



“An-Najah National University”

“Faculty of Engineering”

“Computer Engineering Department”

DOS Project Part-1

Preparing By

Aisha Ishtayeh “12028269”

- **The services in our project:**

- 1) **Front end service**
- 2) **Order service**
- 3) **Catalog service**

1)Front-end service:

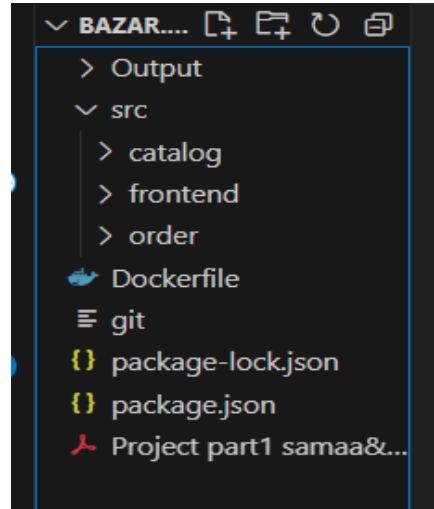
There are three operations in this server

- **1-Search:** the request is sent to catalog server then catalog return the item
[http. get\(http://catalog:4000/search/ Distributed systems \)](http://catalog:4000/search/Distributed%20systems)
- **2-Info:** the request is sent to catalog server then catalog return the info
[http. get\(http://catalog:4000/info/1 \)](http://catalog:4000/info/1)
- **3-Purchase:** The purchase order is sent to the order server
[app.post\(/purchase/:item_number/1\)](#)

2)Order service:

Receives requests from the front-end and transfers them to catalog server When it is sent to the catalog, the quantity of stock checked and modified.

[http.get\(http://catalog:4000/info/1\)](http://catalog:4000/info/1) [axios.put\('http://catalog:4000/update/1\)](#)

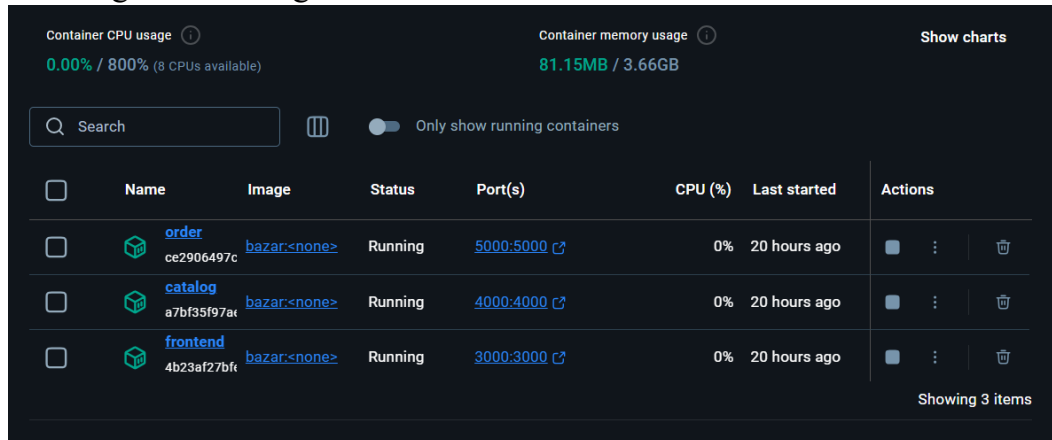


3)Catalog service:

It receives requests from the order server, adjusts the quantity, sends the response to the order, and also sends the response to search and info requests.

How to run the program?

1. Building docker image: ‘#docker build -t bazer .’

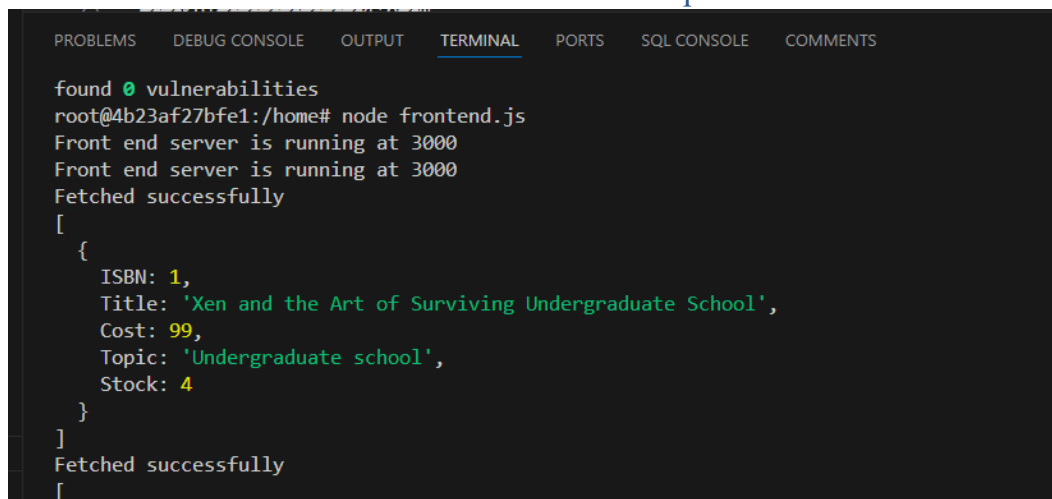


The screenshot shows the Docker Desktop interface. At the top, it displays 'Container CPU usage' at 0.00% / 800% (8 CPUs available) and 'Container memory usage' at 81.15MB / 3.66GB. Below this is a search bar and a toggle for 'Only show running containers'. A table lists three running containers: 'order' (ID: ce2906497c), 'catalog' (ID: a7bf35f97ac), and 'frontend' (ID: 4b23af27bfe). All are using the 'bazar:<none>' image. The 'frontend' container is highlighted. The table columns are: Name, Image, Status, Port(s), CPU (%), Last started, and Actions. The bottom right corner indicates 'Showing 3 items'.

	Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	order	bazar:<none>	Running	5000:5000	0%	20 hours ago	
<input type="checkbox"/>	catalog	bazar:<none>	Running	4000:4000	0%	20 hours ago	
<input type="checkbox"/>	frontend	bazar:<none>	Running	3000:3000	0%	20 hours ago	

2. Create a common network that make the services to communicate: ‘#docker network create bazar’
3. Running docker container for frontend server:

‘#docker run --network=bazar --name=frontend -p 3000:3000 -it -v ./:/home bazar’

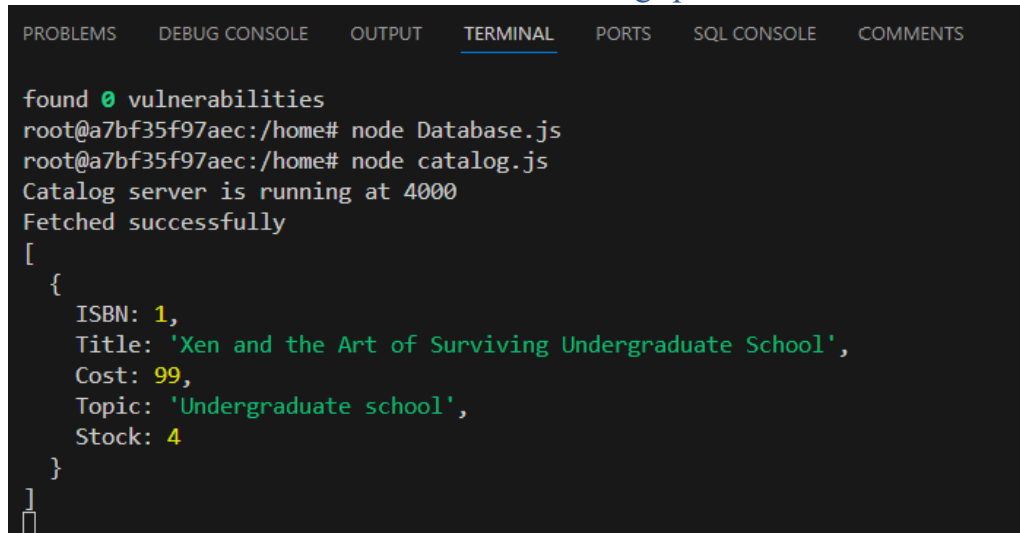


The screenshot shows a terminal window with the following output:

```
found 0 vulnerabilities
root@4b23af27bfe1:/home# node frontend.js
Front end server is running at 3000
Front end server is running at 3000
Fetched successfully
[
  {
    ISBN: 1,
    Title: 'Xen and the Art of Surviving Undergraduate School',
    Cost: 99,
    Topic: 'Undergraduate school',
    Stock: 4
  }
]
Fetched successfully
[
```

4. Running docker container for catalog server:

```
#docker run --network=bazar --name=catalog -p 4000:4000 -it -v ./home bazar'
```



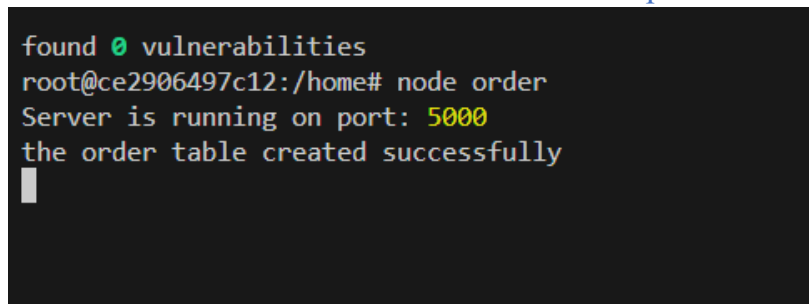
The screenshot shows a Docker terminal window with the following content:

```
PROBLEMS  DEBUG CONSOLE  OUTPUT  TERMINAL  PORTS  SQL CONSOLE  COMMENTS

found 0 vulnerabilities
root@a7bf35f97aec:/home# node Database.js
root@a7bf35f97aec:/home# node catalog.js
Catalog server is running at 4000
Fetched successfully
[
  {
    ISBN: 1,
    Title: 'Xen and the Art of Surviving Undergraduate School',
    Cost: 99,
    Topic: 'Undergraduate school',
    Stock: 4
  }
]
```

5. Running docker container for order serves:

```
#docker run --network=bazar --name=order -p 5000:5000 -it -v ./home bazar'
```



The screenshot shows a Docker terminal window with the following content:

```
found 0 vulnerabilities
root@ce2906497c12:/home# node order
Server is running on port: 5000
the order table created successfully
```

We use Node.js because it provides powerful microservices that are asynchronous and lightweight also it is scalable they are backed by a large community and require specific utilization and expertise.

- Testing in Postman

