OHSIHA 2020, Harjoitustyön vaihe 1

Arttu Saarinen
arttu.saarinen@tuni.fi

Harjoitustyön tarkoituksena on toteuttaa verkkopalvelu, joka hakee muutamasta eri rajapinnasta erilaista finanssimarkkinadataa (mm. osakekursseja, pörssi-indeksejä, valuuttakursseja) ja koostaa niistä eräänlainen dashboard omiin informaatiotarpeisiini. Teknologioiksi valitsin alla listatut työkalut, koska olen Web-ohjelmoinnin kurssin käynyt ja halusin oppia uutta sekä uuden kielen (Javascript) että muuten uusien työkalujen avulla.

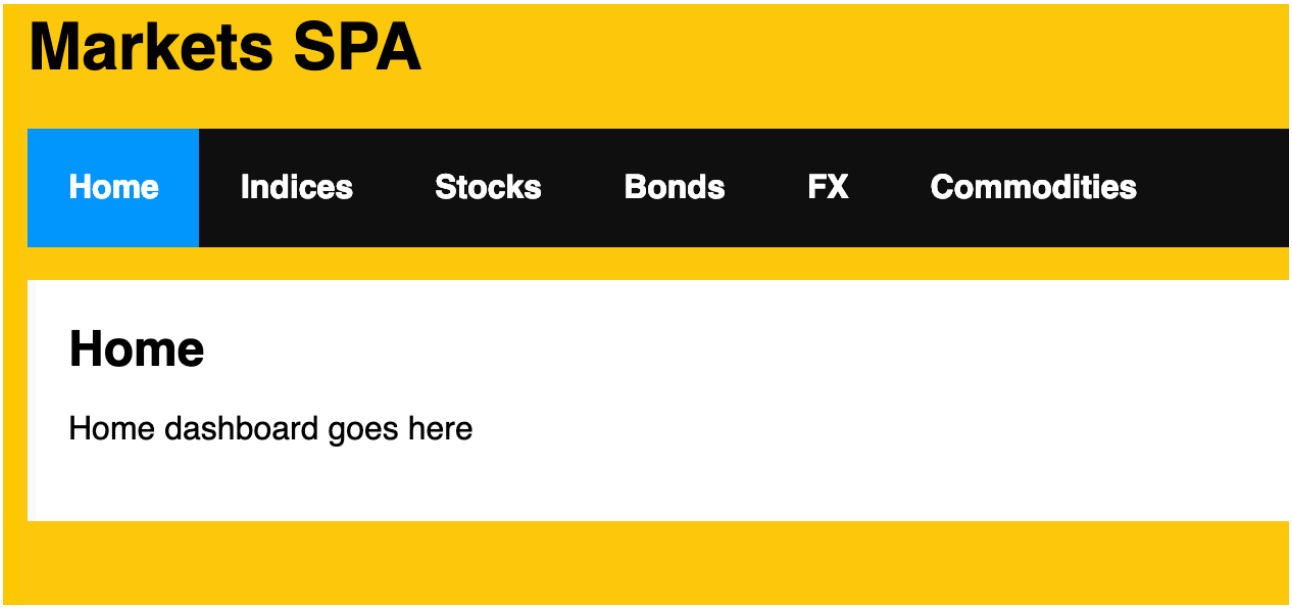
Käytetyt teknologiat

- Frontend: React
- Backend: Node.js
- Tietokanta: Atlas MongoDB
- Versionhallinta: Github
- Editori: Visual Studio Code
- Serveri/sovellusplatform: Heroku
- Hyödynnettyjä kirjastoja: Passport, Mongoose, Nodemon

Asennus ja kehitys tähän asti

Uusia ohjelmia ei harjoitustyötä varten varsinaisesti tarvinnut asentaa, sillä esimerkiksi Visual Studio ja NodeJS löytyivät koneeltani jo valmiiksi aikaisemmin tehtyjen pienempien projektien ansiosta. Muutamien pakettien asennuksen jälkeen kehitysympäristö oli valmis toimimaan.

Create React App-paketilla loin rungon palvelulle ja tein nopeasti jonkinlaisen ulkoasun sekä url-reititykset tässä vaiheessa ajattelemilleni ominaisuuksille. Tietokannan toimintaa en ole vielä sivuston toimintaan varsinaisesti kytkenyt, mutta se on kuitenkin projektiin asennettuna. Tällä hetkellä sivuston etusivu näyttää tältä:



Alla esimerkkejä koodista Main.js-tiedostosta, johon on luotu etusivun komponentit:

```
import React, { Component } from 'react';
import {
    Route,
    NavLink,
    HashRouter
} from 'react-router-dom';
import Home from './Home'
import Bonds from './Bonds'
import Commodities from './Commodities'
import Fx from './Fx'
import Stocks from './Stocks'
import Indices from './Indices'

class Main extends Component {
    render() {
        return(
            <HashRouter>
                <div>
                    <h1>
                        Markets SPA
                    </h1>
                    <ul className='header'>
                    <li><NavLink exact to='/'>Home</NavLink></li>
                    <li><NavLink to='/indices'>Indices</NavLink></li>
                    <li><NavLink to='/stocks'>Stocks</NavLink></li>
                    <li><NavLink to='/bonds'>Bonds</NavLink></li>
                    <li><NavLink to='/fx'>FX</NavLink></li>
                    <li><NavLink to='/commodities'>Commodities</NavLink></li>
                    </ul>
                    <div className='content'>
                        <Route exact path='/' component={Home}/>
                        <Route path='/indices' component={Indices}/>
                        <Route path='/stocks' component={Stocks}/>
                        <Route path='/bonds' component={Bonds}/>
                        <Route path='/fx' component={Fx}/>
                        <Route path='/commodities' component={Commodities}/>

                    </div>
                </div>
            </HashRouter>
        );
    }
}

export default Main;
```

Tietokannan asennus ja konfigurointi

Tietokannaksi valitsin MongoDB Atlaksen, jonka konfigurointi oli suhteellisen suoraviivaista. Loin MongoDB:seen tilin, jonka jälkeen päästiin tietokannan luomiseen. MongoDB:ssä on ensin luotava uusi

klusteri, johon sovelluksen vaatima(t) tietokanta(kannat) laitetaan pyörimään. Tähän klusteriin piti luoda koneelle yhteys, jossa mm. whitelistattiin IP-osoite josta liikenne tietokantaan tulee. Vastaavasti luotiin myös tietokannalle/klusteriin käyttäjä, jonka tiedot tuli laittaa ohjelmakoodiin (.env-tiedostoon sisään). Lisäksi .env-tiedostoon laitettiin MongoDB:stä/Atlaksesta saatava connection string. Ohjelmakoodin sisällä tietokantayhteydet hoidetaan puolestaan mongoose-kirjastolla, jonka käyttöä en tässä vaiheessa vielä suuremmin testejä lukuunottamatta alkanut tekemään. Ensimmäisenä ominaisuutena ajattelin toteuttaa kirjautumisen ja rekisteröitymisen yms. autentikaation, jossa tietokantayhteyksiä pääsee varsinaisesti harjoittelemaan.

Herokun käyttöönotto

Herokun käyttöönotto oli todella suoraviivaista, eikä sen toiminta juurikaan eroa gitin käytöstä versionhallinnasta. Laitoin herokuun pyörimään vasta oikeastaan create-react-appin luoman sovellusrungon. Ensin asennettiin Heroku Command Line Interface (CLI), jonka jälkeen luotiin herokussa App, otettiin herokun projektiin/appiin liittyvä Git URL, johon pushattiin projekti. Tämän jälkeen Heroku pitää huolen oikeastaan kaikesta sovelluksen pyörittämiseen liittyvästä.

Haasteet ja helpot asiat

1. Uusien työkalujen asentaminen ja opiskelu on tässä vaiheessa jo suhteellisen helppoa, kun erilaisten pakettienhallintaohjelmistojen käyttö on tuttua.
2. MongoDB:n ja Herokun käyttöönotto oli todella suoraviivaista, ja niiden omat dokumentaatiot riittivät konffaukseen
3. Haastavaa oli työn aloittaminen ja aikatauluttaminen, muiden kurssien ja kiireiden takia aloitin projektin jo joululomalla, mutta se on ollut viimeisen kolme viikkoa täysin koskematon ja aiheeseen palaaminen raportin kirjoittamista varten oli aikaavievää.
4. Harjoitustyön ohjeen seuraaminen oli hiukan haastavaa myös: usealle sivulla jakautuneet ohjeet eivät välttämättä ole paras metodi ohjeiden antamiseen.

Hyödyllisiä lähteitä

Reactin asennus (Reactin dokumentaatio): https://reactjs.org/docs/create-a-new-react-app.html

Routtaus: https://blog.pusher.com/getting-started-with-react-router-v4/

https://www.kirupa.com/react/creating_single_page_app_react_using_react_router.htm

Herokun käyttöönotto: https://devcenter.heroku.com/

Tietokannan asennus (MongoDB:n dokumentaatio): https://docs.mongodb.com/guides/