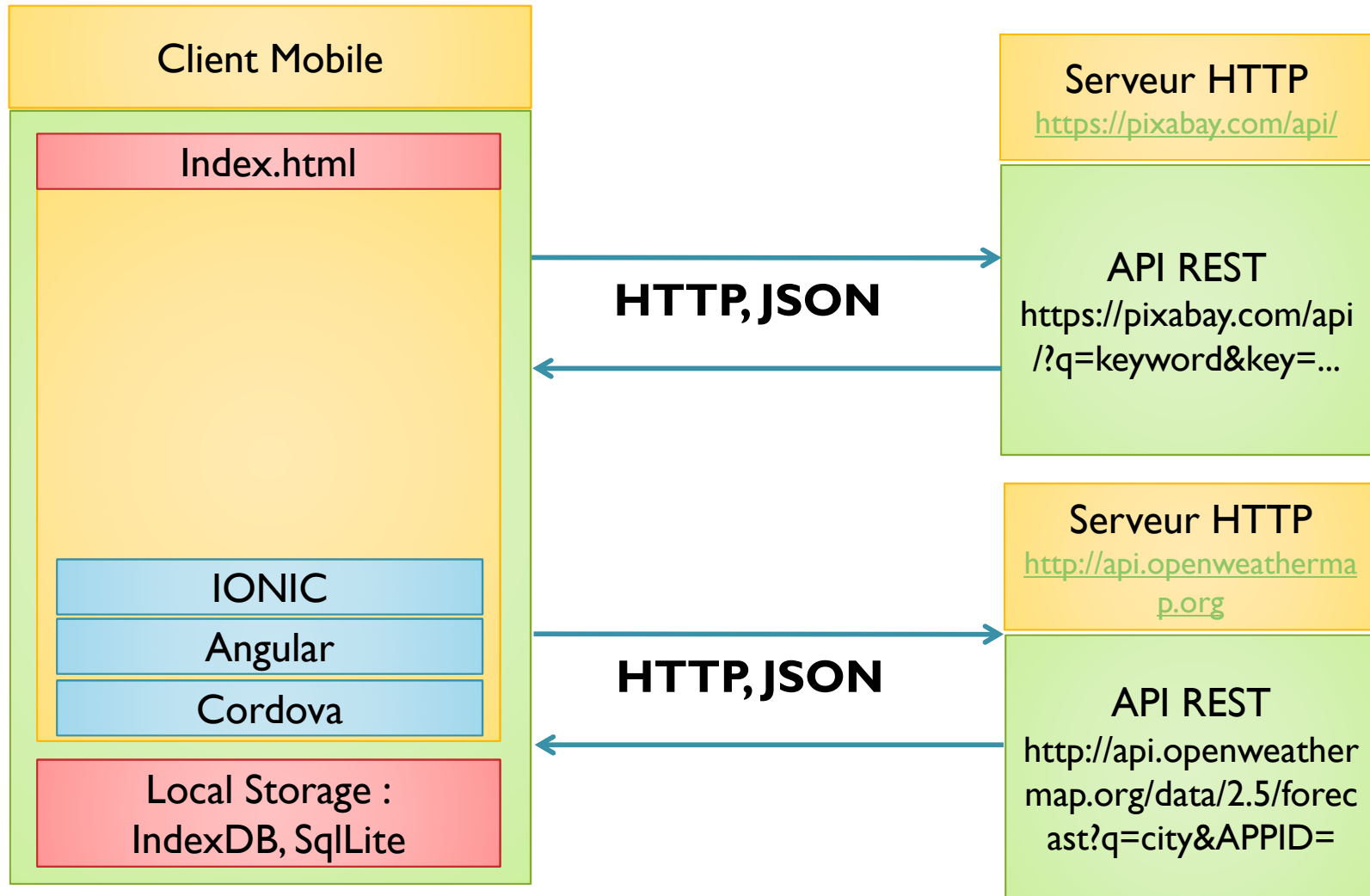
- Plusieurs IDE professionnels peuvent être utilisé pour éditer le code:
 - Web Storm, PHP Storm
 - Visual Studio Code
 - Eclipse avec plugin Angular
- D'autres éditeurs classiques peuvent être aussi utilisé :
 - Atom
 - Sublime Text
 - Etc ...

Application

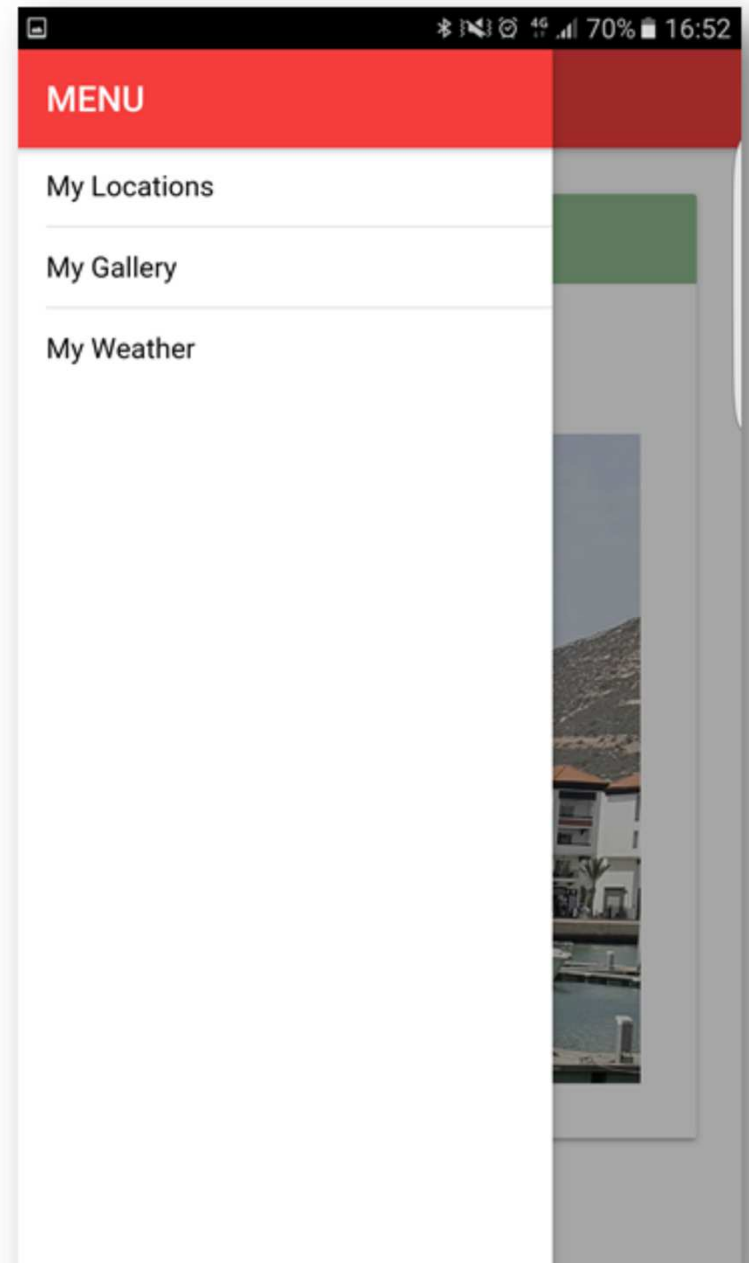
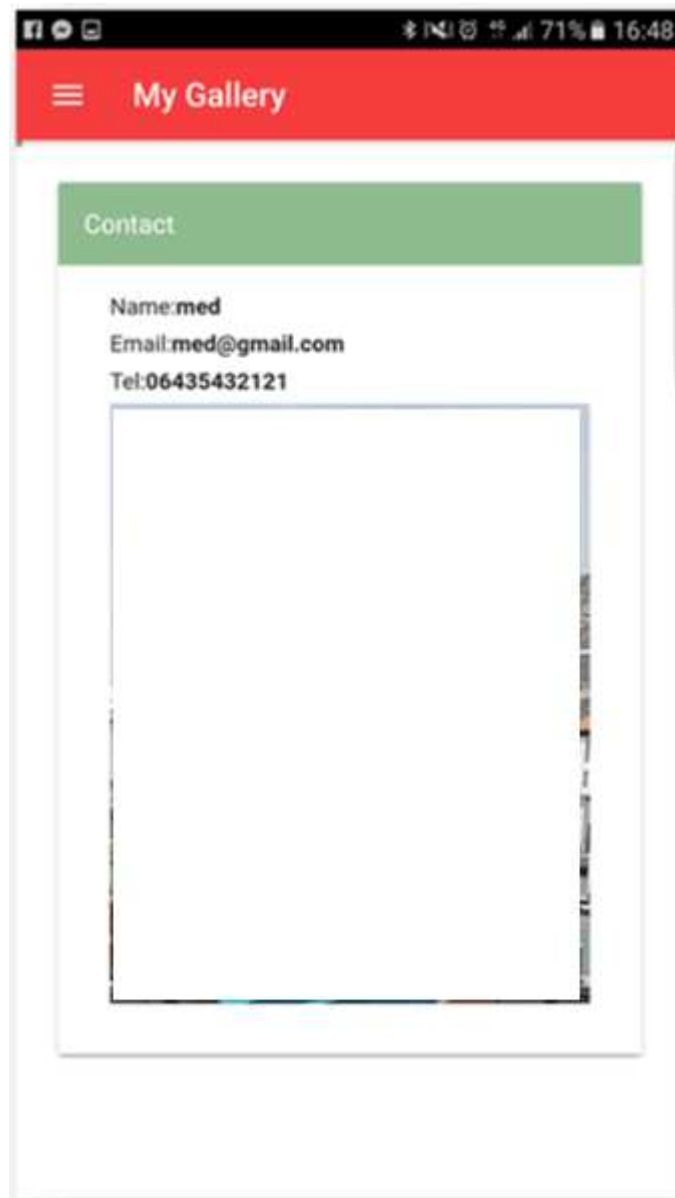
- Créer une application mobile qui permet de :
 - Rechercher et afficher des images exposée par l'API REST (Pixabay)
 - Afficher les données de la météo d'une ville donnée en faisant appel à l'API REST OpenWeather
 - Gérer une galerie de places (Restaurant, Monument, etc...)
 - Chaque place est définie par son titre, sa ville, son pays, l'instant de sa création, ses mots clés, ses coordonnées géographiques (latitude et longitude), une liste de photos
 - L'application permet de :
 - Saisir et ajouter des places dans un local Storage du mobile
 - Afficher et chercher la liste des places.
 - Sélectionner et supprimer des places.
 - Afficher et géo localiser (Google Map) une place sélectionnée
 - Prendre des photos d'une place donnée en utilisant la caméra ou la librairie locale.

◦

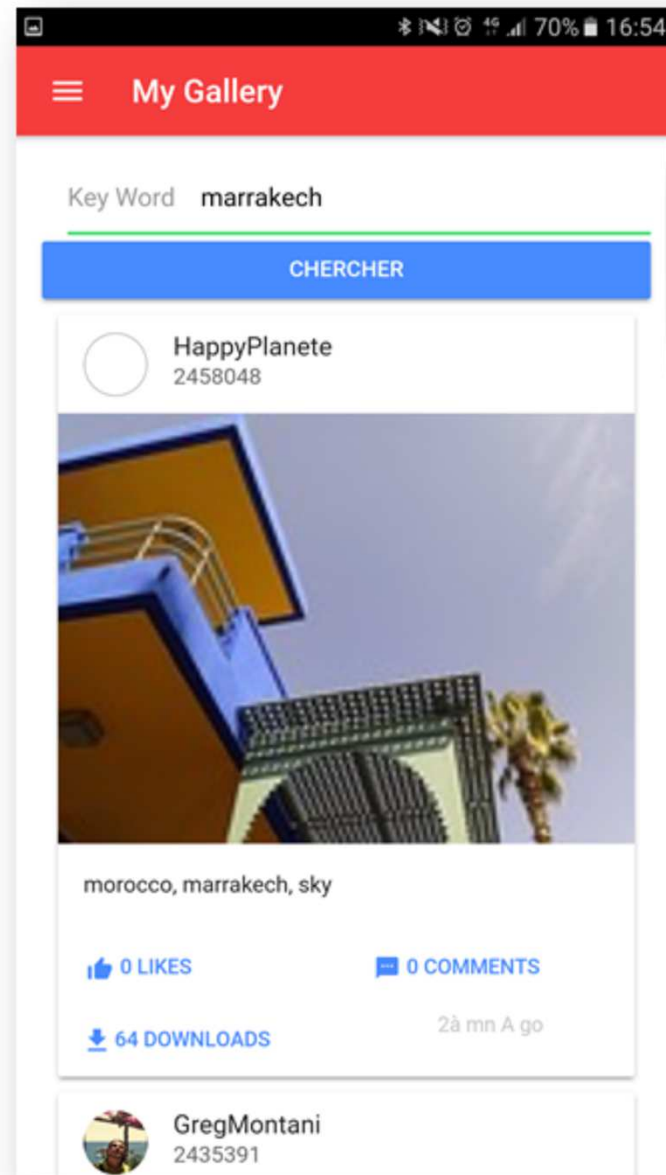
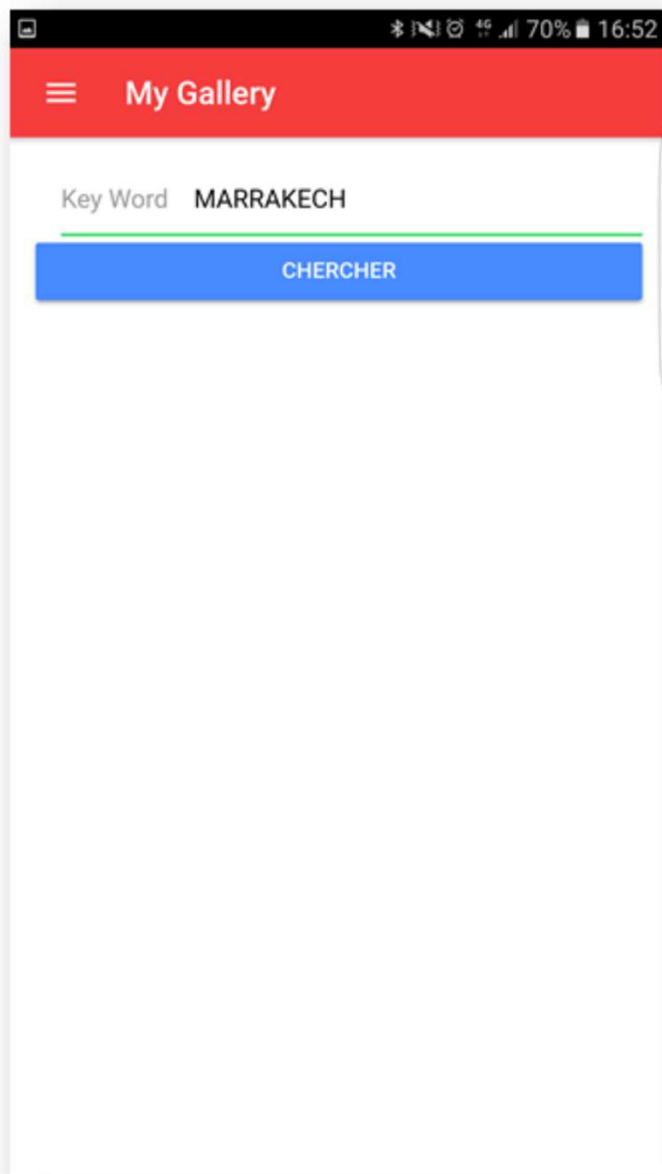
Architecture



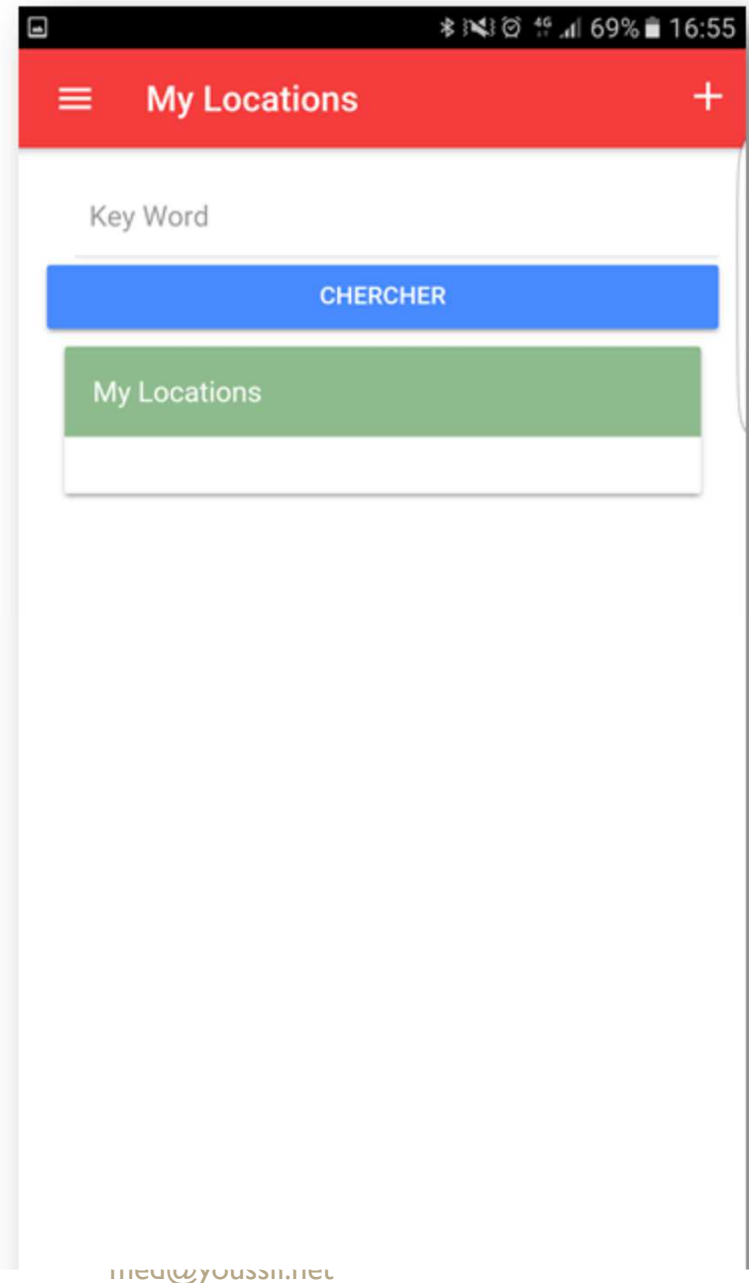
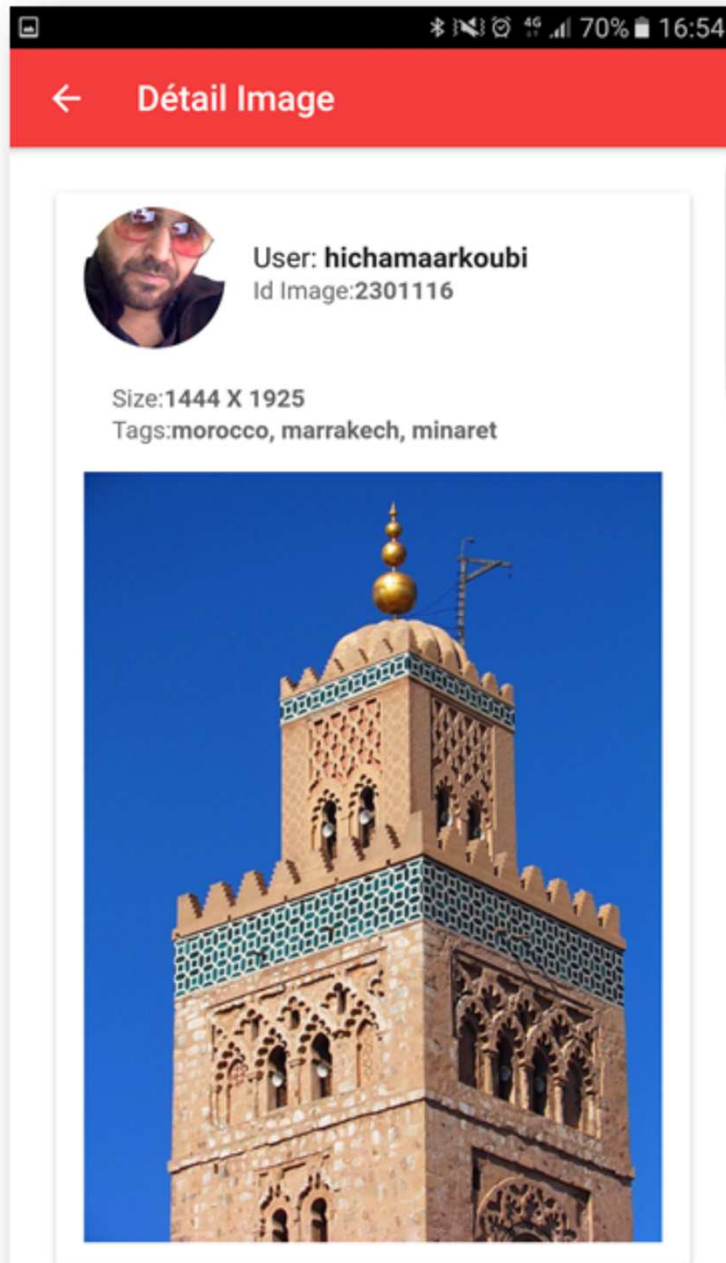
Ecrans de l'applications



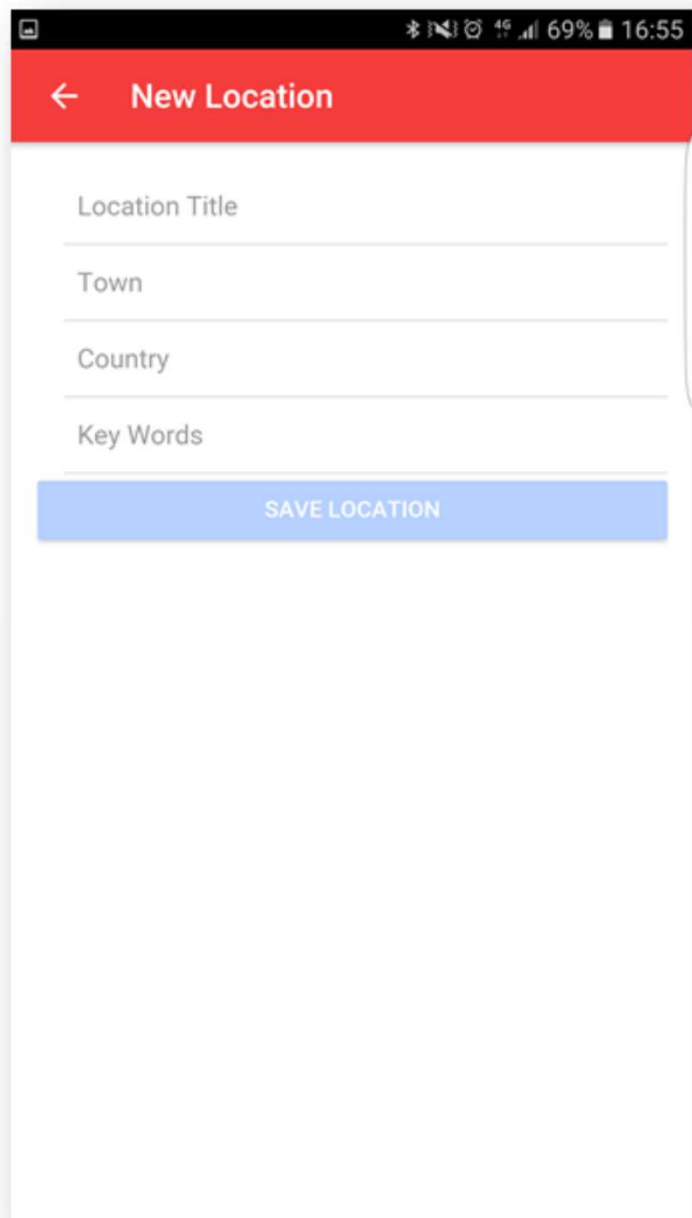
Ecrans de l'applications



Ecrans de l'applications



Ecrans de l'applications



This screenshot shows the 'New Location' form in an application. The form has a red header bar with a back arrow and the title 'New Location'. Below the header, there are four input fields: 'Location Title', 'Town', 'Country', and 'Key Words'. At the bottom of the form is a blue button labeled 'SAVE LOCATION'.

← New Location

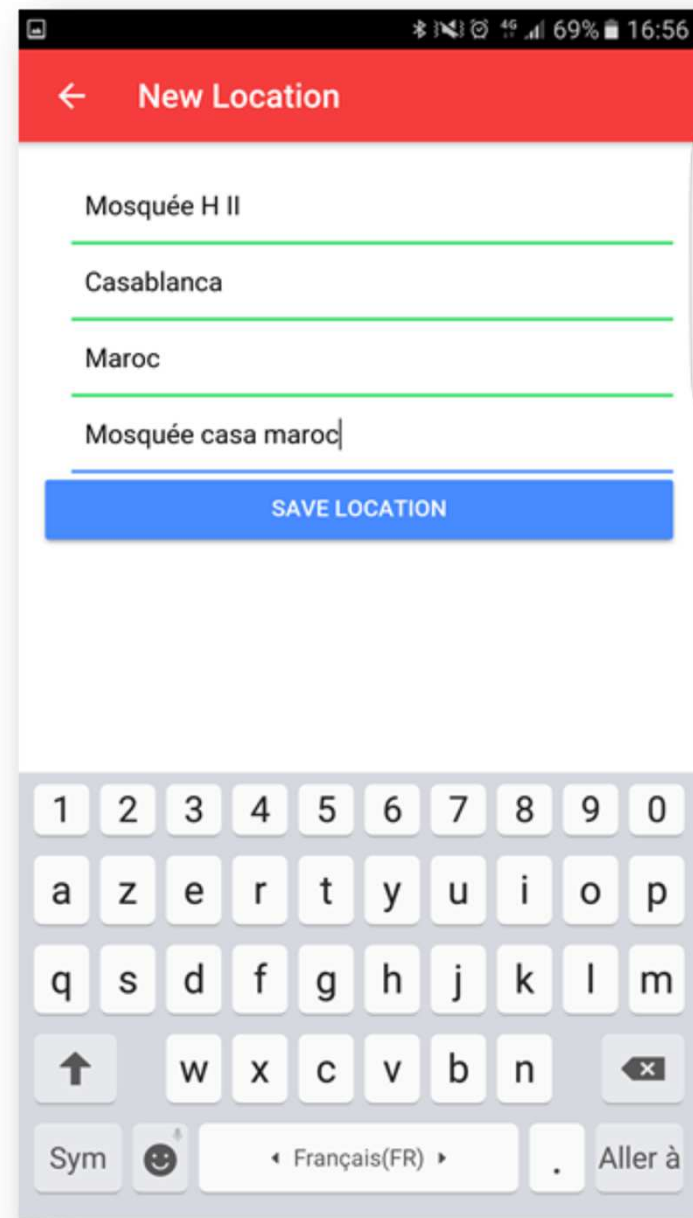
Location Title

Town

Country

Key Words

SAVE LOCATION



This screenshot shows the 'New Location' form with data entered in the first three fields: 'Mosquée H II' for Location Title, 'Casablanca' for Town, and 'Maroc' for Country. The 'Key Words' field contains 'Mosquée casa maroc'. A blue button labeled 'SAVE LOCATION' is at the bottom. A virtual keyboard is overlaid on the bottom half of the screen, showing the text 'Mosquée casa maroc' in the input field.

← New Location

Mosquée H II

Casablanca

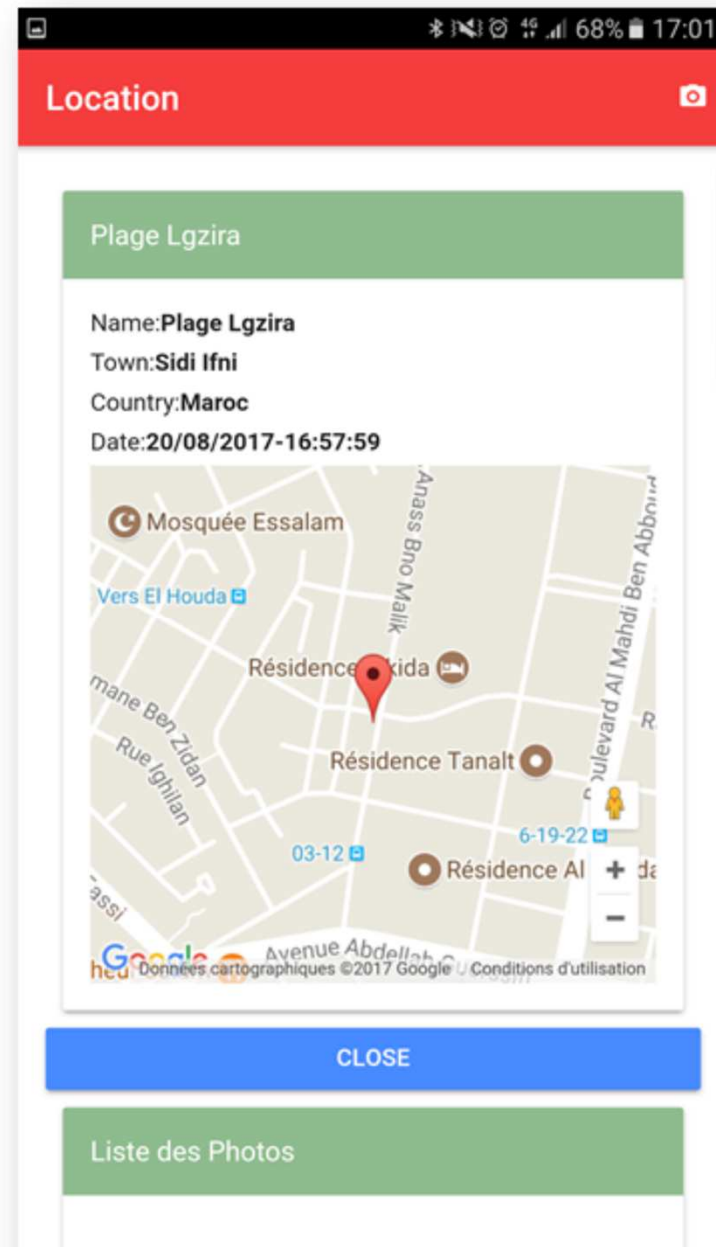
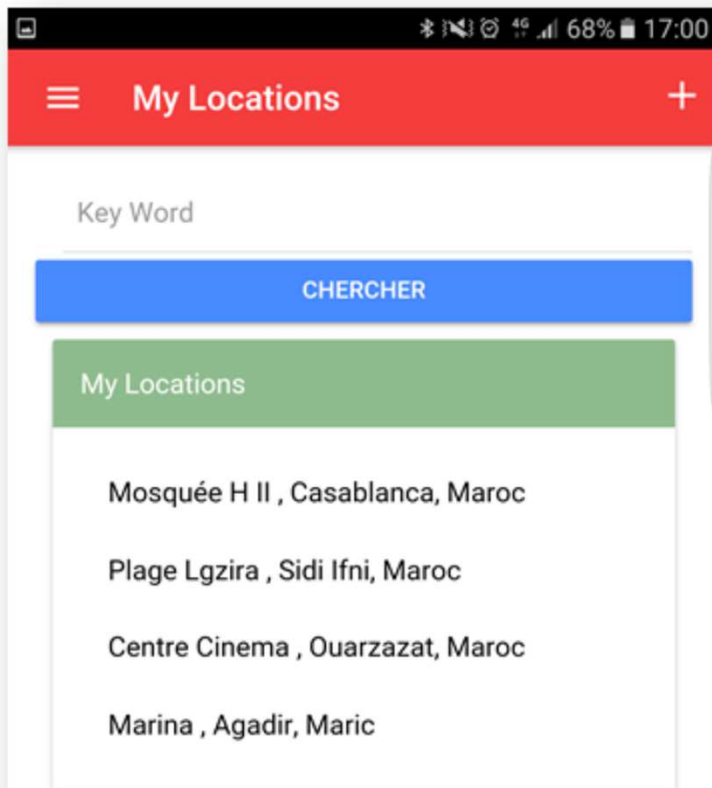
Maroc

Mosquée casa maroc

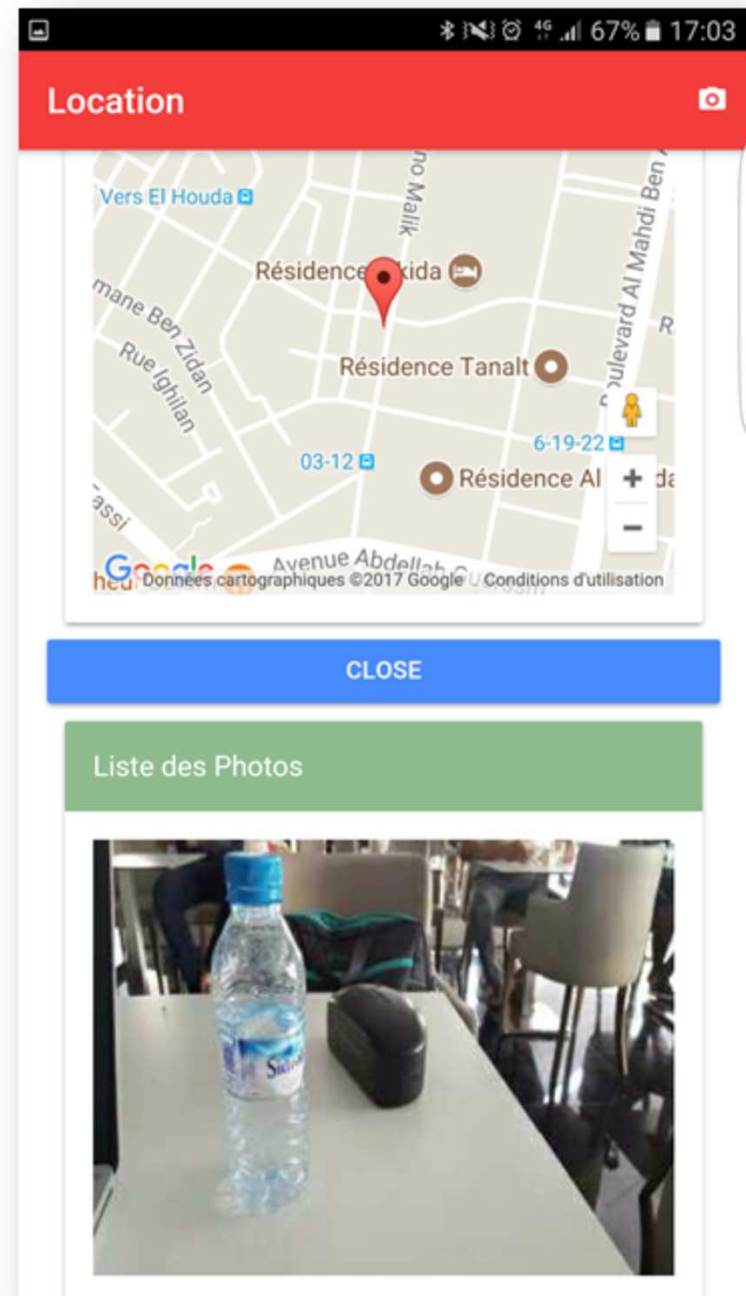
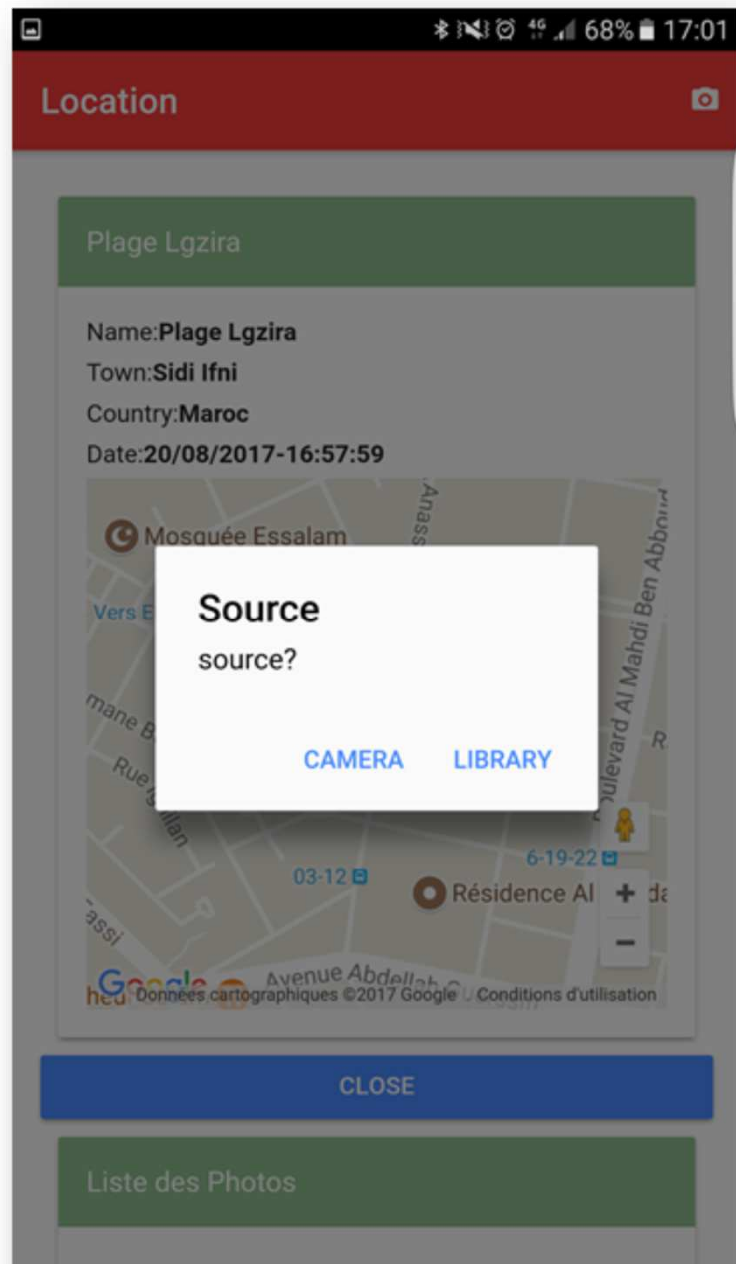
SAVE LOCATION

1 2 3 4 5 6 7 8 9 0
a z e r t y u i o p
q s d f g h j k l m
↑ w x c v b n ⌫
Sym 😊 ◀ Français(FR) ▶ . Aller à

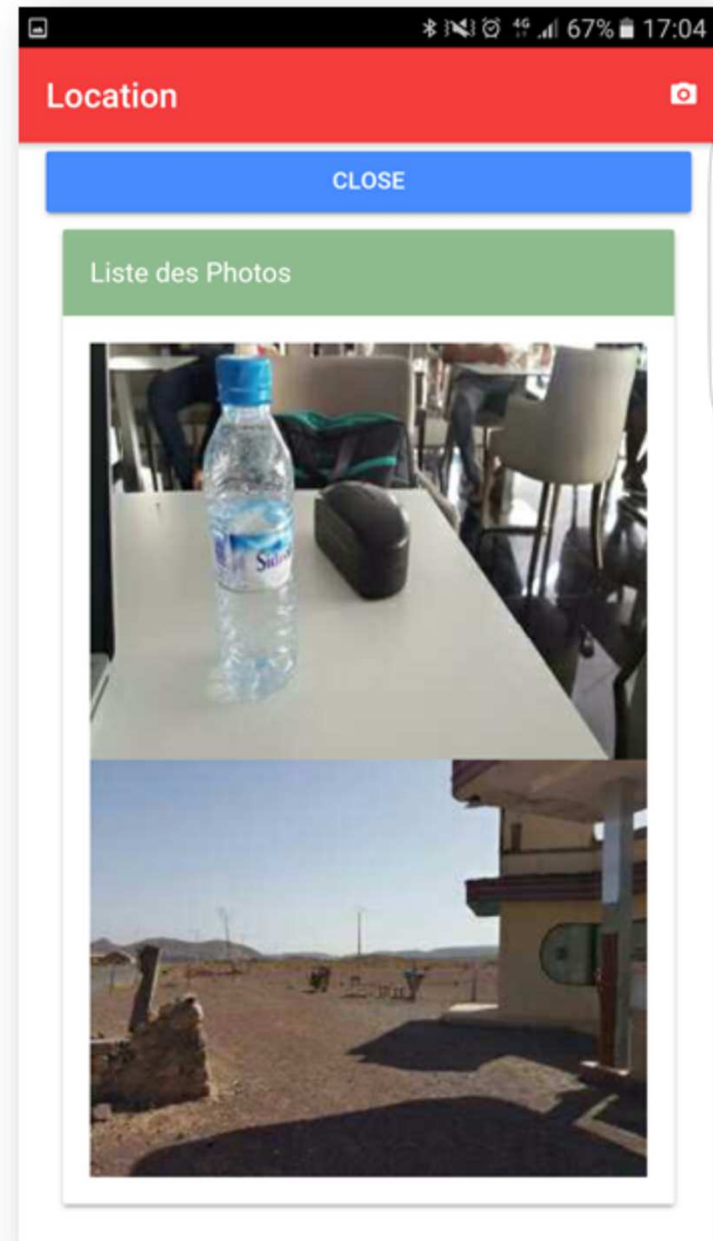
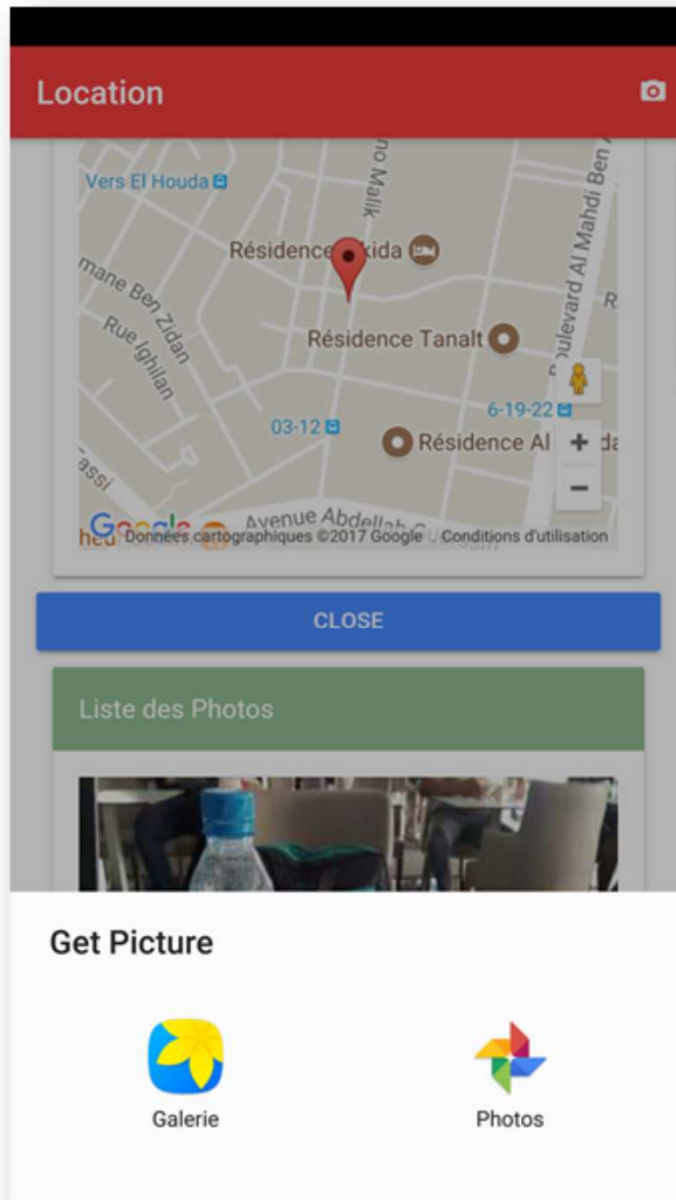
Ecrans de l'applications



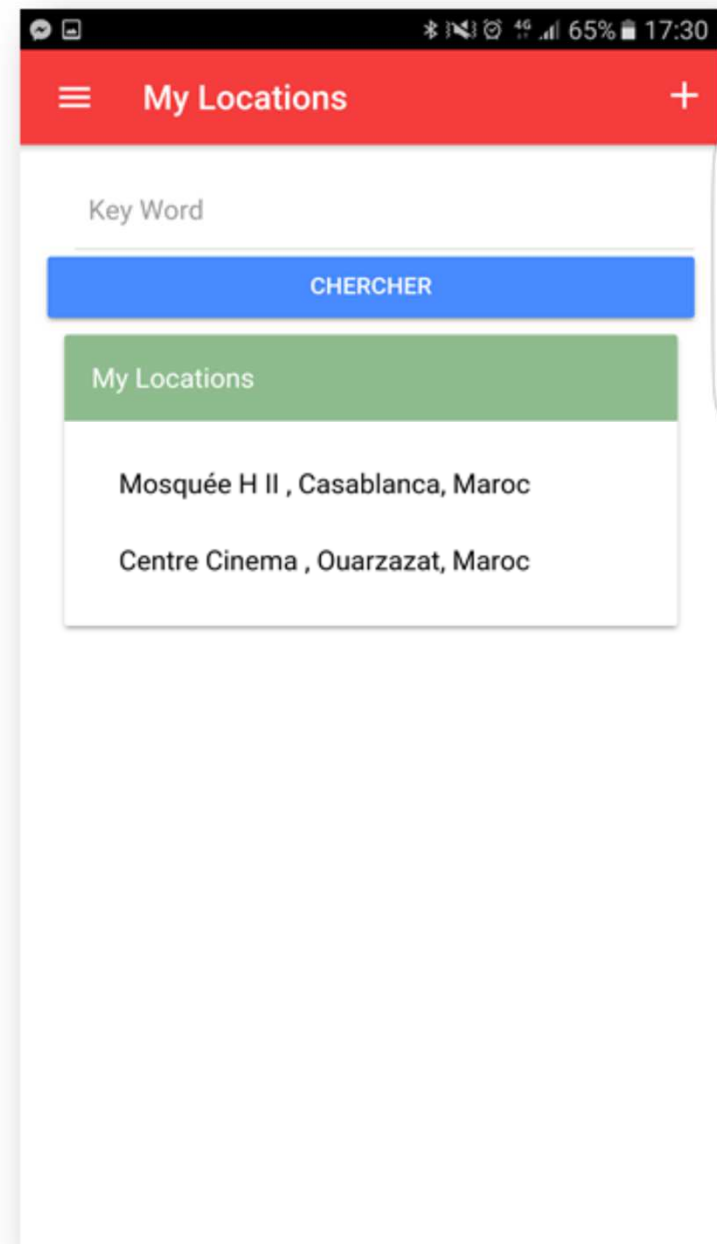
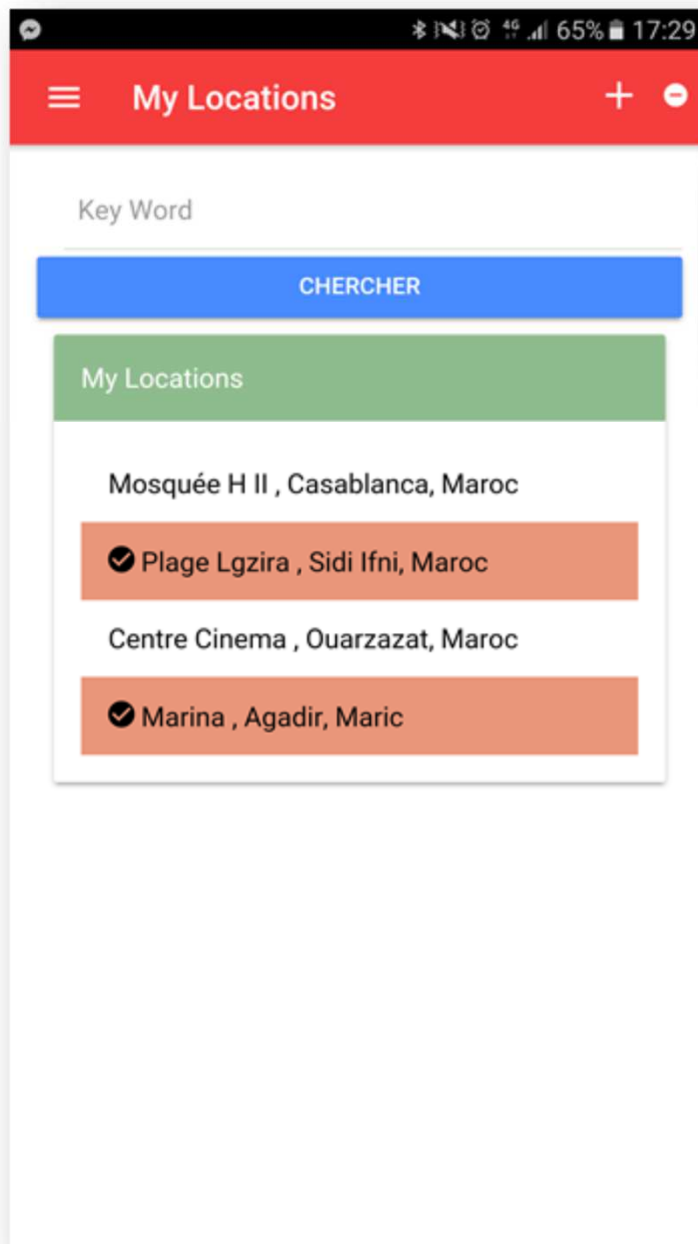
Ecrans de l'applications



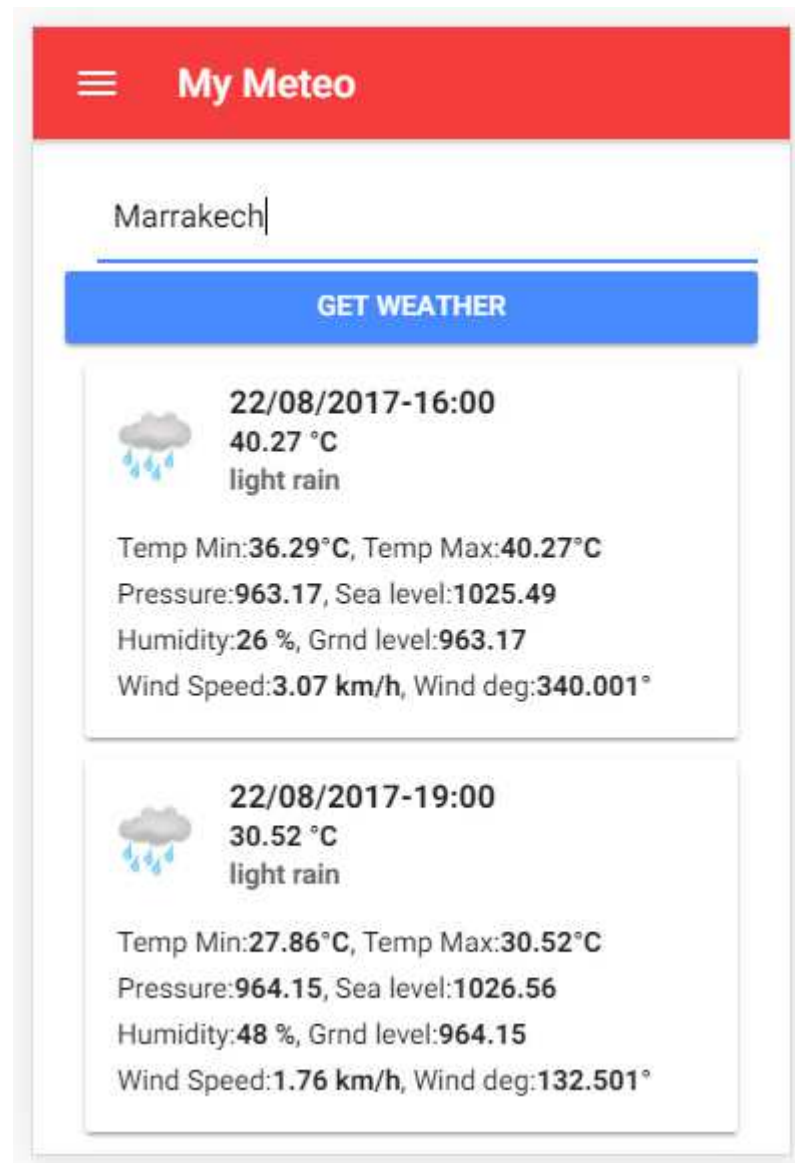
Ecrans de l'applications



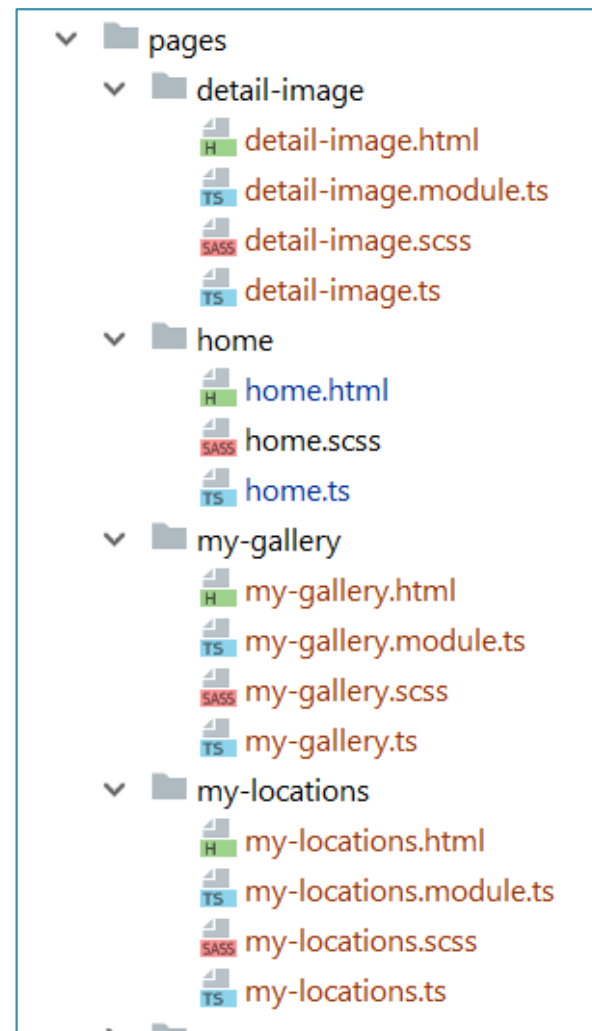
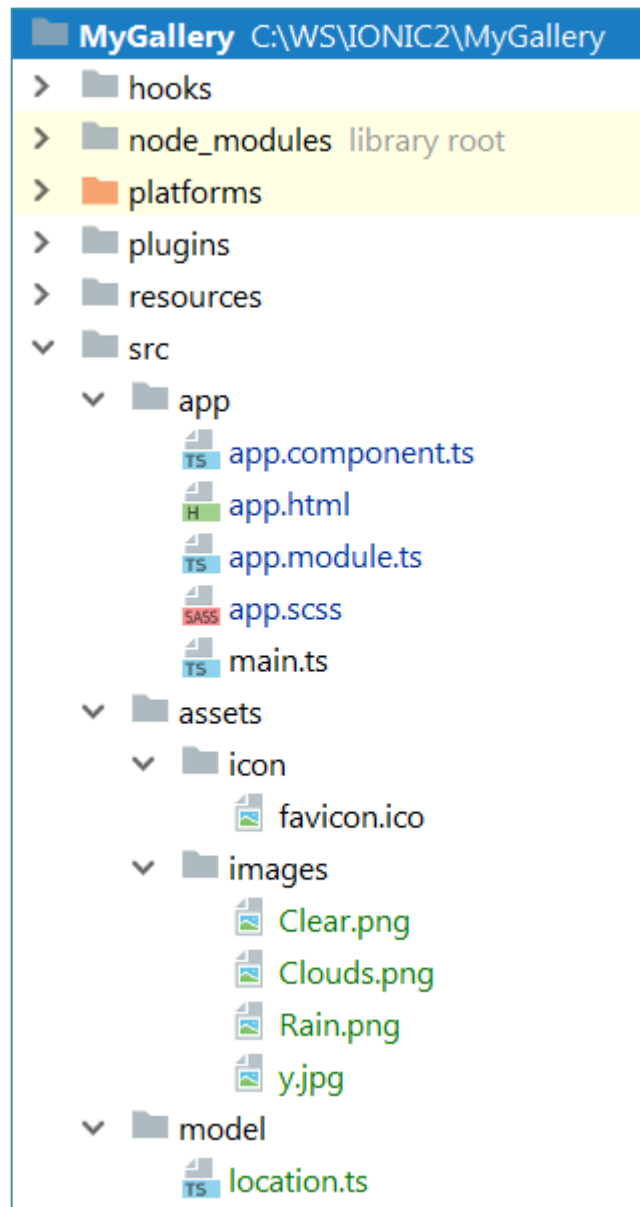
Ecrans de l'applications



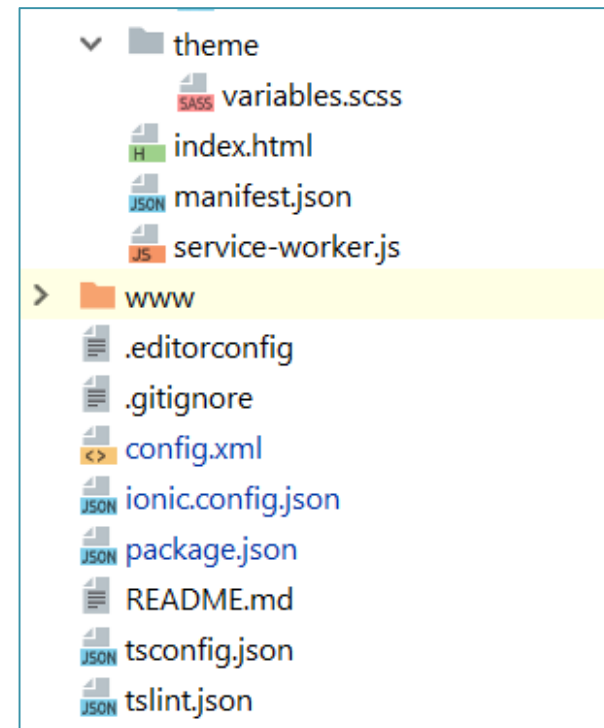
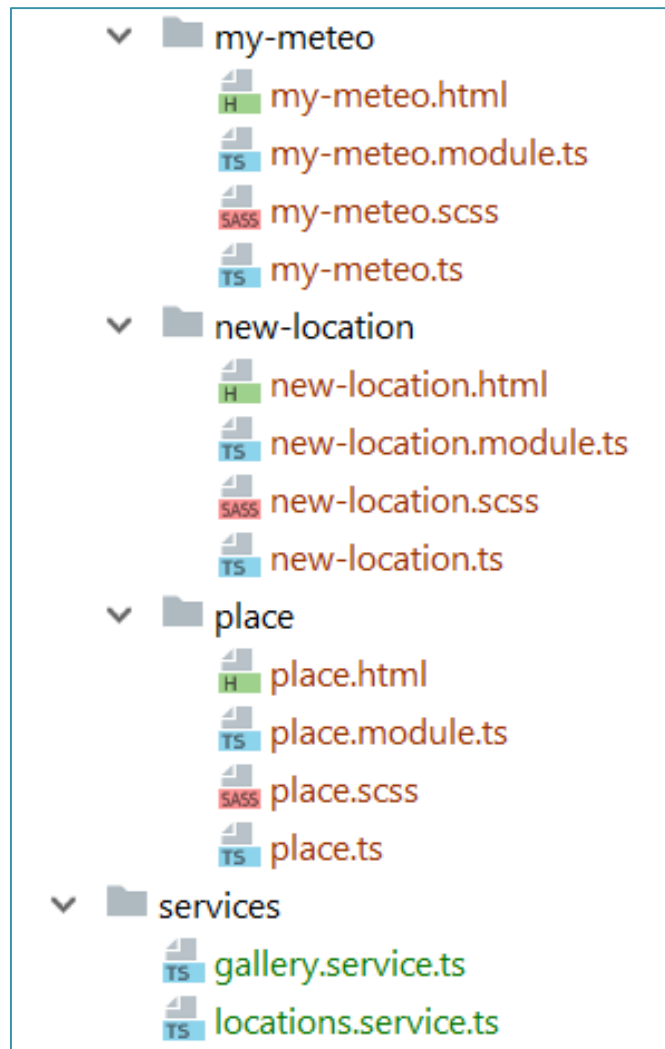
Ecrans de l'applications



Structure du Projet



Structure du Projet (Suite)



Index.html (Aucune modification)

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="UTF-8">
  <title>Ionic App</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0,
minimum-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <meta name="format-detection" content="telephone=no">
  <meta name="msapplication-tap-highlight" content="no">

  <link rel="icon" type="image/x-icon" href="assets/icon/favicon.ico">
  <link rel="manifest" href="manifest.json">
  <meta name="theme-color" content="#4e8ef7">
  <script src="cordova.js"></script>
  <link href="build/main.css" rel="stylesheet">
</head>
<body>
  <ion-app></ion-app>
  <script src="build/polyfills.js"></script>
  <script src="build/vendor.js"></script>
  <script src="build/main.js"></script>
</body>
</html>
```

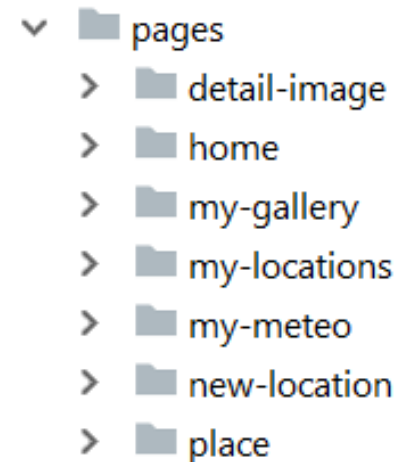
Index.html (Aucune modification)

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="UTF-8">
  <title>Ionic App</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0,
minimum-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <meta name="format-detection" content="telephone=no">
  <meta name="msapplication-tap-highlight" content="no">

  <link rel="icon" type="image/x-icon" href="assets/icon/favicon.ico">
  <link rel="manifest" href="manifest.json">
  <meta name="theme-color" content="#4e8ef7">
  <script src="cordova.js"></script>
  <link href="build/main.css" rel="stylesheet">
</head>
<body>
  <ion-app></ion-app>
  <script src="build/polyfills.js"></script>
  <script src="build/vendor.js"></script>
  <script src="build/main.js"></script>
</body>
</html>
```

Création des pages

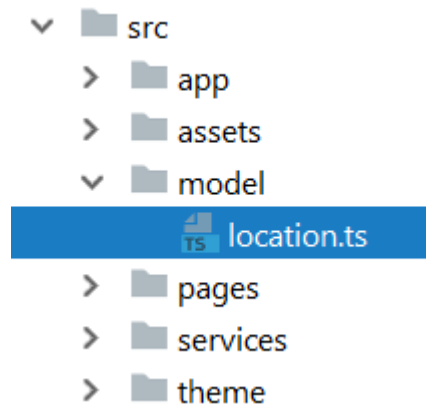
- Pour générer la structure d'une page avec IONIC CLI, on utilise la commande :
 - `ionic generate page nom-page`
- Dans notre cas, nous aurons besoin de générer les pages suivantes :
 - `ionic generate page my-locations`
 - `ionic generate page new-location`
 - `ionic generate page place`
 - `ionic generate page my-gallery`
 - `ionic generate page detail-image`
 - `ionic generate page my-meteo`
 - `ionic generate page my-locations`



Création du Modèle

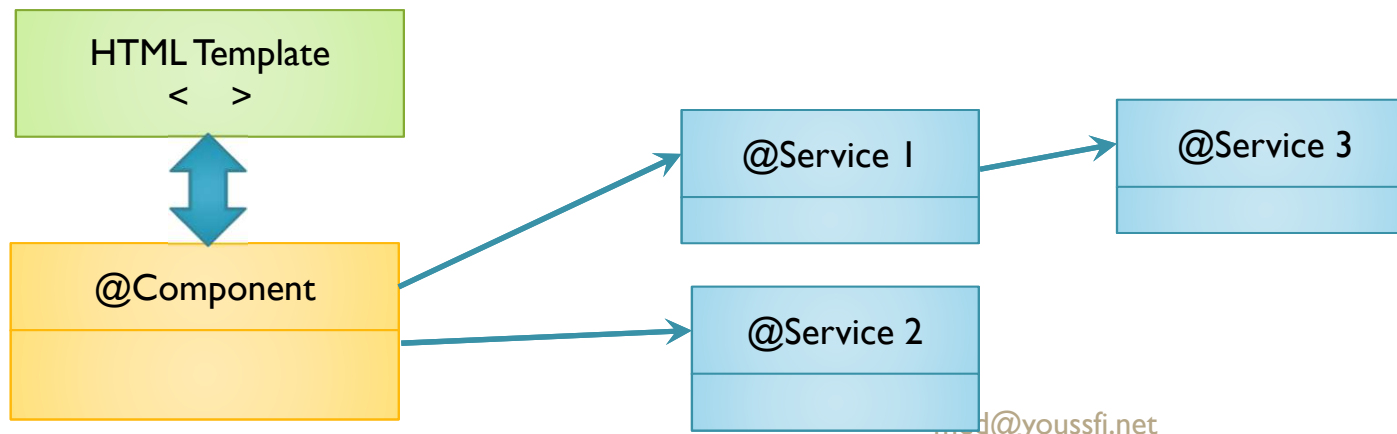
- Dans notre application, nous allons gérer des places. Nous aurons donc besoin de créer une classe nommée Place.
- Cette classe est déclarée dans un fichier TypeScript location.ts du dossier model.

```
export interface Place{  
  title:string,  
  town?:string,  
  country?:string,  
  keyWords?:string,  
  selected?:boolean,  
  timestamp:number,  
  coordinates?:{  
    latitude:number,  
    longitude:number  
  }  
  photos?:string[];  
}
```



Services

- Un service est une catégorie large qui englobe toute valeur, fonction ou fonctionnalité dont votre application a besoin.
- Un service est généralement une classe avec un but étroit et bien défini.
- Généralement, les composants se limite à l'affichage et à la gestion des événements utilisateurs dans la vue du composant. L'exécution des traitements en local ou en back end sont attribués aux services.
- Quand un événement survient dans la vue, le composant fait appel à des fonctions dans les services pour effectuer des traitements et fournir des résultats.
- Généralement, c'est les service qui interagissent avec la partie back end de l'application en envoyant des requêtes HTTP.
- Généralement c'est les composants qui consomme les services, toutefois, un service peut consommer d'autres services.
- l'utilisation d'un service se fait via le principe de l'injection des dépendances.



Exemple de service

exemple.service.ts

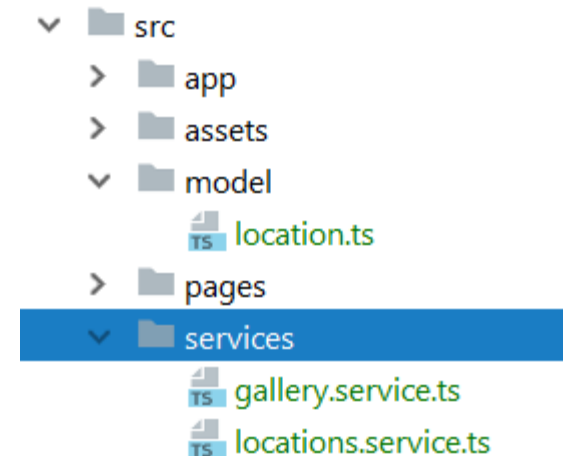
```
import { Injectable } from '@angular/core';
@Injectable()
export class ExempleService {
  constructor() { }
  saveData(data) {
    console.log('saving data at back end....');
  }
  getData() {
    console.log('gettig data from back end ...');
  }
}
```

exemple.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ExempleService } from "../exemple.service";
@Component({
  selector: 'app-exemple', templateUrl: './exemple.component.html',
  styleUrls: ['./exemple.component.css']
})
export class ExempleComponent {
  constructor(private exempleService: ExempleService) { }
  onSave(data) {
    this.exempleService.saveData(data);
  }
  onGetData() {
    return this.exempleService.getData();
  }
}
```

Création des services de l'application

- dans notre application, nous allons créer deux services:
 - Service de gestion des places dans une base de données locale du smartphone : Local storage (IndexedDB ou SQLite):
 - Ajouter, une place
 - Consulter toutes les places
 - Mettre à jour les places
 - Chercher des places
 - Service de recherche des images en faisant appel à l'API REST exposée sur <https://pixabay.com/api/>
 - Rechercher une liste d'image en envoyant les paramètres suivants:
 - key : la clé que vous pouvez récupérer en créant un compte sur le site pixabay. (voir <https://pixabay.com/api/docs/>)
 - q: représente le mot clé de la requête
 - per_page: Le nombre d'enregistrements par page à récupérer pour chaque requête
 - Page: le numéro de la page de données à récupérer
- Ces deux services sont créés dans le dossier services



Dépendances à installer

- Dans cette application, nous aurons besoin d'installer les dépendances suivantes:
 - Local Storage : pour accéder aux services de stockage local des smartphones :
 - First, if you'd like to use SQLite, install the cordova-sqlite-storage plugin:
 - `$ ionic cordova plugin add cordova-sqlite-storage`
 - Next, install the package
 - `$ npm install --save @ionic/storage`
 - Camera : pour prendre des photos en utilisant la caméra du smartphone et la galerie des images du smartphone :
 - `$ ionic cordova plugin add cordova-plugin-camera`
 - `$ npm install --save @ionic-native/camera`
 - Le composant Google Maps pour Angular 2
 - `$ npm install @agm/core --save`
 - Le composant IONIC Long Press pour sélectionner les éléments de la liste en appuyant longuement sur un élément de la liste.
 - `$ npm install --save ionic-long-press`

gallery.service.ts

gallery.service.ts

```
import {Injectable} from "@angular/core";
import {Http} from "@angular/http";

@Injectable()
export class GalleryService{

    constructor(private http:Http){}

    getImages(keyword :string, size:number, page:number) {
        return this.http.get('https://pixabay.com/api/?key=5832566-81dc7429a63c86e3b707d0429&q='+keyword+'&per_page='+size+'&page='+page)
            .map((resp)=> resp.json());
    }
}
```

location.service.ts

gallery.service.ts

```
import {Injectable} from "@angular/core";
import {Place} from "../model/location";
import {Storage} from "@ionic/storage";
@Injectable()
export class LocationsService{
  constructor(private stotage:Storage){ }
  locations:Array<Place>=[];
  addLocation(loc:Place){
    this.locations.push(loc);
    this.saveLocations();
  }
  getLocations(){
    return this.stotage.get('locations')
      .then((data)=>{
        this.locations=data!=null?data:[];
        return this.locations.slice();
      })
  }
  saveLocations(){
    try {
      this.stotage.set('locations',this.locations);
    }
    catch (e){
      console.log(e);
    }
  }
}
```


location.service.ts

gallery.service.ts

```
updateLocations(locations:Array<Place>){
  this.locations=locations;
  this.saveLocations();
}

searchForLocation(keyword:string) {
  let result:Array<Place>=[];
  this.locations.forEach(p=>{
    if(p.keyWords.indexOf(keyword)>=0)
      result.push(p);
  });
  return result;
}

addNewPhoto(photo:string,timestamp:number) {
  for(let i=0;i<this.locations.length;i++){
    if(this.locations[i].timestamp==timestamp){
      this.locations[i].photos.push(photo);
      break;
    }
  }
  this.saveLocations();
}
```

Déclaration des composants de l'application dans le module principale : **app/app.module.ts**

```
import { BrowserModule } from '@angular/platform-browser';
import { ErrorHandler, NgModule } from '@angular/core';
import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
import { SplashScreen } from '@ionic-native/splash-screen';
import { StatusBar } from '@ionic-native/status-bar';

import { MyApp } from './app.component';
import { HomePage } from '../pages/home/home';
import { MyLocationsPage } from '../pages/my-locations/my-locations';
import { MyGalleryPage } from '../pages/my-gallery/my-gallery';
import { MyMeteoPage } from '../pages/my-meteo/my-meteo';
import { NewLocationPage } from '../pages/new-location/new-location';
import { LocationsService } from '../services/locations.service';
import { IonicStorageModule } from '@ionic/storage';
import { Geolocation } from '@ionic-native/geolocation';
import { PlacePage } from '../pages/place/place';
import { AgmCoreModule } from '@agm/core';
import { LongPressModule } from 'ionic-long-press';
import { Camera } from '@ionic-native/camera';
import { HttpModule } from '@angular/http';
import { DetailImagePage } from '../pages/detail-image/detail-image';
import { GalleryService } from '../services/gallery.service';
```

Déclaration des composants de l'application dans le module principale : **app/app.module.ts**

```
@NgModule ({  
  declarations: [  
    MyApp,  
    HomePage,  
    MyLocationsPage,  
    MyGalleryPage,  
    MyMeteoPage,  
    NewLocationPage,  
    PlacePage,  
    DetailImagePage  
  ],
```

```
  imports: [  
    BrowserModule,  
    IonicModule.forRoot(MyApp),  
    LongPressModule,  
    HttpClientModule,  
    AgmCoreModule.forRoot({  
  
    apiKey: 'AIzaSyDJW9IkCVqZrMZF_f7NiWpfBH1tsB22g2Q'  
    })),  
    IonicStorageModule.forRoot({  
      name: '__mydb',  
      driverOrder: ['indexeddb', 'sqlite',  
        'websql']  
    })  
  ],  
  bootstrap: [IonicApp],
```

Déclaration des composants de l'application dans le module principale : **app/app.module.ts**

```
entryComponents: [  
  MyApp,  
  HomePage,  
  MyLocationsPage,  
  MyGalleryPage,  
  MyMeteoPage,  
  NewLocationPage,  
  PlacePage,  
  DetailImagePage  
],
```

```
providers: [  
  Geolocation,  
  StatusBar,  
  SplashScreen,  
  LocationsService,  
  GalleryService,  
  Camera,  
  {provide: ErrorHandler, useClass:  
    IonicErrorHandler}  
]  
}))  
export class AppModule {}
```

Root Component : `app/app.component.ts`

```
import {Component, ViewChild} from '@angular/core';
import {Nav, Platform} from 'ionic-angular';
import { StatusBar } from '@ionic-native/status-bar';
import { SplashScreen } from '@ionic-native/splash-screen';

import { HomePage } from '../pages/home/home';
import { MyLocationsPage } from "../pages/my-locations/my-locations";
import { MyGalleryPage } from "../pages/my-gallery/my-gallery";
import { MyMeteoPage } from "../pages/my-meteo/my-meteo";
@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  rootPage:any = HomePage;
  @ViewChild(Nav) nav:Nav;
  menu:Array<any>=[
    {titile:'My Locations', component:MyLocationsPage},
    {titile:'My Gallery', component:MyGalleryPage},
    {titile:'My Weather', component:MyMeteoPage}
  ];

  constructor(platform: Platform, statusBar: StatusBar, splashScreen: SplashScreen) {
    platform.ready().then(() => {
      statusBar.styleDefault();
      splashScreen.hide();
    });
  }
  onSelect(page) {
    this.nav.setRoot(page.component);
  }
}
```

Root Component : `app/app.html`

```
<ion-menu [content]="content">
  <ion-header>
    <ion-toolbar color="danger">
      <ion-title>MENU</ion-title>
    </ion-toolbar>
  </ion-header>
  <ion-content>
    <ion-list>
      <button *ngFor="let item of menu" ion-item menuClose
(click)="onSelect(item)">
        {{ item.title }}
      </button>
    </ion-list>
  </ion-content>
</ion-menu>
<ion-nav [root]="rootPage" #content></ion-nav>
```


Root Component : `app/app.scss`

```
.header-card1{  
  background-color: darkseagreen;  
  color: white;  
}  
.itemSelected{  
  background-color: darksalmon;  
}  
.circle-pic{  
  width:100px;  
  border-radius: 50%;  
}
```

Le Composant home (home page) : [pages/home/home.ts](#)

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

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {
  contact:any={
    name:"med",
    email:"med@gmail.com",
    tel:'06435432121',
    photo:'assets/images/y.jpg'
  }
  constructor(public navCtrl: NavController) {

  }

}
```

Le Composant home (home page) : <pages/home/home.html>

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>
      My Gallery
    </ion-title>
    <button ion-button menuToggle="">
      <ion-icon name="menu"></ion-icon>
    </button>
  </ion-navbar>
</ion-header>

<ion-content padding>
  <ion-card>
    <ion-card-header class="header-card1">Contact</ion-card-header>
    <ion-card-content>
      <div padding="">
        <p>Name:<strong>{{contact.name}}</strong></p>
        <p>Email:<strong>{{contact.email}}</strong></p>
        <p>Tel:<strong>{{contact.tel}}</strong></p>
        <p></p>
      </div>
    </ion-card-content>
  </ion-card>
</ion-content>
```

Partie : Gestion des places

- Cette partie de l'application mobile permet de gérer une galerie de places (Restaurant, Monument, etc...)
 - Chaque place est définie par son titre, sa ville, son pays, l'instant de sa création, ses mots clés, ses coordonnées géographiques (latitude et longitude), une liste de photos
 - L'application permet de :
 - Saisir et ajouter des places dans un local Storage du mobile
 - Afficher et chercher la liste des places.
 - Sélectionner et supprimer des places.
 - Afficher et géo localiser (Google Map) une place sélectionnée
 - Prendre des photos d'une place données en utilisant la caméra ou la librairie locale.
- Les données de cette partie de l'application sont stockées dans une base de données locale du smartphone (indexDb, sqlLite, etc.)

Le Composant MyLocationsPage : <pages/my-locations/my-locations.html>

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>My Locations</ion-title>
    <button ion-button menuToggle="">
      <ion-icon name="menu"></ion-icon>
    </button>
    <ion-buttons end="">
      <button
        ion-button icon-only=""
        (click)="onNewLocation()"
      >
        <ion-icon name="add"></ion-icon>
      </button>
      <button ion-button *ngIf="selectedPlaces.length>0"
        (click)="onDeleteSelectedPlaces()" >
        <ion-icon name="remove-circle" md="md-remove-circle"></ion-icon>
      </button>
    </ion-buttons>
  </ion-navbar>
</ion-header>
```

Le Composant MyLocationsPage : <pages/my-locations/my-locations.html>

```
<ion-content padding>
  <form #f="ngForm" (ngSubmit)="onSearchPlaces(f.value)">
    <ion-item>
      <ion-input type="text" name="keyWord" ngModel placeholder="Key Word"></ion-input>
    </ion-item>
    <button ion-button block type="submit">Chercher</button>
  </form>
  <ion-card>
    <ion-card-header class="header-card1">My Locations</ion-card-header>
    <ion-card-content padding>

      <ion-list>
        <button ion-item
          *ngFor="let p of locations"
          (click)="onOpenPlace(p)"
          [ngClass]="{'itemSelected':p.selected}"
          ion-long-press
          [interval]="400"
          (onPressStart)="pressed()"
          (onPressing)="active(p)"
          (onPressEnd)="released()"
        >
          <ion-icon name="checkmark-circle" *ngIf="p.selected"></ion-icon>
          {{p.title}} , {{p.town}}, {{p.country}}
        </button>
      </ion-list>
    </ion-card-content>
  </ion-card>
</ion-content>
```

Le Composant MyLocationsPage : `pages/my-locations/my-locations.ts`

```
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams, ModalController } from 'ionic-angular';
import { Place } from "../../model/location";
import { NewLocationPage } from "../../new-location/new-location";
import { LocationsService } from "../../services/locations.service";
import { PlacePage } from "../../place/place";

@IonicPage()
@Component({
  selector: 'page-my-locations',
  templateUrl: 'my-locations.html',
})
export class MyLocationsPage {

  locations: Array<Place>;
  press: boolean=false;
  selectedPlaces: Array<Place>=[];

  constructor(public navCtrl: NavController, public NavParams: NavParams,
               private locService: LocationsService,
               private modalCtrl: ModalController) {

  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad MyLocationsPage');
  }
}
```


Le Composant MyLocationsPage : [pages/my-locations/my-locations.ts](#)

```
ionViewWillEnter() {
  this.locService.getLocations().then((data) => {
    this.locations = data;
  });
}

onNewLocation() {
  this.navCtrl.push(NewLocationPage);
}

onOpenPlace(p) {
  console.log(p);
  this.modalCtrl.create(PlacePage, {place: p}).present();
}

pressed() {
  this.press = true;
  //console.log('pressed');
}

active(p: Place) {
  if (this.press) {
    p.selected = !p.selected;
    if (p.selected) {
      this.selectedPlaces.push(p);
    }
    else {
      this.selectedPlaces.splice(this.selectedPlaces.indexOf(p), 1);
    }
    this.press = false;
    //console.log("Active");
  }
}
```

Le Composant MyLocationsPage : <pages/my-locations/my-locations.ts>

```
released() {  
  this.press=false;  
  //console.log('Released');  
}  
onDeleteSelectedPlaces() {  
  for(let p of this.selectedPlaces) {  
    this.locations.splice(this.locations.indexOf(p),1);  
  }  
  this.selectedPlaces=[];  
  console.log(this.locations);  
  this.locService.updateLocations(this.locations);  
}  
onSearchPlaces(data) {  
  this.locations=this.locService.searchForLocation(data.keyWord);  
}  
}
```

Le Composant NewLocationPage : <pages/new-location/new-location.html>

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>New Location</ion-title>
    <button ion-button menuToggle="">
      <ion-icon name="menu"></ion-icon>
    </button>
  </ion-navbar>
</ion-header>
<ion-content padding>
  <form #f="ngForm" (ngSubmit)="onAddLocation(f.value)">
    <ion-item>
      <ion-input type="text" name="title" required ngModel placeholder="Location
Title"></ion-input>
    </ion-item>
    <ion-item>
      <ion-input type="text" name="town" ngModel placeholder="Town"></ion-input>
    </ion-item>
    <ion-item>
      <ion-input type="text" name="country" ngModel placeholder="Country"></ion-
input>
    </ion-item>
    <ion-item>
      <ion-input type="text" name="keyWords" ngModel placeholder="Key
Words"></ion-input>
    </ion-item>
    <button type="submit" ion-button block [disabled]="!f.valid">Save
Location</button>
  </form>
</ion-content>
```

Le Composant NewLocationPage : <pages/new-location/new-location.ts>

```
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams } from 'ionic-angular';
import { LocationsService } from "../../services/locations.service";
import { Geolocation } from "@ionic-native/geolocation";
import { Place } from "../../model/location";
@IonicPage()
@Component({
  selector: 'page-new-location',
  templateUrl: 'new-location.html',
})
export class NewLocationPage {

  constructor(public navCtrl: NavController, public navParams:
NavParams,
               private locService:LocationsService,
               private geoLocation:Geolocation) {

  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad NewLocationPage');
  }
}
```

Le Composant NewLocationPage : <pages/new-location/new-location.ts>

```
onAddLocation(data) {
  this.geoLocation.getCurrentPosition()
    .then((position)=>{
      let place:Place={
        title:data.title, town:data.town, country:data.country,
        keywords:data.keywords, timestamp:new Date().getTime(),
        coordinates:{
          latitude:position.coords.latitude,
          longitude:position.coords.longitude
        },
        photos:[]
      };
      this.locService.addLocation(place);
      this.navCtrl.pop();
    })
    .catch((err)=>{
      let place:Place={
        title:data.title, town:data.town, country:data.country,
        keywords:data.keywords, timestamp:new Date().getTime(),
        coordinates:{
          latitude:0, longitude:0
        },
        photos:[]
      };
      this.locService.addLocation(place);
      this.navCtrl.pop();
      console.log(err);
    });
}
```

Le Composant PlacePage : `pages/place/place.html`

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>Location</ion-title>
    <ion-buttons end>
      <button ion-button (click)="getPicture()">
        <ion-icon name="camera"></ion-icon>
      </button>
    </ion-buttons>
  </ion-navbar>
</ion-header>
<ion-content padding>
  <ion-card>
    <ion-card-header class="header-card1">{{place.title}}</ion-card-header>
    <ion-card-content padding="">
      <p>Name:<strong>{{place.title}}</strong></p>
      <p>Town:<strong>{{place.town}}</strong></p>
      <p>Country:<strong>{{place.country}}</strong></p>
      <p>Date:<strong>{{place.timestamp|date:'dd/MM/yyyy-HH:mm:ss'}}</strong></p>
      <agm-map [latitude]="place.coordinates.latitude"
[longitude]="place.coordinates.longitude" [zoom]="16">
        <agm-marker [latitude]="place.coordinates.latitude"
[longitude]="place.coordinates.longitude"></agm-marker>
      </agm-map>
    </ion-card-content>
  </ion-card>
  <button ion-button block (click)="close()">Close</button>
  <ion-card>
    <ion-card-header class="header-card1">Liste des Photos</ion-card-header>
    <ion-card-content padding>
      <div *ngFor="let im of place.photos">
        
      </div>
    </ion-card-content>
  </ion-card>
</ion-content>
```

Le Composant PlacePage : `pages/place/place.ts`

```
import { Component } from '@angular/core';
import { AlertController, IonicPage, NavController, NavParams, ViewController }
from 'ionic-angular';
import { Place } from "../../model/location";
import { Camera, CameraOptions } from "@ionic-native/camera";
import { LocationsService } from "../../services/locations.service";

@IonicPage()
@Component({
  selector: 'page-place',
  templateUrl: 'place.html',
})
export class PlacePage {

  place: Place;

  constructor(public navCtrl: NavController,
               public NavParams: NavParams,
               private viewCtrl: ViewController,
               private camera: Camera,
               private alertController: AlertController,
               private locService: LocationsService) {
    this.place = NavParams.get('place');
  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad PlacePage');
  }
}
```


Le Composant PlacePage : `pages/place/place.ts`

```
close() {
  this.viewController.dismiss();
}
getPicture() {
  const options1: CameraOptions = {
    quality: 50,
    destinationType: this.camera.DestinationType.DATA_URL,
    encodingType: this.camera.EncodingType.JPEG,
    mediaType: this.camera.MediaType.PICTURE,
    sourceType: this.camera.PictureSourceType.CAMERA,
    allowEdit: true,
    targetWidth: 320,
    targetHeight: 240
  };
  const options2: CameraOptions = {
    quality: 50,
    destinationType: this.camera.DestinationType.DATA_URL,
    encodingType: this.camera.EncodingType.JPEG,
    mediaType: this.camera.MediaType.PICTURE,
    sourceType: this.camera.PictureSourceType.PHOTOLIBRARY,
    allowEdit: true,
    targetWidth: 320,
    targetHeight: 240
  }
}
```

Le Composant PlacePage : `pages/place/place.ts`

```
let alert=this.alertCtrl.create({
  title:'Source',
  subTitle:'source?',
  buttons:[
    {
      text:'Camera', role:'camera',
      handler:()=>{
        this.takePicture(options1)
      }
    },
    {
      text:'Library',role:'library',
      handler:()=>{
        this.takePicture(options2)
      }
    }
  ]
})
alert.present();
}

takePicture(options){
  this.camera.getPicture(options).then((imageData) => {
    let base64Image = 'data:image/jpeg;base64,' + imageData;
    this.locService.addNewPhoto(base64Image,this.place.timestamp);
  }, (err) => {
    // Handle error
  });
}
}
```

Partie : My Gallery

- Cette partie de l'application permet de chercher une liste de photos à en faisant appel à l'api RESTful pixabay.org/api.



The screenshot shows a web browser window with the address bar displaying the URL: <https://pixabay.com/api/?key=5832566-81dc7429a63c86e3b707d0429&q=casablanca>. The page content displays a JSON response from the Pixabay API, which is expanded to show the details of the first search result. The JSON structure includes a 'totalHits' field and a 'hits' array. The first hit contains various metadata fields for a photo, such as preview dimensions, likes, favorites, tags, and URLs for the full image and user profile.

```
{
  "totalHits": 62,
  "hits": [
    {
      "previewHeight": 150,
      "likes": 7,
      "favorites": 11,
      "tags": "morocco, mosque, building",
      "webformatHeight": 640,
      "views": 1030,
      "webformatWidth": 427,
      "previewWidth": 100,
      "comments": 0,
      "downloads": 575,
      "pageURL": "https://pixabay.com/en/morocco-mosque-building-casablanca-2435391/",
      "previewURL": "https://cdn.pixabay.com/photo/2017/06/23/17/41/morocco-2435391_150.jpg",
      "webformatURL": "https://pixabay.com/get/eb31b20a2bfd003ed95c4518b7494e95e172e4d304b0144196f4c179aeeebc_640.jpg",
      "imageWidth": 3230,
      "user_id": 1014946,
      "user": "GregMontani",
      "type": "photo",
      "id": 2435391,
      "userImageURL": "https://cdn.pixabay.com/user/2015/07/09/08-56-44-639_250x250.jpg",
      "imageHeight": 4844
    },
    {
      "previewHeight": 99,
      "likes": 1,
      "comments": 0
    }
  ]
}
```

Le Composant MyGalleryPage : `pages/my-gallery/my-gallery.html`

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>My Gallery</ion-title>
    <button ion-button menuToggle="">
      <ion-icon name="menu"></ion-icon>
    </button>
  </ion-navbar>
</ion-header>
<ion-content padding>
  <form #f="ngForm" (ngSubmit)="onSearchImages(f.value)">
    <ion-item>
      <ion-label>Key Word</ion-label>
      <ion-input type="text" name="keyWord" required ngModel></ion-input>
    </ion-item>
    <button ion-button="" block="" type="submit">Chercher</button>
  </form>
  <ion-card *ngFor="let im of listImages">
    <ion-item>
      <ion-avatar item-start>
        
      </ion-avatar>
      <h2>{{im.user}}</h2>
      <p>{{im.id}}</p>
    </ion-item>
    
    <ion-card-content padding="">
      <p>{{im.tags}}</p>
    </ion-card-content>
  </ion-card>
</ion-content>
```

Le Composant MyGalleryPage : <pages/my-gallery/my-gallery.html>

```
<ion-row>
  <ion-col>
    <button ion-button icon-left clear small>
      <ion-icon name="thumbs-up"></ion-icon>
      <div>{{im.likes}} Likes</div>
    </button>
  </ion-col>
  <ion-col>
    <button ion-button icon-left clear small>
      <ion-icon name="text"></ion-icon>
      <div>{{im.comments}} Comments</div>
    </button>
  </ion-col>
  <ion-col>
    <button ion-button icon-left clear small>
      <ion-icon name="download"></ion-icon>
      <div>{{im.downloads}} Downloads</div>
    </button>
  </ion-col>
  <ion-col center text-center>
    <ion-note>
      2à mn A go
    </ion-note>
  </ion-col>
</ion-row>
</ion-card>
<ion-infinite-scroll (ionInfinite)="scroll($event)">
  <ion-infinite-scroll-content></ion-infinite-scroll-content>
</ion-infinite-scroll>
</ion-content>
```

Le Composant MyGalleryPage : `pages/my-gallery/my-gallery.ts`

```
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams } from 'ionic-angular';
import { Http } from "@angular/http";
import "rxjs/add/operator/map";
import { DetailImagePage } from "../detail-image/detail-image";
import { GalleryService } from "../../services/gallery.service";

@IonicPage()
@Component({
  selector: 'page-my-gallery',
  templateUrl: 'my-gallery.html',
})
export class MyGalleryPage {
  listImages=[];
  keyword:string;
  currentPage:number=1;
  size:number=10;
  totalPages:number=0;

  constructor(public navCtrl: NavController,
               public NavParams: NavParams,
               private galleryService:GalleryService) {

  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad MyGalleryPage');
  }
}
```

Le Composant MyGalleryPage : `pages/my-gallery/my-gallery.ts`

```
onSearchImages (dataForm) {
  this.listImages=[];
  this.totalPages=0;
  this.currentPage=1;
  this.doSearchImages (dataForm) ;
}
doSearchImages (dataForm) {
  this.keyWord=dataForm.keyWord;
  this.galleryService.getImage (this.keyWord,this.size,this.currentPage)
    .subscribe ((data)=>{
      console.log(data);
      data.hits.forEach (im=>{
        this.listImages.push(im);
      });
      this.totalPages=data.totalHits/this.size;
    })
  ;
}
scroll (infiniteScroll) {
  if (this.currentPage<this.totalPages) {
    ++this.currentPage;
    this.doSearchImages ({keyWord:this.keyWord});
    infiniteScroll.complete();
  }
}
detailImage (im) {
  this.navCtrl.push (DetailImagePage, im);
}
}
```


Le Composant DetailImagePage : <pages/detail-image/detail-image.html>

```
<ion-header>

  <ion-navbar color="danger">
    <ion-title>Détail Image</ion-title>
  </ion-navbar>

</ion-header>
<ion-content padding>
  <ion-card>
    <ion-item>
      <ion-thumbnail item-start="">
        
      </ion-thumbnail>
      <h2>User: <strong>{{image.user}}</strong> </h2>
      <p>Id Image:<strong>{{image.id}}</strong></p>
    </ion-item>
    <ion-card-content>
      <ion-item>
        <p>Size:<strong>{{image.imageWidth}} X
{{image.imageHeight}}</strong></p>
        <p>Tags:<strong>{{image.tags}} </strong></p>
      </ion-item>
      <ion-thumbnail>
        
      </ion-thumbnail>
    </ion-card-content>
  </ion-card>
</ion-content>
```

Le Composant DetailImagePage : `pages/detail-image/detail-image.ts`

```
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams } from 'ionic-angular';
@IonicPage()
@Component({
  selector: 'page-detail-image',
  templateUrl: 'detail-image.html',
})
export class DetailImagePage {
  image=null;

  constructor(public navCtrl: NavController, public NavParams:
NavParams) {
    this.image=this.NavParams.data;
    console.log(this.image);
  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad DetailImagePage');
  }
}
```

My Weather Page

- Cette partie de l'application permet de saisir une ville et afficher les données météo de cette ville en faisant appel à l'API openweathermap.org.



```
{
  "cod": "200",
  "message": "0.0043",
  "cnt": 40,
  "list": [
    {
      "dt": 1503403200,
      "main": {
        "temp": 299.92,
        "temp_min": 298.728,
        "temp_max": 299.92,
        "pressure": 1019.6,
        "sea_level": 1031.37,
        "grnd_level": 1019.6,
        "humidity": 74,
        "temp_kf": 1.19
      },
      "weather": [
        {
          "id": 802,
          "main": "Clouds",
          "description": "scattered clouds",
          "icon": "03d"
        }
      ],
      "clouds": {
        "all": 36
      },
      "wind": {
        "speed": 2.52,
        "deg": 109.5
      },
      "sys": {
        "pod": "d"
      }
    }
  ]
}
```

Le Composant MyMeteoPage : <pages/my-meteo/my-meteo..html>

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>My Meteo</ion-title>
    <button ion-button menuToggle="">
      <ion-icon name="menu"></ion-icon>
    </button>
  </ion-navbar>
</ion-header>
<ion-content padding>
  <div>
    <form #f="ngForm" (ngSubmit)="onGetWeather(f.value)">
      <ion-item>
        <ion-input type="text" name="city" required ngModel
placeholder="City"></ion-input>
      </ion-item>
      <button ion-button block type="submit">Get Weather</button>
    </form>
    <ion-card *ngFor="let m of weather?.list">
      <ion-item>
        <ion-avatar item-start>
          
        </ion-avatar>
        <h2><strong>{{m.dt*1000|date:'dd/MM/yyyy-HH:mm'}}</strong></h2>
        <h3><strong>{{m.main.temp-273.15|number:'1.0-2'}} °C</strong></h3>
        <p><strong>{{m.weather[0].description}}</strong></p>
      </ion-item>
    </ion-card>
  </div>
</ion-content>
```

Le Composant MyMeteoPage : <pages/my-meteo/my-meteo..html>

```
<ion-card-content>
  <p>
    Temp Min:<strong>{{m.main.temp_min-273.15|number:'1.0-2'}}°C</strong>,
    Temp Max:<strong>{{m.main.temp_max-273.15|number:'1.0-2'}}°C</strong>
  </p>
  <p>
    Pressure:<strong>{{m.main.pressure}}</strong>,
    Sea level:<strong>{{m.main.sea_level}}</strong>
  </p>
  <p>
    Humidity:<strong>{{m.main.humidity}} %</strong>,
    Grnd level:<strong>{{m.main.grnd_level}}</strong>
  </p>
  <p>
    Wind Speed:<strong>{{m.wind.speed}} km/h</strong>,
    Wind deg:<strong>{{m.wind.deg}}°</strong>
  </p>
</ion-card-content>
</ion-card>
</div>
</ion-content>
```

Le Composant MyMeteoPage : `pages/my-meteo/my-meteo.ts`

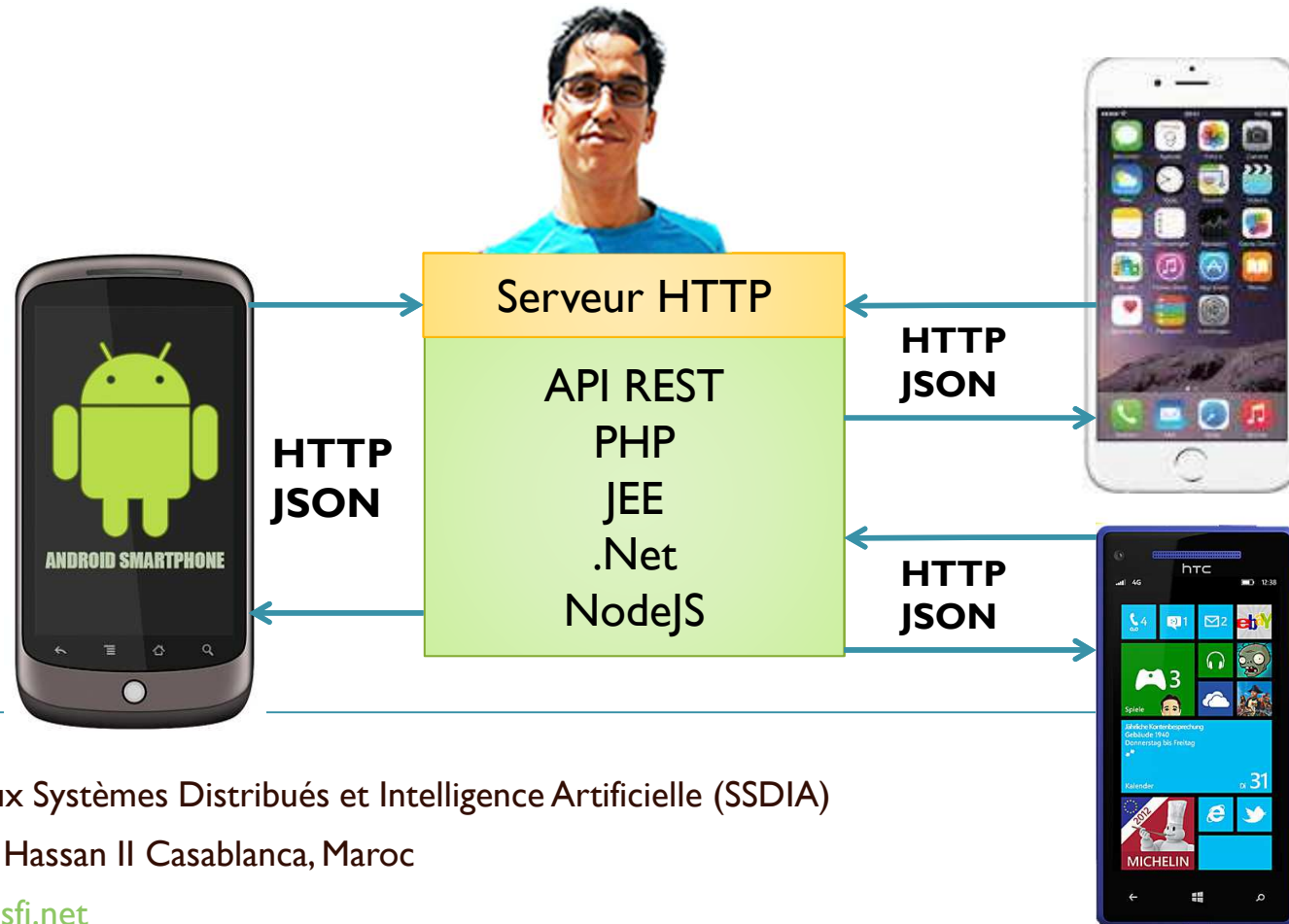
```
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams } from 'ionic-angular';
import { Http } from "@angular/http";

@IonicPage()
@Component({
  selector: 'page-my-meteo',
  templateUrl: 'my-meteo.html',
})
export class MyMeteoPage {
  weather:any=null;
  constructor(public navCtrl: NavController,
               public NavParams: NavParams,
               private http:Http) {

  }
  ionViewDidLoad() {
    console.log('ionViewDidLoad MyMeteoPage');
  }
  onGetWeather(data) {
    this.http.get('http://api.openweathermap.org/data/2.5/forecast?q='+data.city+'&AP
PID=a4578e39643716894ec78b28a71c7110')
      .map(resp=>resp.json())
      .subscribe(data=>{
        this.weather=data;
      })
  }
}
```

Développement Mobile Hybride

IONIC 3 : Part 10



Mohamed Youssfi

Laboratoire Signaux Systèmes Distribués et Intelligence Artificielle (SSDIA)

ENSET, Université Hassan II Casablanca, Maroc

Email : med@youssfi.net

Supports de cours : <http://fr.slideshare.net/mohamedyoussfi9>

Chaîne vidéo : <http://youtube.com/mohamedYoussfi>

Recherche : http://www.researchgate.net/profile/Youssfi_Mohamed/publications