Duaa shehada - 11716595

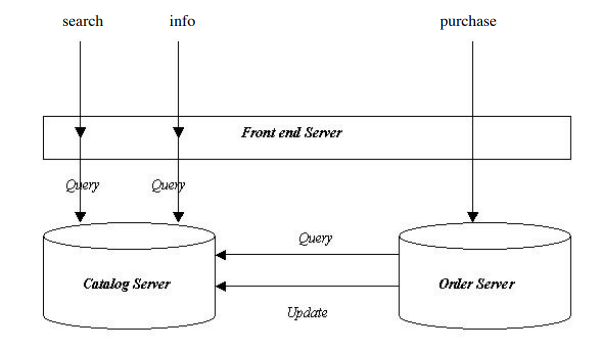
DOS - Bazar project :

This project use NodeJs , ExpressJs which it is a lightweight web framework use to create a RestFul project .it is contain of 3 servers , one for the front-end layer and it is called a Bazar , 2 servers for the back-end and they are the catalog and the order servers .

This project uses docker desktop to finish the task of deploying these servers on different machines and letting them communicate with each other.

I use the idea of monorepo , so all the code for all the servers locate in one repository

The servers talk to each other as mentioned in the project description :



So, the postman talks to the Bazar server , this server has three endpoints , one for the search and this endpoint sends a http request with GET verb and using axios npm package to the catalog server and then returns a response with a list of the matches books to a specific topic .

The other end point is info , also this end point sends a request to the catalog server using GET verbs and axios to send this http request , it returns a response with the information to the specific item .

Last endpoint is the purchase one. It uses the axios to send a http request to the order server with POST verb and a json body with the number of the item , it returns a response where you bought the book successfully or the book is not found .

Catalog server :

It contains three endpoints , one for the search , and it uses the JSON file where my data is there , and it searches for the matched items then returns it as a response , the http verb is GET .

The info one , use the same process but search on the id not the topic .

The update one uses the PUT verb ,it has the id of the item and there is more than one of the number of items in stock so the user can buy this book and it will decrease this number of items .

Order server :

Has one endpoint , which is the purchase one , it sends a request to the catalog server to search for this item then if it is found (there is a number of items more than one ) it will send a request to the catalog server to update this number of items .

Then it will save this purchase by creating a new json object in the OrderList.json file .

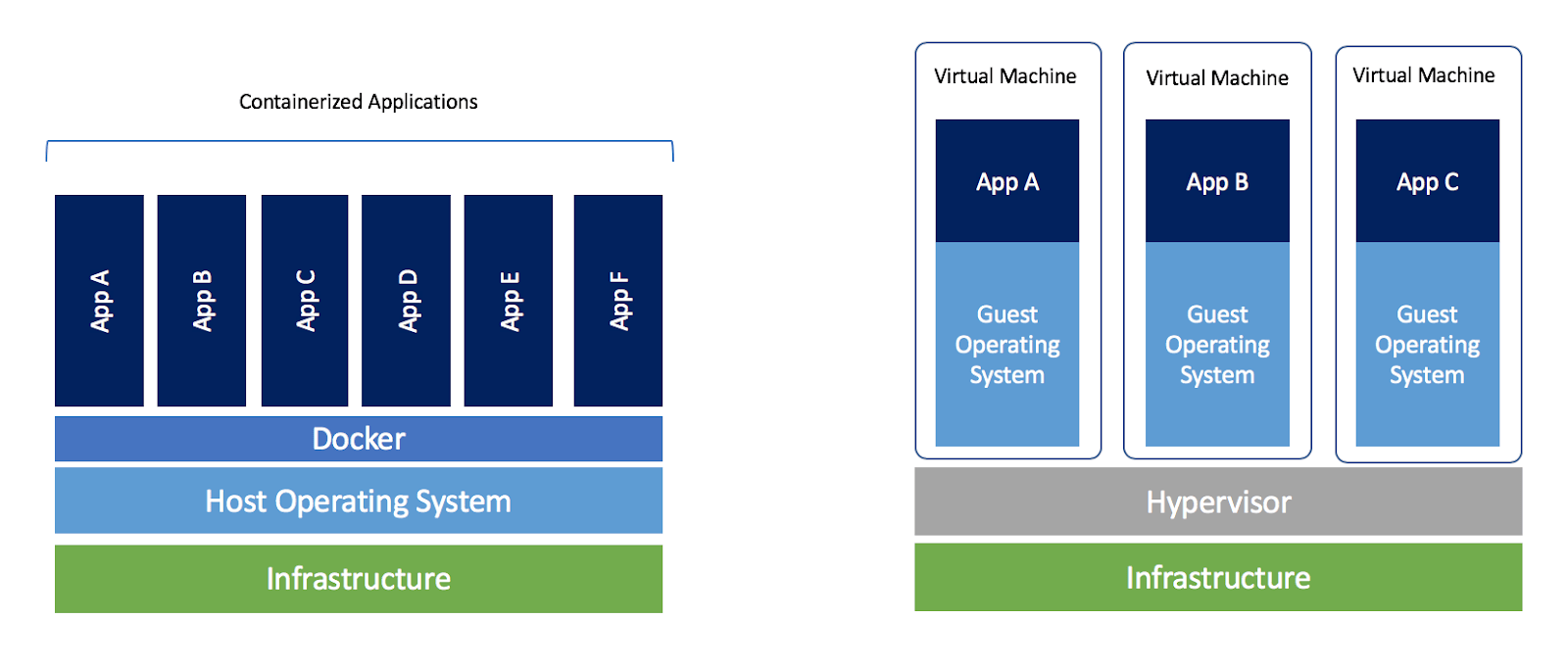
Http verbs :

GET : to display data

POST : to create a new entry in the DB

PUT: to update data in the DB

To deploy these servers , I use the docker desktop which enables me to build and share microservices by creating the Dockerfile . it is use less resources from the PC as shown the differences between docker and vm :



In this project , i create a main image that containe ubuntu , node , packages and the code

Then a create a specific images for each server with the port that the server listen to

Bazar listen to PORT : 8000

Catalog listen to PORT : 3000

Order listen to PORT : 5000

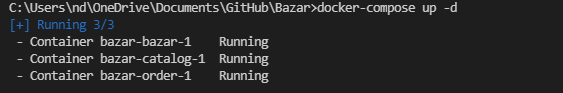
Then I created docker-compose to run all these containers and let them talk to each other on the same network . I use my ip network inside the code instead of the localhost .

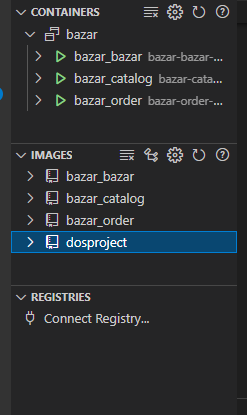
To share the DB files [ BazarBook.json and OrderList.json] I use volume inside my docker-compose , so all containers share these files with the host.

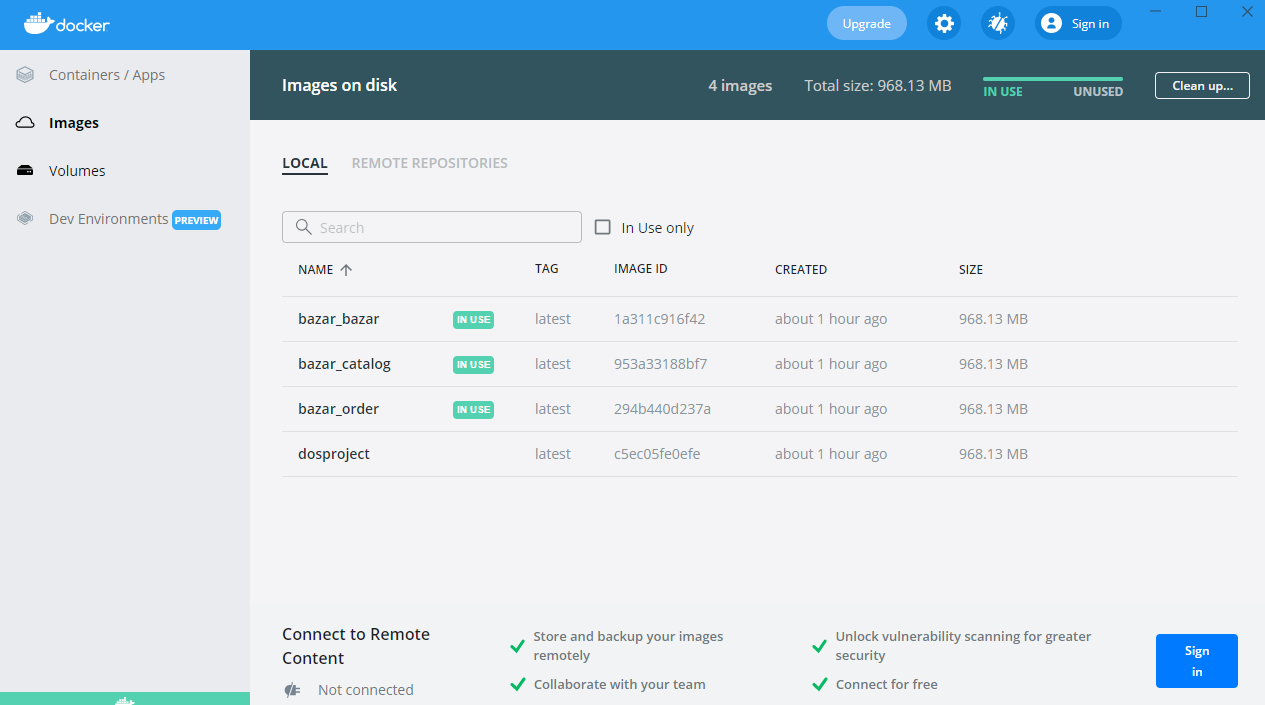
**How my project work :**

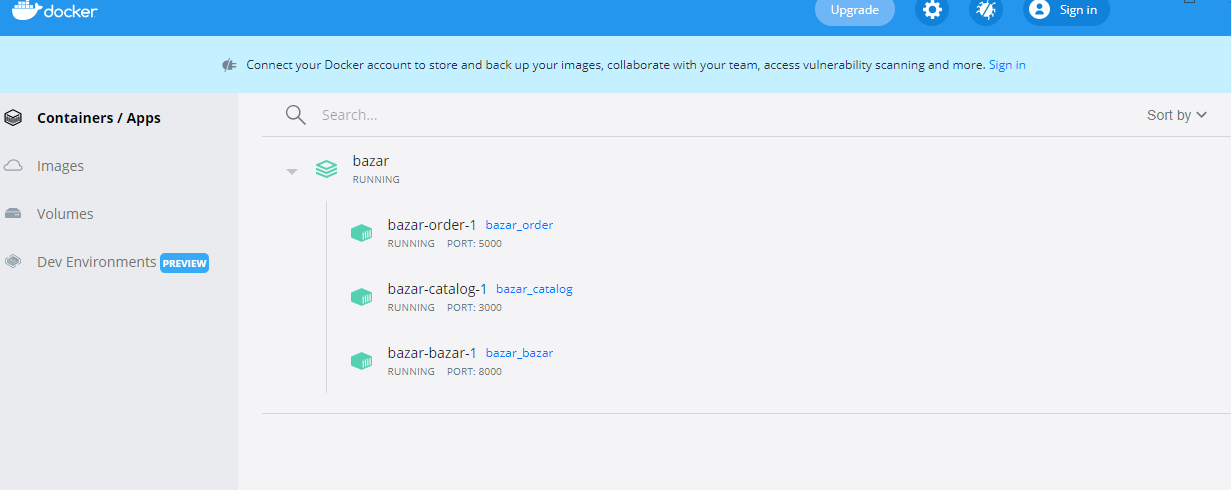
Run a command : docker-compose up -d

Make sure all the containers run :



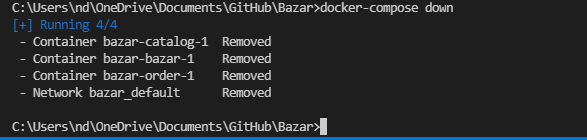


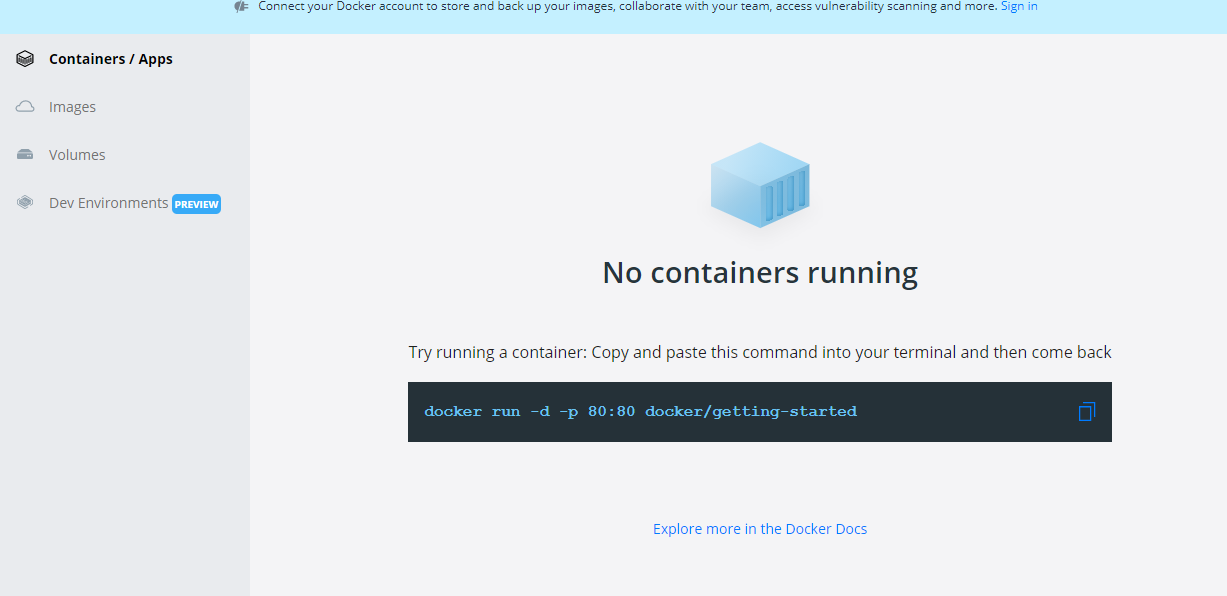




Then use postman to send the requests as shown in the generate output docs

To close the program , use command : docker-compose down





**Improvement :**

* At each endpoint in the servers , I need to write the data from BazarBook.json to take the modified data each time. Instead of duplicating the code , I will separate it into a function and every time I need it I will call the function .
* In the purchase endpoint in the order server , first i send a call to the catalog server to receive info about the book that the user want to buy , then a send another request to the update endpoint to update the number of item in stock , but instead doing this i will just send a request to the update endpoint and inside it i will do all the process.

These two improvements will be achieved in the next part of the project .