NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Deployment as a Service

Cloud Infrastructure Engineering

Nanyang Technological University & Skills Union - 2022/2023

Course Content

- Quick Check-In
- Dive into the basics of Deployment as a Service in AWS
- Explore the Deployment services in AWS
- Explore the different Deployment strategies

Time	What	How or Why
7:15pm - 7:45pm	Part 1 - Presentation	AWS Deployment Services
7:45pm - 8:05pm	Part 2 - Activity	AWS Deployment Services Activity
8:00pm - 8:10pm	Break	
8:10pm - 8:25pm	Part 3 - Activity	Hands-on
8:25pm - 8:40pm	Part 4 - Presentation	Deployment Strategies
8:40pm - 9:00pm	Part 5 - Activity	Deployment Strategies Activity
9:00pm - 10:00pm	Summary & Assignments	

Recap

- SDLC
  - Requirements > Planning > Design > Build > Testing > Deployment & Maintenance
- Deployments
  - Basic, Rolling, Blue-Green, Canary
- CI/CD
- Containerization
- Agile vs Waterfall
- Scrum

Self Study Check-In

Q1) What is deployment as a service?

Q2) What are the Deployment Services that you know available on AWS?

Q3) What are some Deployment Strategies that you know?

Lesson Overview

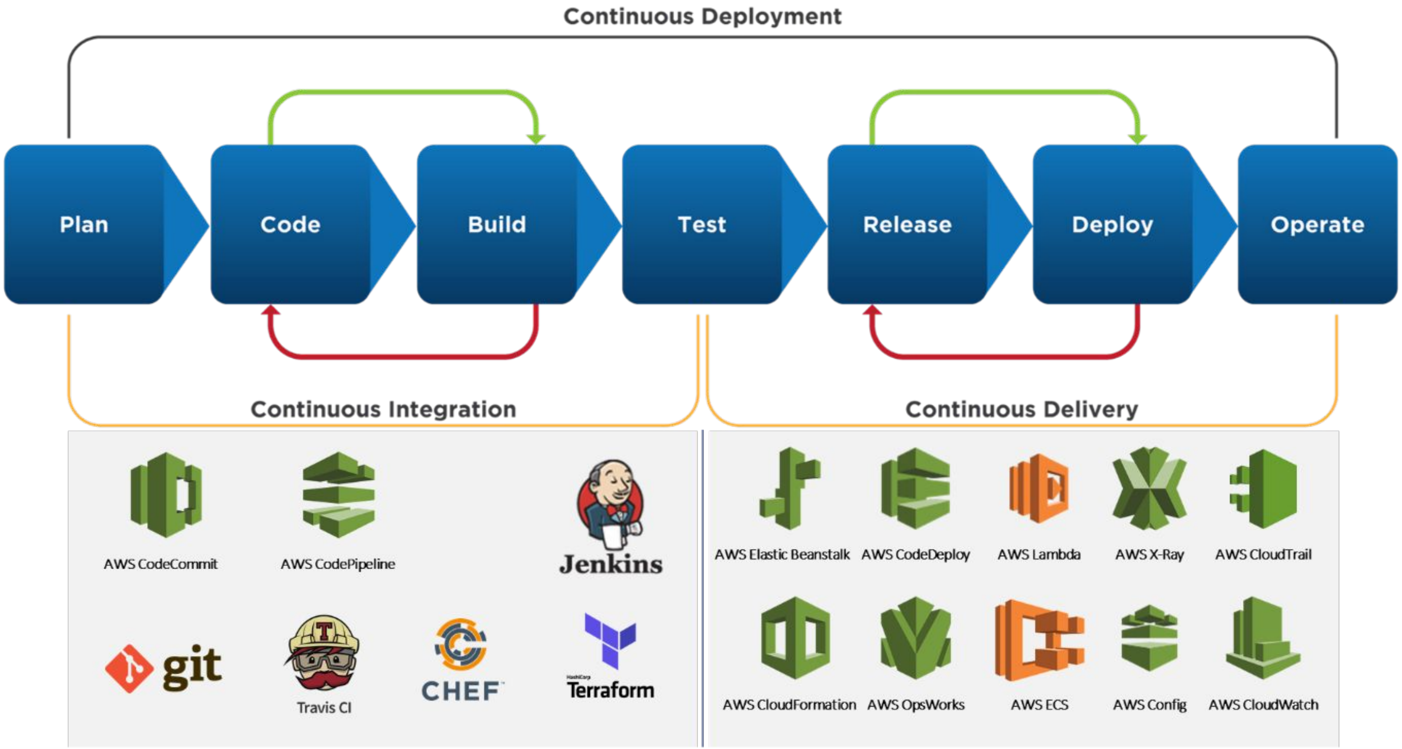
General Overview

AWS offers multiple options for provisioning infrastructure and deploying your applications.

Whether your application architecture is a simple three-tier web application or a complex set of workloads, AWS **offers deployment services** to meet the requirements of your application and your organization.

AWS lays out common features available in these deployment services, and articulates basic strategies for deploying and updating application stacks.

General Overview



AWS Deployment Services

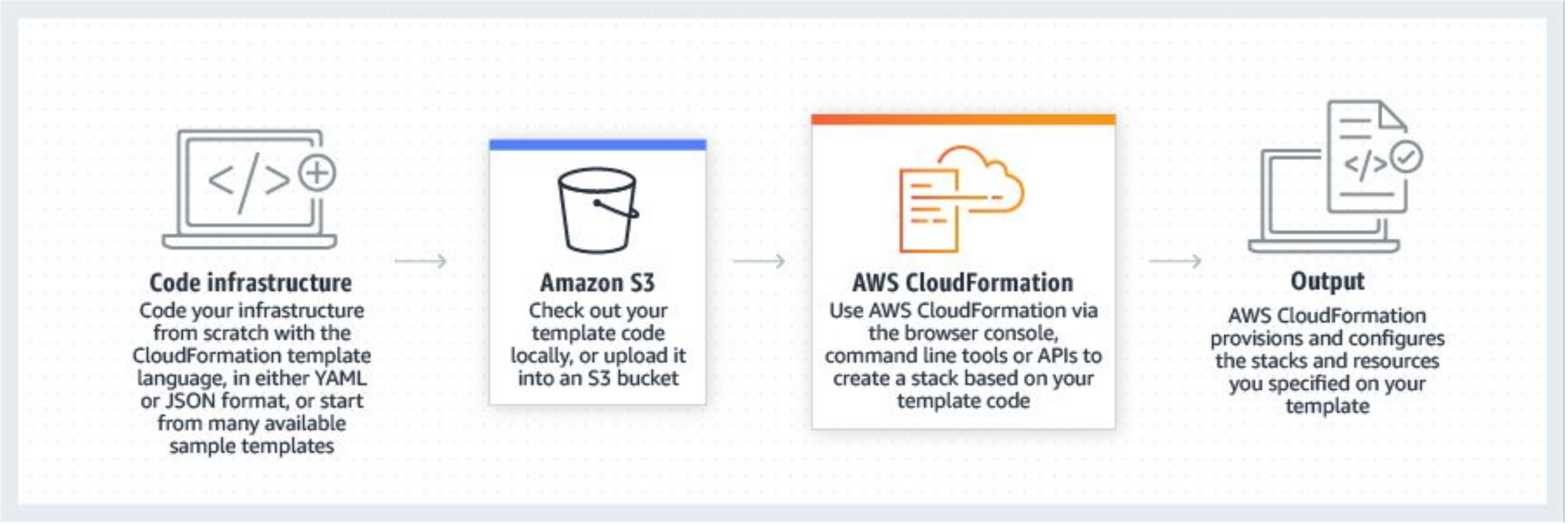
Summary

- AWS Cloudformation
- AWS Elastic Beanstalk
- AWS CodeCommit
- AWS CodePipeline
- AWS CodeBuild
- AWS CodeDeploy
- AWS OpsWork

CloudFormation

AWS CloudFormation is a service that **enables customers to provision and manage almost any AWS resource** using a custom template language expressed in **YAML or JSON.**

CloudFormation



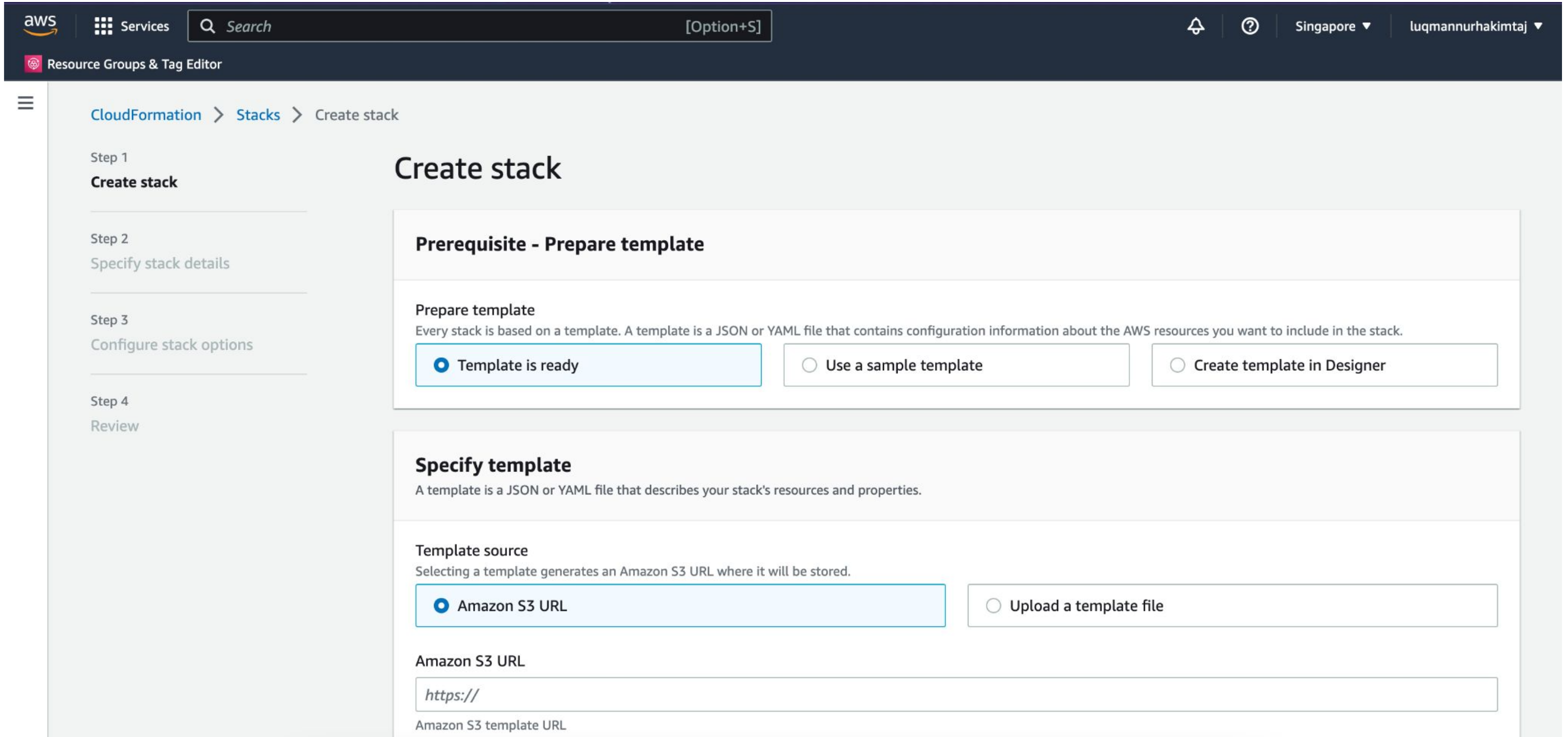
Code infrastructure
Code your infrastructure from scratch with the CloudFormation template language, in either YAML or JSON format, or start from many available sample templates

Amazon S3
Check out your template code locally, or upload it into an S3 bucket

AWS CloudFormation
Use AWS CloudFormation via the browser console, command line tools or APIs to create a stack based on your template code

Output
AWS CloudFormation provisions and configures the stacks and resources you specified on your template

CloudFormation



aws
Services
Search
[Option+S]
Singapore ▼
luqmannurhakimtaj ▼
Resource Groups & Tag Editor
CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review

Create stack

Prerequisite - Prepare template

Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

● Template is ready ○ Use a sample template ○ Create template in Designer

Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

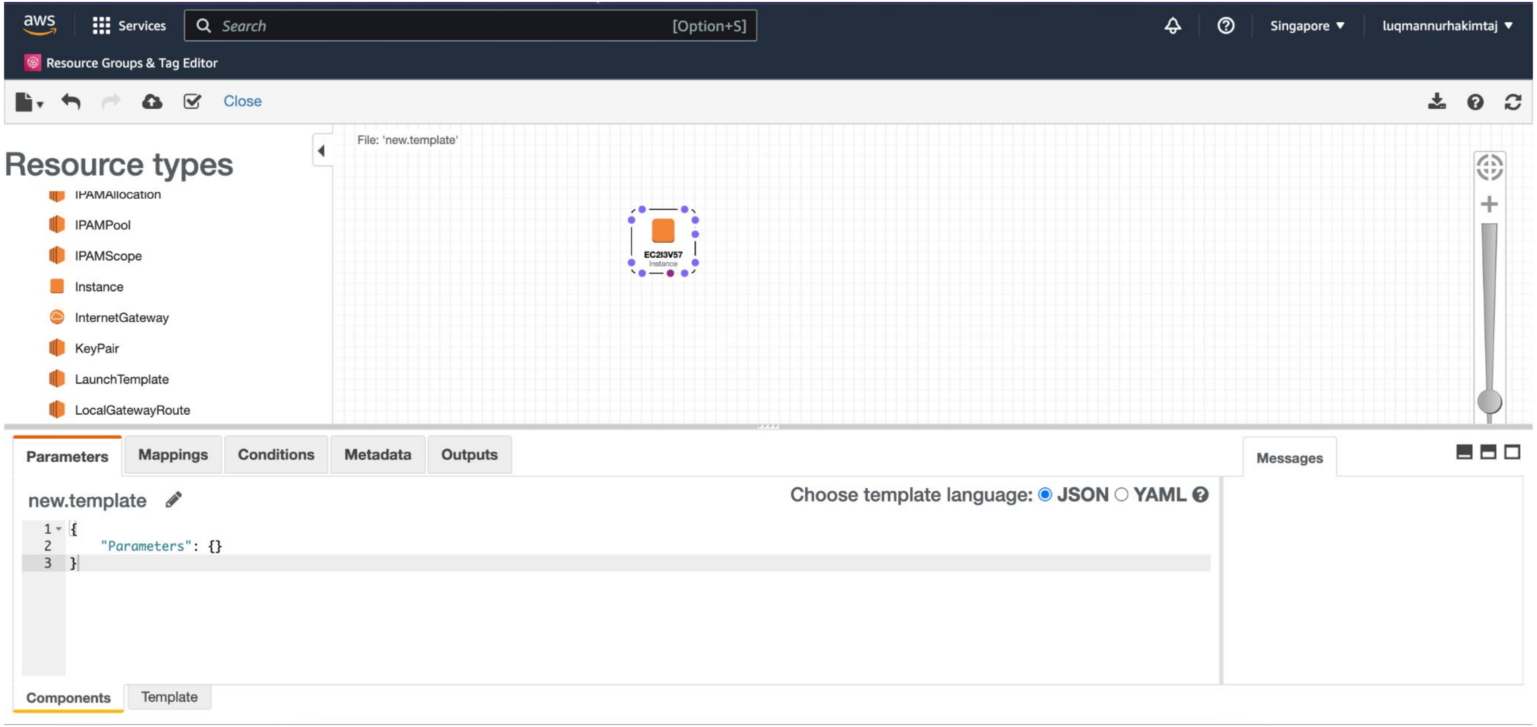
● Amazon S3 URL ○ Upload a template file

Amazon S3 URL

https://

Amazon S3 template URL

CloudFormation - Designer



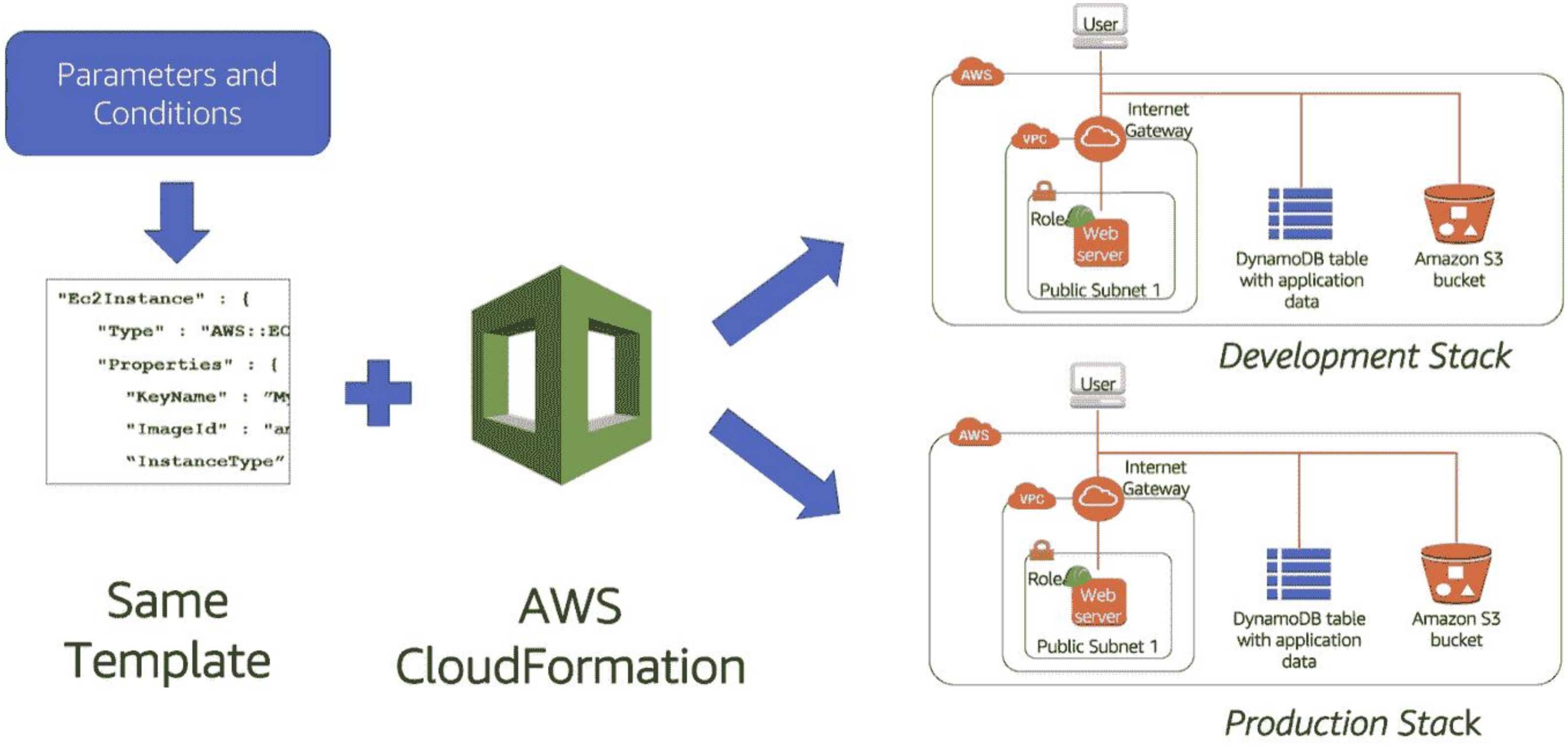
File: 'new.template'

new.template

Choose template language: ● JSON ○ YAML

```
{
    "Parameters": {}
}
```

CloudFormation



Parameters and Conditions

```
"Ec2Instance" : {
    "Type" : "AWS::EC
    "Properties" : {
      "KeyName" : "My
      "ImageId" : "ar
      "InstanceType"
```

Same Template

AWS CloudFormation

Development Stack

Production Stack

CloudFormation

A CloudFormation template **creates infrastructure resources in a group called a “stack,”** and **allows you to define and customize all components needed** to operate your application while retaining full control of these resources.

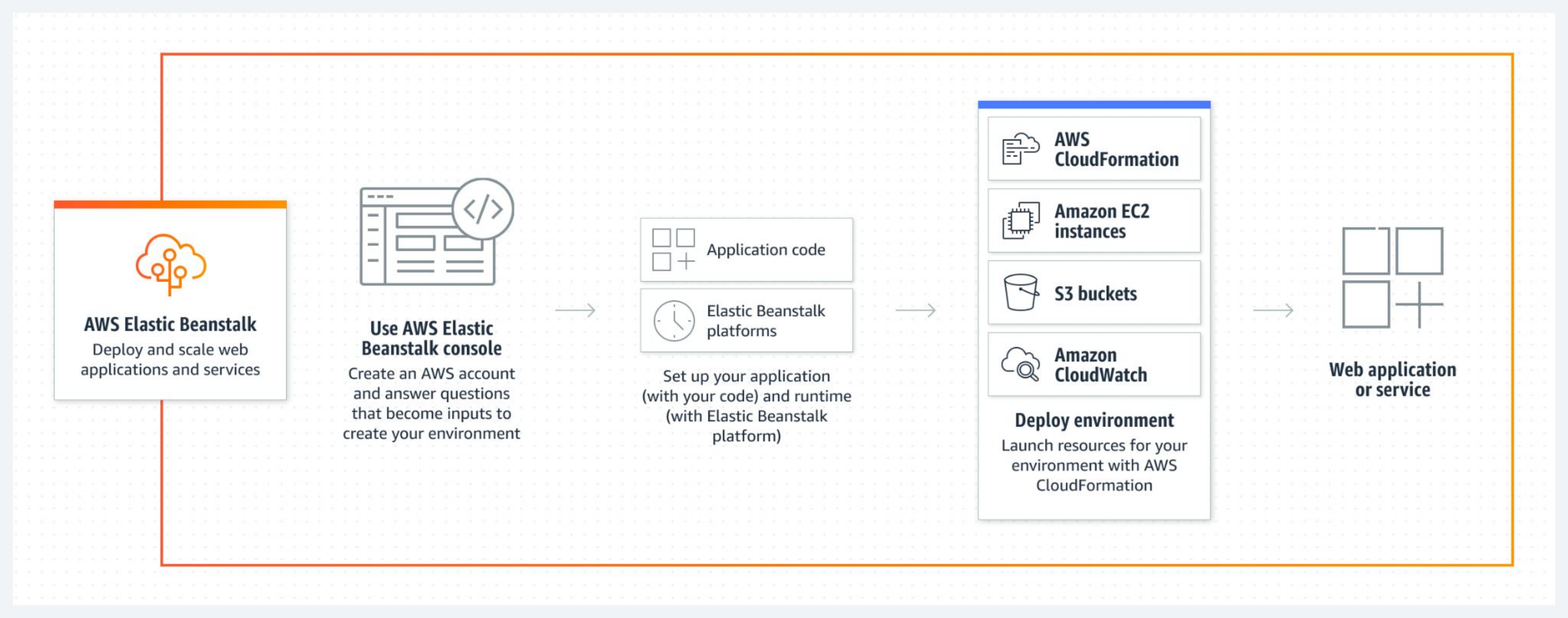
Using templates introduces the ability to implement version control on your infrastructure, and the **ability to quickly and reliably replicate your infrastructure**.

Elastic Beanstalk

AWS Elastic Beanstalk is an easy-to-use service for **deploying** and **scaling web applications and services** developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.

Elastic Beanstalk is a **complete application management solution**, and **manages all infrastructure and platform tasks on your behalf**.

Elastic Beanstalk



AWS Elastic Beanstalk

Deploy and scale web applications and services

Use AWS Elastic Beanstalk console

Create an AWS account and answer questions that become inputs to create your environment

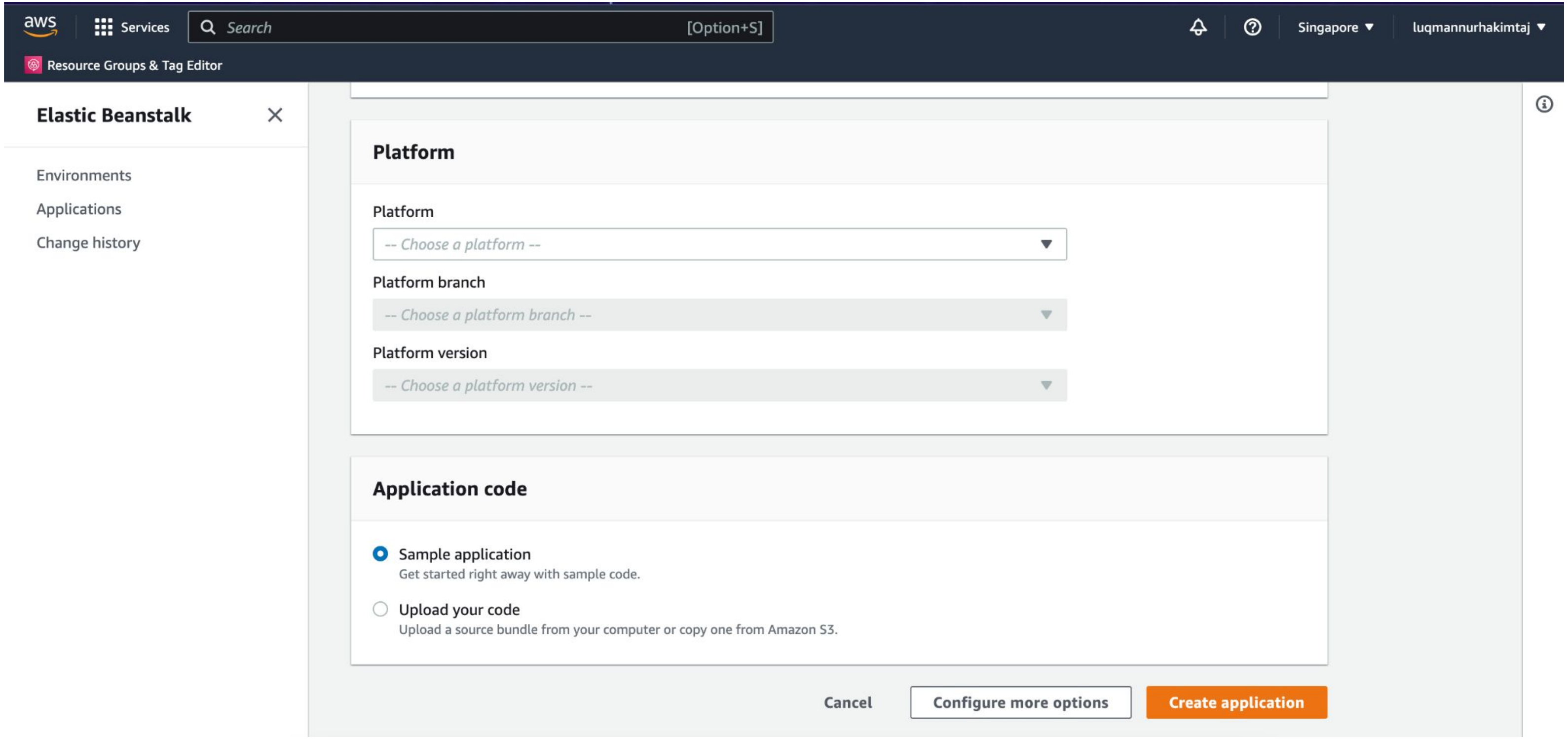
Set up your application (with your code) and runtime (with Elastic Beanstalk platform)

Deploy environment

Launch resources for your environment with AWS CloudFormation

Web application or service

Elastic Beanstalk



Platform

Platform

-- Choose a platform -- ▼

Platform branch

-- Choose a platform branch -- ▼

Platform version

-- Choose a platform version -- ▼

Application code

● Sample application
Get started right away with sample code.

○ Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

Elastic Beanstalk

With Elastic Beanstalk, you can **quickly deploy, manage, and scale applications** without the operational burden of managing infrastructure.

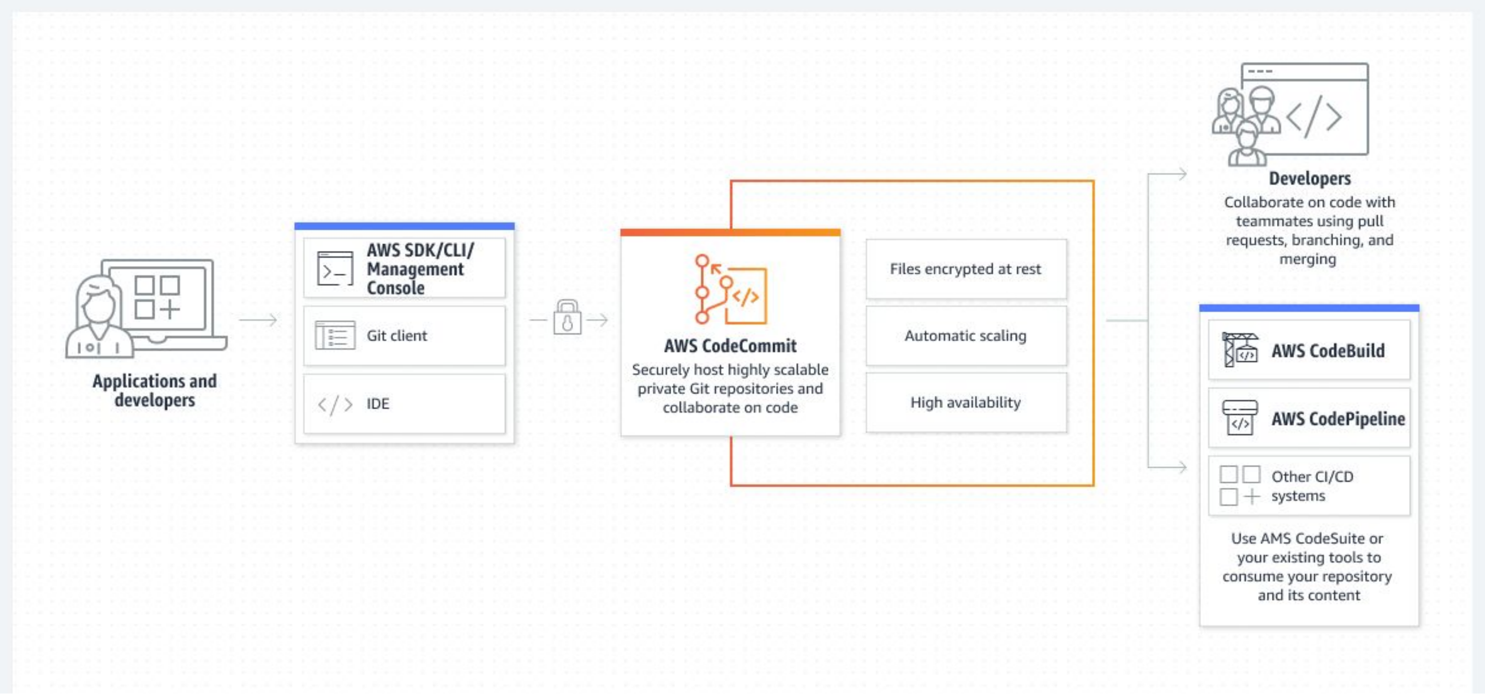
Elastic Beanstalk **reduces management complexity** for web applications, making it a good choice for organizations that are new to AWS or wish to deploy a web application as quickly as possible.

CodeCommit

AWS CodeCommit is a **secure, highly scalable, fully managed source control service that hosts private Git repositories**.

Alternatives include GitHub and Bitbucket.

CodeCommit



Applications and developers

AWS SDK/CLI/ Management Console

Git client

IDE

AWS CodeCommit

Securely host highly scalable private Git repositories and collaborate on code

Files encrypted at rest

Automatic scaling

High availability

Developers

Collaborate on code with teammates using pull requests, branching, and merging

AWS CodeBuild

AWS CodePipeline

Other CI/CD systems

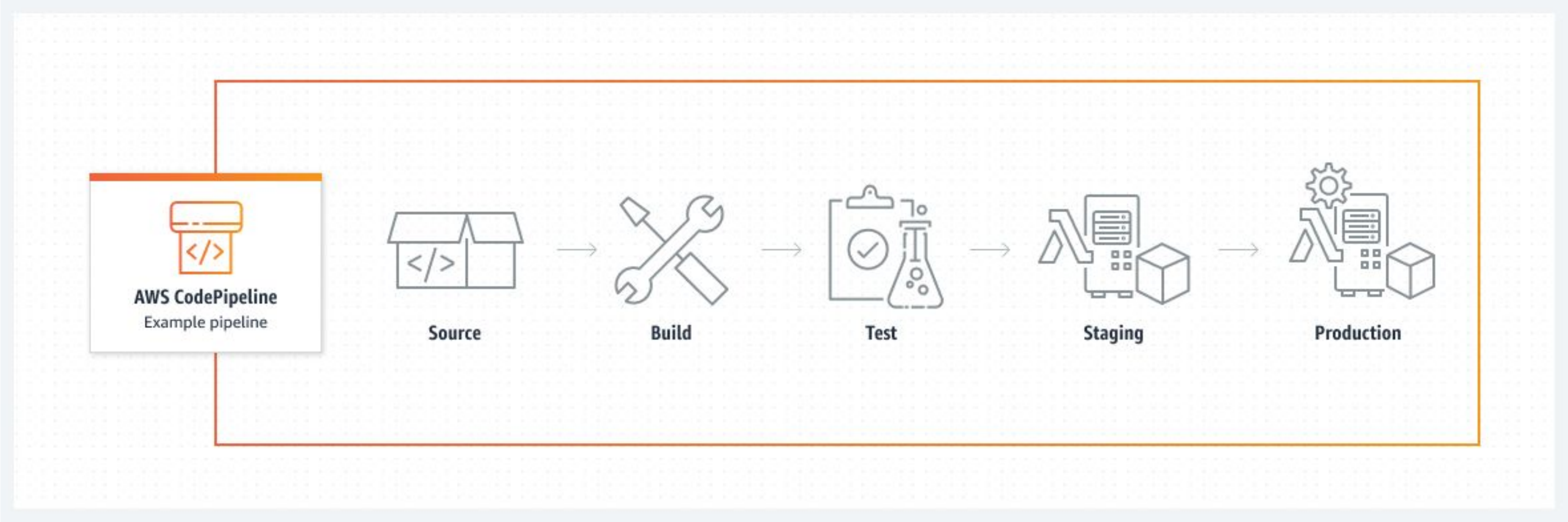
Use AMS CodeSuite or your existing tools to consume your repository and its content

CodePipeline

AWS CodePipeline is a **fully managed continuous delivery service** that helps you automate your release pipelines for fast and reliable application and infrastructure updates.

Alternatives include Jenkins and Gitlab.

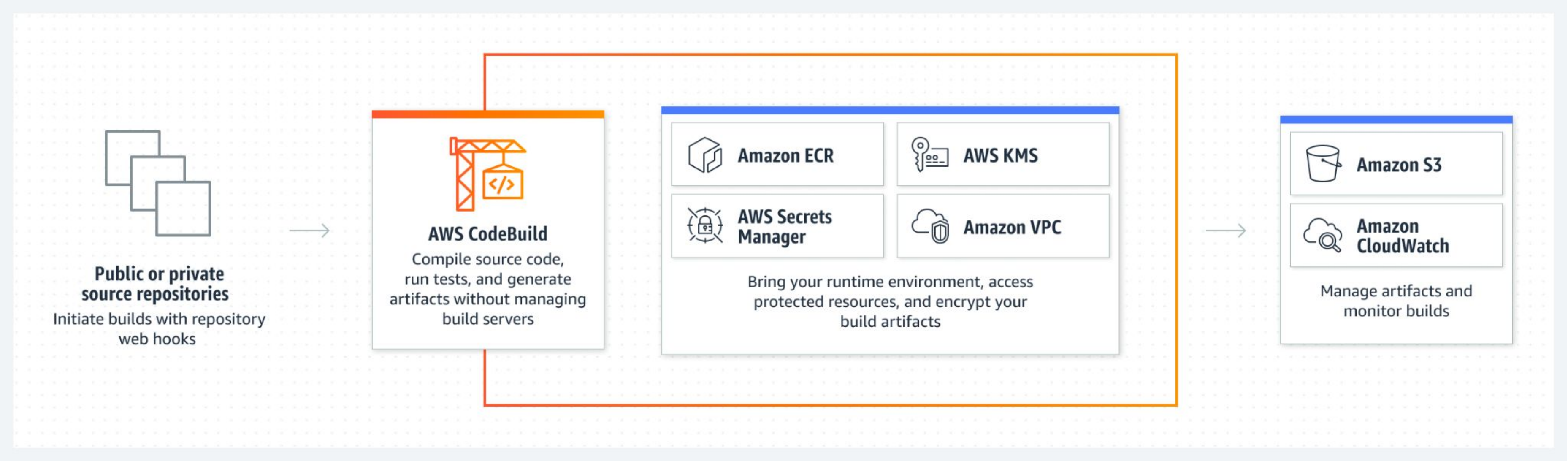
CodePipeline



CodeBuild

AWS CodeBuild is a **fully managed continuous integration service** that compiles source code, runs tests, and produces ready-to-deploy software packages.

CodeBuild



Public or private source repositories

Initiate builds with repository web hooks

AWS CodeBuild

Compile source code, run tests, and generate artifacts without managing build servers

Amazon ECR

AWS KMS

AWS Secrets Manager

Amazon VPC

Bring your runtime environment, access protected resources, and encrypt your build artifacts

Amazon S3

Amazon CloudWatch

Manage artifacts and monitor builds

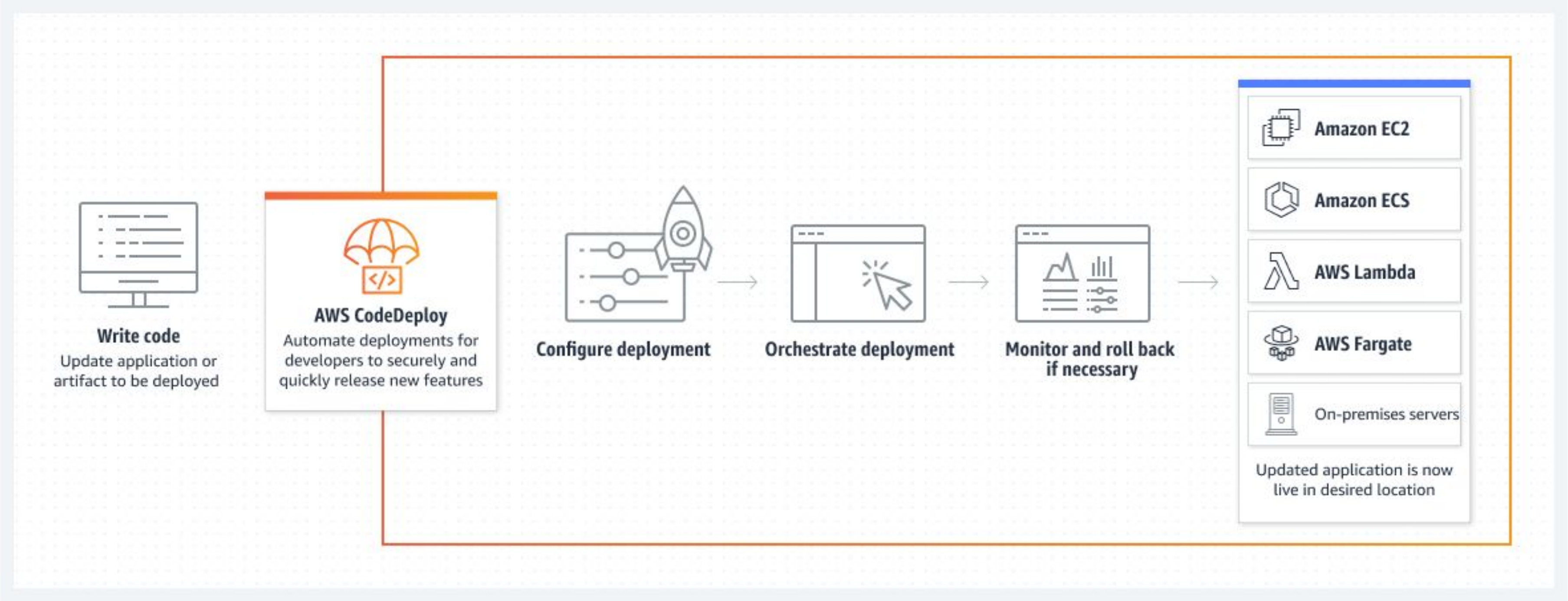
CodeDeploy

AWS CodeDeploy is a **fully managed deployment service** that automates application deployments to compute services.

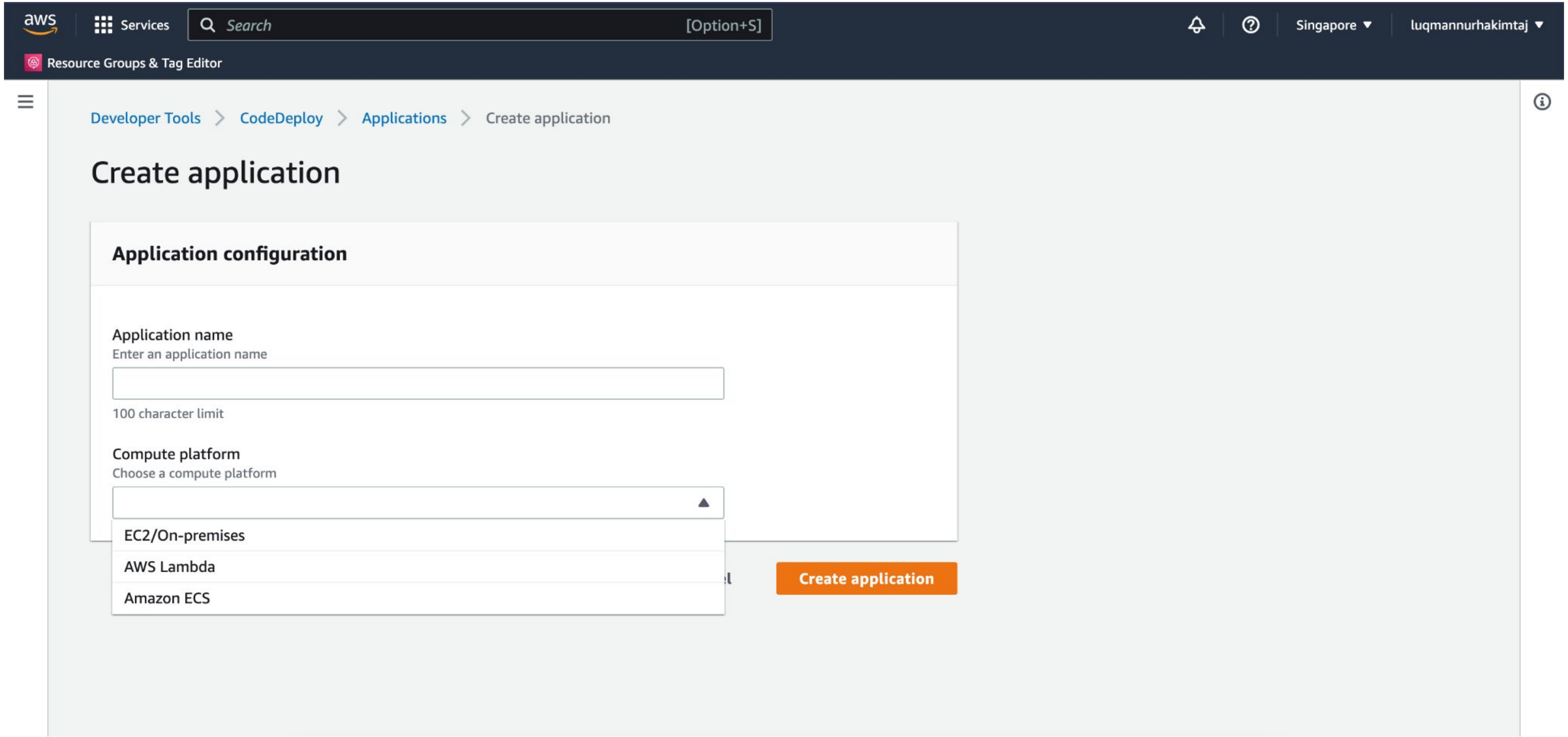
Organizations can use CodeDeploy to **automate deployments** of an application and **remove error prone manual operations** from the deployment process.

CodeDeploy can be used with a wide variety of application content including code, serverless functions, configuration files, and more.

CodeDeploy


Write code
Update application or artifact to be deployed
AWS CodeDeploy
Automate deployments for developers to securely and quickly release new features
Configure deployment
Orchestrate deployment
Monitor and roll back if necessary
Amazon EC2
Amazon ECS
AWS Lambda
AWS Fargate
On-premises servers
Updated application is now live in desired location

CodeDeploy



Developer Tools > CodeDeploy > Applications > Create application

Create application

Application configuration

Application name

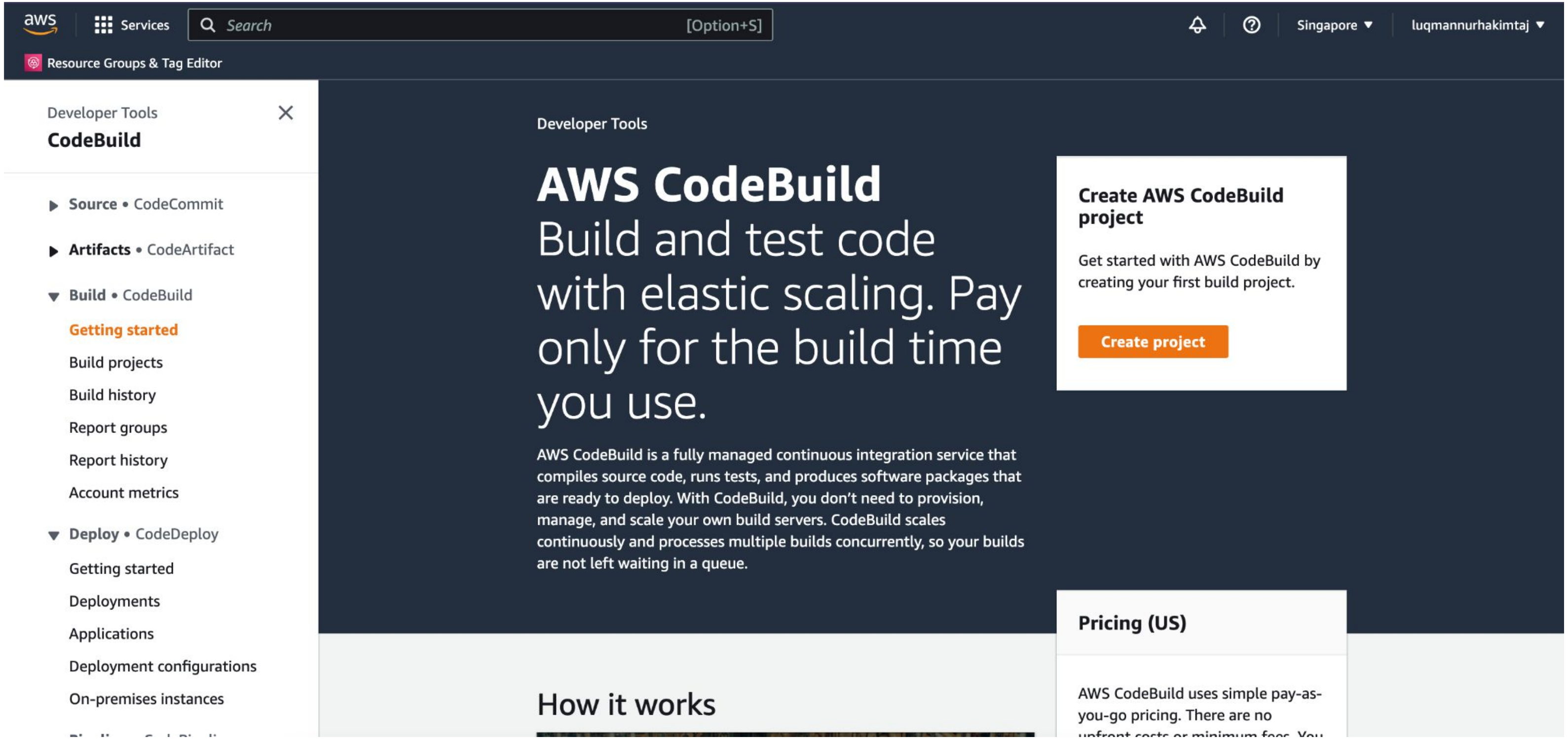
Enter an application name

100 character limit

Compute platform

Choose a compute platform

Code Series



Developer Tools

AWS CodeBuild

Build and test code with elastic scaling. Pay only for the build time you use.

AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy. With CodeBuild, you don't need to provision, manage, and scale your own build servers. CodeBuild scales continuously and processes multiple builds concurrently, so your builds are not left waiting in a queue.

Create AWS CodeBuild project

Get started with AWS CodeBuild by creating your first build project.

Create project

How it works

Pricing (US)

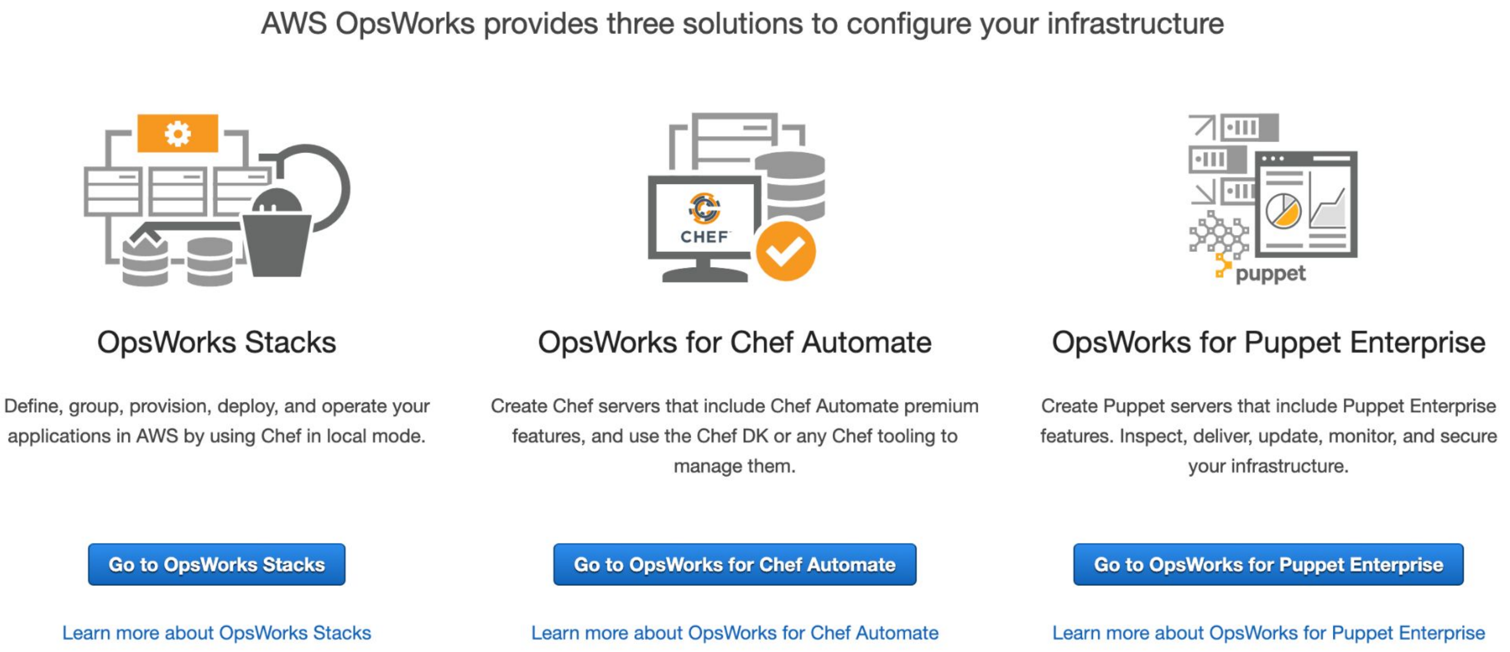
AWS CodeBuild uses simple pay-as-you-go pricing. There are no

OpsWork

OpsWorks is a **configuration management service** that **enables customers to construct, manage, and operate a wide variety of application architectures**, from simple web applications to highly complex custom applications.

Organizations **deploying applications** with OpsWorks use the automation platforms **Chef** or **Puppet** to manage key operational activities like server provisioning, software configurations, package installations, database setups, scaling, and code deployments.

OpsWork


CHEF
puppet
Go to OpsWorks Stacks
Go to OpsWorks for Chef Automate
Go to OpsWorks for Puppet Enterprise

AWS OpsWorks provides three solutions to configure your infrastructure

OpsWorks Stacks

Define, group, provision, deploy, and operate your applications in AWS by using Chef in local mode.

Learn more about OpsWorks Stacks

OpsWorks for Chef Automate

Create Chef servers that include Chef Automate premium features, and use the Chef DK or any Chef tooling to manage them.

Learn more about OpsWorks for Chef Automate

OpsWorks for Puppet Enterprise

Create Puppet servers that include Puppet Enterprise features. Inspect, deliver, update, monitor, and secure your infrastructure.

Learn more about OpsWorks for Puppet Enterprise

Activity

Question	**Answer**
What is your use case	*Answer here*
What is your Deployment Services that you choose	*Answer here*
What is benefit of that Deployment Services	*Answer here*
What is the second option that you will choose	*Answer here*

Deployment Strategies

Deployment Methodologies

- Basic Deployment
- Rolling Update/ Deployment
- Blue-Green Deployment
- Canary Deployment

Credits:

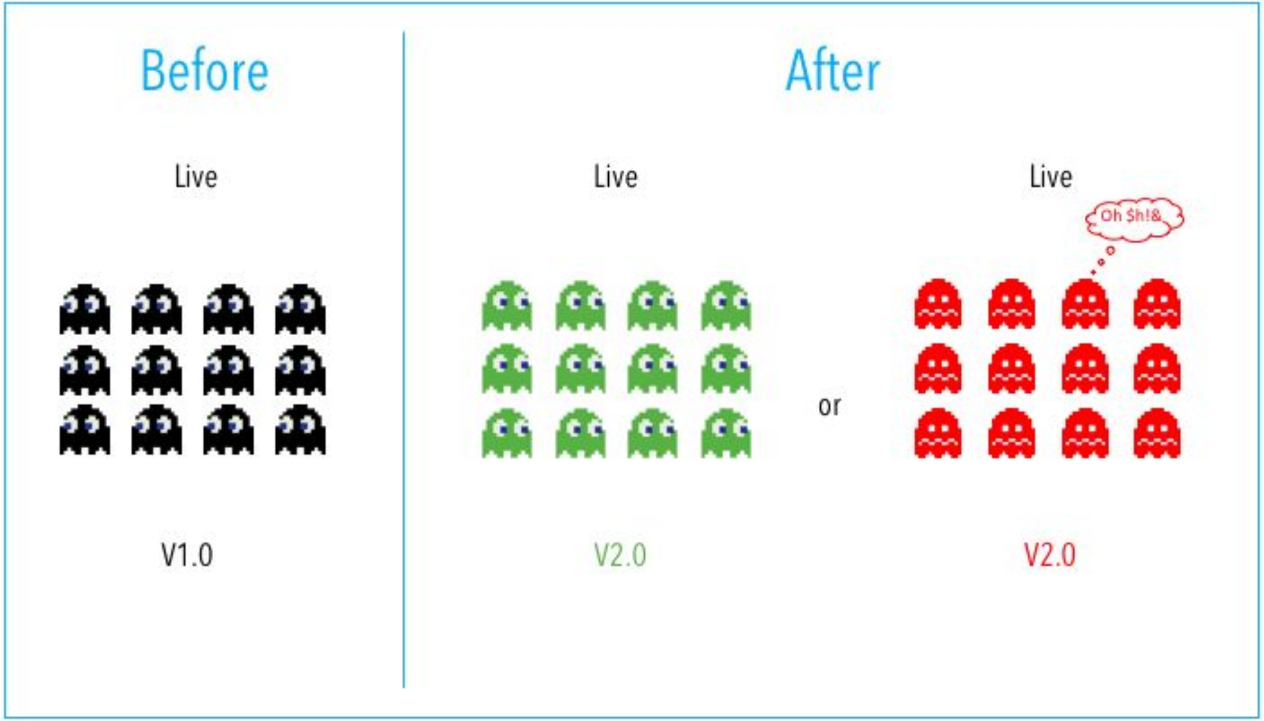
https://www.harness.io/blog/blue-green-canary-deployment-strategies

Basic Deployment

In a basic deployment, **all nodes** within a target environment are **updated at the same time** with a **new service or artifact version.**

Because of this, basic deployments are not outage-proof and they slow down rollback processes or strategies. Of all the deployment strategies shared, it is the riskiest.

Basic Deployment



Basic Deployment

Pros:

The benefits of this strategy are that it is **simple, fast, and cheap**.

Use this strategy if 1) your application service is **not business, mission, or revenue-critical**, or 2) your deployment is to a lower environment, during off-hours, or **with a service that is not in use**.

Basic Deployment

Cons:

Of all the deployment strategies shared, it is the **riskiest** and **does not fall into best practices.**

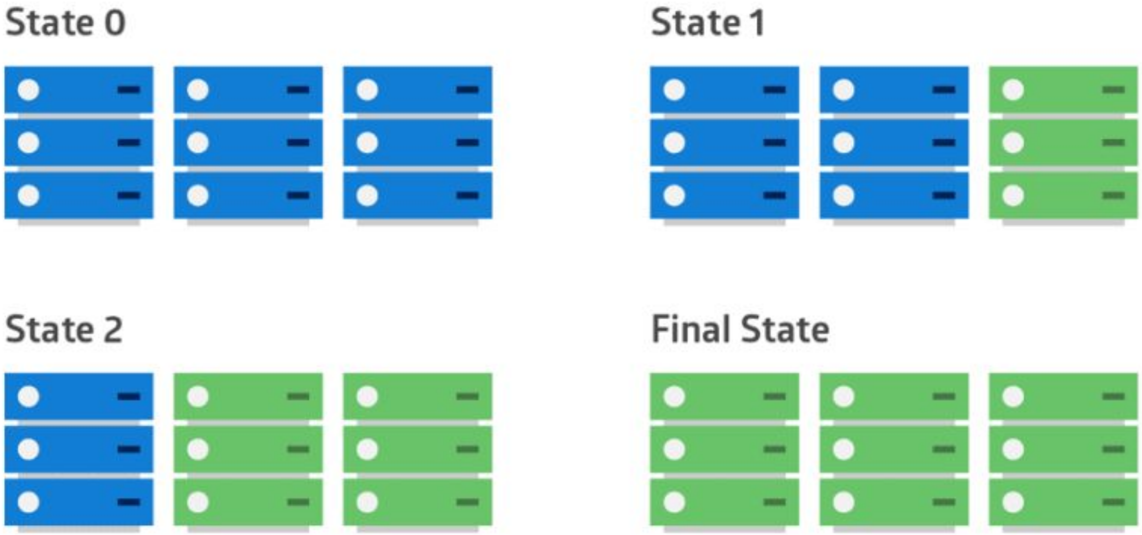
Basic deployments are **not outage-proof** and do not provide for easy rollbacks.

Rolling Deployment

A rolling deployment is a deployment strategy that updates running instances of an application with the new release.

All nodes in a target environment are **incrementally updated** with the service or artifact version in **integer N batches.**

Rolling Deployment



Rolling Deployment

Pros:

The benefits of a rolling deployment are that it is relatively simple to roll back, **less risky than a basic deployment**, and the implementation is **simple**.

Rolling Deployment

Cons:

Since nodes are updated in batches, rolling deployments require services to **support both new and old versions of an artifact**.

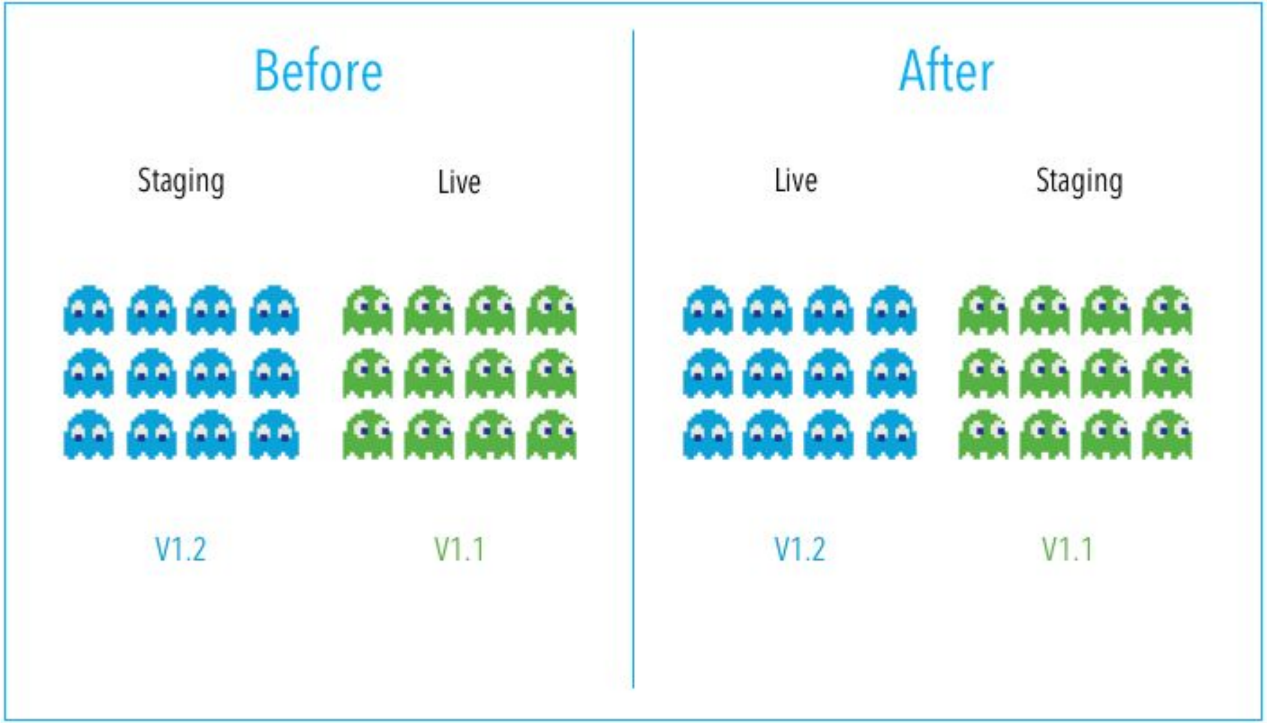
Verification of an application deployment at every incremental change also **makes this deployment slow.**

Blue-Green Deployment

Blue-green deployment starts by having the **original environment plus a duplicate environment**. This enables you to **preserve the old environment** while deploying the new application simultaneously.

Once the new application is deployed, make sure that everything runs properly. When you've determined that the **new environment is free of issues**, you can switch back to the new environment and then end the old environment.

Blue-Green Deployment



Blue-Green Deployment

Pros:

One of the benefits of the blue-green deployment is that it is **simple, fast, well-understood, and easy to implement**.

Rollback is also **straightforward**, because you can simply flip traffic back to the old environment in case of any issues.

Blue-green deployments are therefore not as risky compared to other deployment strategies.

Blue-Green Deployment

Cons:

Replicating a production environment can be complex and **expensive**, especially when working with microservices.

Quality assurance and user acceptance testing **may not identify all of the anomalies or regressions either,** and so shifting all user traffic at once can present risks.

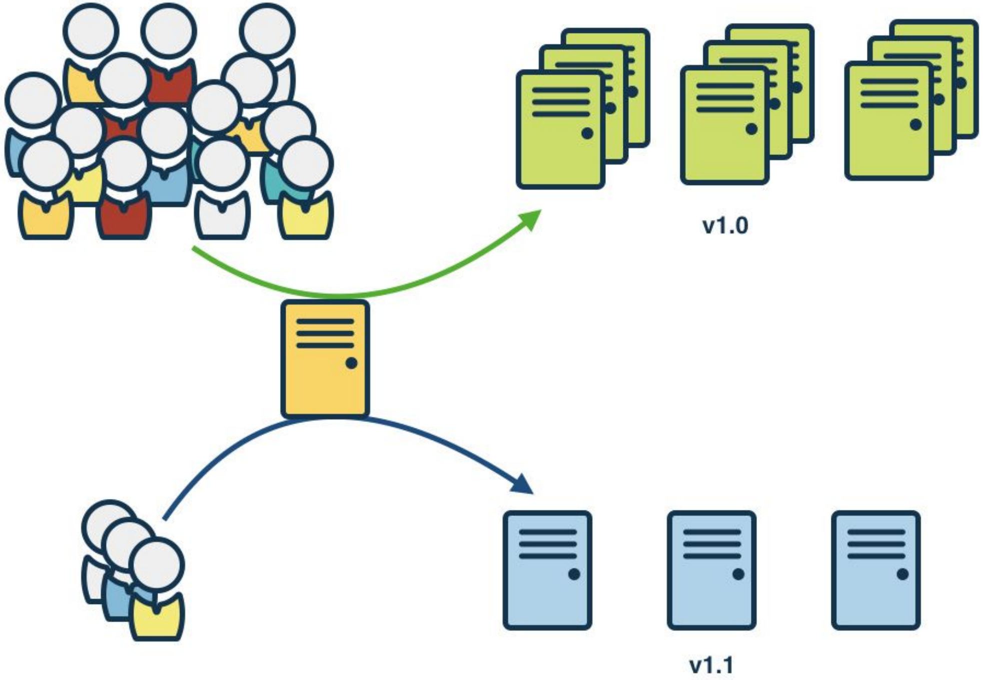
Canary Deployment

A canary deployment is a deployment strategy that **releases an application or service incrementally to a subset of users.**

All infrastructure in a target environment is updated in small phases (e.g: 2%, 25%, 75%, 100%).

A canary release is the **lowest risk-prone,** compared to all other deployment strategies, because of this control.

Canary Deployment



Canary Deployment

Pros:

Canary deployments allow organizations to **test in production with real users** and use cases and **compare different service versions** side by side.

It’s **cheaper** than a blue-green deployment because it does not require two production environments.

And finally, it is **fast and safe to trigger a rollback** to a previous version of an application.

Canary Deployment

Cons:

Drawbacks to canary deployments involve **testing** in production and the implementations needed.

Scripting a canary release can be complex: manual verification or testing can take time, and the required monitoring and instrumentation for testing in production may involve additional research.

Activity

Question	**Answer**
What is your use case	*Answer here*
What is your Deployment Strategies that you choose	*Answer here*
What is benefit of that Deployment Strategies	*Answer here*
What is the second option that you will choose	*Answer here*

Activity

Learner:

- Clean up AWS.
- Remove/delete/terminate all service/ resources that created.

Instructor

- Clean up AWS.
- Remove/delete/terminate all service/ resources that created.
- Check the AWS account after learner clean up.

What’s Next?