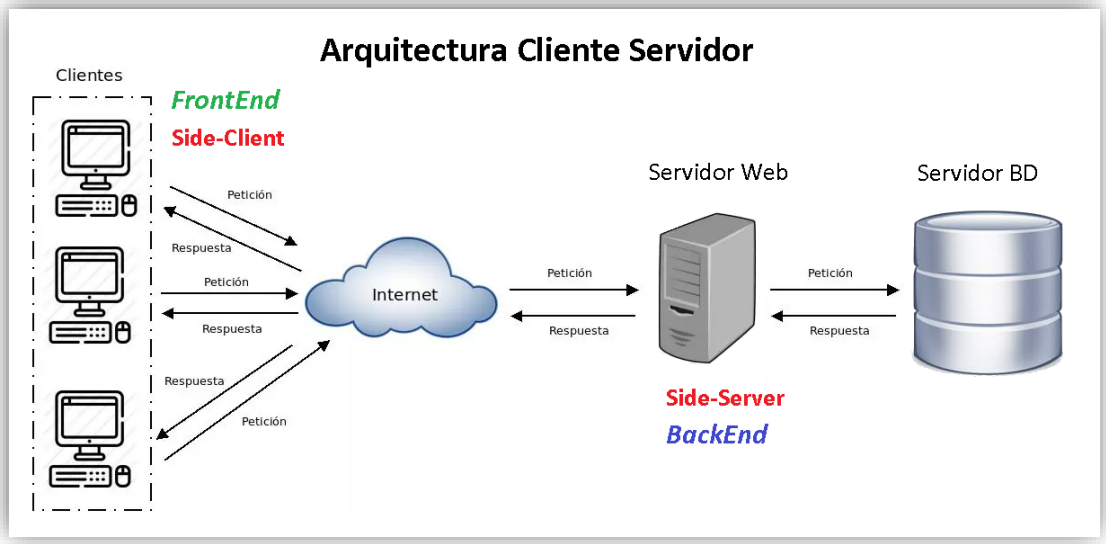
Sesión de Aprendizaje N° 09-A

Desarrollo Backed: Arquitectura y Principios

1. Arquitecturas de las Aplicaciones Web

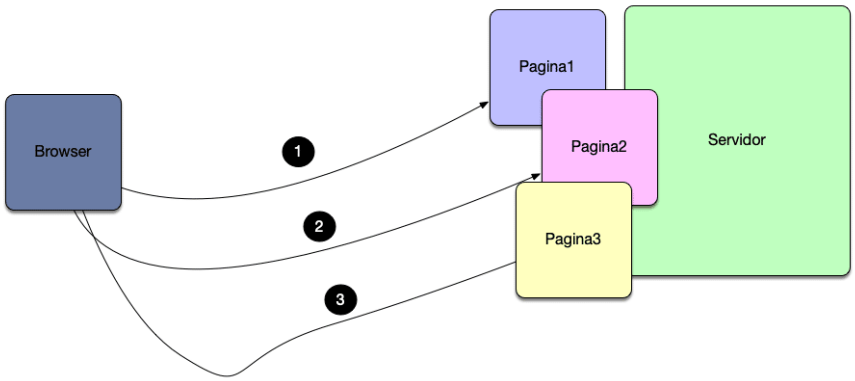
El desarrollo de aplicaciones para Internet está basado en una arquitectura cliente servidor.



Tipos de Arquitectura:

Arquitectura C/S Clásica – Múltiples Páginas

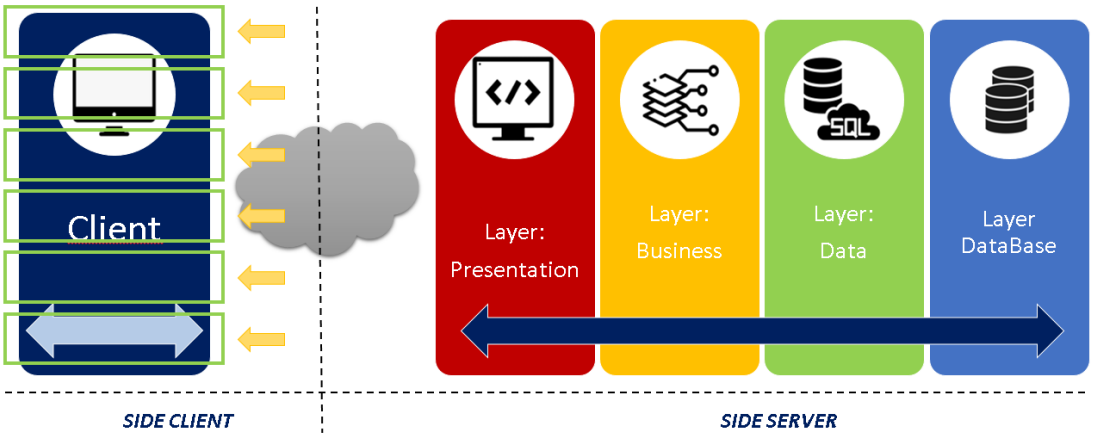
Las *Multi Page Application* son las *Web* Clásicas que contiene *múltiples páginas* y cada una carga diferentes contenidos, físicamente cada *página tiene su archivo html, css, js y recursos*. Se conecta mediante links con las demás y sus rutas son accesibles a un grupo de directorios.



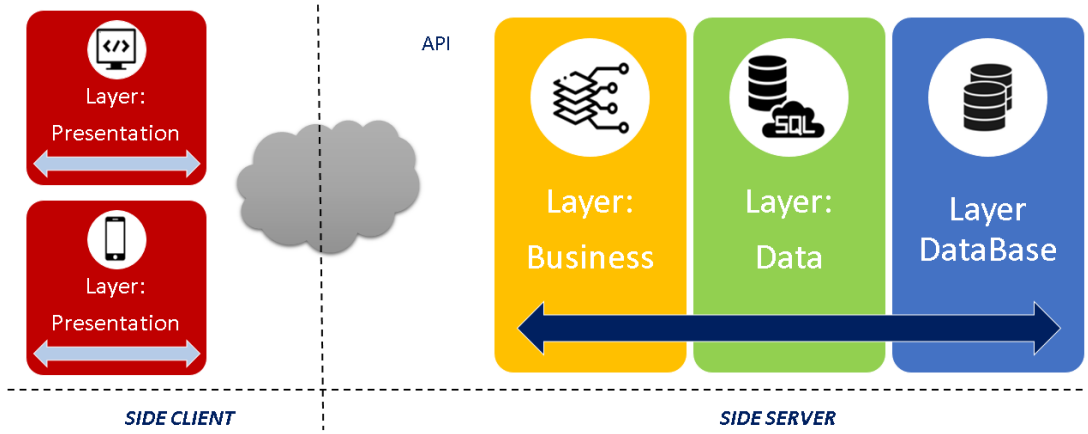
.

Arquitectura C/S Multicapa – Múltiples Páginas

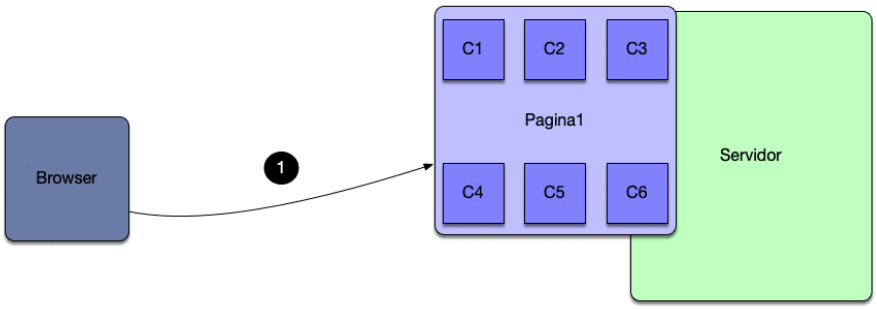
La arquitectura en capas es una técnica Server Side que los Diseñadores/ Desarrolladores de software utilizan para separar un sistema de software muy complejo. Esta separación se conoce como LAYERING, que simplemente significa descomponer un sistema de software en sus partes, o capas según preocupaciones y responsabilidades. Modelo MVC



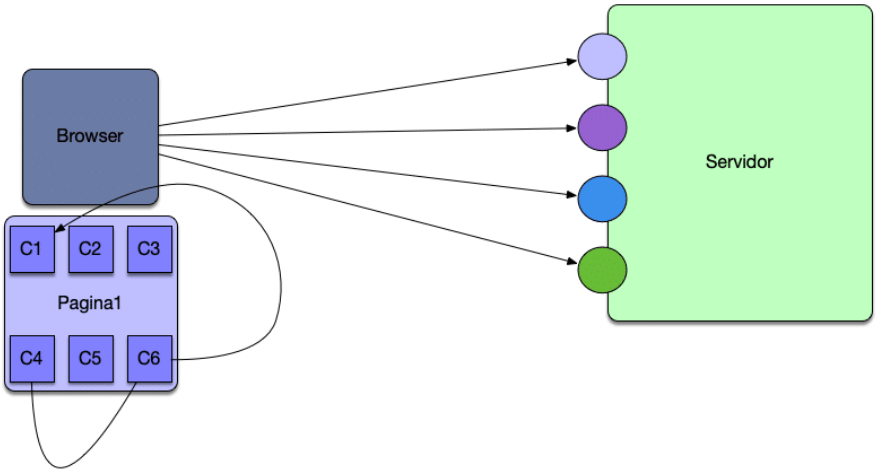
Arquitectura C/S Multicapa – Página Única SPA



Las ***Single Page Applications*** son arquitecturas en el que el servidor únicamente dispone de una página html, carga todos los estilos de un css y todos los formatos en el cliente durante el primer acceso.

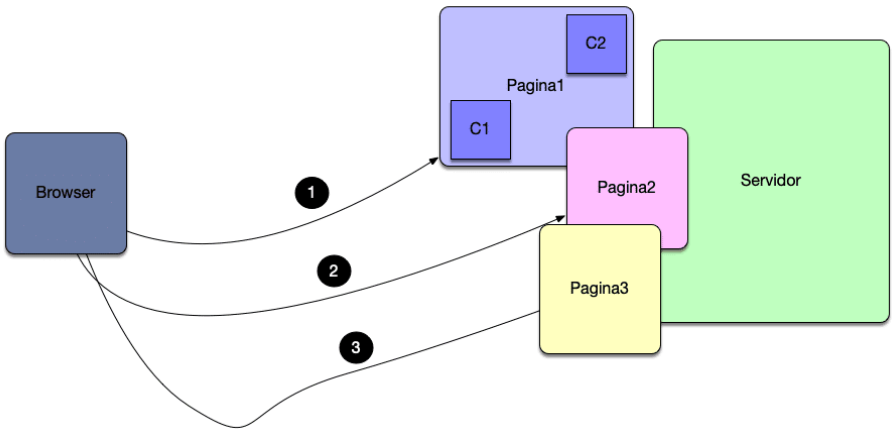


Una vez que el navegador se ha descargado la mega página usa JavaScript para navegar entre los diferentes componentes y dar la sensación al usuario de tener una navegación normal. La aplicación SPA solo pedirá al servidor datos puros.



Este tipo de Arquitectura aporta flexibilidad ya que permite que ***el servidor se centre en enviar información al cliente*** que solo incluya datos. Aportando la posibilidad de que otros clientes se conecten al servidor sin ser páginas web como podría ser por ejemplo una aplicación Android.

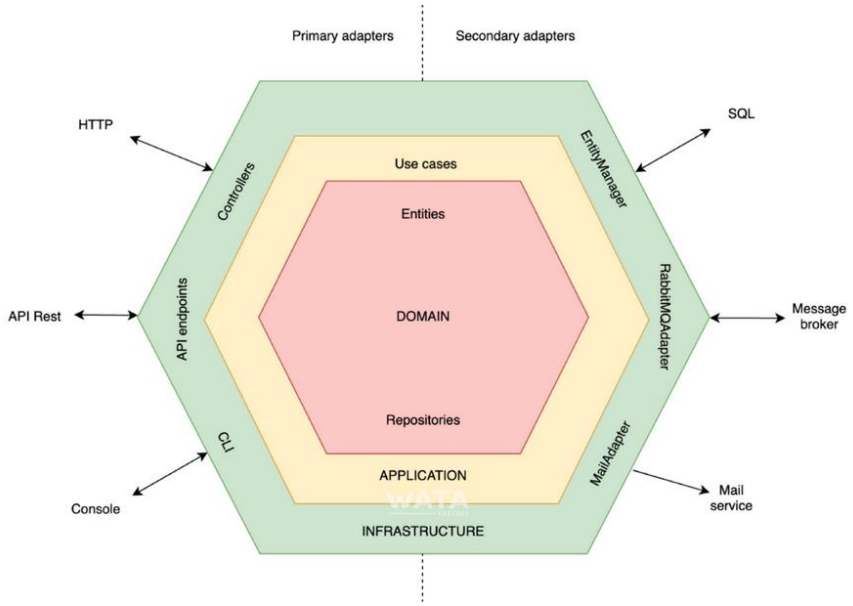
Arquitectura C/S Multicapa –SPA / MPA- Hibridas

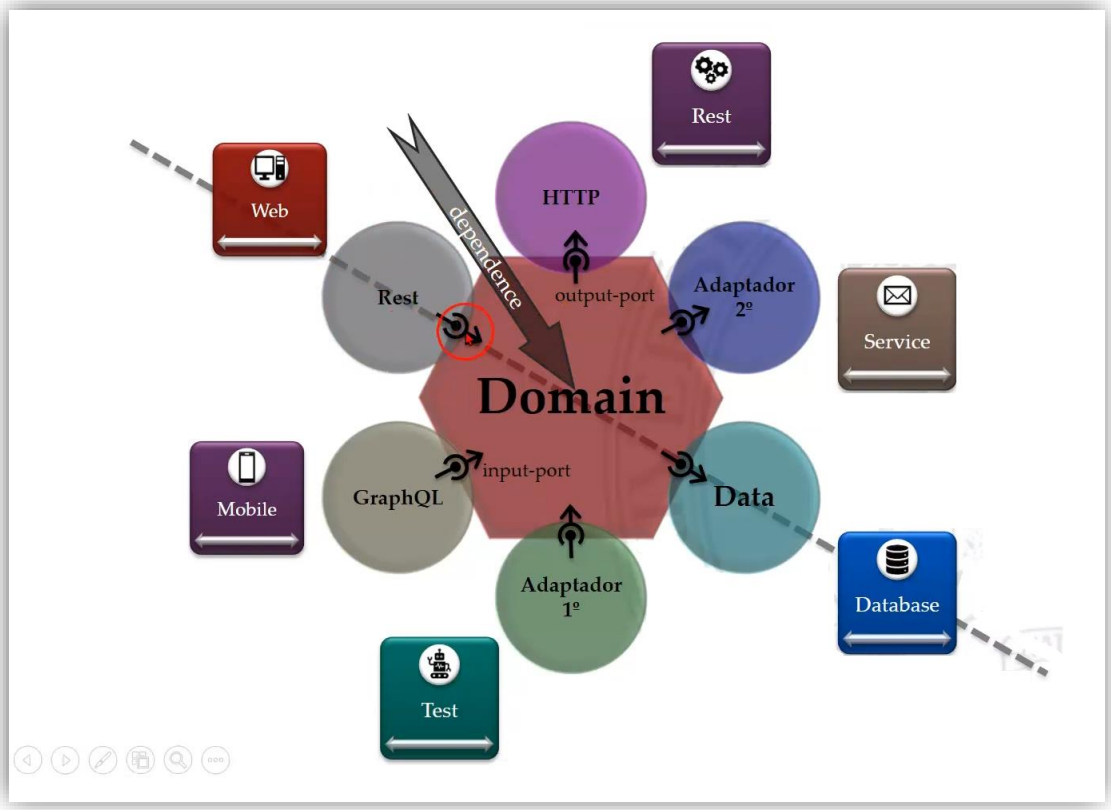


Una aplicación MPA / SPA es que contiene características de SPA como puede ser la construcción de componentes con Next JS aunque mantiene el enfoque multipágina.

Arquitectura Hexagonal

Esta arquitectura, también llamada ***arquitectura de puertos y adaptadores*** *propone separar nuestra aplicación en distintas capas o regiones*, cada una de ellas con su propia responsabilidad permitiendo así que evolucionen de manera aislada, y que cada una de ellas sea testeable e independiente de las demás.





Para conseguir esta independencia de capas se utiliza el *concepto de puertos y adaptadores*. Un **puerto** no *es más que un concepto lógico mediante el cual se define un punto de entrada y salida de la aplicación*. La **función del adaptador** es *la de implementar la conexión con ese puerto y otros servicios externos*. De esta forma podremos tener múltiples adaptadores para un mismo puerto.

Existen dos clases de puertos y adaptadores: primarios y secundarios. La diferencia entre ellos radica en quién desencadena o está a cargo de la conversación.

2. Servidores Web / Servicio Hosting / Cloud Hosting

SERVIDORES WEB

Equipos de gran potencia con **programas** (llamados **daemon**) que tienen como función principal **almacenar** todos los archivos propios de un ***sitio web*** (*html, css, javascript, imágenes, textos, videos, etc.*) y **transmitirlos** a los usuarios a través de los navegadores mediante el servicio HTTP. Bajo una arquitectura cliente / servidor.

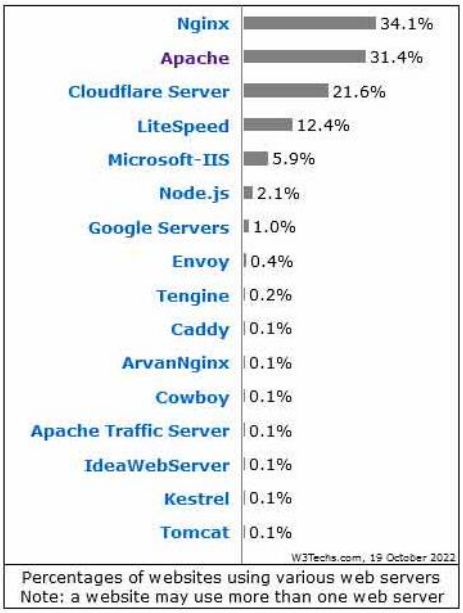
Servidor HP ProLiant DL160 Gen10 Xeon Bronze 3206R

¿Qué Software utiliza un Servidor Web?

Software Servidores Web

Open Source	Licencia
➢ Apache HTTP Server ➢ NGINX	➢ Cloudflare Server ➢ LiteSpeed ➢ Microsoft IIS
➢ Apache Tomcat ➢ Google Servers ➢ IBM Servers ➢ Lighttpd ➢ Node.js ➢ Oracle Servers ➢ Tengine	**Herramientas** ➢ LAMP/LEMP ➢ MEAN/MERN ➢ XAMMP ➢ WAMP

Estadísticas de uso



Percentages of websites using various web servers
Note: a website may use more than one web server

SERVICIO HOSTING

Un ***hosting*** es un **servicio de alojamiento web**, alojan los contenidos de tu web y tu correo electrónico para que puedan accesados desde cualquier dispositivo conectado a Internet. Cada ***servicio hosting*** tiene un plan de costo, espacio y transferencia.

Los ***GB de espacio*** de tu plan de hosting serán el espacio que pueden ocupar tus contenidos en el servidor, y los ***GB de transferencia*** mensual hacen referencia a la cantidad de transferencia de datos que podrás consumir.

Tipos de hosting

- Hosting compartido
- Hosting de servidor privado virtual (VPS)
- Hosting dedicado
- Hosting administrado
- Hosting en la nube

CLOUD HOSTING Y CLOUD COMPUTING

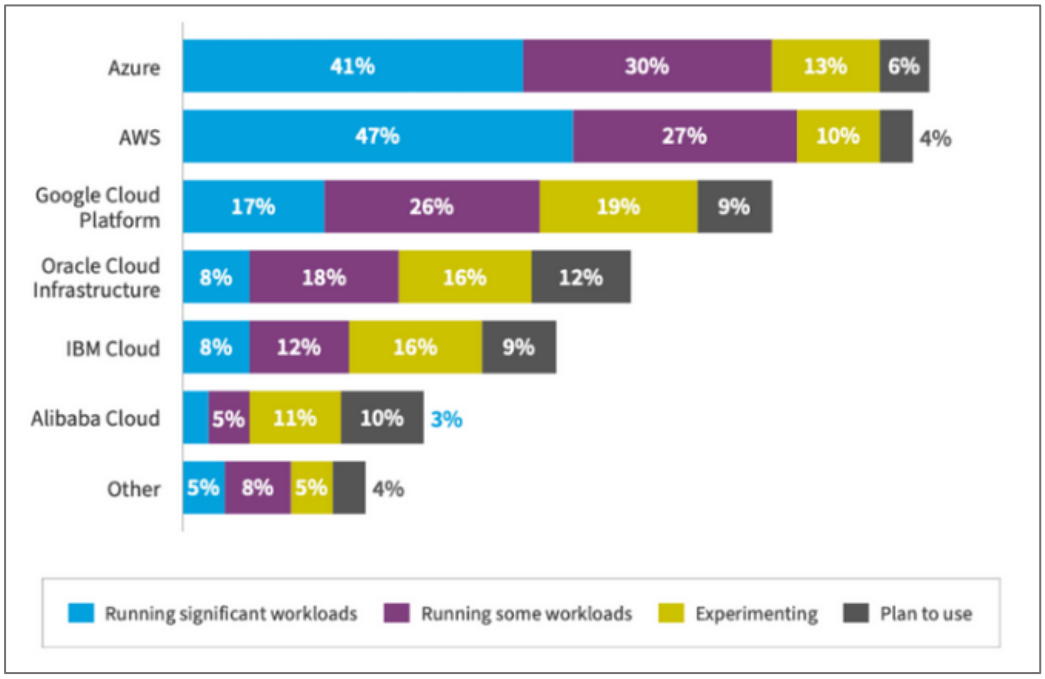
Cloud Hosting reúne los *recursos de computación* de una red de servidores físicos y virtuales, por lo que ofrece una mayor escalabilidad y flexibilidad para hacer cambios rápidamente.

En la mayoría de los casos, el alojamiento en la nube también es de pago por uso, lo que significa que los equipos pagan por lo que usan y no tienen que preocuparse por el sobre aprovisionamiento o el aprovisionamiento de recursos.

Cloud computing

Cloud computing es la disponibilidad bajo demanda de recursos de computación como servicios a través de Internet. Esta tecnología evita que las empresas tengan que encargarse de aprovisionar, configurar o gestionar los recursos y permite que paguen únicamente por los que usen.

Proveedores de alojamiento en la nube



3. Administración de Servidores Web

3.1. GESTIÓN DE NOMBRE DE DOMINIO

Tu nombre de dominio es, sin duda, uno de los elementos más importantes de toda tu presencia online. Es el enlace entre tu público y tu página web, establece el posicionamiento de marca y el marketing en general.

Nombre de Dominio Web / DNS

Es el nombre único que recibe un sitio web en internet.

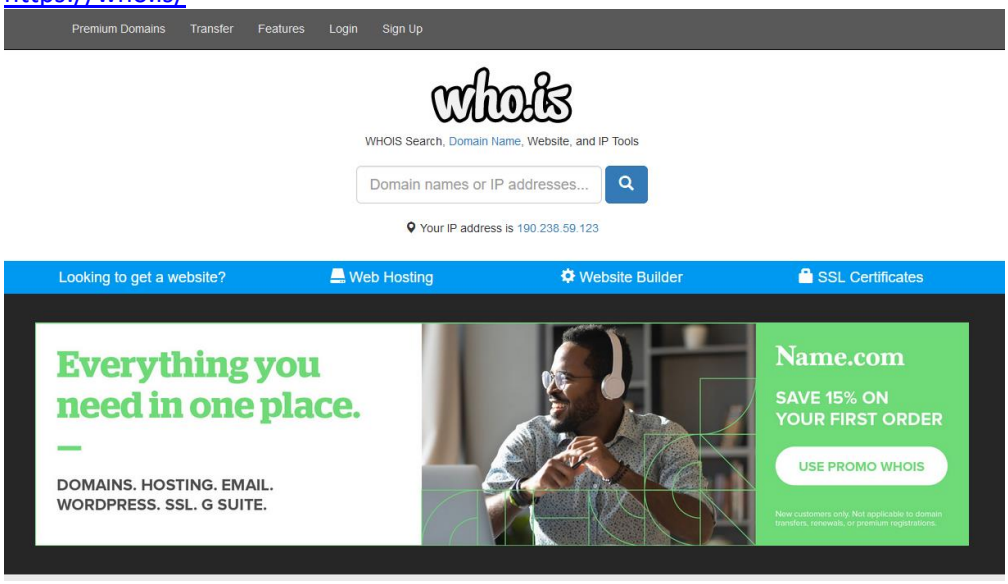
Partes de un dominio web.

La primera parte denominada nombre de dominio es el nombre que queremos dar a nuestro sitio web.

La segunda parte denominada extensión de dominio hace referencia a una ubicación geográfica como .es (España), .mx (México), etc., o a un tipo web en concreto como .org (organizaciones), .net (tecnológicas).

Comprobar Disponibilidad de Dominio

https://who.is/



Registradores de Dominios

Cada registrador de dominios ofrece extensiones, herramientas de gestión o funciones de privacidad distintas.

- Wix
- Domain.com
- Namecheap
- Bluehost
- GoDaddy
- Google Domains
- HostGator

3.2. CONFIGURACIÓN DE SERVIDORES WEB

Es el ***proceso de ajustar y optimizar un servidor*** para que pueda gestionar solicitudes HTTP y servir contenido web de manera *eficiente y segura*. Esto incluye la instalación del software del servidor, la definición de parámetros clave y la implementación de medidas de seguridad.

Archivos de Configuración:

En **Apache HTTP Server**, el archivo principal de configuración es **httpd.conf**, ubicado generalmente en **/etc/httpd/conf/** o **/usr/local/apache2/conf/**.
En Apache Tomcat, la configuración se gestiona principalmente en server.xml, ubicado en el directorio conf/ dentro de la instalación de Tomcat..

Parámetros esenciales en la configuración de un servidor web

- **DocumentRoot:** Define el directorio donde se almacenan los archivos accesibles desde la web.

Apache HTTP: DocumentRoot "/var/www/html"
Apache TomCat: DocumentRoot "/tomcat/htdocs"

- **Listen:** Especifica el puerto en el que el servidor escucha las solicitudes (por defecto, **80** para HTTP y **443** para HTTPS).
- **ServerName:** Indica el dominio o dirección IP del servidor.
- **DirectoryIndex:** Determina el archivo predeterminado que se carga en un directorio (por ejemplo, index.html, index.php, index.jsp).
- **ErrorDocument:** Permite personalizar páginas de error (404, 500, etc.).

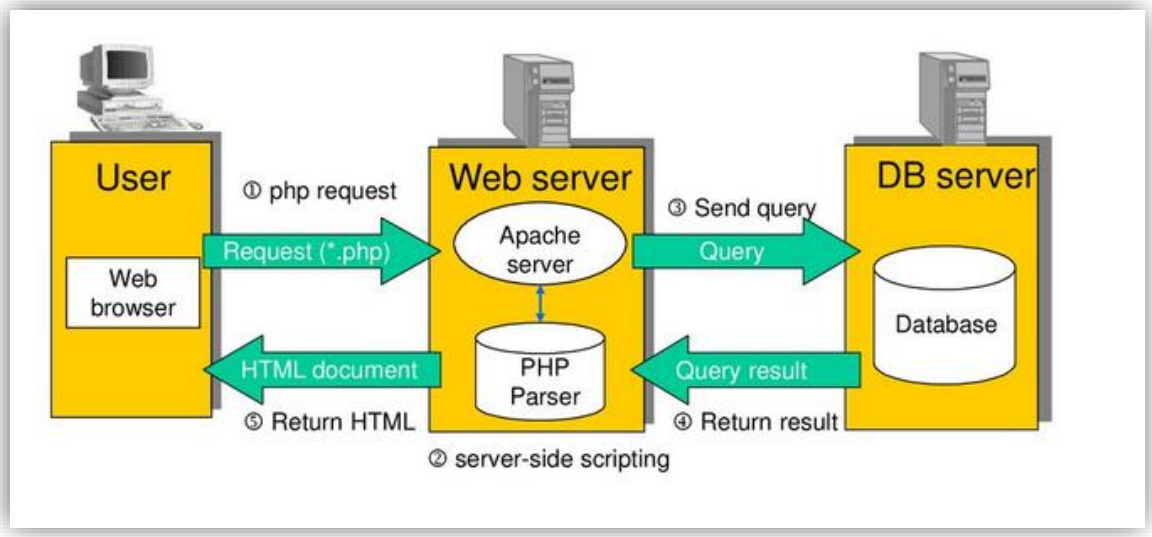
- **SSLEngine:** Activa el soporte para HTTPS con certificados SSL/TLS.
- **MaxClients:** Define el número máximo de conexiones simultáneas que el servidor puede manejar.
- **Timeout:** Establece el tiempo de espera antes de cerrar una solicitud inactiva.

4. Funcionamiento del Server Side

Cuando un usuario *accede a una página web*, su navegador ***1. envía una solicitud HTTP*** al servidor, que puede incluir parámetros, datos de formularios o cookies. El servidor ***2. interpreta la solicitud***, ***3. ejecuta código del lado servidor*** (como PHP, Node.js o Python), ***4. consulta bases de datos*** si es necesario y ***5. genera una respuesta en HTML, JSON o XML***. Luego, el servidor ***6. envía la respuesta al cliente***, que la renderiza en pantalla.

4.1. FUNCIONAMIENTO DEL SERVIDOR APACHE HTTP SERVER CON PHP

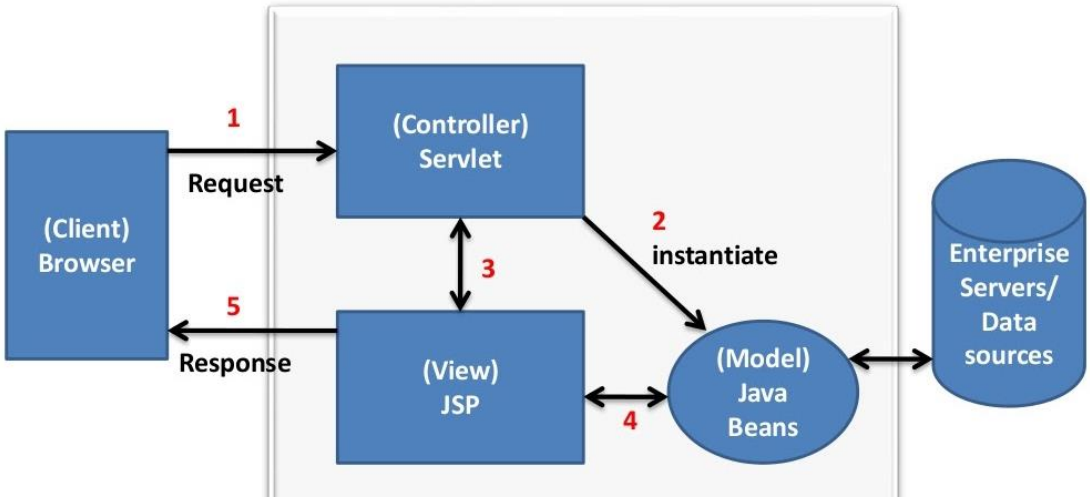
El **funcionamiento de Apache HTTP Server** con PHP **recibe solicitudes HTTP** y, si **detecta un archivo .php**, delega su procesamiento al **motor de PHP** (mediante ***mod_php o PHP-FPM***). **PHP interpreta el código**, ***ejecuta operaciones como manipulación de datos y consultas a bases de datos, y genera una respuesta en HTML o JSON***. Luego, Apache encapsula el contenido en un formato HTTP y lo envía al navegador del usuario.



4.2. FUNCIONAMIENTO DEL SERVIDOR APACHE TOMCAT SERVER CON JSP

El servidor Apache Tomcat es un **contenedor de servlets** y JSP que permite ejecutar aplicaciones web basadas en Java.

Su funcionamiento se basa en la gestión de solicitudes HTTP, donde Tomcat recibe peticiones de clientes y las procesa mediante su **motor de servlets**. Cuando un usuario accede a una página JSP, **Tomcat traduce el código JSP a un servlet Java, lo compila y ejecuta, generando una respuesta en HTML** que se envía al navegador.



5. Ecosistema Server Side

¿Qué se requiere para Desarrollar Aplicaciones del Server Side?

- Sistema Operativo
- Software de Servidor Web
- Lenguaje de Programación Backend Compilado o Script
- Sistema Gestor de BD
- Herramientas de desarrollo

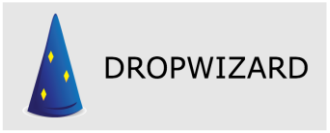
Lenguage Backend












Existen más tecnologías del lado backend.

Lenguage Backend

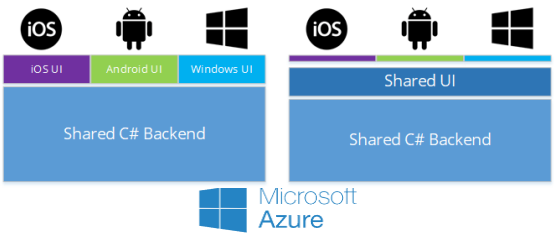










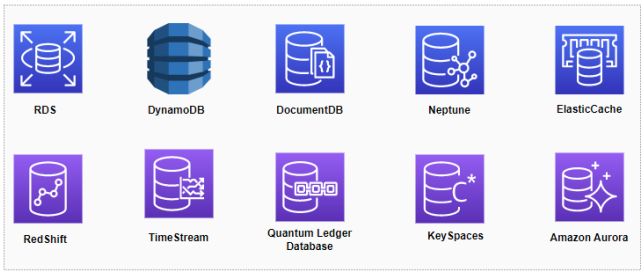

Traditional

Xamarin Forms

Microsoft Azure

Generalmente, los gestores de base de datos mas usandos son:

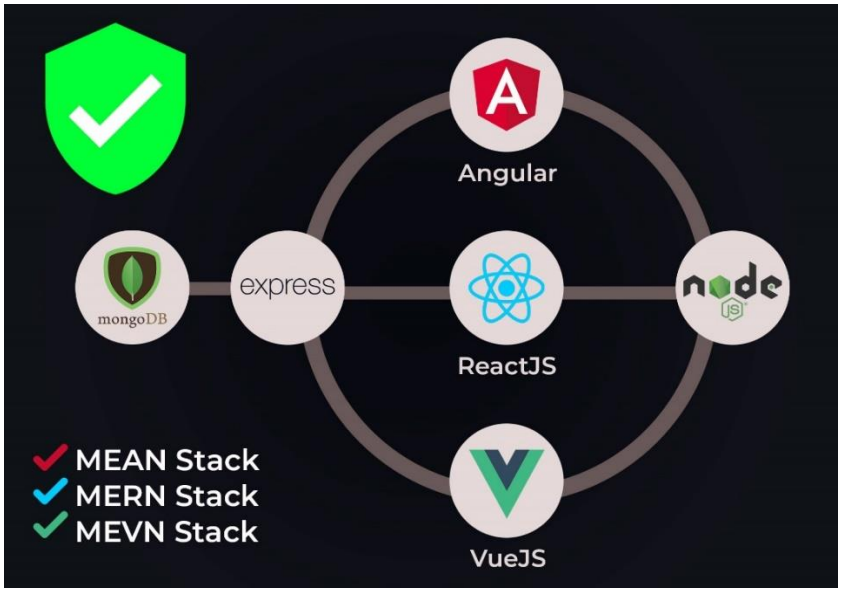
Gestores de BD



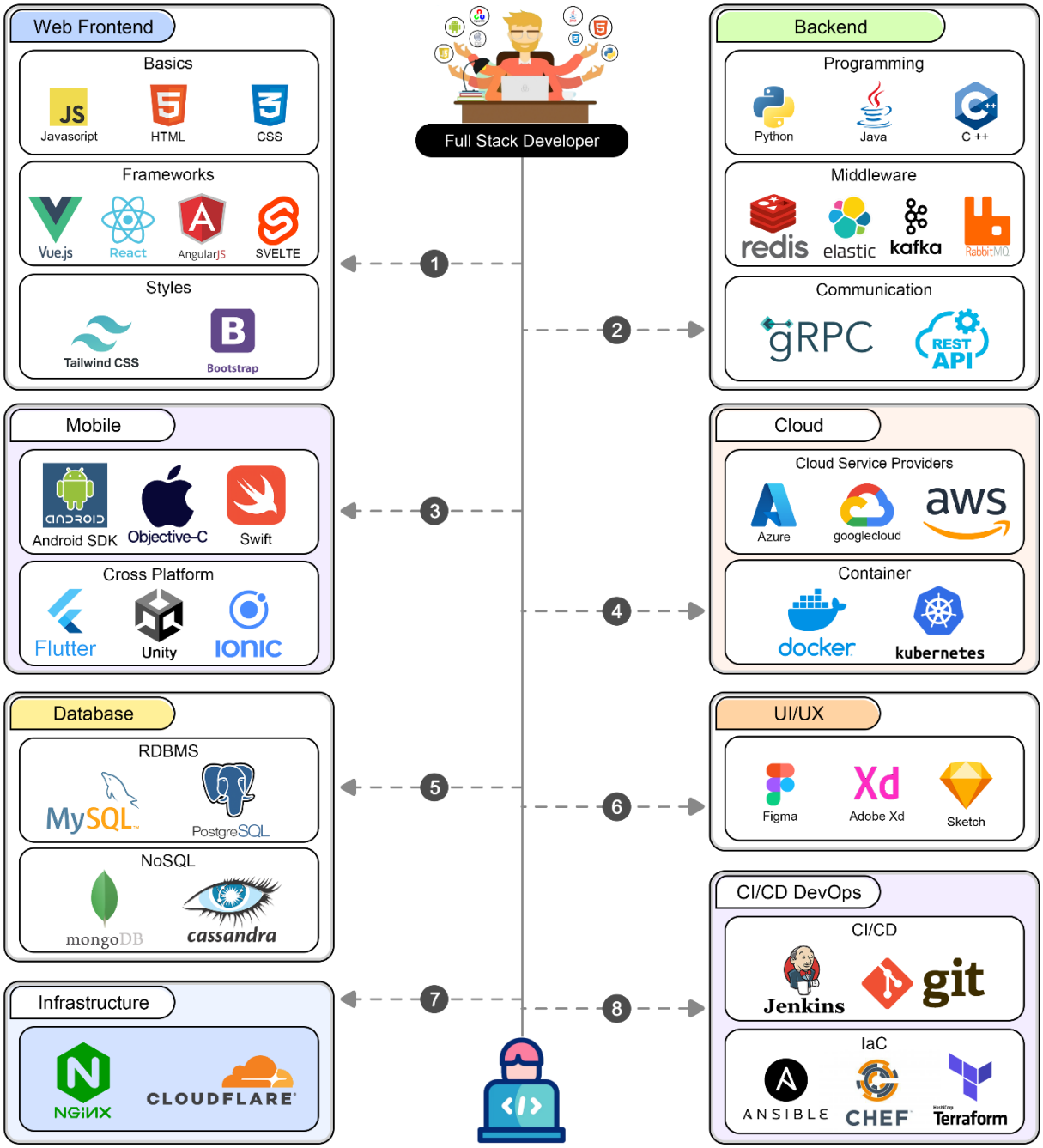
6. Stack Tecnológico en desarrollo web?

Es el ***conjunto de herramientas, lenguajes de programación, frameworks y bases de datos*** que se utilizan para construir y ejecutar aplicaciones web. Se divide en:

Stack Front-End



Stack Back-End



7. Servidor Web Apache Tomcat

Apache Tomcat (también llamado Jakarta Tomcat o simplemente Tomcat) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation.

Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Oracle Corporation (aunque creado por Sun Microsystems).
Tomcat es desarrollado y actualizado por miembros de la *Apache Software Foundation* y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software License.

Las primeras distribuciones de Tomcat fueron las versiones 3.0.x. A partir de la versión 4.0, Jakarta Tomcat utiliza el contenedor de servlets conocido como ***Catalina***.

Las versiones más recientes son las 9.x, que implementan las especificaciones de Servlet 4.0 y de JSP 2.3.

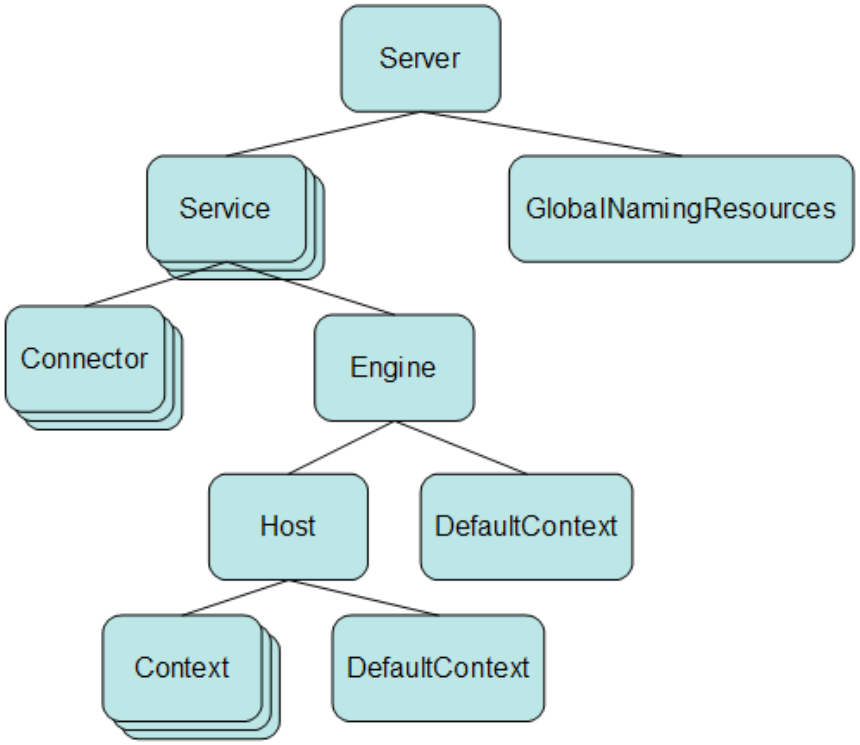
La Estructura de Directorios de Tomcat

La distribución de Tomcat está dividida en los siguientes directorios:

- **bin:** ejecutables y scripts para arrancar y parar Tomcat.
- **common:** clases y librerías compartidas entre Tomcat y las aplicaciones web. Las clases se deben colocar en common/classes, mientras que las librerías en formato JAR se deben poner en common/lib.
- **conf:** ficheros de configuración.
- **logs:** directorio donde se guardan por defecto los logs.
- **server:** las clases que componen Tomcat.
- **shared:** clases compartidas por todas las aplicaciones web.
- **webapps:** directorio usado por defecto como raíz donde se colocan todas las aplicaciones web.
- **work y temp:** directorios para almacenar información temporal

Módulos de Tomcat

Tomcat está compuesto por una serie de módulos cuyo comportamiento es altamente configurable. Incluso se pueden cambiar las clases que utiliza Tomcat por clases propias modificando el fichero `server.xml`. La estructura general de dichos módulos se muestra en la siguiente figura.



Cada módulo viene representado por su correspondiente etiqueta en el fichero server.xml.

- Server: es el propio Tomcat. Solo existe una instancia de este componente.
- GlobalNamingResources: sirve para definir mapeados de JNDI globales a todas las aplicaciones.
- Listener: monitoriza la creación y eliminación de contenedores web
- Service: un objeto de este tipo representa el sistema formado por un conjunto de conectores (connector) que reciben las peticiones de los clientes y las pasan a un engine, que las procesa. Por defecto viene definido el servicio llamado Tomcat-Standalone.
- Connector: acepta ciertos tipos de peticiones para pasarlas al engine. Por defecto, Tomcat incorpora un conector HTTP/1.1 (sin SSL) por el puerto 8080, y otro para comunicación con otros servidores (como Apache). Para cambiar el puerto por el que Tomcat acepta las peticiones HTTP basta con cambiar el atributo port de dicho connector.
- Engine: representa al contenedor web.
- Host: representa un host (o un host virtual). Mediante appBase se especifica el directorio de donde "colgarán" las aplicaciones web (por defecto webapps)
- Context: representa una aplicación web. Veremos de manera más detallada su configuración.
- DefaultContext: se aplica por defecto a aquellas aplicaciones que no tienen context propio.

8. Gestión de Dependencias con Maven

Desde hace varios años todo el mundo utiliza un gestor para construir un proyecto y obtener sus dependencias. Ahora ya no podemos vivir sin estas tecnologías, pero antes se tenían que importar todas las dependencias a mano y meter a un nuevo desarrollador

en un proyecto se podía convertir en un infierno hasta que tenía todo el entorno configurado para empezar a trabajar. https://maven.apache.org/

Hoy entrar en un proyecto es fácil: descargarlo y compilarlo para tener todo funcionando.

Hay dos gestores que se han convertido en los más utilizados: *Maven* y *Gradle*.

Introducción a Gradle

Gradle también es una herramienta de software libre lanzada en 2007 ***pero sirve para gestionar proyectos Java, Groovy o Scala***, aunque ya soporta otros lenguajes. Esta herramienta se configura ***con ficheros DSL basados en Groovy***, en vez de los xml que usaba Maven.

Herramienta Maven

Maven es ***una herramienta de software libre para gestionar proyectos Java*** lanzada en 2002. Se basa ***en archivos xml***. La filosofía de Maven es la estandarización de las estructuras de aplicaciones y de su Configuración, a fin de utilizar modelos existentes en la producción de software.

Con Maven se puede:

- Gestionar las dependencias del proyecto, para descargar e instalar módulos, paquetes y herramientas que sean necesarios para el mismo.
- Compilar el código fuente de la aplicación de manera automática.
- Empaquetar el código en archivos .jar o .zip.
- Instalar los paquetes en un repositorio (local, remoto o en el central de la empresa)
- Generar documentación a partir del código fuente.
- Gestionar las distintas fases del ciclo de vida de las ***build***: validación, generación de código fuente, procesamiento, generación de recursos, compilación, ejecución de test ...

En Maven se puede usar con multitud de elementos denominados ***plugins*** construidos por la comunidad de desarrolladores. Ejemplo: hay plugins que se pueden usar para funciones como desplegar deploy el proyecto en un servidor

Repositorio Maven

Maven también, proporciona un repositorio que contiene librerías de código, plugins y utilidades reutilizables. https://mvnrepository.com/

El repositorio de Maven (o repositorio central) tiene una estructura que permite descargar y almacenar los archivos JAR y WAR en versiones distintas.

Partes del Ciclo de vida

Las partes del ciclo de vida principal del proyecto Maven son:

- ***compile:*** Genera los ficheros .class compilando las fuentes .java
- ***test:*** Ejecuta los test automáticos de JUnit existentes, abortando el proceso si alguno de ellos falla.
- ***package:*** Genera el fichero .jar con los .class compilados

- ***install:*** Copia el fichero .jar a un directorio de nuestro ordenador donde maven deja todos los .jar. De esta forma esos .jar pueden utilizarse en otros proyectos maven en el mismo ordenador.
- ***deploy***: Copia el fichero .jar a un servidor remoto, poniéndolo disponible para cualquier proyecto maven con acceso a ese servidor remoto.

Cuando se ejecuta cualquiera de los comandos maven, por ejemplo, si ejecutamos:

```
> mvn install
```

Maven irá verificando todas las fases del ciclo de vida desde la primera hasta la del comando, ejecutando solo aquellas que no se hayan ejecutado previamente.

¿Qué es el archivo pom.xml?

La unidad básica de trabajo en Maven es el llamado Modelo de Objetos de Proyecto conocido simplemente como POM (de sus siglas en inglés: Project Object Model).

Se trata de un archivo XML llamado pom.xml que se encuentra por defecto en la raíz de los proyectos y que contiene toda la información del proyecto: su configuración, sus dependencias, etc... ()

- El hecho de utilizar un archivo XML revela su edad. Maven se creó en 2002, cuando XML era lo más. Si se hubiese creado unos pocos años después seguramente tendríamos un pom.json y si hubiese sido más reciente un pom.yml. Modas que vienen y van. No obstante, el formato XML, aunque engorroso, es muy útil a la hora de definir con mucho detalle cómo debe ser cada propiedad y, también, para poder comprobarlas.

Ejemplo de un archivo pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>web2</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>20</maven.compiler.source>
        <maven.compiler.target>20</maven.compiler.target>
        <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    </properties>

</project>
```

Integración con IDE

Maven sustituye el entorno integrado de desarrollo (IDE por sus siglas en inglés), por tanto la integración con diferentes IDEs es muy importante. Existen plugins de Maven para crear archivos de configuración del IDE a partir de los POMs. Actualmente se soportan:

- Eclipse
- Netbeans
- IntelliJ IDEA

- JDeveloper 11G (11.1.1.3)

Comandos Maven

Utilice el siguiente comando de Maven para compilar e implementar la aplicación web de plantilla en el servidor Tomcat. Una vez implementado, puede acceder a él a través de la url http://localhost:8080/my-webapp/:

Desplegar un proyecto:

```
$ mvn clean install tomcat7:deploy
```

Redesplegar un proyecto:

```
$ mvn clean install tomcat7:redeploy
```

Quitar un proyecto

```
$ mvn clean install tomcat7:undeploy
```

Segunda Unidad: Desarrollo Web Fronted
GUÍA DE LABORATORIO N° 09:
INSTALACIÓN DE JDK JAKARTA Y TOMCAT

Docente: Jaime Suasnabar Terrel

Fecha: …….. **Duración:** 90 minutos

Instrucciones: Instalar JDK Jakarta, Intelli J Idea y Tomcat.

OBJETIVOS
- Conocer el funcionamiento de Java y Open JDK
- Instalar Open JDK
- Instalar Intelli J Idea.
- Instalar Servidor Web Tomcat

PROCEDIMIENTO

EJERCICIO 01. INSTALACIÓN DE JDK OPEN SOURCE Y SERVIDOR WEB

1. Instalación de JDK para Jakarta

Descargar los binarios de la página oficial de OpenJDK:

- Página principal: https://openjdk.java.net/
- Binario: https://jdk.java.net/19/

Se descomprime el archivo zip descargado.

Clic derecho sobre el archivo y Extraer todo.

- Seleccionamos la ubicación donde queremos que quede los binarios del JDK. En mi caso voy a utilizar: c:\java.
- Se ha creado una carpeta llamada jdk-19.0.2. Debemos recordar esta ruta porque la configuraremos en las variables de entorno.

Creación de las variables de entorno.

- Se etc debe crear la variable de entorno JAVA_HOME y se debe actualizar la variable de entorno PATH.
- La primera se utilizará para que java sepa dónde se encuentra la instalación del JDK y la segunda es para poder ejecutar los comandos de java (como javac, java,) desde cualquier lugar.

Probar Java y JDK.

- Ahora abre una terminal. Puedes comprobar la versión de Java y del compilador de Java ejecutando:

```
>java --version
```

- Y:

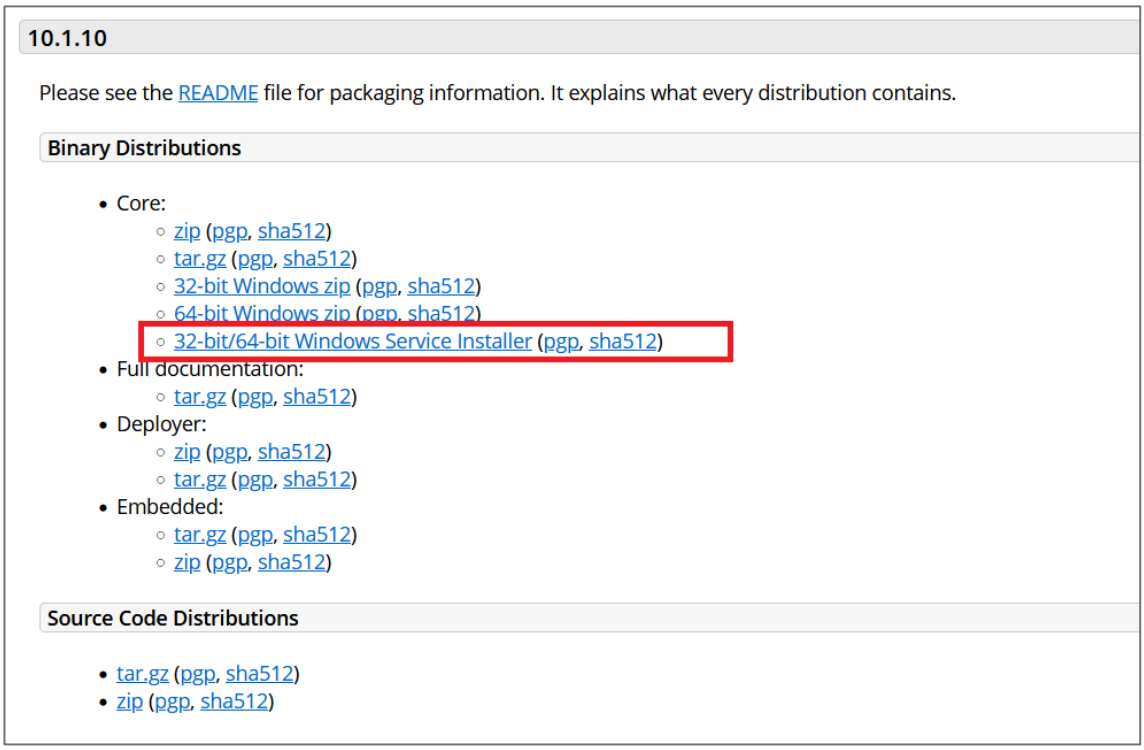
```
javac --version
```

2. TOMCAT – Instalación

Ingresar a la web: http://tomcat.apache.org/

Descargar el instalador estable más reciente

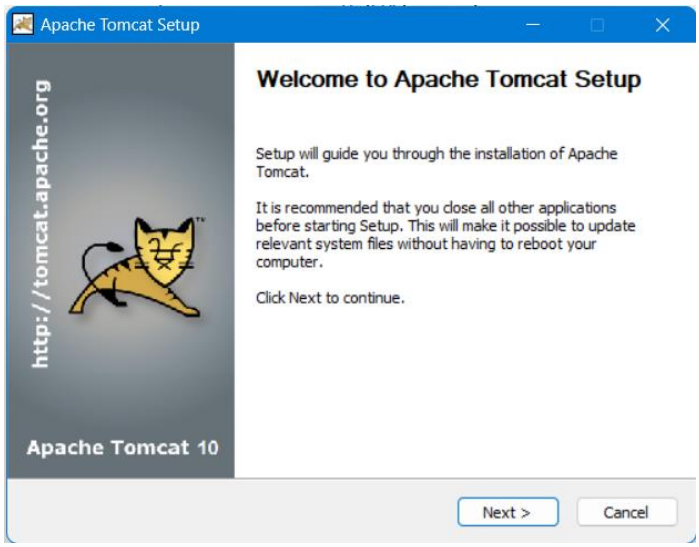
- Haz clic en Tomcat 10 en la barra lateral izquierda. Puedes encontrar esta opción debajo del título "Descargas" (Downloads) en un menú de navegación del lado izquierdo.
- Descarga 32-bit/64-bit Windows Service Installer debajo de "Core". Puedes encontrar esta opción en la sección "Distribuciones binarias" (Binary Distributions) de la parte inferior.



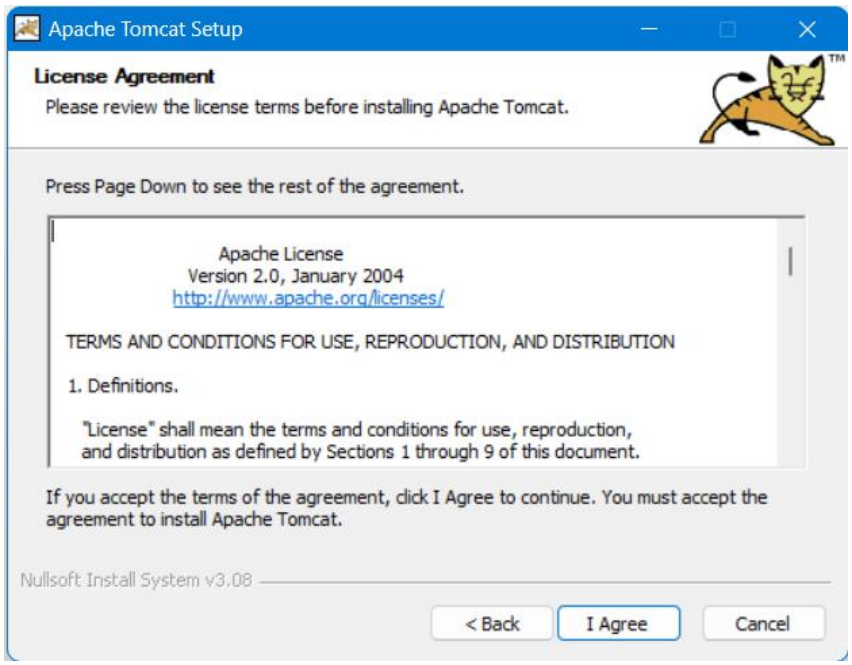
- Si se te pide, selecciona una ubicación para guardar el archivo de instalación.

Ejecutar el instalador de TOMCAT

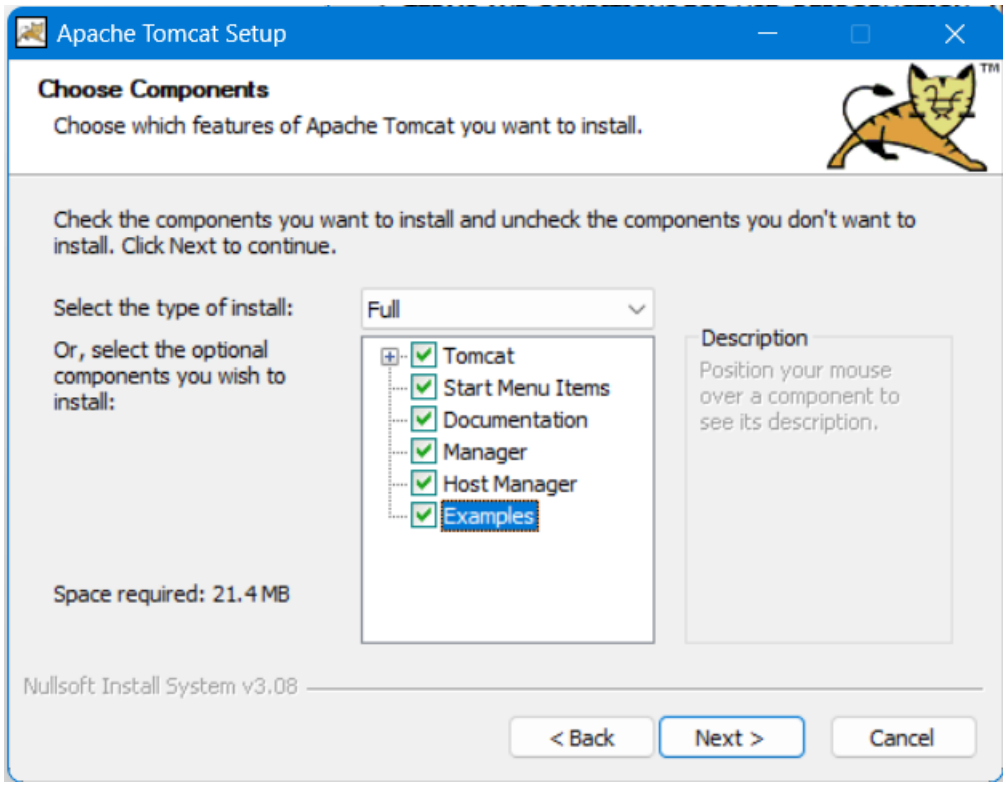
1. Inicia el archivo de instalación en la computadora. Busca el instalador en la carpeta "Descargas", y hazle doble clic para iniciar el asistente de instalación. Y Click en Siguiente.



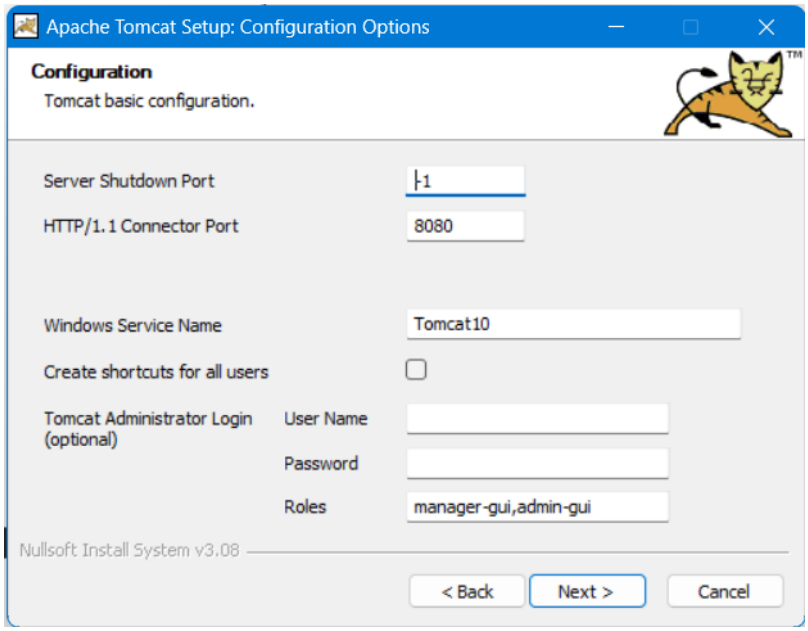
2. Abre el Acuerdo de licencia (License Agreement) en una nueva página. Click en I Agree acepto los acuerdos de licencia.



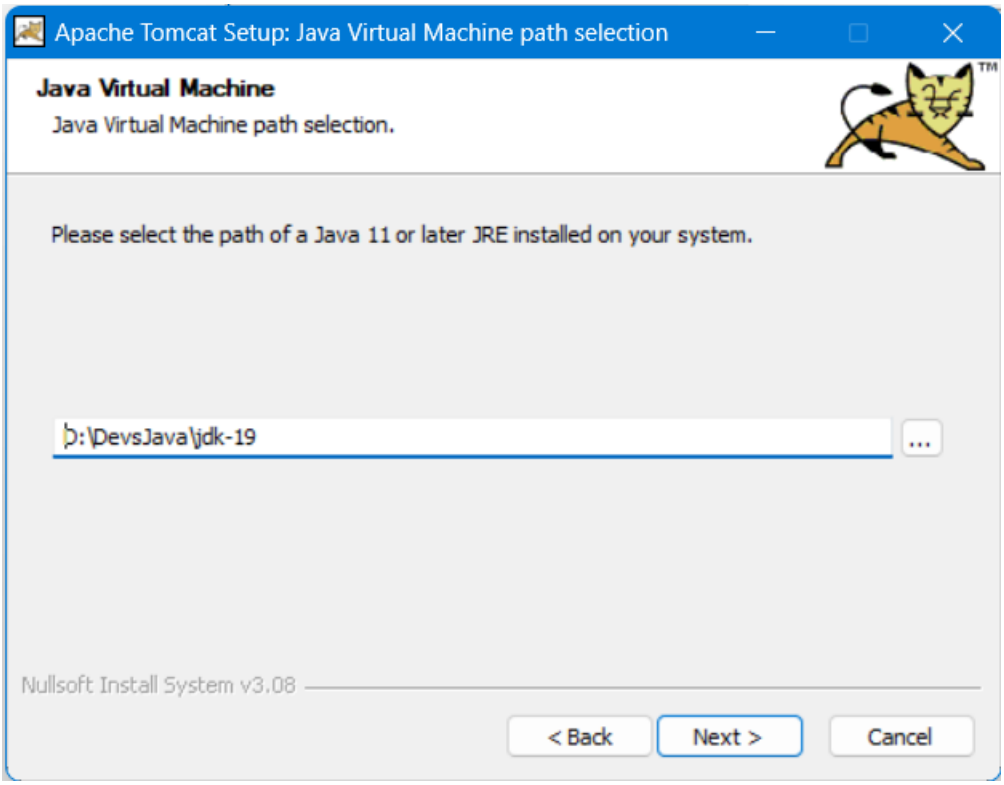
3. En la siguiente ventana. Selecciona el tipo de instalación" (Select the type of install) y selecciona Completo (**Full**) aquí para instalar todos los componentes de Tomcat, incluyendo la documentación y los accesos directos de la aplicación



4. Haz clic en Siguiente en configuración. A menos que vayas a personalizar los puertos, haz clic en Siguiente para continuar.



5. Especificar la ubicación que Java SE tendrá en la computadora en la siguiente página, elegir el botón … si desea examinar.



EJERCICIO 02. CREACIÓN DE PROYECTO WEB JAVA SERVER PAGE - JSP

1. Creación de Proyecto en IntelliJ IDEA

Descargar los binarios de la página oficial de OpenJDK:

- Página principal: https://openjdk.java.net/
- Binario: https://jdk.java.net/19/

Se descomprime el archivo zip descargado.

Clic derecho sobre el archivo y Extraer todo.

2. Configuración de Proyecto en IntelliJ IDEA

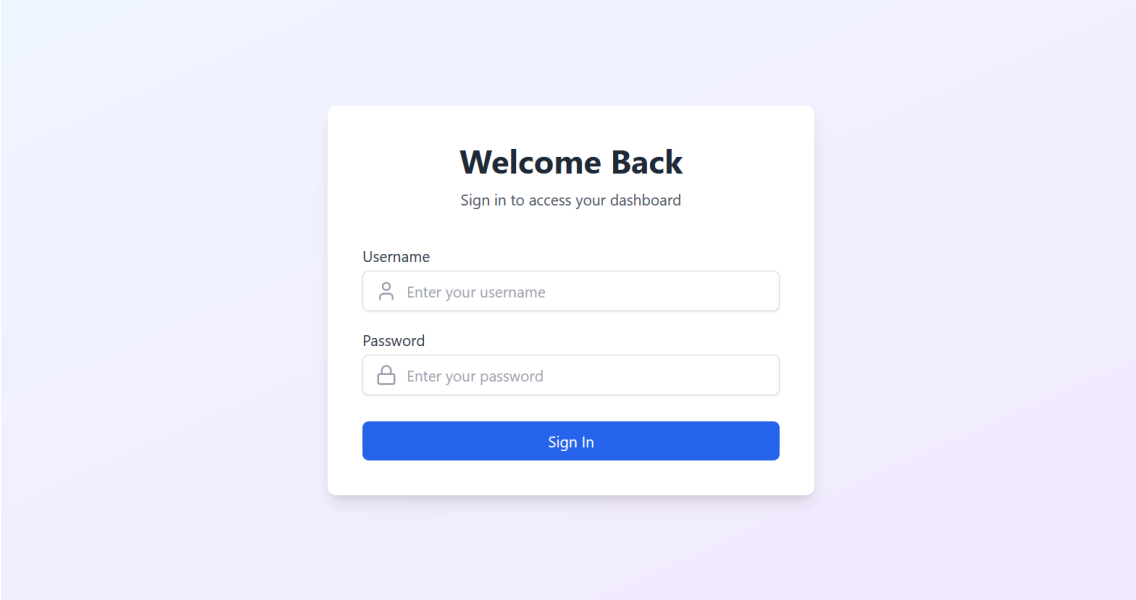
Descargar los binarios de la página oficial de OpenJDK:

- Página principal: https://openjdk.java.net/
- Binario: https://jdk.java.net/19/

Se descomprime el archivo zip descargado.

Clic derecho sobre el archivo y Extraer todo.

SISTEMA WEB



La autenticación basada en tokens es un método de verificación de identidad que permite a los usuarios acceder a sistemas sin necesidad de proporcionar credenciales constantemente. Se basa en la emisión de un token—un conjunto de datos cifrados—que valida la identidad del usuario por un período determinado.

Fundamentos clave:

1. **Generación del Token:** Cuando un usuario inicia sesión con sus credenciales, el sistema autentica su identidad y genera un token único.
2. **Uso del Token:** En lugar de enviar credenciales en cada solicitud, el usuario incluye el token en las peticiones al servidor.
3. **Validación del Token:** El servidor verifica que el token sea válido y esté vigente antes de conceder el acceso.
4. **Expiración y Renovación:** Los tokens suelen tener una duración limitada. Cuando expiran, el usuario debe obtener uno nuevo.
5. **Seguridad:** Generalmente, los tokens están cifrados y firmados digitalmente para evitar manipulación o falsificación.

Tipos de Tokens:

- **JSON Web Token (JWT):** Basado en JSON, se usa ampliamente en aplicaciones web y móviles.
- **OAuth Tokens:** Usados en autenticación delegada para acceder a APIs sin compartir credenciales.
- **SAML Tokens:** Comunes en entornos empresariales y federación de identidades.