# ASSIGNMENT 1

## 1. Explain the difference between frontend, backend, and full-stack development with suitable real-world examples.

Frontend Development: It refers to the part of a website or web application that users interact with directly. It involves the design, structure, and behavior of web pages using technologies like HTML, CSS, and JavaScript.   
Example: When you visit Amazon, the product listings, buttons, and menus you see are all part of the frontend.  
  
Backend Development: It deals with the server-side of an application, including data storage, server logic, and authentication. Backend developers use languages such as Java, Python, Node.js, and PHP.  
Example: When you log in to Facebook, the backend verifies your credentials and fetches your profile data from the database.  
  
Full-Stack Development: A full-stack developer works on both frontend and backend parts of a web application.  
Example: A full-stack developer creating a blog website designs the user interface and also writes server-side logic to store posts in the database.

## 2. Create a simple diagram showing how the client-server model works in web architecture.

Diagram (Text Representation):  
Client (Browser) → sends HTTP Request → Web Server → processes the request → Database (if needed) → Web Server → sends HTTP Response → Client (Browser displays content)

## 3. Describe how a browser requests and displays a web page from a web server.

1. The user enters a URL into the browser.  
2. The browser sends an HTTP request to the web server hosting the website.  
3. The web server receives the request, processes it, and returns an HTTP response containing HTML, CSS, and JavaScript files.  
4. The browser interprets the HTML, applies CSS for styling, and executes JavaScript for interactivity.  
5. The final rendered web page is displayed to the user.

## 4. Identify and list the tools required to set up a web development environment. Explain the purpose of each.

1. Code Editor (VS Code): Used to write and edit code efficiently.  
2. Web Browser (Chrome, Firefox): For testing and debugging websites.  
3. Version Control (Git): Tracks changes in the code.  
4. Package Manager (npm): Manages libraries and dependencies.  
5. Web Server (XAMPP, Node.js): Hosts and runs web applications locally.

## 5. Explain what a web server is and give examples of commonly used servers.

A web server is a software or hardware system that delivers web pages to clients (browsers) upon request via HTTP or HTTPS protocols.  
Examples: Apache HTTP Server, Nginx, Microsoft IIS, and LiteSpeed.

## 6. Define the roles of a frontend developer, backend developer, and database administrator in a project.

Frontend Developer: Designs and implements the user interface and ensures responsiveness and accessibility.  
Backend Developer: Handles server logic, database communication, and authentication processes.  
Database Administrator (DBA): Manages the database, ensuring security, backup, and performance optimization.

## 7. Install VS Code and configure it for HTML, CSS, and JavaScript development. Take a screenshot of the setup.

Steps:  
1. Download and install Visual Studio Code from the official website.  
2. Install necessary extensions: “Live Server”, “Prettier”, “HTML CSS Support”, and “JavaScript (ES6) snippets”.  
3. Create a project folder with index.html, style.css, and script.js files.  
4. Open using VS Code and launch using the Live Server extension.  
(Attach Screenshot Here)

## 8. Explain the difference between static and dynamic websites. Provide an example of each.

Static Websites: Display fixed content that doesn’t change unless manually updated by the developer. Example: A personal portfolio website.  
Dynamic Websites: Display content that changes based on user interaction or database queries. Example: Facebook or an e-commerce site.

## 9. Research and list five web browsers. Explain how rendering engines differ between them.

Five Common Web Browsers:  
1. Google Chrome – Rendering Engine: Blink  
2. Mozilla Firefox – Rendering Engine: Gecko  
3. Microsoft Edge – Rendering Engine: Blink (previously EdgeHTML)  
4. Safari – Rendering Engine: WebKit  
5. Opera – Rendering Engine: Blink  
  
Rendering engines differ in how they parse HTML, apply CSS styles, execute JavaScript, and display web pages. This may cause slight variations in how websites appear across browsers.

## 10. Draw a labeled diagram showing the basic web architecture flow — client, server, database, and APIs.

Diagram (Text Representation):  
Client (Browser) ↔ Web Server ↔ Application Server ↔ Database  
 ↑  
 │  
 APIs  
This flow shows how requests from the client are processed by the web server, interact with APIs or the database, and return responses back to the client.