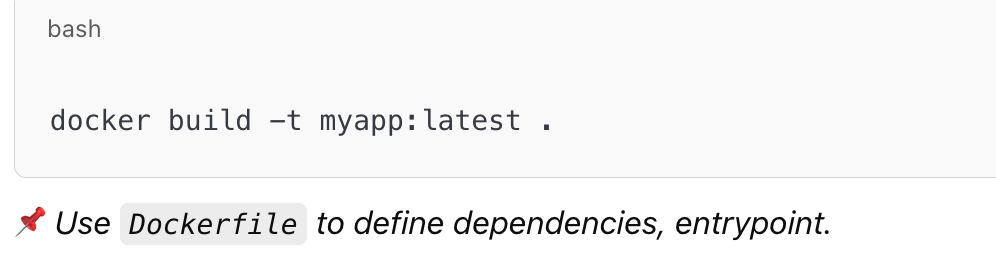
**Docker - Practices**



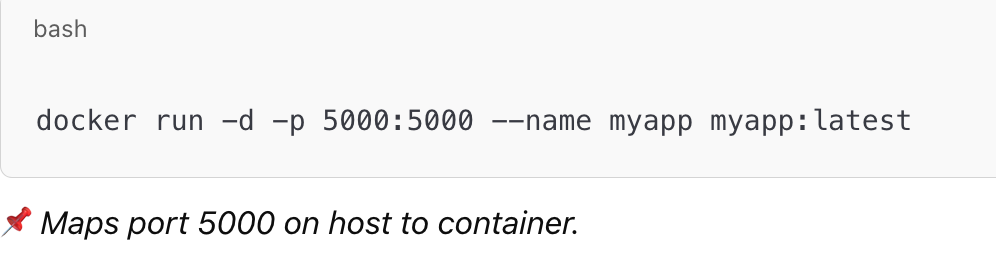
**1.Build Docker Image**

Build a custom image for an app (e.g., Flask, Node.js)



**2. Run Container**

Run the app container locally to test



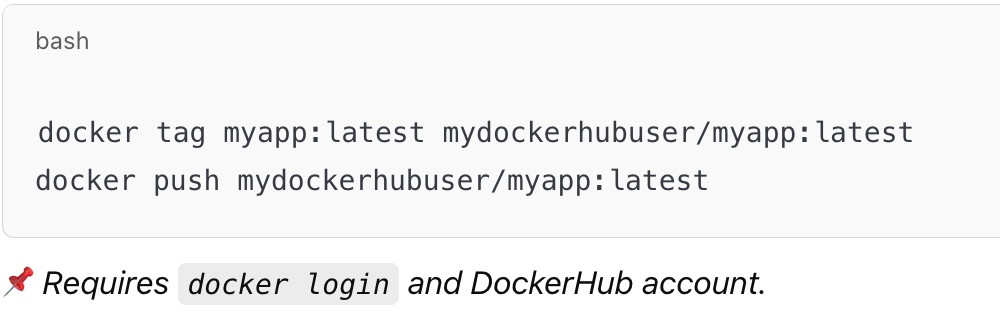
3. **Stop, Start, Remove Containers**

**Manage container lifecycle**

****

**4. Push Docker Image to DockerHub**

**Store and share image for CI/CD**

****

### **5. Create Multi-Stage Dockerfile**

**| | Minimize image size by separating build & runtime |**

****

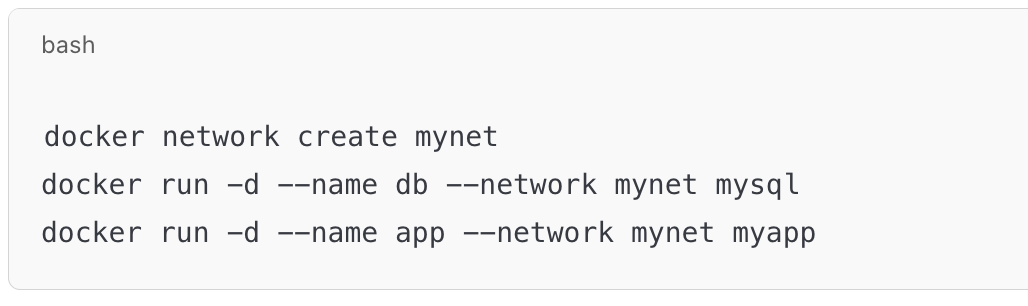
### **6. Use Docker Volumes**

**| Persist data (e.g., databases)**

****

### **7. Create and Use Docker Networks**

**Enable container-to-container communication**

****

### **8. Environment Variables in Container**

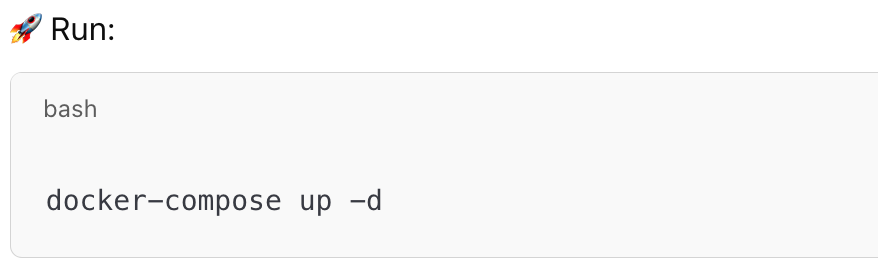
**Set secrets/config at runtime**

****

### **9. Docker Compose for Multi-Service Setup**

**Run multiple containers (e.g., app + db) easily**

****

****

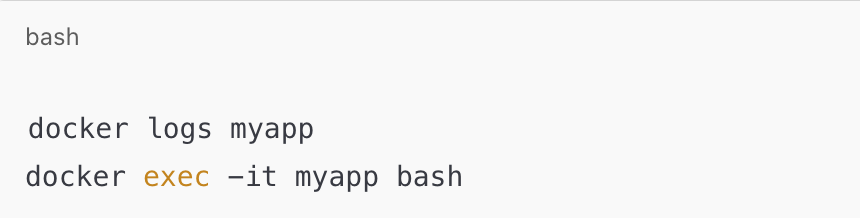
### **10. Dockerfile Optimization**

**Keep layers small, use .dockerignore**

* **Combine RUN instructions**
* **Use slim base images (python:3.9-slim)**
* **Clean up caches (apt-get clean && rm -rf /var/lib/apt/lists/\*)**

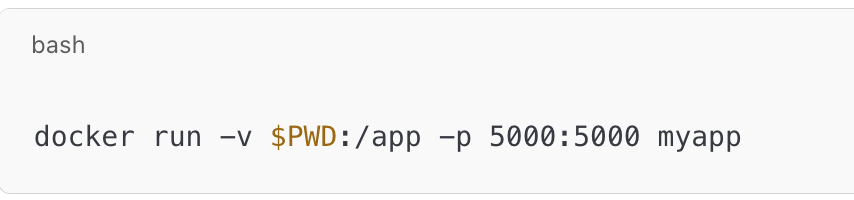
### **11. Inspect and Debug**

**Troubleshoot container issues**

****

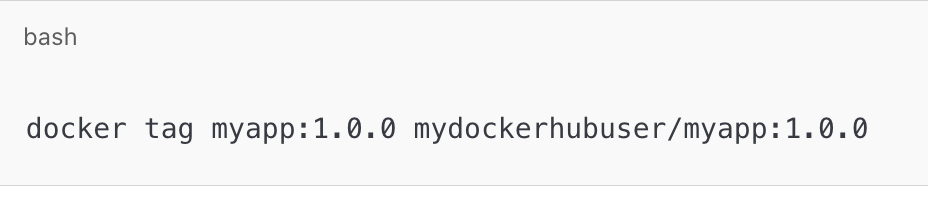
### **12. Bind Mounting Local Files**

**Live code reload without rebuilding**

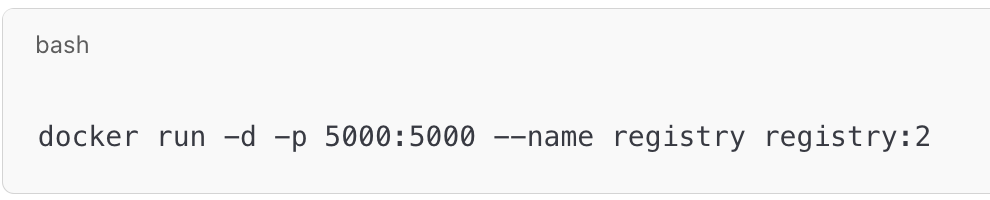
****

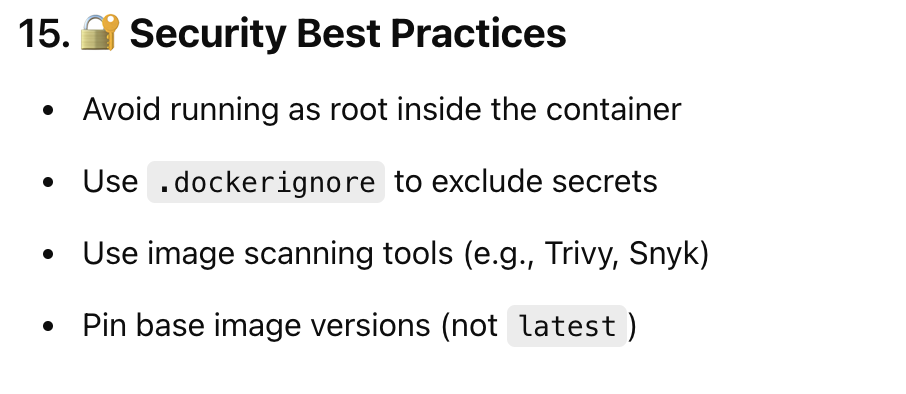
### **13. Tagging Strategy**

**Use version tags for traceability**

****

### **14. Create Private Docker Registry**

**Store internal images (optional but enterprise-friendly)**

****