**Endpoints for Product Service**

**Category Management**

1. **POST /categories**
   * **Description**: Create a new category (e.g., "Men", "Supplements").
   * **Request**: { "name": "Category Name", "parentCategoryId": null, "description": "Description" }.
   * **Response**: Details of the created category.
2. **GET /categories**
   * **Description**: Retrieve all categories with their hierarchical structure.
   * **Response**: List of categories and their subcategories.
3. **GET /categories/{id}**
   * **Description**: Get category details by ID.
   * **Response**: Category details.
4. **PUT /categories/{id}**
   * **Description**: Update category details.
   * **Request**: { "name": "Updated Name", "description": "Updated Description" }.
   * **Response**: Updated category details.
5. **DELETE /categories/{id}**
   * **Description**: Delete a category by ID.
   * **Response**: Success or error if subcategories or products exist.

**Product Type Management**

1. **POST /product-types**
   * **Description**: Create a new product type with attributes.
   * **Request**:

json

Copy code

{

"name": "Product Type Name",

"attributes": [

{ "name": "Attribute Name", "type": "Enum/Text/Number", "allowedValues": ["Value1", "Value2"] }

]

}

* + **Response**: Details of the created product type.

1. **GET /product-types**
   * **Description**: Retrieve all product types with attributes.
   * **Response**: List of product types.
2. **GET /product-types/{id}**
   * **Description**: Get product type details by ID.
   * **Response**: Product type details.
3. **PUT /product-types/{id}**
   * **Description**: Update product type details.
   * **Response**: Updated product type details.
4. **DELETE /product-types/{id}**
   * **Description**: Delete a product type if no associated products exist.
   * **Response**: Success or error.

**Product Management**

1. **POST /products**
   * **Description**: Create a product with variants.
   * **Request**:

json

Copy code

{

"name": "Product Name",

"categoryId": 1,

"productTypeId": 1,

"variants": [

{

"variantAttributes": { "Size": "M", "Color": "Red" },

"stock": 50,

"sku": "PRODUCT-M-RED"

}

]

}

* + **Response**: Details of the created product.

1. **GET /products**
   * **Description**: Get all products with filtering and pagination.
   * **Query Params**: categoryId, priceRange, attributes, etc.
   * **Response**: List of products.
2. **GET /products/{id}**
   * **Description**: Get product details by ID, including variants.
   * **Response**: Product details.
3. **PUT /products/{id}**
   * **Description**: Update product details.
   * **Response**: Updated product details.
4. **DELETE /products/{id}**
   * **Description**: Delete a product by ID.
   * **Response**: Success or error.

**Product Variant Management**

1. **POST /products/{id}/variants**
   * **Description**: Add a variant to an existing product.
   * **Request**:

json

Copy code

{

"variantAttributes": { "Size": "L", "Color": "Blue" },

"stock": 30,

"sku": "PRODUCT-L-BLUE"

}

* + **Response**: Details of the created variant.

1. **GET /products/{id}/variants**
   * **Description**: Get all variants of a product.
   * **Response**: List of variants.
2. **PUT /products/{productId}/variants/{variantId}**
   * **Description**: Update variant details.
   * **Response**: Updated variant details.
3. **DELETE /products/{productId}/variants/{variantId}**
   * **Description**: Delete a variant by ID.
   * **Response**: Success.

**Customer Service**

**Endpoints:**

1. **POST /customers/register**
   * **Description**: Register a new customer or manager.
   * **Request**: { "username": "...", "email": "...", "password": "...", "isManager": true/false, ... }
   * **Response**: Success or error.
2. **POST /customers/login**
   * **Description**: Authenticate user and return details for token generation.
   * **Request**: { "username": "user", "password": "pass" }
   * **Response**: User details and roles.
3. **GET /customers/{userId}/details**
   * **Description**: Fetch the profile details of a customer.
   * **Response**: Customer information.

**Order Service**

**Endpoints:**

1. **POST /carts**
   * **Description**: Create a new cart for a customer.
2. **POST /carts/{cartId}/line-items**
   * **Description**: Add a line item to the cart.
   * **Request**: { "productId": "123", "sku": "sku123", "quantity": 2 }
   * **Response**: Updated cart.
3. **DELETE /carts/{cartId}/line-items/{lineItemId}**
   * **Description**: Remove a line item from the cart.
4. **PUT /carts/{cartId}/line-items/{lineItemId}**
   * **Description**: Update the quantity of a line item.
5. **PUT /carts/{cartId}/state**
   * **Description**: Update cart state (e.g., ACTIVE → ORDERED).
6. **POST /orders**
   * **Description**: Create an order from a cart.
7. **PUT /orders/{orderId}/payment-state**
   * **Description**: Update the payment state (PENDING → PAID).
8. **PUT /orders/{orderId}/order-state**
   * **Description**: Update the order state (OPEN → CONFIRMED).
9. **GET /orders/customer/{customerId}**
   * **Description**: Fetch order history for a customer.

**Payment Service**

**Endpoints:**

1. **POST /payments**
   * **Description**: Initiate a payment for an order.
   * **Request**: { "orderId": "123", "amount": 100.0, "paymentMethodInfo": { ... } }
   * **Response**: Payment status.
2. **PUT /payments/{paymentReference}/status**
   * **Description**: Update the payment status (SUCCESS/FAILED).
3. **GET /payments/{key}**
   * **Description**: Retrieve details of a payment using its key.

**Price Service**

**Endpoints:**

1. **POST /prices**
   * **Description**: Add a new price configuration.
   * **Request**: { "productId": "123", "sku": "sku123", "priceAmount": 100.0, "currency": "USD", ... }
2. **PUT /prices/{id}**
   * **Description**: Update an existing price configuration.
3. **DELETE /prices/{id}**
   * **Description**: Delete a price configuration by ID.
4. **GET /prices/{productId}**
   * **Description**: Retrieve all prices for a product by productId.
5. **GET /prices/dynamic**
   * **Description**: Fetch a dynamic price based on productId, sku, and contextual attributes.
   * **Query Parameters**: { "productId": "...", "sku": "...", "currency": "...", ... }

Each service integrates with others as follows:

* **Customer Service → API Gateway**: For token generation and user authentication.
* **Order Service → product Service**: To fetch prices dynamically for cart items.
* **Order Service → Payment Service**: To validate payment success before creating orders.
* **Order Service → customer Service**: To validate customers before creating carts and fetch customer details like email, address etc.
* **API Gateway → All Services**: Centralized authentication, authorization, and routing.