DESPLIEGUE DE APLICACIONES WEB

INTRODUCCIÓN

1. TIPOS DE APLICACIONES

- Sin conexión: Aplicaciones de escritorio
- Con conexión: Via internet o red local
  - P2P: Conectados a un red en el mismo rango, conectados un nodo directamente con otro. (Torrent)
  - Aplicaciones Clientes-Servidor: Conjunto de ordenadores (Cliente) que se conectan a un ordenador centralizado (Servidor). (Aplicaciones Web)

1.1 Que es una aplicación Web

- La definición tradicional es una aplicación que se ejecuta en un navegador accediendo a un Servidor.
- Pero ahora se puede extender a cualquier aplicación realizada a partir de lenguajes de desarrollo web (HTML, JavaScript, CSS...).
  - Navegador convencional
  - Motor navegador embebido en otro sistema (webview Android o IOS).
  - Aplicaciones híbridas (Tecno. Web exportada)
  - Aplicaciones de escritorio con tecnologia WEB (Framework Electron de JavaScript).

2. Que es la WEB

- Plataforma mundial donde tenemos disponibles gran cantidad de recursos.
- Utilizar el protocolo TCP/IP.
- Su precursura ARPANET
- Se hizo popular en los 90 con el correo electrónico.
- Ha ido evolucionando hasta la actualidad que podemos ver películas por internet.

Elementos de una aplicación WEB

- CLIENTE: Donde está el usuario.
- SERVIDOR: Donde está la aplicación

Funcionamiento

1. El cliente inicia sesión en el servidor
2. El cliente solicita al servidor el recurso o servicio que quiere utilizar (una página web, un documento, subir información, etc.)
3. El servidor recibe la respuesta del cliente, la procesa y decide qué programa debe darle servicio, enviando la petición a dicho programa.
4. El programa responsable procesa la petición, prepara la respuesta y la entrega al servidor.
5. El servidor envía la respuesta al cliente
6. El cliente puede volver al paso 2 y realizar una nueva petición, o bien
7. El cliente termina la sesión en el servidor

Arquitectura de 2 niveles o 3 niveles

- El servidor puede ejecutarse todo en una sola máquina.
- Pero puede estar en varios equipos y tener diferentes niveles, lo que se conoce por arquitectura multicapa o multinivel.
- 2 niveles: Cliente y Servidor
- 3 niveles: Cliente, Servidor Web y Servidor de Base de Datos, es más flexible, más seguro y ofrece mejor renidimiento en sistemas congestionados, al dividir el trabajo.

Ubicación del Servidor. Hosting y nombres de dominio

- Para acceder a un Servidor escribimos www.iessanvicente.com y hay algo en Internet que sabe donde estar el servidor web del IES San Vicente.
- Ubicar el Servidor:
  - Disponer de un Servidor propio con una IP Fija (poco habitual)
  - Contratar espacio en una empresa Hosting. (OVH, Hostinger, Hostalia, 1&1).
- También habrá que comprar un nombre de dominio, es lo que escribimos detrás de la www. (10-15€)

Acceder al Servidor. URL

- La forma de solicitar recursos al servidor por parte del Cliente es a través de URLs (Uniform Resource Locator) identifica univocamente un servidor en internet.
- La URL tiene 3 partes, si tenemos http://www.iessanvicente.com/paginas/pagina.html:
  - El protocolo: Las reglas de comunicación entre Cliente y Servidor. (Http, Https, Ftp...)
  - Nombre del dominio: Identifica al Servidor.
  - Ruta hacia el recurso: Todas las carpeta y subcarpetas y el fichero que deseamos obtener.

Acceder al Servidor. DNS

- El nombre de dominio indicado por el Navegador se transforma en dirección IP a través de los Servidores DNS (Domain Name System). Y con la IP los router saben llegar hasta el Servidor.
- Hay una red de Servidores DNS para conseguir la IP.
- Podemos comprobar qué dirección IP tiene asignada un determinado dominio mediante comandos de terminal como nslookup o ping.
  - Si escribimos nslookup www.iessanvicente.com obtendremos la dirección IP asociada a la web del IES San Vicente
  - Si escribimos ping www.iessanvicente.com intentaremos comunicarnos con dicho servidor, y además, en el terminal nos mostrará la dirección IP con la que está intentando comunicar (que coincidirá con la obtenida con nslookup).

3. Ventajas Aplicaciones WEB vs Escritorio

- No tienen **nada que instalar**.
- El cliente apenas tiene **carga de trabajo** (puede saturar al Servidor y a veces hay tecnologias que liberan trabajo al Servidor)
- **Mantener y actualizar**. No hay que instalar nada en el Cliente y todo esta en el Servidor, y cambiando allí se ve reflejado inmedianto en todos los Clientes.
- **Compatibilidad**. No se depende de ningún SW ni HW
- **Control de Usuario y Seguridad**: Todo esta centralizado en el Servidor.
- **Movilidad**: Todo en el Servidor con tener conexión a internet puedes verlo en cualquier parte.
- **Escabilidad**: Como el trabajo recae en el Servidor con ampliar sus prestaciones, RAM, HW, número de máquinas es suficiente.

Inconvenientes de las Aplicaciones WEB

- Solventadas:
  - **Riqueza gráfica:** Tradicionalmente una aplicación de escritorio siempre ha tenido mejores prestaciones que una WEB, pero tecnologias como Flash, librerias javaScript, CSS3... hacen que la diferencia cada vez sea menor.
  - **Necesidad de conexión**: En una aplicación Web se necesita conexión pero algunas aplicaciones como Google Drive o Drop Box pueden trabajar sin conexión y actualizar los cambios cuando haya, también LocalStorage o SQLite almacenan en local hasta conseguir conexión
- Sin solventar:
  - **Rendimiento y tiempo de respuesta**: Comparado con aplicaiones de escritorio, tanto el rendimientos como tiempo de resupesta suelen ser inferiores al no necesitar conexión.
  - **Tráfico generado**: Al estar en una red, depende de la congestión de la misma y la cantidad de usuarios que trabajen al mismo tiempo.
  - **Apariencia**: La apariencia puede variar según el Navegador que se utilice y a veces canvia el resultado notablemente.

4. Protocolos más utilizados

- Para comunicarse entre Cliente y Servidor se requiere un lenguaje unas reglas de comunicación, esto es un protocolo.
- En red toda la comunicación se basa en el:
  - TCP: Establece como debe estructurarse y fraccionarse la información enviada.
  - IP: Como se identifican los equipos en la red.
- Sobre la base de TCP/IP se estable una serie de protocolos:
  - HTTP: Permite la transferencia de archivos.
  - HTTPS: La misma que la anterior pero con mayor seguridad, ya que los datos de peticiones y respuestas van encriptados. (Se debe configurar el Servidor para ello).
  - SMTP o POP3/IMAP: Envio y recepción de correo electrónico.
  - FTP: Para transferencia de ficheros.

HTTP (utilizado por Aplicaciones WEB)

- PETICIONES (REQUEST), se encapsulan en paquetes y fragmentan según el protocolo HTTP:
  - URL: Del recurso solicitado.
  - Cabecera de la petición: Informa del recurso solicitado y el cliente que lo solicita.
  - Datos adicionales.
- RESPUESTAS (RESPONSE), se encapsula la respuesta del Servidor, con datos o con mensaje de error:
  - Código de estado: Indica si todo ha ido bien o no:
    - 2XX: Todo OK.
    - 3XX: Ha habido algún tipo de redirección
    - 4XX: Error por parte del Cliente. 404 la URL no existe. 403 El cliente no tiene permiso.
    - 5XX: Error por parte del Servidor. Colapsado, Exceso de tiempo...
  - Cabecera de respuesta: Tamaño de respuesta, tipo de contenido, fecha de modificación...
  - Contenido solicitado

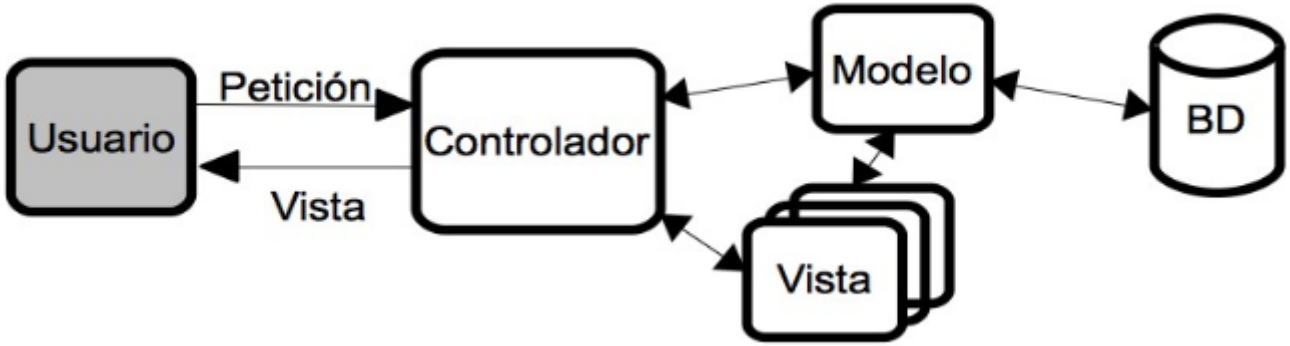
5. PATRONES DE DISEÑO SW

- Consiste en un conjunto de pautas y elementos a desarrollar que dotan a una aplicación de una estructura preestablecida.
- Se pretende:
  - Estandarizar la forma que se desarrolla.
  - Elaborar elementos reutilizables entre diversas aplicaciones.
- El más famoso es MVC (Modelo-Vista-Controlador).

MVC

- Se basa en dividir el desarrollo web en 3 componentes:
  - El modelo: Los datos que maneja la aplicación. Variables y objetos sacados de la BD. Conectar a la BD. Recuperar o guardar información en la BD.
  - La vista: Lo que ve el usuario en pantalla. Páginas, formularios…
  - El controlador: Coordinan el funcionamiento general. Ante peticiones del usuario acceder al Modelo para actualizar o recuperar dato y deciden que Vistas mostrar.

MVC II



MVC III

- Un diseño muy conciso y bien estructurado.
- Divide muy bien las tareas, todo es muy modular:
  - VISTAS: Un diseñador web
  - CONTROLADOR: Un programador de PHP

Alternativas al MVC

- Surgen a partir del patrón MVC, para cuestiones más específicas.
- El MODELO (El acceso y gestión de los datos) y LAS VISTAS (La presentación) son base. El único prescindible es el CONTRALADOR que lo hacen submódulos.
- Al conjunto de patrones que siguen esta idea se llaman MVW (Modelo Vista y lo que sea)

MVVM (Modelo Vista Vista Modelo)

- El usuario interactúa directamente con la VISTA y realiza acciones sobre el MODELO.
- El patrón más relevante es SPA (Single Page Applications), aplicaciones web de 1 sola página que va cambiando dependiendo de las acciones del usuario.

MOVE (Modelo Operaciones Vista Eventos)

- Divide el Controlador en:
  - Operaciones: Conjunto de acciones que la aplicación puede realizar.
  - Eventos: Sucesos que desencadenan que se realice una acción determinada.
- Las acciones de los usuarios son eventos sobre la aplicación que provoca que se ejecuten operaciones.

MVP (Modelo Vista Presentadores)

- Presentadores: Intermediarios entre el modelo y la vista. Cada vista tiene el suyo propio y se utiliza para interactuar con otras vistas, comunicándose con el MODELO.

6. Recursos Necesarios

- En el lado del CLIENTE: Un equipo (puede ser un móvil o tablet) con un navegador instalado.
- En el lado del SERVIDOR: Al menos 1 PC con prestaciones HW potentes (Procesador, RAM y disco) y debe tener instalado:
  - Un servidor WEB: Donde estará alojada la aplicación WEB y atenderá las peticiones de los CLIENTES. (Un servidor web en producción en una IP remota y un Servidor web local para pruebas).
  - Servidor de Base de Datos: Con un SGBD.

LENGUAJES

- En el lado del CLIENTE: Los que permiten que el CLIENTE interactue con la aplicación WEB.
  - Tenemos HTML y CSS: Para diseño de páginas web.
  - JavaScript: Facilitar la interacción Cliente-Servidor.
  - Diversos Frameworks: SAS, Jquery o Angular.
- En el lado del SERVIDOR: Los que permiten que el Servidor realice las tareas solicitadas.
  - ASP.NET (Servidores Windows)
  - JSP (JAVA para aplicaciones WEB)
  - PHP
  - NODE.JS o Backbone.js

EJEMPLOS DE SOFTWARE

- CLIENTE: Navegadores. (Firefox, Chorme, Internet Explorer, Safari, Opera...)
- SERVIDOR WEB:
  - Apache si vamos a usar PHP
  - Tomcat si vamos a usar JAVA
  - IIS si usamos .NET
  - NODE.JS si usamos el grupo de JavaScript.
- SERVIDOR DE BD:
  - MySQl, Postgresql, ORACLE o SQL SERVER.