Complete Docker Cheat Sheet (Basic to Advanced)

Basic Docker Commands

```
Build an image:
  docker build -t myimage .

Run a container:
  docker run -d --name mycontainer myimage

List containers/images:
  docker ps -a
  docker images

Stop/start/restart:
  docker stop mycontainer
  docker start mycontainer
  docker restart mycontainer

Remove container/image:
  docker rm mycontainer
  docker rmi myimage
```

Working with Volumes

```
Create and mount volume:
  docker volume create myvolume
  docker run -v myvolume:/app/data myimage

Bind mount:
  docker run -v $(pwd):/app myimage
```

Dockerfile Essentials

```
Sample Dockerfile:
  FROM python:3.10-slim
  WORKDIR /app
  COPY . .
  RUN pip install -r requirements.txt
  CMD ["python", "main.py"]

Best practices:
  - Use .dockerignore
  - Pin versions (e.g., python:3.10)
  - Combine RUN commands with && to reduce layers
```

Networking & Communication

Complete Docker Cheat Sheet (Basic to Advanced)

```
List networks:
  docker network ls

Create network:
  docker network create mynet

Run containers in same network:
  docker run --network=mynet --name app1 myimage

Inspect container IPs:
  docker inspect container_name | grep IPAddress
```

Docker Compose

```
Basic docker-compose.yml:
  version: '3'
  services:
    app:
      build: .
      ports:
        - "8000:8000"
      volumes:
        - .:/app
      depends_on:
        - db
    db:
      image: postgres
      environment:
        POSTGRES_PASSWORD: example

Commands:
  docker-compose up -d
  docker-compose down
  docker-compose logs -f
```

Logging and Debugging

```
View logs:
  docker logs container_name

Get inside a container:
  docker exec -it container_name /bin/bash

Follow logs:
  docker logs -f container_name

Restart policy:
  docker run --restart=always ...
```

Complete Docker Cheat Sheet (Basic to Advanced)

Advanced: Docker in Production

```
Multi-stage build:
  FROM node:18 AS builder
  WORKDIR /app
  COPY . .
  RUN npm run build

  FROM nginx:alpine
  COPY --from=builder /app/dist /usr/share/nginx/html

Healthcheck:
  HEALTHCHECK CMD curl --fail http://localhost:8000 || exit 1

Limit resources:
  docker run --memory="512m" --cpus="1.0" myimage
```

Docker Security Tips

```
- Use minimal base images (alpine, slim)
- Scan images: docker scan myimage
- Drop root: USER appuser
- Enable seccomp/apparmor profiles
- Keep images updated
```

Docker Cleanup & Maintenance

```
Remove stopped containers:
  docker container prune

Remove unused images/volumes:
  docker image prune -a
  docker volume prune

Clean everything:
  docker system prune -a
```

Docker Compose in Production

```
Best practices:
- Use .env files
- Separate dev and prod files (docker-compose.override.yml)
- Use named volumes for data persistence
- Monitor with Prometheus/Grafana

Deploy:
  docker-compose -f docker-compose.prod.yml up -d
```