Lab2 Kubernetes实践报告

小组分工：

> 普昕：任务一二三
>
> 谢升楼：任务四五
>
> 肖羽平：任务六

1.使用minikube搭建kubernetes集群

运行 `minikube start --driver=docker`

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get po -A
NAMESPACE     NAME                               READY   STATUS    RESTARTS   AGE
kube-system   coredns-66bc5c9577-6d2x8           1/1     Running   0          9m8s
kube-system   coredns-66bc5c9577-mggjv           1/1     Running   0          9m8s
kube-system   etcd-minikube                      1/1     Running   0          9m13s
kube-system   kube-apiserver-minikube            1/1     Running   0          9m13s
kube-system   kube-controller-manager-minikube   1/1     Running   0          9m15s
kube-system   kube-proxy-6f5nf                   1/1     Running   0          9m8s
kube-system   kube-scheduler-minikube            1/1     Running   0          9m13s
kube-system   storage-provisioner                1/1     Running   0          9m11s
```

2. 在Kubernetes集群中部署中间件

在gomall/k8s/middlewares中添加 Kubernetes manifests，包括 Redis、NATS 的 Deployment/Service， MySQL 的 ConfigMap、PV/PVC、StatefulSet、Service：

- nats-deployment.yaml -> NATS Deployment
- nats-service.yaml -> NATS Service
- mysqLconfigmap.yaml -> MySQL初始化 SQL
- mysqLpvpvc.yaml -> PersistentVolume 与PersistentVolumeClaim (hostPath)
- mysql-headless-service.yaml -> MySQL headless Service (StatefulSet 使用)
- mysql-service.yaml -> MySQL ClusterlP service
- mysql-statefulset.yaml -> MySQL StatefulSet(挂载 PVC 并使用 ConfigMap 初始化)

在集群中应用这些文件

```
minikube kubectl -- apply -f gomall/k8s/middlewares
```

```
PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- apply -f gomall/k8s/middlewares ; minikube kubectl -- get pods -o wide ; minikube kubectl
et statefulset,deploy,svc,pv,pvc
configmap/mysql-init-sql created
nats-dbfbf4fdb-psmwr     0/1     ContainerCreating   0          1s    <none>   minikube   <none>           <none>
redis-7699f47487-xh8cx   0/1     ContainerCreating   0          1s    <none>   minikube   <none>           <none>
NAME                     READY   AGE
statefulset.apps/mysql   0/1     1s

NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nats    0/1     1            0           1s
deployment.apps/redis   0/1     1            0           1s

NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
service/kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP             15m
service/mysql           ClusterIP   10.106.170.172   <none>        3306/TCP            1s
service/mysql-headless  ClusterIP   None             <none>        3306/TCP            1s
service/nats            ClusterIP   10.110.158.2     <none>        4222/TCP,8222/TCP   1s
service/redis           ClusterIP   10.107.167.78    <none>        6379/TCP            1s

NAME                     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM               STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
persistentvolume/mysql-pv   5Gi        RWO            Retain           Bound    default/mysql-pvc   manual         <unset>                          1s

NAME                           STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/mysql-pvc   Bound    mysql-pv   5Gi        RWO            manual         <unset>                 1s
```

运行 `minikube kubectl -- get po -A`，查看pod都处于Running状态，说明三个中间件的容器已经成功启动。

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get po -A
NAMESPACE     NAME                               READY   STATUS    RESTARTS      AGE
default       mysql-0                            1/1     Running   1 (40m ago)   3h36m
default       nats-dbfbf4fdb-psmwr               1/1     Running   1 (40m ago)   3h36m
default       redis-7699f47487-xh8cx             1/1     Running   1 (40m ago)   3h36m
kube-system   coredns-66bc5c9577-6d2x8           1/1     Running   1 (40m ago)   3h51m
kube-system   coredns-66bc5c9577-mggjv           1/1     Running   1 (40m ago)   3h51m
kube-system   etcd-minikube                      1/1     Running   1 (40m ago)   3h51m
kube-system   kube-apiserver-minikube            1/1     Running   1 (40m ago)   3h51m
kube-system   kube-controller-manager-minikube   1/1     Running   1 (40m ago)   3h51m
kube-system   kube-proxy-6f5nf                   1/1     Running   1 (40m ago)   3h51m
kube-system   kube-scheduler-minikube            1/1     Running   1 (40m ago)   3h51m
kube-system   storage-provisioner                1/1     Running   1 (40m ago)   3h51m
```

运行 `minikube kubectl -- get svc -n default`，检查 `mysql`,`nats`,和 `redis`已经创建了 `ClusterIP` 类型的 `Service`，说明它们可以在集群内部被访问。

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get svc -n default
NAME             TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
kubernetes       ClusterIP   10.96.0.1        <none>        443/TCP             3h55m
mysql            ClusterIP   10.106.170.172   <none>        3306/TCP            3h40m
mysql-headless   ClusterIP   None             <none>        3306/TCP            3h40m
nats             ClusterIP   10.110.158.2     <none>        4222/TCP,8222/TCP   3h40m
redis            ClusterIP   10.107.167.78    <none>        6379/TCP            3h40m
```

运行 `minikube kubectl -- get cm -n default`，已经创建了 `mysql-init-sql`。

运行 `minikube kubectl -- describe statefulset mysql -n default` 的输出进一步确认了这个 `ConfigMap` 被正确挂载到了容器的 `/docker-entrypoint-initdb.d/init.sql` 路径，用于初始化。

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get cm -n default
NAME               DATA   AGE
kube-root-ca.crt   1      3h55m
mysql-init-sql     1      3h40m
```

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- describe statefulset mysql -n default
Name:               mysql
Namespace:          default
CreationTimestamp:  Wed, 26 Nov 2025 13:40:29 +0800
Selector:           app=mysql
Labels:             app=mysql
Annotations:        <none>
Replicas:           1 desired | 1 total
Update Strategy:    RollingUpdate
Pods Status:        1 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=mysql
  Containers:
   mysql:
    Image:      mysql:8.0
    Port:       3306/TCP
    Host Port:  0/TCP
    Environment:
      MYSQL_ROOT_PASSWORD:  root123
      MYSQL_DATABASE:       gomall
      MYSQL_USER:           gomall
      MYSQL_PASSWORD:       gomall123
    Mounts:
      /docker-entrypoint-initdb.d/init.sql from mysql-init-sql (rw,path="init.sql")
      /var/lib/mysql from mysql-data (rw)
  Volumes:
   mysql-init-sql:
    Type:      ConfigMap (a volume populated by a ConfigMap)
    Name:      mysql-init-sql
    Optional:  false
   mysql-data:
    Type:          PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
    ClaimName:     mysql-pvc
    ReadOnly:      false
  Node-Selectors:  <none>
  Tolerations:     <none>
Volume Claims:     <none>
Events:
  Type    Reason            Age    From                    Message
  ----    ------            ----   ----                    -------
  Normal  SuccessfulCreate  3h40m  statefulset-controller  create Pod mysql-0 in StatefulSet mysql successful
```

综上，MySQL 已经实现了数据持久化。get pvc 显示已经创建了mysql-pvc 的 PersistentVolumeClaim，它的状态是 Bound，说明它成功绑定到了一个名为 `mysql-pv` 的 PersistentVolume。describe statefulset 确认了这个 mysql-pvc 被用作 mysql-data 卷，并挂载到了容器的 `/var/lib/mysql` 目录。

3. 在Kubernetes集群中部署gomall

总体思路

为每个微服务创建 `ConfigMap`（配置）、`Deployment`、`Service`、以及必要的 `Volume`/`PVC`，使得服务在集群内互相发现并稳定运行。

要点

- 微服务通过 `<中间件的服务名>:<端口号>` 访问中间件。在 `ConfigMap` 中把中间件地址设置为服务名：
  - MySQL: `address: mysql`、`port: 3306`
  - Redis: `address: "redis:6379"`
  - NATS: `url: "nats://nats:4222"`
- 使用 `ConfigMap` 存储微服务配置并挂载到容器中。每个微服务 YAML 文件顶部定义了一个 `ConfigMap`，并在 `Deployment.spec.template.spec.volumes` / `volumeMounts` 中以 `subPath` 的方式挂载到 conf.yaml。这样能在不重建镜像的情况下更新配置。

在集群中应用微服务清单

```
minikube kubectl -- apply -f gomall\k8s\microservices
```

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- apply -f gomall/k8s/microservices
configmap/cart-conf created
deployment.apps/cart created
service/cart created
configmap/checkout-conf created
deployment.apps/checkout created
service/checkout created
configmap/email-conf created
deployment.apps/email created
service/email created
configmap/frontend-conf created
deployment.apps/frontend created
service/frontend created
configmap/order-conf created
deployment.apps/order created
service/order created
configmap/payment-conf created
deployment.apps/payment created
service/payment created
configmap/product-conf created
deployment.apps/product created
service/product created
configmap/user-conf created
deployment.apps/user created
service/user created
```

运行 `minikube kubectl -- get pods -o wide`，可能会出现一些pod起不来的情况

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get pods -o wide
NAME                       READY   STATUS             RESTARTS      AGE     IP             NODE       NOMINATED NODE   READINESS GATES
cart-79f7f79fd9-rcchw      0/1     CrashLoopBackOff   4 (64s ago)   3m23s   10.244.0.14    minikube   <none>           <none>
checkout-6cf865d4bd-h8www  1/1     Running            0             3m23s   10.244.0.13    minikube   <none>           <none>
email-c65b8b4b-9kj9h       1/1     Running            0             3m23s   10.244.0.12    minikube   <none>           <none>
frontend-75d7bd8dbd-8ht8s  1/1     Running            0             3m23s   10.244.0.15    minikube   <none>           <none>
mysql-0                    1/1     Running            1 (77m ago)   4h13m   10.244.0.8     minikube   <none>           <none>
nats-dbfbf4fdb-psmwr       1/1     Running            1 (77m ago)   4h13m   10.244.0.9     minikube   <none>           <none>
order-7985b9787c-8dm62     0/1     Error              4 (55s ago)   3m23s   10.244.0.16    minikube   <none>           <none>
payment-85d74754d-nr86c    0/1     CrashLoopBackOff   3 (15s ago)   3m22s   10.244.0.18    minikube   <none>           <none>
product-56cc57768f-ddgg4   0/1     CrashLoopBackOff   3 (39s ago)   3m22s   10.244.0.17    minikube   <none>           <none>
redis-7699f47487-xh8cx     1/1     Running            1 (77m ago)   4h13m   10.244.0.11    minikube   <none>           <none>
user-7d78cbc4cd-bfb7x      0/1     Error              3 (29s ago)   3m21s   10.244.0.19    minikube   <none>           <none>
```

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- logs cart-79f7f79fd9-rcchw
/app/bin/cart
&{Env:dev Kitex:{Service:cart Address::8883 MetricsPort::9993 EnablePprof:false EnableGzip:false EnableAccessLog:false LogLevel:info LogFileName: Lo
gMaxSize:0 LogMaxBackups:0 LogMaxAge:0} MySQL:{DSN:%s:%s@tcp(%s:%d)/%s?charset=utf8mb4&parseTime=True&loc=Local Username:gomall Password:gomall123 A
ddress:mysql Port:3306 Database:cart} Redis:{Address: Username: Password: DB:0} RateLimiter:{Enabled:false Rate:0 BucketSize:0}}

2025/11/26 09:54:53 /gomall/app/cart/biz/dal/mysql/init.go:33
[error] failed to initialize database, got error Error 1044 (42000): Access denied for user 'gomall'@'%' to database 'cart'
panic: Error 1044 (42000): Access denied for user 'gomall'@'%' to database 'cart'

goroutine 1 [running]:
github.com/cloudwego/biz-demo/gomall/app/cart/biz/dal/mysql.Init()
        /gomall/app/cart/biz/dal/mysql/init.go:47 +0x495
github.com/cloudwego/biz-demo/gomall/app/cart/biz/dal.Init(...)
        /gomall/app/cart/biz/dal/init.go:23
main.main()
        /gomall/app/cart/main.go:37 +0x2b
```

运行 `minikube kubectl -- logs cart-79f7f79fd9-rcchw`查看cart 日志，发现 MySQL 还没有为 cart 等服务创建数据库并且授予 gomall 对这些数据库的权限。

优化一下

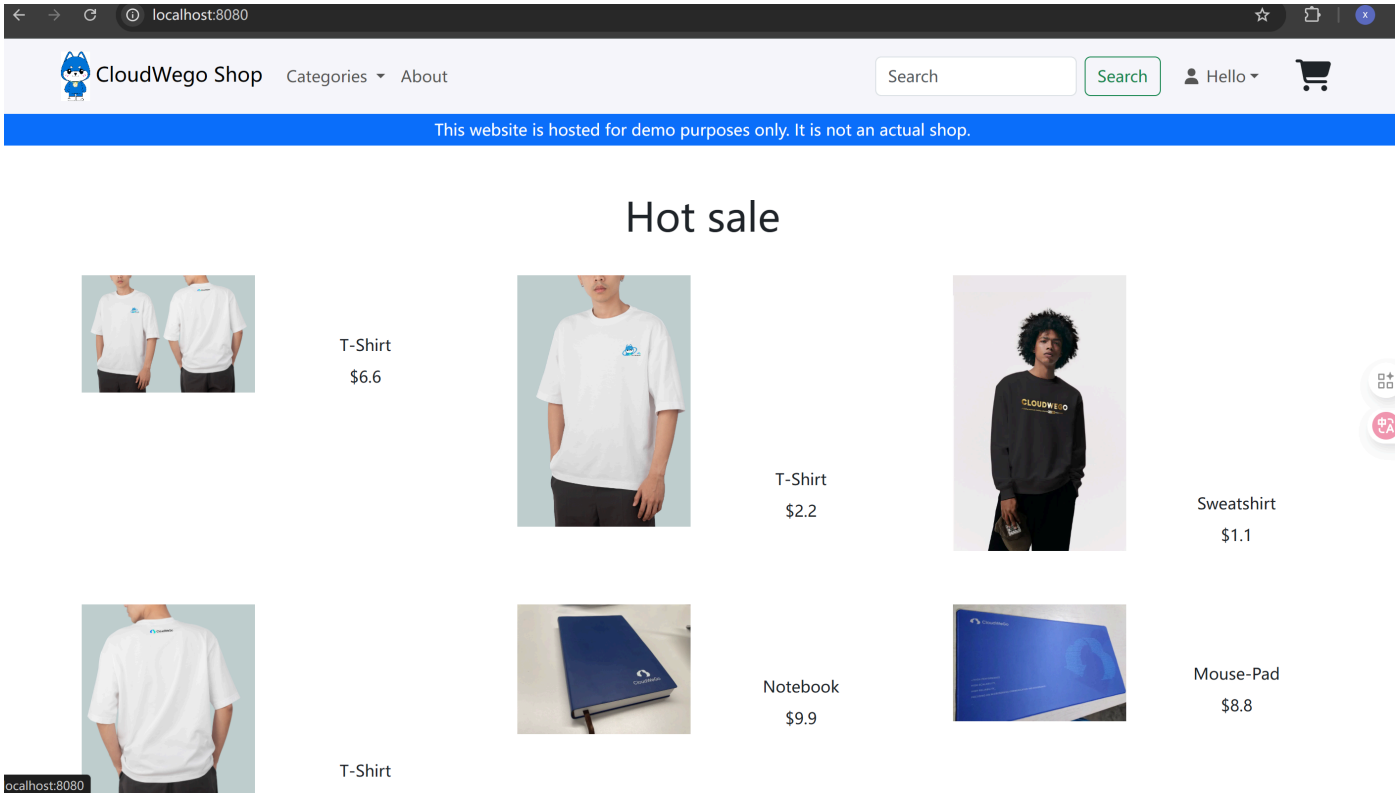
- 添加 mysql-init-job.yaml，等待 MySQL 就绪并执行创建数据库与授权的 SQL。
- 添加 mysql-secret.yaml 保存 MySQL 密码

现在可以了

```
(base) PS E:\Desktop\Junior1\CloudNative\lab\Lab2\cloud-lab2> minikube kubectl -- get pods -A
NAMESPACE     NAME                                 READY   STATUS      RESTARTS   AGE
default       cart-675775f99b-9k4l6                1/1     Running     0          7m15s
default       checkout-6cf865d4bd-9mcjf            1/1     Running     0          7m14s
default       email-c65b8b4b-g9qfd                 1/1     Running     0          7m14s
default       frontend-75d7bd8dbd-ql8vr            1/1     Running     0          7m14s
default       mysql-0                              1/1     Running     0          7m59s
default       mysql-init-job-559nq                 0/1     Completed   0          7m59s
default       nats-dbfbf4fdb-cznhq                 1/1     Running     0          7m59s
default       order-849ffc844d-zjmrd               1/1     Running     0          7m14s
default       payment-57c7494997-4q7kg             1/1     Running     0          7m13s
default       product-6f74cd4986-1b7t5             1/1     Running     0          7m12s
default       redis-7699f47487-6qc2x               1/1     Running     0          7m59s
default       user-7d8ddf855f-xpc8q                1/1     Running     0          7m12s
kube-system   coredns-66bc5c9577-n4q96             1/1     Running     0          16m
kube-system   etcd-minikube                        1/1     Running     0          17m
kube-system   kube-apiserver-minikube              1/1     Running     0          17m
kube-system   kube-controller-manager-minikube     1/1     Running     0          17m
kube-system   kube-proxy-rhdrw                     1/1     Running     0          16m
kube-system   kube-scheduler-minikube              1/1     Running     0          17m
kube-system   storage-provisioner                  1/1     Running     0          16m
```

运行 `kubectl port-forward service/frontend 8080:8080`

发现可以正常访问在 Kubernetes 集群中运行的 gomall 系统。




```
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-secret.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-pv-pvc.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-configmap.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-headless-service.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-statefulset.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\redis-deployment.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\redis-service.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\nats-deployment.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\nats-service.yaml;
minikube kubectl -- apply -f .\gomall\k8s\middlewares\mysql-init-job.yaml;
```

任务四、五测试结果总结

任务四: 在Kubernetes集群中部署gomall

文件列表

```
microservices/
├── 00-namespace.yaml            # Gomall 命名空间
├── cart-configmap.yaml          # Cart 配置
├── cart-deployment.yaml         # Cart 部署
├── cart-service.yaml            # Cart 服务
├── checkout-configmap.yaml      # Checkout 配置
├── checkout-deployment.yaml     # Checkout 部署
├── checkout-service.yaml        # Checkout 服务
├── email-configmap.yaml         # Email 配置
├── email-deployment.yaml        # Email 部署
├── email-service.yaml           # Email 服务
├── frontend-configmap.yaml      # Frontend 配置
├── frontend-deployment.yaml     # Frontend 部署
├── frontend-service.yaml        # Frontend 服务
├── order-configmap.yaml         # Order 配置
├── order-deployment.yaml        # Order 部署
├── order-service.yaml           # Order 服务
├── payment-configmap.yaml       # Payment 配置
├── payment-deployment.yaml      # Payment 部署
├── payment-service.yaml         # Payment 服务
├── product-configmap.yaml       # Product 配置
├── product-deployment.yaml      # Product 部署
├── product-service.yaml         # Product 服务
├── user-configmap.yaml          # User 配置
├── user-deployment.yaml         # User 部署
└── user-service.yaml            # User 服务
```

部署的微服务

服务名称	端口	状态	功能
cart	8883	Running	购物车管理
checkout	8884	Running	结账功能
email	8888	Running	邮件发送

服务名称	端口	状态	功能
order	8885	Running	订单管理
payment	8886	Running	支付功能
product	8881	Running	商品管理
user	8882	Running	用户管理
frontend	8080	Running	前端页面

部署命令记录

1. 创建命名空间和部署所有服务

```
kubectl apply -f gomall\k8s\microservices\
```

2. 查看部署状态

```
# 查看所有 Pod
kubectl get pods -n gomall

# 查看所有 Service
kubectl get svc -n gomall
```

3. 访问前端服务

```
kubectl port-forward service/frontend 8080:8080 -n gomall
```

然后在浏览器访问: http://localhost:8080 经过测试一切服务皆正常运作

任务五：扩缩容与负载均衡实验

测试步骤执行记录

1. 初始状态确认

- 初始副本数：1 个 Pod (`product-5454f95b79-7bxj2`)
- 状态：Running

![alt text](images/image copy 5.png)

2. 扩容操作

```
minikube kubectl -- scale deployment/product --replicas=3
```

```
PS C:\Users\slxie\Desktop\cloud-native\cloud-lab2> minikube kubectl -- scale deployment/product --replicas=3
deployment.apps/product scaled
```

- 扩容后副本数：3 个 Pod
  - `product-5454f95b79-7bxj2` (原有)
  - `product-5454f95b79-rbdrk` (新增)
  - `product-5454f95b79-wrq6b` (新增)

![alt text](images/image copy 3.png)

3. Hey性能测试结果

扩容前

![alt text](images/image copy 6.png)

扩容后

![alt text](images/image copy 7.png)

4. 负载均衡验证结果

通过查看三个 Pod 的日志，确认所有 Pod 都在处理请求：

Pod 1 (7bxj2): 日志显示大量 `ListProductsService:` 请求

Pod 2 (rbdrk):日志显示大量 `ListProductsService:` 请求

Pod 3 (wrq6b):日志显示大量 `ListProductsService:` 请求

Service 端点验证:

```
Endpoints: 10.244.0.15:8881,10.244.0.17:8881,10.244.0.16:8881
```

三个端点都已正确注册到 Service 中，负载均衡配置生效。

![alt text](images/image copy.png) ![alt text](images/image copy 2.png)

6.使用Helm Chart打包部署

使用scoop安装helm

按照任务二的实验过程部署中间件（要求中没有涉及中间件的打包）

将microservices中的每一个yaml文件的内容提取出一个模板写成对应的template

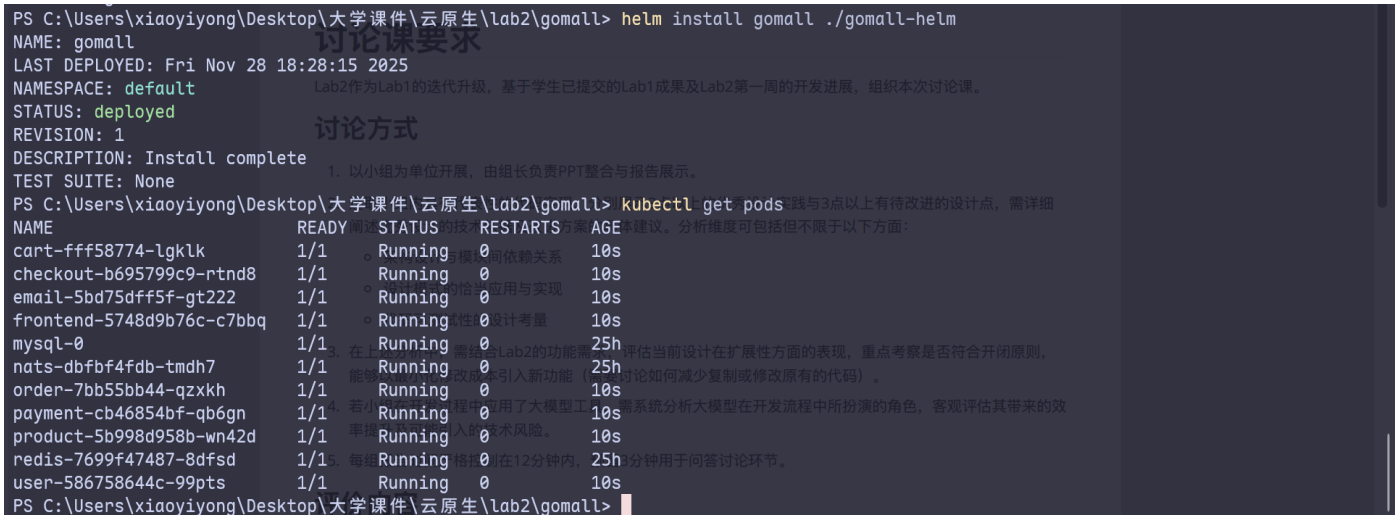
e.g. checkout.yaml将其中需要配置的参数抽取出来，值写在value.yaml中，在template中引用values.yaml的值。

每个资源都需要提取出一个模板（也可以只提取出一个通用的，但可能反而会变得更复杂）

模板提取完成后，将所有的值都分门别类的写入values.yaml.

然后在项目根目录下运行命令 `helm install gomall ./helm`

运行结果：



端口转发之后访问前端，服务都正常运行

```
PS C:\Users\xiaoyiyong\Desktop\大学课件\云原生\lab2\gomall> kubectl port-forward service/frontend 8080:8080
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
Handling connection for 8080
Handling connection for 8080
```