Front-End Development (HTML, CSS, JavaScript)

**HTML:**

* Structure: Understanding the basic structure of an HTML document with elements such as **<html>**, **<head>**, and **<body>**.
* Tags: Familiarity with commonly used HTML tags like headings (**<h1>** to **<h6>**), paragraphs (**<p>**), lists (**<ul>**, **<ol>**, **<li>**), links (**<a>**), images (**<img>**), and more.
* Forms: Creating interactive forms using **<form>** and form elements like input fields (**<input>**), checkboxes (**<input type="checkbox">**), radio buttons (**<input type="radio">**), and buttons (**<button>**).
* Semantic HTML: Understanding the importance of using semantic HTML tags (**<header>**, **<nav>**, **<section>**, **<article>**, **<footer>**, etc.) for better accessibility and SEO.
* Multimedia: Exploring multimedia elements like audio (**<audio>**) and video (**<video>**) tags to embed media content in web pages.

**CSS:**

* Selectors: Mastery of CSS selectors, including element selectors, class selectors (**.class**), ID selectors (**#id**), attribute selectors, and pseudo-classes (**:hover**, **:nth-child()**, etc.).
* Box Model: Understanding the box model concept, including elements' content, padding, border, and margin properties.
* Layout: Familiarity with various CSS layout techniques, including positioning (**position: absolute**, **position: relative**, **position: fixed**), floats (**float: left**, **float: right**), and flexible box layout (**display: flex**).
* Styling: Application of styles to HTML elements using properties like color, background, font, text, borders, margins, padding, and more.
* Responsive Design: Exploring responsive web design principles, including media queries (**@media**) to create adaptable layouts for different devices.
* CSS Grid System: Utilizing the CSS Grid system for creating highly flexible and responsive layouts with rows and columns.

**Bootstrap:**

* Grid System: Understanding and utilizing Bootstrap's grid system for creating responsive layouts.
* Pre-styled Components: Utilizing Bootstrap's extensive library of pre-styled components, such as navigation bars, buttons, cards, modals, and more.
* Responsive Utilities: Using Bootstrap's responsive utility classes to easily adapt content to different screen sizes.
* Customization: Understanding how to customize Bootstrap's default styles and components to match specific design requirements.

**Tailwind CSS:**

* Utility Classes: Working with Tailwind CSS's utility-first approach, using a wide range of pre-defined utility classes to style elements.
* Responsive Design: Utilizing Tailwind CSS's responsive utility classes to create responsive designs without writing custom media queries.
* Customization: Understanding how to customize Tailwind CSS's configuration file to customize the default utility classes or add new ones.

**JavaScript:**

* Variables and Data Types: Understanding JavaScript variables and different data types such as strings, numbers, booleans, arrays, objects, and null/undefined.
* Functions: Defining and using functions to encapsulate reusable blocks of code.
* Conditional Statements: Mastering if/else statements, switch statements, and ternary operators for decision-making in JavaScript.
* Loops: Working with different loop structures like **for**, **while**, and **do...while** to execute code repeatedly.
* DOM Manipulation: Manipulating HTML elements dynamically using the Document Object Model (DOM), including accessing elements, modifying content, adding/removing elements, and handling events.

**ES6:**

* Arrow Functions: Utilizing arrow functions for concise and lexical scoping of functions.
* Template Literals: Using template literals to create more readable and flexible strings with embedded expressions.
* Destructuring Assignments: Employing destructuring assignments to extract values from arrays or objects more easily.
* Spread Syntax: Leveraging the spread syntax to combine arrays, clone objects, or pass multiple arguments to functions.
* Modules: Working with ES6 modules to organize and import/export code across multiple JavaScript files.

**React:**

* Components: Understanding the concept of React components and their lifecycle, including creating functional components.
* JSX: Writing JSX (JavaScript XML) syntax to define the structure and appearance of React components.
* State and Props: Managing component state using the **useState,useEffect** hook, passing data through props between components.

These headings provide a breakdown of the key concepts and skills I have learned in HTML, CSS, Bootstrap, Tailwind CSS, JavaScript, ES6, and React during my fellowship at Bytewise Limited. Continually building upon and expanding my knowledge in these areas will help me grow as a front-end developer.