Project-2 Report

Submitted by:
Zesen Zhuang
zz229

Date: February 22, 2023

1 Overview

The following directory tree shows the structure of the project.

```
├── app.py ..... flask app for recommendation service and frontend with webui
├── docker/ ..... dockerfiles for images
│   ├── api.dockerfile
│   └── ml.dockerfile
├── figures/ ..... images produced by tests
│   ├── modify_replica_num.png
│   ├── update_code.png
│   └── update_data.png
├── generate_model.py
├── k8s/ ..... k8s yaml. I use Kustomize
│   └── base/
│       ├── config.properties ..... training data url and data version stored
│       ├── config.properties.bak
│       ├── deployment.yaml ..... recommendation api and ui
│       ├── kustomization.yaml
│       ├── pod.yaml ..... machine learning container
│       ├── pvc.yaml ..... persistent volume claim
│       └── service.yaml
├── makefile ..... automate the build, deploy and test
├── model.py ..... ml model source code
├── requirements.txt
├── result/ ..... logs produced by tests
│   ├── modify_replica_num.json
│   ├── status_changes.txt
│   ├── update_code.json
│   └── update_data.json
├── static/ ..... flask ui static folder
│   └── css/
│       └── styles.css
├── templates/ ..... flask ui template folder
│   ├── index.html
│   └── recommendation.html
├── test/ ..... api performance test
│   └── test_api_health.py
└── visualization_code/ ..... code to visualize test logs
    └── plot.py
```

GitHub Repo The GitHub repository is `https://github.com/Crinstaniev/cs401/tree/master/project-2/code`

Kustomize I applied `kustomize` to simplify the deployment.

UI I implemented a webui for the recommendation system and deployed in the api deployment.

GitHub Webhook to ArgoCD I configured a webhook in GitHub to trigger ArgoCD to sync the repository. Originally, ArgoCD will sync the repository every 3 minutes. Instead, I configured a webhook to trigger ArgoCD to sync the repository immediately.

PostSync Hook on Pod and Deployment Configuration Instead of manually changing the name of resources, I added ArgoCD hook to force the re-deployment of pods once any of the manifest file updates.

2 Tests

Three tests are conducted in this project. The first test is to change the replica number of `api` deployment. The second test is to change the code version (update the docker image) of `api` deployment. The third test is to change the data source of `ml` pod. The test results are shown in the following sections.

2.1 Test: Change Replicas Number

- **Test Description:** Change the number of replicas from 1 to 5. I change the number of replicas in `deployment.yaml` from 1 to 5.
- **Test Result:** The test passed, ArgoCD detected the change pods number of `api` deployment to 5.
- **Test Analysis:** The test passed because the number of replicas can be changed. The service status during the test is shown in Figure 1. It takes 5 seconds for the service to fully restart and increase the replica number.

2.2 Test: Code Update

- **Test Description:** I modified the health check API `/api/health` so that it return `{ status: online! }` instead of `{ status: online }` when the service is online.
- **Test Result:** The test passed, ArgoCD detected the change and re-deploy the deployment.

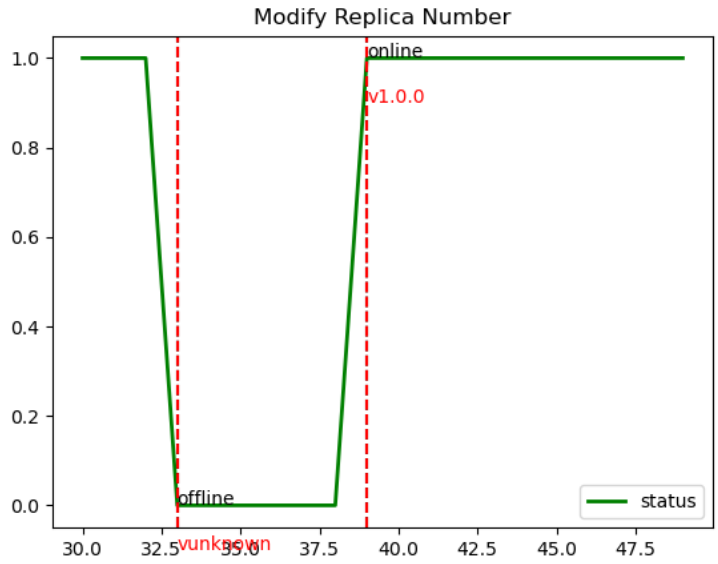

Figure 1: Service Status for Test: Change Replicas Number

The timeline is is trimmed. At time 0, the commit is pushed to GitHub and the webhook is triggered. Within few seconds, ArgoCD syncs the repository. Before 33 seconds, the deployment stayed at the old version. At 33, the service is down and resumes at 38.

- **Test Analysis:** The service status during the test is shown in Figure 2. Compared to test 1, it takes longer to reboot since the replica number increased to 5. After going online again, the service experienced a short-period down for which the reason worth further discussion.

2.3 Test: Update Training Data

- **Test Description:** I changed the training data source from `playlist-sample-ds1.csv` to `playlist-sample-ds2.csv`, and bump the data version in `config.properties` from `1.0.0` to `2.0.0`.
- **Test Result:** The test passed, ArgoCD detected the change and re-deploy the pod.
- **Test Analysis:** The service status during the test is shown in Figure 3. Similar to test 2, it takes longer to reboot since the replica number is 5. The data version also bumpped to `v2.0.0` as the service went online again.

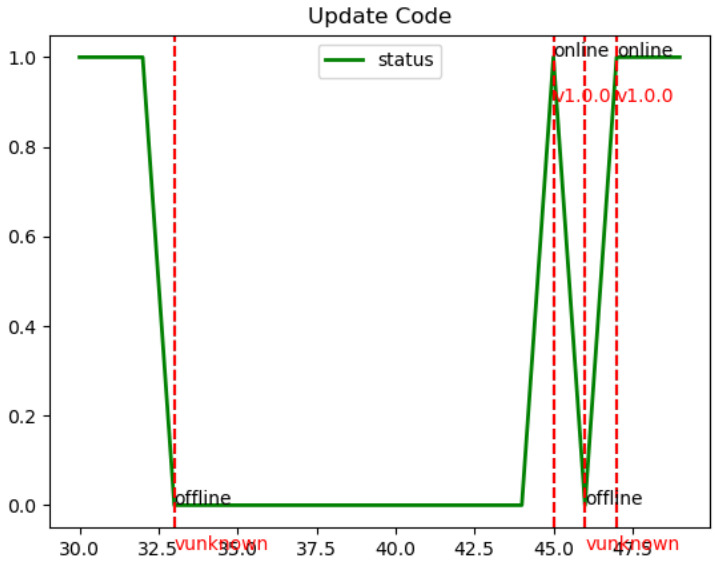

Figure 2: Service Status for Test: Update Code

The service went offline at 33 and back to online at 45. However, it was down again at 46 and went back immidiately at 47.

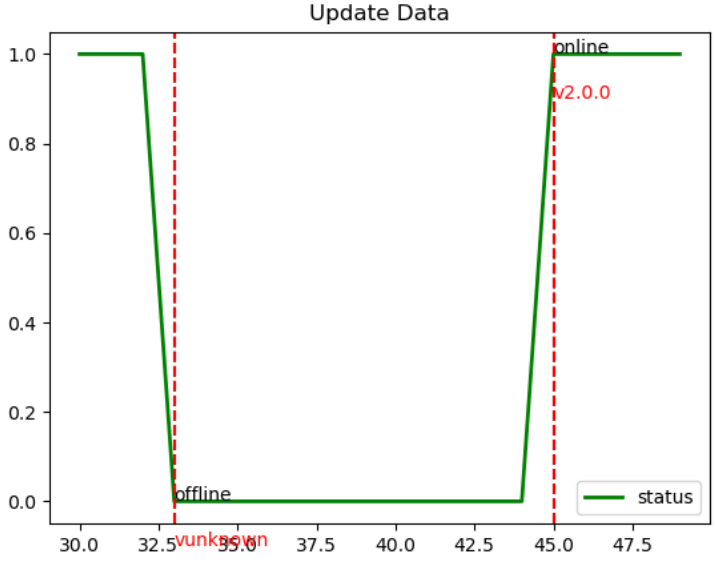

Figure 3: Service Status for Test: Update Data

The service went offline at 33 and back to online at 45.