1. Introducción al Desarrollo Web

Ideas principales:

- **Necesidad de una arquitectura escalable:** Al iniciar un proyecto web, se necesita una arquitectura flexible y escalable (como servlets y JSP) para destacar frente a la competencia.
- **Cliente-Servidor:** Es esencial entender cómo interactúan los clientes web (navegadores) y los servidores web.
- **Objetivo:** Crear aplicaciones web accesibles globalmente para alcanzar el éxito.

Imágenes:

- Diagrama simple de la arquitectura cliente-servidor.
- Imagen de un navegador web interactuando con un servidor.

2. Funciones del Servidor Web

Ideas principales:

- **Procesamiento de solicitudes:** Un servidor web recibe solicitudes de los clientes, encuentra los recursos solicitados y los devuelve.
- **Errores comunes:** Ejemplo de error 404 cuando el recurso solicitado no se encuentra.
- **Hardware vs. Software:** Diferenciación entre el servidor físico (hardware) y la aplicación del servidor web (software).

Imágenes:

- Ejemplo de un mensaje de error 404.
- Diagrama que muestra la diferencia entre hardware y software en un servidor.

3. Funciones del Cliente Web

Ideas principales:

- **Solicitud y visualización:** El cliente (navegador web) solicita recursos al servidor y muestra los resultados al usuario.
- **Interpretación de HTML:** El navegador interpreta el HTML para renderizar páginas web.

Imágenes:

- Captura de pantalla de un navegador renderizando una página web.
- Diagrama de cómo un navegador procesa una solicitud.

4. HTML y HTTP

Ideas principales:

- **HTML:** Lenguaje de marcado utilizado para estructurar y presentar contenido web.
- **HTTP:** Protocolo que facilita la comunicación entre cliente y servidor mediante un esquema de solicitud y respuesta.

Imágenes:

- Código HTML básico y su renderización en un navegador.
- Diagrama de la estructura de un mensaje HTTP.

5. Métodos HTTP: GET y POST

Ideas principales:

- **GET:** Solicita un recurso del servidor, es simple y limitado en la cantidad de datos que puede enviar.
- **POST:** Solicita un recurso y además permite enviar datos adicionales al servidor, ideal para formularios largos o sensibles.
- **Consideraciones:** Diferencias clave entre GET y POST, como limitaciones de tamaño y visibilidad de datos.

Imágenes:

- Ejemplo de una URL con parámetros GET.
- Diagrama comparativo de solicitudes GET y POST.

6. Respuestas HTTP y MIME Types

Ideas principales:

- **Cabeceras y cuerpo en la respuesta HTTP:** La cabecera informa sobre el protocolo y el tipo de contenido, mientras que el cuerpo contiene el contenido real como HTML.
- **MIME Types:** Clasificación del tipo de contenido en la respuesta HTTP (e.g., text/html, image/jpeg).

Imágenes:

- Ejemplo visual de una respuesta HTTP con cabeceras y cuerpo.
- Lista de tipos MIME comunes.

7. Concepto de URL

Ideas principales:

- **Estructura de la URL:** Dirección única para cada recurso en la web, compuesta por protocolo, puerto y recurso.
- **Importancia de los puertos:** Los puertos identifican aplicaciones específicas en el servidor, como HTTP en el puerto 80.

Imágenes:

- Diagrama desglosando la estructura de una URL.
- Tabla de puertos comunes y sus aplicaciones.

8. Estructura de Directorios en Apache

Ideas principales:

- **Organización de un sitio web en Apache:** Ejemplo de estructura de directorios para un sitio web que aloja múltiples aplicaciones.

- **Uso de Apache y Tomcat:** Breve introducción a Apache como servidor web popular y Tomcat como contenedor de servlets.

Imágenes:

- Ejemplo de estructura de directorios de un sitio web.
- Logo de Apache y Tomcat.