HROne Backend Intern Hiring Task

Deadline : 11:59 pm IST - 21st July, 2025

Optimal time to learn, solve and code : 2 - 4 hours

Problem Statement

You need to create a sample backend application in FastAPI and Python. The challenge assumes you have basic knowledge of Python and some knowledge of Flask / Django / FastAPI. You have to learn the basics of FastAPI and build an application with the APIs, and save the data in MongoDB.

Brief

You are building an ecommerce application like Flipkart/Amazon. You need to build the following APIs -

Create Products API

Endpoint - **/products**
Method - **POST**
Request Body -

```
{
  "name": "string",
  "price": 100.0,
  "sizes": [
    {
      "size": "string",
      "quantity": 0
    }
  ]
}
```

Response Body -

```
{
    "id": "1234567890"
}
```

Return Status Code - **201 (CREATED)**

List Products API

Endpoint - **/products**
Method - **GET**
Query Parameters (all optional) to filter / search on the listing -

- Name (can include regex / partial search)
- size =large → Should be able to filter, for those products which have a size=large.
- limit → number of documents to return
- offset → The number of documents to skip while paginating (sorted by _id)

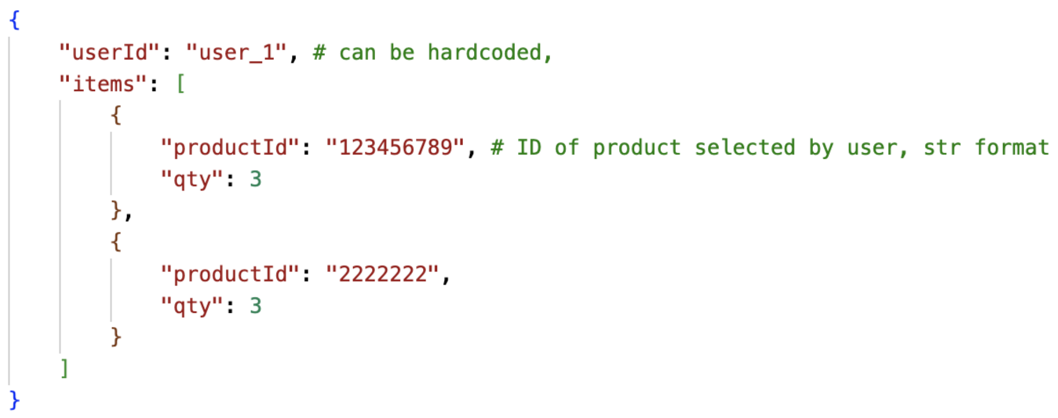
Response Body -

```
{
    "data": [
        {
            "id": "12345",
            "name": "Sample",
            "price": 100.0
            # No sizes in output
        },
        {
            "id": "12346",
            "name": "Sample 2",
            "price": 10.0
        },
        # ...
    ],
    "page": { # for pagination
        "next": "10", # next page starting index
        "limit": 0, # number of records in current page
        "previous": -10  # previous page starting index
    }
}
```

Return Status Code - **200 (OK)**

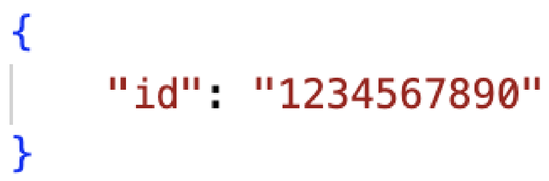
Create Order API

Endpoint - **/orders**
Method - **POST**
Request Body -



```
{
    "userId": "user_1", # can be hardcoded,
    "items": [
        {
            "productId": "123456789", # ID of product selected by user, str format
            "qty": 3
        },
        {
            "productId": "2222222",
            "qty": 3
        }
    ]
}
```

Response Body -



```
{
    "id": "1234567890"
}
```

Return Status Code - **201 (CREATED)**

Get List of Orders

Endpoint - **/orders/<user_id>** → URL Parameter
Method - **GET**
Query Parameters (all optional) to filter / search on the listing -

- limit → number of documents to return
- offset → The number of documents to skip while paginating (sorted by _id)

Response Body -

```
{
    "data": [
        {
            "id": "12345", # Order ID
            "items": [
                {
                    "productDetails": { # we need to join/lookup the product details at query time.
                        "name": "Sample Product",
                        "id": "123456"
                    },
                    "qty": 3
                },
                {
                    "productDetails": {
                        "name": "Sample Product",
                        "id": "123456"
                    },
                    "qty": 3
                }
            ],
            "total": 250.0
        },
        # ... more records
    ],
    "page": { # for pagination
        "next": "10", # next page starting index
        "limit": 0, # number of records in current page
        "previous": -10  # previous page starting index
    }
}
```

Return Status Code - **200 (OK)**

Tech Stack Allowed

- Python - **FastAPI** - Mandatory to be python 3 (3.10 / 3.11 / 3.12 / 3.13)
- Use **MongoDB** as a database with Pymongo or Motor (You can use MongoDB Atlas M0 **free cluster**)

How will we Judge

- Code completeness
    - Should be able to test by running → **This will be judged by an automation script**, matching your URLs, Methods, Request and Response Bodies with given spec above. If your application does not return data in the given output format above, it would be termed as the wrong answer for that scenario.
- Code clarity
    - Cleanliness
    - Well Formatted
    - Documentation
- Structure of APIs created
    - As per given request / response formats.
- Structure of MongoDB Collections / Database models

    - Data models - how you are storing the information in MongoDB or your database.
    - Your queries to MongoDB - how optimised they are (could be fairly good, not the best but certainly not very very performant hit)
    - Structuring relationships and lookups/joins in MongoDB

How to submit

- Push **deploy your code** to Render / Railway **FREE PLAN** and share the link in **this form**.
    - Please make sure you submit the base URL of your application - For example if your products API is such as `https://example.com/something/products`, you should only enter `https://example.com/something` (without the extra slash).
    - Our automatic script based testing will **append** /products to your URL to hit the API.
- You can use **MongoDB Atlas M0 FREE PLAN** for the database to connect your application.
- Make sure you add a README file to somewhat explain your code and structure.
- Make sure your github code repository is **public** or access is given to user **shreybatra** for automated testing.