The issue seems to be that the structure of the API responses is not aligned with your DealItem model. The responses contain a categoryName and a list of dealItems, but your current code assumes that the API returns a list of DealItem directly.

To resolve this, you need to update your service classes to extract the dealItems from the API responses. Here is how you can update your service classes:

Create a new model class to match the API response structure:

java

Copy code

package com.example.ecommerce.model;

import java.util.List;

public class ApiResponse {

private String categoryName;

private List<DealItem> dealItems;

// Getters and Setters

public String getCategoryName() {

return categoryName;

}

public void setCategoryName(String categoryName) {

this.categoryName = categoryName;

}

public List<DealItem> getDealItems() {

return dealItems;

}

public void setDealItems(List<DealItem> dealItems) {

this.dealItems = dealItems;

}

}

Update the AmazonService, EbayService, and WalmartService to use the ApiResponse model:

java

Copy code

package com.example.ecommerce.service;

import com.example.ecommerce.model.ApiResponse;

import com.example.ecommerce.model.DealItem;

import org.springframework.stereotype.Service;

import org.springframework.web.reactive.function.client.WebClient;

import java.util.List;

import java.util.concurrent.CompletableFuture;

@Service

public class AmazonService {

private final WebClient webClient;

public AmazonService(WebClient.Builder webClientBuilder) {

this.webClient = webClientBuilder.baseUrl("http://localhost:8081").build();

}

public CompletableFuture<List<DealItem>> getDeals(String categoryName) {

return webClient.get()

.uri("/backendserver1/amazon/deals/{categoryName}", categoryName)

.retrieve()

.bodyToMono(ApiResponse.class)

.map(ApiResponse::getDealItems)

.doOnNext(deals -> System.out.println("Amazon Service fetched deals: " + deals))

.toFuture();

}

}

Repeat similar changes for EbayService and WalmartService.

Ensure your DealsService uses the updated methods from the service classes.

By updating the service classes to handle the ApiResponse structure, you should be able to correctly map the data from the API responses to your DealItem model and avoid the null values.

Also, ensure that the JSON fields in your DealItem model match the fields in the API response exactly. If there are any discrepancies in the field names, you might need to use Jackson annotations to map them correctly. For example:

java

Copy code

import com.fasterxml.jackson.annotation.JsonProperty;

public class DealItem {

@JsonProperty("itemid")

private String itemId;

@JsonProperty("productTitle")

private String productTitle;

// Other fields and methods

}

This way, the fields in your model will correctly map to the JSON fields in the API response.

I am using different APIs

<http://localhost:8081/backendserver1/amazon/deals/Smartphones>

gives output:

{

"id": 2,

"categoryName": "Smartphones",

"dealItems": [

{

"itemid": "234567890",

"productTitle": "Samsung Galaxy S23",

"size": "128GB",

"brand": "Samsung",

"imageUrl": "https://example.com/images/samsung\_galaxy\_s23.jpg",

"originalPrice": 799.99,

"discountPercentage": 10,

"discountAmount": 80,

"price": 719.99,

"priceTreatment": "LIST\_PRICE",

"stock": 20,

"dealStartDate": "2024-08-01T10:00:00",

"dealEndDate": "2024-11-01T23:59:59"

},

{

"itemid": "345678901",

"productTitle": "iPhone 14 Pro",

"size": "256GB",

"brand": "Apple",

"imageUrl": "https://example.com/images/iphone\_14\_pro.jpg",

"originalPrice": 999.99,

"discountPercentage": 12,

"discountAmount": 120,

"price": 879.99,

"priceTreatment": "LIST\_PRICE",

"stock": 15,

"dealStartDate": "2024-08-01T10:00:00",

"dealEndDate": "2024-11-01T23:59:59"

}

]

}

And similar outputs for:

<http://localhost:8082/backendserver2/ebay/deals/Smartphones>

and

<http://localhost:8083/backendserver3/walmart/deals/Smartphones>

all these APIs are working fine then what am I doing wrong.. why am I getting null value