Lappeenranta University of Technology
LUT UNIVERSITY

Software Development Skills Front-End, Online course

Václav Tůma, 002528535

LEARNING DIARY, FRONT-END MODULE

LEARNING DIARY

03.11.2024

I have enrolled (not only) for this course, because I wanted to learn more about front end. Programming is very important and as an electronics student it is great to have at least basic knowledge about web programming.

After enrolling in the course, I found basic information about the course and tried to get familiar with the content of the course. From the beginning I was a little bit confused, but after reading all the information given, everything became clear.

The first necessary step was environment setup. But as I had course Introduction to web programming, I was already familiar with Git and Visual Studio Code. Moreover, I used Visual Studio Code at my home university. This means, that I did not have to install new plugins, because I had already installed **LiveServer plugin in the VisualStudio Code** recommended and used during Introduction to web programming course. Because of this, beginning was much easier and the only thing what I had to do was refresh my memory of how to create a public repository, create new folder for this course linked to Git and make first commit this time with empty html file called **index.html**.

Introduction and Base HTML

I decided to start the project by watching a video, where the first part is about the introduction of the project. As I never heard about Figma, I had to do a little research about that, create account, I was looking to features because this looks to me as really good starting point if I will want to create something in the future.

First what I learned here was how to get images as background image in hero layer. It is quite simple, just search for which image I wanted and export it. This image needs to be copied to VS code which I gave to separate file called **Images.** After this I prepared CSS called **style.css** and JavaScript with the name **JAVASC.js.**

Right after beginning the code writing, I learned a new thing, which was **!** shortcut for

<html > <head > and <body>. Originally, I wrote this once and then just modified code for next learning tasks, but this shortcut is useful. This is just a basic structure, where linked style and JAVASC.

As I followed tutorial, I also changed font to Poppins, because I don’t like the basic font either.

The first part, that user sees is navigation bar. For that I created nav with container.

04.11.2024

Links and Core CSS

In this module, right from the beginning I learned, that * in CSS represents everything, and it is possible to use it for reset. Basic body and other CSS I made many times, so changing font and its size was not something new for me. What I learned here was :root where we can add for example color as variable and not necessarily user hex number for color repeatedly. This I did not use during the introduction of web programming, but this functionality is useful.

Buttons & Utility Classes

As the name of this module says, the first part was a button design creation for log in. I have made something similar during the web course, so again, a little refresh of memory. During work I was committing to Git after every larger change. Right here, I found a problem with buttons in HERO part, because the design was not working for some reason. After some time and trying to find out where the error is I found out that I forgot to add btn before btn-primary.

06.11.2024

As I ended right before the end of this section, I had to get back where I ended, and as I did not have preview of the video, I returned to the FIGMA webpage. **flex-direction: column**
in. video-**content** is easy way how to have button right under video preview.

09.11.2024

CSS Grid & Cards

In this part, start was simple and like steps before. Creation of basic elements in html and then more work in CSS. With this, the result that we want are cards with basic information.
Here, I learned some new functions with **GRID**. With Grid, it is possible to specify the exact size of the columns, making it more flexible to control dimensions, spacing, and overall alignment of the elements. I have already used p:nth-child(2), but it is a really good way, how to change something only on one element, here a name.
Creation of the pricing cards was the same process and there was nothing new about that. However, how there are a lot of elements, it was necessary to use keyboard shortcuts, so I get deeper into shortcuts and writing of the code was much faster.
Interesting was a problem with card class where background colour did not work due to specific location in CSS.

FAQ elements

This module finally added some JavaScript. Firstly it was necessary to start with html. This part is the same as modules above, with only few things changed as we want it to act differently. Css is right after, but again, this step is similar with different thing where we wanted to change stuff to look different. Javascript was necessery for FAQ menu. Important was addEventListener to continue after page loaded and then after clicking to the part which we want to expand.

After that, I added the last part, which was a footer of the page. This part was again mainly html work, with change in style in css at the end.

10.11.2024

Mobile Menu & Responsiveness

Adapt webpage to mobile devices is crussial. Most of the users use for web browsing mobile phones and design of a webpage needs to be responsive, otherwise users will not want to return back to this webpage.

I learned (refreshed my memory), that for finding which device is used **css media query** is necessary. I divided devices into 3 categories, mobile devices, larger mobile devices and PCs. For small displays the regular menu is set to none (disappear). I learned how to use **Z-index**, which can be used for hamburger menu to be at the front. I didn't want anything to be over that, as it is important part of the navigation. After stylization in CSS it was important to add buttons functionality in JavaScript. Code here was simple. addEventListener for **hamburger-buttons** click and after clicking, it will change the status to active. In the CSS **mobile-menu.active** set position to 0px and hamburger menu is opened.

11.11.2024

As continued with the video, I repeated, how necessary is to improve the design for mobile devices more with changing size of font, pictures and more in every part of the webpage.

At this point I finished the video and made special repository for this "template" project.

13.11.2024

For my project I have chosen train types used by Czech operator ČD, basic specifications, pictures etc.

First thing that I added is little, but important change which was a Favicon. This feature I have learned during creating project for the course of web programming. I also changed the name of the webpage to **Visit Czech Republic**. These are important details, because it looks better for the user and help them with a navigation between tabs inside their web browser.

13.11.2024

The first thing I had to repair were completely different dimensions of the logo I used in navbar. That created a problem with Navbar consistency. After while I found out, it was my small error, as I made the logo too large, and it broke out of the Navbar's dimensions.

17.11.2024

The same problem with too large a logo appears also in the mobile version of the webpage, but as I already knew, where the problem lay, it was an easy fix. Continuation on recreating of the "here" as it still wasn't as good as I expected. Here I caused a problem with dimensions of the background picture, where 100% of a picture was different from 100% of hero's heading. As I had a trouble with background images and I was stuck, I decided to start mobile menu first and optimalization here was much easier. When I optimized mobile version, I returned to regular size, and I noticed that I had typo in one row. This was easy fix and after that, I changed margin for text and buttons. I also added padding for a heading in the hero, because it didn't look good. Only remaining part was a tablet menu, where I was forced to add another @media, as it didn't look good with the current setting. Now hero look exactly as I wanted.

I don't need a video section, so I decided to add a map of the Czech Republic instead. In this section I used knowledge I gain during the previous course. For a map I have used Leaflet with GeoJSON. The first necessary thing was copy of leaflet CSS and JavaScript file link to html. Then the main code is in JavaScript.

I faced a problem, that after adding all necessary links, coordinates and all other parts only thing that it showed was grey box. I asked ChatGPT what might case this problem. First two answers didn't help (bad links for Leaflet and too small size of the map) but third was bad coordination, not my case, but I noticed, that I typed titleLayer instead of tileLayer. After correction of this small mistake, it worked. I wanted to make our republic even more visible, for that I used Geojson.io. For this was important to make separate file **republic.geojson,** where I copied shape of the republic I made with https://geojson.io/. I didn't add any colours, because I wanted to have only borders. For the large screens it was good even without optimalization, but for small screens was the map too close and borders wasn't visible. I achieve that with adding **zoomSnap**: 0.1, after that, I was able to zoom in step 0.1 instead of 1 and I made optimal map zoom for the mobile and desktop view.

After this, I jumped to FAQ, because I thought, that there will not be necessary to do many changes as FAQ is always similar, but current style of FAQ doesn't look good in corporate colours. I created bubbles separated one from each other and underline every question to be even more visible. Function remains the same.

18.11.2024

Continuation on the "**train types,"** was the next step, here I faced a problem, that container for the train types was the same as for the "**classes**". After a short while I found that when I made comment of one part, I gave comment to bad line and I commented </section>, then the end of the section was after the classes part of the page and container looked bigger. After adding pictures and text it didn't look good, CSS improvements was necessary.

I wanted to add a changing image with arrows, but I wasn't sure how to do that. I used these pages as inspiration: https://www.w3schools.com/howto/howto_js_slideshow.asp
But there was a problem, that images are interconnected across cards. I created different IDs for them and that solve a problem, but there showed another problem. This time it was after reloading of the page when pictures weren't visible. It took too long, and I still wasn't sure how to fix that and instead I started with the last section that are **train classes**. Because I like text aligned to block, I added **text-justify,** changed benefits and added pictures representing classes. Because image was too small, and I wanted to let it in cards I decided to make these **images clickable.** After clicking the image, it expands. Help with code I found on stackoverflow
https://stackoverflow.com/questions/19104018/how-to-expand-an-image-in-html-by-clicking-on-it
CSS was easy to implement, longest time took to stylize it to look good for me. I almost forgot about mobile version of this, so I also improved that.

19.11.2024

I found one small problem with display of travel classes where picture was too deformed for second class. This problem was easily fixed by changing the minimum dimensions. I finally found a solution for the train types with arrow buttons. I added a **coverImage**, these pictures are there after refresh of the webpage and after clicking the buttons disappears. For a long time, I had no idea why my last image disappears, bacause it was supposed to stay there, I was that focused on these two conditions, that I forgot, that I prepared there another 2 **if**s and it deleted my picture. Fix of this problem was easy and then I had completely function section. Last thing here was a design, because the text was too small and overall, the design wasn't consistent. After change in CSS, I am happy with the result, and I deleted some unused parts of code and optimized code a bit.

Now I have a website that introduces travelling in Czech Republic. This webpage is optimized for mobile devices with slightly different layout.

24.11.2024

Last thing was Readme and after writing some basic information I wanted to deploy the website, but I had a problem with npm I gh-pages, so I finally used Netlify.com for deploying the website. After deploying I noticed something weird. All texts in mobile version were incorrectly placed. Here I noticed, that for the whole time I had 80% scale for Chrome instead of 100%. It was necessary to change train class style for the small devices, because two cards next to each other didn’t look good and the same was done with footer.