Dockering

Friday, 4 July 2025 11:00 PM

Eliminates Environment Issues: Docker solves the "works on my machine" problem by ensuring containers run consistently across different systems, whether Windows, Mac, or Linux. This enables seamless cross-platform development and deployment.

Enhanced Security and Isolation: The MCP server runs in its own isolated sandbox, preventing it from accidentally affecting the host system or conflicting with other services. Docker allows precise control over what resources the container can access through volumes and device flags.

Simplified Scaling and Management: Multiple server instances can be easily launched by spinning up more containers. Updates are straightforward - just build a new image version and deploy it. The approach integrates well with orchestration tools like Docker Compose or Kubernetes for larger projects.

The main takeaway is that Docker containerization provides a portable, secure, and scalable solution for deploying MCP servers, offering significant advantages in terms of reliability, security, and operational flexibility.

Division of Responsibilities: In most AI engineering roles, you typically won't need to write Dockerfiles yourself. Instead, DevOps teams and engineers handle the containerization, deployment, and infrastructure management.

Deployment Pipeline: The DevOps team takes your application and packages it into Docker containers, then deploys these containers to cloud platforms using orchestration tools like Kubernetes (EKS, GKE, AKS) or serverless solutions.

Hands-on Learning: Despite this typical workflow, the author emphasizes the value of understanding the process by actually building a Dockerfile and creating a container for the MCP server as a learning exercise.

The key message is that while you may not do this in practice as an AI engineer, understanding how Docker containerization works is valuable knowledge for better collaboration with DevOps teams and overall system comprehension.

Create a docker file using this prompt:

> write a docker file to run this in a container, I am using uv checkout this doc: @https://docs.astral.sh/uv/guides/integration/docker/#installing-uv I am running uv using the command uv run server.py inside a virtual environment
>
> Dockerfile get created

Run this docker file using this command:

- Which docker
- brew install colima docker # if not installed
- colima start
- docker build -t shellserver-app .

Few docker basic commands:

```
# Start a new container with interactive shell
docker run -it <image_name> /bin/bash

# Execute shell in running container
docker exec -it <container_name_or_id> /bin/bash

# Use sh if bash isn't available
docker exec -it <container_name_or_id> /bin/sh

# Start a stopped container
docker start <container_name_or_id>
```

```
# Stop a running container
docker stop <container_name_or_id>

# Restart a container
docker restart <container_name_or_id>

# Force stop (kill) a container
docker kill <container_name_or_id>

# List running containers
docker ps

# List all containers (including stopped)
docker ps -a

# List Docker images
docker images

# Show container logs
docker logs <container_name_or_id>

# Follow logs in real-time
docker logs -f <container_name_or_id>
# Remove a container
docker rm <container_name_or_id>

# Remove a Docker image
docker rmi <image_name>

# Remove all stopped containers
docker container prune

# Remove unused images
docker image prune
```