REVIEW MANAGEMENT SYSTEM

DESCRIPTION:

A Review management system allows user to review the products and its specifications individually. Admin can delete the review if it is not suitable for that product.

ASSUMPTIONS:

* Product details and its features are already available.
* Users can search for any product review present in the database.

TECHNOLOGIES USED:

* Frontend: Angular 8
* Middleware: Spring Boot Application
* Backend: Database - MySQL

IN THE SCOPE:

* User can search entire product reviews with product Id.
* User can search for the reviews of any specific feature with their specific Id or also with the product Id.
* User can modify or delete his own review but not others.

OUT OF THE SCOPE:

* Review cannot be retrieved through particular userId, because user can post reviews on different products.

Database

MICROSERVICES:

Why we have chosen microservices is because it allows us to optimize resources,Multiple teams work on independent services, enabling us to deploy more quickly.

Following can be possible microservices:

1. USER SERVICE:

User Service allows to take user credentials and validates the user. With user-service the user can register and login.

* Endpoint-Request Mapping (method=POST, <http://localhost:portnumber/register/user>)
* Endpoint-Request Mapping (method=GET, <http://localhost:portnumber/login/userName>)

1. PRODUCT SERVICE:

Product service is to help the user to Post, Get or Delete reviews for products and its features. It will get the request from review service.

* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ products)

Response Body-JSON Format-

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

}

* Endpoint-Request Mapping (method=GET, https://localhost:portnumber/reviews/ ram)

Response Body-JSON Format-

“ram”: {

“rId”: ,

“ramSize”: ,

“reviewText”: ,

“stars”: ,

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

},

}

* Endpoint-Request Mapping (method=GET, https://localhost:portnumber/reviews/ camera)

Response Body-JSON Format-

“camera”: {

“cId”: ,

“resolution”: ,

“reviewText”: ,

“stars”: ,

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

},}

* Endpoint-Request Mapping (method=GET, https://localhost:portnumber/reviews/ display)

Response Body-JSON Format-

“display”: {

“dId”: ,

“inches”: ,

“reviewText”: ,

“stars”: ,

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

},

}

* Endpoint-Request Mapping (method=GET, https://localhost:portnumber/reviews/ processor)

Response Body-JSON Format-

“processor”: {

“pr”: ,

“prName”: ,

“reviewText”: ,

“stars”: ,

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

},}

* Endpoint-Request Mapping (method=GET, https://localhost:portnumber/reviews/ battery)

Response Body-JSON Format-

“battery”: {

“bId”: ,

“capaciy”: ,

“reviewText”: ,

“stars”: ,

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”:

}

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ product)

Request Body-JSON Format-

“product”: {

“pId”: ,

“pName”: ,

“reviewText”: ,

“stars”: ,

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ display)

Request Body-JSON Format-

“display”: {

“dId”: ,

“inches”: ,

“reviewText”: ,

“stars”: ,

“productId”:

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ battery)

Request Body-JSON Format-

“battery”: {

“bId”: ,

“capacity”: ,

“reviewText”: ,

“stars”: ,

“productId”:

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ camera)

Request Body-JSON Format-

“camera”: {

“cId”: ,

“resolution”: ,

“reviewText”: ,

“stars”: ,

“productId”:

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ ram)

Request Body-JSON Format-

“ram”: {

“rId”: ,

“ramSize”: ,

“reviewText”: ,

“stars”: ,

“productId”:

}

* Endpoint-Request Mapping (method=POST, https://localhost:portnumber/reviews/ processor)

Request Body-JSON Format-

“processor”: {

“prId”: ,

“prName”: ,

“reviewText”: ,

“stars”: ,

“productId”:

}

* Endpoint-Request Mapping (method=DELETE, https://localhost:portnumber/reviews/ processor/processorId)
* Endpoint-Request Mapping (method=DELETE, https://localhost:portnumber/reviews/ ram/ramId)
* Endpoint-Request Mapping (method=DELETE, http://localhost:portnumber/reviews/ display/displayId)
* Endpoint-Request Mapping (method=DELETE, http://localhost:portnumber/reviews/ camera/cameraId)
* Endpoint-Request Mapping (method=DELETE, http://localhost:portnumber/reviews/ product/productId)
* Endpoint-Request Mapping (method=DELETE, http://localhost:portnumber/reviews/ battery/batteryId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ battery/batteryId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ battery/products/productId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ ram/ramId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ ram/products/productId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ camera/cameraId)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ camera/products/productId)
* Endpoint-Request Mapping (method=GET, <http://localhost:portnumber/reviews/display/displayId>)
* Endpoint-Request Mapping (method=GET, http://localhost:portnumber/reviews/ display/products/productId)
* Endpoint-Request Mapping (method=GET, <http://localhost:portnumber/reviews/processor/processorId>)
* Endpoint-Request Mapping (method=GET, <http://localhost:portnumber/reviews/processor/products/productId>)

1. Review Service:

From this service we can give the endpoints selected by the user. Where user can select GET, POST or DELETE the product reviews. User can get the reviews by using any featureId.

* Endpoint-Request Mapping (method=GET, <http://localhost:portnumber/reviews/products/productId/featureId>)
* Endpoint-Request Mapping (method=POST, <http://localhost:portnumber/reviews/products/productId/featureId>)
* Endpoint-Request Mapping (method=DELETE, <http://localhost:portnumber/reviews/products/productId/featureId>)

1. Eureka Server: Eureka server is an application that holds the information about all client-service applications.

* Each and every microservice will register into the Eureka server and Eureka server knows all the client applications running on each port and IP address (<https://localhost:portnumber>).

1. Config server:

* Config server enables to store all the configuration for multiple microservices for different environments in a Git repository.
* A microservice can connect to the config server and identify itself, and also specify the instance it represents.
* It enables to get required configuration.

1. Zuul Server:

* Zuul server is an API Gateway application.
* It handles all the requests and performs the dynamic routing of microservice applications.
* It works as a front door for all the requests.
* It is also known as Edge server
* Zuul is built to enable dynamic routing, monitoring, resiliency, and security.(<https://localhost:portnumber>).

6. Zipkin server:

* Zipkin server is a java-based distributed tracing system to collect and look up data from distributed systems.
* Too many things could happen when a request to an HTTP application is made.
* A request could include a call to a database engine, to a cache server or other microservice.

**Other API’s:**

* Swagger

**JWT Authentication** : (JSON Web Token)

* The user logs in with their username and password or other security credentials.
* Sends POST request with user credentials.
* Credentials checked against the login database. If valid, send token to the user.
* The token inside the authorization send back to the server.
* The server verifies the JWT signature and extracts the user information from the JWT payload.
* The requested resource is sent to the client.

**JWT Structure** :

JWT consist of 3 parts separated by dots ( . ),

* Header
* Payload
* Signature

**Authentication:**

* **POST request :**

[**https://localhost:portnumber/users**](https://localhost:portnumber/users)

* **GET request :**

[**https://localhost:portnumber/reviews/products**](https://localhost:portnumber/reviews/products)

* **GET request :**

[**https://localhost:portnumber/reviews/ram**](https://localhost:portnumber/reviews/ram)

* **GET request :**

[**https://localhost:portnumber/reviews/camera**](https://localhost:portnumber/reviews/camera)

* **GET request :**

[**https://localhost:portnumber/reviews/display**](https://localhost:portnumber/reviews/display)

* **GET request :**

[**https://localhost:portnumber/reviews/procesor**](https://localhost:portnumber/reviews/procesor)

* **GET request :**

[**https://localhost:portnumber/reviews/battery**](https://localhost:portnumber/reviews/battery)

* **POST request :**

[**https://localhost:portnumber/reviews/product**](https://localhost:portnumber/reviews/product)

* **POST request :**

[**https://localhost:portnumber/reviews/display**](https://localhost:portnumber/reviews/display)

* **POST request :**

[**https://localhost:portnumber/reviews/battery**](https://localhost:portnumber/reviews/battery)

* **POST request :**

[**https://localhost:portnumber/reviews/camera**](https://localhost:portnumber/reviews/camera)

* **POST request :**

[**https://localhost:portnumber/reviews/ram**](https://localhost:portnumber/reviews/ram)

* **POST request :**

[**https://localhost:portnumber/reviews/processor**](https://localhost:portnumber/reviews/processor)

* **DELETE request :** [**https://localhost:portnumber/reviews/processor/processorId**](https://localhost:portnumber/reviews/processor/processorId)
* **DELETE request :**

[**https://localhost:portnumber/reviews/ram/ramId**](https://localhost:portnumber/reviews/ram/ramId)

* **DELETE request :** [**https://localhost:portnumber/reviews/display/displayId**](https://localhost:portnumber/reviews/display/displayId)
* **DELETE request :** [**https://localhost:portnumber/reviews/camera/cameraId**](https://localhost:portnumber/reviews/camera/cameraId)
* **DELETE request :** [**https://localhost:portnumber/reviews/product/productId**](https://localhost:portnumber/reviews/product/productId)
* **DELETE request :** [**https://localhost:portnumber/reviews/battery/batteryId**](https://localhost:portnumber/reviews/battery/batteryId)
* **GET request :** [**https://localhost:portnumber/reviews/battery/batteryId**](https://localhost:portnumber/reviews/battery/batteryId)
* **GET request :** [**https://localhost:portnumber/reviews/battery/products/productsId**](https://localhost:portnumber/reviews/battery/products/productsId)
* **GET request :**

[**https://localhost:portnumber/reviews/ram/ramId**](https://localhost:portnumber/reviews/ram/ramId)

* **GET request :** [**https://localhost:portnumber/reviews/ram/products/productsId**](https://localhost:portnumber/reviews/ram/products/productsId)
* **GET request :** [**https://localhost:portnumber/reviews/camera/cameraId**](https://localhost:portnumber/reviews/camera/cameraId)
* **GET request :** [**https://localhost:portnumber/reviews/camera/products/productsId**](https://localhost:portnumber/reviews/camera/products/productsId)
* **GET request :** [**https://localhost:portnumber/reviews/display/displayId**](https://localhost:portnumber/reviews/display/displayId)

**Browser Server**

1. POST /User / login with username and password 2. Create a JWT
2. Returns JWT to browser
3. Sends the JWT on the Authorization Header 5. Check JWT
4. Get response.

**Database Structure & Design:**

Product

productId(pId)

productName(pName)

productReview(reviewText)

productRating(stars)

User

usedId(uId)

username(uName)

Password

mobileNumber

Battery

bId

capacity

pId(product)

batteryReview(reviewText)

batteryRating(stars)

Display

dId

inches

pId(product)

displayReview(reviewText)

displayRating(stars)

Processor

pId

proceName

pId(product)

processorReview(reviewText)

processorRating(stars)

Camera

cId

resolution

pId(product)

cameraReview(reviewtext)

cameraRating(stars)

Ram

rId

ramSize

pId(product)

ramReview(reviewText)

ramRating(stars)

Processor

pId

proceName

pId(product)

Camera

cId

resolution

pId(product)