An-Najah National University

Faculty of Engineering

Computer Engineering Department

Distributed Operation Systems

Project part2

Student's Name:

AreejAmjad sawalha

Ibtisam kharrousheh

Student's ID:

12028958

12028305

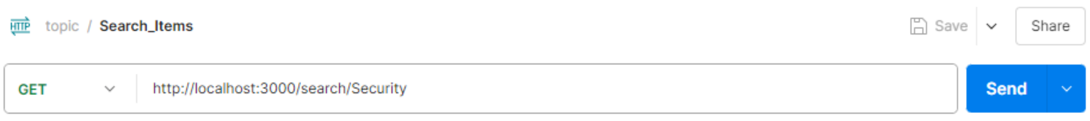
Introduction:

1) **Fronted service:**

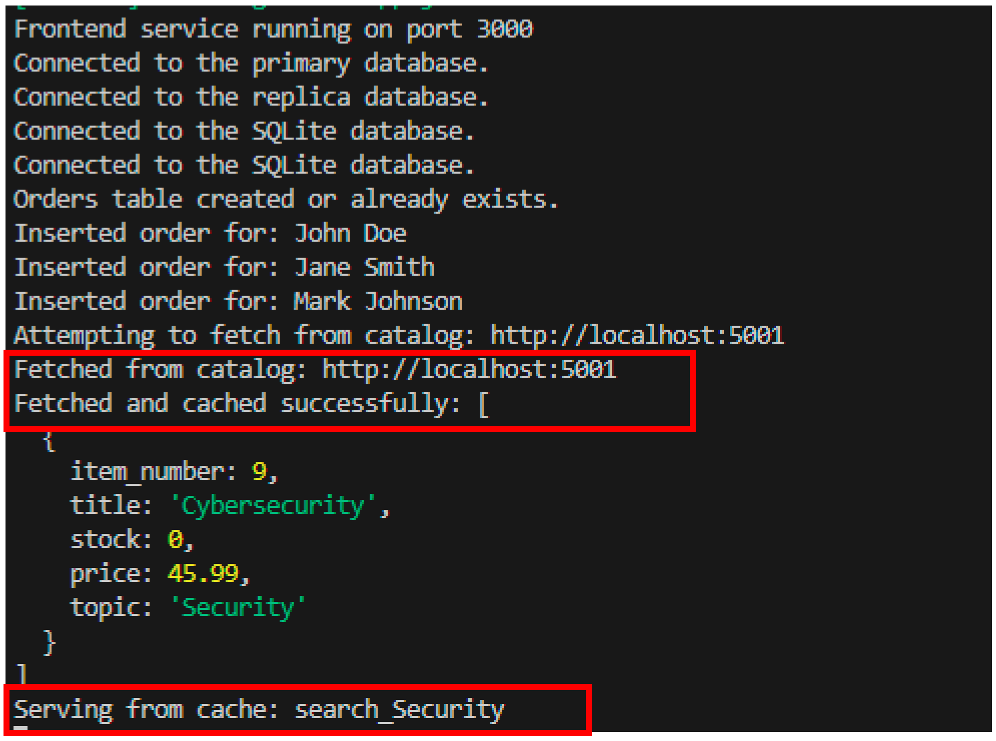
This server has three operations:

Cache : The importance of the cache is if the data is used permanently and is taken to put it in the cache instead of dealing with the service directly. This is beneficial in terms of performance and speed.

1. Search: *When implementing this api for the first time, the data is imported from the server and placed in the cache. Then, when you press Send again, the data is taken from the cache.*

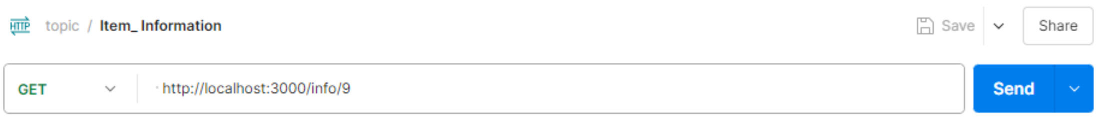


```
http://localhost:3000/search/Security
```



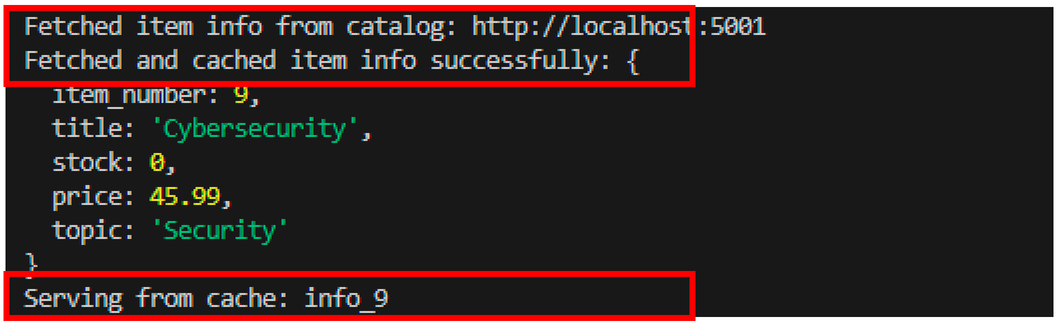
```
Frontend service running on port 3000
Connected to the primary database.
Connected to the replica database.
Connected to the SQLite database.
Connected to the SQLite database.
Orders table created or already exists.
Inserted order for: John Doe
Inserted order for: Jane Smith
Inserted order for: Mark Johnson
Attempting to fetch from catalog: http://localhost:5001
Fetched from catalog: http://localhost:5001
Fetched and cached successfully: [
  {
    item_number: 9,
    title: 'Cybersecurity',
    stock: 0,
    price: 45.99,
    topic: 'Security'
  }
]
Serving from cache: search_Security
```

2. *Information:*



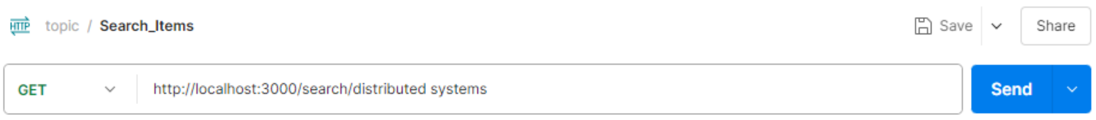
topic / Item_ Information

GET http://localhost:3000/info/9



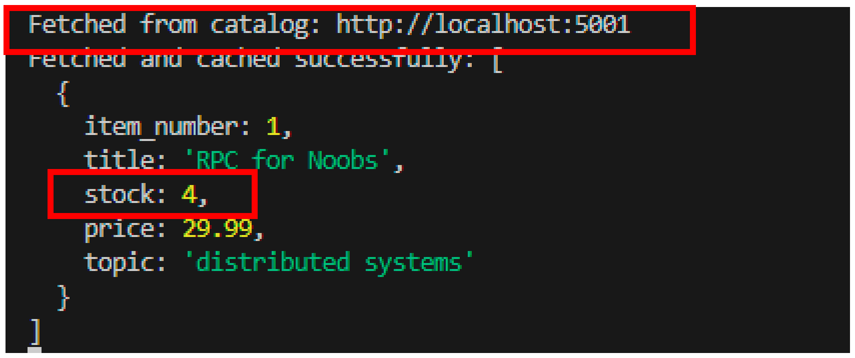
```
Fetched item info from catalog: http://localhost:5001
Fetched and cached item info successfully: {
  item_number: 9,
  title: 'Cybersecurity',
  stock: 0,
  price: 45.99,
  topic: 'Security'
}
Serving from cache: info_9
```

3. *Purchase:*

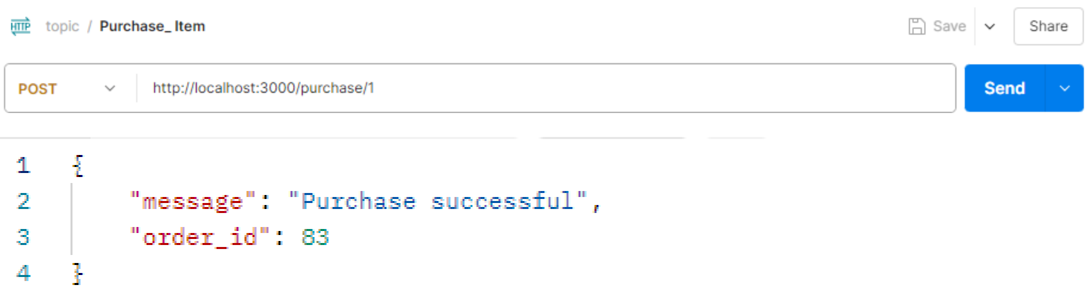


topic / Search_Items

GET http://localhost:3000/search/distributed systems



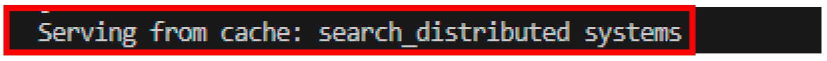
```
Fetched from catalog: http://localhost:5001
Fetched and cached successfully: [
  {
    item_number: 1,
    title: 'RPC for Noobs',
    stock: 4,
    price: 29.99,
    topic: 'distributed systems'
  }
]
```



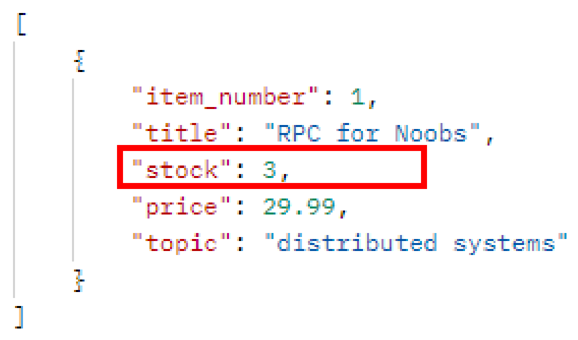
topic / Purchase_ Item

POST http://localhost:3000/purchase/1

```
{
    "message": "Purchase successful",
    "order_id": 83
}
```

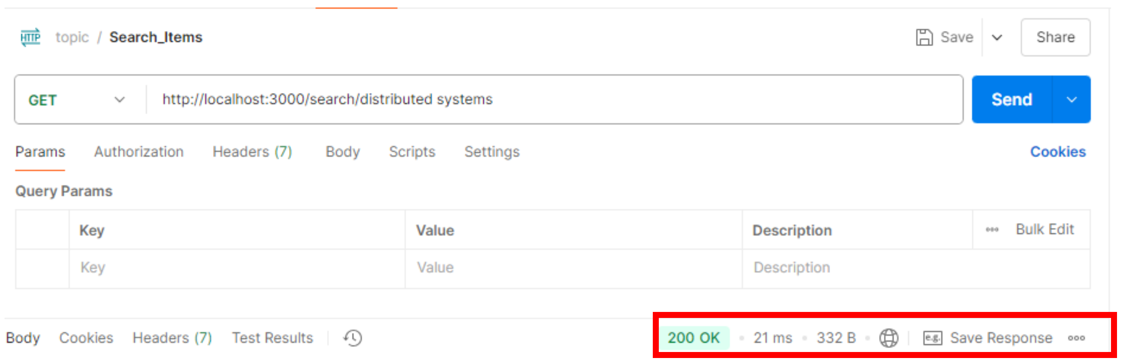


```
Serving from cache: search_distributed systems
```



```
[
    {
        "item_number": 1,
        "title": "RPC for Noobs",
        "stock": 3,
        "price": 29.99,
        "topic": "distributed systems"
    }
]
```

When executing the first search request, accessing the data from the main server takes time __**21ms**__



topic / Search_Items

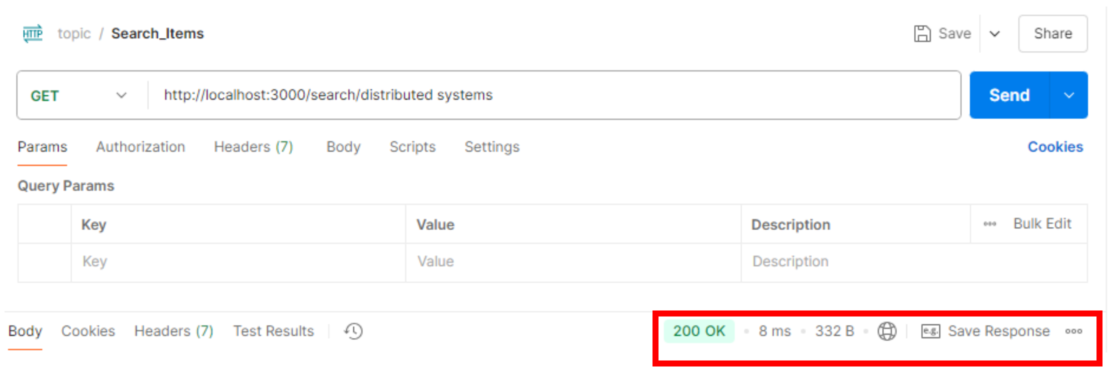
GET http://localhost:3000/search/distributed systems

Query Params

	Key	Value	Description
	Key	Value	Description

200 OK 21 ms 332 B

But when the request is executed again and the data is taken from the cache, the time becomes less, the speed is better, and the performance is better __***8ms***__



topic / Search_Items

GET http://localhost:3000/search/distributed systems

Query Params

	Key	Value	Description
	Key	Value	Description

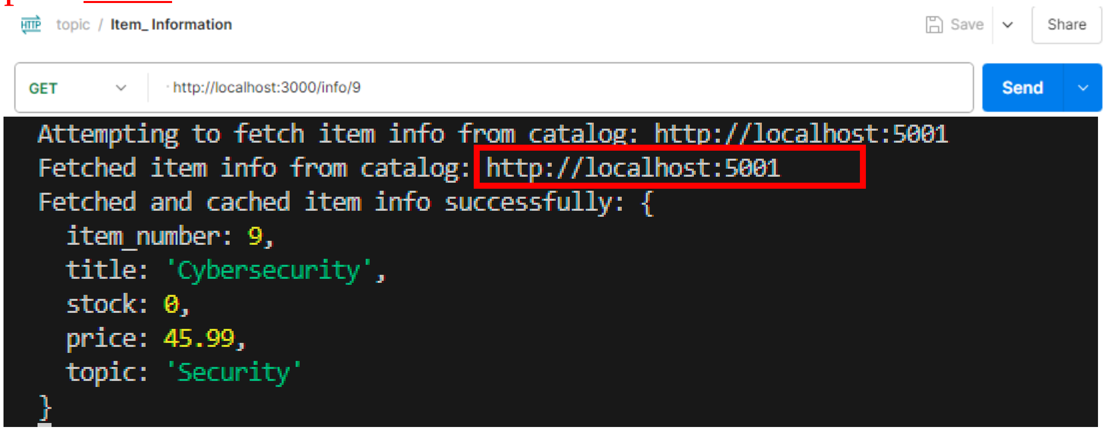
200 OK 8 ms 332 B

__***Raplication server :***__

To distribute the pressure when a request is requested, a copy of the requested data is created from the server. If the request is requested and the underlying service is busy, the request will be executed using replication. This increases performance and execution speed and also reduces pressure on the primary server.

Example :

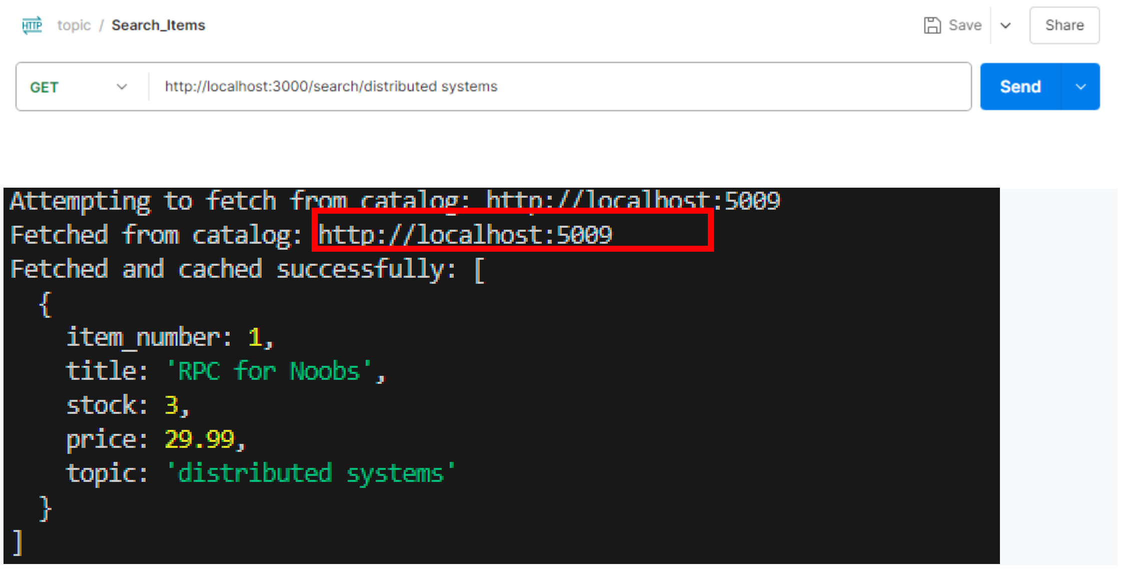
When you click this API, it is executed on the primary server at port 5001



http://localhost:3000/info/9

```
Attempting to fetch item info from catalog: http://localhost:5001
Fetched item info from catalog: http://localhost:5001
Fetched and cached item info successfully: {
  item_number: 9,
  title: 'Cybersecurity',
  stock: 0,
  price: 45.99,
  topic: 'Security'
}
```

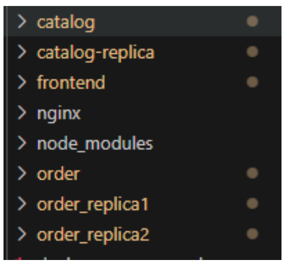
When another request is executed, the replicator server is chosen to relieve pressure on the replicate server 5009



http://localhost:3000/search/distributed systems

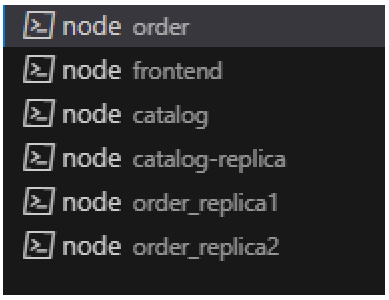
```
Attempting to fetch from catalog: http://localhost:5009
Fetched from catalog: http://localhost:5009
Fetched and cached successfully: [
  {
    item_number: 1,
    title: 'RPC for Noobs',
    stock: 3,
    price: 29.99,
    topic: 'distributed systems'
  }
]
```

This file :



```
> catalog
> catalog-replica
> frontend
> nginx
> node_modules
> order
> order_replica1
> order_replica2
```

<u>This run :</u>



```
node order
node frontend
node catalog
node catalog-replica
node order_replica1
node order_replica2
```

Conclusion :

By storing frequently used data, minimizing the need to constantly get it from slower storage or databases, and guaranteeing quicker response times, cache enhances system performance.
By making copies of data in several places, replication improves data availability and dependability and guarantees continuation in the event of system failures or heavy demand.