Web and Mobile Application Development

UNIVERSITÉ JEAN MONNET

Poulomi NANDY

Supervisor:
Prof. Guillaume Ehret

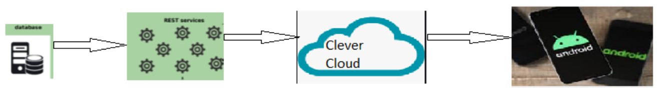
13 January 2021

Contents

1 Introduction

The main aim of the subject is to understand the web world. The server side as well as the client side.There are not only server and client side many other technologies or tools supporting web development that we have came across. We have used Gradle,GitHub repository in order to control the version of our code.In the server side we have used Spring boot which is a frame work of Java programming language.In the client side we have developed a android based mobile application. This is a UI, where user can see different services and also static data. We have done little bit of design. The data are getting fetched from in memory database which is H2. The code editer we have used in this mini project is IntelliJ. My services are deployed in Clever Cloud. We have used Swagger to define various verb of services like @GET, @DELETE, @POST,@PUT.My application is about rooms and the status of the room, the room name and id.

2 The overview



3 Server Side

In this part we have :

3.1 Database:

An application needs access to data, write data, update these data.

3.2 Springboot

This is Java based frame work we have used to create micro services.It is simply by adding dependency in gradle file.

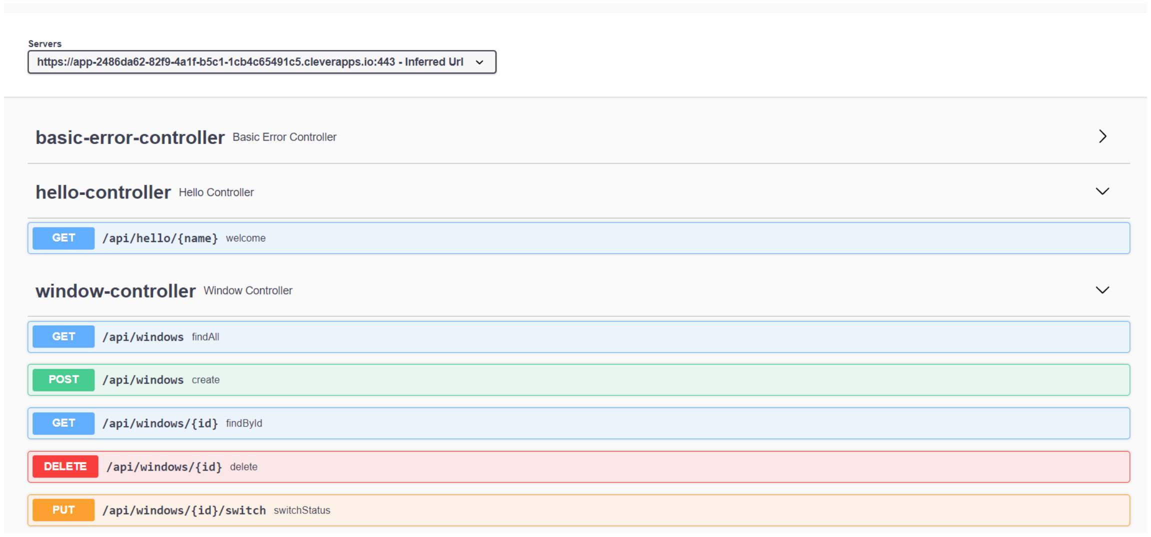
3.3 Dependency injection

Dependency injection is a programming technique that makes a class independent of its dependencies.That enables you to replace dependencies without changing the class that uses them. It also reduces the risk that you have to change a class just because one of its dependencies changed.

3.4 JUnit testing

In order to check if our results are matching we need to do this testing.We have used Mockito to simulate their behaviors. It is a popular mock framework which can be used in conjunction with JUnit.

In the server side we have the below services with the corresponding verb available in Swagger.



Servers

https://app-2486da62-82f9-4a1f-b5c1-1cb4c65491c5.cleverapps.io:443 - Inferred Url

basic-error-controller Basic Error Controller

hello-controller Hello Controller

GET /api/hello/{name} welcome

window-controller Window Controller

GET /api/windows findAll

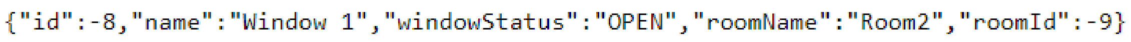
POST /api/windows create

GET /api/windows/{id} findById

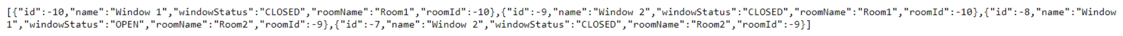
DELETE /api/windows/{id} delete

PUT /api/windows/{id}/switch switchStatus

We can check the json format from Swagger. Below are few snippet.



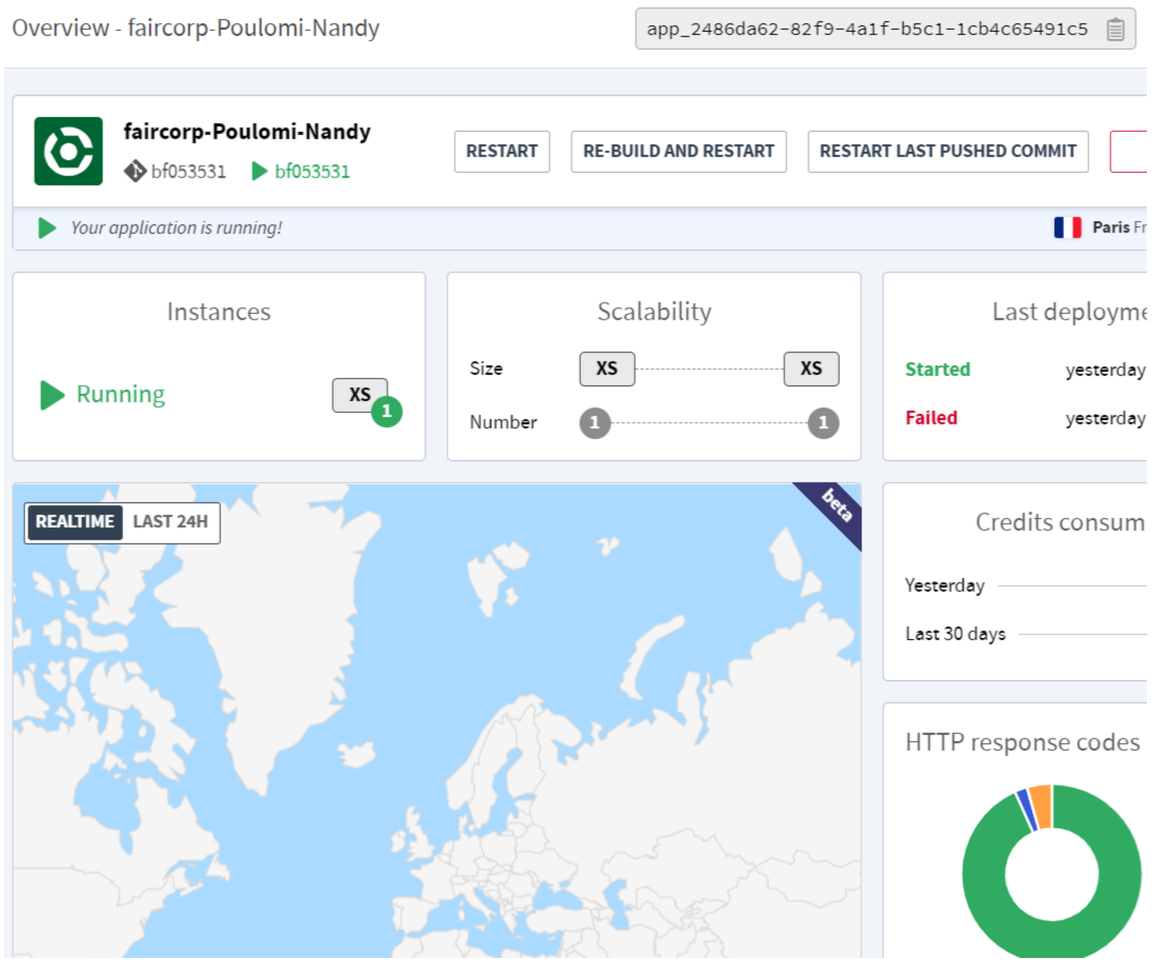
```
{"id":-8,"name":"Window 1","windowStatus":"OPEN","roomName":"Room2","roomId":-9}
```



```
[{"id":-10,"name":"Window 1","windowStatus":"CLOSED","roomName":"Room1","roomId":-10},{"id":-9,"name":"Window 2","windowStatus":"CLOSED","roomName":"Room1","roomId":-10},{"id":-8,"name":"Window
1","windowStatus":"OPEN","roomName":"Room2","roomId":-9},{"id":-7,"name":"Window 2","windowStatus":"CLOSED","roomName":"Room2","roomId":-9}]
```

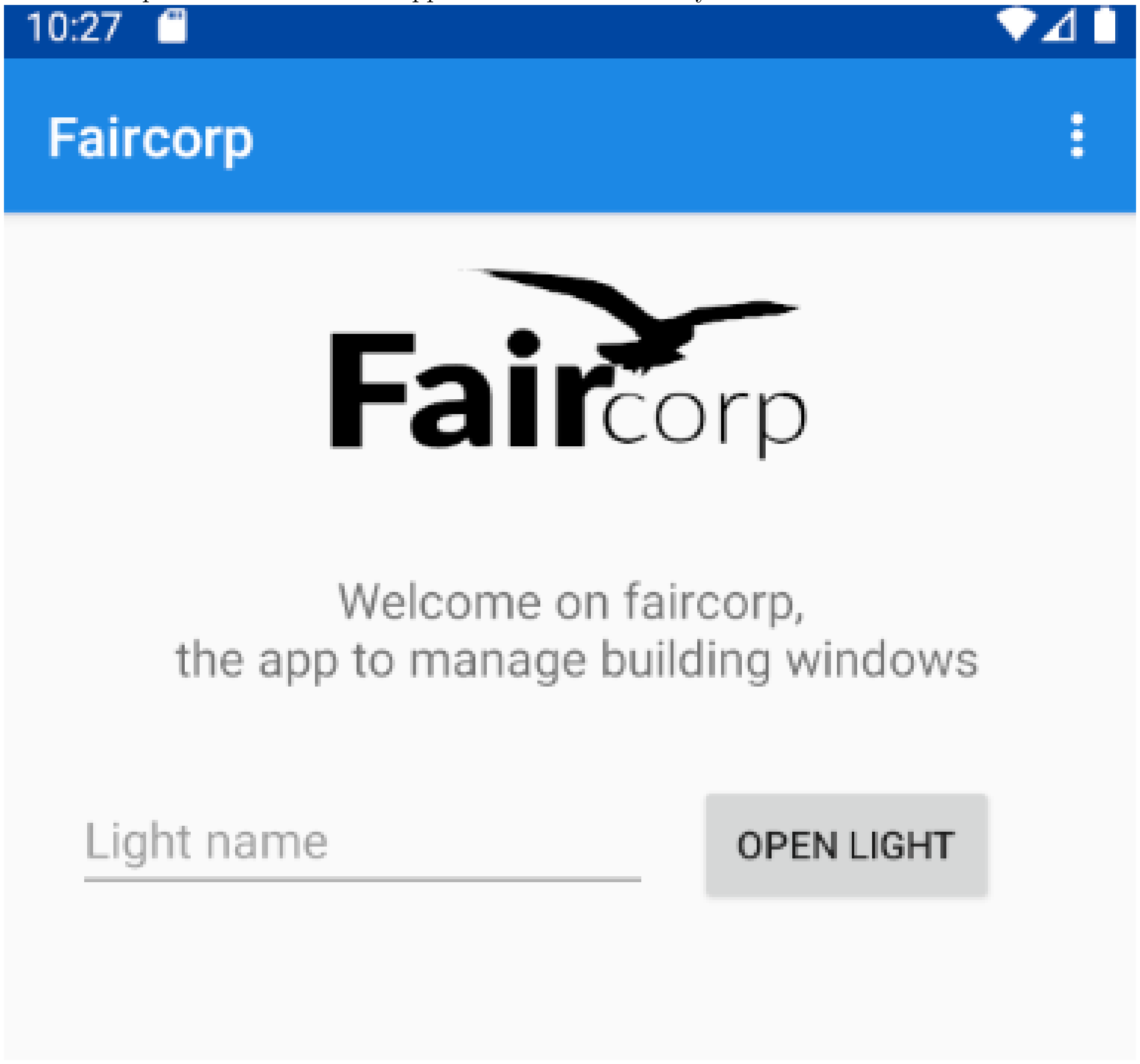
4 Deployment

We have deployed the API in Clever Cloud.

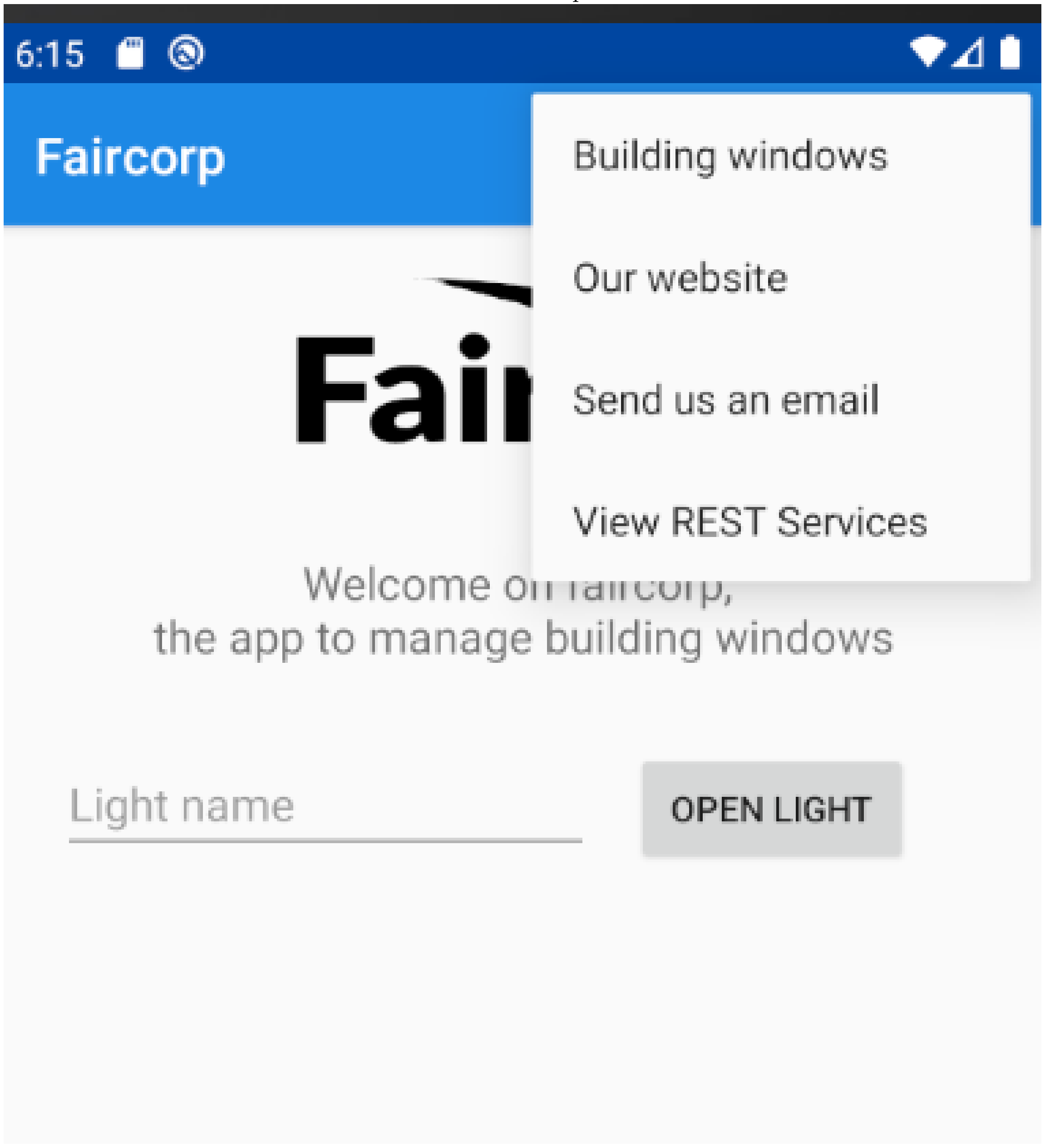


5 Client Side

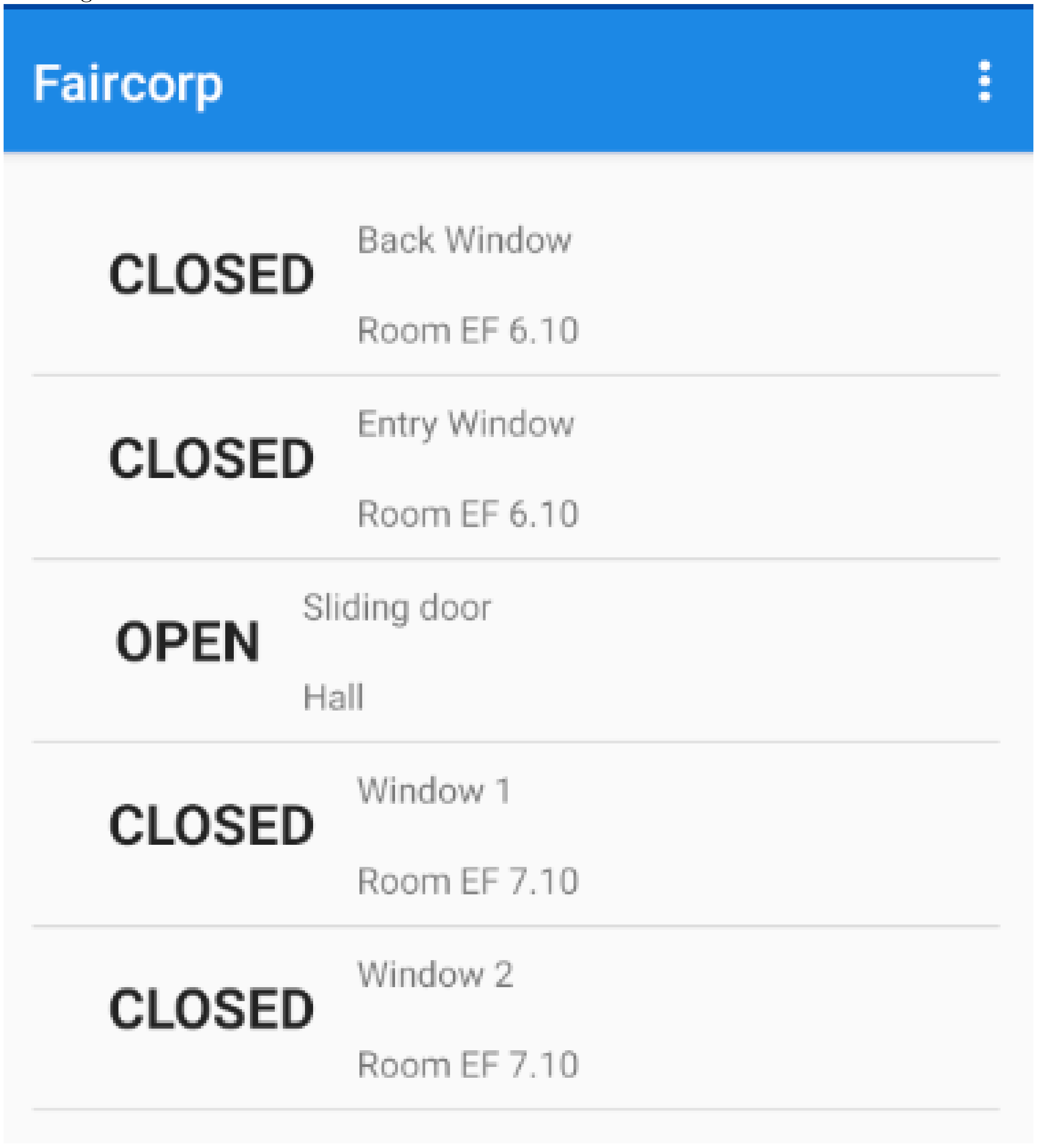
We have developed an android based app here.The start activity:



When we click on the menus we will have the below options:



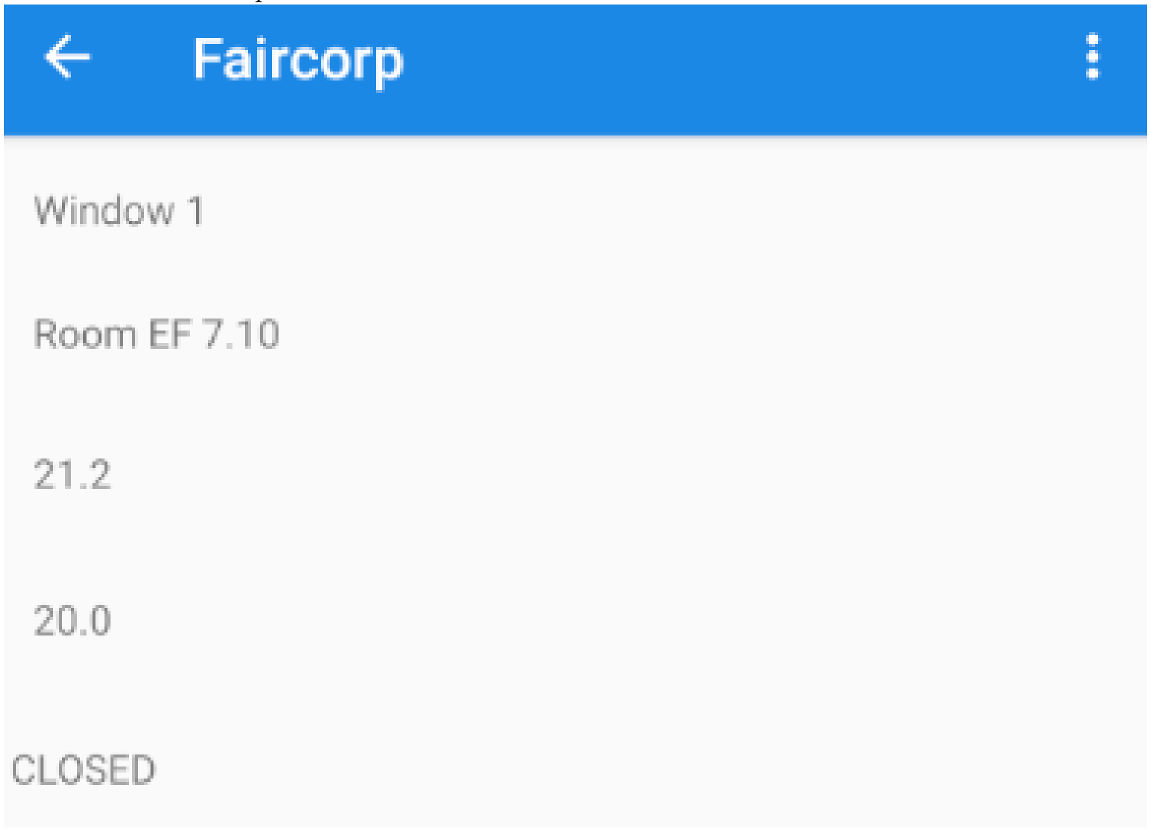
Building Windows is the static data which will be viewed in List form:



Faircorp

CLOSED	Back Window Room EF 6.10
CLOSED	Entry Window Room EF 6.10
OPEN	Sliding door Hall
CLOSED	Window 1 Room EF 7.10
CLOSED	Window 2 Room EF 7.10

We can click on the option and can see the details of the window:



Faircorp

Window 1

Room EF 7.10

21.2

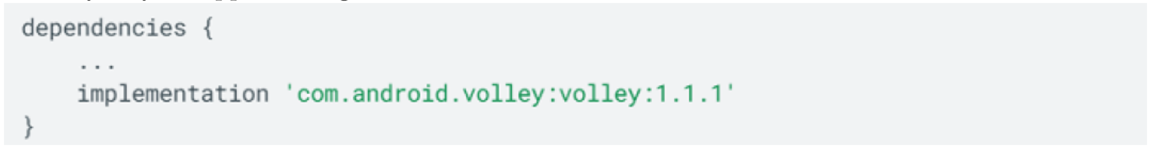
20.0

CLOSED

The other option like email and web site are used to send email from device and to visit the website.

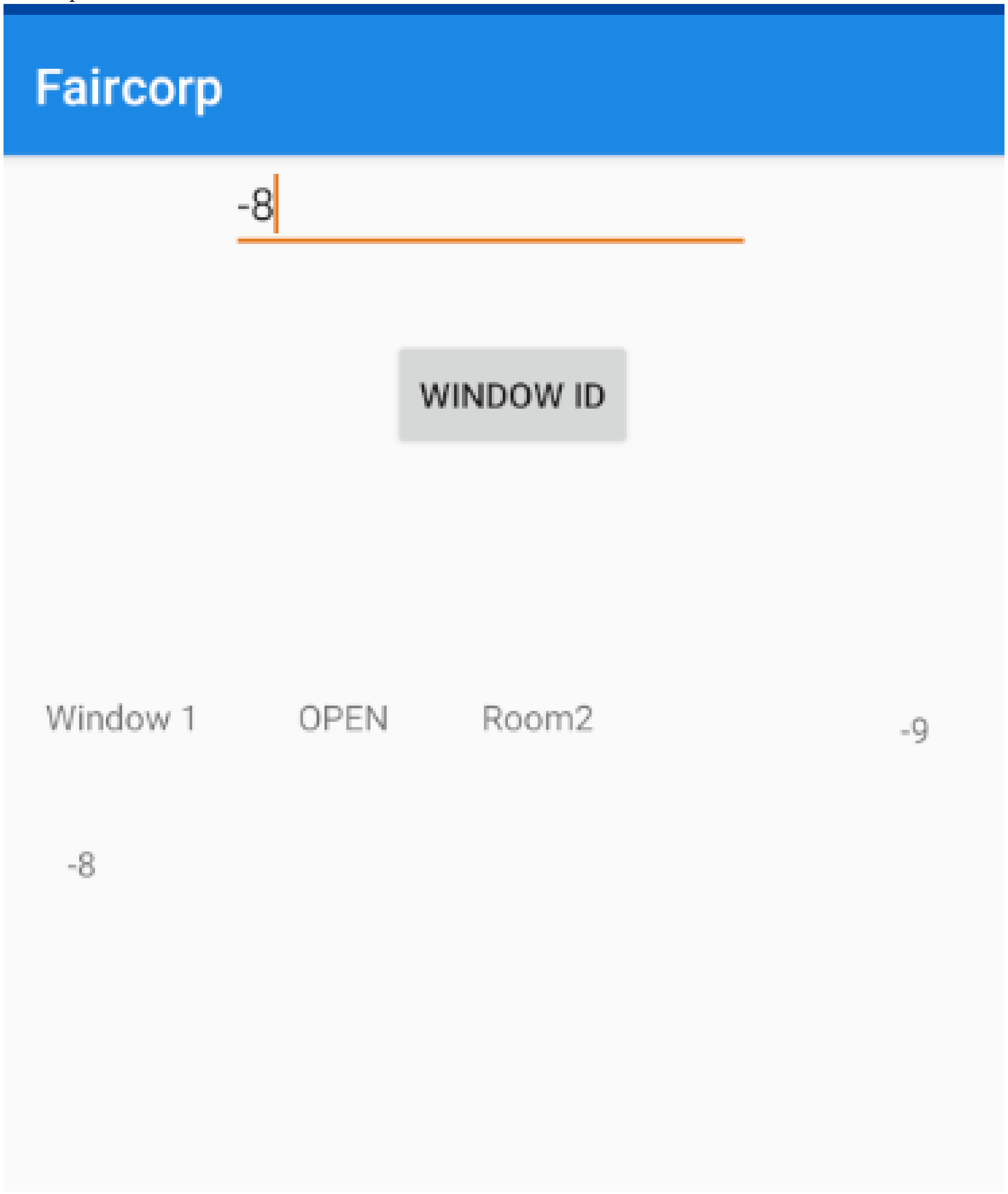
5.1 REST Api call

I have used here Volley library developed by google. Volley offers the following benefits: Automatic scheduling of network requests. Multiple concurrent network connections. Trans- parent disk and memory response caching with standard HTTP cache coherence. Support for request prioritization. Cancellation request API. You can cancel a single request, or you can set blocks or scopes of requests to cancel. Ease of customization, for example, for retry and backoff. Strong ordering that makes it easy to correctly populate our UI with data fetched asynchronously from the network. Debugging and tracing tools. Volley excels at RPC-type operations used to populate a UI, such as fetching a page of search results as structured data. It integrates easily with any protocol and comes out of the box with support for raw strings, images, and JSON. By providing built-in support for the features we need, Volley frees from writing boilerplate code and allows to concentrate on the logic that is specific to the app.The easiest way to add Volley to your project is to add the following dependency to your app's build.gradle file:



```
dependencies {
    ...
    implementation 'com.android.volley:volley:1.1.1'
}
```

The Api UI is :



Window 1 OPEN Room2 -9

-8

6 Conclusion

In this assignment we have developed a Client-Server architecture which involved different technologies and different programming languages. Basically in my report I have tried to showcase all the web development technologies that I have used in this assignment. All the features are working on the server side. But, in client side I have called @GET only as of now. All the codes can be found in the git hub repository.