Illustration - 1

Trading the Mad Cow

Hello code ninja , your mission is to implement *imitation* of a real trading market, while the asset being traded is a mad cow.

What you see on the illustration is just for a sense of the intuition, formally the on UI screen will be providing that information:

1. Current Price
2. Table of buy/sell bids with last 10 bids on each side
3. Table of last deals been made

All the information should be dynamicly updated.

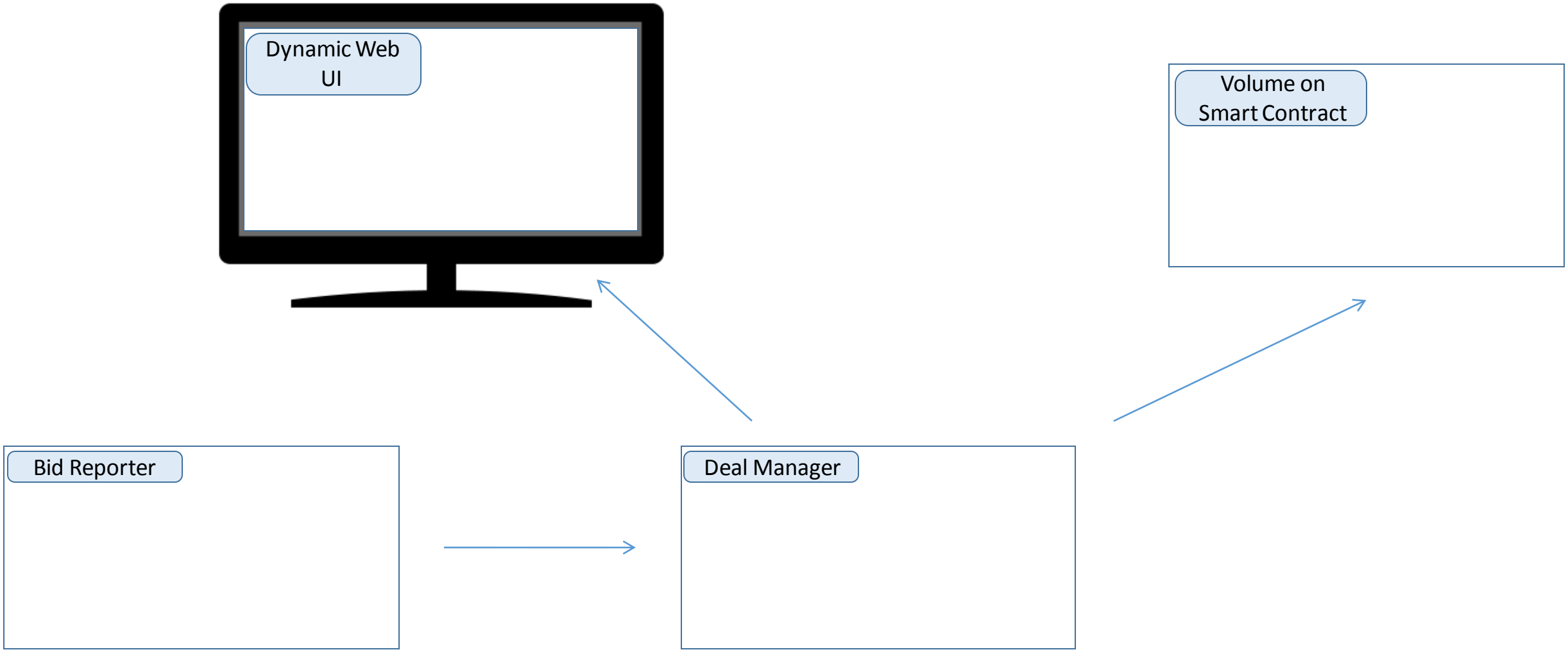

Illustration - 2

Backend Scheme:

1. **Bid Reporter** - backend service reporting randomly generated bids

2. **Deal Manager** - will get generated bids from Bid Reporter and manage the actual state of the asset trading , if the deal can be created from the reported bids it will be created.

3. **Dynamic UI** will reflect the state of the Deal Manager

4. **Volume** is important data and will be reported into a smart contract once each 1 minute. As map of time and cumulative volume until some point of time.

Components Description

Bid Reporter - will be the independent server for generating random (but good simulation) bids for the whole system. You can use any technique for the reporter communication design with other parts of the system.

Deal Manager - That is the heart of the system. The main server that will manage all the current state of the system and reflect it to the users.

The Bids from the Bid Reporter will be received by the Deal Manager and affect the state calculation. The information about new deals, current bids enlisted ledger, volume and the last price it's all the responsibility of the Deal Manager.

We are also connecting the Deal Manager to the Ethereum Block chain, to save the important info on the de-centralized state.

Components Description (continues...)

Dynamic UI - The UI is your module to show the ninja skills on reflecting information for the user. You have the sense of what should be on the UI from the first illustration but it is up to you how the look and feel of the system will be done eventually.

The Deal Manager work reflection in nice and dynamic way is the must , but don't stop there , you obviously can come up with your own ideas and good improvisation is a big plus for your work.

Formally: The UI reflects Bids submitted to buy the assets and Bids submitted to sell the assets (as generated by Bid Manager).

As well it is important to reflect The done deals outcome as generated by the Deal Manager and the current price as it is on some point of time.

Components Description (continues...)

Volume on the Blockchain – You should develop a smart contract that will be holding the volume info reported by the Deal Manager.

The information will be minimalistic only the date/time of the report and the volume of the asset traded until that point.

The information will be reported once a 1 minute and the users will be following the dynamic info on: https://test.ether.camp

What should you use to develop ?

The backend should be developed by using Java / Spring Boot technology and the frontend using JavaScript / Angular / Bootstrap or any better browser UI technology that you know .

The blockchain part should be develop in Solidity using Ethereum Studio to deploy it to the Ethereum test network.

Where to study the info you don't know ?

All the information you might need for the development of the system is on the open Web. However some of the info is intentionally new for you and the idea is to check how you can study new technology in short time.

You can use any forum available on the web especially might be helpful to chat on <ether.camp> slack (auto invite on www.ether.camp)

What should be provided by you:

✓ **The full source code** – you should disclose on a ***private*** github repository full code of all the system including all the modules being developed.

✓ **Documentation** – you should well document the code and also provide a comprehensive description on how to run the system

✓ **Running components** – You should use any available cloud service for making the components run at least couple of days , for us to test the system. We also will ask you for u/p to the servers to see how the system is operating on the machines.

Good luck and remember any Idea you will add to the project will be noticed and appreciated