

# GitHub Action Workflow

[GitHub Actions](#) ? GitHub ??? CI/CD ??????

????????

1. ???????2000 ??/?
2. ???????????

## Tutorials

- [GitHub Actions ?????? Lint?Format ? Type Check - Code and Me](#)
- [How to Build a Production-Ready DevOps Pipeline with Free Tools](#)

## Building Docker Image (workflow)

- Your Repo ? Settings ? Security ? Secrets and variables ? Actions
  - Repository secrets
    - Name: DOCKERHUB\_TOKEN
    - Value: <YOUR-TOKEN>
  - Repository variables
    - Name: DOCKERHUB\_USERNAME
    - Value: <YOUR-USERNAME>
- ?? Docker Registry ????????[Docker Login · Actions · GitHub Marketplace](#)

.github/workflows/deploy.yml :

```
name: Build and Push Docker Image

# =====
# 
# =====
# - push: [ ] master [ ]
# - workflow_dispatch: [ ] GitHub Actions [ ]
# =====

on:
  #push:
  # branches:
```

```
# - main
workflow_dispatch:

# =====
#
# =====

env:
  IMAGE_NAME: gemini-ocr-fastapi

jobs:
  build-and-push:
    runs-on: ubuntu-latest
    steps:
      # -----
      # Step :
      # -----
      # GitHub runner
      #
      # -----
      - name: Checkout
        uses: actions/checkout@v4

      # -----
      # Step :
      # -----
      # GitHub Actions runner 14GB
      #
      # - .NET SDK
      # - Android SDK
      # - GHC Haskell
      # - Docker
      #
      # docker system prune
      # -----
      - name: Free GitHub Actions Disk Space
        run: |
          sudo rm -rf /usr/share/dotnet
          sudo rm -rf /usr/local/lib/android
```

```
sudo rm -rf /opt/ghc
# docker prune
sudo docker system prune -af || true
df -h
```

```
# -----
```

```
# Step : Docker Buildx
```

```
# -----
```

```
# Docker Buildx Docker
```

```
# - linux/amd64, linux/arm64
```

```
# -
```

```
# -
```

```
#
```

```
#
```

```
# - image=moby/buildkit:latest: buildkit
```

```
# - platforms: arm64 amd64
```

```
# -----
```

```
- name: Setup Docker Buildx
```

```
uses: docker/setup-buildx-action@v2
```

```
with:
```

```
  driver-opts: |
```

```
    image=moby/buildkit:latest
```

```
  platforms: linux/amd64,linux/arm64
```

```
# -----
```

```
# Step : Docker Hub Registry
```

```
# -----
```

```
# Docker Hub
```

```
#
```

```
#
```

```
# # - username/password: GitHub Secrets
```

```
#
```

```
# Docker Hub GitHub Variables Secrets
```

```
# - DOCKERHUB_USERNAME (Var)
```

```
# - DOCKERHUB_TOKEN (Secret): Personal Access Token
```

```
# -----
```

```
- name: Log in to Docker Hub
```

```
uses: docker/login-action@v3
```

```
with:
```

```

    username: ${vars.DOCKERHUB_USERNAME }}
    password: ${secrets.DOCKERHUB_TOKEN }}

# -----
# Step 5: Docker ACR
# -----
#
# 1. ./docker/Dockerfile
# 2. ACR
#
#
# - context:
# - file: Dockerfile
# - push: true registry
# - platforms: linux/arm64 ARM64 Apple Silicon ARM
# - tags: commit SHA
#
#
# - cache-from: registry
# - cache-to: registry
# - mode=max:
#
# stktrade.azurecr.io/stk-jixun-model:<commit-sha>
# stktrade.azurecr.io/stk-jixun-model:abc123def456
# -----
- name: Build and push Docker image
  uses: docker/build-push-action@v5
  with:
    context: ${github.workspace }}
    file: ./Dockerfile
    push: true
    platforms: linux/amd64
    tags: ${vars.DOCKERHUB_USERNAME }/${env.IMAGE_NAME }:${github.sha }}
    cache-from: type=registry,ref=${vars.DOCKERHUB_USERNAME }/${env.IMAGE_NAME
}}:buildcache
    cache-to: type=registry,ref=${vars.DOCKERHUB_USERNAME }/${env.IMAGE_NAME
}}:buildcache,mode=max

# -----

```

```

# Step 6: 📄📄📄📄
# -----
# 📄 GitHub Actions 📄 Summary 📄📄📄📄📄📄 Markdown 📄📄
# 📄📄
# - 📄📄📄📄📄registry📄📄📄📄📄
# - 📄📄📄📄📄📄📄📄📄
# - 📄📄📄📄📄📄commit SHA📄📄workflow run ID📄
#
# 📄📄📄📄📄
# - 📄📄📄📄📄📄
# - 📄📄📄📄📄📄📄📄
# - 📄📄📄📄📄📄
# 📄📄📄
# -----
- name: Image Pull Summary
  run: |
    DOCKERHUB_NAME="${{ vars.DOCKERHUB_USERNAME }}"
    IMAGE_NAME="${{ env.IMAGE_NAME }}"
    IMAGE_TAG="${DOCKERHUB_NAME}/${IMAGE_NAME}:${{ github.sha }}"
    COMMIT_SHA="${{ github.sha }}"
    RUN_ID="${{ github.run_id }}"
    echo "## 📄📄Image Build Summary" >> $GITHUB_STEP_SUMMARY
    echo "" >> $GITHUB_STEP_SUMMARY
    echo "### Image Information" >> $GITHUB_STEP_SUMMARY
    echo "- **Registry:** docker.io/\`${DOCKERHUB_NAME}\`" >> $GITHUB_STEP_SUMMARY
    echo "- **Image Name:** \`${IMAGE_NAME}\`" >> $GITHUB_STEP_SUMMARY
    echo "- **Tag:** \`${COMMIT_SHA}\`" >> $GITHUB_STEP_SUMMARY
    echo "- **Full Image:** \`${IMAGE_TAG}\`" >> $GITHUB_STEP_SUMMARY
    echo "" >> $GITHUB_STEP_SUMMARY
    echo "### How to Pull This Image" >> $GITHUB_STEP_SUMMARY
    echo "" >> $GITHUB_STEP_SUMMARY
    echo "1. **Login to Docker Hub:**" >> $GITHUB_STEP_SUMMARY
    echo "\`\`\`\`bash" >> $GITHUB_STEP_SUMMARY
    echo "docker login -u" >> $GITHUB_STEP_SUMMARY
    echo "\`\`\`\`" >> $GITHUB_STEP_SUMMARY
    echo "" >> $GITHUB_STEP_SUMMARY
    echo "2. **Pull the image:**" >> $GITHUB_STEP_SUMMARY
    echo "\`\`\`\`bash" >> $GITHUB_STEP_SUMMARY
    echo "docker pull ${DOCKERHUB_NAME}/${IMAGE_TAG}" >> $GITHUB_STEP_SUMMARY

```

```

echo "\\`\\`\\`" >> $GITHUB_STEP_SUMMARY
echo "" >> $GITHUB_STEP_SUMMARY
echo "3. **Run the container:**" >> $GITHUB_STEP_SUMMARY
echo "\\`\\`\\`bash" >> $GITHUB_STEP_SUMMARY
echo "docker run -d -e \"GOOGLE_API_KEY=███\" -p 8000:8000 ${IMAGE_TAG}" >>
$GITHUB_STEP_SUMMARY
echo "\\`\\`\\`" >> $GITHUB_STEP_SUMMARY
echo "" >> $GITHUB_STEP_SUMMARY
echo "---" >> $GITHUB_STEP_SUMMARY
echo "**Commit SHA:** \`${COMMIT_SHA}\` | **Workflow Run:** \`${RUN_ID}\`" >>
$GITHUB_STEP_SUMMARY

```

## K8s deployment with Helm

- ???? (Bastion) ??? Helm ???? ?
- ???GitHub Actions ? SSH to Bastion ? Execute Helm Upgrade ? K8s Cluster Updated

.github/workflows/deploy.yml :

```

jobs:
  # =====
  # Job 2: ?? Helm ??? Kubernetes ??
  # =====
  # ? job ???
  # 1. ?? build-and-deploy job ??????????????
  # 2. ?? SSH ??? Azure Bastion ??
  # 3. ? Bastion ????? Helm ???
  # 4. ?? Kubernetes ?????????
  #
  # ????? Bastion?
  # - ?????Kubernetes ?????????
  # - ????????? Bastion ?????
  # - ?????Bastion ?????????????
  #
  # ?????
  # GitHub Actions → SSH to Bastion → Execute Helm Upgrade → K8s Cluster Updated
  # =====
  helm-deploy:
    # ?? build-and-push job ?????????
    needs: build-and-push

```

```
# Helm workflow
# workflow file ./github/workflows/helm-deploy-reusable.yml
# workflow name
# -
# - workflow
# -
uses: ./github/workflows/helm-deploy-reusable.yml

# =====
#
# =====
# workflow
#
# - repo_path: Bastion
# - helm_release: Helm release
# - helm_namespace: Kubernetes namespace
#   stktrade-prod
# - helm_chart_path: Helm chart Bastion
# - helm_values_file: Helm values
# - kubeconfig_path: Kubernetes
# - helm_set_values: Helm values
# image.tag=${{ github.sha }} commit SHA
#
# =====

with:
  repo_path: /home/along/gemini-ocr-fastapi
  helm_release: gemini-ocr
  helm_namespace: my-devops-prod
  helm_chart_path: ~/gemini-ocr-fastapi/k8s/chart
  helm_values_file: ~/gemini-ocr-fastapi/k8s/chart/values.yaml
  kubeconfig_path: ~/my.kubeconfig
  helm_set_values: image.tag=${{ github.sha }}

# =====
# SSH
# =====
# secrets GitHub Secrets SSH Bastion
#
# - SSH_PRIVATE_KEY: SSH
```

```

# - SSH_HOST: Bastion IP
# - SSH_USER: SSH
#
#
# - GitHub Secrets
# =====

secrets:
  SSH_PRIVATE_KEY: ${ secrets.MY_BASTION_KEY }
  SSH_HOST: ${ secrets.MY_BASTION_HOST }
  SSH_USER: ${ secrets.MY_BASTION_USERNAME }

```

**.github/workflows/helm-deploy-reusable.yml :**

```

# =====
# GitHub Actions Reusable Workflow: Helm
# =====
#
# Workflow
# workflow reusable workflow workflow
# workflow_call
#
#
# 1. workflow workflow
# 2. inputs
# 3.
#
#
# 1. SSH Azure Bastion
# 2.
# 3. Kubernetes
# 4. Helm
# 5.
#
#
# -
# - workflow
# -
# -
# =====

```

name: Helm Deploy (Reusable)

```
# =====  
#   
# =====  
# workflow_call:   
#   
# workflow   
#   
# =====  
on:  
  workflow_call:  
    # =====  
    # (Inputs)  
    # =====  
    # workflow   
    # required: true  
    # required: false  
    # =====  
  inputs:  
    # -----  
    #   
    # -----  
  
  # repo_path:   
  #   
  # /home/azureuser/stk.jixun.model  
  repo_path:  
    description: 'Path to repository on remote machine'  
    required: true  
    type: string  
  
  # helm_release: Helm release   
  #   
  # namespace   
  helm_release:  
    description: 'Helm release name'  
    required: true  
    type: string  
  
  # helm_namespace: Kubernetes namespace  
  #   

```

```
# stktrade-prod 스택 스택trade-dev 스택
helm_namespace:
  description: 'Kubernetes namespace'
  required: true
  type: string

# helm_chart_path: Helm chart 경로
# Chart 경로
# ~/stk.jixun.model/k8s/chart
helm_chart_path:
  description: 'Path to helm chart on remote machine'
  required: true
  type: string

# helm_values_file: Helm values 파일
# Values 파일
# ~/stk.jixun.model/k8s/chart/values.yaml
helm_values_file:
  description: 'Path to values file on remote machine'
  required: true
  type: string

# -----
# SSH 연결
# -----

# ssh_host: IP 주소
# secrets.SSH_HOST 키
# secrets.SSH_HOST > inputs.ssh_host
ssh_host:
  description: 'Target machine hostname or IP address (optional if SSH_HOST secret is
provided)'
  required: false
  type: string

# ssh_user: SSH 사용자
# secrets.SSH_USER 키
# secrets.SSH_USER > inputs.ssh_user
ssh_user:
  description: 'SSH username (optional if SSH_USER secret is provided)'
```

```
    required: false
    type: string

# ssh_port: SSH port
# port 22 SSH port
# port 22 SSH port
ssh_port:
  description: 'SSH port'
  required: false
  type: string
  default: '22'

# -----
# Kubernetes Helm port
# -----

# kubeconfig_path: Kubernetes kubeconfig
# Kubeconfig path
# path ~/.kube/config Kubernetes kubeconfig
kubeconfig_path:
  description: 'Kubeconfig path on remote machine'
  required: false
  type: string
  default: '~/.kube/config'

# helm_timeout: Helm upgrade timeout
# Helm upgrade timeout
# 5m5s
# 5m + 5s, m=5, s=5
helm_timeout:
  description: 'Helm upgrade timeout'
  required: false
  type: string
  default: '5m'

# helm_wait: wait for pods
# true: wait for Pod ready
# false: skip wait for Pod ready
# default true
helm_wait:
```

```

description: 'Wait for deployment to complete'
required: false
type: boolean
default: true

# helm_set_values: Helm values
# values.yaml
# key1=value1,key2=value2
# image.tag=abc123,replicaCount=3
#
#
# - image.tag=${{ github.sha }}
# -
# -
helm_set_values:
  description: 'Additional --set values (format: key1=value1,key2=value2)'
  required: false
  type: string

# =====
# Secrets
# =====
# Secrets
# secrets workflow workflow
# =====

secrets:
  # SSH_PRIVATE_KEY: SSH
  #
  # authorized_keys
  SSH_PRIVATE_KEY:
    description: 'SSH private key for authentication'
    required: true

  # SSH_HOST: IP
  # secret
  # inputs.ssh_host
  SSH_HOST:
    description: 'Target machine hostname or IP address (optional, use if ssh_host input
is a secret)'
    required: false

```

```
# SSH_USER: SSH username
# secrets.SSH_PRIVATE_KEY secret key
# inputs.ssh_user
SSH_USER:
  description: 'SSH username (optional, use if ssh_user input is a secret)'
  required: false
```

jobs:

```
# =====
# Job: Helm chart
# =====
# job uses SSH action to Helm chart
# =====
```

helm-deploy:

runs-on: ubuntu-latest

steps:

```
# -----
# Step: SSH to Helm chart
# -----
# uses appleboy/ssh-action action
#
# steps
# 1. SSH to
# 2. run script
# 3.
# -----
```

- name: Deploy with Helm

uses: appleboy/ssh-action@v1

with:

```
# SSH key
# secrets > inputs
# secrets.SSH_PRIVATE_KEY secret key
host: ${ secrets.SSH_HOST || inputs.ssh_host }
username: ${ secrets.SSH_USER || inputs.ssh_user }
key: ${ secrets.SSH_PRIVATE_KEY }
port: ${ inputs.ssh_port || '22' }
```

```
# =====
```



```

#
# 1. cd
# 2. git fetch origin:
# 3. git checkout -B:
#   - -B
#   - "${{ github.ref_name }}" workflow master
#   - "origin/${{ github.ref_name }}"
#
#
# -
# - Helm chart values
# -
# =====
cd "${{ inputs.repo_path }}" || exit 1
git fetch origin
# reset --hard
git reset --hard "origin/${{ github.ref_name }}"

# =====
# 2: Kubernetes
# =====
# KUBECONFIG kubectl helm
#
# KUBECONFIG
# - Kubernetes
# -
# - Kubernetes
#
# ~/.kube/config Kubernetes
# kubeconfig_path
# =====
export KUBECONFIG=${{ inputs.kubeconfig_path || '~/kube/config' }}

# =====
# 3: Helm
# =====
# Helm upgrade --install
#
# helm upgrade --install
# - --install: release

```

```

#
#
#
# helm upgrade --install <release-name> <chart-path> \
#   --namespace <namespace> \
#   --values <values-file> \
#   --timeout <timeout>
#
#
# - ${ inputs.helm_release }: release
# - ${ inputs.helm_chart_path }: chart
# - --namespace: namespace
# - --values: values
# - --timeout: 5m
# =====
HELM_CMD="helm upgrade --install ${ inputs.helm_release } ${
inputs.helm_chart_path } --namespace ${ inputs.helm_namespace } --values ${
inputs.helm_values_file } --timeout ${ inputs.helm_timeout || '5m' }"

# =====
# 4: --wait
# =====
# helm_wait true --wait
#
# --wait
# - Pod
# -
# - Pod
#
#
# - helm_wait boolean shell "true" "false"
# -
#
#
# - true
# - false
# =====
if [ "${ inputs.helm_wait }" = "true" ]; then
    HELM_CMD="${HELM_CMD} --wait"
fi

```

```

# =====
# 5:  --set
# =====
# helm_set_values
#
#
# 1. helm_set_values
# 2.
# 3.  --set
#
#
# image.tag=abc123,replicaCount=3
# --set image.tag=abc123 --set replicaCount=3
#
# IFS Internal Field Separator
# - IFS=','
# - read -ra SET_VALUES
# - <<< here-string
#
#
# - image.tag=${{ github.sha }}
# - replicaCount
# -
# =====
HELM_SET_VALUES="${{ inputs.helm_set_values }}"
if [ -n "${HELM_SET_VALUES}" ]; then
    #
    IFS=',' read -ra SET_VALUES <<< "${HELM_SET_VALUES}"
    #  --set
    for set_val in "${SET_VALUES[@]}"; do
        HELM_CMD="${HELM_CMD} --set ${set_val}"
    done
fi

# =====
# 6: Helm
# =====
# eval Helm
#

```

```

# eval
# - HELM_CMD
# -
# - eval
#
#
# - Helm
# - Helm workflow
#
#
# - --wait Pod
# -
# - trap
# =====
eval $HELM_CMD

```

## Manually run

```

# Create a secret for dockerhub-pull-secret
kubectl create secret docker-registry dockerhub-pull-secret \
  --docker-server=docker.io \
  --docker-username=alantw \
  --docker-password='XXXXXXXXXXXXXXXXXXXX' \
  -n my-devops-prod

# Create a secret for gemini-ocr-secret
kubectl create secret generic gemini-ocr-api-secret \
  --from-literal=API_KEY='ThisIsTheAPKey' \
  --from-literal=GEMINI_API_KEY='XXXXXXXXXXXX' \
  -n my-devops-prod

# Deploy with helm
# Usage:
# helm upgrade --install your-app-name k8s/chart \
#   --namespace your-namespace-prod \
#   --create-namespace
cd /path/to/your/repo
helm upgrade --install gemini-ocr k8s/chart --namespace my-devops-prod --values
k8s/chart/values.yml --timeout 5m

```

# Build .deb package (workflow)

- workflow: <https://github.com/zquestz/plank-reloaded/blob/master/.github/workflows/debian-release.yml>
- Build and Release on Debian Bookworm
- Use Package-checking tool - [Lintian](#)

## ???? README ????

- workflow: <https://github.com/hesamsheikh/awesome-openclaw-usecases/blob/main/.github/workflows/update-badge.yml>

---

Revision #28

Created 2026-01-14 11:37:28 CST by A-Lang (Admin)

Updated 2026-03-20 10:15:35 CST by A-Lang (Admin)