

 @RYANWASKIEWICZ

 RWASKIEWICZ

12 FACTOR (FRONTEND) APPS BY

DOCKERIZATION

HOW DO YOUR DEPLOYMENTS MAKE YOU FEEL?



No Pain

Hurts a Little

Hurts even
more

Hurts a lot

Hurts as much
as possible

THE 12 FACTOR APP

- ▶ Use **declarative** formats for setup automation
- ▶ Have a **clean contract** with the underlying OS to offer **maximum portability**
- ▶ **Minimize divergence** between dev and prod
- ▶ **Scale up** without significant changes

THE 12 FACTORS

- ▶ Codebase
- ▶ Dependencies
- ▶ Configuration
- ▶ Backing Services
- ▶ Build, Release, Run
- ▶ Processes
- ▶ Port Binding
- ▶ Concurrency
- ▶ Disposability
- ▶ Dev/Prod Parity
- ▶ Logs
- ▶ Admin Processes

THE 12 FACTORS

- ▶ Codebase
- ▶ Dependencies
- ▶ **Configuration**
- ▶ Backing Services
- ▶ **Build, Release, Run**
- ▶ Processes
- ▶ Port Binding
- ▶ Concurrency
- ▶ **Disposability**
- ▶ **Dev/Prod Parity**
- ▶ Logs
- ▶ Admin Processes

[HTTPS://WWW.DOCKER.COM/](https://www.docker.com/)



docker

DOCKERFILES

- build
- config
- node_modules
- public
- scripts
- src
- ◆ .gitignore
- 🚢 **Dockerfile**
- { } package.json
- 📄 README.md
- 📄 yarn.lock

```
# Set a 'Base Image' as the
# foundation
FROM nginx:alpine
LABEL maintainer="Ryan Waskiewicz".

# Copy our built application to a
# place NGINX can serve it
COPY build/ /usr/share/nginx/html

# Define port that our container
# will listen to at runtime
EXPOSE 80

# Start NGINX in the container
CMD ["nginx", "-g", "daemon off;"]
```

DOCKERFILE → IMAGE → CONTAINER

- ▶ Our Dockerfile creates a 'Docker Image'

```
$ docker build -t puzzle-app .
```

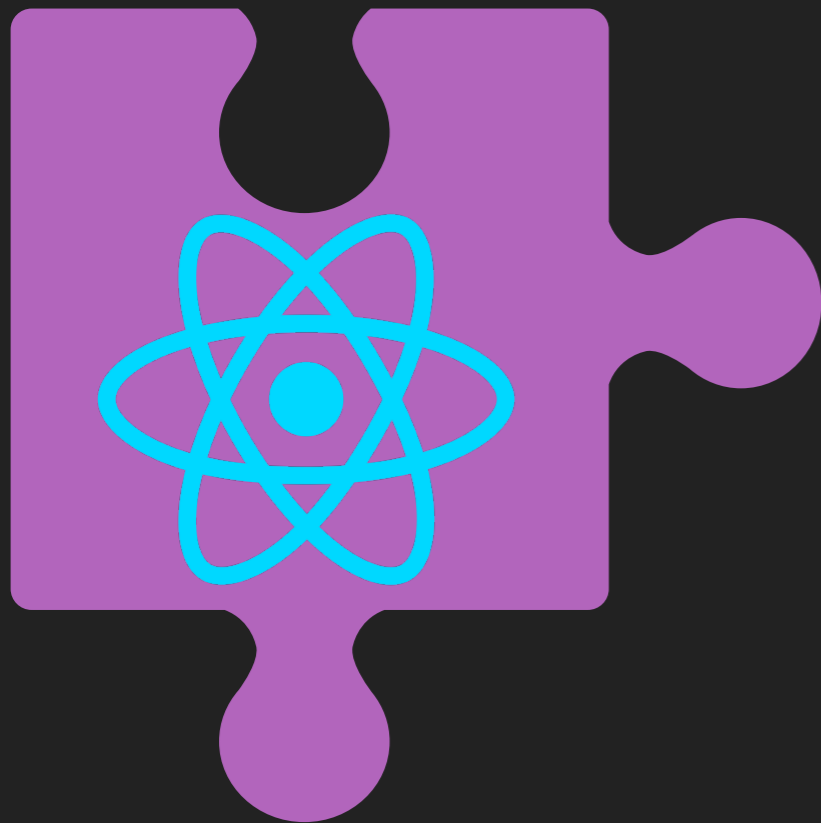
- ▶ That *Image* can be shared with the development community, colleagues
- ▶ From a single *Image* one or more *Containers* can be created

```
$ docker run puzzle-app
```

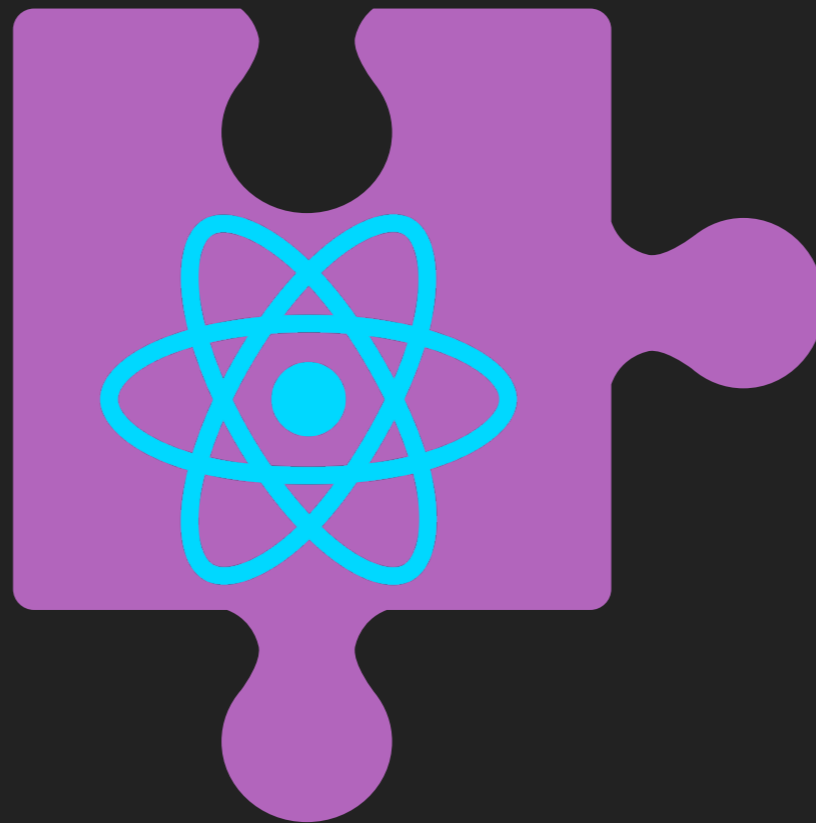


```
$ docker run puzzle-app
```

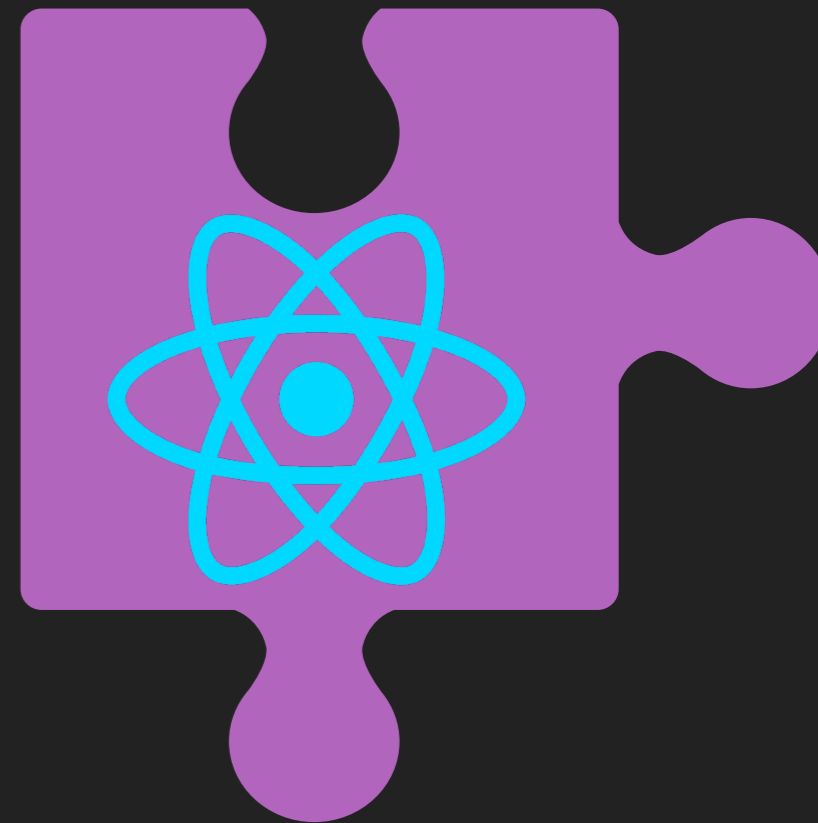
MY LOCAL



EVIE'S MACHINE



TIM'S MACHINE



THE 12 FACTOR APP

- ▶ Use **declarative** formats for setup automation
- ▶ Have a **clean contract** with the underlying OS to offer **maximum portability**
- ▶ **Minimize divergence** between dev and prod
- ▶ **Scale up** without significant changes

```
# Set a 'Base Image' as the
# Foundation of our App
FROM nginx:alpine
LABEL maintainer="Ryan Waskiewicz".

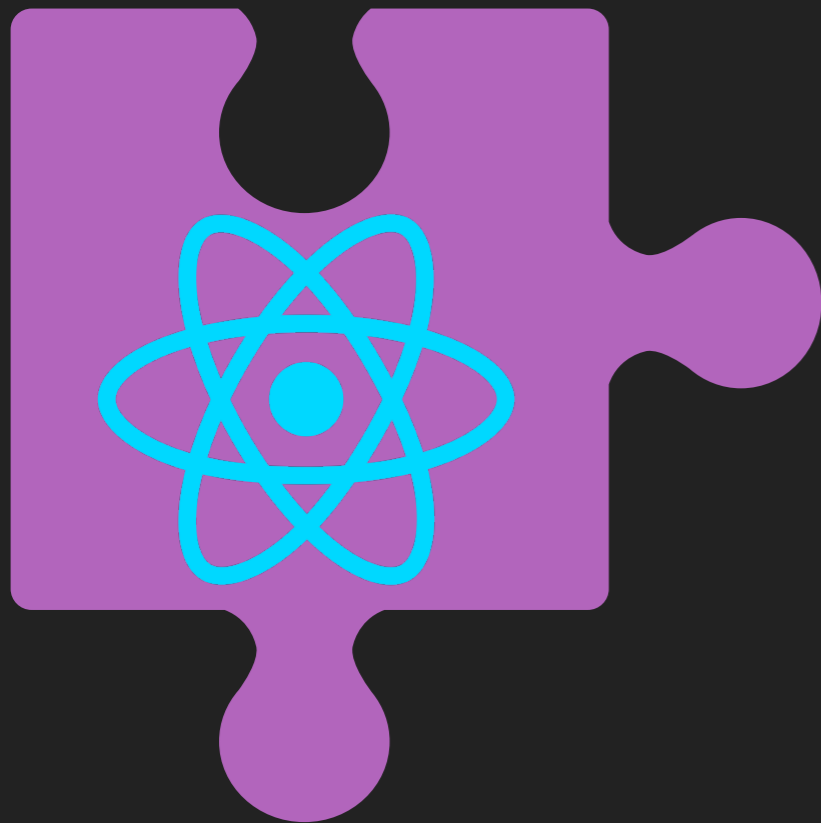
# Copy our built application to a
# place NGINX can serve it
COPY dist/ /usr/share/nginx/html

# Define port that our container
# will listen to at runtime
EXPOSE 80

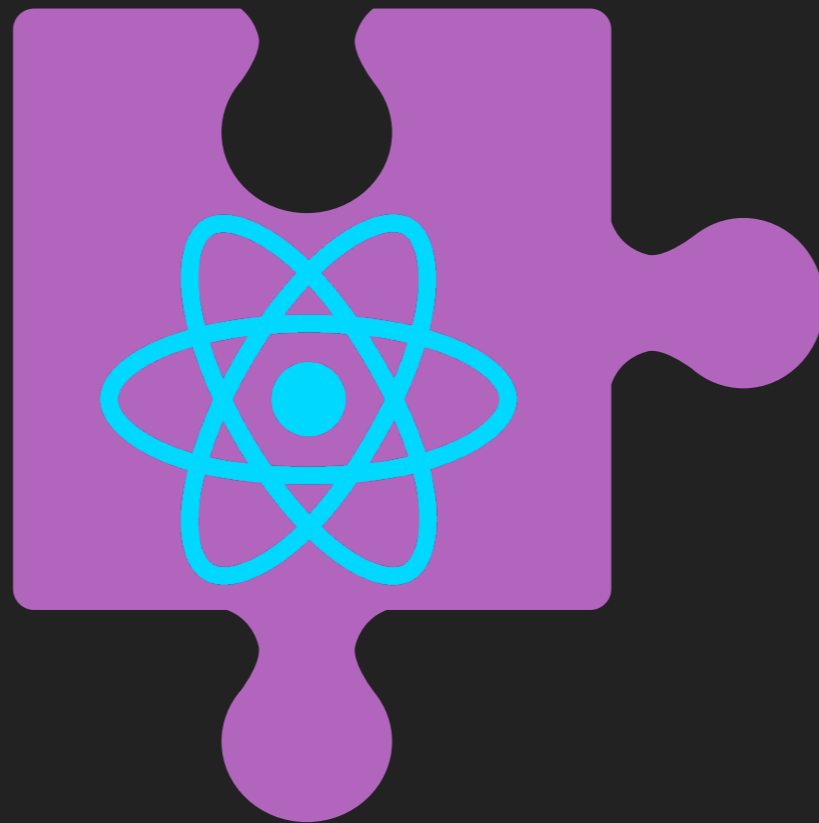
# Start NGINX in the container
CMD ["nginx", "-g", "daemon off;"]
```

```
$ docker run puzzle-app
```

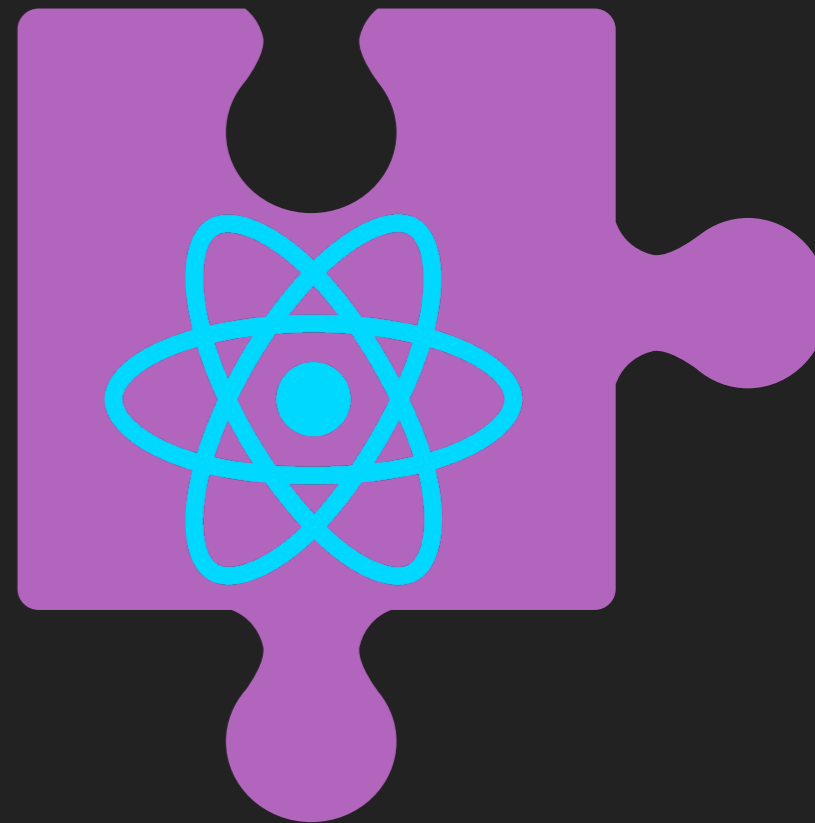
MY LOCAL



EVIE'S MACHINE



TIM'S MACHINE



```
$ docker run puzzle-app
```

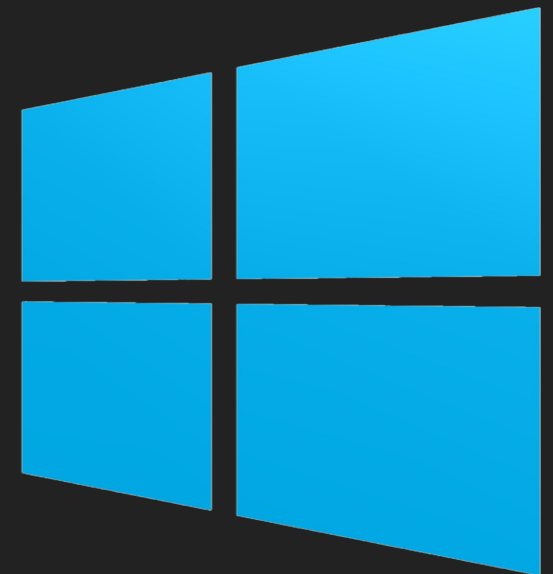
MY LOCAL



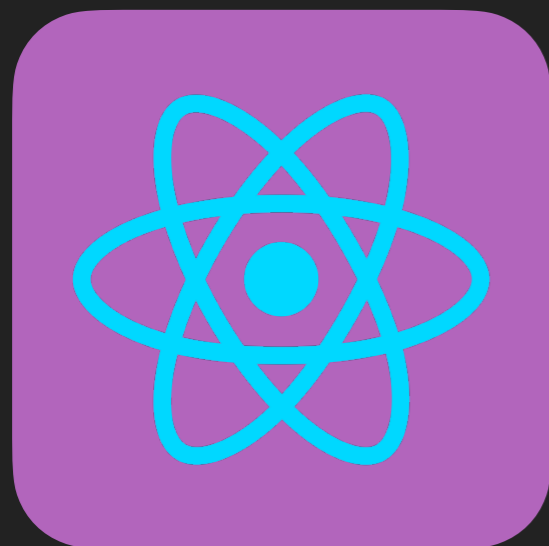
EVIE'S MACHINE



TIM'S MACHINE



MY LOCAL

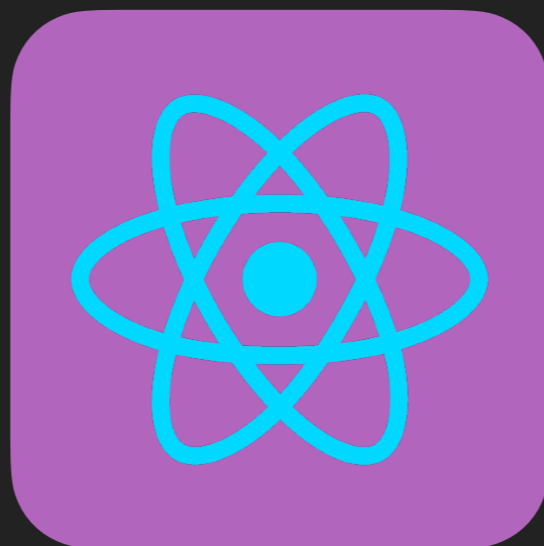


DOCKER ENGINE



INFRASTRUCTURE

EVIE'S MACHINE

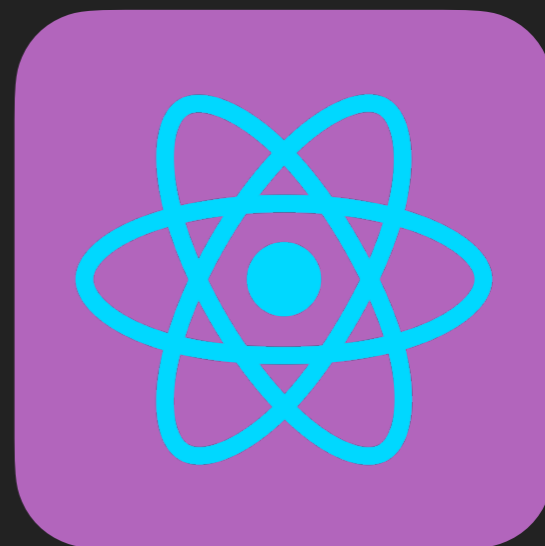


DOCKER ENGINE

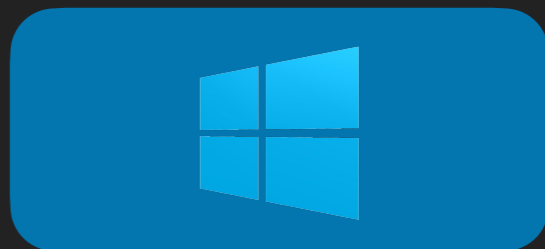


INFRASTRUCTURE

TIM'S MACHINE



DOCKER ENGINE



INFRASTRUCTURE

THE 12 FACTOR APP

- ▶ Use **declarative** formats for setup automation
- ▶ Have a **clean contract** with the underlying OS to offer **maximum portability**
- ▶ **Minimize divergence** between dev and prod
- ▶ **Scale up** without significant changes

CONFIGURATION

CONFIGURATION

Resources handles, credentials to external services, per-deploy values are stored in the environment, not the code.

WHY?

LITMUS TEST

If your codebase were made public today, how many passwords, internal URLs, or other 'secrets' would be exposed?

CONFIGURATION

- build
- config
- node_modules
- public
- scripts
- src
- ⚙️ **.env**
- 📄 .gitignore
- 🐳 Dockerfile
- {} package.json
- 📄 README.md
- 🐶 yarn.lock

```
⚙️ .env x
-----
⚙️ .env
1  REACT_APP_PUPPY_API_URL=https://omg-puppies.dev.foo
```

- ▶ Create a .env file for each environment
- ▶ Allows for granular control of application variables

CONFIGURATION

```
async getPuppyDetails() {
  let puppyApiUrl = '';

  if (process.ENV === 'production') {
    puppyApiUrl = 'https://omg-puppies.foo';
  } else if (process.ENV === 'stage') {
    puppyApiUrl = 'https://omg-puppies.stg.foo';
  } else if (process.ENV === 'test') {
    puppyApiUrl = 'https://omg-puppies.test.foo';
  } else if (process.ENV === 'development') {
    puppyApiUrl = 'https://omg-puppies.dev.foo';
  } else {
    throw new Error(`Some undefined environment was found: ${process.ENV}`);
  }

  return await axios.get(puppyApiUrl);
}
```

```
async getPuppyDetails() {
  return await axios.get(`${process.env.REACT_APP_PUPPY_API_URL}`);
}
```

**BUILD, RELEASE,
RUN**

BUILD, RELEASE, RUN

A codebase is transformed into a deploy through 3 stages:

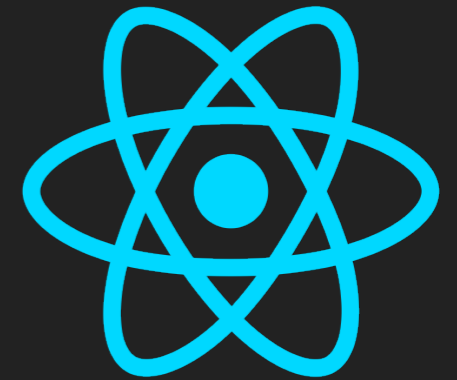
1) Build - Converting code into a executable bundle

2) Release - Combining executable bundle with config

3) Run - Run the application in the execution environment

BUILD, RELEASE, RUN

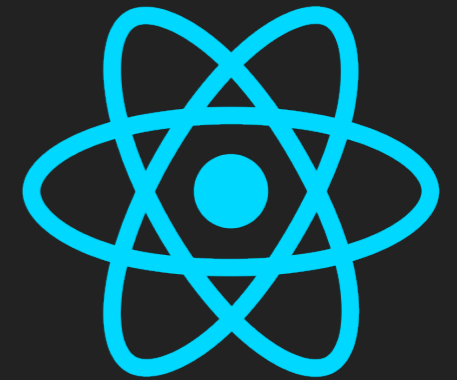
- ▶ Case Study: Create React App
- ▶ Uses Webpack for bundling application
- ▶ Scans filesystem for .env* files
- ▶ Expands them in place in JS



```
async getPuppyDetails() {  
  return await axios.get(`${process.env.REACT_APP_PUPPY_API_URL}`);  
}
```

BUILD, RELEASE, RUN

- ▶ Case Study: Create React App
- ▶ Uses Webpack for bundling application
- ▶ Scans filesystem for .env* files
- ▶ Expands them in place in JS



```
async getPuppyDetails() {  
  return await axios.get(`https://omg-puppies.dev.foo`);  
}
```

DEV/PROD PARITY

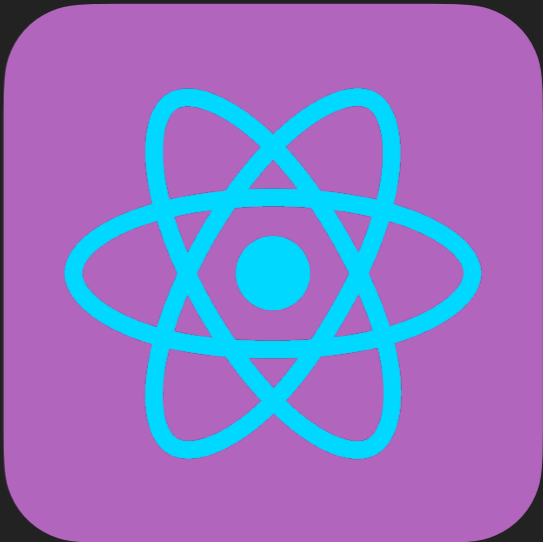
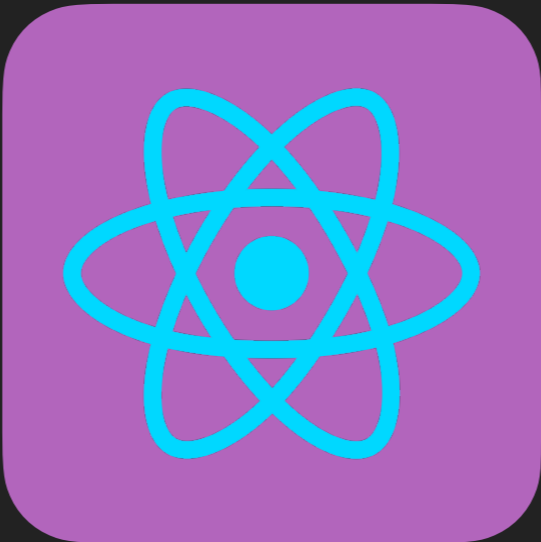
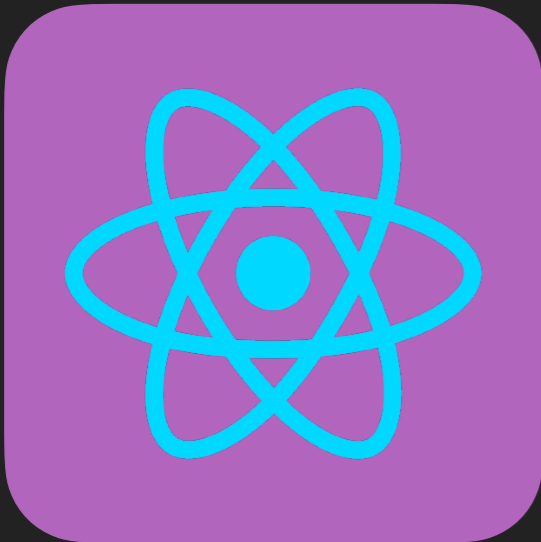
DEV/PROD PARITY

Keep development, staging, production, etc. as similar as possible

LOCAL

DEV

PRODUCTION



DOCKER ENGINE

DOCKER ENGINE

DOCKER ENGINE

HOST OPERATING SYSTEM

HOST OPERATING SYSTEM

HOST OPERATING SYSTEM

INFRASTRUCTURE

INFRASTRUCTURE

INFRASTRUCTURE

THE 12 FACTOR APP

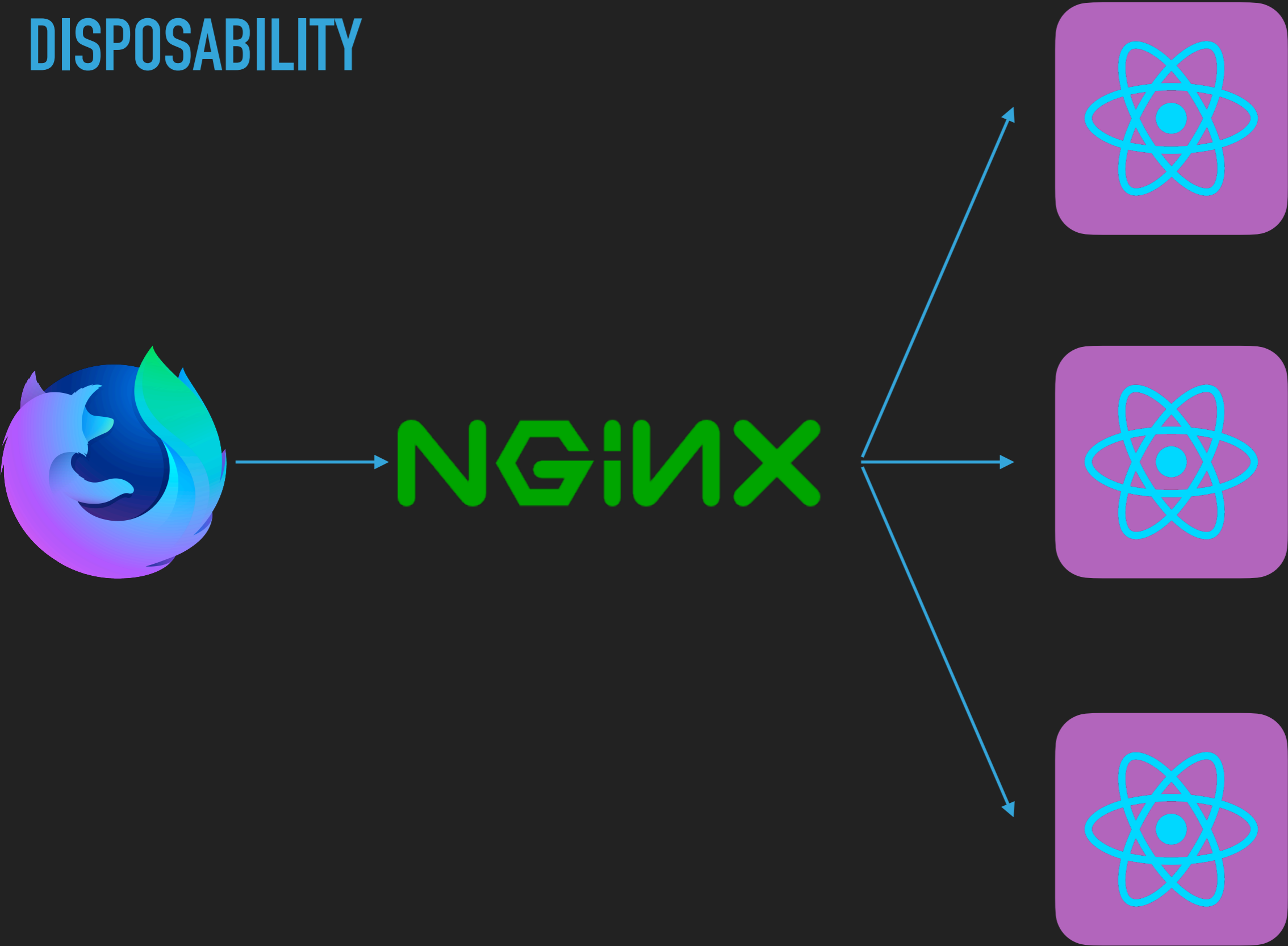
- ▶ Use **declarative** formats for setup automation
- ▶ Have a **clean contract** with the underlying OS to offer **maximum portability**
- ▶ **Minimize divergence** between dev and prod
 - ▶ Configuration, Build Release Run, Dev/Prod Parity
- ▶ **Scale up** without significant changes

DISPOSABILITY

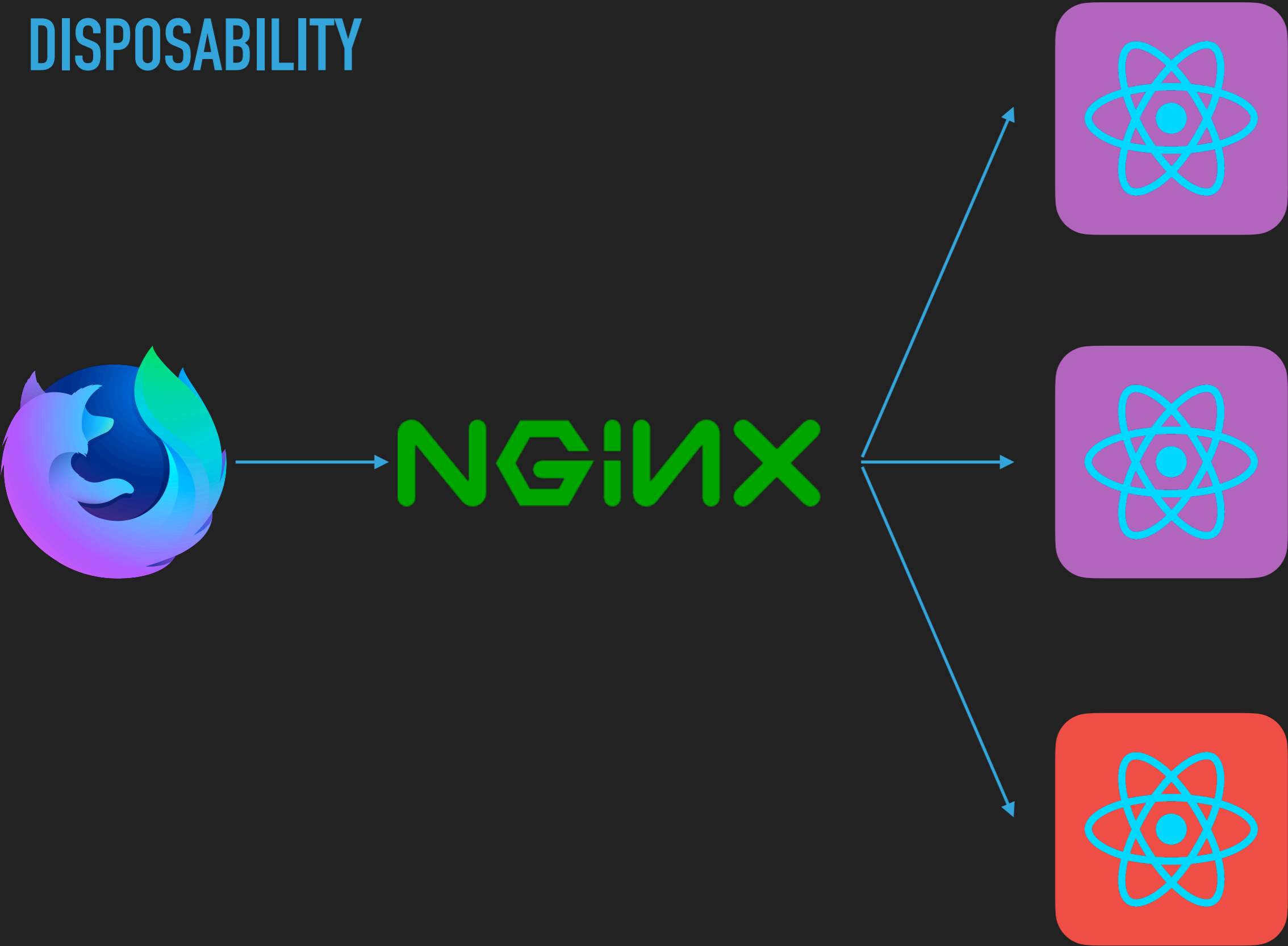
DISPOSABILITY

Processes can be started or stopped at a moment's notice

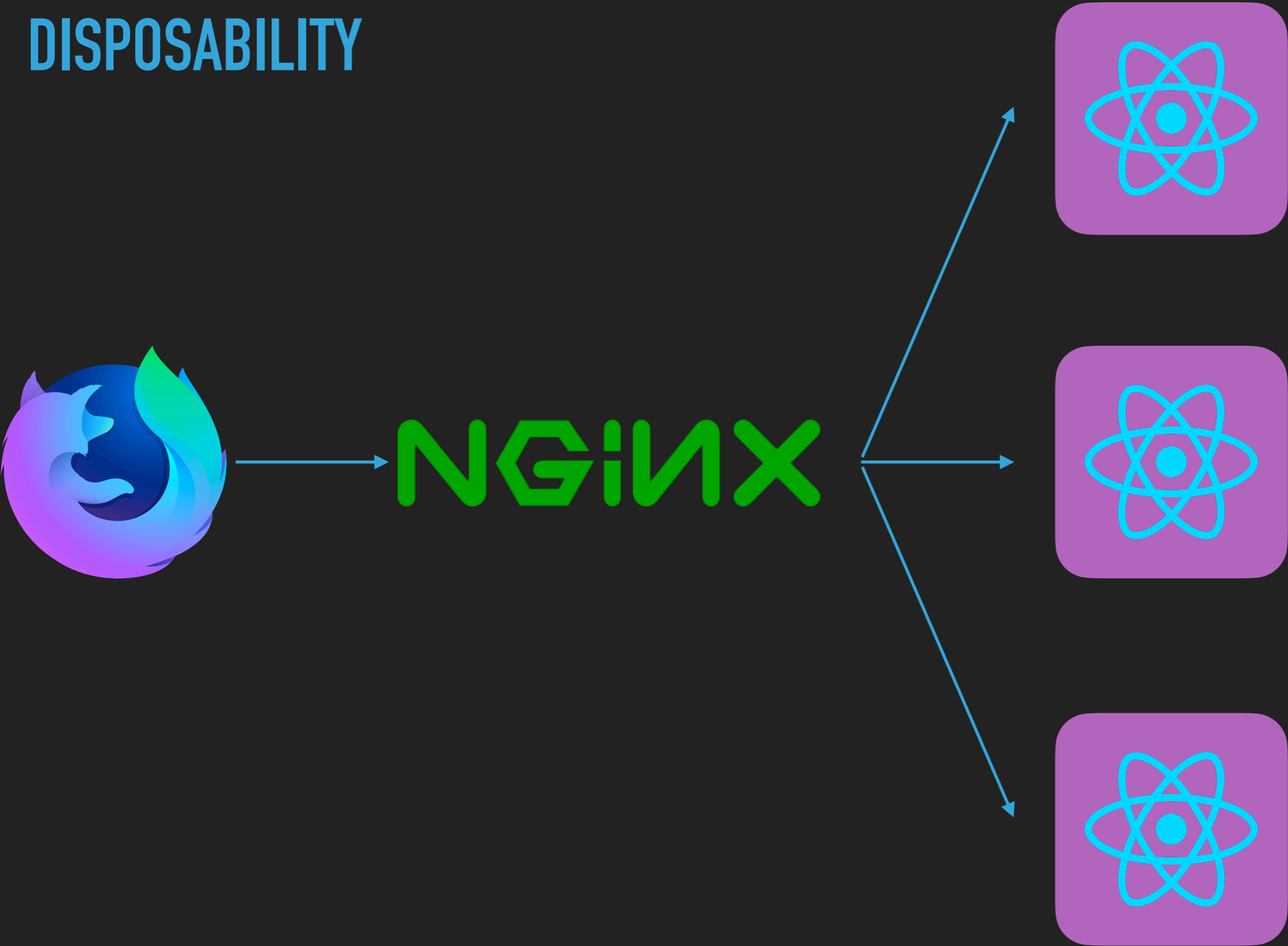
DISPOSABILITY



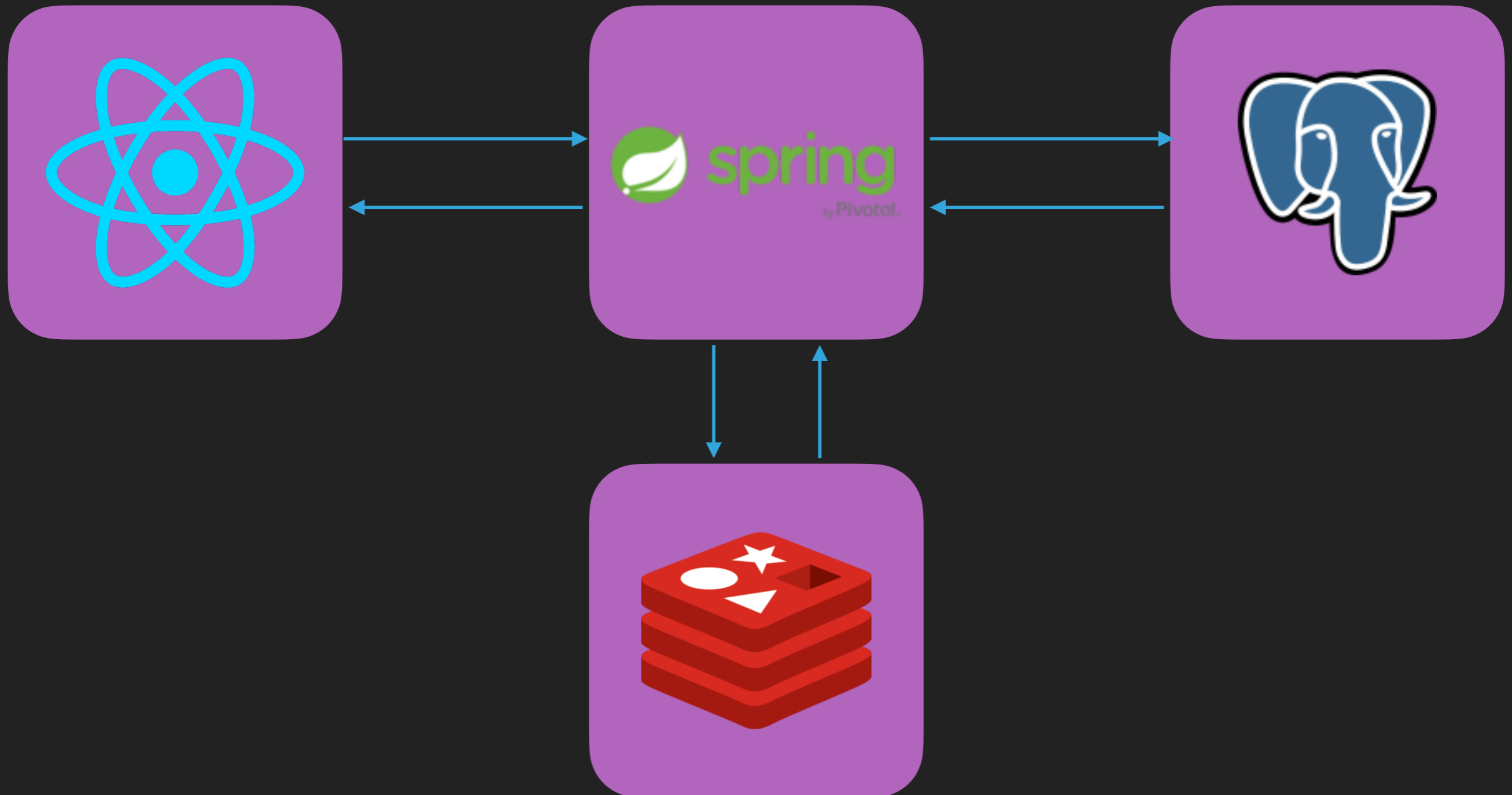
DISPOSABILITY



DISPOSABILITY



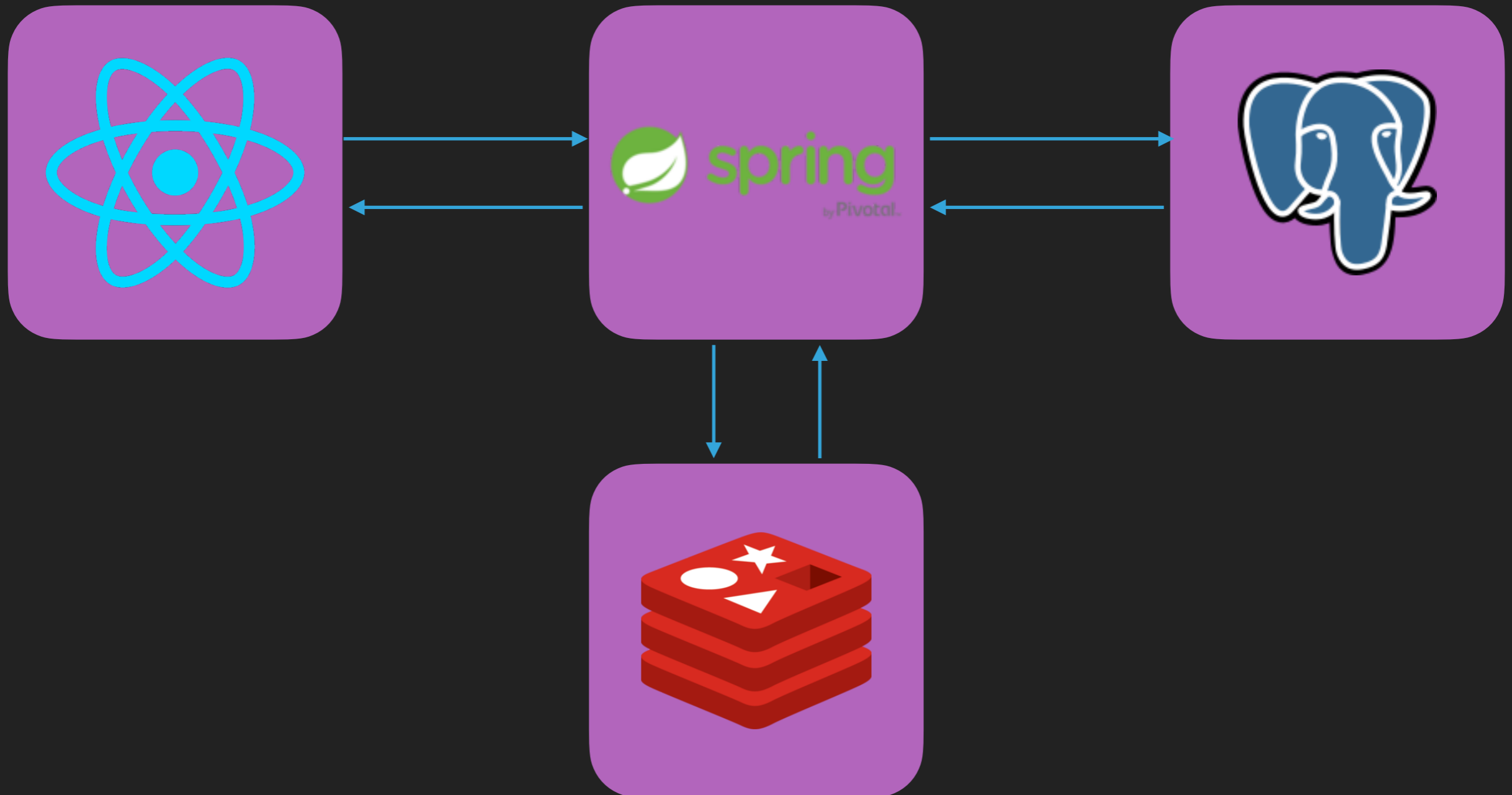
DISPOSABILITY - TESTING



[HTTPS://12FACTOR.NET/DISPOSABILITY](https://12factor.net/disposability)

DISPOSABILITY - TESTING


DISPOSABILITY - TESTING



THE 12 FACTOR APP

- ▶ Have a **clean contract** with the underlying OS to offer **maximum portability**
- ▶ Use **declarative** formats for setup automation
- ▶ **Minimize divergence** between dev and prod
- ▶ **Scale up** without significant changes

THANK YOU

- ▶ The 12 Factor App: <https://12factor.net/>
- ▶ Docker: <https://docs.docker.com/get-started/>
- ▶ This talk:  @rwaskiewicz