Contents

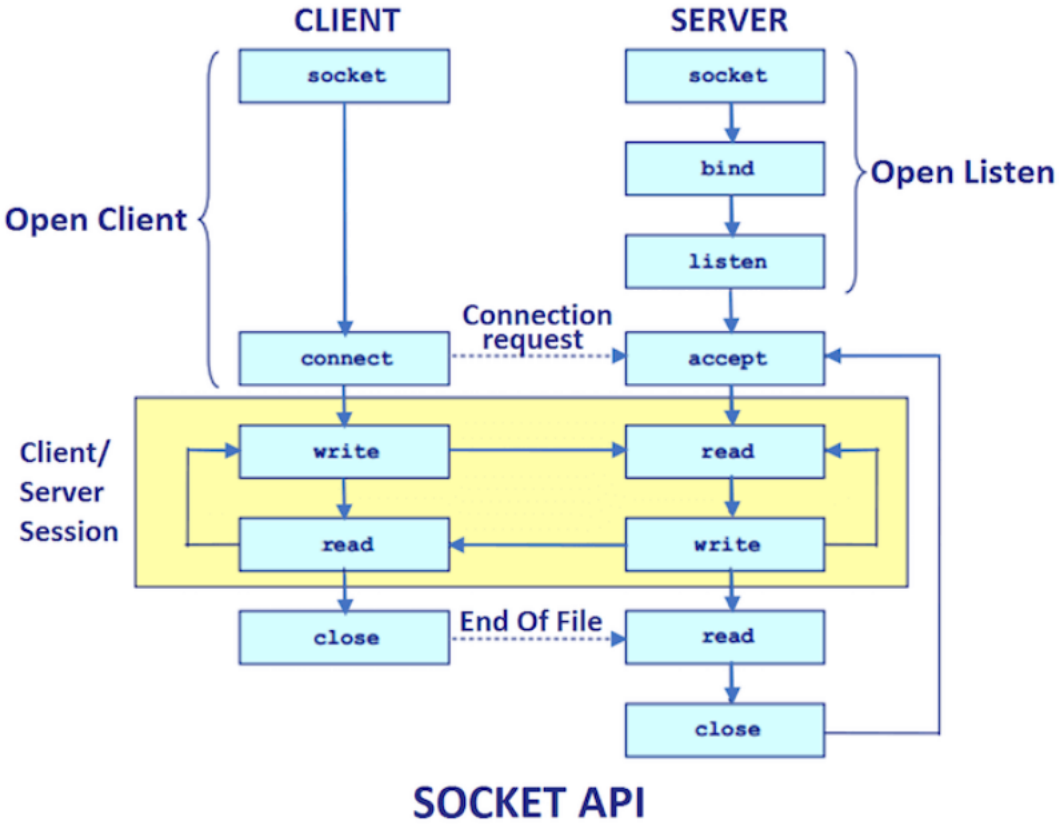
CS558 Client Server Trading System

Paila Mouli Swaroop — 214101035, Patel Miki Maheshbhai — 214101036, and Prateekshya Priyadarshini — 214101037

M.Tech CSE

1 Introduction

This is a client server trading application which has a set of traders who trade with each other. There is a set of items which the traders can trade. There is only one server which performs the functions of order processing and trade matching in addition to acknowledging logins by clients and servicing their requests.



2 Data Structures

2.1 Trader

2.1.1 Properties

- name – Represents the name of trader
- password – Represents password of trader
- username – Represents unique name assigned to trader
- isLoggedIn – Represents whether the trader is currently logged in or not
- uniqueNumber – Represents unique number assigned to each trader
- all_trades – Stores all trades performed by trader

2.1.2 Functions

- setLoginStatus – sets the login status (isLoggedIn)
- setNumber – sets the uniqueNumber
- getLoginStatus – returns login status
- getNumber – returns uniqueNumber
- getName – returns name of trader
- getUserId – returns userId (username) of trader
- getPassword – returns password of trader
- getAllTrades – returns the set of orders performed by trader
- add_order – adds new order to the list of orders

2.2 Order

2.2.1 Properties

- trader – Represents the trader that added the order to system
- match – Represents the trader that fulfilled the order
- orderId – Represents the unique order id of the order in system
- quantity – Represents the order quantity
- price – Represents the order price
- type – Represents the type of order (BUY - 0 or SELL - 1)
- status – Represents the status of order (FILLED or QUEUED)

2.2.2 Functions

- getPrice – returns the price of order
- printOrderDetails – prints the order details
- set_trader – sets the trader that performed order
- set_match – sets the trader that matched the order
- set_orderId – sets the unique order id of order
- set_quantity – sets the quantity of order
- set_price – sets the price of order
- set_type – sets the type of order
- set_remaining_quantity – sets the remaining quantity of order
- set_status – sets the status of order
- get_status – returns the status of order
- get_trader – returns the trader that made the order
- get_match – returns the trader that fulfilled the order
- get_order_id – returns the order id
- get_item_index – returns the index of item for which order is made
- get_quantity – returns the quantity of order
- get_price – returns the price of order
- get_type – returns the type of order
- get_remaining_quantity – returns the remaining quantity of order

2.3 Item

2.3.1 Properties

- itemId – represents the unique id of item
- price – represents the current (last traded) price of item
- itemName – represents the name of item
- buy_book – represents the order list on buy side in descending order
- sell_book – represents the order list on sell side in ascending order

2.3.2 Functions

- insertItem – inserts the new order for the item
- printItemQueue – prints the buy and sell order book on console
- getBuyBook – returns the buy_book of the item
- getSellBook – returns the sell_book of the item
- getBuyBookSize – returns the buy_book size of the item
- getSellBookSize – returns the sell_book size of the item
- getItemName – returns the name of item
- add_buy_order – adds new order to buy_book in descending order
- add_sell_order – adds new order to sell_book in ascending order
- trade_orders – this function matches the orders in buy_book and sell_book recursively until no other match is possible.

3 Modules

3.1 Client-Server architecture and flow

When the server starts, a designated port is assigned to the server, on which server accepts new requests from clients. When the client starts, IP address and port number of server is given to it. Client sends request to create connection on that IP address on a given port. Once server receives the request from client, it accepts the connection and assign that client a new port, with which client will communicate for further tasks.

3.1.1 Login

A server accepts trader logins using different clients. A trader can login using at most one client at a time. Multiple traders are allowed in one client system. So, this makes it one to many

3.1.2 Registration

A trader can register using clients. Trader provides name, username and password while registering from client. Server stores the credentials in a file and allow the trader to perform trades in available Items. Currently at most five registrations are allowed.

3.1.3 Buy

This function takes the details of an order i.e. item name, price, quantity and adds a buy order in the queue.

3.1.4 Sell

This function takes the details of an order i.e. item name, price, quantity and adds a sell order in the queue.

3.1.5 Order Status

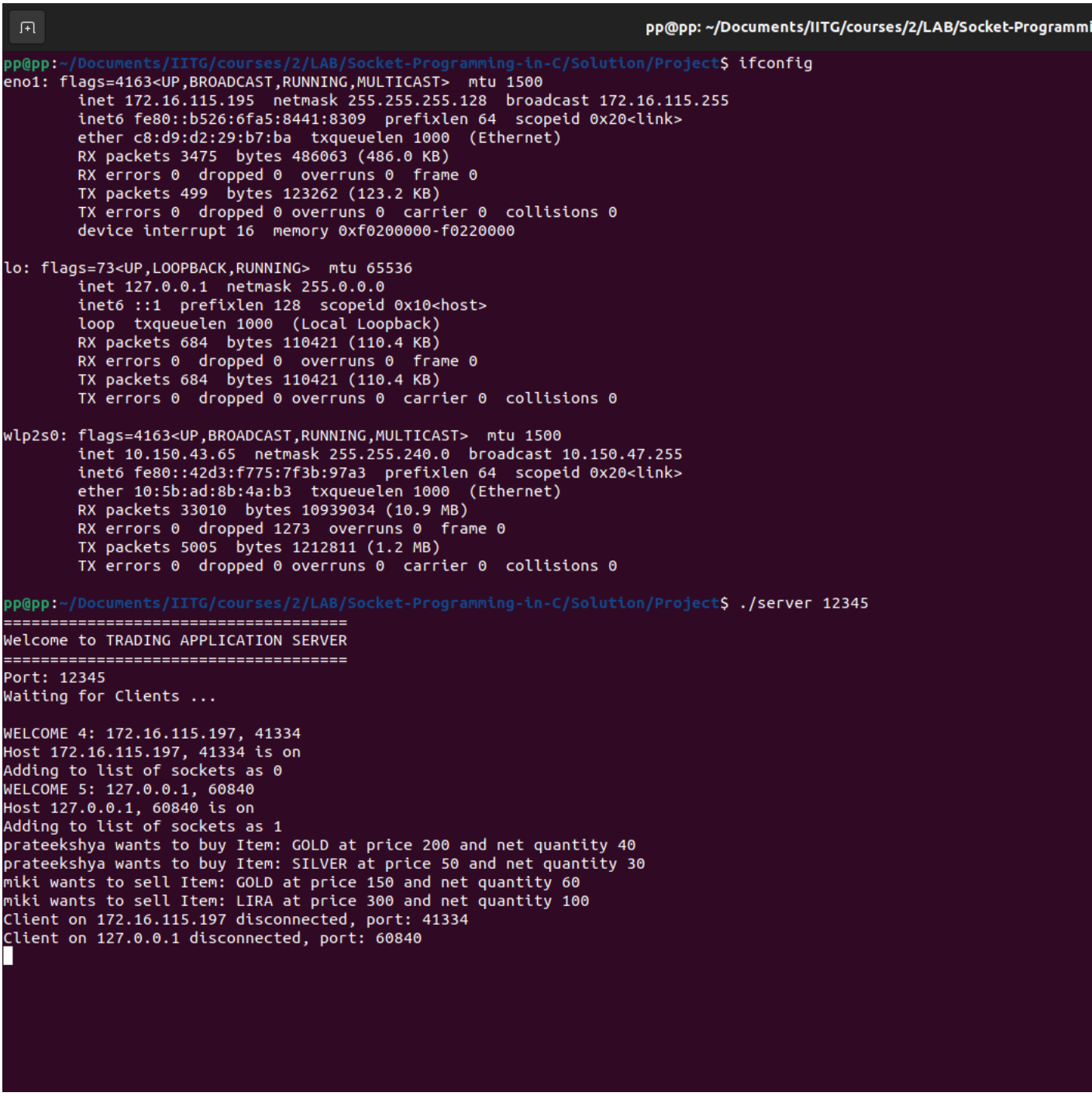
This function shows all the positions of buy and sell orders in the system i.e. the current best sell price and best buy price for each item.

3.1.6 Trade Status

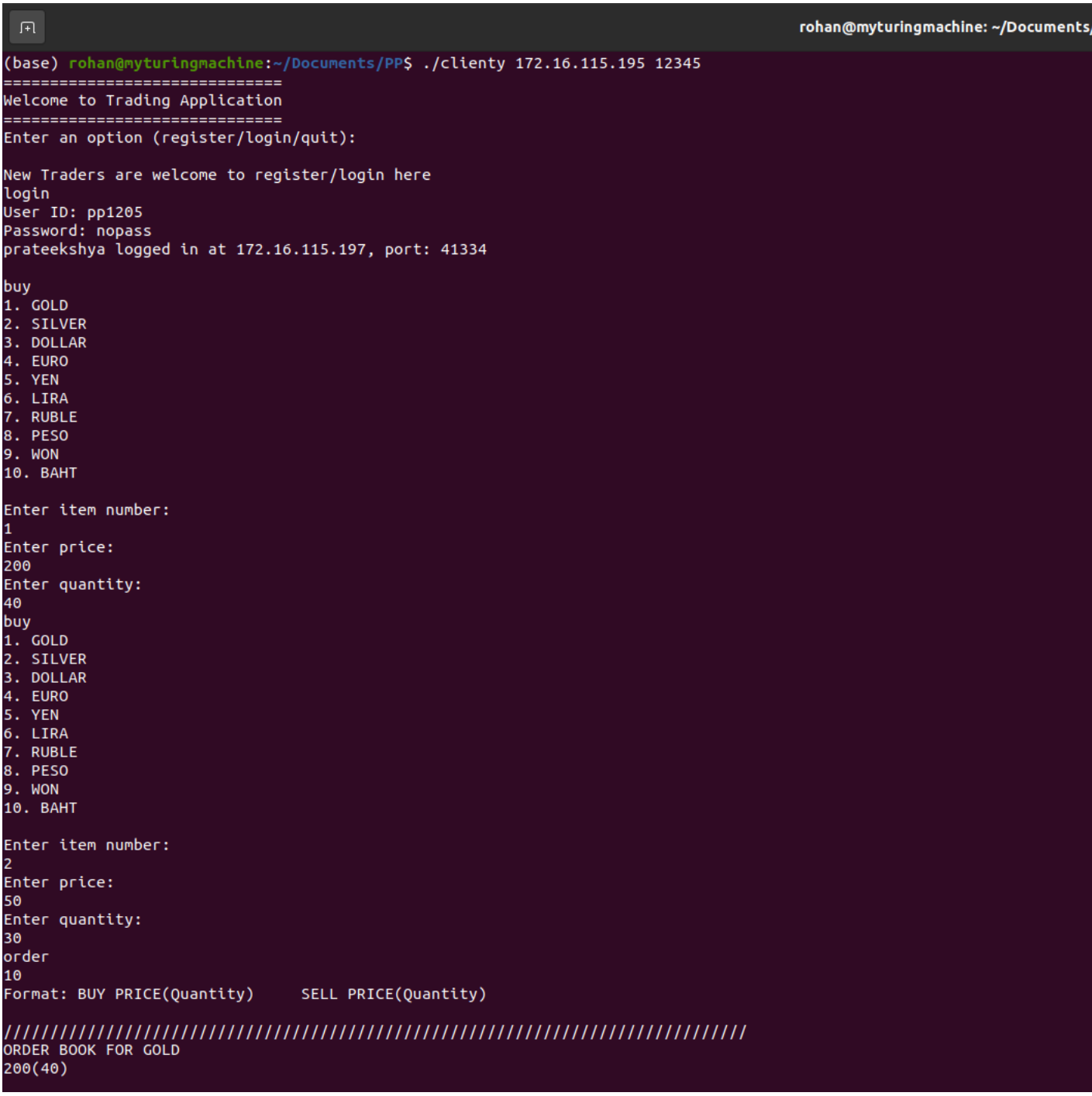
This function shows the matched trades of a particular trader.

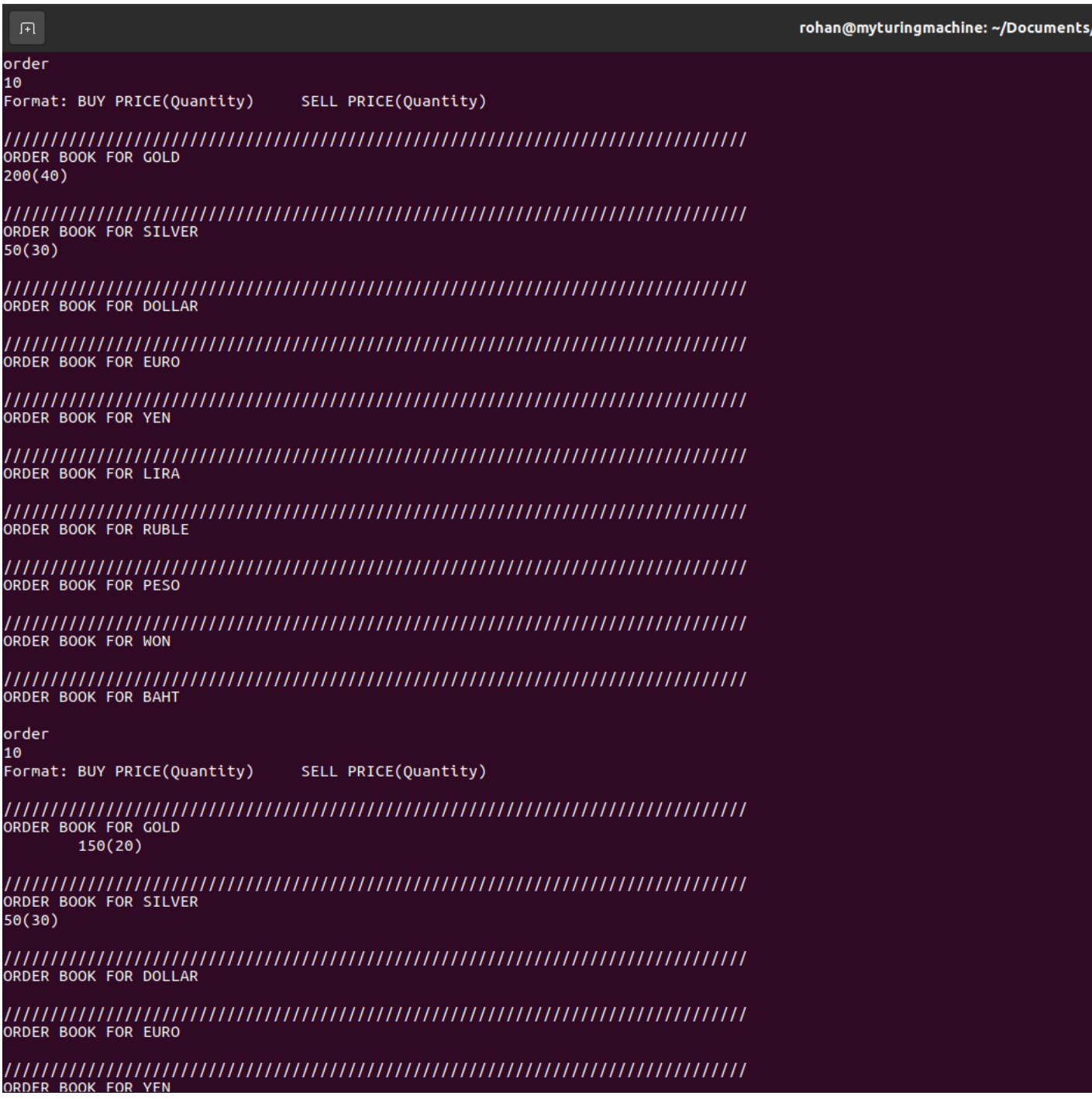
4 Workflow Example

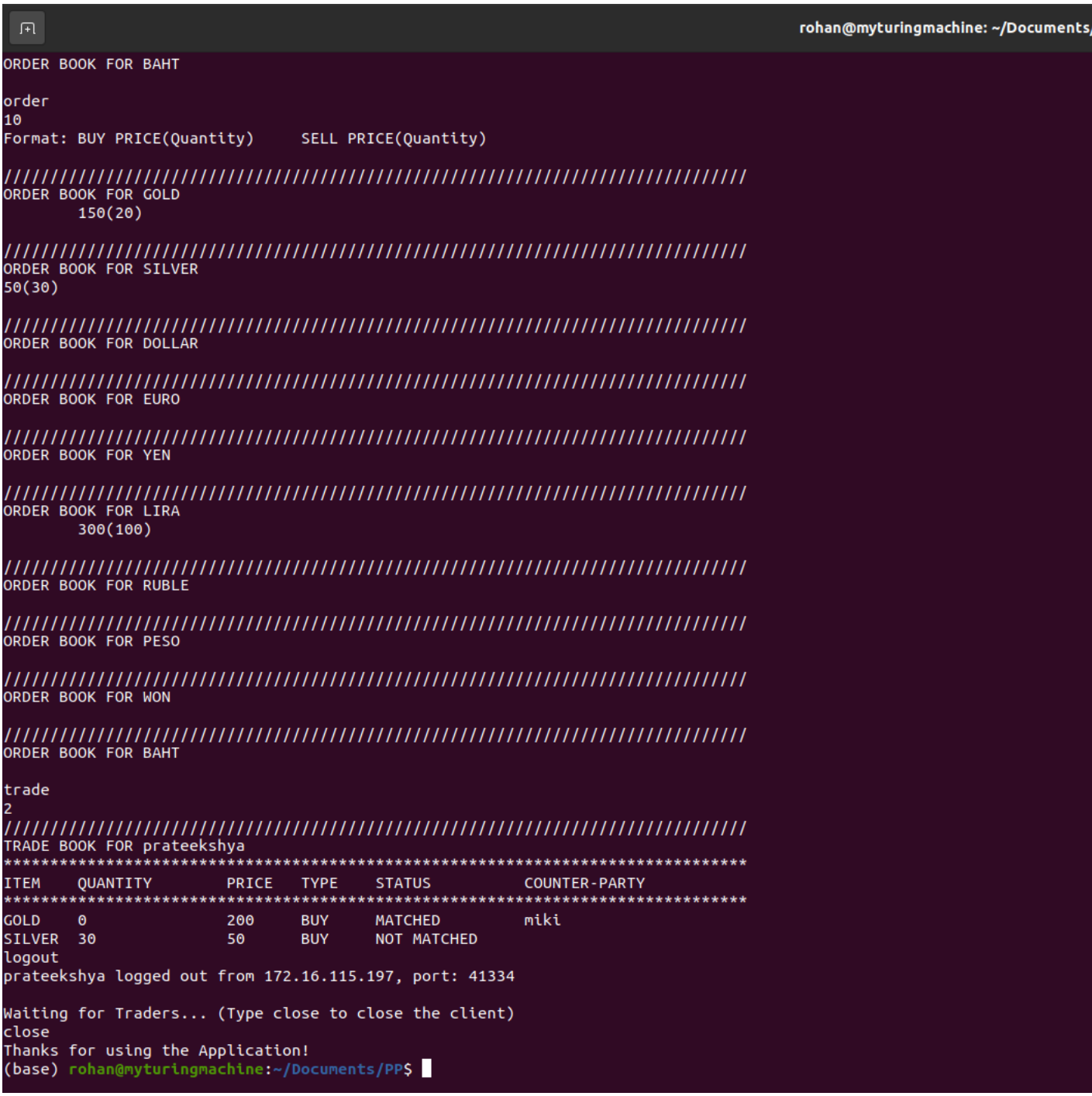
4.1 Server at 172.16.115.195



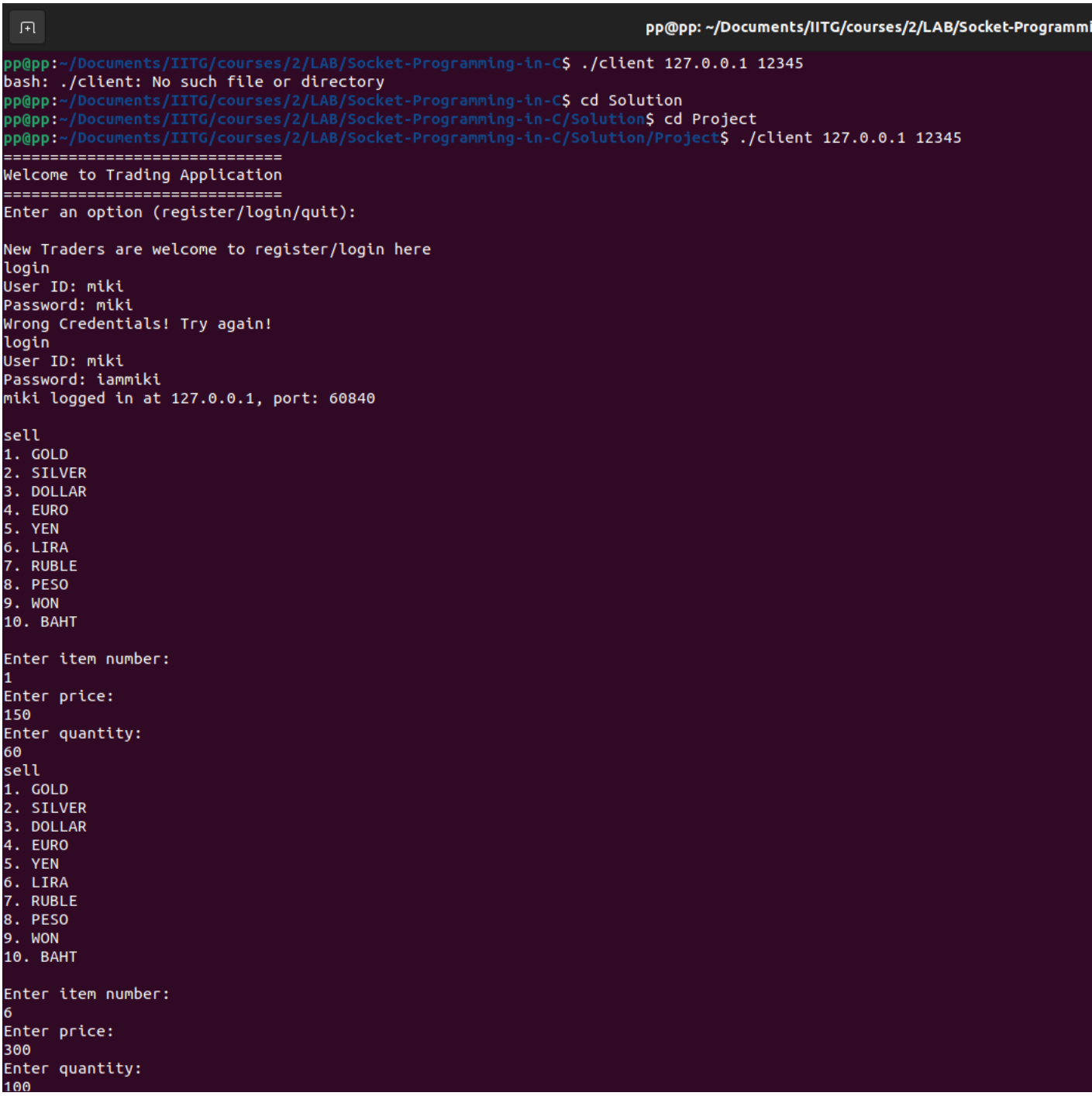
4.2 Client at 172.16.115.197

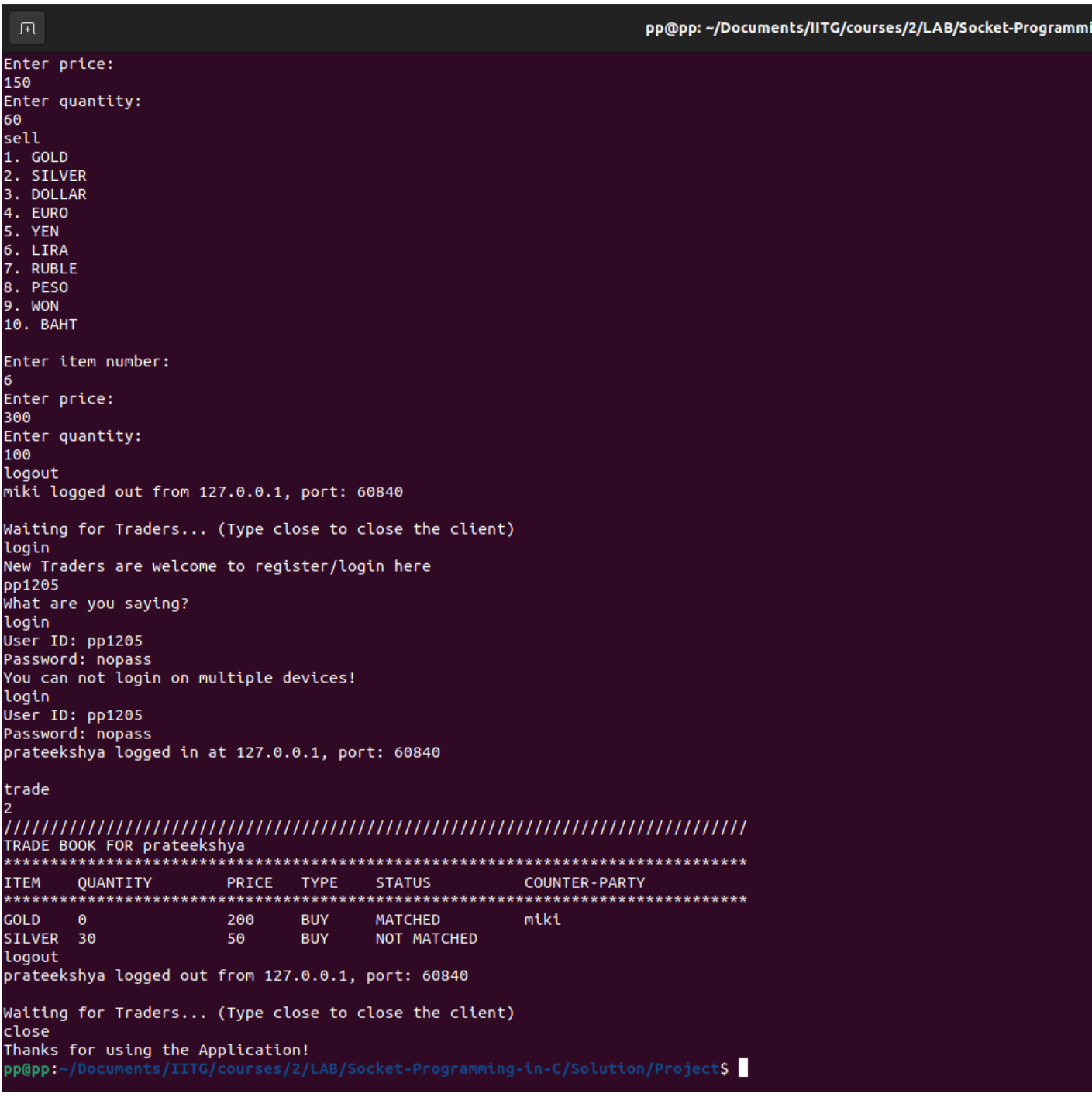



rohan@myturingmachine: ~/Documents/
order
10
Format: BUY PRICE(Quantity) SELL PRICE(Quantity)
//
ORDER BOOK FOR GOLD
200(40)
//
ORDER BOOK FOR SILVER
50(30)
//
ORDER BOOK FOR DOLLAR
//
ORDER BOOK FOR EURO
//
ORDER BOOK FOR YEN
//
ORDER BOOK FOR LIRA
//
ORDER BOOK FOR RUBLE
//
ORDER BOOK FOR PESO
//
ORDER BOOK FOR WON
//
ORDER BOOK FOR BAHT
order
10
Format: BUY PRICE(Quantity) SELL PRICE(Quantity)
//
ORDER BOOK FOR GOLD
150(20)
//
ORDER BOOK FOR SILVER
50(30)
//
ORDER BOOK FOR DOLLAR
//
ORDER BOOK FOR EURO
//
ORDER BOOK FOR YEN


rohan@myturingmachine: ~/Documents/
ORDER BOOK FOR BAHT
order
10
Format: BUY PRICE(Quantity) SELL PRICE(Quantity)
//
ORDER BOOK FOR GOLD
150(20)
//
ORDER BOOK FOR SILVER
50(30)
//
ORDER BOOK FOR DOLLAR
//
ORDER BOOK FOR EURO
//
ORDER BOOK FOR YEN
//
ORDER BOOK FOR LIRA
300(100)
//
ORDER BOOK FOR RUBLE
//
ORDER BOOK FOR PESO
//
ORDER BOOK FOR WON
//
ORDER BOOK FOR BAHT
trade
2
//
TRADE BOOK FOR prateekshya
****
ITEM QUANTITY PRICE TYPE STATUS COUNTER-PARTY
****
GOLD 0 200 BUY MATCHED miki
SILVER 30 50 BUY NOT MATCHED
logout
prateekshya logged out from 172.16.115.197, port: 41334
Waiting for Traders... (Type close to close the client)
close
Thanks for using the Application!
(base) rohan@myturingmachine:~/Documents/PP$

4.3 Client at 172.16.115.195





We can see that the trades are matched and displayed properly. One trader can login from one client at a time and one client can handle multiple logins one after the other.

5 Steps to run the code

1. Use linux and make sure that all the systems that are being used are connected to a single network.

2. Run Server on one system with command ./server ⟨server_port⟩.

3. Run Clients on other systems with command ./client ⟨server_ip⟩ ⟨server_port⟩.

4. Client can type the following commands

   (a) **register**
   Using this command a new trader can register to the trading application.

   (b) **login**
   Using this command an existing trader can login to the application.

   (c) **buy**
   This command is used to place a buy order.

   (d) **sell**
   This command is used to place a new sell order.

   (e) **order**
   This command shows the current best buy price and sell price for each item.

   (f) **trade**
   This command shows the matched orders for a trader.

   (g) **logout**
   This command is used to logout from the application.

   (h) **close**
   This command is used to close the client.

   Any other command will print **What are you saying?**

5. Follow the instructions printed on terminal properly to avoid errors during execution.