Tests and outputs to check full application functionality and Expected Behavior:
(individual microservice and unit tests present in test/ directory)

Test 1 : order creation via frontend with cache involvement

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s -X POST http://localhost:7070/orders \
  -H "Content-Type: application/json" \
  -d '{"name": "Stock1", "type": "buy", "quantity": 5}'

{"data":{"name":"Stock1","type":"buy","quantity":5,"number":1}}%
```

Test 2: order retrieval via frontend

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s http://localhost:7070/orders/1

{"data":{"name":"Stock1","type":"buy","quantity":5,"number":1}}%
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika %
```

Test 3: check cache flow and invalidation

i) cache miss (query from catalog)

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s http://localhost:7070/stocks/Stock1

{"volume":100,"name":"Stock1"}%
```

ii) cache hit (will use cache)

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s http://localhost:7070/stocks/Stock1

{"volume":100,"name":"Stock1"}%
```

iii) invalidate cache

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s -X POST http://localhost:7070/stocks/invalidate/Stock1

Cache invalidated for stock: Stock1%
```

iv) fetch again (cache miss)

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % curl -s http://localhost:7070/stocks/Stock1

{"volume":100,"name":"Stock1"}%
```

Logs from frontend (above 4 steps done in sequence):

```
2025-05-06 01:33:18 - Stock lookup request received: Stock1
2025-05-06 01:33:18 - CACHE MISS: Stock1
2025-05-06 01:33:18 - Fetching Stock1 from catalog at http://localhost:8081
2025-05-06 01:33:18 - Caching result for Stock1
2025-05-06 01:33:43 - Stock lookup request received: Stock1
2025-05-06 01:33:43 - CACHE HIT: Stock1
2025-05-06 01:34:19 - Cache invalidated for Stock1
2025-05-06 01:34:27 - Stock lookup request received: Stock1
2025-05-06 01:34:27 - CACHE MISS: Stock1
2025-05-06 01:34:27 - Fetching Stock1 from catalog at http://localhost:8081
2025-05-06 01:34:27 - Caching result for Stock1
```

Test 4: check replication on all order replicas

```
(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika % for port in 9091 9092 9093; do
  echo "Replica $port:"
  curl -s http://localhost:$port/orders/1
  echo -e "\n"
done

Replica 9091:
{"data":{"name":"Stock1","type":"buy","quantity":5,"number":1}}

Replica 9092:
{"data":{"name":"Stock1","type":"buy","quantity":5,"number":1}}

Replica 9093:
{"data":{"name":"Stock1","type":"buy","quantity":5,"number":1}}

(base) lavanika@Lavanikas-MacBook-Pro spring25-lab3-lava-nika %
```

Test 5: Leader crash simulation

i) kill current leader

ii) still get expected output:

```
(base) lavanika@Lavanikas-MacBook-Pro test % curl -s -X POST http://localhost:7070/orders \
  -H "Content-Type: application/json" \
  -d '{"name": "Stock2", "type": "sell", "quantity": 4}'

{"data":{"name":"Stock2","type":"sell","quantity":4,"number":1}}%
(base) lavanika@Lavanikas-MacBook-Pro test %
```

iii) restart killed replica, verify all replicas have order number 1.

```
(base) lavanika@Lavanikas-MacBook-Pro test %
for port in 9091 9092 9093; do
  echo "Replica on port $port:"
  curl -s http://localhost:$port/orders/1
  echo -e "\n"
done
Replica on port 9091:
{"data":{"name":"Stock2","type":"sell","quantity":4,"number":1}}

Replica on port 9092:
{"data":{"name":"Stock2","type":"sell","quantity":4,"number":1}}

Replica on port 9093:
{"data":{"name":"Stock2","type":"sell","quantity":4,"number":1}}

(base) lavanika@Lavanikas-MacBook-Pro test %
```