

# Configuración prueba Analista Desarrollador

[Prerequisitos](#)

[Instalación](#)

[Backend](#)

[Frontend](#)

[Uso de Swagger](#)

[Funcionalidades en el frontend](#)

## Prerequisitos

Se requiere tener instalado

- Java JDK 17
- MySQL
- NPM y Angular 17

Para la instalación de Angular, se debe ejecutar el siguiente comando en la consola de nodejs con permisos de administrador:

```
npm install -g @angular/cli
```

Se puede validar que quedó instalado con el comando `ng version`

## Instalación

1. Clonar el repositorio que contiene los dos proyectos:

```
git clone https://github.com/Polar7/prueba-tecnica-ptm.git
```

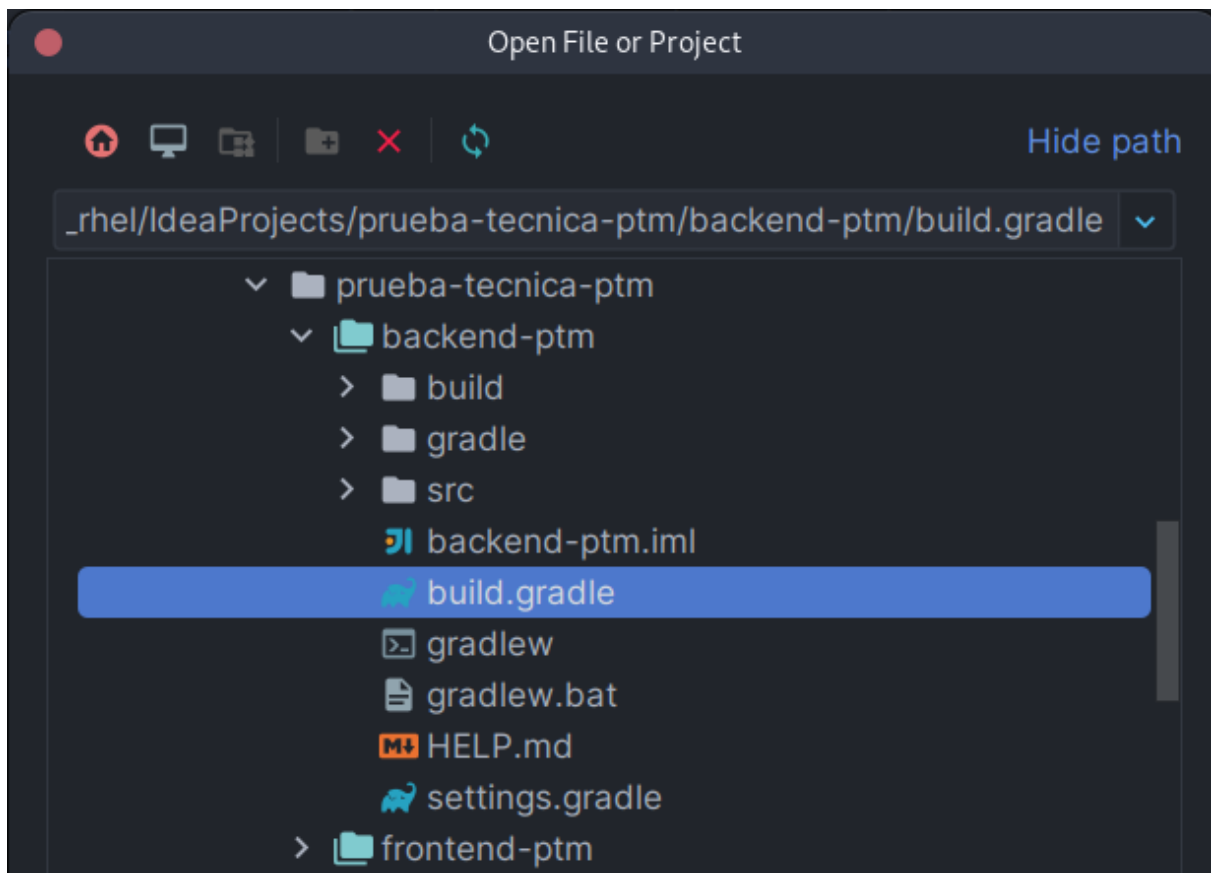
Allí se encontrarán dos carpetas, una para el backend en SpringBoot y otra para el frontend en Angular.

## Backend

2. En la ruta: **prueba-tecnica-ptm/backend-ptm/src/main/resources/static/queries.sql** se encuentran los scripts a ejecutar en MySQL para la creacion de la base de datos y la tabla necesaria.
3. Posterior a la ejecución de los scripts sql, se debe modificar las credenciales de acceso a la base de datos en el archivo application.yml de springboot, encontrado en la ruta **prueba-tecnica-ptm/backend-ptm/src/main/resources/application.yml**

```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/crud_app
    username: root
    password: Root123.
jpa:
```

4. Para la ejecución del backend, existen dos maneras, mediante IDE o por consola manualmente.
- 4.1 En IntelliJ por ejemplo, damos click en el botón de abrir proyecto (Open) y seleccionamos el archivo build.gradle en la ruta **prueba-tecnica-ptm/backend-ptm/build.gradle**



Se cargará el proyecto y nos dejará ejecutarlo con el comando de Run que tiene IntelliJ.

4.2 Para ejecutarlo por consola, debemos ubicarnos en la ruta **prueba-tecnica-ptm/backend-ptm/** y ejecutar los siguientes comandos para generar y correr el JAR :

```
./gradlew clean build
```

```
java -jar ./build/libs/backend-ptm-0.0.1-SNAPSHOT.jar
```

Ya sea desde un IDE o desde la consola, ya nuestro backend deberá iniciar usando el puerto 8090 (en caso de necesitar cambiarlo, se debe modificar el **application.yml**)

```

Terminal Local (2) x Local x +
:: Spring Boot :: (v3.2.2)

2024-02-23T01:06:37.798-05:00 INFO 5893 --- [main] com.backendptm.BackendPtmApplication : Starting BackendPtmApplication v0.0.1-SNAPSHOT using Java 17.0.9 with PID 5893 (/home/polar_rhel/IdeaProjects/prueba-tecnica-ptm/BackendPtmApplication
0.1-SNAPSHOT.jar started by polar_rhel in /home/polar_rhel/IdeaProjects/prueba-tecnica-ptm/BackendPtmApplication)
2024-02-23T01:06:37.800-05:00 INFO 5893 --- [main] com.backendptm.BackendPtmApplication : No active profile set, falling back to 1 default profile: "default"
2024-02-23T01:06:38.367-05:00 INFO 5893 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2024-02-23T01:06:38.402-05:00 INFO 5893 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 28 ms. Found 1 JPA repository interface.
2024-02-23T01:06:38.749-05:00 INFO 5893 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8090 (http)
2024-02-23T01:06:38.761-05:00 INFO 5893 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-02-23T01:06:38.762-05:00 INFO 5893 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.18]
2024-02-23T01:06:38.795-05:00 INFO 5893 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2024-02-23T01:06:38.797-05:00 INFO 5893 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 946 ms
2024-02-23T01:06:38.942-05:00 INFO 5893 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-02-23T01:06:38.942-05:00 INFO 5893 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@2fd1ad8a
2024-02-23T01:06:39.220-05:00 INFO 5893 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-02-23T01:06:39.254-05:00 INFO 5893 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2024-02-23T01:06:39.298-05:00 INFO 5893 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.4.1.Final
2024-02-23T01:06:39.328-05:00 INFO 5893 --- [main] o.h.c.internal.RegionFactoryInitiator : HHH000826: Second-level cache disabled
2024-02-23T01:06:39.531-05:00 INFO 5893 --- [main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup; ignoring JPA class transformer
2024-02-23T01:06:39.986-05:00 INFO 5893 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integ
2024-02-23T01:06:39.988-05:00 INFO 5893 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2024-02-23T01:06:40.287-05:00 WARN 5893 --- [main] jpaBaseConfigurationJpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view ren
sable this warning
2024-02-23T01:06:40.440-05:00 INFO 5893 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8090 (http) with context path ''
2024-02-23T01:06:40.671-05:00 INFO 5893 --- [main] com.backendptm.BackendPtmApplication : Started BackendPtmApplication in 3.198 seconds (process running for 3.511)

```

- Con el backend encendido, podemos acceder a la ruta <http://localhost:8090/swagger-ui/index.html> donde aparecerá una interfaz de Swagger, donde podemos probar los endpoint disponibles.

OpenAPI definition
v0
OAS 3.0
/v3/api-docs

Servers
http://localhost:8090 - Generated server uri

product-controller

GET /products Devuelve el listado de productos

PUT /products Modifica un producto existente

POST /products Agrega un nuevo producto

GET /products/{id} Obtiene un producto dado su ID

DELETE /products/{id} Elimina un producto dado su ID

GET /products/combination/{valueFilter} Devuelve un listado de combinaciones de productos cuya suma de precios sea menor o igual al valor ingresado

## Frontend

- Para encender el frontend, abrimos una consola en la ubicación **prueba-tecnica-ptm/frontend-ptm** y allí ejecutamos `npm install`, el cual instalará las dependencias requeridas.
- Una vez instaladas, ejecutamos `ng serve` y la aplicación iniciará usando el puerto 4200 y lista para acceder.

```
~/IdeaProjects/prueba-tecnica-ptm/frontend-ptm on git develop ng serve
at 01:16:22

: Building...

Initial chunk files | Names      | Raw size
styles.css          | styles     | 273.78 kB |
scripts.js          | scripts    | 123.22 kB |
polyfills.js        | polyfills  | 83.60 kB  |
chunk-4ZMNJXA6.js   | -          | 30.34 kB  |
main.js             | main       | 2.71 kB   |

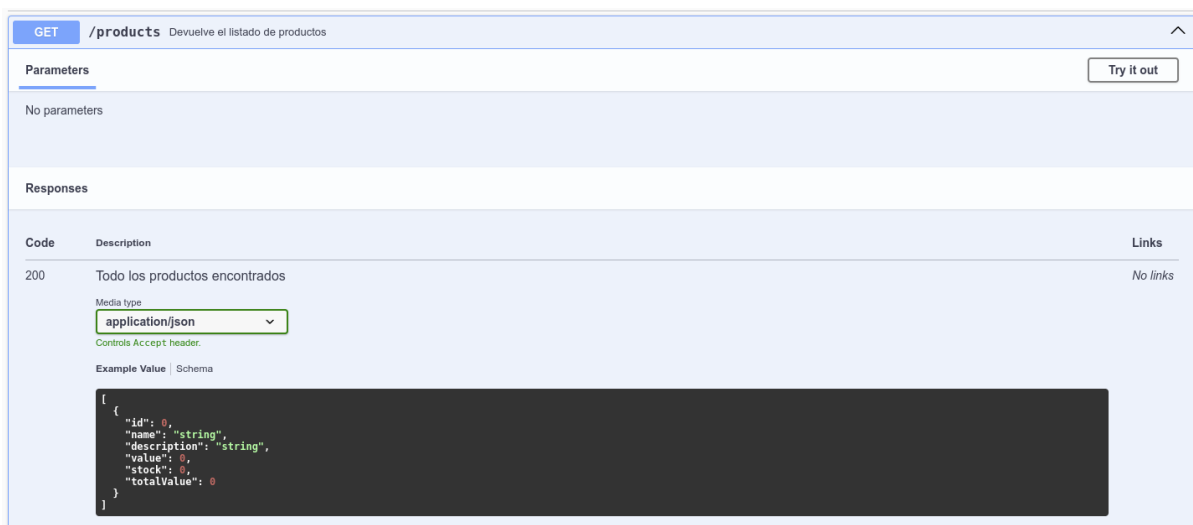
| Initial total | 513.65 kB

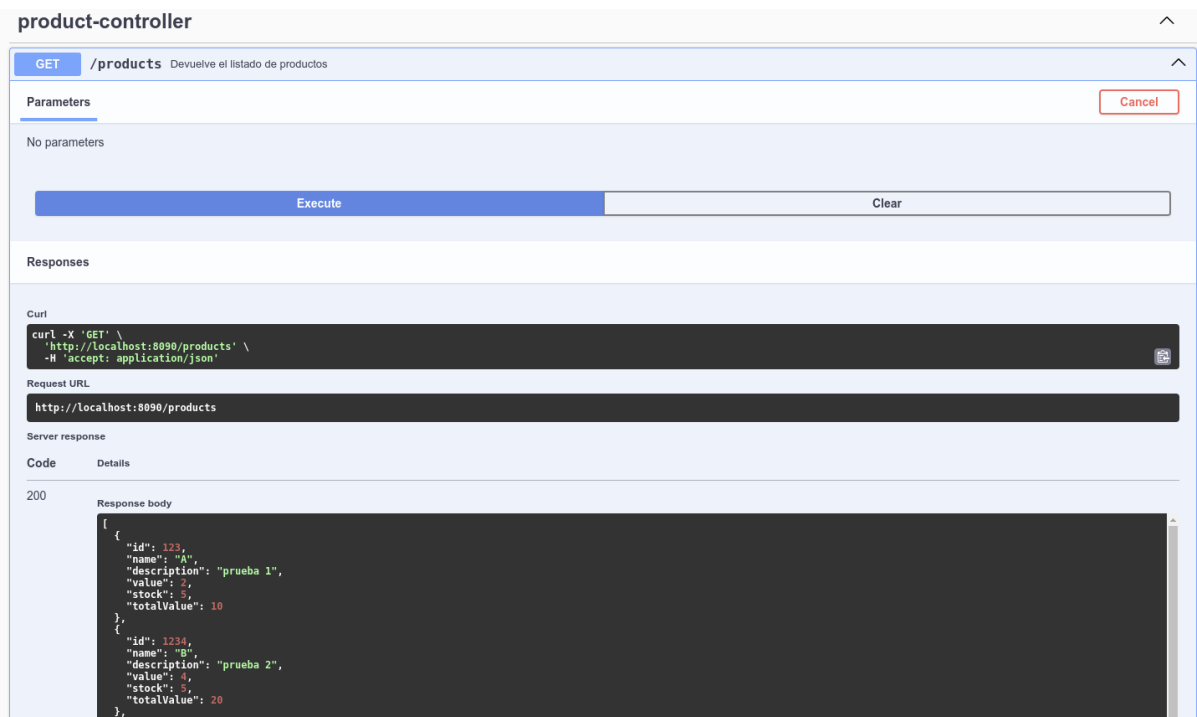
Lazy chunk files | Names      | Raw size
chunk-BWJPAV3V.js | feature-module | 126 bytes |

Application bundle generation complete. [1.114 seconds]
Watch mode enabled. Watching for file changes...
→ Local: http://localhost:4200/
```

## Uso de Swagger

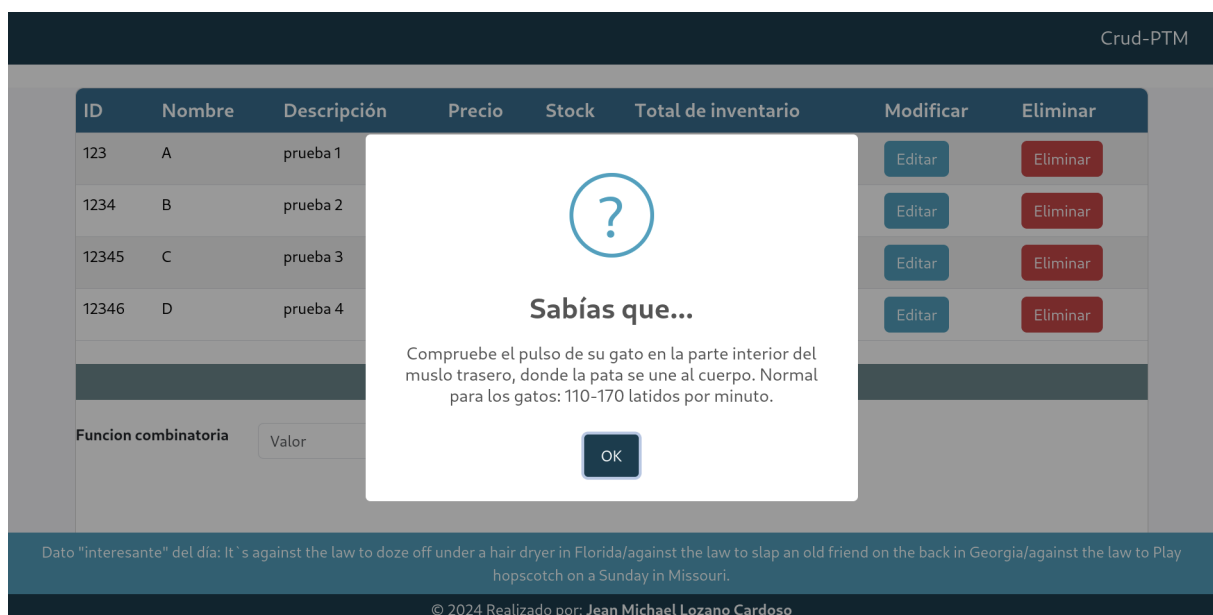
En la interfaz de Swagger tendremos los botones “Try it out” y “Execute” para probar los diferentes endpoints, esto para evitar el uso de aplicaciones externas como Postman.





## Funcionalidades en el frontend

Cuando accedemos al frontend, nos encontramos con la siguiente pantalla:

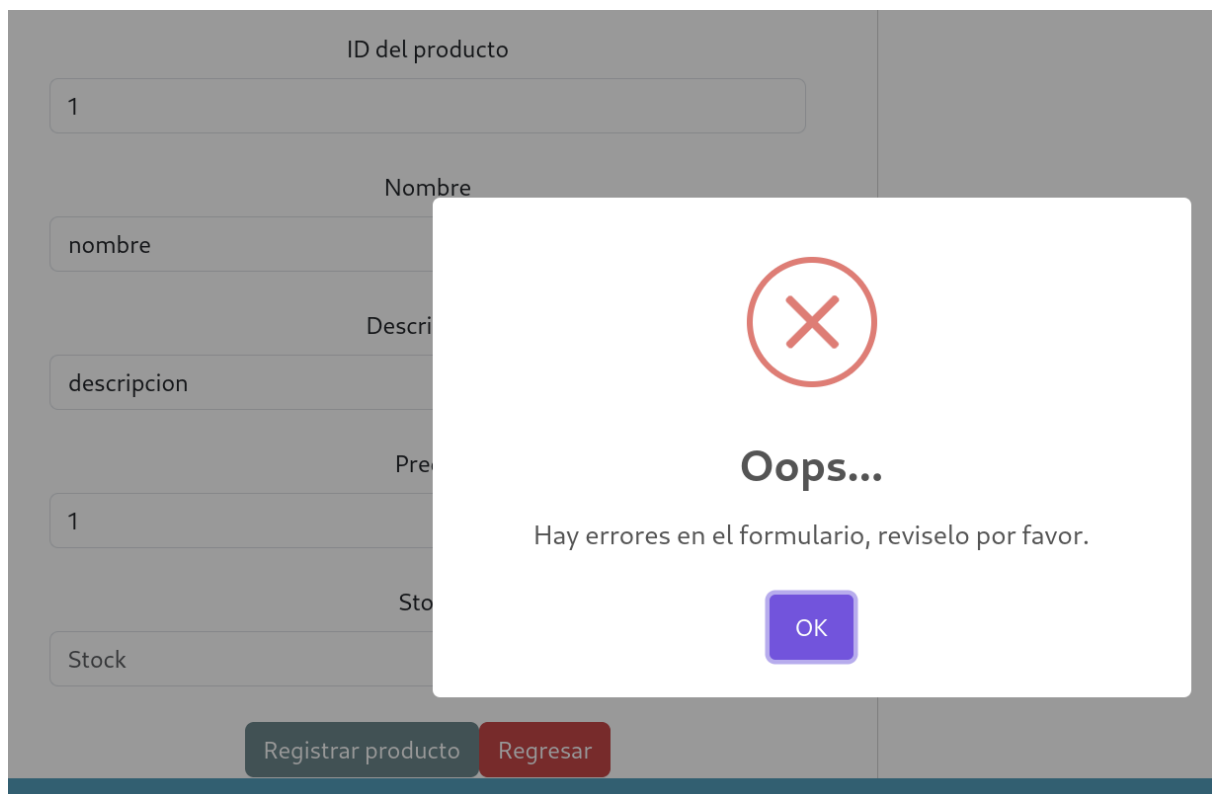


Allí se observa un enfoque en la modal emergente la cual contiene datos de datos, como uno de los requerimientos de la prueba.

Veremos en el pie de página el dato inútil del día, otro requerimiento.

Una vez cerradas las modales podemos ver los datos de los productos que existen en la base de datos y diferentes botones para interactuar con dichos productos.

Cuando se intenta agregar o modificar un producto, existen validaciones de obligatoriedad en todos los campos, si no se llenan, aparecerá una ventana de error.



Y finalmente la sección de la función combinatoria nos mostrará las combinaciones dado el filtro que se solicitó en la prueba.

