

Trabajo Final de Especialización en Tecnologías de la Información

**“Guía y repositorio para el despliegue de aplicaciones Web en servidores Linux utilizando contenedores”**

Autor: Jesus Andres Zini

Director: Emanuel Irrazábal

Año 2023

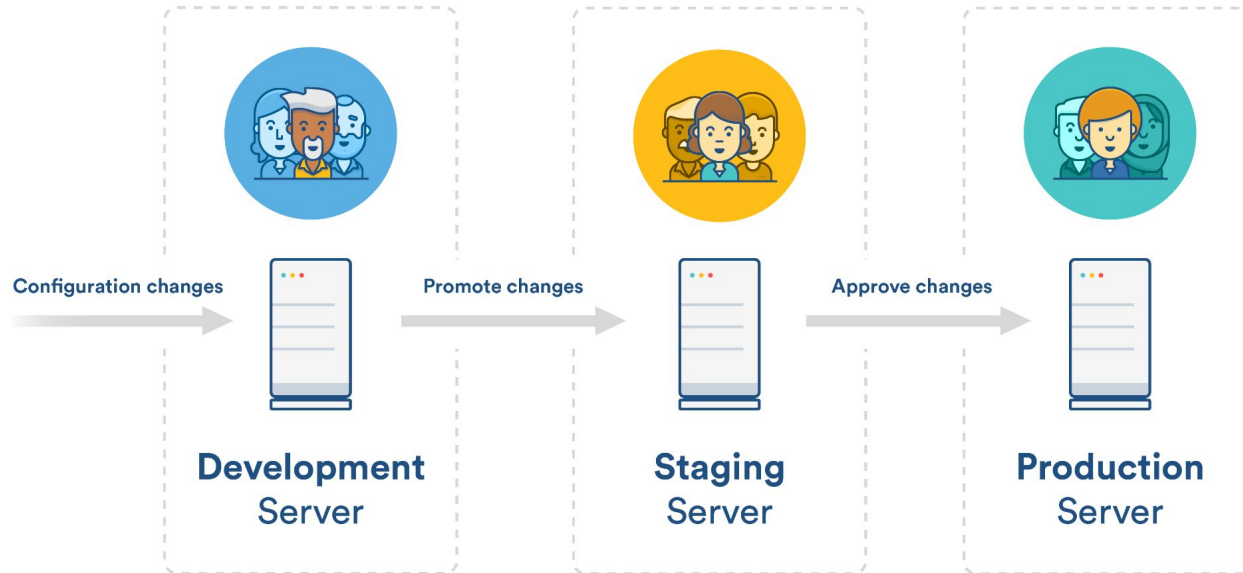
# **1 - El problema a solucionar**

2 - Propuesta de solución

3 - Resultados

# Introducción

Uno de los procesos clave en equipos de desarrollo de software es el despliegue a Internet de las aplicaciones Web.



# Introducción

En empresas pequeñas o compuestas por pequeños equipos de trabajo, no siempre es económicamente viable contratar personal con habilidades específicas para una sola tarea o rol.



# Introducción

En estos casos, lo más común es contar con equipos multidisciplinarios, con integrantes que puedan abordar distintas tareas de manera efectiva.



# Introducción

Un problema frecuente en estos equipos radica en que los desarrolladores, si bien cumplen con su función de construcción del código fuente, cuando llega la hora de desplegar la aplicación, carecen de los conocimientos necesarios para completar esta tarea.



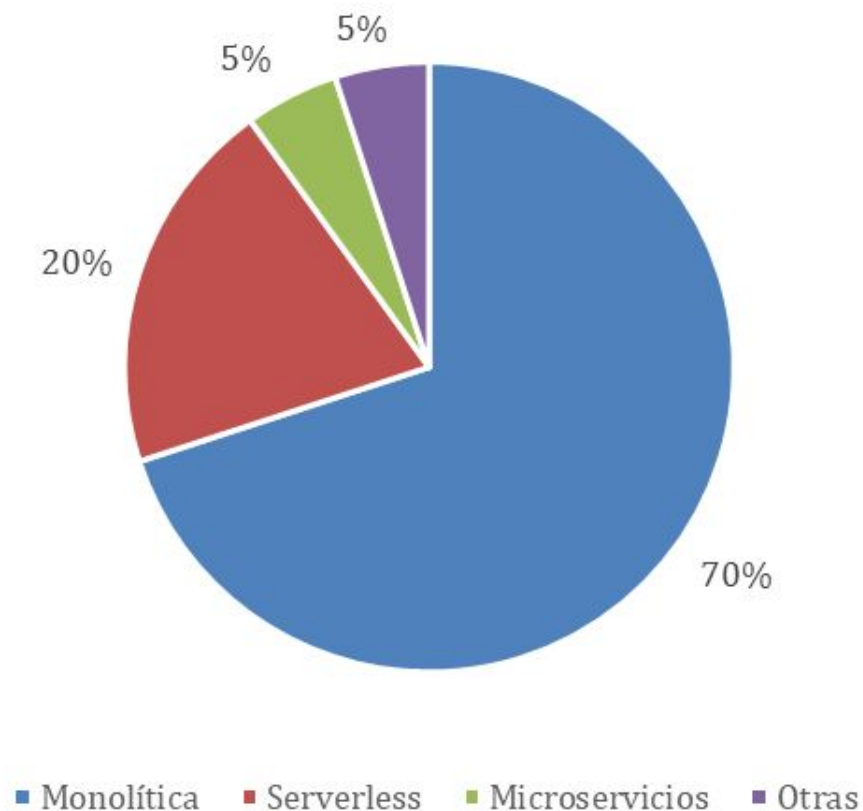
## Interrogantes comunes en el proceso de despliegue.

- ¿Dónde y cómo desplegar el proyecto?
- ¿Cómo configurar el servidor con las dependencias que el proyecto necesita?
- ¿Cómo agregar HTTPS/SSL a la aplicación?
- ¿Cómo desplegar una nueva versión de la aplicación?

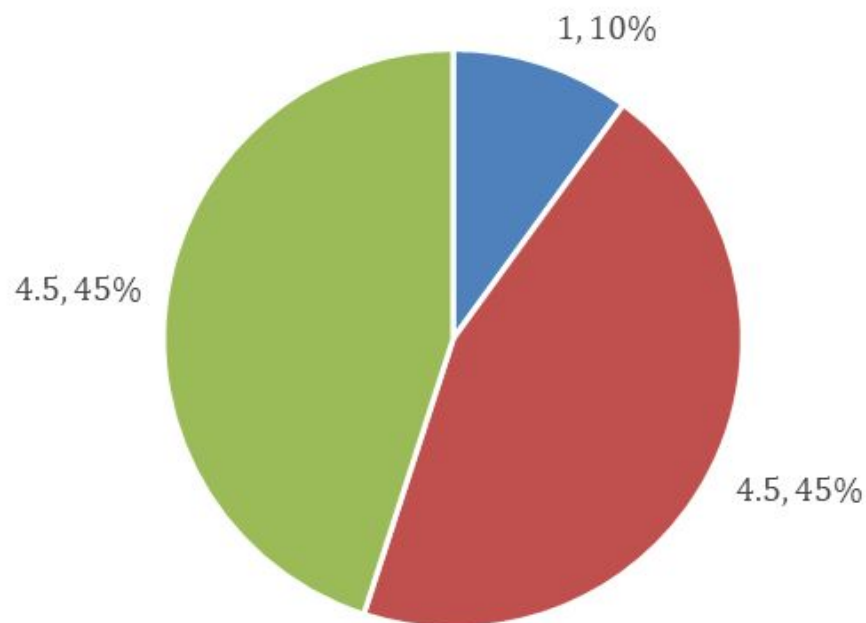
Se realizaron encuestas a equipos de desarrollo de software de la región con el objetivo de comprender el entorno en el que operan.



## Arquitecturas de proyecto más utilizadas.



## Plataformas utilizadas para despliegue de aplicaciones.



■ Servidores dedicados

■ Servidores privados virtuales

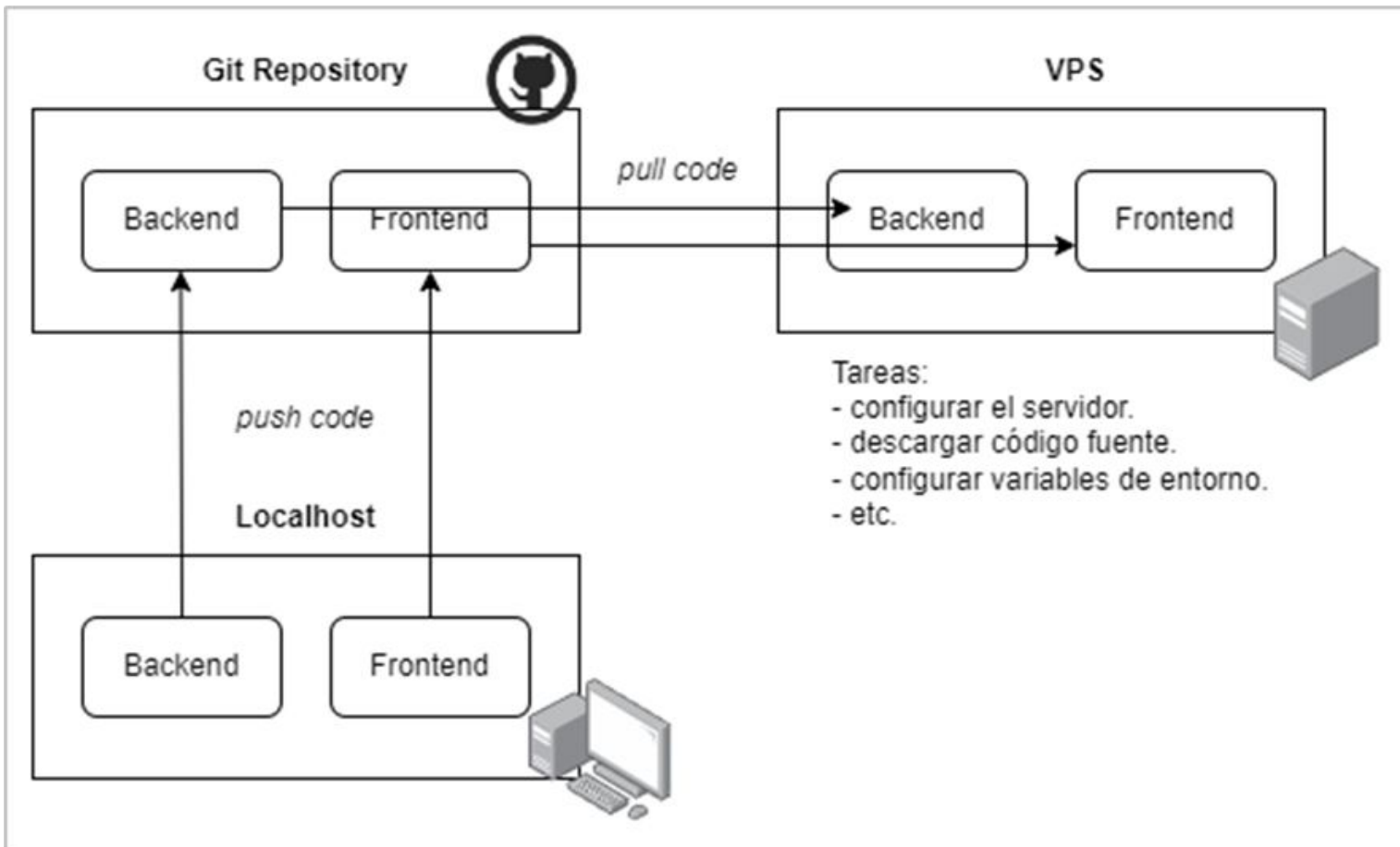
■ Computación en la Nube



Si bien tenemos muchas opciones a la hora de desplegar software, nos vamos a centrar en despliegues de aplicaciones Web monolíticas en servidores privados virtuales (VPS o VM).



## Despliegue de aplicaciones Web en una VM realizado de forma tradicional

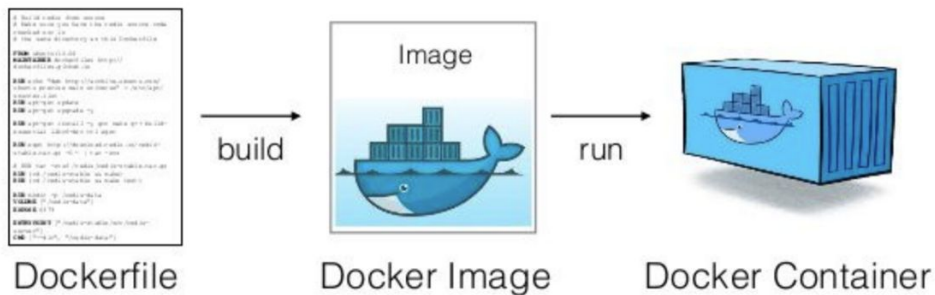


# Inconvenientes comunes en el proceso de despliegue

- Instalar los programas y las dependencias necesarias para la ejecución del proyecto en el servidor.
- Subir el código fuente de la aplicación en su totalidad al servidor.
- Realizar las configuraciones del proyecto en el servidor (variables de entorno, credenciales, etc)
- Migrar el proyector a otro servidor implica realizar todo el proceso nuevamente.
- Desplegar más de una aplicación en el mismo servidor puede generar incompatibilidades.

# Despliegue de aplicaciones Web utilizando contenedores.

La utilización de contenedores puede ser útil para resolver algunos de los problemas anteriormente mencionados, porque se empaqueta el software en unidades estandarizadas para su fácil desarrollo, transporte y despliegue.



## Despliegue de aplicaciones Web utilizando contenedores.

Aunque el uso de contenedores es una mejora significativa en términos de aprovechamiento de recursos y automatización de procesos, su implementación aún puede generar desafíos.

- No existe forma estandarizada de realizar el despliegue en el servidor utilizando contenedores.
- Esto a menudo deriva a la implementación de malas prácticas.

```
jesus_zini@dev-tech-image1-1:~$ sudo docker ps --format "table {{.ID}}\t{{.Names}}\t{{.CreatedAt}}\t{{.Status}}\t{{.Ports}}"
```

CONTAINER ID	NAMES	CREATED AT	STATUS	PORTS
71461f54919c	liber_nginx	2023-07-07 13:39:48 +0000 UTC	Up 9 days	0.0.0.0:8081->80/tcp, :::8081->80/tcp
6e200add1e4c	liber_app	2023-07-07 13:39:48 +0000 UTC	Up 9 days	80/tcp, 3000/tcp
08b783461c9e	liber_api	2023-07-07 13:39:47 +0000 UTC	Up 9 days	4000/tcp
7dee58b42fce	liber_db	2023-07-07 13:39:44 +0000 UTC	Up 9 days	5555/tcp, 0.0.0.0:5555->5432/tcp, :::5555->5432/tcp
d2940ea8fc07	scrap_dtiol_container	2022-11-04 14:57:26 +0000 UTC	Up 2 weeks	0.0.0.0:7005->3000/tcp, :::7005->3000/tcp
7cbcebef18ea	app-noti-front	2022-11-04 14:35:19 +0000 UTC	Up 2 weeks	80/tcp, 3000/tcp
b07716fb446f	scrap_federal_container	2022-11-03 20:48:42 +0000 UTC	Up 2 weeks	0.0.0.0:7002->3002/tcp, :::7002->3002/tcp
883baf004773	nginx-noti	2022-11-03 20:29:48 +0000 UTC	Up 2 weeks	0.0.0.0:6005->80/tcp, :::6005->80/tcp
4c2f7f7857a3	noti_notificaciones	2022-11-03 20:29:45 +0000 UTC	Up 2 weeks	0.0.0.0:7001->7001/tcp, :::7001->7001/tcp
0238bf7c436b	noti-exp	2022-11-03 20:29:45 +0000 UTC	Up 2 weeks	0.0.0.0:6080->6080/tcp, :::6080->6080/tcp
9ac05831b5a2	noti_caducidad	2022-11-03 20:29:39 +0000 UTC	Up 2 weeks	0.0.0.0:6090->6090/tcp, :::6090->6090/tcp
3cf20d5c1880	noti_pagos	2022-11-03 20:29:39 +0000 UTC	Up 2 weeks	0.0.0.0:7000->7000/tcp, :::7000->7000/tcp
fe28684ab0f4	postgres-noti	2022-11-03 20:29:39 +0000 UTC	Up 2 weeks	0.0.0.0:5432->5432/tcp, :::5432->5432/tcp
65e901356ca8	aci-public-page	2022-10-14 12:46:03 +0000 UTC	Up 2 weeks	80/tcp, 0.0.0.0:3000->3000/tcp, :::3000->3000/tcp
88474bcb0b37	aci-front-admin	2022-10-14 12:46:03 +0000 UTC	Up 2 weeks	80/tcp, 0.0.0.0:3008->3001/tcp, :::3008->3001/tcp
133679f8763c	aci-back	2022-10-14 12:46:01 +0000 UTC	Up 2 weeks	0.0.0.0:4000->4000/tcp, :::4000->4000/tcp, 0.0.0.0:8010->8000/tcp, :::8010->8000/tcp

# Despliegue de aplicaciones Web utilizando contenedores.

Malas prácticas en el proceso de despliegue con contenedores:

- Clonar el código fuente del proyecto en el VPS.
- No incluir toda la aplicación en el set de contenedores, (servidor/proxy, base de datos).
- Realizar la construcción (build) de la imagen del contenedor dentro del mismo servidor.
  - sobrecarga la memoria.
  - sobrecarga en almacenamiento.



# Conclusiones hasta aquí

El despliegue de aplicaciones Web en equipos de desarrollo de software pequeños presenta desafíos significativos.

Aunque el uso de contenedores Docker puede ser beneficioso, también introduce nuevas complejidades.

1 - El problema a solucionar

## **2 - Propuesta de solución**

3 - Resultados

# Propuesta de solución

Por lo recién mencionado, lo que se plantea en este trabajo es definir una forma eficiente para realizar el despliegue de aplicaciones Web en servidores VPS.

# Objetivo general

Desarrollar una propuesta de trabajo de despliegue para aplicaciones web que van a ser desplegadas en servidores Linux, mediante la utilización de tecnologías actuales de control de versiones y contenedores con el fin tanto de optimizar y simplificar el proceso de despliegue y la actualización de software en dichos entornos mejorando el rendimiento y disminuyendo la cantidad de errores.

# Objetivos específicos

- Analizar las buenas prácticas de la industria del software que busca resolver las dificultades del despliegue de aplicaciones web y seleccionar las actividades que pueden ser implementadas en el marco de equipos pequeños de desarrollo de software.
- Construir de forma iterativa e incremental un procedimiento para desplegar aplicaciones web con arquitectura monolítica en servidores privados virtuales. Operacionalizar a partir de un repositorio compartido con archivos de configuración base del contenedor y scripts de despliegue.
- Establecer un enfoque de despliegue que se base en la utilización de un repositorio centralizado para almacenar todos los archivos relacionados con el proceso de despliegue de la aplicación, incluyendo su documentación.
- Verificar y mejorar el procedimiento a partir de su uso en un equipo de trabajo de una empresa de la industria del software de la región.


# Propuesta. Metodología adoptada.

En proyectos y equipos de gran tamaño, es común contar con un **repositorio específico** que **almacena todas las configuraciones necesarias para el despliegue de la aplicación**.

```
└─project
  └─project-backend
  └─project-frontend
```


```
└─project
  └─project-backend
  └─project-deployment
  └─project-frontend
```

<https://github.com/jesusandres31/skeleton-deployment>


 jesusandres31 / skeleton-deployment

Q Type to search



[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 **skeleton-deployment** Public Pin Unwatch 1


master 1 branch 0 tags Go to file Add file <> Code

 **jesusandres31** update script name 0c56d96 on Aug 21 10 commits

config	update docs	6 months ago
environments	update configs	5 months ago
proxy	update configs	5 months ago
scripts	update script name	3 months ago
utils	update configs	5 months ago
.gitignore	update configs	5 months ago
README.md	update configs	5 months ago

 **README.md** 

# Skeleton-Deployment



This repository contains the source code for the skeleton-deployment project. It is a simple project that demonstrates how to deploy a web application to a cloud provider using a CI/CD pipeline.

# Repositorio de despliegue en una escala reducida

El "repositorio esqueleto" funciona como de plantilla y punto de partida.

Incluye un conjunto mínimo de documentación, archivos, directorios y configuraciones necesarios para el despliegue:

- Dockerfile
- docker-compose.yml con configuraciones genéricas que pueden ser personalizadas mediante un archivo ".env".
- archivos de configuración para el servidor Web y proxy reverso Nginx
- contenedor encargado de agregar y renovar el certificado SSL/HTTPS a través de Certbot (Let's Encrypt).
- scripts de Bash que permiten a los desarrolladores automatizar tareas repetitivas en lugar de ejecutar comandos manualmente.



# Repositorio de despliegue en una escala reducida

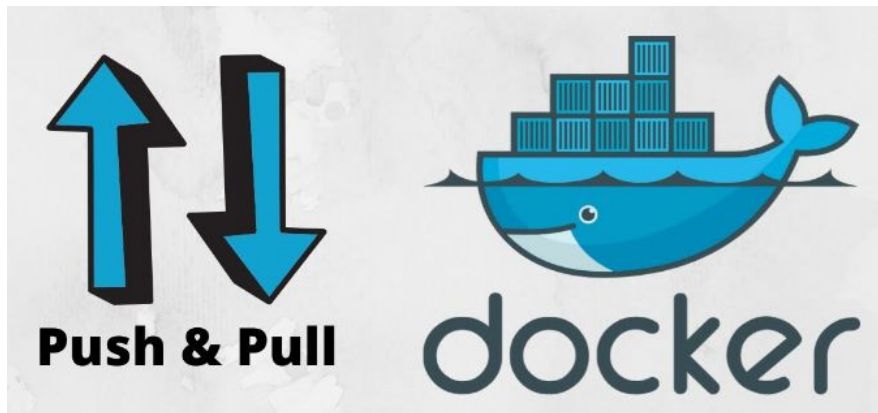
Los scripts de bash ejecutan tareas, tales como:

- Construcción de las imágenes de Docker de la aplicación.
- Subir las imágenes al registro de contenedores (o Container Registry).
- Ejecutar los contenedores con la versión más reciente de la aplicación.

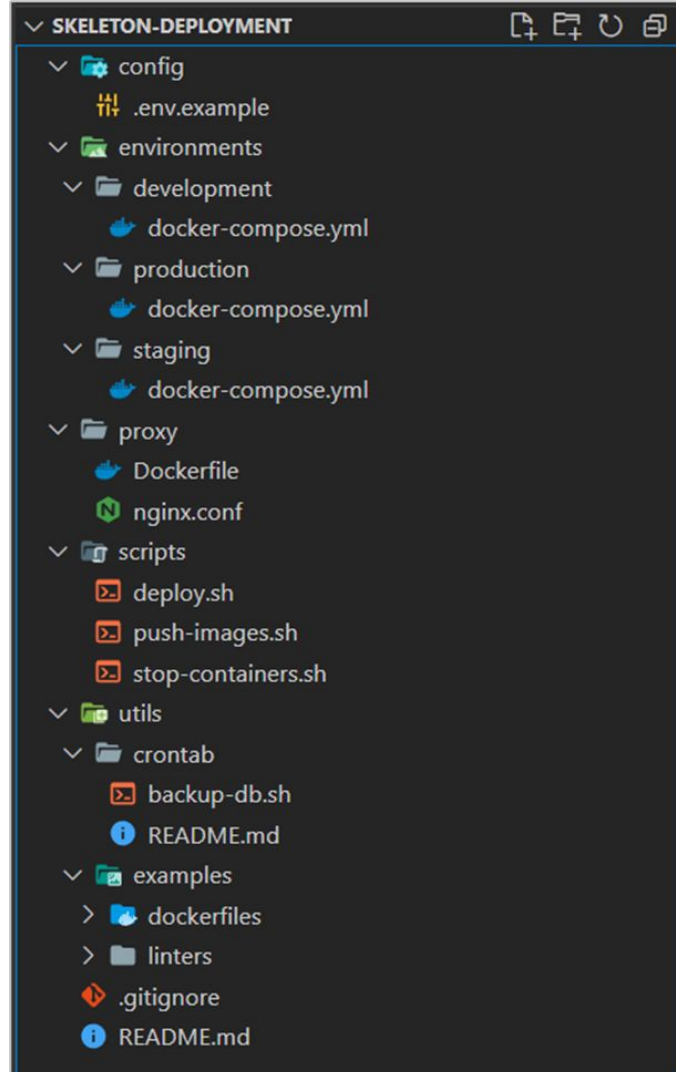


Container Registry para el almacenamiento de las imágenes de Docker.

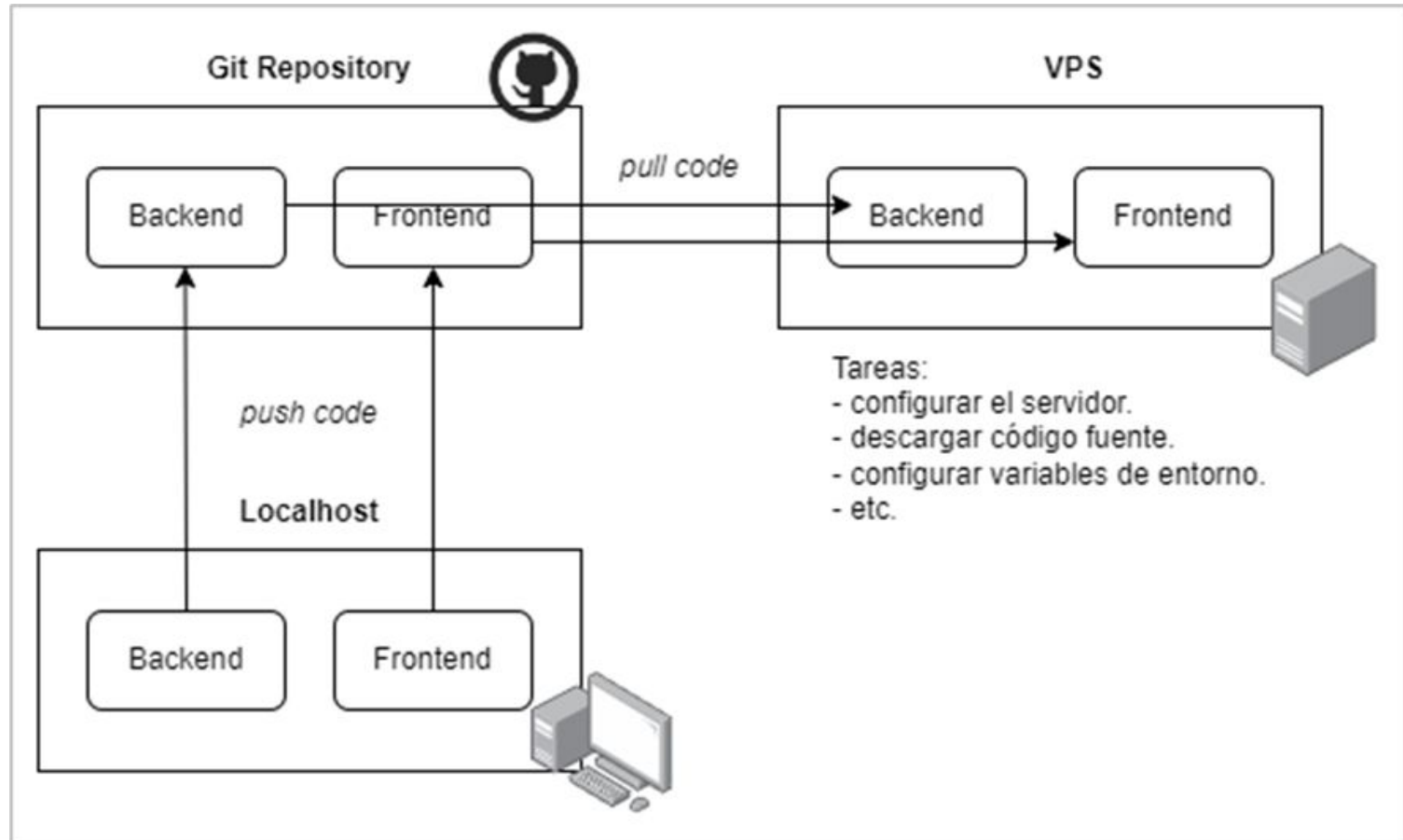
Docker Hub, Gitlab Container Registry, Amazon Elastic Container Registry (ECR), Google Container Registry (GCR), Microsoft Azure Container Registry (ACR), etc.

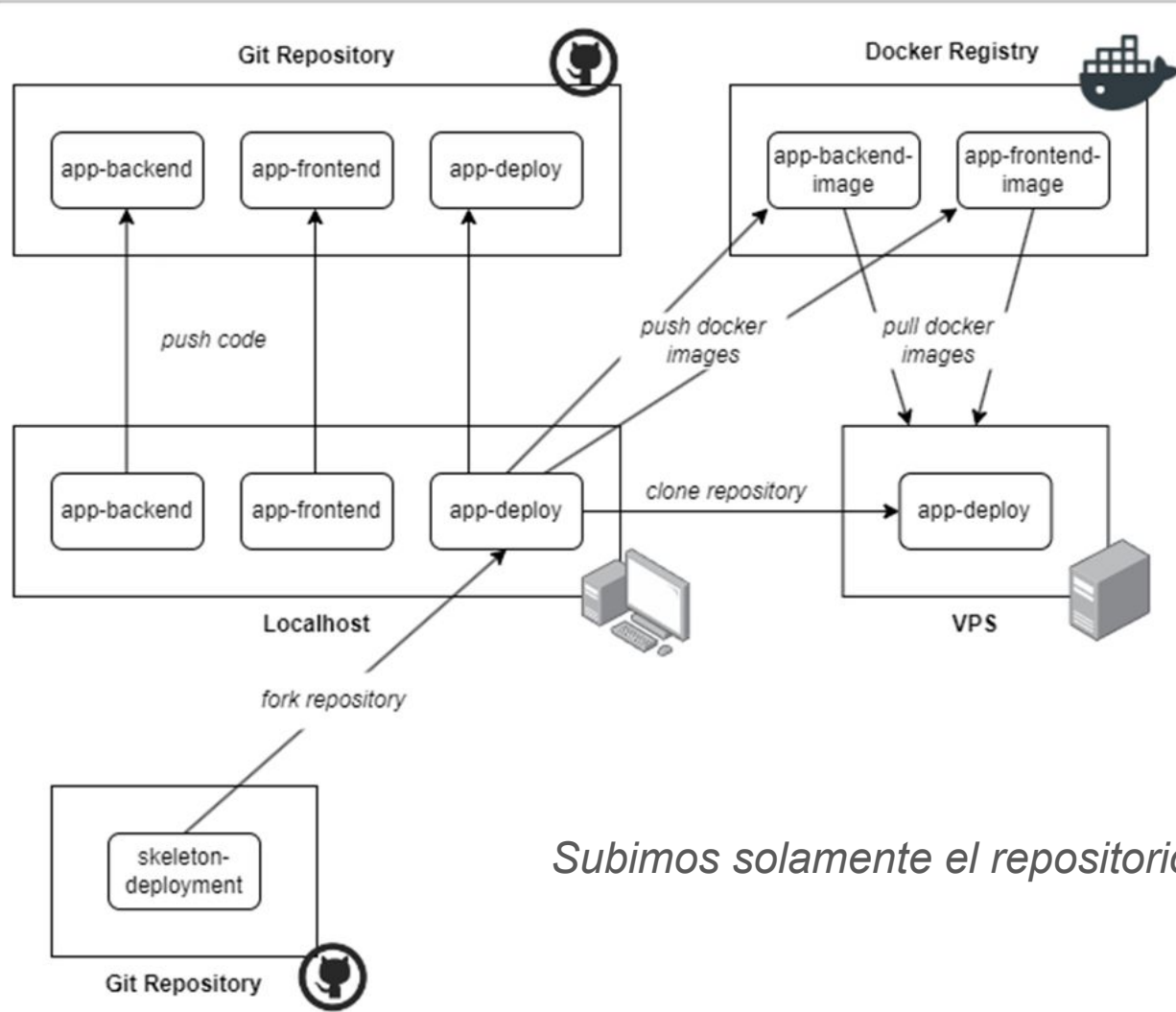


Estructura del repositorio de despliegue.

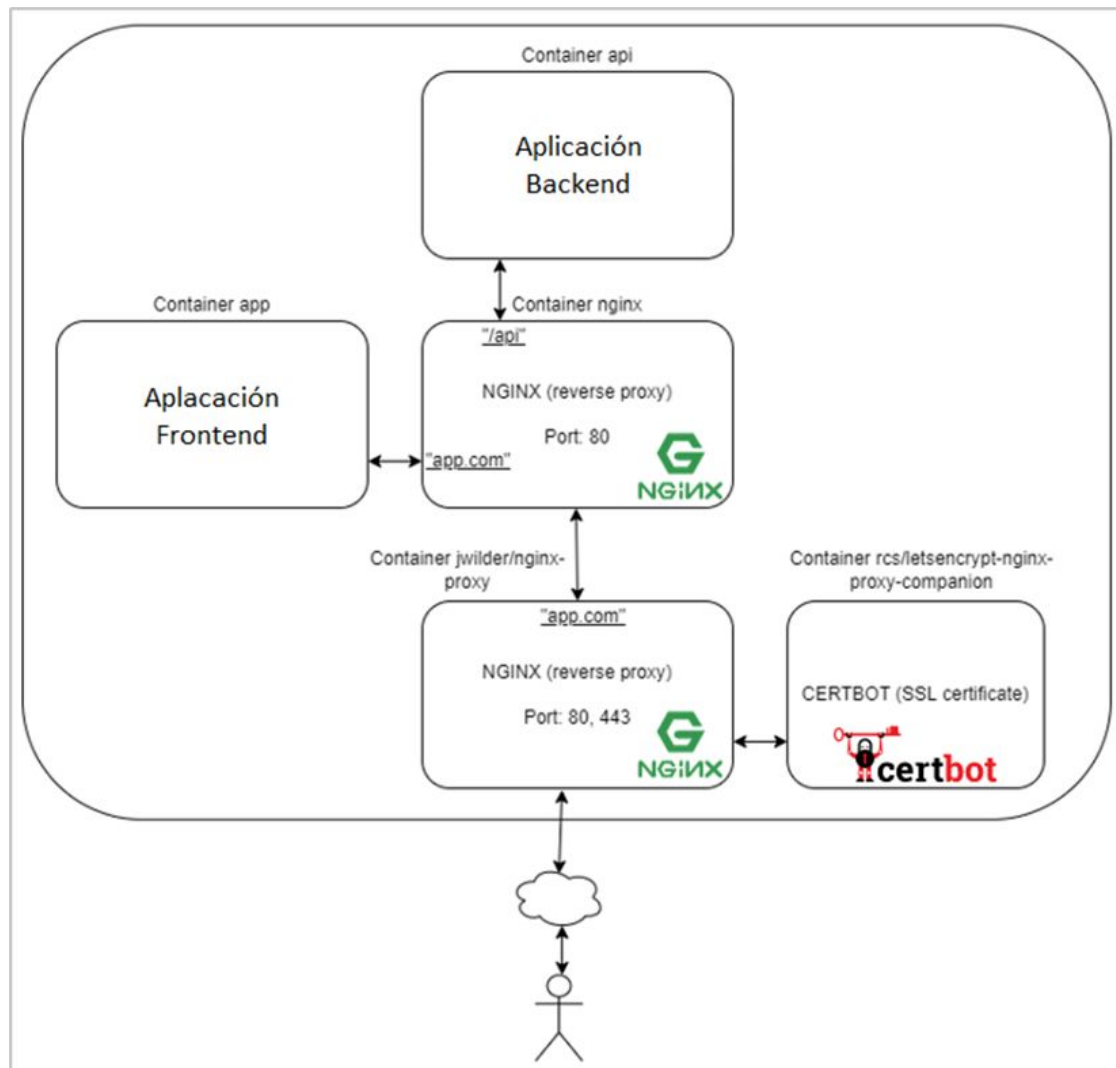


*En vez de subir las el código fuente al servidor...*





*Subimos solamente el repositorio de despliegue.*



# Ventajas del “Repositorio de despliegue”:

- No es necesario subir el código fuente del proyecto al servidor.
- Solo se clona el repositorio de despliegue en el servidor, ahorrando espacio y tiempo.
- No es necesario instalar programas y dependencias adicionales en el servidor, lo que facilita la gestión del mismo.
- El despliegue se realiza en segundos gracias a la automatización del proceso.
- Se puede migrar de servidor fácilmente, ya que todo lo necesario para el despliegue se encuentra en el repositorio de despliegue.
- Proceso más ordenado.
- Más desarrolladores involucrados en el despliegue.

# Conclusiones hasta aquí

El despliegue de aplicaciones Web en equipos de desarrollo de software pequeños presenta desafíos significativos.

Aunque el uso de contenedores Docker puede ser beneficioso, también introduce nuevas complejidades.

Se creó un repositorio plantilla para facilitar el despliegue de proyectos Web monolíticos en VPS.



1 - El problema a solucionar

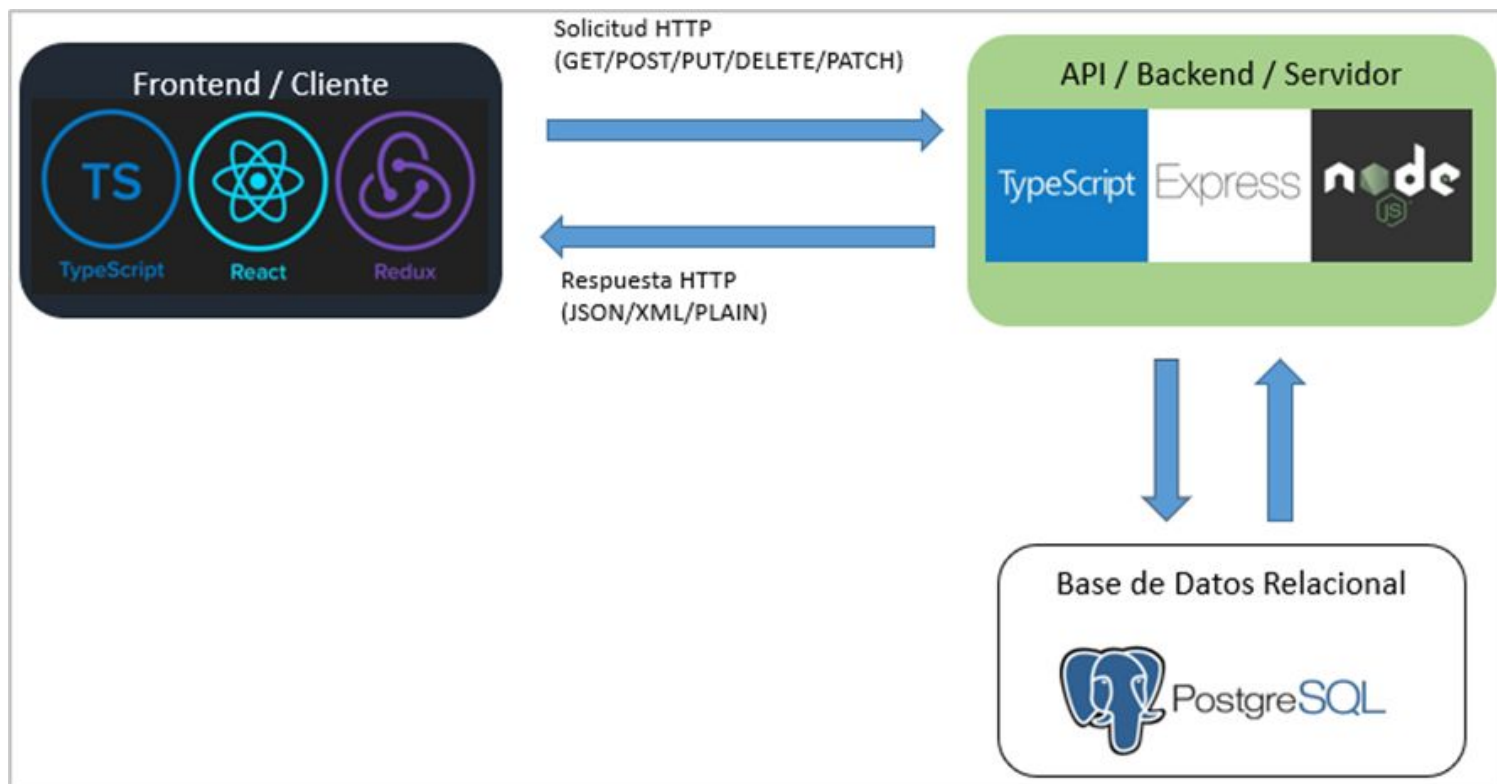
2 - Propuesta de solución

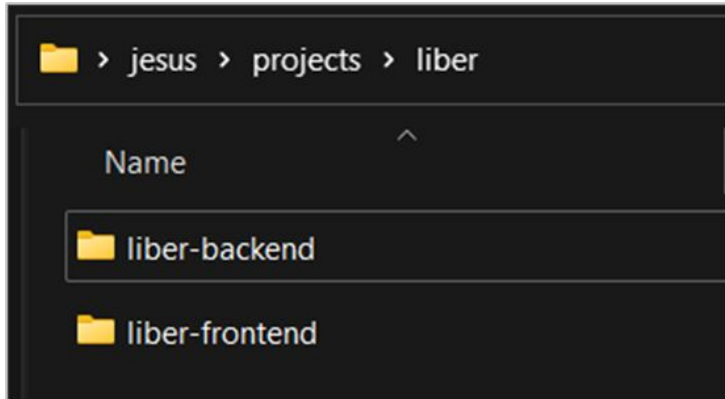
**3 - Resultados**

# Guía para el uso del repositorio

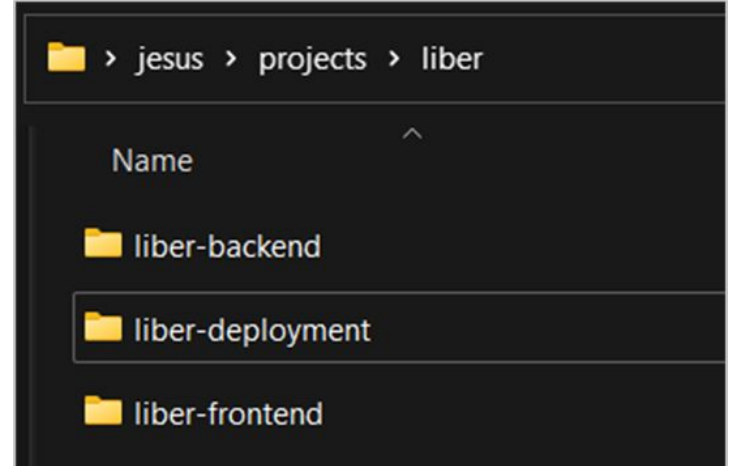
Ejemplo de utilización (Liber App web application)







clona el repositorio de despliegue en el directorio raíz del proyecto.



PS C:\Users\jesus\projects\mti\tfe> ls

Directory: C:\Users\jesus\projects\mti\tfe

Mode	LastWriteTime	Length	Name
d----	3/19/2023 5:38 PM		liber-backend
d----	3/19/2023 5:38 PM		liber-frontend

PS C:\Users\jesus\projects\mti\tfe> git clone https://github.com/jesusandres31/skeleton-deployment.git

Cloning into 'skeleton-deployment'...

remote: Enumerating objects: 11, done.

remote: Counting objects: 100% (11/11), done.

remote: Compressing objects: 100% (10/10), done.

remote: Total 11 (delta 1), reused 10 (delta 0), pack-reused 0

Receiving objects: 100% (11/11), done.

Resolving deltas: 100% (1/1), done.

PS C:\Users\jesus\projects\mti\tfe> ls

Directory: C:\Users\jesus\projects\mti\tfe

Mode	LastWriteTime	Length	Name
d----	3/19/2023 5:38 PM		liber-backend
d----	3/19/2023 5:38 PM		liber-frontend
d----	3/19/2023 8:54 PM		skeleton-deployment

PS C:\Users\jesus\projects\mti\tfe> mv .\skeleton-deployment\ liber-deployment

PS C:\Users\jesus\projects\mti\tfe> ls

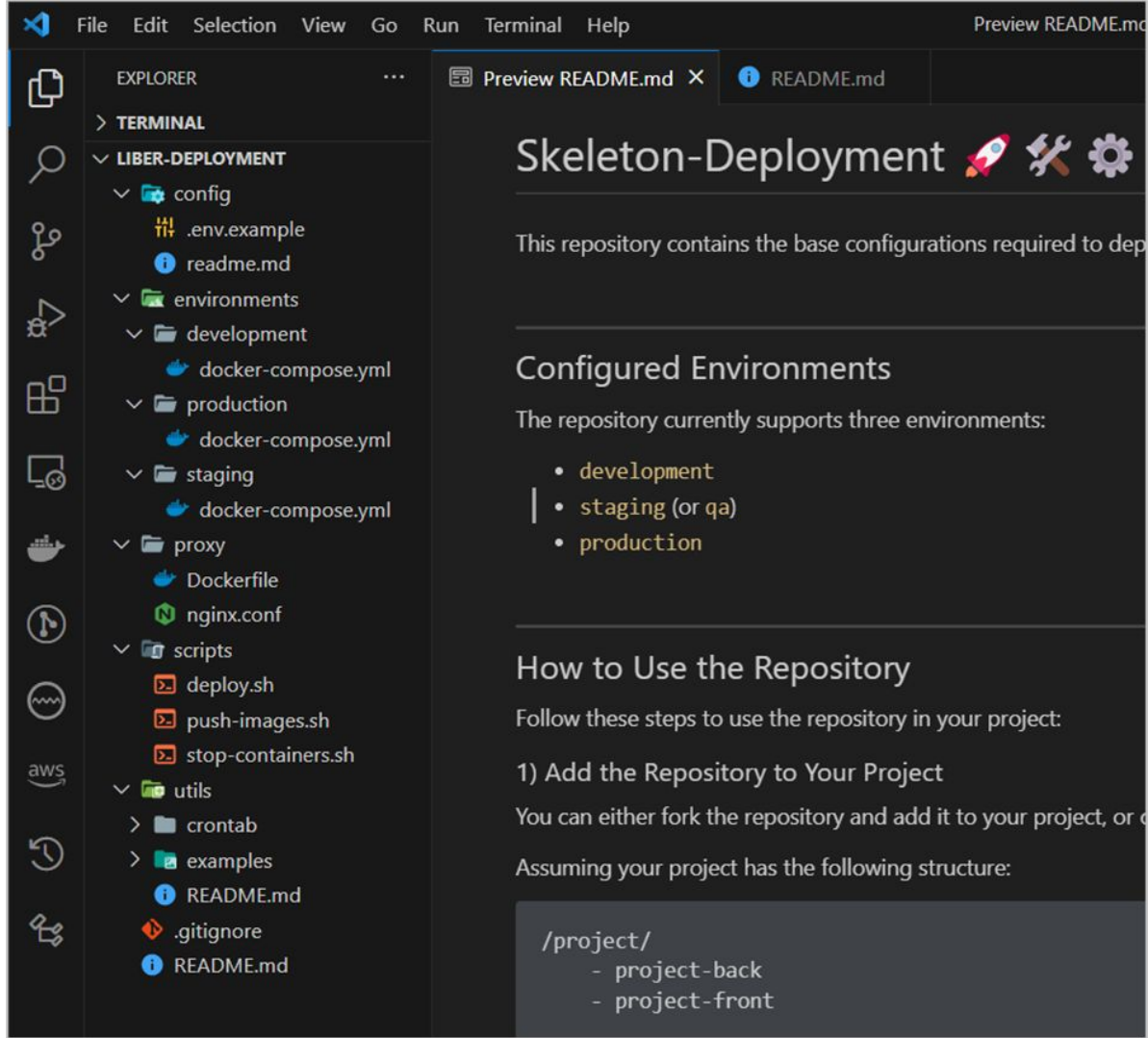
Directory: C:\Users\jesus\projects\mti\tfe

Mode	LastWriteTime	Length	Name
d----	3/19/2023 5:38 PM		liber-backend
d----	3/19/2023 8:54 PM		liber-deployment
d----	3/19/2023 5:38 PM		liber-frontend

PS C:\Users\jesus\projects\mti\tfe>

Empezamos a configurar...

- .env
- config del proxy
- base de datos



File Edit Selection View Go Run Terminal Help

Preview README.md X

# Skeleton-Deployment

This repository contains the base configurations required to dep

## Configured Environments

The repository currently supports three environments:

- development
- staging (or qa)
- production

## How to Use the Repository

Follow these steps to use the repository in your project:


### 1) Add the Repository to Your Project


You can either fork the repository and add it to your project, or c

Assuming your project has the following structure:




```
/project/  
- project-back  
- project-front
```

Repositorio de código fuente (Bitbucket).

 [Your work](#) [Pull requests](#) [Repositories](#)


 [Contact Devlights](#) / [Projects](#) / [Liber](#)

## Repositories

Name	Size
 <a href="#">liber-backend</a>	16 MB
 <a href="#">liber-frontend</a>	7.3 MB
 <a href="#">liber-deployment</a>	1012.2 KB

En las instrucciones del proyecto se encuentran los pasos para crear y registrar las imágenes del proyecto en un Container Registry

jesusandres31 > Liber > Container Registry



## There are no container images stored for this project

With the Container Registry, every project can have its own space to store its Docker images. [More Information](#)

### CLI Commands

If you are not already logged in, you need to authenticate to the Container Registry by using your GitLab username and password. If you have [Two-Factor Authentication](#) enabled, use a [Personal Access Token](#) instead of a password.

```
docker login registry.gitlab.com
```

You can add an image to this registry with the following commands:

```
docker build -t registry.gitlab.com/jesusandres31/liber .
```

```
docker push registry.gitlab.com/jesusandres31/liber
```



Se inicia sesión de la cuenta del Container Registry

✓ **TERMINAL**

Username (jesusandres31):

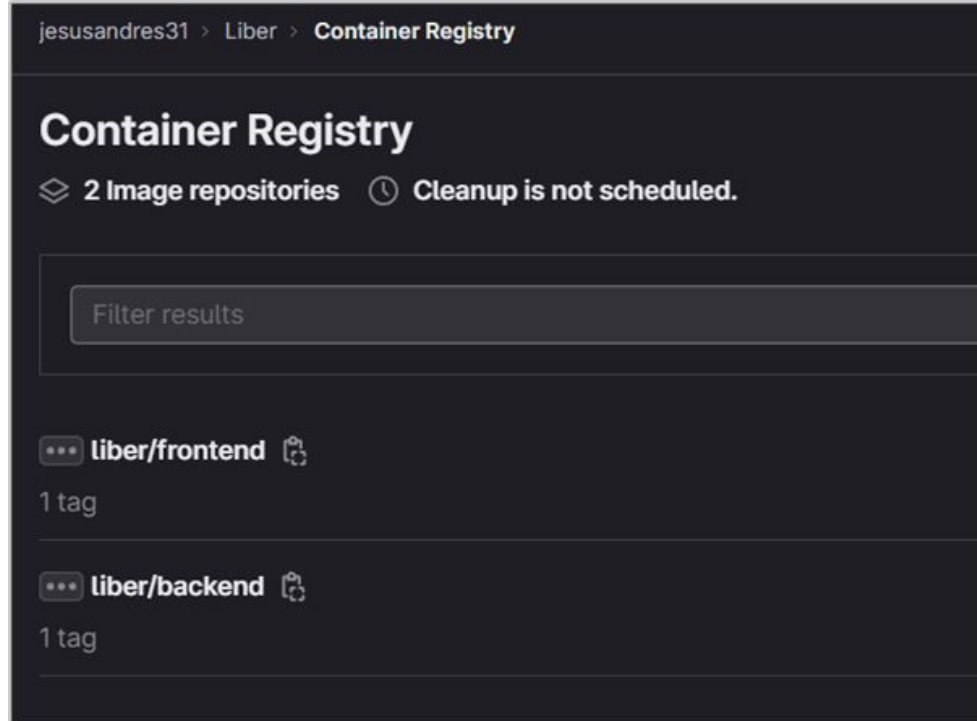
Password:

Login Succeeded

Se ejecuta el script que sube las imágenes del proyecto al Container Registry (push-images.sh).

```
jesus@DESKTOP-NKS091D MINGW64 ~/projects/liber/liber-deployment (main)
$ sh scripts/push-images.sh
liber-backend
/c/Users/jesus/projects/liber/liber-backend
Already on 'develop'
Your branch is up to date with 'origin/develop'.
Already up to date.
* develop
  feature/notifications
  master
[+] Building 15.9s (8/10)
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 32B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 34B 0.0s
=> [internal] load metadata for docker.io/library/node:14-alpine 1.7s
```

Repositorio de contenedores (Gitlab).



Se clona el repositorio de despliegue en el servidor VPS donde se desplegará el proyecto.

```
poli@poliserv:~/projects$ git clone https://github.com/jesusandres31/liber-deployment.git
Cloning into 'liber-deployment' ...
remote: Enumerating objects: 44, done.
remote: Counting objects: 100% (44/44), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 44 (delta 10), reused 42 (delta 8), pack-reused 0
Receiving objects: 100% (44/44), 7.64 KiB | 1.53 MiB/s, done.
Resolving deltas: 100% (10/10), done.
```

Se inicia sesión de la cuenta del Container Registry.

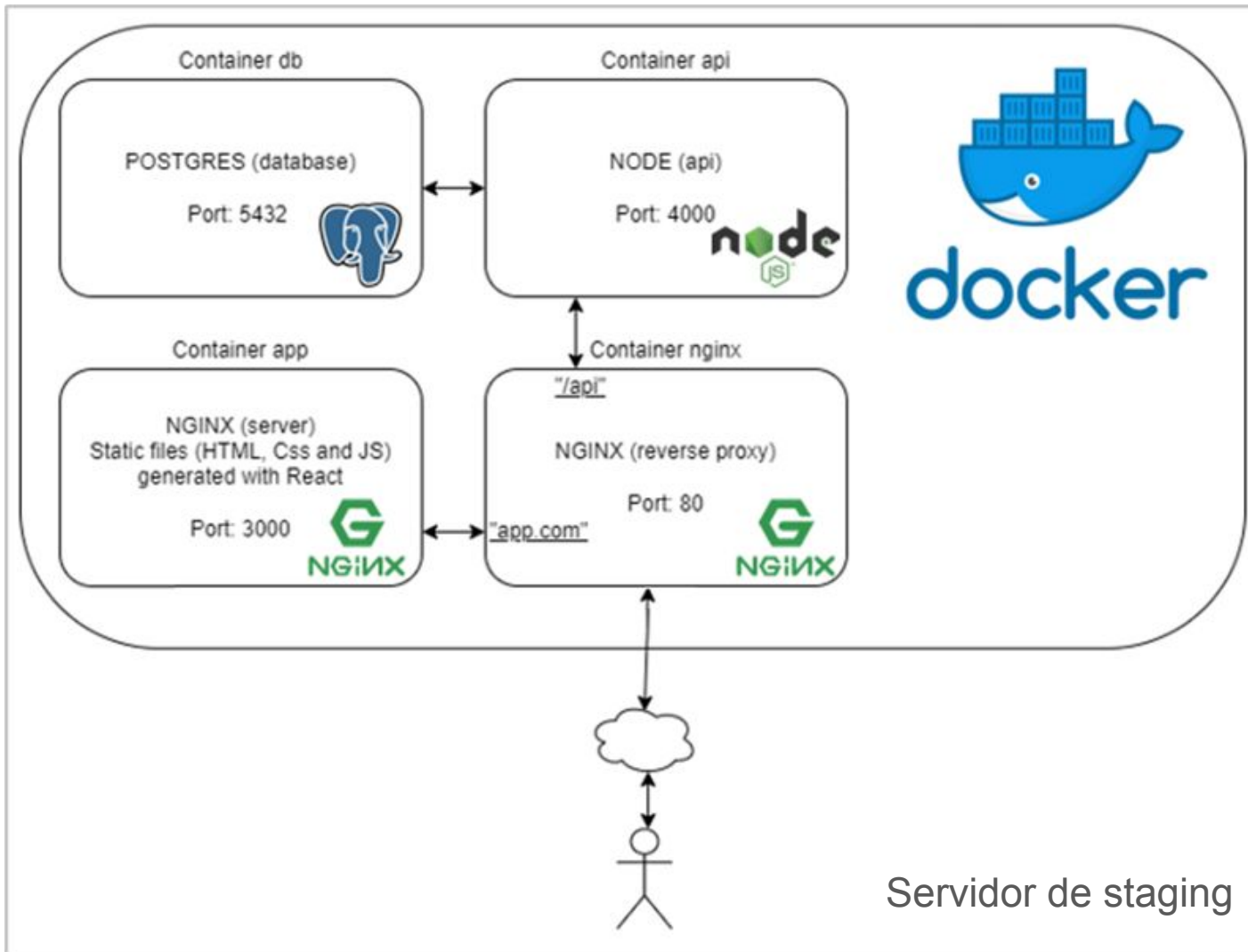
```
poli@poliserv:~/projects/liber/liber-deployment$ docker login registry.gitlab.com
Username: jesusandres31
Password:
```

En el servidor, se ejecuta el script de despliegue (deploy.sh).

```
[+] Running 5/5
  :: Network staging_default Created
  :: Container liber_db Started
  :: Container liber_api Started
  :: Container liber_app Started
  :: Container liber_nginx Started
Done!
```

Contenedores en ejecución.

```
jesus_zini@dev-tech-vm:~/liber/liber-production/env.staging/deploy$ sudo docker ps --format "t
0f610935fcfc liber_nginx 0.0.0.0:8081->80/tcp, :::8081->80/tcp
64bfecf1c9fb liber_app 80/tcp, 3000/tcp
63fa18ff1f18 liber_api 4000/tcp
5e926ef81f5c liber_db 5555/tcp, 0.0.0.0:5555->5432/tcp, :::5555->5432/tcp
```



# LIBΞR

## Iniciar sesión

Email



Contraseña



☐ Recordarme

INGRESAR

[¿Olvidaste tu contraseña?](#)

El desarrollador puede detener la aplicación ejecutando el script correspondiente (stop-containers.sh).

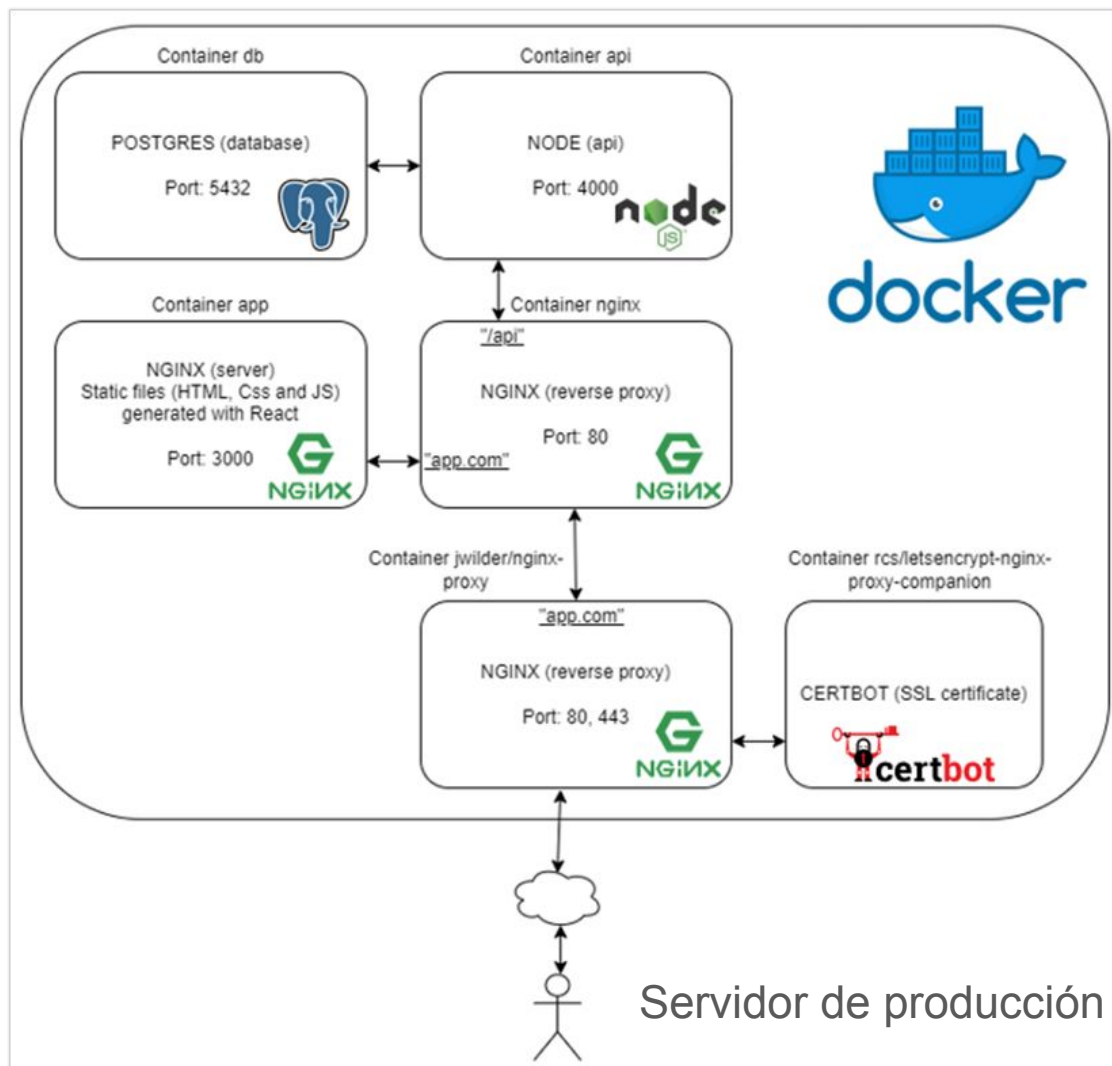
```
poli@poliserv:~/projects/liber/liber-deployment$ sh scripts/stop-containers.sh -s
Stop STAGING env ...
Stopping containers:
[+] Running 5/5
 # Container liber_nginx      Removed
 # Container liber_app        Removed
 # Container liber_api        Removed
 # Container liber_db         Removed
 # Network staging_default    Removed
Done!
```

Para desplegar una nueva versión de la aplicación, el desarrollador sube las imágenes actualizadas al Container Registry y ejecuta nuevamente el script de despliegue en el servidor



## Script de despliegue

```
34 while getopts ":psd" opt; do
35     case $opt in
36         p)
37             echo "Start PRODUCTION env..."
38             cd environments/production
39             deploy
40             ;;
41         s)
42             echo "Start STAGING env..."
43             cd environments/staging
44             deploy
45             ;;
46         d)
47             echo "Start DEVELOPMENT env..."
48             cd environments/development
49             deploy
50             ;;
51         \?)
52             echo "Invalid Option: -$OPTARG"
53             exit 1
54             ;;
55     esac
56 done
```



## Contenedores en ejecución...

```
PowerShell
[root@vps-2384053-x ~] # docker ps --format "table {{.ID}}\t{{.Image}}\t{{.Ports}}"
CONTAINER ID    IMAGE                                PORTS
e24d7a68607a    envproduction_nginx                80/tcp
3b5ff703aa17    envproduction_app                  80/tcp, 3000/tcp
a5b1aec08c3d    envproduction_api                  4000/tcp
4443183c1057    jrcs/letsencrypt-nginx-proxy-companion
ba1120298f39    jwilder/nginx-proxy                0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp
fee40ec722e1    postgres:12-alpine                 5432/tcp
```



Liber App



https://app.liberarg.com/login



# LIBER

## Iniciar sesión

Email



Contraseña

☐

Recordarme

INGRESAR

[¿Olvidaste tu contraseña?](#)

# Conclusiones

- Se abordaron los desafíos del despliegue de aplicaciones web en equipos pequeños de desarrollo de software.
- Se ha analizado y seleccionado buenas prácticas de la industria del software, y se construyó un procedimiento para desplegar aplicaciones web con arquitectura monolíticas en VPS,
- La implementación de soluciones como la propuesta en este trabajo permite la automatización del proceso de despliegue y el compartir conocimiento entre desarrolladores.
- Si bien la propuesta concreta de este estudio no es aplicable en todos los casos en específico, se recomienda considerar la adopción de un marco de trabajo para el despliegue de aplicaciones que involucren la documentación y el almacenamiento de archivos relevantes a dicho proceso.

# Trabajos Futuros

- Ampliar el alcance de la solución para soportar mayor variedad de lenguajes y tecnologías. Hacerlo más versátil y adaptable a diferentes tipos de aplicaciones.
- Agregar un script o tarea programada (scheduler) para el backup automático de base de datos.
- Creación de una aplicación panel web que sirva como monitor de toda la aplicación.