Part 1: Short Answer Questions

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

Client-side is the web browser where web pages are displayed and certain behaviors happen.
Server-side is where the routing occurs, and where files and web pages are stored and served.

The main difference between client-side and server-side in web development is the location where the code is executed. Client-side code runs on the user's device, typically in a web browser, while server-side code runs on the server that hosts the website.

2.What is an HTTP request and what are the different types of HTTP requests?
An HTTP request is a message sent by a client (usually a web browser) to a server, asking for a specific resource or action.

There are several types of HTTP requests:
GET,POST,PUT,DELETE,PATCH etc.

3.What is JSON and what is it commonly used for in web development?
JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is primarily used for data transmission between a client and server in web development.

It is commonly used for APIs (Application Programming Interfaces).

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

In web development, middleware refers to functions that sit between the web application and the server. It intercepts and handles requests and responses, performing specific tasks before passing them to the next component in the processing chain. Middleware acts as a bridge, adding functionality to the application or modifying the request/response as required.

For example of middleware usage is authentication middleware. It can be used to validate and authenticate user requests before allowing access to protected routes or resources.

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

In web development, a controller is a component or module that handles the logic and behavior of a specific route or request in an application. It acts as an intermediary between the user interface (views) and the data model, processing user input, manipulating data, and coordinating the flow of information.

In the Model-View-Controller (MVC) architecture, the controller is responsible for receiving user input from the view, interpreting it, and triggering the appropriate actions or operations on the model (data) layer.