



**An-Najah National University  
Faculty of Engineering and IT**

جامعة النجاح الوطنية  
جامعة الهندسة وتكنولوجيا المعلومات

Distributed Operating Systems

10636456

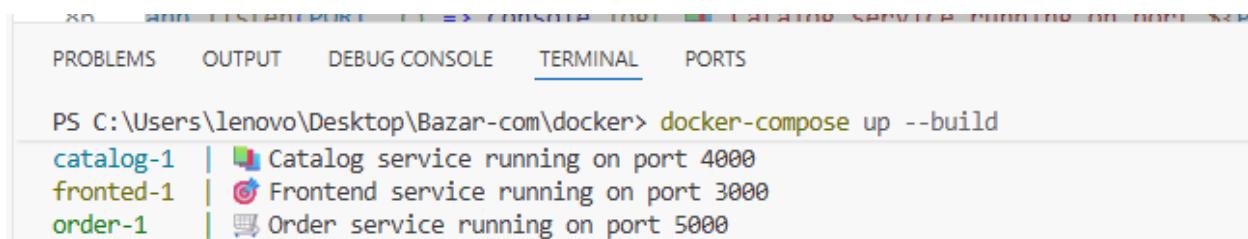
1<sup>st</sup> Semester | 2025-2026

Bazar-com Project | Part 1 | Results

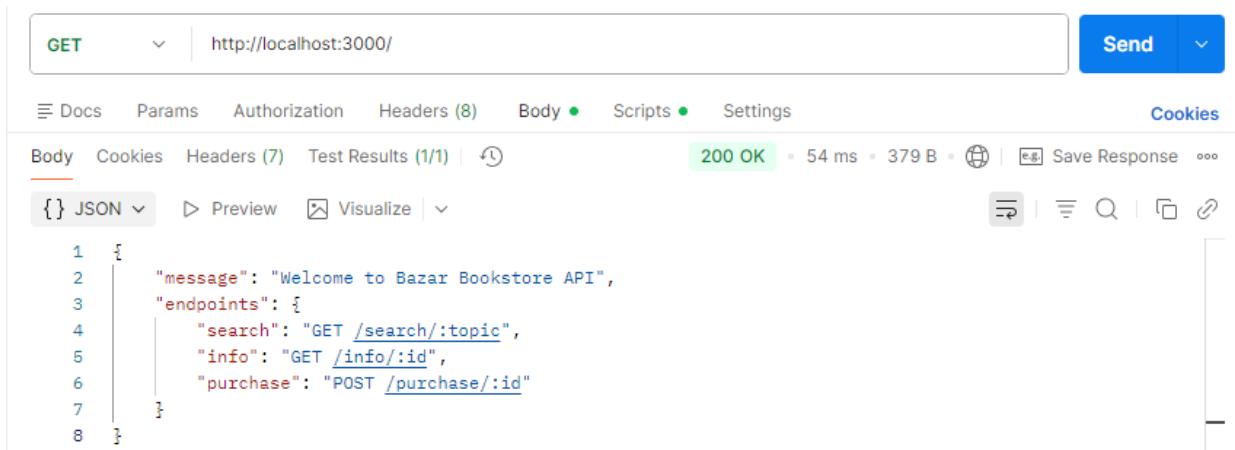
Jana Moe'n Tahayni

12220650

Once the system is running:



```
PS C:\Users\lenovo\Desktop\Bazar-com\docker> docker-compose up --build
catalog-1 | Catalog service running on port 4000
fronted-1 | Frontend service running on port 3000
order-1 | Order service running on port 5000
```

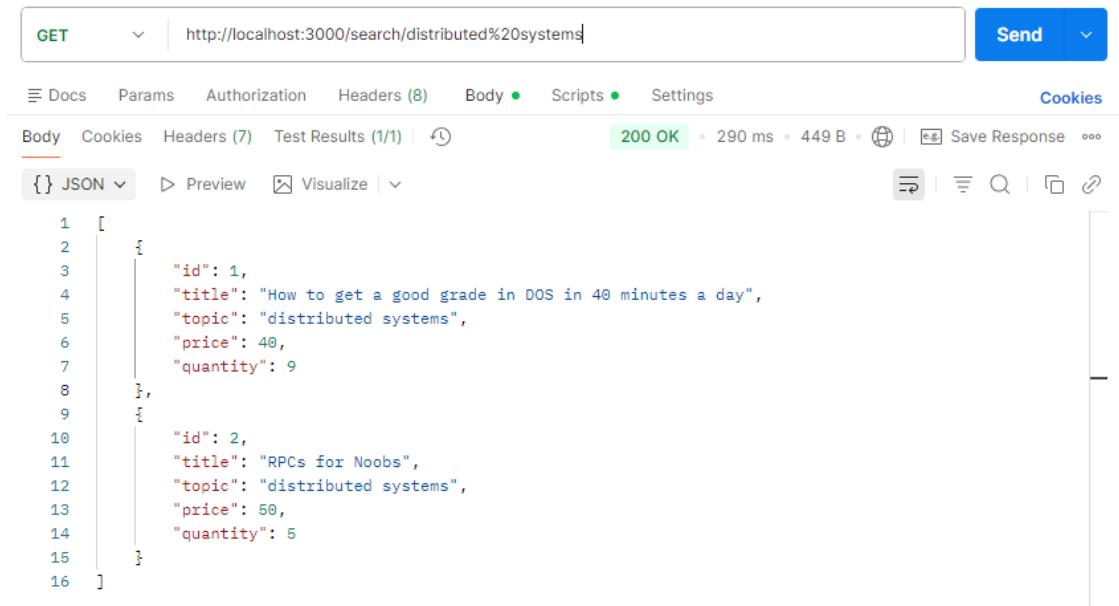


GET http://localhost:3000/

Body { } JSON

```
1 {
2   "message": "Welcome to Bazar Bookstore API",
3   "endpoints": {
4     "search": "GET /search/:topic",
5     "info": "GET /info/:id",
6     "purchase": "POST /purchase/:id"
7   }
8 }
```

Searching for books with topic “distributed systems”:



GET http://localhost:3000/search/distributed%20systems

Body { } JSON

```
1 [
2   {
3     "id": 1,
4     "title": "How to get a good grade in DOS in 40 minutes a day",
5     "topic": "distributed systems",
6     "price": 40,
7     "quantity": 9
8   },
9   {
10     "id": 2,
11     "title": "RPCs for Noobs",
12     "topic": "distributed systems",
13     "price": 50,
14     "quantity": 5
15   }
16 ]
```

```
fronted-1 | 🔎 Frontend: Search request for topic: distributed systems
catalog-1 | 📚 Search for "distributed systems": Found 2 books
fronted-1 | ✅ Frontend: Search successful, found 2 books
```

Searching for books with topic “undergraduate school”:

The screenshot shows a Postman request for a GET operation at `http://localhost:3000/search/undergraduate%20school`. The response status is 200 OK, with a duration of 27 ms and a size of 471 B. The response body is a JSON array containing two book objects:

```
1 [ 
2   {
3     "id": 3,
4     "title": "Xen and the Art of Surviving Undergraduate School",
5     "topic": "undergraduate school",
6     "price": 35,
7     "quantity": 5
8   },
9   {
10    "id": 4,
11    "title": "Cooking for the Impatient Undergrad",
12    "topic": "undergraduate school",
13    "price": 25,
14    "quantity": 4
15  }
16 ]
```

```
fronted-1 | 🔎 Frontend: Search request for topic: undergraduate school
catalog-1 | 📚 Search for "undergraduate school": Found 2 books
fronted-1 | ✅ Frontend: Search successful, found 2 books
```

Get information about book with id=3:

The screenshot shows a Postman request for a GET operation at `http://localhost:3000/info/3`. The response status is 200 OK, with a duration of 27 ms and a size of 359 B. The response body is a JSON object representing a single book:

```
1 {
2   "id": 3,
3   "title": "Xen and the Art of Surviving Undergraduate School",
4   "topic": "undergraduate school",
5   "price": 35,
6   "quantity": 5
7 }
```

```
fronted-1 | iFrontend: Info request for book ID: 3
fronted-1 | ✓ Frontend: Info retrieved for: "Xen and the Art of Surviving Undergraduate School"
catalog-1 | iInfo request for book ID: 3 - "Xen and the Art of Surviving Undergraduate School"
```

Purchase for book with not available id:

The screenshot shows a POST request to `http://localhost:3000/purchase/5`. The response status is 404 Not Found. The response body is a JSON object with one key, `error`, containing the value `"Book not found"`.

```
{ } JSON ▾ 404 Not Found 184 ms 268 B
[] 1
2   "error": "Book not found"
3 }
```

```
fronted-1 | 📺 Frontend: Purchase request for book ID: 5
order-1 | ✗ Purchase error: Request failed with status code 404
fronted-1 | ✗ Frontend: Purchase error: Request failed with status code 404
```

Purchase for book with id=4:

The screenshot shows a POST request to `http://localhost:3000/purchase/4`. The response status is 200 OK. The response body is a JSON object containing a message, an order object, and a remaining quantity.

```
POST http://localhost:3000/purchase/4 200 OK 56 ms 442 B
[] 1
2   "message": "Purchase successful!",
3   "order": {
4     "id": 10,
5     "bookId": 4,
6     "bookTitle": "Cooking for the Impatient Undergrad",
7     "price": 25,
8     "status": "completed",
9     "timestamp": "2025-11-29T18:31:29.218Z"
10   },
11   "remainingQuantity": 3
12 }
```

```
fronted-1 | 📱 Frontend: Purchase request for book ID: 4
catalog-1 | 🗂️ Info request for book ID: 4 - "Cooking for the Impatient Undergrad"
order-1 | 📱 Purchase request for: "Cooking for the Impatient Undergrad" - Current quantity: 4

catalog-1 | 📱 Updated book ID: 4 - Quantity: 3, Price: 25
fronted-1 | ✅ Frontend: Purchase successful - Order ID: 10
order-1 | ✅ Purchase successful! Order ID: 10, Remaining quantity: 3
```

Purchase a book out of stock:

The screenshot shows a Postman request for a 'Purchase' endpoint. The request method is POST, and the URL is `http://localhost:3000/purchase/4`. The response status is 400 Bad Request, indicating an error. The response body is a JSON object with a single key 'error': "Book out of stock".

```
POST http://localhost:3000/purchase/4
400 Bad Request
{"error": "Book out of stock"}
```

Below the request, a log of events is displayed:

```
fronted-1 | 📱 Frontend: Purchase request for book ID: 4
order-1 | 📱 Purchase request for: "Cooking for the Impatient Undergrad" - Current quantity: 0
catalog-1 | 🗂️ Info request for book ID: 4 - "Cooking for the Impatient Undergrad"

fronted-1 | ❌ Frontend: Purchase error: Request failed with status code 400
```