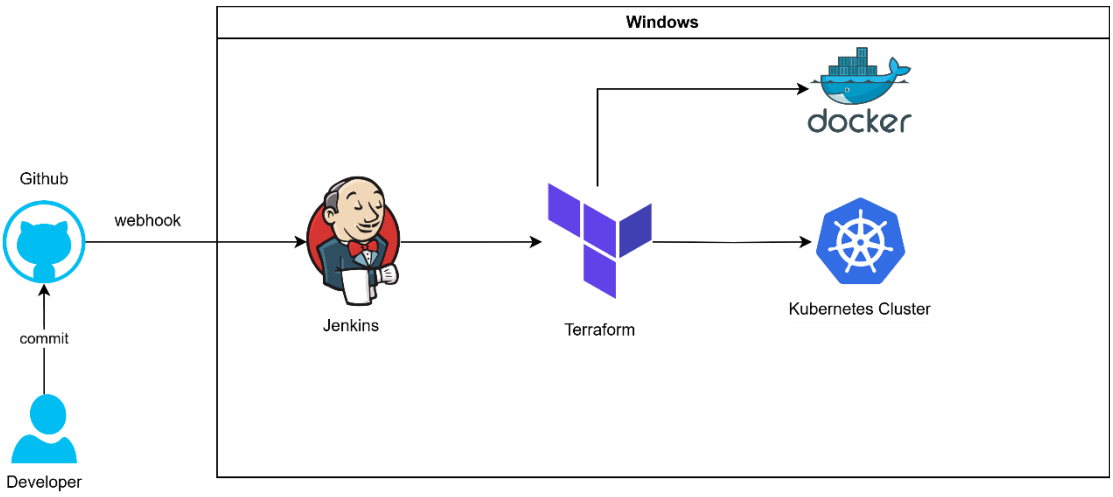
Daniel Velasquez

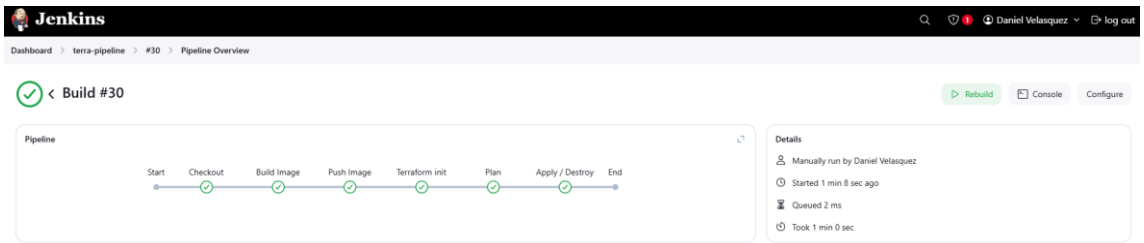
Flujo y diseño CI/CD despliegue de aplicativo Python a Kubernetes

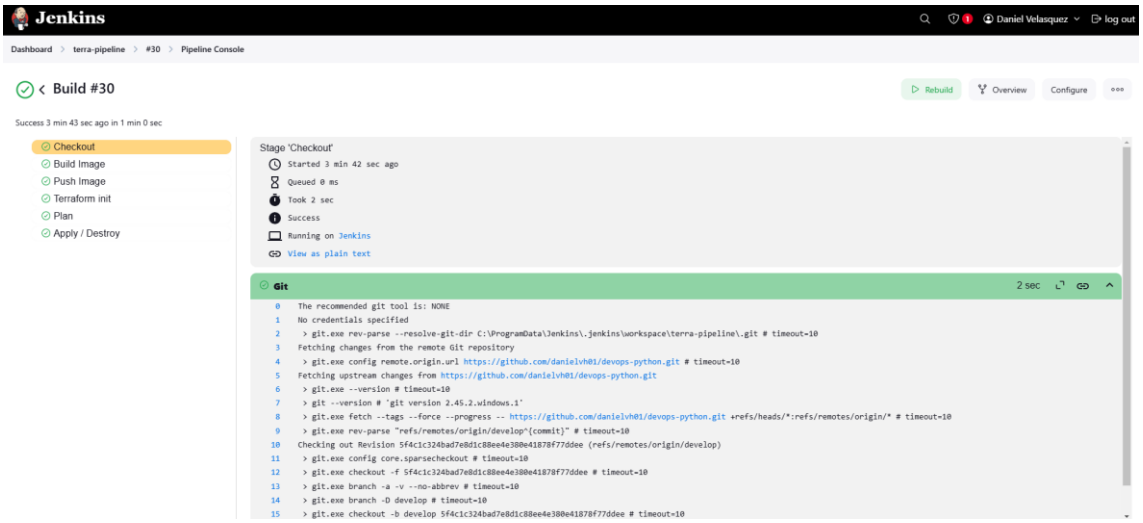


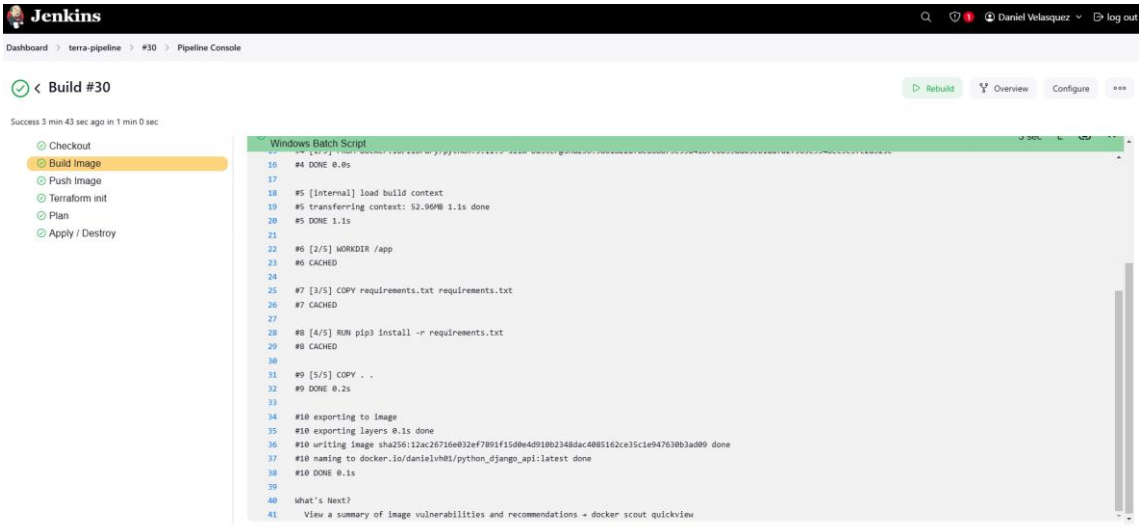
Explicación:

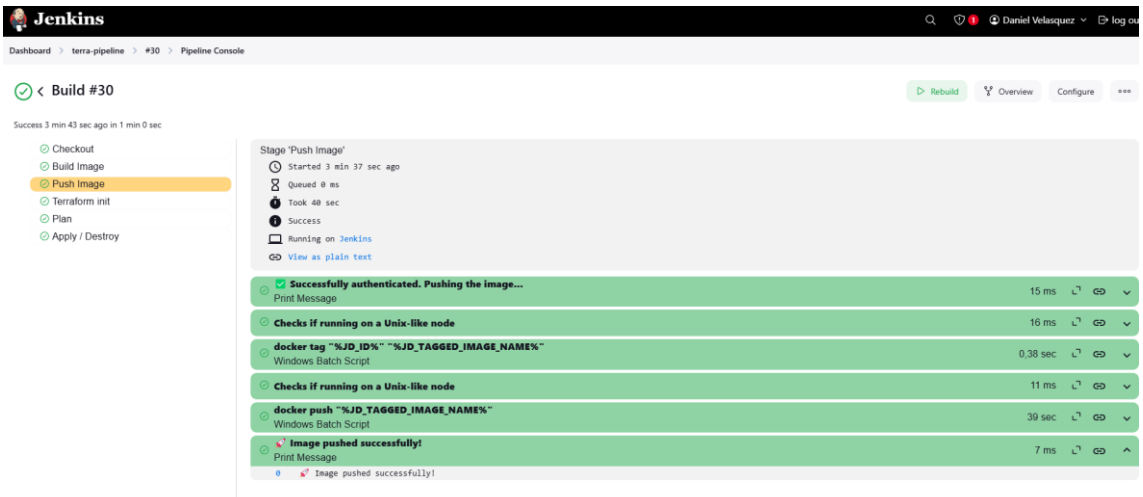
1. developer realiza un push a un repositorio github.
2. Github dispara a través de un webhook, el pipeline a través de Jenkins.
3. Jenkins a través de “stages” arma la imagen en Docker, realiza push a dockerhub y ejecuta terraform para el despliegue del aplicativo al cluster de Kubernetes.
4. Terraform se encarga de la creación de servicios, namespaces, mapear el provider (KUbernetes cluster de Docker Desktop). Se realiza terraform init, plan y apply.

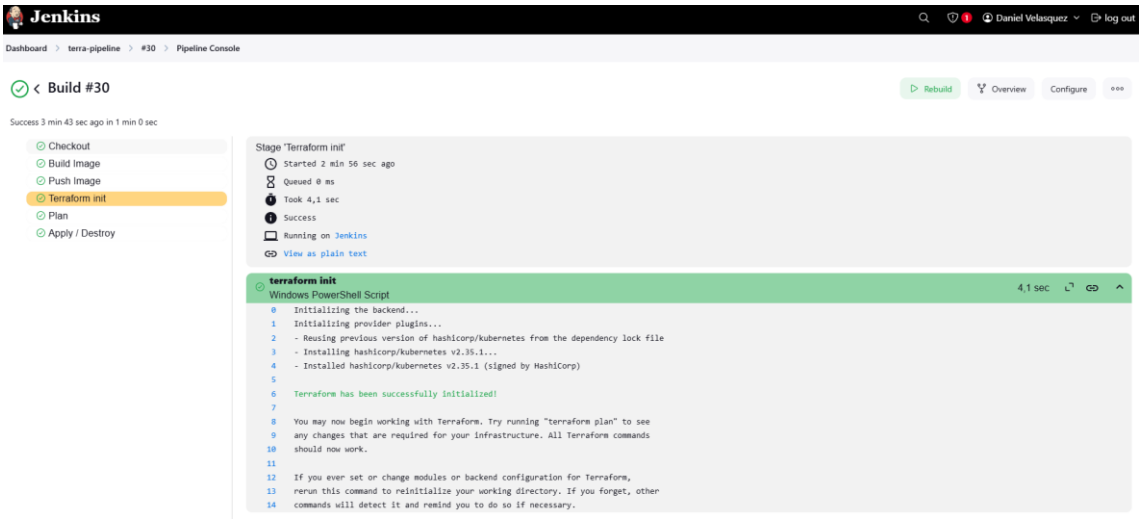
Pruebas de ejecución paso a paso de pipeline:



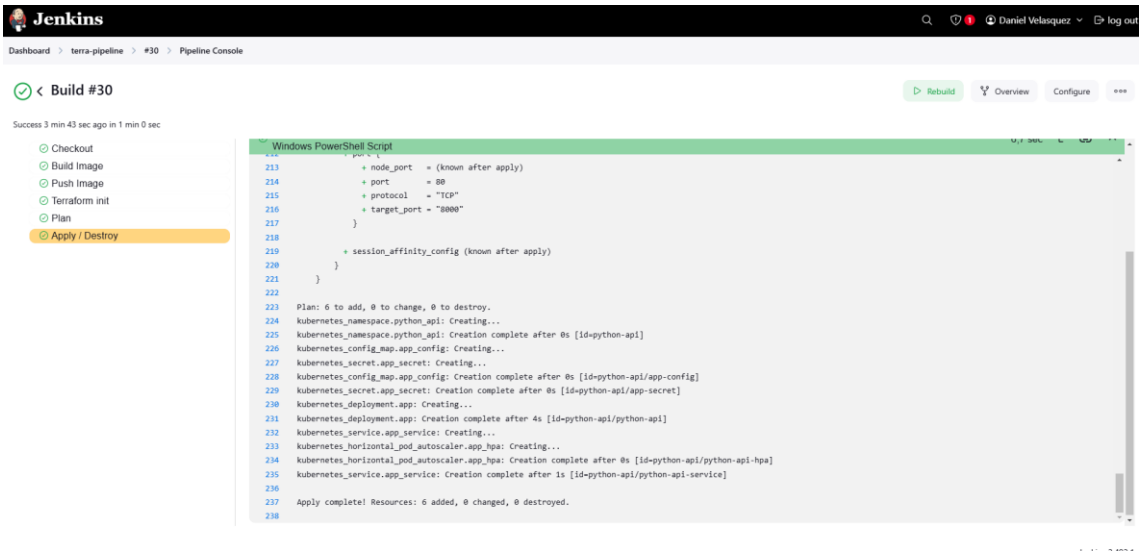

Jenkins
Daniel Velasquez
log out
Dashboard
terra-pipeline
#30
Pipeline Console
Build #30
Rebuild
Overview
Configure
Success 3 min 43 sec ago in 1 min 0 sec
Checkout
Build Image
Push Image
Terraform init
Plan
Apply / Destroy
Stage 'Checkout'
Started 3 min 42 sec ago
Queued 0 ms
Took 2 sec
Success
Running on Jenkins
View as plain text
Git
2 sec
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\terra-pipeline\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/danielvh01/devops-python.git # timeout=10
Fetching upstream changes from https://github.com/danielvh01/devops-python.git
> git.exe --version # timeout=10
> git --version # 'git version 2.45.2.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/danielvh01/devops-python.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/develop^{commit}" # timeout=10
Checking out Revision 5f4c1c324bad7e8d1c88ee4e380e41878f77ddee (refs/remotes/origin/develop)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 5f4c1c324bad7e8d1c88ee4e380e41878f77ddee # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D develop # timeout=10
> git.exe checkout -b develop 5f4c1c324bad7e8d1c88ee4e380e41878f77ddee # timeout=10


Jenkins
Daniel Velasquez
log out
Dashboard
terra-pipeline
#30
Pipeline Console
Build #30
Rebuild
Overview
Configure
Success 3 min 43 sec ago in 1 min 0 sec
Checkout
Build Image
Push Image
Terraform init
Plan
Apply / Destroy
Windows Batch Script
#4 DONE 0.0s
#5 [internal] load build context
#5 transferring context: 52.96MB 1.1s done
#5 DONE 1.1s
#6 [2/5] WORKDIR /app
#6 CACHED
#7 [3/5] COPY requirements.txt requirements.txt
#7 CACHED
#8 [4/5] RUN pip3 install -r requirements.txt
#8 CACHED
#9 [5/5] COPY . .
#9 DONE 0.2s
#10 exporting to image
#10 exporting layers 0.1s done
#10 writing image sha256:12ac26716e032ef7891f15d0e4d910b2348dac4085162ce35c1e947630b3ad09 done
#10 naming to docker.io/danielvh01/python_django_api:latest done
#10 DONE 0.1s
What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview

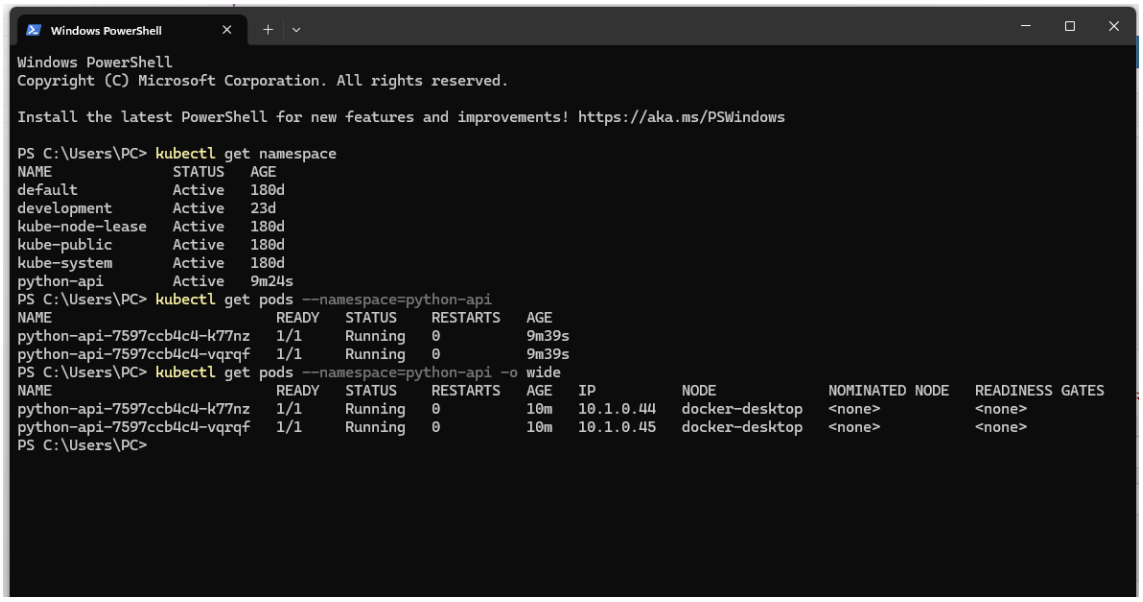

Jenkins
Daniel Velasquez
Dashboard
terra-pipeline
#30
Pipeline Console
Build #30
Rebuild
Overview
Configure
Success 3 min 43 sec ago in 1 min 0 sec
Checkout
Build Image
Push Image
Terraform init
Plan
Apply / Destroy
Stage 'Push Image'
Started 3 min 37 sec ago
Queued 0 ms
Took 40 sec
Success
Running on Jenkins
View as plain text
Successfully authenticated. Pushing the image...
Print Message
15 ms
Checks if running on a Unix-like node
16 ms
docker tag "%JD_ID%" "%JD_TAGGED_IMAGE_NAME%"
Windows Batch Script
0,38 sec
Checks if running on a Unix-like node
11 ms
docker push "%JD_TAGGED_IMAGE_NAME%"
Windows Batch Script
39 sec
Image pushed successfully!
Print Message
7 ms
Image pushed successfully!

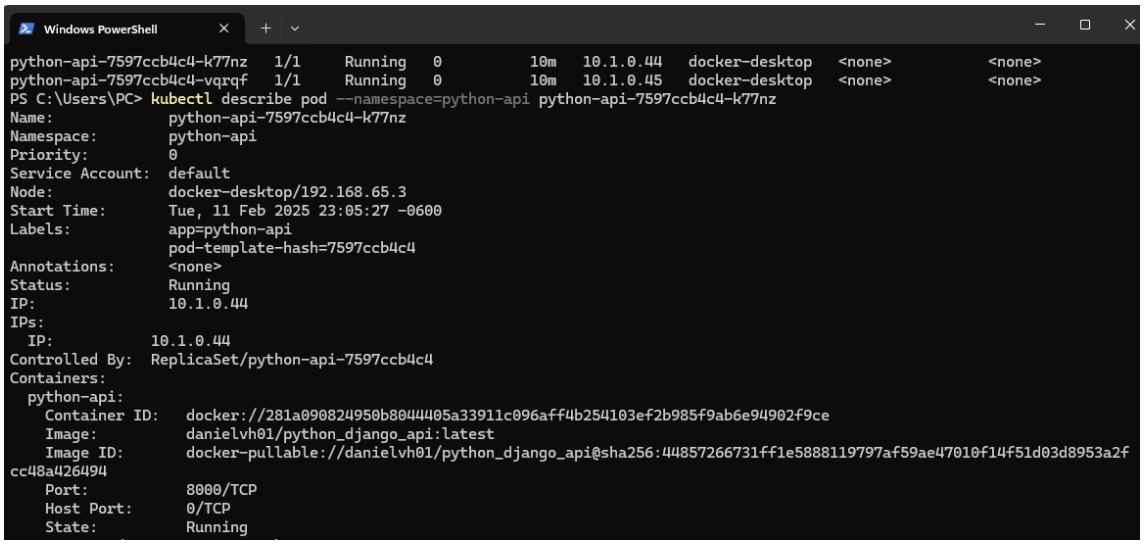

Jenkins
Daniel Velasquez
log out
Dashboard
terra-pipeline
#30
Pipeline Console
Build #30
Rebuild
Overview
Configure
Success 3 min 43 sec ago in 1 min 0 sec
Checkout
Build Image
Push Image
Terraform init
Plan
Apply / Destroy
Stage "Terraform init"
Started 2 min 56 sec ago
Queued 0 ms
Took 4,1 sec
Success
Running on Jenkins
View as plain text
terraform init
Windows PowerShell Script
4,1 sec
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/kubernetes from the dependency lock file
- Installing hashicorp/kubernetes v2.35.1...
- Installed hashicorp/kubernetes v2.35.1 (signed by HashiCorp)
Terraform has been successfully initialized!
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.


Jenkins
Daniel Velasquez
log out
Dashboard
terra-pipeline
#30
Pipeline Console
Build #30
Rebuild
Overview
Configure
Success 3 min 43 sec ago in 1 min 0 sec
Checkout
Build Image
Push Image
Terraform init
Plan
Apply / Destroy
Windows PowerShell Script
+ publish_not_ready_addresses = false
+ selector = {
+ "app" = "python-api"
}
+ session_affinity = "None"
+ type = "LoadBalancer"
+ port {
+ node_port = (known after apply)
+ port = 80
+ protocol = "TCP"
+ target_port = "8000"
}
+ session_affinity_config (known after apply)
}
}
Plan: 6 to add, 0 to change, 0 to destroy.
Saved the plan to: tfplan
To perform exactly these actions, run the following command to apply:
terraform apply "tfplan"

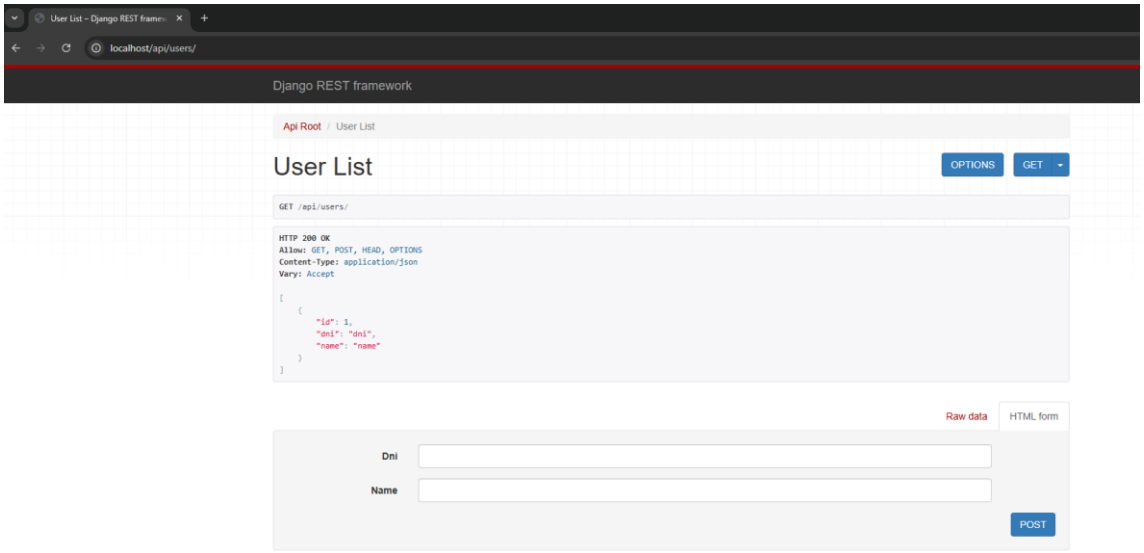


Validación de pods en kubectl





Prueba de uso de aplicativo Python hosteado en Kubernetes mediante load balancer expuesto en puerto 80.



Repositorio publico de la solución:

https://github.com/danielvh01/devops-python.git