Lappeenrannan teknillinen yliopisto

School of Business and Management

Sofware Development Skills

Timo Hyvönen, 0600254

LEARNING DIARY, “CT70A9130 - Back-End”- MODULE

**LEARNING DIARY**

I read through the text portion of the course information and quickly checked out the videos included in the course. I started writing this learning diary and started watching the first video titled "Rest Intro"

**REST Intro:**

I learned that the API stands for Application Programming Interface, the basic concept of API. REST is an acronym for (Representational State Transfer) and that the Rest API works on top of the HTTP-protocol.

Facebook Graph API is located in the address: graph.facebook.com. In the Facebook Graph API, the API address and parameters section are separated with a question mark "?". Parameter name and values are separated with "=" and multiple parameters are separated with a "&"-symbol, so for example in the video "address=chigago&sensor=false".

Googlemaps API is located in: maps.googleapis.com

Instagrams API is located in: instagram.com/developers which is provided by the Apigee-service. I checked out the list of APIs in the page Programmableweb.com.

For HTTP API requests, "Get"-type of request is used to consume data and "Post" is used to send data to the API. I made sure that I still have the Postman-program installed on my computer. I also learned the basic concept of the OAuth-authentication.

I installed the GIT for windows and started a new project for the coursework code.

**NodeJS-video:**

NodeJS was already familiar to me so the video did not have that much new information to me. I did learn that Chromes V8 engine is written in C++.

I made sure that I have the Node.js installed and I installed the Visual Studio code.

I typed in the examples in the video and tested them just to make sure that I did not miss any basics of NodeJS. I created a new local repository, committed all of the example code and pushed all of the commits from my local repository to a github repository <https://github.com/Uppanen/CT70A9130-Back-End-course>

**MongoDB:**

MongoDB was completely new to me, though previously I've been using Microsoft SQL, SQLite, MySQL and InfluxDB. I learned that MongoDB Is a NoSQL type of SQL database, the data is stored as documents and stored in Json like syntax. I downloaded and installed the MongoDB server software. I started to follow the tutorial to create the needed subfolders to the MongoDB installation folder. The MongoDB installation has changed from the installation file used in the video and there was no need to do the configuration of the mongodb-service, the msi-installation does it automatically.

I learned how to create a database, user, collection and how to add a document to the collection. Also how to show the documents in a collection, add multiple documents with one command, update existing documents, increment an existing value, remove a field, rename a field, remove a document, find documents using logical parameters, sort the find result, count the find result, limit the number documents retrieved. Lot’s of new information

**ExpressJS:**

I have not used ExpressJS before, but I’ve worked with other MVC-frameworks (ASP.Net, .Net core). ExpressJS seems to very simple and see through way of creating webpages. In ASP.Net there is a lot of things going on in the background which you need to keep in mind and creating just a simple webpage contains a lot of different components. I followed the tutorial video and typed the example code while watching. I learned how to create views and partial views, how to pass in parameters to the views and how to use the ExpresJS-notation.

**Final project:**

I started to plan out my own project by reviewing the requirements (REST with Node, Express and MongoDB). I decided to create a simple warehouse inventory system which could track the items in the storage. The items would have the following properties: Name, Description, ItemCode, Quantity, WeighPerItem, Location. There would be a separate page to add items with a "add item"-form. For existing items there would be an edit-page with a form to edit the values. The frontpage would show a list of items in the stock and there would be a link to delete the item on every item row.

I started the project by copying the frame of the demo-project from the ExpressJS-lesson. I removed all the extra code and left only the view engine part, basic view code and the root get handler. I added the "Add"-view and made the basic form to fill for a new item. I added the add-request post handler and tested adding an item using the add-form. The item was successfully added when I checked the warehouse-collection from the mongodb command line. Next, I added the list of items in the frontpage to show the items in the database. Adding the edit-page was just a simple copy and paste from the Add-page but with a put-method to update the item instead of using the post-method. For the delete-link I used the delete-method. Finally, I cleared out the mongodb from the existing warehouse-database and tested that the warehouse inventory page works correctly.

I recorded a video which shows the page in action and uploaded the video to YouTube, uploaded this learning diary to the GitHub repository and returned my work to be graded.