

# 持续部署 快速入门 产品文档



腾讯云

**【版权声明】**

©2013-2024 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

**【商标声明】**

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

**【服务声明】**

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

# 快速入门

最近更新时间：2024-01-03 11:38:04

本文为您详细介绍在 CODING 中持续部署的基本操作。

## 前提条件

使用 CODING 项目管理的前提是，您的腾讯云账号需要开通 CODING DevOps 服务。

## 进入项目

1. 登录 CODING 控制台，单击**立即使用**进入 CODING 使用页面。
2. 单击工作台首页左侧的



，进入持续部署控制台。

## 功能介绍

CODING 持续部署用于把控构建之后的项目发布与部署交付流程。能够无缝对接上游 Git 仓库、制品仓库以实现全自动化部署。在稳定的技术架构、运维工具等基础上，具备蓝绿发布，灰度发布（金丝雀发布），滚动发布，快速回滚等能力。

下文将以一个简单的 Demo 项目为例，演示如何使用 CODING 持续部署控制台将应用发布至腾讯云集群。

## 前置准备

开启持续部署部署设置权限。

一个可被 CODING 持续部署访问的 Kubernetes 集群，单击了解如何申请 [腾讯云标准集群](#)。

导入 [示例代码库](#)。

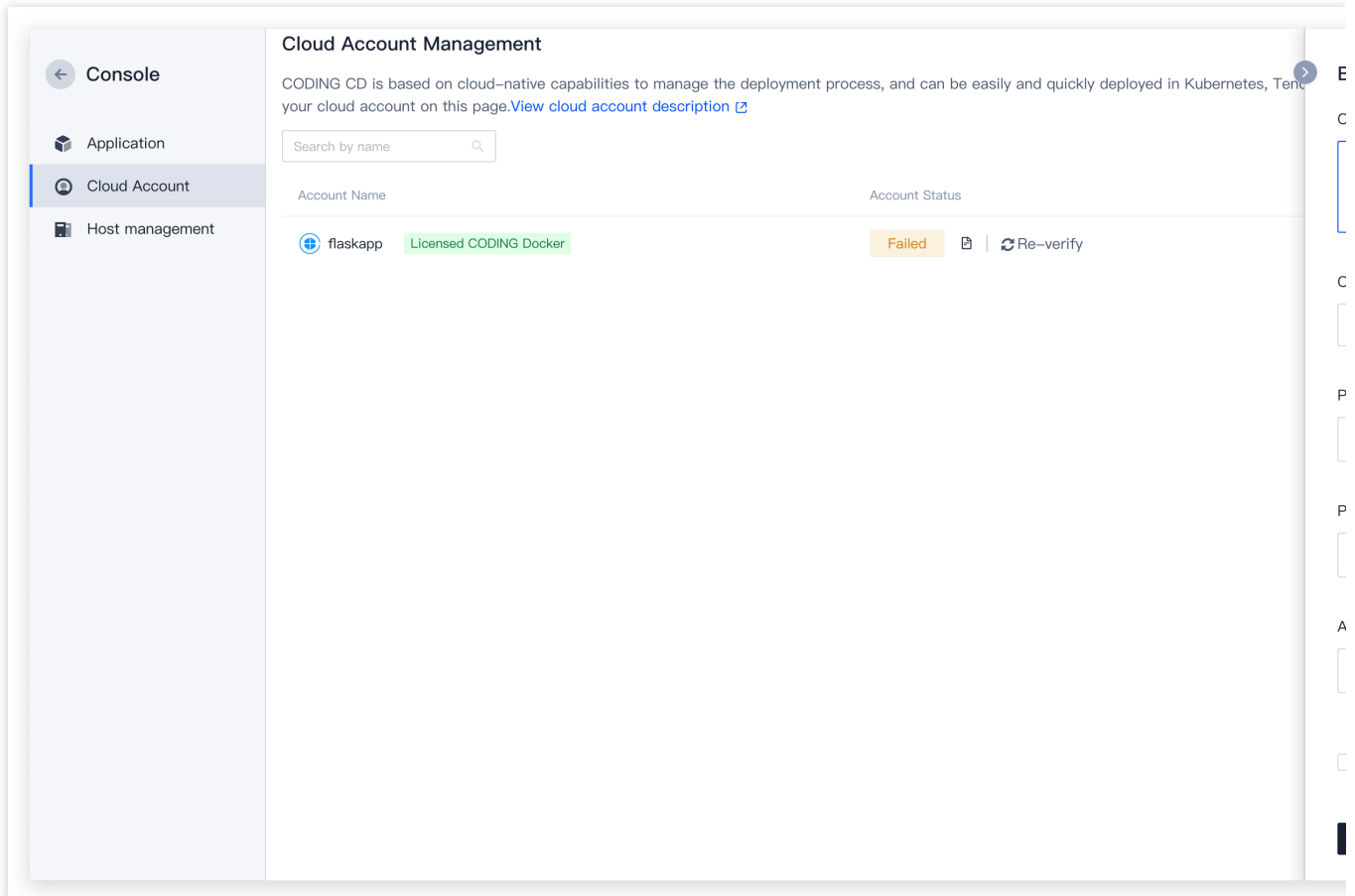
Docker 制品仓库，单击了解如何使用项目中的 [Docker 制品仓库](#)。

## 操作步骤

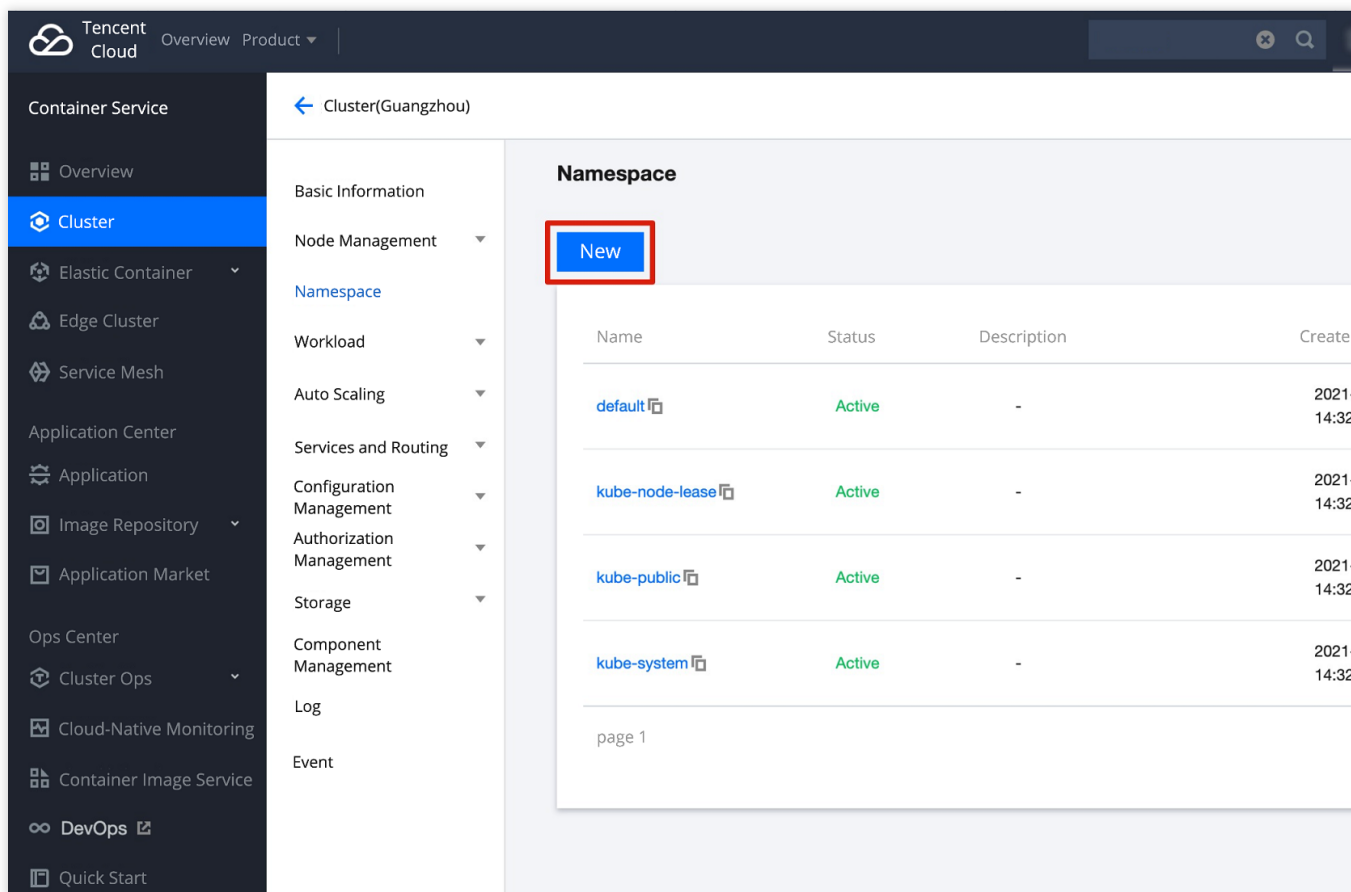
## 步骤1. 获取并关联云账号

因使用了腾讯云容器服务，部署后应用将发布至集群，本示例使用的团队账号已在**团队管理 > 服务集成**中关联腾讯云账号。

1. 单击首页左侧的**部署控制台**，在**云账号**中绑定腾讯云账号。云账号名称可以自拟，选择地域后将自动获取对应集群。

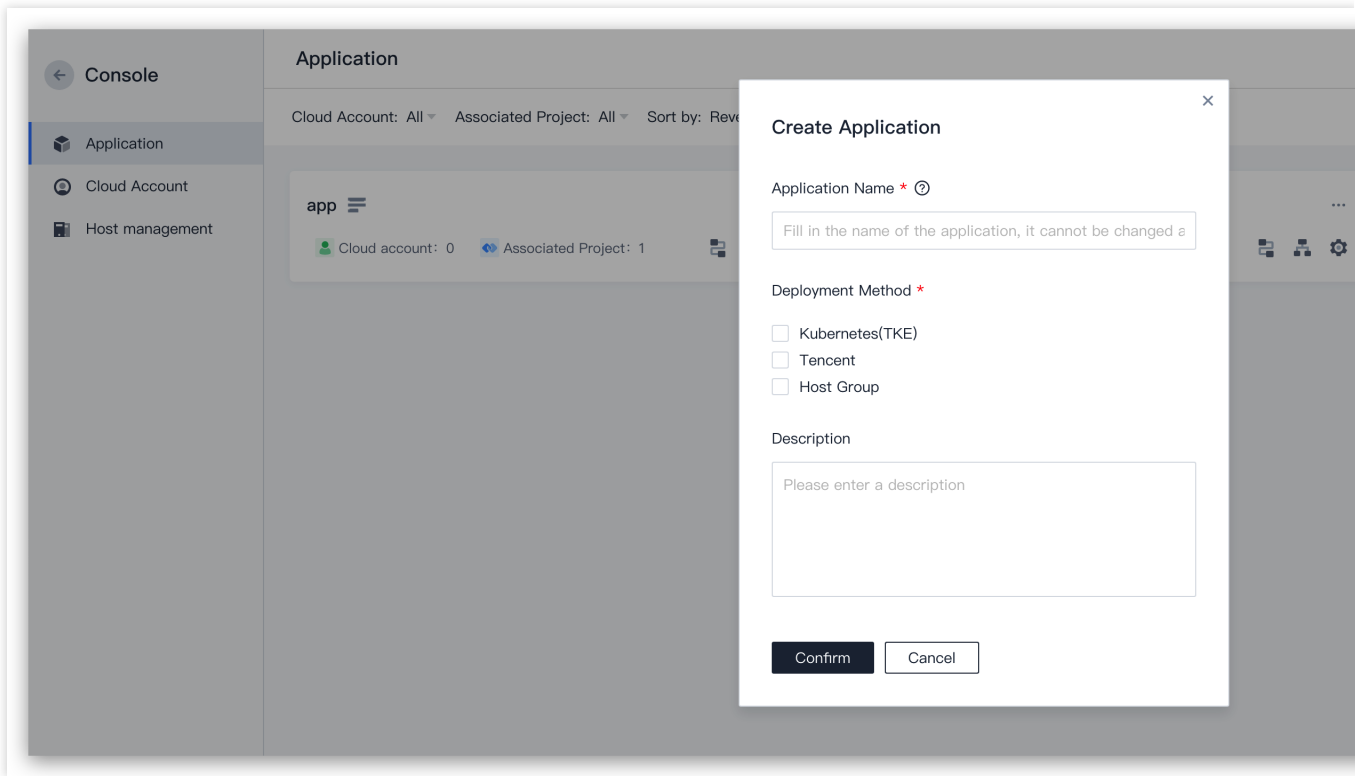


2. 自动生成的制品仓库访问凭证将存储于**命名空间**，您可以在腾讯云控制台中新建。

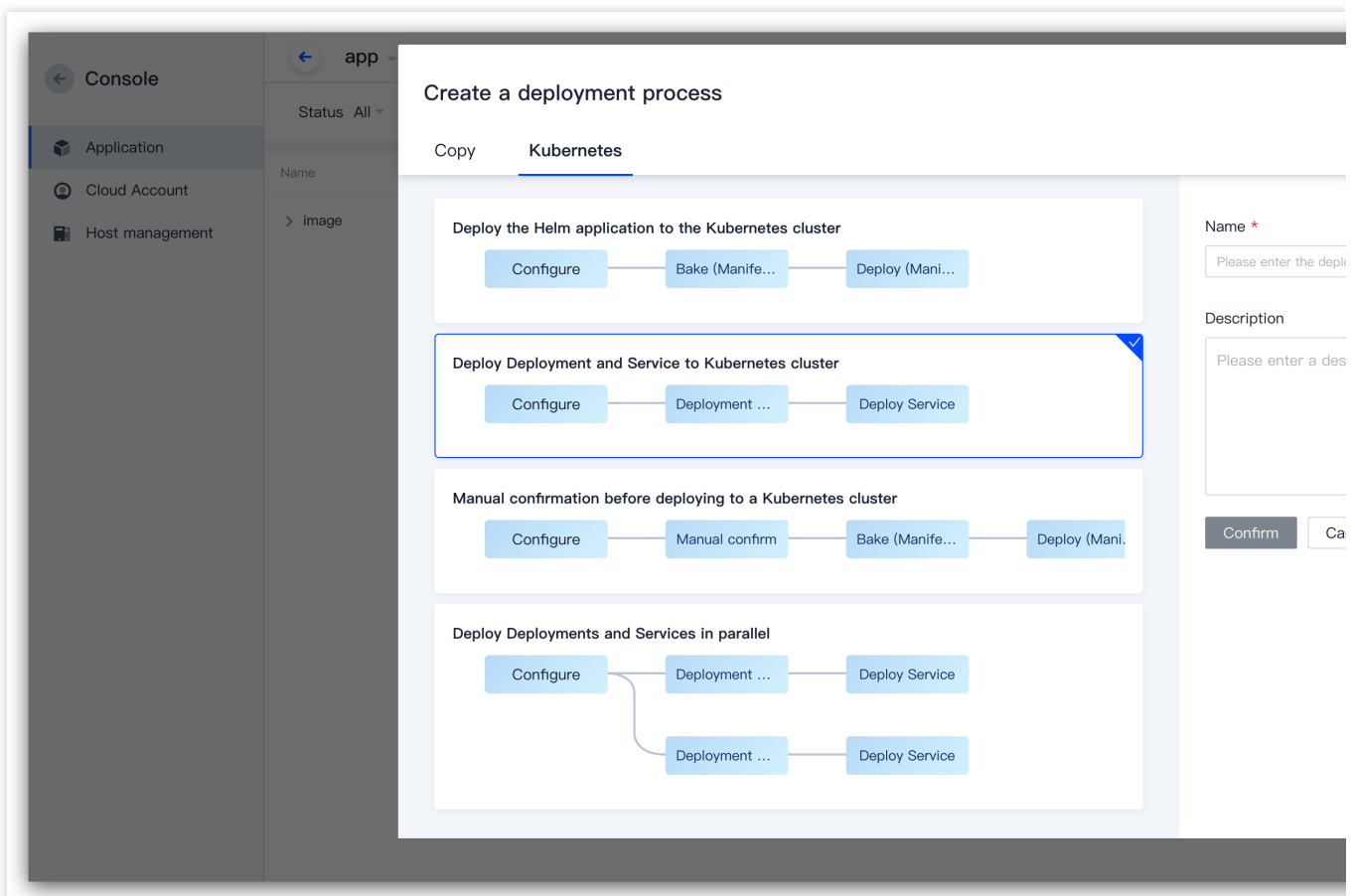


## 步骤2. 配置应用

1. 成功添加云账号后，在部署控制台中单击**创建应用**，填写应用名与选择部署方式。

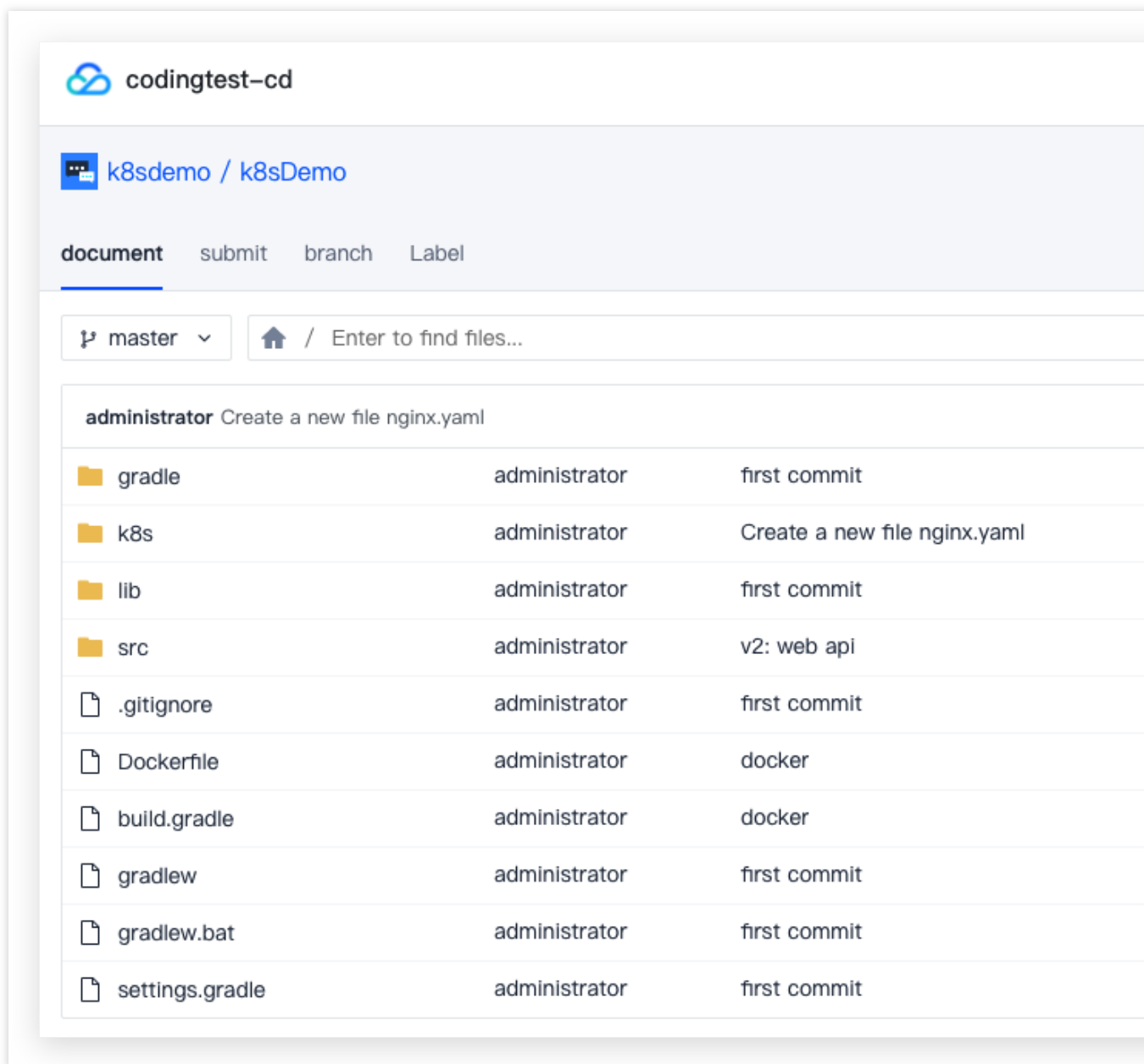


2. 选择部署到 **Kubernetes 集群**模板，填写名称与描述后完成创建。



### 步骤3. 初始化项目

1. 此步骤主要用于配置持续部署所涉及代码仓库与制品仓库。在**代码仓库**中点选导入外部仓库，访问 [示例仓库](#) 并克隆仓库地址。




2. 导入完成后进行制品管理，将拟发布的 Docker 制品托管至 CODING 制品仓库，详情请单击 [参考阅读](#)。

### Operation Guide

Configure Credentials

**Pull**

Mirror Acceleration 

[? Help Center](#)

### Pull

Enter the following pull information to generate the pull co

Name:

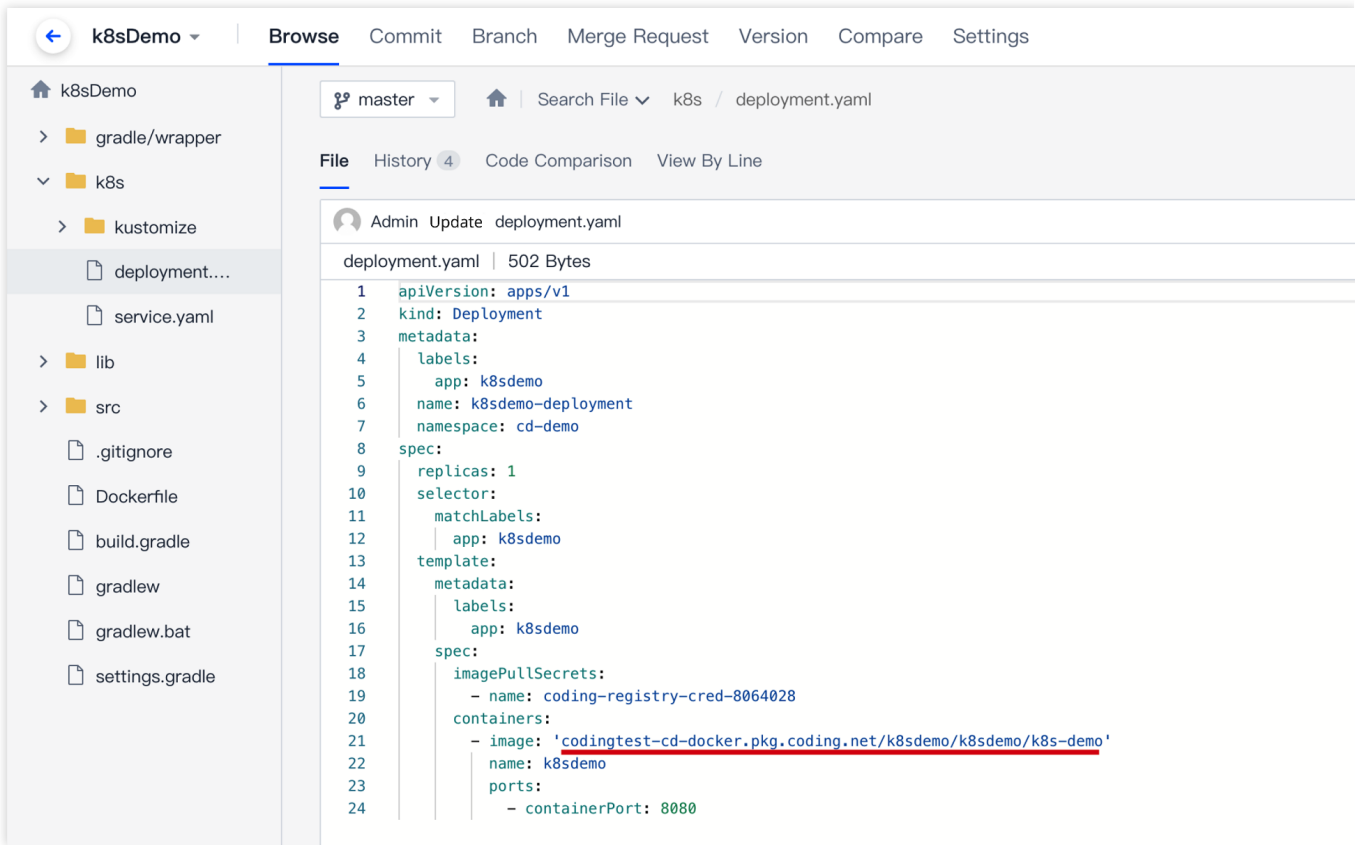
Version:

Run the following command in the command line to pull.

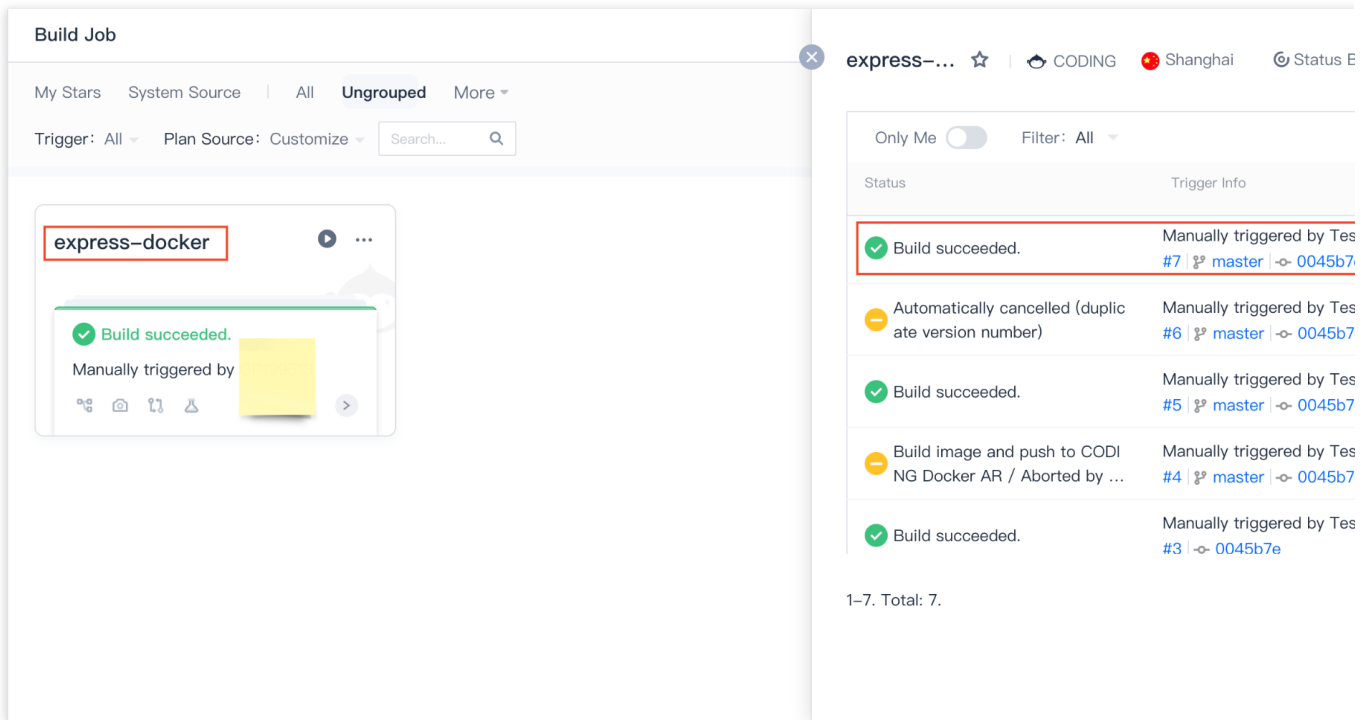
```
docker pull StrayBirds-docker.pkg.coding.net/cc
```

3. 推送至制品仓库后，获取制品的拉取地址并填写至代码仓库中 `/k8s/deployment.yaml` 中的 `image` 地址。

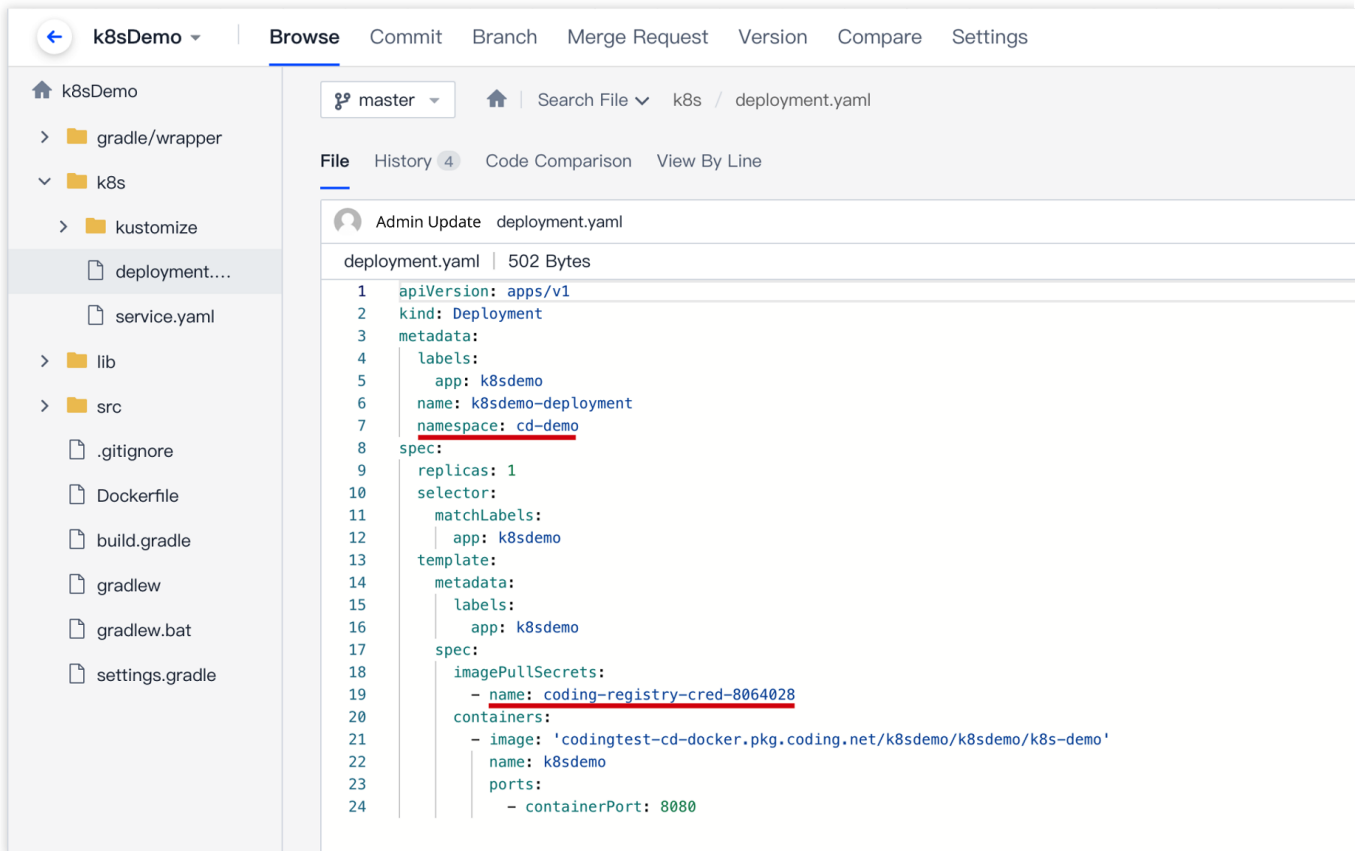




4. 接下来需导入云账号的 `imagePullSecrets` 至代码仓库中。在**部署控制台** > **云账号**中单击查看详情后，复制名称。

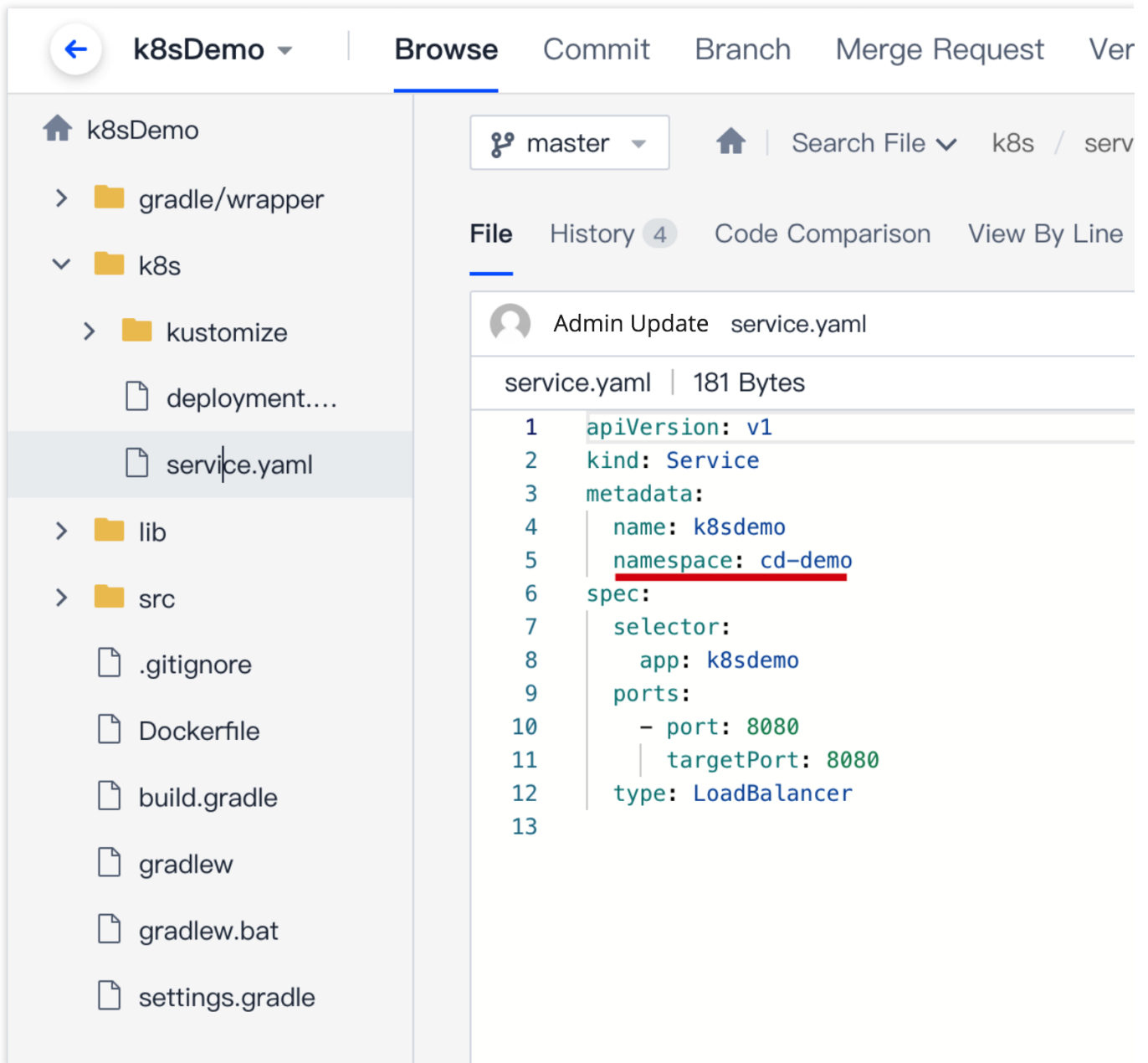


5. 粘贴至代码仓库中的 `deployment.yaml` 文件中，同时需注意 `namespace` 内容是否与上文中指定的命名空间一致。



```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    labels:
5      app: k8sdemo
6    name: k8sdemo-deployment
7    namespace: cd-demo
8  spec:
9    replicas: 1
10   selector:
11     matchLabels:
12       app: k8sdemo
13   template:
14     metadata:
15       labels:
16         app: k8sdemo
17     spec:
18       imagePullSecrets:
19         - name: coding-registry-cred-8064028
20       containers:
21         - image: 'codingtest-cd-docker.pkg.coding.net/k8sdemo/k8sdemo/k8s-demo'
22           name: k8sdemo
23           ports:
24             - containerPort: 8080
```

6. 同一层级的 `service.yaml` 文件中的 `namespace` 内容也需保持一致。



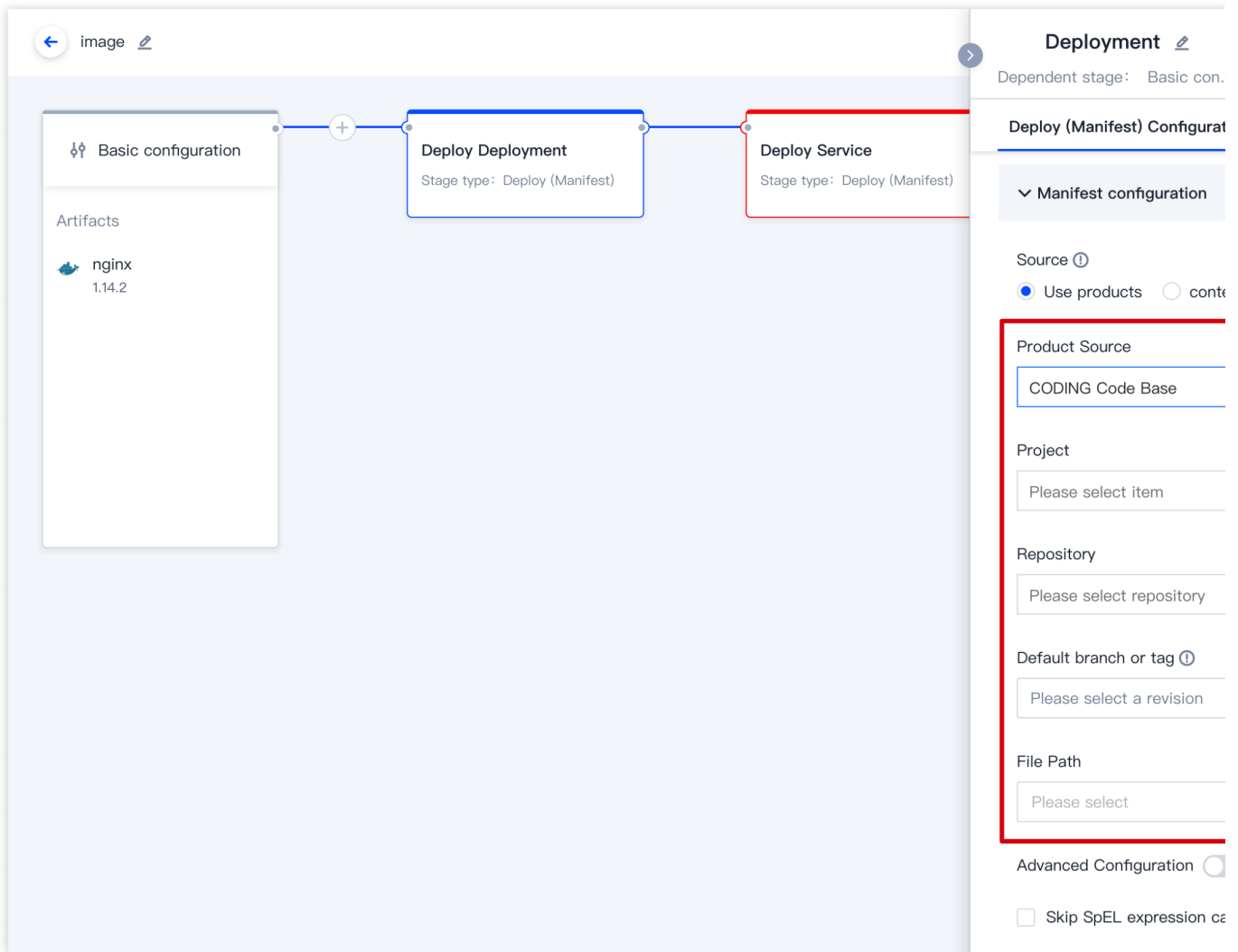
#### 步骤4. 配置部署流程

进入部署流程配置页面，可以为此流程设定：  
流程的执行选项（在此示例中保持默认即可）。

部署 Deployment 阶段以及部署 Service 阶段所需制品。

手动或自动触发。

1. 首先配置部署（Manifest）阶段。基础设置选择已绑定的云账号，在 Manifest 来源选择 **CODING 代码库**，填写相应的路径，镜像版本配置选择自动获取。



2. 配置部署 Service 阶段时步骤同上，但在文件路径处选择 `k8s/service.yaml` 文件。

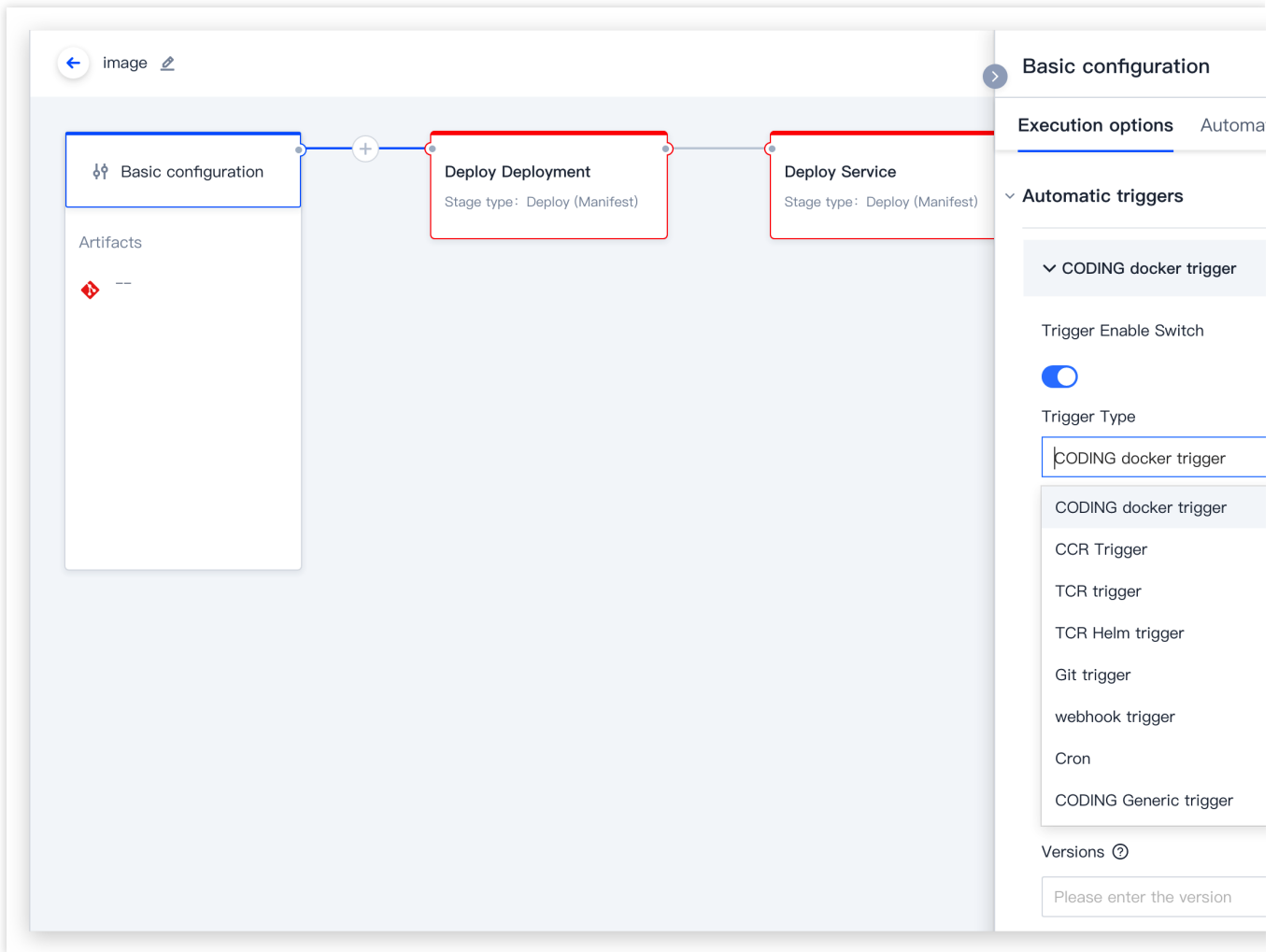
### 步骤5. 触发器配置

1. 完成部署阶段配置后，您可以使用自动化触发器、手动提交发布单执行部署。

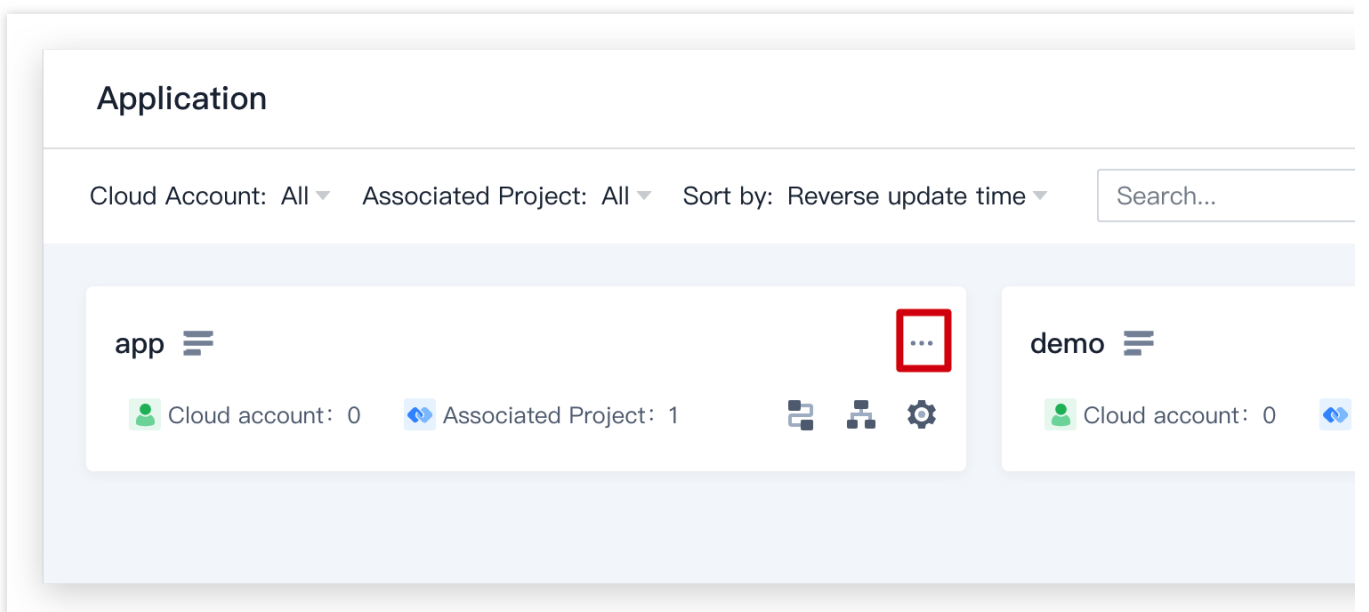
自动触发

手动提交发布单

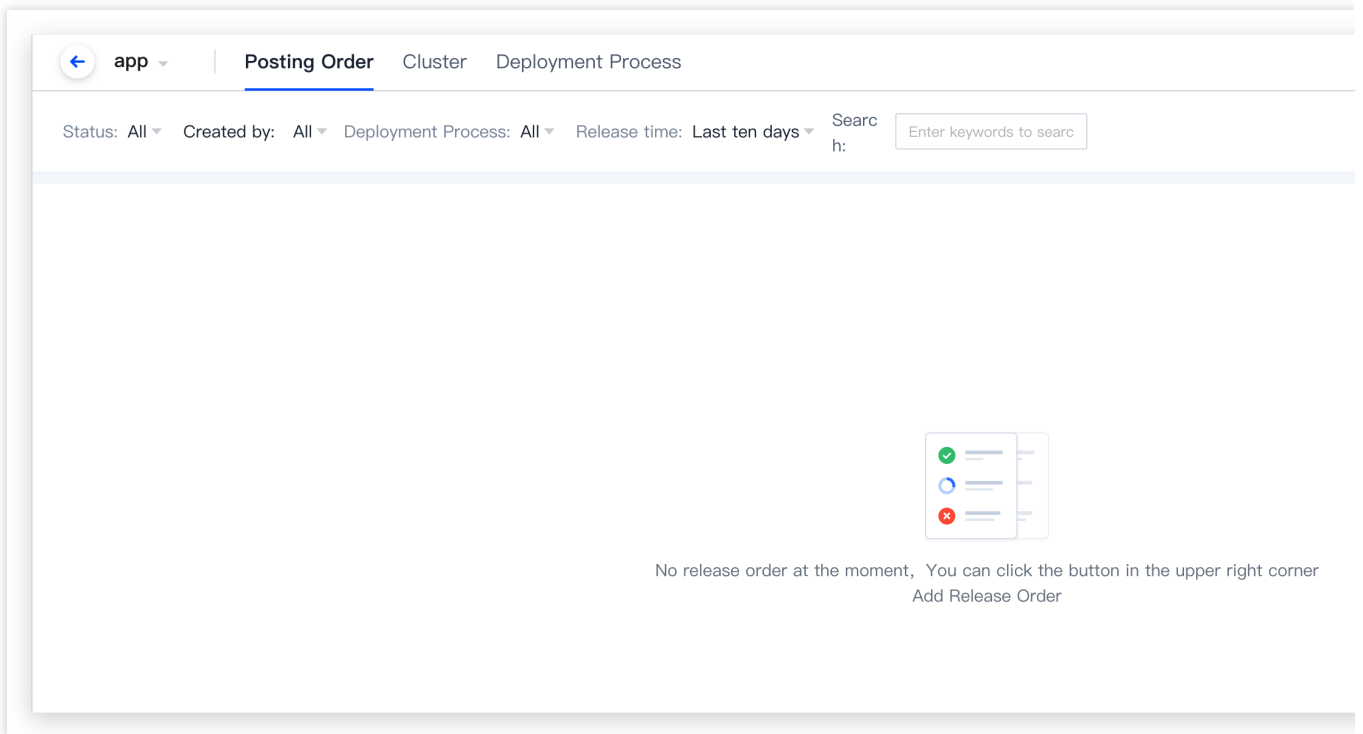
在**基础配置**中点选触发器类型，选择 Docker 仓库触发器。当开发人员更新代码仓库并使用 CI 将镜像打包推送至制品库后，Docker 镜像版本的更新将自动触发部署流程并将应用发布至 Kubernetes (TKE) 集群，完成后可以在基础设施页面查看并确认应用是否发布成功。



若希望通过手动提交发布单的形式触发部署流程，那么可以将**应用**（如本范例的 flaskapp）与项目关联。在部署控制台的**应用列表**中搜索项目名称进行关联：



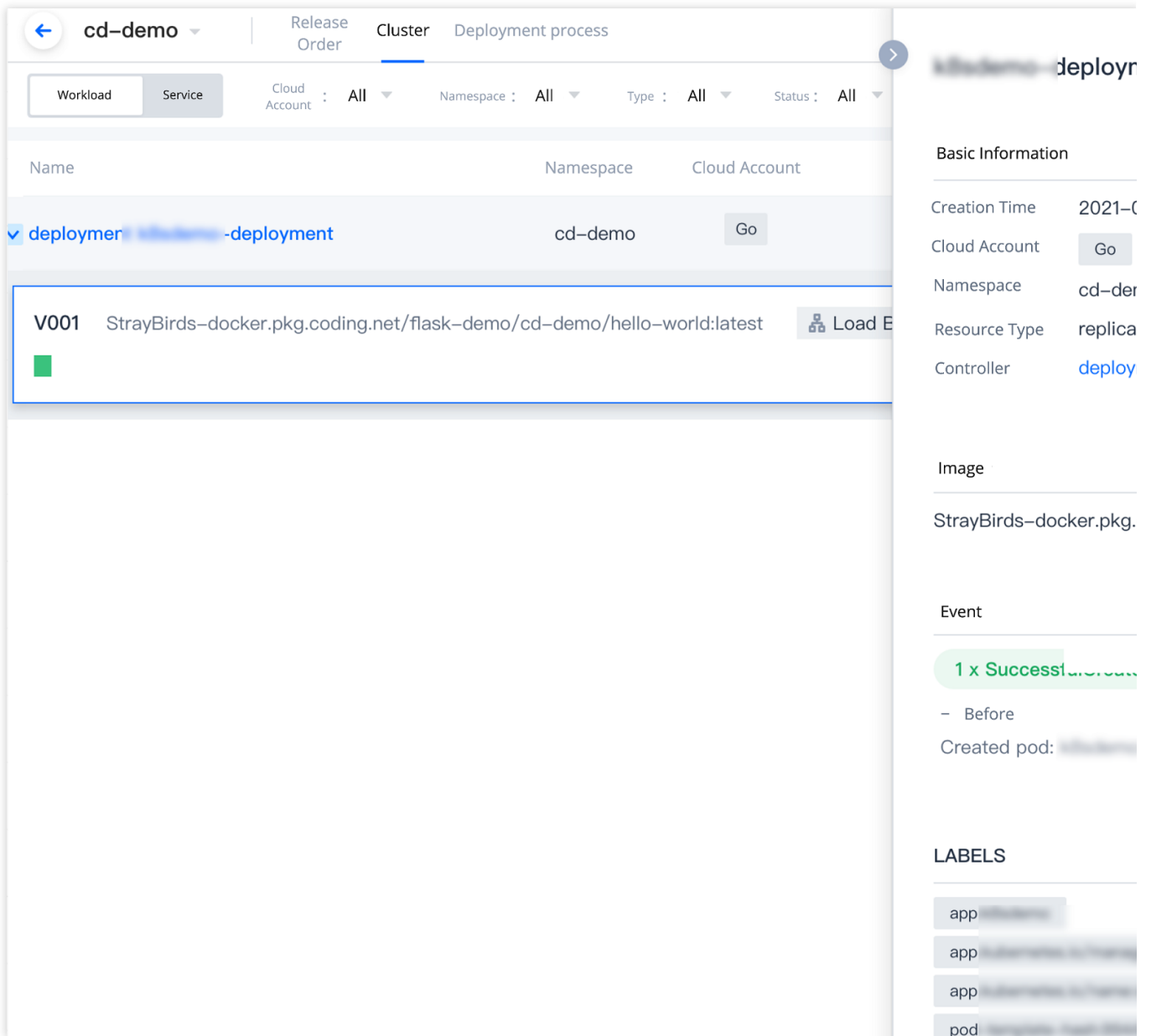
2. 关联完成后，单击项目中的**持续部署** > **Kubernetes** 手动提交发布单。



## 步骤6. 发布完成

1. 发布成功后，可以查看发布的制品及启动参数及阶段执行详情等信息。

2. 当需要查看某个资源在集群中的运行状态时，单击**集群**下的工作负载即可查看详情（如工作负载的 Pod 实例，日志等信息）。



3. 在腾讯云的容器服务中查看工作负载。



Container Service

Cluster(Guangzhou)

- Overview
- Cluster**
- Elastic Container
- Edge Cluster
- Service Mesh
- Application Center
- Application
- Image Repository
- Application Market
- Ops Center
  - Cluster Ops
- Cloud-Native Monitoring
- Container Image Service
- DevOps
- Quick Start

Basic Information

Node Management

Namespace

Workload

- Deployment**
- StatefulSet
- DaemonSet
- Job
- CronJob

Auto Scaling

Services and Routing

Configuration Management

Authorization Management

Storage

Component Management

Log

Event

### Deployment

New Monitor

Namespace: cd-demo

<input type="checkbox"/>	Name	Labels	Selector	Number of running/expected p
<input type="checkbox"/>	cd-demo-deployment	app=cd-demo	app=cd-demo	1/1

page 1