

1 Output of Test Cases:

```
C:\Users\thewi\Desktop\CS677\lab3\src\test-cases>py testcases.py 10.0.0.246 56893
```

```
#####  
##Test Case 1 - Lookup Functionality: ##  
#####
```

```
Input Request: GET /stocks/nvidia
```

```
JSON Reply: {'data': {'name': 'nvidia', 'price': 240.63, 'quantity': 1000}}
```

```
#####  
##Test Case 2 - Lookup Error Handling (Stock not found): ##  
#####
```

```
Input Request: GET /stocks/imaginarycompany
```

```
JSON Reply: {'error': {'code': 404, 'message': 'stock not found'}}
```

```
#####  
##Test Case 3 - Sell Functionality: ##  
#####
```

```
Input Request: POST /orders
```

```
Input JSON: {"name": "meta", "type": "sell", "quantity": 10}
```

```
Stock Information before Request: {'data': {'name': 'meta', 'price': 194.02, 'quantity': 1000}}
```

```
JSON Reply: {'data': {'transaction_number': 0}}
```

```
Stock Information after Request: {'data': {'name': 'meta', 'price': 194.02, 'quantity': 1010}}
```

```
#####  
##Test Case 4 - Buy Functionality: ##  
#####
```

```
Input Request: POST /orders
```

```
Input JSON: {"name": "gamestop", "type": "buy", "quantity": 10}
```

```
Stock Information before Request: {'data': {'name': 'gamestop', 'price': 20.0, 'quantity': 1000}}
```

```
JSON Reply: {'data': {'transaction_number': 1}}
```

```
Stock Information after Request: {'data': {'name': 'gamestop', 'price': 20.0, 'quantity': 990}}
```

```
#####  
##Test Case 5 - Buy Error Handling (Amount to buy > num of stocks avail.): ##  
#####
```

```
Input Request: POST /orders
```

```
Input JSON: {"name": "nvidia", "type": "buy", "quantity": 10000000}
```

```
JSON Reply: {'error': {'code': 404, 'message': 'not enough stocks available to buy'}}
```

```
#####  
##Test Case 6 - Buy/Sell Error Handling (Invalid quantity of stocks): ##  
#####
```

Note: PROTO already defines quantity needs to be an int. As such, the only invalid value will be negat

```
Input Request: POST /orders
```

```
Input JSON: {"name": "amazon", "type": "buy", "quantity": -1000}
```

```
JSON Reply: {'error': {'code': 404, 'message': 'invalid number of stocks'}}
```

```
Input Request: POST /orders
```

```
Input JSON: {"name": "amazon", "type": "sell", "quantity": -1000}
```

```
JSON Reply: {'error': {'code': 404, 'message': 'invalid number of stocks'}}
```

```
#####  
##Test Case 7 - Buy/Sell Error Handling (Invalid request type): ##
```

```
#####
Input Request: POST /orders
Input JSON: {"name": "meta", "type": "trade", "quantity": 100}
JSON Reply: {'error': {'code': 400, 'message': 'invalid request type'}}

#####
##Test Case 8 - Buy/Sell Error Handling (Stock not found): ##
#####
Input Request: POST /orders
Input JSON: {"name": "imaginarycompany", "type": "buy", "quantity": 100}
JSON Reply: {'error': {'code': 404, 'message': 'stock not found'}}

#####
##Test Case 9 - Caching Functionality: ##
#####
Request 1a (s.t. it is stored in local cache): GET /stocks/amazon
Request 1b (s.t. we can retrieve from cache): GET /stocks/amazon
Latency with caching: 0.00027210000553168356
Request 2a (trade stock so cache is invalidated): POST /orders
Request 2b (lookup stock without cache): GET /stocks/amazon
Latency without caching: 0.0015700000076321885
Difference in time: 0.001297900002100505
Percent speed up: 82.6687895408323

#####
##Test Case 10 - Order Query Functionality: ##
#####
Input Request: POST /orders
Input JSON: {"name": "tesla", "type": "sell", "quantity": 10}
Client Locally Stored Information: {3: {'number': 3, 'name': 'tesla', 'type': 'sell', 'quantity': 10}}
Input Request: GET /orders/3
Output of Order Query: {'data': {'number': 3, 'name': 'tesla', 'type': 'sell', 'quantity': 10}}

#####
##Test Case 11 - Order Query Error Handling: ##
#####
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "sell", "quantity": 10}
Client Locally Stored Information: {4: {'number': 4, 'name': 'amc', 'type': 'sell', 'quantity': 10}}
Input Request: GET /orders/5
Output of Order Query: {'error': {'code': 404, 'message': 'transaction not found'}}

#####
##Test Case 12 - Propagation Functionality: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "gamestop", "type": "sell", "quantity": 31}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "gamestop", "type": "sell", "quantity": 14}
JSON Reply: {'data': {'transaction_number': 1}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "gamestop", "type": "sell", "quantity": 31}, "1": {"name": "gamestop", "type": "sell"}
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "gamestop", "type": "sell", "quantity": 31}, "1": {"name": "gamestop", "type": "sell"}
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{"0": {"name": "gamestop", "type": "sell", "quantity": 31}, "1": {"name": "gamestop", "type": "sell"}
ALL THREE FILES ARE THE SAME: True
```

```

#####
##Test Case 13 - Largest Replica ID Dead on Arrival: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "tesla", "type": "sell", "quantity": 15}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "tesla", "type": "sell", "quantity": 19}
JSON Reply: {'data': {'transaction_number': 1}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "tesla", "type": "buy", "quantity": 67}
JSON Reply: {'data': {'transaction_number': 2}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "tesla", "type": "sell", "quantity": 15}, "1": {"name": "tesla", "type": "sell", "quantity": 19}]
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "tesla", "type": "sell", "quantity": 15}, "1": {"name": "tesla", "type": "sell", "quantity": 19}]
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{}']
ALL THREE FILES ARE THE SAME: False
REPLICAS 2 AND 1 ARE THE SAME: True

#####
##Test Case 14 - Leader Crashes During Execution: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "meta", "type": "sell", "quantity": 69}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "meta", "type": "sell", "quantity": 71}
JSON Reply: {'data': {'transaction_number': 1}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "meta", "type": "buy", "quantity": 30}
JSON Reply: {'data': {'transaction_number': 2}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "meta", "type": "sell", "quantity": 69}, "1": {"name": "meta", "type": "sell", "quantity": 71}, "2": {"name": "meta", "type": "buy", "quantity": 30}}]
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "meta", "type": "sell", "quantity": 69}, "1": {"name": "meta", "type": "sell", "quantity": 71}, "2": {"name": "meta", "type": "buy", "quantity": 30}}]
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{"0": {"name": "meta", "type": "sell", "quantity": 69}}']
ALL THREE FILES ARE THE SAME: False
REPLICAS 2 AND 1 ARE THE SAME: True

#####
##Test Case 15 - Follower Crashes During Execution: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "sell", "quantity": 47}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "buy", "quantity": 83}
JSON Reply: {'data': {'transaction_number': 1}}
----- Input -----
Input Request: POST /orders

```

```

Input JSON: {"name": "amc", "type": "buy", "quantity": 45}
JSON Reply: {'data': {'transaction_number': 2}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 47}, "1": {"name": "amc", "type": "buy", "quantity": 45}]
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 47}, "1": {"name": "amc", "type": "buy", "quantity": 45}]
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 47}, "1": {"name": "amc", "type": "buy", "quantity": 45}]
ALL THREE FILES ARE THE SAME: False
REPLICAS 3 AND 1 ARE THE SAME: True

#####
##Test Case 16 - Leader Crashes and Rejoins During Execution: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "sell", "quantity": 7}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "buy", "quantity": 40}
--- Content of Replica 3 at Time of Crash: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 7}}']
JSON Reply: {'data': {'transaction_number': 1}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "sell", "quantity": 75}
JSON Reply: {'data': {'transaction_number': 2}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "amc", "type": "sell", "quantity": 70}
JSON Reply: {'data': {'transaction_number': 3}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 7}, "1": {"name": "amc", "type": "buy", "quantity": 40}]
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 7}, "1": {"name": "amc", "type": "buy", "quantity": 40}]
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{"0": {"name": "amc", "type": "sell", "quantity": 7}, "1": {"name": "amc", "type": "buy", "quantity": 40}]
ALL THREE FILES ARE THE SAME: True
REPLICAS 3 AND 1 ARE THE SAME: True

#####
##Test Case 17 - Follower Crashes and Rejoins During Execution: ##
#####
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "nvidia", "type": "buy", "quantity": 44}
JSON Reply: {'data': {'transaction_number': 0}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "nvidia", "type": "sell", "quantity": 6}
--- Content of Replica 2 at Time of Crash: ---
['{"0": {"name": "nvidia", "type": "buy", "quantity": 44}, "1": {"name": "nvidia", "type": "sell", "quantity": 6}}']
JSON Reply: {'data': {'transaction_number': 1}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "nvidia", "type": "buy", "quantity": 94}
JSON Reply: {'data': {'transaction_number': 2}}
----- Input -----
Input Request: POST /orders
Input JSON: {"name": "nvidia", "type": "buy", "quantity": 51}

```

```

JSON Reply: {'data': {'transaction_number': 3}}
--- Content of Replica 1 at C:\Users\thewi\Desktop\CS677\lab3\src\order1.txt: ---
['{"0": {"name": "nvidia", "type": "buy", "quantity": 44}, "1": {"name": "nvidia", "type": "sell", "qu
--- Content of Replica 2 at C:\Users\thewi\Desktop\CS677\lab3\src\order2.txt: ---
['{"0": {"name": "nvidia", "type": "buy", "quantity": 44}, "1": {"name": "nvidia", "type": "sell", "qu
--- Content of Replica 3 at C:\Users\thewi\Desktop\CS677\lab3\src\order3.txt: ---
['{"0": {"name": "nvidia", "type": "buy", "quantity": 44}, "1": {"name": "nvidia", "type": "sell", "qu
ALL THREE FILES ARE THE SAME: True
REPLICAS 3 AND 1 ARE THE SAME: True

```

```
C:\Users\thewi\Desktop\CS677\lab3\src\test-cases>
```

2 Proof of Functionality:

2.1 Basic Functionality:

```

ubuntu@ip-172-31-21-21: ~/src/catalog-service
ubuntu@ip-172-31-21-21:~/src/catalog-service$ python3.11 catalogServer.py
Catalog server started on: 172.31.21.21:56892

Select ubuntu@ip-172-31-21-21: ~/src/order-service
ubuntu@ip-172-31-21-21:~/src/order-service$ python3.11 orderServer.py 56891 3 172.31.21.21 56893 1
Unique ID of this order service: 3
Order server started on: 172.31.21.21:56891
Followers Assigned: {'followerOne': {'ip': '172.31.21.21', 'port': '56890'}, 'followerTwo': {'ip': '172.31.21.21', 'port': '56889'}}

ubuntu@ip-172-31-21-21: ~/src/order-service
ubuntu@ip-172-31-21-21:~/src/order-service$ python3.11 orderServer.py 56890 2 172.31.21.21 56893 1
Unique ID of this order service: 2
Order server started on: 172.31.21.21:56890

ubuntu@ip-172-31-21-21: ~/src/order-service
ubuntu@ip-172-31-21-21:~/src/order-service$ python3.11 orderServer.py 56889 1 172.31.21.21 56893 1
Unique ID of this order service: 1
Order server started on: 172.31.21.21:56889

ubuntu@ip-172-31-21-21: ~/src/front-end
ubuntu@ip-172-31-21-21:~/src/front-end$ python3.11 frontEnd.py 3 172.31.21.21:56891 2 172.31.21.21:56890 1 172.31.21.21:56889 1
http server is starting...
Front-end server started on: 172.31.21.21:56893
http server is running...

```

Figure 1: Output of the initialized microservices.

```
ubuntu@ip-172-31-16-170: ~/src/catalog-service
ubuntu@ip-172-31-16-170:~/src/catalog-service$ python3.11 catalogServer.py
Catalog server started on: 172.31.16.170:56892

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56891 3 172.31.16.170 56893 1
Unique ID of this order service: 3
Order server started on: 172.31.16.170:56891
Followers Assigned: {'followerOne': {'ip': '172.31.16.170', 'port': '56890'}, 'followerTwo': {'ip': '172.31.16.170', 'port': '56889'}}
Written on Replica 3: {41: {'name': 'nike', 'type': 'buy', 'quantity': 41}}
Written on Replica 3: {42: {'name': 'nvidia', 'type': 'sell', 'quantity': 33}}
Written on Replica 3: {43: {'name': 'nvidia', 'type': 'sell', 'quantity': 99}}
Written on Replica 3: {44: {'name': 'nvidia', 'type': 'buy', 'quantity': 97}}

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56890 2 172.31.16.170 56893 1
Unique ID of this order service: 2
Order server started on: 172.31.16.170:56890
Propagated on Replica 2: {41: {'name': 'nike', 'type': 'buy', 'quantity': 41}}
Propagated on Replica 2: {42: {'name': 'nvidia', 'type': 'sell', 'quantity': 33}}
Propagated on Replica 2: {43: {'name': 'nvidia', 'type': 'sell', 'quantity': 99}}
Propagated on Replica 2: {44: {'name': 'nvidia', 'type': 'buy', 'quantity': 97}}

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56889 1 172.31.16.170 56893 1
Unique ID of this order service: 1
Order server started on: 172.31.16.170:56889
Propagated on Replica 1: {41: {'name': 'nike', 'type': 'buy', 'quantity': 41}}
Propagated on Replica 1: {42: {'name': 'nvidia', 'type': 'sell', 'quantity': 33}}
Propagated on Replica 1: {43: {'name': 'nvidia', 'type': 'sell', 'quantity': 99}}
Propagated on Replica 1: {44: {'name': 'nvidia', 'type': 'buy', 'quantity': 97}}

ubuntu@ip-172-31-16-170: ~/src/front-end
ubuntu@ip-172-31-16-170:~/src/front-end$ python3.11 frontEnd.py 3 172.31.16.170:56891 2 172.31.16.170:56890 1 172.31.16.170:56889 1
http server is starting...
Front-end server started on: 172.31.16.170:56893
http server is running...
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/tesla HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/intel HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/intel HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/nike HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/amazon HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/gamestop HTTP/1.1" 200 -
172.31.16.170 - - [27/Apr/2023 21:33:06] "GET /cache/nike HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "POST /orders HTTP/1.1" 200 -
172.31.16.170 - - [27/Apr/2023 21:33:06] "GET /cache/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "POST /orders HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "GET /stocks/nvidia HTTP/1.1" 200 -
172.31.16.170 - - [27/Apr/2023 21:33:06] "GET /cache/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "POST /orders HTTP/1.1" 200 -
172.31.16.170 - - [27/Apr/2023 21:33:06] "GET /cache/nvidia HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:06] "POST /orders HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:07] "GET /orders/41 HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:07] "GET /orders/42 HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:07] "GET /orders/43 HTTP/1.1" 200 -
73.186.87.78 - - [27/Apr/2023 21:33:07] "GET /orders/44 HTTP/1.1" 200 -
```

Figure 2: Example output of the stock service. "GET /cache/stock name" requests shown proving server-push invalidation functionality. Replication shown between the 3 order replicas.

```
client-1 - call py HTTPClient.py 54.146.246.125 56893 0.5.0
----- Lookup -----
Input: meta
Output: {'data': {'name': 'meta', 'price': 194.02, 'quantity': 1000}}
----- Trade -----
Input: tesla
Output: {'data': {'name': 'tesla', 'price': 183.26, 'quantity': 1000}}
----- Order Query Verification -----
Successful Trades Made: 1
----- Verifying Order 2 -----
Local: {'number': 2, 'name': 'tesla', 'type': 'sell', 'quantity': 45}
Server: {'number': 2, 'name': 'tesla', 'type': 'sell', 'quantity': 45}
MATCH
ALL ORDERS VERIFIED
----- Average Latency of Requests -----
Average Latency of Catalog Lookup Request: 0.1881238999994914
Average Latency of Order Trade Request: 0.14610838000039842
Average Latency of Order Query Lookup: 0.13563879999946946
C:\Users\theud\Desktop\CS677\lab3\src\client>

client-2 - call py HTTPClient.py 54.146.246.125 56893 0.5.0
----- Lookup -----
Input: nike
Output: {'data': {'name': 'nike', 'price': 126.13, 'quantity': 1000}}
----- Trade -----
Input: tesla
Output: {'data': {'name': 'tesla', 'price': 183.26, 'quantity': 1000}}
----- Order Query Verification -----
Successful Trades Made: 1
----- Verifying Order 0 -----
Local: {'number': 0, 'name': 'nike', 'type': 'sell', 'quantity': 6}
Server: {'number': 0, 'name': 'nike', 'type': 'sell', 'quantity': 6}
MATCH
ALL ORDERS VERIFIED
----- Average Latency of Requests -----
Average Latency of Catalog Lookup Request: 0.175807149999985503
Average Latency of Order Trade Request: 0.14165370000046096
Average Latency of Order Query Lookup: 0.1352614999996149
C:\Users\theud\Desktop\CS677\lab3\src\client>

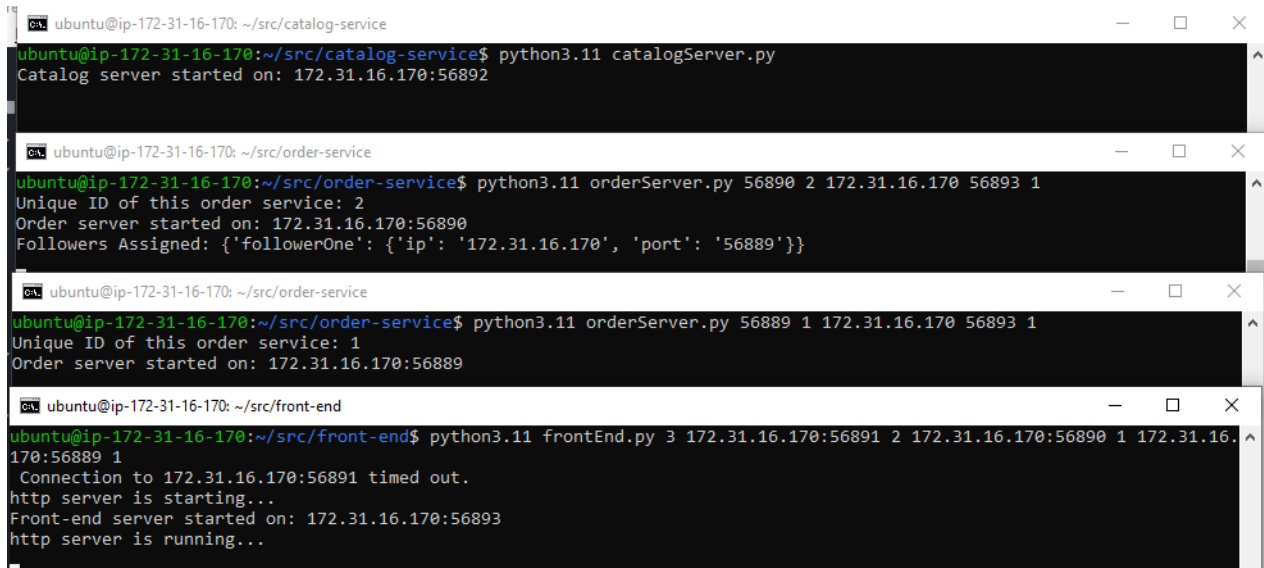
client-3 - call py HTTPClient.py 54.146.246.125 56893 0.5.0
----- Lookup -----
Input: apple
Output: {'data': {'name': 'apple', 'price': 152.59, 'quantity': 1000}}
----- Trade -----
Input: apple
Output: {'data': {'name': 'apple', 'price': 152.59, 'quantity': 1000}}
----- Order Query Verification -----
Successful Trades Made: 1
----- Verifying Order 1 -----
Local: {'number': 1, 'name': 'apple', 'type': 'sell', 'quantity': 27}
Server: {'number': 1, 'name': 'apple', 'type': 'sell', 'quantity': 27}
MATCH
ALL ORDERS VERIFIED
----- Average Latency of Requests -----
Average Latency of Catalog Lookup Request: 0.17505544999994527
Average Latency of Order Trade Request: 0.13676399999985733
Average Latency of Order Query Lookup: 0.1361722999999836
C:\Users\theud\Desktop\CS677\lab3\src\client>

client-4 - call py HTTPClient.py 54.146.246.125 56893 0.5.0
----- Lookup -----
Input: nvidia
Output: {'data': {'name': 'nvidia', 'price': 240.63, 'quantity': 1000}}
----- Trade -----
Input: intel
Output: {'data': {'name': 'intel', 'price': 28.01, 'quantity': 1000}}
----- Order Query Verification -----
Successful Trades Made: 0
ALL ORDERS VERIFIED
----- Average Latency of Requests -----
Average Latency of Catalog Lookup Request: 0.18006004999961078
Average Latency of Order Trade Request: N/A
Average Latency of Order Query Lookup: N/A
C:\Users\theud\Desktop\CS677\lab3\src\client>

client-5 - call py HTTPClient.py 54.146.246.125 56893 0.5.0
----- Lookup -----
Input: intel
Output: {'data': {'name': 'intel', 'price': 28.01, 'quantity': 1000}}
----- Trade -----
Input: amd
Output: {'data': {'name': 'amd', 'price': 4.78, 'quantity': 1000}}
----- Order Query Verification -----
Successful Trades Made: 1
----- Verifying Order 3 -----
Local: {'number': 3, 'name': 'amd', 'type': 'sell', 'quantity': 26}
Server: {'number': 3, 'name': 'amd', 'type': 'sell', 'quantity': 26}
MATCH
ALL ORDERS VERIFIED
----- Average Latency of Requests -----
Average Latency of Catalog Lookup Request: 0.175694150000254
Average Latency of Order Trade Request: 0.16128209999988803
Average Latency of Order Query Lookup: 0.1157501999995234
C:\Users\theud\Desktop\CS677\lab3\src\client>
```

Figure 3: Example output of the 5 clients connecting to the stock service. Clients display order query verification functionality.

2.2 Fault Tolerance:



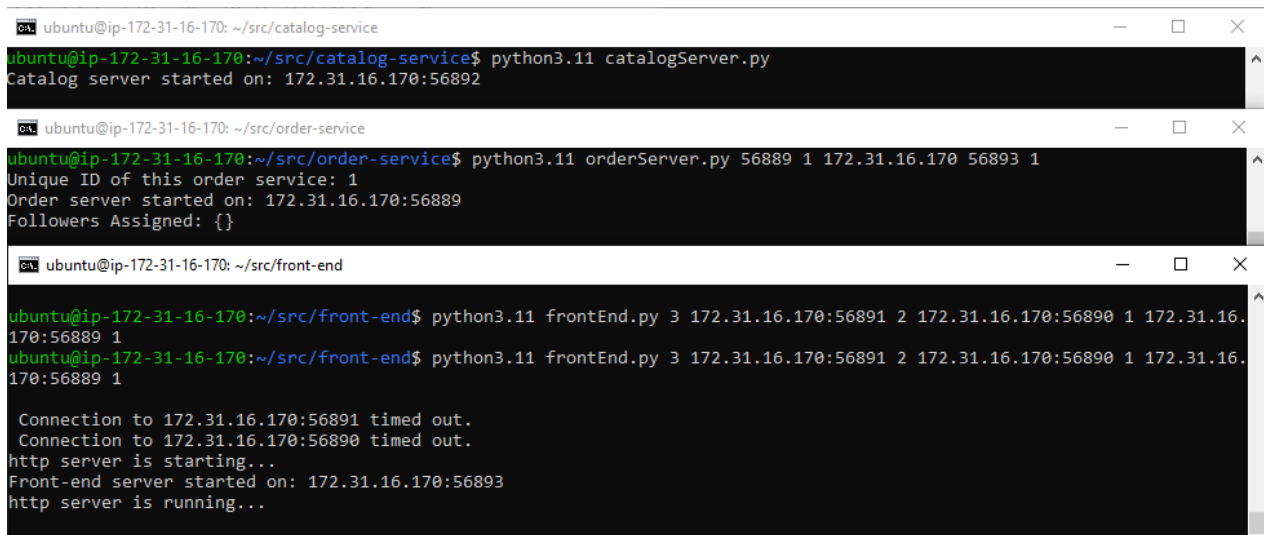
```
ubuntu@ip-172-31-16-170: ~/src/catalog-service
ubuntu@ip-172-31-16-170:~/src/catalog-service$ python3.11 catalogServer.py
Catalog server started on: 172.31.16.170:56892

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56890 2 172.31.16.170 56893 1
Unique ID of this order service: 2
Order server started on: 172.31.16.170:56890
Followers Assigned: {'followerOne': {'ip': '172.31.16.170', 'port': '56889'}}

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56889 1 172.31.16.170 56893 1
Unique ID of this order service: 1
Order server started on: 172.31.16.170:56889

ubuntu@ip-172-31-16-170: ~/src/front-end
ubuntu@ip-172-31-16-170:~/src/front-end$ python3.11 frontEnd.py 3 172.31.16.170:56891 2 172.31.16.170:56890 1 172.31.16.170:56889 1
Connection to 172.31.16.170:56891 timed out.
http server is starting...
Front-end server started on: 172.31.16.170:56893
http server is running...
```

Figure 4: Example output of when the largest replica ID is dead on arrival.



```
ubuntu@ip-172-31-16-170: ~/src/catalog-service
ubuntu@ip-172-31-16-170:~/src/catalog-service$ python3.11 catalogServer.py
Catalog server started on: 172.31.16.170:56892

ubuntu@ip-172-31-16-170: ~/src/order-service
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56889 1 172.31.16.170 56893 1
Unique ID of this order service: 1
Order server started on: 172.31.16.170:56889
Followers Assigned: {}

ubuntu@ip-172-31-16-170: ~/src/front-end
ubuntu@ip-172-31-16-170:~/src/front-end$ python3.11 frontEnd.py 3 172.31.16.170:56891 2 172.31.16.170:56890 1 172.31.16.170:56889 1
Connection to 172.31.16.170:56891 timed out.
Connection to 172.31.16.170:56890 timed out.
http server is starting...
Front-end server started on: 172.31.16.170:56893
http server is running...
```

Figure 5: Example output of when the two largest replica IDs are dead on arrival.


```
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56891 172.31.16.170 56891 1
Unique ID of this order service: 2
Order server started on: 172.31.16.170:56891
Followers Assigned: {'followerOne': ('ip': '172.31.16.170', 'port': '56890'), 'followerTwo': ('ip': '172.31.16.170', 'port': '56889')}
Written on Replica 3: (0: {'name': 'amazon', 'type': 'buy', 'quantity': 68})
Written on Replica 3: (1: {'name': 'nvidia', 'type': 'buy', 'quantity': 45})
Written on Replica 3: (2: {'name': 'ford', 'type': 'sell', 'quantity': 29})
Written on Replica 3: (3: {'name': 'apple', 'type': 'sell', 'quantity': 54})
Written on Replica 3: (4: {'name': 'amc', 'type': 'buy', 'quantity': 77})
Written on Replica 3: (5: {'name': 'amc', 'type': 'sell', 'quantity': 48})
Written on Replica 3: (6: {'name': 'nvidia', 'type': 'sell', 'quantity': 8})
Written on Replica 3: (7: {'name': 'gamestop', 'type': 'buy', 'quantity': 23})
Written on Replica 3: (8: {'name': 'apple', 'type': 'buy', 'quantity': 13})
Written on Replica 3: (9: {'name': 'amazon', 'type': 'buy', 'quantity': 98})
Written on Replica 3: (10: {'name': 'apple', 'type': 'buy', 'quantity': 24})
C\Traceback (most recent call last):
  File "/home/ubuntu/src/order-service/orderServer.py", line 236, in <module>
    serve()
  File "/home/ubuntu/src/order-service/orderServer.py", line 224, in serve
    server.wait_for_termination()
  File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_server.py", line 1118, in wait_for_termination
    return common.wait(self._state.termination_event.wait,
File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_common.py", line 150, in wait
    wait_once(wait_fn, MAXIMUM_WAIT_TIMEOUT, spin_cb)
File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_common.py", line 112, in wait_once
    wait_fn(timeout=timeout)
File "/usr/lib/python3.11/threading.py", line 622, in wait
    signaled = self._cond.wait(timeout)
File "/usr/lib/python3.11/threading.py", line 324, in wait
    gotit = waiter.acquire(True, timeout)
KeyboardInterrupt

ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56891 172.31.16.170 56891 1
Unique ID of this order service: 3
Order server started on: 172.31.16.170:56891
Synchronized on Replica 3: (11: {'name': 'intel', 'type': 'buy', 'quantity': 51})
Synchronized on Replica 3: (12: {'name': 'gamestop', 'type': 'sell', 'quantity': 36})
Synchronized on Replica 3: (13: {'name': 'amc', 'type': 'buy', 'quantity': 61})
Synchronized on Replica 3: (14: {'name': 'meta', 'type': 'buy', 'quantity': 90})
Synchronized on Replica 3: (15: {'name': 'gamestop', 'type': 'sell', 'quantity': 100})
Synchronized on Replica 3: (16: {'name': 'amc', 'type': 'buy', 'quantity': 87})
Synchronized on Replica 3: (17: {'name': 'amc', 'type': 'buy', 'quantity': 95})
Synchronized on Replica 3: (18: {'name': 'meta', 'type': 'buy', 'quantity': 82})
Synchronized on Replica 3: (19: {'name': 'amazon', 'type': 'buy', 'quantity': 37})
Synchronized on Replica 3: (20: {'name': 'intel', 'type': 'sell', 'quantity': 90})
Propagated on Replica 3: (21: {'name': 'gamestop', 'type': 'buy', 'quantity': 44})
Propagated on Replica 3: (22: {'name': 'intel', 'type': 'buy', 'quantity': 43})
Propagated on Replica 3: (23: {'name': 'intel', 'type': 'sell', 'quantity': 70})
Propagated on Replica 3: (24: {'name': 'intel', 'type': 'sell', 'quantity': 19})
Propagated on Replica 3: (25: {'name': 'gamestop', 'type': 'buy', 'quantity': 68})
Propagated on Replica 3: (26: {'name': 'meta', 'type': 'buy', 'quantity': 68})
Propagated on Replica 3: (27: {'name': 'ford', 'type': 'buy', 'quantity': 13})
Propagated on Replica 3: (28: {'name': 'tesla', 'type': 'sell', 'quantity': 75})
Propagated on Replica 3: (29: {'name': 'tesla', 'type': 'sell', 'quantity': 73})
Propagated on Replica 3: (30: {'name': 'tesla', 'type': 'sell', 'quantity': 73})
Propagated on Replica 3: (31: {'name': 'ford', 'type': 'sell', 'quantity': 78})
Propagated on Replica 3: (32: {'name': 'amc', 'type': 'sell', 'quantity': 69})
Propagated on Replica 3: (33: {'name': 'nvidia', 'type': 'sell', 'quantity': 83})
Propagated on Replica 3: (34: {'name': 'meta', 'type': 'sell', 'quantity': 68})
Propagated on Replica 3: (35: {'name': 'intel', 'type': 'sell', 'quantity': 90})
Propagated on Replica 3: (36: {'name': 'apple', 'type': 'buy', 'quantity': 17})
Propagated on Replica 3: (37: {'name': 'intel', 'type': 'sell', 'quantity': 26})
```

Figure 8: Example output when the leader replica crashes during execution and rejoins. Replica ID 3 shown on left, replica ID 2 shown on right.

```
ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56891 172.31.16.170 56891 1
Unique ID of this order service: 3
Order server started on: 172.31.16.170:56891
Followers Assigned: {'followerOne': ('ip': '172.31.16.170', 'port': '56890'), 'followerTwo': ('ip': '172.31.16.170', 'port': '56889')}
Written on Replica 3: (0: {'name': 'tesla', 'type': 'sell', 'quantity': 49})
Written on Replica 3: (1: {'name': 'intel', 'type': 'sell', 'quantity': 49})
Written on Replica 3: (2: {'name': 'gamestop', 'type': 'buy', 'quantity': 41})
Written on Replica 3: (3: {'name': 'intel', 'type': 'sell', 'quantity': 24})
Written on Replica 3: (4: {'name': 'apple', 'type': 'sell', 'quantity': 19})
Written on Replica 3: (5: {'name': 'amc', 'type': 'sell', 'quantity': 81})
Written on Replica 3: (6: {'name': 'gamestop', 'type': 'buy', 'quantity': 33})
Written on Replica 3: (7: {'name': 'nike', 'type': 'sell', 'quantity': 82})
Written on Replica 3: (8: {'name': 'meta', 'type': 'sell', 'quantity': 9})
Written on Replica 3: (9: {'name': 'apple', 'type': 'sell', 'quantity': 69})
Written on Replica 3: (10: {'name': 'nvidia', 'type': 'buy', 'quantity': 71})
Written on Replica 3: (11: {'name': 'amc', 'type': 'buy', 'quantity': 80})
Written on Replica 3: (12: {'name': 'amc', 'type': 'buy', 'quantity': 37})
Written on Replica 3: (13: {'name': 'amc', 'type': 'sell', 'quantity': 90})
Written on Replica 3: (14: {'name': 'amc', 'type': 'buy', 'quantity': 58})
Written on Replica 3: (15: {'name': 'ford', 'type': 'sell', 'quantity': 62})
Follower address does not exist 172.31.16.170:56889
Written on Replica 3: (16: {'name': 'meta', 'type': 'sell', 'quantity': 94})
Written on Replica 3: (17: {'name': 'nike', 'type': 'sell', 'quantity': 59})
Written on Replica 3: (18: {'name': 'nvidia', 'type': 'sell', 'quantity': 63})
Written on Replica 3: (19: {'name': 'ford', 'type': 'buy', 'quantity': 92})
Written on Replica 3: (20: {'name': 'gamestop', 'type': 'buy', 'quantity': 9})
Written on Replica 3: (21: {'name': 'tesla', 'type': 'buy', 'quantity': 68})
Written on Replica 3: (22: {'name': 'apple', 'type': 'sell', 'quantity': 8})
Written on Replica 3: (23: {'name': 'gamestop', 'type': 'buy', 'quantity': 16})
Written on Replica 3: (24: {'name': 'apple', 'type': 'buy', 'quantity': 40})
Written on Replica 3: (25: {'name': 'ford', 'type': 'buy', 'quantity': 42})
Written on Replica 3: (26: {'name': 'ford', 'type': 'sell', 'quantity': 69})
Written on Replica 3: (27: {'name': 'nike', 'type': 'sell', 'quantity': 63})
Written on Replica 3: (28: {'name': 'gamestop', 'type': 'buy', 'quantity': 33})
Written on Replica 3: (29: {'name': 'apple', 'type': 'sell', 'quantity': 88})
Written on Replica 3: (30: {'name': 'nike', 'type': 'buy', 'quantity': 57})
Replica 3 Synchronizing with 172.31.16.170:56890
Followers Before Synchronization: {'followerTwo': ('ip': '172.31.16.170', 'port': '56889'), 'followerOne': ('ip': '172.31.16.170', 'port': '56890')}
Written on Replica 3: (31: {'name': 'apple', 'type': 'sell', 'quantity': 85})
Written on Replica 3: (32: {'name': 'nike', 'type': 'sell', 'quantity': 65})
Written on Replica 3: (33: {'name': 'amc', 'type': 'buy', 'quantity': 43})
Written on Replica 3: (34: {'name': 'intel', 'type': 'sell', 'quantity': 48})
Written on Replica 3: (35: {'name': 'gamestop', 'type': 'sell', 'quantity': 93})
Written on Replica 3: (36: {'name': 'nvidia', 'type': 'sell', 'quantity': 25})
Written on Replica 3: (37: {'name': 'nvidia', 'type': 'buy', 'quantity': 13})
Written on Replica 3: (38: {'name': 'gamestop', 'type': 'sell', 'quantity': 5})
Written on Replica 3: (39: {'name': 'meta', 'type': 'buy', 'quantity': 63})
Written on Replica 3: (40: {'name': 'intel', 'type': 'sell', 'quantity': 85})
Written on Replica 3: (41: {'name': 'nike', 'type': 'sell', 'quantity': 38})
Written on Replica 3: (42: {'name': 'apple', 'type': 'sell', 'quantity': 39})
Written on Replica 3: (43: {'name': 'gamestop', 'type': 'sell', 'quantity': 6})
Written on Replica 3: (44: {'name': 'amc', 'type': 'buy', 'quantity': 72})
Written on Replica 3: (45: {'name': 'intel', 'type': 'buy', 'quantity': 17})
Written on Replica 3: (46: {'name': 'amazon', 'type': 'buy', 'quantity': 40})
Written on Replica 3: (47: {'name': 'tesla', 'type': 'sell', 'quantity': 52})
Written on Replica 3: (48: {'name': 'nvidia', 'type': 'buy', 'quantity': 32})
Written on Replica 3: (49: {'name': 'ford', 'type': 'sell', 'quantity': 49})
Written on Replica 3: (50: {'name': 'gamestop', 'type': 'sell', 'quantity': 77})
Written on Replica 3: (51: {'name': 'gamestop', 'type': 'buy', 'quantity': 23})
Written on Replica 3: (52: {'name': 'gamestop', 'type': 'buy', 'quantity': 75})
Written on Replica 3: (53: {'name': 'nvidia', 'type': 'buy', 'quantity': 85})
Written on Replica 3: (54: {'name': 'apple', 'type': 'buy', 'quantity': 39})

ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56890 172.31.16.170 56891 1
Unique ID of this order service: 2
Order server started on: 172.31.16.170:56890
Propagated on Replica 2: (0: {'name': 'tesla', 'type': 'sell', 'quantity': 49})
Propagated on Replica 2: (1: {'name': 'intel', 'type': 'sell', 'quantity': 89})
Propagated on Replica 2: (2: {'name': 'gamestop', 'type': 'buy', 'quantity': 4})
Propagated on Replica 2: (3: {'name': 'intel', 'type': 'sell', 'quantity': 24})
Propagated on Replica 2: (4: {'name': 'apple', 'type': 'buy', 'quantity': 19})
Propagated on Replica 2: (5: {'name': 'amc', 'type': 'sell', 'quantity': 81})
Propagated on Replica 2: (6: {'name': 'gamestop', 'type': 'buy', 'quantity': 33})
Propagated on Replica 2: (7: {'name': 'nike', 'type': 'sell', 'quantity': 62})
Propagated on Replica 2: (8: {'name': 'meta', 'type': 'sell', 'quantity': 9})
Propagated on Replica 2: (9: {'name': 'apple', 'type': 'sell', 'quantity': 69})
Propagated on Replica 2: (10: {'name': 'amc', 'type': 'sell', 'quantity': 97})
Propagated on Replica 2: (11: {'name': 'nike', 'type': 'buy', 'quantity': 88})
Propagated on Replica 2: (12: {'name': 'nvidia', 'type': 'sell', 'quantity': 27})
Propagated on Replica 2: (13: {'name': 'amc', 'type': 'sell', 'quantity': 36})
Propagated on Replica 2: (14: {'name': 'amc', 'type': 'buy', 'quantity': 58})
C\Traceback (most recent call last):
  File "/home/ubuntu/src/order-service/orderServer.py", line 236, in <module>
    serve()
  File "/home/ubuntu/src/order-service/orderServer.py", line 224, in serve
    server.wait_for_termination()
  File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_server.py", line 1118, in wait_for_termination
    return common.wait(self._state.termination_event.wait,
File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_common.py", line 150, in wait
    wait_once(wait_fn, MAXIMUM_WAIT_TIMEOUT, spin_cb)
File "/home/ubuntu/.local/lib/python3.11/site-packages/grpc/_common.py", line 112, in wait_once
    wait_fn(timeout=timeout)
File "/usr/lib/python3.11/threading.py", line 622, in wait
    signaled = self._cond.wait(timeout)
File "/usr/lib/python3.11/threading.py", line 324, in wait
    gotit = waiter.acquire(True, timeout)
KeyboardInterrupt

ubuntu@ip-172-31-16-170:~/src/order-service$ python3.11 orderServer.py 56890 172.31.16.170 56891 1
Unique ID of this order service: 2
Order server started on: 172.31.16.170:56890
Synchronized on Replica 2: (15: {'name': 'ford', 'type': 'sell', 'quantity': 62})
Synchronized on Replica 2: (16: {'name': 'meta', 'type': 'sell', 'quantity': 94})
Synchronized on Replica 2: (17: {'name': 'nike', 'type': 'sell', 'quantity': 59})
Synchronized on Replica 2: (18: {'name': 'nvidia', 'type': 'buy', 'quantity': 73})
Synchronized on Replica 2: (19: {'name': 'ford', 'type': 'buy', 'quantity': 92})
Synchronized on Replica 2: (20: {'name': 'gamestop', 'type': 'buy', 'quantity': 9})
Synchronized on Replica 2: (21: {'name': 'tesla', 'type': 'buy', 'quantity': 68})
Synchronized on Replica 2: (22: {'name': 'apple', 'type': 'sell', 'quantity': 8})
Synchronized on Replica 2: (23: {'name': 'gamestop', 'type': 'buy', 'quantity': 16})
Synchronized on Replica 2: (24: {'name': 'apple', 'type': 'buy', 'quantity': 40})
Synchronized on Replica 2: (25: {'name': 'ford', 'type': 'buy', 'quantity': 42})
Synchronized on Replica 2: (26: {'name': 'ford', 'type': 'sell', 'quantity': 69})
Synchronized on Replica 2: (27: {'name': 'nike', 'type': 'sell', 'quantity': 63})
Synchronized on Replica 2: (28: {'name': 'gamestop', 'type': 'buy', 'quantity': 33})
Synchronized on Replica 2: (29: {'name': 'apple', 'type': 'sell', 'quantity': 88})
Synchronized on Replica 2: (30: {'name': 'nike', 'type': 'buy', 'quantity': 57})
Propagated on Replica 2: (31: {'name': 'apple', 'type': 'sell', 'quantity': 85})
Propagated on Replica 2: (32: {'name': 'nike', 'type': 'sell', 'quantity': 65})
Propagated on Replica 2: (33: {'name': 'amc', 'type': 'buy', 'quantity': 43})
Propagated on Replica 2: (34: {'name': 'intel', 'type': 'sell', 'quantity': 48})
Propagated on Replica 2: (35: {'name': 'gamestop', 'type': 'sell', 'quantity': 93})
Propagated on Replica 2: (36: {'name': 'nvidia', 'type': 'sell', 'quantity': 25})
Propagated on Replica 2: (37: {'name': 'apple', 'type': 'sell', 'quantity': 4})
Propagated on Replica 2: (38: {'name': 'gamestop', 'type': 'sell', 'quantity': 5})
```

Figure 9: Example output when a follower replica crashes during execution and rejoins. Replica ID 3 shown on left, replica ID 2 shown on right.