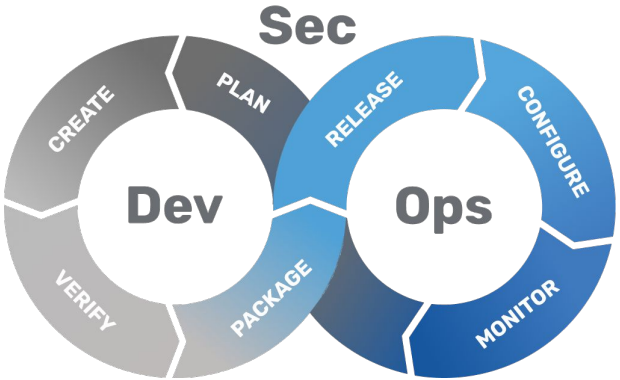
DevOps
Transformation #5



Wifi: Skooldio Guest
Pass: skooldioupskill

Jirayut Nimsaeng (Dear)

CEO & Founder, Opsta (Thailand) Co.,Ltd.

Skooldio Workshop @ MBK Tower
July 2-4, 2025

https://to.skooldio.com/devops-public-workshop-5

Instructor

Jirayut Nimsaeng (Dear)

Jirayut has been involved in DevSecOps, Container, Cloud Technology and Open Source for over 10 years. He has experienced and succeeded in transforming several companies to deliver greater values and be more agile.

- **He is Founder and CEO of Opsta (Thailand) Co.,Ltd.**
- **He is Cloud/DevSecOps Transformation Consultant and Solution Architecture**
- **He is the first Certified Kubernetes Administrator (CKA) and Certified Kubernetes Security Specialist (CKS) in Thailand**
- **He is first Thai Google Cloud Developer Expert (GDE) in Thailand**
- **Google Cloud Certified - Professional Cloud Architect and Associate Cloud Engineer**

Agenda Day 1

- **Introduction**
  - What is DevSecOps?
  - DevSecOps Flow
  - DevSecOps Workshop Overview
- **Version Control**
  - Introduction to Git and GitLab
  - Basic Git Commands
  - Branching System
- **Container**
  - Introduction to Docker
  - Basic Docker Commands
  - Dockerfile and Docker Compose
  - Make Ratings Service run on Container
  - Practice with others services

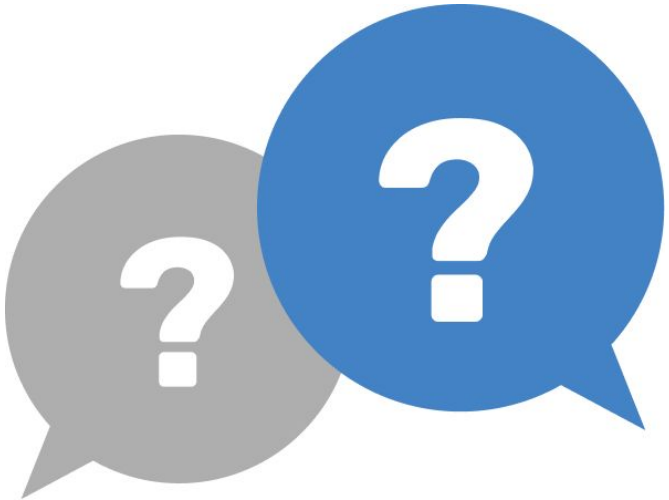
Agenda Day 2

- **Container Cluster**
  - Introduction to Kubernetes
  - Basic kubectl commands
  - Deploy Application with Kubernetes Manifest Files
- **Helm**
  - Introduction to Helm
  - Deploy MongoDB with Public Helm Charts
  - How to write Helm Chart
  - Deploy Application with Helm

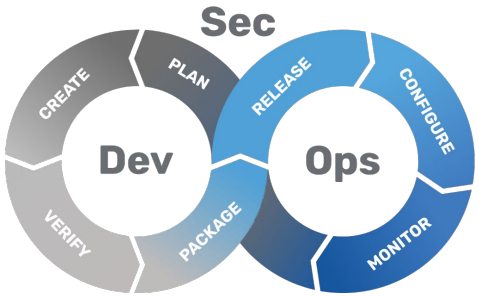
Agenda Day 3

- **CI/CD**
  - Introduction to GitLab CI/CD
  - GitLab CI/CD Concept
  - GitLab CI/CD Runner
  - .gitlab-ci.yml
  - Develop Pipeline to Deploy Application
  - Pipeline as Code Template
- **Practice with other services**
- **Q/A**
- **Wrap up**

You can ask questions anytime



What is DevSecOps?

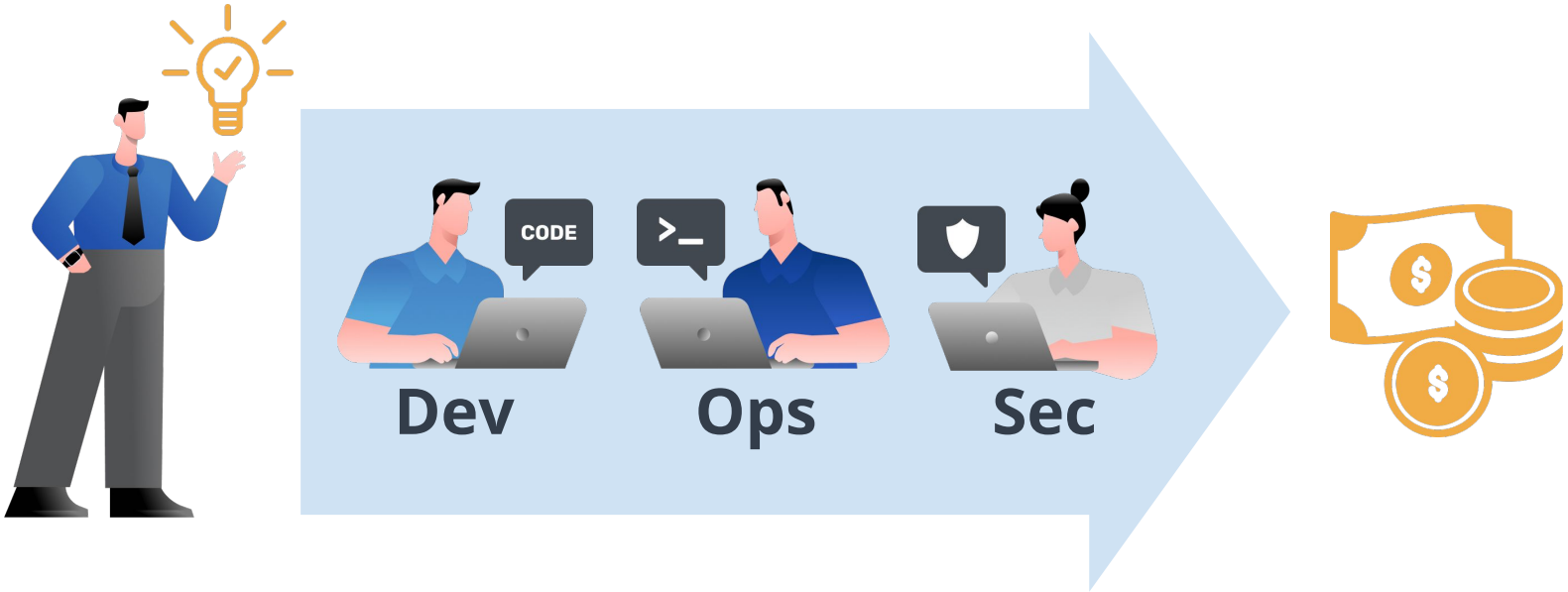


Transformation

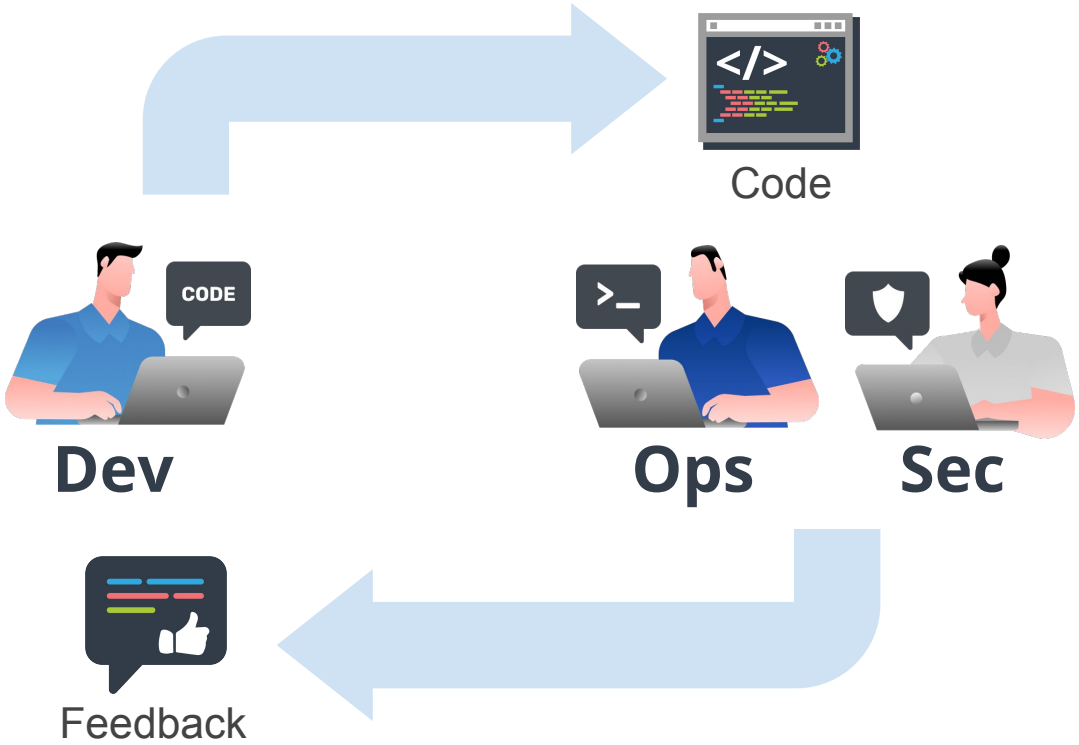
Opsta Education

Skooldio

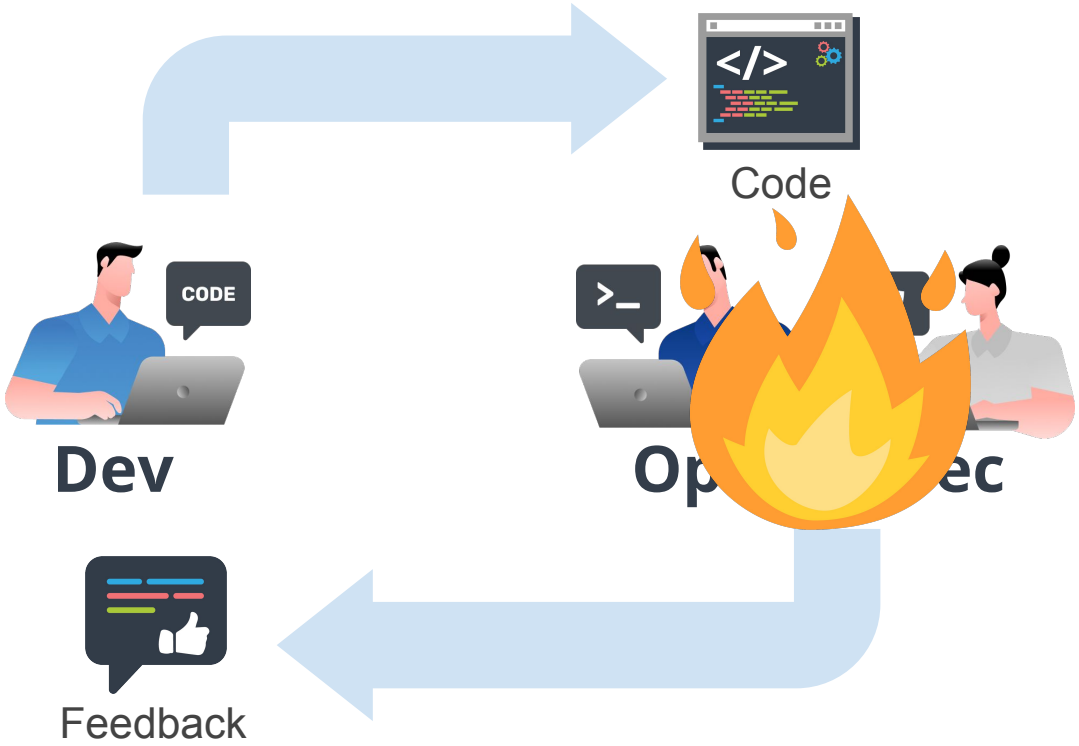
Business in IT Industy



Ideal Development Cycle



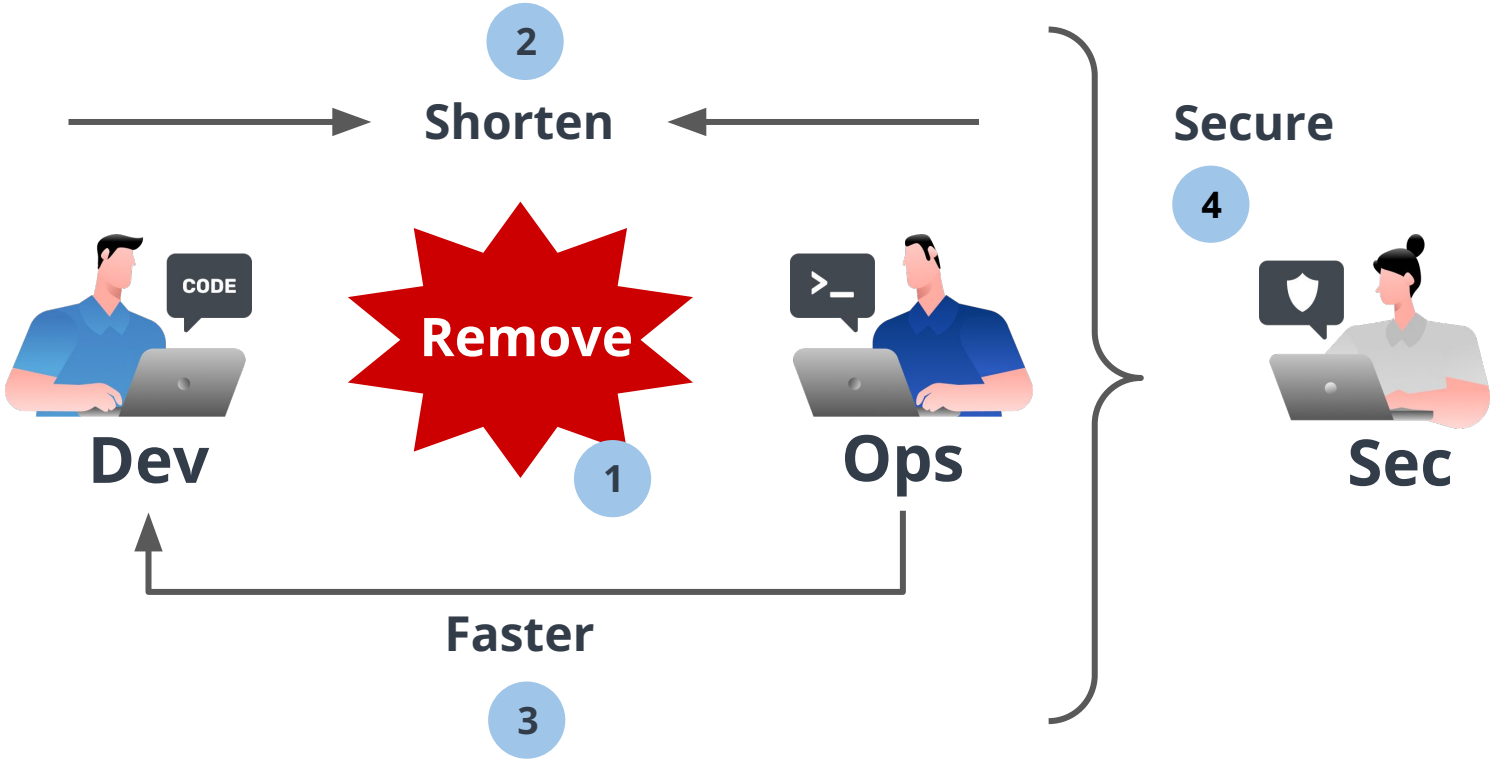
Reality



Silo

The Wall

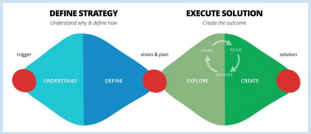
DevSecOps Culture



Making an IT product

Discovery

Design Thinking

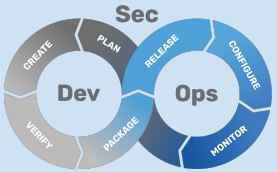


Develop

Scrum Kanban

Delivery

DevOps DevSecOps



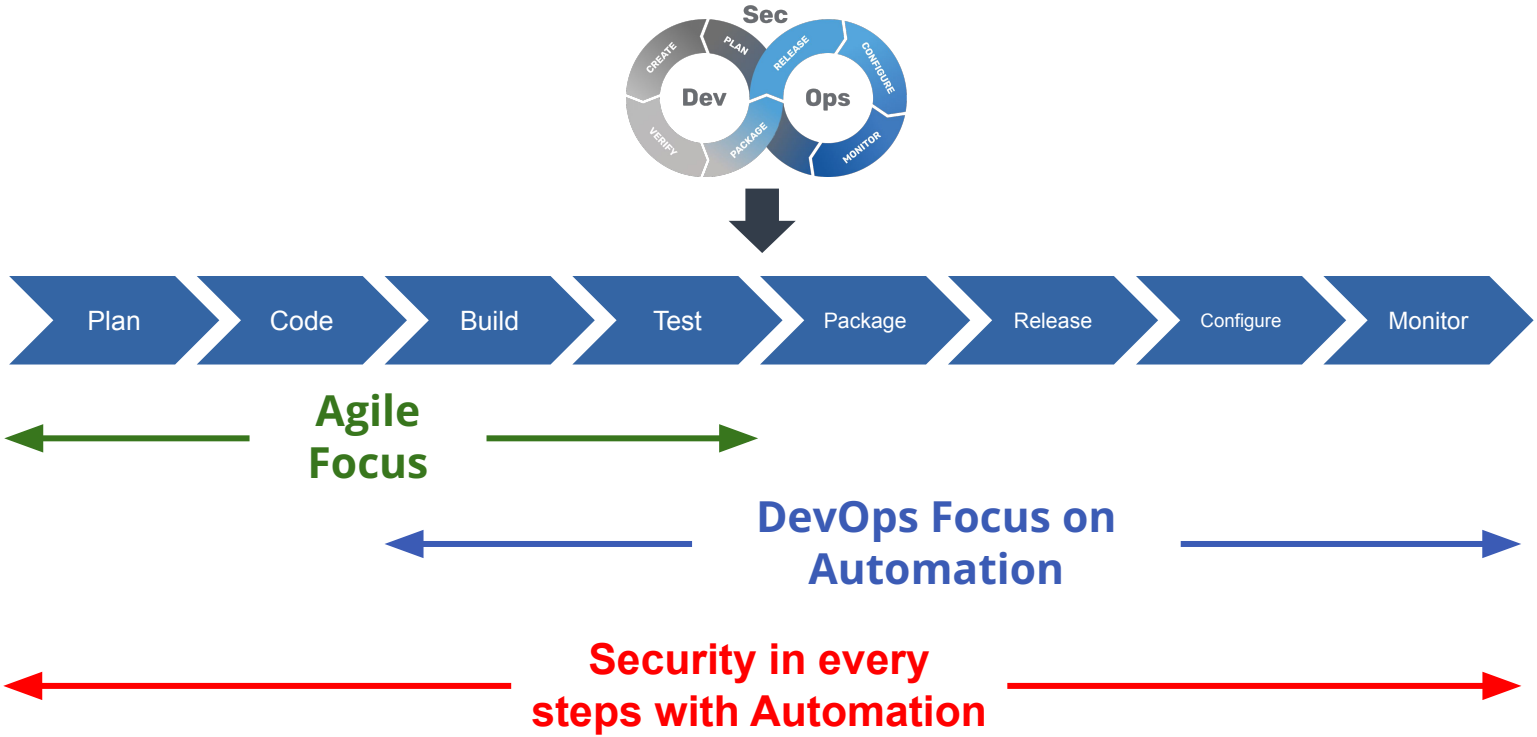
Data

Data Lake Data Mesh

Artificial Intelligence

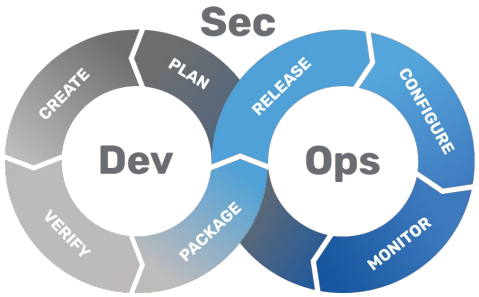
Skooldio Opsta Education

DevSecOps and Agile



Skooldio Opsta Education

DevSecOps Flow & Technologies

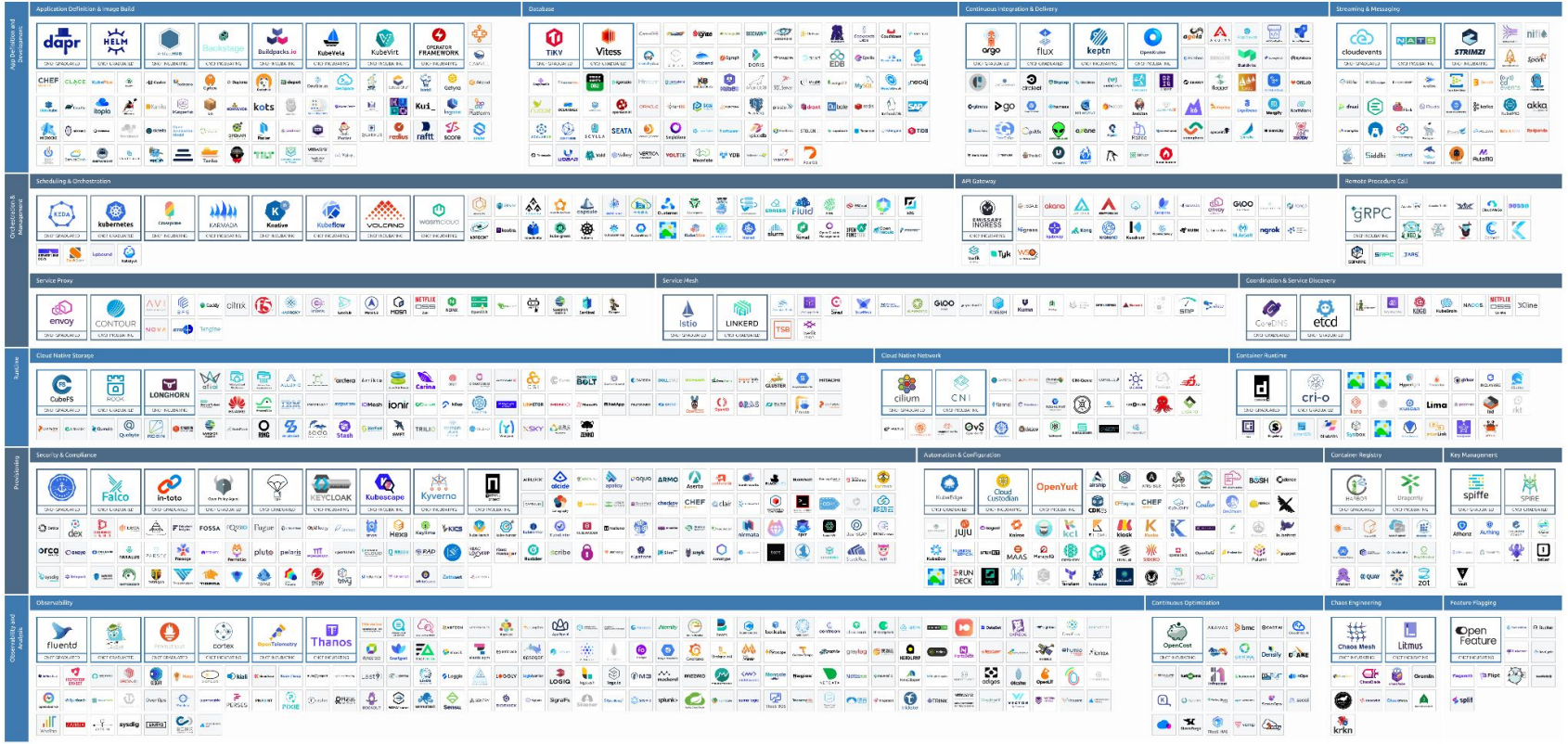


Transformation

Opsta Education

Skooldio

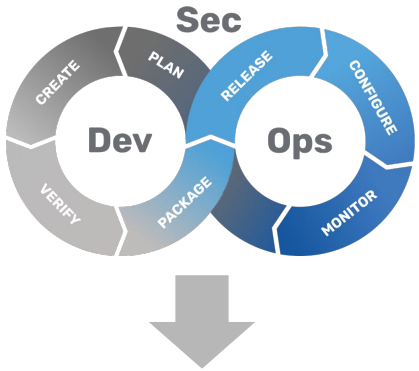
CNCF Landscape / DevSecOps Technologies



https://l.cncf.io

Skooldio Opsta Education

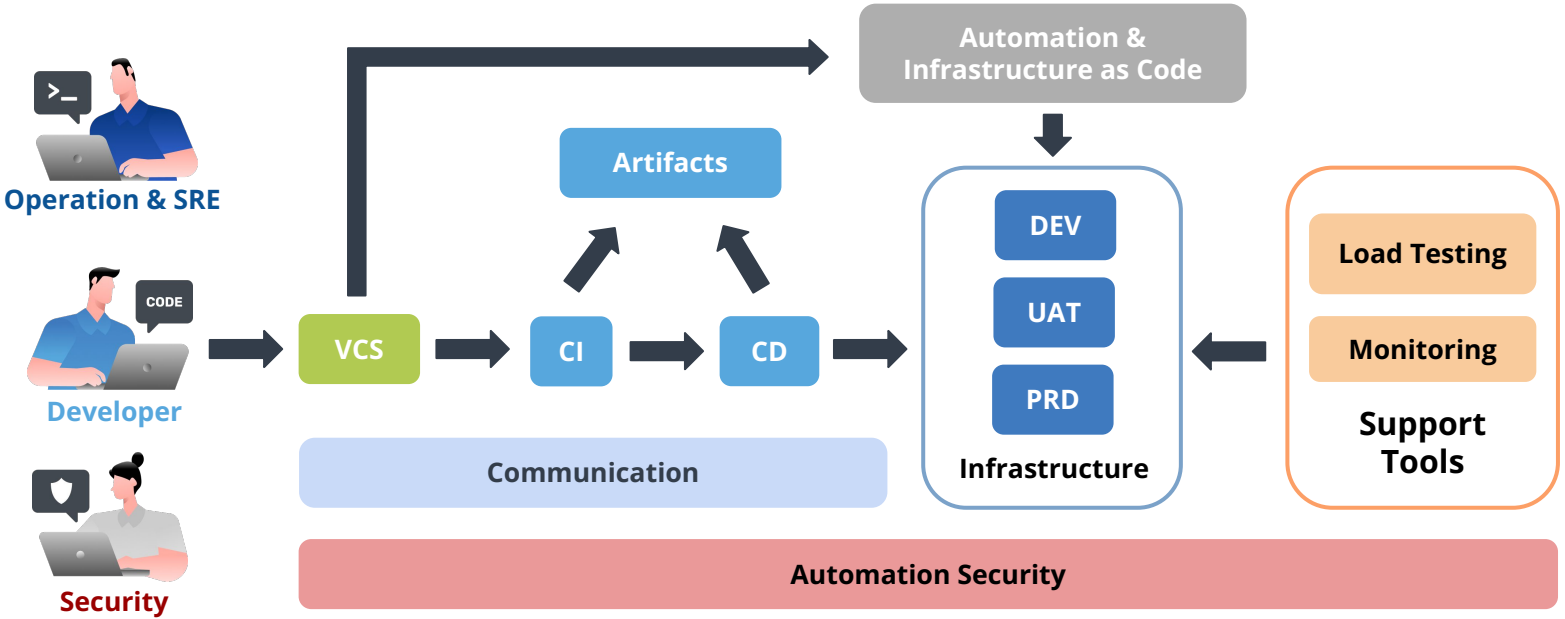
DevSecOps Flow



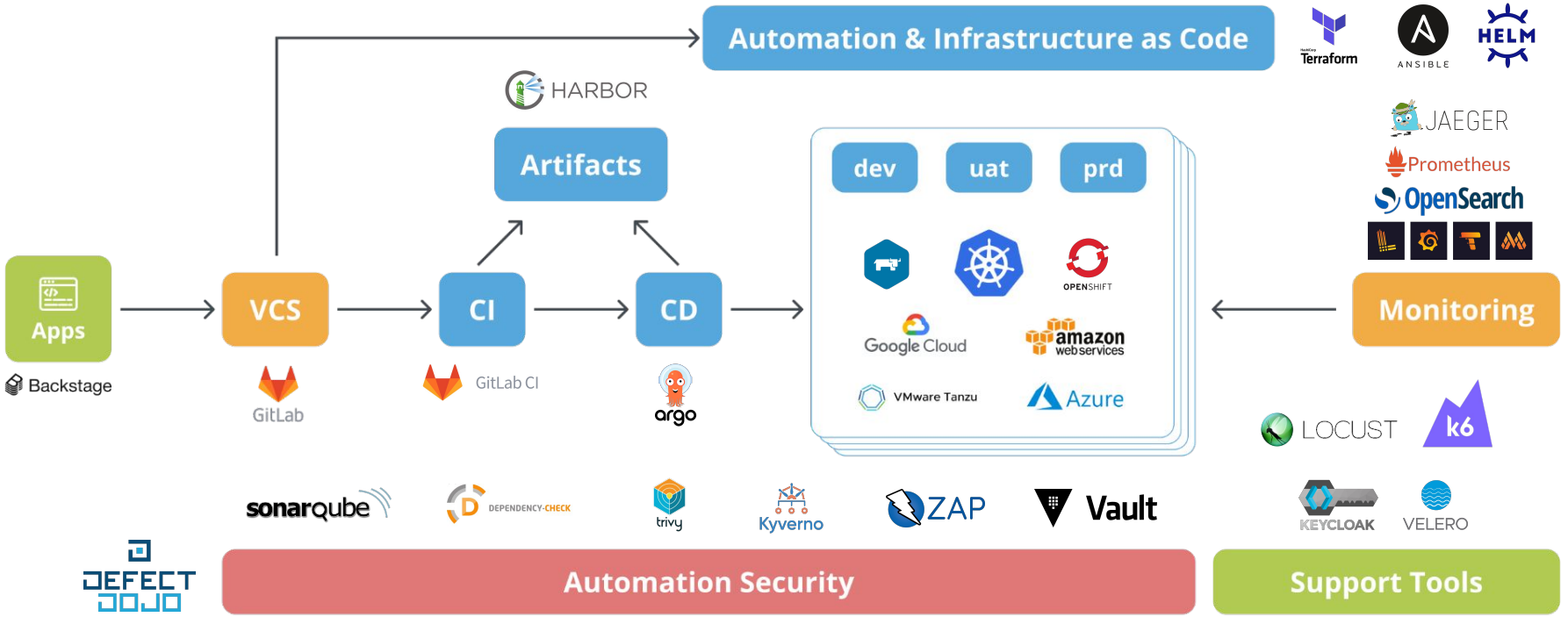
Plan → Code → Build → Test → Package → Release → Configure → Monitor

Security

Generic DevSecOps Flow & Components

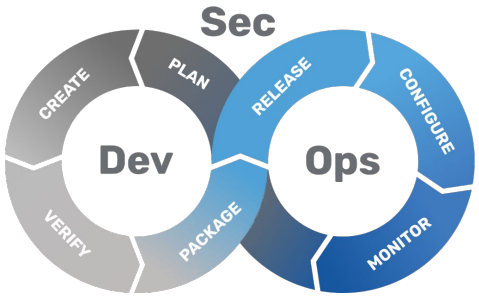


Cloud Native and DevSecOps Components



Skooldio Opsta Education

Workshop Overview

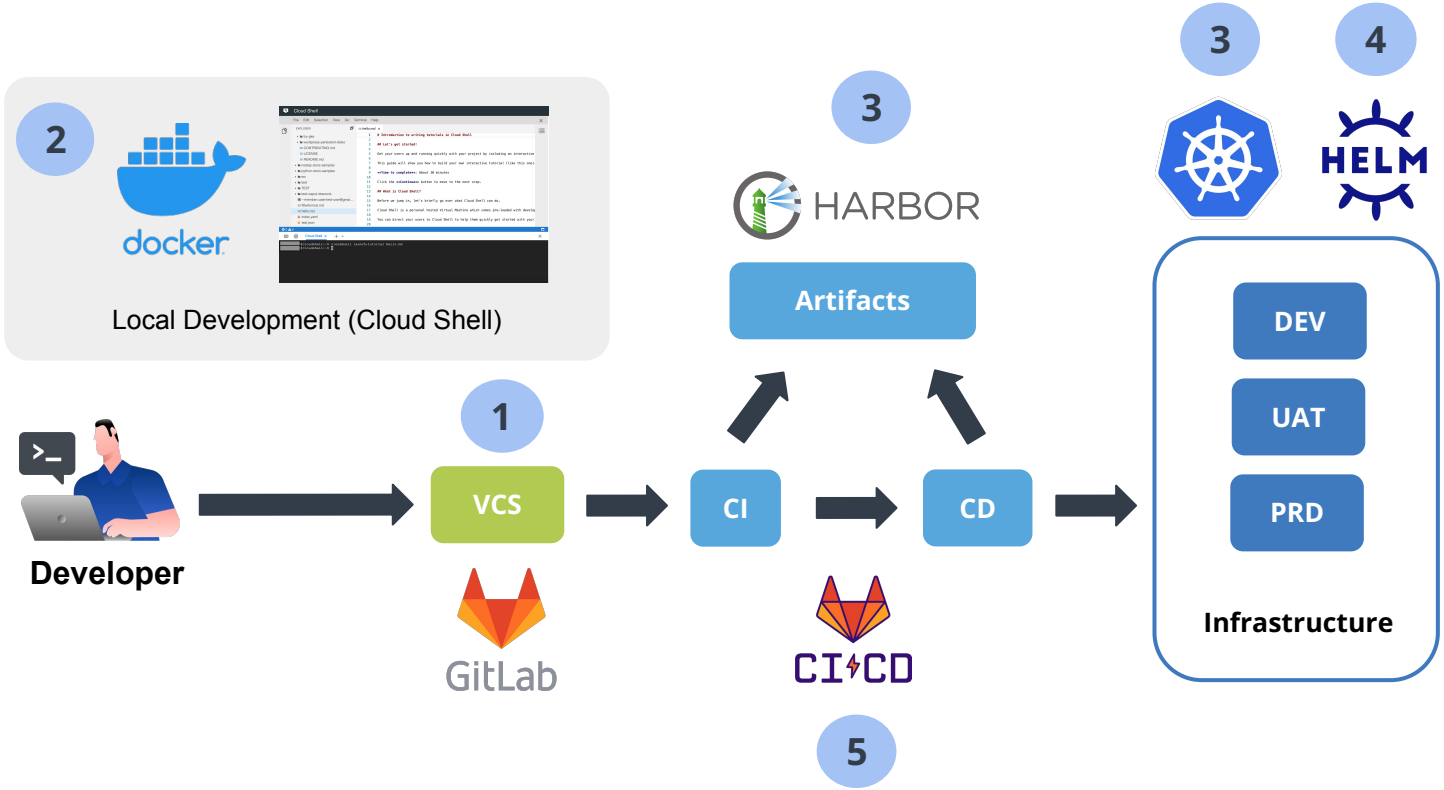


Transformation

Opsta Education

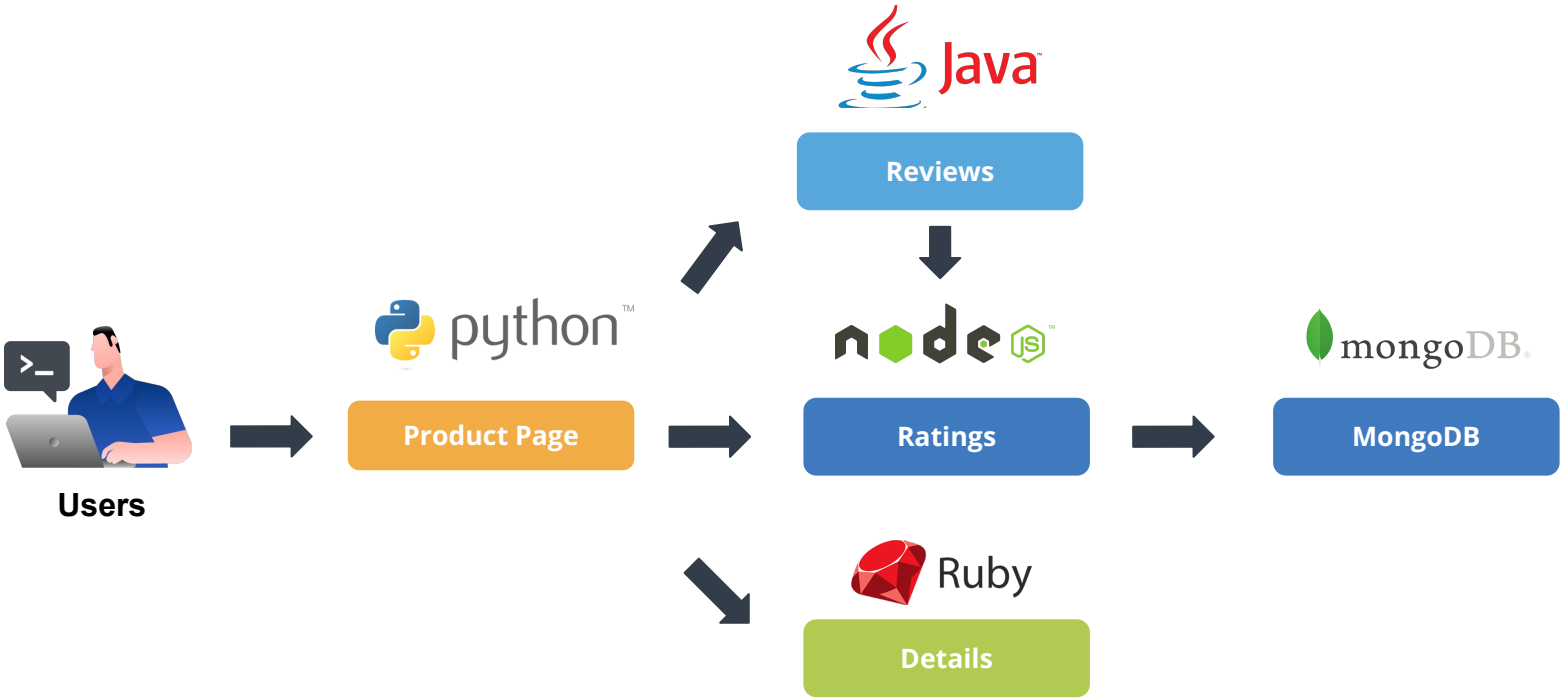
Skooldio

DevOps Workshop Overview

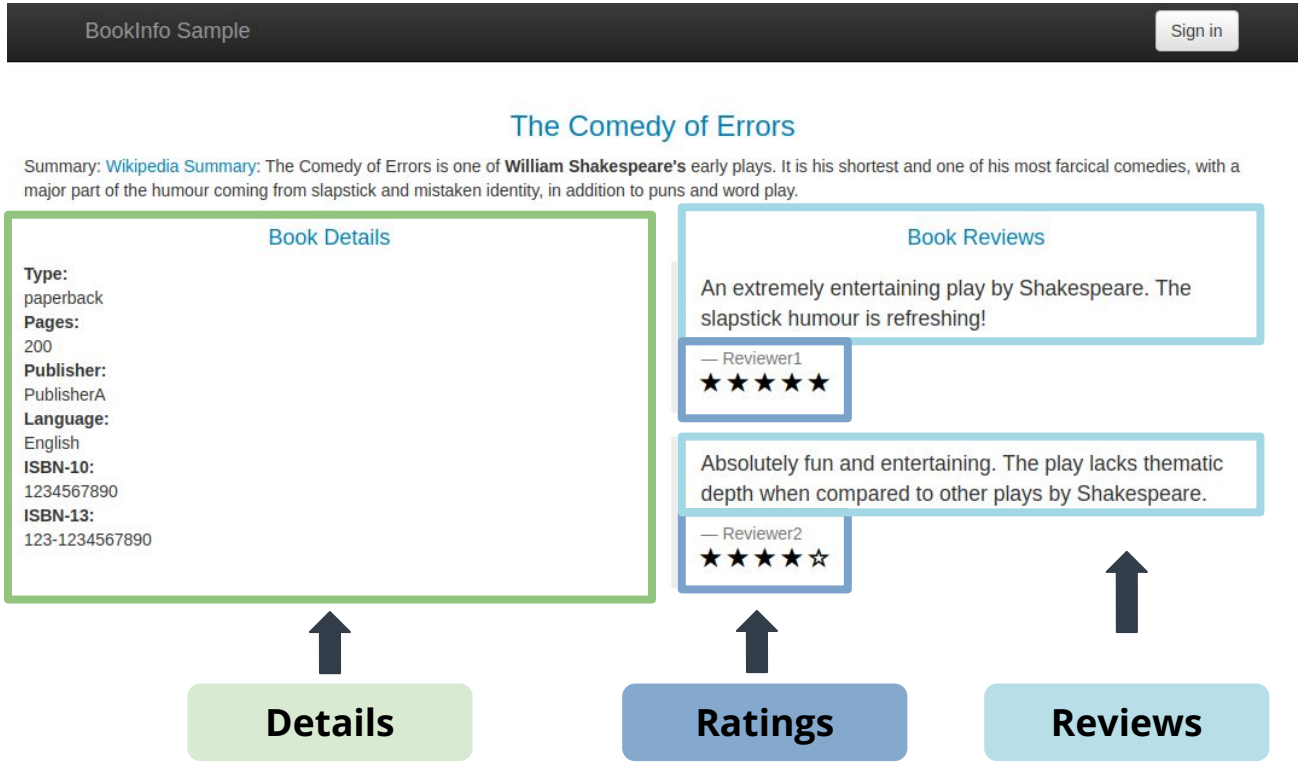


Skooldio Opsta Education

Bookinfo Application Overview



Bookinfo Product Page



Bookinfo Services Information

Services	Pages	Paths
Product Page	Health Check	*GET /health*
	Main Page	*GET /productpage*
Reviews	Health Check	*GET /health*
	Get Reviews API	*GET /reviews/[productid]*
Ratings	Health Check	*GET /health*
	Get Ratings API	*GET /ratings/[productid]*
Details	Health Check	*GET /health*
	Get Details API	*GET /details/[productid]*

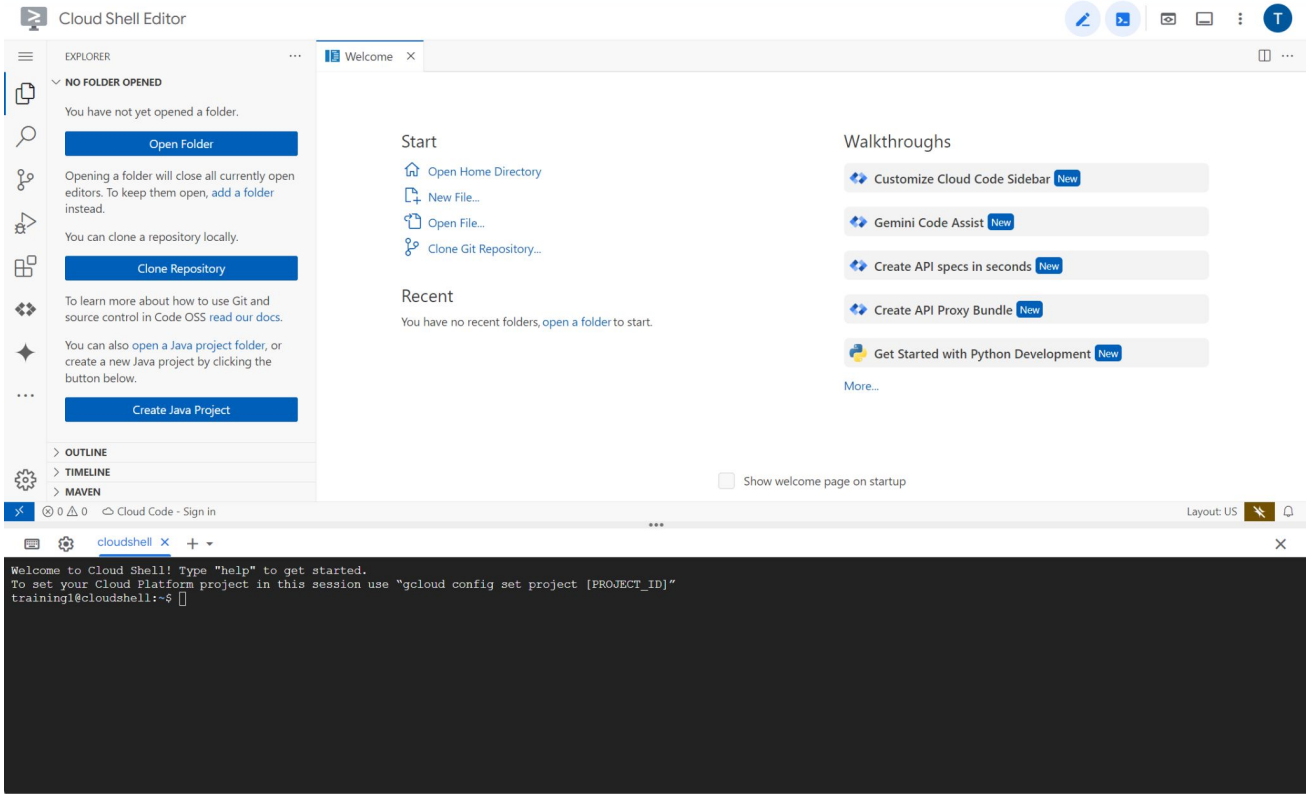
DevOps Tools Information

Tools	Description
Google Cloud Shell	https://ssh.cloud.google.com user: [YOURGMAIL]@gmail.com pass: [YOURPASSWORD]
Gitlab	https://git.demo.opsta.co.th user: training1, training2, ... , training30 pass: Bookinfo1234!
Workshop Document	https://git.demo.opsta.co.th/sdt5/devops-workshop/-/blob/main/README.md
Harbor	https://harbor.demo.opsta.co.th user: bookinfo Pass: Bookinfo1234!
Put your Gmail here	https://docs.google.com/spreadsheets/d/1bQ2cki0nQzAAf4PhzwaPHmAj9f4_0Lj2P9nL2hDs_EI/edit?usp=sharing

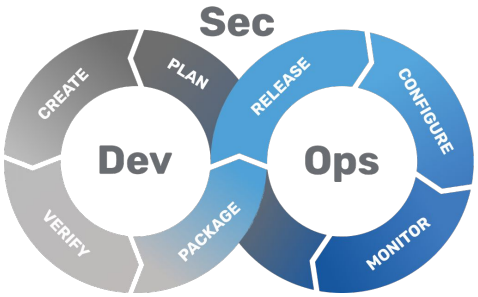
Bookinfo Domain Information

Bookinfo Application Domain Environment	Development: http://bookinfo.dev.opsta.co.th UAT: http://bookinfo.uat.opsta.co.th Production: http://bookinfo.opsta.co.th
Bookinfo Application Domain	http://bookinfo.opsta.co.th/training[X]/productpage http://bookinfo.opsta.co.th/training[X]/reviews http://bookinfo.opsta.co.th/training[X]/details http://bookinfo.opsta.co.th/training[X]/ratings

Google Cloud Shell



Introduction to Git & GitLab



Transformation

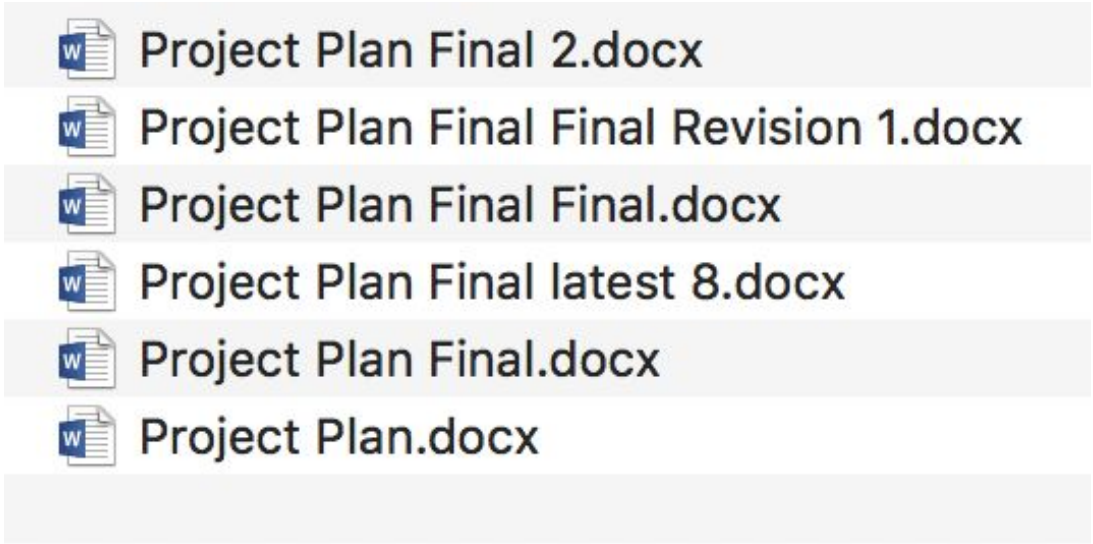
GitLab

git

Opsta Education

Skooldio

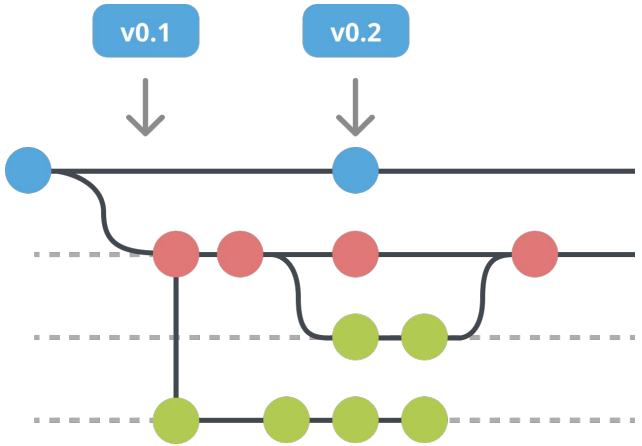
Classic Problem



Version Control System (VCS)

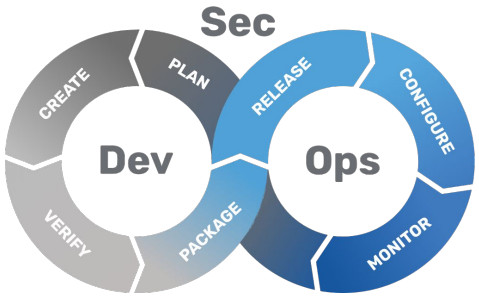
Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.

- Good for text file
- Not recommend for binary file



Version Control Software

Git Version Control



Transformation

GitLab

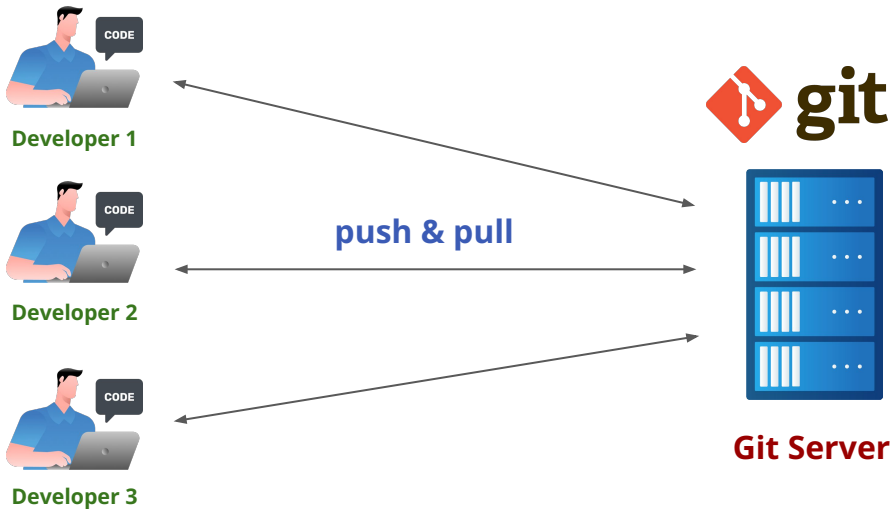
git

Opsta Education

Skooldio

Git Version Control

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.



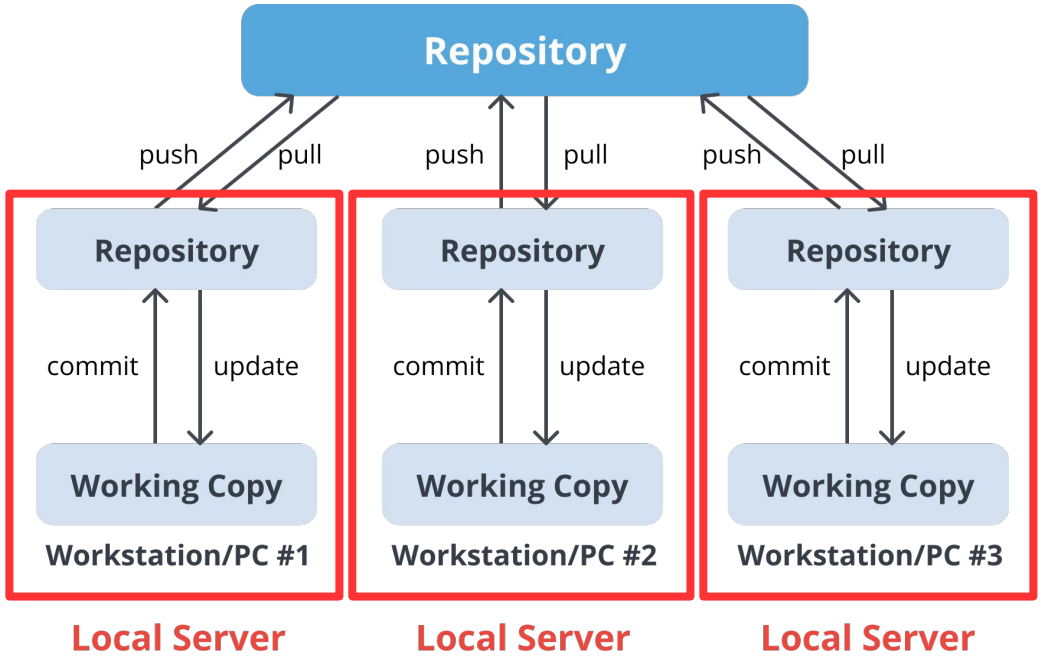
Popular Git Software

Server

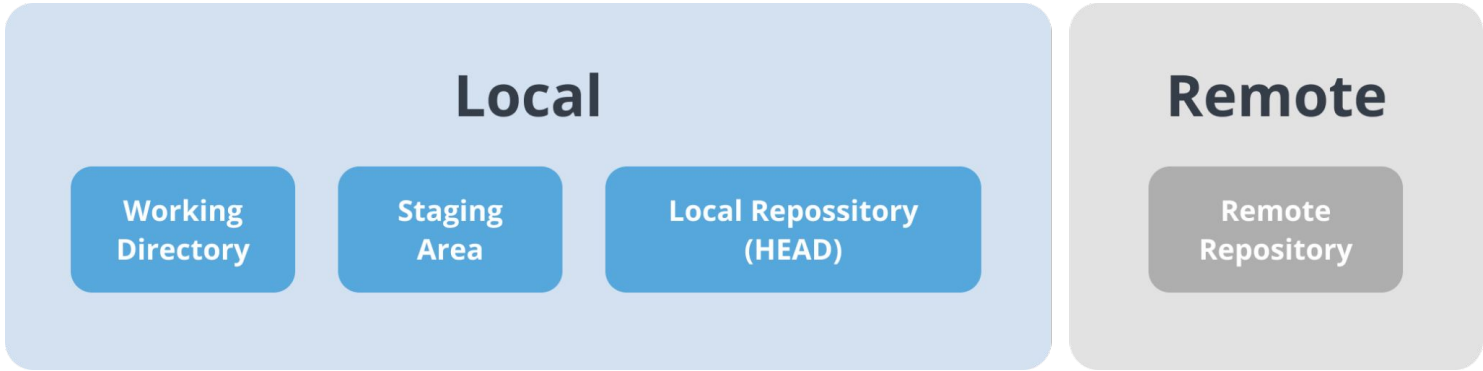
Client

Git Architecture

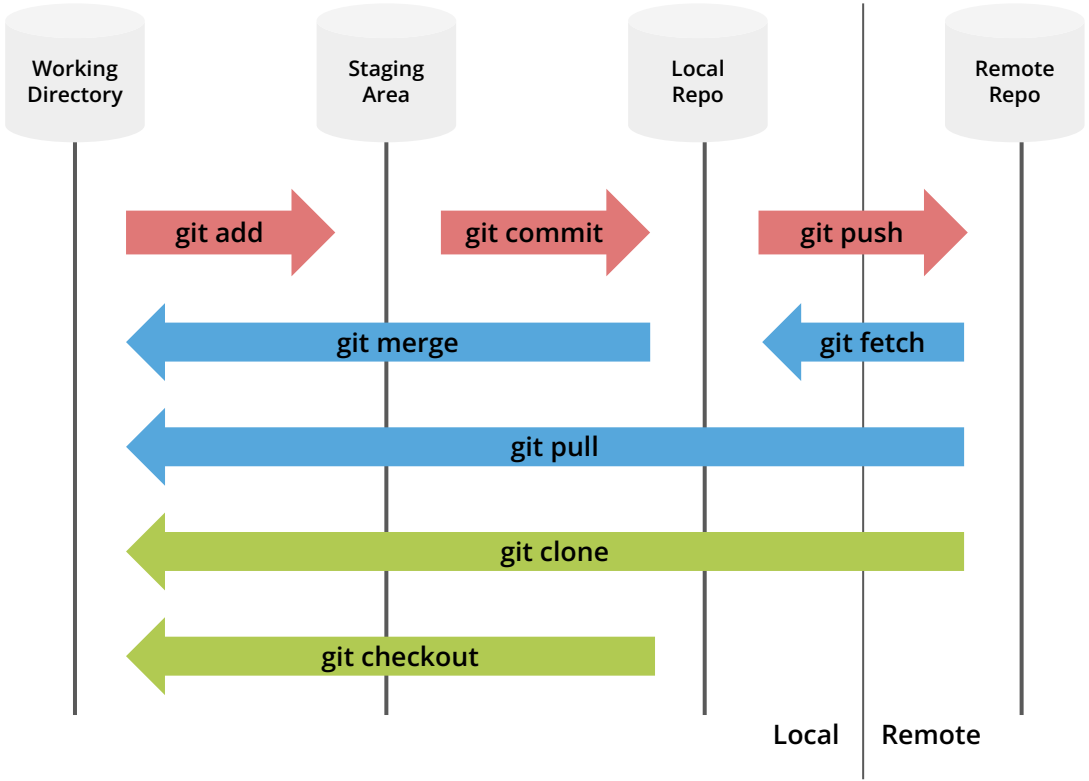
Distributed version control system server



Git File System

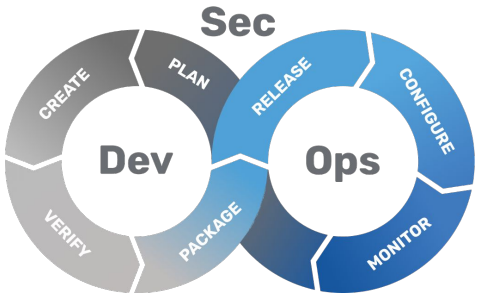


Git Workflow



https://twitter.com/alexxubyte/status/1630972442712604682

Git Features



Transformation

GitLab

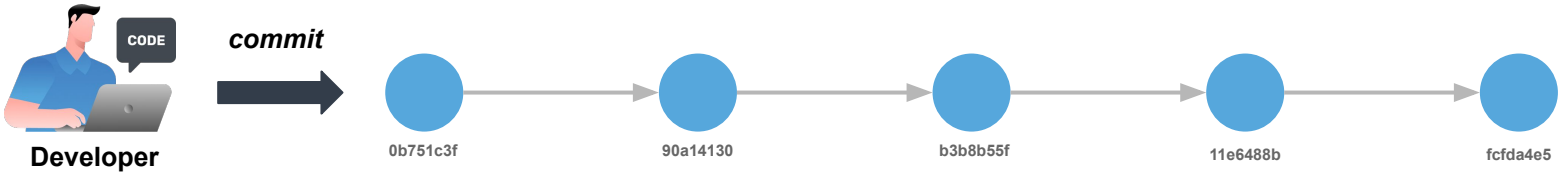
git

Opsta Education

Skooldio

Git Features: History

See what exactly be change and when.



Commit ID

16 Jan, 2016 1 commit

Fix wrong file mode written to index when mark/unmark as symlink ...
ch3cooli committed a year ago — fcfda4e5 — Browse Files

12 Jan, 2016 1 commit

Refactoring: Reduce indention ...
Sven Strickroth committed a year ago — 11e6488b — Browse Files

10 Jan, 2016 1 commit

Fix possible division by zero ...
Sven Strickroth committed a year ago — b3b8b55f — Browse Files

08 Jan, 2016 4 commits

Refactor: Use smart pointers ...
Sven Strickroth committed a year ago — 90a14130 — Browse Files

Initialize SYS_IMAGE_LIST after initializing OLE ...
Sven Strickroth committed a year ago — 0b751c3f — Browse Files

Fix typo ...
Sven Strickroth committed a year ago — 3fd7fb7a — Browse Files

Remove compilation warning ...
Jiří Engelthaler committed a year ago — 1e4021bd — Browse Files

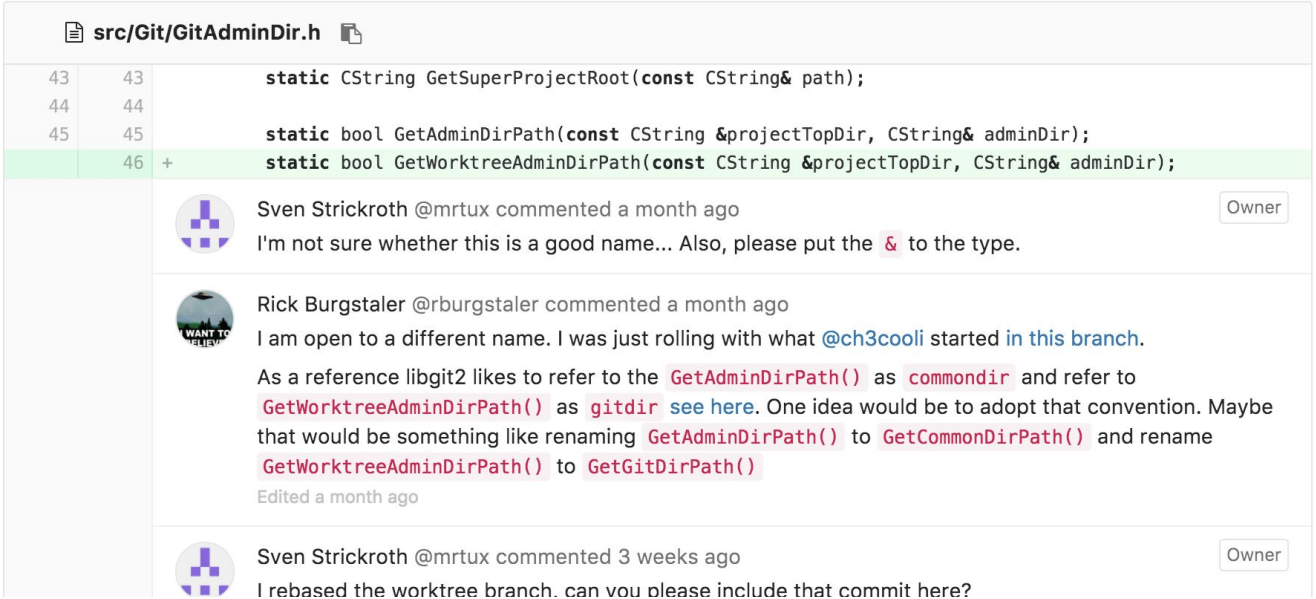
Git Features: History

Really exact. Line by line history.

Added skip-worktree flag to prop... Sven Strickroth committed 4 years ago	d95de360	176	}
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	177	if (executable == BST_CHECKED)
Fixed issue #696: Allow specifyi... Sven Strickroth committed 4 years ago	8288e1c7	178	{
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	179 180	if (!(e->mode & 0111)) {
Fix wrong file mode written to i... ch3cooli committed a year ago	fcfda4e5	181	e->mode = GIT_FILEMODE_BLOB_EXECUTA
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	182 183	changed = true; }
Fixed issue #696: Allow specifyi... Sven Strickroth committed 4 years ago	8288e1c7	184	}
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	185	else if (executable != BST_INDETERMINATE)
Fixed issue #696: Allow specifyi... Sven Strickroth committed 4 years ago	8288e1c7	186	{
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	187 188	if (e->mode & 0111) {
Fix wrong file mode written to i... ch3cooli committed a year ago	fcfda4e5	189	e->mode = GIT_FILEMODE_BLOB;
Allow to change flags of multipl... Sven Strickroth committed 4 years ago	a9804d7f	190 191	changed = true; }

Git Features: Code Review

Allow micro level collaboration between team member.



src/Git/GitAdminDir.h

```
43  43      static CString GetSuperProjectRoot(const CString& path);
44  44
45  45      static bool GetAdminDirPath(const CString &projectTopDir, CString& adminDir);
    46  +   static bool GetWorktreeAdminDirPath(const CString &projectTopDir, CString& adminDir);
```

Sven Strickroth @mrtux commented a month ago — Owner

I'm not sure whether this is a good name... Also, please put the `&` to the type.

Rick Burgstaler @rburgstaler commented a month ago

I am open to a different name. I was just rolling with what @ch3cooli started in this branch.

As a reference libgit2 likes to refer to the `GetAdminDirPath()` as `commondir` and refer to `GetWorktreeAdminDirPath()` as `gitdir` see here. One idea would be to adopt that convention. Maybe that would be something like renaming `GetAdminDirPath()` to `GetCommonDirPath()` and rename `GetWorktreeAdminDirPath()` to `GetGitDirPath()`

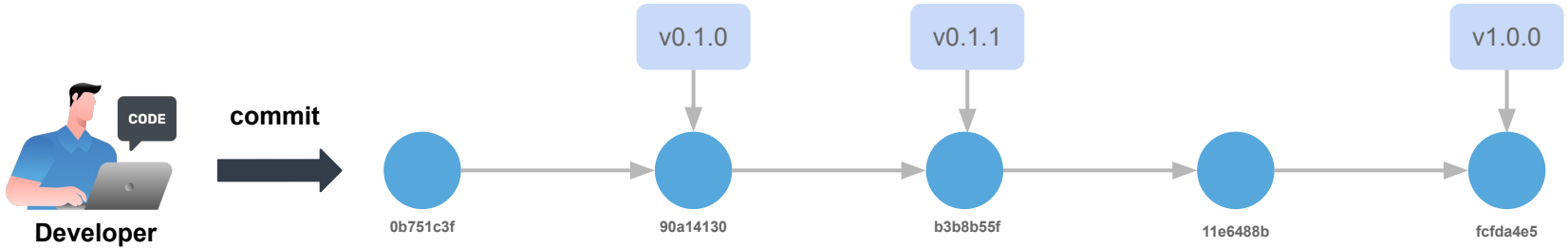
Edited a month ago

Sven Strickroth @mrtux commented 3 weeks ago — Owner

I rebased the worktree branch, can you please include that commit here?

Git Features: Tag

Tagging is generally used to capture a point in history that is used for a marked version release (i.e. v1.0.1)



REL_2.3.0.0_EXTERNAL Released TortoiseGit 2.3.0.0
d2068cc0 · Prepare release and bump version numbers to 2.3.0.0 · 4 months ago

REL_2.2.0.0_EXTERNAL Released TortoiseGit 2.2.0
7d0cf461 · Prepare release and bump version numbers to 2.2.0.0 · 7 months ago

REL_2.1.0.0_EXTERNAL Released TortoiseGit 2.1.0.0
010b3875 · Prepare release and bump version numbers to 2.1.0.0 · 10 months ago

REL_2.0.0.0_EXTERNAL Released TortoiseGit 2.0.0.0
920ed288 · Prepare release and bump version numbers to 2.0.0.0 · 11 months ago

REL_1.8.16.0_EXTERNAL Released TortoiseGit 1.8.16.0
9a490362 · Prepare release and bump version numbers to 1.8.16.0 · a year ago

How to versioning?

Semantic Versioning (semver)

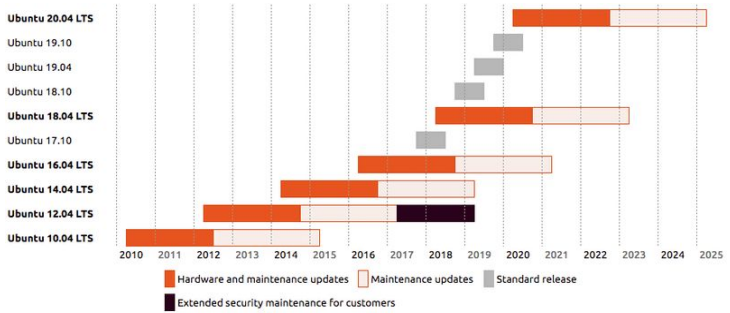
8 . 1 . 14

MAJOR MINOR PATCH

Calendar Versioning (calver)

2020.1.4018-pre

year release build status



Skooldio Opsta Education

Git Conventional Commits 1.0.0

1. **fix**: a commit of the *type* fix patches a bug in your codebase (this correlates with **PATCH** in Semantic Versioning).
2. **feat**: a commit of the *type* feat introduces a new feature to the codebase (this correlates with **MINOR** in Semantic Versioning).
3. **BREAKING CHANGE**: a commit that has a footer BREAKING CHANGE:, or appends a ! after the type/scope, introduces a breaking API change (correlating with **MAJOR** in Semantic Versioning). A BREAKING CHANGE can be part of commits of any *type*.
4. *types* other than fix: and feat: are allowed, for example **@commitlint/config-conventional** (based on the **Angular convention**) recommends build:, chore:, ci:, docs:, style:, refactor:, perf:, test:, and others.

```
<type>[optional scope]: <description>

[optional body]

[optional footer(s)]
```

```
docs: correct spelling of CHANGELOG
```

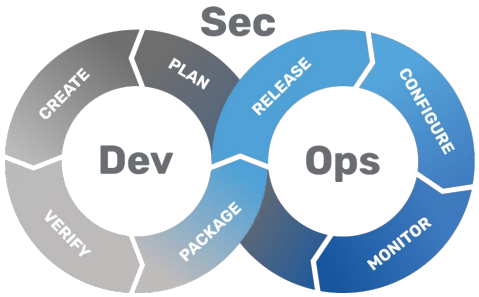
```
fix: prevent racing of requests
```

```
feat(api)!: send an email to the customer when a product is shipped
```

```
feat: allow provided config object to extend other configs

BREAKING CHANGE: `extends` key in config file is now used for extending other config files
```


https://www.conventionalcommits.org/en/v1.0.0/

Git Repository Strategy



Transformation

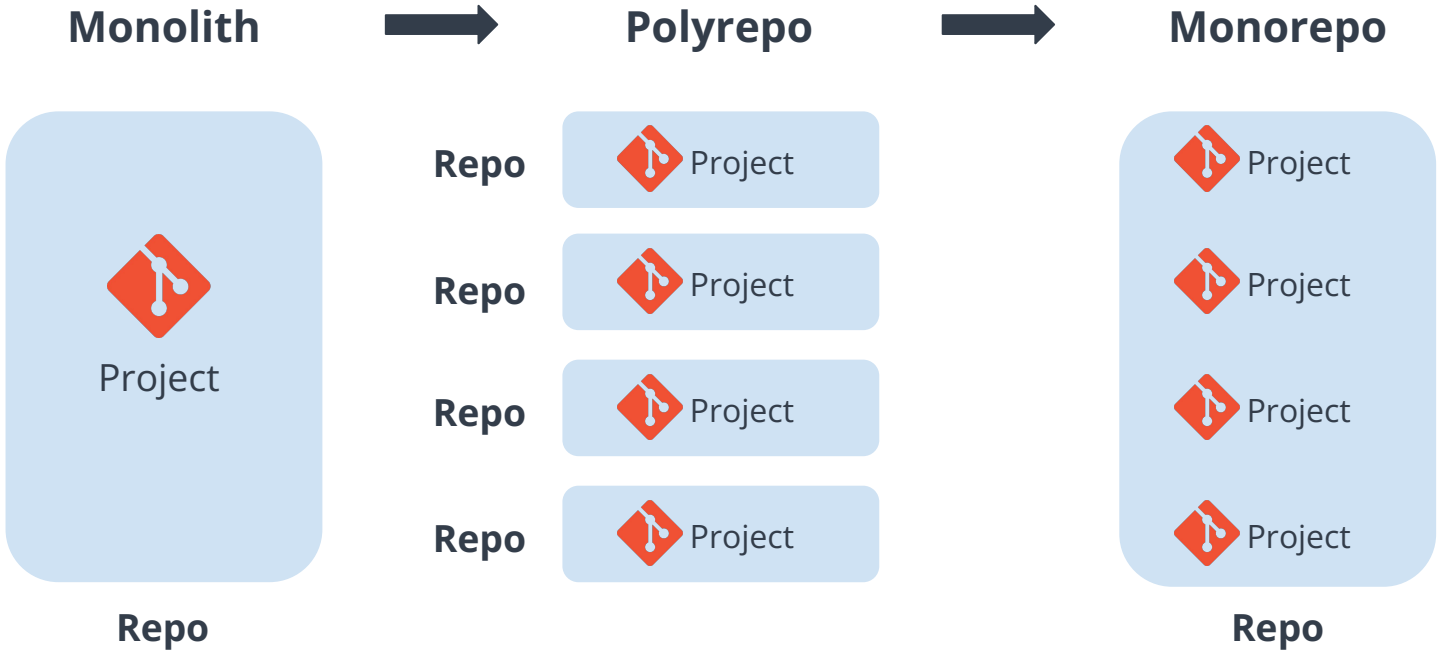
GitLab

git

Opsta Education

Skooldio

Git Repository Strategy



Monorepo Pros-Cons

Pros

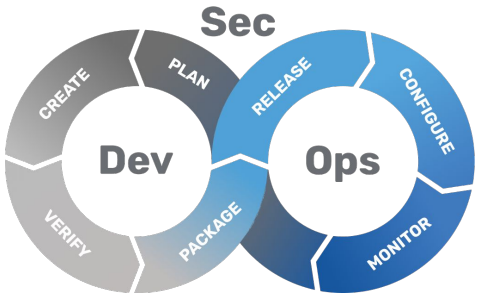
- One source of truth
- Consistent dependency management
- Consistent coding practices
- Visibility
- Code reuse
- Easy to adding a new project
- Breaking changes are easy to tackle
- Unified CI/CD
- Unified build process

Cons

- Poor git performance
- Broken master
- Learning curve
- Difficult code ownership
- Difficult code reviews
- Difficult testing
- Unable to restrict access

https://hannadrehman.com/monorepo-at-healthifyme

Git Branching Strategies



Transformation

GitLab

git

Opsta Education

Skooldio

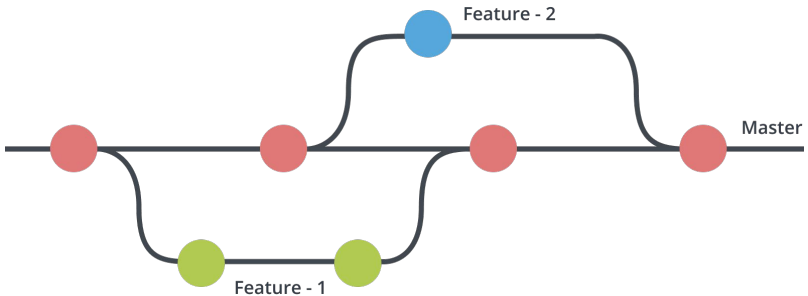
Choosing Git Branching Strategy

By Concept

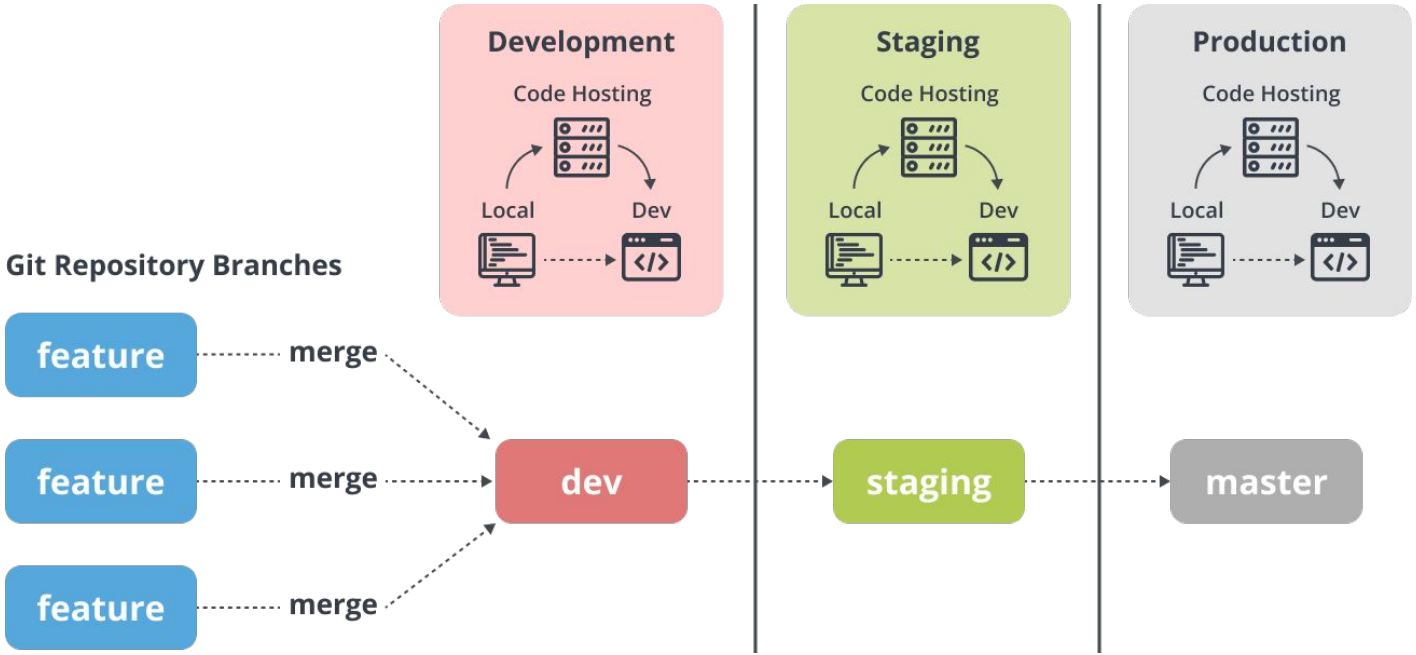
- Environment Branch Strategy
- Feature Branch Strategy
- Scheduled Release Strategy

Full Proposing Idea

- GitLab Flow
- GitHub Flow
- Git Flow

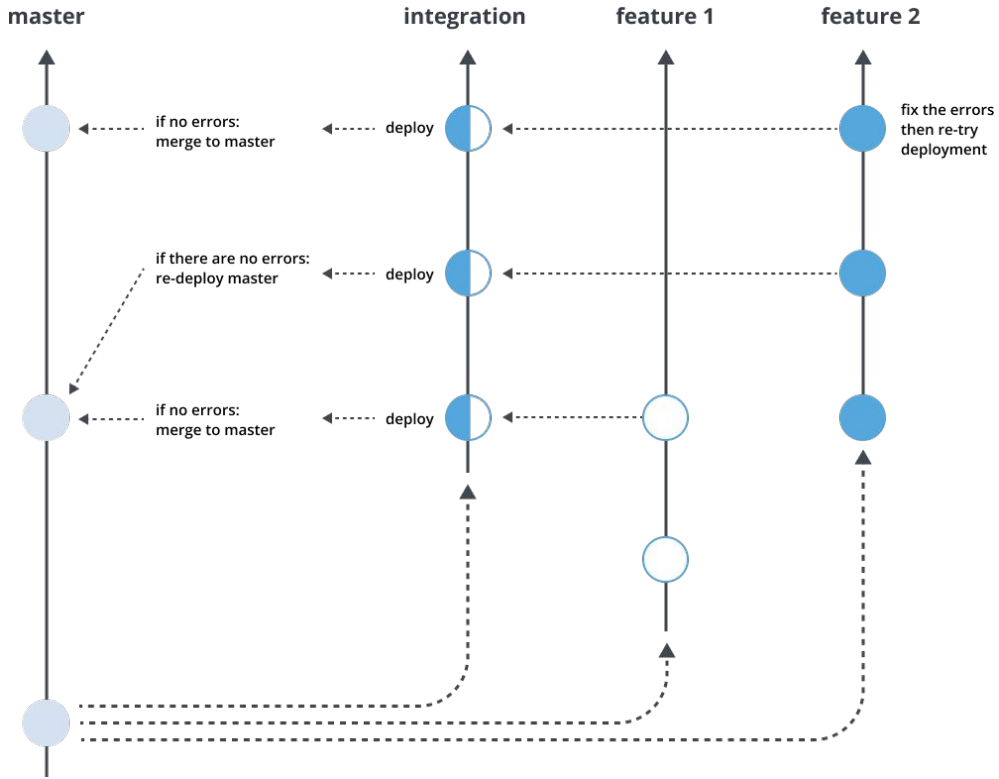


Environment Branch Strategy



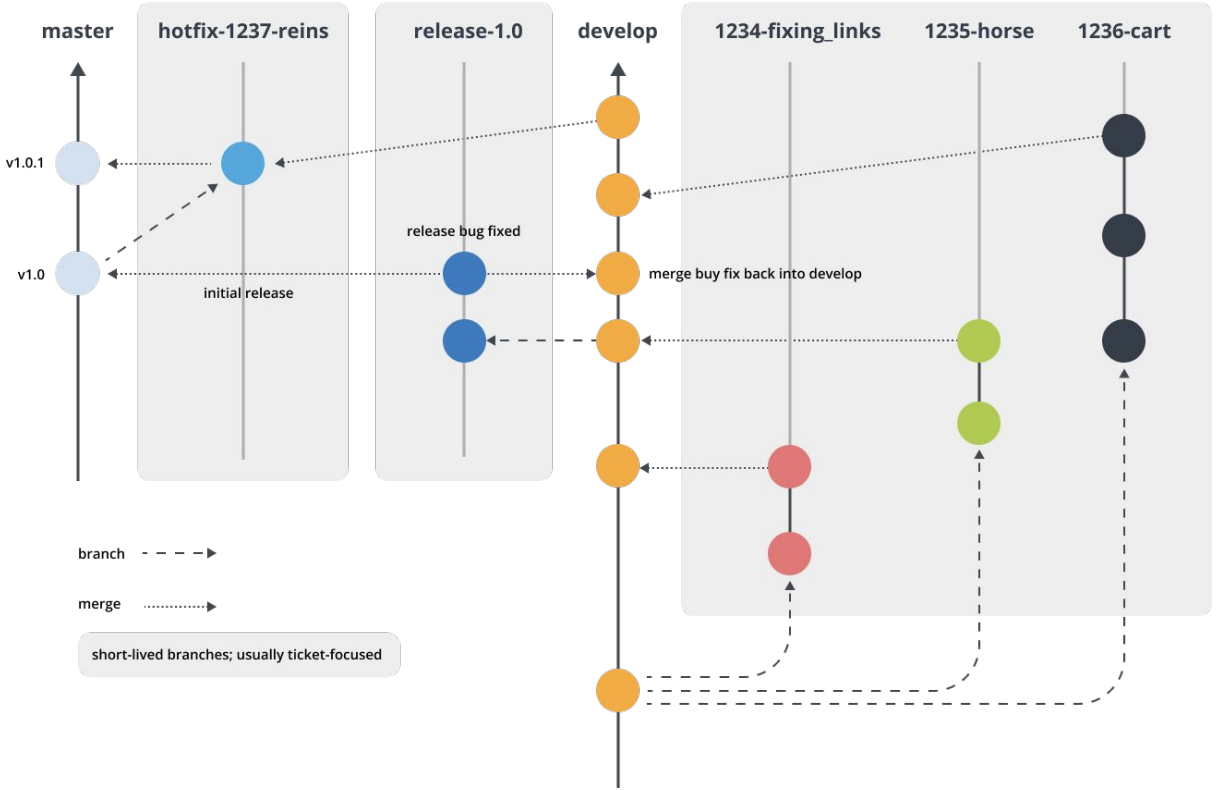
http://emmajane.github.io/git-adventure-mapping/#/4/5

Feature Branch Strategy



http://emmajane.github.io/git-adventure-mapping/#/4/6

Scheduled Release Strategy



http://emmajane.github.io/git-adventure-mapping/#/4/7

Patterns for Managing Source Code Branches

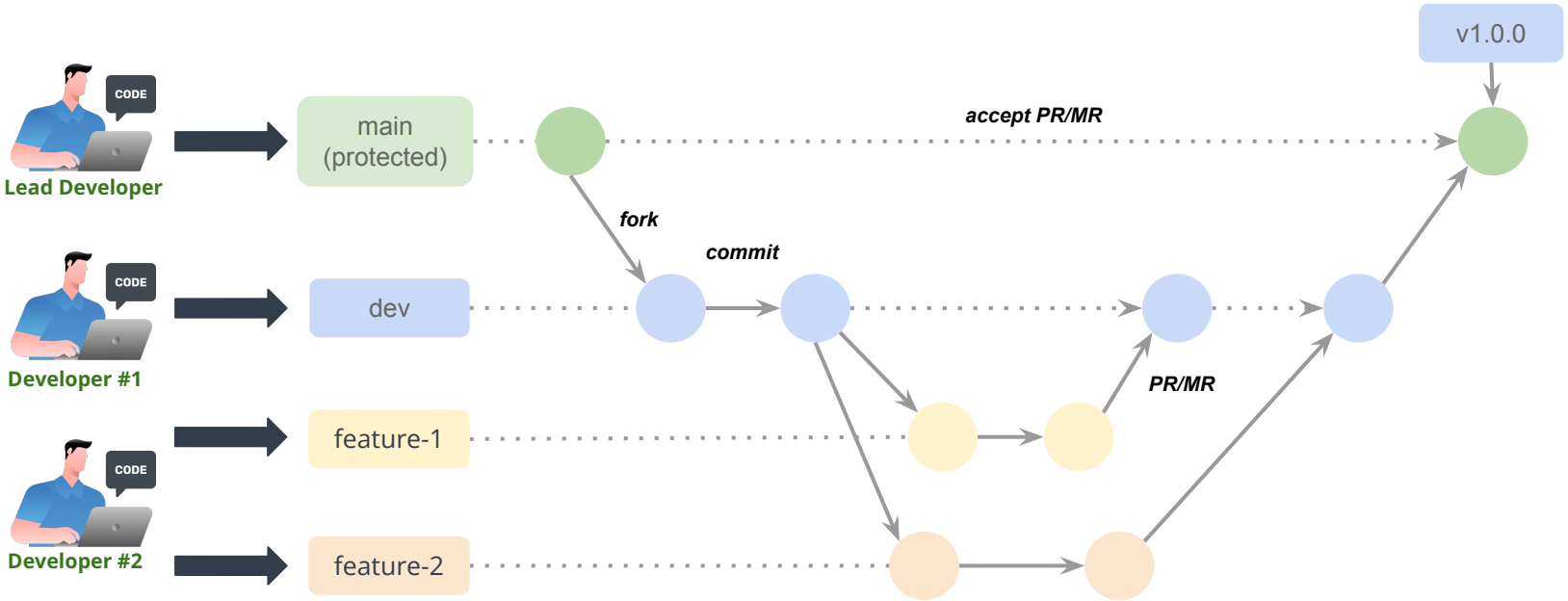
Martin Fowler

CONTENTS

https://martinfowler.com/articles/branching-patterns.html

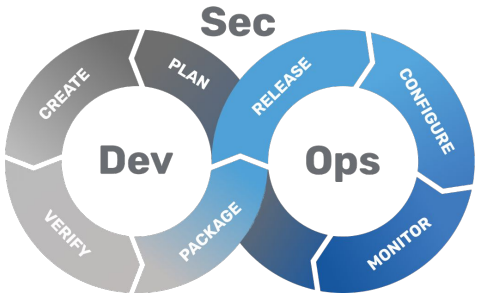
Git Branching Strategy in this workshop

Git Branching means you diverge from the main line of development and continue to do work without messing with that main line



Skooldio Opsta Education

SSH with Public/Private Key



Transformation

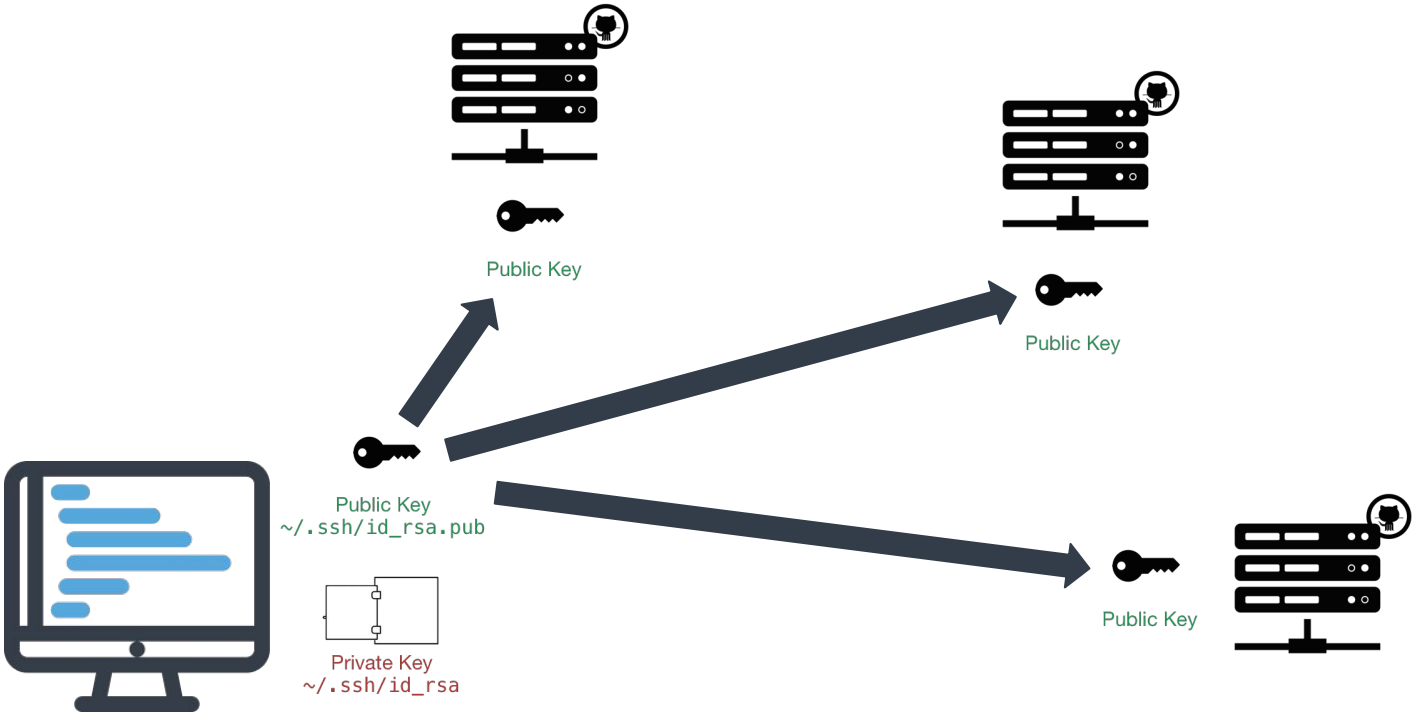
GitLab

git

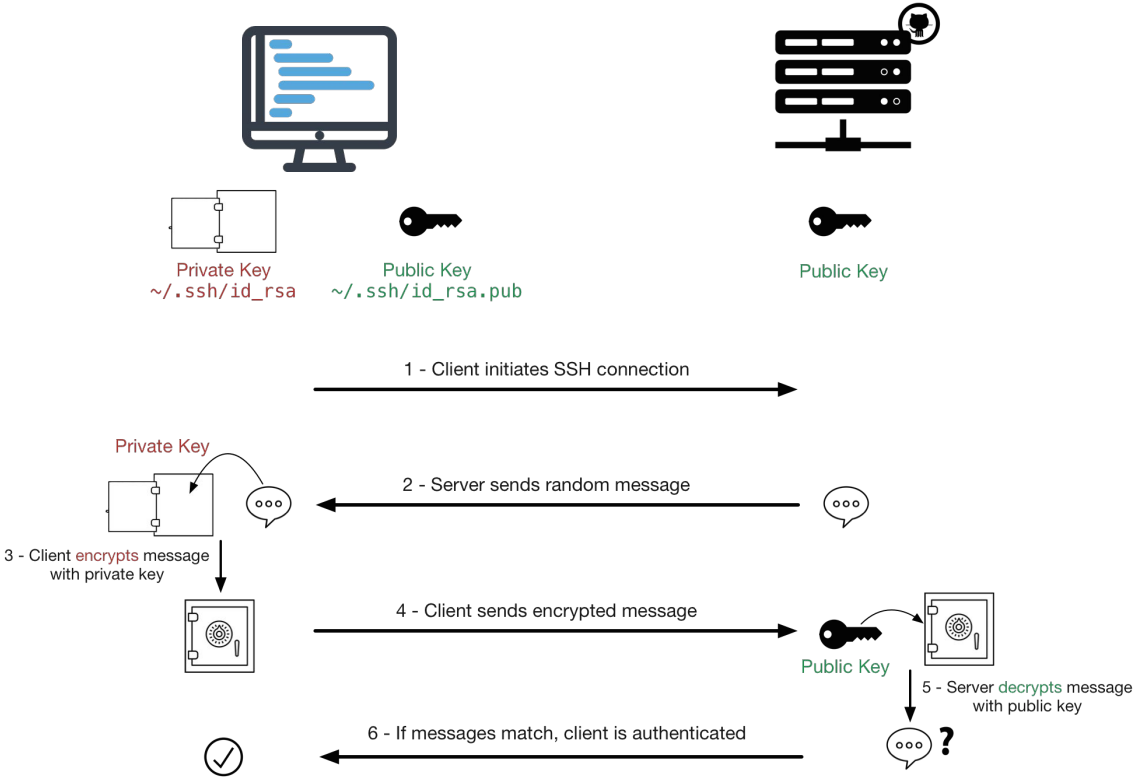
Opsta Education

Skooldio

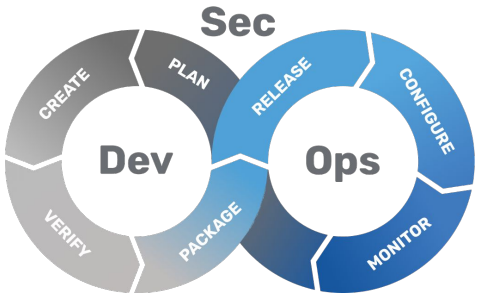
Public and Private Key



Authentication before git push



Markdown Syntax



Transformation

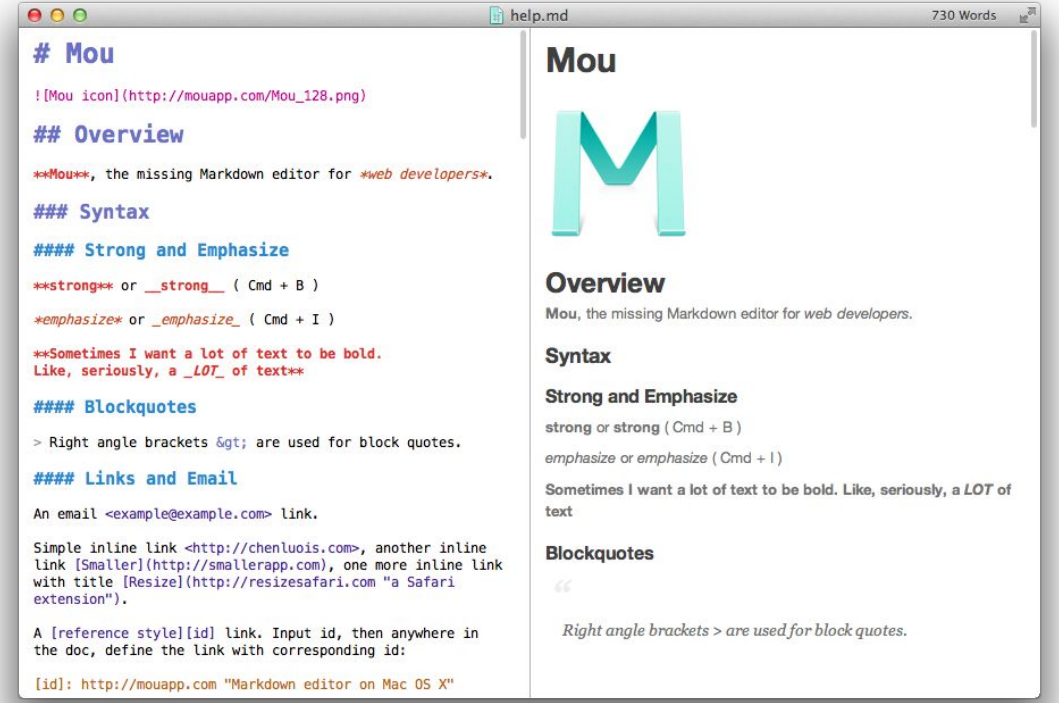
GitLab

git

Opsta Education

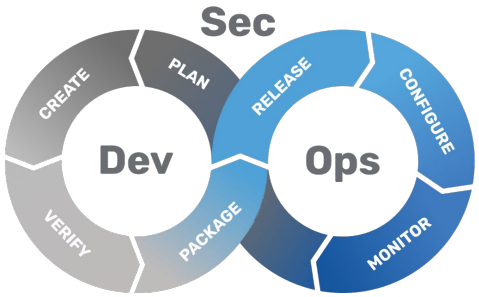
Skooldio

Markdown



https://www.markdowntutorial.com

Git & GitLab Workshop



Transformation

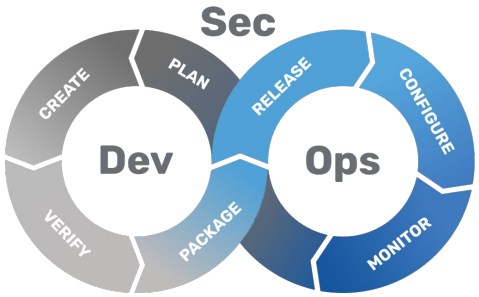
GitLab

git

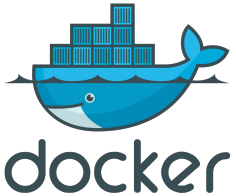
Opsta Education

Skooldio

Introduction to Docker



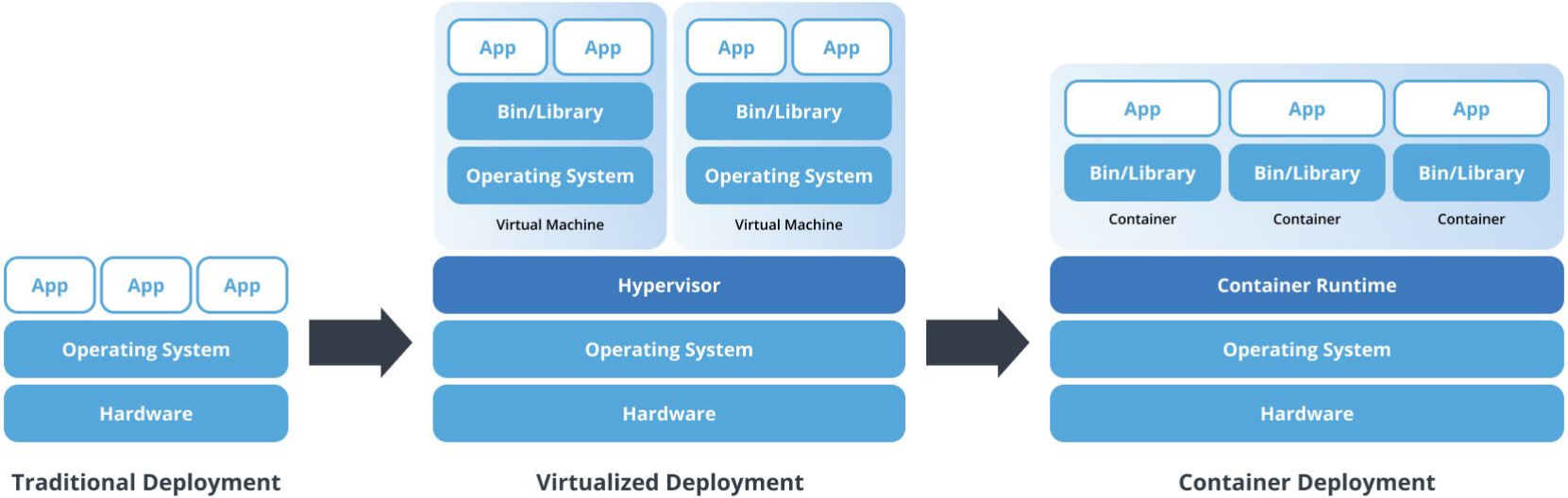
Transformation



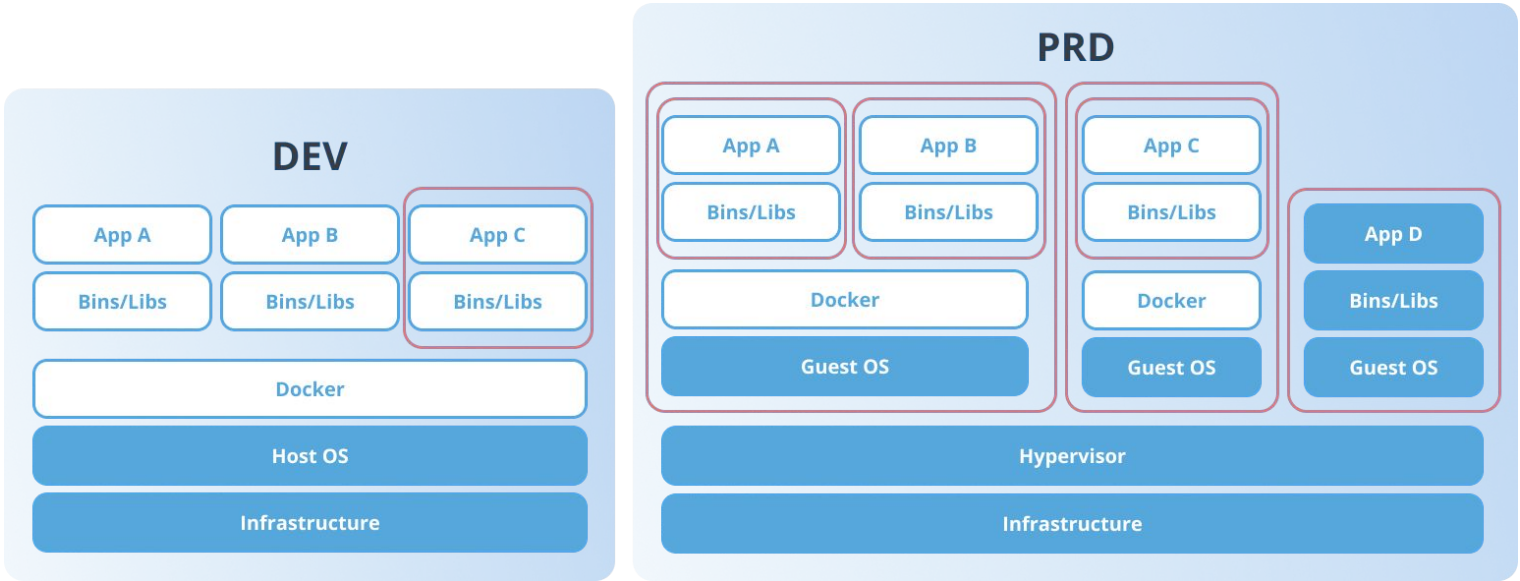
Opsta Education

Skooldio

Evolution of Deployment



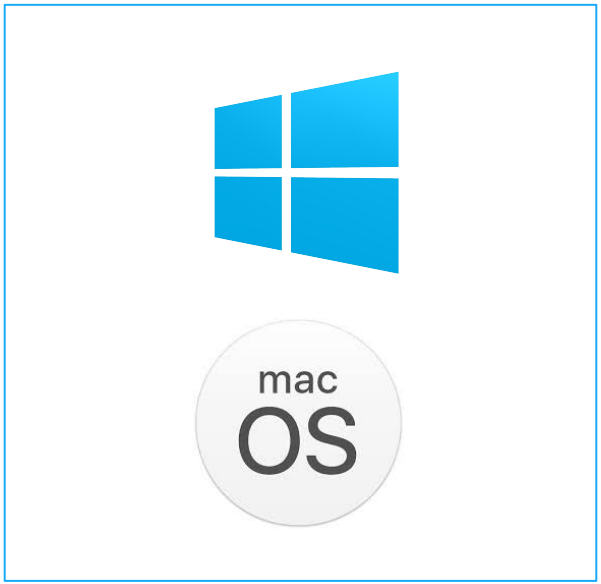
VM and Container together



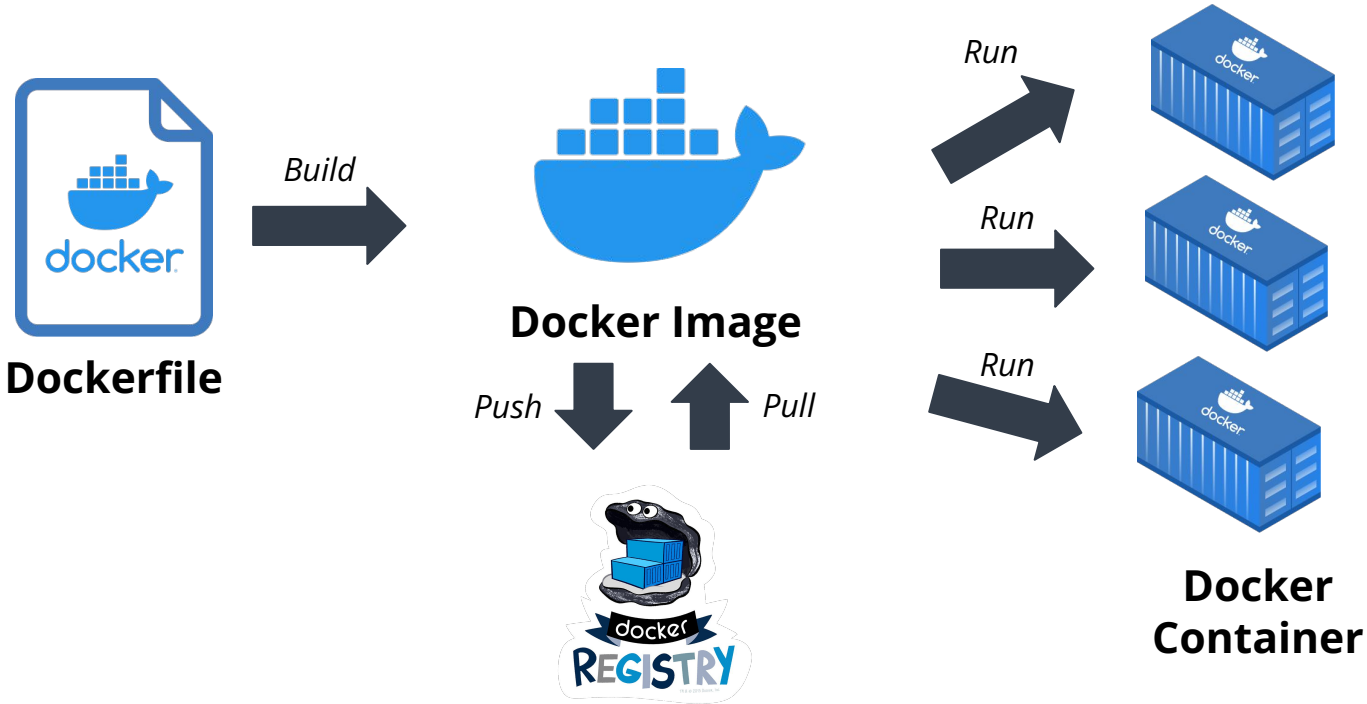
Where Docker can run

Native

VM



Docker Lifecycle

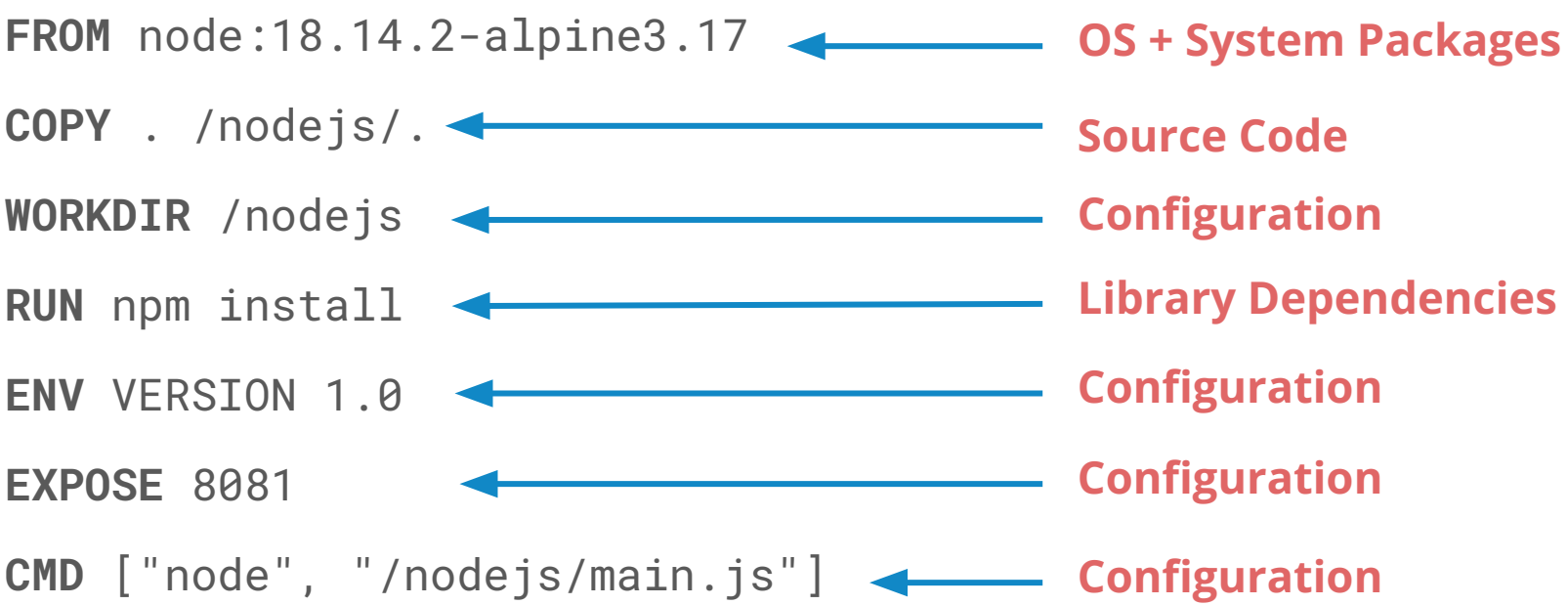


Dockerfile

- Dockerfile is instructions to build Docker Image
  - How to run commands
  - Add files or directories
  - Create environment variables
  - What process to run when launching container
- Result from building Dockerfile is Docker Image

Dockerfile

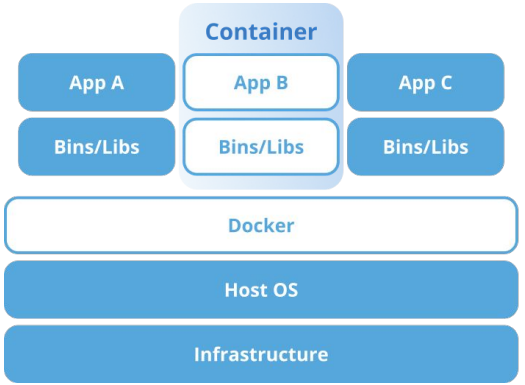
Sample Dockerfile



Docker Image

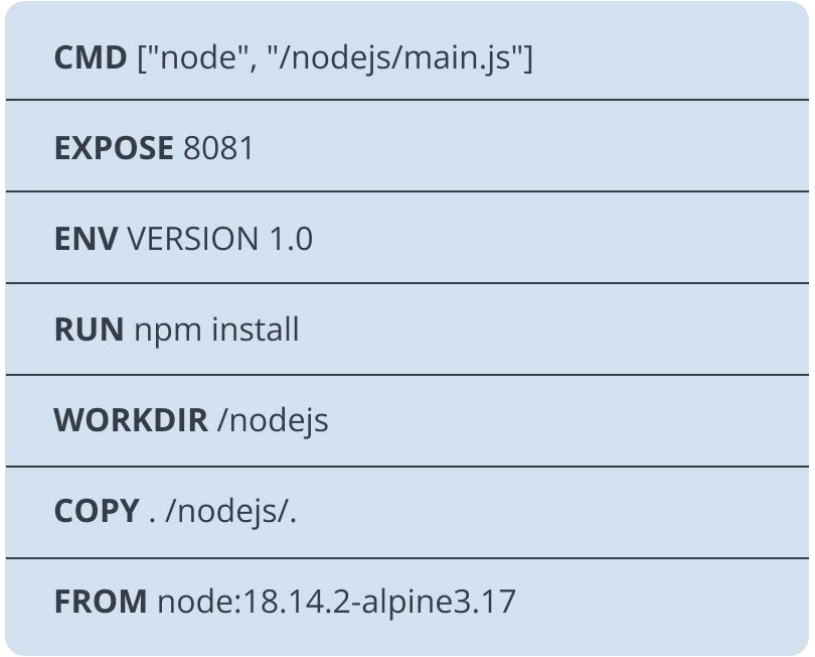
A Docker image is a lightweight, standalone, and executable package that contains all the necessary components to run an application. It includes

- **Base OS** (Ubuntu, CentOS, Alpine)
- **System Packages** (APT, YUM)
- **Library Dependencies** (PIP, NPM, Composer)
- **Configuration**
- **Source Code**

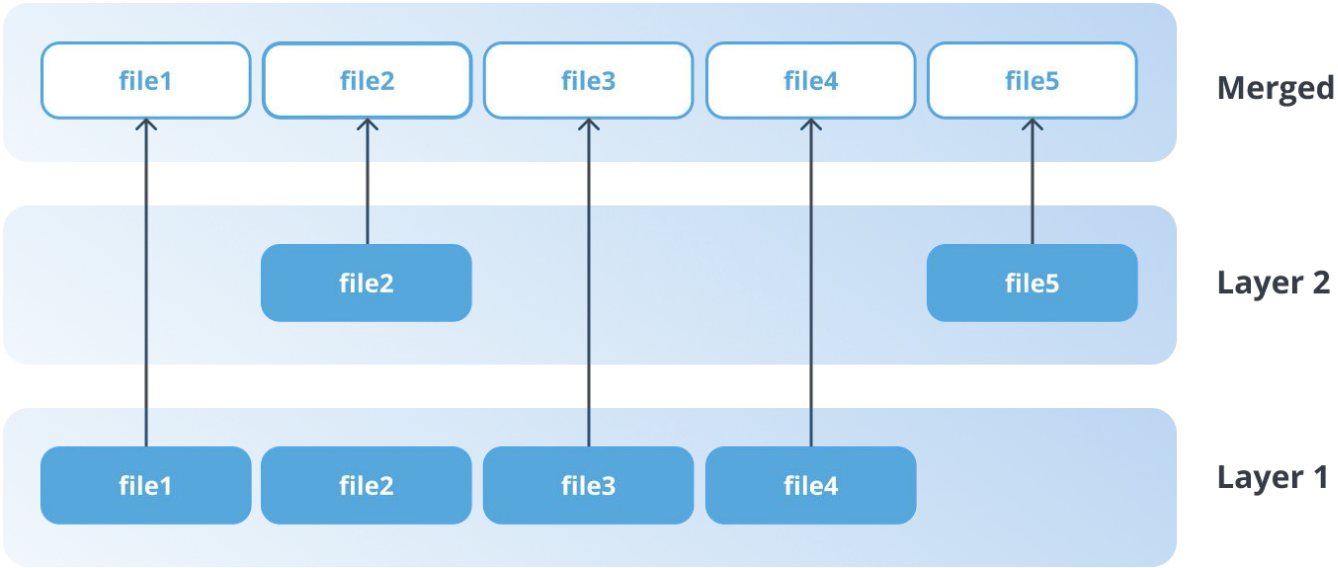


Dockerfile and Docker Image Layer

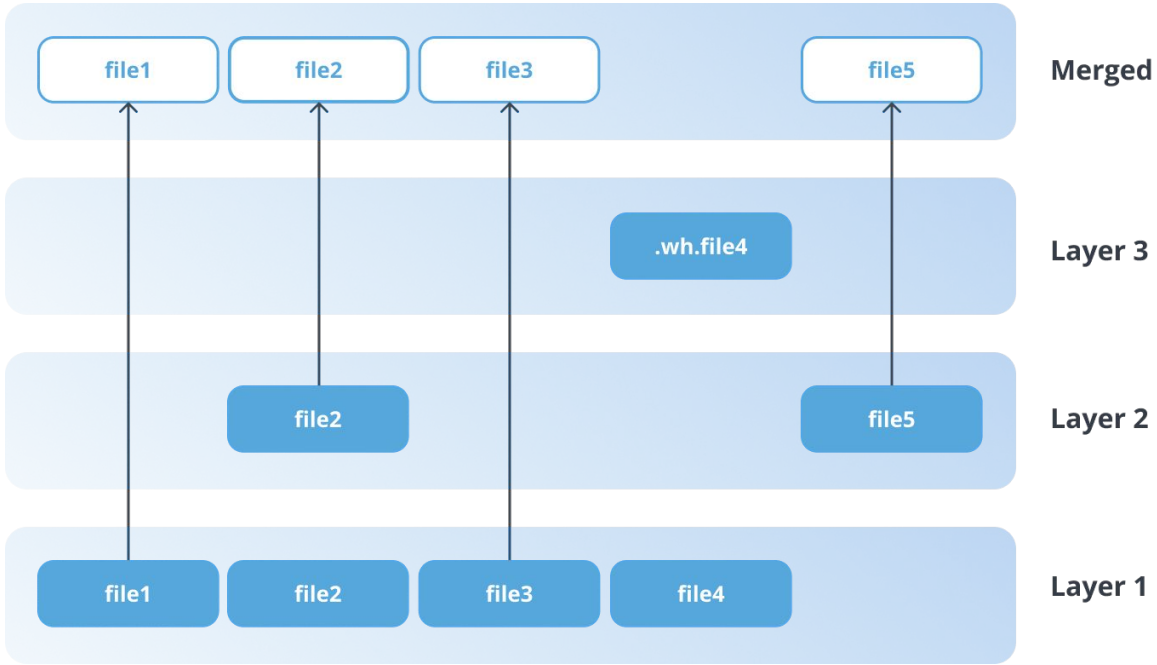
```
# Dockerfile
FROM node:18.14.2-alpine3.17
COPY . /nodejs/.
WORKDIR /nodejs
RUN npm install
ENV VERSION 1.0
EXPOSE 8081
CMD ["node", "/nodejs/main.js"]
```



Docker Image Layer (1)



Docker Image Layer (2)



Size of Docker Image

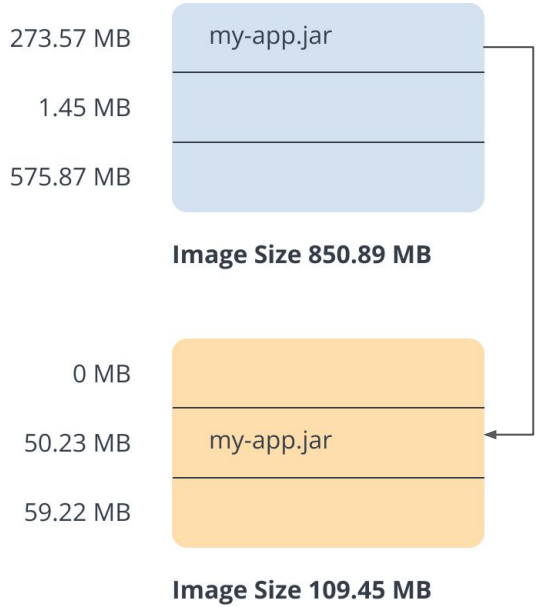
Total Image Size 850.89 MB

0 MB	**CMD** ["java", "-jar", "my-app.jar"]
0 MB	**RUN** rm -rf .m2/*
575.87 MB	**RUN** mvn clean install
1.45 MB	**COPY** . /java/.
273.57 MB	**FROM** maven:3.9.0-eclipse-temurin-17-focal

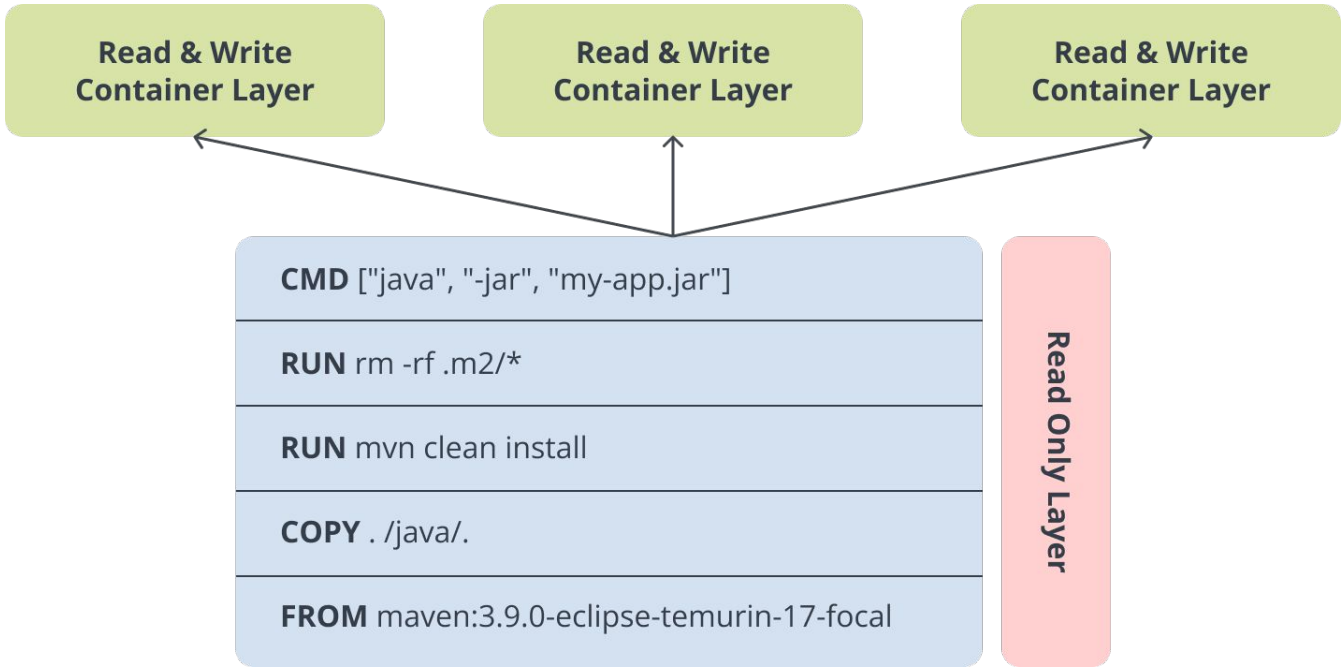
Multi-stage builds

```
# Dockerfile
# Stage 1
FROM maven:3.9.0-eclipse-temurin-17-focal AS builder
COPY . /java/.
RUN mvn clean install
```

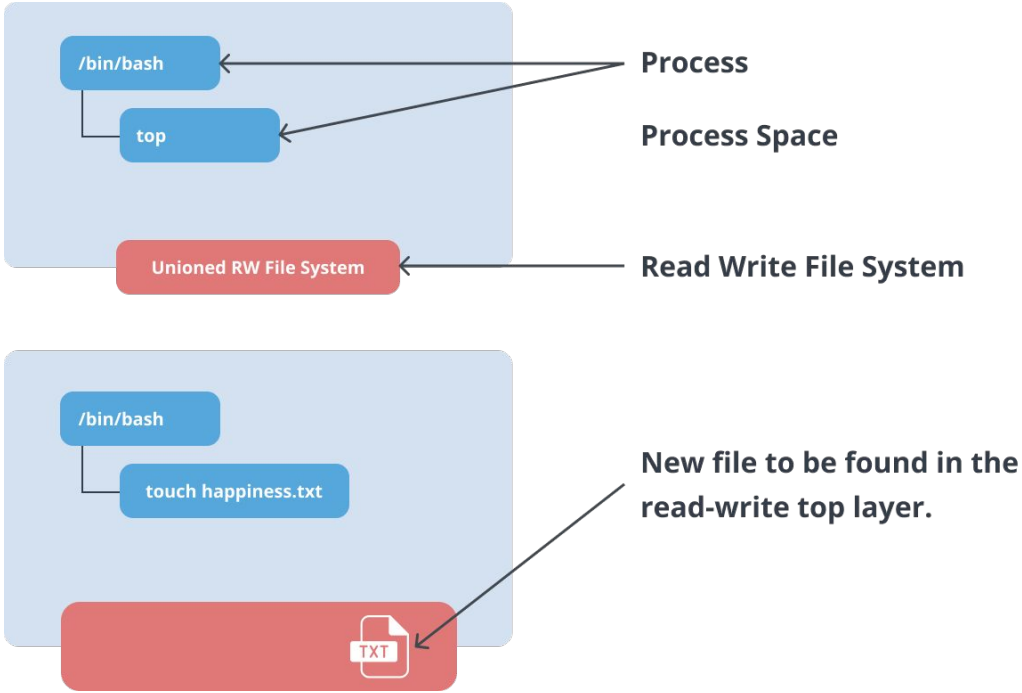
```
# Stage 2
FROM eclipse-temurin:17.0.6_10-jre-alpine
COPY --from=builder target/my-app.jar my-app.jar
ENTRYPOINT ["java”]
CMD ["-jar", "my-app.jar"]
```



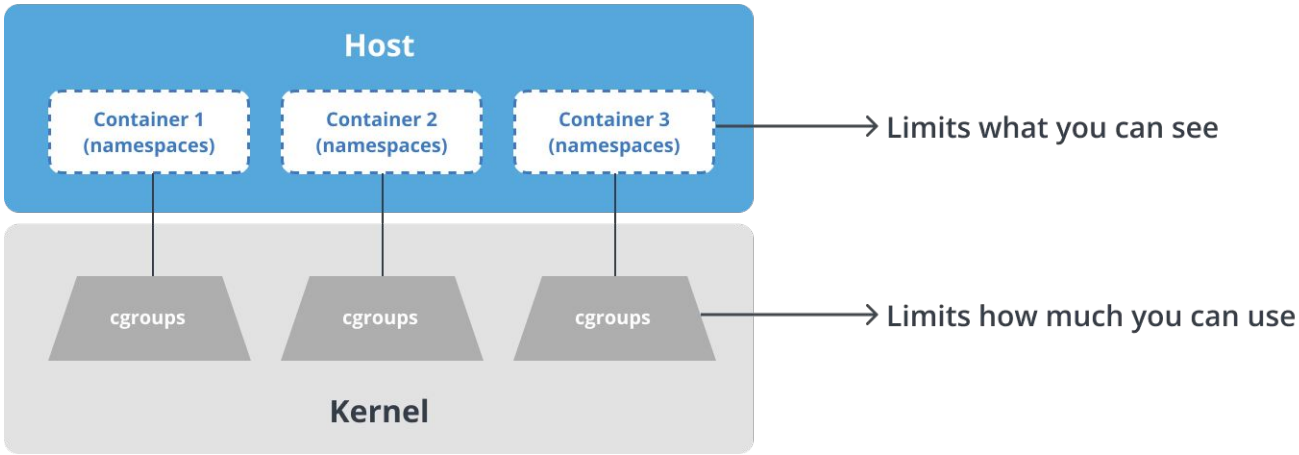
Docker Container



Running Container

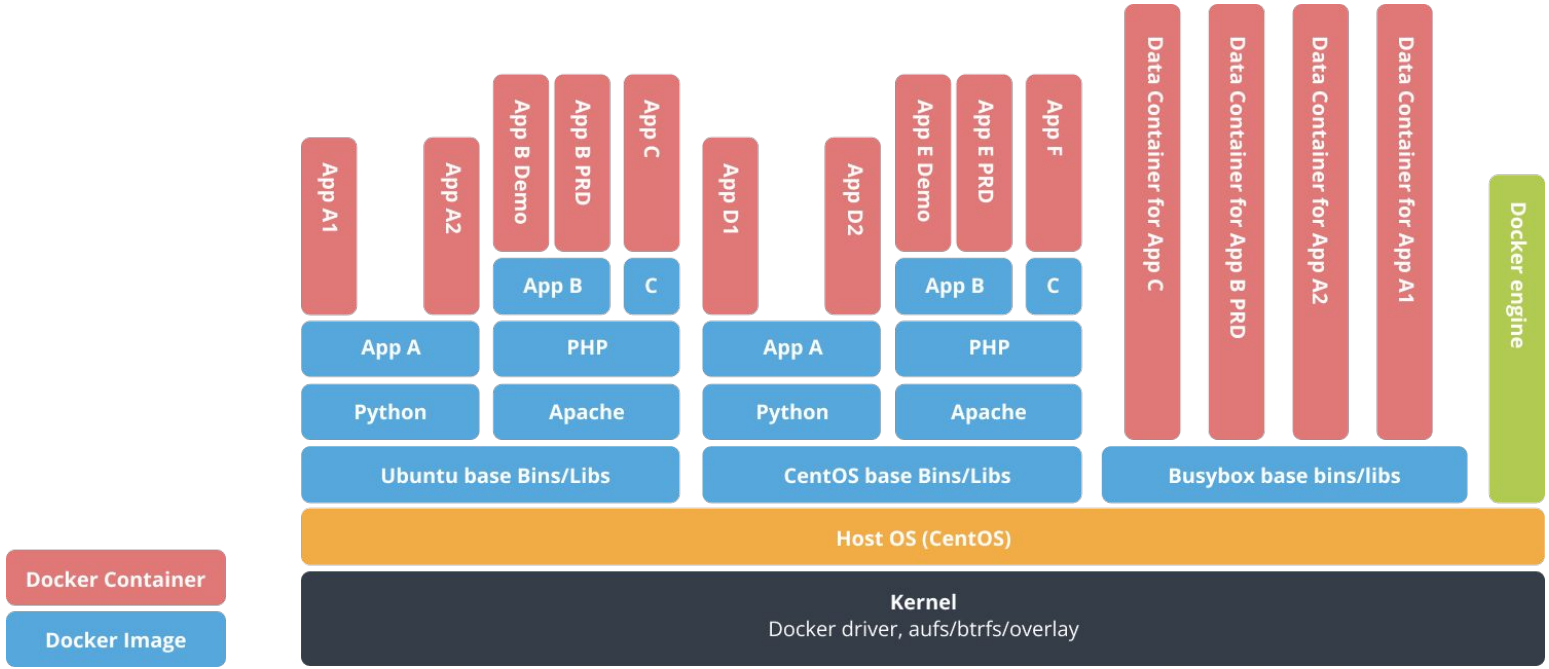


Kernel namespaces and cgroups



https://bikramat.medium.com/namespace-vs-cgroup-60c832c6b8c8

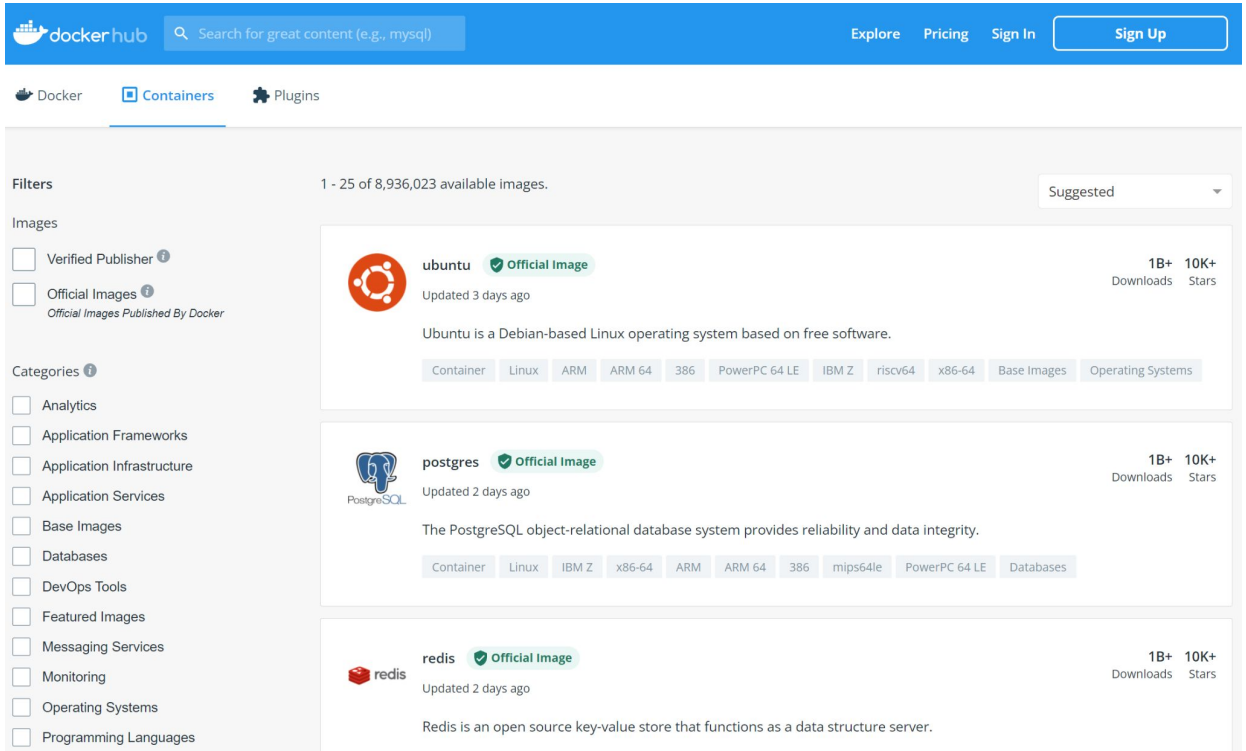
Docker Images and Containers Layers



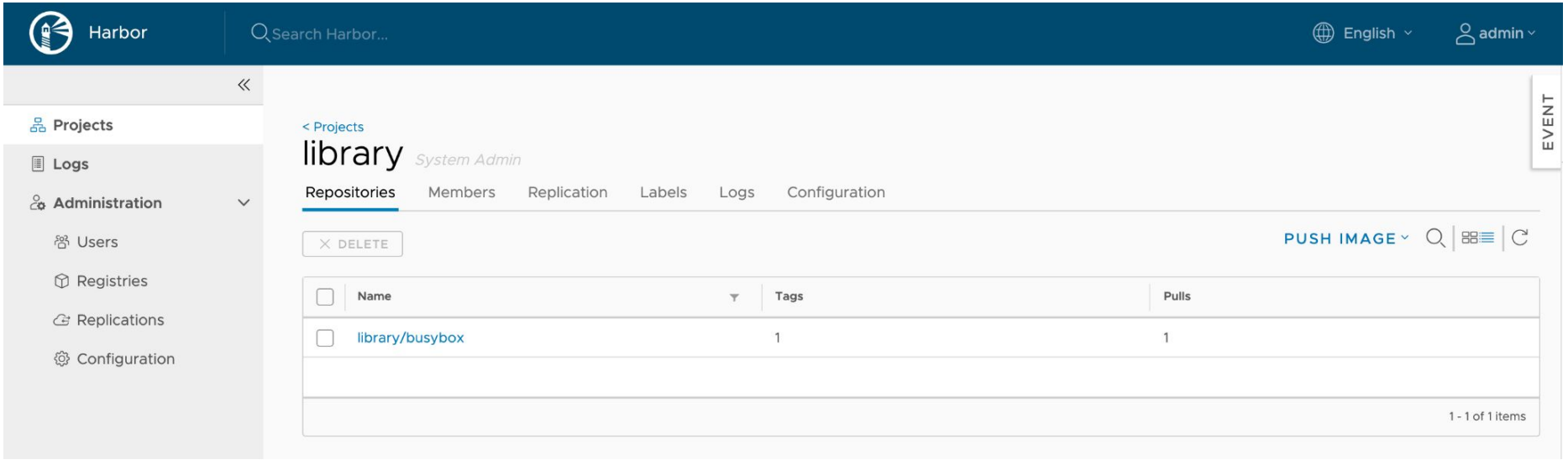
Docker Registry

- Docker Registry is the store for Docker Image
- Docker Hub is public Docker Registry like Github
- Using Docker client to push and pull Docker Image from Docker Registry
- You can create your own Private Docker Registry

Docker Hub


docker hub
Search for great content (e.g., mysql)
Explore
Pricing
Sign In
Sign Up
Docker
Containers
Plugins
Filters
1 - 25 of 8,936,023 available images.
Suggested
Images
Verified Publisher
Official Images
Official Images Published By Docker
Categories
Analytics
Application Frameworks
Application Infrastructure
Application Services
Base Images
Databases
DevOps Tools
Featured Images
Messaging Services
Monitoring
Operating Systems
Programming Languages
ubuntu
Official Image
Updated 3 days ago
1B+ Downloads
10K+ Stars
Ubuntu is a Debian-based Linux operating system based on free software.
Container
Linux
ARM
ARM 64
386
PowerPC 64 LE
IBM Z
riscv64
x86-64
Base Images
Operating Systems
postgres
Official Image
Updated 2 days ago
1B+ Downloads
10K+ Stars
The PostgreSQL object-relational database system provides reliability and data integrity.
Container
Linux
IBM Z
x86-64
ARM
ARM 64
386
mips64le
PowerPC 64 LE
Databases
redis
Official Image
Updated 2 days ago
1B+ Downloads
10K+ Stars
Redis is an open source key-value store that functions as a data structure server.

Harbor



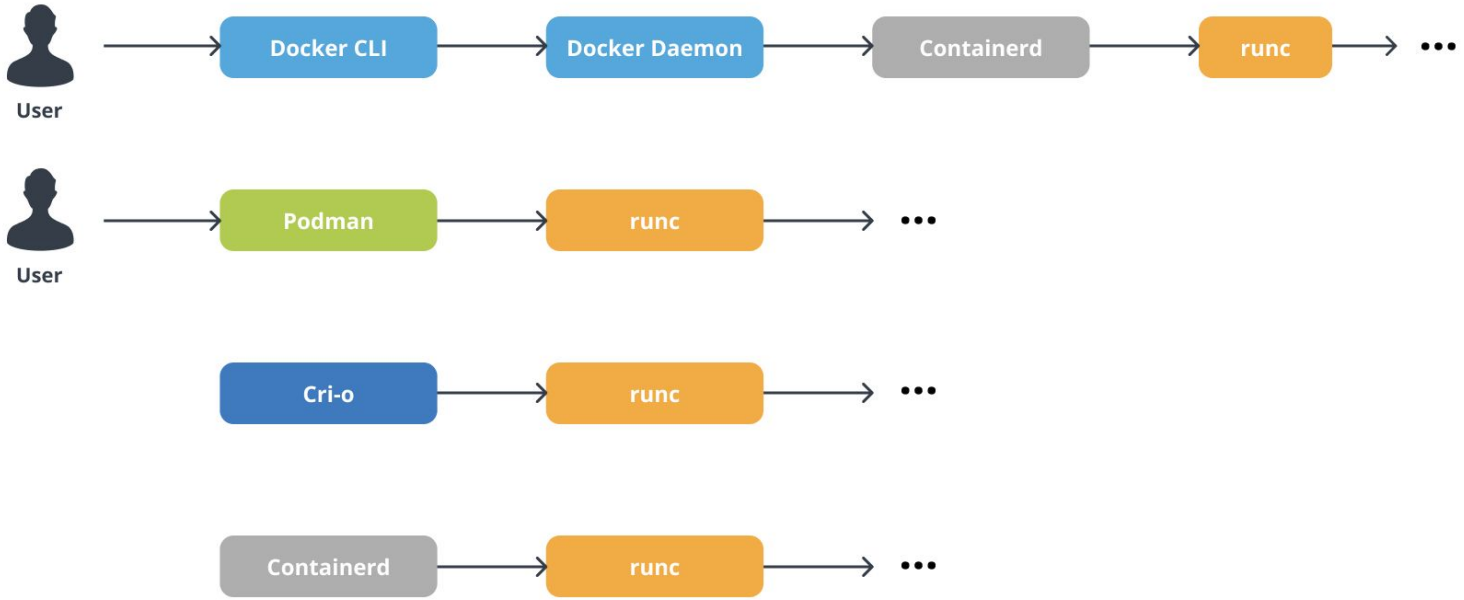
Skooldio Opsta Education

Open Container Initiative (OCI)

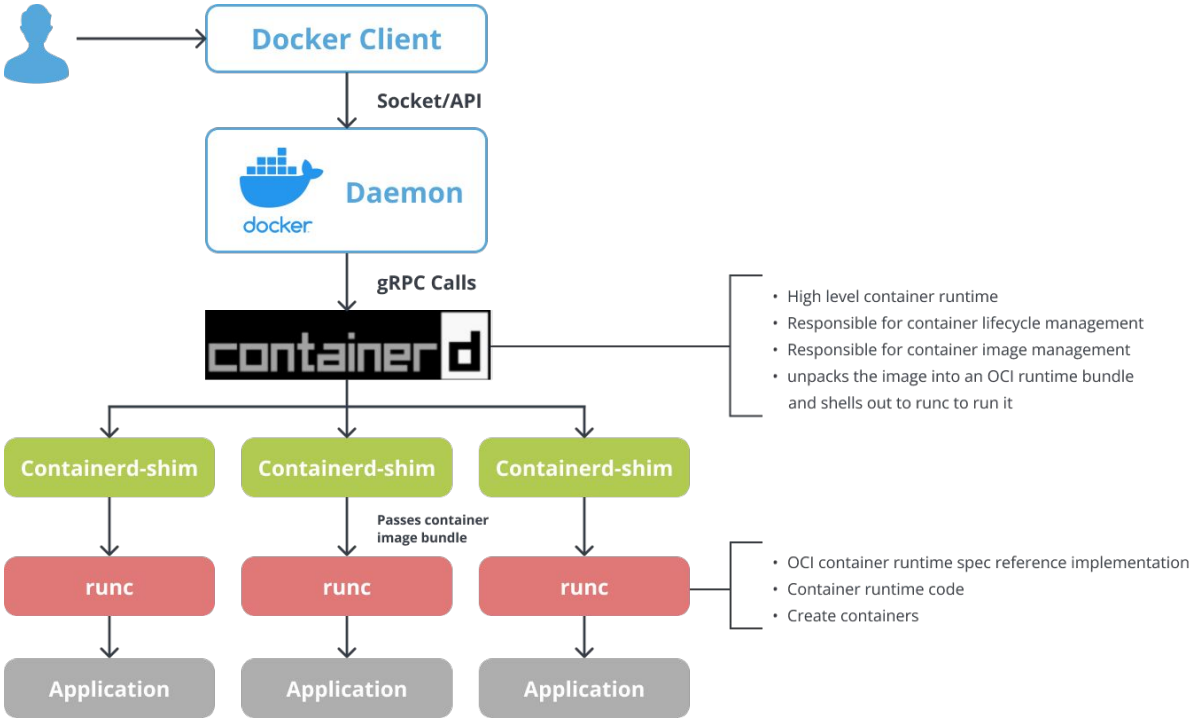
The **Open Container Initiative (OCI)** is an open governance structure for the express purpose of creating **open industry standards** around **container formats and runtimes**. OCI was launched on June 22nd 2015 by Docker, CoreOS and other leaders in the container industry.

- Runtime Specification (runtime-spec)
- Image Specification (image-spec)
- Distribution Specification (distribution-spec)

Open Container Initiative (OCI) Runtimes

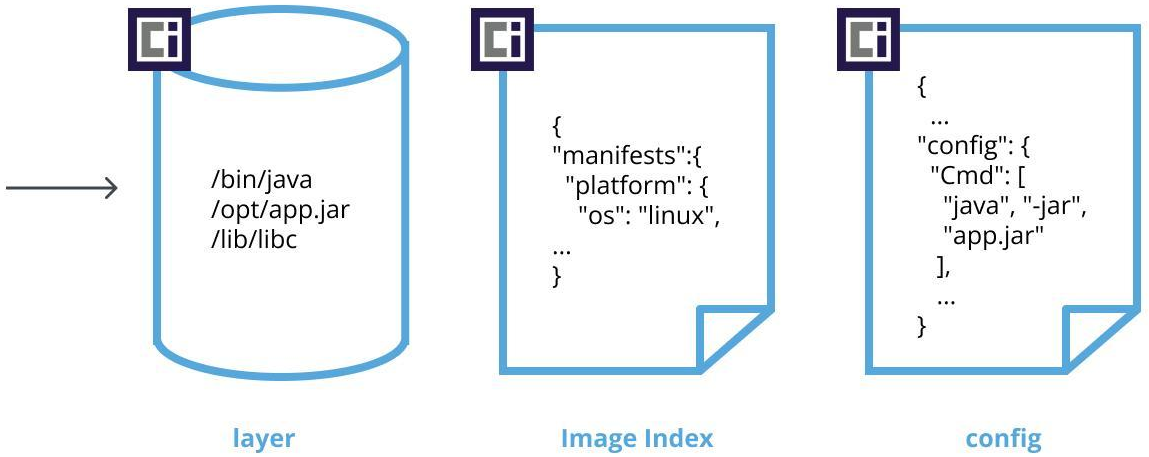


Docker Engine and Containerd

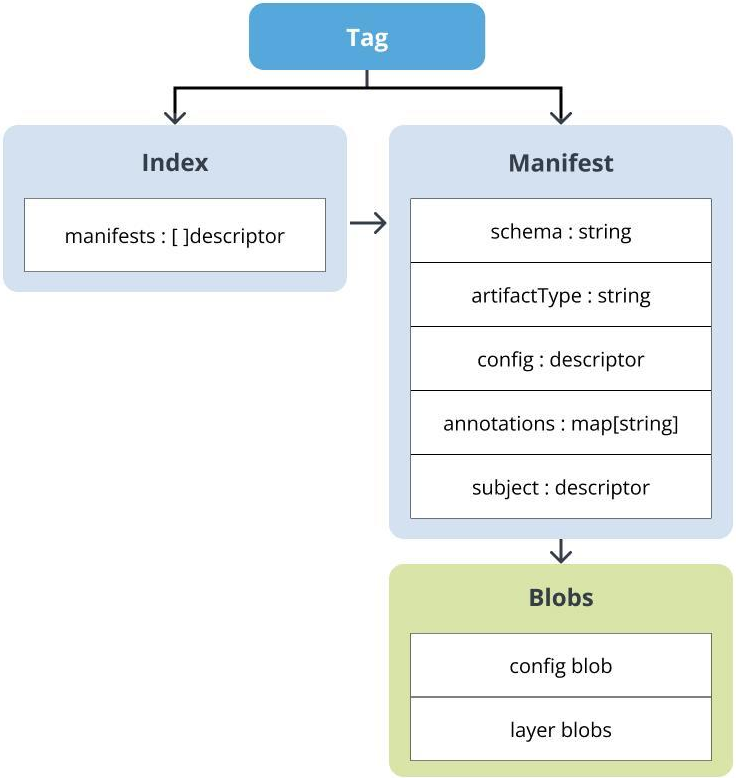


OCI Image Specification

```
public class Helloworld f
    public static void main(String[] args) {
        System.out.println("Hello, World");
    }
}
```



OCI Distribution Specification and Artifacts



"application/vnd.**docker.container.image**.v1+json"

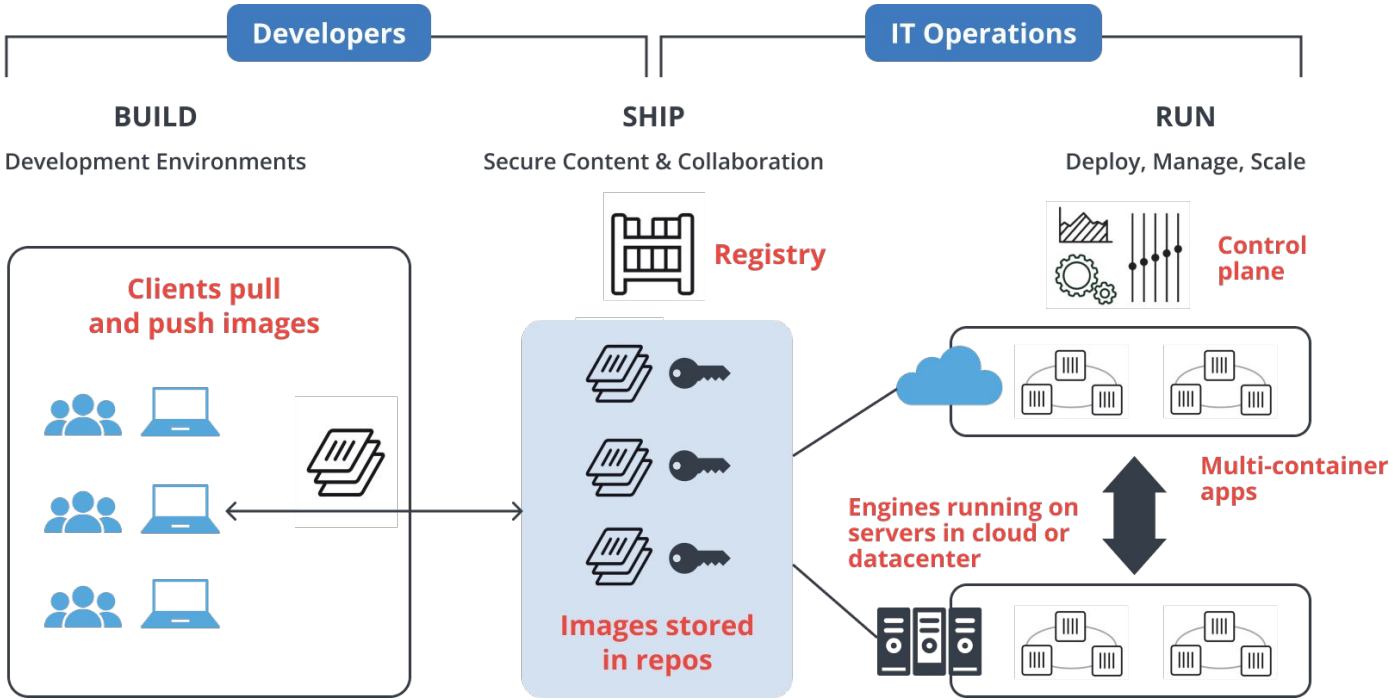
"application/vnd.**oci.image**.config.v1+json"

"application/vnd.**cncf.helm.chart**.config.v1+json"

"application/vnd.**sylabs.sif**.config.v1+json"

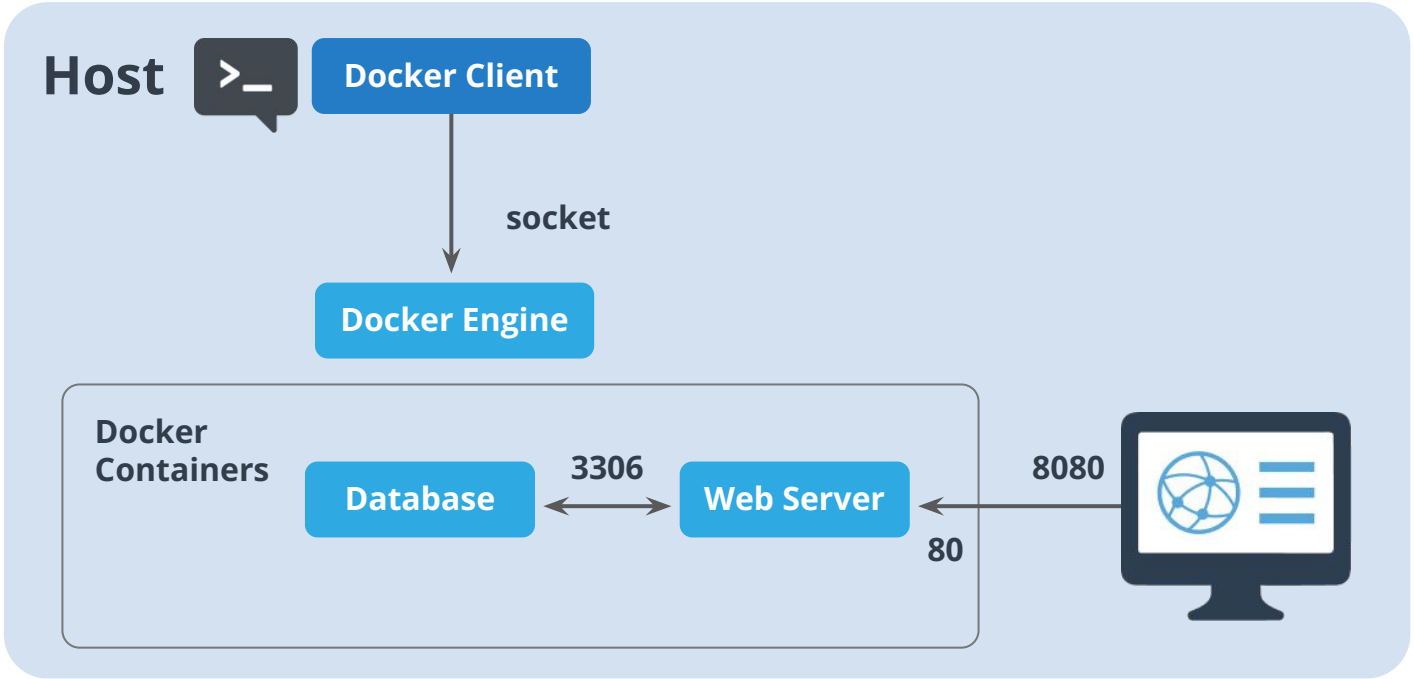
"application/vnd.cncf.openpolicyagent.config.v1+json"

Docker and DevOps

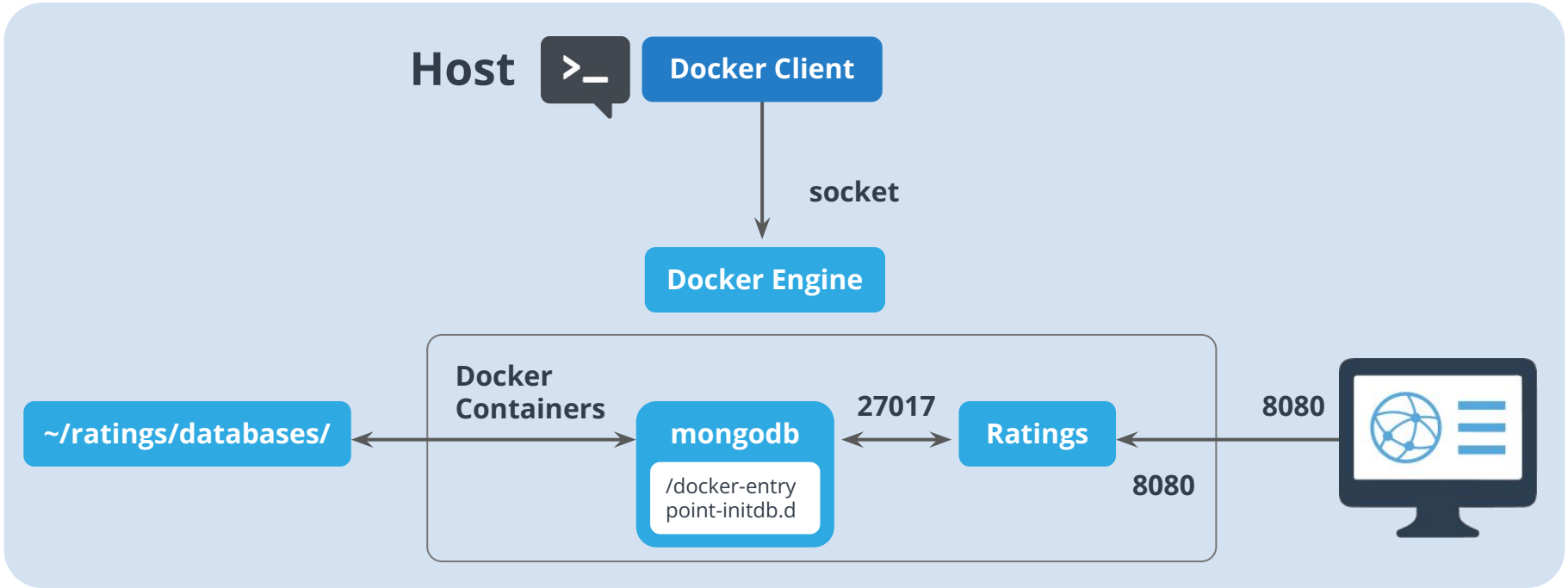


Skooldio Opsta Education

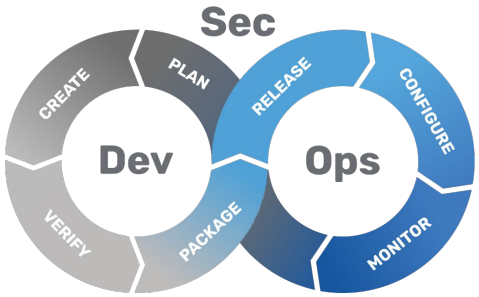
Docker Architecture



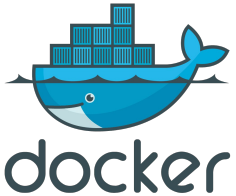
Ratings Service



Docker Workshop



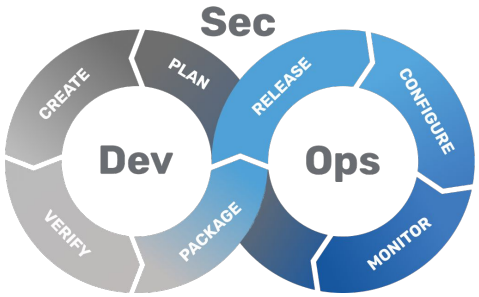
Transformation



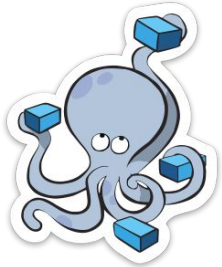
Opsta Education

Skooldio

Docker Compose



Transformation



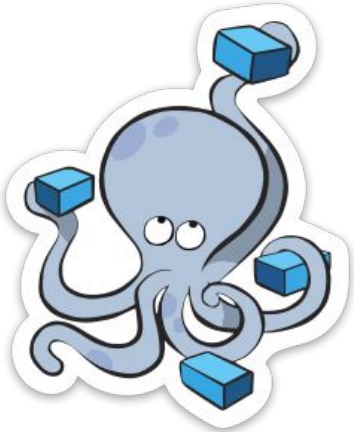
Opsta Education

Skooldio

Docker Compose

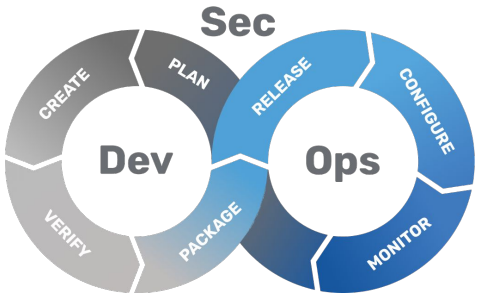
Tool for defining and running multi-container applications with YAML configuration file(s)

```
services:
  web:
    build: .
    ports:
    - "5000:5000"
  redis:
    image: "redis:alpine"
```

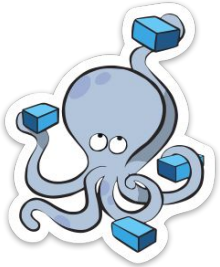


https://collabnix.com/docker-compose-cheatsheet/

YAML



Transformation

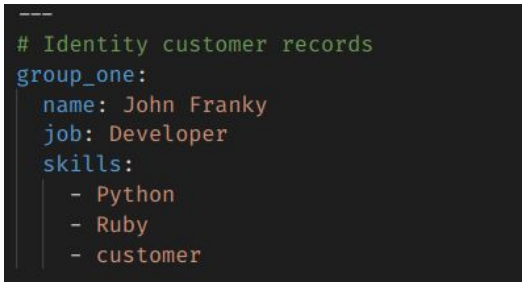


Opsta Education

Skooldio

YAML

- YAML is markup language. It is **human-readable** and easy to understanding syntax.
- YAML has format for specific **configuration** information such as Docker Compose or Kubernetes definitions.
- Similar **JSON language**
- Advantage to use YAML format
  - **Infrastructure as a code** (IaC) - keep it on standard template.
  - **Stability** - keep it on version control, keep history change.
  - **Flexibility** - higher dynamic than use command line.



YAML Basic Rules

- YAML files should end in *.yaml* or *.yml*
- YAML is case sensitive
- YAML structure with indentation
- YAML does not allow the use of tabs
- YAML indentation level can be one or more spaces
- YAML Data Types
    - scalars (strings / numbers)
    - sequences (arrays / lists)
    - mappings (hashes / dictionaries)

YAML Scalars

- A scalar could be a **boolean** property, **integer** (number), or a **string**
- Scalars are often called variables in programming
- Most scalars are unquoted, but if you are typing a string that uses punctuation and other elements that can be confused with YAML syntax (dashes, colons, etc.) you may want to quote this data using single ' or double " quotation marks. Double quotation marks allow you to use escapings to represent ASCII and Unicode characters.
- *true*, *false*, *null*, ~ and *dates* have special meaning in YAML. Please quote them if you do not want to use them as a *boolean*, *null* or *datetime* type
- *version numbers*, they should be quoted to separate them from *float* values.

YAML Sequences

- It is a basic list with each item in the list placed in its own line with an opening dash.

YAML Mappings

- Mapping gives you the ability to list keys with values. This is useful in cases where you are assigning a name or a property to a specific element.

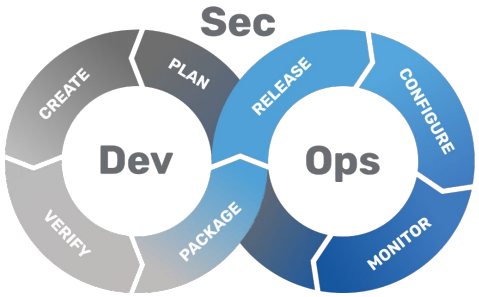
YAML Data Types

```yaml
---
# Scalars
doe: "a deer, a female deer"
ray: "a drop of golden sun"
pi: 3.14159
xmas: true
xmas-str: "true"
rench-hens: 3
```

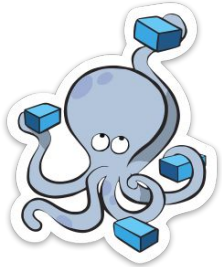
```yaml
---
# Sequences
calling-birds:
- huey
- dewey
- louie
- free

---
# Sequences
calling-birds:
  - huey
  - dewey
  - louie
  - free
```

```yaml
---
# Mappings
xmas-fifth-day:
  calling-birds: four
  french-hens: 3
  golden-rings: 5
  partridges:
    count: 1
    location: "a pear tree"
  turtle-doves: two
```

Docker Compose Workshop



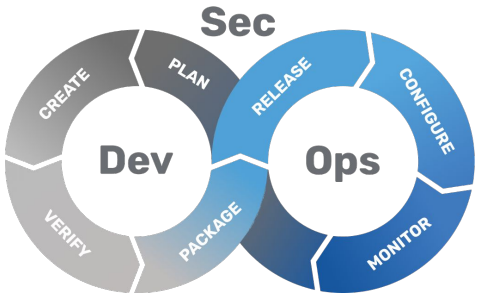
Transformation



Opsta Education

Skooldio

Introduction to Kubernetes



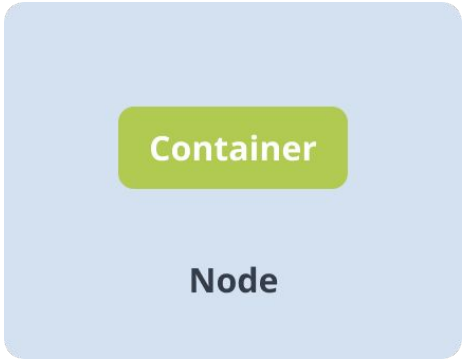
Transformation

kubernetes

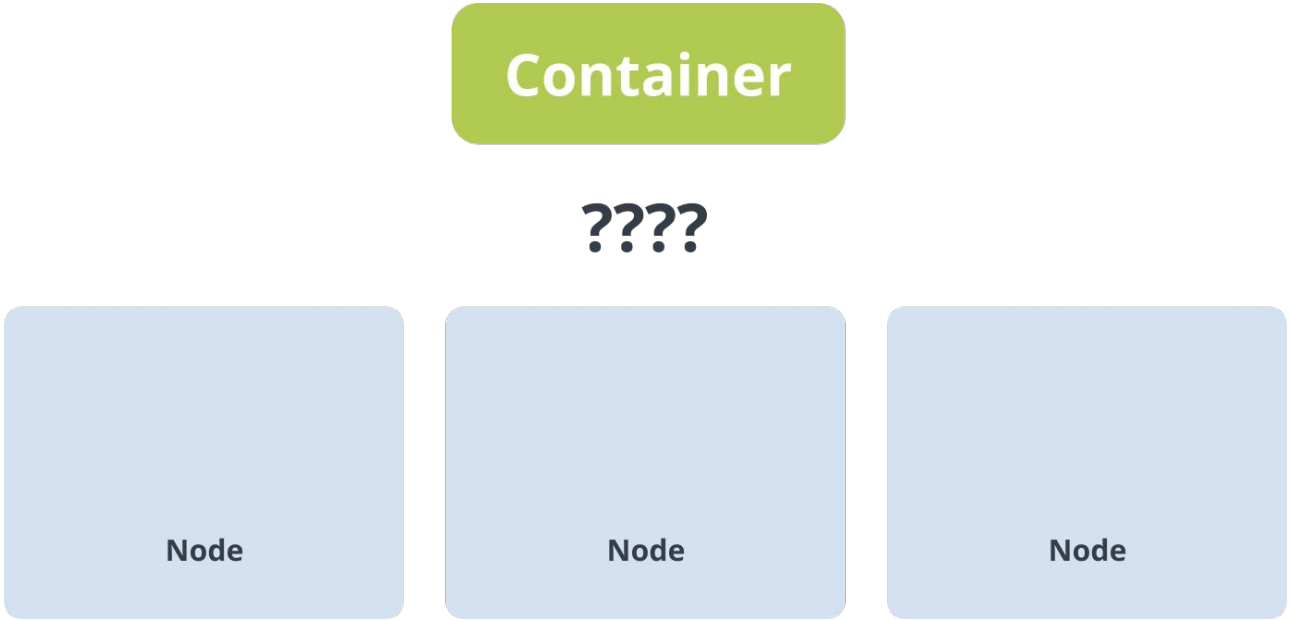
Opsta Education

Skooldio

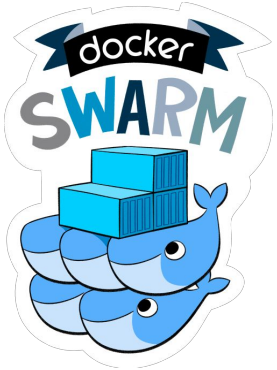
One Server



Multiple Servers



Container Cluster Solution



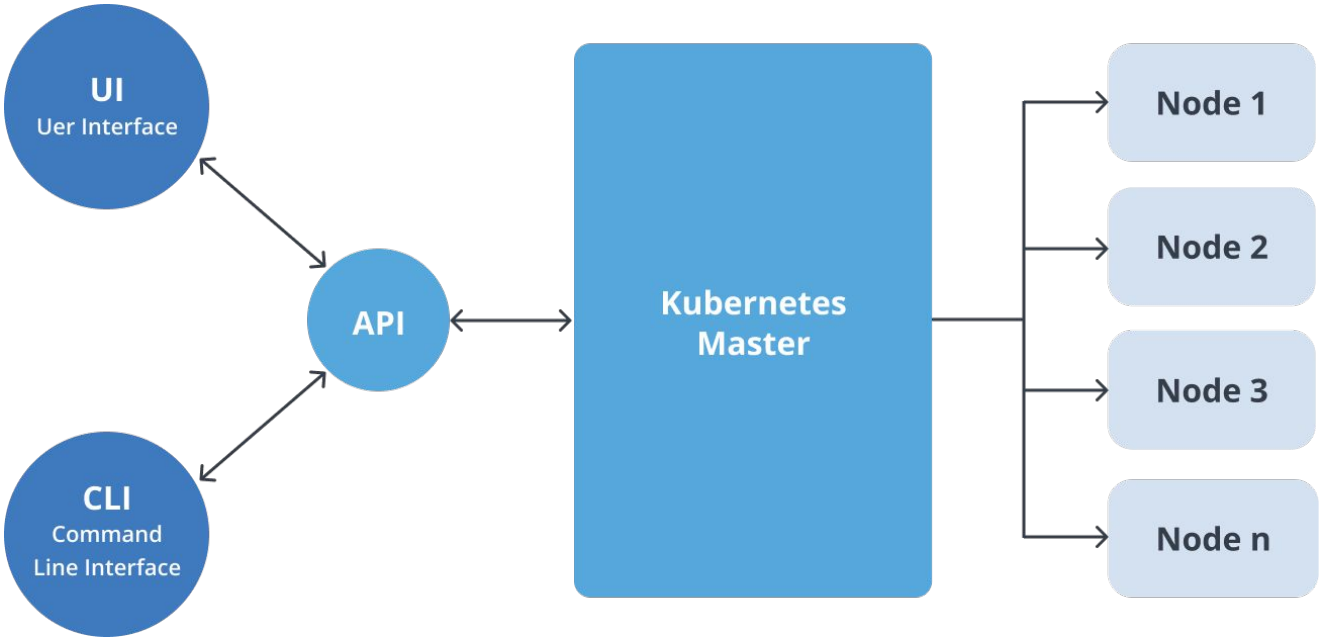
What is Kubernetes?

- **Kubernetes**, in Greek, means the Helmsman, or pilot of the ship, pilot of a ship of containers
- Kubernetes is a software written in **Go** for **automating deployment**, **scaling**, and **management** of containerized applications
- Focus on manage **applications**, not machines
- Open source, open API container **orchestrator**
- Supports **multiple cloud** and **bare-metal** environments
- Inspired and informed by 15 years of Google's experiences and internal systems

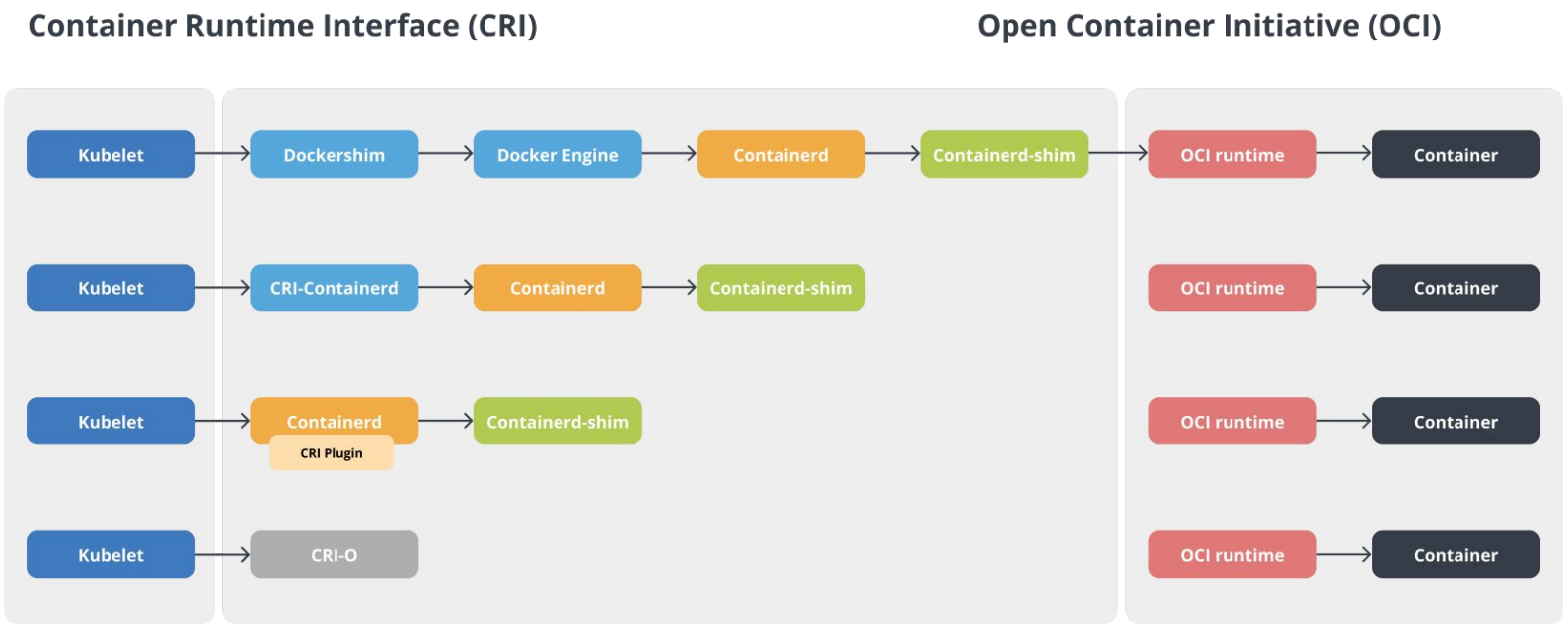
Kubernetes Key Features

- Automatic bin packing
- Self-healing
- Horizontal manual/auto-scaling
- Service discovery & load balancing
- Automated rollouts and rollbacks
- Secret and configuration management
- Storage orchestration
- Batch execution

Kubernetes Architecture



Containerd in Kubernetes



Distribution of Kubernetes

Installation Tools

K3D

KUBESPRAY

kops

kubeadm

Open Source

kind

minikube

K3S

MicroK8s

Commercial

OPENSHIFT

VMware Tanzu

RANCHER

Public Cloud

Azure Container Service

HUAWEI

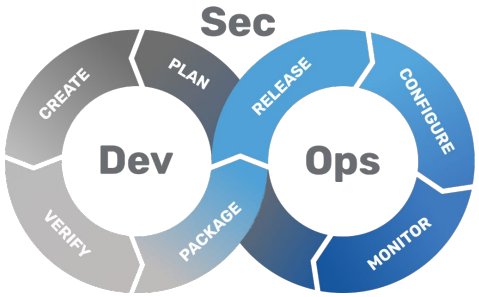
Amazon EKS

Tencent Cloud

IBM Cloud Container Service

Alibaba Cloud

Kubernetes
Pod and Deployment



Transformation

kubernetes

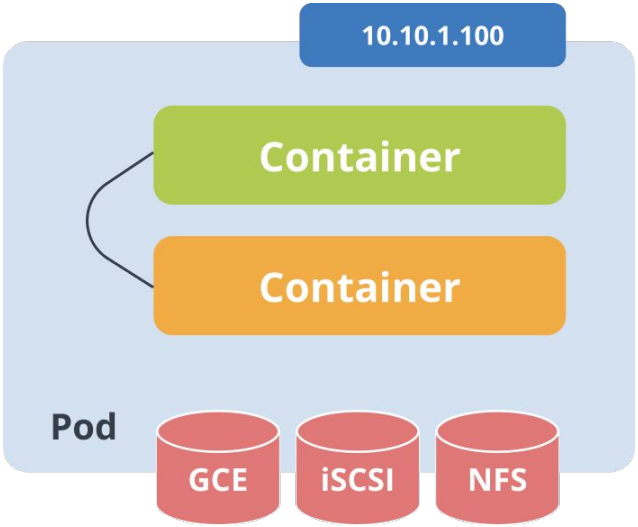
Opsta Education

Skooldio

Pods

Logical Application

- One or more containers and volumes
- Shared namespaces
- One IP per pod



Deployments

Drive current state towards desired state

app: hello
replicas: 1

Deployments

Drive current state towards desired state

app: hello
replicas: 3

Deployments

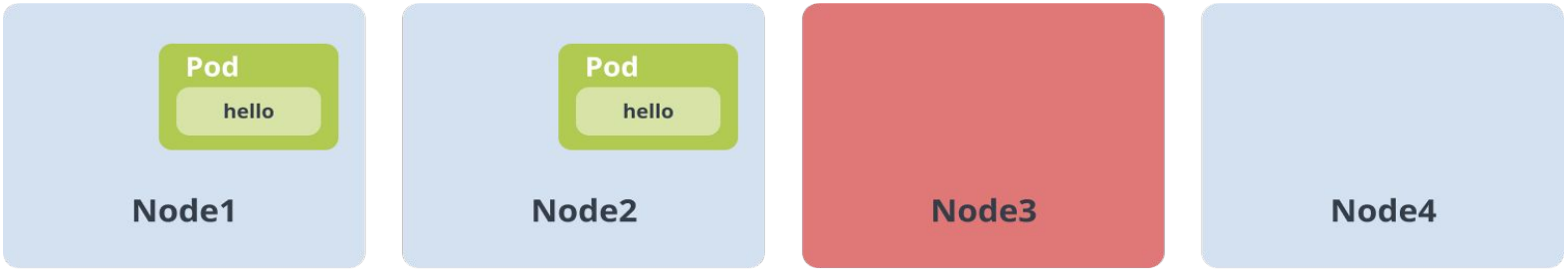
Drive current state towards desired state

app: hello
replicas: 3

Deployments

Drive current state towards desired state

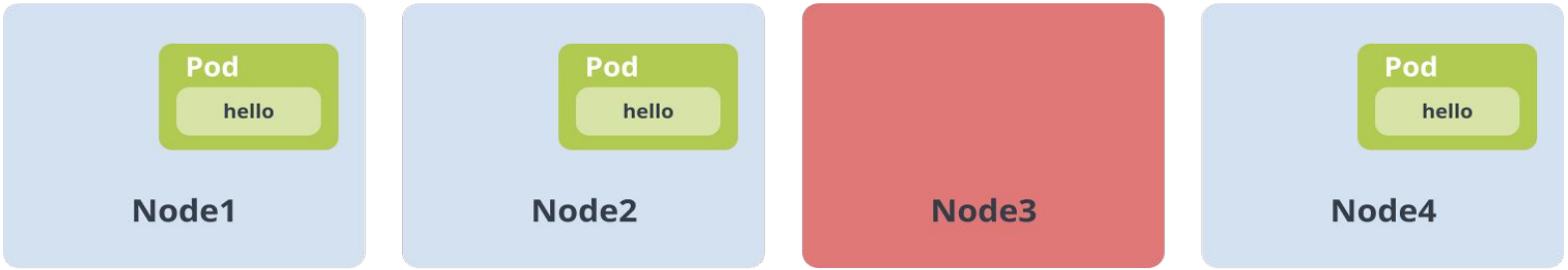
app: hello
replicas: 3



Deployments

Drive current state towards desired state

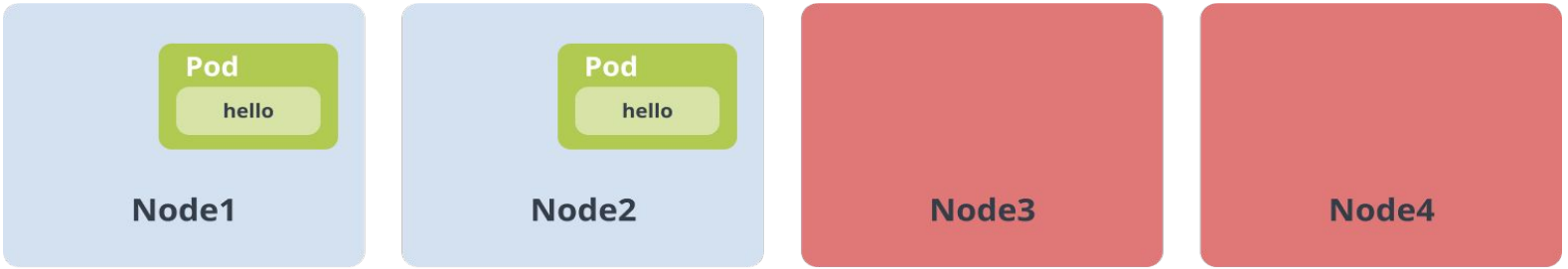
app: hello
replicas: 3



Deployments

Drive current state towards desired state

app: hello
replicas: 3

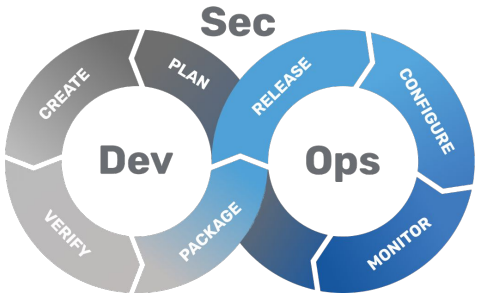


Deployments

Drive current state towards desired state

app: hello
replicas: 3

Kubernetes Service



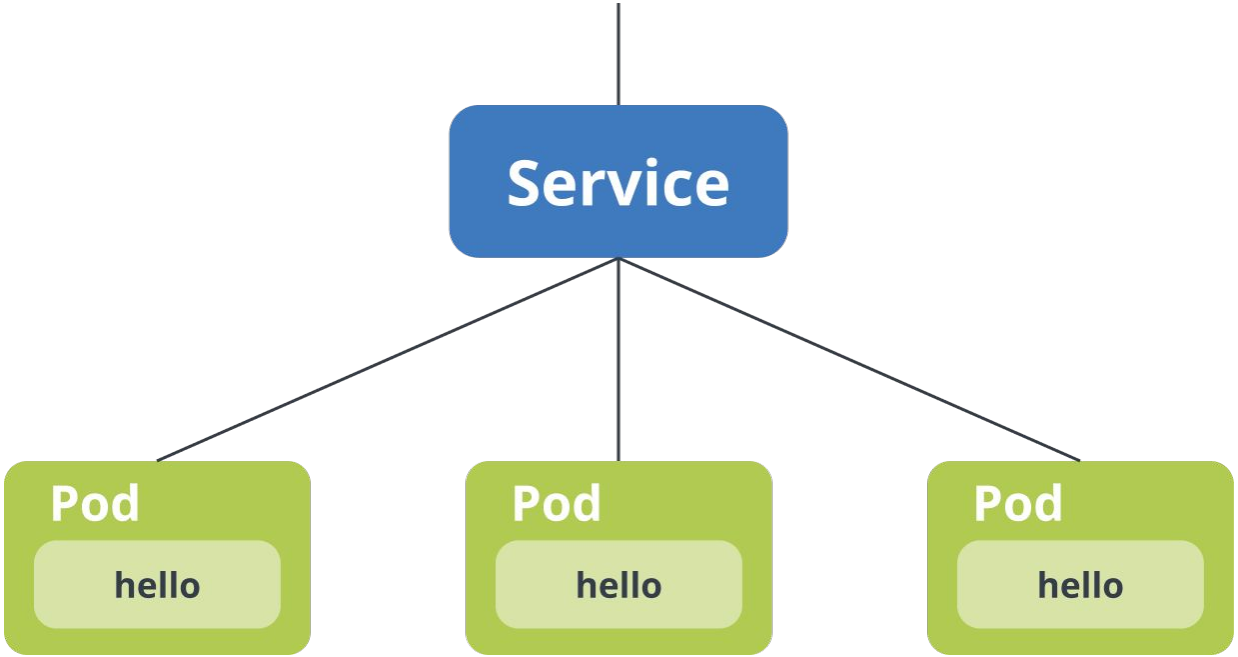
Transformation

kubernetes

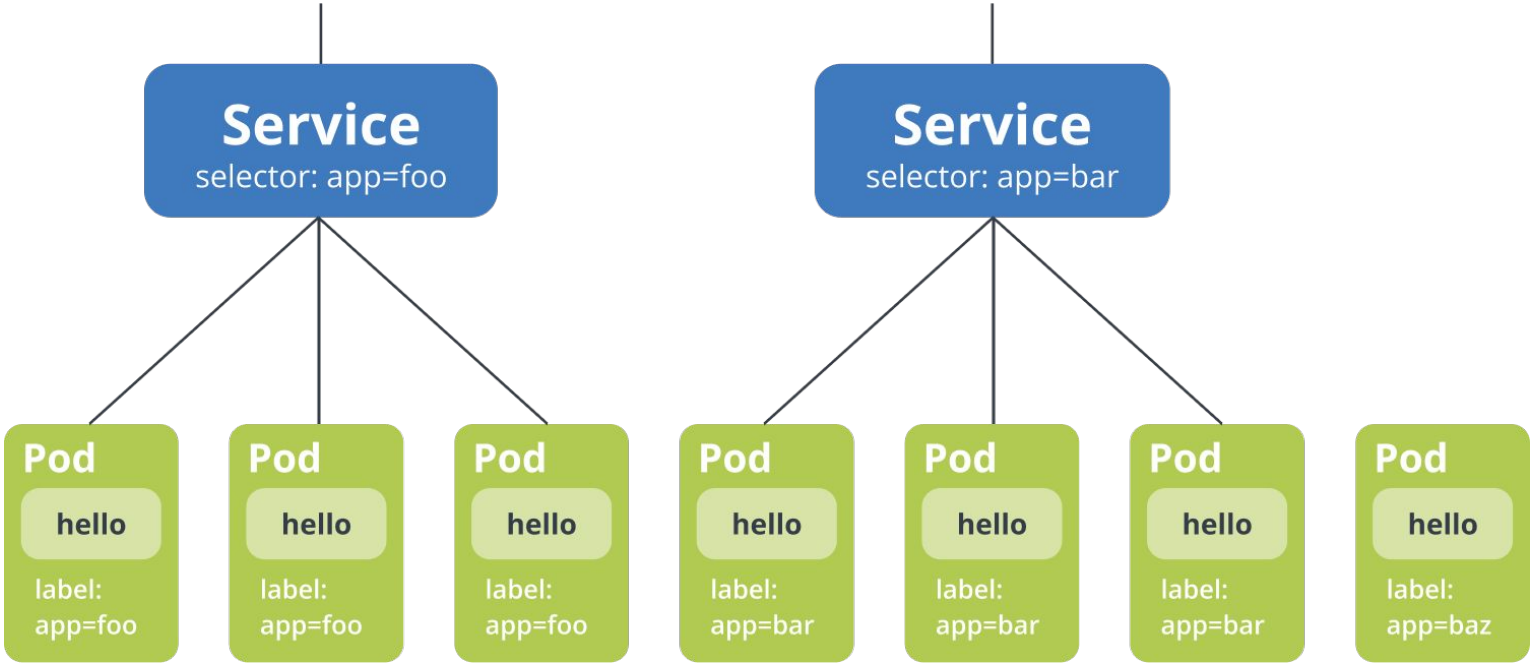
Opsta Education

Skooldio

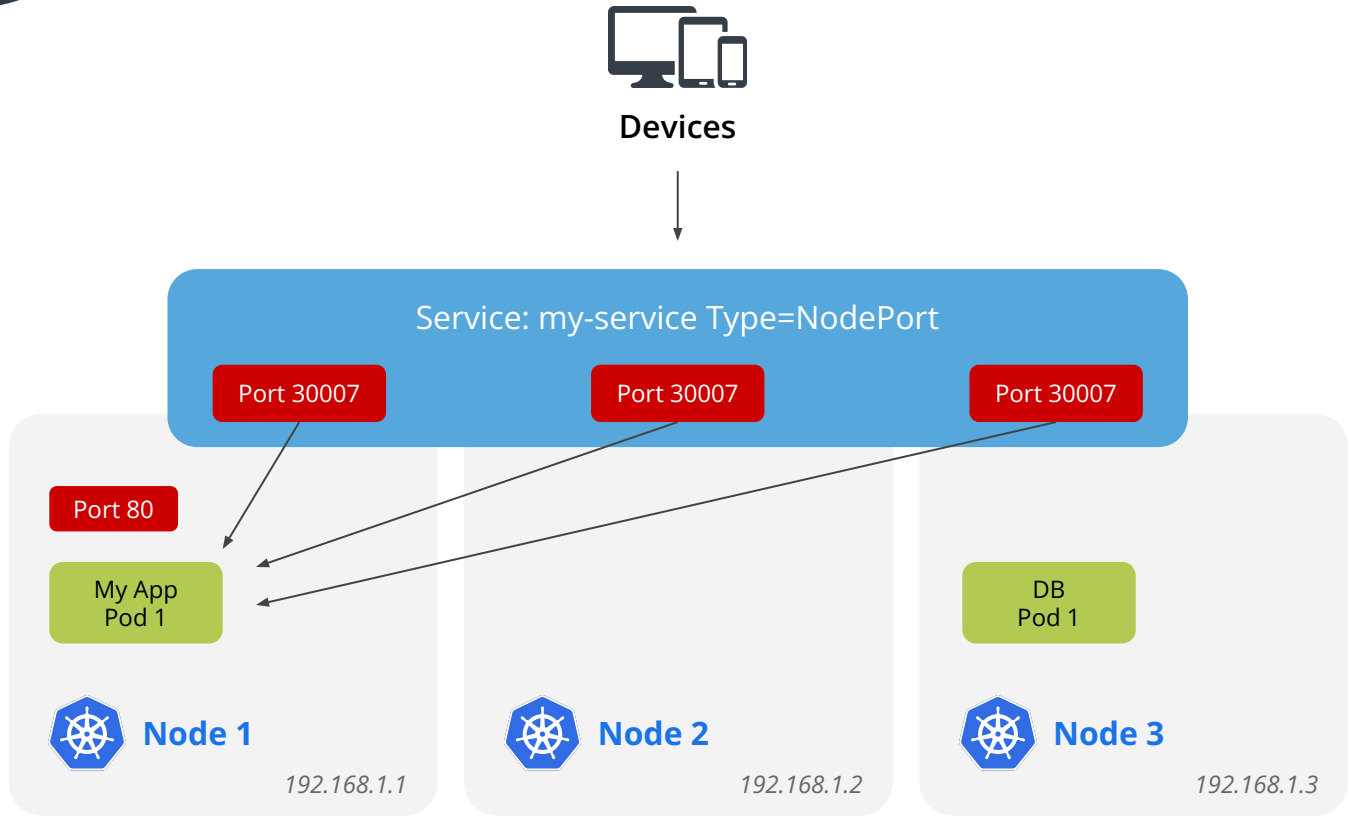
Services



Labels and Selectors

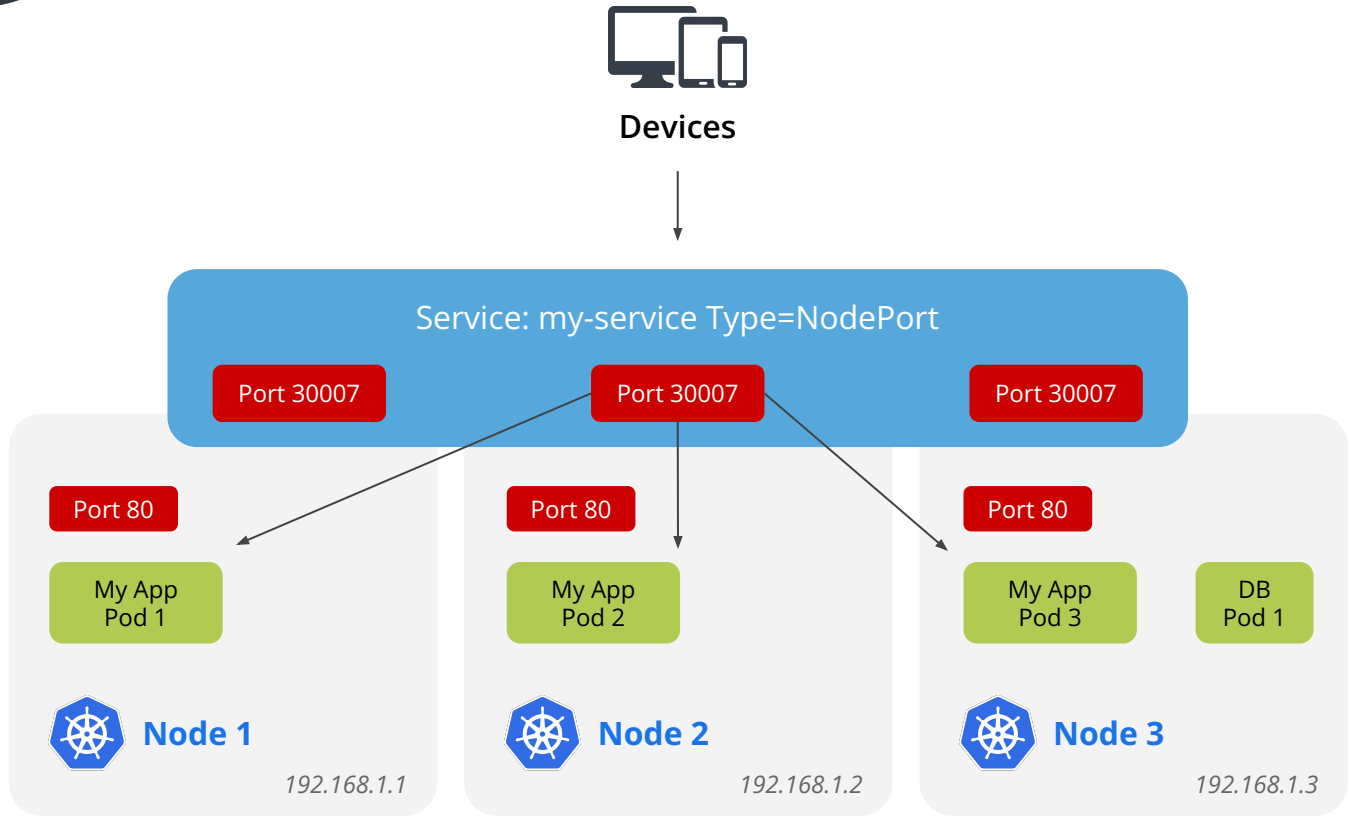


Service Type: Node Port



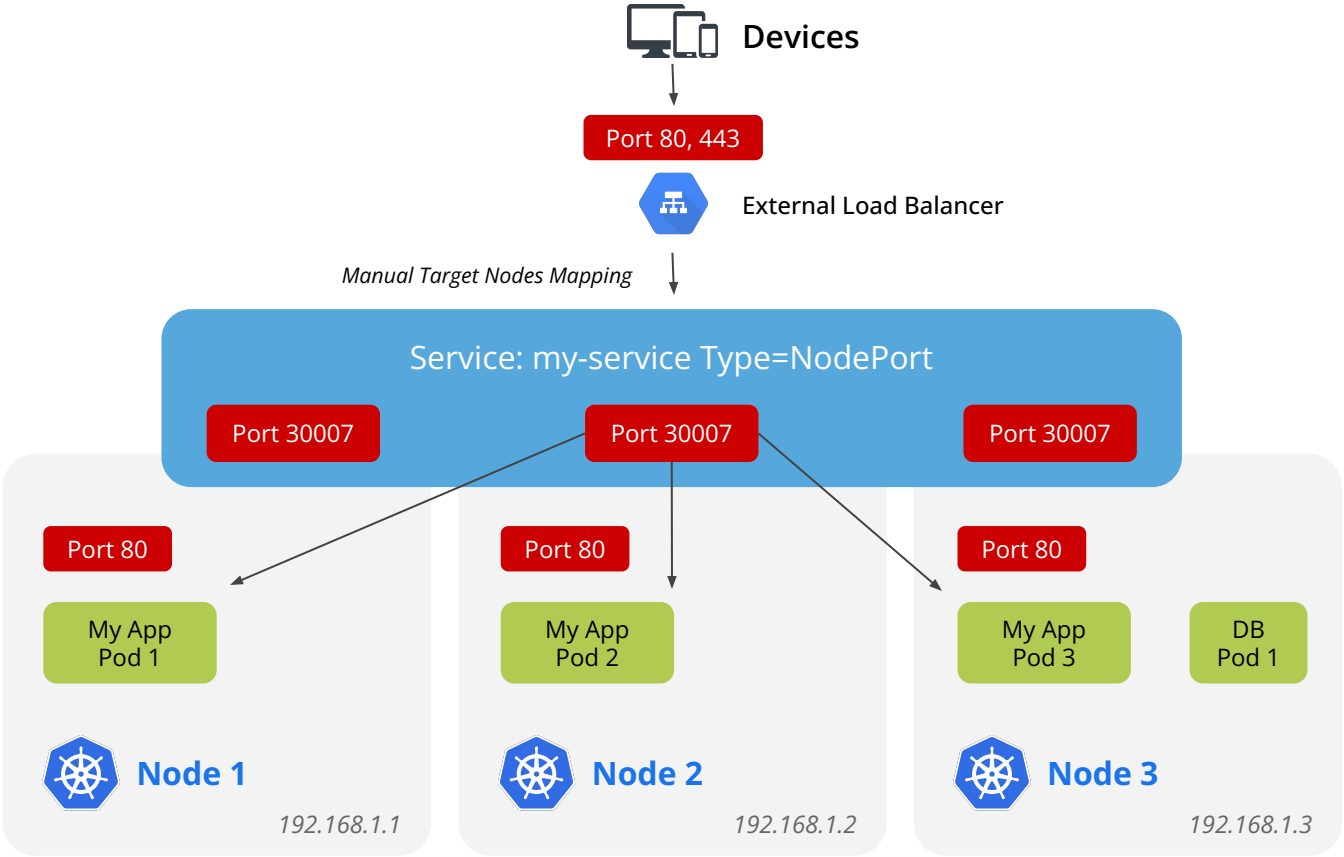

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
      # By default and for convenience, the `targetPort` is set to the same value as the `port` field.
    - port: 80
      targetPort: 80
      # Optional field
      # By default and for convenience, the Kubernetes control plane will allocate a port from a range (default: 30000-32767)
      nodePort: 30007
```

Service Type: Node Port




```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
      # By default and for convenience, the `targetPort` is set to the same value as the `port` field.
    - port: 80
      targetPort: 80
      # Optional field
      # By default and for convenience, the Kubernetes control plane will allocate a port from a range (default: 30000-32767)
      nodePort: 30007
```

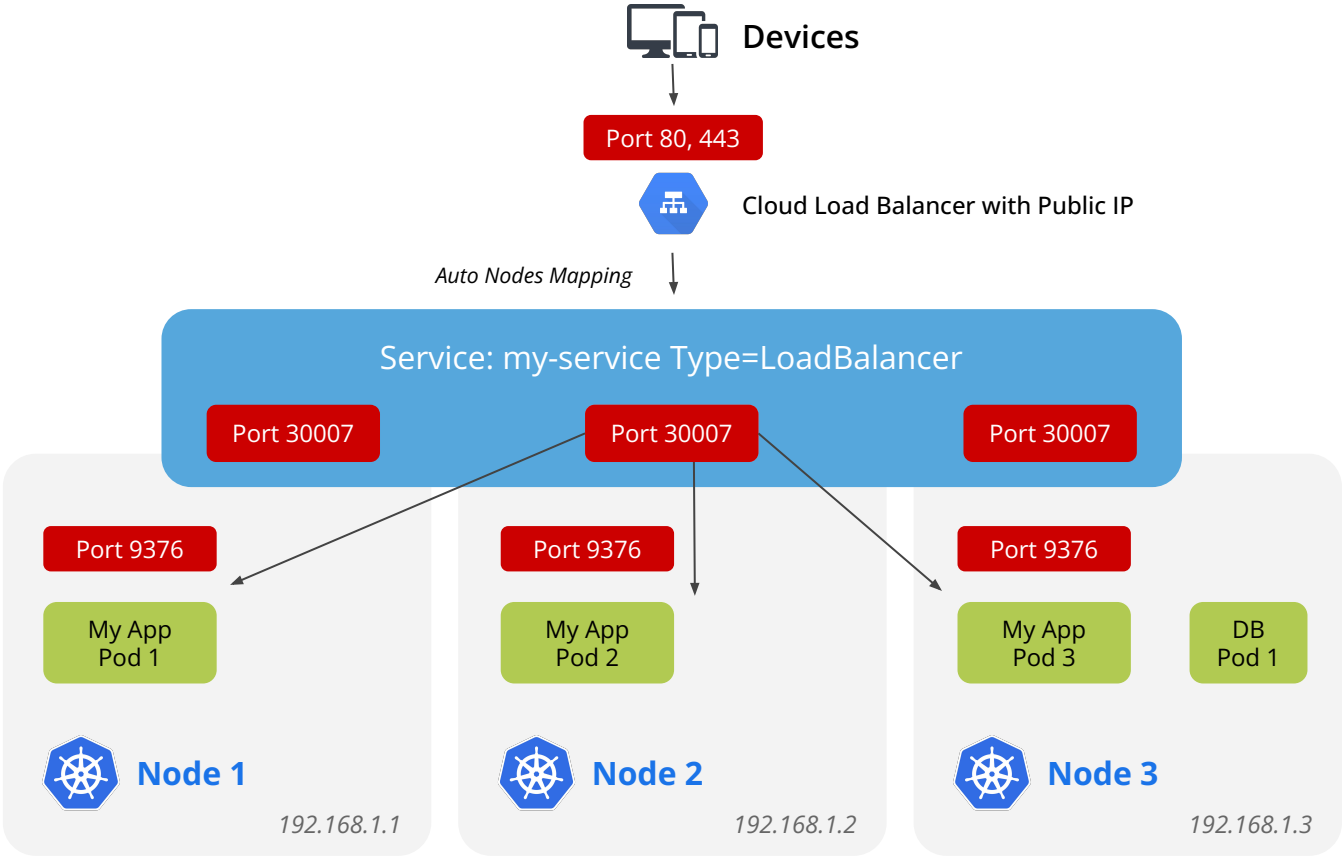
Node Port with External Load Balancer




```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
      # By default and for convenience, the `targetPort` is set to the same value as the `port` field.
    - port: 80
      targetPort: 80
      # Optional field
      # By default and for convenience, the Kubernetes control plane will allocate a port from a range (default: 30000-32767)
      nodePort: 30007
```

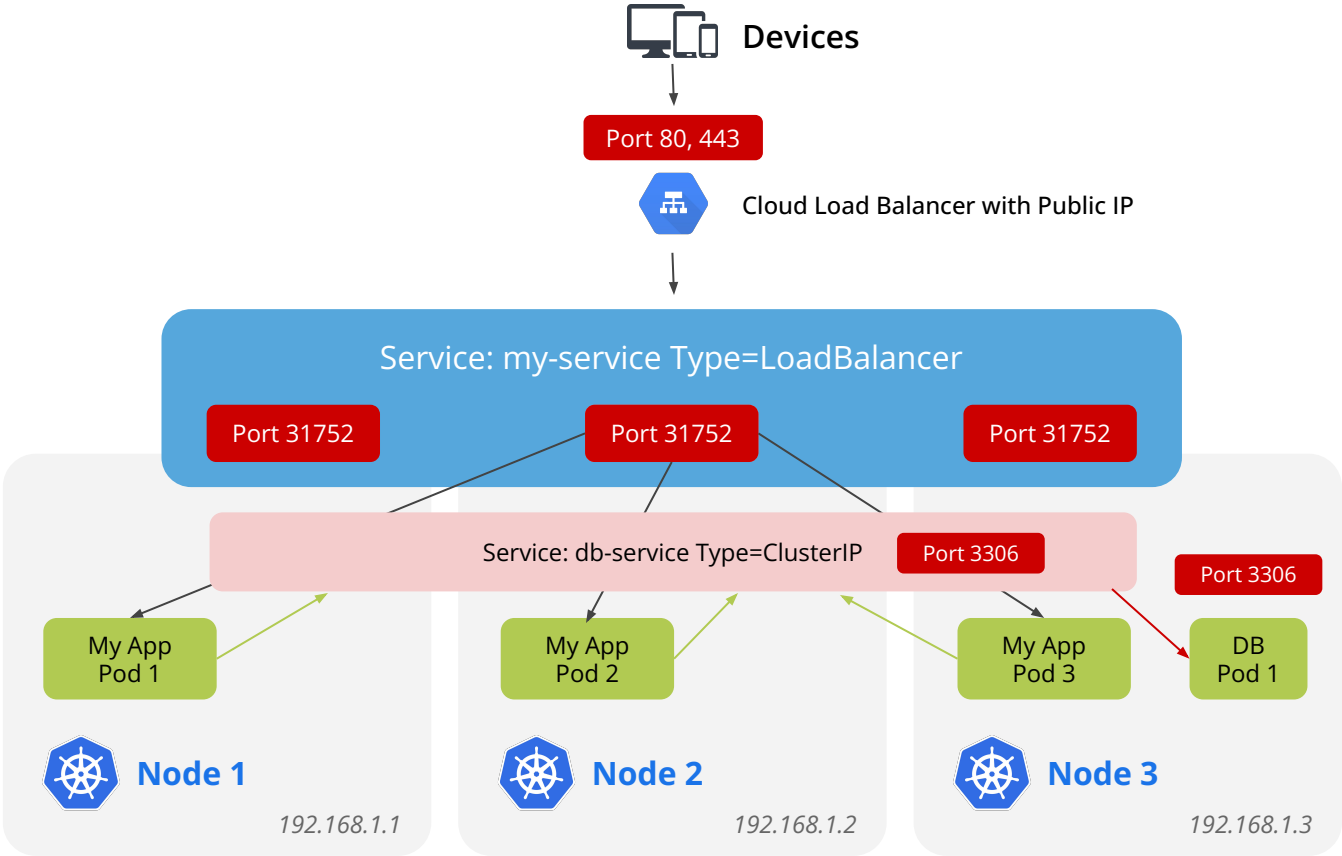
Skooldio Opsta Education

Service Type: LoadBalancer



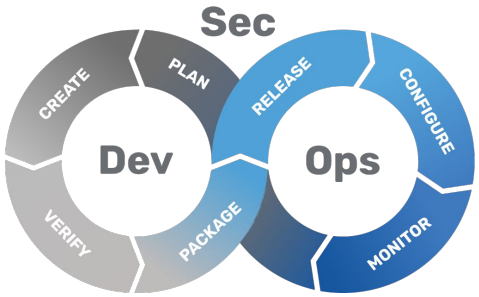

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
  type: LoadBalancer
```

Service Type: ClusterIP




```yaml
apiVersion: v1
kind: Service
metadata:
  name: db-service
spec:
  selector:
    app: DB
  ports:
    - protocol: TCP
      port: 3306
      targetPort: 3306
  type: ClusterIP
```

Kubernetes Ingress



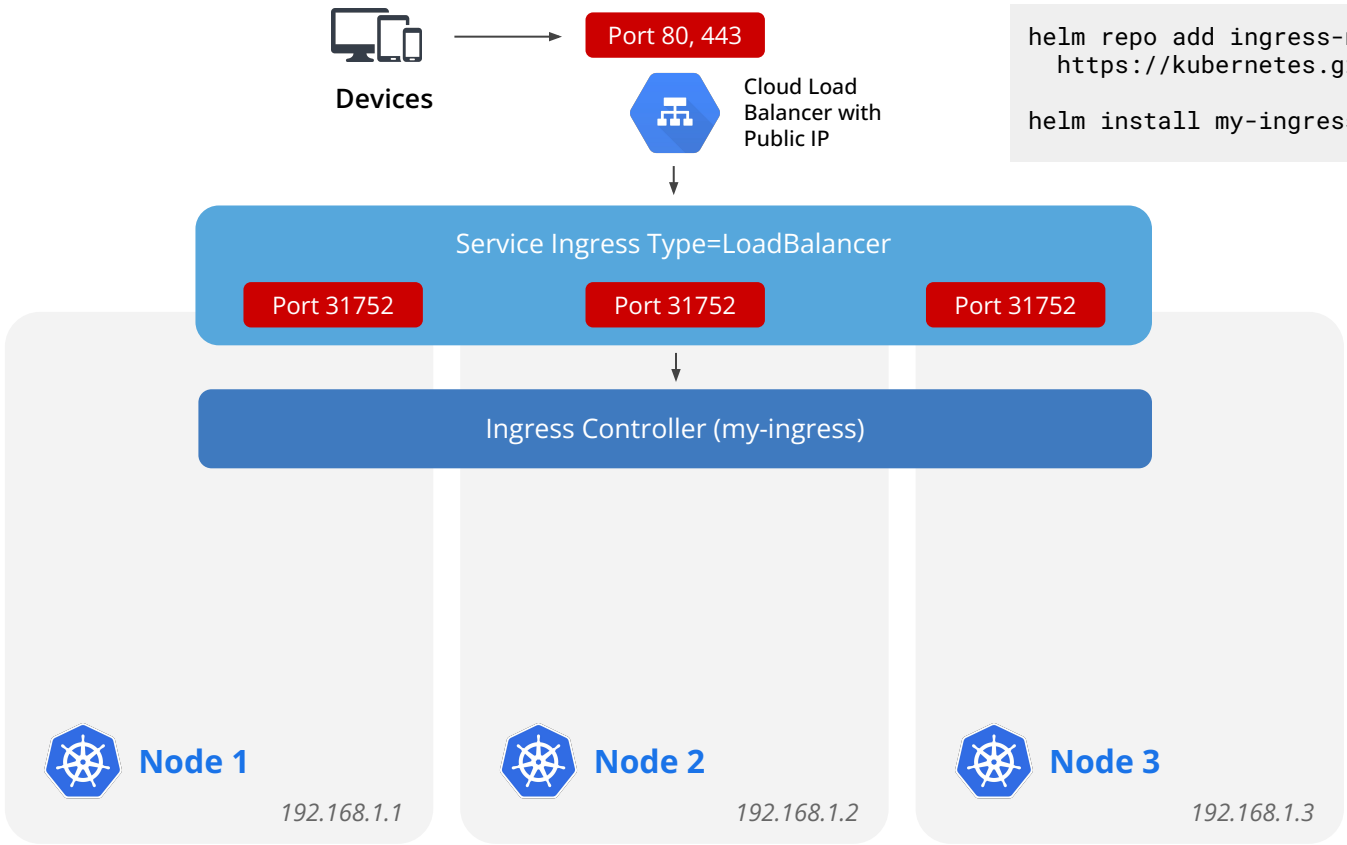
Transformation

kubernetes

Opsta Education

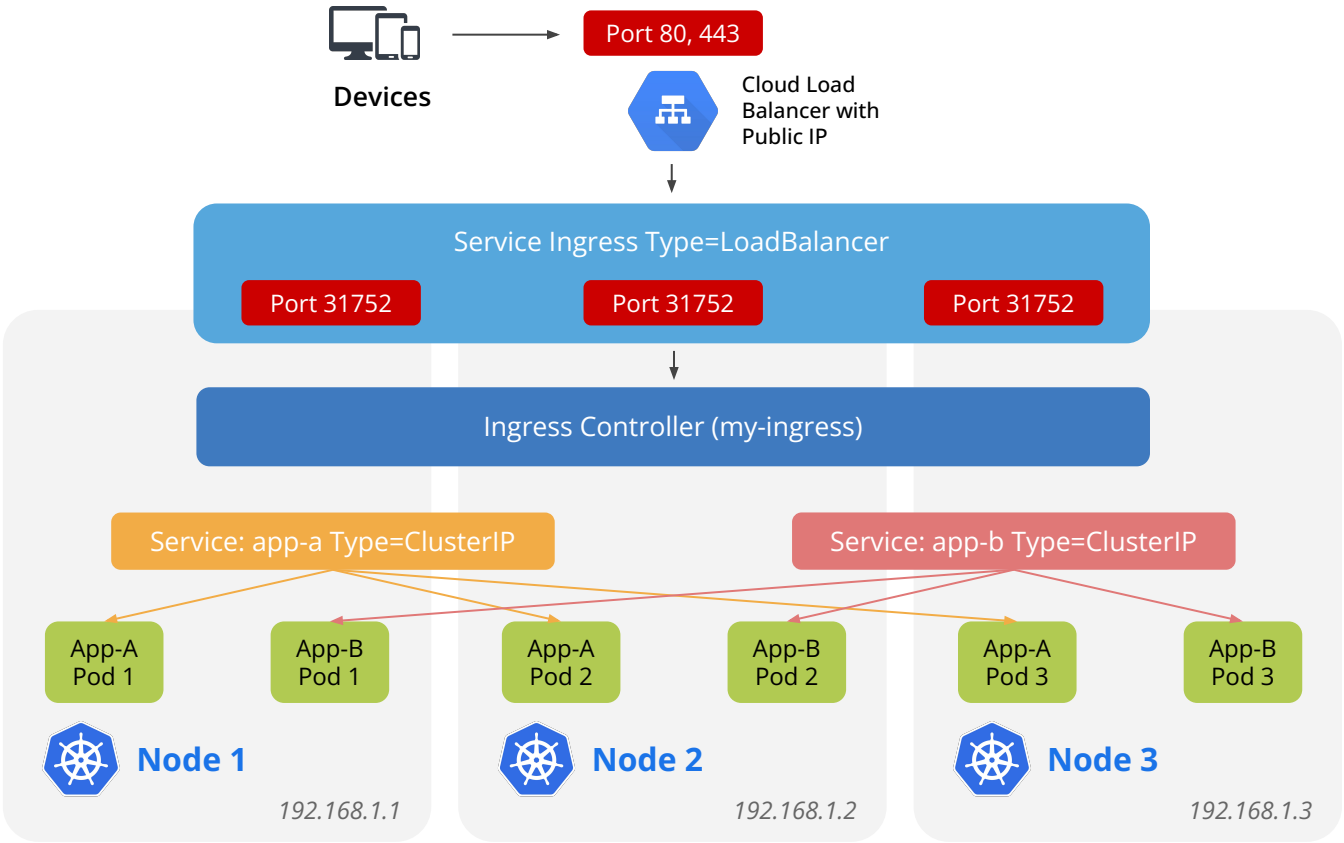
Skooldio

Service Type: Ingress (1)



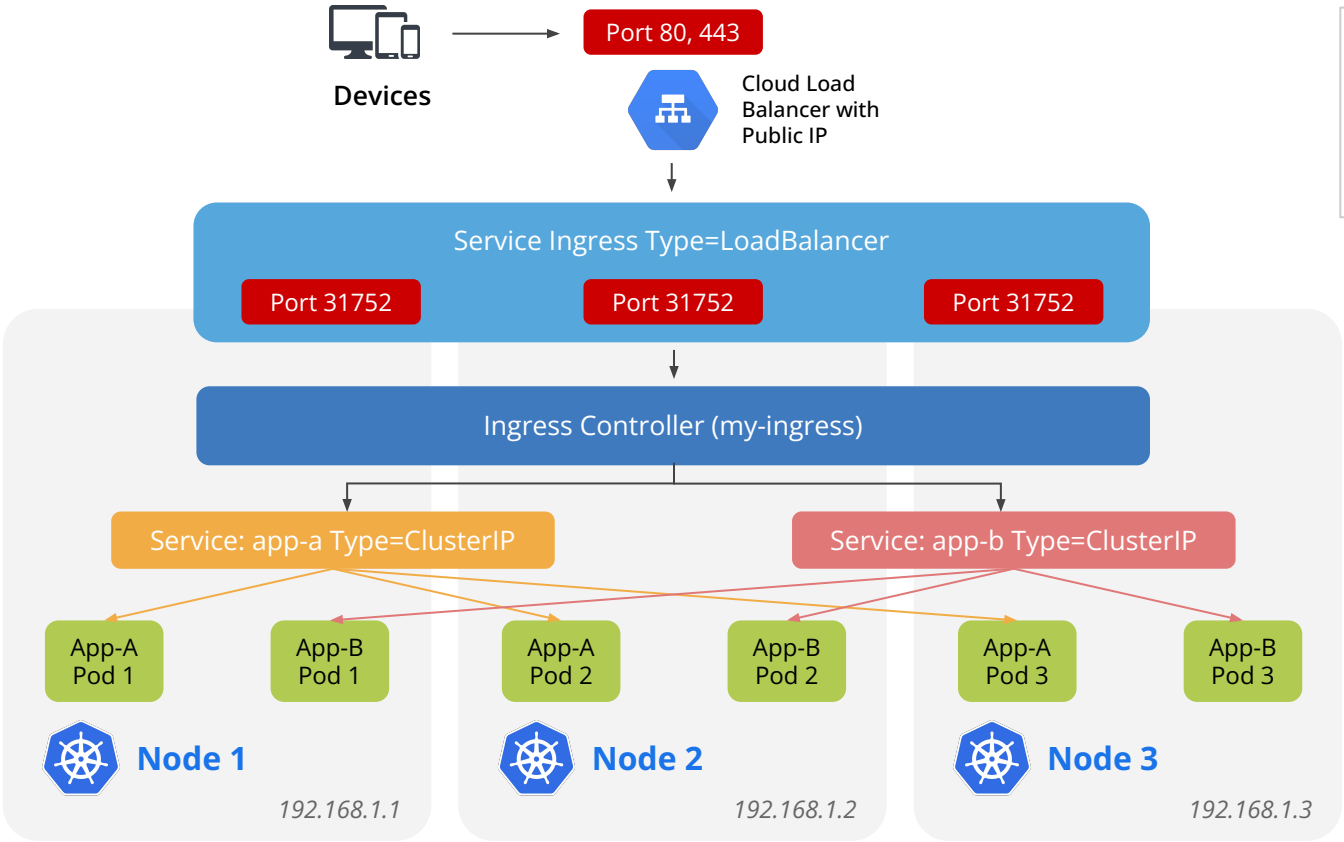

```
helm repo add ingress-nginx \
  https://kubernetes.github.io/ingress-nginx

helm install my-ingress ingress-nginx/ingress-nginx
```

Service Type: Ingress (2)




```yaml
apiVersion: v1
kind: Service
metadata:
  name: app-a
spec:
  selector:
    app: App-A
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
```

Service Type: Ingress (3)

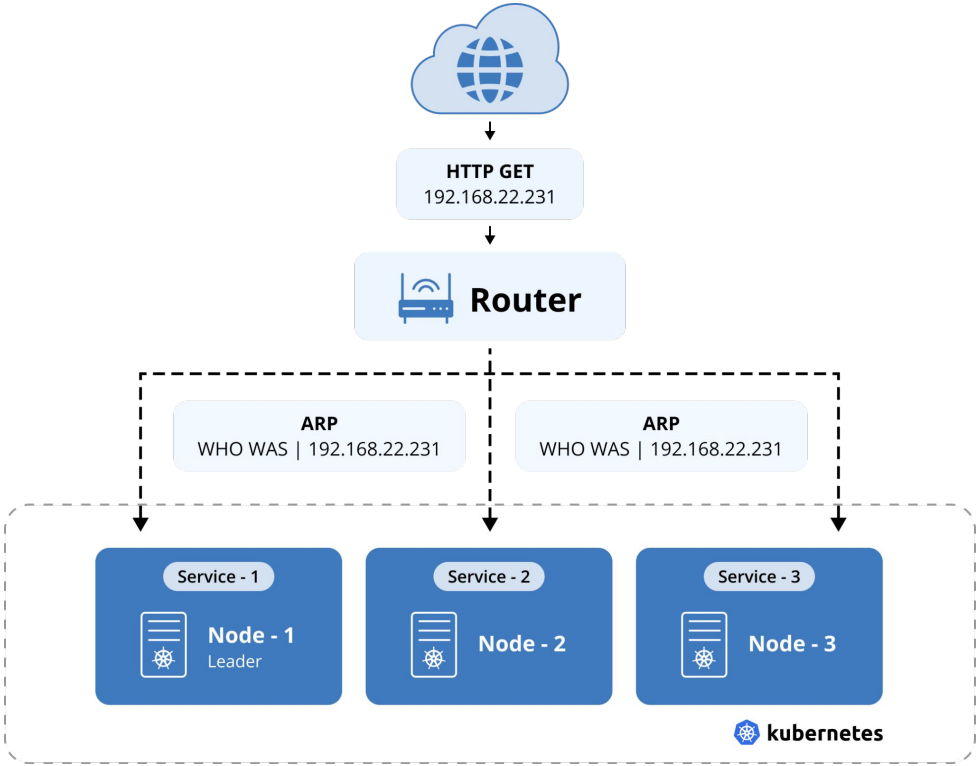


Ingress Resource

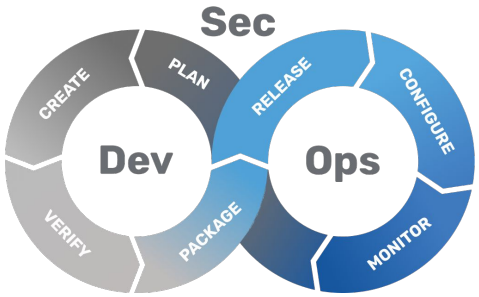
- foo.com - Service A
- bar.com - Service B

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress-foo
spec:
  rules:
  - host: "foo.com"
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: app-a
            port:
              number: 8080
```

MetalLB



Kubernetes Gateway API



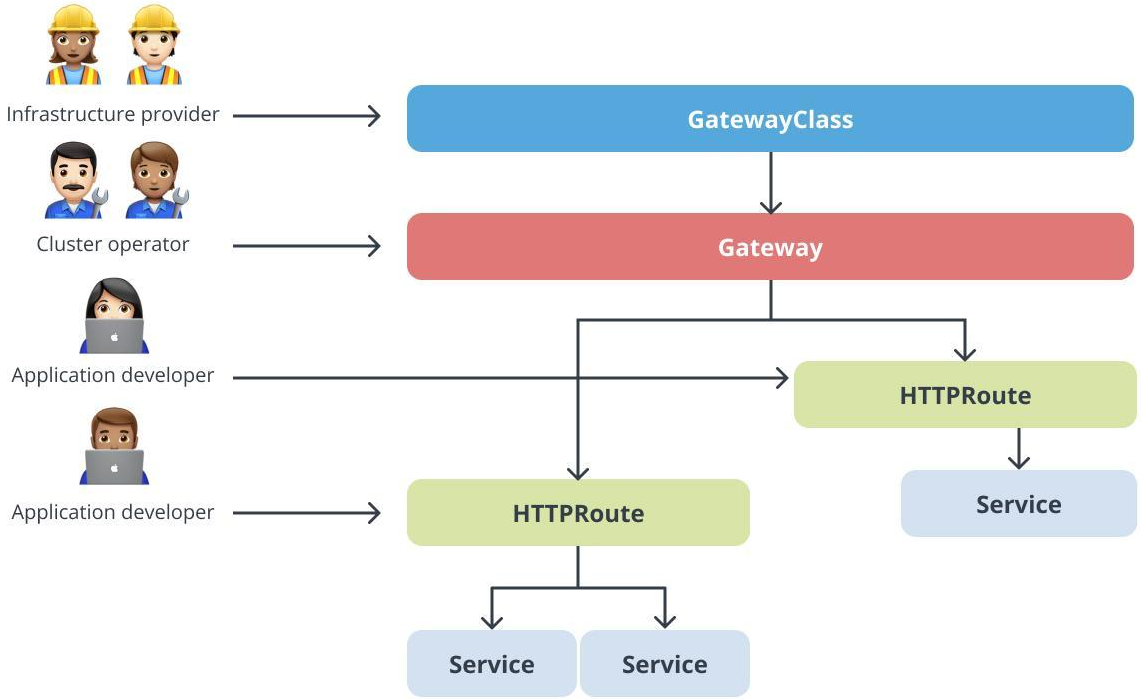
Transformation

gateway api

Opsta Education

Skooldio

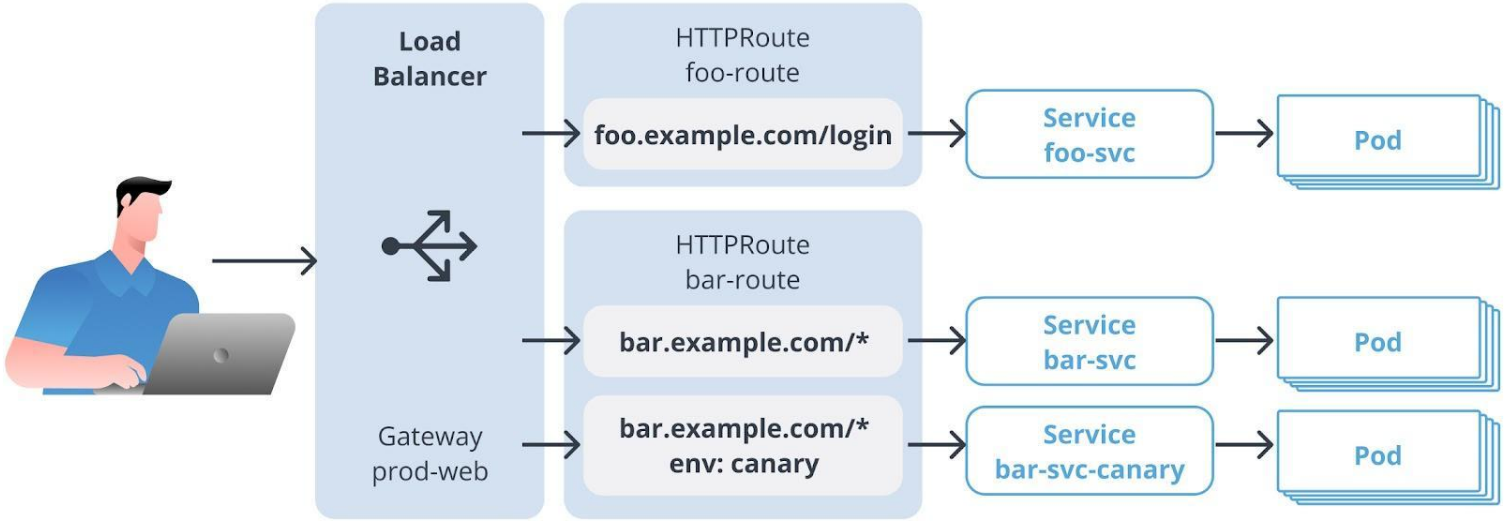
Gateway API



A modern set of APIs for deploying L4 and L7 routing in Kubernetes

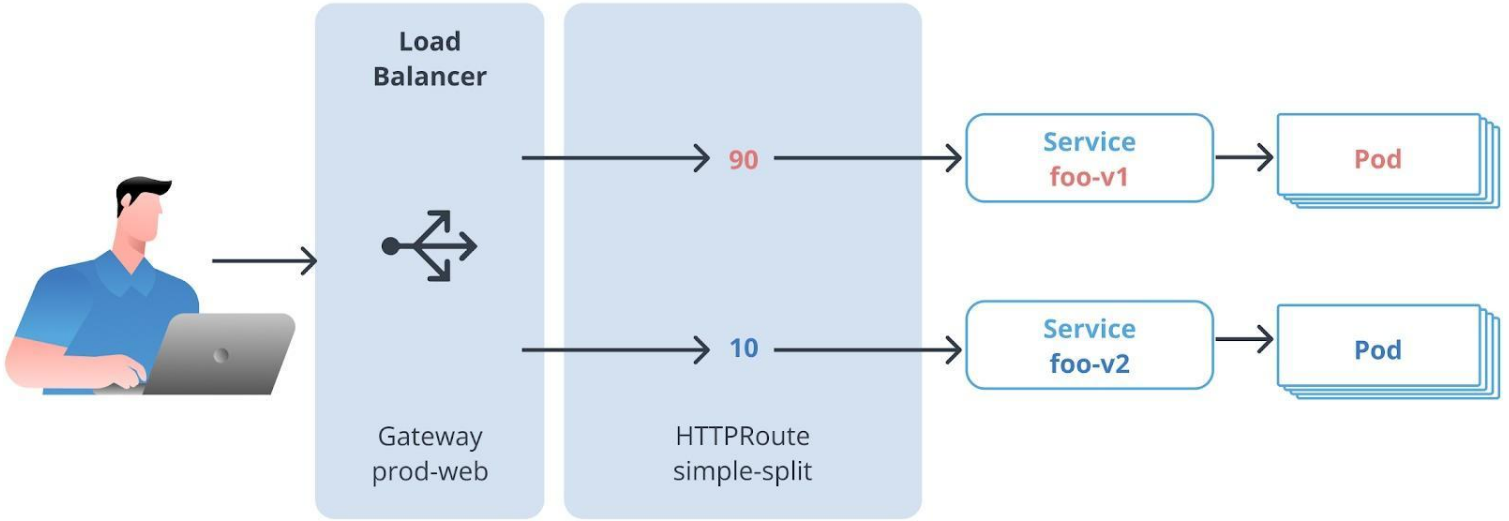
Designed to be generic, expressive, extensible, and role-oriented

HTTP routing with header



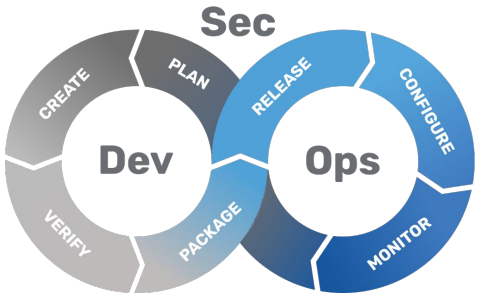
https://gateway-api.sigs.k8s.io/guides/http-routing/

HTTP traffic splitting



https://gateway-api.sigs.k8s.io/guides/traffic-splitting/

Kubernetes Namespace



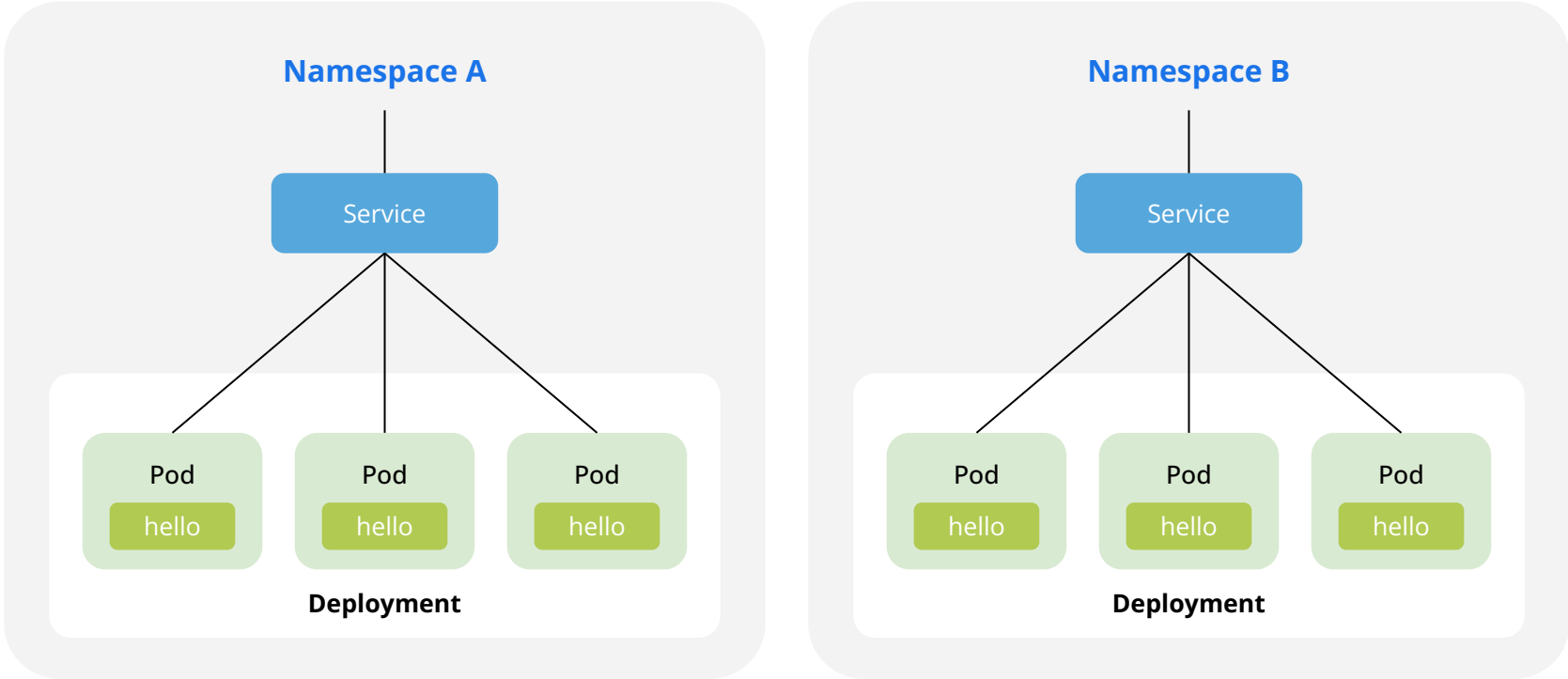
Transformation

kubernetes

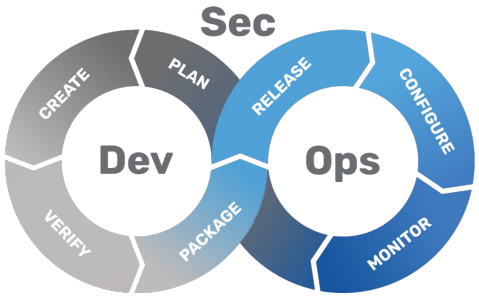
Opsta Education

Skooldio

Namespace



Kubernetes Rolling Update



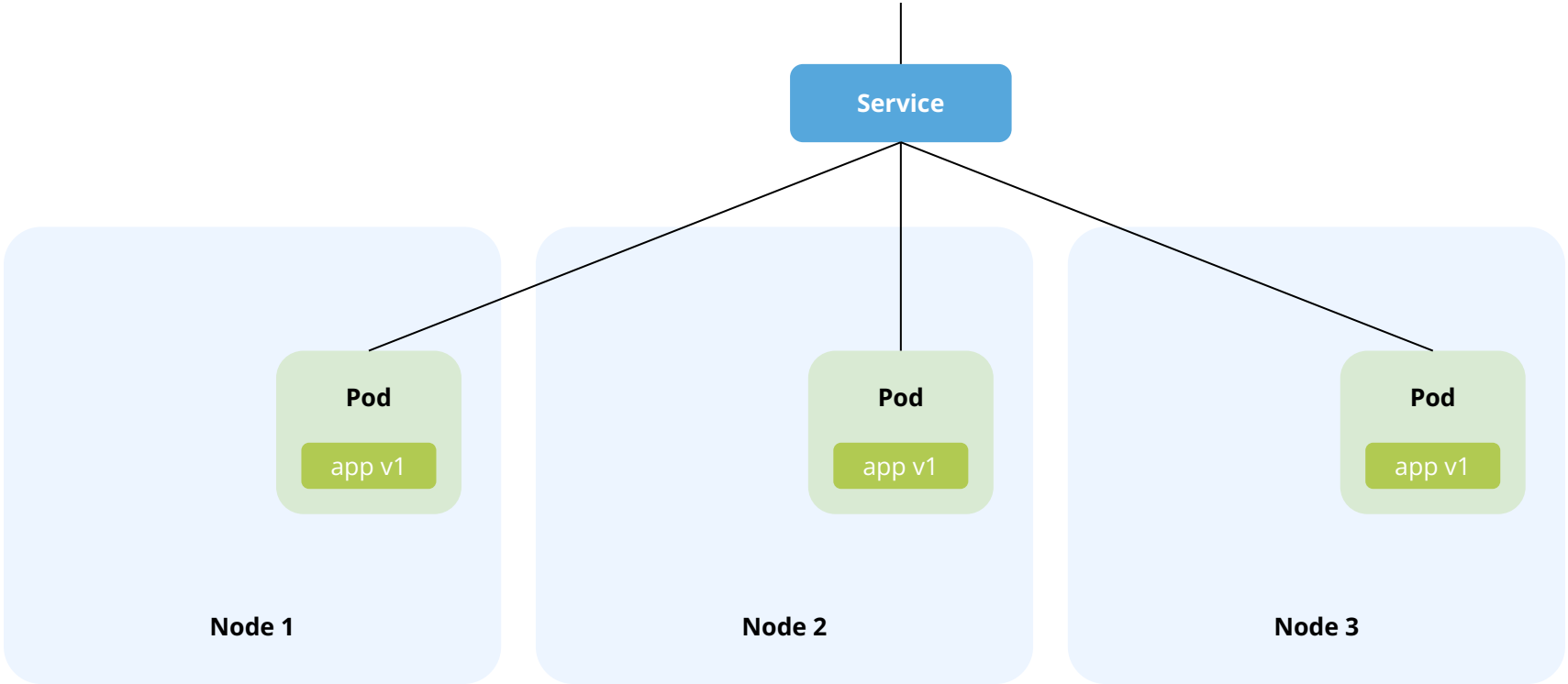
Transformation

kubernetes

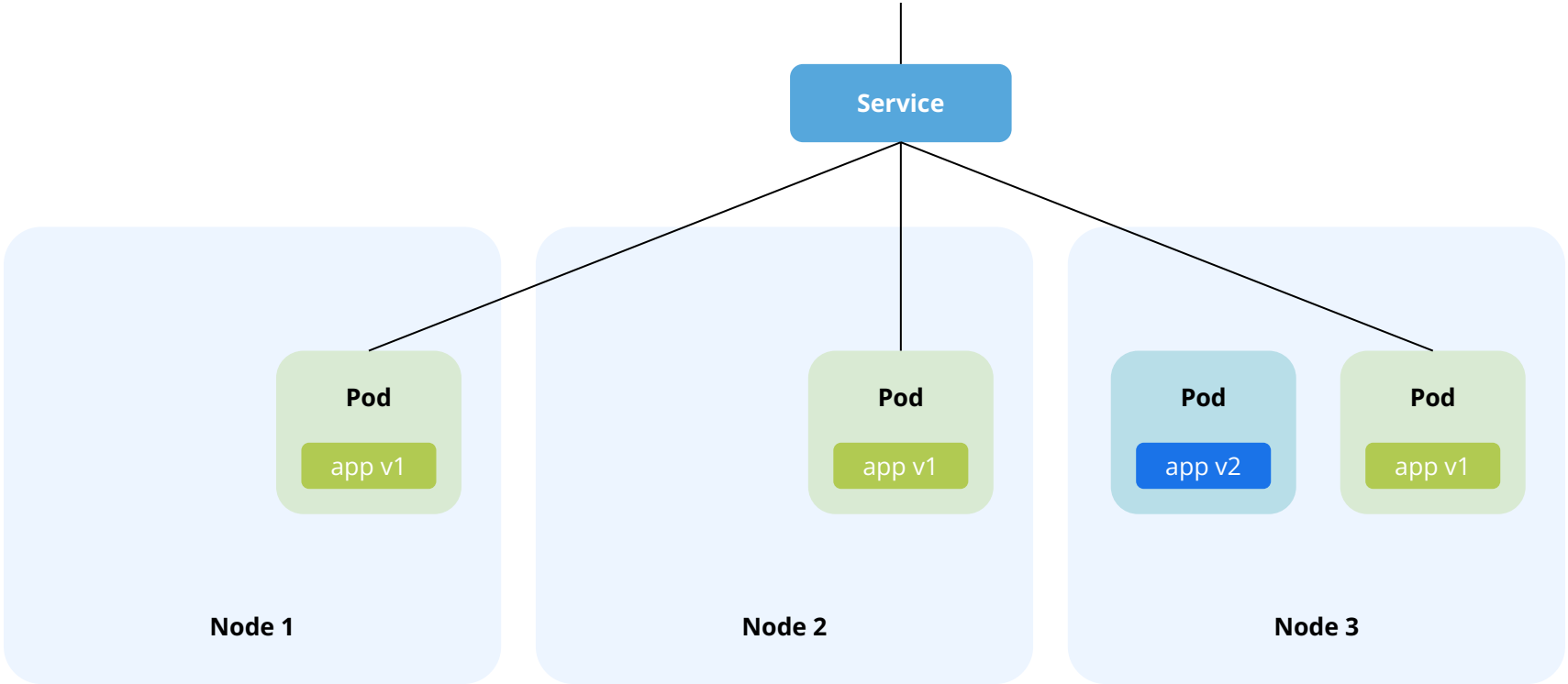
Opsta Education

Skooldio

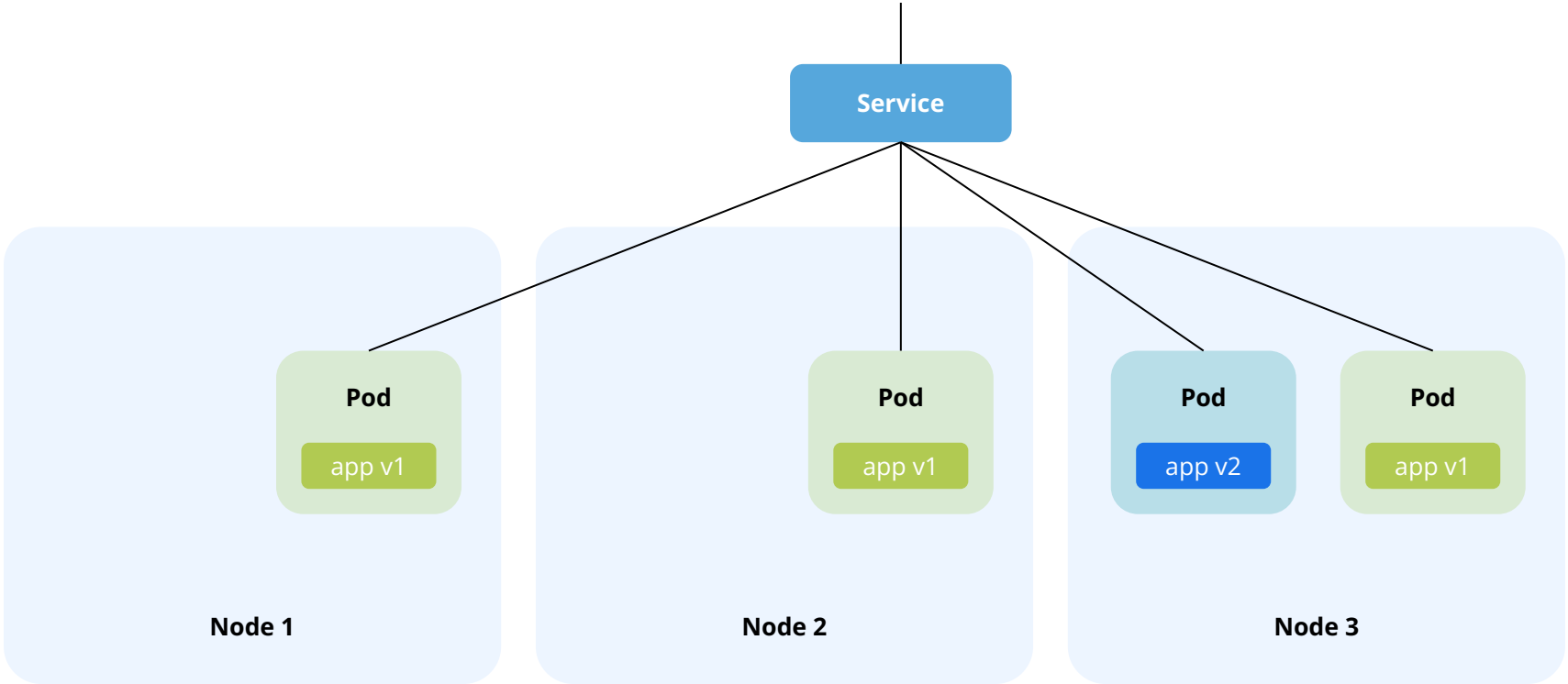
Rolling Update



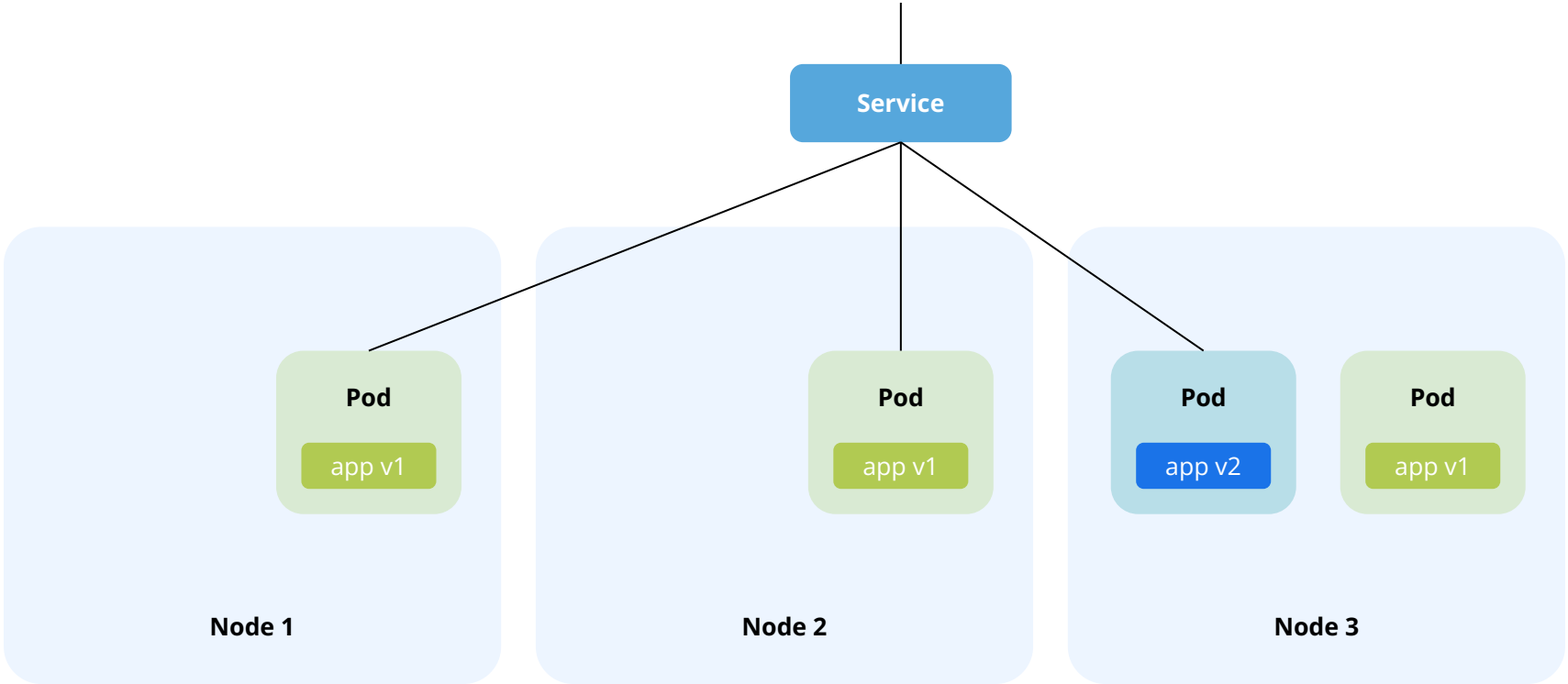
Rolling Update



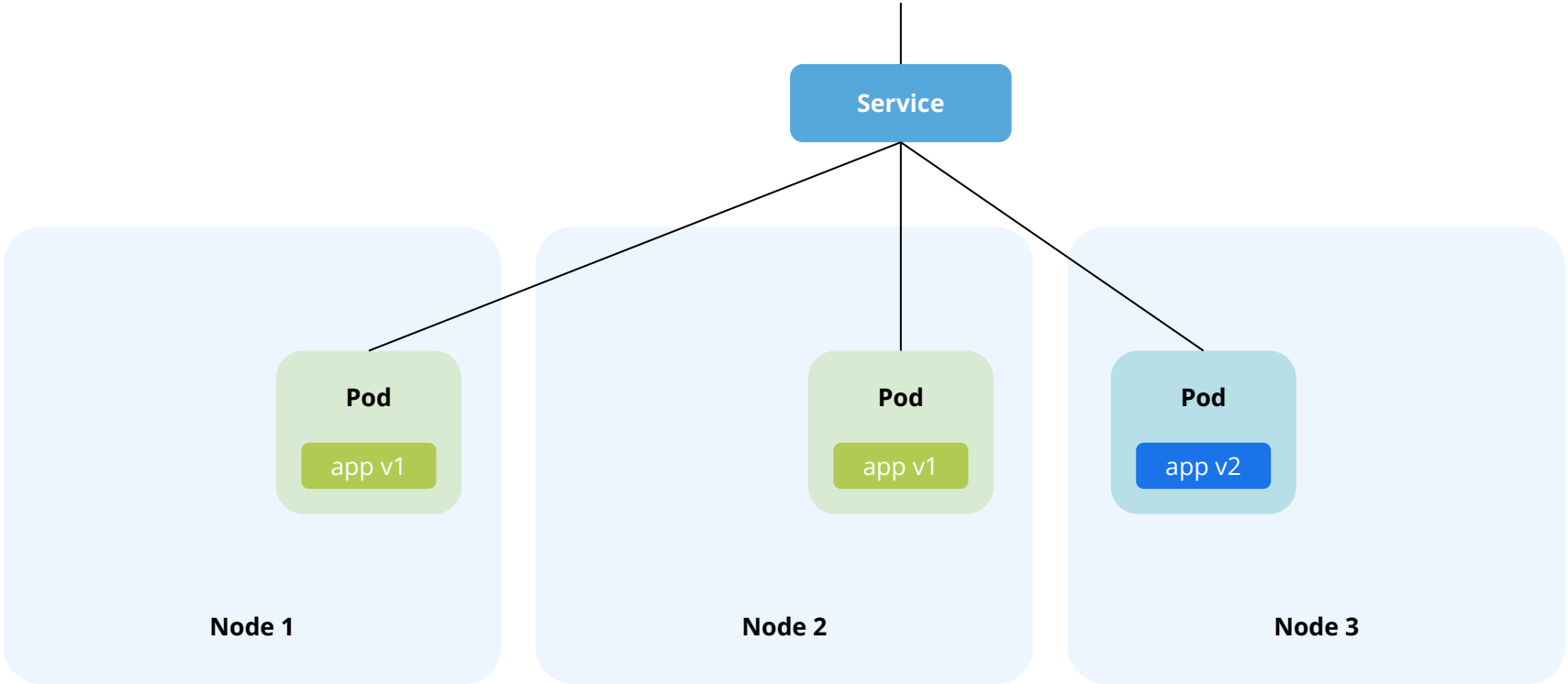
Rolling Update



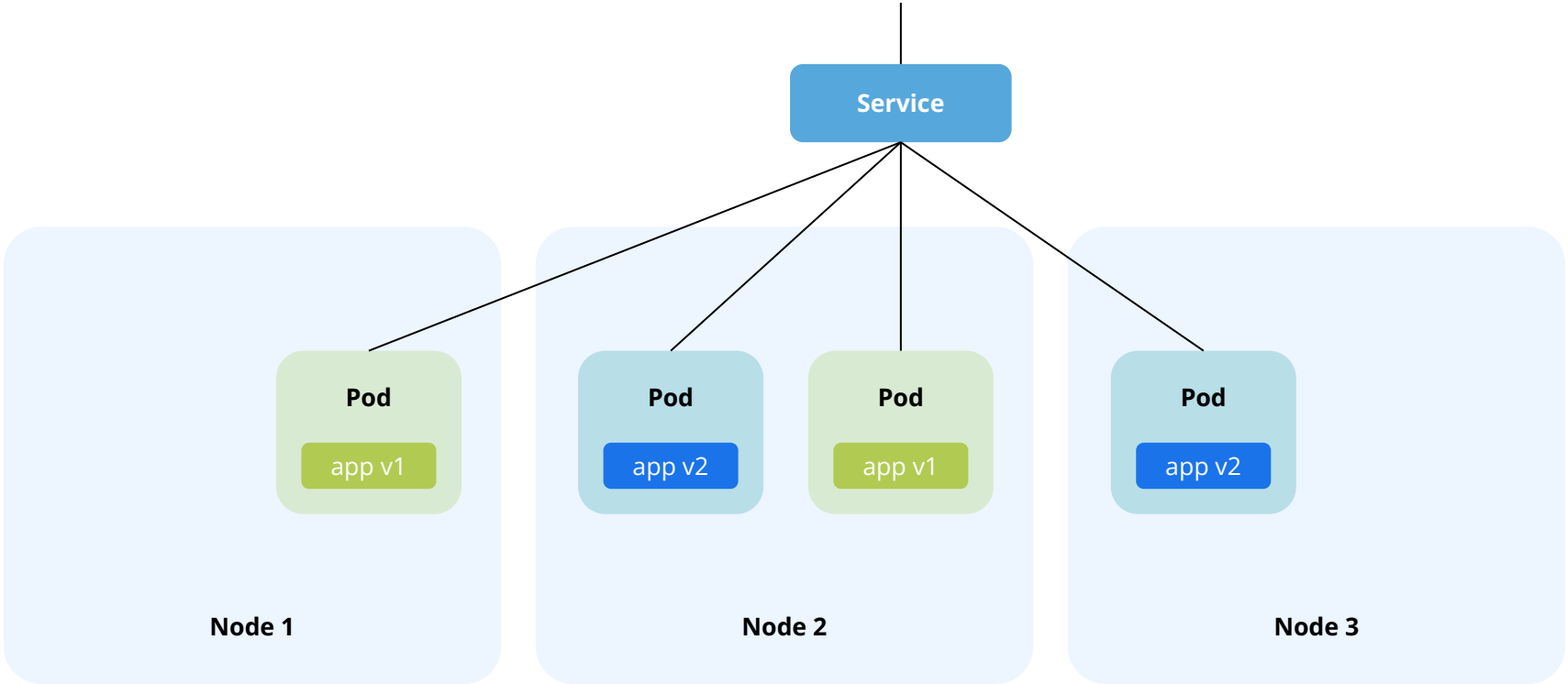
Rolling Update



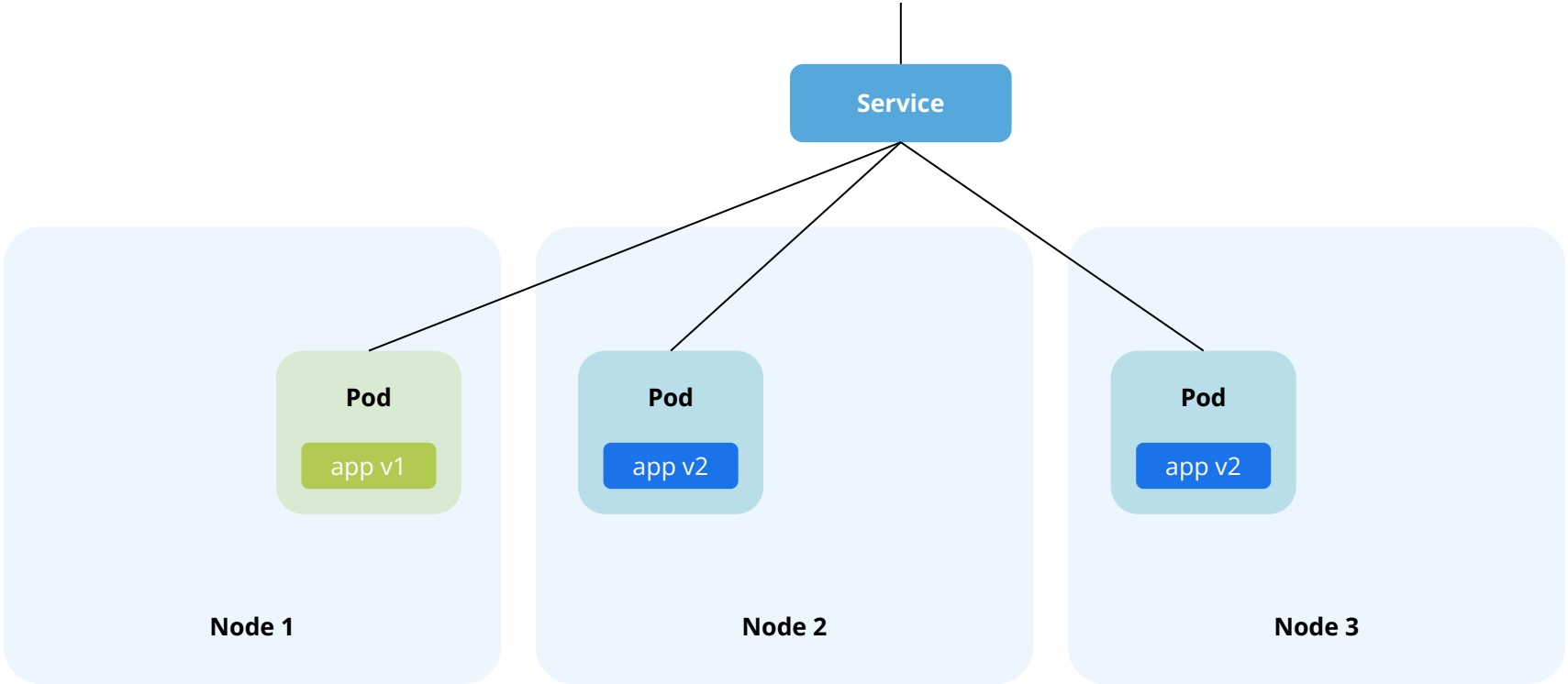
Rolling Update



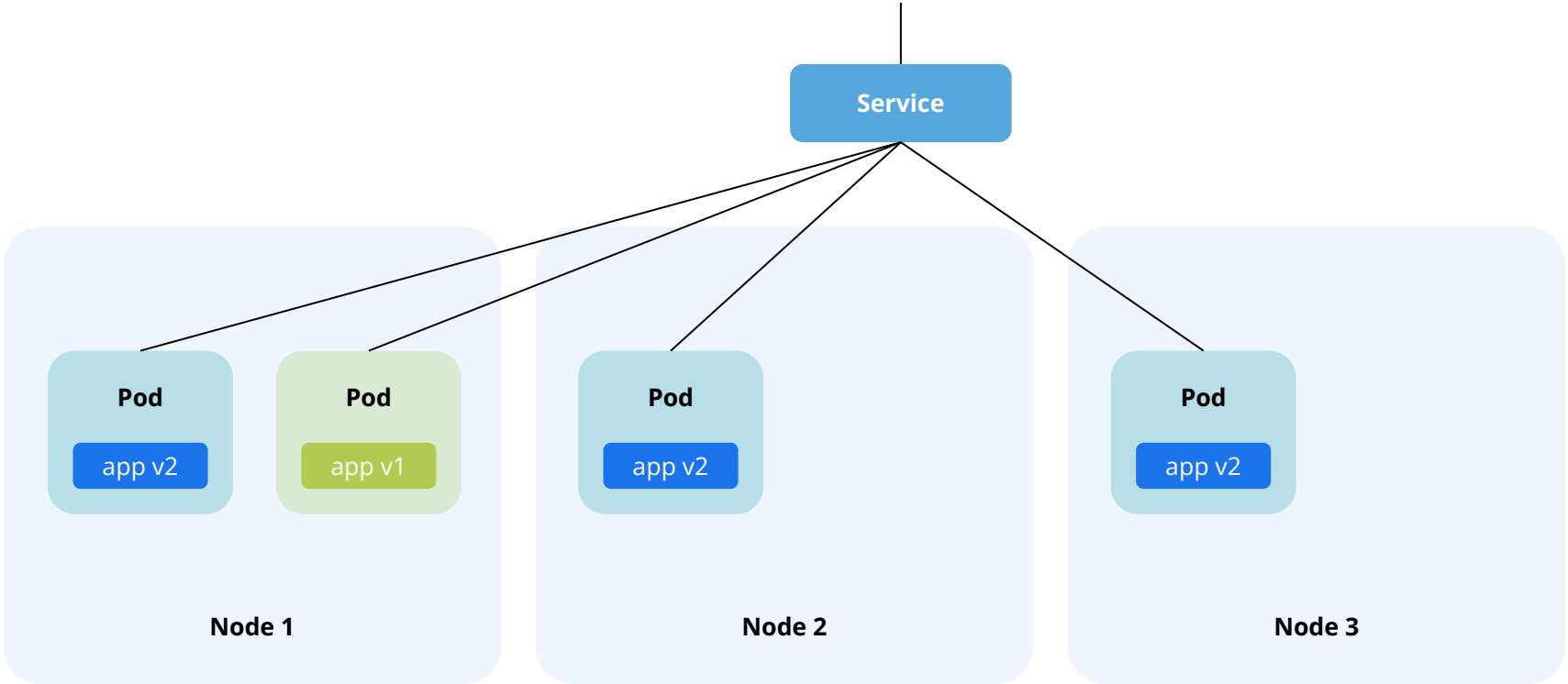
Rolling Update



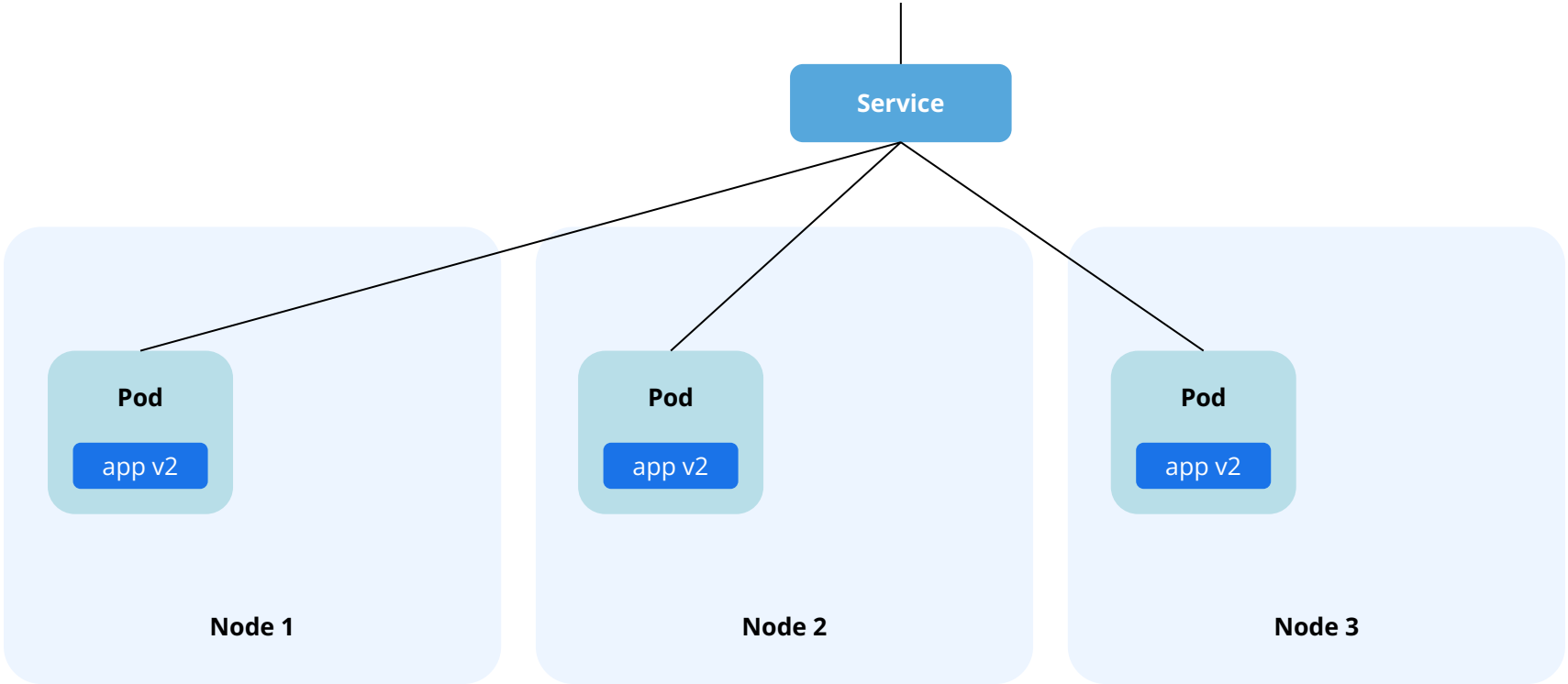
Rolling Update



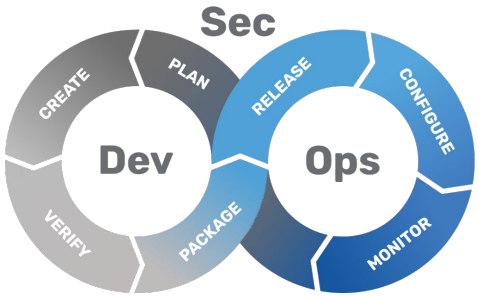
Rolling Update



Rolling Update



Kubernetes Workshop



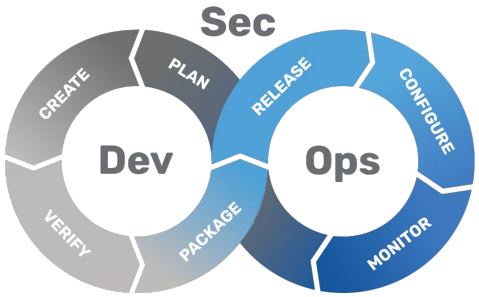
Transformation

kubernetes

Opsta Education

Skooldio

Kubernetes Manifest File



Transformation

kubernetes

Opsta Education

Skooldio

Kubernetes Manifest File

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: busybox
  labels:
    app: busybox
  namespace: default
spec:
  containers:
  - name: busybox
    image: busybox
    command:
      - sleep
      - "3600"
```

This is what called
Infrastructure as Code (IaC)

Kubernetes Manifest Structure

```
apiVersion: v1
kind: Pod
metadata:
  name: busybox
  labels:
    app: busybox
  namespace: default
spec:
  containers:
  - name: busybox
    image: busybox
    command:
      - sleep
      - "3600"
```

Define resource APIs version

Define resource kind

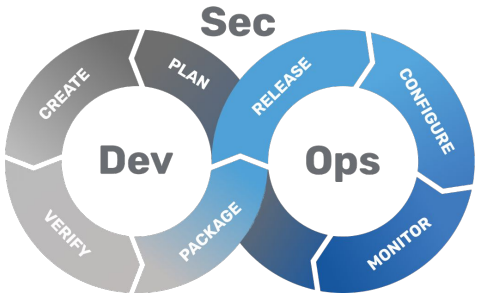
Define name of resource

[optional] define label of resource

[optional] define resource namespace

Define metadata of resource

Kubernetes Health Check

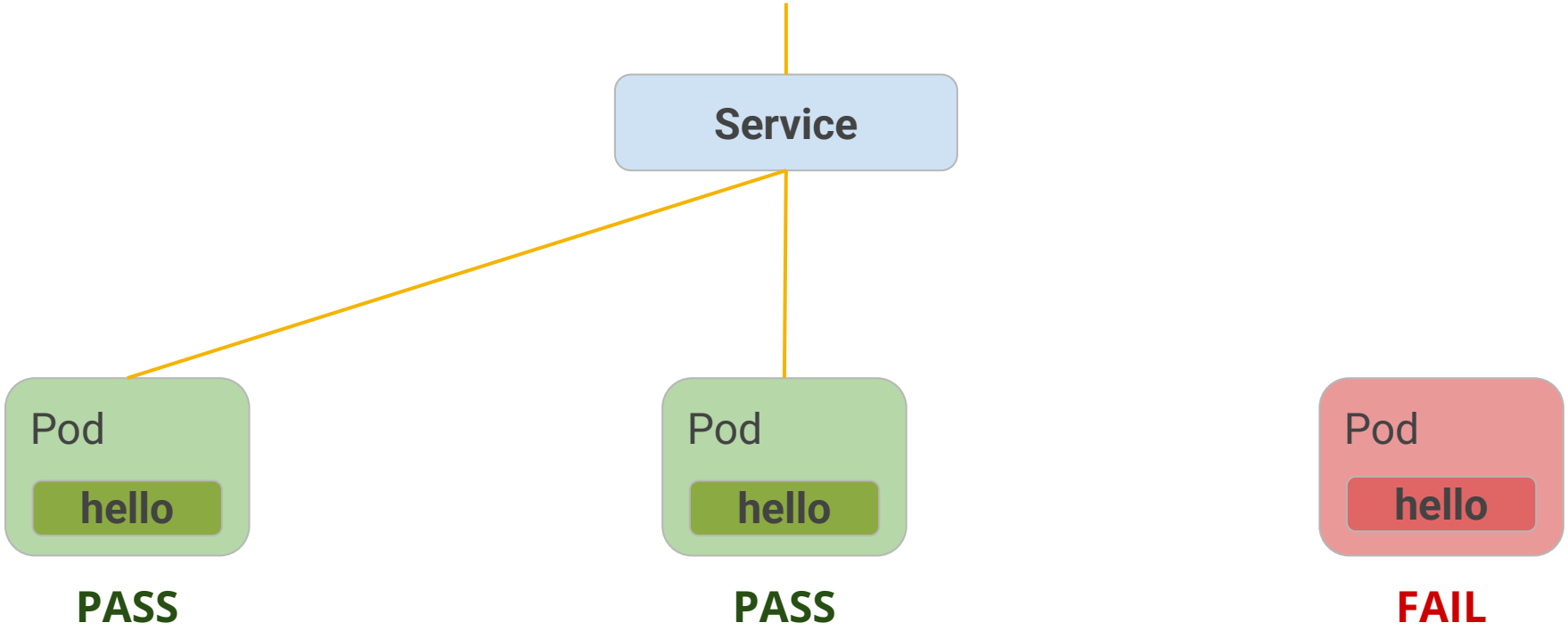


Transformation

Opsta Education

Skooldio

What is Health Check?

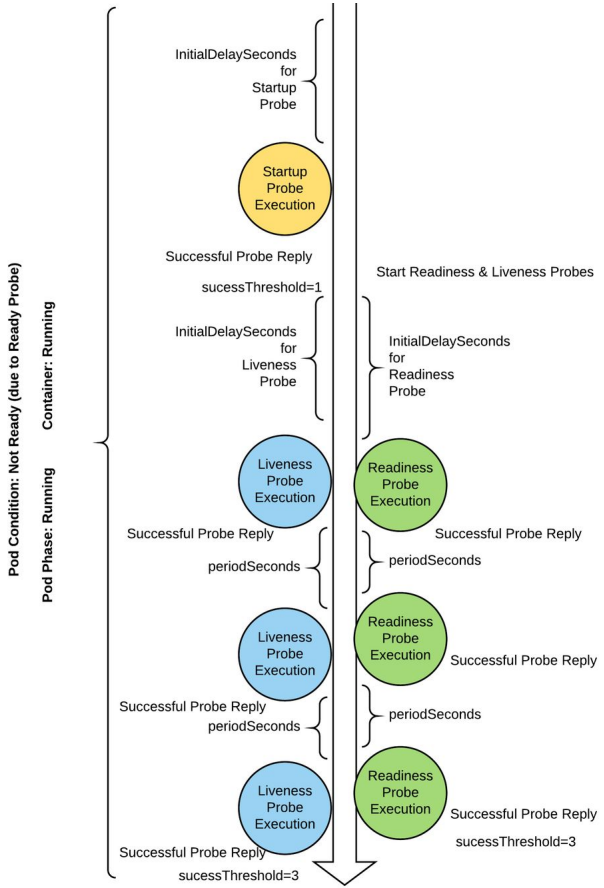


Type of Probes

- **exec**
  Executes a specified command inside the container. The diagnostic is considered successful if the command exits with a status code of 0.
- **grpc**
  Performs a remote procedure call using gRPC. The target should implement gRPC health checks. The diagnostic is considered successful if the status of the response is SERVING.
- **httpGet**
  Performs an HTTP GET request against the Pod's IP address on a specified port and path. The diagnostic is considered successful if the response has a status code greater than or equal to 200 and less than 400.
- **tcpSocket**
  Performs a TCP check against the Pod's IP address on a specified port. The diagnostic is considered successful if the port is open. If the remote system (the container) closes the connection immediately after it opens, this counts as healthy.

https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/#probe-check-methods

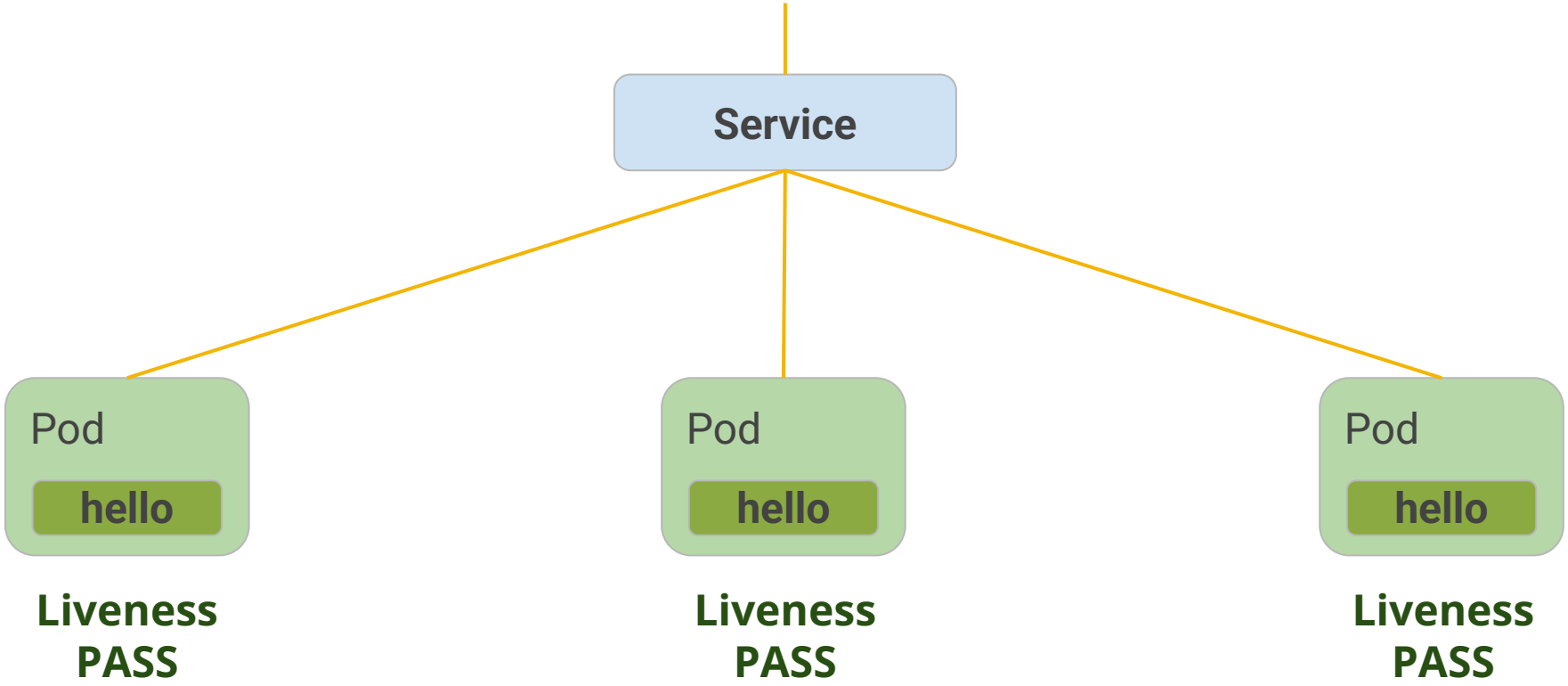
Startup, Liveness, and Readiness Probes



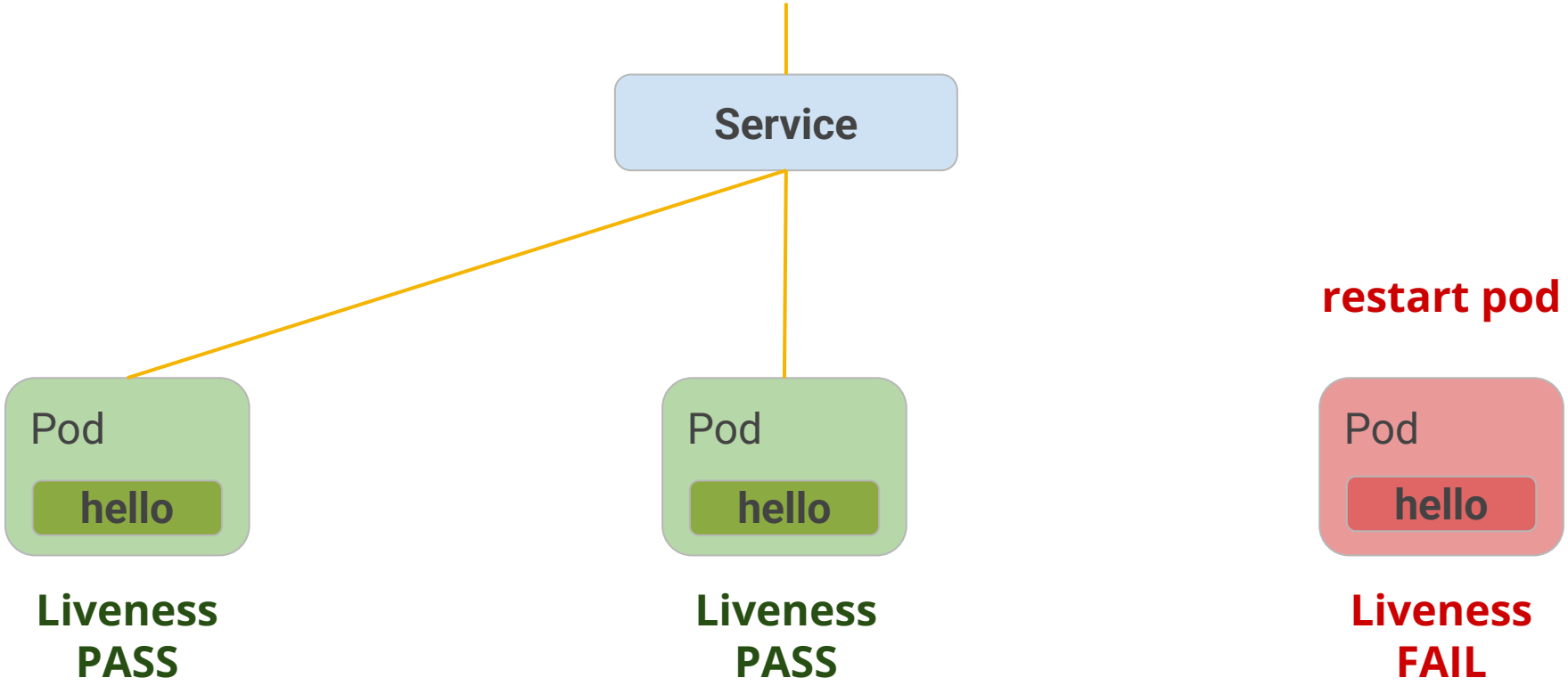
https://loft.sh/blog/kubernetes-probes-startup-liveness-readiness/

Skooldio Opsta Education

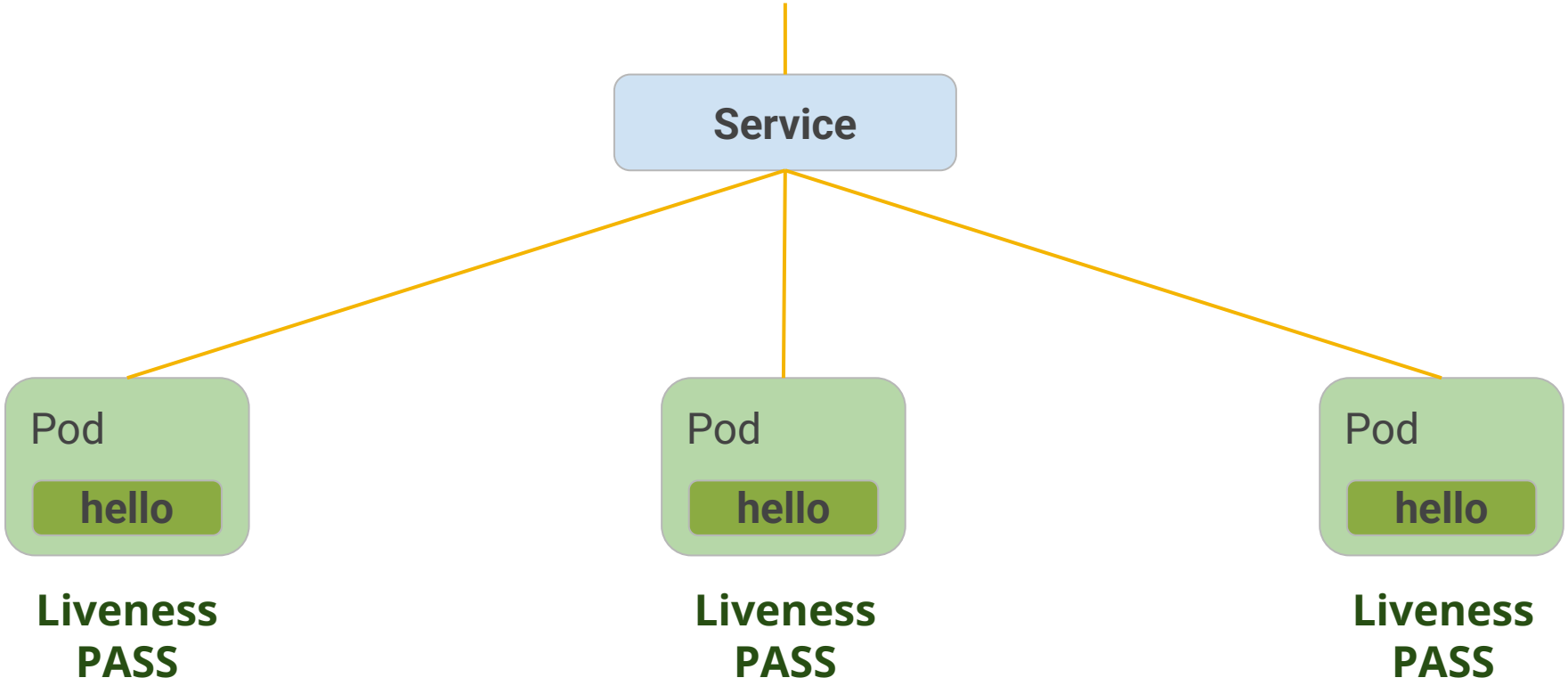
Startup Probe and Liveness Probe



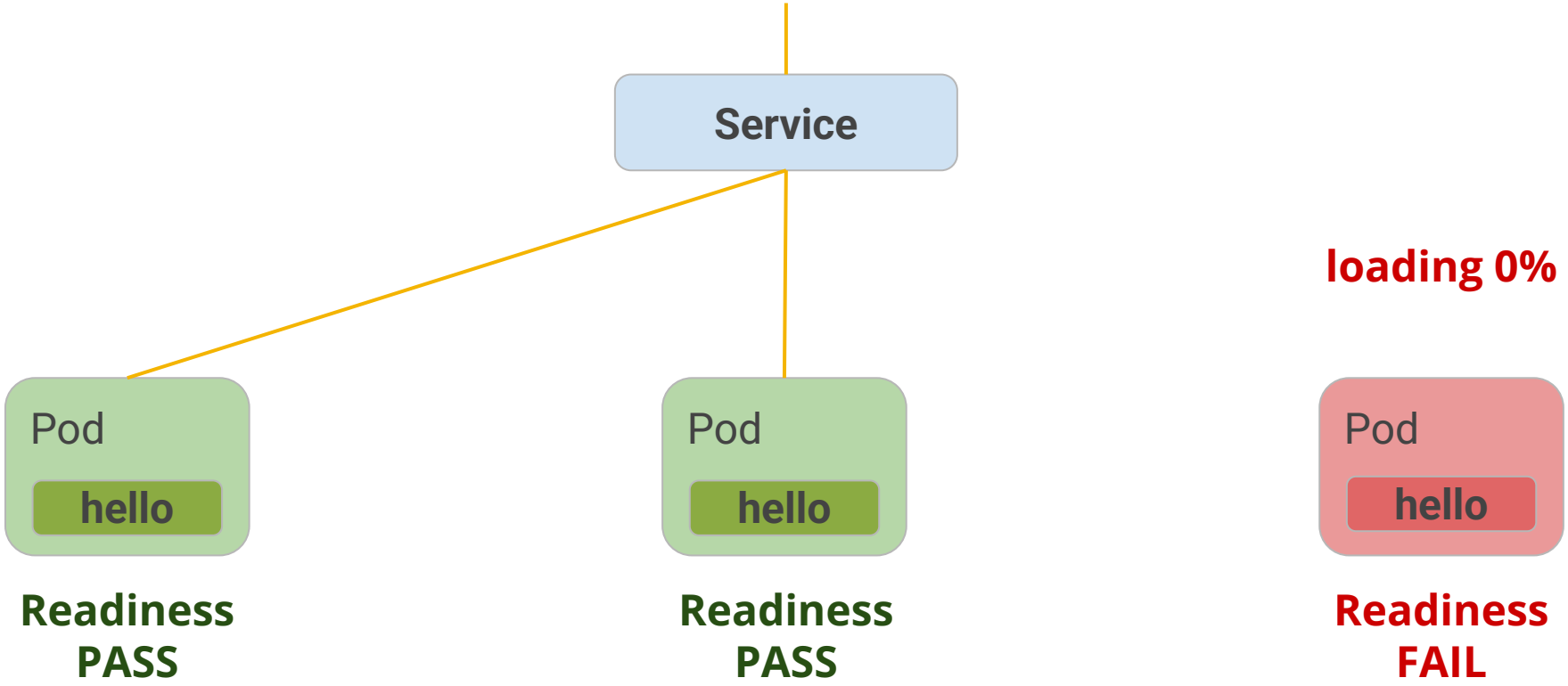
Startup Probe and Liveness Probe



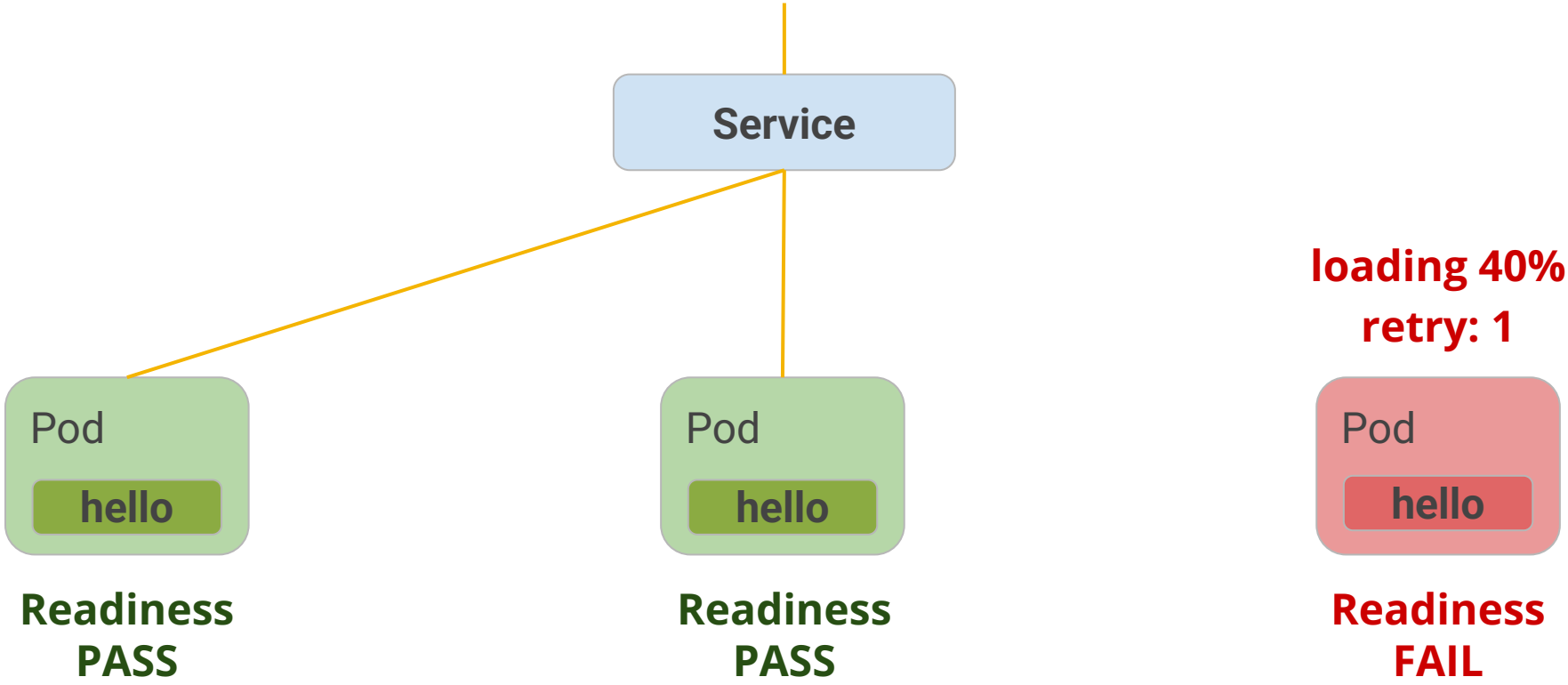
Startup Probe and Liveness Probe



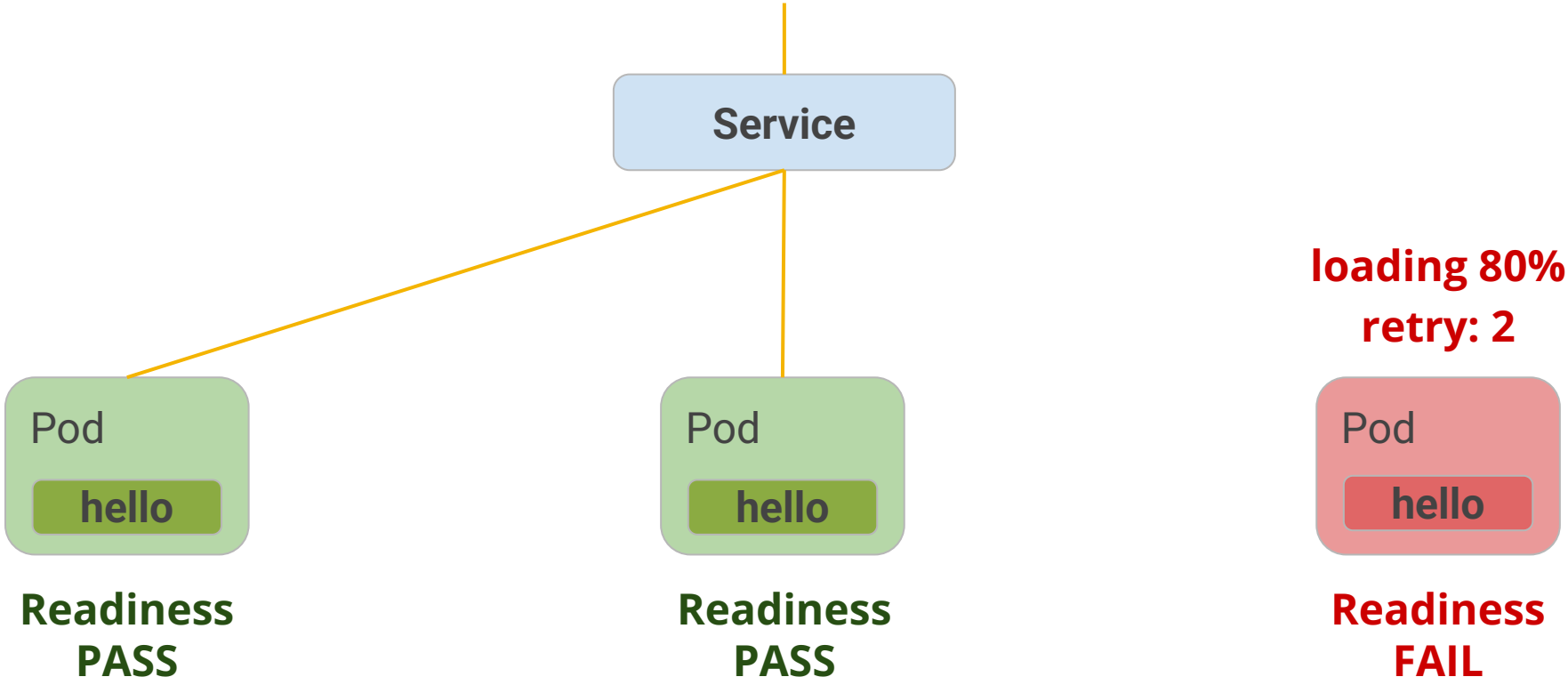
Readiness Probe



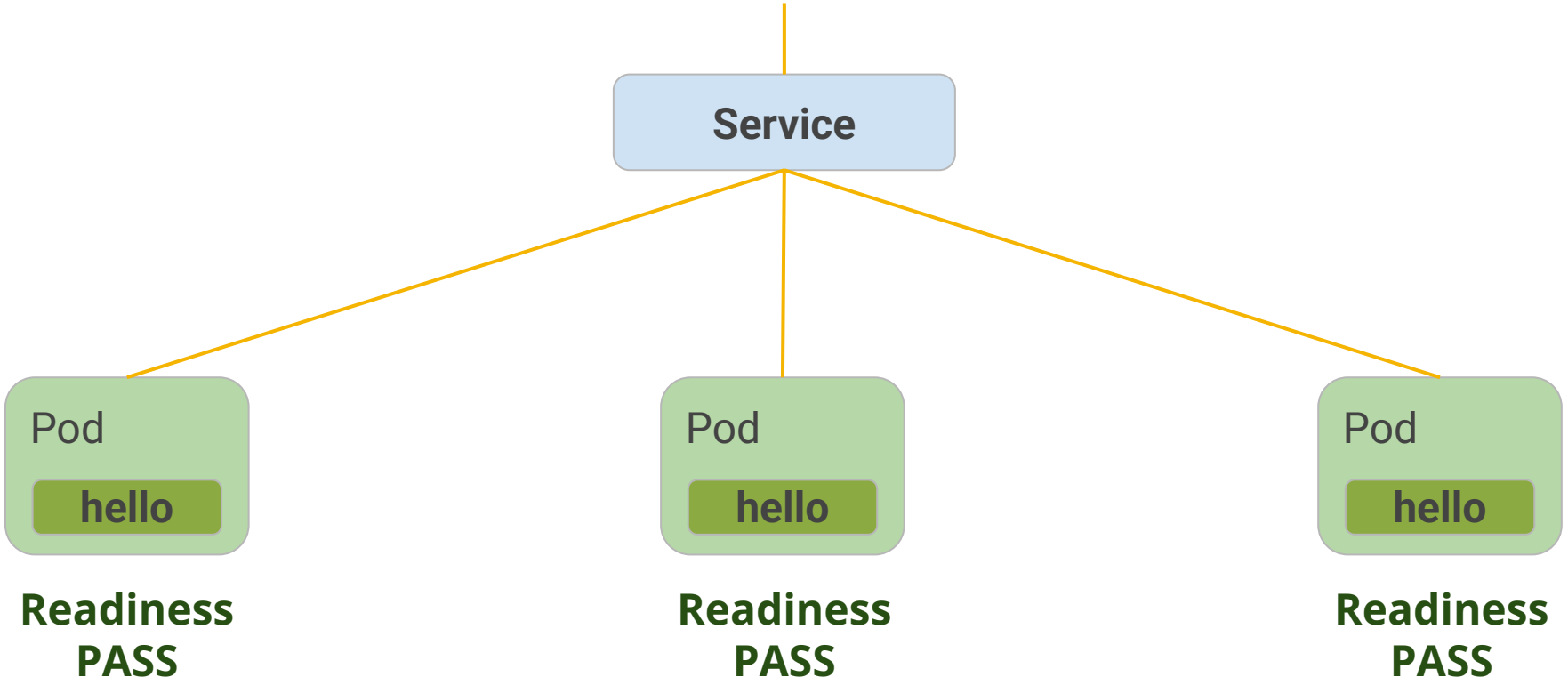
Readiness Probe



Readiness Probe



Readiness Probe

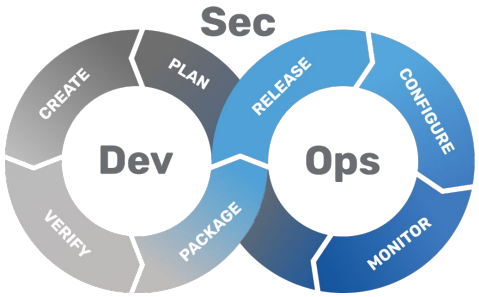


Probe Configuration

- **initialDelaySeconds:** How many seconds to wait after the container has started (default: 0)
- **periodSeconds:** Wait time between probe executions (default: 10)
- **timeoutSeconds:** Time to wait for the reply (default: 1)
- **successThreshold:** Number of successful probe executions to mark the container healthy (default: 1)
- **failureThreshold:** Number of failed probe executions to mark the container unhealthy (default: 3)

https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/#configure-probes

Kubernetes Manifest File Workshop



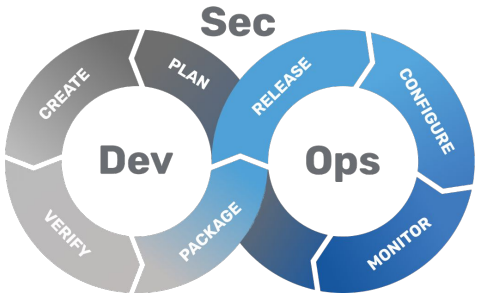
Transformation

kubernetes

Opsta Education

Skooldio

Kubernetes ConfigMap



Transformation

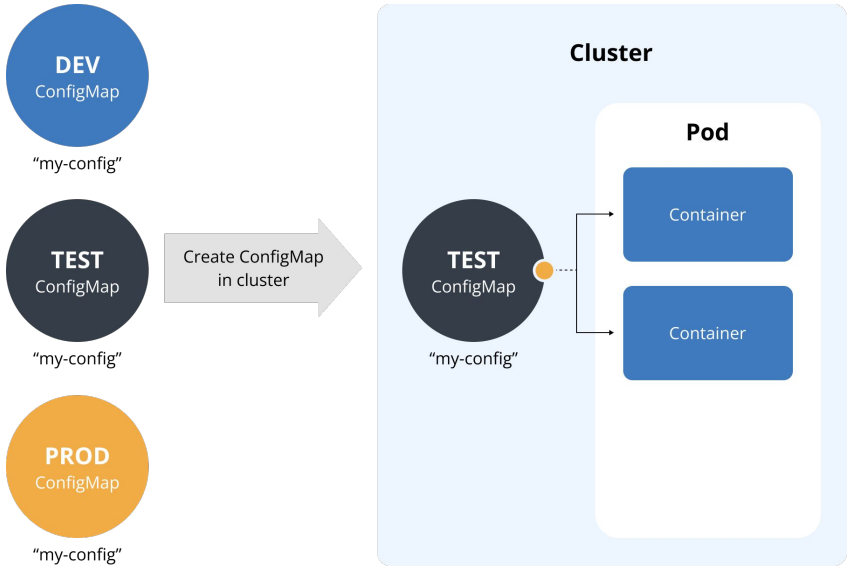
kubernetes

Opsta Education

Skooldio

Kubernetes ConfigMap

ConfigMap manages variable configurations. It **decouples** configurations from container images, ensuring the **portability** of containers. ConfigMap is actually a "map" in the **key: value** format, where a "key" indicates a config **name** while its "value" indicates the **content**. The ConfigMap size is limited to 1 MB.



ConfigMap configMapKeyRef Usage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: game-key
data:
  player_initial_lives: "3"
  ui_theme: "dark"
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-cm-key
spec:
  containers:
  - name: demo
    image: alpine
    command: ["sleep", "3600"]
    env:
    - name: PLAYER_INITIAL_LIVES
      valueFrom:
        configMapKeyRef:
          name: game-key
          key: player_initial_lives
    - name: UI_THEME
      valueFrom:
        configMapKeyRef:
          name: game-key
          key: ui_theme
```



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-cm-key
spec:
  containers:
  - name: demo
    image: alpine
    command: ["sleep", "3600"]
    env:
    - name: PLAYER_INITIAL_LIVES
      value: "3"


    - name: UI_THEME
      value: "dark"
```

ConfigMap configMapRef Usage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: game-key
data:
  PLAYER_INITIAL_LIVES: "3"
  UI_THEME: "dark"
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-cm-key
spec:
  containers:
  - name: demo
    image: alpine
    command: ["sleep", "3600"]
    envFrom:
    - configMapRef:
        name: game-key
```

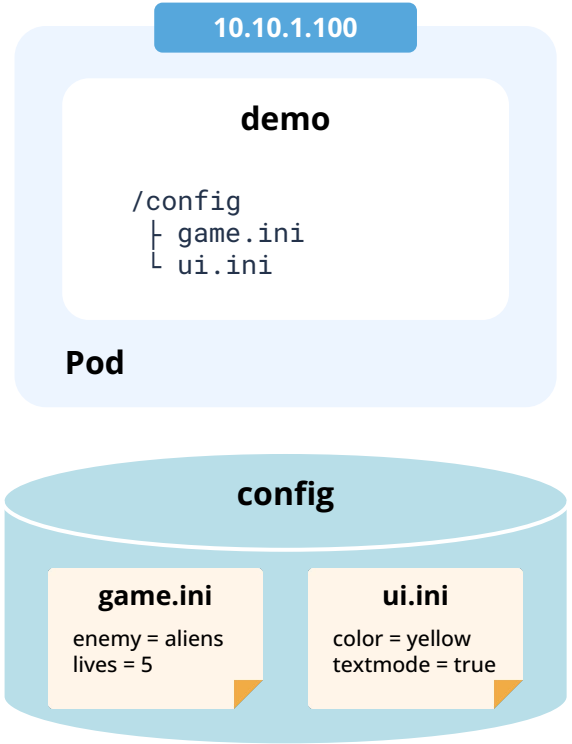


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-cm-key
spec:
  containers:
  - name: demo
    image: alpine
    command: ["sleep", "3600"]
    env:
    - name: PLAYER_INITIAL_LIVES
      value: "3"


    - name: UI_THEME
      value: "dark"
```

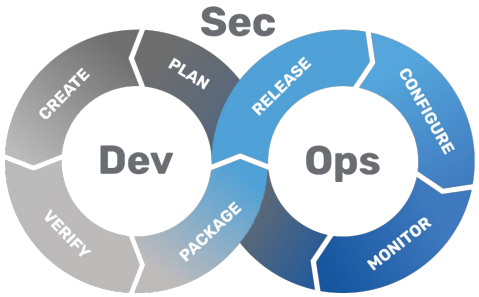
ConfigMap File Usage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: game-file
data:
  game.ini: |
    enemy = aliens
    lives = 5
  ui.ini: |
    color = yellow
    textmode = true
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-cm-file
spec:
  containers:
  - name: demo
    image: alpine
    command: ["sleep", "3600"]
    volumeMounts:
    - name: config
      mountPath: "/config"
      readOnly: true
  volumes:
  - name: config
    configMap:
      name: game-file
      items:
      - key: "game.ini"
        path: "game.ini"
      - key: "ui.ini"
        path: "ui.ini"
```



Kubernetes Secret



Transformation

kubernetes

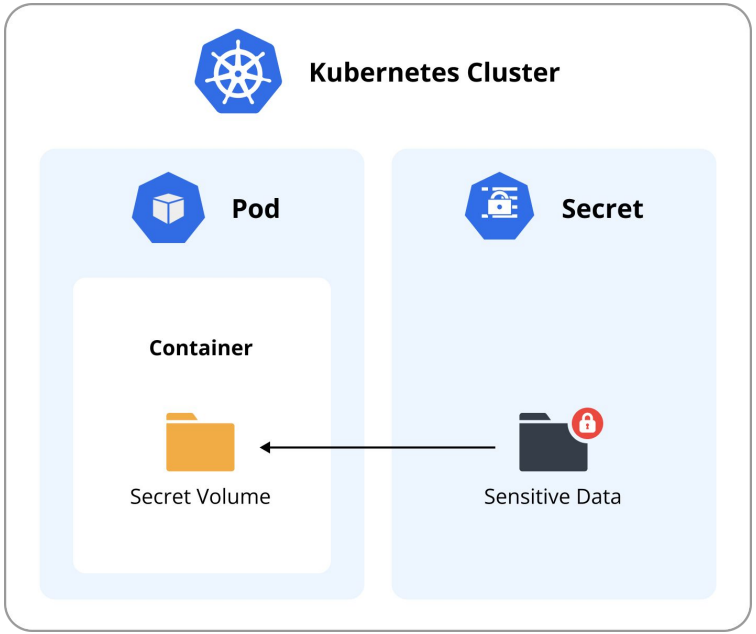
Opsta Education

Skooldio

Kubernetes Secret

Kubernetes Secret is a resource object used to store some **sensitive** information, such as passwords and tokens. The sensitive information is **stored in Base64 encoding**. There are four types of Secret data

- **Opaque**: a common Secret file
- **dockerconfigjson**: pulls images from a private warehouse
- **Service-account-token**: authenticates the service account identity
- **bootstrap.token** verifies node access to clusters



How to use Kubernetes Secrets

Create Secret

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: rabbitmq
  namespace: default
type: Opaque
# Write in base64
data:
  RabbitPass: cmFiYml0bXE=
# Write in plaintext
stringData:
  RabbitPlain: P@ssw0rd
```

Inject Secret into Container Environment Variable

```yaml
apiVersion: apps/v1
kind: Pod
metadata:
  name: myapp
  namespace: default
spec:
  containers:
  - name: myapp
    image: myapp:latest
    env:
      - name: RABBITMQ_PASSWORD
        valueFrom:
          secretKeyRef:
            name: rabbitmq
            key: RabbitPass
```

Read Secret from Code

[NodeJS]

```
process.env.RABBITMQ_PASSWORD;
```

[Python]

```
import os
os.getenv('RABBITMQ_PASSWORD')
```

[Golang]

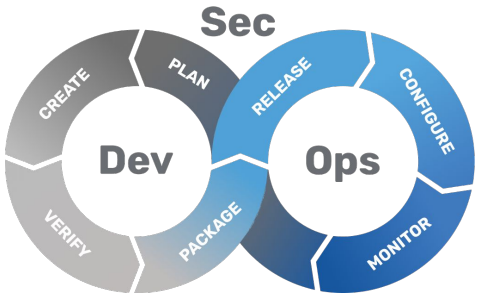
```
package main
import (
    "os"
)
os.Getenv("RABBITMQ_PASSWORD")
```

Using Private Docker Registry with Secrets

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: myregistrykey
type: kubernetes.io/dockerconfigjson
stringData:
  .dockerconfigjson: |
    {
      "auths": {
        "https://index.docker.io/v1/": {
          "auth": "c3R...zE2"
        }
      }
    }
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: app-private
spec:
  containers:
  - name: app-private
    image: private-image:latest
  imagePullSecrets:
  - name: myregistrykey
```

Kubernetes Ratings Service Workshop

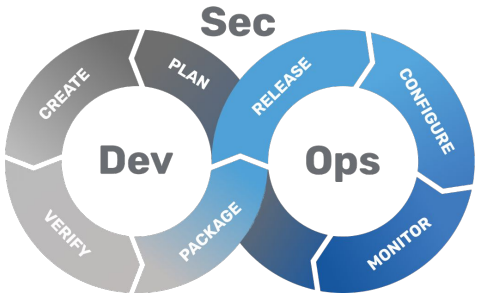


Transformation

Opsta Education

Skooldio

Introduction to Helm



Transformation

HELM

Opsta Education

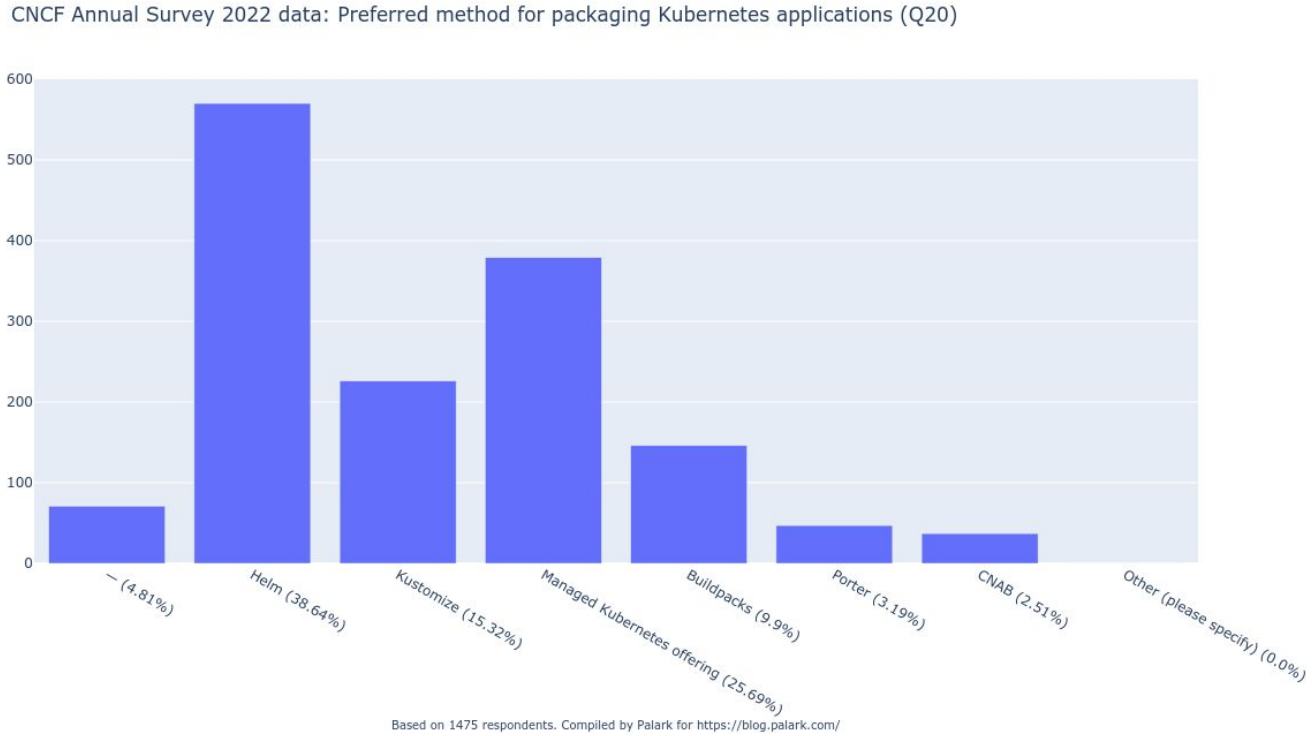
Skooldio

Problem with K8s Manifest File

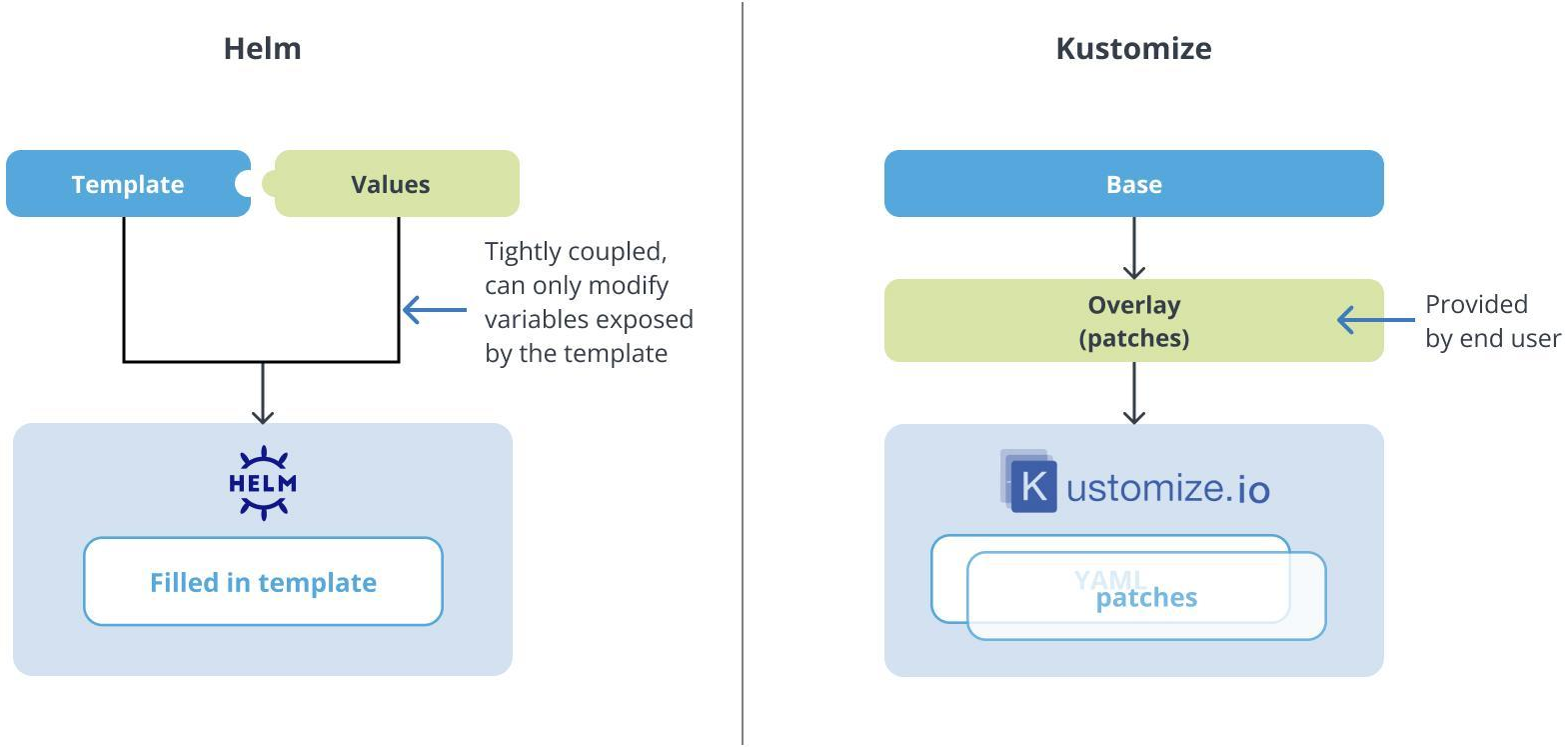
- 1 Microservice consist of:
  - Deployment
  - Service
  - Ingress
  - Configmap
  - Secret
  - Job (if needed)
- More effort for **operate** and difficult control **environment** values.
- Hard to manage release (Rollback, Rollout, history).
- Hard to reuse configuration template cause specification environment.

Kubernetes Packaging Tools

Survey packaging Kubernetes application



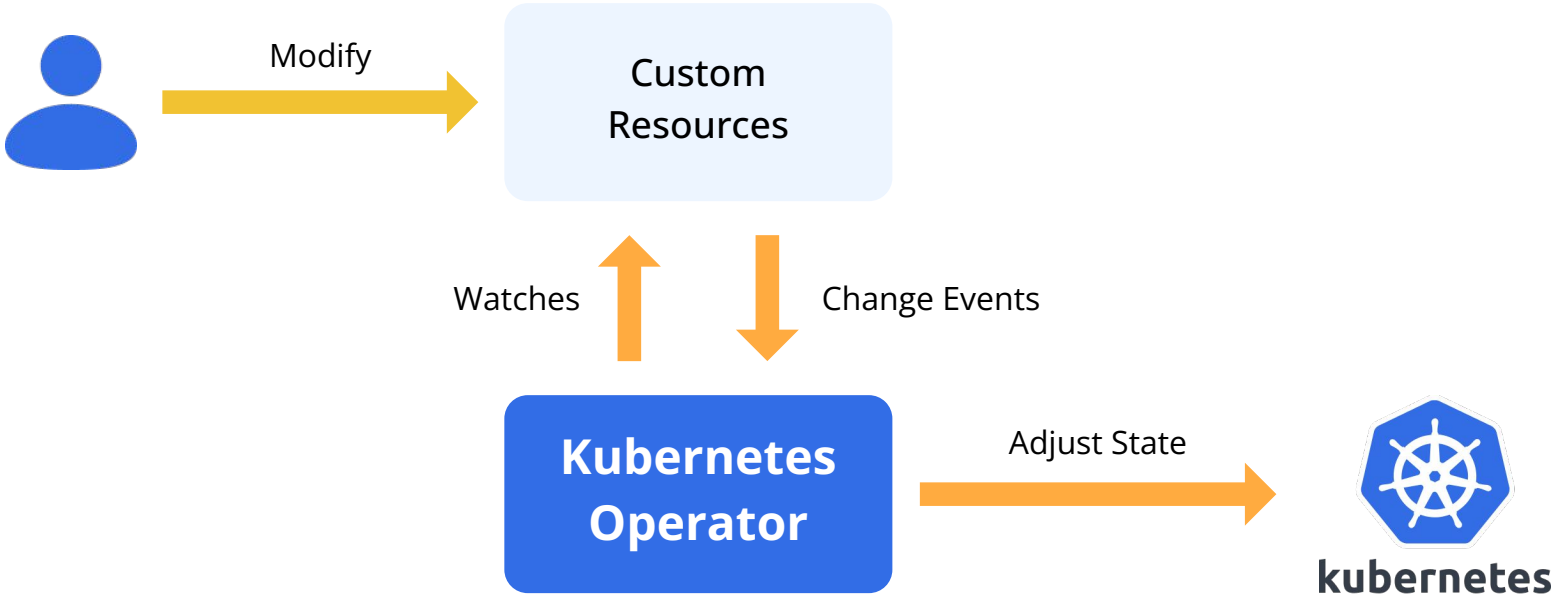
Helm vs Kustomize



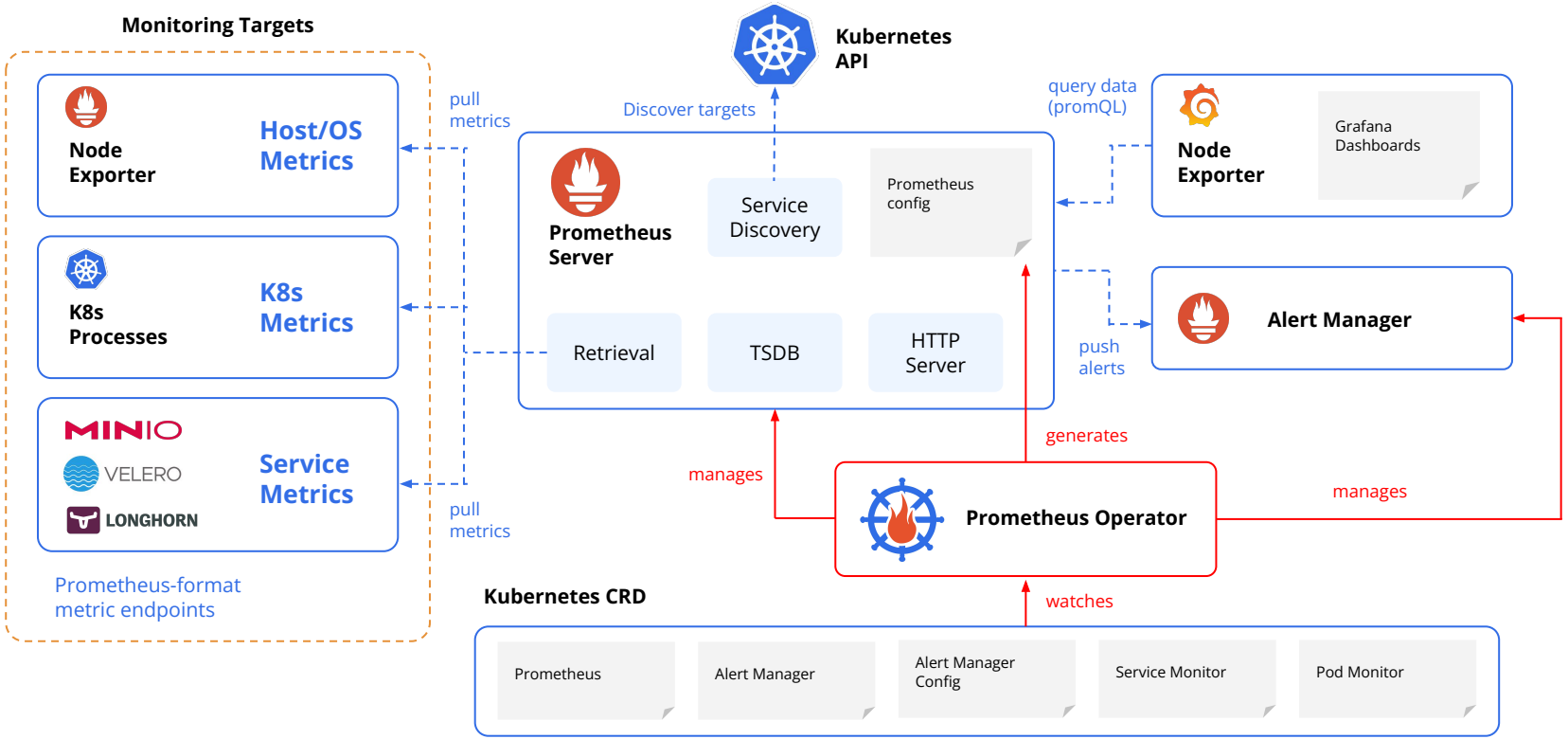
Helm and Operator

Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
Basic Install	**Seamless Upgrades**	**Full Lifecycle**	**Deep Insights**	**Auto Pilot**
Automated application provisioning and configuration management	Patch and minor version upgrades supported	App lifecycle, storage lifecycle (backup, failure recovery)	Metrics, alerts, log processing and workload analysis	Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning

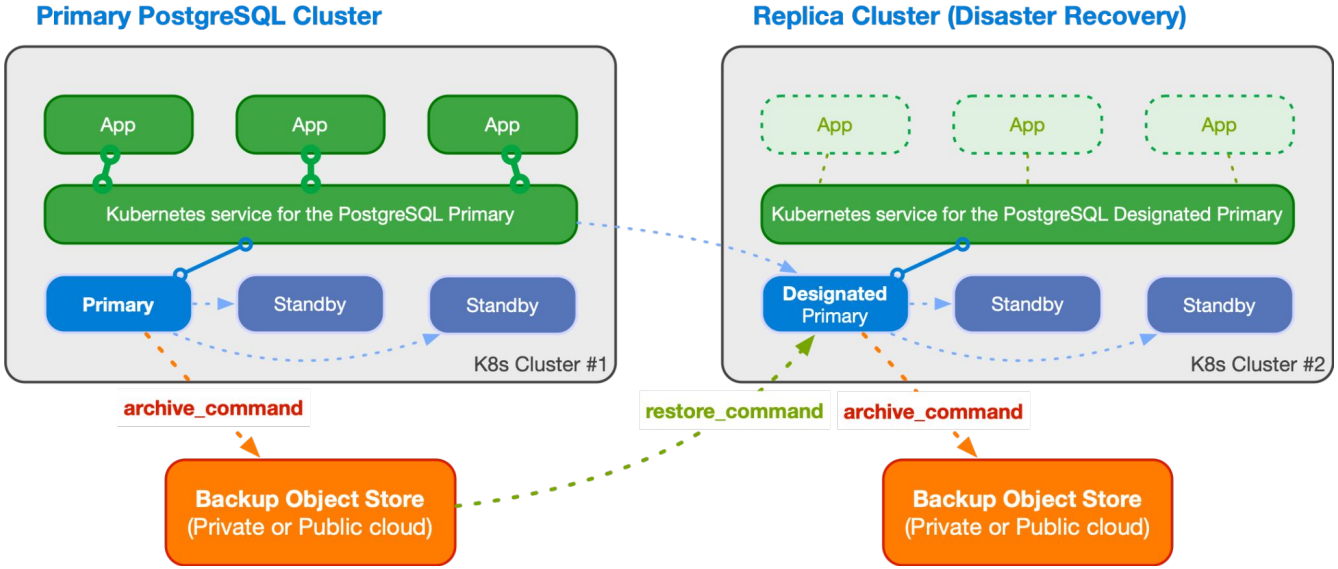
Kubernetes Operator Pattern



Prometheus Operator Architecture



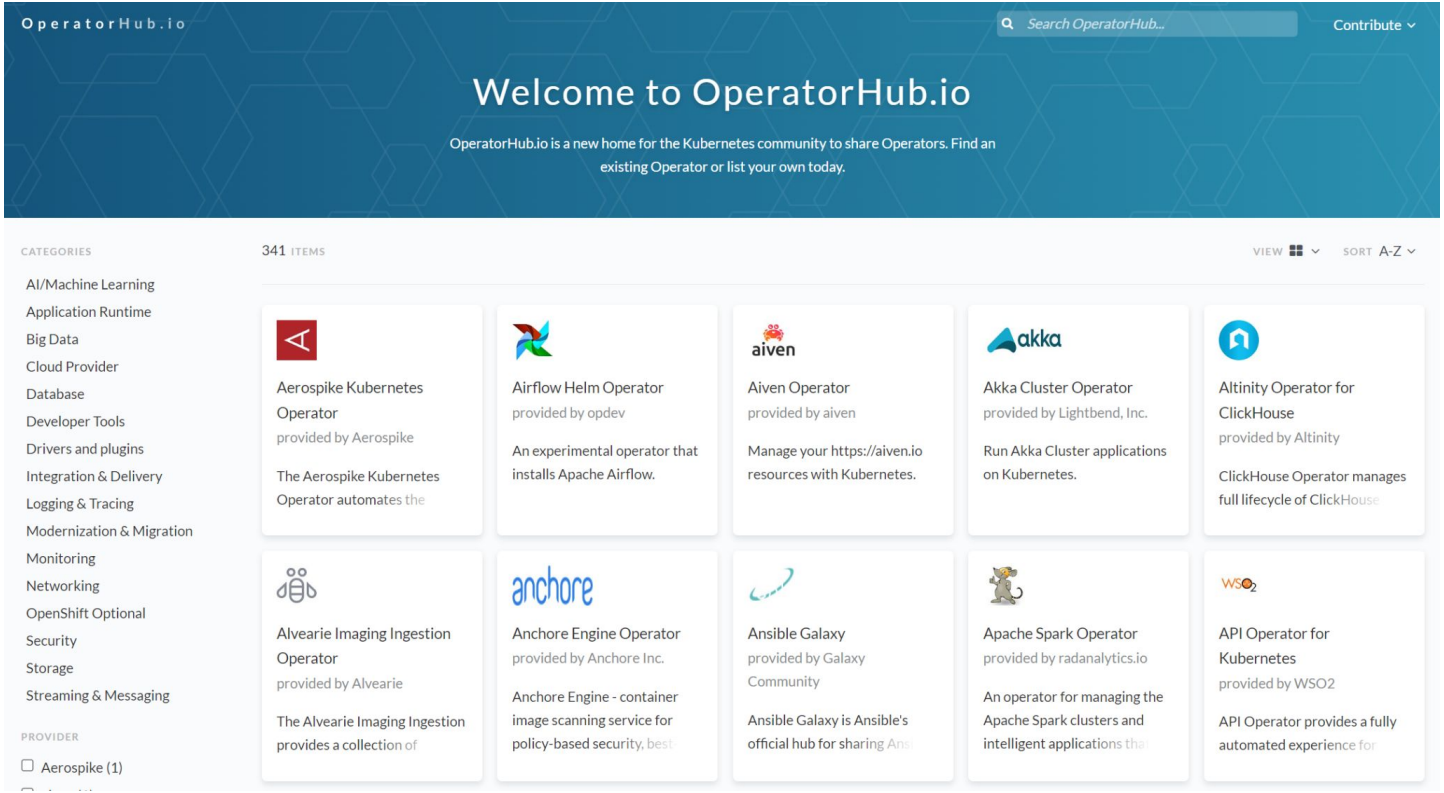
CloudNativePG



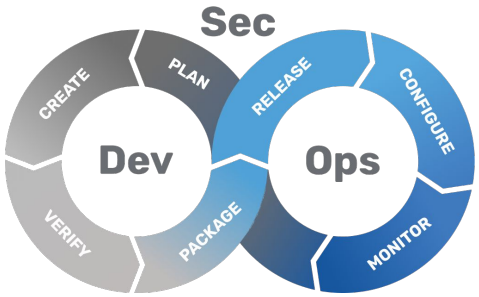

```yaml
apiVersion: postgresql.cnpg.io/v1
kind: Cluster
metadata:
  name: postgres-cluster
spec:
  instances: 3
  bootstrap:
    initdb:
      database: ratings
      owner: ratings
      secret:
        name: ratings-postgres-secret
  storage:
    size: 1Gi
  walStorage:
    size: 1Gi
  backup:
    volumeSnapshot:
      className: volume-snapshot
    retentionPolicy: "10d"
    barmanObjectStore:
      destinationPath: "gs://opsta-backup"
  monitoring:
    enablePodMonitor: true
```


Skooldio Opsta Education

OperatorHub.io



Helm



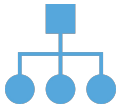
Transformation

Opsta Education

Skooldio

Helm

Helm is the package manager for Kubernetes



Manage Complexity

Charts describe even the most complex apps; provide repeatable application installation, and serve as a single point of authority.

Easy Updates

Take the pain out of updates with in-place upgrades and custom hooks.

Simple Sharing

Charts are easy to version, share, and host on public or private servers.

Rollbacks

Use helm rollback to roll backto an older version of a release with ease.

Helm Terminology

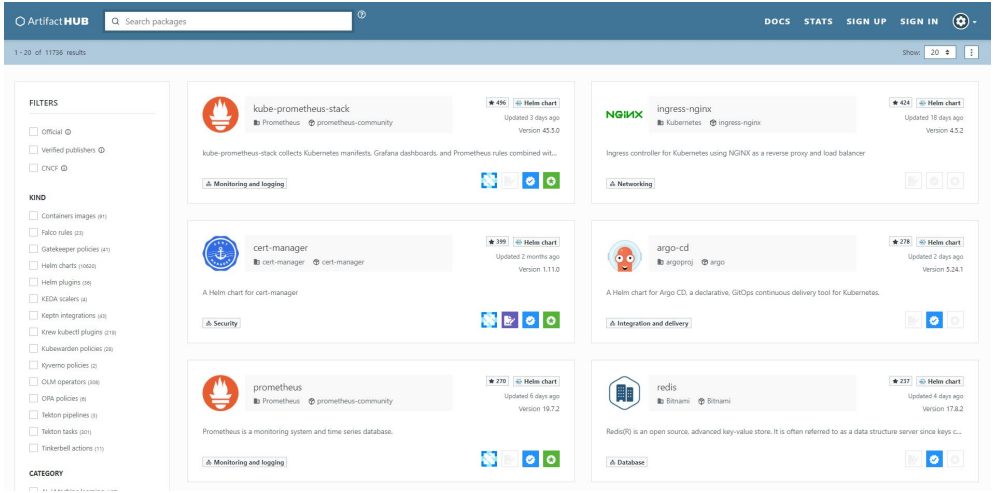
Repository
a collection of released charts.

Chart
a bundle of information necessary to create an instance of a kubernetes application

Release
a running instance of a chart, combined with a specific configuration

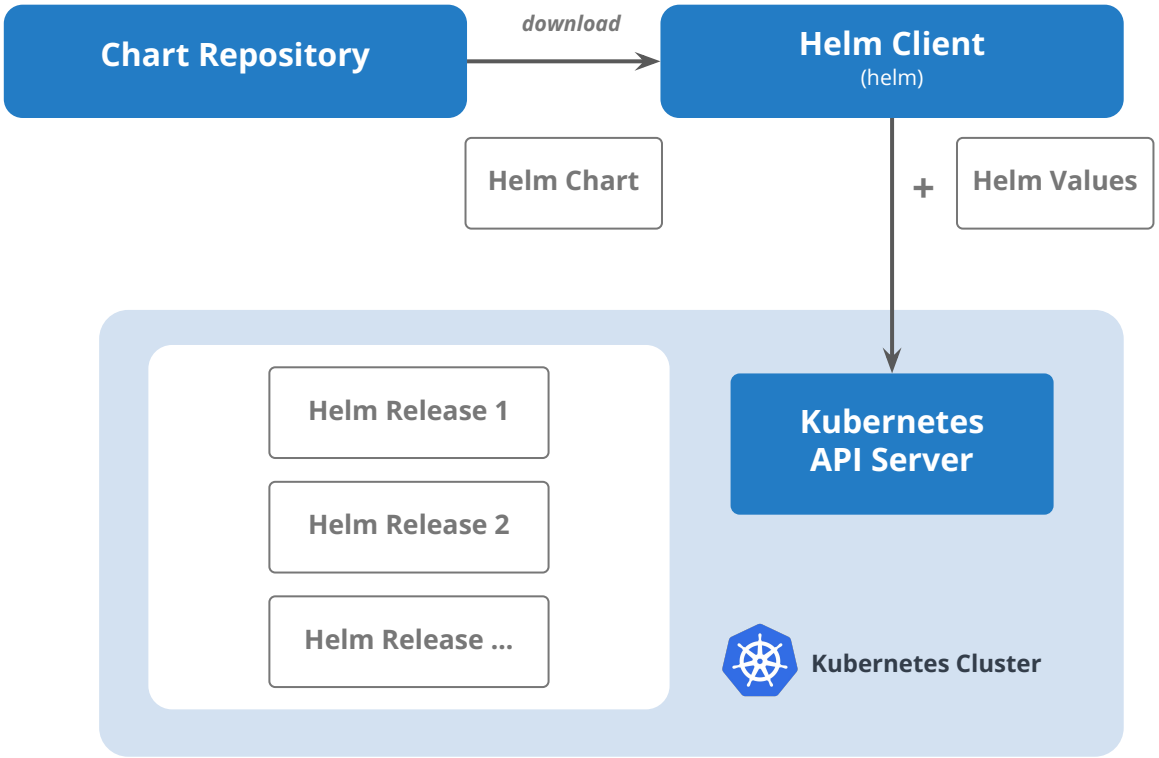
Helm Charts

- Helm Charts helps you define, install, and upgrade
- Charts are easy to create, version, share, and publish
- Public Helm Charts are at https://artifacthub.io

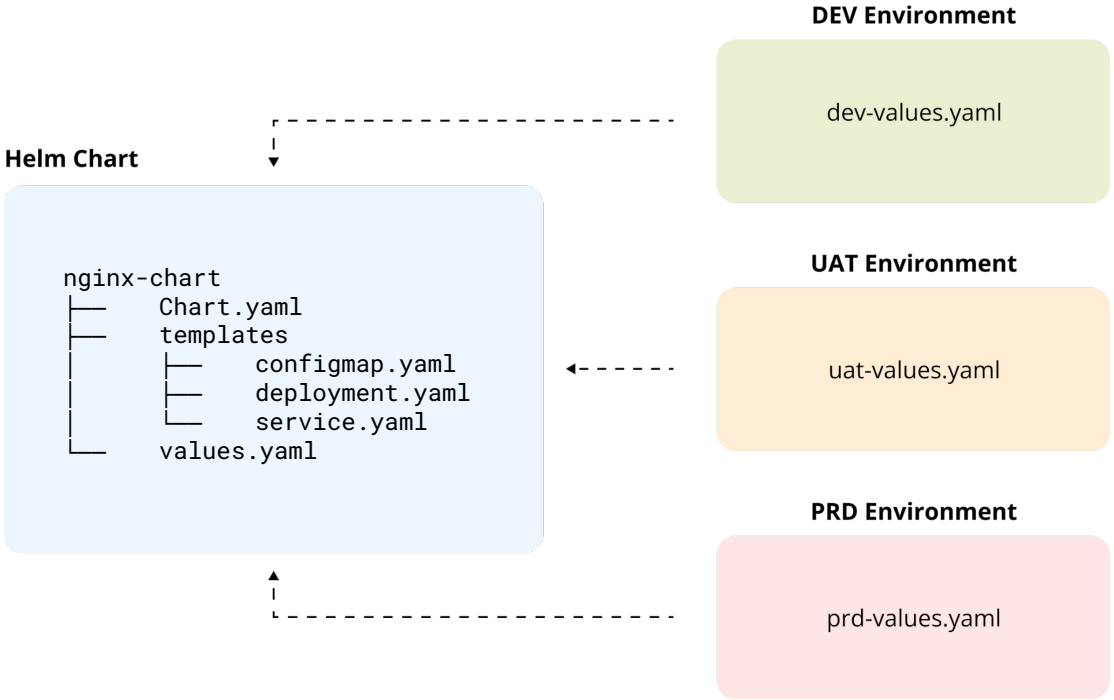


Skooldio Opsta Education

Helm 3 Architecture



Helm Chart and Helm Value Structure



Helm Templating

- Using standard Kubernetes YAML manifest files
- Parameterizable via Go template language
- Sprig function library included (https://masterminds.github.io/sprig/)
- Chart should provide default values in *values.yaml* file

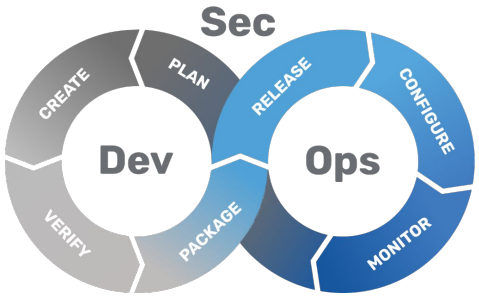
values file

```
name: ratings
image:
  repository: ratings
  tag: dev
  pullPolicy: Always
```

manifest file

```
apiVersion: v1
kind: Pod
metadata:
  name: {{ .Values.name }}
spec:
  containers:
  - name: {{ .Values.name }}
    image: {{ .Values.image.repository }}:{{ .Values.image.tag }}
    imagePullPolicy: {{ .Values.image.pullPolicy }}
```

Helm Structure



Transformation

Opsta Education

Skooldio

Helm Structure (1)

1. **Charts**
    a. Templates
    b. Helpers
    c. Metadata (README.md)
2. **Values**

Helm Structure (2)

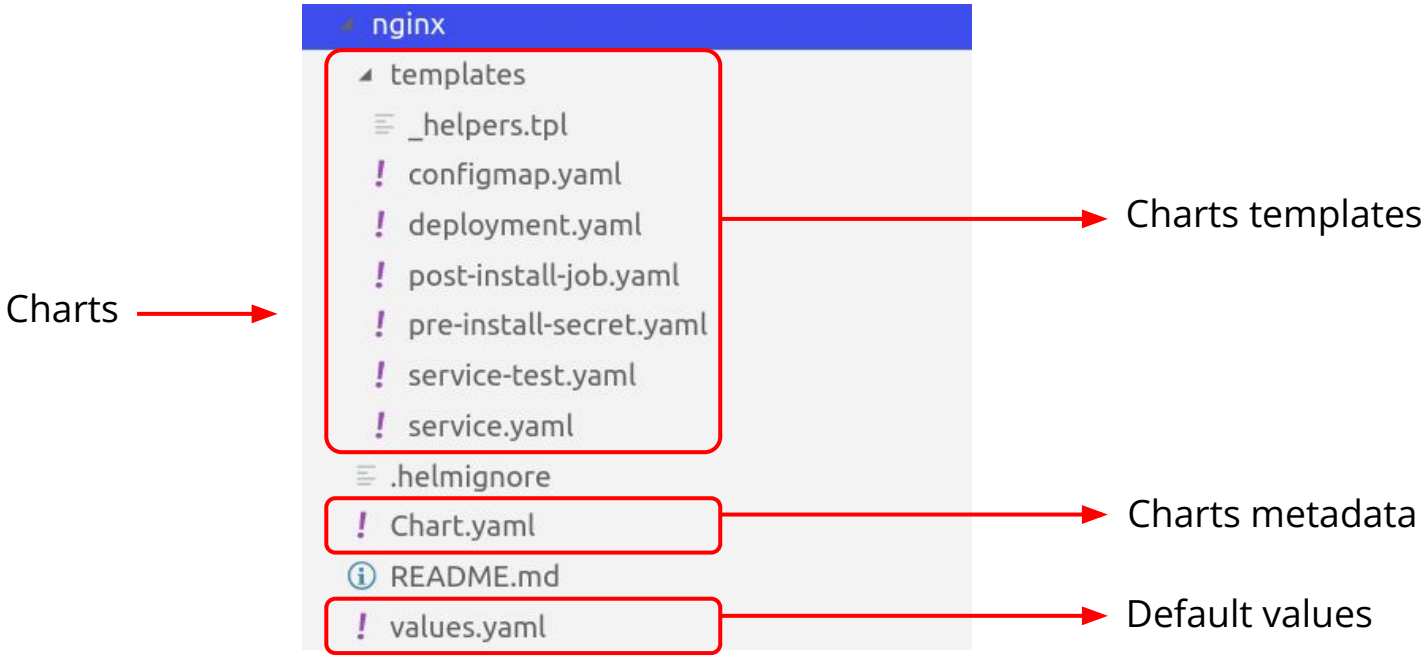
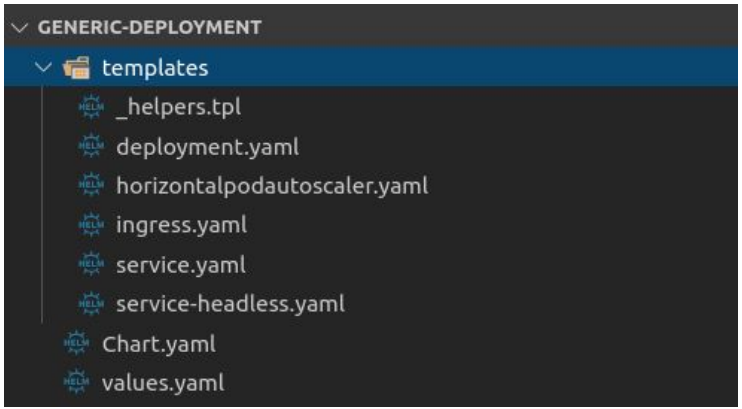


Chart Template

- Template file contain resources **.yaml** for needed application.
- Each resource should be definite **own resource**
- Template file name should reflect with **resource kind** in name (app-deployment.yml, app-service.yaml)



Built-in Object

- **Release**: Describe the release itself (.Release.Name, .Release.Namespace)
- **Value**: Passed into template from values.yaml (.Values)
- **Chart**: The content in Chart (.Chart.Name, .Chart.Version)
- **Template**: Contains information about current template (template ...)

Helm Value File (1)

- Values file is formatted in **.yaml**
- Chart will merge value to apply with **template**.
- There are 2 type format consist of **Nested** (Recommend) and **Flat**.
    - **Flat** format ease to read and use, appropriate less variables and uncomplex application.
    - **Nested** format is layers structure, can use more exception case or support more variables to use.

```yaml
# Nested
application:
  name: example
  port: 8080

# Flat
application_name: example
application_port: 8080
```

Helm Value File (2)

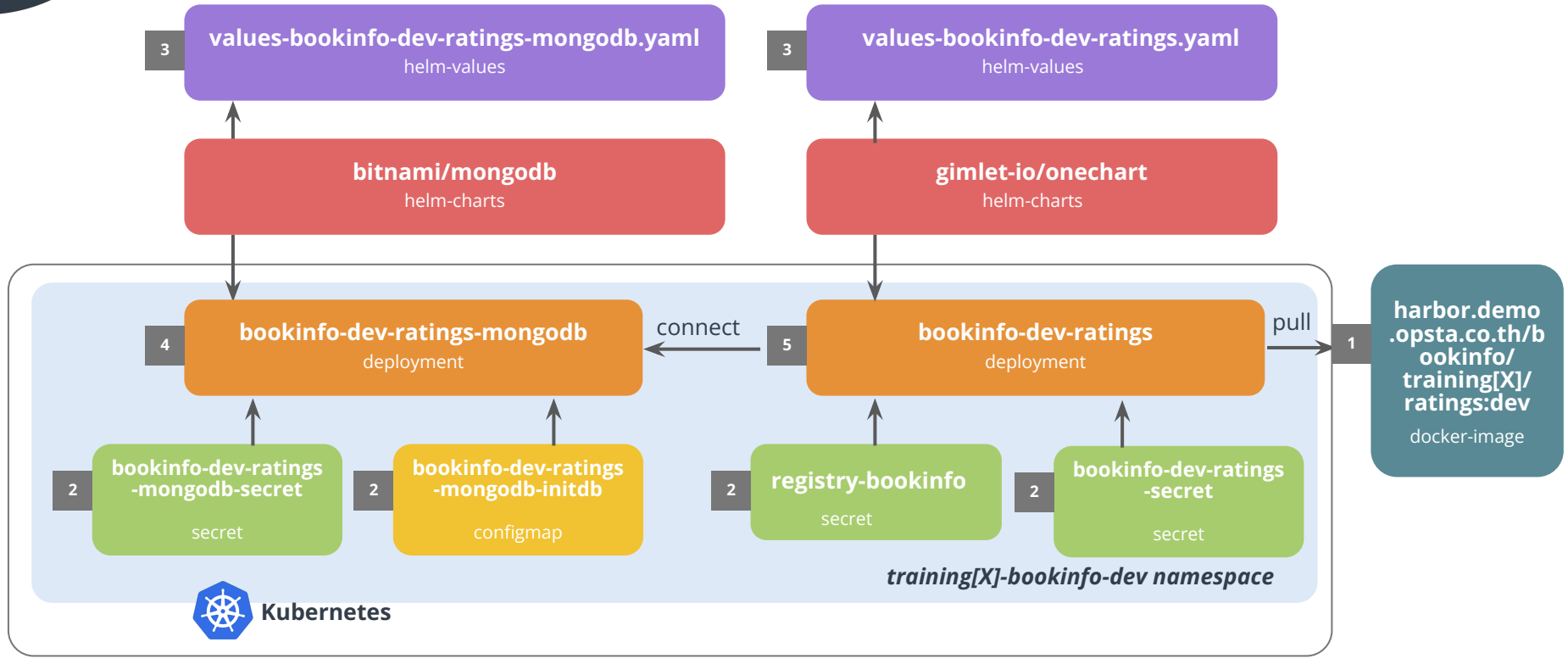
- Naming convention of variable should reflect with action it.
- Variable should begin with lowercase
- Value will be apply with helm **install** or **upgrade**

  (Ex. helm install/upgrade stable/mysql)
- Value can passed to **--set-string** flag or **--set-file** flag

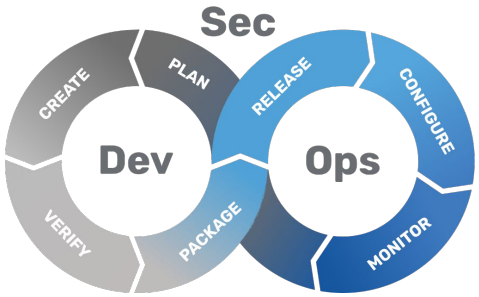
  for import file value configure

```yaml
replicaCount: 1

image:
  repository: sample/sample
  tag: latest
  pullPolicy: IfNotPresent
# imagePullSecrets:
#   - sample-image-secret

restartPolicy: Always

nameOverride: generic-deployment
fullnameOverride: generic-deployment
```

Rating Services Resources in a namespace



Helm Workshop



Transformation

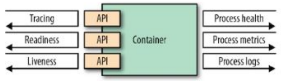
HELM

Opsta Education

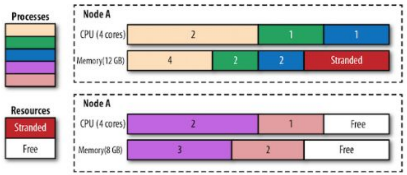
Skooldio

10 design patterns for Kubernetes

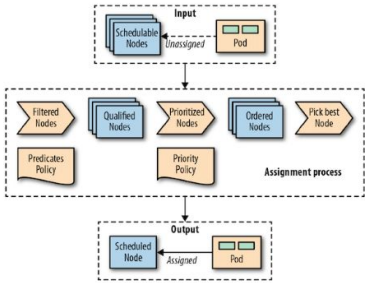
Foundational



Health Probe

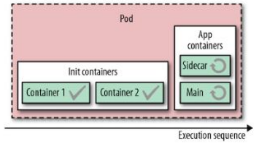


Predictable Demands

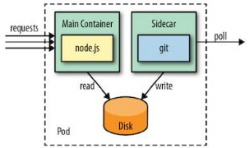


Automated Placement

Structural

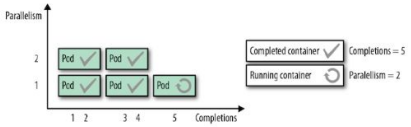


Init Container

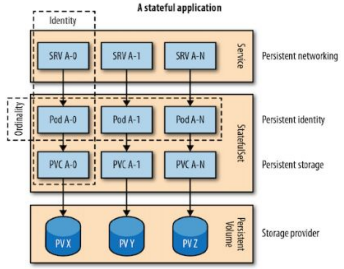


Sidecar

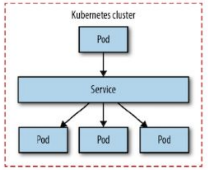
Behavioural



Batch Job

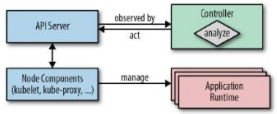


Stateful Service

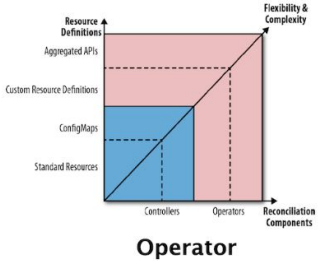


Service Discovery

Higher-level



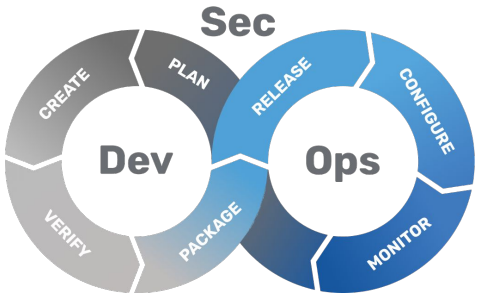
Controller



Operator

https://developers.redhat.com/blog/2020/05/11/top-10-must-know-kubernetes-design-patterns

Introduction to CI/CD



Transformation

Opsta Education

Skooldio

CI/CD

Continuous Integration (CI)

and **Continuous Delivery/Deployment (CD)**

Continuous Integration

Continuous Integration is the system that aims to continuously integrate the code generated by the development team with the codebase.

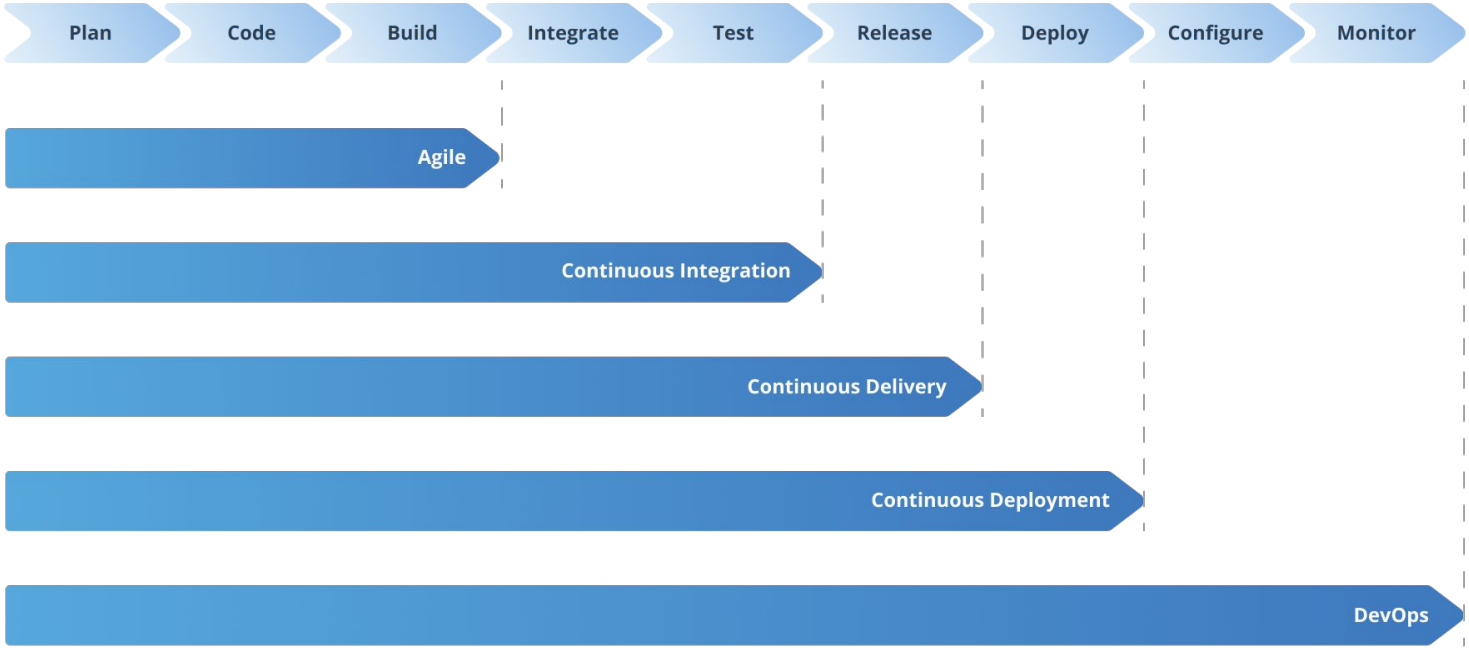
- fetching dependencies from repos
- building codes
- running automated tests
- running code analytics
- notification to developers about possible results
- packaging the build

Continuous Delivery and Deployment

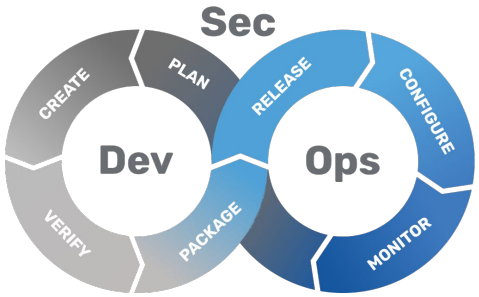
Continuous Delivery is used for releasing of the package that is generated in CI phase, to each **non-production** environment. This releasing work can be scheduled or automated. Quality assurance (QA) tests are run in this phase.

Continuous Deployment is the automated **production** environment releasing. It should have automated user acceptance tests (UATs) as well.

CI/CD Flow


Plan
Code
Build
Integrate
Test
Release
Deploy
Configure
Monitor
Agile
Continuous Integration
Continuous Delivery
Continuous Deployment
DevOps

Continuous Integration



Transformation

Opsta Education

Skooldio

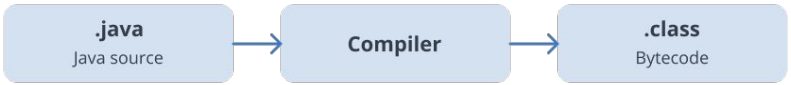
Dependencies Manager

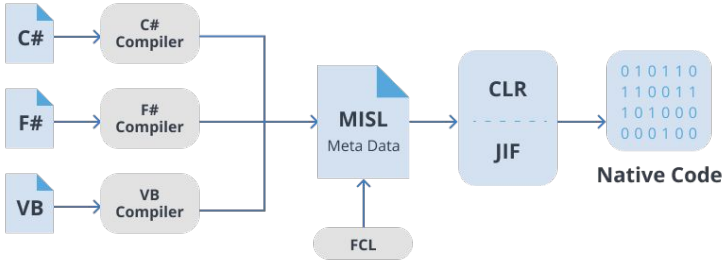
Build | Integrate | Test | Release | Deploy

Build

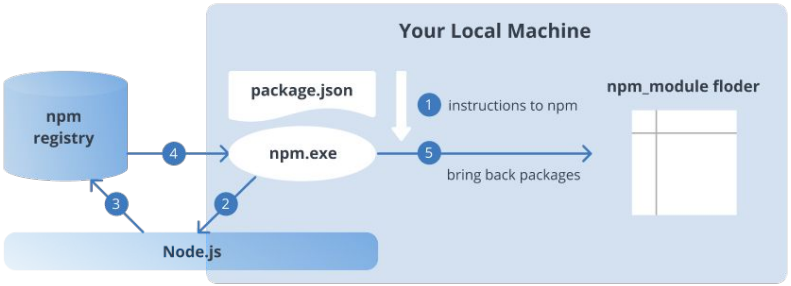
Build | Integrate | Test | Release | Deploy

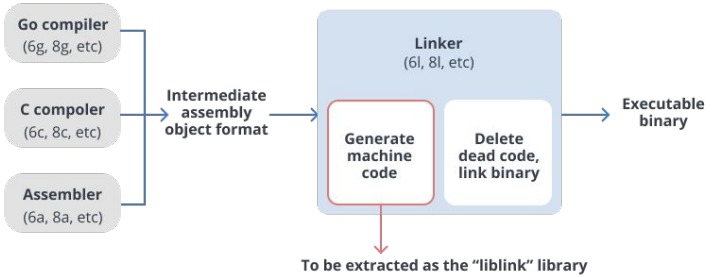
At compile time





Simplified npm work flow





Skooldio Opsta Education

Type of Artifacts

Build | **Integrate** | Test | Release | Deploy

docker

Gradle

nuget

npm

debian

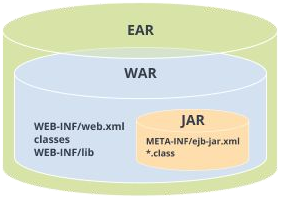
rpm

<xml />

HTML

GO

sbt



Skooldio Opsta Education

Artifacts Server

Build | Integrate | Test | Release | Deploy

Type of Testing

Build → Integrate → **Test** → Release → Deploy

- **Unit Test**
- **Integration Test**
- **Functional Test**
- **Performance Test**
- **Acceptance Test**
- **UI Test**

Test Automation Tools (1)

Build | Integrate | **Test** | Release | Deploy

Unit Test

Browser Simulation Test

Skooldio Opsta Education

Test Automation Tools (2)

Build | Integrate | **Test** | Release | Deploy

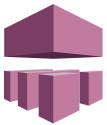
Behaviour Driven Development (BDD)

cucumber

Jasmine

appium

Cloud-Based Test



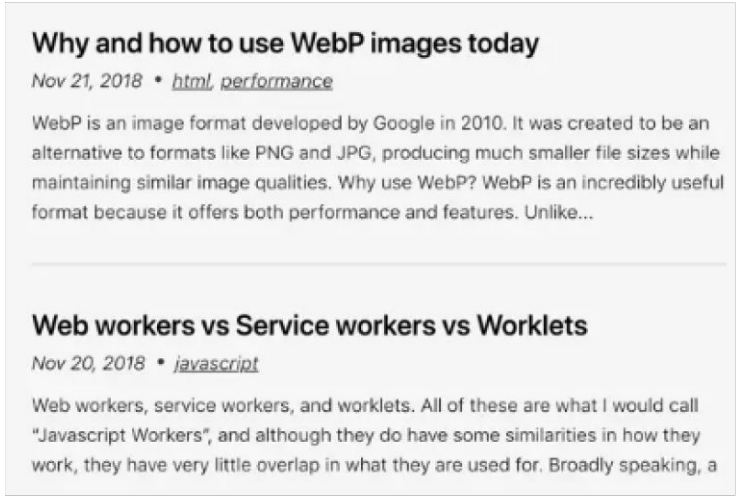
SAUCELABS

Firebase Test Lab

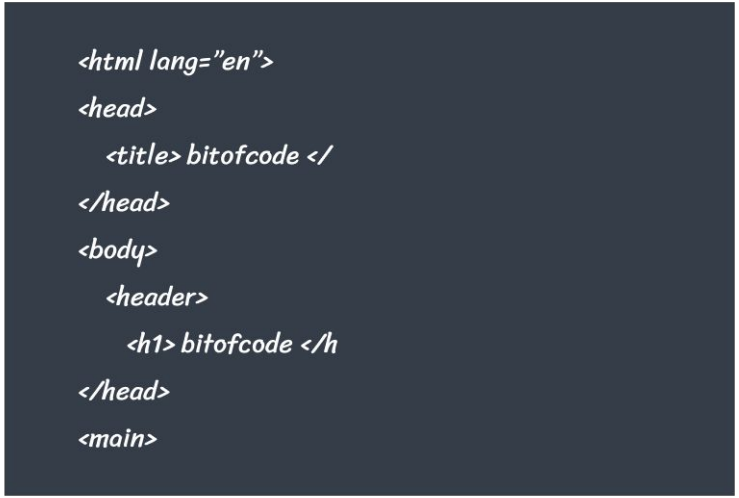
App Center

BrowserStack

Headless Browser



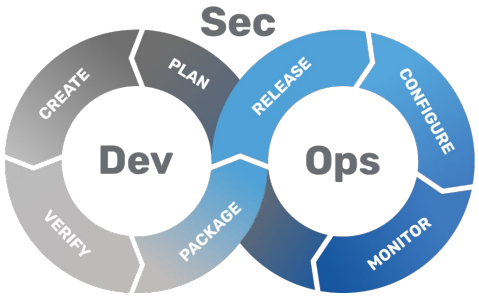
Headful



Headless

Skooldio Opsta Education

Continuous Delivery



Transformation

Opsta Education

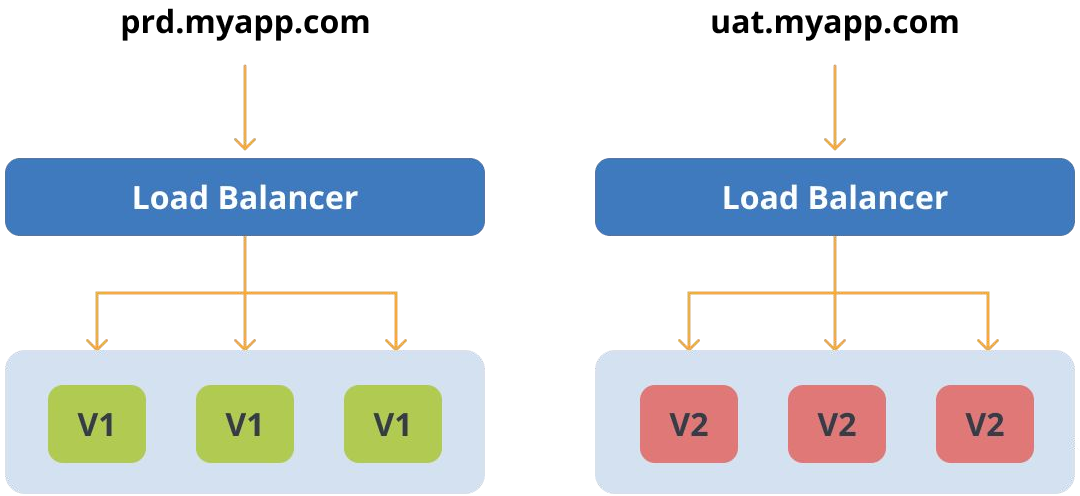
Skooldio

Release Strategies

Build | Integrate | Test | **Release** | Deploy

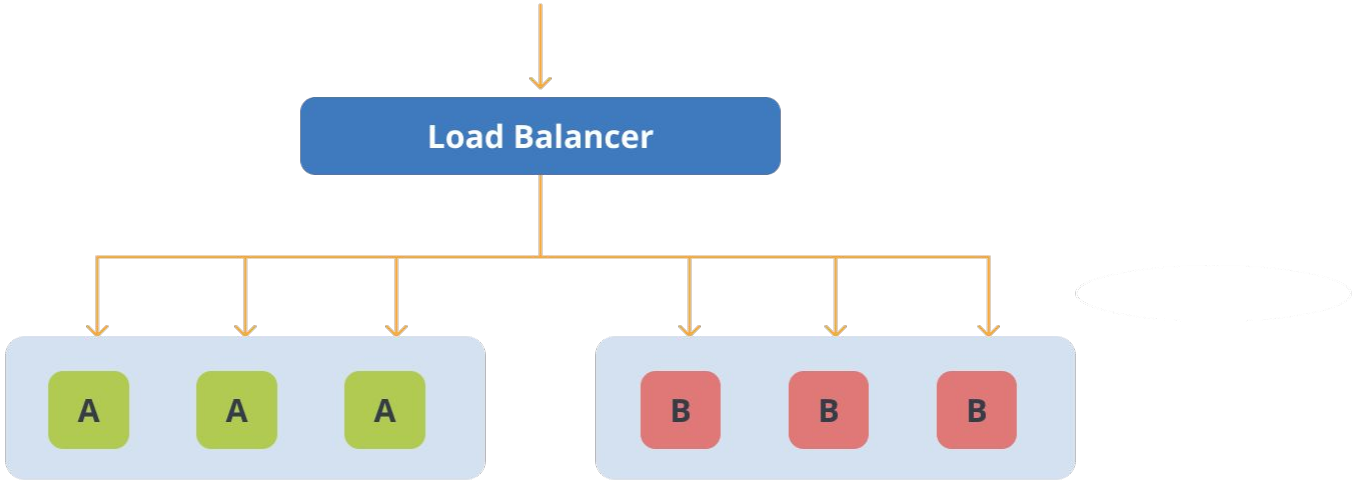
- Environment Separation
- User Targeted
- Feature Toggles or Feature Flag

Environment Separation

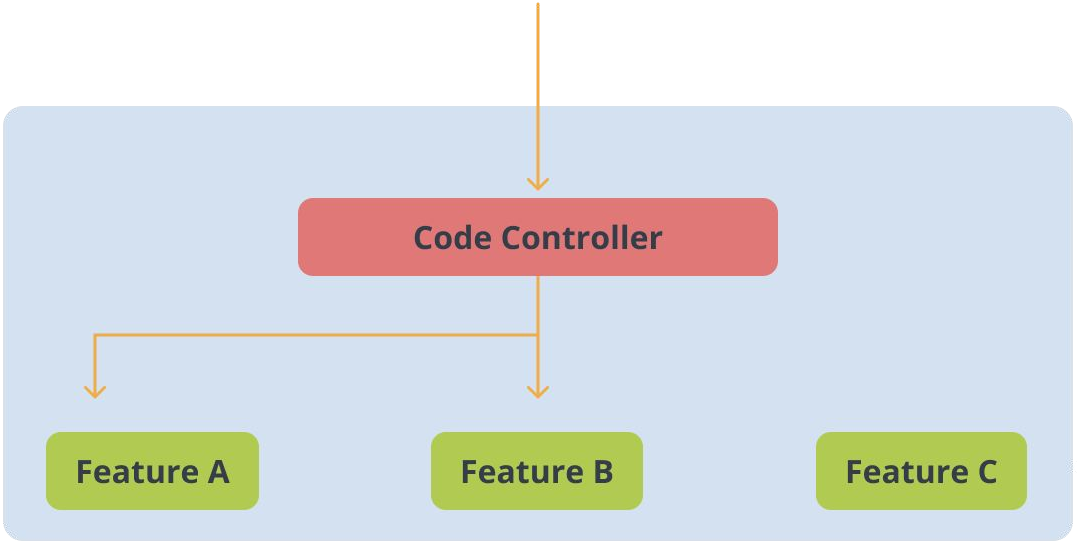


User Targeted

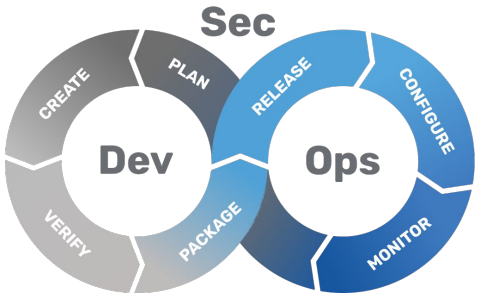
- Sticky Session
- User Agent
- Source IP Address
- Header
- Cookie
- User Specific



Feature Toggle or Feature Flag



Continuous Deployment



Transformation

Opsta Education

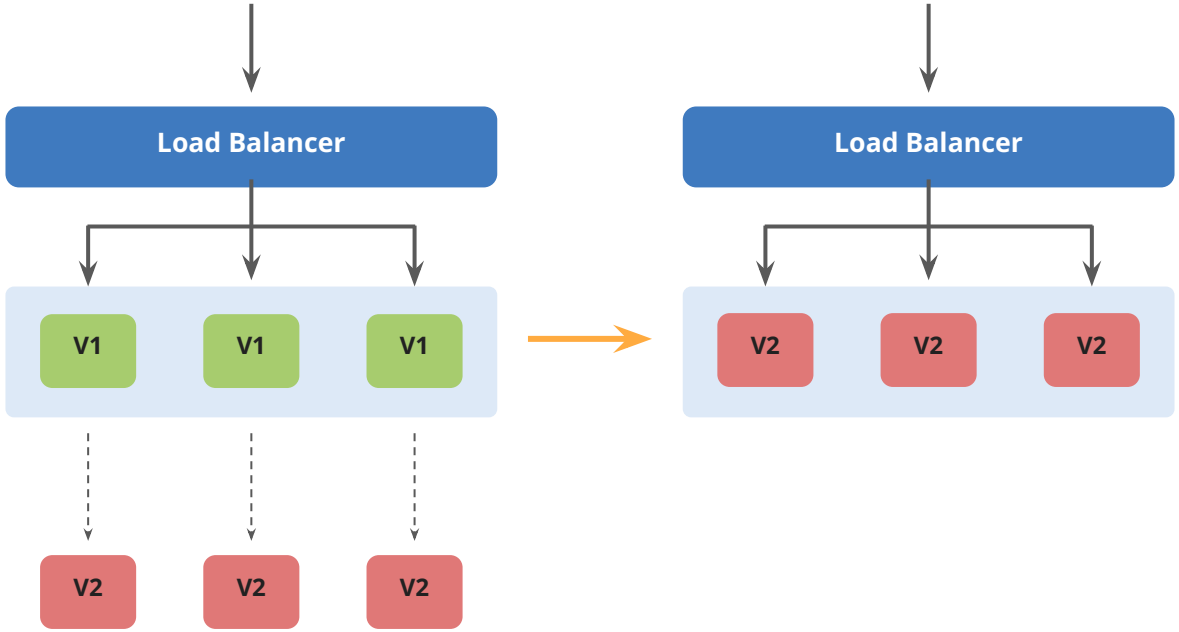
Skooldio

Deployment Strategies

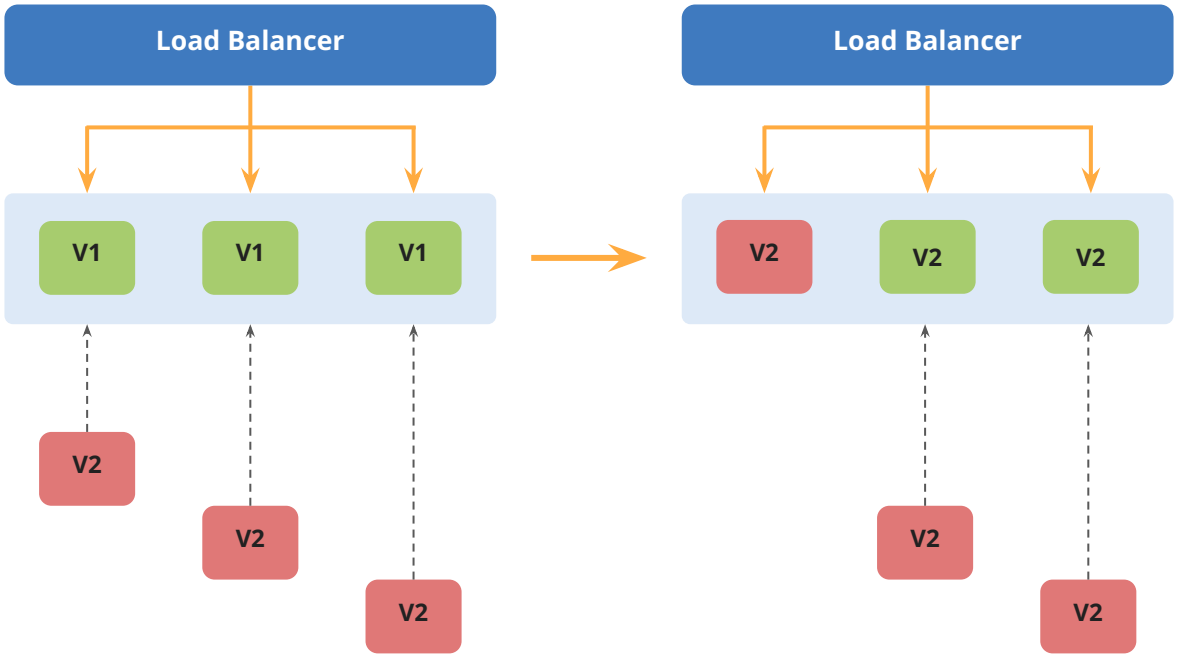
Build | Integrate | Test | Release | **Deploy**

- Recreate / Replace Deployment
- Rolling Deployment
- Blue-Green / Red-Black Deployment
- Canary Deployment

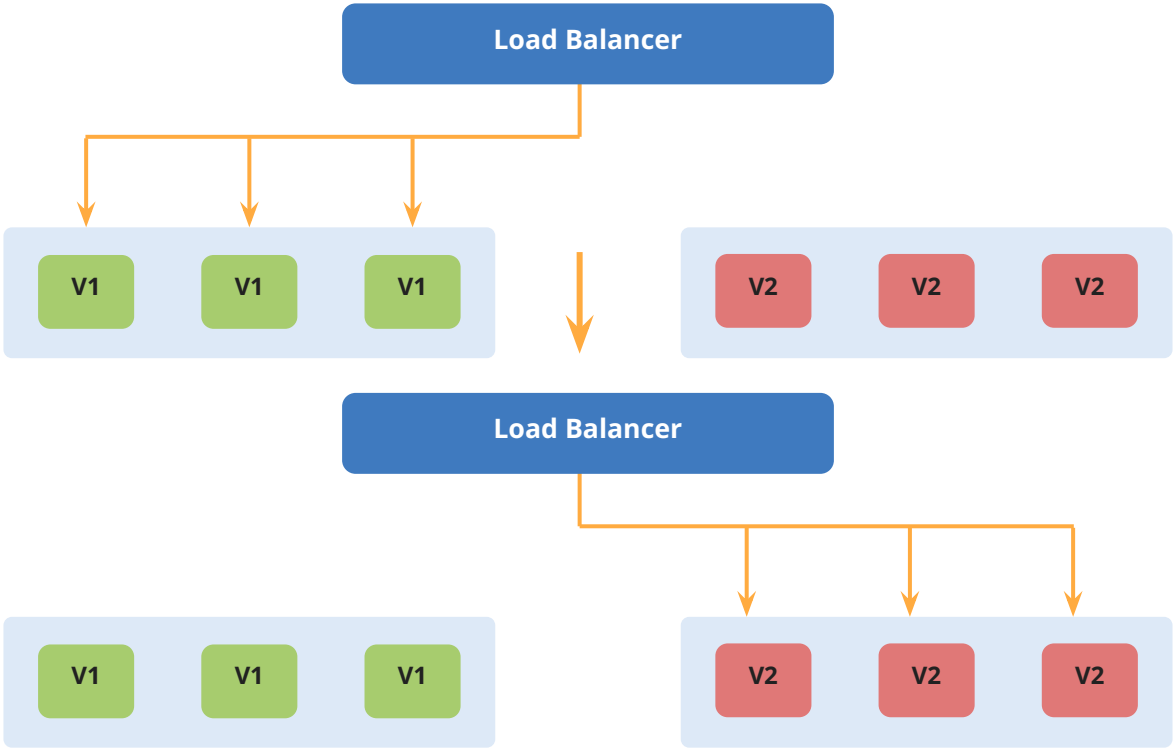
Recreate / Replace Deployment



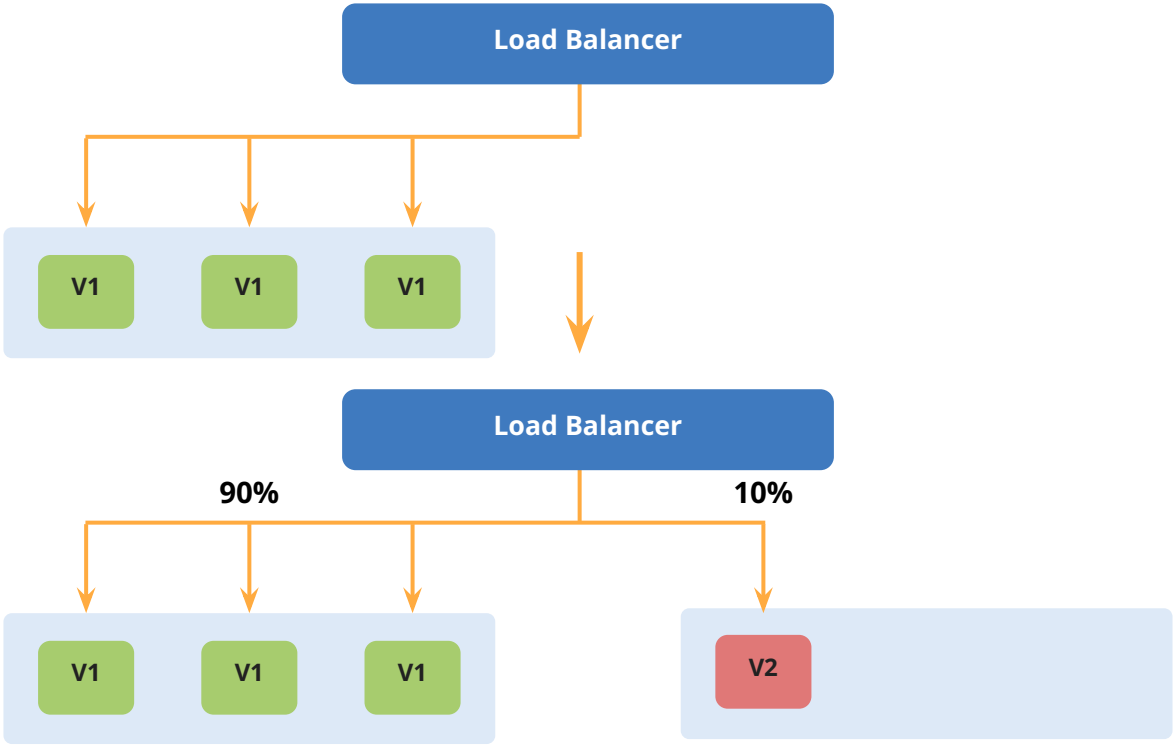
Rolling Deployment



Blue-Green / Red-Black Deployment



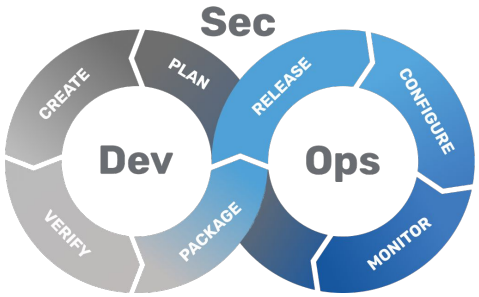
Canary Deployment



Compare Deployment Strategies

Strategy	Zero Downtime	Targeted Users	Cost	Rollback Time	Impact on User	Setup Complexity
Recreate	✗	✗	★	★★★	★★★	★
Rolling	✓	✗	★	★★★	★	★★
Blue-Green	✓	✗	★★★	★	★	★★★
Canary	✓	✓	★★	★★	★	★★★

CI/CD Pipeline and Tools



Transformation

Jenkins

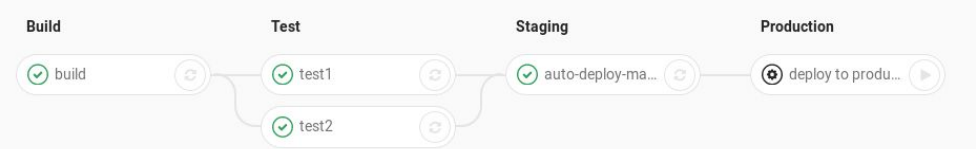
Opsta Education

Skooldio

Popular CI/CD Tools

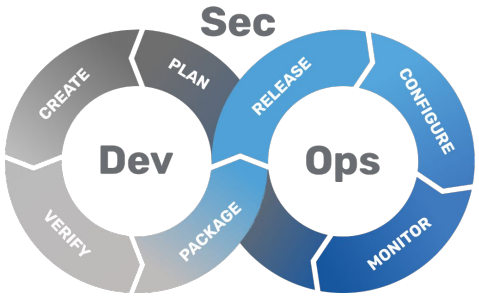
On-Premise Open Source	      
On-Premise Commercial	  
On-Cloud	      

Pipeline as Code




```yaml
test:
  script:
  - apt-get update -qy
  - apt-get install -y nodejs
  - bundle install --path /cache
  - bundle exec rake db:create RAILS_ENV=test
  - bundle exec rake test

staging:
  type: deploy
  script:
  - gem install dpl
  - dpl --provider=heroku --app=gitlab-ci-ruby-test-staging --api-key=$HEROKU_STAGING_API_KEY
  only:
  - master

production:
  type: deploy
  script:
  - gem install dpl
  - dpl --provider=heroku --app=gitlab-ci-ruby-test-prod --api-key=$HEROKU_PRODUCTION_API_KEY
  only:
  - tags
```

Introduction to GitLab CI/CD



Transformation

CI/CD

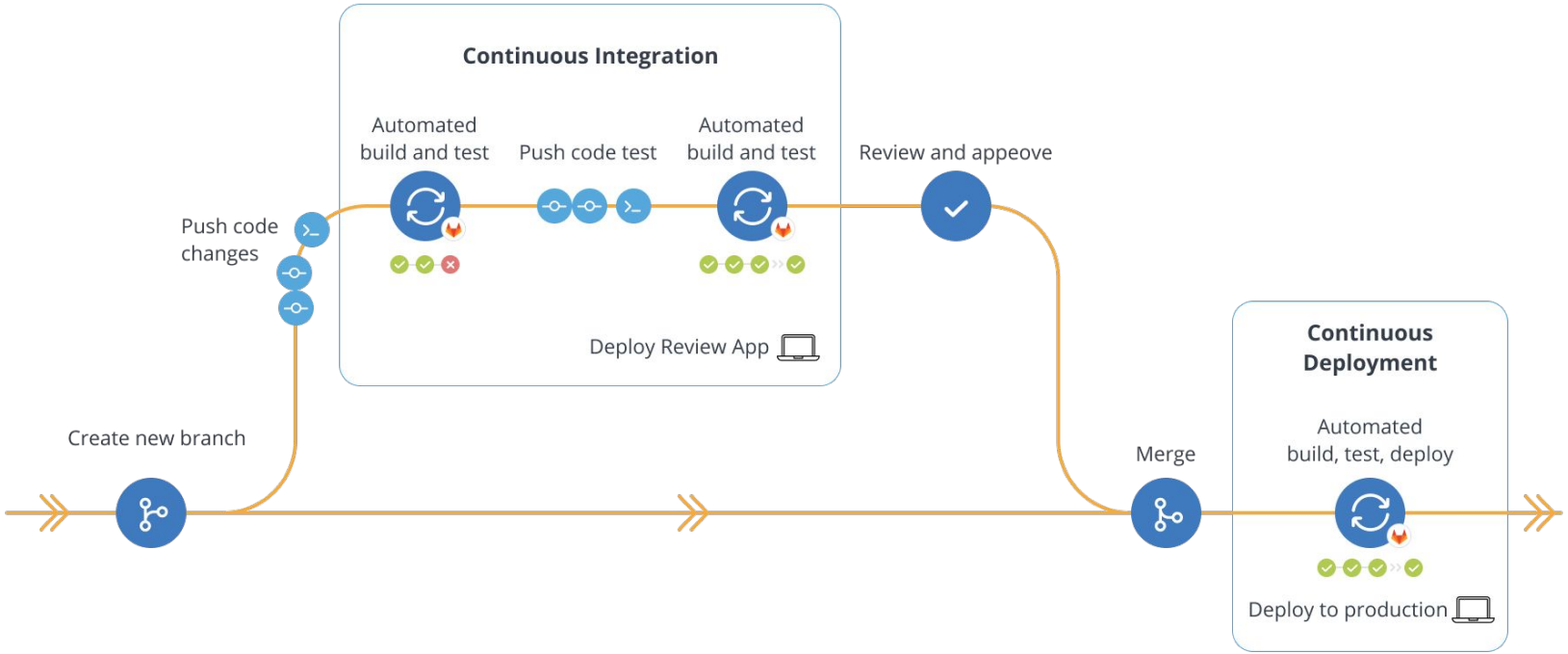
Opsta Education

Skooldio

GitLab CI/CD

- GitLab CI/CD is a tool for software development using the continuous methodologies
- GitLab CI/CD is out-of-the-box with GitLab
  - It can check the CI/CD Pipeline via GitLab UI
  - Runner is integrate with GitLab
- GitLab CI/CD use YAML syntax
  - Easy to read and understand
  - Easier to maintain

GitLab CI/CD Workflow



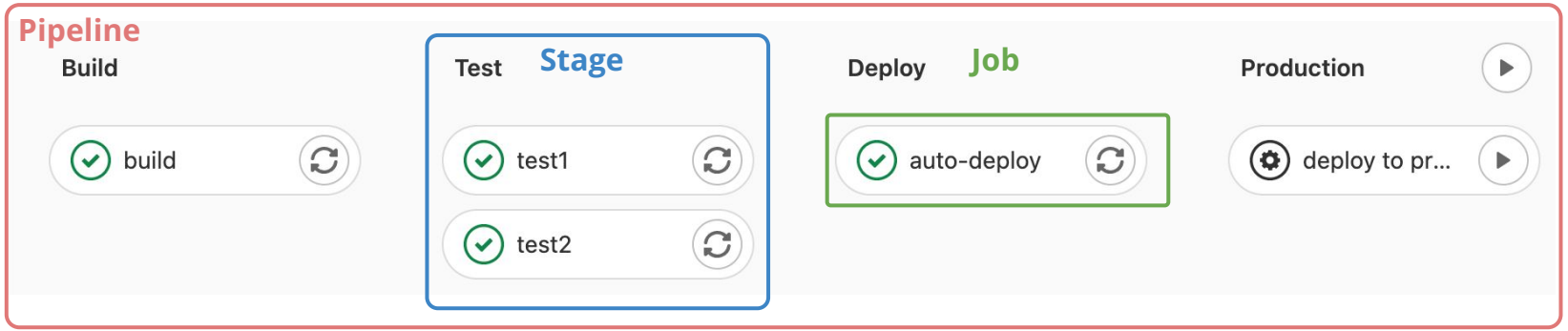
Skooldio Opsta Education

GitLab CI/CD Concepts

- Pipelines, Jobs, and Stages
- CI/CD variables
- Cache and Artifacts
- GitLab Runner

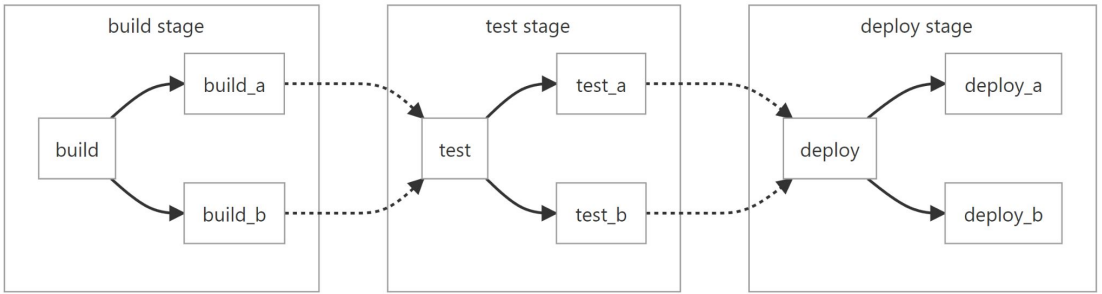
GitLab CI/CD Pipeline

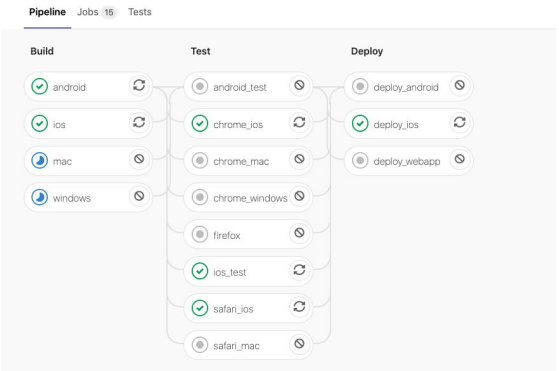
- **Pipelines** are the top-level component of CI/CD
- **Jobs**, which define *what* to do. For example, jobs that compile or test code.
- **Stages**, which define *when* to run the jobs. For example, stages that run tests after stages that compile the code.
- Jobs are executed by **Runners**. Multiple jobs in the same stage are executed in parallel, if there are enough concurrent runners.



GitLab CI/CD Pipeline Types

- **Basic**: Good for straightforward projects where all the configuration is in one easy to find place.
- **Directed Acyclic Graph (DAG)**: Good for large, complex projects that need efficient execution.
- **Child/Parent Pipelines**: Good for monorepos and projects with lots of independently defined components.





Skooldio Opsta Education

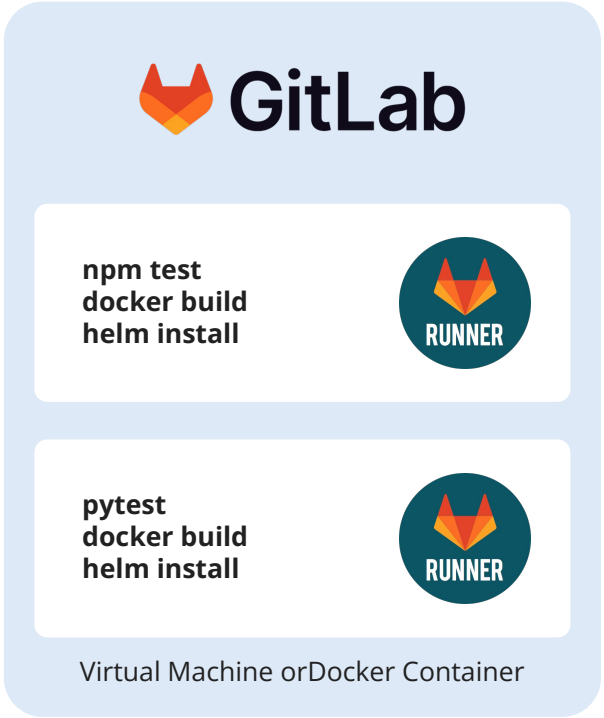
GitLab CI/CD Runner

- **GitLab Runner** is an application that works with GitLab CI/CD to run jobs in a pipeline.
- GitLab Runner is open-source and written in **Go**. It can be run as a single binary; no language-specific requirements are needed.
- GitLab Runner can also run inside a **Docker** container or be deployed into a **Kubernetes** cluster.
- There are 3 types of GitLab Runner
    - **Shared runners** are for use by all projects
    - **Group runners** are for all projects and subgroups in a group
    - **Specific runners** are for individual projects

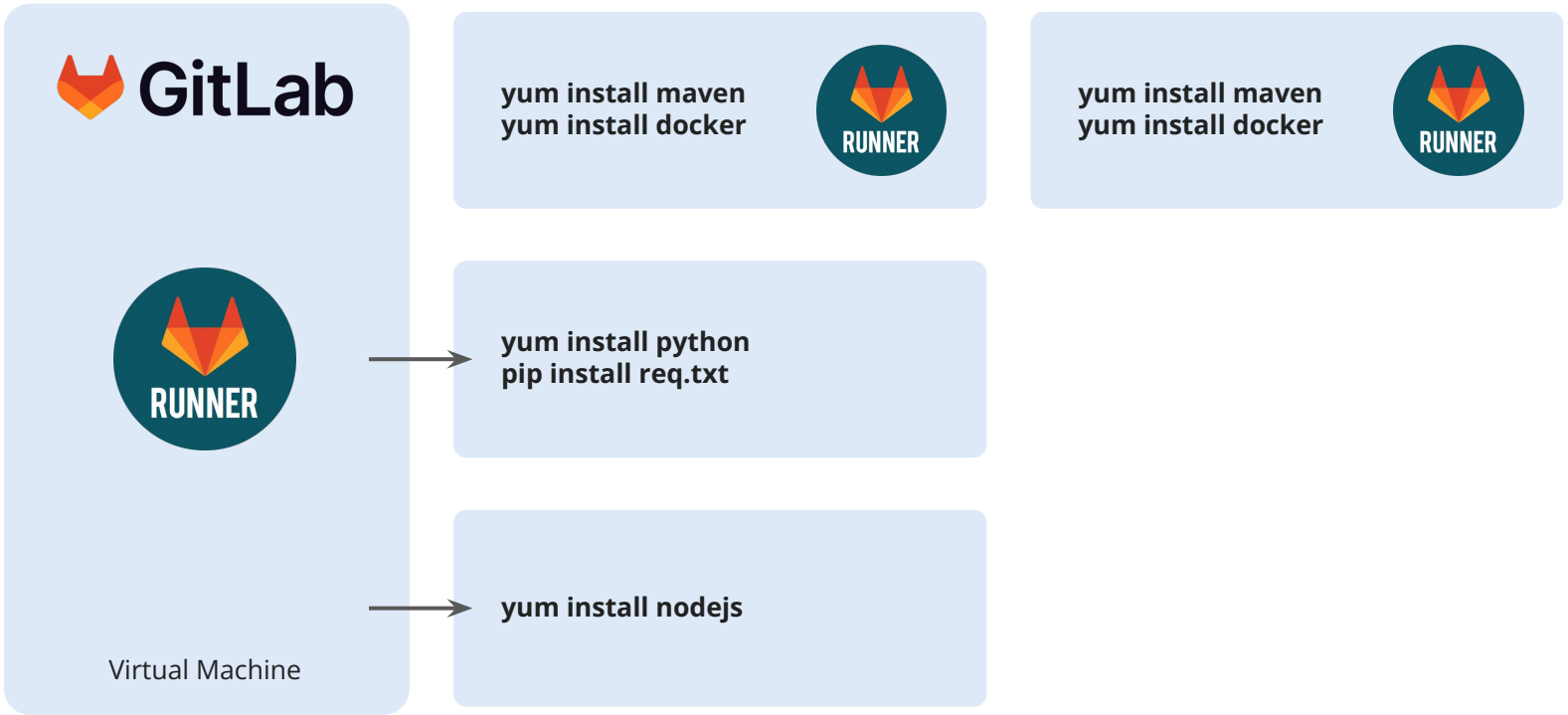
GitLab CI/CD Runner Executors

- **Shell**
  - The simplest executor
  - Required dependencies to be installed manually
- **SSH**
  - The least supported among all executors
  - Connect to an external server and runs the builds
- **Docker**
  - To run clean build environment, with easy dependency management
- **Kubernetes**
  - Use an existing Kubernetes cluster for your builds
  - The executor will call the Kubernetes cluster API and create a new Pod

GitLab CI/CD Runner Shell Executor



GitLab CI/CD Runner Shell or SSH Executor



GitLab CI/CD Runner Docker Executor

node:6(stage test)
npm build
npm test

Docker

node:6(stage build)
npm build
npm install
push artifacts

Virtual Machine

RUNNER

node:6

node:11

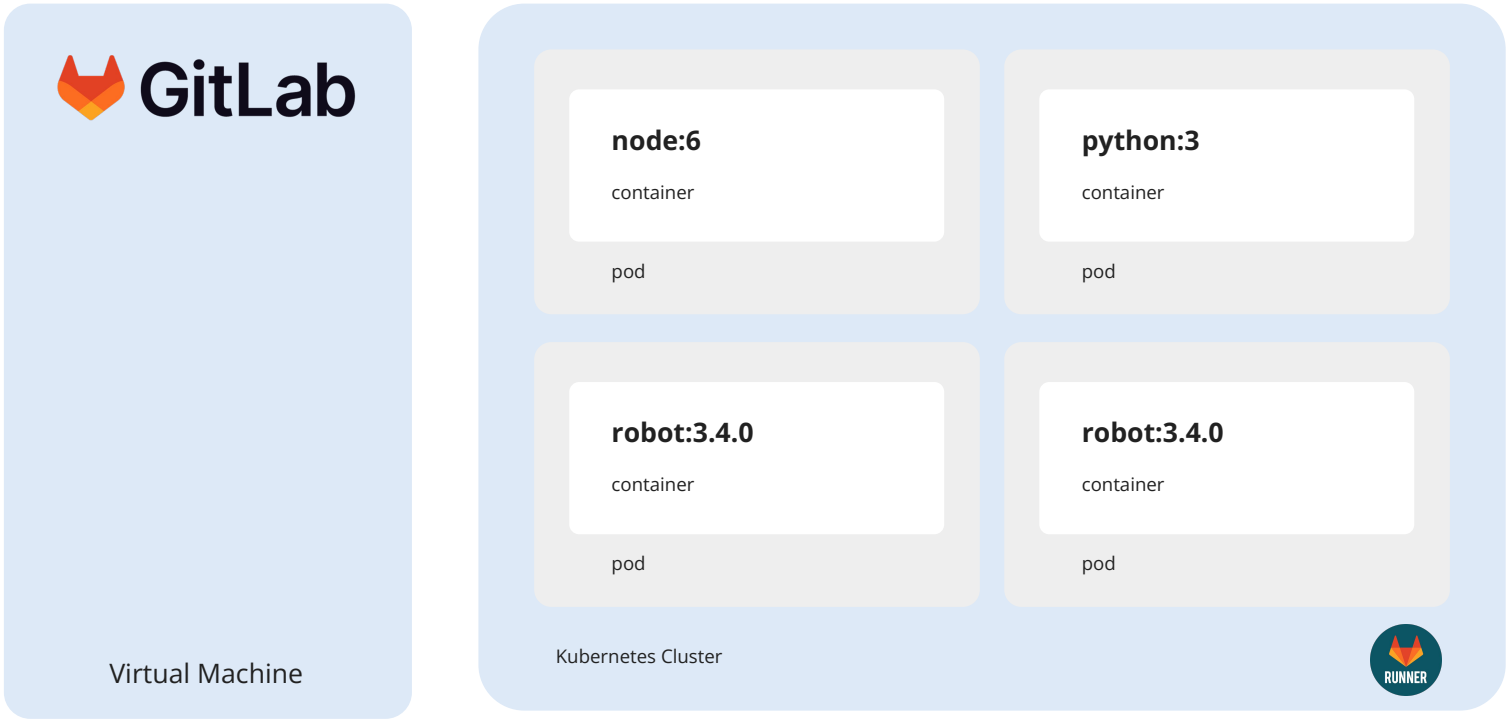
node:12

node:6-robot

docker:dind

Skooldio Opsta Education

GitLab CI/CD Runner Kubernetes Executor



GitLab CI/CD Runner Executors Summary

Executor	SSH	Shell	Docker	Kubernetes
Clean build environment for every build	✗	✗	✓	✓
Reuse previous clone if it exists	✓	✓	✓	✗
Runner file system access protected	✓	✗	✓	✓
Migrate runner machine	✗	✗	✓	✓
Zero-configuration support for concurrent builds	✗	✗	✓	✓
Complicated build environments	✗	✗	✓	✓
Debugging build problems	**easy**	**easy**	**medium**	**medium**

.gitlab-ci.yml

- The **.gitlab-ci.yml** file is a YAML file where you configure specific instructions for GitLab CI/CD.
- In this file, you define:
  - The structure and order of jobs that the runner should execute.
  - The decisions the runner should make when specific conditions are encountered.

.gitlab-ci.yml sample structure

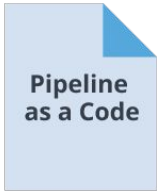
```yaml
stages:
  - build
  - deploy

image: alpine

build:
  stage: build
  script:
    - echo "This job builds something."

deploy:
  stage: deploy
  script:
    - echo "This job deploys something.""
```

GitLab CI/CD Variables

- CI/CD variables are a type of environment variable
  - Control the behavior of jobs and pipelines
  - Store values you want to re-use
  - Avoid hard-coding values in your .gitlab-ci.yml file
- Where we can define variables
  - Predefined CI/CD variables
  - Variables in the .gitlab-ci.yml file
  - Project CI/CD variables
  - Group CI/CD variables
  - Instance CI/CD variables

CI/CD Responsibility

Build → Integrate → Test → Release → Deploy



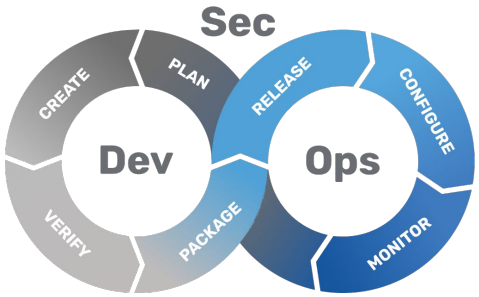
Developer

- Structure
- Build
- Test
- Deploy

Operation

- Review
- Optimize
- Security

GitLab CI/CD Workshop



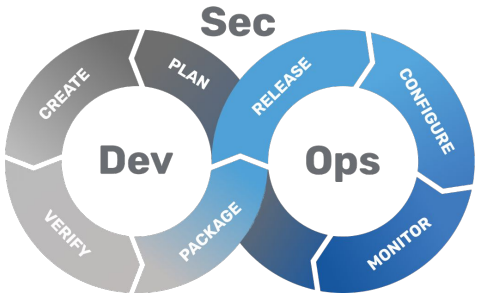
Transformation

CI/CD

Opsta Education

Skooldio

GitLab CI/CD Templates



Transformation

CI/CD

Opsta Education

Skooldio

GitLab CI/CD Templates

Android

Continuous integration and deployment template to test and deploy your Android project.

Use template

Bash

Continuous integration and deployment template to test and deploy your Bash project.

Use template

C++

Continuous integration and deployment template to test and deploy your C++ project.

Use template

Clojure

Continuous integration and deployment template to test and deploy your Clojure project.

Use template

Composer

Continuous integration and deployment template to test and deploy your Composer project.

Use template

Crystal

Continuous integration and deployment template to test and deploy your Crystal project.

Use template

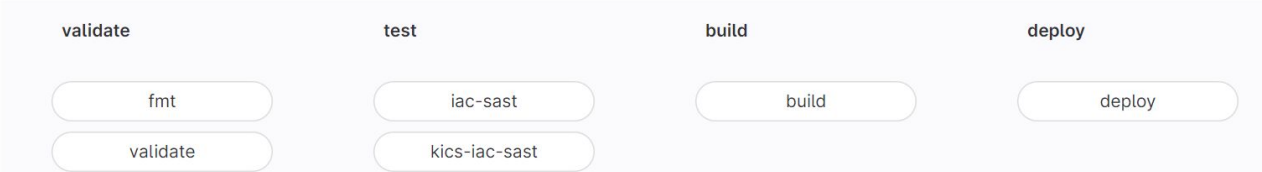
GitLab CI/CD Pipeline Editor

Edit | Visualize | Validate | Full configuration

Browse templates | Help

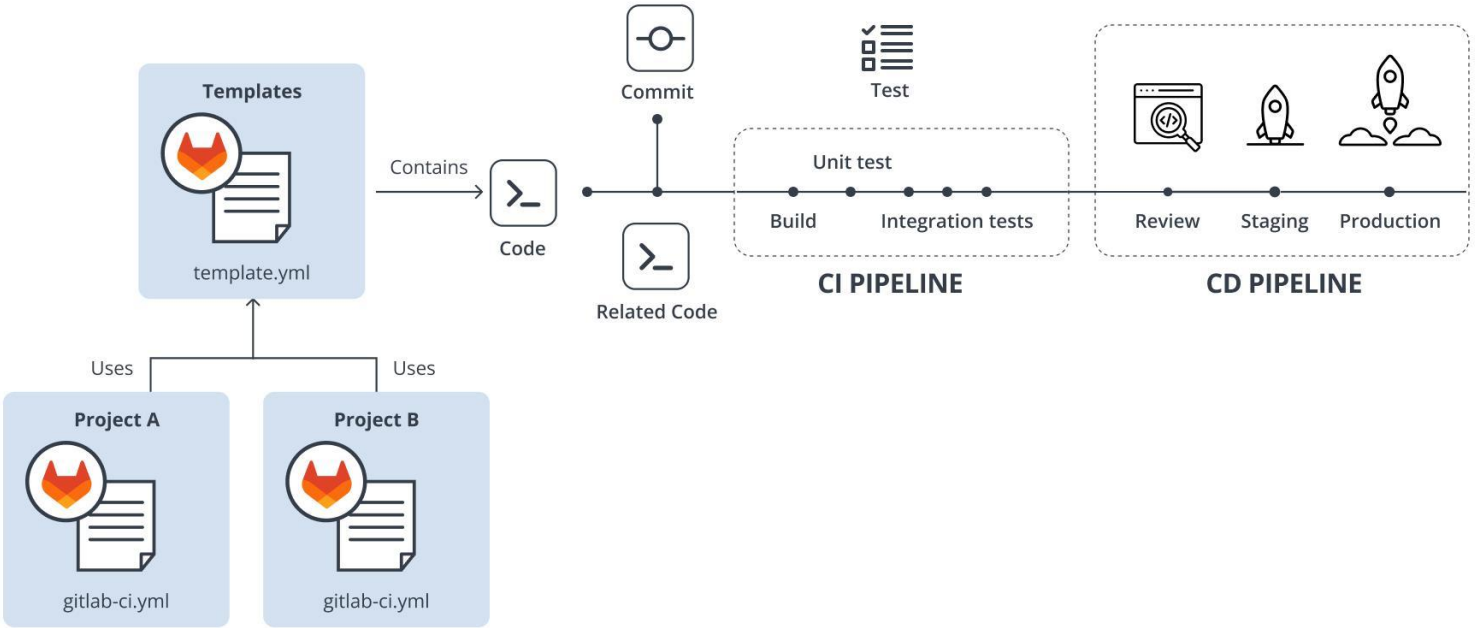
```
1  # This file is a template, and might need e
2  # To contribute improvements to CI/CD templ
3  # https://docs.gitlab.com/ee/development/ci
4  # This specific template is located at:
5  # https://gitlab.com/gitlab-org/gitlab/-/bl
6
7  include:
8    - template: Terraform/Base.gitlab-ci.yml
9    - template: Jobs/SAST-IaC.gitlab-ci.yml
10
11 stages:
12   - validate
13   - test
14   - build
15   - deploy
16   - cleanup
17
18 fmt:
19   extends: .terraform:fmt
20   needs: []
21
22 validate:
23   extends: .terraform:validate
24   needs: []
25
26 build:
```

Edit | Visualize | Validate | Full configuration



Skooldio | Opsta Education

GitLab CI/CD Includes

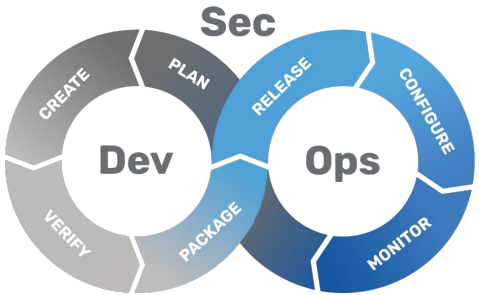


GitLab CI/CD Include and Hidden Jobs

```yaml
# This is a hidden job in template file
# npm-test.yml

variables:
  TYPE: string

.npm-test:
  stage: test
  script:
    - npm test $TYPE
```

```yaml
# This is the usage of the hidden job with include
# .gitlab-ci.yml

include:
  - template: npm-test.yml

test-string:
  extends: .npm-test

test-int:
  extends: .npm-test
  variables:
    TYPE: int
```

GitLab CI/CD Templates Workshop



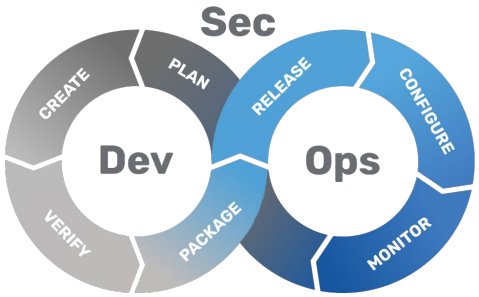
Transformation

CI/CD

Opsta Education

Skooldio

DevOps & DevSecOps Trends

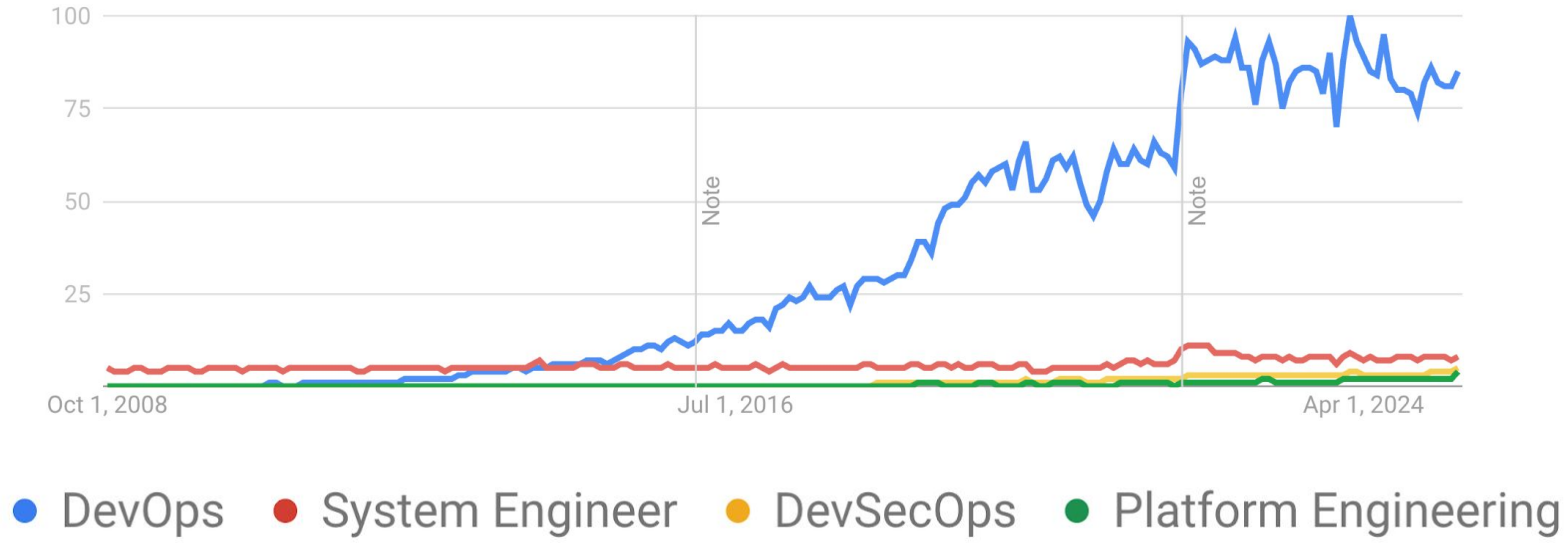


Transformation

Opsta Education

Skooldio

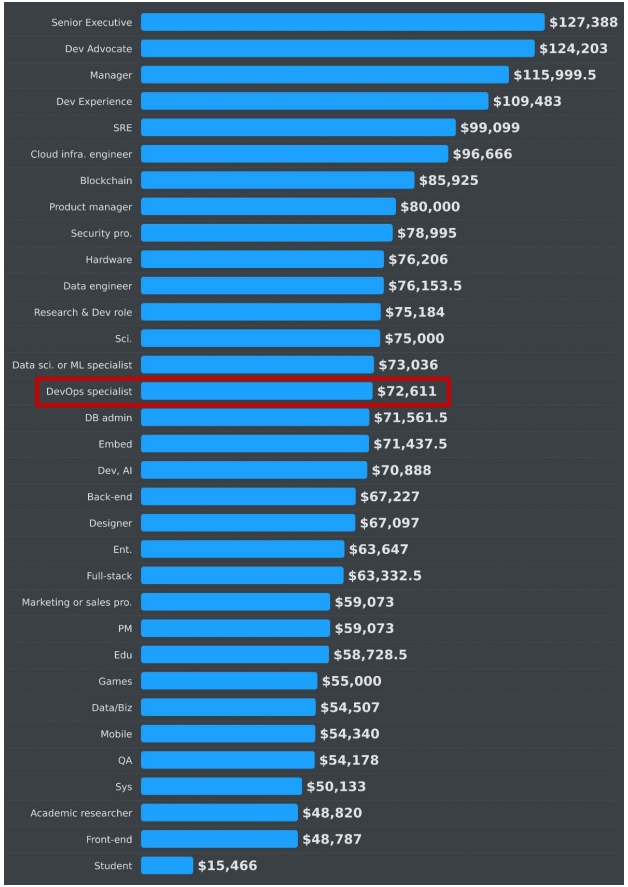
DevOps Trend



https://trends.google.co.th/trends/explore?date=2008-10-13%202024-07-29&q=DevOps,System%20Engineer,DevSecOps,Platform%20Engineering

Skooldio Opsta Education

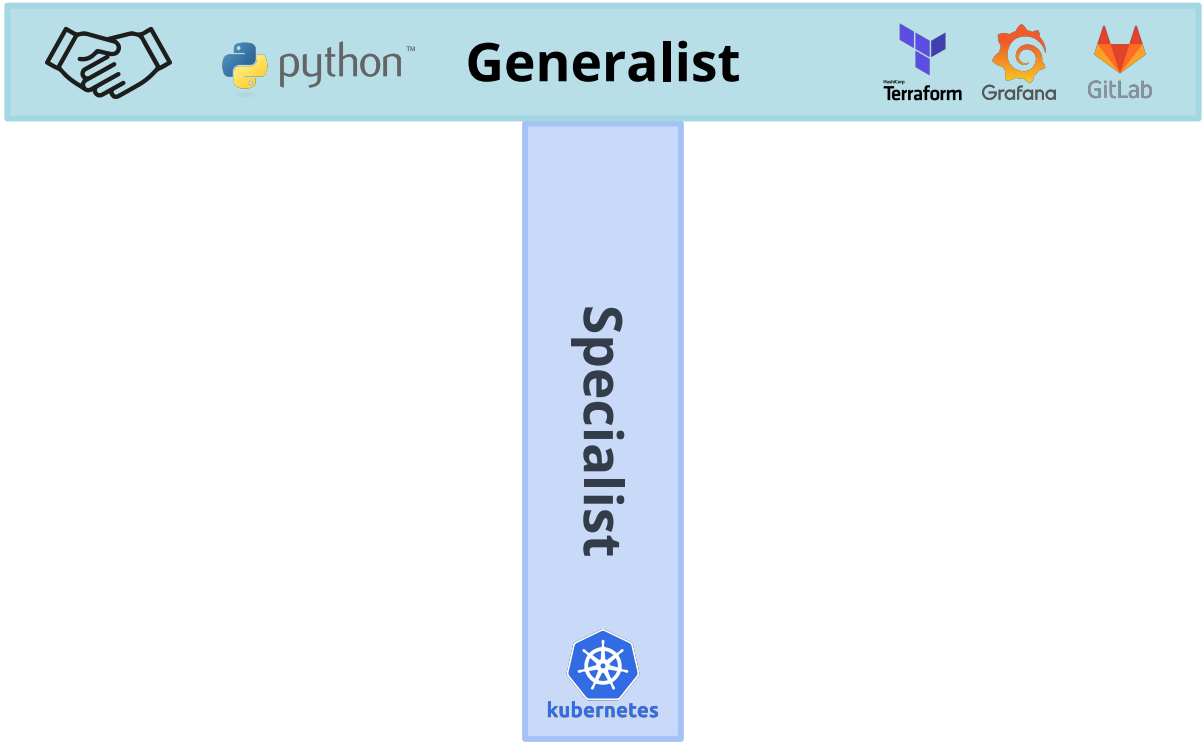
Salary by developer type



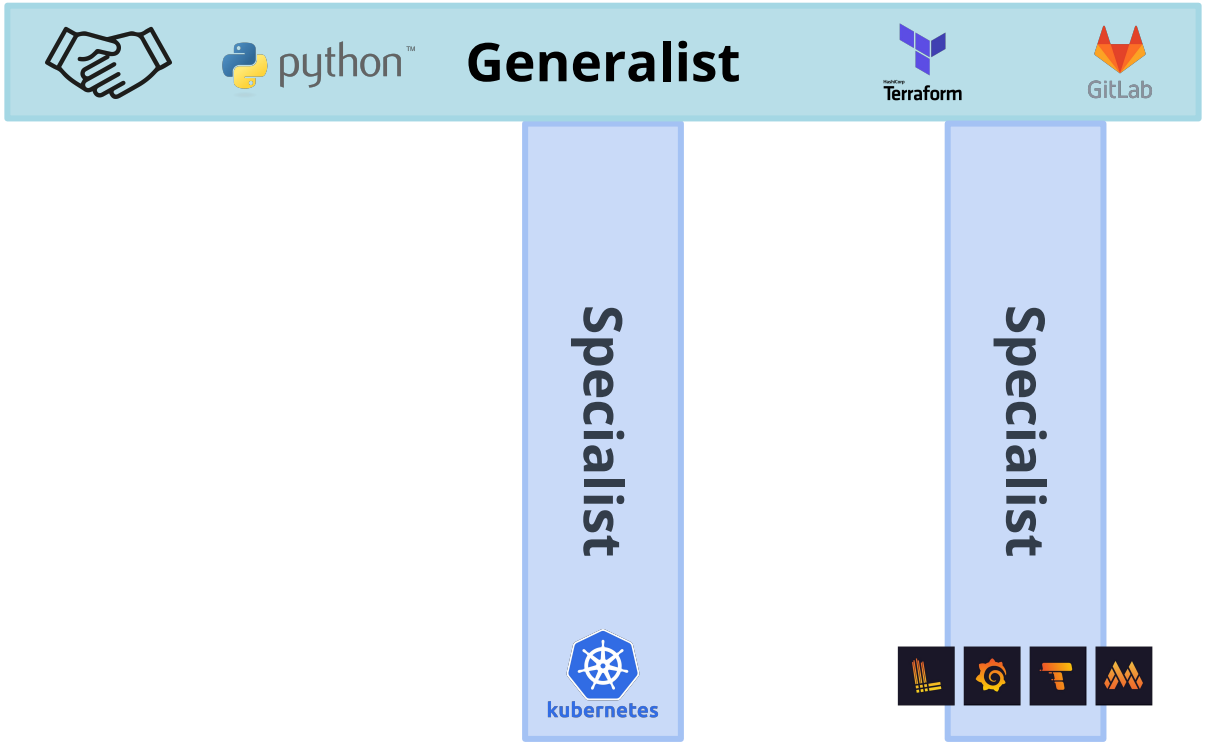
https://survey.stackoverflow.co/2024/work#salary

Skooldio Opsta Education

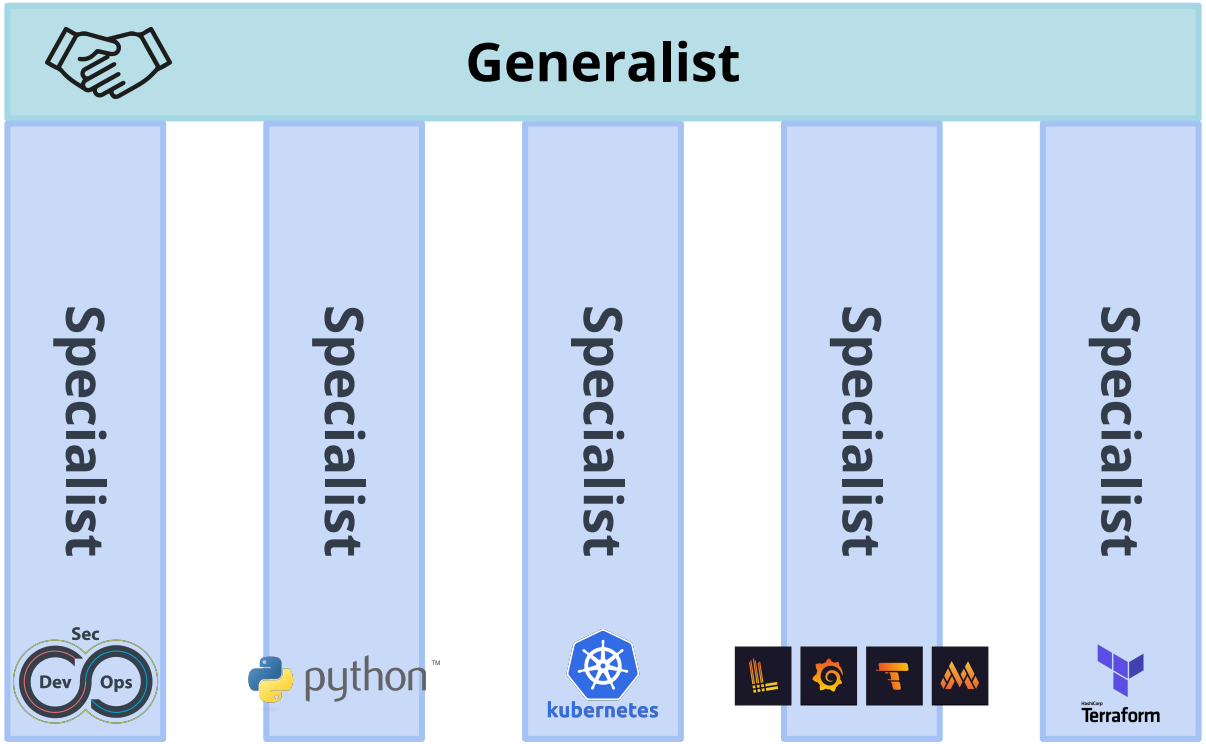
T-Shaped Skills



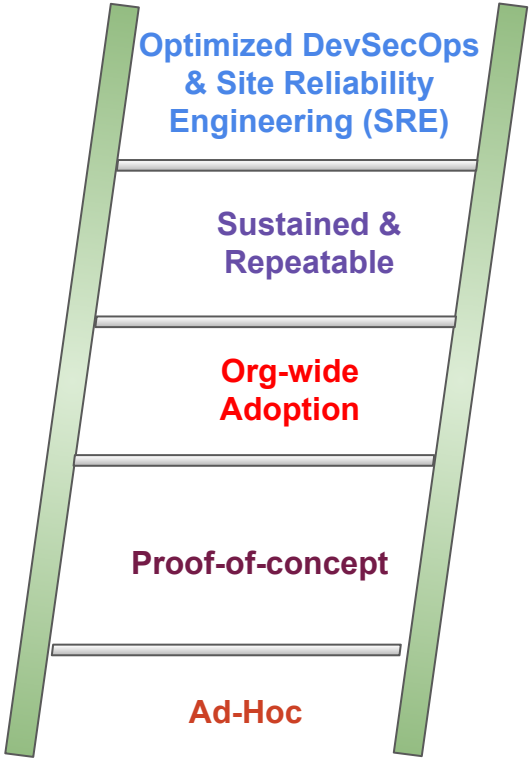
Pi-Shaped Skills



Comb-Shaped Skills



DevSecOps Maturity Levels Explaination



- Lessons learnt using continuous feedback is incorporated back to improve the DevSecOps implementation
- Appropriate mix of tools & frameworks used for optimized outcomes
- Development processes updated to optimize DevSecOps outcomes

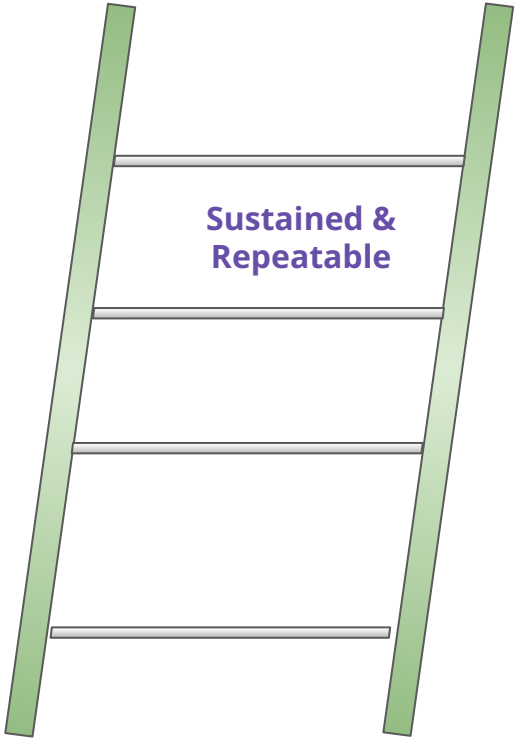
- Sustained DevSecOps implementation across different teams
- DevSecOps Governance
- Tracking & reporting

- DevSecOps rolled out for all the teams
- Mentoring sessions across the organization
- DevSecOps implementation governed & reported

- DevSecOps implementation planned
- Team mentored
- Tools & processes chosen
- 3-4 teams chosen
- Implementation done; Lessons learnt

- Team across organization doing ad-hoc implementation
- Different tools & framework & processes used across difference team

Evolution of DevSecOps Culture



- Sustained DevSecOps implementation across different teams
- DevSecOps Governance
- Tracking & reporting

This is Platform

with Platform Engineering Team

Skooldio Opsta Education

What is Platform?

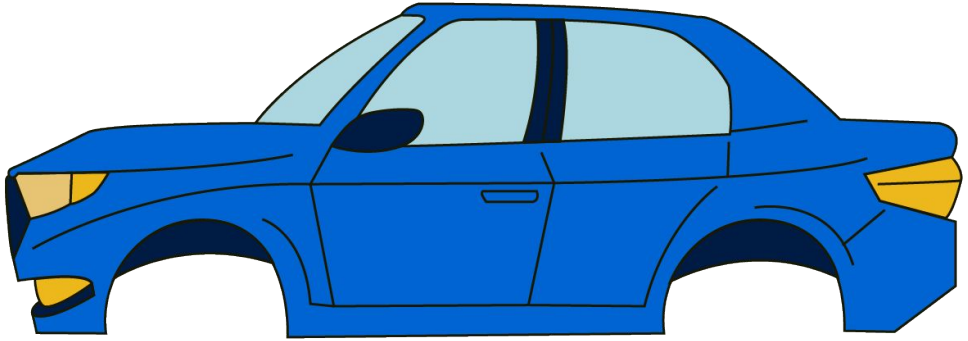
plat·form

/ˈplatˌfôrm/

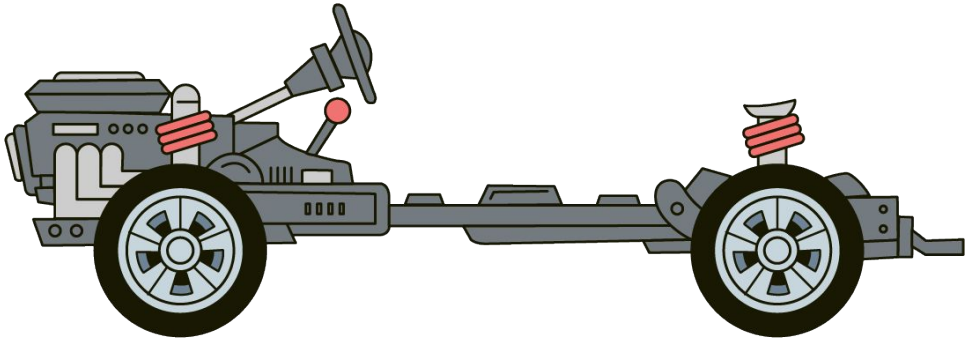
noun

1. a raised level surface on which people or things can stand.

Platform Engineering



HAT
Look & Feel



PLATFORM
Engineering

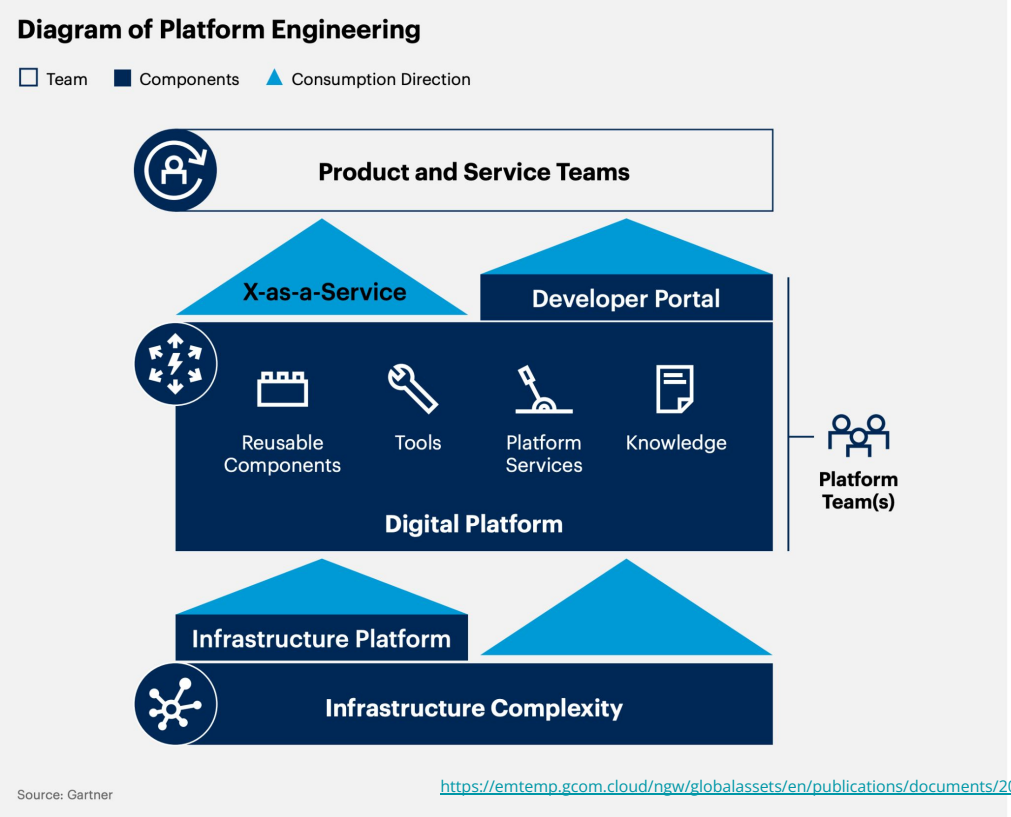
Platform means

Not start from zero

[Gartner] Top Strategic Technology Trends 2023

Choose your business goal ↓	Sustainable Technology	AI TRiSM	Wireless-Value Realization	Industry Cloud Platforms	Digital Immune System	Platform Engineering	Superapps	Adaptive AI	Applied Observability	Metaverse
	Now	0-1 year			1-2 years			2-3 years		
Create secure foundations		●	●		●	●		●		
Maximize value from data		●	●					●	●	
Protect and grow your brand	●	●			●					●
Attract and retain talent	●					●	●			●
Grow revenue	●			●			●			●
Accelerate digital			●	●	●	●			●	

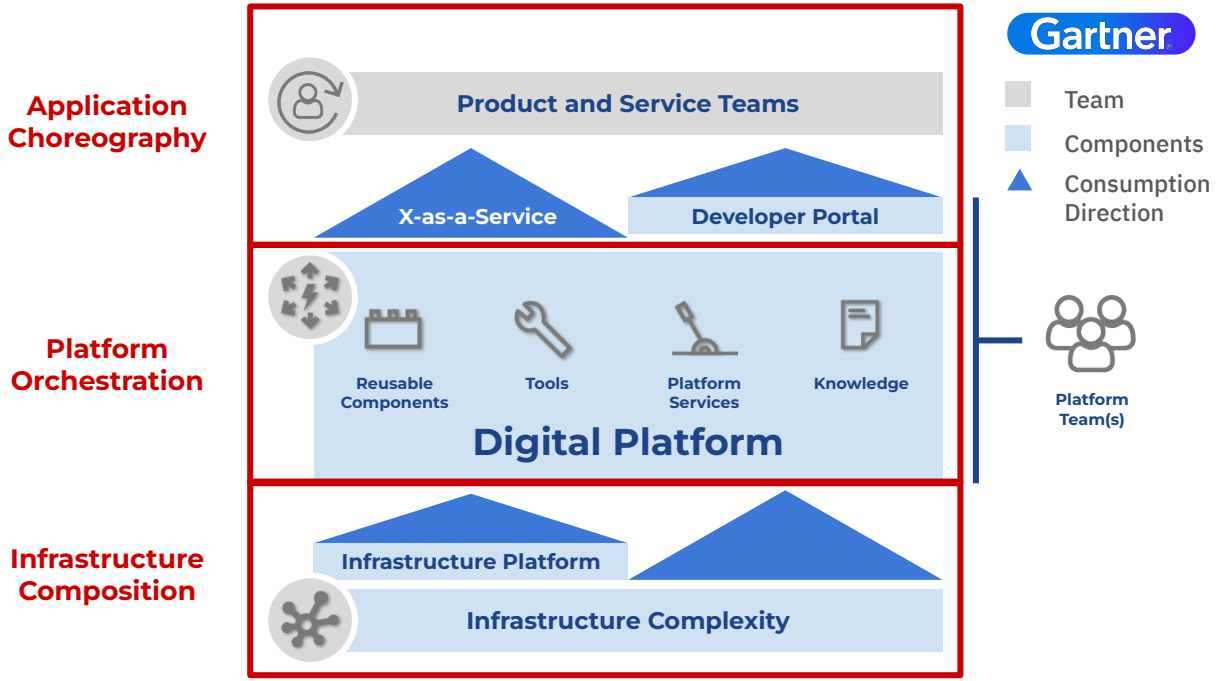
[Gartner] Top Strategic Technology Trends 2024



Source: Gartner

https://emtemp.gcom.cloud/ngw/globalassets/en/publications/documents/2024-gartner-top-strategic-technology-trends-ebook.pdf

Platform Engineering Diagram

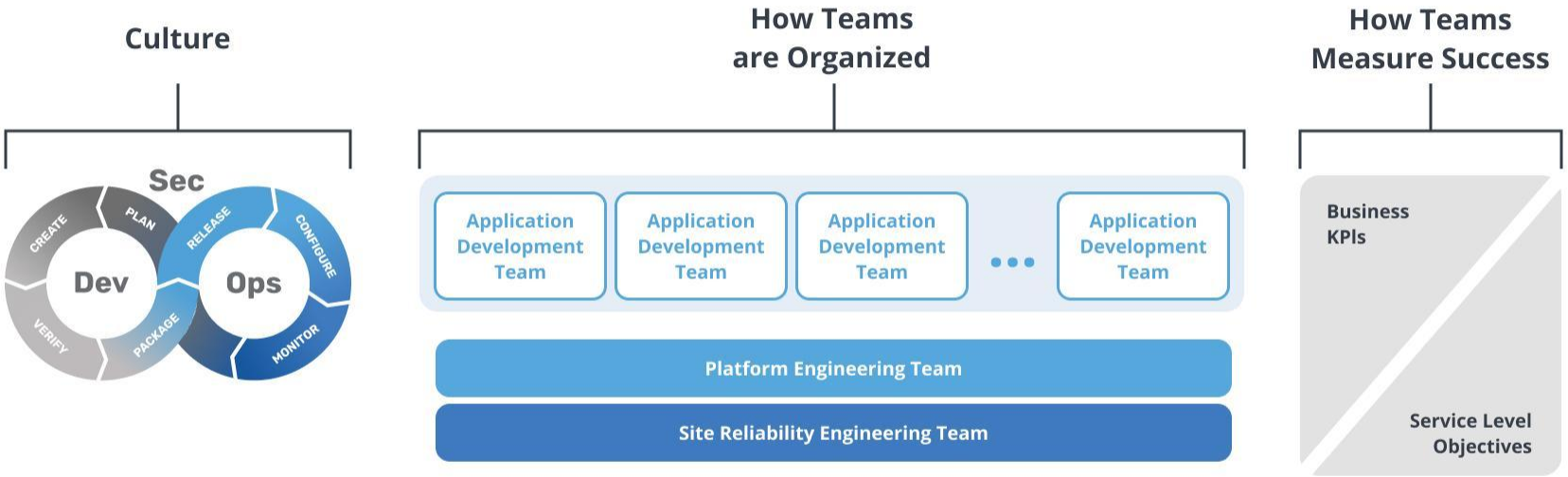


https://www.gartner.com/en/infrastructure-and-it-operations-leaders/topics/platform-engineering

Layer of Platform Engineering

Layer	Why and How?	Who?	What	Example tech
Application Choreography	“Code, ship, run” Developer Control Plane	App developers, full stack engineers, DevOps, SREs	UI (Portals), CLI, Declarative config *Software dev lifecycle*	Backstage, Heroku CLI & Netflix Newt, Score & KubeVela (OAM)
Platform Orchestration	“Design, enable, optimize” Platform Orchestrator	Platform engineers, Engineering enablement, DevEx engineers, SREs	Platform API *Platform lifecycle*	Kratix Promise, Humanitec Resource Definition, Argo/Flux CRDs
Infrastructure Orchestration/ Composition	“plan, build, maintain” Infrastructure Control Plane	Platform engineers, DevOps, Operators, Sysadmins, Infrastructure engineers	laC, Bash scripts, CRDs *Infrastructure lifecycle*	Terraform, Crossplane Managed Resource, Ansible

DevSecOps, SRE, and Platform Engineering



https://www.getambassador.io/resources/rise-of-cloud-native-engineering-organizations/

Internal Developer Platform (IDP)

Developer

Use

IDP

Develop

Platform
Engineering Team

Manage

GitLab Jenkins docker kubernetes ANSIBLE Terraform elastic Prometheus Grafana

Nexus sonarqube amazon web services Google Cloud Azure HUAWEI HUAWEI CLOUD VMware Tanzu

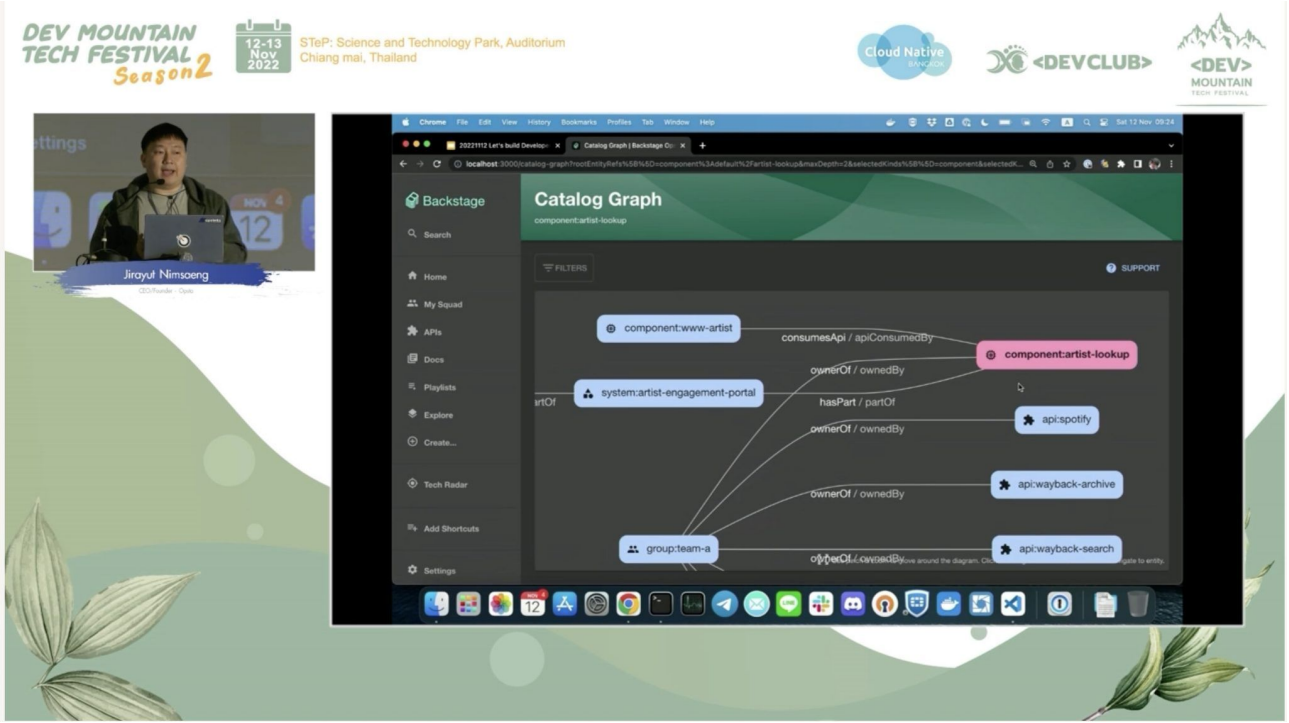
Automation

DevOps | **Infrastructure** | **Operation**

Security

Skooldio Opsta Education

IDP with Backstage

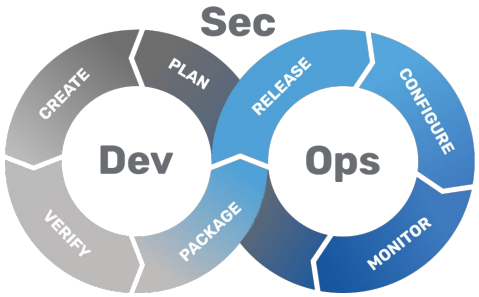


https://youtu.be/u_nLbgWDwsA?t=853

Skooldio Opsta Education

Everyone is a part of Platform Engineering

Benefits

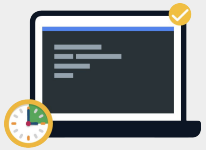


Transformation

Opsta Education

Skooldio

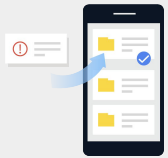
DevOps Research and Assessment (DORA)



Change lead time:
the time it takes for a code commit or change to be successfully deployed to production.

Deployment frequency:
how often application changes are deployed to production.

Change fail rate:
the percentage of deployments that cause failures in production, requiring hotfixes or rollbacks.



Failed deployment recovery time:
the time it takes to recover from a failed deployment.

Measuring DevOps

Software delivery performance metric	Low	Medium	High
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	Between once per month and once every 6 months	Between once per week and once per month	On-demand (multiple deploys per day)
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Between one month and six months	Between one week and one month	Between one day and one week
For Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Between one week and one month	Between one day and one week	Less than one day
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g.,lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	46%-60%	16%-30%	0%-15%

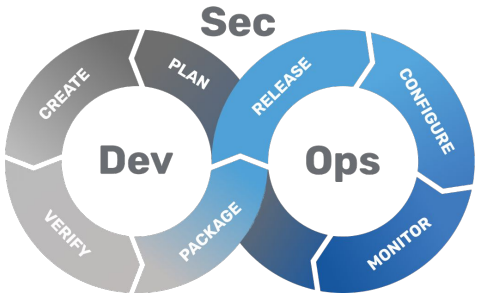
State of DevOps Report 2022

Benefits of DevSecOps

- **Early detection of vulnerabilities** reducing the risk of security breaches
- **Faster time-to-market** reduce the time it takes to bring a product to market by enabling continuous integration and deployment
- **Increased efficiency** by automating security testing and other processes
- **Better compliance** to help organizations meet regulatory compliance requirements by building security into the development process from the start

https://chat.openai.com

The Twelve-Factor App

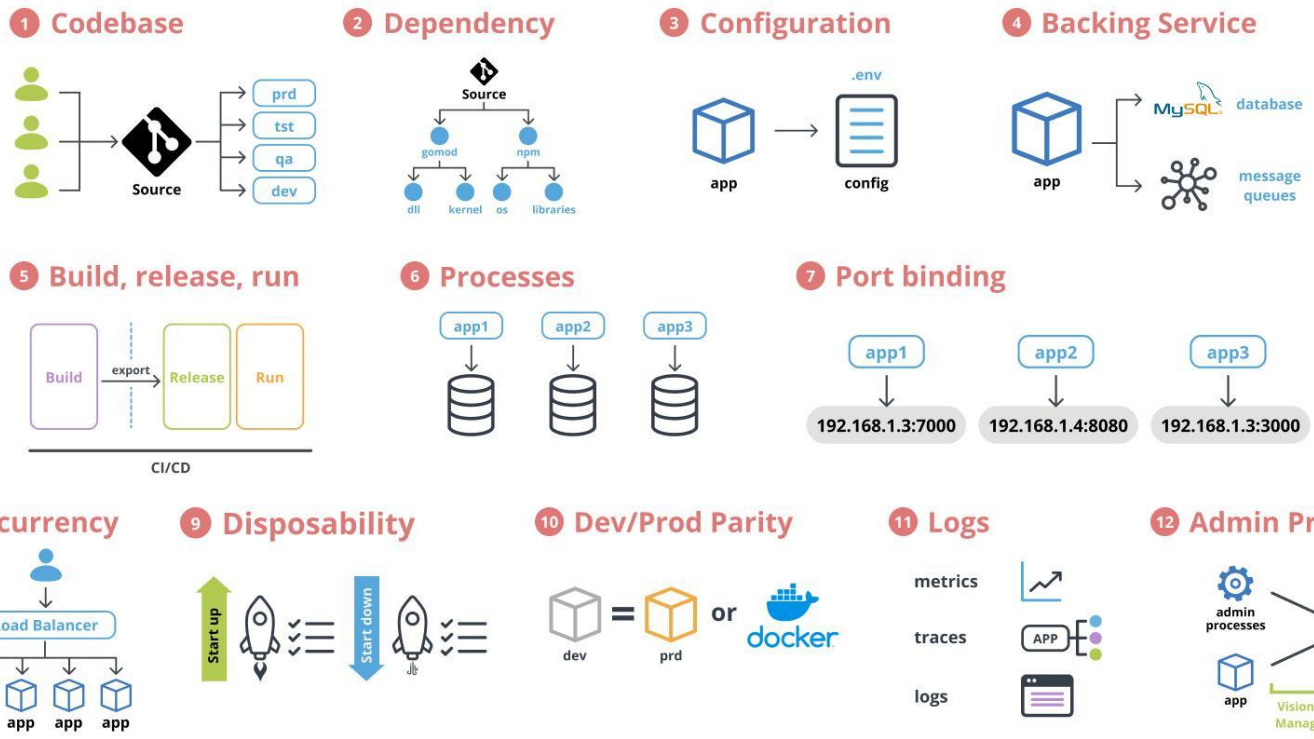


Transformation

Opsta Education

Skooldio

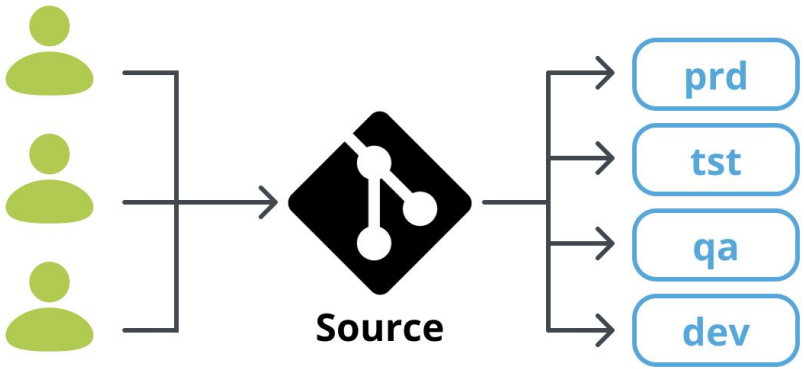
The Twelve-Factor App



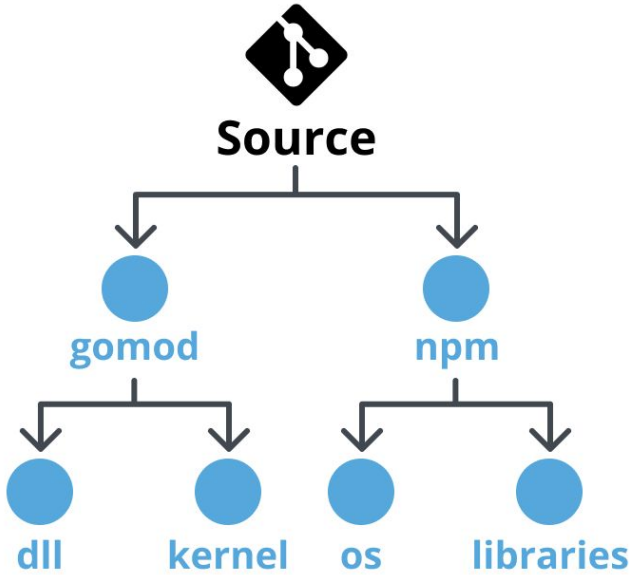
https://architecturenotes.co/12-factor-app-revisited/

1. Code Base

One codebase tracked in revision control, many deploys.
If there are **multiple codebases**, it's not an app – it's a **distributed system**. Each component in a distributed system is an app, and each can individually comply with twelve-factor.

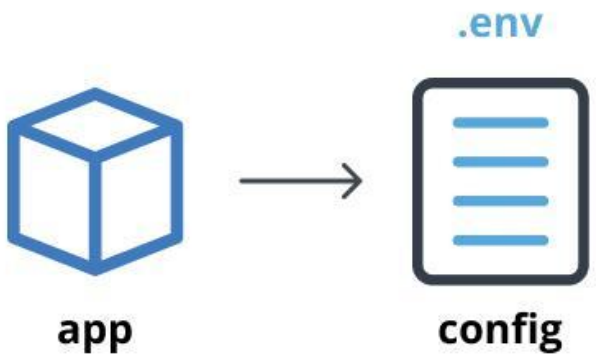


2. Dependencies



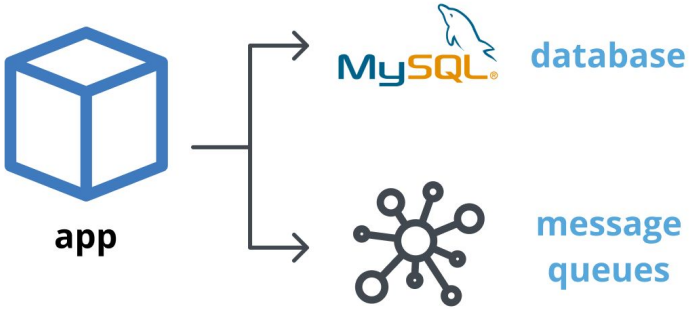
All dependencies should be declared, with no implicit reliance on system tools or libraries. With this introduction of various manifest files, our application can be rebuilt with consistency and accuracy.

3. Configuration



Configuration that varies between deployments should be stored in the environment. As tools take a more opinionated approach, these choices are being **moved** more and more towards tools like **Vault** or **service discovery systems** which manage and store configuration in environment-aware systems.

4. Backing Services



All backing services are treated as attached resources and attached and detached by the execution environment. This has been taken further when building a system that others will leverage as a platform with an API-first approach.

5. Build, Release, Run

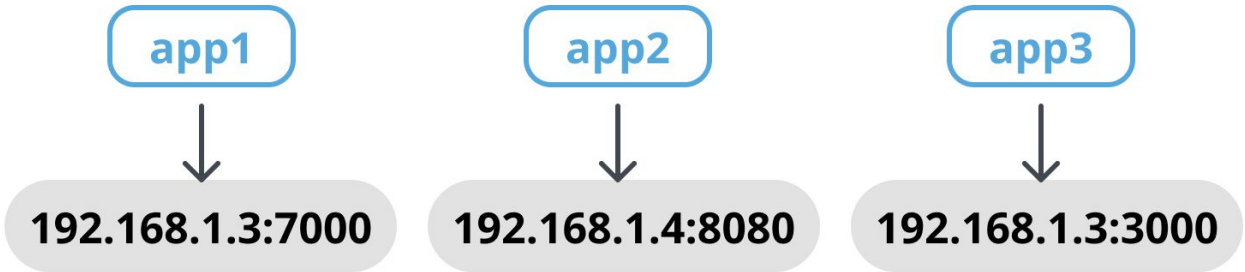
The delivery pipeline should strictly consist of build, release, run. It is not happen regularly is the **split** between the **build and release steps**, which is crucial to making sane deployment and rollback flows.

6. Processes

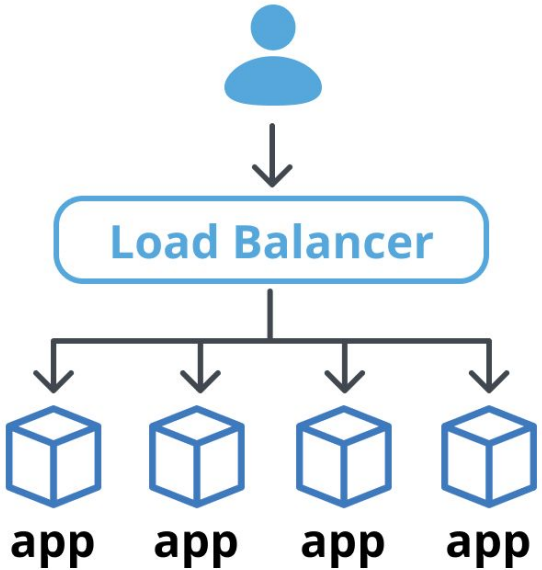
Applications should be deployed as one or more stateless processes with persisted data stored on a backing service. All communication across these applications should remain stateless or stored in another backing system.

7. Port Binding

Self-contained services should make themselves available to other services by specified ports. The main idea is that each application should have a specific port mapping.



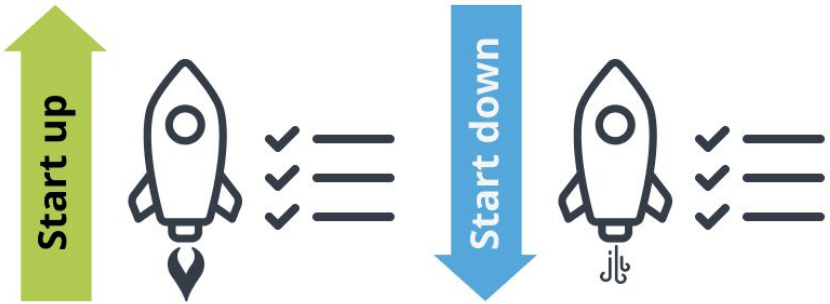
8. Concurrency



Concurrency is advocated by scaling individual processes. This factor summarizes various attributes of **horizontal scaling** and newer entrants into this space, such as **functions as a service**.

9. Disposability

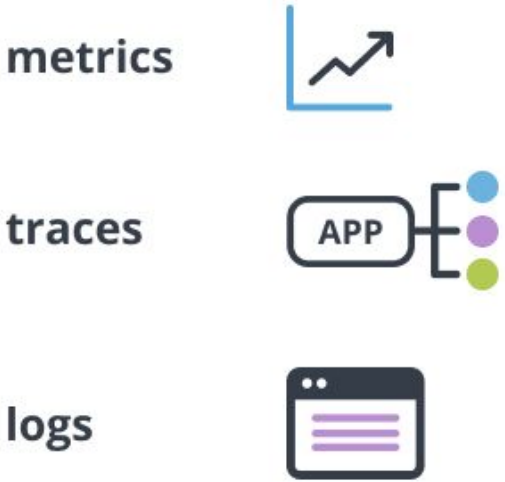
Fast startup and shutdown are advocated for a more robust and resilient system. Start-up checks are crucial to ensure systems are operating and functional. **Health checks** ensure the system continues in this state or sees itself removed from rotation.



10. Development/Production Parity

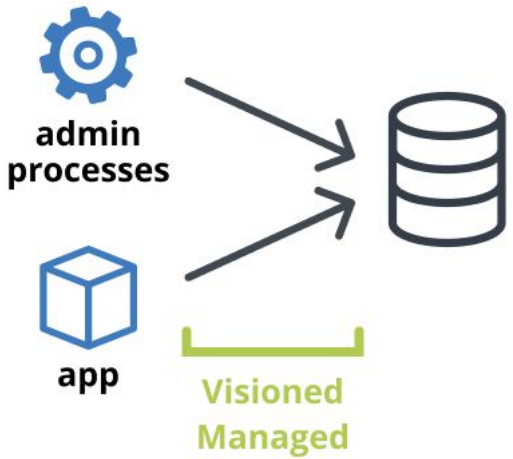
All environments should be as similar as possible. This factor is getting **worse** as we move more of our infrastructure into private clouds with **vendor lock-in**. Attempting to fit all those systems on a machine **readily** available to all **developers**.

11. Logs



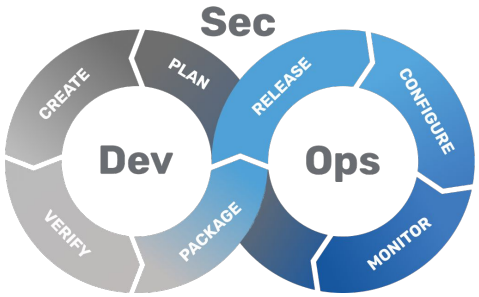
Applications should produce logs as event streams and leave the execution environment to aggregate. They will need to track **logs**, **metrics** and **traces** to truly understand and maintain their systems as they grow. Therefore, this factor likely needs to be updated to reflect application should be **observable without** the need to **modify** said system.

12. Admin Processes



Any needed admin tasks should be kept in source control and packaged with the application. This includes migration processes, dependency management, and even one-off processes for clean-up and management.

Q/A

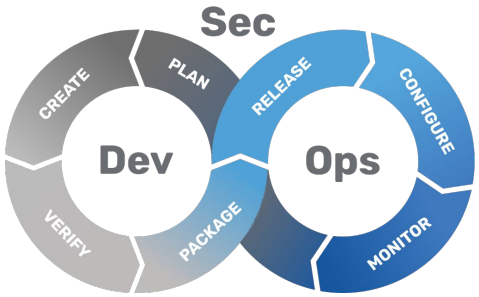


Transformation

Opsta Education

Skooldio

Wrap Up

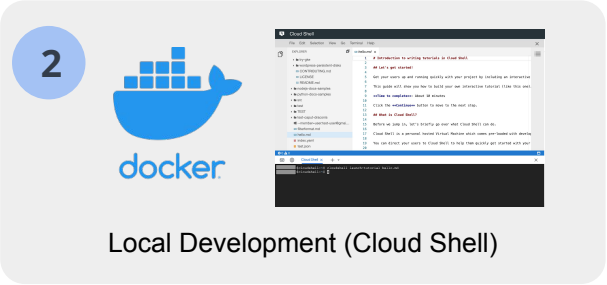


Transformation

Opsta Education

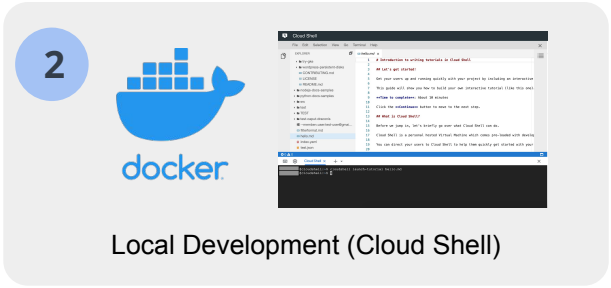
Skooldio

Day 1: Local Deployment

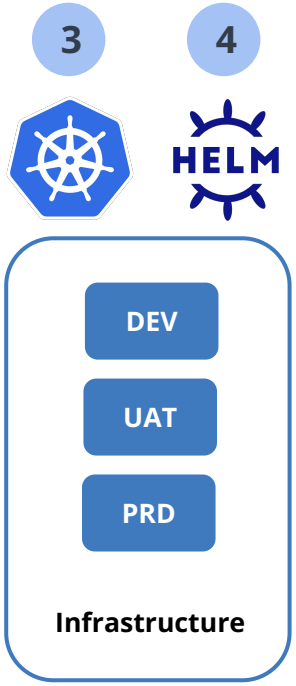


1. Put code into **Version Control**
2. Improve code to be **Twelve-Factor** App https://12factor.net
3. Build Docker Image to make it ready run in container with **Dockerfile**
4. Develop **docker-compose.yml** to run your application easier
5. Secure your application at the beginning such as using **secret** and authentication
6. **README.md** can be your document

Day 2: Manual Deployment

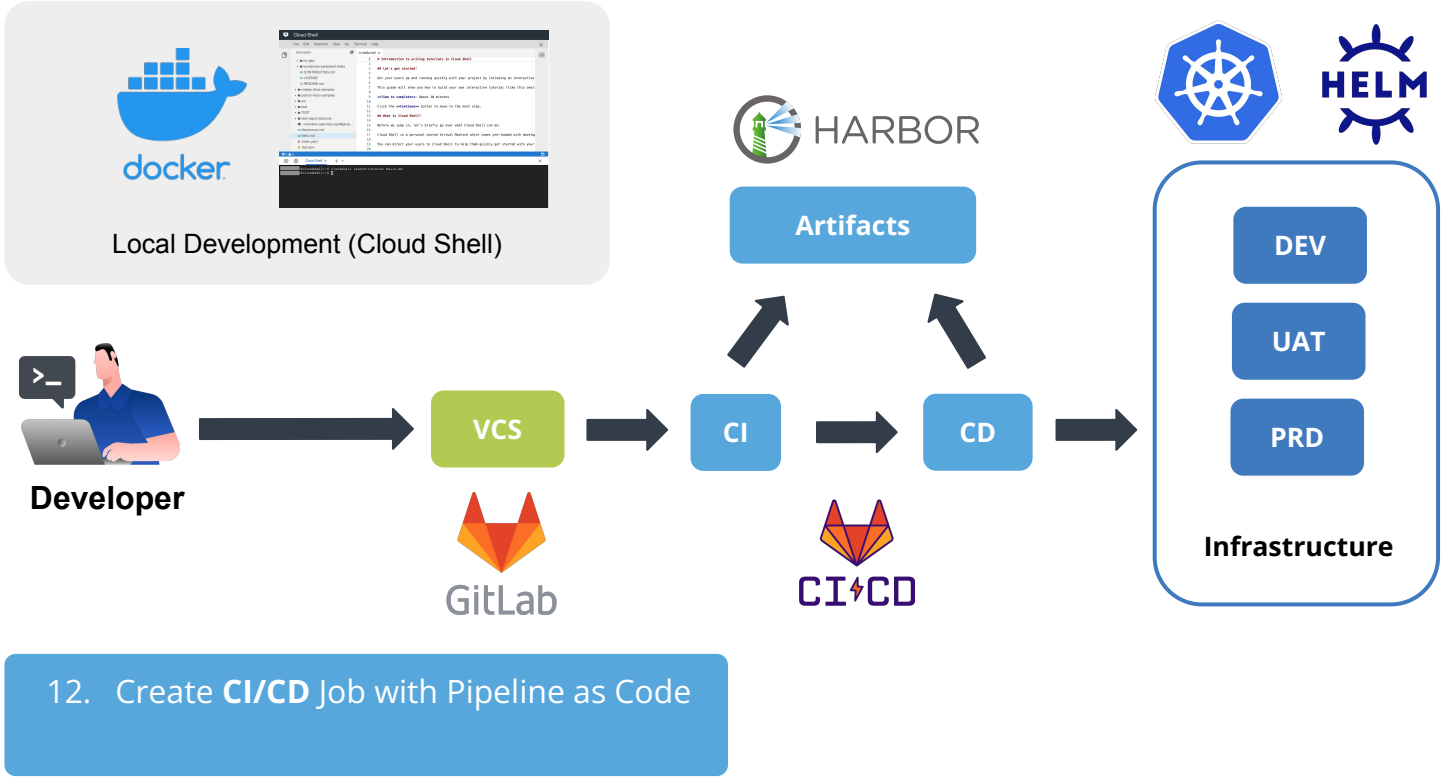


7. Push Docker Image to Private **Registry**
8. Manual Deploy Application with Kubernetes **Manifest File**
9. Convert Manifest to **Helm Chart**
10. Create **Helm Value**
11. **Manual** Deploy with Helm



Skooldio Opsta Education

Day 3: CI/CD & Artifacts



12. Create **CI/CD** Job with Pipeline as Code

Feedback and Contact Us

Jirayut Nimsaeng (Dear)

Founder & CEO

Facebook: fb.me/DearJirayut

Linkedin: www.linkedin.com/in/jirayut/

Email: jirayut@opsta.co.th

Website: www.opsta.co.th

Feedback


SECURITY
IN DEVSECOPS
docker
kubernetes
HashiCorp
Vault


INFRASTRUCTURE AS CODE
AND GITOPS
ANSIBLE
HashiCorp
Terraform


OBSERVABILITY
THE NEXT LEVEL MONITORING
Prometheus
Grafana
LOCUST