Firebase

Instalação

- **Instalar Firebase e AngularFire**:

```
npm install firebase @angular/fire
```

- **Configurar o Firebase**: No Console do Firebase, crie um projeto e ative o Realtime Database em modo de teste.
- **Adicionar Configurações do Firebase**: No arquivo `src/environments/environment.ts`, configure as chaves do Firebase, que podem ser encontradas na seção "Configurações do Projeto" no Console do Firebase:

```
export const environment = {
  production: false,
  firebaseConfig: {
    apiKey: "YOUR_API_KEY",
    authDomain: "YOUR_AUTH_DOMAIN",
    databaseURL: "YOUR_DATABASE_URL",
    projectId: "YOUR_PROJECT_ID",
    storageBucket: "YOUR_STORAGE_BUCKET",
    messagingSenderId: "YOUR_MESSAGING_SENDER_ID",
    appId: "YOUR_APP_ID",
  },
};
```

- **Configurar Firebase no AppModule**: No arquivo `src/app/app.module.ts`, importe e configure o AngularFireModule e o AngularFireDatabaseModule:

```
import { AngularFireModule } from '@angular/fire/compat';
import { AngularFireDatabaseModule } from '@angular/fire/compat/database';
import { environment } from '../environments/environment';

@NgModule({
  imports: [
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireDatabaseModule,
  ],
})
export class AppModule {}
```

Serviço de CRUD para Firebase Realtime Database

Com as dependências do Firebase disponíveis no projeto, iremos criar um novo serviço para realizar as operações básicas de criação/listagem, atualização e exclusão de registros. Neste examplo, o nome será CrudDatabase

1. Criar o Serviço de CRUD

No diretório `src/app/services/` , crie um novo arquivo chamado `crud.service.ts` .

```
import { Injectable } from '@angular/core';
import { AngularFireDatabase } from '@angular/fire/compat/database';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root',
})
export class CrudService {
  private dbPath = '';

  constructor(private db: AngularFireDatabase) {}


  withPath(path:string){
    this.dbPath = path
    return this
  }


  createItem(item: any): void {
    this.db.list(this.dbPath).push(item);
  }


  getItems(): Observable<any[]> {
    return this.db.list(this.dbPath).snapshotChanges();
  }


  updateItem(key: string, value: any): Promise<void> {
    return this.db.list(this.dbPath).update(key, value);
  }


  deleteItem(key: string): Promise<void> {
    return this.db.list(this.dbPath).remove(key);
  }


  deleteAll(): Promise<void> {
    return this.db.list(this.dbPath).remove();
```

```
  }
}
```

Utilizando o serviço de CRUD em componentes

Como qualquer outro serviço, basta declarar como um atributo privado do construtor que o Angular irá injetar uma instância do mesmo para uso. Veja o exemplo.

```
import { Component, OnInit } from '@angular/core';
import { CrudService } from '../services/crud.service';

@Component({
  selector: 'app-home',
  templateUrl: './home.page.html',
  styleUrls: ['./home.page.scss'],
})
export class HomePage implements OnInit {
  items: any[] = [];
  newItem: string = '';

  constructor(private crudService: CrudService) {}

  ngOnInit() {
    this.getItems();
  }


  createItem() {
    if (this.newItem) {
      this.crudService.createItem({ name: this.newItem });
      this.newItem = '';
  }


  getItems() {
    this.crudService.getItems().subscribe((data) => {
      this.items = data.map((e) => ({
        key: e.key,
        ...e.payload.val(),
      }));
    });
  }


  updateItem(key: string, newName: string) {
    this.crudService.updateItem(key, { name: newName });
  }


  deleteItem(key: string) {
    this.crudService.deleteItem(key);
```

```
  }


  deleteAll() {
    this.crudService.deleteAll();
  }
}
```