```
online-retail-microservices
│
├── charts
│   ├── product-service
│   │   ├── templates
│   │   │   ├── deployment.yaml
│   │   │   ├── service.yaml
│   │   │   └── ingress.yaml
│   │   ├── values.yaml
│   │   └── Chart.yaml
│   │
│   ├── order-service
│   │   ├── templates
│   │   │   ├── deployment.yaml
│   │   │   ├── service.yaml
│   │   │   └── ingress.yaml
│   │   ├── values.yaml
│   │   └── Chart.yaml
│   │
│   ├── user-service
│   │   ├── templates
│   │   │   ├── deployment.yaml
│   │   │   ├── service.yaml
│   │   │   └── ingress.yaml
│   │   ├── values.yaml
│   │   └── Chart.yaml
│   │
│   └── common
│       ├── templates
│       │   ├── database.yaml
│       │   ├── configmap.yaml
│       │   └── ingress.yaml
│       ├── values.yaml
│       └── Chart.yaml
│
├── templates
│   ├── _helpers.tpl
│   ├── ingress.yaml
│   └── service.yaml
│
```

```
├── values.yaml
└── Chart.yaml
```

write all the yaml files

Common Service (common/templates/database.yaml):

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: common-database
spec:
  replicas: 1
  selector:
    matchLabels:
      app: common-database
  template:
    metadata:
      labels:
        app: common-database
    spec:
      containers:
      - name: common-database
        image: your-database-image
        ports:
        - containerPort: 5432 # Adjust based on your database
---
apiVersion: v1
kind: Service
metadata:
  name: common-database-service
spec:
  selector:
    app: common-database
  ports:
    - protocol: TCP
      port: 5432
```

Common Service (common/templates/configmap.yaml):

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: common-configmap
data:
  APP_ENV: production
  LOG_LEVEL: info
  # Add other configuration properties here
```

Common Service (common/templates/ingress.yaml):

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: common-ingress
spec:
  rules:
  - host: common.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: common-service
            port:
              number: 80
```

Product Service (charts/product-service/templates/deployment.yaml):

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: product-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: product-service
  template:
    metadata:
      labels:
        app: product-service
    spec:
      containers:
      - name: product-service
        image: your-product-service-image
        ports:
        - containerPort: 8080 # Adjust based on your service port
---
apiVersion: v1
kind: Service
metadata:
  name: product-service
spec:
```

```yaml
  selector:
    app: product-service
  ports:
    - protocol: TCP
      port: 8080
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: product-service-ingress
spec:
  rules:
  - host: product.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: product-service
            port:
              number: 8080
```

Order Service (charts/order-service/templates/deployment.yaml):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: order-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: order-service
  template:
    metadata:
      labels:
        app: order-service
    spec:
      containers:
      - name: order-service
        image: your-order-service-image
        ports:
        - containerPort: 8080 # Adjust based on your service port
---
apiVersion: v1
kind: Service
metadata:
  name: order-service
spec:
  selector:
    app: order-service
  ports:
    - protocol: TCP
      port: 8080
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: order-service-ingress
spec:
  rules:
  - host: order.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: order-service
            port:
              number: 8080
```

User Service (charts/user-service/templates/deployment.yaml):

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: user-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: user-service
  template:
    metadata:
      labels:
        app: user-service
    spec:
      containers:
      - name: user-service
        image: your-user-service-image
        ports:
        - containerPort: 8080 # Adjust based on your service port
---
apiVersion: v1
kind: Service
metadata:
  name: user-service
spec:
  selector:
    app: user-service
  ports:
    - protocol: TCP
```

```
      port: 8080
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: user-service-ingress
spec:
  rules:
  - host: user.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: user-service
            port:
              number: 8080
```

API Gateway (templates/service.yaml):

```
apiVersion: v1
kind: Service
metadata:
  name: api-gateway
spec:
  selector:
    app: api-gateway
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

API Gateway (templates/ingress.yaml):

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: api-gateway-ingress
spec:
  rules:
  - host: api.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: api-gateway
            port:
              number: 80
```

Top-Level Ingress (templates/ingress.yaml):

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: top-level-ingress
spec:
  rules:
  - host: example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: api-gateway
            port:
              number: 80
```

Top-Level Values (values.yaml):

```
common:
  database:
    image: your-database-image
  configmap:
    appEnv: production
    logLevel: info
    # Add other configuration properties here

product-service:
  image: your-product-service-image

order-service:
  image: your-order-service-image

user-service:
  image: your-user-service-image

api-gateway:
  image: your-api-gateway-image
```

Top-Level Chart (Chart.yaml):

```
apiVersion: v2
name: online-retail-microservices
description: A Helm chart for an online retail microservices architecture
version: 1.0.0
```

Certainly! The **path** and **pathType** in the Ingress definition are used to specify how incoming requests should be matched and directed to the corresponding services. Let's break down the values for the provided example:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: user-service-ingress
spec:
  rules:
  - host: user.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: user-service
            port:
              number: 8080
```

In this example:

- **path**: **/**
- This means that any request with a URL path that starts with **/** (e.g., **http://user.example.com/**) will match this Ingress rule.
- **pathType**: **Prefix**
- The **Prefix** path type means that the specified **path** is a prefix that should be matched. In other words, any URL path that starts with the specified prefix (**/** in this case) will be directed to the specified backend service.

So, with the provided values:

- If a request comes in for **http://user.example.com/**, it will match because the path starts with **/**. The request will be directed to the **user-service** on port **8080**.
- If a request comes in for **http://user.example.com/somepath**, it will also match because it starts with **/**, and it will be directed to the **user-service** on port **8080**.

Adjust the **path** and **pathType** values based on your specific routing requirements. If you want an exact path match, you can set **path** to the specific path you want to match, and you can choose **Exact** for **pathType**. If you want to capture all paths, you can use **/** as the path with **Prefix**.