**PRN: 22510111**

**Name: Daivik Liladhar Karbhari**

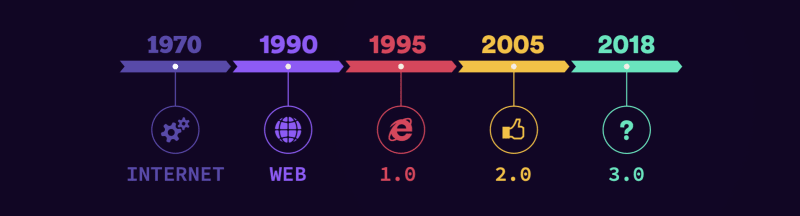
**Practical No. 1**

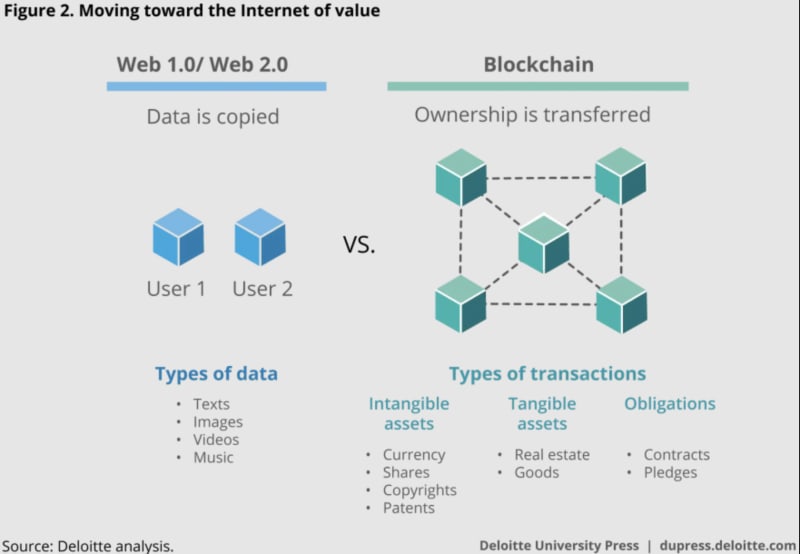
**Study of Web and its basics**

**Problem Statement 1:**

Study and describe the following concepts in your words:

1. Evolution of Web including web 3.0





* Web 1.0 was the first stage of the World Wide Web revolution, usually referred to as **read-only web**.
* Web 2.0 was the second stage of the evolution of the web, also called the **read-write web** and it was the phase when websites grew in terms of user interaction. **participative social web**.
* Web 3.0 is the next generation of web, also termed as the **executable web** or **read- write-execute** web.
* The future of Web 3.0 points to universal applications which can be read and used by a large number of devices and software types, making the ways in which we indulge business and leisure increasingly convenient.

1. Which ports and protocols are used by web? Describe those in detail

-> **HTTP (Hypertext Transfer Protocol)**

* **Port:** HTTP typically uses **port 80**.
* **Protocol:** It is a protocol for transferring hypertext requests and information on the internet.
* **Details:**
  + **Request/Response Model:** HTTP operates on a request/response model. The client sends an HTTP request to the server, and the server returns an HTTP response.
  + **Stateless:** Each request from a client to a server is treated as an independent transaction. This means that the server does not retain any information (state) between different requests.
  + **Methods:** Common HTTP methods include:
    - **GET:** Requests data from a specified resource.
    - **POST:** Submits data to be processed to a specified resource.
    - **PUT:** Updates a specified resource.
    - **DELETE:** Deletes a specified resource.
    - **HEAD:** Similar to GET but returns only the headers and not the body of the response.
  + **Status Codes:** HTTP responses come with status codes indicating the result of the request, such as 200 (OK), 404 (Not Found), 500 (Internal Server Error).

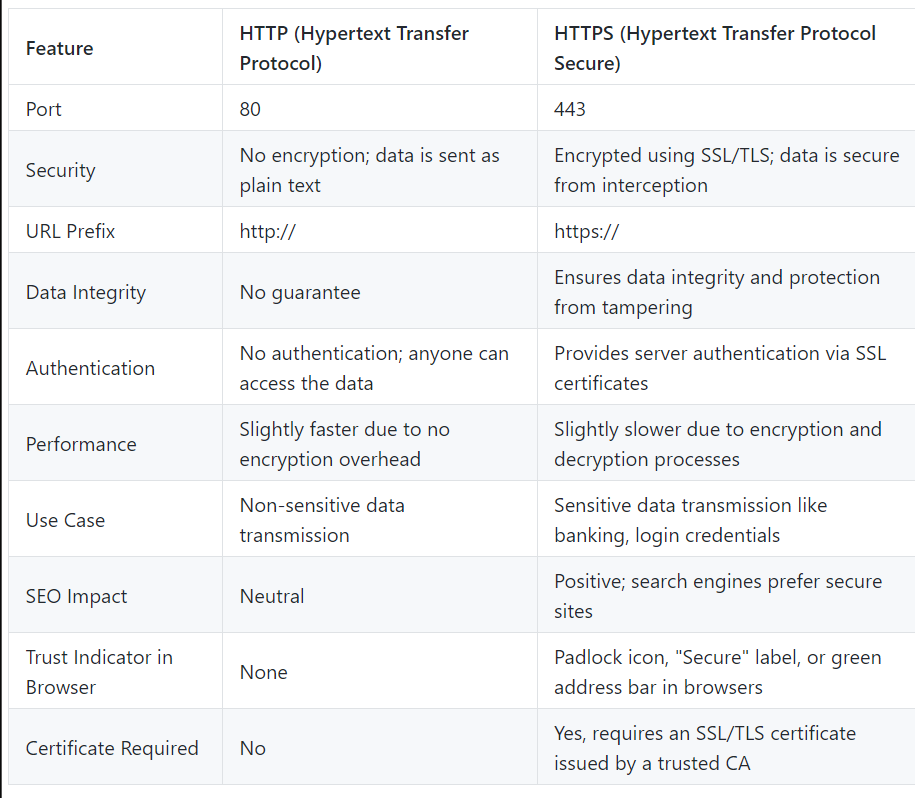
**HTTPS (Hypertext Transfer Protocol Secure)**

* **Port:** HTTPS typically uses **port 443**.
* **Protocol:** HTTPS is HTTP with encryption. It uses Transport Layer Security (TLS) or its predecessor, Secure Sockets Layer (SSL), to encrypt the data transferred between the client and the server.
* **Details:**
  + **Encryption:** HTTPS ensures that the data transferred between the client and the server is encrypted, protecting it from eavesdropping and man-in-the-middle attacks.
  + **Certificate-Based Authentication:** HTTPS uses certificates issued by trusted Certificate Authorities (CAs) to verify the identity of the server, and optionally the client.
  + **TLS Handshake:** During the initial connection, a TLS handshake occurs where the client and server agree on the encryption methods to be used and establish a secure connection.

**FTP (File Transfer Protocol)**

* **Ports:** FTP typically uses **port 21** for command/control and **port 20** for data transfer.
* **Protocol:** FTP is used to transfer files between a client and a server on a computer network.
* **Details:**
  + **Control Connection:** Established on port 21 to send commands from the client to the server and receive responses.
  + **Data Connection:** Established on port 20 for the actual data transfer. This can operate in active or passive mode, affecting how connections are established.

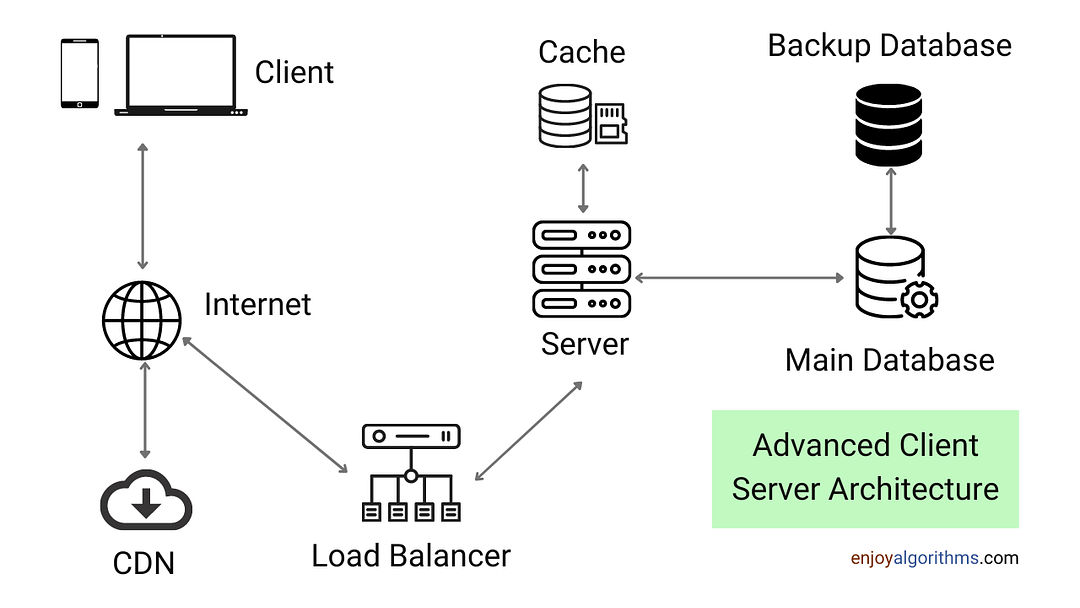
1. Difference Between HTTP & HTTPS



1. What is web developer tools and why it is needed?

* Web developer tools, commonly referred to as devtools, are a set of built-in features in web browsers that assist developers in building, debugging, and optimizing websites or web applications.
* **Element Inspector**: Allows developers to examine and modify the HTML and CSS of a webpage in real-time.
* **Console**: Provides a command-line interface for developers to log information, run JavaScript code, and debug errors.
* **Network Monitor**: Tracks all network requests made by the page, including loading times, HTTP status codes, and transferred data size.
* **Performance Panel**: Helps analyze the performance of the web page, identifying bottlenecks and rendering issues.
* **Sources Panel**: Allows developers to view, edit, and debug JavaScript code, set breakpoints, and step through code execution.
* **Application Panel**: Provides access to various web storage APIs, including localStorage, sessionStorage, cookies, and IndexedDB.
* **Security Panel**: Displays information about the security of the webpage, including SSL/TLS certificate details and potential security issues.
* **Accessibility Panel**: Helps ensure web content is accessible to all users, including those with disabilities, by providing tools to check and improve accessibility.
* Debugging, Performance Optimization, Responsive Design Testing, Real-Time Editing, Security, Accessibility, Network Analysis, Code Quality.

1. Elaborate with diagram client server architecture and MVC architecture. When to use which architecture?



Client-server architecture is a computing model in which the server hosts, delivers and manages most of the resources and services to be consumed by the client. This type of architecture has one or more client computers connected to a central server over a network or internet connection.

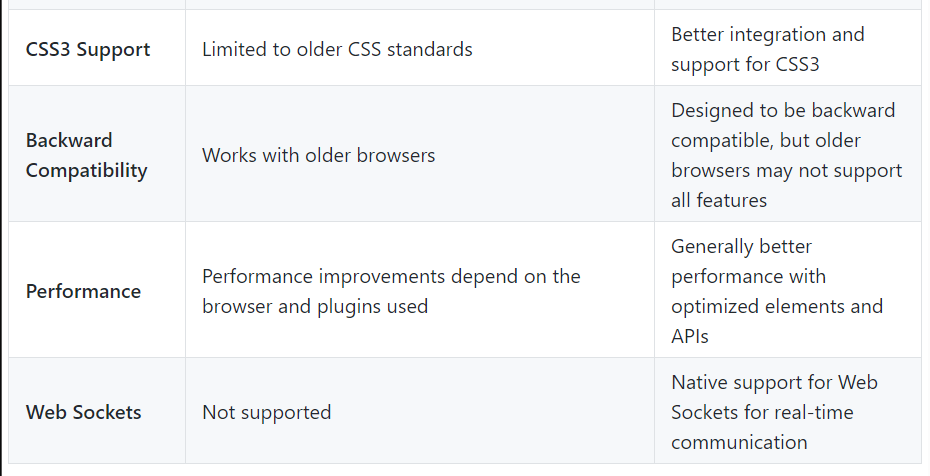
**MVC** (Model-View-Controller) is a pattern in software design commonly used to implement user interfaces, data, and controlling logic. It emphasizes a separation between the software's business logic and display. This "separation of concerns" provides for a better division of labor and improved maintenance. Some other design patterns are based on MVC, such as MVVM (Model-View-Viewmodel), MVP (Model-View-Presenter), and MVW (Model-View-Whatever).

The three parts of the MVC software-design pattern can be described as follows:

1. Model: Manages data and business logic.
2. View: Handles layout and display.
3. Controller: Routes commands to the model and view parts.
4. What is HTML and HTML5?







1. Which are the currently used versions of HTML and CSS for web development?

**HTML**:

* **HTML5**: The latest version of HTML, which includes new elements, attributes, and behaviors, as well as a larger set of technologies that allow more diverse and powerful websites and applications.

**CSS**:

* **CSS3**: The latest standard for CSS, which introduces new features such as flexbox, grid layout, animations, transitions, and various new selectors and properties to enhance styling capabilities.

1. Which tools are available for front end development and backend development?

**Frontend Development Tools**:

* **Text Editors/IDEs**: Visual Studio Code, Sublime Text, Atom, WebStorm.
* **Frameworks/Libraries**: React, Angular, Vue.js, Svelte.
* **Build Tools**: Webpack, Gulp, Parcel, Vite.
* **Version Control**: Git, GitHub, GitLab, Bitbucket.
* **Design Tools**: Adobe XD, Figma, Sket/ch.

**Backend Development Tools**:

* **Languages**: Node.js (JavaScript), Python (Django, Flask), Ruby (Rails), Java (Spring), PHP (Laravel).
* **Databases**: MySQL, PostgreSQL, MongoDB, Redis, SQLite.
* **Servers**: Apache, Nginx, Express.js (for Node.js).
* **Version Control**: Git, GitHub, GitLab, Bitbucket.
* **API Development**: Postman, Swagger.

1. What MERN stack includes? Why and when it is preferred for web development?

The MERN stack consists of:

* **MongoDB**: A NoSQL database for storing data as JSON-like documents.
* **Express.js**: A web application framework for Node.js, used to build the backend.
* **React**: A JavaScript library for building user interfaces, primarily for single-page applications.
* **Node.js**: A JavaScript runtime environment that allows the execution of JavaScript on the server side.

When and why:

* **Unified Language**: Uses JavaScript for both frontend and backend, simplifying the development process.
* **Performance**: Efficient handling of asynchronous operations with Node.js.
* **Scalability**: Suitable for building scalable and maintainable web applications.
* **Active Community**: Strong community support and a wide range of libraries and tools.

1. List out newly introduced input types, APIs, form elements, and elements that support media content in HTML5.

**New Input Types**:

* email, url, tel, number, range, date, datetime-local, month, week, time, color

**New APIs**:

* **Canvas API**: For drawing graphics on a web page.
* **Web Storage API**: localStorage and sessionStorage for client-side storage.
* **Geolocation API**: For accessing the user's location.
* **Web Workers API**: For running scripts in the background.
* **Drag and Drop API**: For enabling drag and drop functionality.

**New Form Elements**:

* <datalist>, <output>, <progress>, <meter>

**Media Elements**:

* <audio>, <video>, <source>, <track>

1. Explain HTML5 Web storage.

HTML5 Web Storage provides a way to store data on the client's browser that is more secure and faster than traditional cookies. It introduces two types of storage: localStorage and sessionStorage. localStorage allows data to be stored with no expiration date, meaning it persists even after the browser is closed and reopened. sessionStorage stores data for the duration of the page session, which means data is cleared when the page session ends. Web Storage is useful for storing user preferences, session information, or any data that needs to be retained between page loads without sending it to the server, thus improving performance and reducing server load.

**Problem Statement 2:**

Study of Different HTML and CSS tags:

1. Study different tags of HTML and CSS

**Structural Elements:**

* <html>: Root element of an HTML document.
* <head>: Contains metadata and links to scripts and styles.
* <title>: Sets the title of the document.
* <body>: Contains the content of the document.
* <header>: Defines a header section for a document or section.
* <footer>: Defines a footer for a document or section.
* <main>: Specifies the main content of the document.
* <section>: Defines sections in a document.
* <article>: Defines independent, self-contained content.
* <aside>: Defines content aside from the main content.
* <nav>: Defines navigation links.

**Content Elements:**

* <h1> to <h6>: Heading tags.
* <p>: Paragraph.
* <div>: Generic container for content.
* <span>: Inline container for text.
* <a>: Anchor/link.
* <img>: Image.
* <ul>: Unordered list.
* <ol>: Ordered list.
* <li>: List item.
* <table>: Table.
* <tr>: Table row.
* <th>: Table header cell.
* <td>: Table data cell.
* <form>: Form for user input.
* <input>: Input field.
* <button>: Button.
* <label>: Label for form element.
* <textarea>: Multi-line text input.
* <select>: Drop-down list.
* <option>: Option in a drop-down list.

**Media Elements:**

* <audio>: Embeds sound content.
* <video>: Embeds video content.
* <source>: Specifies multiple media resources for media elements (<audio>, <video>).
* <track>: Specifies text tracks for media elements.

**Semantics and Metadata:**

* <meta>: Metadata about the HTML document.
* <link>: Links to external resources like stylesheets.
* <style>: Embeds CSS within the HTML document.
* <script>: Embeds or references JavaScript.

**Commonly Used CSS Properties**

**Box Model:**

* width: Sets the width of an element.
* height: Sets the height of an element.
* padding: Sets the padding space inside an element.
* margin: Sets the margin space outside an element.
* border: Sets the border around an element.

**Typography:**

* font-family: Specifies the font for text.
* font-size: Sets the size of the font.
* font-weight: Sets the weight (boldness) of the font.
* color: Sets the color of the text.
* text-align: Sets the horizontal alignment of text.
* line-height: Sets the height of lines of text.

**Background:**

* background-color: Sets the background color of an element.
* background-image: Sets a background image for an element.
* background-size: Sets the size of the background image.
* background-position: Sets the initial position of the background image.

**Positioning:**

* position: Specifies the type of positioning for an element (static, relative, absolute, fixed, sticky).
* top, right, bottom, left: Specifies the position of an element.
* z-index: Sets the stack order of an element.

**Display and Visibility:**

* display: Specifies the display behavior of an element (block, inline, flex, grid, etc.).
* visibility: Specifies whether an element is visible or hidden.
* overflow: Specifies what happens if content overflows an element's box (visible, hidden, scroll, auto).

**Flexbox:**

* display: flex: Enables flexbox layout for an element.
* flex-direction: Specifies the direction of flex items (row, column, etc.).
* justify-content: Aligns flex items along the main axis.
* align-items: Aligns flex items along the cross axis.

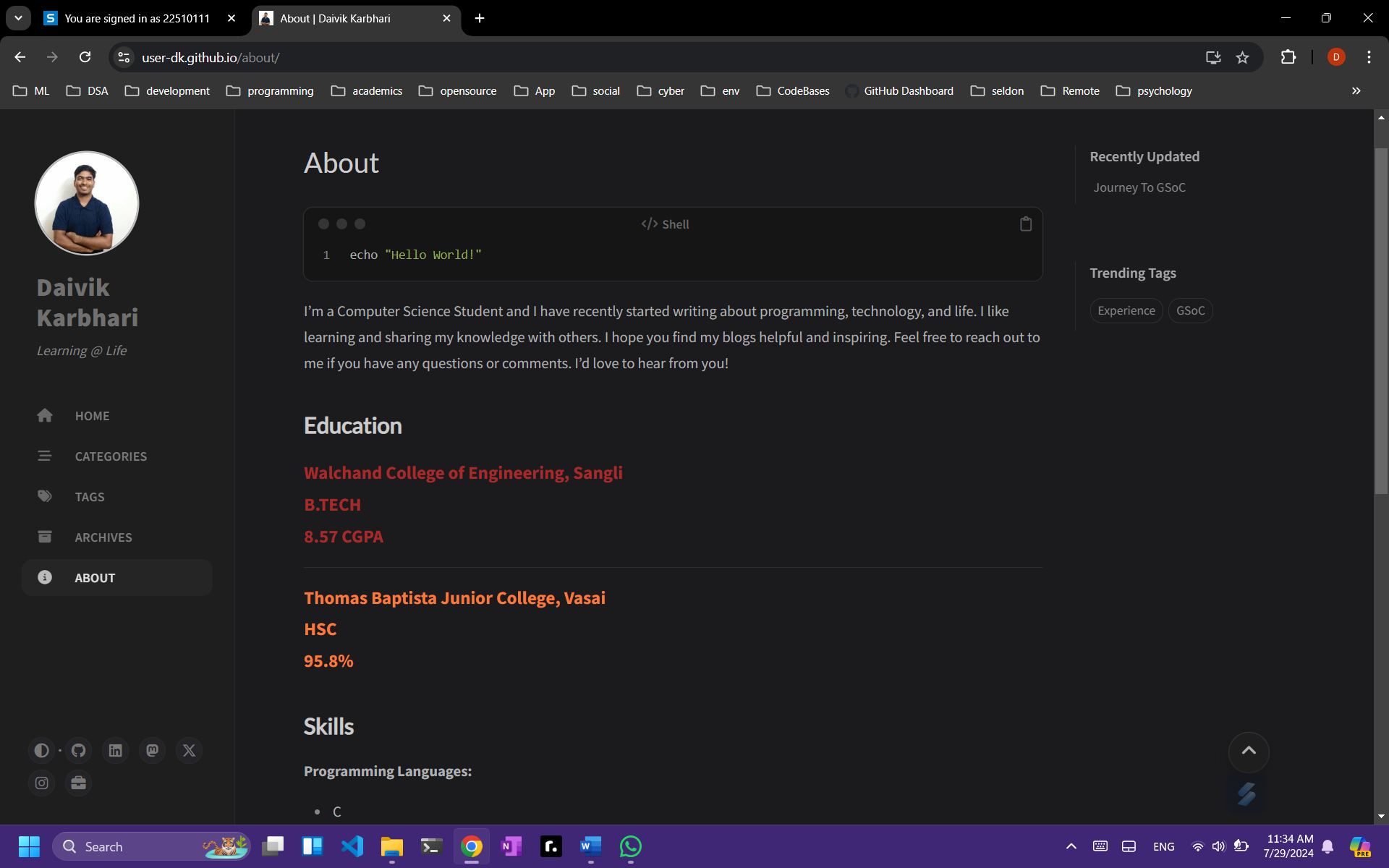
**Grid:**

* display: grid: Enables grid layout for an element.
* grid-template-columns: Defines the columns of the grid.
* grid-template-rows: Defines the rows of the grid.
* gap: Sets the gap between grid items.

**Animations and Transitions:**

* transition: Specifies the transition effect for properties.
* transform: Applies a transformation (rotate, scale, etc.) to an element.
* animation: Defines keyframe animations.

1. 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



Note:

1. Create a **document** of the above website with screenshots.

2. Scan the document and **create a pdf file** with **“ExamSeatNum\_P#PS#” as its name**.

3. Upload the file on the **WCE Moodle** before the given deadline.