MODULE 1: introduction to web technology(HTML/ CSS)

🔴 indicates VVIP

WEB ESSENTIALS

Q.1) write short note on Server, client and Communication.

⇒

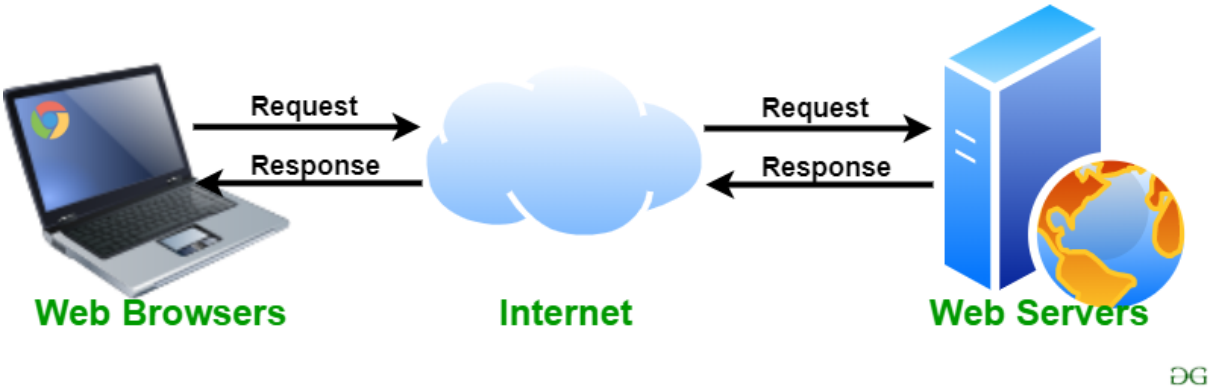
Server

A server is a computer or software system that provides resources, services, or functionality to clients over a network.

Servers are specialized for handling requests, processing data, and sending responses back to clients, making them a core part of network infrastructure.

Types of servers include:

- **Web Server**: Hosts websites and serves web pages to clients (browsers). For example, Apache and Nginx are popular web servers.
  When a user accesses a website, the web server processes this request and delivers HTML, CSS, JavaScript, and other content to the client.
- **Database Server**: Manages databases and responds to queries from client applications. For instance, MySQL and PostgreSQL are database servers that allow applications to store, retrieve, and manipulate data.
- **Mail Server**: Manages and distributes email. When a user sends an email, the email client (like Outlook) connects to the mail server to send it to the recipient's mail server. The recipient's email client will then retrieve it from the server.



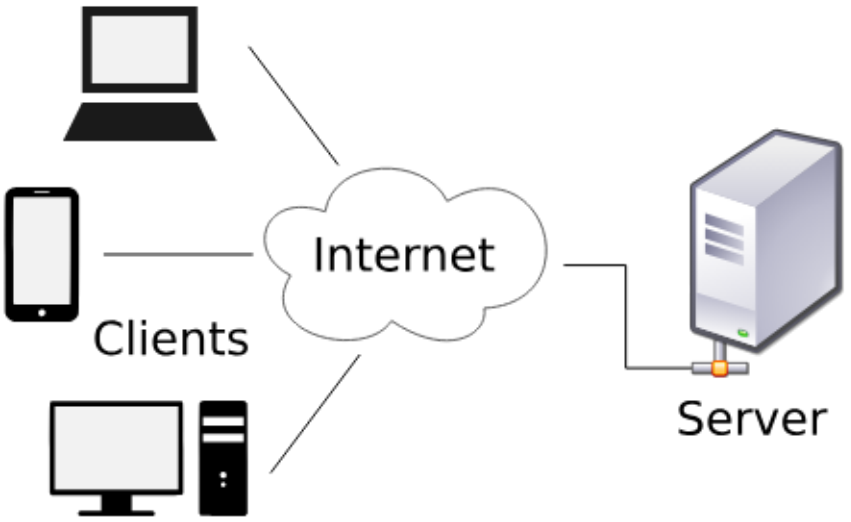
Client

A client is any device or software application that requests resources or services from a server. Clients could be web browsers, mobile applications, or desktop software. Clients initiate the communication process by sending requests to servers. Some common examples include:

- **Web Browser**: A browser (like Chrome or Firefox) is a client that requests web pages from a web server.
- **Email Client**: An application like Outlook or Gmail that requests and manages emails from a mail server.

Example: When you type a URL in your browser, the browser (client) sends a request to the server hosting that website.

The server processes this request and sends back the necessary HTML, CSS, and JavaScript files for the browser to render the page.



Communication

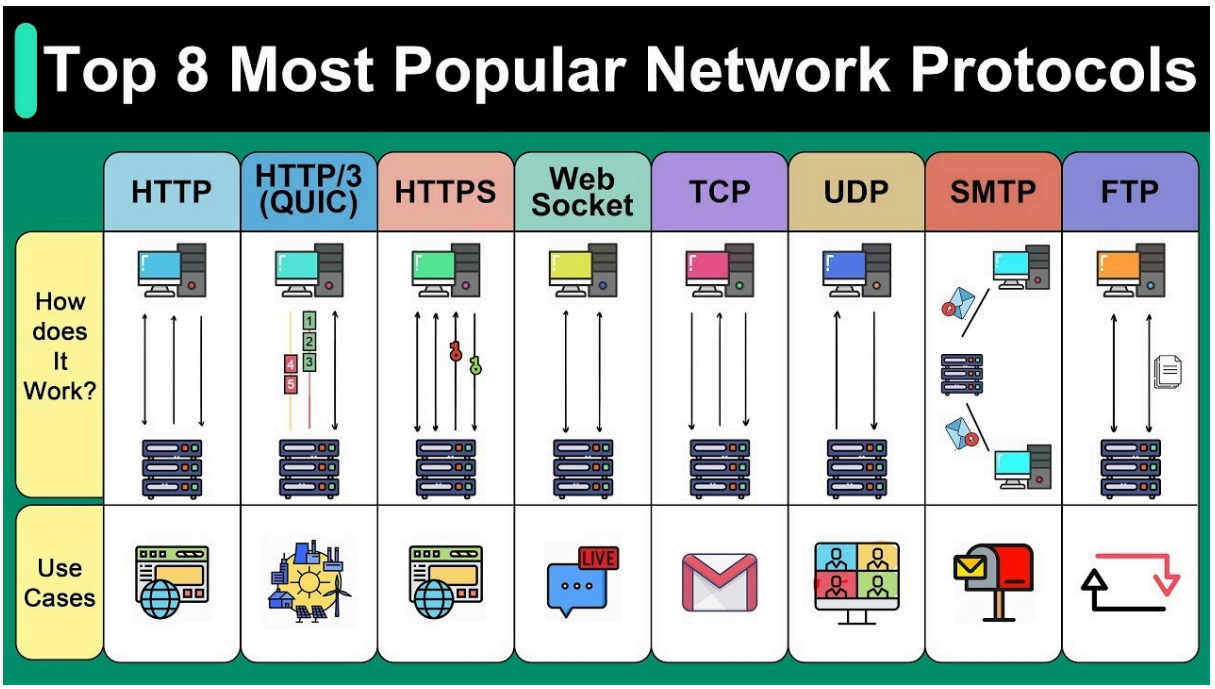
Communication refers to the data exchange process between a client and server. This interaction is facilitated through standard protocols, which define the structure and handling of data, ensuring that both client and server can interpret the messages correctly. Key components of this communication include:

1. **Request**: The client initiates a request, such as asking for a web page or database record.
2. **Processing**: The server processes this request, retrieves the necessary data or action, and prepares a response.
3. **Response**: The server sends back the requested information, which the client receives and displays or processes.

Example: For web browsing, uthis communication uses the HTTP protocol. The client sends an HTTP request to the web server, which responds with an HTML page, allowing the client to render and display the page to the user.

Q.2) explain the basic internet protocol, which are essential for transferring data and sending email.

⇒



Internet protocols are sets of rules that dictate how data is sent, received, and structured across a network.

The main protocols for data transfer and email are:

1. IP (Internet Protocol)

- **Purpose**: Handles addressing and routing data packets across networks.
- **How It Works**: IP assigns each device a unique IP address.
  When data is sent from one device to another, IP handles the packet's path, ensuring it reaches the correct address.
- **Example**: IP addresses like `192.168.1.1` or IPv6 addresses like `2001:0db8:85a3:0000:0000:8a2e:0370:7334` help identify devices uniquely on a network.

2. TCP (Transmission Control Protocol)

- **Purpose**: Ensures reliable and ordered delivery of data packets.
- **How It Works**: TCP establishes a connection between the client and server. It checks that all packets are received and in the correct order, re-transmitting any missing packets.

- **Example**: When loading a webpage, TCP verifies that all components (text, images) are fully loaded, in the correct order, for accurate rendering.

3. UDP (User Datagram Protocol)

- **Purpose**: Provides faster, but less reliable, data transfer.
- **How It Works**: UDP sends packets without confirming receipt, making it faster than TCP but less reliable.
- **Example**: Online gaming and streaming often use UDP, where speed is more important than perfect accuracy.

4. HTTP (HyperText Transfer Protocol)

- **Purpose**: Transfers web data (HTML files, images, etc.) from a server to a client.
- **How It Works**: HTTP operates as a request-response protocol, where the client requests a resource, and the server responds with the resource data.
- **Example**: When visiting a webpage, the browser sends an HTTP request to the server, which returns the webpage content.

5. SMTP (Simple Mail Transfer Protocol)

- **Purpose**: Responsible for sending emails between email servers.
- **How It Works**: SMTP transfers outgoing emails from the client's email server to the recipient's server.
- **Example**: When you send an email via Gmail, SMTP moves your message from Gmail's server to the recipient's server.

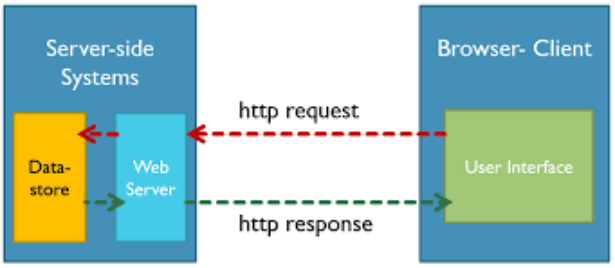
6. IMAP (Internet Message Access Protocol) and POP3 (Post Office Protocol 3)

- **Purpose**: IMAP and POP3 retrieve emails from the server for clients.
- **How They Work**:
  - **IMAP** allows clients to manage and access emails directly on the server, suitable for accessing mail from multiple devices.
  - **POP3** downloads emails from the server to the client, often deleting the original emails on the server after download.

- **Example**: Gmail and Outlook use IMAP, allowing you to access emails on different devices and sync any actions (like deleting or marking as read) across devices.

Q.3) What is HTTP describe structure of HTTP request and response message.

⇒



HTTP (HyperText Transfer Protocol) is the primary protocol for transferring hypertext (web pages) over the internet.

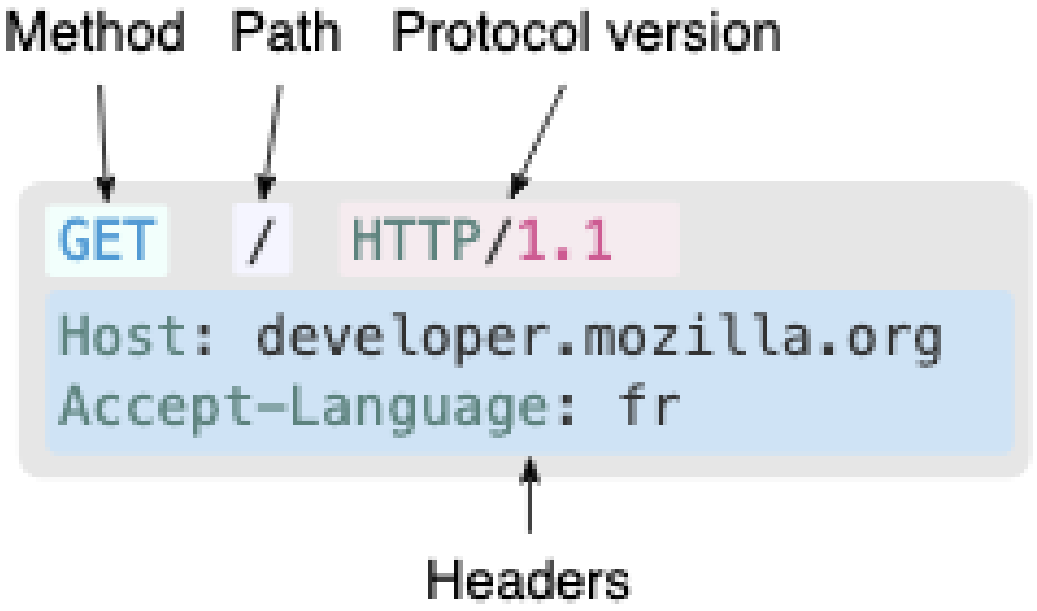
It is a request-response protocol, where the client (typically a web browser) sends an HTTP request, and the server replies with an HTTP response.

It operates over TCP and is stateless, meaning each request is independent of previous ones.

Structure of an HTTP Request Message

An HTTP request message includes the following components:

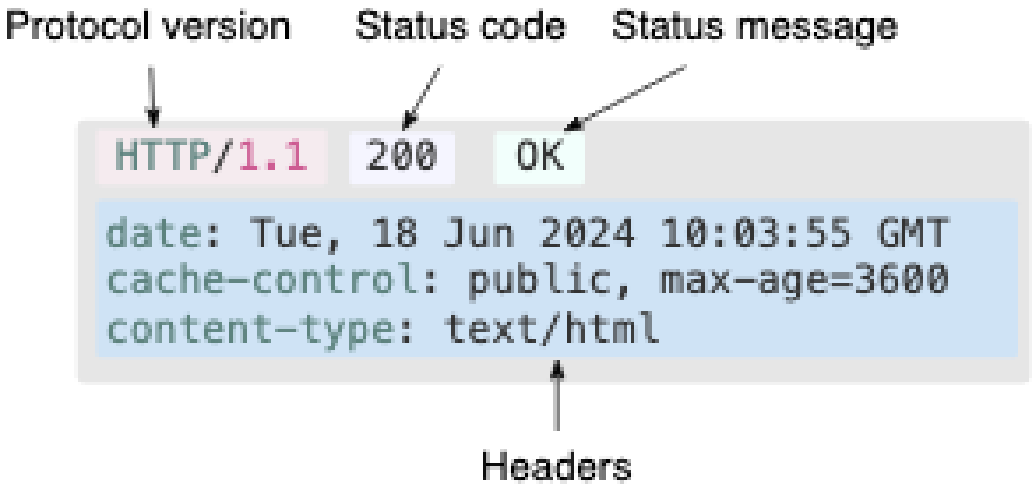
1. **Request Line**: Specifies the request method, URL, and HTTP version.
   - **Example**: `GET /index.html HTTP/1.1`
   - **Explanation**:
     - `GET`: The HTTP method (can also be `POST`, `PUT`, `DELETE`, etc.)
     - `/index.html`: The URL (path to the resource requested)
     - `HTTP/1.1`: The HTTP version used



1. **Headers**: Contain key-value pairs providing additional information about the request.
    - **Example**:

      ```
      Host: www.example.com
      User-Agent: Mozilla/5.0
      Accept: text/html
      ```

    - **Explanation**:
        - `Host`: Specifies the domain name of the server.
        - `User-Agent`: Information about the client making the request (e.g., browser type).
        - `Accept`: Indicates the types of content the client can process.



1. **Blank Line**: A blank line separates headers from the body of the request.
2. **Body (Optional)**: Contains data sent by the client, typically used with `POST` or `PUT` requests.
   - **Example**:

     ```
     { "username": "user123", "password": "securepass" }
     ```

Structure of an HTTP Response Message

An HTTP response message from the server includes these components:

1. **Status Line**: Indicates the HTTP version, status code, and status message.
   - **Example**: `HTTP/1.1 200 OK`
   - **Explanation**:
     - `HTTP/1.1`: The HTTP version used.
     - `200`: Status code (e.g., 200 for success, 404 for not found, 500 for server error).
     - `OK`: Status message that describes the code.
2. **Headers**: Contain additional information about the response, such as content type and length.
   - **Example**:

     ```
     Content-Type: text/html
     ```

```
Content-Length: 1234
```

- **Explanation**:
  - `Content-Type`: Specifies the type of content (e.g., `text/html` or `application/json`).
  - `Content-Length`: The size of the response body in bytes.

3. **Blank Line**: Separates the headers from the response body.
4. **Body (Optional)**: Contains the data requested by the client, such as an HTML page, JSON data, or image file.
   - **Example (HTML)**:

```
<html>
  <head><title>Example Page</title></head>
  <body><h1>Welcome to Example.com!</h1></body>
</html>
```

This structure helps browsers understand and render content correctly, allowing smooth interaction between clients and web servers.

HTML5

Q.4) differentiate between HTML and HTML5

⇒

Difference Between HTML and HTML5

HTML5 is an improvement over the original HTML standard, adding many new features and making web pages more versatile and user-friendly. Here's a closer look at the differences:

Feature	HTML	HTML5
Doctype Declaration	HTML4 and earlier versions required long and complex	HTML5 simplifies it with `<!DOCTYPE html>`, making it easier and consistent.

	doctype declarations, often confusing to set correctly.	
Audio and Video	Audio and video were embedded using external plugins like Flash, which were not natively supported.	HTML5 has built-in support for `<audio>` and `<video>` tags, allowing easier multimedia embedding without third-party plugins.
APIs	HTML had very limited support for APIs, requiring JavaScript workarounds.	HTML5 includes powerful new APIs, like Geolocation, Web Storage, and Canvas, for interactive graphics, allowing direct interactions with the webpage.
Browser Compatibility	HTML worked across older browsers but lacked uniform functionality across them.	HTML5 introduced cross-browser compatibility for modern web apps, ensuring they perform consistently across various devices and browsers.
Semantic Elements	Semantic meaning was implied through `<div>` and `<span>`, which provided no meaning or structure.	HTML5 introduces semantic elements like `<header>`, `<footer>`, `<article>`, and `<section>`, which give structure and meaning to the content, improving SEO and accessibility.

Q.5) what is the difference between class selector and ID selector?

⇒

Both class and ID selectors are used to style elements in CSS, but they serve different purposes and are used in different contexts.

- **Class Selector (`.class`)**:
  - **Purpose**: Used to apply styles to multiple elements that share a common property, such as buttons or card elements.
  - **Definition**: In CSS, a class selector starts with a `.` followed by the class name. You can then apply this class to any number of elements on the page.
  - **Example**:

```
<style>
 .highlight {
  color: blue;
```

```
    font-weight: bold;
  }
</style>

<p class="highlight">This paragraph is highlighted.</p>
<p class="highlight">Another highlighted paragraph.</p>
```

- **ID Selector (`#id`)**:
  - **Purpose**: Used to apply styles to a unique element on a page. It's used for single, specific elements (e.g., a header or main content container).
  - **Definition**: An ID selector starts with a `#` in CSS, followed by the ID name. Only one instance of a specific ID should appear in a document.
  - **Example**:

```
<style>
  #main-header {
    background-color: gray;
    color: white;
  }
</style>

<div id="main-header">Main Header</div>
```

Key Differences:

- **Uniqueness**: An ID is unique and should be used only once per page, while a class can be used multiple times.
- **Specificity**: ID selectors have higher specificity than class selectors, so styles applied via an ID will override class styles if there's a conflict.

Q.5) Explain any five semantic tag of HTML5 with example.

⇒

Semantic tags in HTML5 describe the purpose of the content they enclose, making it easier for search engines, developers, and screen readers to

understand and navigate web content. Here are five commonly used semantic tags:

1. `<header>`: Defines a header for a document or section, often containing navigation, logo, or introductory information.

```
<header>
  <h1>My Website</h1>
  <nav>
    <a href="#home">Home</a>
    <a href="#services">Services</a>
    <a href="#contact">Contact</a>
  </nav>
</header>
```

2. `<nav>`: Represents navigation links in a page. It is typically used for primary site navigation but can be used anywhere you need a set of navigational links.

```
<nav>
  <ul>
    <li><a href="#about">About</a></li>
    <li><a href="#portfolio">Portfolio</a></li>
    <li><a href="#contact">Contact</a></li>
  </ul>
</nav>
```

3. `<article>`: Contains independent, self-contained content like a blog post, news article, or forum post. It should make sense even if taken out of the context of the page.

```
<article>
  <h2>Understanding HTML5 Semantic Tags</h2>
  <p>HTML5 introduces several new tags to improve the structure of w
eb documents...</p>
</article>
```

4. `<section>`: Defines a generic section within a document, often containing a group of related content or a thematic grouping of content.

```
<section>
  <h2>Our Services</h2>
  <p>Details about our services go here.</p>
</section>
```

5. `<footer>`: Represents the footer of a document or section. It typically contains copyright information, contact details, or navigation links.

```
<footer>
  <p>&copy; 2023 My Website. All rights reserved.</p>
</footer>
```

Using these tags improves the document's structure, making it clearer to browsers, search engines, and assistive technologies.

Q.6) Explain <audio> and <video> control of HTML5 with appropriate example.

⇒

HTML5 introduced native tags for handling multimedia content, allowing audio and video files to be embedded directly into HTML without third-party plugins.

- `<audio>` **Tag**: Allows audio playback with built-in controls.
  - **Attributes**:
    - `controls`: Adds playback controls like play/pause and volume.
    - `autoplay`: Starts playing automatically when the page loads.
    - `loop`: Repeats the audio after it finishes.
  - **Example**:

```
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
```

- `<video>` **Tag**: Allows video playback with built-in controls.

- **Attributes**:
  - `controls`: Adds playback controls like play/pause, volume, and fullscreen.
  - `width` and `height`: Specifies dimensions for the video.
  - `poster`: Sets an image displayed before the video starts.
  - `autoplay`, `loop`, and `muted` are also commonly used.
- **Example**:

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  Your browser does not support the video element.
</video>
```

These tags simplify multimedia usage and allow customization for video and audio presentations on websites.

Q.7) explain the working of rowspan and colspan of table when used in HTML with suitable example

⇒

HTML tables support `rowspan` and `colspan` attributes, which allow cells to span across multiple rows or columns.

- `rowspan`: Makes a cell extend vertically over multiple rows.
  - **Usage**: When you need a single cell to cover multiple rows.
  - **Example**:*Explanation*: The "Name" cell spans two rows, covering both rows in the first column.

```
<table border="1">
  <tr>
    <td rowspan="2">Name</td>
    <td>Age</td>
  </tr>
  <tr>
    <td>25</td>
```

```
  </tr>
</table>
```

- `colspan`: Makes a cell extend horizontally across multiple columns.
  - **Usage**: When you want a single cell to span several columns.
  - **Example**:*Explanation*: The "Name" cell spans across two columns, covering the entire first row horizontally.

```
<table border="1">
  <tr>
    <td colspan="2">Name</td>
  </tr>
  <tr>
    <td>Age</td>
    <td>25</td>
  </tr>
</table>
```

CSS3

Q.8) explain how shadow effect can be applied on text using CSS with suitable example.

⇒

In CSS, the `text-shadow` property is used to apply a shadow effect to text. This property allows you to add one or more shadows to text elements by specifying the shadow's horizontal and vertical offset, blur radius, and color.

Syntax:

```
text-shadow: h-shadow v-shadow blur-radius color;
```

- **h-shadow**: Horizontal shadow (right if positive, left if negative).
- **v-shadow**: Vertical shadow (down if positive, up if negative).

- **blur-radius**: Optional. The blur radius of the shadow (larger values mean more blur).
- **color**: The color of the shadow.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    h1 {
      color: #333;
      text-shadow: 5px 10px 15px red; /* Shadow effect */
    }
  </style>
</head>
<body>
  <h1>Shadowed Text Example</h1>
</body>
</html>
```

Explanation:

- In this example, the `h1` element's text has a shadow effect.
- `5px` right and `10px` down for the shadow offset, `15px` for the blur effect, and `red` for the shadow color.

You can also add multiple shadows by separating each shadow with a comma:

```
text-shadow: 10px 10px 15px red, -2px -2px 5px blue;
```

Q.9) explain CSS3 animation with example.

⇒

CSS3 animations allow elements to transition between different styles smoothly over a period of time. Animations are defined using `@keyframes`, which specify the styles at various points in the animation timeline.

Key Properties:

- **@keyframes**: Defines the animation steps.
- **animation-name**: Names the animation defined by `@keyframes`.
- **animation-duration**: Specifies how long the animation takes.
- **animation-timing-function**: Defines the pace of the animation (e.g., `linear`, `ease`).
- **animation-delay**: Adds a delay before the animation starts.
- **animation-iteration-count**: Specifies the number of times the animation should repeat.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    /* Define the animation */
    @keyframes slide {
      from { margin-left: 0%; }
      to { margin-left: 50%; }
    }

    /* Apply the animation to an element */
    .animated-box {
      width: 100px;
      height: 100px;
      background-color: #3498db;
      animation-name: slide;
      animation-duration: 2s;
      animation-timing-function: ease-in-out;
      animation-iteration-count: infinite;
    }
  </style>
</head>
<body>
  <div class="animated-box"></div>
```

```
</body>
</html>
```

Explanation:

- The `@keyframes slide` rule defines a sliding effect for the `.animated-box`.
- The box moves from a margin of `0%` (starting position) to `50%` (halfway across the container).
- The animation lasts `2 seconds`, eases in and out, and repeats infinitely.

Q.10) list and explain the three ways to add the style sheet (CSS) to an HTML webpage with suitable example.

⇒

CSS can be applied to HTML in three ways: **inline**, **internal**, and **external** styles. Each method is suitable for different scenarios.

1. Inline CSS

- **Definition**: CSS is applied directly to HTML elements using the `style` attribute.
- **Use**: Best for styling a single element quickly, but not efficient for large projects.
- **Example**:

```
<h1 style="color: blue; font-size: 24px;">Hello, Inline CSS!</h1>
```

2. Internal CSS

- **Definition**: CSS is included within the `<style>` tag inside the `<head>` section of the HTML document.
- **Use**: Useful for styling a single page without creating an external stylesheet.

- **Example**:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    body {
      background-color: lightgray;
    }
    h1 {
      color: blue;
    }
  </style>
</head>
<body>
  <h1>Internal CSS Example</h1>
</body>
</html>
```

3. External CSS

- **Definition**: CSS is written in a separate `.css` file and linked to the HTML file.
- **Use**: Best for larger projects where multiple pages share the same styles.
- **Example**:
  1. **CSS File (styles.css)**:

```
body {
  background-color: lightgray;
}
h1 {
  color: blue;
}
```

  2. **HTML File (index.html)**:

```
<!DOCTYPE html>
<html>
```

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <h1>External CSS Example</h1>
</body>
</html>
```

Each method has its use cases, with external CSS being the most scalable and maintainable approach for multi-page websites.

Q.) what is inheritance in CSS explain CSS animation properties.

⇒Inheritance in CSS

Inheritance is a CSS feature that allows child elements to inherit styles from their parent elements. Some CSS properties are automatically inherited (like font family, color), while others are not (like padding, border).

- **Example**:
  In this example, the `p` element inside the `div` will inherit the `color` and `font-size` properties from its parent.

```
<style>
  div {
    color: blue;
    font-size: 20px;
  }
</style>
<body>
  <div>
    Parent Div Text
    <p>Child Paragraph Text (inherits color and font-size from div)</p
>
  </div>
</body>
```

CSS Animation Properties

CSS animations are controlled with several key properties:

- **animation-name**: Specifies the name of the `@keyframes` animation.
- **animation-duration**: Sets the duration of the animation (e.g., `2s`, `500ms`).
- **animation-timing-function**: Defines the speed curve of the animation (e.g., `ease`, `linear`, `ease-in-out`).
- **animation-delay**: Adds a delay before the animation begins (e.g., `1s`).
- **animation-iteration-count**: Defines the number of times the animation should play (e.g., `infinite` or a specific number).
- **animation-direction**: Specifies whether the animation should play forwards, backwards, or alternate (e.g., `normal`, `reverse`, `alternate`).

Example with Multiple Animation Properties:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    @keyframes bounce {
      0% { transform: translateY(0); }
      50% { transform: translateY(-20px); }
      100% { transform: translateY(0); }
    }

    .animated-element {
      width: 100px;
      height: 100px;
      background-color: coral;
      animation-name: bounce;
      animation-duration: 2s;
      animation-timing-function: ease;
      animation-delay: 1s;
      animation-iteration-count: infinite;
      animation-direction: alternate;
    }
  </style>
</head>
<body>
```

```
    <div class="animated-element"></div>
</body>
</html>
```

Explanation:

- **animation-name**: The animation defined by `@keyframes` is called `bounce`.
- **animation-duration**: Each cycle of the animation lasts `2 seconds`.
- **animation-timing-function**: The animation eases in and out.
- **animation-delay**: The animation starts after a `1-second` delay.
- **animation-iteration-count**: The animation repeats infinitely.
- **animation-direction**: The animation alternates, playing forwards then backwards.

This example creates a "bouncing" effect by moving the element up and down repeatedly.

CODE :

Q.11) write a code to drag an image from outside the box and drop it inside the box.

⇒

1. Drag and Drop an Image into a Box

To create a drag-and-drop effect where an image can be dragged from outside a box and dropped inside, we can use the HTML5 Drag and Drop API.

HTML and JavaScript Code:

```
<!DOCTYPE html>
<html>
<head>
```

```
    <style>
        #dropBox {
            width: 300px;
            height: 200px;
            border: 2px dashed #000;
            display: flex;
            align-items: center;
            justify-content: center;
            text-align: center;
            margin-top: 20px;
        }
        .draggable {
            width: 100px;
            cursor: grab;
        }
    </style>
</head>
<body>

<h2>Drag and Drop Example</h2>
<p>Drag the image below and drop it into the box.</p>

<img src="example.jpg" id="dragImage" class="draggable" draggable="tr
ue" alt="Draggable Image">

<div id="dropBox">Drop here</div>

<script>
    const dragImage = document.getElementById("dragImage");
    const dropBox = document.getElementById("dropBox");

    dragImage.addEventListener("dragstart", function(event) {
        event.dataTransfer.setData("text/plain", event.target.id);
    });

    dropBox.addEventListener("dragover", function(event) {
        event.preventDefault();
    });
```

```
    dropBox.addEventListener("drop", function(event) {
        event.preventDefault();
        const imageId = event.dataTransfer.getData("text/plain");
        const image = document.getElementById(imageId);
        dropBox.appendChild(image);
    });
</script>

</body>
</html>
```

Explanation:

- The `dragstart` event attaches data (the image ID) to the dragged image.
- The `dragover` event on the drop box allows the item to be dropped.
- The `drop` event transfers the image inside the `#dropBox` div.

Q.12) Create an external style sheet and link it to an HTML form. The style sheet should contain the following.

i)An header in text with red text colour and yellow background colour.

ii) a double lined (border) table.

iii) the table should have five rows and three columns.

iv) in the first column sr. number of the product, second column image with hyperlink and name of the product, and third column the description of the product.

⇒

First, create the external CSS file `styles.css` with the required styles, and then link it to an HTML form.

styles.css:

```css
/* Header styles */
h1 {
    color: red;
    background-color: yellow;
}

/* Table styles */
table {
    border: double;
    width: 100%;
    border-collapse: collapse;
}

table, th, td {
    border: 2px double black;
    padding: 8px;
    text-align: center;
}
```

HTML Code:

```html
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>Product List</h1>

<table>
    <tr>
        <th>Sr. No.</th>
        <th>Product</th>
        <th>Description</th>
    </tr>
    <tr>
```

```
      <td>1</td>
      <td><a href="product1.html"><img src="product1.jpg" alt="Product 1"
width="50"> Product 1</a></td>
      <td>Product 1 Description</td>
    </tr>
    <tr>
      <td>2</td>
      <td><a href="product2.html"><img src="product2.jpg" alt="Product
2" width="50"> Product 2</a></td>
      <td>Product 2 Description</td>
    </tr>
    <tr>
      <td>3</td>
      <td><a href="product3.html"><img src="product3.jpg" alt="Product
3" width="50"> Product 3</a></td>
      <td>Product 3 Description</td>
    </tr>
    <tr>
      <td>4</td>
      <td><a href="product4.html"><img src="product4.jpg" alt="Product
4" width="50"> Product 4</a></td>
      <td>Product 4 Description</td>
    </tr>
    <tr>
      <td>5</td>
      <td><a href="product5.html"><img src="product5.jpg" alt="Product
5" width="50"> Product 5</a></td>
      <td>Product 5 Description</td>
    </tr>
</table>

</body>
</html>
```

Explanation:

- `styles.css` defines styles for the header and table, with red text and yellow background for the header and a double-bordered table with centered

content.

Q.13) write an external style sheet and link it with HTML. The style sheet should include the following.

i) the webpage will have the background image “img1.jpg”.

ii) the table heading will have red background colour.

iii) background, colour of alternate paragraphs are different.

iv) the hyperlinks on the webpage will not have underline.

⇒

External Style Sheet Linked to HTML with Background and Styling

styles.css:

```
/* Set background image */
body {
    background-image: url('img1.jpg');
    background-size: cover;
}

/* Table header background color */
th {
    background-color: red;
    color: white;
}

/* Alternate paragraph background colors */
p:nth-of-type(odd) {
    background-color: yellow;
}
p:nth-of-type(even) {
    background-color: brown;
}
```

```css
/* Remove underline from hyperlinks */
a {
  text-decoration: none;
  color: blue;
}
```

HTML Code:

```html
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>Welcome to Our Webpage</h1>

<table>
  <tr>
    <th>Heading 1</th>
    <th>Heading 2</th>
  </tr>
  <tr>
    <td>Data 1</td>
    <td>Data 2</td>
  </tr>
  <tr>
    <td>Data 3</td>
    <td>Data 4</td>
  </tr>
</table>

<p>First paragraph with background color.</p>
<p>Second paragraph with different background color.</p>
<p>Third paragraph with background color.</p>

<a href="<https://www.example.com>">Visit Example</a>
```

```
</body>
</html>
```

Explanation:

- This code includes an external stylesheet that:
  - Sets a background image for the body.
  - Styles the table header background color to red.
  - Alternates paragraph background colors for visual variety.
  - Removes the underline from hyperlinks.