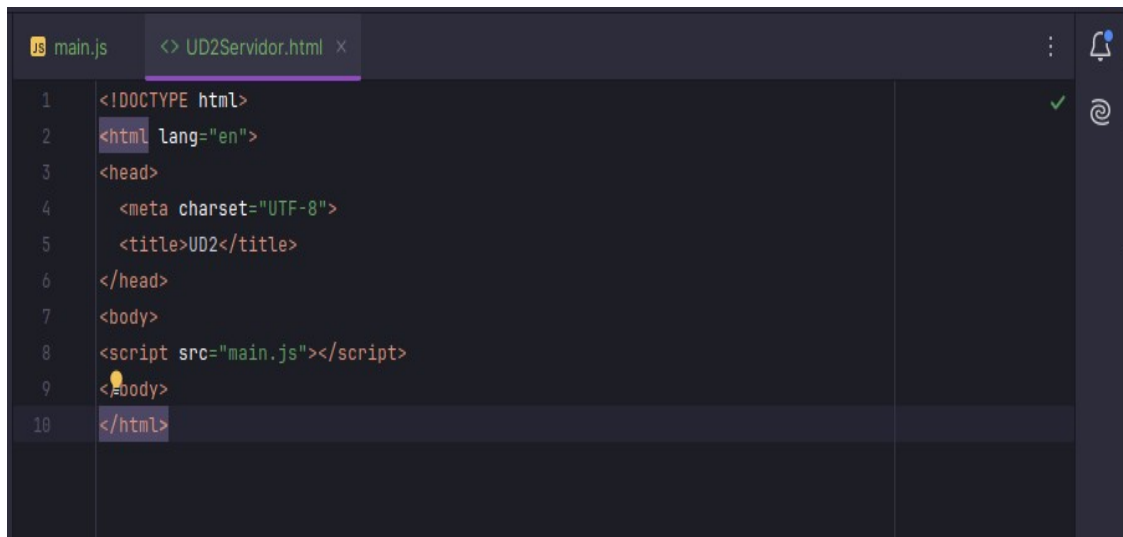


1.Inserción de Código en HTML:

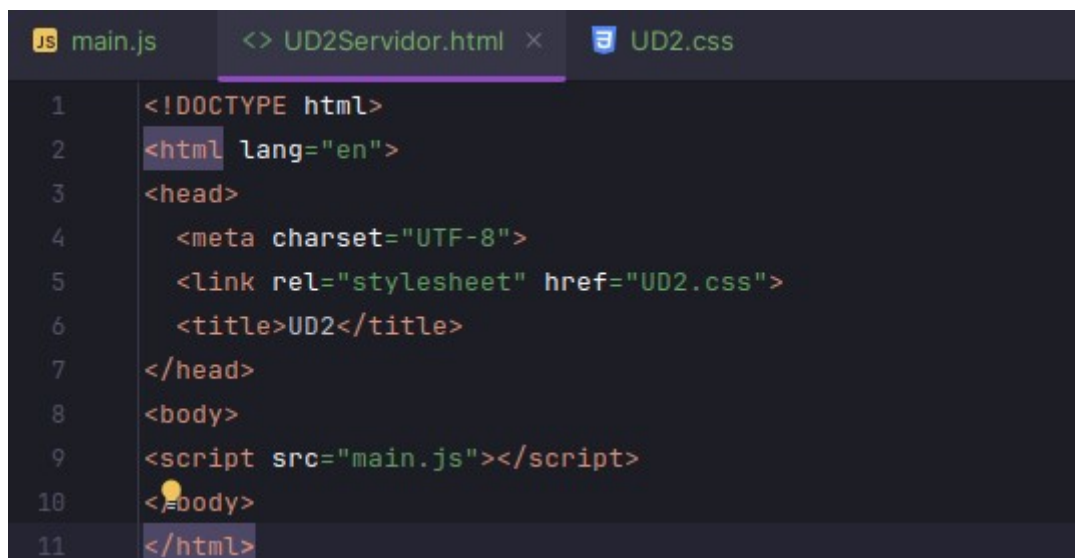
- Identifica y describe las etiquetas <script>, <link>, y <style> en HTML.
- Explica cómo se utilizan estas etiquetas para incorporar JavaScript y CSS en páginas web.

<script>: Se utiliza para incluir o vincular archivos JavaScript dentro de un documento HTML.
- Ejemplo :Esto inserta un archivo JavaScript externo llamado “main.j”.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>UD2</title>
6 </head>
7 <body>
8   <script src="main.js"></script>
9 </body>
10 </html>
```

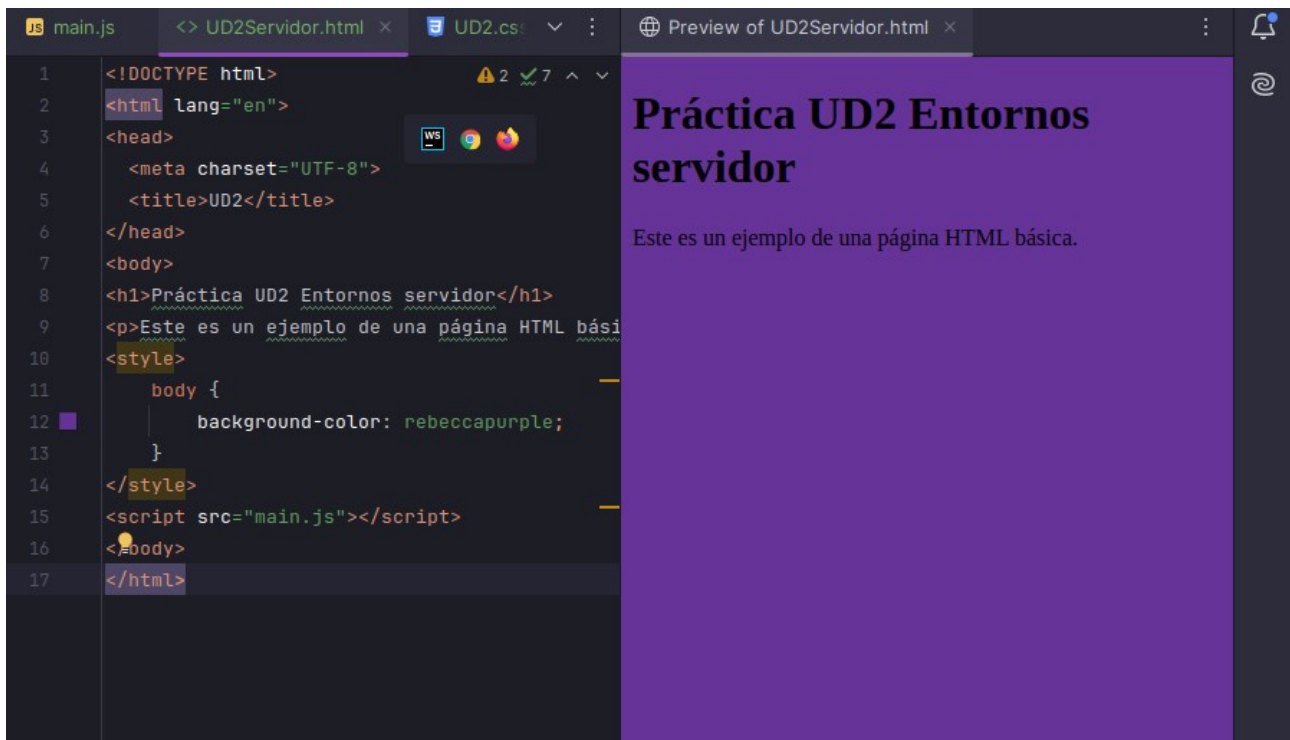
<link>: Se usa para vincular archivos externos de CSS. Generalmente, se coloca en la sección <head> del HTML.
- Ejemplo:Esto enlaza un archivo CSS llamado “UD2.css” para definir el estilo de la página.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <link rel="stylesheet" href="UD2.css">
6   <title>UD2</title>
7 </head>
8 <body>
9   <script src="main.js"></script>
10 </body>
11 </html>
```

`<style>`: Permite la inclusión de estilos CSS directamente dentro de un documento HTML, sin necesidad de un archivo CSS externo.

- Ejemplo:



2.Inserción de Código en Lenguajes del Lado del Servidor:

- Dibuja un esquema que muestre cómo PHP, ASP.NET, JSP y Java Servlets se integran en el desarrollo web.

| Tecnologia | Tipo de archivo | Sintaxis | Lengua | Función |
|------------|-----------------|---------------------------|-------------|---|
| PHP | .php | <?php ... ?> | PHP | Genera contenido dinámico que se incrusta directamente en HTML |
| ASP.NET | .aspx | <% ... %> o
<%= ... %> | C# o VB.NET | Es parte del ecosistema .NET de Microsoft, una opción popular en entornos empresariales, especialmente aquellos que requieren integración con otros productos de Microsoft. |
| JSP | .jsp | <% ... %> | Java | Conocida por su portabilidad y eficiencia en el manejo de solicitudes en el |

| | | | | |
|---------------|-------|-------------|------|--|
| | | | | servidor, ideal para sitios web y aplicaciones con lógica de negocio compleja. |
| Java Servlets | .java | Código java | Java | Conocida por su portabilidad y eficiencia en el manejo de solicitudes en el servidor, ideal para sitios web y aplicaciones con lógica de negocio compleja. |

- Incluye ejemplos de etiquetas y sintaxis específicas utilizadas en cada lenguaje para la inserción de código.

PHP:

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  <meta charset="UTF-8">
5  <title>UD2</title>
6  </head>
7  <body>
8
9
10 <h1>Práctica UD2 Entornos servidor</h1>
11 <p>Este es un ejemplo de una página HTML básica.</p>
12
13 <?php
14 $mensaje = "Bienvenido a nuestra práctica de entornos servidor.";
15 echo "<p>$mensaje</p>";
16 ?>
17

```

ASP:

```
4 <html lang="en">
5 <head>
6   <meta charset="UTF-8">
7   <title>UD2</title>
8   <style>
9     body {
10       background-color: rebeccapurple;
11     }
12   </style>
13 </head>
14 <body>
15   <h1>Práctica UD2 Entornos servidor</h1>
16   <p>Este es un ejemplo de una página ASP.NET básica.</p>
17
18   <p>
19     <%
20       // Generar contenido dinámico con C#
21       string mensaje = "Bienvenido a nuestra práctica de entornos servidor.";
22       Response.Write(mensaje);
23     %>
24   </p>
25
26   <script src="main.js"></script>
27 </body>
```

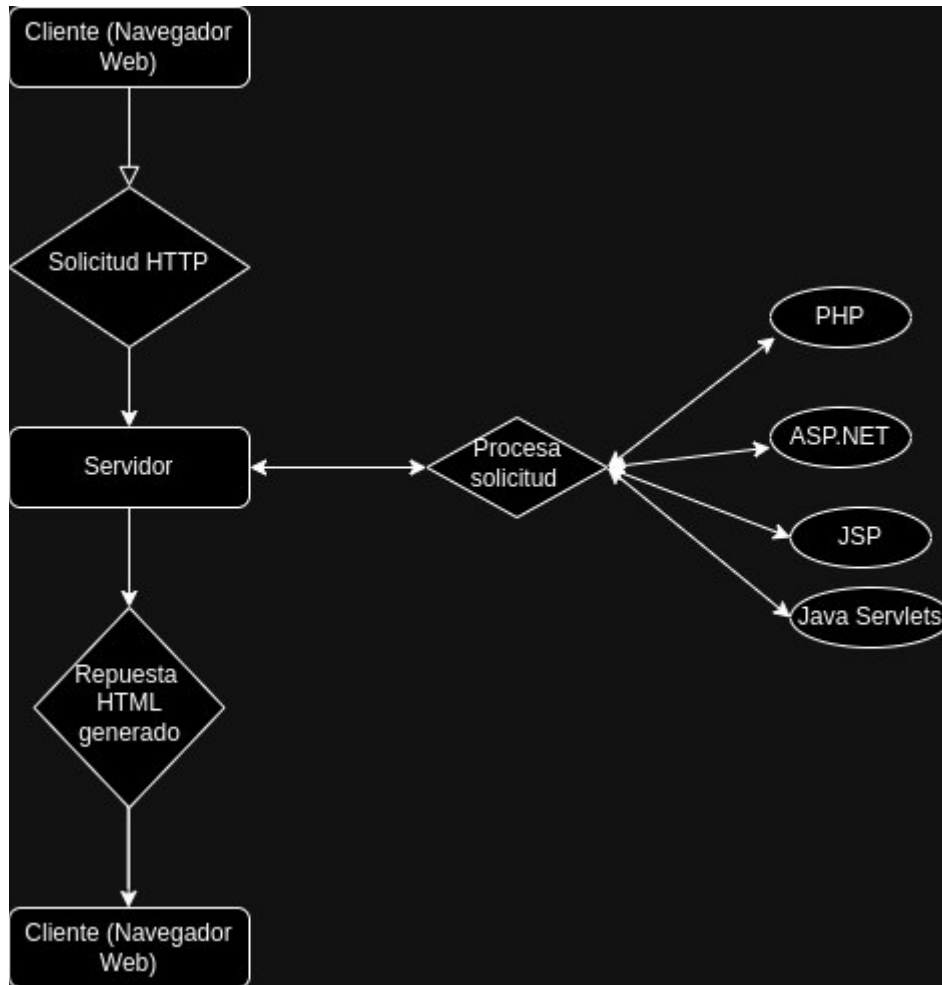
JSP:

```
11   </style>
12 </head>
13 <body>
14   <h1>Práctica UD2 Entornos servidor</h1>
15   <p>Este es un ejemplo de una página JSP básica.</p>
16
17   <p>
18     <%
19       // Generar contenido dinámico con Java
20       String mensaje = "Bienvenido a nuestra práctica de entornos servidor.";
21       out.println(mensaje);
22     %>
23   </p>
24
25   <script src="main.js"></script>
26 </body>
27 </html>
28 |
```

En Java Servlets habría que hacer una clase para el servlet donde se integre el código java y luego configurar un archivo web.xml para añadir tal clase.

3.Interacción Cliente-Servidor:

- Elabora un diagrama que ilustre el flujo de datos entre el cliente y el servidor, destacando el papel de la inserción de código en este proceso.



4.Mejores Prácticas:

- Enumera y explica las mejores prácticas en la inserción de código, incluyendo claridad, mantenimiento, seguridad y eficiencia.

1. Claridad:

- Nombres Descriptivos: Utiliza nombres de variables, funciones y clases que sean descriptivos, evita nombres demasiado largos. Por ejemplo, “calcularRadio” es mejor que “calculo1”.
- Comentarios Útiles: Añade comentarios para explicar qué hace el código y por qué.
- Formato Consistente: Usa el espaciado, indentación como ayuda a que el código sea más fácil de leer y entender.

2. Mantenimiento:

- Modularidad: Divide el código en funciones o métodos pequeños. Esto facilita la reutilización y mantenimiento.
- Uso de Frameworks y Bibliotecas: Evita escribir todo desde cero. Acelera el desarrollo y proporciona soluciones a errores ya comprobados, al integrar como una plantilla predefinida.
- Control de Versiones: Usa sistemas de control de versiones , por ejemplo Git, así puedes controlar los cambios hechos en el proyecto y permite la colaboración de otros desarrolladores en el proyecto a la vez.

3. Seguridad:

- Validación de Entrada: Se usa para prevenir ataques como inyección de SQL, cross-site scripting (XSS) .
- Uso de Autenticación y Autorización: Implementa mecanismos de autenticación como doble factor de autenticación y asegura de que los usuarios solo tengan acceso a los recursos que les corresponden.

4. Eficiencia:

- Optimización de Algoritmos Elige algoritmos y estructuras de datos adecuados para tus necesidades.
- Evita Códigos Repetitivos: Usa funciones y métodos para evitar la duplicación de código.

5. Pruebas y Validación:

- Pruebas Unitarias: validamos que cada parte del código funcione correctamente, detectando errores en fases tempranas del desarrollo.
- Integración Continua: con la CD (entrega continua) se despliega automáticamente el proyecto y con la CI (integración continua) se automatizan las pruebas unitarias y de integración para que se vayan realizando a cada código generado nuevo.