Osama Salah 12042036

Alaa Hamadneh 11927150

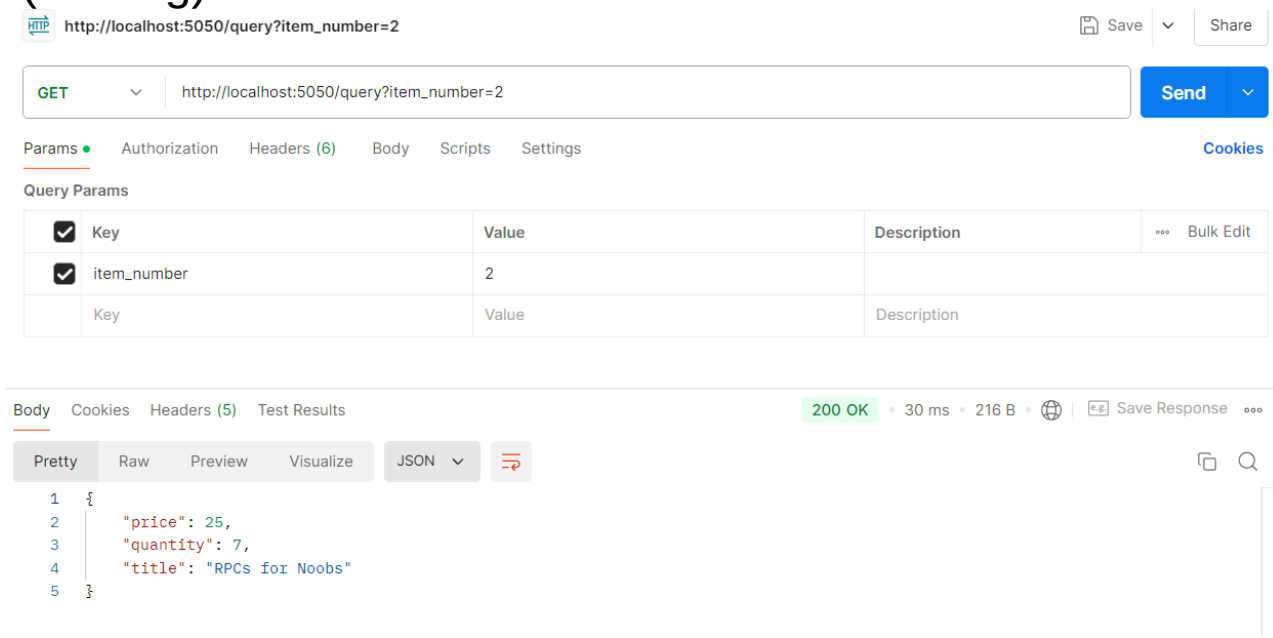
DOS Project Report Part 2

Part1: Cache Consistency

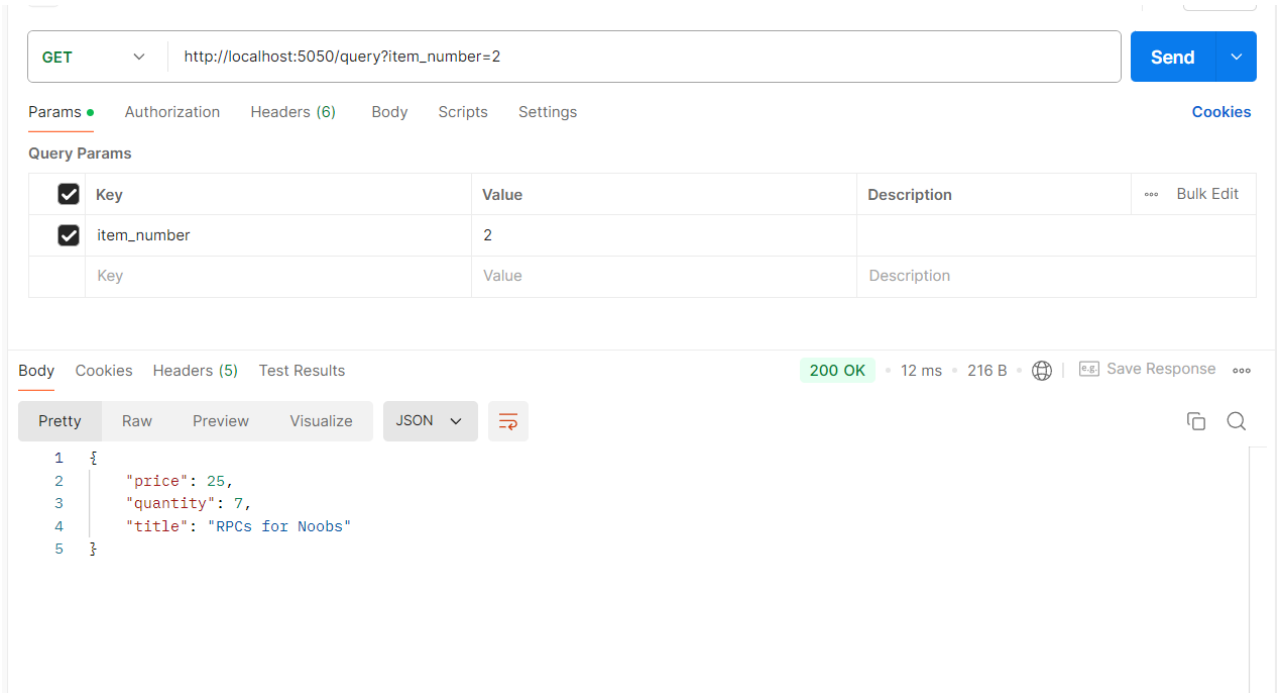
```python
class LRUCache:  1 usage
    def __init__(self, capacity=5):
        self.cache = OrderedDict()  # OrderedDict to maintain insertion order
        self.capacity = capacity  # Max number of items the cache can hold

    def get(self, key):  1 usage
        """Get item from cache."""
        if key in self.cache:
            self.cache.move_to_end(key)  # Move the accessed item to the end (most recently used)
            return self.cache[key]
        return None  # Return None if the key is not in cache

    def put(self, key, value):  1 usage
        """Put item in cache."""
        if key in self.cache:
            self.cache.move_to_end(key)  # Move to end if already in cache
        self.cache[key] = value
        if len(self.cache) > self.capacity:  # If cache exceeds capacity, remove the oldest item
            self.cache.popitem(last=False)

    def invalidate(self, key):  1 usage
        """Invalidate (remove) an item from cache."""
        if key in self.cache:
            del self.cache[key]
# Initialize cache with a capacity of 10 items
cache = LRUCache(capacity=10)
# Load balancing state for catalog and order servers
catalog_replicas = ["http://localhost:5050", "http://localhost:5052"]
order_replicas = ["http://localhost:5061", "http://localhost:5062"]
catalog_index = 0
order_index = 0
```

- When i send `GET` request the **first time**, **<u>before Caching the data</u>**:

(catalog)



now from the cache:

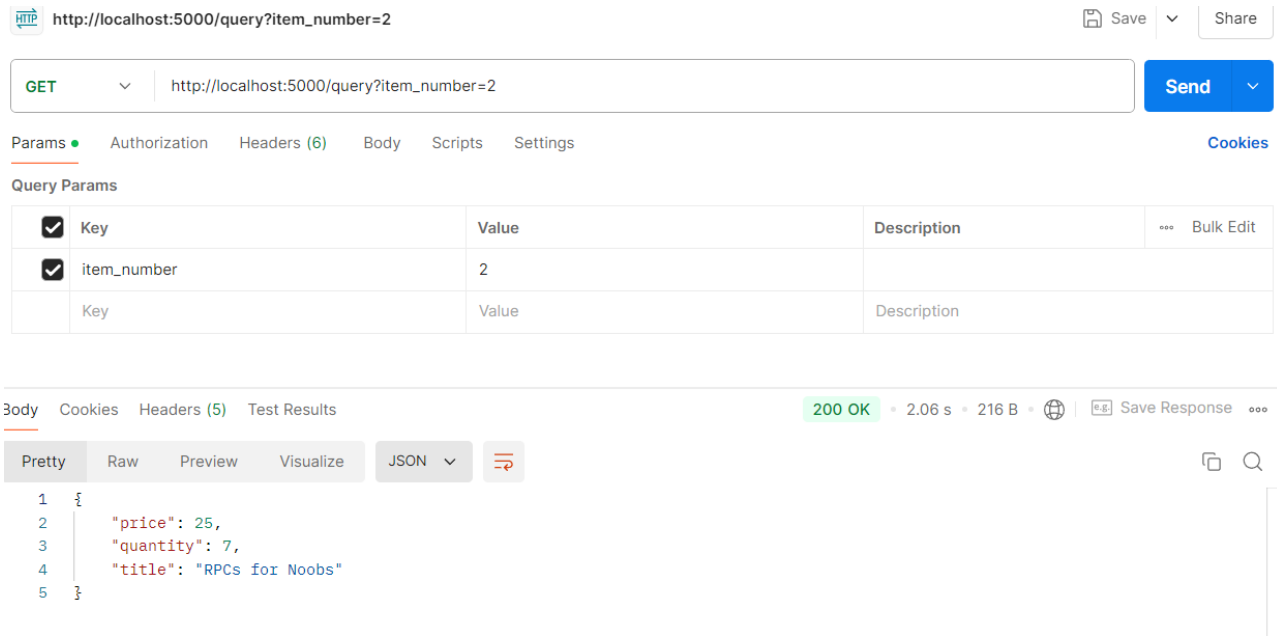


Q1) Compute the average response time (query/buy) of your new systems. What is the response time with and without caching?

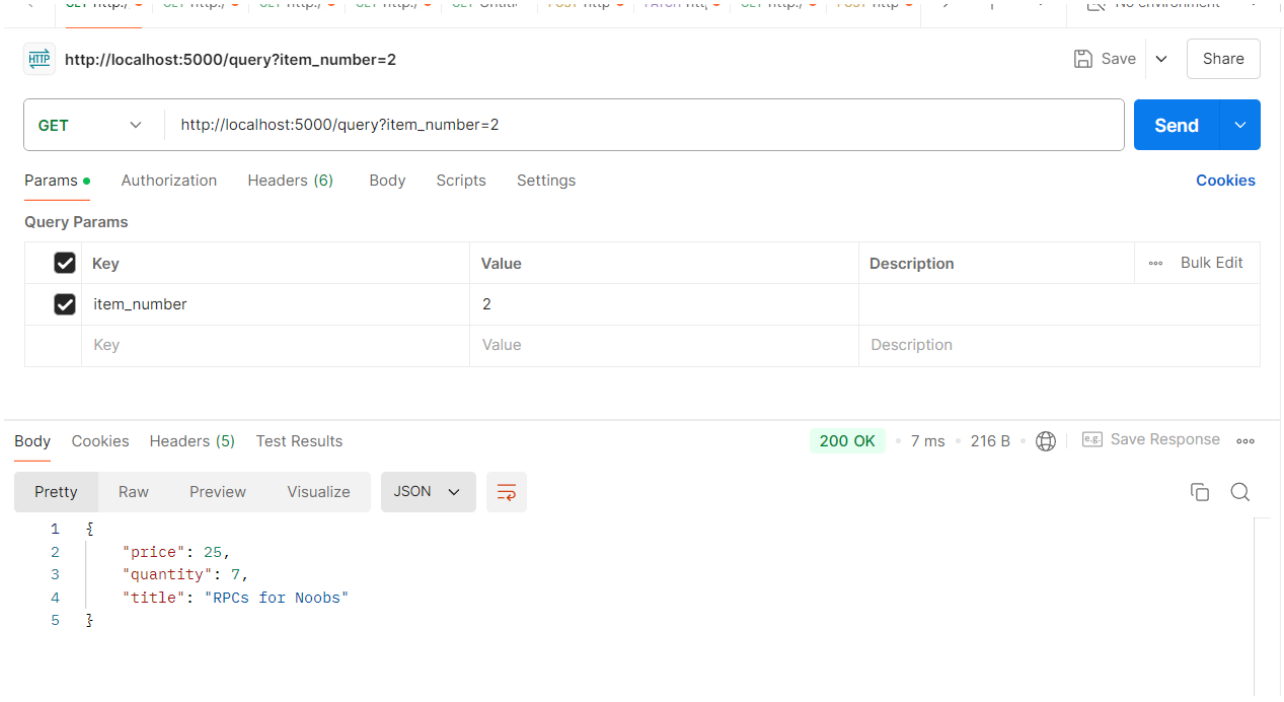
- Answers
  - for info: **30ms**
  - for search: **12ms**

(gateway)

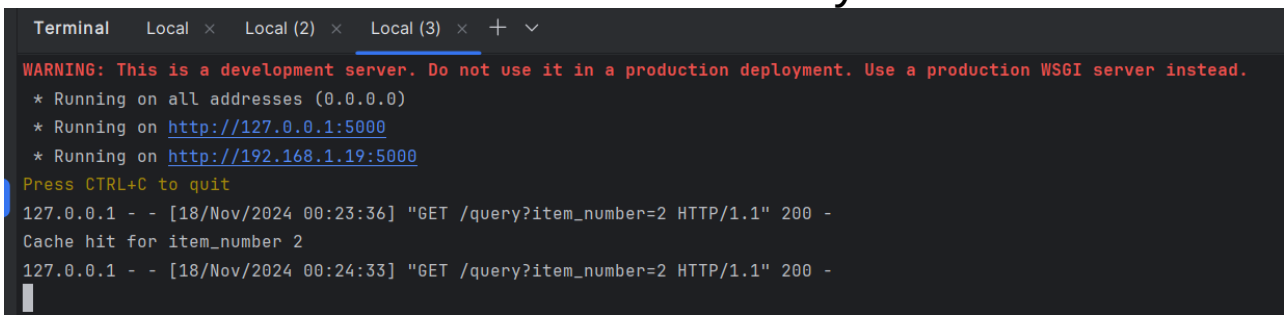
- Without Cache:

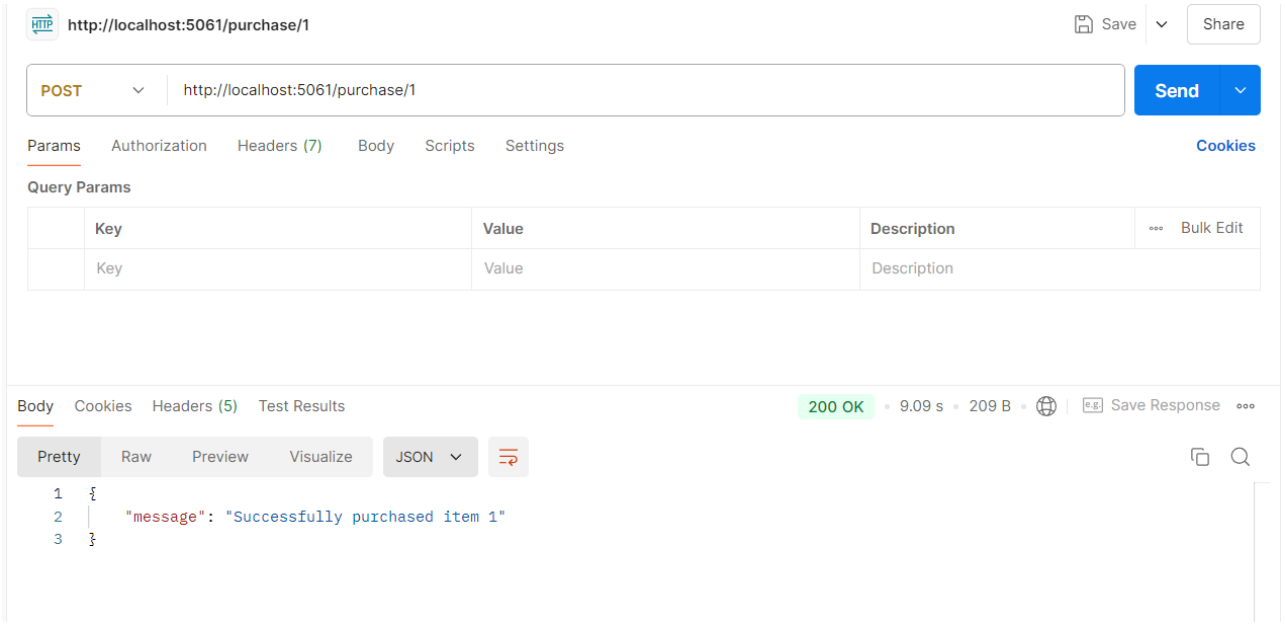


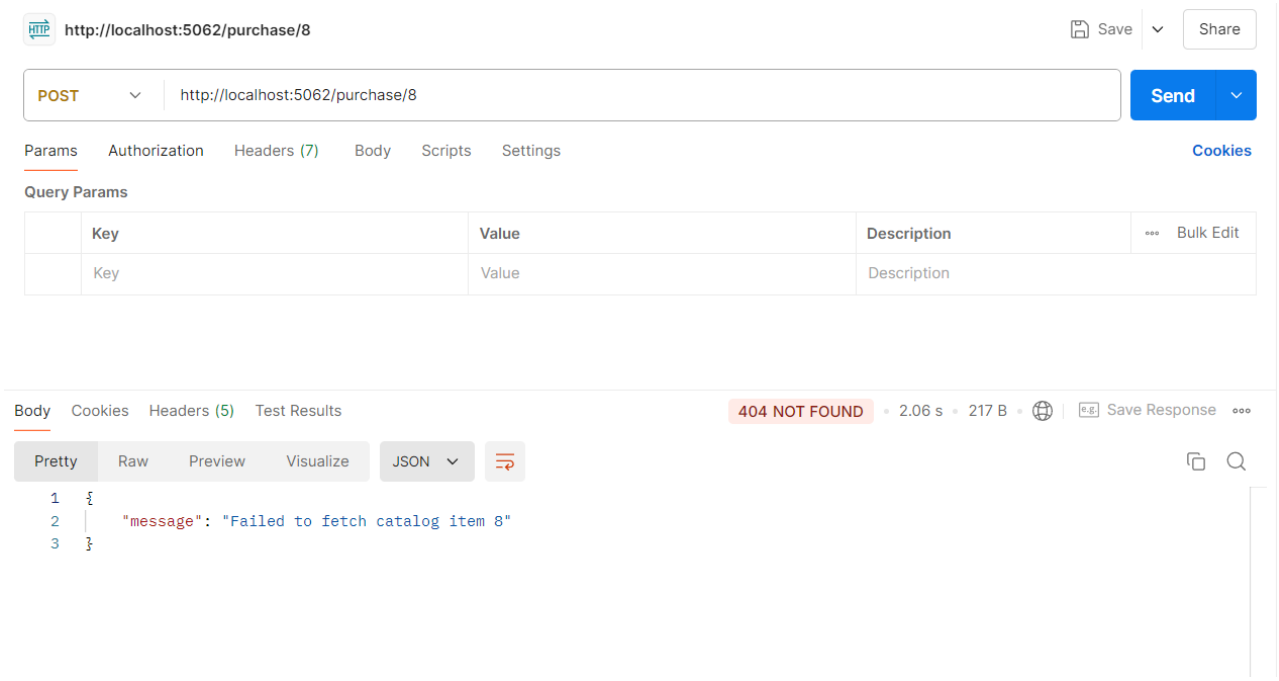
With cache:



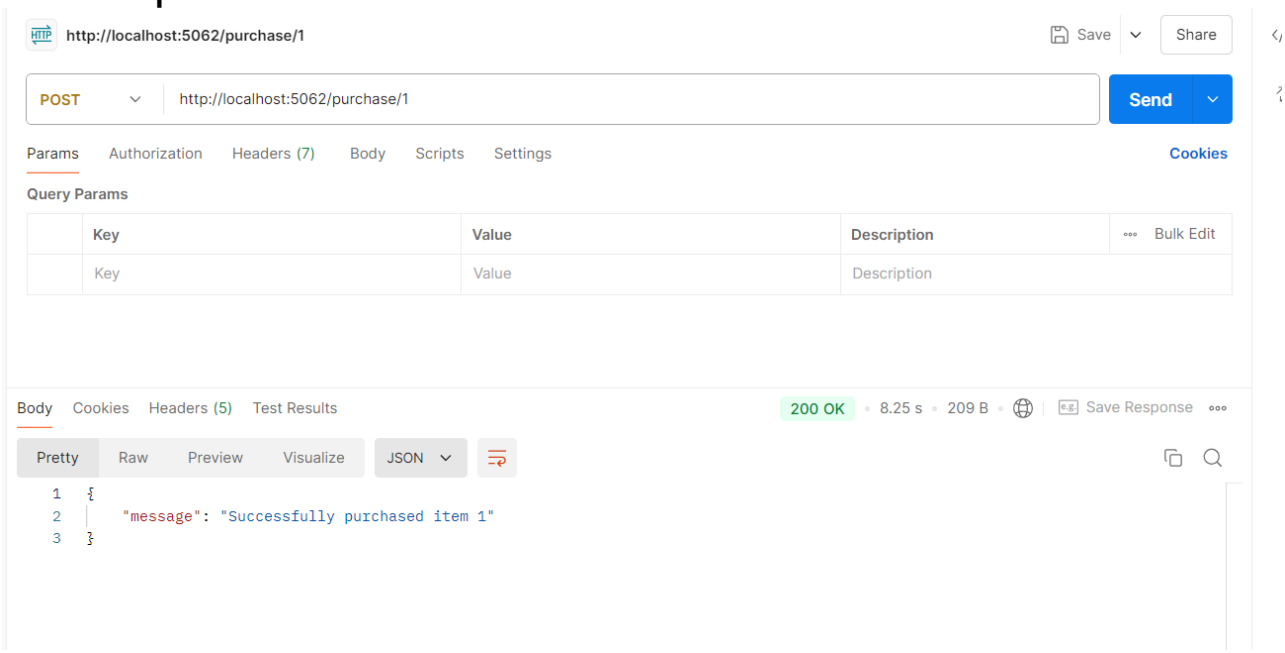
The first time the cache memory was empty the second time it was located in the cache memory:

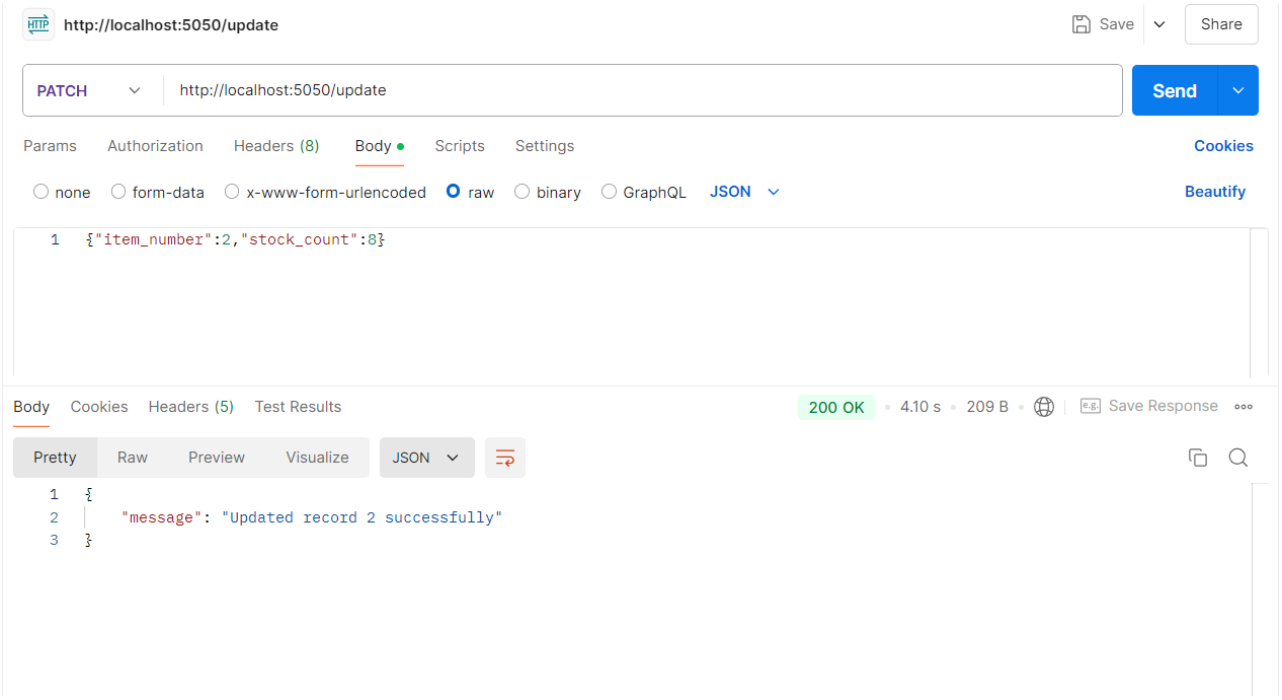




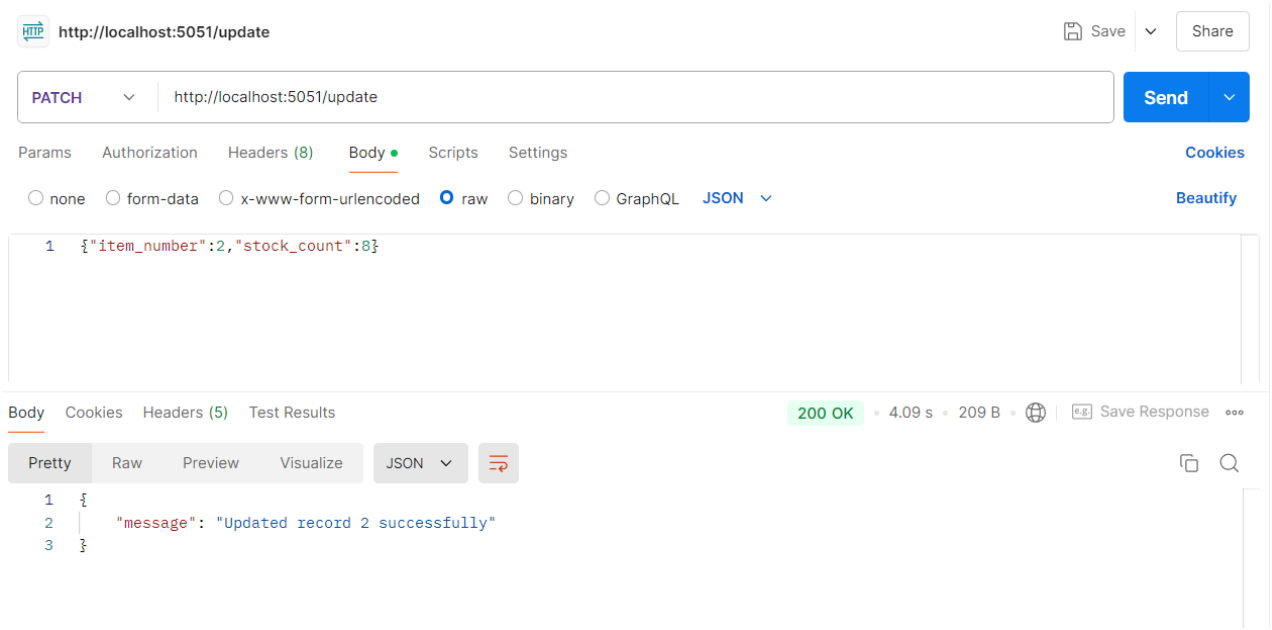


With replica :





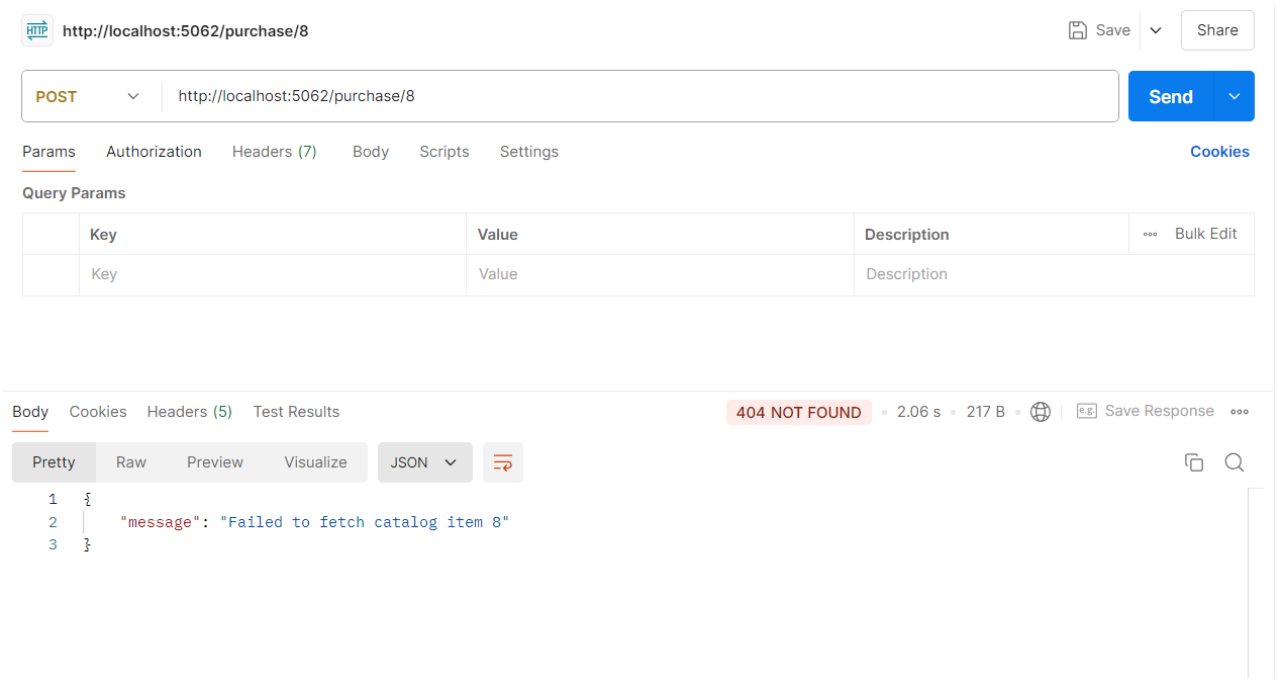
With replica :

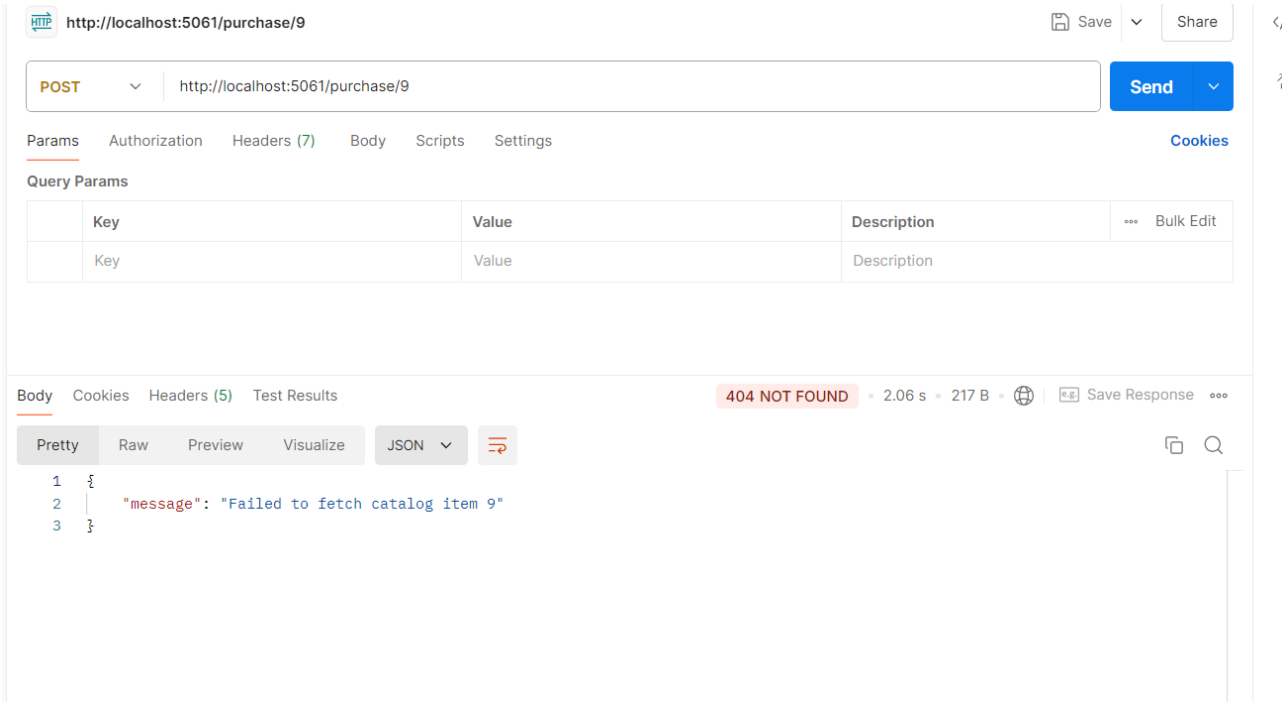


Q2) How much does caching help?

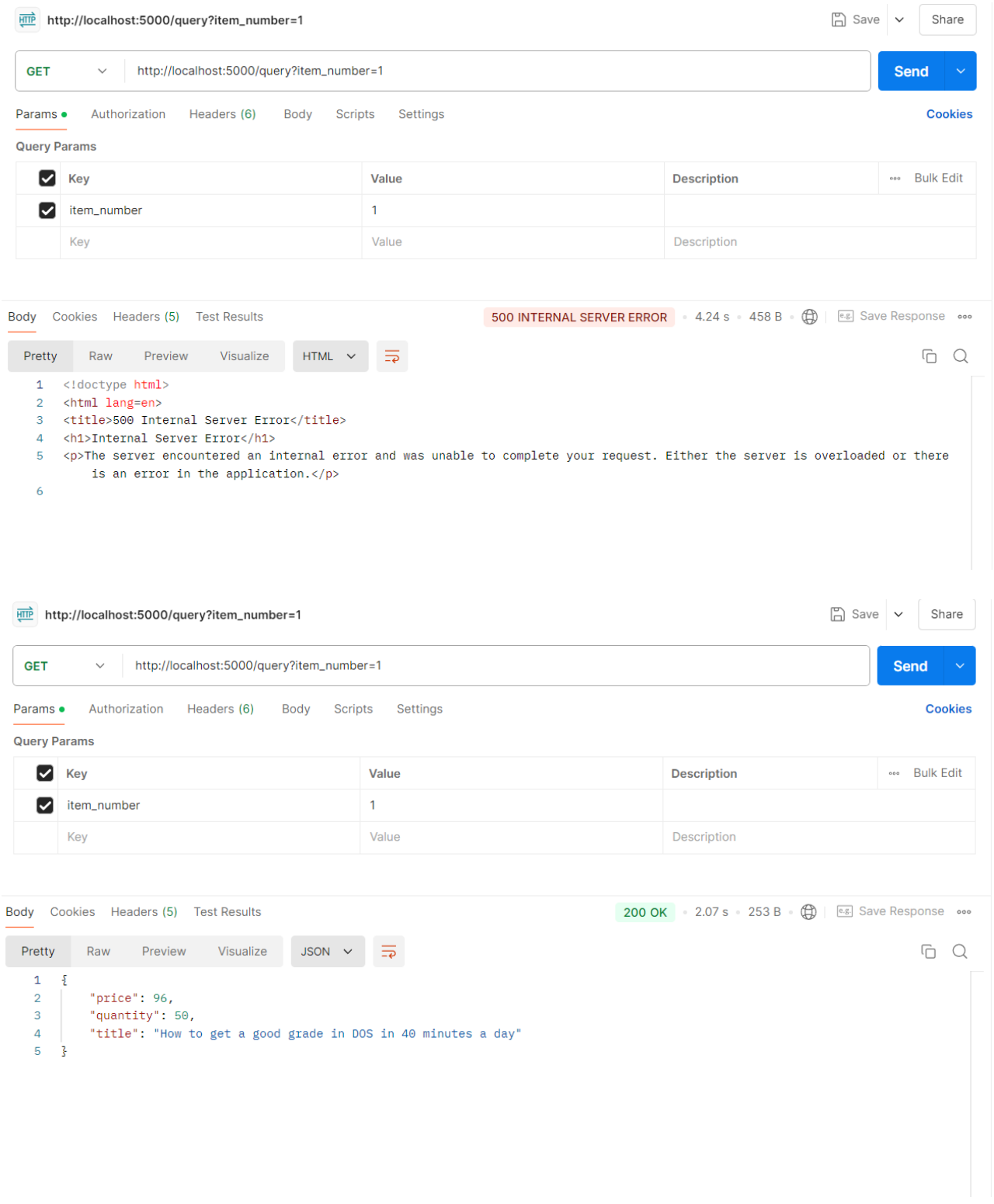
- Answers
  - for catalog server: **12ms , 30/12 —> 2.5 Faster than without cache**
  - for gateway server: **7ms, 2.06s/7ms —> 294 Faster than without using cache**

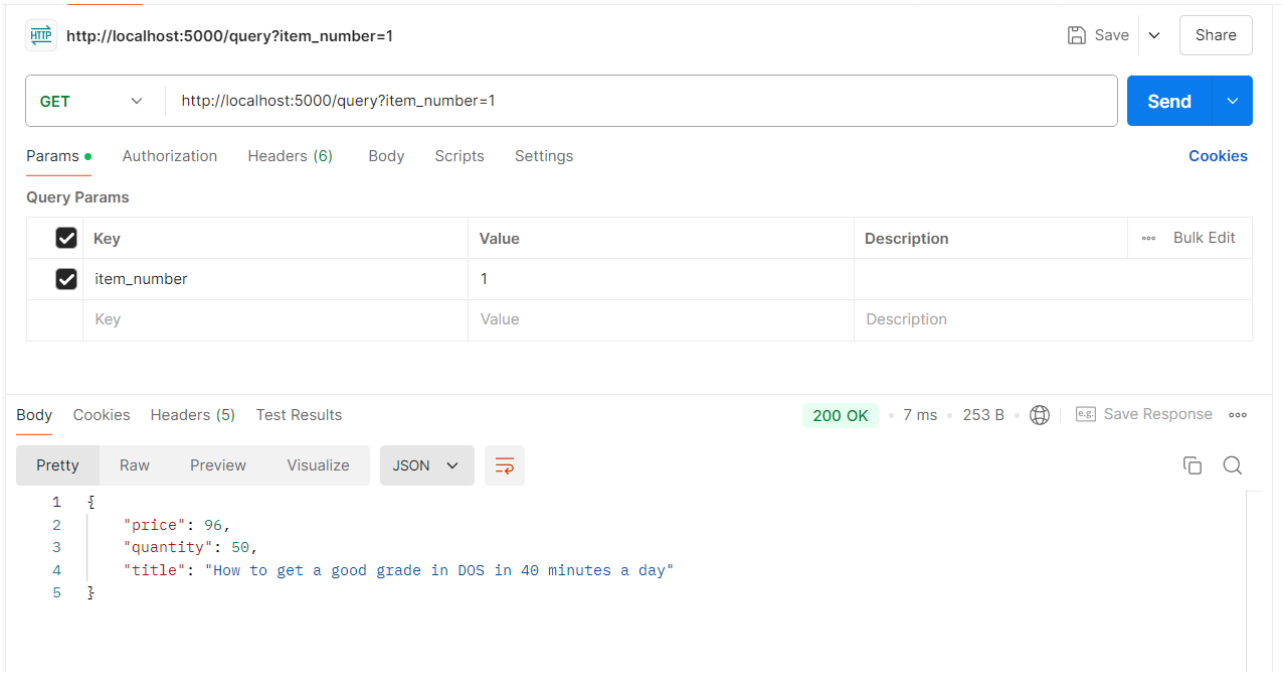
Invalidate Message



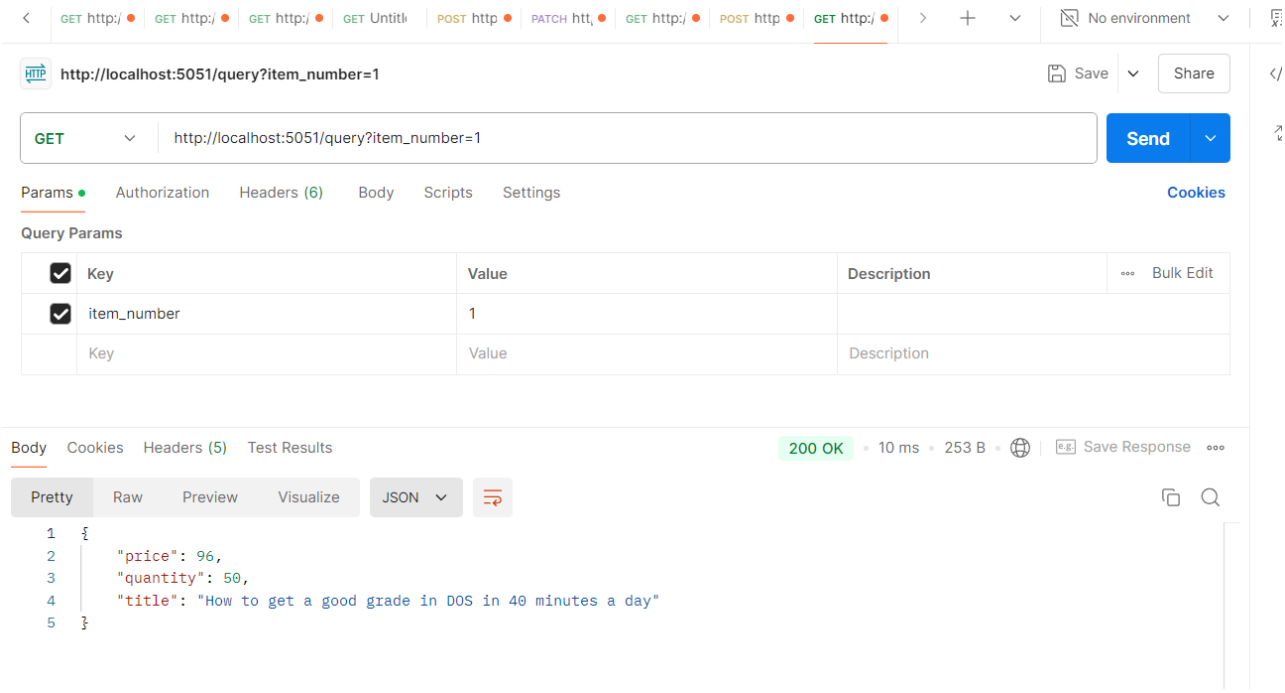

http://localhost:5061/purchase/9
Save
Share
POST
http://localhost:5061/purchase/9
Send
Params
Authorization
Headers (7)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
404 NOT FOUND
2.06 s
217 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "message": "Failed to fetch catalog item 9"
3 }

- for Origianl services:

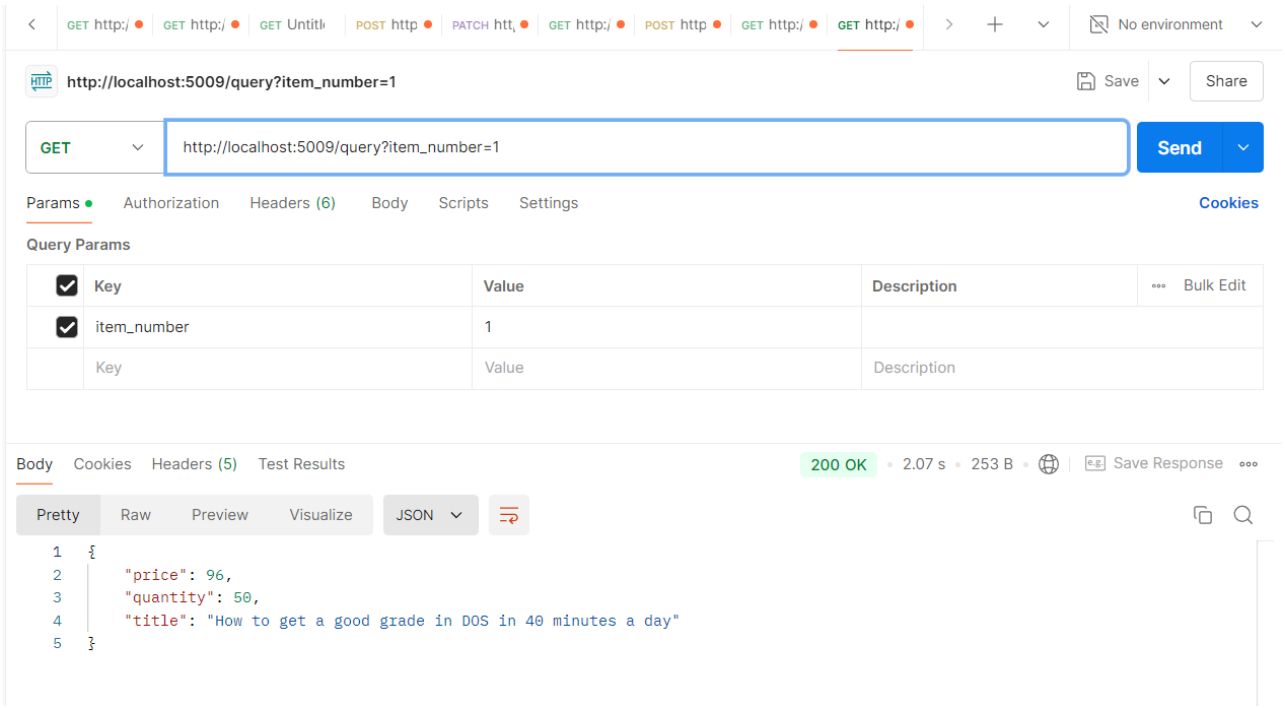



http://localhost:5000/query?item_number=1
Save
Share
GET
http://localhost:5000/query?item_number=1
Send
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
item_number
1
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
7 ms
253 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"price": 96,
"quantity": 50,
"title": "How to get a good grade in DOS in 40 minutes a day"
}

- for Replica Service



For gateway replica :



Part2: Dockerize your Application (Optional part):

I Construct my project Dockerized from scratch, i create docker container (image) for each service, and each service has it's own port, and i use volume for sharing data between the Guest OS (Docker Containers) & Host OS , and i create docker-compose file to run all containers at the same time with 1 simple command .

My `docker-compose.yml` **file:**

```yaml
version: '3.8'

services:
  catalog_api:
    build:
      context: ./catalog
    ports:
      - "5050:5000"
    networks:
      - app_network

  purchase_api:
    build:
      context: ./purchase
    ports:
      - "5060:5000"
    networks:
      - app_network

  gateway_api:
    build:
      context: ./gateway
    ports:
      - "8080:5000"
    networks:
      - app_network
    depends_on:
      - catalog_api
      - purchase_api

  load_balancer_api:
    build:
      context: ./loadbalancer
    ports:
      - "8090:4000"
    networks:
      - app_network
    depends_on:
      - catalog_api
      - purchase_api
```

```
networks:
  app_network:
    driver: bridge
```

Docker Containers(images) while running:

```
time="2024-11-18T00:46:20+02:00" level=warning msg="C:\\Users\\User\\Desktop\\library_microservices-main\\
library_microservices-main\\compose.yaml: the attribute `version` is obsolete, it will be ignored, please
remove it to avoid potential confusion"
[+] Running 4/4[+] Running 2/4
 ✓ Container library_microservices-main-purchase_api-1       Started                    1.7s
 ✓ Container library_microservices-main-catalog_api-1        Started                    1.7s
 ✓ Container library_microservices-main-load_balancer_api-1  Started                    4.6s
 ✓ Container library_microservices-main-gateway_api-1        Started                    4.3s
```