Lappeenrannan teknillinen yliopisto

School of Business and Management

Sofware Development Skills

Jhuma Kabir Mim, 000327482

**LEARNING DIARY**

**Learning diary 1**

10. April.2022

I checked the general information and understood the main focus of the course, which is This lecture introduction about NodeJs and its application, which is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine, developed in c++, and executes JavaScript code outside a web browser. NodeJs lets developers use JavaScript to write command-line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Also, this lecture talks about the core fundamentals of JavaScript. I learned that javascript is an asynchronous language, how nodeJs can be set up, and how we can install different packages and frameworks in node js with the 'npm install' command. Also, I learned how to create a node server, a small micro API service, and get output in a JASON format.

To practice package installation I have test npm install command to install vue.js. To do that I have used npm install -g @vue/cli which has globally install vueJs framework in my pc.

**Learning diary 2 - MongoDB**

05. May.2022

This course is about MongoDB, which is a database, but it's a NoSQL database. It is scalable and very popular with the JavaScript community since it has similar JavaScript syntax to work with. Working with Mongo DB is simple; we just need to create a database name and collection and connect to the server. For practice, I'm using the localhost default port. Creating a collection is like creating a Table similar to an SQL database. MongoDB can be used as a cloud service. For my final project, I will use the MongoDB cloud service provided by AWS. Both cloud versions and the local version of MongoDB support the same command for creating, updating, or deleting databases. It's better to know some basic syntax for database management using the command line rather than visual mode. I'm trying to learn command line syntaxes, and I found it interesting since it's similar to JavaScript syntax.

For using MongoDB, first, I have created a free account and selected AWS as a cloud server. I have selected Eastern-Europe as could server location for fast access to the database. I had to create a DB name, user name, and password. Then I have create function to connect with mongoDB.

async function loadPostsCollection() {

const client = await mongodb.MongoClient.connect(

'mongodb+srv://jhuma:Common@12!@cluster0.wsaic.mongodb.net/?retryWrites=true&w=majority',

{

useNewUrlParser: true

}

);

return client.db('Cluster0').collection('posts');

}

**Learning diary 3- Express Js**

12. May.2022

This lecture is about express js, which is a back-end web application framework for Node.js, released as free and open-source software under the MIT License. Before working with any JavaScript framework, it's better to know, for example, JavaScript function, array, object, data type, and different kinds of operation, statements, and conditions. I learned basic JavaScript before learning ExpressJs, which helped me further learn about advanced arrow functions, map, JSON format, etc. First, I have learned basic route handling, HTTP request handling, query string, and URL parts. So to use it first, I need a node server that I have already installed, then I have to use POSTman for API testing. In my project, I will use express as routing the different pages, redirecting (for example, redirecting a successful user to new pages), and showing HTTP responses (for example, 404 page not found, 500 server error). Though I have learned a little advanced, for example, API creation with express js where I have used four fundamental commands GET, UPDATE, POST, and DELETE. He gets a request to get information that can show output from a file ( i.e., Html file), folder, or JSON format data.

In my project I have used ExpressJS for server side code. I have wrote a ‘JS’ file and declare following properties like which port server will run (I have used port 500 since I will run my client at port 800), declaring Express, API support, resolving issue of ‘CROS’ etc. Without CROS enable Brower creates problem if client and server not in same host.

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const app = express();

// Middleware

app.use(bodyParser.json());

app.use(cors());

const posts = require('./routes/api/posts');

app.use('/api/posts', posts);

// Handle production

if (process.env.NODE\_ENV === 'production') {

// Static folder

app.use(express.static(\_\_dirname + '/public/'));

// Handle SPA

app.get(/.\*/, (req, res) => res.sendFile(\_\_dirname + '/public/index.html'));

}

const port = process.env.PORT || 5000;

app.listen(port, () => console.log(`Server started on port ${port}`));

**Learning diary 4- Angular js**

12. May.2022

Angular is a TypeScript-based free and open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. There are similar other frameworks, namely React and Vue js, that can be used almost the same way as angular. In my project, I have used Vuejs instead of angular. However, I have learned the basics of angular use and some of its features. Besides, it's backed by Google and has a huge community; there are several features and functionality that made angular very popular.

Single Page Applications: To reduce project load and run time, single-page application is getting popular nowadays. The need for Single Page Applications is increased as developers have to make web pages with less code. Even with single-page applications, we can develop mobile applications.

Front-End Applications: Another reason for the popularity of Angularjs is Front-end development. For enhancing the business, we should have attractive web and mobile properties. So, for every business, the Front-end of the web and mobile applications plays an important role. As JavaScript is used by almost every browser, frameworks of JavaScript like Angularjs are used to develop Front-End Applications. Websites like Google, Paypal, Netflix, and Virgin America are Created on Angularjs.

Model View Controller - Angular follows MVC, which is the Model that is used for handling the application data. The view presents the data in a particular format. In my application, the file format '.vue' is an example of a view similar in angular. Otherhand, controller handles the user inputs and carries out actions on the data model objects.

Dependency Injection - Angularjs has an inbuilt Dependency Injection feature. This feature assists developers in making the application easy to develop and test. Dependency Injections enable us to request dependencies. For example, I have used a CSS framework name bootstrap in my project, and I have used that dependency.

**Project- MEVN**

1. June.2022

What MEAN and MEVN?

MEAN stand for MongoDB, ExpressJS, Angular and NodeJs

MEVN stand for MongoDB, ExpressJS, VueJs and NodeJs

In my project in using Vue instead of Angularthats why it is MEVN.

For project I have create small blog page where I can create a post and delete it. I have used Mongo DB[[1]](#footnote-1) for database, Expressjs for API creation , Vuejs for frontend management and Nodejs for server handling.

Frist I create an API that can support GET, POST and Delete request.

// Get Posts

router.get('/', async (req, res) => {

const posts = await loadPostsCollection();

res.send(await posts.find({}).toArray());

});

// Add Post

router.post('/', async (req, res) => {

const posts = await loadPostsCollection();

await posts.insertOne({

text: req.body.text,

createdAt: new Date()

});

res.status(201).send();

});

// Delete Post

router.delete('/:id', async (req, res) => {

const posts = await loadPostsCollection();

await posts.deleteOne({ \_id: new mongodb.ObjectID(req.params.id) });

res.status(200).send({});

});

async function loadPostsCollection() {

const client = await mongodb.MongoClient.connect(

'mongodb+srv://jhuma:Common@12!@cluster0.wsaic.mongodb.net/?retryWrites=true&w=majority',

{

useNewUrlParser: true

}

);

return client.db('Cluster0').collection('posts');

}

After that I have create a client that shows the content and perform GET, POST and DELETE operations of post. I have also used a third party weather API to show weather data to learn and practice of using third party API.

Client side code for creating new post, showing the post and deleting the post:

<template>

<div class="container">

<h1>Latest Post</h1>

<div class="create-post form-group">

<label for="create-post">Write anything you want!</label>

<br>

<label for="create-post">Double click to remove the post!</label>

<br>

<input

class="form-control"

type="text"

id="create-post"

v-model="text"

placeholder="Don't be shy"

>

<br>

<button class="btn btn-primary" v-on:click="createPost">Post my name</button>

</div>

<hr>

<p class="error" v-if="error">{{error}}</p>

<div class="posts-container">

<div

class="post"

v-for="(post, index) in posts"

v-bind:item="posts"

v-bind:index="index"

v-bind:key="post.\_id"

v-on:dblclick="deletePost(post.\_id)"

>

{{ `${post.createdAt.getDate()}/${post.createdAt.getMonth()}/${post.createdAt.getFullYear()}`}}

<p class="text">{{ post.text }}</p>

</div>

</div>

</div>

</template>

To test the project we have to run both server and client. In client part it has separate Readme file for running client. Running server also has separate Readme files.

1. https://mlab.com/ [↑](#footnote-ref-1)