

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

## Introducción a la contenerización

La contenerización es una forma de virtualización del sistema operativo en la que se ejecutan aplicaciones monolíticas y/o microservicios [4], en espacios de usuario aislados llamados contenedores, en un entorno informático portable, empaquetado, aislado, con los insumos necesarios para que una aplicación requiera ejecutar (binarios, bibliotecas, dependencias, archivos de configuración), permitiendo la interoperabilidad de grupos de trabajo basados en DevSecOps (Developer – Security - Operations) [1], que tienen procesos y procedimientos de flujos en integración y entrega continua (Pipeline CI/CD) [2], en un ambiente controlado con el sistema operativo base implementado en Bare Metal, On-Premises y/o Cloud Computing [3].



**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

#### Ventajas y beneficios

- **Portabilidad:** Un contenedor con servicios y/o microservicios, puede ser generado, transferido, y ejecutado en diferentes ambientes front and back end (desarrollo, preproducción, producción), en la plataforma transversal Institucional.
- **Velocidad:** Los equipos de desarrolladores pueden generar servicios y/o microservicios en contenedores reduciendo el tiempo de la implementación en los ambientes propuestos por la plataforma transversal, así como también pueden reutilizarse para otros proyectos gracias al acceso a registro de imágenes base.
- **Escalabilidad:** La tecnología del contenedor permite el crecimiento vertical, mediante reconfiguraciones automatizadas en la administración de recursos de hardware, dentro de la plataforma transversal, para los diferentes ambientes, con base al ciclo de vida del proyecto.
- **Agilidad:** El motor para ejecutar contenedores (Runtimes) [5], para diferentes sistemas operativos para monousuario o multiusuario bajo la administración de la iniciativa Open Container Initiative (OCI) [6], permitiendo que los equipos de desarrolladores puedan utilizar herramientas y procesos en DevSecOps.
- **Eficiencia:** Las aplicaciones ejecutadas en entornos en contenedores comparte el kernel del sistema operativo del host Bare Metal, los desarrolladores pueden compartir los servicios y/o microservicios con tiempos de respuesta ágiles para su implementación en los diferentes ambientes de la plataforma transversal, reduciendo los costos de operación, y licenciamiento.
- **Aislamiento de eventualidades:** La contenerización permite aislar los procesos con base a la lógica del servicio y/o microservicio, por lo que si ocurre una excepción en tiempo de ejecución el contenedor con eventualidades puede ser removido sin afectar a los otros contenedores que estén en el front and back end, para reducir riesgos en la operación en los diferentes ambientes.
- **Privacidad y seguridad:** La independencia de los contenedores evita que el código malicioso afecte a otros servicios y/o microservicios permitiendo niveles de permisos a nivel del sistema operativo para bloquear acceso de forma automática a componentes o comandos no deseados.
- **Facilidad de administración:** Con una plataforma transversal para la orquestación de los contenedores dentro del Instituto, puede automatizar la instalación, administración, escalabilidad, registro de versiones, depuración por ciclo de vida terminado, en la operación de servicios y/o microservicios en contenedores.
- **Continuidad de operaciones:** La plataforma transversal de contenerización permite que en caso de contingencia puedan transferirse manual o automáticamente los contenedores front and backend que tengan actividades de misión crítica.

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

#### Ambiente de implementación

Con base en las mejores prácticas del Cloud Native Computing Foundation (CNCF) [7], es pertinente generar una plataforma transversal orientada a la implementación de la Container Runtime Interface (CRI) [8], para que utilice instancias y entornos en tiempo de ejecución [Runtimes & Engines: (runC, crun y Kata containers)] avalados por la Open Container Initiative (OCI), considerando las siguientes características:

- **Entorno de almacenamiento (Storage Interface):** La plataforma transversal de contenerización deberá estar conectada a los esquemas de almacenamiento SAN (Storage Area Network), NAS (Network Attached Storage) [9], para que los contenedores generen volúmenes virtuales a físicos, con base al ciclo de vida del proyecto.
- **Entornos en tiempo de ejecución (Runtime):** La plataforma transversal de contenerización deberá estar conectada a la estrategia de utilizar herramienta para los entornos de desarrollo con técnicas basadas en DevSecOps
- **Entorno en conectividad (Networking Interface):** La plataforma transversal de contenerización deberá tener un direccionamiento de red de tipo interno y externo para el acceso a los servicios y/o microservicios basados en contenedores, así como también utilizar un dominio con balanceo global y certificado de seguridad de tipo "wildcard", para la transferencia de datos [10].
- **Entorno de registro (Registry):** La plataforma transversal de contenerización deberá tener un repositorio que permite tener el registro de las imágenes base que utilizan los contenedores para habilitar servicios y/o microservicios en el flujo automatizado con base al ciclo de vida del proyecto [11].
- **Privacidad:** La plataforma transversal de contenerización deberá tener un entorno de integridad, confidencialidad, disponibilidad de los contenedores generados para los ambientes de desarrollo, preproducción y producción con base al ciclo de vida del proyecto [12].



**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

Nota: una de la estrategia con mayor madurez para orquestar arquitecturas basadas en contenerización es CRI-O [13], el grupo conformado para la especificación, herramientas y control de versiones está en la referencia [14].

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Orquestación de contenedores con base a PODS [Almacenamiento, Procesamiento, Transferencia]**

- Lugar [On-premises - Cloud]
- Aplicación [Monolítico, Microservicios]
- Lenguaje [Java, .NET, Python entre otros]
- Sistema Operativo [Kernel Linux]
- Implementación a escala
- Programación de cargas de trabajo
- Supervisión de estado
- Conmutación cuando se produce una eventualidad en un nodo
- Escalado o reducción vertical
- Funciones de red
- Detección de servicios
- Coordinación de las actualizaciones de aplicaciones
- Afinidad de nodos de clúster
- Seguridad [Disponibilidad, Confidencialidad, Integridad]



**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Seguridad de imágenes de contenedores:** estas capacidades se integran directamente con los entornos de desarrollo existentes, lo que ayuda a garantizar que las imágenes de contenedores comiencen su vida útil de acuerdo con las mejores prácticas de diseño moderno. Esto incluye escanear en busca de vulnerabilidades conocidas o de día cero en las imágenes, evitar que sean infectadas por malware, no permitir que las credenciales codificadas se filtren en ellas, etc. Los resultados de los escaneos de vulnerabilidad de contenedores deben alinearse y clasificarse de acuerdo con los modelos de evaluación de riesgos.

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Seguridad del registro de contenedores:** proporciona visibilidad, control de acceso y seguridad continuos para las imágenes de contenedor almacenadas en los registros, lo que garantiza que las imágenes válidas no se vean comprometidas y se evite el acceso no autorizado, las modificaciones de imágenes o la infiltración de contenedores no autorizados.





**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Seguridad de la plataforma de orquestación:** la plataforma de orquestación de contenedores en sí debe estar protegida adecuadamente en todas las capas de su infraestructura subyacente, desde la seguridad de los sistemas host hasta la implementación de la segmentación de la red, el aislamiento de la carga de trabajo y la protección de todas las interfaces de administración. Se deben implementar tanto el endurecimiento proactivo como el monitoreo en tiempo real, junto con la administración de la configuración y el gobierno de acceso integral, haciendo cumplir los principios de segregación de funciones y privilegios mínimos.

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Monitoreo de contenedores en tiempo de ejecución:** proporciona visibilidad continua en tiempo real de las actividades dentro de los contenedores en ejecución, utilizando tanto la detección basada en firmas como el análisis de comportamiento impulsado por ML para identificar amenazas en tiempo de ejecución. Las plataformas de seguridad de contenedores deben utilizar toda la gama de controles de seguridad en los niveles de host, red, contenedor y aplicación para bloquear o mitigar las amenazas detectadas de forma rápida y automática.



**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Gestión de incidentes:** estas capacidades permiten a los analistas de seguridad reaccionar rápidamente a las amenazas identificadas, realizar investigaciones forenses, tomar las decisiones correctas y, finalmente, automatizar la remediación de amenazas utilizando una combinación de controles de orquestación nativos y herramientas de seguridad especializadas.

**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Auditoría y cumplimiento:** el cumplimiento normativo es un desafío importante y, al mismo tiempo, un impulsor del negocio para organizaciones de cualquier tamaño o industria. La retención de datos de seguridad y los informes de cumplimiento completos son las capacidades básicas aquí. El soporte listo para usar para marcos regulatorios como GDPR, HIPAA o PCI es un diferenciador importante para muchos clientes.



**Plataformas de diseño de microservicios:** Herramientas como Kubernetes, Podman, Docker.

**Integraciones:** las soluciones de seguridad de contenedores no pueden funcionar como herramientas independientes sin integraciones profundas con servicios en la nube existentes, plataformas de orquestación de contenedores, canalizaciones DevOps y DevSecOps, así como plataformas SIEM y otras herramientas de operaciones de seguridad. Mantener un ecosistema abierto de 3Rd Las integraciones de partes son un diferenciador clave para los proveedores.

Podman:

Es una herramienta de administración de contenedores que se utiliza para crear, administrar y ejecutar contenedores en sistemas Linux. A menudo se compara con Docker, otra herramienta popular de administración de contenedores. Sin embargo, Podman tiene algunas características únicas que lo distinguen de Docker y lo hacen especialmente atractivo para ciertos casos de uso y entornos.

Origen y Motivación:

Podman es un proyecto de código abierto desarrollado por Red Hat y la comunidad de código abierto. Fue diseñado para abordar las limitaciones de Docker, especialmente en entornos donde se necesitaba un enfoque más seguro y compatible con Kubernetes.



#### Características Clave:

- **Daemonless:** A diferencia de Docker, Podman no requiere un daemon (proceso en segundo plano) para ejecutar contenedores. Esto mejora la seguridad y facilita la administración de contenedores.
- **Compatibilidad con Docker:** Podman es compatible con la mayoría de los comandos de Docker, lo que facilita la transición desde Docker a Podman.
- **Pods:** Los pods son una característica exclusiva de Podman que permite agrupar varios contenedores y compartir el mismo espacio de red y volúmenes. Esto es útil para aplicaciones que necesitan comunicarse entre sí de manera eficiente.
- **Rootless:** Podman permite a los usuarios ejecutar contenedores sin privilegios de root, lo que mejora la seguridad.
- **CRI-O Integration:** Podman se integra con CRI-O, un proyecto de contenedores OCI (Open Container Initiative) que es una parte fundamental de Kubernetes. Esto facilita la ejecución de contenedores en entornos Kubernetes.

#### Ventajas:

**Seguridad Mejorada:** El enfoque "sin daemon" y la capacidad de ejecutar contenedores sin privilegios de root hacen que Podman sea más seguro en comparación con Docker.

**Compatibilidad con Docker:** Los usuarios que están familiarizados con Docker pueden cambiar a Podman sin una curva de aprendizaje significativa.

**Pods:** La capacidad de crear pods es útil para aplicaciones que requieren múltiples contenedores trabajando juntos.





#### Casos de Uso:

Podman es adecuado para una variedad de casos de uso, incluyendo el desarrollo de aplicaciones, la implementación de contenedores en servidores y la orquestación de contenedores en Kubernetes.

#### Comunidad y Soporte:

Podman cuenta con una comunidad activa de desarrolladores y usuarios que brindan soporte y contribuyen al proyecto. Además, Red Hat ofrece soporte comercial para empresas que utilizan Podman en entornos empresariales.


<https://podman-desktop.io/>

Ejemplo de Uso: crear y ejecutar un contenedor con Podman:

```
podman run -it --rm ubuntu:20.04 /bin/bash
```

#### Referencias

- [1] O. Díaz, M. Muñoz and J. Mejía, "Responsive infrastructure with cybersecurity for automated high availability DevSecOps processes," 2019 8th International Conference On Software Process Improvement (CIMPS), 2019, pp. 1-9, doi: 10.1109/CIMPS49236.2019.9082439.
- [2] Khushalani, Prateek. (2022). CI/CD Systems. 10.1007/978-1-4842-8032-4\_3.
- [3] Tfrifonov, Daniel & Valchanov, Hristo. (2018). VIRTUAIZATION AND CONTAINERIZATION SYSTEMS FOR BIG DATA.
- [4] Jha, Devki Nandan & Garg, Saurabh & Jayaraman, Prem Prakash & Buyya, Rajkumar & Li, Zheng (Eddie) & Morgan, Graham & Ranjan, R.. (2019). A study on the evaluation of HPC microservices in containerized environment. Concurrency and Computation: Practice and Experience. 33. 10.1002/cpe.5323.
- [5] Wang, Xingyu & Du, Junzhao & Liu, Hui. (2022). Performance and isolation analysis of RunC, gVisor and Kata Containers runtimes. Cluster Computing. 25. 1-17. 10.1007/s10586-021-03517-8.
- [6] <https://opencontainers.org/community/overview/>
- [7] <https://www.cncf.io/about/members/>
- [8] <https://github.com/kubernetes/kubernetes/blob/242a97307b34076d5d8f5bbeb154fa4d97c9ef1d/docs/devel/container-runtime-interface.md>
- [9] (1) Gandhi, Arun & Varki, Elizabeth & Bhatia, Swapnil. (2002). Reader-Writer Locks for Network Attached Storage and Storage Area Networks.. 402-405. (2) N. Zhao et al., "Large-Scale Analysis of Docker Images and Performance Implications for Container Storage Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 4, pp. 918-930, 1 April 2021, doi: 10.1109/TPDS.2020.3034517.
- [10] M. M. Khalel, M. Arul Pugazhendhi and G. R. Raj, "Enhanced Load Balancing in Kubernetes Cluster By Minikube," 2022 International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN), 2022, pp. 1-5, doi: 10.1109/ICSTSN53084.2022.9761317.
- [11] Buchanan, Steve & Rangama, Janaka & Bellavance, Ned. (2020). Container Registries. 10.1007/978-1-4842-5519-3\_2.
- [12] Colman, Matt. (2022). Containers and Kubernetes: Security is not an Afterthought. ITNOW. 64. 44-45. 10.1093/itnow/bwac023.
- [13] <https://cri-o.io/>
- [14] <https://github.com/opencontainers>



Windows

Podman Desktop for Windows

Download Now

Windows installer x64, version v1.4.0

Other Windows downloads:

Installer:

x64

arm64

Portable binary:

x64

arm64

Installer for restricted environments:


x64

arm64

Package Managers Guide

Using winget? Install in one command:

winget install -e --id Re...



macOS

Podman Desktop for macOS

Download Now

Universal \*.dmg, version v1.4.0

Other macOS downloads:


Intel

Arm

Disk Image for restricted environments

Using Brew? Install in one command:

brew install podman-desktop



Linux

Podman Desktop for Linux

Download Now

Linux \*.flatpak, version v1.4.0

Other Linux downloads:

AMD64 binary (tar.gz)

Using FlatHub ? Install in one command:

flatpak install flathub io...

<https://podman-desktop.io/downloads>

PS C:\WINDOWS\system32> podman

Manage pods, containers and images

Usage:

podman.exe [options] [command]

Available Commands:

attach

Attach to a running container

build

Build an image using instructions from Containerfiles

commit

Create new image based on the changed container

container

Manage containers

cp

Copy files/folders between a container and the local filesystem

create

Create but do not start a container

diff

Display the changes to the object's file system

events

Show podman system events

exec

Run a process in a running container

export

Export container's filesystem contents as a tar archive

generate

Generate structured data based on containers, pods or volumes

healthcheck

Manage health checks on containers

help

Help about any command

history

Show history of a specified image

image

Manage images

images

List images in local storage

import

Import a tarball to create a filesystem image

info

Display podman system information

init

Initialize one or more containers

inspect

Display the configuration of object denoted by ID

kill

Kill one or more running containers with a specific signal

kube

Play containers, pods or volumes from a structured file

load

Load image(s) from a tar archive

login

Login to a container registry

logout

Logout of a container registry

logs

Fetch the logs of one or more containers

machine

Manage a virtual machine

manifest

Manipulate manifest lists and image indexes

network

Manage networks

pause

Pause all the processes in one or more containers

pod

Manage pods

port

List port mappings or a specific mapping for the container

ps

List containers

pull

Pull an image from a registry

push

Push an image to a specified destination

rename

Rename an existing container

restart

Restart one or more containers

rm

Remove one or more containers

rmi

Remove one or more images from local storage

run

Run a command in a new container

save

Save image(s) to an archive

search

Search registry for image

secret

Manage secrets

start

Start one or more containers

stats

Display a live stream of container resource usage statistics

stop

Stop one or more containers

system

Manage podman

tag

Add an additional name to a local image

top

Display the running processes of a container

unpause

Unpause the processes in one or more containers

untag

Remove a name from a local image

update

Update an existing container

version

Display the Podman version information

volume

Manage volumes

wait

Block on one or more containers

Options:

-c, --connection string

Connection to use for remote Podman service (default "podman-machine-default")

--help

Help for podman

--identity string

path to SSH identity file, (CONTAINER\_SSHKEY) (default "C:\\Users\\oswaldos.diaz\\.ssh\\podman-machine-default")

--log-level string

Log messages above specified level (trace, debug, info, warn, warning, error, fatal, panic) (default "warn")

--nooc

do not output to stdout

--ssh string

define the ssh mode (default "golang")

--storage-opt stringarray

Used to pass an option to the storage driver

--url string

URL to access Podman service (CONTAINER\_HOST) (default "ssh://user@127.0.0.1:54745/run/user/1000/podman/podman.sock")

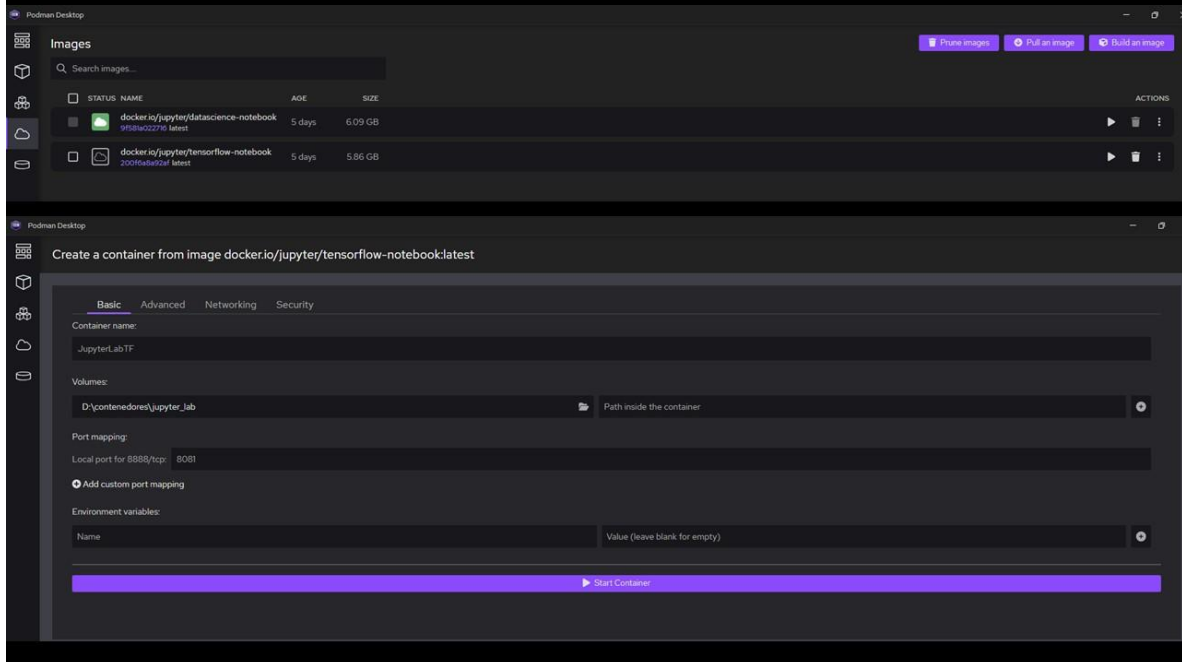
Maestría en ciencia de datos | Ingeniería de datos – agosto 2023 | edgar.diaz@ucags.edu.mx



Universidad  
de la Ciudad de  
Aguascalientes

podman pull jupyter/datascience-notebook

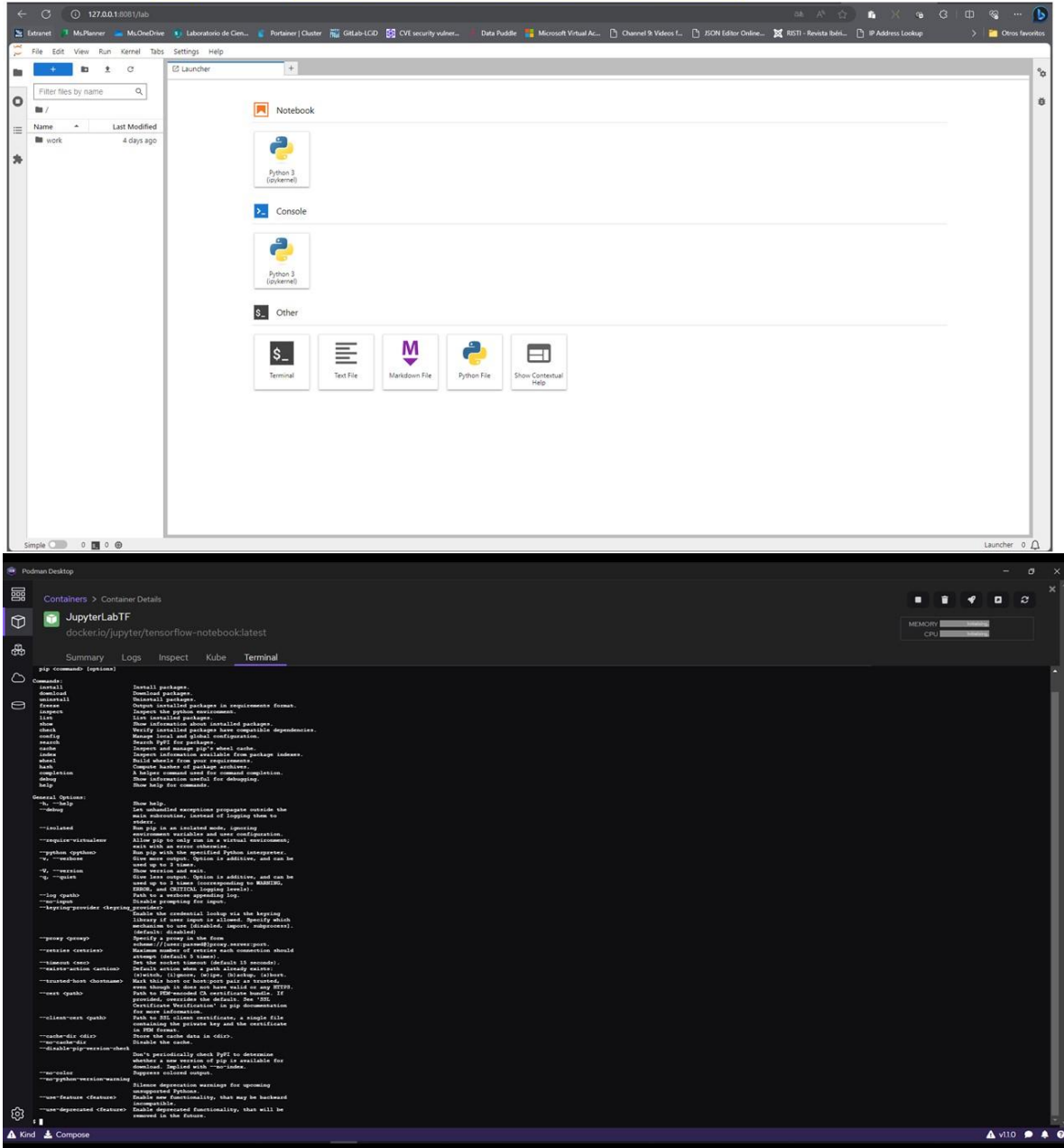
```
PS C:\WINDOWS\system32> podman pull jupyter/datascience-notebook
Resolving "jupyter/datascience-notebook" using unqualified-search registries (/etc/containers/registries.conf.d/999-podman-machine.conf)
Trying to pull docker.io/jupyter/datascience-notebook:latest...
Getting image source signatures
Copying blob sha256:e400b2a4039cadfc753f6a514b252d18c9eeda7cda426f0c5047858f1b1cd479
Copying blob sha256:445a6a12be54b4da18d7c7794a41bc4746bc422f1f4325a60ff4fc7ea2e5d
Copying blob sha256:11e1ee1abc5d8e5f6312a3f233135a5fc8b8ffe70e911d5499ad61ece828a839
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:0f8cf0b1206d25952244faeeb085f5383201c7a824c65287b34b293eb258c3e
Copying blob sha256:5d5ce3b8d94ae4da40d5f41e0662773237cc6c0db1671a10b58ee899b8f84
Copying blob sha256:69e63b115056ef369e03789d6d46833c53aa594504cf0b51a51deb92fc3ae2
Copying blob sha256:076dc83d0f3fdadb5fea52d412db55ef48f1db82423965fe93ad5ae2e8436ce6
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:d10452475fc5533598b02cbea6921f38b56c945562bf2efdb3b1365894cda4c7d
Copying blob sha256:62356d69e3f25d9205f2b6819310fa2272c30f50413aa4fa43780b706991f
Copying blob sha256:46fc3672055880932d458b05b51a922f59d008e7acb3edf7e28edbf75a194f43
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:a2f7efa28e0c135006702ff6065f3477b51c866aa78d066e4ac05b52fd5e75f
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:ee5f0e7c1e8729191ade50b347ba5c1d94bfe901a08e63a6d14455bd29b40d5
Copying blob sha256:cbc32f5198e3dec1b34e363f8394cd85c5261f368737cf80bebf74d9c41157b
Copying blob sha256:8884824b46e2fc968ac6253842d1703d03ea3ba20479f24f204a9b046d58072
Copying blob sha256:b3ee3983cd17c4a029ff5319ff6b40792c5b88cbcd0d4edfb013976a0b4ba9a5
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:ba7b13690c4a6851f2b79c4ff013ed9178c05a0013a4c4ff70e865740515cae
Copying blob sha256:94b8438ac8efff8b3f457a3837a85cc0056ce852aefc3cdda77e8543c751a91
Copying blob sha256:7dcb9cf837cb7554afbc5ef173d0f85ac7b1ca2b1225b9e79c4a39b132bc
Copying blob sha256:0c4b28054baaa4e9abc5ef99c9489a9783b7b2c05487ba1a8cccb95bb4106873
Copying blob sha256:60c60f4846b94732bc4504554c9e025cf91f6280297875489d27778b072c03e0
Copying blob sha256:93caab7240275c8aa8bdc1434d50238c4e6bdadfb8e2a1233fb4447095421
Copying blob sha256:47597c6677ecb376db400723496453b29ea8cc3678cde4169768fdda901355ea
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:7d10f7e835cd4536f64838e800d986bfa60d53901d0342c8e21cff98873c2dd
Copying blob sha256:69842e2748f611bce6b54783ba45e33f85f093848d91acdd0fc1b1e693661888
Copying blob sha256:444fb700ef54461cfa02571ae0db9a0dc1e0cd5577484a6d75e68dc38e8ac1
Copying blob sha256:687f6af67b21b4766195d4f8aa41bd298872109e44aa9067a0a5cc76a1104022
Copying blob sha256:9d3959174e346624654e12445a69b705818471c75309314ac60ff8a0bb3b7e86
Copying blob sha256:dbe4d322f8593e1ec6980fa6f34909d2760ebadad335192eabb6ada70293652
Copying blob sha256:9eafaa128fc604e244fcab2ca689696f5697c69ed2ae4060d86cc41bacfb
Copying config sha256:9f581a022716e38d45d7b8a7ec06ee75a7db0e4b40a91fc301d1d9b83bc189bb
Writing manifest to image destination
Storing signatures
9f581a022716e38d45d7b8a7ec06ee75a7db0e4b40a91fc301d1d9b83bc189bb
PS C:\WINDOWS\system32>
```



The screenshot displays the Podman Desktop interface. The top section shows a list of containers with columns for STATUS, NAME, IMAGE, AGE, and ACTIONS. Two containers are listed: JupyterLabTF (running, 16 seconds) and JupyterLab (running, 30 minutes). The bottom section shows the details for the JupyterLabTF container, including its name, image, and a summary of its state. The summary indicates that the container is running and has a status of 'Running'.



<http://127.0.0.1:8081/lab?token=c70dae7297075f44425aecef2c4d93f2fab57dbf098be39e>







Podman Desktop

Containers > Container Details

JupyterLabTF

docker.io/jupyter/tensorflow-notebook:latest

Summary Logs Inspect Kube Terminal

Summary

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bbc8aa366ecd	docker.io/jupyter/datascience-notebook:latest	start-notebook.sh	41 minutes ago	Up 41 minutes (healthy)	0.0.0.0:8080->8888/tcp	JupyterLab
a7a34fd8367b	docker.io/jupyter/tensorflow-notebook:latest	start-notebook.sh	11 minutes ago	Up 11 minutes (healthy)	0.0.0.0:8081->8888/tcp	JupyterLabTF

PS D:\contenedores\jupyter\_lab> podman ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

bbc8aa366ecd docker.io/jupyter/datascience-notebook:latest start-notebook.sh 41 minutes ago Up 41 minutes (healthy) 0.0.0.0:8080->8888/tcp JupyterLab

a7a34fd8367b docker.io/jupyter/tensorflow-notebook:latest start-notebook.sh 11 minutes ago Up 11 minutes (healthy) 0.0.0.0:8081->8888/tcp JupyterLabTF

PS D:\contenedores\jupyter\_lab> podman image ls

REPOSITORY	IMAGE ID	TAG	CREATED	SIZE
docker.io/jupyter/datascience-notebook	9f581a022716	latest	4 days ago	6.09 GB
docker.io/jupyter/tensorflow-notebook	200f6a8a92af	latest	4 days ago	5.86 GB

PS D:\contenedores\jupyter\_lab> podman network ls

NETWORK ID	NAME	DRIVER
2f259bab93aa	podman	bridge

127.0.0.1:8081/lab

File Edit View Run Kernel Tabs Settings Help

Filter files by name

work 4 days ago

Usage:

pip command: [options]

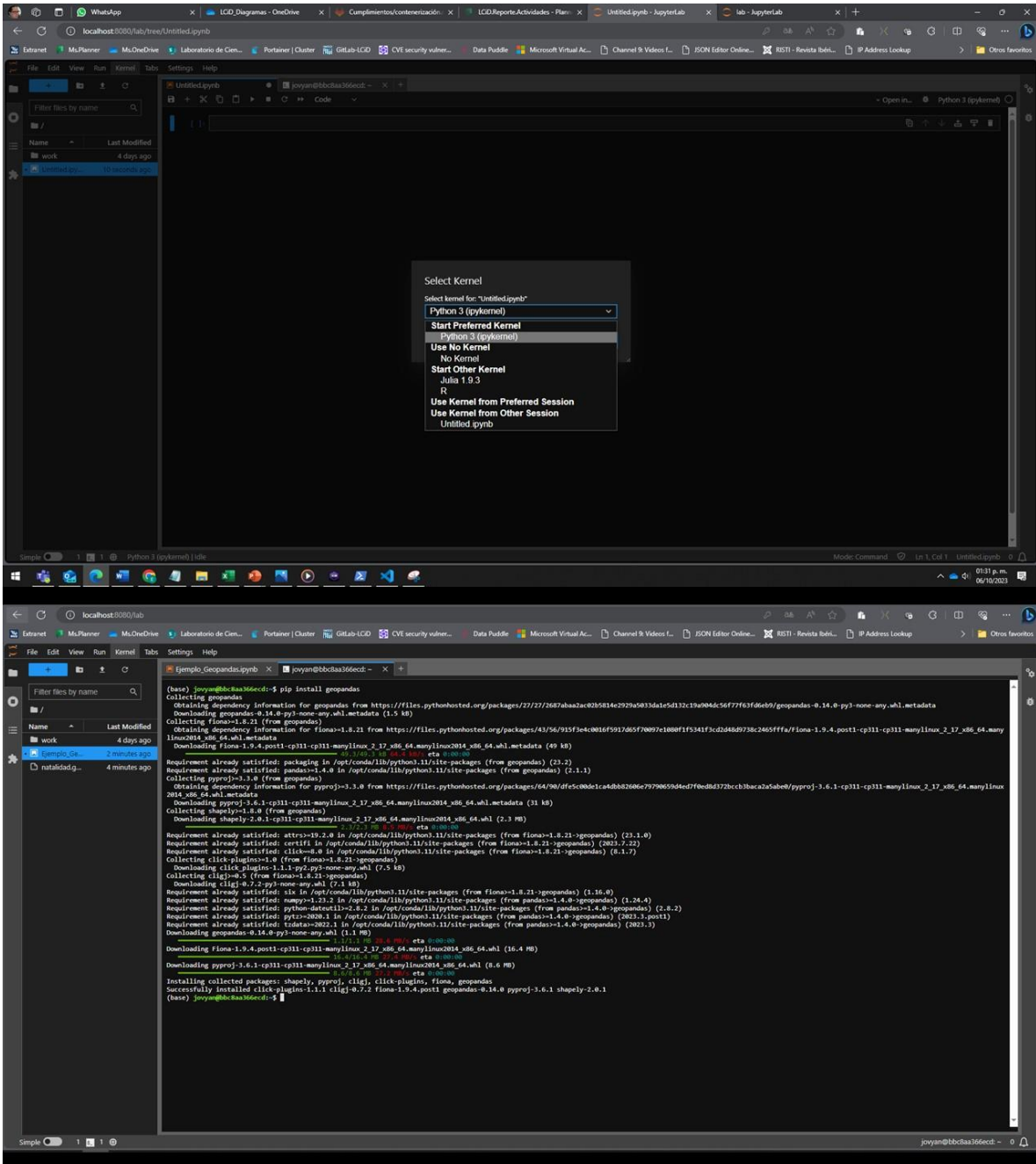
Commands:

- install: Install packages.
- download: Download packages.
- uninstall: Uninstall packages.
- freeze: Output installed packages in requirements format.
- list: List installed packages.
- show: Show information about installed packages.
- check: Verify installed packages have compatible dependencies.
- config: Manage local and global configuration.
- search: Search PyPI for packages.
- cache: Inspect and manage pip's wheel cache.
- index: Inspect information available from package indexes.
- wheel: Build wheels from your requirements.
- hash: Compute hashes of package archives.
- completion: A helper command used for command completion.
- debug: Show information useful for debugging.
- help: Show help for commands.

General Options:

- h, --help: Show help.
- debug: Let unhandled exceptions propagate outside the main subroutine, instead of logging them to stderr.
- isolated: Run pip in an isolated mode, ignoring environment variables and user configuration.
- require-virtualenv: Allow pip to only run in a virtual environment; exit with an error otherwise.
- python python: Run pip with the specified Python interpreter.
- v, --verbose: Give more output. Option is additive, and can be used up to 3 times.
- V, --version: Show version and exit.
- q, --quiet: Give less output. Option is additive, and can be used up to 3 times (corresponding to WARNING, ERROR, and CRITICAL logging levels).
- log path: Path to a verbose appending log.
- no-input: Disable prompting for input.
- keyring-provider keyring\_provider: Enable the credential lookup via the keyring library if user input is allowed. Specify which mechanism to use [disabled, import, subprocess]. (default: disabled)
- proxy proxy: Specify a proxy in the form scheme://[user:password]@proxy.server:port.
- retries retries: Maximum number of retries each connection should attempt (default 5 times).
- timeout timeout: Set the socket timeout (default 15 seconds).
- exists-action action: Default action when a path already exists: (s)witch, (i)gnore, (u)pdate, (a)bort.
- trusted-host host: Mark this host as trusted, even though it does not have valid or any HTTPS.
- cert path: Path to PEM-encoded CA certificate bundle. If provided, overrides the default. See 'SSL Certificate Verification' in pip documentation for more information.
- client-cert path: Path to SSL client certificate, a single file containing the private key and the certificate in PEM format.
- cache-dir dir: Store the cache data in dir.
- no-cache-dir: Disable the cache.
- disable-pip-version-check: Don't periodically check PyPI to determine whether a new version of pip is available for download. Implied with --no-index.
- no-color: Suppress colored output.
- no-python-version-warning: Silence deprecation warnings for upcoming unsupported Python versions.
- use-feature feature: Enable new functionality, that may be backward incompatible.
- use-deprecated feature: Enable deprecated functionality, that will be removed in the future.

(base) jovyan@a7a34fd8367b:~\$ pip





Localhost:8080/lab/tree/Ejemplo\_Geopandas.py

File Edit View Run Kernel Tabs Settings Help

Filter files by name

File Explorer: /  
- Ejemplo\_Geopandas.py  
- natalidad\_geopson

Code Editor: Ejemplo\_Geopandas.py

```
[4]: import geopandas as gpd
import matplotlib.pyplot as plt
# Cargar la capa vectorial
natalidad = "natalidad_geopson"
map_data = gpd.read_file(natalidad)

# Control del tamaño de la figura del mapa
fig, ax = plt.subplots(figsize=(10, 10))

# Control del título y los ejes
ax.set_title('Datos muestra [Natalidad por Provincias en España, año 2018]',
            pad = 10,
            fontdict={'fontsize':15, 'color': 'darkblue'})
ax.set_xlabel('Longitud')
ax.set_ylabel('Latitud')

# Mostrar el mapa finalizado
map_data.plot(column="NAT2018", cmap="plasma", ax=ax, border=1)
# Mostrar el contenido del archivo geopson
map_data.html()
```

Output:

	NAME_1	NAME_2	CC_2	NAT2018	geometry
0	Andalucía	Almería	04	10.48	MULTIPOLYGON (((-3.03042 35.94236, -3.03042 35...
1	Andalucía	Cádiz	11	6.11	MULTIPOLYGON (((-6.21958 36.38110, -6.21958 36...
2	Andalucía	Córdoba	14	7.90	MULTIPOLYGON (((-5.04854 37.83690, -5.04854 37...
3	Andalucía	Granada	18	7.78	MULTIPOLYGON (((-3.35014 36.72952, -3.35014 36...
4	Andalucía	Huelva	21	7.57	MULTIPOLYGON (((-6.83648 37.11547, -6.83648 37...

Datos muestra [Natalidad por Provincias en España, año 2018]

Single Python 3 (ipykernel) idle Mode: Command Ln 13, Col 59 Ejemplo\_Geopandas.py

Archivo C:/Users/oswald.diaz/Downloads/Ejemplo\_Geopandas.pdf

Extranet McPlanner McOneDrive Laboratorio de Cien... Portainer Cluster CVE security vulner... Data Puddle Microsoft Virtual Ac... Channel 9 Videos L... JSON Editor Online... RSTI - Revista Béli... IP Address Lookup Otros favoritos

1 de 2

Ejemplo\_Geopandas

October 6, 2023

```
[5]: import geopandas as gpd
import matplotlib.pyplot as plt
# Cargar la capa vectorial
natalidad = "natalidad_geopson"
map_data = gpd.read_file(natalidad)

# Control del tamaño de la figura del mapa
fig, ax = plt.subplots(figsize=(10, 10))

# Control del título y los ejes
ax.set_title('Datos muestra [Natalidad por Provincias en España, año 2018]',
            pad = 10,
            fontdict={'fontsize':15, 'color': 'darkblue'})
ax.set_xlabel('Longitud')
ax.set_ylabel('Latitud')

# Mostrar el mapa finalizado
map_data.plot(column="NAT2018", cmap="plasma", ax=ax, border=1)
# Mostrar el contenido del archivo geopson
map_data.html()
```

Output:

	NAME_1	NAME_2	CC_2	NAT2018	geometry
0	Andalucía	Almería	04	10.48	MULTIPOLYGON (((-3.03042 35.94236, -3.03042 35...
1	Andalucía	Cádiz	11	6.11	MULTIPOLYGON (((-6.21958 36.38110, -6.21958 36...
2	Andalucía	Córdoba	14	7.90	MULTIPOLYGON (((-5.04854 37.83690, -5.04854 37...
3	Andalucía	Granada	18	7.78	MULTIPOLYGON (((-3.35014 36.72952, -3.35014 36...
4	Andalucía	Huelva	21	7.57	MULTIPOLYGON (((-6.83648 37.11547, -6.83648 37...

Datos muestra [Natalidad por Provincias en España, año 2018]



Universidad  
de la Ciudad de  
Aguascalientes

https://github.com/edgarOswaldoDiaz/data\_engineer/tree/main

edgarOswaldoDiaz / data\_engineer

main 1 branch 0 tags

Go to file Add file + Code +

edgarOswaldoDiaz Update README.md cal0514 · now 6 commits

- Ejemplo\_geopandas.ipynb Ejemplo de Geopandas 5 minutos ago
- README.md Update README.md now
- natalidad.geosjon DataSet para ejemplo de Geopandas 6 minutos ago

README.md

### Data engineer (Plataforma para ingeniería de datos masivos)

Contenedor basado en Jupyter Lab

```
podman pull jupyter/datascience-notebook
```

• Download Podman Desktop <https://podman-desktop.io/>

**About**  
Data engineer (Plataforma para ingeniería de datos masivos)  
Readme  
Activity  
0 stars  
1 watching  
0 forks

**Releases**  
No releases published  
[Create a new release](#)

**Packages**  
No packages published  
[Publish your first package](#)

**Languages**  
Jupyter Notebook 100.0%

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)