Part 1:

Question no 1: What is client-side and server-side in web development, and what is the main difference between the two?
Answer the question no 1: Client-side refers to the execution of code on the user's device (such as a web browser), while server-side refers to the execution of code on the server hosting the website or web application. The main difference is that client-side code runs on the user's device, while server-side code runs on the server before sending the result to the client.

Question no 2: What is an HTTP request and what are the different types of HTTP requests?
Answer the question no 2: An HTTP request is a message sent by a client to a server to initiate a specific action or retrieve information. The different types of HTTP requests are GET, POST, PUT, DELETE, HEAD, OPTIONS, and PATCH.

- GET is used to retrieve data from a server.
- POST is used to send data to the server to create a new resource.
- PUT is used to send data to update an existing resource on the server.
- DELETE is used to request the removal of a resource.
- HEAD is similar to GET but only returns the response headers.
- OPTIONS is used to retrieve the available methods or options for a resource.
- PATCH is used to partially update a resource on the server
-

Question no 3: What is JSON and what is it commonly used for in web development?
Answer the question no 3: 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 commonly used for data transmission between a server and a client in web development. JSON is often used to send structured data, such as configuration settings, API responses, or exchanging data between different parts of a web application. It is a popular choice due to its simplicity, wide browser support, and compatibility with various programming languages.

Question no 4: What is a middleware in web development, and give an example of how it can be used?
Answer the question no 4: In web development, middleware is a software component that sits between the client and server, intercepting and processing requests and responses. It adds functionality to the web application's request-response cycle.

For example, a common use of middleware is authentication. When a user makes a request to access a protected route, the authentication middleware checks if the user is authenticated or not. If the user is authenticated, the request proceeds to the intended route. If not, the middleware may redirect the user to a login page or return an error response. Middleware can also be used for tasks like logging, request parsing, caching, error handling, and more, allowing developers to modularize and customize the request-response flow.

Question no 5: What is a controller in web development, and what is its role in the MVC architecture?

Answer the question no 5: In web development, a controller is a component responsible for handling user requests, processing data, and coordinating the interaction between the user interface (views) and the data model. It plays a vital role in the Model-View-Controller (MVC) architectural pattern.

The controller receives input from the user, often through HTTP requests, and determines the appropriate actions to be taken. It interacts with the model to fetch or update data and then passes the processed data to the view for presentation to the user. The controller essentially acts as a mediator, facilitating the flow of information and logic between the model and view components, ensuring separation of concerns and maintaining the application's overall structure and behavior**.**