CONFIGURACIÓN DE SERVIDORES

Calle Siavichay Mateo Sebastián
mateo.calle@ucuenca.edu.ec

Universidad de Cuenca – Computación
Cuenca, Ecuador

Resumen – Esta actividad 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.

Palabras Clave – *Programación Web, Servidor Web, Servidor de Aplicaciones, Hosting..*

I. Objetivos

A. *Objetivo General*

Desarrollar habilidades y competencias en la configuración de servidores web y de aplicaciones, así como en el despliegue de aplicaciones en la nube, mediante la práctica de configurar tres servidores web, implementar ajustes de configuración de puertos, firewall y estructura de carpetas.

B. *Objetivos específicos*

a) Configurar adecuadamente los tres servidores web, asegurando su correcto funcionamiento y acceso

b) Establecer la estructura de carpetas y recursos utilizados por las aplicaciones alojadas, facilitando así la gestión y mantenimiento de los servidores y sus contenidos.

c) Conocer acerca de los servidores cloud, lo que nos ofrecen y sus limitaciones.

II. Introducción

La configuración de servidores web es un aspecto fundamental en el desarrollo y despliegue de aplicaciones en entornos digitales. Los servidores web, como Apache XAMPP o WAMP, Apache Tomcat y las soluciones en la nube, actúan como pilares tecnológicos que permiten la entrega eficiente de contenido web y la ejecución de aplicaciones en línea. Desde la configuración de puertos y firewall hasta la organización de carpetas y la implementación de políticas de seguridad, cada aspecto de la configuración web desempeña un papel crucial en la estabilidad, el rendimiento y la seguridad de los servicios en línea.

La presente práctica se centra en el proceso de configuración de tres tipos de servidores web: Apache (WAMP o XAMPP), Apache Tomcat y un servidor en la nube. A través de este ejercicio, se busca no solo desarrollar habilidades técnicas en la configuración y administración de servidores, sino también comprender la importancia de cada aspecto de la configuración, desde la asignación de puertos hasta la gestión de carpetas y la implementación de medidas de seguridad.

III. Marco Teórico

A. *Servidor Web*

Los servidores web son programas informáticos que utilizan el protocolo HTTP para servir contenido estático y dinámico a los usuarios a través de la World Wide Web. Estos programas están diseñados para escuchar las solicitudes entrantes de los clientes (navegadores web u otros clientes HTTP) y responder con los recursos solicitados, como páginas web, imágenes, archivos, etc. Los servidores web también pueden manejar otros protocolos, como HTTPS (HTTP seguro), para proporcionar comunicaciones seguras a través de SSL/TLS.

Ejemplos:

- Apache HTTP Server: Es uno de los servidores web más populares y ampliamente utilizados en el mundo. Es de código abierto y es conocido por su estabilidad, flexibilidad y amplia comunidad de soporte.
- Nginx: Es un servidor web de alto rendimiento, ligero y escalable, conocido por su capacidad para manejar grandes volúmenes de tráfico de manera eficiente.
- Microsoft Internet Information Services (IIS): Es el servidor web desarrollado por Microsoft para el sistema operativo Windows, ampliamente utilizado en entornos corporativos y para aplicaciones que dependen de tecnologías de Microsoft, como ASP.NET.

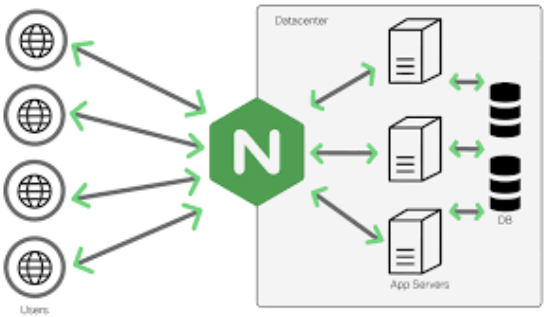


Fig.1. Funcionamiento nginx

B. Servidor de Aplicaciones

Los servidores de aplicaciones son entornos de software diseñados para ejecutar aplicaciones empresariales y proporcionar servicios a dichas aplicaciones, como la gestión de transacciones, la seguridad, la integración con bases de datos y sistemas externos, la escalabilidad y el balanceo de carga. Estos servidores proporcionan un marco para ejecutar aplicaciones web y empresariales en un entorno controlado y gestionado.

Ejemplos:

- Apache Tomcat: Es un servidor de aplicaciones de código abierto que implementa las especificaciones de Java Servlet, JavaServer Pages (JSP), y Java WebSocket, lo que lo hace adecuado para ejecutar aplicaciones web Java.
- JBoss Application Server (WildFly): Es un servidor de aplicaciones de código abierto y de alta performance desarrollado por Red Hat. Es conocido por su escalabilidad y por su soporte para tecnologías Java EE.
- IBM WebSphere: Es una familia de servidores de aplicaciones empresariales desarrollados por IBM, diseñados para implementar y gestionar aplicaciones empresariales complejas y de alta disponibilidad.

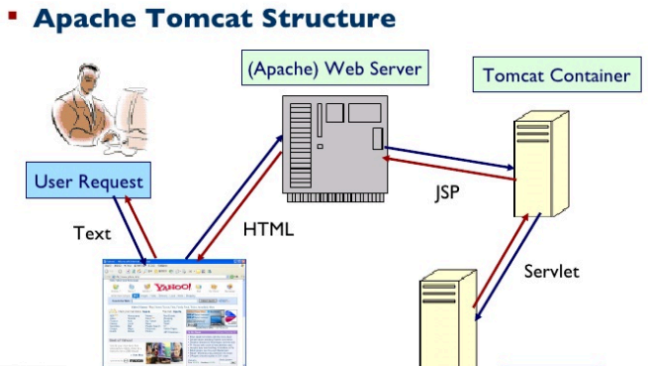


Fig.2. Estructura Apache Tomcat

C. Hosting

El hosting, también conocido como alojamiento web, se refiere al servicio que permite a individuos y organizaciones publicar sus sitios web y aplicaciones en Internet. Estos servicios son proporcionados por empresas de hosting, que mantienen servidores conectados a Internet y ofrecen capacidades de almacenamiento, ancho de banda, bases de datos, correo electrónico, entre otros, para que los usuarios puedan alojar sus sitios y aplicaciones web.

Ejemplos:

- Bluehost: Es una empresa de hosting que ofrece una amplia gama de servicios de hosting compartido, VPS (Servidores Privados Virtuales), dedicados, y servicios de alojamiento en la nube, además de herramientas de construcción de sitios web y soporte técnico.
- HostGator: Es otro proveedor de hosting muy conocido que ofrece servicios de hosting compartido, VPS, y dedicados, así como herramientas de creación de sitios web y aplicaciones.
- SiteGround: Es una empresa de hosting reconocida por su enfoque en la seguridad y el rendimiento, ofreciendo hosting compartido, hosting en la nube y servicios especializados para aplicaciones populares como WordPress y Joomla.

Fig.3. Logo SiteGround

D. Cloud Computing

Cloud computing es un modelo que permite el acceso a recursos informáticos compartidos a través de Internet bajo demanda. Estos recursos pueden incluir servidores, almacenamiento, bases de datos, redes, software, entre otros. En lugar de invertir en infraestructura física y mantenerla internamente, las organizaciones pueden utilizar recursos en la nube según sea necesario, pagando solo por lo que utilizan. Los servicios en la nube pueden ser proporcionados por proveedores de servicios en la nube, que gestionan y mantienen la infraestructura subyacente.

Ejemplos:

- Amazon Web Services (AWS): Es uno de los proveedores líderes en servicios de nube pública, ofreciendo una amplia gama de servicios, como almacenamiento, computación, bases de datos, análisis, inteligencia artificial, entre otros.
- Microsoft Azure: Es la plataforma de nube pública de Microsoft que ofrece servicios de infraestructura, plataforma y software como servicio (IaaS, PaaS, SaaS) para el desarrollo, implementación y gestión de aplicaciones y servicios.
- Google Cloud Platform (GCP): Es la plataforma de nube pública de Google, que proporciona una variedad de servicios en la nube, incluyendo computación, almacenamiento, bases de datos, aprendizaje automático, análisis, y más.

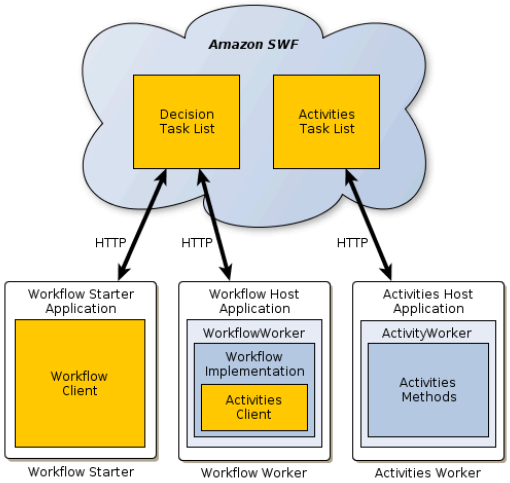


Fig.4. Estructura AWS

E. Despliegue continuo

El despliegue continuo es una práctica en el desarrollo de software que automatiza el proceso de implementación de cambios en una aplicación en un entorno de producción. Con el despliegue continuo, cada vez que se realiza un cambio en el código, ya sea una nueva función, una corrección de errores, o cualquier otro tipo de modificación, se ejecutan automáticamente las pruebas y se despliega la versión actualizada en el ambiente de producción.

Ejemplos:

- Jenkins: Es una herramienta de automatización de código abierto que se utiliza para automatizar el proceso de construcción, prueba y despliegue de aplicaciones de software.
- GitLab CI/CD: Es una característica integrada en GitLab que permite la automatización del proceso de integración continua y despliegue continuo directamente desde el repositorio de código.
- CircleCI: Es una plataforma de integración continua y entrega continua basada en la nube que permite a los equipos de desarrollo automatizar el proceso de construcción, prueba y despliegue de sus aplicaciones.

IV. PRÁCTICA

A. APACHE (WAMP O XAMPP)

Para esta práctica vamos a configurar el servidor APACHE XAMPP en la distribución de Windows.

La versión utilizada de APACHE XAMPP es la 8.2.12 y nos la podemos descargar desde el siguiente enlace:

https://www.apachefriends.org/es/download.html

La instalación y ejecución de nuestro servidor es sencilla:

- Descargamos nuestro servidor
- Ejecutamos el .exe descargado
- Seguimos todos los pasos que aparezcan
- Esperamos que se instale

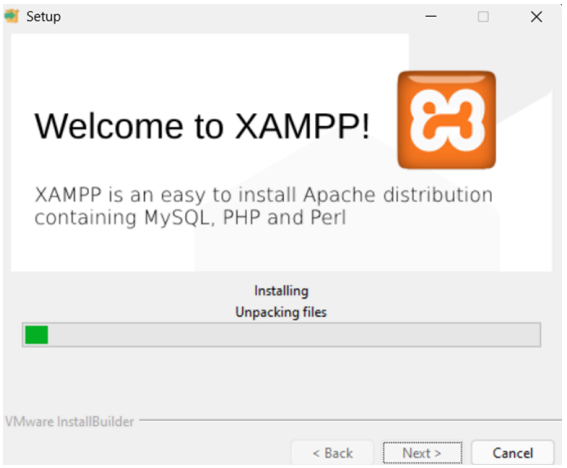


Fig.5. Instalando XAMPP

Una vez instalado correctamente buscamos XAMPP Control Panel y procedemos a abrir.

Al abrirlo podemos observar un panel que nos muestra diferentes ventanas y acciones que podemos realizar para nuestro servidor, lo podemos observar en la Fig. 6.

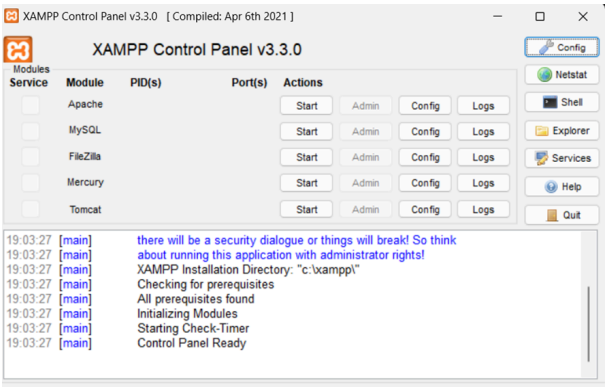


Fig.6. Panel XAMPP

Ahora, para poder cambiar nuestro puerto de localhost, lo podemos hacer de la siguiente manera:

- En el Módulo Apache, damos click en Config, en donde se nos despliegan opciones y escogemos Apache (httpd.conf)

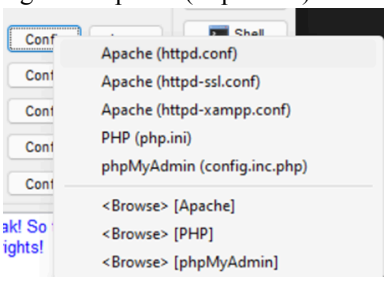


Fig.7. Opción Apache (httpd.conf)

- Al hacer click se nos abrirá un bloc de notas, en donde buscamos la palabra “listen” y nos

ubicamos en donde se coloca el número del puerto

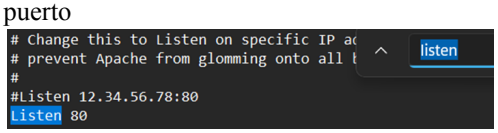


Fig.8. Línea para cambiar puerto

- Ahora solamente colocamos el puerto que deseamos utilizar, en este caso utilizaremos el puerto 8087 y guardamos los cambios.

Para ejecutar nuestro servidor APACHE, solamente debemos dar click en Start ubicado en Actions.

Fig.9. Ejecutando nuestro servidor

Fig.10. Servidor funcionando

Para lograr desplegar una página Web, debemos ubicarnos en la carpeta “htdocs” que se encuentra dentro de nuestra carpeta xampp, que es en donde instalamos nuestro servidor.

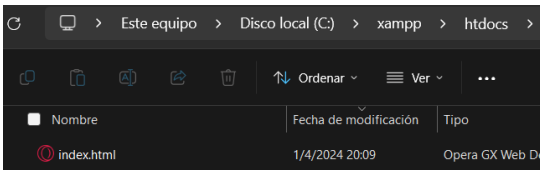


Fig.11. Creación archivo html

Una vez dentro de esta carpeta creamos un archivo .html en donde podemos desplegar nuestra página web.

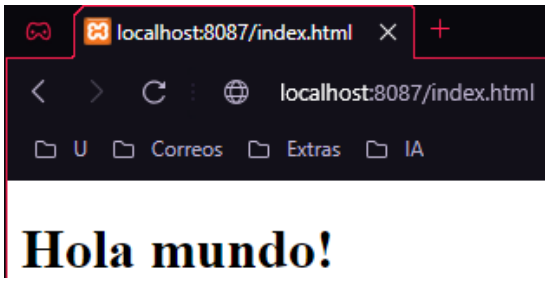


Fig.12. Página Web desplegada

Ahora vamos a crear una carpeta llamada ”CarpetaPública” para que sea la carpeta de publicación.

Para hacer que sea la carpeta de publicación debemos seguir los siguientes pasos:

- Damos click en Config y escogemos <Browse>[Apache]

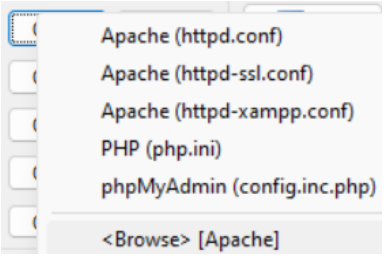


Fig.13. Opción <Browse>[Apache]

- Se nos abrirá un directorio e ingresamos a la carpeta conf.
- Ahora abrimos el archivo httpd.conf, ya sea con el bloc de notas o algún editor de texto.
- Buscamos las palabras “DocumentRoot” y nos ubicamos en esta parte:

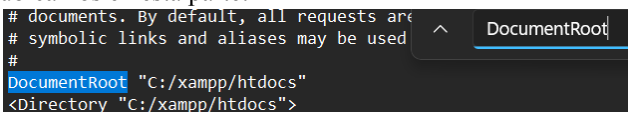
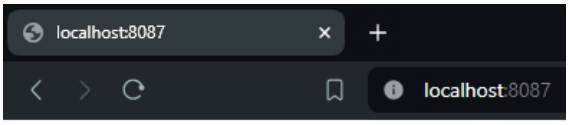


Fig.14. Línea para cambiar puerto

- Ahora, colocamos el directorio que queremos sea el nuevo directorio de publicación y guardamos cambios.

  DocumentRoot "C:/xampp/htdocs/CarpetaPública"
  <Directory "C:/xampp/htdocs/CarpetaPública">

- En el caso de que no hayamos detenido nuestro servidor, debemos detenerlo e iniciarlo de nuevo para notar los cambios.



Hola mundo!

Fig.15. Página Web desplegada

Como podemos observar, tenemos una nueva carpeta de publicación, ya que nuestro archivo index.html lo introdujimos dentro de la carpeta CarpetaPúblcia.

Para configurar el firewall y que sirva exclusivamente en el puerto 8087, seguimos los siguientes pasos:

- Buscamos Windows Defender Firewall con seguridad avanzada y entramos
- Nos dirigimos a reglas de salida
- Al costado derecho escogemos Nueva Regla
- En tipo de regla seleccionamos Puerto
- Ahora seleccionamos Puertos y Protocolos

- Colocamos el puerto 8087

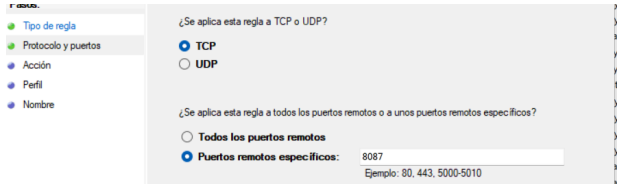


Fig.16. Configuración Firewall puerto 8087

- Luego Permitimos la conexión
- Damos siguiente y colocamos el nombre de nuestra regla y finalizamos.

B. APACHE TOMCAT

Para esta práctica vamos a configurar el servidor APACHE TOMCAT en la distribución de Windows.

La versión utilizada de APACHE XAMPP es la 9.0.87 y nos la podemos descargar desde el siguiente enlace:

https://tomcat.apache.org/download-90.cgi

En donde escogemos la siguiente opción:

- 64-bit Windows zip (pgp, sha512)

Una vez descargada, descomprimimos el archivo y abrimos el IDE de Netbeans. Dentro de Netbeans creamos un nuevo proyecto Java with Maven y dentro de este escogemos la opción Web Application.

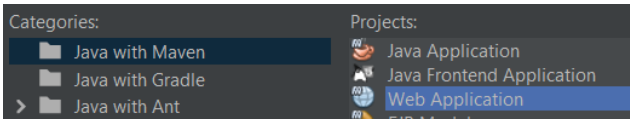


Fig.17. Creación proyecto web

- Damos click en Next
- Colocamos los datos de nombres y ubicación.
- Ahora en el apartados Settings, damos click en Add y seleccionamos Apache Tomcat y damos click en Next.

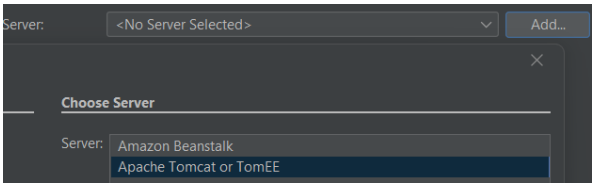


Fig.18. Escogemos apache tomcat

- Ahora necesitamos colocar la ubicación del servidor. Escogemos la ubicación de nuestro servidor, colocamos Usuario y contraseña y damos click en Finalizar y luego de nuevo click en Finalizar.

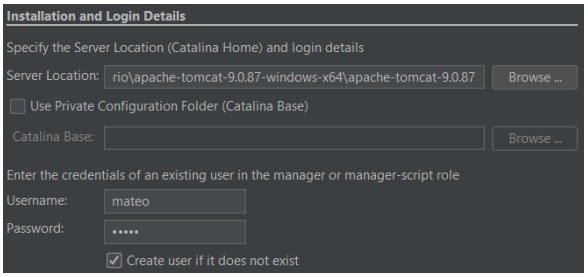


Fig.19. Localización y credenciales

Ahora podemos ejecutar nuestro servidor web, que actualmente se encuentra en el puerto 8080

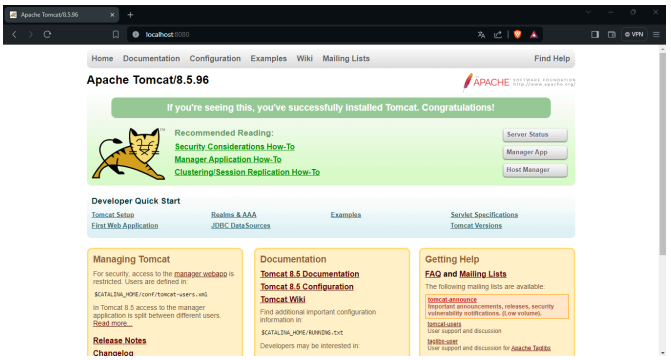


Fig.20. Servidor funcionando

Para cambiar el puerto nos dirigimos a la carpeta en donde hemos descomprimido nuestro archivo descargado, ingresamos a la carpeta conf y abrimos con un gestor de texto el archivo server.xml. En donde tenemos que buscar lo siguiente:

```
<Connector port="8084" protocol="HTTP/1.1"
```

Fig.21. Línea para cambiar puerto

En donde colocamos el puerto que deseamos, en este caso el puerto 8084.

Ahora, para lograr cambiar nuestra carpeta de publicación, en el mismo archivo server.xml debemos buscar lo siguiente:

```
<Host name="localhost"  appBase="webapps"
      unpackWARs="true" autoDeploy="true">
```

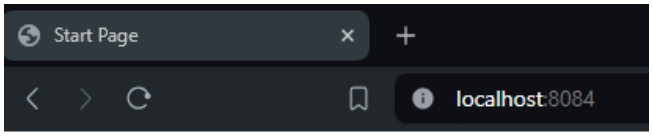
Fig.22. Línea para añadir la ruta

Debajo de esto debemos colocar lo siguiente:

<Context path="" docBase="ruta_.html"/>

En donde debemos especificar la ruta a nuestra carpeta de publicación, en este caso CarpetaPublica.

Ahora podemos ejecutar nuestro servidor y observamos que se ejecuta correctamente en la nueva carpeta..



Hola Mundo desde Tomcat!

Fig.23. Página web desplegada

Para configurar el firewall y que sirva exclusivamente en el puerto 8084, seguimos los siguientes pasos:

- Buscamos Windows Defender Firewall con seguridad avanzada y entramos
- Nos dirigimos a reglas de salida
- Al costado derecho escogemos Nueva Regla
- En tipo de regla seleccionamos Puerto
- Ahora seleccionamos Puertos y Protocolos
- Colocamos el puerto 8084

Fig.24. Configuración Firewall puerto 8084

- Luego Permitimos la conexión
- Damos siguiente y colocamos el nombre de nuestra regla y finalizamos.

C. *Servidor Cloud Computing*

Para configurar nuestro servidor web en Microsoft Azure debemos seguir los siguientes pasos:

- Ingresamos a nuestro portal azure y damos click en Crear nuevo recurso y escogemos Aplicación Web

Fig.25. Creación de nuevo recurso

- En la pestaña de Datos Básicos podemos ingresar lo siguiente:

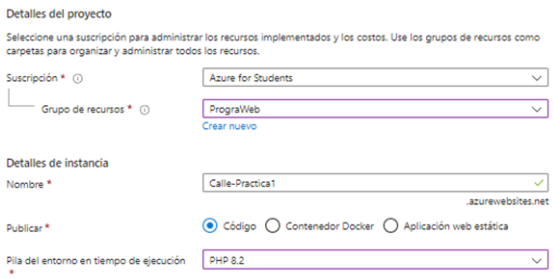


Fig.26. Ingreso de datos

- Omitimos la pestaña de base de datos ya que no es necesaria
- Habilitamos la implementación continua en la pestaña Implementación y configuramos los datos de nuestro repositorio Github
- En la pestaña de redes activamos el acceso público

Fig.27. Habilitar acceso público

- Omitimos la pestaña de Supervisión
- En este caso no es necesario la configuración de etiquetas por lo que también omitimos esa pestaña
- En la pestaña de revisar y crear podemos ver todos los detalles de nuestro servidor web

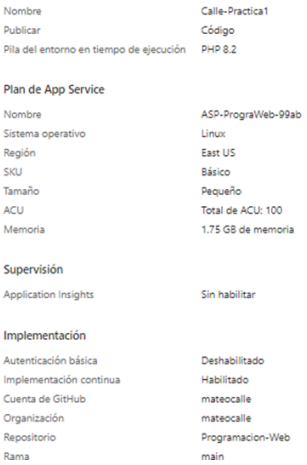
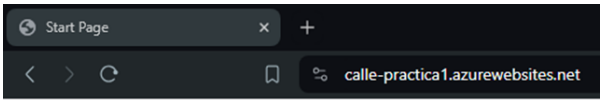


Fig.28. Detalles de nuestro servidor

- Por último, damos pulsamos el botón de crear
- Esperamos que se implemente correctamente todo el servidor y subimos a GitHub nuestro html.
- Una vez subido podemos observar nuestra página web desplegada



Hola Mundo!

Servidor Azure Mateo Calle

Fig.29. Página web desplegada

D. *Servidor Cloud Heroku*

Heroku es una plataforma en la nube que permite a los desarrolladores desplegar, administrar y escalar aplicaciones modernas. Ofrece una variedad de servicios y características, así como diferentes planes de precios para satisfacer las necesidades de cada aplicación y organización.

Costos

- **Eco y Basic:** Desde $5 al mes. Son dynos de bajo costo para probar ideas o ejecutar aplicaciones que ven uso intermitente.
- **Producción:** Aproximadamente $0.035/hora y más (máximo de $25/mes y más). Para

aplicaciones enfocadas en negocios, como aplicaciones web internas o de cara al cliente y APIs.

- **Avanzado:** Aproximadamente $0.347/hora y más (máximo de $250/mes y más). Para aplicaciones críticas para la misión con funcionalidad compleja que requiere alta disponibilidad, latencia muy baja y manejo de un alto volumen de solicitudes concurrentes.
- **Empresarial**: Se debe contactar a ventas para obtener precios personalizados. Para aplicaciones que cumplen con las necesidades de control, cumplimiento y colaboración de organizaciones a gran escala.

Restricciones de tecnología

- **Límites de la API:** Las llamadas a la API de Heroku están limitadas a un máximo de 4500 llamadas por hora.
- **Ancho de banda de la red:** El ancho de banda de la red está limitado a 2TB por aplicación por mes.
- **Memoria del Dyno:** Diferentes tamaños de dyno ofrecen diferentes cantidades de RAM máxima. Por ejemplo, los dynos eco, basic y standard-1x tienen 512 MB; standard-2x y private-s tienen 1 GB; performance-m, private-m, shield-m tienen 2.5 GB; performance-l, private-l, shield-l tienen 14 GB.
- **Restricciones de tiempo de espera HTTP:** Las solicitudes HTTP tienen una ventana inicial de 30 segundos en la que el proceso web debe devolver datos de respuesta.

Heroku ofrece una amplia gama de servicios y características que pueden adaptarse a las necesidades de diferentes tipos de aplicaciones y organizaciones. Sin embargo, es importante tener en cuenta las restricciones y limitaciones de la plataforma, como los límites de la API y el ancho de banda de la red. Además, aunque Heroku ofrece una variedad de planes de precios, los costos pueden aumentar rápidamente para aplicaciones más grandes o más complejas. Por lo tanto, es crucial considerar estos factores al elegir Heroku como plataforma de alojamiento.

Fig.30. Logo Servidor Cloud Herkou

E. Estructura de archivos

La estructura de archivos para la publicación de un sitio web puede variar dependiendo de diversos factores como el tamaño del proyecto, el tipo de tecnologías utilizadas, entre otros. Sin embargo, hay algunas convenciones comunes que se pueden seguir para organizar los archivos de manera clara y eficiente.

Por ejemplo, una manera común de estructurar los archivos para su publicación es de la siguiente manera:

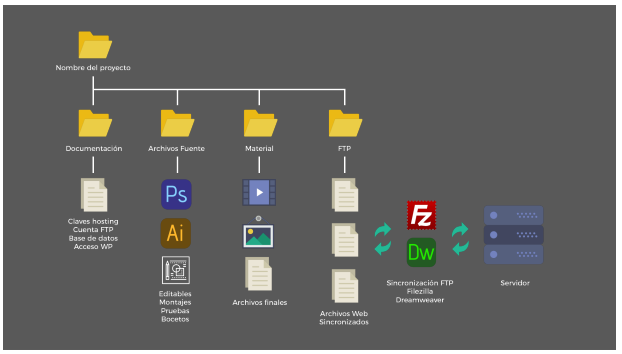


Fig.31. Estructura de archivos

En donde tenemos primero el nombre de nuestro proyecto que es la carpeta principal, luego tiene subcarpetas en donde tenemos:

- Documentación: Todos los archivos necesarios para documentar nuestro proyecto.
- Archivos Fuente: se refieren a los archivos originales utilizados para crear un sitio web.
- Material: Aquí se incluye todo lo que hemos utilizado para la creación de nuestra página web.
- FTP: Archivos web sincronizados

Como ya se mencionó anteriormente, la estructura de archivos puede cambiar de acuerdo al proyecto, por ejemplo, un tipo de estructura que se recomienda en Apache Tomcat es la siguiente:

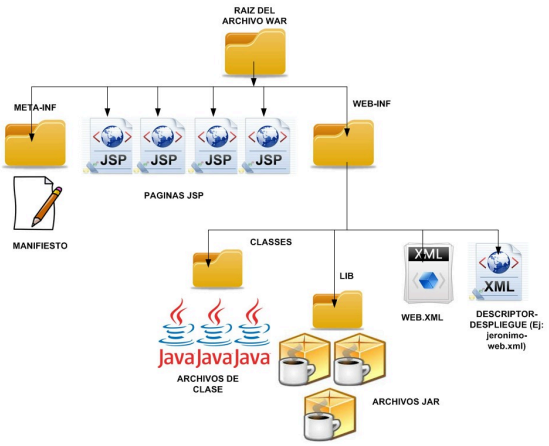


Fig.32. Estructura de archivos Apache Tomcat

En el proyecto realizado en Apache XAMPP incluimos una organización por carpetas simple solo para demostración, en donde se incluyó una carpeta para los estilos css, otra carpeta para imágenes y otra carpeta para subpáginas.

La estructura quedó de la siguiente manera:

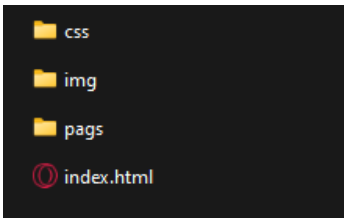


Fig.33. Estructura Proyecto XAMPP

Con esta organización de carpetas nuestra página desplegada se vería de la siguiente manera:

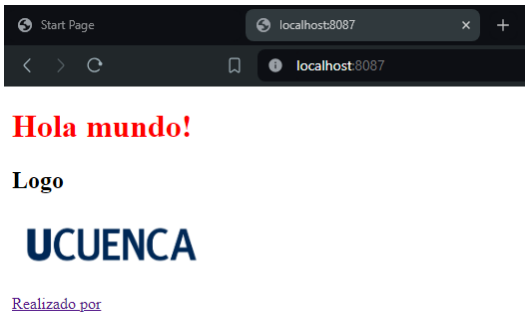


Fig.34. Página web con estructura de archivos

Al desplegar la subpágina:

Fig.35. Subpágina desplegada

V. Conclusiones

Se ha logrado exitosamente el objetivo general de la práctica al configurar adecuadamente los tres servidores web, implementar ajustes de configuración de puertos, firewall y estructura de carpetas. La configuración realizada garantiza el correcto funcionamiento y acceso a los servidores, lo que demuestra la adquisición de habilidades y competencias en la gestión de servidores web y de aplicaciones.

Esta práctica nos brinda una sólida base de conocimientos y experiencia práctica en el ámbito de la programación web y la gestión de servidores. A través de la configuración de diferentes tipos de servidores web y la exploración de los servicios en la nube, hemos adquirido una comprensión más profunda de los desafíos y oportunidades que presenta el mundo de la programación web. Esta experiencia nos prepara de manera efectiva para enfrentar los retos del desarrollo web en entornos dinámicos y en evolución, dotándonos de las habilidades necesarias para abordar proyectos futuros con confianza y eficacia.

VI. Referencias

[1] https://www.youtube.com/watch?v=q5VIxWXIL3A

[2] https://www.heroku.com/pricing

[3] https://rockcontent.com/es/blog/que-es-un-servidor/

[4]https://www.webempresa.com/hosting/que-es-servidor-web.html

[5]https://www.ionos.es/digitalguide/servidores/know-how/servidor-de-aplicaciones/

[6]https://www.ibm.com/docs/es/i/7.4?topic=serving-application-servers

[7] https://www.hostinger.es/tutoriales/que-es-un-hosting

[8] https://blog.hubspot.es/website/hosting

[9]https://cloud.google.com/learn/what-is-cloud-computing?hl=es

[10]https://www.salesforce.com/mx/cloud-computing/

[11]https://www.ibm.com/es-es/topics/continuous-deployment

[12]https://docs.github.com/es/actions/deployment/about-deployments/about-continuous-deployment