OAS로 API 설계

는 OpenAPI Specification의 약자로, "OpenAPI 명세서" 의미

OpenAPI 또는 OAS(OpenAPI Specification)로 불리며, RESTful API를 정해진 표준 규칙에 따라 API Spec을 json 혹은 yaml 로 표현하는 방식 , 즉 "Restful API 디자인의 정의 방법의 표준"

예

API 설계 예제 YAML

```yaml
openapi: 3.0.3
info:
  title: Sample Ecommerce App
  description: >
    'This is a ***sample ecommerce app API***.  You can find out more about Swagger
at [swagger.io](http://swagger.io).
    Description supports markdown markup. For example, you can use the `inline
code` using back ticks.'
  termsOfService: https://github.com/PacktPublishing/Modern-API-Development-with-
Spring-6-and-Spring-Boot-3/blob/main/LICENSE
  contact:
    name: Packt Support
    url: https://www.packt.com
    email: support@packtpub.com
  license:
    name: MIT
    url: https://github.com/PacktPublishing/Modern-API-Development-with-Spring-6-
and-Spring-Boot-3/blob/main/LICENSE
  version: 1.0.0
externalDocs:
  description: Any document link you want to generate along with API.
  url: http://swagger.io
servers:
  - url: https://ecommerce.swagger.io/v2
tags:
  - name: cart
    description: Everything about cart
    externalDocs:
      description: Find out more (extra document link)
      url: http://swagger.io
  - name: order
    description: Operation about orders
  - name: user
    description: Operations about users
  - name: customer
    description: Operations about user's persona customer
```

```yaml
    - name: address
      description: Operations about user's address
    - name: payment
      description: Operations about payments
    - name: shipping
      description: Operations about shippings
    - name: product
      description: Operations about products
    - name: card
      description: card operation

paths:
  /api/v1/carts/{customerId}:
    get:
      tags:
        - cart
      summary: Returns the shopping cart
      description: Returns the shopping cart of given customer
      operationId: getCartByCustomerId
      parameters:
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: successful operation
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Cart'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Cart'
        404:
          description: Given customer ID doesn't exist
          content: {}
    delete:
      tags:
        - cart
      summary: Delete the shopping cart
      description: Deletes the shopping cart of given customer
      operationId: deleteCart
      parameters:
```

```yaml
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      responses:
        204:
          description: successful operation
        404:
          description: Given customer ID doesn't exist
          content: {}
  /api/v1/carts/{customerId}/items:
    get:
      tags:
        - cart
      summary: Returns the list of products in user's shopping cart
      description: Returns the items in shopping cart of given customer
      operationId: getCartItemsByCustomerId
      parameters:
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: successful operation
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
        404:
          description: Given customer ID doesn't exist
          content: {}
    post:
      tags:
        - cart
      summary: Adds an item in shopping cart
      description: Adds an item to the shopping cart if it doesn't already exist,
or increment quantity by the specified number of items if it does.
      operationId: addCartItemsByCustomerId
```

```yaml
      parameters:
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      requestBody:
        description: Item object
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/Item'
          application/json:
            schema:
              $ref: '#/components/schemas/Item'
      responses:
        201:
          description: Item added successfully
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
        404:
          description: Given customer ID doesn't exist
          content: {}
    put:
      tags:
        - cart
      summary: Replace/add an item in shopping cart
      description: Adds an item to the shopping cart if it doesn't already exist,
or replace with given item if it does.
      operationId: addOrReplaceItemsByCustomerId
      parameters:
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      requestBody:
        description: Item object
        content:
```

```yaml
              application/xml:
                schema:
                  $ref: '#/components/schemas/Item'
              application/json:
                schema:
                  $ref: '#/components/schemas/Item'
      responses:
        201:
          description: Item added successfully
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
        404:
          description: Given customer ID doesn't exist
          content: {}
/api/v1/carts/{customerId}/items/{itemId}:
  get:
    tags:
      - cart
    summary: Returns given item from user's shopping cart
    description: Returns given item from shopping cart of given customer
    operationId: getCartItemsByItemId
    parameters:
      - name: customerId
        in: path
        description: Customer Identifier
        required: true
        schema:
          type: string
      - name: itemId
        in: path
        description: Item (product) Identifier
        required: true
        schema:
          type: string
    responses:
      200:
        description: If item exists in cart
        content:
          application/xml:
            schema:
              type: array
```

```yaml
                items:
                  $ref: '#/components/schemas/Item'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Item'
        404:
          description: Given item (product) ID doesn't exist
          content: {}
    delete:
      tags:
        - cart
      summary: Delete the item from shopping cart
      description: Deletes the item from shopping cart of given customer
      operationId: deleteItemFromCart
      parameters:
        - name: customerId
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
        - name: itemId
          in: path
          description: Item (product) Identifier
          required: true
          schema:
            type: string
      responses:
        202:
          description: Accepts the request, regardless of whether the specified
item exists in the cart or not.
  /api/v1/orders:
    post:
      tags:
        - order
      summary:  Creates a new order for the given order request
      description: Creates a new order for the given order request.
      operationId: addOrder
      requestBody:
        description: New Order Request object
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/NewOrder'
          application/json:
            schema:
              $ref: '#/components/schemas/NewOrder'
      responses:
```

```yaml
      201:
        description: Order added successfully
        content:
          application/xml:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Order'
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Order'
      406:
        description: If payment is not authorized.
        content: {}
  get:
    tags:
      - order
    summary: Returns the orders of given user
    description: Returns orders of given user
    operationId: getOrdersByCustomerId
    parameters:
      - name: customerId
        in: query
        description: Customer Identifier
        required: true
        schema:
          type: string
    responses:
      200:
        description: If order exists.
        content:
          application/xml:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Order'
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Order'
      404:
        description: Order doesn't exist for given user.
        content: {}
/api/v1/orders/{id}:
  get:
    tags:
      - order
```

```yaml
        summary: Returns the order of given order ID
        description: Returns orders of given order ID
        operationId: getOrdersByOrderId
        parameters:
          - name: id
            in: path
            description: Order Identifier
            required: true
            schema:
              type: string
        responses:
          200:
            description: If order exists.
            content:
              application/xml:
                schema:
                  $ref: '#/components/schemas/Order'
              application/json:
                schema:
                  $ref: '#/components/schemas/Order'
          404:
            description: Order doesn't exist for given user.
            content: {}
  /api/v1/customers:
    get:
      tags:
        - customer
      summary: Returns all customers
      description: Returns all customers, or empty collection if no use found
      operationId: getAllCustomers
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/User'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/User'
  /api/v1/customers/{id}:
    get:
      tags:
        - customer
      summary: Returns a customer
      description: Returns a customers identifiable by given ID
```

```yaml
        operationId: getCustomerById
        parameters:
          - name: id
            in: path
            description: Customer Identifier
            required: true
            schema:
              type: string
        responses:
          200:
            description: For successful fetch.
            content:
              application/xml:
                schema:
                  $ref: '#/components/schemas/User'
              application/json:
                schema:
                  $ref: '#/components/schemas/User'
    delete:
      tags:
        - customer
      summary: Deletes the customer
      description: Deletes the customer identifiable by given ID
      operationId: deleteCustomerById
      parameters:
        - name: id
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
      responses:
        202:
          description: Request accepted, returns this status even if user does not
exist
          content: {}
  /api/v1/customers/{id}/cards:
    get:
      tags:
        - customer
      summary: Returns all customer's cards
      description: Returns all customer's cards based on given customer ID
      operationId: getCardsByCustomerId
      parameters:
        - name: id
          in: path
          description: Customer Identifier
          required: true
          schema:
            type: string
```

```yaml
    responses:
      200:
        description: For successful fetch.
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/Card'
          application/json:
            schema:
              $ref: '#/components/schemas/Card'
/api/v1/customers/{id}/addresses:
  get:
    tags:
      - customer
    summary: Returns all customer's addresses
    description: Returns all customer's addresses based on given customer ID
    operationId: getAddressesByCustomerId
    parameters:
      - name: id
        in: path
        description: Customer Identifier
        required: true
        schema:
          type: string
    responses:
      200:
        description: For successful fetch.
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/Address'
          application/json:
            schema:
              $ref: '#/components/schemas/Address'
/api/v1/addresses:
  get:
    tags:
      - address
    summary: Returns all user's addresses
    description: Returns all user's addresses, else empty collection
    operationId: getAllAddresses
    responses:
      200:
        description: For successful fetch.
        content:
          application/xml:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Address'
```

```yaml
                application/json:
                  schema:
                    type: array
                    items:
                      $ref: '#/components/schemas/Address'
    post:
      tags:
        - address
      summary: Creates a new user addresses
      description: Creates a new user addresses. Does nothing if address already
exists.
      operationId: createAddress
      requestBody:
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/AddAddressReq'
          application/json:
            schema:
              $ref: '#/components/schemas/AddAddressReq'
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Address'
            application/json:
              schema:
                $ref: '#/components/schemas/Address'
  /api/v1/addresses/{id}:
    get:
      tags:
        - address
      summary: Returns user's address
      description: Returns user's address based on given address ID.
      operationId: getAddressesById
      parameters:
        - name: id
          in: path
          description: address Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
```

```yaml
                  $ref: '#/components/schemas/Address'
            application/json:
              schema:
                $ref: '#/components/schemas/Address'
    delete:
      tags:
        - address
      summary: Deletes user's address
      description: Deletes user's address based on given address ID.
      operationId: deleteAddressesById
      parameters:
        - name: id
          in: path
          description: address Identifier
          required: true
          schema:
            type: string
      responses:
        202:
          description: Accepts the deletion request and perform deletion. If ID
does not exist, does nothing.
          content: {}
  /api/v1/cards:
    get:
      tags:
        - card
      summary: Returns all user's cards
      description: Returns all user's cards, else empty collection
      operationId: getAllCards
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Card'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Card'
    post:
      tags:
        - card
      summary: Creates a  new card
      description: Creates a new card. or replaces the existing one
      operationId: registerCard
      requestBody:
```

```yaml
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/AddCardReq'
            application/json:
              schema:
                $ref: '#/components/schemas/AddCardReq'
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Card'
            application/json:
              schema:
                $ref: '#/components/schemas/Card'
  /api/v1/cards/{id}:
    get:
      tags:
        - card
      summary: Returns user's card
      description: Returns user's card based on given card ID.
      operationId: getCardById
      parameters:
        - name: id
          in: path
          description: card Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Card'
            application/json:
              schema:
                $ref: '#/components/schemas/Card'
    delete:
      tags:
        - card
      summary: Deletes card's address
      description: Deletes card's address based on given card ID.
      operationId: deleteCardById
      parameters:
        - name: id
          in: path
```

```yaml
          description: card Identifier
          required: true
          schema:
            type: string
      responses:
        202:
          description: Accepts the deletion request and perform deletion. If ID
does not exist, does nothing.
          content: {}
  /api/v1/payments:
    post:
      tags:
        - payment
      summary: Authorize a payment request
      description: Authorize a payment request.
      operationId: authorize
      requestBody:
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/PaymentReq'
          application/json:
            schema:
              $ref: '#/components/schemas/PaymentReq'
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Authorization'
            application/json:
              schema:
                $ref: '#/components/schemas/Authorization'
    get:
      tags:
        - payment
      summary: Returns the payment authorization
      description: Return the payment authorization for the specified order
      operationId: getOrdersPaymentAuthorization
      parameters:
        - name: id
          in: query
          description: Order Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: For successful fetch.
```

```yaml
            content:
              application/xml:
                schema:
                  type: array
                  items:
                    $ref: '#/components/schemas/Authorization'
              application/json:
                schema:
                  type: array
                  items:
                    $ref: '#/components/schemas/Authorization'
  /api/v1/shipping:
    post:
      tags:
        - shipping
      summary: Ship the specified shipping request
      description: Ship the specified shipping request
      operationId: shipOrder
      requestBody:
        content:
          application/xml:
            schema:
              $ref: '#/components/schemas/ShippingReq'
          application/json:
            schema:
              $ref: '#/components/schemas/ShippingReq'
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Authorization'
            application/json:
              schema:
                $ref: '#/components/schemas/Authorization'
    get:
      tags:
        - shipping
      summary: Return the Shipment
      description: Return the Shipment for the specified order
      operationId: getShipmentByOrderId
      parameters:
        - name: id
          in: query
          description: Order Identifier
          required: true
          schema:
            type: string
      responses:
```

```yaml
      200:
        description: For successful fetch.
        content:
          application/xml:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Shipment'
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Shipment'
/api/v1/products:
  get:
    tags:
      - product
    summary: Returns all the matched products
    description: Returns the products that matches the given query criteria
    operationId: queryProducts
    parameters:
      - name: tag
        in: query
        description: Product tag
        required: false
        schema:
          type: string
      - name: name
        in: query
        description: Product name
        required: false
        schema:
          type: string
      - name: page
        in: query
        description: Query page number
        required: false
        schema:
          type: integer
          format: int32
          default: 1
      - name: size
        in: query
        description: Query page size
        required: false
        schema:
          type: integer
          format: int32
          default: 10
    responses:
```

```yaml
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Product'
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Product'
  /api/v1/products/{id}:
    get:
      tags:
        - product
      summary: Returns a product
      description: Returns the product that matches the given product ID
      operationId: getProduct
      parameters:
        - name: id
          in: path
          description: Product Identifier
          required: true
          schema:
            type: string
      responses:
        200:
          description: For successful fetch.
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/Product'
            application/json:
              schema:
                $ref: '#/components/schemas/Product'
components:
  schemas:
    Cart:
      description: Shopping Cart of the user
      type: object
      properties:
        customerId:
          description: Id of the customer who possesses the cart
          type: string
        items:
          description: Collection of items in cart.
          type: array
          items:
```

```yaml
          $ref: '#/components/schemas/Item'
    Item:
      description: Items in shopping cart
      type: object
      properties:
        id:
          description: Item Identifier
          type: string
        quantity:
          description: The item quantity
          type: integer
          format: int32
        unitPrice:
          description: The item's price per unit
          type: number
          format: double
    Order:
      description: Represents an order
      type: object
      properties:
        id:
          description: Order identifier
          type: string
        customer:
          $ref: '#/components/schemas/User'
        address:
          $ref: '#/components/schemas/Address'
        card:
          $ref: '#/components/schemas/Card'
        date:
          description: Order's data and time details
          type: string
          format: date-time
        items:
          description: Collection of order items.
          type: array
          items:
            $ref: '#/components/schemas/Item'
        total:
          description: Order total
          type: number
          format: double
        payment:
          $ref: '#/components/schemas/Payment'
        shipment:
          $ref: '#/components/schemas/Shipment'
        status:
          description: Order Status
          type: string
          enum:
```

```yaml
                - CREATED
                - PAID
                - SHIPPED
                - PAYMENT_FAILED
                - SHIPMENT_FAILED
                - COMPLETED
      xml:
        name: Order
    AddAddressReq:
      allOf:
        - $ref: '#/components/schemas/Address'
        - type: object
          properties:
            userId:
              type: string

      xml:
        name: AddAddressReq
    Address:
      type: object
      properties:
        number:
          description: house of flat number
          type: string
        residency:
          description: Socieity or building name
          type: string
        street:
          description: street name
          type: string
        city:
          description: city name
          type: string
        state:
          description: state name
          type: string
        country:
          description: country name
          type: string
        pincode:
          description: postal code
          type: string
      xml:
        name: Address
    Card:
      type: object
      properties:
        cardNumber:
          description: Card Number
          type: string
```

```yaml
      expires:
        description: Expiration date
        type: string
      ccv:
        description: CCV code
        type: string
    xml:
      name: Card
AddCardReq:
  description: Request object for new card registration.
  allOf:
    - $ref: '#/components/schemas/Card'
    - type: object
      properties:
        userId:
          type: object
    xml:
      name: AddCardReq
Payment:
  type: object
  properties:
    authorized:
      description: Flag that specified whether payment is authorized or not
      type: boolean
    message:
      description: Approval or rejection message
      type: string
    xml:
      name: Payment
Shipment:
  type: object
  properties:
    orderId:
      description: Order Identifier
      type: string
    carrier:
      description: Shipping Carrier
      type: string
    trackingNumber:
      description: Shipping Tracking Number
      type: string
    estDeliveryDate:
      description: Estimated Delivery Date
      type: string
      format: date
    xml:
      name: Shipment
ShippingReq:
  description: Contains information required for Shipping request
  type: object
```

```yaml
      properties:
        orderId:
          description: Order Identifier
          type: string
        address:
          $ref: '#/components/schemas/Address'
        itemCount:
          description: The number of items in the order
          type: integer
          format: int32
      xml:
        name: ShippingReq

  User:
    type: object
    properties:
      id:
        type: integer
        format: int64
      username:
        type: string
      firstName:
        type: string
      lastName:
        type: string
      email:
        type: string
      password:
        type: string
      phone:
        type: string
      userStatus:
        type: integer
        description: User Status
        format: int32
    xml:
      name: User
  NewOrder:
    description: Contains the new order request information
    type: object
    properties:
      customer:
        #description: URI that should be used to fetch the customer
        $ref: '#/components/schemas/URI'
      address:
        #description: URI that should be used to fetch the address
        $ref: '#/components/schemas/URI'
      card:
        #description: URI that should be used to fetch the payment card
        $ref: '#/components/schemas/URI'
```

```yaml
          items:
            #description: URI that should be used to fetch the items from shopping
cart
            $ref: '#/components/schemas/URI'
        xml:
          name: NewOrder
    URI:
      type: object
      xml:
        name: URI
    Authorization:
      type: object
      properties:
        orderId:
          description: Order Identification
          type: string
        time:
          description: Timestamp when this authorization was created
          type: string
          format: date-time
        authorized:
          description: Flat that specify whether the payment is authorized
          type: boolean
        message:
          description: Approavl or rejection message
          type: string
        error:
          description: Processing error description, if any
          type: string
      xml:
        name: Authorization
    PaymentReq:
      description: Contains the payment request information
      type: object
      properties:
        orderId:
          description: Order Identifier
          type: string
        customer:
          $ref: '#/components/schemas/CustomerInfoOnCard'
        address:
          $ref: '#/components/schemas/Address'
        card:
          $ref: '#/components/schemas/Card'
        amount:
          description: Payment amount
          type: number
          format: double
      xml:
        name: PaymentReq
```

```yaml
    CustomerInfoOnCard:
      description: Customer information required for payment processing
      type: object
      properties:
        firstName:
          description: Customer first name
          type: string
        lastName:
          description: Customer last name
          type: string
      xml:
        name: CustomerInfoOnCard
    Product:
      description: Product information
      type: object
      properties:
        id:
          description: Product identifier
          type: string
        name:
          description: Product Name
          type: string
        description:
          description: Prodcut's description
          type: string
        imageUrl:
          description: Product image's URL
          type: string
        price:
          description: Product price
          type: number
          format: double
        count:
          description: Product count
          type: integer
          format: int32
        tag:
          description: Tags associated with the product
          type: array
          uniqueItems: true
          items:
            type: string
```

<https://editor.swagger.io/> 사이트를 통하여 확인

⟲Revision #5
★Created 2 February 2025 12:30:26 by Admin
✎Updated 2 February 2025 12:38:52 by Admin