edureka!

Microsoft Azure DevOps Solution Certification (AZ-400)

COURSE OUTLINE

Azure AZ-400

MODULE 1: Introduction to Azure DevOps

MODULE 2: Implementing Continuous Integration

MODULE 3: Build Containers with Azure DevOps

MODULE 4: Designing a Dependency Management Strategy and Managing

Artifact Versioning

MODULE 5: Setting up Release Management Workflow

MODULE 6: Implementing Deployment Models and Services

MODULE 7: Implement and Optimize Continuous Feedback Mechanism

MODULE 8: Azure Tools: Infrastructure and Configuration, and Third-Party Tools

MODULE 9: Implementing Compliance and Security

MODULE 10: Azure Case Studies

edureka!

Setting up Release Management Workflow

Topics

Following are the topics covered in this module:

- Continuous Delivery
- Azure Release Pipeline
- Deployment Patterns
- Release Tools
- Tasks and Templates
- Integrating Secrets with Release Pipeline
- Automating Health Inspection

Objectives

After completing this module, you should be able to:

- Configure CI/CD pipeline as code with YAML
- Use secrets in the pipeline with Azure Key Vault
- Set up and run functional tests
- Release deployment control using Azure Monitor as release gate
- Create a release dashboard to share information, monitor progress, and trends

Continuous Delivery

edureka!

Cisco’s Outsourced Project

Aesir co. has received a project from cisco. To discuss this, Tim, a team leader, has called for a meeting with Mark, an Azure DevOps Engineer

Project Discussion

This is cisco’s Project. They want to update the products' feedback in the database and accordingly deploy the code

Tim
The Lead

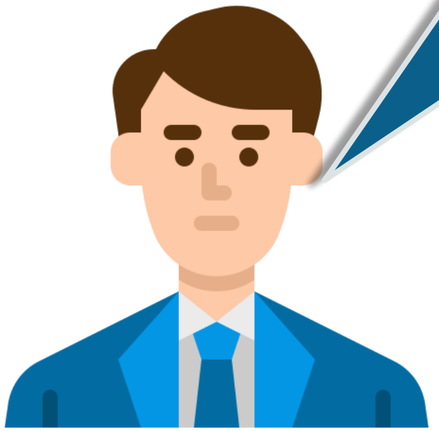
Okay!

Mark
Azure DevOps Engineer

But there are two issues here. I was hoping you could figure out a way to solve these

The Issue

- First, this data must be restricted and accessible only by their application
- Secondly, the admin for Cisco should be able to monitor the progress in their business and the trends in the feedback



Tim
The Lead

Got it. I will discuss this with the team and get back to you

Mark
Azure DevOps Engineer

The Research

From my research, it is clear that we have to do the following:

- Use Azure Key Vault in the database connection
- Create a release dashboard to share information and monitor progress and trends

Mark
Azure DevOps Engineer

The Solution

To store and update the products' feedback in the database and deploy the code:

- Mark configured CI/CD Pipeline as code with YAML and used secrets in the pipeline with Azure Key Vault
- After that, you then set up and run the functional test on the same

To help the admin monitor the business and trends in the feedback:

- Released the deployment control using Azure monitor as a release gate
- Then created a release dashboard where information can be shared and monitored by cisco's admin

Project Submission

Congratulations! Great work, guys! We have successfully delivered the project, and cisco is pleased with our work!

Tim
The Lead

Continuous Delivery

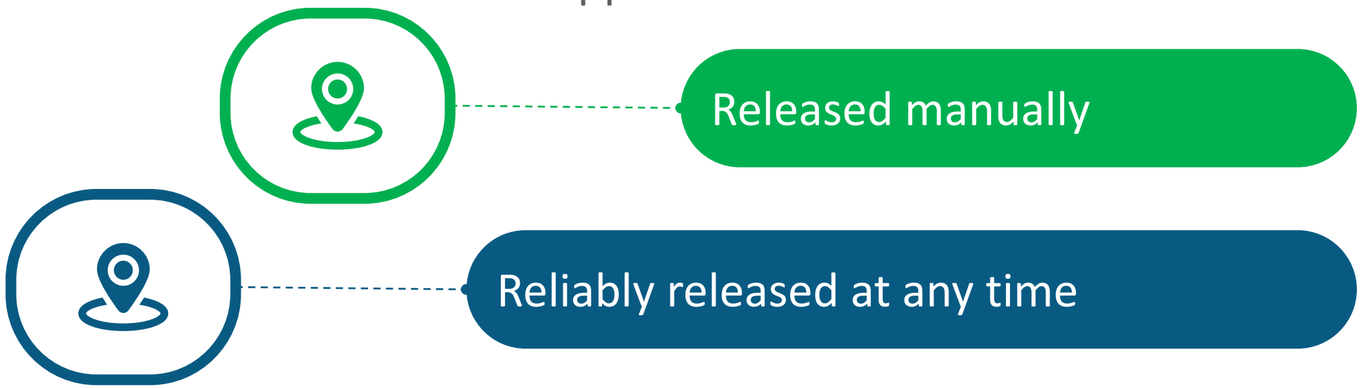
edureka!

What is Continuous Delivery?

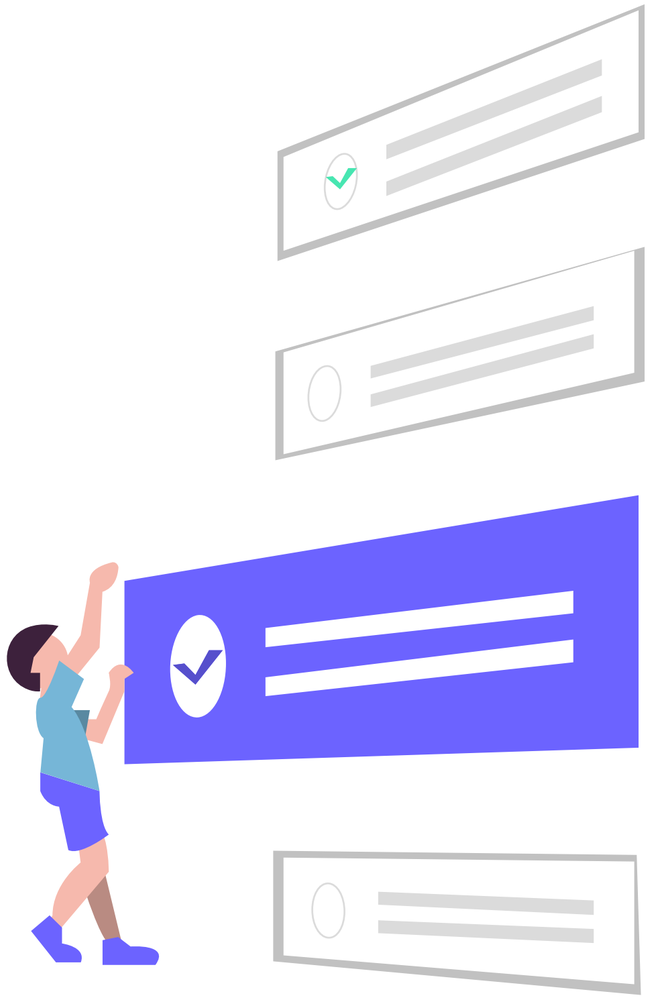
Continuous delivery provides a consistent way to continuously test, deploy, and monitor application each time the code is checked-in

- It is an approach where software is produced in short cycles
- This helps in validating the change introduced frequently by developers

This approach ensures that the software can be:



Need for Continuous Delivery



Continuous delivery provides a consistent way to continuously

- test
- deploy and
- monitor

Purpose of Continuous Delivery

Build, test, and release software frequently

Reduces cost, time, and risk of delivering the changes

Benefits of Continuous Delivery

01. Higher quality and faster ROI
02. Early feedback in case of any issue
03. Better planning and collaboration
04. Less production issues
05. Predictable release
06. Deploy during business hours
07. Faster response to market changes
08. Change is delivered quickly without significant delay
09. Fast, repeatable, and configurable deployments

edureka!

When to Implement Continuous Delivery?

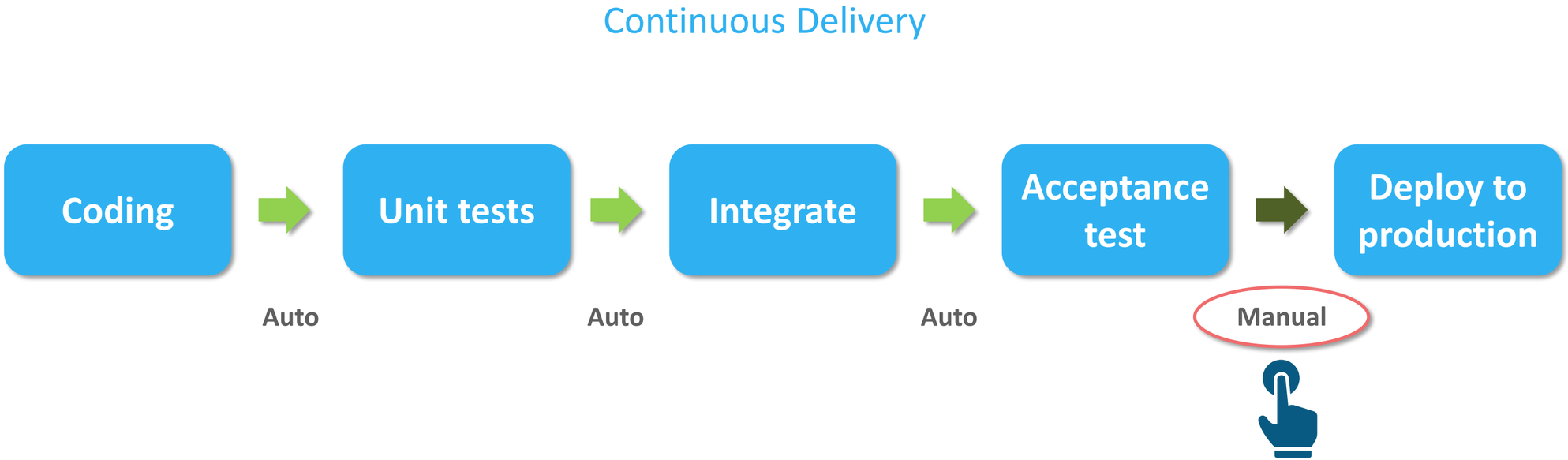
When Continuous Integration and extensive automation are available through all possible parts of the delivery process

When software is deployable throughout its lifecycle

When push-button deployments can be performed on any version of the software to any environment on demand

edureka!

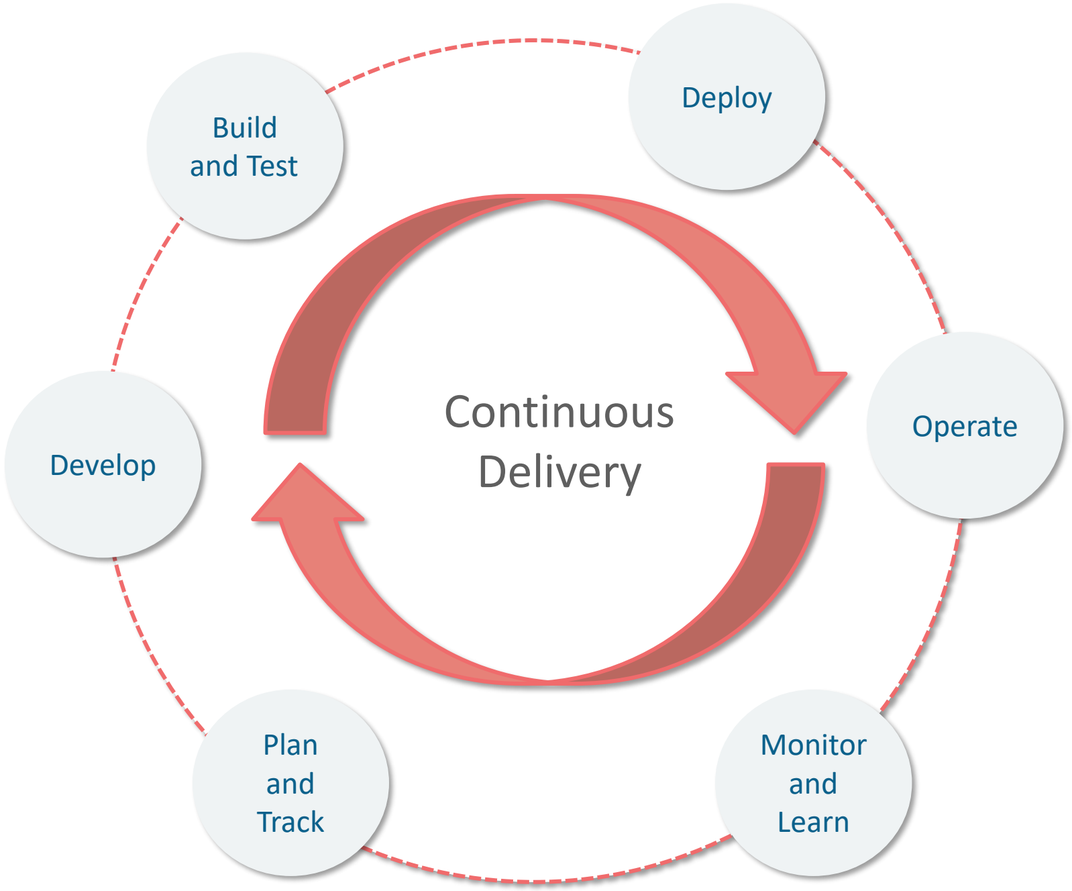
Stages of Continuous Delivery



In the above image, the stages of Continuous Delivery are shown

Continuous Integration is a prerequisite for Continuous Delivery

Continuous Delivery Lifecycle



In the above image, the cycle of Continuous Delivery is shown

Continuous Delivery vs. Continuous Deployment

Continuous Deployment	Continuous Delivery
☑ Every change goes through the pipeline and automatically gets deployed into production	☑ This means that developers may or may not do frequent deployments
☑ Results in many production deployments every day	☑ The decision to do the deployment is decided through the deployment strategy of the business
☑ This is an automated process from end to end	☑ It is required before going ahead with continuous deployment

edureka!

Azure Release Pipeline

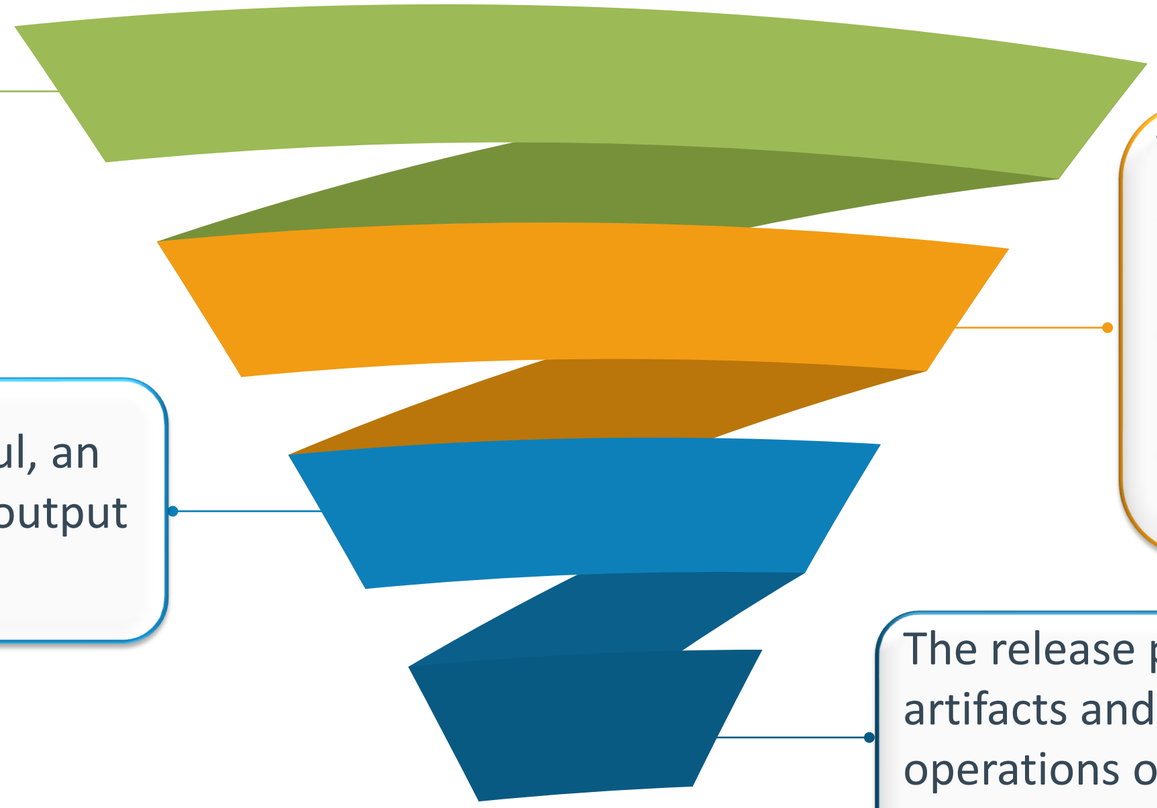
edureka!

Azure Release Pipeline

- Helps the team continuously deliver software to customers at a faster pace and with lower risk
- Automates the testing and delivery of software that can be automated from development to production deployment
- Provides approval for a process to move from one stage to another stage

Building a High-Quality Release Pipeline

The continuous deployment contains a trigger to start the process and at least one stage or deployment phase

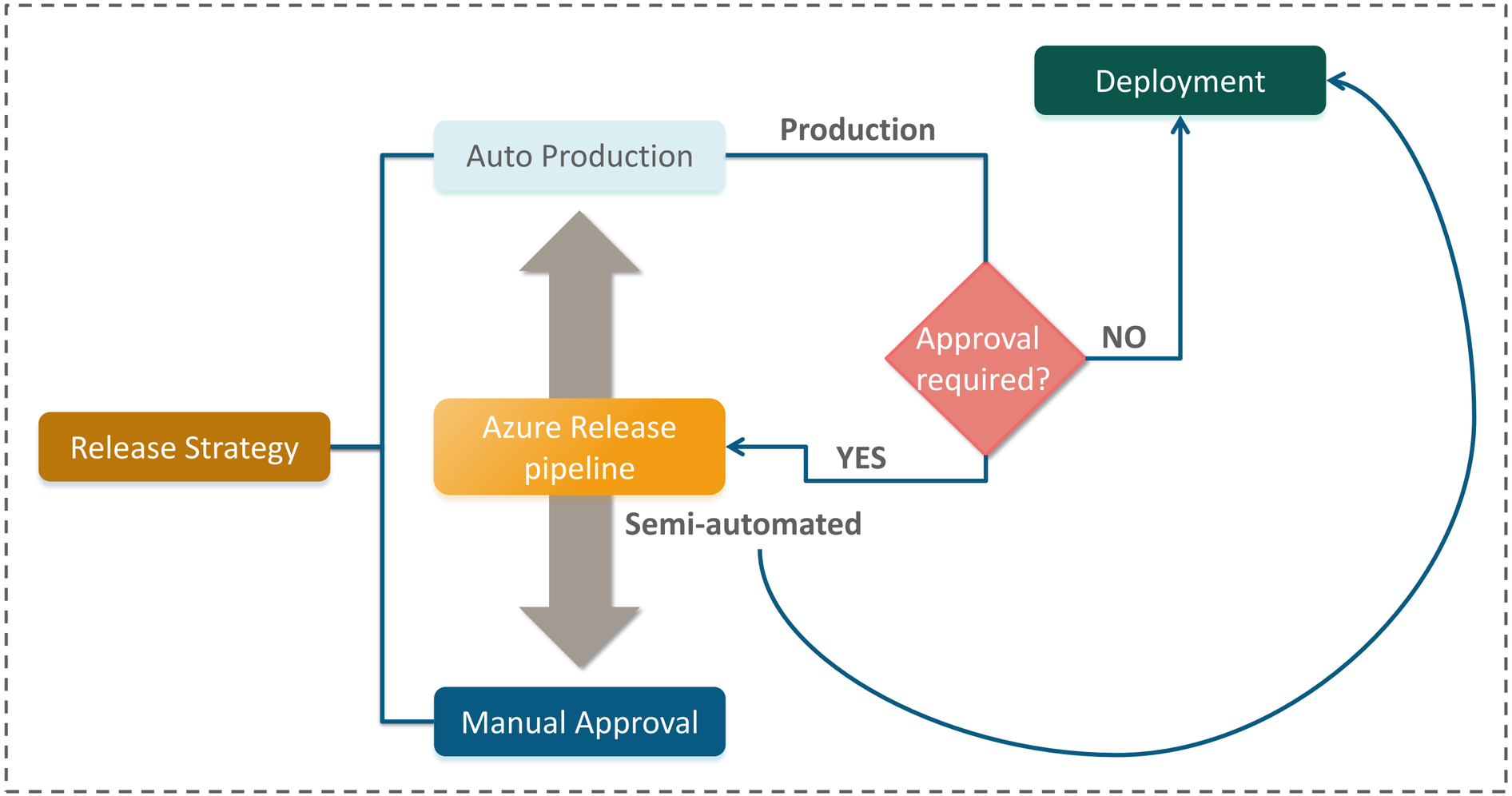


The trigger can be an artifact creation, resulting in the build output of the CI pipeline (developer code check-in -> build is triggered)

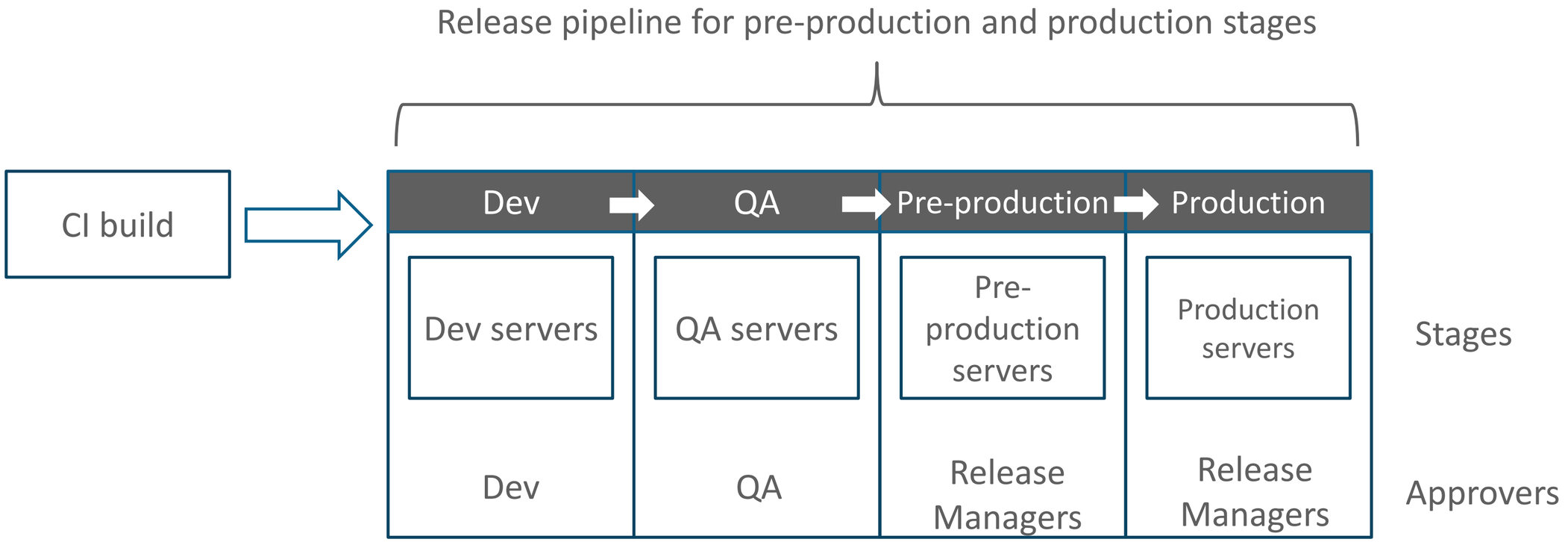
After the build is successful, an artifact (exe, dll, or other output packages) gets created

The release pipeline picks the artifacts and does the subsequent operations of taking the artifact and deploying it to production

Release Strategy Recommendations



Picking Artifacts for Development



Steps to Release Pipeline

Pre-Deployment Approval

- Azure pipeline checks whether any pre-deployment approval is required from any stakeholder
- If it is required, then it sends mail to the concerned person/approver for approval

Queue Deployment Job

- The azure pipeline then queues the deployment job to an available agent
- The agent runs the deployment tasks

Agent Selection

- An agent picks up the job
- Release pipeline contains the setting of agents which is used for agent selection

Steps to Release Pipeline (Cont.)

Download Artifact

- This is generally the starting point of the release pipeline where agent downloads the artifact specified during release pipeline creation
- The agent understands only Azure Pipelines artifacts and Jenkins artifacts

Run the Deployment Task

- The agent then runs all the deployment tasks to deploy the software to the target server

Steps to Release Pipeline (Cont.)

Progress logs

- Detailed logs for each step are created by the agent and are pushed back to Azure Pipeline

Post Deployment Approval

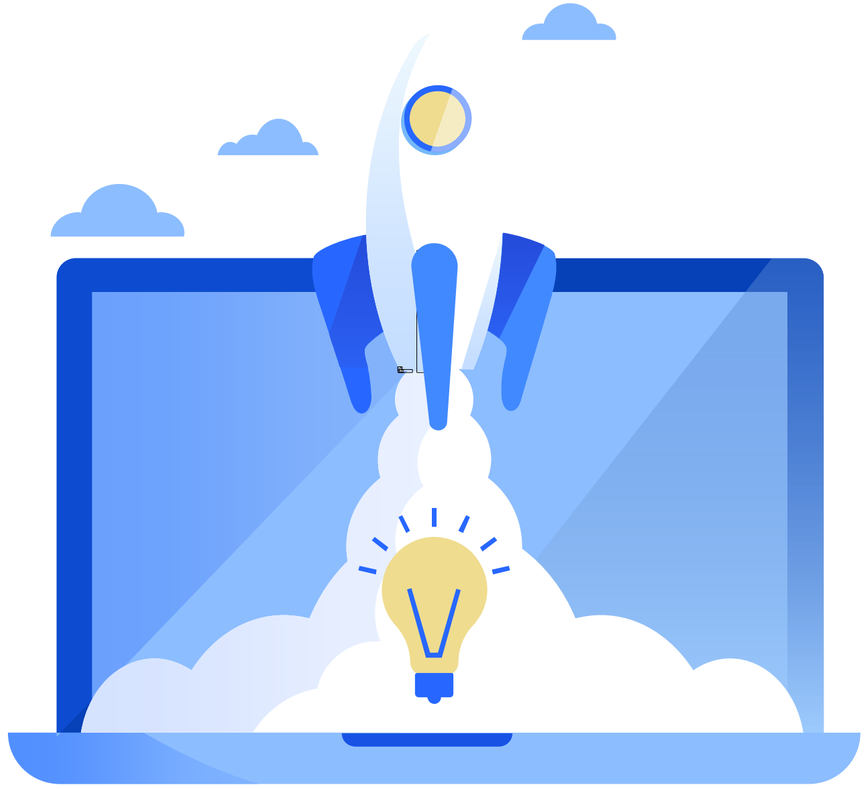
- After the deployment is complete, the Azure pipeline checks if any post-deployment approval is required
- If it is required, then the pipeline sends mail to the concerned approver

Deployment Patterns

edureka!

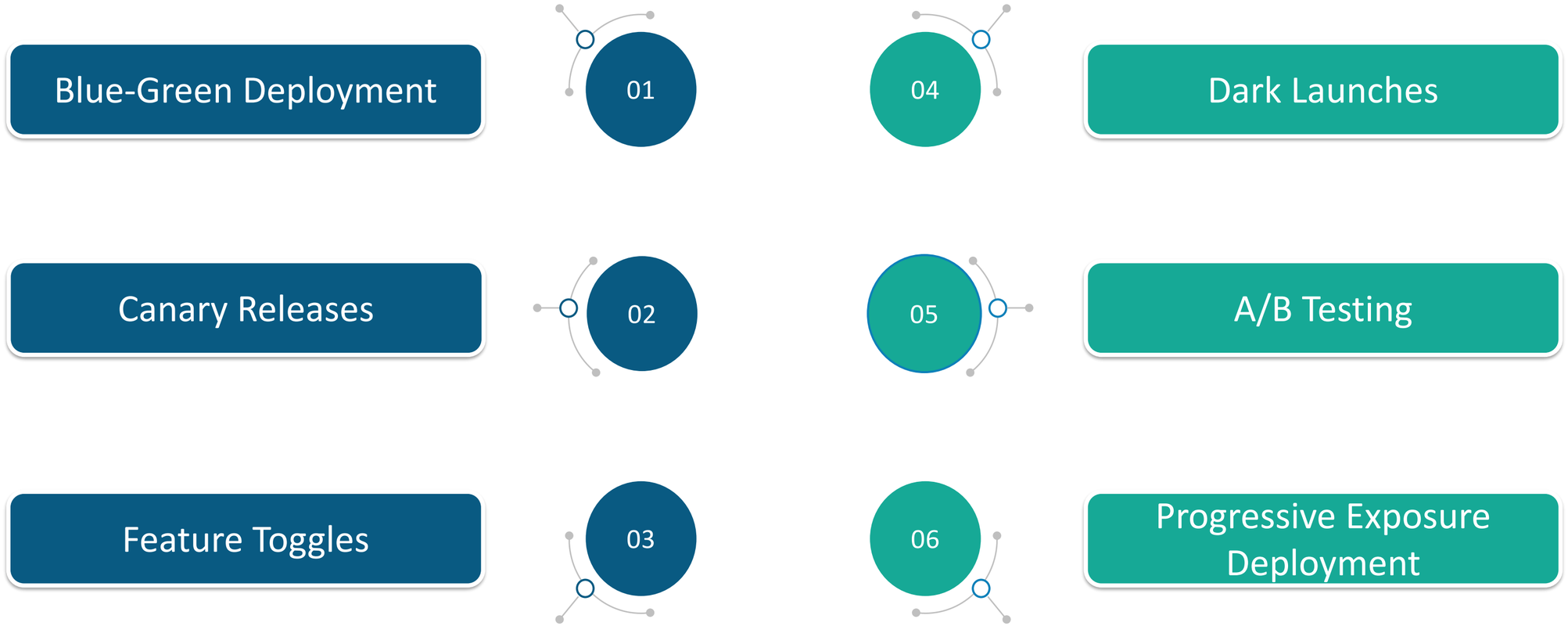
What are Deployment Patterns?

- It is an automated way of smoothly deploying the application with new application features
- The deployment pattern's objective is to roll out the deployment with accuracy and with the least downtime



edureka!

Types of Deployment Pattern



What is Blue-Green Deployment?

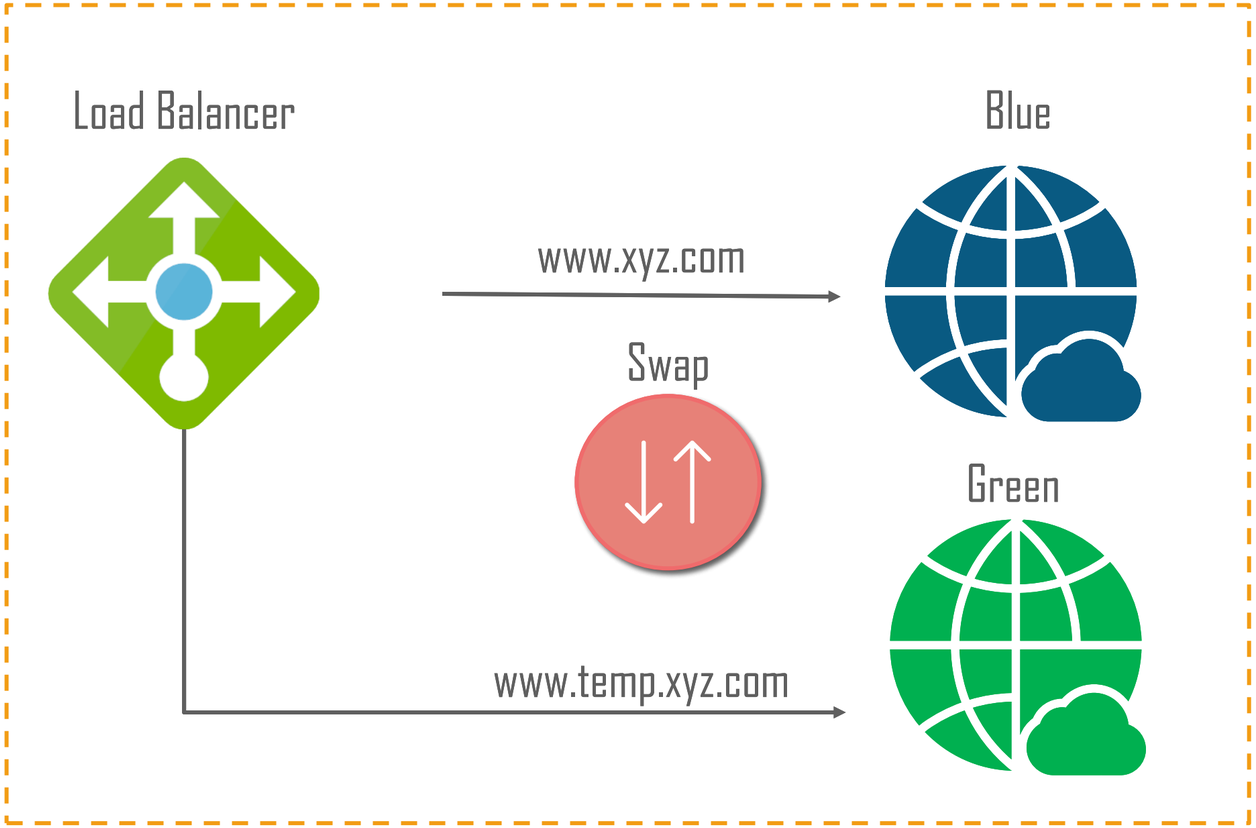
- It involves two identical deployment servers
- The new features are deployed to other servers (Green server) before the testing is done
- On successful testing, the user requests are routed to the Green server. This way, new features are exposed to the user

Blue-green deployment

Benefits of Blue-Green Deployment

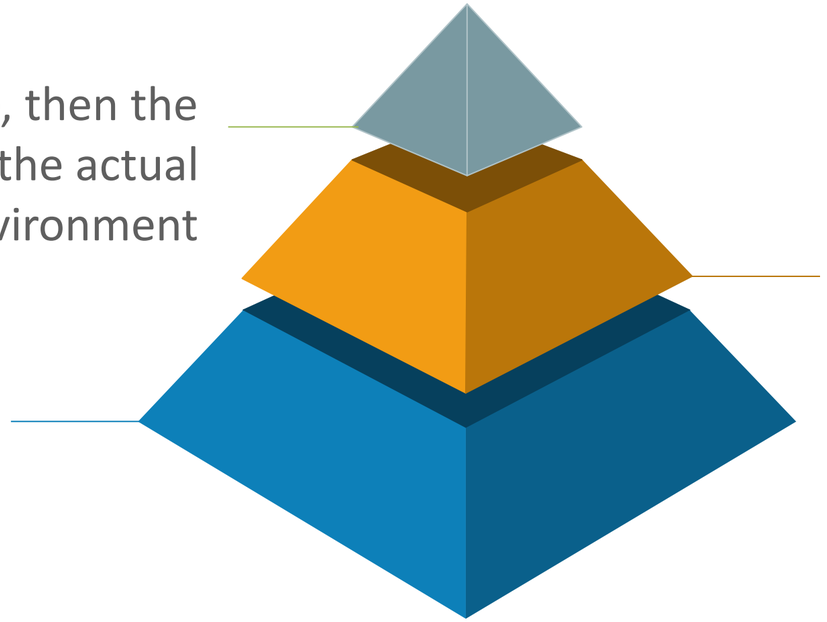
It gives us a fast way to roll back changes. In case of any issue,

- The user requests are sent back to the Blue server
- The switching of user requests is generally done through a Load Balancer



What is Canary Release?

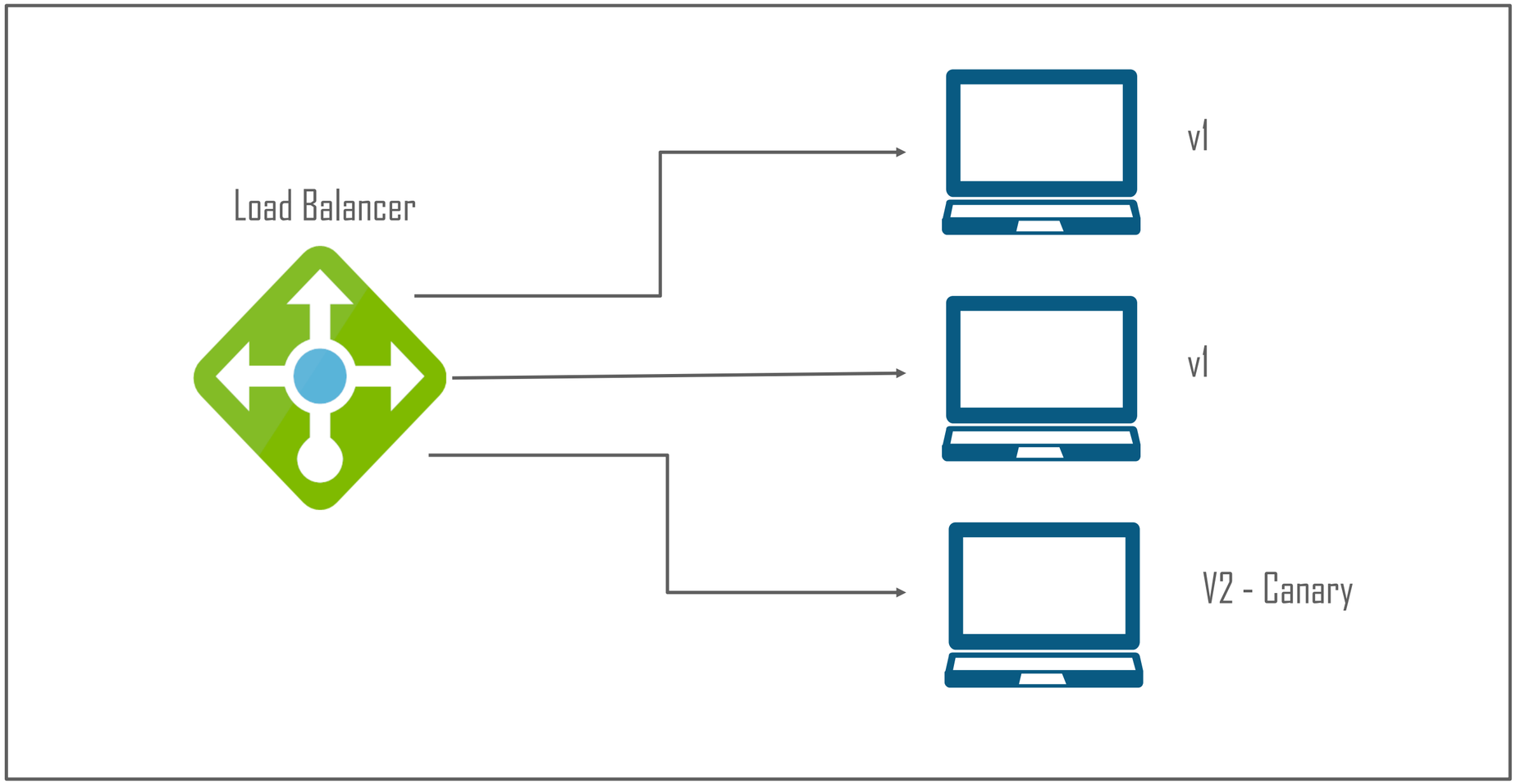
Once the solution is stable, then the solution is moved to the actual production environment



The solution is monitored for some duration. If any issue happens, then the issue is fixed

In this pattern, new features are exposed to only some users

What is Canary Release?



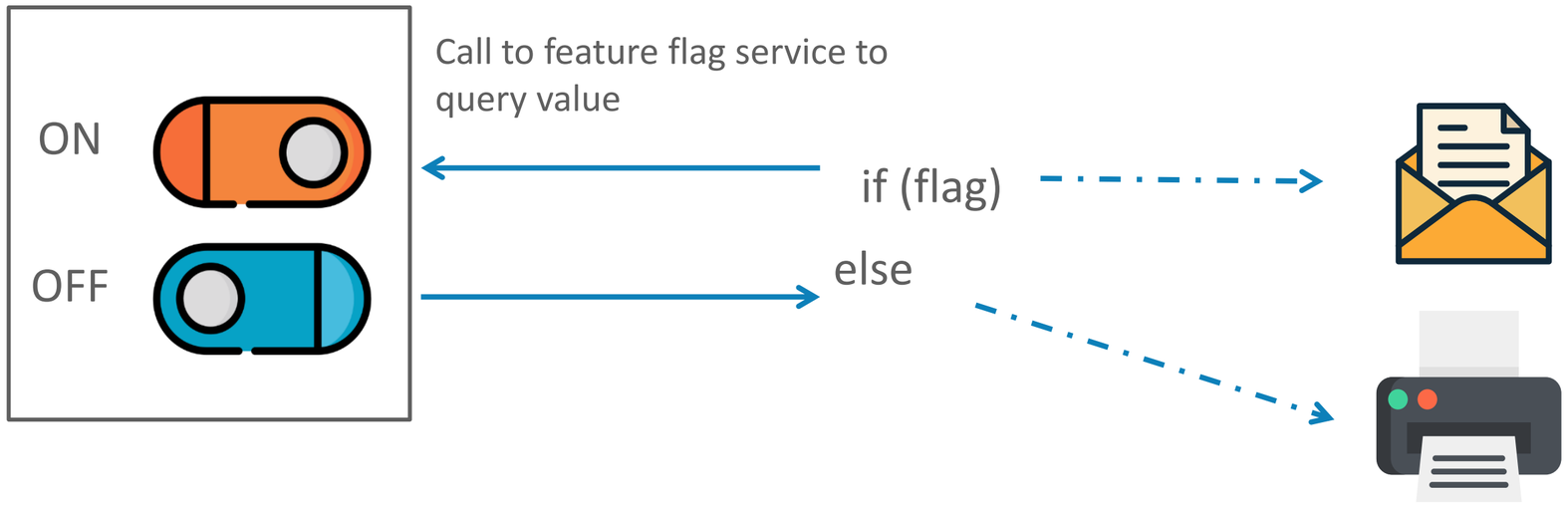
What is Feature Toggles?

- It is a kind of conditional access for users
- Initially, the new features are set to “Off” during deployment
- Then features are set to “ON” for some users
- Depending on the results monitored by the actions, this feature can be set to "ON" for all users, a subset of users, or more users

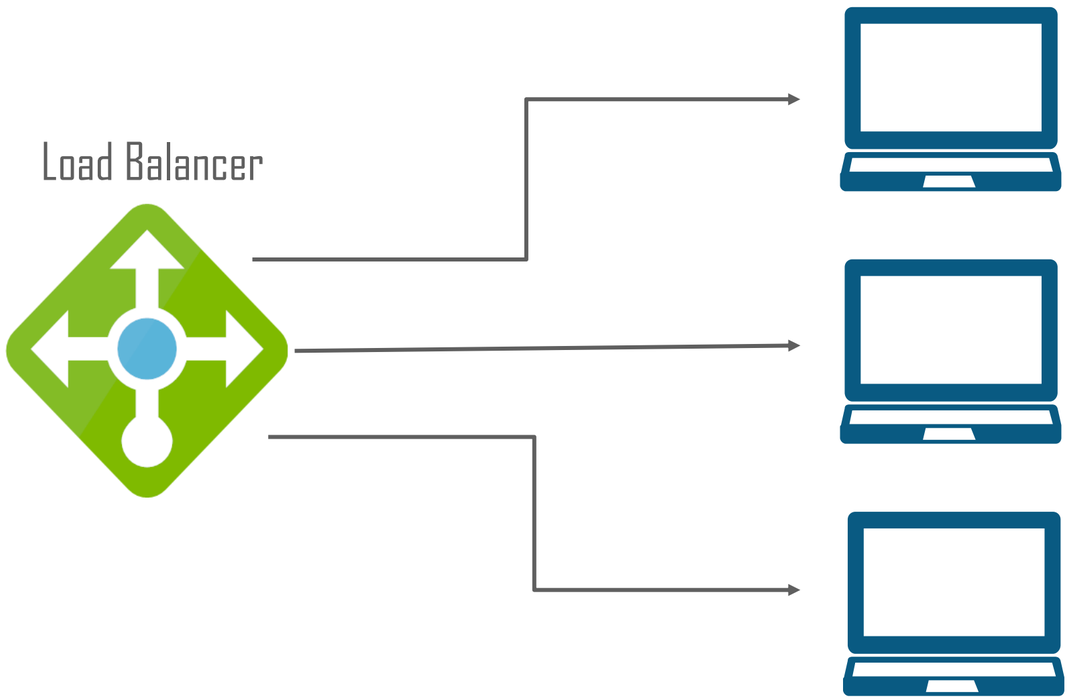
It helps roll back changes. If the new features get some error in production, then the flag can be set off and thus rolling in back.

Feature Toggles: Example

Depending on the flag, the email features for the document or printing the document will be enabled

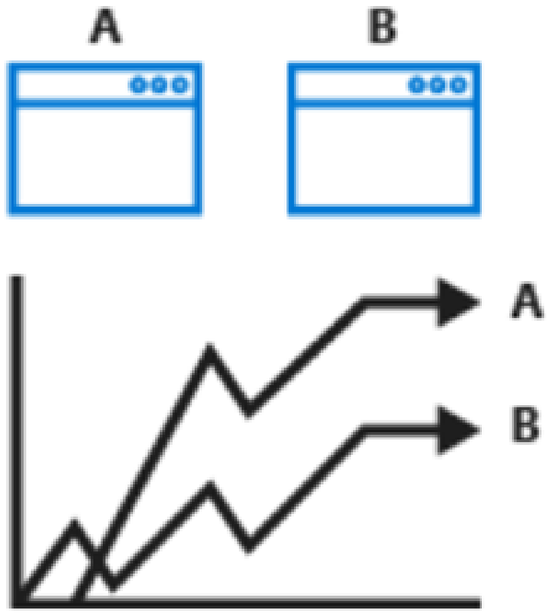


What are Dark Launches?



- Dark Launches are similar to Canary Release and Feature Toggles
- The features are exposed to only some users
- Users are not aware of these new features. That is why it is called Dark Launches

What is A/B Testing?



In this, A/B testing compares two versions of a webpage or app to determine which perform better

In these two versions of the web page, developers can send one version to half of the users and the other version to the rest of the users and use some metrics to see which layout works better for the application goals

Release Tools

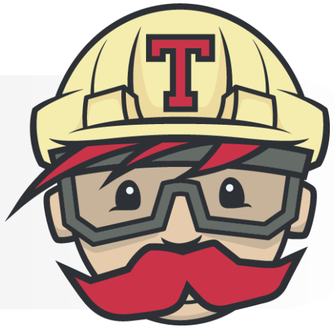
edureka!

Types of Release Tools

Jenkins

Circle CI

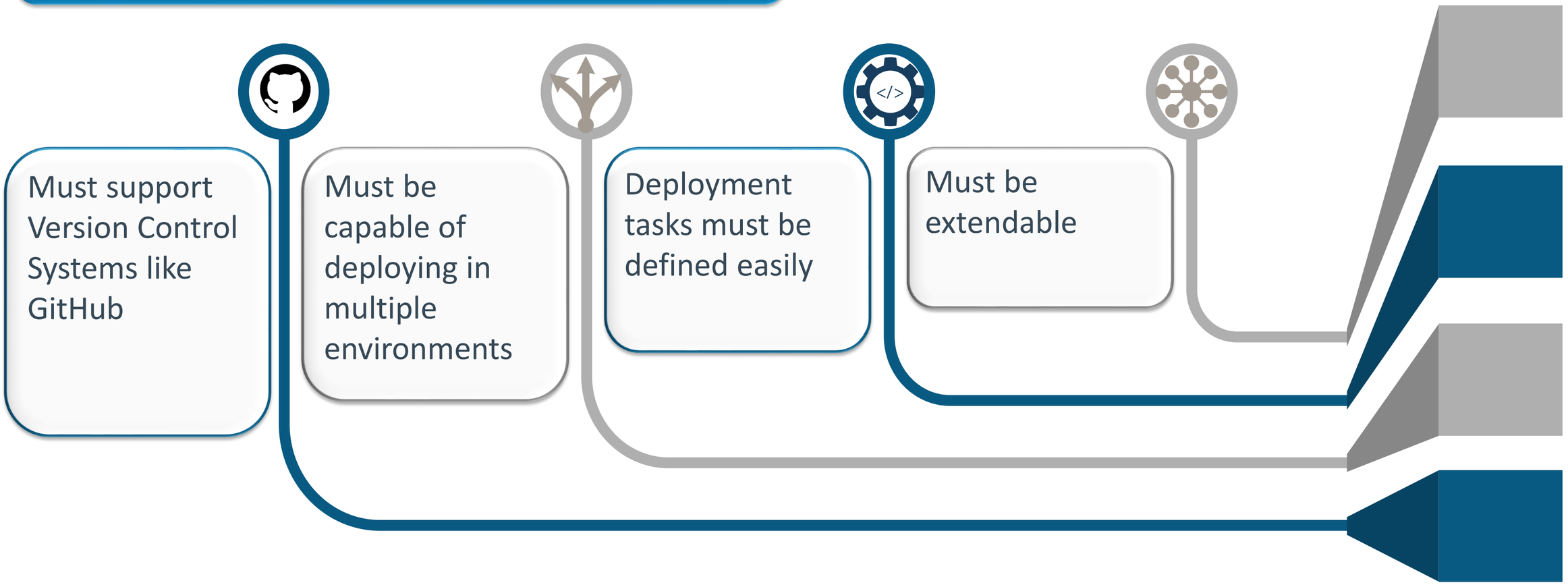
Travis CI



Azure Pipelines

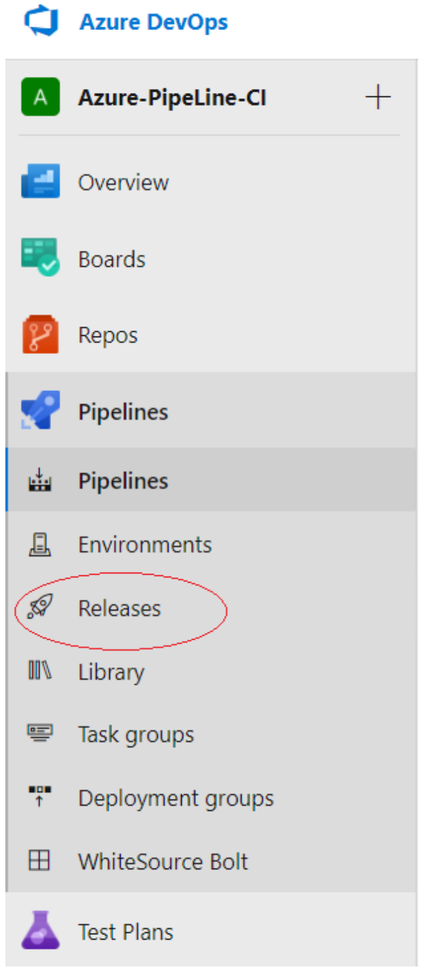
Choosing the Right Release Management Tool

Release tools need to be selected based on the following criteria:



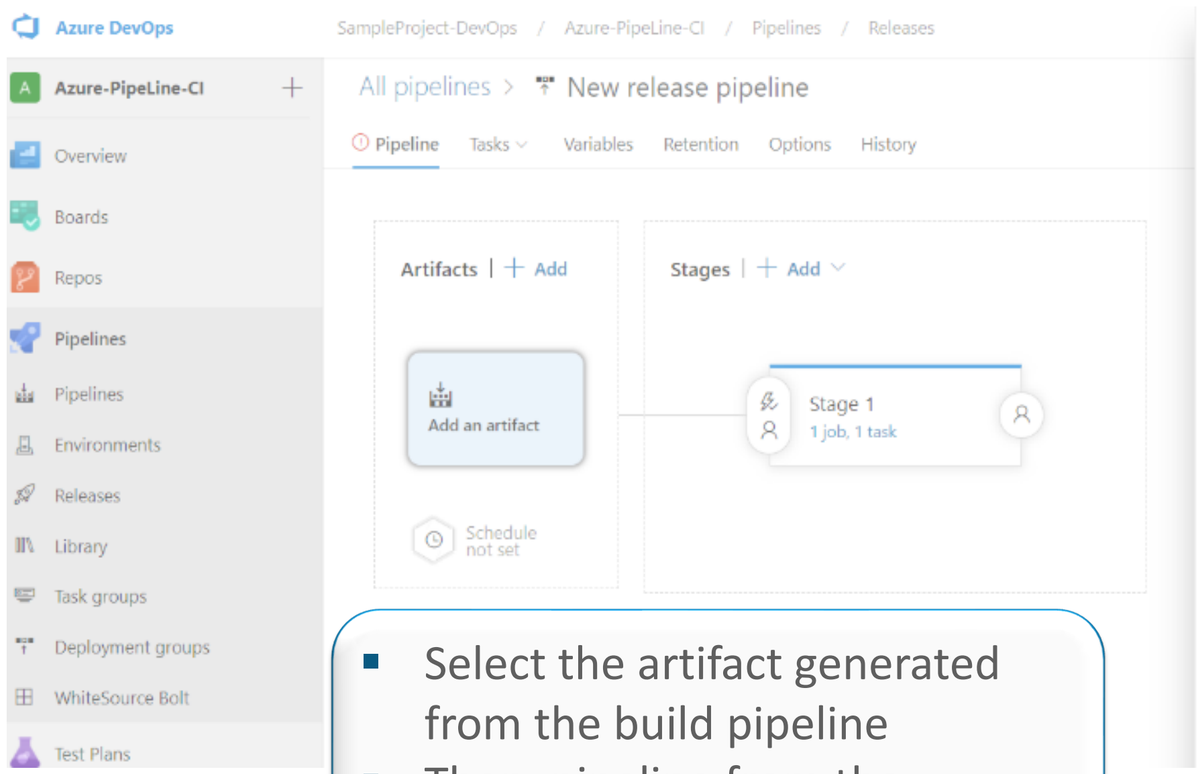
Creating a Release Pipeline

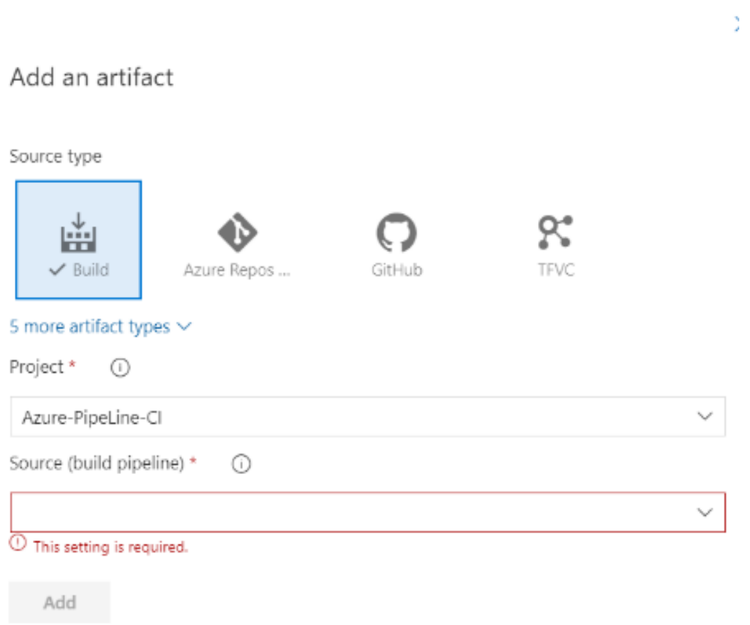
Release pipeline can be created from the release button in the Pipeline section, as shown below in image:



Creating a Release Pipeline: Stage 1

Click on Release and create a new Release pipeline as shown below.



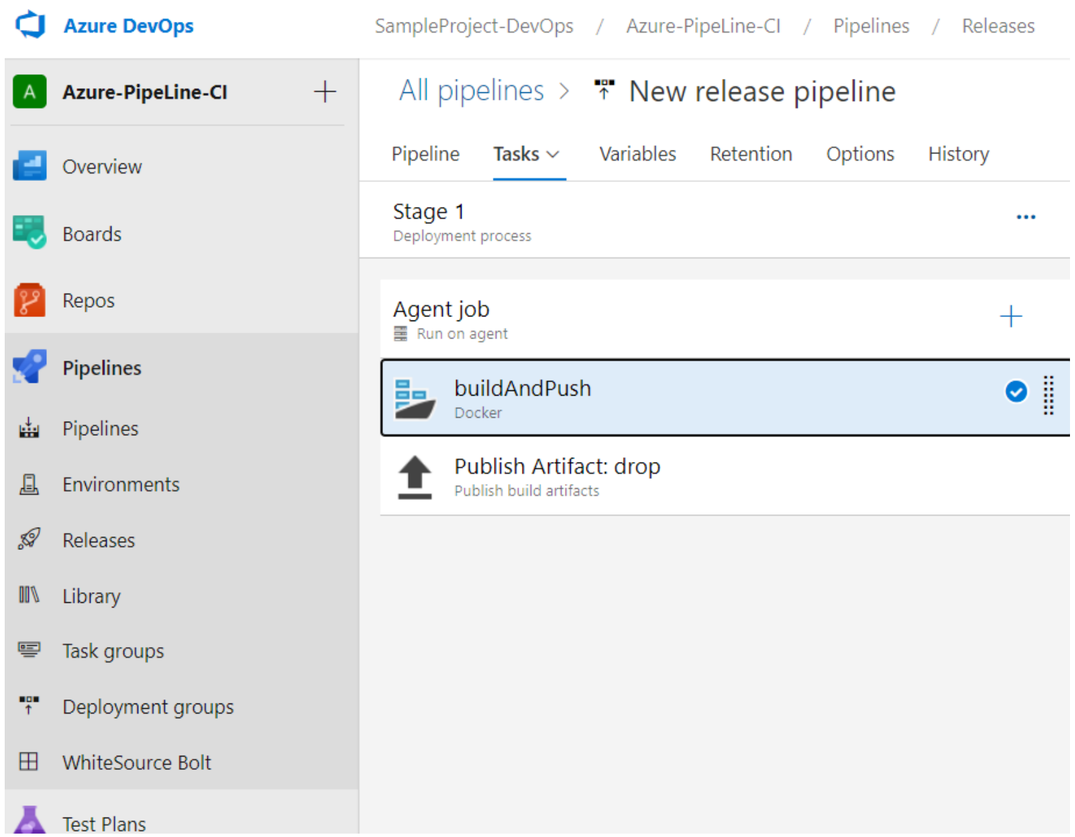


- Select the artifact generated from the build pipeline
- Then, pipeline from the dropdown menu

edureka!

Creating a Release Pipeline: Stage 2

Click on the Job link and the below screen appears.



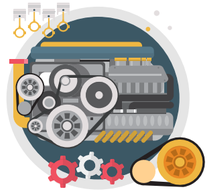
- Add the tasks by clicking on the “+” button. This way, the tasks will get added for execution

Provision and Configure Environments

Dev

QA

Test



Staging or Production

Deployment

Kubernetes Cluster

Virtual Machine

Environment

An environment is where the deployment happens

edureka!

Provision and Configure Environments: Advantages

Deployment history

- Pipeline name, and run details are captured

- Jobs from the pipeline can be viewed
- Work items and commits can be viewed from the environment

Work Items and commit traceability

Permissions

- Permission to selected users and selected pipelines that can target the environment

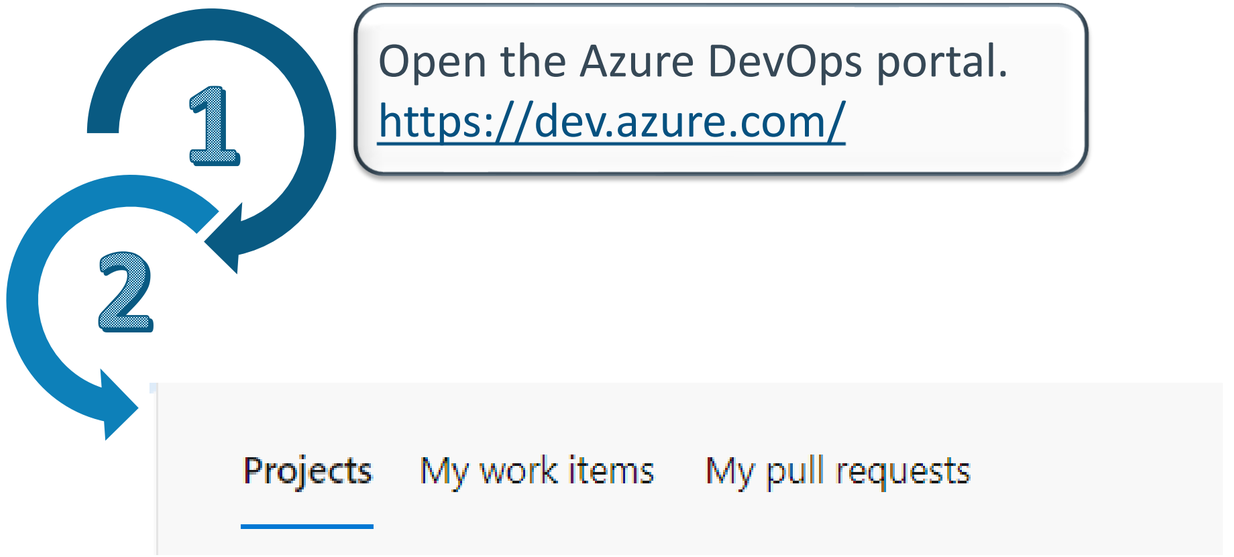
- It can be validated whether the application is functioning correctly at its desired health

Diagnose resource health

Provision and Configure Environments: Stage 1

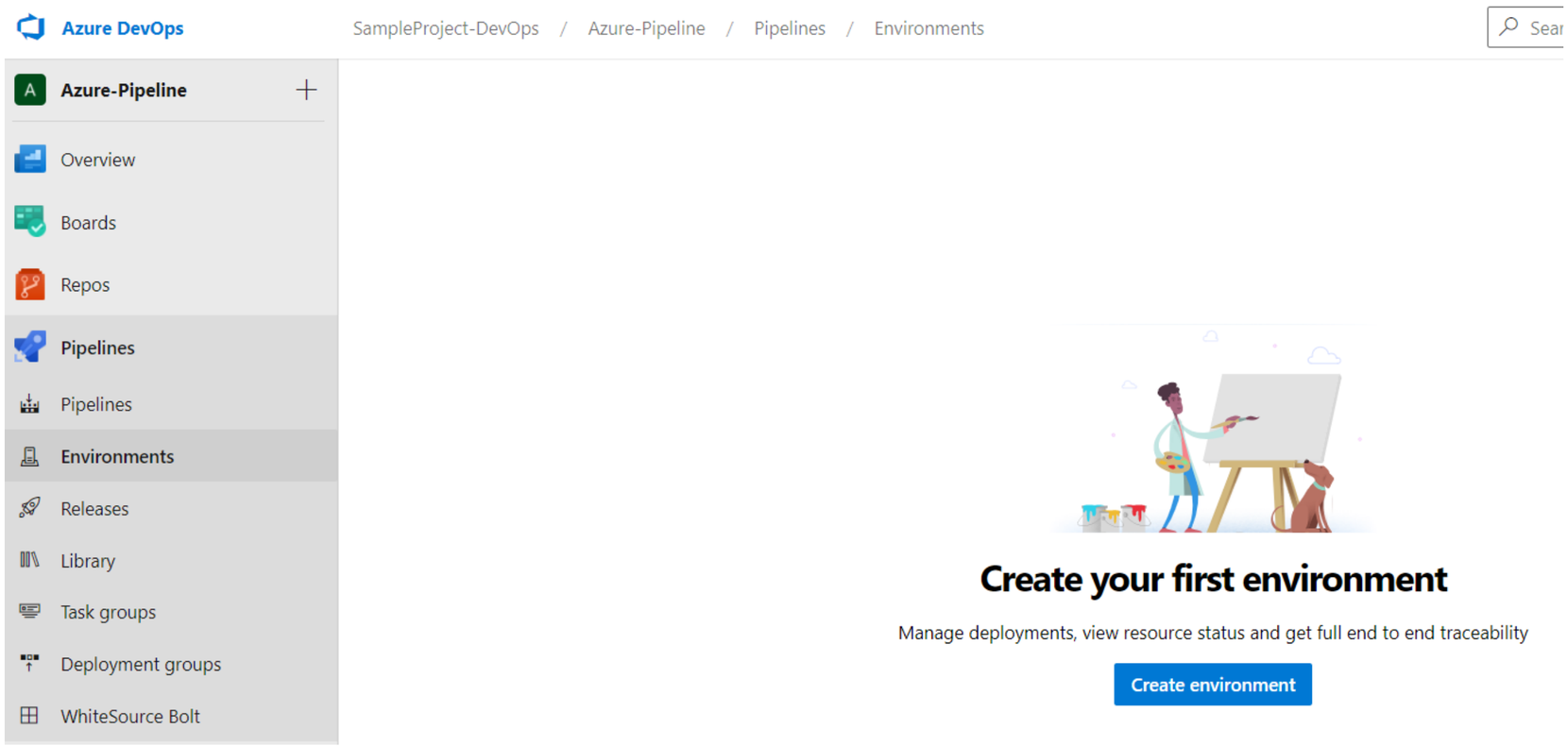
Open the Azure DevOps portal.
https://dev.azure.com/

Select the project where a user wants to create the environment. It can be seen under the Projects Tab



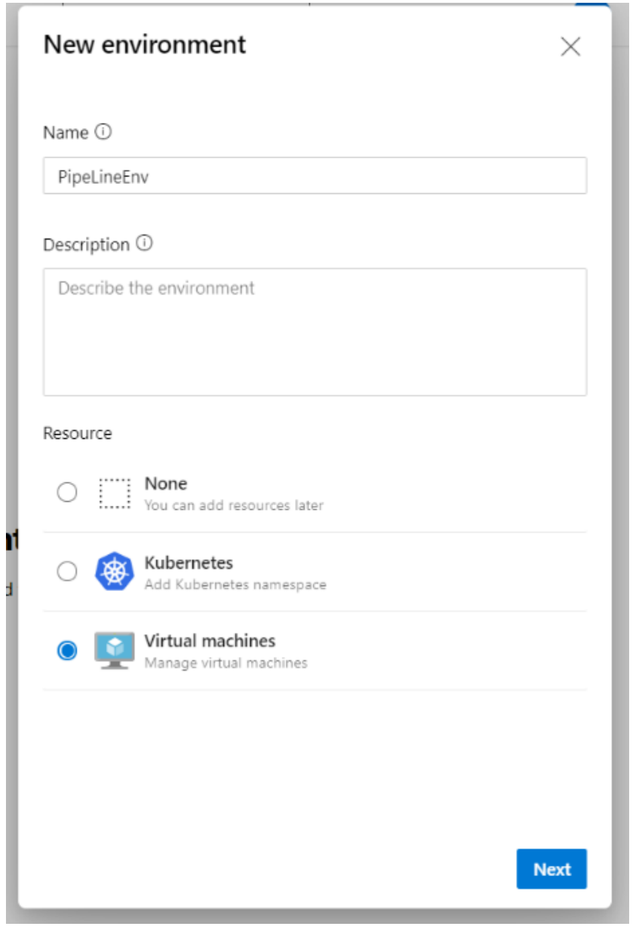
Provision and Configure Environments: Stage 2

Click on Environments



edureka!

Provision and Configure Environments: Stage 3

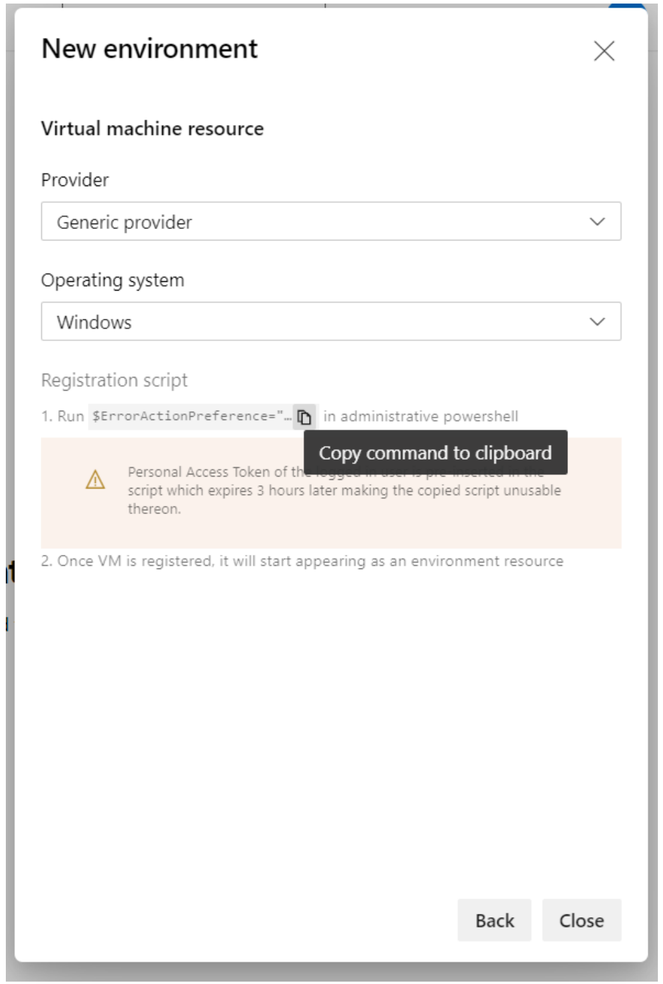


- Select either Kubernetes or Virtual Machine for environment
- Select Virtual Machine in this case

Click on Next

Provision and Configure Environments: Stage 4

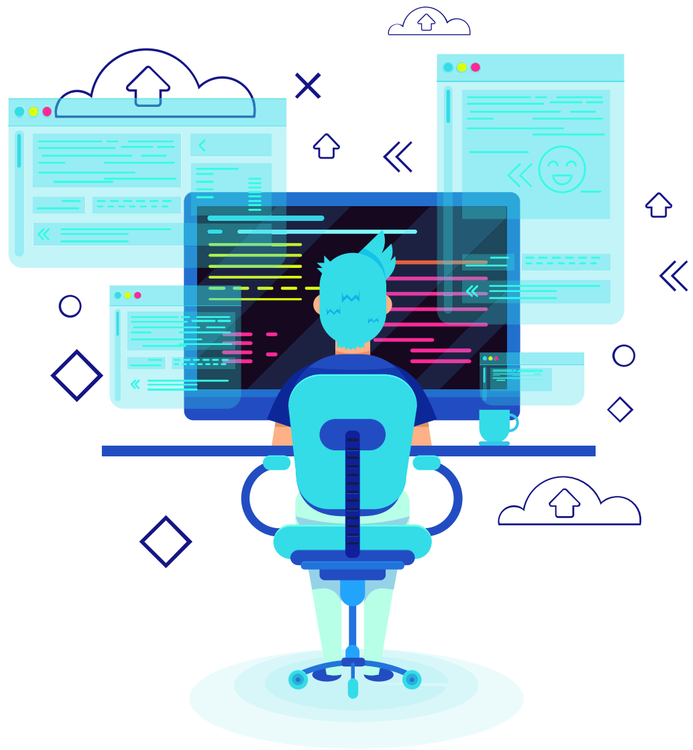
- Click on Copy command to the clipboard, as shown
- Run the command on any of the Azure VMs through PowerShell
- Once this command is executed, the VM will be registered for environment



Provision and Configure Environments: Stage 5

This environment can be used in the YAML code of the Azure pipeline, as shown below.

```
jobs:
- deployment: VMDeploy
  displayName: web
  environment:
    name: PipeLineEnv
    resourceType: VirtualMachine
  strategy:
#The above code states that this virtual machine will be used as an environment for deployment.
```



edureka!

Tasks and Templates

edureka!

Tasks and Templates: Manage and Modularize

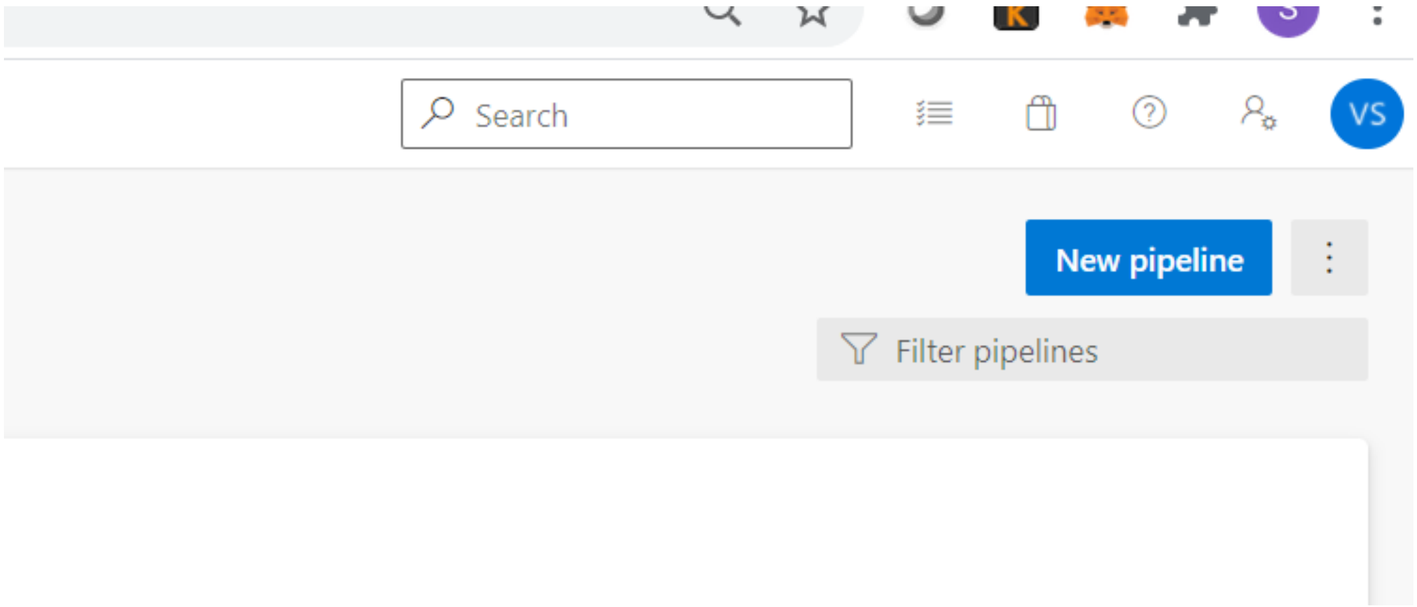
- The task is the building block of the pipeline
- The pipeline consists of many tasks

- The task can be a script or procedure
- Sometimes for a task to start, a dependent task must be completed
- During the pipeline's execution, the agent is chosen based on which agent can execute all these tasks

edureka!

Manage and Modularize Tasks: Stage 1

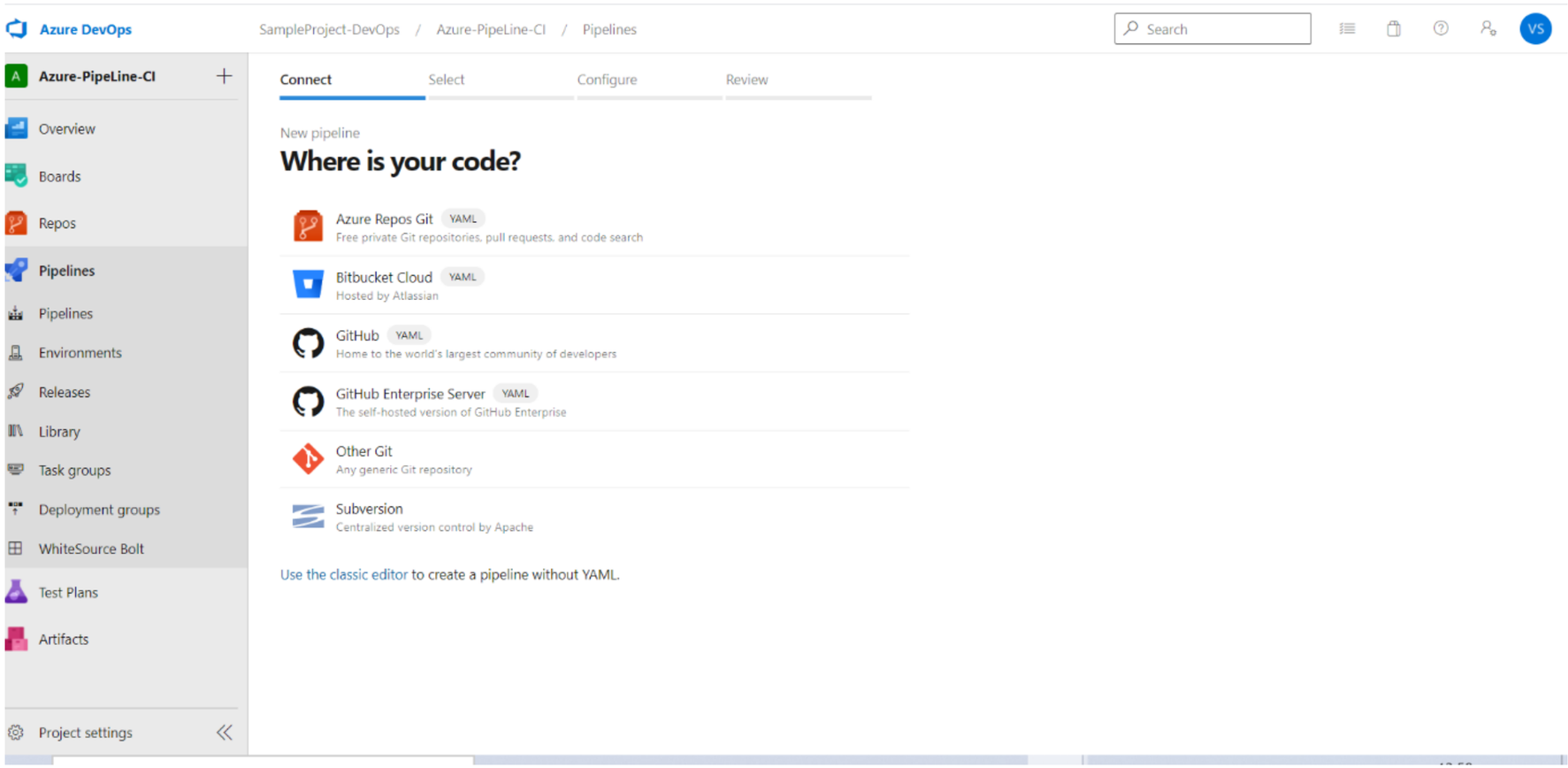
All the tasks run in the same context and are defined when we create a pipeline, as shown below:



- Go to Azure DevOps
- Go to the project for which we need to define the pipeline
- Click on New pipeline

Manage and Modularize Tasks: Stage 2

Click on the Classic Editor of the screen, as shown below.

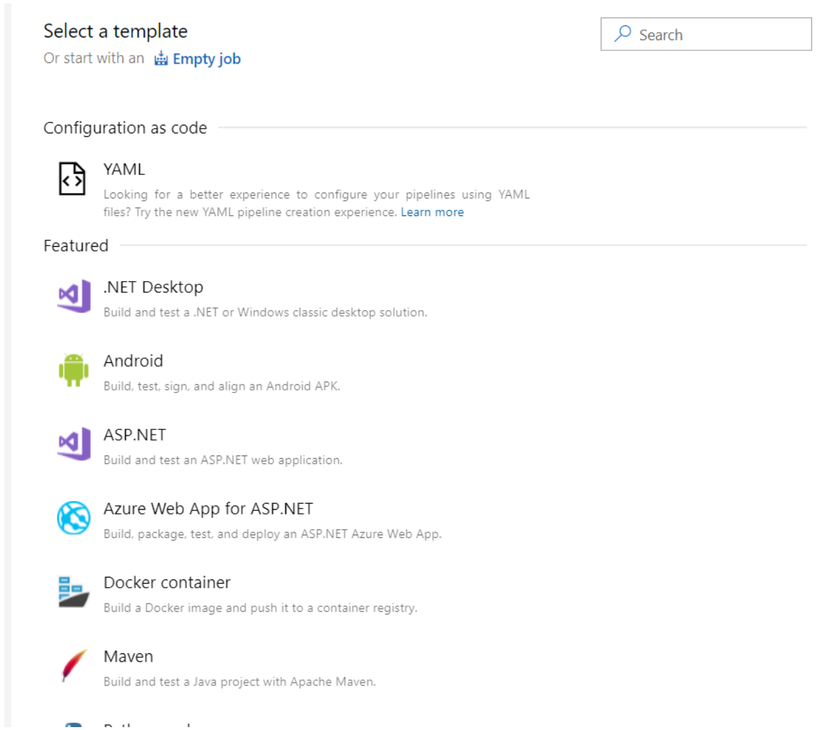


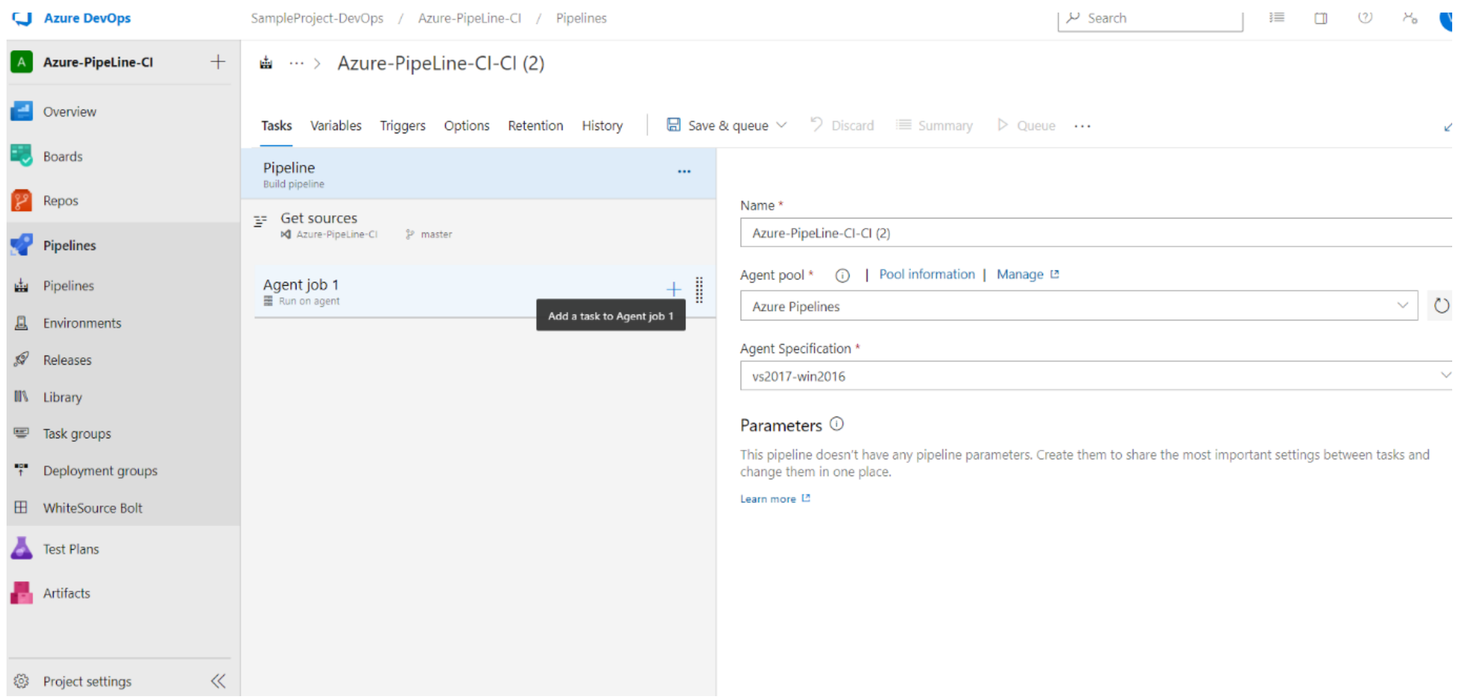
edureka!

Manage and Modularize Tasks: Stage 3

Click on Empty Job

The next image shows the agent, and the user can add the tasks

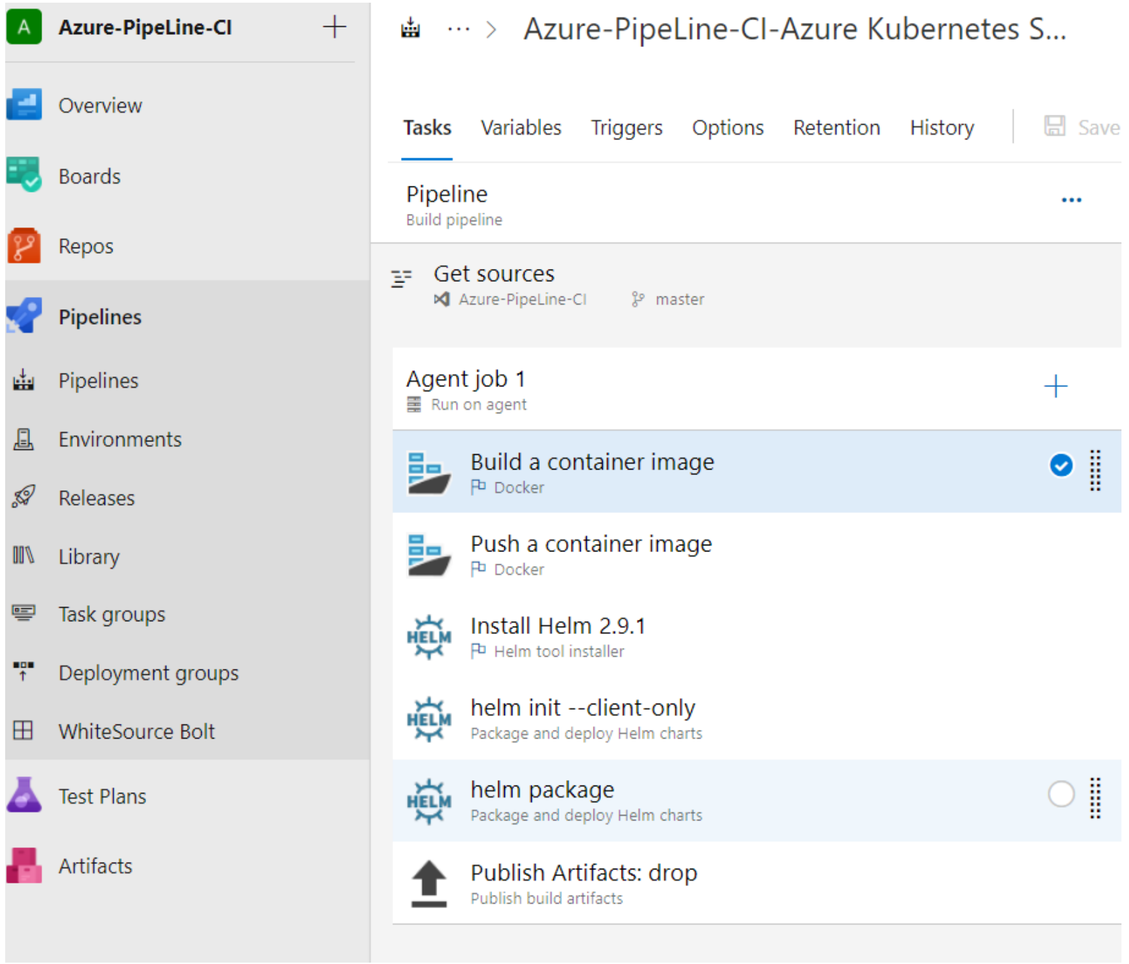




edureka!

Manage and Modularize Tasks: Stage 4

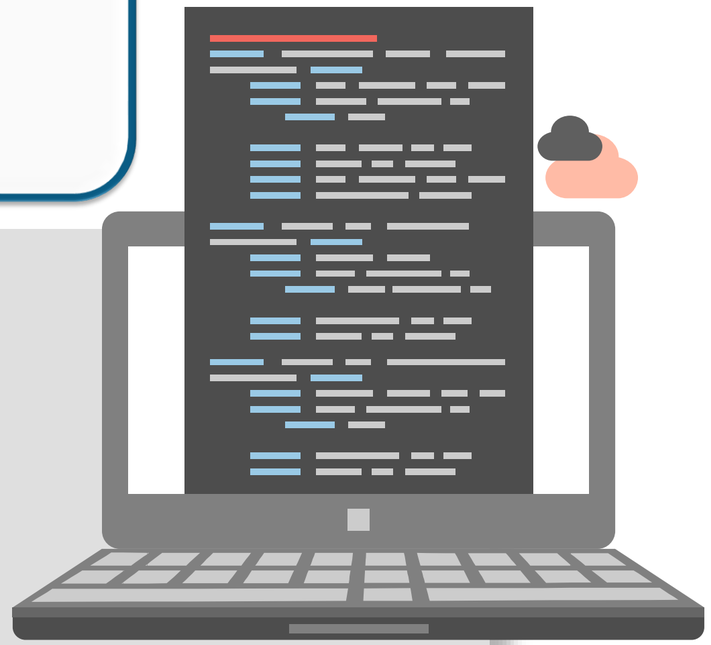
- Users can add tasks that agents need to perform by clicking on the "+" button
- These tasks can be defined in the YAML file
- The tasks can be grouped into Task Group



edureka!

Manage and Modularize Template

- It is a kind of library where a series of steps are defined, and we use these steps as part of our pipeline
- This helps us reuse the existing YAML code



```
 jobs:
- job: Linux
  pool:
    vmImage: 'ubuntu-16.04'
  steps:
  - template: resourceTemplate.yml
# The template file “resourceTemplate.yml” will run
as part of steps.
```

Demo: Configuring CI/CD Pipeline as Code with YAML

edureka!

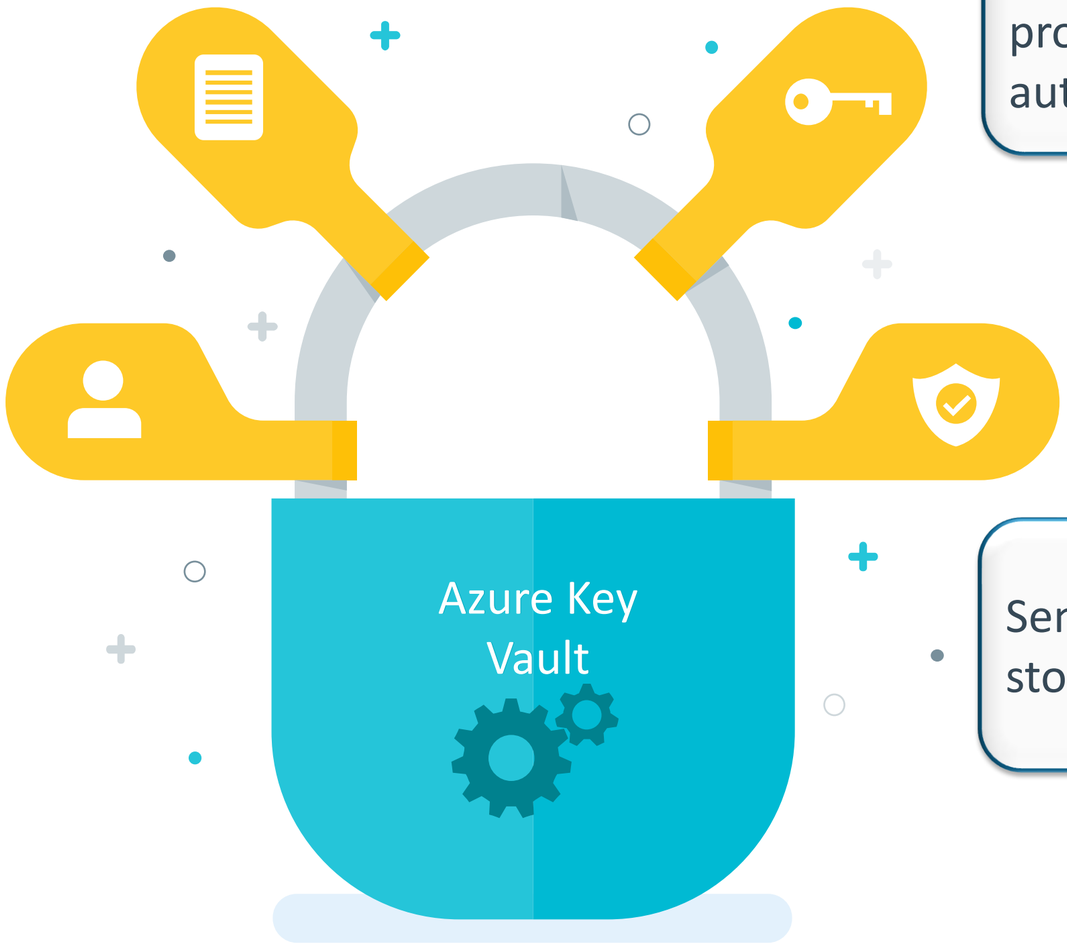
Integrating Secrets with Release Pipeline

edureka!

Integrate Secrets With the Release Pipeline

The application needs to access connection string, secrets, API keys, certificates, etc

For access, it requires proper authentication and authorization



They are not mentioned as part of the Dev Code or pipeline

Sensitive information is stored

Azure Key Vault

Azure key vault is centralized storage, safeguarded by industry-standard algorithms, key lengths, and even hardware security modules.

```
# In the pipeline YAML file, the key vault is defined as:
#Download Azure Key Vault secrets
- task: AzureKeyVault@1
  inputs:
    azureSubscription:
    keyVaultName:
    secretsFilter: '*'
    runAsPreJob: false #Azure DevOps Services only
```

edureka!

Azure Key Vault: Parameters

azureSubscription

The Azure subscription required to create resource in azure

keyVaultName

Key Vault name which needs to be accessed to get the secrets

secretsFilter

A comma-separated list of secrets that need to be retrieved from Azure Key Vault

runAsPreJob

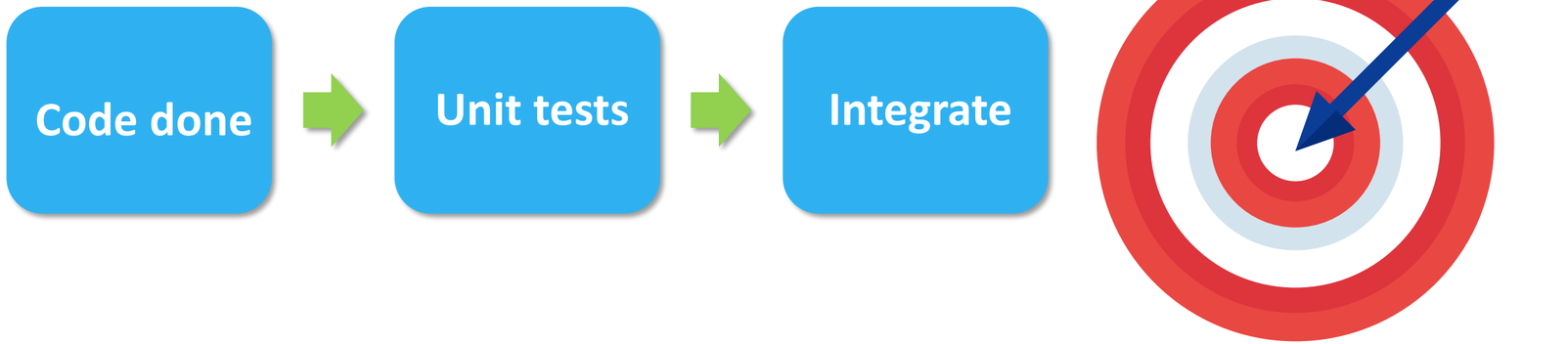
Boolean value. If true: exposes secrets to all tasks of Job. False(default): not expose the secret to all the Jobs

Demo: Using Secrets in Pipeline with Azure Key Vault

edureka!

Configuring Automated Integration

- Any new change/integration will trigger an automated build and testing to check for any integration issue
- The goal is to make sure there is no impact on application due to any new change



Continuous Integration

Configuring Automated Integration: Objectives

Objectives
1 Improved collaboration
2 Enable smooth parallel development
3 Minimize integration debt
4 Act as a quality gate
5 Automation of all steps

edureka!

Configuring Functional Test Automation

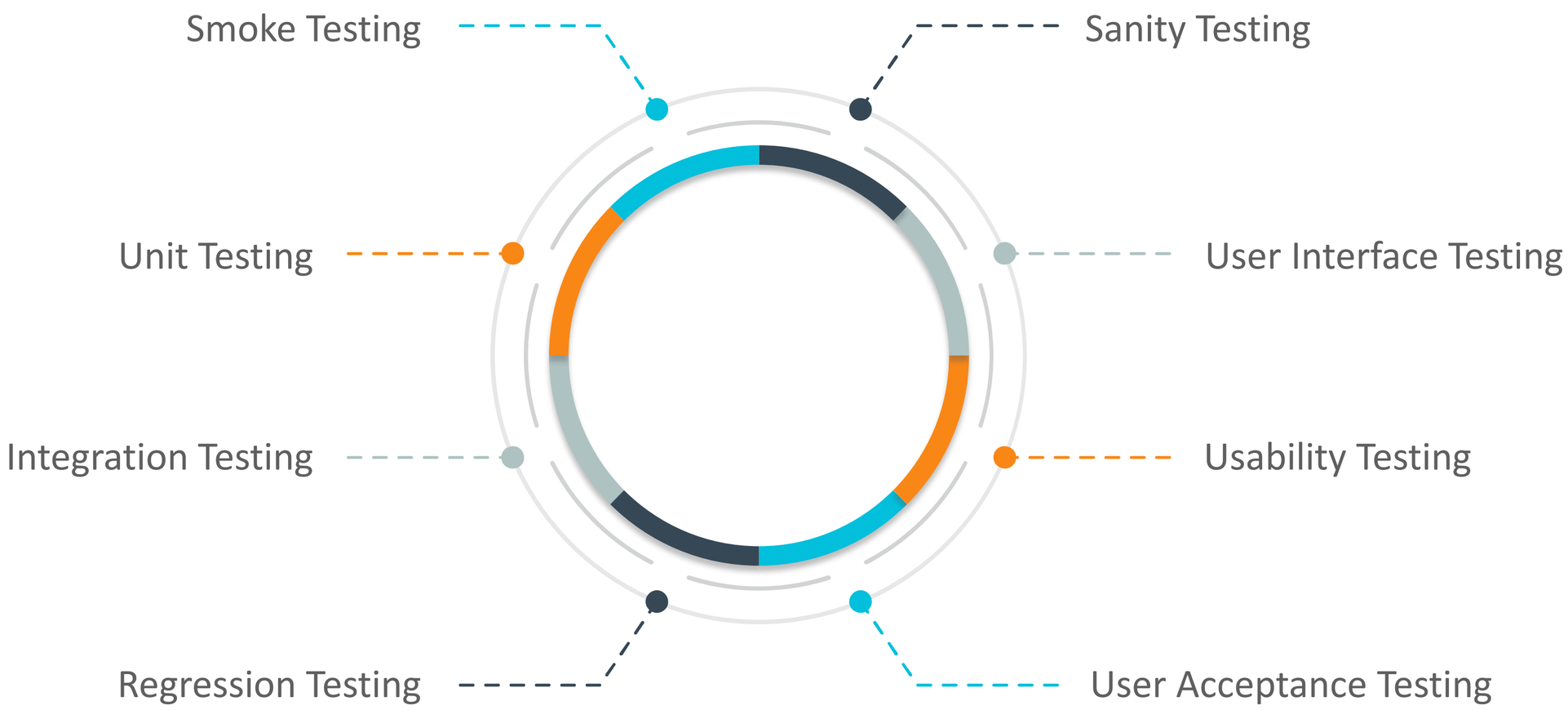
01 Each function of software does what is expected

02 The tests focus on the outcome and not on how it is implemented

03 As part of the test, inputs are provided to test cases, and its output is checked with the expected output

04 If both are the same, then the test passes, and if both are not the same, then the test fails

Types of Functional Test



Demo: Setting Up and Running Functional Tests

edureka!

Automating Health Inspection

edureka!

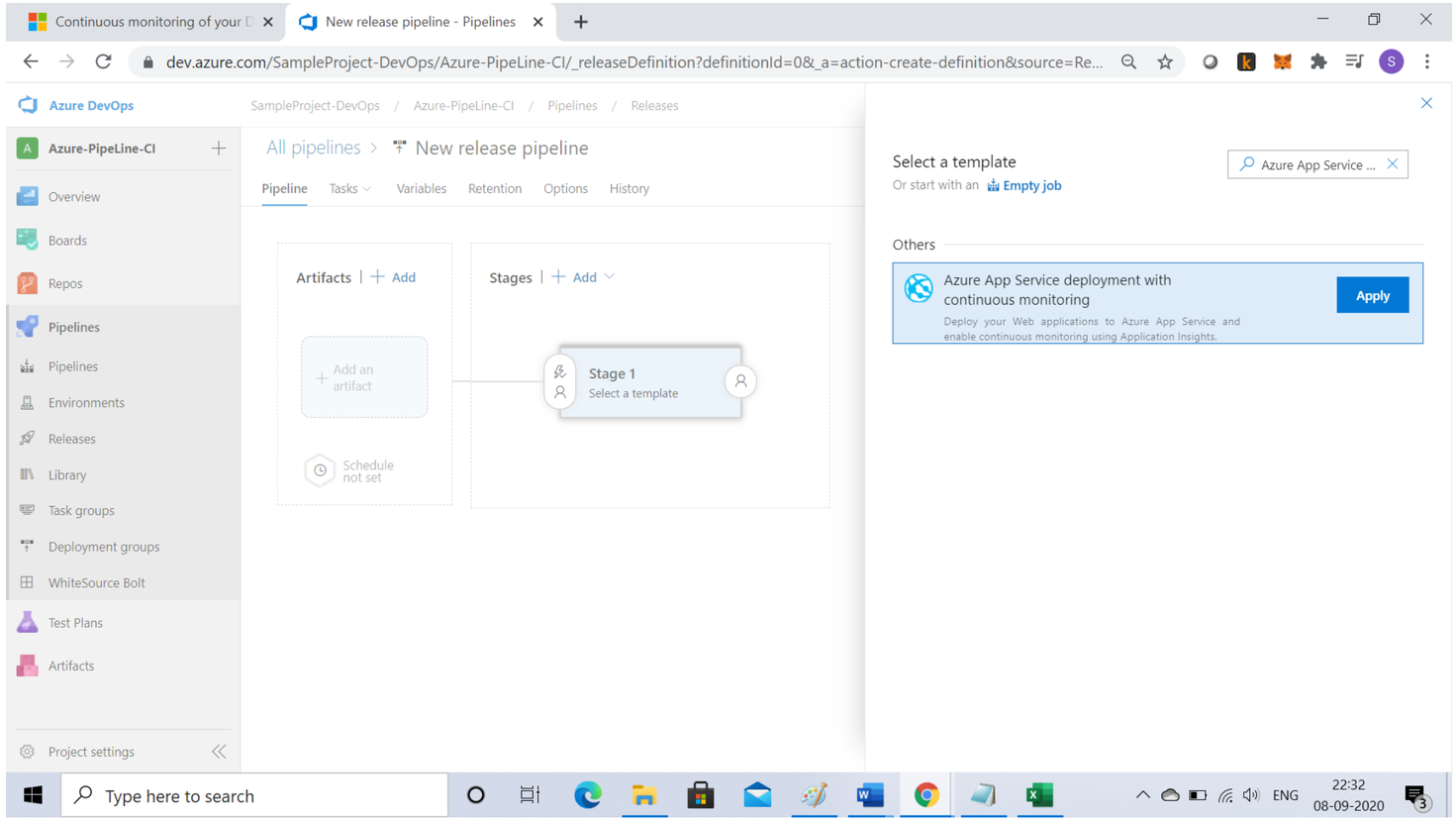
Automating Health Inspection

- The release pipeline health inspection can be done through Continuous monitoring
- Azure Application Insights can be used as part of the Azure pipeline for continuous monitoring
- When the release pipeline detects an Application Insights alert, the pipeline can gate or roll back the deployment until the alert is resolved
- If all checks pass, deployments can proceed automatically from test to production, without the need for manual intervention

Automating Health Inspection: Continuous Monitoring

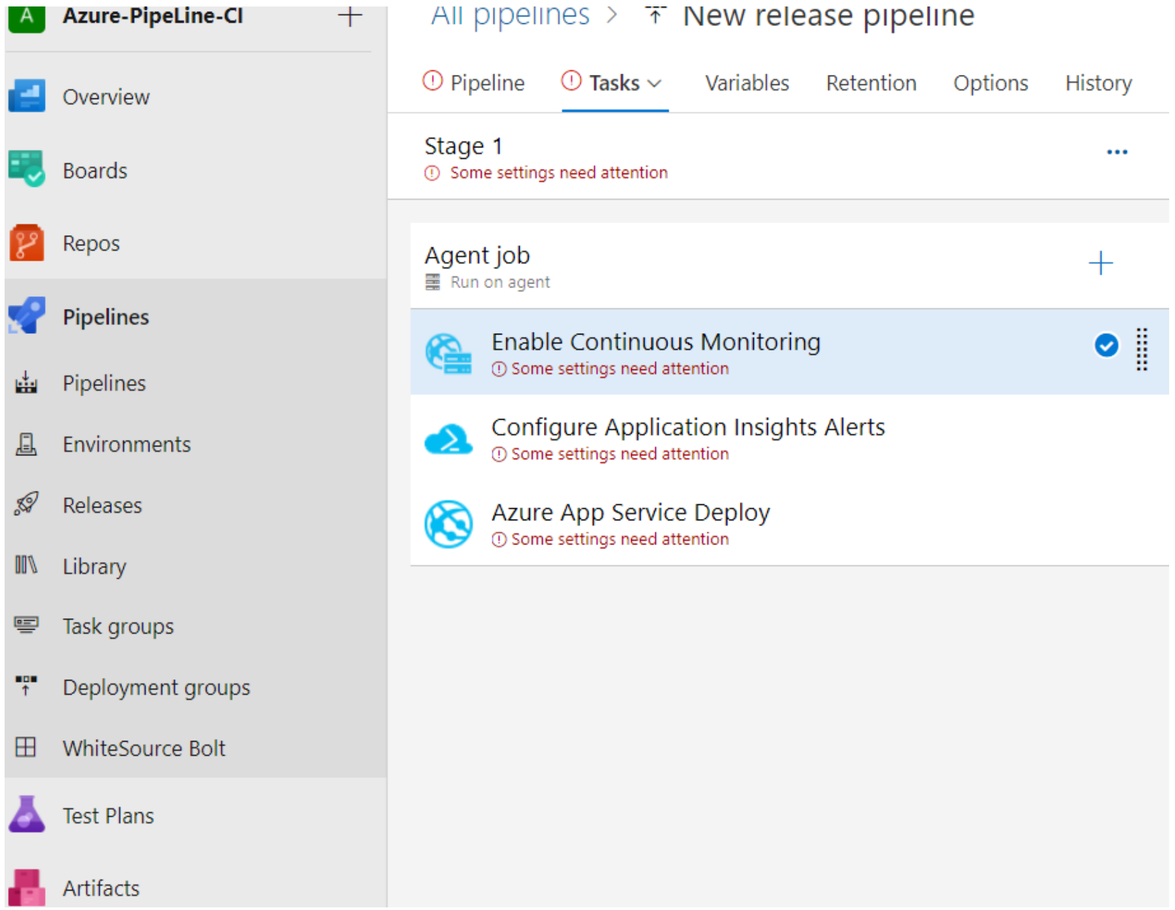
Steps to implement continuous monitoring in the release pipeline:

- Open the Azure DevOps and go to the pipeline section of any project
- Go to Pipeline → Releases
- Create a new release pipeline
- Select “Azure App Service deployment with continuous monitoring” as part of the stage, as shown here



edureka!

Automating Health Inspection: Continuous Monitoring

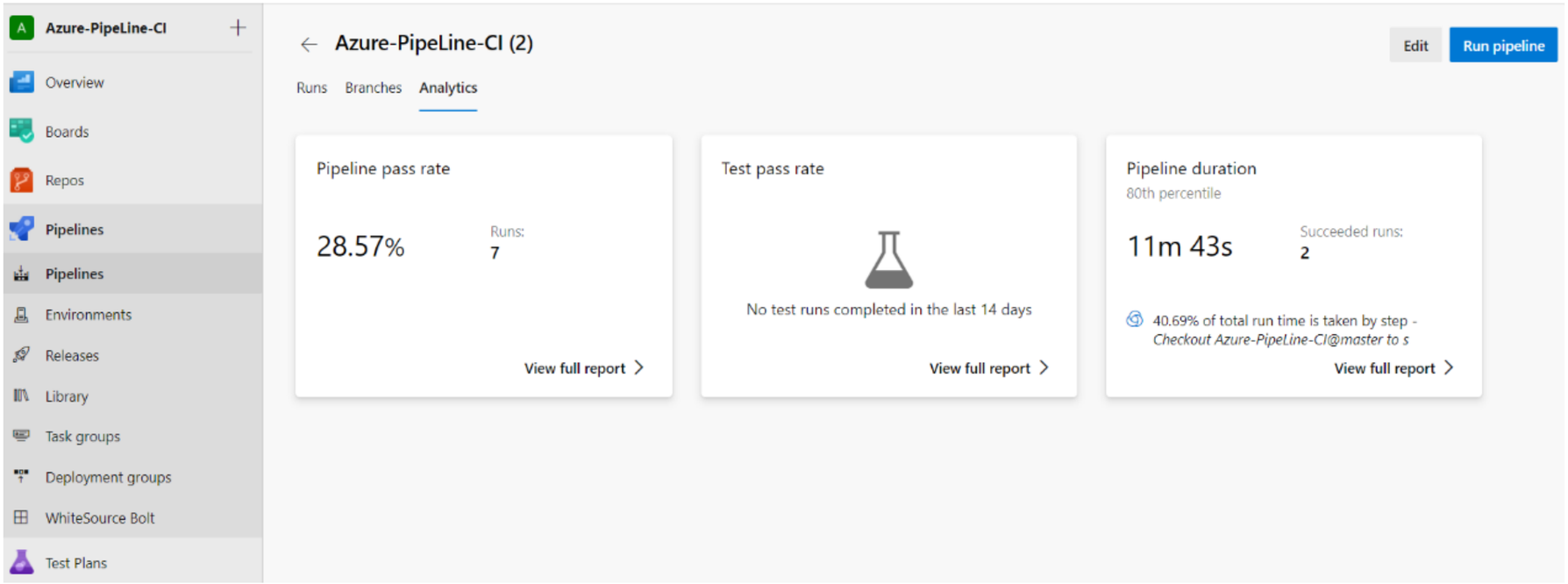


Settings like Azure subscription, app service name, etc., need to be provided for these jobs, and this way, Azure Application Insights will be integrated. This way, the health of the pipeline can be monitored.

edureka!

Automating Health Inspection: Analytics

Apart from the previous way, any pipeline will also retain the Analytics of its previous runs, as shown below.



edureka!

Demo: Release Deployment Control Using Azure Monitor as Release Gate

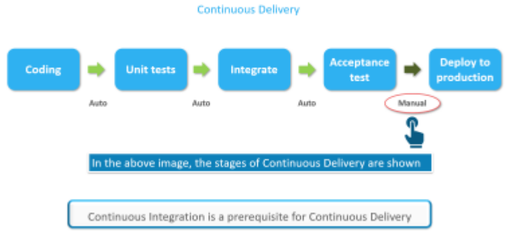
edureka!

Demo: Create a Release Dashboard to Share Information, Monitor Progress and Trends

edureka!

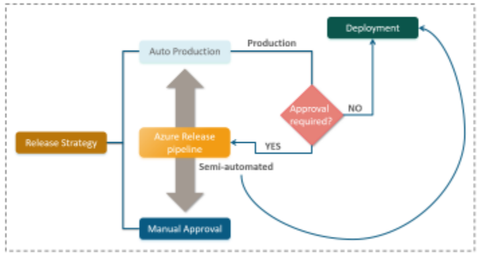
Summary

Stages of Continuous Delivery



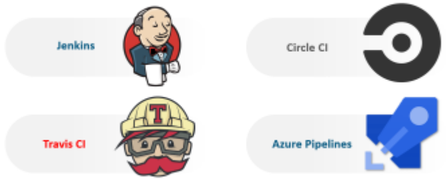
edureka!

Release Strategy Recommendations



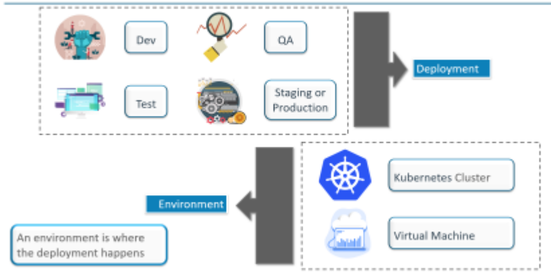
edureka!

Types of Release Tools



edureka!

Provision and Configure Environments



edureka!

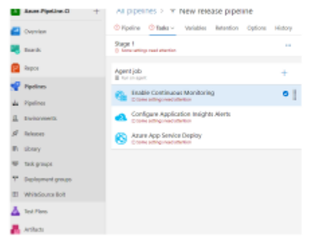
Manage and Modularize Template

- It is a kind of library where a series of steps are defined, and we use these steps as part of our pipeline
- This helps us reuse the existing YAML code

```
jobs:
- job: Linux
  pool:
    vmImage: 'ubuntu-16.04'
  steps:
  - template: resourceTemplate.yml
# The template file "resourceTemplate.yml" will run as part of steps.
```

edureka!

Automating Health Inspection: Continuous Monitoring



Settings like Azure subscription, app service name, etc., need to be provided for these jobs, and this way, Azure Application Insights will be integrated. This way, the health of the pipeline can be monitored.

edureka!

edureka!

edureka!

Thank You

For more information please visit our website
www.edureka.co