



Increasing Developer Productivity with Podman Desktop

Jayapriya Pai

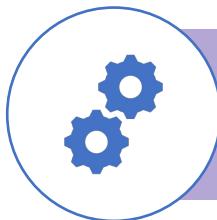
Senior Software Engineer, RedHat

About me

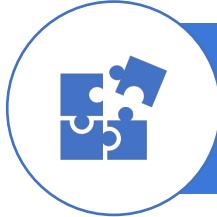


- Jayapriya Pai
- Senior Software Engineer, RedHat
- SRE turned Software Engineer
- Likes Playing Badminton
- slashpai (GitHub) 
- techieslash (Twitter/X) 
- jayapriyapai (LinkedIn) 

Takeaways



Containers and Podman Brief

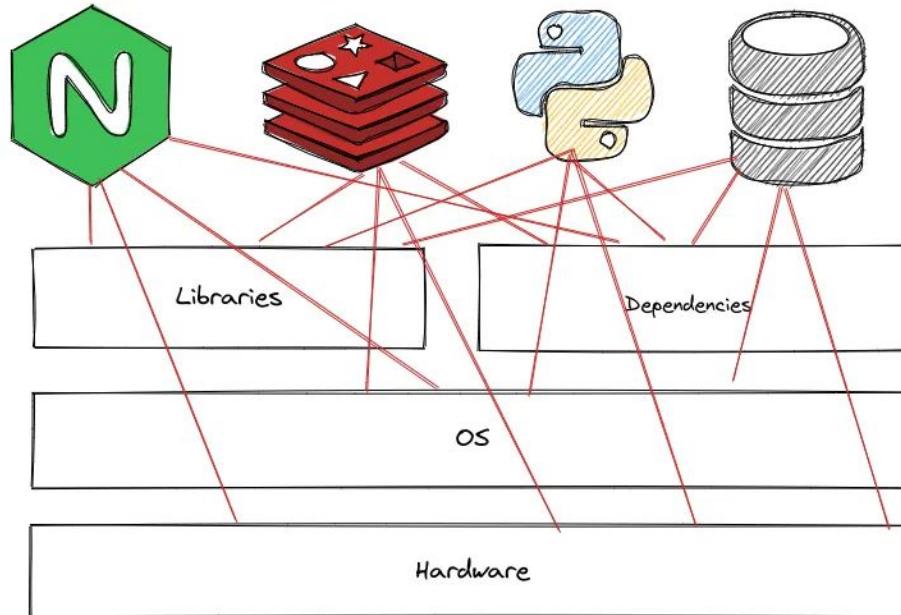


Podman Desktop to work with
Containers and Kubernetes



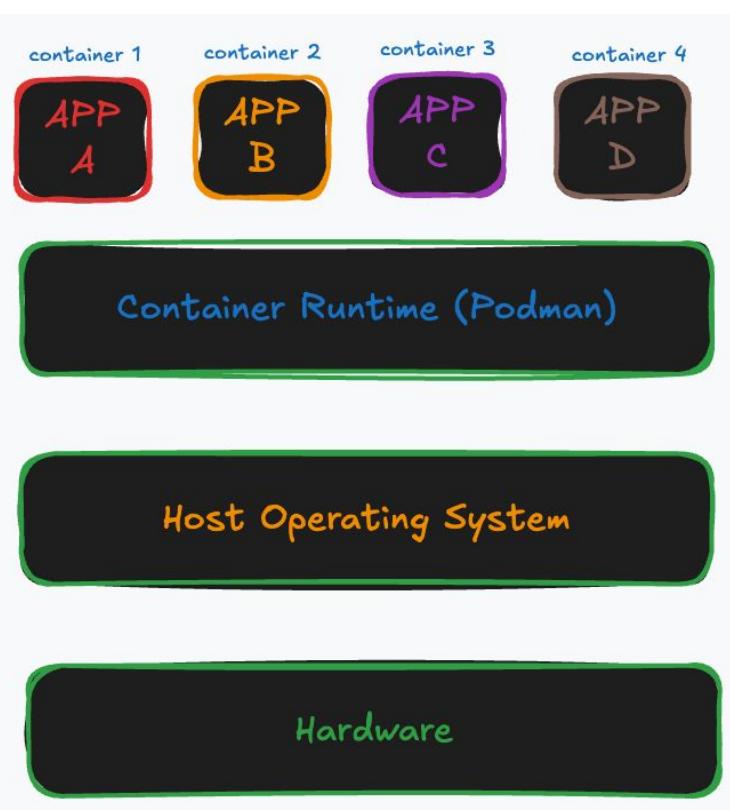
Podman AI Lab to test LLMs

Why Containerization?



Often referred
to as Matrix of
hell

Containerization



Each Container will have



The background features a minimalist design with abstract, overlapping circles in various shades of purple. A solid black rectangular bar is positioned horizontally across the middle of the image. In the lower right quadrant, there is a solid black square. The word "Podman" is centered in white, sans-serif font within this black square.

Podman



Podman Features

- Seamless way to work with containers and Kubernetes
- Open source first

Fast & Light

Daemonless with
seamless systemd
integration

Secure

Rootless containers
allow you to contain
privileges without
compromising the
functionality

Compatible

Compatible with other OCI
compliant container formats
including Docker as well as docker
compose files

Podman Desktop

Podman Desktop



“Podman Desktop builds upon the innovations brought by Podman, offering a powerful yet user-friendly environment for containerized development.”

“Now a CNCF Sandbox Project”

Read more:

<https://podman-desktop.io/blog/2024/11/14/podman-desktop-cncf>

Podman Desktop Features

- From a developer perspective, Podman Desktop really eases the pain of building, managing and running containers locally
- Enables developers to run Kubernetes locally, mirroring their production environment
- Enhanced by a plug-in system

Podman Machine

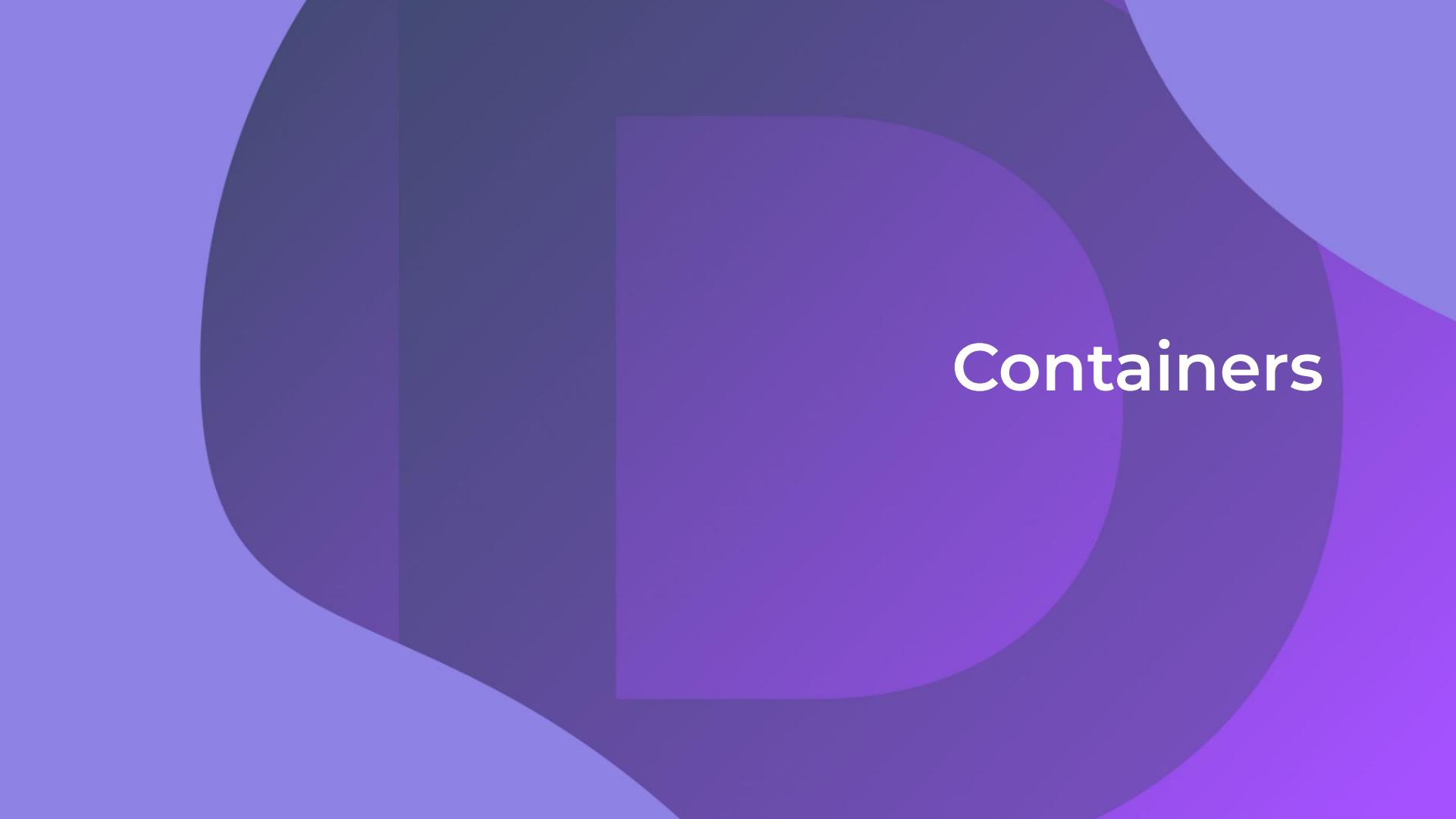
A screenshot of the Podman Machine creation interface. On the left is a sidebar with icons for Settings, Resources, Proxy, Registries, Authentication, CLI Tools, Kubernetes, Experimental (with a warning icon), and Preferences. The main area shows the "Create Podman machine" screen with the following fields:

- Name: podman-machine-default
- CPU(s): 8 (represented by a slider with a dot at the middle)
- Memory: 4 GB (represented by a slider with a dot near the beginning)
- Disk size: 100 GB (represented by a slider with a dot near the beginning)
- Image Path (Optional): A file browser icon.
- Image URL or image reference (Optional): A text input field with placeholder text: "Use 'registry/org/image:version' for image reference".
- Machine with root privileges: A toggle switch set to Enabled.
- Provider Type: A dropdown menu set to "default (Apple HyperVisor)".
- Start the machine now: A toggle switch set to Enabled.

At the bottom right are "Close" and "Create" buttons.

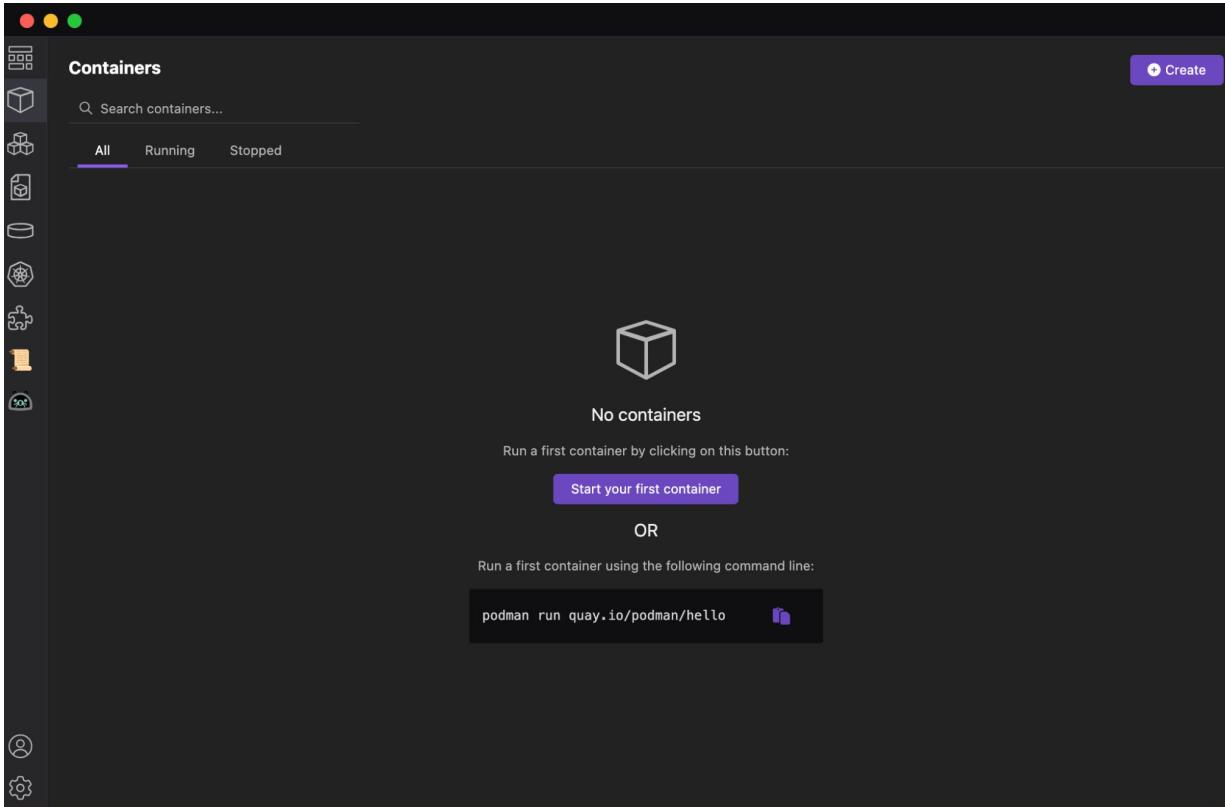
MacOs and Windows requires a virtual machine because containers are Linux

Optional on Linux

The background features a minimalist design with abstract, overlapping purple circles of varying sizes. A solid black rectangular frame is positioned in the center, containing the word "Containers" in white. The overall aesthetic is clean and modern.

Containers

Create the container



The screenshot shows the 'Containers' screen of the Podman desktop application. At the top, there's a search bar labeled 'Search containers...' and three tabs: 'All' (selected), 'Running', and 'Stopped'. On the left, there's a sidebar with icons for various container-related operations like creating, deleting, and inspecting containers. In the center, a large cube icon represents the absence of containers, with the text 'No containers' below it. Below this, there's a purple button labeled 'Start your first container'. Further down, the text 'Run a first container by clicking on this button:' is followed by another purple button with the command 'podman run quay.io/podman/hello' and a small terminal icon.

Containers

Search containers...

All Running Stopped

No containers

Start your first container

OR

Run a first container using the following command line:

```
podman run quay.io/podman/hello
```

Image



The screenshot shows the "Images" section of the Podman application. The interface has a dark theme with purple buttons. On the left, there's a sidebar with icons for various operations like Load, Import, Pull, and Build. Below that is a search bar labeled "Search images...". The main area displays a large icon of a document with a cube inside it, and the text "No images". Below this, instructions say "Pull a first image by clicking on this button:" followed by a purple button labeled "Pull your first image". Then it says "OR" and "Pull a first image using the following command line:" followed by a terminal-style window showing the command "podman pull quay.io/podman/hello" and a small icon of a terminal.

Images

Load Import Pull Build

Search images...

No images

Pull a first image by clicking on this button:

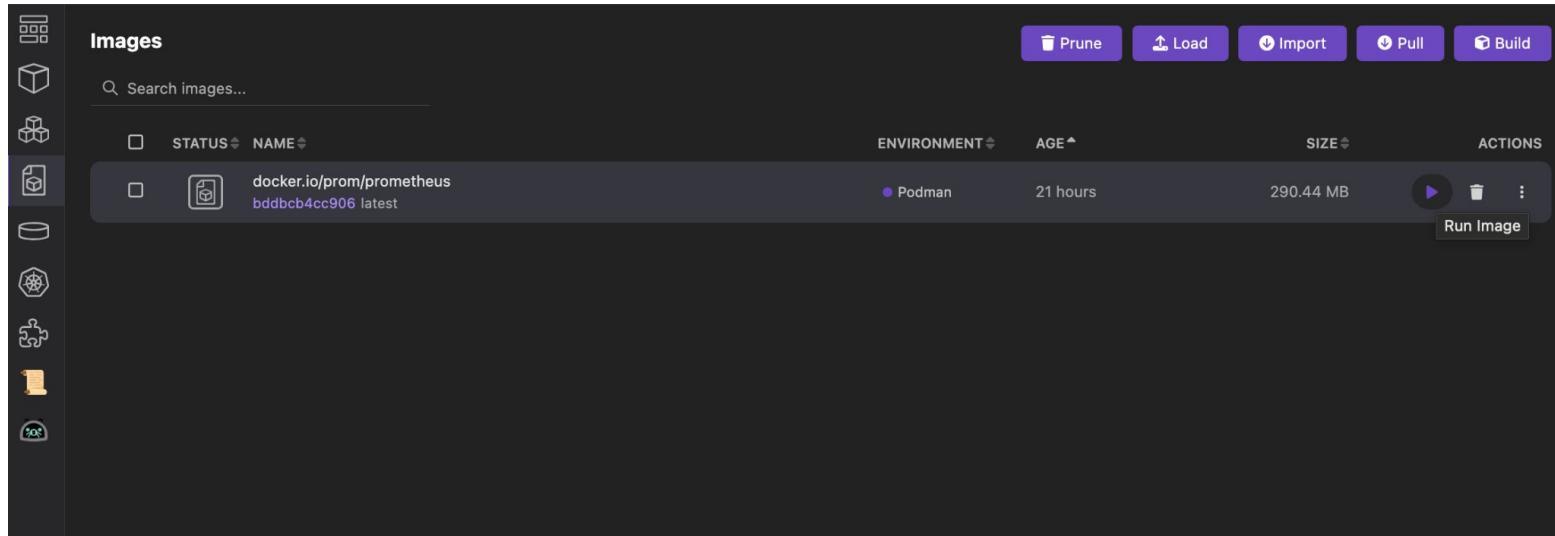
Pull your first image

OR

Pull a first image using the following command line:

```
podman pull quay.io/podman/hello
```

Running container from an image



The screenshot shows a dark-themed user interface for managing Docker containers. On the left, there's a vertical sidebar with icons for different operations: Images, Containers, Services, Networks, Volumes, Build, and Settings. The main area is titled "Images" and contains a search bar with the placeholder "Search images...". Below the search bar is a table header with columns: STATUS, NAME, ENVIRONMENT, AGE, SIZE, and ACTIONS. A single row is visible in the table, representing the Docker image "docker.io/prom/prometheus" with the ID "bddbcbb4cc906" and tag "latest". The environment is listed as "Podman". The image was created 21 hours ago and has a size of 290.44 MB. The "ACTIONS" column includes a play button icon, a trash can icon, and a three-dot menu icon, with a "Run Image" button highlighted in a light gray box.

STATUS	NAME	ENVIRONMENT	AGE	SIZE	ACTIONS
<input type="checkbox"/>	docker.io/prom/prometheus bddbcbb4cc906 latest	Podman	21 hours	290.44 MB	   Run Image

Running container from an image

The screenshot shows a Docker interface for creating a new container. The top navigation bar includes icons for Home, Images, Containers, and Run Image, with 'Images' currently selected. The main title is 'Create a container from image docker.io/prom/prometheus:latest'. The configuration page has tabs for Basic, Advanced, Networking, and Security, with 'Basic' selected.

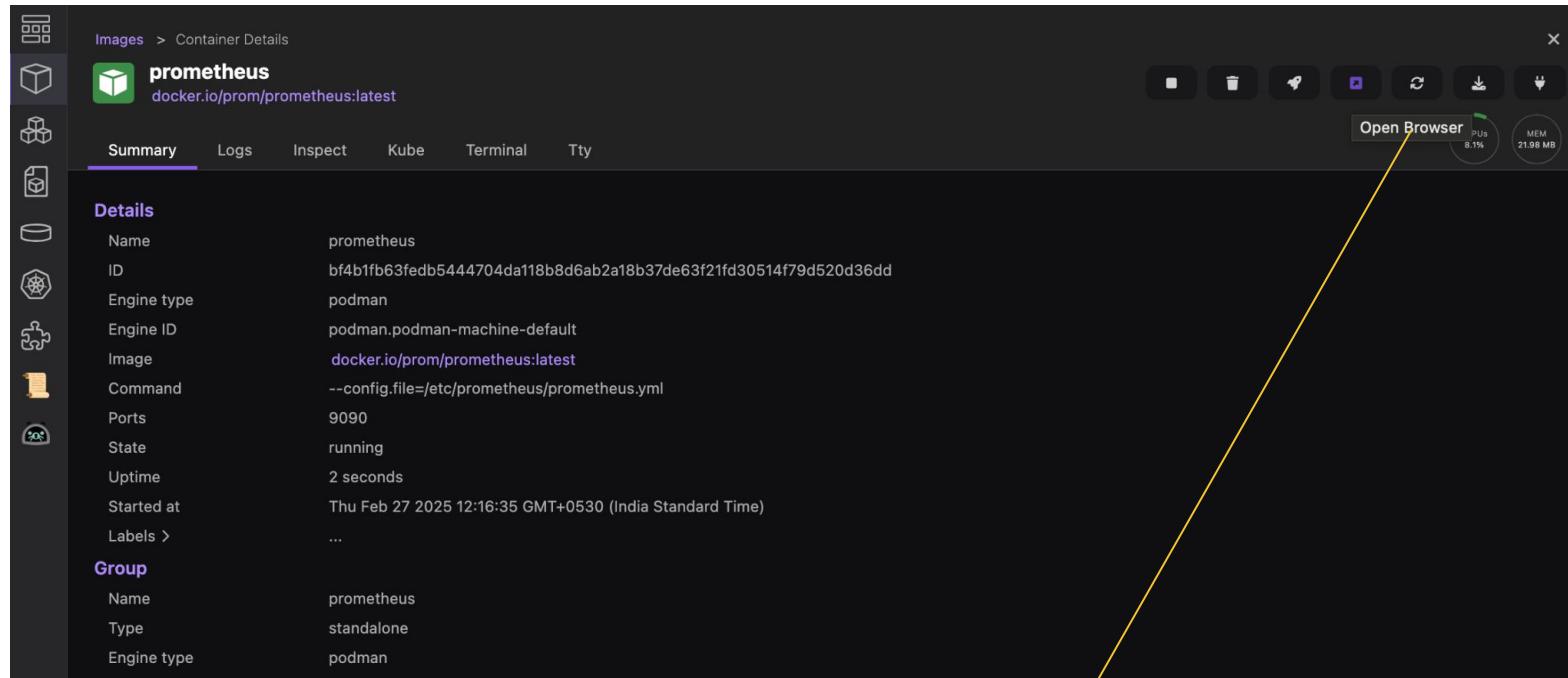
Basic Configuration:

- Name:** prometheus
- Entrypoint:** /bin/prometheus
- Command:** --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/prometheus
- Volumes:** A table with one row showing a host path (Path on the host) and a container path (Path inside the container). The host path is empty, and the container path is /prometheus. There is a '+' button to add more volumes.
- Port mapping:** A table with one row for port 9090/tcp. The local port is 9090 and the container port is 9090. There is a '-' button to remove this mapping and a link to 'Add custom port mapping'.

Actions:

- A large purple button at the bottom left labeled '▶ Start Container'.
- A close button 'X' in the top right corner.

Running container from an image



Images > Container Details

prometheus
docker.io/prom/prometheus:latest

Summary Logs Inspect Kube Terminal Tty

Details

Name	prometheus
ID	bf4b1fb63fedb5444704da118b8d6ab2a18b37de63f21fd30514f79d520d36dd
Engine type	podman
Engine ID	podman.podman-machine-default
Image	docker.io/prom/prometheus:latest
Command	--config.file=/etc/prometheus/prometheus.yml
Ports	9090
State	running
Uptime	2 seconds
Started at	Thu Feb 27 2025 12:16:35 GMT+0530 (India Standard Time)
Labels >	...

Group

Name	prometheus
Type	standalone
Engine type	podman

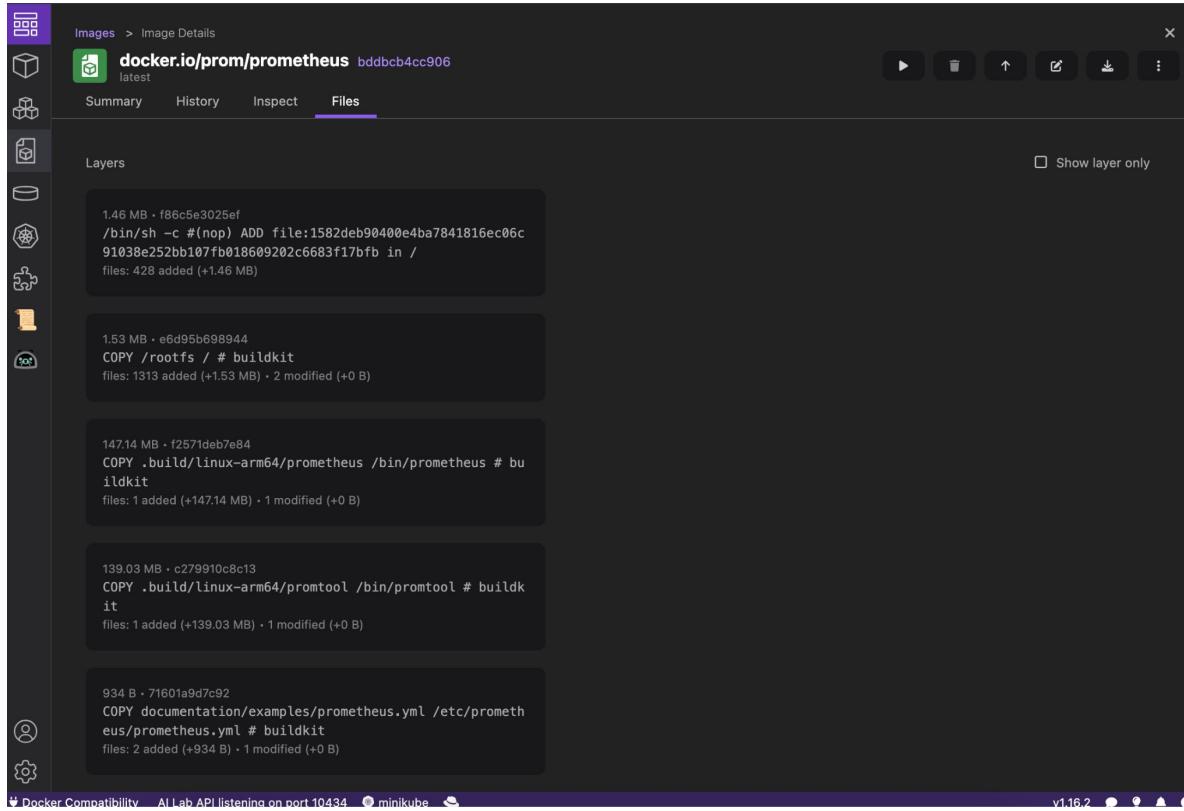
Port forward

Port forwarded



A screenshot of the Prometheus Time Series Collector interface. The browser title bar shows "Prometheus Time Series Collector". The address bar displays "localhost:9090/query". The main header includes the Prometheus logo, a search bar labeled "Query", an "Alerts" button, a "Status" dropdown, and user profile icons. Below the header is a query input field with placeholder text "Enter expression (press Shift+Enter for newlines)" and an "Execute" button. Underneath the input field are three tabs: "Table" (selected), "Graph", and "Explain". A "Evaluation time" selector with arrows is present. The main content area displays the message "No data queried yet". At the bottom left is a "Add query" button.

View Layers in Image

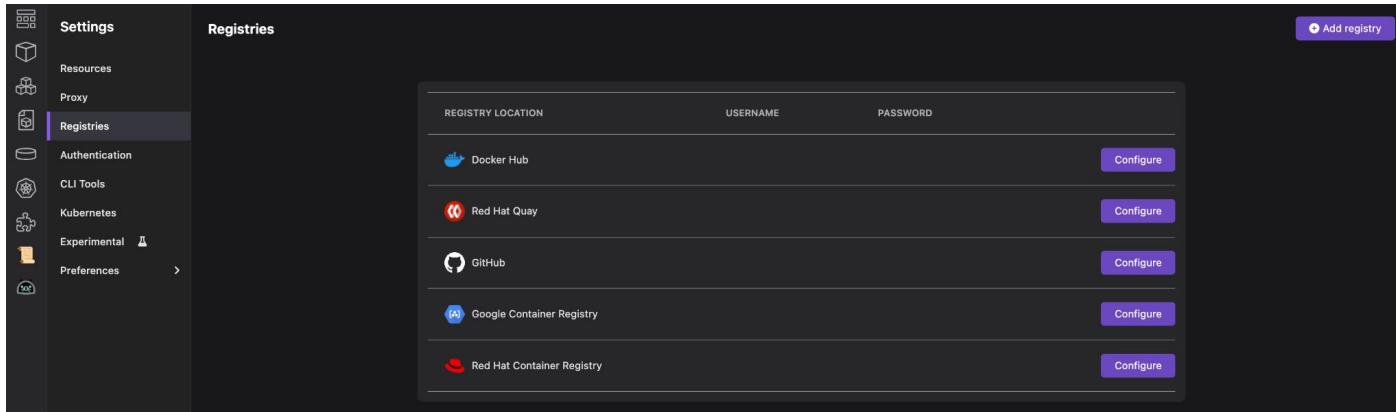


A screenshot of the Docker Desktop interface, specifically the "Image Details" view for the "docker.io/prom/prometheus" image. The "Files" tab is selected. The interface shows five stacked layers of the Docker image, each with its size, ID, and a snippet of the Dockerfile or build command.

- Layer 1: 1.46 MB · f86c5e3025ef
/bin/sh -c #(nop) ADD file:1582deb90400e4ba7841816ec06c91038e252bb107fb18609202c6683f17bfb in /
files: 428 added (+1.46 MB)
- Layer 2: 1.53 MB · e6d95b698944
COPY /rootfs / # buildkit
files: 1313 added (+1.53 MB) · 2 modified (+0 B)
- Layer 3: 147.14 MB · f2571deb7e84
COPY .build/linux-arm64/prometheus /bin/prometheus # buildkit
files: 1 added (+147.14 MB) · 1 modified (+0 B)
- Layer 4: 139.03 MB · c279910c8c13
COPY .build/linux-arm64/promtool /bin/promtool # buildkit
files: 1 added (+139.03 MB) · 1 modified (+0 B)
- Layer 5: 934 B · 71601a9d7c92
COPY documentation/examples/prometheus.yml /etc/prometheus/prometheus.yml # buildkit
files: 2 added (+934 B) · 1 modified (+0 B)

At the bottom left, there are icons for Docker Compatiblity, API lab API listening on port 10434, and minikube. At the bottom right, the version v16.2 is displayed along with other status icons.

Container Registries



A screenshot of a user interface for managing container registries. On the left, a sidebar menu lists various settings: Settings, Resources, Proxy, Registries (which is selected and highlighted in purple), Authentication, CLI Tools, Kubernetes, Experimental (with a warning icon), and Preferences. The main content area is titled "Registries". It displays a table with five rows, each representing a registry entry:

REGISTRY LOCATION	USERNAME	PASSWORD
Docker Hub		Configure
Red Hat Quay		Configure
Github		Configure
Google Container Registry		Configure
Red Hat Container Registry		Configure

In the top right corner of the main area, there is a purple button labeled "Add registry" with a plus sign icon.

Kubernetes

Local Kubernetes environment



Extensions

kind Filtered out 15 items of 16

Installed Catalog

 Kind extension ACTIVE

Integration for Kind: run local Kubernetes clusters using container "nodes"

Podman Desktop built-in extension v1.16.2



Extensions

minikube Filtered out 15 items of 16

Installed Catalog

 Minikube extension ACTIVE

Integration for minikube: a tool that makes it easy to run Kubernetes locally

v0.4.0



Local Kubernetes environment



Resources
Additional provider information is available under [Extensions](#)

The screenshot shows a sidebar with a "Settings" icon and a "Resources" section containing the following items:

- Podman**: A card for a "Podman Machine" which is "RUNNING". It shows a CPU usage of 8, Disk size of 98.9 GB, and Memory of 19.47 GB. It also lists "Podman endpoint" as unix:///var/folders/b4/yzjc9lb15r7... and "gpu-machine" as OFF.
- OpenShift Local**: A card for "Red Hat OpenShift Local" which is described as the quickest way to get started building OpenShift clusters. It links to "Red Hat OpenShift Local".
- minikube**: A card for a "minikube" cluster which is "RUNNING". It shows a CPU usage of 8, Disk size of 99.86 GB, and Memory of 22 GB. It lists "Kubernetes endpoint" as https://localhost:40543.
- kubectl**: A card for "kubectl" which is described as a command line tool for communicating with a Kubernetes cluster's control plane. It links to "kubernetes.io".
- Kind**: A card for "Kind" which is described as a Kubernetes utility for running local clusters using single-container "nodes", providing an easy way to create and manage Kubernetes environments for development and testing. It links to "kind.sigs.k8s.io".

Kind Cluster



Resources > Kind

Create Kind cluster

Settings

Name: kind-cluster

Provider Type: podman

HTTP Port: 9090

HTTPS Port: 9443

Setup an ingress controller (Contour <https://projectcontour.io>):

Enabled

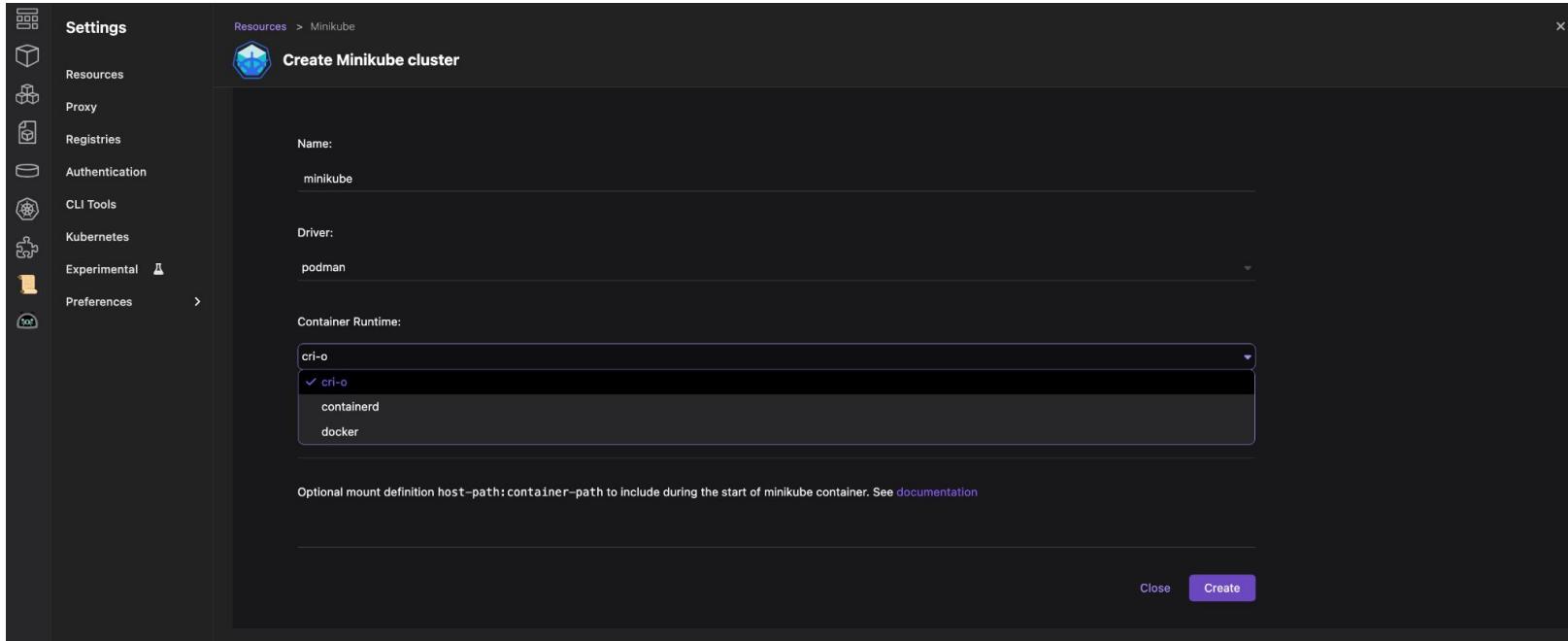
Node's container image (Available image tags on [kind/releases](#))

Leave empty for using latest.

Close **Create**

A screenshot of a web-based interface for managing clusters. On the left is a sidebar with icons for Settings, Resources, Proxy, Registries, Authentication, CLI Tools, Kubernetes, Experimental (with a warning icon), and Preferences. The main area shows a 'Create Kind cluster' dialog. It includes fields for 'Name' (set to 'kind-cluster'), 'Provider Type' (set to 'podman'), and ports for 'HTTP Port' (set to 9090) and 'HTTPS Port' (set to 9443). There is also a section for setting up an 'ingress controller' with a 'Contour' link and a 'Enabled' toggle switch which is turned on. At the bottom of the dialog are 'Close' and 'Create' buttons. The overall theme is dark with light-colored text and UI elements.

Minikube Cluster



The screenshot shows a dark-themed user interface for creating a Minikube cluster. On the left, a sidebar lists various settings: Settings, Resources, Proxy, Registries, Authentication, CLI Tools, Kubernetes, Experimental (with a warning icon), and Preferences. The main area is titled "Create Minikube cluster". It includes fields for "Name" (set to "minikube"), "Driver" (set to "podman"), and "Container Runtime" (a dropdown menu showing "cri-o" selected, along with "containerd" and "docker"). Below these fields is an optional note about host-path mounts. At the bottom right are "Close" and "Create" buttons.

Resources > Minikube

Create Minikube cluster

Name: minikube

Driver: podman

Container Runtime:

cri-o

✓ cri-o

containerd

docker

Optional mount definition host-path:container-path to include during the start of minikube container. See [documentation](#)

Close Create

Minikube Cluster



Containers

Prune Create

Search containers...

All Running Stopped

	Status	Name	Environment	Image	Age	Actions
<input type="checkbox"/>	RUNNING	minikube	Podman	gcr.io/k8s-minikube/kicbase:v0.0.45	8 minutes	Stop Remove ⋮
<input type="checkbox"/>	RUNNING	prometheus	Podman	docker.io/prom/prometheus:latest	31 minutes	Stop Remove ⋮

Kubernetes Dashboard



Kubernetes

Dashboard Connected

Nodes

Deployments

Services

Ingresses & Routes

Persistent Volume Claims

ConfigMaps & Secrets

Port Forwarding

Dashboard

Here you can manage and interact with Kubernetes clusters with features like connecting to clusters, and viewing workloads like deployments and services.

Get up and running by clicking one of the menu items!

[Kubernetes documentation](#)

Metrics

Nodes Active 1 Total 1

Deployments Active 0 Total 0

Services Total 1

Ingresses & Routes Total 0

Persistent Volume Claims Total 0

ConfigMaps & Secrets Total 1

> [Explore articles and blog posts](#)

From container to pod

Containers

Search containers... × Create Pod On 1 selected items.

Prune Create

All Running Stopped

STATUS	NAME	ENVIRONMENT	IMAGE	AGE	ACTIONS
RUNNING	minikube	Podman	gcr.io/k8s-minikube/kicbase:v0.0.45	13 minutes	⋮
RUNNING	prometheus	Podman	docker.io/prom/prometheus:latest	37 minutes	⋮

Open Logs Generate Kube Deploy to Kubernetes

Open Browser Open Terminal Restart Container

Export Container Generate Quadlet

From container to pod

Containers > Deploy to Kubernetes

Deploy generated pod to Kubernetes

Generated Kubernetes YAML:

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    creationTimestamp: "2025-02-27T07:26:52Z"
5    labels:
6      app: prometheus-pod
7      name: prometheus-pod
8  spec:
9    containers:
10      - args:
11        - "--config.file=/etc/prometheus/prometheus.yml"
12        - "--storage.tsdb.path=prometheus"
```

Pod Name:
`prometheus-pod`

Kubernetes Services:
 Replace `.hostPort` exposure on containers by Services. It is the recommended way to expose ports, as a cluster policy may prevent to use `hostPort`.

Restricted Security Context:
 Update Kubernetes manifest to respect the Pod security [restricted profile](#).

Expose Service Locally Using Kubernetes Ingress:
 Create an Ingress to get access to the local ports exposed, at the default Ingress Controller location. Example: On default kind cluster created with Podman Desktop, it will be accessible at '`localhost:9090`'. Requirements: Your cluster must have an Ingress Controller.

Kubernetes Context:
`minikube`

Kubernetes Namespace:
`default`

 Deploy

From container to pod



Containers > Deploy to Kubernetes

Deploy generated pod to Kubernetes

Prometheus-pod

Kubernetes Services:

Replace `.hostPort` exposure on containers by Services. It is the recommended way to expose ports, as a cluster policy may prevent to use `hostPort`.

Restricted Security Context:

Update Kubernetes manifest to respect the Pod security [restricted profile](#).

Expose Service Locally Using Kubernetes Ingress:

Create an Ingress to get access to the local ports exposed, at the default Ingress Controller location. Example: On default kind cluster created with Podman Desktop, it will be accessible at `'localhost:9090'`. Requirements: Your cluster must have an Ingress Controller.

Kubernetes Context:
minikube

Kubernetes Namespace:
default

Created pod:

Name: prometheus-pod
Phase: Running

Container statuses:

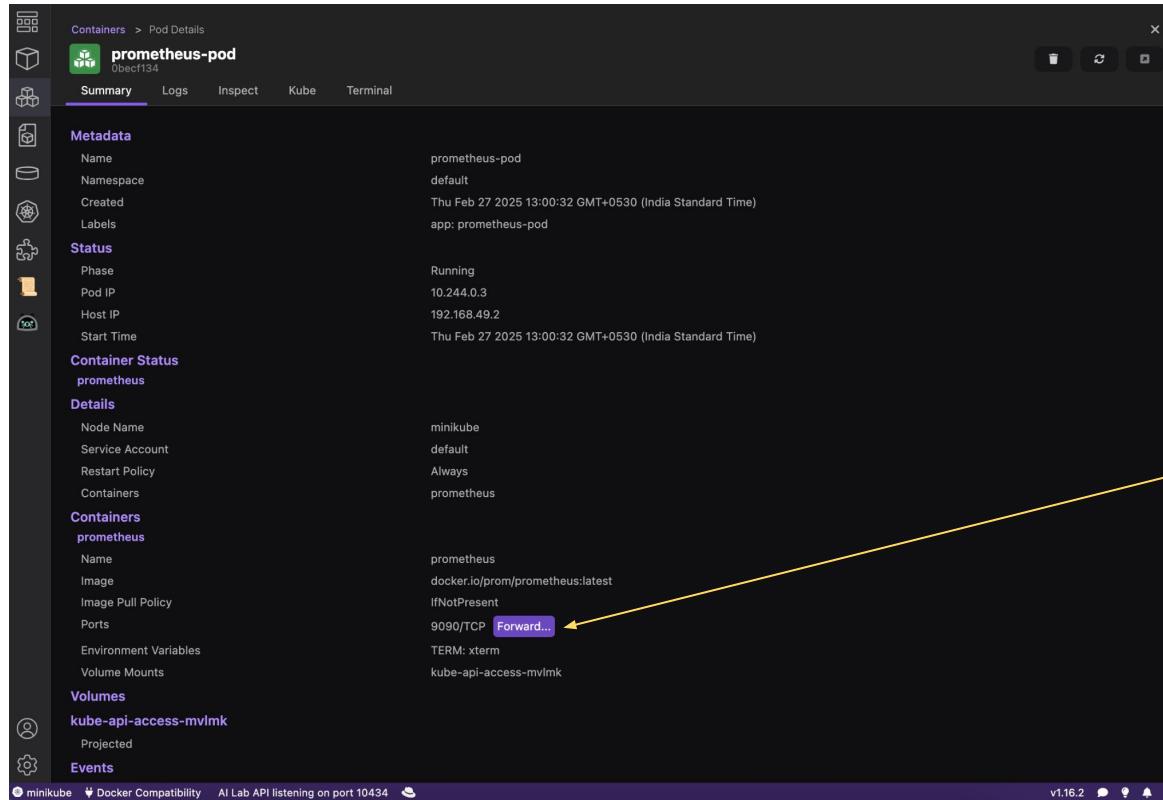
- `prometheus` Ready (Running)

```
74  status:
118  |  podIP: 10.244.0.3
119  |  podIPs:
120  |    - ip: 10.244.0.3
121  |  qosClass: BestEffort
122  |  startTime: 2025-02-27T07:30:32.000Z
```

Done **Open Pod**

minikube Docker Compatibility AI Lab API listening on port 10434 v1.16.2

Port Forwarding



Containers > Pod Details

prometheus-pod 0becf134

Summary Logs Inspect Kube Terminal

Metadata

Name	prometheus-pod
Namespace	default
Created	Thu Feb 27 2025 13:00:32 GMT+0530 (India Standard Time)
Labels	app: prometheus-pod

Status

Phase	Running
Pod IP	10.244.0.3
Host IP	192.168.49.2
Start Time	Thu Feb 27 2025 13:00:32 GMT+0530 (India Standard Time)

Container Status

prometheus	
-------------------	--

Details

Node Name	minikube
Service Account	default
Restart Policy	Always
Containers	prometheus

Containers

prometheus	
-------------------	--

Name	prometheus
Image	docker.io/prom/prometheus:latest
Image Pull Policy	IfNotPresent
Ports	9090/TCP Forward...
Environment Variables	TERM: xterm
Volume Mounts	kube-api-access-mvilmk

Volumes

kube-api-access-mvilmk	
-------------------------------	--

Events

minikube Docker Compatibility AI Lab API listening on port 10434 v1.16.2

Option to directly port forward

Podify



“Allows you to select multiple containers and group them together as a single unit of deployment for Kubernetes”

The screenshot shows the Podify interface for managing containers. On the left is a sidebar with icons for different resources: Container (selected), Pod, Service, Deployment, Ingress, ConfigMap, Secret, and Event. The main area has a dark header with a search bar, a trash can icon, a "Create Pod" button, and buttons for "Prune" and "Create". Below the header is a table titled "Containers" with columns: STATUS, NAME, ENVIRONMENT, IMAGE, AGE, and ACTIONS. The table lists three containers:

STATUS	NAME	ENVIRONMENT	IMAGE	AGE	ACTIONS
<input type="checkbox"/>	minikube	Podman	gcr.io/k8s-minikube/kicbase:v0.0.45	31 minutes	<input type="checkbox"/> <input type="trash"/> <input type="more"/>
<input checked="" type="checkbox"/>	nginx	Podman	docker.io/library/nginx:latest	30 seconds	<input type="checkbox"/> <input type="trash"/> <input type="more"/>
<input checked="" type="checkbox"/>	prometheus	Podman	docker.io/prom/prometheus:latest	55 minutes	<input type="checkbox"/> <input type="trash"/> <input type="more"/>

Podify



Containers > Create Pod

Copy containers to a pod

⚠ Containers nginx and nginx use same port 80

Name of the pod:

my-pod

Containers to replicate to the pod:

	1. prometheus	(bf4b1fb)
	2. nginx	(6574f0d)

All selected ports will be exposed:

<input checked="" type="checkbox"/>	Port 9090	prometheus
<input checked="" type="checkbox"/>	Port 8080	nginx
<input checked="" type="checkbox"/>	Port 9000	nginx

[Close](#) [Create Pod](#)

Podify



Pods > Pod Details

my-pod
7d6ff6a7

Summary Logs Inspect Kube

Details

Name	my-pod
ID	7d6ff6a7eb5463a0a54b6a2d5ecd7a507186f187ad947e5420e4ac5f4c4ff256
Created	Thu Feb 27 2025 13:13:26 GMT+0530 (India Standard Time)
Age	11 seconds

Pod Status

Status	running
--------	---------

Containers

7d6ff6a7eb54-infra	7bdfd6d3cf4a3bd4405e8956befaea780ecf682b02d430e136a21638e725553
prometheus-podified	c9d705f34184c16089c04d1adb4c4123171b4ea64dba1cc186c3ba143919b0d6
nginx-podified	e256d2d005a94c8c3812e42c5796a5eafdf994446df95177e63a84ffada21707c

Actions

- Open 8080
- Open 9090

Streamlining the image iteration process



How to make images build with podman readily available in Kubernetes cluster?



Streamlining the image iteration process

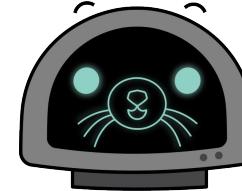
- To push directly to CRI-O, configure Podman client on your host using the `podman-env` command in your shell:

```
eval $(minikube podman-env)
```

- You should now be able to use Podman client on the command line on your host machine talking to the Podman service inside the minikube
- Turn off the `imagePullPolicy:Always` (use `imagePullPolicy:IfNotPresent` or `imagePullPolicy:Never`), as otherwise Kubernetes won't use images you built locally.

Podman AI Lab

Podman AI Lab



“Extension to work with LLMs (Large Language Models) on Podman Desktop”

Helps to start building on AI

Curated catalog called recipes to try AI Use Cases and models

Playgrounds to experiment and test models

Perfectly fits into your local container workflow and exposes LLMs through inference APIs that you can directly access from your application containers.



Podman AI Lab

The screenshot shows the Podman Extensions interface. On the left, there's a sidebar with icons for different categories: Container, Image, Volume, Network, and Pod. The main area is titled "Extensions" and has a search bar with the query "ai". Below the search bar, it says "Filtered out 14 items of 15". There are two tabs: "Installed" (which is selected) and "Catalog". A purple button at the top right says "Install custom...". In the main list, there is one item: "Podman AI Lab extension" (ACTIVE). It has a "v1.4.0" badge and several small icons for managing the extension.

Extensions

ai

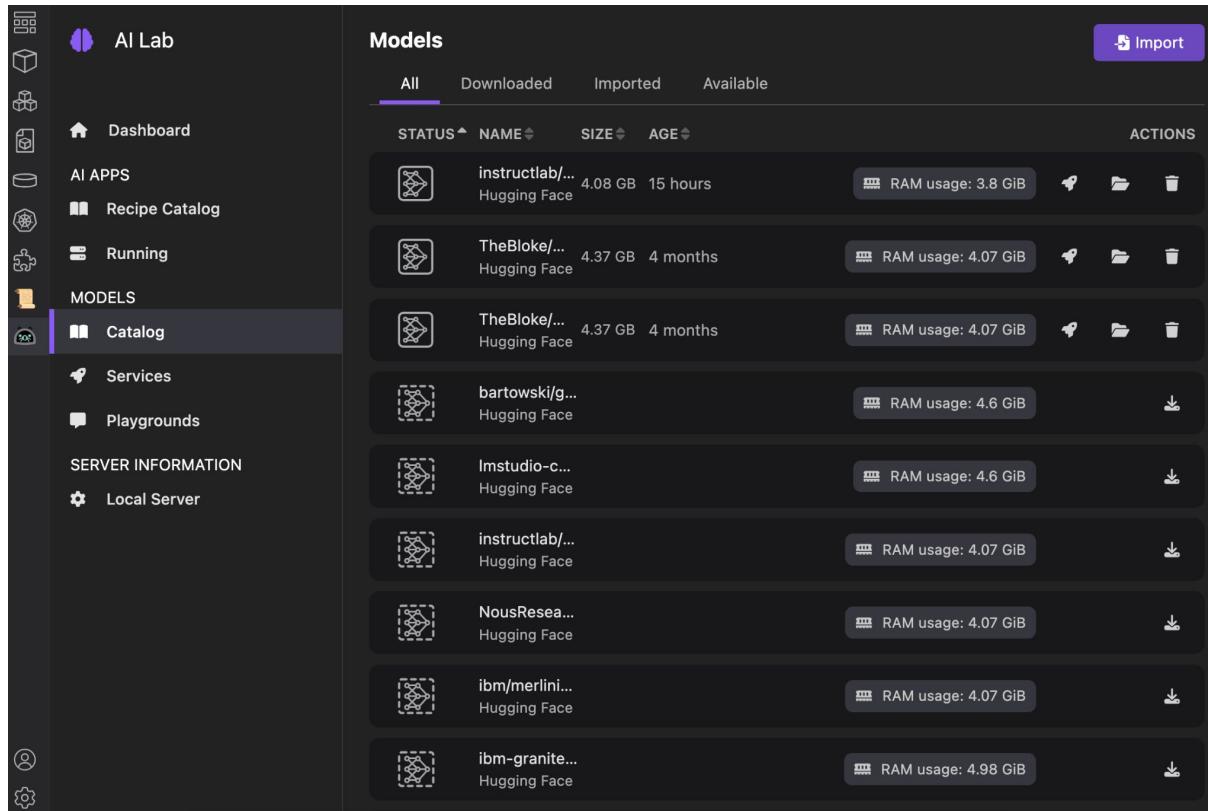
Filtered out 14 items of 15

Installed Catalog

Podman AI Lab extension ACTIVE v1.4.0

Podman AI Lab lets you work with LLMs locally, exploring AI fundamentals, experimenting with models and prompts, and serving models while maintaining data security and privacy.

Podman AI Lab: Model Catalog



The screenshot shows the 'AI Lab' interface with the 'Catalog' tab selected in the sidebar. The main area displays a list of available models, each with its name, size, age, and RAM usage.

NAME	SIZE	AGE	RAM usage
instructlab/...	4.08 GB	15 hours	3.8 GiB
TheBloke/...	4.37 GB	4 months	4.07 GiB
TheBloke/...	4.37 GB	4 months	4.07 GiB
bartowski/g...	Hugging Face		4.6 GiB
lmstudio-c...	Hugging Face		4.6 GiB
instructlab/...	Hugging Face		4.07 GiB
NousResear...	Hugging Face		4.07 GiB
ibm/merlini...	Hugging Face		4.07 GiB
ibm-granite...	Hugging Face		4.98 GiB

Podman AI Lab: Recipe Catalog

 AI Lab

 Dashboard

 AI APPS

 Recipe Catalog

 Running

 MODELS

 Catalog

 Services

 Playgrounds

SERVER INFORMATION

 Local Server

Recipe Catalog

Natural Language Processing

ChatBot
This recipe provides a blueprint for developers to create their own AI-powered chat applications using Streamlit.
v1.3.3.2 

Summarizer
This recipe guides into creating custom LLM-powered summarization applications using Streamlit.
v1.3.3.2 

Code Generation
This recipes showcases how to leverage LLM to build your own custom code generation application.
v1.3.3.2 

RAG Chatbot
This application illustrates how to integrate RAG (Retrieval Augmented Generation) into LLM applications enabling to interact with your own documents.
v1.3.3.2 

Java-based ChatBot (Quarkus)
This is a Java Quarkus-based recipe demonstrating how to create an AI-powered chat applications.
v1.3.3.2 

Node.js based ChatBot
This is a NodeJS based recipe demonstrating how to create an AI-powered chat applications.
v1.3.3.2 

Function calling
This recipes guides into multiple function calling use cases, showing the ability to structure data and chain multiple tasks, using Streamlit.
v1.3.3.2 

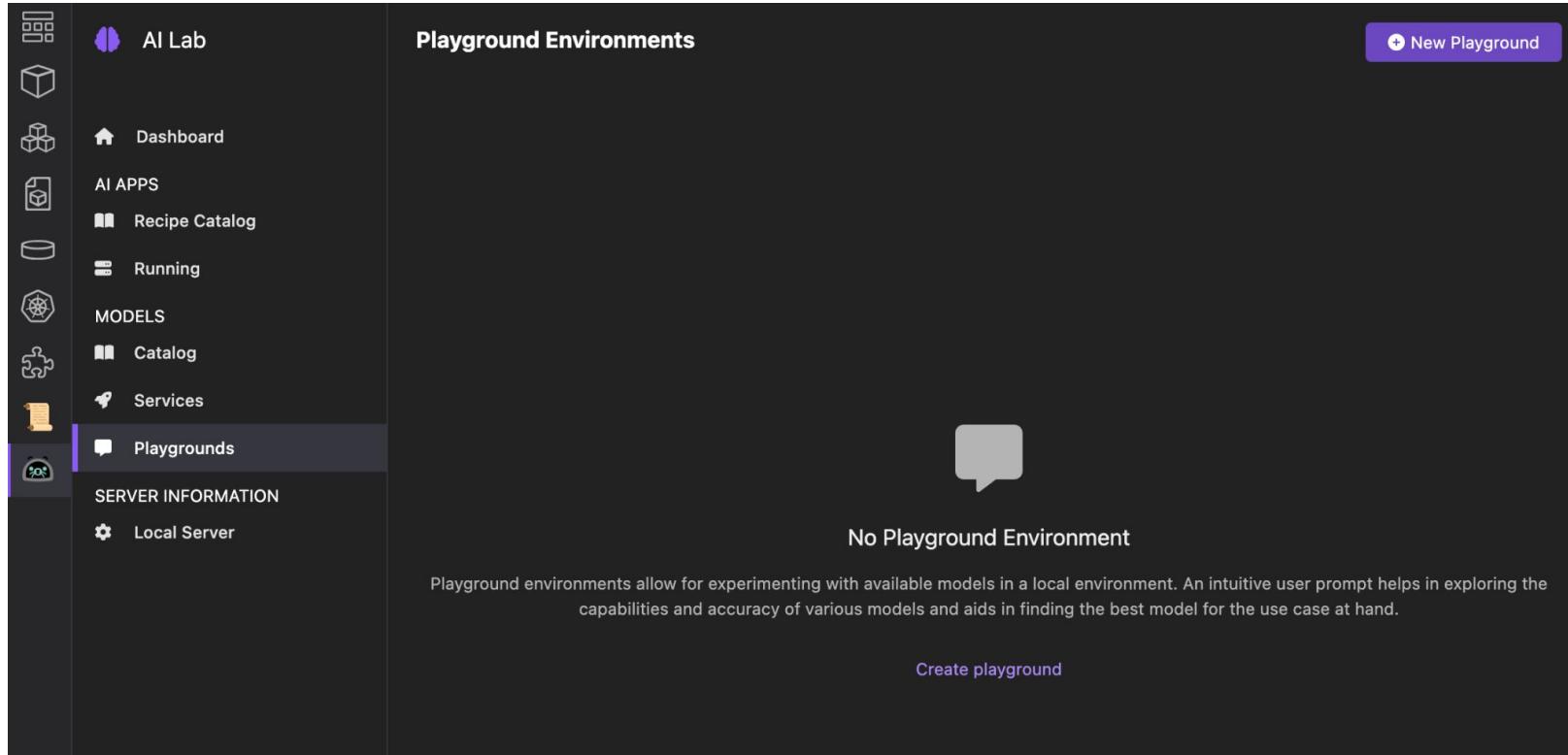
Audio

Audio to Text
This application demonstrate how to use LLM for transcribing an audio into text.
v1.3.3.2 

Computer Vision

Object Detection
This recipe illustrates how to use LLM to interact with images and build object detection applications.
v1.3.3.2 

Podman AI Lab: Playground



The screenshot shows the Podman AI Lab interface with a dark theme. On the left is a sidebar with various icons and links:

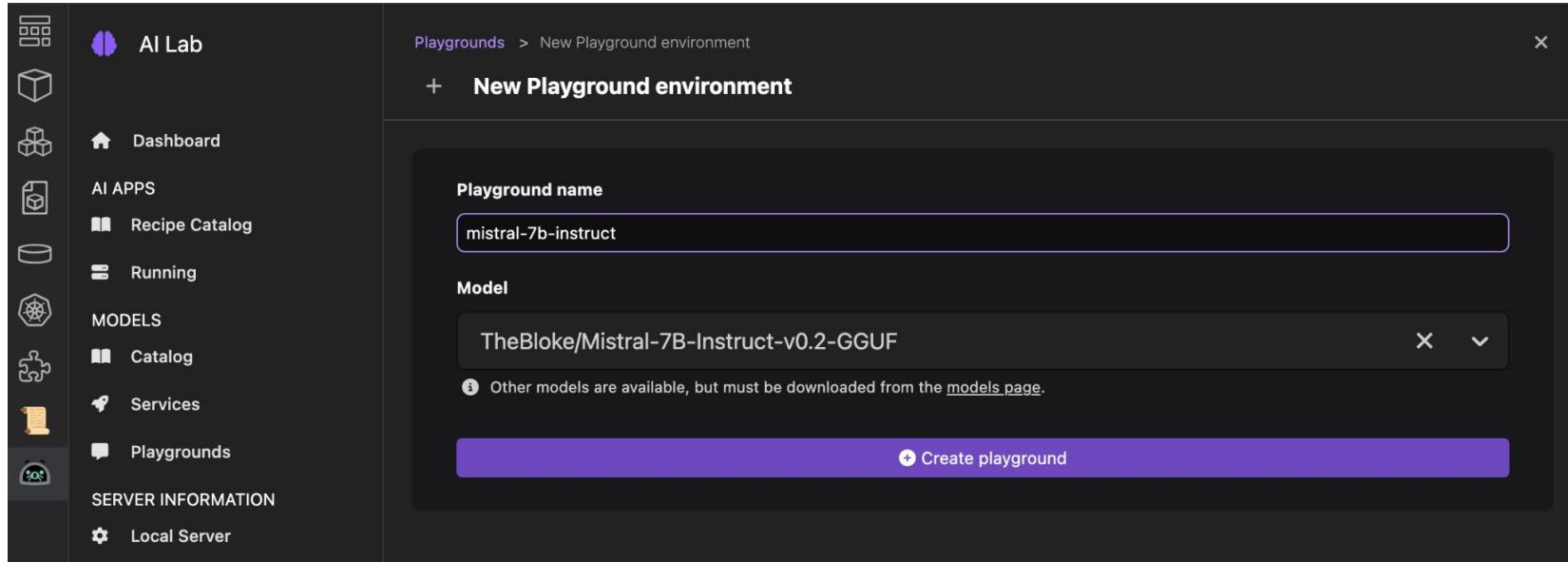
- AI Lab (selected)
- Dashboard
- AI APPS
- Recipe Catalog
- Running
- MODELS
- Catalog
- Services
- Playgrounds (highlighted with a purple bar)
- SERVER INFORMATION
- Local Server

The main content area is titled "Playground Environments". It features a large button labeled "New Playground" with a plus sign icon. Below it is a large speech bubble icon. The text "No Playground Environment" is displayed. At the bottom, there is a link "Create playground".

Playground environments allow for experimenting with available models in a local environment. An intuitive user prompt helps in exploring the capabilities and accuracy of various models and aids in finding the best model for the use case at hand.

Create playground

Podman AI Lab: Playground



The screenshot shows the Podman AI Lab interface with a dark theme. On the left, there is a sidebar with various icons and labels:

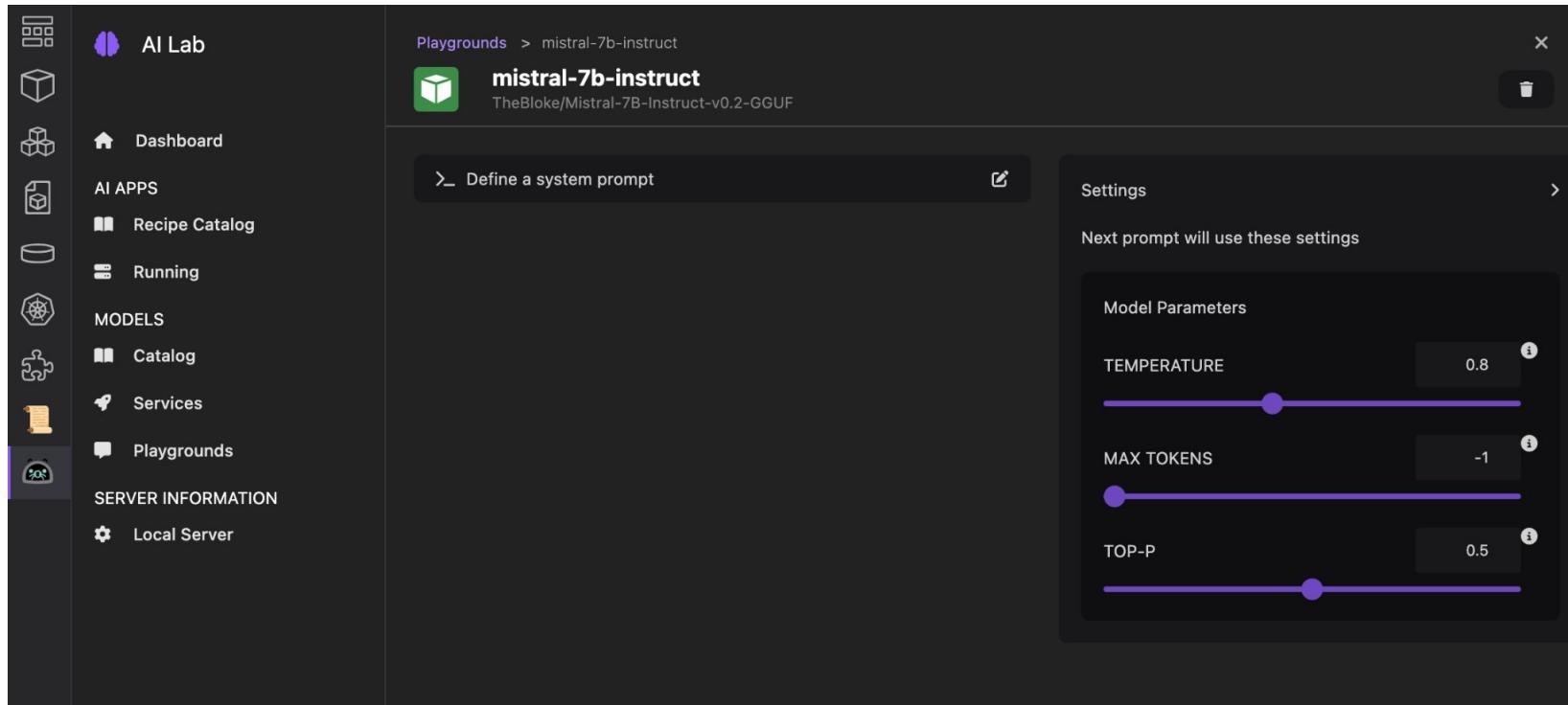
- AI Lab (selected)
- Dashboard
- AI APPS
- Recipe Catalog
- Running
- MODELS (selected)
- Catalog
- Services
- Playgrounds
- SERVER INFORMATION
- Local Server

The main area is titled "Playgrounds > New Playground environment". It has a heading "+ New Playground environment". Below it, there is a form with fields:

- Playground name:** mistral-7b-instruct
- Model:** TheBloke/Mistral-7B-Instruct-v0.2-GGUF

A note below the model field says: "Other models are available, but must be downloaded from the [models page](#)". At the bottom is a purple button labeled "+ Create playground".

Podman AI Lab: Playground



The screenshot shows the Podman AI Lab interface with a dark theme. On the left is a sidebar with various icons and labels:

- AI Lab (selected)
- Dashboard
- AI APPS
- Recipe Catalog
- Running
- MODELS
- Catalog
- Services
- Playgrounds (selected)
- SERVER INFORMATION
- Local Server

The main area displays the "Playgrounds > mistral-7b-instruct" screen. It features a card for "mistral-7b-instruct" by "TheBloke/Mistral-7B-Instruct-v0.2-GGUF". Below the card is a text input field labeled "Define a system prompt" with a "Edit" icon. To the right is a "Settings" panel titled "Settings" with the sub-instruction "Next prompt will use these settings". The "Model Parameters" section contains three sliders:

- TEMPERATURE: Set to 0.8
- MAX TOKENS: Set to -1
- TOP-P: Set to 0.5

Podman AI Lab: Playground

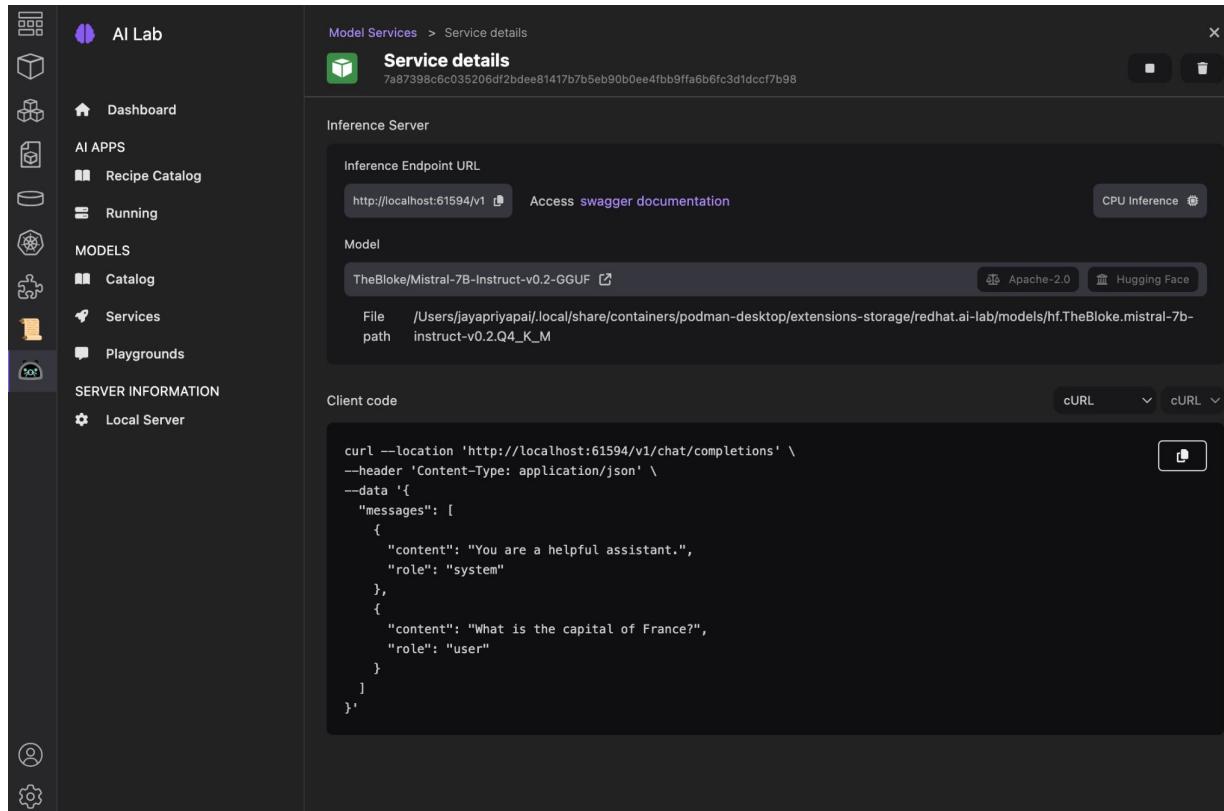
The screenshot shows the Podman AI Lab interface with a dark theme. On the left is a sidebar with various icons and labels:

- AI Lab (selected)
- Dashboard
- AI APPS
- Recipe Catalog
- Running
- MODELS
- Catalog
- Services
- Playgrounds (selected)
- SERVER INFORMATION
- Local Server

The main area displays a playground session titled "mistral-7b-instruct". The session details show it's based on "TheBloke/Mistral-7B-Instruct-v0.2-GGUF". The user prompt is "What is Prometheus?". The AI response is a detailed explanation of what Prometheus is, mentioning its creation by Brian Fernandes and Sauleau Lucas from SoundCloud, its use for monitoring and alerting, and its powerful query language. The response concludes with information about its open-sourcing in 2012 and widespread popularity.

At the bottom of the response box, it says "13.6 s". To the right, there are "Settings" and "Model Parameters" sections. The "Model Parameters" section includes sliders for TEMPERATURE (0.8), MAX TOKENS (-1), and TOP-P (0.5).

Podman AI Lab: Inference Server



The screenshot shows the Podman AI Lab interface with a dark theme. On the left, there's a sidebar with various icons and sections: AI Lab (selected), Dashboard, AI APPS, Recipe Catalog, Running, MODELS, Catalog, Services, Playgrounds, SERVER INFORMATION, and Local Server.

The main area is titled "Service details" for a service named "7a87398c6c035206df2bdee81417b7b5eb90b0ee4fbb9ffa6b6fc3d1dccf7b98". The service is identified as "Inference Server".

Key details shown include:

- Inference Endpoint URL:** http://localhost:61594/v1 (with a copy icon)
- Model:** TheBloke/Mistral-7B-Instruct-v0.2-GGUF (with a copy icon)
- File path:** /Users/jayapriyapai/.local/share/containers/podman-desktop/extensions-storage/redhat.ai-lab/models/hf.TheBloke.mistral-7b-instruct-v0.2.Q4_K_M
- HTTP Headers:** Apache-2.0, Hugging Face

Below this, under "Client code", there are two cURL examples:

```
curl --location 'http://localhost:61594/v1/chat/completions' \
--header 'Content-Type: application/json' \
--data '{
  "messages": [
    {
      "content": "You are a helpful assistant.",
      "role": "system"
    },
    {
      "content": "What is the capital of France?",
      "role": "user"
    }
  ]
}'
```

Integrating the AI model with a local application

Example application config file pointed to inference server url

```
model:  
  name: gpt-3.5-turbo  
  # update with the model service url before running the application  
  inference_endpoint_url: http://localhost:61594/v1  
  api_key: key-not-required
```

```
chatbot.py > ...  
You, 18 hours ago | 1 author (You)  
1 import os  
2  
3 import utils  
4  
5 import streamlit as st  
6 from openai import OpenAI  
7  
8 st.title("Simple AI ChatBot")  
9 config = utils.parseConfig()  
10 api_key=os.environ.get("API_KEY", "key-not-required")  
11  
12 # using local running model  
13 # TODO: Use env variable  
14 client = OpenAI(  
15     base_url= config['model']['inference_endpoint_url'],  
16     api_key=api_key,  
17 )  
18  
19 if "llm_model" not in st.session_state:  
20     st.session_state["llm_model"] = config['model']['name']  
21  
22 if "messages" not in st.session_state: You, 18 hours ago * feat: create a simple ai chat app -  
23     st.session_state.messages = []  
24  
25 for message in st.session_state.messages:  
26     with st.chat_message(message["role"]):  
27         st.markdown(message["content"])  
28  
29 if prompt := st.chat_input("Hello 🌟, how can I help you?"): You, 18 hours ago * feat: create a simple ai chat app -  
30     st.session_state.messages.append({"role": "user", "content": prompt})  
31     with st.chat_message("user"):  
32         st.markdown(prompt)  
33  
34     with st.chat_message("assistant"):  
35         stream = client.chat.completions.create(  
36             model=st.session_state["llm_model"],  
37             messages=[  
38                 {"role": m["role"], "content": m["content"]}  
39                 for m in st.session_state.messages  
40             ],  
41             stream=True,  
42         )  
43         response = st.write_stream(stream)  
44     st.session_state.messages.append({"role": "assistant", "content": response})  
45
```

Simple AI ChatBot

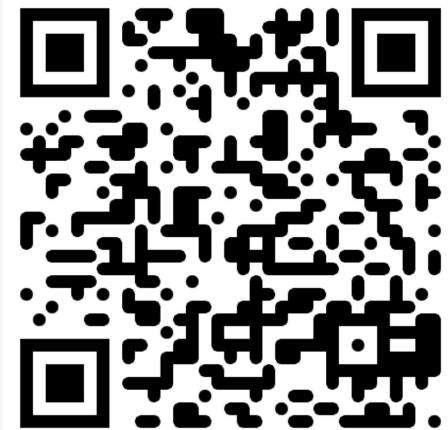


What is Kubernetes?



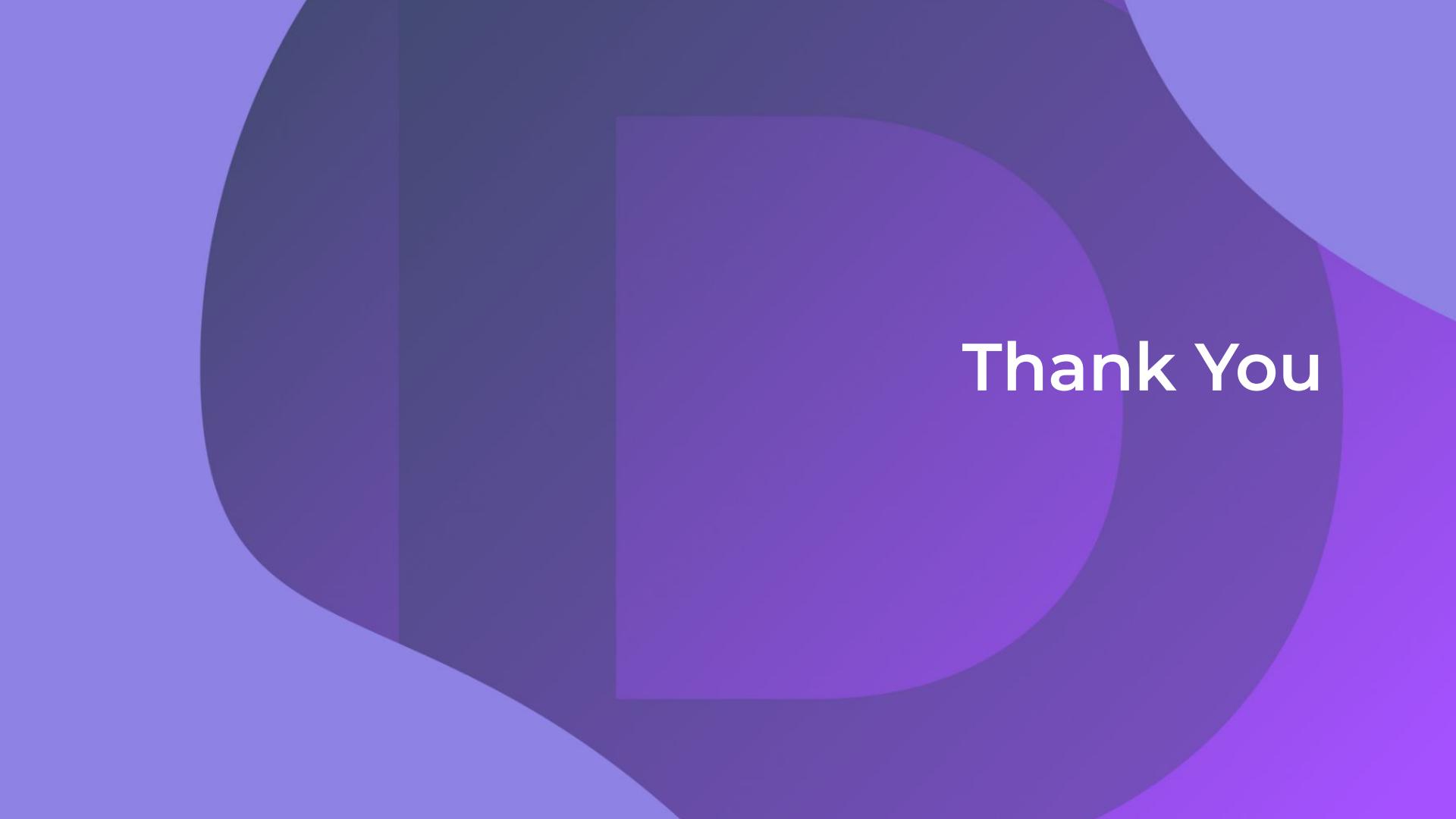
Kubernetes is an open-source container orchestration system for automating the deployment, scaling, and management of containerized applications. It was originally designed by Google and is now maintained by the Cloud Native Computing Foundation (CNCF). Kubernetes provides a platform that enables you to run, manage, and deploy containers in a distributed environment at scale. With Kubernetes, you can automate tasks such as rolling updates and rollbacks, self-healing restarts of containers, load balancing, and resource allocation. It also offers features for service discovery, secret and configuration management, and monitoring and logging. Overall, Kubernetes aims to simplify the complexity of managing containerized applications in a production environment, making it easier to develop, deploy, and run applications at scale with high availability and resilience.

Hello 🙌, how can I help you? >



References and Credits

- <https://podman.io>
- <https://podman-desktop.io/docs/intro>
- <https://podman-desktop.io/docs/ai-lab>
- <https://github.com/containers/ai-lab-recipes>
- <https://podman-desktop.io/blog/podman-quadlet>
- <https://developers.redhat.com/products/podman-desktop>
- <https://minikube.sigs.k8s.io/docs/handbook/pushing/#3-pushing-directly-to-in-cluster-cri-o-podman-env>

The background features a minimalist design with three overlapping circles in shades of purple. A large circle in the center-right overlaps a smaller square in the middle. A third, larger circle is partially visible behind the others.

Thank You