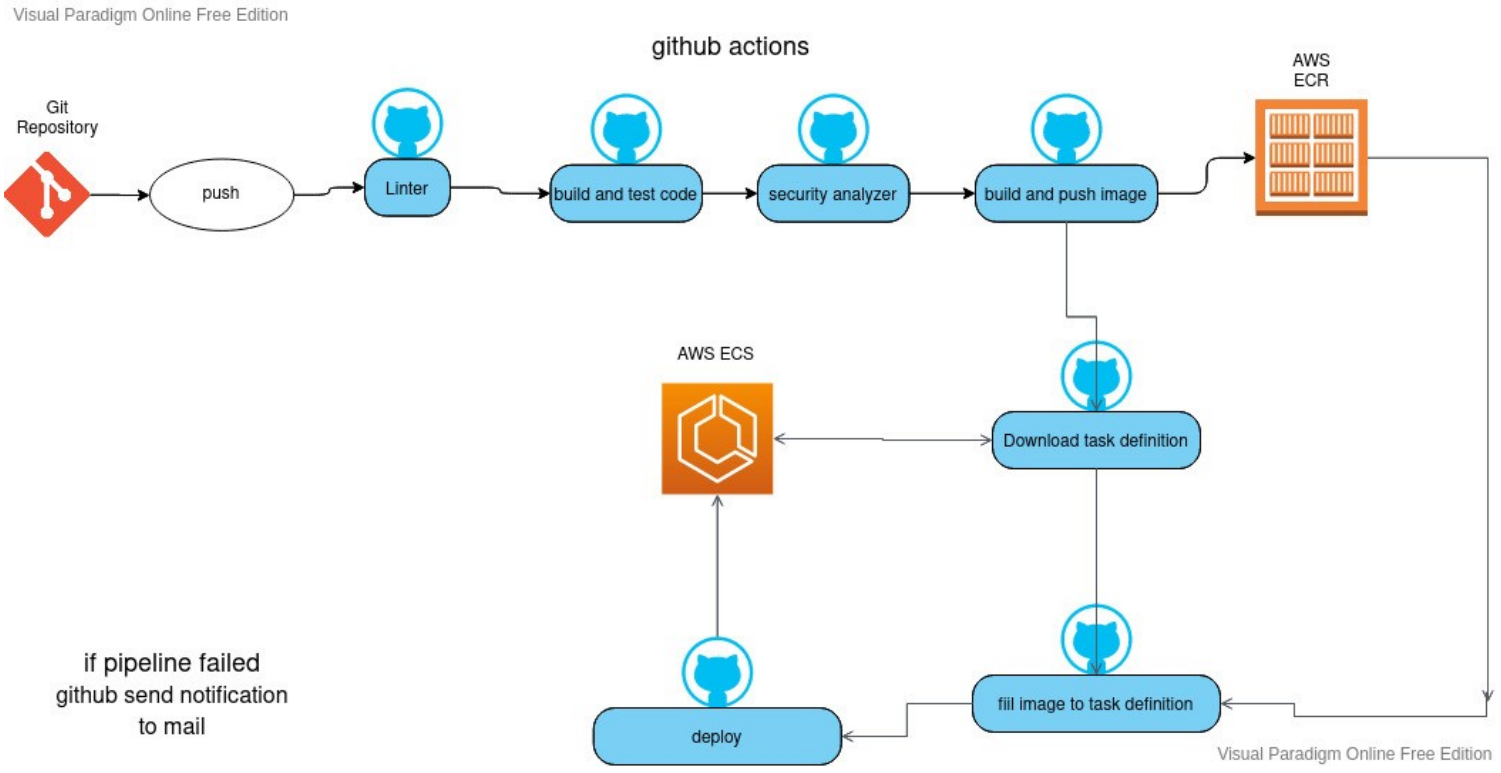
Andersen DevOps exam

30.07.2021

by Gizar Zigangirov

CI/CD diagram



Linter:
python ← pylint | golang ← golangci-lint

```yaml
pylint:

  runs-on: ubuntu-latest

  steps:
  - uses: actions/checkout@v2
  - name: Set up Python 3.9
    uses: actions/setup-python@v2
    with:
      python-version: 3.9
  - name: Install dependencies
    run: |
      python -m pip install --upgrade pip
      pip install pylint
      if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
  - name: Lint with pylint
    run: |
      pylint --exit-zero $(ls -R | grep .py$ | xargs)
```

```yaml
golangci:
  name: lint
  runs-on: ubuntu-latest
  steps:
    - uses: actions/checkout@v2
    - name: golangci-lint
      uses: golangci/golangci-lint-action@v2
      with:
        # Optional: version of golangci-lint to
        version: latest
```

Build/test:
python ← pytest | golang ← builtin (go build/go test)

```yaml
pytest:
  needs: pylint
  runs-on: ubuntu-latest

  steps:
  - uses: actions/checkout@v2
  - name: Set up Python 3.9
    uses: actions/setup-python@v2
    with:
      python-version: 3.9
  - name: Install dependencies
    run: |
      python -m pip install --upgrade pip
      pip install pytest
      if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
  - name: Test with pytest
    run: |
      pytest
```

```yaml
build-test:
  needs: golangci

  runs-on: ubuntu-latest
  steps:
  - uses: actions/checkout@v2

  - name: Set up Go
    uses: actions/setup-go@v2
    with:
      go-version: 1.16.6

  - name: Build
    run: go build -v ./...

  - name: Test
    run: go test -v ./...
```

Security analyzer:
python ← bandit | golang ← gosec

```yaml
gosec:
  needs: build-test
  runs-on: ubuntu-latest
  env:
    GO111MODULE: on
  steps:
    - name: Checkout Source
      uses: actions/checkout@v2

    - name: Run Gosec Security Scanner
      uses: securego/gosec@master
      with:
        args: ./...
```

```yaml
bandit_test:

  runs-on: ubuntu-latest

  needs: pytest
  steps:
  - uses: actions/checkout@v2

  - name: Set up Python 3.9
    uses: actions/setup-python@v2
    with:
      python-version: 3.9

  - name: Install dependencies
    run: |
      python -m pip install --upgrade pip
      pip install bandit
      if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
  - name: analyze source code with bandit
    run: |
      bandit -c bandit.yml -s B104 -r .
```

Build & push docker image:
Step 1: aws login

```yaml
steps:
- name: Checkout
  uses: actions/checkout@v2

- name: Configure AWS credentials
  uses: aws-actions/configure-aws-credentials@v1
  with:
    aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
    aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
    aws-region: eu-central-1

- name: Login to Amazon ECR
  id: login-ecr
  uses: aws-actions/amazon-ecr-login@v1
```

Build & push docker image:
Step 2: build, tag and push docker image to AWS ECR

```yaml
- name: Build, tag, and push image to Amazon ECR
  id: build-image
  env:
    ECR_REGISTRY: ${{ steps.login-ecr.outputs.registry }}
    ECR_REPOSITORY: ${{ secrets.REPO_NAME }}
    IMAGE_TAG: ${{ github.sha }}
    L: "latest"
  run: |
    # Build a docker container and
    # push it to ECR so that it can
    # be deployed to ECS.
    docker build -t $ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG -t $ECR_REGISTRY/$ECR_REPOSITORY:$L .
    docker push $ECR_REGISTRY/$ECR_REPOSITORY --all-tags
    echo "::set-output name=image::$ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG"
```

Download task defintion from active service of AWS ECS

```
- name: Download task definition
  run: |
    aws ecs describe-task-definition --task-definition andersen-exam-web --query taskDefinition > task-definition.json
```

```
- name: Download task definition
  run: |
    aws ecs describe-task-definition --task-definition andersen-exam-web-golang --query taskDefinition > task-definition.json
```

Fill in the new image in the AWS ECS task definition

```yaml
- name: Fill in the new image ID in the Amazon ECS task definition
  id: task-def
  uses: aws-actions/amazon-ecs-render-task-definition@v1
  with:
    task-definition: task-definition.json
    container-name: andersen-exam-python-container
    image: ${{ steps.build-image.outputs.image }}
```

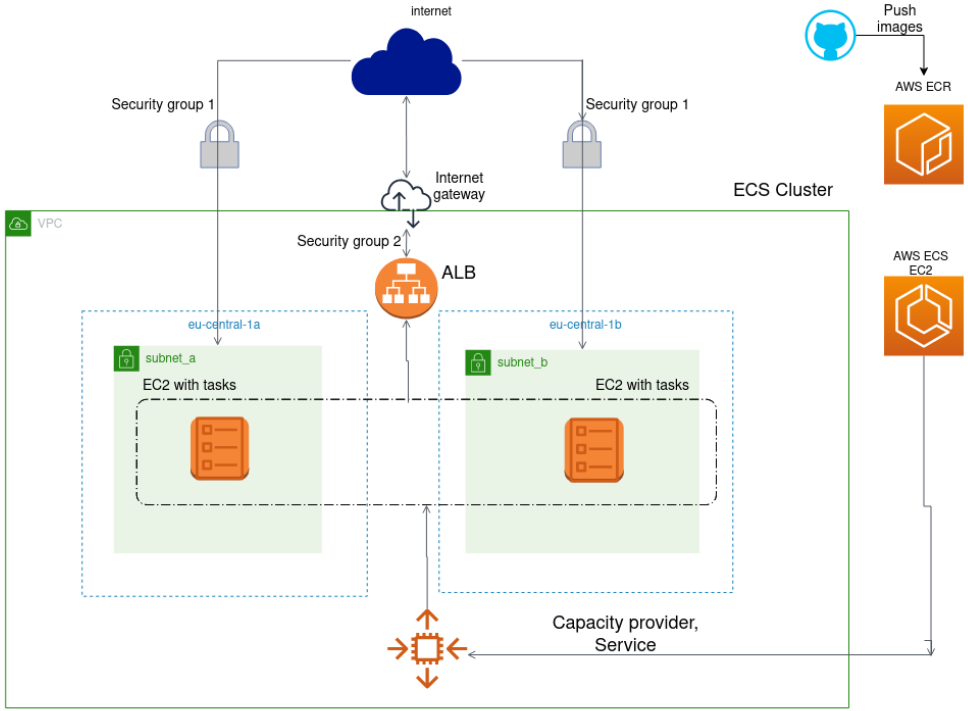
```yaml
- name: Fill in the new image ID in the Amazon ECS task definition
  id: task-def
  uses: aws-actions/amazon-ecs-render-task-definition@v1
  with:
    task-definition: task-definition.json
    container-name: andersen-exam-golang-container
    image: ${{ steps.build-image.outputs.image }}
```

Deploy AWS ECS task definition

```yaml
- name: Deploy Amazon ECS task definition
  uses: aws-actions/amazon-ecs-deploy-task-definition@v1
  with:
    task-definition: ${{ steps.task-def.outputs.task-definition }}
    service: andersen-exam-python-service
    cluster: andersen-exam-python-cluster
    wait-for-service-stability: true
```

```yaml
- name: Deploy Amazon ECS task definition
  uses: aws-actions/amazon-ecs-deploy-task-definition@v1
  with:
    task-definition: ${{ steps.task-def.outputs.task-definition }}
    service: andersen-exam-golang-service
    cluster: andersen-exam-golang-cluster
    wait-for-service-stability: true
```

AWS diagram


internet
Push images
AWS ECR
Security group 1
Security group 1
Internet gateway
ECS Cluster
VPC
Security group 2
ALB
AWS ECS EC2
eu-central-1a
eu-central-1b
subnet_a
subnet_b
EC2 with tasks
EC2 with tasks
Capacity provider, Service

Utopia CI/CD (my opinion)