Demo Project: Create Helm Chart for Microservices

This project demonstrates how to create a shared Helm chart for multiple microservices to reuse common `Deployment` and `Service` configurations. Additionally, it explains how to create a separate Helm chart for Redis as a third-party service.

Step 1: Create the Helm Chart for all microservices

1. **Create a Helm Chart Directory**: `helm create microservice`
    - This will create a default folder structure for the Helm chart under `microservice`.
2. **Clean Up Default Templates**:
    - Navigate to the `templates/` directory and delete all default files:
        - `cd microservice/templates`
        - `rm -rf *`
    - Clear the contents of `values.yaml` in the `microservice` root directory.
3. **Create the Basic Templates**:

- `deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ .Values.appName }}
spec:
  replicas: {{ .Values.appReplicas }}
  selector:
    matchLabels:
      app: {{ .Values.appName }}
  template:
    metadata:
      labels:
        app: {{ .Values.appName }}
    spec:
      containers:
      - name: {{ .Values.appName }}
        image: "{{ .Values.appImage }}:{{ .Values.appVersion }}"
        ports:
        - containerPort: {{ .Values.containerPort }}
```

- `service.yaml`:

```
apiVersion: v1
kind: Service
metadata:
  name: {{ .Values.appName }}
spec:
  type: {{ .Values.serviceType }}
  selector:
    app: {{ .Values.appName }}
  ports:
  - protocol: TCP
    port: {{ .Values.servicePort }}
    targetPort: {{ .Values.containerPort }}
```

4. **Define Default Values in `values.yaml`:**
   - Example:

```yaml
appName: servicename
appImage: gcr.io/google-samples/microservices-demo/servicename
appVersion: v0.0.0
appReplicas: 1
containerPort: 8080
containerEnvVars:
- name: ENV_VAR_ONE
  value: "valueone"
- name: ENV_VAR_TWO
  value: "valuetwo"

servicePort: 8080
serviceType: ClusterIP
```

5. **Create Custom Values Files**:
   - Example: `email-service-values.yaml`

```yaml
appName: emailservice
appImage: gcr.io/google-samples/microservices-demo/emailservice
appVersion: v0.8.0
appReplicas: 2
containerPort: 8080
containerEnvVars:
- name: PORT
  value: "8080"

servicePort: 5000
```

Step 2: Validate Helm Chart Configuration

1. **Render Chart Templates Locally**:
   - Use `helm template` to check the rendered templates:

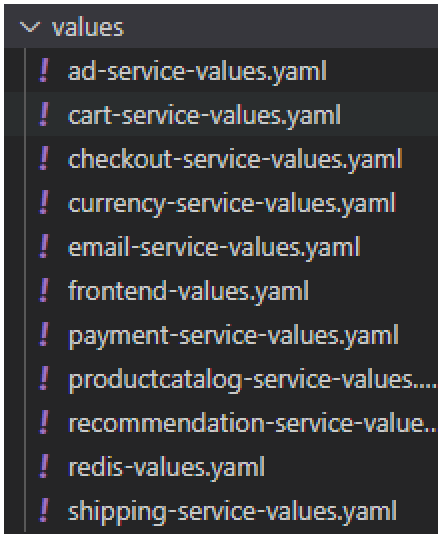
     `helm template -f email-service-values.yaml microservice`
2. **Lint the Helm Chart**:
   - Use `helm lint` to check for errors or warnings:

     `helm lint -f email-service-values.yaml microservice`

Step 3: Deploy Microservices Using Helm

1. **Deploy a Service**:
   - Deploy the `email-service` microservice:

     `helm install -f email-service-values.yaml emailservice microservice`
2. **Verify the Deployment**:
   - List Helm releases: `helm ls`
   - Confirm running pods: `kubectl get pod`
3. **Create Values Files for All Microservices**:
   - Repeat the process for each microservice (e.g., `cart-service-values.yaml`, `checkout-service-values.yaml`, etc.).
4. **Organize Files**:
   - Create a `values` directory:
     - `mkdir values`
     - `mv *.yaml values/`
   - The structure should look like this:



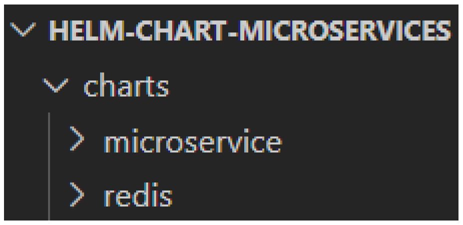
```
values
  ad-service-values.yaml
  cart-service-values.yaml
  checkout-service-values.yaml
  currency-service-values.yaml
  email-service-values.yaml
  frontend-values.yaml
  payment-service-values.yaml
  productcatalog-service-values....
  recommendation-service-value...
  redis-values.yaml
  shipping-service-values.yaml
```

Step 4: Create Redis Helm Chart

Since Redis is a third-party service with unique stateful requirements, it needs a dedicated Helm chart separate from the microservices.

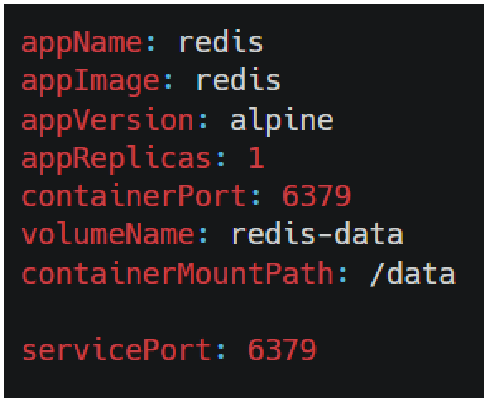
1. **Create and Navigate to the `charts/` directory:**
   - `mkdir charts`
   - `cd charts`

2. **Create a New Helm Chart**: `helm create redis`
3. **Clean Up**:
   - Delete default files and clean up `values.yaml`
4. **Organize Structure**:
   - Move the `microservice` directory into `charts`.
   - Folder structure show look like this:



```
HELM-CHART-MICROSERVICES
  charts
    microservice
    redis
```

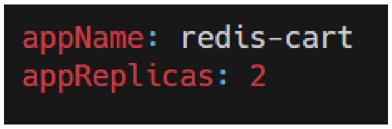
5. **Create Redis-Specific Templates**:
   - Define `deployment.yaml` and `service.yaml` for Redis.
   - Add default values to `values.yaml` for Redis. Example:



```yaml
appName: redis
appImage: redis
appVersion: alpine
appReplicas: 1
containerPort: 6379
volumeName: redis-data
containerMountPath: /data

servicePort: 6379
```

6. **Override Values**:
   - Create a custom `redis-values.yaml` file:



```yaml
appName: redis-cart
appReplicas: 2
```

Step 5: Validate and Deploy Redis

1. **Validate Redis Chart**:
   - Render templates locally: `helm template -f values/redis-values.yaml charts/redis`
   - Dry-run the installation (This checks generated manifest without installing the chart):

     `helm install --dry-run -f values/redis-values.yaml rediscart charts/redis`

Step 6: There are two ways to Deploy:

Option A: Automate Deployment with a Script

1. **Create** `install.sh`

```
helm install -f values/redis-values.yaml rediscart charts/redis

helm install -f values/email-service-values.yaml emailservice charts/microservice
helm install -f values/cart-service-values.yaml cartservice charts/microservice
helm install -f values/currency-service-values.yaml currencyservice charts/microservice
helm install -f values/payment-service-values.yaml paymentservice charts/microservice
helm install -f values/recommendation-service-values.yaml recommendationservice charts/microservice
helm install -f values/productcatalog-service-values.yaml productcatalogservice charts/microservice
helm install -f values/shipping-service-values.yaml shippingservice charts/microservice
helm install -f values/ad-service-values.yaml adservice charts/microservice
helm install -f values/checkout-service-values.yaml checkoutservice charts/microservice
helm install -f values/frontend-values.yaml frontendservice charts/microservice
```

2. **Make Script Executable:** `chmod u+x` `install.sh`
3. **Execute the script:** `./install.sh`
4. **Verify pods:** `kubectl get pod`

```
kubectl get pod

NAME                                       READY   STATUS    RESTARTS   AGE
adservice-54b74b55b7-dgd6l                 1/1     Running   0          29s
adservice-54b74b55b7-xbgjk                 1/1     Running   0          29s
cartservice-dd4f7764f-bk6qg                1/1     Running   0          43s
cartservice-dd4f7764f-vgwnf                1/1     Running   0          43s
checkoutservice-767cc65db4-bjczw           1/1     Running   0          26s
checkoutservice-767cc65db4-qs4wg           1/1     Running   0          26s
currencyservice-7dcb6cb45f-dtm5v           1/1     Running   0          41s
currencyservice-7dcb6cb45f-qb8c7           1/1     Running   0          41s
emailservice-6d488b67f8-bpm45              1/1     Running   0          52m
emailservice-6d488b67f8-d2zsw              1/1     Running   0          52m
frontend-696858b68d-4b99d                  1/1     Running   0          23s
frontend-696858b68d-xgcxw                  1/1     Running   0          23s
paymentservice-795d5d4bd8-kshxb            1/1     Running   0          39s
paymentservice-795d5d4bd8-n8l4n            1/1     Running   0          39s
productcatalogservice-7cf54479b6-ncrpr     1/1     Running   0          34s
productcatalogservice-7cf54479b6-v6m7h     1/1     Running   0          34s
recommendationservice-76887dcc55-7rjxs     1/1     Running   0          36s
recommendationservice-76887dcc55-vspc9     1/1     Running   0          36s
redis-cart-7ff4c98f7-9n528                 1/1     Running   0          47s
redis-cart-7ff4c98f7-svhhb                 1/1     Running   0          47s
shippingservice-d84b79bb-ff22b             1/1     Running   0          32s
shippingservice-d84b79bb-jwt64             1/1     Running   0          32s
```

5. **Clean-Up:**
   - **Create** `uninstall.sh`

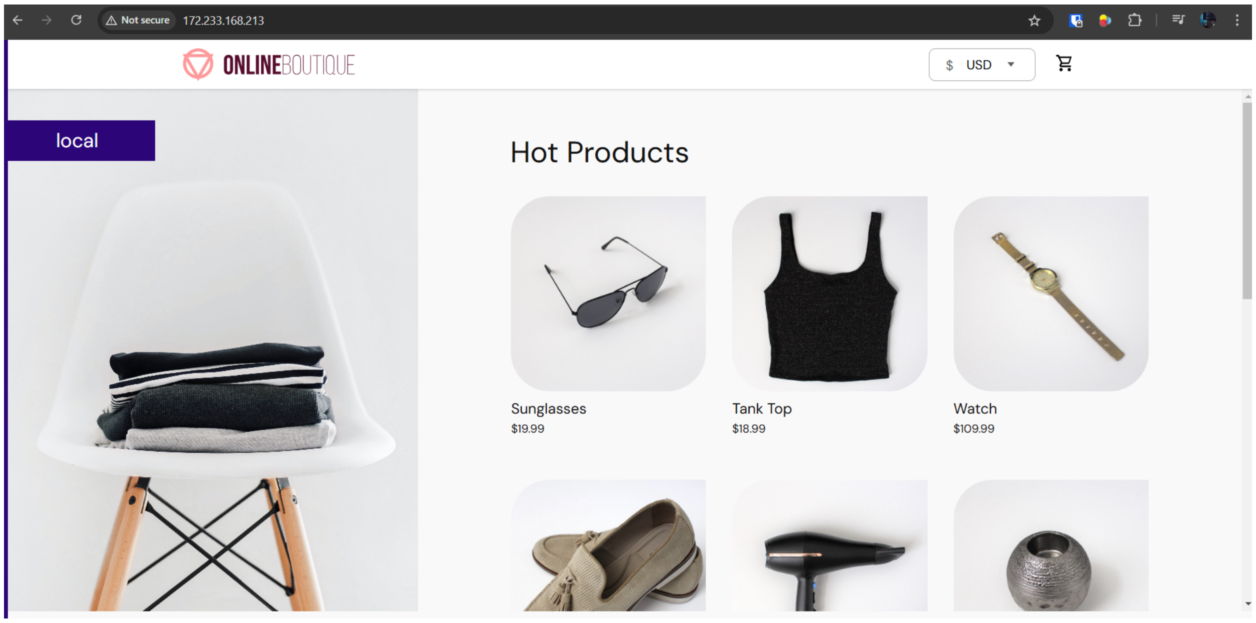
- **Make Script Executable:** `chmod u+x un` `install.sh`
- **Execute the script:** `./uninstall.sh`
- **Confirm pods are terminated:** `kubectl get pod`

Option B: Manage Helm Releases with Helmfile

1. **Install helmfile:** `brew install helmfile`
    - In case you don't have "brew" install with these steps:
        1. **Install Homebrew**:
           Run the following command to install Homebrew:
           ```
           /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh )"
           ```
        2. **Add Homebrew to your PATH**:
           Follow the on-screen instructions after installation, or add this to your `.zshrc` file:
           ```
           echo 'eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"' >> ~/.zshrc
           source ~/.zshrc
           ```
        - **Verify Installation**: `helmfile --version`
2. **Create** `helmfile.yaml`:
    - Define all services and Redis in `helmfile.yaml`.
3. **Deploy Using Helmfile**: `helmfile sync`
4. **Verify running pods:** `kubectl get pod`

```
kubectl get pod
NAME                                     READY   STATUS    RESTARTS   AGE
adservice-54b74b55b7-ft6zg               1/1     Running   0          4m52s
adservice-54b74b55b7-xz8st               1/1     Running   0          4m53s
cartservice-dd4f7764f-wqpd7              1/1     Running   0          4m52s
cartservice-dd4f7764f-zj2l8              1/1     Running   0          4m51s
checkoutservice-767cc65db4-8w8cf         1/1     Running   0          4m54s
checkoutservice-767cc65db4-9bj4k         1/1     Running   0          4m54s
currencyservice-7dcb6cb45f-blmtd         1/1     Running   0          4m51s
currencyservice-7dcb6cb45f-n46qf         1/1     Running   0          4m51s
emailservice-6d488b67f8-5lt7z            1/1     Running   0          4m52s
emailservice-6d488b67f8-rhxrr            1/1     Running   0          4m53s
frontend-696858b68d-47q2w                1/1     Running   0          4m54s
frontend-696858b68d-nftjt                1/1     Running   0          4m54s
paymentservice-795d5d4bd8-7ggbx          1/1     Running   0          4m54s
paymentservice-795d5d4bd8-7m7zf          1/1     Running   0          4m53s
productcatalogservice-7cf54479b6-nlnc4   1/1     Running   0          4m52s
productcatalogservice-7cf54479b6-wxkr4   1/1     Running   0          4m53s
recommendationservice-76887dcc55-j8vvm   1/1     Running   0          4m54s
recommendationservice-76887dcc55-xx7ck   1/1     Running   0          4m54s
redis-cart-6dfd6dfd75-68pk7              1/1     Running   0          4m54s
shippingservice-d84b79bb-5ck2p           1/1     Running   0          4m54s
shippingservice-d84b79bb-kgdsv           1/1     Running   0          4m54s
```

5. **Validate the Application**:
   - Use the Linode NodeBalancer IP to access the application in the browser:



6. **Clean up:** `helmfile destroy`