

**Computer Engineering Department
College of Engineering**



Report of the first part of the project

Obaida Aws 11923787

Abdallah Adas 11924993


Docker :

In the initial scenario, where servers weren't distributed across multiple containers, the process was sluggish. For instance, it involved sending data from the browser to the client server, then to the order server, followed by communication with the catalog server, and finally waiting for the result to return. However, upon utilizing Docker and distributing them across multiple containers, a significant improvement in speed became evident.

```
docker build -t clientserver .
docker run --name clientserver -v C:\Users\hp\PycharmProjects\DOS:/home/ -p 5500:5500 -it clientserver sh

docker build -t order .
docker run --name order -v C:\Users\hp\PycharmProjects\DOS:/home/ -p 5050:5050 -it order sh

docker build -t catalog .
docker run --name catalog -v C:\Users\hp\PycharmProjects\DOS:/home/ -p 5000:5000 -it catalog sh
```

<input type="checkbox"/>	 clientserver 1702f3a683d6	clientserver	Exited (137)	N/A	5500:5500	24 minutes ago	▶	:	🗑
<input type="checkbox"/>	 order 8e9a670c2a27	order	Exited (137)	N/A	5050:5050	24 minutes ago	▶	:	🗑
<input type="checkbox"/>	 catalog cb6345ef1958	catalog	Exited (137)	N/A	5000:5000	24 minutes ago	▶	:	🗑

❖ What follows are the results of the process of trying out all the possibilities :

1- Search By topic then enter topic name :

```
C:\Users\hp\PycharmProjects\DOS\venv\Scripts\python.exe C:\Users\hp\PycharmProjects\DOS\clientBrowser.py
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
1
Enter Topic name
distributed systems
[
  {
    "id": 1,
    "title": "How to get a good grade in DOS in 40 minutes a day"
  },
  {
    "id": 2,
    "title": "RPCs for Noobs"
  }
]
```

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
1
Enter Topic name
undergraduate school
[
  {
    "id": 3,
    "title": "Xen and the Art of Surviving Undergraduate School"
  },
  {
    "id": 4,
    "title": "Cooking for the Impatient Undergrad"
  }
]
```

2- Search By Id :

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
2
Enter Id of book
1
{
  "price": 30,
  "quantity": 12,
  "title": "How to get a good grade in DOS in 40 minutes a day"
}
```

Enter the number of Operation:

1 --> Search by Topic

2--> Search by Id

3-->Purchase

4--> Exit

2

Enter Id of book

2

{

"price": 40,

"quantity": 19,

"title": "RPCs for Noobs"

}

Enter the number of Operation:

1 --> Search by Topic

2--> Search by Id

3-->Purchase

4--> Exit

2

Enter Id of book

3

{

"price": 30,

"quantity": 29,

"title": "Xen and the Art of Surviving Undergraduate School"

}

Enter the number of Operation:

1 --> Search by Topic

2--> Search by Id

3-->Purchase

4--> Exit

2

Enter Id of book

4

{

"price": 40,

"quantity": 40,

"title": "Cooking for the Impatient Undergrad"

}

3- Purchase :

As we can see the Quantity = 12

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
2
Enter Id of book
1
{
  "price": 30,
  "quantity": 12,
  "title": "How to get a good grade in DOS in 40 minutes a day"
}
```

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
3
Enter Id of book
1
{
  "msg": "The book was purchased successfully"
}
```

Then, after successfully completing the purchase process, it became = 11

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
2
Enter Id of book
1
{
  "price": 30,
  "quantity": 11,
  "title": "How to get a good grade in DOS in 40 minutes a day"
}
```

Here we note that quantity = 0

```
Enter the number of Operation:
1 --> Search by Topic
2--> Search by Id
3-->Purchase
4--> Exit
2
Enter Id of book
2
{
  "price": 40,
  "quantity": 0,
  "title": "RPCs for Noobs"
}
```

Therefore, the purchase will not be completed

```
Enter the number of Operation:
 1 --> Search by Topic
 2--> Search by Id
 3-->Purchase
 4--> Exit
3
Enter Id of book
2
{
  "error": "can not purchase this book because it out of stock."
}
```

In the picture below, the results of the operations we performed are printed in a file

```
{
  "price": 40,
  "quantity": 40,
  "title": "Cooking for the Impatient Undergrad"
}

({'error': 'Failed to fetch data from the API'}, 500)
http://172.17.0.3:5050/purchase/1
{
  "msg": "The book was purchased successfully"
}

http://172.17.0.4:5000/info/1
{
  "price": 30,
  "quantity": 11,
  "title": "How to get a good grade in DOS in 40 minutes a day"
}

http://172.17.0.4:5000/info/2
{
  "price": 40,
  "quantity": 19,
  "title": "RPCs for Noobs"
}

http://172.17.0.4:5000/info/2
{
  "price": 40,
  "quantity": 0,
  "title": "RPCs for Noobs"
}

http://172.17.0.3:5050/purchase/2
{
  "error": "can not purchase this book because it out of stock."
}
```

Order Server code :

```
order.py x
1 from flask import Flask, Response, jsonify
2 import sqlite3
3 import requests
4
5 app = Flask(__name__)
6 conn = sqlite3.connect("my_database.db", check_same_thread=False)
7 cursor = conn.cursor()
8
9
10 @app.route('/purchase/<itemNumber>', methods=['PUT'])
11 def queryPurchase(itemNumber):
12     api_url = 'http://172.17.0.4:5000/info/'+itemNumber
13     response = requests.get(api_url)
14     if response.status_code == 200:
15         data = response.json()
16         if data['quantity'] >= 1:
17             api_url = 'http://172.17.0.4:5000/update/' + itemNumber
18             response = requests.put(api_url)
19             print(response.text)
20             return response.json()
21         else:
22             print("{'error': 'can not purchase this book because it out of stock.'}")
23             return jsonify(
24                 {'error': 'can not purchase this book because it out of stock.'})
25     elif response.status_code == 404:
26         resource = jsonify({"error": "book not found"}, 404)
27         resource.status_code = 404
28         print(response.text)
29         return resource
30     else:
31         print("{'error': 'Failed to fetch data from the API'}", 500)
32         return jsonify({'error': 'Failed to fetch data from the API'}, 500)
33
34
35 if __name__ == '__main__':
36     app.run('0.0.0.0', 5050, debug=True)
37
```


Client Server Code :

```
clientServer.py X
1 from flask import Flask, Response, jsonify
2 import requests
3
4
5 app = Flask(__name__)
6
7 catalogIpPort = "172.17.0.4:5000"
8 orderIpPort = "172.17.0.3:5050"
9
10
11 Abdallah Adas
12 @app.route('/client/info/<itemNumber>')
13 def clientInfo(itemNumber):
14     api_url = 'http://' + catalogIpPort + '/info/' + itemNumber
15     response = requests.get(api_url)
16     if response.status_code == 200:
17         with open("logFile.txt", "a") as file:
18             file.write(api_url + "\n")
19             file.write(response.text + "\n")
20         return response.json()
21     elif response.status_code == 404:
22         resource = jsonify({"error": "book not found"}, 404)
23         resource.status_code = 404
24         with open("logFile.txt", "a") as file:
25             file.write(api_url + "\n")
26             file.write(response.text + "\n")
27         return resource
28     else:
29         with open("logFile.txt", "a") as file:
30             file.write("{ 'error': 'Failed to fetch data from the API', 500 }\n")
31         return jsonify({'error': 'Failed to fetch data from the API'}, 500)
32
```

```
clientServer.py x
Abdallah Adas
33 @app.route('/client/search/<topic>')
34 def clientSearch(topic):
35     api_url = 'http://' + catalogIpPort + '/search/' + topic
36     response = requests.get(api_url)
37     if response.status_code == 200:
38         with open("logFile.txt", "a") as file:
39             file.write(api_url + "\n")
40             file.write(response.text + "\n")
41         return response.json()
42     elif response.status_code == 404:
43         resource = jsonify({"error": "there is no books belong this topic"}, 404)
44         resource.status_code = 404
45         with open("logFile.txt", "a") as file:
46             file.write(api_url + "\n")
47             file.write(response.text + "\n")
48         return resource
49     else:
50         with open("logFile.txt", "a") as file:
51             file.write("({'error': 'Failed to fetch data from the API'}, 500)\n")
52         return jsonify({'error': 'Failed to fetch data from the API'}, 500)
```

```
clientServer.py x
Abdallah Adas
53
54
55 @app.route('/client/purchase/<itemNumber>', methods=['PUT'])
56 def clientPurchase(itemNumber):
57     api_url = 'http://' + orderIpPort + '/purchase/' + itemNumber
58     response = requests.put(api_url)
59     if response.status_code == 200:
60         with open("logFile.txt", "a") as file:
61             file.write(api_url + "\n")
62             file.write(response.text + "\n")
63         return response.json()
64     elif response.status_code == 404:
65         resource = jsonify({"error": "book not found"}, 404)
66         resource.status_code = 404
67         print(response.text)
68         return resource
69     else:
70         with open("logFile.txt", "a") as file:
71             file.write("({'error': 'Failed to fetch data from the API'}, 500)\n")
72         return jsonify({'error': 'Failed to fetch data from the API'}, 500)
73
74
75 if __name__ == '__main__':
76     app.run('0.0.0.0', 5500, debug=True)
```

Client Browser Code :

```
clientServer.py x clientBrowser.py x
1 from flask import Flask, Response, jsonify
2 import sqlite3
3 import requests
4
5
6 app = Flask(__name__)
7 conn = sqlite3.connect("my_database.db", check_same_thread=False)
8 cursor = conn.cursor()
9
10
11 clientIpPort = "localhost:5500"#input("Enter the IP:Port for the Front-end Server Ex:localhost:5500\n")
12 while 1:
13     # Performs a request towards the ClientServer based on the Input
14     UserInput = input("Enter the number of Operation:\n 1 --> Search by Topic \n 2--> Search by Id \n 3-->Purchase \n 4--> Exit \n")
15     if UserInput == "1":
16         topic = input("Enter Topic name\n")
17         response = requests.get("http://" + clientIpPort + "/client/search/" + topic)
18         print(response.text)
19
20     elif UserInput == "2":
21         ID = input("Enter Id of book\n")
22         response = requests.get("http://" + clientIpPort + "/client/info/" + ID)
23         print(response.text)
24
25     elif UserInput == "3":
26         ID = input("Enter Id of book\n")
27         response = requests.put("http://" + clientIpPort + "/client/purchase/" + ID)
28         print(response.text)
29
30     elif UserInput == "4":
31         break
32
33     else:
34         print("Wrong Input!, Try again")
35
```

Catalog Server Code :

```
catalog.py x
1 from flask import Flask, Response, jsonify
2 import sqlite3
3
4 app = Flask(__name__)
5 conn = sqlite3.connect("my_database.db", check_same_thread=False)
6 cursor = conn.cursor()
7
8
9 Abdullah Adas
10 @app.route('/info/<itemNumber>')
11 def queryInfo(itemNumber):
12     cursor.execute("SELECT title, quantity, price FROM book WHERE id = ?", (itemNumber,))
13     row = cursor.fetchone()
14     print(row)
15     if row:
16         book_data = {
17             "title": row[0],
18             "quantity": row[1],
19             "price": row[2]
20         }
21         print(book_data)
22         return jsonify(book_data)
23     else:
24         msg = jsonify({"error": "book not found"}, 404)
25         msg.status_code = 404
26         print(msg)
27         return msg
28
```

```
catalog.py x
27
28
    Abdallah Adas
29 @app.route('/search/<topic>')
30 def querySearch(topic):
31     cursor.execute("SELECT id, title FROM book WHERE topic = ?", (topic,))
32     rows = cursor.fetchall()
33     print(rows)
34     user_list = []
35     if len(rows) == 0:
36         msg = jsonify({"error": "there is no books belong this topic"}, 404)
37         msg.status_code = 404
38         print(msg)
39         return msg
40     else:
41         for row in rows:
42             book_data = {
43                 "id": row[0],
44                 "title": row[1]
45             }
46             user_list.append(book_data)
47         print(user_list)
48         return jsonify(user_list)
49
```

```
catalog.py x
50
    Abdallah Adas
51 @app.route('/update/<itemNumber>', methods=['PUT'])
52 def queryUpdate(itemNumber):
53     cursor.execute("SELECT title, quantity, price FROM book WHERE id = ?", (itemNumber,))
54     row = cursor.fetchone()
55     if row[1] >= 1:
56         cursor.execute("UPDATE book set quantity=quantity-1 WHERE id = ? ", (itemNumber,))
57         conn.commit()
58         print({"msg": 'The book was purchased successfully'})
59         return jsonify(
60             {'msg': 'The book was purchased successfully'})
61     else:
62         print({"error": 'can not purchase this book because it out of stock.'})
63         return jsonify(
64             {'error': 'can not purchase this book because it out of stock.'})
65
66
67 if __name__ == '__main__':
68     app.run("0.0.0.0", 5000, debug=True)
69
```