Proiect *Smart Parking controlled from Cloud*

Dezvoltarea aplicatiilor de tip Cloud

Echipa: Powerpuff Girls

- Bianca Beatrice BARLEA
- Mihaela TUTU
- Bogdan NISTOR

Cuprins:

1. Conceptul de Smart city si Smart parking
2. Scopul proiectului
3. Implementare
4. Tehnologii folosite
   - 4.1 Google Cloud Platform – API Google Maps
   - 4.2 Firebase
   - 4.3 Android Studio
   - 4.4 Limbajul de programare Java
   - 4.5 Git
5. Bibliografie

1. Conceptul de Smart city si Smart parking

Conceptul de Internetul Lucrurilor *(Internet of Things)* va conecta obiectele lumii atât într-o manieră senzorială cât şi inteligentă combinând evoluţia tehnologică in identificarea elementelor, senzori şi reţele de senzori wireless, sisteme încorporate şi nanotehnologii.

Aceste sistemele inteligente au început să facă parte tot mai mult din viaţa contemporană cu scopul de a uşura şi simplifica anumite aspecte ale acesteia. Toate acestea fiind posibile datorită internetului de mare viteza care permite comunicarea între dispozitive în timp real

Câteva exemple de astfel de sisteme inteligente ar fi:

- Frigiderul capabil de managementul alimentelor
- Cuptorul controlat printr-o aplicaţie mobilă
- Termostatul controlat prin aplicaţie mobilă
- Bec controlat prin aplicaţie mobilă
- Optimizarea automată a traficului
- Parcare care recunoaşte locurile libere şi transmite informaţia printr-o aplicaţie mobilă/web

Sistemul de Smart Parking se încadrează sub acest concept prin simpla sa funcţionalitate, prin scopul îndeplinit şi prin modul de comunicare cu utilizatorul. Permiţând acces şi interacţiune cu o largă gamă de dispozitive cum ar fi: electrocasnice, camere de supraveghere, senzori de monitorizare, display-uri, vehicule şi aşa mai departe, Internetul Lucrurilor va găzdui dezvoltarea unor aplicaţii care pot folosi cantităţi enorme şi variate de date generate de obiecte pentru a furniza noi servicii cetăţenilor, companiilor şi administraţiilor publice. Smart Parking presupune un sistem mecanizat menit să mărească capacitatea de parcare dintr-un anumit spaţiu.

2. Scopul proiectului

Scopul proiectului este unul didactic si urmareste familiarizarea noastra cu:

- tehnologiile Cloud.;
- etapele de dezvolare ale unui proiect;
- distribuirea muncii sub forma de task-uri;
- estimarea timpului necesar implementarii acestor task-uri.

3. Implementare

In specificatiile proiectului aveam 3 cerinte obligatorii:

- Worker in Cloud, de care s-a ocupat Mihaela TUTU
- Repository in Git functional

- Utilizarea API-ului pentru Google Maps si dashboard pentru afisarea informatiilor utile – Bianca BARLEA

Worker-ul genereaza date (0 si 1) care sunt salvate in Firebase, de unde trebuie luate in aplicatia Android care afiseaza harta si locurile ocupate/disponibile.

Worker-ul din Cloud:

Worker-ul implementat are genereaza valori aleatoare, 0 si 1 pe care le pune in baza de date create in Firebase.

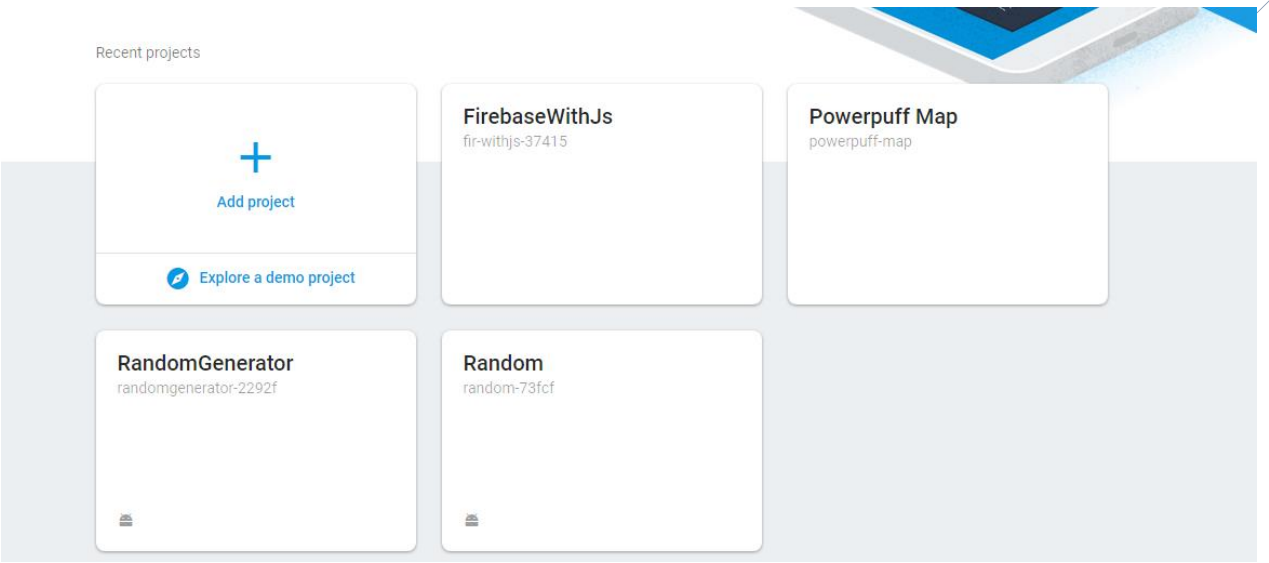
Este un script JavaScript, care ruleaza intr-o pagina index.html.

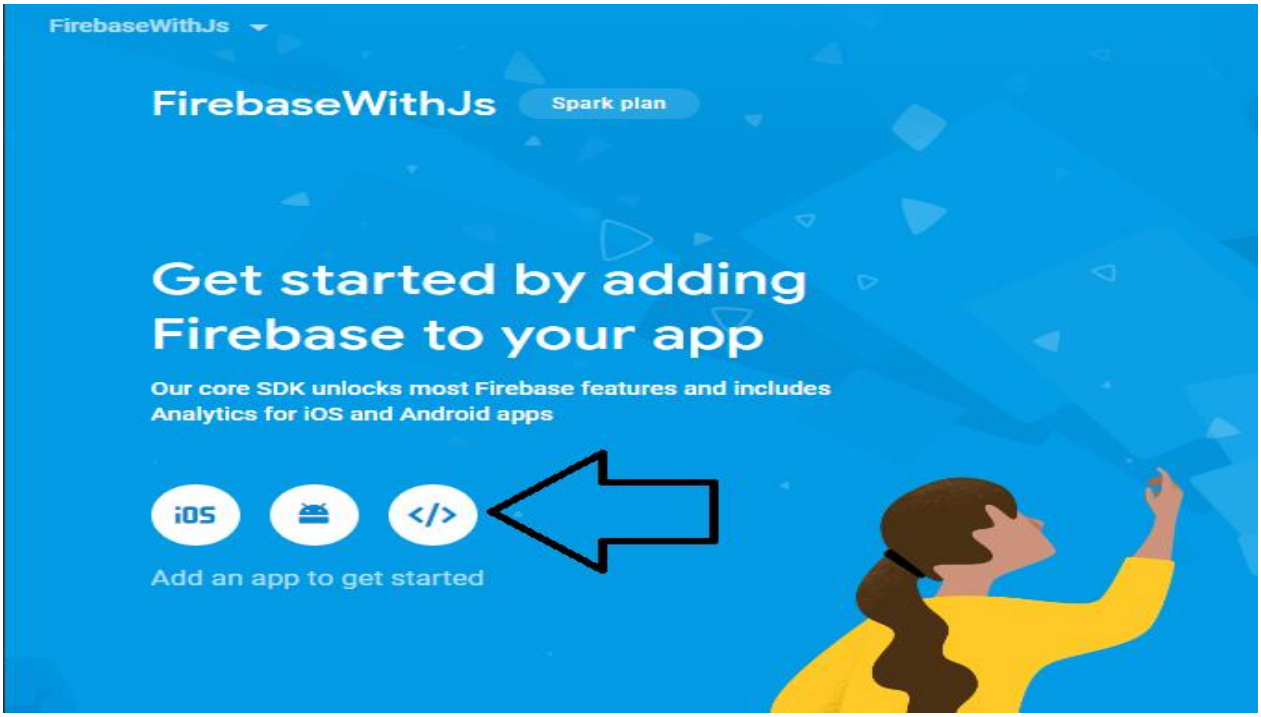
Ideea de worker care automatizeaza sarcini si reduce efortul introducerii manuale a datelor este sustinuta de script partial. Am incercat folosirea unei bucle infinite while(true) in script, dar fiind rulat dintr-o pagina html, aceasta devenea *"irresponsive"*, motiv pentru care am ales o bucla for cu un numar limitat de cicluri, multiplu de 4 (4 fiind numarul de locuri de parcare disponibile in parcarea noastra).

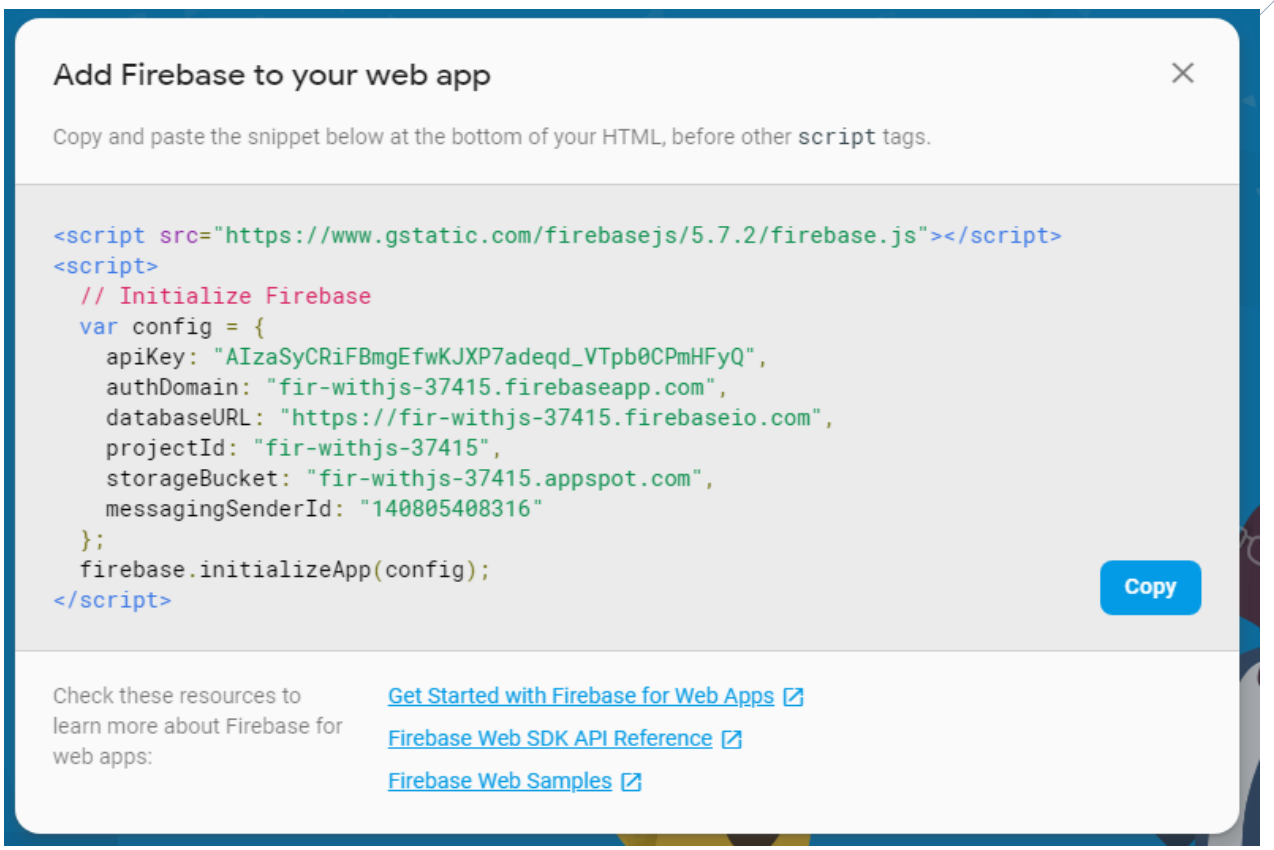
```
<script>
    function myFunction() {

        var i = 0;
        while(i < 12){
            var random = Math.floor(Math.random() * 2);
            var random1 = Math.floor(Math.random() * 2);
            var random2 = Math.floor(Math.random() * 2);
            var random3 = Math.floor(Math.random() * 2);
            console.log(random);
            console.log(random1);
            console.log(random2);
            console.log(random3);
           // alert(random);
            // the reference to my database
            var firebaseRef = firebase.database().ref();
            firebaseRef.child("isOccupied").set(random);
            var firebaseRef = firebase.database().ref();
            firebaseRef.child("isOccupied1").set(random1);
            var firebaseRef = firebase.database().ref();
            firebaseRef.child("isOccupied2").set(random2);
            var firebaseRef = firebase.database().ref();
            firebaseRef.child("isOccupied3").set(random3);
            i++;
        }
}
    myFunction();
    </script>
```

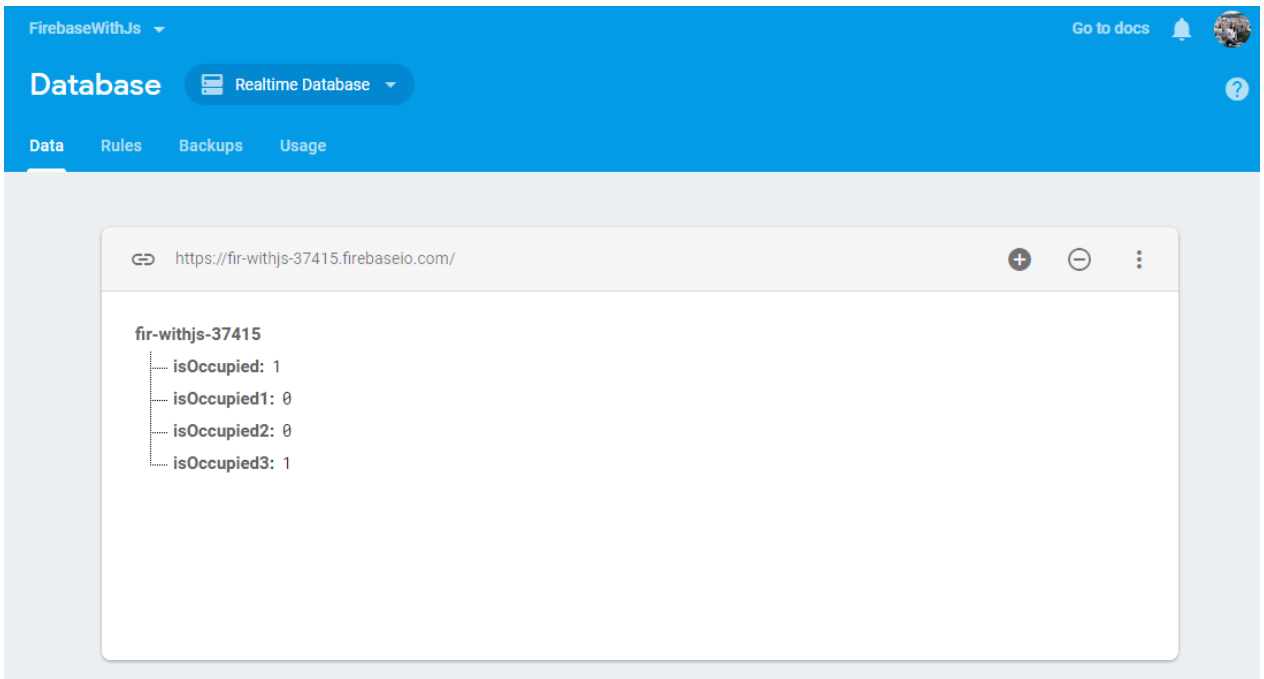
Pentru a putea pune datele in baza de date, la crearea proiectului in Firebase am ales WebApp ca tipul proiectului, motiv pentru care a fost nevoie ca inainte de rularea script-ului de generare si inserare date, sa rulam script-urile de conectare la baza de date.

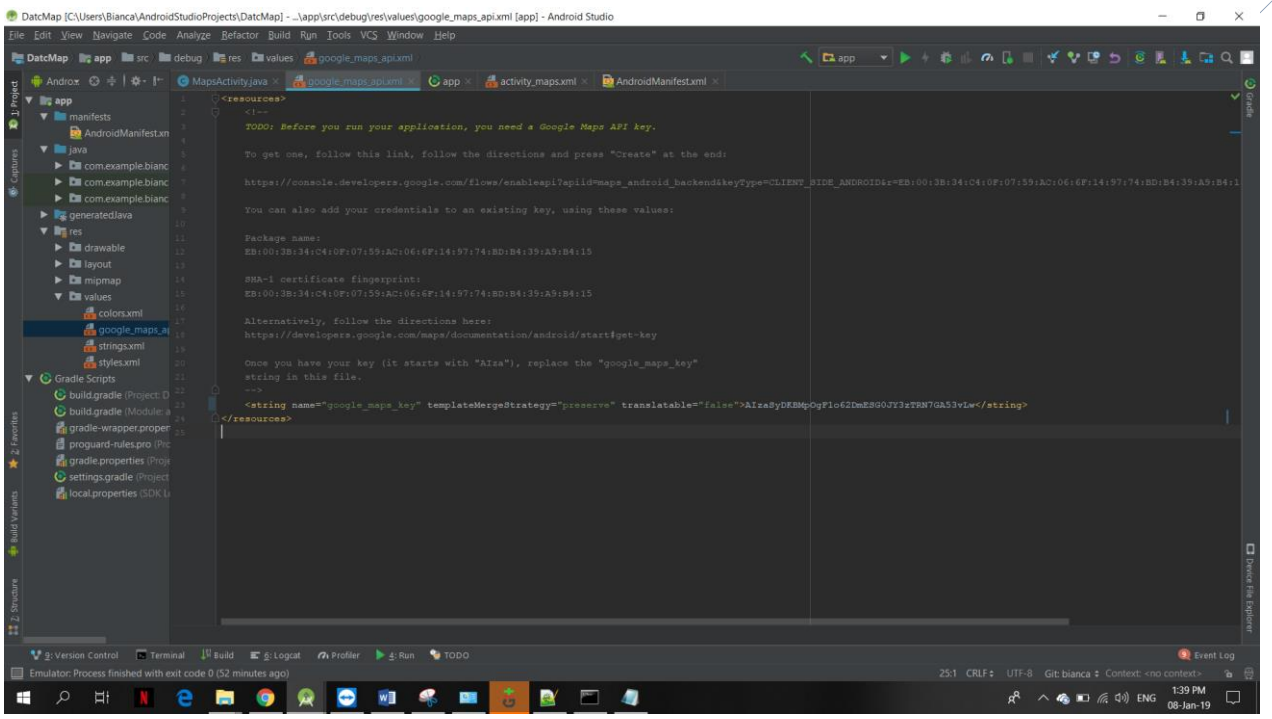
```
<script src="https://www.gstatic.com/firebasejs/5.7.2/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyCRiFBmgEfwKJXP7adeqd_VTpb0CPmHFyQ",
    authDomain: "fir-withjs-37415.firebaseapp.com",
    databaseURL: "https://fir-withjs-37415.firebaseio.com",
    projectId: "fir-withjs-37415",
    storageBucket: "",
    messagingSenderId: "140805408316"
  };
  firebase.initializeApp(config);
</script>
```

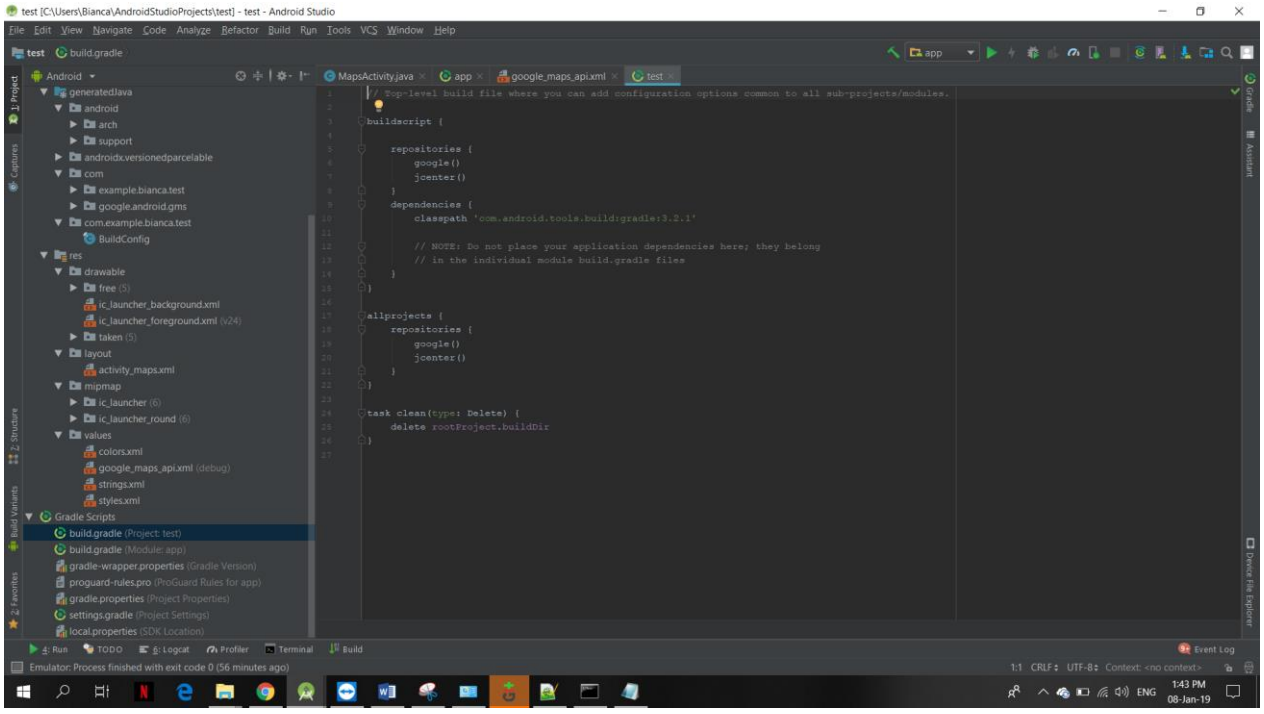

Recent projects
Add project
Explore a demo project
FirebaseWithJs
fir-withjs-37415
Powerpuff Map
powerpuff-map
RandomGenerator
randomgenerator-2292f
Random
random-73fcf


FirebaseWithJs
FirebaseWithJs
Spark plan
Get started by adding Firebase to your app
Our core SDK unlocks most Firebase features and includes Analytics for iOS and Android apps
iOS
</>
Add an app to get started


Add Firebase to your web app
Copy and paste the snippet below at the bottom of your HTML, before other script tags.
<script src="https://www.gstatic.com/firebasejs/5.7.2/firebase.js"></script>
<script>
// Initialize Firebase
var config = {
apiKey: "AIzaSyCRiFBmgEfwKJXP7adeqd_VTpb0CPmHFyQ",
authDomain: "fir-withjs-37415.firebaseapp.com",
databaseURL: "https://fir-withjs-37415.firebaseio.com",
projectId: "fir-withjs-37415",
storageBucket: "fir-withjs-37415.appspot.com",
messagingSenderId: "140805408316"
};
firebase.initializeApp(config);
</script>
Copy
Check these resources to learn more about Firebase for web apps:
Get Started with Firebase for Web Apps
Firebase Web SDK API Reference
Firebase Web Samples


FirebaseWithJs
Go to docs
Database
Realtime Database
Data
Rules
Backups
Usage
https://fir-withjs-37415.firebaseio.com/
fir-withjs-37415
isOccupied: 1
isOccupied1: 0
isOccupied2: 0
isOccupied3: 1


DatcMap [C:\Users\Bianca\AndroidStudioProjects\DatcMap] - ...\app\src\debug\res\values\google_maps_api.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MapsActivity.java
google_maps_api.xml
app
activity_maps.xml
AndroidManifest.xml
<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&r=EB:00:3B:34:C4:0F:07:59:AC:06:6F:14:97:74:BD:B4:39:A9:B4:1
You can also add your credentials to an existing key, using these values:
Package name:
EB:00:3B:34:C4:0F:07:59:AC:06:6F:14:97:74:BD:B4:39:A9:B4:15
SHA-1 certificate fingerprint:
EB:00:3B:34:C4:0F:07:59:AC:06:6F:14:97:74:BD:B4:39:A9:B4:15
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">AIzaSyDKBMpOgFlo62DmESG0JY3xTRN7GA53vLw</string>
</resources>
Emulator: Process finished with exit code 0 (52 minutes ago)
1:39 PM
08-Jan-19


test [C:\Users\Bianca\AndroidStudioProjects\test] - test - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
build.gradle
MapsActivity.java
app
google_maps_api.xml
test
// Top-level build file where you can add configuration options common to all sub-projects/modules.
buildscript {
repositories {
google()
jcenter()
}
dependencies {
classpath 'com.android.tools.build:gradle:3.2.1'
// NOTE: Do not place your application dependencies here; they belong
// in the individual module build.gradle files
}
}
allprojects {
repositories {
google()
jcenter()
}
}
task clean(type: Delete) {
delete rootProject.buildDir
}
Emulator: Process finished with exit code 0 (56 minutes ago)
1:43 PM
08-Jan-19

4. Tehnologii folosite

4.2 Firebase

Cele mai multe aplicatii au nevoie de o baza de date pentru a stoca date de care programul are nevoie, precum setarile aplicatiei sau cel mai bun scor – in cazul unui joc. Pentru o aplicatie *Android* se pune problema folosirii unei baze de date locala, care va fi specifica fiecarui *device* – utila pentru datele care nu trebuie transferate sau corelate cu alte date sau folosirea unei baze de date care nu "traieste" pe dispozitivul unde e folosita, ci e comuna mai multor *device-uri* – "*off device database*".

Am ales *Google Firebase* din mai multe motive, printre care:

- Costul – Firebase este gratis, existand totusi un numar de proiecte pe care le poate sustine – Se mai pot sterge dintre ele, dar dureaza o perioada pana la finalizarea procesului;
- Este relative usor de folosit;
- Nu presupune utilizarea limbajului SQL pentru operatiile din baza de date, ci utilizeaza formatul JSON si ofera consola – *Google Firebase Console* – pentru manipularea datelor;
- Este sustinuta de Google, fiind bine documentata si destul de sigura.

Configurare Google Firebase:

Am creat un proiect Firebase (realtime database) nou in Google Firebase Console;

Am incercat folosirea lui intr-un proiect Android, lucru ce a condus la adaugarea unor dependinte in proiectul din Android, dar nu am reusit inserarea de date de acolo.

Mai multe detalii referitoare la modul in care am configurat Firebase am prezentat la implementarea worker-ului.

4.3 Android Studio

Android Studio este un IDE dezvoltat de Google si inlocuieste Eclipse Android Development Tools (ADT) ca IDE pentru aplicatiile native Android, permitand rularea codului scris in limbajul Java.

5. Bibliografie

1. https://stackoverflow.com/questions/18773598/creating-folders-inside-github-com-repo-without-using-git

2. https://console.firebase.google.com/?hl=ro

3. https://cloud.google.com/gcp/getting-started/