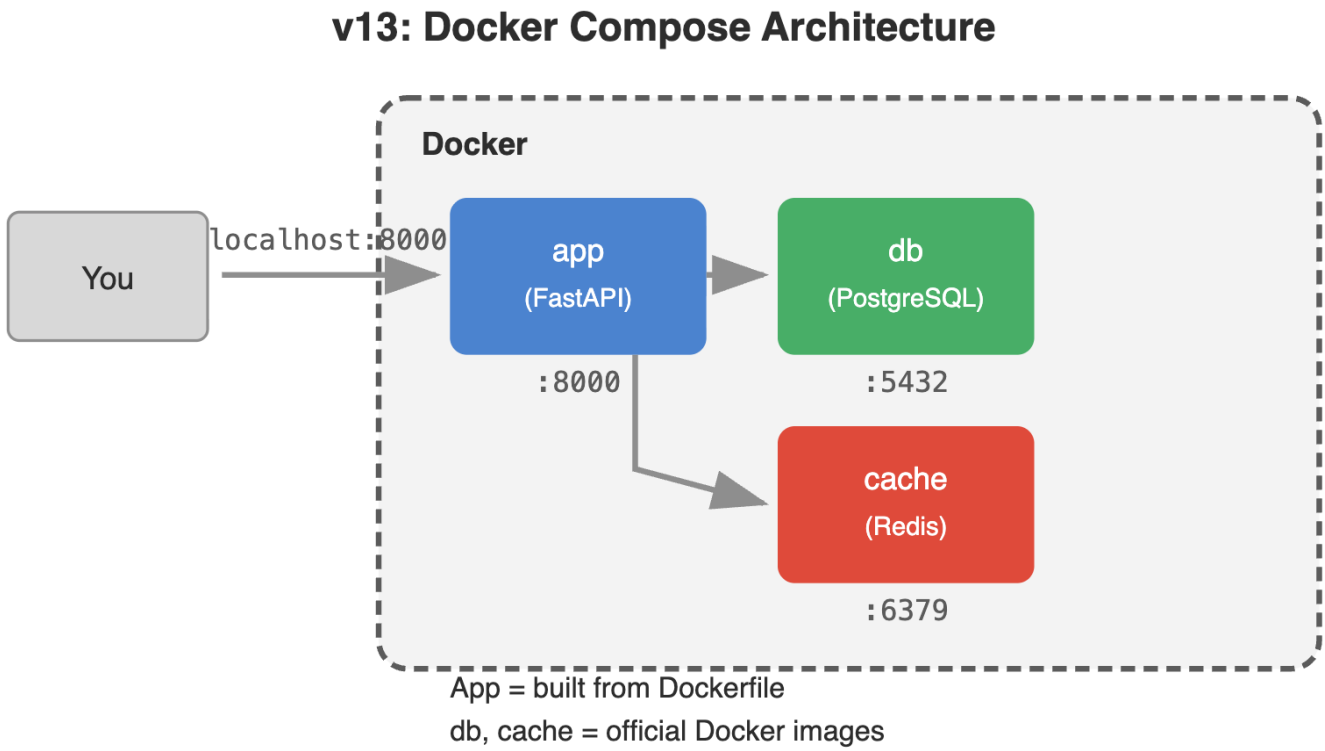
Docker Primer

What Docker Does in This Project

In v13, we containerize the entire application. Instead of installing PostgreSQL and Redis on your laptop, Docker runs them in isolated containers.

Our setup (docker-compose.yml):



Three services:

- `app` - Our FastAPI application (built from Dockerfile)
- `db` - PostgreSQL database (official image)
- `cache` - Redis cache (official image)

Why Docker?

- No need to install PostgreSQL or Redis locally
- Same environment for everyone
- One command to start everything
- Easy cleanup when done

Installation

Download Docker Desktop: https://www.docker.com/products/docker-desktop

Verify installation:

```
docker --version
docker-compose --version
```

Basic Commands

Images

```
docker images                  # list images
docker pull nginx              # download an image
docker rmi nginx               # remove an image
docker build -t myapp .        # build image from Dockerfile
```

Containers

```
docker ps                      # list running containers
docker ps -a                   # list all containers (including stopped)
docker run nginx               # run a container
docker run -d nginx            # run in background (detached)
docker run -p 8080:80 nginx    # map port 8080 (host) to 80 (container)
docker run --name web nginx    # give container a name
docker stop web                # stop a container
docker start web               # start a stopped container
docker rm web                  # remove a container
docker logs web                # view container logs
docker logs -f web             # follow logs (live)
docker exec -it web bash       # open shell inside container
```

Cleanup

```
docker stop $(docker ps -q)    # stop all running containers
docker rm $(docker ps -aq)     # remove all containers
docker system prune            # remove unused data
docker system prune -a         # remove everything unused
```

Docker Compose

For multi-container apps (like v13 with app + postgres + redis).

```
docker-compose up              # start all services
docker-compose up -d           # start in background
docker-compose up --build      # rebuild images and start
docker-compose down            # stop and remove containers
docker-compose logs            # view all logs
```

```
docker-compose logs app          # view logs for one service
docker-compose ps                # list running services
docker-compose exec app bash     # shell into a service
```

Running Containers Separately

Instead of docker-compose, you can run each container individually.

PostgreSQL

```
# Pull
docker pull postgres:15

# Run
docker run -d --name restbucks-db \
  -e POSTGRES_USER=postgres \
  -e POSTGRES_PASSWORD=postgres \
  -e POSTGRES_DB=restbucks \
  -p 5432:5432 \
  postgres:15

# Stop and remove
docker stop restbucks-db
docker rm restbucks-db
```

Redis

```
# Pull
docker pull redis:7

# Run
docker run -d --name restbucks-cache \
  -p 6379:6379 \
  redis:7

# Stop and remove
docker stop restbucks-cache
docker rm restbucks-cache
```

Application

```
# Build
docker build -t restbucks-app .

# Run (after db and cache are running)
docker run -d --name restbucks-app \
```

```
  -e
DATABASE_URL=postgresql://postgres:postgres@host.docker.internal:5432/rest
bucks \
  -e REDIS_URL=redis://host.docker.internal:6379/0 \
  -p 8000:8000 \
  restbucks-app

# Stop and remove
docker stop restbucks-app
docker rm restbucks-app
```

Note: `host.docker.internal` lets the app container reach services on your host machine.

Quick Reference for v13

```
# Start everything
docker-compose up --build

# In another terminal, test it
python test_client.py

# Stop everything
Ctrl+C
docker-compose down

# Clean slate (removes database data too)
docker-compose down -v
```

Common Issues

Port already in use:

```
docker-compose down              # stop existing containers
# or change port in docker-compose.yml
```

Container won't start:

```
docker-compose logs app          # check error messages
```

Database connection refused:

```
docker-compose down -v           # reset volumes
docker-compose up --build        # fresh start
```