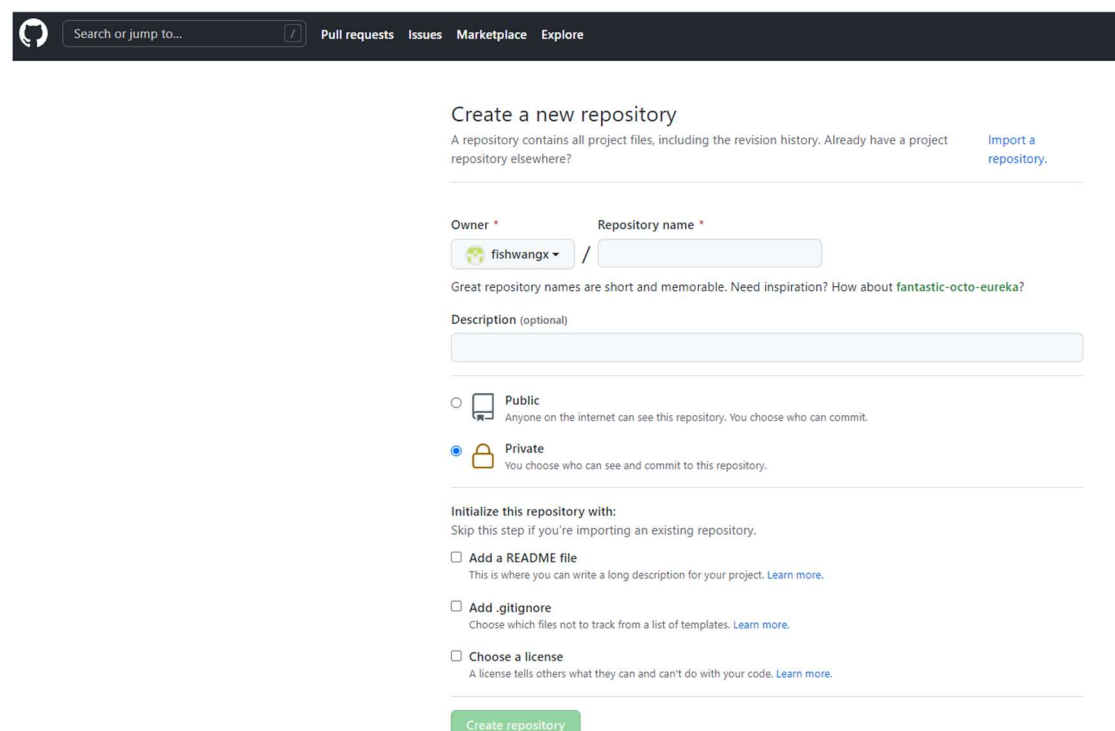


使用 Github Action 来 build 和推送镜像到 AWS ECR

本文受卢申乐大师启发，并加入自己的探讨做出来的。主要目的是用于减少在本地 Build 和推送镜像到 AWS ECR 的时间。由于网络原因，本人试过 5 个小时也没办法 build 成功一次，使用本文的方法，一般只需要 10 分钟左右（从 Build 到推送到 ECR 的时间）。

本文例子上传到 github：https://github.com/fishwangx/github\_action

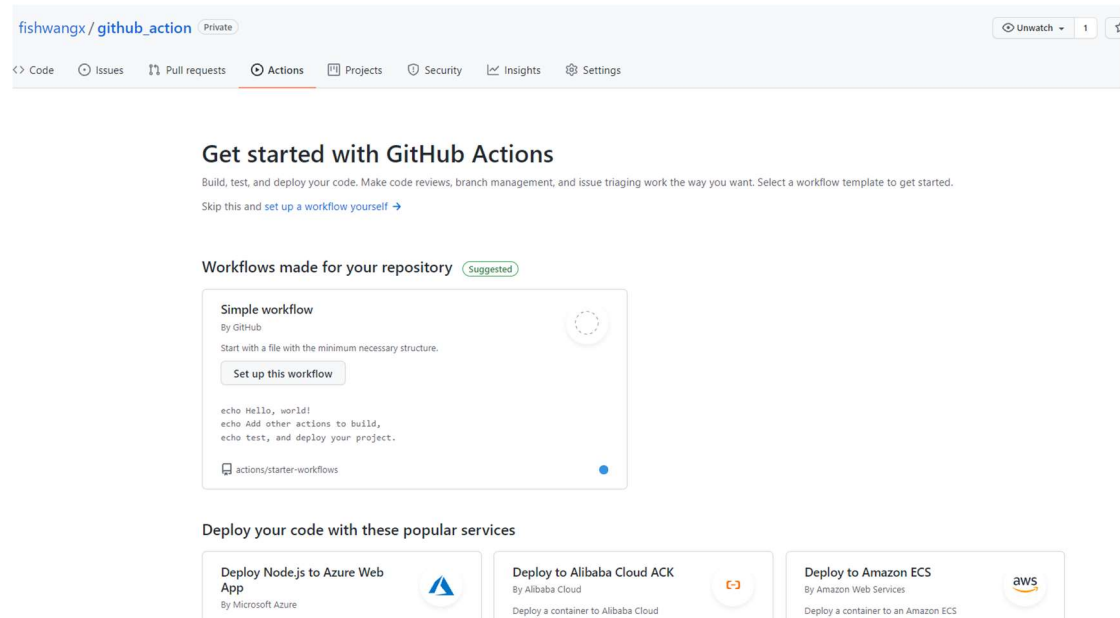
第一步：创建 github repository，请改成 private 的。



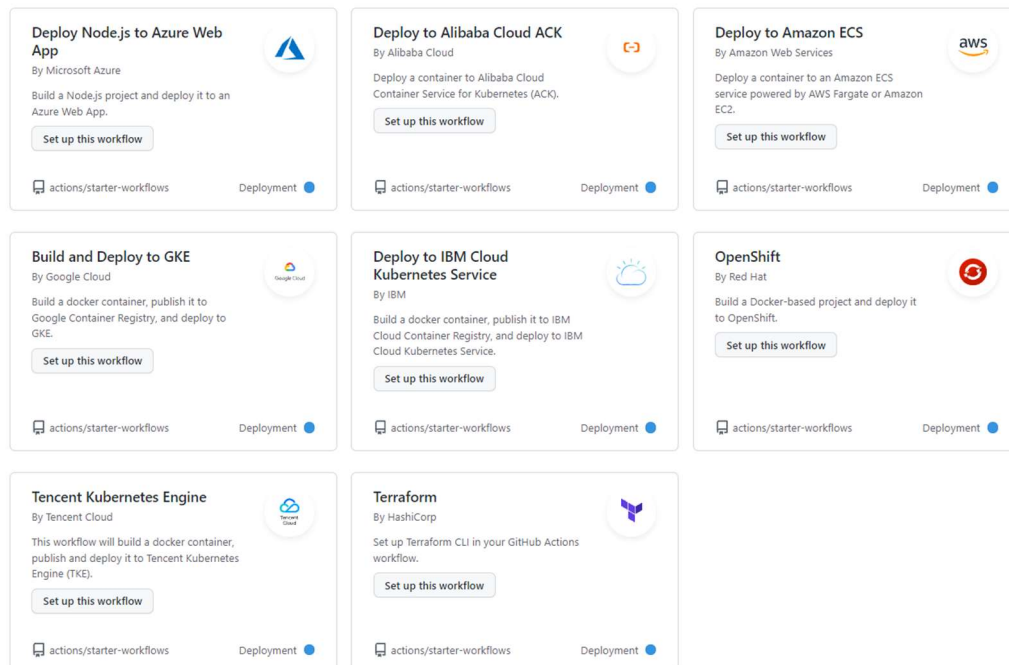
以上建完以后，就会转到一系列的命令提示页面，主要是初始化仓库。  
把代码输入到你的代码所在文件夹。（如下，Ubuntu 的输入）

```
(base) yu@DESKTOP-77RS18H:github-action$ echo "# github_action" >> README.md
(base) yu@DESKTOP-77RS18H:github-action$ git init
Initialized empty Git repository in /mnt/d/OneDrive/ASU/CSE546 cloud computing/week5/Project3/github-action/.git/
(base) yu@DESKTOP-77RS18H:github-action$ git add README.md
(base) yu@DESKTOP-77RS18H:github-action$ C:/Users/yhome/miniconda3/Scripts/activate
-bash: C:/Users/yhome/miniconda3/Scripts/activate: No such file or directory
(base) yu@DESKTOP-77RS18H:github-action$ conda activate base
(base) yu@DESKTOP-77RS18H:github-action$ git commit -m "first commit"
[master (root-commit) 4c6ca68] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
(base) yu@DESKTOP-77RS18H:github-action$ git branch -M main
(base) yu@DESKTOP-77RS18H:github-action$ git remote add origin git@github.com:fishwangx/github_action.git
(base) yu@DESKTOP-77RS18H:github-action$ git push -u origin main
Counting objects: 3, done.
Writing objects: 100% (3/3), 228 bytes | 20.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:fishwangx/github_action.git
 * [new branch]    main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
(base) yu@DESKTOP-77RS18H:github-action$
```

然后点 Actions，出现下面的界面。点 Deploy to Amazon ECS



### Deploy your code with these popular services



出现下面的界面，编辑 aws.yml，主要是把前面几行填好和删除 ECS 相关的行，剩下 ECR 的就行。使用本 github 修改过的 aws.yml 即可，只需要修改地区和你的 ECR 仓库名称。

The image consists of two screenshots of the GitHub Actions interface, showing the process of editing a workflow file named `aws.yml`.

**Top Screenshot:** The interface shows the `aws.yml` file in the `main` branch. The file content is as follows:

```
1 # This workflow will build and push a new container image to Amazon ECR,
2 # and then will deploy a new task definition to Amazon ECS, when a release is created
3 #
4 # To use this workflow, you will need to complete the following set-up steps:
5 #
6 # 1. Create an ECR repository to store your images.
7 #   For example: "aws ecr create-repository --repository-name my-ecr-repo --region us-east-2".
8 #   Replace the value of the "ECR_REPOSITORY" environment variable in the workflow below with your repository's name.
9 #   Replace the value of the "AWS_REGION" environment variable in the workflow below with your repository's region.
10 #
11 # 2. Create an ECS task definition, an ECS cluster, and an ECS service.
12 #   For example, follow the Getting Started guide on the ECS console:
13 #   https://us-east-2.console.aws.amazon.com/ecs/home?region=us-east-2#/first-run
14 #   Replace the value of the "ECS_SERVICE" environment variable in the workflow below with the name you set for the Amazon ECS service.
15 #   Replace the value of the "ECS_CLUSTER" environment variable in the workflow below with the name you set for the cluster.
16 #
17 # 3. Store your ECS task definition as a JSON file in your repository.
18 #   The format should follow the output of "aws ecs register-task-definition --generate-cli-skeleton".
19 #   Replace the value of the "ECS_TASK_DEFINITION" environment variable in the workflow below with the path to the JSON file.
20 #   Replace the value of the "CONTAINER_NAME" environment variable in the workflow below with the name of the container
21 #   in the "containerDefinitions" section of the task definition.
22 #
23 # 4. Store an IAM user access key in GitHub Actions secrets named "AWS_ACCESS_KEY_ID" and "AWS_SECRET_ACCESS_KEY".
24 #   See the documentation for each action used below for the recommended IAM policies for this IAM user,
25 #   and best practices on handling the access key credentials.
26
27 name: Deploy to Amazon ECS
28
```

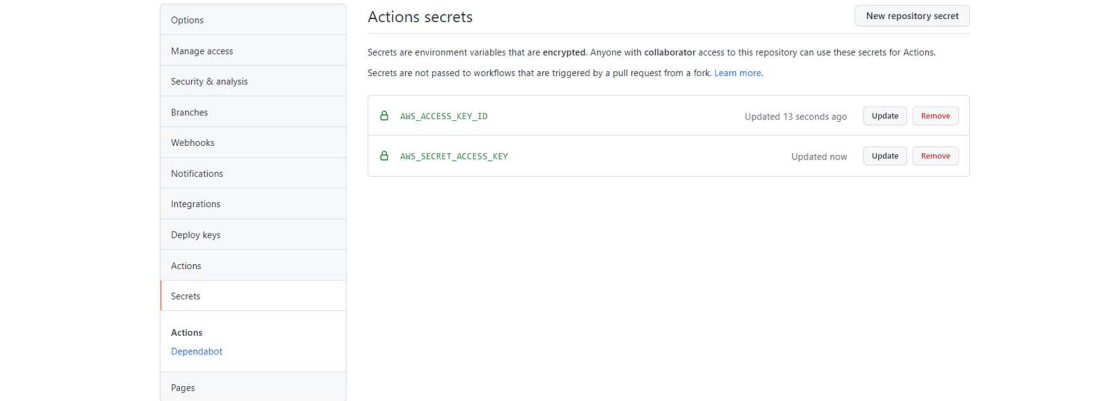
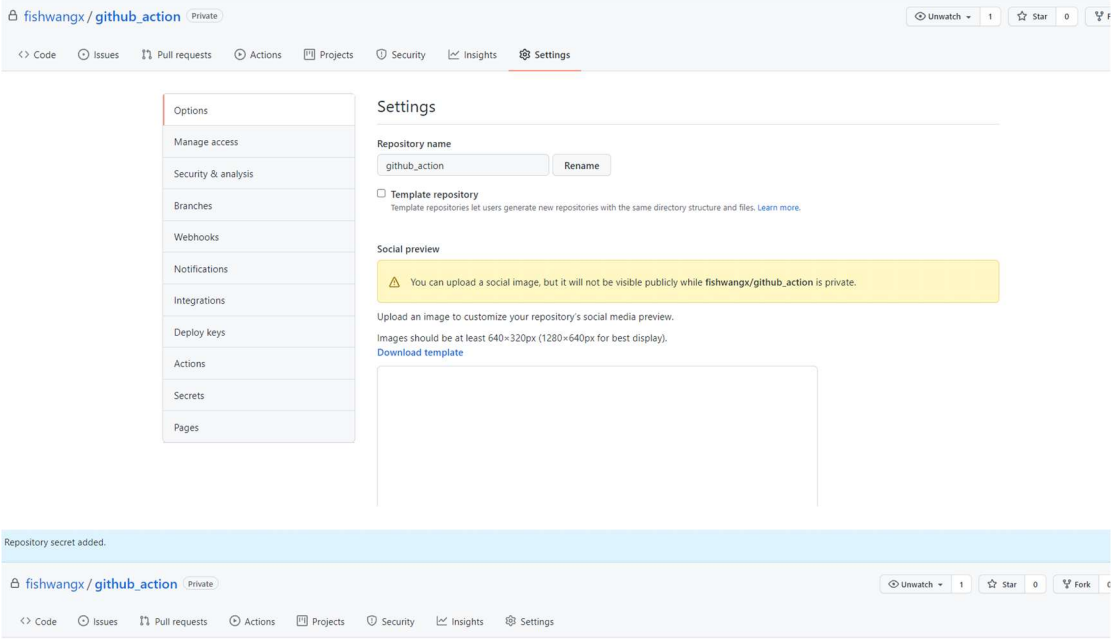
The right sidebar shows the **Commit new file** dialog with the filename `create aws.yml` and the option to commit directly to the `main` branch.

**Bottom Screenshot:** The interface shows the `aws.yml` file after editing. The file content is as follows:

```
24 # See the documentation for each action used below for the recommended IAM policies for this IAM user,
25 # and best practices on handling the access key credentials.
26
27 name: Deploy to Amazon ECS
28
29 on:
30   push:
31     types: [created]
32
33 env:
34   AWS_REGION: ap-northeast-2 # set this to your preferred AWS region, e.g. us-west-1
35   ECR_REPOSITORY: hello_docker # set this to your Amazon ECR repository name
36
37   CONTAINER_NAME: hello # set this to the name of the container in the
38                           # containerDefinitions section of your task definition
39
40 jobs:
41   deploy:
42     name: Deploy
43     runs-on: ubuntu-latest
44     environment: production
45
46 ..
```

The right sidebar shows the **Commit changes** dialog with the filename `updated aws.yml` and the option to commit directly to the `main` branch.

下面是添加 credentials ， 点 Secrets



```
(base) yu@DESKTOP-77RSI8H:github-action$ git push -u origin main
Warning: Permanently added the RSA host key for IP address '13.229.188.59' to the list of known hosts.
To github.com:fishwangx/github_action.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'git@github.com:fishwangx/github_action.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
(base) yu@DESKTOP-77RSI8H:github-action$ git pull
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 10 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (10/10), done.
From github.com:fishwangx/github_action
 4c6ca68..893f1e1  main      -> origin/main
Updating 4c6ca68..893f1e1
Fast-forward
 .github/workflows/aws.yml | 88 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 88 insertions(+)
 create mode 100644 .github/workflows/aws.yml
(base) yu@DESKTOP-77RSI8H:github-action$ git push -u origin main
Branch 'main' set up to track remote branch 'main' from 'origin'.
Everything up-to-date
(base) yu@DESKTOP-77RSI8H:github-action$
```

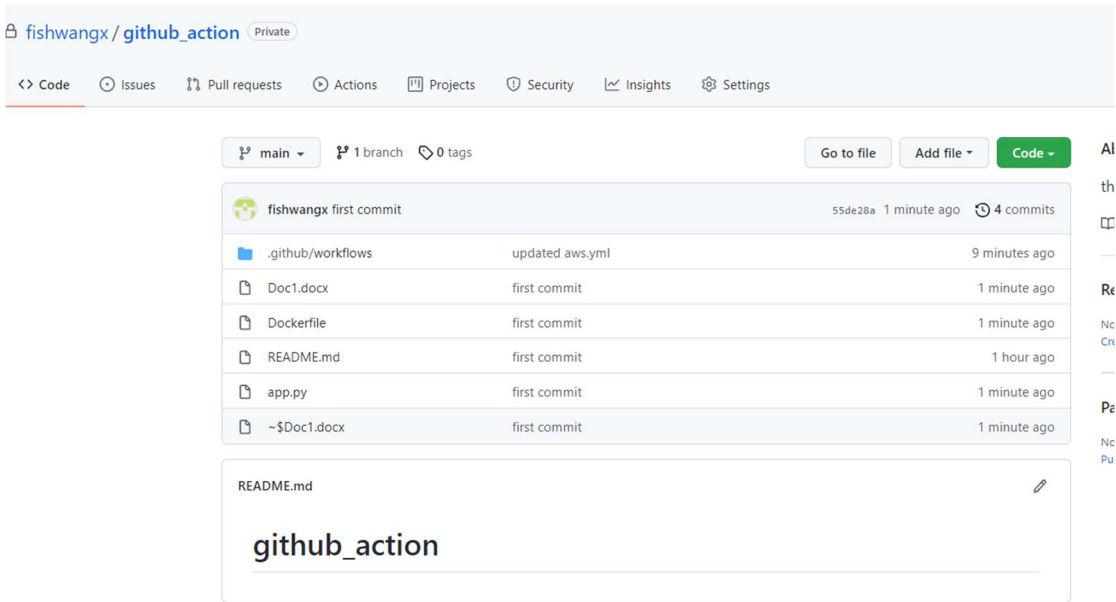
```
(base) yu@DESKTOP-77RSI8H:github-action$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

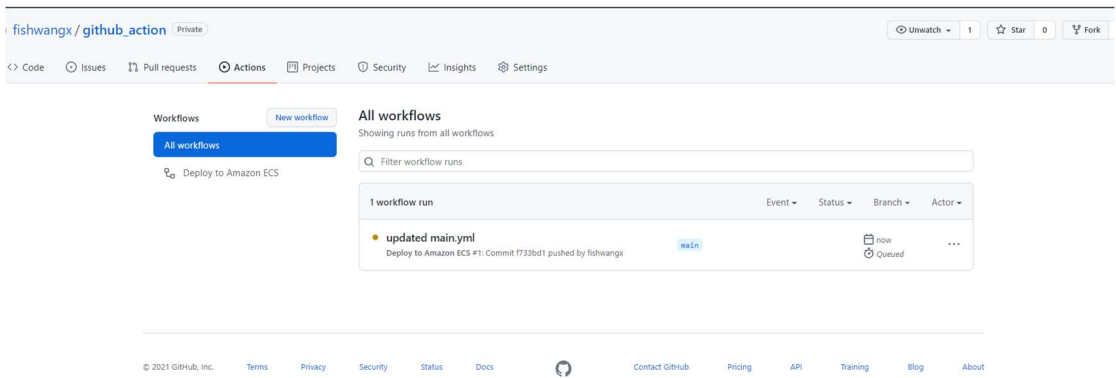
        Doc1.docx
        Dockerfile
        app.py
        ~$Doc1.docx

nothing added to commit but untracked files present (use "git add" to track)
(base) yu@DESKTOP-77RSI8H:github-action$ git add .
(base) yu@DESKTOP-77RSI8H:github-action$ git commit -m "first commit"
[main 55de28a] first commit
4 files changed, 7 insertions(+)
create mode 100644 Doc1.docx
create mode 100644 Dockerfile
create mode 100644 app.py
create mode 100644 ~$Doc1.docx
(base) yu@DESKTOP-77RSI8H:github-action$ git push -u origin main
Counting objects: 6, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 725.65 KiB | 2.84 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0)
To github.com:fishwangx/github_action.git
   893f1e1..55de28a  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
(base) yu@DESKTOP-77RSI8H:github-action$
```

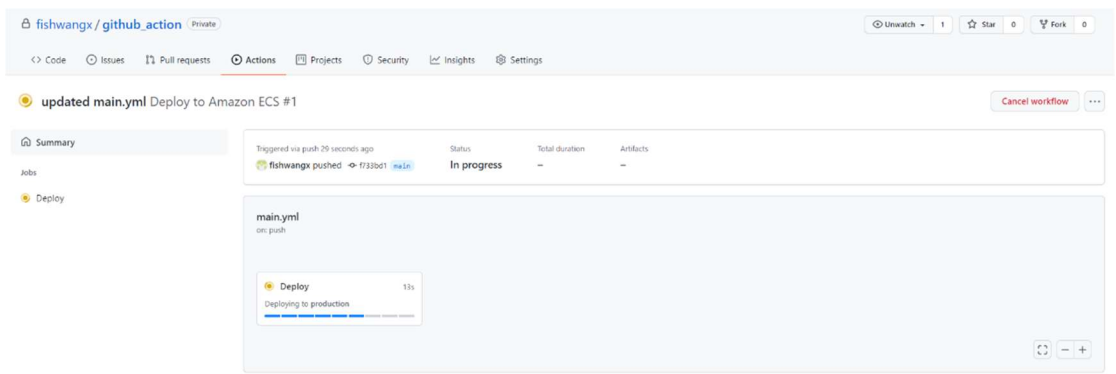
检查一下，看看文件是否都已经 push 到了这个仓库。如果是，点开 Actions，看看状态。



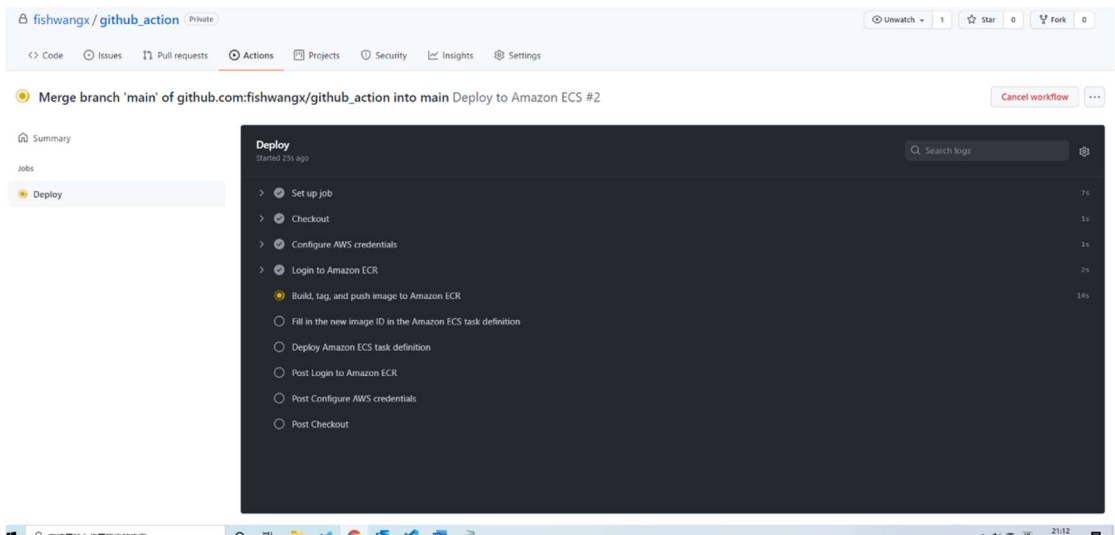
点 Actions 应该可以看到下面的情况。表面在排队运行。



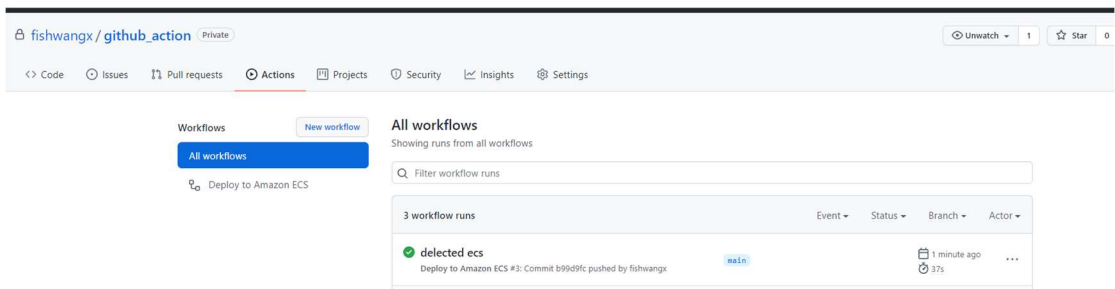
点进去可以看进度



点开看进度



当都执行完后，有个绿色的钩子。



这个时候，查看 Amazon ECR，应该可以看到推送过来的镜像了。就可以进行下一步处理。

