Ejercicio 1 - Linux y Automatización

### **1. Introducción**

El objetivo de este ejercicio es automatizar el deploy, a través de un script en bash, de una aplicación web (DevOps Travel) junto a su base de datos, su servidor y demás herramientas necesarias para que la aplicación esté en línea.

La misma será construida a partir de una arquitectura LAMP, es decir, utilizando Linux, Apache como servidor, MySQL como base de datos y PHP para procesar nuestros datos dinámicos.

De esta forma, el deploy será automático, evitando errores, rápido y más seguro.

### **2. Requisitos Previos**

Tener conocimientos básicos en Linux, habiendo instalado previamente una distribución de dicho Sistema Operativo para llevar a cabo nuestro trabajo.

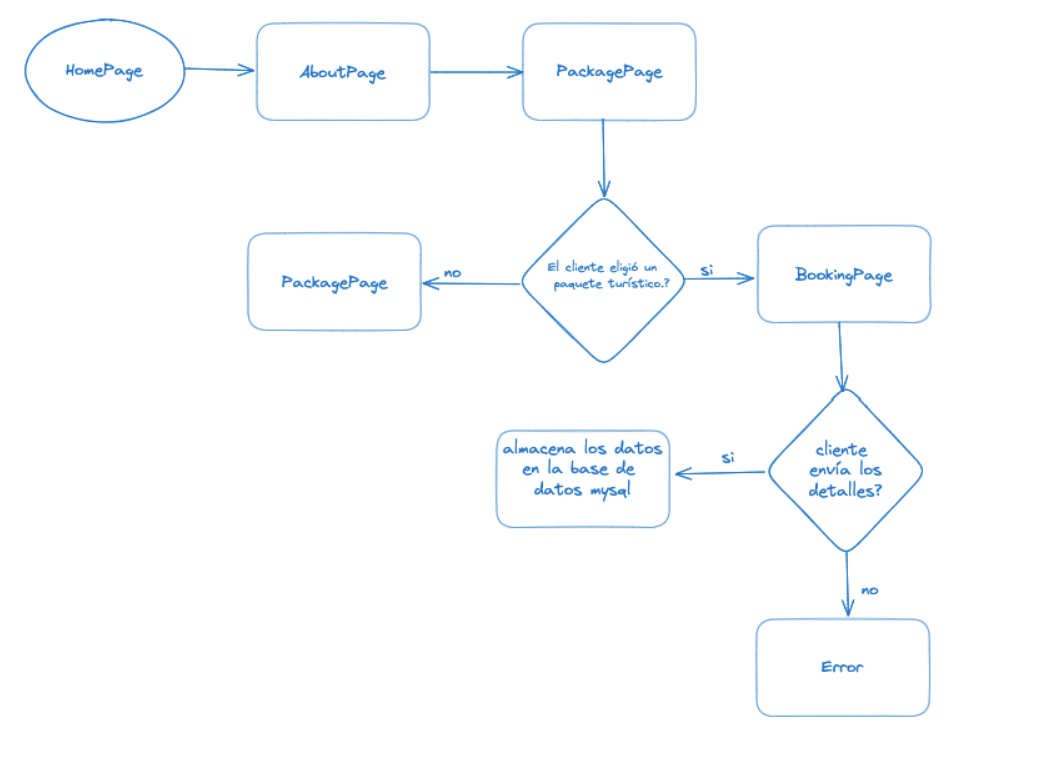
Para ello se puede contar con una instalación nativa o es recomendable instalar alguna máquina virtual como puede ser VirtualBox, VMware, etc. A su vez contar con los requisitos mínimos para poder instalar la distribución preferida. Para este caso particular, se utilizó el WSL (Windows Subsystem Linux, con una distribución Ubuntu.

### **3. Arquitectura del Proyecto**

En el diagrama de arquitectura, los usuarios inician una solicitud HTTP accediendo a la aplicación a través del navegador utilizando "localhost" o la dirección IP del servidor. El servidor, con Apache instalado, responde entregando el archivo a los usuarios, solicitándoles que completen sus datos, incluido su nombre, correo electrónico y descripción.

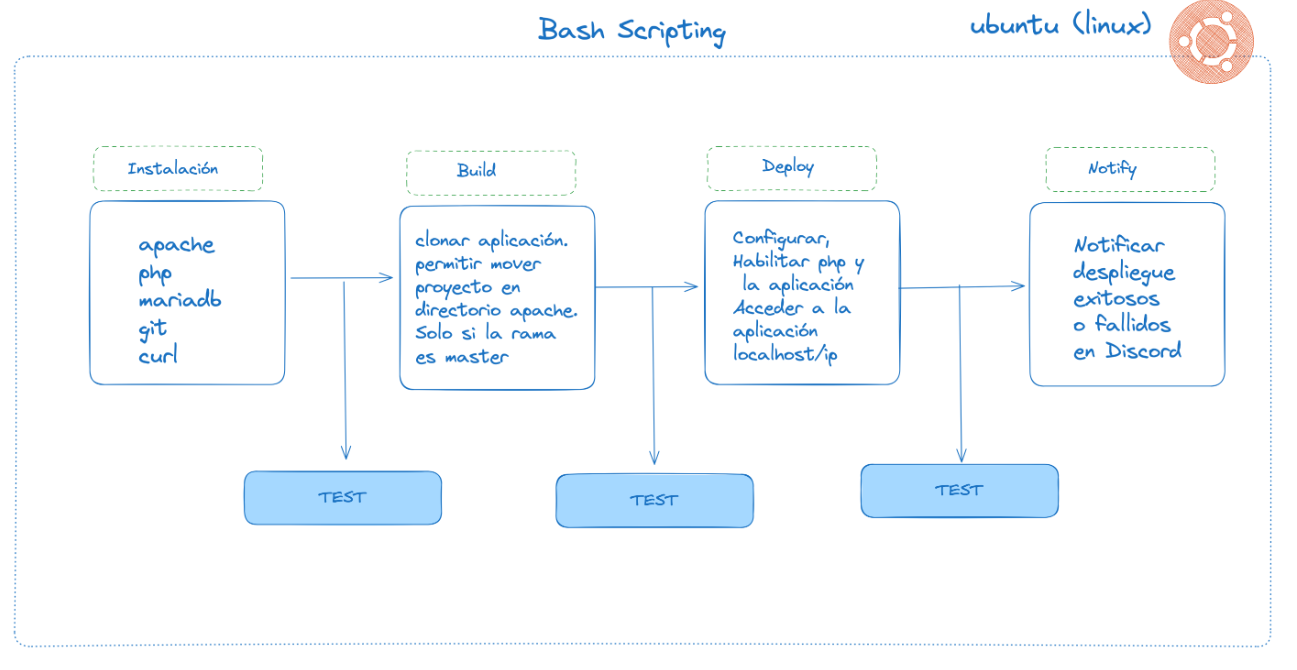
Al completar el formulario, los usuarios envían los datos al servidor. Luego, Apache reenvía los datos enviados a un script PHP responsable de almacenar esta información en la base de datos MySQL. Si los datos se almacenan correctamente, MySQL comunica este éxito al script PHP, que responde con un mensaje HTML que se muestra en el navegador del usuario. Por otro lado, si hay un problema al guardar los datos, el script PHP devuelve un mensaje de error al navegador del usuario, notificándole el problema encontrado.

#### **Diagrama de la aplicación**



#### 

#### **Diagrama del deploy automático**



### **5. Scripts y Código**

El código lo encontrarán en el siguiente repositorio:

<https://github.com/tjiam1/Bootcamp_Roxsross_2023/blob/master/Clase2_Linux_Devops/deploy.sh>

El mismo cuenta de 5 etapas.

En primera instancia se declararán las variables necesarias para que el script funcione, cómo así también de otras que hacen al código mucho más claro. En esta etapa, a su vez, se comprueba que el usuario ejecutor sea root.

Luego, el script se dividirá en 4 Stages:

#### **STAGE 1 : Init**

En esta etapa se buscará:

* Actualizar el servidor,
* Se instalará git, software para control de versiones
* Se instalará Maria DB, como sistema de gestión de base de datos, derivado de MySQL. Además, se validará que el mismo se encuentra habilitado.
* Se instalará Apache, como servidor web. A su vez, se validará que el mismo esté habilitado y corriendo
* Por último, se instalará y se bajarán los paquetes de PHP como lenguaje para poder procesar nuestros datos.

#### **STAGE 2 : Build**

En esta etapa comenzamos a construir nuestra web.

Para ello se revisará primero:

* la existencia de la carpeta correspondiente al repositorio que bajaremos desde github.

Si el repo ya existiese:

* Se hará un git pull para descargar los nuevos cambios, y luego un **git checkout clase2-linux-bash** para posicionarnos en el repo y la rama donde se aloja nuestra web.

Si la carpeta no existe,

* se hará un **git clone** para clonar el repositorio en cuestión. Este repositorio tendrá todo lo necesario para deployar nuestra web.

Luego se procederá a:

* mover al directorio donde se guardan los archivos de configuración de apache /var/www/html/ los archivos recientemente clonados
* Ajustar la config. de php, agregando el index.php.
* Configuramos base de datos MariaDB.

#### **STAGE 3 : Deploy**

* Configuramos apache2
* Editamos el config.php para que le pasemos por parámetro la password de la base de datos.
* Lanzamos un reload del apache para confirmar los cambios.
* Por último, ejecutamos un curl para que checkear el status ok de nuestra web.

#### **STAGE 4 : Notify**

A partir de un webhook, avisamos a un canal de discord, cuando nuestra web se encuentra en línea o no, una vez finalizado los pasos anteriores.

### **5. Conclusión**

El script cumple con el objetivo de deployar una web desarrollada con la arquitectura LAMP

Sin embargo podríamos modularizar aún más los pasos del script a través de funciones, para que el código sea aún más claro, y podamos tener códigos de respuesta por ejemplo cuando alguno de los stage finaliza con algún error, terminando con la ejecución del programa.

Además, se podría agregar un archivo de log para que justamente nos deje un registro de cada paso y saber cómo se ejecutó más allá del estado final y así poder hacer un debug pensando en proyectos mucho más grandes que requieran muchos más controles durante su ejecución.