Web Engineering Viva and Interview Questions

Table of Contents

Introduction to Web Engineering

Basics of Web Development

What is the Web?

Client-Server Architecture

URL, HTTP, and HTTPS

Differences between Static and Dynamic Websites

HTML (Hypertext Markup Language)

Basic Structure of HTML

Tags and Attributes

Forms and Input Elements

Semantic HTML

Advanced HTML Features

CSS (Cascading Style Sheets)

Basics of CSS

Selectors and Properties

Layout Techniques (Flexbox, Grid)

Responsive Design

Advanced Concepts (Transitions, Animations, Pseudo-elements)

JavaScript (JS)

Basics of JavaScript

DOM Manipulation

Events and Event Handling

Promises and Asynchronous Programming

ES6+ Features

Error Handling

Advanced Topics: Closures, Hoisting, and Scope

React.js

Introduction to React

React Components

State and Props

Lifecycle Methods

React Hooks (useState, useEffect, useContext, etc.)

Routing in React

Advanced React Topics (Optimization, Virtual DOM, State Management Libraries)

Output-Oriented Discussion and Notes

Comparison of Technologies

Integration of HTML, CSS, JS, and React

Best Practices for Web Development

Common Mistakes and Debugging

Advanced Interview Output Points

1. Introduction to Web Engineering

Key Questions:

\*\*What is Web Engineering, and how does it differ from general software engineering?\*\*

Web Engineering is a systematic approach to developing web applications. It focuses on methodologies, tools, and processes specific to web-based systems, while general software engineering deals with all types of software development.

\*\*Why is web engineering important in today’s world?\*\*

Web engineering is essential because the internet powers modern businesses, e-commerce, education, and communication. It ensures reliable, scalable, and secure web systems.

\*\*What are the major phases of the web development lifecycle?\*\*

The phases include:

1. Requirement Analysis

2. Design

3. Development

4. Testing

5. Deployment

6. Maintenance

\*\*Explain the concept of client-server architecture with an example.\*\*

Client-server architecture is a model where a client (browser) requests resources, and the server provides them. For example, when you open a webpage, the browser (client) requests files from the web server.

\*\*What are the core components of a web application?\*\*

1. Frontend (HTML, CSS, JS)

2. Backend (Server-side programming)

3. Database

4. Web Server

2. Basics of Web Development

Key Questions:

\*\*What is the World Wide Web (WWW), and how does it work?\*\*

The World Wide Web is a system of interlinked documents accessible through the internet. It works via HTTP requests and responses between clients and servers.

\*\*Differentiate between HTTP and HTTPS.\*\*

- \*\*HTTP\*\*: Hypertext Transfer Protocol, unencrypted.

- \*\*HTTPS\*\*: Secure version of HTTP using SSL/TLS encryption for security.

\*\*Explain the structure of a URL with an example.\*\*

URL: `protocol://domain:port/path`

Example: `https://www.example.com:80/index.html`

- Protocol: https

- Domain: www.example.com

- Port: 80

- Path: /index.html

\*\*What are static and dynamic websites? Give examples.\*\*

- \*\*Static Websites\*\*: Content doesn’t change, e.g., HTML-only pages.

- \*\*Dynamic Websites\*\*: Content changes based on user input, e.g., Facebook.

\*\*Define web servers and web browsers.\*\*

- \*\*Web Server\*\*: A system that delivers content to the web (e.g., Apache, Nginx).

- \*\*Web Browser\*\*: Software to access and display web pages (e.g., Chrome, Firefox).

\*\*Explain the role of DNS in web browsing.\*\*

DNS (Domain Name System) translates domain names (e.g., www.google.com) into IP addresses.

Key Points for Notes:

- \*\*HTTP\*\*: Stateless protocol, request-response model.

- \*\*HTTPS\*\*: Encrypted version of HTTP.

- \*\*URL\*\*: Uniform Resource Locator structure - protocol://domain:port/path.

- \*\*Static Websites\*\*: Fixed content, e.g., HTML-only pages.

- \*\*Dynamic Websites\*\*: Content changes based on user input, e.g., PHP, JS, React.

3. HTML (Hypertext Markup Language)

Basic HTML Questions:

\*\*What is HTML, and why is it used?\*\*

HTML is the standard markup language for creating web pages. It provides structure to content.

\*\*What is the difference between HTML and XHTML?\*\*

- HTML: Flexible syntax, not strict.

- XHTML: Stricter, XML-compliant version of HTML.

\*\*What are the main components of an HTML document?\*\*

1. `<!DOCTYPE>` Declaration

2. `<html>` Element

3. `<head>` Element

4. `<body>` Element

\*\*Explain the purpose of the <!DOCTYPE> declaration.\*\*

It tells the browser which version of HTML to render.

Intermediate Questions:

\*\*What are semantic elements in HTML? Why are they important?\*\*

Semantic elements provide meaning to content (e.g., `<article>`, `<section>`). They improve accessibility and SEO.

\*\*Explain the difference between inline and block-level elements.\*\*

- \*\*Inline\*\*: Takes up space only for its content (e.g., `<span>`).

- \*\*Block-level\*\*: Starts on a new line and stretches to full width (e.g., `<div>`).

\*\*How do you create a form in HTML? List form attributes.\*\*

Forms are created using `<form>` tags. Key attributes:

- `action`: URL to send data to

- `method`: GET or POST

\*\*What are the types of input elements in HTML?\*\*

- Text, password, checkbox, radio, file, submit, and more.

Advanced Questions:

\*\*How does HTML5 differ from HTML4?\*\*

HTML5 introduced new elements (e.g., `<header>`, `<section>`), multimedia support, and form enhancements.

\*\*What is the purpose of the data-\* attribute?\*\*

It allows storing custom data attributes in elements.

\*\*How do you embed multimedia content in HTML?\*\*

Using `<audio>`, `<video>`, and `<embed>` tags.

Output Points:

- Semantic HTML improves accessibility and SEO.

- Example of tags: `<header>`, `<article>`, `<section>`, etc.

- Use proper nesting of elements.

4. CSS (Cascading Style Sheets)

Basic CSS Questions:

\*\*What is CSS? Why is it important?\*\*

CSS styles HTML content, ensuring a visually appealing layout.

\*\*What is the difference between internal, external, and inline CSS?\*\*

- \*\*Inline\*\*: Within elements

- \*\*Internal\*\*: In `<style>` tags

- \*\*External\*\*: Linked via `<link>`

\*\*What is a selector in CSS? List different types of selectors.\*\*

A selector targets HTML elements for styling. Types:

- Class, ID, Universal, Attribute, Pseudo-classes.

Intermediate Questions:

\*\*Explain the box model in CSS.\*\*

It includes content, padding, border, and margin.

\*\*How does the position property work?\*\*

- static, relative, absolute, fixed, sticky.

\*\*What is Flexbox, and how is it used?\*\*

Flexbox is a layout model for one-dimensional alignment of elements.

\*\*Explain the difference between Flexbox and CSS Grid.\*\*

- Flexbox: One-dimensional layout

- Grid: Two-dimensional layout.

\*\*What are media queries, and how do they enable responsive design?\*\*

Media queries apply styles based on screen size or device.

Advanced CSS Questions:

\*\*How can you create animations and transitions in CSS?\*\*

Using `@keyframes` for animations and `transition` for smooth property changes.

\*\*What are pseudo-classes and pseudo-elements? Provide examples.\*\*

- Pseudo-class: `:hover`

- Pseudo-element: `::after`

\*\*How does the z-index property work?\*\*

It controls the stacking order of elements.

Output Points:

- CSS Grid vs. Flexbox: Grid for 2D layouts, Flexbox for 1D layouts.

- Responsive Design: Mobile-first approach.

- Use transitions/animations for interactive UI.

End of Document