

Documentation du Projet CI/CD avec Jenkins sur GCP

Introduction

Ce document décrit les étapes nécessaires pour créer et configurer un projet CI/CD avec Jenkins sur Google Cloud Platform (GCP). Il explique comment intégrer des Webhooks GitHub pour automatiser le processus d'intégration et de livraison continues (CI/CD). Grâce à cette configuration, chaque fois qu'un nouveau commit est poussé sur le dépôt GitHub, le pipeline Jenkins se déclenche automatiquement pour construire, tester et déployer le code.

Étapes de Configuration

1. Création du Projet sur GitHub

Nous avons créé un projet sur GitHub nommé `ci-cd-pipeline-avec-jenkins-sur-gcp`.

2. Clonage du Projet

Clonez le projet sur votre ordinateur en utilisant la commande suivante :

```
git clone https://github.com/Serchgalarza/ci-cd-pipeline-avec-jenkins-sur-gcp.git
```

Accédez ensuite au répertoire du projet :

```
cd ci-cd-pipeline-avec-jenkins-sur-gcp
```

3. Création du Jenkinsfile

Créez un fichier Jenkinsfile avec l'éditeur de texte `nano` :

```
nano Jenkinsfile
```

Ajoutez le contenu suivant au Jenkinsfile :

```
pipeline {
  agent any
  stages {
    stage('Build') {
      steps {
        echo 'Building...'
      }
    }
    stage('Test') {
```

```

        steps {
            echo 'Testing...'
        }
    }
    stage('Deploy') {
        steps {
            echo 'Deploying...'
        }
    }
}
}

```

Ce Jenkinsfile définit les étapes de base pour le pipeline CI/CD, incluant les phases de construction, de test et de déploiement.

Pour sauvegarder les modifications dans `nano`, utilisez la combinaison de touches `Ctrl + O`, puis appuyez sur `Enter`. Pour fermer `nano`, utilisez `Ctrl + X`.

4. Commit et Push des Modifications

4.1. Commit Direct à la Branche `main`

Si vous choisissez de faire des modifications directement sur la branche `main`, utilisez les commandes suivantes :

```

git add .
git commit -m "Ajouter le Jenkinsfile pour configurer le pipeline CI/CD" -m
"Ce commit ajoute le Jenkinsfile qui définit les étapes de construction, de
test et de déploiement pour le pipeline CI/CD."
git push origin main

```

4.2. Utiliser une Branche pour les Modifications

Pour éviter des problèmes potentiels et suivre les bonnes pratiques, créez une nouvelle branche pour les modifications :

1. Créer une Nouvelle Branche :

```
git checkout -b feature/ajouter-jenkinsfile
```

2. Ajouter et Valider les Changements :

```

git add Jenkinsfile
git commit -m "Ajouter le Jenkinsfile pour configurer le pipeline
CI/CD"
git push origin feature/ajouter-jenkinsfile

```

3. Créer une Pull Request (PR) :

Accédez à GitHub et créez une Pull Request pour fusionner les changements de votre branche `feature/ajouter-jenkinsfile` dans `main`.

1. Allez sur la page du dépôt GitHub.
 2. Cliquez sur l'onglet "Pull Requests".
 3. Cliquez sur "New Pull Request".
 4. Sélectionnez `feature/ajouter-jenkinsfile` comme branche source et `main` comme branche de destination.
 5. Ajoutez une description si nécessaire et soumettez la Pull Request.
4. **Révision et Fusion :**

Une fois la Pull Request soumise, elle doit être révisée et approuvée par les membres de l'équipe. Après approbation, fusionnez la Pull Request avec la branche `main`.


5. Supprimer la Branche (optionnel) :


Après fusion, vous pouvez supprimer la branche de fonctionnalité si elle n'est plus nécessaire :




```
git branch -d feature/ajouter-jenkinsfile
git push origin --delete feature/ajouter-jenkinsfile
```

5. Validation sur GitHub


Une fois les modifications ajoutées et la branche créée, vérifiez les informations sur GitHub. Vous pouvez également basculer entre les branches pour confirmer que les changements ont été appliqués correctement.




 **ci-cd-pipeline-avec-jenkins-sur-gcp** Public Pin Unwatch 1

 **feature/ajouter-jenkinsfile** had recent pushes 42 seconds ago Compare & pull request

 **feature/ajouter-jenkinsf...**  **2 Branches**  **0 Tags** t Add file Code

This branch is **1 commit ahead of** `main` . Contribute

 **Serchgalarza** Ajouter le Jenkinsfile pour configurer le pipeline CI/CD 08de1f1 · 11 minutes ago 2 Commits

 <code>.gitignore</code>	Initial commit	41 minutes ago
 <code>Jenkinsfile</code>	Ajouter le Jenkinsfile pour configurer le pipeline CI/CD	11 minutes ago
 <code>README.md</code>	Initial commit	41 minutes ago

Déploiement et Configuration de Jenkins sur GCP

1. Lancer l'Instance sur Google Cloud Platform (GCP)

Pour notre démonstration, nous avons créé une instance de type micro avec Ubuntu comme système d'exploitation.

- Accéder à Google Cloud Console :**
 - Connectez-vous à Google Cloud Console.
- Créer une Nouvelle Instance :**
 - Allez dans le menu "Compute Engine", puis sélectionnez "VM instances".
 - Cliquez sur "Create Instance" pour lancer une nouvelle instance.
 - Sélectionnez un type de machine micro (par exemple, e2-micro).
 - Choisissez Ubuntu comme système d'exploitation dans les options de configuration de la machine.
- Configurer les Règles de Pare-feu (VPC) :**
 - Dans les paramètres de l'instance, configurez les règles de pare-feu pour permettre le trafic sur le port 8080, nécessaire pour accéder à Jenkins.
 - Vous pouvez le faire en cochant les options "Allow HTTP traffic" et "Allow HTTPS traffic", et en ajoutant une règle personnalisée pour le port 8080 si ce n'est pas déjà fait.
- Lancer l'Instance :**
 - Cliquez sur "Create" pour déployer l'instance.

INSTANCIAS

OBSERVABILIDAD

PROGRAMAS DE LAS INSTANCIAS

Instancias de VM

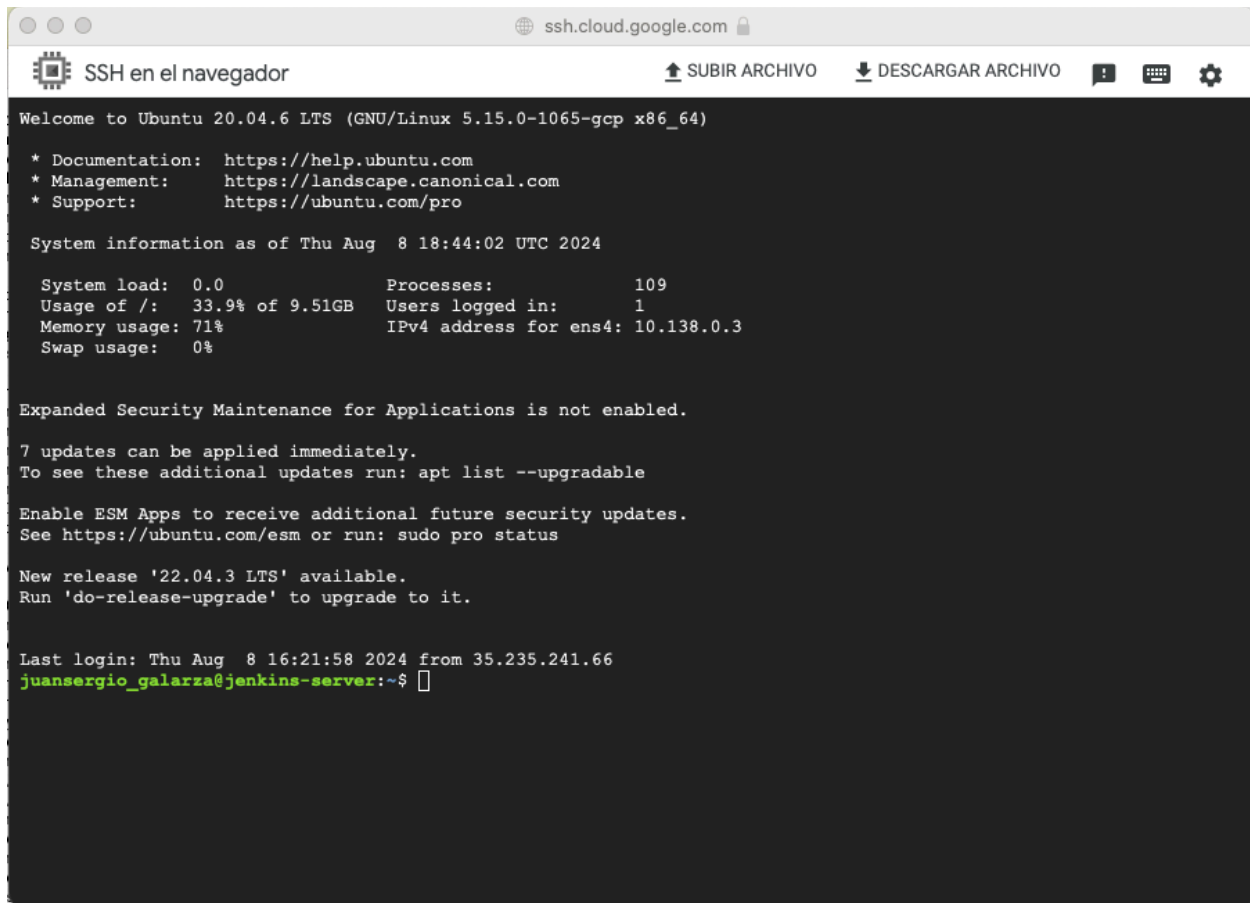
Filtro

Ingresar el nombre o el valor de la propiedad

<input type="checkbox"/> Estado	Nombre ↑	Zona	Recomendaciones	En uso por	IP interna	IP externa	Conectar
<input checked="" type="checkbox"/>	jenkins-server	us-west1-a			10.138.0.3 (nic0)	34.19.104.169 ↗ (nic0)	SSH ▾

2. Accéder à la Configuration de l'Instance via SSH

- Ouvrir la Console SSH :**
 - Dans Google Cloud Console, allez sur "Compute Engine" -> "VM instances".
 - Cliquez sur le bouton "SSH" pour ouvrir une session terminal directement dans le navigateur ou utilisez un client SSH pour vous connecter à l'instance.



```
ssh.cloud.google.com
SSH en el navegador SUBIR ARCHIVO DESCARGAR ARCHIVO
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1065-gcp x86_64)
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro
System information as of Thu Aug  8 18:44:02 UTC 2024
System load:  0.0      Processes:            109
Usage of /:   33.9% of 9.51GB   Users logged in:     1
Memory usage: 71%      IPv4 address for ens4: 10.138.0.3
Swap usage:   0%
Expanded Security Maintenance for Applications is not enabled.
7 updates can be applied immediately.
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
Last login: Thu Aug  8 16:21:58 2024 from 35.235.241.66
juansergio_galarza@jenkins-server:~$
```

3. Installer et Configurer Jenkins

1. **Mettre à Jour le Système** : Assurez-vous que le système est à jour :

```
sudo apt update
sudo apt upgrade -y
```

2. **Installer Java (si nécessaire)** : Jenkins nécessite Java. Installez OpenJDK :

```
sudo apt install openjdk-11-jdk -y
```

3. **Ajouter le Dépôt Jenkins et Installer Jenkins** : Ajoutez le dépôt Jenkins et installez Jenkins :

```
wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-
key add -
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary >
/etc/apt/sources.list.d/jenkins.list'
sudo apt update
sudo apt install jenkins -y
```

4. **Démarrer Jenkins** : Assurez-vous que Jenkins est démarré :

1. **Accéder à Jenkins** : Connectez-vous à votre instance Jenkins via l'URL `http://34.19.104.169:8080/`.
2. **Créer un Nouveau Job** :
 - Cliquez sur le symbole + ou "New Item" pour créer un nouveau job.

- Entrez le nom de votre projet, par exemple : ci-cd-pipeline-avec-jenkins-sur-gcp.
- Sélectionnez "Pipeline" comme type de job.
- Cliquez sur "OK" pour créer le job.

Jenkins 🔍 búsqueda (%+K) ? 🔔 1 🛡️ 1 👤 Juan Sergio Galarza 🚪 Desconectar

Panel de Control > Todo > Nuevo Tarea

Nuevo Tarea

Enter an item name

ci-cd-pipeline-avec-jenkins-sur-gcp

Select an item type

- Crear un proyecto de estilo libre**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**
Gestiona actividades de larga duración que pueden abarcar varios agentes de construcción. Apropiado para construir pipelines (conocidas anteriormente como workflows) y/o para la organización de actividades complejas que no se pueden articular fácilmente con tareas de tipo freestyle.
- Crear un proyecto multi-configuración**
Adecuado para proyectos que requieran un gran número de configuraciones diferentes, como testear en múltiples entornos, ejecutar sobre plataformas concretas, etc.
- Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

3. Configurer le Job :

- **Description** : Ajoutez une petite description pour votre job si nécessaire.
- **Build Triggers** : Sous la section "Build Triggers", sélectionnez l'option "GitHub hook trigger for GITScm polling". Cela permet à Jenkins de déclencher le job automatiquement à chaque commit sur GitHub.
- **Pipeline** : Sous la section "Pipeline", choisissez l'option "Pipeline script from SCM".
- **SCM** : Sélectionnez "Git" comme système de gestion de versions.
 - **URL du Repository** : Entrez l'URL de votre repository GitHub :

`https://github.com/Serchgalzarza/ci-cd-pipeline-avec-jenkins-sur-gcp.git`

- **Branch** : Pour cet exemple, utilisez la branche `feature/ajouter-jenkinsfile`. Vous pouvez également utiliser `main` ou `master` si c'est la branche principale.
- Cliquez sur "Save" pour enregistrer les configurations du job.

The screenshot shows the Jenkins 'Configure' page for a job named 'ci-cd-pipeline-avec-jenkins-sur-gcp'. The page is divided into several sections:

- Configure** (Left sidebar):
 - General (Selected)
 - Advanced Project Options
 - Pipeline
- Configuration** (Main area):
 - General**:
 - ☐ Construir tras otros proyectos ?
 - ☐ Ejecutar periódicamente ?
 - ☒ GitHub hook trigger for GITScm polling ?
 - ☐ Consultar repositorio (SCM) ?
 - ☐ Periodo de espera ?
 - ☐ Lanzar ejecuciones remotas (ejem: desde 'scripts') ?
 - Advanced Project Options**:
 - Avanzado ▾
 - Pipeline**:
 - Definition**: Pipeline script from SCM
 - SCM**: Git
 - Repositories**:
 - Repository URL: `https://github.com/Serchgalzarza/ci-cd-pipeline-avec-jenkins-sur-gcp.git`
 - Credentials**

At the bottom, there are two buttons: 'Guardar' (Save) and 'Apply'.

5. Configurer le Webhook sur GitHub

Pour que Jenkins se déclenche automatiquement à chaque commit, vous devez configurer un Webhook dans votre dépôt GitHub :

1. Accéder aux Paramètres du Dépôt :

- Allez sur la page de votre dépôt GitHub : `https://github.com/Serchgalzarza/ci-cd-pipeline-avec-jenkins-sur-gcp`.
 - Cliquez sur l'onglet "Settings".
2. **Ajouter un Webhook :**
- Dans le menu de gauche, sélectionnez "Webhooks".
 - Cliquez sur "Add webhook".
 - **Payload URL :** Entrez l'URL suivante :

`http://34.19.104.169:8080/github-webhook/`
 - **Content type :** Sélectionnez `application/json`.
 - **Which events would you like to trigger this webhook? :** Sélectionnez "Just the push event".
 - Cliquez sur "Add webhook" pour enregistrer le webhook.

Payload URL *

`http://34.19.104.169:8080/github-webhook/`

Content type *

`application/json`

Secret

SSL verification

 By default, we verify SSL certificates when delivering payloads.

☒ **Enable SSL verification** ☐ **Disable (not recommended)**

Which events would you like to trigger this webhook?

- ☒ Just the push event.
- ☐ Send me **everything**.
- ☐ Let me select individual events.

☒ **Active**







We will deliver event details when this hook is triggered.

Add webhook

Conclusion

Une fois ces étapes complètes, votre Jenkins est configuré pour surveiller les commits dans votre dépôt GitHub et déclencher automatiquement le pipeline CI/CD défini dans votre Jenkinsfile. Vous pouvez vérifier le bon fonctionnement en effectuant un commit sur votre dépôt GitHub et en observant si le pipeline Jenkins se déclenche correctement.

Todo +

S	W	Nombre ↓	Último Éxito	Último Fallo	Última Duración	
		ci-cd-pipeline	2 Hor 19 Min #1	N/D	10 Seg	
		ci-cd-pipeline-avec-jenkins-sur-gcp	29 Seg #1	N/D	3 Seg	

Icono: S M L ...