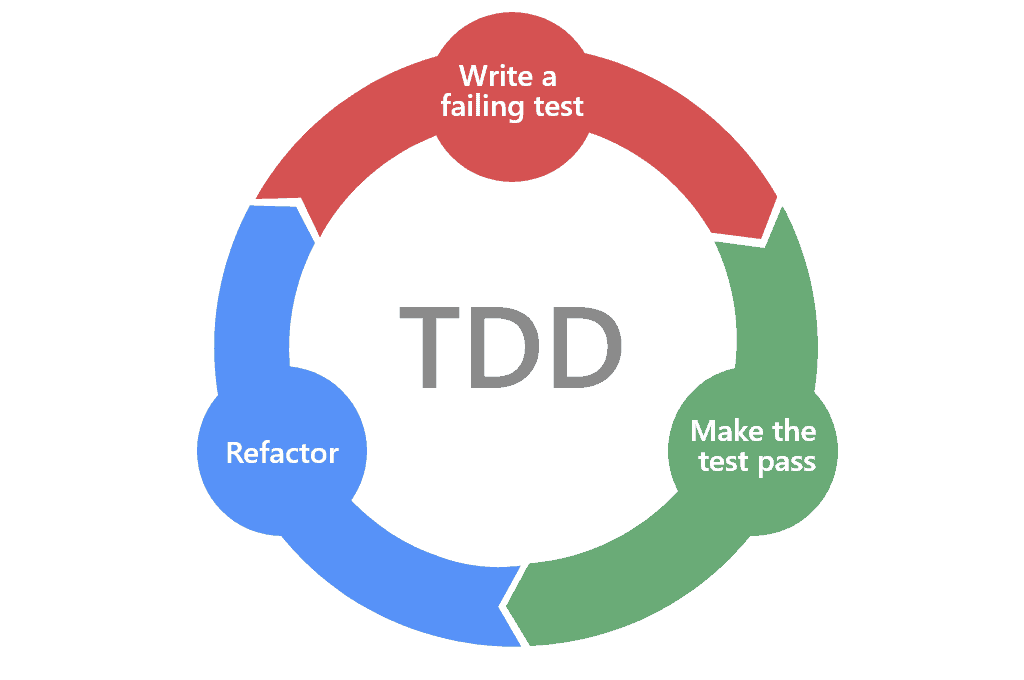
TDD



Pablo Manuel Puch García

**Índice**

* Código del método…………………………………………3
* Código de la Prueba……………………………………….5
* Capturas de las Pruebas…………………………………7
* Enlaces de documentos………………………………….8

**Código del método**

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

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

import { Observable, of } from 'rxjs';

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

import { Project } from '../models/project';

import { AuthService } from '../auth/auth.service';

import { Storage } from '@ionic/storage';

@Injectable({

  providedIn: 'root'

})

export class ProjectsService {

  token="";

  httpOptions = {

    headers: new HttpHeaders({

      'Content-Type': 'application/json',

      'Authorization': `Bearer ${this.token}`

    })

  };

  httpOptionsUsingUrlEncoded = {

    headers: new HttpHeaders({

      'Content-Type': 'application/x-www-form-urlencoded',

      'Authorization': `Bearer ${this.token}`

    })

  };

  endpoint = "http://localhost:8000/api/projects";

  constructor(private httpClient: HttpClient) {

    // this.storage.create()

    // this.storage.get("token").then((token) => {

    //   this.httpOptions.headers = new HttpHeaders({

    //     'Content-Type': 'application/json',

    //     'Authorization': `Bearer ${token}`

    //   });

    // })

  }

  getProjects(): Observable<Project[]> {

    // console.log(this.token);

    console.log(this.httpOptions.headers)

    return this.httpClient.get<Project[]>(this.endpoint, this.httpOptions).pipe(

      tap(\_ => console.log("Project retrieved")),

      catchError(this.handleError<Project[]>("Get project", []))

    );

  }

  getProjectById(id: number): Observable<Project> {

    return this.httpClient.get<Project>(this.endpoint + "/" + id, this.httpOptions).pipe(

      tap(\_ => console.log(`Project retrieved: ${id}`)),

      catchError(this.handleError<Project>(`Get project id=${id}`))

    );

  }

  createProjectUsingJSON(project: Project): Observable<Project> {

    return this.httpClient.post<Project>(this.endpoint, JSON.stringify(project), this.httpOptions).pipe(

      catchError(this.handleError<Project>("Error ocurred"))

    );

  }

  updateProject(idProject, project: Project): Observable<any> {

    let bodyEncoded = new URLSearchParams();

    bodyEncoded.append("season\_id", project.season\_id.toString());

    bodyEncoded.append("name", project.name);

    bodyEncoded.append("starDate", project.starDate.toString());

    bodyEncoded.append("endDate", project.endDate.toString());

    const body = bodyEncoded.toString();

    return this.httpClient.put(this.endpoint + "/" + idProject, body, this.httpOptionsUsingUrlEncoded);

  }

  deleteProject(idProject: number): Observable<Project[]> {

    return this.httpClient.delete<Project[]>(this.endpoint + "/" + idProject, this.httpOptions);

  }

  handleError<T>(operation = 'operation', result?: T) {

    return (error: any): Observable<T> => {

      console.error(error);

      console.log(`${operation} failed: ${error.message}`);

      return of(result as T);

    };

  }

}

**Código de la Prueba**

import { of } from 'rxjs';

import { Project } from '../models/project';

import { ProjectsService } from './projects.service';

import { Storage } from '@ionic/storage';

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

describe('ProjectsService', () => {

  let service: ProjectsService;

  let httpClientSpy: {get: jasmine.Spy};

  beforeEach(() => {

    httpClientSpy = jasmine.createSpyObj('HttpClient', ['get']);

    service = new ProjectsService(httpClientSpy as any);

  });

  fit('porjects-get', () => {

    const mockProjects: Project[] = [

      {

          id: 1,

          season\_id: 3,

          name: "azul",

          starDate: new Date("2021-11-21"),

          endDate: new Date("2021-11-21"),

          season: {

              id: 3,

              name: "primavera",

              starDate: new Date("2021-11-01"),

              endDate: new Date("2021-11-07"),

              noteSeason: "adios",

          }

      },

      {

          id: 7,

          season\_id: 2,

          name: "verde",

          starDate: new Date("2021-10-15"),

          endDate: new Date("2021-10-21"),

          season: {

              id: 2,

              name: "varano",

              starDate: new Date("2021-11-10"),

              endDate: new Date("2021-11-17"),

              noteSeason: "buenos dias",

          }

      }

  ];

  httpClientSpy.get.and.returnValue(of(mockProjects));

  });

  fit('should return an error when the server returns a 404', (done: DoneFn) => {

    const errorResponse = new HttpErrorResponse({

      error: 'test 404 error',

      status: 404, statusText: 'Not Found'

    });

    httpClientSpy.get.and.returnValue(of(errorResponse));

    service.getProjects().subscribe(

      project => done.fail('expected an error, not project'),

      error  => {

        expect(error.message).toContain('test 404 error');

        done();

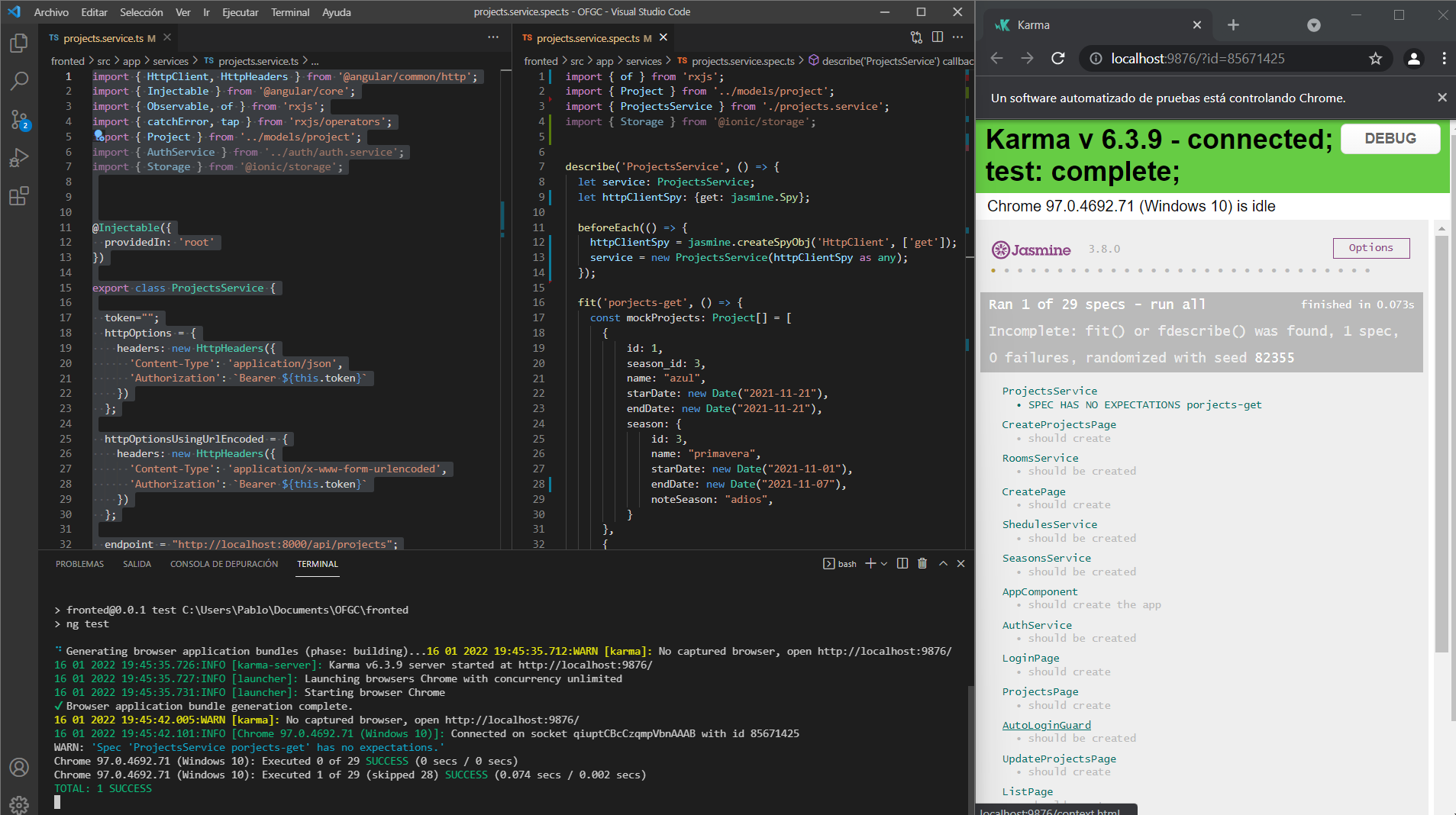
      }

    );

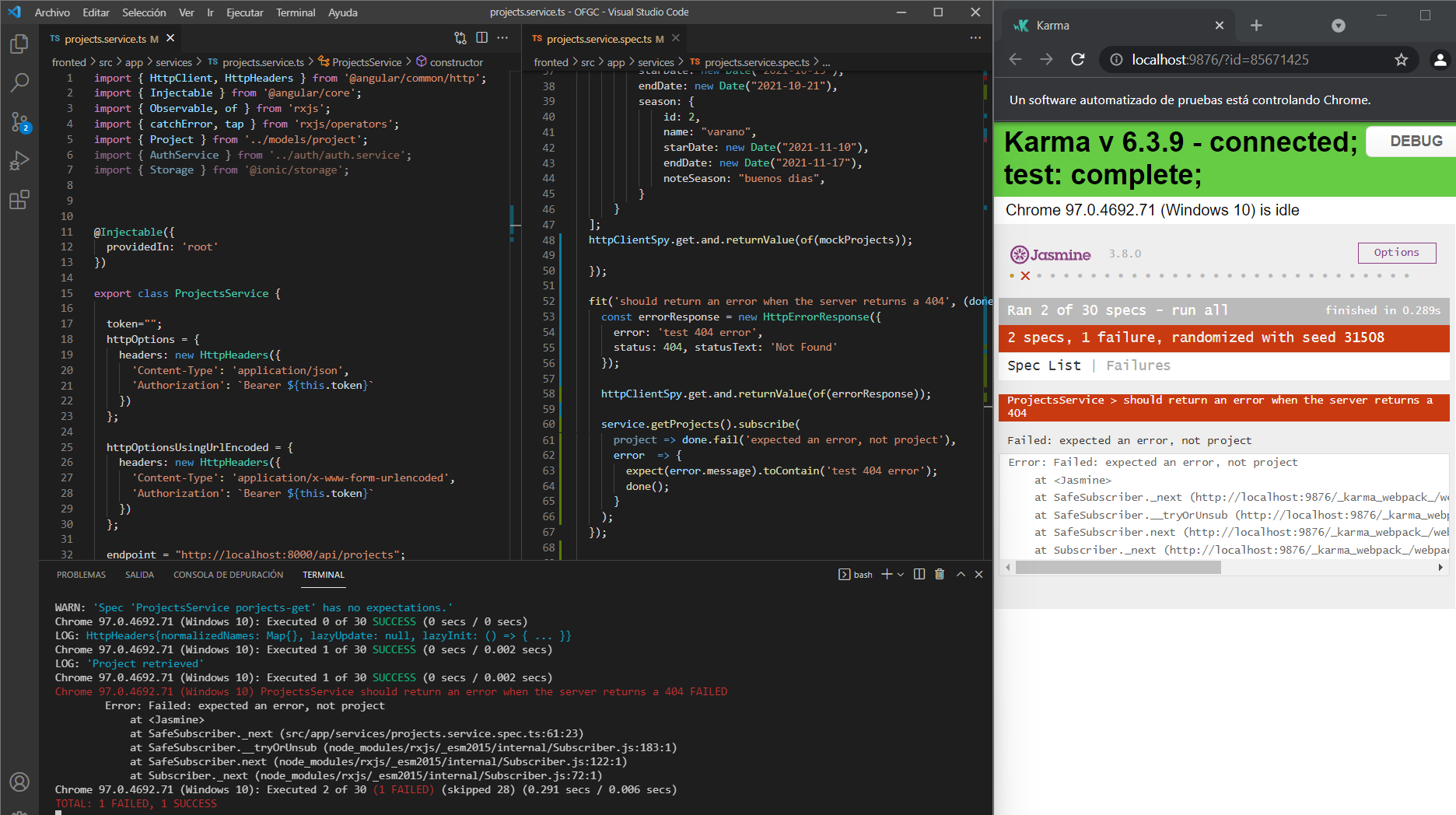
  });

});

**Capturas de las Pruebas**



**Prueba no falla**



**Enlaces**

<https://angular.io/guide/testing-services>

<https://www.youtube.com/watch?v=KsmXSJcfasU>