



ESTUDIO 1º UNIDAD

ÁLVARO GARCÍA GONZÁLEZ

1. Protocolos de comunicaciones: IP, TCP, HTTP, HTTPS.	4
2. Modelo de comunicaciones cliente – servidor y su relación con las aplicaciones web.	5
3. Estudio sobre los métodos de petición HTTP /HTTPS más utilizados.	6
4. Estudio sobre el concepto de URI (Identificador de Recursos Uniforme) /URL/URN, estructura, utilidad y relación con el protocolo HTTP/HTTPS.	7
5. Modelo de desarrollo de aplicaciones multicapa – comunicación entre capas – componentes – funcionalidad de cada capa.	8
6. Modelo de división funcional front-end / back-end para aplicaciones web.	9
7. Página web estática – página web dinámica – aplicación web – mashup .	10
Estudios propuestos para el módulo Desarrollo Web en Entorno Servidor	12

Realiza un estudio sobre los siguientes conceptos:

1. Protocolos de comunicaciones: IP, TCP, HTTP, HTTPS.

El https es el **protocolo básico** de comunicación. El **puerto** por defecto es el 443.

Para que funcione necesita las siguientes partes que forman parte del protocolo TCP/IP:

HTTP: es el protocolo que realiza la petición de datos y recursos. La petición de datos es iniciada por el cliente. La **diferencia** con https es que este último añade una **capa de seguridad** a través del cifrado. Puerto por defecto:80

TCP: se encarga de **establecer y mantener** la conexión entre el emisor y receptor. TCP se considera más fiable en el apartado de entregas correctas comparado con el resto de protocolos.

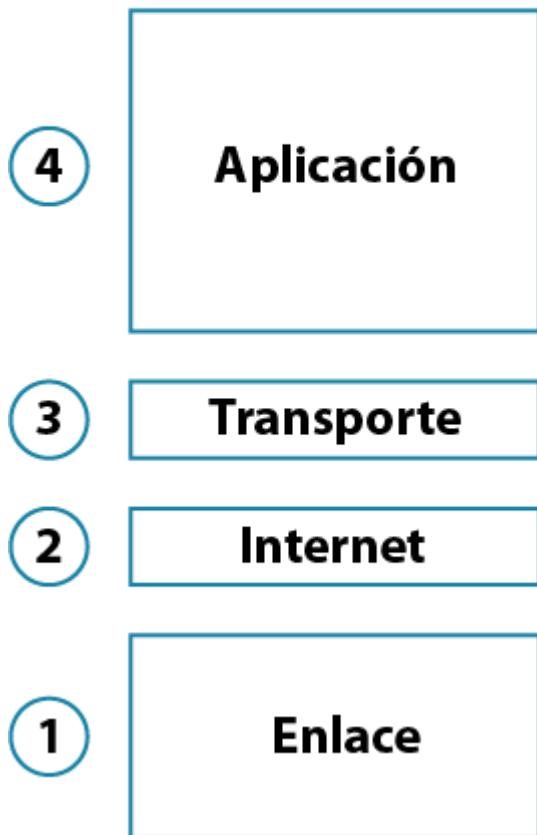
IP: es el **conjunto de reglas** que define como se comunican los dispositivos.

Más información

-> <https://www.cloudflare.com/es-es/learning/network-layer/internet-protocol/>

En la siguiente foto, la capa de aplicación correspondería con el HTTPS/HTTP, la capa de transporte con el TCP y la de internet con la IP

Modelo TCP/IP



2. Modelo de comunicaciones cliente – servidor y su relación con las aplicaciones web.

Es un modelo donde el **cliente** es siempre el que **comienza el dialogo** y el **servidor responde** a esas peticiones. **Todas** las aplicaciones web son **cliente-servidor**.



3. Estudio sobre los métodos de petición HTTP /HTTPS más utilizados.

Estos métodos son las peticiones que hacen los clientes para que el servidor entienda las solicitudes.

Algunos de los métodos más utilizados son “get”, “head”, “post”, “put” y “delete”

Método HTTP	Significado en Restful Web Services
GET	Se utiliza para operaciones de sólo lectura. No generan ningún cambio en el servidor.
DELETE	Elimina un recurso en específico. Ejecutar esta operación múltiples veces no tiene ningún efecto.
POST	Cambia la información de un recurso en el servidor. Puede o no regresar información.
PUT	Almacena información de un recurso en particular. Ejecutar esta operación múltiples veces no tiene efecto, ya que se está almacenando la misma información sobre el recurso.
HEAD	Regresa solo el código de respuesta y cualquier cabecera HTTP asociado con la respuesta.

Para información más detallada sobre los métodos: https://estilow3b.com/metodos-http-post-get-put-delete/#Metodo_HTTP_DELETE

4. Estudio sobre el concepto de URI (Identificador de Recursos Uniforme) /URL/URN, estructura, utilidad y relación con el protocolo HTTP/HTTPS.

Para que el servidor entienda las peticiones del cliente, tiene que realizar una petición formal que deben seguir un formato concreto.

URL: Las direcciones suelen ser mediante URL que contiene una **dirección**, la **referencia al protocolo utilizado** y la **ruta al objeto** que queremos

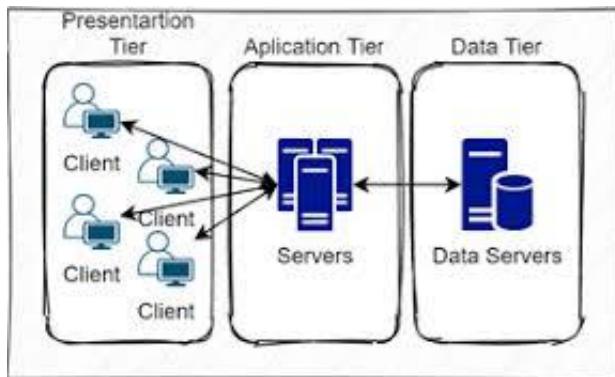
URN: Asigna un código de identificación a los objetos y este no varía, aunque el recurso se traslada. Las partes son: 1. información sobre el esquema URN 2. El NID (identificador de espacio de nombres) 3. El NSS, que es el identificador como tal.

URI: Es el identificador uniforme de recursos y sirve para acceder a todos los tipos de recursos. Abajo hay una foto que explica la estructura.



5. Modelo de desarrollo de aplicaciones multicapa – comunicación entre capas – componentes – funcionalidad de cada capa.

El modelo multicapa que se usa para poder **separar un proyecto en varias partes**. Es útil para poder **desarrollar una capa sin afectar a las otras**. Por ejemplo, cuando actualizamos la capa de negocio, nos aseguramos de no afectar a los datos que se encuentran en la capa de persistencia. El modelo funciona de esta manera:



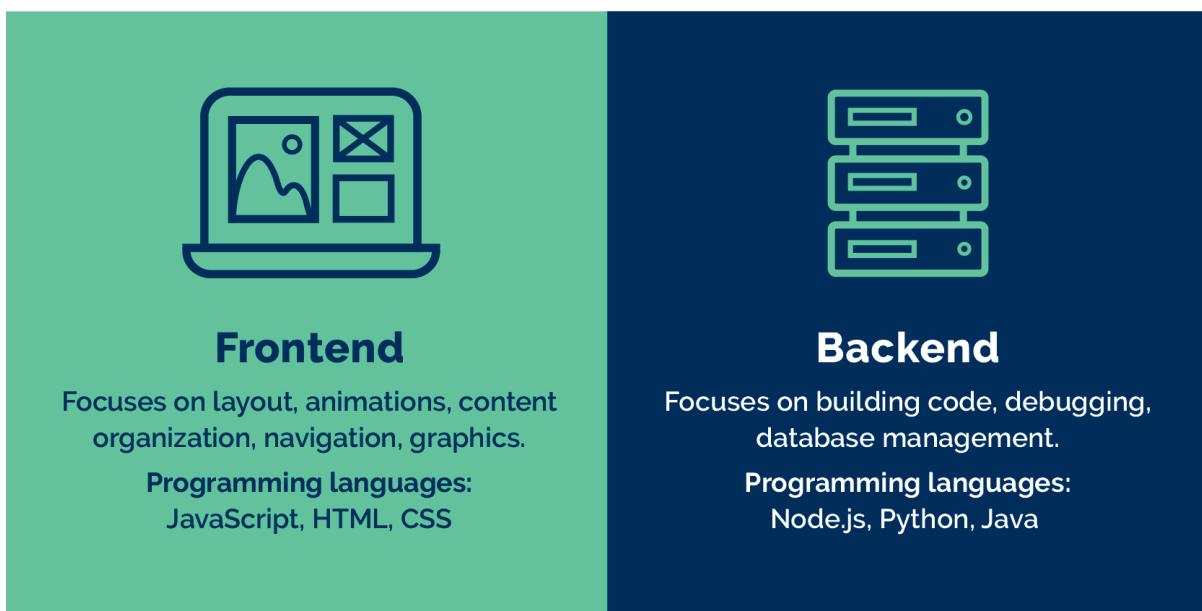
Capa de presentación: es lo que ve el usuario y recoge la interacción de este.

Capa de negocio: gestiona las funcionalidades de la aplicación web. Es donde, normalmente, se reciben las peticiones de los usuarios y se devuelven las respuestas.

Capa de persistencia: es donde se almacenan los datos y desde donde se accede a ellos. Reciben solicitudes de almacenamiento o de recuperación de información.

6. Modelo de división funcional front-end / back-end para aplicaciones web.

El **front-end** trata todo aquello que los **usuarios normales pueden ver e interactuar**, mientras que el **back-end** es lo que hace que la **aplicación funcione y la parte de** la aplicación dedicada a **los usuarios especiales** (administrador, publicador, etc....). Ambas partes se deben **desarrollar de forma consciente la una de la otra**, ya que ciertos aspectos del front-end pueden mejorar la velocidad y el rendimiento de la aplicación.



Para indagar más en las diferencias: <https://aws.amazon.com/es/compare/the-difference-between-frontend-and-backend/>

7. Página web estática – página web dinámica – aplicación web – mashup .

Página web estática: Su contenido **nunca varía** (salvo que el programador lo haga, claro). Y solo necesitan un servidor comunicándose con la web para que funcionen.

Página web dinámica: su contenido **cambia dependiendo de ciertas variables** (por ejemplo, las acciones que has realizado anteriormente en esa web). También puede ser que la página sea **generada completamente** de forma dinámica (Por ejemplo, tu correo, cada uno tiene sus mensajes).

Aplicación web: es la evolución de una página web dinámica, ya que usa estas últimas para **crear aplicaciones** (sin necesidad de instalación) que se **ejecutan en un servidor y se muestran en la web**

Mashup: Es una forma de integración en la que una aplicación web es usada por otra para utilizar su funcionalidad, normalmente mediante APIs públicas

8. Componentes de una aplicación web.
9. Programas ejecutados en el lado del cliente y programas ejecutados en el lado del servidor - lenguajes de programación utilizados en cada caso.
10. Lenguajes de programación utilizados en el lado servidor de una aplicación web (características y grado de implantación actual).
11. Características y posibilidades de desarrollo de una plataforma XAMPP.
12. En qué casos es necesaria la instalación de la máquina virtual Java (JVM) y el software JDK en el entorno de desarrollo y en el entorno de explotación.
13. IDE más utilizados (características y grado de implantación actual).
14. Servidores HTTP /HTTPS más utilizados (características y grado de implantación actual).
15. Apache HTTP vs Apache Tomcat
16. Navegadores HTTP /HTTPS más utilizados (características y grado de implantación actual).
17. Generadores de documentación HTML (PHPDoc): PHPDocumentor, ApiGen, ...
(Esperar hasta que lo explique Amor)
18. Repositorios de software – sistemas de control de versiones: GIT, CVS, Subversion, ...
19. Propuesta de configuración del entorno de desarrollo para la asignatura de Desarrollo web del lado servidor en este curso (incluyendo las versiones): xxx-US ED y xxx-WX ED. (Explicado en tema 2)
20. Propuesta de configuración del entorno de explotación para la asignatura de Desarrollo web del lado servidor en este curso (incluyendo las versiones): xxx-US EE.(Explicado en tema 2)
21. Realizar un estudio sobre los siguientes conceptos y su relación con el desarrollo de aplicaciones web: CMS – Sistema de gestión de contenidos ERP – Sistema de planificación de los recursos empresariales
22. Elegir y realizar un estudio y una presentación para la exposición del trabajo sobre una de las siguientes arquitecturas de desarrollo de Aplicaciones Web:
 - MEAN (con MongoDB y con MySQL)
 - Java EE vs Spring
 - Microsoft .NET
 - Angular 7
 - Symfony
 - Laravel

- CakePHP
- CodeIgniter

Estudios propuestos para el módulo Desarrollo Web en Entorno Servidor

➤ Contenidos y la diferencia entre los módulos que tienes en este curso.

DWES, DWEC, DIW, DAW.

En DWEC estudiamos todo lo que tiene que ver en el desarrollo por parte del cliente, es decir, todo aquello que pueden ver los usuarios comunes. En DWES también desarrollamos, pero en la parte del servidor, es decir, la parte que realiza los cálculos y la que más cerca está de la información (aparte de tener que crear otra web para los usuarios especiales que administran la aplicación). DIW trata sobre el diseño de nuestra aplicación, que la hará bonita y apetecible para que nos quedemos en ella. Por último esta DAW, que trata sobre el despliegue de la aplicación.

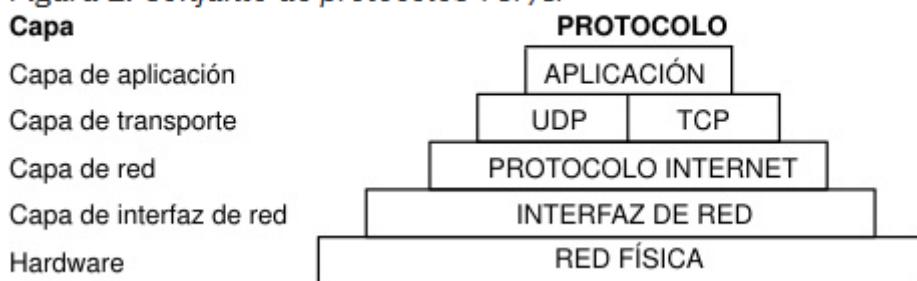
➤ Protocolos TCP/IP. Socket.

El protocolo TCP/IP es un conjunto de reglas que permiten la comunicación y la transferencia de datos entre dispositivos.

Las diferentes capas son: Aplicación, transporte, Internet, Acceso a red. La capa de acceso a red a veces se divide en dos: Interfaz de red y red física.

<https://www.ibm.com/docs/es/aix/7.1.0?topic=protocol-tcpip-protocols>

Figura 1. Conjunto de protocolos TCP/IP



Socket

<https://psp.codeandcoke.com/apuntes:sockets>

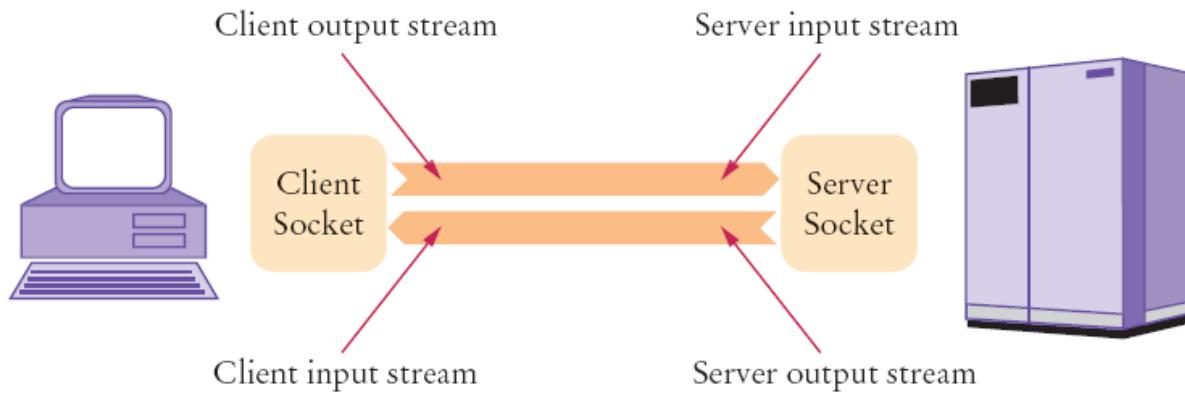


Figure 5 Client and Server Sockets

➤ Protocolo HTTP / HTTPS

Siglas de **protocolo de transferencia de hipertexto**. Forma parte de la capa de aplicación del protocolo TCP/IP y se utiliza para cargar páginas web mediante los enlaces de hipertexto. La diferencia con el protocolo HTTPS es que añade una capa de cifrado, el resto es igual.

➤ HTML

Es un lenguaje de marcas que sirve para darle la estructura a una página web.

➤ XML

Es un lenguaje de marcas donde debes definir tus propias etiquetas y que sirve principalmente para compartir datos.

➤ JSON

<https://www.json.org/json-es.html>

➤ Lenguajes de programación embebidos en HTML

Instrucciones que se encuentran dentro del código de HTML y enmarcados en etiquetas específicas. Así puedes ejecutar instrucciones para crear contenido interactivo o generar HTML. Ejemplos: PHP, JS, JSP

➤ Arquitecturas de desarrollo web

➤ Framework de desarrollo Web

➤ ERP

➤ CMS

➤ PHP

- IDE
- Navegador
- Repositorio
- Entorno de Desarrollo
- Entorno de Explotación o Producción
- Gestión de la configuración. Control de cambios. Mantenimiento de la aplicación.
- Web Services
- AJAX
- Desarrollo de aplicaciones multicapa. Estrategias de diseño de aplicaciones Web. ➤ Aplicaciones basadas en microservicios
- SaaS: Software as a Service
- Control de acceso a la aplicación web o los Web Services
- Validación de entrada de datos a una aplicación Web
- Posicionamiento de una aplicación Web
- Historia, situación actual y evolución del diseño de aplicaciones Web
- Filosofías de desarrollo del software