Design a RESTful API for an online store

Assumption:

1- Considering the products are not segregated by the category in the online store. Introducing the category for the product will increase the complexity and it will involve a lot's of consideration for designing the Application and the Database Schema.

For making a simple interfaces for online store which only deals in products Add, delete, Edit, Search, I have removed the category association with the product.

In Real scenario, we need to consider the category association with product.

Content:

1- Api Design for product

2- Data Contract Between Client & Server

3- Authentication

4- Database Design

**1- Api Design for product :**

For an online store, we are providing the below functionality :

**a- Adding the Product:**

In this, we will be creating a brand new product item in our products collection database.

**Endpoint:**

**POST /products**

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

**Payload:**

{

"title": "abc",

"description": "This is used for xyz",

"price": {

"currency": "INR",

"rate": 100

},

"qty": 10

}

**Response:**

For Authorized user who has access to create products, will have the response like:

StatusCode: 200 OK

{

"reqId": "9bc27360-33a2-11e7-98dc-19e76b750bf0",

"id": "9bc2c180-33a2-11e7-98dc-19e76b750bf0",

"status": "success"

}

**For UnAuthorized user, response :**

i- StatusCode: 401 UnAuthorized

{

"reqId": "15339f80-33a3-11e7-98dc-19e76b750bf0",

"errorType": "Unauthorized",

"name": "ApiError",

"messages": "Authentication credentials missing or incorrect",

"innerError": "",

"statusCode": 401

}

**If the payload is not valid while creating new product:**

i- if qty is negative while adding a new product

**POST /products**

**payload:**

{

"title": "abc",

"description": "sdgsg",

"price": {

"currency": "INR",

"rate": 1

},

"qty": -1

}

**Response:**

**BadRequest | StatusCode- 400**

{

"reqId": "c27f1d70-4791-11e7-9b24-55dfe4976fc6",

"statusCode": 400,

"errorType": "ValidationError",

"name": "ApiError",

"messages": "Bad Request",

"innerError": [

{

"field": "data.qty",

"message": "is less than minimum"

}

]

}

ii- if rate is negative of zero while adding a new product

**POST /products**

**payload:**

{

"title": "abc",

"description": "sdgsg",

"price": {

"currency": "INR",

"rate": 0

},

"qty": 10

}

**Response:**

**BadRequest | StatusCode- 400**

{

"reqId": "2e630c90-4792-11e7-9b24-55dfe4976fc6",

"statusCode": 400,

"errorType": "ValidationError",

"name": "ApiError",

"messages": "Bad Request",

"innerError": [

{

"field": "data.price.rate",

"message": "is less than minimum"

}

]

}

iii- if mandatory field like qty is not supplied while adding a new product

**POST /products**

**payload:**

{

"title": "abc",

"description": "sdgsg",

"price": {

"currency": "INR",

"rate": 100

}

}

**Response:**

**BadRequest | StatusCode- 400**

{

"reqId": "739bd800-4792-11e7-9b24-55dfe4976fc6",

"statusCode": 400,

"errorType": "ValidationError",

"name": "ApiError",

"messages": "Bad Request",

"innerError": [

{

"field": "data.qty",

"message": "is required"

}

]

}

iv- If additional properties are supplied in payload

**POST /products**

**payload:**

{

"title": "abc",

"description": "sdgsg",

"price": {

"currency": "INR",

"rate": 100

},

"qty": 10,

"category": "xyz"

}

**Response:**

**BadRequest | StatusCode- 400**

{

"reqId": "d7b8a890-4792-11e7-9b24-55dfe4976fc6",

"statusCode": 400,

"errorType": "ValidationError",

"name": "ApiError",

"messages": "Bad Request",

"innerError": [

{

"field": "data",

"message": "has additional properties"

}

]

}

v- If enum values for currency is not in **INR** of **Rs**.

**POST /products**

**payload:**

{

"title": "abc",

"description": "sdgsg",

"price": {

"currency": "USD",

"rate": 100

},

"qty": 10

}

**Response:**

**BadRequest | StatusCode- 400**

{

"reqId": "3ecebbf0-4793-11e7-9b24-55dfe4976fc6",

"statusCode": 400,

"errorType": "ValidationError",

"name": "ApiError",

"messages": "Bad Request",

"innerError": [

{

"field": "data.price.currency",

"message": "must be an enum value"

}

]

}

**b- Retrieving All Products:**

This will retrieve all products from the products collection.

**Endpoint:**

**GET /products**

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

**Response:**

In the Response body, product data will be on data key.

**For Authorized User:**

StatusCode 200 OK

{

"reqId": "cf7b2750-33a3-11e7-98dc-19e76b750bf0",

"status": "success",

"data": [

{

"\_id": "3f146070-335b-11e7-8442-bbd709bdb62b",

"title": "aabc",

"description": "",

"price": {

"currency": "INR",

"rate": 10

},

qty: 100

}

]

}

**For UnAuthorized user, response :**

i- StatusCode: 401 UnAuthorized

{

"reqId": "15339f80-33a3-11e7-98dc-19e76b750bf0",

"errorType": "Unauthorized",

"name": "ApiError",

"messages": "Authentication credentials missing or incorrect",

"innerError": "",

"statusCode": 401

}

**c- Retrieving Product By Id:**

This will retrieve the product by it's id.

**Endpoint:**

**GET /products/3f146070-335b-11e7-8442-bbd709bdb62b**

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

**Response:**

In the Response body, product data will be on data key.

For Authorized User:

StatusCode 200 OK

{

"reqId": "73e80f20-4785-11e7-a4ed-3f75c9312a5a",

"status": "success",

"data": [

{

"\_id": "1b6e8fc0-462e-11e7-bd48-cb84ab42347f",

"title": "aabc",

"description": "This is for xyz",

"price": {

"currency": "INR",

"rate": 10

},

"qty": 5

}

]

}

**For UnAuthorized user, response :**

i- StatusCode: 401 UnAuthorized

{

"reqId": "15339f80-33a3-11e7-98dc-19e76b750bf0",

"errorType": "Unauthorized",

"name": "ApiError",

"messages": "Authentication credentials missing or incorrect",

"innerError": "",

"statusCode": 401

}

**d- Edit Product By Id:**

This will update the product details.

**Endpoint:**

**PUT /products/3f146070-335b-11e7-8442-bbd709bdb62b**

**Payload:**

{

"title": "zzzzzz",

"description": "sdgsg",

"price": {

"currency": "INR",

"rate": 1000

},

"qty": 10

}

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

**Response:**

In the Response body, it will have status 'success' on Http method PUT for that it.

**For Authorized User:**

StatusCode 200 OK

{

"reqId": "97ae6fc0-33b8-11e7-80e9-7dbcd69c6941",

"id": "3f146070-335b-11e7-8442-bbd709bdb62b",

"status": "success"

}

**e- Delete Product By Id:**

This will delete the product details.

**Endpoint:**

**DELETE /products/ 3f146070-335b-11e7-8442-bbd709bdb62b**

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

In the Response body, it will have status 'success' on Http method DELETE for that it.

For Authorized User:

StatusCode 200 OK

{

"reqId": "97ae6fc0-33b8-11e7-80e9-7dbcd69c6941",

"id": "3f146070-335b-11e7-8442-bbd709bdb62b",

"status": "success"

}

**f- Add/Remove Product Qty By Id:**

**Add Qty:**

This will increment the product qty.(Qty > 0)

**Endpoint:**

**PATCH**

**/products/3f146070-335b-11e7-8442-bbd709bdb62b/addRemoveQty**

**Payload:**

{

'qty': 5

}

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

For Authorized User:

StatusCode 200 OK

{

"reqId": "97ae6fc0-33b8-11e7-80e9-7dbcd69c6941",

"id": "3f146070-335b-11e7-8442-bbd709bdb62b",

"status": "success"

}

**Remove Qty:**

This will decrement the product qty. (Qty < 0)

**Endpoint:**

**PATCH**

**/products/3f146070-335b-11e7-8442-bbd709bdb62b/addRemoveQty**

**Payload:**

{

'qty': -5

}

**Header:**

**{"Authorization":"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwcmFjdGljZU5hbWUiOiJjYW50YWhlYWx0aCIsImlhdCI6MTQ4MjM5NjE2NSwiZXhwIjoxNTEzOTMyMTY1fQ.9UsC8BzokREFOFBbGFKVNzBIu15W\_f7nKPO\_4g1XUGk"}**

For Authorized User:

StatusCode 200 OK

{

"reqId": "97ae6fc0-33b8-11e7-80e9-7dbcd69c6941",

"id": "3f146070-335b-11e7-8442-bbd709bdb62b",

"status": "success"

}

**2- Data Contract Between Client & Server**

We have exposed the endpoints between client and server. Clients needs to comply with the contact.

**3- Authentication**

We have implemented JSON WEB TOKEN (Jwt)– Auth0 implementation for Authentication.

For Generating the Token, client can send the valid userId to the server and server will check the user and it will generate the Token if userId is valid.

**Endpoint:**

**POST /generateToken**

**Payload:**

**{**

**"userId": "shaktiskm@gmail.com"**

**}**

**Response:**

StatusCode 200 OK

{

"reqId": "0b722e30-33bc-11e7-80e9-7dbcd69c6941",

"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbElkIjoic2hha3RpIiwiaWF0IjoxNDk0MjI2ODM5LCJleHAiOjE0OTQyNjI4Mzl9.VGNFRmlvQt8B7sXv9JCI\_gdRTApSv0sUWNGhEpV1-pM",

"status": "success"

}

\* While generating the Access Token, we should to pass the JWT options for expiresIn, which will expire the token after certain time and will enforce the client to ask for new Token.

options = { **"expiresIn"**: **"10d"** }

**If Token is not provided or invalid token is provided by the client in the Authorization Header, then Response:**

ii- StatusCode: 401 UnAuthorized

{

"reqId": "15339f80-33a3-11e7-98dc-19e76b750bf0",

"errorType": "Unauthorized",

"name": "ApiError",

"messages": "Authentication credentials missing or incorrect",

"innerError": "",

"statusCode": 401

}

**4- Database Design**

A Single collection **products** is used on **MongoDB** database. This is choosen for making the flexible schema and it will be modeled according to the business needs.

If we are considering the categories, then we can have some more collections and can do the linking using referencing or embedding depending upon the business need.

**Collection:**

products:

document -->

{

'\_id': '15339f80-33a3-11e7-98dc-19e76b750bf0',

'title': 'zzzzzz',

'description': 'This is for xyz',

'price': {

'currency': 'INR',

'rate': 100

},

'qty': 10

}