***RETAIL PRODUCT MANAGEMENT SYSTEM***

***Project Members:***

* SHIVENDU KUMAR (897018)
* SIDDHARTHA PAUL (897009)
* BINOY KOLAY (897007)

***Project Overview:***

A leading Retail Organization wants to strengthen its Middleware by exposing the core logic related to Product Management as Microservices. This middle ware Microservices will be hosted on Cloud so that all the up/downstream applications can get an access to this for performing business transactions.

***Project Details:***

***Microservices:***

* ***Authorization -***
  + The Authorization microservice is used to create JWT tokens. It used for authorization of other microservices.
* ***Product Microservice -***
* An authorized customer can search for product using product id or name. Post Authorization, the product id, name, price, product image reference and description details are returned as output. If the product is inactive or out of stock then it is send to the wishlist. The customer can add a Rating which gets associated to the product.
* ***Vendor Microservice-***
* During the order fulfillment cycle, a vendor will be mapped to the order. The vendor will ship the product from their warehouse and deliver it to the customer location. ProceedToBuy Microservice interacts with Vendor Microservice to assign a vendor for the customer’s order.
* ***ProceedToBuy-***
* An authorized customer can add the product to the cart. An authorized customer can add product to cart and view the product details (Product Price, Delivery Date, Vendor details) in the checkout/cart page. If Product is not available, the customer can add the product to wishlist.
* ***E-Commerce Portal-***
  + E-Commerce Portal microservice consists of all jsp pages which are connected to various other microservices.

***Port Number:***

* + Authorization – 8081
  + Product – 8082
  + Vendor – 8083
  + Proceed To Buy – 8084
* E-Commerce Portal – 8080

***Login Credentials:***

* Username - shiv
* Password – shiv

**-OR-**

You can just Sign-up and Start using

***Amazon Web Services (AWS) – Elastic Beanstalk:***

* All the Microservices are deployed on AWS Elastic Beanstalk.
* AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.
* For more information about AWS Beanstalk, refer the following link:- <https://aws.amazon.com/elasticbeanstalk/>

***AWS Elastic Beanstalk deployed microservices links:***

* Authorization Microservice

http://retailauthms-env.eba-gvhpebqq.ap-south-1.elasticbeanstalk.com

* Product Microservice

http://retailproductms.eba-gvhpebqq.ap-south-1.elasticbeanstalk.com

* Vendor Microservice

http://retail-product-vendor-ms.eba-gvhpebqq.ap-south-1.elasticbeanstalk.com

* ProceedToBuy Microservice

<http://retail-product-proceedtobuy-ms.eba-gvhpebqq.ap-south-1.elasticbeanstalk.com>

* E-Commerce Portal

<http://localhost:8080/>

NOTE:(These services must be consumed from an MVC app running in a local environment)

***Database:***

* All the microservices are independently deployed.
* An In-memory database (H2 database) has been used in the application.

H2 Console links:

1. <http://localhost:8081/api/h2-console>

* Tables
* USER\_ID
* FNAME
* LNAME
* UNAME
* UPASSWORD
* EMAIL
* ADDRESS
* ZCODE

1. <http://localhost:8082/api/h2-console>

* Tables
* ID
* COUNT
* DESCRIPTION
* IMAGE\_NAME
* NAME
* PRICE
* RATING

1. <http://localhost:8083/api/h2-console>

* Tables
* VENDOR\_ID
* DELIVERY\_CHARGE
* RATING
* VENDOR NAME
* Tables
* VENDOR\_STOCK\_ID
* EXPECTED\_STOCK\_REPLENISHMENT\_DATE
* PRODUCT\_ID
* STOCK\_IN\_HAND
* VENDOR\_ID

1. <http://localhost:8084/api/h2-console>

* Tables
* WISHLIST\_ID
* ADDING\_DATE\_TO\_WISHLIST
* PRODUCT\_ID
* CUSTOMER\_ID
* QUANTITY

***EclEmma Code Coverage:***

[EclEmma](http://www.eclemma.org/) is a great Java code coverage tool that has an Eclipse plugin.

It’s very simple and intuitive and has all you would expect from a code coverage tool. With it, you can:

* See code coverage for a java application that you’ve run (and potentially merge multiple run instances)
* See code coverage for JUnit tests and maven tests
* See which classes have which amount of coverage.