**Case Study:**

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.

The goal is to create a RESTful service that can retrieve product and price details by ID.

**Technology Stack:**

Spring boot + MongoDB

**Tools used for Development & Testing:**

IntelliJ IDEA and POSTMAN for testing RESTful API.

**Evidence of Testing:**

Postman screenshots

**Security:**

Basic authentication.

**Solution:**

This API (MyRetail) provides the ability to:

1. Save Product and Price information. Before saving, the product title is retrieved via a RESTful Web service call while the product id, price and currency entered by the user.

Method: POST

Mapping: /products/saveProduct

Sample Payload:

{

    "productId" : "300",

    "price" : "12.50",

    "currency" : "USD"

 }

Credentials: admin/password

A screenshot of a computer

Description automatically generated

1. Retrieve Product and Price information. These details will be retrieved from MongoDB.

Method: GET

Mapping: /products/product/300

Credentials: admin/password & user/password

Sample Response:

{

    "id": "621628a8f211713751df3af9",

    "productId": "300",

    "title": "Product:300",

    "price": 1000.1,

    "currency": "USD"

}

Graphical user interface, text, application, email

Description automatically generated

1. Update Price Information by Id.

Method: PUT

Mapping: /products/product/621628a8f211713751df3af9

Credentials: admin/password

Payload:

{

    "price" : "1000.10"

}

Sample Response:

{

    "id": "621628a8f211713751df3af9",

    "productId": "300",

    "title": "Product:300",

    "price": 1000.1,

    "currency": "USD"

}

A screenshot of a computer

Description automatically generated