Assignment Part-1 (Short Answer Questions)

Q:1) What is client-side and server-side in web development, and what is the main difference between the two?

Ans: Client-side refers to processing that occurs on the user's device, handling the user interface and interactions using technologies like HTML, CSS, and JavaScript. Server-side refers to processing that happens on the remote server, handling requests, business logic, and generating dynamic content. The main difference is where the processing occurs: client-side on the user's device and server-side on the server.

Q:2) What is an HTTP request and what are the different types of HTTP requests?

Ans: An HTTP request is a message from a client to a server, specifying the desired action. It is part of the HTTP protocol used for communication between clients and servers on the web. Common types of HTTP requests include GET, POST, PUT, PATCH, DELETE etc.

Q:3) What is JSON and what is it commonly used for in web development?

Ans: JSON is a format used in web development to exchange data between servers and web applications. It is easy for humans to read and write, and computers can understand it easily too. JSON allows structured data to be represented in a simple way, making it ideal for transmitting information. It is widely used in web APIs and AJAX requests to send and receive data efficiently.

Q:4) What is a middleware in web development, and give an example of how it can be used.

Ans: Middleware is software that sits between a web application and the server, adding functionalities to the application. One example is authentication middleware, which enforces user authentication for protected routes. It checks if a user is authenticated before allowing access to certain parts of the application, enhancing security and controlling access to sensitive resources.

Q:5) What is a controller in web development, and what is its role in the MVC architecture?

Ans: Controller is a component that handles user requests and orchestrates the flow of a web application. It plays a central role in the Model-View-Controller (MVC) architecture by receiving user input, processing data, and coordinating actions between the view and the model. The controller helps separate concerns, improves modularity, and ensures efficient handling of user interactions in the web application.