

# devOps实验报告

---

## 1 持续规划与设计

### 1.1 创建华为云CodeArts项目

在华为云官网创建codeArts项目，项目名称命名为“凤凰商城”

### 1.2 使用Scrum项目模板进行项目规划

相关知识

Epic：史诗，是项目的愿景目标。通过Epic的落地达成，使公司可以获得相应的市场地位和回报，具有战略价值。通常需要数月完成。

Feature：可以带来价值的产品功能和特性。相比Epic，Feature更具体，更形象，客户可以感知，具有业务价值。通常需要数周，多个Sprint才能够完成。

Story：通常所说的用户故事，是User Story的简称。Story是从用户角度对产品功能的详细描述，承接Feature，并放入产品Backlog中，持续规划，滚动调整，始终让高优先级Story交付给客户，具有用户价值。Story要符合INVEST原则（Independent、Negotiable、Valuable、Estimable、Small、Testable），通常需要数天，并在一个Sprint中完成。

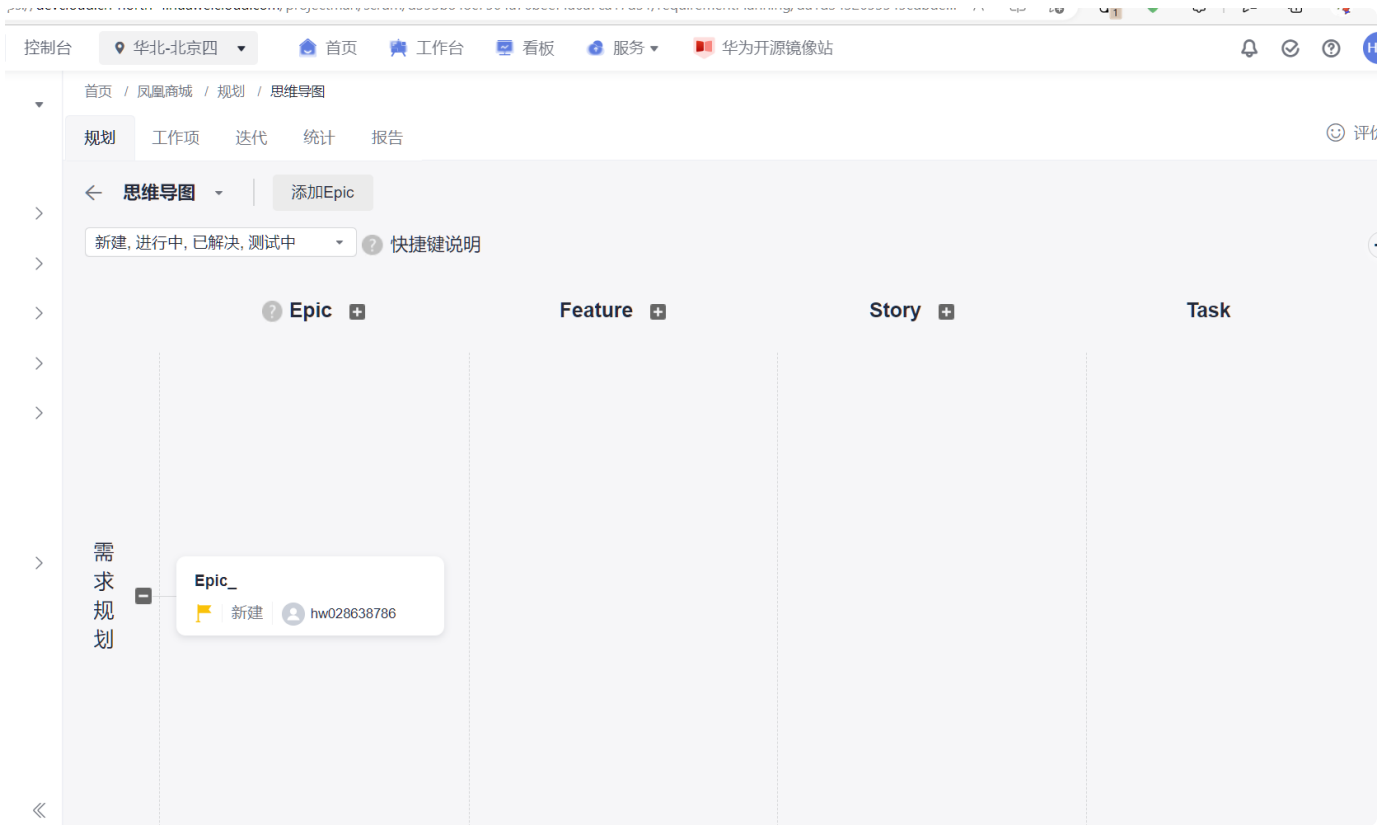
Task：是团队成员要完成的具体任务。在Sprint计划会议上，将Story分配给成员，然后由成员分解为Task，并预估工时，通常在一天内完成。

#### 1.2.1

打开凤凰商城项目，单击“规划”，点击“思维导图规划”，输入“思维导图”，点击确认。引入思维导图对于项目进行规划。

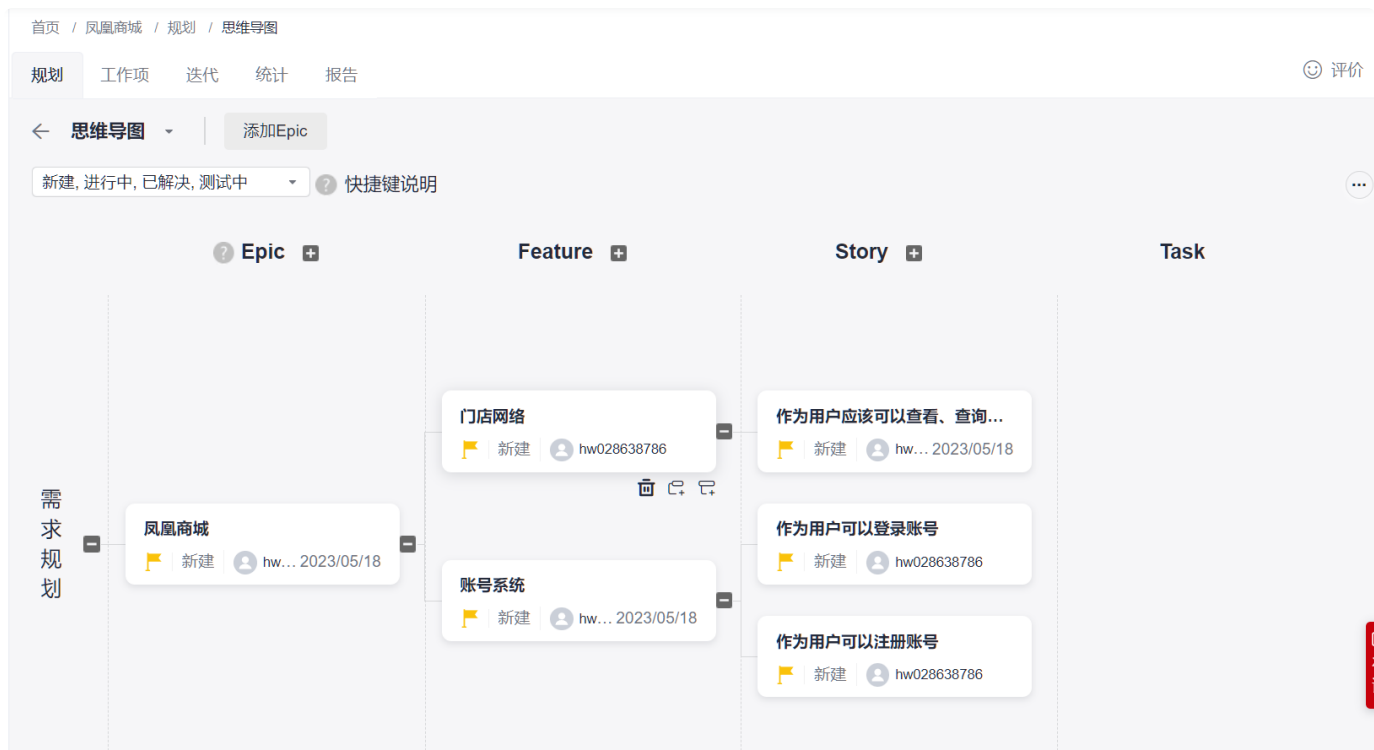
#### 1.2.2

首先需要在思维导图中的的第一部分“EPIC”创建需求规划



### 1.2.3

其次，在Epic后插入子节点，新建Feature，输入标题 " 门店网络 "，回车保存；相同方式新建Feature“账号系统”。为Feature " 门店网络 " 插入子节点，添加Story " 作为用户应该可以查看、查询所有门店网络 "；为Feature“账号系统”插入子节点Story“作为用户可以登录账号”、“作为用户可以注册账号”。通过上述步骤我们将一个项目的前期完整规划通过思维导图体现了出来。以下是最终结果：



### 1.3 使用Scrum项目模板管理Backlog并进行迭代开发

#### 相关知识

Backlog 英文意思为“积压的工作”，Product Backlog 其实就是一个具有优先级的需求列表， 并对每个需求进行了粗略的估算。

迭代定义为：如果算法的定义没有包含算法本身，则叫做迭代法。迭代式开发 也被称作 迭代增量式开发 或 迭代进化式开发，是一种与传统的 瀑布式开发 相反的 软件开发过程，它弥补了传统开发方式中的一些弱点，具有更高的成功率和生产率。

每日站立会议(晨会) 这是在工作日特定的时间举行的短小（15分钟）的会议，开发团队的每一成员都将参与，通常可以选择在早上或者下午下班前进行。为了保证其短小精悍，与会成员都保持站立（所以叫“站立会议”）。以此提供给开发团队机会来汇报交流成果和阐述任何存在的障碍。

#### 1.3.1

首先，进入界面。单击 " 工作>工作项>Backlog "，进入Backlog页面。

选择上文思维导图中的story，进行编辑，详细描述相关信息

## #64355386 作为用户应该可以查看、查询所有门店网络

描述信息 子工作项(0) 关联(0) 详细工时 操作历史

作为用户，我想要查看所有门店网络信息，以便于我决定去一家距离最近的门店

标签 ?

附件 ?

+ 点击添加附件或拖拽文件到此处上传

评论

@通知他人, Ctrl+V粘贴截图

状态:	新建
* 处理人:	hw028638786
模块:	门店网络
迭代:	请选择
预计开始...	2023/05/17
预计结束...	2023/05/18
优先级顺序:	1
* 优先级:	中
* 重要程度:	重要
抄送人:	请选择
父工作项:	Feature 门店网络
领域:	请选择
* 预计工时:	24.00   3.00

### 1.3.2

其次，对于Backlog列表，我们同时可以设置自己关注的用户故事，方便查询。

我们还可以对于Backlog进行高级管理。可以通过快速过滤器方便地查询特定的工作项，也可以使用 " 高级过滤 " 实现特定字段指定条件的过滤。并且可以自定义过滤器，寻找最符合自己当前需求的过滤选项。

### 1.3.3

接着，我们可以创建迭代进行项目管理。

单击 " 工作 > 迭代 "，进入迭代管理视图。单击 " 创建迭代 "，在弹框中输入迭代名称 " 迭代4 "、设置迭代计划时间，单击 " 新建 " 便能够创建迭代，在迭代中可以根据自己的实际需求设置迭代的开始日期和结束日期，下面是在凤凰商城项目中已经创建好的迭代



### 1.3.4

接下来，我们需要往迭代中添加任务，其中最重要的工作 " 作为用户应该可以查看、查询所有门店网络 " 需要在本迭代完成并上线。

在迭代页面，单击 " 未规划工作项 "，找到Story " 作为用户应该可以查看、查询所有门店网络 "，鼠标拖拽工作项至 " 迭代4 "。

单击 " 迭代4 " 的 " 作为用户应该可以查看、查询所有门店网络 "，可以设置Story的预计开始日期与预计结束日期。下面是我已经向迭代四中添加项目的情况：



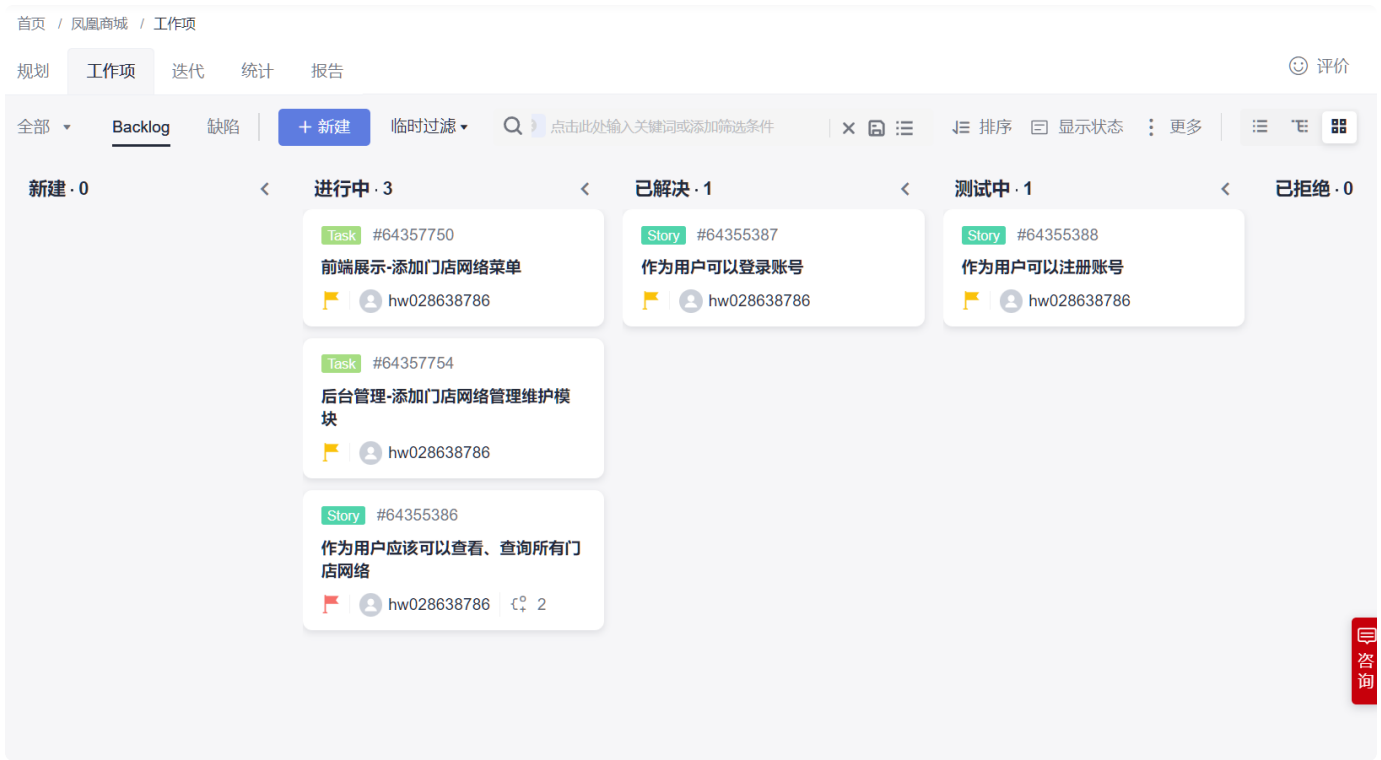
### 1.3.5

接下来，我们需要将story分解成更加详细的task，拆分到开发任务级别，并指派给对应的负责人。击工作项列表最右侧 " 操作 " 列中的图标，新建子Task。输入Task标题 " 前端展示-添加门店网络菜单 "，并选择处理人，单击 " 确定 " 完成。下面是我拆分之后的story：



1.3.6

看板视图使用。点击右上角图标，切换视图为 " 卡片模式 " 。在此模式下，可通过拖拽修改工作项状态。下面是我的凤凰商城项目的看板：



1.4 项目管理配置

1.4.1

项目基本信息设置。进入项目，单击页面左侧导航 " 设置 > 通用设置 " 。

接着，单击页面左侧菜单 " 基本信息 "，进入 " 基本信息 " 页面。管理员可根据情况进行修改项目名称、项目描述，移交创建人，完成编辑，单击 " 保存 "。可以根据自己的实际需要更改项目的详细信息。

其次，可以管理项目人员。单击页面左侧菜单“成员管理”，进入“成员管理”页面。点击添加成员，在下拉菜单中选择“邀请其他企业用户”，在弹出窗口的“企业用户”文本框输入小组成员账号进行添加。重复该步骤直到小组全部成员添加完毕。

站内通知以及邮件通知设置。进入项目，单击页面左侧导航 " 设置 > 项目设置 "。单击页面左侧菜单 " 通知设置 "，进入 " 通知设置 " 页面。根据需要勾选通知场景、方式、对象即可，系统将自动保存。

模块添加与管理。进入项目，单击页面导航 " 设置 > 项目设置 "。单击页面左侧菜单 " 模块设置 "，进入 " 模块设置 " 页面。单击 " 添加 "，输入名称、描述，选择负责人，单击 " 确定 " 保存。

1.4.2

1 增加工作项字段。进入项目，单击页面导航 " 设置 > 项目设置 "。单击页面左侧菜单 " Story设置 > 字段与模板 "，进入 " 工作项模板 " 页面。单击页面右上角 " 编辑模板 "，选择 " 新建字段 "。在弹框中输入字段名称 " 验收标准 "，勾选字段类型 " 多行文本 "，单击 " 确定 " 保存。

模块设置

Q 请输入关键字，按enter键搜索

添加 一键替换

名称	描述	负责人	操作
模块设置	模块设置	hw028638786	+
促销管理	促销管理	hw028638786	+
配件管理	配件管理	hw028638786	+
订单管理	订单管理	hw028638786	+
会员管理	会员管理	hw028638786	+
门店网络	门店网络	hw028638786	+

总条数: 6 < 1 >

2 增加工作项状态。进入项目，单击页面上方导航 " 设置 > 项目设置 "。单击页面左侧菜单 " 公共状态设置 "，进入 " 状态管理 " 页面。单击右上角 " 添加状态 "，在弹框中输入状态 " 验收中 "，选择状态属性为 " 进行态 "，单击 " 添加 " 保存。

3 单击页面左侧菜单 " Story设置 > 状态与流转 "，进入 " 工作项状态 " 页面。单击 " 添加已有状态 "，在弹框中勾选 " 验收中 "，单击 " 确定 " 保存。

最终结果如下：

## 工作项状态

状态管理

流转方向

自动流转

1. 此处可以为该类型工作项新增状态、拖动调整顺序、删除状态；
2. 新增、删除和调整顺序仅对本工作项类型生效。

名称	状态属性②	描述	操作
新建	开始态		
 进行中	进行态		
 已解决	进行态		
 测试中	进行态		
 验收中	进行态		
 已拒绝	结束态		
已关闭	结束态		

+ 添加已有状态    + 创建新的状态

## 2 持续开发与集成

### 2.1 使用CloudIDE修改和提交代码

#### 2.1.1

登录华为云，使用CloudIDE

#### 2.1.2

选择“我的IDE”栏，“新建实例”。来源选择“私有仓库”，项目名称选择您在CodeArts创建的样例项目名称，仓库地址选择“phoenix-sample.git”，确定。

#### 2.1.3

等待CloudIDE加载完毕后，在左侧导航中找到前端界面代码文件 " /vote/templates/index.html " 并打开，在179行添加菜单 " 门店网络 "。



2.1.4

提交修改至代码仓库。单击左侧边栏图标，打开Git功能。单击修改文件后方的图标 (git add)，将修改内容添加进提交内容当中。在输入框中输入提交信息： " fix #工作项编码 本次提交的注释信息 " 。单击打勾图标 (git commit)提交本次修改。单击图标，在下拉列表中单击 " Push " ，推送代码到代码仓库。

2.1.5

查看代码提交记录。返回 " 代码托管 " 页面，在phoenix-sample代码仓库中选择 " 历史 " 页签，即可查看是否提交成功。交记录如下：

添加了一个名为"fix"的提交记录



2.1.6

跳转至 " 工作>工作项>Backlog " 页面，单击Story " 作为用户应该可以查看、查询所有门店网络 " 。在 " 关联 " 页签中，单击 " 代码提交记录 " ，也可看到相应提交记录。

#64357754 后台管理-添加门店网络管理维护模块

描述信息 关联(1) 详细工时 操作历史

▼ 关联工作项(0)

▼ 关联Wiki(0)

^ 代码提交记录(1)

一头雾水,不知如何开始? [点击这里](#)

分支	描述	提交者	提交时间
master	4d989636 - fix #64357754	hw028638...	2023/05/1...

▼ 关联代码分支(0)

标签 ?

附件 ?

状态: 已解决

\* 处理人: hw028638786

模块: 门店网络

迭代: 迭代4

自动填充为迭代起始日期? ☐ 是 ☒ 否

预计开始... 2023/05/15

预计结束... 2023/05/26

优先级顺序: 1

\* 优先级: 中

\* 重要程度: 一般

抄送人: 请选择

父工作项: Story 作为用户应该可以...

领域: 请选择

2.1.7

打开vote目录下的app.py源文件，如右下角出现“Linter pylint is not installed”，点击“Install”安装。点击左下角选择“Python 3.8.x”版本解释器

2.1.8

打开vote下Dockerfile文件，修改“FROM python:2.7-alpine”为“python:3.8-alpine”。

2.1.9

在终端窗口安装如下模块：pip install flask redis --user

2.1.10

在app.py中替换代码：

导入import importlib

修改reload(sys)为importlib.reload(sys)

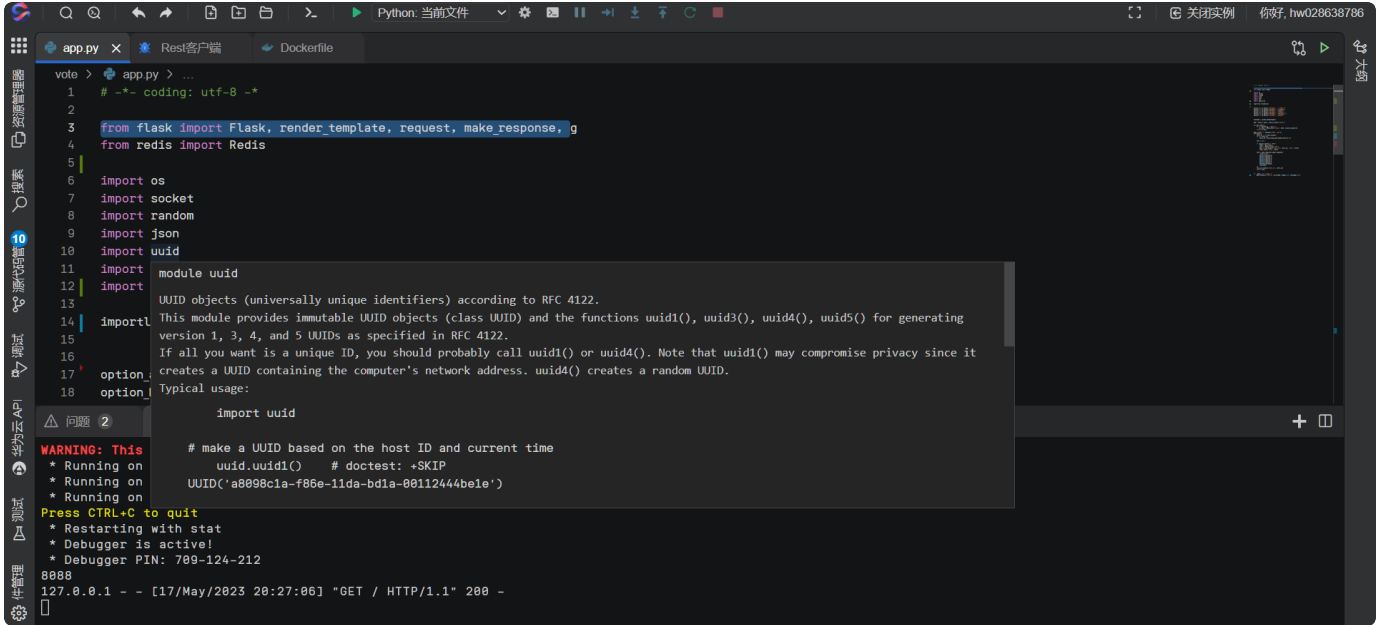
删除sys.setdefaultencoding('utf-8')

```
app.py 修改追溯 历史 对比 1.5 KB hw028638786 提交 ba66580c 于 2023/05/17 21:00:55 GMT+08:00 全屏 复制代码
```

```
1 # -*- coding: utf-8 -*-
2
3 from flask import Flask, render_template, request, make_response, g
4 from redis import Redis
5
6 import os
7 import socket
8 import random
9 import json
10 import uuid
11 import sys
12 import importlib
13
14 importlib.reload(sys)
15
16
17 option_a = os.getenv('OPTION_A', "空气滤芯")
18 option_b = os.getenv('OPTION_B', "汽车电瓶")
19 option_c = os.getenv('OPTION_C', "刹车片")
20 option_d = os.getenv('OPTION_D', "汽油滤芯")
21 option_e = os.getenv('OPTION_E', "电瓶搭火线")
22 option_f = os.getenv('OPTION_F', "刹车钳")
23
24
25 hostname = socket.gethostname()
26
27 app = Flask(__name__, static_folder='assets')
28
29 def get_redis():
30     if not hasattr(g, 'redis'):
31         g.redis = Redis(host="redis", db=0, socket_timeout=5)
32     return g.redis
33
34 @app.route("/", methods=['POST','GET'])
```

2.1.11

安装完成后将app.py中第64行代码中的端口号改为8088（8000–9000皆可）， 点击右上角运行图标， 此时前端程序已执行。



```
1 # -*- coding: utf-8 -*-
2
3 from flask import Flask, render_template, request, make_response, g
4 from redis import Redis
5
6 import os
7 import socket
8 import random
9 import json
10 import uuid
11 import
12 import
13
14 importl
15
16
17 * option_
18 option_
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
WARNING: This
* Running on
* Running on
* Running on
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 789-124-212
8088
127.0.0.1 - - [17/May/2023 20:27:06] "GET / HTTP/1.1" 200 -
```

2.2 使用Git分支+合并请求方式提交代码并进行代码检视

2.2.1

将master分支设置为受保护分支。进入项目，单击页面上方导航 " 代码 > 代码托管 "，进入代码托管服务。单击仓库名称，进入代码仓库。

选择 " 设置 " 页签，在左侧导航中单击 " 仓库管理 > 保护分支管理 "。

单击 " 新建分支保护 "，根据需要在弹框中选择配置，单击 " 确定 " 保存。



2.2.2

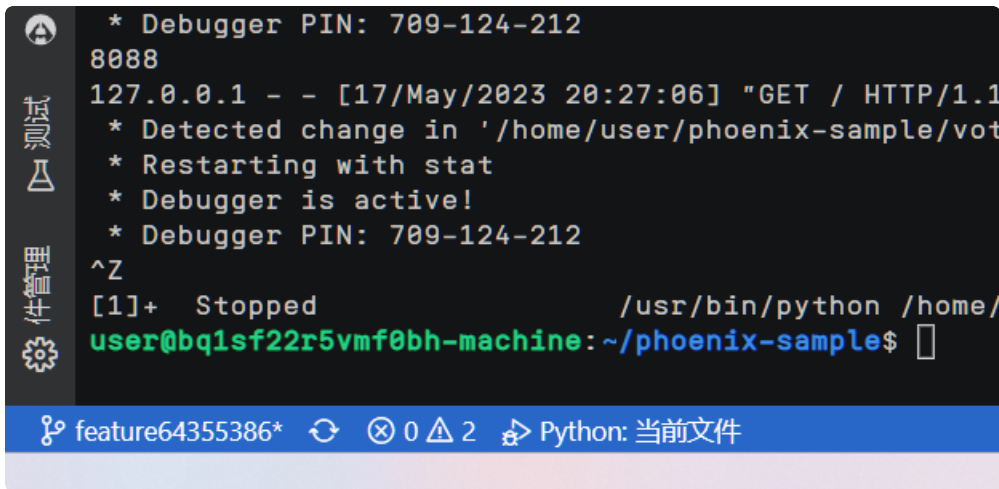
创建分支并进行新功能开发。进入代码仓库，选择 " 分支 " 页签，单击右上角 " 新建分支 "，在弹框中输入分支名称 " feature工作项编号 "（注意：不要有+号等特殊符号）命名分支，单击 " 确定 " 保存。

2.2.3

用CloudIDE打开代码仓库，选择新创建的分支进入（如果是已打开的CloudIDE界面，可单击页面左下角分支名称，在页面上方的下拉列表中选择刚刚创建分支。可以看到左下角从 " master " 变成了新建分支名）。

更改之后如下图所示：





```
* Debugger PIN: 709-124-212
8088
127.0.0.1 - - [17/May/2023 20:27:06] "GET / HTTP/1.1"
* Detected change in '/home/user/phoenix-sample/vot
* Restarting with stat
* Debugger is active!
* Debugger PIN: 709-124-212
^Z
[1]+  Stopped /usr/bin/python /home/
user@bq1sf22r5vmf0bh-machine: ~/phoenix-sample$
```

## 2.3 合并请求（登录模块）

### 2.3.1

创建合并请求，进入代”码托管，选择合并请求页签，单击“新建合并请求”。源分支选择刚刚创建的分支，与目标分支选择 " master "，单击 " 下一步 "。输入标题、描述（选填），选择合并人、评审人，单击 " 确定 " 完成。" 合并人 " 即接受合并请求的人，" 评审人 " 是由合并发起人邀请参与的评审者。对于不合格的合并请求，管理员可以关闭。在评分不够时，无法完成该合并请求。

### 2.3.2

代码检视以及评分，评审人进入代码仓库后，在 " 合并请求 " 页签中找到需要评审的合并请求，单击该请求，查看合并请求详情。评审者可以在 " 合并请求 " 页签中发表评审意见，对合并请求进行评分。

### 2.3.3

评审成员提交评分。若只有一个账号，请将允许分支合并的最低评分设置为2分。

### 2.3.4

合并人进入代码仓库后，在 " 合并请求 " 页签中找到需要评审的合并请求，单击该请求，查看合并请求详情。选择 " 合入 "，系统将提示 " 合并成功 "。



## 2.4 代码检查任务和管理代码检查规则集

### 2.4.1

编辑代码检查任务包含语言。进入项目，单击页面上方导航 " 代码 > 代码检查 "，进入代码检查服务。单击代码检查任务 " phoenix-codecheck-worker "，进入 " 代码检查详情 " 页面。选择 " 设置 " 页签，在页面左侧导航中单击 " 规则集 "。规则集中默认包含的语言是 " Java "。单击图标重新获取代码仓库语言，在刷新的列表中将Python语言对应的开关打开。

### 2.4.2

启动代码检查任务。进入代码检查任务 " phoenix-codecheck-worker " 的 " 代码检查详情 " 页面。单击 " 开始检查 "，启动代码检查任务。当页面提示 " 分支 " master " 最近一次检查成功！"，表示任务执行成功。



## 2.4.3

创建自定义检查规范。进入代码检查服务，选择 "规则集" 页签，在下拉列表中选择 "Java"。在过滤出的列表中找到规则集 "关键检查规则集"，单击图标，在下拉列表中选择 "复制"。在弹框中输入新规则集名称为 "phoenix-java-rule-set"，单击 "确定" 保存。

## 2.4.4

启用自定义检查规范。进入代码检查任务 "phoenix-codecheck-worker" 的 "代码检查详情" 页面。选择 "设置" 页签，在页面左侧导航中单击 "规则集"。在Java语言规则集中，勾选规则集 "phoenix-java-rule-set"。如下是设置的自定义规则：



## 2.5 使用自动化编译提高代码质量验证速度

### 2.5.1

查看编译构建任务。进入项目，单击页面上方导航 " 构建&发布 > 编译构建 "，进入编译构建服务。找到编译构建任务 " phoenix-sample-ci "。单击 " phoenix-sample-ci " 右侧，进入 " 编辑 " 页面。

### 2.5.2

点击 " 源码选择 " 栏， " 仓库分支 " 选择 " master "

### 2.5.3

配置SWR服务。项目应用镜像存放需要使用到华为云容器镜像服务 (SWR)，因此需要首先配置SWR服务。点击 " 构建步骤 " 栏，找到 " 华为云容器镜像服务 " 并点击，

单击右上角 " 登录指令 "，系统生成并弹框显示docker login指令。指令中，-u之后的字符串为用户名，-p之后的字符串为密码（此为临时用户名和密码，24小时刷新，隔天使用需重新查看用户名、密码。最后为服务器地址

### 2.5.4

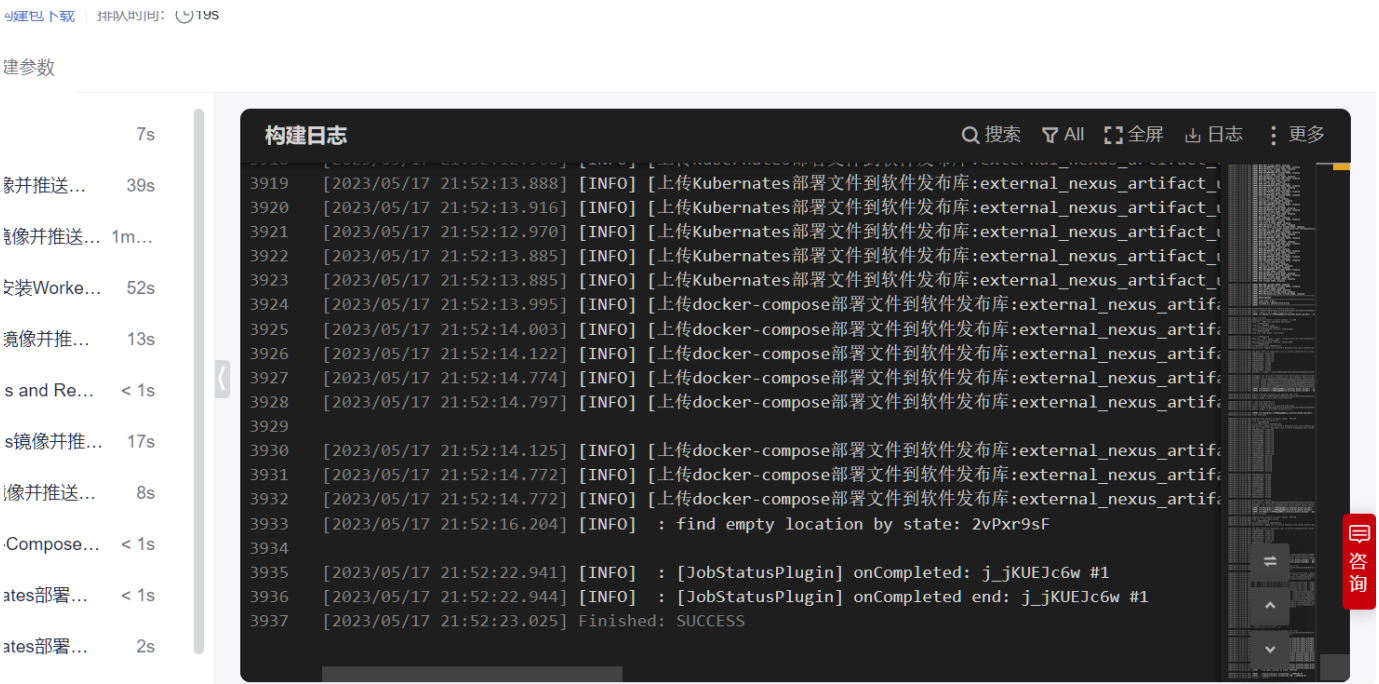
返回CodeArts页面，选择 " 参数设置 " 页签，编辑以下两个参数。



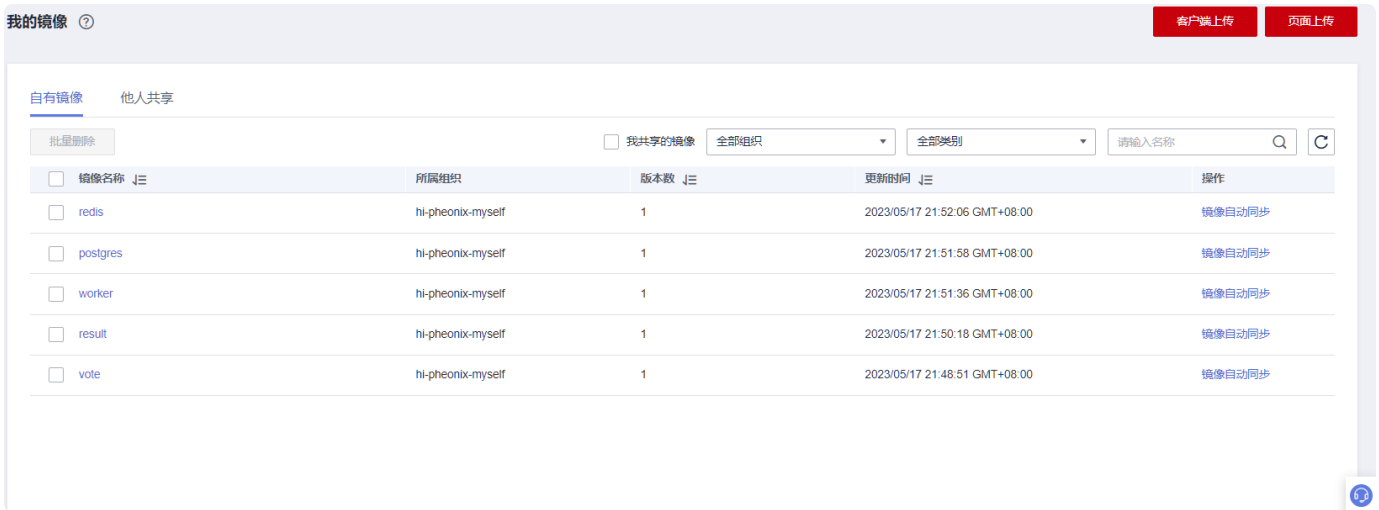
- | dockerOrg: 输入在之前创建的组织。
- | dockerServer: 输入在之前记录的内容。

2.5.5

在左边找到持续交付-编译构建，选择phoenix-sample-ci项目执行，等待编译构建结束，以下是构建成功时的截图：



同时在SWR中可以找到我们构建好的镜像：



单击页面上方导航 " 构建&发布 > 发布 "，进入发布服务。列表中可找到下图所示的两个文件夹：



## 2.5.6

持续集成配置。进入编译构建任务 " phoenix-sample-ci " 的 " 编辑 " 页面，选择 " 执行计划 " 页签，打开 " 提交代码触发执行 " 开关，单击 " 保存 " 。

持续集成执行计划配置。在 " 执行计划 " 页签，开启“定时执行”，根据需要设置定时执行计划，单击 " 保存 " 。

## 2.6 使用开源镜像站服务改进自动化编译

在执行编译构建任务时自动获取开源镜像站中的镜像：

将镜像源设置为对应的开源镜像站仓库地址配置在代码仓库的配置文件中，查看方式 " 代码>代码托管>phoenix-sample>文件 " 。配置方式有以下三种：

Python配置方法如下图：

Dockerfile

修改追溯

历史

对比

612 Bytes

提交信息

```

1 # Using official python runtime base image
2 FROM python:3.8-alpine
3
4 # Set the application directory
5 WORKDIR /app
6
7 # Install our requirements.txt
8 ADD requirements.txt /app/requirements.txt
9 RUN pip install -i https://repo.huaweicloud.com/repository/pypi/simple -r requirements.txt
10
11 # Copy our code from the current folder to /app inside the container
12 ADD . /app
13
14 # Make port 80 available for links and/or publish
15 EXPOSE 80
16
17 # Define our command to be run when launching the container
18 CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]

```

## 3 持续部署与发布

### 3.1 购买并配置弹性云服务器

购买弹性主机，并且选择 " 入方向规则 " 页签，单击 " 添加规则 "，添加一条入方向规则 " 允许访问 5000以及5001端口 "。

#### 3.1.1

新建主机组。在左侧选项栏持续交付-部署中选择一个项目，选择环境管理页签，新建主机组。

单击 " 添加主机 "，在弹框中输入刚刚购买的ECS主机信息，单击 " 添加 " 保存。

当连通性验证成功之后表示主机添加成功



## 3.2 使用CodeArts部署服务配置主机环境

### 3.2.1

进入项目，单击页面上方导航 " 持续交付 > 部署 "，进入部署服务。点击右上角 " 新建任务 "，自定义任务名称，进入下一步；

点击基本信息输入任务名称。选择 " 空白模板 "，进入下一步；

### 3.2.2

在部署步骤中添加步骤 " 安装/卸载Docker "，设置主机组为前述已添加的授信主机组；

### 3.2.3

点击 " 安装/卸载Docker " 方框下方的，继续添加步骤 " 执行shell命令 "，在shell命令框中输入如下命令行

```
sudo apt-get install libssl-dev libffi-dev python-dev build-essential libxml2-dev libxslt1-dev -y
```

```
pip3 install six --user -U
```

```
pip3 install -i https://repo.huaweicloud.com/repository/pypi/simple docker-compose==1.17.1
```

### 3.2.4

安装python3，选择版本python-3.7.1

### 3.2.5

保存并执行，等待成功安装。

< install #1

部署成功 #1 人 hw028638786 执行于 2023/05/20 21:06:04 GMT+08:00 耗时 04分03秒

回退到此版本

部署日志 执行参数 访问方式 目标主机组

#### 部署步骤

- 初始化 1.820s
- 安装/卸载Docker 26.530s
- 执行shell命令 61.722s
- 安装Python 152.752s

#### 日志

```
139 [2023/05/20 21:10:06.358] TASK [Configure environment PATH] *****
140 [2023/05/20 21:10:06.358] changed: [1_***.***.***.131]
141 [2023/05/20 21:10:06.666]
142 [2023/05/20 21:10:06.666] TASK [Installation verification] *****
143 [2023/05/20 21:10:06.666] changed: [1_***.***.***.131]
144 [2023/05/20 21:10:06.718]
145 [2023/05/20 21:10:06.719] TASK [View installation result1] *****
146 [2023/05/20 21:10:06.719] ok: [1_***.***.***.131] => {
147 [2023/05/20 21:10:06.719]   "msg": "Python-3.7.1 setup was successfully"
148 [2023/05/20 21:10:06.719] }
149 [2023/05/20 21:10:06.742]
150 [2023/05/20 21:10:06.742] PLAY RECAP *****
151 [2023/05/20 21:10:06.742] 1_***.***.***.131 : ok=35 changed=19 unreachable=0 fail=
152 [2023/05/20 21:10:06.742]
```

## 3.3 使用自动化部署实现一键部署

### 3.3.1

持续部署配置。进入项目，单击页面上方导航 " 持续交付 > 部署 "，进入部署服务。找到部署任务 " phoenix-sample-standalone "，点击最右侧，选择 " 编辑 "。

### 3.3.2

选择 " 部署步骤 " 页签，单击部署步骤 " 选择部署来源 "，编译以下信息。

选择源类型：选择 " 构建任务 "。

主机组：选择在添加授信主机中创建的主机组。

请选择构建任务：选择 " phoenix-sample-ci "。

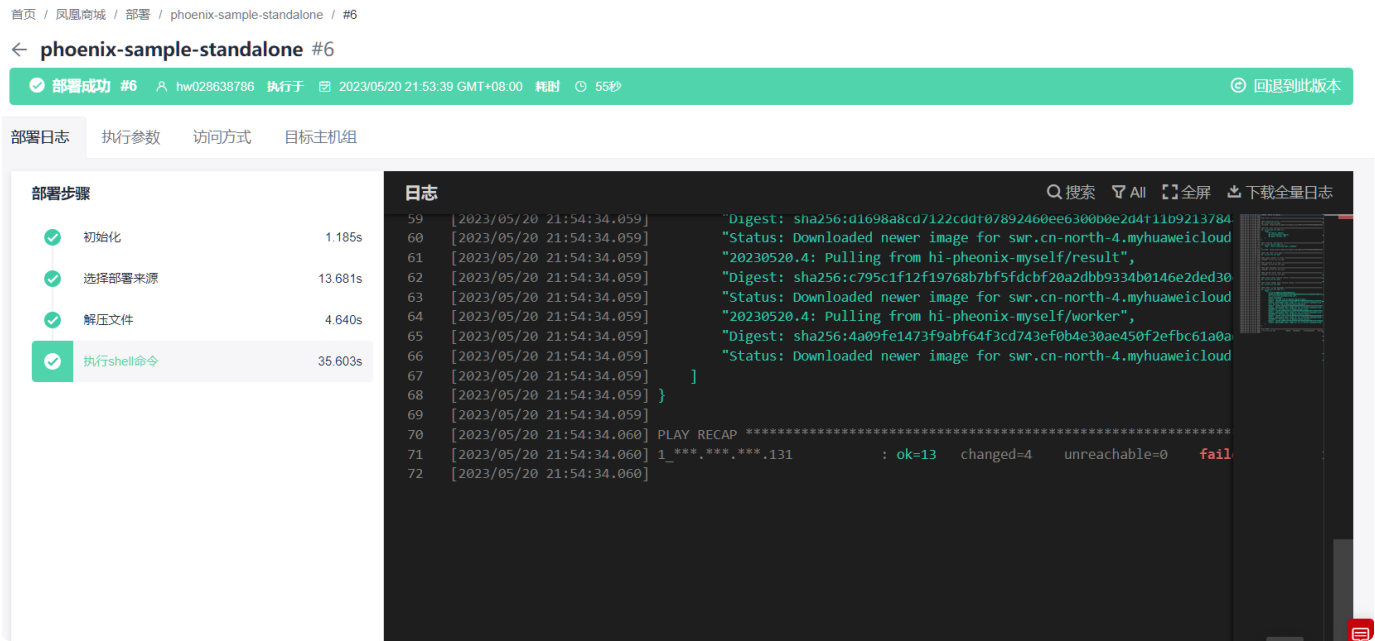
### 3.3.3

选择 " 参数设置 " 页签，根据SWR服务登录指令填写参数。登录指令通过SWR控制台获取，操作方式通过之前在容器镜像服务登录选项中的登录指令获取。

### 3.3.4

单击 " 保存并执行 "，启动部署任务。系统自动跳转至 " 部署详情 " 页面，可以查看任务执行进展。

以下是两个任务的部署情况：



### 3.4 持续交付流水线

#### 3.4.1

创建并触发持续交付流水线。进入项目，单击页面上方导航 " 持续交付> 流水线 "，进入流水线服务。单击 " 新建流水线 "。

#### 3.4.2

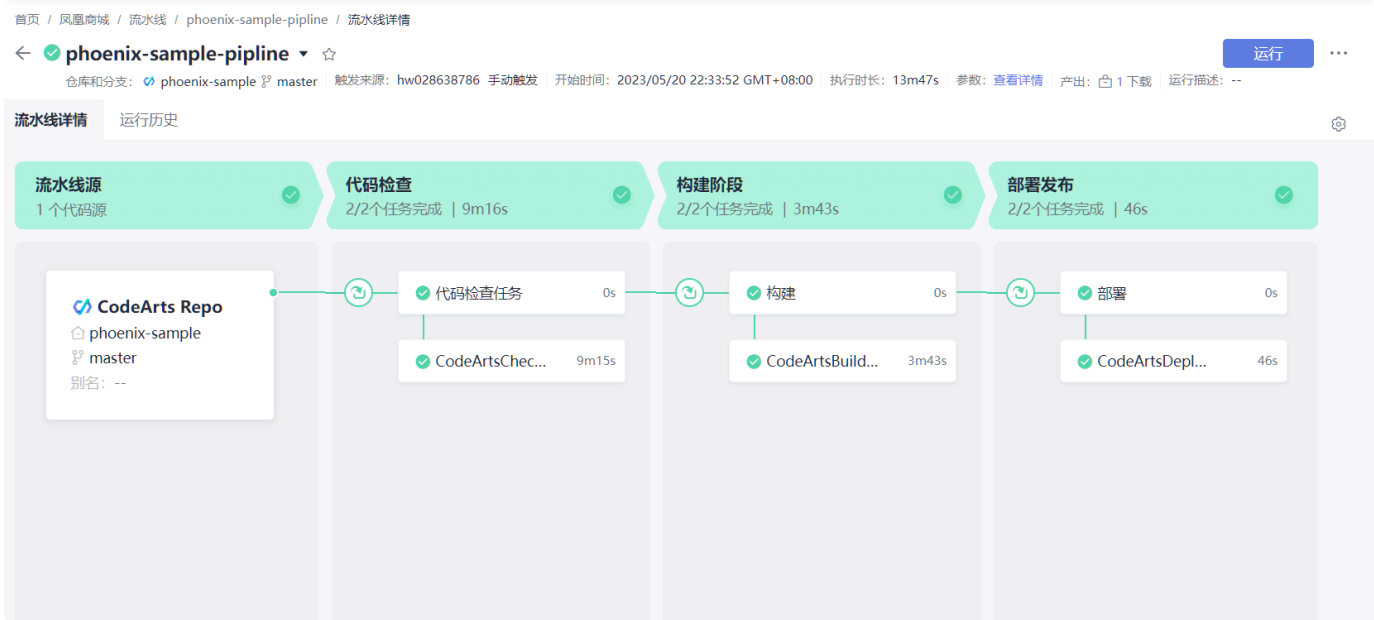
单击阶段 " 代码检查 " 中添加（编辑）任务。在右侧滑出框中，类型选择 " 代码检查 "，任务勾选 " phoenix-codecheck-worker "，单击 " 保存 "。

在 " 构建阶段 "，添加（编辑）构建任务 " phoenix-sample-ci "。

在“部署发布”阶段添加（编辑）任务。

流水线任务阶段设置完成后点击 " 保存并执行 "，启动流水线。

根据实际项目需要添加流水线上的任务，当前项目的流水线任务如下所示：



### 3.4.3

在当前任务的页签中找到执行计划。打开“代码提交时触发”，单击“保存”。

### 3.4.4

添加人工审核。进入流水线任务“phoenix-sample-pipeline”的“流水线配置”页面。在阶段“部署”下，单击“添加任务”。在右侧滑框中，选择类型“流水线控制”，勾选“人工审核”，并在“审核人”下拉列表中勾选所需的审核人。

返回“流水线配置”页面，将任务“流水线控制”拖动到任务“部署”上方，单击“保存”。

启动流水线，当执行到“人工审核”时，页面中将弹框提示任务需审核。





## 实验总结：

通过华为凤凰商城项目，我了解到如何使用DevCloud，实践操作软件的持续规划、开发和部署。学习软件开发过程中的敏捷开发。

在软件项目开发的前期，我们需要进行项目规划，于是使用Scrum进行项目规划。通过思维导图模板和看板的模式，能够更加明确各部门各小组的分工和任务，并且能够清晰的显示相关的负责人员。同时使用Backlog进行迭代开发，能够通过燃尽图判断当前的项目进度。通过DevCloud可以快速进行项目规划。

在软件项目开发阶段，通过DecCloud对代码进行管理。通过部署自动化流水线，实现代码检查、代码自动化编译等功能，最终将大大提高开发效率。同时通过成员审查，可以对于合并请求的代码进行其他成员检查确认，提高了开发的安全性和协同性。

在软件部署阶段，通过在codeArts中关联服务器，并将项目关联到响应的云服务器上，实现自动化部署，并且设置流水线配置，同样可以实现自动化部署。

总而言之，华为的凤凰商城项目让我对于现在企业实际中的项目开发流程有了一定的了解，对于软件开发管理过程有了一定的认识。通过本次实验，我掌握了如何使用华为云平台进行软件项目开发的流程，希望在之后的软件开发学习过程中，继续使用华为云平台进行软件的创新，和团队成员一同创造价值。