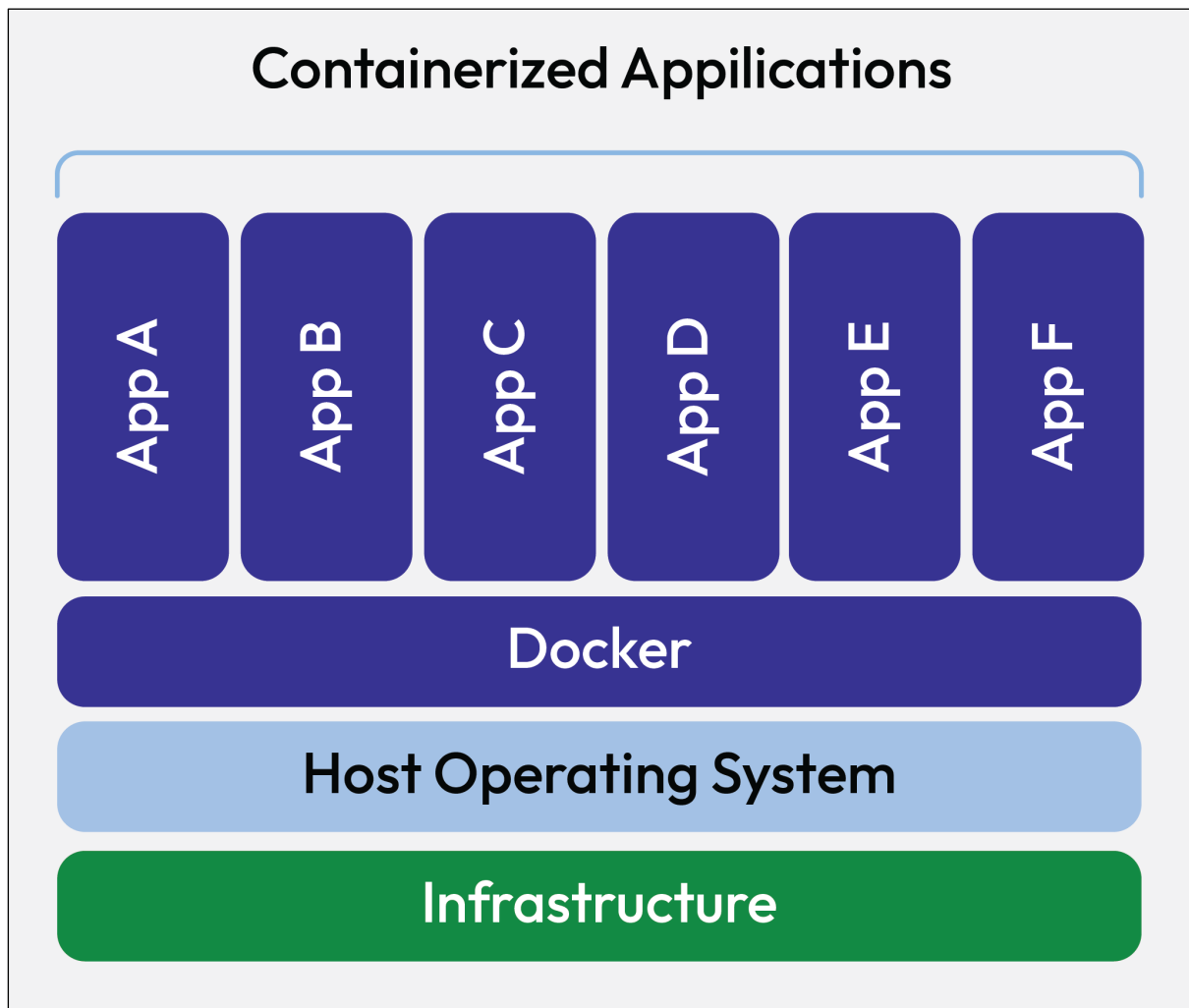
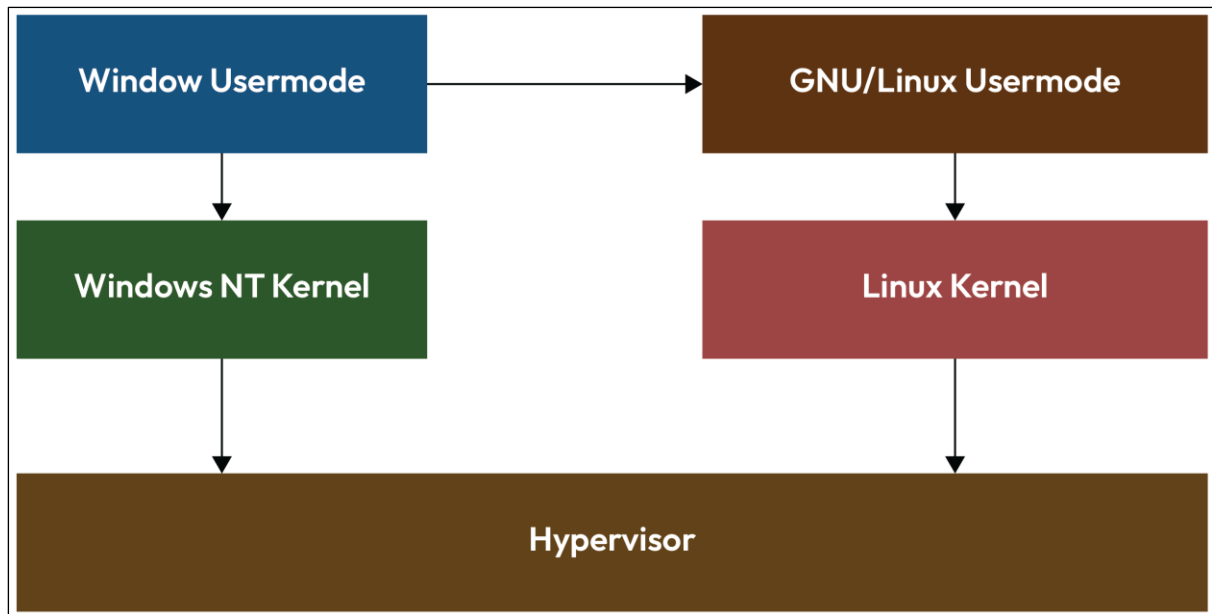


Chapter 1: Installing and Managing Docker on Windows 11



```
PS C:\Users\MikeSmith> wsl --list --verbose
```

NAME	STATE	VERSION
* Ubuntu	Running	2
docker-desktop	Running	2
docker-desktop-data	Running	2



```
PS C:\Users\Mike> docker info
```

```
Client:
 Context:    default
 Debug Mode: false

Server:
 Containers: 3
  Running: 1
  Paused: 0
  Stopped: 2
 Images: 12
 Server Version: 27.0.1
 Storage Driver: overlay2
  Backing Filesystem: ext4
  Supports d_type: true
  Native Overlay Diff: true
```

```
C:\Users\MikeSmith>docker --version
Docker version 24.0.7, build afdd53b
```

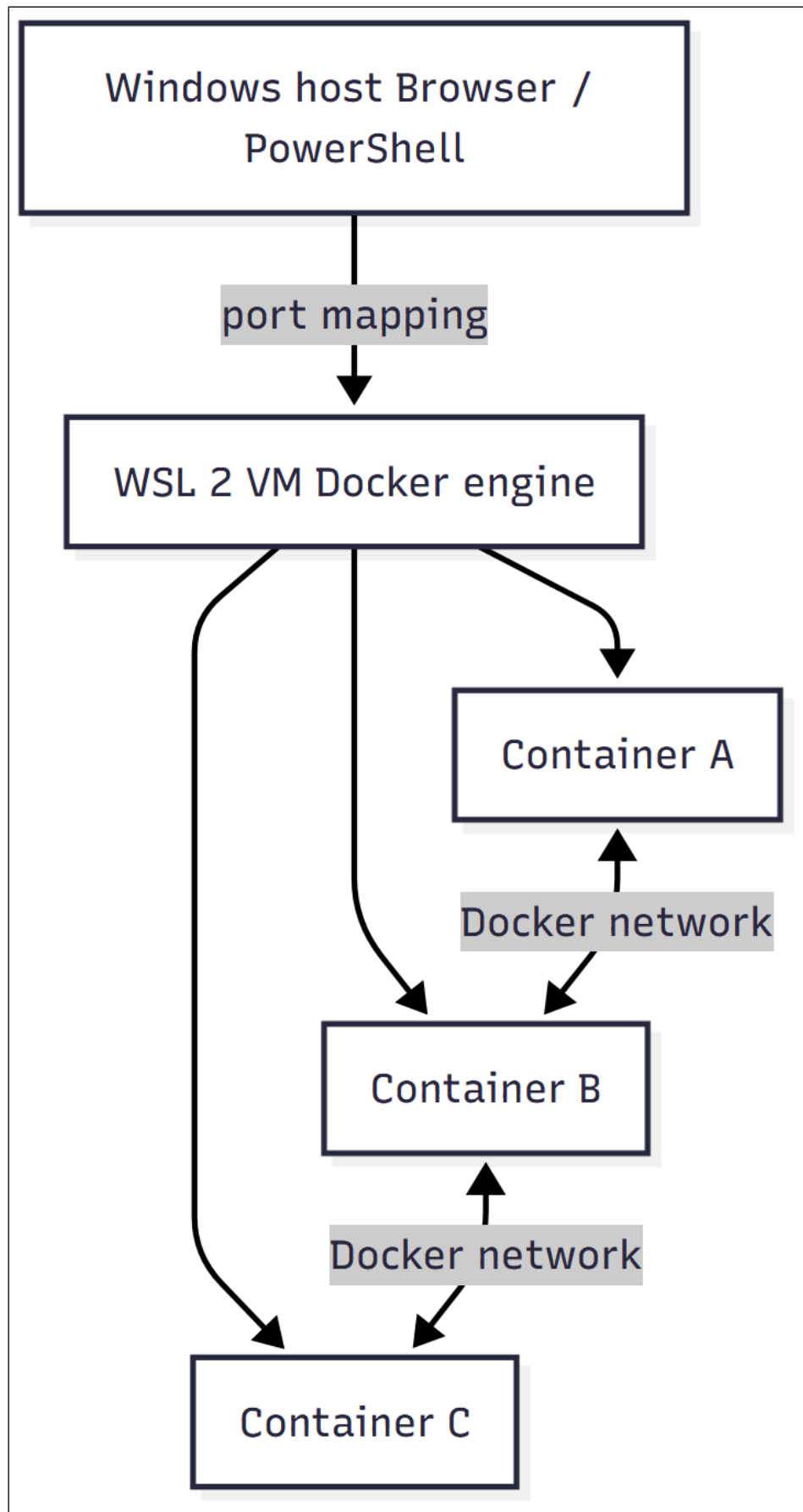
```
PS C:\Users\MikeSmith> wsl --list --verbose
```

NAME	STATE	VERSION
* Ubuntu	Running	2
docker-desktop	Running	2
docker-desktop-data	Running	2

```
[ws12]  
memory=4GB  
processors=2  
swap=2GB
```

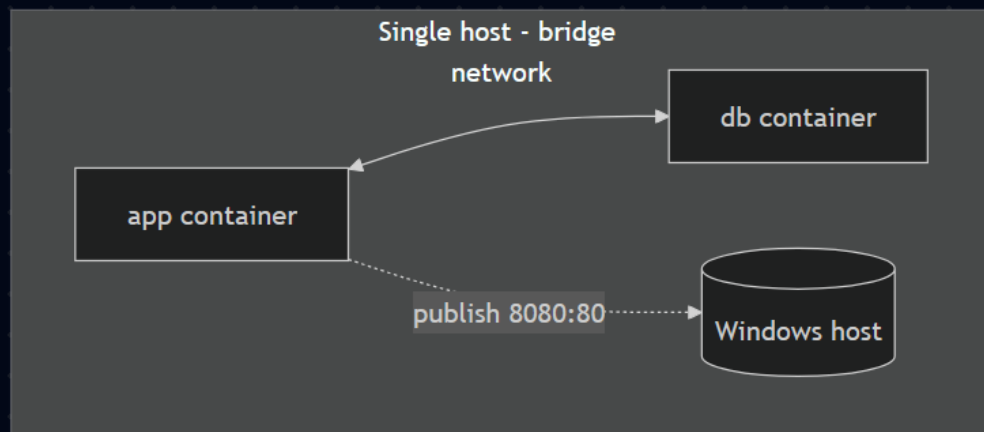
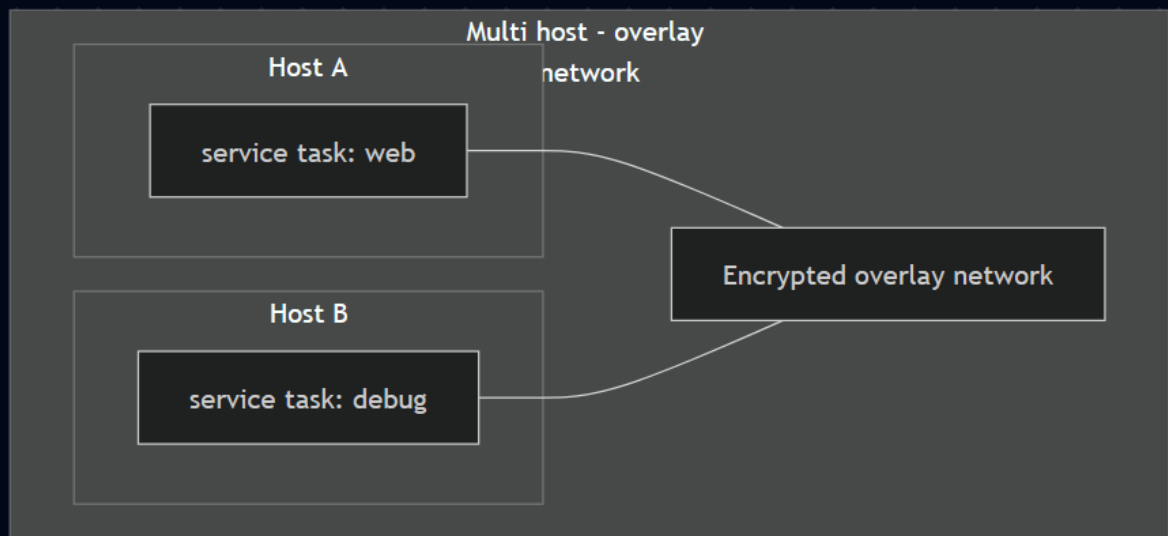
```
FROM node:18  
WORKDIR /app  
COPY package*.json ./  
RUN npm install  
COPY . .  
CMD [ "node", "server.js" ]
```

Chapter 2: Implementing Docker Networking on Windows 11



```
PS C:\Users\MikeSmith> docker inspect my-container
```

```
[
  {
    "HostConfig": {
      "NetworkMode": "host"
    },
    "NetworkSettings": {
      "Bridge": "",
      "SandboxID": "",
      "HairpinMode": false,
      "Ports": {},
      "IPAddress": "",
      "MacAddress": "",
      "Networks": {}
    }
  }
]
```



```
PS C:\Users\MikeSmith> docker network inspect my-secure-overlay
[
  {
    "Name": "my-secure-overlay",
    "Id": "b7f637f4088d7b7323c302ae06b7ed223e04dfacac94cfcb18f1474204fbcdfa",
    "Created": "2024-04-14T17:43:01.6723195Z",
    "Scope": "swarm",
    "Driver": "overlay",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "10.0.0.0/24"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {
      "encrypted": "true"
    },
    "Labels": {}
  }
]
```

```
PS C:\Users\MikeSmith> docker service inspect web
```

```
[
  {
    "Spec": {
      "Name": "web",
      "TaskTemplate": {
        "ContainerSpec": {
          "Image": "nginx:latest"
        }
      },
      "Mode": {
        "Replicated": {
          "Replicas": 1
        }
      },
      "Networks": [
        {
          "Aliases": [ "web" ]
        }
      ],
      "EndpointSpec": {
        "Ports": [
          {
            "Protocol": "tcp",
            "TargetPort": 80,
            "PublishedPort": 8080
          }
        ]
      }
    }
  }
]
```

```
/ # ping web
PING web (10.0.0.2): 56 data bytes
64 bytes from 10.0.0.2: seq=0 ttl=64 time=0.112 ms
64 bytes from 10.0.0.2: seq=1 ttl=64 time=0.084 ms
64 bytes from 10.0.0.2: seq=2 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: seq=3 ttl=64 time=0.065 ms
```

```
/ # ping db
ping: db: Name or service not known
```

```
/ # ping db
PING db (172.18.0.3): 56 data bytes
From 172.18.0.2 icmp_seq=1 Destination Host Unreachable
From 172.18.0.2 icmp_seq=2 Destination Host Unreachable
From 172.18.0.2 icmp_seq=3 Destination Host Unreachable
```

```

"NetworkSettings": {
  "Bridge": "",
  "SandboxID": "2c587d9f39e14787dfd9cf78c29b8d1ef0d6b99c1352ff3e90b8f69b7a9a364b",
  "HairpinMode": false,
  "LinkLocalIPv6Address": "",
  "LinkLocalIPv6PrefixLen": 0,
  "Ports": {},
  "SandboxKey": "/var/run/docker/netns/2c587d9f39e1",
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "0cbad0c1f3a2179462d06c81a568ba89ad310d2543ecfba21f8718e650c64fc1",
  "Gateway": "172.18.0.1",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "172.18.0.2",
  "IPPrefixLen": 16,
  "IPv6Gateway": "",
  "MacAddress": "02:42:ac:12:00:02",
  "Networks": {
    "my-network-2": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": [
        "app",
        "3b28be8283d1"
      ],
      "NetworkID": "2e24a3b27865495a83c28db37d9e0cf91d30d3b8c0605e53ab3ceec78087f053",
      "EndpointID": "0cbad0c1f3a2179462d06c81a568ba89ad310d2543ecfba21f8718e650c64fc1",
      "Gateway": "172.18.0.1",
      "IPAddress": "172.18.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:12:00:02",
      "DriverOpts": null
    }
  }
}

```

```

PS C:\Users\MikeSmith> wsl hostname -I
172.25.16.1

```

```

PS C:\Users\MikeSmith> curl http://localhost:8080
curl: (7) Failed to connect to localhost port 8080: Connection refused

```

```
PS C:\Users\MikeSmith> docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
8f6d8abf9e1e	bridge	bridge	local
9a79e0e7d12a	host	host	local
b3d6ccf0f04a	none	null	local

```
PS C:\Users\MikeSmith> curl http://172.25.16.1:8080
<!DOCTYPE html>
<html>
<head><title>Welcome to nginx!</title></head>
<body>
<h1>Success! The nginx web server is working inside Docker.</h1>
</body>
</html>
```

```

PS C:\Users\MikeSmith> docker inspect my-app
[
  {
    "Id": "e1bf2f4e9f26bd8a50b7f7c24c7a527b1fc2674e7d59d1fc4f1d9d90db9c6c1",
    "Created": "2024-04-14T13:15:52.7168465Z",
    "Path": "node",
    "Args": [
      "server.js"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "StartedAt": "2024-04-14T13:16:03.8213642Z",
      "FinishedAt": "0001-01-01T00:00:00Z"
    },
    "NetworkSettings": {
      "Bridge": "",
      "SandboxID": "c2c2b1e7d0c91ea8d4a8db9d7a8dc4b2",
      "HairpinMode": false,
      "IPAddress": "172.18.0.2",
      "IPPrefixLen": 16,
      "Gateway": "172.18.0.1",
      "Ports": {
        "8080/tcp": [
          {
            "HostIp": "0.0.0.0",
            "HostPort": "8080"
          }
        ]
      }
    }
  }
]

```

```

/ # ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever

15: eth0@if16: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.2/16 brd 172.18.255.255 scope global eth0
        valid_lft forever preferred_lft forever

```

```

PS C:\Users\MikeSmith> docker network inspect my-bridge-network
[
  {
    "Name": "my-bridge-network",
    "Id": "c2b4416e6e2f5d8cd24d9d6c395a12e9d3087bcb7f196f38e232aadcd947b3a6",
    "Created": "2024-04-14T14:27:15.8312345Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Containers": {
      "e1bfb2f4e9f2": {
        "Name": "my-app",
        "EndpointID": "3cc9a0f4f9f37f9d2e1a43d905a0a2c22170a5cd23d92f8df4c7b7de1",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]

```

```

PS C:\Users\MikeSmith> docker network ls

```

NETWORK ID	NAME	DRIVER	SCOPE
8f6d8abf9e1e	bridge	bridge	local
9a79e0e7d12a	host	host	local
b3d6ccf0f04a	none	null	local
c2b4416e6e2f	my-network	bridge	local

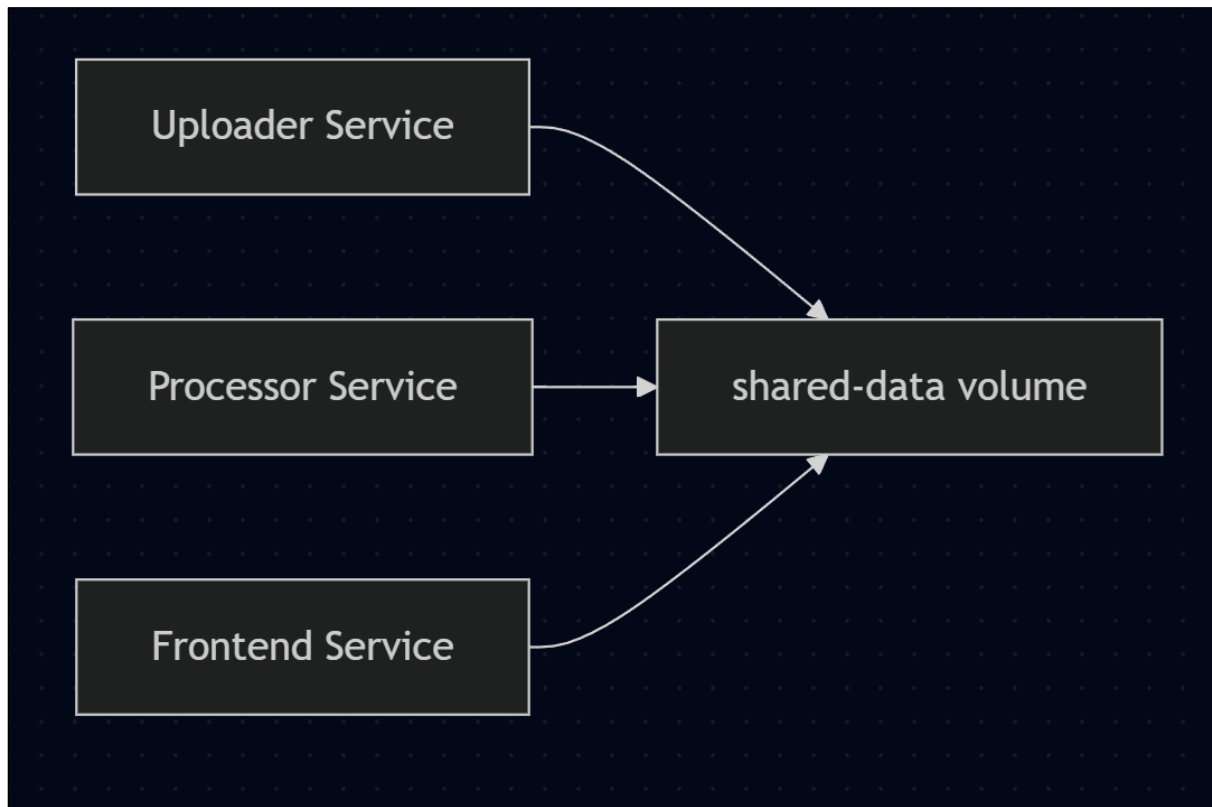
```
PS C:\Users\MikeSmith> docker network inspect my-network
[
  {
    "Name": "my-network",
    "Id": "c2b4416e6e2f5d8cd24d9d6c395a12e9d3087bcb7f196f38e232aadcd947b3a6",
    "Created": "2024-04-14T15:12:34.1234567Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "Containers": {},
    "Options": {},
    "Labels": {}
  }
]
```

```
PS C:\Users\MikeSmith> docker exec -it container-one /bin/sh
/ # ping container-two
PING container-two (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: seq=0 ttl=64 time=0.094 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.057 ms
64 bytes from 172.18.0.3: seq=2 ttl=64 time=0.056 ms
64 bytes from 172.18.0.3: seq=3 ttl=64 time=0.063 ms
```

```
PS C:\Users\MikeSmith> docker network inspect my-network
```

```
[
  {
    "Name": "my-network",
    "Id": "c2b4416e62f58dc24d9d6c395a12e9d387bcb7f196f38e232aadcd947b3a6",
    "Created": "2024-04-14T15:12:34.1234567Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Containers": {
      "e1bfb2f4e9f2": {
        "Name": "container-one",
        "EndpointID": "3cc9a0f4f9f37f9d2e1a43d905a0a2c22170a5cd23d92f8df4c7b7de12345",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      },
      "b3d6ccf0f04a": {
        "Name": "container-two",
        "EndpointID": "4bdc9f4baf9f37f9d2e1a43d905a0a2c22170a5cd23d92f8df4c7b7de67890",
        "MacAddress": "02:42:ac:12:00:03",
        "IPv4Address": "172.18.0.3/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]
```

Chapter 3: Managing Docker Volumes and Data Persistence



```
$ docker volume prune
```

```
WARNING! This will remove all local volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
```

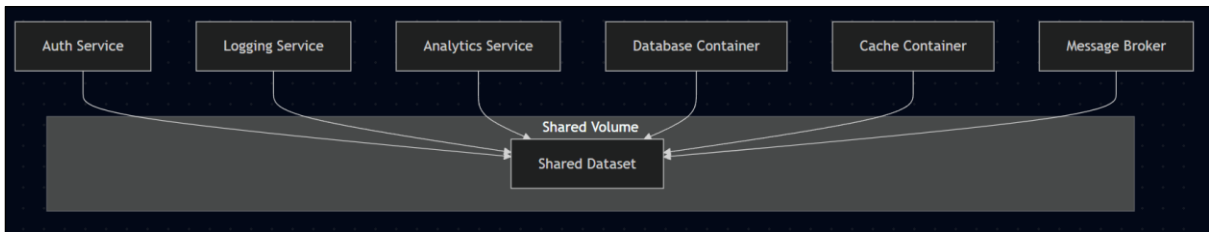
```
Deleted Volumes:
```

```
my_old_volume1
temp_data_volume
unused_cache
```

```
Total reclaimed space: 45.3MB
```

```
$ docker system df
```

TYPE	TOTAL	ACTIVE	SIZE	RECLAIMABLE
Images	5	2	1.73GB	1.10GB (3 unused)
Containers	4	1	305.3MB	250.1MB (3 stopped)
Local Volumes	6	3	912.7MB	420.4MB (3 unused)
Build Cache	8	0	1.02GB	1.02GB



Chapter 4: Orchestrating Multi-Container Applications with Docker Compose



```
1  services:
2    proxy:
3      image: nginx
4      networks:
5        - frontend
6        - backend
7
8    api:
9      build: ./api
10     networks:
11       - backend
12
13    web:
14      build: ./frontend
15      networks:
16        - frontend
17
18  networks:
19    frontend:
20    backend:
```

```

1  networks:
2    frontend_net:
3    backend_net:
4
5  services:
6    db:
7      image: postgres:15
8      networks:
9        backend_net:
10       aliases: [postgres, db]
11
12    api:
13      build: ./api
14      networks: [backend_net]
15
16    auth:
17      build: ./auth
18      networks: [backend_net]
19
20    frontend:
21      build: ./frontend
22      networks: [frontend_net]
23
24    proxy:
25      image: nginx:alpine
26      networks: [frontend_net, backend_net] # bridge between the two

```

PS C:\Users\MikeSmith\my-app> docker compose up

[+] Running 4/4

```

✓ Container my-app-db-1      Started
✓ Container my-app-cache-1   Started
✓ Container my-app-api-1     Started
✓ Container my-app-frontend-1 Started

```

```
PS C:\Users\MikeSmith\my-app> docker compose logs api
api-1 | Waiting for database connection...
api-1 | Attempting to connect to db:5432
api-1 | Error: connect ECONNREFUSED 172.18.0.3:5432
api-1 | Retrying in 5 seconds...
api-1 | Attempting to connect to db:5432
api-1 | Error: connect ECONNREFUSED 172.18.0.3:5432
api-1 | Retrying in 5 seconds...
```

```
PS C:\Users\MikeSmith> wsl hostname -I
172.25.240.1
```

```
PS C:\Users\MikeSmith> netsh interface portproxy add v4tov4 `
>> listenport=3000 listenaddress=127.0.0.1 `
>> connectport=3000 connectaddress=172.25.240.1

Ok.
```

```
PS C:\Users\MikeSmith> curl http://localhost:3000
{"status":"ok","message":"API server is running"}
```

```
PS C:\Users\MikeSmith\my-app> docker compose logs app
app-1 | Starting app server...
app-1 | Connecting to PostgreSQL at db:5432...
app-1 | Error: connect ECONNREFUSED db:5432
app-1 | Retrying in 5 seconds...
app-1 | Connecting to PostgreSQL at db:5432...
app-1 | Error: connect ECONNREFUSED db:5432
app-1 | Retrying in 5 seconds...
app-1 | Connecting to PostgreSQL at db:5432...
app-1 | Connected!
```

```
PS C:\Users\MikeSmith\my-app> docker compose logs app
app-1 | wait-for.sh: waiting for db:5432...
app-1 | wait-for.sh: connection refused, retrying in 2 seconds...
app-1 | wait-for.sh: connection refused, retrying in 2 seconds...
app-1 | wait-for.sh: db is available!
app-1 | Starting app server...
app-1 | App running at http://localhost:3000
```

```
PS C:\Users\MikeSmith\my-app> docker inspect app

[
  {
    "Id": "8a7c8e3f8b27...",
    "Name": "/my-app_app_1",
    "NetworkSettings": {
      "Networks": {
        "my-app_default": {
          "IPAMConfig": null,
          "Links": null,
          "Aliases": [
            "app",
            "8a7c8e3f8b27"
          ],
          "NetworkID": "e1f8a5d4c73b...",
          "EndpointID": "3cd9bfe37b2c...",
          "Gateway": "172.25.0.1",
          "IPAddress": "172.25.0.2",
          "IPPrefixLen": 16,
          "IPv6Gateway": "",
          "GlobalIPv6Address": "",
          "MacAddress": "02:42:ac:19:00:02"
        }
      }
    }
  }
]
```

```
1  version: "3.9"
2
3  services:
4    api:
5      build: ./api
6      ports:
7        - "3000:3000"
8      environment:
9        - DATABASE_URL=postgres://user:pass@db:5432/appdb
10       - REDIS_URL=redis://cache:6379
11       - QUEUE_URL=redis://queue:6379
12      depends_on:
13        - db
14        - cache
15        - queue
16
17    frontend:
18      build: ./frontend
19      ports:
20        - "8080:80"
21      depends_on:
22        - api
23        - auth
24
25    auth:
26      build: ./auth
27      environment:
28        - DATABASE_URL=postgres://user:pass@db:5432/appdb
29      depends_on:
30        - db
31
32    worker:
33      build: ./worker
34      depends_on:
35        - queue
36        - db
37
38    db:
39      image: postgres:15
40      environment:
41        POSTGRES_USER: user
42        POSTGRES_PASSWORD: pass
43        POSTGRES_DB: appdb
44      volumes:
45        - db_data:/var/lib/postgresql/data
46
47    cache:
48      image: redis:alpine
49
50    queue:
51      image: redis:alpine
52
53  volumes:
54    db_data:
```

```
1  version: "3.9"
2
3  services:
4    api:
5      build: ./api
6      ports:
7        - "3000:3000" # API service port
8
9    frontend:
10     build: ./frontend
11     ports:
12       - "8080:80" # Frontend served on HTTP
13
14    database:
15     image: postgres:15
16     environment:
17       POSTGRES_USER: admin
18       POSTGRES_PASSWORD: secret
19     volumes:
20       - db_data:/var/lib/postgresql/data
21
22    redis:
23     image: redis:alpine
24
```

```
PS C:\Users\MikeSmith\my-app> docker volume ls
```

DRIVER	VOLUME NAME
local	myproject_db_data
local	myproject_cache_data

```
PS C:\Users\MikeSmith\my-app> docker volume inspect myproject_db_data
```

```
[
  {
    "CreatedAt": "2025-09-21T12:34:56Z",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/lib/docker/volumes/myproject_db_data/_data",
    "Name": "myproject_db_data",
    "Options": null,
    "Scope": "local"
  }
]
```

```
1  version: "3.9"
2
3  services:
4    frontend:
5      build: ./frontend
6      ports:
7        - "8080:80"
8      depends_on:
9        - api
10       - auth
11
12    api:
13      build: ./api
14      environment:
15        DATABASE_URL: postgres://user:pass@database:5432/appdb
16        CACHE_HOST: cache
17      depends_on:
18        - database
19        - cache
20        - queue
21
22    auth:
23      build: ./auth
24      environment:
25        DATABASE_URL: postgres://user:pass@database:5432/authdb
26      depends_on:
27        - database
28
29    queue:
30      image: redis:alpine
31
32    worker:
33      build: ./worker
34      environment:
35        QUEUE_HOST: queue
36        DATABASE_URL: postgres://user:pass@database:5432/appdb
37      depends_on:
38        - queue
39        - database
40
41    database:
42      image: postgres:15
43      environment:
44        POSTGRES_USER: user
45        POSTGRES_PASSWORD: pass
46        POSTGRES_DB: appdb
47      volumes:
48        - db_data:/var/lib/postgresql/data
49
50    cache:
51      image: redis:alpine
52
53  volumes:
54    db_data:
55
```

```
1 # docker-compose.yml
2 services:
3   api:
4     build: ./api
5     environment:
6       - DATABASE_URL=postgres://user:pass@db:5432/appdb
7     # plus the rest of the Bookings stack...
```

```
1 # docker-compose.override.yml
2 services:
3   api:
4     volumes:
5       - ./api:/app
6     command: ["npm","run","dev"]
7   frontend:
8     volumes:
9       - ./frontend:/usr/share/nginx/html:ro
```

```
1 # docker-compose.prod.yml
2 services:
3   api:
4     environment:
5       - NODE_ENV=production
6     command: ["node","server.js"]
7   frontend:
8     image: myorg/bookings-frontend:1.4.0
```

```
1  services:
2    proxy:
3      image: nginx:alpine
4      volumes:
5        - ./ops/nginx.conf:/etc/nginx/nginx.conf:ro
6      ports:
7        - "8080:80"
8      depends_on: [api]
9      networks: [frontend_net, backend_net]
10
11    api:
12      build: ./api
13      # no direct port mapping when scaling
14      ports: []
15      networks: [backend_net]
16
17  networks:
18    frontend_net: {}
19    backend_net: {}
```

```
1  version: "3.9"
2
3  services:
4    api:
5      build: ./api
6      ports:
7        - "3000:3000"
8      environment:
9        - NODE_ENV=development
10
11    db:
12      image: postgres:15
13      environment:
14        POSTGRES_USER: user
15        POSTGRES_PASSWORD: pass
16        POSTGRES_DB: appdb
17      volumes:
18        - db_data:/var/lib/postgresql/data
19
20    cache:
21      image: redis:alpine
22
23    frontend:
24      build: ./frontend
25      ports:
26        - "8080:80"
27
28  volumes:
29    db_data:
```

```
1  version: "3.9"
2
3  services:
4    api:
5      build: ./api
6      networks:
7        - backend
8
9    database:
10     image: postgres:15
11     networks:
12       - backend
13
14  networks:
15    backend:
16      name: my_backend_net
17
```

```
1  services:
2    database:
3      image: postgres:15
4      networks:
5        backend:
6          aliases:
7            - db
8            - postgres
9
10  networks:
11    backend:
12
```

Chapter 5: Docker Security and Best Practices on Windows

```
PS C:\Users\MikeSmith> docker run --rm alpine id
uid=0(root) gid=0(root) groups=0(root)
```

```
PS C:\Users\MikeSmith> docker stats
```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
a1b2c3d4e5f6	web-app	2.33%	78.5MiB / 2GiB	3.84%	1.2MB / 973kB
b2c3d4e5f6a7	db	1.12%	250.4MiB / 2GiB	12.25%	8.1MB / 3.4MB
c3d4e5f6a7b8	background-worker	89.76%	1.5GiB / 2GiB	75.00%	15.4MB / 14.9MB

```
PS C:\Users\MikeSmith> docker events --since 10m
```

2024-04-14T15:12:32.431234100Z container stop a1b2c3d4e5f6 (image=web-app:latest, name=web-app)
2024-04-14T15:12:35.984623400Z container destroy a1b2c3d4e5f6 (image=web-app:latest, name=web-app)
2024-04-14T15:13:10.106347800Z container create d5e6f7g8h9i0 (image=web-app:latest, name=web-app)
2024-04-14T15:13:10.245789000Z container start d5e6f7g8h9i0 (image=web-app:latest, name=web-app)

```
PS C:\Users\MikeSmith> trivy image node:18-alpine
```

2024-04-14T16:07:42.331Z INFO Need to update DB
2024-04-14T16:07:43.119Z INFO Downloading vulnerability database...

node:18-alpine (alpine 3.18.2)
=====

Total: 3 (HIGH: 1, MEDIUM: 2)

Library	Vulnerability	Severity	Installed Version	Fixed Version
musl	CVE-2023-24536	HIGH	1.2.3-r2	1.2.3-r4
libcurl	CVE-2023-23914	MEDIUM	7.88.1-r0	7.88.1-r2
busybox	CVE-2022-30065	MEDIUM	1.35.0-r19	1.35.0-r21

```
FROM mcr.microsoft.com/dotnet/framework/runtime:4.8
```

```
2025-09-30T12:34:56Z INFO Vulnerability scanning...
dotnet-framework/runtime:4.8 (windows/amd64)
```

LIBRARY	VULNERABILITY	SEVERITY	FIX VERSION
openssl	CVE-2023-5678	HIGH	1.1.1w
zlib	CVE-2022-37434	MEDIUM	1.2.13

```
1  # Use a minimal Node.js base image
2  FROM node:18-alpine
3
4  # Set working directory inside the container
5  WORKDIR /app
6
7  # Copy only the dependency definition files
8  COPY package*.json ./
9
10 # Install only production dependencies and clean up cache
11 RUN npm install --production && npm cache clean --force
12
13 # Copy the rest of the application code
14 COPY . .
15
16 # Start the application
17 CMD ["node", "index.js"]
18
```

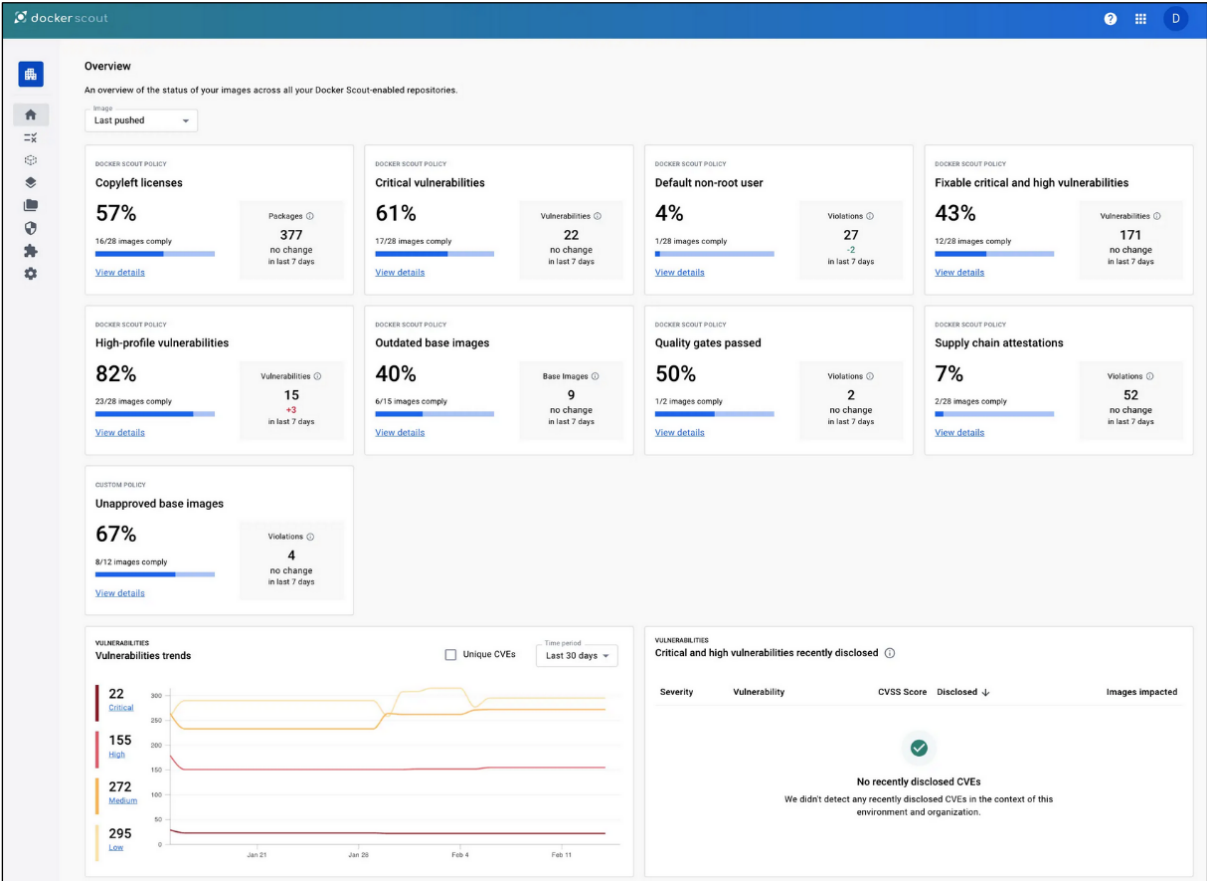
PS C:\Users\MikeSmith> trivy image node:18-alpine

2024-04-14T12:44:55.012Z INFO Need to update DB
2024-04-14T12:44:55.512Z INFO Downloading vulnerability database...

node:18-alpine (alpine 3.18.2)

Total: 5 (HIGH: 2, MEDIUM: 2, LOW: 1)

Library	Vulnerability	Severity	Installed Version	Fixed Version
musl	CVE-2023-24536	HIGH	1.2.3-r2	1.2.3-r4
libcurl	CVE-2023-23914	MEDIUM	7.88.1-r0	7.88.1-r2
busybox	CVE-2022-30065	LOW	1.35.0-r19	1.35.0-r21
openssl	CVE-2023-2650	HIGH	1.1.1t-r0	1.1.1u-r0
zlib	CVE-2022-37434	MEDIUM	1.2.13-r0	1.2.13-r1



```
PS C:\Users\MikeSmith> docker run --rm alpine ps aux
```

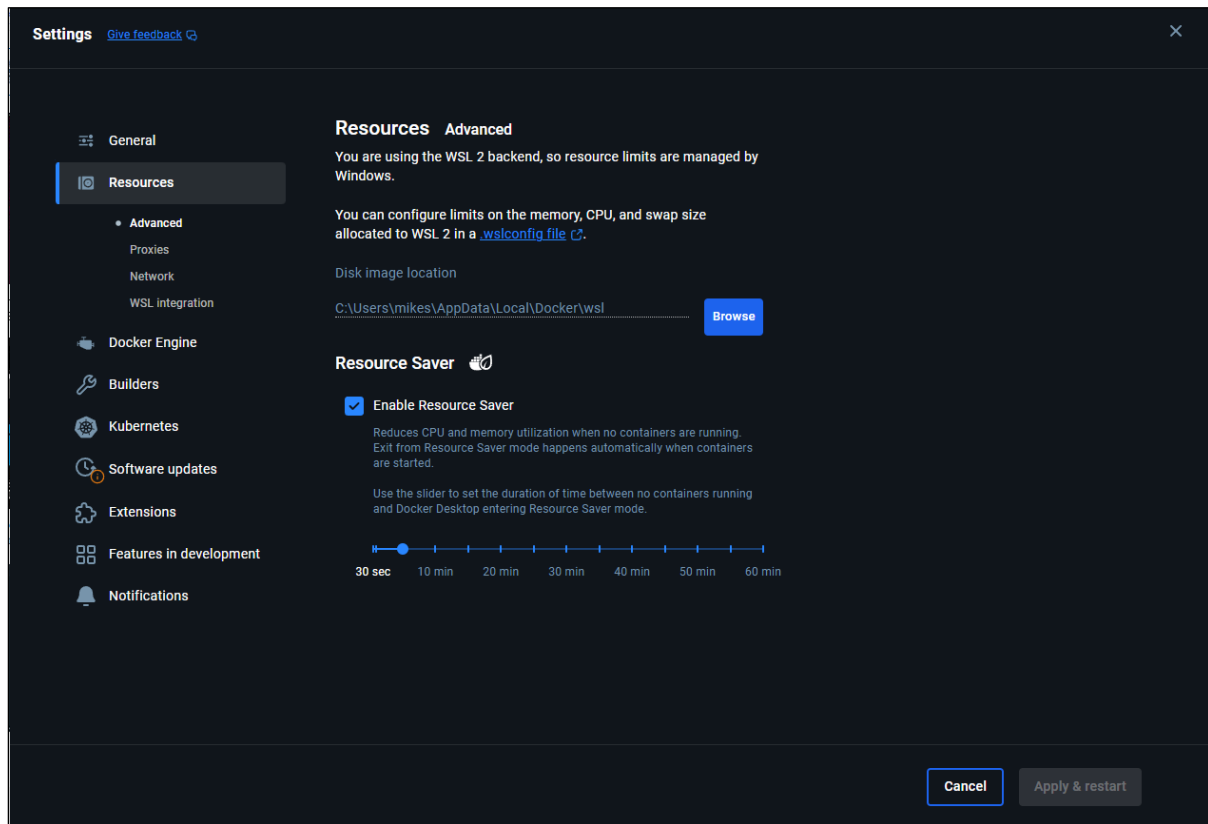
PID	USER	TIME	COMMAND
1	root	0:00	ps aux

```
PS C:\Users\MikeSmith> docker run --rm --pid host alpine ps aux
```

PID	USER	TIME	COMMAND
1	root	0:03	/init
7	root	0:00	/usr/bin/dockerd
14	root	0:00	containerd
22	root	0:00	docker-proxy
27	root	0:00	nginx: master process nginx
34	root	0:00	nginx: worker process
45	root	0:00	alpine
46	root	0:00	ps aux

```
{  
  "userns-remap": "default"  
}
```

Chapter 6: Optimizing Docker for Performance on Windows



```
1  services:
2    web:
3      image: my-web-app
4      deploy:
5        resources:
6          limits:
7            cpus: '0.50'
8            memory: 512M
```

```

1  services:
2    frontend:
3      image: my-frontend
4      deploy:
5        resources:
6          limits:
7            cpus: '0.25'
8            memory: 256M
9
10   backend:
11     image: my-backend
12     deploy:
13       resources:
14         limits:
15           cpus: '1.0'
16           memory: 512M
17
18   db:
19     image: postgres
20     deploy:
21       resources:
22         limits:
23           cpus: '0.75'
24           memory: 768M
25
26
27
28

```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
f3b9e7c2e5b1	web-frontend	1.23%	85.4MiB / 256MiB	33.36%	1.3MB / 1.0MB
8c1e19e4715f	api-service	4.87%	144.6MiB / 512MiB	28.24%	3.1MB / 1.5MB
6db24a11bc92	redis-cache	0.09%	3.8MiB / 128MiB	2.97%	512kB / 432kB
9f5c79b8142a	db-postgres	7.44%	416.2MiB / 1GiB	40.65%	6.5MB / 2.7MB

```

"HostConfig": {
  "CpuShares": 512,
  "Memory": 536870912,
  "MemorySwap": 1073741824,
  "NanoCpus": 1000000000
},
"Config": {
  "Image": "my-web-app:latest",
  "Hostname": "my-container",
  "Cmd": [
    "node",
    "server.js"
  ]
}

```

```

1  services:
2    db:
3      image: postgres:15
4      deploy:
5        replicas: 3
6        resources:
7          limits:
8            cpus: "2"
9            memory: 3G
10   proxy:
11     image: haproxy:alpine
12     ports:
13       - "5432:5432"
14
15

```

ID	NAME	IMAGE	NODE	DESIRED STATE
a1bcdefghijk	web.1	my-web:1.0	docker-node1	Running
b2cdefghijk1	web.2	my-web:1.0	docker-node2	Running
c3defghijklm	web.3	my-web:1.0	docker-node3	Running

```

1  services:
2    web:
3      image: my-web
4      ports:
5        - "8080:80"
6      deploy:
7        replicas: 4
8

```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
a1b2c3d4e5f6	web-app	2.35%	112MiB / 512MiB	21.88%	1.1MB / 800kB
b2c3d4e5f6a7	api-service	6.80%	225MiB / 1GiB	22.00%	2.3MB / 1.7MB
c3d4e5f6a7b8	redis-cache	0.10%	7.1MiB / 128MiB	5.55%	200kB / 150kB
d4e5f6a7b8c9	db-postgres	9.75%	614MiB / 2GiB	30.00%	4.5MB / 3.2MB

```

2025-06-12T10:14:21.123456Z container stop a1b2c3d4e5f6 (image=web-app:latest, name=web-app)
2025-06-12T10:14:23.654321Z container destroy a1b2c3d4e5f6 (image=web-app:latest, name=web-app)
2025-06-12T10:15:10.789012Z container create d5e6f7g8h9i0 (image=web-app:latest, name=web-app)
2025-06-12T10:15:12.345678Z container start d5e6f7g8h9i0 (image=web-app:latest, name=web-app)

```

Starting workload...

Allocating memory...

Killed

Error: Process completed with exit code 137

```

1  [ws12]
2  memory=4GB
3  processors=2
4  swap=2GB
5  localhostForwarding=true

```

```
PS C:\Users\MikeSmith> docker stats
```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
9a2f3e7b22d1	api-service	4.35%	110.3MiB / 512MiB	21.54%	2.1MB / 980kB
b4c1f7a88d03	redis-cache	0.12%	4.3MiB / 128MiB	3.36%	400kB / 300kB
c8a4e3df9912	db-postgres	7.89%	398.7MiB / 1GiB	38.95%	6.5MB / 3.4MB
f7d9a0ccbd2e	web-frontend	1.02%	85.1MiB / 256MiB	33.25%	3.3MB / 2.2MB

```
1 FROM node:18-alpine
2
3 WORKDIR /app
4
5 COPY package*.json ./
6 RUN npm install --production && npm cache clean --force
7
8 COPY . .
9
10 CMD ["node", "index.js"]
11
```

```
1 FROM node:18
2
3 RUN apt-get update && apt-get install -y git curl vim
4
5 COPY . /app
6
7 WORKDIR /app
8
9 RUN npm install
10
```

```
1  # Stage 1: Build everything
2  FROM node:18-alpine as builder
3  WORKDIR /app
4  COPY . .
5  RUN npm install && npm run build
6
7  # Stage 2: Only copy built code
8  FROM node:18-alpine
9  WORKDIR /app
10 COPY --from=builder /app/dist .
11 CMD ["node", "index.js"]
12
13
```

```
$ time docker run --rm my-image
Starting application...
Application running on port 3000

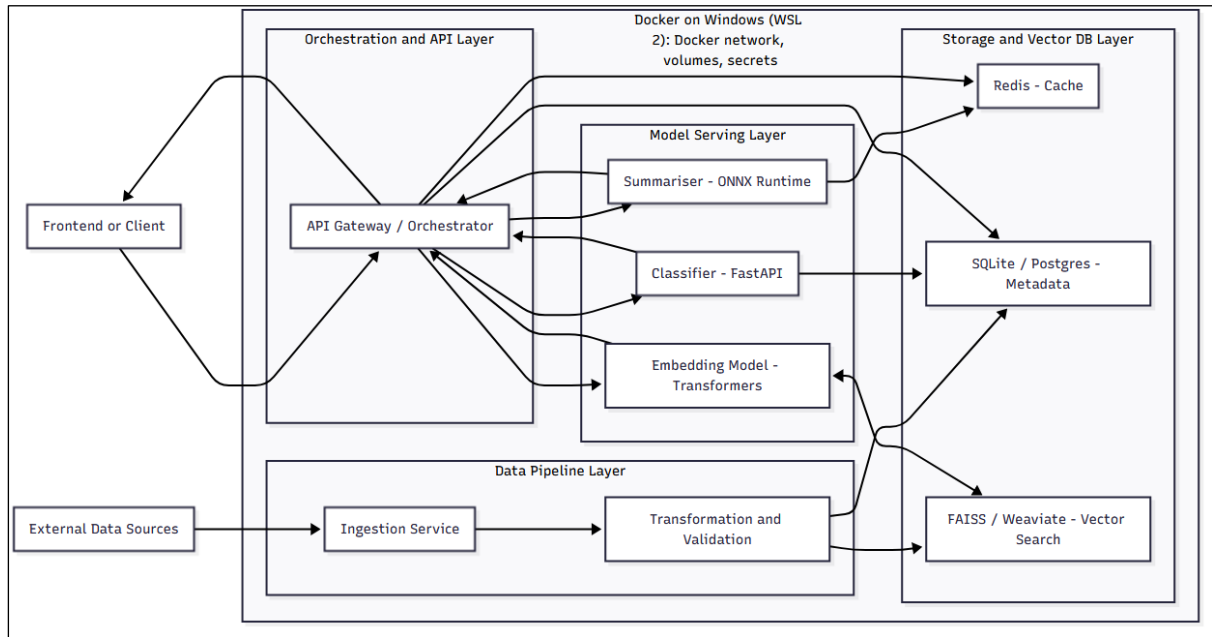
real    0m1.327s
user    0m0.056s
sys     0m0.048s
```

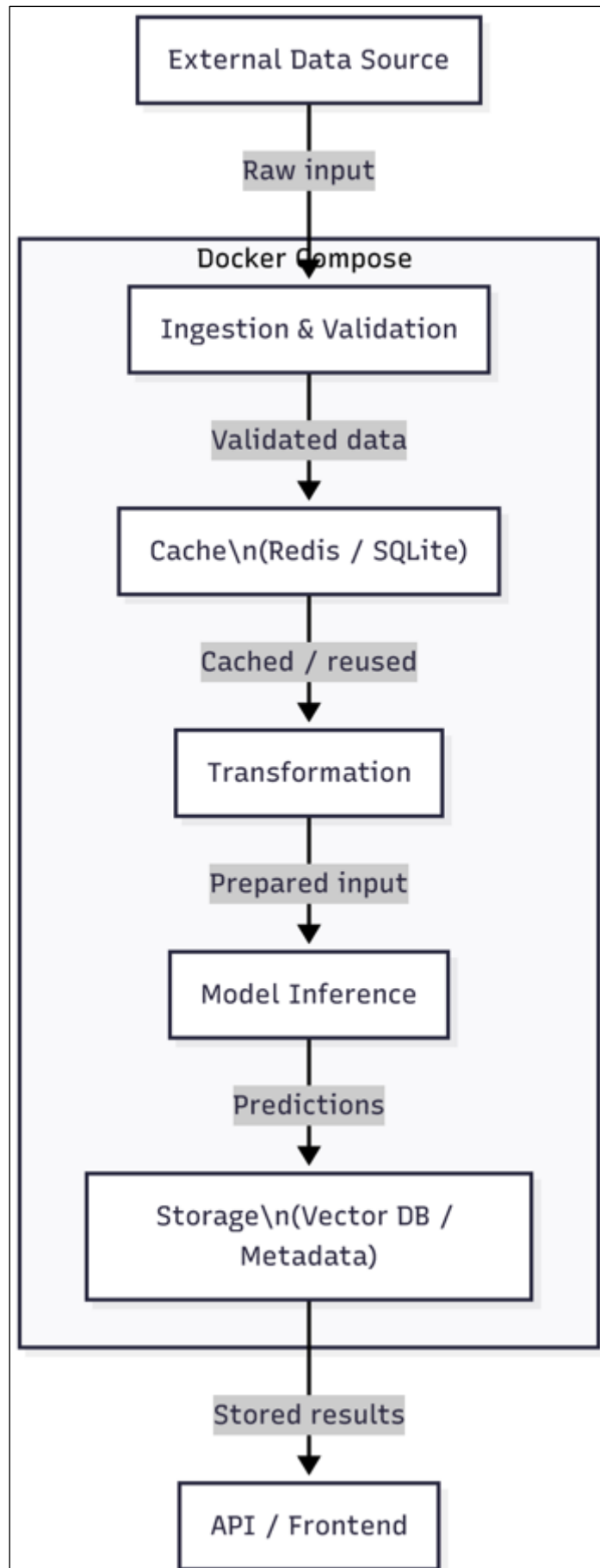
```
1  [ws12]
2  memory=6GB
3  processors=4
4  swap=2GB
5  localhostForwarding=true
```

Killed

Error: Process completed with **exit** code 137

Chapter 7: Docker GenAI Stack on Windows





```
1 FROM python:3.11-slim
2 WORKDIR /app
3 COPY requirements.txt .
4 RUN pip install --no-cache-dir -r requirements.txt
5 COPY ingest.py .
6 CMD ["python", "ingest.py"]
7
```

```
1 import pandas as pd
2
3 # simulate some incoming data
4 data = [
5     {"id": 1, "value": 42},
6     {"id": 2, "value": None}, # Bad record
7     {"id": 3, "value": 17}
8 ]
9
10 valid_records = []
11
12 for record in data:
13     try:
14         if record["value"] is None:
15             raise ValueError("Missing value")
16         valid_records.append(record)
17     except Exception as e:
18         print(f"REJECTED record {record['id']}: {e}")
19
20 # lets output cleaned data
21 df = pd.DataFrame(valid_records)
22 df.to_csv("/data/cleaned.csv", index=False)
23 print(f"INGESTED {len(valid_records)} records successfully")
24
```

```
1   version: '3.8'
2   √ services:
3   √ |   cache:
4   |   image: redis:7
5   √ |   volumes:
6   |   |   - cache-data:/data
7   √ volumes:
8   |   cache-data:
9
10
```

```
1   FROM python:3.11-slim
2   WORKDIR /app
3   RUN pip install fastapi sqlite-utils
4   COPY storage.py .
5   CMD ["python", "storage.py"]
6
```

```

1  from fastapi import FastAPI
2  import sqlite3
3  import os
4
5  # Path to the database file inside the Docker volume.
6  # This will sit inside our mounted volume so it survives restarts.
7  DB_PATH = "/data/storage.db"
8
9  # Make sure the directory exists (Docker volumes can sometimes start fresh)
10 os.makedirs(os.path.dirname(DB_PATH), exist_ok=True)
11
12 # Connect to the database – creates the file if it does not exist yet.
13 conn = sqlite3.connect(DB_PATH)
14 cursor = conn.cursor()
15
16 # Create the table if it is not already there.
17 # This keeps the service fully self-contained: no migrations, no manual steps.
18 cursor.execute("""
19 ✓ CREATE TABLE IF NOT EXISTS embeddings (
20     id INTEGER PRIMARY KEY AUTOINCREMENT,
21     text TEXT NOT NULL,
22     vector BLOB NOT NULL
23 )
24 """)
25 conn.commit()
26
27 app = FastAPI()
28
29 # Basic endpoint to store a new embedding record.
30 # Simple and clean: we store both the text and its vector.
31 @app.post("/store")
32 def store_embedding(text: str, vector: bytes):
33     cursor.execute("INSERT INTO embeddings (text, vector) VALUES (?, ?)", (text, vector))
34     conn.commit()
35     return {"status": "stored"}
36
37 # Simple lookup endpoint to retrieve an embedding by ID.
38 # If the ID is not found, we return a friendly error.
39 @app.get("/lookup/{id}")
40 def get_embedding(id: int):
41     cursor.execute("SELECT text, vector FROM embeddings WHERE id = ?", (id,))
42     result = cursor.fetchone()
43     if result:
44         return {"text": result[0], "vector": result[1]}
45     else:
46         return {"error": "Not found"}

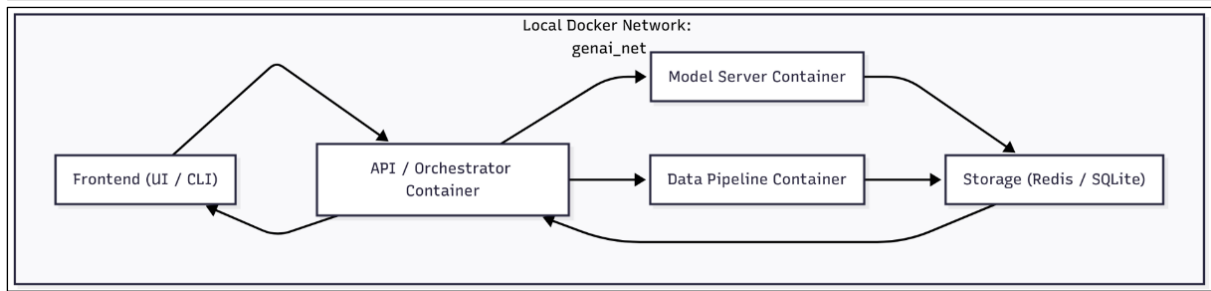
```

```
1  version: '3.8'
2  services:
3    ingester:
4      build: ./ingester
5      volumes:
6        - cleaned-data:/data
7
8    transformer:
9      build: ./transformer
10     volumes:
11       - cleaned-data:/data:ro
12       - transformed-data:/out
13
14     model:
15       image: genai-model:latest
16       volumes:
17         - transformed-data:/input:ro
18       depends_on:
19         - transformer
20
21  volumes:
22    cleaned-data:
23    transformed-data:
```

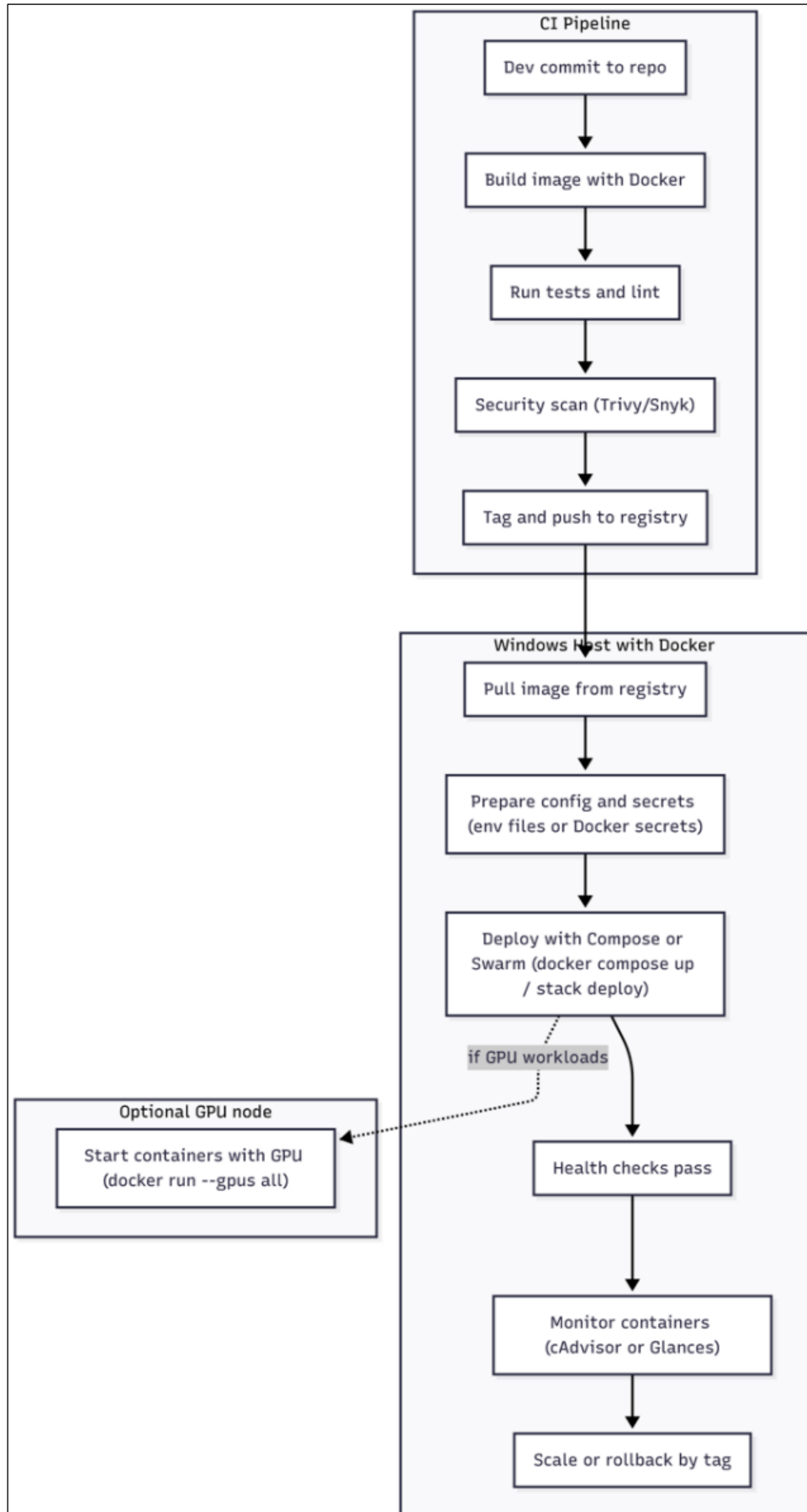
```
1  # Stage 1: Build everything
2  FROM python:3.11-slim as builder
3  WORKDIR /app
4  COPY requirements.txt .
5  RUN pip install --no-cache-dir -r requirements.txt
6  COPY . .
7
8  # Stage 2: Minimal runtime
9  FROM python:3.11-slim
10 WORKDIR /app
11 COPY --from=builder /app .
12 CMD ["uvicorn", "app:app", "--host", "0.0.0.0", "--port", "8000"]
```

```
1  version: '3.8'
2
3  services:
4    model-server:
5      build:
6        context: ./model-server
7      ports:
8        - "8000:8000"
9      depends_on:
10        - preprocessor
11      networks:
12        - genai-net
13
14    preprocessor:
15      build:
16        context: ./preprocessor
17      ports:
18        - "8001:8001"
19      networks:
20        - genai-net
21
22    logger:
23      build:
24        context: ./logger
25      ports:
26        - "8002:8002"
27      networks:
28        - genai-net
29
30    networks:
31      genai-net:
```

```
8 ENV DEFAULT_MODEL=/models/stable-checkpoint.pt
9
```



```
1  version: "3.9"
2
3  services:
4    summariser:
5      image: registry.mycorp.com/genai/summariser:1.0
6      deploy:
7        resources:
8          limits:
9            cpus: "1.0"
10           memory: 2g
11      environment:
12        - MODEL_PATH=/models/summariser.onnx
13      secrets:
14        - openai_key
15      healthcheck:
16        test: ["CMD", "curl", "-f", "http://localhost:8000/health"]
17        interval: 30s
18        retries: 3
19      restart: always
20      user: appuser
21      networks:
22        - genai_net
23
24    orchestrator:
25      build:
26        context: ./orchestrator
27        target: runtime      # comes from a multi-stage Dockerfile
28      ports:
29        - "8080:8080"
30      depends_on:
31        summariser:
32          condition: service_healthy
33      secrets:
34        - db_password
35      user: appuser
36      networks:
37        - genai_net
38
39  secrets:
40    openai_key:
41      file: ./secrets/openai_key.txt
42    db_password:
43      file: ./secrets/db_password.txt
44
45  networks:
46    genai_net:
47      driver: bridge
48
```



```
PS C:\notes-app> curl http://localhost:5001/notes/summary
{
  "api_instance": "api-7c4d1b9fa2e1",
  "genai_instance": "genai-a-4fd232981ab3",
  "summary": "Welcome to the Notes App...",
  "note_count": 3
}

PS C:\notes-app> curl http://localhost:5001/notes/summary
{
  "api_instance": "api-7c4d1b9fa2e1",
  "genai_instance": "genai-b-91aa7e133cd5",
  "summary": "Welcome to the Notes App...",
  "note_count": 3
}

PS C:\notes-app> curl http://localhost:5001/notes/summary
{
  "api_instance": "api-7c4d1b9fa2e1",
  "genai_instance": "genai-a-4fd232981ab3",
  "summary": "Welcome to the Notes App...",
  "note_count": 3
}
```

```
1 FROM python:3.11-slim
2 WORKDIR /app
3 COPY requirements.txt .
4 RUN pip install --no-cache-dir -r requirements.txt
5 COPY . .
6 EXPOSE 8000
7 CMD ["uvicorn", "app:app", "--host", "0.0.0.0", "--port", "8000"]
8
```

```

1  from fastapi import FastAPI
2  import onnxruntime as ort
3
4  model = ort.InferenceSession("model.onnx")
5  app = FastAPI()
6
7  @app.post("/predict")
8  async def predict(data: dict):
9      inputs = data["inputs"]
10     output = model.run(None, {"input": inputs})
11     return {"output": output}
12

```

```

PS C:\Users\MikeSmith\GenAI> curl http://localhost:8000/predict -H

{"output": [[[0.823, 0.114, 0.063]]]}

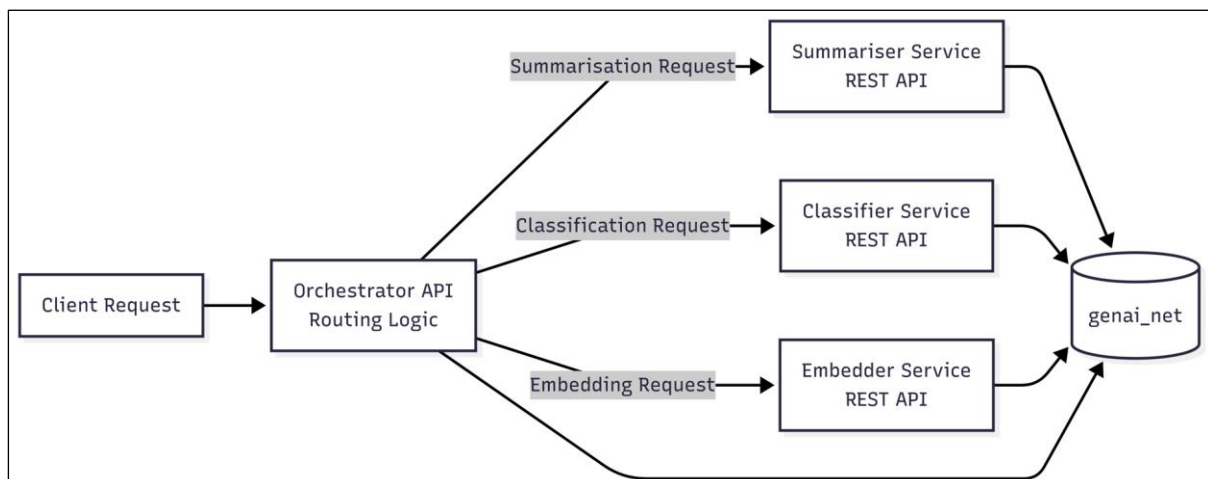
```

```

1  FROM ubuntu:22.04
2
3  RUN apt-get update && apt-get install -y libprotobuf-dev libomp5 && \
4  | python3 -m pip install --no-cache-dir fastapi uvicorn onnxruntime
5

```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
ab12cd34ef56	genai-model	27.35%	512MiB / 1GiB	50.00%	3.2MB / 2.1MB



```
1  version: "3.9"
2  services:
3    summariser:
4      build: ./summariser
5      ports:
6        - "8001:8000"
7      networks:
8        - genai_net
9      deploy:
10       replicas: 2
11      environment:
12       - MODEL_PATH=/models/summariser.onnx
13
14    classifier:
15      build: ./classifier
16      ports:
17        - "8002:8000"
18      networks:
19        - genai_net
20      environment:
21       - MODEL_PATH=/models/classifier.onnx
22
23    embedder:
24      build: ./embedder
25      ports:
26        - "8003:8000"
27      networks:
28        - genai_net
29      environment:
30       - MODEL_PATH=/models/embedder.onnx
31
32    orchestrator:
33      build: ./orchestrator
34      ports:
35        - "8080:8080"
36      depends_on:
37        - summariser
38        - classifier
39        - embedder
40      networks:
41        - genai_net
42
43    networks:
44      genai_net:
45        driver: bridge
46
```

Chapter 8: Implementing Docker for Enterprise Workloads on Windows

```
FROM mcr.microsoft.com/dotnet/aspnet:7.0-windowsservercore-ltsc2022
WORKDIR /app
COPY ./publish /app
EXPOSE 80
ENTRYPOINT ["dotnet", "YourApp.dll"]
```

```
1  version: '3.8'
2
3  services:
4    enterprise-app:
5      image: enterprise-app:1.0.0
6      volumes:
7        - enterprise-data:/data
8
9  volumes:
10 enterprise-data:
```

```
1  version: '3.8'
2
3  services:
4    enterprise-app:
5      image: enterprise-app:1.0.0
6      volumes:
7        - enterprise-data:/data
8
9  volumes:
10 enterprise-data:
11
12
```

```

1 FROM mcr.microsoft.com/dotnet/framework/aspnet:4.8
2
3 COPY . /inetpub/wwwroot
4
5 RUN powershell -Command `
6
7     Import-Module WebAdministration; `
8
9     New-Website -Name "EnterpriseApp" -PhysicalPath
        "C:\inetpub\wwwroot" -Port 80

```



```

{
  "Serilog": {
    "Using": [ "Serilog.Sinks.Console" ],
    "MinimumLevel": "Information",
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "formatter": "Serilog.Formatting.Json.JsonFormatter, Serilog"
        }
      }
    ],
    "Enrich": [ "FromLogContext", "WithMachineName", "WithThreadId" ]
  }
}

```

```

> Run All Services
1 services:
  > Run Service
2 enterprise-app:
3   image: enterprise-app:1.0.0
4   healthcheck:
5     test: ["CMD", "curl", "-f", "http://localhost/healthcheck"]
6     interval: 30s
7     retries: 3
8

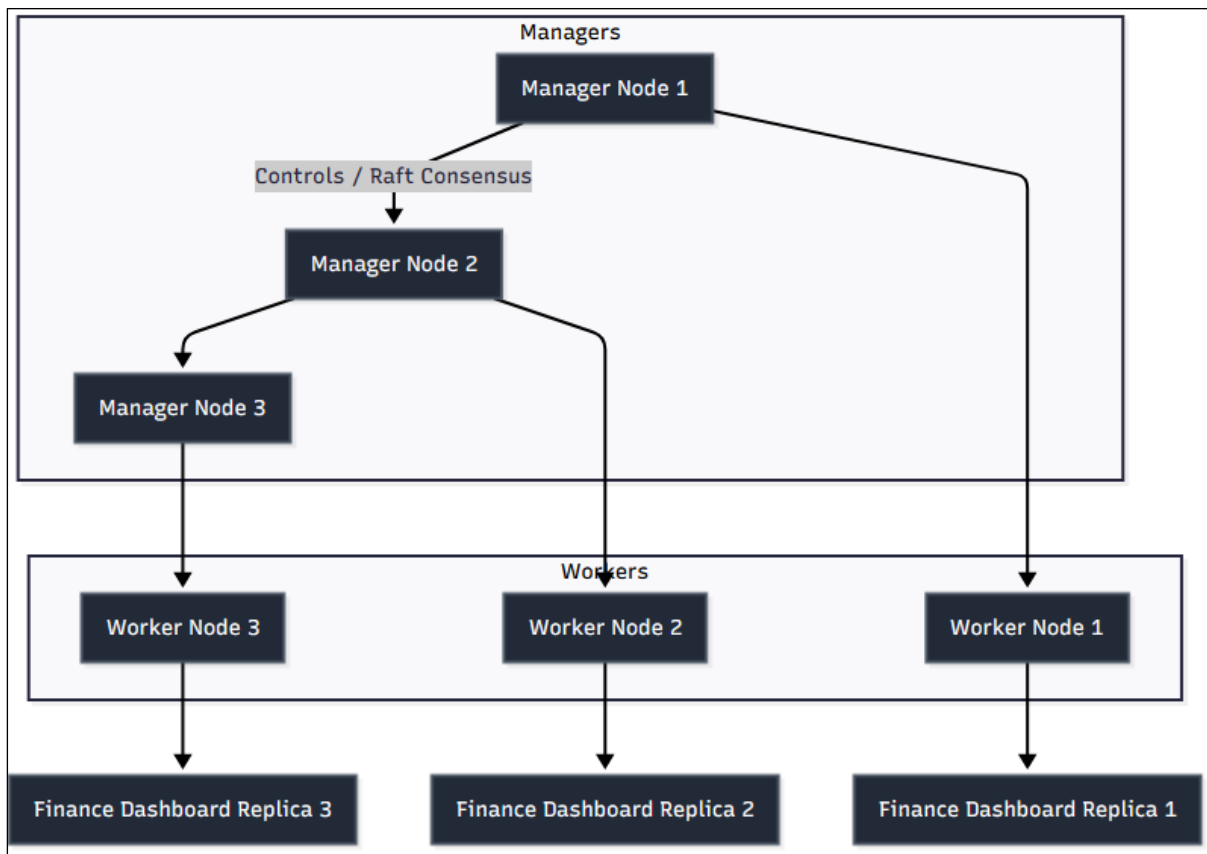
```

```
$ docker service ls
```

ID	NAME	MODE	REPLICAS	IMAGE	PORTS
x8gk1z5w9s5p	webapp	replicated	3/3	myregistry/webapp:latest	*:80->80/tcp

```
$ docker service ps webapp
```

ID	NAME	IMAGE	NODE	DESIRED STATE	CURRENT STATE	ST
ocmlj1u3u7qe	webapp.1	myregistry/webapp:latest	node1	Running	Running	5
4v9enrqv8k3b	webapp.2	myregistry/webapp:latest	node2	Running	Running	4
c9kz0f7r3e6x	webapp.3	myregistry/webapp:latest	node3	Running	Running	4



```
$ docker node promote node2
```

Node node2 promoted to a manager in the swarm.

```
$ docker node ls
```

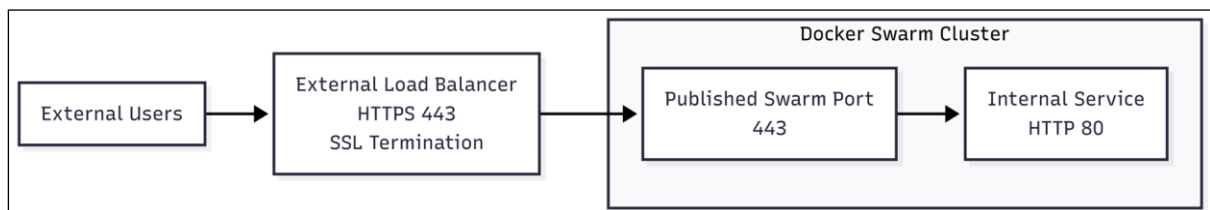
ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS
v0dr0a3k7u5z77fwe1twlh95z *	node1	Ready	Active	Leader
n18ktas3b6pu9nwn0k3k49skx	node2	Ready	Active	Reachable
zw7ifxt8qpjkbywe1fw1xwj28	node3	Ready	Active	Reachable

```
docker service create --name backend \  
--replicas 2 \  
--constraint 'node.labels.role == backend' \  
myregistry/backend:latest
```

```
FROM mcr.microsoft.com/dotnet/aspnet:7.0  
WORKDIR /app  
COPY ./publish /app  
EXPOSE 80  
ENTRYPOINT ["dotnet", "YourApp.dll"]
```

```
$ docker service ps frontend
```

ID	NAME	IMAGE	NODE	DESIRED STATE
ut5sy6g8d55h	frontend.1	myregistry/frontend:1.0.0	node-1	Running
ksh2f3z8shs9	frontend.2	myregistry/frontend:1.0.0	node-2	Running
q9jf73jw2zyx	frontend.3	myregistry/frontend:1.0.0	node-3	Running



```
1 upstream myapp {
2     server app1.internal:8080;
3     server app2.internal:8080;
4 }
5
6 server {
7     listen 443 ssl;
8     location / {
9         proxy_pass http://myapp;
10    }
11 }
```

```
$ docker node inspect node-1 --format '{{ json .Spec.Labels }}'
```

```
{
  "env": "production",
  "region": "eu-west",
  "type": "database"
}
```

```
1 networks:
2     app-net:
3         driver: bridge
4 services:
5     frontend:
6         networks:
7             - app-net
8         depends_on:
9             - backend
```

```
1 [ws12]
2 memory=6GB
3 processors=4
4 swap=2GB
```

```
1 deploy:
2   resources:
3     limits:
4       cpus: '0.50'
5       memory: 512M
```

```
$ docker events --filter 'event=stop' --filter 'event=start' --filter 'container=enterprise-app'
```

```
2025-07-16T08:42:10.123456789Z container stop enterprise-app (container=abc123)
```

```
2025-07-16T08:42:12.987654321Z container start enterprise-app (container=abc123)
```

```
$ docker stats
```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
e4d1b8c3f0a1	frontend-app	12.47%	150.3MiB / 512MiB	29.36%	1.2MB / 980kB
a9b2c4d5e6f7	backend-service	38.21%	420.8MiB / 1GiB	41.13%	4.5MB / 4.1MB
b7c8d9e0f1a2	db-service	7.35%	780.5MiB / 2GiB	38.15%	120kB / 230kB
c1d2e3f4g5h6	redis-cache	0.42%	60.1MiB / 256MiB	23.48%	50kB / 48kB

```
1 secrets:
2   db_password:
3     external: true
4
5 services:
6   backend:
7     secrets:
8     - db_password
```

```
1 FROM mcr.microsoft.com/dotnet/framework/aspnet:4.8
2 COPY . /inetpub/wwwroot
```

```
1 $cutoff = (Get-Date).AddDays(-7)
2 docker image ls --format "{{.Repository}} {{.ID}} {{.CreatedAt}}" | ForEach-Object {
3     $parts = $_ -split " "
4     $created = Get-Date "$($parts[2]) $($parts[3]) $($parts[4])"
5     if ($created -lt $cutoff -and $parts[0] -eq "<none>") {
6         docker image rm $parts[1] -f
7     }
8 }
9
```

```
$ docker ps --format "table {{.Names}}\t{{.Status}}\t{{.Image}}"
```

NAMES	STATUS	IMAGE
web-frontend	Up 3 minutes	myregistry/frontend:3.2.1
api-gateway	Restarting (137) 20 seconds ago	myregistry/api:2.8.0
worker-job	Up 2 hours	myregistry/worker:5.0.0
auth-service	Up 45 minutes	myregistry/auth:1.4.7

```
$ docker stats
```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O
d4f9c8a1e5c2	web-frontend	82.34%	450MiB / 512MiB	87.89%	28.4MB / 12.1MB
9a3e7e44b7fc	api-gateway	95.67%	1.2GiB / 2GiB	60.00%	63.1MB / 58.2MB
3a8b6c7f3e12	auth-service	13.45%	300MiB / 1GiB	29.30%	15.6MB / 14.2MB
7b1f927c3ad2	worker-job	3.12%	75MiB / 256MiB	29.30%	3.4MB / 3.7MB

```
Starting up API service...
```

```
Connecting to database at db.internal:5432...
```

```
[INFO] Loaded config from /etc/myapp/config.yml
```

```
[INFO] Listening on port 8080
```

```
Unhandled exception: timeout while contacting auth service
```

```
Retrying...
```

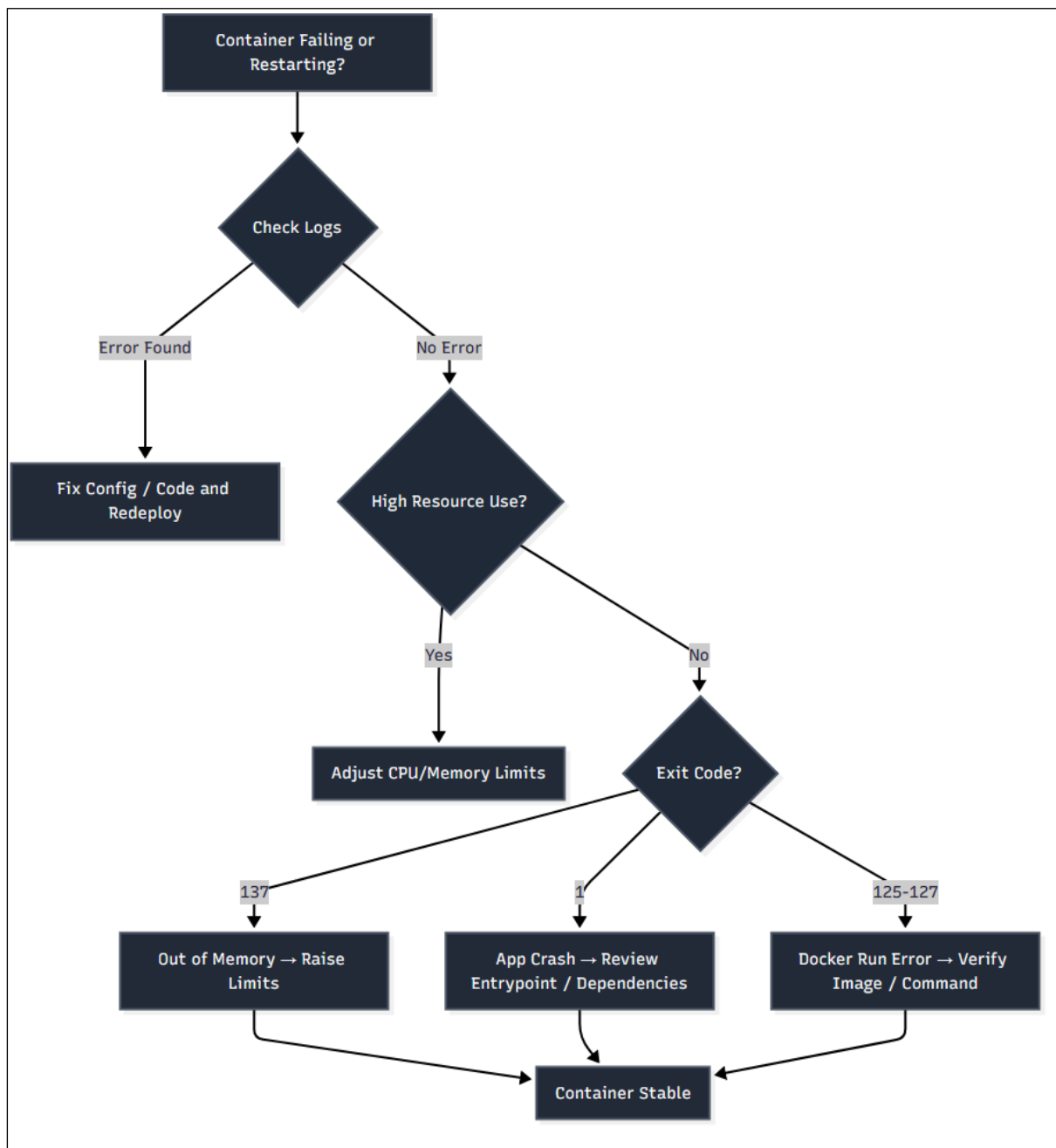
```
Unhandled exception: timeout while contacting auth service
```

```
Fatal error: auth service unavailable
```

```
Shutting down...
```

```
2025-07-16T09:12:01.712443000Z container start 5d1f3a9b9c99 (image=my-api:1.2.3, name=my-api-s
2025-07-16T09:12:15.842117000Z container die 5d1f3a9b9c99 (exitCode=137, image=my-api:1.2.3,
2025-07-16T09:12:16.003944000Z container restart 5d1f3a9b9c99 (image=my-api:1.2.3, name=my-api-
2025-07-16T09:12:30.126487000Z container die 5d1f3a9b9c99 (exitCode=137, image=my-api:1.2.3,
2025-07-16T09:12:30.291027000Z container restart 5d1f3a9b9c99 (image=my-api:1.2.3, name=my-api-
```

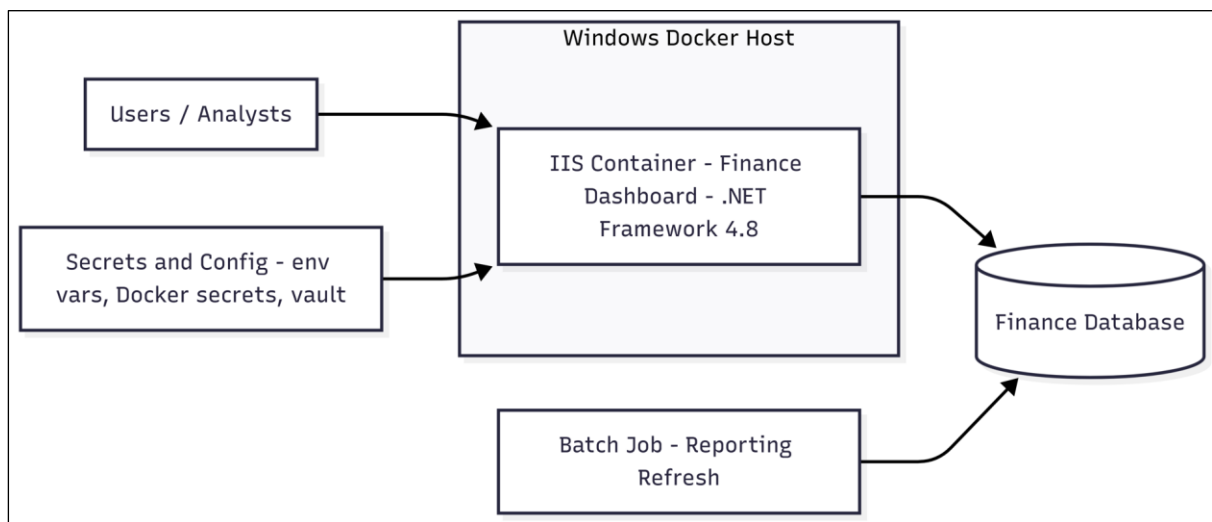
```
$ docker inspect my-api-service --format='{{.State.ExitCode}}'
137
```



```

1  services:
2    heavy-worker:
3      image: analytics-job:latest
4      deploy:
5        resources:
6          limits:
7            cpus: '2'
8            memory: 2G
9

```



```

1  # Finance Dashboard - legacy .NET Framework 4.8 application
2  FROM mcr.microsoft.com/dotnet/framework/aspnet:4.8-windowsservercore-ltsc2019
3
4  # Set the working directory inside the container
5  WORKDIR /inetpub/wwwroot
6
7  # Copy the published web application from your build output
8  COPY ./publish/ .
9
10 # Expose port 80 for HTTP traffic
11 EXPOSE 80
12
13 # Define the default entry point
14 ENTRYPOINT ["C:\\ServiceMonitor.exe", "w3svc"]
15
16

```

```

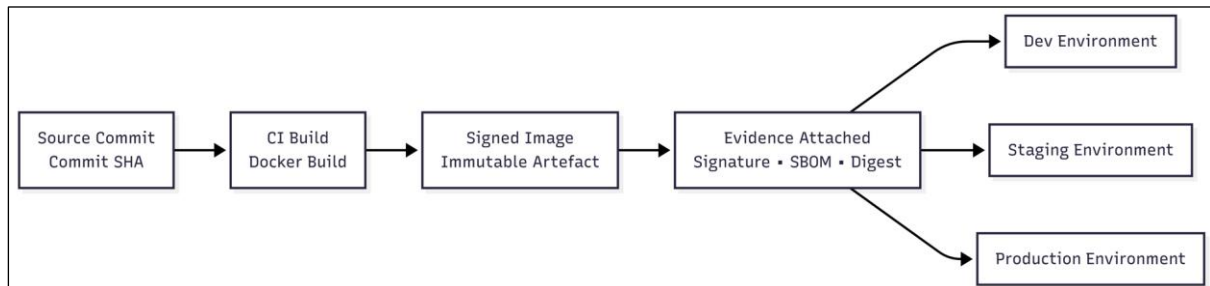
1  services:
2    finance-dashboard:
3      image: finacr.azurecr.io/finance-dashboard:1.0.5
4      secrets:
5        - db_connection
6        - sftp_password
7      environment:
8        - API_KEY_FILE=/run/secrets/db_connection
9        - IMPORT_SFTP_USER=import_user
10
11  secrets:
12    db_connection:
13      external: true
14    sftp_password:
15      external: true
16

```

```

1  FROM mcr.microsoft.com/dotnet/framework/sdk:4.8 AS build-env
2  WORKDIR /app
3  COPY . .
4  RUN msbuild /p:Configuration=Release
5
6  FROM mcr.microsoft.com/dotnet/framework/runtime:4.8
7  COPY --from=build-env /app/bin/Release /inetpub/wwwroot |

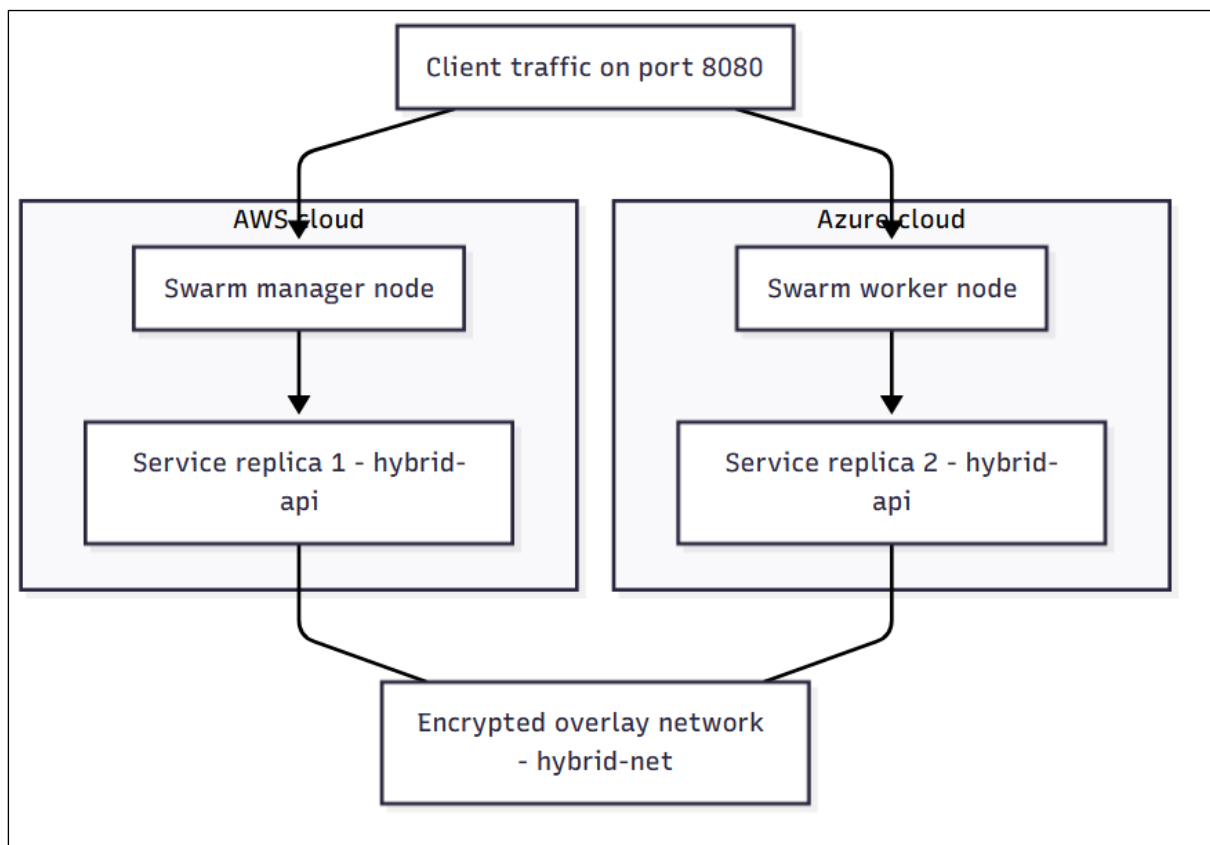
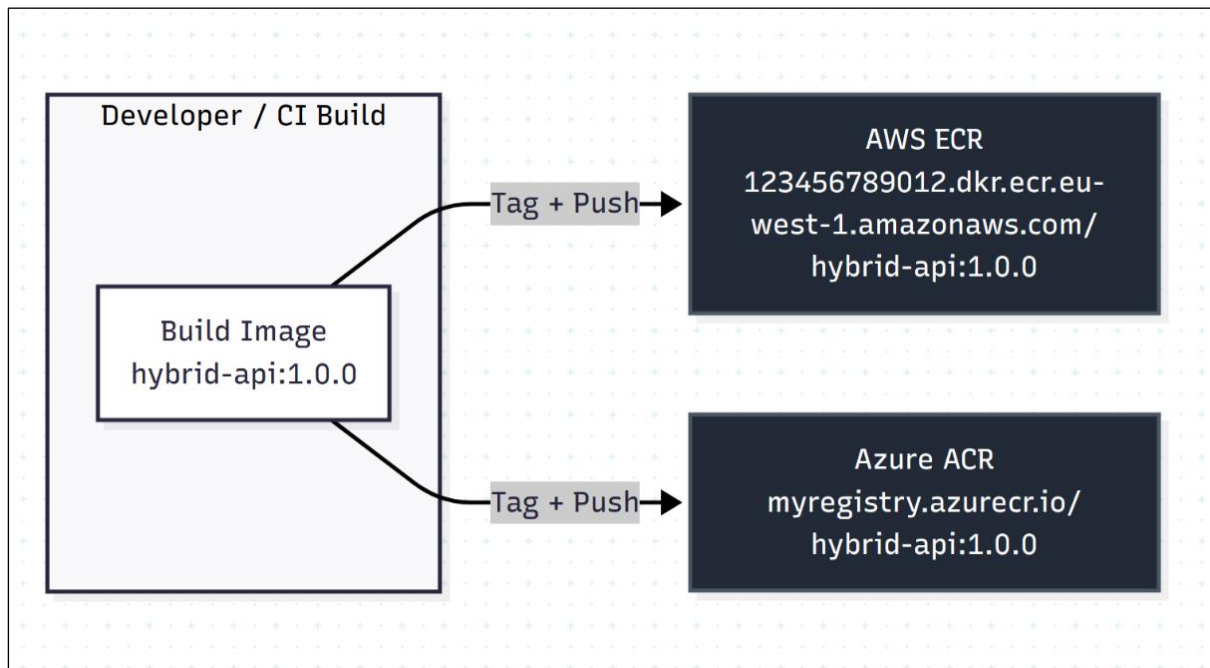
```



```
$ docker build -t myregistry/enterprise-app:1.4.2 .
[+] Building 8.3s (10/10) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 32B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3.11-slim 2.4s
=> [1/5] FROM docker.io/library/python:3.11-slim 0.0s
=> [2/5] WORKDIR /app 0.1s
=> [3/5] COPY . /app 0.1s
=> [4/5] RUN pip install --no-cache-dir -r requirements.txt 3.5s
=> [5/5] CMD ["python", "app.py"] 0.0s
=> exporting to image 0.9s
=> => tagging image myregistry/enterprise-app:1.4.2 0.1s
=> => writing image to destination 0.8s

$ docker push myregistry/enterprise-app:1.4.2
The push refers to repository [myregistry/enterprise-app]
1ab2cd34ef56: Pushed
7890ab12cd34: Pushed
Digest: sha256:abc1234def567890ghijklmnopqrstu890123456789abcdefabcdef12
Status: Pushed myregistry/enterprise-app:1.4.2
```

Chapter 9: Docker and Hybrid Cloud Integration



```
mike@dev-machine:~$ docker network create -d overlay --attachable hybrid-net
hybrid-net
```

```
mike@dev-machine:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
x3k1k9gk1z0d        bridge              bridge              local
z8n4n6pl6g0a        host                host                local
11d2n3m4a5o6        none                null                local
h7i8j9k011m2        ingress            overlay             swarm
hy3b4rid5net         hybrid-net          overlay             swarm
```

```
mike@dev-machine:~$ docker network inspect hybrid-net
```

```
[
  {
    "Name": "hybrid-net",
    "Id": "hy3b4rid5net",
    "Driver": "overlay",
    "Attachable": true,
    "Scope": "swarm"
  }
]
```

```
1  version: '3.8'
2  services:
3    web:
4      image: myapp:latest
5      ports:
6        - "80:80"
7      networks:
8        - hybrid-net
9  networks:
10   hybrid-net:
11     external: true
12
13
```

Creating network hybrid-net

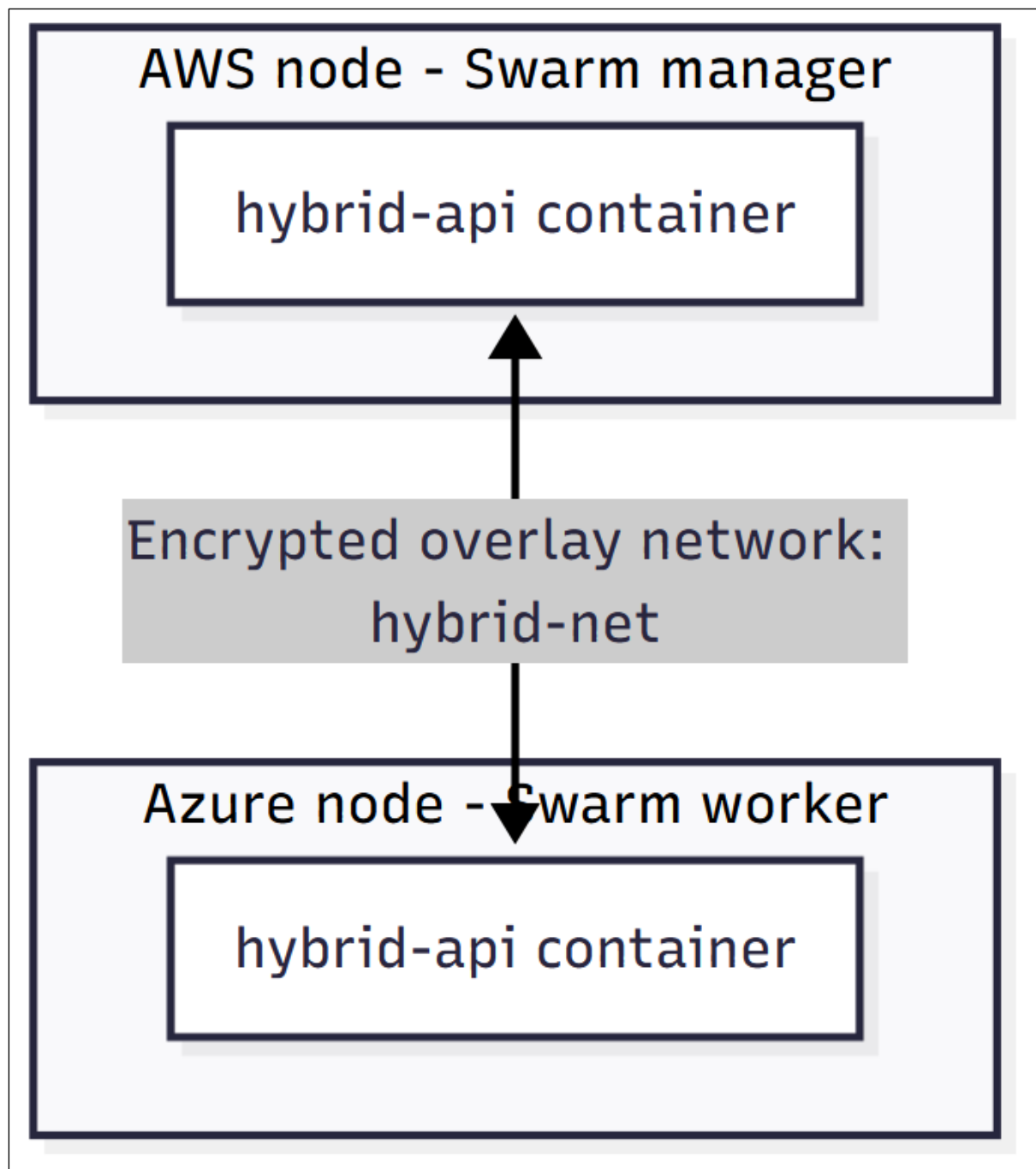
Creating service hybrid-stack_web

Stack hybrid-stack deployed

NAME	SERVICES	ORCHESTRATOR
hybrid-stack	1	Swarm

```
DOCKER_CONTENT_TRUST=1 docker push myregistry.azurecr.io/myapp:1.3.0
The push refers to repository [myregistry.azurecr.io/myapp]
Preparing manifest for docker trust
Signing and pushing trust metadata
Signing and pushing tag: myapp:1.3.0
Calculating and signing hashes...
Successfully signed image: myregistry.azurecr.io/myapp:1.3.0

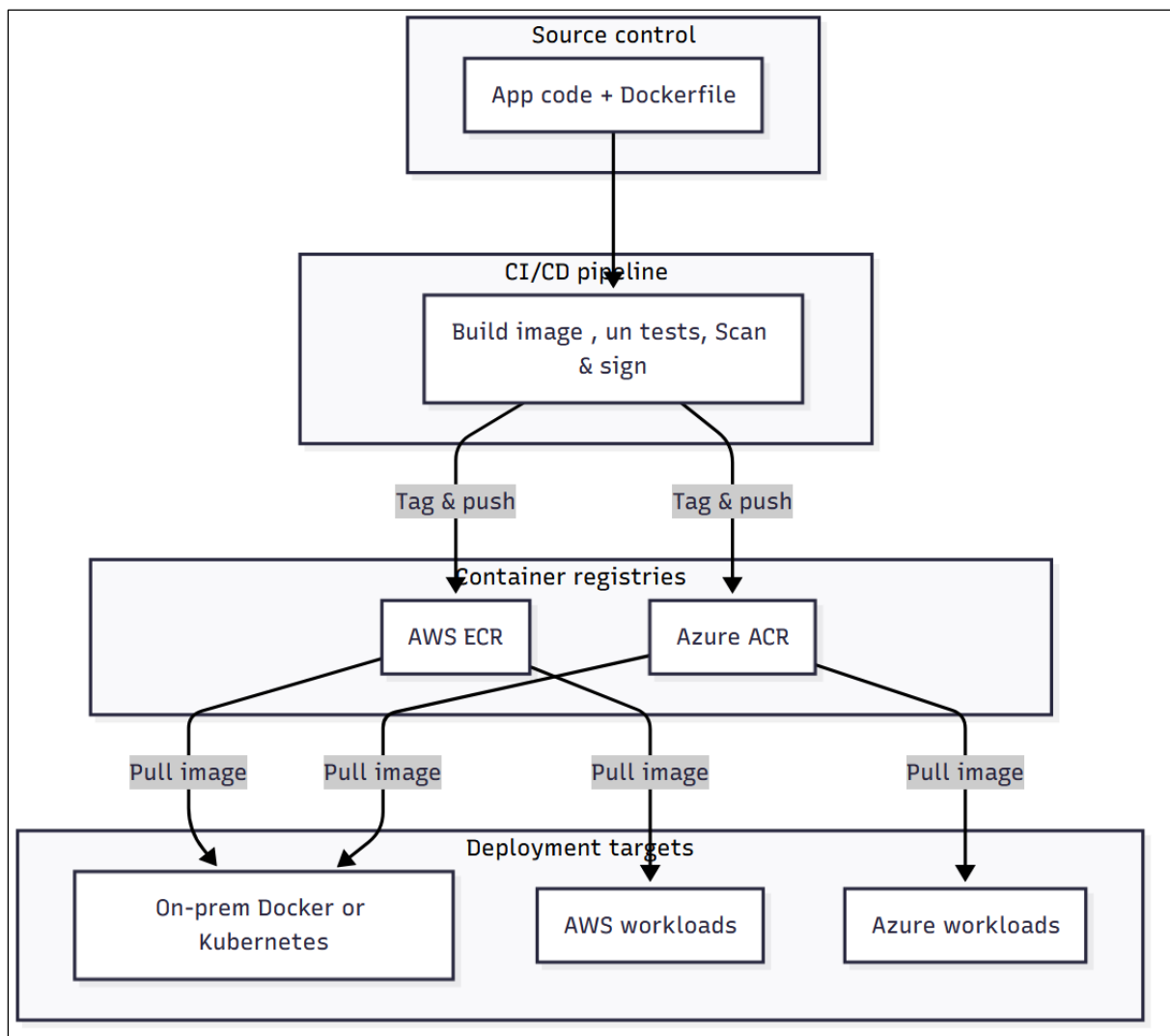
latest: digest: sha256:98f9c90af443b76c1234567abc89f3f51234efbc6789def000aabbccddeeff11 size:
Tagging complete
Pushed: myregistry.azurecr.io/myapp@sha256:98f9c90af443b76c1234567abc89f3f51234efbc6789def000a
Signing complete for tag: myapp:1.3.0
```



```
1 jobs:
2   build-and-push:
3     runs-on: ubuntu-latest
4     steps:
5       - uses: actions/checkout@v3
6       - name: Build image
7         run: docker build -t myapp:${GITHUB_SHA} .
8       - name: Push to AWS
9         run: docker push 123456789012.dkr.ecr.eu-west-1.amazonaws.com/myapp:${GITHUB_SHA}
10      - name: Push to Azure
11        run: docker push myregistry.azurecr.io/myapp:${GITHUB_SHA}
```

```
1  #docker-compose.yml
2  version: "3.9"
3  services:
4    web:
5      image: myapp:latest
6      ports:
7        - "80:80"
8      environment:
9        - ENVIRONMENT=production
10
```

```
1  #docker-compose.override.yml
2  version: "3.9"
3  services:
4    web:
5      environment:
6        - ENVIRONMENT=development
7
8
9
10
```



```
1  services:
2    db-client:
3      image: myapp:latest
4      secrets:
5        - db_password
6  secrets:
7    db_password:
8      external: true
9
```

```

1  services:
2    | reports:
3    |   image: reports-app:latest
4    |   volumes:
5    |     - secure-data:/data:ro
6  volumes:
7    | secure-data:
8    |   driver: local
9

```

```

1  permissions:
2    | id-token: write
3    | contents: read
4

```

```

1  name: Secure Docker Build
2
3  on:
4    | workflow_dispatch:
5
6  jobs:
7    | build-and-push:
8    |   runs-on: [self-hosted, ephemeral] # This ensures the runner is created for
9    |   this job only
10   |   steps:
11   |     - name: Checkout code
12   |       uses: actions/checkout@v3
13
14   |     - name: Login to AWS ECR
15   |       run: |
16   |         aws ecr get-login-password --region eu-west-1 | \
17   |         docker login --username AWS --password-stdin 123456789012.dkr.ecr.
18   |         eu-west-1.amazonaws.com
19
20   |     - name: Build Docker image
21   |       run: docker build -t mysecureapp:latest .
22
23   |     - name: Tag and Push
24   |       run: |
25   |         docker tag mysecureapp:latest 123456789012.dkr.ecr.eu-west-1.
26   |         amazonaws.com/mysecureapp:latest
27   |         docker push 123456789012.dkr.ecr.eu-west-1.amazonaws.com/
28   |         mysecureapp:latest

```

```
$ docker swarm init
Swarm initialized: current node (abcd1234) is now a manager.

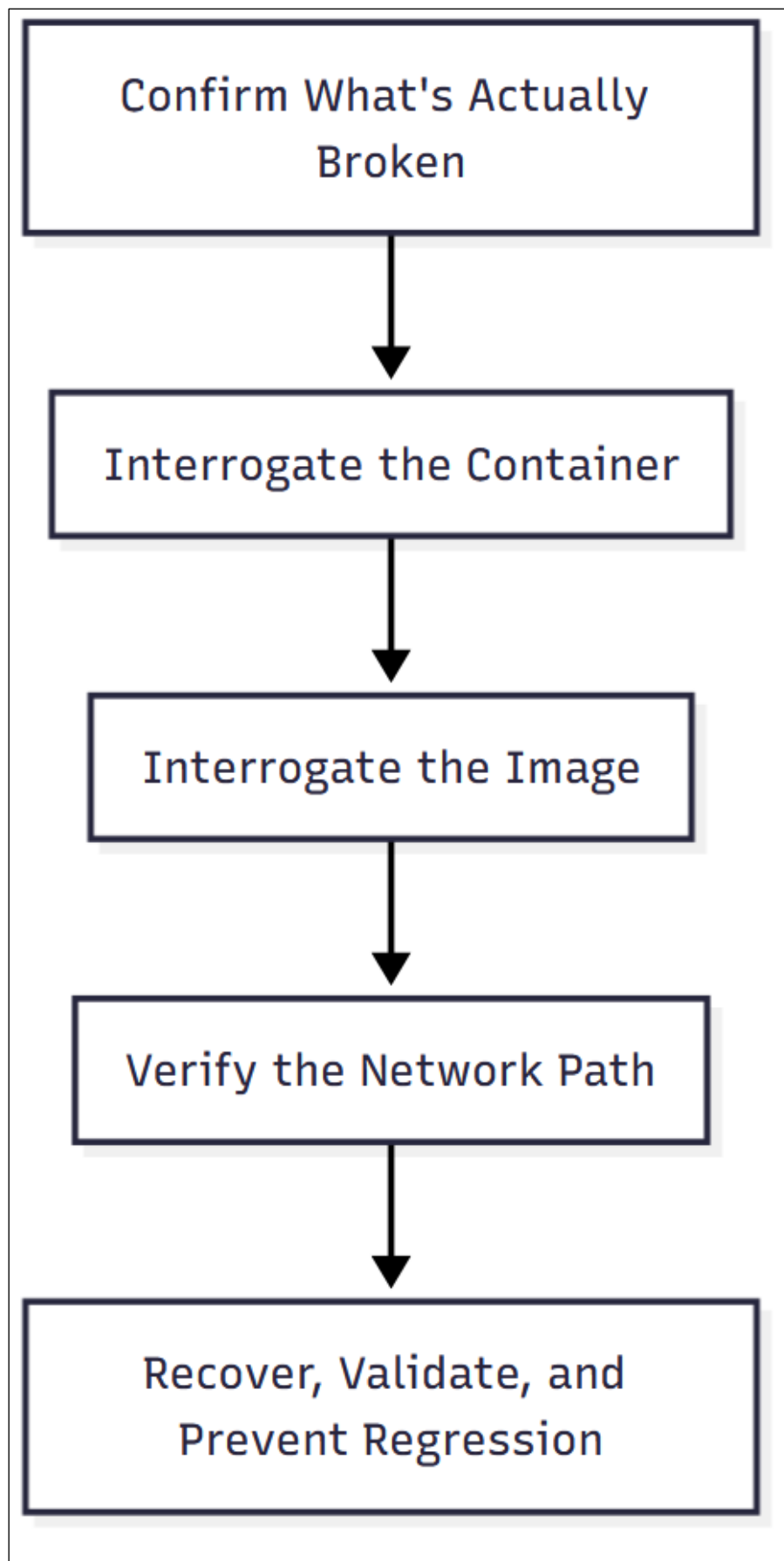
$ docker swarm join --token SWMTKN-1-abc123xyz <manager-ip>:2377
This node joined a swarm as a worker.

$ docker node ls
ID                HOSTNAME        STATUS    AVAILABILITY    MANAGER STATUS
abcd1234          node1           Ready     Active           Leader
efgh5678          node2           Ready     Active

$ docker service create --name webapp --replicas 3 -p 80:80 nginx:latest
vxyz8901qwe123 created

$ docker service ls
ID                NAME        MODE          REPLICAS    IMAGE
vxyz8901q         webapp      replicated    3/3          nginx:latest
```

Chapter 10: Troubleshooting Docker in Production Environments



```
PS C:\Users\mike> dive myapp:prod
```

```
Analyzing Image... (sha256:a92f1dbf0b33)
```

Size	Change	Command
148 MB	145 MB	RUN npm install
12 MB	12 MB	COPY ./dev-config/ /app/config/
3 MB	3 MB	COPY ./src /app/src
1 KB	1 KB	COPY ./package.json /app/
5 MB	5 MB	FROM node:18-alpine

```
Layer Details - Highlights:
```

- ✓ Large dependency layer detected (145 MB)
- ✗ Unexpected directory: /app/config/dev.json
- ✗ Missing expected file: /app/config/redis.json
- ✗ dev-config directory included in production image
- ✓ Build metadata inconsistent with staging image

```
Image inefficiency score: 27% (Poor)
```

```
PS C:\Users\mike> docker run -it --rm myapp:prod sh
```

```
sh: can't open '/app/worker.sh': No such file or directory
```

```
/app # pwd
```

```
/app
```

```
/app # ls -lah
```

```
total 12K
```

```
drwxr-xr-x  1 root    root    4.0K Nov 14 09:20 .
```

```
drwxr-xr-x  1 root    root    4.0K Nov 14 09:20 ..
```

```
drwxr-xr-x  2 root    root    4.0K Nov 14 09:20 config
```

```
drwxr-xr-x  3 root    root    4.0K Nov 14 09:20 src
```

```
-rwxr-xr-x  1 root    root      0 Nov 14 09:20 worker.sh
```

```
/app # cat worker.sh
```

```
# (empty file)
```

```
    "Networks": {
      "frontend": {
        "IPAMConfig": {},
        "Links": null,
        "Aliases": [
          "api",
          "api_1"
        ],
        "NetworkID": "8f3c9d1c2a4a",
--
      "bridge": {
        "IPAMConfig": null,
        "Links": null,
        "Aliases": [
          "api"
        ],
        "NetworkID": "3b19a8d0f6b1"
      }
    }
  }
}
```

```
[
  {
    "Name": "backend",
    "Id": "7c1f98d1f2bc3e3c5d8d2484c41e50a4c9e4afba7e20fb89a3f2e2e12af17df3",
    "Created": "2025-11-14T09:10:22.123456789Z",
    "Scope": "local",
    "Driver": "bridge",
    "IPAM": {
      "Driver": "default",
      "Config": [
        {
          "Subnet": "10.0.3.0/24",
          "Gateway": "10.0.3.1"
        }
      ]
    },
    "Containers": {
      "c82afba13d9fdd12e41b1487af2911c327c2e3a55b7ee3cf4f6d2b8e1b33216c": {
        "Name": "worker",
        "EndpointID": "4de82d939fcb85a1277c9f2e57a8b4620c9d80ec5b0e238cbaf18ac3b108e2e2",
        "MacAddress": "02:42:0a:00:03:05",
        "IPv4Address": "10.0.3.5/24",
        "IPv6Address": ""
      },
      "f3e9ac1d77d5c1a819bf2fbc11c572bd450f06f3b491fe8ce1e8932a0c9d34ab": {
        "Name": "redis",
        "EndpointID": "59cfe7397c2a83c9a44c572cc3d2a511064d4716b69226638a85c0384bc4b9ef",
        "MacAddress": "02:42:0a:00:03:06",
        "IPv4Address": "10.0.3.6/24",
        "IPv6Address": ""
      },
      "stale-endpoint-12cf": {
        "Name": "api_old_1",
        "EndpointID": "",
        "MacAddress": "",
        "IPv4Address": "",
        "IPv6Address": ""
      }
    }
  }
]
```

```
PS C:\Users\mike> docker exec -it api sh -c "nslookup redis"
```

```
Server:      127.0.0.11
Address 1: 127.0.0.11
```

```
** server can't find redis: NXDOMAIN
```

```
PS C:\Users\mike> docker exec -it api sh -c "curl -v redis:6379"
* Trying 10.0.3.12:6379...
* connect to 10.0.3.12 port 6379 failed: Connection refused
* Failed to connect to redis port 6379 after 2 ms: Connection refused
* Closing connection 0
curl: (7) Failed to connect to redis port 6379: Connection refused
```

```
PS C:\Users\mike> docker compose up -d --force-recreate api worker
[+] Recreating containers
✓ Container myapp_worker   Recreated
1.4s
✓ Container myapp_api      Recreated
1.2s

PS C:\Users\mike> docker ps --format "table {{.Names}}\t{{.Status}}"
NAMES          STATUS
myapp_api      Up 5 seconds
myapp_worker   Up 4 seconds
```

```
PS C:\Users\mike> docker logs -f worker

2025-11-14T09:41:12Z   INFO   Worker starting...
2025-11-14T09:41:12Z   INFO   Loading configuration from /app/config/worker.json
2025-11-14T09:41:12Z   INFO   Connecting to Redis at redis://redis:6379
2025-11-14T09:41:12Z   INFO   Redis connection established successfully
2025-11-14T09:41:12Z   INFO   Worker initialisation complete
2025-11-14T09:41:13Z   INFO   Processing queued tasks...
2025-11-14T09:41:14Z   INFO   Task 4821 completed successfully
2025-11-14T09:41:15Z   INFO   Task 4822 completed successfully
```

```
PS C:\Users\mike> docker stats
CONTAINER    CPU %       MEM USAGE / LIMIT   MEM %      NET I/O      BLOCK I/O      PIDS
myapp_api    1.23%       82.4MiB / 512MiB    16.0%      8.1kB / 7.9kB  0B / 0B        12
myapp_worker 0.67%       54.2MiB / 512MiB    10.5%      12.7kB / 10.3kB 0B / 0B        9
redis        0.15%       10.3MiB / 100MiB    10.3%      5.1kB / 4.4kB  0B / 0B        4
```

NAMES	STATUS	IMAGE
api	Up 7 minutes	myapp/api:prod
worker	Restarting (1) 17 seconds ago	myapp/worker:prod
redis	Up 7 minutes	redis:7.2

```
2025-11-14T09:32:11Z INFO Request GET /reports/summary started
2025-11-14T09:32:11Z ERROR Redis connection timeout after 2000ms
2025-11-14T09:32:11Z WARN Falling back to stale cache data
2025-11-14T09:32:11Z INFO Request GET /reports/summary completed in 2412ms (status: 500)

2025-11-14T09:32:12Z INFO Request POST /tasks/queue started
2025-11-14T09:32:12Z ERROR Failed to enqueue task: Redis not reachable
2025-11-14T09:32:12Z ERROR Upstream error: redis:6379 connection refused
2025-11-14T09:32:12Z INFO Request POST /tasks/queue completed in 918ms (status: 503)

2025-11-14T09:32:13Z INFO Health check: starting
2025-11-14T09:32:15Z ERROR Health check failed: unable to ping Redis
2025-11-14T09:32:15Z WARN API degraded – retrying in 5s

2025-11-14T09:32:16Z INFO Request GET /status started
2025-11-14T09:32:16Z INFO Request GET /status completed in 3ms (status: 200)

2025-11-14T09:32:17Z INFO Request GET /reports/summary started
2025-11-14T09:32:17Z ERROR Redis connection timeout after 2000ms
2025-11-14T09:32:17Z INFO Request GET /reports/summary completed in 2311ms (status: 500)
```

```
2025-11-14T09:32:01Z INFO Worker starting...
2025-11-14T09:32:01Z INFO Loading configuration from /app/config/worker.json
2025-11-14T09:32:01Z INFO Connecting to Redis at redis://localhost:6379
2025-11-14T09:32:01Z ERROR Redis connection failed: dial tcp 127.0.0.1:6379:
connect: connection refused
2025-11-14T09:32:01Z WARN Retry scheduled in 2 seconds...

2025-11-14T09:32:03Z INFO Retrying Redis connection...
2025-11-14T09:32:03Z ERROR Redis connection failed: dial tcp 127.0.0.1:6379:
connect: connection refused

2025-11-14T09:32:03Z ERROR Worker initialisation failed: unable to connect to
Redis
2025-11-14T09:32:03Z INFO Shutting down worker...
2025-11-14T09:32:03Z INFO Exiting with status code 1
```

```
PS C:\Users\mike> docker exec -it worker sh
/app # pwd
/app
/app # ls -lah
total 20K
drwxr-xr-x    1 appuser appuser 4.0K Nov 14 09:20 .
drwxr-xr-x    1 root    root    4.0K Nov 14 09:20 ..
drwxr-xr-x    2 appuser appuser 4.0K Nov 14 09:20 config
drwxr-xr-x    3 appuser appuser 4.0K Nov 14 09:20 src
-rwxr-xr-x    1 appuser appuser 1.1K Nov 14 09:20 worker.sh

/app # env | grep REDIS
REDIS_HOST=localhost
REDIS_PORT=6379

/app # env | head
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
REDIS_HOST=localhost
REDIS_PORT=6379
NODE_ENV=production
APP_ENV=production
```

```
/app # ping redis
PING redis (10.0.3.12): 56 data bytes
64 bytes from 10.0.3.12: seq=0 ttl=64 time=185.7 ms
64 bytes from 10.0.3.12: seq=1 ttl=64 time=243.1 ms
Request timeout for icmp_seq 2
64 bytes from 10.0.3.12: seq=3 ttl=64 time=392.6 ms
Request timeout for icmp_seq 4
64 bytes from 10.0.3.12: seq=5 ttl=64 time=267.4 ms
^C
--- redis ping statistics ---
6 packets transmitted, 4 packets received, 33% packet loss
round-trip min/avg/max = 185.7/272.2/392.6 ms
```

```
/app # ls -lah /app/config
total 12K
drwxr-xr-x    2 appuser appuser 4.0K Nov 14 09:20 .
drwxr-xr-x    1 appuser appuser 4.0K Nov 14 09:20 ..
-rw-r--r--    1 appuser appuser 1.2K Nov 14 09:20 workerr.json
-rw-r--r--    1 appuser appuser 612 Nov 14 09:20 redis.json
```

```
PS C:\Users\mike> docker inspect --format='{{.Config.Image}}' worker  
myapp:latest
```

```
PS C:\Users\mike> docker history myapp:prod
```

IMAGE	CREATED	CREATED BY	SIZE
a92f1dbf0b33	2 days ago	COPY ./dev-config/ /app/config/	12MB
9b77e23bbbd0	2 days ago	RUN npm install	145MB
1c22bbde4411	2 days ago	COPY ./src /app/src	3MB
574e12cc9aa0	2 days ago	COPY ./package.json /app/	1KB
c42a1a0c7ad3	4 weeks ago	FROM node:18-alpine	5MB