第17组 云原生大作业说明文档

> 云原生斗地主 GROUP nju17

项目地址：

👉👉👉github

👉👉👉gitee

小组成员

姓名	学号
张铭铭	211250234
胡家睿	211250020
宋毅恒	211250022

功能要求

实现接口和限流功能

在项目中新建一个`Controller`，实现Rest接口如下：

```
@RestController
public class DemoController {

    private final RateLimiter rateLimiter = RateLimiter.create(100.0);

    @GetMapping("/api/text")
    @ResponseStatus(HttpStatus.OK)
    @ResponseBody
    public String getText(){
        if(!rateLimiter.tryAcquire(1))
            throw new HttpStatusCodeException(HttpStatus.TOO_MANY_REQUESTS) {
            };
        return "{\"name\":\"云原生斗地主\",\"number\":\"nju17\"}";
    }

    @GetMapping( "/api/json")
    @ResponseStatus(HttpStatus.OK)
    @ResponseBody
    public String getJson() {
        if(!rateLimiter.tryAcquire(1)) {
            throw new HttpStatusCodeException(HttpStatus.TOO_MANY_REQUESTS) {};
        }
        JSONObject json = new JSONObject();
        try {
            json.put("name", "云原生斗地主");
```

```
            json.put("number", "nju17");
        } catch (JSONException e) {
            throw new RuntimeException(e);
        }
        String ret = json.toString();
        return ret;
    }
}
```

其中，限流功能使用了`RateLimiter`相关接口来实现，限制每秒最多处理100个请求。如果请求过于频繁，则会返回`429`错误，如下所示（使用`springboot test`测试）：

```
CloudNativeApplicat: 1秒 74毫秒
testTmp()            1秒 74毫秒

测试 失败: 1共 1 个测试 - 1秒 74毫秒

2023-08-15T19:21:28.161+08:00  INFO 50716 --- [           main] c.e.c.CloudNativeApplicationTests
Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bo

com.example.cloudnative.Controller.DemoController$1: 429 TOO_MANY_REQUESTS

    at com.example.cloudnative.Controller.DemoController.getText(DemoController.java:24)
    at com.example.cloudnative.CloudNativeApplicationTests.testTmp(CloudNativeApplicationTests.java:70)
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 个内部行>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <28 个内部行>
```

本地运行后，该接口可以通过访问`http://localhost:8080/api/json` 或 `http://localhost:8080/api/text` 来测试可用性：

localhost:8080/api/json

{"number":"nju17","name":"云原生斗地主"}

实现Prometheus监控

在项目的application.properties中添加如下配置：

vy | CloudNativeApplicationTests.java | 第17组 云原生大作业说明文档.md | application.properties

```
server.address = 0.0.0.0
server.port = 8080

management.endpoints.web.exposure.include=prometheus
management.endpoint.prometheus.enabled=true
management.endpoint.metrics.enabled=true
management.prometheus.metrics.export.enabled=true
management.metrics.tags.application=Cloud-Native
```

并在pom.xml中添加相关依赖：

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
    <groupId>io.micrometer</groupId>
    <artifactId>micrometer-registry-prometheus</artifactId>
</dependency>
```

本地运行后，即可在`http://localhost:8080/actuator/prometheus` 中查看到相关监控信息：

localhost:8080/actuator/prometheus

```
# HELP jvm_gc_max_data_size_bytes Max size of long-lived heap memory pool
# TYPE jvm_gc_max_data_size_bytes gauge
jvm_gc_max_data_size_bytes{application="Cloud-Native",} 4.234149888E9
# HELP jvm_classes_unloaded_classes_total The total number of classes unloaded since the Java virtual machine has started execution
# TYPE jvm_classes_unloaded_classes_total counter
jvm_classes_unloaded_classes_total{application="Cloud-Native",} 0.0
# HELP jvm_memory_committed_bytes The amount of memory in bytes that is committed for the Java virtual machine to use
# TYPE jvm_memory_committed_bytes gauge
jvm_memory_committed_bytes{application="Cloud-Native",area="heap",id="G1 Eden Space",} 2.097152E7
jvm_memory_committed_bytes{application="Cloud-Native",area="nonheap",id="Metaspace",} 3.7224448E7
jvm_memory_committed_bytes{application="Cloud-Native",area="heap",id="G1 Survivor Space",} 4194304.0
jvm_memory_committed_bytes{application="Cloud-Native",area="nonheap",id="Compressed Class Space",} 5505024.0
jvm_memory_committed_bytes{application="Cloud-Native",area="heap",id="G1 Old Gen",} 3.145728E7
jvm_memory_committed_bytes{application="Cloud-Native",area="nonheap",id="CodeCache",} 1.0223616E7
# HELP jvm_buffer_count_buffers An estimate of the number of buffers in the pool
# TYPE jvm_buffer_count_buffers gauge
jvm_buffer_count_buffers{application="Cloud-Native",id="direct",} 7.0
jvm_buffer_count_buffers{application="Cloud-Native",id="mapped",} 0.0
jvm_buffer_count_buffers{application="Cloud-Native",id="mapped - 'non-volatile memory'",} 0.0
# HELP system_cpu_usage The "recent cpu usage" of the system the application is running in
# TYPE system_cpu_usage gauge
system_cpu_usage{application="Cloud-Native",} 0.0
# HELP jvm_memory_max_bytes The maximum amount of memory in bytes that can be used for memory management
# TYPE jvm_memory_max_bytes gauge
jvm_memory_max_bytes{application="Cloud-Native",area="heap",id="G1 Eden Space",} -1.0
jvm_memory_max_bytes{application="Cloud-Native",area="nonheap",id="Metaspace",} -1.0
jvm_memory_max_bytes{application="Cloud-Native",area="heap",id="G1 Survivor Space",} -1.0
jvm_memory_max_bytes{application="Cloud-Native",area="nonheap",id="Compressed Class Space",} 1.073741824E9
jvm_memory_max_bytes{application="Cloud-Native",area="heap",id="G1 Old Gen",} 4.234149888E9
jvm_memory_max_bytes{application="Cloud-Native",area="nonheap",id="CodeCache",} 5.0331648E7
# HELP jvm_gc_live_data_size_bytes Size of long-lived heap memory pool after reclamation
# TYPE jvm_gc_live_data_size_bytes gauge
jvm_gc_live_data_size_bytes{application="Cloud-Native",} 0.0
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{application="Cloud-Native",error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/api/json",} 1.0
http_server_requests_seconds_sum{application="Cloud-Native",error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/api/json",} 0.0407193
http_server_requests_seconds_count{application="Cloud-Native",error="none",exception="none",method="GET",outcome="CLIENT_ERROR",status="404",uri="/**",} 1.0
http_server_requests_seconds_sum{application="Cloud-Native",error="none",exception="none",method="GET",outcome="CLIENT_ERROR",status="404",uri="/**",} 0.0044793
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{application="Cloud-Native",error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/api/json",} 0.0407193
http_server_requests_seconds_max{application="Cloud-Native",error="none",exception="none",method="GET",outcome="CLIENT_ERROR",status="404",uri="/**",} 0.0044793
# HELP tomcat_sessions_active_max_sessions
# TYPE tomcat_sessions_active_max_sessions gauge
tomcat_sessions_active_max_sessions{application="Cloud-Native",} 0.0
# HELP jvm_buffer_memory_used_bytes An estimate of the memory that the Java virtual machine is using for this buffer pool
# TYPE jvm_buffer_memory_used_bytes gauge
jvm_buffer_memory_used_bytes{application="Cloud-Native",id="direct",} 57344.0
jvm_buffer_memory_used_bytes{application="Cloud-Native",id="mapped",} 0.0
```

统一限流

统一限流暂未实现~

DevOps 要求

Dockerfile与K8s容器编排

- Dockerfile

```
FROM openjdk:17

LABEL authors="nju17"

COPY ./release/Cloud-Native-0.0.1-SNAPSHOT.jar /app/Cloud-Native.jar

WORKDIR /app

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "Cloud-Native.jar"]
```

- deployment.yaml

```yaml
apiVersion: apps/v1 #api版本
kind: Deployment
metadata:
  labels: #资源标签
    app: cloud-native
  name: cloud-native #资源名
  namespace: nju17 #资源部署的名空间
spec:
  replicas: 3 #副本数目
  strategy:
    type: RollingUpdate #滚动更新策略
    rollingUpdate:
      maxSurge: 25% #最大额外副本
      maxUnavailable: 25% #更新中进入不可用状态Pod的最大值
  selector:
    matchLabels:
      app: cloud-native #匹配标签
  template:
    metadata:
      annotations: #自定义注解，接入Prometheus
        prometheus.io/path: /actuator/prometheus
        prometheus.io/port: "8080"
        prometheus.io/scheme: http
        prometheus.io/scrape: "true"
      labels:
        app: cloud-native #资源标签
    spec:
      containers:
        - image: harbor.edu.cn/nju17/cloud-native:9 #镜像地址
          name: cloud-native #容器名
---
apiVersion: v1 #api版本
kind: Service
metadata:
  name: cloud-native #资源名
  labels:
    app: cloud-native #资源标签
spec:
  type: NodePort
  selector:
    app: cloud-native
  ports:
    - name: tcp8080 #端口名
      protocol: TCP #TCP协议
      port: 8080 #service端口
      targetPort: 8080 #容器暴露的端口
```

相关注解在代码注释中。截图中的代码适配下述方案二流水线，在Jenkins部分会进行说明。

下述部分是另外一套文件，适配方案一流水线：

- Dockerfile

```
FROM openjdk:17

LABEL authors="nju17"

COPY ./target/Cloud-Native-0.0.1-SNAPSHOT.jar /app/Cloud-Native.jar

WORKDIR /app

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "Cloud-Native.jar"]
```

- deployment.yaml 仅修改图示部分

```
      labels:
        app: cloud-native #资源标签
    spec:
      containers:
        - image: harbor.edu.cn/nju17/cloud-native:{VERSION} #镜像地址
          name: cloud-native #容器名
---
apiVersion: v1 #api版本
kind: Service
metadata:
  name: cloud-native #资源名
```

Jenkins

由于软院Jenkins服务器的master节点一直被占用，因此我们原本做好的Jenkin流水线无法运行，后来临时做了另一条流水线。结果后面master节点又正常工作，又成功运行了原本的方案。两套方案如下：

- 方案一：原本的Jenkin流水线方案，同时使用了master节点和slave节点进行构建，其中master节点完成了持续集成的功能，slave节点完成了持续部署的功能。流水线如下：

```
pipeline {
    agent none
    stages {
        stage('Clone Code') {
            agent {
                label 'master'
            }
            steps {
                echo "1.Clone From Gitee"
                sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
                        'username=' + '211250234' +
                        '&' +
                        'password=' + 'xxxxxxx' + '"' //密码已隐藏
```

```
                git url: 'https://gitee.com/irisalt/cloud-native.git', branch:
'main'
            }
        }

        stage('Maven Build') {
            agent {
                docker {
                    image 'maven:latest'
                    args ' -v /root/.m2:/root/.m2'
                }
            }
            steps {
                echo "2. Using Maven to Build"
                sh 'mvn -B clean package'
            }
        }

        stage('Build Image') {
            agent {
                label 'master'
            }
            steps {
                echo "3. Build Image"
                sh 'docker build -t cloud-native:${BUILD_ID} .'
                sh 'docker tag cloud-native:${BUILD_ID} harbor.edu.cn/nju17/cloud-
native:${BUILD_ID}'
            }
        }

        stage('Push Image') {
            agent {
                label 'master'
            }
            steps {
                echo "4. Push Docker Image"
                sh 'docker login harbor.edu.cn ' +
                        '-u ' + 'nju17' +
                        ' -p ' + 'nju172023'
                sh 'docker push harbor.edu.cn/nju17/cloud-native:${BUILD_ID}'
            }
        }
    }
}

node('slave') {
    container('jnlp-kubectl') {
        stage('Clone & Change YAML') {
            echo "5. Clone YAML to Slave and Change YAML"
            //xxx needs to be replaced
            sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
                    'username=' + '211250234' +
                    '&' +
                    'password=' + 'xxxxxxx' + '"' //密码已隐藏
```

```
            git url: 'https://gitee.com/irisalt/cloud-native.git', branch: 'main'
            sh 'sed -i "s#{VERSION}#${BUILD_ID}#g" deployment.yaml'
        }

        stage ('Deploy') {
            echo "6. Deploy to K8s"
            //xxx needs to be replaced
            sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
                    'username=' + '211250234' +
                    '&' +
                    'password=' + 'xxxxxxx' + '"' //密码已隐藏
            sh 'docker login harbor.edu.cn ' +
                        '-u ' + 'nju17' +
                        ' -p ' + 'nju172023'
            sh 'docker pull harbor.edu.cn/nju17/cloud-native:${BUILD_ID}'
            sh 'kubectl apply -f deployment.yaml -n nju17'
        }

        stage('Monitor') {
            echo "7. Start Monitor"
            sh 'kubectl apply -f monitor.yaml -n monitoring'
        }
    }
}
```

slave节点完成了从镜像仓库拉取镜像，部署到K8s集群的任务。在部署时，会自动修改deployment.yaml文件中的镜像TAG，以实现持续部署的功能。

master节点完成了代码上传，镜像构建，上传到镜像仓库的任务。此外在Maven构建时，已经通过了本地写好的单元测试，如下所示：

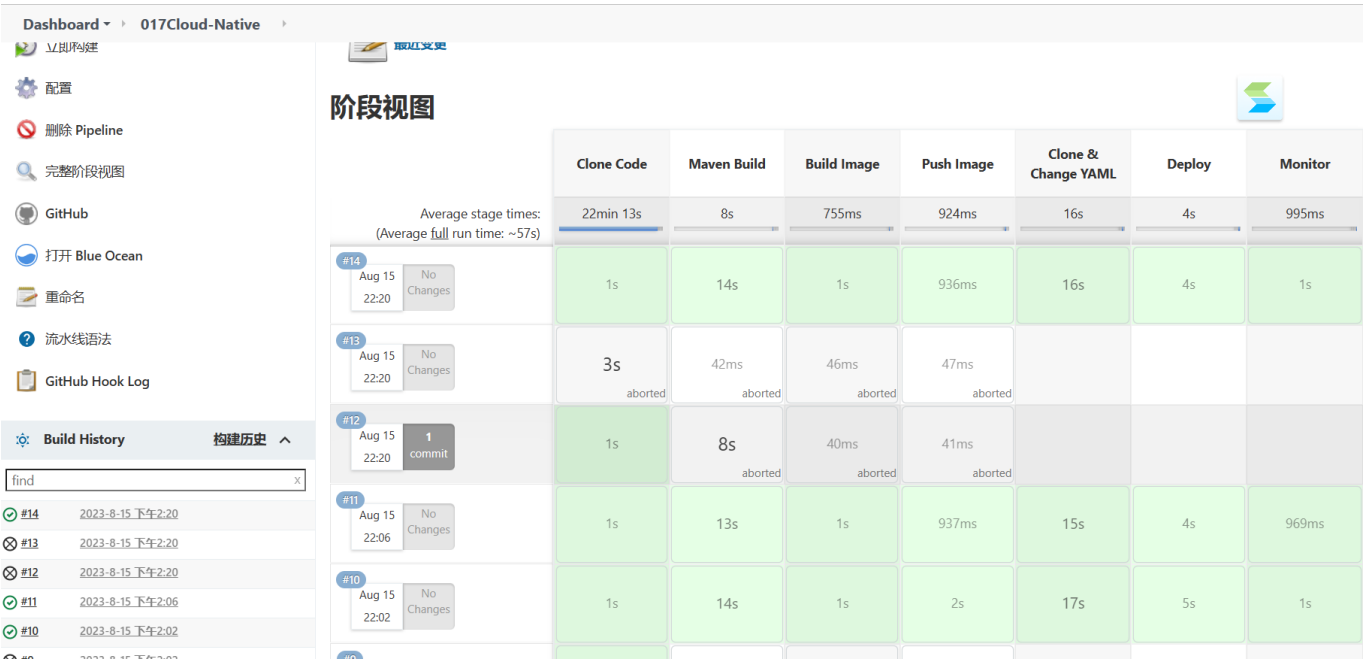
```
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 5.6 s - in com.example.cl
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jar:3.3.0:jar (default-jar) @ Cloud-Native ---
[INFO] Building jar: F:\Cloud-Native\target\Cloud-Native-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot:3.1.2:repackage (repackage) @ Cloud-Native ---
[INFO] Replacing main artifact F:\Cloud-Native\target\Cloud-Native-0.0.1-SNAPSHOT.jar with repac
[INFO] The original artifact has been renamed to F:\Cloud-Native\target\Cloud-Native-0.0.1-SNAPS
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  13.129 s
```

编写的单元测试代码如下：

```
@SpringBootTest
class CloudNativeApplicationTests {
```

```
    private DemoController demoController = new DemoController();
    private static final String EXPECTED_TEXT = "{\"name\":\"云原生斗地主
\",\"number\":\"nju17\"}";
    @BeforeEach
    void initAll() {demoController = new DemoController();}
    @Test
    void testGetText() { // 测试可用性
        String result = demoController.getText();
        assert(result.equals(EXPECTED_TEXT));
    }
    @Test
    void test429(){ // 测试限流功能，应该爆429并且通过测试
        try {
            for(int i = 0; i < 100; i++) {
                Thread.sleep(5);
                demoController.getText();
            }
            assert false;
        } catch (Exception e) {
            assert(e.getMessage().equals("429 TOO_MANY_REQUESTS"));
        }
    }
    @Test
    void test429Two(){ // 测试限流功能，应该爆429并且通过测试
        try {
            for(int i = 0; i < 100; i++) {
                demoController.getText();
            }
            assert false;
        } catch (Exception e) {
            assert(e.getMessage().equals("429 TOO_MANY_REQUESTS"));
        }
    }
    @Test
    void testEdge(){ // 边界测试，应该表现为不会爆429
        try {
            for(int i = 0; i < 100; i++) {
                Thread.sleep(11);
                demoController.getText();
            }
            assert true;
        } catch (Exception e) {
            if(e.getMessage().equals("429 TOO_MANY_REQUESTS")){
                assert false;
            };
            assert true;
        }
    }
}
```

流水线构建结果如下：




```
[nju17@host-172-29-4-18 ~]$ kubectl get pods -n nju17
NAME                              READY   STATUS    RESTARTS   AGE
cloud-native-866946c48c-d74t4     1/1     Running   0          36s
cloud-native-866946c48c-nl2cb     1/1     Running   0          36s
cloud-native-866946c48c-wttzw     1/1     Running   0          36s
[nju17@host-172-29-4-18 ~]$ kubectl get service -n nju17
NAME          TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
cloud-native  NodePort   10.103.149.215   <none>        8080:30713/TCP   39s
[nju17@host-172-29-4-18 ~]$
```

- 方案二：由于准备开始构建流水线的时候master节点都用不了，因此得修改流水线。这是因为master节点和slave节点的功能是不一样的，经过本人测试，在slave节点上我们无法使用`mvn docker`等命令（应该是没有预装相关的环境）。因此小组同时使用了另一套方案：
  - 在本地手动将docker镜像上传到镜像仓库，并手动指定TAG：

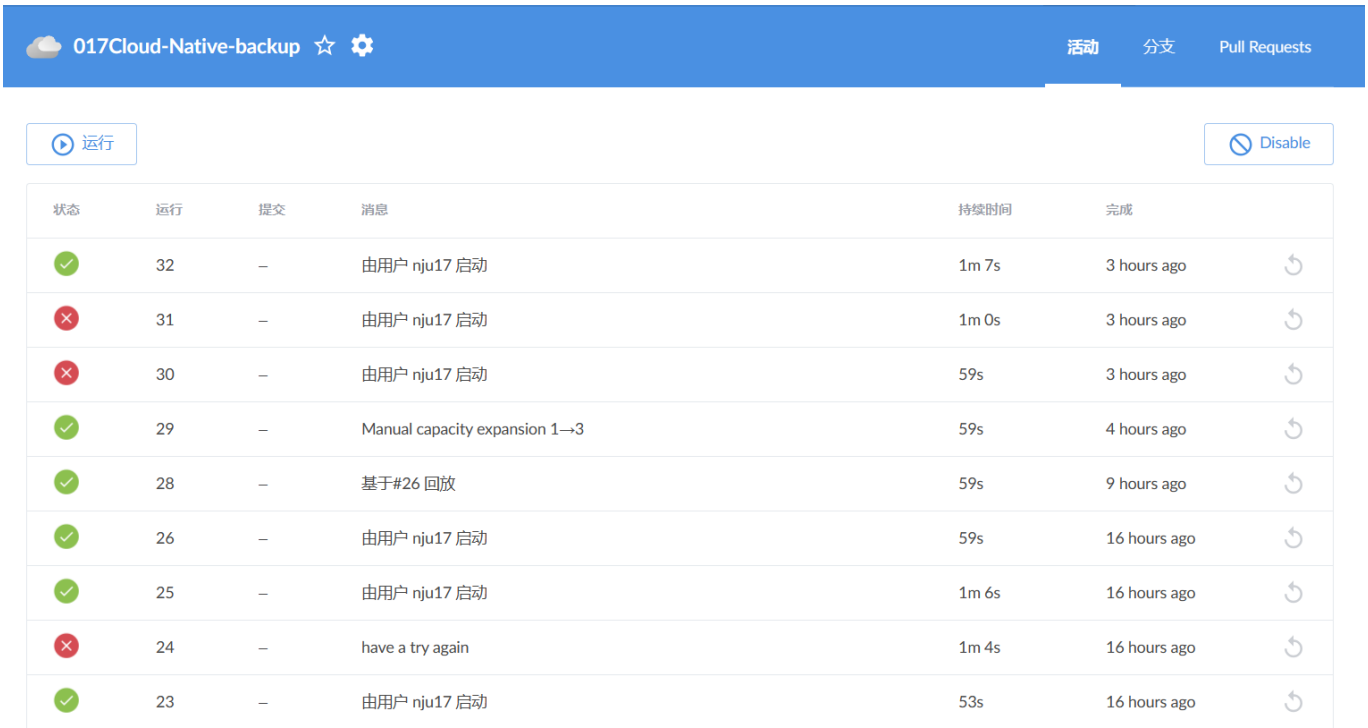
```
ttha@LAPTOP-IRV0R5A3:~/Cloud-Native$ docker login harbor.edu.cn
Username: nju17
Password:
WARNING! Your password will be stored unencrypted in /home/ttha/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ttha@LAPTOP-IRV0R5A3:~/Cloud-Native$ docker build cloud-native:9 .
ERROR: "docker buildx build" requires exactly 1 argument.
See 'docker buildx build --help'.

Usage:  docker buildx build [OPTIONS] PATH | URL | -

Start a build
ttha@LAPTOP-IRV0R5A3:~/Cloud-Native$ docker build -t cloud-native:9 .
[+] Building 31.6s (8/8) FINISHED                                                docker:default
 => [internal] load build definition from Dockerfile                                        0.0s
 => => transferring dockerfile: 235B                                                        0.0s
 => [internal] load .dockerignore                                                           0.0s
 => => transferring context: 2B                                                             0.0s
 => [internal] load metadata for docker.io/library/openjdk:17                              31.4s
 => [internal] load build context                                                           0.0s
 => => transferring context: 90B                                                            0.0s
 => [1/3] FROM docker.io/library/openjdk:17@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f87662e7b3f38cb7  0.0s
 => CACHED [2/3] COPY ./release/Cloud-Native-0.0.1-SNAPSHOT.jar /app/Cloud-Native.jar       0.0s
 => CACHED [3/3] WORKDIR /app                                                               0.0s
 => exporting to image                                                                      0.0s
 => => exporting layers                                                                     0.0s
 => => writing image sha256:c58d5bc28e3708feeced81cb5e73fa54680f2c373ff4aaf861fc8c717915f0b9  0.0s
 => => naming to docker.io/library/cloud-native:9                                           0.0s
ttha@LAPTOP-IRV0R5A3:~/Cloud-Native$ docker tag cloud-native:9 harbor.edu.cn/nju17/cloud-native:9
ttha@LAPTOP-IRV0R5A3:~/Cloud-Native$ docker push harbor.edu.cn/nju17/cloud-native:9
The push refers to repository [harbor.edu.cn/nju17/cloud-native]
5f70bf18a086: Pushed
aaef6f773b7c: Pushed
dc9fa3d8b576: Pushed
27ee19dc88f2: Pushed
c8dd97366670: Pushed
9: digest: sha256:63e943e866a49e420dc1becaa08d659cdb866f83fcc386c9852cbe5be881608b size: 1372
```

- 构建流水线如下，该流水线实际上实现了持续部署的功能：

```
pipeline {
  agent none
  stages {
      stage('Clone Code') {
          agent {
              label 'slave'
          }
          steps {
              echo "1.Clone From Gitee"
              //xxx needs to be replaced
              sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
                      'username=' + '211250234' +
                      '&' +
                      'password=' + 'xxxxxxx' + '"' //密码已隐藏
              git url: 'https://gitee.com/irisalt/cloud-native.git',
branch: 'main'
          }
      }
  }
}

node('slave') {
container('jnlp-kubectl') {
stage('Clone & Change YAML') {
echo "2. Clone YAML to Slave and Change YAML"
//xxx needs to be replaced
sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
'username=' + '211250234' +
'&' +
'password=' + 'xxxxxxx' + '"' //密码已隐藏
git url: 'https://gitee.com/irisalt/cloud-native.git', branch: 'main'
}

    stage ('Deploy') {
        echo "3. Deploy to K8s"
        //xxx needs to be replaced
        sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
                'username=' + '211250234' +
                '&' +
                'password=' + 'xxxxxxx' + '"' //密码已隐藏
        sh 'docker login harbor.edu.cn ' +
                '-u ' + 'nju17' +
                ' -p ' + 'nju172023'
        sh 'docker pull harbor.edu.cn/nju17/cloud-native:9'
        // sh 'kubectl delete deployment cloud-native -n nju17'
        sh 'kubectl apply -f deployment.yaml -n nju17'
        // sh 'kubectl scale deployment cloud-native --replicas 1 -n
nju17'
    }

    stage('Monitor') {
```

```
            echo "4. Start Monitor"
            sh 'kubectl apply -f monitor.yaml -n monitoring'
        }
    }
}
```

实际运行的情况如下，访问的url为`http://172.29.4.18:32510/`：




```
Tolerations:     node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                 node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:          <none>
[nju17@host-172-29-4-18 ~]$ kubectl get pods -n nju17
NAME                          READY   STATUS    RESTARTS   AGE
cloud-native-866946c48c-55bjv   1/1     Running   0          3m46s
[nju17@host-172-29-4-18 ~]$ kubectl get pods -n nju17
NAME                          READY   STATUS    RESTARTS   AGE
cloud-native-866946c48c-55bjv   1/1     Running   0          4m45s
[nju17@host-172-29-4-18 ~]$ kubectl get service -n nju17
NAME           TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
cloud-native   NodePort   10.107.125.168   <none>        8080:32510/TCP   26s
[nju17@host-172-29-4-18 ~]$
```

不安全 | 172.29.4.18:32510/actuator/prometheus

```
# HELP tomcat_sessions_active_current_sessions
# TYPE tomcat_sessions_active_current_sessions gauge
tomcat_sessions_active_current_sessions{application="Cloud-Native",} 0.0
# HELP application_started_time_seconds Time taken (ms) to start the application
# TYPE application_started_time_seconds gauge
application_started_time_seconds{application="Cloud-Native",main_application_class="com.example.cloudnative.CloudNativeApplication",} 4.039
# HELP disk_free_bytes Usable space for path
# TYPE disk_free_bytes gauge
disk_free_bytes{application="Cloud-Native",path="/app/.",} 2.03615039488E11
# HELP tomcat_sessions_created_sessions_total
# TYPE tomcat_sessions_created_sessions_total counter
tomcat_sessions_created_sessions_total{application="Cloud-Native",} 0.0
# HELP executor_pool_size_threads The current number of threads in the pool
# TYPE executor_pool_size_threads gauge
executor_pool_size_threads{application="Cloud-Native",name="applicationTaskExecutor",} 0.0
# HELP application_ready_time_seconds Time taken (ms) for the application to be ready to service requests
# TYPE application_ready_time_seconds gauge
application_ready_time_seconds{application="Cloud-Native",main_application_class="com.example.cloudnative.CloudNativeApplication",} 4.114
# HELP jvm_classes_unloaded_classes_total The total number of classes unloaded since the Java virtual machine has started execution
# TYPE jvm_classes_unloaded_classes_total counter
jvm_classes_unloaded_classes_total{application="Cloud-Native",} 0.0
# HELP system_cpu_usage The "recent cpu usage" of the system the application is running in
# TYPE system_cpu_usage gauge
system_cpu_usage{application="Cloud-Native",} 0.004780958398343114
# HELP tomcat_sessions_active_max_sessions
# TYPE tomcat_sessions_active_max_sessions gauge
tomcat_sessions_active_max_sessions{application="Cloud-Native",} 0.0
# HELP executor_queue_remaining_tasks The number of additional elements that this queue can ideally accept without blocking
# TYPE executor_queue_remaining_tasks gauge
executor_queue_remaining_tasks{application="Cloud-Native",name="applicationTaskExecutor",} 2.147483647E9
# HELP tomcat_sessions_rejected_sessions_total
# TYPE tomcat_sessions_rejected_sessions_total counter
tomcat_sessions_rejected_sessions_total{application="Cloud-Native",} 0.0
# HELP http_server_requests_active_seconds_max
# TYPE http_server_requests_active_seconds_max gauge
http_server_requests_active_seconds_max{application="Cloud-Native",exception="none",method="GET",outcome="SUCCESS",status="200",uri="UNKNOWN",} 0.004912892
# HELP http_server_requests_active_seconds
# TYPE http_server_requests_active_seconds summary
http_server_requests_active_seconds_active_count{application="Cloud-Native",exception="none",method="GET",outcome="SUCCESS",status="200",uri="UNKNOWN",} 1.0
http_server_requests_active_seconds_duration_sum{application="Cloud-Native",exception="none",method="GET",outcome="SUCCESS",status="200",uri="UNKNOWN",} 0.004899692
# HELP jvm_threads_states_threads The current number of threads
# TYPE jvm_threads_states_threads gauge
jvm_threads_states_threads{application="Cloud-Native",state="terminated",} 0.0
jvm_threads_states_threads{application="Cloud-Native",state="runnable",} 7.0
jvm_threads_states_threads{application="Cloud-Native",state="new",} 0.0
jvm_threads_states_threads{application="Cloud-Native",state="blocked",} 0.0
jvm_threads_states_threads{application="Cloud-Native",state="waiting",} 11.0
jvm_threads_states_threads{application="Cloud-Native",state="timed-waiting",} 3.0
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
```

不安全 | 172.29.4.18:32510/api/json

```
{"number":"nju17","name":"云原生斗地主"}
```

扩容场景

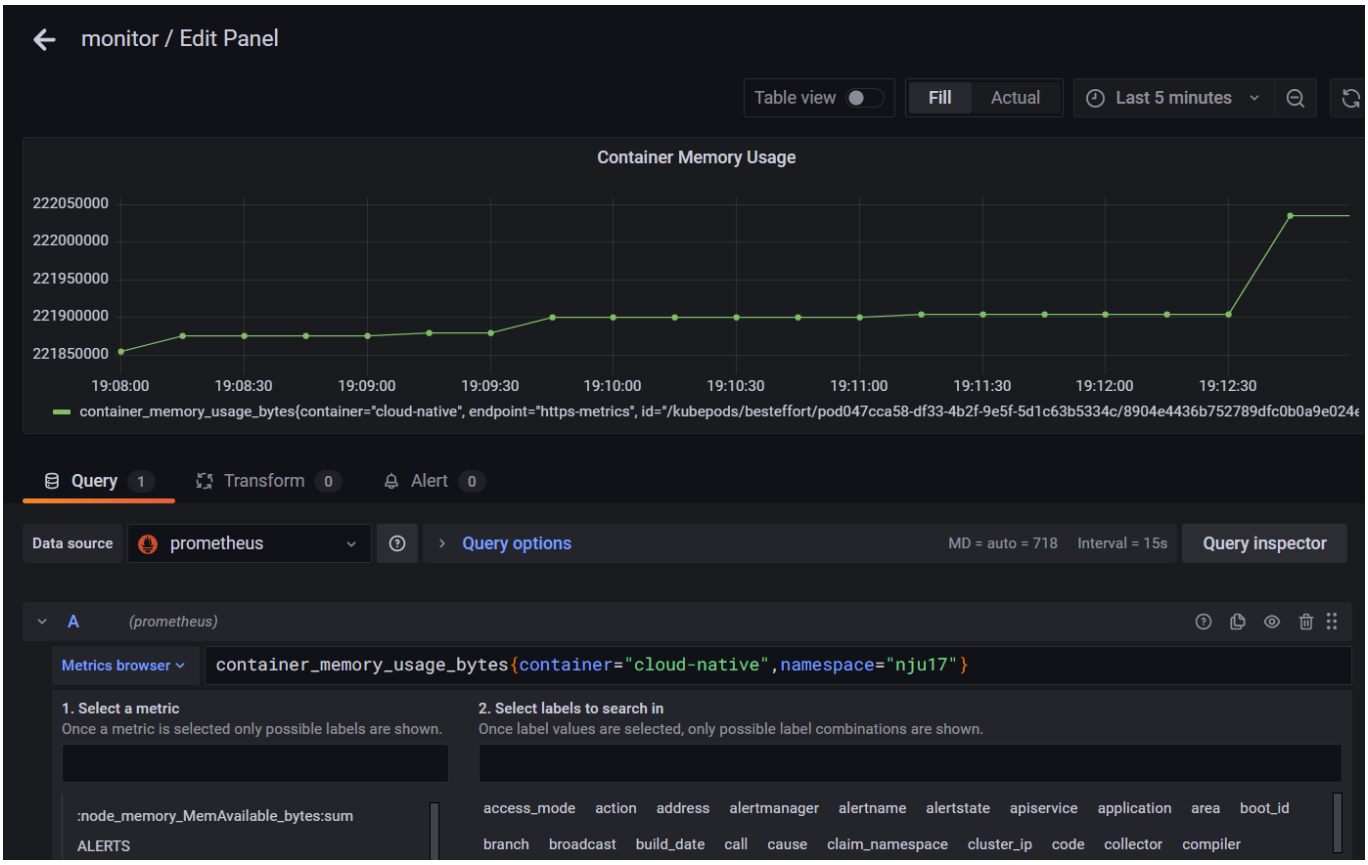
Prometheus metrics接口

配置一个ServiceMonitor，用于监控应用的metrics接口，访问的url为`http://172.29.4.18:32510/actuator/prometheus`，配置如下图所示：

```yaml
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    k8s-app: cloud-native
  name: cloud-native
  namespace: monitoring
spec:
  endpoints:
    - interval: 30s #监控数据抓取间隔设为30s
      port: tcp8080 #端口名
      path: /actuator/prometheus
      scheme: 'http' #接口协议
  selector:
    matchLabels:
      app: cloud-native #监控目标Service的标签
  namespaceSelector:
    matchNames:
      - nju17
```

Grafana监控

流水线部署完成后，在软件研发效能支撑平台的grafana平台上可以直接查询到对应的容器和名空间。
通过可视化工具可以直接创建所需的图表（如下图）。

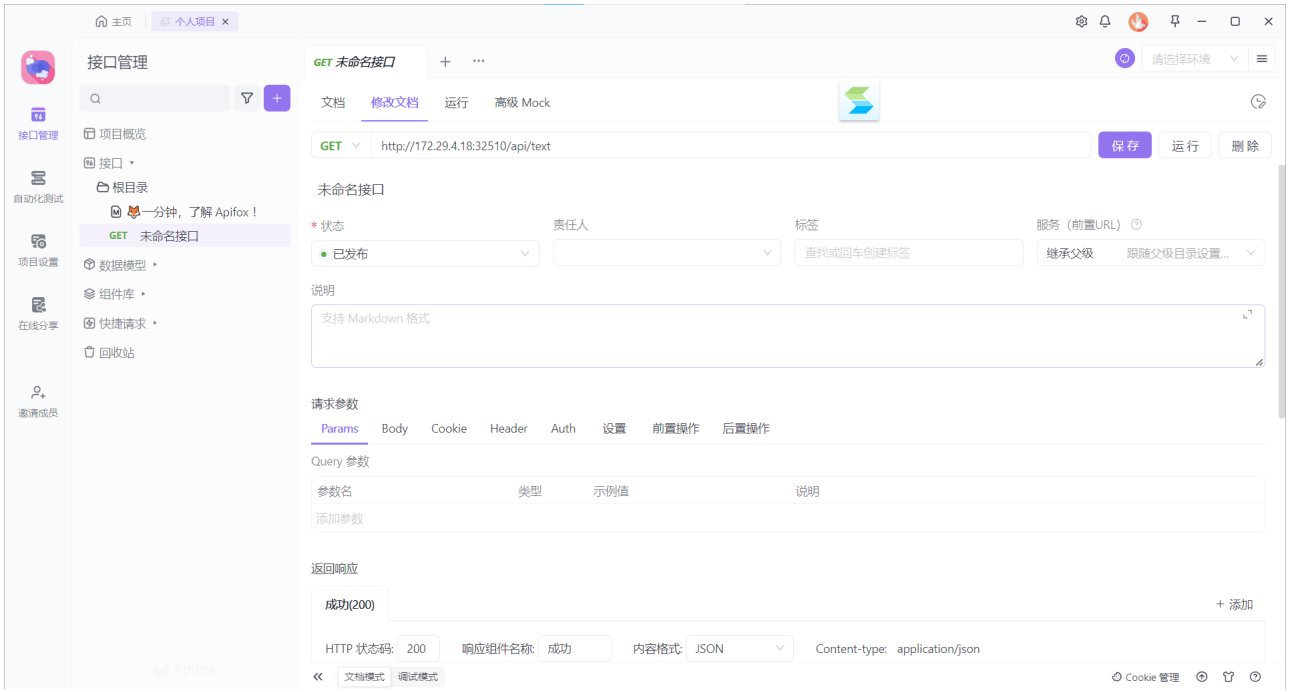


通过此方式创建CPU、内存、JVM的空间使用图表（如下图）。

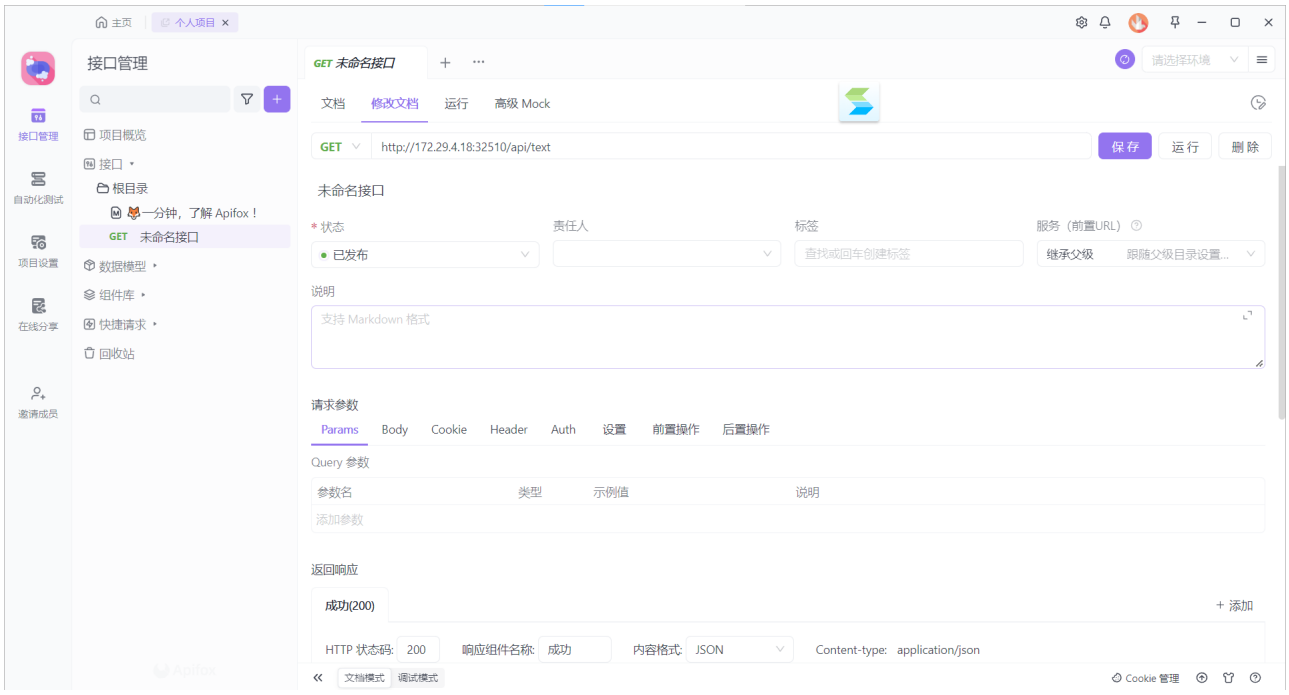
压力测试

- 使用Apifox进行压力测试

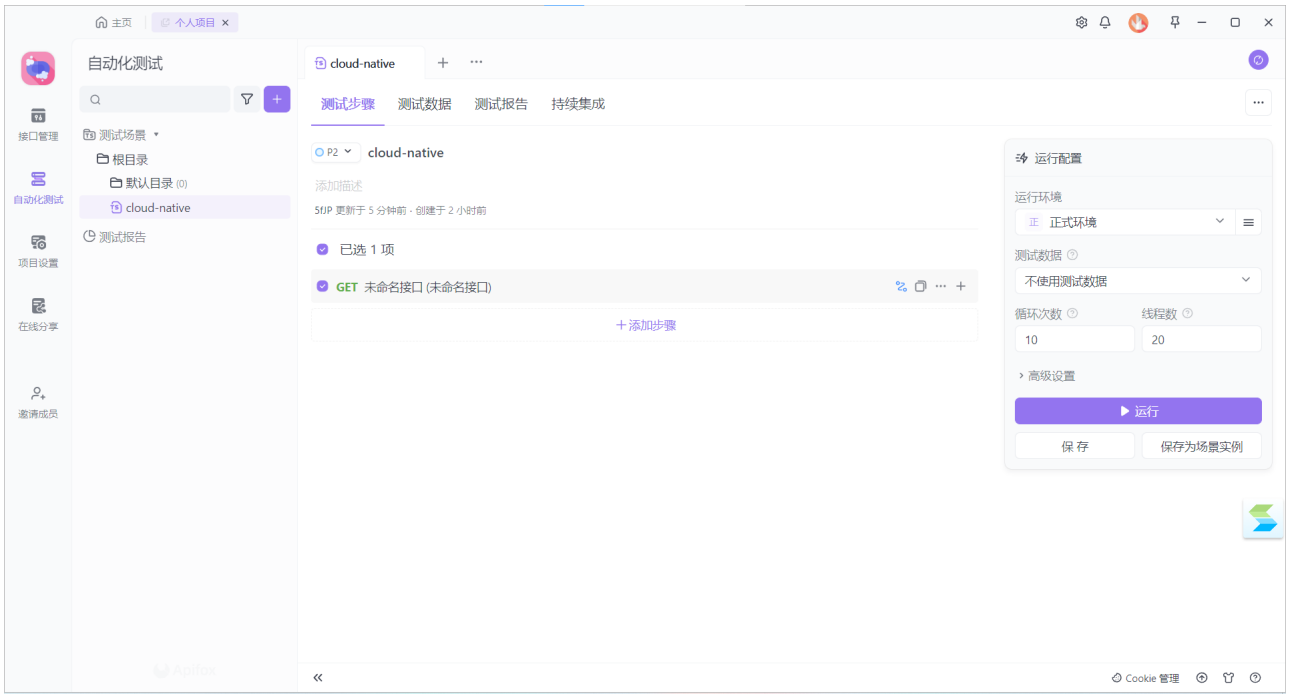
  在Apifox中添加实现的接口（如下图）。



  使用Apifox的自动化测试功能,设置循环次数为10次，线程数为20(如下图)。

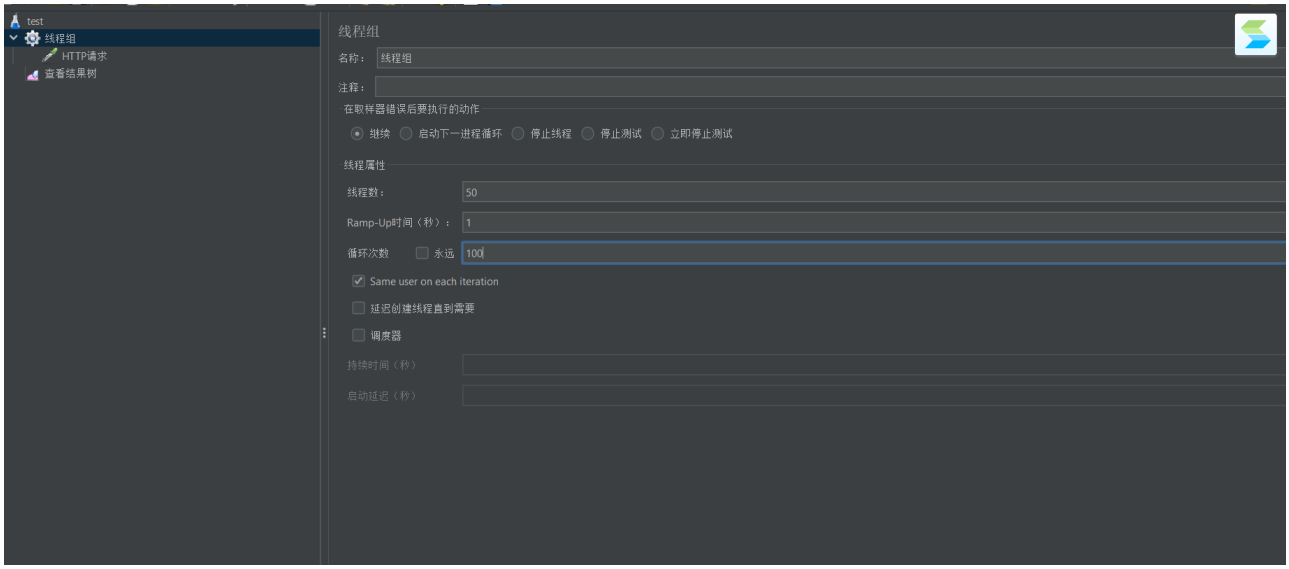


测试完成后可以查看grafana中容器内存使用有明显上升（由于网络限制测试时间花费较长，故没有触发限流）。

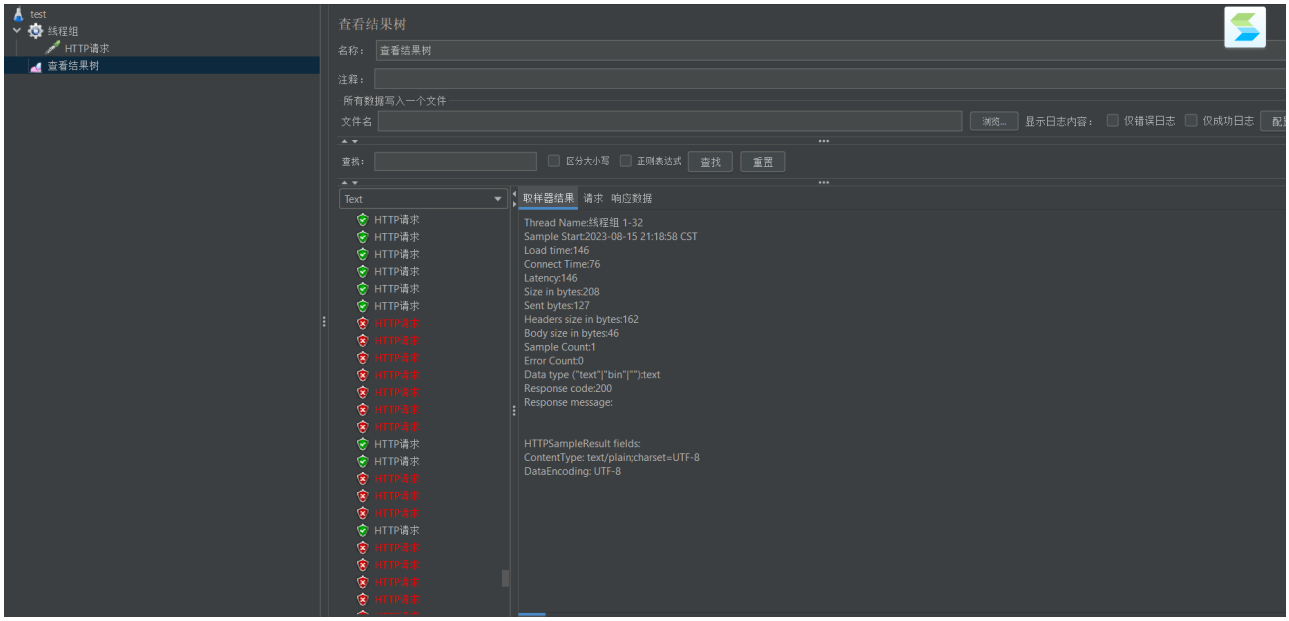


- 使用Apache JMeter进行压力测试

由于ApiFox的测试时间较长，没有达到理想的效果，故再采用Apache JMeter进行压力测试。



测试完成后可以看到出现了429错误，说明限流功能正常。



查看监控大屏，大屏出现了明显的资源占用上升：

手工扩容

可以修改deployment.yaml文件中replicas的值，再次部署。此处采取修改流水线的方式，直接修改Deploy阶段，增加对replicas的修改：

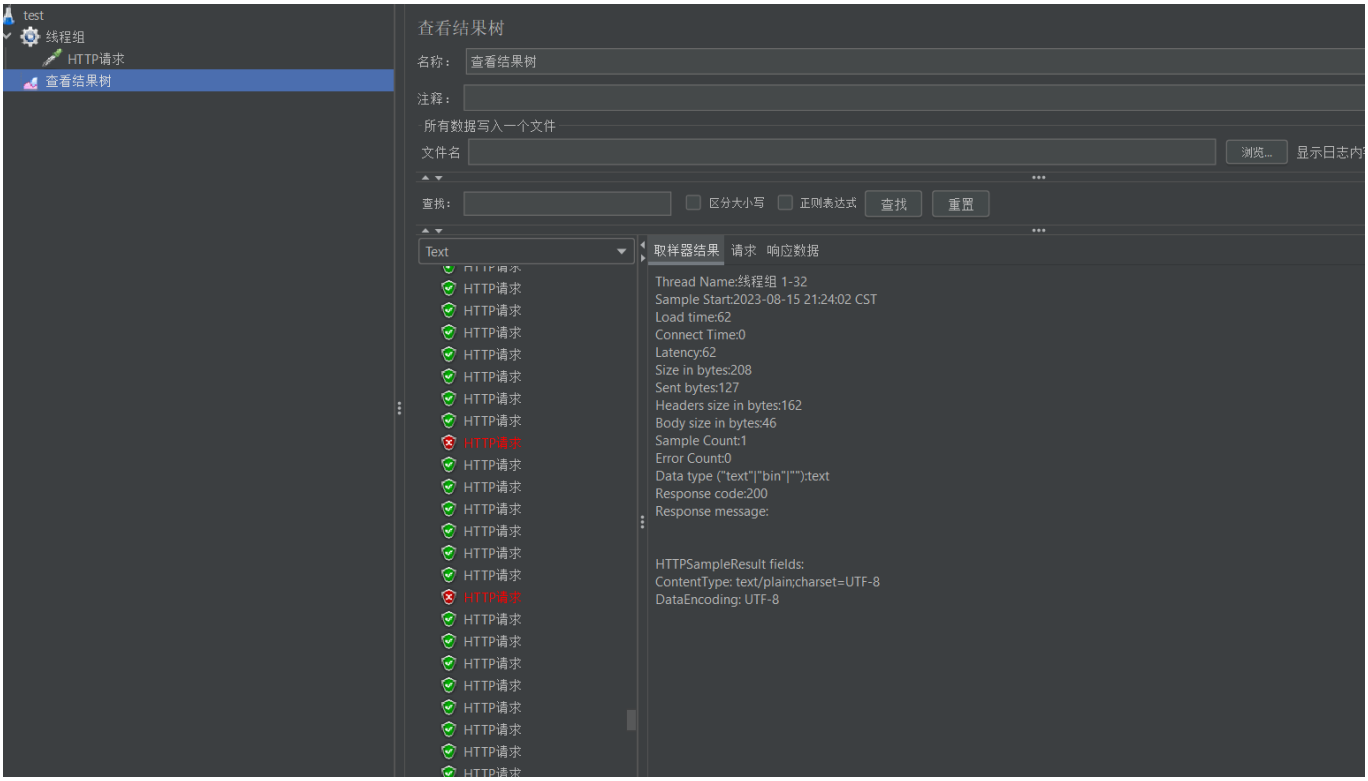
```
// 其余部分省流

stage ('Deploy') {
    echo "3. Deploy to K8s"
    sh 'curl "http://p.nju.edu.cn/portal_io/login?' +
            'username=' + '211250234' +
            '&' +
            'password=' + 'xxxxxxx' + '"' //密码已隐藏
    sh 'docker login harbor.edu.cn ' +
            '-u ' + 'nju17' +
            ' -p ' + 'nju172023'
```

```
        sh 'docker pull harbor.edu.cn/nju17/cloud-native:9'
        // sh 'kubectl delete deployment cloud-native -n nju17'
        sh 'kubectl apply -f deployment.yaml -n nju17'
         sh 'kubectl scale deployment cloud-native --replicas 3 -n nju17'
    }
```

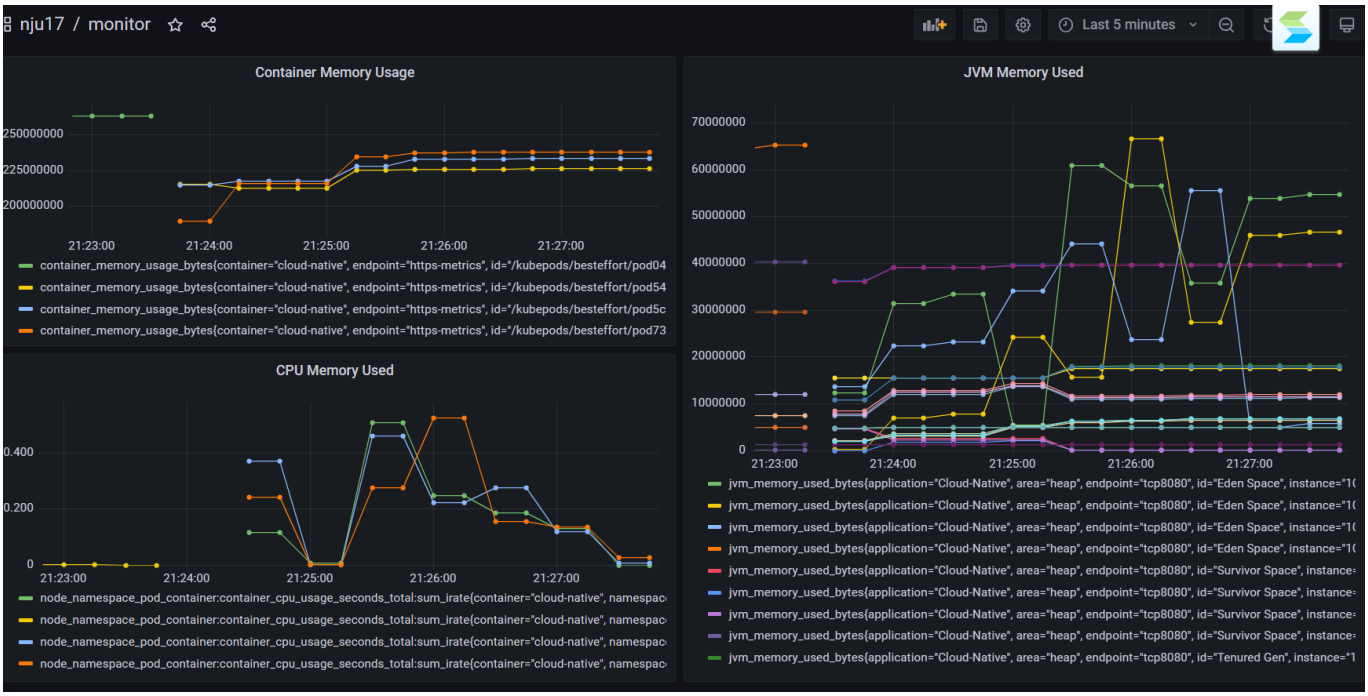
增加`sh 'kubectl scale deployment cloud-native --replicas 3 -n nju17'`即可在部署时实现扩容，扩容结果如下。

```
[nju17@host-172-29-4-18 ~]$ kubectl get pods -n nju17
NAME                              READY   STATUS    RESTARTS   AGE
cloud-native-866946c48c-jqm8l     1/1     Running   0          3h26m
[nju17@host-172-29-4-18 ~]$ kubectl get pods -n nju17
NAME                              READY   STATUS    RESTARTS   AGE
cloud-native-866946c48c-c7559     1/1     Running   0          12s
cloud-native-866946c48c-p2xxl     1/1     Running   0          12s
cloud-native-866946c48c-v55bt     1/1     Running   0          12s
```

扩容后，再次对接口进行压力测试，可以看到压力测试的结果如下，请求成功的次数大大增加：



监控大屏也可以观察到相应的变化，container变为3个，曲线呈上升趋势：



扩容前后对比：

自动扩容

自动扩容不写了~