# Using Jenkins X for Cloud-native CI/CD

## JENKINS X AERIAL VIEW

**Andrew Morgan**
INDEPENDENT
@mogronalol

# Overview

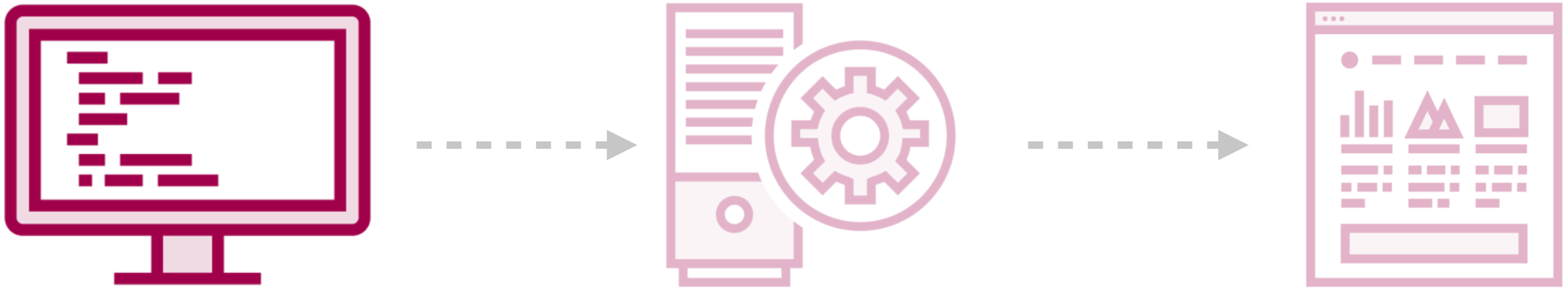
What does a classic CI/CD platform look like and what are its shortcomings?

Introduction to Jenkins X and how it addresses these shortcomings

Jenkins X architecture

Differences between classic Jenkins and Jenkins X

# A Typical CI/CD System



Local code

CI/CD Server
(Jenkins)

Production

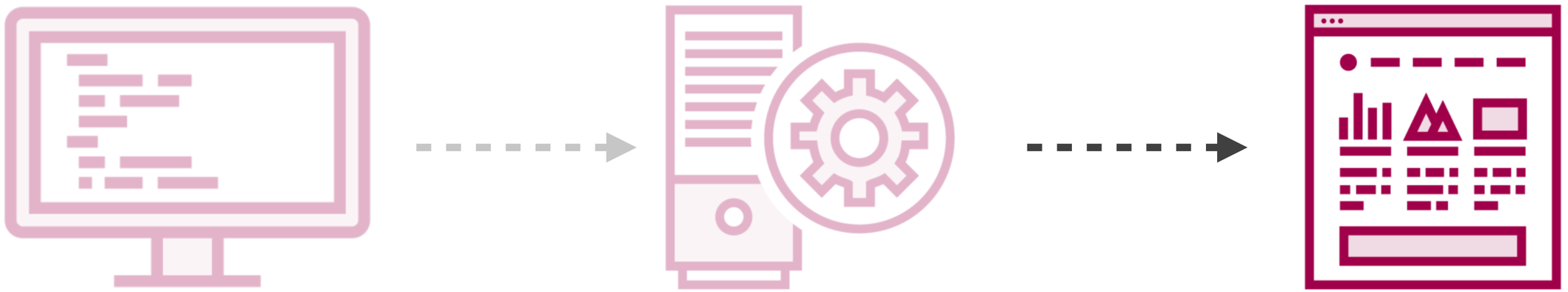
# A Typical CI/CD System

Local code

**CI/CD Server (Jenkins)**

Production

# A Typical CI/CD System



Local code

CI/CD Server
(Jenkins)

**Production**

A core problem with traditional CI/CD platforms like classic Jenkins is that they are un-opinionated and require heavy customization

# Too Much Extra Scaffolding

- Installation and configuration
- Version control platform integration
- Environment creation
- Project creation
- Custom pipelines
- Packaging strategy
- Registry provisioning
- Deployment strategies

# Running in the Cloud

**Not cloud-native**

# Running in the Cloud



Not cloud-native

**Static and resource hungry**

# Running in the Cloud



Not cloud-native



Static and resource hungry

**Provisioning new agents**

# Custom Deployment Strategy

**Deployments may not be versioned in Git (GitOps)**

# Custom Deployment Strategy

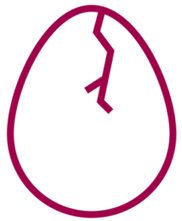
Deployments may not be versioned in Git (GitOps)

**They may not be observable and give free reign to the operations cowboy**

# Custom Deployment Strategy

Deployments may not be versioned in Git (GitOps)

They may not be observable and give free reign to the operations cowboy



**They may not be declarative and idempotent**

# Pipeline Repetition

```
pipeline {
    agent { docker { image 'maven:3.3.3' } }
    stages {
        stage('build') {
            steps {
                sh ‘./mvn install'
            }
        }
        stage(‘dockerize') {
            steps {
                sh ‘docker build'
            }
        }
    }
}
```

# Pipeline Repetition

```
pipeline {
    agent { docker { image 'maven:3.3.3' } }
    stages {
        stage('build') {
            steps {
                sh ‘./mvn install'
            }
        }
        stage(‘dockerize') {
            steps {
                sh ‘docker build'
            }
        }
    }
}
```

Same for all Maven builds

# Pipeline Repetition

```
pipeline {
    agent { docker { image 'maven:3.3.3' } }
    stages {
        stage('build') {
            steps {
                sh ‘./mvn install'
            }
        }
        stage(‘dockerize') {
            steps {
                sh ‘docker build'
            }
        }
    }
}
```

Same for all Docker projects

Jenkins X is an opinionated, cloud-native CI/CD platform built on top of Kubernetes

# The Opinions of Jenkins X

**Our application runs on Kubernetes**

# Kubernetes

**Don’t worry if you’re a beginner!**

**Container orchestration tool built for the cloud**

- Unified declarative deployment model
- Service-discovery and load balancing
- Horizontal scaling
- Self-healing

**Check out a Kubernetes Pluralsight course**

# The Opinions of Jenkins X

**Our application runs on Kubernetes**

**We use Helm and Docker**

# The Opinions of Jenkins X

**Our application runs on Kubernetes**

**We use Helm and Docker**

**We use GitOps**

# The Opinions of Jenkins X

**Our application runs on Kubernetes**

**We use Helm and Docker**

**We use GitOps**

**Each language has a re-useable default structure (buildpack)**

# The Opinions of Jenkins X

**Our application runs on Kubernetes**

**We use Helm and Docker**

**We use GitOps**

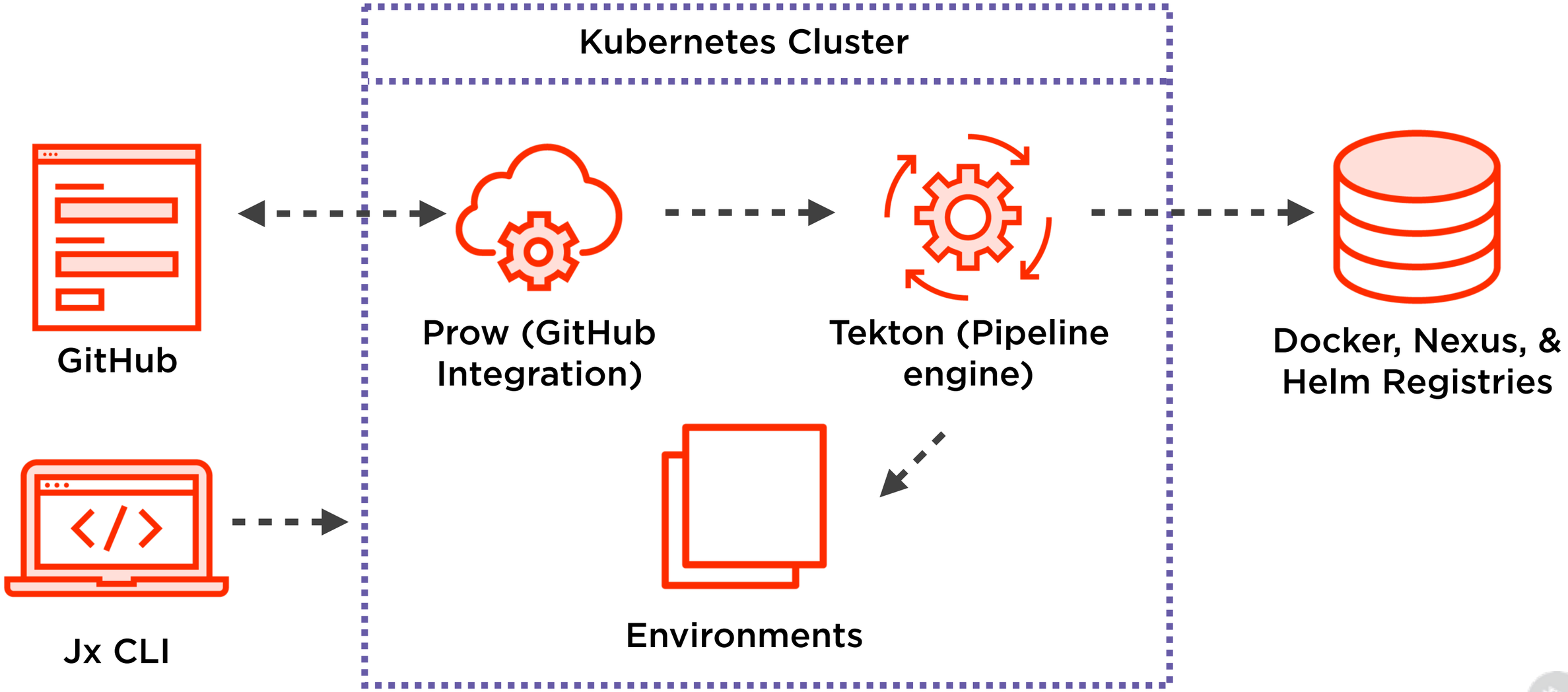
**Each language has a re-useable default structure (buildpack)**

**Pipelines are extendable and re-usable**

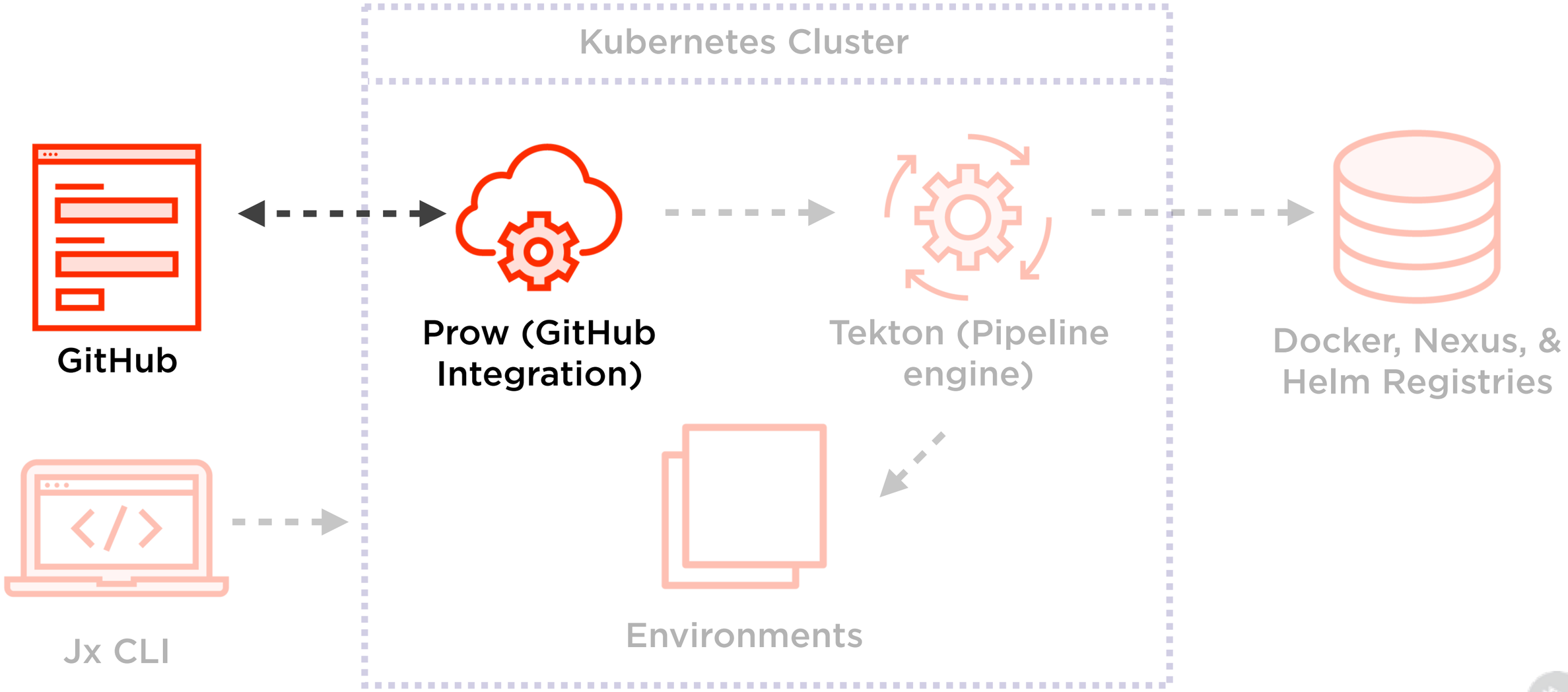
# The Opinions of Jenkins X

| | | |
|---|---|---|
| Our application runs on Kubernetes | We use Helm and Docker | We use GitOps |
| Each language has a re-useable default structure (buildpack) | Pipelines are extendable and re-usable | Its components are elastic |

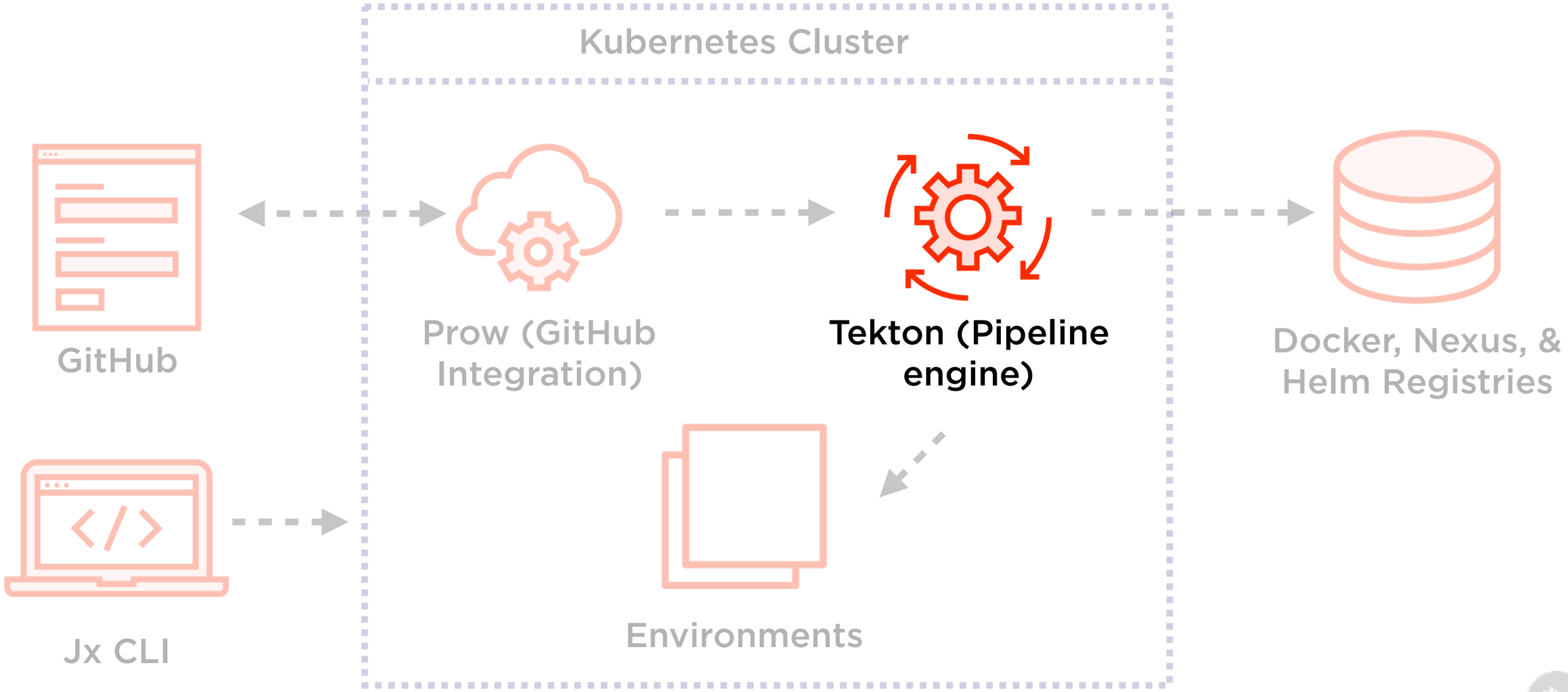
# High-level Jenkins X Architecture



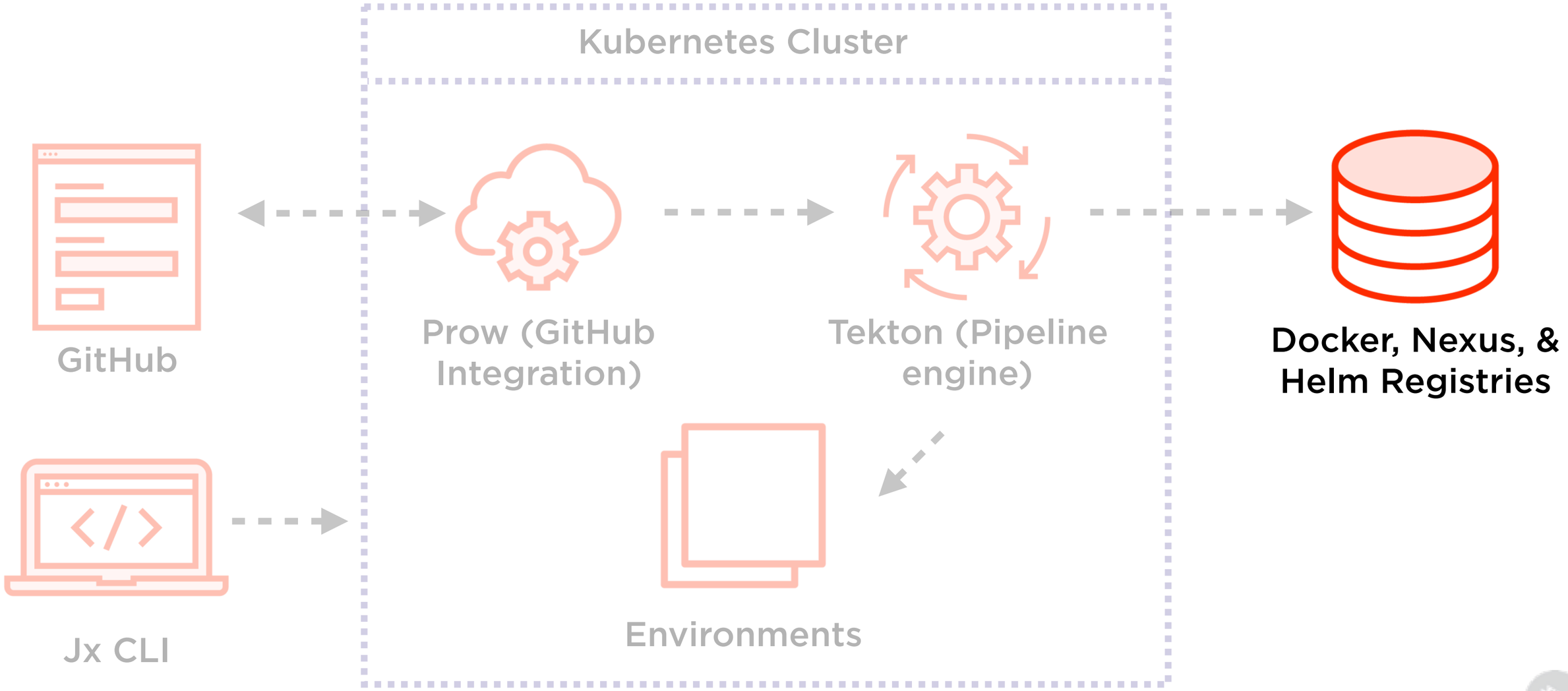
# High-level Jenkins X Architecture



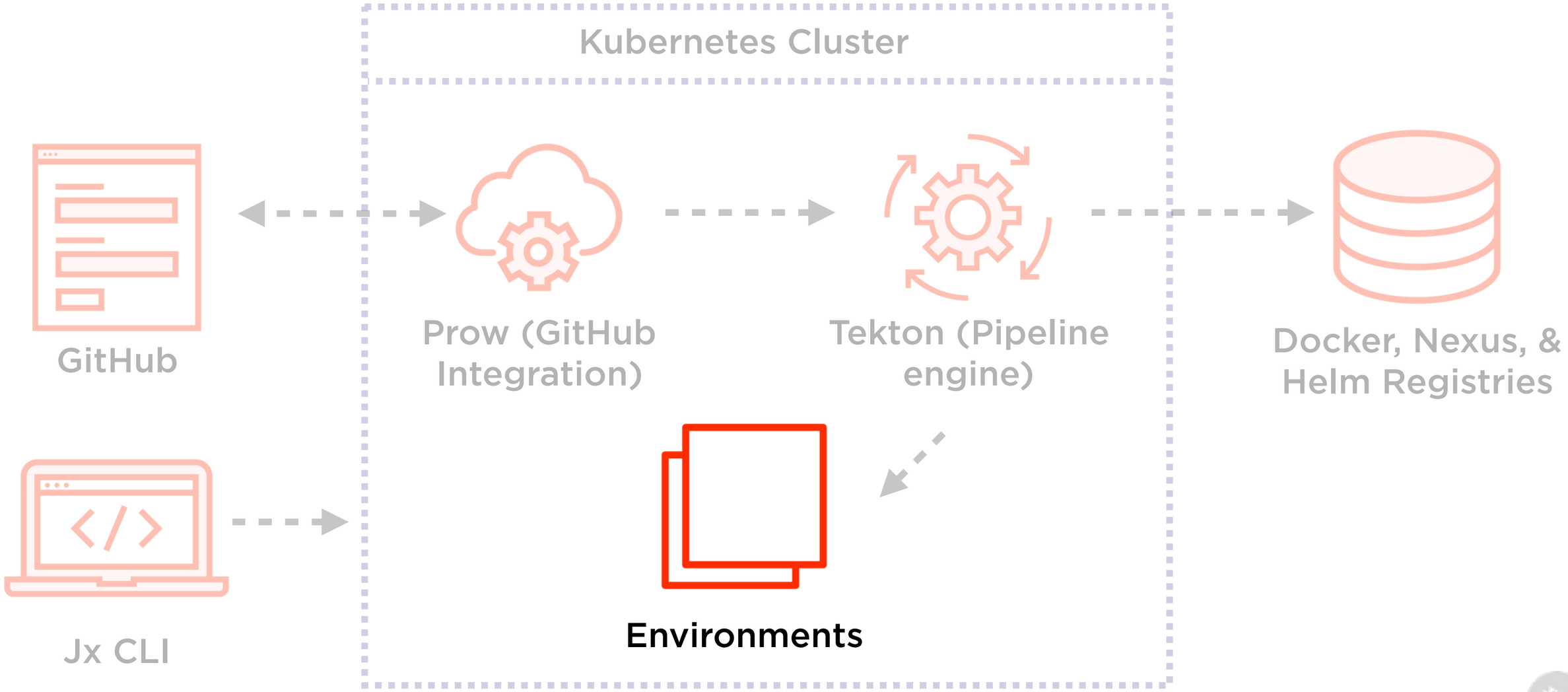
# High-level Jenkins X Architecture



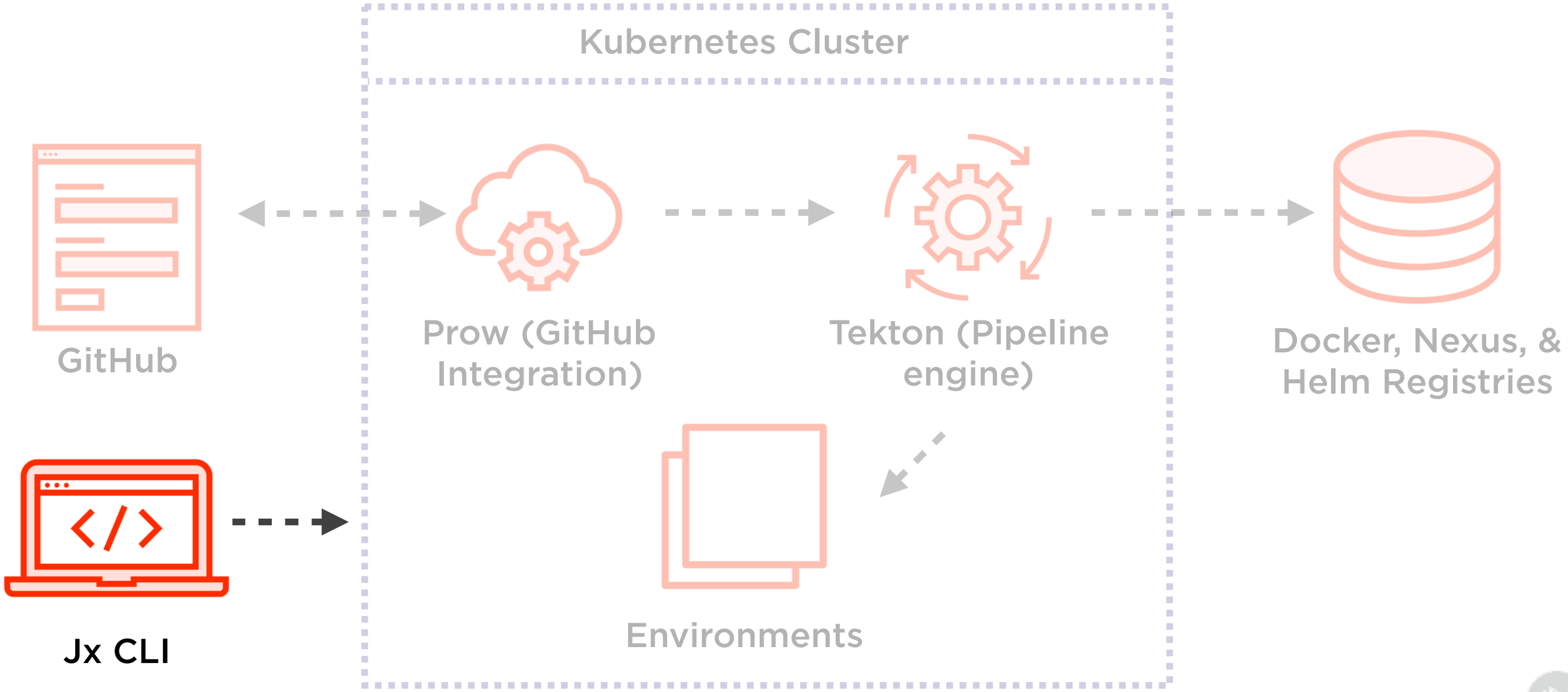
# High-level Jenkins X Architecture



# High-level Jenkins X Architecture



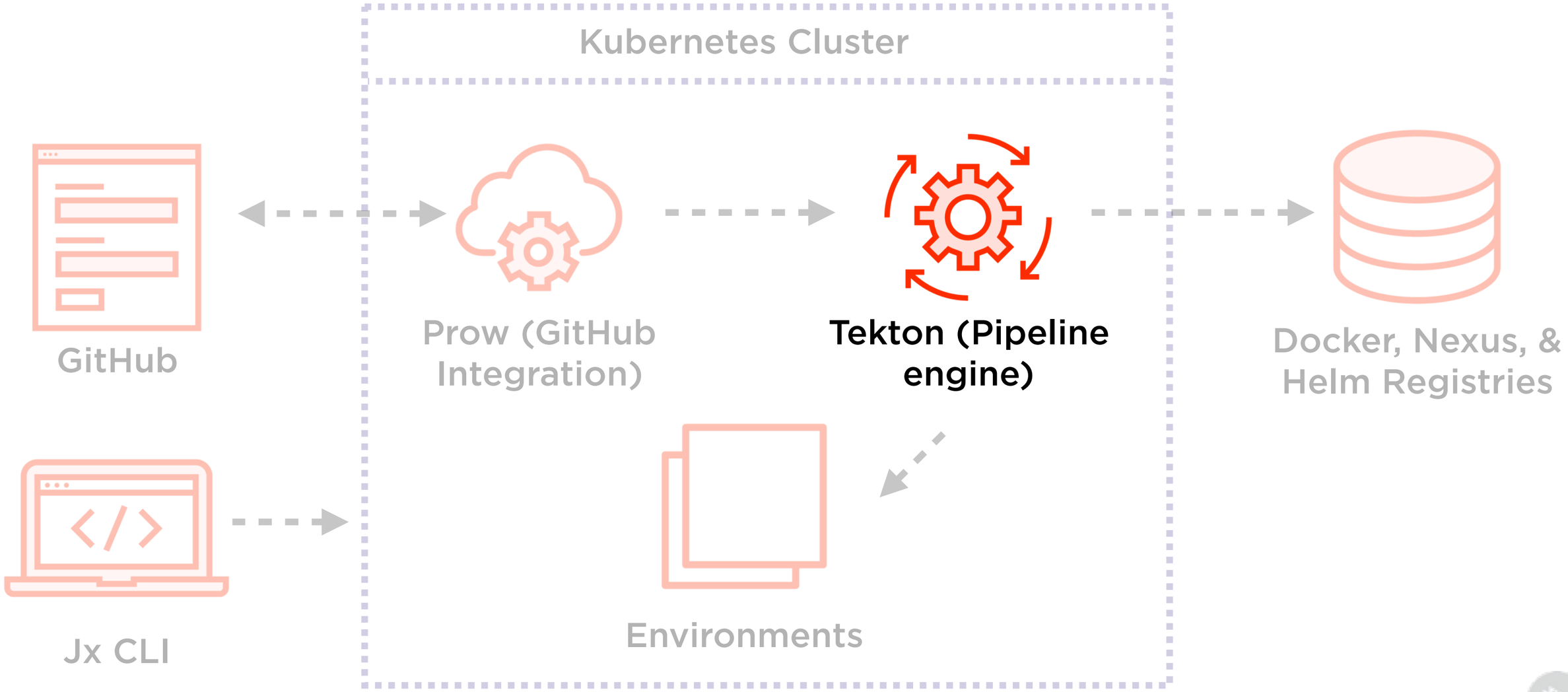
# High-level Jenkins X Architecture



# The Differences between Classic Jenkins and Jenkins X

| Classic Jenkins | Jenkins X |
| --- | --- |
| General purpose CI/CD server | Entire end-to-end CI/CD platform |
| Runs wherever you like | Runs on Kubernetes |
| Requires a custom deployment and packaging implementation | Applications use Helm and Docker for packaging and deployments |
| You must set up your own infra for hosting environments | Environments up and running by default within Kubernetes |
| Pipeline and project creation boilerplate | Sensible defaults for projects and pipelines |
| Was originally the pipeline engine for Jenkins X | No longer used by Jenkins X all and replaced with Tekton |

# The Difference to Classic Jenkins



# Summary

Classic CI/CD with Jenkins is not opinionated enough and requires too much customization

Jenkins X is an opinionated ecosystem which gives you an entire end-to-end CI/CD platform out of the box

Classic Jenkins was originally the pipeline engine for Jenkins X, but is now no longer part of it