Lappeenrannan teknillinen yliopisto

School of Business and Management

Sofware Development Skills

<Oriyomi Oladele>, <Insert student number here>

LEARNING DIARY, <FRONT-END> MODULE

**LEARNING DIARY**

I checked the general information and understood the main focus of the course, which is to find my passion as a software developer and create a unique project to represent my skills. As I intend to take all the courses under the module, I have chosen to start with the Front-End. I went through the general information of the course, I also went through the page where environment setup is discussed. I started reading the front end page of the course and I understood that the goal is to understand how to combine HTML and CSS with JavaScript to create various effects in web page. Then I started watching the Part 1 video under "Introduction to workflow and sass". I set up my work environment including configuring the VS Code (I installed the required extensions such as live server, bracket pair colorizer and prettier code formatter) which I have been using for a long time though with other programming languages. I added the path to the VSCode code and bin folder to the environment variables since I have not needed to do that in the past. I also decided to continue using Git over bitbucket since I have been using Git a long time also. I already had node installed before now for other projects.

Created a dist folder: This is where all the html files, the compiled CSS, JavaScripts and all the images goes.

Create an index.html file and the live server is started by right-clicking on the index.html file and choose 'start with live server'.

Create a scss folder where all the scss files or sass files goes. Create main.scss file.

Because the browser does not read scss files, we need to install node-sass which is responsible for compiling sass files into CSS.

Before installing node-sass, we need to create a package.json file which is like a manifest for our project. From the VSCode terminal, enter the command 'npm init'. We will need to answer all the questions and the package.json file will be created.

Then we can install node sass with a command 'npm install node-sass'. Anytime we install a package with npm, it is included in the package.json file under 'dependencies'.

To set-up node-sass, we need to change 'test' text under scripts in package.json to 'sass' and the value should be changed to 'node-sass -w scss/ -o dist/css/ --recursive'.

The above command runs node-sass programs by watching the scss folder for changes in scss files and as a result outputs a regular CSS file which is created in the dist directory. The --recursive flag is needed to take care of partials (scss files with name starting with \_) during auto-reload.

Use the command 'npm run sass' to start node-sass.

We need to include the link to our compiled CSS file inside the html file. Whenever we make any change to our sass file, it will cause the auto-reload of the web page.

Before making this project directory a git repository, we need to create a .gitignore file under the project directory and add node\_modules to the file. This ensures that node-modules and its contents are not included in the project directory. To initialize a git repository, use the command 'git init' from the command terminal. This creates a hidden folder with a .git extension. then use 'git add .' to add all the files in the project directory to this git repository except those included in the .gitignore file. After this we can make a git commit with the command "git commit -m 'Title of the commit'"

Homepage and Core sass/CSS

Creation of the menu items (menu nav, menu branding, the button lines). An img folder is also created under the dist folder. The img folder is where all the image files goes. A partial \_config.scss file is also created under the scss folder which is where all the variables under @mixin are put. In order to get this to work with our application, we need to import the partial in our main.scss file with the command "@import 'config';"

Creating the button for the Menu Bar.

Create a new partial called \_menu.scss. Import it into the main.scss with the command "@import 'menu';"

The \_menu.scss file contains all the specifications for the menu button. Sometimes after creating a new sass or partial file, we might need to stop and restart npm-sass.

Menu button continuation, JavaScript part coding and rotating the menu button.

The main.js file is added to the project inside the dist directory. This is where the logic for toggling the menu button's state to show and hide the menu items including the menuBranding and navItems.

Adding overlay consisting of menu list sliding in and out on button click.

The \_menu.scss file is modified to take care of the menu overlay. it sets the properties of the menu including its position, width, opacity and setting the visibility of the menu.

Adding a portrait to the slide in overlay. Here the properties for the portrait image is set in the menu.scss file.

Making the nav menu items slide in from the right on button click.

Setting the transition delay to the nav item html children. This is used to make the items slide in one after the other not just all at once.

Addition of media queries and making of menu overlay responsive.

Addition of text color function and the about page. This is follow by the work page addition and the responsiveness of the page.

The example website was deployed to github after completion.

The link to the website is shown below

https://yomites.github.io/modern\_portfolio/about.html

The link to my project website is given below and also the youtube link for the project explanation.

https://yomites.github.io/my\_website\_project/index.html

https://youtu.be/MgoCjglVdtI