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?

Ans: **Client-side:** client-side web development is the process of creating web pages that are rendered on the user's browser.

Server-side: Server-side is the process of dynamic and interactive web pages that are generated by a server-side.

Difference between client-side and server-side

Feature	Client-side	Server-side
Location	Users browser	Web browser
Purpose	Display web pages, handle user interaction	Store data, generate dynamic content
Pros	Fast, easy to develop	More powerful, secure
Cons	Less powerful, less secure	Slow, more complex to develop

2. What is an HTTP request and what are the different types of HTTP requests?

Ans: An HTTP request is a message that is sent from a client to a server. It is used to request resources, such as a web page, from the server.

There are the most useful types of HTTP requests:

- GET
- POST
- PUT
- PATCH
- DELETE

3. What is JSON and what is it commonly used for in web development?

Ans: JSON stands for Javascript Object Notation. It is a lightweight data interchange format. JSON is easy to read and write for humans and machines. It is language-independent, so it can be used with any programming language. JSON is also commonly used for Data storing, Data transfer, APIs, and Web development frameworks,

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

Ans: Middleware is a software layer that sits between the client and the server in a web application. It is responsible for handling tasks such as authentication, authorization, and

routing. Middleware can also be used to add additional functionality to a web application, such as logging, caching, and error handling.

One example of how middleware can be used is to implement authentication. When a user visits a web application, the middleware can be used to check if the user is logged in. If the user is not logged in, the middleware can redirect the user to the login page. Once the user has logged in, the middleware can allow the user to access the protected resources on the web application.

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

Ans: In web development, a controller is a part of the Model-View-Controller (MVC) design pattern. It is responsible for receiving user input, interacting with the model, and selecting the view to render. The controller is the central point of communication between the model and the view, and it is responsible for ensuring that the user's input is valid and that the view is rendered correctly.

The controller is typically implemented as a class that inherits from a base controller class. The base controller class provides methods for handling common tasks, such as receiving user input, interacting with the model, and rendering the view. The controller class can then be extended to provide specific functionality for a particular web application.

The role of the controller in the MVC architecture is to:

- Receive user input
- Interpret user input and determine the appropriate action to take
- Interact with the model to retrieve or update data
- Select the appropriate view to render
- Render the view to the user