

CLOUD NATIVE DEVELOPMENT

Charles Moulliard

May 17th - Riviera Dev

<https://goo.gl/kgxCRI>



WHO

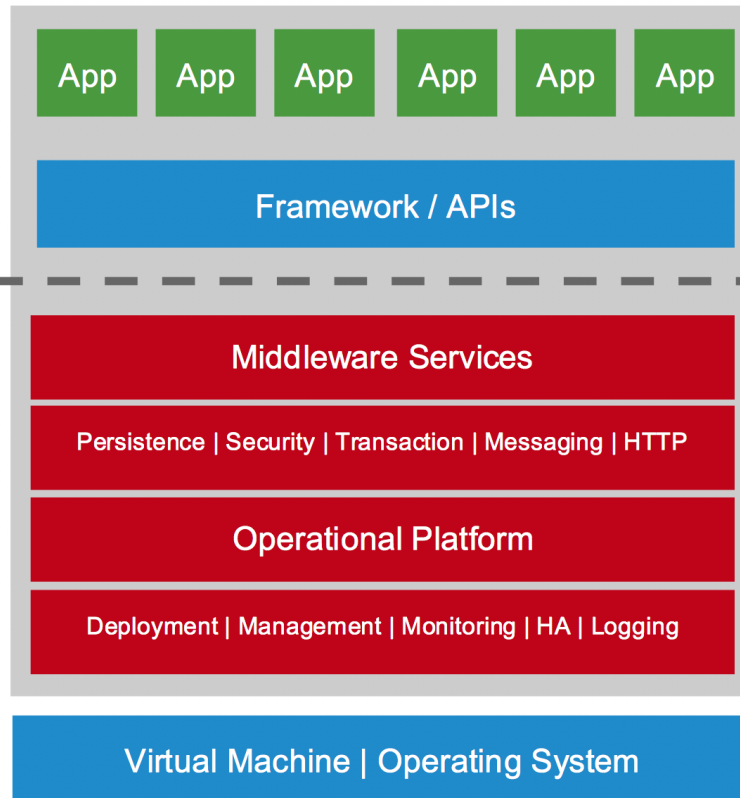
- Charles Moulliard
- Software Eng. Manager (SpringBoot)
- Technology evangelist
- Twitter: @cmoulliard
- Email: cmoulliard@redhat.com



AGENDA

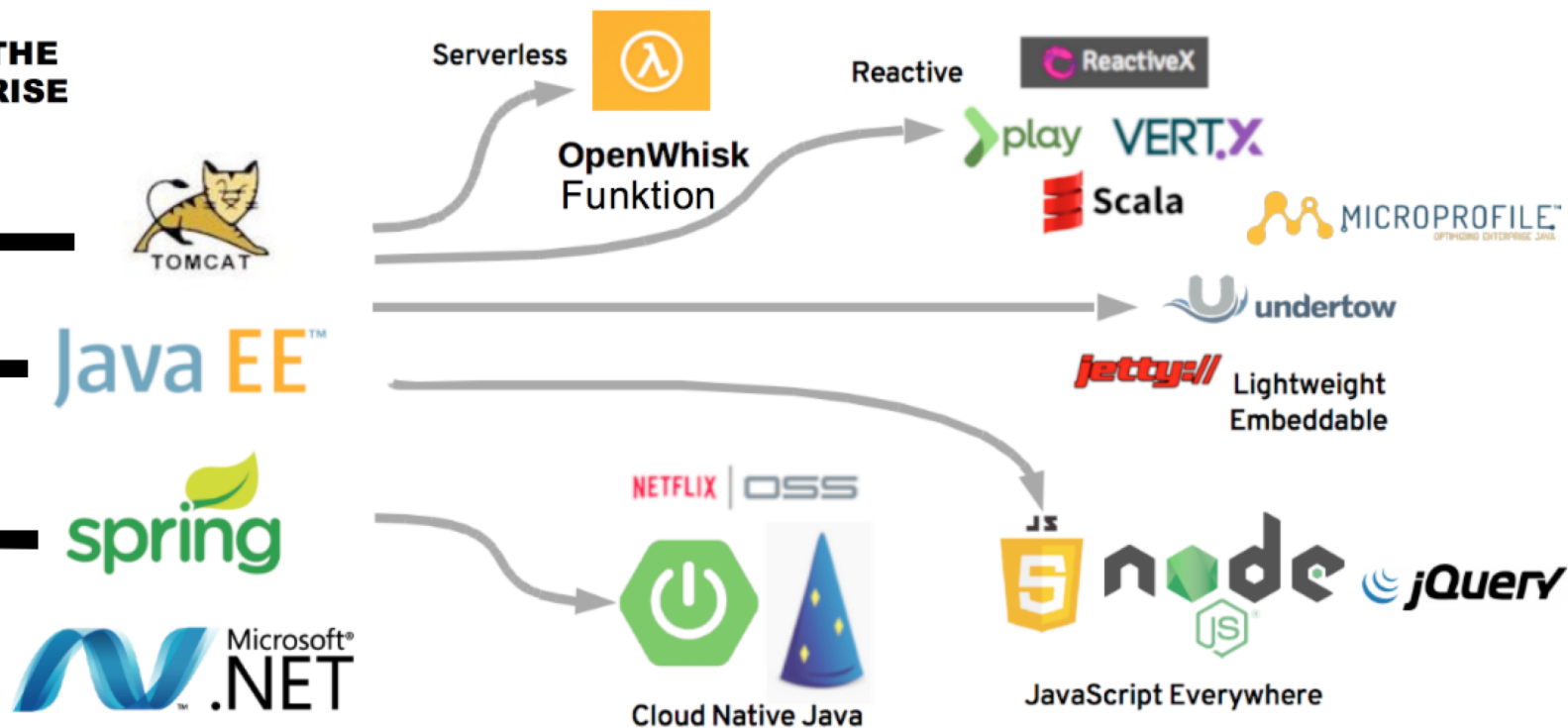
- Cloud Native Development
- Principles
- What do I expect as **C**oder
- Demo time

THE APPSERVER 2000-2014

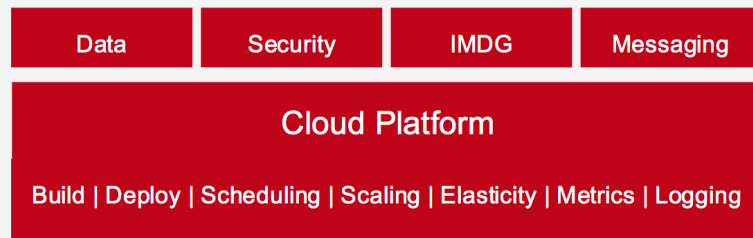
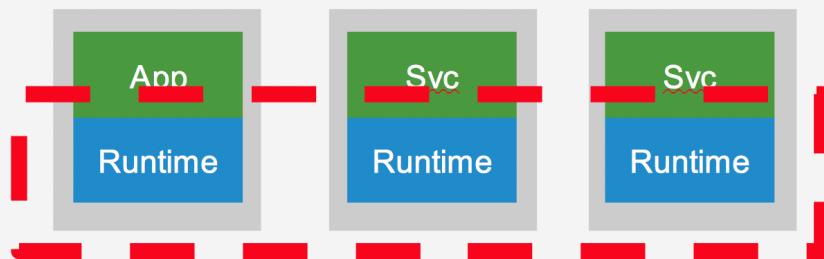


WHAT CHANGED ...

**50% OF THE
ENTERPRISE
APP
MARKET**



THE APPSERVER 2014-...



Application Logic

- > Client-side Load Balancing
- > Service Registration
- > Circuit Breaker
- > Distributed Tracing

Support Services

- > Smart Routing
- > API Management
- > Caching Service
- > Configuration
- > Messaging
- > SSO
- > Registry

Application Logic

- > Client-side Load Balancing
- > Circuit Breaker

Support Services

- > Distributed Tracing
- > API Management
- > Caching Service
- > Messaging
- > SSO



- > Registry
- > Configuration
- > Server-side Load Balancing

Application Logic

Support Services

- > API Management
- > Caching Service
- > Messaging
- > SSO



- > Registry
- > Configuration
- > Server-side Load Balancing
- > Client-side Load Balancing
- > Distributed Tracing
- > Circuit Breaker
- > Fault Injection

2014

Current

Future

IT GOALS

- Speed to develop
- Agility to deliver new features
- Increase margin (cost)
- Maximise infrastructure/tools

DEFINITION

“Cloud-native is an approach to **build** and **run** applications that can **leverage** the capabilities of the cloud platform”

PRINCIPLE - 1

- **Adopt Linux Container**
- unit of packaging
- executable
- portable

PRINCIPLE - 2

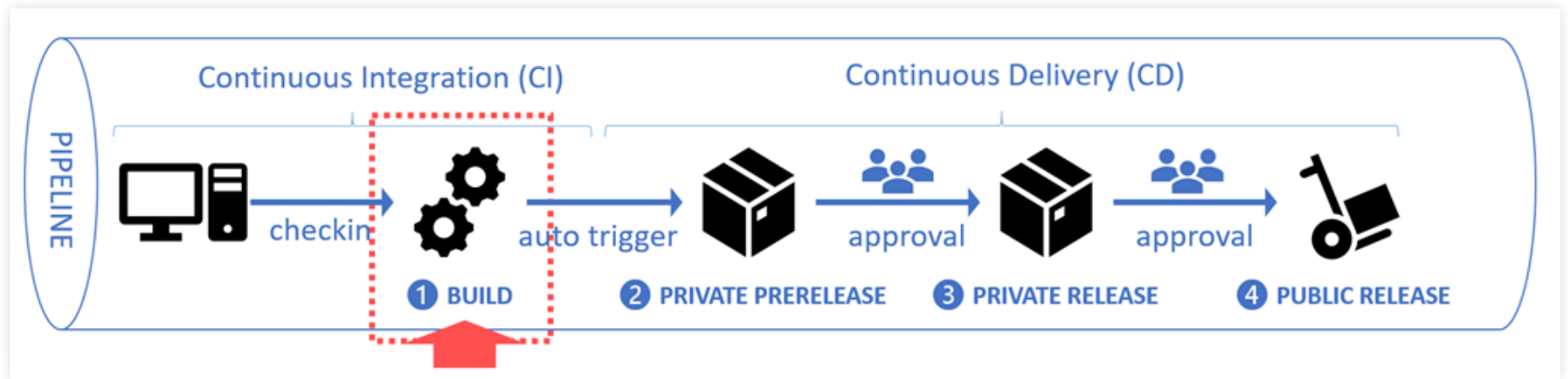
- Design **MicroService** based Architecture
- Isolation
- Health Check
- Circuit Breaker
- Scalability

PRINCIPLE - 3

- Use Cloud Native features
- Access provisioned **MicroServices**
- RBAC & Security
- Consume Services from **Catalog**
- Routing, ACL, A/B testing

PRINCIPLE - 4

- **DevOps** : CI/CD pipelines to automate the build/deployment process



WHAT DO I EXPECT AS CODER



TOOL - MANAGE

```
dabou@dabou ~$ oc -h
OpenShift Client
```

This client helps you develop, build, deploy, and run your applications on any OpenShift or Kubernetes compatible platform. It also includes the administrative commands for managing a cluster under the 'adm' subcommand.

Basic Commands:

types	An introduction to concepts and types
login	Log in to a server
new-project	Request a new project
new-app	Create a new application
status	Show an overview of the current project
project	Switch to another project
projects	Display existing projects
explain	Documentation of resources
cluster	Start and stop OpenShift cluster

BUILD - DEPLOY

Build and Deploy Commands:

<code>rollout</code>	Manage a Kubernetes deployment or OpenShift deployment config
<code>rollback</code>	Revert part of an application back to a previous deployment
<code>new-build</code>	Create a new build configuration
<code>start-build</code>	Start a new build
<code>cancel-build</code>	Cancel running, pending, or new builds
<code>import-image</code>	Imports images from a Docker registry
<code>tag</code>	Tag existing images into image streams

TOOLBOX



LAUNCH

Continuous application delivery,
built and deployed on OpenShift.

LAUNCH YOUR PROJECT

Supported Runtimes



WildFly Swarm offers an innovative approach to packaging and running Java EE applications by packaging them with just enough of the server runtime to "java -jar" your application.

[Learn more](#)



Eclipse Vert.x is a tool-kit for building reactive applications on the JVM.

[Learn more](#)



Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run".

[Learn more](#)



Red Hat® Fuse is a lightweight, flexible integration platform that uses Apache Camel at his core.

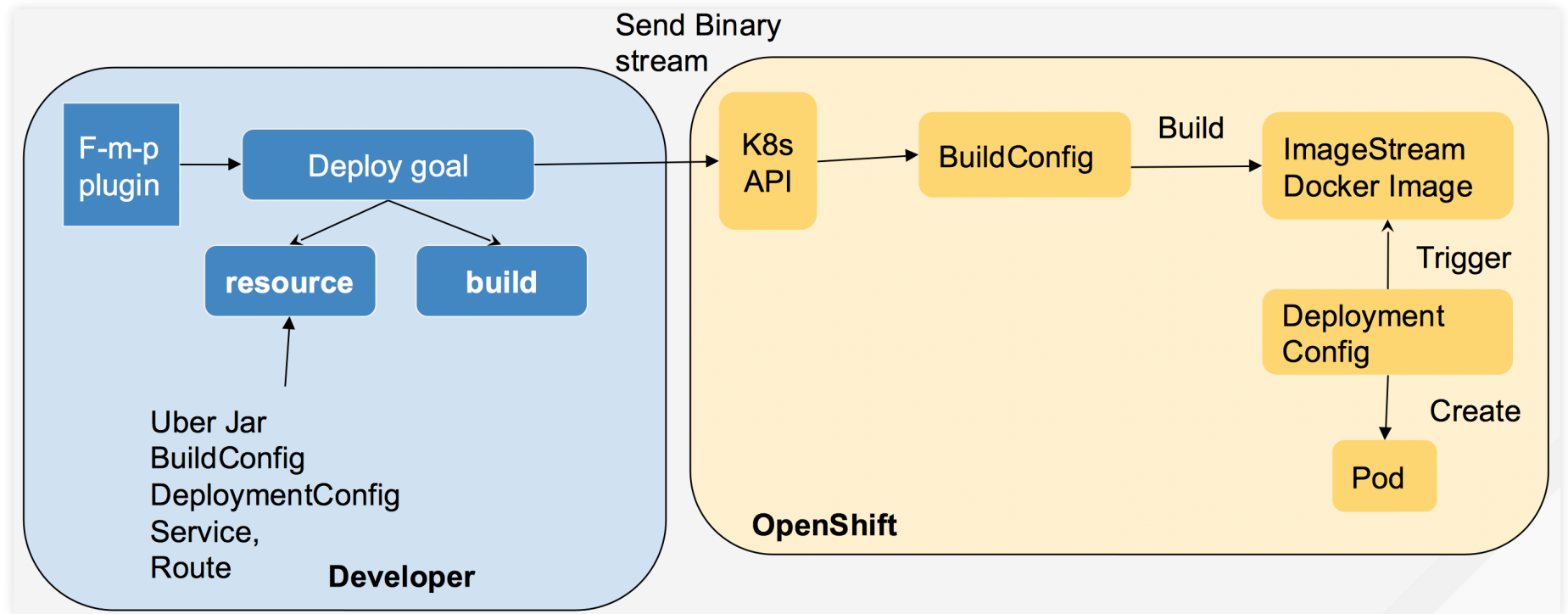
[Learn more](#)



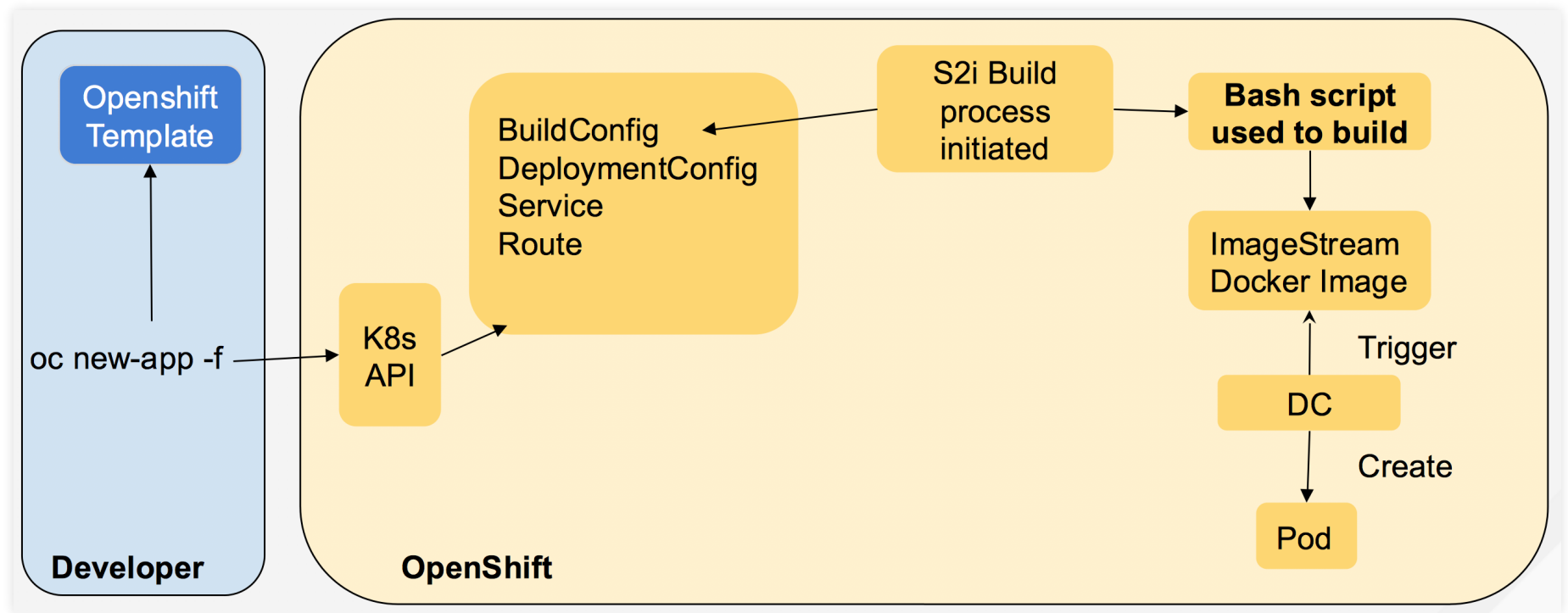
Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient.

[Learn more](#)

LOCAL BUILD → JAR PUSH



PUSH SOURCE → BUILD ON OPENSHIFT



AUTOMATED → JENKINS

The screenshot shows the OpenShift Origin web console. The browser address bar indicates a URL: `https://195.201.87.126:8443/console/project/user14/browse/pipelines`. The page header shows 'OPENSIFT ORIGIN' (likely a typo for OpenShift) and the user 'user14'. The left sidebar contains navigation links: Overview, Applications, Builds (selected), Resources, and Storage. The main content area is titled 'Pipelines' and shows a pipeline named 'cloud-native-backend-user14' created 16 hours ago. The source repository is listed as `https://github.com/timothyvandenbrande/cloud-native-backend.git`. Under 'Recent Runs', 'Build #1' is shown as successful, completed 16 hours ago. The build process is visualized as a sequence of steps: 'Test' (17s), 'Use appropriate name...' (0s), and 'Deploy' (35s), all marked with green checkmarks. At the bottom, links for 'View Pipeline Runs' and 'Edit Pipeline' are provided.

← → ↻ ⚠ Not Secure | <https://195.201.87.126:8443/console/project/user14/browse/pipelines>

Apps RedHat Internal xpaas-qe maven-kubernetes-j...

OPENSIFT ORIGIN

☰ user14 ▾

🌐 Overview

📦 Applications >

🏗 Builds >

📁 Resources >

💾 Storage

Pipelines [Learn More](#)

[cloud-native-backend-user14](#) created 16 hours ago

Source Repository: <https://github.com/timothyvandenbrande/cloud-native-backend.git>

Recent Runs

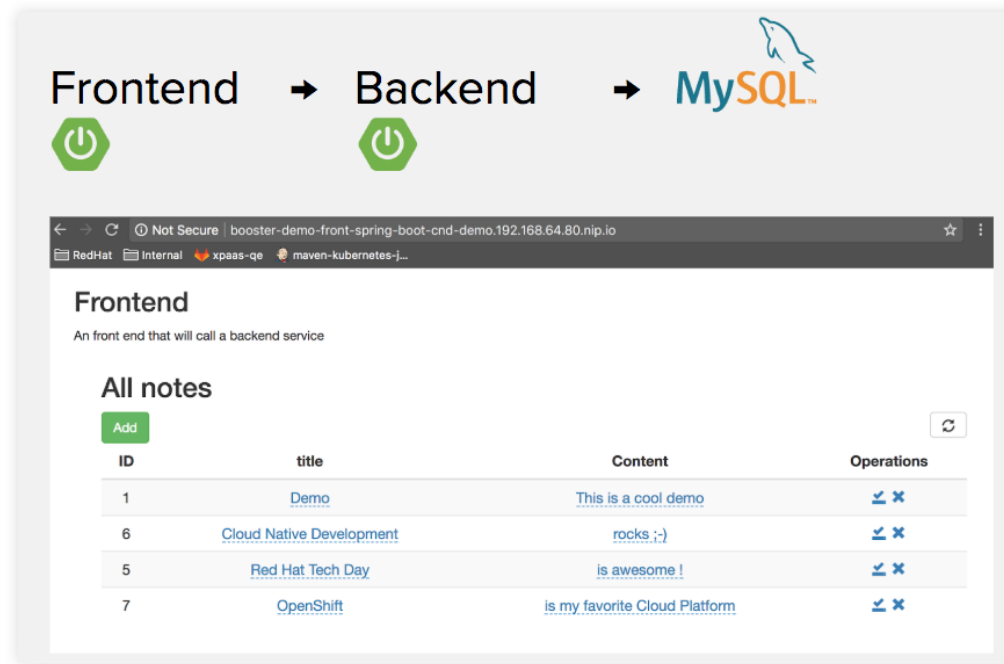
✓ Build #1 16 hours ago View Log	Test 17s	→	Use appropriate name... 0s	→	Deploy 35s
---	-------------	---	-------------------------------	---	---------------

[View Pipeline Runs](#) | [Edit Pipeline](#)

WHAT'S ELSE

- Service Catalog
- Security (RBAC, Keycloak - OAuth2,...)
- Metrics (Prometheus, Actuator)
- Remote Debugging
- Integration testing (Arquillian)
- Logging (Jaeger)
- Routing/ACL/CircuitBreaker ... (Google Istio)

DEMO



<https://github.com/snowdrop/cloud-native-lab>