Kubernetes Argo CD 实战应用

张岩峰老师微信，加我微信，邀请你加入 VIP 交流答疑群：

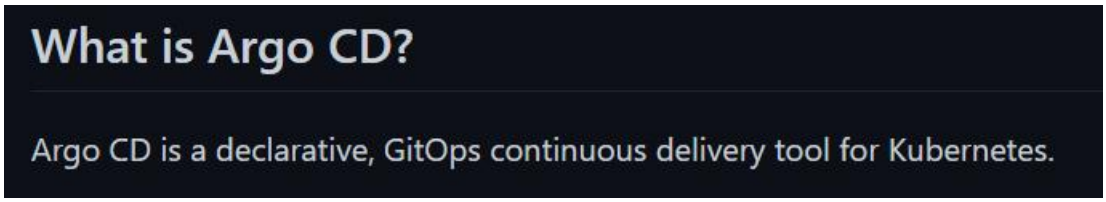
微信号：ZhangYanFeng0429

二维码：

1、什么是 Argo CD

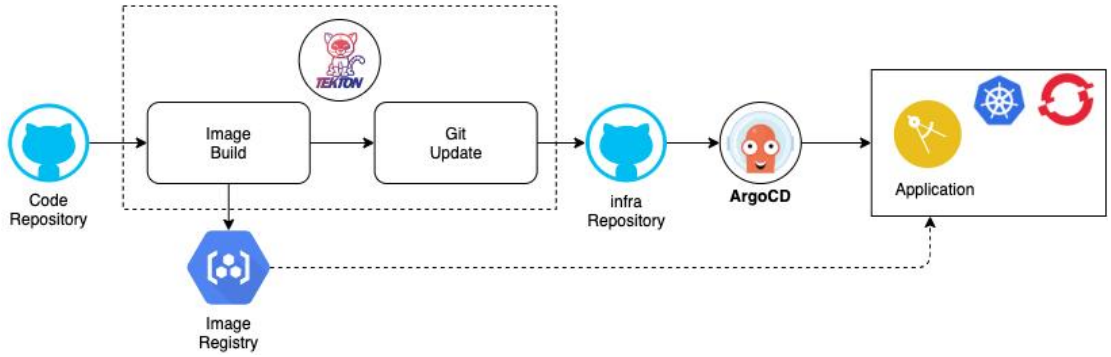
来自官网的介绍：

官网地址：https://argoproj.github.io/

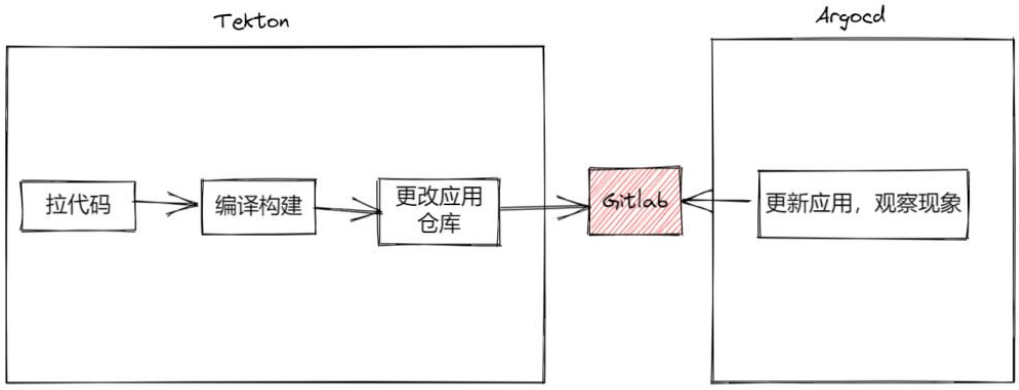


Argo CD 是一个基于 Kubernetes 的声明式的 GitOps 工具。

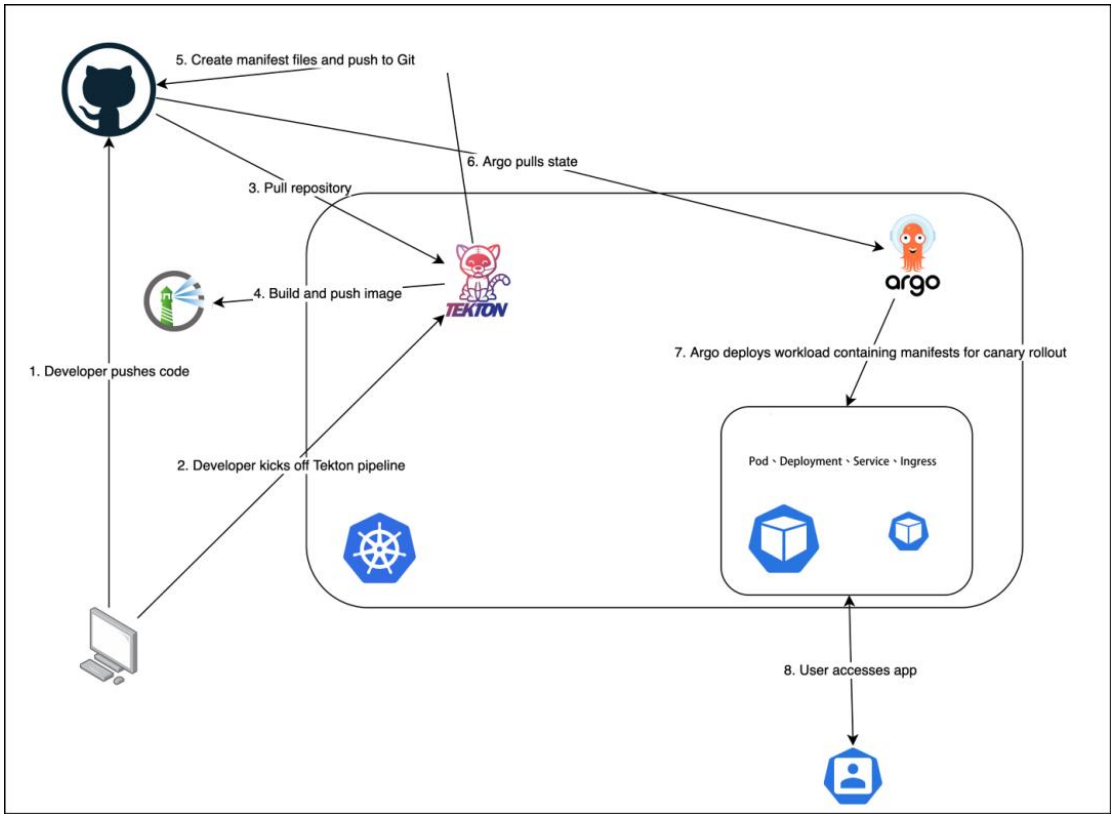
前面我们使用 Tekton 完成了应用的 CI 流程，但是 CD 在 Tekton 的任务中去完成，可控性不强，因此现在我们使用 GitOps 的方式来改造我们的流水线，将 CD 部分使用 Argo CD 来完成，以实现应用部署及应用部署回滚的可控性。



回顾下前面的 Tekton 自动化流水线项目流程，整个流水线包括 clone 源代码(Clone)、单元测试(Test)、源码构建(Build)、容器镜像构建及推送(docker)、应用部署（deploy）、应用部署回滚（rollback）几个部分的任务，最后的应用部署（deploy）和应用部署回滚（rollback）属于 CD 部分，这部分可替换为 Argo CD 来部署及回滚操作。



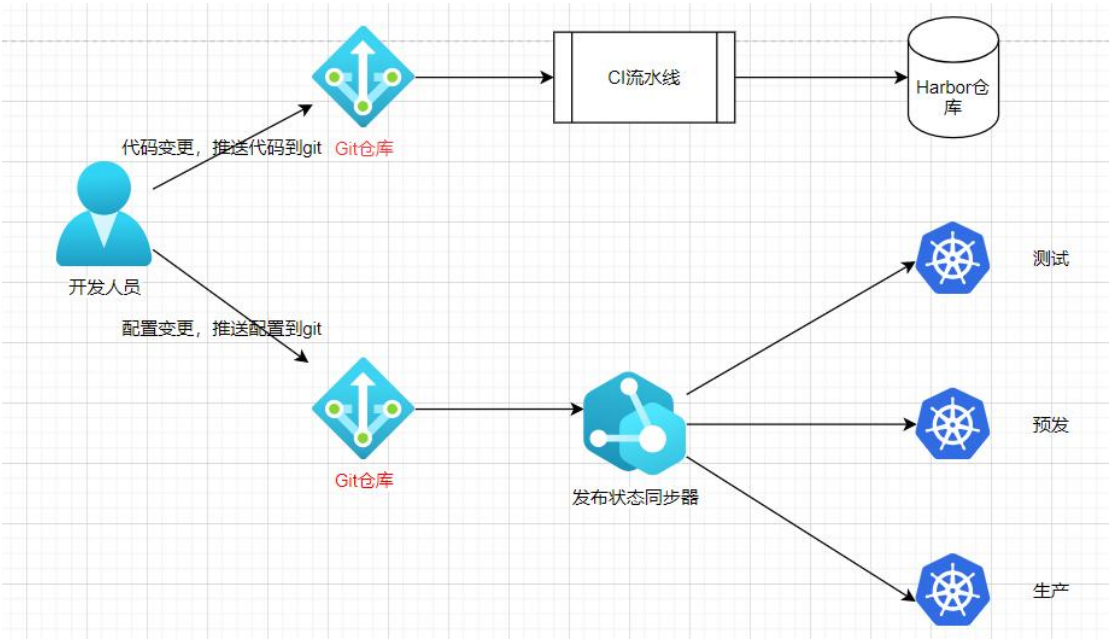
最后用一张图来总结下 Tekton 结合 Argo CD 来实现 GitOps 的工作流：



2、什么是 GitOps

GitOps 是以 Git 为基础，使用 CI/CD 来更新运行在云原生环境的应用，它秉承了 DevOps 的核心理念“构建它并交付它（you built it you ship it）”。

概念说起来有点虚，我画了张图，看了就明白了。



1、当开发人员将开发完成的代码推送到 git 仓库会触发 CI 制作镜像并推送到镜像仓库。

2、CI 处理完成后，可以手动或者自动修改应用配置，再将其推送到 git 仓库。

3、GitOps 会同时对比目标状态和当前状态，如果两者不一致会触发 CD 将新的配置部署到集群中。

其中，目标状态是 Git 中的状态，现有状态是集群的里的应用状态。

不用 GitOps 可以么？

当然可以，我们可以使用 kubectl、helm 等工具直接发布配置，但这会存在一个很严重的安全问题，那就是密钥共享。

为了让 CI 系统能够自动的部署应用，我们需要将集群的访问密钥共享给它，这会带来潜在的安全问题。

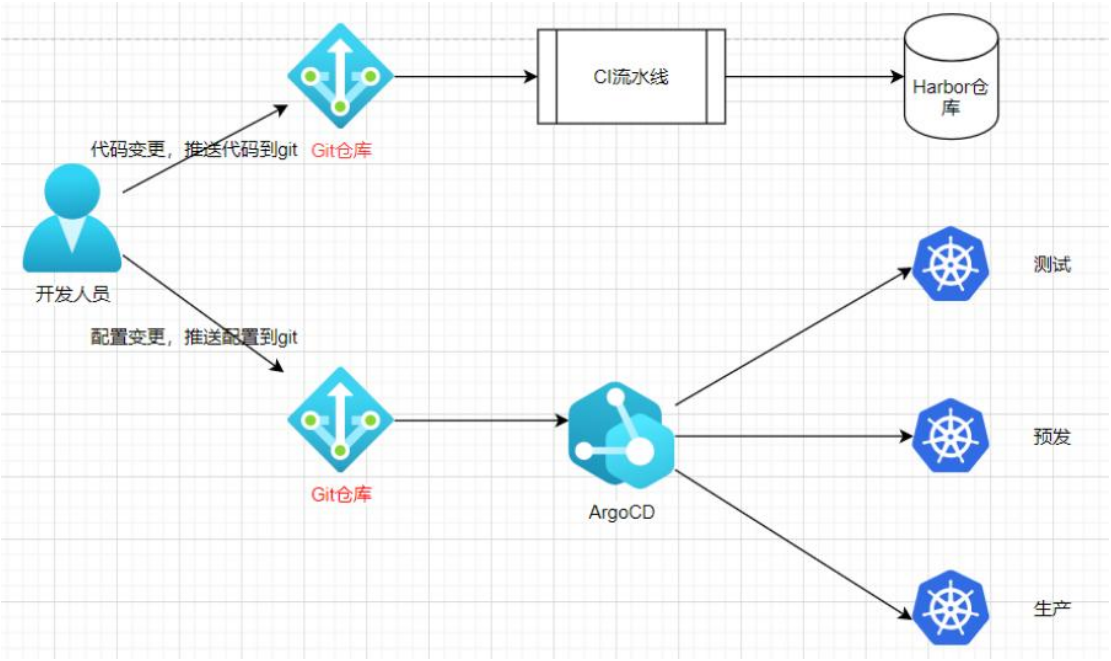
3、Argo CD 功能介绍

Argo CD 遵循 GitOps 模式，使用 Git 存储库存储所需应用程序的配置。

Argo CD 实现为 kubernetes 控制器，它持续监视运行中的应用程序，并将当前的活动状态与期望的目标状态进行比较（如 Git repo 中指定的那样）。如果已部署的应用程序的活动状态偏离了目标状态，则认为是 OutOfSync。Argo CD 报告和可视化这些差异，同时提供了方法，可以自动或手动将活动状态同步回所需的目标状态。在 Git repo 中对所需目标状态所做的任何修改都可以自动应用并反映到指定的目标环境中。

Argo CD 处在如下位置：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。



它的优势总结如下：

- 应用定义、配置和环境信息是声明式的，并且可以进行版本控制。
- 应用部署和生命周期管理是全自动化的，是可审计的，清晰易懂。
- Argo CD 是一个独立的部署工具，支持对多个环境、多个 Kubernetes 集群上的应用进行统一部署和管理。

4、Argo CD 应用实战

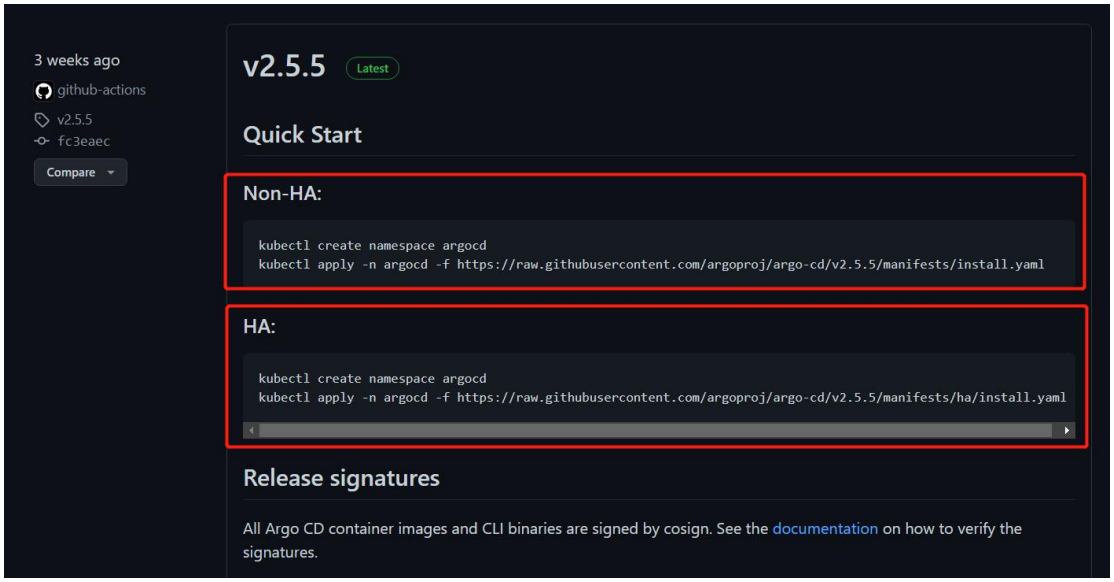
4.1、基于 Kubernetes 部署 Argo CD

github 地址：

https://github.com/argoproj/argo-cd/releases

这里我们部署 v2.5.5 最新版本，这里有两种部署方式，一种是非 HA 方式部署，一种是 HA 方式部署。显然 HA 会更占资源，但是会更稳定。

相关的部署命令，官方已经提供，如下图：



这里演示 Non-HA 方式部署。我们已经讲了这么多了，相信大家部署这种东西已经再简单不过了。

注意：如果要部署 HA，node 工作节点至少要 3 个以上。

（1）以 Non-HA 方式部署 argocd

```
# 创建 argocd 名称空间
[root@k8s-master01 ~]# kubectl create namespace argocd
namespace/argocd created

# 部署 argocd 应用到 argocd 名称空间下
[root@k8s-master01 ~]# wget https://raw.githubusercontent.com/argoproj/argo-cd/v2.5.5/manifests/install.yaml
[root@k8s-master01 ~]# kubectl apply -n argocd -f install.yaml
```

（2）argocd pod 状态

```
[root@k8s-master01 ~]# kubectl get pods -n argocd
[root@k8s-master01 ~]# kubectl get pods -n argocd
NAME                                                READY   STATUS    RESTARTS   AGE
argocd-application-controller-0                     1/1     Running   0          101s
argocd-applicationset-controller-7f78755d6f-7r98h   1/1     Running   0          101s
argocd-dex-server-765fd57989-z7552                  1/1     Running   0          101s
argocd-notifications-controller-7bc647c65d-fr79d    1/1     Running   0          101s
argocd-redis-547f5d94cd-nqqm5                       1/1     Running   0          101s
argocd-repo-server-74b6bdf8b8-7f2vr                 1/1     Running   0          101s
argocd-server-7467749665-xbs6l                      1/1     Running   0          101s
```

（3）访问 Argo server

访问 Argo server 的方式有两种：1、通过 web ui。2、使用 argocd 客户端

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

工具。我这里直接使用 web ui 进行管理。

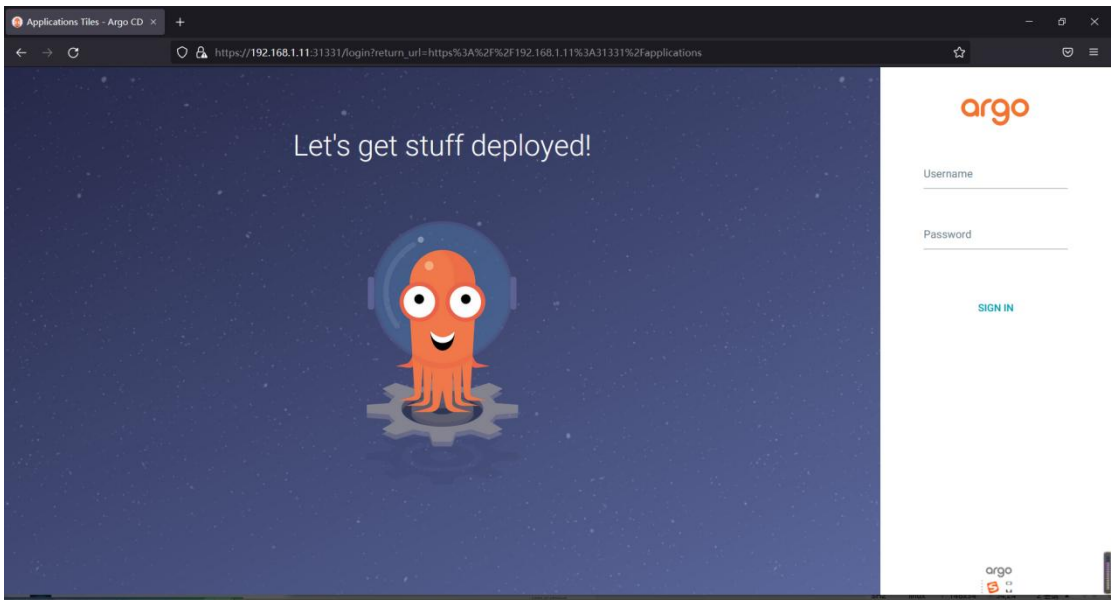
修改 argocd-server 为 NodePort 暴露方式：

```
[root@k8s-master01 ~]# kubectl edit svc argocd-server -n argocd
  type: NodePort

[root@k8s-master01 ~]# kubectl get svc -n argocd
```

```
[root@k8s-master01 ~]# kubectl get svc -n argocd
NAME                                      TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)                      AGE
argocd-applicationset-controller          ClusterIP   10.10.70.13     <none>        7000/TCP,8080/TCP            2m59s
argocd-dex-server                         ClusterIP   10.10.30.104    <none>        5556/TCP,5557/TCP,5558/TCP   2m59s
argocd-metrics                            ClusterIP   10.10.140.14    <none>        8082/TCP                     2m59s
argocd-notifications-controller-metrics   ClusterIP   10.10.58.184    <none>        9001/TCP                     2m59s
argocd-redis                              ClusterIP   10.10.79.90     <none>        6379/TCP                     2m59s
argocd-repo-server                        ClusterIP   10.10.29.163    <none>        8081/TCP,8084/TCP            2m59s
argocd-server                             NodePort    10.10.152.137   <none>        80:30866/TCP,443:30487/TCP   2m59s
argocd-server-metrics                     ClusterIP   10.10.47.168    <none>        8083/TCP                     2m59s
```

浏览器访问“https://IP:30866”访问 Argo server web ui。



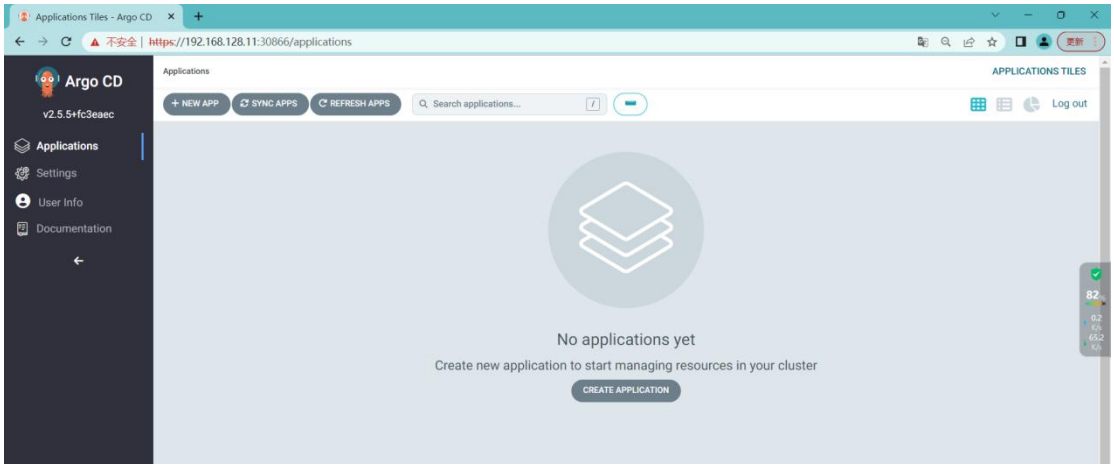
登陆账号为 admin。密码通过如下命令获取：

```
[root@k8s-master01 ~]# kubectl -n argocd get secret argocd-initial-admin-secret -o template="{{ .data.password | base64decode }}"
97XCyy3lnoPTuU46
```

默认情况下安装好的 Argo CD 会启用基于 Basic Auth 的身份校验，我们可以在 Secret 资源中找到对应的密码。但需要注意的是这个名字为 argocd-initial-admin-secret 的 sercret 资源是等到 Pod 处于 Running 状态后才会写入。

登陆成功：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。



4.2、修改 Web Ui 密码

默认密码我们只能查看 sercret 资源获取。我们可以使用客户端工具修改密码，也可以在 web ui 界面进行密码的修改。

- 方法一：客户端工具修改密码

（1）下载客户端，在 github 上下载即可，地址为：
https://github.com/argoproj/argo-cd/releases

（2）下载好之后，上传到服务器，添加执行权限：

```
[root@k8s-master01 ~]# ll argocd-linux-amd64
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
-rw-r--r-- 1 root root 131412372 8月  10 11:24 argocd-linux-amd64
[root@k8s-master01 ~]# chmod +x argocd-linux-amd64
[root@k8s-master01 ~]# mv argocd-linux-amd64 /usr/local/bin/argocd
```

（3）用客户端登录

```
[root@k8s-master01 ~]# argocd login 192.168.128.11:30866
```

```
[root@k8s-master01 ~]# argocd login 192.168.128.11:30866
WARNING: server certificate had error: x509: cannot validate certificate for 192.168.128.11 because it doesn't contain any IP SANs. Proceed insecurely (y/n)?
 y
Username: admin
Password:
'admin:login' logged in successfully
Context '192.168.128.11:30866' updated
```

（4）修改密码

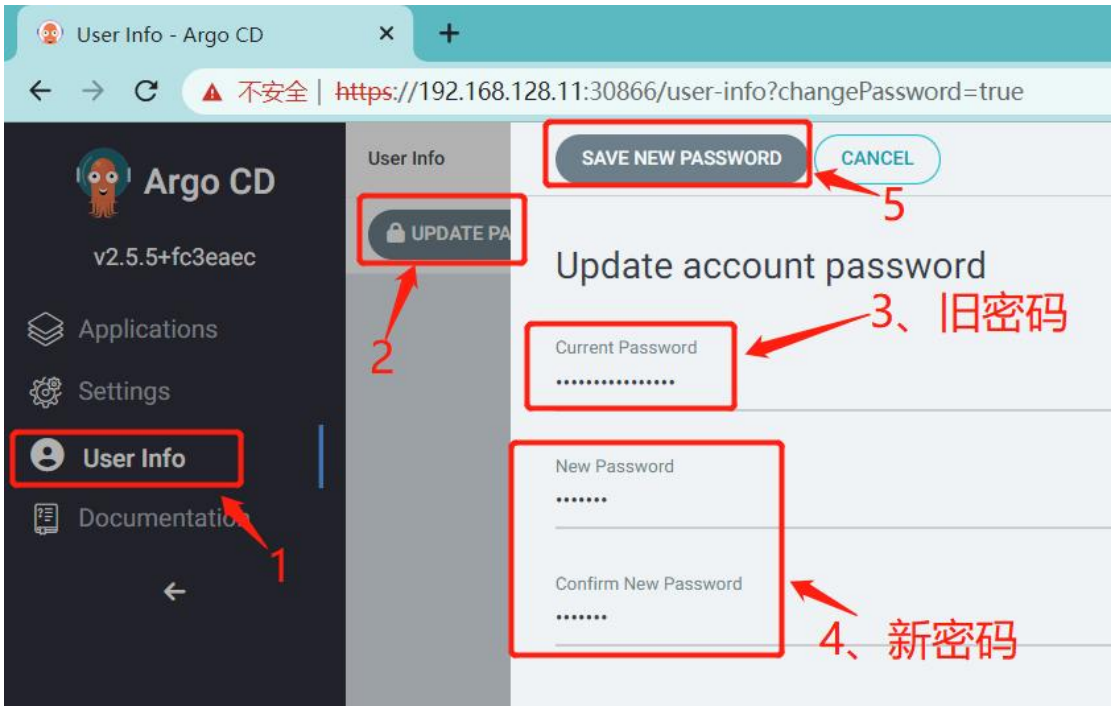
```
[root@k8s-master01 ~]# argocd account update-password \
--account admin \
--current-password 97XCyy3lnoPTuU46 \
--new-password zhangyf123456..
Password updated
Context '192.168.128.11:30866' updated

参数说明：
--account：指定要修改密码的用户名
--current-password：旧密码
--new-password：新密码
```

- 方法二：web ui 界面进行密码的修改

“User Info”->“UPDATE PASSWORD”->“填写旧密码和新密码”->“SAVE NEW PASSWORD”，具体操作如下图：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

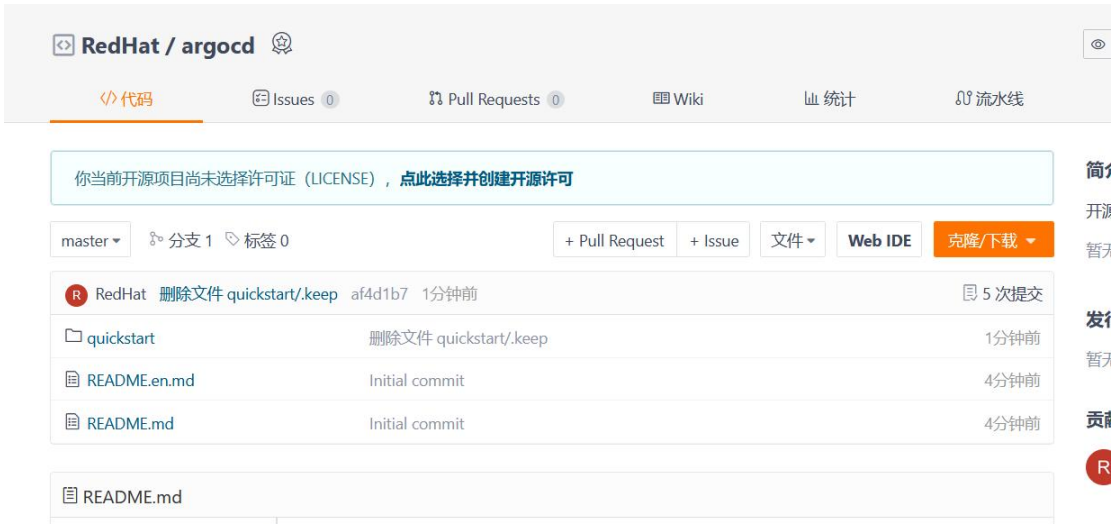


注意：密码必须为 8-32 位。

4.3、准备 Git 仓库

在码云上创建项目，取名为 argocd，为了方便实验将仓库设置为开源仓库。在仓库中创建 quickstart 目录，在目录中创建两个 yaml 资源文件，分别是 myapp-deployment.yaml 和 myapp-service.yaml。

图一：



图二：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

myapp-deployment.yaml 文件内容：

```
    apiVersion: apps/v1
    kind: Deployment
    metadata:
      name: myapp
    spec:
      replicas: 1
      selector:
        matchLabels:
          app: myapp
      template:
        metadata:
          labels:
            app: myapp
        spec:
          containers:
          -                                                    image:
registry.cn-shanghai.aliyuncs.com/public-namespace/myapp:v1
            name: myapp
            ports:
            - containerPort: 80
```

myapp-service.yaml 文件内容：

```
    apiVersion: v1
    kind: Service
    metadata:
      name: myapp
    spec:
      ports:
      - port: 80
        targetPort: 80
        nodePort: 32060
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
    type: NodePort
    selector:
      app: myapp
```

4.4、创建 Argo CD App

首先创建一个命名空间 devops 用于 Argo CD 部署应用。

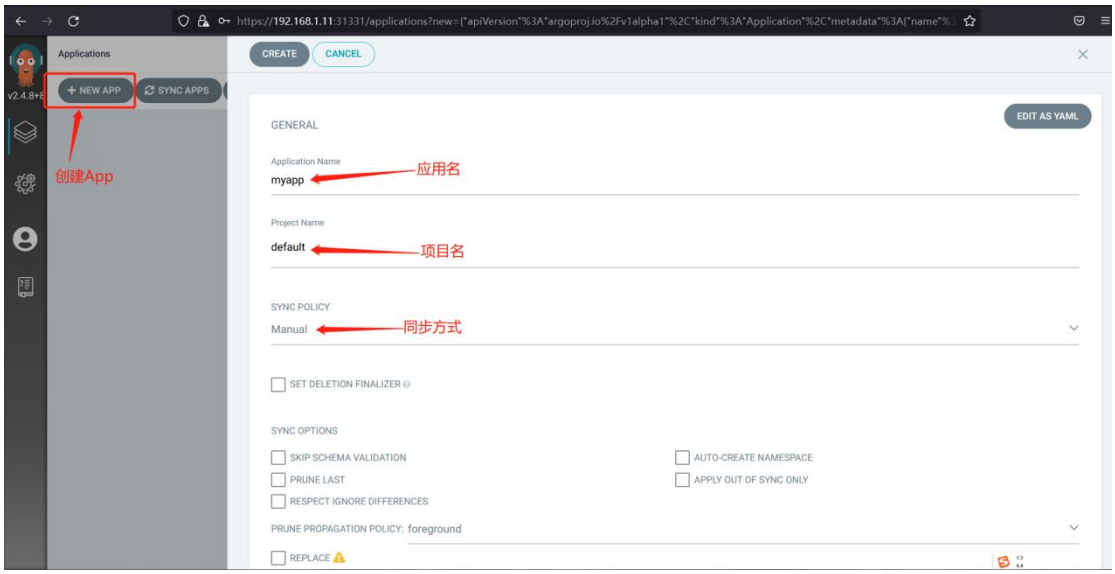
```
[root@k8s-master01 ~]# kubectl create ns devops
namespace/devops created
```

可以有三种方式创建 app，分别介绍如下：

4.1.1、使用 UI 创建 App

（1）创建 New App

“Applications” -> “NEW APP”，具体操作如下图：



Application Name：myapp

Project Name：default

SYNC POLICY：Manual

说明：

Application Name：自定义的应用名。

Project Name：使用默认创建好的 default 项目。

SYNC POLICY：同步方式，可以选择自动或者手动，这里我们选择手动同步。

往下翻：

Repository URL：https://gitee.com/redhat_feng/argocd.git
Revision：master
Path：quickstart

说明：

Repository URL：项目的 Git 地址。
Revision：分支名。
Path：yaml 资源文件所在的相对路径。

再往下翻：

Cluster URL：https://kubernetes.default.svc
Namespace：devops

说明：

Cluster URL：Kubernetes API Server 的访问地址，由于 Argo CD 和下发应用的 Kubernetes 集群是同一个，因此可以直接使用 https://kubernetes.default.svc 来访问。关于 Kubernetes 中 DNS 解析规则可以查看 Pod 与 Service 的 DNS。

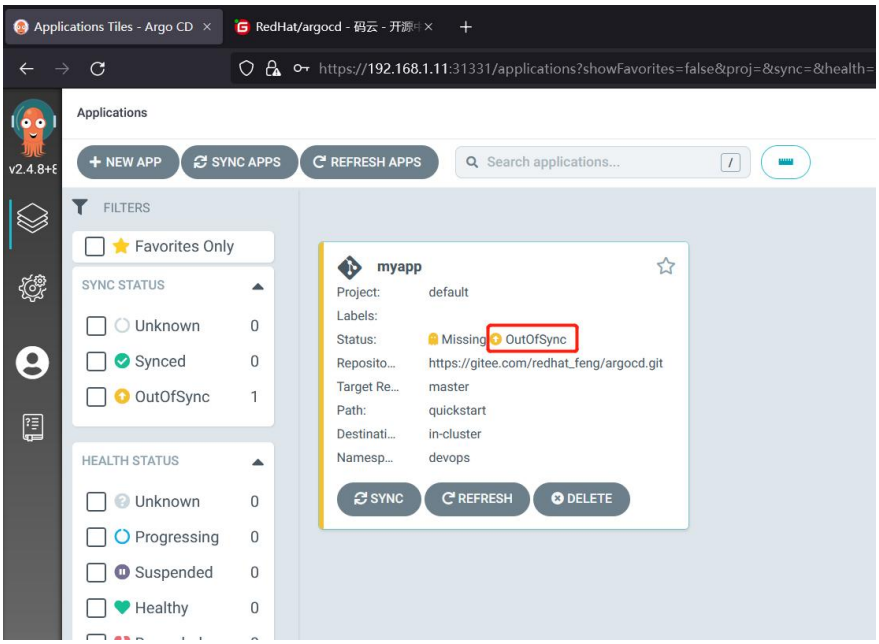
Namespace：部署应用的命名空间。

配置完成之后，选择“create”创建。

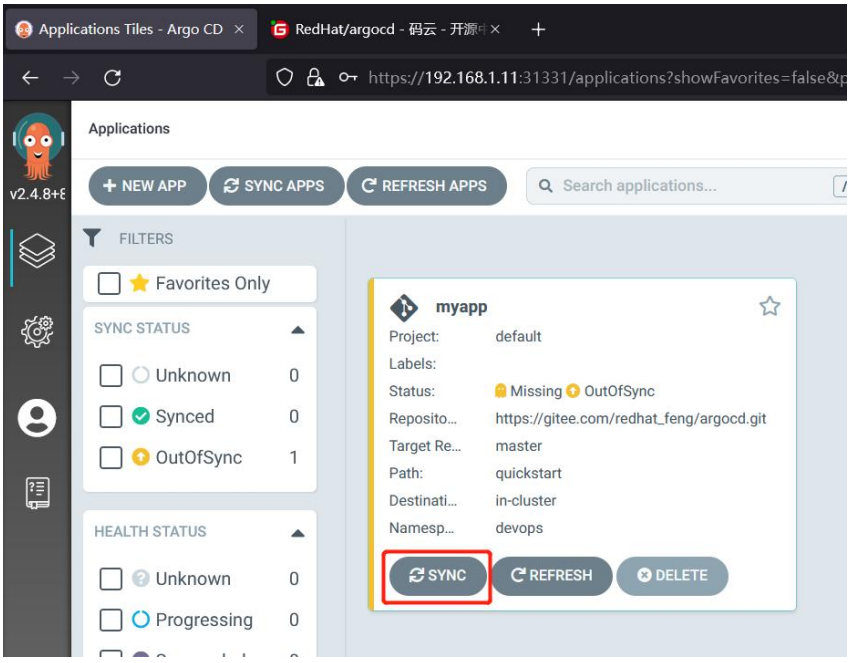
（2）同步 Argo cd 状态，创建 pod

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

创建完成后如下图所示，此时处于 OutOfSync 的状态：

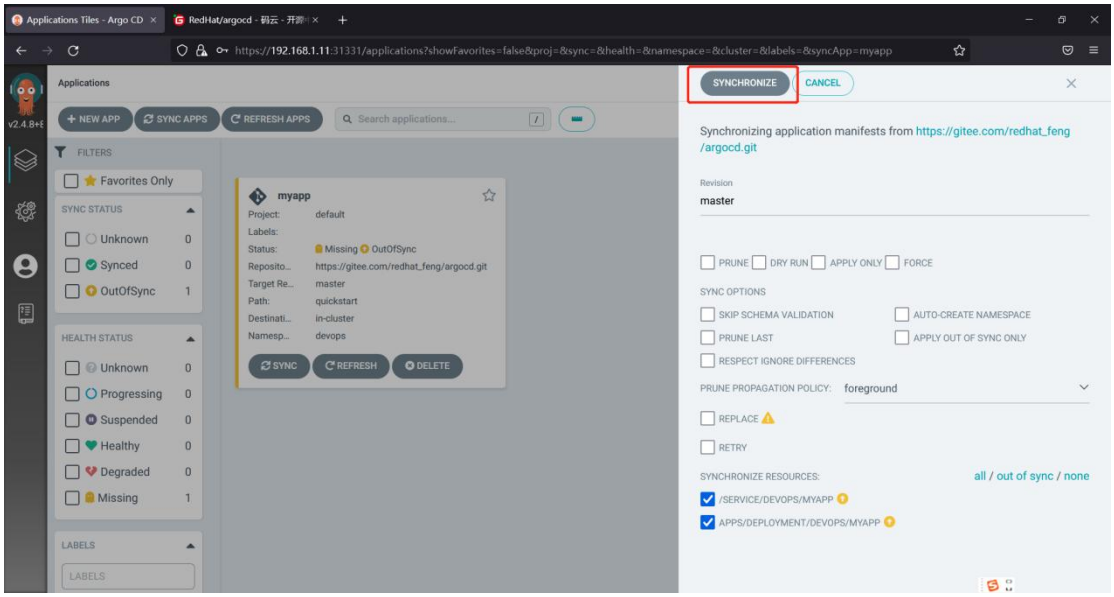


由于设置的是手动同步，因此需要点一下下面的 SYNC 进行同步：

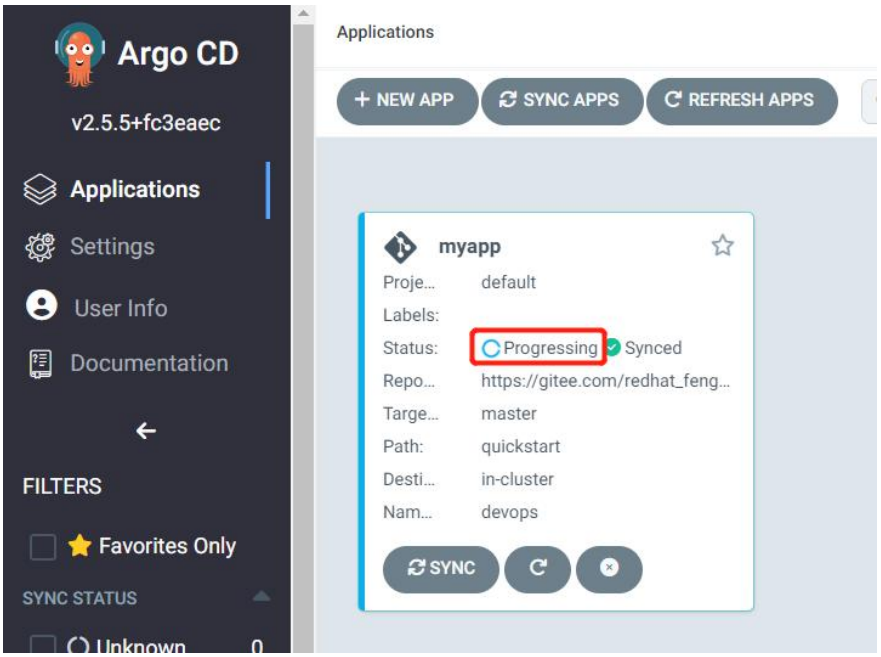


在弹出框点击 SYNCHRONIZE，确认同步：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

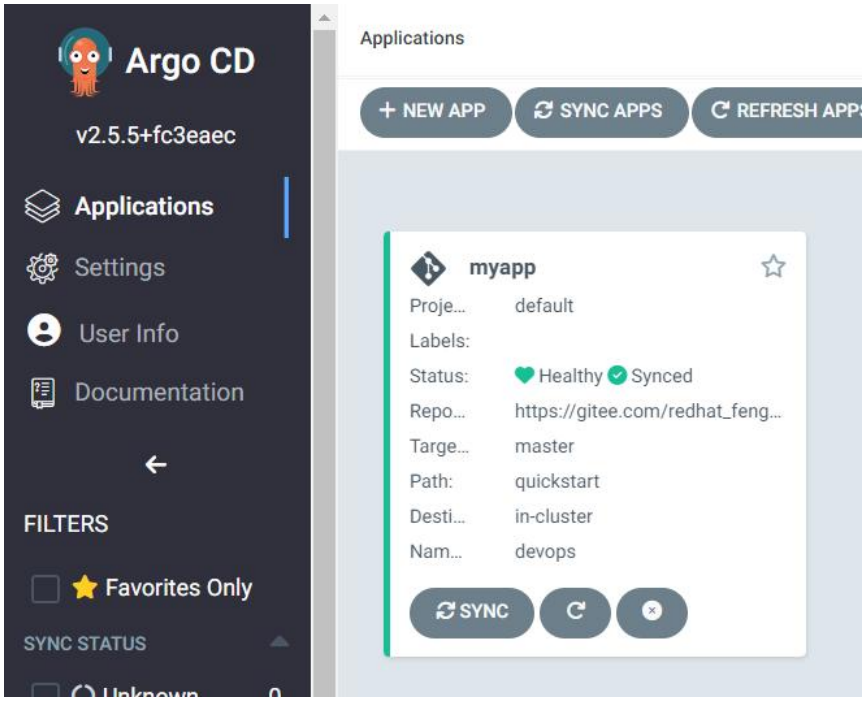


等待同步完成：

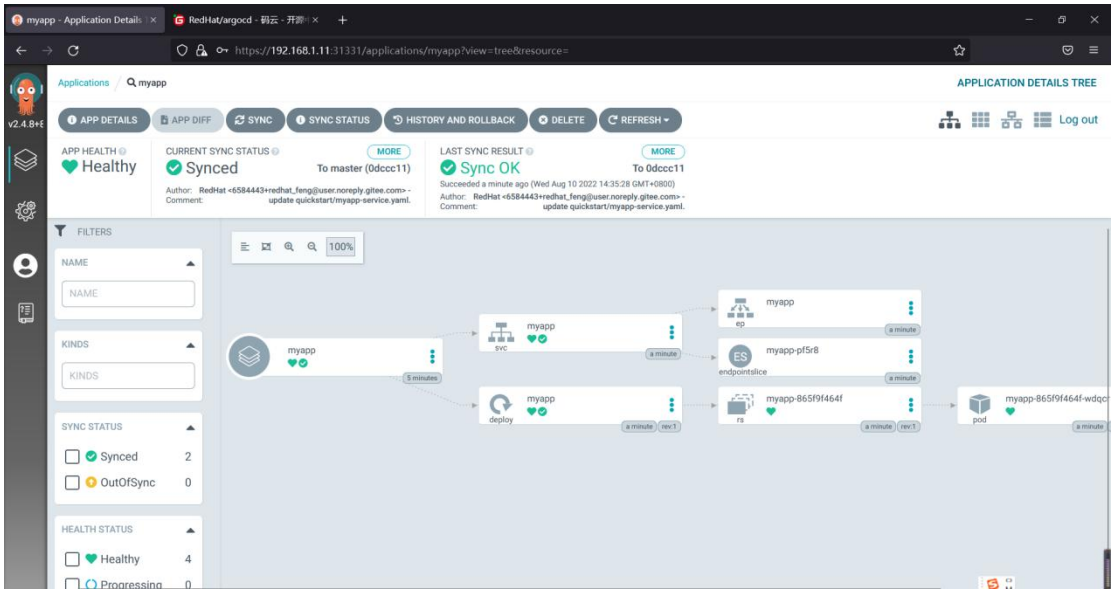


同步完成：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。



在 Argo CD 上点击“myapp”应用进入查看详情，如下图：



底层检查：

```
[root@k8s-master01 ~]# kubectl get pods -n devops
```

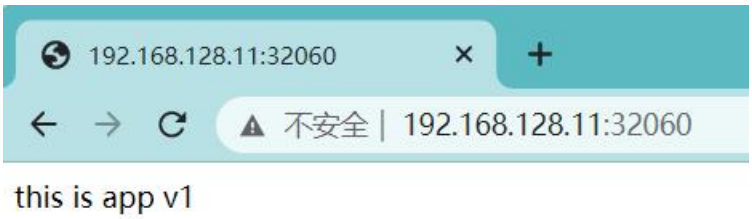
```
[root@k8s-master01 ~]# kubectl get pods -n devops
NAME                     READY   STATUS    RESTARTS   AGE
myapp-7cb6d6cbc5-tvb4l   1/1     Running   0          3m33s
```

```
[root@k8s-master01 ~]# kubectl get svc -n devops
```

```
[root@k8s-master01 ~]# kubectl get svc -n devops
NAME    TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
myapp   NodePort   10.10.224.41   <none>        80:32060/TCP   4m2s
```

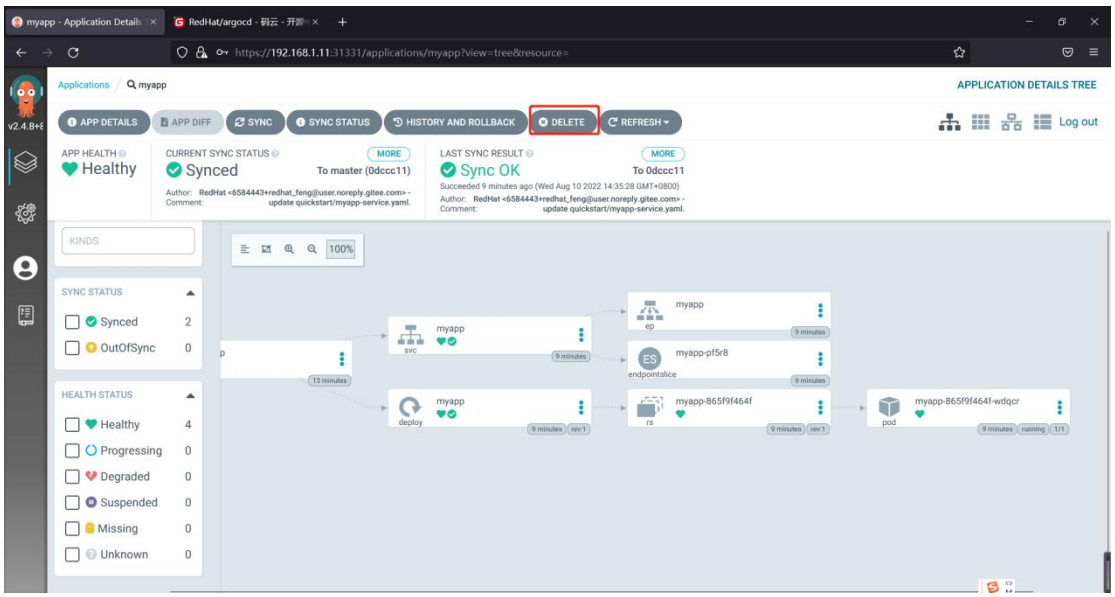
访问测试：

浏览器输入“http://192.168.128.11:32060/”访问：

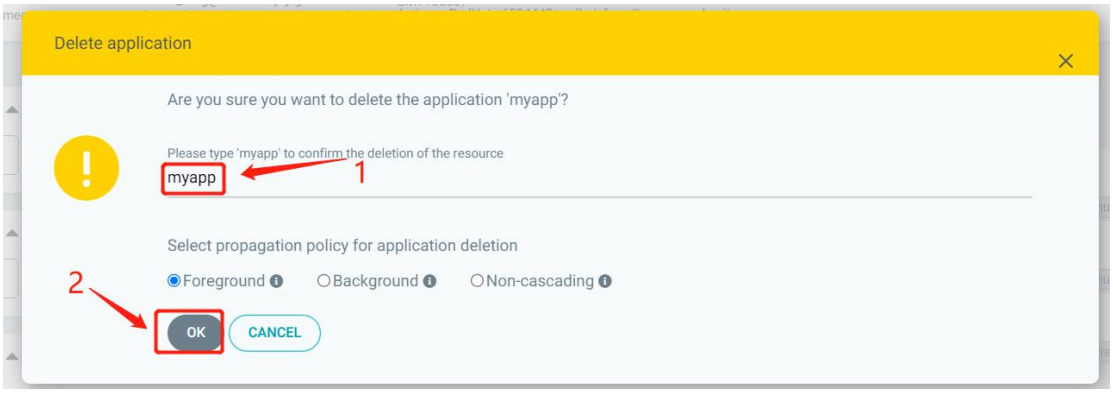


（3）删除 App 资源

点击“DELETE”按钮



请输入“myapp”以确认资源的删除



底层检查：

```
[root@k8s-master01 ~]# kubectl get all -n devops
No resources found in devops namespace.
```

4.1.2、使用 CLI 创建 APP

（1）使用 CLI 命令行工具创建应用：

```
[root@k8s-master01 ~]# argocd app create myapp \
--repo https://gitee.com/redhat_feng/argocd.git \
--path quickstart \
--dest-server https://kubernetes.default.svc \
--dest-namespace devops
application 'myapp' created
```

开启自动同步：--sync-policy auto

（2）列出所有应用

```
[root@k8s-master01 ~]# argocd app list
```

```
[root@k8s-master01 ~]# argocd app list
NAME   CLUSTER                         NAMESPACE  PROJECT  STATUS     HEALTH   SYNCPOLICY  CONDITIONS  REPO                                    PATH        TARGET
myapp  https://kubernetes.default.svc  devops     default  OutOfSync  Missing  <none>      <none>      https://gitee.com/redhat_feng/argocd.git  quickstart
```

（3）查看指定应用的详细信息

```
[root@k8s-master01 ~]# argocd app get myapp
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
[root@k8s-master01 ~]# argocd app get myapp
Name:               myapp
Project:            default
Server:             https://kubernetes.default.svc
Namespace:          devops
URL:                https://192.168.1.11:31331/applications/myapp
Repo:               https://gitee.com/redhat_feng/argocd.git
Target:
Path:               quickstart
SyncWindow:         Sync Allowed
Sync Policy:        <none>
Sync Status:        OutOfSync from  (0dccc11)
Health Status:      Missing

GROUP  KIND        NAMESPACE  NAME   STATUS     HEALTH   HOOK  MESSAGE
       Service     devops     myapp  OutOfSync  Missing
apps   Deployment  devops     myapp  OutOfSync  Missing
```

（4）进行同步

```
[root@k8s-master01 ~]# argocd app sync myapp
```

```
[root@k8s-master01 ~]# argocd app sync myapp
TIMESTAMP                  GROUP        KIND   NAMESPACE                  NAME    STATUS    HEALTH        HOOK  MESSAGE
2022-08-10T15:02:30+08:00            Service     devops                 myapp  OutOfSync  Missing
2022-08-10T15:02:30+08:00   apps  Deployment     devops                 myapp  OutOfSync  Missing
2022-08-10T15:02:30+08:00            Service     devops                 myapp  OutOfSync  Missing              service/myapp created
2022-08-10T15:02:30+08:00   apps  Deployment     devops                 myapp  OutOfSync  Missing              deployment.apps/myapp created

Name:               myapp
Project:            default
Server:             https://kubernetes.default.svc
Namespace:          devops
URL:                https://192.168.1.11:31331/applications/myapp
Repo:               https://gitee.com/redhat_feng/argocd.git
Target:
Path:               quickstart
SyncWindow:         Sync Allowed
Sync Policy:        <none>
Sync Status:        Synced to  (0dccc11)
Health Status:      Healthy

Operation:          Sync
Sync Revision:      0dccc113983746b67c9fb09208bd1d27e68081a6
Phase:              Succeeded
Start:              2022-08-10 15:02:30 +0800 CST
Finished:           2022-08-10 15:02:30 +0800 CST
Duration:           0s
Message:            successfully synced (all tasks run)

GROUP  KIND        NAMESPACE  NAME   STATUS  HEALTH   HOOK  MESSAGE
       Service     devops     myapp  Synced  Healthy        service/myapp created
apps   Deployment  devops     myapp  Synced  Healthy        deployment.apps/myapp created
```

```
# 再次查看应用的详细信息
[root@k8s-master01 ~]# argocd app get myapp
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
[root@k8s-master01 ~]# argocd app get myapp
Name:               myapp
Project:            default
Server:             https://kubernetes.default.svc
Namespace:          devops
URL:                https://192.168.1.11:31331/applications/myapp
Repo:               https://gitee.com/redhat_feng/argocd.git
Target:
Path:               quickstart
SyncWindow:         Sync Allowed
Sync Policy:        <none>
Sync Status:        Synced to  (0dccc11)
Health Status:      Healthy

GROUP  KIND        NAMESPACE  NAME   STATUS  HEALTH   HOOK  MESSAGE
       Service     devops     myapp  Synced  Healthy        service/myapp created
apps   Deployment  devops     myapp  Synced  Healthy        deployment.apps/myapp created
```

（5）查看 pod：

```
[root@k8s-master01 ~]# kubectl get pods -n devops
NAME                     READY   STATUS    RESTARTS   AGE
myapp-7cb6d6cbc5-2n6np   1/1     Running   0          23s
[root@k8s-master01 ~]# kubectl get svc -n devops
NAME    TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
myapp   NodePort   10.10.151.251   <none>        80:32060/TCP   33s
```

浏览器访问测试：

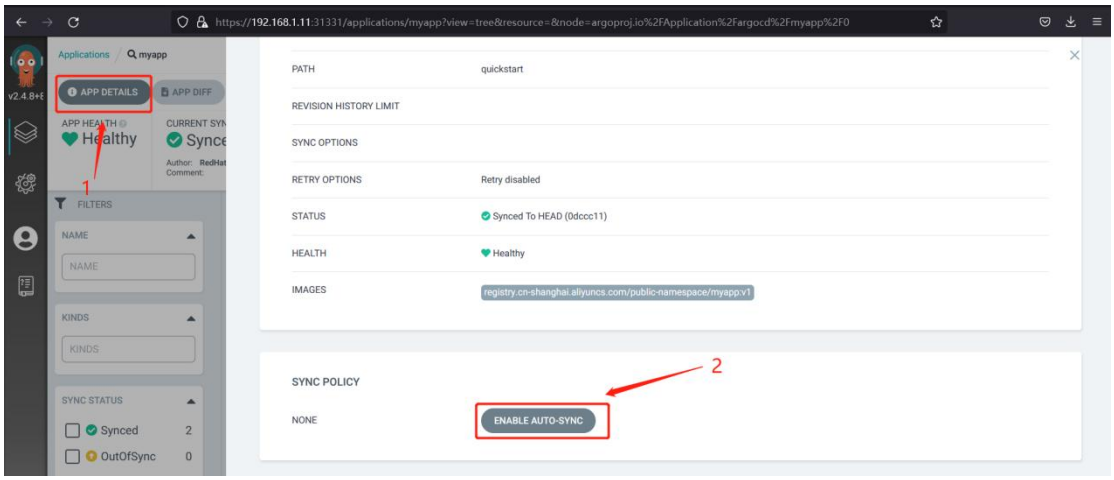
（6）删除 App（可以暂时不执行，后面我们要做版本升级和回滚）

```
[root@k8s-master01 ~]# argocd app delete myapp
Are you sure you want to delete 'myapp' and all its resources? [y/n]
y
application 'myapp' deleted
```

4.5、版本升级

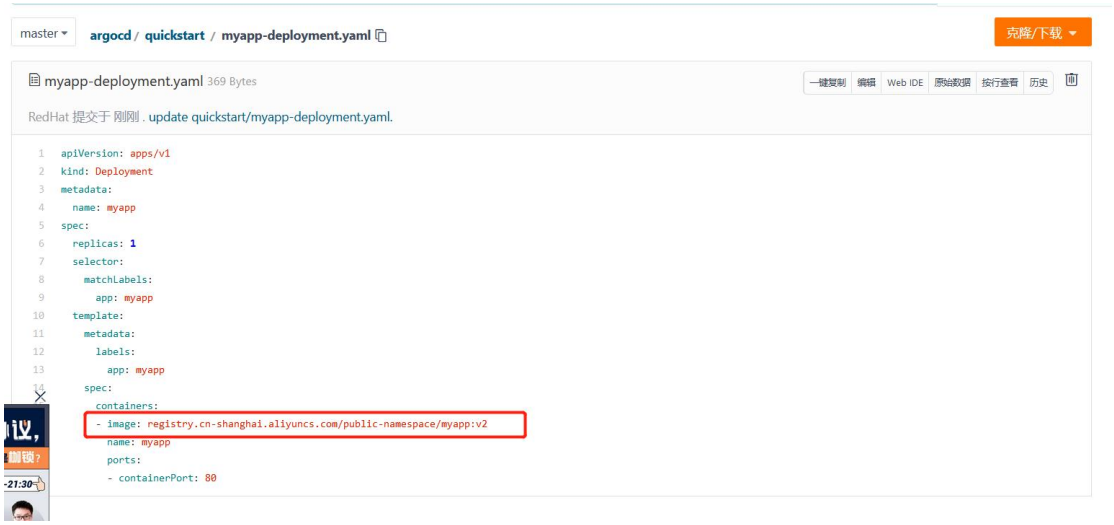
（1）修改应用为自动同步

将 myapp 应用从手动同步改成自动同步。点击 APP DETAILS->SYNC POLICY，点击 ENABLE AUTO-SYNC：

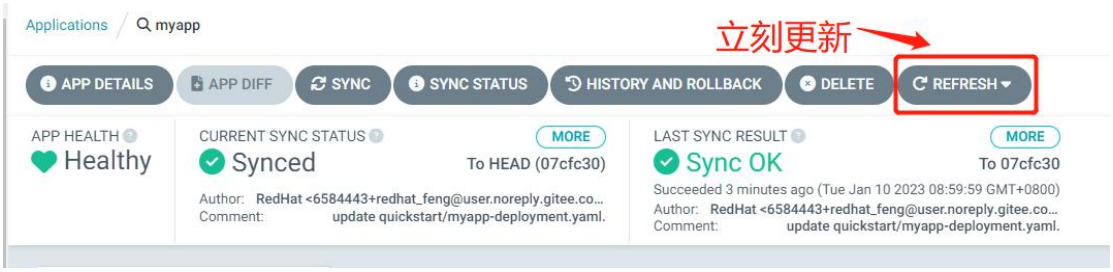


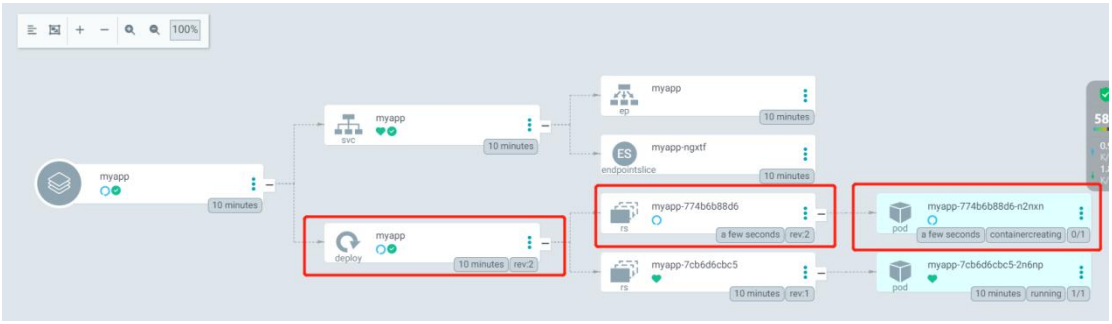
（2）版本升级

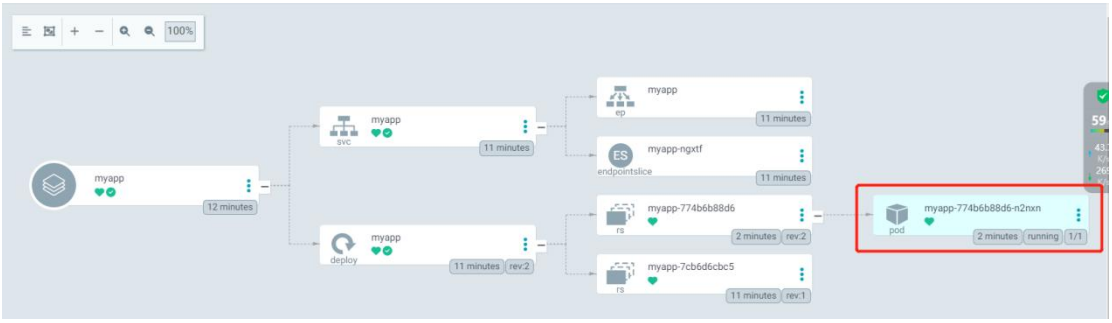
在代码仓库更新 myapp-deployment.yaml 文件：



等待一会 Argo CD 会自动更新应用，如果等不及可以点击 Refresh，Argo CD 会去立即获取最新的资源文件。可以看到此时 myapp Deployment 会新创建 v2 版本的 Replicaset，v2 版本的 Replicaset 会创建并管理 v2 版本的 Pod。





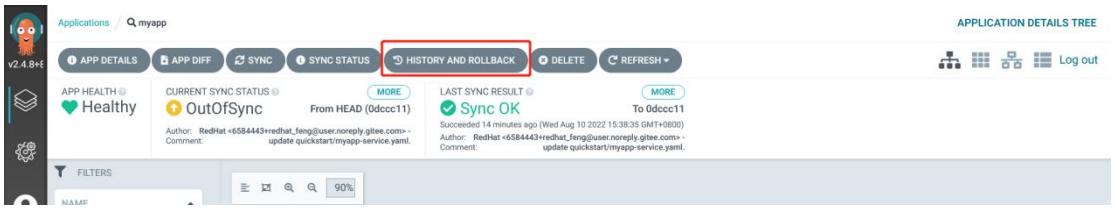


测试访问：

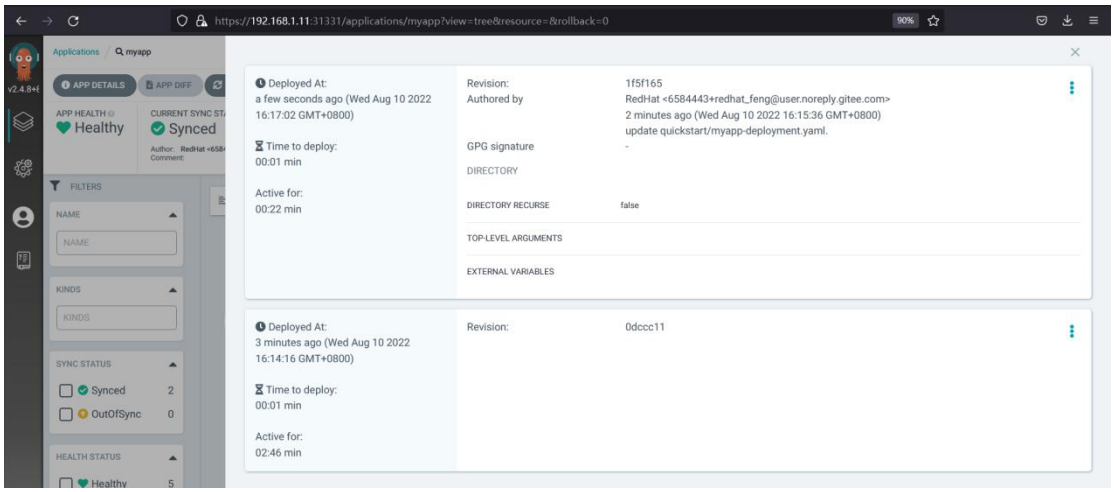
4.6、版本回滚

升级到 v2 版本以后，v1 版本的 Replicaset 并没有被删除，而是继续保留，这是为了方便我们回滚应用。在 myapp 应用中点击 HISTORY AND ROLLBACK 查看历史记录，可以看到有 2 个历史记录：

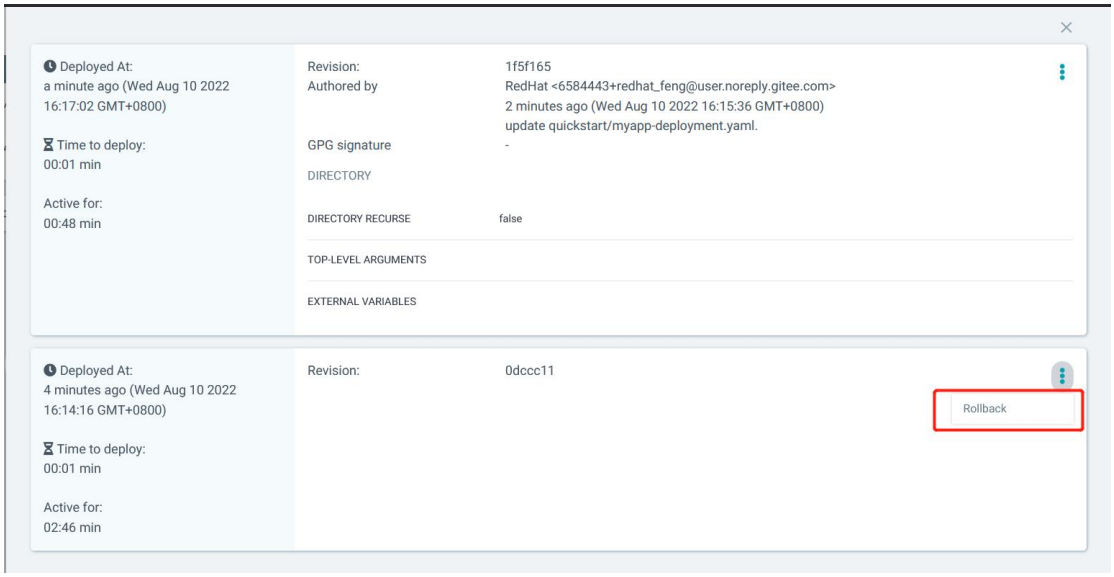
第一步：



第二步：

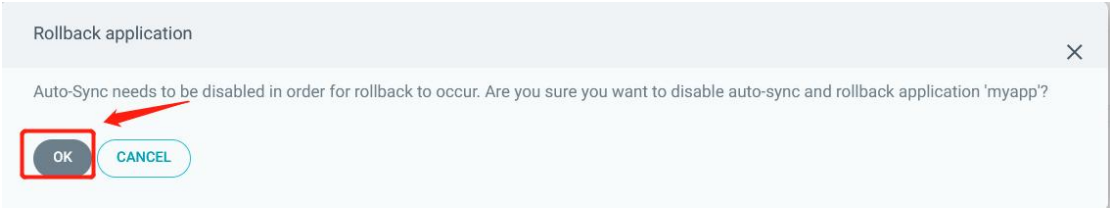


假设刚刚上线的 v2 版本出现了问题，需要回滚回 v1 版本，那么可以选中 v1 版本，然后点击 Rollback 进行回滚：

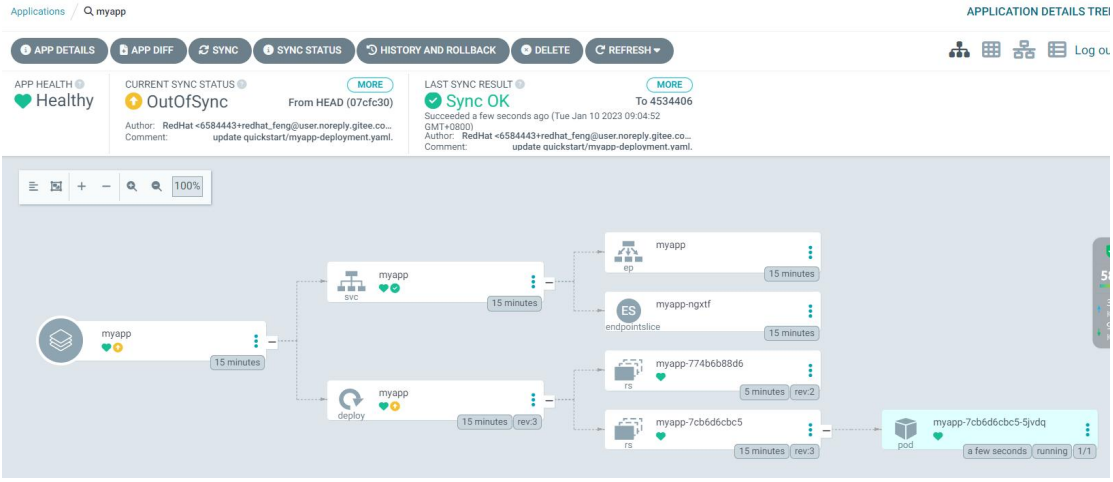


在回滚的时候需要禁用 AUTO-SYNC 自动同步，点击 OK 确认即可：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。



等待一会可以看到此时已经回滚成功，此时 Pod 是 v1 版本的，并且由于此时线上的版本并不是 Git 仓库中最新的版本，因此此时同步状态是 OutOfSync：



浏览器访问测试：

那么以上就是 Argo CD 的应用，Argo CD 的功能仅在可持续交付、可持续部署、业务回滚功能这一块。通常结合 Tekton 来进行应用。