Lappeenrannan teknillinen yliopisto

School of Business and Management

Sofware Development Skills

Petri Kotiranta, 0589731

LEARNING DIARY, Software Development Skills: Full-Stack MODULE

**Oppimispäiväkirja**

20.4.2020  
Aloitin kurssin kirjautumalla kurssiavaimella Moodlen kurssisivulle. Halusin aloittaa FullStack-kurssin, koska oli epävarmuutta pääsenkö työvoimatoimiston järjestämälle kurssille. Itselläni on aikaisemmista työsuhteista varsin paljon tietokantaosaamista, mutta frontend-osaamiseni kaipaisi päivitystä. Muutenkaan Angularista, Vuesta tai Reactista ei itselläni ole hirveästi aiempaa kokemusta, joten näiden teknologioiden osalta tietoni kaipasivat päivittämistä. Tällä kurssilla opitaan Angularia.

Aloitin tutustumalla kurssin Moodle-sivuun ja lukemalla läpi kurssin vaatimukset. Sitten asensin kurssilla vaadittavat ohjelmat. Näistä Atom oli itselleni uusi tuttavuus ja sen käytössä oli hieman opettelua. Se kuitenkin vaikutti varsin näppärältä koodieditorilta. Lisäksi se on vielä avoimen lähdekoodin ilmaissovellus. joka toimii Macissa.

21.4.2019

Aloitin käymään läpi kurssin sisältöä. Kurssi näyttää koostuvan itse opiskeltavasta materiaalista ja palautettavasta MEAN-harjoitustyöstä. Ensimmäisenä oli video, jossa tutustuttiin NodeJS:ään. Videossa asennettiin NodeJS ja tutustuttiin hieman perusasioihin. Lopuksi tehtiin yksinkeratinen web-palvelin NodeJS:llä.

NodeJS oli itselleni uutta asiaa. JavaScriptissä on menty varsin pitkälle, sillä nyt kokonaisen web-palvelinsovelluksen voi tehdä JavaScript-sovelluksen avulla. Itse olen tottunut tekemään palvelinohjelmistoja PHP:llä tai Javalla, mutta on mielenkiintoista, että samoja asioita voi tehdä JavaScriptillä.

22.4.2019

Seuraavassa videossa käytiin läpi MongoDB:n toimintaa. MongoDB on itselleni tuttu jo ennestään, joten tässä videossa ei tullut itselleni mitään uutta asiaa. Olen aikaisemmissa työtehtävissäni asentanut MongoDB-klusterin, joten MongoDB:n toimintaan on tullut tutustuttua jo varsin yksityiskohtaisesti. Videon sisältö oli itselleni lähinnä kertausta.

23.4.

Seuraavana vuorossa oli video ExpressJS:stä. Tässä videossa oli jo monimutkaisempaa asiaa ensimmäiseen NodeJS:ään nähden. Tein omaa versiota ohjelmasta videon mukana. Router oli itselleni erityisesti uutta asiaa. Tällaista en ollut aikaisemmin tehnyt. Router mahdollistaa, että web-ohjelmassa voidaan tehdä kyselyitä erilaisiin URL-polkuihin. Tämän avulla ohjelma voidaan jakaa frontend- ja backend-osiin. Templatejen käytöstä minulla oli jo kokemusta. Olen aikaisemmin käyttänyt Smarty template-moottoria ohjelmoidessani PHP:llä. Tässä videossa oli jo paljon enemmän opittavaa aikaisempaan nähden.

24.4.

Seuraavana vuorossa oli Angular-sovelluksen teko. Aikaisemmasta poiketen materiaalina oli tutoriaali web-sivulla, eikä videota. Löysin myös Youtubesta Angularin perehdytysvideon samalta tekijältä, joka oli tehnyt aikaisemmat videot ja katsoin senkin läpi. Koodasin Tour of Heroes -sovelluksen ohjeiden mukaan. Oli oikeastaan parempi työskennellä web-sivuna olevien ohjeiden mukaan, kuin seurata videota, sillä nyt ei tarvinnut huolehtia videon pysäyttämisestä ohjelmoinnin ajaksi. Lisäksi jokaisella sivulla oli lopuksi esimerkkikoodit. Nämä olivat hyödyllisiä, jos huomasi omassa koodissaan bugeja. Esimerkkikoodien kautta oli helpompi selvittää. mikä on vikana. Uutena asiana tässä harjoituksena tuli servicet. Nämä ovat erillisiä moduuleita, jonka avulla frontend-toiminnot käyttävät backend-kutsuja.

25.4-28.4.

Viimeisenä oli vuorossa kokonaisen MEAN-stack -sovelluksen tekeminen. MEAN-stack sovelluksen tekeminen käytiin läpi 10-osaisella videolla. Aloitin katsomalla videot kertalleen läpi ja toisella kerralla koodasin mukana. Viimeinen harjoitus sisälsi jo varsin paljon asiaa, joten kertaus oli hyödyllistä, jotta omaksui asian hyvin. Esimerkkiprojekti oli hyödyllinen, sillä oman koodin bugit sai tehokkaasti korjattua sen avulla. Lopputuloksena syntyi sovellus, jossa käytettiin MongoDB-tietokantaa tiedon tallentamiseen ja siinä oli erillinen backend- ja frontend-osa. Sovelluksessa käytettiin sekä routeria, että servicejä, jotka esiteltiin aikaisemmissa tutoriaaleissa.

Toteutin esimerkkisovelluksen pohjalta oman sovelluksen, jossa käytin esimerkkisovelluksen koodeja. Tein sovelluksesta elokuvien listaussovelluksen, jossa jokainen kirjautunut käyttäjä voi lisätä omia elokuviaan ja asettaa niille arvosanan. Elokuvat näytetään etusivulla, kun käyttäjä on kirjautunut. Elokuvia voi myös poistaa painamalla Delete-nappia.

Kun lähti tekemään omaa sovellusta piti paremmin ymmärtää angular-sovelluksen toimintaa. Elokuvan lisääminen oli varsin helppo toteuttaa esimerkissä toteutetun rekisteröinnin pohjalta. Elokuvien hakeminen sen sijaan tuotti hankaluuksia, sillä käytin siihen get-komentoa. Aikani säädettyäni sain ratkaisun tehtyä Tour of Heroes -sovelluksen pohjalta. Pienen googlettelun avulla sain selville, että get-pyynnön kautta tulevat parametrit otetaan req.query:llä req.body sijaan. Lopulta sain elokuvien hakemisen gettinä toimimaan ja elokuvat listautuivat oikein dashborard-sivulle. Lisäsin myös elokuvan poiston ja mahdollisuuden antaa elokuvalle arvosana. Arvosana esitetään elokuvan vieressä elokuvalistauksessa.