

An-Najah National University
Computer Engineering
Distributed Operation System

BAZAR.COM part2

Results report and screen shots

Students

Malik Ali 12114096

Nassar Harashi 12144079

Dr. Samer Arandi

1. Start services:

Services started using `docker-compose up --build -d`. Shown: container names, state, ports (frontend:3002, catalog:3001, order:3000).

```
PS C:\MyDocuments\university-courses\Dos\bazar-com> docker-compose up --build -d
=> => exporting layers
=> => exporting manifest sha256:ec7477348c44316608793bd12ddae77c2515d016e297d091f20408c145637
=> => exporting config sha256:8bba75aae82beff4dff96ecefcd3e36f588daefad02a9686dea48b7c4b33c6dd
=> => exporting attestation manifest sha256:63914f2fd9e71960f6364501e05d92482c253a6d2440379f
=> => exporting manifest list sha256:468b4b26aedas5c04b2f7fa19ffa4446d587584da6001b7890c55fb5
=> => naming to docker.io/library/bazar-com-catalog:latest
=> => unpacking to docker.io/library/bazar-com-catalog:latest
=> [catalog] resolving provenance for metadata file
=> [frontend 5/5] COPY .
=> [frontend] exporting to image
=> => exporting layers
=> => exporting manifest sha256:c4469d8bd90b52a36ed2b0ee8614382f9c21ce6db6dacf00aadd3a926434c
=> => exporting config sha256:5a44877f78d1bd9fffc1a2799c4bc0bd5b7d1bff95f5ecc49ec009d5817c27
=> => exporting attestation manifest sha256:d3e33273c6369857f92096778227792d816f46d9d9c8806
=> => exporting manifest list sha256:c4b1d655e34bbd99749ed8813bd2fad08b2f14829572763a0cf7cf
=> => naming to docker.io/library/bazar-com-frontend:latest
=> => unpacking to docker.io/library/bazar-com-frontend:latest
=> [order 5/5] COPY .
=> [order] exporting to image
=> => exporting layers
=> => exporting manifest sha256:dcbfb1dd49bb68d7a57ad0a4bb40194848221763fe80005a6a11afa188d3
=> => exporting config sha256:f95a4c8f7c78455f13a81799f03946064d1a1d3c15ac35840251dd9536eb85e
=> => exporting attestation manifest sha256:12f16f73ac7090083f9eb1fa4bd2d3e0f7f93ed2027c535dd
=> => exporting manifest list sha256:3777e1eb19bf7097b9f70ec7a1261178de68bf160706231ea1c887d
=> => naming to docker.io/library/bazar-com-order:latest
=> => unpacking to docker.io/library/bazar-com-order:latest
=> [frontend] resolving provenance for metadata file
=> [order] resolving provenance for metadata file
[+] Running 7/7
✓ bazar-com-catalog      Built
✓ bazar-com-order       Built
✓ bazar-com-frontend    Built
✓ Network bazar-network Created
✓ Container catalog-service Started
✓ Container order-service Started
✓ Container frontend-service Started
```

2. Frontend Health:

Health check: reports frontend status and dependency checks for catalog and order.

Note [status](#) (healthy/degraded) and service URLs."

```

PS C:\MyDocuments\universtiy-courses\Dos\bazar-com> Invoke-RestMethod -Uri "http://localhost:3002/health"

status      : healthy
timestamp   : 2025-12-21T18:27:19.487Z
uptime      : 337.218445984
environment : production
serviceId   : frontend-1766341300935
services    : @{catalog=; order=}
cache       : @{keys=0; stats=}

PS C:\MyDocuments\universtiy-courses\Dos\bazar-com>

```

3. Cache stats before tests:

Cache initial state (keys, hits/misses). Use as baseline for cache performance comparison.

```

● PS C:\MyDocuments\universtiy-courses\Dos\bazar-com> Invoke-RestMethod -Uri " http://localhost:3002/cache-stats"

  keys stats                                count serviceId
  ----- -----
  {}    @{hits=0; misses=0; keys=0; ksize=0; vsize=0}    0 frontend-1766341300935

○ PS C:\MyDocuments\universtiy-courses\Dos\bazar-com>

```

4. Search endpoint — show MISS then HIT:

Search test: compare MISS vs HIT. Record counts and response source fields (e.g., [source: catalog-service](#) vs [source: cache](#)). Note time differences if you measured."

Invalidate cache:

Command :

```

Invoke-RestMethod `

-Uri "http://localhost:3002/invalidate-cache" `

-Method POST `

-Headers @{ "Content-Type" = "application/json" } `

-Body (@{ bookId = "1" } | ConvertTo-Json)

```

```

PS C:\MyDocuments\universtiy-courses\Dos\bazar-com> Invoke-RestMethod
>> -Uri "http://localhost:3002/invalidate-cache"
>> -Method POST
>> -Headers @{
    "Content-Type" = "application/json"
}
● >> -Body (@{ bookId = "1" } | ConvertTo-Json)
>>

status message           serviceId
----- ----- 
success Cache invalidated frontend-1766341300935

```

PS C:\MyDocuments\universtiy-courses\Dos\bazar-com>

Request 1(cache miss):

The screenshot shows the Postman interface with the following details:

- Collection:** BAZAR-COM / frontend-server
- Request Type:** GET
- URL:** http://localhost:3002/search/undergraduate%20school
- Headers:** Authorization, Headers (8), Body, Scripts, Settings
- Query Params:** Key, Value, Description
- Body:** JSON, Preview, Visualize
- Response:**
 - Status: 200 OK
 - Time: 32 ms
 - Size: 458 B
 - Content: A JSON response object containing success: true, source: catalog-service, count: 2, and data: [{ id: "3", title: "Xen and the Art of Suiviving Undergraduate School" }, { id: "4", title: "Cooking for the Impatient Undergrad" }].

Request 2(cache hit):

The screenshot shows the Postman interface with the following details:

- Left Sidebar:** Collections (api-rest, BAZAR-COM), Environments (History), Flows (Courses-Project, Doslab1, learnNodejsAPIs, simpleProject), and Files (BETA).
- Top Bar:** Overview, GET info, PUT update, POST purchase, GET orders, GET info, POST purchase, GET search.
- Request URL:** BAZAR-COM / frontend-server / search
- Method:** GET
- URL:** http://localhost:3002/search/undergraduate%20school
- Params:** Query Params (Key: Key, Value: Value)
- Body:** JSON (200 OK, 5 ms, 448 B, 8:59 PM, 12/21/2025)
- Response Body:**

```
1 {  
2   "success": true,  
3   "source": "cache",  
4   "count": 2,  
5   "data": [  
6     {  
7       "id": "3",  
8       "title": "Xen and the Art of Surviving Undergraduate School"  
9     },  
10    {  
11      "id": "4",  
12      "title": "Cooking for the Impatient Undergrad"  
13    }  
14  ]  
15 }
```
- Bottom Status Bar:** Cloud View, Find and replace, Console, Weather (58°F, Mostly cloudy), Search, Start Proxy, Cookies, Vault, Runner, Start Proxy, Cookies, Vault, Trash, Help, 8:59 PM, 12/21/2025.

5. Book info /info/1 — MISS then HIT

Invalidate cache :

```
>> -Uri "http://localhost:3002/invalidate-cache" `get a good
>> -Method POST `^
>> -Headers @{ "Content-Type" = "application/json" } `^
● >> -Body (@{ bookId = "1" } | ConvertTo-Json)-com>
```

| status | message | serviceId |
|---------|-------------------|------------------------|
| ----- | ----- | ----- |
| success | Cache invalidated | frontend-1766341300935 |

```
○ PS C:\MyDocuments\universty-courses\Dos\bazar-com>
```

Request 1(miss):

The screenshot shows the Postman application interface. On the left, the sidebar displays collections, environments, and flows. The main workspace shows a request to the '/info' endpoint of the 'BAZAR-COM / frontend-server'. The request method is 'GET', and the URL is 'http://localhost:3002/info/1'. The response status is '200 OK' with a response time of '20 ms' and a size of '407 B'. The response body is a JSON object:

```
{ "success": true, "source": "catalog-service", "data": { "title": "How to get a good grade in DOS in 40 minutes a day", "quantity": 18, "price": 100 } }
```

Request 2(HIT)

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections like 'Collections', 'Environments', 'History', 'Flows', and 'Files'. The main area shows a 'GET' request to 'http://localhost:3002/info/1'. The 'Body' tab displays the JSON response:

```

1 {
2   "success": true,
3   "source": "cache",
4   "data": {
5     "title": "How to get a good grade in DOS in 40 minutes a day",
6     "quantity": 18,
7     "price": 100
8   }
9 }

```

A blue box highlights the 'data' field in the JSON response.

6. Purchase flow (end-to-end) and confirm [orders.csv](#):

End-to-end purchase: frontend -> order -> catalog stock decrement. Include purchase response ([orderId](#), [price](#)) and verify [orders.csv](#) contains the generated order. Note any changes in catalog stock.

Purchase:

```

PS C:\MyDocuments\universtity-courses\Dos\bazar-com> Invoke-RestMethod -Method POST -Uri "http://localhost:3002/purchase
/1" |
>> ConvertTo-Json -Depth 10
{
    "success": true,
    "source": "order-service",
    "data": {
        "message": "Purchase successful",
        "book": "How to get a good grade in DOS in 40 minutes a day",
        "orderId": "1766344524241",
        "price": 100
    }
}

```

Show from host:

```

Windows PowerShell      +  -
"price": 100
}
PS C:\MyDocuments\universtity-courses\Dos\bazar-com> type .\order-server\orders.csv
ORDER_ID,BOOK_ID,TITLE,QUANTITY,TOTAL_PRICE,TIMESTAMP
1762992825675,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:13:45
1762992826985,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:13:46
1762993208881,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:08
1762993239276,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:39
1762993240530,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:40
1762993718017,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:28:38
1762994246614,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:37:26
1762994472206,2,RPCs for Noobs,1,30,2025-11-13 00:41:12
1763823620531,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-22 15:00:20
1763824201412,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-22 15:10:01
1764341576145,2,RPCs for Noobs,1,100,2025-11-28 14:52:56
1764341714504,2,RPCs for Noobs,1,100,2025-11-28 14:55:14
1764341718714,2,RPCs for Noobs,1,100,2025-11-28 14:55:18
1766237349735,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-20 13:29:09
1766237362708,2,RPCs for Noobs,1,100,2025-12-20 13:29:22
1766237387359,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:29:47
1766237914633,2,RPCs for Noobs,1,100,2025-12-20 13:38:34
1766237915171,3,Xen and the Art of Surviving Undergraduate School,1,25,2025-12-20 13:38:35
1766237915698,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:38:35
1766237917886,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-20 13:38:37
1766237918417,2,RPCs for Noobs,1,100,2025-12-20 13:38:38
1766237918970,3,Xen and the Art of Surviving Undergraduate School,1,25,2025-12-20 13:38:39
1766237919547,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:38:39
1766344524241,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-21 19:15:24
PS C:\MyDocuments\universtity-courses\Dos\bazar-com>

```

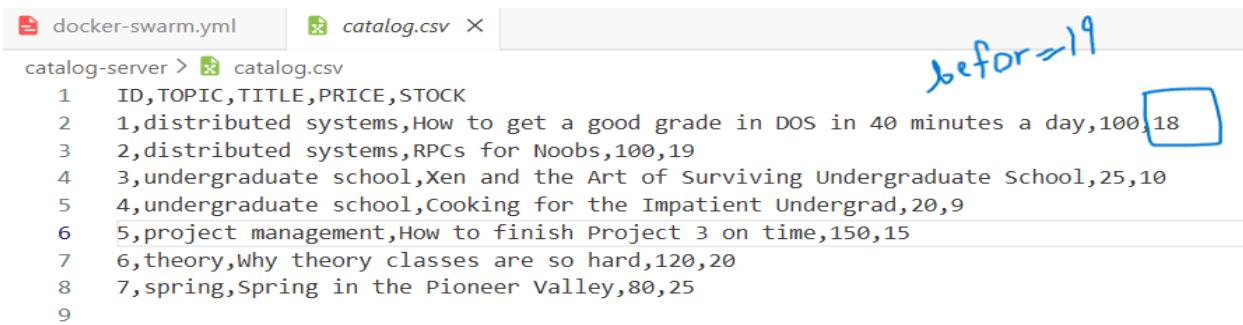
Show from docker container:

```

PS C:\MyDocuments\universtity-courses\Dos\bazar-com> docker-compose exec order cat /app/orders.csv
ORDER_ID,BOOK_ID,TITLE,QUANTITY,TOTAL_PRICE,TIMESTAMP
1762992825675,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:13:45
1762992826985,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:13:46
1762993208881,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:08
1762993239276,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:39
1762993240530,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:20:40
1762993718017,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:28:38
1762994246614,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-13 00:37:26
1762994472206,2,RPCs for Noobs,1,30,2025-11-13 00:41:12
1763823620531,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-22 15:00:20
1763824201412,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-11-22 15:10:01
1764341576145,2,RPCs for Noobs,1,100,2025-11-28 14:52:56
1764341714504,2,RPCs for Noobs,1,100,2025-11-28 14:55:14
1764341718714,2,RPCs for Noobs,1,100,2025-11-28 14:55:18
1766237349735,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-20 13:29:09
1766237362708,2,RPCs for Noobs,1,100,2025-12-20 13:29:22
1766237387359,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:29:47
1766237914633,2,RPCs for Noobs,1,100,2025-12-20 13:38:34
1766237915171,3,Xen and the Art of Surviving Undergraduate School,1,25,2025-12-20 13:38:35
1766237915698,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:38:35
1766237917886,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-20 13:38:37
1766237918417,2,RPCs for Noobs,1,100,2025-12-20 13:38:38
1766237918970,3,Xen and the Art of Surviving Undergraduate School,1,25,2025-12-20 13:38:39
1766237919547,4,Cooking for the Impatient Undergrad,1,20,2025-12-20 13:38:39
1766344524241,1,How to get a good grade in DOS in 40 minutes a day,1,100,2025-12-21 19:15:24
PS C:\MyDocuments\universtity-courses\Dos\bazar-com>

```

The stock in catalog.csv:

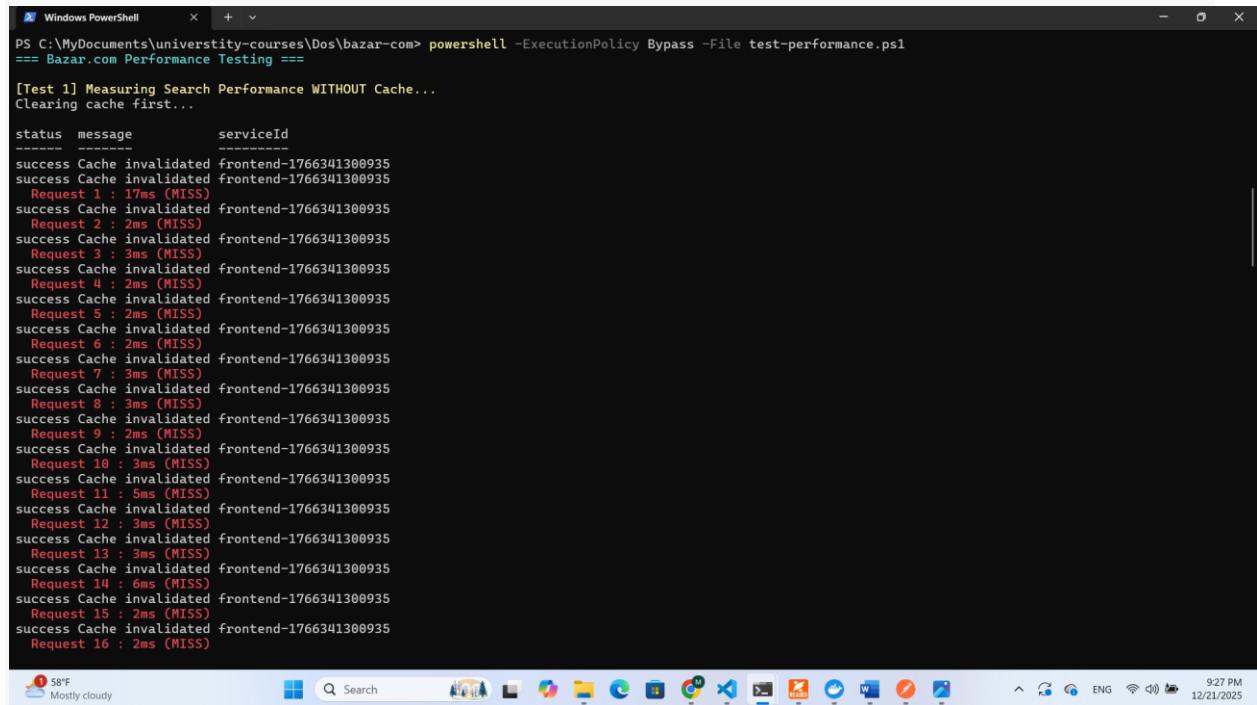


```
docker-swarm.yml  catalog.csv <input>
catalog-server > catalog.csv
1 ID,TOPIC,TITLE,PRICE,STOCK
2 1,distributed systems,How to get a good grade in DOS in 40 minutes a day,100,18
3 2,distributed systems,RPCs for Noobs,100,19
4 3,undergraduate school,Xen and the Art of Surviving Undergraduate School,25,10
5 4,undergraduate school,Cooking for the Impatient Undergrad,20,9
6 5,project management,How to finish Project 3 on time,150,15
7 6,theory,Why theory classes are so hard,120,20
8 7,spring,Spring in the Pioneer Valley,80,25
9
```

7. Run performance script and collect CSV:

Command (PowerShell): [powershell -ExecutionPolicy Bypass -File test-performance.ps1](#)

Performance benchmark results: include averages, medians, min/max for Search (with/without cache), Info (with/without cache), Purchase, and Cache Invalidation. Attach the [performance-results.csv](#) and add short commentary about observed cache improvement percentages.



```
Windows PowerShell
PS C:\MyDocuments\university-courses\Dos\bazar-com> powershell -ExecutionPolicy Bypass -File test-performance.ps1
== Bazar.com Performance Testing ==
[Test 1] Measuring Search Performance WITHOUT Cache...
Clearing cache first...
status message serviceId
---- -- -- -----
success Cache invalidated frontend-1766341300935
success Cache invalidated frontend-1766341300935
    Request 1 : 1ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 2 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 3 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 4 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 5 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 6 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 7 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 8 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 9 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 10 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 11 : 5ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 12 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 13 : 3ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 14 : 6ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 15 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
    Request 16 : 2ms (MISS)
```

```
Windows PowerShell + × - ○ X
success Cache invalidated frontend-1766341300935
Request 17 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
Request 18 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
Request 19 : 2ms (MISS)
success Cache invalidated frontend-1766341300935
Request 20 : 2ms (MISS)

[Test 2] Measuring Search Performance WITH Cache...
Request 1 : 2ms (HIT)
Request 2 : 3ms (HIT)
Request 3 : 2ms (HIT)
Request 4 : 2ms (HIT)
Request 5 : 3ms (HIT)
Request 6 : 2ms (HIT)
Request 7 : 2ms (HIT)
Request 8 : 11ms (HIT)
Request 9 : 2ms (HIT)
Request 10 : 2ms (HIT)
Request 11 : 2ms (HIT)
Request 12 : 2ms (HIT)
Request 13 : 2ms (HIT)
Request 14 : 2ms (HIT)
Request 15 : 3ms (HIT)
Request 16 : 2ms (HIT)
Request 17 : 2ms (HIT)
Request 18 : 2ms (HIT)
Request 19 : 4ms (HIT)
Request 20 : 2ms (HIT)

[Test 3] Measuring Info Performance WITHOUT Cache...
success Cache invalidated frontend-1766341300935
Request 1 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 2 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 3 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 4 : 7ms (MISS)
success Cache invalidated frontend-1766341300935

Windows PowerShell + × - ○ X
Tomorrow's high
Near record 9:27 PM
Search ENG 12/21/2025
```

```
Windows PowerShell + × - ○ X
success Cache invalidated frontend-1766341300935
Request 5 : 8ms (MISS)
success Cache invalidated frontend-1766341300935
Request 6 : 6ms (MISS)
success Cache invalidated frontend-1766341300935
Request 7 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 8 : 11ms (MISS)
success Cache invalidated frontend-1766341300935
Request 9 : 8ms (MISS)
success Cache invalidated frontend-1766341300935
Request 10 : 6ms (MISS)
success Cache invalidated frontend-1766341300935
Request 11 : 5ms (MISS)
success Cache invalidated frontend-1766341300935
Request 12 : 6ms (MISS)
success Cache invalidated frontend-1766341300935
Request 13 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 14 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 15 : 9ms (MISS)
success Cache invalidated frontend-1766341300935
Request 16 : 6ms (MISS)
success Cache invalidated frontend-1766341300935
Request 17 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 18 : 7ms (MISS)
success Cache invalidated frontend-1766341300935
Request 19 : 8ms (MISS)
success Cache invalidated frontend-1766341300935
Request 20 : 7ms (MISS)

[Test 4] Measuring Info Performance WITH Cache...
Request 1 : 3ms (HIT)
Request 2 : 3ms (HIT)
Request 3 : 2ms (HIT)
Request 4 : 2ms (HIT)
Request 5 : 3ms (HIT)
Request 6 : 2ms (HIT)
Request 7 : 2ms (HIT)

Windows PowerShell + × - ○ X
Tomorrow's high
Near record 9:28 PM
Search ENG 12/21/2025
```

```
Windows PowerShell x + -
```

```
Request 8 : 3ms (HIT)
Request 9 : 2ms (HIT)
Request 10 : 2ms (HIT)
Request 11 : 2ms (HIT)
Request 12 : 2ms (HIT)
Request 13 : 2ms (HIT)
Request 14 : 2ms (HIT)
Request 15 : 2ms (HIT)
Request 16 : 2ms (HIT)
Request 17 : 2ms (HIT)
Request 18 : 2ms (HIT)
Request 19 : 2ms (HIT)
Request 20 : 2ms (HIT)

[Test 5] Measuring Purchase + Cache Invalidation Overhead...
Purchase 1 (Book 2): 56ms
Purchase 2 (Book 3): 22ms
Purchase 3 (Book 4): 26ms
Purchase 4 (Book 5): 27ms
Purchase 5 (Book 6): 27ms
Purchase 6 (Book 7): 22ms
Purchase 7 (Book 1): 23ms
Purchase 8 (Book 2): 22ms
Purchase 9 (Book 3): 24ms
Purchase 10 (Book 4): 20ms

[Test 6] Measuring Cache Miss Latency After Invalidation...
success Cache invalidated frontend-1766341300935
Test 1 - Invalidation: 4ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 2 - Invalidation: 3ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 3 - Invalidation: 4ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 4 - Invalidation: 4ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 5 - Invalidation: 3ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 6 - Invalidation: 4ms, Next Request: 6ms
success Cache invalidated frontend-1766341300935
Test 7 - Invalidation: 3ms, Next Request: 6ms
```

Tomorrow's high
Near record

Search 9:28 PM 12/21/2025

```
Windows PowerShell x + -
```

```
success Cache invalidated frontend-1766341300935
Test 8 - Invalidation: 3ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 9 - Invalidation: 3ms, Next Request: 5ms
success Cache invalidated frontend-1766341300935
Test 10 - Invalidation: 4ms, Next Request: 7ms

== Performance Test Results ==

Search Requests (without cache):
Average: 3.4ms
Median: 2ms
Min: 2ms
Max: 17ms

Search Requests (with cache):
Average: 2.7ms
Median: 2ms
Min: 2ms
Max: 1ms

Search Cache Improvement: 20.59%

Info Requests (without cache):
Average: 7.15ms
Median: 7ms
Min: 5ms
Max: 11ms

Info Requests (with cache):
Average: 2.2ms
Median: 2ms
Min: 2ms
Max: 3ms

Info Cache Improvement: 69.23%

Purchase Operations:
Average: 26.9ms
Median: 23.5ms
Min: 20ms
```

58°F Mostly cloudy

Search 9:29 PM 12/21/2025

```

Windows PowerShell x + v
Median: 7ms
Min: 5ms
Max: 11ms

Info Requests (with cache):
Average: 2.2ms
Median: 2ms
Min: 2ms
Max: 3ms

Info Cache Improvement: 69.23%

Purchase Operations:
Average: 26.9ms
Median: 23.5ms
Min: 20ms
Max: 56ms

Cache Invalidation:
Average: 3.5ms
Median: 3.5ms
Min: 3ms
Max: 4ms

Results exported to: docs\performance-results.csv
== Test Complete ==

PS C:\MyDocuments\univeristy-courses\Dos\bazar-com> |

```

8. Final status and shutdown

Commands:

`docker-compose ps`

`docker-compose down`

`docker-compose ps`

```

PS C:\MyDocuments\univeristy-courses\Dos\bazar-com> docker-compose ps
NAME           IMAGE          COMMAND         SERVICE      CREATED        STATUS
PORTS
catalog-service   bazar-com-catalog    "docker-entrypoint.s..." catalog
0.0.0.0:3001->3001/tcp, [::]:3001->3001/tcp
frontend-service  bazar-com-frontend   "docker-entrypoint.s..." frontend
0.0.0.0:3002->3002/tcp, [::]:3002->3002/tcp
order-service     bazar-com-order     "docker-entrypoint.s..." order
0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp
PS C:\MyDocuments\univeristy-courses\Dos\bazar-com> docker-compose down
[+] Running 4/4
✓ Container frontend-service Removed
✓ Container order-service Removed
✓ Container catalog-service Removed
✓ Network bazar-network Removed
PS C:\MyDocuments\univeristy-courses\Dos\bazar-com> docker-compose ps
NAME           IMAGE          COMMAND         SERVICE      CREATED        STATUS
PORTS
PS C:\MyDocuments\univeristy-courses\Dos\bazar-com>

```

THANK YOU 😊