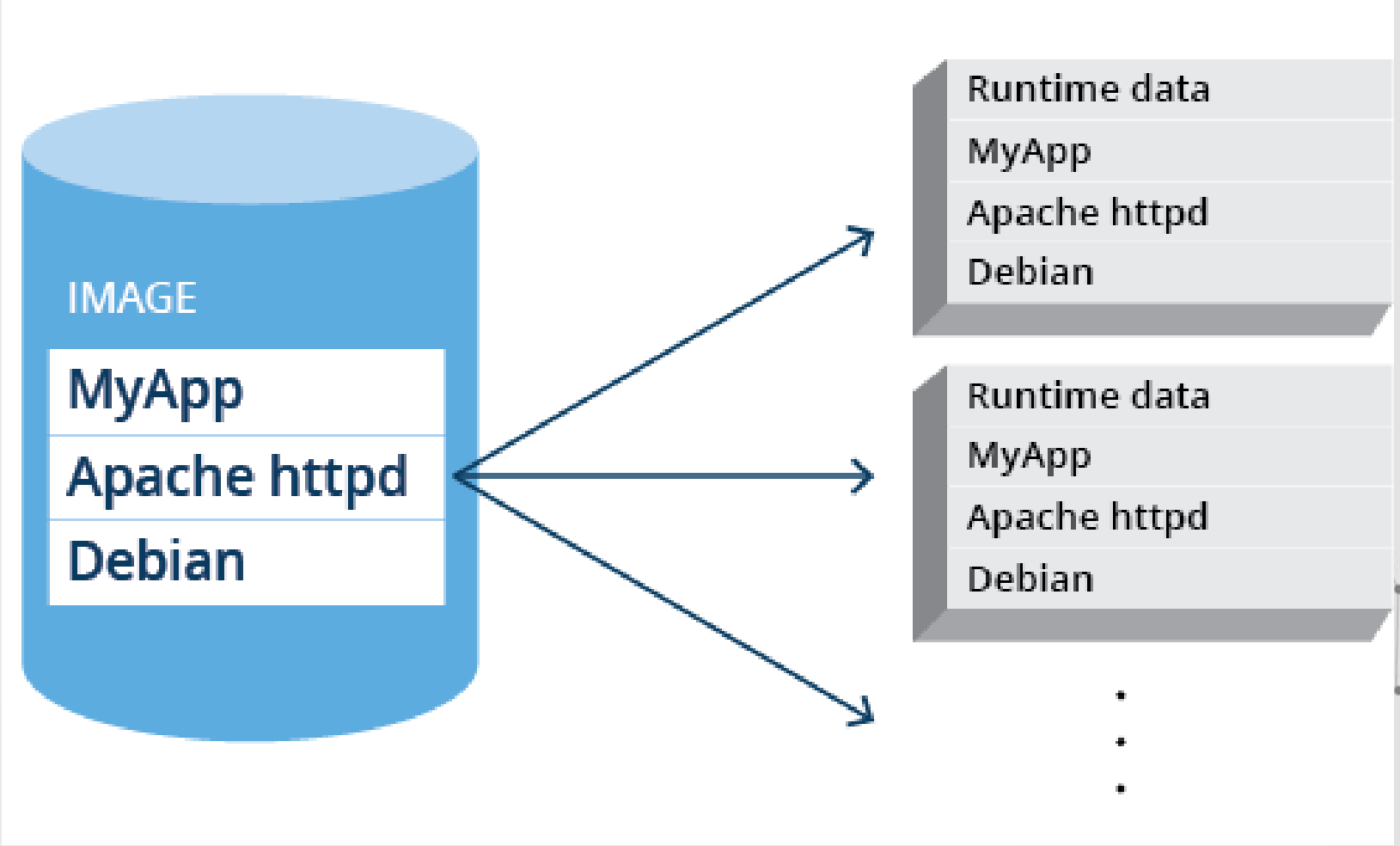
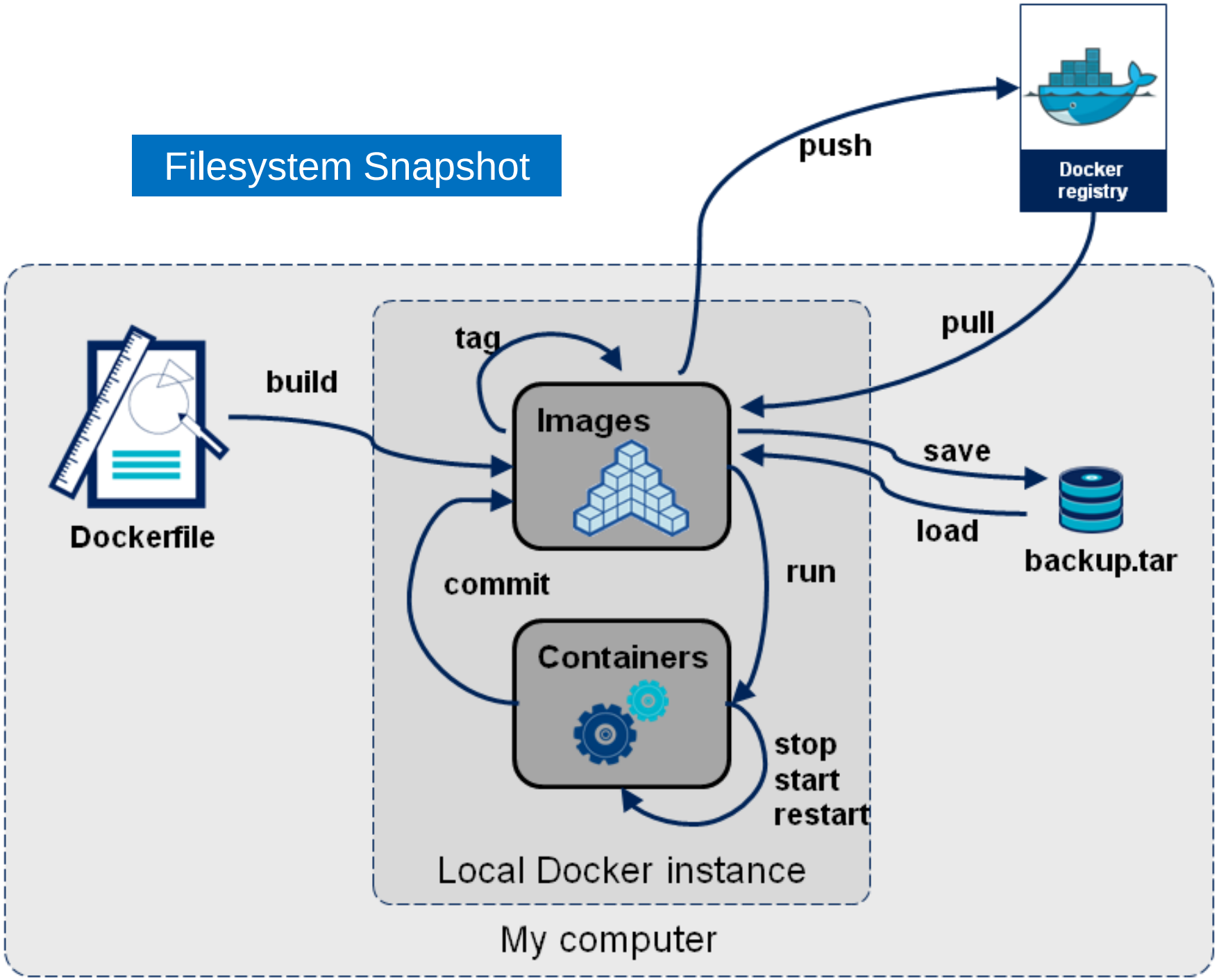
A Docker image is a lightweight, standalone, executable package that contains everything needed to run a piece of software, including the code, runtime, libraries, dependencies, and system tools.

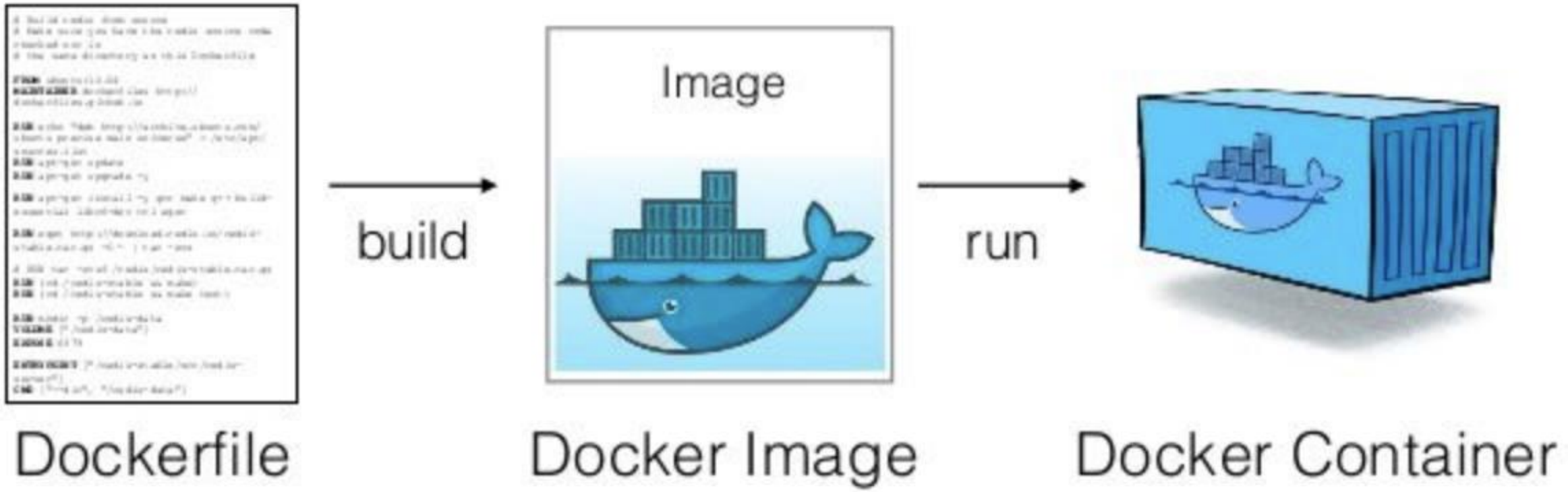


Filesystem Snapshot



The Dockerfile specifies the instructions for building the image, including the base image to use, commands to install dependencies, copy application code, and define runtime settings.

Docker images are designed to be run inside Docker containers. Containers are lightweight, isolated instances created from Docker images.



Docker images can be tagged with a name and version (e.g., my-app:v1).

Building the Image

```
docker build -t my-app-image:v1 .
```

Tagging the Image

```
docker tag my-app-image:v1 us-central1-docker.pkg.dev/ambient-tuner-373213/gke-repo/my-app-image:v1
```

Configure the Docker command-line tool to authenticate to Artifact Registry

```
gcloud auth configure-docker us-central1-docker.pkg.dev
```

Push the Docker image to the Artifact repository

```
docker push us-central1-docker.pkg.dev/ambient-tuner-373213/gke-repo/my-app-image:v1
```

Connecting to the cluster

```
gcloud container clusters get-credentials cluster1 --region us-central1 --project ambient-tuner-373213
```

Creating a Deployment

```
kubectl create deployment my-app-deployment --image=us-central1-docker.pkg.dev/ambient-tuner-373213/gke-repo/my-app-image:v1
```

Exposing the Deployment

```
kubectl expose deployment my-app-deployment --name=hello-app-service --type=LoadBalancer --port 80 --target-port 5000
```

Listing the Services

```
kubectl get svc
kubectl get services
```