INDUSTRIAL TRAINING REPORT

PORTFOLIO WEBSITE

Submitted in partial fulfilment of the

Requirements for the award of

Degree of Bachelor of Technology in Computer Science & Engineering

SUBMITTED BY:

Name: Ankush Singh

University Roll No.: 04096402721

SUBMITTED TO:

Department of Computer Science & Engineering

MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY GGSIPU, DELHI

CERTIFICATE

udemy

Certificate no: UC-81b9ebf4-ff3c-457e-8a10-fe8a4e11ec36
Certificate url: ude.my/UC-81b9ebf4-ff3c-457e-8a10-fe8a4e11ec36
Reference Number: 0004

CERTIFICATE OF COMPLETION

The Complete 2023 Web Development Bootcamp

Instructors **Dr. Angela Yu**

Ankush Singh

Date **31 Jul 2023**
Length **66 total hours**

DECLARATION

I hereby declare that the Training Report entitled Portfolio Website is an authentic record of my own work as requirements of 5 weeks Training during the period from to for the award of degree of B.Tech. (Computer Science & Engineering), GGSIPU, under the guidance of Dr. Angela Yu.

(Signature of student)

ANKUSH SINGH

04096402721

Date: _____

Certified that the above statement made by the student is correct to the best of our knowledge and belief.

Signatures

Examined by:

1. Dr. Angela Yu
(Guide/Trainer)

2.
(Faculty Coordinator)

Head of Department

(Signature and Seal)

ACKNOWLEDGEMENT

The successful completion of this course report would not have been possible without the support and assistance of many individuals and organizations. I feel immensely happy to have chosen this course. I have learnt a lot of new concepts and things during the completion of this course. I would like to take this opportunity to offer my earnest admiration to each and every one of them.

First and foremost, I am highly indebted to **Angela Yu** who has taught the Full Stack Web Development with utmost dedication and preciseness. I had a wonderful and unforgettable experience learning new things.

Then, I would like to thanks my teachers who decided and encouraged me to engage in some useful stuff like this course in these summer break to enhance myskillset so that it could proof fruitful to me in future.

(Signature of the student)

Ankush Singh

04096402721

ABOUT THE PLATFORM

UDEMY

Fig 1. Udemy Platform

1.1 Introduction:

Udemy is one of the leading world-wide online learning marketplaces (e-learning). More than 9 million students participate worldwide. With more than 80,000 courses most topics are covered and taught by experts. The courses are available on demand so that the learning is possible anywhere at any time, on the own pace. Udemy courses are available in 80 languages.

Instruction is delivered by teachers from universities and colleges but by removing the university name and accreditation process it makes top quality training much more affordable. Offering more than 80,000 online courses with many available free of charge, there is definitely something for everyone from business to personal interests.

Udemy has redesigned its platform making e-learning an enjoyable experience through up to date and fully functional online tools. The modern search facility allows students faster answers to questions and quizzes make for an engaging hands-on experience.

By entering the website, you have the menu bar on the left side. You can sort the offered courses by language with the menu bar. Then you can check all the offered courses and filter them by subject. Prices are low with discounts every month and fees are charged per course rather than monthly which means you only pay for what you need. Purchased courses can be saved off-line meaning one payment gives you the keys to your future.

Payments can be affected with the mobile app that uses Apple's or Google Play's payment system depending on the device, often in your local currency. For all offered topics there are also free courses available! To start with Udemy, you can first try a free course.

1.2 Mission and Vision

- We work at a purpose-driven company that places a high value on results, so we can continue to grow and give as many people as possible access to learning resources. We immerse ourselves in data and communicate directly with our instructors and students to ensure we're making real progress toward our goals—and theirs.

- With a mission to improve lives through learning, it's no surprise that we place tremendous emphasis on the role of learning in our own lives and work. We understand that learning isn't an interruption of our work or a sign of weakness—it's foundational to our growth, both as individuals and as a business.

- Individuals make great contributions, but it takes collaboration, compromise, and kindness to build a great business. We succeed as a team, leave our egos at the door, and take pride in our shared efforts.

- We embrace our quirks and bring our whole selves to work. We recognize that it's our differences that produce the best work, not conformity, so we are excited to welcome coworkers with varied backgrounds, experiences, identities, and ideas. Our openness allows us to better serve the diverse people who use Udemy across the globe.

1.3 Origin

Growing up in a small Turkish village with a one-room schoolhouse, our founder Eren Bali had limited educational opportunities, until his family got a computer. Fueled by his dreams to compete internationally in chess and mathematics, Eren used the internet to access learning resources and connect with people all over the world. With the help of these communities, he earned a silver medal in the International Math Olympiad.

After online learning changed his life, Eren partnered with co-founders Oktay Caglar and Gagan Biyani toward a common goal: to make quality education more accessible and improve lives through learning.

1.4 Overview

Udemy is a platform that allows instructors to build online courses on their preferred topics. Using Udemy's course development tools, they can upload videos, PowerPoint presentations, PDFs, audio, ZIP files and live classes to create courses. Instructors can also engage and interact with users via online discussion boards.

Courses are offered across a breadth of categories, including business and entrepreneurship, academics, the arts, health and fitness, language, music, and technology. Most classes are in practical subjects such as Excel software or using an iPhone camera. Udemy also offers Udemy for Business, enabling businesses access to a targeted suite of over 7,000 training courses on topics from digital marketing tactics to office productivity, design, management, programming, and more. With Udemy for Business, organizations can also create custom learning portals for corporate training.

Courses on Udemy can be paid or free, depending on the instructor. In 2015, the top 10 instructors made more than $17 million in total revenue.

In April 2013, Udemy offered an app for Apple iOS, allowing students to take classes directly from iPhones; The Android version was launched in January 2014. As of January 2014, the iOS app had been downloaded over 1 million times, and 20 percent of Udemy users access their courses via mobile. In July 2016, Udemy expanded their iOS platform to include Apple TV. On January 11, 2020, the Udemy mobile app became the #1 top grossing Android app in India .

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter-1: INTRODUCTION

1.1 Introduction:

During my summer break after 2nd year of my graduation, I have completed this course, which included a project on full stack Web Development.

I have selected this course named "The Complete 2023 Web Development Bootcamp" by the instructor Dr. Angela Yu. This course was completed on July 31st 2023, summing up to the durations of 4-6 weeks.

During the course, I was working on the project entitled "Portfolio Website" which is a full-stack website using HTML, CSS, JavaScript, Express.js, Node.js and MongoDB. This website is a complete full-stack project with modern, interactive and responsive design.

Web Development:

Web development is the building and maintenance of websites; it's the work that happens behind the scenes to make a website look great, work fast and perform well with a seamless user experience.

Web developers, or 'devs', do this by using a variety of coding languages. The languages they use depends on the types of tasks they are preforming and the platforms on which they are working.

Web development skills are in high demand worldwide and well paid too – making development a great career option. It is one of the easiest accessible higher paid fields as you do not need a traditional university degree to become qualified.

The field of web development is generally broken down into front-end (the user- facing side) and back-end (the server side). Let's delve into the details.

Overview Of Web Development:

A front-end dev takes care of layout, design and interactivity using HTML, CSS and JavaScript. They take an idea from the drawing board and turn it into reality.

What you see and what you use, such as the visual aspect of the website, the drop down menus and the text, are all brought together by the front-end dev, who writes a series of programmes to bind and structure the elements, make them look good and add interactivity. These programmes are run through a browser.

The backend developer engineers what is going on behind the scenes. This is where the data is stored, and without this data, there would be no frontend. The backend of the web consists of the server that hosts the website, an application for running it and a database to contain the data.

The backend dev uses computer programs to ensure that the server, the application and the database run smoothly together. This type of dev needs to analyse what a company's needs are and provide efficient programming solutions. To do all this amazing stuff they use a variety of server-side languages, like PHP, Ruby, Python and Java.

Fig-2. Web Development

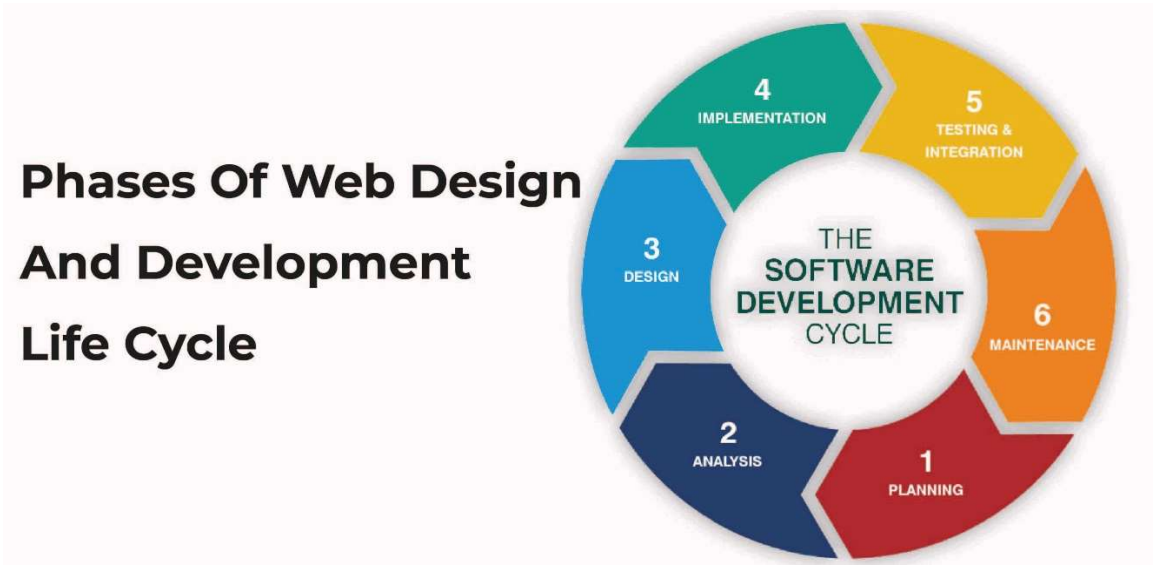

Fig-3. Different phases of web development

ABOUT THE COURSE

1.1 Description:

Welcome to the Complete Web Development Bootcamp, **the only course you need** to learn to code and become a full-stack web developer. With 150,000+ ratings and a 4.8 average, my Web Development course is one of the HIGHEST RATED courses in the history of Udemy!

At 65+ hours, this Web Development course is without a doubt the **most comprehensive** web development course available online. Even if you have **zero** programming experience, this course will take you from **beginner to mastery**. Here's why:

- The course is taught by the **lead instructor** at the App Brewery, London's **leading in-person programming bootcamp**.
- The course has been updated to be **2023 ready** and you'll be learning the latest tools and technologies used at large companies such as Apple, Google and Netflix.
- This course doesn't cut any corners, there are beautiful **animated explanation videos** and tens of **real-world projects** which you will get to build.
- The curriculum was developed over a period of **four years**, with comprehensive student testing and feedback.
- We've taught over a **million** students how to code and many have gone on to **change their lives** by becoming professional developers or starting their own tech startup.
- You'll save yourself **over $12,000** by enrolling, but still get access to the same teaching materials and learn from the same instructor and curriculum as our in-person programming bootcamp.
- The course is **constantly updated** with new content, with new projects and modules determined by students - that's you!

We'll take you **step-by-step** through engaging video tutorials and teach you everything you need to know to succeed as a web developer.

The course includes over **65 hours** of HD video tutorials and builds your programming knowledge while making real-world websites and web apps.

Throughout this comprehensive course, we cover a massive amount of tools and technologies, including:

- Front-End Web Development
- **HTML** 5
- **CSS** 3
- **Flexbox**
- **Grid**
- Bootstrap 5
- **Javascript** ES6
- DOM Manipulation
- **jQuery**
- Bash Command Line
- Git, **GitHub** and Version Control
- Backend Web Development
- **Node**.js
- NPM

- **Express**.js
- EJS
- REST
- **APIs**
- Databases
- SQL
- **MongoDB**
- Mongoose
- Authentication
- Firebase
- **React**.js
- React Hooks
- Web Design
- Deployment with GitHub Pages, Heroku and MongoDB Atlas
- Web3 Development on the Internet Computer
- Blockchain technology
- Token contract development
- NFT minting, buying and selling logic

By the end of this course, you will be **fluently** programming and be ready to **make any website you can dream of.**

You'll also build a **portfolio** of over 32+ websites that you can **show off** to any potential employer.

Sign up today, and look forward to:

- Animated Video Lectures
- Code Challenges and Coding Exercises
- Beautiful Real-World Projects
- Quizzes & Practice Tests
- Downloadable Programming Resources and Cheatsheets
- Our best selling 12 Rules to Learn to Code eBook
- $12,000+ worth of web development bootcamp course materials and course curriculum

Don't just take my word for it, check out what existing students have to say about the course:

"Angela is just incredible, awesome and just fantastic in this course. I've never had such an instructor; **detailed** in every aspect of the course, gives precise explanations, gives you the anxiety to learn etc. She's got that ability to make fun while explaining things for better understanding. I really love this course." - Ekeu MonkamUlrich

"Angela is very thorough without ever being boring. I've taken MANY online courses in my life including my Bachelors and Masters degrees. She is by far the best instructor I've ever had. This course is **packed with thousands of dollars worth of great instruction**, and paced well enough for anyone to pick coding up and run with it- Thank you!" - J Carlucci

"Love the way Angela explains things. Easy to follow and full of logic. I can say she must have spent a lot of energy creating this great course. Thank you and I recommend it to all who's interested in coding!" - Yiqing Zheng

"So far (on my third day) this course has taught me **more than I was able to learn in multiple other programming courses**. This course is clearly outlined and builds upon itself gradually in an easy to understand way." - Normal Ramsey

"This course will take you from beginner to intermediate level for real. If you don't know how to put together the pieces of web development this is what you're looking for. Angela explains in an amazing way by **creating projects** all the way during this course, explaining the concepts in real practice. Thank you very much, Angela. I will always consider you my mentor. Look forward to taking more courses with you." - Moises Dionisio Cruz

"An amazing course, perfect for absolute beginners at the start of their coding journey! Angela is an amazing tutor and can explain in the most simple and comprehensible way even complex coding notions. **Learning web development cannot get any more fun!**" - Zoe Moyssoglou

"It's a different approach to teaching Web Development. I like that you are **given everything possible to succeed** from the onset." - Ronick Thomas

The tutor is simply AMAZING, by far the best tutor I have ever had. I would give her 10 stars out of 5. She is not just punching the code and talking to herself, but she is actually explaining things. She keeps on giving really useful hints and she will give you a great load of other references. I always knew what I was doing and why I was doing it. All the extra challenges have just made me remember and understand things better. - Peter Dlugos

REMEMBER… I'm so confident that you'll love this course that we're offering a FULL money-back guarantee for 30 days! So it's a complete no-brainer, sign up today with ZERO risk and EVERYTHING to gain.

So what are you waiting for? Click the buy now button and join the world's highest-rated web development course.

1.2 Training Objective:

- Training are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships or courses in order to receive real world experience and develop their skills which can be further used in jobs .

- An objective for this position should emphasize the skills you already possess in the area and your interest in learning more.

- Internships, Training and courses are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.

- Some courses is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.

- Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

1.3 Motivation:

After thinking the idea of the project and creating the basic framework for it, main task was to explore new and cool features to add in the website to make it look more attractive and wiser. So, for that I surfed a lot of websites and watched a lot of videos to incorporate those cool features in my project.

E-learning is a subset of education technology which offers an online learning and teaching platform to disperse knowledge with the help of internet technology. E-learning offers conceptual and experimental learning through machines, media platforms and network solutions. E-learning takes place both inside and outside of the classrooms. As, Udemy is one of the most reputable E-learning platforms, at first, I came to know about the course through LinkedIn. But later after researching further I got its complete details. Being impressed by the objectives and content of the course, I decided to choose this course.

Chapter-2: TOOLS AND TECHNOLOGY USED

2.1 Languages and frameworks used:

HTML (Hypertext Markup Language):

- HTML is the standard markup language used to create web pages.
- It consists of a set of elements and tags that define the structure and content of a webpage.
- HTML is essential for creating the basic structure of a webpage, including headings, paragraphs, links, images, and more.

CSS (Cascading Style Sheets):

- CSS is used for styling and formatting web pages created with HTML.
- It allows you to control the layout, design, and presentation of web content.
- CSS can be used to define colors, fonts, spacing, and positioning of elements on a webpage.

JavaScript:

- JavaScript is a versatile programming language used for adding interactivity to web pages.
- It can be used to manipulate HTML and CSS, handle user input, create animations, and communicate with web servers.
- JavaScript is essential for creating dynamic and responsive web applications.

Express.js:

- Express.js is a minimal and flexible Node.js web application framework.
- It simplifies the process of building robust and scalable web applications by providing a set of powerful features and middleware.
- Express.js is often used for creating RESTful APIs and handling routing in web applications.

Node.js:

- Node.js is a runtime environment that allows you to run JavaScript on the server-side.
- It is built on the V8 JavaScript engine and enables developers to create server-side applications, including web servers and APIs.
- Node.js is known for its non-blocking, event-driven architecture, making it suitable for handling I/O-intensive tasks.

MongoDB:

- MongoDB is a NoSQL database management system.
- It stores data in a flexible, JSON-like format called BSON (Binary JSON).
- MongoDB is commonly used for handling large volumes of unstructured or semi-structured data and is often chosen for web applications that require scalability and flexibility in data storage.

These technologies are often used together to build modern web applications. HTML and CSS handle the structure and styling of web pages, JavaScript adds interactivity, Express.js and Node.js manage the server-side logic, and MongoDB provides a data storage solution.

2.2 Software's Used:

Visual Studio Code (VS Code):

Visual Studio Code (VS Code) is a widely embraced, free-to-use code editor created by Microsoft. It stands out for its speed, adaptability, and an extensive ecosystem of extensions, making it a top choice for developers across various domains.

Fig-4. VS Code

One of VS Code's key strengths is its cross-platform compatibility. It runs seamlessly on Windows, macOS, and Linux, catering to the preferences of developers irrespective of their operating systems. The editor is known for its lightweight nature, ensuring swift startup times and efficient performance, even when handling extensive codebases.

The standout feature of VS Code is its extensibility. It boasts a thriving community of developers who contribute an array of extensions that enable users to customize and enrich their coding environment. These extensions span various programming languages, tools, and integrations, enhancing the editor's capabilities and tailoring it to specific development needs.

Despite being primarily a code editor, VS Code offers an impressive range of integrated development environment (IDE) features through its extensions. These include robust code navigation, debugging tools, seamless version control (such as Git integration), and an integrated terminal for command-line operations, all within a single, cohesive interface.

Intelligent code assistance is another forte of VS Code, offering features like auto-completion, code formatting, syntax highlighting, and linting. These aid developers in writing clean and error-free code, ultimately boosting productivity.

Additionally, VS Code is renowned for its versatility in supporting multiple programming languages, with built-in support for many and the capability to add language-specific extensions for others. This flexibility makes it an attractive choice for developers engaged in various projects and technologies.

Chapter-3: TECHNICAL CONTENTS

3.1 Introduction to HTML

What is HTML?

- HTML stands for **Hypertext Markup Language**.
- It is the standard markup language for creating web pages.
- HTML is used to structure content on the web, including text, images, links, forms, and multimedia.

How Does HTML Work?

- HTML documents consist of a series of elements (tags) that define the structure and content of a web page.
- Browsers interpret these elements to render web pages to users.

Basic Structure of an HTML Document:

```
<!DOCTYPE html>
<html>
<head>
    <title>Page Title</title>
</head>
<body>
    <h1>Heading 1</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

Fig-5. Basic Structure of HTML

Output:

HTML Elements and Tags

HTML Elements:

- Elements are the building blocks of HTML documents.
- An element consists of an opening tag, content, and a closing tag.

Common HTML Tags:

- **<html>**: The root element that contains all other HTML elements.
- **<head>**: Contains meta-information about the document, like title and links to stylesheets.
- **<title>**: Sets the title of the web page (displayed in the browser's title bar).
- **<body>**: Contains the visible content of the web page.
- **<h1>**, **<h2>**, **<h3>**, ... **<h6>**: Headings with decreasing levels of importance.
- **<p>**: Represents a paragraph of text.
- **<a>**: Creates hyperlinks to other web pages.
- **<img>**: Embeds images on a web page.
- **<ul>**, **<ol>**, **<li>**: Used for creating unordered and ordered lists.
- **<table>**, **<tr>**, **<td>**: Create tables for data presentation.
- **<form>**: Used to create input forms.

HTML Attributes

Attributes:

- HTML elements can have attributes that provide additional information about the element.
- Attributes are always specified in the opening tag.
- **Common HTML Attributes:**
- **class**: Used for applying CSS styles or for JavaScript manipulation.
- **id**: Provides a unique identifier for an element.
- **src**: Specifies the source URL for elements like images and scripts.
- **href**: Specifies the destination URL for links.
- **alt**: Provides alternative text for images (for accessibility).
- **style**: Used to apply inline CSS styles.
- **target**: Specifies where to open linked documents (e.g., in a new tab or window).

HTML Forms

HTML Forms:

- HTML forms allow users to input data and submit it to a web server for processing.
- Common form elements include **<input>**, **<textarea>**, **<select>**, and **<button>**.

Form Attributes:

- **action**: Specifies the URL where the form data should be sent.
- **method**: Defines the HTTP method (GET or POST) used to send data.

Input Types:

- **<input>** elements can have various types, such as text, password, checkbox, radio, etc.

HTML Semantic Elements

Semantic HTML:

- Semantic HTML elements convey the meaning of their content to both browsers and developers.
- Examples: **<header>**, **<nav>**, **<article>**, **<section>**, **<footer>**, **<aside>**, **<figure>**, **<figcaption>**

HTML5 Multimedia

Multimedia Elements:

- HTML5 introduced elements like **<audio>** and **<video>** for embedding multimedia content.

HTML Links and Anchors

Hyperlinks:

- Hyperlinks are used to navigate between web pages or resources.
- **<a>** tags are used to create hyperlinks.

Anchor Tags:

- **<a>** tags use the **href** attribute to specify the destination URL.

Fig-6. HTML Logo

3.2 Introduction to CSS

What is CSS?

- CSS stands for Cascading Style Sheets.
- It is a stylesheet language used to describe the presentation (styling) of HTML documents.
- CSS allows you to control the layout, colors, fonts, and other visual aspects of a web page.

The Role of CSS:

- HTML provides the structure and content of a web page.
- CSS enhances the presentation and appearance of that content.

Basic Syntax of CSS:

```
selector {
    property: value;
}
```

Fig-7. Basic syntax of CSS

CSS Selectors and Properties:

CSS Selectors:

- Selectors are patterns that define which HTML elements the CSS rules should apply to.
- Common selectors include element selectors, class selectors (.classname), and ID selectors (#elementID).

CSS Properties:

- Properties are attributes that define how an element should be styled.
- Examples include color, font-size, background-color, margin, padding, and border.

CSS Box Model

Box Model:

- The CSS box model describes how elements are rendered as rectangular boxes.
- It consists of content, padding, border, and margin.

Box Model Properties:

- width and height: Define the size of the content box.

- padding: Space between the content and the border.
- border: The border around the content and padding.
- margin: Space outside the border, creating separation between elements.

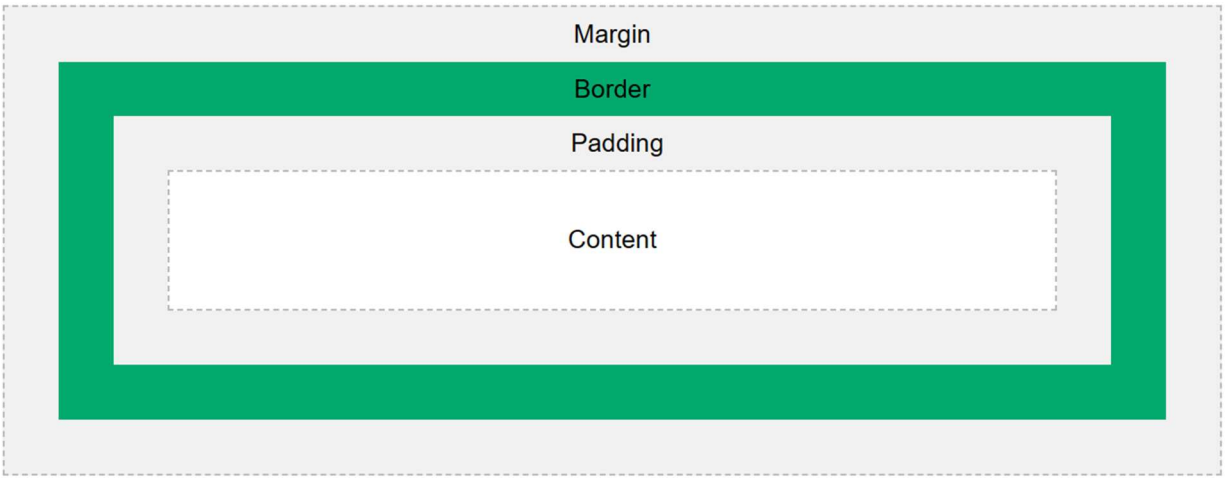

Fig-8. CSS Box Model

CSS Layout

CSS Positioning:

- CSS provides different positioning schemes, including static, relative, absolute, and fixed.

Display Property:

- The display property controls how an element is displayed in the layout flow.
- Common values include block, inline, inline-block, and none.

Floats:

- The float property is used to make elements float left or right within their parent container.

CSS Flexbox and Grid

Flexbox:

- CSS Flexbox is a layout model that simplifies the alignment and distribution of elements in a container.
- It's particularly useful for one-dimensional layouts.

Grid Layout:

- CSS Grid is a two-dimensional layout system that allows precise control over rows and columns in a grid container.

CSS Transitions and Animations

CSS Transitions:

- Transitions enable smooth property changes, such as hover effects or fade-ins.
- Properties like transition-property, transition-duration, and transition-timing-function control transitions.

CSS Animations:

- CSS animations allow for more complex, timed animations.
- Keyframes are used to define animation steps.

Responsive Web Design

Responsive Design:

- CSS is crucial for creating responsive web designs that adapt to different screen sizes and devices.
- Media queries and relative units like percentages and em are commonly used.

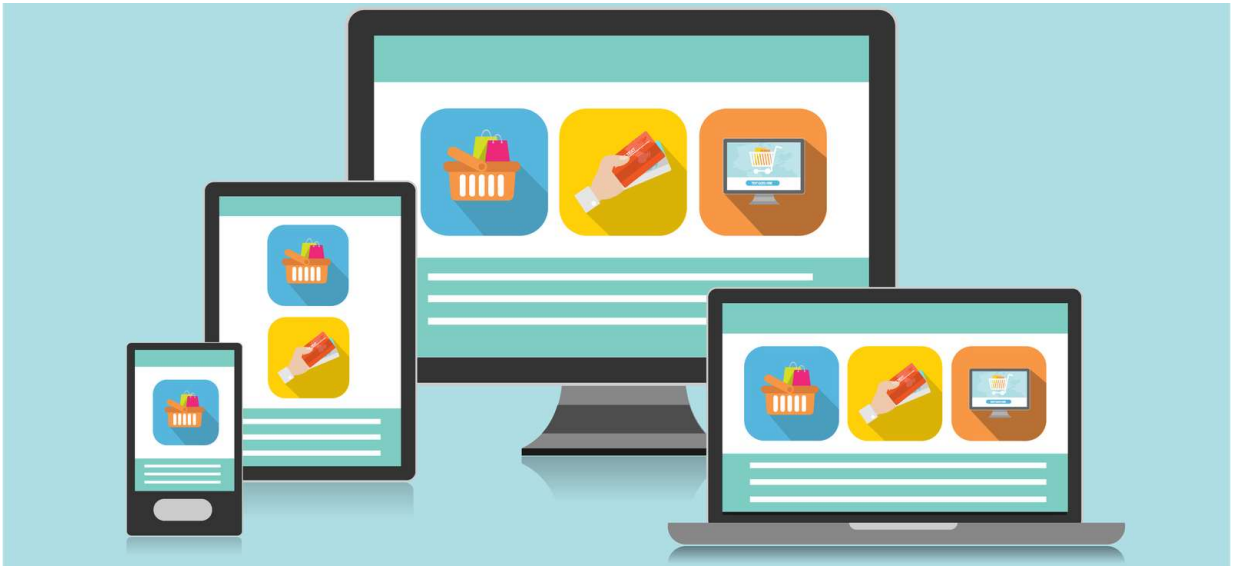

Fig-9. Responsive Web Design

CSS Preprocessors

CSS Preprocessors:

- Preprocessors like Sass and Less extend CSS with variables, functions, and more, making it easier to manage and reuse styles.

3.3 Introduction to JavaScript

What is JavaScript?

- JavaScript is a versatile, high-level, and interpreted programming language.
- It was initially created for web development to add interactivity to web pages.
- JavaScript is now widely used for both front-end and back-end development.

JavaScript in Web Development:

- JavaScript enhances the user experience by providing interactivity and dynamic functionality on web pages.
- It complements HTML and CSS, allowing developers to create feature-rich web applications.

Basic Syntax of JavaScript:

```javascript
// Variables
var variableName = value;
let anotherVariable = value;
const constantVariable = value;

// Functions
function functionName(parameters) {
    // Function code
}

// Conditional Statements
if (condition) {
    // Code to execute if the condition is true
} else {
    // Code to execute if the condition is false
}

// Loops
for (let i = 0; i < length; i++) {
    // Loop code
}

// Data Types
let number = 42;
let string = "Hello, World!";
let boolean = true;
let array = [1, 2, 3];
let object = { key: "value" };
```

Fig-10. Basic JavaScript Syntax

JavaScript Variables and Data Types

Variables in JavaScript:

- Variables store data values and can be changed during execution.
- Use **var**, **let**, or **const** to declare variables.
- **var** has function scope, while **let** and **const** have block scope.

Common Data Types:

- JavaScript supports various data types, including numbers, strings, booleans, arrays, objects, and more.

Data Type	Example	Description
Number	**42**, **3.14**	Numeric values
String	**"Hello, World!"**	Text
Boolean	**true**, **false**	True or false values
Array	**[1, 2, 3]**	Ordered list of values
Object	**{ key: "value" }**	Collection of key-value pairs
Function	**function() {...}**	Reusable blocks of code

Table-1. JavaScript Data Types

Type Coercion:

- JavaScript can automatically convert data types when needed, which is called type coercion.
- For example, adding a number to a string can result in concatenation.

JavaScript Functions and Control Flow

Functions in JavaScript:

- Functions are blocks of code that can be reused.
- They can accept parameters and return values.
- Functions can be declared using the **function** keyword or as arrow functions (**() => {}**).

Conditional Statements:

- JavaScript supports if, else if, and else statements for conditional logic.

Loops:

- Use for, while, and do...while loops for repetitive tasks.

Switch Statement:

- The switch statement simplifies multiple if...else conditions.

JavaScript Objects and Arrays

JavaScript Objects:

- Objects are collections of key-value pairs.
- They can represent complex data structures.

JavaScript Arrays:

- Arrays are ordered collections of values.
- They are versatile and can hold different data types.

JavaScript DOM Manipulation:

DOM (Document Object Model):

- The DOM represents the structure of an HTML document as a tree of objects.
- JavaScript can manipulate the DOM to change content, structure, and styles dynamically.

Selecting DOM Elements:

- JavaScript can select elements using methods like getElementById, querySelector, and getElementsByClassName.

Modifying DOM Elements:

- JavaScript can change element content, attributes, and styles.

JavaScript Events

Events in JavaScript:

- Events are actions or occurrences that happen in the browser.
- JavaScript can respond to events like clicks, mouse movements, and keyboard inputs.

Event Listeners:

- Event listeners are used to trigger JavaScript code when specific events occur.

JavaScript Asynchronous Programming

Asynchronous JavaScript:

- JavaScript supports asynchronous operations, like fetching data from a server or handling user input.

Callbacks, Promises, and Async/Await:

- Callback functions, Promises, and async/await are used to manage asynchronous code.

Advantages

JavaScript is executed on client side

This means that the code is executed on the user's processor instead of the web server thus saving bandwidth and strain on the web server.

JavaScript is relatively easy language

The JavaScript language is relatively easy to learn and comprises of syntax that is close to English. It uses the DOM model that provides plenty of prewritten functionality to the various objects on pages making it a breeze to develop a script to solve a custom purpose.

JavaScript is relatively fast to end users

As the code is executed on the user's computer, results and processing is completed almost instantly depending on the task as it does not need to be processed in the site's web server and sent back to the user consuming local as well as server bandwidth.

Extended functionality to web pages

Third party add-ons like Grease monkey enable JavaScript developers to write snippets of JavaScript which can execute on desired web pages to extend its functionality. If you use a website and require a certain feature to be included, you can write it yourself and use an add-on like grease monkey to implement it on the web page.

3.4 Introduction to Express.js

Fig-11. Express.js Logo

What is Express.js?

- Express.js is a web application framework for Node.js.
- It simplifies the development of web applications and APIs by providing a structured and minimalistic approach.

Why Use Express.js?

- Express.js streamlines the creation of web servers and APIs.
- Offers a wide range of middleware for various tasks.
- Supports extensibility through third-party libraries.
- Suitable for both small projects and large-scale applications.

Basic Syntax of Express.js:

Express.js applications typically start by creating an instance of the **express** object and defining routes and middleware.

```
1  const express = require('express');
2  const app = express();
3  const port = 3000;
4
5  app.get('/', (req, res) => {
6      res.send('Hello, Express!');
7  });
8
9  app.listen(port, () => {
10     console.log(`Server is running on port ${port}`);
11 });
```

Fig-12. Basic Syntax of Express.js

Express.js Routing

Routing in Express.js:

- Express.js allows developers to define routes for handling different HTTP methods (e.g., GET, POST, PUT, DELETE).
- Routes specify how the application responds to client requests based on the requested URL and HTTP method.

HTTP Method	Example Route	Description
GET	**app.get('/users', ...)**	Handling GET requests to **/users**
POST	**app.post('/login', ...)**	Handling POST requests to **/login**
PUT	**app.put('/profile', ...)**	Handling PUT requests to **/profile**
DELETE	**app.delete('/delete', ...)**	Handling DELETE requests to **/delete**

Table-2. Express.js HTTP Routing methods

Express.js Middleware

Middleware in Express.js:

- Middleware functions are used to perform various tasks in the request-response cycle.
- They can be applied globally or to specific routes and are executed in the order they are defined.
- Middleware is often used for tasks like authentication, logging, and data parsing.

Express.js Template Engines

Template Engines in Express.js:

- Template engines facilitate the dynamic generation of HTML pages.
- Popular template engines for Express.js include EJS, Handlebars, and Pug (formerly Jade).
- Template engines allow developers to inject data into HTML templates to produce dynamic content.

Express.js Middleware for Handling Forms and Data

Handling Forms and Data:

- Express.js can handle form data and file uploads using middleware like **body-parser** and **multer**.
- Middleware is used to parse incoming data and make it accessible to route handlers.

Express.js Error Handling

Error Handling in Express.js:

- Express.js provides error-handling middleware to manage errors that occur during request processing.

- Custom error-handling middleware can be defined with four parameters (err, req, res, next).

Express.js Static Files

Serving Static Files:

- Express.js can serve static assets like CSS, JavaScript, and images using the **express.static** middleware.
- Static files are typically placed in a designated directory and are made accessible to clients.

Express.js with RESTful APIs

RESTful APIs with Express.js:

- Express.js is commonly used to create RESTful APIs for client-server communication.
- Routes and middleware are used to define API endpoints and handle HTTP requests and responses.

```
app.get('/api/users', (req, res) => {
    // Retrieve and send user data as JSON
});

app.post('/api/users', (req, res) => {
    // Create a new user and send a response
});
```

Express.js is a powerful and flexible web application framework for Node.js. It simplifies the process of building web servers, handling routes, middleware, and templates. With its extensive ecosystem and strong community support, Express.js is a popular choice for developing web applications and RESTful APIs. Understanding the basic concepts of Express.js is essential for modern web development.

3.5 Introduction to Node.js

What is Node.js?

- Node.js is an open-source, server-side JavaScript runtime environment.
- It allows you to run JavaScript code on the server, extending JavaScript's capabilities beyond the browser.

Key Features of Node.js:

- **Asynchronous and Non-blocking:** Node.js is designed to handle multiple tasks simultaneously without blocking the execution of other tasks. It uses event-driven architecture and callbacks to achieve this.

- **Single-threaded:** While Node.js uses a single thread, it efficiently manages concurrency, making it suitable for handling a large number of connections.

- **V8 JavaScript Engine:** Node.js is built on the V8 engine from Google, which compiles JavaScript code to native machine code for high performance.

- **npm (Node Package Manager):** npm is the default package manager for Node.js, providing access to a vast ecosystem of reusable libraries and modules.

Basic Syntax of Node.js:

- Node.js code follows JavaScript syntax and conventions, making it easy for developers familiar with JavaScript to transition to Node.js.
- Variables, functions, conditionals, loops, and data types are used similarly to browser-based JavaScript.

```
// Importing modules
const fs = require('fs');

// Defining variables
let variableName = value;

// Creating functions
function functionName(parameters) {
    // Function code
}

// Using callbacks
fs.readFile('file.txt', 'utf8', (err, data) => {
    if (err) throw err;
    console.log(data);
});
```

Fig-13. Basic syntax of Node.js

Node.js Modules

Node.js Modules:

- Node.js uses a module system for organizing code into reusable and maintainable units.
- Modules can be core modules (built-in), local modules (created by the developer), or third-party modules (installed via npm).

Importing and Exporting Modules:

- Modules are imported using the **require** function.
- Modules can export functions, objects, or variables for use in other parts of the application.

Node.js File System (fs) Module

File System (fs) Module:

- The **fs** module in Node.js allows you to work with the file system, including reading, writing, and manipulating files and directories.
- It provides both synchronous and asynchronous methods for file operations.

Node.js HTTP Module

HTTP Module:

- Node.js's **http** module is used to create HTTP servers and handle HTTP requests and responses.
- This module is fundamental for building web applications, RESTful APIs, and serving web content.

Node.js Package Management (npm)

npm (Node Package Manager):

- npm is a package manager for Node.js that simplifies package installation, version management, and dependency resolution.
- Developers can publish and share packages on the npm registry.

Basic npm commands:

Command	Description
npm init	Initialize a new Node.js project.
npm install packageName	Install a package and save it to package.json.
npm start	Run the start script defined in package.json.
npm test	Run tests defined in the test script of package.json.
npm publish	Publish a package to the npm registry.

Table-3. Basic npm commands

Node.js Asynchronous Programming

Asynchronous Programming:

- Node.js excels at asynchronous programming, allowing tasks to run concurrently without blocking the main thread.
- Callbacks, Promises, and **async/await** are commonly used patterns to handle asynchronous operations gracefully.

Node.js Events and EventEmitter

Events and EventEmitter:

- Node.js uses an event-driven architecture where objects (EventEmitters) emit events, and listeners respond to those events.
- The **events** module provides the EventEmitter class to create and manage custom events.

Node.js Built-in Modules

Built-in Modules:

- Node.js offers a rich set of core modules for various tasks, such as file I/O (**fs**), network operations (**http**, **https**), path manipulation (**path**), and more.
- These modules simplify common tasks and provide access to low-level system functions.

Node.js is a versatile and powerful runtime environment that extends the capabilities of JavaScript beyond the browser. Understanding the fundamentals of Node.js, its module system, asynchronous programming, and best practices is crucial for developing scalable, high-performance applications, APIs, and server-side solutions. By mastering these concepts, developers can harness the full potential of Node.js for building modern web applications and server-based software.

3.6 Introduction to MongoDB

What is MongoDB?

- MongoDB is a NoSQL database management system designed for storing, retrieving, and managing large volumes of data.
- Unlike traditional relational databases, MongoDB uses a document-oriented data model, which means data is stored in flexible, JSON-like BSON (Binary JSON) documents.

Key Features of MongoDB:

- **Document-Oriented**: MongoDB stores data in collections of documents, where each document can have a different structure.
- **No Fixed Schema**: There's no requirement for a fixed schema across all documents in a collection, making MongoDB highly flexible.
- **Horizontal Scalability**: MongoDB supports horizontal scaling, distributing data across multiple servers or clusters for high availability and performance.
- **Rich Query Language**: MongoDB provides a robust query language with support for complex queries, indexing, and geospatial operations.
- **Replication**: MongoDB offers built-in data replication for fault tolerance and data redundancy.
- **Aggregation Framework**: It includes an aggregation framework for performing complex data transformations and analytics.
- **Full-Text Search**: MongoDB supports full-text search capabilities.

MongoDB Basic Syntax:

```
// Switch to a database (creates if it doesn't exist)
use mydb

// Insert a document into a collection
db.users.insert({
    name: "Alice",
    age: 30,
    email: "alice@example.com"
})

// Find documents in a collection
db.users.find({ name: "Alice" })

// Update documents in a collection
db.users.update(
    { name: "Alice" },
    { $set: { age: 31 } }
)

// Remove documents from a collection
db.users.remove({ name: "Alice" })
```

Fig-14. MongoDB Basic syntax

Installing MongoDB:

- To get started with MongoDB, you need to install the MongoDB server on your system. The installation process varies depending on your platform. Refer to the official MongoDB documentation for detailed installation instructions.

Starting MongoDB Server:

- You can start the MongoDB server using the **mongod** command. This command initializes the server and prepares it to accept connections from clients.

Basic MongoDB Commands:

- MongoDB commands are primarily used in the MongoDB shell (**mongo**), which provides an interactive environment for working with MongoDB.

Command	Description
use databaseName	Switches to a specific database or creates it if it doesn't exist.
db.collectionName.insert(data)	Inserts a document into a collection.
db.collectionName.find(query)	Retrieves documents from a collection based on a query.
db.collectionName.update(query, update)	Updates documents in a collection based on a query.
db.collectionName.remove(query)	Removes documents from a collection based on a query.
show collections	Lists the collections in the current database.
db.collectionName.drop()	Deletes a collection and all its documents.

Table-4. Basic MongoDB commands

MongoDB Data Modeling

Data Modeling in MongoDB:

- Data modeling in MongoDB involves designing how data will be structured and organized within collections.
- Collections in MongoDB are similar to tables in relational databases, and documents are similar to rows.

Embedding vs. Referencing:

- When designing MongoDB data models, you can choose between embedding documents within other documents or referencing documents in different collections.
- Embedding is suitable for one-to-few relationships, while referencing is more appropriate for one-to-many or many-to-many relationships.

MongoDB Indexing

Indexing in MongoDB:

- Indexes improve query performance by providing a faster way to access data.
- MongoDB automatically creates an index on the **_id** field for each document (to facilitate fast lookups).
- Custom indexes can be created on other fields to optimize query performance.

MongoDB Aggregation

Aggregation in MongoDB:

- The aggregation framework in MongoDB is a powerful tool for data transformation and analysis.
- It allows you to perform operations like filtering, grouping, sorting, and computing aggregated results.

MongoDB Replication and Sharding

Replication in MongoDB:

- Replication in MongoDB involves maintaining multiple copies (replica sets) of data across different servers.
- This provides fault tolerance and high availability. If one server fails, another can take over.

Sharding in MongoDB:

- Sharding is a technique to distribute data across multiple servers or clusters called shards.
- It helps MongoDB handle large datasets and heavy workloads by horizontally scaling data storage and processing.

MongoDB is a versatile NoSQL database that excels in handling unstructured or semi-structured data. Its flexibility, scalability, and rich feature set make it suitable for a wide range of applications, from simple document storage to complex data analysis. Understanding MongoDB's basic syntax, data modeling options, indexing, and aggregation framework is essential for effectively working with this database system. By following best practices, you can ensure the security, performance, and reliability of your MongoDB-based applications.

Training Period

During the training period, my main job is to learn about the course content which is front-end development including the HTML, CSS, JavaScript technologies and to practically implement these concepts. So, to implement it practically, I have to develop a project based on these things, which is my portfolio.

Setting up the things:

- So, my work is that for the creation of project, I have to first think about the project structure and basic needs and necessities of the project like what are the basic things inside a portfolio website, for example: make it work in all devices, navigation bar, search buttons, social media handles, etc.

Ordering the content:

- After thinking about all these basic requirements, I have to figure out the design of the website that is, where should I place navigation bar, social media icons, my contact and all.

Designing phase:

- After that, look of the website is created, which means what color combination will suits the website, design of the buttons and icons, what extra things to be included to make site look realistic. I also had to see how to make it work in mobile devices too.

Folder Structuring:

- Then folder structure is created, which means that proper locations to the files are assigned like CSS files will be placed in different folder, JavaScript files will be placed in the different folder, image files will be placed in different folder. This is done because it is easy and fast to sort things out if proper order of the files and folders is maintained.

3.7 PROJECT STRUCTURE

Briefing about my project, my project is a website which is basically a portfolio which has full-stack covered. My Project title is "PORTFOLIO WEBSITE". As the name suggests it is related to description of my resume. In this, there is the description about me, it tells my skills, it consists of the project I have made and also contains my contact details and a blog page too.

3.8 FILE STRUCTURE

```
- src/
  - app.js
  - node_modules/
  - public/
    - images/
    - css/
      - styles.css
  - views/
    - partials/
      - header.ejs
      - footer.ejs
    - about.ejs
    - addProject.ejs
    - blog.ejs
    - compose_blog.ejs
    - contact.ejs
    - home.ejs
    - portfolio.ejs
    - post.ejs
    - project.ejs
    - thanksforcontacting.ejs
  - .gitignore
  - package-lock.json
  - package.json
```

Fig-15. Hierarchical representation of File structure

In the "src" directory:

- **app.js** - The main JavaScript file that serves as the entry point for my Node.js application.
- **node_modules** - A directory where my project's dependencies and packages are stored.
- **public** - A directory containing publicly accessible assets like images and CSS files.
- **views** - A directory that stores the EJS templates used for rendering different pages of my web application.
- **.gitignore** - A file specifying which files and directories should be excluded from version control.
- **package-lock.json** - A JSON file that locks the versions of dependencies to ensure consistent builds.

- **package.json** - A JSON file containing metadata and dependency information for my Node.js project.

In the "views" directory:

- **about.ejs** - An EJS template for the "About" page of my web application.
- **addProject.ejs** - An EJS template for adding a new project to my portfolio.
- **blog.ejs** - An EJS template for displaying blog posts.
- **compose_blog.ejs** - An EJS template for composing new blog posts.
- **contact.ejs** - An EJS template for the "Contact" page.
- **home.ejs** - An EJS template for the homepage of my web application.
- **portfolio.ejs** - An EJS template for displaying my portfolio projects.
- **post.ejs** - An EJS template for displaying a single blog post.
- **project.ejs** - An EJS template for displaying details about a specific project.
- **thanksforcontacting.ejs** - An EJS template for a thank-you message after submitting a contact form.

In the "partials" directory:

- **header.ejs** - An EJS partial for the header section of my web pages.
- **footer.ejs** - An EJS partial for the footer section of my web pages.

In the "public" directory:

- **images** - A directory containing image assets used in my web application.
- **css** - A directory containing CSS styles for styling my web pages.

In the "css" directory:

- **styles.css** - The main CSS file that defines the styles for my web application.

3.9 WEBSITE STRUCTURE

As a viewer or user of the website, we can divide the website mainly in three sections:

- Header Section
- Content Area
- Footer Section

Briefly discussing each section, will form the image in our mind like this:

Header Section:

- This section is the top section of our website. This section can also be termed to as the navigation bar of the website. This is called as navigation bar because we can easily switch to other pages from one page if I have to add more pages in the website in the future.
- Navigation bar in our project consists of my name in the top left corner of the screen as it is a portfolio. This name depicts the name of the person whose portfolio it is, here it is my name as I am describing my portfolio.
- Just at the right of the navigation bar consists of Home, About, Portfolio, Contact, Blog and finally links to social media.

Content Area:

- The main content section varies for different pages on the website. It contains page-specific content, such as text, images, forms, and other elements. Each page template (e.g. Home, About, Blog, Contact, Portfolio) represents this section.

Footer:

- The footer section usually appears at the bottom of each web page and includes elements like copyright information, links to social media profiles, contact me link, and other relevant information. It provides users with additional resources and information after they've explored the main content.

3.10 WEBSITE DESIGN

I carefully planned how my project should look and work to make it user-friendly and attractive. Here's what I did:

1. **Colors:** I picked colors that go well together and match my project's style.
2. **Text Style:** I chose fonts and how text looks, so it's easy to read and looks the same across the project.
3. **Layout:** I organized everything on the page so it looks neat, whether you're using a computer or a phone.
4. **Menu:** I made a menu at the top so you can easily find different parts of the project.
5. **Pictures:** I used nice pictures and icons to make things look interesting and explain stuff better.
6. **Fits Any Screen:** No matter if you use a computer or a phone, my project will work and look good.
7. **Buttons and Forms:** I made buttons and forms easy to use and understand.

8. **My Logo and Colors:** I used my logo, tagline, and the same colors everywhere so you can remember my project.
9. **Help for Everyone:** I made sure that even if you have trouble seeing or using the website, you can still use it.
10. **Cool Movements:** There are some small movements and things you can click on to make the project fun to use.
11. **Everything Looks the Same:** I kept everything looking the same, so it looks nice and professional.
12. **Easy to Use:** My main goal was to make everything easy to find and use, so you have a good time using my project.

By personally executing this comprehensive design strategy, I've aimed to deliver a web application that not only communicates its purpose effectively but also engages and delights its target audience.

APIs and Libraries in My Project:

1. **Twilio API (for Sending Messages):**
   I've integrated Twilio into my project to send messages to my phone when someone fills out the contact form. This way, I can stay updated and respond quickly to user inquiries.

2. **Quill Editor (for Attractive Blog Posts):**
   To make writing blog posts more appealing, I've incorporated the Quill Editor. It allows me to create blog content with attractive formatting and styling, making the posts visually engaging.

3. **Font Awesome Library (for Icons):**
   I've included the Font Awesome library to easily add icons to my project. Icons help convey information visually and enhance the overall design, making it more user-friendly.

4. **Mongoose (for Connecting to MongoDB Server Online):**
   I've used Mongoose to connect my project to an online MongoDB server. This enables me to store and manage data like blog posts and user information securely and efficiently.

5. **JavaScript's AOS (for Animations):**
   With JavaScript's AOS (Animate On Scroll) library, I've added animations to my project. This brings elements to life as users scroll down the page, creating a dynamic and engaging experience.

Used HTML Tags:

1. **HTML TAG:** Defines the root of the HTML document, serving as the container for all other elements.
2. **HEAD TAG:** Contains metadata about the document, such as character set, page description, and keywords, and is located between the **<html>** and **<body>** tags.
3. **LINK TAG:** Establishes relationships between the current document and external resources, often used to link to external style sheets (e.g., **style.css** in the project).
4. **META TAG:** Defines metadata about the HTML document, placed within the **<head>** element, and typically used for specifying character sets, page descriptions, and viewport settings.
5. **TITLE TAG:** Sets the title of the document, displayed in the browser's title bar or tab, crucial for search engine optimization (SEO).
6. **SCRIPT TAG:** Embeds client-side scripts (JavaScript) within the document, either containing scripting statements or pointing to external script files.
7. **BODY TAG:** Defines the document's body and contains all the visible content of an HTML document.
8. **SECTION TAG:** Defines a section within the document, aiding in structuring content; often used to separate different sections like About Me, Skills, and Projects.
9. **DIV TAG:** Defines a division or section, serving as a container for HTML elements and customizable through CSS or JavaScript.
10. **A TAG:** Creates hyperlinks for navigation, with the **href** attribute indicating the link's destination; commonly used in navigation menus and linking to external pages or profiles.
11. **<h1> to <h6> Tags:** Define headings with varying levels of importance, with **<h1>** being the most important and **<h6>** the least; used to structure content hierarchically.
12. **SPAN TAG:** An inline container used for marking up and styling specific parts of text or documents.
13. **P TAG:** Defines paragraphs, organizing text content; commonly used for descriptions and explanations.
14. **UL TAG:** Creates unordered (bulleted) lists, often combined with **<li>** (list item) tags to form lists.
15. **LI TAG:** Defines list items within ordered or unordered lists; used to list items within menus and content.
16. **IMG TAG:** Embeds images in an HTML page, requiring attributes such as **src** (image path) and **alt** (alternate text for image).

Used CSS Units:

Absolute Length:

Absolute length units are not recommended for use on screen, because screen sizes vary so much. However, they can be used if the output medium is known, such as for print layout.

Px - Pixels (px) are relative to the viewing device. For low-dpi devices, 1px is one device pixel (dot) of the display. For printers and high resolution screens 1px implies multiple device pixels.

Relative Length:

Relative length units specify a length relative to another length property. Relative length units scale better between different rendering medium.

UNITS	DESCRIPTION
REM	Relative to font-size of the root element
%	Relative to the parent element
VW	Relative to 1% of the width of the viewport
VH	Relative to 1% of the height of the viewport

Table-5. Relative Lengths in CSS

MEDIA QUERY IN WEBSITE:

The @media rule is used in media queries to apply different styles for different media types/devices.

Media queries can be used to check many things, such as:

- width and height of the viewport
- width and height of the device
- orientation (is the tablet/phone in landscape or portrait mode?)

Key frames:

- The @keyframes rule specifies the animation code.
- The animation is created by gradually changing from one set of CSS styles to another.
- During the animation, you can change the set of CSS styles many times.
- There are many times where I have used @keyframes in Portfolio, for instance the hamburger sign which appears in mobile size has @keyframe used in it Because of the @keyframes it is glowing and making it look realistic.
- Secondly it has been used in the section just below navigation menu ie. hero section. Because of it, it is showing a box transition over the greeting and changing the color of my name.

Chapter-4: SNAPSHOTS

app.js code:

```
const express = require("express");
const bodyParser = require("body-parser");
const ejs = require("ejs");
const mongoose = require('mongoose');
const twilio = require('twilio');

const app = express();

app.use(express.static('public'));
app.use(bodyParser.urlencoded({ limit: '10mb', extended: true }));
app.use(bodyParser.json({ limit: '10mb' }));

app.use((req, res, next) => {
  const nonce = generateNonce(); // Generate a random nonce value

  res.setHeader(
    'Content-Security-Policy',
    `default-src 'self'; style-src 'self' 'unsafe-inline' https://cdn.quilljs.com https://cdnjs.cloudflare.com https://unpkg.com https://fonts.googleapis.com; script-src 'self' https://cdn.quilljs.com
    'nonce-${nonce}'; img-src 'self' data:; font-src 'self' https://cdnjs.cloudflare.com;`
  );

  res.locals.nonce = nonce; // Make nonce accessible in templates if needed

  next();
});

function generateNonce() {
  const chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
  let nonce = '';
  for (let i = 0; i < 16; i++) {
    nonce += chars.charAt(Math.floor(Math.random() * chars.length));
  }
  return nonce;
}


app.set('view engine', 'ejs');

// Routes for different pages
app.get("/", function(req, res){
    res.render("home");
});

app.get("/about", function(req, res){
    res.render("about");
});

app.get("/contact", function(req, res){
    res.render("contact");
});

app.get("/thanksforcontacting", function(req, res){
    res.render("thanksforcontacting");
});

// Portfolio logic
const db2 = mongoose.createConnection("mongodb+srv://Ankush7163:ankushsingh490@cluster0.f50reqx.mongodb.net/myProjectDB", {useNewUrlParser: true, useUnifiedTopology: true});

const projectSchema = new mongoose.Schema({
    project_title: String,
    projectAim: String,
    desc: String,
    tech_used: [String],
    projectImageURL: String,
    projectDate : String,
    challengesNlearnings: String,
    keyFeatures: [String],
    projectConclusion: String,
    projectLink: String
});

const Project = db2.model('Project', projectSchema);

app.get("/addProject", function(req, res){
    res.render("addProject");
});

app.get("/portfolio", function(req, res){
  Project.find({}, function(err, projects){
    res.render("portfolio", {
      projects: projects
      });
  });
});

app.get("/portfolio/projects/:projectId", function(req, res){

    const requestedProjectId = req.params.projectId;

      Project.findOne({_id: requestedProjectId}, function(err, project){
        res.render("project", {
          project_title: project.project_title,
          projectAim: project.projectAim,
          desc: project.desc,
          tech_used: project.tech_used,
          projectImageURL: project.projectImageURL,
          projectDate: project.projectDate,
          keyFeatures: project.keyFeatures,
          challengesNlearnings: project.challengesNlearnings,
          projectConclusion: project.projectConclusion,
          projectLink: project.projectLink
        });
      });

});
```

```
// Post requests
app.post("/compose", function(req, res){
    const post = new Post({
        title: req.body.postTitle,
        content: req.body.postBody,
        currentDate: req.body.postDate
      });


      post.save(function(err){
        if (!err){
            res.redirect("/blog");
        }
      });
})

app.listen(3000, function(){
    console.log("Server is started at port 3000..");
});
```

home.ejs code:

```
<%- include("partials/header"); -%>

<section class="introContainer">
  <div class="introProfileContainer">
    <img src="/images/Introduction.png" alt="" class="introImage" />
    <div class="typing">
      <p>
        Hey there, I'm Ankush Singh, a tech enthusiast on an exciting journey through the world of computer science. Currently, I'm deep into my third year of B.Tech at MAIT, Delhi.
        When I'm not busy coding and exploring full-stack web development, you'll find me passionately supporting open source initiatives. But hey, life isn't all about work! In my
        free time, I absolutely adore playing chess and satisfying my wanderlust through adventurous travels.<br>So, let's connect and embark on this thrilling tech adventure together!
        🚀🤩
      </p>
    </div>
  </div>
  <img src="/images/profilepic2.png" class="profileImage" alt="" srcset="" />
</section>

<div class="skillsHome">
  <div>
    <img src="/images/tt.jpg" class="introImage" alt="" />
  </div>
  <div class="logos">
    <div class="logos-slide">
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S1.png" />
        <img src="/images/skillLogos/S2.png" />
      </div>
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S3.png" />
        <img src="/images/skillLogos/S4.png" />
      </div>
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S5.png" />
        <img src="/images/skillLogos/S6.png" />
      </div>
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S7.png" />
        <img src="/images/skillLogos/S8.png" />
      </div>
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S9.png" />
        <img src="/images/skillLogos/S10.png" />
      </div>
      <div class="skillImageContainer">
        <img src="/images/skillLogos/S11.png" />
```

```
<div class="servicesContainer">
  <img src="/images/services.jpg" alt="" srcset="" />
  <div class="serviceItem" data-aos="fade-right" data-aos-duration="500">
    <h3>1. Responsive Website Development</h3>
    <p>
      Creating websites that are optimized for different devices and screen
      sizes, ensuring a seamless user experience across desktop, tablet, and
      mobile.
    </p>
  </div>
  <div class="serviceItem" data-aos="fade-left" data-aos-duration="500">
    <h3>2. Custom Web Application Development</h3>
    <p>
      Building tailored web applications to meet specific business requirements,
      such as inventory management systems, CRM platforms, or project management
      tools.
    </p>
  </div>
```

```
  <div class="ctaContainer">
    <img src="/images/lbsa.svg" alt="">
    <h2>You're just one step away from unlocking the extraordinary.</h2>
    <h2>
      Contact me now to reveal the cutting-edge web solutions that will propel
      your business to new horizons.
    </h2>
    <button>
      <a href="/contact" target="_blank" class="flip">
        <div class="inner">
          <div class="front">Hover me!</div>
          <div class="back">Contact now!</div>
        </div>
      </a>
    </button>
  </div>

  <script nonce="<%= nonce %>" src="https://unpkg.com/aos@next/dist/aos.js"></script>
  <script nonce="<%= nonce %>">
    AOS.init();
  </script>

  <%- include("partials/footer"); -%>
```

styles.css code:

```css
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  overscroll-behavior: smooth;
}

/* Preloader code */
#preloader {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background-color: #f2f2f2;
  display: flex;
  align-items: center;
  justify-content: center;
  z-index: 9;
}

.loader {
  width: 4rem;
  height: 4rem;
  border-radius: 50%;
  border: 8px solid #00adb5;
  animation: pulsate 4s linear infinite;
}
```

```css
nav {
  display: flex;
  align-items: center;
  flex-direction: column;
}

nav ul {
  list-style: none;
  display: flex;
  margin: 0;
  padding: 0;
}

nav li {
  margin: 0 10px;
}

nav a {
  color: #fff;
  text-decoration: none !important;
  display: inline-block;
  transition: transform 0.3s;
}

nav a:hover {
  color: #7effff;
  transform: scale(1.1);
}
```

```css
/* Footer Styling starts here.. */
footer {
  position: relative;
  bottom: 0;
  right: 0;
  left: 0;
  background-color: #333;
  color: #fff;
  padding: 20px 0;
}

footer .container {
  max-width: 960px;
  margin: 0 auto;
  text-align: center;
}

footer .container p {
  margin: 0;
}

footer ul {
  list-style-type: none;
  padding: 0;
  margin: 10px 0;
}

footer ul li {
  display: inline-block;
  margin-right: 10px;
}

footer ul li a {
  color: #fff;
  text-decoration: none;
}

footer ul li a:hover {
  text-decoration: underline;
}
```

Chapter-5: RESULTS & DISCUSSION

Introduction:

This report presents the culmination of my summer training after completing my second year of undergraduate studies. The training was undertaken through the renowned online E-learning platform, Udemy, focusing on full-stack web development. The key highlight of this training journey is the successful completion of the course titled "The Complete 2023 Web Development Bootcamp" and the accompanying project, the "Portfolio Website."

Course Overview:

The course, led by the highly respected instructor, Angela Yu, provided an immersive learning experience covering a wide spectrum of technologies and frameworks essential for full-stack web development. These technologies included HTML, CSS, JavaScript, Express, Node.js, and MongoDB. Additionally, the course introduced invaluable web development tools such as Chrome extensions and developer tools. Practical application was emphasized through the completion of small-scale projects, which proved instrumental in reinforcing our newfound knowledge.

Project Scope:

The focal point of this report is the "Portfolio Website" project. The project's primary focus was on front-end development, but it also incorporated dynamic features for uploading new projects and blogs, making it a well-rounded representation of my web development skills.

Technologies Utilized:

The technologies employed in the project closely aligned with the ones covered in the course:

1. **HTML:** The foundation of the project, HTML, was used to structure the website's content and define its layout.
2. **CSS:** Cascading Style Sheets (CSS) were employed to style and format the website, ensuring a visually appealing and responsive design.
3. **JavaScript:** JavaScript added interactivity to the website, allowing for dynamic features and user-friendly navigation.
4. **Express and Node.js:** These technologies facilitated the back-end of the project, enabling server-side operations and data management.
5. **MongoDB:** MongoDB served as the project's database, storing and retrieving data seamlessly.

Project Structure:

The website's fundamental structure consisted of three main components:

1. **Header:** Serving as the website's navigation bar, the Header included essential sections such as "Home" and "Menu." This streamlined user navigation, providing a one-click solution for users to access their desired pages.
2. **Content Area:** The central space of the website, the Content Area, was the showcase for my work and content across various pages. It allowed for the organized presentation of projects, blogs, and other relevant information.
3. **Footer:** The Footer provided a final touch to the website, offering contact information and links to social media profiles, enhancing user engagement.

Testing and Development:

Throughout the development phase, Google Chrome served as the primary testing platform, ensuring the website's compatibility and functionality across different devices and browsers. Rigorous testing was conducted to identify and address any issues, resulting in a seamless user experience.

Training Report:

The training report encompasses:

1. **Course Details:** A thorough breakdown of the course, including its content, duration, and objectives.
2. **Instructor's Impact:** Reflections on the guidance and mentorship provided by Angela Yu, whose expertise and teaching style played a pivotal role in my learning journey.
3. **Technical Insights:** An in-depth exploration of the project's technical aspects, including the implementation of various technologies, the project's file structure, and website layout.
4. **Visual Documentation:** Screenshots of the final result, showcasing the website's design, functionality, and responsiveness.
5. **Personal Reflections:** A section dedicated to personal growth and insights gained throughout the training, highlighting achievements and areas for improvement.
6. **Self-Evaluation:** A holistic assessment of my performance and accomplishments during the training period, providing valuable feedback for future endeavors.

In conclusion, the "Portfolio Website" project, a part of the "The Complete 2023 Web Development Bootcamp," marks a significant milestone in my journey toward becoming a skilled full-stack web developer. This experience has provided crucial technical skills while nurturing my personal & professional growth. It underscores my dedication to excel in web development.

Chapter-6: CONCLUSION & FUTURE SCOPE

CONCLUSION:

In retrospect, embarking on this Full-Stack Web Development journey has been both transformative and gratifying. It is undeniable that I've acquired a wealth of knowledge and skills throughout this course, and I'm excited to reflect on my progress.

Given my initial lack of familiarity with HTML, CSS, JavaScript, Express, Node.js, and MongoDB, I can confidently say that the time invested in researching and mastering these languages and technologies has been exceptionally rewarding. This newfound expertise has allowed me to devise effective solutions to critical aspects of web design and development.

Beyond the technical proficiency I've gained, this experience has cultivated essential soft skills such as time management, self-motivation, and a strong work ethic. These attributes have proven invaluable throughout the course, reinforcing my commitment to success in the field of full-stack web development.

Working extensively with HTML, CSS, JavaScript, Express, Node.js, and MongoDB has not only deepened my interest in these technologies but has also honed my creative aptitude. These skills are not only personally fulfilling but also hold significant potential for future endeavors.

Building a Portfolio website using HTML, CSS, JavaScript and integrating backend has been a pivotal experience, significantly boosting my confidence and knowledge in web development. In the process of creating websites, I've uncovered a multitude of possibilities these skills offer. Freelancing, for example, presents an opportunity to gain practical experience while generating income—a compelling prospect for future ventures.

Link to the project: portfolio-website-1xm3.onrender.com

FUTURE SCOPE:

In the digital age, a website stands as a paramount asset for any business seeking to engage with online audiences. A website serves as the virtual storefront, reflecting a company's identity and offerings. With this realization, the demand for proficient web developers and designers continues to soar, promising an expansive scope for those skilled in MERN Stack.

The ever-evolving landscape of web development underscores the need for web developers and designers who can craft innovative and user-friendly platforms. In response to this demand, professionals are harnessing their technical expertise to create unique and compelling online experiences.

Advancements in web development tools, techniques, technologies, and frameworks have streamlined the website creation process. Agile methodologies have reduced development time, allowing businesses to establish their online presence swiftly and efficiently.

This ongoing transformation underscores the promising future of web development.

While JavaScript initially emerged as a browser-centric language, it has evolved to find applications in diverse environments. Today, JavaScript stands as the most widely adopted browser language, seamlessly integrated with HTML and CSS to create dynamic web experiences.

The integration of JavaScript in various domains, coupled with its adaptability and versatility, positions it as a cornerstone of modern web development. Its enduring relevance underscores the continued growth and innovation within the field.

Chapter-7: WEEKLY JOB SUMMARY

Week 1:

Description of Activity, duty responsibility	Performed with team	Performed alone	Time spent
Studying about e-learning Platform and getting used to it	No	Yes	2 hrs.
Getting started with Front-end Web Development	No	Yes	2 hrs.
HTML, Anatomy of HTML, History of HTML	No	Yes	2hrs.
Structure Of HTML, HTML content, Elements	No	Yes	2hrs.
Lists, Images, Creating Links, HTML Tags	No	Yes	3hrs.

One thing that went particularly well this week:

I successfully started my journey into front-end web development and gained a good understanding of HTML basics.

The most challenging thing this week:

Learning about the structure of HTML and grasping its various elements was a bit challenging, but I made progress.

Self-evaluation:

A^+ (A) A^- B^+ B B^- C^+ C C^- D^+ D D^- F

List one way you can improve your performance:

To improve, I should continue practicing HTML and start applying what I've learned to build web pages.

Week 2:

Description of Activity, duty responsibility	Performed with team	Performed alone	Time spent
Continuing Front-end Development	No	Yes	2 hrs.
CSS Basics and Styling HTML Elements	No	Yes	2 hrs.
Introduction to JavaScript	No	Yes	2hrs.
Exploring HTML Forms	No	Yes	2hrs.
Adding CSS Styles to a Webpage	No	Yes	3hrs.

One thing that went particularly well this week:

I successfully delved into CSS styling, making my web pages visually appealing.

The most challenging thing this week:

Understanding JavaScript concepts and how to use it for interactivity was a bit challenging, but I'm making progress.

Self-evaluation:

A^+ (**A**) **A^-** **B^+** **B** **B^-** **C^+** **C** **C^-** **D^+** **D** **D^-** **F**

List one way you can improve your performance:

I should practice JavaScript regularly and apply it to create dynamic elements on my website.

Week 3:

Description of Activity, duty responsibility	Performed with team	Performed alone	Time spent
Advancing JavaScript Skills	No	Yes	2 hrs.
Introduction to Node.js	No	Yes	2 hrs.
Learning Express.js	No	Yes	2hrs.
Setting Up a MongoDB Database	No	Yes	2hrs.
Planning Website Structure	No	Yes	3hrs.

One thing that went particularly well this week:

I made significant progress in my JavaScript skills and started exploring server-side development with Node.js and Express.js.

The most challenging thing this week:

Setting up and understanding MongoDB for database management was a challenge, but I'm working on it.

Self-evaluation:

A^+ (**A**) **A^-** **B^+** **B** **B^-** **C^+** **C** **C^-** **D^+** **D** **D^-** **F**

List one way you can improve your performance:

Continue practicing with Node.js and Express.js and explore more advanced concepts.

Week 4:

Description of Activity, duty responsibility	Performed with team	Performed alone	Time spent
Building Website Pages (Home, About)	No	Yes	4 hrs.
Implementing Animation with AOS Library	No	Yes	1 hrs.
Adding Service and Skills Section	No	Yes	3 hrs.
Styling and Layout Improvements	No	Yes	3 hrs.
Integrating Font Awesome Icons	No	Yes	1 hrs.

One thing that went particularly well this week:

I successfully built the Home and About pages, added animations using AOS library, and integrated Font Awesome icons for a polished look.

The most challenging thing this week:

Achieving the desired layout and styling was challenging, but I learned valuable CSS techniques to overcome these hurdles.

Self-evaluation:

A^+ (**A**) **A^-** **B^+** **B** **B^-** **C^+** **C** **C^-** **D^+** **D** **D^-** **F**

List one way you can improve your performance:

One way to improve performance is to continue refining my CSS skills and explore more advanced animation techniques.

Week 5:

Description of Activity, duty responsibility	Performed with team	Performed alone	Time spent
Creating Portfolio and Blog Pages	No	Yes	4 hrs.
Implementing Project Slider	No	Yes	3 hrs.
Building CMS for Dynamic Project and Blog Posts	No	Yes	4 hrs.
Integrating Twilio API for Contact	No	Yes	3 hrs.
Incorporating Quill Editor for Blog Posts	No	Yes	3 hrs.

One thing that went particularly well this week:

I successfully created the Portfolio and Blog pages, implemented dynamic project sliders and a CMS for content management, and integrated the Twilio API for contact.

The most challenging thing this week:

Building a dynamic CMS for blog posts using Quill Editor was challenging, but it's a valuable addition to my website.

Self-evaluation:

A^+ (A) A^- B^+ B B^- C^+ C C^- D^+ D D^- F

List one way you can improve your performance:

Continue working on content management systems and explore additional APIs for enhancing user interaction.

Chapter-8: LEARNINGS AFTER TRAINING

Completing the full-stack web development course and crafting the portfolio website project has been an enriching learning experience, offering insights into various aspects of web development. As a student, I've gained valuable knowledge and skills:

1. **Comprehensive Technical Proficiency:**
   The full-stack web development course served as a comprehensive immersion into a diverse array of web technologies. From the very fundamentals of HTML, CSS, and JavaScript to the intricacies of server-side development with Express and Node.js, and the intricacies of database management using MongoDB, I gained proficiency in both front-end and back-end development. These technical skills formed the core of my learning journey.

2. **Responsive Design Mastery:**
   Crafting a responsive web design emerged as a fundamental skill during the course. I learned the art of creating websites that seamlessly adapt to various screen sizes and devices. This aspect of web development is crucial in delivering a satisfying user experience across a wide range of platforms.

3. **Dynamic User Experiences with JavaScript:**
   JavaScript played a pivotal role in enhancing the interactivity of websites. I learned to create dynamic features, interactive forms, and responsive user interfaces. This newfound skillset allowed me to build web applications that respond intuitively to user actions.

4. **Server-Side Development Expertise:**
   The course delved deep into server-side development using Express and Node.js. This facet of web development empowered me to design and implement robust server logic, enabling data management, routing, and server-side operations.

5. **Effective Database Management with MongoDB:**
   Understanding and utilizing MongoDB for database management was a significant milestone. I learned how to store, retrieve, and manipulate data within web applications, crucial for creating dynamic and data-driven websites.

6. **Project Planning and Organization:**
   The portfolio website project demanded meticulous planning and organization. I honed my project management skills by breaking down complex tasks into manageable steps, establishing timelines, and ensuring seamless coordination between front-end and back-end development.

7. **Thorough Testing and Debugging:**
   Rigorous testing and debugging became second nature throughout the development

process. I realized the importance of identifying and rectifying issues promptly to ensure the reliability and functionality of the website.

8. **User-Centric Design and UX Principles:**
   I discovered the significance of user-centric design principles. Focusing on creating intuitive and engaging user experiences became a fundamental aspect of my work. User interface (UI) and user experience (UX) considerations were central in my design and development decisions.

9. **Version Control and Collaboration:**
   Implementing version control systems, particularly Git, transformed the way I collaborated on projects. It allowed for efficient tracking of changes, seamless collaboration with team members, and the ability to revert to previous versions when needed.

10. **Problem-Solving and Critical Thinking:**
    Web development frequently posed challenges, ranging from technical hurdles to design decisions. The experience honed my problem-solving skills, encouraging me to think critically and creatively to find effective solutions.

11. **Thorough Documentation Practices:**
    Maintaining clear and organized documentation became an essential part of my workflow. It ensured that I could keep track of project progress, collaborate effectively with others, and troubleshoot issues efficiently.

12. **Self-Reflection and Continuous Improvement:**
    As a student, I embraced self-reflection as a means of assessing my progress and identifying areas for improvement. This reflective approach allowed me to adapt my learning strategies and refine my skills continually.

13. **Personal and Professional Growth:**
    Beyond technical competencies, this learning journey fostered personal and professional growth. It emphasized the importance of adaptability, resilience, and the capacity to embrace new technologies and methodologies in the ever-evolving field of web development.

Chapter-9: REFRENCES

Books referred for Web Development:

- **"Introducing HTML5" by Remy and Bruce**
- **"CSS Mastery 3rd Edition" by Andy Budd & Emil Björklund**
- **"JavaScript: The Good Parts" by Douglas Crockford**
- **"The Full Stack Developer: Your Essential Guide to the Everyday Skills Expected of a Modern Full Stack Web Developer 1st Edition" by Chris Northwood**

Web Links:

For MERN stack development, the links I referred the most are:

- www.geeksforgeeks.org
- www.w3schools.com
- www.javatpoint.com
- www.w3docs.com/