ASSIGNMENT – 1

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

Frontend Development:

- Deals with the visual part of a website or application that users interact with.
- Involves technologies like HTML, CSS, JavaScript, and frameworks like React or Angular.
- Ensures responsive design, animations, and user experience (UX). - **Example:** The Netflix homepage layout and video thumbnails.

Backend Development:

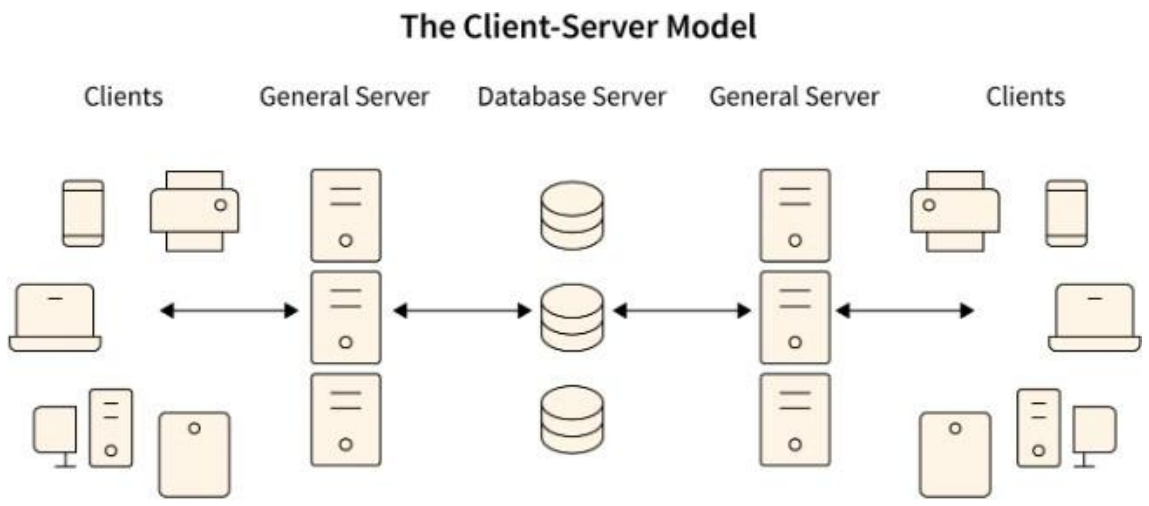
- Handles logic, database connections, and server communication.
- Uses languages like Node.js, Python, Java, or PHP.
- Manages data storage, authentication, and API creation.
- **Example:** When you log in to Netflix, the backend verifies your credentials and fetches your watchlist.

Full-Stack Development:

- Combines both frontend and backend development skills.
- Full-stack developers can design interfaces and build the entire application logic and APIs.- **Example:** A full-stack developer builds both the Netflix login page and the logic that fetches video data.

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

Client-Server Interaction Flow:



1. User opens a browser (client).
2. The client sends an HTTP/HTTPS request to the server.
3. The server processes the request and retrieves data from the database.
4. The server sends a response (HTML/CSS/JS) back to the client.
5. The client displays the data as a web page.

Example: When you search on Google, your query is sent to Google's server, which returns the results.

3. Describe how a browser requests and displays a web page from a web server.
1. User enters a URL or clicks a link.
2. The browser performs a DNS lookup to find the server's IP address.
3. The browser sends an HTTP request to the web server.
4. The server responds with the web page files (HTML, CSS, JavaScript).
5. The browser's rendering engine parses HTML, applies CSS, and executes JS.
6. The page is rendered on the user's screen.

Example: Typing "www.wikipedia.org" sends a request to Wikipedia's server, which returns the homepage.

4. Identify and list the tools required to set up a web development environment. Explain the purpose of each.
1. **VS Code:** A lightweight code editor for writing HTML, CSS, and JS code.
2. **Web Browser (Chrome/Firefox):** To view and test websites.
3. **Node.js:** Allows running JavaScript on the server and managing packages via npm.
4. **Git & GitHub:** Version control and online code repository for collaboration.
5. **Live Server Extension:** Runs a local development server with live reload feature.
6. **Prettier/ESLint:** Tools for code formatting and maintaining code quality.

5. Explain what a web server is and give examples of commonly used servers. -
A web server stores, processes, and delivers web content to clients upon request via HTTP/HTTPS protocols.
- It handles incoming requests and serves the required web pages or APIs.

Examples of popular web servers:
1. Apache HTTP Server
2. Nginx
3. Microsoft Internet Information Services (IIS)
4. LiteSpeed
5. Node.js Express Server

Example: When you visit YouTube, Google's Nginx servers handle your video requests.

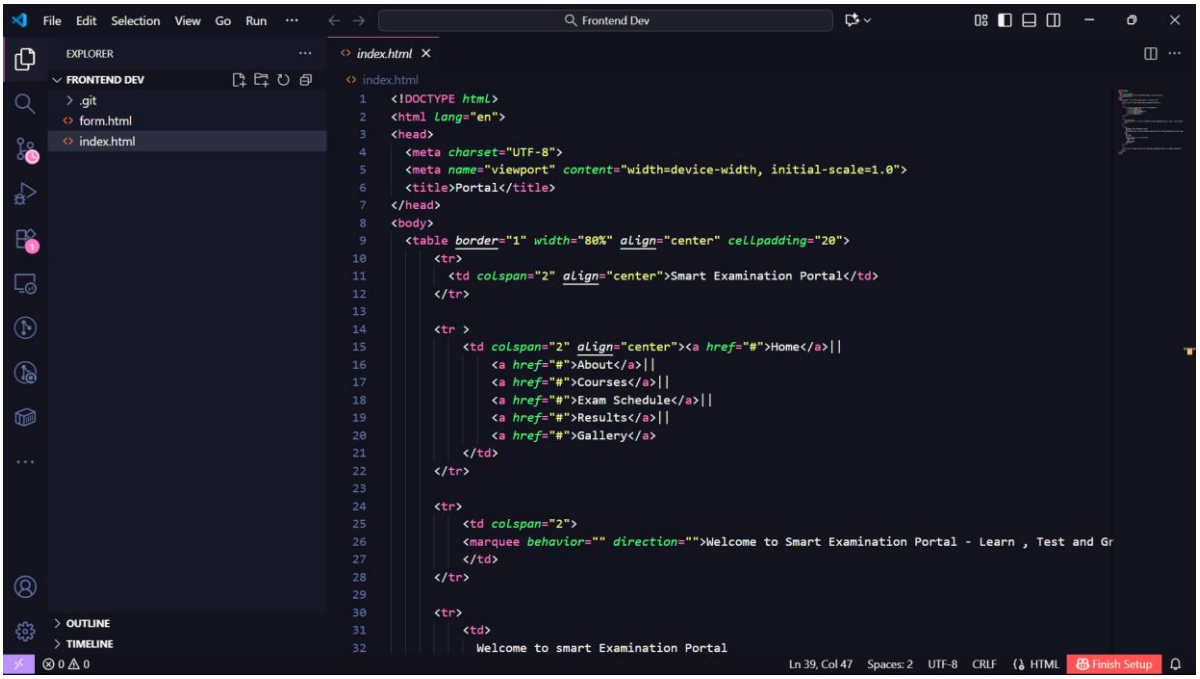
6. **Define the roles of a frontend developer, backend developer, and database administratorin a project.**

Frontend Developer:
- Designs the user interface using HTML, CSS, and JavaScript.
- Ensures responsive and accessible design.
- Implements interactivity and integrates APIs.

Backend Developer:
- Builds the server-side logic and APIs.
- Manages authentication, routing, and security- Connects and queries databases.

Database Administrator (DBA):
- Manages, organizes, and secures the database.
- Ensures backup, performance tuning, and data integrity.
- Works closely with backend developers to optimize queries.

7. Install VS Code and configure it for HTML, CSS, and JavaScript development.
Steps:



1. Download and install VS Code from the official website.
2. Install extensions: Live Server, Prettier, and JavaScript (ES6) snippets.
3. Create a new folder and files (index.html, style.css, script.js).
4. Use “Open with Live Server” to preview the webpage.

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

Static Website:
• Same content for all users.
• Built with HTML and CSS only.
• Faster loading but not interactive.
• Example: A company’s informational page.

Dynamic Website:
• Content changes dynamically based on user input or data.
• Uses backend technologies like PHP, Node.js, or Python.
• Interactive and personalized.
• Example: Facebook, YouTube, or Gmail.

9. Research and list five web browsers. Explain how rendering engines differ between them.
1. Google Chrome – **Blink Engine**
2. Mozilla Firefox – **Gecko Engine**
3. Apple Safari – **WebKit Engine**
4. Microsoft Edge – **Blink Engine** 5. Opera – **Blink Engine**

Rendering Engine Differences:
- Blink (Chrome/Edge/Opera) is fast and optimized for modern web standards.- Gecko (Firefox) offers flexibility and open-source customization.
- WebKit (Safari) is optimized for Apple devices, focusing on efficiency and performance.

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

Basic Flow:

1. Client (browser/app) sends a request through an API.
2. The server receives the request and processes it.
3. Server queries the database for required data.
4. Database returns data to the server.
5. Server sends a response back to the client through API.

Example: An e-commerce website fetching product details from a database to display to the user.