

Informe de Laboratorio 02

Tema: Git Docker

Nota

Estudiante	Escuela	Asignatura
Kevin Andree Llacma Quispe kllacma@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación web 2 Semestre: I Código: 20200585

Laboratorio	Tema	Duración
02	Git Docker	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	-	-

Programación Web

Laboratorio 02

Tema: Git y GitHub

13 de mayo de 2024

1. Especificaciones del Laboratorio

1.1. Objetivos del curso

- Proporcionar los conocimientos y habilidades para el uso de las principales metodologías de análisis y diseño de sistemas.
- Brindar los conocimientos para la utilización de técnicas para el análisis y diseño de sistemas web.
- Proporcionar conocimientos y habilidades en el manejo de herramientas para el desarrollo de sistemas Web.
- Desarrollar sistemas de información dentro de una arquitectura cliente servidor.

1.2. Objetivos del laboratorio

- Utilizar el sistema de control de versiones Git.
- Utilizar la plataforma GitHub para replicar y administrar proyectos de programación.

1.3. Equipos, materiales y temas

- Sistema Operativo (GNU/Linux de preferencia).
- Editor de texto plano (GNU Vim de preferencia).
- Navegador Web: Chrome, Firefox, Edge, Brave, Opera, etc.
- Git.
- Cuenta en GitHub asociada al correo institucional.

2. Marco teórico

2.1. PowerShell

PowerShell es una solución de automatización de tareas multiplataforma compuesta por un shell de línea de comandos, un lenguaje de secuencias de comandos y un marco de gestión de configuración. PowerShell se ejecuta en Windows, Linux y macOS.

2.2. La W3C y los Estándares Web

El World Wide Web Consortium o simplemente la W3C, desarrolla estándares y pautas para ayudar a todos a construir una web basada en los principios de accesibilidad, internacionalización, privacidad y seguridad.

Los estándares web son planos, o bloques de construcción, de un mundo conectado digitalmente consistente y armonioso. Se implementan en navegadores, blogs, motores de búsqueda y otro software que potencia nuestra experiencia en la web.

2.3. Vim

Vim es un editor de texto altamente configurable creado para hacer que la creación y el cambio de cualquier tipo de texto sean muy eficientes. Se incluye como "vi" en la mayoría de los sistemas UNIX y con Apple OS X.

Vim es muy estable y se desarrolla continuamente para mejorar aún más. Entre sus características se encuentran:

- árbol de deshacer persistente de varios niveles
- amplio sistema de complementos
- soporte para cientos de lenguajes de programación y formatos de archivo
- poderosa búsqueda y reemplazo
- se integra con muchas herramientas

```
$ sudo apt-get install vim # Vim en GNU/Linux  
$ brew install macvim # Vim en MacOSX
```

2.4. Visual Studio Code

Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como C++, C, Java, Python, PHP, Go, .NET).

Videotutoriales: <https://code.visualstudio.com/docs/getstarted/introvideos>

Descarga: <https://code.visualstudio.com/Download>

2.5. Git

Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia.

Documentación: <https://git-scm.com/doc>

Descargas: <https://git-scm.com/downloads>

2.6. GitHub

GitHub es una plataforma de hospedaje de código para el control de versiones y la colaboración. Este permite que tú y otras personas trabajen juntos en proyectos desde donde sea.

Documentación: <https://docs.github.com/es>

2.7. Arquitectura Git

Git tiene tres estados principales en los que se pueden encontrar tus archivos: confirmado (committed), modificado (modified), y preparado (staged).

- Confirmado: significa que los datos están almacenados de manera segura en tu base de datos local.
- Modificado: significa que has modificado el archivo pero todavía no lo has confirmado a tu base de datos.
- Preparado: significa que has marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.

Esto nos lleva a las tres secciones principales de un proyecto de Git: El directorio de Git (Git directory), el directorio de trabajo (working directory), y el área de preparación (staging area).

2.8. Estándares de codificación

Los estándares de código son una serie de reglas definidas para un lenguaje de programación, o bien, un estilo de programación específico.

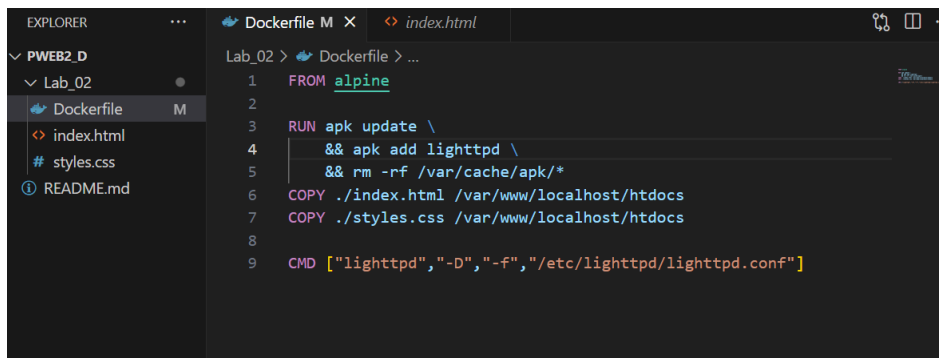
El estilo garantiza que todos los ingenieros que contribuyen a un proyecto tengan una forma única de diseñar su código, lo que da como resultado una base de código coherente, asegurando una fácil lectura y mantenimiento.

El uso de estándares es muy importante en la calidad de software, sin embargo mantener todos los proyectos cumpliendo a la perfección con esto no es una tarea fácil, requiere un gran esfuerzo y constancia por parte del equipo de desarrollo.

Mientras más y más compañías han adoptado estándares, todavía hay aquellas que realizan el desarrollo de sus proyectos sin ellos.

3. Desarrollo del lab

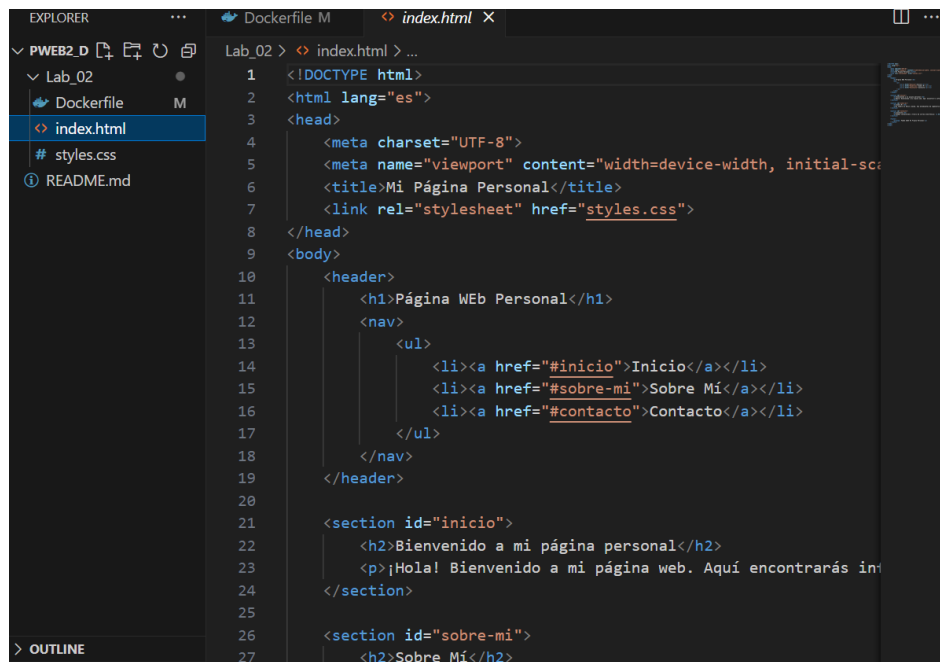
Primero se instala el docker a windows Para lograr correr una pagina web con docker contenedores: Se crea el archivo "Dockerfile" donde colocamos:



```
EXPLORER
...
PWEB2_D
└─ Lab_02
   └─ Dockerfile M
      index.html
      styles.css
      README.md

Lab_02 > Dockerfile > ...
1 FROM alpine
2
3 RUN apk update \
4   && apk add lighttpd \
5   && rm -rf /var/cache/apk/*
6 COPY ./index.html /var/www/localhost/htdocs
7 COPY ./styles.css /var/www/localhost/htdocs
8
9 CMD ["lighttpd", "-D", "-f", "/etc/lighttpd/lighttpd.conf"]
```

Seguido se uso un archivo index.html con sus estilos para que pueda usarlos

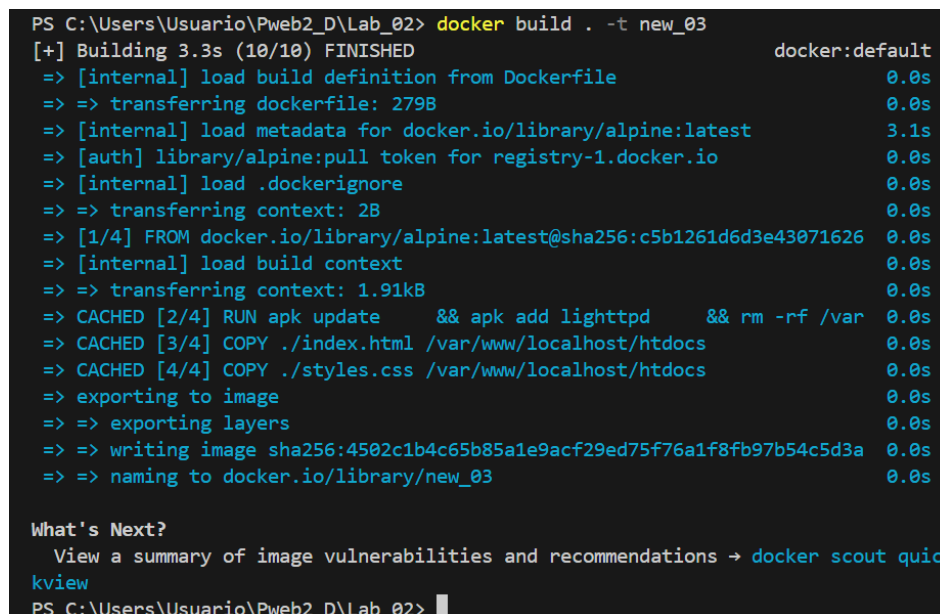


```

1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Mi Página Personal</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10  <header>
11    <h1>Página Web Personal</h1>
12    <nav>
13      <ul>
14        <li><a href="#inicio">Inicio</a></li>
15        <li><a href="#sobre-mi">Sobre Mi</a></li>
16        <li><a href="#contacto">Contacto</a></li>
17      </ul>
18    </nav>
19  </header>
20
21  <section id="inicio">
22    <h2>Bienvenido a mi página personal</h2>
23    <p>¡Hola! Bienvenido a mi página web. Aquí encontrarás información sobre mi proyecto.</p>
24  </section>
25
26  <section id="sobre-mi">
27    <h2>Sobre Mi</h2>

```

Ahora usamos docker build para crear una imagen Docker esto usando el archivo Dockerfile



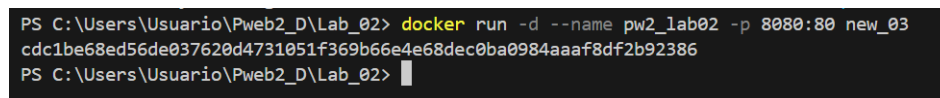
```

PS C:\Users\Usuario\Pweb2_D\Lab_02> docker build . -t new_03
[+] Building 3.3s (10/10) FINISHED          docker:default
=> [internal] load build definition from Dockerfile      0.0s
=> => transferring dockerfile: 279B                    0.0s
=> [internal] load metadata for docker.io/library/alpine:latest 3.1s
=> [auth] library/alpine:pull token for registry-1.docker.io 0.0s
=> [internal] load .dockerignore                        0.0s
=> => transferring context: 2B                          0.0s
=> [1/4] FROM docker.io/library/alpine:latest@sha256:c5b1261d6d3e43071626 0.0s
=> [internal] load build context                        0.0s
=> => transferring context: 1.91kB                      0.0s
=> CACHED [2/4] RUN apk update && apk add lighttpd && rm -rf /var 0.0s
=> CACHED [3/4] COPY ./index.html /var/www/localhost/htdocs 0.0s
=> CACHED [4/4] COPY ./styles.css /var/www/localhost/htdocs 0.0s
=> exporting to image                                  0.0s
=> => exporting layers                                  0.0s
=> => writing image sha256:4502c1b4c65b85a1e9acf29ed75f76a1f8fb97b54c5d3a 0.0s
=> => naming to docker.io/library/new_03                0.0s

What's Next?
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS C:\Users\Usuario\Pweb2_D\Lab_02>

```

Se uso el comando Docker run para crear y ejecutar un contenedor Docker a partir de una imagen Docker existente.



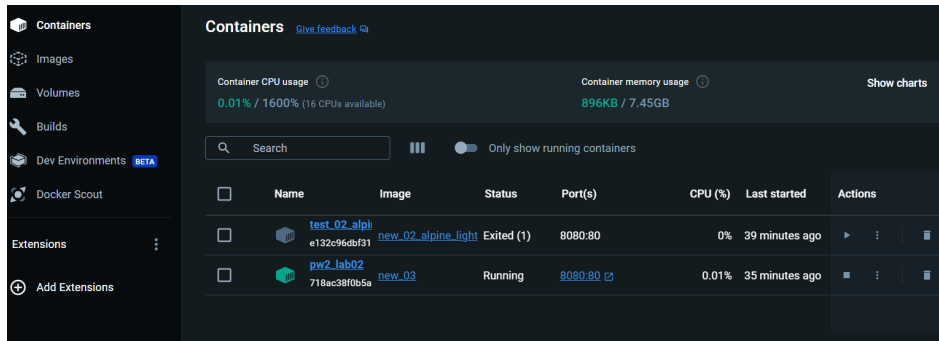
```

PS C:\Users\Usuario\Pweb2_D\Lab_02> docker run -d --name pw2_lab02 -p 8080:80 new_03
cdc1be68ed56de037620d4731051f369b66e4e68dec0ba0984aaaf8df2b92386
PS C:\Users\Usuario\Pweb2_D\Lab_02>

```

Al ejecutar el comando podemos ver que en docker desktop aparece el contenedor que esta ejecun-

tandose



Ahora podemos ver en el puerto 8080 que esta corriendo la pagina a travez de docker

