Firebase Workshop Session#2 March-2020

Firebase workshop by Cristian Arce & Alfredo Bonilla

Session #2

Firebase Databases

Step 1: Navigate to https://firebase.google.com/, look for the Database options, and access rules for both **Cloud Firestore** and **Realtime Database**

When the project was created, the selected schema was **production**, so, the database can not be accessed without authentication; therefore it is required to add the authentication for **read/write**

Cloud Firestore

Override the current rules using:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Realtime Database

Override the current rules using:

```
// These rules grant access to a node matching the
authenticated
// user's ID from the Firebase auth token
{
  "rules": {
    "user": {
      ".read": "auth.uid != null",
      ".write": "auth.uid != null"
    }
  }
}
```

Step 2: Clone the base project from repo in a folder

```
$ git clone git@github.com:cristianarceGL/firebase-
workshop.git

$ cd firebase-workshop

$ git checkout Session2-Base

$ git pull
```

After pulling the repo, it is mandatory to install all dependencies and run

the app

```
$ npm install && npm start
```

After running this, open the browser using the url **http://localhost:4200/**, a login page must be displayed, that's the one we will be modifying from now

Step 3: Create a folder **src/data** and create a json file **firebase-workshop-realtime.json**, and add this content:

```
{
    "user": {
        "YmiEtwanKzfGImufwGLw": {
            "address": "San José",
            "cellphone": 50688881111,
            "displayName": "Angular User",
            "email": "angular@angular.com",
            "phoneNumber": 50688881111,
            "photoURL": "",
            "uid": "fb-uid-Ng",
            "id": "YmiEtwanKzfGImufwGLw"
        },
        "hWK2l08gjkyqhyH37zsmrw": {
            "address": "Alajuela",
            "cellphone": 50688882222,
            "displayName": "React User",
            "email": "react@react.com",
            "phoneNumber": 50688882222,
```

```
            "photoURL": "",
            "uid": "fb-uid-React",
            "id": "hWK2l08gjkyqhyH37zsmrw"
        },
        "6Z1hUjQDKkKvdhZhts0": {
            "address": "Cartago",
            "cellphone": 50688883333,
            "displayName": "JQuery User",
            "email": "jquery@jquery.com",
            "phoneNumber": 50688883333,
            "photoURL": "",
            "uid": "fb-uid-Jquery",
            "id": "6Z1hUjQDKkKvdhZhts0"
        },
        "UtrLZibdgUK5RV": {
            "address": "Heredia",
            "cellphone": 50688884444,
            "displayName": "VueJs User",
            "email": "vue@vue.com",
            "phoneNumber": 50688884444,
            "photoURL": "",
            "uid": "fb-uid-Vue",
            "id": "UtrLZibdgUK5RV"
        }
    }
}
```

Angular Project

It is necessary now to update the ***admin-user.component.ts*** file with the firebase references

Step 1: First, a new import for the user service at the top of the component

```
...

import { UserService } from './admin-user.service';

...
```

Step 2: Then the constructor is overridden for including the new user service, which has the access to Firebase modules for accessing the database

```
...

constructor(private userService: UserService, public
dialog: MatDialog, private subService:
SubscriptionService) {}

...
```

Step 3: It is required to override the ngOnInit, for getting the list of user from user service, and that response will fill up the datasource:

```
public ngOnInit(): void {
    this.userService
        .getUserList()

 .pipe(takeUntil(this.subService.unsubscribe$))
        .subscribe(data => {
            const users = data.map(element => {
                return this.convertToUser(element);
            });
            this.dataSource = new
MatTableDataSource(users);
        });
}
```

Step 4: Now that the reference to Firebase is alive, we can execute CRUD operations; for this workshop proposes, we are overriding also **update** and **delete** as follow:

```
public updateUser(user: User) {
    this.userService.updateUser(user);
}

public deleteUser(user: User): void {
    this.userService.deleteUser(user.id);
}
```

Step 5: The steps above would cause an error since the user service is not defined yet, so in **src/features/admin-user** a new file is created as **admin-user.service.ts**, add it this content:

```
import { Observable } from 'rxjs';
import { Injectable } from '@angular/core';
import { takeUntil } from 'rxjs/operators';
import { AngularFirestore } from
'@angular/fire/firestore';

import { User } from '@app/features/models/user';
import { AngularFireDatabase } from
'@angular/fire/database';
import { SubscriptionService } from
'@app/features/firebase/subscription.service';

@Injectable({
providedIn: 'root',

})
export class UserService {

    constructor(private firestore: AngularFirestore,
private realtime: AngularFireDatabase, private
subService: SubscriptionService) {}

    // Create
    public createUser(user: User) {
        return this.firestore
        .collection('user')
        .doc(user.id)
        .set(user);
    }
```

```
    // Get Single
    public getUser(id: string): Observable<any> {
        return
this.firestore.doc(`user/${id}`).snapshotChanges();
    }

    // Get List
    public getUserList(): Observable<any> {
        this.realtime
            .list(`user`)
            .valueChanges()

 .pipe(takeUntil(this.subService.unsubscribe$))
            .subscribe(realtimeUsers =>
realtimeUsers.map(user => this.createUser(user as
User)));
        return
this.firestore.collection('user').snapshotChanges();
    }

    // Update
    public updateUser(user: User): Promise<void> {
        return
this.firestore.doc(`user/${user.id}`).update({
            uid: user.uid,
            displayName: user.displayName,
            email: user.email,
            address: user.address,
            cellphone: user.cellphone,
```

```
        });
    }

    // Delete
    public deleteUser(id: string): Promise<void> {
        return
this.firestore.doc(`user/${id}`).delete();
    }
}
```