Git Bash + Docker I/O Setup Guide (with Diagram)

Purpose

To enable smooth input/output (I/O) operations between Docker containers and the Windows host when operating from a Git Bash environment. This is especially critical for use cases like deploying machine learning pipelines (e.g., Raspberry Pi marker detection).

Key Constraints

- Git Bash **cannot safely parse `-v C:/...:/...`** in `docker run` commands (colon misinterpreted).
- **Use `docker-compose.yml` instead**: YAML interprets paths as plain strings.
- Avoid using `/mnt/c/...` or `$PWD` paths directly in Git Bash.
- Windows-style absolute paths like `C:/Users/...` are safe inside `docker-compose.yml`.

Folder Structure

```
docker-test/
├── docker-compose.yml
├── Dockerfile
├── output/   <-- Host folder to receive output
```

Dockerfile (example)

```
FROM python:3.11-slim
WORKDIR /app
```

docker-compose.yml

```
version: "3.9"

services:
  writer:
    build: .
    volumes:
      - "C:/Users/myname/Documents/docker-test/output:/app/output"
    command: sh -c "mkdir -p /app/output && echo hello > /app/output/hello.txt"
```

> This writes a test file (`hello.txt`) into the shared host folder.

Execution (Git Bash)

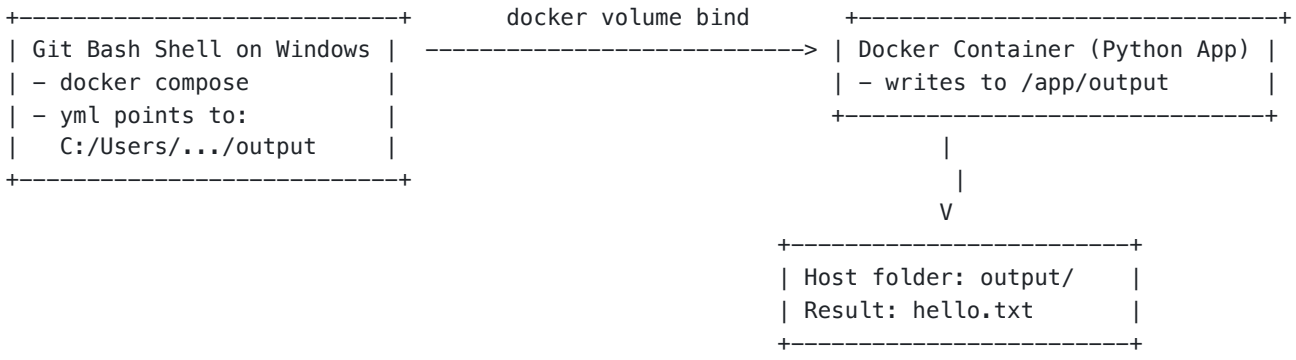
```
cd /c/Users/myname/Documents/docker-test

docker compose up --build
```

Success Verification

```
cat output/hello.txt
# Output should be: hello
```

Architecture Diagram (Docker Compose I/O)



Summary

- Avoid `docker run` with `-v` in Git Bash
- Use `docker-compose.yml` with absolute Windows paths
- Mount host folders explicitly for I/O
- Git Bash safely runs `docker compose up`
- Works perfectly for ML model output, image saving, etc.