Bazar.com

Student1 : Samia Hamed ,12113019
Student2 : Assel Zaid,12112794

What is Bazaar.com?

Bazar.com is a micro-e-commerce application built on a microservices architecture, designed to simulate the online buying and selling of books.

Each service within the project is responsible for a specific part of the system, facilitating development, testing, and future expansion.

Operating and testing steps:

- After individually building each service and creating a dedicated **Docker file** for each one, and we configured a Docker Compose file to orchestrate and run all services together as a single multi-container application.
  The **Docker Compose** file defines how each service is built, connected, and exposed, allowing them to communicate seamlessly within the same network.
  Finally, we used the following command to build and start all services simultaneously in detached mode:
  `docker compose up -d --build`
  This command rebuilds the images (to include any recent changes), then starts all containers in the background so the application can run smoothly as an integrated system.

```
=> [catalog runtime 3/3] COPY --from=build /app/publish .
=> [catalog] exporting to image
=> => exporting layers
=> => exporting manifest sha256:78205133f74d08ca5f38fd6a9d5e5d75abce09102cfab88c145df345684b535f
=> => exporting config sha256:3538d53a6300d9aecbcc536c3678fd192370c0ce8cb786d0adb6a8351b6f0ab0
=> => exporting attestation manifest sha256:d3a9defb5b2b852014c76c260194a92fdbc78e17499f3b23f6441
=> => exporting manifest list sha256:029c65415acf11405bc9f3d901f5296f0c61ee4fbeb437392dd0da5523bd
=> => naming to docker.io/library/bazarcom-catalog:latest
=> => unpacking to docker.io/library/bazarcom-catalog:latest
=> [catalog] resolving provenance for metadata file
+] Running 7/7
✔bazarcom-catalog          Built
✔bazarcom-order            Built
✔bazarcom-front            Built
✔Network bazarcom_default  Created
✔Container bazar_catalog   Started
✔Container bazar_order     Started
✔Container bazar_front     Started
```

- The command docker ps shows all the containers that are currently running.
  Each row in the output represents one running service (container) in the project.
  The columns display useful information such as:
  CONTAINER ID: the unique ID of each container.
  IMAGE: the Docker image used to create the container.
  COMMAND: what the container is running (e.g., dotnet, python).
  STATUS: shows that the container is up and running.
  PORTS: shows which ports are exposed and mapped to the host machine.
  NAMES: the name given to the container.
  In our project, the output confirms that the three services — Catalog, Order, and Front ,are all running successfully.

```
C:\Users\samya\OneDrive\Desktop\Bazar.com>docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                                       NAMES
151abab79da4   bazarcom-catalog   "dotnet Catalog_Serv…"   31 seconds ago   Up 30 seconds   0.0.0.0:5142->5142/tcp, [::]:5142->5142/tcp   bazar_catalog
67e025330639   bazarcom-front     "python app.py"          27 minutes ago   Up 27 minutes   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp   bazar_front
dad724deab35   bazarcom-order     "docker-entrypoint.s…"   27 minutes ago   Up 27 minutes   0.0.0.0:3002->3002/tcp, [::]:3002->3002/tcp   bazar_order
```

- is used to access and query the SQLite database inside the running bazar_catalog container.
  - docker exec -it bazar_catalog → runs a command inside the bazar_catalog container interactively.
  - sqlite3 /app/catalog.db → opens the SQLite database file located at /app/catalog.db.
  - "SELECT * FROM Books;" → runs an SQL query that retrieves all the records from the Books table.

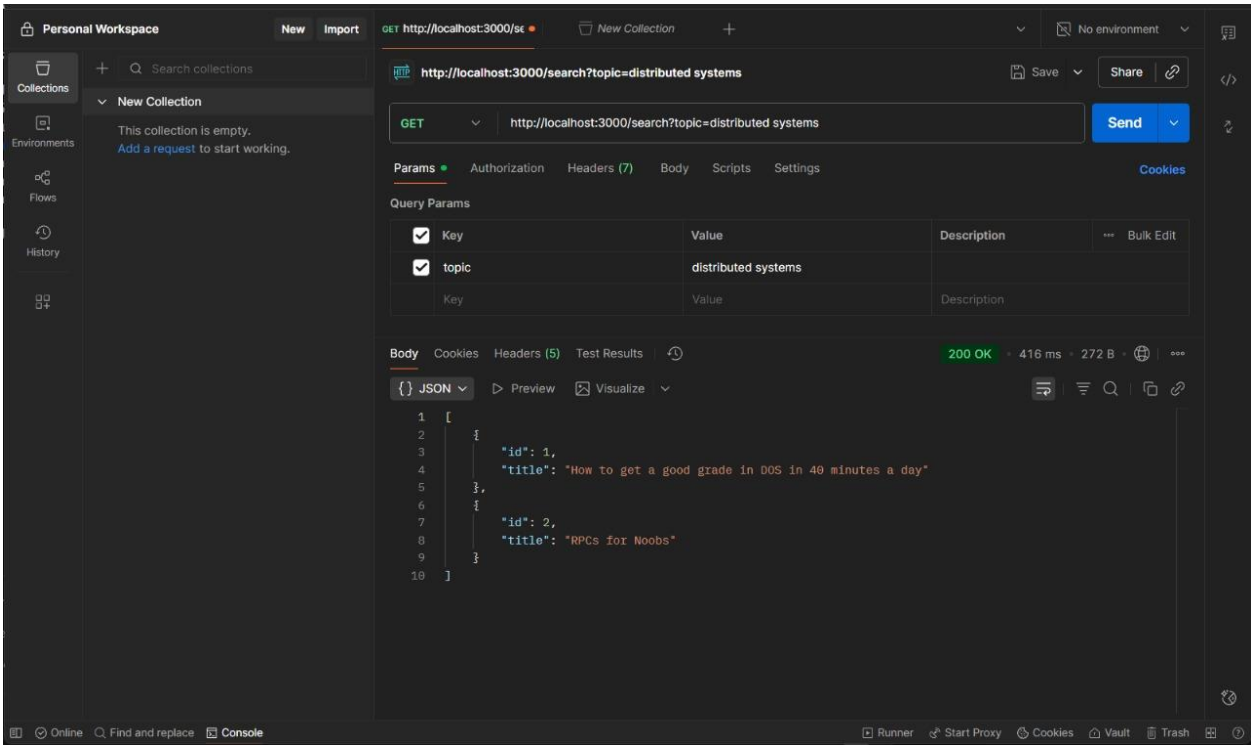
  The output displayed the stored books data , which confirms that the database was created successfully and preloaded with sample data.

```
C:\Users\samya\OneDrive\Desktop\Bazar.com>docker exec -it bazar_catalog sqlite3 /app/catalog.db "SELECT * FROM Books;"
1|How to get a good grade in DOS in 40 minutes a day|distributed systems|5|45.0
2|RPCs for Noobs|distributed systems|10|50.0
3|Xen and the Art of Surviving Undergraduate School|undergraduate school|3|40.0
4|Cooking for the Impatient Undergrad|undergraduate school|7|30.0
```

- http://localhost:3000/search?topic=distrbutedSystem

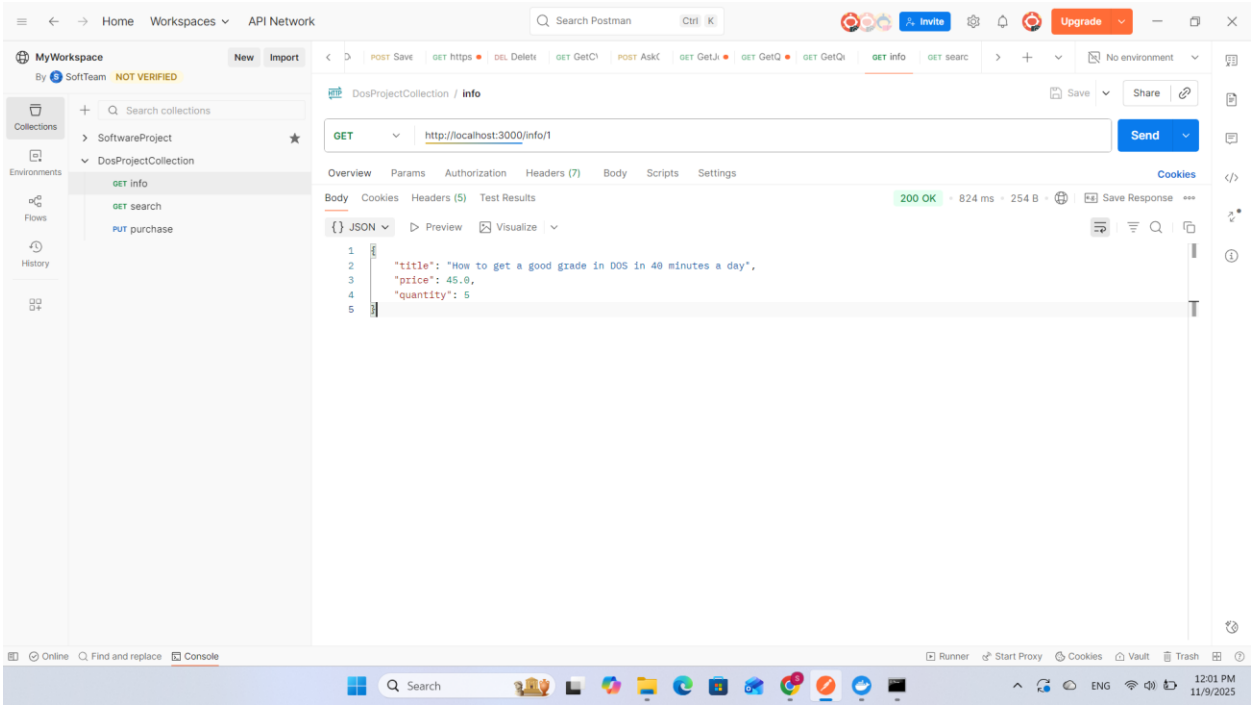
is used to **search for books by topic** on the front-end.

It sends a request to the Catalog Service and displays the matching books with **Title, and Id.**



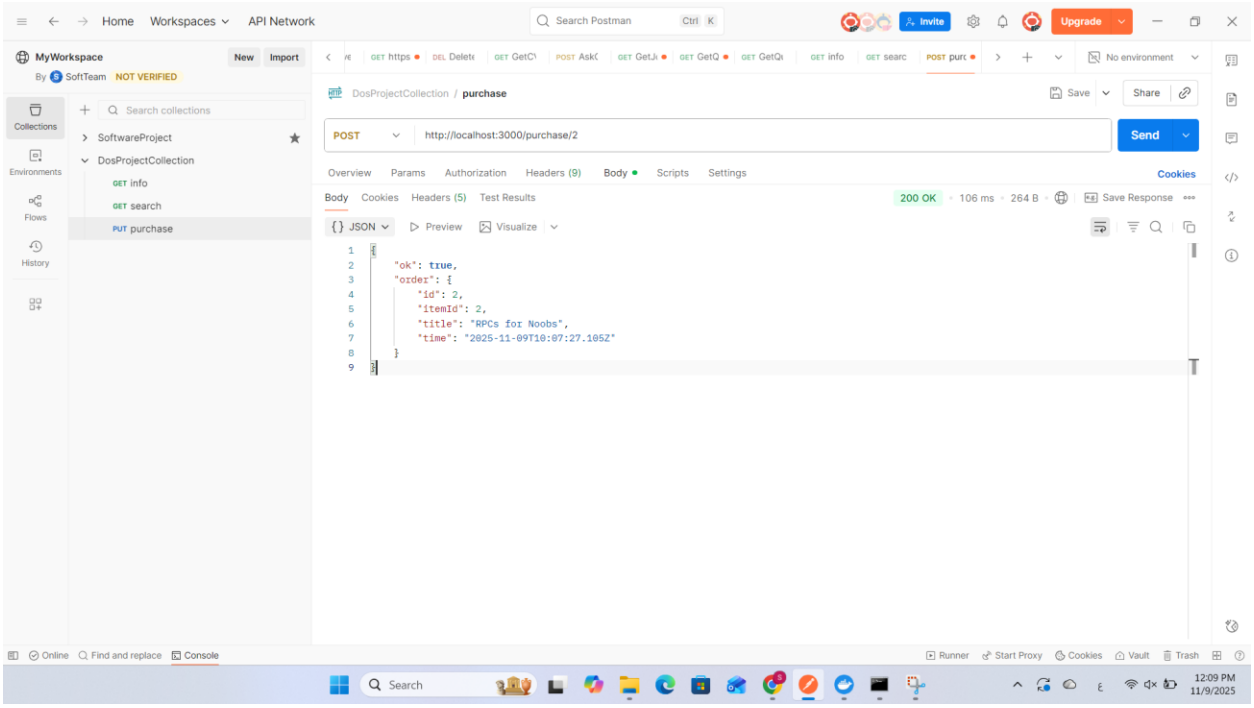
- http://localhost:3000/info/1

is a **front-end route** that shows the **details of a specific book** with ID = 1.
It fetches the book's information from the **Catalog Service** and displays its **Title, Quantity, and Price** to the user.



- http://localhost:3000/purchase/2

is a **front-end route** used to **purchase a book** with ID = 2.
It sends a request to the **Order Service**, which processes the purchase and updates the book's quantity in the Catalog Service.




```
root@5b12693ca7f1:/app# sqlite3 /app/catalog.db "SELECT * FROM Books;"
1|How to get a good grade in DOS in 40 minutes a day|distributed systems|5|45.0
2|RPCs for Noobs|distributed systems|9|50.0
3|Xen and the Art of Surviving Undergraduate School|undergraduate school|3|40.0
4|Cooking for the Impatient Undergrad|undergraduate school|7|30.0
```

```
root@5b12693ca7f1:/app# sqlite3 /app/catalog.db "SELECT * FROM Books;"
1|How to get a good grade in DOS in 40 minutes a day|distributed systems|5|45.0
2|RPCs for Noobs|distributed systems|8|50.0
3|Xen and the Art of Surviving Undergraduate School|undergraduate school|3|40.0
4|Cooking for the Impatient Undergrad|undergraduate school|7|30.0
root@5b12693ca7f1:/app#
```