1. Explain the difference between frontend, backend, and full-stack development with

suitable real-world examples.

Ans: **Frontend vs. Backend vs. Full-Stack**

- **Frontend:** The visual part of a website you interact with (the user interface). It's built with HTML, CSS, and JavaScript. **Example:** The layout and buttons on a social media site.
- **Backend:** The "behind-the-scenes" part that makes the site work (server, database). It handles data and user requests. **Example:** Processing a login or saving a post.
- **Full-Stack:** A developer who can work on both the frontend and the backend

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

Ans: **Client-Server Model Diagram**

This diagram shows how your browser (**Client**) asks for a website from a **Server**, which then sends it back.

Code snippet

```
graph TD
  Client -- 1. Request (e.g., "GET homepage") --> Server;
  Server -- 2. Response (HTML, CSS, JS files) --> Client;
```

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

Ans: **URL to IP:** You type a URL (e.g., google.com). The browser uses **DNS** (like an internet phonebook) to find the server's **IP address**.

- **Request:** The browser sends an **HTTP request** to that IP address, asking for the webpage.
- **Response:** The server sends the webpage's files (HTML, CSS, JS) back in an **HTTP response**.
- **Render:** The browser reads the files and **renders** (draws) the visual webpage on your screen.

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

purpose of each.

Ans: **Code Editor:** To write and edit code. **Example:** VS Code.

- **Web Browser:** To view, test, and debug your website. **Example:** Chrome, Firefox.
- **Version Control:** To track code changes and collaborate. **Example:** Git.
- **Command Line (CLI):** To run commands and manage tools. **Example:** Terminal or PowerShell.

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

Ans: A **web server** is software that stores website files and delivers them to users' browsers upon request. Its main job is to "serve" web pages.

Common Examples: Apache, Nginx, Microsoft IIS.

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

in a project.

- Ans: **Frontend Developer:** Builds the user interface and visual elements of the website.
- **Backend Developer:** Manages the server, application logic, and database connections.
- **Database Administrator (DBA):** Designs, manages, and secures the database to ensure data is safe and accessible.

7. Install VS Code and configure it for HTML, CSS, and JavaScript development. Take a

screenshot of the setup.

- Ans: **Install VS Code:** Download and install it from code.visualstudio.com.
- **Install Extensions:** Open the Extensions view (sidebar icon) and install these key extensions:
  Live Server: Launches a local server that auto-refreshes your page on save.
  Prettier - Code formatter: Automatically formats your code for consistency.

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

each.

Ans: **Static Website:** The content is fixed and delivered to every user exactly as stored. It's fast and simple.

- **Example:** A personal portfolio or a simple informational brochure site.
- **Dynamic Website:** The content is generated in real-time based on user interaction or other data from a database.
- **Example:** Facebook, where your news feed is unique to you, or an e-commerce site showing personalized recommendations.

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

them.

Ans: A **rendering engine** is the core component of a browser that takes HTML and CSS and draws the visual webpage.

1. **Google Chrome:** Uses **Blink**.
2. **Mozilla Firefox:** Uses **Gecko**.
3. **Apple Safari:** Uses **WebKit**.
4. **Microsoft Edge:** Uses **Blink** (same as Chrome).

5. **Opera:** Uses **Blink** (same as Chrome).

Most modern browsers have consolidated around the **Blink** and **WebKit** engines, leading to better compatibility across websites. Firefox remains a notable exception with its independent **Gecko** engine.

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

database, and APIs.

Ans: This diagram shows how the main components interact. The client talks to the server, which uses APIs to communicate with the database to get or store data.

```
graph TD
  subgraph "User's Device"
    A[Client: Browser]
  end
  subgraph "Server Side"
    B[Web Server]
    C[APIs]
    D[Database]
  end
  A -- HTTP Request --> B
  B -- Calls --> C
  C -- Queries / Updates --> D
  D -- Returns Data --> C
  C -- Returns Data --> B
  B -- HTTP Response (Webpage) --> A
```