# **Integrating Jenkins pipelines with GitHub webhooks on GCP**

# **Creating GCP project**

1. Login to <https://console.cloud.google.com/>
2. Create a new project: “QA Learning”
3. Once the project is created, make sure it is selected
4. Click on the navigation menu, open the compute engine option, and click on VM instances.
5. If the platform asks to enable the Compute Engine API, then click on “ENABLE” button
6. Click on “CREATE INSTANCE” button
7. Set the settings to:
   1. Name: learning-vm
   2. Region: us-west1 (Oregon)
   3. Zone: us-west1-b
   4. Series: E2
   5. Machine type: 2vCPU, 4 GB memory
   6. Boot disk: 10 GB
   7. Firewall: enable HTTP and HTTPS
8. Click on “CREATE” button

# **Installing the required applications**

1. Connect to the VM using the SSH button.
2. Install the open JDK
   1. sudo apt update
   2. sudo apt install openjdk-11-jre
   3. java -version
3. Install Jenkins
   1. curl -fsSL https://pkg.jenkins.io/debian/jenkins.io.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null
   2. echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null
   3. sudo apt-get update
   4. sudo apt-get install jenkins
   5. sudo systemctl start jenkins
   6. sudo systemctl status jenkins
4. Install GIT
   1. sudo apt install git-all
   2. git --version
5. Install docker
   1. sudo apt-get update
   2. sudo apt-get install ca-certificates curl gnupg lsb-release
   3. sudo mkdir -p /etc/apt/keyrings
   4. curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
   5. echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
   6. sudo apt-get update
   7. sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
   8. docker –v
   9. docker compose version

# **Setting up the tools**

1. VM configuration
   1. Go back to GCP and open the virtual machine that we created, click on its name.
   2. Scroll down until you find the “Network interfaces” information
   3. Under the “Network” column, click on “default” link
   4. Click on “FIREWALLS” tab
   5. Open “vpc-firewall-rules”
   6. Click on “ADD FIREWALL RULE”
   7. Set the settings to:
      1. Name: default-allow-jenkins
      2. Targets: All instances in the network
      3. Source IPv4 ranges: 0.0.0.0/0
      4. Specified protocols and ports: TCP 8080
   8. Click “CREATE” button
   9. Wait a minute and go to http://{{vm-public-ip}}:8080
   10. The unlock Jenkins webpage should be displayed
2. Jenkins configuration
   1. sudo cat /var/lib/jenkins/secrets/initialAdminPassword
   2. Copy the secret and paste it in the unlock Jenkins webpage
   3. Click on continue
   4. Click on “Install suggested plugins”
   5. Create the admin user:
      1. Username: jnkns-sr
      2. Password: jnkns-psswrd
      3. Full name: Jenkins
      4. Email: [jenkins@admin.com](mailto:jenkins@admin.com)
   6. Leave the instance configuration as the default value
   7. Click on “Save and Finish”
   8. Click on “Start using Jenkins”
   9. Click on “Manage Jenkins”
   10. Click on “Manage Plugins”
   11. Go to “Available Plugins” and install:
       1. HTML Publisher
       2. Multibranch Scan Webhook Trigger
   12. Open a terminal in the VM and run:
       1. sudo visudo
   13. Go until the end of the file and add the following like:
       1. jenkins ALL=(ALL) NOPASSWD: ALL
   14. Press control + o and hit enter
   15. Press control + x and hit enter
   16. Go to /var/lib/jenkins
       1. cd /var/lib/jenkins
   17. Create a new directory named “init.groovy.d”
       1. sudo mkdir init.groovy.d
   18. Enter inside the directory
       1. cd init.groovy.d
   19. Create a file named “startup-properties.groovy”
       1. sudo touch startup-properties.groovy
   20. Open the file
       1. sudo vim startup-properties.groovy
   21. Add the following lines inside it
       1. import jenkins.model.Jenkins
       2. System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "font-src 'self' data:; style-src \* http://\* 'unsafe-inline' 'unsafe-eval''");
   22. Save and exit from the file
       1. Press esc button
       2. Type :wq
       3. Hit enter
3. Docker configuration
   1. sudo groupadd docker
   2. sudo usermod -aG docker jenkins
   3. sudo usermod -aG docker {yourUser}
   4. newgrp docker
   5. getent group docker
   6. docker run hello-world
4. Restart jenkins
   1. sudo systemctl restart jenkins

# **Creating and configuring the Jenkins pipeline**

1. Create the pipeline
   1. From the Jenkins Dashboard, click on “New Item”
   2. Add the name “Cypress Default”
   3. Select “Multibranch Pipeline”
   4. Click on “Ok” button
2. Pipeline configuration
   1. In the “Branch Sources” click on “Add source”
   2. Select “Git”
   3. Paste the “projectLink” in the “Project Repository” field.
   4. Click on “Save” button
   5. Click on “Credentials” option
   6. Click on the domain that belongs to your project’s name
   7. Click on “Add Credentials”
   8. In the “Kind” dropdown, select Secret text
   9. In the “Secret” field, enter your personal email
   10. In the “ID” field, enter “NOTIFICATIONS\_EMAIL”
   11. Go back to your project and click on “Build Now”

# **Setting up our email server**

1. Register in any email server and get the configurations from the service provider, you will need:
   1. SMTP server address
   2. SMTP server port
   3. Credentials
2. Add the email server configurations in Jenkins
   1. Go to the Jenkins dashboard
   2. Click on “Manage Jenkins”
   3. Click on “Configure System”
   4. Scroll down until you see the section “System Admin e-mail address”
   5. Set the field to “Jenkins <noreply@automationtesting.com>"
   6. Scroll down until you see the section “Extended E-mail Notification”
   7. In the SMPT server field enter “smtp-relay.sendinblue.com”
   8. In SMTP port enter “587”
   9. Click on “Advanced”
   10. In the credentials section, click on “Add” button
   11. Select “Jenkins”
   12. Set the fields to:
       1. Kind: Username with password
       2. Username: jokewe4439@offsala.com
       3. Password: x7TUpJcbh9v24Cir
       4. ID: EMAIL\_CREDENTAILS
   13. Click on “Add” button
   14. For the “Credentials” dropdown, select the created credential
   15. Click on “Save” button

# **Setting up our GitHub build triggers**

1. Change the pipeline triggers in the Jenkins project:
   1. Go to the Jenkins dashboard and open the “Cypress Default” project.
   2. Click on “Configure”
   3. Click on “Scan Multibranch Pipeline Triggers”
   4. Enable the “Scan by webhook” setting
   5. In the trigger token, enter any string to use it as a token.
      1. Token: “13ad7492-d971-4435-a60a-2142f33b4aec”
   6. Click on Save button
2. Enable the GitHub Webhook feature
   1. Go to your GitHub project
   2. Click on “Settings”
   3. Click on “Webhooks”
   4. Click on “Add Webhook”
   5. You might be asked to enter your GitHub account password
   6. Set the fields to:
      1. Payload URL: <http://35.233.146.249:8080/multibranch-webhook-trigger/invoke?token=TOKENHERE>
      2. Content type: application/json
      3. Secret: Leave this field empty
   7. Click on “Add webhook”

# **Links of interest**

HTML publisher plugin: <https://plugins.jenkins.io/htmlpublisher/>

Multibranch webhook plugin: <https://plugins.jenkins.io/multibranch-scan-webhook-trigger/>

Content Security Policy: <https://content-security-policy.com/#source_list>

Disposable email: <https://temp-mail.org/en/>

Email server provider: <https://www.sendinblue.com/>