What is DevOps ?

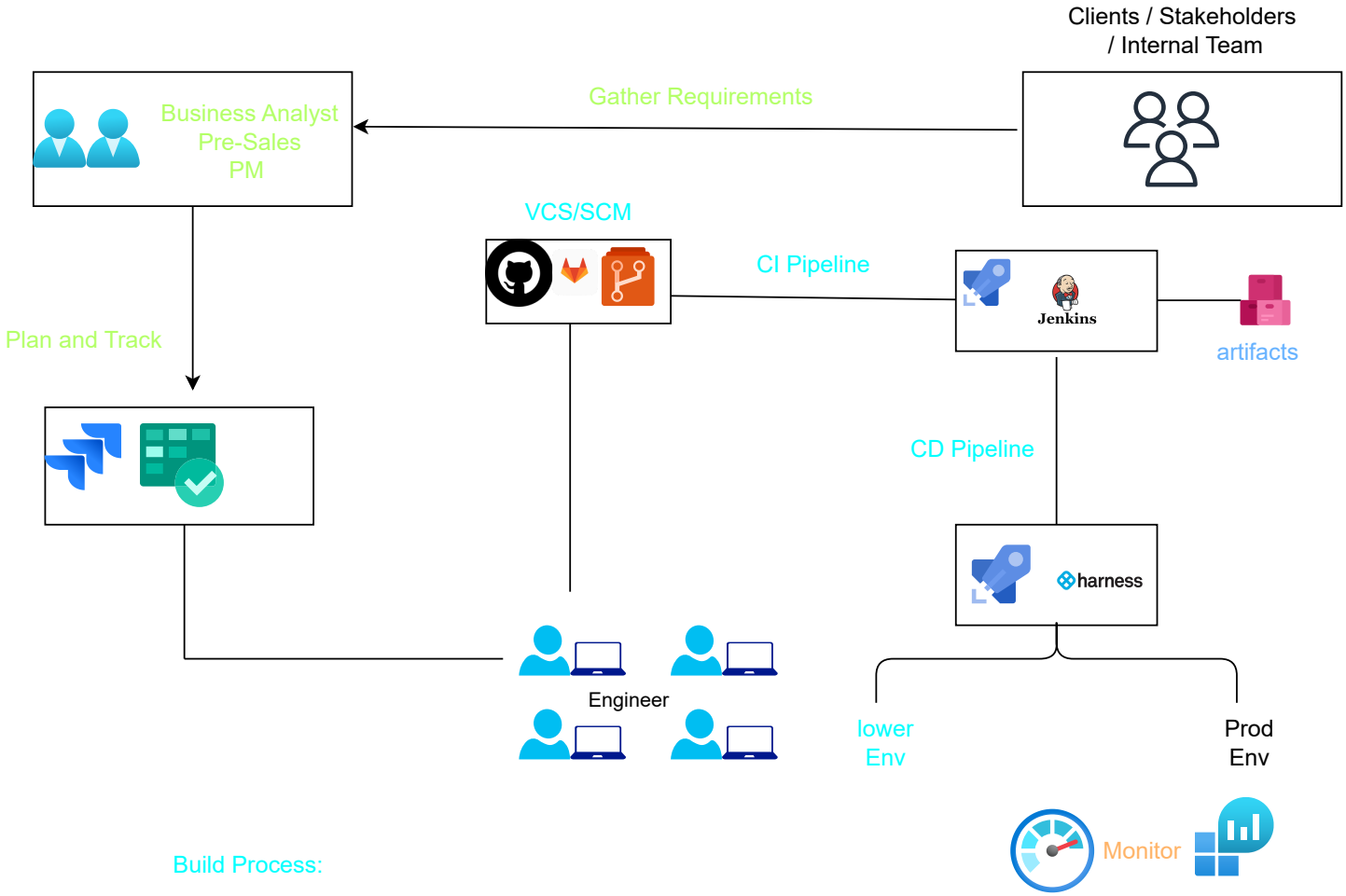
1. A methodology - We can't buy or Sell instead we develop it
2. Combination of people, process and Products with the aim to Reduce TTM and Increase Product Delivery Quality

PM / SM / Agile Coach
Developers
Ops
Testing

↓

So no more siloed

Automation Benefits:
1. Avoid Human Mistakes
2. to reduce time on Repetitive process
3. Promote Re-usability



Build Process:

In Programming :
English ==> Binary (0,1)

in IaC :
Terraform Plan ==> what infra you are going to create

Continuous Deployment:
Deploy code to infra

In programming:
Deploy code to Infra

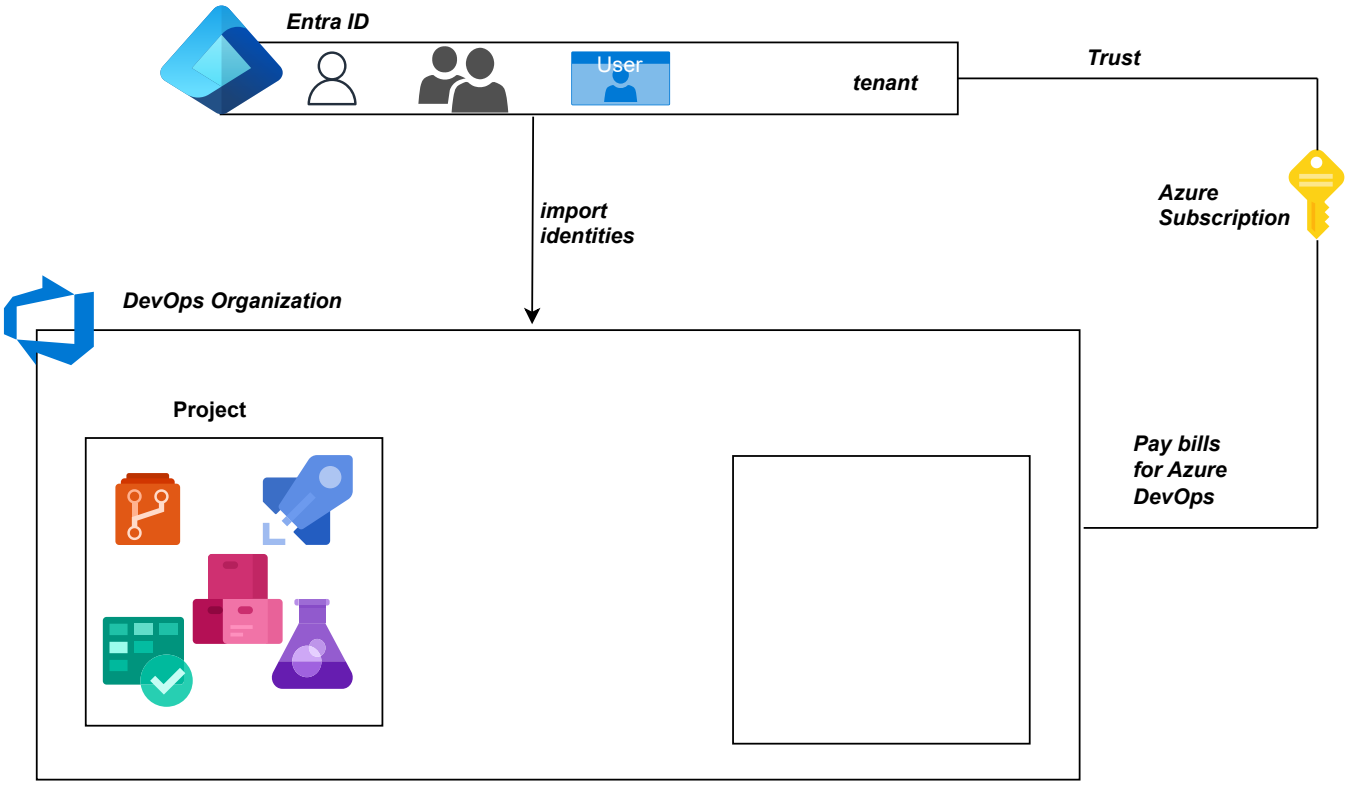
Continuous Integration (aka Build Process):

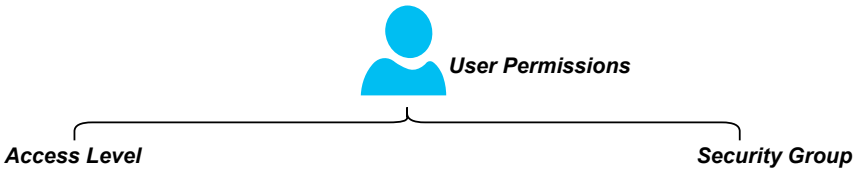
Build ==> Test ==> Output

Azure DevOps Service: SaaS
Azure DevOps Server : You manage hardware

Azure DevOps : Suite of Tools

Use an organization to connect groups of related projects and help scale up your enterprise. You can use a personal Microsoft account, GitHub account, or a work or school account. Using your work or school account automatically connects your organization to your Microsoft Entra ID.





Custom Security Group : permissions to include

Access levels in Azure DevOps control which web portal features are available or not. Access levels supplement security groups, which allow or deny specific tasks. Administrators ensure that their user base has access to the features they need and only pay for those specific features. It's an efficient way to manage costs while providing the necessary functionality to users.

All users in Azure DevOps belong to one or more default security groups. Security groups get assigned permissions that either Allow or Deny access to features or tasks.

Members inherit the permissions assigned to their security group.
Permissions get defined at different levels: organization/collection, project, or object.
Some permissions get managed through role-based assignments (for example, team administrator, extension management, or pipeline resource roles).
Administrators can define custom security groups to manage permissions for different functional areas.

You can restore deleted objects, projects, organization, anything but within 28 days.

Business Requirements → **Product backlog** → **Capacity Planning** → **Sprint Planning** → **Work Items - EPIC, User Stories**

Defines the building blocks of the work item tracking system and supports the Inheritance process model for Azure Boards. This model supports customization of projects through a What You See Is What You Get (WYSIWYG) user interface.

Basic: Is the most lightweight and is in a selective preview.
Scrum: Is the next most lightweight.
Agile: Supports many Agile method terms.
CMMI: Provides the most support for formal processes and change management.

User Stories

In consultation with the customer or product owner, the team divides up the work to be done into functional increments called "user stories."

User Story Template

The "role-feature-reason" template is one of the most commonly recommended aids to write user stories: As a ... I want ... So that ...

Given – When – Then

The Given-When-Then formula is a template intended to guide the writing of acceptance tests for a User Story: (Given) some context, (When) some action is carried out, (Then) a particular set of observable consequences should obtain.

Definition of Done

The definition of done is an agreed upon list of the activities deemed necessary to get a product increment, usually represented by a user story, to a done state by the end of a sprint.

Definition of Ready

Definition of Ready involves creating clear criteria that a user story must meet before being accepted into an upcoming iteration. This is typically based on the INVEST matrix.

Epic

An epic is a large user story that cannot be delivered as defined within a single iteration or is large enough that it can be split into smaller user stories.

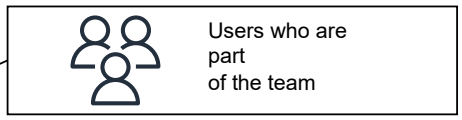
https://learn.microsoft.com/en-us/azure/devops/boards/boards/media/alm-kb-workflow.png?view=azure-devops

Area paths group work items by team, product, or feature area. Iteration paths group work into sprints, milestones, or other time-related periods. Both fields support hierarchical paths. Define area and iteration paths for a project, and teams can select which paths to use for their backlog and Agile tools

Frontend Team - UI/UX tasks (Area Path)

Beta Release - Area Path

Alpha Release Team ==> Azure Boards Teams



sprint

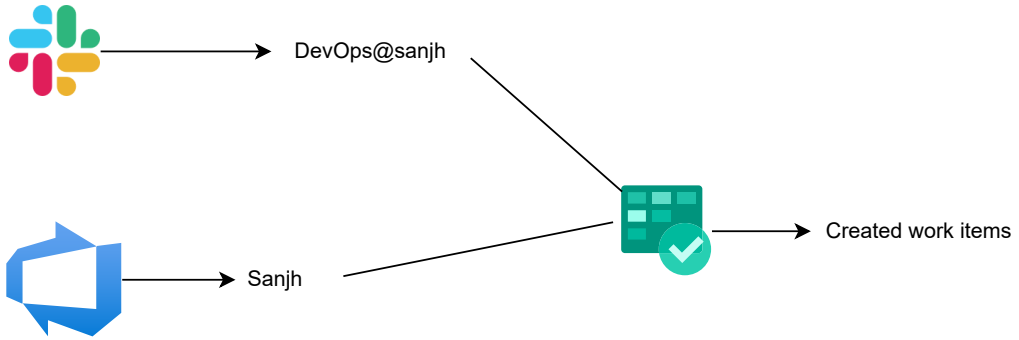
(3-feb-
17-feb)

work that needs to
be done
(Backlog)

Burndown charts
etc
Dashboard

Agile team capacity is how much time the team has available within the sprint to complete high-quality work. Note that capacity isn't simply a measure of the number of team members x the number of hours in their working day x the number of days in the sprint. That assumes an unhealthy 100% utilization rate

A burndown chart is a graphical representation of the work remaining versus time in a project or sprint. It helps visualize progress by showing how much work is left to be completed and whether the team is on track to meet their goals within the allotted time.



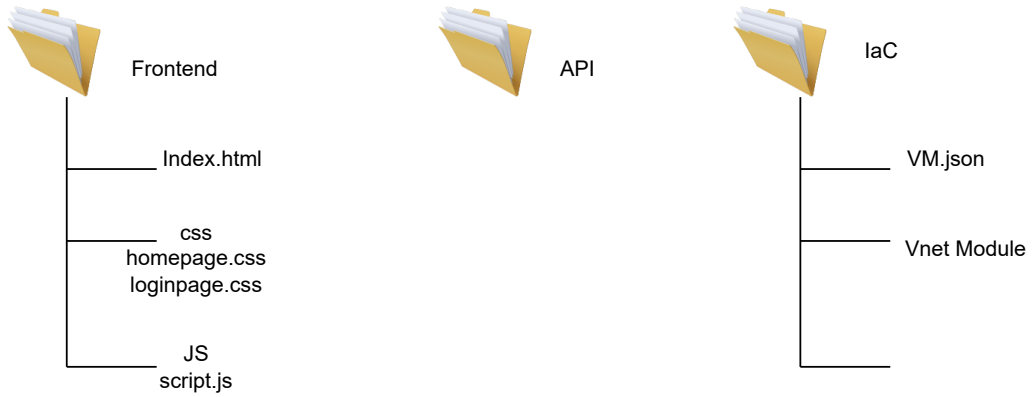
Service hook **publishers** define a set of *events* that you can subscribe to. **Subscriptions** listen for these *events* and define **actions** to take based on the event.

Subscriptions also target **consumers**, which are external services that can run their own actions when events occur.

Service Hooks let you run tasks on other services when events happen in your project in Azure DevOps.

For example, you can create a card in Trello when a work item gets created or send a push notification to your team's mobile devices when a build fails. You can also use service hooks in custom apps and services as a more efficient way to drive activities when events happen in your projects.

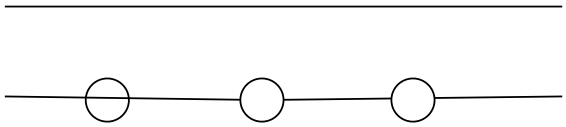
Version control system aka Source Code Management



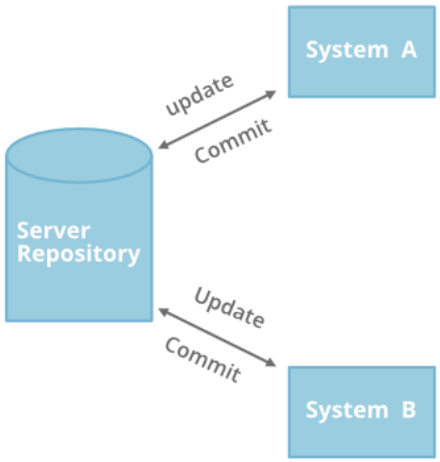
Version History

Multiple work and Collaborate

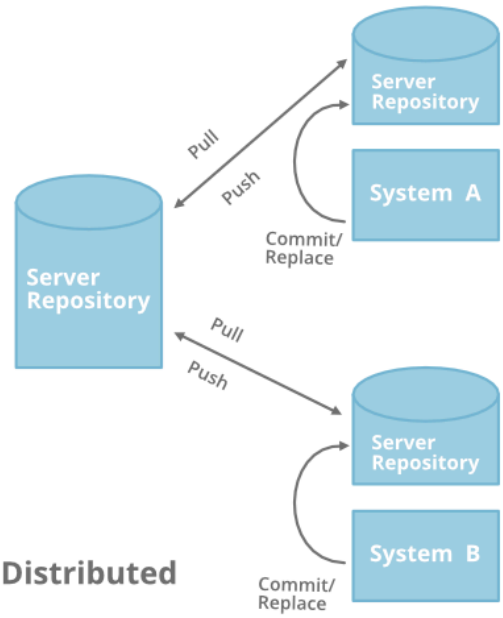
Branching and Merging



History of changes
operations prov Boolean selectGear(g : Gear) reqd Torque getTorque()
properties prov temperature : Integer reqd geometry : Spline



Centralized



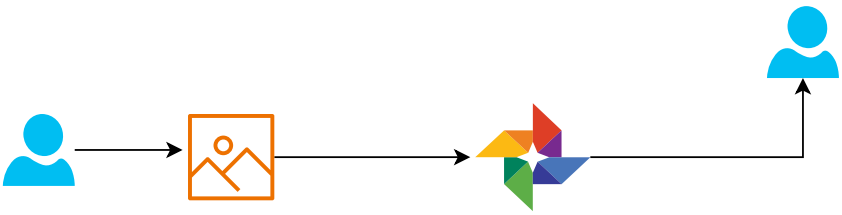
Distributed

Subversion, TFVC - Centralized

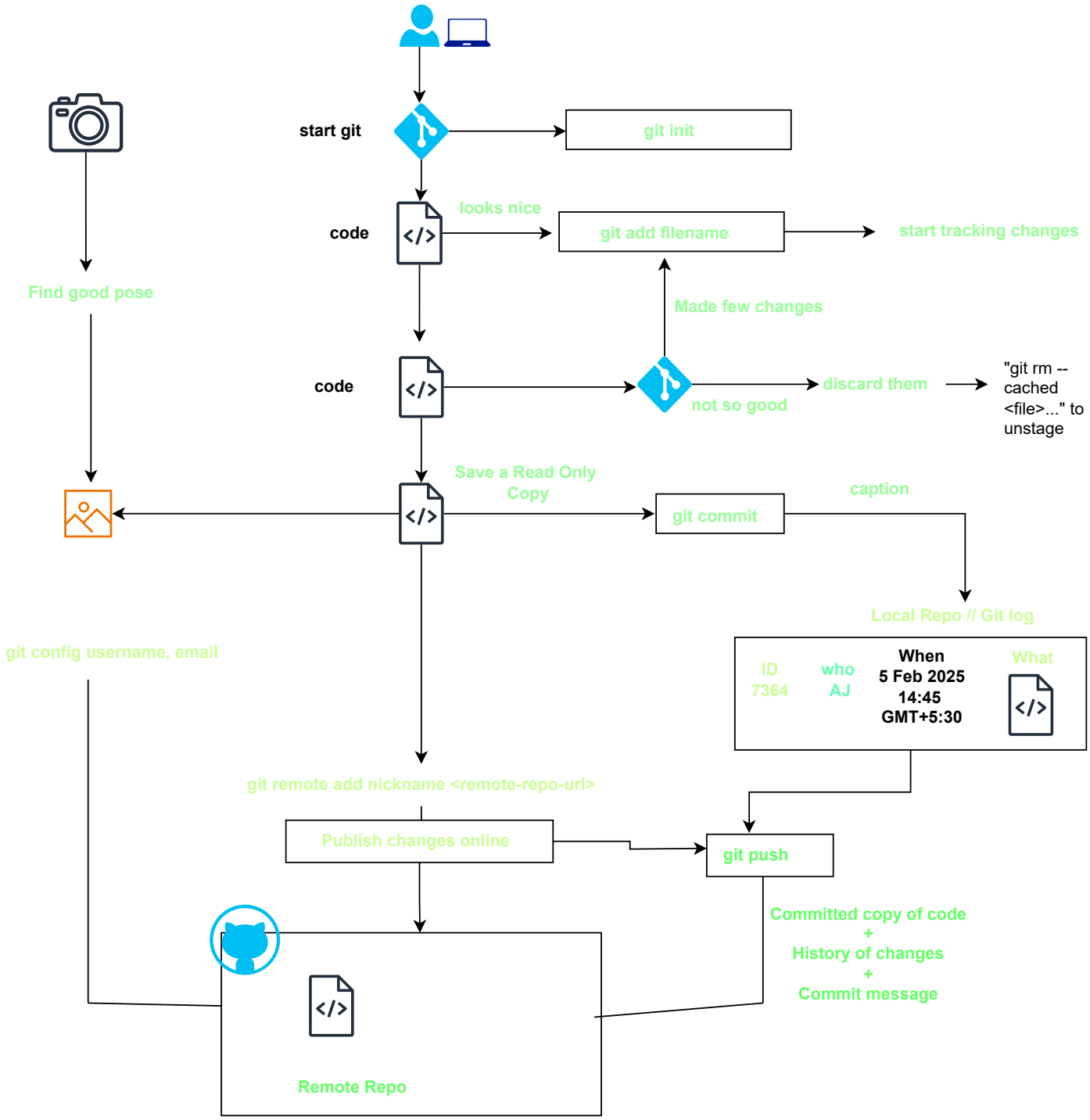
Git - is a disturbed VCS software
Mercurial: same as Git

Azure Repo, Github, Gitlab, bitbucket

1. Follow same Git commands
2. Storage / Management of Hardware => Vendor's Responsibility



Create Code **Upload online**




```
git status
On branch main


No commits yet


Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   Readme.md


Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Readme.md
```

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

Existing code - where you need to contribute

You have Permissions (Added in as Contributor)

You do NOT have permissions (Opensource)

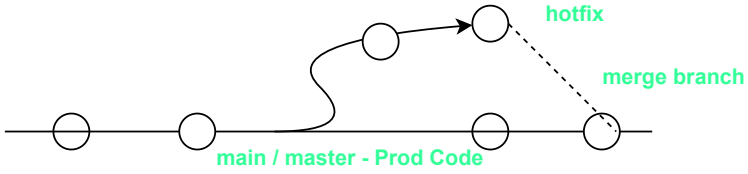
git clone

Fork the Repo

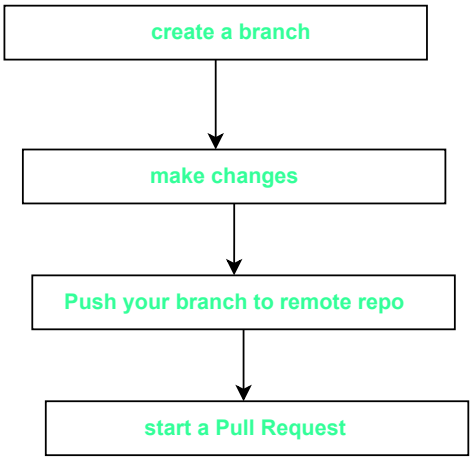
```
git push
remote: Permission to
ansible/ansible.git denied to Trainer-
AJ.
fatal: unable to access
'https://github.com/ansible/ansible.git/':
The requested URL returned error:
403
```

Create a Pull request → Approved / Rejected by approver

git recommended create a branch to isolate your proposed code changes



"Git branch represents independent line of development "



Parent branch

code + history of changes

copied to

New branch

code + history of changes → all new changes visible here now

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about diff comparisons here.

Merge Conflict:

When In two branches same content is getting changed
Git gets confused and raises conflict
which you solve manually

```
git merge feature-branch-2
Auto-merging example.txt
CONFLICT (content): Merge conflict in example.txt
Automatic merge failed; fix conflicts and then commit the result.
```

fix it Mannually

git add example.txt
git commit -m "Resolved merge conflict in example.txt"

```
git push
To https://github.com/Trainer-AJ/5feb25.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/Trainer-AJ/5feb25.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Git Pull

This branch has conflicts that must be resolved

Use the web editor or the command line to resolve conflicts.

Conflicting files

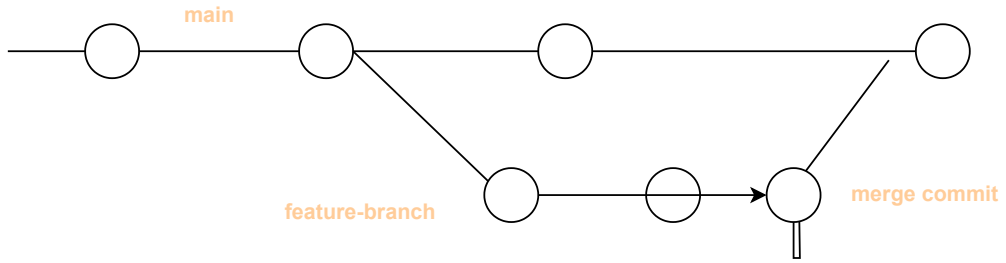
```
Readme.md

example.txt
```

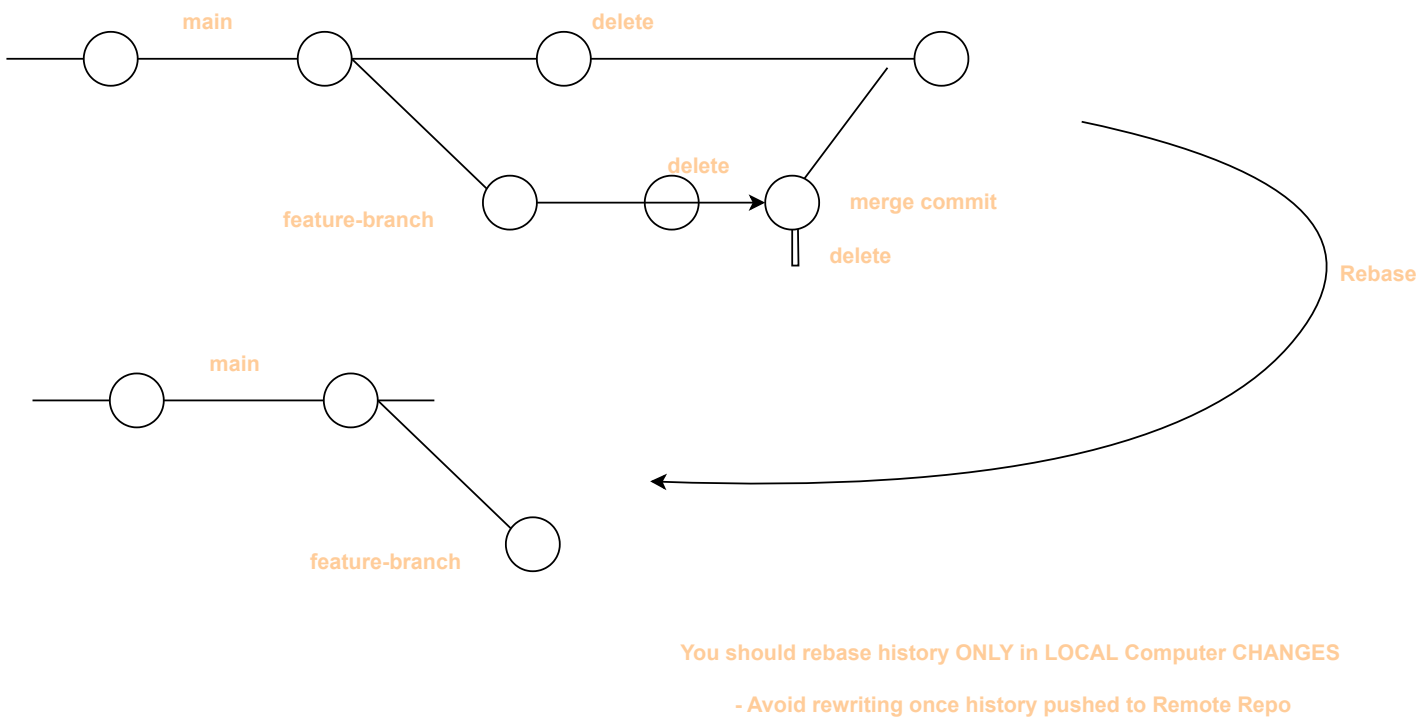
This branch is out-of-date

Update branch to merge the latest changes from the upstream repository into this branch.

Discard 5 commits to make this branch match the upstream repository. 5 commits will be removed from this branch.

Learn more about syncing a fork





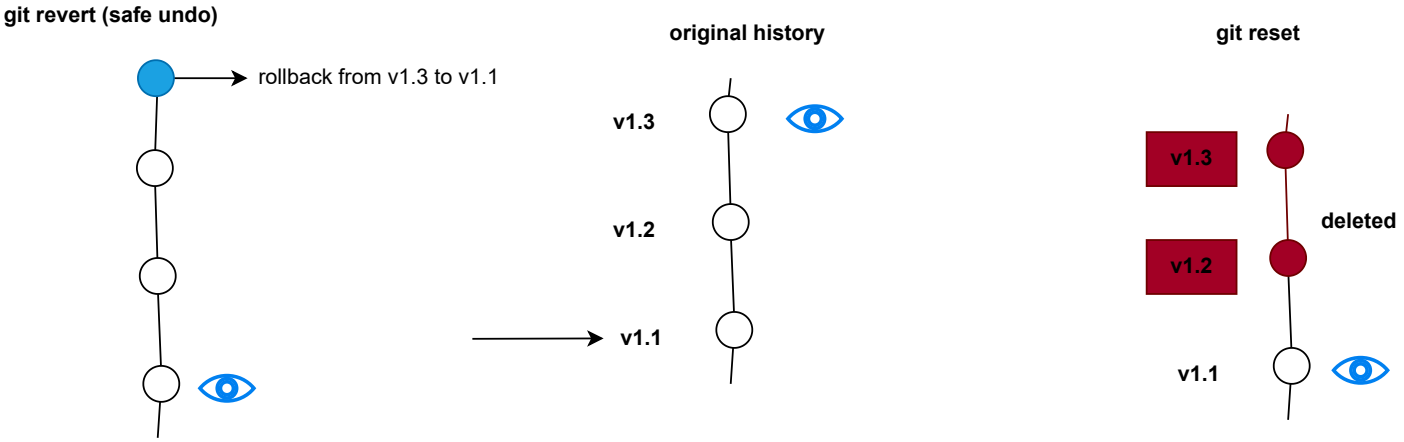
Branch policies help teams protect their important branches of development. Policies enforce your team's code quality and change management standards. This article describes how to set and manage branch policies. For an overview of all repository and branch policies and settings, see Git repository settings and policies.

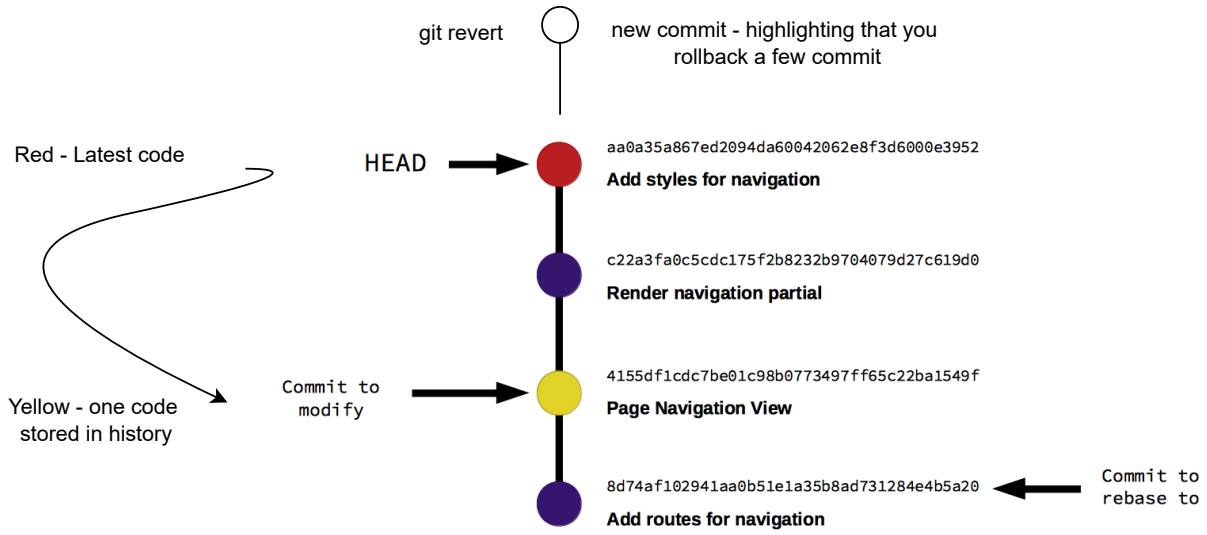
A branch with required policies configured can't be deleted, and requires pull requests (PRs) for all changes.

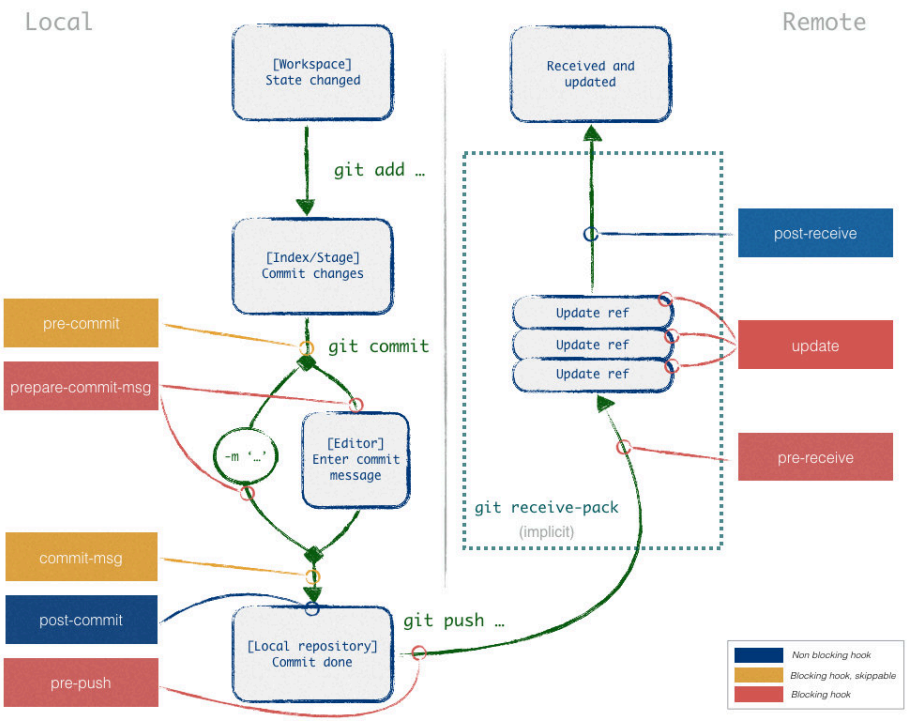
Effect of Branch Policy:

TF402455: Pushes to this branch are not permitted; you must use a pull request to update this branch.

https://etherpad.opendev.org/p/esiaz40







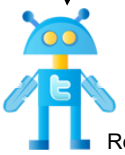
Git hook - is a collection of user created scripts to enforce users Policies and best Practices

.git/hooks and here remove the extension sample :)

It stays on local device only

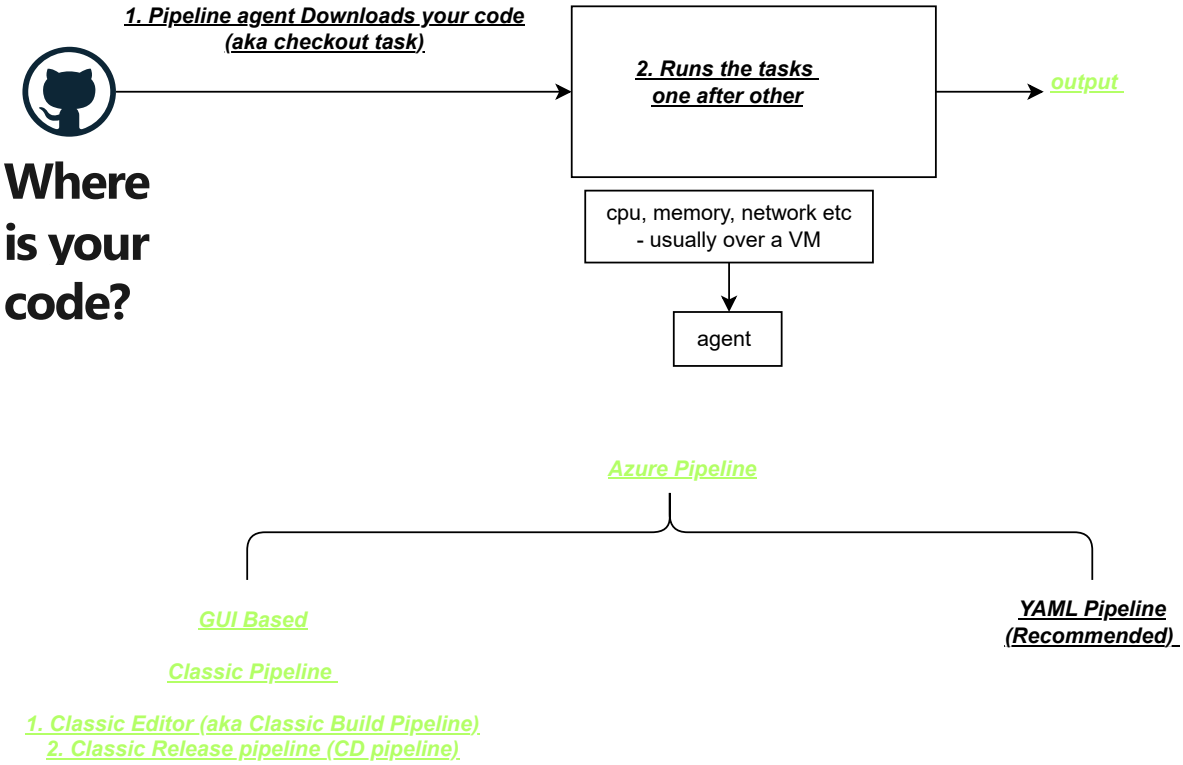
Azure Pipeline

Helps maintain constant flow of changes
collection of tasks
Conditions that specifies when to run Pipeline = Trigger



Robot

Should be created for repeatable, time consuming tasks



Microsoft-hosted agents

If your pipelines are in Azure Pipelines, then you've got a convenient option to run your jobs using a **Microsoft-hosted agent**. With Microsoft-hosted agents, maintenance and upgrades are taken care of for you. You always get the latest version of the VM image you specify in your pipeline. Each time you run a pipeline, you get a fresh virtual machine for each job in the pipeline. The virtual machine is discarded after one job (which means any change that a job makes to the virtual machine file system, such as checking out code, will be unavailable to the next job). Microsoft-hosted agents can run jobs directly on the VM or in a container.

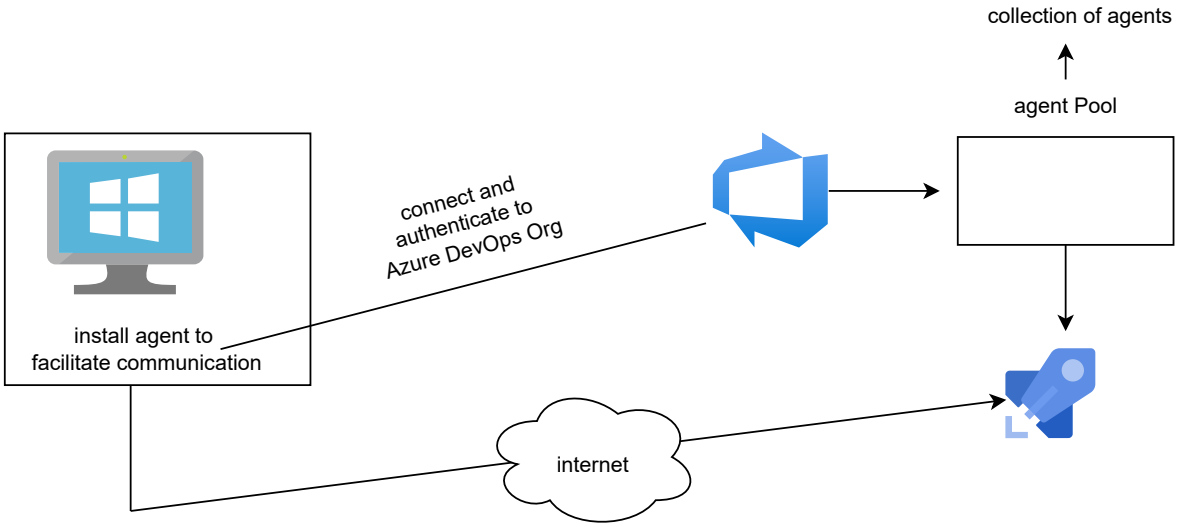
Azure Pipelines provides a predefined agent pool named **Azure Pipelines** with Microsoft-hosted agents.

https://github.com/actions/runner-images/blob/main/images/windows/Windows2022-Readme.md

Self-hosted agents

An agent that you set up and manage on your own to run jobs is a **self-hosted agent**. You can use self-hosted agents in Azure Pipelines or Azure DevOps Server. Self-hosted agents give you more control to install dependent software needed for your builds and deployments. Also, machine-level caches and configuration persist from run to run, which can boost speed.

Parallel jobs represents the number of jobs you can run at the same time in your organization. If your organization has a single parallel job, you can run a single job at a time in your organization, with any other concurrent jobs being queued until the first job completes. To run two jobs at the same time, you need two parallel jobs. In Azure Pipelines, you can run parallel jobs on Microsoft-hosted infrastructure or on your own (self-hosted) infrastructure.



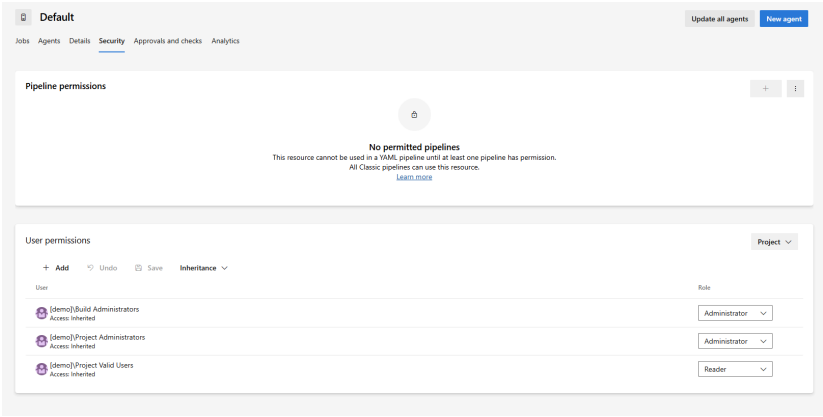
An agent pool is a collection of agents. Instead of managing each agent individually, you organize agents into agent pools. When you configure an agent, it is registered with a single pool, and when you create a pipeline, you specify the pool in which the pipeline runs. When you run the pipeline, it runs on an agent from that pool that meets the demands of the pipeline.

Why self hosted agent ?

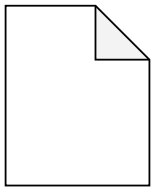
1. Deploy in restricted
2. Install My required software
3. Re-use my pipeline agent even after pipeline completes

AT Org Settings level - You select under which Project agent pool shows UP

@ Project settings you
control which Pipeline can use your agent Pool



Trigger:
- any file / branch
- Schedule
- pipeline chaining
- Manual



Azure Pipeline

agent → 1. Job can inherit Agent from Pipeline
OR
2. Job Can have its own agent

Job: A collection of task

1. Task Unit test
2. Integration test
3. Security Test
(Code scanning)

job-1
job-2
job-3

Stage : Collection of Jobs

1. Container Image creation:
- Build code
- test
- Create Image
- test Image

2. Create Infra
- Create A K8s Cluster
- Specify Network, PV etc

here is no guarantee that the job will execute in the same order. → Define Dependency in Jobs

Two jobs share no data by default. → Publish artifacts

By default job run parallelly.

Third Party Tool / Outsider
(Azure DevOps, GIthub actions, Terraform etc)

Service Connection:
YOu can connect your
azure devops
Pipeline
to any
External Environment

entra Tenant

Service Principal / App Registration /
managed Identity

RBAC (role based access control)
- you grant permissions to any object

success

f you need to connect your pipeline with GitHub.
Ananay Ojha
You need to create a service connection.

1. Check Pre-Req
- software
- Lib / binaries
- code

2. Build - test Code

3. Deploy

- Caching : Builds
- small independent pieces ==> Output of pipeline one input of pipeline 2.

Every CI -CD tool will just have terminology and syntax different.

Agent

trigger

task

Pipeline

.github/workflows

Runner

on

Job

Workflow

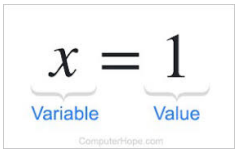
```
trigger:
- main

pool:
  vmImage: ubuntu-latest

steps:
- script: echo Hello, world!
  displayName: 'Run a one-line script'

- script: |
    echo Add other tasks to build, test, and deploy your project.
    echo See https://aka.ms/yaml
  displayName: 'Run a multi-line script'
```

```
name: First Pipeline

# trigger - when to RUn?
on:
  push:
    branches:
      - main

# where to RUN - agent pool
jobs:
  job1:
    runs-on: ubuntu-latest

    steps:
    - name: Get The COde
      uses: actions/checkout@v4.2.2

    - name: RUn a one-line Script
      run: echo "Hello WOlrd"
```

How to make a Pipeline Re-useable?

1. Variable

2. templates



1. Variable

Project Level (across multiple pipelines example: Subscription ID → **Variable Groups**

Pipeline Level (example: tags, Region Name file folder names) → **Variables in YAML/ classic editor**

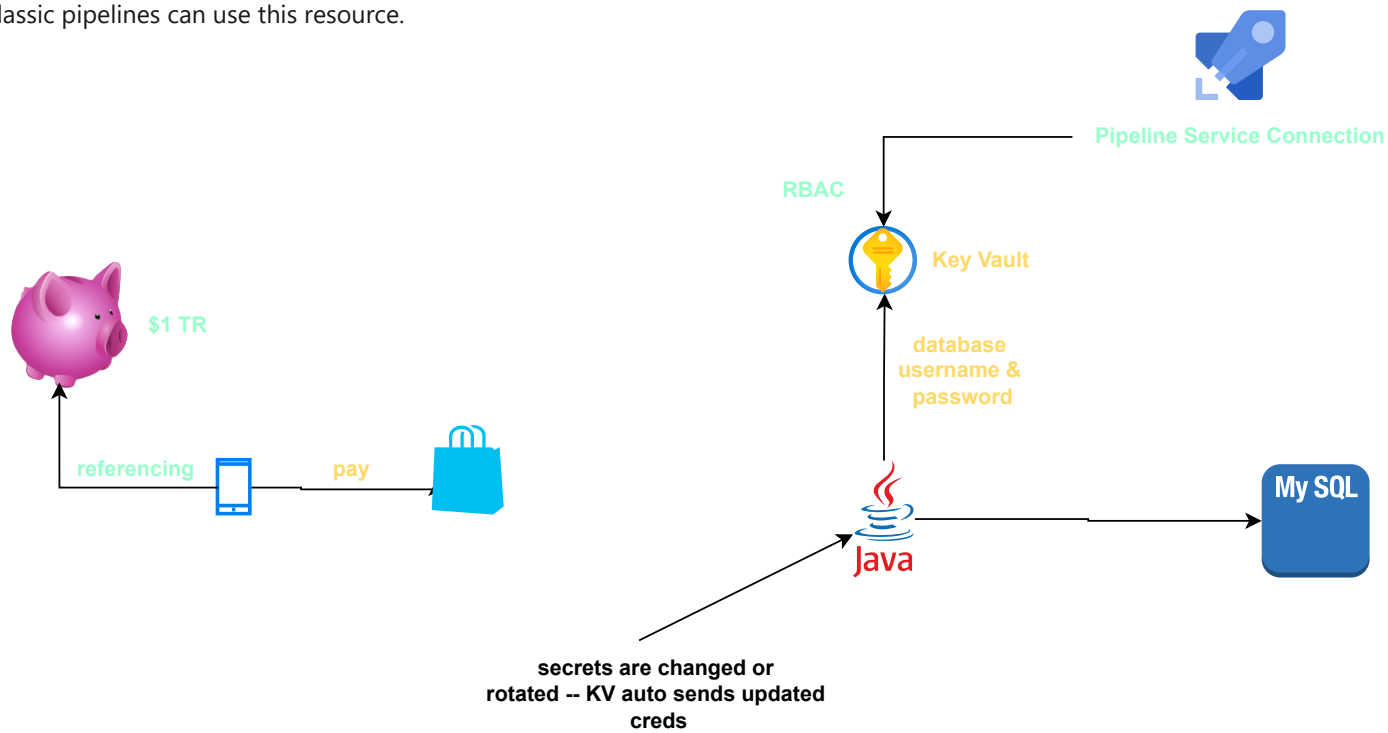
Task Level Variables exmaple: terraform plan - out-file name → **Environment variable**

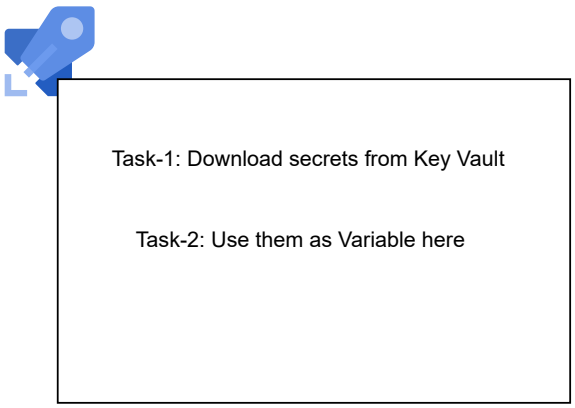
Example: Source Code folder BuildNumber) → **System Defined Variables**

Environment variable: user input that controlled behavior of running process

No permitted pipelines

This resource cannot be used in a YAML pipeline until at least one pipeline has permission.
All Classic pipelines can use this resource.






```
This pipeline needs
permission to access
a resource before
this run can continue
```

Templates let you define reusable content, logic, and parameters in YAML pipelines

Best Practices

- **Use parameterization**: This helps in defining a generic template that can be used across multiple pipelines with different configurations.
- **Keep templates in a centralized location**: Store your reusable YAML templates in a specific repository or folder to make management easier.
- **Use naming conventions**: Define clear names for your templates, so it's easy to identify their purpose (e.g., `build-template.yml`, `deploy-template.yml`).
- **Version control for templates**: You can create versioned templates and pin your pipeline to a specific version to avoid accidental breaking changes.

prebuilt itinerary → **website** → **download** → **make changes based on your needs** → **enjoy your vacations**

Libraries → **package manager (pip - python**

NPM - NodeJS

Maven / Gradle - JAVA

apt - ubuntu

YUM - RHEL)

→ **download** → **re-use them in my code** → **Build your app**

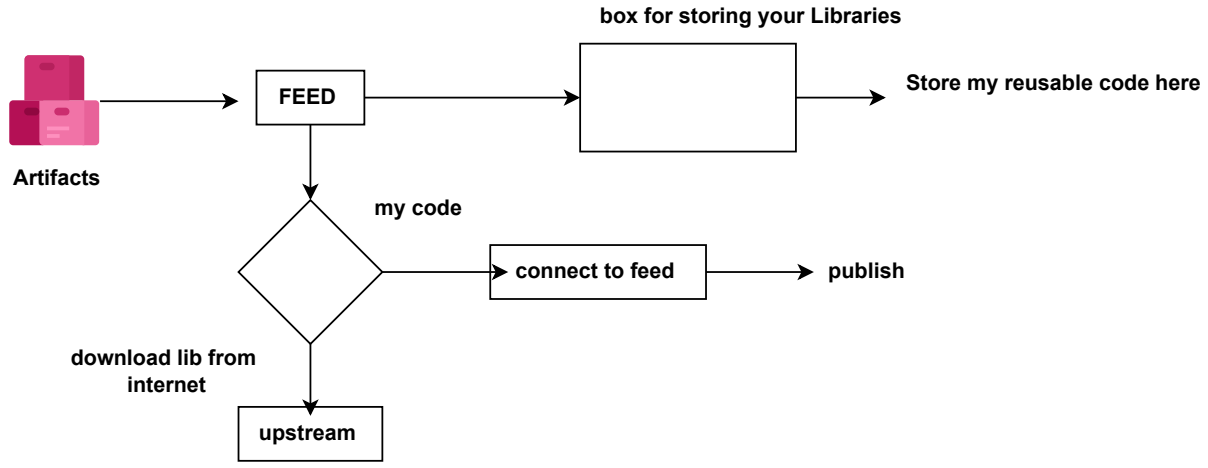
Compliant or security concerned organization will allow the developers to download packages directly from Internet ??

LOTS OF Firewall Rules needs to be opened / removed

Azure Artifacts provides developers with a streamlined way to manage all their dependencies from a single feed. These feeds serve as repositories for storing, managing, and sharing packages, whether within your team, across organizations, or publicly online.

Azure Artifacts supports multiple package types, including NuGet, npm, Python, Maven, Cargo, and Universal Packages.

Azure Artifacts feeds are organizational constructs that enable you to store, manage, and share your packages while maintaining access control. Feeds are not limited to specific package types; you can store a variety of packages, such as npm, NuGet, Maven, Python, Cargo, and Universal Packages in a single feed



Create new feed

Feeds host your packages and let you control permissions.

Name

AJ-s-Re-Use-ABLE-Packages

Visibility

- (●) Members of your Microsoft Entra tenant
  Any member of your Microsoft Entra tenant can view the packages in this feed
- ( ) Members of GD-CS-SUBM-0420
  Any member of your organization can view the packages in this feed
- ( ) Specific people
  Only users you grant access to can view the packages in this feed

Upstream sources

[x] Include packages from common public sources

For example: nuget.org, npmjs.com

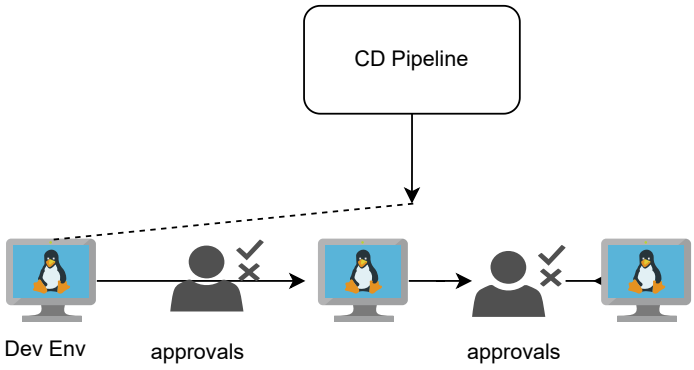
Scope

- (●) Project: test (Recommended)
  The feed will be scoped to the test project.

GitHub Packages is a software package hosting service that allows you to host your software packages privately or publicly and use packages as dependencies in your projects.

Support for package registries

Language	Description	Package format	Package client
JavaScript	Node package manager	`package.json`	`npm`
Ruby	RubyGems package manager	`Gemfile`	`gem`
Java	Apache Maven project management and comprehension tool	`pom.xml`	`mvn`
Java	Gradle build automation tool for Java	`build.gradle` or `build.gradle.kts`	`gradle`
.NET	NuGet package management for .NET	`nupkg`	`dotnet` CLI
N/A	Docker container management	`Dockerfile`	`Docker`



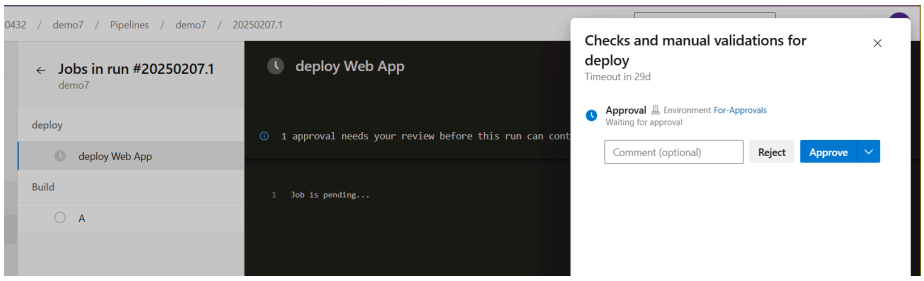
Azure Policy helps to enforce organizational standards and to assess compliance at-scale. Through its compliance dashboard, it provides an aggregated view to evaluate the overall state of the environment, with the ability to drill down to the per-resource, per-policy granularity.

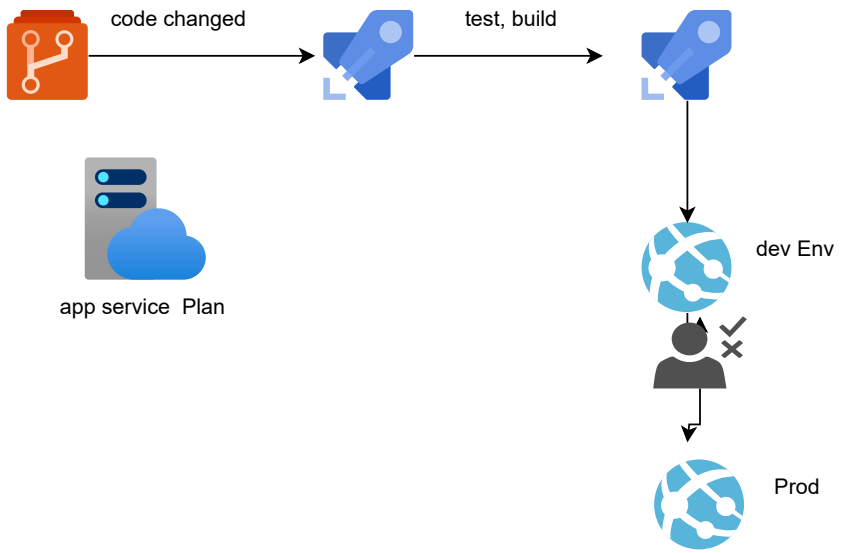
Classic Release Pipeline

Teams can also take advantage of the Approvals and Gates feature to control the workflow of the deployment pipeline. Each stage in a release pipeline can be configured with pre-deployment and post-deployment conditions that can include waiting for users to manually approve or reject deployments, and checking with other automated systems that specific conditions are met.

A user must manually validate the change request and approve the deployment to a certain stage.	Pre-deployment approvals
A user must manually sign out after deployment before the release is triggered to other stages.	Post-deployment approvals

An environment represents a logical target where your pipeline deploys software. Typical environment names are Dev, Test, QA, Staging, and Production.





Pre-deployment conditions

Prod

Triggers

Define the trigger that will start deployment to this stage

Pre-deployment approvals

Enabled

Select the users who can approve or reject deployments to this stage

Approvers

student1CPDA | Search users and groups for approvers

Timeout

30 | Days

Approval policies

- [ ] The user requesting a release or deployment should not approve it
- [ ] Revalidate identity of approver before completing the approval
- [ ] Skip approval if the same approver approved the previous stage

RELEASE GATES

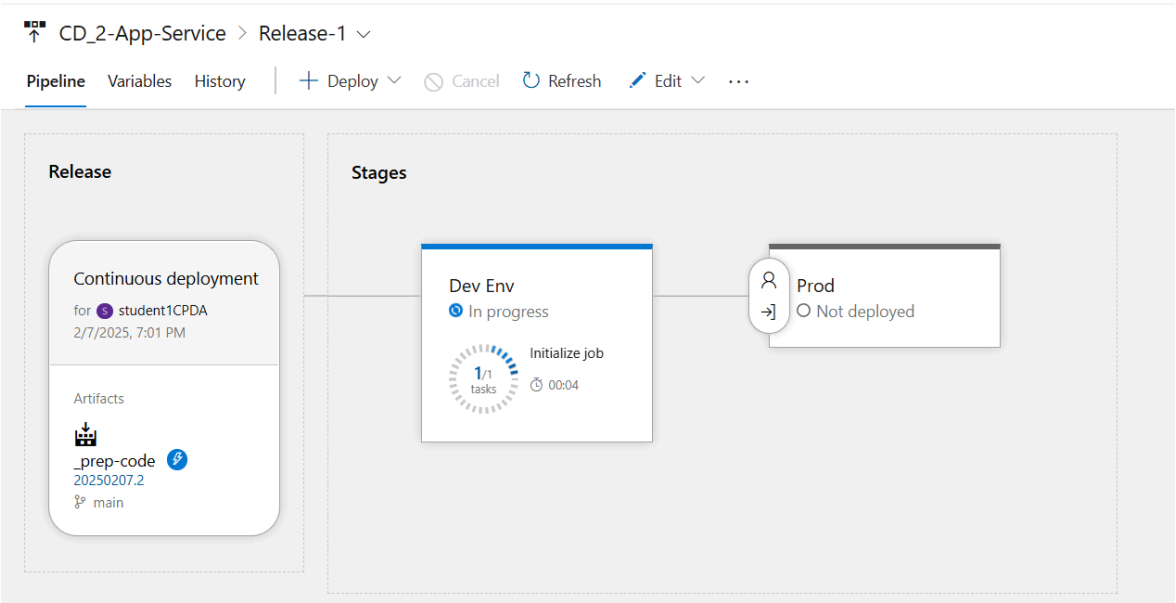
Check Azure Policy compliance
Security and compliance assessment for Azure Policy

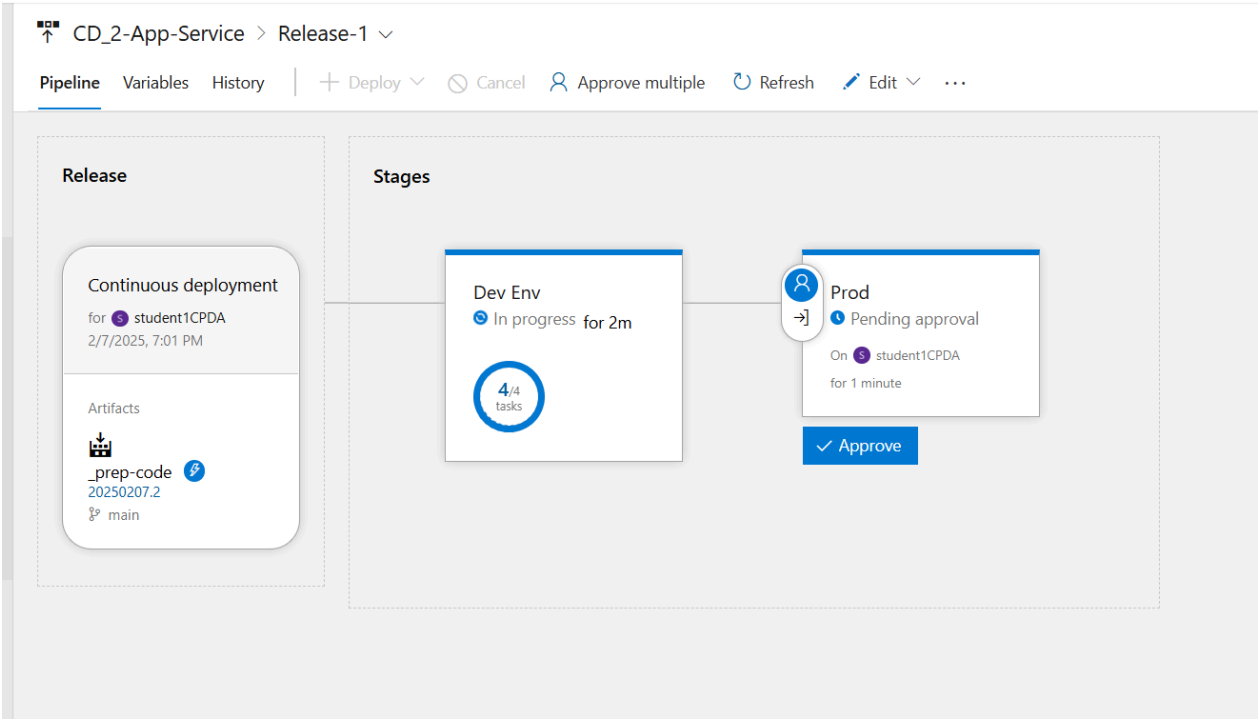
Invoke Azure Function
Invoke an Azure Function

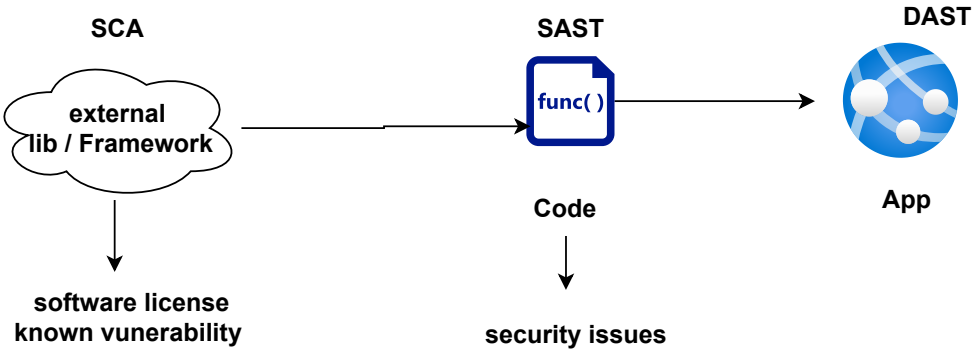
Invoke REST API
Invoke a REST API as a part of your pipeline.

Query Azure Monitor alerts
Observe the configured Azure Monitor rules for active alerts

Query work items
Execute a work item query and check the number of items returned


CD_2-App-Service > Release-1
Pipeline
Variables
History
Deploy
Cancel
Refresh
Edit
Release
Stages
Continuous deployment
for student1CPDA
2/7/2025, 7:01 PM
Artifacts
_prep-code
20250207.2
main
Dev Env
In progress
1/1 tasks
Initialize job
00:04
Prod
Not deployed


CD_2-App-Service > Release-1
Pipeline
Variables
History
Deploy
Cancel
Approve multiple
Refresh
Edit
Release
Stages
Continuous deployment
for student1CPDA
2/7/2025, 7:01 PM
Artifacts
_prep-code
20250207.2
main
Dev Env
In progress for 2m
4/4 tasks
Prod
Pending approval
On student1CPDA
for 1 minute
Approve


SCA
external
lib / Framework
software license
known vunerability
SAST
func( )
Code
security issues
DAST
App

SAST:

1. Sonarqube
2. GitHub Advanced Security - CodeQL
3. Gitlab SAST
4. ESLint
5. Bandit
6. OWASP dependency check
7. Trivy (Container image scanning)

DAST

1. OWASP ZAP
2. Burpsuite
3. Checkmarx
4. Wapiti
5. Arachni

SCA:
mend.io

DevSecOps: Automation + Security

- Add tasks in your pipeline
- generate and Review Reports
- take actions to solve problems

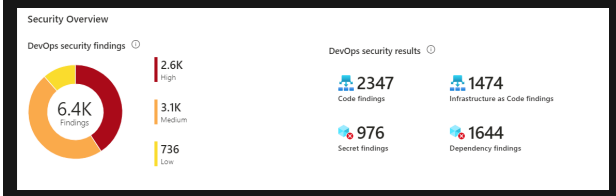
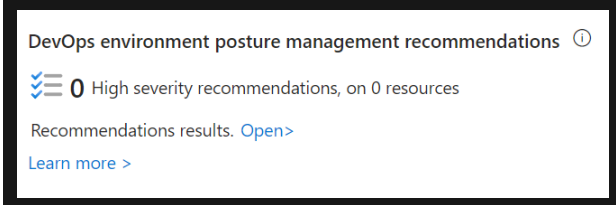
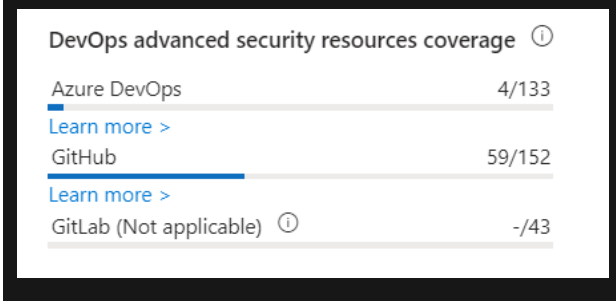
	Public repository	Private repository without Advanced Security	Private repository with Advanced Security
Code scanning	✓	✗	✓
CodeQL CLI	✓	✗	✓
Secret scanning	✓	✗	✓
Custom auto-triage rules	✓	✗	✓
Dependency review	✓	✗	✓

A GitHub Advanced Security license provides the following additional features for private repositories:

- **Code scanning** - Search for potential security vulnerabilities and coding errors in your code using CodeQL or a third-party tool. See About code scanning and About code scanning with CodeQL.
- **CodeQL CLI** - Run CodeQL processes locally on software projects or to generate code scanning results for upload to GitHub. See About the CodeQL CLI.
- **Secret scanning** - Detect secrets, for example keys and tokens, that have been checked into private repositories. If push protection is enabled, GitHub also detects secrets when they are pushed to your repository. Secret scanning alerts for users and push protection are available and free of charge for all public repositories on GitHub.com. See About secret scanning and About push protection.
- **Custom auto-triage rules** - Help you manage your Dependabot alerts at scale. With custom auto-triage rules you have control over the alerts you want to ignore, snooze, or trigger a Dependabot security update for. For more information, see About Dependabot alerts and Customizing auto-triage rules to prioritize Dependabot alerts.
- **Dependency review** - Show the full impact of changes to dependencies and see details of any vulnerable versions before you merge a pull request. See About dependency review.

https://learn.microsoft.com/en-us/azure/defender-for-cloud/defender-for-devops-introduction

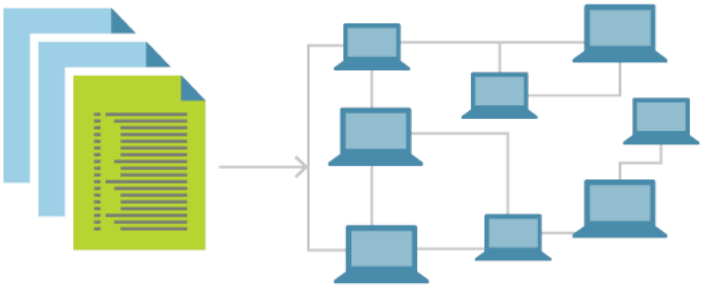
Microsoft Defender for Cloud DevOps security

Page section	Description
	Total number of DevOps security scan findings (code, secrets, dependency, infrastructure-as-code) grouped by severity level and by finding type.
	Provides visibility into the number of DevOps environment posture management recommendations highlighting high severity findings and number of affected resources.
	Provides visibility into the number of DevOps resources with advanced security capabilities out of the total number of resources onboarded by environment.

Infrastructure as code (IaC) is the ability to provision and support your computing infrastructure using code instead of manual processes and settings. Any application environment requires many infrastructure components like operating systems, database connections, and storage.

if you're working on larger scale.

Then code is your best friend.

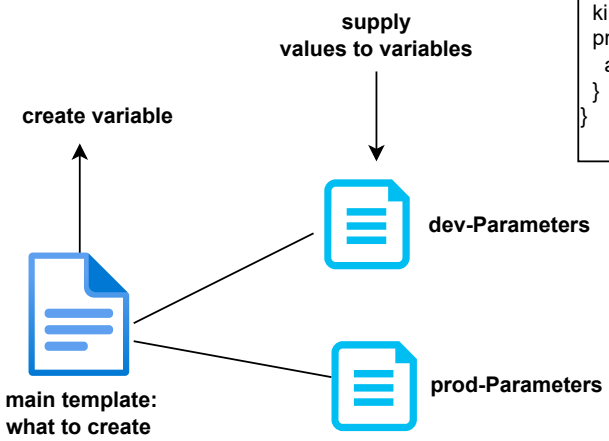


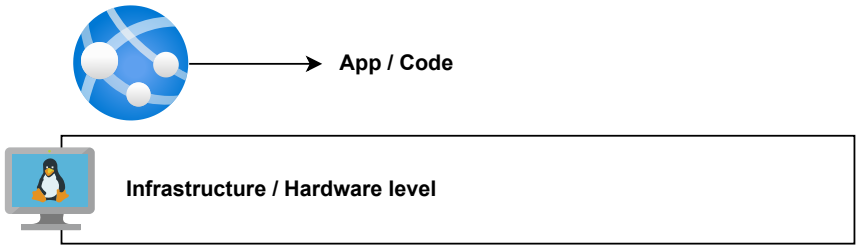
Azure provides native support for IaC via the Azure Resource Manager model. Teams can define declarative ARM templates using JSON syntax or Bicep to specify the infrastructure required to deploy solutions. Third-party solutions like Terraform through specific Azure providers are also available.

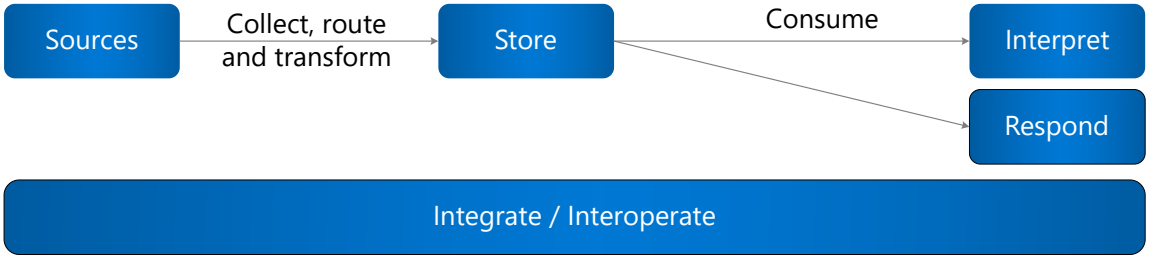
```
{
 "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",
 "parameters": {
  "location": {
   "type": "string",
   "defaultValue": "[resourceGroup().location]"
  },
  "storageAccountName": {
   "type": "string",
   "defaultValue": "[format('toylaunch{0}', uniqueString(resourceGroup().id))]"
  }
 },
 "resources": [
  {
   "type": "Microsoft.Storage/storageAccounts",
   "apiVersion": "2023-05-01",
   "name": "[parameters('storageAccountName')]",
   "location": "[parameters('location')]",
   "sku": {
    "name": "Standard_LRS"
   },
   "kind": "StorageV2",
   "properties": {
    "accessTier": "Hot"
   }
  }
 ]
}
```

Bicep

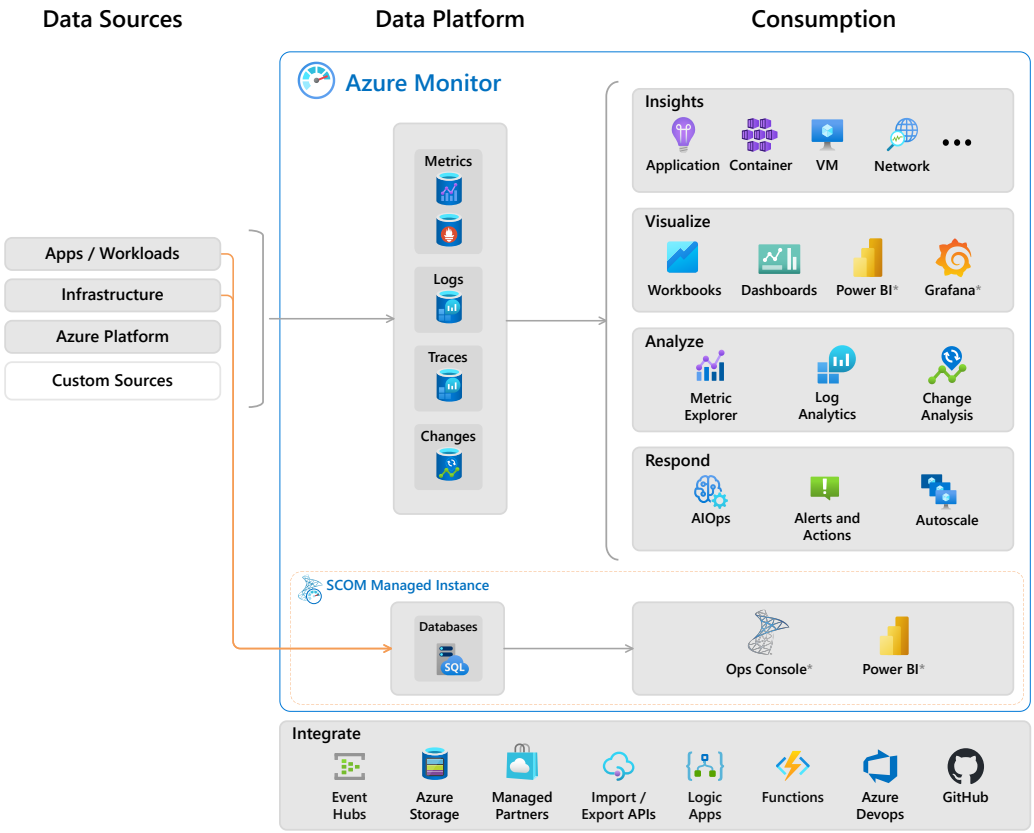
```
param location string = resourceGroup().location
param storageAccountName string = 'toylaunch${uniqueString(resourceGroup().id)}'


resource storageAccount 'Microsoft.Storage/storageAccounts@2023-05-01' = {
  name: storageAccountName
  location: location
  sku: {
    name: 'Standard_LRS'
  }
  kind: 'StorageV2'
  properties: {
    accessTier: 'Hot'
  }
}
```

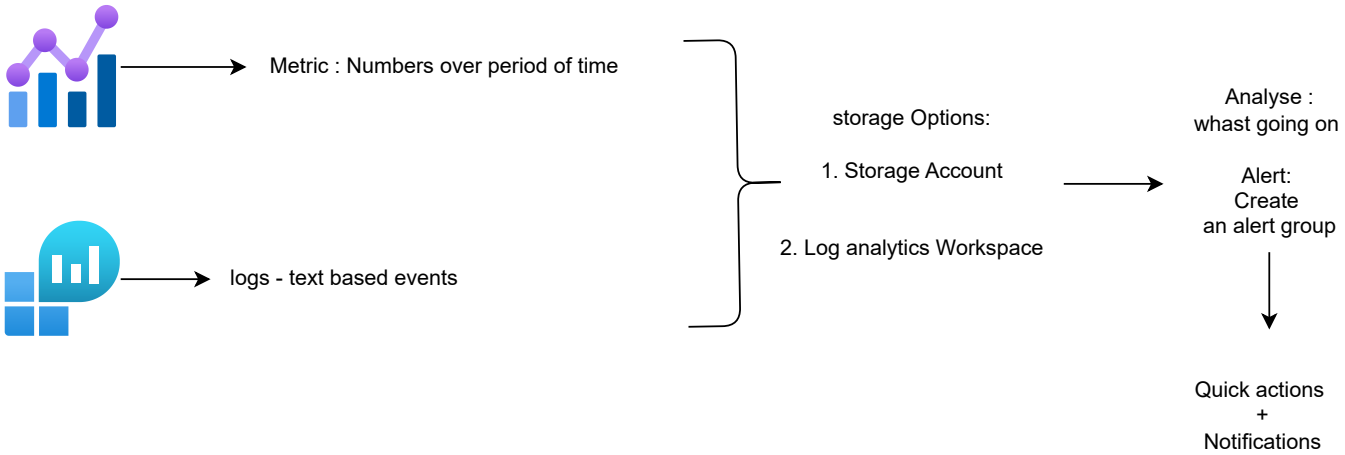




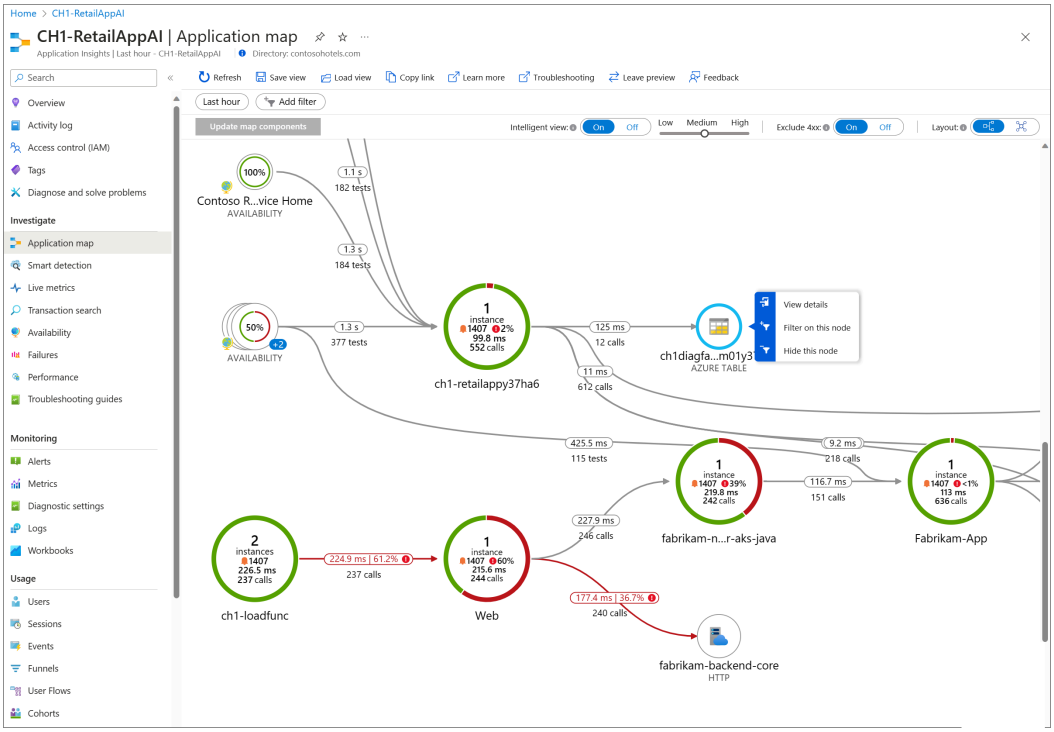


Azure Monitor is a comprehensive monitoring solution for collecting, analyzing, and responding to monitoring data from your cloud and on-premises environments. You can use Azure Monitor to maximize the availability and performance of your applications and services. It helps you understand how your applications are performing and allows you to manually and programmatically respond to system events.





Kusto Query Language (KQL) is a powerful tool to explore your data and discover patterns, identify anomalies and outliers, create statistical modeling, and more. KQL is a simple yet powerful language to query structured, semi-structured, and unstructured data.



nvestigate

- Application dashboard: An at-a-glance assessment of your application's health and performance.
- Application map: A visual overview of application architecture and components' interactions.
- Live metrics: A real-time analytics dashboard for insight into application activity and performance.
- Transaction search: Trace and diagnose transactions to identify issues and optimize performance.
- Availability view: Proactively monitor and test the availability and responsiveness of application endpoints.
- Failures view: Identify and analyze failures in your application to minimize downtime.
- Performance view: Review application performance metrics and potential bottlenecks.