ACADEMIND

Containers Are Always Great!

In Development	In Production
Isolated, standalone environment	Isolated, standalone environment
Reproducible environment, easy to share and use	Reproducible environment, easy to share and use

No surprises!
What works on your machine (in a container) will also work after deployment

ACADEMIND

Development to Production: Things To Watch Out For

Bind Mounts shouldn't be used in Production!

Containerized apps **might need a build step** (e.g. React apps)

Multi-Container projects might need to be **split** (or should be split) across multiple hosts / remote machines

Trade-offs between **control** and **responsibility** might be worth it!

ACADEMIND

Bind Mounts, Volumes & COPY

In Development

Containers should encapsulate the runtime environment but not necessarily the code

Use “Bind Mounts” to provide your local host project files to the running container

Allows for instant updates without restarting the container

In Production

Image / Container is the “single source of truth”

A container should really work standalone, you should NOT have source code on your remote machine

Use COPY to copy a code snapshot into the image

Ensures that every image runs without any extra, surrounding configuration or code

ACADEMIND

A Basic First Example: Standalone NodeJS App

Just NodeJS, no database, nothing else

1 Image & Container

Possible Deployment Approach

Install Docker on a remote host (e.g. via SSH), push and pull image, run container based on image on remote host

ACADEMIND

Hosting Providers

There are hundreds and thousands of Docker-supporting hosting providers out there!

Amazon Web Services

Microsoft Azure

Google Cloud

Example: Deploy to AWS EC2

AWS EC2 is a service that allows you to spin up and manage your own remote machines

1. Create and launch EC2 instance, VPC and security group
2. Configure security group to expose all required ports to WWW
3. Connect to instance (SSH), install Docker and run container

ACADEMIND

Deploy Source Code vs Image

Option 1: Deploy Source	Option 2: Deploy Built Image
Build image on remote machine	Build image before deployment (e.g. on local machine)
Push source code to remote machine, run `docker build` and then `docker run`	Just execute `docker run`
Unnecessary complexity	Avoid unnecessary remote server work

ACADEMIND

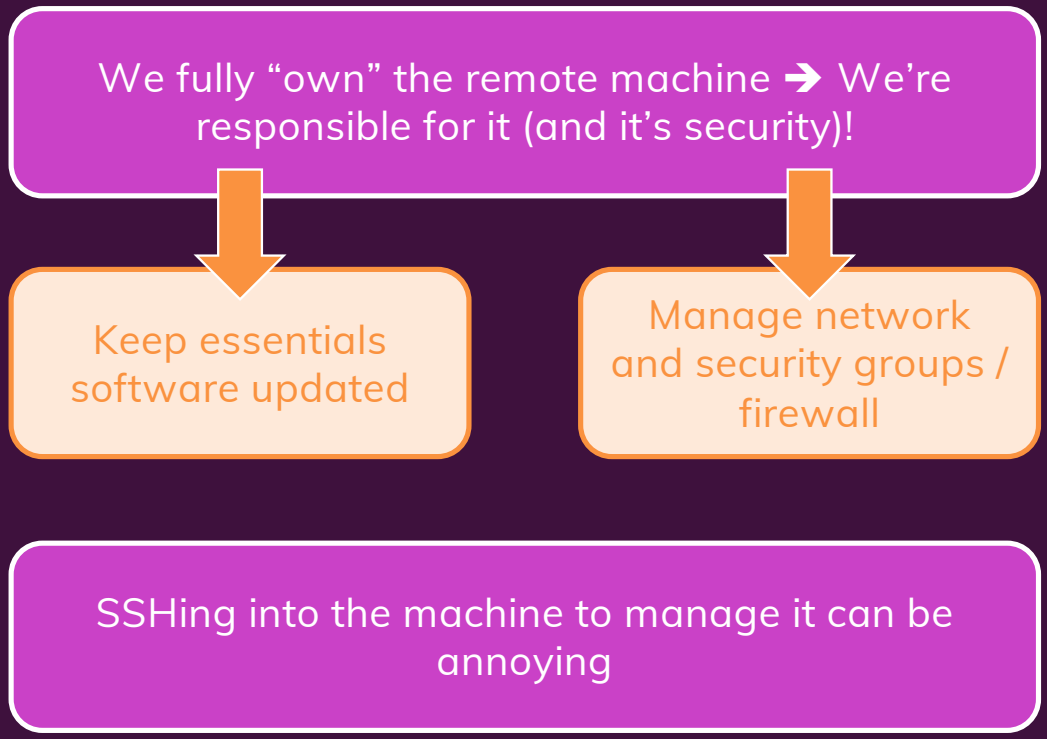
Docker Is Awesome!

Only Docker needs to be installed (no other runtimes or tools!)

Uploading our “code” is very easy

It’s the exact same app and environment as on our machine

“Do-it-yourself” Approach – Disadvantages



ACADEMIND

A Managed / Automated Approach

Your Own Remote Machines
e.g. AWS EC2

You need to create them, manage them, keep them updated, monitor them, scale them etc.

Great if you're an experienced admin / cloud expert

Managed Remote Machines
e.g. AWS ECS

Creation, management, updating is handled automatically, monitoring and scaling is simplified

Great if you simply want to deploy your app / containers

ACADEMIND

A Note about Databases

You can absolutely manage your own Database containers

but ...

Scaling & managing **availability** can be challenging

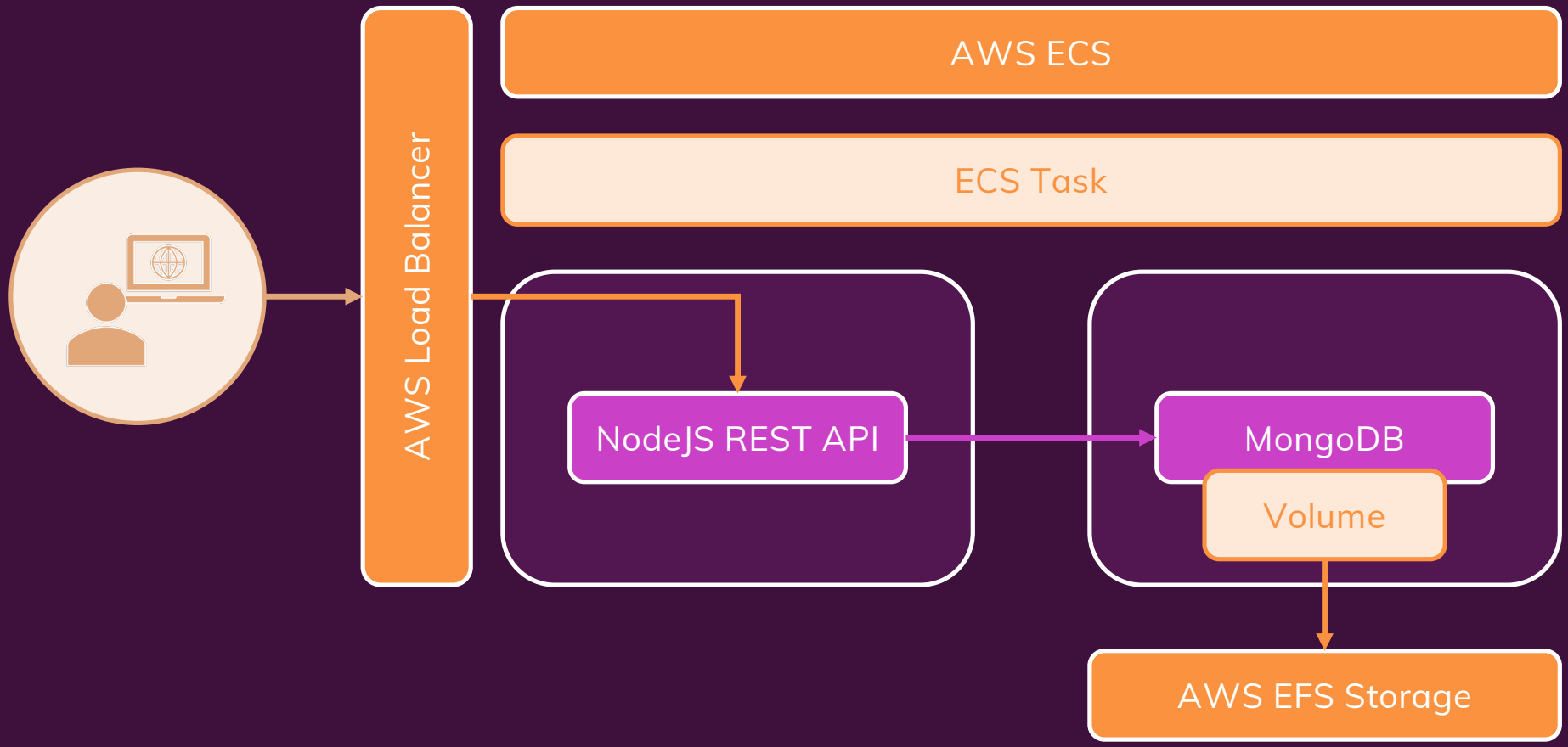
Performance (also during traffic spikes) could be bad

Taking care about **backups** and **security** can be challenging

Consider using a **managed Database service** (e.g. *AWS RDS*, *MongoDB Atlas*, ...)

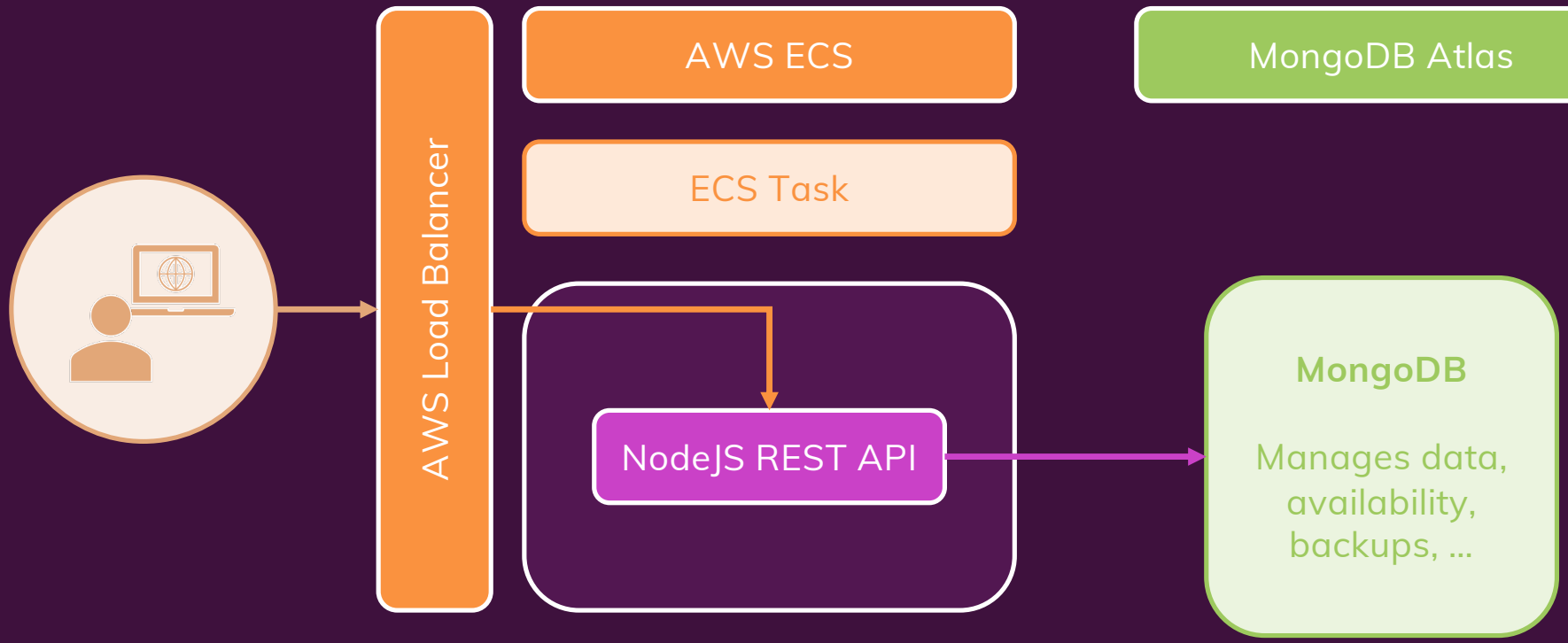
ACADEMIND

Our Current App Architecture



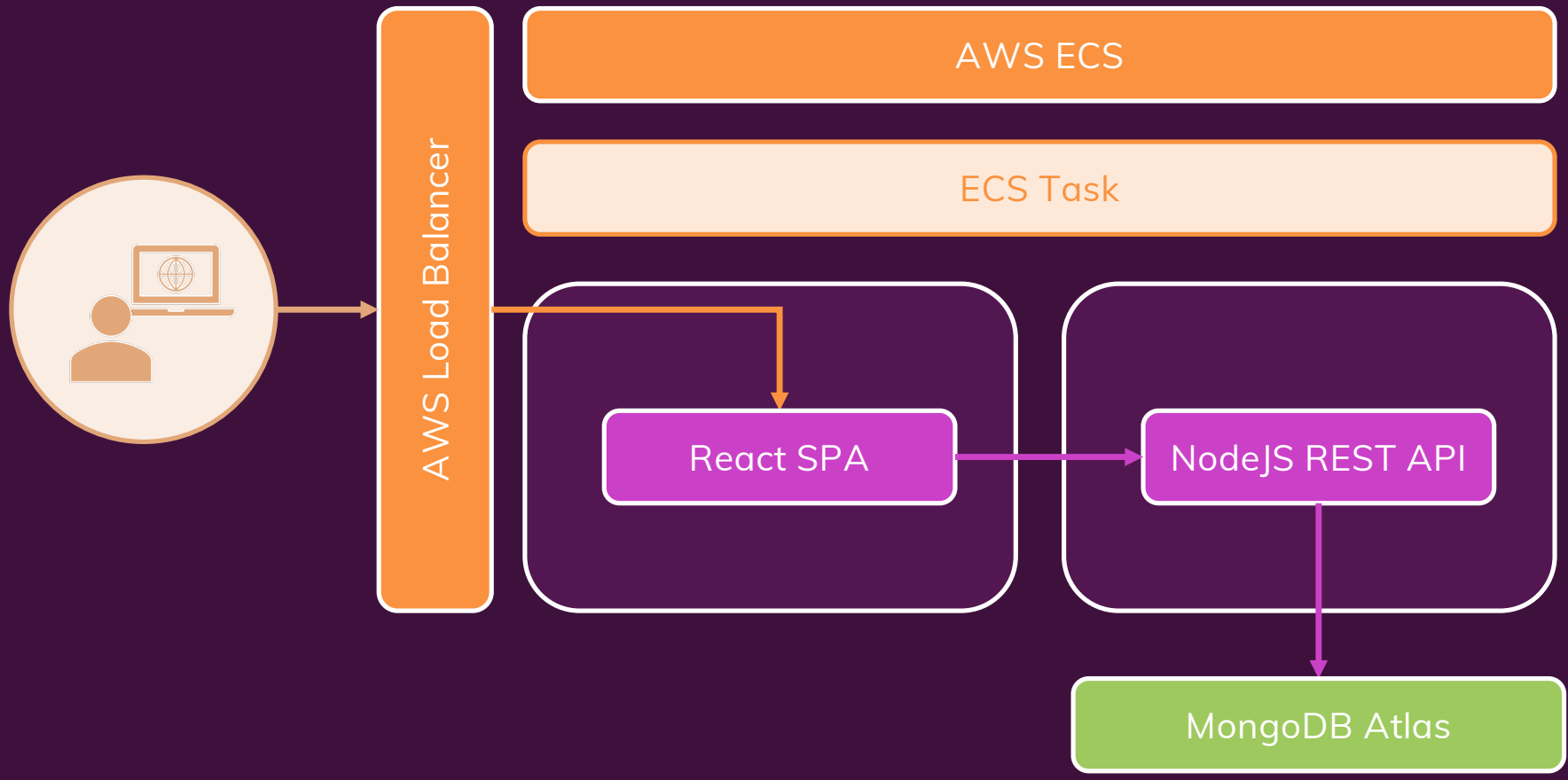
ACADEMIND

Our New App Architecture



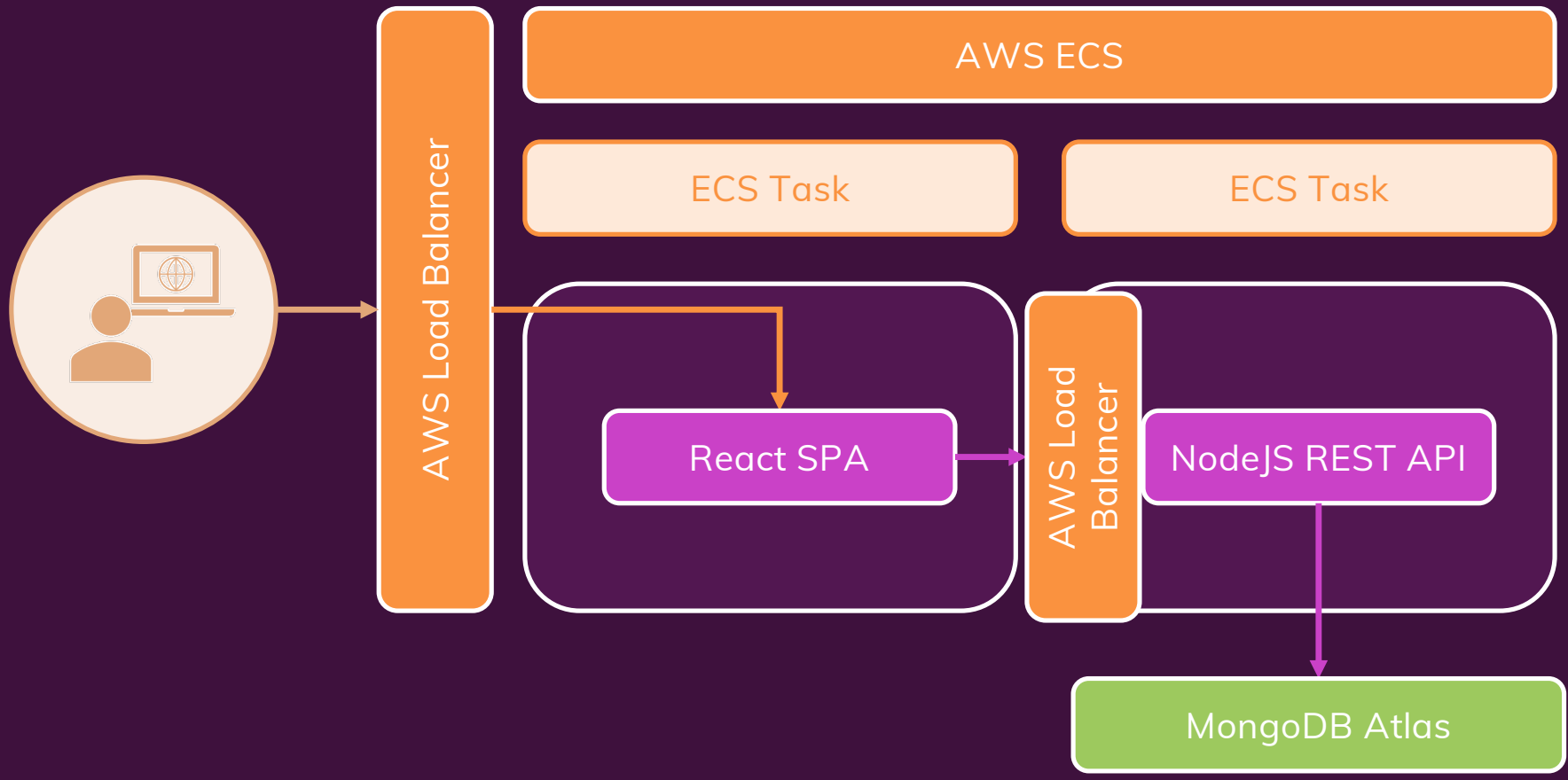
ACADEMIND

Our Final App Architecture



ACADEMIND

Our Final App Architecture



ACADEMIND

Apps with Development Servers & Build Steps

Some apps / projects require a build step

e.g. optimization script that needs to be executed ***AFTER development*** but ***BEFORE deployment***

Development Setup ← IS NOT EQUAL TO → Production Setup

(not entirely)

e.g. ReactJS App

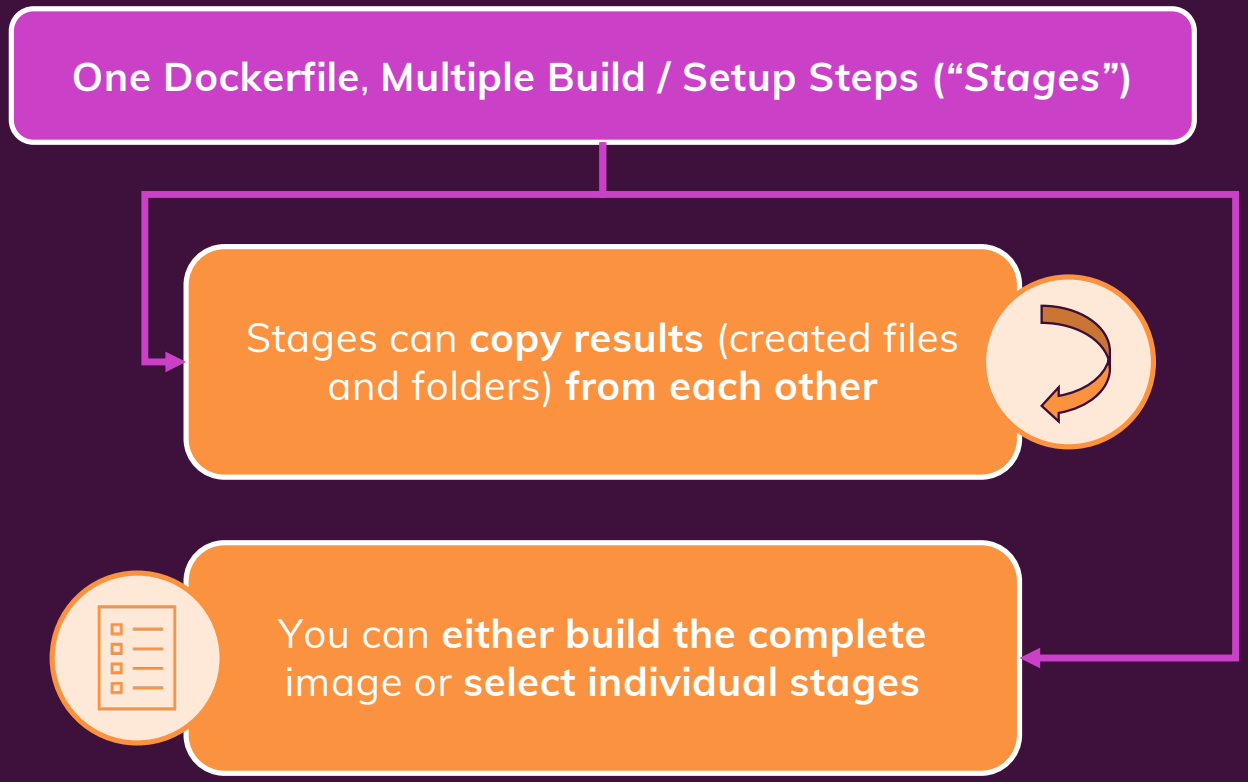
Uses live-reloading development server, uses unoptimized / unsupported JS features

Build Step / Script →

No attached server, optimized, fully browser-compatible code

ACADEMIND

Introducing Multi-Stage Builds



ACADEMIND

From AWS To Other Providers

AWS was just the example provider in this section!

Manual installation of Docker + management of the underlying server

Possible with any provider that gives you fully controlled remote hosts / instances

Managed Container/ Docker services like ECS

Many cloud providers offer managed Docker / Container services

ACADEMIND

Can We Do Better?

Containers allow us **to encapsulate app code and environment** for both **development and production**

Thinking about production forces us to build containers / app code with **more scenarios** in mind (e.g. multi-stage builds)

If we DON'T manage Docker and remote machines manually, we must work with the **tools and rules imposed by the managed service**

Different cloud providers == Different rules

Depending on provider, **features** like load balancing might be **challenging** to implement

Kubernetes