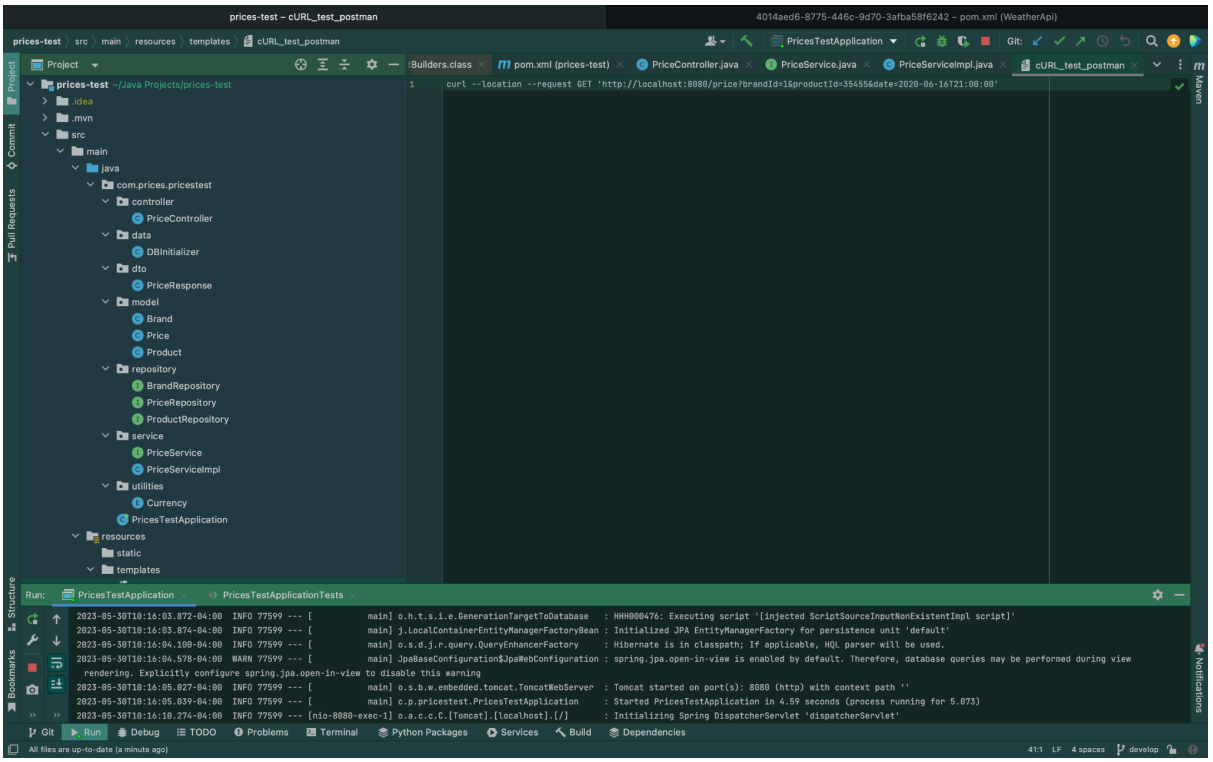
Lista de Precios Coding Test

Saludos, este documento tiene como finalidad explicar como se desarrollo la solución para el coding Test de Lista de Precios. Para esto fue creado un projecto utilizando Spring boot, Maven, H2 (base de datos en memoria).



Estructura de Clases

Entity o Model

En este paquete se guardan las entidades involucradas en la solución.

- Brand: Entidad de la marca, obdece a la relación entre la tabla de brands y prices.
- Product: Entidad del producto, obedece a la relación entre la tabla products y prices.
- Price: Entidad que obedece a la tabla prices.

Repository

En este paquete se guardan los repositorios de las entidades creadas anteriormente, donde usando Spring Repository podemos utilizar las funciones básicas de persistencia para los repositorios de Brand, Product y Price. Adicionalmente para este último (Price) fue

necesario agregar una nueva funcionalidad con el query específico para filtrar la data de acuerdo como se requiere en la necesidad de negocio.

- BrandRepository
- ProductRepository
- PriceRepository

Dto

Para esta solución fue creado un DTO con los atributos que se requieren retornar al cliente que hace la consulta. El DTO PriceResponse contiene los siguientes atributos:

- productId
- brandId
- priceList
- startDate
- endDate
- finalPrice

Este objeto se crea para estar alineado con lo que el cliente espera recibir sin tener que exponer nuestras entidades.

Controller

Se creó un controlador donde se específica un único endpoint para la solución, el cual tiene como nombre getPrice y recibe como parámetros (query param) el id de la marca (brandId) el id del producto (productId) y la fecha (date). Al final se retorna el response determinado y en caso de ser satisfactorio (ok) se retorna un objeto de tipo PriceReponse, el DTO mencionado en el paso anterior.

Service

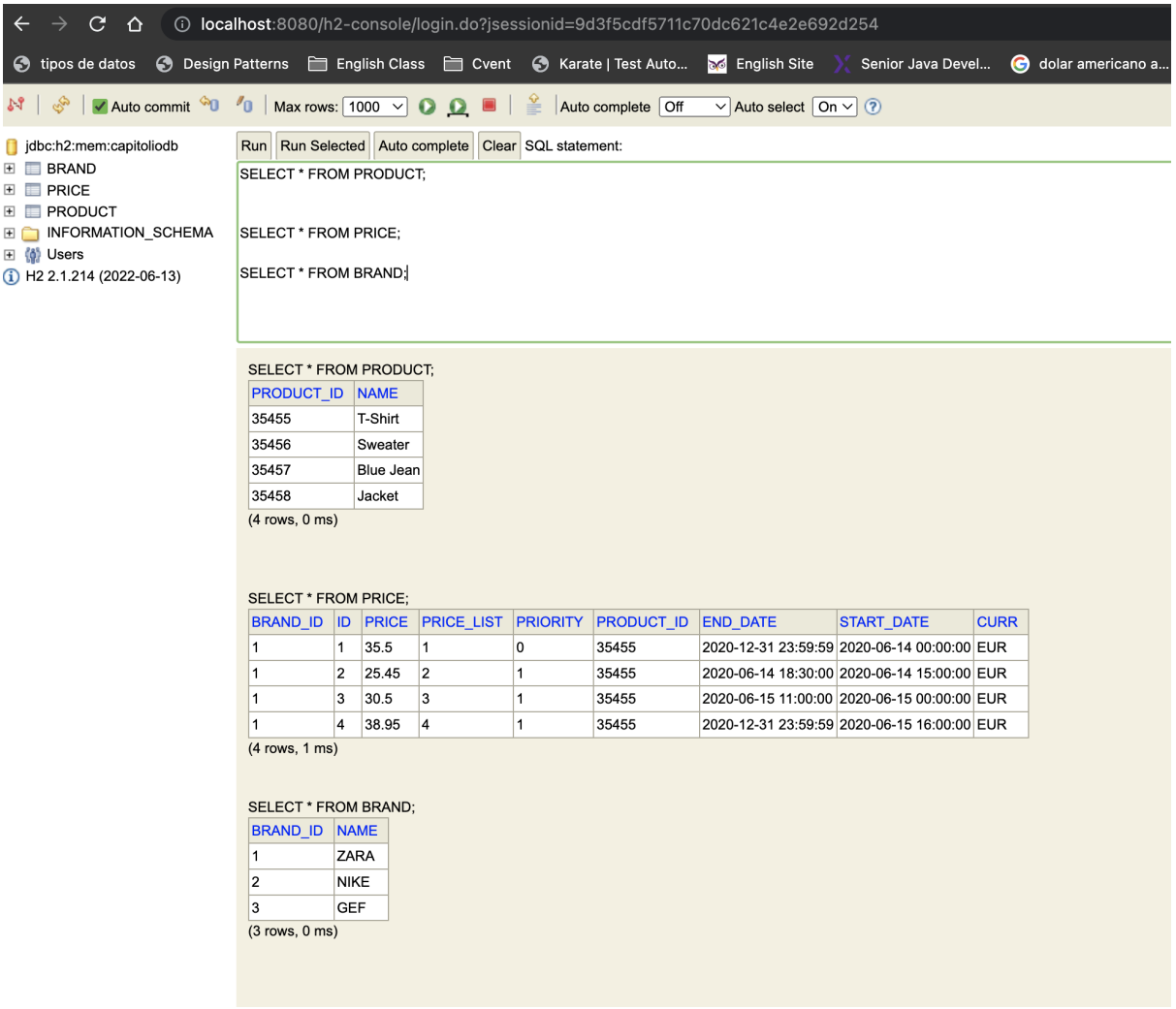
Para esta solución y siguiendo el conjunto de buenas practicas se implemento un servicio donde se maneja lógica de llamado a la capa de persistencia, se valida el resultado y se retorna una respuesta mapeada al objeto de negocio dependiendo del la información obtenida de la base de datos.

Utilities

Fue creada la enumeracíon Currency solo con la finalidad de hacer el escenario más real, no agrega valor de negocio a la solución.

Data

En esta categoria fue creada una clase con un initilializer para la base datos en memoria, con finalidad que una vez desplegada la solución se puedan ejecutar desde postman los request propuestos en el documento del Coding Test.

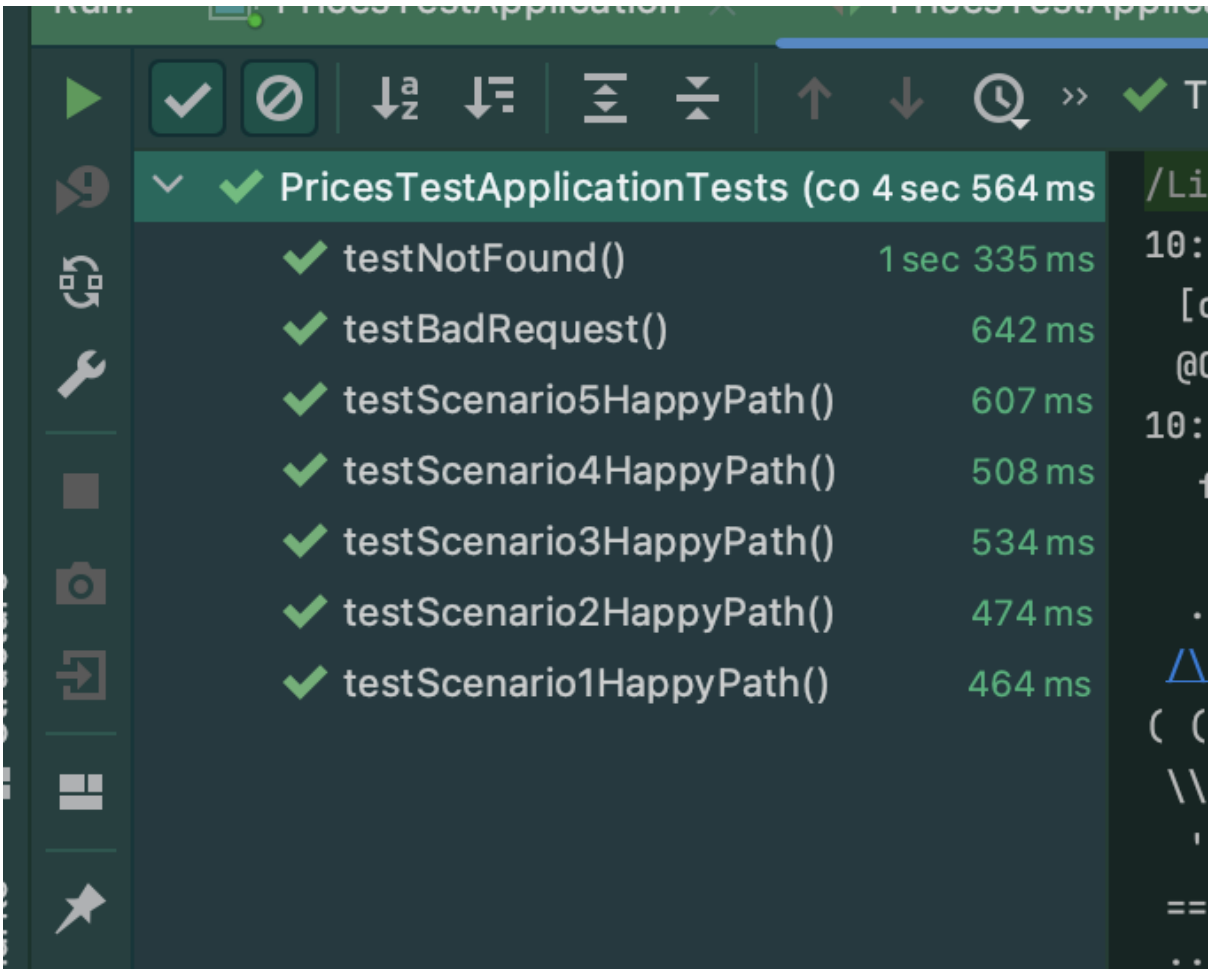


Resources

En la carpeta templates fue creado un archivo plan con un cURL de ejemplo para realizar las pruebas funcionales desde postman.

Pruebas de Integracion

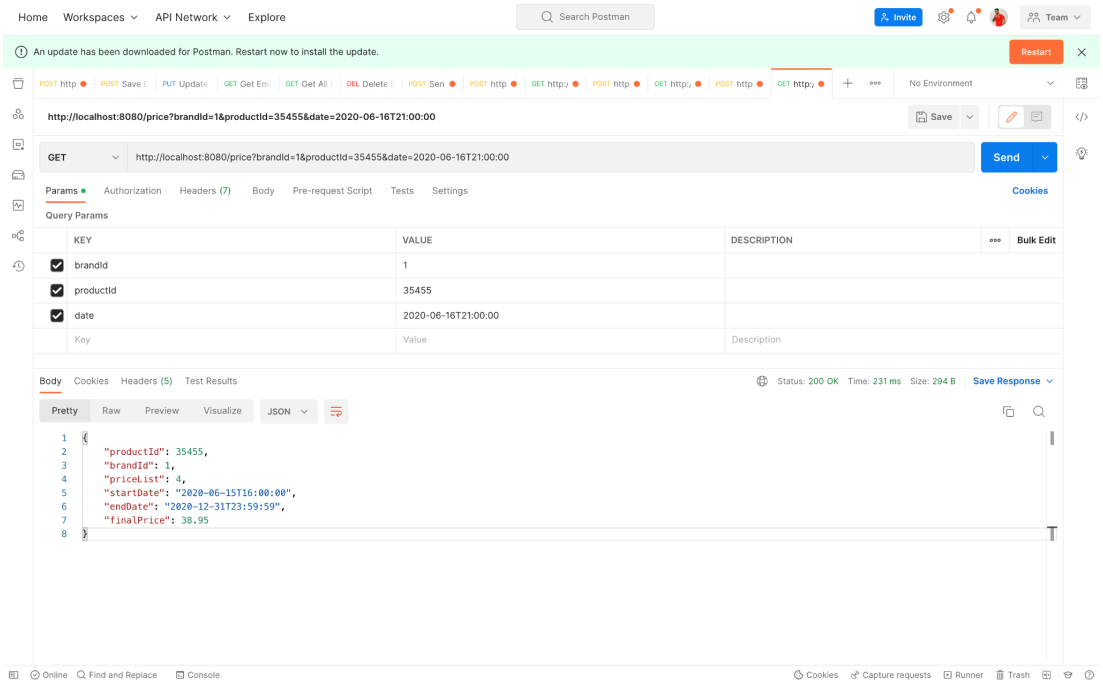
Fueron creadas las pruebas de integración en la clase PricesTestAplicationTests, las cuales cubren los escenarios planteados en el problema y se centran solo en los criterios de aceptación, no en ampliar cobertura.



Pruebas desde Postman

Para las pruebas en Postman podemos usar el siguiente request y validar los diferentes resultados variando los valores de los parametros

```
curl --location --request GET
'http://localhost:8080/price?brandId=1&productId=35455&date=2020-06-16T21:00:00'
```



Gracias por la atención prestada y no dude en consultarme cuauquier inquietud.