Marketplace

We have been asked to build a marketplace which brings together buyers and sellers of goods. The marketplace works as follows:

- Sellers *offer* items for sale. An *offer* consists of an item ID, a user ID, a quantity and a price per unit the seller is willing to sell at
- Buyers *bid* for items. A *bid* consists of an item ID, user ID, a quantity and a price per unit the buyer is willing to pay
- Sellers can enter offers even if there are no buyers for that item in the market
- Buyers can enter bids even if there are no sellers in the market for that item
- A bid and offer match when:
  - The bid item ID and sell item ID are the same
  - The bid price is greater than or equal to the offer price
  - The offer quantity is greater than or equal to the bid quantity
- If a bid matches many offers it should be matched against the first received offer
- If an offer matches many bids it should be matched against the first received bid
- When a bid and offer match:
  - An *order* is created. An *order* consists of a buyer ID, a seller ID, an item ID, a quantity, and a price per unit. The price of the order is the minimum of the matched bid and offer price
  - The bid is removed from the marketplace
  - The offer quantity is reduced by the amount of the matching bid. If the offer quantity is reduce to 0, it is removed from the marketplace
- Bids and offers stay in the marketplace until they are matched

As a first step we have been asked to implement this marketplace, providing the following API:

- Add a bid
- Add an offer
- List the bids for a given buyer user ID
- List the offers for a given seller user ID
- List the orders for a given seller user ID
- List the orders for a given buyer user ID
- Retrieve the current bid price for a given item ID (i.e. the highest price of all bids for that item)
- Retrieve the current offer price for a given item ID (i.e. the lowest price of all offers for that item)

Please see examples on the following page.

Please provide an implementation of the requirements above. No persistence or UI is required; an in-memory solution providing an API is absolutely fine. The solution should include everything you would check in to your source code management system, and the code should be of production quality. I would strongly recommend using IntelliJ for this problem.

Bids	Offers	Orders for user 1	Current Bid Price for item 12345	Current Offer Price for item 12345
Bid entered: itemId: 12345, quantity: 10, pricePerUnit: 25, user: 1				
`{ itemId: 12345, quantity: 10, pricePerUnit: 25, user: 1 }`	`[]`		`25`	
Offer entered: itemId: 12345, quantity: 5, pricePerUnit: 25, user: 2 // not matched as offer quantity < bid quantity				
`{ itemId: 12345, quantity: 10, pricePerUnit: 25, user: 1 }`	`{ itemId: 12345, quantity: 5, pricePerUnit: 25, user: 2 }`		`25`	`25`
Offer entered: itemId: 12345, quantity: 10, pricePerUnit: 24, user: 3 // matched				
	`{ itemId: 12345, quantity: 5, pricePerUnit: 25, user: 2 }`	`{ itemId: 12345, quantity: 10, pricePerUnit: 24, buyerId: 1, sellerId: 3 }`		`25`
Bid entered: itemId: 12345, quantity: 5, pricePerUnit: 23, user: 4 // not matched as bid price < offer price				
`{ itemId: 12345, quantity: 5, pricePerUnit: 24, user: 4 }`	`{ itemId: 12345, quantity: 5, pricePerUnit: 25, user: 2 }`	`{ itemId: 12345, quantity: 10, pricePerUnit: 24, buyerId: 1, sellerId: 3 }`	`24`	`25`