**Section12 Authentication in Angular**

**Lesson01 How Authentication Works**

**Notes: -**

**1-the client send authentication data into the server to validate the authentication token is valid or not**

**2-client and server is decoupled from each other so its communicate with each other through http request and each time its send auth data and server send token if its valid**

**(The client side store the token into storage data such as local storage and each time the client send token instead of the auth data and the server validate the token validation)**

**Store Token**

**Storage**

**Client**

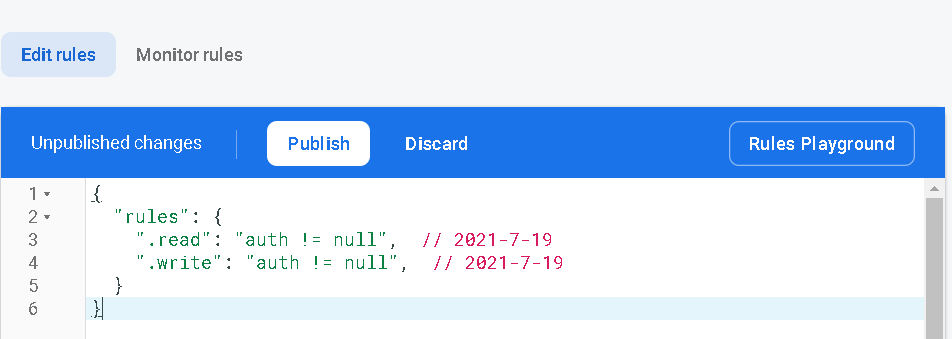
**Store Token is sent to authorize subsequent request**

**Send Auth Data**

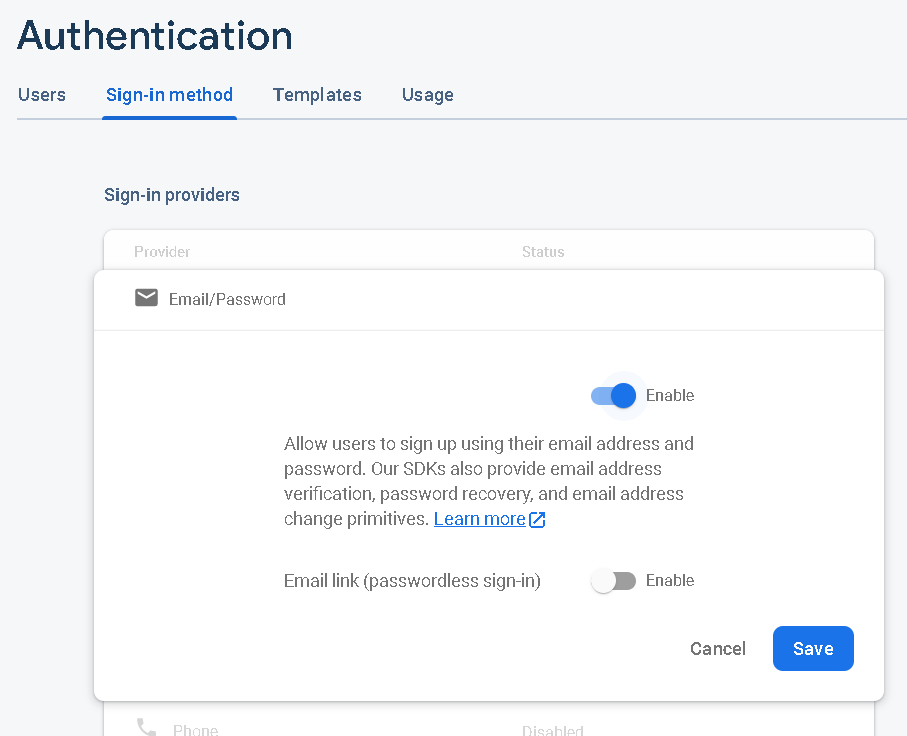
**Token**

**Client**

**3-on firebase > Realtime Database > make sure that you update Rules as below to allow only authenticate data as below**



**4-on Firebase > Authentication > Email/Password > enable it**



**With the below URL https://firebase.google.com/docs/reference/rest/auth**

**You can see that it will show the API documentation about handling API request to firebase and handling error messages**

**Tap operator is used to run some code without break observables code**

**5-there are different ways to store the token in order to use it in request**

**A-by using Subject Behavior**

**B-by using local storage / session storage with interceptor**

**Subject Behavior with local storage: -**

**//on the Auth Service we using the Behavior Subject instead of Subject in order to access to its //value without using subscribe , direct access**

**import { HttpClient, HttpErrorResponse } from '@angular/common/http';**

**import { Injectable } from '@angular/core';**

**import { BehaviorSubject, Observable, Subject } from 'rxjs';**

**import { catchError, tap } from 'rxjs/operators';**

**import { AuthResponseData } from '../models/dto/auth-resposne-data';**

**import { Signin } from '../models/signin';**

**import {throwError} from 'rxjs';**

**import { User } from '../models/user';**

**@Injectable({providedIn: 'root'})**

**export class AuthService {**

**apiKey: string = 'AIzaSyDlndkAleImIpkEPIGT\_rZK0RM-KaNjTW8';**

**constructor(private http: HttpClient) {}**

**user = new BehaviorSubject<User>(null);**

**signup(usrName: string, pwd: string): Observable<AuthResponseData> {**

**let signIn = new Signin(usrName, pwd, true);**

**return this.http**

**.post<AuthResponseData>(**

**`https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=${this.apiKey}`,signIn)**

**.pipe(catchError(this.handleError),tap(resData => {this.handleAuthenticate(resData)}));}**

**signin(usrName: string, pwd: string): Observable<AuthResponseData> {**

**let signIn = new Signin(usrName, pwd, true);**

**return this.http**

**.post<AuthResponseData>(**

**`https://identitytoolkit.googleapis.com/v1/accounts:signInWithPassword?key=${this.apiKey}`,**

**signIn).pipe(catchError(this.handleError),tap(resData => {this.handleAuthenticate(resData)}));}**

**private handleAuthenticate(resData:AuthResponseData){**

**//we get the time in 1970 + get time differences between now and 1970 in ms \* 1000**

**const expirationDate = new Date(new Date().getTime() + +resData.expireIn\* 1000);**

**const usr = new User(resData.email,resData.localId,resData.idToken,expirationDate);**

**this.user.next(usr);}**

**private handleError(errorRes: HttpErrorResponse) {**

**let errorMsg: string = null;**

**if (!errorRes.error || !errorRes.error.error) {**

**return throwError(errorMsg);}**

**switch (errorRes.error.error.message) {**

**case 'EMAIL\_EXISTS':**

**errorMsg = 'This email already exists';**

**break;**

**case 'EMAIL\_NOT\_FOUND':**

**errorMsg = 'This email not found';**

**break;**

**case 'INVALID\_PASSWORD':**

**errorMsg = 'Invalid Password';**

**break;**

**default:**

**errorMsg = 'An unknow error occured!';**

**break;}**

**return throwError(errorMsg);}}**

**fetchRecipes(): Observable<Recipe[]> {**

**//we get the first element on the behavior subject and using it as parameter on the request URL**

**return this.authService.user.pipe(**

**take(1),**

**exhaustMap((usr) => {**

**return this.http.get<Recipe[]>(**

**'https://enbehpro-default-rtdb.europe-west1.firebasedatabase.app/recipes.json',{**

**params:new HttpParams().set('auth',usr.token)});}),**

**map((recipes) => {**

**return recipes.map((recipe) => {**

**return {...recipe,ingredients: recipe.ingredients ? recipe.ingredients : []};});}),**

**//tap is used to execute some code without effect on the response return to the component**

**tap((recipes) => {this.recipeService.setRecipes(recipes);}));}**