Lab 5: Angular, Node och Express

Översikt
Skapa en trading-sida för aktier med hjälp av Angular, Node och Express. Om du tänker göra bonusuppgiften så kan det vara bra att göra dem samtidigt som grunduppgiften. Webbsidan är tänkt att simulera handel med aktier mellan olika människor.
Webbsidan hanterar tre sorters objekt. Securities (Aktier, värdepapper, osv), Orders (aktiva köp och sälj-ordrar samt Trades (historik över avslutade köp).

Användaren ska få möjligheten att kunna utföra följande tre uppgifter:

- Lägga till en Security: t.ex. Lägga till aktien **Ericsson**.
- Lägga till en Order: t.ex. lägg a till en köporder för 10 st **Nokia** aktier för 15 kr styck.
- Lista Trades: t.ex. Lista alla köp som har gjorts med **Ericsson** aktier.

Ett köp är avslutat när en köporder matchas med pris för en sälj-order. Man ska matcha med den äldsta av ordrarna och om inte ordrarna är lika stora så fortsätter matchningen tills att antingen alla är matchade eller tills att det inte finns några fler ordrar med rätt pris. Efter det så läggs eventuella rester av orden in i systemet.

Datan (databasen för bonus-uppgiften) ska ha följande struktur:

Security: // security betyder värdepapper

- int id;
- String name;

Order:

- int id;
- String name; // one of the securities above
- String type; // “buy” or “sell”
- double price;
- int amount;
- String uid // the name of uid (customer) placing order

Trade:

- int id;
- String name; // one of the securities above
- double price;
- int amount;
- Date dt; // at what date and time was this trade created.
- String buyer; // ett uid (customer) från buy-order
- String seller; // ett uid (customer) från sell-order

Specifika krav:

- Multipla Orders kan matcha samma nya order och resterna läggs in i Orders.
- MVC (MVW, MV*, …) används ordentligt.
- Du har en förståelse för hur komponenterna hänger ihop och varför de är uppdelade på det sättet de är.
- När man väljer ut en aktie som ska visas så skickas det som en AJAX-request ($http.get) och datan visas upp på en sido-ruta som sen uppdateras vid nya entries via websockets. Detta gäller både för Trades och Orders.

Tips:

- Börja med att experimentera med det givna exemplet innan ni börjar med själva uppgiften. 60% av arbetet är att förstå hur saker hänger ihop.
- Utgå från exemplet och gör små inkrementella ändringar för att göra om exemplet till något som löser uppgiften. Angular kan vara ganska svårt att debugga i början om man gjort stora ändringar.

Testsekvens:
Det här är den testsekvens som vi normalt testar med. (använd samma pris men olika antal för all orders)

1. Skapa en ny security
2. Lägg upp 4 st köpordrar för 1 aktier. (en i taget)
3. Lägg upp 1 säljorder för 10 aktier
4. Kolla att 4 st köpordrar blev uppfyllda
5. Läpp upp 2 st köpordrar för 1 aktie (en i taget)
6. Kolla att de två köpordrarna blev uppfyllda
7. Lägg upp 2 st ordrar på 4 st aktier var (en i taget)
8. Kolla att en av dem blev uppfylld och att den andra är kvar.

Anteckning:
Lägg märke till att strukturen på koden ni får är väldigt förenklad och det är fritt fram att göra valfria förbättringar på den. Det finns väldigt mycket material om man bara söker lite.

Frågor:
Vad är ett promise?
Vad är dependency injection?

Bonus
Använd dig av en ORM/ODM och spara all data i en databas. Ni får med ett db.sql script i repot.

Questions:
Varför vill man använda websockets? Vad löser det för problem?
Vad kan man göra om websockets inte finns (för gammal webbläsare eller något liknande)?