Part 1: Client and Server Output

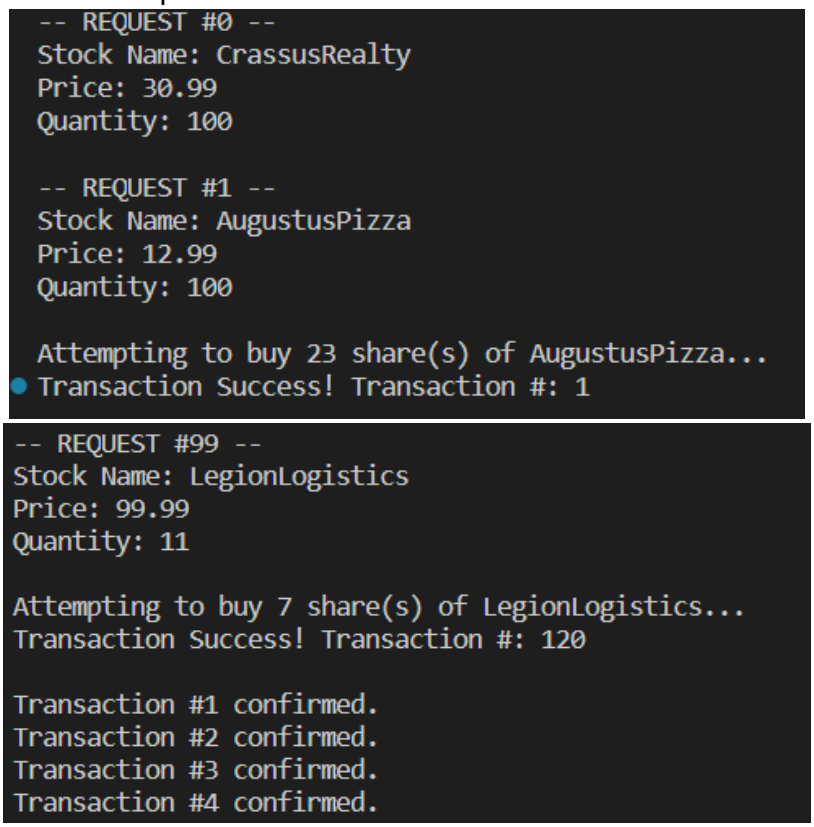


Figure 1: Sample client session for the beginning and end of a session. The bottom picture depicts the end of the transaction where trade requests are confirmed.

```
Found leader! Order Service 3 at ('34.205.23.26', 5004)
 * Serving Flask app 'FrontEndServer' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.31.80.169:5000/ (Press CTRL+C to quit)
108.7.79.28 - - [30/Apr/2023 04:48:28] "GET /stocks/GameStart HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:28] "GET /stocks/DivineComics HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /invalidate/DivineComics HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "POST /orders HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "GET /stocks/CrassusRealty HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "GET /stocks/MenhirCo HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "GET /stocks/MercuryExpress HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /invalidate/MercuryExpress HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "POST /orders HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "GET /orders/0 HTTP/1.1" 200 -
108.7.79.28 - - [30/Apr/2023 04:48:29] "GET /orders/1 HTTP/1.1" 200 -
```

```
 * Serving Flask app 'OrderServer' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.31.80.169:5004/ (Press CTRL+C to quit)
34.205.23.26 - - [30/Apr/2023 04:48:02] "GET /sync HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:06] "GET /sync HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:14] "GET /ping HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /buy HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /sell HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "GET /lookup-order/0 HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "GET /lookup-order/1 HTTP/1.1" 200 -
```

```
 * Serving Flask app 'OrderServer' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.31.80.169:5002/ (Press CTRL+C to quit)
34.205.23.26 - - [30/Apr/2023 04:48:14] "POST /leader-broadcast HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /push HTTP/1.1" 200 -
34.205.23.26 - - [30/Apr/2023 04:48:29] "POST /push HTTP/1.1" 200 -
```

Figure 2: Microservice outputs for the front end and order service replicas. The top image depicts the front end server starting, the middle image depicts the operation of the order service leader, and the last image depicts a typical interaction between the order service leader and replica.

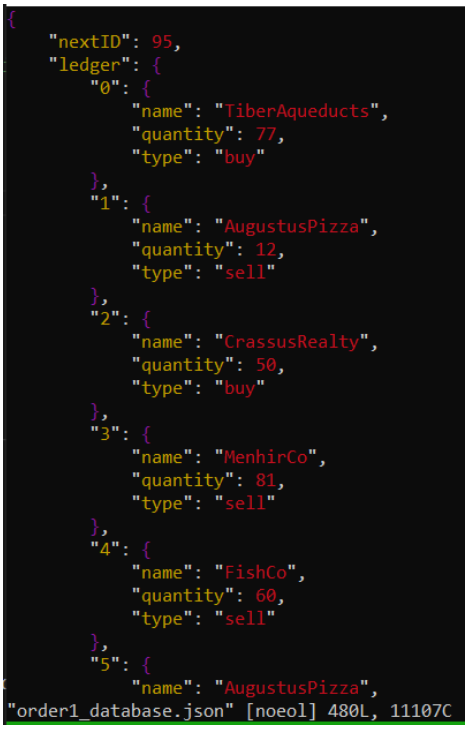
Part 2: Fault Tolerance

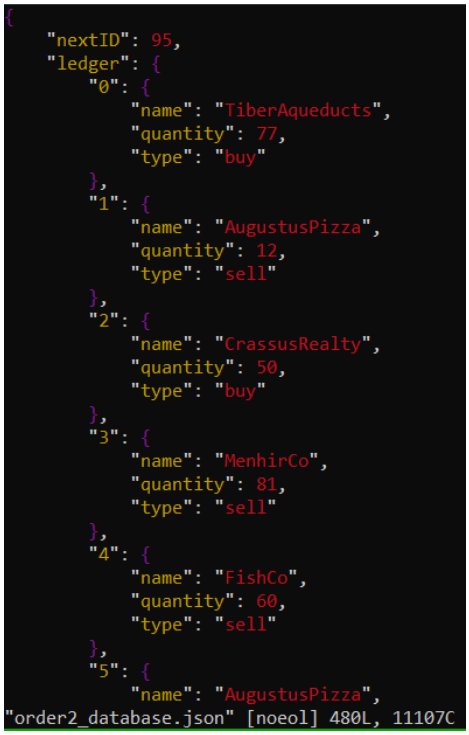
```
-- REQUEST #99 --
Stock Name: TiberAqueducts
Price: 200.99
Quantity: 25

Attempting to sell 6 share(s) of Tibe
rAqueducts...
Transaction Success! Transaction #: 8
6

Transaction #0 confirmed.
Transaction #1 confirmed.
Transaction #2 confirmed.
```

```
-- REQUEST #99 --
Stock Name: FishCo
Price: 19.99
Quantity: 74

Attempting to sell 12 share(s) of Fi
shCo...
Transaction Success! Transaction #:
91

Transaction #3 confirmed.
Transaction #4 confirmed.
Transaction #6 confirmed.
```

```
* Serving Flask app 'OrderServer' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://172.31.88.120:5002/ (Press CTRL+C to quit)
4.230.74.28 - - [01/May/2023 00:11:14] "POST /leader-broadcast HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:52] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:52] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
4.230.74.28 - - [01/May/2023 00:11:53] "POST /push HTTP/1.1" 200 -
CTraceback (most recent call last):
  File "OrderServer.py", line 525, in <module>
    if parent_conn.recv():
  File "/usr/lib/python3.6/multiprocessing/connection.py", line 250, in recv
    buf = self._recv_bytes()
```

Figure 3: Output of client after order replica 1 experiences a crash. Note that the clients were able to successfully finish their sessions despite the failure of one of the order services.





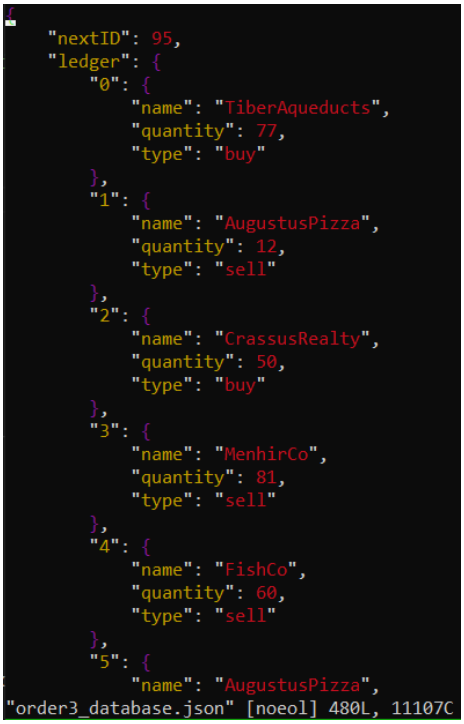


Figure 4: Databases after the crashed database recovered at the end of each client session. Note that all databases are synchronized.

```
-- REQUEST #7 --
Stock Name: DivineComics
Price: 9.99
Quantity: 186

-- REQUEST #8 --
Stock Name: CrassusRealty
Price: 30.99
Quantity: 100

-- REQUEST #9 --
Stock Name: GameStart
Price: 15.99
Quantity: 90

Transaction #0 confirmed.
Transaction #1 confirmed.
Transaction #2 confirmed.
Transaction #3 confirmed.
```

```
Found leader! Order Service 3 at ('34.230.74.28', 5004)
 * Serving Flask app 'FrontEndServer' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.31.88.120:5000/ (Press CTRL+C to quit)
```

```
34.230.74.28 - - [01/May/2023 00:17:12] "POST /invalidate/DivineComics HTTP/1.1" 200 -
Found leader! Order Service 2 at ('34.230.74.28', 5003)
34.230.74.28 - - [01/May/2023 00:17:12] "POST /invalidate/DivineComics HTTP/1.1" 500 -
108.7.79.28 - - [01/May/2023 00:17:12] "POST /orders HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /stocks/CrassusRealty HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /stocks/BoarCo HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /stocks/DivineComics HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /stocks/CrassusRealty HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /stocks/GameStart HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /orders/0 HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /orders/1 HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /orders/2 HTTP/1.1" 200 -
108.7.79.28 - - [01/May/2023 00:17:12] "GET /orders/3 HTTP/1.1" 200 -
```

Figure 5: The client and front end service output both before and after the lead order service crashed. Note here that the client was able to finish its session and retrieve its orders, and the front end was able to make order service 2 the new leader.