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
```

clientserver 1702f3a683d6	clientserver	Exited (137)	N/A	5500:5500	24 minutes ago		
order 8e9a670c2a27	order	Exited (137)	N/A	5050:5050	24 minutes ago		
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 :

```python
from flask import Flask, Response, jsonify
import sqlite3
import requests

app = Flask(__name__)
conn = sqlite3.connect("my_database.db", check_same_thread=False)
cursor = conn.cursor()


Abdallah Adas *
@app.route('/purchase/<itemNumber>', methods=['PUT'])
def queryPurchase(itemNumber):
    api_url = 'http://172.17.0.4:5000/info/'+itemNumber
    response = requests.get(api_url)
    if response.status_code == 200:
        data = response.json()
        if data['quantity'] >= 1:
            api_url = 'http://172.17.0.4:5000/update/' + itemNumber
            response = requests.put(api_url)
            print(response.text)
            return response.json()
        else:
            print("{'error': 'can not purchase this book because it out of stock.'}")
            return jsonify(
                {'error': 'can not purchase this book because it out of stock.'})
    elif response.status_code == 404:
        resource = jsonify({"error": "book not found"}, 404)
        resource.status_code = 404
        print(response.text)
        return resource
    else:
        print("({'error': 'Failed to fetch data from the API'}, 500)")
        return jsonify({'error': 'Failed to fetch data from the API'}, 500)


if __name__ == '__main__':
    app.run('0.0.0.0', 5050, debug=True)
```

Client Server Code :

clientServer.py

```python
from flask import Flask, Response, jsonify
import requests


app = Flask(__name__)

catalogIpPort = "172.17.0.4:5000"
orderIpPort = "172.17.0.3:5050"


Abdallah Adas
@app.route('/client/info/<itemNumber>')
def clientInfo(itemNumber):
    api_url = 'http://'+catalogIpPort+'/info/'+itemNumber
    response = requests.get(api_url)
    if response.status_code == 200:
        with open("logFile.txt", "a") as file:
            file.write(api_url+"\n")
            file.write(response.text+"\n")
        return response.json()
    elif response.status_code == 404:
        resource = jsonify({"error": "book not found"}, 404)
        resource.status_code = 404
        with open("logFile.txt", "a") as file:
            file.write(api_url+"\n")
            file.write(response.text+"\n")
        return resource
    else:
        with open("logFile.txt", "a") as file:
            file.write("({'error': 'Failed to fetch data from the API'}, 500)\n")
        return jsonify({'error': 'Failed to fetch data from the API'}, 500)
```

clientServer.py

Abdallah Adas

```python
@app.route('/client/search/<topic>')
def clientSearch(topic):
    api_url = 'http://'+catalogIpPort+'/search/'+topic
    response = requests.get(api_url)
    if response.status_code == 200:
        with open("logFile.txt", "a") as file:
            file.write(api_url+"\n")
            file.write(response.text+"\n")
        return response.json()
    elif response.status_code == 404:
        resource = jsonify({"error": "there is no books belong this topic"}, 404)
        resource.status_code = 404
        with open("logFile.txt", "a") as file:
            file.write(api_url+"\n")
            file.write(response.text+"\n")
        return resource
    else:
        with open("logFile.txt", "a") as file:
            file.write("({'error': 'Failed to fetch data from the API'}, 500)\n")
        return jsonify({'error': 'Failed to fetch data from the API'}, 500)
```

clientServer.py

Abdallah Adas

```python
@app.route('/client/purchase/<itemNumber>', methods=['PUT'])
def clientPurchase(itemNumber):
    api_url = 'http://'+orderIpPort+'/purchase/'+itemNumber
    response = requests.put(api_url)
    if response.status_code == 200:
        with open("logFile.txt", "a") as file:
            file.write(api_url+"\n")
            file.write(response.text+"\n")
        return response.json()
    elif response.status_code == 404:
        resource = jsonify({"error": "book not found"}, 404)
        resource.status_code = 404
        print(response.text)
        return resource
    else:
        with open("logFile.txt", "a") as file:
            file.write("({'error': 'Failed to fetch data from the API'}, 500)\n")
        return jsonify({'error': 'Failed to fetch data from the API'}, 500)


if __name__ == '__main__':
    app.run('0.0.0.0', 5500, debug=True)
```

Client Browser Code :

clientServer.py clientBrowser.py

```python
from flask import Flask,Response, jsonify
import sqlite3
import requests


app = Flask(__name__)
conn = sqlite3.connect("my_database.db", check_same_thread=False)
cursor = conn.cursor()


clientIpPort = "localhost:5500"#input("Enter the IP:Port for the Front-end Server Ex:localhost:5500\n")
while 1:
    # Performs a request towards the ClientServer based on the Input
    UserInput = input("Enter the number of Operation:\n 1 --> Search by Topic \n 2--> Search by Id \n 3-->Purchase \n 4--> Exit \n")
    if UserInput == "1":
        topic = input("Enter Topic name\n")
        response = requests.get("http://"+clientIpPort + "/client/search/"+topic)
        print(response.text)

    elif UserInput == "2":
        ID = input("Enter Id of book\n")
        response = requests.get("http://"+clientIpPort + "/client/info/"+ID)
        print(response.text)

    elif UserInput == "3":
        ID = input("Enter Id of book\n")
        response = requests.put("http://"+clientIpPort + "/client/purchase/"+ID)
        print(response.text)

    elif UserInput == "4":
        break

    else:
        print("Wrong Input!, Try again")
```

Catalog Server Code :

catalog.py

```python
from flask import Flask,Response, jsonify
import sqlite3

app = Flask(__name__)
conn = sqlite3.connect("my_database.db", check_same_thread=False)
cursor = conn.cursor()


Abdallah Adas
@app.route('/info/<itemNumber>')
def queryInfo(itemNumber):
    cursor.execute("SELECT title, quantity, price FROM book WHERE id = ?", (itemNumber,))
    row = cursor.fetchone()
    print(row)
    if row:
        book_data = {
            "title": row[0],
            "quantity": row[1],
            "price": row[2]
        }
        print(book_data)
        return jsonify(book_data)
    else:
        msg = jsonify({"error": "book not found"}, 404)
        msg.status_code = 404
        print(msg)
        return msg
```

catalog.py

Abdallah Adas

```python
@app.route('/search/<topic>')
def querySearch(topic):
    cursor.execute("SELECT id, title FROM book WHERE topic = ?", (topic,))
    rows = cursor.fetchall()
    print(rows)
    user_list = []
    if len(rows) == 0:
        msg = jsonify({"error": "there is no books belong this topic"}, 404)
        msg.status_code = 404
        print(msg)
        return msg
    else:
        for row in rows:
            book_data = {
                "id": row[0],
                "title": row[1]
            }
            user_list.append(book_data)
        print(user_list)
        return jsonify(user_list)
```

catalog.py

Abdallah Adas

```python
@app.route('/update/<itemNumber>', methods=['PUT'])
def queryUpdate(itemNumber):
    cursor.execute("SELECT title, quantity, price FROM book WHERE id = ?", (itemNumber,))
    row = cursor.fetchone()
    if row[1] >= 1:
        cursor.execute("UPDATE book set quantity=quantity-1 WHERE id = ? ", (itemNumber,))
        conn.commit()
        print("{'msg': 'The book was purchased successfully'}")
        return jsonify(
            {'msg': 'The book was purchased successfully'})
    else:
        print("{'error': 'can not purchase this book because it out of stock.'}")
        return jsonify(
            {'error': 'can not purchase this book because it out of stock.'})


if __name__ == '__main__':
    app.run("0.0.0.0",5000,debug=True)
```