DOS Project Report Part 2

Student Name: Muna Khwaireh Stu#: 12028473

Student Name: Samah Qaradeh Stu#: 12028923

Introduction:

This project aims to improve Bazar.com's online bookstore by reducing response times and handling higher user demand. To achieve this, we incorporated **caching** and **replication** to enhance performance and ensure data consistency, especially during frequent data updates. Additionally, we used **Dockerization** to simplify deployment and management of the bookstore's microservices as containerized applications.

By implementing caching for frequently accessed data and distributing requests across replicated servers, we reduced backend load and improved response times. This report outlines the updated system's design, evaluates performance gains, and demonstrates how these enhancements support a more scalable and responsive application.

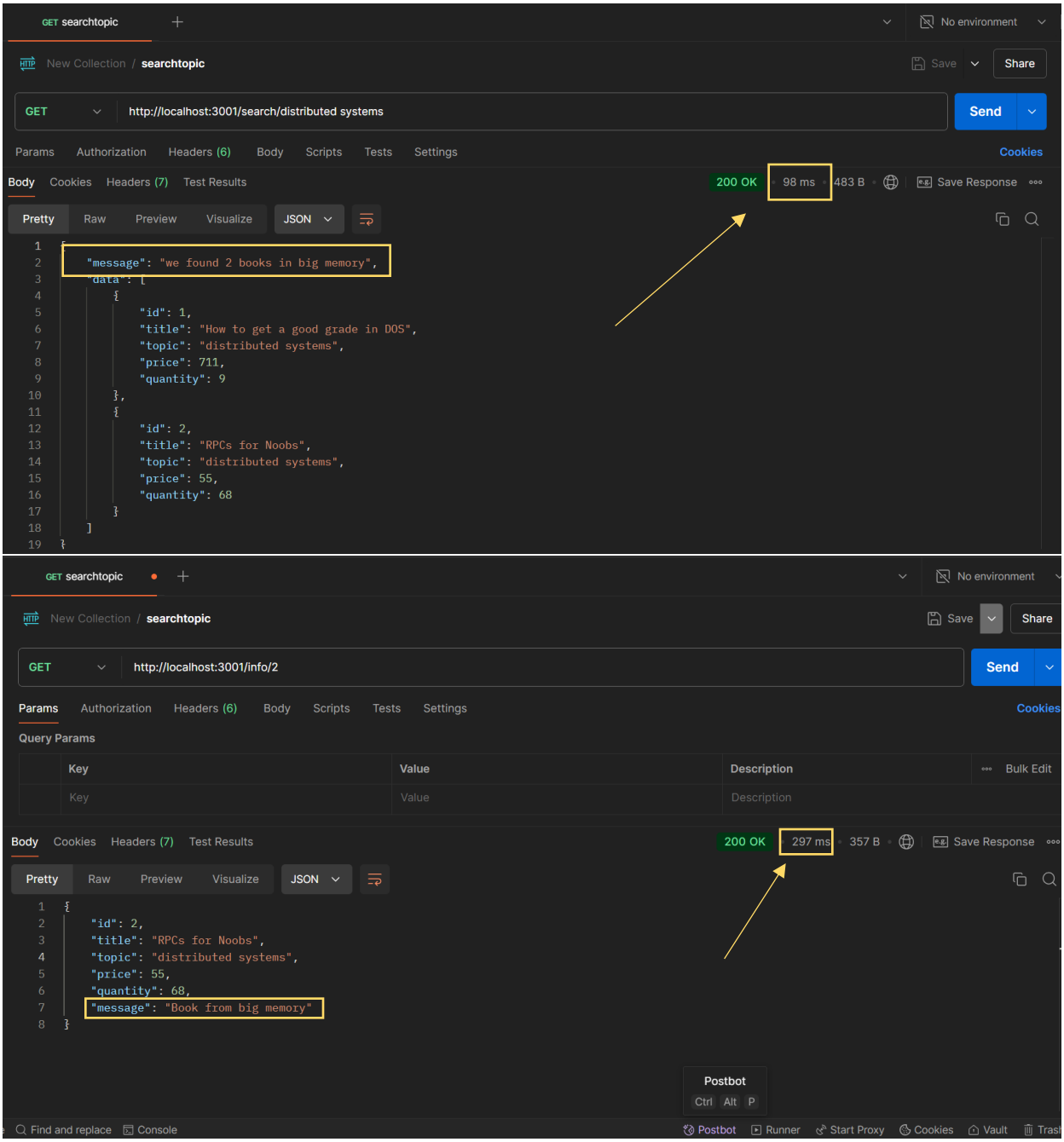
Part1: Cache Consistency & load balancing:

```
// Route to get book information by item ID
app.get('/info/:id', async (req, res) => {
  const itemId = req.params.id;
  try {


    // Forward the info request to the catalog service
  const response1=await axios.get(`http://catalogcash:3004/info/${itemId}`);
  const msg1=response1.data.message;
  const specificMessage1 = 'Book from small cash';

   if (msg1 === specificMessage1) {
            res.json(response1.data);
       } else {

       const response2 = await axios.get(`http://catalog:3002/info/${itemId}`);

       if(response2.data.message==="Book from big memory"){
            const response3 = await axios.post(`http://ordercash:3005/add_new_book`, response2.data);
            res.json(response2.data);}
       else {

         res.json({
       message: "Book not found in big memory and not found in small cash"
    });
       }
```

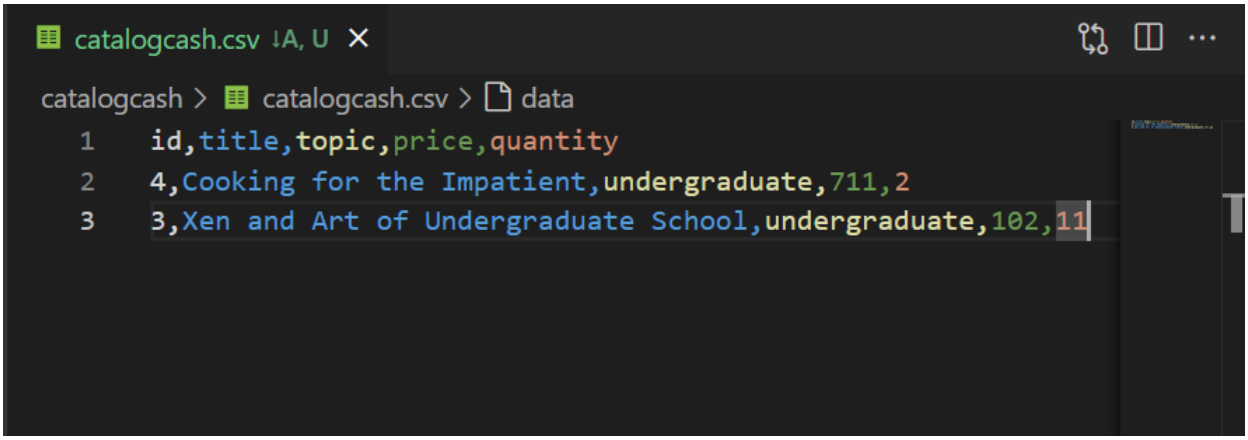
Note: we has 5 operations: search,info, purchase, uodate cost and update quantity.

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

So the data will come from the memory


GET searchtopic
No environment
New Collection / searchtopic
Save
Share
GET
http://localhost:3001/search/distributed systems
Send
Params Authorization Headers (6) Body Scripts Tests Settings
Cookies
Body Cookies Headers (7) Test Results
200 OK 98 ms 483 B Save Response
Pretty Raw Preview Visualize JSON
{
"message": "we found 2 books in big memory",
"data": [
{
"id": 1,
"title": "How to get a good grade in DOS",
"topic": "distributed systems",
"price": 711,
"quantity": 9
},
{
"id": 2,
"title": "RPCs for Noobs",
"topic": "distributed systems",
"price": 55,
"quantity": 68
}
]
}
GET searchtopic
No environment
New Collection / searchtopic
Save
Share
GET
http://localhost:3001/info/2
Send
Params Authorization Headers (6) Body Scripts Tests Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body Cookies Headers (7) Test Results
200 OK 297 ms 357 B Save Response
Pretty Raw Preview Visualize JSON
{
"id": 2,
"title": "RPCs for Noobs",
"topic": "distributed systems",
"price": 55,
"quantity": 68,
"message": "Book from big memory"
}
Postbot
Ctrl Alt P
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash

Our cache capacity is 2 books:

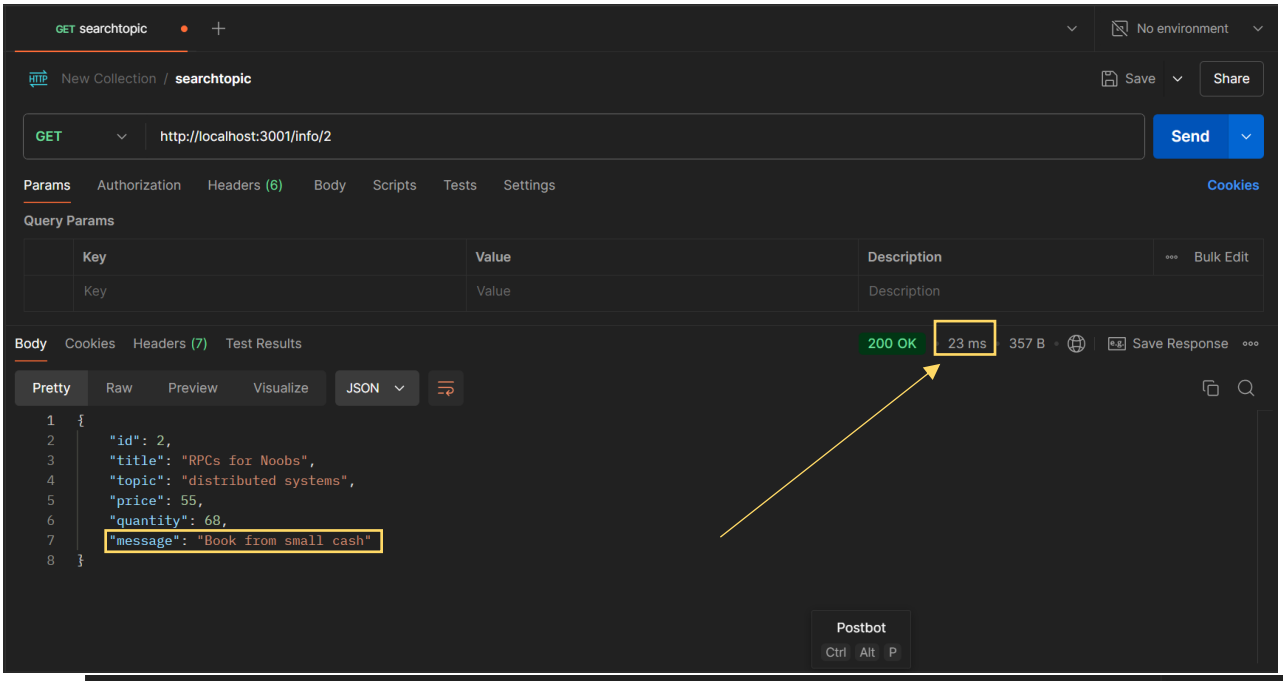


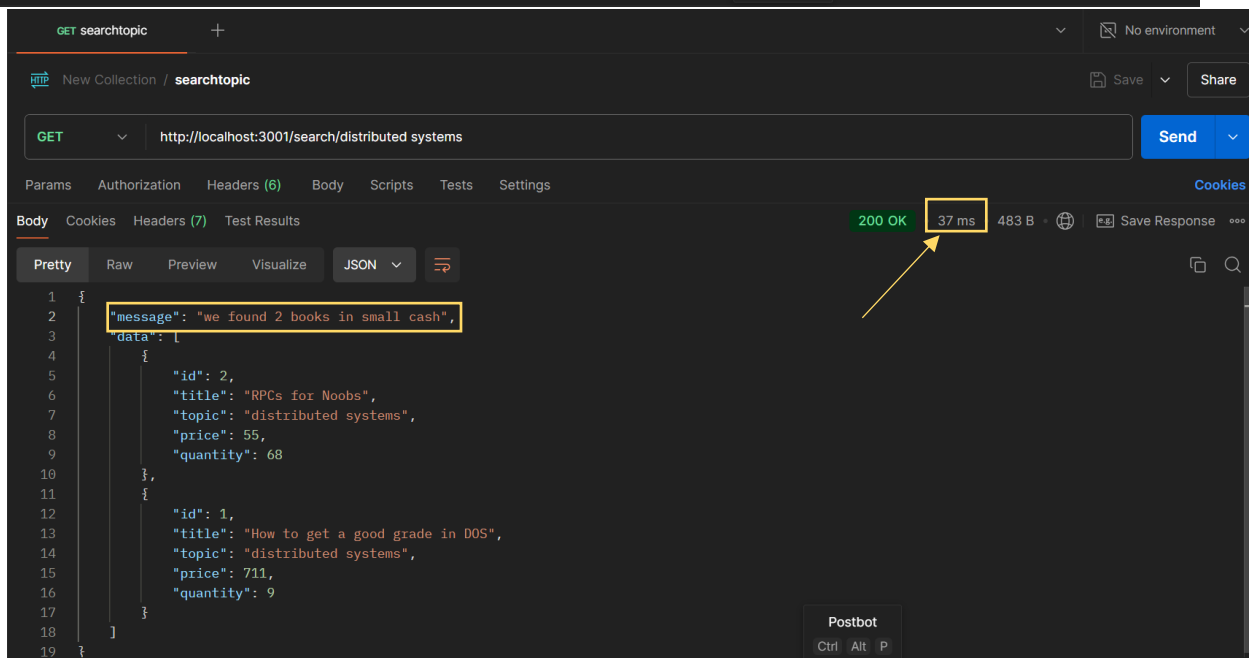
Q1) Compute the average response time (query/buy) of your new systems.
What is the response time with and without caching?

- Answers
  - for info: **297ms** for
  - search: **98ms**

after the above requests, the book will be added to the cache , and the below requests shows that the result come from the cache 😊

with cash:



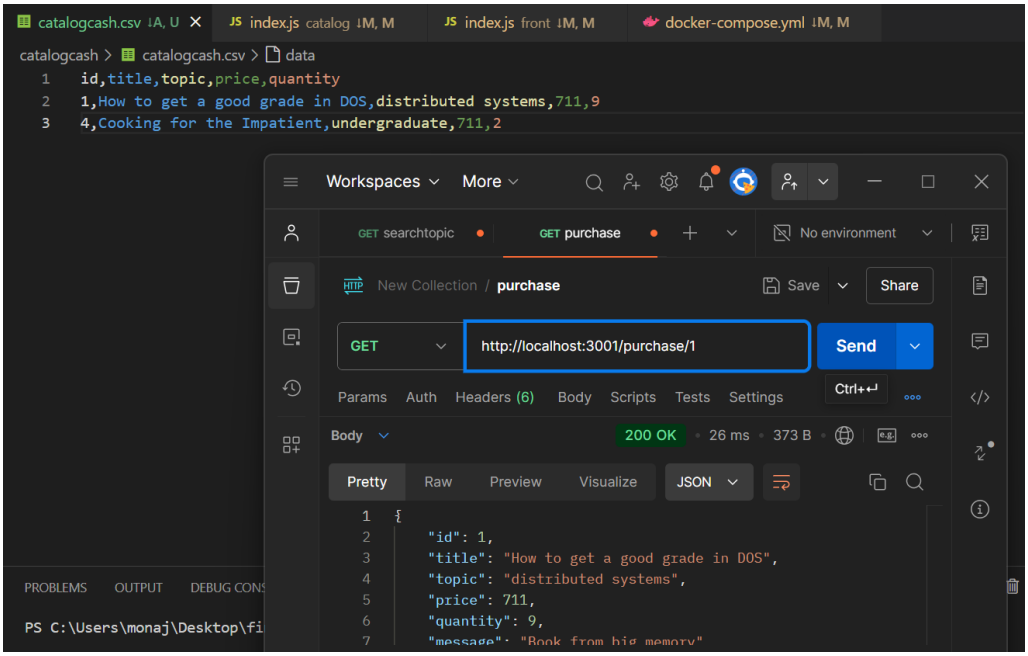


Q2) How much does caching help?

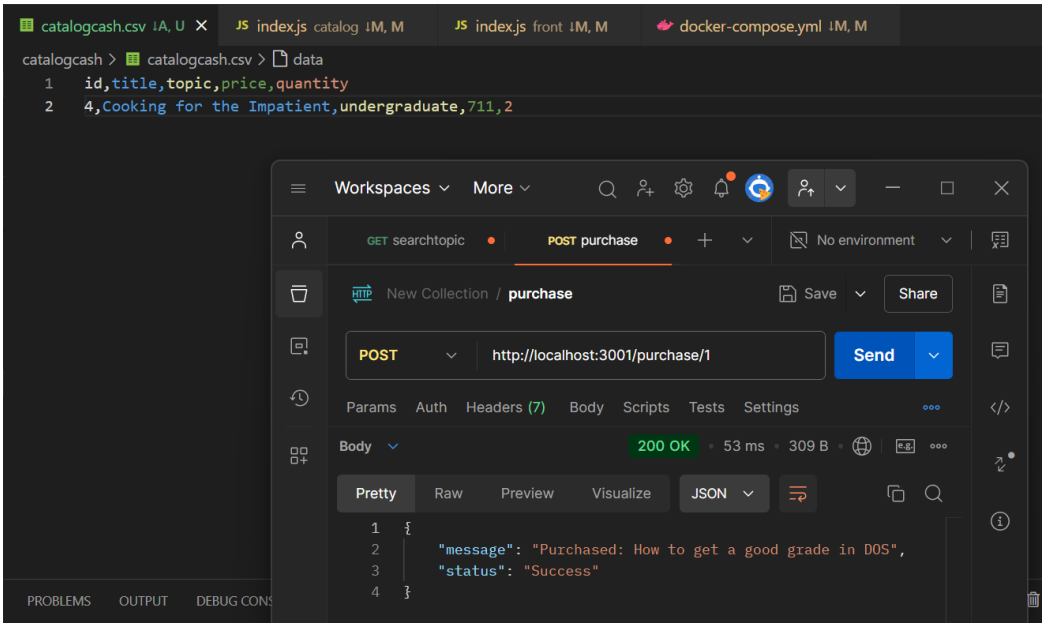
- Answers for info: **23ms , 297/23 —> 12.9 Faster than without cache**
  - search: **37ms, 98/37—> 2.65 Faster than without using cache**
  -

Invalidate Message

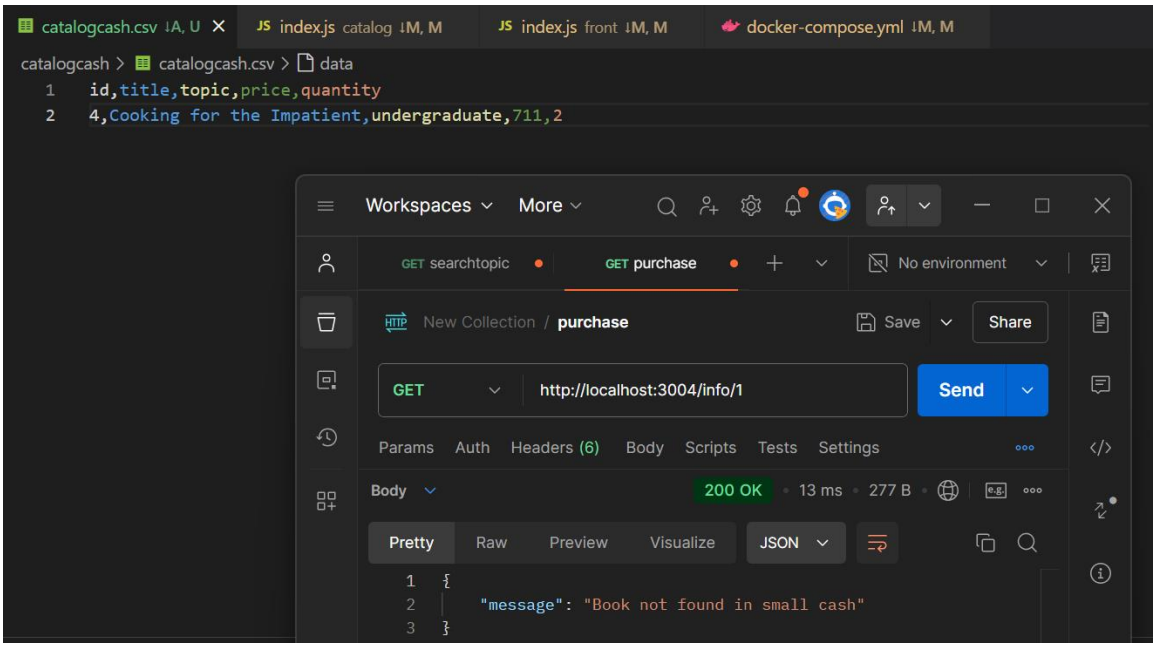
At the first I will purchase a book that is already at the cache:
This is before make the request:



And here after the request, the book just get out of the cache:

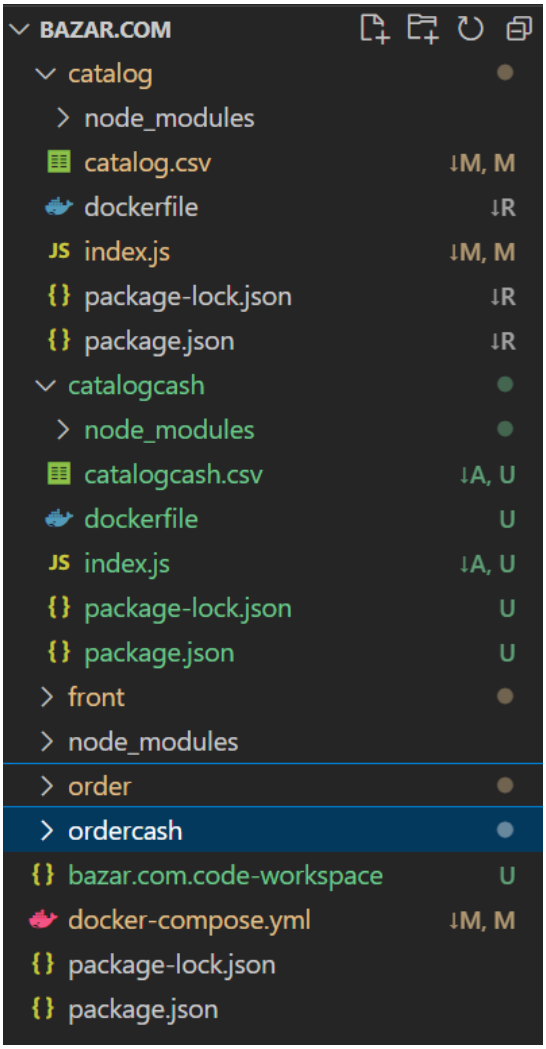


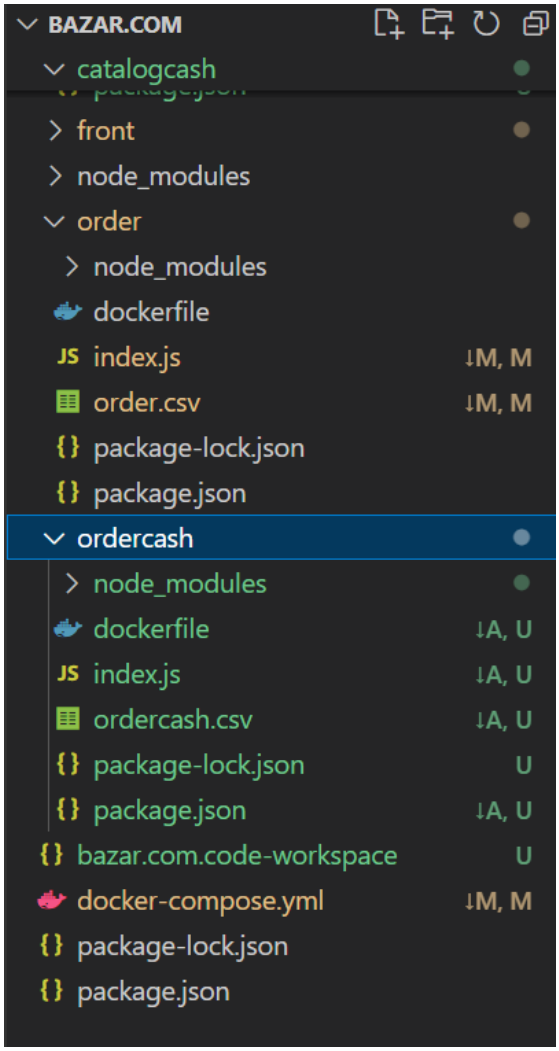
And if I search at cache for it:



Expirement Run	Without Cache	With Cache	#of Times when using Cache speed
1	46	9	46/9 = 5.11 Times
2	26	7	26/7 = 3.71 Times
3	27	9	27/7 = 3.85 Times

For Replication Services & Database:





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 compose.yml file:

```yaml
version: '3.8'

services:
  frontend:
    build: ./front
    ports:
      - "3001:3001"
    depends_on:
      - catalog
      - order
      - catalogcash
      - ordercash
    networks:
      - app-network

  catalog:
    build: ./catalog
    ports:
      - "3002:3002"
    volumes:
      - ./catalog:/catalog  # Mount the catalog directory
    networks:
      - app-network
order:
    build: ./order
    ports:
      - "3003:3003"
    volumes:
      - ./catalog:/catalog  # Mount the same catalog directory for access
      - ./order:/order
    networks:
      - app-network
```

```yaml
  catalogcash:
    build: ./catalogcash
    ports:
      - "3004:3004"
    depends_on:
      - catalog
      - order
    volumes:
      - ./catalogcash:/catalogcash  # Mount the catalog directory
    networks:
      - app-network

  ordercash:
    build: ./ordercash
    ports:
      - "3005:3005"
    depends_on:
      - catalog
      - order
    volumes:
      - ./catalogcash:/catalogcash  # Mount the same catalog directory for
access
      - ./ordercash:/ordercash
    networks:
      - app-network

networks:
  app-network:
    driver: bridge
```

Docker Containers(images) while running:

```
C:\Users\monaj\Desktop\fin1\fin1\bazar.com> docker ps
CONTAINER ID   IMAGE                 COMMAND                  CREATED       STATUS       PORTS                    NAMES
c7323d23e529   bazarcom-frontend     "docker-entrypoint.s…"   3 hours ago   Up 3 hours   0.0.0.0:3001->3001/tcp   bazarcom-frontend-
1
74cb86116a58   bazarcom-catalogcash  "docker-entrypoint.s…"   3 hours ago   Up 3 hours   0.0.0.0:3004->3004/tcp   bazarcom-catalogca
sh-1
6eb982aff944   bazarcom-ordercash    "docker-entrypoint.s…"   3 hours ago   Up 3 hours   0.0.0.0:3005->3005/tcp   bazarcom-ordercash
-1
a58eb8934a12   bazarcom-order        "docker-entrypoint.s…"   3 hours ago   Up 3 hours   0.0.0.0:3003->3003/tcp   bazarcom-order-1
1b415185698d   bazarcom-catalog      "docker-entrypoint.s…"   3 hours ago   Up 3 hours   0.0.0.0:3002->3002/tcp   bazarcom-catalog-1
```

For Running The Project, it’s the same for Part1:

```
docker-compose up -d -- build
```