DT00BN92 Development of Client-side Interactive Web Applications

# Fall 2019
# Project 2

**You can collect up to 60 points for this assignment**

# ReactJS

***This course work is individual.*** *Discussing the course tasks, assignments and specific issues with other course participants is allowed and even encouraged. However, you should be the only author of all the solutions you provide in this assignment. Team work, pair programming or copying solutions or program code from other persons is consider plagiarism and it will be handled following the Åbo Akademi protocol for such cases.*

## Project Description

The customer needs an application to visualize the performance of stock portfolios. The working name of the software to be developed is **SPMS**, short for *Stock Portfolio Management System.* In the program, a user can manage several portfolios. A **portfolio** is a collection of stocks. A **stock** is a collection of shares of a given corporation (identified by a symbol of 3 or 4 letters – for instance, NOK for Nokia, or MSFT for Microsoft). A **share** is an atomic unit of ownership of a company characterized by a *symbol* and a *value.*

## Assumptions:

- There is only one user for the current application. No authentication is required.

## Functional Requirements of SPMS

1. Create portfolio **10pt**
    a. The user can create an empty portfolio
    b. User should be able to enter the portfolio name
    c. A user should be able to create a decent number (min. 5-10) of portfolios
2. Add stock to portfolio **10p**
    a. User must be able to add a stock to the portfolio by entering:
        i. the **symbol** of the stock
        ii. the **number of shares** he/she owns
        iii. the **date of purchase**
        ~~the current value of one share at that moment as entered by the user (for instance the purchase value – no API needs to be contacted).~~
    b. There is no limit on the number of stocks a portfolio can contain
3. View portfolio **10pt**
    a. The user must be able to view **the purchase value** and the **current value of the stocks** in the portfolio. The ~~current values~~ **current value** and the **purchase value** of the stocks should be fetched from a stock market exchange API. For simplicity, a Refresh button can be used.

b. The user must be able **to change the currency** between US dollar and Euro when displaying a portfolio (and its stocks). For simplicity, the USD/Euro exchange rate can be hardcoded (not fetched from an API).
c. The user must be able to view **the total value** of the portfolio and its stocks updated with the latest values in the currently selected currency.

4. Compare stock value performances in a portfolio **10pt**
    a. User must be able to see a graph showing the historic valuation of the stocks
    b. User must be able to adjust the time window of the graph by selecting the starting and ending date of the graph.
5. Remove stock **5pt –** A user can remove individual stocks from a portfolio
6. Remove a portfolio **5pt**
    a. The user can delete a portfolio
    b. When deleting a portfolio all associated stocks should be removed
7. The portfolios should be persistent over browser sessions **5pt**
    a. You need to use the persistent local storage to save all data (portfolios, stocks, number of shares, initial values, latest values, and latest historical values) related to the created portfolios.
    That is, after closing and opening the browser the portfolio information should still be available regardless of having internet connection

## Non-functional requirements:

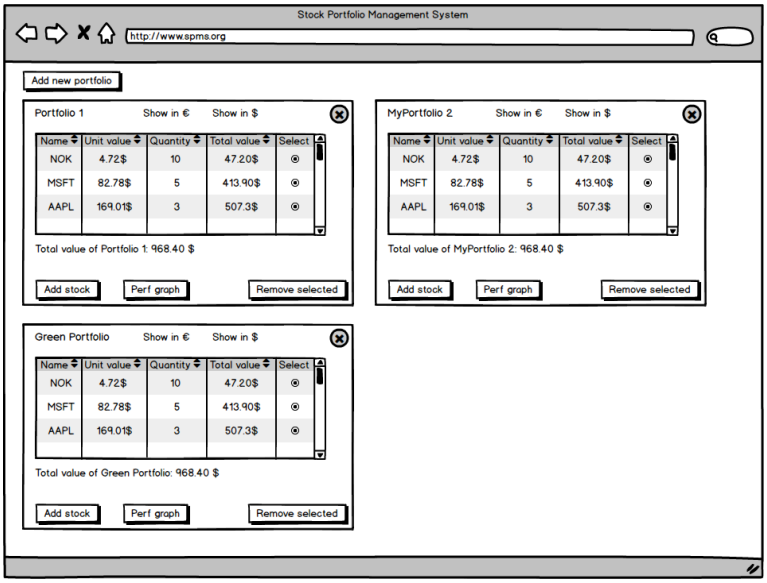
1. The layout should be responsive **5pt**
    - You are free to decide on the layout and look of your application. The usability of your application will be considered when evaluating your solution.

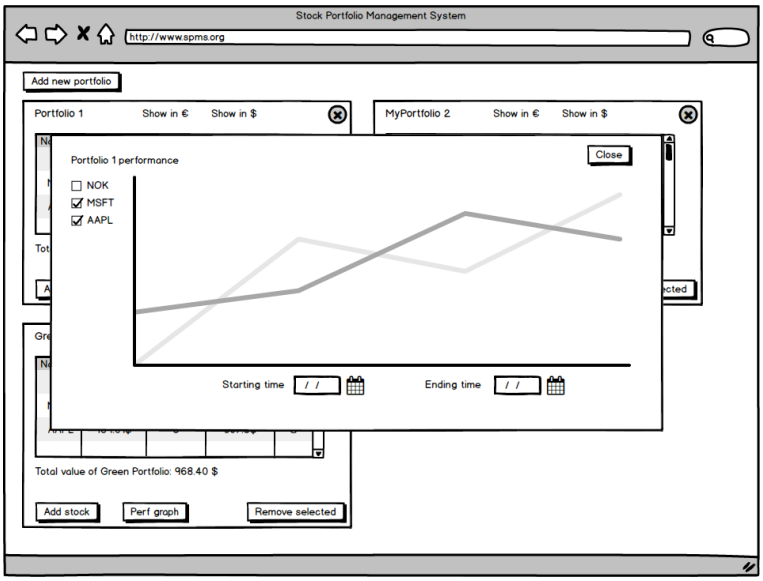
## Constraints:

- The project should be implemented using ReactJS.

## Mock-up of Visual Layout

The following are only mock-up examples, you do **not** have to strictly follow them!



## Hints

- Currency exchange rates, stock prices and historical prices are available via the following API: https://iexcloud.io/docs/api/.
- The above has a Free plan which requires registration. They also have a sandbox that in theory allows unlimited number of requests https://sandbox.iexapis.com/stable/stock/twtr/chart/5d?token=Tsk_21d37f6babe141e3bdd74a95de7106e2.
- An alternative is https://www.alphavantage.co/documentation, but there are some stricter limitations (5 API requests per minute; 500 API requests per day)
- For creating graphs, we recommend http://recharts.org/ but you are free to use other libraries if you like.

- **Comment your source code such that anyone reading your code can easily understand its logic.**
- Remember that the link to your GitHub repository might be requested during a job interview, and your source code been looked at by your future employer.
- DO NOT FORGET TO ENABLE GITHUB PAGES. A nice package for publishing React code on GH pages can be found here. https://github.com/gitname/react-gh-pages. This is needed only if you use node.js to build and deploy your project.

Some hints for the IEXcloud API (please use your own API-key in your application):

- Complete API spec is available at https://cloud.iexapis.com/stable/
- Latest quote info for a stock is available at https://sandbox.iexapis.com/stable/stock/aapl/quote/?token=Tpk_391653b184fb45f2a8e9b1270c0306e9
- Latest price for a stock can be obtained https://sandbox.iexapis.com/stable/stock/aapl/quote/latestPrice?token=Tpk_391653b184fb45f2a8e9b1270c0306e9
- Historic quote for given symbol is https://sandbox.iexapis.com/stable/stock/nok/chart/5d?token=Tsk_21d37f6babe141e3bdd74a95de7106e2
- Stock value for a given day and symbol
    - /stable/stock/{symbol}/chart/date/YYYYMMDD?chartByDay=true
    - https://sandbox.iexapis.com/stable/stock/nok/chart/date/20191003?chartByDay=true&token=Tsk_21d37f6babe141e3bdd74a95de7106e2
- The same information can be obtained in some cases in different ways. The above links is one possible solution.
- Also note that historical data is provided reliably up to **5 years back** for the iexcloud API.