Product Catalog Microservice

The Product Catalog is our second part of the incremental project. It serves as the foundation of products that belong to the future e-commerce project. It should provide the e-commerce merchandise information and images about the listed products.

User stories

- As a admin, I want to be able to create a product
- As a admin, I want to be able to update the product that I created
- As a admin, I want to be able to delete a product that I created
- As a guest user, I want to be able to see a list of products
- As a guest user, I want to be able to get a specific product by the id of the product

Non-functional requirements

Product stories usually don't define non-functional requirements, such as security, development principles, technology stack, etc. So let's list them here separately.

- The API should be made using NestJS
- Only authenticated users (admin) can perform operations on products
- Authentication provider should be the one used for the Users microservices
- The REST API should be documented using swagger
- A RESTful API is provided for interacting with the product catalog
- You can use MongoDB but is also optional to switch to Postgres
- Products should be modeled having a category, product type, and variants (refer to next page for more details on modeling a product)
- Products should contain images
- When listing products, the user can filter/search with query params (such as categories / tags, name, variants, createdAt and others)
- All API calls should be tracked with action being made and who made the action which can be consumed by an admin when using the Analytics endpoint. Analytics should work asynchronous and should use node.js events approach

Optional requirements

- Product should use HATEOAS for the REST API
- listing products can also be exposed using GraphQL
- listing one specific product can also be exposed using GraphQL
- at least 80% of unit testing is highly appreciated
- [optional]: a full e2e testing is highly appreciated

Product Model

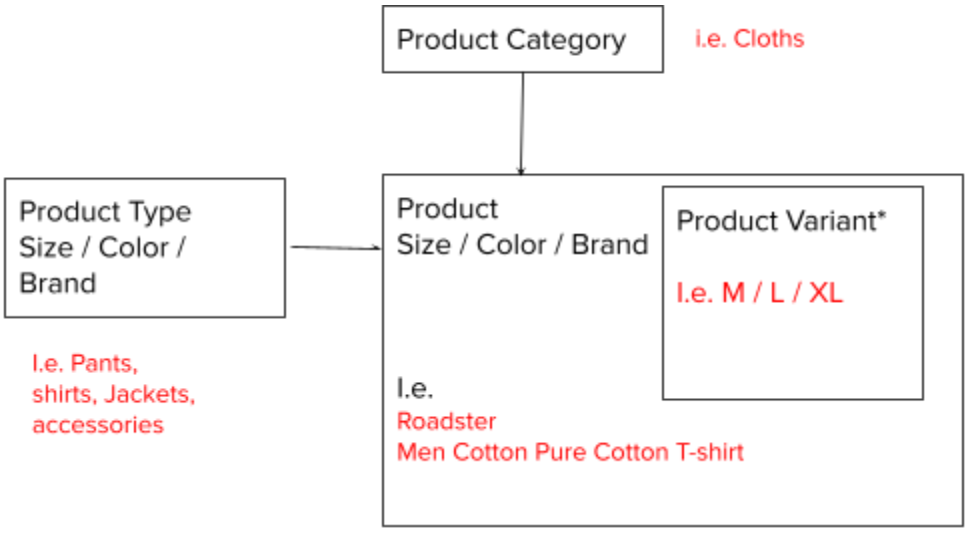
Components be created as resources in Product Catalog service

1. Category: Category is used to group products. You can create categories to group product characteristics, a product may have multiple categories as well.

2. Product Type: A set of attributes which acts as a template for a group of similar products.

3. Product: An actual sellable item with a set of attributes defined by a Product Type. Products are not sellable directly, Instead they act as a parent structure for Product Variants. Each Product must have at least one Product Variant, which is called the Master Variant.

4. Product Variant: A concrete sellable item. Product Variants are generally mapped to specific SKUs. Inventory is modeled per variant.



*An individual Product can have only one Product Type. Multiple Products can use the same Product Type
* A Product can have multiple categories
* A Product should have at least one Variant, called the Main Variant. In case, it can have any number of additional Product Variants as well

Product Categories

Product Categories can be used to logically group products. With Categories one can create multiple classifications of Products for shopping / navigation / searching etc.

Product

Product information is stored in two dimensions: Product Attributes and Product Variants. Product Attributes describe individual properties of a product, and Product Variants describe individually sellable variations of a particular product.

Product variants

Product Variants generally represent a distinct SKU or sellable items.
For example, a clothing store might have a product, "Women's Pants" which has the following variants:

- Red Women's Pants in size 36
- Green Women's Pants in size 38
- Green Women's Pants in size 40

Product catalog endpoints

POST /products
    Created a product

GET /products
    List products

GET /products/:productId
    Get a product per id

PUT /products/:productId
    Fully updates a product per id

PATCH /products/:productId
    Partially updates a product per id

DELETE /products/:productId
    Deletes a product per id

PATCH /products/:productId/variants/:variantId

Partially update a variant from a product per id

DELETE /products/:productId/variants/:variantId
Delete a variant of a product per id

PUT /products/:productId/variants
Batch update/replace variants for a product id

PUT /products/:productId/images/:imageId
Update a image of a product per id

DELETE /products/:productId/iages/:imageId
Delete a image of a product per id

PUT /products/:productId/images
Batch update/replace images of a product