|  |
| --- |
|  |

|  |
| --- |
| **eMart (Mid Tier) Phase4 v4.0** |
| Case Study |
|  |
| This document covers Software Requirements of eMart, along with list of Technologies to be used to develop this Software System, and also includes some details on the Architecture |
|  |
| **IIHT** |
| **2/7/2019** |
|  |

Table of Contents

[1. Business Requirement(eMart) 2](#_Toc28991912)

[1.1. Roles 2](#_Toc28991913)

[1.1.1. Buyer Use Cases. 2](#_Toc28991914)

[1.1.2. Seller Use Cases 2](#_Toc28991915)

[1.1.3. Admin Use Cases(optional) 3](#_Toc28991916)

[1.1.4. Data Fields 3](#_Toc28991917)

[2. Design Inputs 6](#_Toc28991918)

[3. Microservices Integration and Security 6](#_Toc28991919)

[3. Spring Microservices Tools to be used 6](#_Toc28991920)

[4. JWT Authentication 7](#_Toc28991921)

[5. Architecture/Design 7](#_Toc28991922)

[6. Technical Spec – Solution Development Environment 8](#_Toc28991923)

[6.1. Front End Layer 8](#_Toc28991924)

[6.2. Middle Tier Layer 8](#_Toc28991925)

[6.3. Database & Integration Layer 8](#_Toc28991926)

[6.4. Ancillary Layer 8](#_Toc28991927)

[Controllers can be tested using Postman Tool 8](#_Toc28991928)

[6.5. Security 8](#_Toc28991929)

[6.6. Deployment & Infrastructure 8](#_Toc28991930)

[6.7. Editors 9](#_Toc28991931)

[7. Assessment Deliverables 9](#_Toc28991932)

[8. Important Instructions 9](#_Toc28991933)

# Business Requirement(eMart)

eMart is a online eCommerce portal used to search and buy an item online. Buyer can add items to the cart, checkout and perform other operations. Admin can block/unblock Seller, Buyer, selling items. Below are eMart Features in detail.

## Roles

Below are the different roles, which need to be supported by above Software System.

#1. Buyer

#2. Seller

#3. Admin(optional)

Below are the Use Cases which need to be supported by each of above Roles

### Buyer Use Cases.

Login/Logout

Signup

Should be able to search an Item.

Once list of items are displayed, it should be possible to filter items based on Price, Manufacturer, etc...

When a specific item is selected by Buyer, complete details of item need to be displayed along with picture(one or more) and (list of)specifications which depends on Category and Sub Category of the item. Add to Cart button need to be provided.

It should be possible to open the Cart and checkout. Tax amount need to be displayed.

Items in the Cart can be deleted

It should be possible to apply discount, before checking out.

It should be possible to view History of purchases, along with Item, numbers

Integration with any Payment Gateway(optional)

### Seller Use Cases

Login/Logout

Signup with details such as Company/individual name, email id, Postal Address, GSTIN, Bank details

Add an item to be sold along with the number of items(in Stock)

Should be able to view Inventory(items sold and remaining)

Should be able to generate report like number of items sold and which items, over certain period.

Tax Calculation need to be performed.(optional)

### Admin Use Cases(optional)

Login/Logout

Block/Unblock a specific Seller

Block/Unblock a specific item sold by a Seller

Block/Unblock a specific Buyer

Add/remove Categories, Sub Category along with GST

Add/update discounts - discount code, start date, end date

View daily Turnover Category wise

**NOTE: Shipping/Delivery Tracking related Use Cases are not in the scope, it is assumed that Seller need to manage Shipping/Delivery Tracking offline.**

### Data Fields

Below are data fields, for your reference, based on your Analysis you made add/remove data fields

**Buyer:** Buyer’s login and profile details

id

username

password

emailid

mobile number

created datetime

**Seller:** Seller’s login and Selling company details

id

username

password

companyname

GSTIN

brief about company

postal\_address

website

emailid

contact number

**Category:** List of Categories, for example Electronic, Fashion, etc…

category\_id

category\_name

brief\_details

**Subcategory:** Sub Category of each Category. For example Electronic Category can have Mobile, TV, Laptop, etc… as Sub categories

subcategory\_id

subcategory\_name

category\_id

brief\_details

GST %

For example Mobiles, TV, etc... can be Sub Catgeories in Electronic Category

**Items:** Selling Item details

id

category\_id

subcategory\_id

price

item\_name

description

stock\_number

remarks

**Purchase History:**

Id

Buyer\_id

Seller\_id

Transaction\_id

Item\_id

Number\_of\_items

Date\_time

remarks

**Transactions:** Transactions performed during Checkout, etc…

id

user\_id

seller\_id

transaction\_type(Eg. debit or credit)

date\_time

remarks

**Discounts:** Discount details

Id

Discount\_code

percentage

start\_date

end\_date

description

# Design Inputs

Next sections in this document provides inputs on designing the solution for above requirements.

Design inputs provided in this document are just for your reference purpose, Associates can make changes or additions to the Design, based on their analysis.

# Microservices Integration and Security

Assuming that you are done with developing individual Microservices in previous Phase, current Phase includes creating and integrating Zuul gateway, Eureka Server and Eureka client in each Microservice. This is shown in architecture Diagram, in next section.

Zuul Gateway(create a Zuul based Project using Spring Initilaizer or STS IDE), add required annotation. Authentication and JWT Token validation can be performed in Zuul’s Pre Filter.

Add below details to yml or property file

1. add route configurations
2. port number & url of eureka Server

Eureka Server(create a Eureka Discovery Server using Spring Initializer or STS IDE), add required annotation & port number in yaml configuration file

Add Eureka Discovery Client to all the Microservice

Now open Eureka Server Dashboard by opening and crosscheck if all Microservices are registered in the dashboard

Now start sending the requests to Zuul Gateway which further routes to a specific Microservice based on the url pattern

Develop code for Unit Testing

PostMan, to test REST end points

# Spring Microservices Tools to be used

As already specified under Full Stack Technologies Microservice Architecture need to be followed. Ensure that the Application is divided into multiple Microservices, along with database/tables each Microservice Manages. Below Spring Microservices Tools need to be used

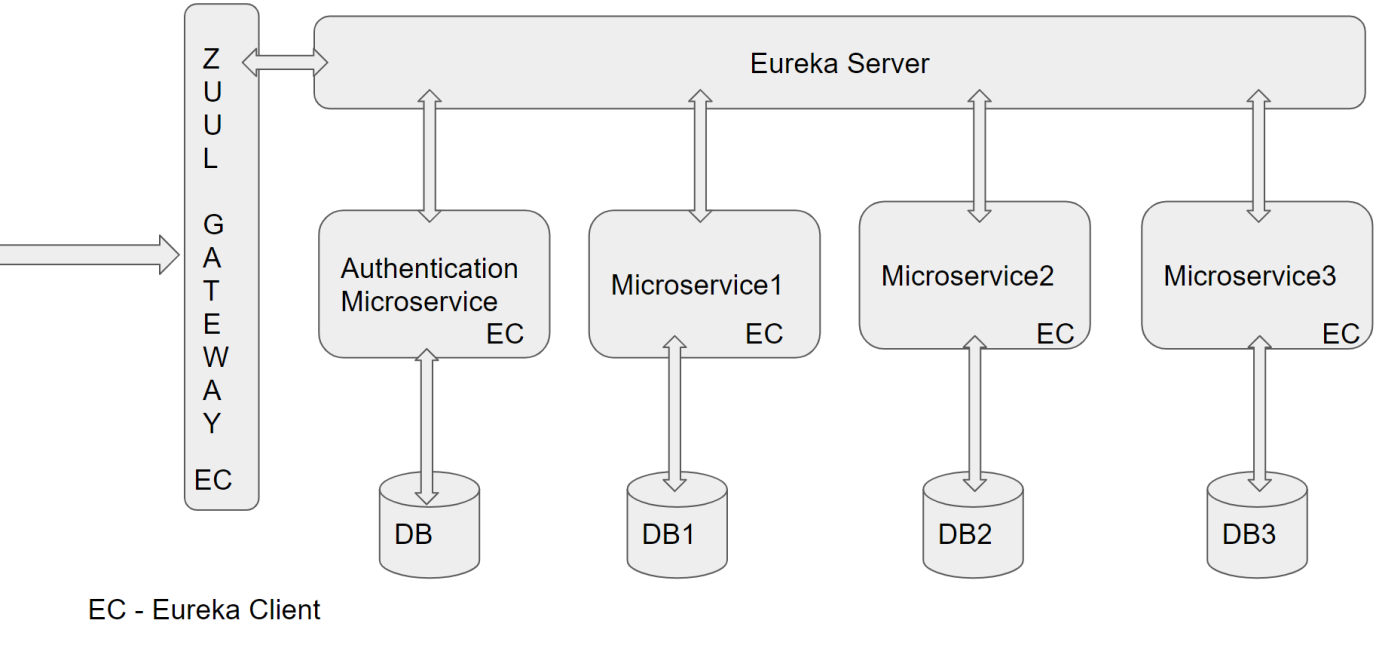
* Zuul API Gateway
* Eureka Service Registry & Discovery
* Ribbon Client side Load Balancer(optional)
* Feign Client
* Hystrix Circuit Breaker & Fault Tolerant Tool(optional)

# JWT Authentication

Create additional Microservice which takes care of authentication and role activities, and JWT Token validation. Spring Security need to be used for Authentication. On successful authentication or token validation the actual request need to be forwarded to the corresponding Microservice. Invoke authentication REST endpoints from Zuul Gateway. Use PreFilter to perform JWT Token validation by invoking REST endpoint of this Microservice.

Instead of JWT, any other security protocol such as OAuth2 can be used. Authentication data can be stored in MySQL DB or LDAP or any other data source.

# Architecture/Design



# Technical Spec – Solution Development Environment

## Front End Layer

|  |  |
| --- | --- |
| **Framework(s)/SDK/Libraries** | **Version** |
| Angular with TypeScript | 4/6 |
| Bootstrap | 3.0 or above |
| CSS | 3 |
| HTML | 5 |
| JavaScript | 1.8 or above |
| JQuery | 1.3 |

## Middle Tier Layer

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Java Stack | Spring Boot | 1.5 or above |
| Spring MVC | 4.0 or above |
| JDK | 1.7 or above |
| Maven | 3.x or above |

## Database & Integration Layer

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Java Stack | Hibernate | 4.0 or above |
| JAX-RS Jersey/ Spring Restful |  |
| MySQL | 5.7.19 |
| MongoDB | MongoDB | 3.4 |
| NoSQL |  |

## Ancillary Layer

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Source Code Management Tool | GIT | 2.14.2 |
| Build Tool/JAVA Stack | Maven | 3.x |
| Testing Tool/JAVA Stack | JUnit/Mockito | 4.x |
| Testing Tool/JAVA Stack | Spring Test | 4.x |

## Controllers can be tested using Postman Tool

## Security

|  |  |
| --- | --- |
| **Name** | **Version** |
| Spring Boot Security |  |
| JWT |  |

## Deployment & Infrastructure

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Docker | - |  |
| Apache Tomcat | - |  |
| Jenkins(CI/CD) | - |  |
| Node | - |  |

## Editors

|  |  |
| --- | --- |
| **Name** | **Version** |
| STS(Spring Tool Suite) |  |
| Visual Studio Code |  |

# 

# Assessment Deliverables

1. Check into Git, Zuul Gateway Project, Eureka Server Project, all Microservices with Eureka Client added
2. Screen shots showing sample invocation of Zuul Gateway end points from Post Man Tool.
3. Eureka Dashboard Screenshots with registered instances of Microservices
4. Few Steps on how to run the solution.

# Important Instructions

1. Consider using below Java features
2. Lambda Expressions
3. Collection Streams
4. Generics
5. Sample Design provided is just for reference, Associates can make changes over it or follow their own Design.
6. Based on your current work, alternate Technologies can be used, for example ReactJS instead of Angular, etc…, however prior approval from the Mentor is required.
7. Please make sure that your code does not have any compilation errors while submitting your case study solution.
8. The final solution should be a zipped code having solution. Solution code will be used to perform Static code evaluation.
9. Implement the code using best design standards/family Design Patterns.
10. Use Internationalization for all the labels and messages in Rest API Development.
11. Do not use System out statements or console.log for logging in Rest API and FrontEnd respectively. Use appropriate logging methods for logging statements/variable/return values.
12. If you are using Spring Restful or Jersey JAX-RS to develop Rest API, then use Maven to build the project and create WAR file.
13. Write web service which takes input and return required details from database.
14. Use JSON format to transfer the results.
15. For any further queries you can contact fullstack@iiht.com