**Programming Lab 3**

**Practical N0. 1**

**Name: Triahul Balaso Patil**

**PRN : 22510051**

**Batch: T5**

**Topic: Study ofWeb and its basics**

**Problem Statement 1:**

Study and describe the following concepts in your words:

**Q.1)Evolution of Web including web 3.0**

### Web 1.0 (Static Web)

* Period: 1990s - Early 2000s
* Characteristics: Static pages, read-only content, basic HTML, limited interactivity, early search engines.
* Examples: Early websites like GeoCities, Yahoo!.

### Web 2.0 (Social Web)

* Period: Early 2000s - Present
* Characteristics: Dynamic and user-generated content, interactive and social media, AJAX and JavaScript, user accounts, and collaboration.
* Examples: Facebook, YouTube, Wikipedia.

### Web 3.0 (Semantic Web)

* Period: Emerging
* Characteristics: Decentralized, blockchain-based, AI and machine learning, data interoperability, personalized and intelligent services.
* Examples: Decentralized applications (dApps), blockchain platforms like Ethereum, AI-driven services.

**Q.2)Which ports and protocols are used by web? Describe those in detail**

### Ports

1. Port 80 (HTTP)
   * Purpose: Used for unencrypted web traffic.
   * Description: HTTP (HyperText Transfer Protocol) is the foundation of data communication on the web. When a user types a URL in the browser, it sends an HTTP request to the server on port 80. The server responds with the requested web page or resource.
2. Port 443 (HTTPS)
   * Purpose: Used for encrypted web traffic.
   * Description: HTTPS (HyperText Transfer Protocol Secure) is the secure version of HTTP. It uses SSL/TLS (Secure Sockets Layer/Transport Layer Security) to encrypt data between the client and server, ensuring secure communication. Port 443 is standard for HTTPS, providing confidentiality, integrity, and authentication.

### Protocols

1. HTTP (HyperText Transfer Protocol)
   * Purpose: Facilitates the transfer of hypertext documents (web pages) on the internet.
   * Description: HTTP is a request-response protocol. Clients send HTTP requests (GET, POST, PUT, DELETE, etc.) to servers, which respond with status codes (200 OK, 404 Not Found, etc.) and the requested content. HTTP operates over TCP (Transmission Control Protocol), providing reliable data transmission.
2. HTTPS (HyperText Transfer Protocol Secure)
   * Purpose: Secure version of HTTP for encrypted communication.
   * Description: HTTPS uses SSL/TLS to encrypt data between clients and servers. This ensures that data cannot be intercepted or tampered with during transmission. HTTPS is essential for securing sensitive information like login credentials, payment details, and personal data.
3. TCP (Transmission Control Protocol)
   * Purpose: Provides reliable, ordered, and error-checked delivery of data.
   * Description: TCP is a core protocol of the internet protocol suite. It ensures that data packets are delivered in the correct order and without errors. HTTP and HTTPS rely on TCP to provide reliable communication between clients and servers.
4. SSL/TLS (Secure Sockets Layer/Transport Layer Security)
   * Purpose: Provides encryption and security for data transmission.
   * Description: SSL and its successor, TLS, are cryptographic protocols designed to provide secure communication over a computer network. They establish an encrypted link between a web server and a browser, ensuring that all data passed between them remains private and secure. HTTPS uses SSL/TLS for secure communication.

**Q.3)Difference Between HTTP & HTTPS**

### HTTP (HyperText Transfer Protocol)

1. Security:
   * Encryption: HTTP does not encrypt data. Information is sent as plain text, making it vulnerable to interception by hackers.
   * Security Risks: Susceptible to eavesdropping, man-in-the-middle attacks, and data tampering.
2. Port:
   * Default Port: Port 80.
3. SSL/TLS:
   * Usage: Does not use SSL/TLS for data transmission.
   * Certificate Requirement: No need for SSL/TLS certificates.
4. URL Prefix:
   * Prefix: URLs begin with http://.
5. Performance:
   * Speed: Slightly faster than HTTPS because it does not need to encrypt/decrypt data.
   * Overhead: Lower overhead due to the absence of encryption.
6. Common Uses:
   * Usage: Often used for non-sensitive data and information where security is not a concern, such as static content and public information websites.

### HTTPS (HyperText Transfer Protocol Secure)

1. Security:
   * Encryption: HTTPS encrypts data using SSL/TLS, making it secure from eavesdropping and man-in-the-middle attacks.
   * Security Benefits: Ensures data confidentiality, integrity, and authenticity.
2. Port:
   * Default Port: Port 443.
3. SSL/TLS:
   * Usage: Utilizes SSL/TLS to establish a secure connection.
   * Certificate Requirement: Requires an SSL/TLS certificate issued by a Certificate Authority (CA) to authenticate the server.
4. URL Prefix:
   * Prefix: URLs begin with https://.
5. Performance:
   * Speed: Slightly slower than HTTP due to the overhead of encryption/decryption processes.
   * Overhead: Higher overhead because of encryption, but modern optimizations like HTTP/2 can mitigate some performance impacts.

**Q.4)What is web developer tools and why it is needed?**

### What Are Web Developer Tools?

Web Developer Tools (often referred to as DevTools) are a set of utilities built into modern web browsers that allow developers to inspect, debug, and analyze websites and web applications. These tools provide a way to interact with and manipulate the code of a webpage in real-time, making it easier to develop, optimize, and troubleshoot web projects.

Most popular browsers like Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari come with built-in developer tools that offer a range of features for developers.

### Key Features of Web Developer Tools

1. Inspect Element:
   * Allows developers to view and edit the HTML and CSS of a webpage.
   * Useful for testing and modifying the design and structure of a page without needing to change the source code directly.
2. Console:
   * A command-line interface within the browser that logs errors, warnings, and other messages from JavaScript.
   * Developers can run JavaScript code directly in the console to test functions and debug issues.
3. Network Monitoring:
   * Tracks all network requests made by the webpage, including HTTP requests, file loads, and API calls.
   * Useful for analyzing the performance of a website, understanding loading times, and identifying slow or failing requests.
4. Performance Profiling:
   * Provides tools to analyze the performance of a webpage, including frame rates, memory usage, and CPU load.
   * Helps developers optimize the speed and efficiency of a web application.
5. Debugging Tools:
   * Allows setting breakpoints in JavaScript code to pause execution and inspect variables, functions, and the call stack.
   * Essential for identifying and fixing bugs in complex web applications.
6. Responsive Design Testing:
   * Simulates different screen sizes and resolutions, allowing developers to see how their website looks and behaves on various devices.
   * Important for ensuring that websites are mobile-friendly and responsive.
7. Application and Storage Inspection:
   * Lets developers view and manage data stored in the browser, such as cookies, local storage, and session storage.
   * Provides insights into how web applications are managing client-side data.
8. Security Panel:
   * Displays information about the security of the webpage, such as SSL certificates, mixed content issues, and security headers.
   * Helps developers ensure that their web applications meet security standards.

### Why Are Web Developer Tools Needed?

1. Efficient Debugging:
   * DevTools enable developers to quickly identify and fix issues with HTML, CSS, and JavaScript. This speeds up the development process and reduces the time spent troubleshooting problems.
2. Performance Optimization:
   * By analyzing network requests, memory usage, and rendering performance, developers can identify bottlenecks and optimize the speed and efficiency of their web applications.
3. Responsive Design:
   * With the rise of mobile devices, ensuring that websites work well across different screen sizes is crucial. DevTools allow developers to test and tweak responsive designs directly within the browser.
4. Real-Time Changes:
   * Developers can experiment with changes in real-time without needing to modify the

**Q.5)Elaborate with diagram client server architecture and MVC architecture. When to use which architecture?**

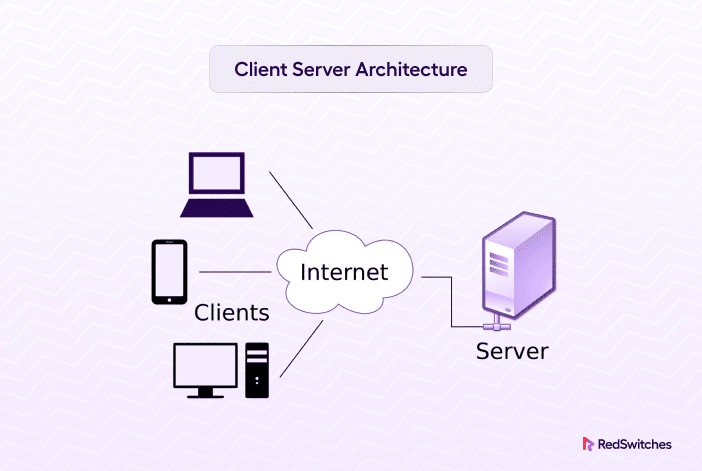
### 1. **Client-Server Architecture**

#### **Overview**

Client-Server Architecture is a distributed application structure that divides tasks between servers, which provide a resource or service, and clients, which request the resource or service. The server hosts, delivers, and manages most of the resources and services to be consumed by the client.

* Client: The device or application that requests services or resources from the server. It could be a web browser, mobile app, etc.
* Server: The system that provides services or resources to the client. It could be a web server, database server, application server, etc.

#### **Diagram**



#### **How It Works**

1. Client Request: The client sends a request to the server over a network (e.g., requesting a webpage, data from a database, etc.).
2. Server Processing: The server receives the request, processes it (e.g., fetching data, performing computations), and prepares a response.
3. Response to Client: The server sends the response back to the client (e.g., the requested webpage, data, etc.).
4. Client Displays: The client processes and displays the information to the user.

#### **When to Use Client-Server Architecture**

* Centralized Resource Management: When you need centralized management of resources and services.
* Scalability: When the system must support multiple clients simultaneously.
* Security: When you need to secure resources in a centralized manner.
* Complexity: Suitable for applications that require complex processing that should be centralized (e.g., banking systems, online games).

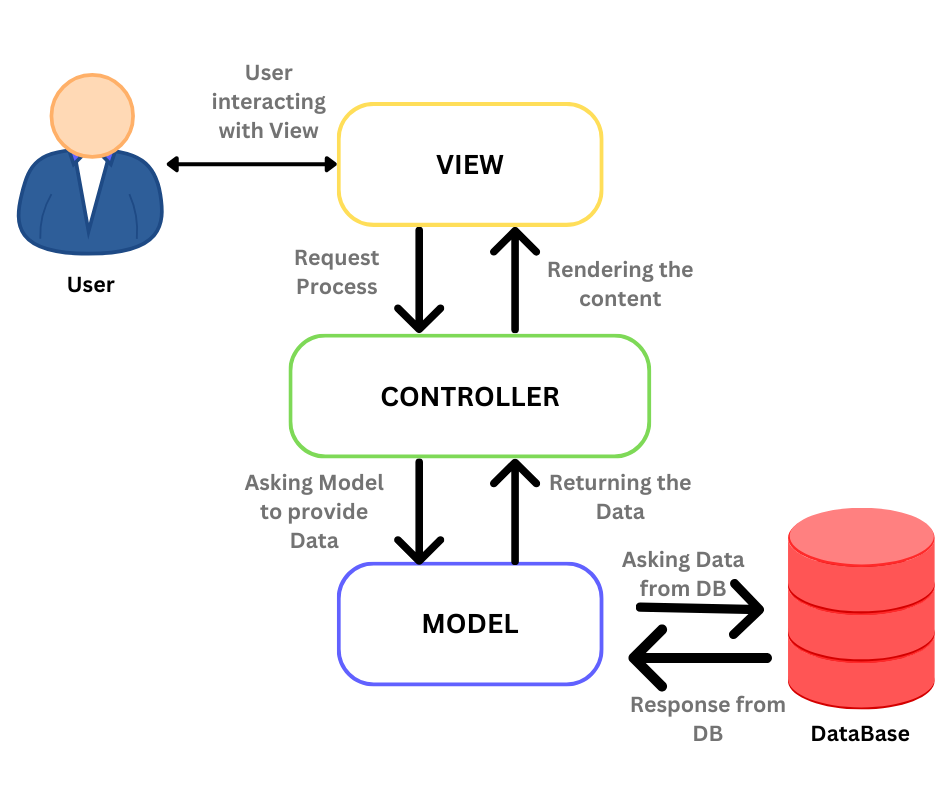
### 2. **MVC Architecture (Model-View-Controller)**

#### **Overview**

MVC (Model-View-Controller) is a software design pattern that separates an application into three main logical components: the Model, the View, and the Controller. This separation helps manage complex applications by dividing responsibilities among different components.

* Model: Represents the application's data and business logic. It handles data retrieval, storage, and processing.
* View: Represents the user interface of the application. It displays data to the user and sends user commands to the Controller.
* Controller: Manages the flow of the application. It processes user input, interacts with the Model, and updates the View.

#### **Diagram**



#### **How It Works**

1. User Interaction (View): The user interacts with the UI (e.g., clicking a button, entering data).
2. Request Handling (Controller): The Controller receives the user input, processes it, and determines what needs to be done (e.g., retrieve data, submit a form).
3. Data Manipulation (Model): The Controller interacts with the Model to retrieve, update, or delete data as necessary.
4. Updating the View: The Model provides data to the Controller, which then updates the View to reflect the new state of the application.
5. User Feedback: The updated View is presented to the user, completing the cycle.

#### **When to Use MVC Architecture**

* Separation of Concerns: When you want to separate the application's logic, user interface, and control flow.
* Modularity: When building complex applications that require modular development.
* Maintainability: When the application needs to be easily maintainable and scalable, making it easier to update, test, and debug.
* Reuse: When different views can be created for the same data model, making the application more flexible (e.g., web applications, desktop applications).

### **Comparing Client-Server and MVC Architectures**

* Client-Server focuses on how resources and services are distributed and managed between clients and servers, usually in a networked environment. It’s more about the interaction between different machines.
* MVC is about structuring the code within an application, typically a single software running on a server. It’s more about organizing and separating concerns within a single application.

### **When to Use Which Architecture**

* Use Client-Server Architecture when:
  + You are building a distributed system where multiple clients need to access shared resources hosted on a central server.
  + Network communication between different devices is a key component.
  + Centralized control and security are important.
* Use MVC Architecture when:
  + You are developing an application where clean separation between data, business logic, and UI is required.
  + The application will benefit from modularity, making it easier to manage and scale.
  + The application needs to be flexible and maintainable, particularly for web or software applications.

**Q.6) What is HTML and HTML5 ?**

HTML (HyperText Markup Language) is the standard language used to create and design web pages. It provides the basic structure of a webpage, which is then enhanced and modified by other technologies like CSS (Cascading Style Sheets) and JavaScript. HTML uses "tags" to define elements within a webpage, such as headings, paragraphs, links, images, and other content.

#### Key Features of HTML:

* Markup Language: HTML uses a system of tags to annotate text and images, defining how they should be displayed in a web browser.
* Document Structure: HTML defines the structure of a webpage using elements like <head>, <body>, <title>, <header>, <footer>, etc.
* Content Formatting: It includes tags to format content, such as <h1> for headings, <p> for paragraphs, <a> for hyperlinks, <img> for images, and <table> for tables.
* Basic Interactivity: HTML can include forms (<form>, <input>, <button>), which allow user interaction.

### **What is HTML5?**

HTML5 is the fifth and latest major version of HTML, introduced to improve and enhance the capabilities of HTML. It was developed by the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG). HTML5 includes new elements, attributes, and behaviors, giving developers more powerful tools to build modern web applications and sites.

#### Key Features of HTML5:

* New Semantic Elements: HTML5 introduces new tags like <header>, <footer>, <article>, <section>, and <nav> to better define the structure and meaning of content on the page.
* Multimedia Support: HTML5 provides native support for audio and video elements with the <audio> and <video> tags, eliminating the need for third-party plugins like Flash.
* Graphics and Animation: HTML5 includes the <canvas> element, which allows for drawing graphics directly in the browser using JavaScript. It also supports scalable vector graphics (SVG) for creating vector images.
* Forms and Input Types: HTML5 enhances form elements with new input types like <input type="email">, <input type="date">, <input type="range">, and others, making it easier to collect and validate user input.
* Geolocation API: HTML5 introduces the Geolocation API, allowing web applications to access the geographical location of the user.
* Offline and Storage: HTML5 includes new features like local storage and session storage, which allow web applications to store data on the client-side. It also supports offline capabilities with the Application Cache.
* Improved Accessibility: HTML5 is designed to improve accessibility with better support for screen readers and other assistive technologies.
* Performance Enhancements: HTML5 focuses on performance improvements, including faster parsing of HTML documents and better handling of scripts.

**Q.7)Which are the currently used versions of HTML and CSS for web development?**

### 1. **HTML5**

* Version: HTML5 is the current standard version of HTML.
* Overview: HTML5 was officially released by the World Wide Web Consortium (W3C) in October 2014, and it has been widely adopted for web development. It introduced new semantic elements, multimedia support, and APIs, making it a powerful tool for creating modern, interactive, and responsive web applications.
* Why It’s Used: HTML5 offers better structure, more capabilities (like video, audio, canvas, and local storage), and improved accessibility compared to earlier versions. It is designed to be cross-platform and is supported by all modern browsers.

### 2. **CSS3**

* Version: CSS3 is the latest standard version of CSS (Cascading Style Sheets).
* Overview: CSS3 is not a single monolithic specification but rather a collection of modular specifications. It introduced new features such as transitions, animations, shadows, gradients, and flexible box layouts (Flexbox), as well as grid layouts.
* Why It’s Used: CSS3 allows for more design flexibility and control, enabling developers to create visually appealing, responsive, and interactive web pages. It also enhances performance and compatibility across different browsers and devices.

**Q.8)Which tools are available for front end development and back end development?**

### **Front-End Development Tools:**

1. Languages & Libraries:
   * HTML5, CSS3, JavaScript
   * React, Angular, Vue.js
2. CSS Frameworks:
   * Bootstrap, Tailwind CSS
3. Build Tools:
   * Webpack, Gulp
4. Text Editors:
   * VS Code, Sublime Text
5. Testing:
   * Jest, Cypress

### **Back-End Development Tools:**

1. Languages & Frameworks:
   * Node.js (Express.js), Python (Django, Flask), Ruby (Rails), Java (Spring Boot), C# (.NET)
2. Databases:
   * MySQL, PostgreSQL, MongoDB
3. Version Control:
   * Git, GitHub
4. CI/CD & Deployment:
   * Docker, Jenkins, AWS

**Q.9)What MERN stack includes? Why and when it is preferred for web development?**

### **What MERN Stack Includes:**

* M: MongoDB (NoSQL database)
* E: Express.js (Web application framework for Node.js)
* R: React (Front-end JavaScript library)
* N: Node.js (JavaScript runtime environment)

### **Why and When to Use the MERN Stack:**

* Full JavaScript Stack: JavaScript is used across the entire application (front-end, back-end, database).
* Single-Page Applications (SPAs): Ideal for building dynamic, responsive SPAs.
* Fast Development: React's component-based architecture and Node.js's non-blocking I/O model speed up development.
* Scalability: MongoDB's flexible schema and Node.js's event-driven architecture make scaling easier.
* When Preferred: Use MERN for building modern, high-performance web apps that require dynamic, real-time user interactions.

**Q.10)List out newly introduced input types, APIs, form elements, and elements that support media content in HTML5.**

### **Newly Introduced in HTML5:**

#### **Input Types:**

* type="email": For email addresses.
* type="date": For selecting dates.
* type="color": For color picker.
* type="range": For sliders.
* type="number": For numeric input.
* type="tel": For telephone numbers.
* type="url": For URLs.

#### **APIs:**

* Geolocation API: Access user's location.
* Web Storage API: localStorage and sessionStorage for storing data.
* Canvas API: Drawing graphics on the web.
* Offline Web Applications: AppCache for offline use.
* WebSocket API: Real-time communication.

#### **Form Elements:**

* <datalist>: Auto-suggestions for input fields.
* <output>: Displays the result of a calculation.
* <progress>: Shows the progress of a task.
* <meter>: Displays a scalar measurement within a known range.

#### **Media Content Elements:**

* <audio>: Embeds audio files.
* <video>: Embeds video files.
* <track>: Adds subtitles or captions to media.
* <source>: Specifies multiple media resources.

**Q.11)Explain HTML5 Web storage**

HTML5 Web Storage

HTML5 Web Storage provides a way to store data in the user's browser more securely and efficiently than traditional cookies. It offers two key mechanisms:

#### **1. localStorage**

* Persistent Storage: Data is stored with no expiration date, remaining even after the browser is closed and reopened.
* Scope: Data is accessible within the same domain across all tabs and windows.
* Storage Limit: Typically 5-10 MB per domain.
* Use Cases: Storing user preferences, themes, or data that needs to persist across sessions.

#### **2. sessionStorage**

* Temporary Storage: Data is stored for the duration of the page session and is deleted when the page is closed.
* Scope: Data is accessible only within the tab where it was set and is not shared across different tabs or windows.
* Storage Limit: Similar to localStorage, but data is cleared when the session ends.
* Use Cases: Managing temporary state, like shopping cart items or form data during a session.

### **Key Features:**

* Key-Value Pairs: Data is stored as simple key-value pairs, where both keys and values are strings.
* JavaScript Access: Data is easily accessed and manipulated using JavaScript through methods like setItem(), getItem(), removeItem(), and clear().

**Problem Statement 2:**

Study of different HTML and CSS tags:

**2.Create a static web page for “Portfolio” of your own. Which will include photo, name,**

**class, College name, Achievements/ Certificates, Extracurricular Activities, Courses**

**Completed, hobbies, Technical expertise, etc**

