FOSSASIA 2025 summit by
Wentao Leo Liu

Happy Integration! Insights Of Deploying GenAI Applications On Minikube

Agenda

0 Something about myself

1 Overview of Whisper-web

2 System Decomposition for Scalability

3 Containerization with Docker

4 Kubernetes Fundamentals

5 Minikube Cluster Setup

6 Deployment and Service YAML Configuration

7 Lessons Learned

8 Demo and Molly’s Greetings & Q&A

0 Something About Myself

- Cloud native learner
- Apache OpenMeetings contributor
- AI Video translator, transcriber
- Startup Founder omfoss.com
- Talked in FOSSASIA summit 2023

My Open Source Road From 2015 To 2025

foss asia

foss asia


Breda
Amsterdam C
07:49 IC
HAPPY CHINESE LUNAR NEW YEAR!


we
foss
asia

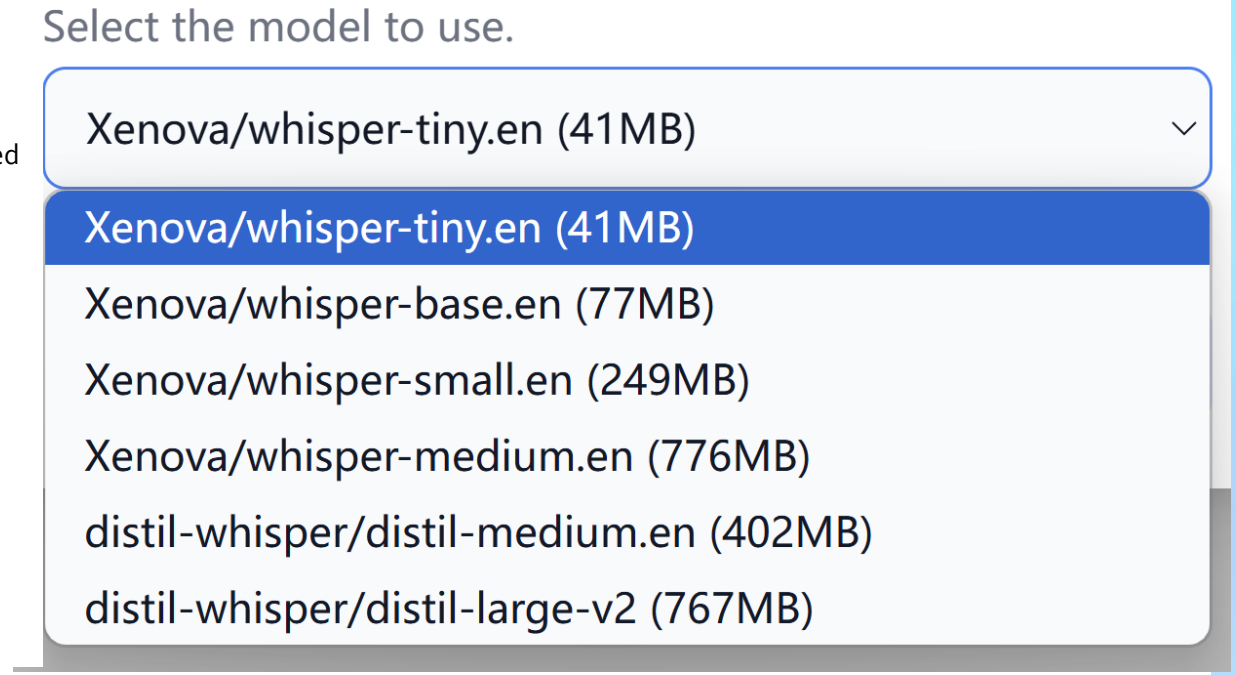
foss asia

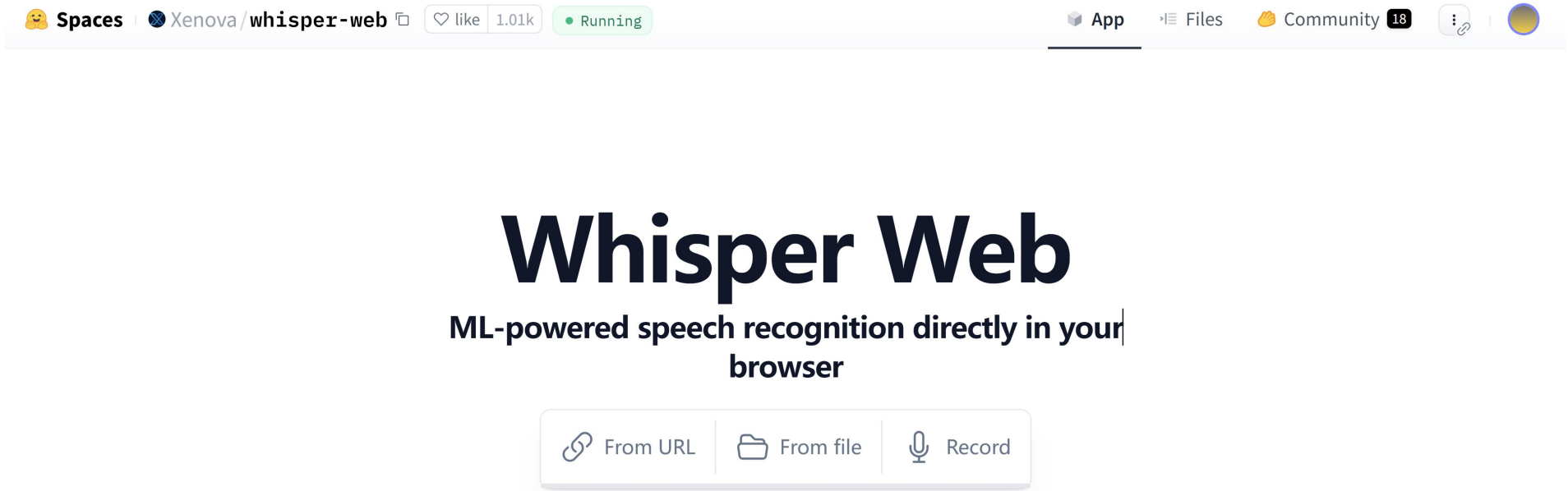
1 Overview of Whisper-web

- What is Whisper Web?
  - ✓ ML-powered speech recognition directly in your browser! Built with 🤗 Transformers.js TypeScript
  - ✓ Experimental WebGPU support has been added for GPU acceleration
- Openai whisper models
  - ✓ Whisper is a general-purpose speech recognition model. It is trained on a large dataset of diverse audio (680K hours)
  - ✓ a multitasking model that can perform multilingual speech recognition, speech translation, and language identification
  - ✓ Whisper's code and model weights are released under the MIT License.

Models and Parameters

Size	Parameters	English-only model	Multilingual model	Required VRAM	Relative speed
tiny	39 M	tiny.en	tiny	~1 GB	~10x
base	74 M	base.en	base	~1 GB	~7x
small	244 M	small.en	small	~2 GB	~4x
medium	769 M	medium.en	medium	~5 GB	~2x
large	1550 M	N/A	large	~10 GB	1x
turbo	809 M	N/A	turbo	~6 GB	~8x





Running locally

1. Clone the repo and install dependencies:
   git clone https://github.com/xenova/whisper-web.git
   cd whisper-web
   npm install
2. Run the development server
   npm run dev
3. Open the application in your browser
   https://localhost:5173

Note:if you wish to run this in dev mode in firefox, you will have to enable it manually via about:config and set dom.workers.modules.enabled to true

foss asia

2 System Decomposition for Scalability

Microservice Architecture

Frontend Component

The frontend serves as the user interface, allowing users to upload audio files and view transcriptions.

It is built using React and Node.js, providing a responsive and interactive experience.

API Gateway

The API Gateway routes requests to the appropriate services and handles authentication.

It ensures secure and efficient communication between the frontend and backend services.

Model Service

The Model Service runs the Whisper model for speech-to- text transcription.

It leverages PyTorch and CUDA for high- performance inference on GPUs.

3 Containization with Docker

- Docker image selection
  - Choosing the right base image is crucial for performance and compatibility.
  - For a high speed model service, a CUDA base image is required to enable GPU acceleration.
- Dependency management
  - Docker allows for the installation of necessary dependencies within the container.
  - This ensures consistency and avoids conflicts between different components.
- Multistage Builds
  - Multi- stage builds help reduce the final image size by separating build and runtime stages.
  - This improves security and reduces the attack surface of the container.

A simplest Dockerfile

```
FROM node:20                    #  select the base image

WORKDIR /whisper-web            #  setting work directory

COPY . .                        #  copy project files

RUN npm install                 #  install dependencies

EXPOSE 5173                     #  expose port

CMD ["npm", "run", "dev"]       #  define the cmd to start the app
```

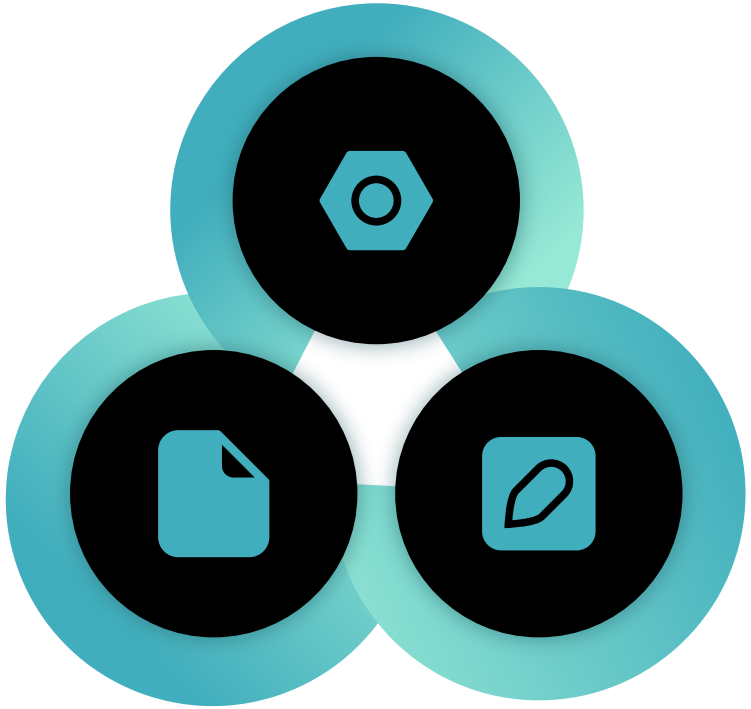
4 Kubernetes Fundamentals

Kubernetes Architecture

Pods

Pods are the smallest deployable units in Kubernetes, containing one or more containers.

They share resources such as storage and network, enabling efficient resource utilization.



Nodes

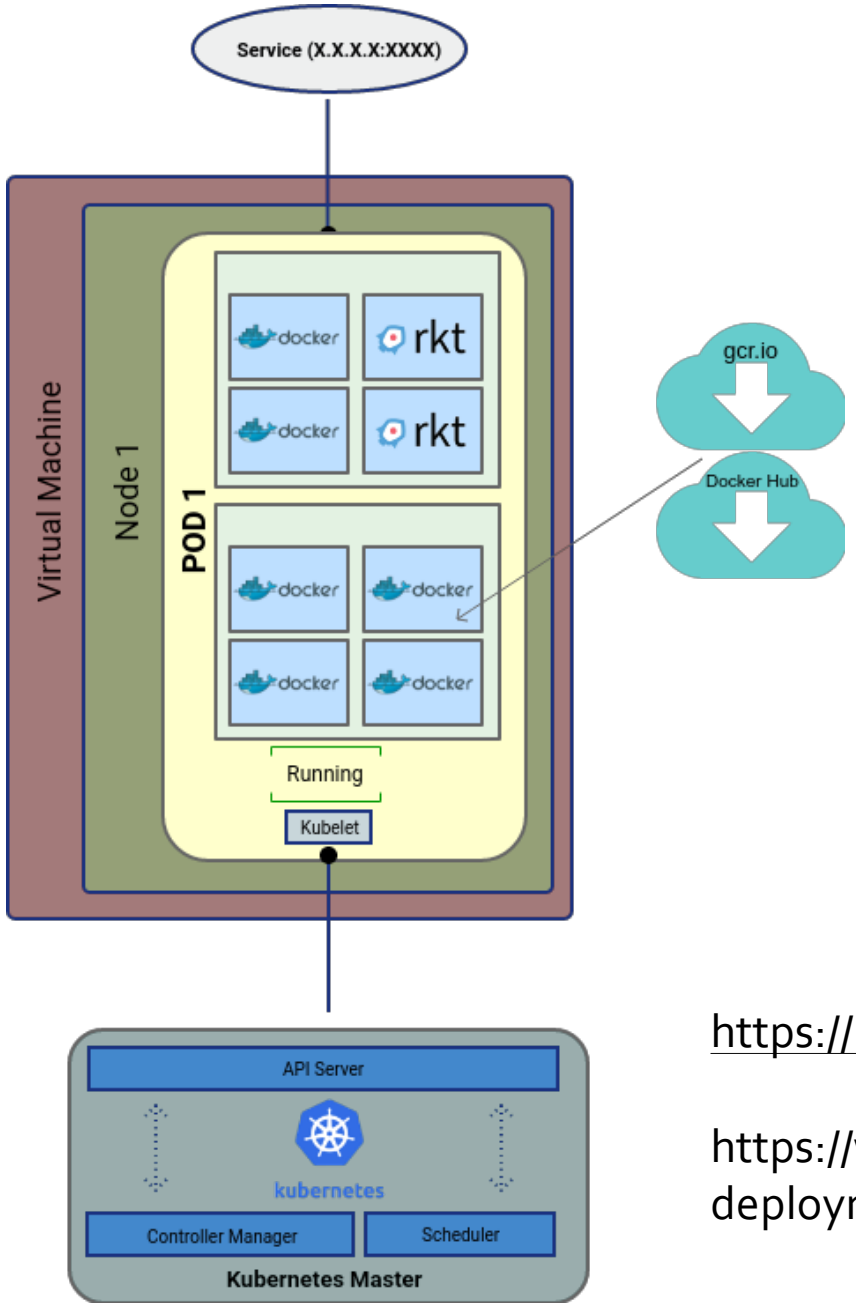
Nodes are the worker machines that run the pods.

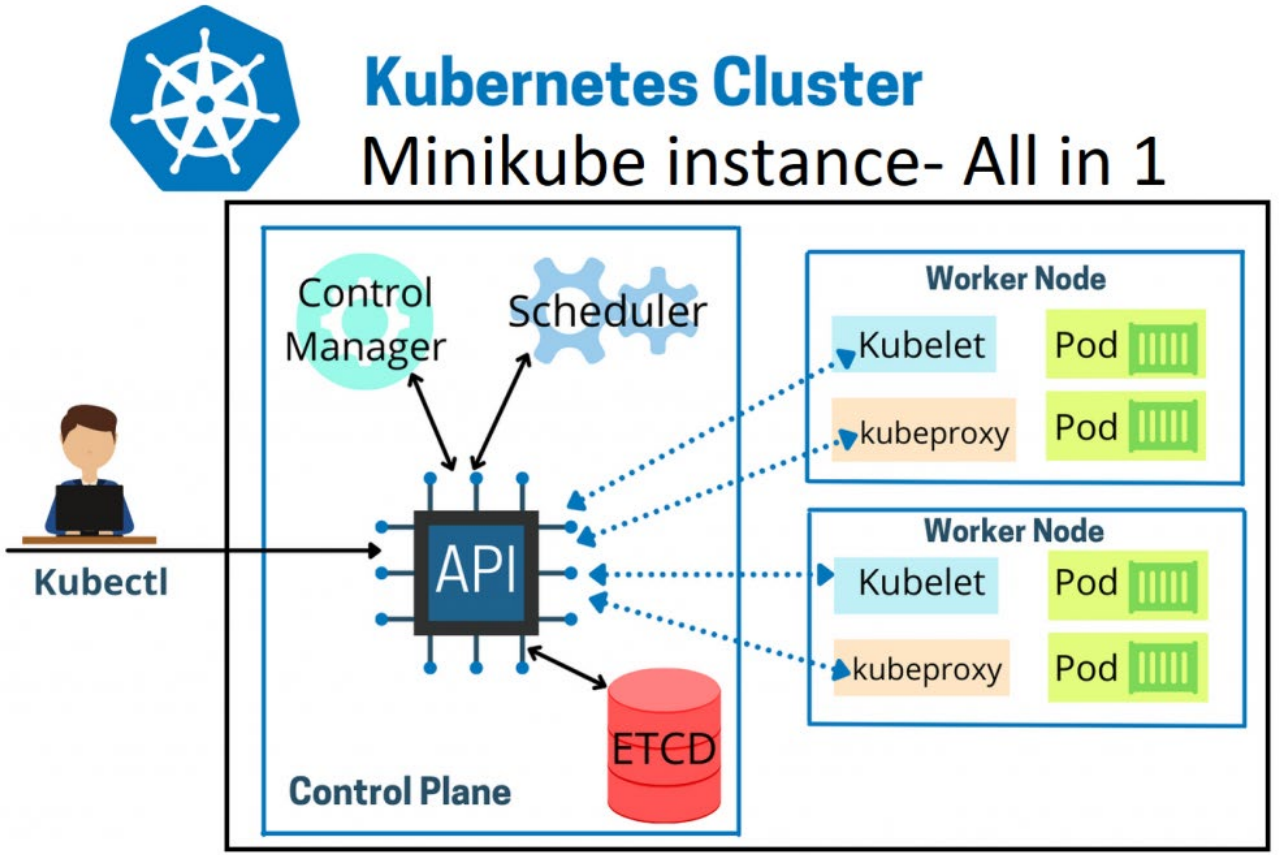
They can be physical or virtual machines, providing the necessary compute resources.

Control Plane

The Control Plane manages the cluster, including the API Server, Scheduler, and etcd.

It ensures the desired state of the cluster is maintained and provides high availability.





https://richardroseblog.wordpress.com/2017/11/01/minikube-creating-a-cluster/

https://www.linkedin.com/pulse/day-32-task-launching-your-kubernetes-cluster-deployment-singh/

foss asia

5 Minikube Cluster Setup

Setting up Minikube

Starting the Cluster

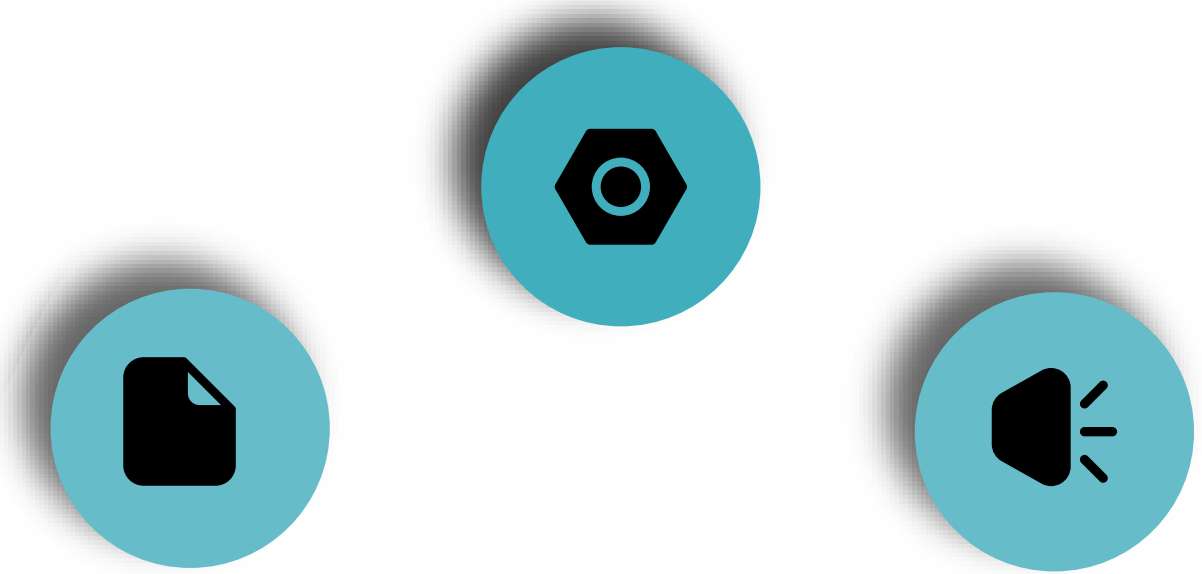
The cluster can be started with specific configurations such as CPU and memory allocation.

Enabling addons like Ingress allows for advanced networking capabilities.

Installation Process

Minikube requires Docker, kubectl, and itself to be installed on the local machine.

It provides a simple way to run a single- node Kubernetes cluster for development purposes.



Benefits of Minikube

Minikube simulates a production- like environment, making it easier to test deployments locally.

It allows developers to experiment with Kubernetes without the need for a full production setup.

Installation process

- Install docker https://docs.docker.com/engine/install/ubuntu/

- Testing

sudo docker run hello-world

- Install minikube

curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

- Install kubectl https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/

- Start minikube cluster

6 Deployment and Service configuration

Deployment YAML

Deployment Structure

The Deployment YAML file defines the desired state of the application, including the number of replicas.

It specifies the container image and resource requirements, such as GPU limits for AI workloads.

Label Selectors

Label selectors are used to match pods with the desired deployment.

They ensure that the correct pods are managed and scaled according to the deployment specifications.

Scaling and Redundancy

Deployments can be scaled horizontally by adjusting the number of replicas.

This provides redundancy and ensures high availability of the application.

Service YAML

01

Exposing Services

The Service YAML file exposes pods to the outside world or within the cluster. It provides a stable IP address and DNS name for accessing the application.

02

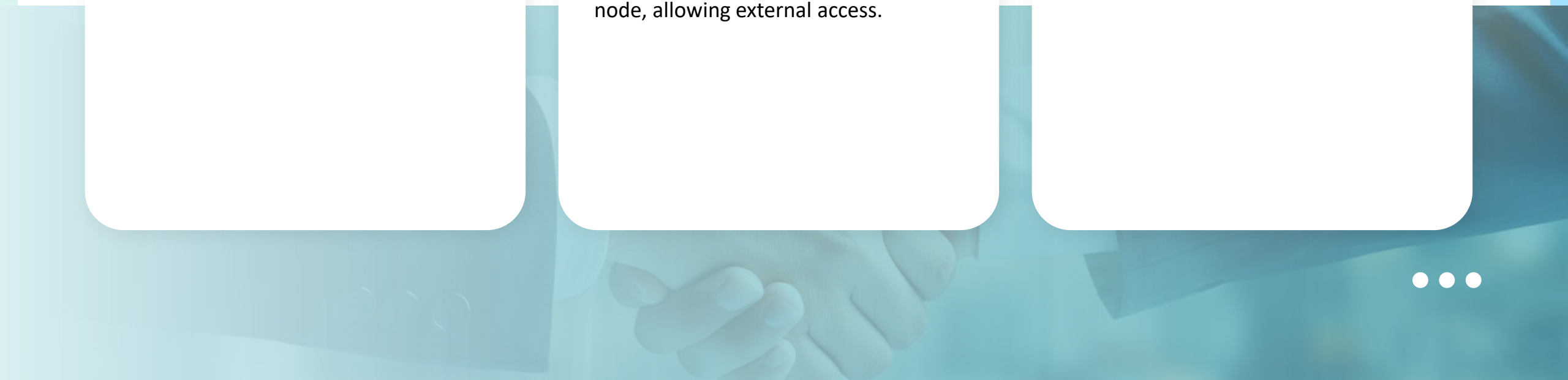
Service Types

ClusterIP services are used for internal communication between pods. NodePort services expose the application on a specific port of the node, allowing external access.

03

Load Balancing

Services can also provide load balancing across multiple pods, ensuring even distribution of traffic.



deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
    name: whisperweb-app
spec:
    replicas: 2
    selector:
        matchLabels:
            app: whisperweb-app
    template:
        metadata:
            labels:
                app: whisperweb-app
    spec:
        containers:
            - name: whisperweb-app
              image: cscmucmm/whisperweb-node-app:latest
              imagePullPolicy: Never
              ports:
                - containerPort: 5173
```

service.yaml

```
apiVersion: v1
kind: Service
metadata:
    name: whisperweb-app-service
spec:
    selector:
        app: whisperweb-app
    ports:
        - protocol: TCP
        port: 80
        targetPort: 5173
    type: LoadBalancer
```

7 Lessons learned

Challenges and Solutions

http_proxy and no_proxy environment variable setup

When using proxy to access internet, we need setup http_proxy and no_proxy environment variable for accessing internet resources and bypassing proxy for a certain range of IP address.

Using mirror image

Under the circumstance of no internet connection to docker.io service, you can not pull base images, so you need configure mirror image source or using proxy to pull kicbase image

Reduce Docker container image size

We use multi-stage build to reduce docker image size from 1.5G to 48M. This improvement greatly reduce the deployment time.

Multi-stage build dockerfile

```
FROM node:16-alpine AS build          # Build stage. Using node:16-alpine image as a base image

WORKDIR /whisper-web                  # Create app directory

COPY . .                              # Copy the source files

npm install                           # Install app dependenciesRUN

RUN npm run build                     # Build the React app for production

FROM nginx:alpine                     # Serve the application stage

COPY --from=build /whisper-web/dist /usr/share/nginx/html     # Copy build files to Nginx

EXPOSE 80                             # expose port

CMD ["nginx", "-g", "daemon off;"] #cmd to run nginx
```

8 Live Demo & Molly's Greetings & Q/A

Demo steps

Starting the Cluster

The demo begins with starting a Minikube cluster with docker driver. This sets up the environment for deploying the GenAI application.

Deploying Services

Two deployments and services, using single stage build and multi-stage build docker images, are deployed using kubectl apply.

This demonstrates the ease of deploying applications on Kubernetes.

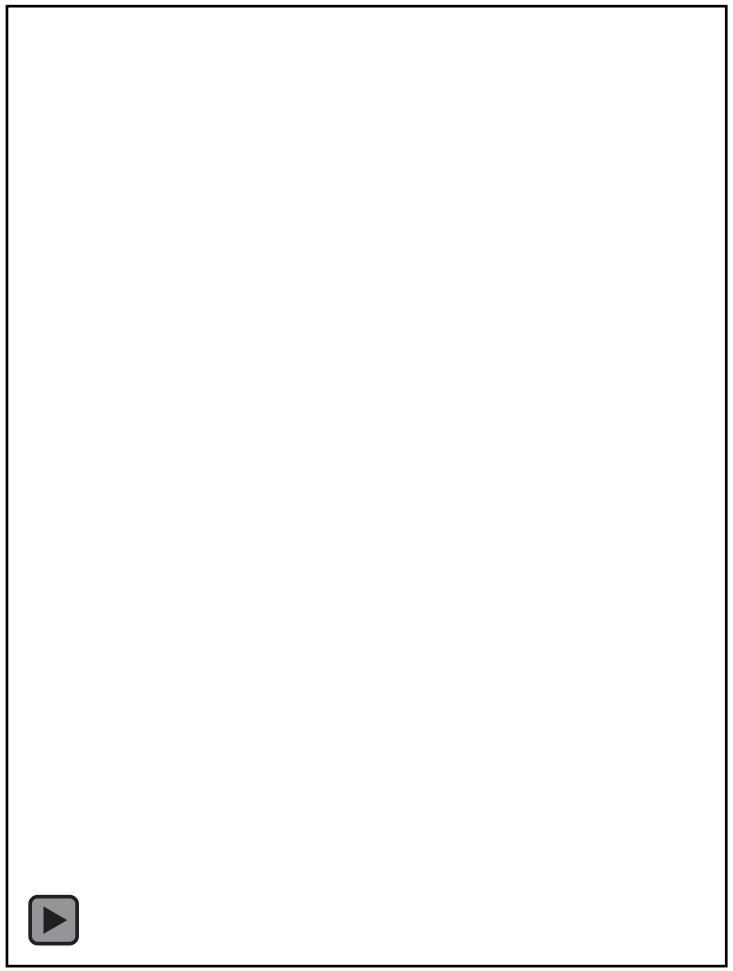
Transcribing Audio

An audio file is transcribed using the deployed application.

And an audience will record voice and testing the transcribing.

Note: Need close the system proxy setup

Molly’s greetings created by GenAI & Q/A session



THANK YOU

Wentao Liu
+86-18670022879
wliu@omfoss.com
www.omfoss.com