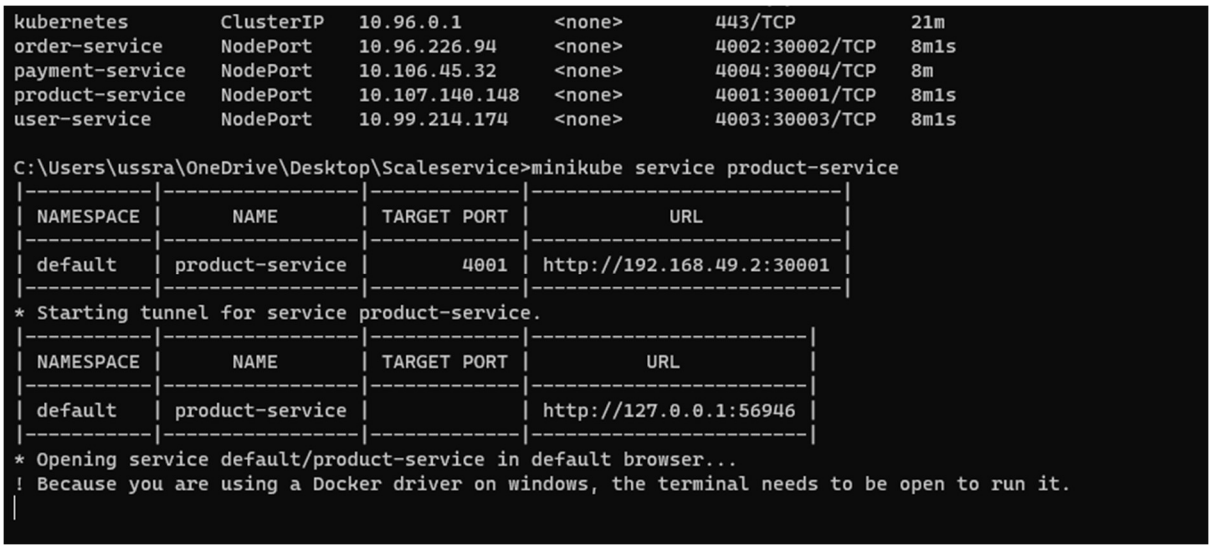
1. **Microservices Development**
    - Four Node.js microservices (Product, Order, User, Payment) are scaffolded with separate folders, models, routes, and app files.
    - Each service uses its own MongoDB Atlas database (Database per Service pattern).
    - REST APIs are implemented for each service.
2. **Dockerization**
    - Each service has its own Dockerfile.
    - A docker-compose.yml is present to orchestrate all services (for local development).
3. **Kubernetes/Minikube Deployment**
    - You have deployment and service YAMLs for each microservice.
    - You have built and pushed images to Docker Hub.
    - You have applied the manifests to Minikube and accessed the services via NodePort/tunnel.
4. **Testing**
    - You have tested endpoints locally and in Minikube.
5. **.gitignore**
    - A .gitignore file is present to exclude sensitive and unnecessary files.



Microservices-Based E-commerce Platform Documentation

1. Application Description

This project is a microservices-based e-commerce platform built with Node.js and MongoDB Atlas. The application is architected as four independent services:

- **Product Service:** Manages product catalog and details.
- **Order Service:** Handles order creation, retrieval, update, and deletion.
- **User Service:** Manages user registration, profiles, and authentication data.
- **Payment Service:** Processes and records payment transactions.

Each service is containerized with Docker, maintains its own database for data isolation, and exposes RESTful APIs for communication. The system can be run locally, orchestrated with Docker Compose, or deployed to Kubernetes using Minikube.

2. System Architecture

2.1 Microservices Architecture Diagram

- **Independence:** Each service is stateless and independently deployable.
- **Data Isolation:** Each service uses the Database per Service pattern with a dedicated MongoDB Atlas database.
- **Communication:** Services interact via REST APIs, with potential for asynchronous messaging in future enhancements.

2.2 Database Design

- **Product DB:** Stores product details.
- **Order DB:** Stores orders, referencing product and user IDs.
- **User DB:** Stores user profiles and credentials.
- **Payment DB:** Stores payment records linked to orders and users.

2.3 Communication Flow

Example: When an order is placed, the Order Service may call Product Service to check inventory and User Service to verify user information.

3. Step-by-Step Execution

3.1 Running Each Microservice Locally

1. **Clone or download the repository.**
2. For each service (e.g., product-service):
    - Open a terminal and navigate to the folder.
    - Install dependencies:

      bash

      npm install

    - Copy .env.example to .env and set your MongoDB Atlas connection string (provided for each service).

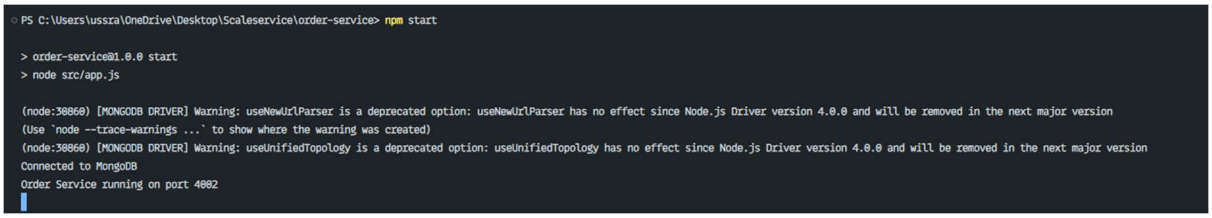
- Start the service:

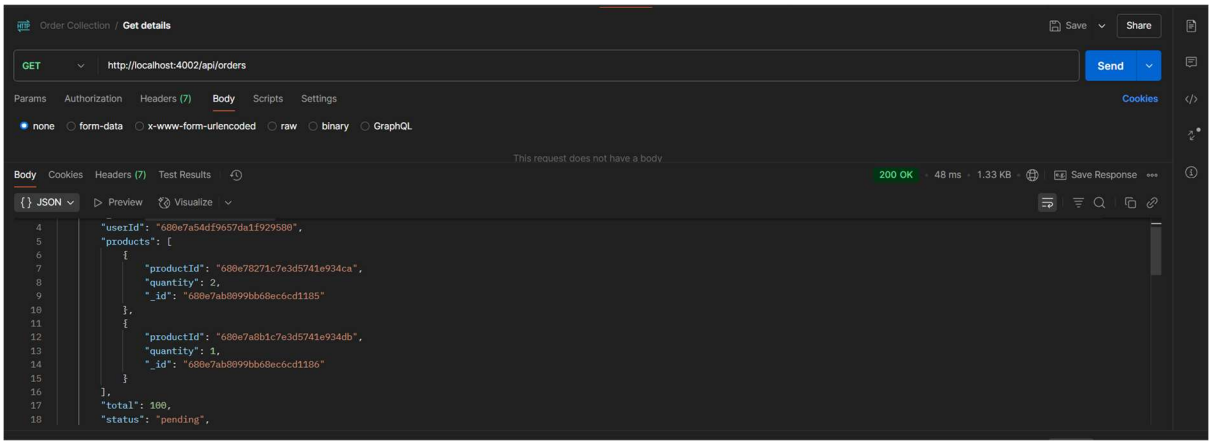
  bash

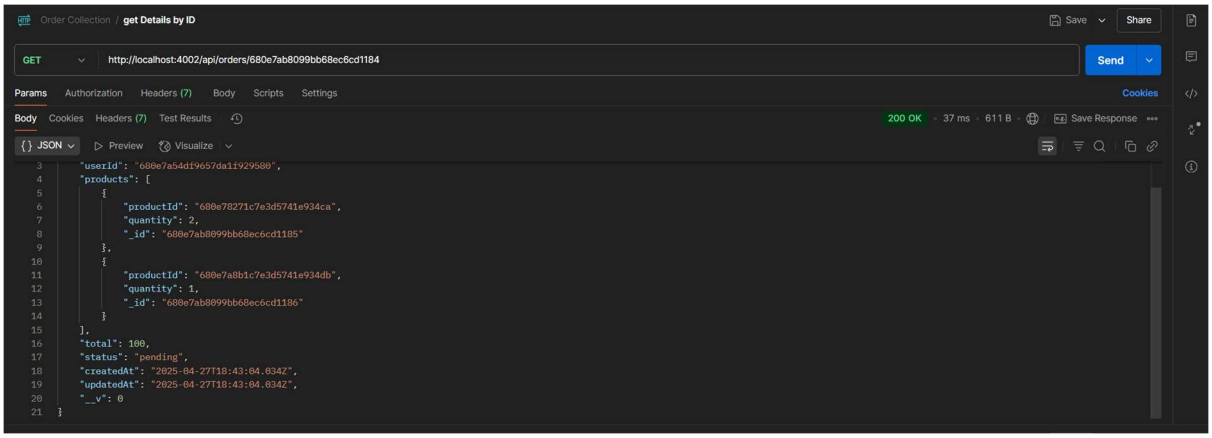
  npm run dev

- The service will run on its designated port (e.g., 4001 for Product Service).

Screenshot:







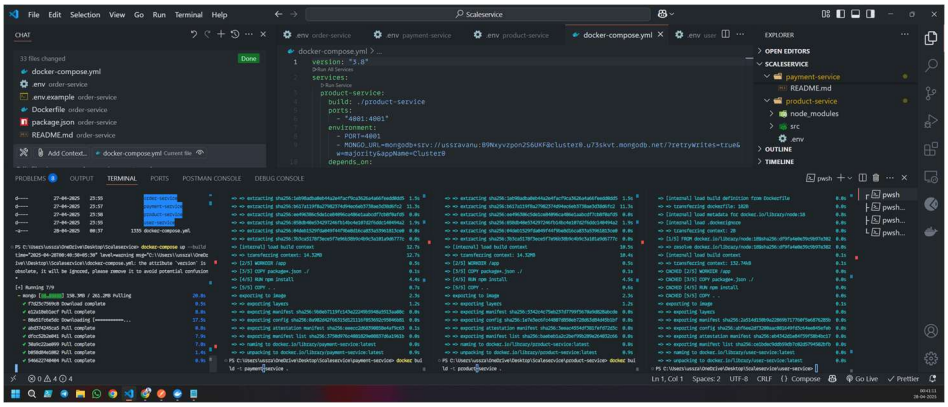
3.2 Running All Services with Docker Compose

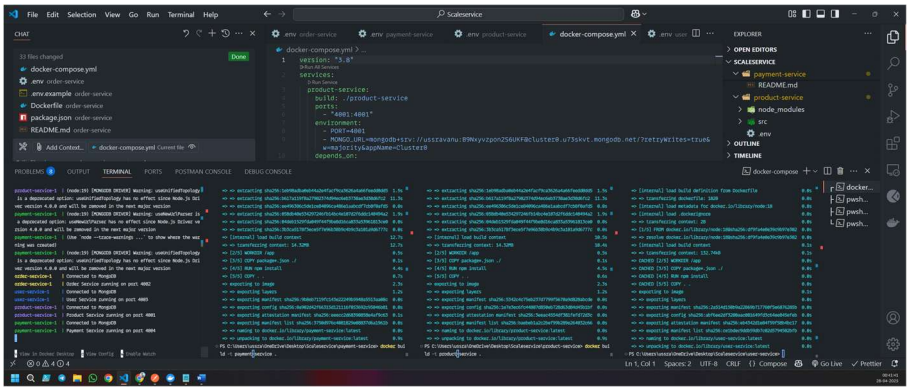
1. Ensure all .env files have the correct MongoDB Atlas URLs.
2. In the project root, run:

   bash

   docker-compose up --build
3. All services will be available on their respective ports (4001–4004).

Screenshot:





3.3 Deploying to Minikube

1. **Start Minikube:**

   bash

   minikube start

2. **Push Docker images to Docker Hub** .

3. **Update deployment YAMLs** to use your Docker Hub images (already completed).

4. **Apply the manifests:**

   bash

   kubectl apply -f product-deployment.yaml

   kubectl apply -f order-deployment.yaml

   kubectl apply -f user-deployment.yaml

   kubectl apply -f payment-deployment.yaml

5. **Check pods and services:**

   bash

   kubectl get pods

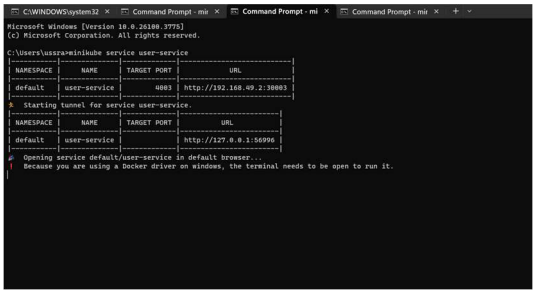
   kubectl get services

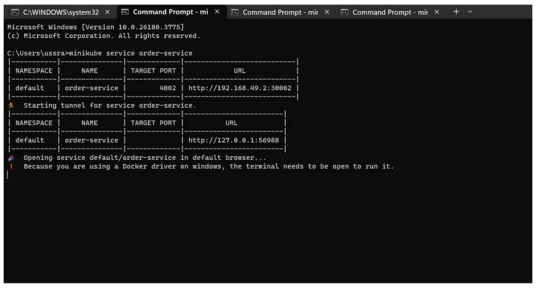
6. **Access a service:**

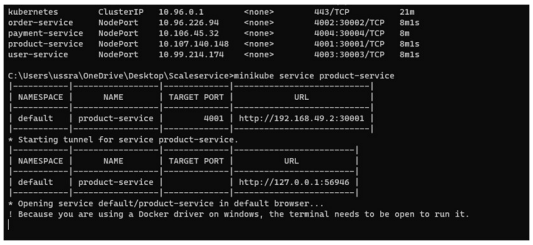
   bash

   minikube service product-service

Screenshot:

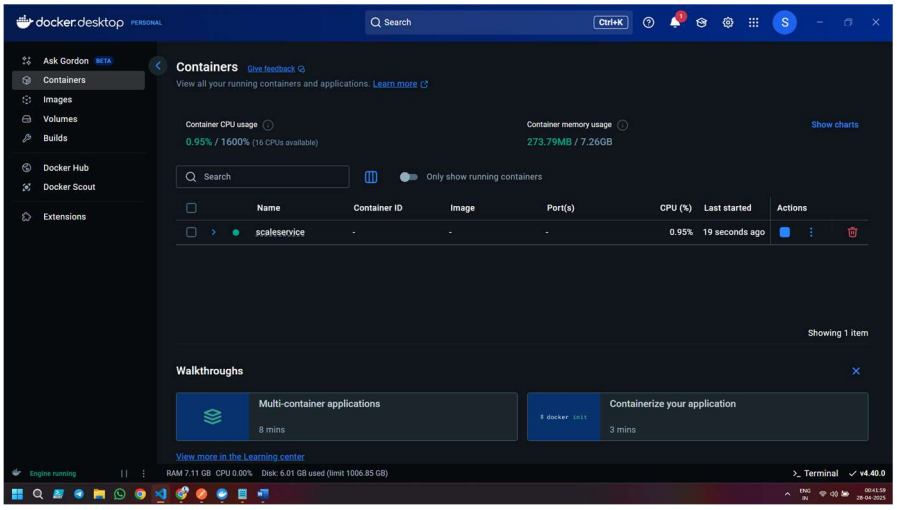




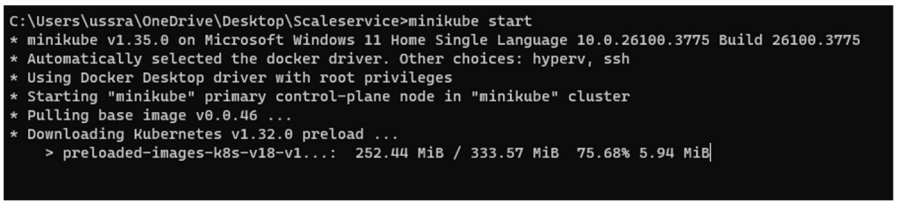


4. Screenshots

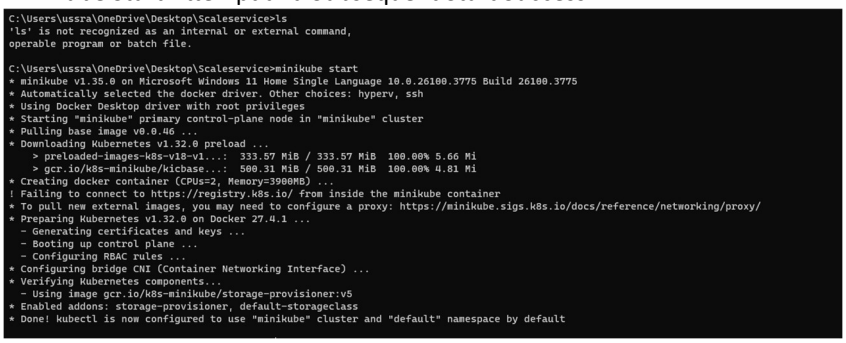
Docker Desktop - Running Containers Overview



Minikube Start Command Output



Minikube Start Attempt and Subsequent Start Success



Minikube Docker Environment Configuration

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>minikube docker-env
SET DOCKER_TLS_VERIFY=1
SET DOCKER_HOST=tcp://127.0.0.1:55732
SET DOCKER_CERT_PATH=C:\Users\ussra\.minikube\certs
SET MINIKUBE_ACTIVE_DOCKERD=minikube
REM To point your shell to minikube's docker-daemon, run:
REM @FOR /f "tokens=*" %i IN ('minikube -p minikube docker-env --shell cmd') DO @%i
```

Executing Minikube Docker Environment Command

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>@FOR /f "tokens=*" %i IN ('minikube -p minikube docker-env --shell cmd') DO @%i

C:\Users\ussra\OneDrive\Desktop\Scaleservice>
```

Docker Build Process for Product Service

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>docker build -t product-service:latest ./product-service
[+] Building 55.0s (11/11) FINISHED                                                                docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 182B                                                                          0.0s
 => [internal] load metadata for docker.io/library/node:18                                                    2.8s
 => [auth] library/node:pull token for registry-1.docker.io                                                   0.0s
 => [internal] load .dockerignore                                                                             0.1s
 => => transferring context: 2B                                                                               0.0s
 => [1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e  43.8s
 => => resolve docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e  0.0s
 => => sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e 6.41kB / 6.41kB               0.0s
 => => sha256:de20d623379fc7c7ccf845a22c3153b920d57446ba7c8e64ba25d21a60b48ad6 6.39kB / 6.39kB               0.0s
 => => sha256:23b7d26ef1d294256da0d70ce374277b9aab5ca683015073316005cb63d33849 48.49MB / 48.49MB             20.8s
 => => sha256:aa6c239d30ee04dede270729f9502389b1a9546687ce656872536340ee0a9e03 2.49kB / 2.49kB               0.0s
 => => sha256:07d1b5af933d2dfc3d0dd509d6e20534825e4a537f7b006a6cb5b8e5a1f20905 24.01MB / 24.01MB              9.4s
 => => sha256:1eb98adba0eb44a2e4facf9ca3626a4a66feedd0dd56d159cca90a35205744e7 64.40MB / 64.40MB             26.8s
 => => sha256:b617a119f8a27982374d94ec6eb3738ae3d38d6fc2c34c865813926cf596a621 211.33MB / 211.33MB           34.4s
 => => sha256:ee496386c5de1ce84096ca486e1aabcdf7cb8f0afd5a9b4863dbf870b340744f 3.32kB / 3.32kB               21.1s
 => => extracting sha256:23b7d26ef1d294256da0d70ce374277b9aab5ca683015073316005cb63d33849                     3.0s
 => => sha256:058db40e534297246fb14bc4e107d2f6ddc140494a26793e1e90696cd6b2507a 45.68MB / 45.68MB             34.7s
 => => extracting sha256:07d1b5af933d2dfc3d0dd509d6e20534825e4a537f7b006a6cb5b8e5a1f20905                     0.8s
 => => extracting sha256:1eb98adba0eb44a2e4facf9ca3626a4a66feedd0dd56d159cca90a35205744e7                     4.1s
 => => sha256:04deb1529fda049f44f9be8d16ca833a53961813ce07eda3cefd50cd3fd74880 1.25MB / 1.25MB               27.6s
 => => sha256:3b3ca5178f3ece5f7e96b38b9c4b9c3a101a9d6777cf0f5c320f869512c80024 448B / 448B                   27.9s
 => => extracting sha256:b617a119f8a27982374d94ec6eb3738ae3d38d6fc2c34c865813926cf596a621                     7.0s
 => => extracting sha256:ee496386c5de1ce84096ca486e1aabcdf7cb8f0afd5a9b4863dbf870b340744f                     0.0s
 => => extracting sha256:058db40e534297246fb14bc4e107d2f6ddc140494a26793e1e90696cd6b2507a                     1.9s
 => => extracting sha256:04deb1529fda049f44f9be8d16ca833a53961813ce07eda3cefd50cd3fd74880                     0.0s
 => => extracting sha256:3b3ca5178f3ece5f7e96b38b9c4b9c3a101a9d6777cf0f5c320f869512c80024                     0.0s
 => [internal] load build context                                                                             0.6s
 => => transferring context: 14.32MB                                                                          0.6s
 => [2/5] WORKDIR /app                                                                                        0.2s
 => [3/5] COPY package*.json ./                                                                               0.0s
 => [4/5] RUN npm install                                                                                     2.1s
 => [5/5] COPY . .                                                                                            0.6s
 => exporting to image                                                                                        0.2s
 => => exporting layers                                                                                       0.2s
 => => writing image sha256:32c944088918a446c8cae3588d7460d746bcebab82c911ab2bbf7b865162c983                  0.0s
 => => naming to docker.io/library/product-service:latest                                                     0.0s
```

Docker Build Process for Order Service

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>docker build -t order-service:latest ./order-service
[+] Building 4.3s (10/10) FINISHED                                                                 docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 182B                                                                          0.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e  0.0s
 => [internal] load build context                                                                             0.6s
 => => transferring context: 14.32MB                                                                          0.6s
 => CACHED [2/5] WORKDIR /app                                                                                 0.0s
 => [3/5] COPY package*.json ./                                                                               0.1s
 => [4/5] RUN npm install                                                                                     2.3s
 => [5/5] COPY . .                                                                                            0.3s
 => exporting to image                                                                                        0.2s
 => => exporting layers                                                                                       0.2s
 => => writing image sha256:f1120586cf0d047fe6f518626dfdf3f3045b7b2e0ed3f22925013fb55028bbc8                  0.0s
 => => naming to docker.io/library/order-service:latest                                                       0.0s
```

Docker Build Process for User Service

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>docker build -t user-service:latest ./user-service
[+] Building 4.2s (10/10) FINISHED                                                                 docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 182B                                                                          0.0s
 => [internal] load metadata for docker.io/library/node:18                                                    0.9s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e  0.0s
 => [internal] load build context                                                                             0.6s
 => => transferring context: 14.32MB                                                                          0.5s
 => CACHED [2/5] WORKDIR /app                                                                                 0.0s
 => [3/5] COPY package*.json ./                                                                               0.1s
 => [4/5] RUN npm install                                                                                     2.3s
 => [5/5] COPY . .                                                                                            0.4s
 => exporting to image                                                                                        0.2s
 => => exporting layers                                                                                       0.2s
 => => writing image sha256:ebc2cb1a119ce0d58c9ce1fe670772329e650325c7932667e4035bdcf3b96153                  0.0s
 => => naming to docker.io/library/user-service:latest                                                        0.0s

C:\Users\ussra\OneDrive\Desktop\Scaleservice>
```

Docker Build Process for User Service

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>docker build -t payment-service:latest ./payment-service
[+] Building 4.4s (10/10) FINISHED                                                                 docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 182B                                                                          0.0s
 => [internal] load metadata for docker.io/library/node:18                                                    0.8s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e  0.0s
 => [internal] load build context                                                                             0.5s
 => => transferring context: 14.32MB                                                                          0.5s
 => CACHED [2/5] WORKDIR /app                                                                                 0.0s
 => [3/5] COPY package*.json ./                                                                               0.0s
 => [4/5] RUN npm install                                                                                     2.0s
 => [5/5] COPY . .                                                                                            0.3s
 => exporting to image                                                                                        0.2s
 => => exporting layers                                                                                       0.2s
 => => writing image sha256:b709c1cbbd8637165eaa3005feaf99db0f97e3951362bc40ed674646285b005f                  0.0s
 => => naming to docker.io/library/payment-service:latest                                                     0.0s
```

Kubectl Apply Commands for Deployments and Services

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl apply -f product-deployment.yaml
deployment.apps/product-service created
service/product-service created

C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl apply -f order-deployment.yaml
deployment.apps/order-service created
service/order-service created

C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl apply -f user-deployment.yaml
deployment.apps/user-service created
service/user-service created

C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl apply -f payment-deployment.yaml
deployment.apps/payment-service created
service/payment-service created

C:\Users\ussra\OneDrive\Desktop\Scaleservice>
```

Kubectl Get Pods and Get Services Output - ImagePullBackOff Errors

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl get pods
NAME                               READY   STATUS             RESTARTS   AGE
order-service-75ff6f6796-pp2bw     0/1     ImagePullBackOff   0          2m31s
payment-service-5f68c7f56b-vdlcw   0/1     ImagePullBackOff   0          2m24s
product-service-6b6fbf4fcc-nqxh2   0/1     ImagePullBackOff   0          2m41s
user-service-977d49c78-pqnqz       0/1     ImagePullBackOff   0          2m27s

C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl get services
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP          7m19s
order-service     NodePort    10.105.253.58   <none>        4002:30002/TCP   2m37s
payment-service   NodePort    10.102.53.185   <none>        4004:30004/TCP   2m30s
product-service   NodePort    10.107.112.79   <none>        4001:30001/TCP   2m47s
user-service      NodePort    10.99.171.223   <none>        4003:30003/TCP   2m33s
```

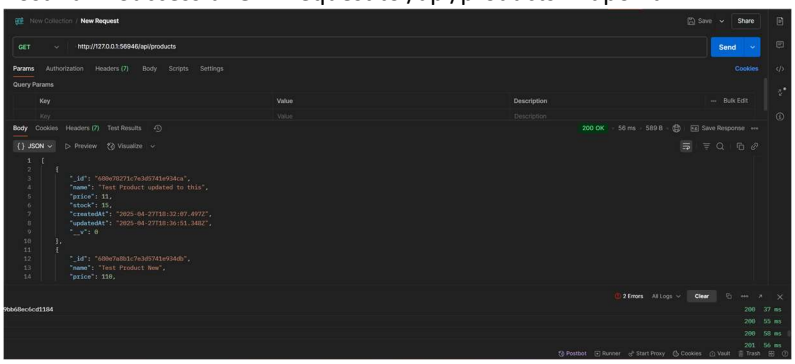
Kubectl Get Pods Output - All Pods Running

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
order-service-54746767d9-8p2ps     1/1     Running   0          34s
payment-service-864f994d4c-75dpv   1/1     Running   0          33s
product-service-cbf8c549b-8ghjr    1/1     Running   0          35s
user-service-5669c97fc-58lpr       1/1     Running   0          34s
```

Kubectl Get Pods and Get Services Output - Running Pods and Service Details

```
C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
order-service-54746767d9-8p2ps     1/1     Running   0          34s
payment-service-864f994d4c-75dpv   1/1     Running   0          33s
product-service-cbf8c549b-8ghjr    1/1     Running   0          35s
user-service-5669c97fc-58lpr       1/1     Running   0          34s

C:\Users\ussra\OneDrive\Desktop\Scaleservice>kubectl get services
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes        ClusterIP   10.96.0.1        <none>        443/TCP          21m
order-service     NodePort    10.96.226.94     <none>        4002:30002/TCP   8m1s
payment-service   NodePort    10.106.45.32     <none>        4004:30004/TCP   8m
product-service   NodePort    10.107.140.148   <none>        4001:30001/TCP   8m1s
user-service      NodePort    10.99.214.174    <none>        4003:30003/TCP   8m1s
```

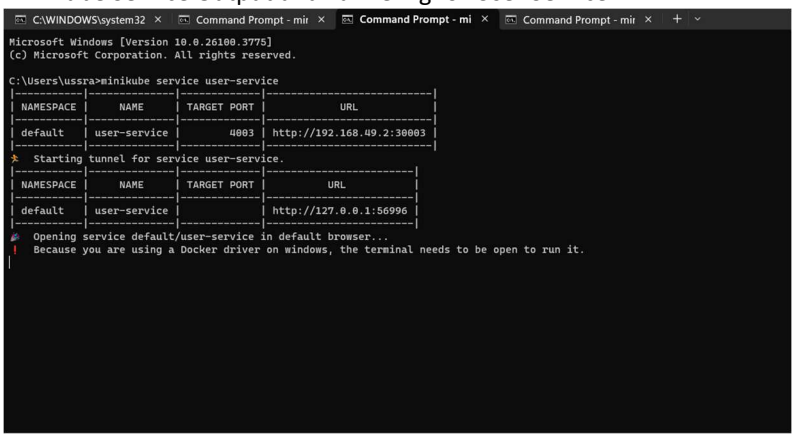
Postman - Successful GET Request to /api/products Endpoint



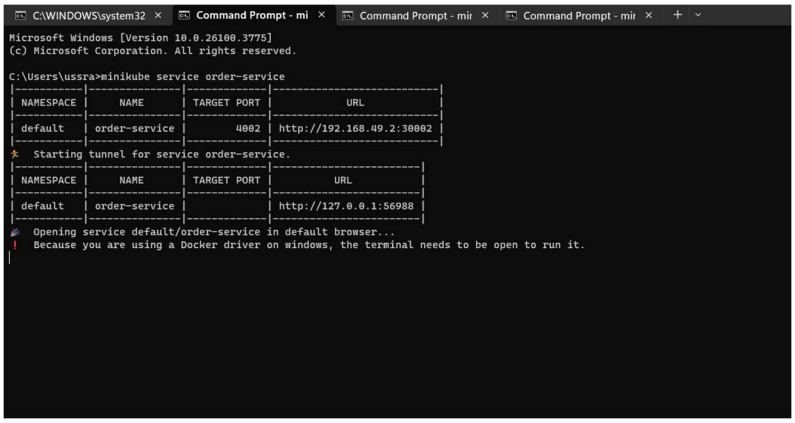
Minikube Service Output and Tunneling for Payment Service

```
Microsoft Windows [Version 10.0.26100.3775]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ussra>http://127.0.0.1:56996/
'http:' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\ussra>minikube service payment-service
|-----------|-----------------|-------------|---------------------------|
| NAMESPACE |      NAME       | TARGET PORT |            URL            |
|-----------|-----------------|-------------|---------------------------|
| default   | payment-service |        4004 | http://192.168.49.2:30004 |
|-----------|-----------------|-------------|---------------------------|
*  Starting tunnel for service payment-service.
|-----------|-----------------|-------------|------------------------|
| NAMESPACE |      NAME       | TARGET PORT |          URL           |
|-----------|-----------------|-------------|------------------------|
| default   | payment-service |             | http://127.0.0.1:57003 |
|-----------|-----------------|-------------|------------------------|
*  Opening service default/payment-service in default browser...
!  Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

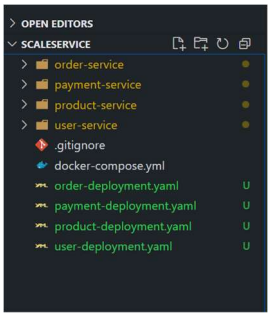
Minikube Service Output and Tunneling for User Service



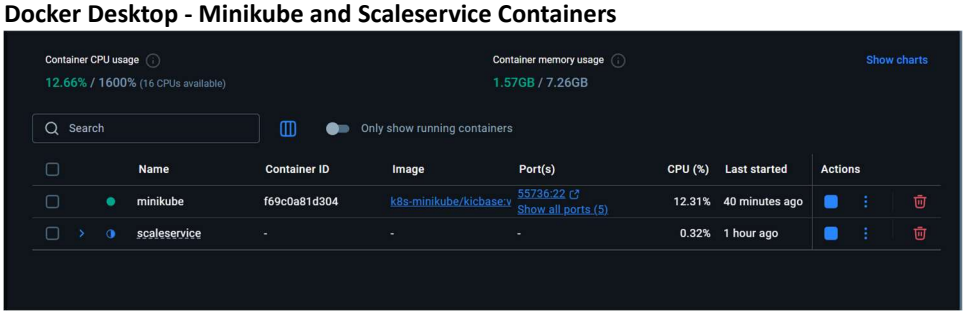
Minikube Service Output and Tunneling for Order Service



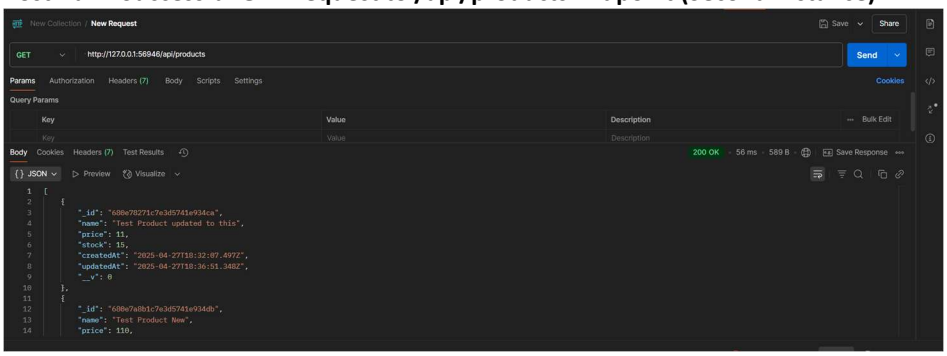
Folder Structure



Docker Desktop - Minikube and Scaleservice Containers



Postman - Successful GET Request to /api/products Endpoint (Second Instance)



Example API Payloads:

- **Order Service:**

```
{
 "userId": "USER_ID_HERE",
 "products": [
  { "productId": "PRODUCT_ID_1", "quantity": 2 }
 ],
 "total": 100,
 "status": "pending"
}
```

- **User Service:**

```
{
 "name": "John Doe",
 "email": "john@example.com",
 "password": "securepassword",
 "address": "123 Main St"
}
```

- **Payment Service:**

```
{
 "orderId": "ORDER_ID_HERE",
 "userId": "USER_ID_HERE",
 "amount": 100,
 "status": "pending",
 "method": "credit_card"
}
```