**Program Implementation: Turning the Bazar into an Amazon**

**داليا حامد سليم دويكات 11923878**

**I run the code using PowerShell for each server on different port,**

**frontend server : localhost 5000**

**catalog server : localhost 5002**

**catalog server- replica : localhost 5003**

**order server : localhost 5004**

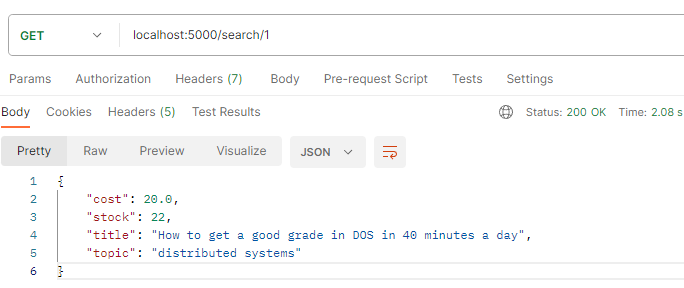
**order server - replica : localhost 5005**

**Senario 1 : Get book with id = 1**

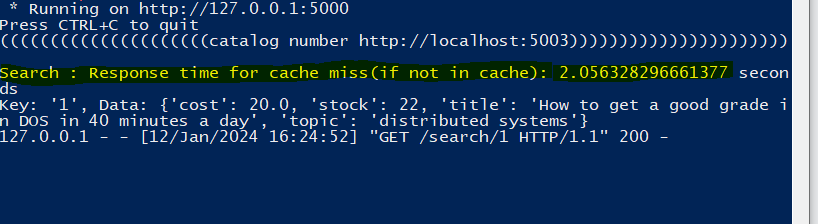
* **using search requist**

**request : localhost:5000/search/1-GET**

**response :**

****

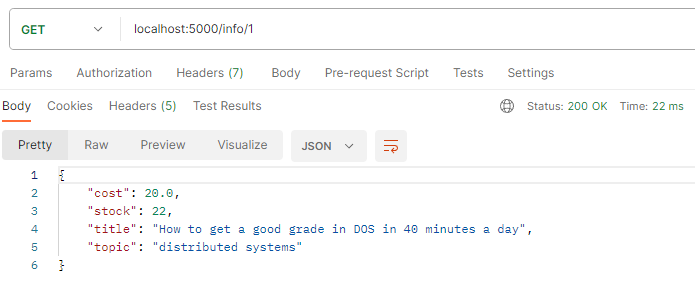
* **Book 1 is not in the cache, so the system will take the book from the catalog- -replica database chosen by the system using the round Robin algorithm, the time response (shows the color yellow in the image), and the data inside the cache is printed on the screen also:**

****

* **using info request**

**request : localhost:5000/info/1-GET**

**response :**

****

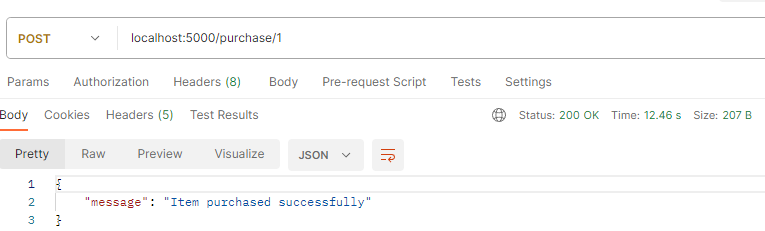
* **Book 1 is now inside the cache so the system directly will return the data without the need to go to the database, which takes a little time, and makes the performance better, and the time response is in the image:**

****

**Scenario 2: Purchase Book 1**

**request : localhost:5000/purchase/1 - POST**

**response :**

****

**book 1 is inside the cache, but this operation will update the database and decrement one of book 1 from the database, so the cache must know that the value is changed in the database, and it can’t get the value inside it to any request, so the catalog server when will change the value in the database, will send a request for frontend server to tell the cache that the value for book 1 is updated, so the cache will do something to highlight that item, in my program, the value for the item in cache will change to**

**“{**

**"message": "Item purchased successfully"**

**}**

**”**

**so when any new request is sent and this item is the response, the system first will check the value for this item so if it is equal to this {**

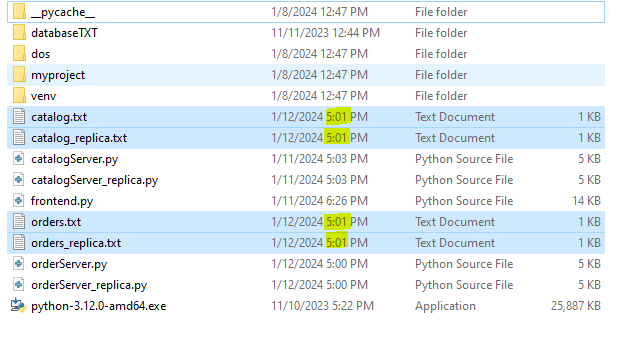
**"message": "Item purchased successfully"**

**}**

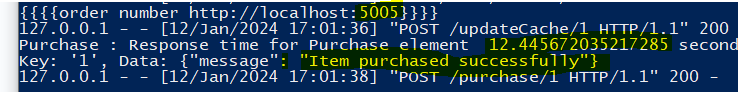
**, then the system will delete this element from the cache and again will get the item from the database, also the catalog server and the catalog server - replica will tell each other to rechange the data in their databases to be the is updated all the time.**

**also, the order server and the replica will be sent to each other to change is database files.**

**this image show that the changes happen for each file:**

****

**and this image shows that the system by round robin algo used order -replica with port 5005, and shows the purchase time and also sees the element inside the cache with the new value before is deleted and changed :**

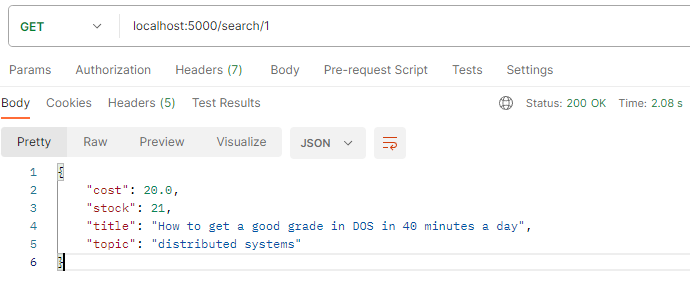
****

**Scenario 3: try to get book 1 information and show the time :**

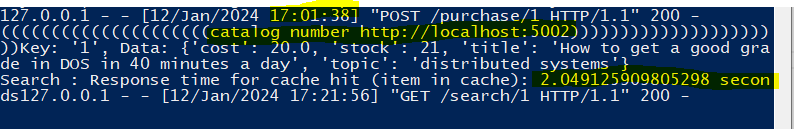
**As I mentioned in the previous scenario, the data will not get from the database because the data is changed so will notice that when seeing the time response.**

**request : localhost:5000/search/1**

**response:**

****

**notice that the element has a new stock value of 21, notice the time in the image, also notice that the information is given from the catalog server after is changed by round robin, also see the response time after updating the data found in the cache with new values,**

****

**This applies to all items in the database.**

**—------------------------------------------------------------------------------------------**

**Tabel to show the time with cache and without cache:**

* **know that I run the code program for lab 1 to get the time response without cache, and for lab 2 to get the time without cache :**

| **Scenario** | **Program Without cache**  **(Lab 1)** | **Program with cache**  **(element not in cache)** | **Program with cache**  **(element in cache )** |
| --- | --- | --- | --- |
| **Search** | **2.0867 seconds** | **2.07204008102417 s** | **8ms** |
| **Search** | **2.0590 seconds** | **2.081376314163208 s** | **16ms** |
| **Search avg** | **2.07285** | **2.076708197593689** | **0.012 seconds** |
| **Info** | **2.0403 seconds** | **2.0542657375335693 s** | **11ms** |
| **Info** | **2.0471 seconds** | **2.0478622913360596 s** | **10ms** |
| **Info avg** | **2.0437** | **2.0510640144348145** | **10.5ms** |
| **Purchase** | **6.2020 s** | **12.458033800125122 s** | **12.437422275543213 s** |
| **Purchase** | **6.2081 s** | **12.451984643936157 s** | **12.434534311294556 s** |
| **Purchase avg** | **6.20505 s** | **12.45500922203064 s** | **12.435978293418884 s** |

**The average response time of a search operation depends on whether the code includes a cache and whether the data is present in the cache. Here's a description for each scenario:**

**1. No Cache (2.07285 seconds):**

**- In this case, the search operation does not utilize any caching mechanism.**

**- Each search request involves retrieving data directly from the source or performing a computation, leading to a relatively higher response time.**

**- The average response time is 2.07285 seconds, indicating the time it takes for the search operation without any caching optimization.**

**2. \*\*Cache, Data Not Inside (2.076708197593689 seconds):\*\***

**- The code includes a caching mechanism, suggesting an attempt to optimize search performance.**

**- However, in this specific instance, the data needed for the search is not found inside the cache.**

**- The code still needs to retrieve the data from the original source or perform the necessary computation, resulting in a response time slightly higher than the case with the cache being absent.**

**- The average response time is 2.076708197593689 seconds, indicating the time it takes when the cache is present but the required data is not inside it.**

**3. \*\*Cache, Data Inside (0.012 seconds):**

**- In this scenario, the code includes a caching mechanism, and the data required for the search operation is found inside the cache.**

**- As a result, the search operation can be satisfied quickly by retrieving the data from the cache, leading to a significantly lower response time.**

**- The average response time is 0.012 seconds, indicating the time it takes when both the cache is present and the required data is inside it.**

**In summary, the presence and efficiency of a caching mechanism significantly impact the response time of search operations. Caching can substantially reduce response times when the required data is already stored in the cache, providing a more optimized and responsive system.**

**and the same thing for the info operation.**

**Purchase operation :**

**The purchasing process differs from the process of fetching information from the database, and thus, we observe a considerable difference in time in each case. In the first case, when the program does not have a cache inside, the data modification process does not take much time compared to the second and third cases. In the second and third cases, the time does not differ whether the data is in the cache or not. This is because the data modification process requires going back to the database, explaining the increased execution time. This is because modifications occur in both the server's database and the replica's database. Additionally, modification occurs in the requests file, which, in turn, sends a request to the replica's request server to modify its files. Moreover, a request is sent to the cache to inform it of the data change so that it takes into account the data stored inside it. Therefore, in this process, approximately four files change in addition to a request to the cache. Consequently, this process takes less time in the program that does not have a cache.**