Online Book Store Project

Part 1

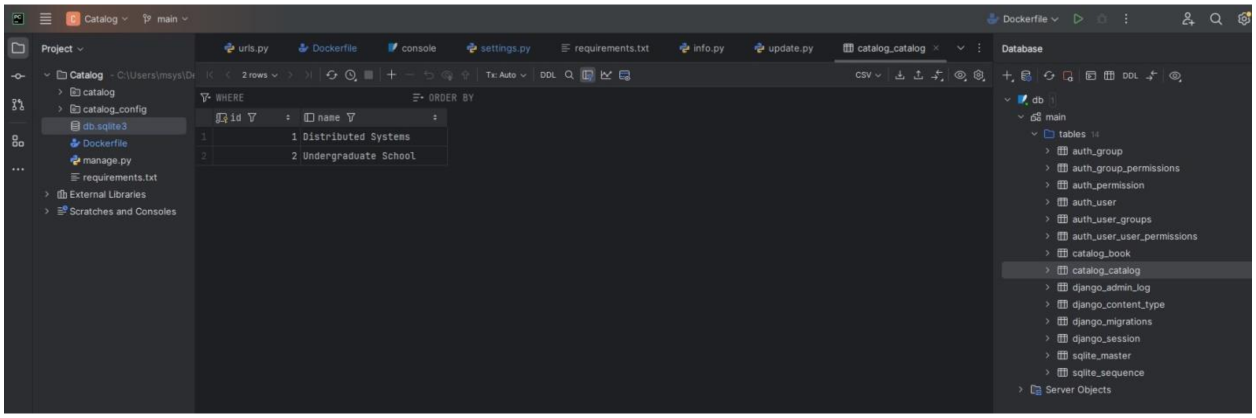
Tala Zayed and Dima Eid

We have implemented this project using Django framework is a powerful high-level framework, it provides an easy way to define models of data, and it is enabling the setup of RESTful API endpoints, easy communication between services and the data can be accessed and updated smoothly across different service, scalability of adding new features and replicas.

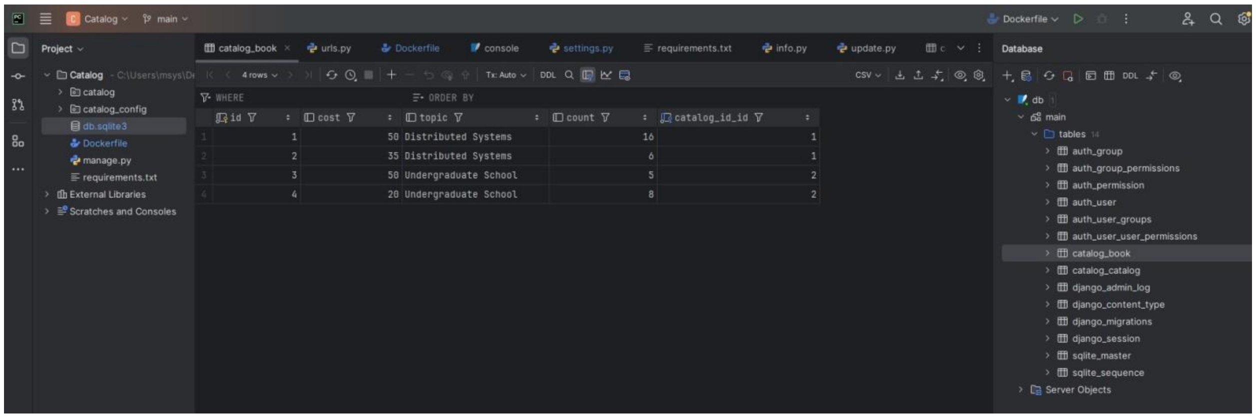
We could've used lighter framework like Flask, but we want to practice Django since we are using as backend framework in graduation project, and we have already managed its setups.

The files structure contain routs for URLs of APIs, controllers for handling the requests, methods to do the functionality on models of data. This structure is to implement clean separation of concerns, this keeps separation between requests handling, functionality and data representation. This modular approach allows for easier testing, debugging, and future scaling, as each component can be worked on independently without impacting others.

Database

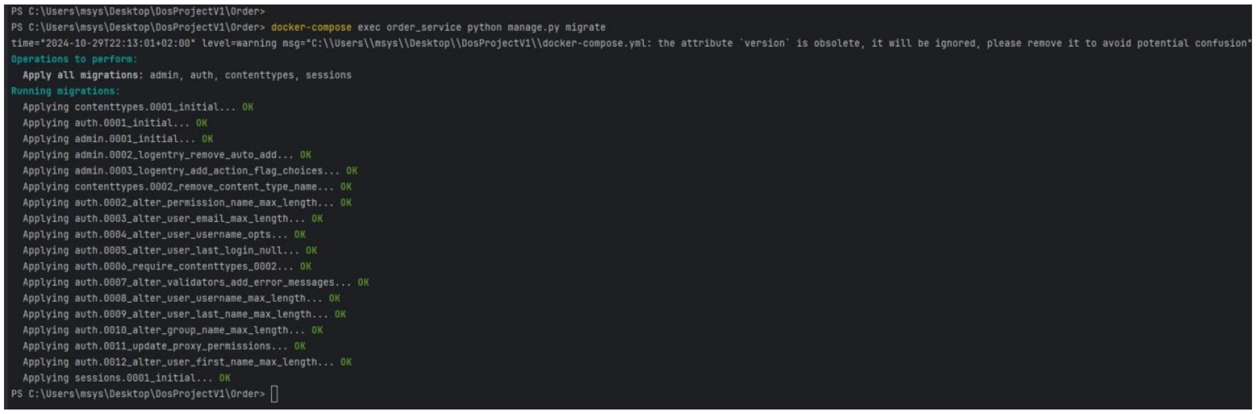


id	name
1	Distributed Systems
2	Undergraduate School



id	cost	topic	count	catalog_id_id
1	50	Distributed Systems	16	1
2	35	Distributed Systems	6	1
3	50	Undergraduate School	5	2
4	20	Undergraduate School	8	2

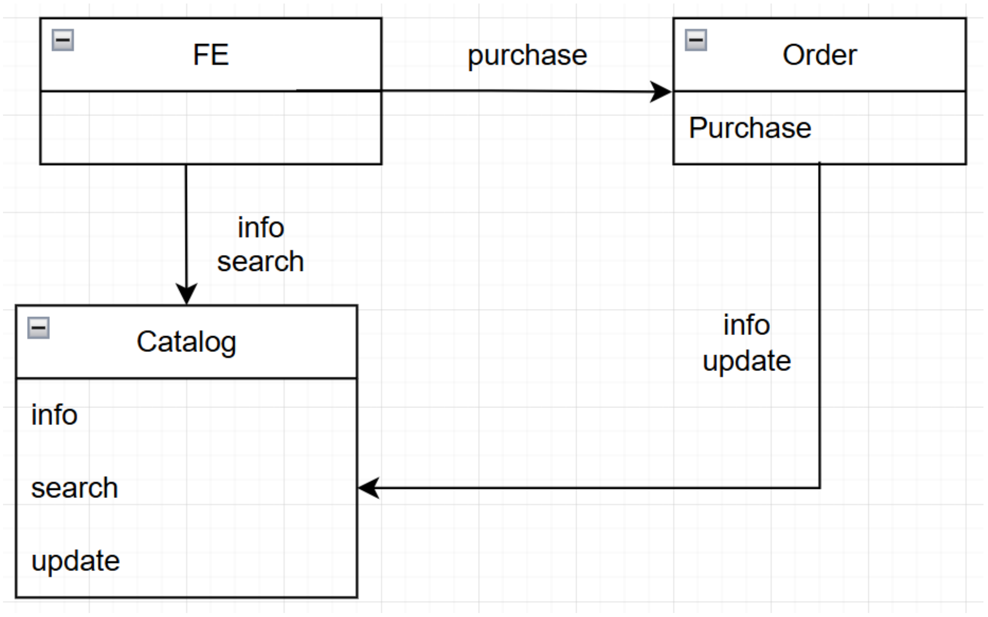
- Migrate command



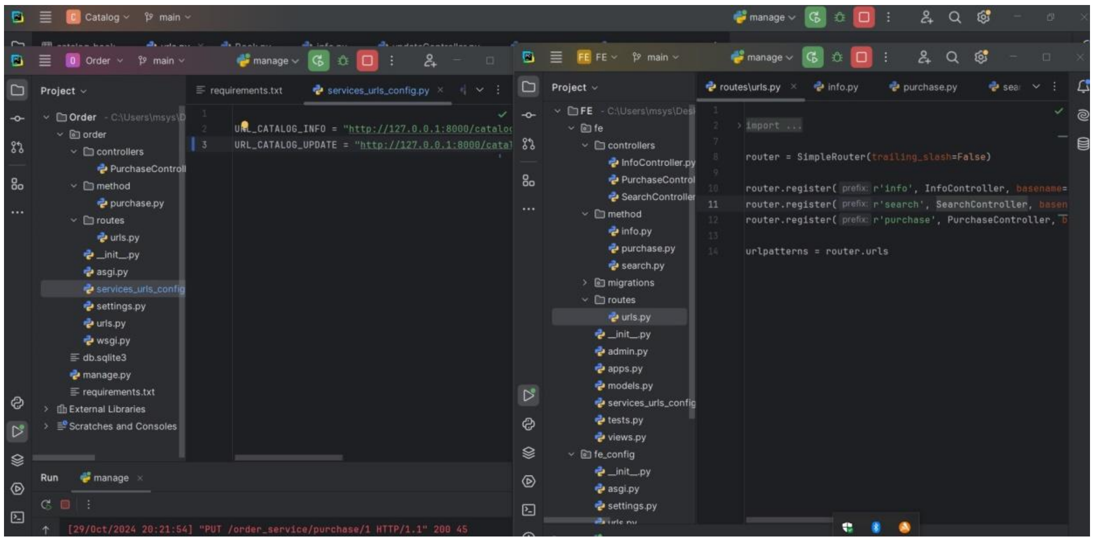
```
PS C:\Users\msys\Desktop\DosProjectV1\Order>
PS C:\Users\msys\Desktop\DosProjectV1\Order> docker-compose exec order_service python manage.py migrate
time="2024-10-29T22:13:01+02:00" level=warning msg="C:\\Users\\msys\\Desktop\\DosProjectV1\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
PS C:\Users\msys\Desktop\DosProjectV1\Order> 
```

The migrate command is very important in a Django project. It’s essential for setting up and maintaining the database schema according to your Django models. Here's what the migrate command provides.

Services



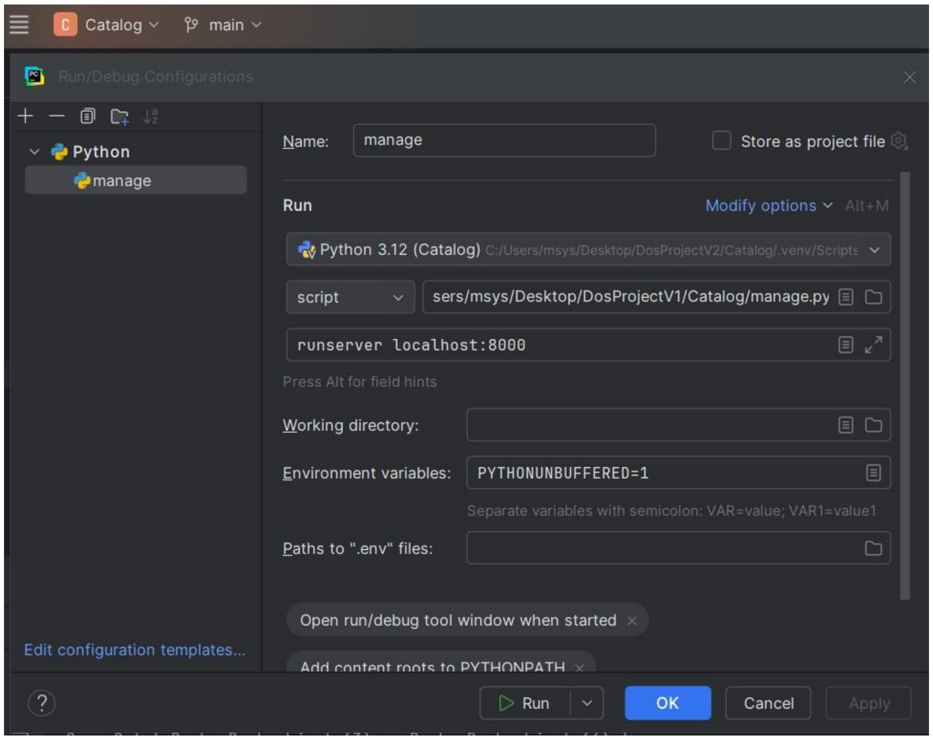
We have three services, each one run on server with different port.



```
URL_CATALOG_INFO = "http://127.0.0.1:8000/catalo
URL_CATALOG_UPDATE = "http://127.0.0.1:8000/catal
```

```
[29/Oct/2024 20:21:54] "PUT /order_service/purchase/1 HTTP/1.1" 200 45
```

```
import ...

router = SimpleRouter(trailing_slash=False)

router.register( prefix: r'info', InfoController, basename=
router.register( prefix: r'search', SearchController, basen
router.register( prefix: r'purchase', PurchaseController, b

urlpatterns = router.urls
```



Run/Debug Configurations

Name: manage

Store as project file

Run

Python 3.12 (Catalog) C:/Users/msys/Desktop/DosProjectV2/Catalog/.venv/Scripts

script sers/msys/Desktop/DosProjectV1/Catalog/manage.py

```
runserver localhost:8000
```

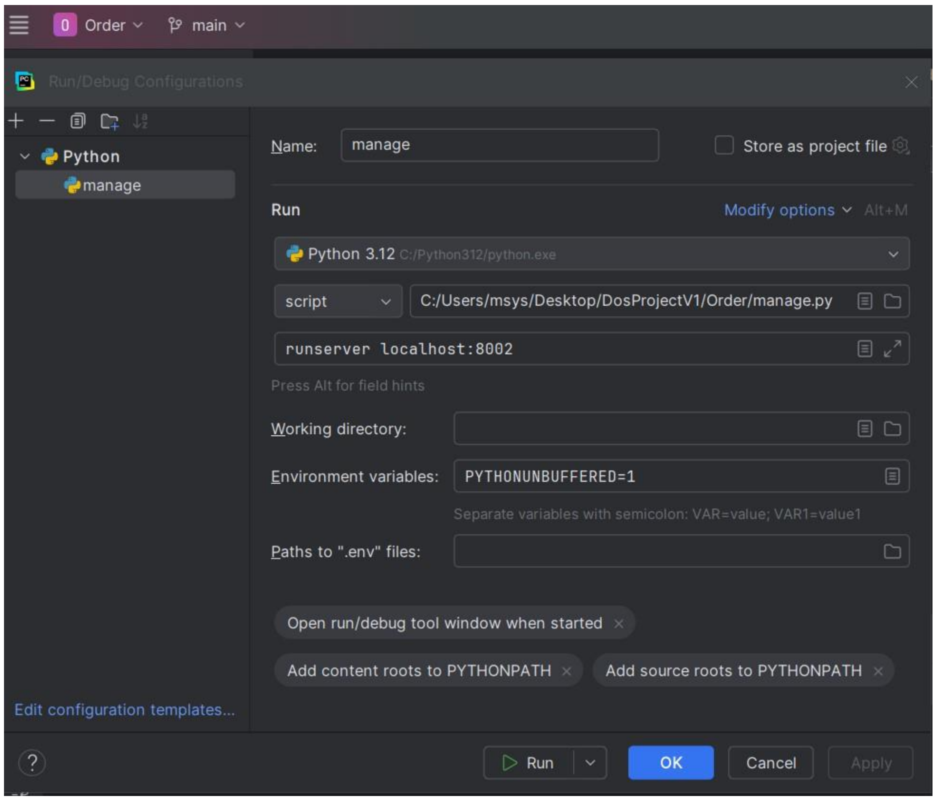
Working directory:

Environment variables: PYTHONUNBUFFERED=1

Paths to ".env" files:

Open run/debug tool window when started

Add content roots to PYTHONPATH



Name: manage

Store as project file

Run

Python 3.12 C:/Python312/python.exe

script C:/Users/msys/Desktop/DosProjectV1/Order/manage.py

```
runserver localhost:8002
```

Press Alt for field hints

Working directory:

Environment variables: PYTHONUNBUFFERED=1

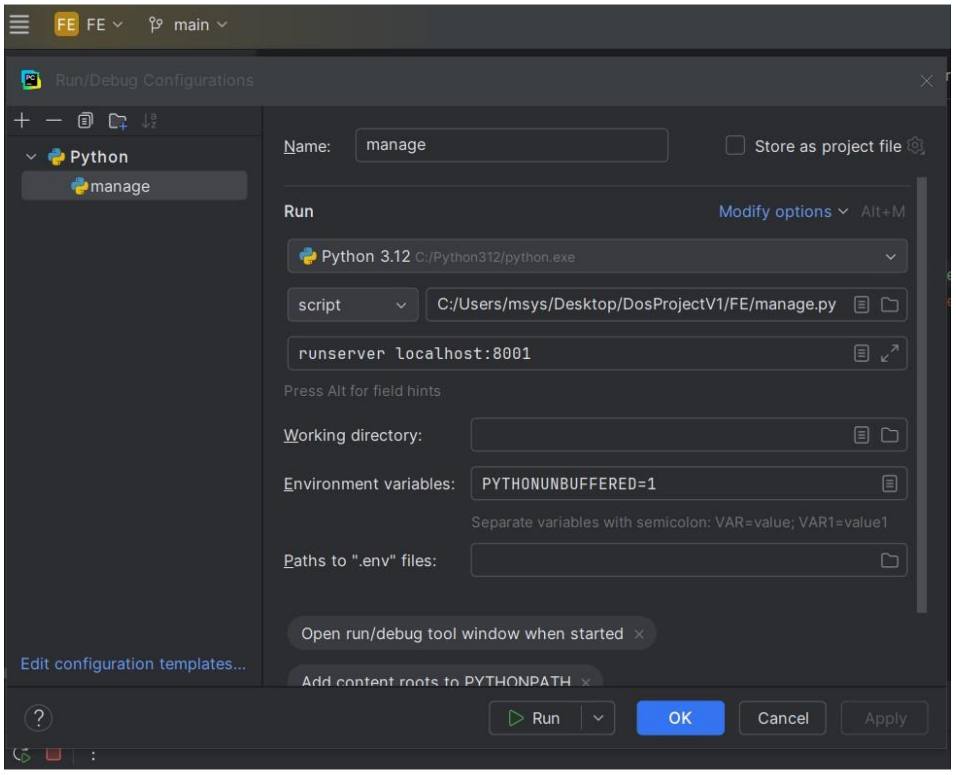
Separate variables with semicolon: VAR=value; VAR1=value1

Paths to ".env" files:

Open run/debug tool window when started

Add content roots to PYTHONPATH

Add source roots to PYTHONPATH



Name: manage

Store as project file

Run

Python 3.12 C:/Python312/python.exe

script C:/Users/msys/Desktop/DosProjectV1/FE/manage.py

```
runserver localhost:8001
```

Press Alt for field hints

Working directory:

Environment variables: PYTHONUNBUFFERED=1

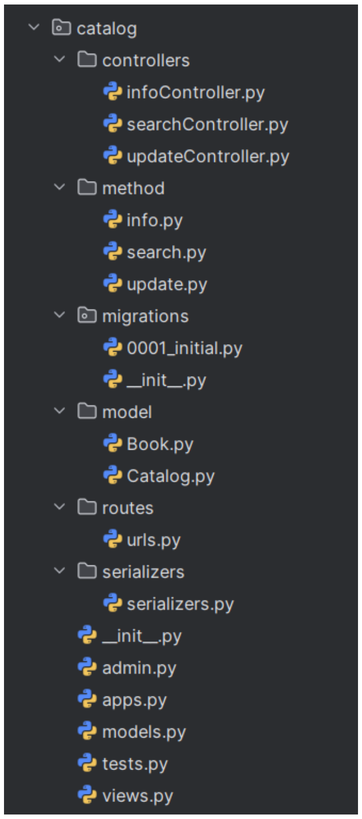
Separate variables with semicolon: VAR=value; VAR1=value1

Paths to ".env" files:

Open run/debug tool window when started

Add content roots to PYTHONPATH

Catalog service

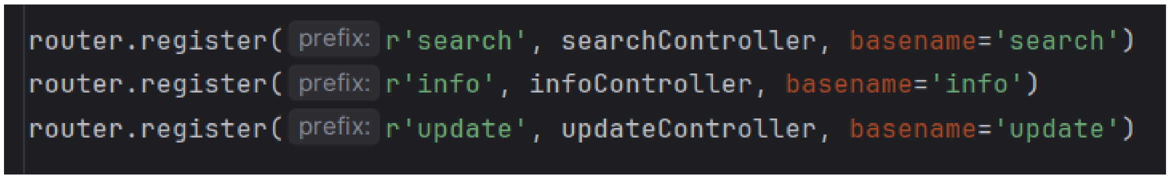


```
catalog
  controllers
    infoController.py
    searchController.py
    updateController.py
  method
    info.py
    search.py
    update.py
  migrations
    0001_initial.py
    __init__.py
  model
    Book.py
    Catalog.py
  routes
    urls.py
  serializers
    serializers.py
  __init__.py
  admin.py
  apps.py
  models.py
  tests.py
  views.py
```

Routes

- **urls.py**

Register search, info, update routes of APIs that already defined in controller files.

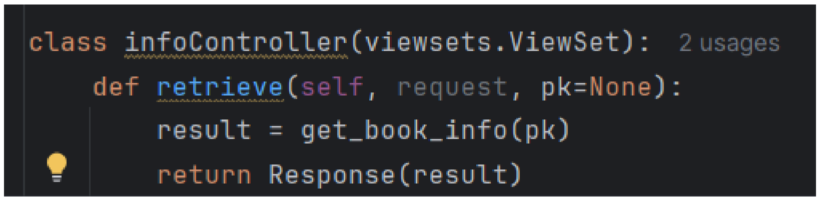


```
router.register( prefix: r'search', searchController, basename='search')
router.register( prefix: r'info', infoController, basename='info')
router.register( prefix: r'update', updateController, basename='update')
```

Controllers

- **InfoController.py**

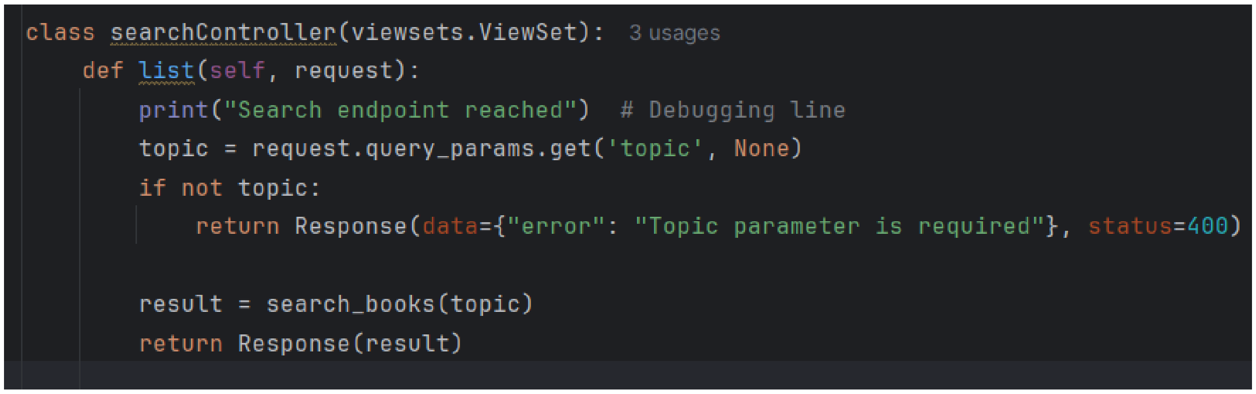
Retrieve method handles requests to fetch information about a specific book, it takes id of book then it calls the get_book_info(pk) function to get the book's details and then returns the result in the HTTP response.



```python
class infoController(viewsets.ViewSet):  2 usages
    def retrieve(self, request, pk=None):
        result = get_book_info(pk)
        return Response(result)
```

- **searchController.py**

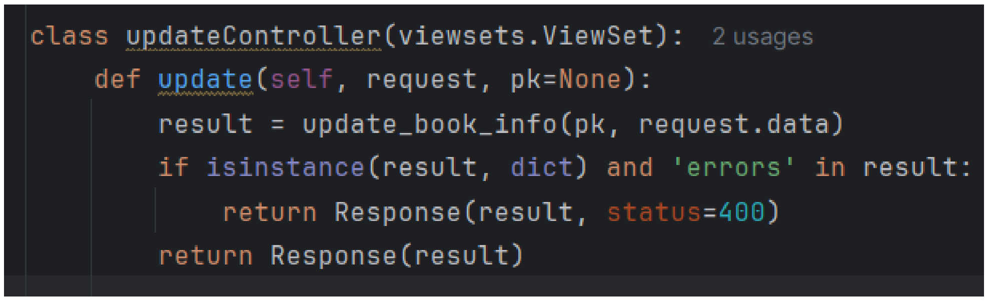
List method handles search requests for books based on a topic query parameter, if no topic, it returns an error response, if the parameter is present, it calls search_books(topic) to get results and returns the response with the search results.



```python
class searchController(viewsets.ViewSet):  3 usages
    def list(self, request):
        print("Search endpoint reached")  # Debugging line
        topic = request.query_params.get('topic', None)
        if not topic:
            return Response(data={"error": "Topic parameter is required"}, status=400)

        result = search_books(topic)
        return Response(result)
```

- **<u>UpdateController.py</u>**

update method to handles requests for updating book information then it calls update_book_info (pk, request.data) with the book id and the data from the request.



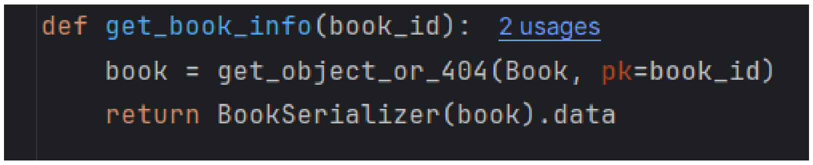
```python
class updateController(viewsets.ViewSet):  2 usages
    def update(self, request, pk=None):
        result = update_book_info(pk, request.data)
        if isinstance(result, dict) and 'errors' in result:
            return Response(result, status=400)
        return Response(result)
```

Update in Django = PUT it takes pk or id, update based on pk unlike create = to POST doesn’t take pk

<u>Methods</u>

- **Info.py**

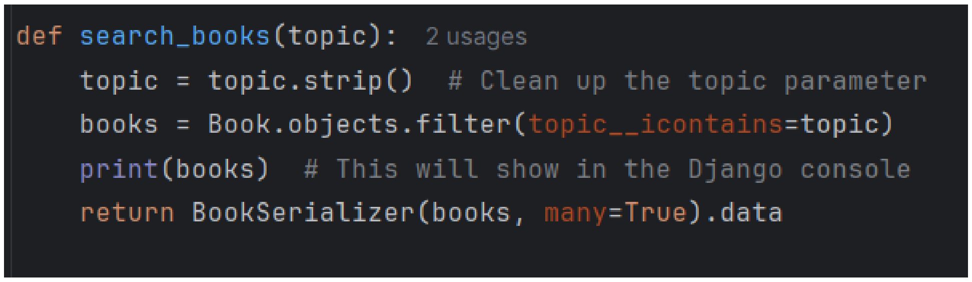
It has get_book_info function retrieves detailed information about a specific book by its id.



```
def get_book_info(book_id):  2 usages
    book = get_object_or_404(Book, pk=book_id)
    return BookSerializer(book).data
```

- **Search.py**

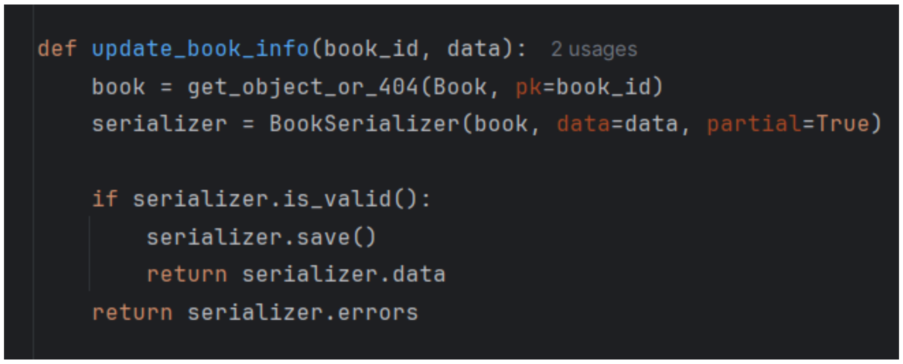
It has a search_books that filters the Book model where the topic field contains the given topic, it serializes the filtered books using BookSerializer and returns the data.



```
def search_books(topic):  2 usages
    topic = topic.strip()  # Clean up the topic parameter
    books = Book.objects.filter(topic__icontains=topic)
    print(books)  # This will show in the Django console
    return BookSerializer(books, many=True).data
```

- **Update.py**

It has update_book_info that updates a specific book's information based on the provided id and data.



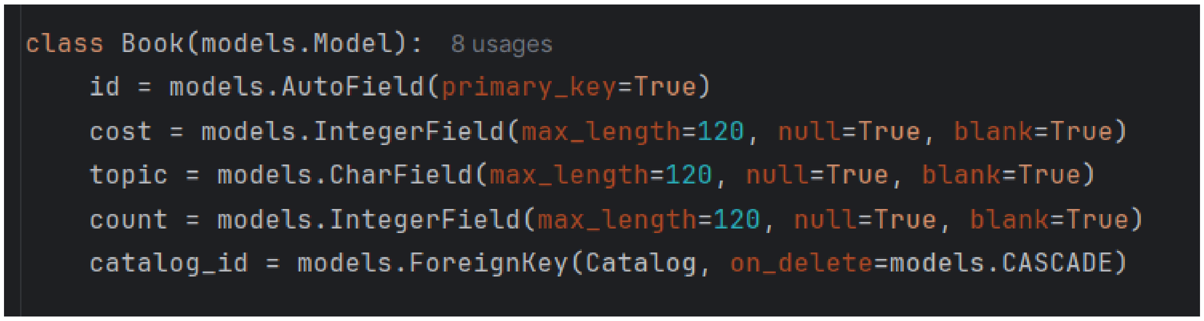
```
def update_book_info(book_id, data):  2 usages
    book = get_object_or_404(Book, pk=book_id)
    serializer = BookSerializer(book, data=data, partial=True)

    if serializer.is_valid():
        serializer.save()
        return serializer.data
    return serializer.errors
```

Models

Models defines the database structure. Each model is a table, with fields as columns. It determines how data is stored.

Book.py

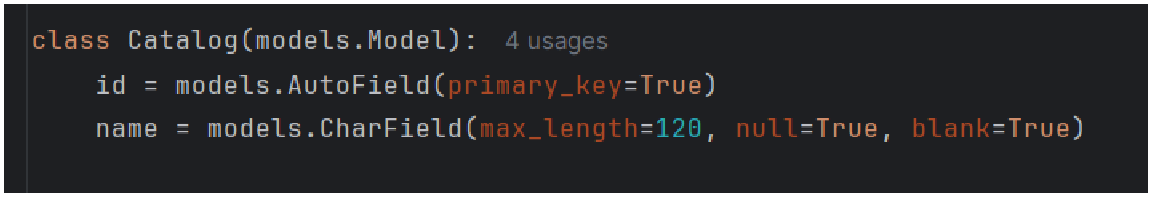
Includes fields of book model.



```
class Book(models.Model):   8 usages
    id = models.AutoField(primary_key=True)
    cost = models.IntegerField(max_length=120, null=True, blank=True)
    topic = models.CharField(max_length=120, null=True, blank=True)
    count = models.IntegerField(max_length=120, null=True, blank=True)
    catalog_id = models.ForeignKey(Catalog, on_delete=models.CASCADE)
```

- ### Catalog.py

Includes fields of catalog model.

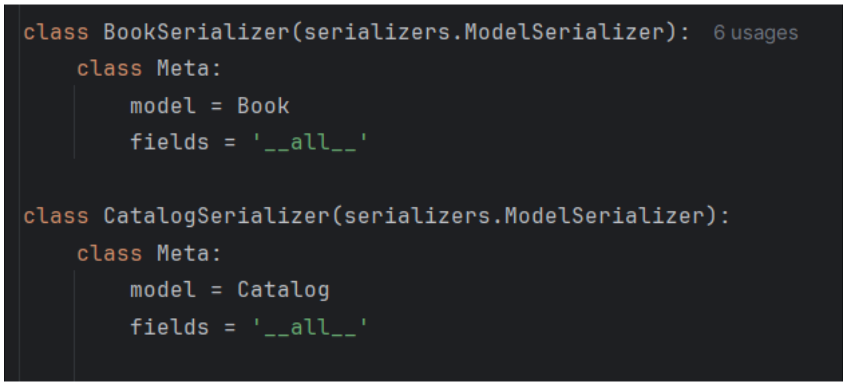


```
class Catalog(models.Model):   4 usages
    id = models.AutoField(primary_key=True)
    name = models.CharField(max_length=120, null=True, blank=True)
```

Serializers:

Serializers Converts data to and from JSON format. It prepares data for front-end use or validates incoming data before saving.

- ### Serializers.py



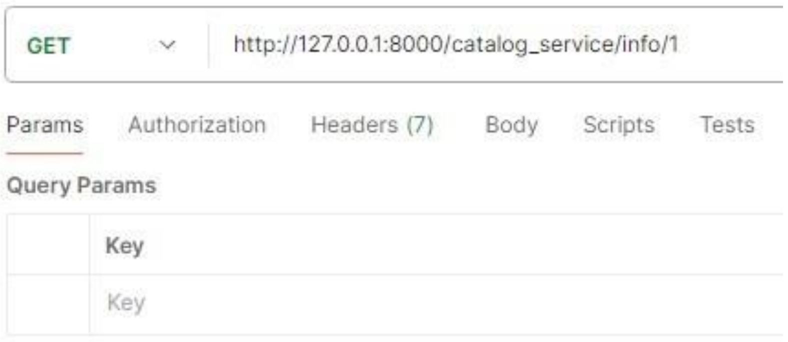
```
class BookSerializer(serializers.ModelSerializer):   6 usages
    class Meta:
        model = Book
        fields = '__all__'

class CatalogSerializer(serializers.ModelSerializer):
    class Meta:
        model = Catalog
        fields = '__all__'
```

Running migrate applies the schema defined in models to the actual database, creating tables and columns. Migrations allow changes in the model to be reflected in the database without manual adjustments.

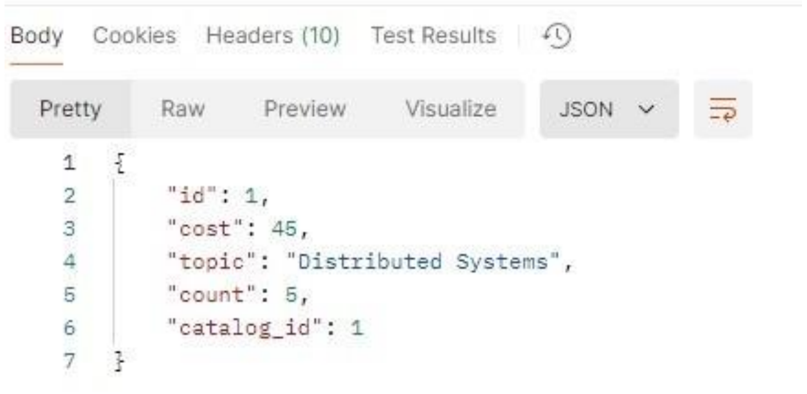
Testing

- **Info**

Case(1): requesting information about available item.



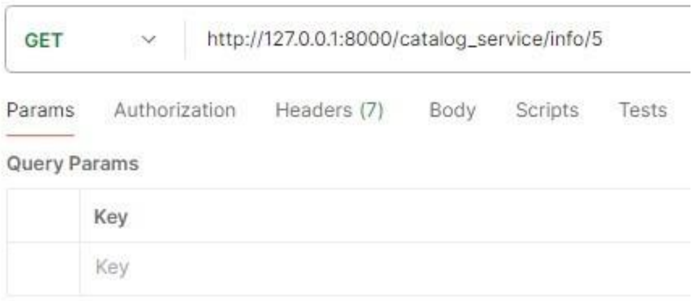
```
GET http://127.0.0.1:8000/catalog_service/info/1
```

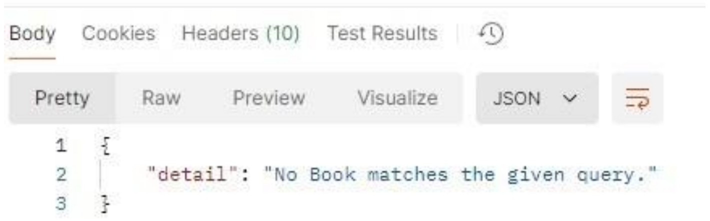


```
{
    "id": 1,
    "cost": 45,
    "topic": "Distributed Systems",
    "count": 5,
    "catalog_id": 1
}
```

-

Case (2): requesting information about unavailable item.

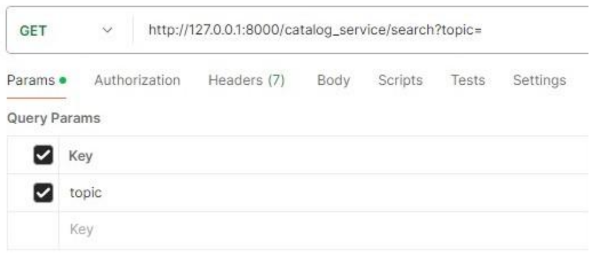

GET http://127.0.0.1:8000/catalog_service/info/5

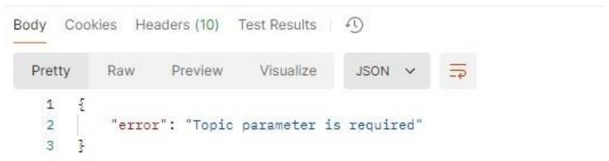


```
{
    "detail": "No Book matches the given query."
}
```

- **Search**

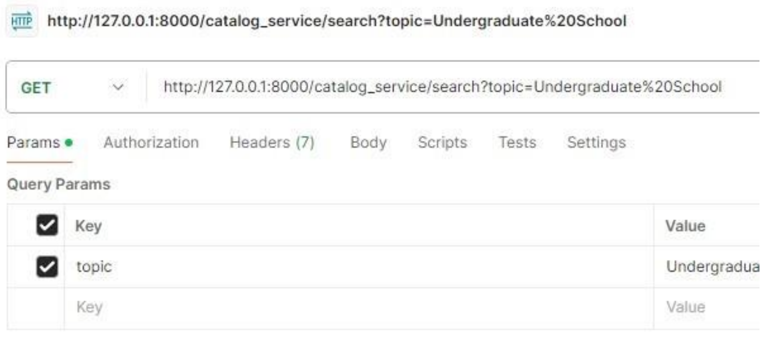
Case (1): no given topic in request


GET http://127.0.0.1:8000/catalog_service/search?topic=



```
{
    "error": "Topic parameter is required"
}
```

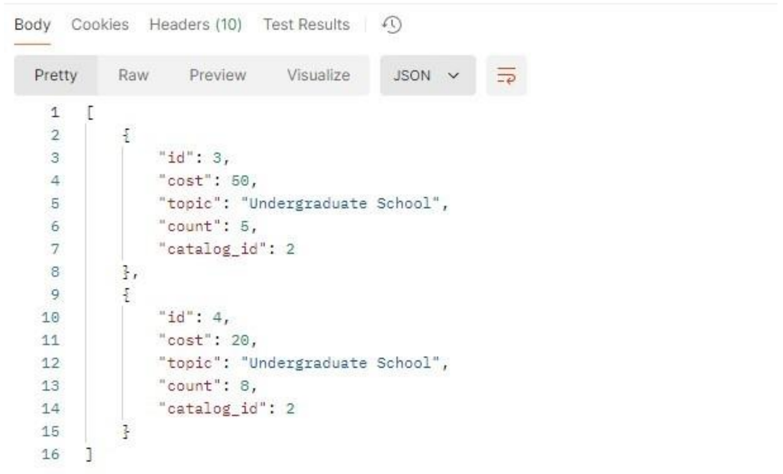
Case (2): topic is in URL



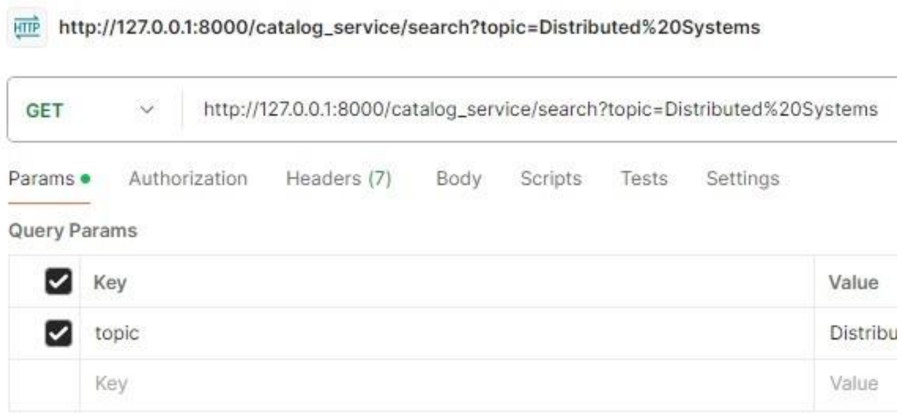
http://127.0.0.1:8000/catalog_service/search?topic=Undergraduate%20School

GET http://127.0.0.1:8000/catalog_service/search?topic=Undergraduate%20School

	Key	Value
✓	topic	Undergradua
	Key	Value

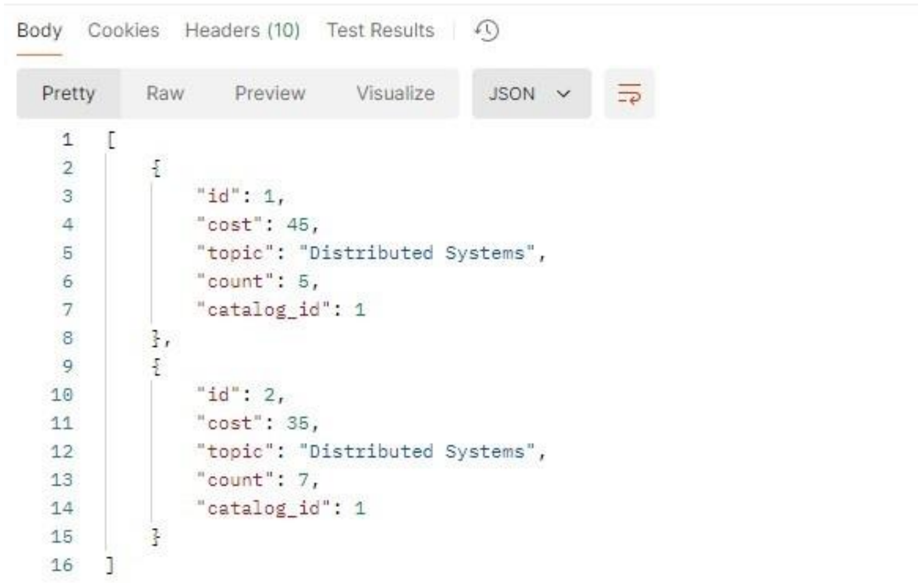


```
[
    {
        "id": 3,
        "cost": 50,
        "topic": "Undergraduate School",
        "count": 5,
        "catalog_id": 2
    },
    {
        "id": 4,
        "cost": 20,
        "topic": "Undergraduate School",
        "count": 8,
        "catalog_id": 2
    }
]
```


http://127.0.0.1:8000/catalog_service/search?topic=Distributed%20Systems

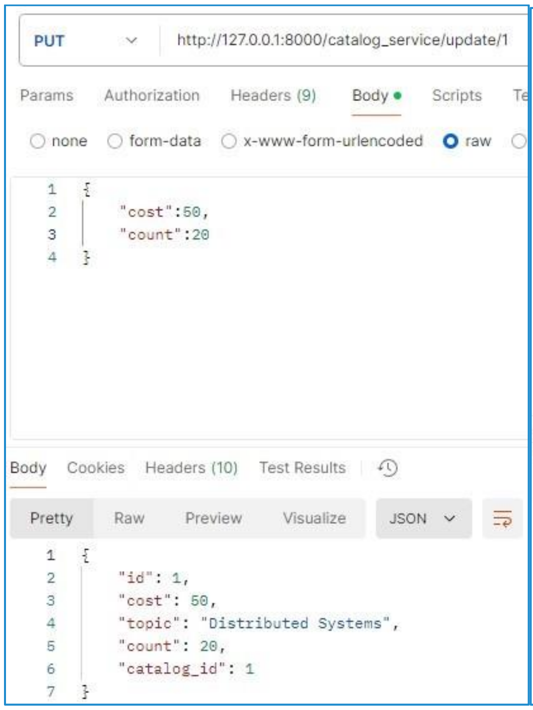
GET http://127.0.0.1:8000/catalog_service/search?topic=Distributed%20Systems

	Key	Value
✓	topic	Distribu
	Key	Value



```
[
    {
        "id": 1,
        "cost": 45,
        "topic": "Distributed Systems",
        "count": 5,
        "catalog_id": 1
    },
    {
        "id": 2,
        "cost": 35,
        "topic": "Distributed Systems",
        "count": 7,
        "catalog_id": 1
    }
]
```

- **Update**

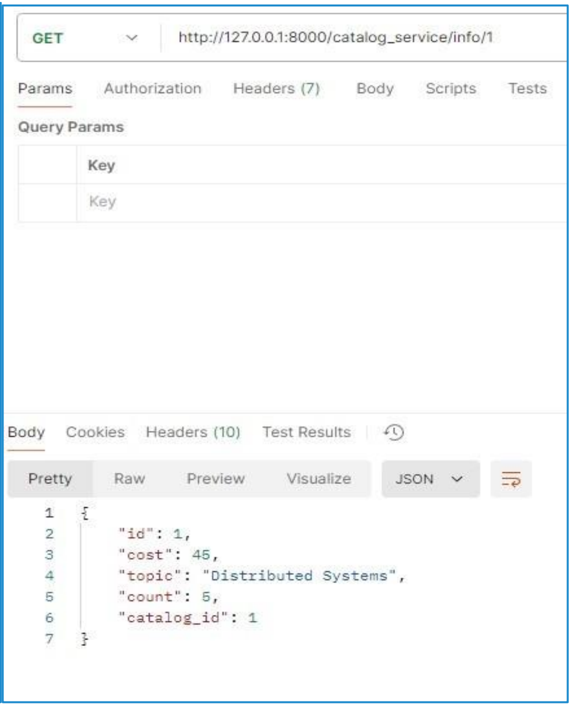

PUT http://127.0.0.1:8000/catalog_service/update/1

Body (raw):

```
{
    "cost":50,
    "count":20
}
```

Response:

```
{
    "id": 1,
    "cost": 50,
    "topic": "Distributed Systems",
    "count": 20,
    "catalog_id": 1
}
```

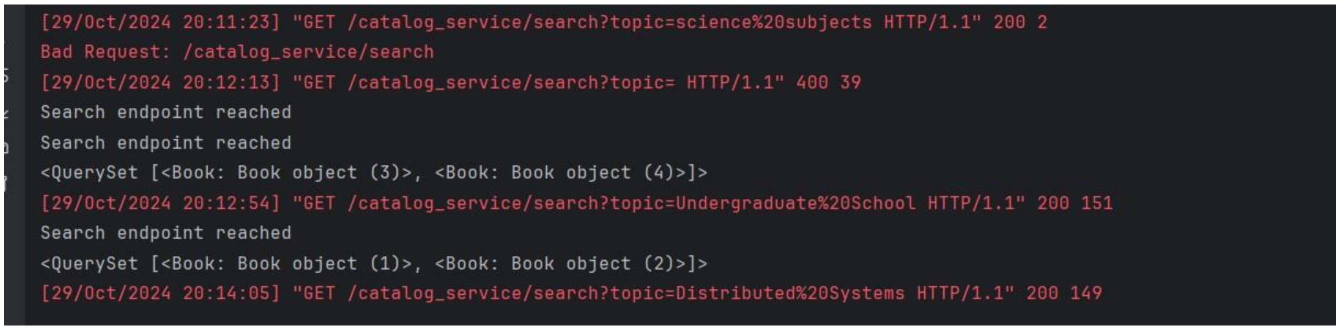

GET http://127.0.0.1:8000/catalog_service/info/1

Response:

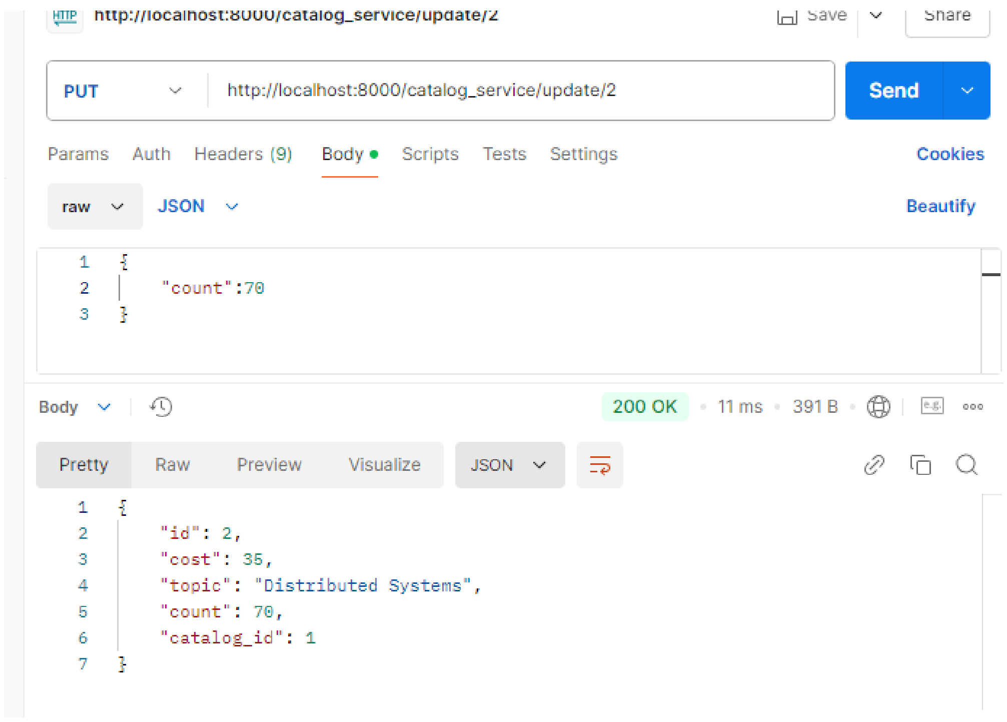
```
{
    "id": 1,
    "cost": 45,
    "topic": "Distributed Systems",
    "count": 5,
    "catalog_id": 1
}
```

after update before update

req & res on terminal


```
[29/Oct/2024 20:11:23] "GET /catalog_service/search?topic=science%20subjects HTTP/1.1" 200 2
Bad Request: /catalog_service/search
[29/Oct/2024 20:12:13] "GET /catalog_service/search?topic= HTTP/1.1" 400 39
Search endpoint reached
Search endpoint reached
<QuerySet [<Book: Book object (3)>, <Book: Book object (4)>]>
[29/Oct/2024 20:12:54] "GET /catalog_service/search?topic=Undergraduate%20School HTTP/1.1" 200 151
Search endpoint reached
<QuerySet [<Book: Book object (1)>, <Book: Book object (2)>]>
[29/Oct/2024 20:14:05] "GET /catalog_service/search?topic=Distributed%20Systems HTTP/1.1" 200 149
```

DB before this req:



http://localhost:8000/catalog_service/update/2

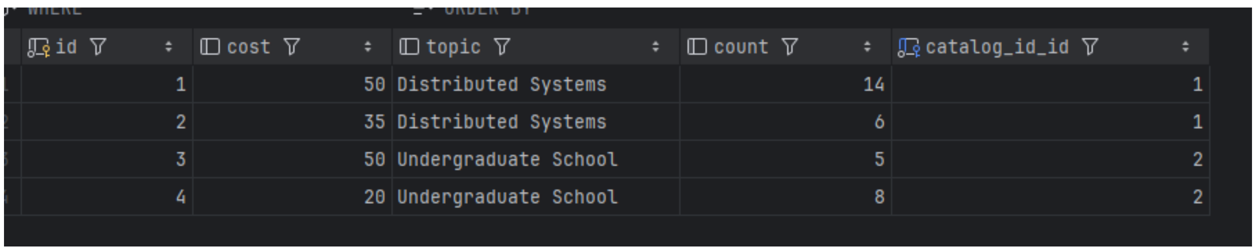
PUT http://localhost:8000/catalog_service/update/2

```
{
    "count":70
}
```

200 OK · 11 ms · 391 B

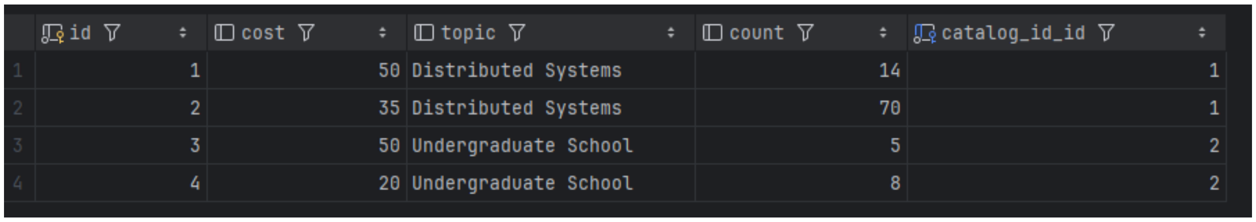
```
{
    "id": 2,
    "cost": 35,
    "topic": "Distributed Systems",
    "count": 70,
    "catalog_id": 1
}
```

“cost” for book id 2 is 35



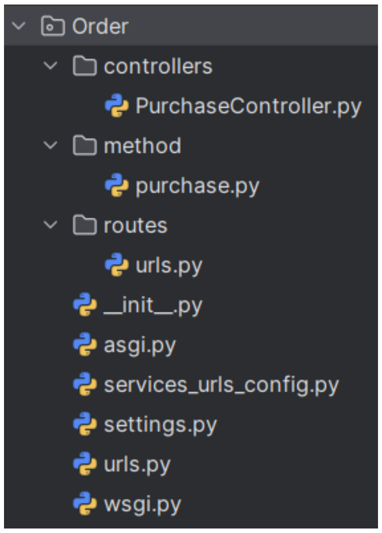
id	cost	topic	count	catalog_id_id
1	50	Distributed Systems	14	1
2	35	Distributed Systems	6	1
3	50	Undergraduate School	5	2
4	20	Undergraduate School	8	2

DB after this req “cost”:70



id	cost	topic	count	catalog_id_id
1	50	Distributed Systems	14	1
2	35	Distributed Systems	70	1
3	50	Undergraduate School	5	2
4	20	Undergraduate School	8	2

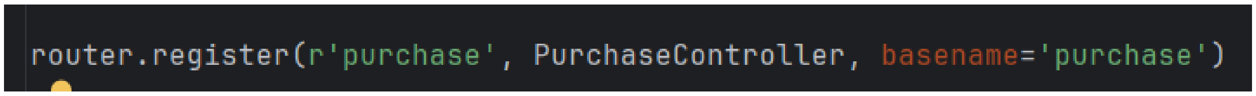
Order service


```
Order
  controllers
    PurchaseController.py
  method
    purchase.py
  routes
    urls.py
  __init__.py
  asgi.py
  services_urls_config.py
  settings.py
  urls.py
  wsgi.py
```

Routes:

- **Urls.py**

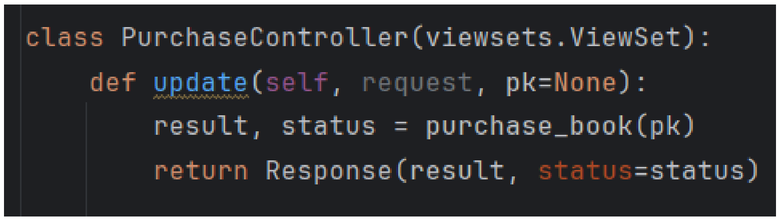
Register purchase API.


```python
router.register(r'purchase', PurchaseController, basename='purchase')
```

Controllers

- **PurchaseController.py**

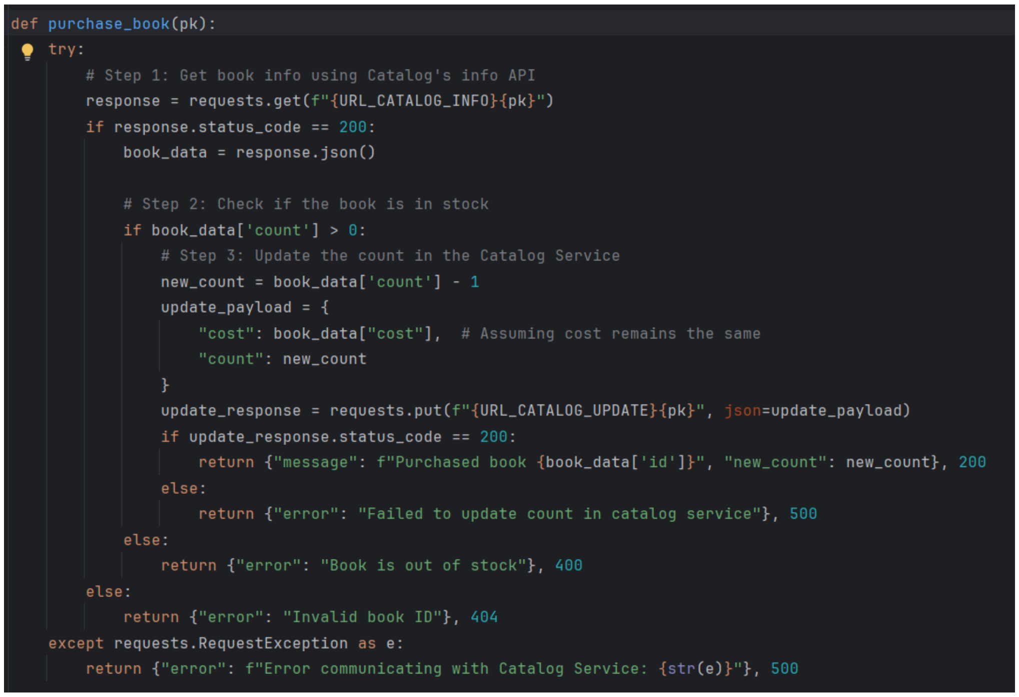
It has update method to handles book purchase requests. it calls purchase_book(pk), passing the book id. The function returns a result and status.


```python
class PurchaseController(viewsets.ViewSet):
    def update(self, request, pk=None):
        result, status = purchase_book(pk)
        return Response(result, status=status)
```

Methods

- **Purchase.py**

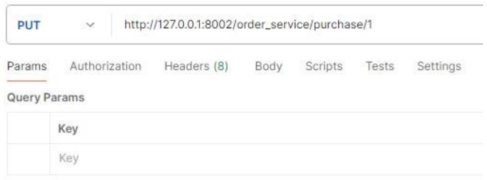
It first checks if the book is in stock; if available, it reduces the count by one and sends an update request to the catalog. On success, it returns a confirmation message with the updated stock. If the book is out of stock, the id is invalid, or there's an issue with the catalog service, it responds with an appropriate error message and status code, ensuring clear feedback for each scenario.



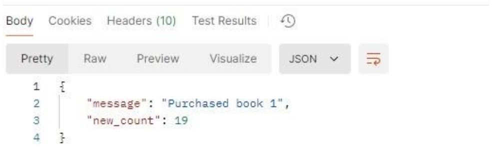
```python
def purchase_book(pk):
    try:
        # Step 1: Get book info using Catalog's info API
        response = requests.get(f"{URL_CATALOG_INFO}{pk}")
        if response.status_code == 200:
            book_data = response.json()

            # Step 2: Check if the book is in stock
            if book_data['count'] > 0:
                # Step 3: Update the count in the Catalog Service
                new_count = book_data['count'] - 1
                update_payload = {
                    "cost": book_data["cost"],  # Assuming cost remains the same
                    "count": new_count
                }
                update_response = requests.put(f"{URL_CATALOG_UPDATE}{pk}", json=update_payload)
                if update_response.status_code == 200:
                    return {"message": f"Purchased book {book_data['id']}", "new_count": new_count}, 200
                else:
                    return {"error": "Failed to update count in catalog service"}, 500
            else:
                return {"error": "Book is out of stock"}, 400
        else:
            return {"error": "Invalid book ID"}, 404
    except requests.RequestException as e:
        return {"error": f"Error communicating with Catalog Service: {str(e)}"}, 500
```

Testing:

- **Purchase**

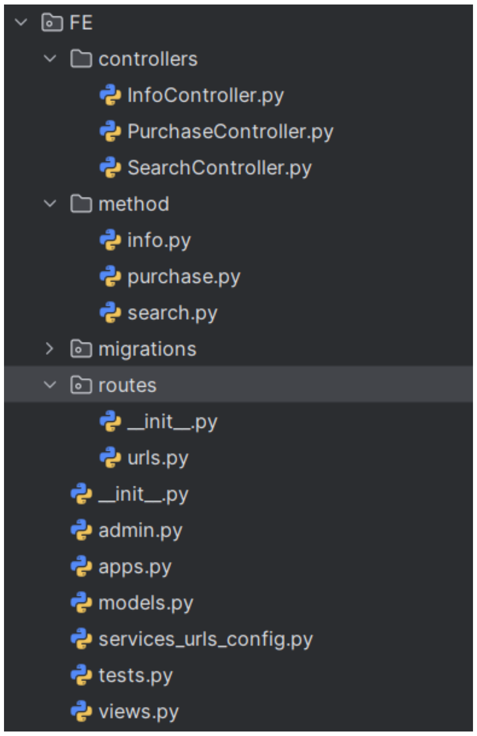


```
PUT  http://127.0.0.1:8002/order_service/purchase/1
```



```json
{
    "message": "Purchased book 1",
    "new_count": 19
}
```

FE Service

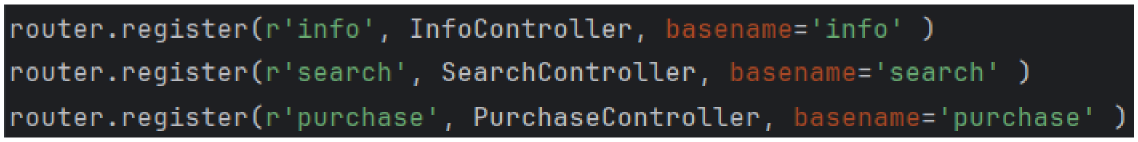


```
FE
  controllers
    InfoController.py
    PurchaseController.py
    SearchController.py
  method
    info.py
    purchase.py
    search.py
  migrations
  routes
    __init__.py
    urls.py
  __init__.py
  admin.py
  apps.py
  models.py
  services_urls_config.py
  tests.py
  views.py
```

Routes

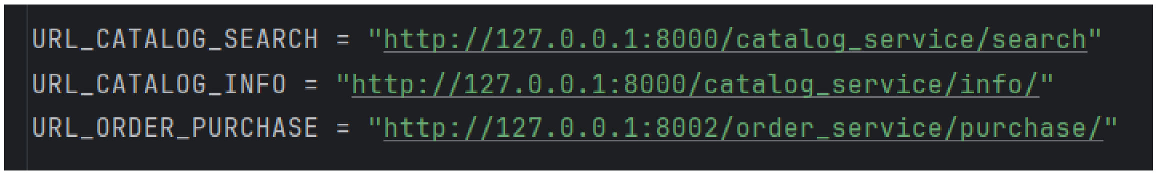
- **Urls.py**

Register info, purchase, search urls.



```
router.register(r'info', InfoController, basename='info' )
router.register(r'search', SearchController, basename='search' )
router.register(r'purchase', PurchaseController, basename='purchase' )
```

- **Service_urls_config.py**

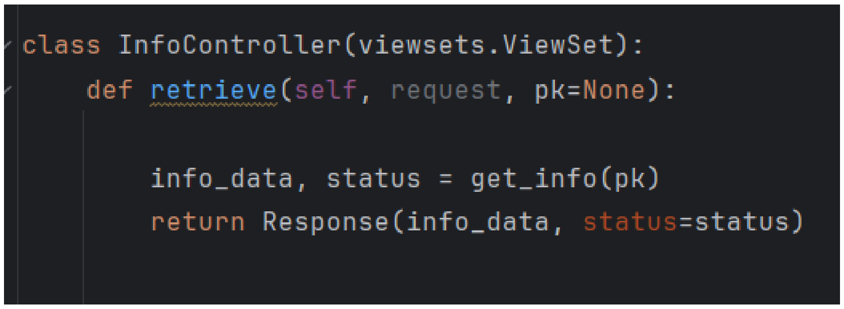


```
URL_CATALOG_SEARCH = "http://127.0.0.1:8000/catalog_service/search"
URL_CATALOG_INFO = "http://127.0.0.1:8000/catalog_service/info/"
URL_ORDER_PURCHASE = "http://127.0.0.1:8002/order_service/purchase/"
```

<u>Controllers</u>

- **InfoController.py**

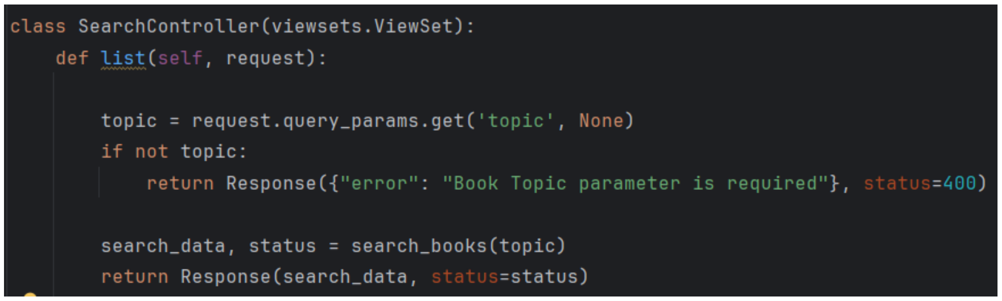
Same logic of retrieve in infoController in catalog.



```python
class InfoController(viewsets.ViewSet):
    def retrieve(self, request, pk=None):

        info_data, status = get_info(pk)
        return Response(info_data, status=status)
```

- **SearchController.py**

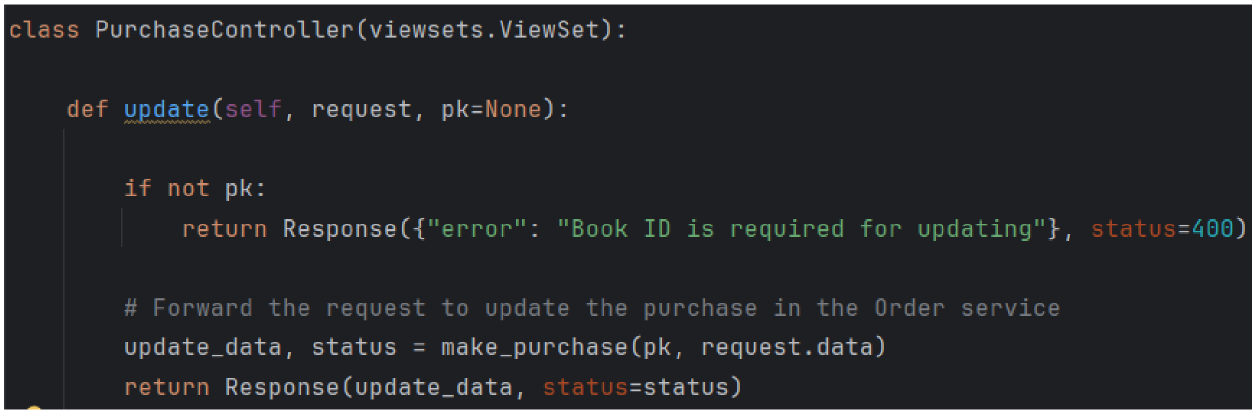
Same logic of search in searchController in catalog.



```python
class SearchController(viewsets.ViewSet):
    def list(self, request):

        topic = request.query_params.get('topic', None)
        if not topic:
            return Response({"error": "Book Topic parameter is required"}, status=400)

        search_data, status = search_books(topic)
        return Response(search_data, status=status)
```

- **PurchaseController.py**

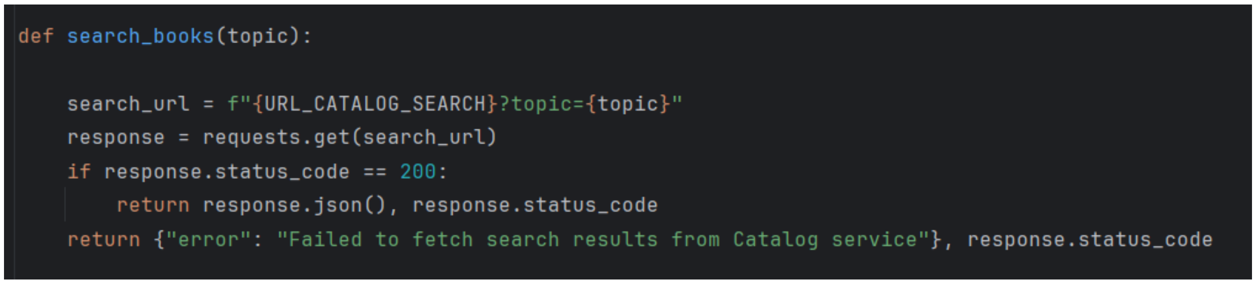
Same logic of purchase in purchaseController in order.



```python
class PurchaseController(viewsets.ViewSet):

    def update(self, request, pk=None):

        if not pk:
            return Response({"error": "Book ID is required for updating"}, status=400)

        # Forward the request to update the purchase in the Order service
        update_data, status = make_purchase(pk, request.data)
        return Response(update_data, status=status)
```

Methods

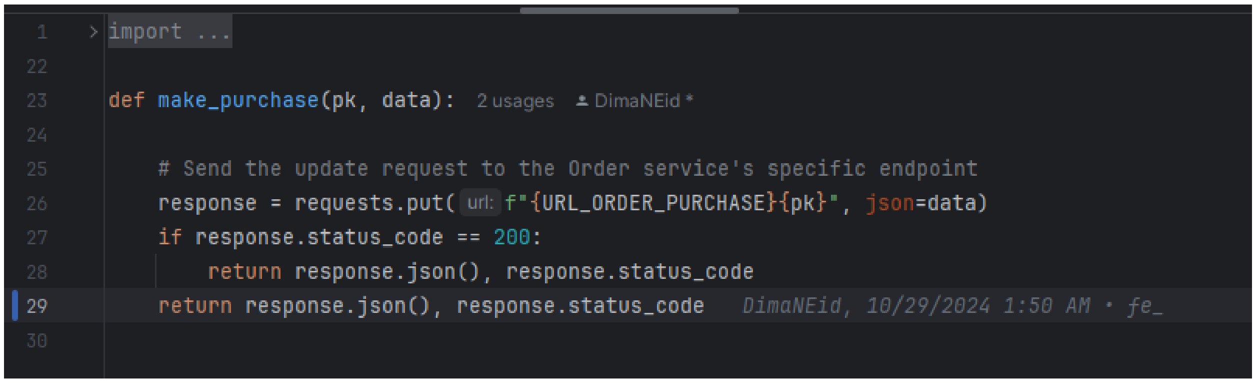
- **Search.py**



```python
def search_books(topic):

    search_url = f"{URL_CATALOG_SEARCH}?topic={topic}"
    response = requests.get(search_url)
    if response.status_code == 200:
        return response.json(), response.status_code
    return {"error": "Failed to fetch search results from Catalog service"}, response.status_code
```

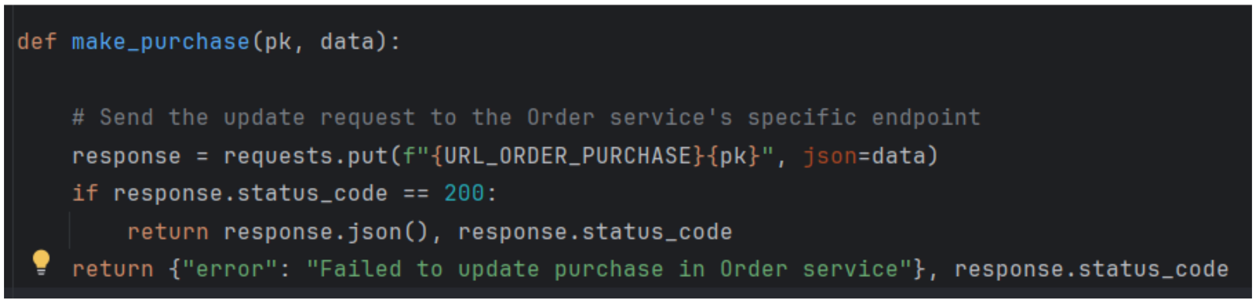
- **Info.py**

  Return response directly from order service



```python
import ...

def make_purchase(pk, data):  2 usages  DimaNEid *

    # Send the update request to the Order service's specific endpoint
    response = requests.put( url: f"{URL_ORDER_PURCHASE}{pk}", json=data)
    if response.status_code == 200:
        return response.json(), response.status_code
    return response.json(), response.status_code   DimaNEid, 10/29/2024 1:50 AM • fe_

```

- **Purchase.py**

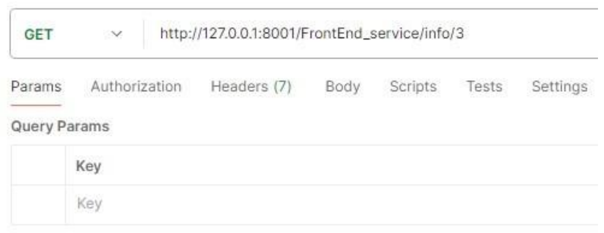


```python
def make_purchase(pk, data):

    # Send the update request to the Order service's specific endpoint
    response = requests.put(f"{URL_ORDER_PURCHASE}{pk}", json=data)
    if response.status_code == 200:
        return response.json(), response.status_code
    return {"error": "Failed to update purchase in Order service"}, response.status_code
```

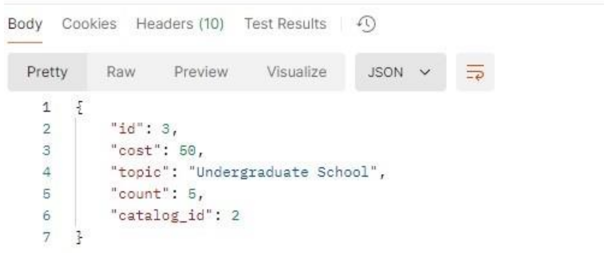
Testing

- **Info**

Getting info from frontend service



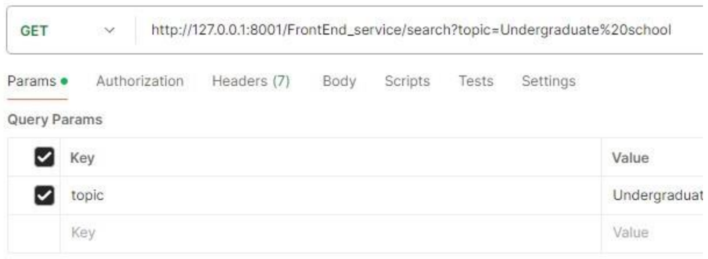
```
http://127.0.0.1:8001/FrontEnd_service/info/3
```



```
{
    "id": 3,
    "cost": 50,
    "topic": "Undergraduate School",
    "count": 5,
    "catalog_id": 2
}
```

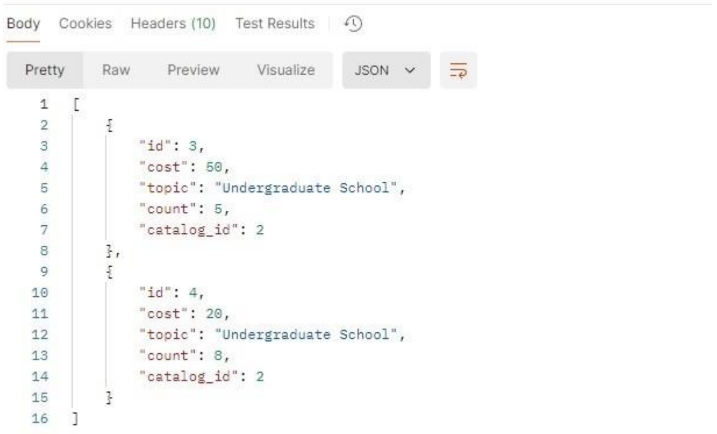
- **Search**

Make a search from frontend



GET http://127.0.0.1:8001/FrontEnd_service/search?topic=Undergraduate%20school

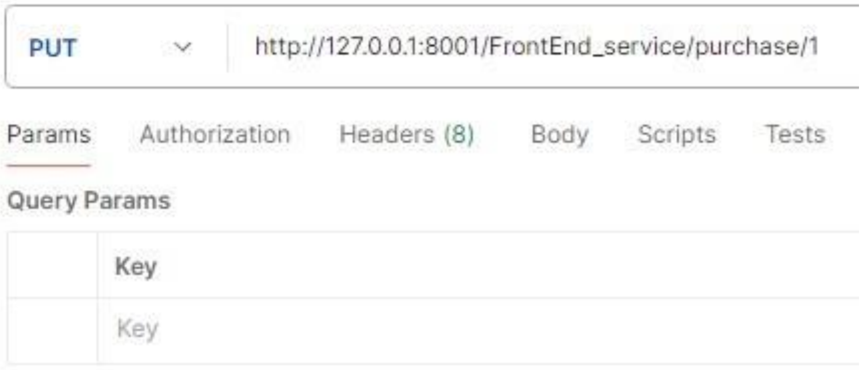
	Key	Value
✓	topic	Undergraduat
	Key	Value



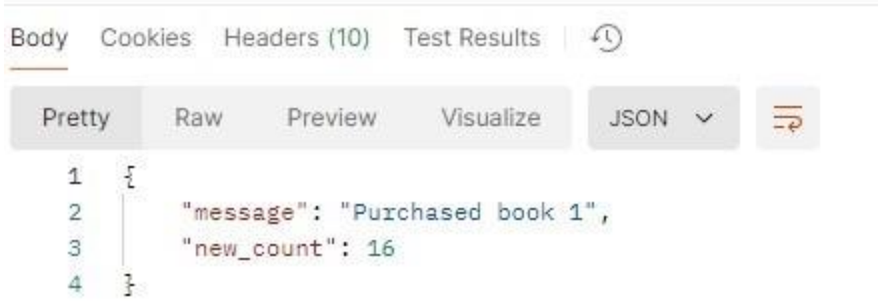
```
[
    {
        "id": 3,
        "cost": 50,
        "topic": "Undergraduate School",
        "count": 5,
        "catalog_id": 2
    },
    {
        "id": 4,
        "cost": 20,
        "topic": "Undergraduate School",
        "count": 8,
        "catalog_id": 2
    }
]
```

- **Purchase**

Make purchase from frontend.



PUT http://127.0.0.1:8001/FrontEnd_service/purchase/1



```
{
    "message": "Purchased book 1",
    "new_count": 16
}
```

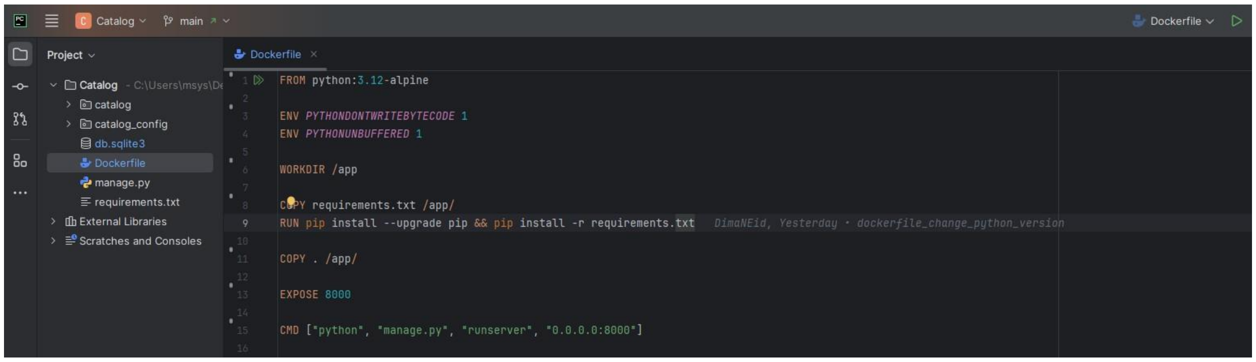
<u>Dockerfile & Compose file</u>

The d**ocker file** is to create a single Docker image with everything needed to run the application but d**ocker compose** is used to orchestrate multiple containers to run together at the same time and it uses a configuration file with .yml extension to manage all services. Docker desktop must be open while doing all these steps.

<u>Docker file</u>

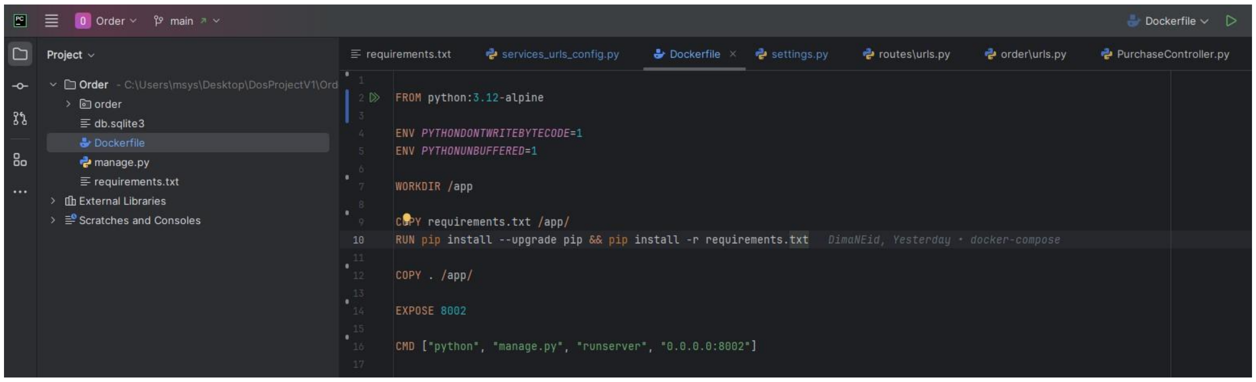
The Docker file creates docker image using the Alpine-based Python image, installs dependencies from requirements file to copy the application code into the container, specific port and make the application can be accessible externally.

- Docker file in catalog service



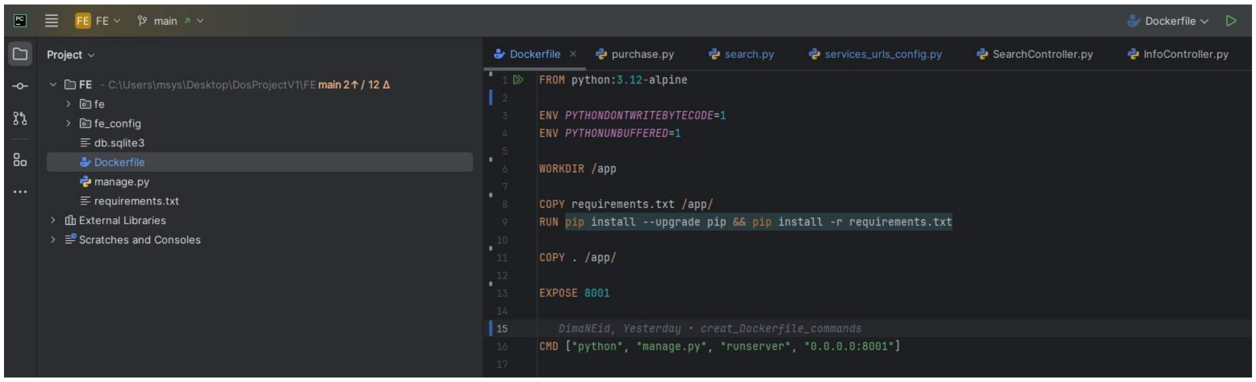
```
FROM python:3.12-alpine

ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

WORKDIR /app

COPY requirements.txt /app/
RUN pip install --upgrade pip && pip install -r requirements.txt   DimaNEid, Yesterday • dockerfile_change_python_version

COPY . /app/

EXPOSE 8000

CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

- Docker file in order service
-



```

FROM python:3.12-alpine

ENV PYTHONDONTWRITEBYTECODE=1
ENV PYTHONUNBUFFERED=1

WORKDIR /app

COPY requirements.txt /app/
RUN pip install --upgrade pip && pip install -r requirements.txt   DimaNEid, Yesterday • docker-compose

COPY . /app/

EXPOSE 8002

CMD ["python", "manage.py", "runserver", "0.0.0.0:8002"]
```

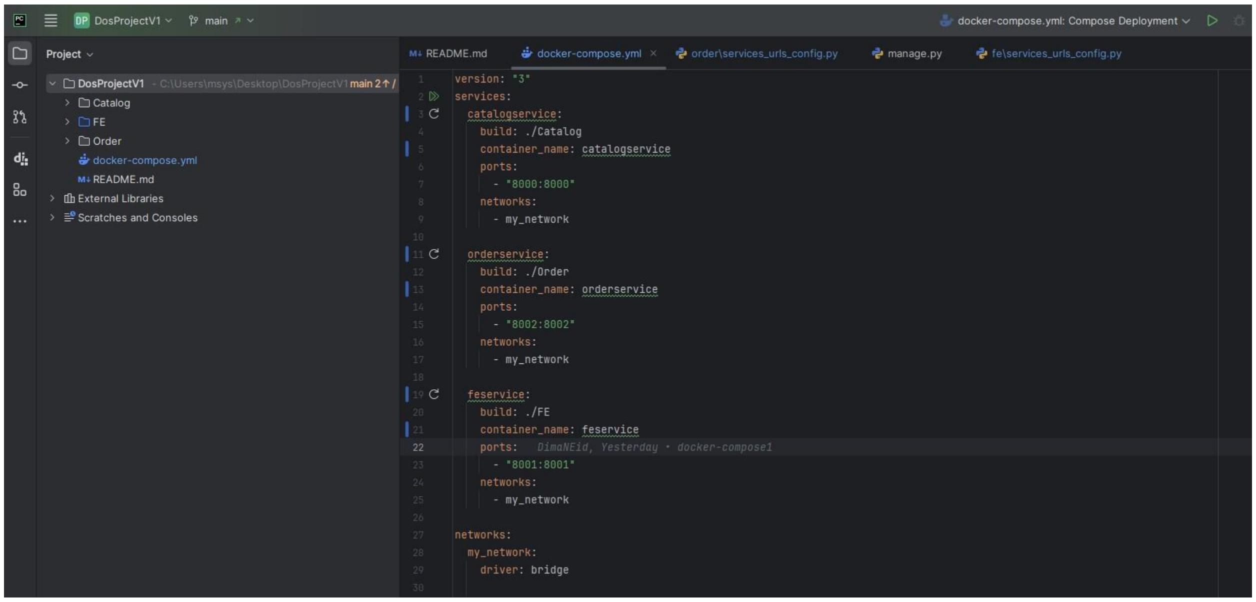
- Docker file in FE service



```
FROM python:3.12-alpine

ENV PYTHONDONTWRITEBYTECODE=1
ENV PYTHONUNBUFFERED=1

WORKDIR /app

COPY requirements.txt /app/
RUN pip install --upgrade pip && pip install -r requirements.txt

COPY . /app/

EXPOSE 8001


CMD ["python", "manage.py", "runserver", "0.0.0.0:8001"]
```

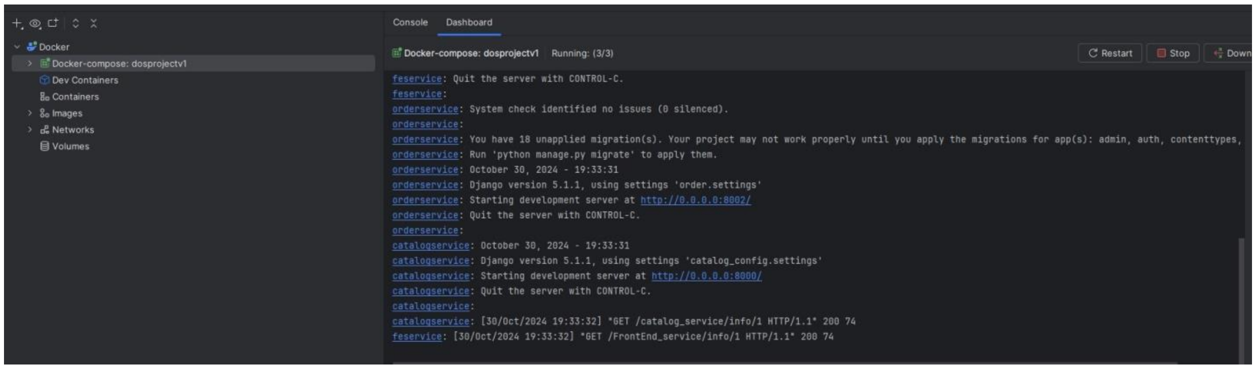
<u>Docker compose file</u>

The docker compose file defines the three services catalogservice, feservice and orderservice, notice that each service must be named like “catalogservice,” and we have an issue when we write it like “catalog_service”, each service is built from its directory with unique container name. The services are exposed on different ports 8000,8001,8002 allowing them to communicate over a custom Docker network called **my_network.**



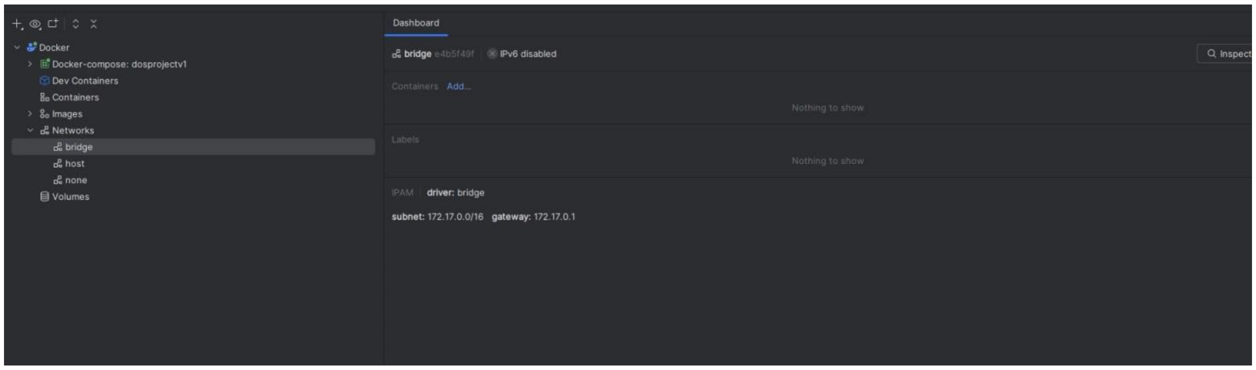
```yaml
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

As we can see, all services containers were successfully run and each service was accessible by its port, confirming proper deployment and functionality within the Docker environment.



```
feservice: Quit the server with CONTROL-C.
feservice:
orderservice: System check identified no issues (0 silenced).
orderservice:
orderservice: You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes,
orderservice: Run 'python manage.py migrate' to apply them.
orderservice: October 30, 2024 - 19:33:31
orderservice: Django version 5.1.1, using settings 'order.settings'
orderservice: Starting development server at http://0.0.0.0:8002/
orderservice: Quit the server with CONTROL-C.
orderservice:
catalogservice: October 30, 2024 - 19:33:31
catalogservice: Django version 5.1.1, using settings 'catalog_config.settings'
catalogservice: Starting development server at http://0.0.0.0:8000/
catalogservice: Quit the server with CONTROL-C.
catalogservice:
catalogservice: [30/Oct/2024 19:33:32] "GET /catalog_service/info/1 HTTP/1.1" 200 74
feservice: [30/Oct/2024 19:33:32] "GET /FrontEnd_service/info/1 HTTP/1.1" 200 74
```

The Docker creates bridge network with a subnet of 172.17.0.0/16 and gateway 172.17.0.1.



bridge e4b5f49f IPv6 disabled

Containers Add...

Nothing to show

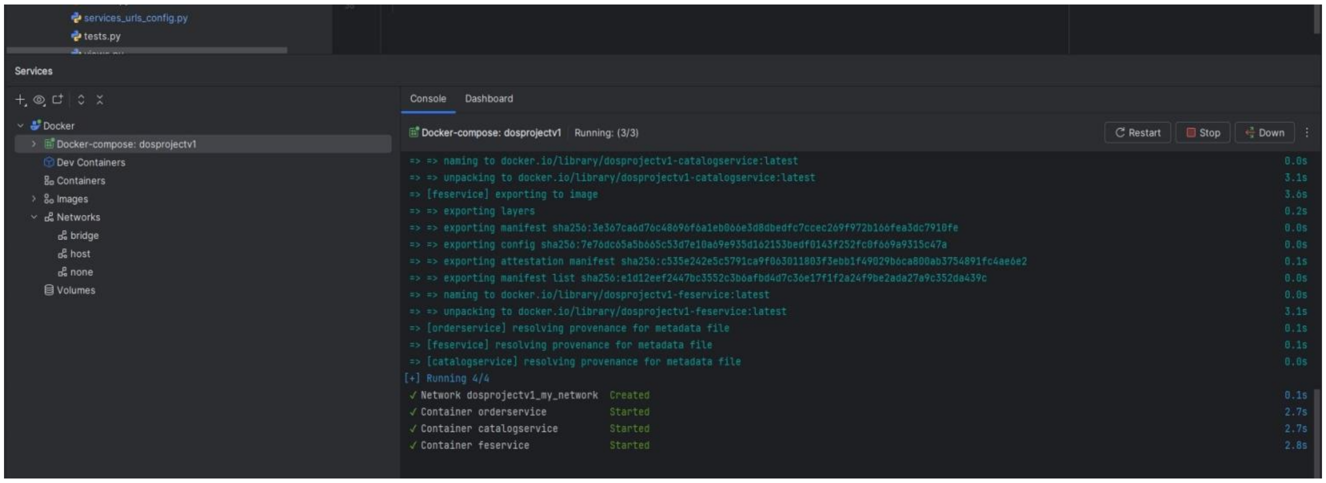
Labels

Nothing to show

IPAM driver: bridge

subnet: 172.17.0.0/16 gateway: 172.17.0.1

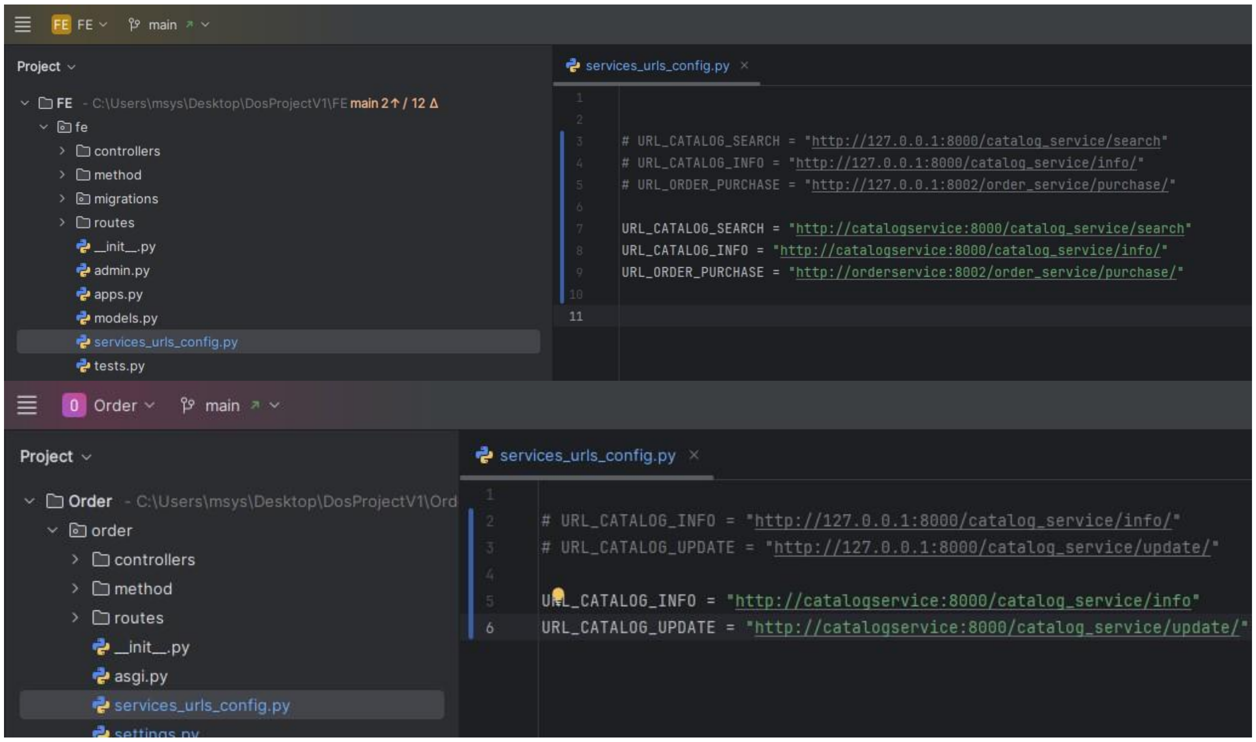
The three containers are successfully running and ready for use.



```
=> => naming to docker.io/library/dosprojectv1-catalogservice:latest                                   0.0s
=> => unpacking to docker.io/library/dosprojectv1-catalogservice:latest                                3.1s
=> [feservice] exporting to image                                                                      3.6s
=> => exporting layers                                                                                 0.2s
=> => exporting manifest sha256:3e367ca6d76c48696f6a1eb066e3d8dbedfc7ccec269f972b166fea3dc7910fe       0.0s
=> => exporting config sha256:7e76dc65a5b665c53d7e10a69e935d162153bedf0143f252fc0f669a9315c47a         0.0s
=> => exporting attestation manifest sha256:c535e242e5c5791ca9f063011803f3ebb1f49029b6ca800ab3754891fc4ae6e2  0.1s
=> => exporting manifest list sha256:e1d12eef2447bc3552c3b6afbd4d7c36e17f1f2a24f9be2ada27a9c352da439c  0.0s
=> => naming to docker.io/library/dosprojectv1-feservice:latest                                        0.0s
=> => unpacking to docker.io/library/dosprojectv1-feservice:latest                                     3.1s
=> [orderservice] resolving provenance for metadata file                                               0.1s
=> [feservice] resolving provenance for metadata file                                                  0.1s
=> [catalogservice] resolving provenance for metadata file                                             0.0s
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created                                                            0.1s
 ✓ Container orderservice           Started                                                            2.7s
 ✓ Container catalogservice         Started                                                            2.7s
 ✓ Container feservice              Started                                                            2.8s
```

- **services_urls_config.py**

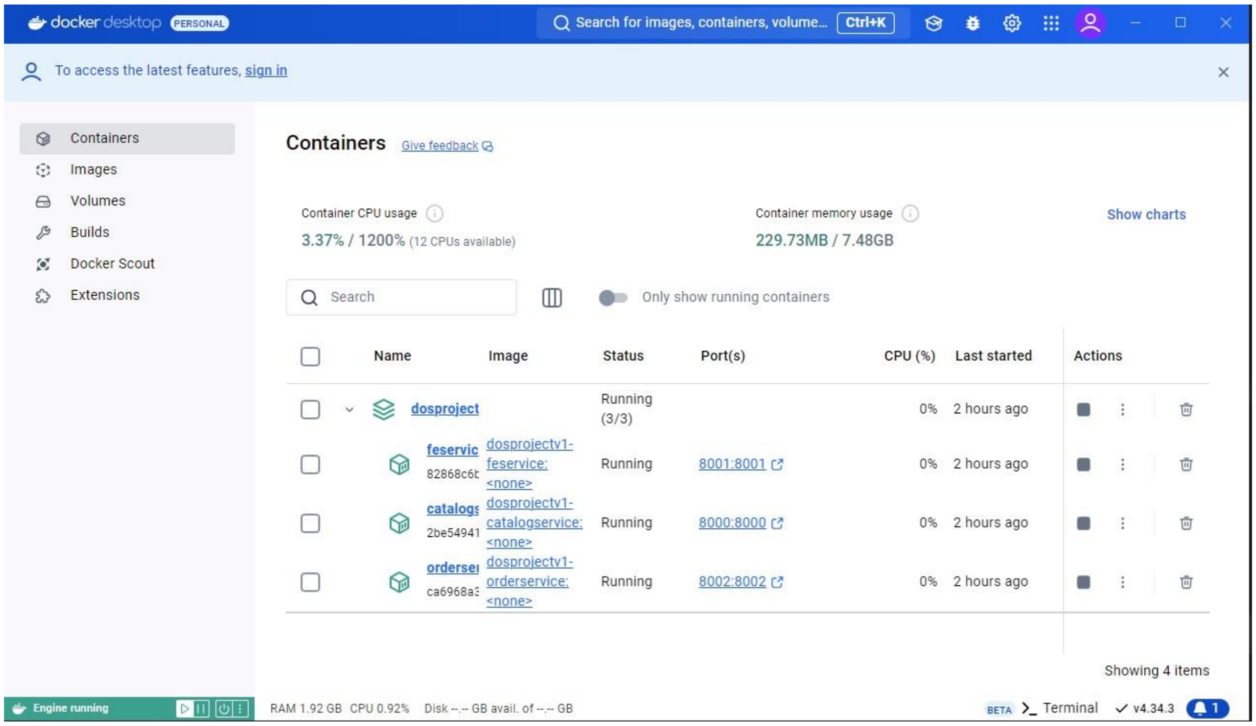
The files in both the FE and Order directories define URLs for accessing various services in the dosprojectv1 setup. The configuration uses container names catalogservice and orderservice instead of localhost.



```
# URL_CATALOG_SEARCH = "http://127.0.0.1:8000/catalog_service/search"
# URL_CATALOG_INFO = "http://127.0.0.1:8000/catalog_service/info/"
# URL_ORDER_PURCHASE = "http://127.0.0.1:8002/order_service/purchase/"

URL_CATALOG_SEARCH = "http://catalogservice:8000/catalog_service/search"
URL_CATALOG_INFO = "http://catalogservice:8000/catalog_service/info/"
URL_ORDER_PURCHASE = "http://orderservice:8002/order_service/purchase/"

```

```

# URL_CATALOG_INFO = "http://127.0.0.1:8000/catalog_service/info/"
# URL_CATALOG_UPDATE = "http://127.0.0.1:8000/catalog_service/update/"

URL_CATALOG_INFO = "http://catalogservice:8000/catalog_service/info"
URL_CATALOG_UPDATE = "http://catalogservice:8000/catalog_service/update/"
```

Containers


docker desktop PERSONAL

To access the latest features, sign in

Containers Give feedback

Container CPU usage 3.37% / 1200% (12 CPUs available)

Container memory usage 229.73MB / 7.48GB

Show charts

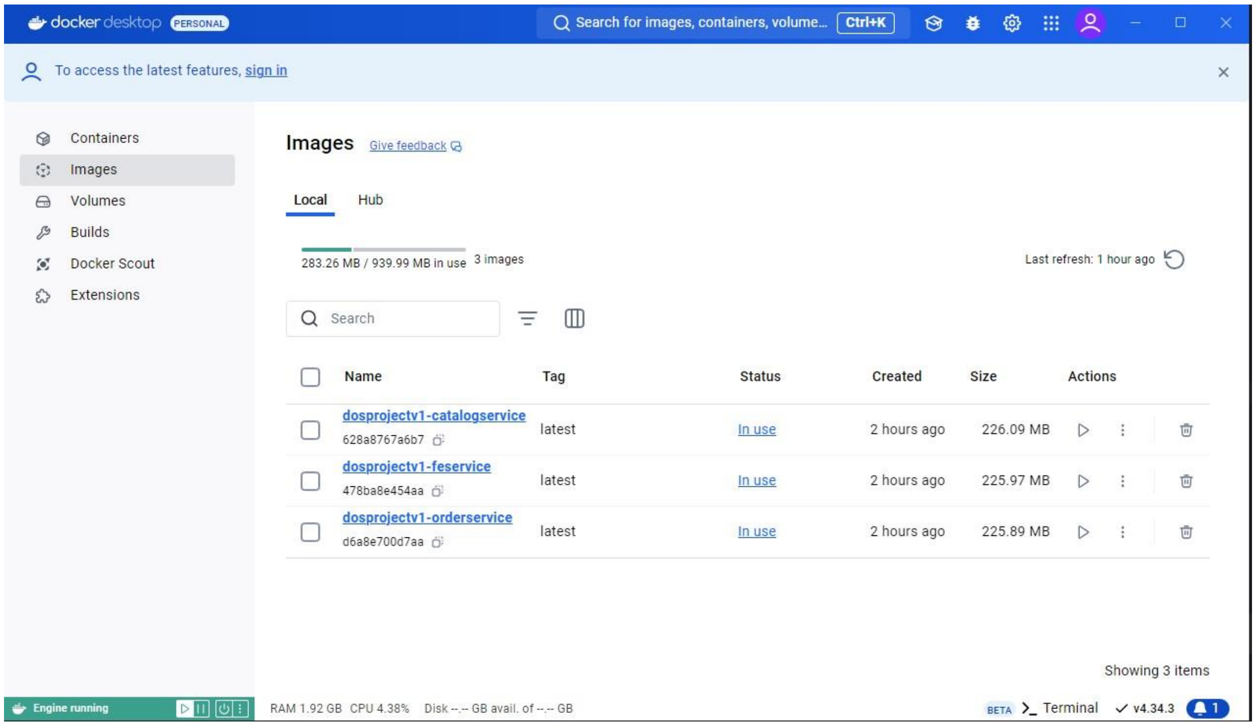
Only show running containers

Name	Image	Status	Port(s)	CPU (%)	Last started
dosproject		Running (3/3)		0%	2 hours ago
feservic 82868c6b	dosprojectv1-feservice:<none>	Running	8001:8001	0%	2 hours ago
catalogs 2be54941	dosprojectv1-catalogservice:<none>	Running	8000:8000	0%	2 hours ago
orderser ca6968a3	dosprojectv1-orderservice:<none>	Running	8002:8002	0%	2 hours ago

Showing 4 items

Engine running RAM 1.92 GB CPU 0.92% Disk --.-- GB avail. of --.-- GB

Images


docker desktop PERSONAL

To access the latest features, sign in

Images Give feedback

Local Hub

283.26 MB / 939.99 MB in use 3 images

Last refresh: 1 hour ago

Name	Tag	Status	Created	Size
dosprojectv1-catalogservice 628a8767a6b7	latest	In use	2 hours ago	226.09 MB
dosprojectv1-feservice 478ba8e454aa	latest	In use	2 hours ago	225.97 MB
dosprojectv1-orderservice d6a8e700d7aa	latest	In use	2 hours ago	225.89 MB

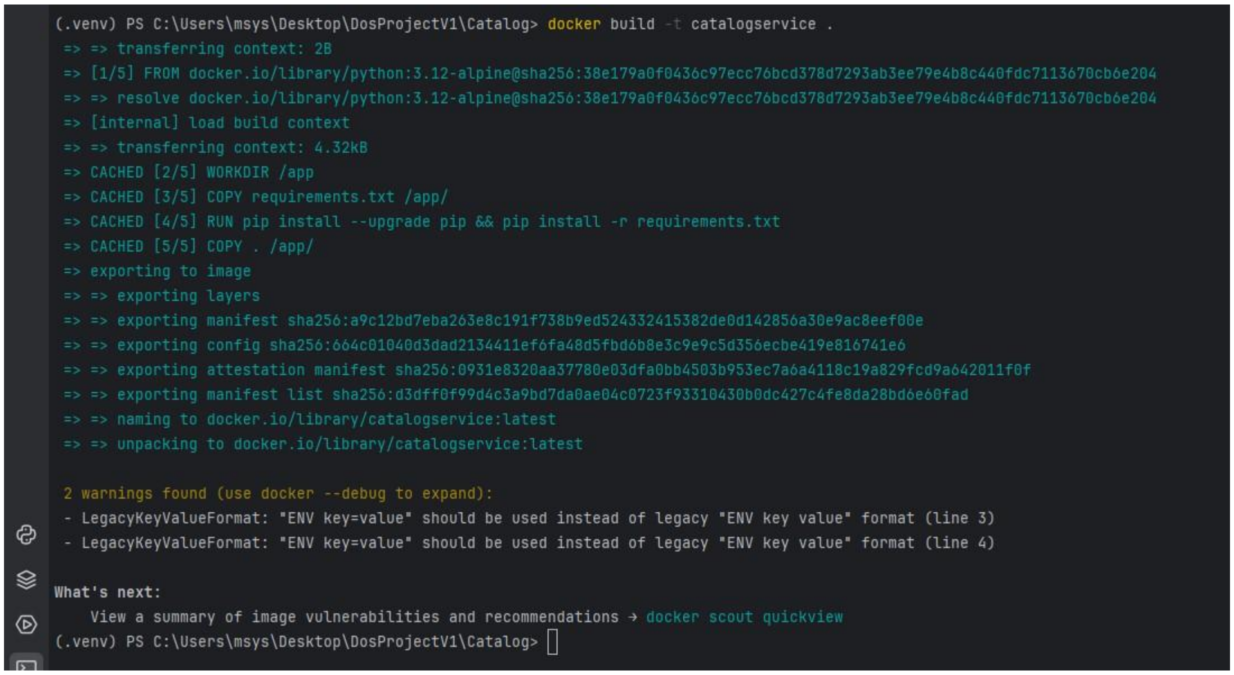
Showing 3 items

Engine running RAM 1.92 GB CPU 4.38% Disk --.-- GB avail. of --.-- GB

Run the containers using terminal commands

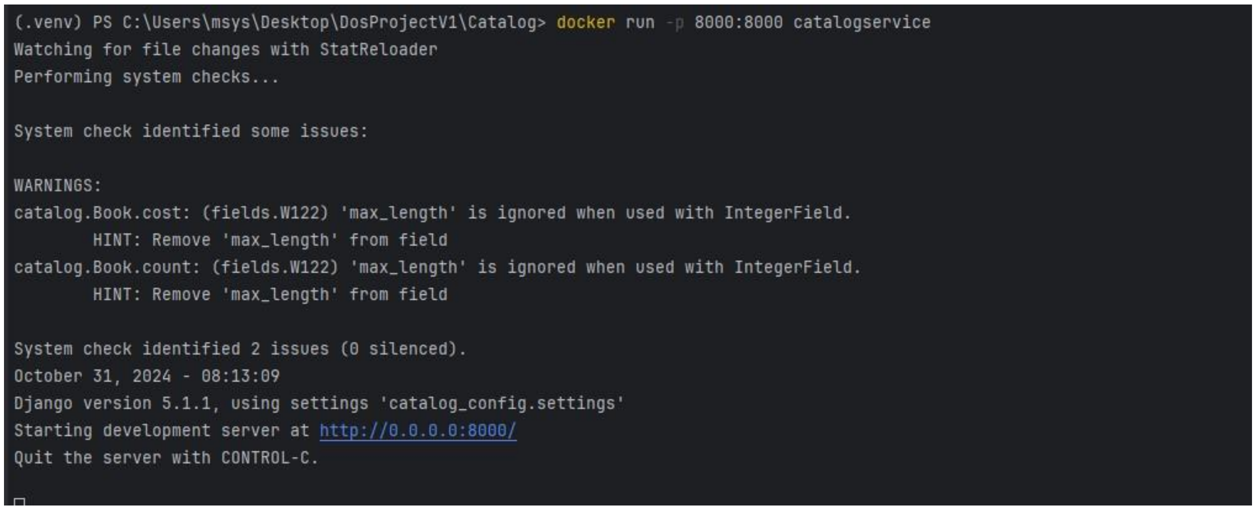
Start each service individually first to check for any issues and confirm they're working properly. This way, you can troubleshoot services one by one, making it easier to identify problems before combining them. Once everything runs smoothly, use the compose file to start all services together, which streamlines management and deployment.

- **Building and running catalog service**
  - **Docker build –t catalogservice**



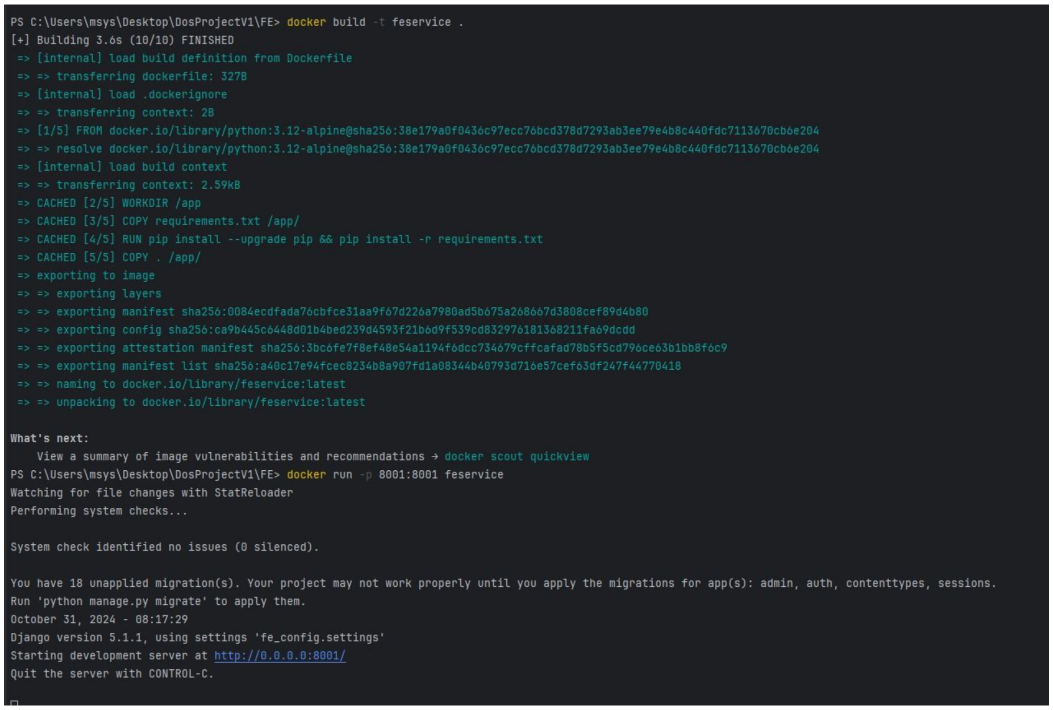
```
(.venv) PS C:\Users\msys\Desktop\DosProjectV1\Catalog> docker build -t catalogservice .
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => => resolve docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => [internal] load build context
 => => transferring context: 4.32kB
 => CACHED [2/5] WORKDIR /app
 => CACHED [3/5] COPY requirements.txt /app/
 => CACHED [4/5] RUN pip install --upgrade pip && pip install -r requirements.txt
 => CACHED [5/5] COPY . /app/
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:a9c12bd7eba263e8c191f738b9ed524332415382de0d142856a30e9ac8eef00e
 => => exporting config sha256:664c01040d3dad2134411ef6fa48d5fbd6b8e3c9e9c5d356ecbe419e816741e6
 => => exporting attestation manifest sha256:0931e8320aa37780e03dfa0bb4503b953ec7a6a4118c19a829fcd9a642011f0f
 => => exporting manifest list sha256:d3dff0f99d4c3a9bd7da0ae04c0723f93310430b0dc427c4fe8da28bd6e60fad
 => => naming to docker.io/library/catalogservice:latest
 => => unpacking to docker.io/library/catalogservice:latest

 2 warnings found (use docker --debug to expand):
 - LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "ENV key value" format (line 3)
 - LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "ENV key value" format (line 4)

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
(.venv) PS C:\Users\msys\Desktop\DosProjectV1\Catalog>
```

- docker run -p 8000:8000 catalogservice



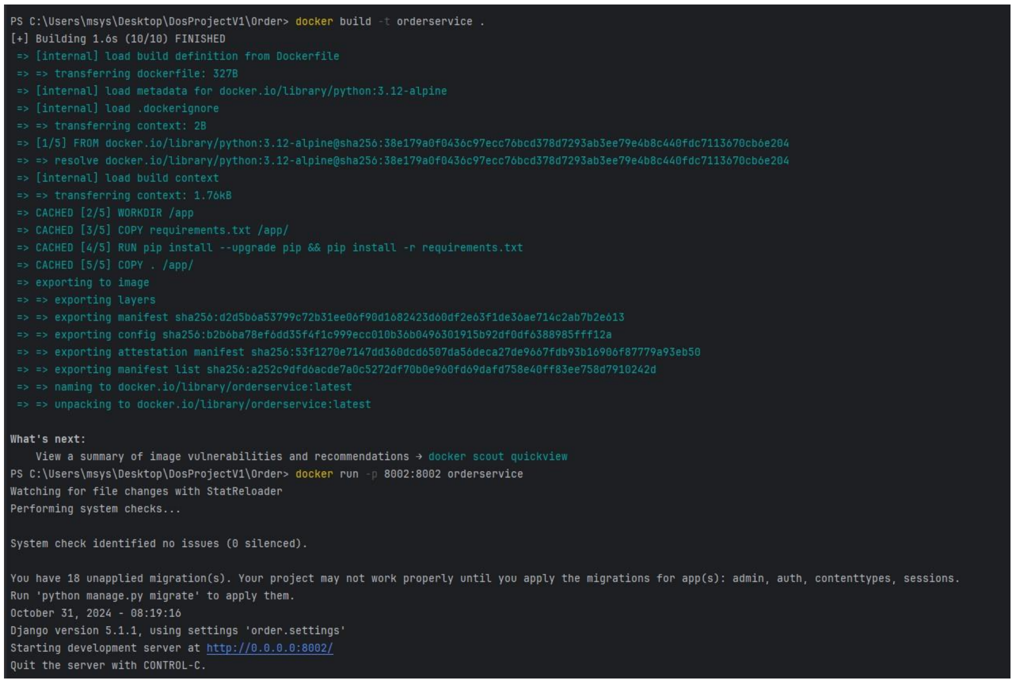
```
(.venv) PS C:\Users\msys\Desktop\DosProjectV1\Catalog> docker run -p 8000:8000 catalogservice
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
catalog.Book.cost: (fields.W122) 'max_length' is ignored when used with IntegerField.
        HINT: Remove 'max_length' from field
catalog.Book.count: (fields.W122) 'max_length' is ignored when used with IntegerField.
        HINT: Remove 'max_length' from field

System check identified 2 issues (0 silenced).
October 31, 2024 - 08:13:09
Django version 5.1.1, using settings 'catalog_config.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

- **Building and running fe service**
  - Docker build –t feservice



```
PS C:\Users\msys\Desktop\DosProjectV1\FE> docker build -t feservice .
[+] Building 3.6s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 327B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => => resolve docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => [internal] load build context
 => => transferring context: 2.59kB
 => CACHED [2/5] WORKDIR /app
 => CACHED [3/5] COPY requirements.txt /app/
 => CACHED [4/5] RUN pip install --upgrade pip && pip install -r requirements.txt
 => CACHED [5/5] COPY . /app/
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:0084ecdfada76cbfce31aa9f67d226a7980ad5b675a268667d3808cef89d4b80
 => => exporting config sha256:ca9b445c6448d01b4bed239d4593f21b6d9f539cd83297618136821 1fa69dcdd
 => => exporting attestation manifest sha256:3bc6fe7f8ef48e54a1194f6dcc734679cffcafad78b5f5cd796ce63b1bb8f6c9
 => => exporting manifest list sha256:a40c17e94fcec8234b8a907fd1a08344b40793d716e57cef63df247f44770418
 => => naming to docker.io/library/feservice:latest
 => => unpacking to docker.io/library/feservice:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\msys\Desktop\DosProjectV1\FE> docker run -p 8001:8001 feservice
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
October 31, 2024 - 08:17:29
Django version 5.1.1, using settings 'fe_config.settings'
Starting development server at http://0.0.0.0:8001/
Quit the server with CONTROL-C.
```

- **Building and running order service**
  - Docker build –t orderservice

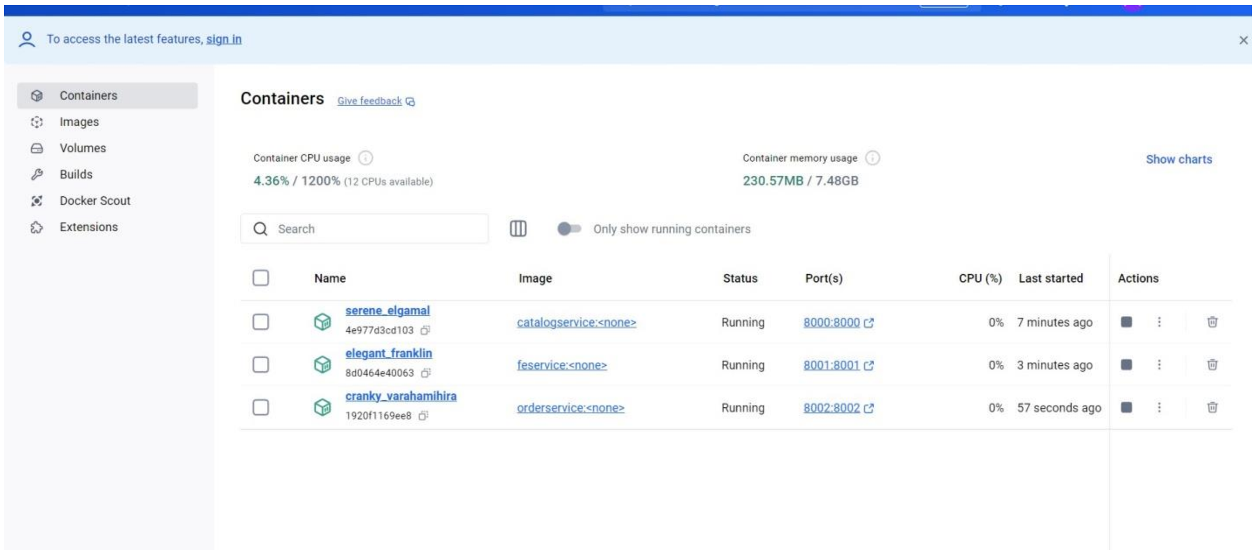


```
PS C:\Users\msys\Desktop\DosProjectV1\Order> docker build -t orderservice .
[+] Building 1.6s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 327B
 => [internal] load metadata for docker.io/library/python:3.12-alpine
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => => resolve docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204
 => [internal] load build context
 => => transferring context: 1.76kB
 => CACHED [2/5] WORKDIR /app
 => CACHED [3/5] COPY requirements.txt /app/
 => CACHED [4/5] RUN pip install --upgrade pip && pip install -r requirements.txt
 => CACHED [5/5] COPY . /app/
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:d2d5b6a53799c72b31ee06f90d1682423d60df2e63f1de36ae714c2ab7b2e613
 => => exporting config sha256:b2b6ba78ef6dd35f4f1c999ecc010b36b0496301915b92df0df6388985fff12a
 => => exporting attestation manifest sha256:53f1270e7147dd360dcd6507da56deca27de9667fdb93b16906f87779a93eb50
 => => exporting manifest list sha256:a252c9dfd6acde7a0c5272df70b0e960fd69dafd758e40ff83ee758d7910242d
 => => naming to docker.io/library/orderservice:latest
 => => unpacking to docker.io/library/orderservice:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\msys\Desktop\DosProjectV1\Order> docker run -p 8002:8002 orderservice
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
October 31, 2024 - 08:19:16
Django version 5.1.1, using settings 'order.settings'
Starting development server at http://0.0.0.0:8002/
Quit the server with CONTROL-C.
```

- docker-compose down command

This command stops and removes all containers, networks, and other resources defined in the docker-compose.yml file. It’s used to tear down all running services in the docker-compose setup.

Containers



To access the latest features, sign in

Containers Give feedback

Container CPU usage 4.36% / 1200% (12 CPUs available)

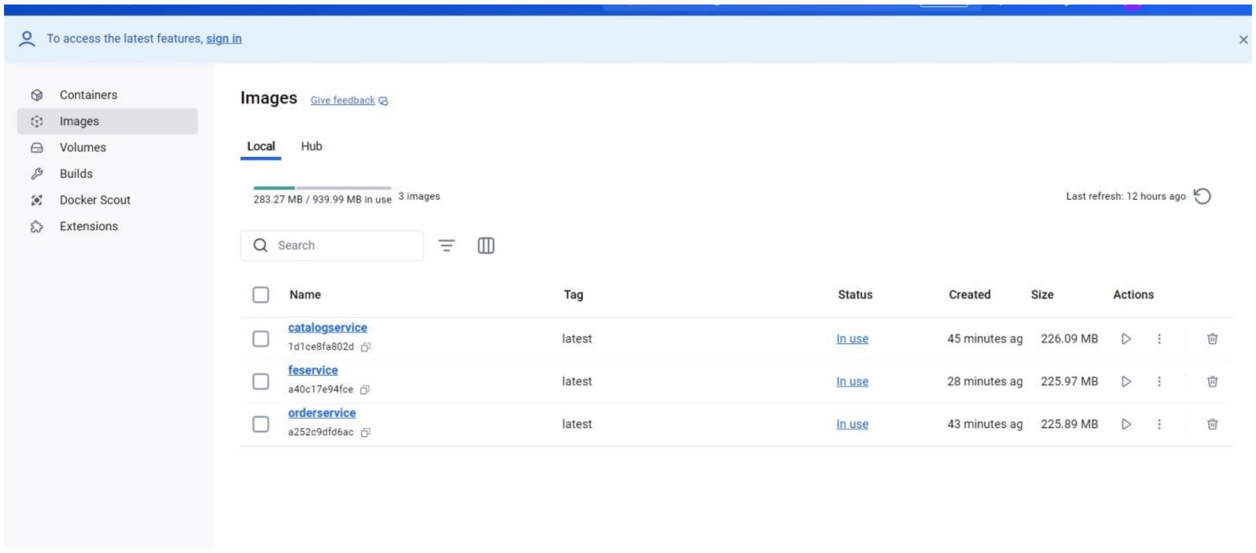
Container memory usage 230.57MB / 7.48GB

Show charts

Only show running containers

Name	Image	Status	Port(s)	CPU (%)	Last started
serene_elgamal 4e977d3cd103	catalogservice:<none>	Running	8000:8000	0%	7 minutes ago
elegant_franklin 8d0464e40063	feservice:<none>	Running	8001:8001	0%	3 minutes ago
cranky_varahamihira 1920f1169ee8	orderservice:<none>	Running	8002:8002	0%	57 seconds ago

Images



To access the latest features, sign in

Images Give feedback

283.27 MB / 939.99 MB in use 3 images

Last refresh: 12 hours ago

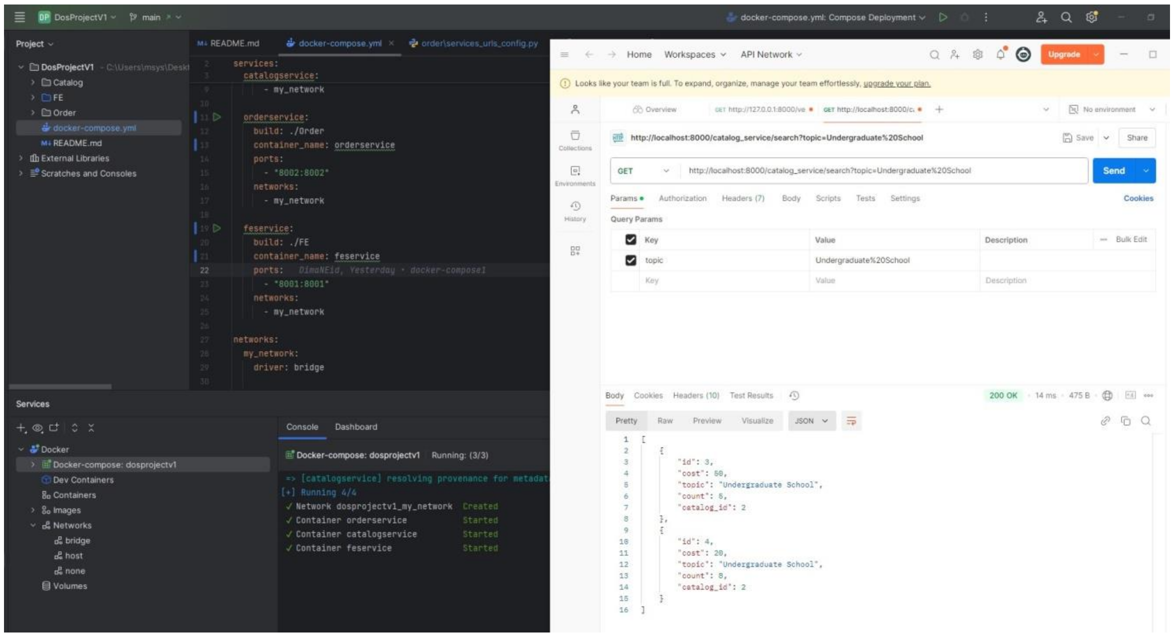
Name	Tag	Status	Created	Size
catalogservice 1d1ce8fa802d	latest	In use	45 minutes ag	226.09 MB
feservice a40c17e94fce	latest	In use	28 minutes ag	225.97 MB
orderservice a252c9dfd6ac	latest	In use	43 minutes ag	225.89 MB

<u>Testing</u>

The following screenshots confirm that the services are correctly networked and accessible as expected in Docker.

Catalog service

- **search**



```yaml
services:
  catalogservice:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:   DimaNEid, Yesterday • docker-compose1
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

Docker-compose: dosprojectv1   Running: (3/3)

```
=> [catalogservice] resolving provenance for metadat
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
```

Looks like your team is full. To expand, organize, manage your team effortlessly, upgrade your plan.

http://localhost:8000/catalog_service/search?topic=Undergraduate%20School

GET http://localhost:8000/catalog_service/search?topic=Undergraduate%20School

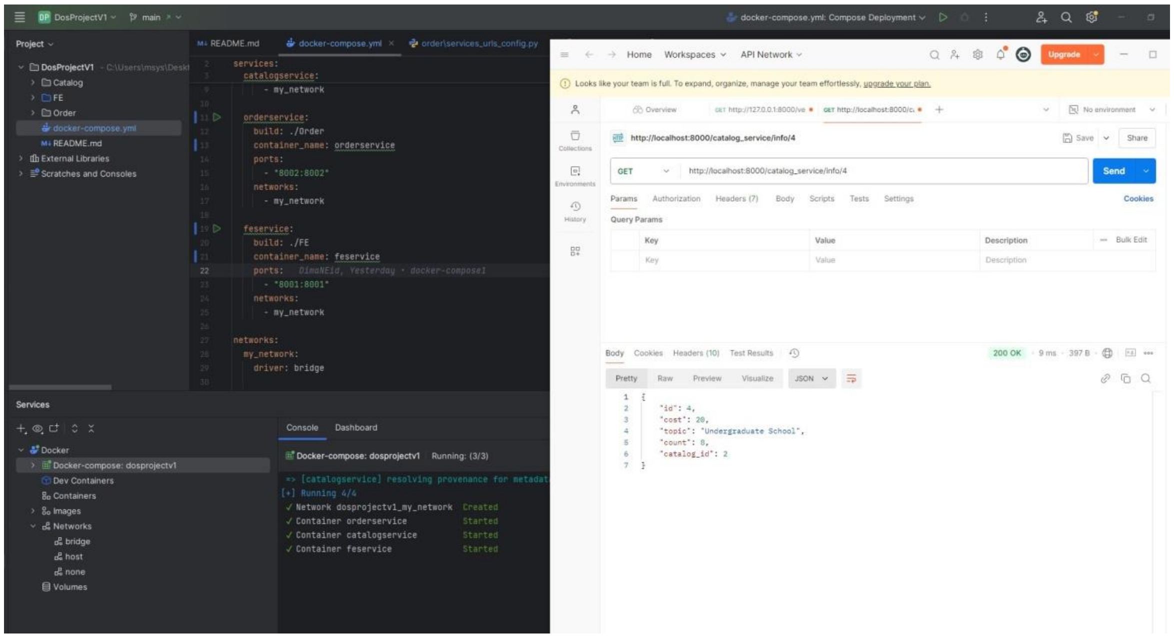
Query Params

	Key	Value	Description
✓	topic	Undergraduate%20School	

200 OK · 14 ms · 475 B

```json
[
    {
        "id": 3,
        "cost": 50,
        "topic": "Undergraduate School",
        "count": 5,
        "catalog_id": 2
    },
    {
        "id": 4,
        "cost": 20,
        "topic": "Undergraduate School",
        "count": 8,
        "catalog_id": 2
    }
]
```

- **info**



```yaml
services:
  catalogservice:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:   DimaNEid, Yesterday • docker-compose1
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

Docker-compose: dosprojectv1   Running: (3/3)

```
=> [catalogservice] resolving provenance for metadat
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
```

Looks like your team is full. To expand, organize, manage your team effortlessly, upgrade your plan.

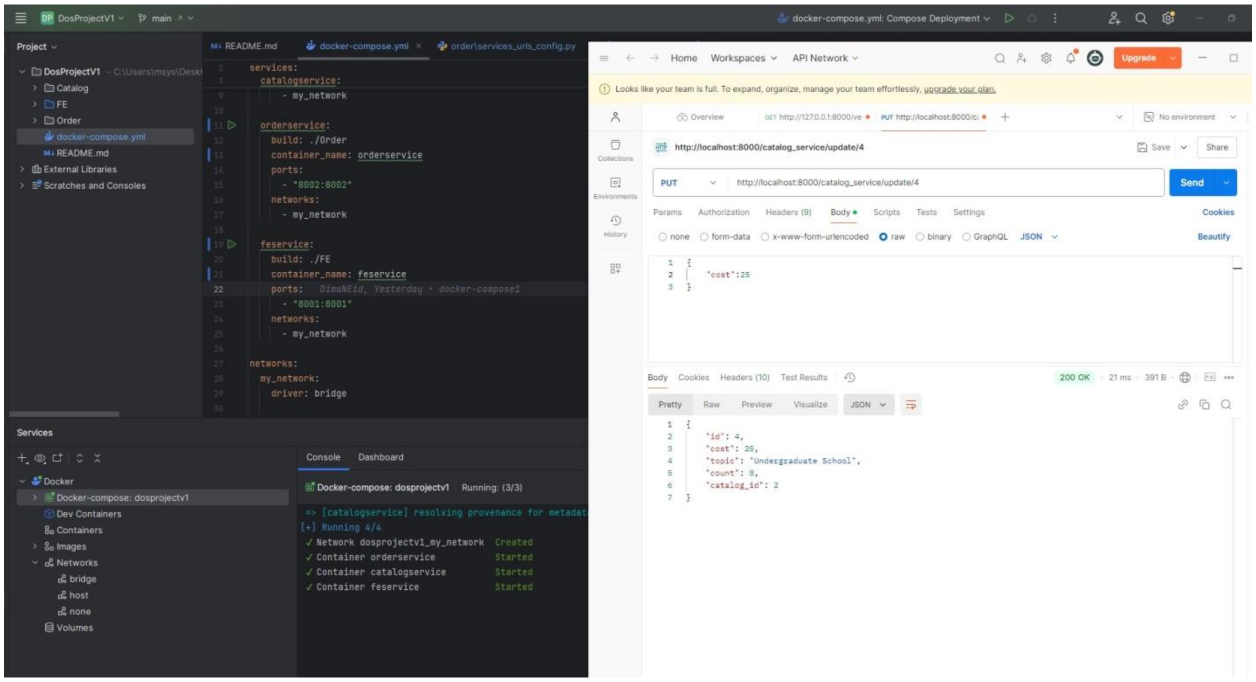
http://localhost:8000/catalog_service/info/4

GET http://localhost:8000/catalog_service/info/4

200 OK · 9 ms · 397 B

```json
{
    "id": 4,
    "cost": 20,
    "topic": "Undergraduate School",
    "count": 8,
    "catalog_id": 2
}
```

- Update



```yaml
services:
  catalogservice:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:   DimaNEid, Yesterday • docker-compose1
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

Docker-compose: dosprojectv1   Running: (3/3)

```
=> [catalogservice] resolving provenance for metadat
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
```

PUT http://localhost:8000/catalog_service/update/4

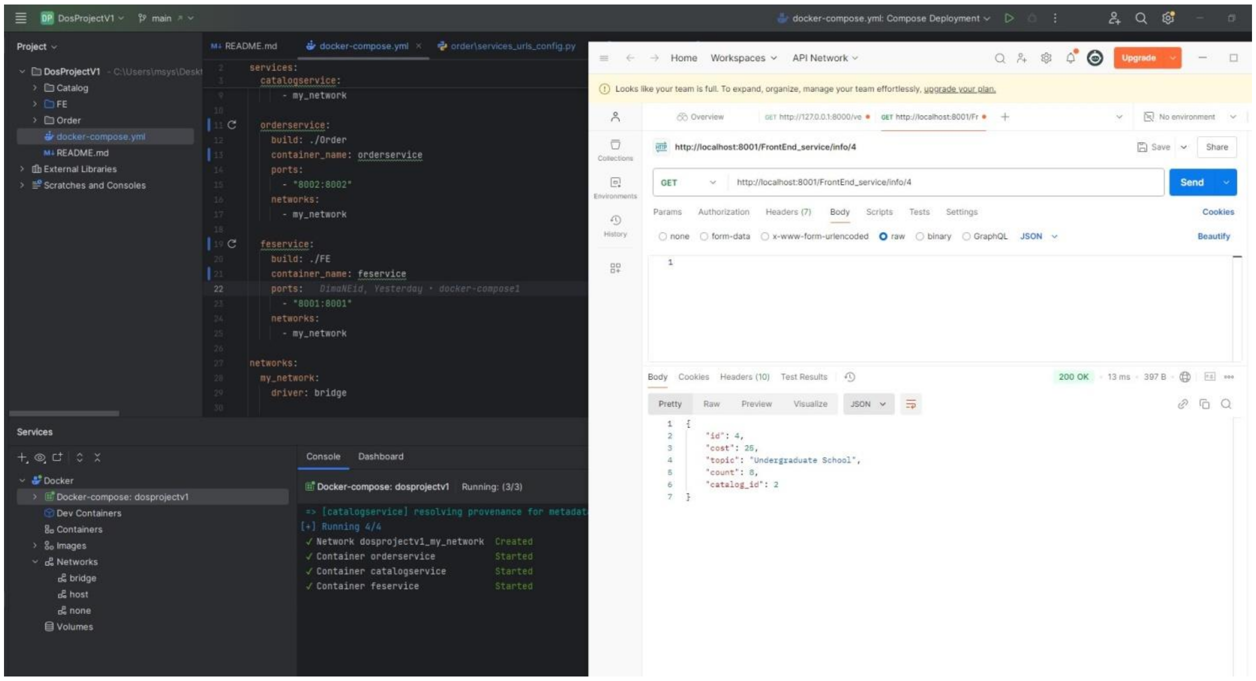
```json
{
    "cost":25
}
```

200 OK · 21 ms · 391 B

```json
{
    "id": 4,
    "cost": 25,
    "topic": "Undergraduate School",
    "count": 8,
    "catalog_id": 2
}
```

FE service

- **info**



```yaml
services:
  catalogservice:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:   DimaNEid, Yesterday • docker-compose1
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

Docker-compose: dosprojectv1   Running: (3/3)

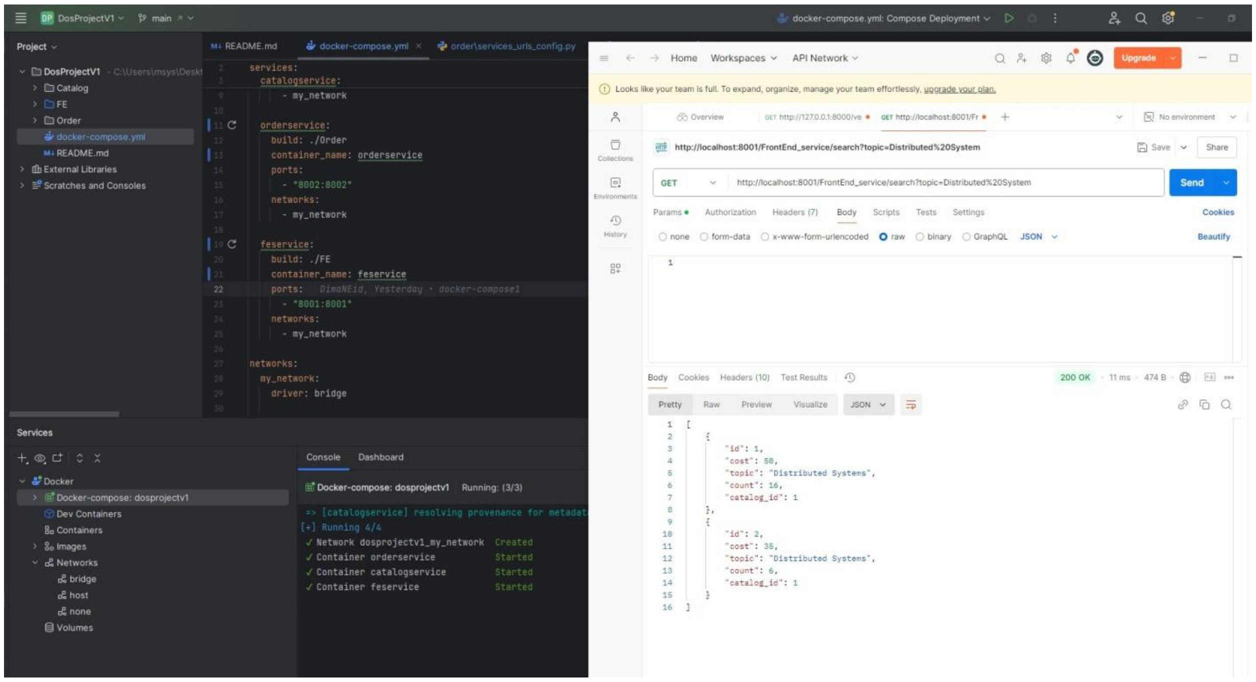
```
=> [catalogservice] resolving provenance for metadat
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
```

GET http://localhost:8001/FrontEnd_service/info/4

200 OK · 13 ms · 397 B

```json
{
    "id": 4,
    "cost": 25,
    "topic": "Undergraduate School",
    "count": 8,
    "catalog_id": 2
}
```

- **search**



```
services:
  catalogservice:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:
    build: ./FE
    container_name: feservice
    ports:   DimaNEid, Yesterday • docker-compose1
      - "8001:8001"
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

```
Docker-compose: dosprojectv1    Running: (3/3)
=> [catalogservice] resolving provenance for metadat
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
```

Looks like your team is full. To expand, organize, manage your team effortlessly, upgrade your plan.

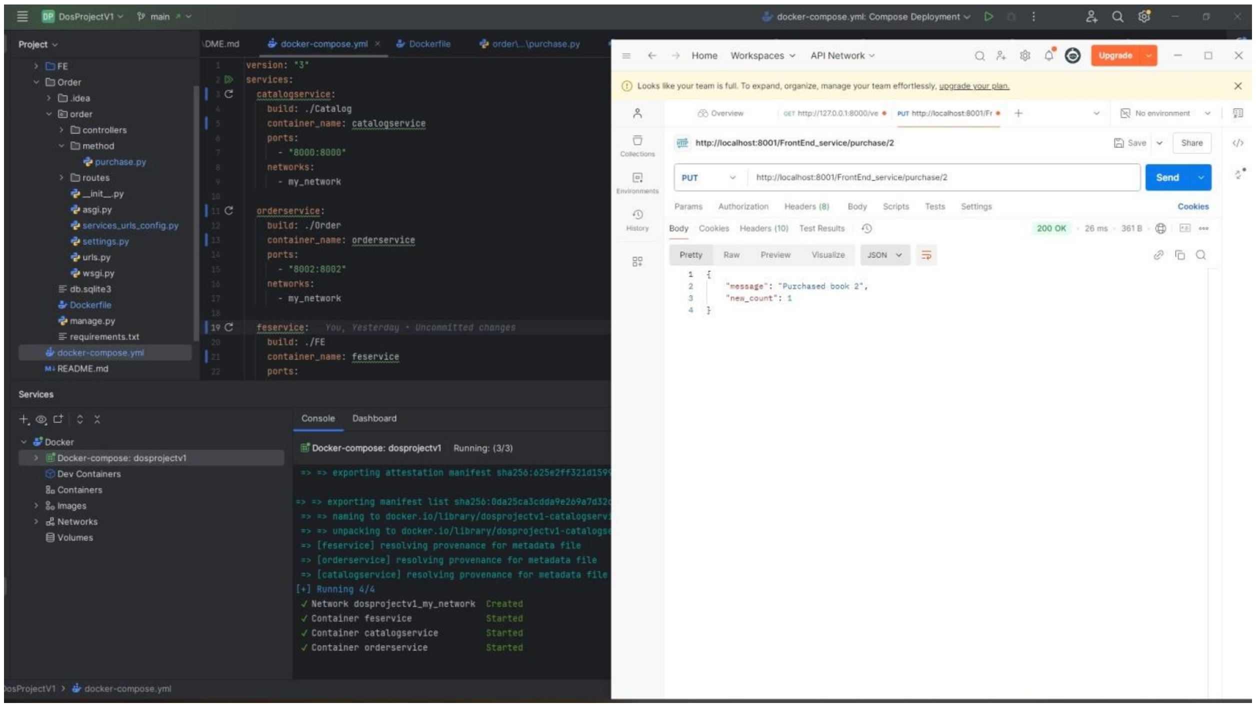
http://localhost:8001/FrontEnd_service/search?topic=Distributed%20System

GET http://localhost:8001/FrontEnd_service/search?topic=Distributed%20System

200 OK · 11 ms · 474 B

```
[
    {
        "id": 1,
        "cost": 50,
        "topic": "Distributed Systems",
        "count": 16,
        "catalog_id": 1
    },
    {
        "id": 2,
        "cost": 35,
        "topic": "Distributed Systems",
        "count": 6,
        "catalog_id": 1
    }
]
```

- Purchase



```
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:   You, Yesterday • Uncommitted changes
    build: ./FE
    container_name: feservice
    ports:
```

```
Docker-compose: dosprojectv1    Running: (3/3)
=> => exporting attestation manifest sha256:625e2ff321d159

=> => exporting manifest list sha256:0da25ca3cdda9e269a7d32
=> => naming to docker.io/library/dosprojectv1-catalogserv
=> => unpacking to docker.io/library/dosprojectv1-catalogs
=> [feservice] resolving provenance for metadata file
=> [orderservice] resolving provenance for metadata file
=> [catalogservice] resolving provenance for metadata file
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container feservice              Started
 ✓ Container catalogservice         Started
 ✓ Container orderservice           Started
```

Looks like your team is full. To expand, organize, manage your team effortlessly, upgrade your plan.

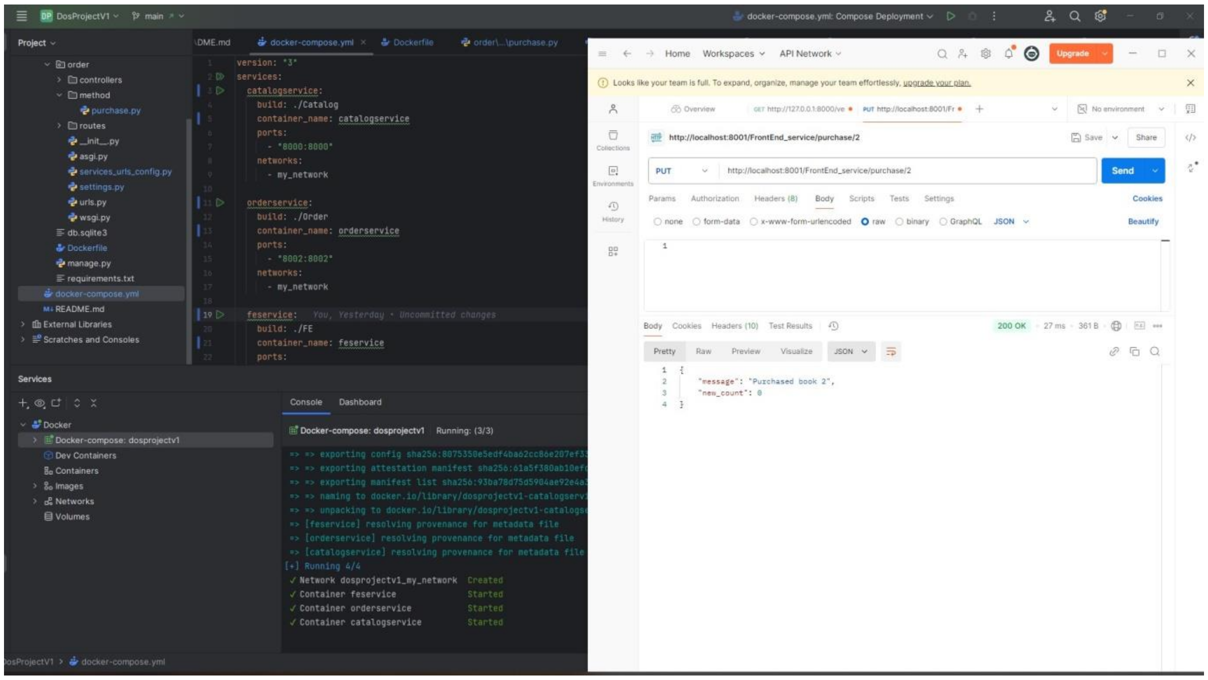
http://localhost:8001/FrontEnd_service/purchase/2

PUT http://localhost:8001/FrontEnd_service/purchase/2

200 OK · 26 ms · 361 B

```
{
    "message": "Purchased book 2",
    "new_count": 1
}
```

- No items left case



```yaml
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:   You, Yesterday • Uncommitted changes
    build: ./FE
    container_name: feservice
    ports:
```

Docker-compose: dosprojectv1   Running: (3/3)

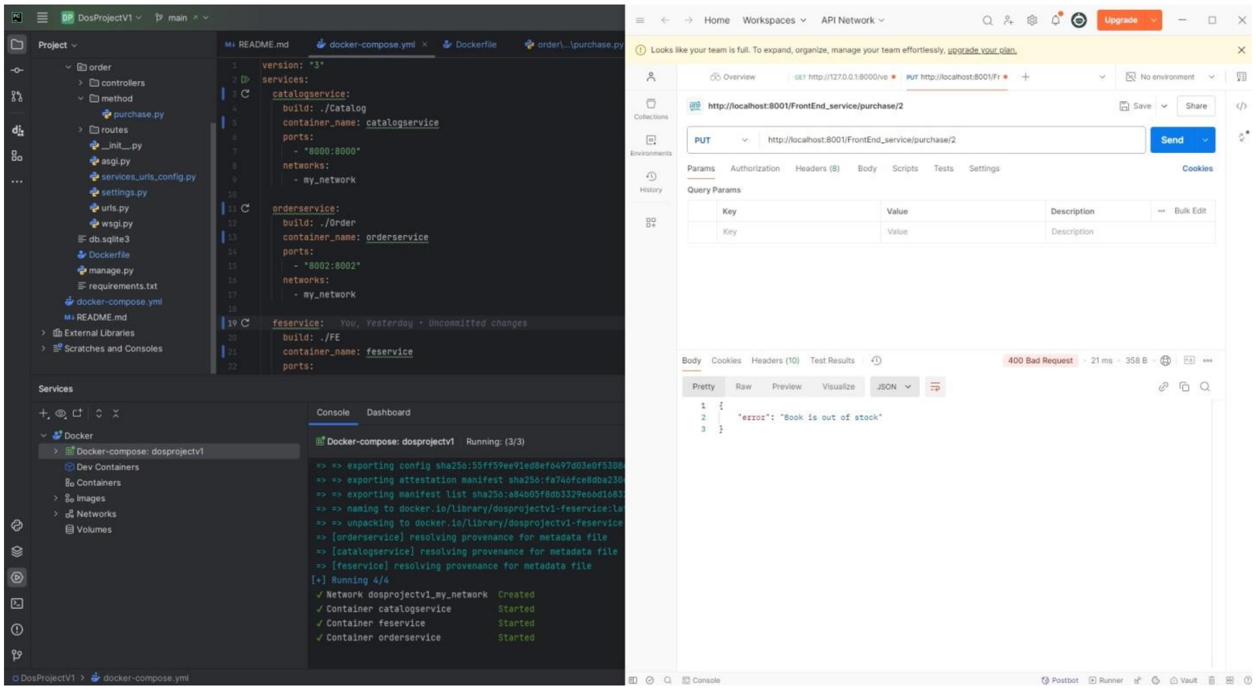
```
=> => exporting config sha256:8075358e5edf4ba62cc86e207ef3
=> => exporting attestation manifest sha256:61a5f380ab10ef
=> => exporting manifest list sha256:93ba78d75d5904ae92e4a3
=> => naming to docker.io/library/dosprojectv1-catalogserv
=> => unpacking to docker.io/library/dosprojectv1-catalogse
=> [feservice] resolving provenance for metadata file
=> [orderservice] resolving provenance for metadata file
=> [catalogservice] resolving provenance for metadata file
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container feservice              Started
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
```

PUT http://localhost:8001/FrontEnd_service/purchase/2

200 OK · 27 ms · 361 B

```json
{
    "message": "Purchased book 2",
    "new_count": 0
}
```

- Out of stock case



```yaml
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:   You, Yesterday • Uncommitted changes
    build: ./FE
    container_name: feservice
    ports:
```

Docker-compose: dosprojectv1   Running: (3/3)

```
=> => exporting config sha256:55ff59ee91ed8ef6497d03e0f5308
=> => exporting attestation manifest sha256:fa746fce8dba230
=> => exporting manifest list sha256:a84b05f8db3329e66d1683
=> => naming to docker.io/library/dosprojectv1-feservice:la
=> => unpacking to docker.io/library/dosprojectv1-feservice
=> [orderservice] resolving provenance for metadata file
=> [catalogservice] resolving provenance for metadata file
=> [feservice] resolving provenance for metadata file
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container catalogservice         Started
 ✓ Container feservice              Started
 ✓ Container orderservice           Started
```

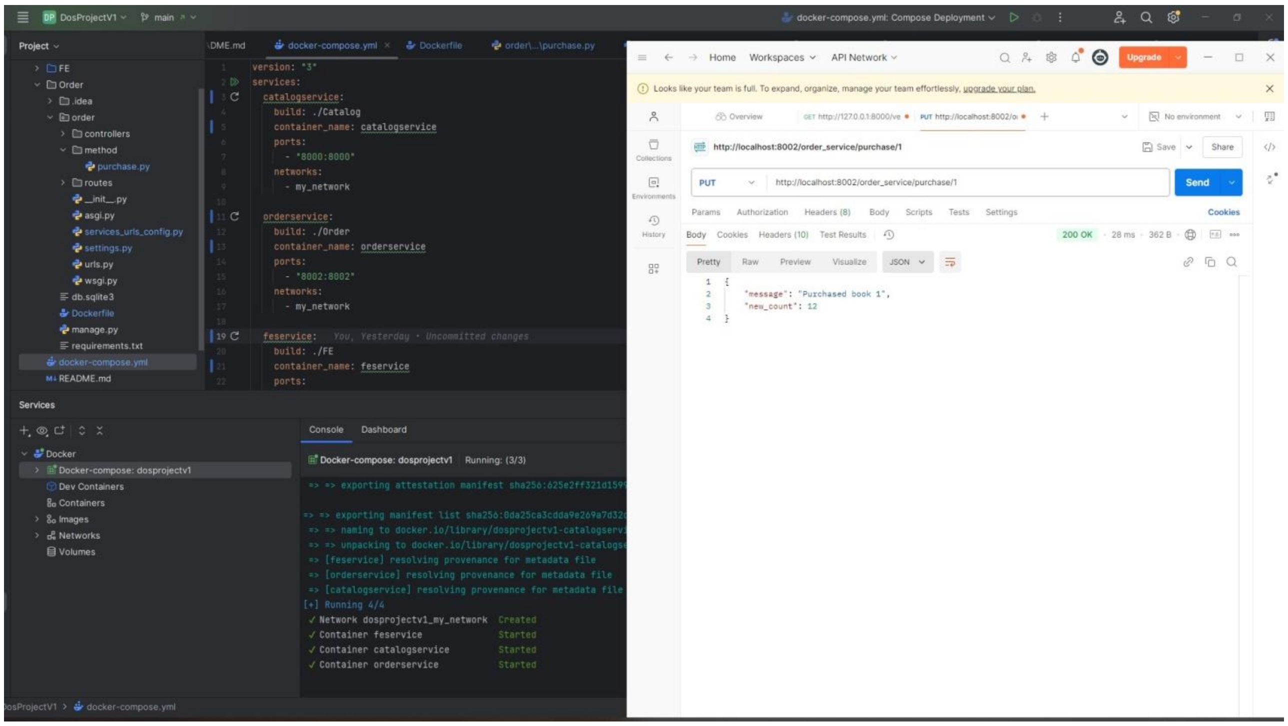
PUT http://localhost:8001/FrontEnd_service/purchase/2

400 Bad Request · 21 ms · 358 B

```json
{
    "error": "Book is out of stock"
}
```

Order service

- **Purchase**



```yaml
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:   You, Yesterday • Uncommitted changes
    build: ./FE
    container_name: feservice
    ports:
```

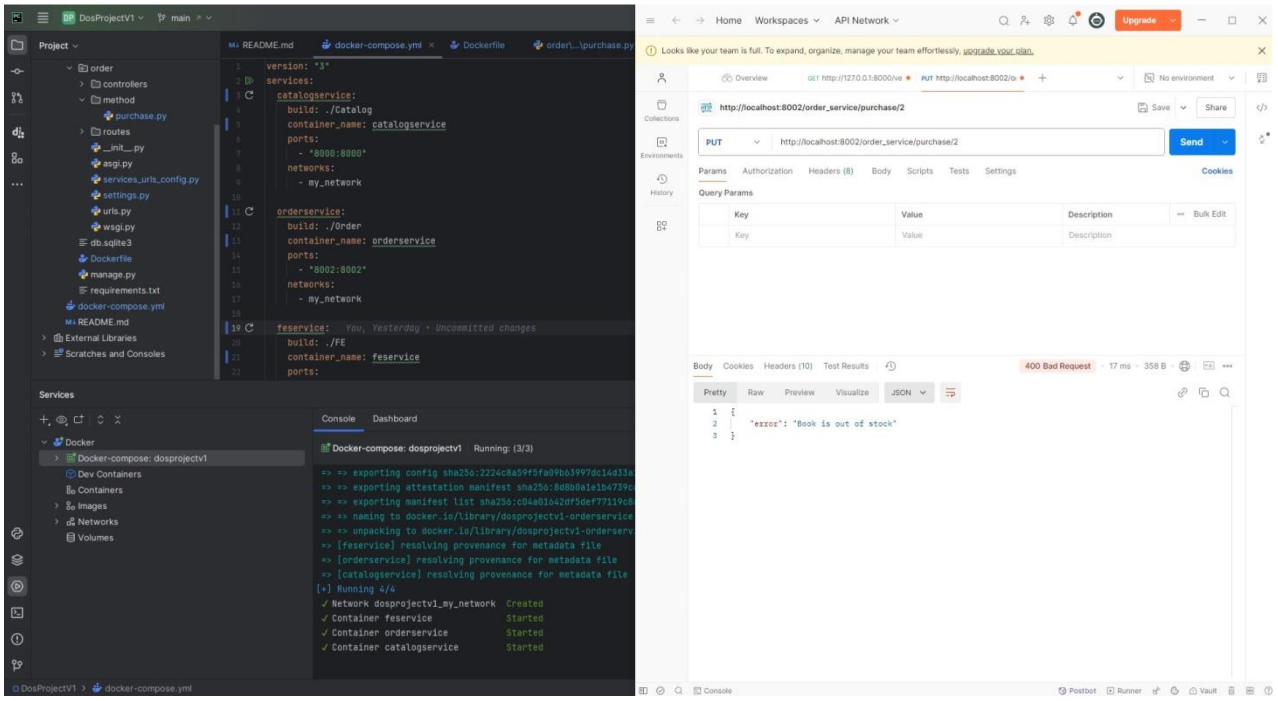
```
Docker-compose: dosprojectv1    Running: (3/3)
 => => exporting attestation manifest sha256:625e2ff321d159
=> => exporting manifest list sha256:0da25ca3cdda9e269a7d32
 => => naming to docker.io/library/dosprojectv1-catalogserv
 => => unpacking to docker.io/library/dosprojectv1-catalogse
 => [feservice] resolving provenance for metadata file
 => [orderservice] resolving provenance for metadata file
 => [catalogservice] resolving provenance for metadata file
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container feservice              Started
 ✓ Container catalogservice         Started
 ✓ Container orderservice           Started
```

PUT http://localhost:8002/order_service/purchase/1

200 OK · 28 ms · 362 B

```json
{
    "message": "Purchased book 1",
    "new_count": 12
}
```

- Out of stock case



```yaml
version: "3"
services:
  catalogservice:
    build: ./Catalog
    container_name: catalogservice
    ports:
      - "8000:8000"
    networks:
      - my_network

  orderservice:
    build: ./Order
    container_name: orderservice
    ports:
      - "8002:8002"
    networks:
      - my_network

  feservice:   You, Yesterday • Uncommitted changes
    build: ./FE
    container_name: feservice
    ports:
```

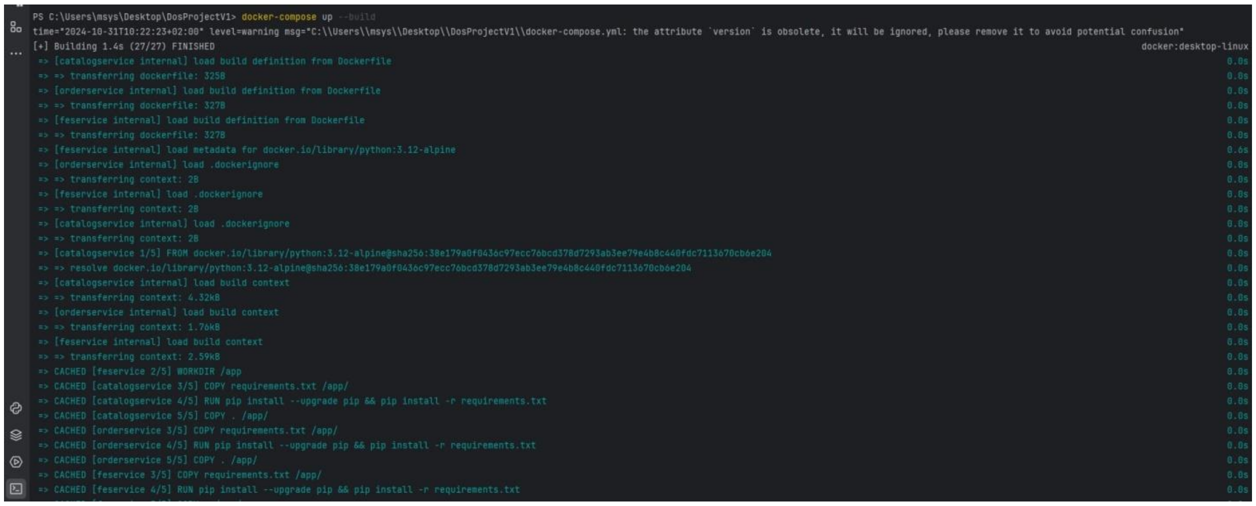
```
Docker-compose: dosprojectv1    Running: (3/3)
 => => exporting config sha256:2224c8a59f5fa09b63997dc14d33a
 => => exporting attestation manifest sha256:8d8b0a1e1b4739c
 => => exporting manifest list sha256:c04a01642df5def77119c8
 => => naming to docker.io/library/dosprojectv1-orderservice
 => => unpacking to docker.io/library/dosprojectv1-orderserv
 => [feservice] resolving provenance for metadata file
 => [orderservice] resolving provenance for metadata file
 => [catalogservice] resolving provenance for metadata file
[+] Running 4/4
 ✓ Network dosprojectv1_my_network  Created
 ✓ Container feservice              Started
 ✓ Container orderservice           Started
 ✓ Container catalogservice         Started
```

PUT http://localhost:8002/order_service/purchase/2

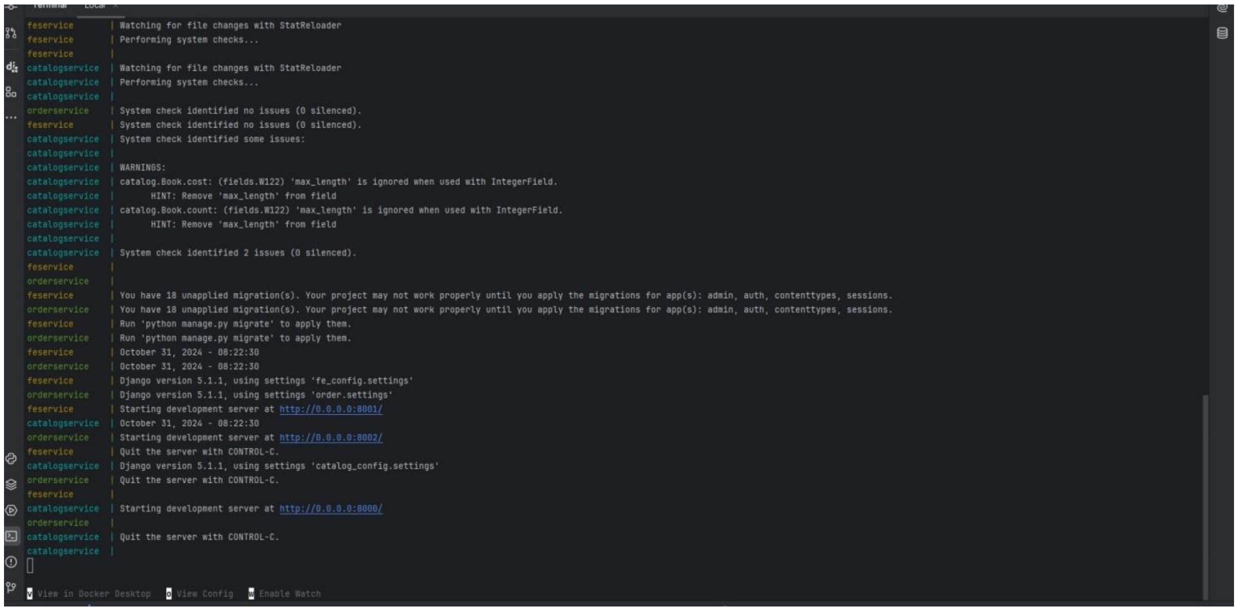
400 Bad Request · 17 ms · 358 B

```json
{
    "error": "Book is out of stock"
}
```

- **Building all services together**
  - **docker-compose up –build in DosProjectV1**

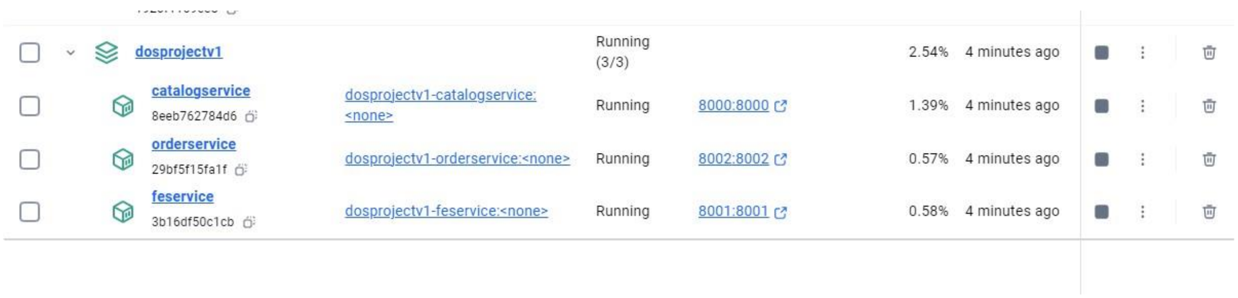


```
PS C:\Users\msys\Desktop\DosProjectV1> docker-compose up --build
time="2024-10-31T10:22:23+02:00" level=warning msg="C:\\Users\\msys\\Desktop\\DosProjectV1\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Building 1.4s (27/27) FINISHED                                                                      docker:desktop-linux
 => [catalogservice internal] load build definition from Dockerfile                                                    0.0s
 => => transferring dockerfile: 325B                                                                                   0.0s
 => [orderservice internal] load build definition from Dockerfile                                                      0.0s
 => => transferring dockerfile: 327B                                                                                   0.0s
 => [feservice internal] load build definition from Dockerfile                                                         0.0s
 => => transferring dockerfile: 327B                                                                                   0.0s
 => [feservice internal] load metadata for docker.io/library/python:3.12-alpine                                        0.6s
 => [orderservice internal] load .dockerignore                                                                         0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [feservice internal] load .dockerignore                                                                            0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [catalogservice internal] load .dockerignore                                                                       0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [catalogservice 1/5] FROM docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204  0.0s
 => => resolve docker.io/library/python:3.12-alpine@sha256:38e179a0f0436c97ecc76bcd378d7293ab3ee79e4b8c440fdc7113670cb6e204  0.0s
 => [catalogservice internal] load build context                                                                       0.0s
 => => transferring context: 4.32kB                                                                                    0.0s
 => [orderservice internal] load build context                                                                         0.0s
 => => transferring context: 1.76kB                                                                                    0.0s
 => [feservice internal] load build context                                                                            0.0s
 => => transferring context: 2.59kB                                                                                    0.0s
 => CACHED [feservice 2/5] WORKDIR /app                                                                                0.0s
 => CACHED [catalogservice 3/5] COPY requirements.txt /app/                                                            0.0s
 => CACHED [catalogservice 4/5] RUN pip install --upgrade pip && pip install -r requirements.txt                       0.0s
 => CACHED [catalogservice 5/5] COPY . /app/                                                                           0.0s
 => CACHED [orderservice 3/5] COPY requirements.txt /app/                                                              0.0s
 => CACHED [orderservice 4/5] RUN pip install --upgrade pip && pip install -r requirements.txt                         0.0s
 => CACHED [orderservice 5/5] COPY . /app/                                                                             0.0s
 => CACHED [feservice 3/5] COPY requirements.txt /app/                                                                 0.0s
 => CACHED [feservice 4/5] RUN pip install --upgrade pip && pip install -r requirements.txt                            0.0s
```

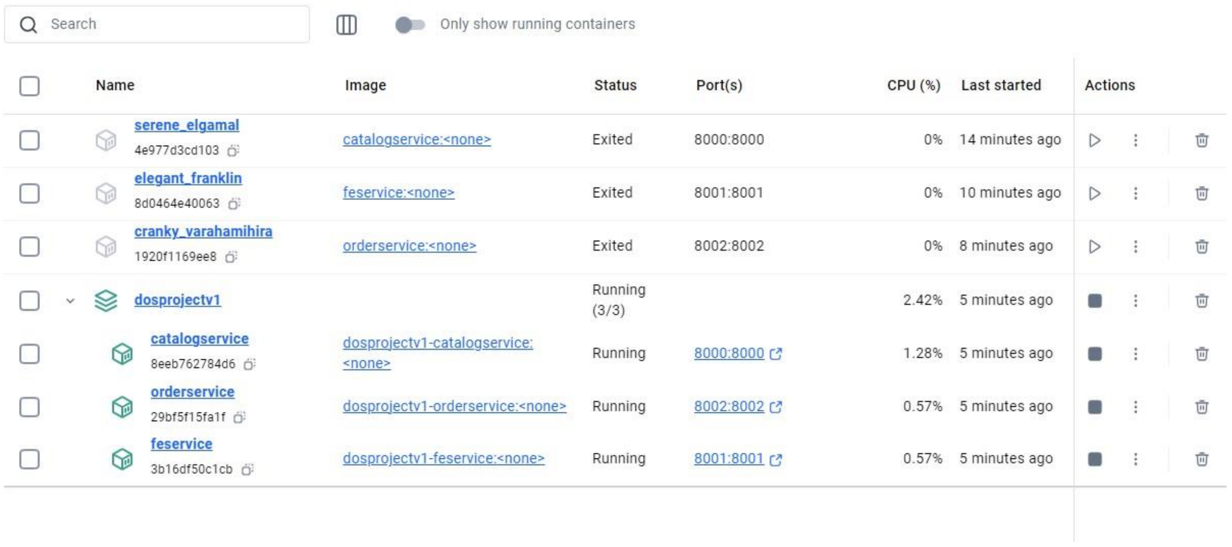


```
feservice       | Watching for file changes with StatReloader
feservice       | Performing system checks...
feservice       |
catalogservice  | Watching for file changes with StatReloader
catalogservice  | Performing system checks...
catalogservice  |
orderservice    | System check identified no issues (0 silenced).
feservice       | System check identified no issues (0 silenced).
catalogservice  | System check identified some issues:
catalogservice  |
catalogservice  | WARNINGS:
catalogservice  | catalog.Book.cost: (fields.W122) 'max_length' is ignored when used with IntegerField.
catalogservice  |       HINT: Remove 'max_length' from field
catalogservice  | catalog.Book.count: (fields.W122) 'max_length' is ignored when used with IntegerField.
catalogservice  |       HINT: Remove 'max_length' from field
catalogservice  |
catalogservice  | System check identified 2 issues (0 silenced).
feservice       |
orderservice    |
feservice       | You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
orderservice    | You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
feservice       | Run 'python manage.py migrate' to apply them.
orderservice    | Run 'python manage.py migrate' to apply them.
feservice       | October 31, 2024 - 08:22:30
orderservice    | October 31, 2024 - 08:22:30
feservice       | Django version 5.1.1, using settings 'fe_config.settings'
orderservice    | Django version 5.1.1, using settings 'order.settings'
feservice       | Starting development server at http://0.0.0.0:8001/
catalogservice  | October 31, 2024 - 08:22:30
orderservice    | Starting development server at http://0.0.0.0:8002/
feservice       | Quit the server with CONTROL-C.
catalogservice  | Django version 5.1.1, using settings 'catalog_config.settings'
orderservice    | Quit the server with CONTROL-C.
feservice       |
catalogservice  | Starting development server at http://0.0.0.0:8000/
orderservice    |
catalogservice  | Quit the server with CONTROL-C.
catalogservice  |

v View in Docker Desktop   o View Config   w Enable Watch
```

Containers take different ids each we run Docker Containers.



dosprojectv1		Running (3/3)		2.54%	4 minutes ago	
catalogservice 8eeb762784d6	dosprojectv1-catalogservice:<none>	Running	8000:8000	1.39%	4 minutes ago	
orderservice 29bf5f15fa1f	dosprojectv1-orderservice:<none>	Running	8002:8002	0.57%	4 minutes ago	
feservice 3b16df50c1cb	dosprojectv1-feservice:<none>	Running	8001:8001	0.58%	4 minutes ago	



Search

Only show running containers

Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
serene_elgamal 4e977d3cd103	catalogservice:<none>	Exited	8000:8000	0%	14 minutes ago	
elegant_franklin 8d0464e40063	feservice:<none>	Exited	8001:8001	0%	10 minutes ago	
cranky_varahamihira 1920f1169ee8	orderservice:<none>	Exited	8002:8002	0%	8 minutes ago	
dosprojectv1		Running (3/3)		2.42%	5 minutes ago	
catalogservice 8eeb762784d6	dosprojectv1-catalogservice:<none>	Running	8000:8000	1.28%	5 minutes ago	
orderservice 29bf5f15fa1f	dosprojectv1-orderservice:<none>	Running	8002:8002	0.57%	5 minutes ago	
feservice 3b16df50c1cb	dosprojectv1-feservice:<none>	Running	8001:8001	0.57%	5 minutes ago	

Container Isolation

Docker is a powerful tool for development because it provides a virtualized environment where you can freely test and process applications without risking changes to projects on your local machine. Here’s how it benefits database management and testing:

- **Isolated Environment**: Each Docker container is independent, so changes made in the container (like database updates) don’t affect the original files on the host.

- **Data Consistency**: Docker uses a copy of files like `db.sqlite3`, ensuring that testing doesn’t alter the original database. This setup preserves database state even if the container is restarted or moved.

- **Reproducibility**: Docker packages all dependencies, making it easy to replicate the exact setup on any machine, free from conflicts or missing components.

Here I have accessed db.sqlite3 copy in the container to make sure that after each request db is modified correctly using the following commands:

docker exec -it catalogservice /bin/bash

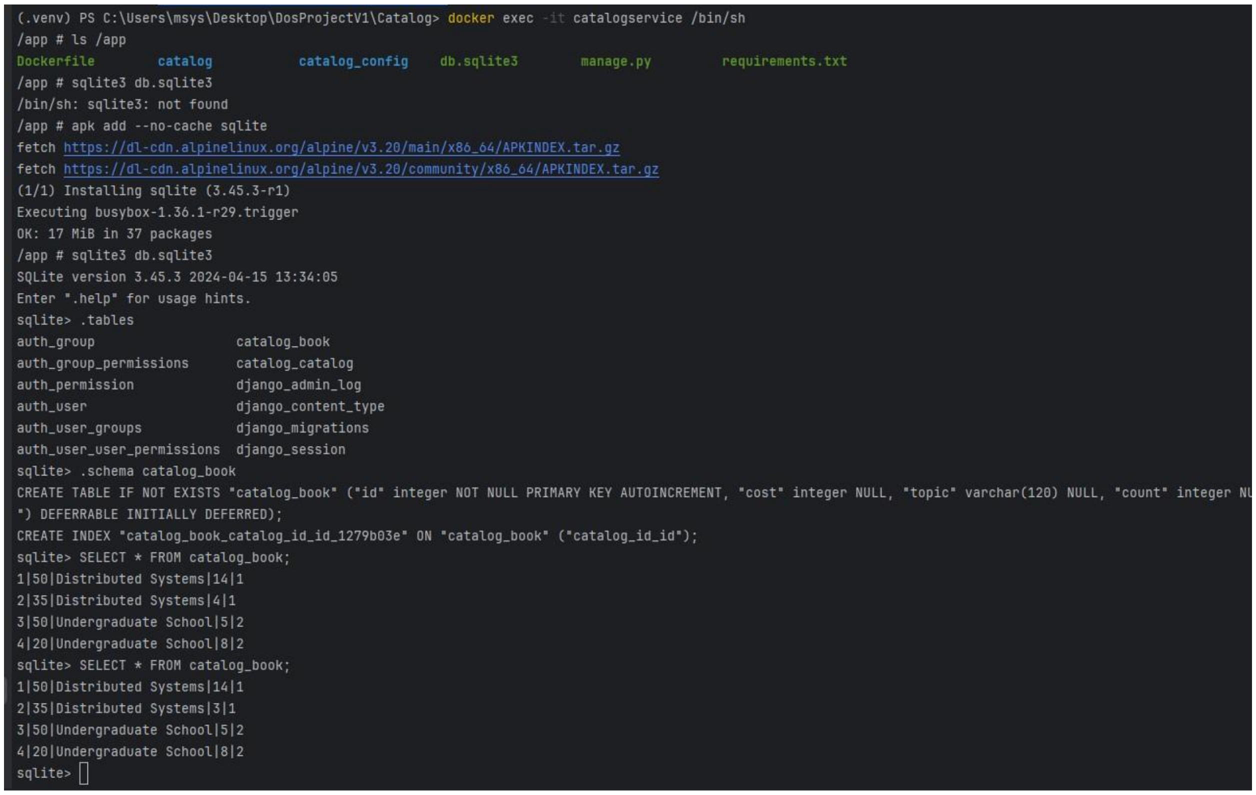
sqlite3 db.sqlite3

apk add --no-cache sqlite   (it wasn't downloaded)

.tables

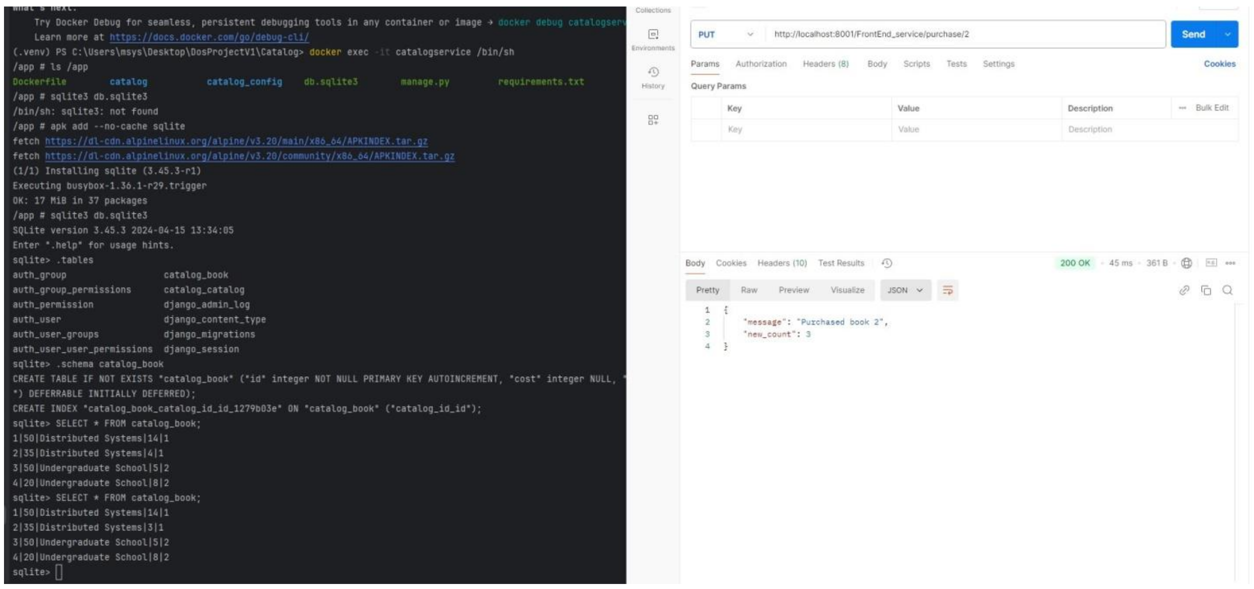
.schema catalog_book

SELECT * FROM catalog_book;



```
(.venv) PS C:\Users\msys\Desktop\DosProjectV1\Catalog> docker exec -it catalogservice /bin/sh
/app # ls /app
Dockerfile        catalog           catalog_config    db.sqlite3        manage.py         requirements.txt
/app # sqlite3 db.sqlite3
/bin/sh: sqlite3: not found
/app # apk add --no-cache sqlite
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/community/x86_64/APKINDEX.tar.gz
(1/1) Installing sqlite (3.45.3-r1)
Executing busybox-1.36.1-r29.trigger
OK: 17 MiB in 37 packages
/app # sqlite3 db.sqlite3
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .tables
auth_group                  catalog_book
auth_group_permissions      catalog_catalog
auth_permission             django_admin_log
auth_user                   django_content_type
auth_user_groups            django_migrations
auth_user_user_permissions  django_session
sqlite> .schema catalog_book
CREATE TABLE IF NOT EXISTS "catalog_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "cost" integer NULL, "topic" varchar(120) NULL, "count" integer NU
") DEFERRABLE INITIALLY DEFERRED);
CREATE INDEX "catalog_book_catalog_id_id_1279b03e" ON "catalog_book" ("catalog_id_id");
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|4|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|3|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> 
```

As we notice after this purchase request count becomes 3 for book id 2, and it matches the last displayed db in the container.



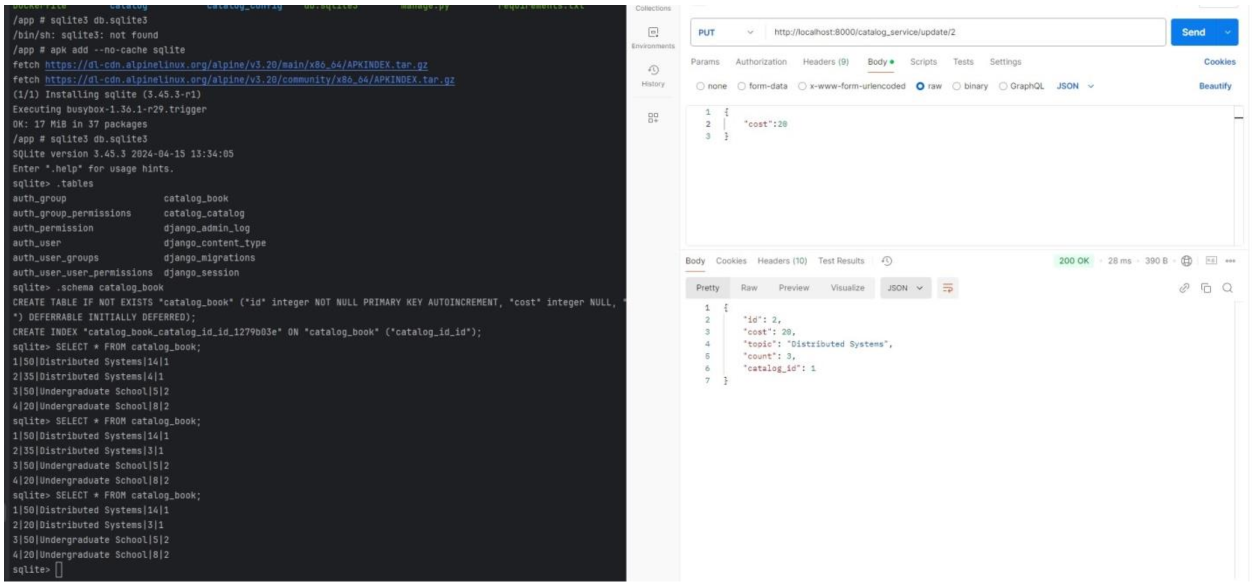
```
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug catalogserv
    Learn more at https://docs.docker.com/go/debug-cli/
(.venv) PS C:\Users\msys\Desktop\DosProjectV1\Catalog> docker exec -it catalogservice /bin/sh
/app # ls /app
Dockerfile        catalog           catalog_config    db.sqlite3        manage.py         requirements.txt
/app # sqlite3 db.sqlite3
/bin/sh: sqlite3: not found
/app # apk add --no-cache sqlite
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/community/x86_64/APKINDEX.tar.gz
(1/1) Installing sqlite (3.45.3-r1)
Executing busybox-1.36.1-r29.trigger
OK: 17 MiB in 37 packages
/app # sqlite3 db.sqlite3
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .tables
auth_group                  catalog_book
auth_group_permissions      catalog_catalog
auth_permission             django_admin_log
auth_user                   django_content_type
auth_user_groups            django_migrations
auth_user_user_permissions  django_session
sqlite> .schema catalog_book
CREATE TABLE IF NOT EXISTS "catalog_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "cost" integer NULL, "
") DEFERRABLE INITIALLY DEFERRED);
CREATE INDEX "catalog_book_catalog_id_id_1279b03e" ON "catalog_book" ("catalog_id_id");
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|4|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|3|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite>
```

PUT http://localhost:8001/FrontEnd_service/purchase/2

200 OK · 45 ms · 361 B

```
{
    "message": "Purchased book 2",
    "new_count": 3
}
```

Also after this update request cost is updated to 20 it was 35 for book id, we can see this in the terminal.



```
/app # sqlite3 db.sqlite3
/bin/sh: sqlite3: not found
/app # apk add --no-cache sqlite
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/community/x86_64/APKINDEX.tar.gz
(1/1) Installing sqlite (3.45.3-r1)
Executing busybox-1.36.1-r29.trigger
OK: 17 MiB in 37 packages
/app # sqlite3 db.sqlite3
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .tables
auth_group                  catalog_book
auth_group_permissions      catalog_catalog
auth_permission             django_admin_log
auth_user                   django_content_type
auth_user_groups            django_migrations
auth_user_user_permissions  django_session
sqlite> .schema catalog_book
CREATE TABLE IF NOT EXISTS "catalog_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "cost" integer NULL, "
") DEFERRABLE INITIALLY DEFERRED);
CREATE INDEX "catalog_book_catalog_id_id_1279b03e" ON "catalog_book" ("catalog_id_id");
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|4|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|35|Distributed Systems|3|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|20|Distributed Systems|3|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite>
```

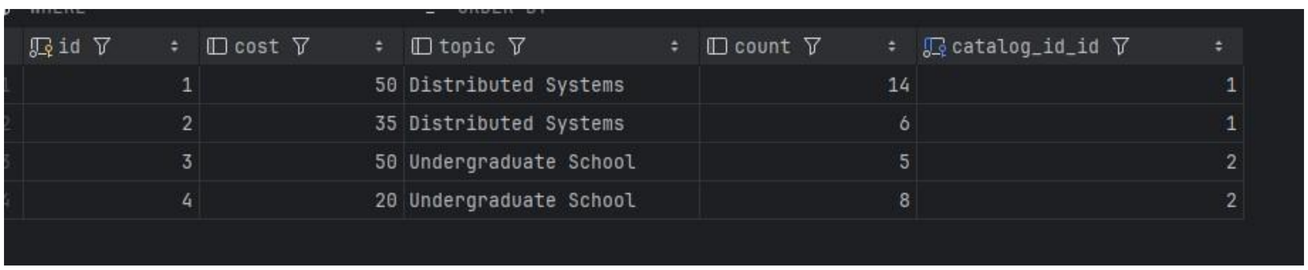
PUT http://localhost:8000/catalog_service/update/2

```
{
    "cost":20
}
```

200 OK · 28 ms · 390 B

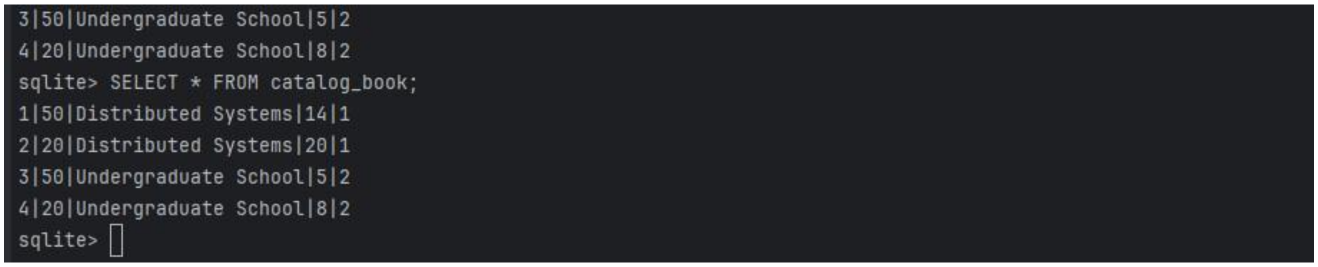
```
{
    "id": 2,
    "cost": 20,
    "topic": "Distributed Systems",
    "count": 3,
    "catalog_id": 1
}
```

Here is the db schema in the local host machine.



id	cost	topic	count	catalog_id_id
1	50	Distributed Systems	14	1
2	35	Distributed Systems	6	1
3	50	Undergraduate School	5	2
4	20	Undergraduate School	8	2

db schema in the container.



```
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> SELECT * FROM catalog_book;
1|50|Distributed Systems|14|1
2|20|Distributed Systems|20|1
3|50|Undergraduate School|5|2
4|20|Undergraduate School|8|2
sqlite> 
```