



App de Rodalies

*Aplicació Android xarxa de
trens de Rodalies*

Rodrigo Rafael Fernandez Sanchez
Maria del Carmen Busquets Frigola
Joan Esmandia Blanche
Christian Oliva Bonilla

Idea de negoci

El nostre projecte es basa en el desenvolupament d'una app d'Android basada en la xarxa de trens de Catalunya (Rodalies). La nostra idea de negoci es basa en crear una app que permeti consultar les rutes que podem utilitzar per anar d'una estació a una altre, informant de l'hora de sortida i arribada, poder consultar les línies, etc.

Cal dir que aquesta app actualment existeix, la podem trobar com "App de Rodalies", però ens va cridar l'atenció intentar realitzar una app semblant amb lleugeres diferències.

A part de mostrar la ruta entre dues estacions, també vam decidir que l'usuari pugui consultar les incidències actuals (que no acostumen a ser poques), informant de les línies que es veuen afectades i del retard.

Per últim, l'usuari també podrà posar-se en contacte amb nosaltres a través de la pestanya "Contacte", on entrant la informació indicada, podrà enviar un correu per informar de qualsevol "bug", recomanació, etc... respecte l'app.

Afegir que teníem en ment, realitzar una pestanya de localització que permetis observar la posició geogràfica de l'usuari en un mapa i indicar l'estació més propera a ell, però no ens ha estat possible degut a les complicacions que hem tingut al llarg del projecte.

Requeriments funcionals

- Donada dues estacions de la mateixa línia, l'app troba i mostra per pantalla l'hora de sortida i d'arribada.
- L'usuari pot consular les incidències que hi ha actualment, amb la informació del motiu i el temps de retard.
- L'usuari pot posar-se en contacte amb nosaltres a través de la pestanya "Contacte".
- L'usuari també pot consular les línies que existeixen (que tenim a la nostre BDD).
- Mostrem la informació de Rodalies, per tal de poder consultar qualsevol cosa directament amb l'empresa que realitza el servei (és important recordar que nosaltres fem una simulació de l'app de Rodalies).

Requeriments no funcionals

- Disponibilitat només per dispositius Android.
- Test realitzat amb Java JDK 1.8.
- La part del servidor està realitzada amb JavaEE.
- Per realitzar qualsevol modificació a la BDD s'ha de modificar el codi de l'app.
- L'app resulta molt visual i és molt senzilla d'utilitzar.

Documentació de l'API REST

GET

- **/ruta?origen=___&desti=___**

Aquest GET ens retorna una llista de totes les rutes possibles que hi ha entre l'estació amb id origen, i l'estació amb id destí (*els espais "___" representa que són el número identificador de les estacions origen i destí respectivament*). En cada ruta tenim la informació del número d'identificador de la ruta, la direcció (dreta o esquerra) cap a on va, el color, el tipus de tren^o i una llista dels trams, on en cada tram tenim el número d'identificador del tram, l'hora en que surt el tren, el número d'identificació de l'estació i el nom de l'estació.

- **/color**

Aquest GET ens retorna una llista de tots els colors que tenim guardats a la BDD. De cada color ens indica el seu número d'identificació i el seu nom.

- **/color/id**

Aquest GET ens retorna tota la informació d'un color (*"id" que trobem en la crida l'hem de substituir per el id del color que volem consultar la informació*). Podrem consultar l'id del color, el nom i el llistat de totes les estacions que formen part del color (id i nom de cada estació).

- **/estacio**

Aquest GET retorna una llista de totes les estacions guardades a la BDD. De cada estació ens indica el seu id i el seu nom.

- **/estació/id**

Aquest GET ens retorna tota la informació d'una estació (*"id" que trobem en la crida l'hem de substituir per el id de l'estació que volem consultar*). Podem consultar l'id de l'estació, el nom, la latitud i longitud on es troba, el telèfon de l'estació, i un llistat de tots els colors/línies que passen per aquesta estació (amb l'id i el nom de cada color).

- **/incidencia**

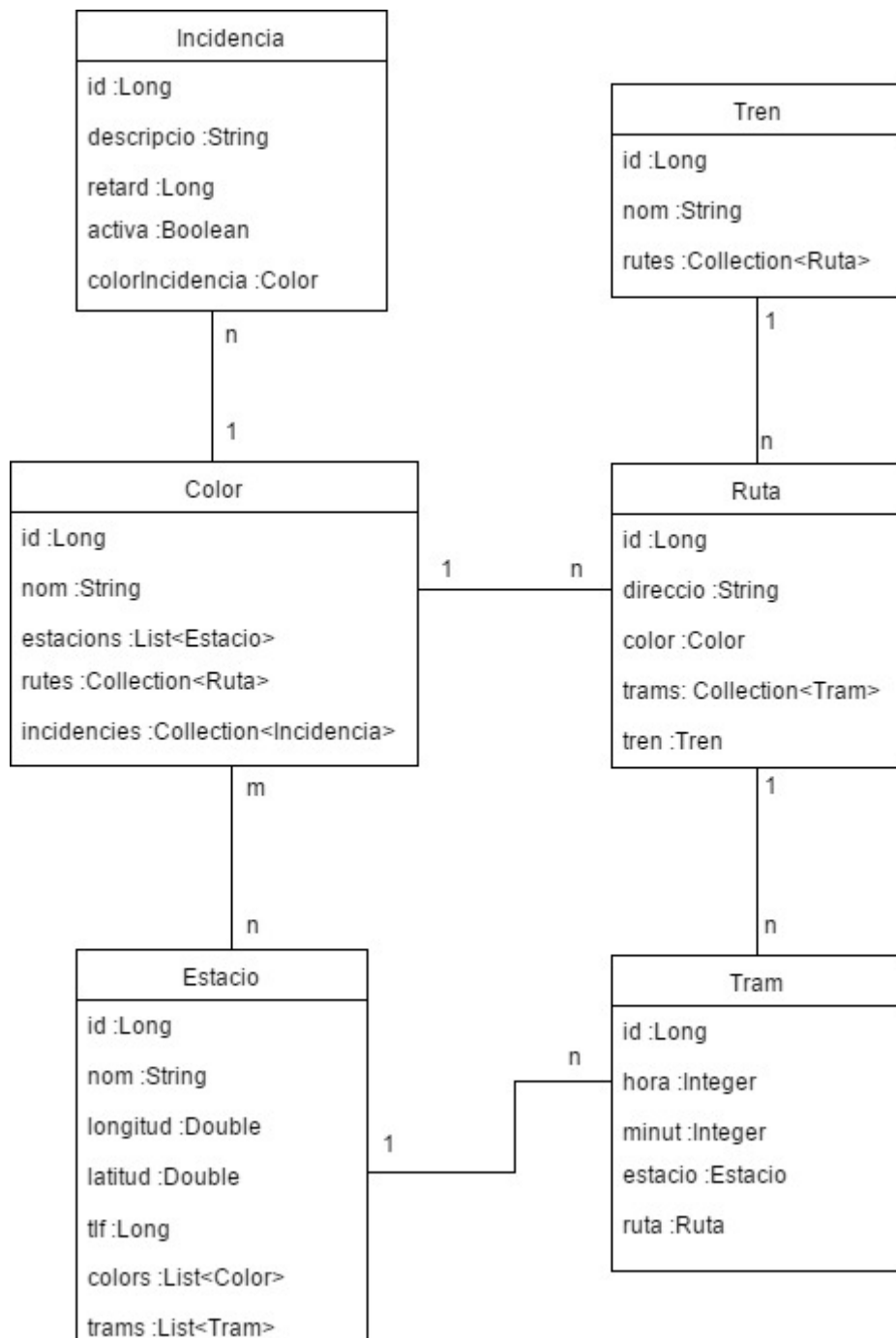
Aquest GET ens retorna una llista de totes les incidències que hi ha guardades a la BDD. De cada incidència trobem l'id, la descripció que descriu la incidència, el retard previst en minuts, una variable de tipus bool (*"true"* si la incidència està activa en aquest moment o *"false"* si actualment no ho està), i el color que afecta aquesta incidència (id i nom del color).

PUT

- **/comentari?email=emailtest&assumppte=assumptetest&text=texttest**

Aquest PUT ens guarda a la BDD un correu redactat per l'usuari des de la pestanya "Contacte".

Diagrama entitat relació de la BDD



URLs del repositoris GIT

Repositori GIT de la part servidor → <https://github.com/charry95/javaee>

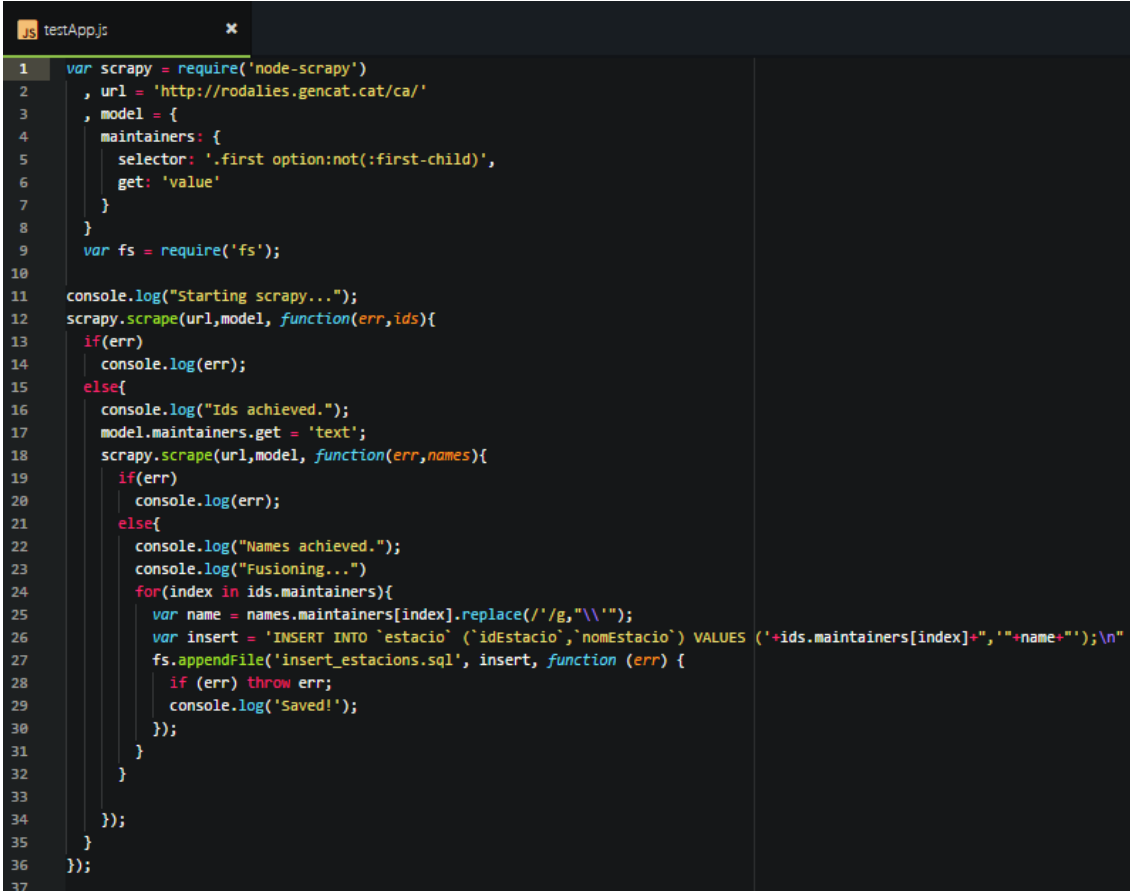
Repositori GIT de la part client (Android) → <https://github.com/Carmen08/CarmenPDS>

Extres que hem utilitzat

A més a més del codi de servidor, hem realitzat també un petit script amb node.js per tal d'obtenir les estacions de rodalies de Catalunya mitjançant el mètode scrapy.

Per fer-ho hem utilitzat el mòdul node-scrapy de node, el qual primer agafem tots els values de la web <http://rodalies.gencat.cat/ca/> que es troben sota el selector de css .first option: not(:first-child).

Aquest codi ara mateix ens genera un fitxer .sql, amb els inserts de cada estació on s'inserta el codi d'estació i el nom corresponent.



```
1 var scrapy = require('node-scrapy')
2   , url = 'http://rodalies.gencat.cat/ca/'
3   , model = {
4     maintainers: {
5       selector: '.first option: not(:first-child)',
6       get: 'value'
7     }
8   }
9   var fs = require('fs');
10
11 console.log("Starting scrapy...");
12 scrapy.scrape(url, model, function(err, ids){
13   if(err)
14     console.log(err);
15   else{
16     console.log("Ids achieved.");
17     model.maintainers.get = 'text';
18     scrapy.scrape(url, model, function(err, names){
19       if(err)
20         console.log(err);
21       else{
22         console.log("Names achieved.");
23         console.log("Fusioning...")
24         for(index in ids.maintainers){
25           var name = names.maintainers[index].replace(/'/g, "\\'");
26           var insert = 'INSERT INTO `estacio` (`idEstacio`,`nomEstacio`) VALUES ('+ids.maintainers[index]+'+', '"+name+"');\n";
27           fs.appendFile('insert_estacions.sql', insert, function (err) {
28             if (err) throw err;
29             console.log('Saved!');
30           });
31         }
32       }
33     });
34   }
35 }
36 });
37
```

Problemes e incidències

Volem destacar la quantitat de problemes que hem tingut per “omplir” la nostre BDD amb les dades de Rodalies, ja que al no poder accedir a cap BDD de Rodalies, totes les dades les hem tingut que entrar nosaltres a mà (o utilitzant un “scrappy” com hem fet amb les estacions). Aquest és el principal motiu pel qual no tenim totes les línies disponibles a la nostre app.

El problema més important respecte la part de servidor, és el tema de realitzar transbords. En el moment que hem de canviar de línia (o color) per arribar el nostre destí, no hem estat capaços de solucionar aquest problema. Hem modificat forces vegades el codi per tal d’adaptar-lo el màxim possible a poder realitzar els transbords (per exemple, hem creat una funció que ens retorna les permutacions mínimes de colors, necessàries per arribar de l’estació d’origen a l’estació destí), però finalment no ha estat possible degut a la dificultat que té.

Al principi ens va costar bastant acabar d'entendre el funcionament d'Android, per tant, les primeres setmanes, tot i que vam tocar coses, ens vam centrar sobretot en entendre com funcionava.

També vam tenir problemes amb el repositori, ja que durant els primers sprints, IDEA ens penjava els seus fitxers de configuració.

Per últim, un altre problema que ens va portar bastants maldecaps va ser quan vam ajuntar Android i servidor, ja que al principi no s'hi connectava. Fins que no vam veure que l'adreça base enlloc de localhost:8080 havia de ser 10.0.2.2:8080, vam estar-hi una bona estona (<https://developer.android.com/studio/run/emulator-networking.html>).

Aportació de cada membre del grup al treball

Hem dividit la feina en dos grups, la part de servidor (en la qual treballaven en Christian i en Joan), i la part client d'Android (en la qual treballaven la Carmen i en Rodrigo).

Part d'Android

Rodrigo: m'he encarregat de fer la part gràfica de la pantalla inicial i la de contacte. També he creat les funcions a la classe TodoApi per fer les crides al servidor.

Carmen: m'he encarregat de fer la part gràfica de la pantalla d'incidències i la de les línies (junt amb la pantalla de les estacions de cada línia). He implementat el retrofit.

Entre els dos hem implementat junts la interfície gràfica de les rutes.

Part servidor

Joan: entre en Cristian i jo ens hem repartit la feina de crear les classes, realitzar les relacions i crear les funcions corresponents als GETs i al PUT. També hem entrat les dades de Rodalies a la BDD.

Christian: entre en Joan i jo ens hem repartit la feina de crear les classes, realitzar les relacions i crear les funcions corresponents als GETs i al PUT. També hem entrat les dades de Rodalies a la BDD. He realitzat el codi scrappy.

Valoració percentual de cada membre

Rodrigo → 20%

Carmen → 30%

Christian → 25%

Joan → 25%