This workflow will build and push a new container image to Amazon ECR,

and then will deploy a new task definition to Amazon ECS, when there is a push to the "main" branch.

#

To use this workflow, you will need to complete the following set-up steps:

#

1. Create an ECR repository to store your images.

For example: `aws ecr create-repository --repository-name my-ecr-repo --region us-east-2`.

Replace the value of the `ECR_REPOSITORY` environment variable in the workflow below with your repository's name.

Replace the value of the `AWS_REGION` environment variable in the workflow below with your repository's region.

#

2. Create an ECS task definition, an ECS cluster, and an ECS service.

For example, follow the Getting Started guide on the ECS console:

https://us-east-2.console.aws.amazon.com/ecs/home?region=us-east-2#/firstRun

Replace the value of the `ECS_SERVICE` environment variable in the workflow below with the name you set for the Amazon ECS service.

Replace the value of the `ECS_CLUSTER` environment variable in the workflow below with the name you set for the cluster.

#

3. Store your ECS task definition as a JSON file in your repository.

The format should follow the output of `aws ecs register-task-definition --generate-cli-skeleton`.

Replace the value of the `ECS_TASK_DEFINITION` environment variable in the workflow below with the path to the JSON file.

Replace the value of the `CONTAINER_NAME` environment variable in the workflow below with

```
the name of the container
#    in the `containerDefinitions` section of the task definition.
#
# 4. Store an IAM user access key in GitHub Actions secrets named `AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`.
#    See the documentation for each action used below for the recommended IAM policies for this IAM user,
#    and best practices on handling the access key credentials.

name: Deploy to Amazon ECS

on:
  push:
    branches: [ "main" ]

env:
  AWS_REGION: MY_AWS_REGION                   # set this to your preferred AWS region, e.g. us-west-1
  ECR_REPOSITORY: MY_ECR_REPOSITORY           # set this to your Amazon ECR repository name
  ECS_SERVICE: MY_ECS_SERVICE                 # set this to your Amazon ECS service name
  ECS_CLUSTER: MY_ECS_CLUSTER                 # set this to your Amazon ECS cluster name
  ECS_TASK_DEFINITION: MY_ECS_TASK_DEFINITION # set this to the path to your Amazon ECS task definition
                                              # file, e.g. .aws/task-definition.json
  CONTAINER_NAME: MY_CONTAINER_NAME           # set this to the name of the container in the
```

```
                    # containerDefinitions section of your task definition

permissions:
  contents: read

jobs:
  deploy:
    name: Deploy
    runs-on: ubuntu-latest
    environment: production

    steps:
    - name: Checkout
      uses: actions/checkout@v4

    - name: Configure AWS credentials
      uses: aws-actions/configure-aws-credentials@v4
      with:
        aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
        aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
        aws-region: ${{ env.AWS_REGION }}

    - name: Login to Amazon ECR
      id: login-ecr
      uses: aws-actions/amazon-ecr-login@v2
```

```
    - name: Build, tag, and push image to Amazon ECR
      id: build-image
      env:
        ECR_REGISTRY: ${{ steps.login-ecr.outputs.registry }}
        IMAGE_TAG: ${{ github.sha }}
      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 .
        docker push $ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG
                                echo    "image=$ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG"    >>
$GITHUB_OUTPUT

    - 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: ${{ env.ECS_TASK_DEFINITION }}
        container-name: ${{ env.CONTAINER_NAME }}
        image: ${{ steps.build-image.outputs.image }}

    - name: Deploy Amazon ECS task definition
      uses: aws-actions/amazon-ecs-deploy-task-definition@v2
      with:
        task-definition: ${{ steps.task-def.outputs.task-definition }}
```

service: ${{ env.ECS_SERVICE }}

cluster: ${{ env.ECS_CLUSTER }}

wait-for-service-stability: true