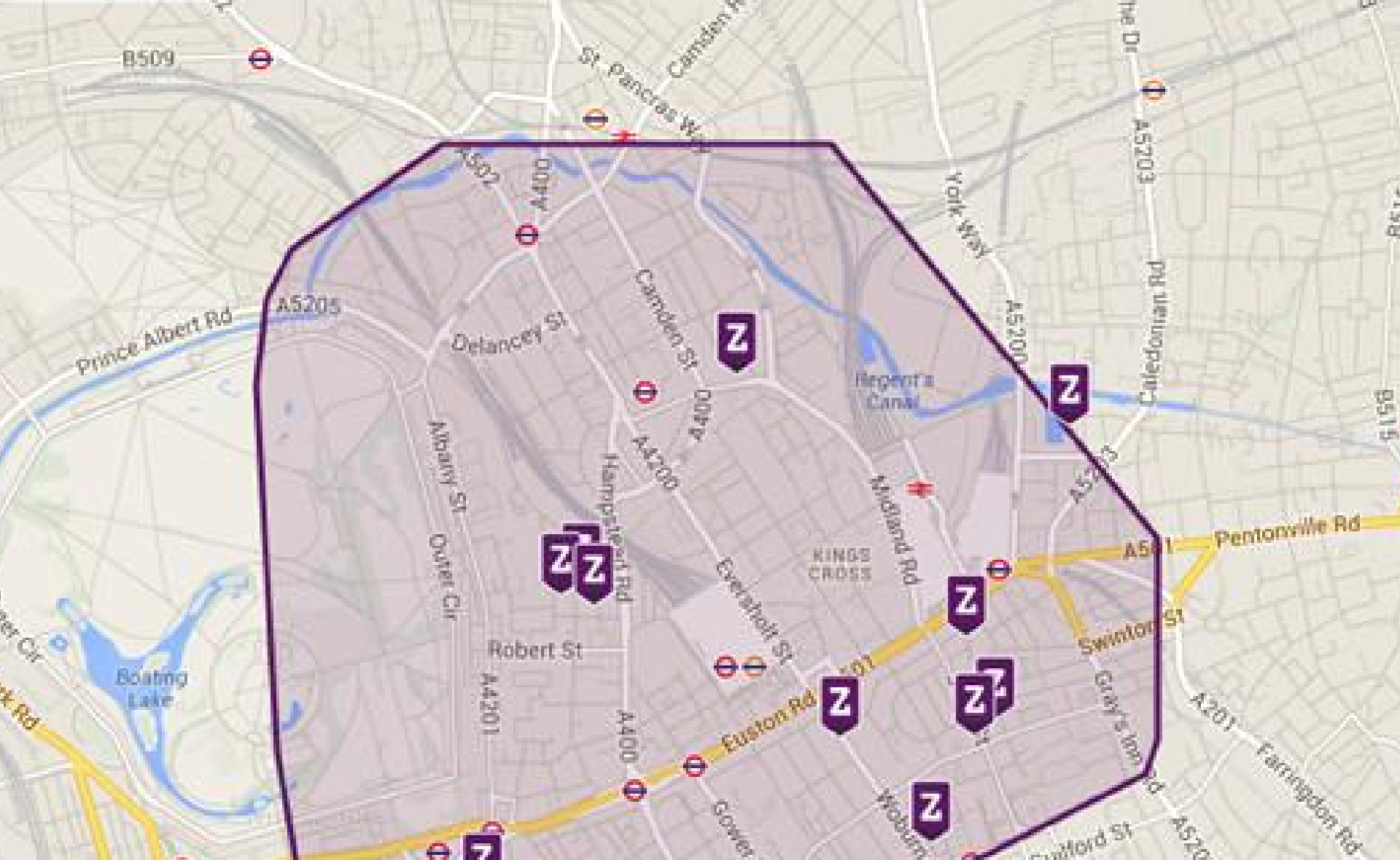


Geo Catching Project Kick-off

12/12/2016

Cécile Camillieri/Clément Duffau





Reminder

Time to start!

Requirements: first version

Drawing of zones on a map to create a game field

User login and joining of a game

Geolocation of the logged player

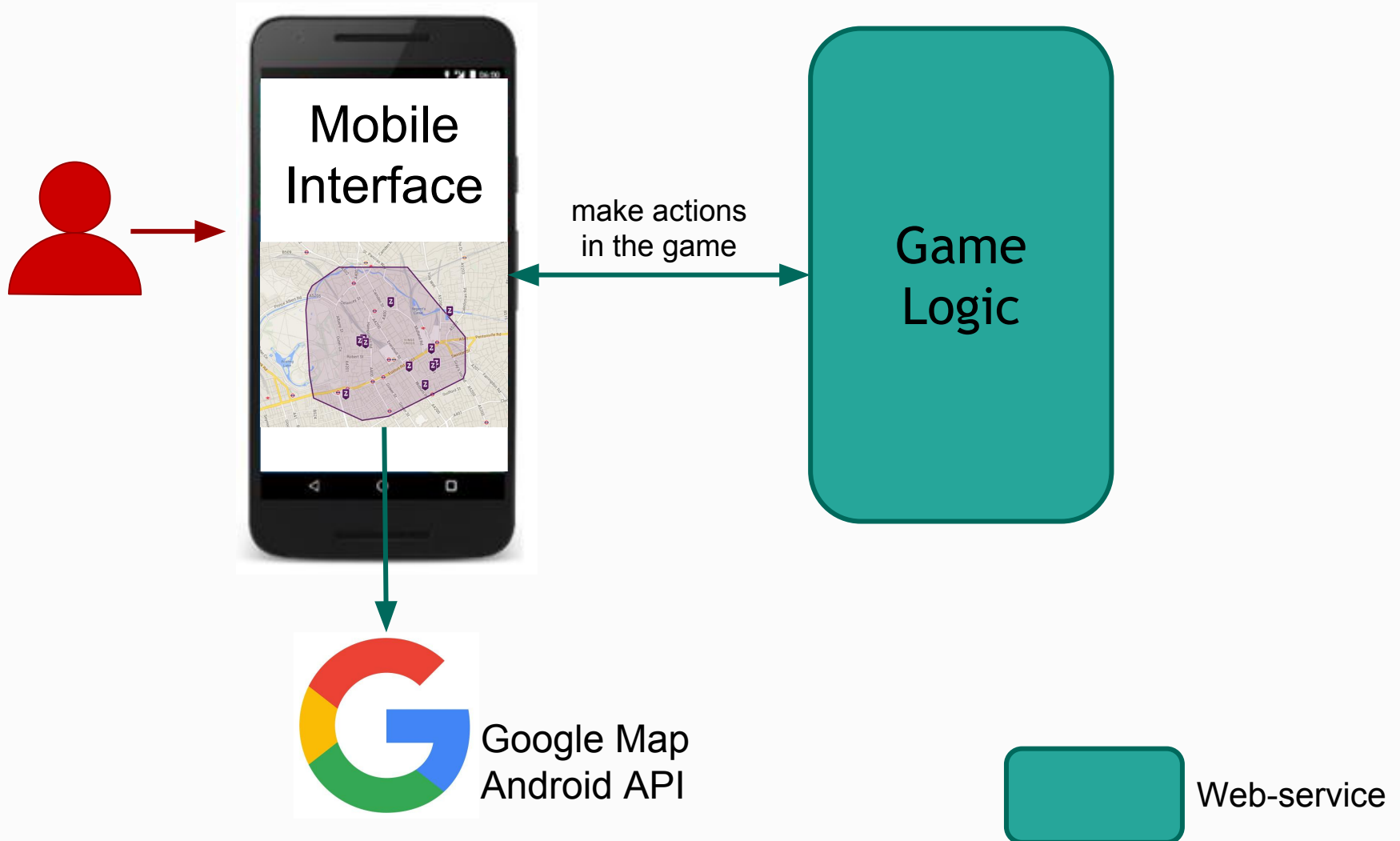
Capture of a zone when the player comes in

Display location of the other players on the map

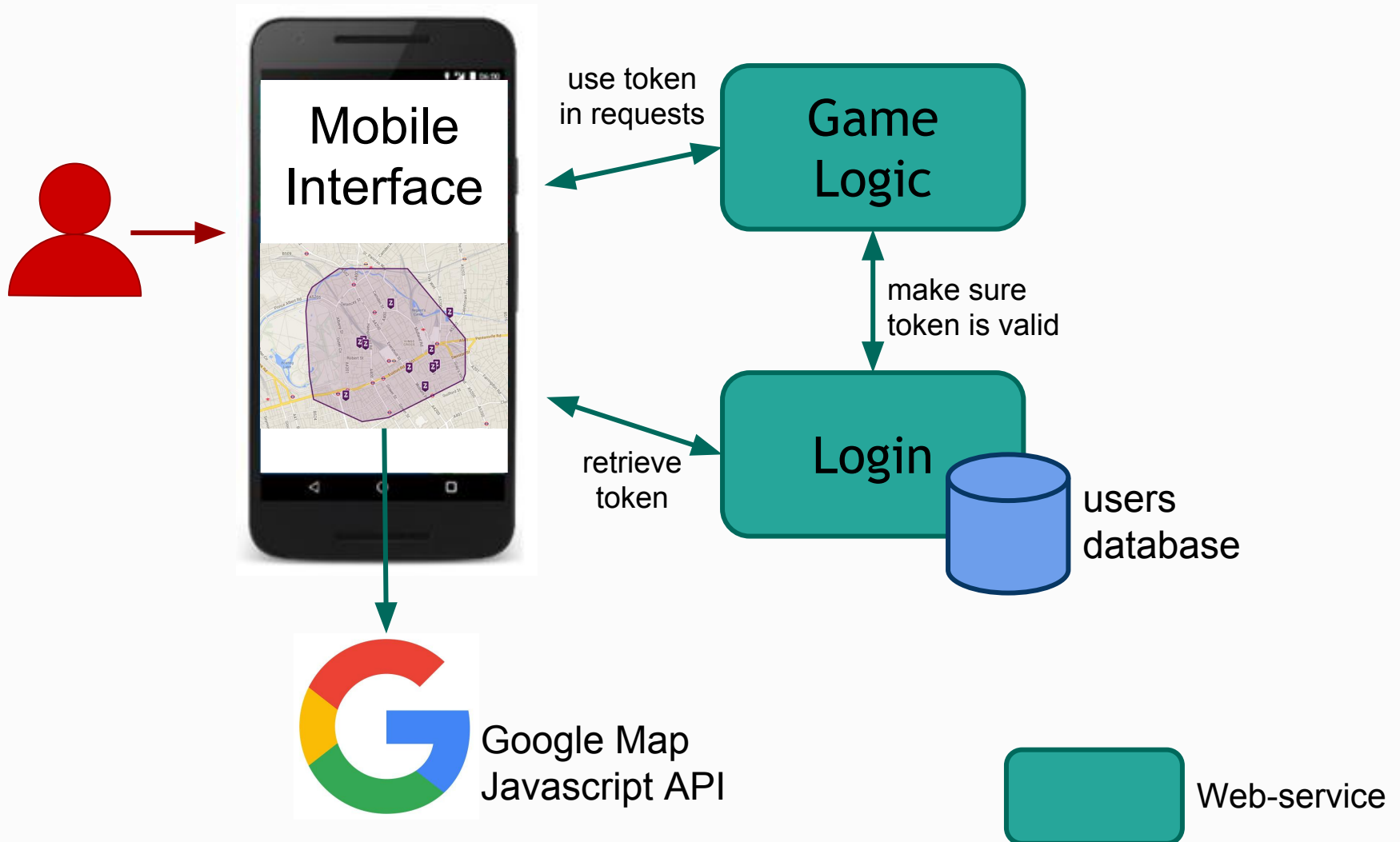
Display the name of the “owner” of a zone by clicking on it

Guidelines

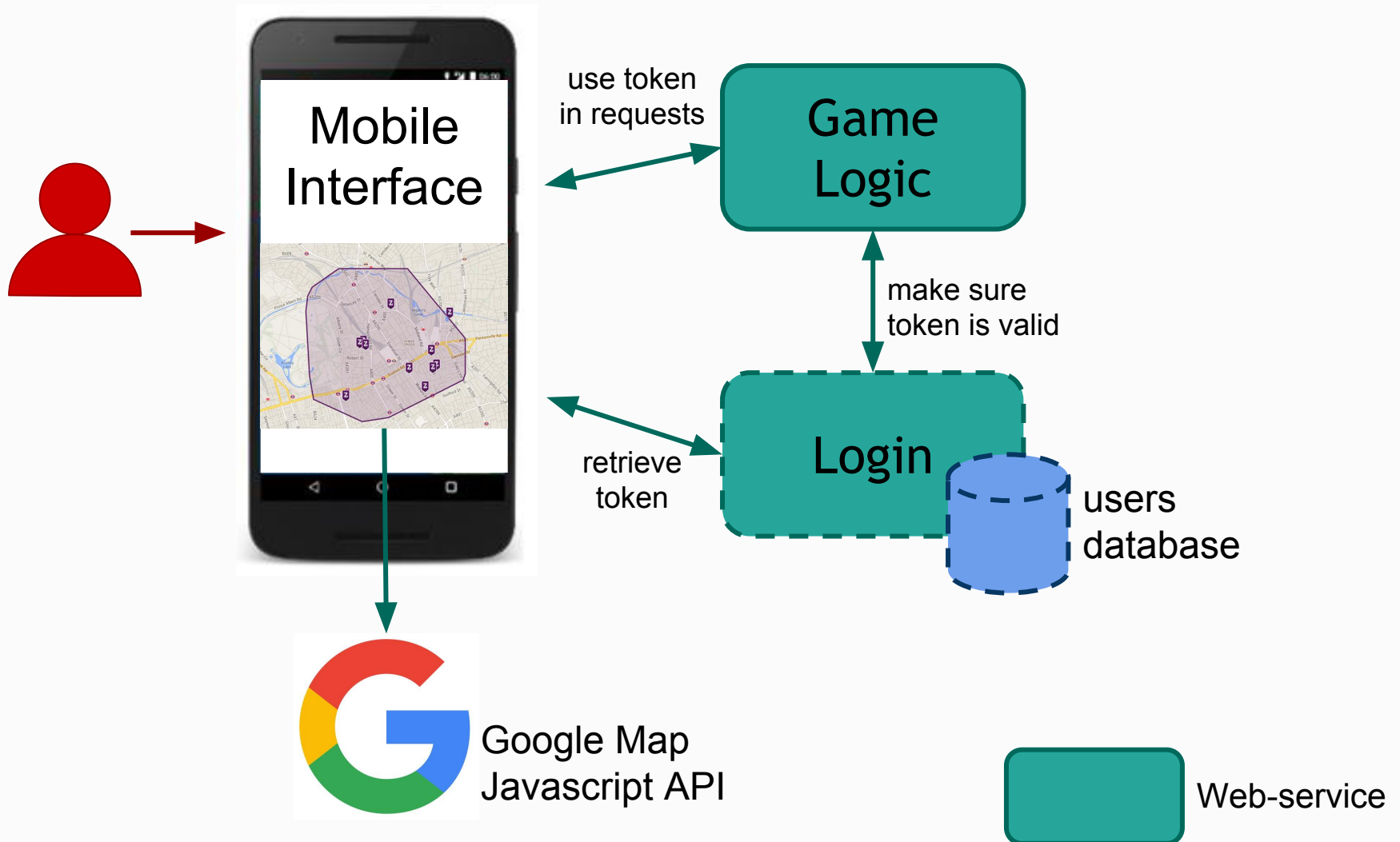
Basic Architecture



Proposed Architecture



Proposed Architecture



Implementation

A Java web-service to handle the **logic of the game**

- Knows who the players are
- Knows the zones for the game
- Tracks location of players
- Checks if a player is in a zone

Connect to the **login web-service**

An Android user interface using Google Map's API and the web-services above.

General Guidelines

Don't forget about **project management** !

Define and share your **tasks** properly.

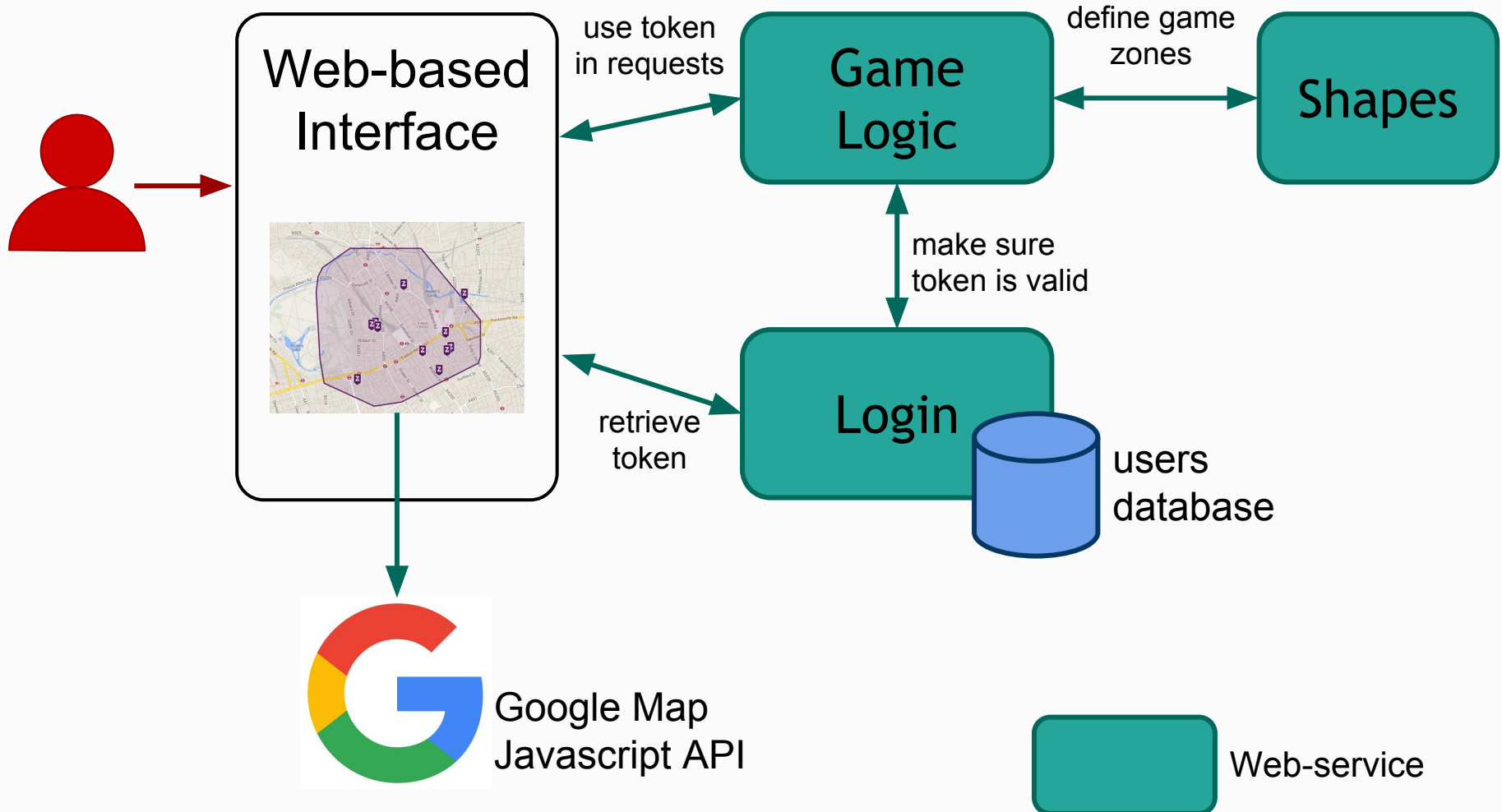
Commits should correspond to tasks in Jira

- Example : “MyTaskId: Fix map display bug on Safari”

How to...

Handle login

Reminder: Proposed Architecture



Token-based Authentication

- 1- User provides their username and password
- 2- User Interface queries **authentication Web-Service**
- 3- **Web-Service** checks the credentials and generates a unique session token
- 4- Every time a request that requires authentication is made to **your service**, the token is passed as a parameter
- 5- **Your web service** must check with the **authentication service** that the token is valid.
- 6- If so, the request can continue as normal.

First step

- **Provided for the first sprint:**
 - Log user and generate token
 - Checks that a token is valid
 - Service is deployed, you just have to query it
- **Each of you has an account:**
 - login and password: your student id
- **In the future (sprint 2):**
 - we may or may not give you the source code
 - you have to re reimplement/improve the service

Operations: login

- <http://iut-outils-gl.i3s.unice.fr/jetty/authentication/>
- Method: **POST**
- Content-Type: **application/x-www-form-urlencoded**
- Parameters (in request body):
 - username
 - password, hashed with MD5 algorithm
- Response:
 - success: code 200 (ok) + token
 - failure: code 401 (unauthorized)

Operations: token validation

- <http://iut-outils-gl.i3s.unice.fr/jetty/authentication/session>
- Method: **POST**
- Content-Type: **application/x-www-form-urlencoded**
- Parameters (in request body):
 - token
- Response:
 - success: code 204 (no content)
 - failure: code 401 (unauthorized)

How to...

Deploy your web service

Back to: Plugins for custom goals

mvn jetty:run

run a jetty (server) instance based on the configuration

```
<configuration>
  <scanIntervalSeconds>10</scanIntervalSeconds>
  <webApp>
    <contextPath>/</contextPath>
    <descriptor> [...] web.xml</descriptor>
  </webApp>
</configuration>
```



Contains configuration for the web-service

Back to: Plugins for custom goals

mvn jetty:run

run a jetty (server) instance based on the configuration

=> Deploy the war on a jetty instance

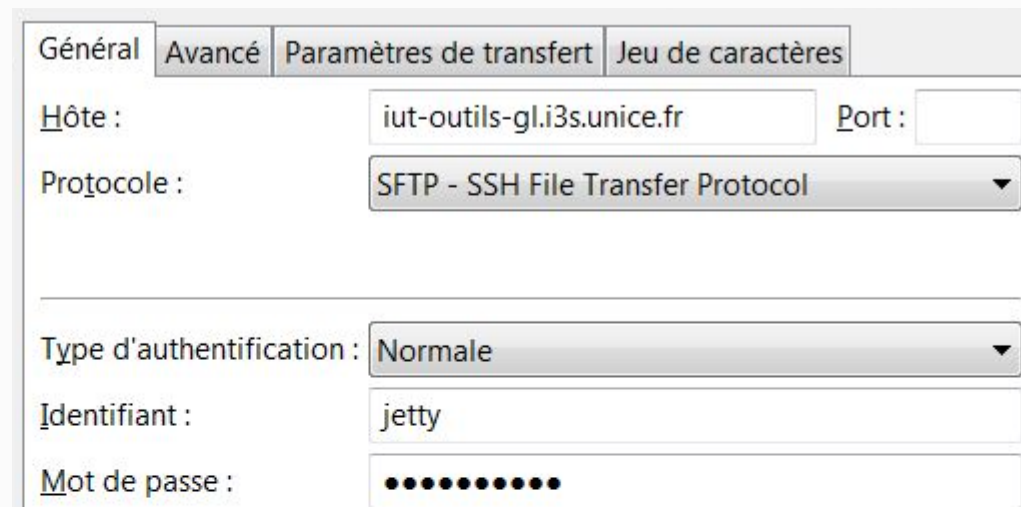
=> Available at localhost:8080

Deploying the war on a server

- Jetty server available at <http://iut-outils-gl.i3s.unice.fr/jetty/>
Username: 'jetty' Password: 'deploy2016'

- Deployment:

- **Command line:** `scp /path/to/war jetty@iut-outils-gl.i3s.unice.fr:/opt/jetty/webapp/my.war`
- Using a tool like [FileZilla](#):



The image shows a screenshot of the FileZilla configuration dialog box, specifically the 'Général' (General) tab. The dialog is used to configure an SFTP connection. The fields are as follows:

- Hôte :** iut-outils-gl.i3s.unice.fr
- Port :** (empty)
- Protocole :** SFTP - SSH File Transfer Protocol
- Type d'authentification :** Normale
- Identifiant :** jetty
- Mot de passe :** (masked with dots)

Deploying the war on a server

- Your war must be directly in the folder `/opt/jetty/webapp/`
 - No sub-folder
- Only upload a single war
 - Always use the same name for your war
- The name of your war must follow this pattern
 - Start with 'dam-x' where x is the id of your team (a to f)
 - A file 'dam-z.war' will result in your web-service to be deployed at the url:

<http://iut-outils-gl.i3s.unice.fr/jetty/dam-z/{whats/defined/in/your/war}>

How to...

Maps API

For Android...



WHOOOPS...

Setting up with an existing project

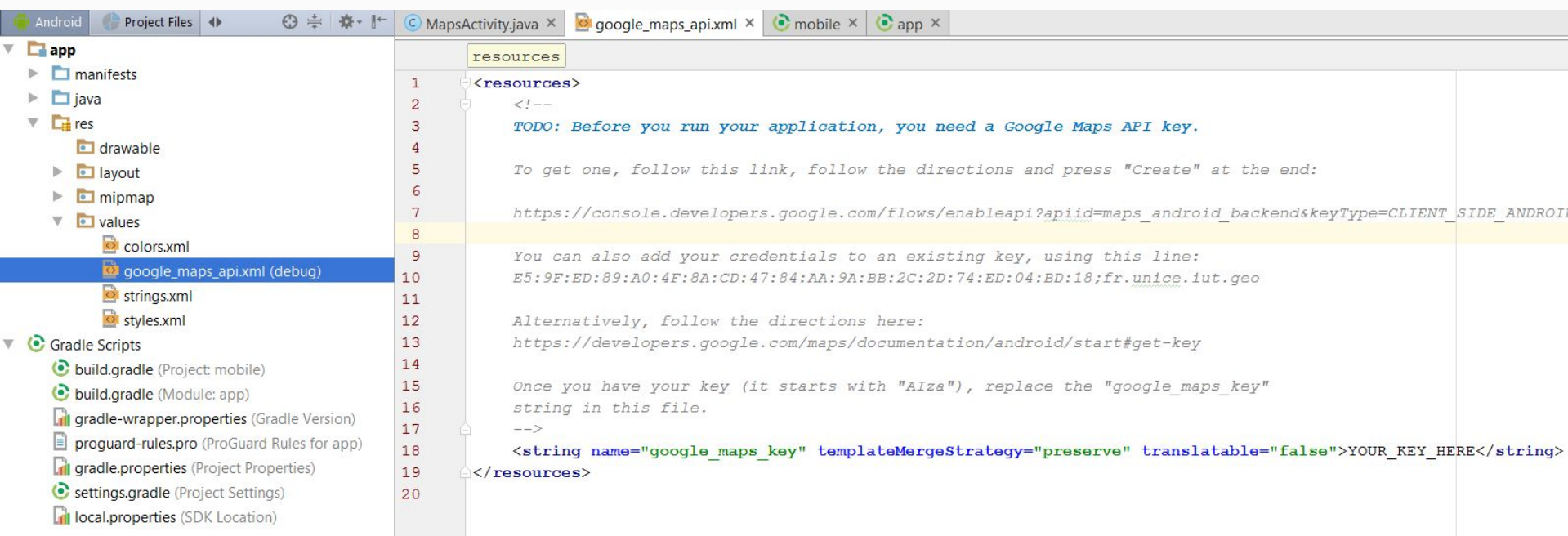
- Create a new Google Maps Activity

The screenshot shows an IDE's 'New' menu with 'Activity' selected. The 'Activity' option is highlighted in blue. Below the 'Activity' option, a list of activity templates is displayed, including 'Gallery...', 'Always On Wear Activity (Requires minSdk >= 20)', 'Android TV Activity (Requires minSdk >= 21)', 'Basic Activity (Requires minSdk >= 9)', 'Blank Wear Activity (Requires minSdk >= 20)', 'Empty Activity (Requires minSdk >= 9)', 'Fullscreen Activity (Requires minSdk >= 11)', 'Login Activity (Requires minSdk >= 8)', 'Master/Detail Flow (Requires minSdk >= 7)', 'Navigation Drawer Activity (Requires minSdk >= 7)', 'Scrolling Activity (Requires minSdk >= 7)', 'Settings Activity (Requires minSdk >= 11)', and 'Tabbed Activity (Requires minSdk >= 13)'. The 'Gallery...' option is highlighted in blue. Below the list of activity templates, the following dependencies are listed:

```
t:appcompat-v7:23.4.0'  
.gms:play-services:10.0.1'  
12'
```

Setting up with an existing project

- A file 'google_maps_api.xml' will be added in your resources

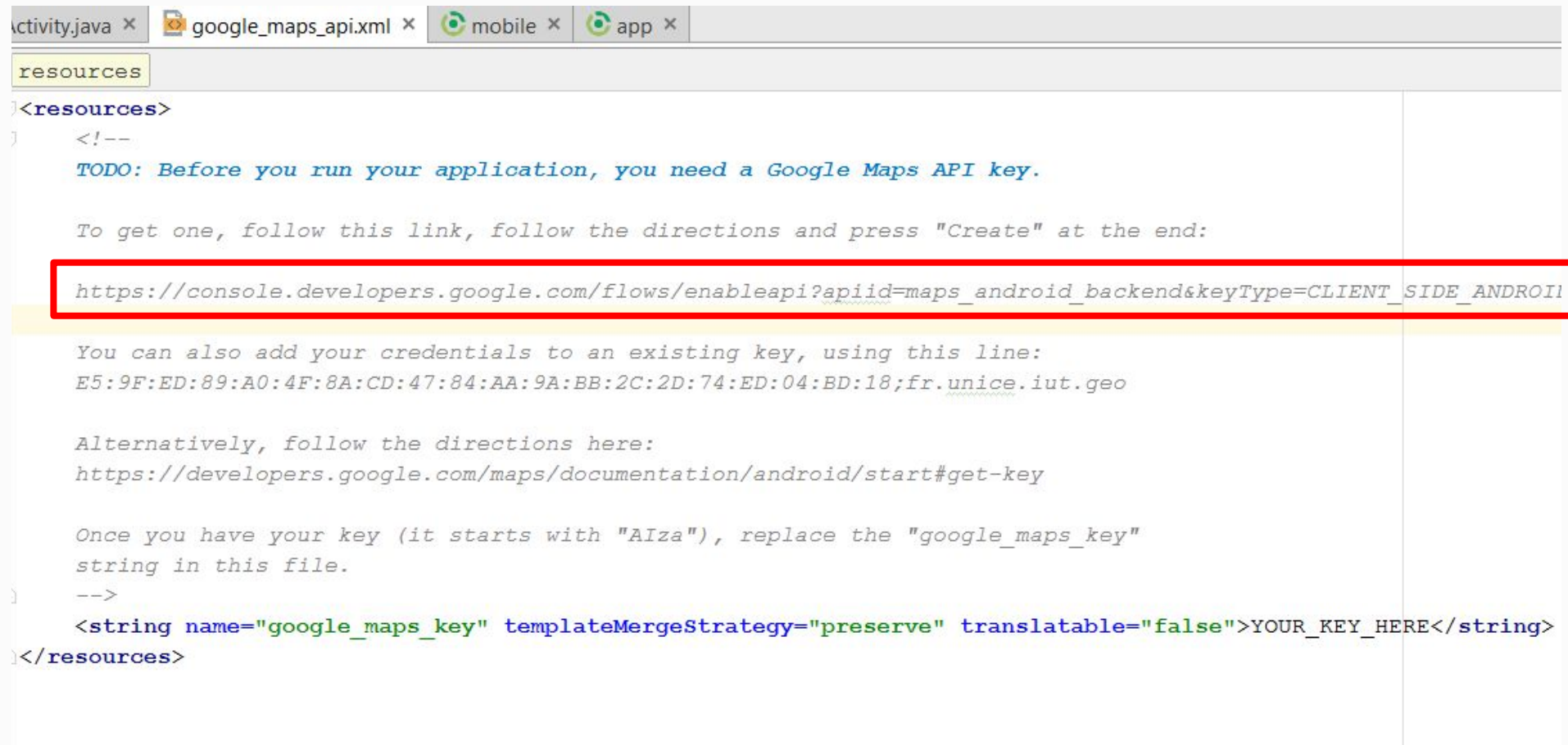


The screenshot shows an IDE interface with a project tree on the left and a code editor on the right. The project tree shows the 'app' directory with sub-directories 'manifests', 'java', and 'res'. The 'res' directory is expanded to show 'drawable', 'layout', 'mipmap', and 'values'. The 'values' directory is further expanded to show 'colors.xml', 'google_maps_api.xml (debug)', 'strings.xml', and 'styles.xml'. The code editor shows the content of 'google_maps_api.xml' with the following text:

```
1 <resources>
2 <!--
3  TODO: Before you run your application, you need a Google Maps API key.
4
5  To get one, follow this link, follow the directions and press "Create" at the end:
6
7  https://console.developers.google.com/flows/enableapi?apiid=maps\_android\_backend&keyType=CLIENT\_SIDE\_ANDROID
8
9  You can also add your credentials to an existing key, using this line:
10 E5:9F:ED:89:A0:4F:8A:CD:47:84:AA:9A:BB:2C:2D:74:ED:04:BD:18;fr.unice.iut.geo
11
12 Alternatively, follow the directions here:
13 https://developers.google.com/maps/documentation/android/start#get-key
14
15 Once you have your key (it starts with "AIza"), replace the "google_maps_key"
16 string in this file.
17 -->
18 <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
19 </resources>
20
```

Setting up with an existing project

- Follow this link to get a key



```
activity.java x google_maps_api.xml x mobile x app x
resources
} <resources>
  <!--
  TODO: Before you run your application, you need a Google Maps API key.

  To get one, follow this link, follow the directions and press "Create" at the end:

  https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID

  You can also add your credentials to an existing key, using this line:
  E5:9F:ED:89:A0:4F:8A:CD:47:84:AA:9A:BB:2C:2D:74:ED:04:BD:18;fr.unice.iut.geo

  Alternatively, follow the directions here:
  https://developers.google.com/maps/documentation/android/start#get-key

  Once you have your key (it starts with "AIza"), replace the "google_maps_key"
  string in this file.
  -->
  <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
} </resources>
```

Setting up with an existing project

- Follow the instructions

☰ Google APIs



Enregistrer l'application pour Google Maps Android API dans la Console d'API Google

Console d'API Google vous permet de gérer votre application et de surveiller l'utilisation de l'API.

Vous n'avez aucun projet existant. Un projet nommé "My Project" va être créé.

Veillez m'envoyer par e-mail des informations concernant les nouvelles fonctionnalités annoncées, des suggestions pour améliorer les performances, des enquêtes de satisfaction et des offres spéciales.

Oui Non

Je reconnais que l'utilisation de tous les [services et API associées](#) est soumise aux [Conditions d'utilisation](#).

Oui Non

Accepter et continuer

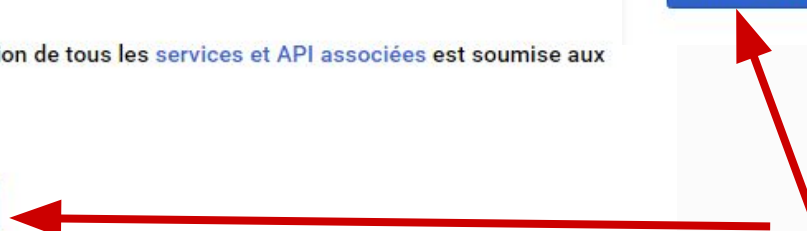
☰ Google APIs



Le projet a été créé et "Google Maps Android API" a été activé.

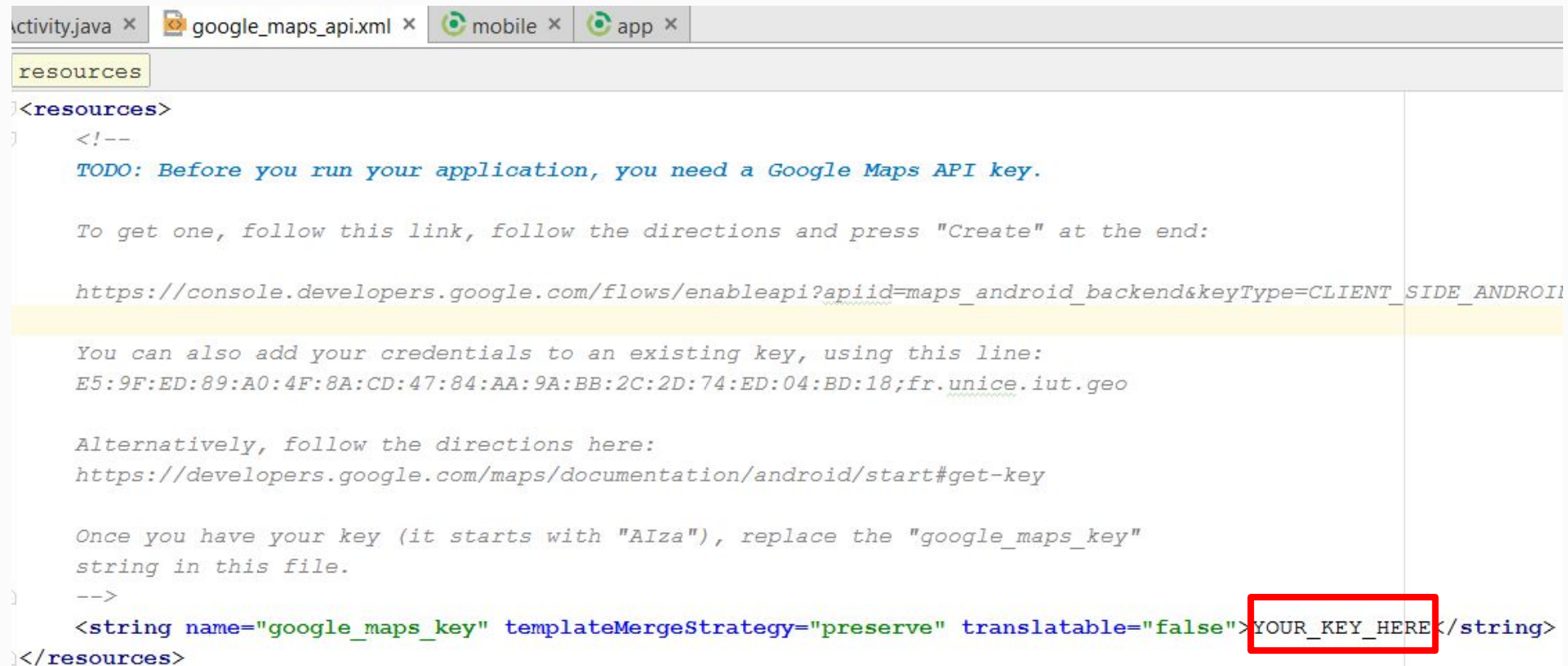
Ensuite, vous devez créer une clé API afin d'appeler l'API.

Créer la clé d'API



Setting up with an existing project

- Put your key in the file



```
activity.java x google_maps_api.xml x mobile x app x
resources
<resources>
  <!--
  TODO: Before you run your application, you need a Google Maps API key.

  To get one, follow this link, follow the directions and press "Create" at the end:

  https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID

  You can also add your credentials to an existing key, using this line:
  E5:9F:ED:89:A0:4F:8A:CD:47:84:AA:9A:BB:2C:2D:74:ED:04:BD:18;fr.unice.iut.geo

  Alternatively, follow the directions here:
  https://developers.google.com/maps/documentation/android/start#get-key

  Once you have your key (it starts with "AIza"), replace the "google_maps_key"
  string in this file.
  -->
  <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
</resources>
```

Project configuration

- **build.gradle** should have a google play services dependency

```
compile 'com.google.android.gms:play-services:10.0.1'
```

- **the manifest** should define the proper permissions

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

```
<uses-permission android:name="android.permission.INTERNET" />
```

- **the manifest** should define the key and play services version

```
<meta-data
```

```
    android:name="com.google.android.geo.API_KEY"
```

```
    android:value="@string/google_maps_key" />
```

```
<meta-data
```

```
    android:name="com.google.android.gms.version"
```

```
    android:value="@integer/google_play_services_version" />
```


Setting up without Android Studio

- Detailed instructions

- <https://developers.google.com/maps/documentation/android-api/config?hl=fr>
- https://developers.google.com/maps/documentation/android-api/signup?hl=fr#obtenir_une_cle_dapi_android

- For Xamarin

- https://developer.xamarin.com/guides/android/platform_features/maps_and_location/maps/part_2_-_maps_api/
- https://developer.xamarin.com/guides/android/platform_features/maps_and_location/maps/obtaining_a_google_maps_api_key/

- Documentation

- <https://developers.google.com/maps/documentation/android-api/intro?hl=fr>

How to... Emulate Android

With Genymotion

- Genymotion is a free, powerful Android emulator
 - Create a free account: <https://www.genymotion.com/>
 - Download & Install
 - Launch emulation for pre-existing devices
- Plugin for Android Studio
 - Use from Android Studio: <https://www.genymotion.com/plugins/>
- Enable use of Google Play Services (needed for maps)
 - <https://inthecheesefactory.com/blog/how-to-install-google-services-on-genymotion/en>

How to...

Continuous integration

Jenkinsfile at repository root

```
node ('master') {  
  
    stage('back-maven') {  
        checkout scm  
        sh 'mvn -f mavenfolder/pom.xml clean package'  
    }  
  
    stage('front-android') {  
        checkout scm  
        sh 'cd androidfolder; gradle assemble'  
    }  
  
}
```

Summary

What's expected - Functionalities

Drawing of zones on a map to create a game field

User login and joining of a game

Geolocation of the logged player

Capture of a zone when the player comes in

Display location of the other players on the map

Display the name of the “owner” of a zone by clicking on it

Some references

- Drawing of zones (Polygones) on the map
 - <https://developers.google.com/maps/documentation/android-api/shapes?hl=en>
- Geolocation of the player
 - <https://developers.google.com/maps/documentation/android-api/location?hl=fr>
- Display other players location
 - <https://developers.google.com/maps/documentation/android-api/marker?hl=fr>
 - <https://developers.google.com/maps/documentation/android-api/infowindows>
- HTTP Requests in Android
 - <https://developer.android.com/training/volley/index.html>
 - <https://developer.android.com/reference/java/net/URLConnection.html>
- MD5 Hashing in Android
 - <http://www.kospol.gr/204/create-md5-hashes-in-android/>

What's expected - Release

- Deploy your web-service on the provided server.
- Release on Github and Jira.
- The repository should contain:
 - A folder with your android project
 - A folder with your game logic web-service
 - A docs/ folder with your slides for this sprint
 - A readme file saying how to run the project and describing what is done and the contents on the repository
 - A Jenkinsfile setting a pipeline for your maven and android project
- Repository is queried by a script. **Any problem → 0**
- We only grade what's on the **master branch**.
- We get the tag **"sprint-1"**. **No tag → 0**
- Deadline is **January 15 at 23:59**. **Late → 0**

What's expected - defense

January 17

10 minutes to present your work and a short demo

10 minutes of questions

Minimal content

- Current state of the project (including demo)
- Pros/Cons of your solution
- What's next

Too much to do!!

- Start **early**
- Start **small**
- Use **mocks**
- Share **tasks**
- Set **deadlines**



- The following are **not problems** (for now):
 - Interface doesn't look nice on all devices
 - Network issues are not handled properly
 - No persistence



GO!!

