CSI3007 - ADVANCED PYTHON PROGRAMMING

LAB ACTIVITY – 23

E-Commerce Discount and Vendor Info API using FastAPI and Docker

NAME : SONALI R

REG NO : 22MID0061

Goal: This project implements a FastAPI-based RESTful service that provides discount and vendor details for a given product in an e-commerce environment. The API accepts a product's details (ID, name, price) via a POST request, processes them, and returns a dynamic discount offer and vendor name.

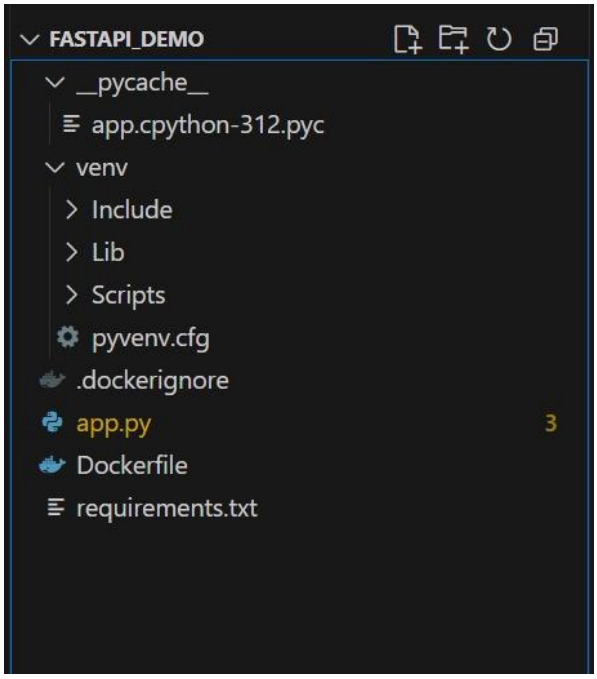
The application is containerized using Docker for portability and uses AWS Lambda compatibility via Mangum to enable serverless deployment.

Reasons to choose FastAPI:

It is one of the fastest Python web frameworks. It Simplifies handling JSON requests and responses using Pydantic models. Uses async/await for non-blocking I/O, improving concurrency.

Project Overview

Project Structure:



Application Code - app.py

```
from fastapi import FastAPI, HTTPException, Request
from fastapi.middleware.cors import CORSMiddleware
from mangum import Mangum
import random

app = FastAPI(title="E-Commerce API", version="1.0")

# Enable CORS for frontend integration
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

# --- API Endpoint ---
@app.post("/discount_vendor_info")
async def add_to_cart(request: Request):
    try:
        data = await request.json()
        print(" Received data:", data)

        product_id = data.get("product_id")
        name = data.get("name")
        price = data.get("price")

        if not all([product_id, name, price]):
            raise HTTPException(status_code=400, detail="Missing product details")
```

```
        # Generate random discount and vendor
        discount = random.choice([0, 5, 10, 15, 20])
        vendor = random.choice(["TechWorld", "ShopEase", "ElectroMart", "MegaDeals"])

        if discount > 0:
            message = f"{discount}% discount available from {vendor} for {name}."
        else:
            message = f"No current discounts for {name}."

        return {"message": message}

    except Exception as e:
        raise HTTPException(status_code=500, detail=f"Processing failed: {str(e)}")

# --- AWS Lambda handler ---
lambda_handler = Mangum(app)
```

Docker Configuration

Dockerfile

```
FROM public.ecr.aws/lambda/python:3.11

WORKDIR /var/task

# Install system dependencies
RUN yum install -y gcc make rust cargo && \
    python3.11 -m pip install --upgrade pip setuptools wheel

# Install dependencies
COPY requirements.txt .
```

```
RUN python3.11 -m pip install --no-cache-dir -r requirements.txt

# Copy source code
COPY . .

# Set entry point for AWS Lambda
CMD ["ECommerce_API.lambda_handler"]
```

.dockerignore

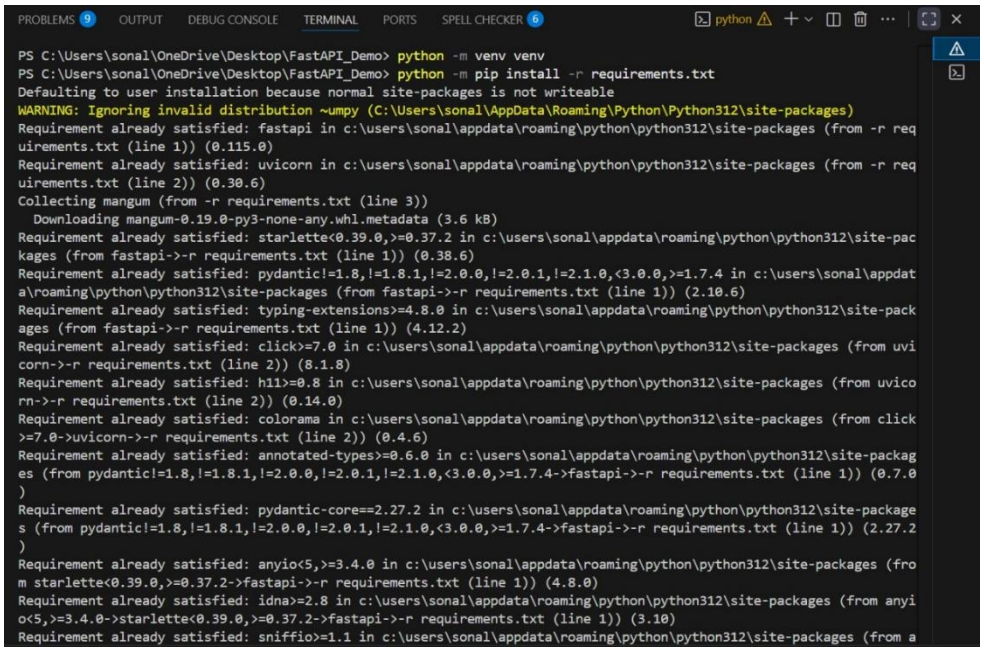
To reduce image size and avoid unnecessary files being copied into the Docker image:

```
# Ignore virtual environment
venv/
.env/
ENV/
.venv/

# Ignore Python cache and compiled files
__pycache__/
```

- venv/ — excludes local virtual environment files.
- __pycache__/ — removes unnecessary compiled Python bytecode.
- Keeps Docker image lightweight and clean.

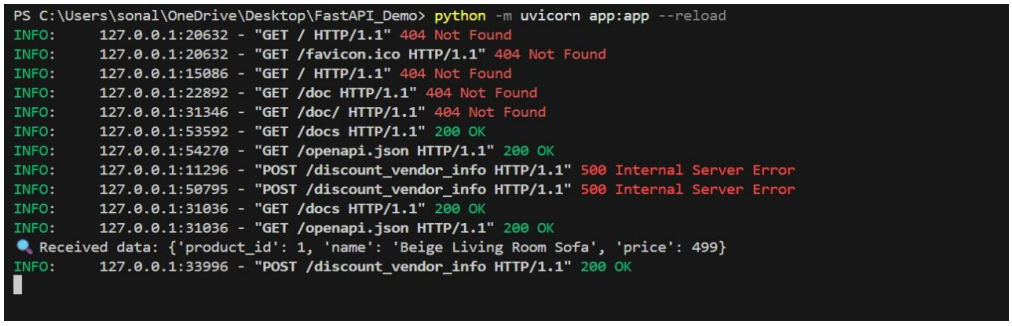
Installing Dependencies: Before building the image, install dependencies locally to test

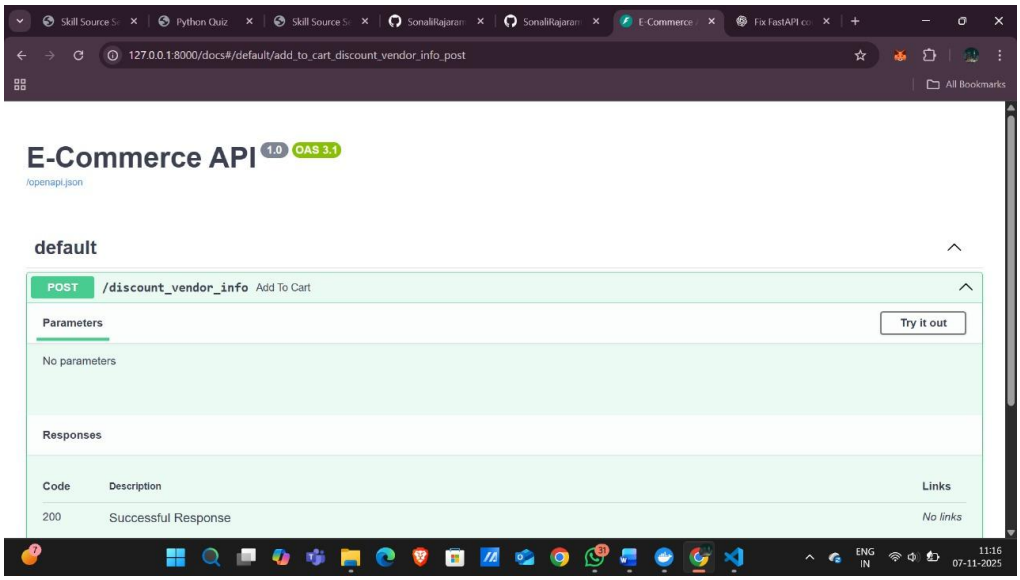
- python -m pip install -r requirements.txt

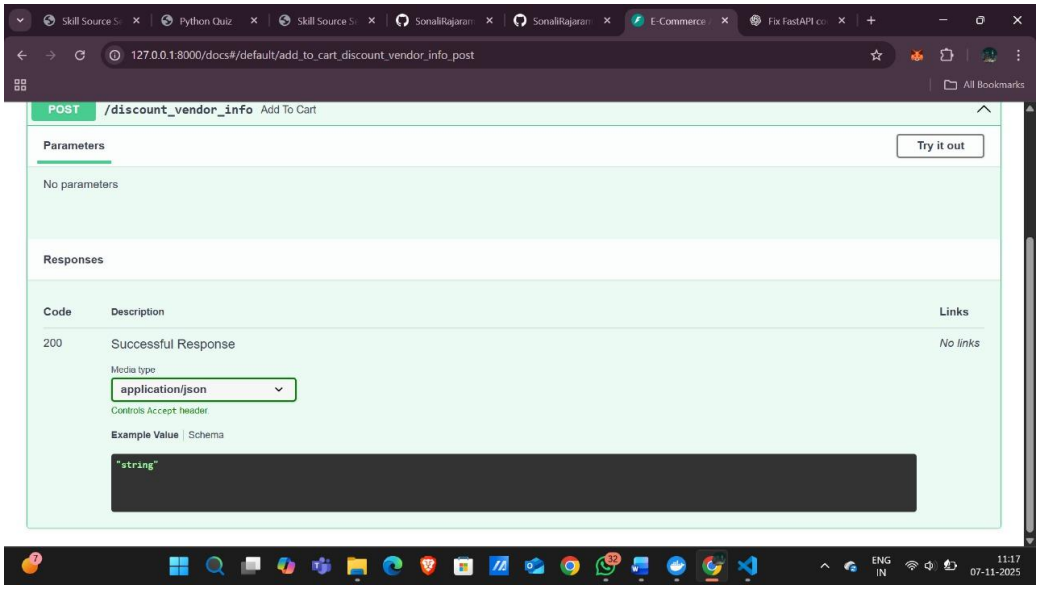


Running the FastAPI App Locally: Run the app on your local system

- python -m uvicorn app:app --reload







Testing with PowerShell (Invoke-RestMethod)

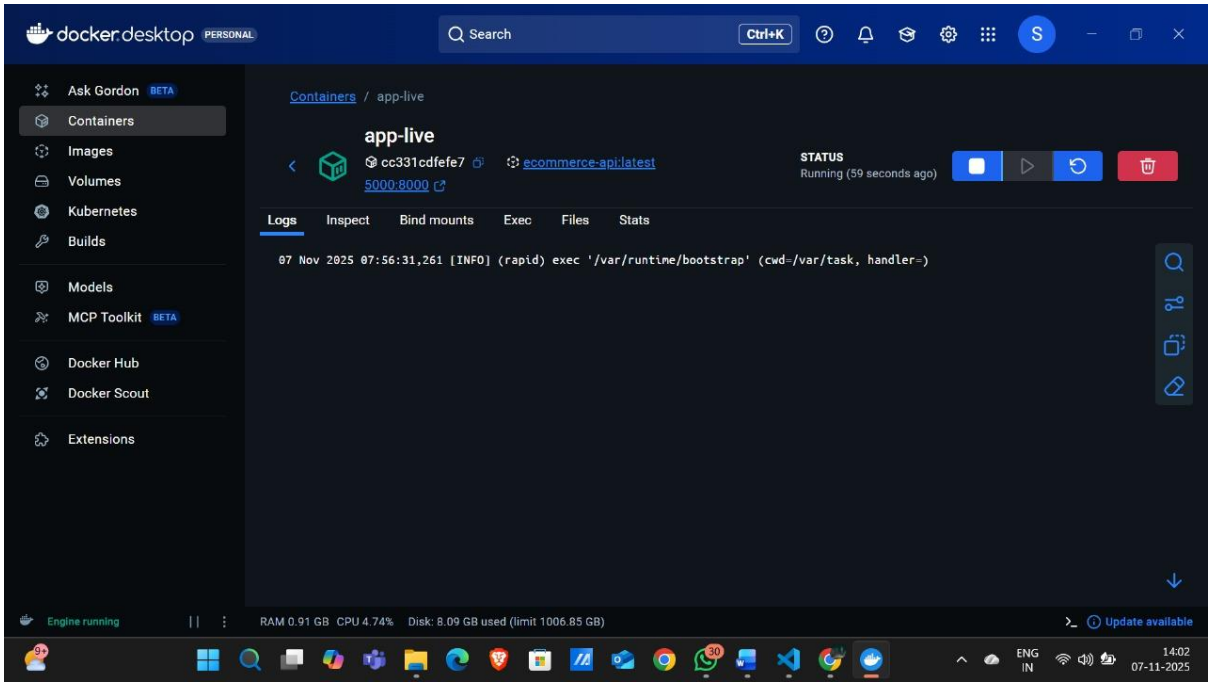
- Invoke-RestMethod -Uri "http://127.0.0.1:8000/discount_vendor_info" -Method POST -Headers @{ "Content-Type" = "application/json" } -Body '{"product_id": 1, "name": "Beige Living Room Sofa", "price": 499}'

```
PS C:\Users\sonal\OneDrive\Desktop\FastAPI_Demo> Invoke-RestMethod -Uri "http://127.0.0.1:8000/discount_vendor_info"
 -Method POST -Headers @{ "Content-Type" = "application/json" } -Body '{"product_id": 1, "name": "Beige Living Room
Sofa", "price": 499}'

message
-------
15% discount available from MegaDeals for Beige Living Room Sofa.


PS C:\Users\sonal\OneDrive\Desktop\FastAPI_Demo>
```

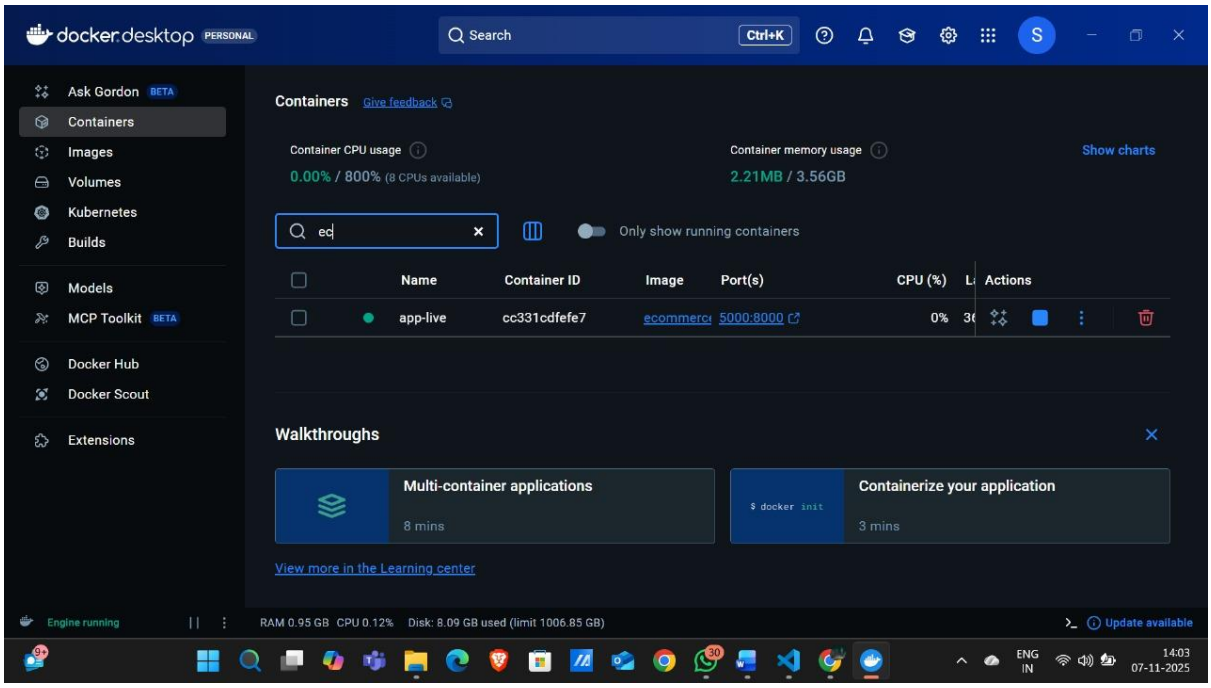
Building and Running Docker Image

```
PS C:\Users\sonal\OneDrive\Desktop\FastAPI_Demo> docker build --network=host -t ecommerce-api .
[+] Building 968.5s (11/11) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                  0.1s
 => => transferring dockerfile: 469B                                                                                  0.0s
 => [internal] load metadata for public.ecr.aws/lambda/python:3.11                                                    4.5s
 => [internal] load .dockerignore                                                                                     0.1s
 => => transferring context: 154B                                                                                     0.0s
 => [internal] load build context                                                                                     0.0s
 => => transferring context: 125B                                                                                     0.0s
 => [1/6] FROM public.ecr.aws/lambda/python:3.11@sha256:462443cde0453d1bb1aa8e3f50923fd26a8f458d3aa506eeedb11ea037b  0.1s
 => => resolve public.ecr.aws/lambda/python:3.11@sha256:462443cde0453d1bb1aa8e3f50923fd26a8f458d3aa506eeedb11ea037b  0.1s
 => CACHED [2/6] WORKDIR /var/task                                                                                    0.0s
 => [3/6] RUN yum install -y gcc make rust cargo &&     python3.11 -m pip install --upgrade pip setuptools wheel    723.2s
 => [4/6] COPY requirements.txt .                                                                                     0.2s
 => [5/6] RUN python3.11 -m pip install --no-cache-dir -r requirements.txt                                          121.9s
 => [6/6] COPY . .                                                                                                    0.5s
 => exporting to image                                                                                              117.0s
 => => exporting layers                                                                                              95.4s
 => => exporting manifest sha256:bac7f2727369dba525527d9cec91574a19ab422333af9286f8f9b8aad9f420d8                     0.2s
 => => exporting config sha256:3f205e9a07cc4b248b571cfe8d87b511970936bedb54c2f7fc2344855fa7083a                       0.1s
 => => exporting attestation manifest sha256:86b267812c274015bfb4e6c3ad3dc135e28e2bdb793389831238cd23317397f7         0.1s
 => => exporting manifest list sha256:e6938f3550a15e32112ce1df087d27d6e78f0d22e4efa7b3c07219f402e8631c                0.1s
 => => naming to docker.io/library/ecommerce-api:latest                                                               0.1s
 => => unpacking to docker.io/library/ecommerce-api:latest                                                           20.7s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/rrolio35a2bwto5db4xonsx4p
PS C:\Users\sonal\OneDrive\Desktop\FastAPI_Demo>  docker run -d -p 5000:8000 --name app-live ecommerce-api
cc331cdfefe73f45fee1250238f2d827a1e05634240a005ed1de7bcb56b3d7fd
PS C:\Users\sonal\OneDrive\Desktop\FastAPI_Demo>
```

Docker Image:



Docker Container:



Troubleshooting:

- Error: Invoke-RestMethod -Uri "http://127.0.0.1:8080/discount_vendor_info" -Method POST -Headers @{ "Content-Type" = "application/json" } -Body '{"product_id": 1, "name": "Beige Living Room Sofa", "price": 499}'
- Resolved: Invoke-RestMethod -Uri "http://127.0.0.1:8000/discount_vendor_info" -Method POST -Headers @{ "Content-Type" = "application/json" } -Body '{"product_id": 1, "name": "Beige Living Room Sofa", "price": 499}'
  (wrong port - 8080)