# TEMA 1: INTRODUCCIÓN A LAS APLICACIONES PARA INTERNET

**¿Qué es una aplicación para Internet?**

* Usa el buscador como cliente
* En una aplicación de internet basada en el servidor, usa el protocolo de Internet HTTP para recibir solicitudes del **cliente (normalmente un navegador)** 🡪 procesa el código asociado 🡪 Devuelve datos al navegador.
* **Aplicación web:** Es una aplicación que se almacena en un servidor remoto y se distribuye mediante una interfaz del navegador (Gmail, Youtube, etc.)
* Ejecutan en Internet, datos se almacenan y procesan dentro de la web
* Codifica en un lenguaje soportado por los navegadores web
* Sistema Web: Infraestructura o sistema que permite funcionar a una aplicación web (GitHub API)

|  |  |
| --- | --- |
| **VENTAJAS** | **DESVENTAJAS** |
| Compatibilidad Multiplataforma (+S.O.) | Menor Funcionalidad que las aplicaciones de escritorio |
| Acceso desde Internet (Aplicaciones Multiusuario, todos accedemos) | Limitaciones del navegador |
| Aplicaciones Ligeras (No hace falta instalarse nada para ejecutarlo) | Alta dependencia del servidor |
| Fáciles de Actualizar (Usuario no tiene que descargar o instalar nuevas act) | Limitación del protocolo HTTP utilizado |
| Acceso Inmediato (Acceder Inmediatamente) | Cuello de botella en el ancho de banda de los datos |
| Compatibilidad de Versiones (No hay problema de compatibilidad de versiones, ya que todos los usuarios trabajan con la misma versión) | Necesidad de encriptar los datos |
| Consumo de recursos bajo, Alta Disponibilidad, Difícil de atacar mediante virus | Necesidad de software adicional o versiones específicas de navegadores |

**Sitio web vs aplicación web**

|  |  |
| --- | --- |
| **SITIO WEB** | **APLICACIÓN WEB** |
| +++ páginas web en 1 dominio | Suelen ser el propio negocio |
| Requieren poca o nula interacción por parte del usuario | Requieren interacción por parte del usuario y normalmente registrarse |
| Pueden crear con un CMS (Content Managment System) |  |
| **Ejemplo:** [**https://www.uspceu.com/**](https://www.uspceu.com/) | **Ejemplo:** intranet del CEU |

**Aplicación web vs Servicio web**

**NOTA(GEPETO):** En resumen, una **aplicación web (GMAIL, NETFLIX o Facebook)** es una herramienta interactiva para, mientras que un **servicio web (Google Maps API, Stripe API, OpenWeatherMap API)** es un recurso que facilita la comunicación y el intercambio de datos entre sistemas o aplicaciones.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Arquitectura**

**Diagrama, Forma, Flecha

Descripción generada automáticamente**

**MODELO BÁSICO:**

* Ordenador dedicado ejecutando un servidor
* Conexión a una red para pedir peticiones y devolver respuestas
* Si la red es interna a una organización: **intranet**

**MODELO CON SEPARACIÓN DE FUNCIONES:**

* Modelo básico, pero cambiando la 3ª por:
* **Conectividad** a otros servidores (de **bases de datos**, por ejemplo)

**MODELO CON SEPARACIÓN DE REDES**

* Modelo básico, pero cambiando la 3ª por:
* Conectividad a otros servidores en una red interna a través de un cortafuegos

**MODELO DE SEPARACIÓN COMPLETA DE FUNCIONES:**

* Ordenadores para servir contenidos estáticos (páginas HTML)
* Ordenadores diferentes para generar contenidos dinámicos
* Ordenadores diferentes para bases de datos
* Ordenadores diferentes para gestión, seguridad, etc.

**MODELO PARA UN ALTO RENDIMIENTO:**

* Varios ordenadores dedicados ejecutando el mismo servidor
* Conexión a una red a través de un balanceador de carga para recibir peticiones y devolver respuestas
* Conectividad a otros servidores (de bases de datos, por ejemplo)

**MODELO PARA UNA ALTA DISPONIBILIDAD:**

* El repartidor de carga envía la petición a dos o más servidores
* Todos los elementos están duplicados

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteFRONT-END Y BACK END**

**NOTAS Front-end:**

* Web Performance: Reducir tiempos de carga
* Cross-browser compatibility: Comprobar en diferentes navegadores Rendimiento, Funcionalidad, etc.
* Tecnologias: HTML, CSS, JS, Frameworks (Bootstrap, React, Angular, etc.)

**NOTAS Back-end:**

* Automated testing: Comprobación del servidor,APIs, etc.
* Data Access: Roles, No confiar en las peticiones del cliente
* High Availability: Despliegue en diferentes regiones, Bajo tiempo de recuperación, Amazon S3 (AWS)
* Tecnologías: PHP, Python, Node.js, C#, BBDD