Configuración de Servidores Locales y Cloud

Santiago Ismael Tigre Cajas
santiago.tigre@ucuenca.edu.ec

Universidad de Cuenca - Facultad de Ingeniería
Carrera de Computación - Programación Web
Cuenca, 1 de Abril del 2024

***Resumen*— Este trabajo tiene como objetivo entender la configuración de servidores web locales, así como también la configuración de servidores de aplicaciones. Además, se persigue entender la diferencia entre un servidor web local y un servidor que permita despliegue en la nube.**

I. Marco Teórico

- Servidor Web
  Un servidor web es un programa o software que se ejecuta en un ordenador (físico o virtual) y que se encarga de almacenar, procesar y entregar archivos de sitios web a los usuarios que los solicitan a través de un navegador web.

  - Es como una biblioteca gigante que guarda todos los archivos que componen una página web (texto, imágenes, videos, etc.).
  - Cuando un usuario escribe una dirección web en su navegador, éste envía una solicitud al servidor web para que le envíe esos archivos.
  - Es el encargado de procesar la solicitud y enviar los archivos al navegador, que los muestra al usuario en forma de página web.

  Aspectos importantes a tener en cuenta:

  - **Hardware**: El servidor web se ejecuta en un ordenador que debe tener suficiente potencia y capacidad de almacenamiento para alojar los archivos del sitio web y gestionar el tráfico de usuarios.
  - **Software**: El software del servidor web se encarga de la comunicación con los navegadores web, la gestión de archivos y la seguridad del sitio web. Algunos de los servidores web más populares son Apache, Nginx y IIS.
  - **Protocolo HTTP**: El protocolo HTTP (Hypertext Transfer Protocol) es el conjunto de reglas que define cómo se comunican los navegadores web con los servidores web.
  - **Tipos de servidores web**: Existen diferentes tipos de servidores web, como servidores web compartidos, servidores web dedicados y servidores en la nube.

  Ejemplos de servidores:

  - **Apache**: Es uno de los servidores web más populares y utilizados del mundo. Es gratuito, de código abierto y muy versátil.
  - **Nginx**: Es otro servidor web muy popular que se caracteriza por su alta velocidad y eficiencia.
  - **IIS**: Es el servidor web desarrollado por Microsoft para Windows.
  - **Amazon Web Services (AWS)**
  - **Google Cloud Platform (GCP)**

- Servidor de Aplicaciones
  Un servidor de aplicaciones es un software que se ejecuta en un ordenador (físico o virtual) y que se encarga de alojar y ejecutar aplicaciones web. Las aplicaciones web son programas que se ejecutan en un navegador web, en lugar de en un ordenador local.

  - Un servidor de aplicaciones es como una plataforma que permite ejecutar aplicaciones web.
  - Las aplicaciones web se desarrollan utilizando diferentes tecnologías, como Java, Python o PHP.

- El servidor de aplicaciones proporciona los recursos necesarios para que las aplicaciones web se ejecuten correctamente.

Aspectos importantes a tener en cuenta:

- **Hardware**: El servidor de aplicaciones se ejecuta en un ordenador que debe tener suficiente potencia y capacidad de almacenamiento para alojar las aplicaciones web y gestionar el tráfico de usuarios.
- **Software**: El software del servidor de aplicaciones se encarga de la comunicación con los navegadores web, la gestión de las aplicaciones web y la seguridad del servidor. Algunos de los servidores de aplicaciones más populares son Apache Tomcat, JBoss y GlassFish.
- **Tipos de servidores de aplicaciones**: Existen diferentes tipos de servidores de aplicaciones, como servidores de aplicaciones web, servidores de aplicaciones empresariales y servidores de aplicaciones en la nube.

Ejemplos de servidores:

- **Apache Tomcat**: Es un servidor de aplicaciones web gratuito y de código abierto muy popular. Es compatible con Java y se utiliza para ejecutar aplicaciones web como JSP y servlets.
- **JBoss**: Es un servidor de aplicaciones empresariales de código abierto que ofrece una amplia gama de características para el desarrollo y la implementación de aplicaciones web complejas.
- **GlassFish**: Es un servidor de aplicaciones web desarrollado por Oracle que es compatible con Java EE.

- Hosting
  El hosting, también conocido como alojamiento web, es un servicio que proporciona espacio en un servidor para que puedas almacenar los archivos de tu sitio web o aplicación web.

  - Es como alquilar un espacio en un edificio para guardar muebles.
  - El servidor es el edificio y el hosting es el espacio que alquilas dentro del edificio.
  - Los archivos de tu sitio web o aplicación web son los muebles.

  Aspectos importantes a tener en cuenta:

  - **Tipos de hosting**: Existen diferentes tipos de hosting, como hosting compartido, hosting VPS, hosting dedicado y hosting en la nube.
  - **Capacidad de almacenamiento**: La cantidad de espacio que necesitas dependerá del tamaño de tu sitio web o aplicación web.
  - **Ancho de banda**: El ancho de banda es la cantidad de datos que se pueden transferir entre el servidor y los usuarios.
  - **Correo electrónico**: Algunos proveedores de hosting también ofrecen servicios de correo electrónico.
  - **Soporte técnico**: Es importante elegir un proveedor de hosting que ofrezca un buen soporte técnico.

Proveedores de Hosting:

- **GoDaddy**: Es uno de los proveedores de hosting más populares del mundo. Ofrece una amplia gama de servicios de hosting a precios competitivos.
- **Bluehost**: Es otro proveedor de hosting muy popular que ofrece planes de hosting compartido a precios muy económicos.
- **HostGator**: Es un proveedor de hosting que ofrece una buena relación calidad-precio.
- **Amazon Web Services (AWS)**
- **Google Cloud Platform (GCP)**

- Cloud Computing
  Conocido como computación en la nube, es un modelo de entrega de servicios de TI a través de Internet. En lugar de instalar y ejecutar aplicaciones en tus propios servidores, puedes acceder a ellas a través de la nube.

  - Es como alquilar una computadora en lugar de comprarla.
  - La computadora está en la nube y puedes acceder a ella desde cualquier dispositivo con conexión a Internet.
  - Puedes usar la computadora para ejecutar aplicaciones, almacenar archivos y mucho más.

  Aspectos importantes a tener en cuenta:

  - **Tipos de servicios en la nube**: Existen diferentes tipos de servicios en la nube, como Infrastructure as a Service (IaaS), Platform as a Service (PaaS) y Software as a Service (SaaS).
  - **Escalabilidad**: La nube te permite escalar tus recursos de TI de forma rápida y sencilla.
  - **Costo**: La nube puede ser una forma más económica de obtener recursos de TI que comprar e instalar tu propio hardware y software.
  - **Seguridad**: Es importante elegir un proveedor de servicios en la nube que ofrezca un alto nivel de seguridad.

  Proveedores de Cloud Computing

  - **Amazon Web Services (AWS)**
  - **Microsoft Azure**
  - **Google Cloud Platform (GCP)**

- Despliegue continuo
  El Despliegue Continuo (CD) es una práctica de DevOps que automatiza la entrega de código a producción. Esto significa que los cambios en el código se pueden probar e implementar de forma rápida y segura, sin necesidad de intervención manual.

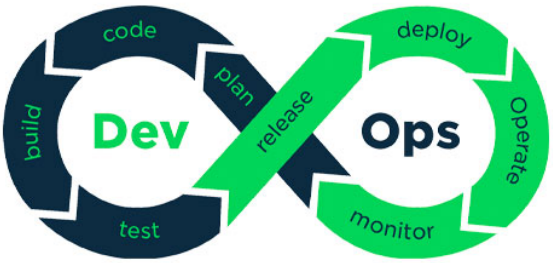

Fig. 1. Funcionamiento DevOps.

- Es como una tubería que lleva el código desde el desarrollo hasta la producción.
- La tubería está automatizada, por lo que el código puede fluir de forma rápida y segura.
- Esto permite a las empresas lanzar nuevas funciones y corregir errores con mayor frecuencia.

Aspectos importantes a tener en cuenta:

- **Automatización**: La automatización es clave para el Despliegue Continuo. Se utilizan herramientas para automatizar las tareas de prueba, integración e implementación.
- **Pruebas**: Es importante realizar pruebas exhaustivas del código antes de implementarlo en producción.
- **Integración**: El código debe integrarse con otros sistemas antes de implementarlo en producción.
- **Implementación**: El código debe implementarse en producción de forma segura y sin interrupciones.

Beneficios del Despliegue Continuo:

- **Aumento de la velocidad de entrega**: Las empresas pueden lanzar nuevas funciones y corregir errores con mayor frecuencia.
- **Mejora de la calidad del software**: La automatización de las pruebas ayuda a mejorar la calidad del software.
- **Reducción de costes**: El Despliegue Continuo puede ayudar a reducir los costes de desarrollo y producción.
- **Mayor satisfacción del cliente**: Los clientes pueden acceder a nuevas funciones y correcciones de errores con mayor rapidez.

Herramientas para el Despliegue Continuo:

- **Jenkins**: Es una herramienta de automatización de código abierto muy popular.
- **Travis CI**: Es otra herramienta de automatización de código abierto muy popular.
- **CircleCI**: Es una herramienta de automatización de pago.
- **Azure DevOps**: Es una herramienta de automatización de Microsoft.

II. Servidor Local

Para realizar la configuración de los diferentes tipos de servidores locales se ha optado por usar XAMPP, dado que en este se pueden realizar las configuraciones de ambos tipos.

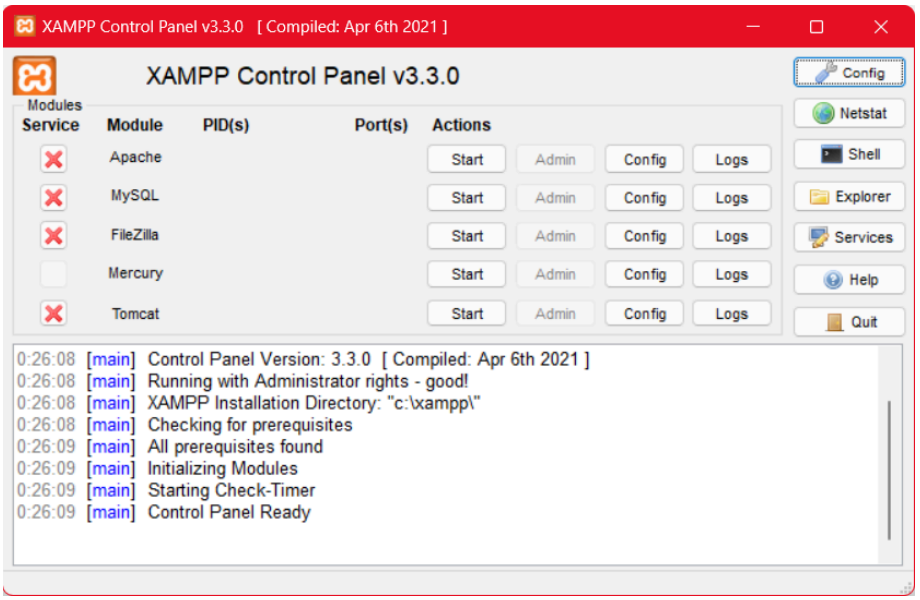

Fig. 2. Interfaz Principal de XAMPP.

A. Apache

a. Configuración Puerto

Para realizar la configuración del puerto es necesario editar el archivo **httpd.conf**, se puede ingresar al mismo mediante el botón **Config**, y nos debemos ubicar las siguientes líneas dentro del archivo.

Archivo	httpd.conf

```
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.

# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.

#Listen 12.34.56.78:80
Listen 8086
```

Aquí se puede poner el puerto determinado para el servidor apache que vamos a ocupar, en este ejemplo se utilizara el puerto 8086.

b. Configuración Carpeta Pública

Para realizar la configuración de la carpeta compartida igualmente se debe editar el archivo **httpd.conf** y debemos ubicar las siguientes líneas dentro del archivo.

Archivo	httpd.conf

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
```

```
DocumentRoot "C:\Users\saant\Escritorio\Programación Web\Trabajo 1\Carpeta Publica"
<Directory "C:\Users\saant\Escritorio\Programación Web\Trabajo 1\Carpeta Publica">
```

Aquí es donde se configura la carpeta donde se van a almacenar los archivos del servidor, se debe colocar la misma ruta a la carpeta en la sección **DocumentRoot** y **Directory**.

Una vez realizadas las configuraciones necesarias para nuestro servidor local, en la interfaz de XAMPP el servicio que nos interesa es **Apache**, lo iniciamos haciendo clic en **Start**, y en el log de eventos nos debería mostrar el siguiente mensaje:

```
[Apache]    Attempting to start Apache app...
[Apache]    Status change detected: running
```

Y en la interfaz deberíamos ver el servicio ejecutándose y mostrando el puerto que configuramos anteriormente.

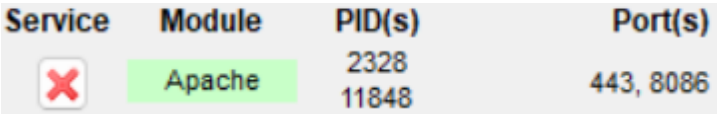

Fig. 3. Servicio Iniciado y Puertos.

Ahora podemos verificar el funcionamiento del servidor local dandole al boton **admin** dentro de la interfaz o ingresando "**localhost:#puerto**" dentro de cualquier navegador.

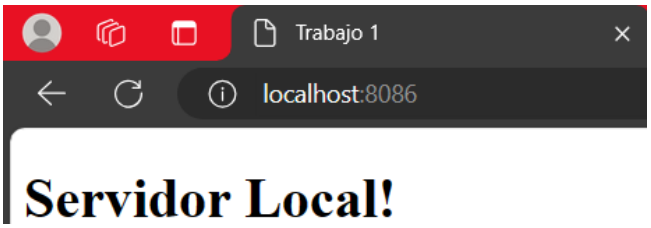

Fig. 4. Página Web.

B. Apache Tomcat

a. Configuración Puerto

Para realizar la configuración del puerto es necesario editar el archivo **server.xml**, se puede ingresar al mismo mediante el botón **Config**, y debemos ubicar las siguientes líneas dentro del archivo.

Archivo	server.xml

```
<Connector port="8084" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           />
```

Aquí se puede poner el puerto determinado para el servidor apache que vamos a ocupar, en este ejemplo se utilizara el puerto 8084.

b. Configuración Carpeta Pública

Para realizar la configuración de la carpeta compartida igualmente se debe editar el archivo **server.xml** y debemos ubicar las siguientes líneas dentro del archivo.

Archivo	server.xml
<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true"> **<Context path="" docBase="C:/Users/saant/Escritorio/Programación Web/Trabajo 1/Carpeta Publica"/>**	

Aquí es donde se configura la carpeta donde se van a almacenar los archivos del servidor, para esto se debe agregar la línea que está en negrita con la ruta de nuestra carpeta.

Una vez realizadas las configuraciones necesarias para nuestro servidor local, en la interfaz de XAMPP el servicio que nos interesa es **TomCat**, lo iniciamos haciendo clic en **Start**, y en el log de eventos nos debería mostrar el siguiente mensaje:

[Tomcat] Attempting to start Tomcat app...

Y en la interfaz deberíamos ver el servicio ejecutándose y mostrando el puerto que configuramos anteriormente.

Fig. 3. Servicio Iniciado y Puertos.

Ahora podemos verificar el funcionamiento del servidor local dandole al boton **admin** dentro de la interfaz o ingresando "**localhost:#puerto**" dentro de cualquier navegador.

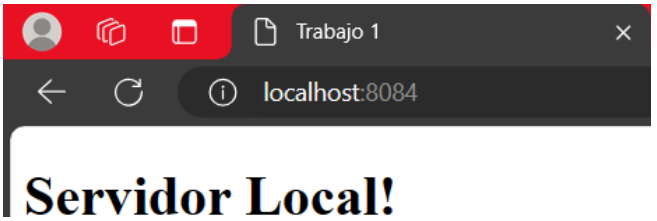

Fig. 4. Página Web.

III. Servidor Cloud

Para desplegar un servidor cloud vamos a usar Microsoft Azure.

- **¿Qué es Microsoft Azure?**
  Azure es una plataforma de computación en la nube creada por Microsoft. Ofrece un amplio conjunto de servicios para construir, implementar y administrar aplicaciones, servicios y recursos en la nube.
- **¿Para qué sirve?**
  - Desarrollar y ejecutar aplicaciones: Crea aplicaciones web, móviles, de escritorio y back-end utilizando una amplia gama de lenguajes de programación y herramientas.
  - Almacenar y administrar datos: Almacena datos en la nube de forma segura y escalable con diferentes opciones de bases de datos, como SQL Server, Cosmos DB y Blob Storage.
  - Analizar datos: Obtén información valiosa de tus datos con herramientas de análisis como Azure Data Lake Analytics y Power BI.
  - Implementar infraestructura de TI: Crea y administra máquinas virtuales, redes, almacenamiento y otros recursos de TI.
  - Proteger tus datos y aplicaciones: Protege tu información con las sólidas funciones de seguridad de Azure, como Azure Security Center y Azure Key Vault.
  - Aprovechar la inteligencia artificial: Implementa soluciones de inteligencia artificial y aprendizaje automático con Azure Cognitive Services, Bot Service y Machine Learning Studio.
- **Costos**
  Azure tiene un modelo de precios de pago por uso, lo que significa que solo pagas por los recursos que consumes. Puedes elegir entre diferentes planes de precios según tus necesidades, como planes por hora, por minuto o por mes. También hay planes gratuitos para algunos servicios. Para este trabajo optamos por utilizar el plan para estudiantes, que además nos ofrece $100 dentro de la plataforma.

  Para más información revisar el siguiente enlace https://azure.microsoft.com/es-mx/pricing

- **Restricciones de tecnología**
  Azure es una plataforma flexible y escalable, pero hay algunas restricciones que debes tener en cuenta:

  - Disponibilidad de servicios: No todos los servicios de Azure están disponibles en todas las regiones.
  - Límites de recursos: Hay límites en la cantidad de recursos que puedes usar, como CPU, memoria y almacenamiento.
  - Compatibilidad: Es posible que algunos servicios no sean compatibles con todas las plataformas o tecnologías.

- **Tipos de servicios**
  Azure ofrece una amplia gama de servicios, que se pueden agrupar en las siguientes categorías:

  - Computación: Máquinas virtuales, contenedores, Azure Batch, etc.
  - Almacenamiento: Blob Storage, Disk Storage, SQL Database, etc.

- Redes: Virtual Network, Azure VNet, Load Balancing, etc.
- Bases de datos: SQL Database, Cosmos DB, MySQL Database, etc.
- Análisis: Data Lake Analytics, Power BI, Stream Analytics, etc.
- Inteligencia artificial: Cognitive Services, Bot Service, Machine Learning Studio, etc.
- Seguridad: Azure Security Center, Key Vault, Identity Manager, etc.
- DevOps: Azure DevOps, Visual Studio Team Services, etc.
- IoT: IoT Hub, Device Provisioning Service, Stream Analytics, etc.
- Medios: Media Services, Video Analyzer, etc.

- **Creación de Aplicación Web**

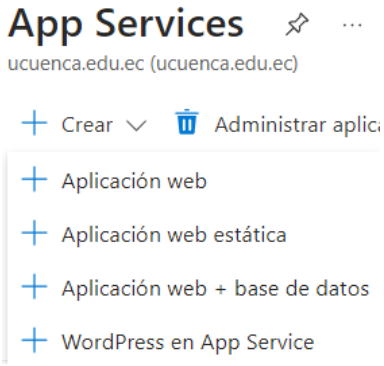

Fig. 5. Azure App Services.

Podemos hacer uso de los diferentes tipos de implementaciones de **App Services** que nos ofrece Azure, para esta demostración utilizamos **Aplicación Web** y **Aplicación Web Estática**.

Dentro de ambas opciones se hizo uso de la suscripción para estudiantes que se mencionó anteriormente, y a las que les aplicó las siguientes configuraciones.

Fig. 6. Opciones de Publicación.

Dentro de las opciones de publicación variará el modo que se desee realizar, si es con código después de la implementación se deberá proveer en el centro de implementación el recurso de código desde el cual obtendrá los requerimientos, de otra manera si se realiza con aplicación web estática se realizará en ese momento y no se podrá cambiar después en las configuraciones.

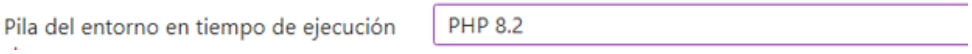

Fig. 7. Pila de Entorno.

La opción de la pila de entorno solo estará disponible para elegir cuando se vaya a publicar código.

Después de realizar estas configuraciones básicas se realiza la implementación del **App Service**, si se realizó con una página web estática esta se encontrará disponible al instante en cambio sí publicó mediante código es necesario configurar el medio. Para este ejemplo en los dos casos se realizó la implementación directamente desde un repositorio en GitHub.

Las páginas web y repositorio son accesibles desde los siguientes enlaces.

- Repositorio - https://github.com/saantitiger/Trabajo_1_PW
- Despliegue usando Código - https://saantitiger.azurewebsites.net o https://trabajo1.azurewebsites.net
- Despliegue usando Aplicación Web Estática https://happy-ground-02cbe2110.5.azurestaticapps.net

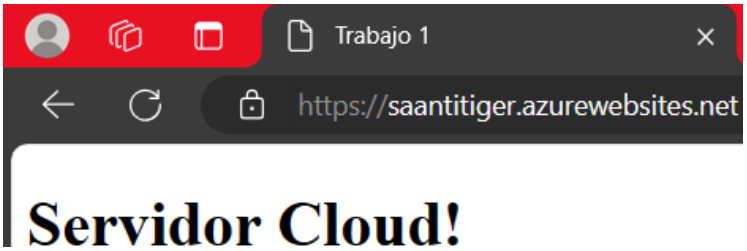

Fig. 8. Página Desplegada.

IV. Configuración Firewall

Para configurar el Firewall en Windows y que los servidores locales funcionen exclusivamente en los puertos que se desean es necesario la creación de una **Regla de Entrada** dentro de la configuración avanzada del Firewall con los siguientes configuraciones.

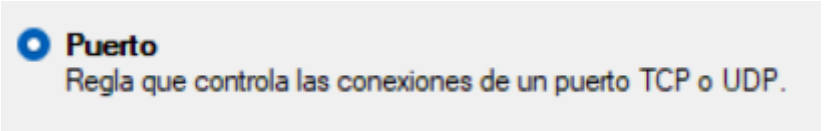

Fig. 9. Tipo de Regla.

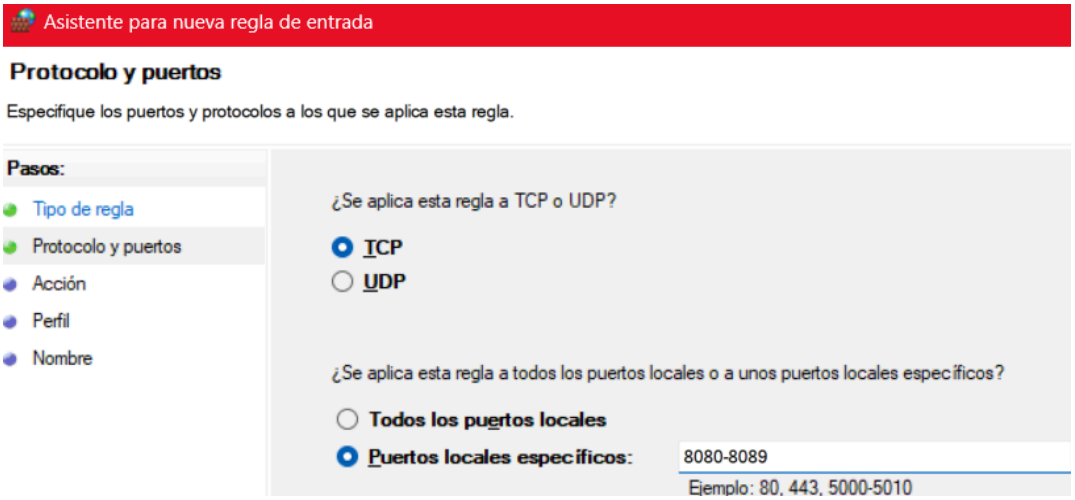

Fig. 10. Puertos Especificados.

Los demás pasos los dejamos por defecto y al final colocamos un nombre para nuestra regla, de esta manera estaremos indicando los puertos locales específicos para que sirvan exclusivamente los servidores locales.

V. Estructura de Archivos

Tener un orden en la estructura de carpetas de un proyecto web es fundamental para la eficiencia, la colaboración, el mantenimiento y la escalabilidad del proyecto, es por eso que se presenta la importancia y razones para tener una buena estructura.

1. Facilita la navegación y el acceso a los archivos:
- Una estructura de carpetas bien organizada permite a los desarrolladores y colaboradores encontrar rápidamente los archivos que necesitan, lo que reduce el tiempo de búsqueda y aumenta la eficiencia.
- Se evita la pérdida de tiempo y la frustración al buscar archivos perdidos o mal ubicados.

2. Mejora la colaboración:
- Un sistema de organización claro y consistente facilita la colaboración entre varios desarrolladores, ya que todos saben dónde encontrar los archivos y cómo se organizan.
- Se reduce la posibilidad de errores y confusiones al trabajar en equipo.

3. Simplifica el mantenimiento del proyecto:
- Una estructura ordenada facilita la adición, eliminación y modificación de archivos, ya que se tiene una visión clara de la organización del proyecto.
- Se reduce el riesgo de errores al realizar cambios en el proyecto.

4. Aumenta la legibilidad y la comprensión del código:
- Una estructura de carpetas que refleja la lógica del proyecto facilita la comprensión del código y la arquitectura del mismo.
- Se facilita la detección de errores y la refactorización del código.

5. Mejora la escalabilidad del proyecto:
- Una estructura ordenada facilita la expansión del proyecto a medida que se añaden nuevas funcionalidades o características.
- Se reduce el riesgo de que el proyecto se vuelva caótico y difícil de manejar.

Razones por las que se debe tener orden

1. Eficiencia: El tiempo dedicado a la búsqueda de archivos se reduce considerablemente, lo que permite una mayor productividad.

2. Claridad: La organización facilita la comprensión del proyecto y su funcionamiento.

3. Colaboración: El trabajo en equipo se vuelve más eficiente y se reduce la posibilidad de errores.

4. Mantenimiento: Se facilita la actualización y el mantenimiento del proyecto.

5. Escalabilidad: El proyecto puede crecer y adaptarse a nuevas necesidades sin perder su organización.

Una estructura de carpetas básica se puede ver de la siguiente manera:

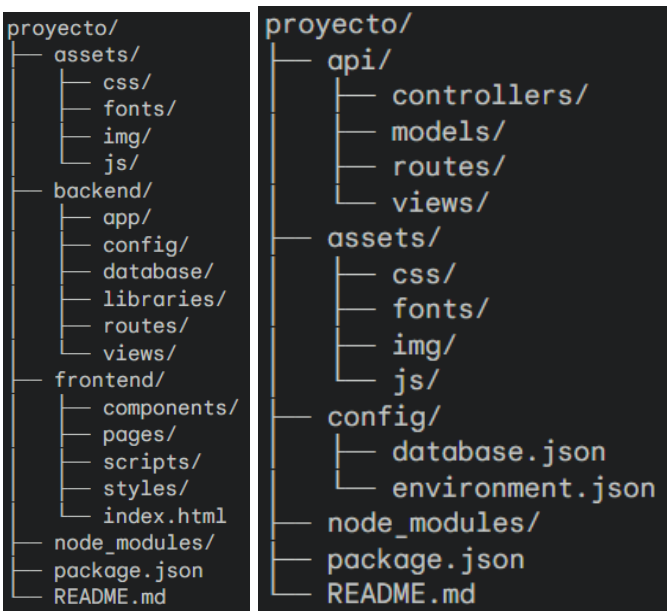

Fig. 11. Estructuras de Carpetas.

Explicación de las carpetas

- assets: Contiene todos los archivos estáticos del proyecto, como imágenes, hojas de estilo CSS, scripts JavaScript y fuentes.
- backend: Contiene el código del servidor del proyecto, que puede estar escrito en PHP, Python, Node.js, Java, entre otros lenguajes.
- frontend: Contiene el código del lado del cliente del proyecto, que generalmente está escrito en HTML, CSS y JavaScript.
- node_modules: Contiene las dependencias de Node.js del proyecto.
- package.json: Contiene información sobre el proyecto, como el nombre, la versión y las dependencias.
- README.md: Contiene un archivo README con información sobre el proyecto, como la instalación, el uso y la configuración.

Consideraciones adicionales

- Subcarpetas: Se pueden crear subcarpetas dentro de las carpetas existentes para organizar mejor el contenido.
- Versiones: Se pueden crear subcarpetas con el nombre de la versión para mantener diferentes versiones del proyecto.
- Entornos: Se pueden crear subcarpetas para diferentes entornos, como desarrollo, pruebas y producción.
- Herramientas de control de versiones: Se recomienda utilizar una herramienta de control de versiones como Git para mantener un historial del proyecto y facilitar la colaboración.

VI. Bibliografía

[1]
https://es.wikipedia.org/wiki/Servidor_web
https://www.webempresa.com/hosting/que-es-servidor-web.html
https://www.cloudcenterandalucia.es/blog/que-es-un-servidor-web-funcionamiento-y-tipos/
https://blog.hubspot.es/website/que-es-servidor-web
https://m.youtube.com/watch?v=ugn6fM5Rc-E
https://m.youtube.com/watch?v=ugn6fM5Rc-E
https://es.linkedin.com/pulse/estructuras-de-carpetas-carolina-romero
https://azure.microsoft.com/en-us/blog/
https://en.wikipedia.org/wiki/Continuous_deployment
https://www.atlassian.com/continuous-delivery
https://en.wikipedia.org/wiki/Cloud_computing
https://www.hostinger.es/tutoriales/que-es-un-hosting