Lappeenrannan teknillinen yliopisto
School of Business and Management

Software Development Skills

Laura Calleja García-Navas, 002529628

LEARNING DIARY, SOFTWARE DEVELOPMENT SKILLS: FRONT-END MODULE

Learning Diary for Software Course Playlist

Week 1

Day 1

Time Spent: 2 hours

Videos Watched:

- Video 1: "Introduction to Web Programming and HTML Basics"
- Video 2: "Advanced HTML: Forms and Semantic Elements"

What I Learned:

1. HTML forms the foundation of web pages, consisting of tags, attributes, and semantic elements.
2. Key semantic elements such as `<header>`, `<footer>`, and `<article>` improve readability and accessibility.
3. Forms are essential for collecting user input, and attributes like `action` and `method` define how form data is handled.

Reflections:

HTML surprised me with its simplicity yet powerful ability to structure web content. The focus on semantic elements made me realize the importance of accessibility and search engine optimization. Building forms demonstrated the versatility of HTML in creating interactive elements, but I struggled with combining them effectively with CSS. My goal for the next sessions is to practice creating multi-page web applications to solidify my knowledge.

Day 2

Time Spent: 2 hours

Videos Watched:

- Video 3: "Introduction to CSS for Styling Web Pages"
- Video 4: "Advanced CSS: Flexbox and Grid Layouts"

What I Learned:

1. CSS is crucial for designing visually appealing web pages, using selectors, properties, and values.
2. Flexbox and Grid are powerful layout tools for creating responsive, adaptive designs.
3. The importance of external, internal, and inline CSS for structuring styles effectively.

Reflections:
CSS felt challenging but rewarding. Learning Flexbox and Grid opened up a world of possibilities for modern web design. The examples of creating responsive layouts were impressive, but implementing them in practice was tricky. I need to spend more time experimenting with CSS properties to become comfortable with customizing web designs. For the next session, I'll combine HTML and CSS to build a portfolio website.

Week 2

Day 1
Time Spent: 2 hours
Videos Watched:

- Video 5: "JavaScript Basics: Variables, Functions, and Events"
- Video 6: "DOM Manipulation with JavaScript"

What I Learned:

1. JavaScript is a dynamic programming language used to make web pages interactive.
2. Variables, functions, and event listeners allow for dynamic behavior on websites.
3. The DOM (Document Object Model) is a representation of the web page, which JavaScript can manipulate to update content dynamically.

Reflections:
JavaScript brought the web to life for me. The ability to manipulate the DOM and create interactive elements like buttons and animations was exciting. The examples of updating page content with event listeners were particularly engaging, but debugging errors in scripts was a challenge. I plan to practice building interactive forms to strengthen my skills.

Day 2

Time Spent: 2 hours

Videos Watched:

- Video 7: "JSON and AJAX for Data Communication"
- Video 8: "Using APIs with JavaScript"

What I Learned:

1. JSON (JavaScript Object Notation) is a lightweight data format used for data exchange between servers and clients.
2. AJAX (Asynchronous JavaScript and XML) enables asynchronous requests to update web content without refreshing the page.
3. Working with APIs involves sending requests, handling responses, and integrating external data into web applications.

Reflections:

This session made me appreciate the power of JavaScript in creating dynamic, data-driven applications. Understanding AJAX was a breakthrough moment; it demystified how modern websites fetch and display data seamlessly. The integration of APIs felt like unlocking new possibilities, but parsing JSON data sometimes confused me. I'll focus on building small projects using public APIs to solidify this knowledge.

Week 3

Day 1

Time Spent: 2 hours

Videos Watched:

- Video 9: "Introduction to Node.js"
- Video 10: "Building a Web Server with Express.js"

What I Learned:

1. Node.js allows JavaScript to run on the server side, enabling full-stack development with one language.
2. Express.js is a minimal and flexible Node.js framework for building web servers.
3. Middleware in Express.js handles request and response transformations.

Reflections:
Learning Node.js and Express.js expanded my understanding of backend development. Building a simple web server and routing requests felt empowering. The concept of middleware was initially complex, but examples clarified its purpose. I'm eager to integrate a database into an Express project to manage data effectively.

Day 2
Time Spent: 2 hours
Videos Watched:

- Video 11: "Introduction to MongoDB"
- Video 12: "CRUD Operations with MongoDB"

What I Learned:

1. MongoDB is a NoSQL database that stores data in a flexible, JSON-like format.
2. CRUD operations (Create, Read, Update, Delete) are the building blocks for managing database records.
3. Integrating MongoDB with Node.js allows seamless interaction between the backend and the database.

Reflections:
Working with MongoDB felt intuitive due to its JSON-like structure. Performing CRUD operations solidified my understanding of database interactions. Combining MongoDB with Express.js was challenging but rewarding. I'll focus on creating a small web application to manage user data and practice these concepts further.

Week 4

Day 1

Time Spent: 2 hours

Videos Watched:

- Video 13: "Docker Basics for Developers"
- Video 14: "Containerizing Applications with Docker"

What I Learned:

1. Docker simplifies deployment by containerizing applications and their dependencies.
2. Key Docker concepts include images, containers, and Dockerfiles.
3. Containerization ensures consistency across development and production environments.

Reflections:

Docker was a game-changer in understanding how to manage software environments. Creating a Dockerfile and running containers gave me a practical perspective on its utility. However, troubleshooting container issues was complex. My next step is to containerize a full-stack application for hands-on experience.

Day 2

Time Spent: 2 hours

Videos Watched:

- Video 15: "Introduction to React.js"
- Video 16: "State and Props in React"

What I Learned:

1. React.js is a powerful library for building dynamic, component-based user interfaces.
2. Components in React allow for reusable and modular UI development.
3. State and props enable data management within and across components.

Reflections:

React.js felt revolutionary compared to traditional JavaScript approaches. Building components and managing state created a structured and dynamic workflow. Props and state initially

confused me, but practice clarified their roles. I'm eager to create a single-page application to fully explore React's capabilities.

Concluding Thoughts:
This four-week journey through web programming technologies has been both challenging and rewarding. From building the foundational layers with HTML and CSS to mastering JavaScript, Node.js, and React, I've gained a comprehensive understanding of modern web development. Moving forward, I plan to integrate these skills into a capstone project, ensuring they become second nature.

Learning Diary for Software Course Playlist

Week 1

Day 1
Time Spent: 2 hours
Videos Watched:

- Video 1: "Introduction to Web Programming and HTML Basics"
- Video 2: "Advanced HTML: Forms and Semantic Elements"

What I Learned:

1. HTML forms the foundation of web pages, consisting of tags, attributes, and semantic elements.
2. Key semantic elements such as `<header>`, `<footer>`, and `<article>` improve readability and accessibility.
3. Forms are essential for collecting user input, and attributes like `action` and `method` define how form data is handled.

Reflections:
HTML surprised me with its simplicity yet powerful ability to structure web content. The focus on semantic elements made me realize the importance of accessibility and search engine optimization. Building forms demonstrated the versatility of HTML in creating interactive

elements, but I struggled with combining them effectively with CSS. My goal for the next sessions is to practice creating multi-page web applications to solidify my knowledge.

Day 2

Time Spent: 2 hours

Videos Watched:

- Video 3: "Introduction to CSS for Styling Web Pages"
- Video 4: "Advanced CSS: Flexbox and Grid Layouts"

What I Learned:

1. CSS is crucial for designing visually appealing web pages, using selectors, properties, and values.
2. Flexbox and Grid are powerful layout tools for creating responsive, adaptive designs.
3. The importance of external, internal, and inline CSS for structuring styles effectively.

Reflections:

CSS felt challenging but rewarding. Learning Flexbox and Grid opened up a world of possibilities for modern web design. The examples of creating responsive layouts were impressive, but implementing them in practice was tricky. I need to spend more time experimenting with CSS properties to become comfortable with customizing web designs. For the next session, I'll combine HTML and CSS to build a portfolio website.

Week 2

Day 1

Time Spent: 2 hours

Videos Watched:

- Video 5: "JavaScript Basics: Variables, Functions, and Events"
- Video 6: "DOM Manipulation with JavaScript"

What I Learned:

1. JavaScript is a dynamic programming language used to make web pages interactive.
2. Variables, functions, and event listeners allow for dynamic behavior on websites.
3. The DOM (Document Object Model) is a representation of the web page, which JavaScript can manipulate to update content dynamically.

Reflections:
JavaScript brought the web to life for me. The ability to manipulate the DOM and create interactive elements like buttons and animations was exciting. The examples of updating page content with event listeners were particularly engaging, but debugging errors in scripts was a challenge. I plan to practice building interactive forms to strengthen my skills.

Day 2
Time Spent: 2 hours
Videos Watched:

- Video 7: "JSON and AJAX for Data Communication"
- Video 8: "Using APIs with JavaScript"

What I Learned:

1. JSON (JavaScript Object Notation) is a lightweight data format used for data exchange between servers and clients.
2. AJAX (Asynchronous JavaScript and XML) enables asynchronous requests to update web content without refreshing the page.
3. Working with APIs involves sending requests, handling responses, and integrating external data into web applications.

Reflections:
This session made me appreciate the power of JavaScript in creating dynamic, data-driven applications. Understanding AJAX was a breakthrough moment; it demystified how modern websites fetch and display data seamlessly. The integration of APIs felt like unlocking new possibilities, but parsing JSON data sometimes confused me. I'll focus on building small projects using public APIs to solidify this knowledge.

Week 3

Day 1

Time Spent: 2 hours

Videos Watched:

- Video 9: "Introduction to Node.js"
- Video 10: "Building a Web Server with Express.js"

What I Learned:

1. Node.js allows JavaScript to run on the server side, enabling full-stack development with one language.
2. Express.js is a minimal and flexible Node.js framework for building web servers.
3. Middleware in Express.js handles request and response transformations.

Reflections:

Learning Node.js and Express.js expanded my understanding of backend development. Building a simple web server and routing requests felt empowering. The concept of middleware was initially complex, but examples clarified its purpose. I'm eager to integrate a database into an Express project to manage data effectively.

Day 2

Time Spent: 2 hours

Videos Watched:

- Video 11: "Introduction to MongoDB"
- Video 12: "CRUD Operations with MongoDB"

What I Learned:

1. MongoDB is a NoSQL database that stores data in a flexible, JSON-like format.
2. CRUD operations (Create, Read, Update, Delete) are the building blocks for managing database records.

3. Integrating MongoDB with Node.js allows seamless interaction between the backend and the database.

Reflections:
Working with MongoDB felt intuitive due to its JSON-like structure. Performing CRUD operations solidified my understanding of database interactions. Combining MongoDB with Express.js was challenging but rewarding. I'll focus on creating a small web application to manage user data and practice these concepts further.

Week 4

Day 1
Time Spent: 2 hours
Videos Watched:

- Video 13: "Docker Basics for Developers"
- Video 14: "Containerizing Applications with Docker"

What I Learned:

1. Docker simplifies deployment by containerizing applications and their dependencies.
2. Key Docker concepts include images, containers, and Dockerfiles.
3. Containerization ensures consistency across development and production environments.

Reflections:
Docker was a game-changer in understanding how to manage software environments. Creating a Dockerfile and running containers gave me a practical perspective on its utility. However, troubleshooting container issues was complex. My next step is to containerize a full-stack application for hands-on experience.

Day 2
Time Spent: 2 hours
Videos Watched:

- Video 15: "Introduction to React.js"

- Video 16: "State and Props in React"

What I Learned:

1. React.js is a powerful library for building dynamic, component-based user interfaces.
2. Components in React allow for reusable and modular UI development.
3. State and props enable data management within and across components.

Reflections:
React.js felt revolutionary compared to traditional JavaScript approaches. Building components and managing state created a structured and dynamic workflow. Props and state initially confused me, but practice clarified their roles. I'm eager to create a single-page application to fully explore React's capabilities.

Concluding Thoughts:
This four-week journey through web programming technologies has been both challenging and rewarding. From building the foundational layers with HTML and CSS to mastering JavaScript, Node.js, and React, I've gained a comprehensive understanding of modern web development. Moving forward, I plan to integrate these skills into a capstone project, ensuring they become second nature.