# RISK SERVER MANUAL

Abhimanyue Tanwar

| CONTENTS | |
|---|---|
| 1.) | Compilation and running instructions |
| 2.) | Features and Future Prospects |

# Compilation and Running Instructions

1.) Download the zip file, unzip it, and you should see directories named, Client, Server, FinancialInstrument, Order, PortfoliioRiskManager, and Config.

2.) First, compile the server.

```
cd Server
g++ -std=c++17 server.cpp -o server
```

3.) Compile the client.

```
cd ../Client
G++ -std=C++17 client.cpp -o client
```

4.) For running the program, first, we need to start the server, the server can accept connections from a maximum of 30 clients at the moment.

5.) To run the server, go to the Server directory and run the following command.

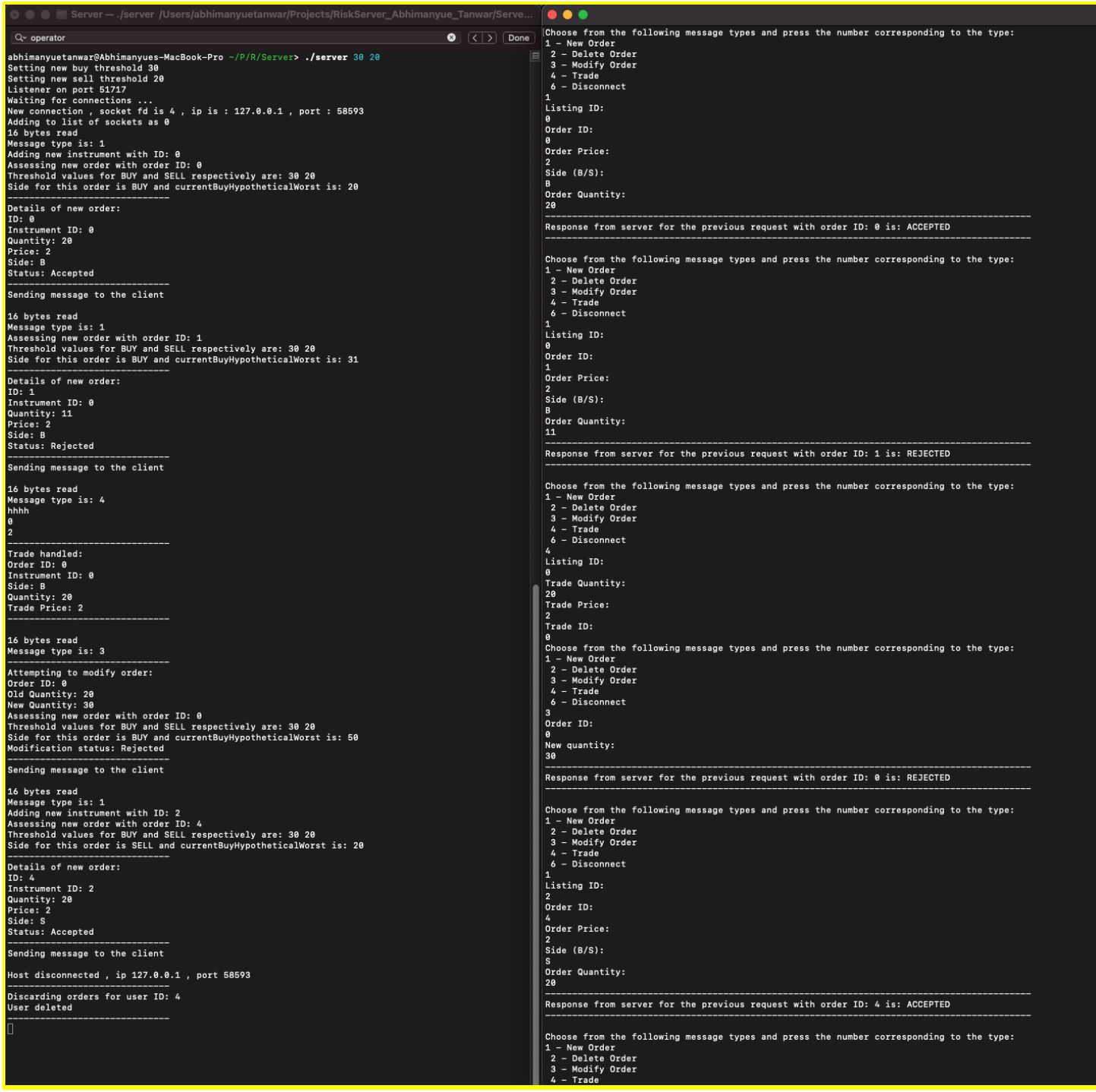
```
./server 30 20
```

The values 30 and 20 are for BUY and SELL thresholds respectively, make sure that the server is up and running and is ready to accept connections from the clients. If the server doesn't run because of the port issue, you can change the port in the server.cpp accordingly and recompile the program.

6.) Now run the client in a separate terminal. Go to the Client directory and run the client program.

```
./client
```

## Sample Flow



```
Server — ./server /Users/abhimanyuetanwar/Projects/RiskServer_Abhimanyue_Tanwar/Serve...
operator
Done
abhimanyuetanwar@Abhimanyues-MacBook-Pro ~/P/R/Server> ./server 30 20
Setting new buy threshold 30
Setting new sell threshold 20
Listener on port 51717
Waiting for connections ...
New connection , socket fd is 4 , ip is : 127.0.0.1 , port : 58593
Adding to list of sockets as 0
16 bytes read
Message type is: 1
Adding new instrument with ID: 0
Assessing new order with order ID: 0
Threshold values for BUY and SELL respectively are: 30 20
Side for this order is BUY and currentBuyHypotheticalWorst is: 20
--------------------------------
Details of new order:
ID: 0
Instrument ID: 0
Quantity: 20
Price: 2
Side: B
Status: Accepted
--------------------------------
Sending message to the client

16 bytes read
Message type is: 1
Assessing new order with order ID: 1
Threshold values for BUY and SELL respectively are: 30 20
Side for this order is BUY and currentBuyHypotheticalWorst is: 31
--------------------------------
Details of new order:
ID: 1
Instrument ID: 0
Quantity: 11
Price: 2
Side: B
Status: Rejected
--------------------------------
Sending message to the client

16 bytes read
Message type is: 4
hhhh
0
2
--------------------------------
Trade handled:
Order ID: 0
Instrument ID: 0
Side: B
Quantity: 20
Trade Price: 2
--------------------------------

16 bytes read
Message type is: 3
--------------------------------
Attempting to modify order:
Order ID: 0
Old Quantity: 20
New Quantity: 30
Assessing new order with order ID: 0
Threshold values for BUY and SELL respectively are: 30 20
Side for this order is BUY and currentBuyHypotheticalWorst is: 50
Modification status: Rejected
--------------------------------
Sending message to the client

16 bytes read
Message type is: 1
Adding new instrument with ID: 2
Assessing new order with order ID: 4
Threshold values for BUY and SELL respectively are: 30 20
Side for this order is SELL and currentBuyHypotheticalWorst is: 20
--------------------------------
Details of new order:
ID: 4
Instrument ID: 2
Quantity: 20
Price: 2
Side: S
Status: Accepted
--------------------------------
Sending message to the client

Host disconnected , ip 127.0.0.1 , port 58593
--------------------------------
Discarding orders for user ID: 4
User deleted
--------------------------------
```

```
Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
 6 - Disconnect
1
Listing ID:
0
Order ID:
0
Order Price:
2
Side (B/S):
B
Order Quantity:
20
--------------------------------------------------------------------------------
Response from server for the previous request with order ID: 0 is: ACCEPTED
--------------------------------------------------------------------------------

Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
 6 - Disconnect
1
Listing ID:
0
Order ID:
1
Order Price:
2
Side (B/S):
B
Order Quantity:
11
--------------------------------------------------------------------------------
Response from server for the previous request with order ID: 1 is: REJECTED
--------------------------------------------------------------------------------

Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
 6 - Disconnect
4
Listing ID:
0
Trade Quantity:
20
Trade Price:
2
Trade ID:
0
Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
 6 - Disconnect
3
Order ID:
0
New quantity:
30
--------------------------------------------------------------------------------
Response from server for the previous request with order ID: 0 is: REJECTED
--------------------------------------------------------------------------------

Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
 6 - Disconnect
1
Listing ID:
2
Order ID:
4
Order Price:
2
Side (B/S):
S
Order Quantity:
20
--------------------------------------------------------------------------------
Response from server for the previous request with order ID: 4 is: ACCEPTED
--------------------------------------------------------------------------------

Choose from the following message types and press the number corresponding to the type:
1 - New Order
 2 - Delete Order
 3 - Modify Order
 4 - Trade
```

# Features and Future Prospects

1.) The program is divided into multiple parts for ease of readability and maintainability. A lot of components have been modeled as header-only, to provide library-like support.

2.) Have utilized C-based socket programming based on the client-server model and it runs a TCP server. The server can accept multiple clients at the same time.

3.) Have used smart pointers to minimize memory management and leaks.

4.) Used const refs as much as possible in function parameters to reduce copies.

Future Prospects:

1.) Need to write extensive unit tests. (tried writing, but ran out of time.)

2.) Make the program more robust in terms of memory management and figure out ways to reduce latency.

3.) Create a separate portfolio for each trader and manage risks per trader.