KUBERNETES INGRESS

- Ingress helps to expose the HTTP and HTTPS routes from outside of the cluster.
- Ingress supports
  - Path-based
  - Host-based routing
- Ingress supports Load balancing and SSL termination.
- It redirects the incoming requests to the right services based on the Web URL or path in the address.
- Ingress provides the encryption feature and helps to balance the load of the applications.

Why do we use Ingress because the load balancer supports the same thing?

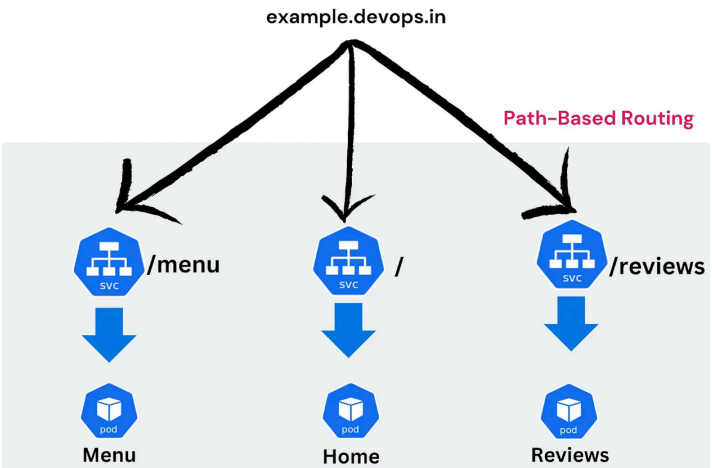
Ingress is used to manage the external traffic to the services within the cluster which provides features like host-based routing, path-based routing, SSL termination, and more. Where a Load balancer is used to manage the traffic but the load balancer does not provide the fine-grained access control like Ingress.

Example:

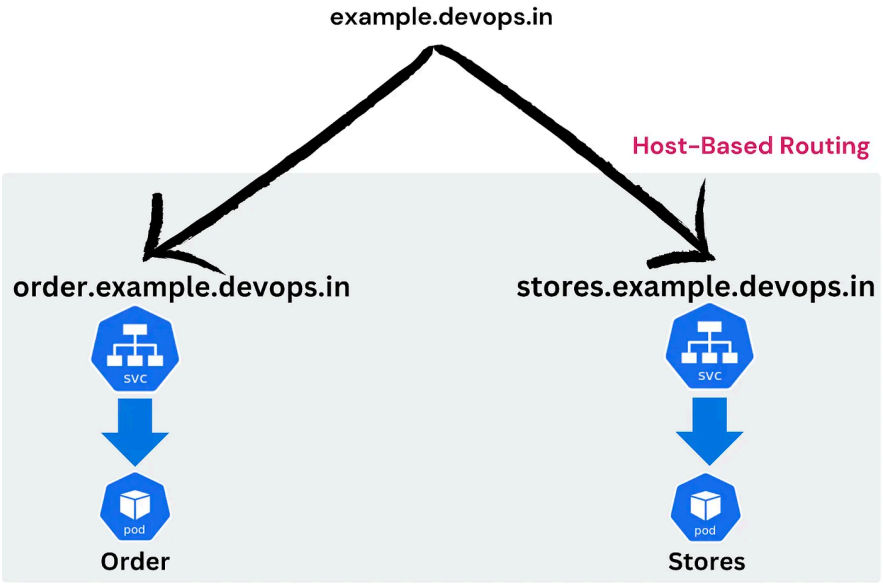
- Suppose you have multiple Kubernetes services running on your cluster and each service serves a different application such as example.com/app1 and example.com/app2. With the help of Ingress, you can achieve this. However, the Load Balancer routes the traffic based on the ports and can't handle the URL-based routing.

There are two types of Routing in Ingress:

- **Path-based routing:** Path-based routing directs traffic to the different services based on the path such as example.com/app1.



Host-based routing: Host-based routing directs traffic to the different services based on the Website's URL such as demo.example.com.



To install ingress, firstly we have to install nginx ingress controller:

command: kubectl create -f https://raw.githubuserconte nt.com/kubernetes/ingress-nginx/controller-v1.2.1/deploy/static/provider/cloud/deploy.yaml

Once we install ingress controller, we have to deploy 2 applications.

github url: **https://github.com/mustafaprojectsindevops/kubernetes/tree/master/ingress**

After executing all the files, use **kubectl get ing** to get ingress. After 30 seconds it will provide one load balancer dns.

access those applications using dns/nginx and dns/httpd. So the traffic will route into both the applications as per the routing

ONE.YML:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: one
spec:
  replicas: 2
  selector:
    matchLabels:
      app: swiggy
  template:
    metadata:
      labels:
        app: swiggy
    spec:
      containers:
      - name: cont-1
        image: nginx
        ports:
        - containerPort: 80
        env:
        - name: TITLE
          value: "NGINX APP1"
---
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  type: NodePort
  ports:
  - port: 80
  selector:
    app: swiggy
```

TWO.YML

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: two
spec:
  replicas: 2
  selector:
    matchLabels:
      app: zomato
  template:
    metadata:
      labels:
        app: zomato
    spec:
      containers:
      - name: cont-2
        image: httpd
        ports:
        - containerPort: 80
        env:
        - name: TITLE
          value: "APACHE APP2"
---
apiVersion: v1
kind: Service
metadata:
  name: httpd
spec:
  type: NodePort
  ports:
  - port: 80
  selector:
    app: zomato
```

INGRESS.YML:

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: k8s-ingress
  annotations:
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
    nginx.ingress.kubernetes.io/use-regex: "true"
    nginx.ingress.kubernetes.io/rewrite-target: /$2
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
      - path: /nginx(/|$)(.*)
        pathType: Prefix
        backend:
          service:
            name: nginx
            port:
              number: 80
      - path: /httpd(/|$)(.*)
        pathType: Prefix
        backend:
          service:
            name: httpd
            port:
              number: 80
      - path: /(.*)
        pathType: Prefix
        backend:
          service:
            name: nginx
            port:
              number: 80
```