Create a CI Pipeline

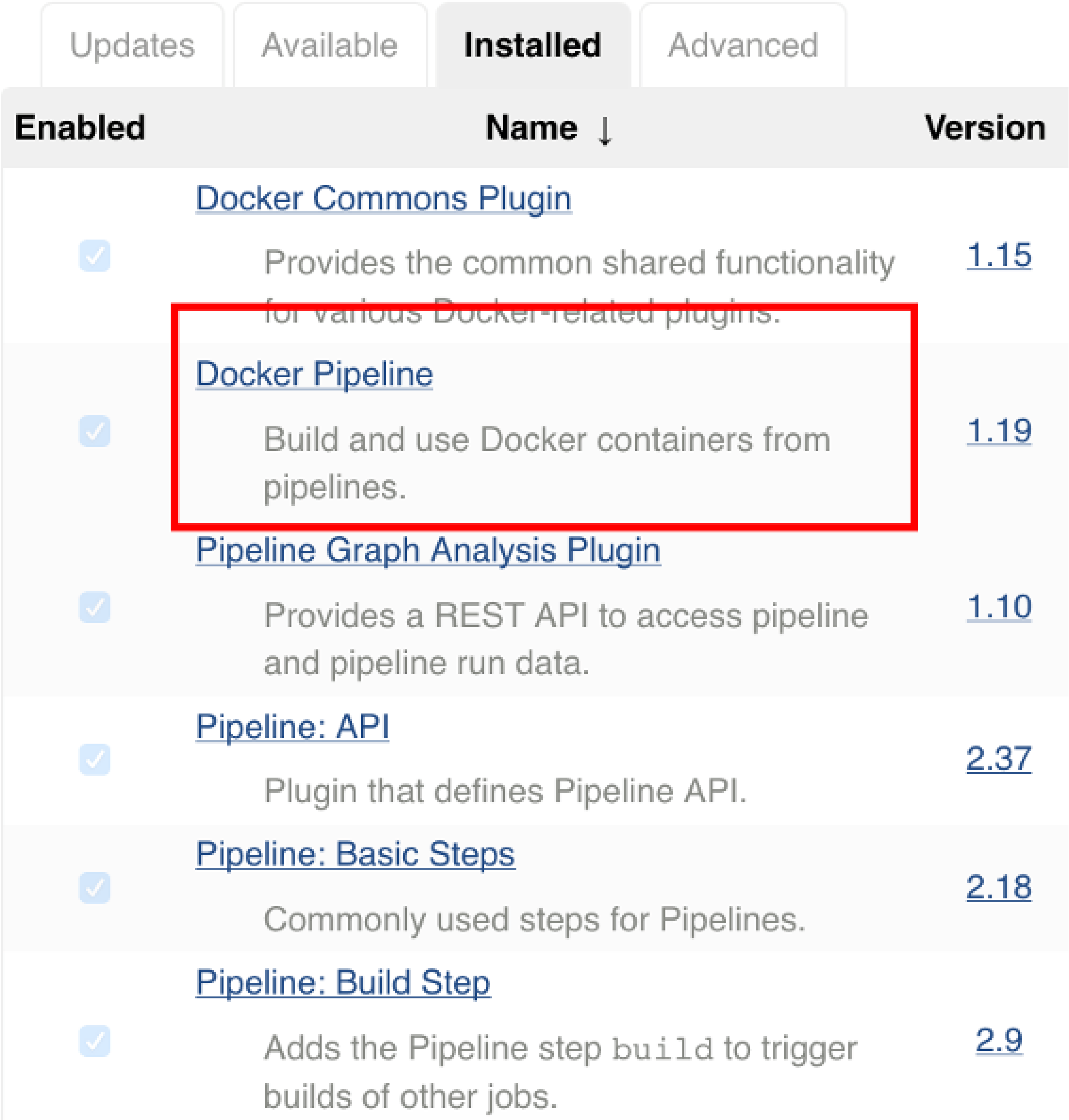
With Jenkins and Kubernetes Engine

Agenda

- Prepare Jenkins
- Configure a Google Cloud Platform service account
- Create and enable a GitHub repository webhook
- Create a Jenkins pipeline project
- Add credentials to Jenkins
- Create simple “Hello World” Project
- Deployment
- Commit, Test and Repeat

Prepare Jenkins

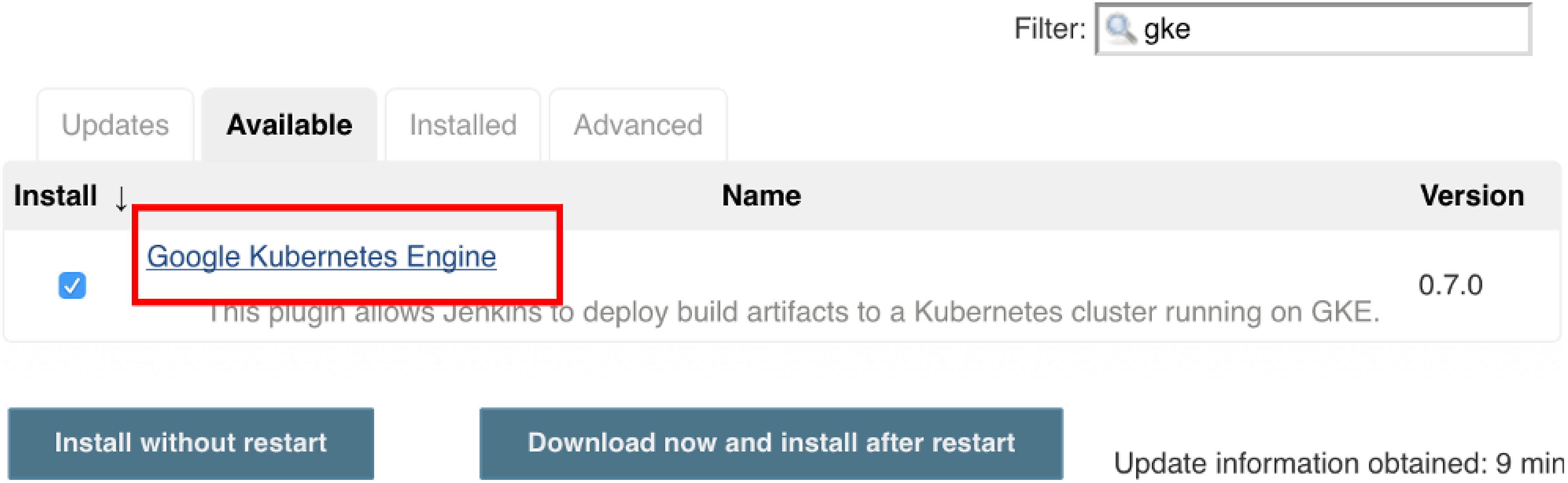
- For installation refer:
  - 3-DevOps in Action - 3. Jenkins.txt
- Navigate to the "Manage Jenkins -> Manage Plugins" page.
- On the resulting page, select the "Installed" tab and confirm that
  - "Docker Pipeline" and "GitHub" plugins are installed



Enabled	Name ↓	Version
☑	Docker Commons Plugin Provides the common shared functionality for various Docker-related plugins.	1.15
☑	Docker Pipeline Build and use Docker containers from pipelines.	1.19
☑	Pipeline Graph Analysis Plugin Provides a REST API to access pipeline and pipeline run data.	1.10
☑	Pipeline: API Plugin that defines Pipeline API.	2.37
☑	Pipeline: Basic Steps Commonly used steps for Pipelines.	2.18
☑	Pipeline: Build Step Adds the Pipeline step `build` to trigger builds of other jobs.	2.9

Prepare Jenkins

- Select the "Available" tab and select the "Google Kubernetes Engine" plugin



Filter: gke

Install ↓	Name	Version
☑	Google Kubernetes Engine This plugin allows Jenkins to deploy build artifacts to a Kubernetes cluster running on GKE.	0.7.0

Update information obtained: 9 min

Configure a Google Cloud Platform service account

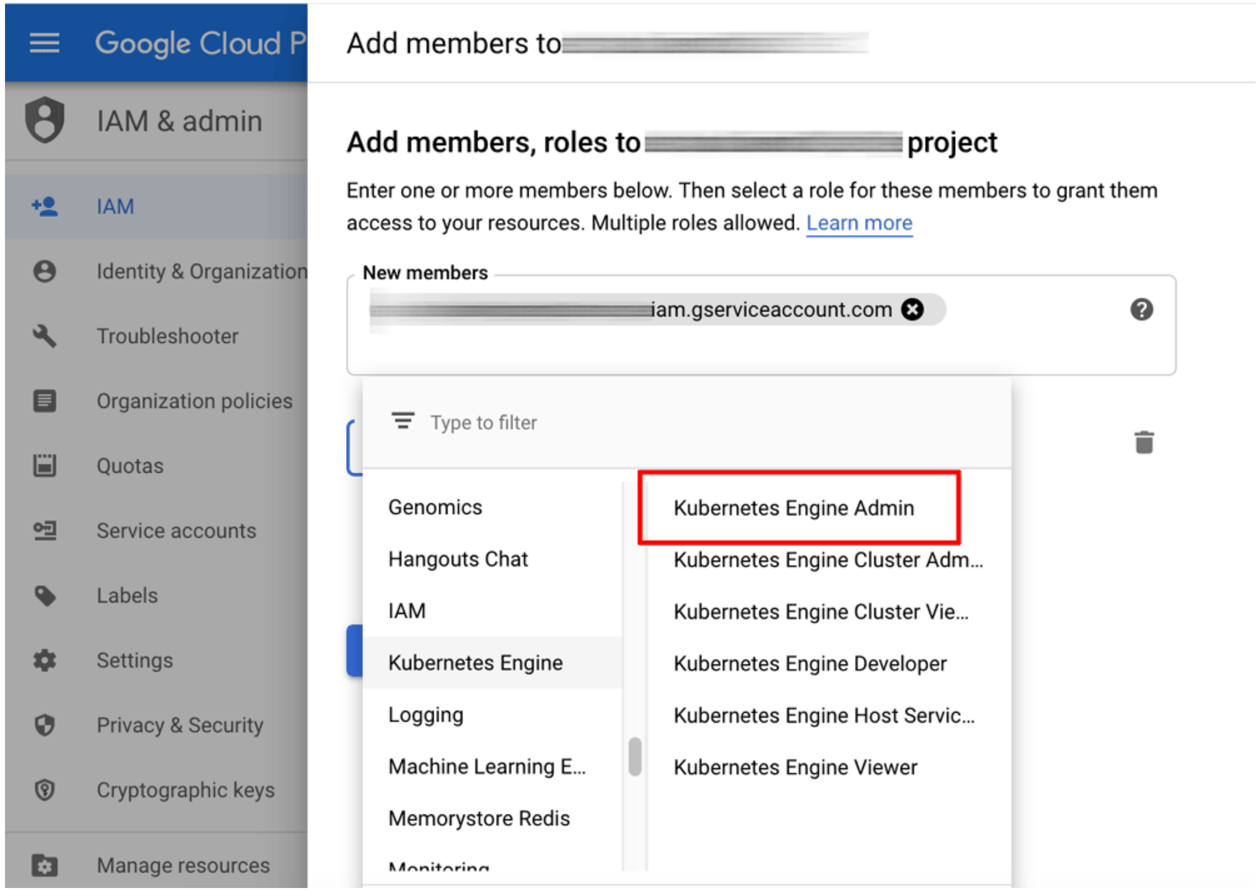
- Log in to Google Cloud Platform and select your project.
- Navigate to the "IAM & admin -> Service accounts" page
- Create a new service account
- Create a new JSON key for the service account
- Download and save this key

Enable the APIs needed by the Jenkins GKE plugin

- Navigate to the "APIs & services -> Library" page.
- Search for and enable each of the following APIs:
  - Compute Engine API
  - Kubernetes Engine API
  - Service Management API
  - Cloud Resource Manager API
- After enabling each API, click the "Manage" button on the API detail page and confirm that the service account created previously has access to the API (or add access if required).

Enable the APIs needed by the Jenkins GKE plugin

- Navigate to the "IAM & admin & IAM" page.
- Click the "Add" button. Select the service account created in the previous step and assign it the "Kubernetes Engine Admin" role.



Add members to [redacted]

Add members, roles to [redacted] project

Enter one or more members below. Then select a role for these members to grant them access to your resources. Multiple roles allowed. Learn more

New members

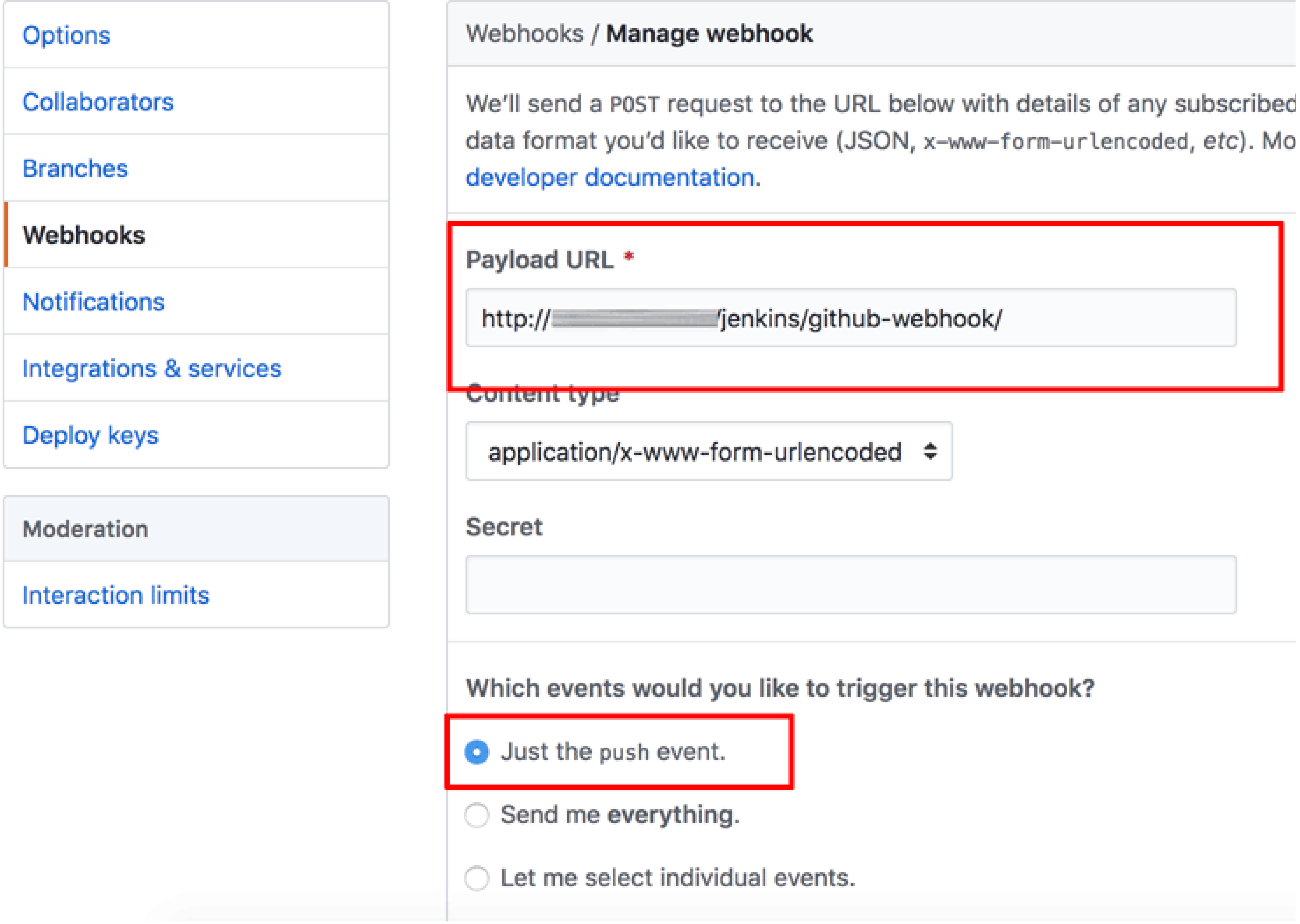
[redacted]iam.gserviceaccount.com

Setup Kubectl

- We need to setup Kubectl on Jenkins server:
  - curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"
  - chmod +x ./kubectl
  - sudo mv ./kubectl /usr/local/bin/kubectl
  - kubectl version --client

Create and enable a GitHub repository webhook

- Log in to GitHub
- Create a new repository
- "Settings" tab at the top
- "Webhooks" sub-menu
- "Payload URL" field, enter
    - http://IP-ADDRESS:8080/github-webhook/



We'll send a POST request to the URL below with details of any subscribed data format you'd like to receive (JSON, x-www-form-urlencoded, *etc*). Mo developer documentation.

Payload URL *

http://[redacted]jenkins/github-webhook/

Content type

application/x-www-form-urlencoded

Secret

Which events would you like to trigger this webhook?

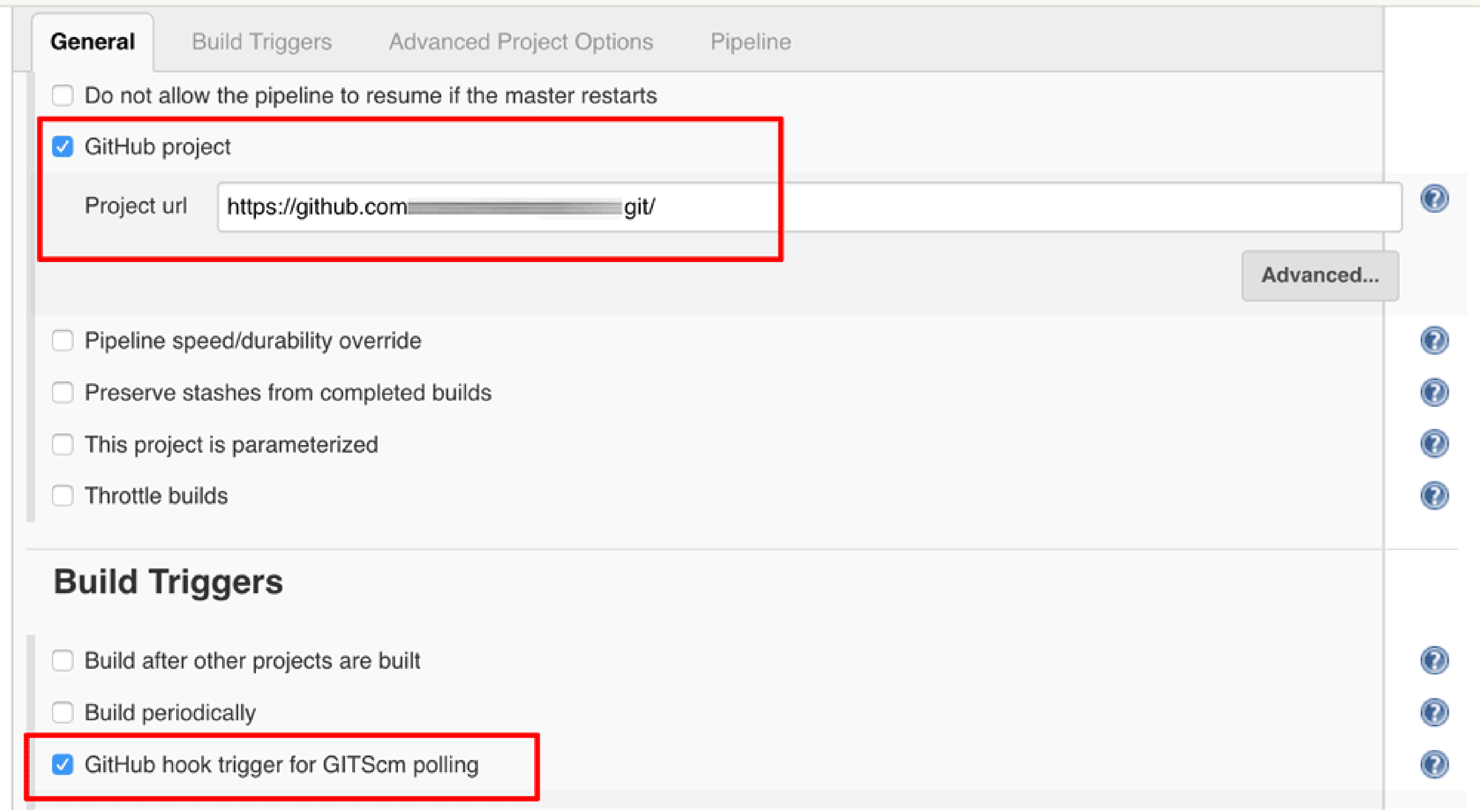
● Just the push event.

○ Send me **everything**.

○ Let me select individual events.

Create a Jenkins pipeline project

- Create "Pipeline“ project



□ Do not allow the pipeline to resume if the master restarts

☑ GitHub project

Project url: https://github.com[redacted]git/

□ Pipeline speed/durability override

□ Preserve stashes from completed builds

□ This project is parameterized

□ Throttle builds

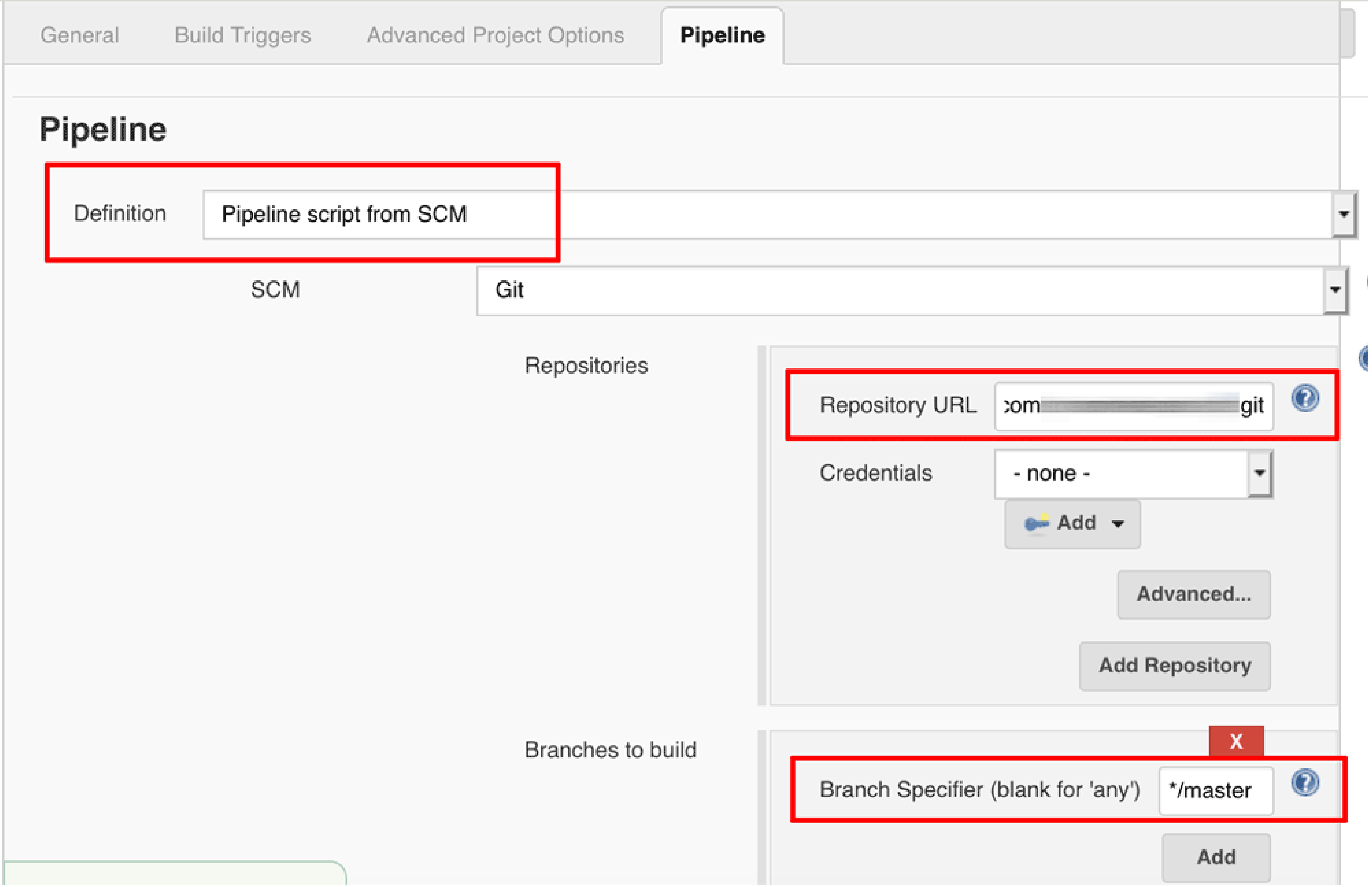
Build Triggers

□ Build after other projects are built

□ Build periodically

☑ GitHub hook trigger for GITScm polling

Create a Jenkins pipeline project



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: :om[redacted]git

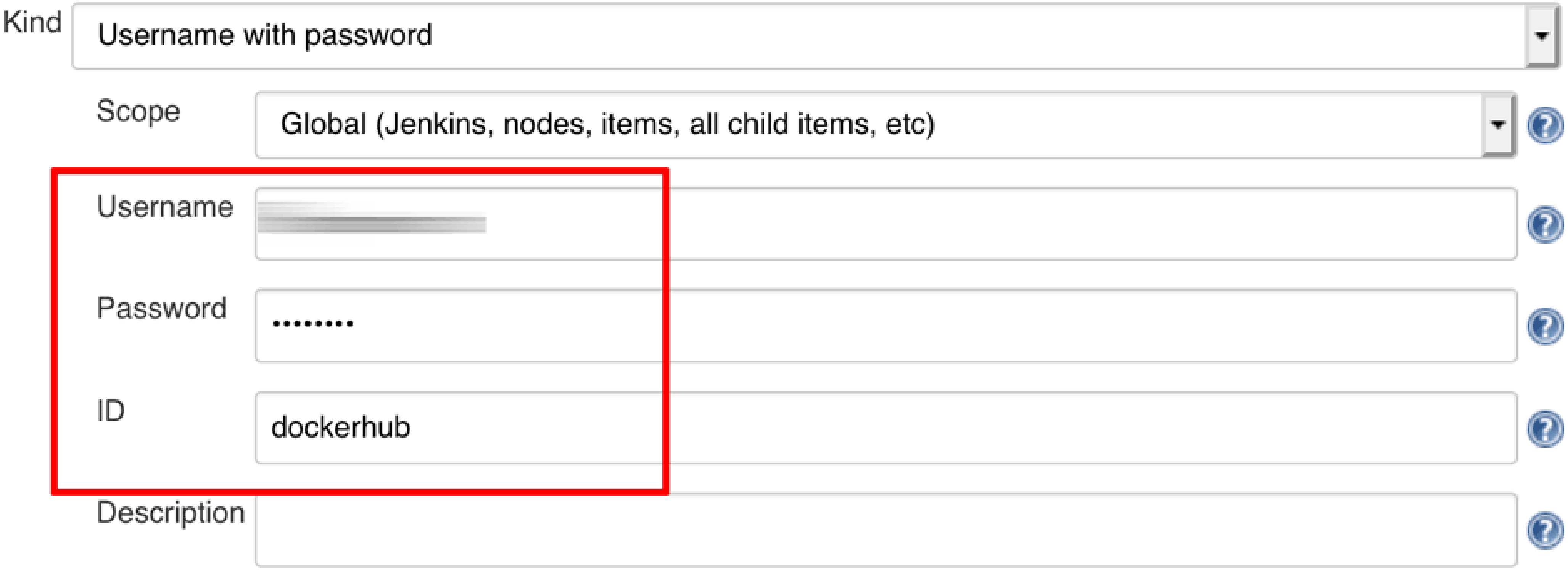
Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

Add credentials to Jenkins

- To communicate with
  - Docker Hub registry



Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: [redacted]

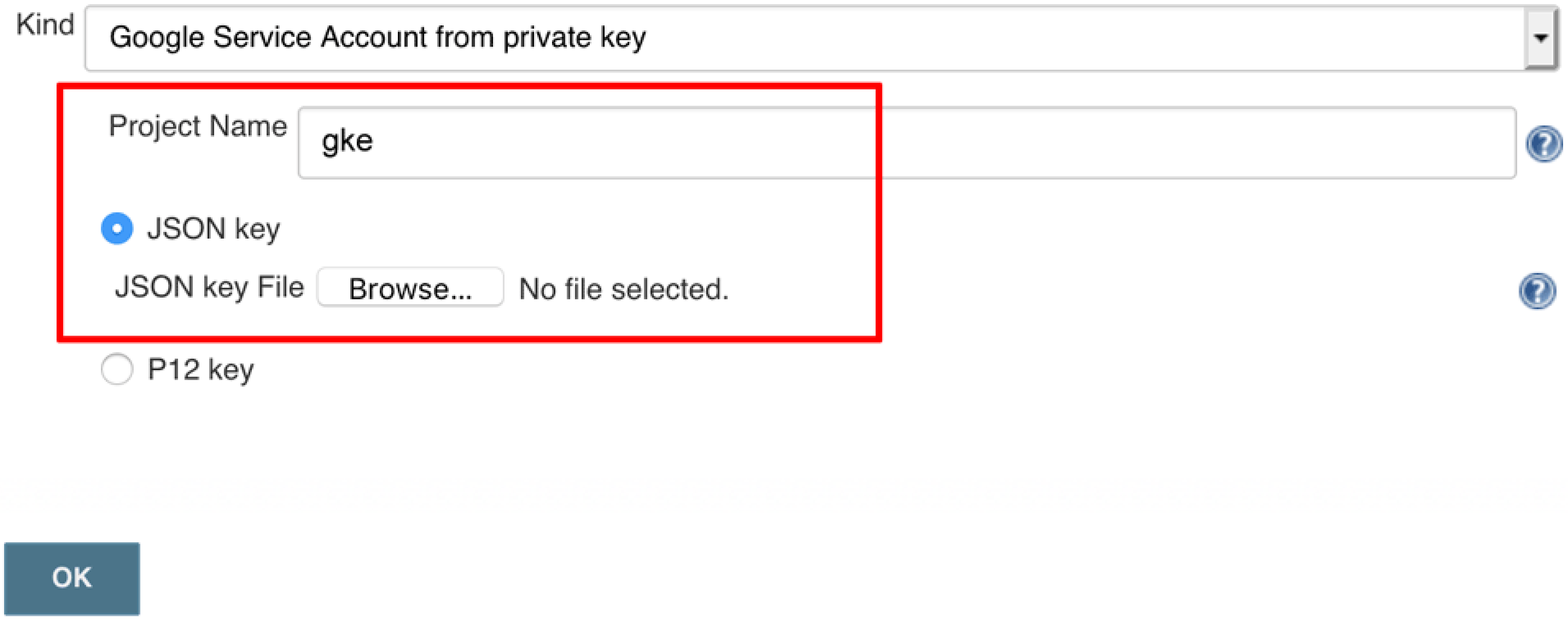
Password: •••••••••

ID: dockerhub

Description:

Add credentials to Jenkins

- To communicate with
  - Kubernetes cluster



Kind: Google Service Account from private key

Project Name: gke

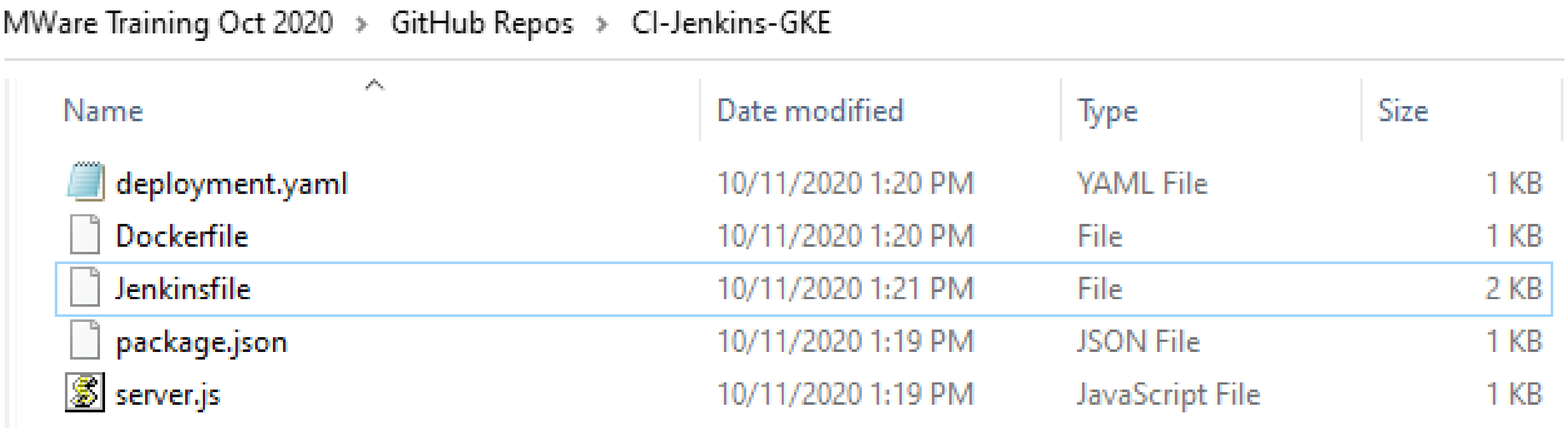
● JSON key

JSON key File: Browse... No file selected.

○ P12 key

Write code

https://github.com/atingupta2005/CI-Jenkins-GKE.git



MWare Training Oct 2020 › GitHub Repos › CI-Jenkins-GKE

Name	Date modified	Type	Size
deployment.yaml	10/11/2020 1:20 PM	YAML File	1 KB
Dockerfile	10/11/2020 1:20 PM	File	1 KB
Jenkinsfile	10/11/2020 1:21 PM	File	2 KB
package.json	10/11/2020 1:19 PM	JSON File	1 KB
server.js	10/11/2020 1:19 PM	JavaScript File	1 KB

- package.json
  - Listing the dependencies for the project
- server.js
  - Express application which returns a "Hello world" message on access
- deployment.yaml
  - Replace the DOCKER-HUB-USERNAME
- Jenkinsfile

Troubleshooting

- Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://%2Fvar%2Frun%2Fdocker.sock/v1.40/build?

- sudo groupadd docker
- sudo usermod -aG docker ${USER}
- sudo usermod -aG docker jenkins
- sudo usermod -a -G docker jenkins
- sudo chmod 777 /var/run/docker.sock
- sudo /etc/init.d/jenkins restart

Create Kubernetes Cluster

- Create Kubernetes Cluster on Google Cloud
  - Name: ibm-cluster-1
  - Location Type: Zone
  - Zone: us-central1-c

Deployment

- In pipeline script, replace the
  - PROJECT-ID
  - CLUSTER-NAME
  - CLUSTER-LOCATION and
  - DOCKER-HUB-USERNAME
- With
  - Google Compute Project project identifier
  - Kubernetes cluster name
  - Kubernetes cluster location and
  - Docker Hub username

Commit, Test and Repeat

- Pushing this commit should automatically trigger the pipeline in Jenkins

- On Google Console:
  - kubectl get deployments
  - kubectl get services

- Browse to port 5000 of the load balancer IP address
  - http://34.121.200.185:5000/

Thanks