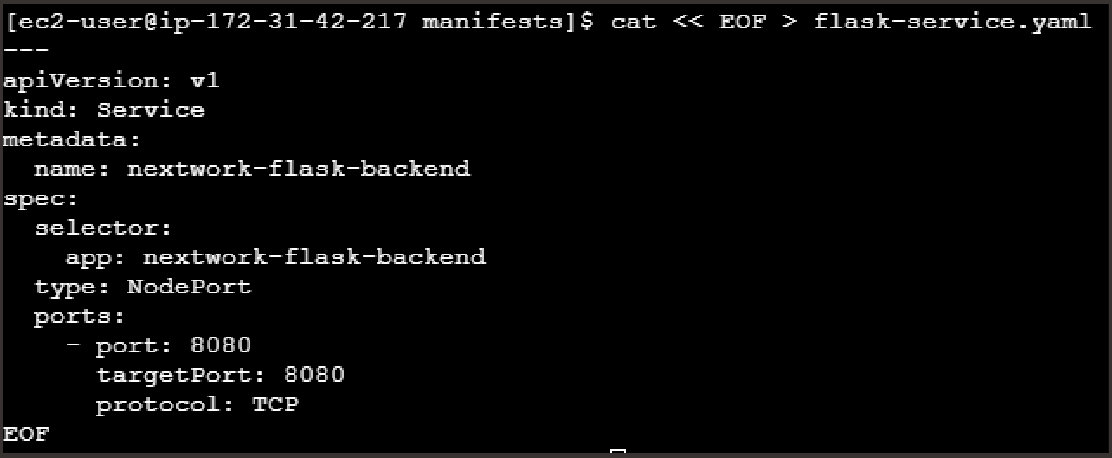
nextwork.org

Create Kubernetes Manifests

Nicolás Aversa



```
[ec2-user@ip-172-31-42-217 manifests]$ cat << EOF > flask-service.yaml
---
apiVersion: v1
kind: Service
metadata:
  name: nextwork-flask-backend
spec:
  selector:
    app: nextwork-flask-backend
  type: NodePort
  ports:
    - port: 8080
      targetPort: 8080
      protocol: TCP
EOF
```

Introducing Today's Project!

In this project, I will set up a Deployment manifest that tells Kubernetes how to deploy my backend and set up a Service manifest that tells Kubernetes how to expose my backend to my users.

Tools and concepts

I used Amazon EKS, eksctl to create the cluster that acted as the foundation of this project. Key concepts include using manifests, eksctl commands, git commands, docker commands, etc.

Project reflection

I did this project to learn more about Kubernetes and finally gets hands-on experience with it. This project met my goals; it was nice getting to learn how Deployment and Service manifests work.

This project took me approximately 3 hours. The most challenging part was understanding what each line defined in the Deployment manifest. My favourite part was the one that challenged me the most! Diving deep into what each part of the Deployment manifest does gave me the opportunity to really grasp how Kubernetes orchestrates applications under the surface.

Project Set Up

Kubernetes cluster

To set up today's project, I launched a Kubernetes cluster. Steps I took to do this included: 1. Launching an EC2 instance. 2. Connecting to my instance using EC2 Instance Connect. 3. Run a command in my EC2 instance's terminal to download, extract, and install eksctl from a GitHub repo. 4. Attaching an IAM role 'nextwork-eks-instance-role' to my EC2 instance, which has AdministratorAccess. 5. Run a command to create my EKS cluster; specifying name, nodegroup name, node type, number of nodes (desired, min and max).

Backend code

I retrieved the backend that I plan to deploy by configuring Git with my GitHub credentials and cloning the backend application repository onto my EC2 instance. Backend is the "brain" of an application. It's where the app processes user requests and stores and retrieves data.

Container image

Once I cloned the backend code, I built a container image because this container image lets Kubernetes set up multiple, identical containers so my application runs consistently across different environments. I used Docker to do so.

I also pushed the container image to a container registry, because it lets EKS deploy my container image with minimal authentication setup, since ECR is also an AWS service. To push the image to ECR, I ran the push commands ECR provides.

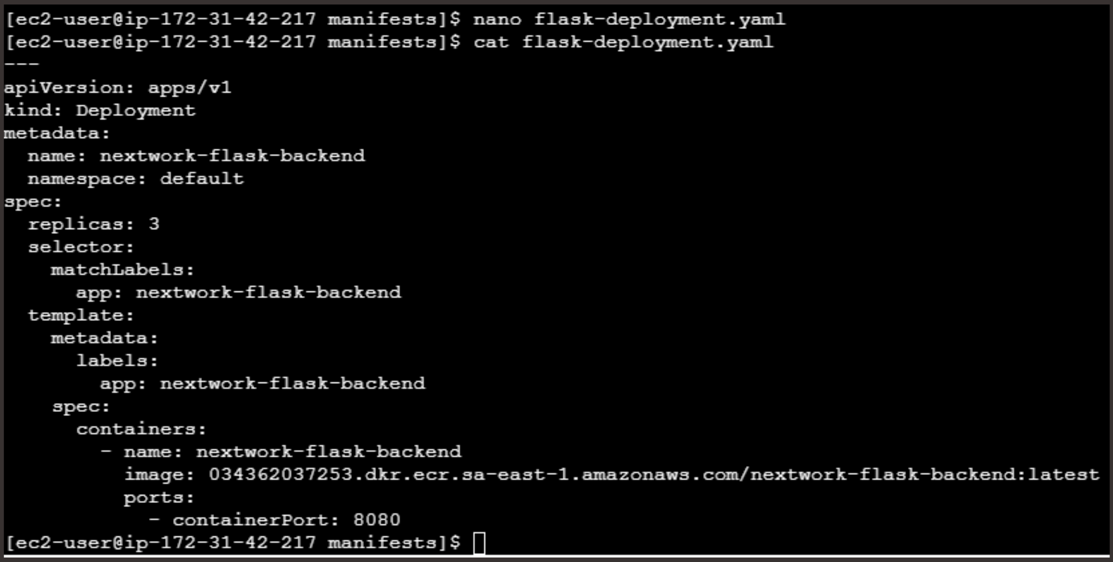
Manifest files

Kubernetes manifests are a set of instructions that tells Kubernetes how to run your app. Manifests are helpful because without manifests, Kubernetes wouldn't know how to set up and manage your app automatically. This means you'd have to manually configure each container every time you deploy, which would be confusing, error-prone, and hard to repeat. Manifests make deployment simple, clear, and consistent.

A Kubernetes deployment manifest manages the desired state of your application within a Kubernetes cluster, defining how your containerized workload should be deployed, updated, and maintained. The container image URL in my Deployment manifest tells Kubernetes where to pull the container image from when it deploys my application.

Nicolás Aversa
NextWork Student

NextWork.org

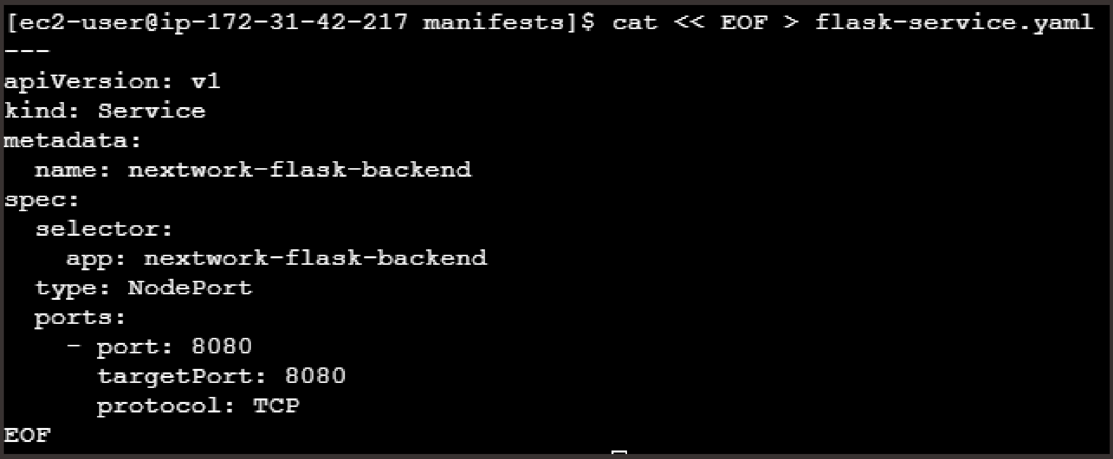


```
[ec2-user@ip-172-31-42-217 manifests]$ nano flask-deployment.yaml
[ec2-user@ip-172-31-42-217 manifests]$ cat flask-deployment.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nextwork-flask-backend
  namespace: default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nextwork-flask-backend
  template:
    metadata:
      labels:
        app: nextwork-flask-backend
    spec:
      containers:
        - name: nextwork-flask-backend
          image: 034362037253.dkr.ecr.sa-east-1.amazonaws.com/nextwork-flask-backend:latest
          ports:
            - containerPort: 8080
[ec2-user@ip-172-31-42-217 manifests]$ 
```

Service Manifest

A Kubernetes Service exposes your application, making it accessible to network traffic (from within the cluster or from external sources). You need a Service manifest to tell Kubernetes to create/update a Service using details like which pods to route traffic to, the type of traffic it should handle, and which ports should be used.

My Service manifest sets up a Service resource in Kubernetes to make nextwork-flask-backend accessible from outside the cluster. Simply put, this Service lets me access my backend using my node's IP and a port number.



```
[ec2-user@ip-172-31-42-217 manifests]$ cat << EOF > flask-service.yaml
---
apiVersion: v1
kind: Service
metadata:
  name: nextwork-flask-backend
spec:
  selector:
    app: nextwork-flask-backend
  type: NodePort
  ports:
    - port: 8080
      targetPort: 8080
      protocol: TCP
EOF
```

Deployment Manifest

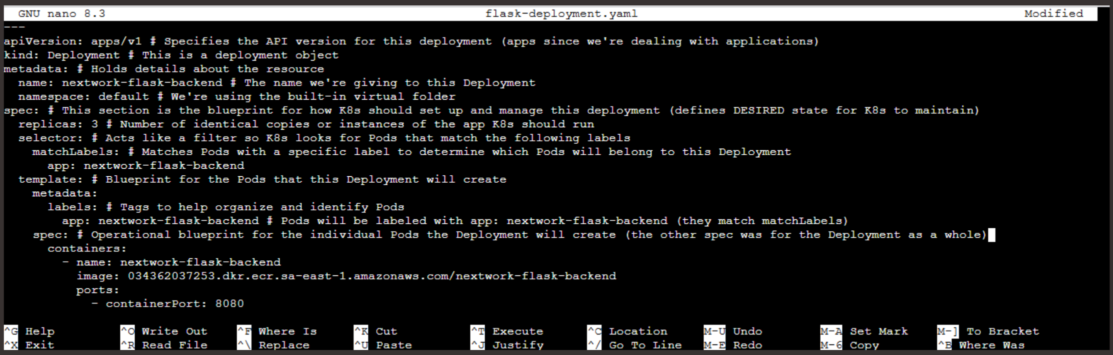
Annotating the Deployment manifest helped me understand the structure and purpose of each component in a Kubernetes deployment because it provided clear, layered explanations that connected technical terms to their real-world functions. By breaking down the file from the API level down to individual containers, I could see how Kubernetes orchestrates applications through this hierarchy.

A notable line in the Deployment manifest is the number of replicas, which means identical copies (or instances) of the app Kubernetes should run. Pods are relevant to this because depending the number of replicas set, it means the same amount of identical Pods will be created. Pods are the smallest deployable units in Kubernetes - you can't deploy containers on their own; they must be inside a Pod.

One part of the Deployment manifest I still want to know more about is how the selector and template work together because I'm not completely clear on how they ensure the Deployment manages the right Pods. Specifically, I want to understand why the labels in selector.matchLabels must exactly match the labels in template.metadata.labels, and what happens if they don't. Does this mean the Deployment will only control Pods with those exact labels, and will it automatically fix things if the labels get changed?

Nicolás Aversa
NextWork Student

NextWork.org



```
  GNU nano 8.3                          flask-deployment.yaml                          Modified
---
apiVersion: apps/v1 # Specifies the API version for this deployment (apps since we're dealing with applications)
kind: Deployment # This is a deployment object
metadata: # Holds details about the resource
  name: nextwork-flask-backend # The name we're giving to this Deployment
  namespace: default # We're using the built-in virtual folder
spec: # This section is the blueprint for how K8s should set up and manage this deployment (defines DESIRED state for K8s to maintain)
  replicas: 3 # Number of identical copies or instances of the app K8s should run
  selector: # Acts like a filter so K8s looks for Pods that match the following labels
    matchLabels: # Matches Pods with a specific label to determine which Pods will belong to this Deployment
      app: nextwork-flask-backend
  template: # Blueprint for the Pods that this Deployment will create
    metadata:
      labels: # Tags to help organize and identify Pods
        app: nextwork-flask-backend # Pods will be labeled with app: nextwork-flask-backend (they match matchLabels)
    spec: # Operational blueprint for the individual Pods the Deployment will create (the other spec was for the Deployment as a whole)
      containers:
        - name: nextwork-flask-backend
          image: 034362037253.dkr.ecr.sa-east-1.amazonaws.com/nextwork-flask-backend
          ports:
            - containerPort: 8080

^G Help        ^O Write Out   ^F Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^B Where Was
```

Everyone should be in a job they love.

Check out nextwork.org for more projects