Lappeenrannan teknillinen yliopisto

School of Business and Management

Sofware Development Skills

**Juha Mursula, 0621738**

# LEARNING DIARY, BACK-END MODULE

# LEARNING DIARY

### 3.11.2020: REST Intro

From this short video I learned that REST is an abbreviation from the words "Representational State Transfer". It is an architectural model to create application programming interfaces (APIs) which allow different software to change data between themselves. REST it is based on HTTP protocol.

In the video some practical examples about using REST was shown. In the examples the JSON (JavaScript Object Notation) format for sending data was used but probably also some other formats can be used. I also learned that Postman is a great tool when you need to test your restful APIs. It can be used for GET and POST requests as well as other requests too.

### 4.-5.11.2020: Node.JS

In the beginning of the video the "lecturer" went through some basic issues about Node.JS. I learned that with Node.JS you can run JavaScript code on the server. The advantage is that you can use JavaScript language in both front end and back end programming. So, you do not necessarily need to study Java, C#, Python or other languages before you can regard yourself as a full-stack developer.

I also learned that Node.JS uses only one thread, so it is asynchronous, which means that the thread does not stop and wait an answer for the request. Still, Node.JS is scalable, it supports tens of thousands concurrent connections. Node.JS can be used in all kinds of projects except in CPU intensive projects because heavy CPU calculations can block the thread for a long period.

Next, the "lecturer" talked about NPM (Node Package Manager) and Node modules. I have used NPM in front-end course, so I knew something about it beforehand. Now I got a little bit deeper introduction to NPM and learned also how to create your own Node.JS modules and how use them elsewhere in your project. I installed a module called "Nodemon". It is a handy tool in coding web pages with Node.JS because you do not have to restart the server after you have made updates to your code after Nodemon installation.

Next, some core Node.JS modules were presented in the video. Now I know how to get file and directory names and how to create file paths with “path” module. I learned how to create directories and files and how to read, update and rename files with “FS” (File System) module. I also familiarized myself with “OS” (Operating System) module which provides methods related to the operating system. In addition, I learned the way to build URLs and how to catch different parts of URL and how to add dynamically search parameters to URL. All this can be done with “URL” module. Next, the “lecturer” presented “events” and “http” modules. Now I understand the meaning of an event listener and how events can be fired (emitted). It was also funny to observe that it needs only some rows of code to build a simple web server.

At this point, we start handling the main issue of the video: building up a real web server. Of course, some Node.JS modules mentioned above were needed when creating the server. The code was quite easy to understand. I learned how to build the server in such a way that you do not have to hardcode the call of every single HTTP page in the code, but you can call them dynamically. I also learned how to handle errors. For example, if the user gives an address that does not exist (error code 404), a special error page will be shown. Now I also understand the meaning of content type. I acquired additional coding skills in JavaScript too, even if this is not a JavaScript course. In general, my understanding of the function of the web application in its entirety grew a lot when I watched the video and coded the example project.

Finally, I created a Heroku account and installed Heroku Command Line Interface (CLI). After that I deployed my example application to Heroku. Heroku produced an internet address for my application and deployed it on the internet. Now anyone who knows the address of my little application (https://desolate-gorge-89154.herokuapp.com/) can see it in the web. The deployment was easy and straightforward. It is good to know that there exist places where you can install your test web applications for free.

## 6.11.2020: MongoDB

After I had watched the video first time, I installed MongoDB, also including MongoDB Compass. Everything went well but the installation of MongoDB Compass took a long time, maybe 10-15 minutes. I thought that the installation was stuck but then I closed all other programs and finally the installation ended.

As in the video, I first opened MongoDB Compass and did some database operations. I created a new database and a new collection there and after that I inserted some data to the collection and updated and deleted data. MongoDB Compass tool looks quite handy if you need to manipulate the database, because you do not have to remember the "query language" commands. However, if you are a database manager you also have to know how to manipulate database with command line (Mongo shell), because you often need to generate very complex queries. But in easy database tasks, MongoDB Compass may be useful.

Next, I did the same database operations with Mongo shell presented in the video. I learned how to create a database and a collection and how to insert, update and delete data. I also learned e.g. how you can sort collections and find records based on a search criterion. I have a very long experience of relational databases and SQL. The query language used in MongoDB differs totally from SQL. It remembers more a programming language than SQL and it takes time until you can use it fluently. At least for me, it looks quite complicated. Also, MongoDB itself is quite different than a relational database. MongoDB is much more flexible; you do not necessarily have to define a tight "schema" which expresses the structure of the table in relational databases. In MongoDB instead, you can create new fields on the fly. MongoDB uses a BSON data format for both as a data storage and a data transfer format which in many cases makes transferring data between databases simpler than between relational databases. However, I am not totally convinced in using MongoDB with larger databases. I think that because of the flexibility, there is a bigger risk to cause a complete mess compared with relational databases. At least you must be very careful and know exactly what you are doing.

The last issue presented in the video was MongoDB Atlas which is a cloud database for MongoDB. At this point I did not create an account. I just watched the video and learned e.g. how to create a cluster and how you can connect to Atlas from Mongo shell.

**11.-12.11.2020: Express.JS**

At this point I took a two-hour crash course concerning JavaScript so that I would better understand the logic of some JavaScript properties, e.g. arrow functions, for each loops, filters, operators and so on. I also studied the basics of the HTTP protocol: request methods, responses, and status codes.

Next, I started to watch the video. I was taught that Express.JS is a server-side framework that makes it easier to build web applications with Node.JS. I also learned the basic structure of an Express.JS application and how the request are handled in the application. I was also taught that with middleware functions you can make changes to request and response objects.

Second, we proceeded to the practical section. First, the “lecturer” taught how easy it is to create a “static” web server with Express.JS compared with bare Node.JS. In Express.JS you only have to create a static folder and put all your HTML files there and that’s it. Express.JS takes care of the rest. With Node.JS you have to specify content types and file paths manually.

Next, the “lecturer” created a simple middleware function (logger) only as an example to show how middleware works and how it is implemented. I found this little example very demonstrative.

Because we do not have a database in this exercise, we created a separate JavaScript file (called Members) that includes three JSON formatted objects in an array. Then we programmed a router that receives different HTTP requests concerning these objects. I learned how a router is taken in use, how it should be implemented: how to return all objects or just one object based on the object id, how to add objects to the array, how to update objects and how to delete them. The code was easy to understand because it was quite straightforward and some of the stuff was taught earlier in Node.JS section. Of course, if we had had a database in use it would have been better, because we would have done the operations all the way (select data from the database and also insert and update data).

The last thing presented in the video was Express Handlebars which is a template (view) machine with which you can easily create server-side web applications. We also took Bootstrap, a CSS framework in use. I learned that Express Handlebars requires a certain directory structure and a HTML page wrapper (main layout) to work. The syntax was a little bit strange with all those double and triple curly brackets, but I think that I understood the main idea. The meaning of Bootstrap classes was not explained, so I learned them from W3Schools web site.

**19.11.2020: REST**

Many things explained in this video was presented earlier in Brad Traversy's videos. For example, such things like creating a server and building routers were discussed in Traversy's videos. However, these repetitions were not unnecessary at all. You always learn a little bit new also. The most important new thing in this REST video (the "teacher" in the video did not introduce himself, so I do not know his name) was how to connect to MongoDB database using Mongoose and how to do database operations.

I learned how to create a MongoDB document schema using Mongoose. The schemas in MongoDB remind a lot the schemas used in relational databases, although the syntax how to define the schema is quite different. You can e.g. specify document field's datatype, default value and allowed values. You can also define if the field is obligatory or not as well as many other attributes.

The code in the video which was used to do database operations was quite easy to understand. I learned how to find objects (documents) from MongoDB database, how to insert them using schema and how to update or delete them. I also learned that those operations were asynchronous, which means that they are independent of the main program flow.