Funcionamiento de las Arquitecturas Web

1. Arquitectura Cliente-Servidor

¿Cómo funciona?

1. El **cliente** (navegador o app) envía una **solicitud** al servidor (HTTP/HTTPS).
2. El **servidor** recibe la solicitud.
3. Procesa la lógica necesaria (validaciones, consultas).
4. Accede a la **base de datos** si es necesario.
5. Devuelve una **respuesta** al cliente (HTML, JSON, XML, etc.).

Flujo básico:

```
Cliente → Solicitud → Servidor → Procesa → Respuesta → Cliente
```

Clave:

- El cliente **no accede directamente a los datos**.
- El servidor controla la seguridad y la lógica.

2. Arquitectura de Dos Capas (2-Tier)

¿Cómo funciona?

1. El **cliente** contiene la interfaz y parte de la lógica.
2. El cliente se conecta **directamente a la base de datos**.
3. Realiza consultas (SELECT, INSERT, UPDATE).
4. La base de datos responde con los resultados.

Flujo:

```
Cliente ↔ Base de Datos
```

Clave:

- No hay capa intermedia.
- Más rápida en sistemas pequeños.
- Riesgosa en seguridad (credenciales en el cliente).

3. Arquitectura de Tres Capas (3-Tier)

¿Cómo funciona?

1. **Capa de presentación**:
    - Usuario interactúa con la interfaz web.
2. **Capa de negocio**:
    - Procesa reglas del sistema.
    - Valida datos.
3. **Capa de datos**:
    - Gestiona la base de datos.

Flujo:

```
Usuario → Presentación → Negocio → Datos
                      ←          ←
```

Clave:

- Cada capa es independiente.
- Cambiar una capa no afecta a las otras.
- Muy usada en sistemas profesionales.

4. Arquitectura MVC (Modelo-Vista-Controlador)

¿Cómo funciona?

1. El **usuario interactúa con la Vista**.
2. La Vista envía la acción al **Controlador**.
3. El Controlador:
    - Procesa la petición.
    - Llama al **Modelo**.
4. El Modelo:
    - Accede a la base de datos.
    - Devuelve los datos.
5. El Controlador envía los datos a la Vista.
6. La Vista se actualiza.

Flujo:

```
Usuario → Vista → Controlador → Modelo → BD
                      ↓
                    Vista
```

Clave:

- Separación clara de responsabilidades.
- Código ordenado y mantenible.
- Base de frameworks modernos.

5. Arquitectura Monolítica

¿Cómo funciona?

1. Todo el sistema está en **una sola aplicación**.
2. Una petición entra al sistema.
3. La misma aplicación:
    - Maneja la interfaz.
    - Procesa la lógica.
    - Accede a la base de datos.
4. Devuelve la respuesta.

Flujo:

```
Cliente → Aplicación Única → Base de Datos
```

Clave:

- Fácil de iniciar.
- Difícil de escalar.
- Un error puede afectar todo el sistema.

6. Arquitectura de Microservicios

¿Cómo funciona?

1. El cliente hace una solicitud.
2. Un **API Gateway** la recibe.

3. La solicitud se redirige al **microservicio específico**.
4. Cada microservicio:
    - Tiene su propia lógica.
    - Tiene su propia base de datos.
5. El microservicio responde al cliente.

Flujo:

```
Cliente → API → Microservicio → BD
```

Clave:

- Servicios independientes.
- Escalan por separado.
- Mayor complejidad técnica.

7. Arquitectura SOA (Orientada a Servicios)

¿Cómo funciona?

1. El sistema se compone de **servicios reutilizables**.
2. Los servicios se comunican a través de un **ESB (Enterprise Service Bus)**.
3. El ESB:
    - Gestiona mensajes.
    - Traduce formatos.
    - Coordina servicios.
4. Los servicios responden al sistema solicitante.

Flujo:

```
Aplicación → ESB → Servicio → ESB → Aplicación
```

Clave:

- Ideal para integrar sistemas antiguos.
- Más acoplada que microservicios.
- Muy usada en empresas grandes.

8. Arquitectura Serverless

¿Cómo funciona?

1. El usuario genera un evento (petición HTTP, archivo, clic).
2. El proveedor en la nube ejecuta una **función**.
3. La función:
    - Corre solo cuando se necesita.
    - Accede a servicios en la nube.
4. Termina su ejecución.
5. Se cobra solo el tiempo usado.

Flujo:

```
Evento → Función → Servicio Cloud → Respuesta
```

Clave:

- No gestionas servidores.
- Escala automáticamente.
- Dependencia del proveedor (AWS, Azure, Google).