

Requirements

myRetail RESTful service

myRetail is a rapidly growing company with HQ in Richmond, VA and over 200 stores across the east coast. myRetail wants to make its internal data available to any number of client devices, from myRetail.com to native mobile apps.

The goal for this exercise is to create an end-to-end Proof-of-Concept for a products API, which will aggregate product data from multiple sources and return it as JSON to the caller.

Your goal is to create a RESTful service that can retrieve product and price details by ID. The URL structure is up to you to define, but try to follow some sort of logical convention.

Build an application that performs the following actions:

- Responds to an HTTP GET request at `/products/{id}` and delivers product data as JSON (where `{id}` will be a number).
- Example product IDs: 15117729, 16483589, 16696652, 16752456, 15643793)
- Example response: `{"id":13860428,"name":"The Big Lebowski (Blu-ray (Widescreen))","current_price":{"value": 13.49,"currency_code":"USD"}}`
- Performs an HTTP GET to retrieve the product name from an external API. (For this exercise the data will come from `redsky.target.com`, but let's just pretend this is an internal resource hosted by myRetail)
- Example:
http://redsky.target.com/v2/pdp/tcin/13860428?excludes=taxonomy,price,promotion,bulk_shipping,rating_and_review_reviews,rating_and_review_statistics,question_answer_statistics
- Reads pricing information from a NoSQL data store and combines it with the product id and name from the HTTP request into a single response.
- BONUS: Accepts an HTTP PUT request at the same path (`/products/{id}`), containing a JSON request body similar to the GET response, and updates the product's price in the data store.

Solution:

Tools Used

Spring Boot: Server side framework

Cassandra: NoSQL database for persistent storage.

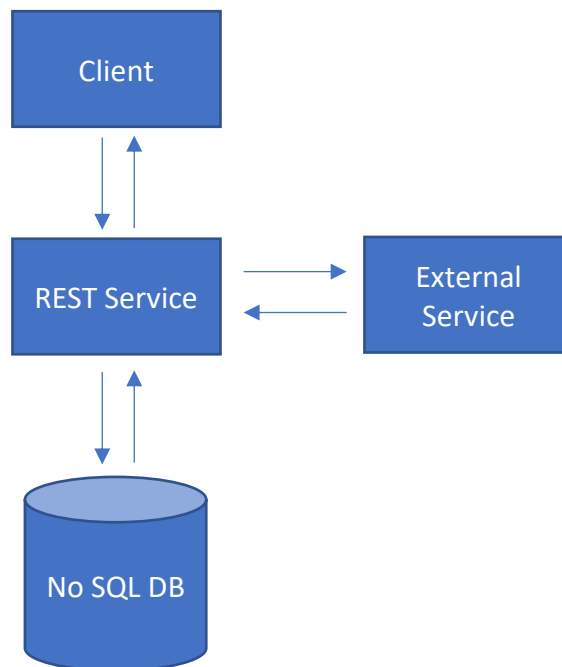
Feign: For connecting and consuming external service.

Swagger: Api – API Documentation

Database Scripts

The Cassandra.cql file in the project has the Cassandra scripts with the table schema

High-Level Design



Running the Project

- 1) Execute the CQL scripts in the Cassandra.cql file at (src/main/resource)
- 2) Update the below settings in the application.properties file.
 - a. spring.data.cassandra.contact-points=[localhost](#)
 - b. spring.data.cassandra.port=[9042](#)

Sample Data

<http://localhost:8080/v1/products/13860428>
<http://localhost:8080/v1/products/77400978>
<http://localhost:8080/v1/products/77285200>

Support Contact

Email: gersonbothello@gmail.com
Phone: 603 264 5005

Retrieve Data

URL for GET: <http://localhost:8080/v1/products/13860428>

Untitled Request Comments 0

GET <http://localhost:8080/v1/products/13860428> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

body Cookies Headers (5) Test Results Status: 200 OK Time: 2.07 s Size: 309 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "productId": 13860428,
3   "productName": "The Big Lebowski (Blu-ray)",
4   "currentPrice": {
5     "productId": 13860428,
6     "currentPrice": 10.20,
7     "currencyCode": "USD"
8   }
9 }
```

Modify Data

URL for PUT: <http://localhost:8080/v1/products/13860428>

Jntitled Request

Comments 0

PUT

http://localhost:8080/v1/products/13860428

Send

Save

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Cod

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1 {

2 "productId": 13860428,

3 "productName": null,

4 "currentPrice": {

5 "productId": "13860428",

6 "currentPrice": 13.00,

7 "currencyCode": "USD"

8 }

9 }

body

Cookies

Headers (5)

Test Results

Status: 201 Created

Time: 409 ms

Size: 182 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1

Swagger - <http://localhost:8080/swagger-ui.html#/>

The screenshot shows the Swagger UI interface in a web browser. The browser's address bar displays `localhost:8080/swagger-ui.html/`. The Swagger UI header is green with the Swagger logo and a dropdown menu labeled "Select a spec" with "default" selected. The main content area features the title "myRetail Service" with a "Version 1.0" badge. Below the title, it shows the base URL: "[Base URL: localhost:8080/]" and a link to <http://localhost:8080/v2/api-docs>. The "Product" section is expanded, showing the "Product Controller". It lists two endpoints: a GET endpoint `/v1/products/{id}` with the description "Find Product by Id." and a PUT endpoint `/v1/products/{id}` with the description "Modify the product pricing". A "Models" section is visible at the bottom with a right-pointing arrow.

localhost:8080/swagger-ui.html/

Apps IKWeb Educative MY INSITE - Sign In Imported From Fir... Practice Mock Int... The 14 Best Drops... Other Bookmarks

swagger Select a spec default

myRetail Service Version 1.0

[Base URL: localhost:8080/]
<http://localhost:8080/v2/api-docs>

Product Product Controller

GET `/v1/products/{id}` Find Product by Id.

PUT `/v1/products/{id}` Modify the product pricing

Models