Problem Statement

Implement a high-frequency Exchange that receives both BUY orders & SELL orders and matches them based on the Symbol, Price and the sequence in which they were received. For each successful match, it creates a "Trade" record between the two parties that includes the two parties (Buyer & Seller) along with attributes such as the Symbol, Price, Trade Date, etc.

E.g.

Given an orders feed with the following rows:
Party A, SELL, IBM, 110
Party A, SELL, INFY, 600
Party A, SELL, GOOG, 500
Party B, BUY, IBM, 110
Party C, BUY, IBM, 110
Party C, BUY, INFY, 600

The following two matched trades will be created:

Trade 1:
Seller = Party A, Buyer = Party B, Stock = IBM, Price = 110, Trade Date = 06/26/2020
Trade 2:
Seller = Party A, Buyer = Party C, Stock = INFY, Price = 600, Trade Date = 06/26/2020

You can assume any channel in which these orders are being fed into the Exchange - a Queue, HTTP end-point or even a mock interface. Please assume large volumes of orders coming in through the channel at a short time. Hence care must be taken to design the matching system to have high throughput.

Expose two REST end-points for the end-users to query:
1. List of trades with the ability to filter based on Parties, SYMBOL or Date
2. List of orders that are not yet matched, with the ability to filter based on SYMBOL, Price

General Instructions

- Follow clean coding practices
- Include automated tests
- Submit the solution via a new repository in your GitHub account
- Include a README file in the repo to document key design decisions as well as instructions to set up and run it in our environment
- Include any supporting files (such as any test data used for development) that are needed to run and test the application