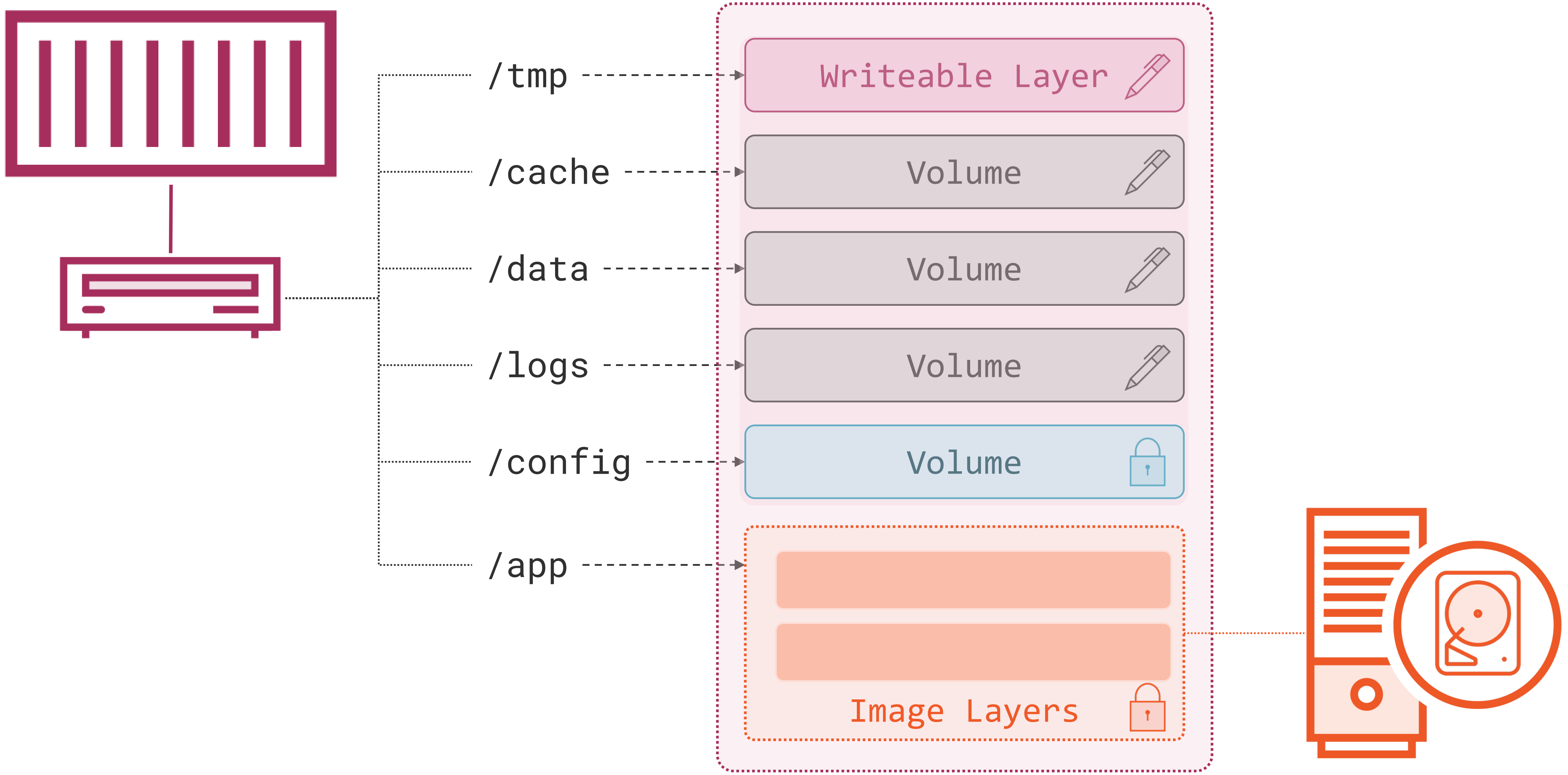
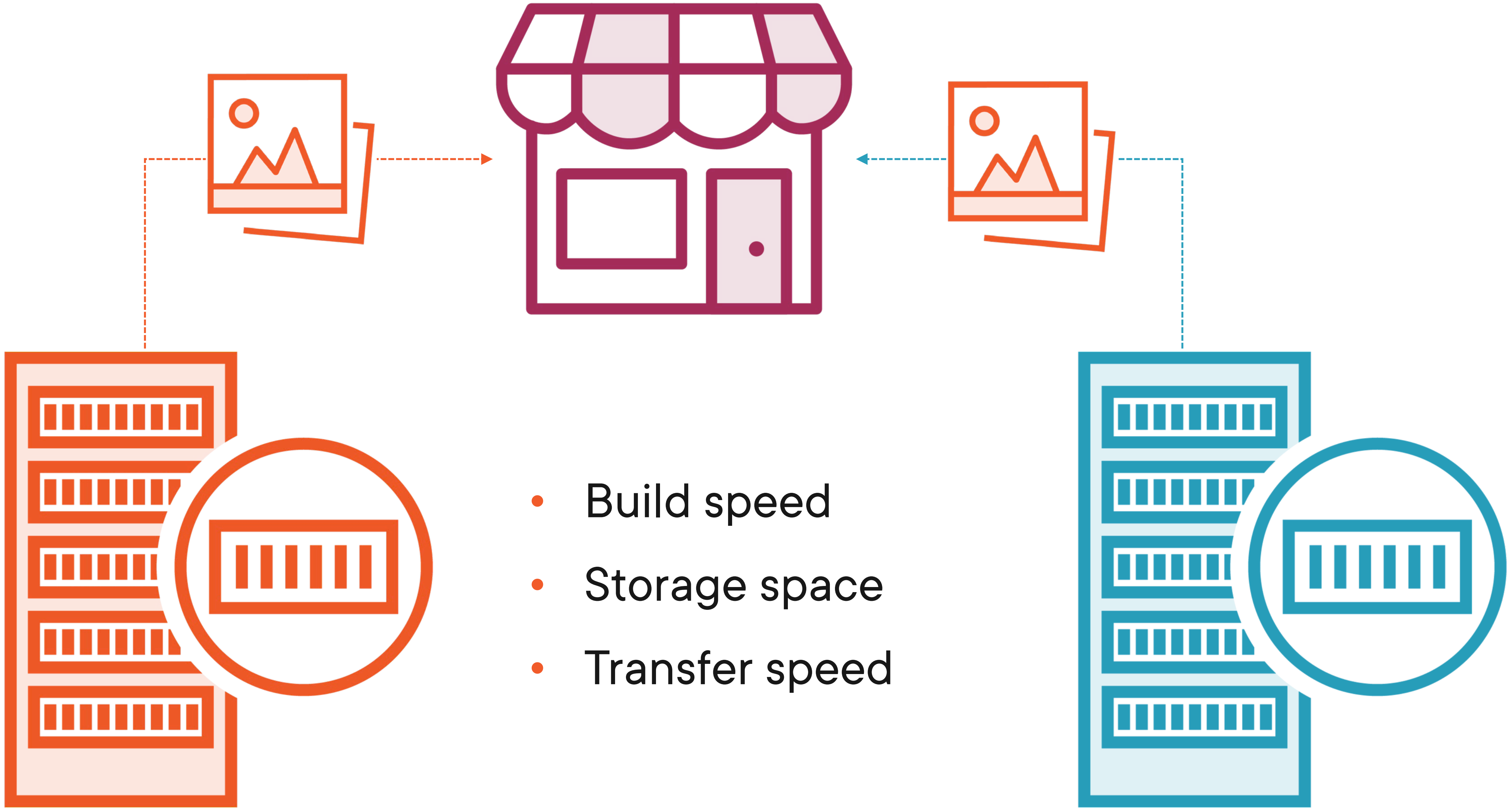
Managing Storage on Servers and Registries

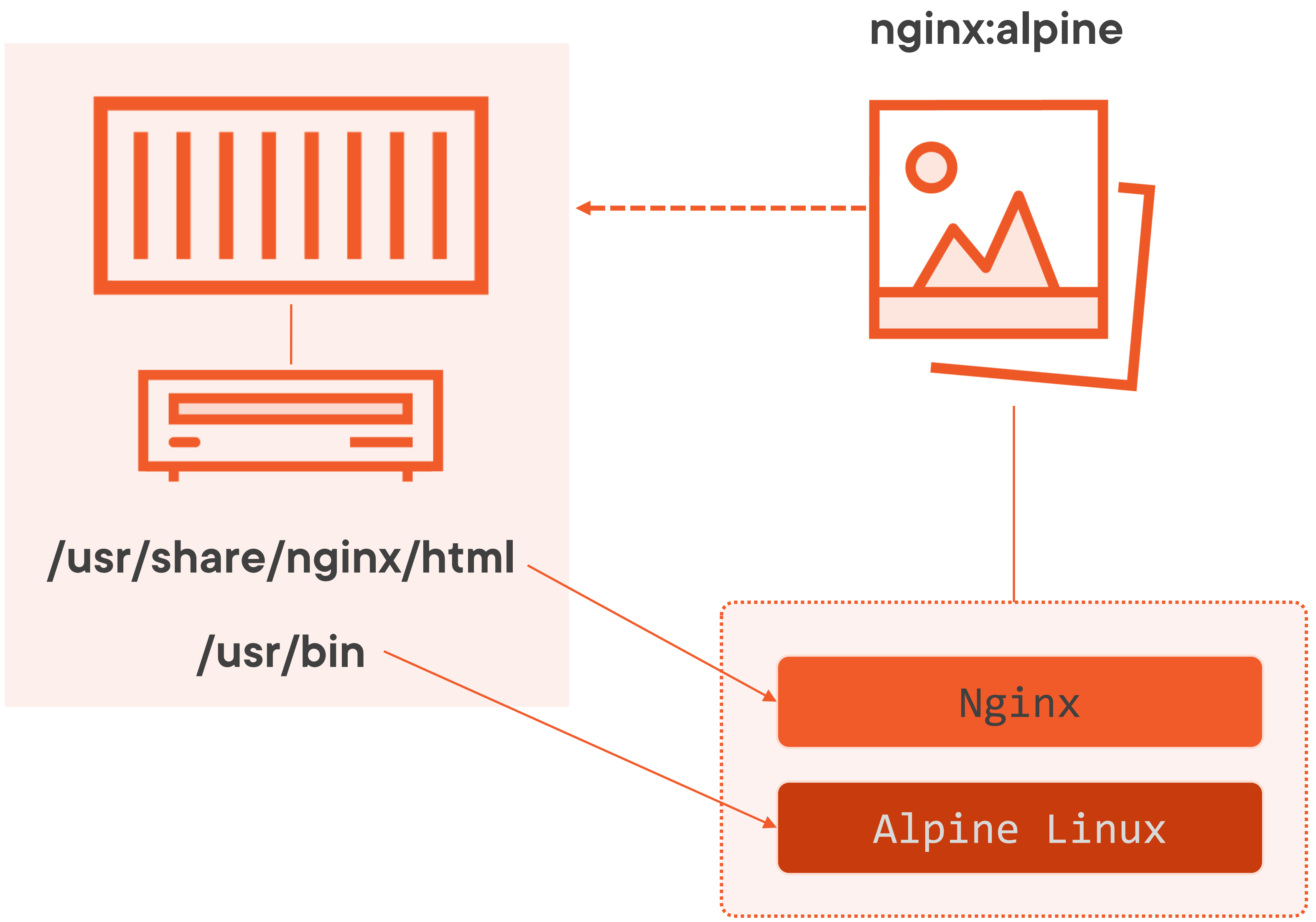
Elton Stoneman
Consultant & Trainer

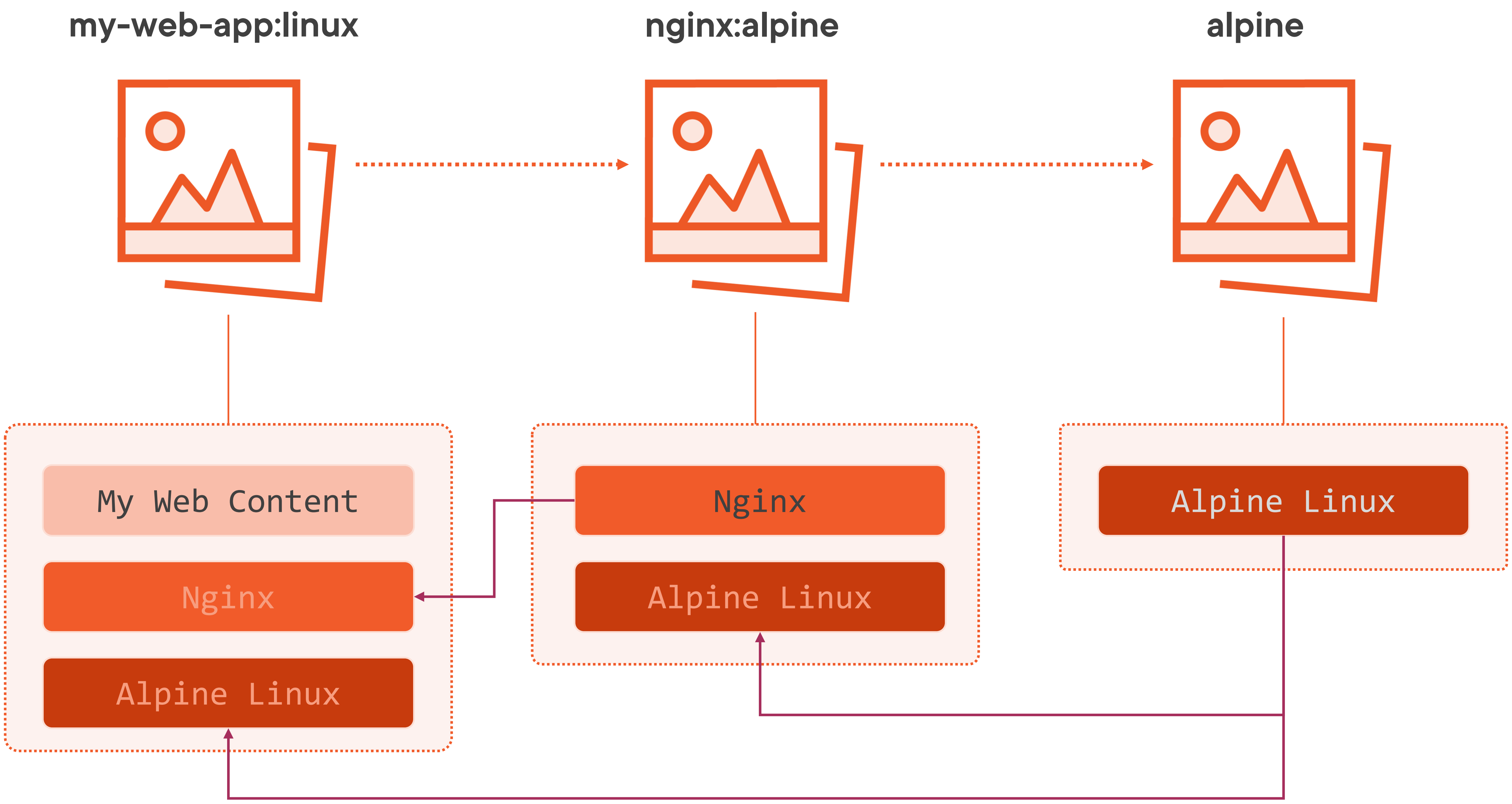
@EltonStoneman blog.sixeyed.com


/tmp
/cache
/data
/logs
/config
/app
Writeable Layer
Volume
Volume
Volume
Volume
Image Layers

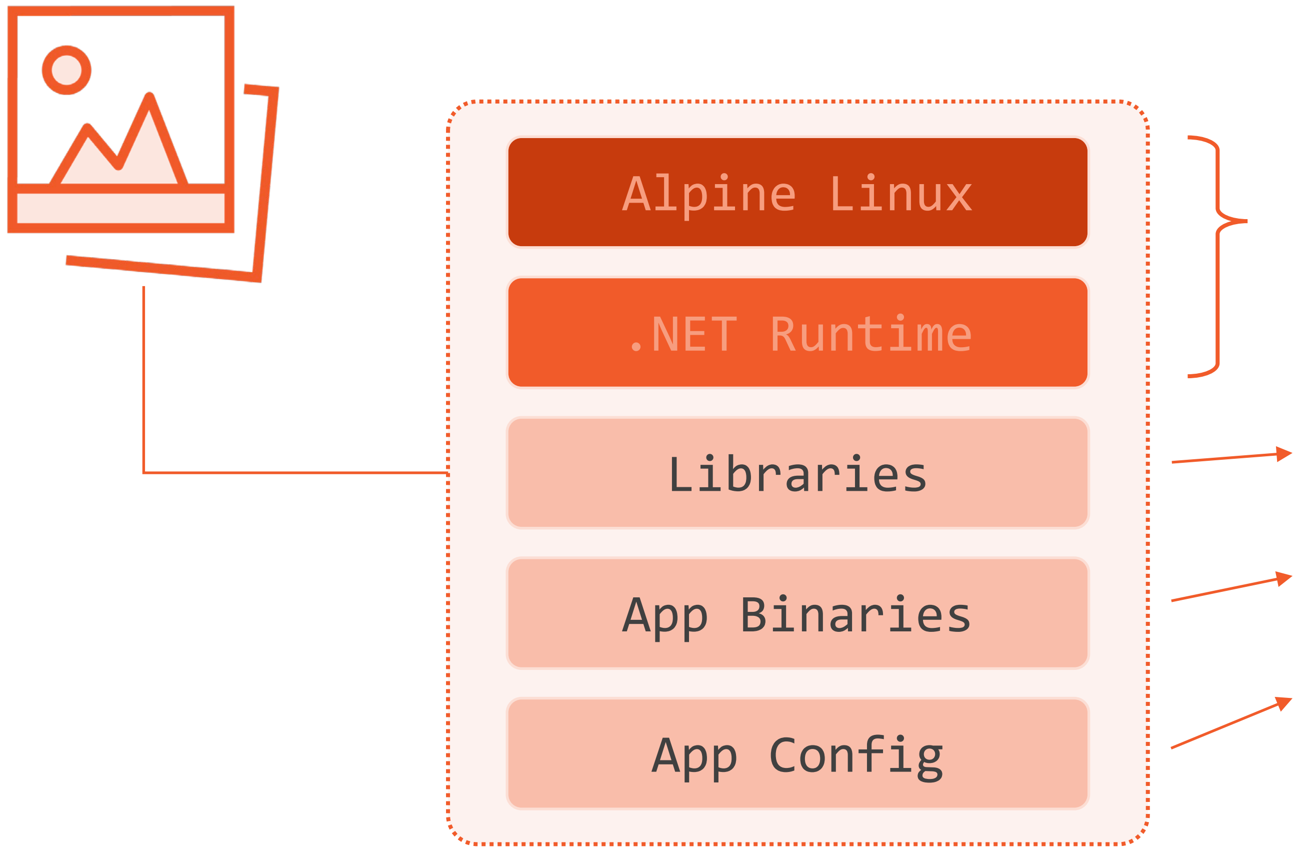


- Build speed
- Storage space
- Transfer speed


nginx:alpine
/usr/share/nginx/html
/usr/bin
Nginx
Alpine Linux


my-web-app:linux
nginx:alpine
alpine
My Web Content
Nginx
Alpine Linux
Nginx
Alpine Linux
Alpine Linux

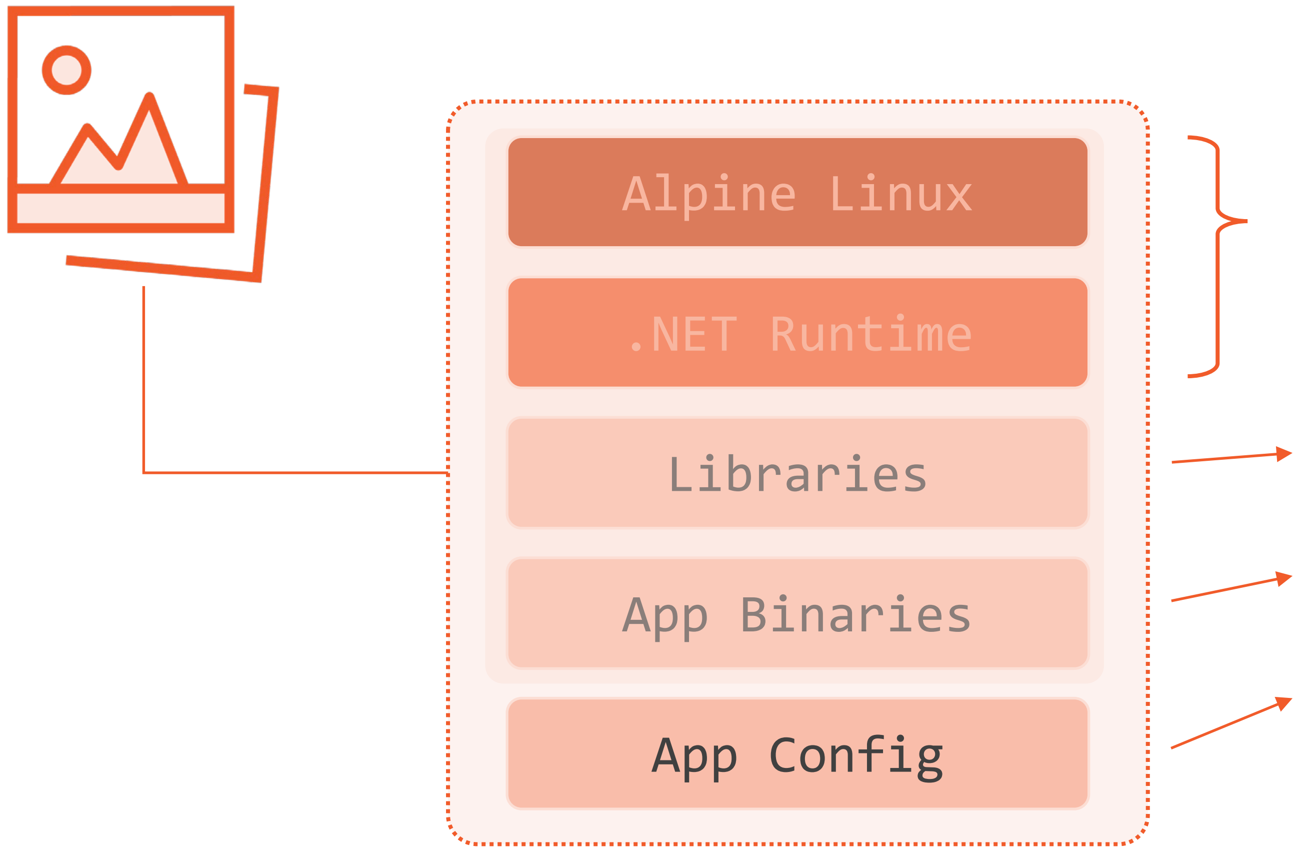
widgetario/web:22.05



Dockerfile

```
FROM dotnet/aspnet:6.0-alpine

WORKDIR /app

COPY /lib/ .

COPY /app/ .

COPY /conf/ .
```

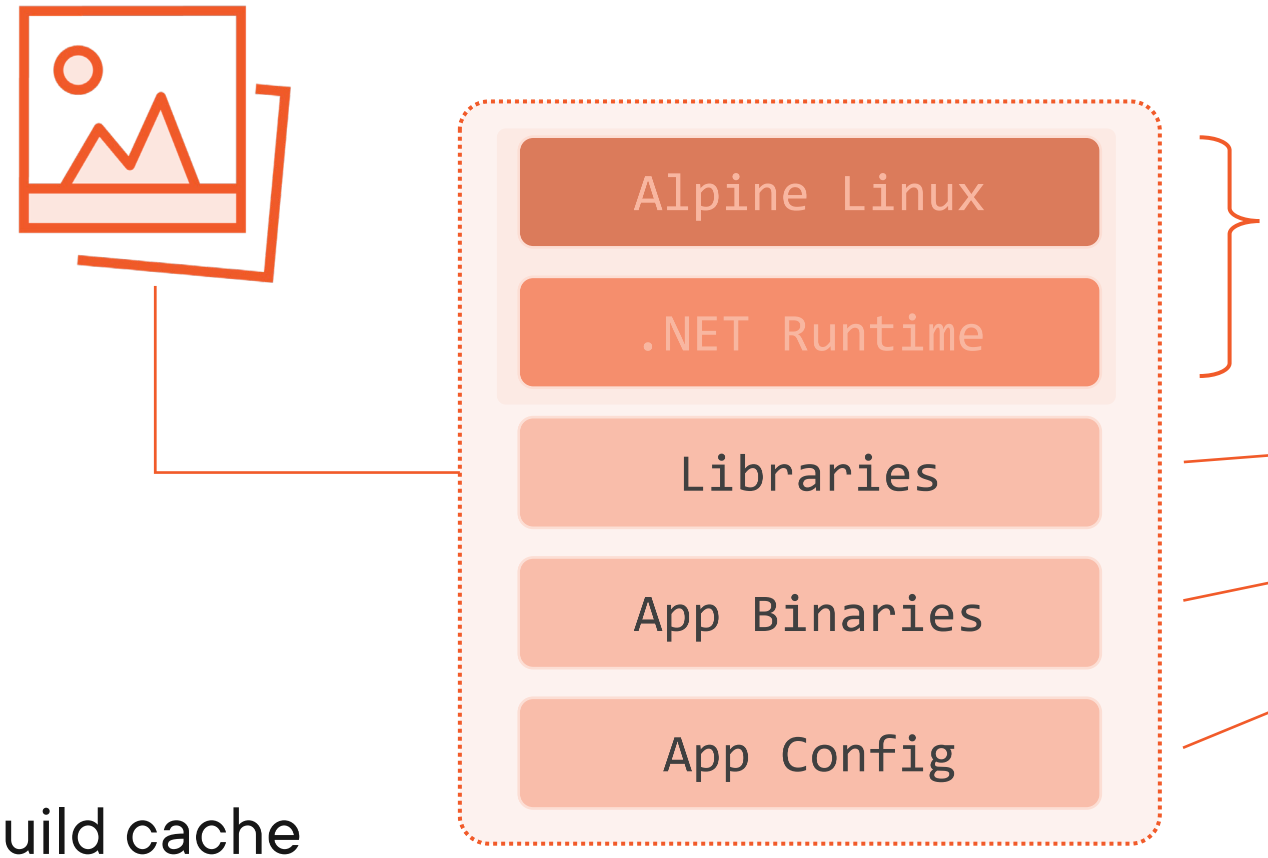
widgetario/web:22.05



Dockerfile

```
FROM dotnet/aspnet:6.0-alpine

WORKDIR /app

COPY /lib/ .

COPY /app/ .

COPY /conf/ .
```

widgetario/web:22.05



Dockerfile

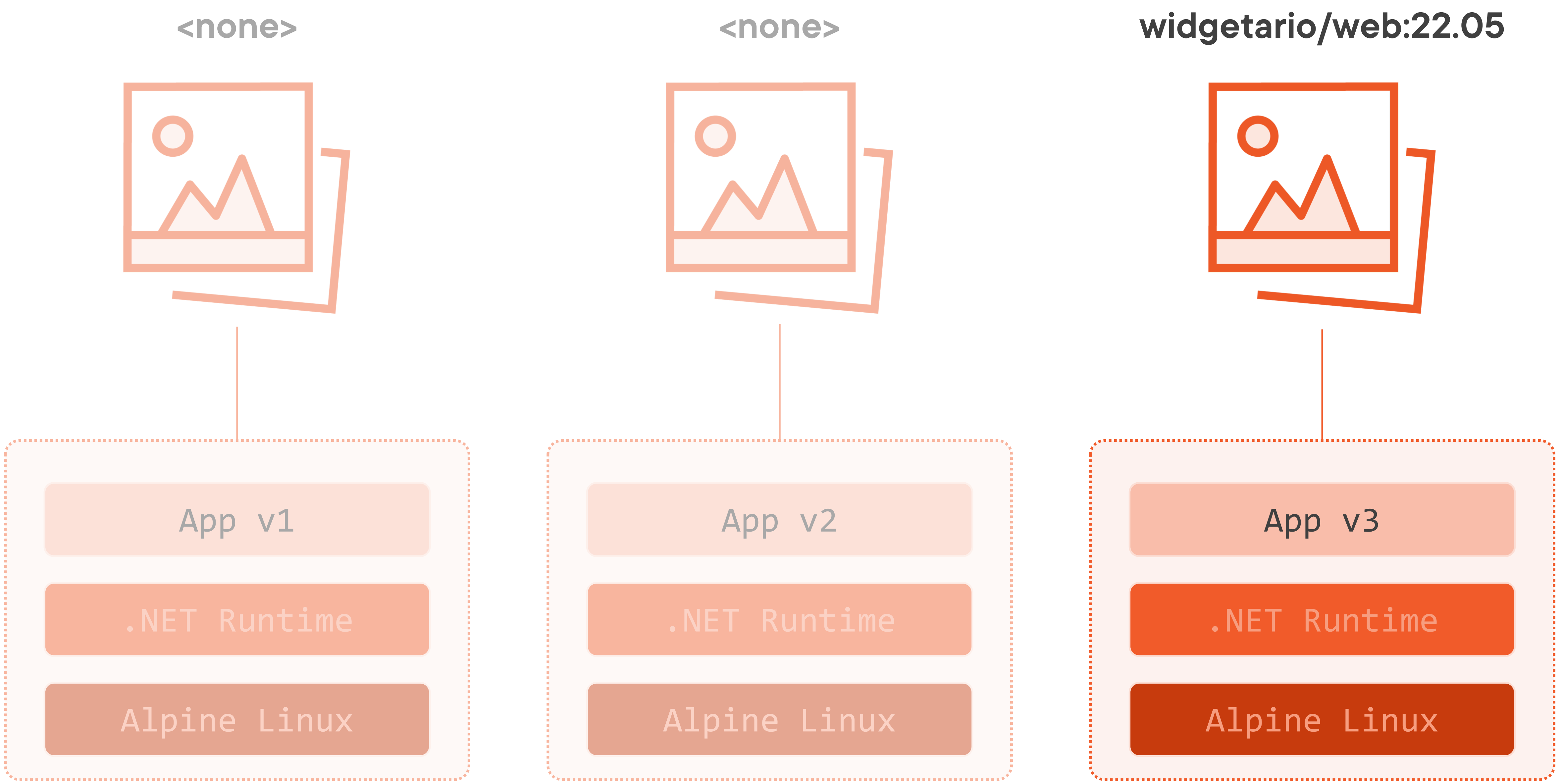
```
FROM dotnet/aspnet:6.0-alpine

WORKDIR /app

COPY /lib/ .

COPY /app/ .

COPY /conf/ .
```

- Build cache
- Size and speed
- Leftover image layers

Demo

Optimizing Storage in Docker Builds

- Understanding dangling images
- Optimizing builds for speed
- Optimizing builds for storage



<none>
<none>
widgetario/web:22.05
App v1
.NET Runtime
Alpine Linux
App v2
.NET Runtime
Alpine Linux
App v3
.NET Runtime
Alpine Linux

```
docker image prune

docker builder prune
```

Pruning image builds

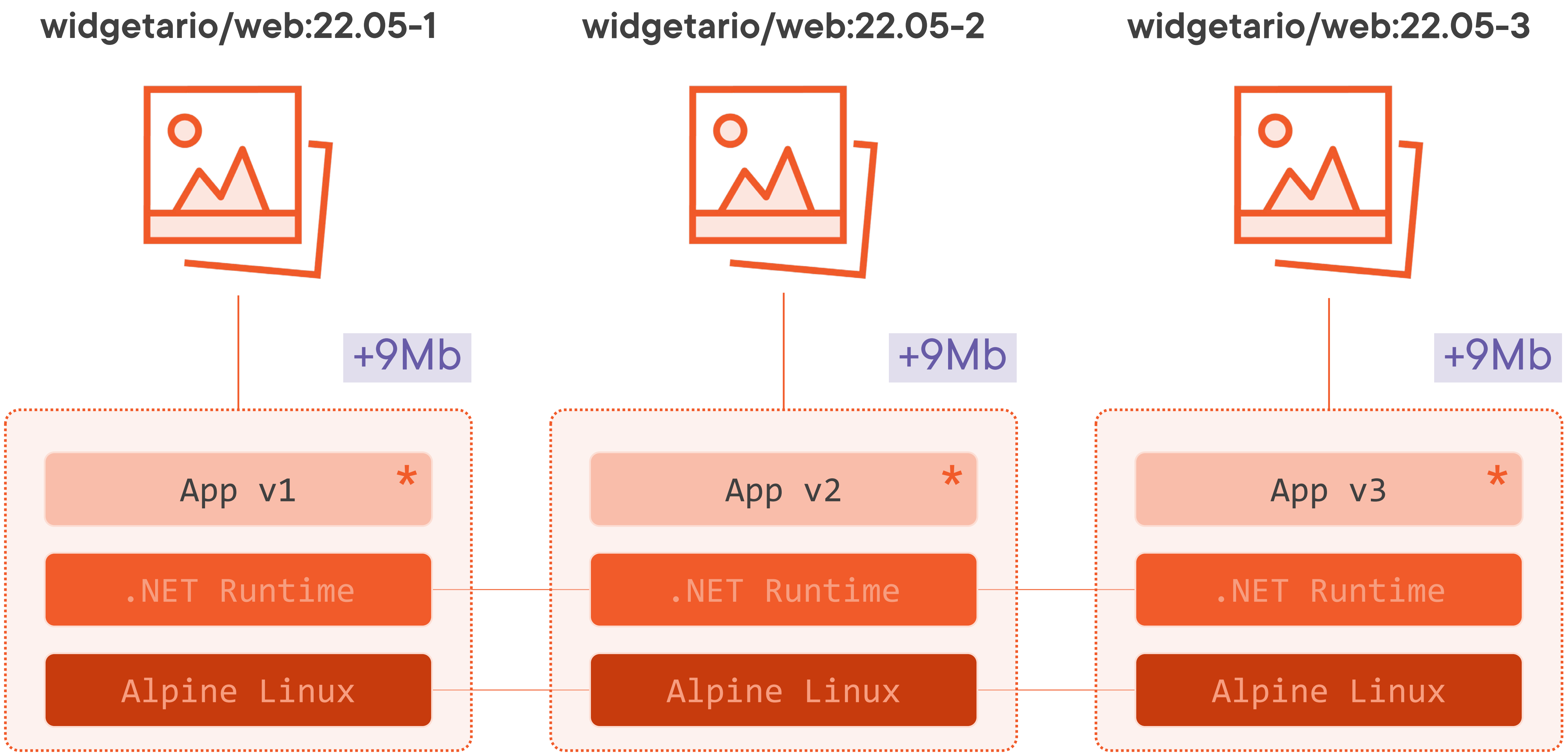
Removes unused layers & clears cache

Multi-stage Dockerfiles

```
FROM mcr.microsoft.com/dotnet/sdk:6.0-alpine AS builder

WORKDIR /src
COPY src/WiredBrain.Web/WiredBrain.Web.csproj .
RUN dotnet restore

COPY src/WiredBrain.Web/ .
RUN dotnet publish --no-restore -c Release -o /out WiredBrain.Web.csproj

# app image
FROM mcr.microsoft.com/dotnet/aspnet:6.0-alpine

ENTRYPOINT ["dotnet", "/app/WiredBrain.Web.dll"]

WORKDIR /app
COPY --from=builder /out/ .
```

- Build cache
- Reuse dependencies

- Minimal app image
- Runtime + binaries


widgetario/web:22.05-1
widgetario/web:22.05-2
widgetario/web:22.05-3
+9Mb
+9Mb
+9Mb
App v1 *
.NET Runtime
Alpine Linux
App v2 *
.NET Runtime
Alpine Linux
App v3 *
.NET Runtime
Alpine Linux

Optimized Dockerfiles - Build Stage

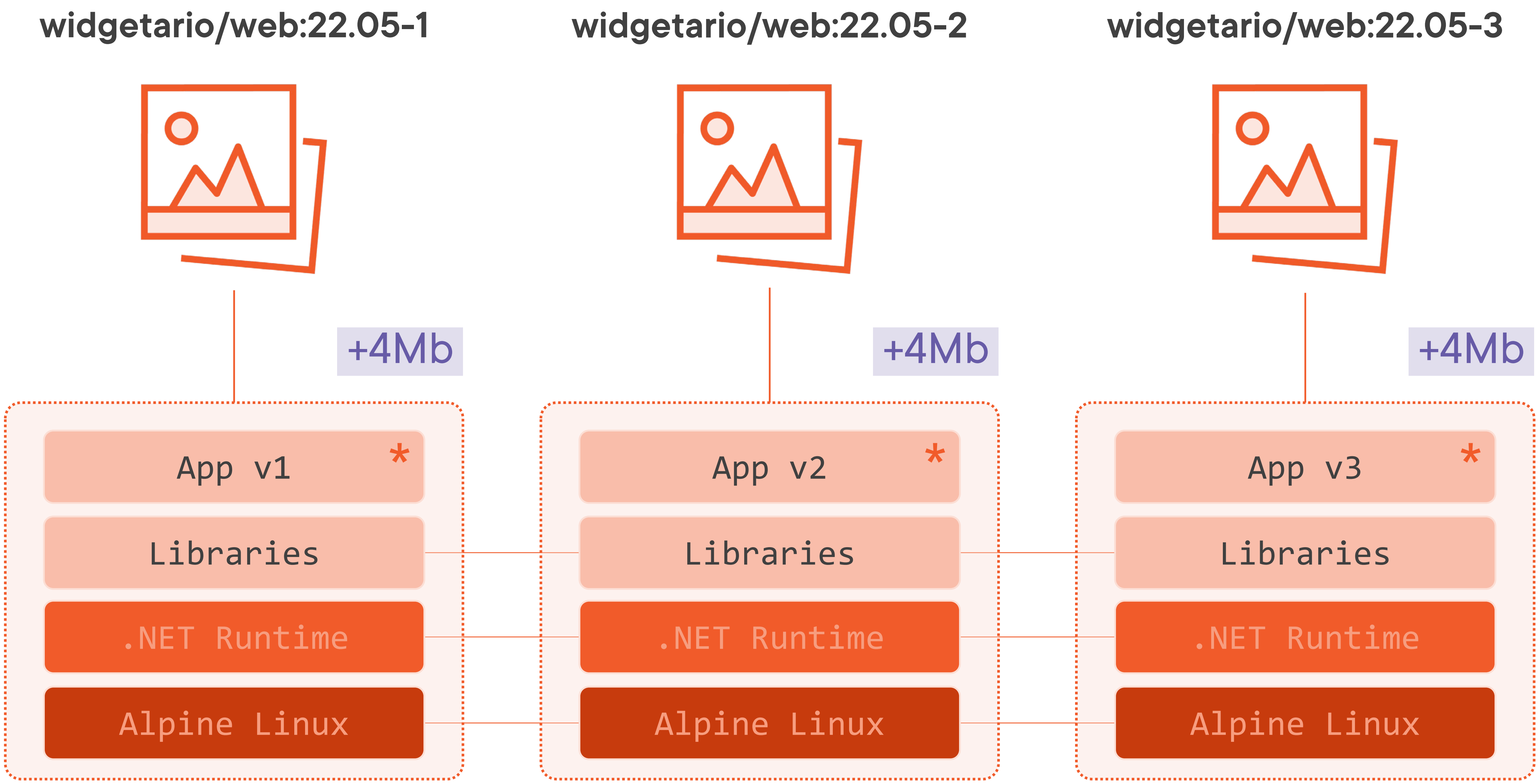
```
FROM mcr.microsoft.com/dotnet/sdk:6.0-alpine AS builder

WORKDIR /src
COPY src/WiredBrain.Web/WiredBrain.Web.csproj .
RUN dotnet restore

COPY src/WiredBrain.Web/ .
RUN dotnet publish --no-restore -c Release -o /out WiredBrain.Web.csproj

RUN mkdir -p /out/wiredbrain-out && \
    mv /out/wwwroot /out/wiredbrain-out/ && \
    mv /out/WiredBrain.* /out/wiredbrain-out/ && \
    mv /out/appsettings*.json /out/wiredbrain-out/
```

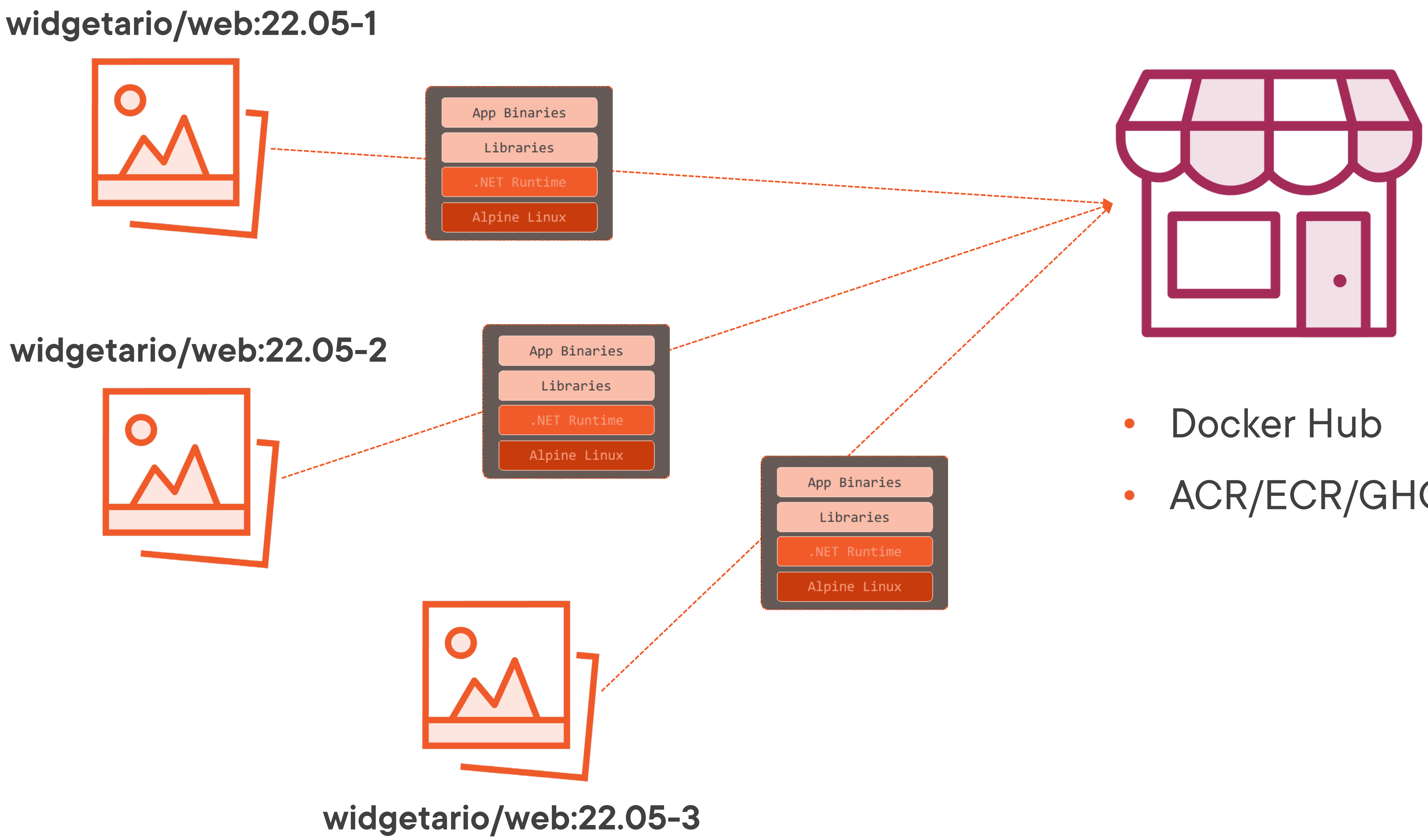
- Split output directory
- Separate libraries

Optimized Dockerfiles - App Stage

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0-alpine

ENV ProductsApi:Url="http://products-api/products" \
    StockApi:Url="http://stock-api/stock"

ENTRYPOINT ["dotnet", "/app/WiredBrain.Web.dll"]

WORKDIR /app

COPY --from=builder /out/runtimes/ ./runtimes/
COPY --from=builder /out/*.dll ./
COPY --from=builder /out/wiredbrain-out/ ./
```

- Separate layers
- Cache libraries & statics


widgetario/web:22.05-1
widgetario/web:22.05-2
widgetario/web:22.05-3
+4Mb
+4Mb
+4Mb
App v1 *
Libraries
.NET Runtime
Alpine Linux
App v2 *
Libraries
.NET Runtime
Alpine Linux
App v3 *
Libraries
.NET Runtime
Alpine Linux

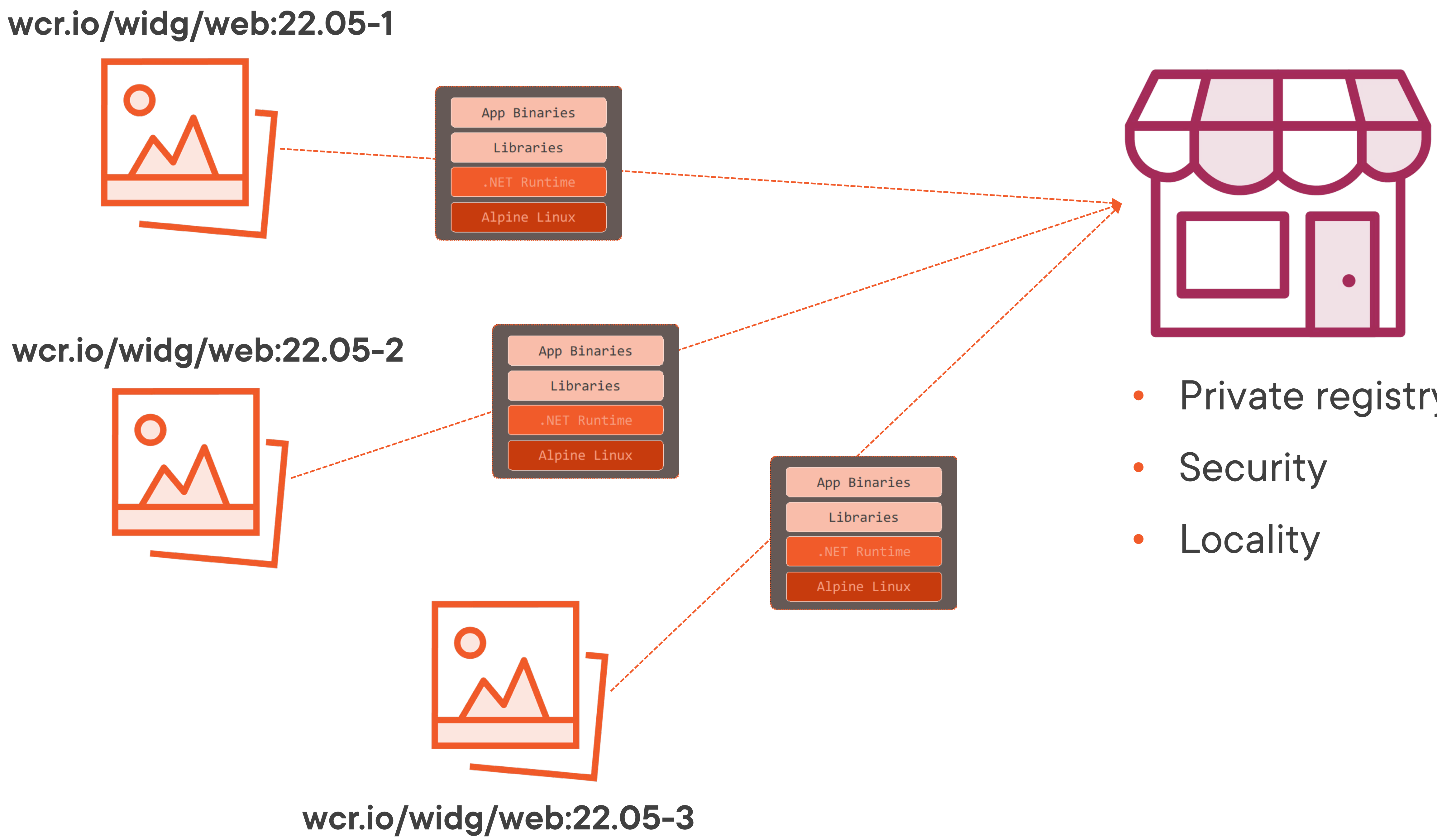


- Docker Hub
- ACR/ECR/GHCR

Demo

Managing Container Registries

- Tagging and pushing images
- Inspecting remote images
- Scripting registry cleanup



- Private registry
- Security
- Locality

```
az acr login -n wiredbrain

docker image push --all-tags
   wiredbrain.azurecr.io/wiredbrain/web
```

Private registries

Docker tooling + platform security

Docker Compose Variables

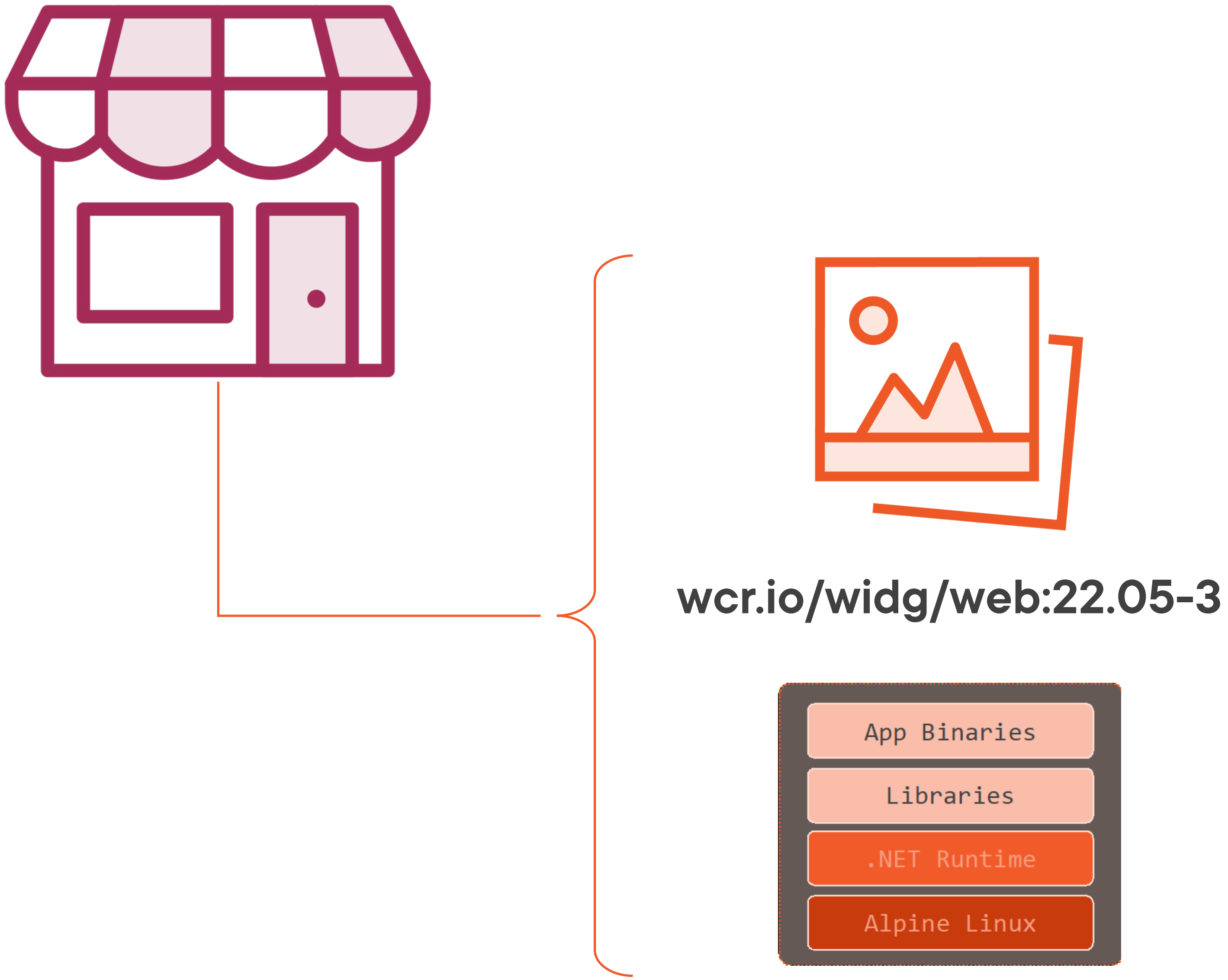
- Environment variables
- Default values
- Local & server build

```yaml
version: '3'

services:
  web:
    image: ${REGISTRY:-docker.io}/wiredbrain/web:22.05-m5-${BUILD_NUMBER}
    build:
      context: web
      dockerfile: Dockerfile.optimized
```

```
$env:REGISTRY='wiredbrain.azurecr.io'
$env:BUILD_NUMBER=$i

docker-compose build
docker-compose push
```

Portable build scripts

Devs use same tooling as CI/CD pipeline

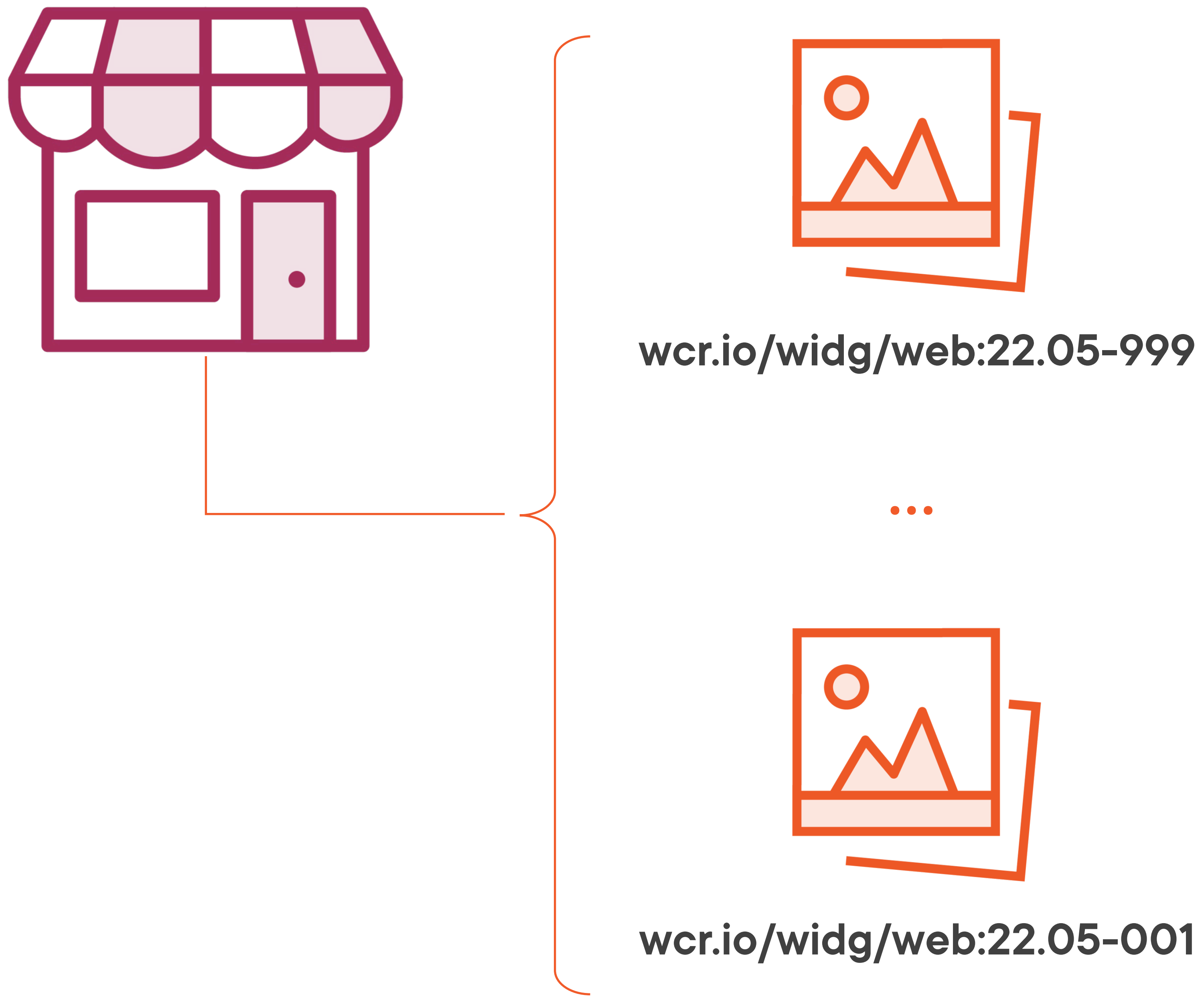


- OCI standard
- Push/pull
- Manifest inspect

Image Manifest

```
[
 {
  "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
  "size": 2716477,
  "digest": "sha256:9981e73032c8833e387a8f96986e560edbed12c38119e0edb0439c9c2234eac9"
 },
 {
  "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
  "size": 1731757,
  "digest": "sha256:e169fc8fe05db53b6d671795f8782eafdbb4f382e312c537d67fb7dedfb1d6bb"
 },
 {
  "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
  "size": 29306525,
  "digest": "sha256:2487dc352bcde5ac331cfa4aa559578781f9c67d13aeedefbfecfc15182e6019"
 },
 {
  "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
  "size": 9060018,
  "digest": "sha256:1c5dfda45fee571817f08f0f2b6b10401b5b7732dc6fc25d9722149a7d07e116"
 }
]
```

- Remote server
- Debug builds
- Check optimization

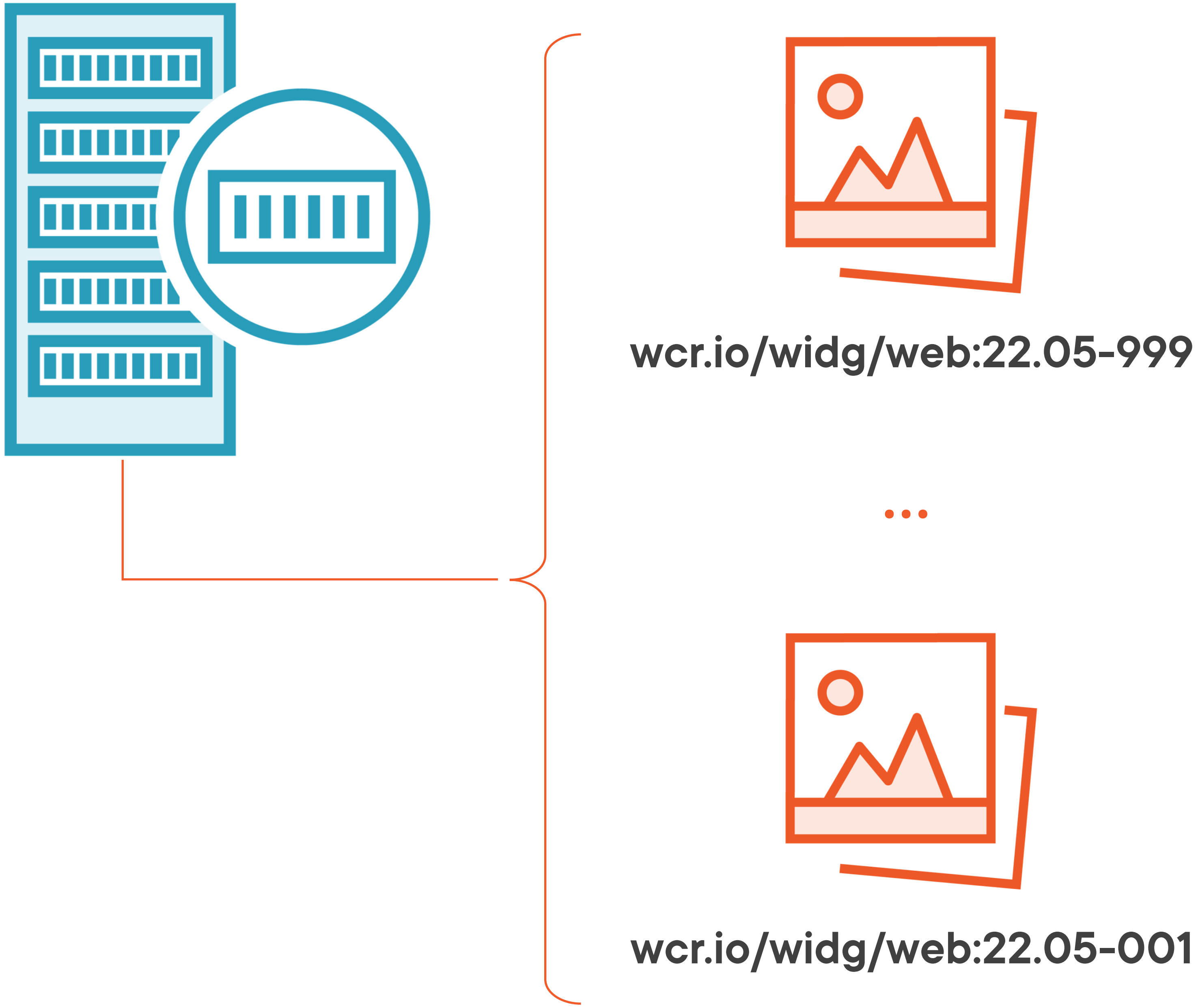


- Store all builds
- Storage charge
- Cleanup job

Pruning Registry Images

```
foreach (repository in get_repositories)

{
 tags = list_tags_descending_except 'latest','22.05'
 if (tags.length > max_image_tags)

  {

   for (i=max_image_tags; i < tags.length; i++)

   {

    image_name = "repository:tags[i]"

    delete_tag image_name

   }

  }

}
```

- Loop descending
- Ignore specific tags
- Delete oldest



- Disk is finite
- Dev & build servers
- Platform servers

Demo

Cleaning up image storage

- Pruning Docker servers
- Pruning Kubernetes nodes

```
docker image prune --all --force

docker builder prune -f

docker system prune -af --volumes
```

No garbage collection

Manual pruning of volumes, images, build cache

```
# if you really need to:
crictl rmi --prune
```

Automatic garbage collection

Images pruned at disk threshold (80%)

Image Pruning Job

```yaml
spec:
  restartPolicy: Never
  containers:
    - name: prune-images
      image: sixeyed/crictl:v1.24.1
      command: [ "sh", "-c", "crictl rmi --prune" ]
      volumeMounts:
        - name: containerd
          mountPath: /var/run/containerd/containerd.sock
  volumes:
    - name: containerd
      hostPath:
        path: /var/run/containerd/containerd.sock
        type: Socket
```

- CRI tooling
- containerd setup
- Dangerous...

Module Summary

Image optimization

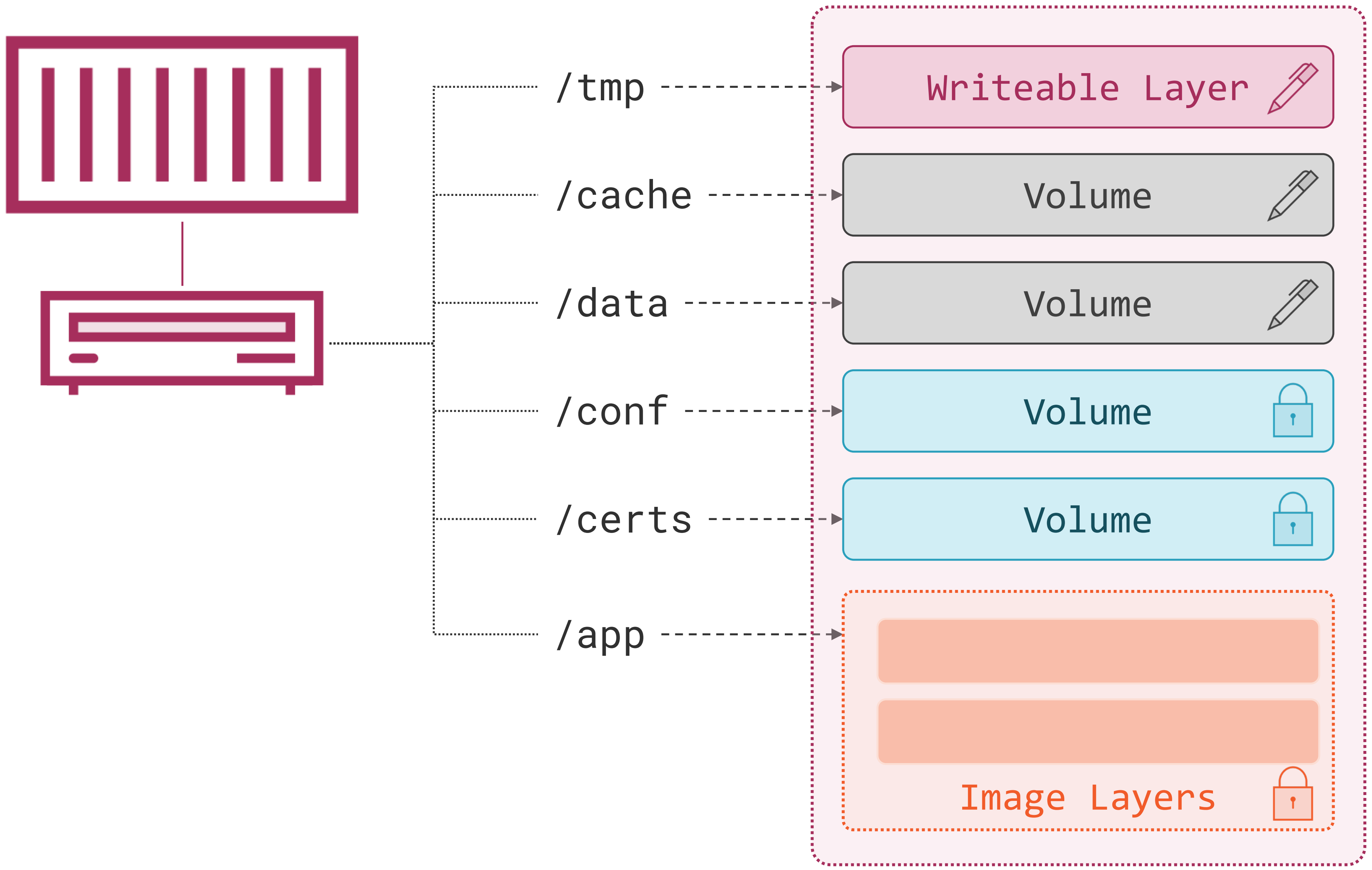
- Build speed
- Layer reuse
- Maximize the cache

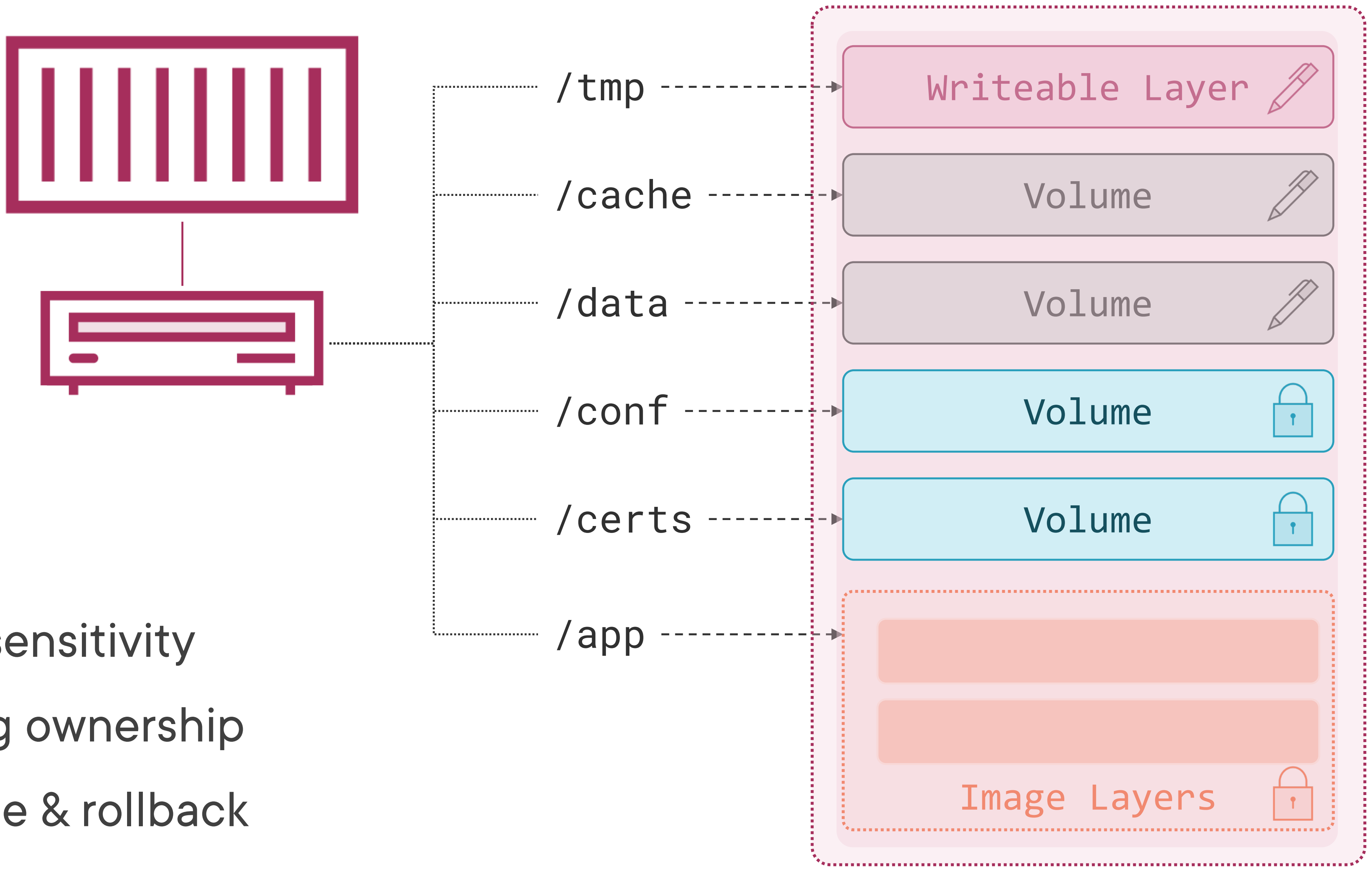
Registry management

- Private registry servers
- Network optimization
- Storage cleanup

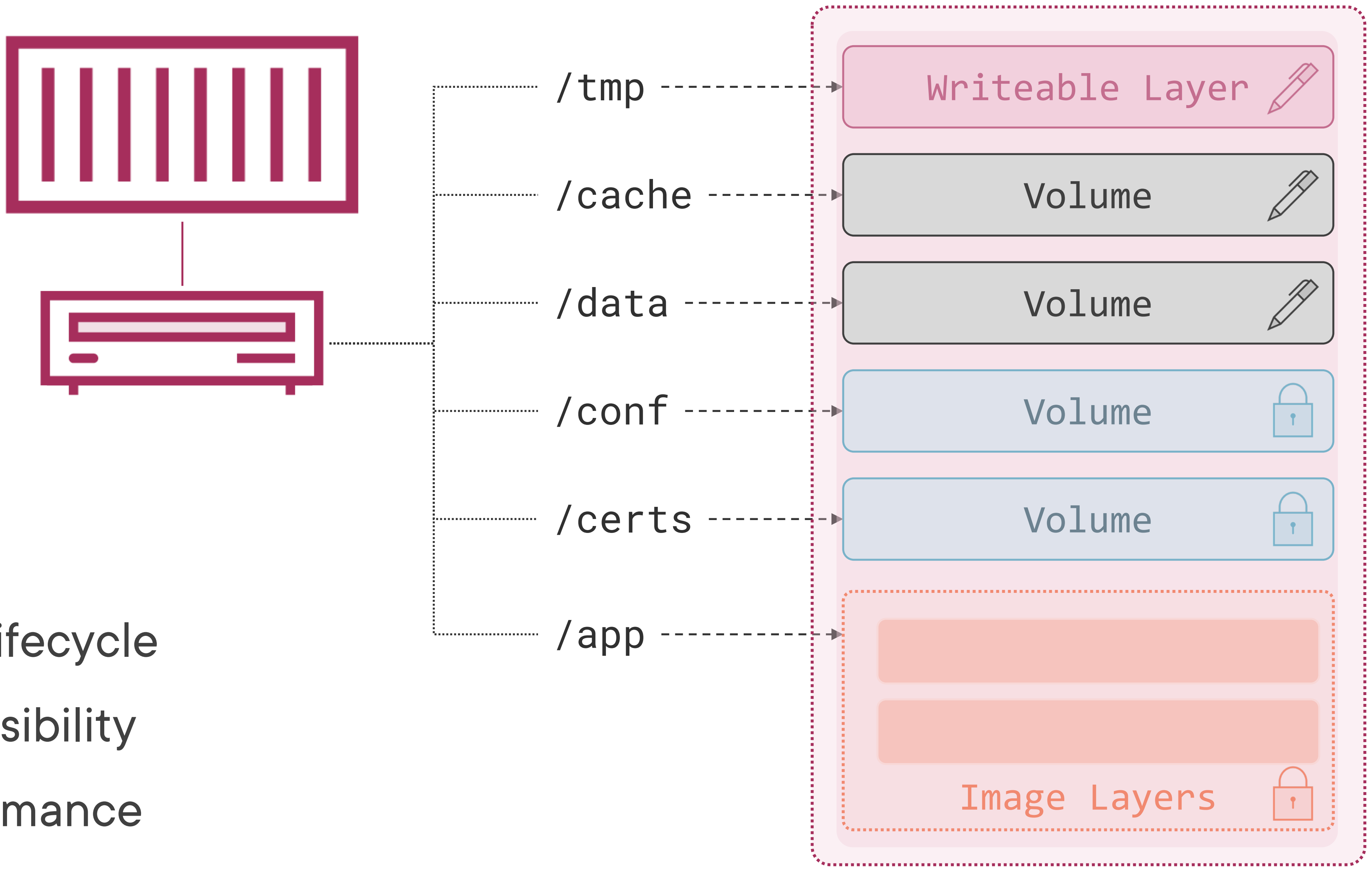
Container runtime

- Docker prune commands
- Kubernetes GC

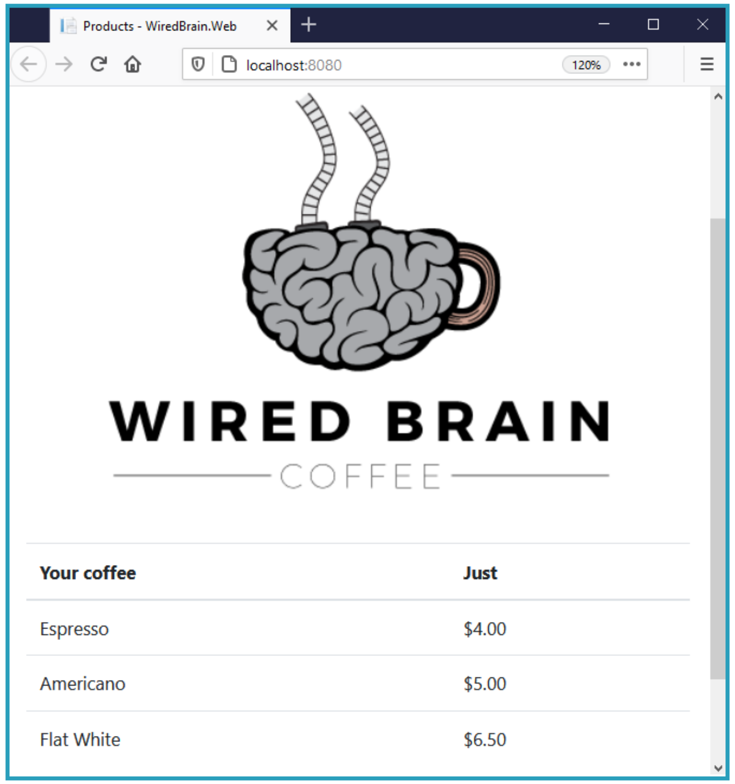

/tmp
/cache
/data
/conf
/certs
/app
Writeable Layer
Volume
Volume
Volume
Volume
Image Layers



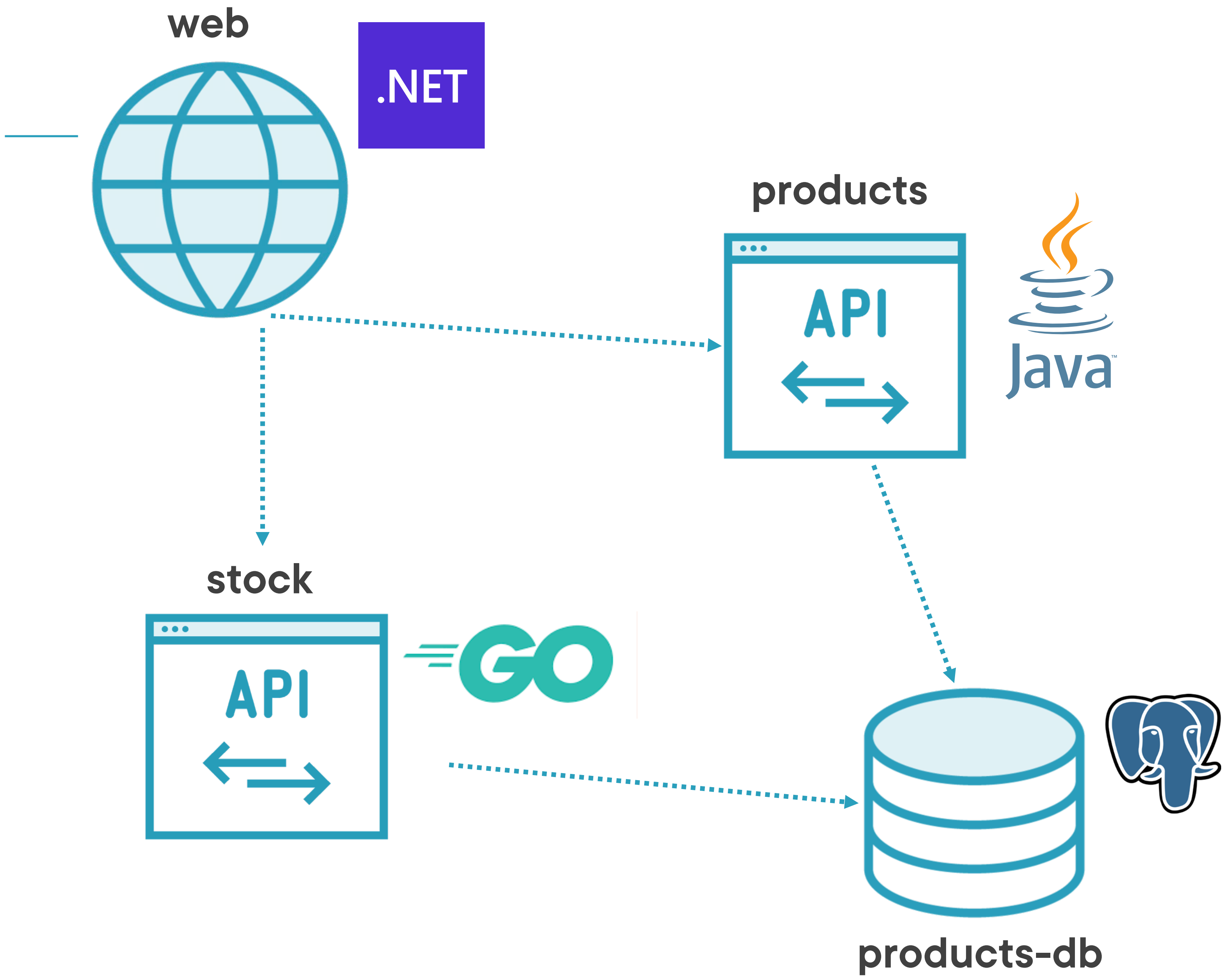
- Data sensitivity
- Config ownership
- Update & rollback



- Data lifecycle
- Accessibility
- Performance



Your coffee	Just
Espresso	$4.00
Americano	$5.00
Flat White	$6.50



We're Done!

So…

- **Please leave a rating**
- **Follow** @EltonStoneman **on Twitter**
- **And watch my other courses ☺**