

Azure DevOps



Section 1 part 1

Waterfall method

Requirements

Of the system need
be developed are
documented

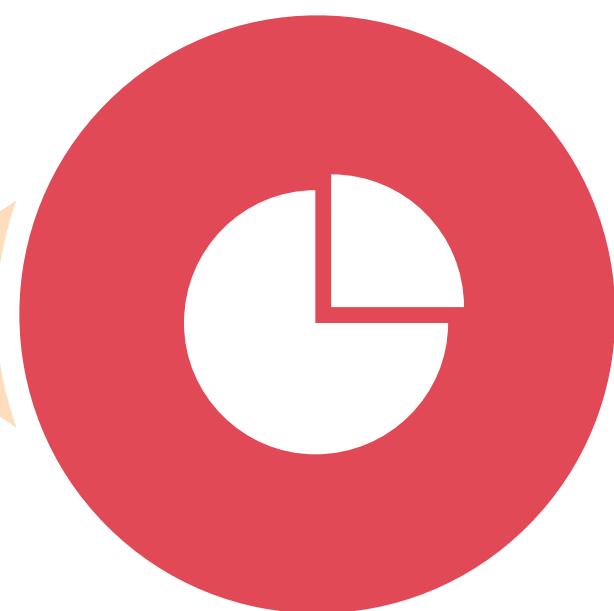


Helps in specifying
the hardware and
system
requirements

Design

Development

Developers
develops the code
for the application

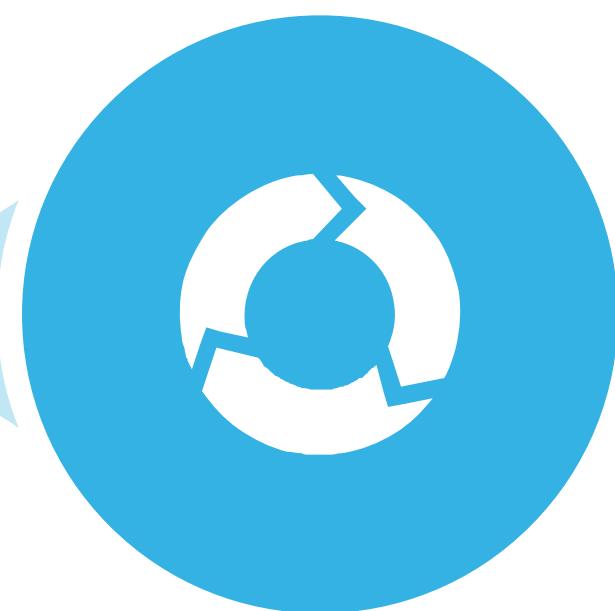


The entire system is
tested for any faults
and failures.

Testing

Deployment

App is deployed in
the customer
environment/
Market



To fix Issues
patches are
released

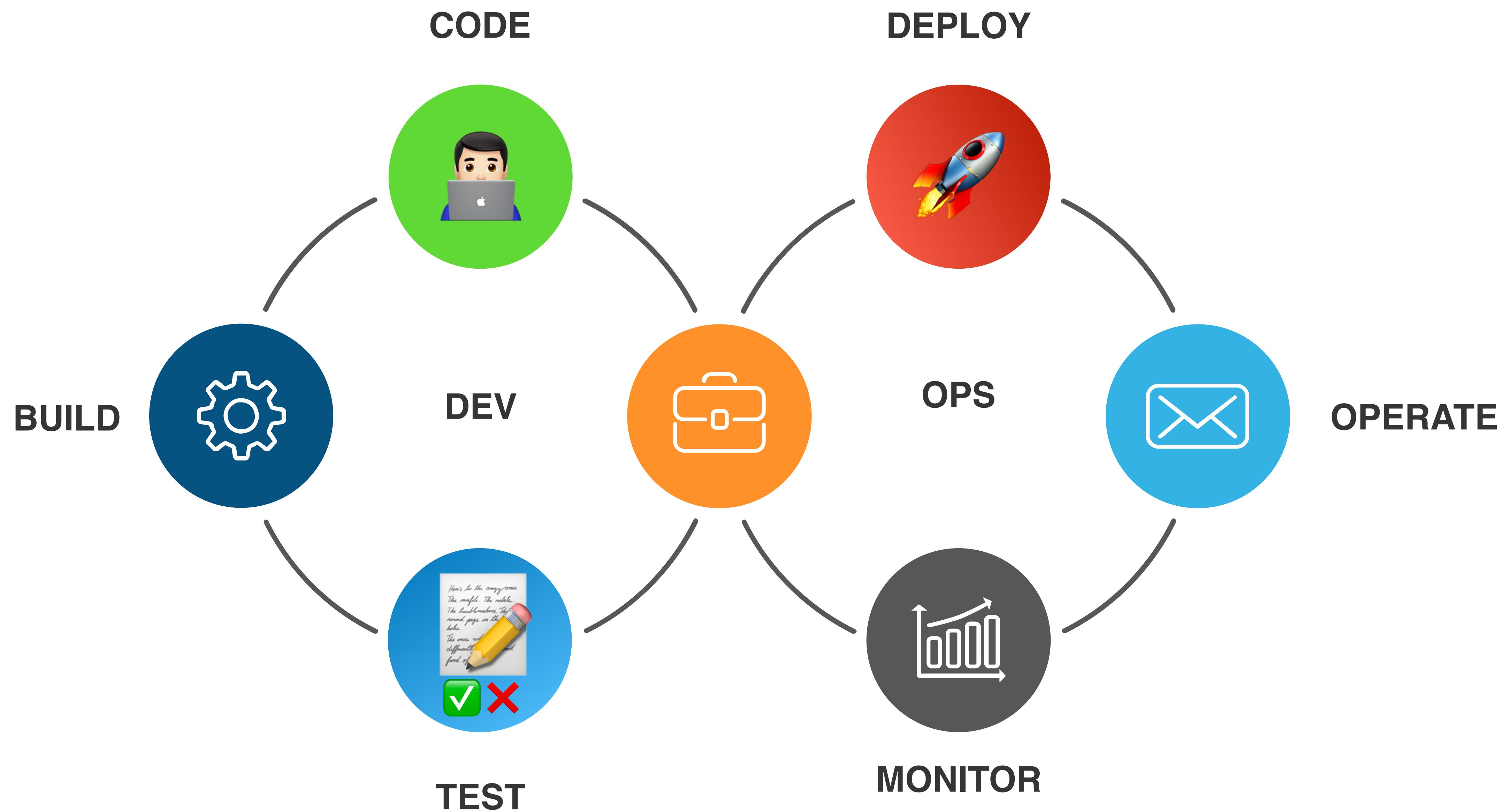
Maintenance



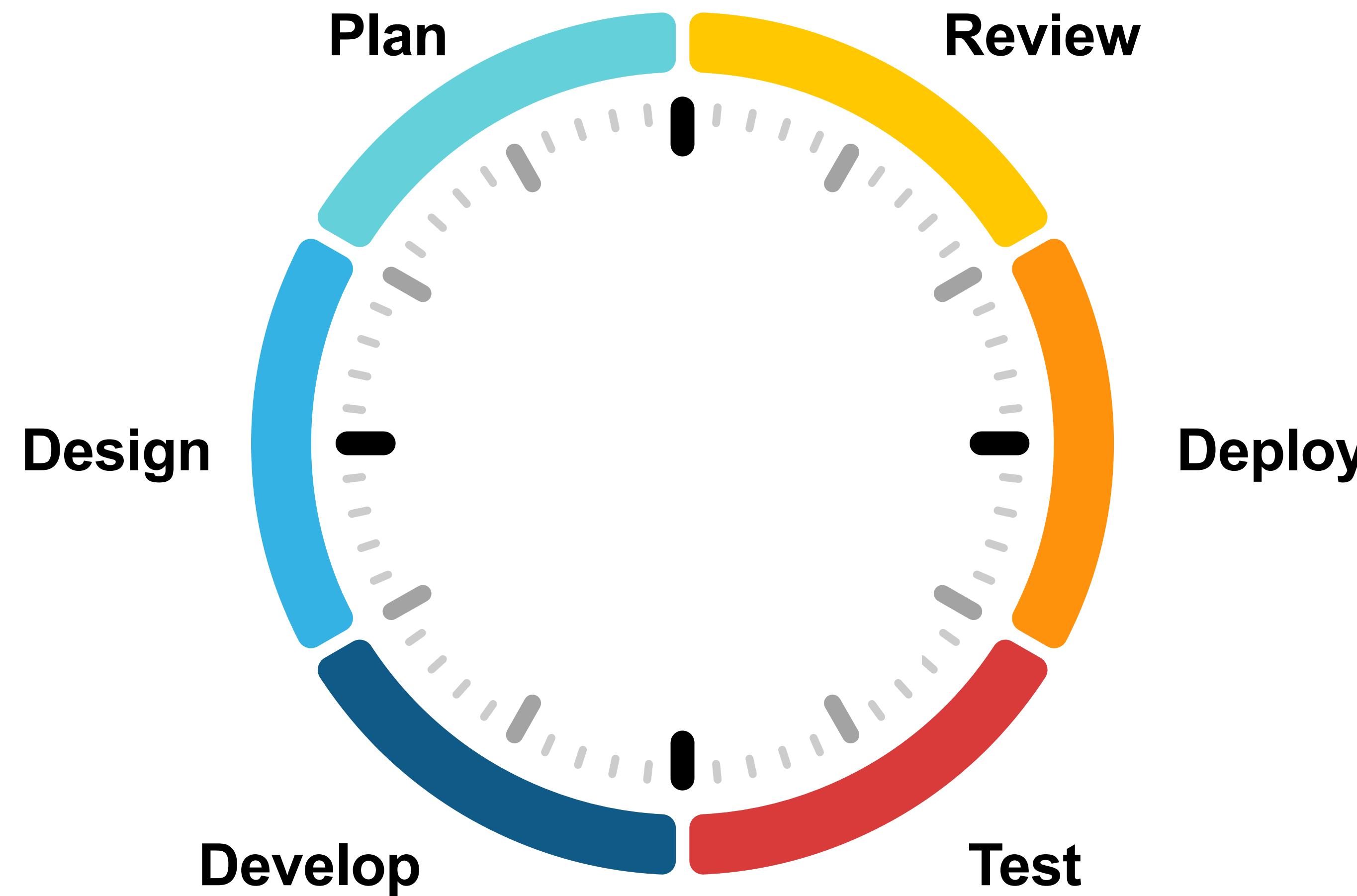
Problems by Waterfall Methodology



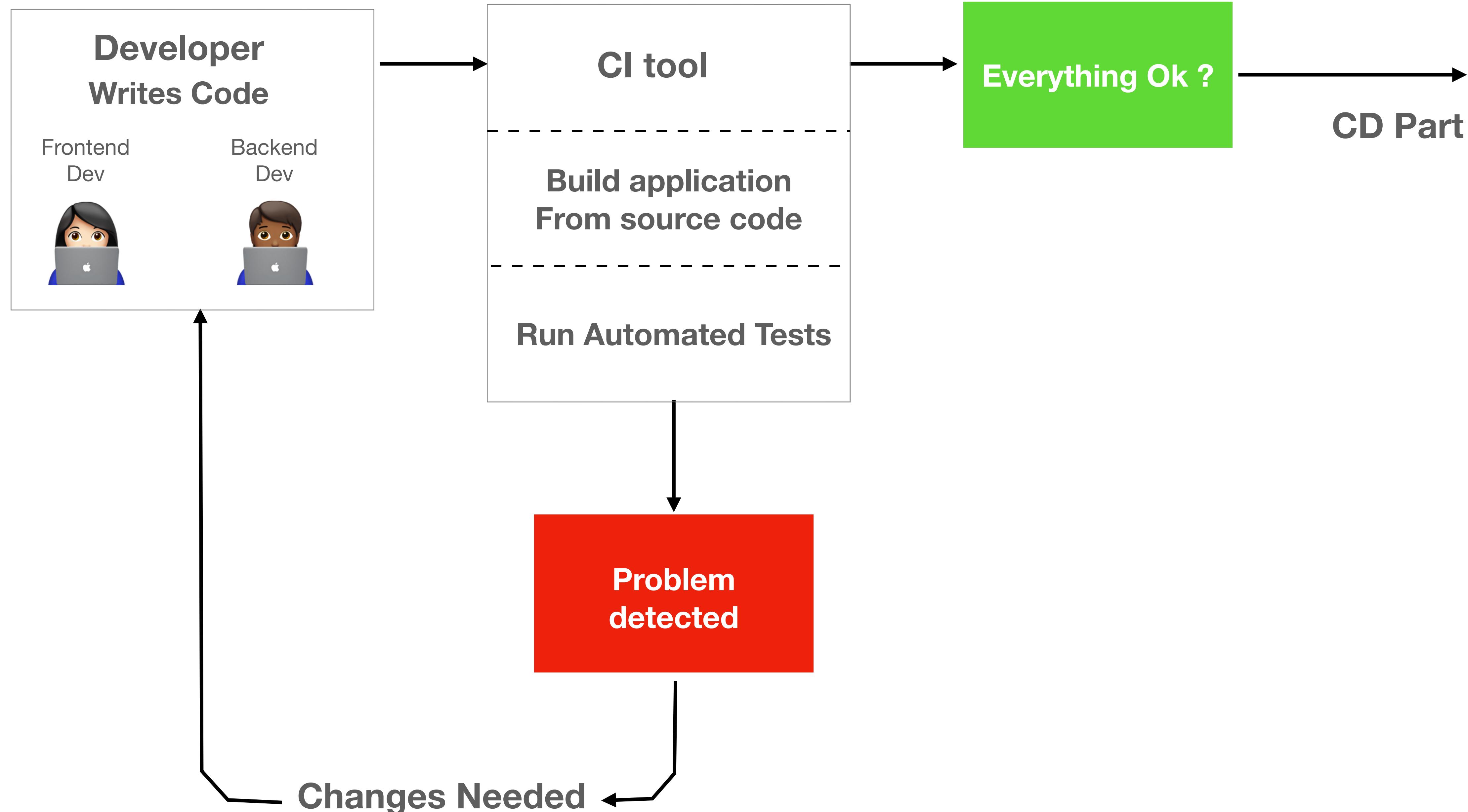
DevOps 🤔

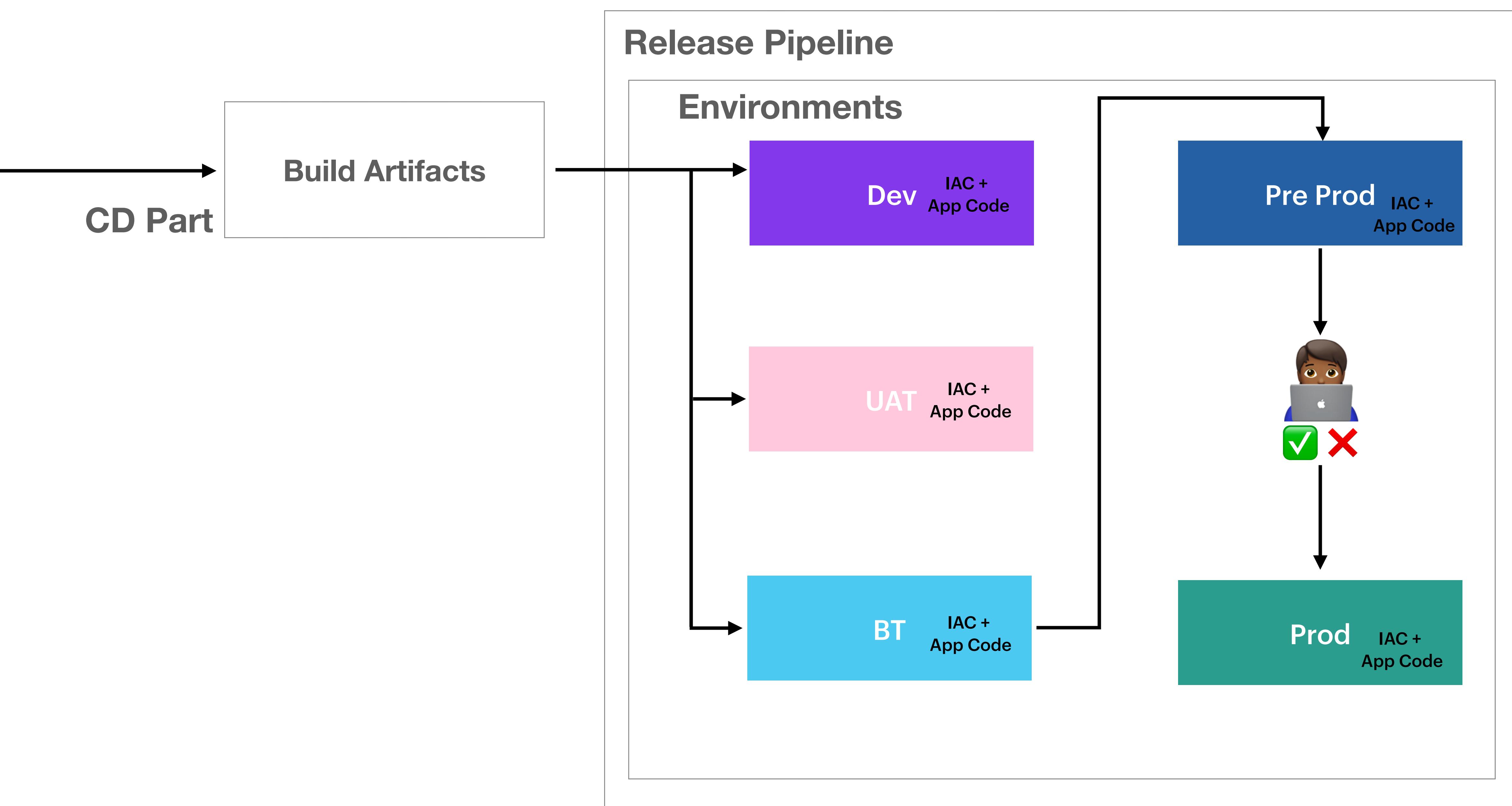


Agile Methodology

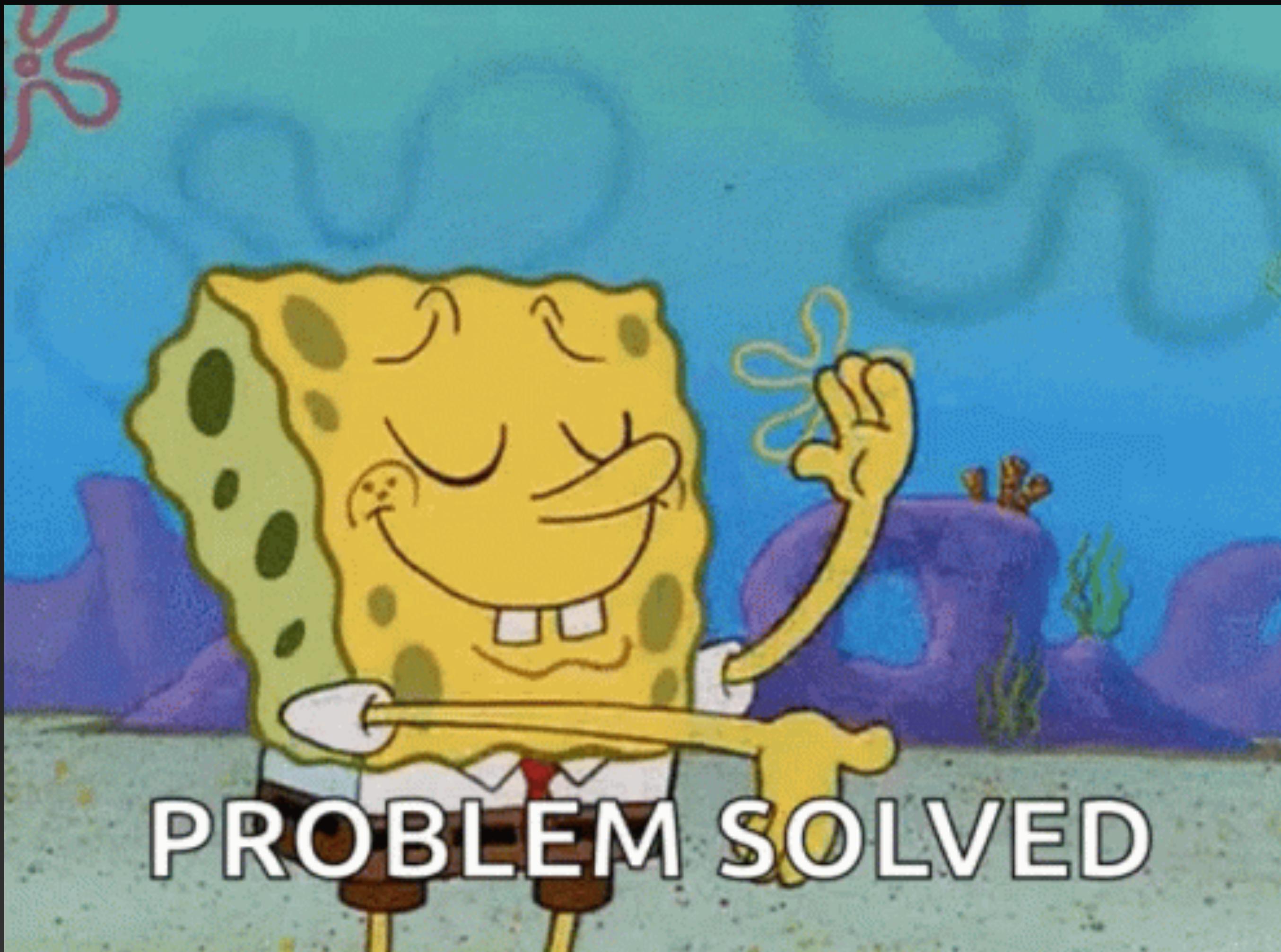


Continuous Integration





Problems solved by DevOps



01



Bringing value to
customers with
faster production
cycles



02



Increased
productivity results
in higher customer
satisfaction



03



Saving costs
through faster time
to market



04



Enhanced quality
with test
automation



05

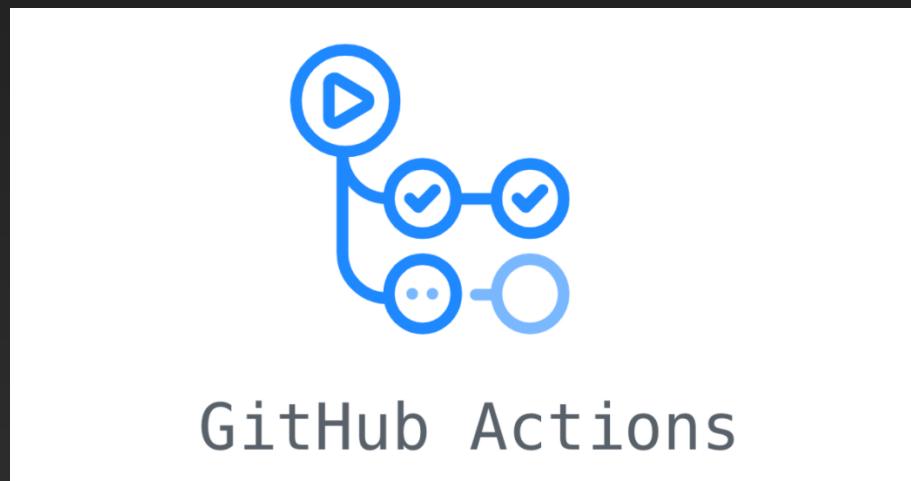


Improving problem
resolution





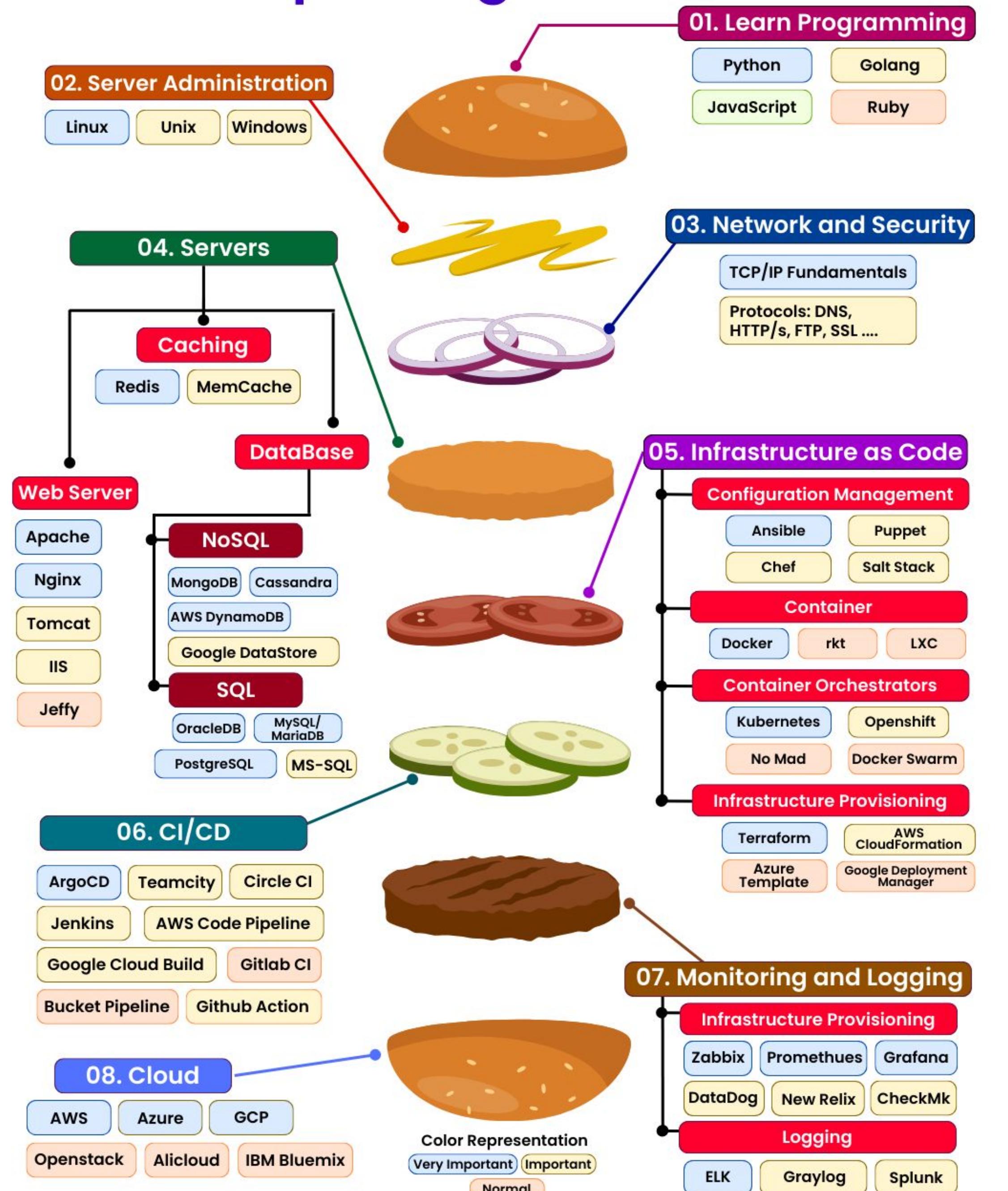
DevOps Tools



many more ...

DevOps Periodic Table

The DevOps Burger !!



Requirements

- Azure cloud knowledge
- Basic knowledge on build steps for programming language



Azure DevOps

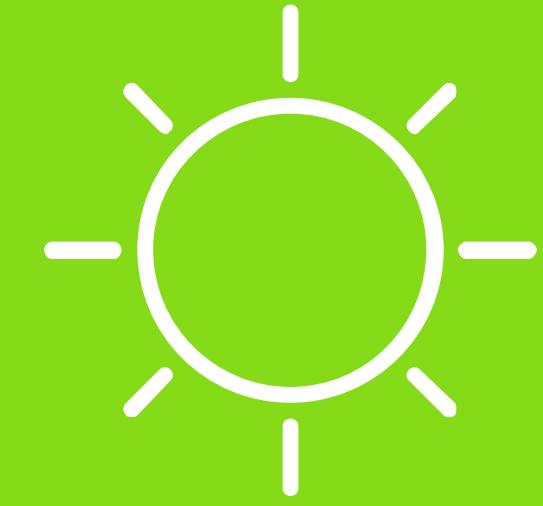




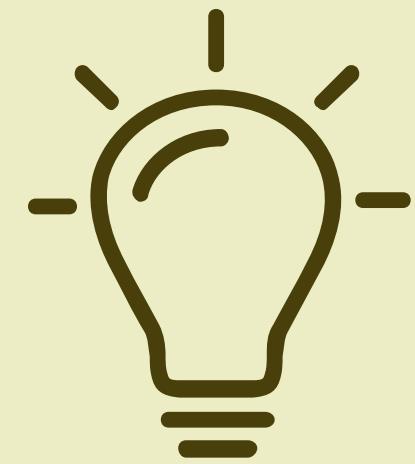
Azure Boards

- Azure Boards is a web-based service that enables teams to plan, track, and discuss work across the entire development process
- Azure Boards provides a customizable platform for managing work items, allowing teams to collaborate effectively and streamline their workflow

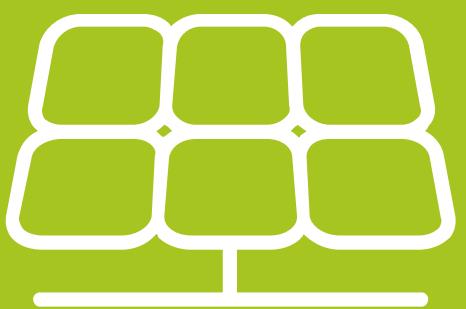
Basic



Agile

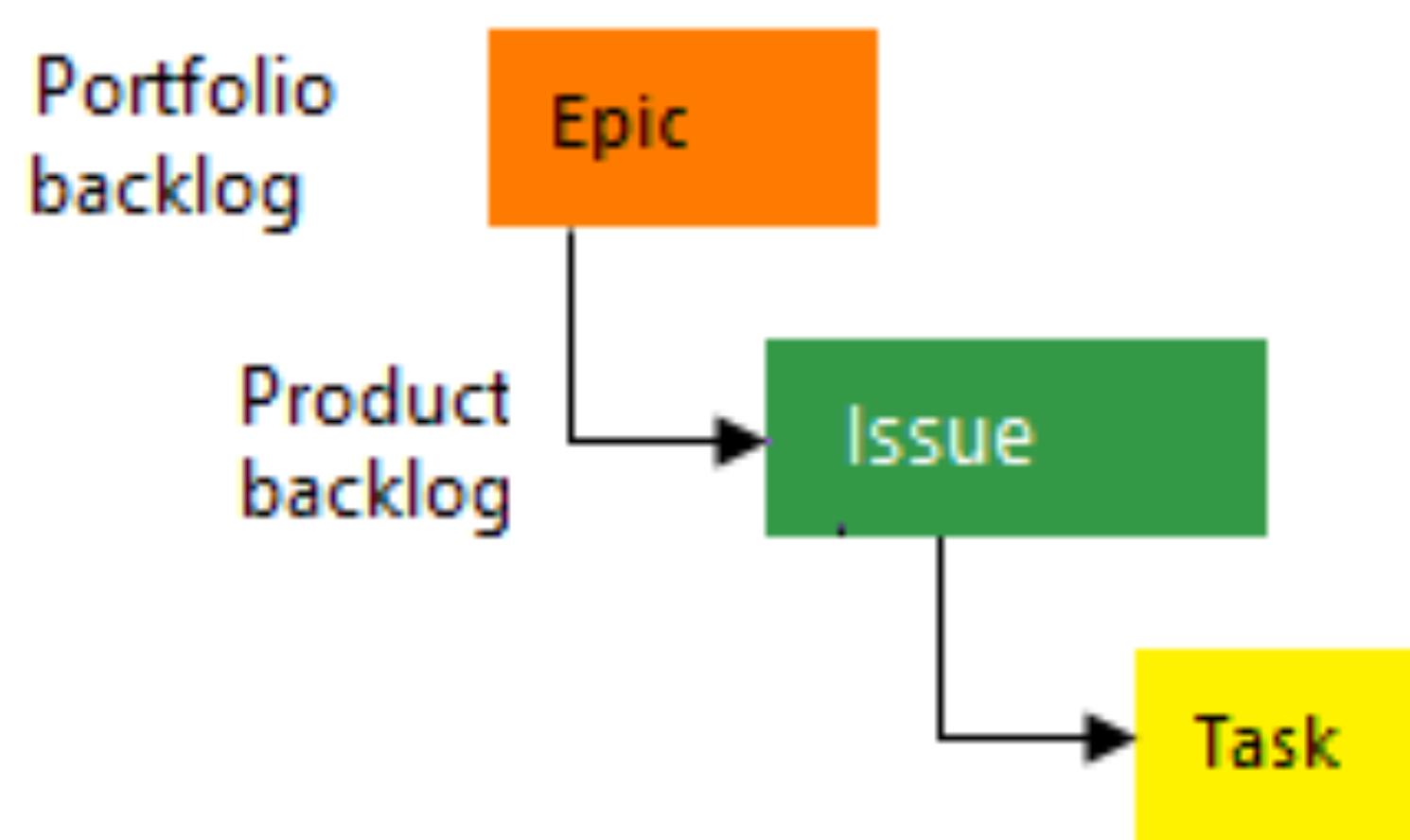


Scrum



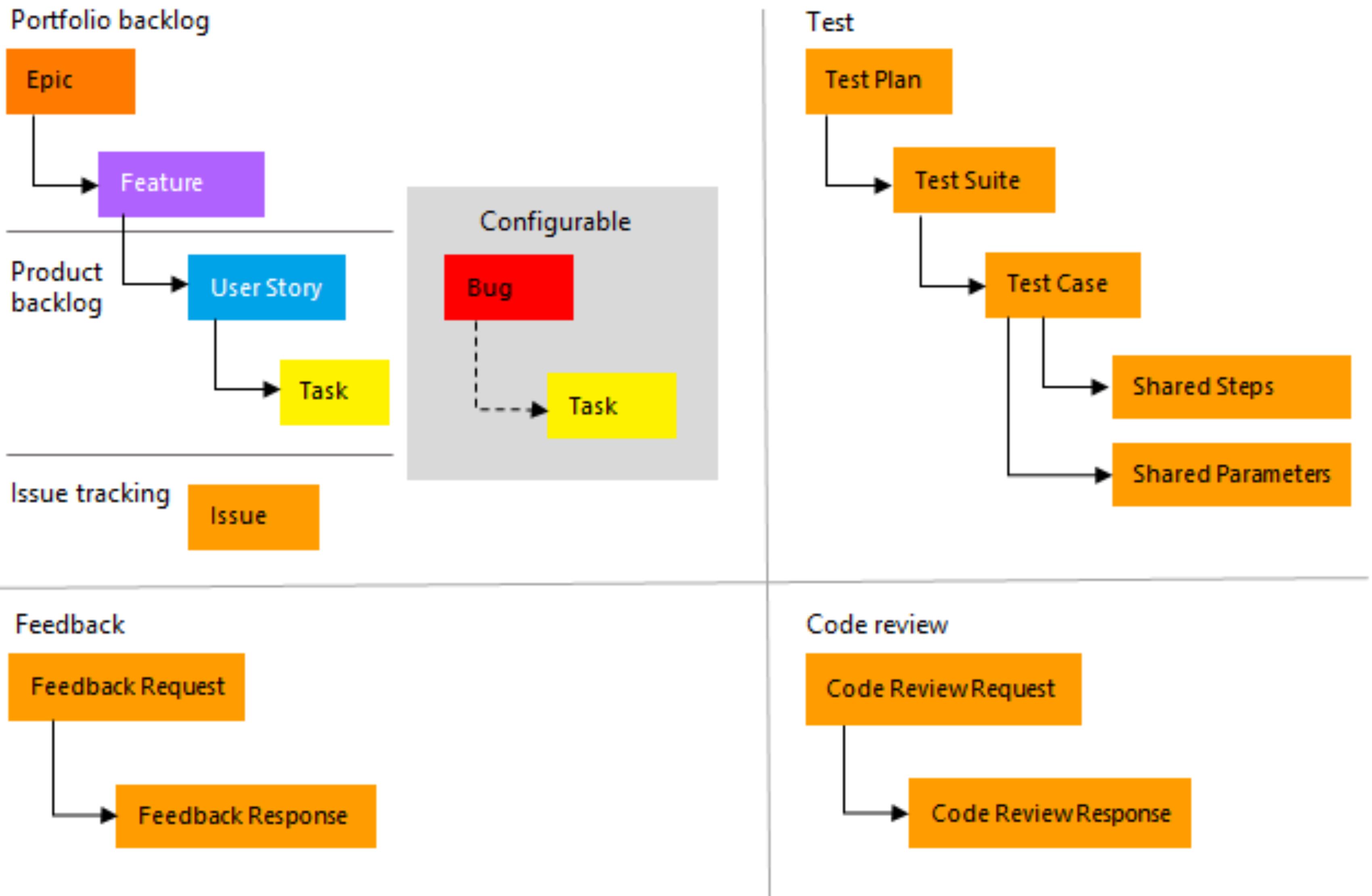
CMMI

Basic Process

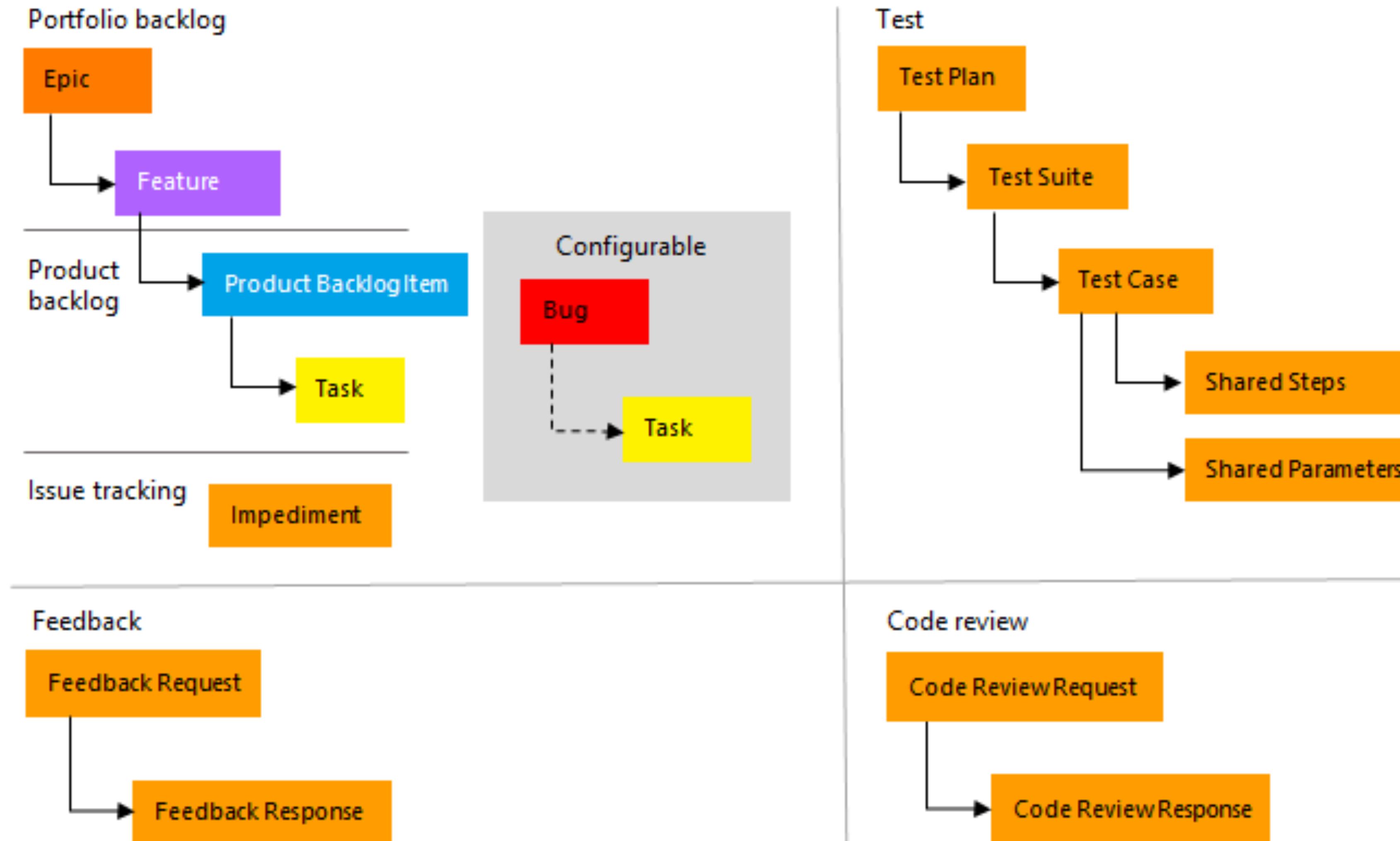


State	Title
● To Do	▼ 🏰 Web site updates
● Doing	⚠ Secure sign-in
● Doing	▼ ⚠ Hello World web site
● To Do	✓ Design welcome screen
● To Do	✓ Standardize form factors
● To Do	✓ Change background color
● To Do	✓ About screen
● To Do	⚠ Welcome back page
● To Do	⚠ Change initial view
● To Do	▼ 🏰 Service status
● Doing	⚠ Resolve service status issues
● Doing	⚠ Check performance
● To Do	⚠ Change new item

Agile Process



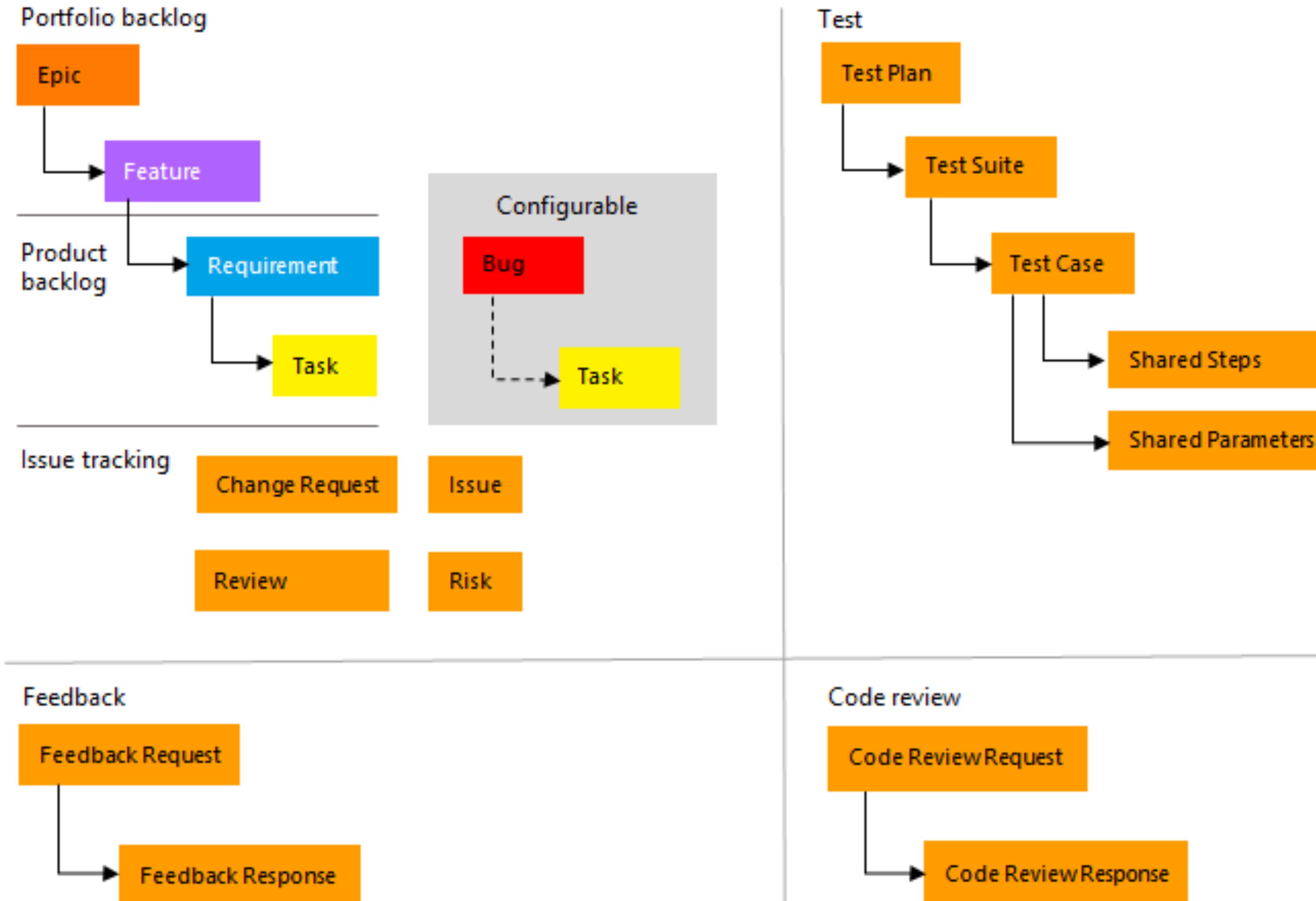
Scrum Process



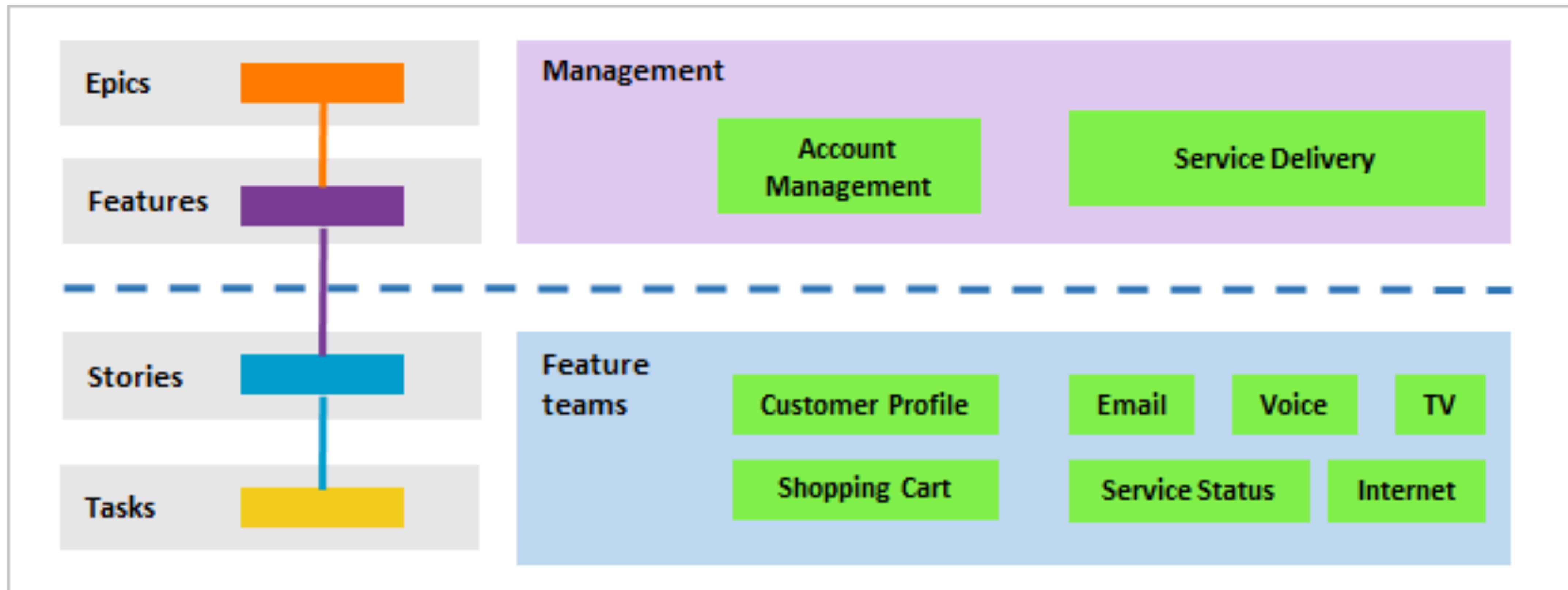
When to use Agile or Scrum

- Agile means “incremental, allowing teams to develop projects in small increments.
- Scrum is one of the many types of agile methodology
- Scrum is an iterative ongoing cycle of designing, developing, testing, rinse and repeat. Which include sprints
- agile emphasizes time and supporting team members to get X number of tasks completed by the end of the designated period of time.

CMMI Process (Capability Maturity Model Integration)



Teams hierarchy



Kanban board

The screenshot shows the Azure DevOps interface for the project "fabrikam / Fabrikam Fiber". The left sidebar has a red box around the "Boards" item under the "Backlogs" section. The main area shows the "Fabrikam Fiber Team" backlog with a card for "Add an information form" by Raisa Pokrovskaya, assigned to Johnnie McLeod, in Sprint 3. A red box highlights the "Backlog items" link in the top navigation bar.

Azure DevOps fabrikam / Fabrikam Fiber

Fabrikam Fiber +

FF Fabrikam Fiber Team Backlog items

Overview

Backlog Analyze 5/10 Develop

New item

Add an information form Raisa Pokrovskaya Iteration ... Sprint 3 0/2

Welcome back page Johnnie McLeod Iteration ... Sprint 3 0/4 1

Slow form Jam 0/2

Boards

Work Items

Boards

Backlogs

Sprints

- **Product backlog:** By default lists User Stories (Agile), Issues (Basic), Product Backlog Items and Bugs (Scrum), or Requirements (CMMI). Provides options to show **Parents**, **Forecast**, and **In Progress** or **Completed** child items.
- **Portfolio backlog:** By default lists Features (all process models) for the Features backlog, and Epics (Agile, Scrum, and CMMI) for the Epic backlog. Provides options to show **Parents** and **In Progress** or **Completed** child items.
- **Sprint backlog:** By default lists all product backlog items assigned to the selected iteration, regardless of status. Provides options to show **Work details**.

Boards walkthrough

Design Agile Team ▾ ⭐ ⓘ

Board Analytics View as Backlog Stories ▾ ⚙️ 🔍 🎯 🚧 ↗

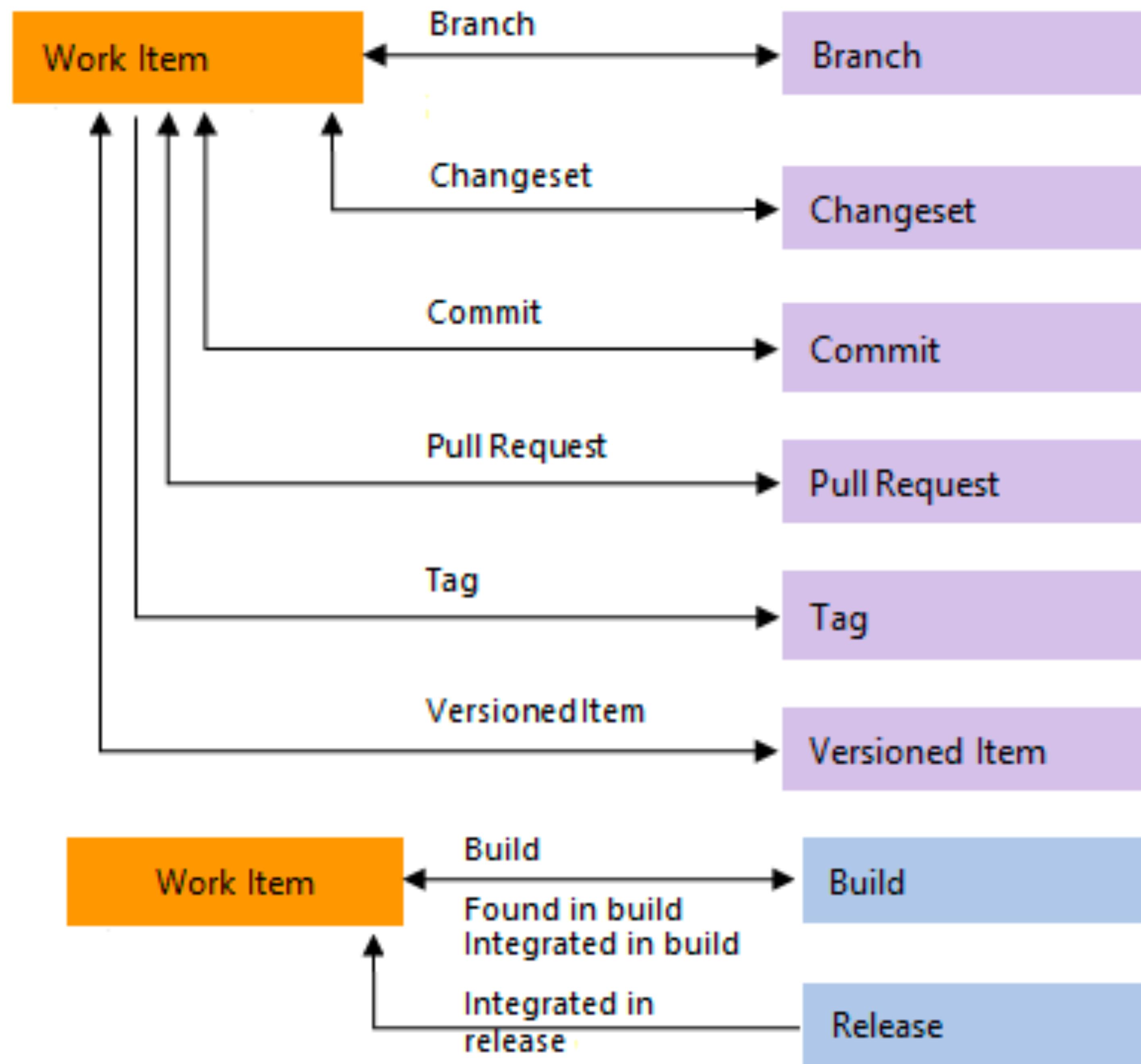
New < Active 2/5 Resolved 1/5 Closed <

	Item ID	Description	Assignee	Priority	State	Parent	Completed Tasks	Total Tasks
New item		950 Slow response on information form	Jia-hao Tseng	3	New	Customer Web -...	0/2	1
	951	Hello World Web Site	Raisa Pokrovskaya	5	Active	Customer Web -...	0/3	
	952	Change initial view	Christie Church	3	Resolved	Customer Web -...	+ Add Task	
	954	Design feedback interface	Helena Petersen	8	Closed	Mobile feedback	<input type="checkbox"/> Add switching context	
	964	Technician dashboard improvements	Christie Church	5	Active	Customer Servic...	<input type="checkbox"/> Apply security measures	
	956	Check performance	Jamal Hartnett	5	New	Refresh web loo...	<input type="checkbox"/> screen design	

GitHub to Azure Boards work items

- Connect Azure Boards to GitHub :: Projects –> GitHub connections
- AB#{ID}
- Fixed AB#123
- Fixes AB#123, AB#124, and AB#126
- Fixing multiple bugs: issue #123 and user story AB#234
- **Add link** under the Development section - PR, Commit, issue

- Capture a work item as a template
- Manage work item templates
- Boards configurations under project settings
- Add a custom field to a work item type (Inheritance process)
- <https://learn.microsoft.com/en-us/azure/devops/organizations/settings/work/add-custom-field?view=azure-devops>
- You can apply the new process to existing project
- Use Excel to create all types of workobjects (only windows)
- <https://learn.microsoft.com/en-us/azure/devops/boards/backlogs/office/bulk-add-modify-work-items-excel?view=azure-devops&tabs=agile-process>
- Bulk import using csv (Mac/ windows)



Keyboard shortcuts

- Keyboard shortcuts
- ? - to show all keyboard shortcuts

Exercises for Section 1 Part 1

- Create a agile Project , write a query to retrieve bugs
- integrate github with Azure boards, fix an issue from commit

Section 1 part 2

Some important metrics

- Burndown chart - begin with the total amount of planned work and then as work is completed graphs the remaining work.
- Velocity and velocity chart - Velocity provides a useful metric for gaining insight into how much work your team can complete during a sprint cycle.
- Burnup chart - Burnup charts track work as it is completed over time. They are useful to show the rate at which work is getting completed
- Lead Time - Lead time measures the total time elapsed from the creation of work items to their completion
- Cycle time - measures the time it takes for your team to complete work items once they begin actively working on them
- Cumulative flow diagram

Scrum best practices

- sprint planning meetings
- Set sprint goals
- Manage your technical debt
- Address any known bug debt every sprint
- Role of the Scrum Master
- Daily Scrum meetings
- Sprint review meetings
- Sprint retrospective meetings

Wikis walkthrough

Sprint backlog capacity

Work details capacity View

Custom dashboards using powerBI

Exercises for Section 1 Part 2

- Configure dashboard and add all chart types
- Assign work items in sprint and capacity planning

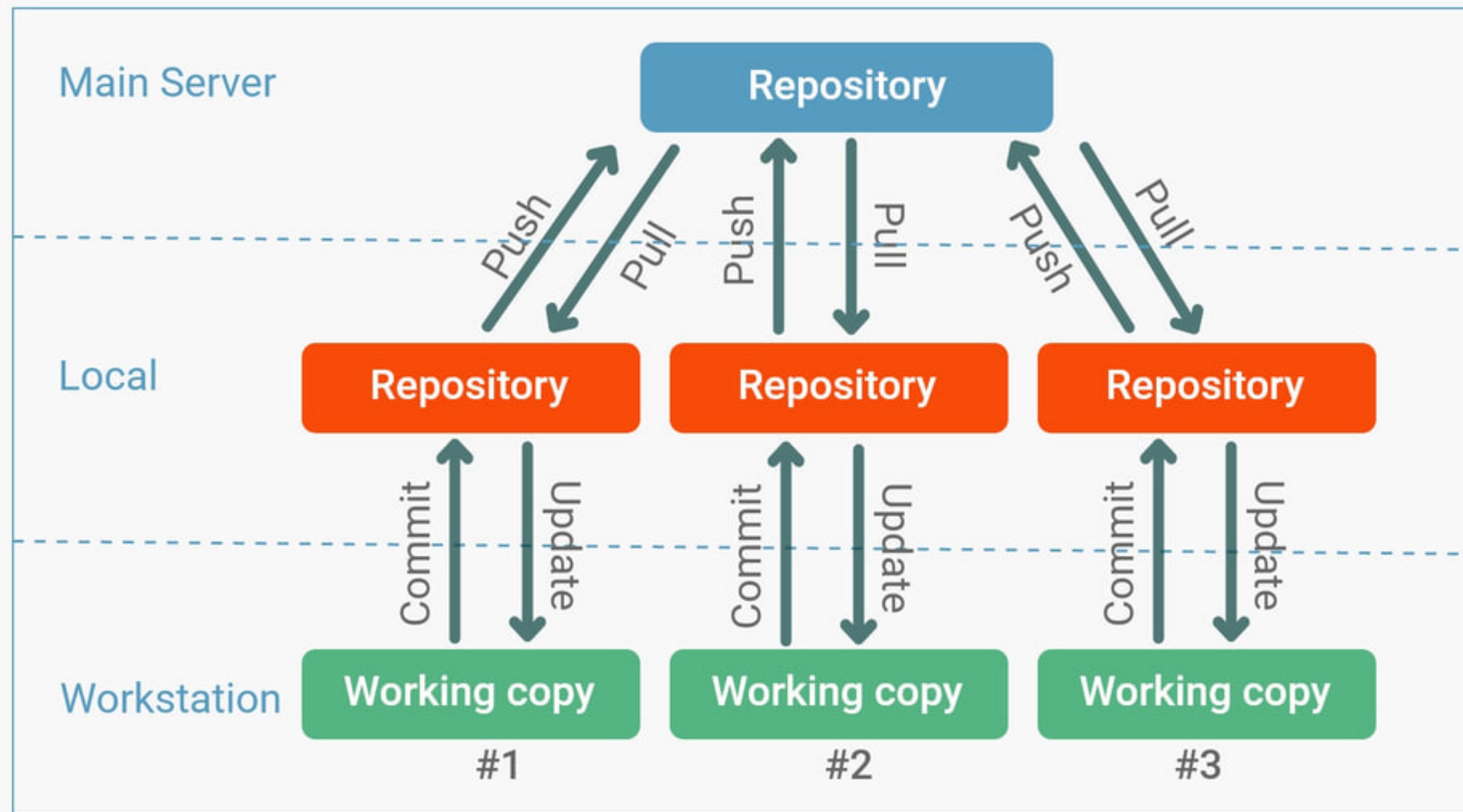
Section 2 part 1



Azure Repos

- Git: distributed version control
- Team Foundation Version Control (TFVC): centralized version control

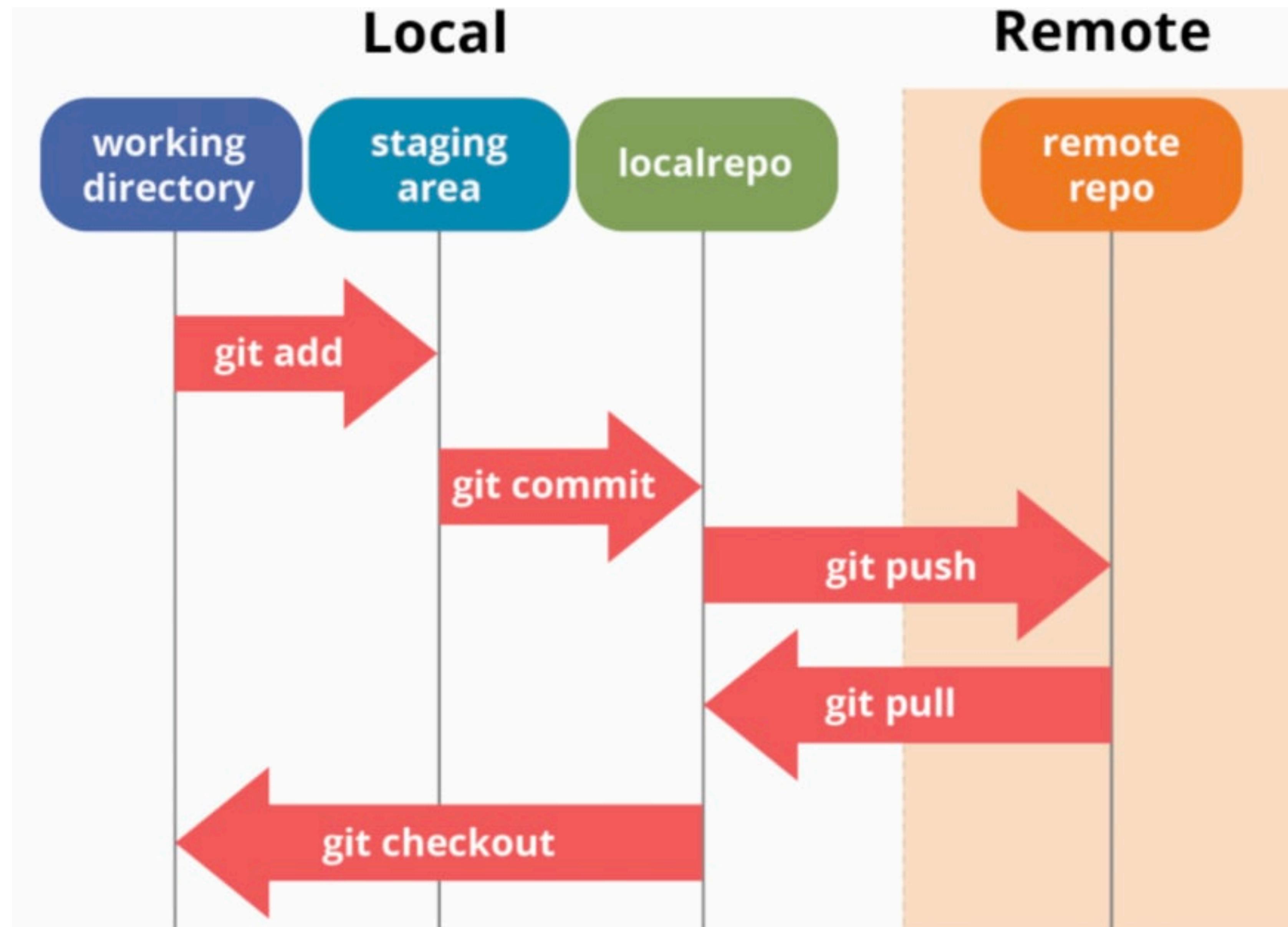
Distributed Version Control System



Azure Repos - Git Key concepts

- Branch
- Branch policies
- Clone
- Commit
- Fork
- Git workflow
- Git pull
- Git push
- Branch merge
- Pull request
- Ignore files
- Git config
- Git branching strategy
- Branch permissions
- Merge types
- Lock branch
- Merge conflicts
- Cherry-pick

Git commit Flow



Git config

- `git config --global user.name "FIRST_NAME LAST_NAME"`
- `git config --global user.email "MY_NAME@example.com"`
- `git remote add origin <url>`

`git remote remove origin`

Exercises for Section 2 part 1

- Create a repo in azure Repos and push code
- Create a new develop branch from master

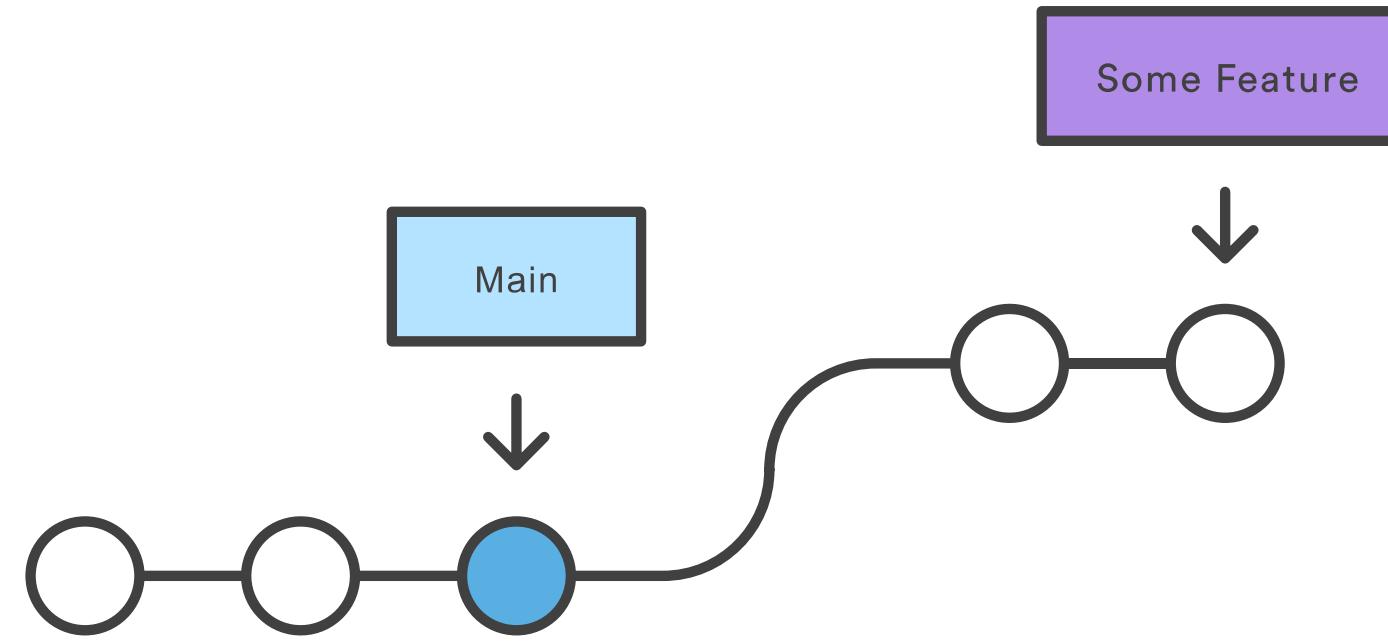
Section 2 part 2

Merge Types in Azure DevOps

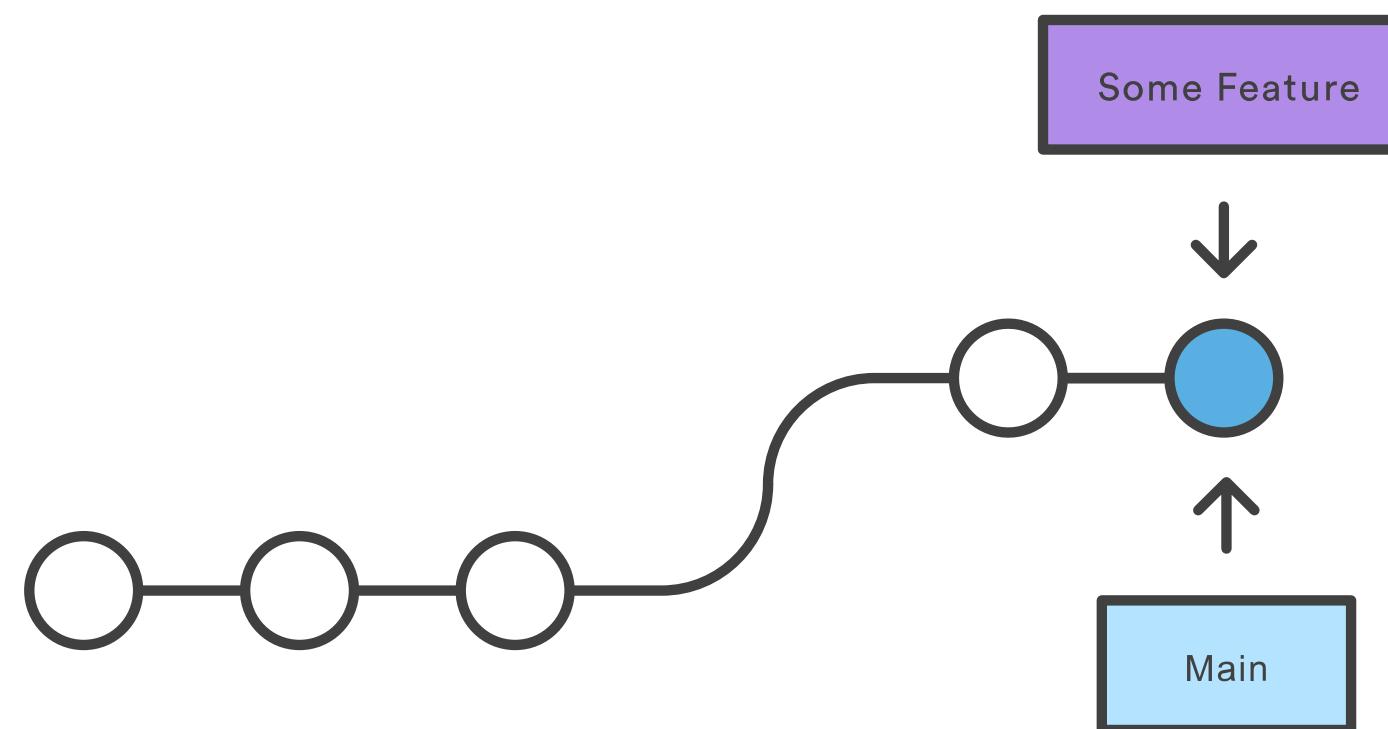
- Merge (no fast forward)
- Squash commit
- Rebase and fast forward
- Semi-linear merge

Fast Forward Merge

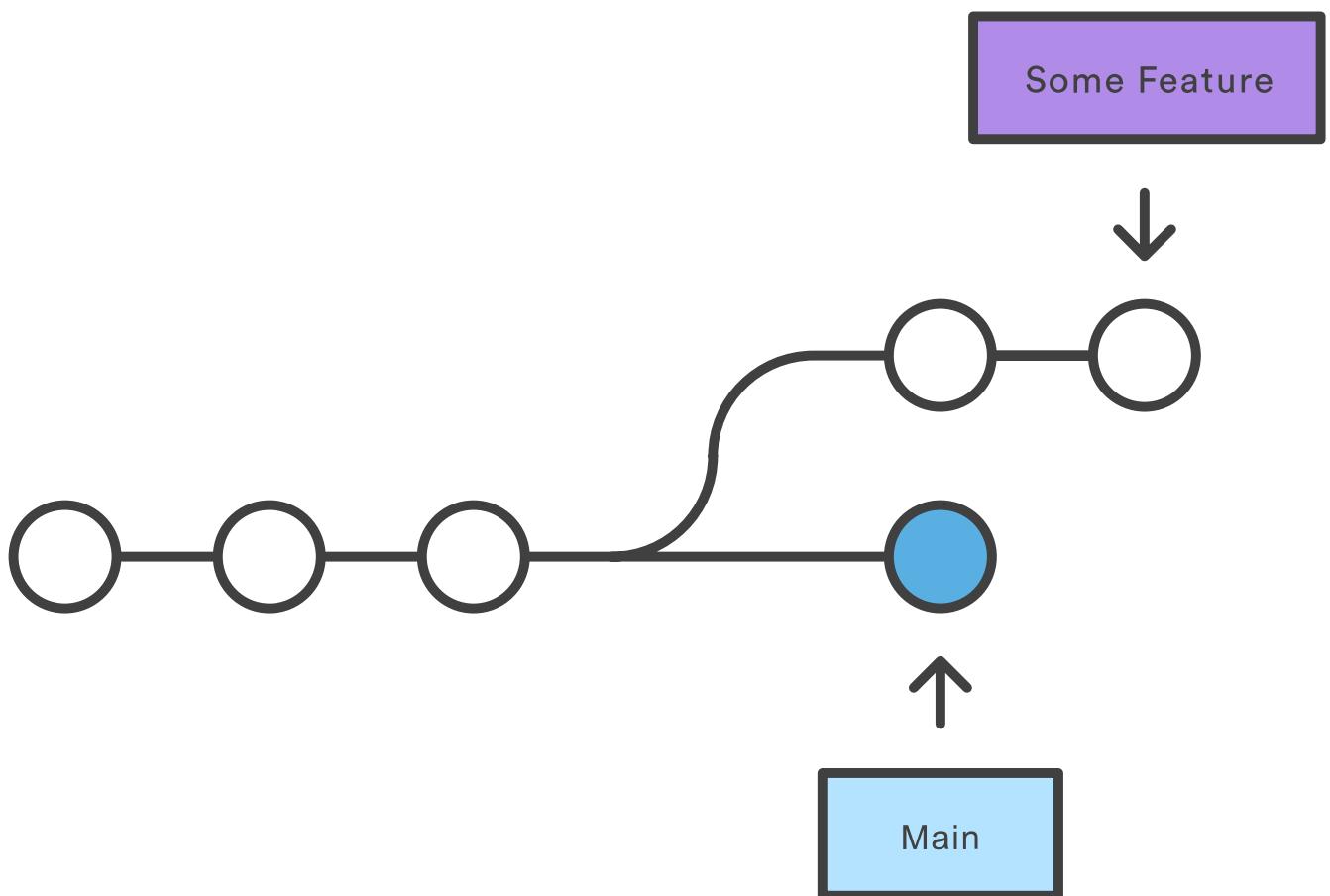
Before Merging



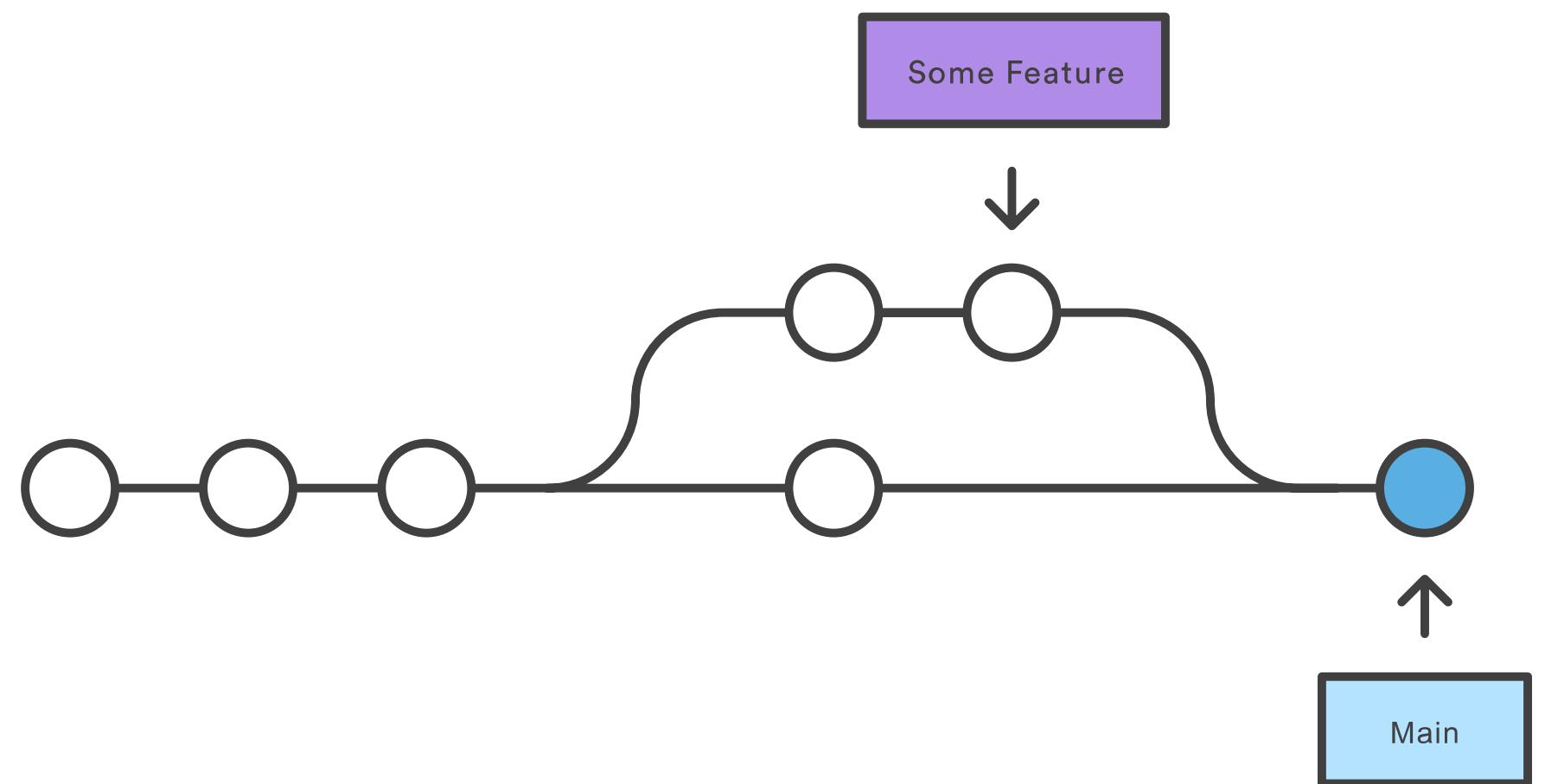
After a Fast-Forward Merge



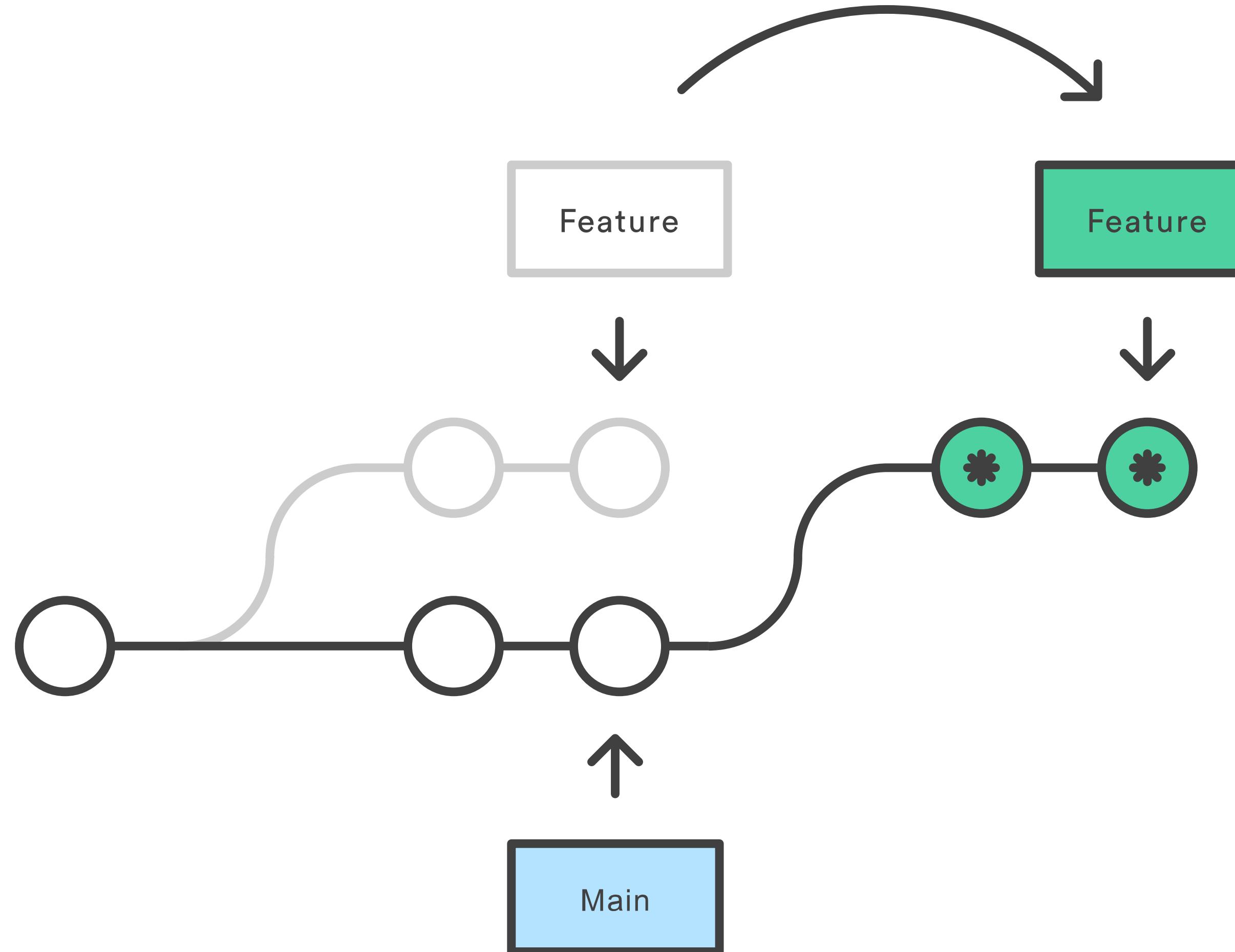
Before Merging



After a 3-way Merge



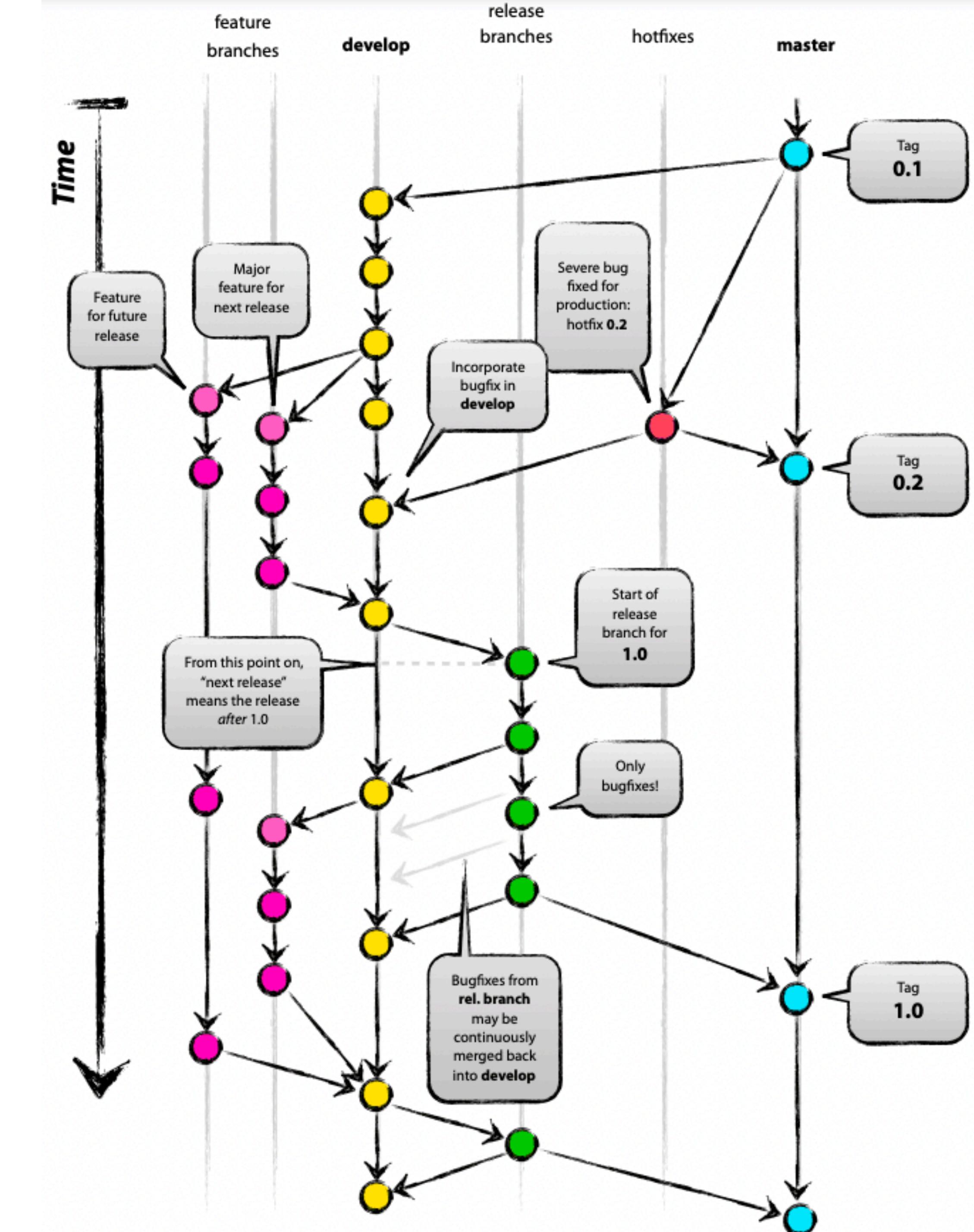
Git rebase



* Brand New Commits

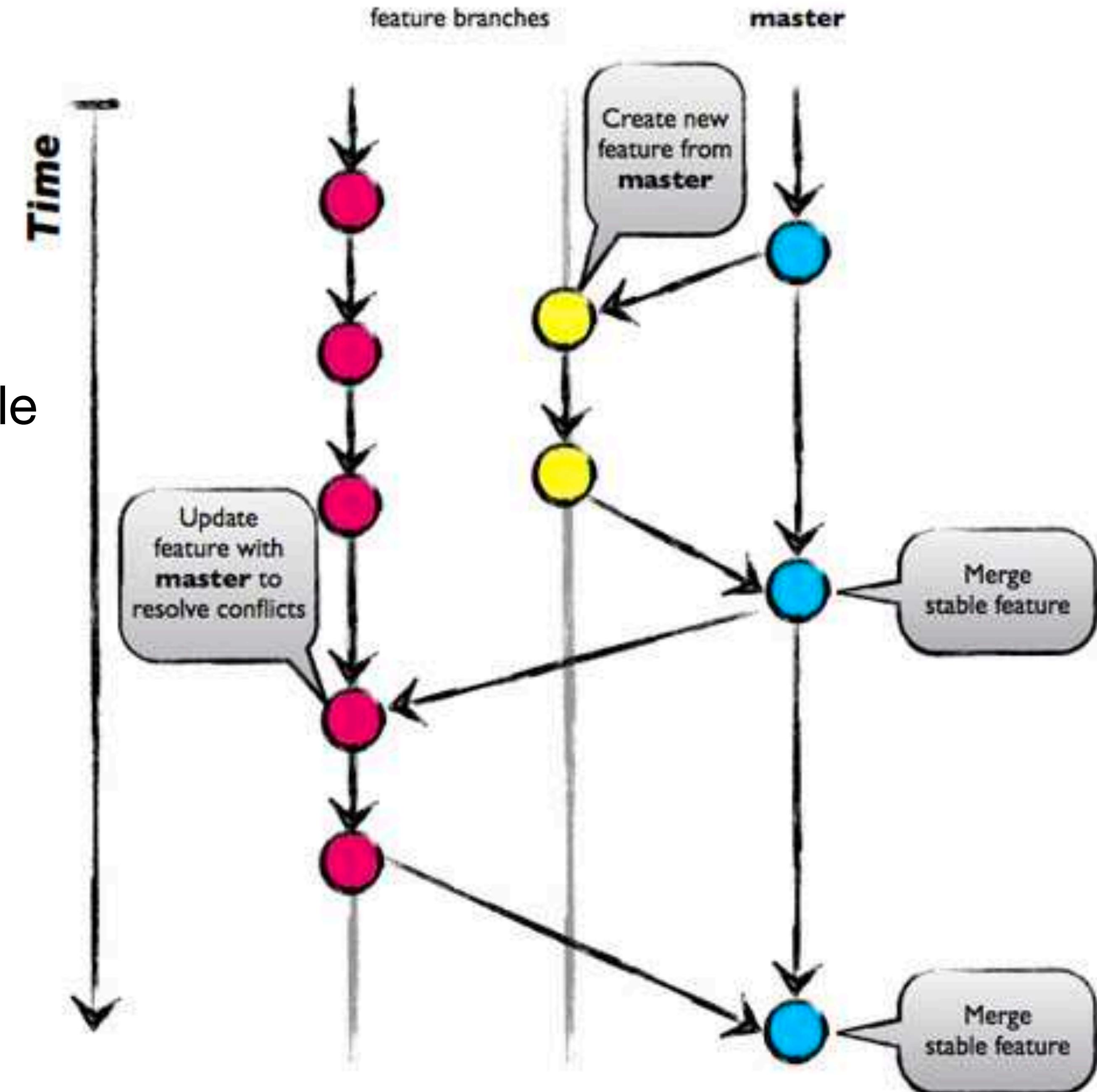
Git Flow

- Master, develop - infinite lifetime
- Feature - created on top of develop
- Release - prod release , develop merged back to develop and master
- Hot-fix - quick prod release , bug
- Remaining branches are short lived branches



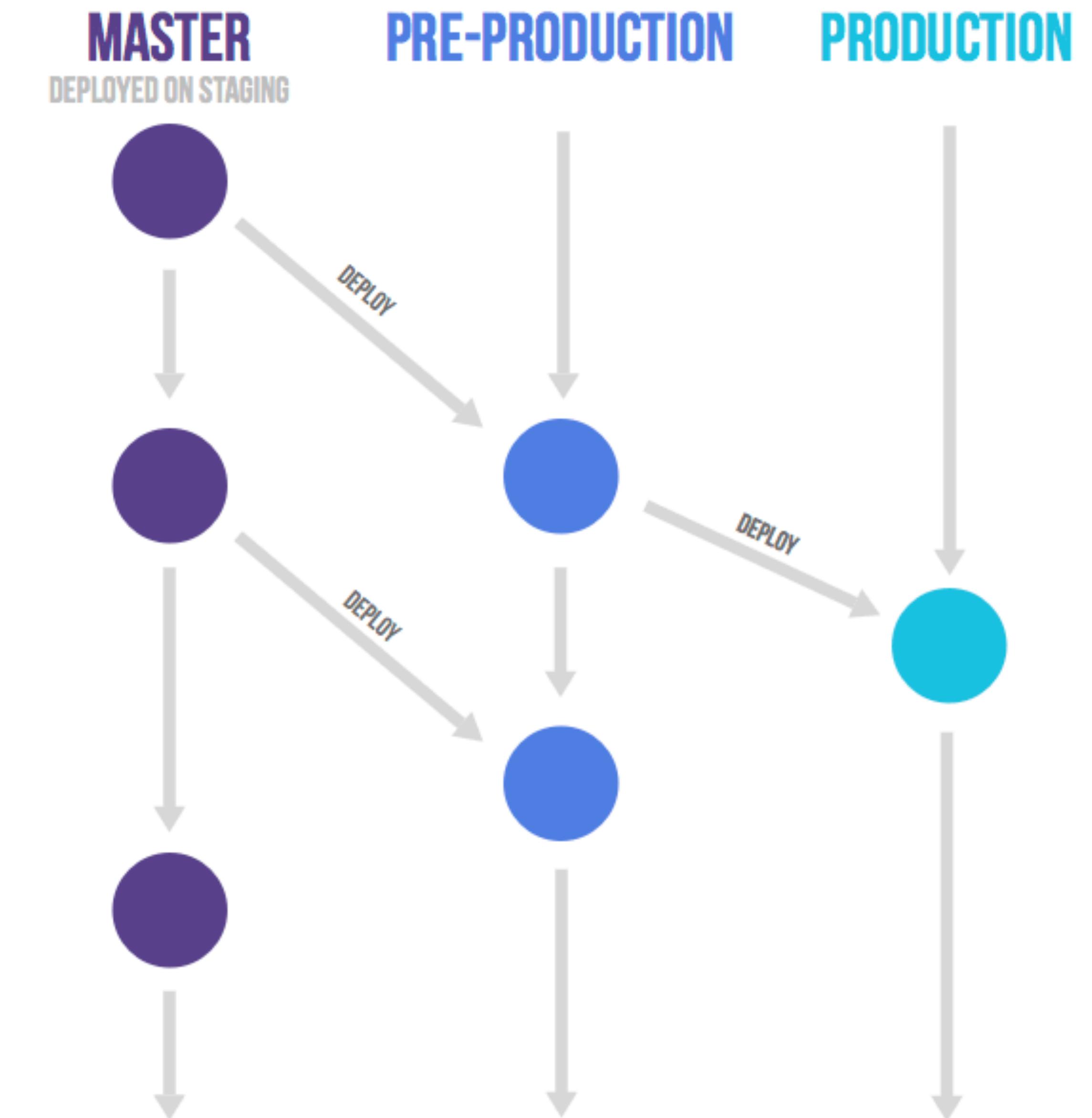
GitHub Flow

- Feature branches from master
- master code in a constant deployable state
- support continuous integration and continuous delivery processes.
- Focus on agile method



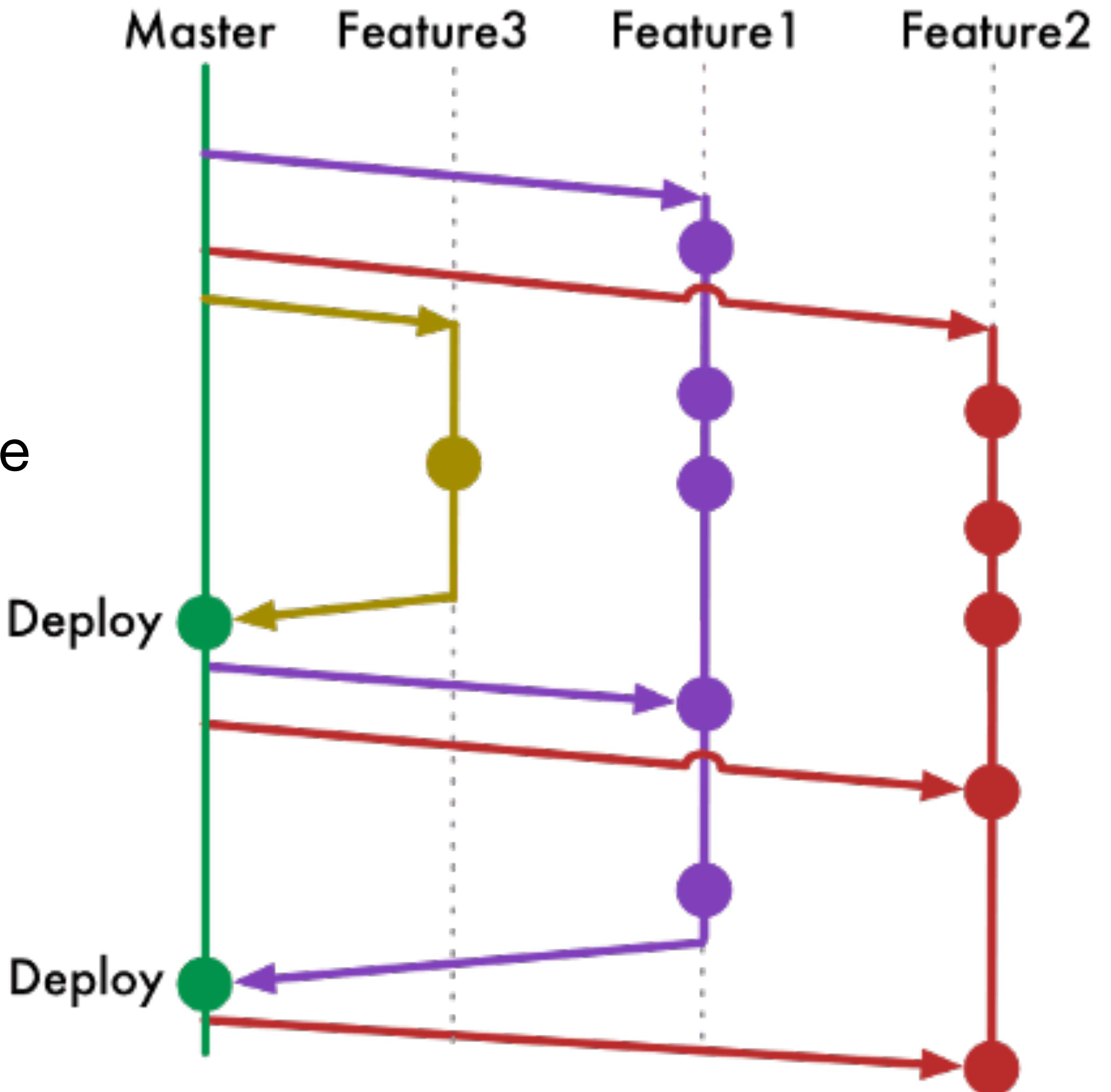
GitLab Flow

- Feature branches from master
- Multiple version of code in different environments
- Resolve the issue before going to production



Trunk-based Development

- Requires no branches
- Code changes into shared trunk
- This trunk can be released at anytime
- All developers work on the same branch.
- Used for feature flags



Best branching strategy for your team

Product type and its release method	Team size	Collaboration maturity	Applicable mainstream branch mode
All	Small team	High	Trunk-Based Development (TBD)
Products that support continuous deployment and release, such as SaaS products	Middle	Moderate	GitHub-Flow and TBD
Products with a definite release window and a periodic version release cadence, such as iOS apps	Middle	Moderate	Git-Flow and GitLab-Flow with release branch
Products that are demanding for product quality and support continuous deployment and release, such as basic platform products	Middle	Moderate	GitLab-Flow
Products that are demanding for product quality and have a long maintenance cycle for released versions, such as 2B basic platform products	Large	Moderate	Git-Flow

Project settings -> repositories walkthrough

- Permissions for different groups

Repos -> branches -> 3 dot on branch -> branch policies

- This will give build validation , status checks , approvals

*These concepts can be explained after pipeline creation, dependency on build pipeline

Cross service integration

- End-to-end traceability
- Traceability and linking
- Branch from a requirement
- Create a pull request from a requirement
- Add and run tests from requirements
- Deployment and Development controls
- Bug traceability

Exercises for Section 2 part 2

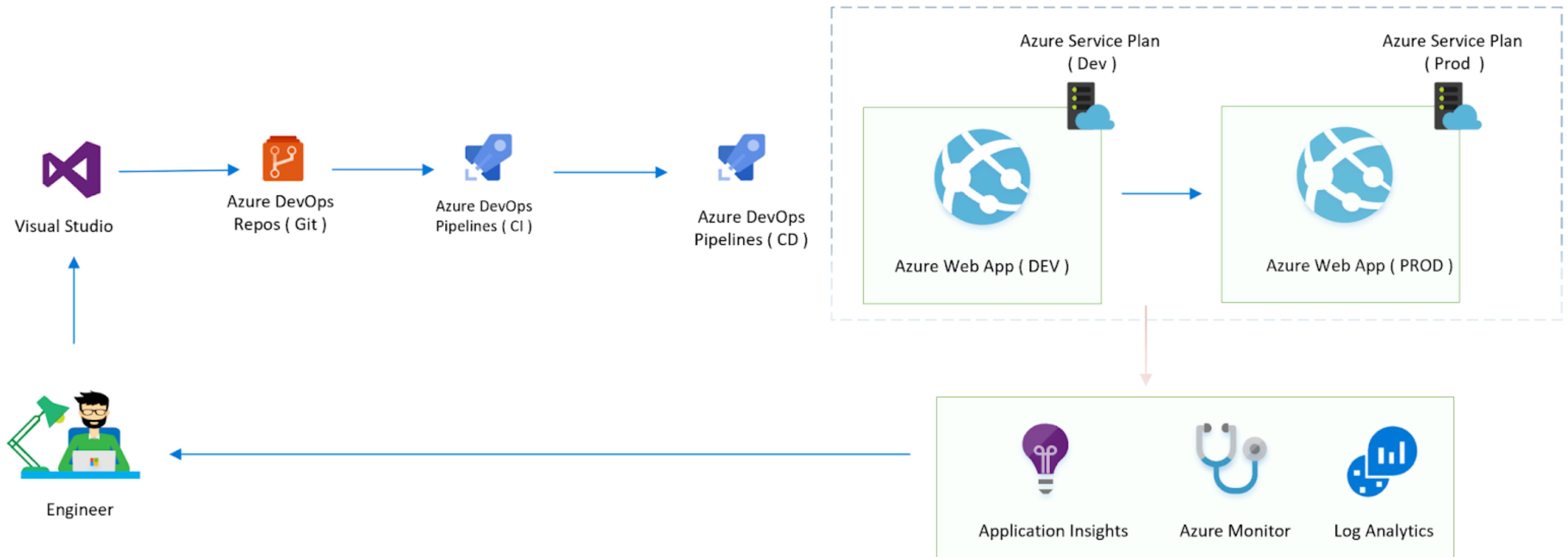
- Merge a branch using squash merge with PR
- Merge a branch using rebase with some commits ahead
- Create branch policies for main branches
- Build validation , status checks
- Add reviewers in PR , link work items at all stages commit, pr , merge
- Lock a branch
- Create and resolve conflicts in branches (learn)

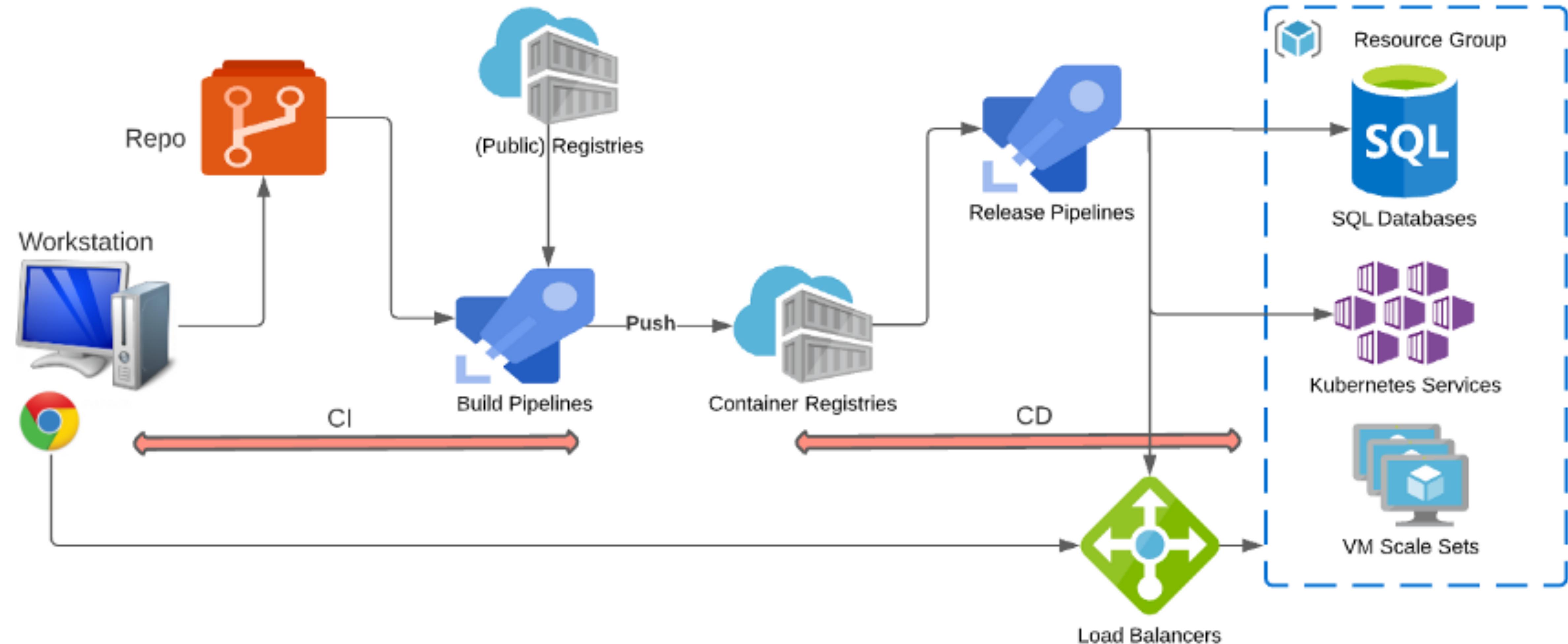
Section 3



Azure Pipelines

- Continuous Integration
- Continuous Delivery
- Continuous Testing
- Azure DevOps Services
- Azure DevOps Server







Free grant of parallel jobs

- <https://aka.ms/azpipelines-parallelism-request>

/ Settings / Parallel jobs

The screenshot shows the 'Parallel jobs' settings page in the Azure Pipelines interface. It displays grants for both private and public projects.

Private projects:

Provider	Grant Type	Count	Action
Microsoft-hosted	Free tier	1 parallel job up to 1800 mins/mo	Purchase parallel jobs
Self-hosted	Free parallel jobs	2 Parallel jobs	
Visual Studio Enterprise subscribers	Free parallel jobs	1	
Monthly purchases	Free parallel jobs	0 Change	

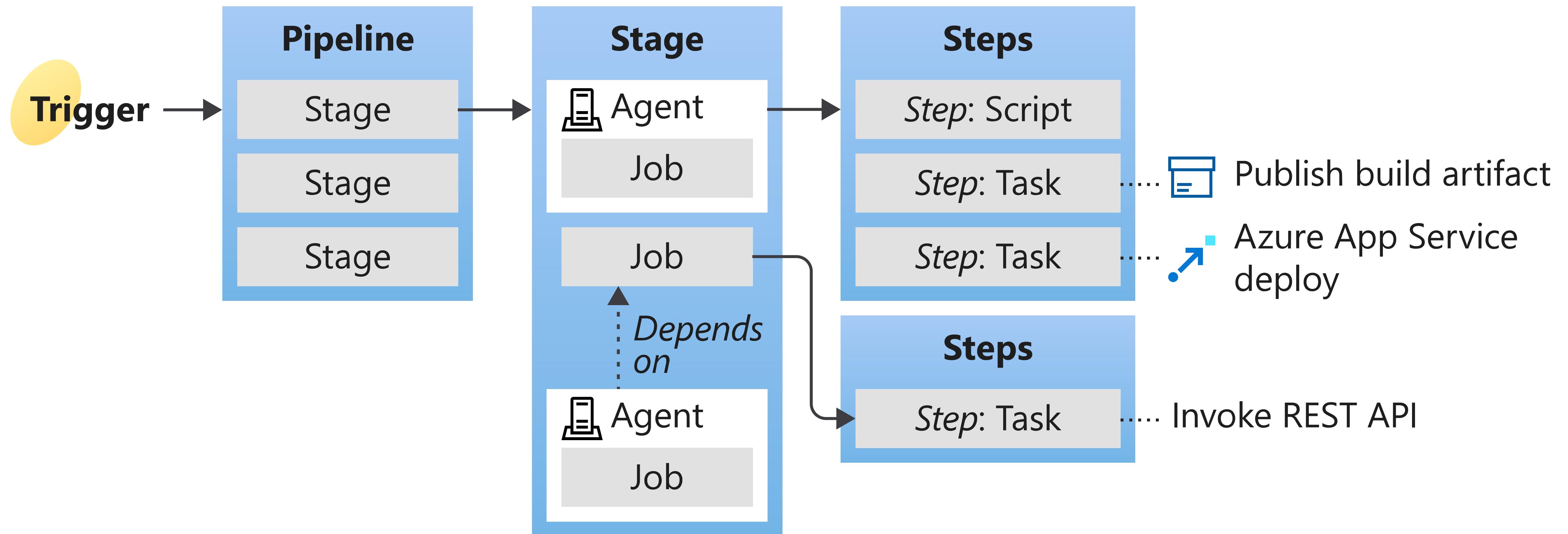
Public projects:

Provider	Grant Type	Count	Action
Microsoft-hosted	Free parallel jobs	0 Parallel jobs	
Self-hosted	Unlimited	Unlimited Parallel jobs	

Key concepts

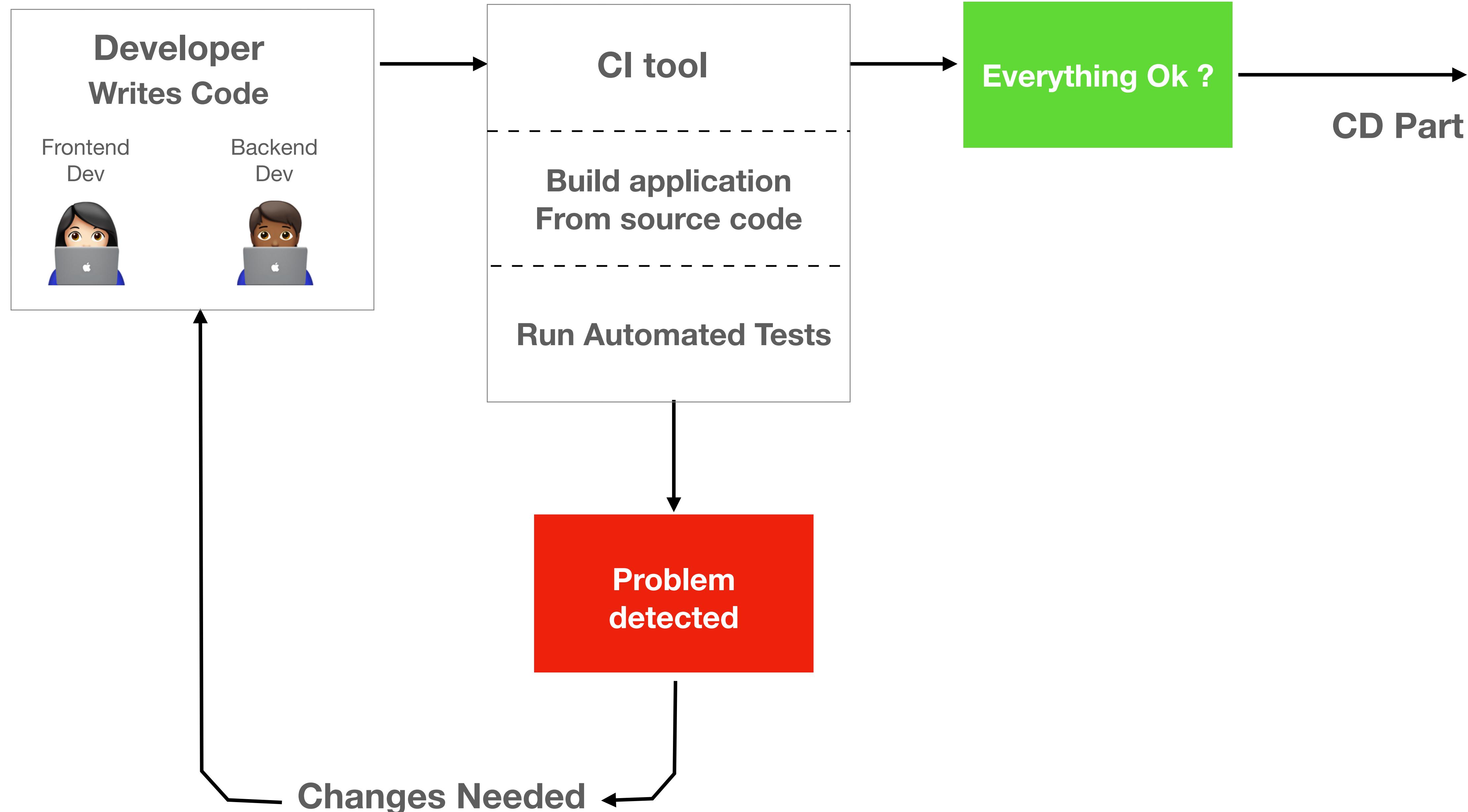
- Agent
- Approvals , checks , gates
- Artifact
- Continuous delivery
- Continuous integration
- Deployment
- Deployment group
- Environment
- Agent
- Job
- Pipeline
- Release
- Run
- Script
- Stage
- Step
- Task
- Triggers
- Library , variables
- Taskgroup
- Templates
- Caching
- Tests
- Flaky tests

Key concepts overview



Feature Availability

- <https://learn.microsoft.com/en-us/azure/devops/pipelines/get-started/pipelines-get-started?view=azure-devops#feature-availability>



Self Hosted agent setup

- PAT
- VM creation
- Git, .net
- Post vm steps
- Install visual studio

YAML concepts

- — — — ...
- Key-value pairs
- List
- Dictionary
- Comments

Change project private to public

Azure DevOps AshokDevOps5 / Settings / Policies

Policies

Application connection policies

- Off Third-party application access via OAuth [🔗](#)
- On SSH authentication [🔗](#)

Security policies

- Off Log Audit Events [🔗](#)
- On Allow public projects [🔗](#)
- On Additional protections when using public package registries [🔗](#)
- Off Enable IP Conditional Access policy validation on non-interactive flows [🔗](#)

User policies

Organization Settings
AshokDevOps5

Search Settings

General

- Overview
- Projects
- Users
- Billing
- Global notifications
- Usage
- Extensions
- Azure Active Directory

Security

- Policies
- Permissions

AshokDevOps5 / SampleProject / Settings / Overview

Project Settings
SampleProject

General

- Overview
- Teams
- Permissions
- Notifications
- Service hooks
- Dashboards

Boards

- Project configuration
- Team configuration
- GitHub connections

Pipelines

Project details

Name: SampleProject

Description: sample project to explain concepts

Process: mynewProcess

Visibility: Public

This determines who can view this project. [Learn more about project visibility.](#)

Save

Build Pipeline Creation

Continuous integration

Sample pipeline creation in Azure Devops

- Pre-defined variables in pipelines
- [https://learn.microsoft.com/en-us/azure/devops/pipelines/build/variables?
view=azure-devops&tabs=yaml](https://learn.microsoft.com/en-us/azure/devops/pipelines/build/variables?view=azure-devops&tabs=yaml)

Sonar cloud

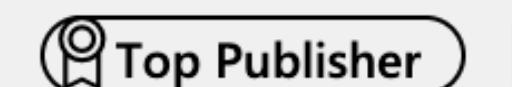
- Code definition
- Quality Gates (pass /failed)

Visual Studio | Marketplace

Azure DevOps > Azure Pipelines > SonarCloud



SonarCloud

SonarSource  sonarsource.com  |  62,614 instances

Detect bugs, vulnerabilities and code smells across project branches and pull requests.

[Get it free](#)

Mendbolt

- FREE extension, which scans all your projects and detects open source components, their license and known vulnerabilities
- Mend tab

The screenshot shows a screenshot of the Mend Bolt (formerly WhiteSource) extension page in the Visual Studio Marketplace. At the top, there's a navigation bar with the Visual Studio logo and links for 'Visual Studio' and 'Marketplace'. Below that is a breadcrumb trail: 'Azure DevOps > Azure Pipelines > Mend Bolt (formerly WhiteSource)'. The main content area features a large circular logo with a stylized 'W' or 'M' shape in white on a blue gradient background. To the right of the logo, the text 'Mend Bolt (formerly WhiteSource)' is displayed in bold. Below it, 'WhiteSource' is mentioned along with install statistics ('18,107 installs') and a rating ('4.3 stars'). A green button at the bottom says 'Get it free'.

Visual Studio | Marketplace

Azure DevOps > Azure Pipelines > Mend Bolt (formerly WhiteSource)

Mend Bolt (formerly WhiteSource)

WhiteSource | 18,107 installs | ★★★★☆ (43) | Free

Get real-time security alerts and compliance issues on your open source dependencies in the Azure DevOps Services environment.

Get it free

Create work item on failure

Triggers

- Commit
- PR
- Scheduled triggers - cron
- Branch considerations
- filters - include / exclude

Template usage

- template jobs
- Template with parameters
- Variables
- Secret variables

Task control options

```
- task: string # reference to a task and version, e.g. "VSBUILD@1"
  condition: expression # see below
  continueOnError: boolean # 'true' if future steps should run even if this step fails; defaults to 'false'
  enabled: boolean      # whether or not to run this step; defaults to 'true'
  retryCountOnTaskFailure: number # Max number of retries; default is zero
  timeoutInMinutes: string # how long to wait before timing out the task
  target: string        # 'host' or the name of a container resource to target
```

Jobs

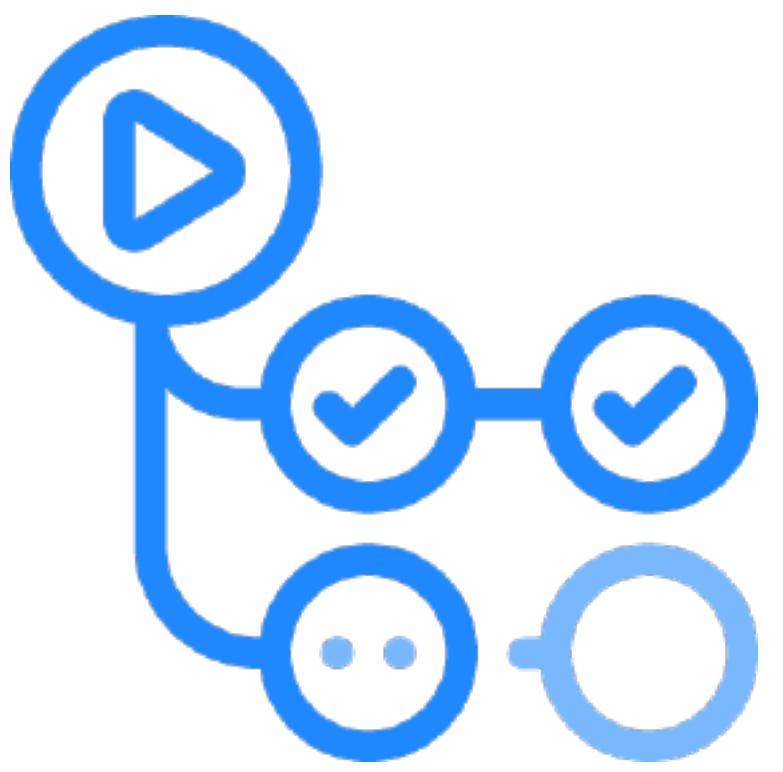
- Dependent jobs
- Conditions job
- Multi-job configs

Deployment jobs

- special type - Deployment job
- Deployment history
- Apply deployment strategy
 - runOnce
 - rolling
 - canary
- <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/deployment-jobs?view=azure-devops#schema>

Deployment Job

- lifecycle hooks - preDeploy, deploy,
- routeTraffic,
- postRouteTraffic
- on: failure on: success



Github Actions

- Yaml
- Workflows
- Jobs
- Github Runners
- Self Hosted Runners
- WorkFlow Dispatch

Exercises for Section 3

- Create CI pipeline for favourite programming language - Microsoft Hosted
- Use self hosted for same CI pipeline
- Create multiple agents from single VM
- For CI , use YAML and Classic pipelines
- Integrate SonarQube , MendBolt
- Deploy app to Azure WebApp
- Include secrets in pipelines using key vault
- Implement CI/CD pipeline for app using Github Actions

Section 4



Azure Artifacts

- Feeds - store, manage, and group your packages
- FeedViews - have package versions that have been tested and validated but hold back on packages
- UpStream Sources - store packages from various sources in a single feed
- Azure Artifacts pricing

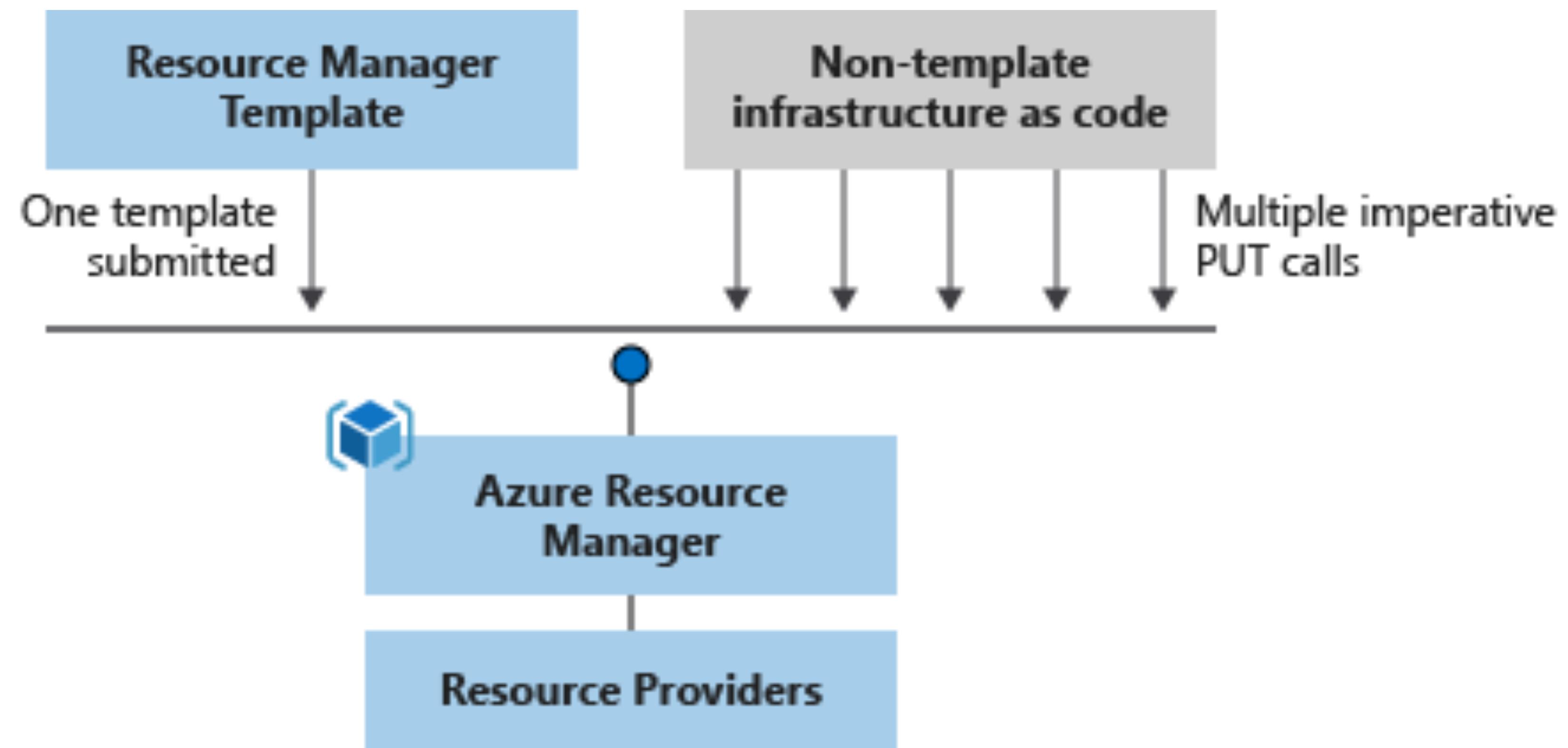


Azure Testplans

- Rich and powerful tools everyone in the team can use to drive quality and collaboration throughout the development process.
- Easy-to-use, browser-based test management solution provides all the capabilities required for planned manual testing
- Test plans Pricing

ARM Template structure

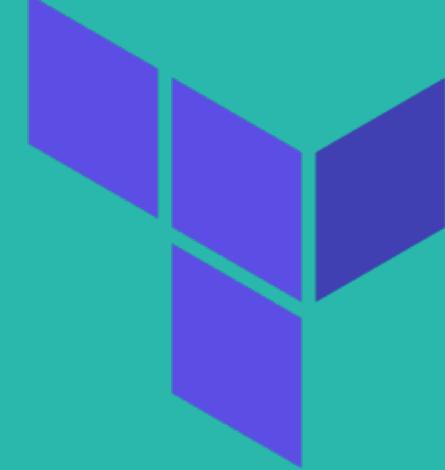
- \$schema
- contentVersion
- parameters
- variables
- resources
- outputs



Infra as Code



Bicep



HashiCorp

Terraform



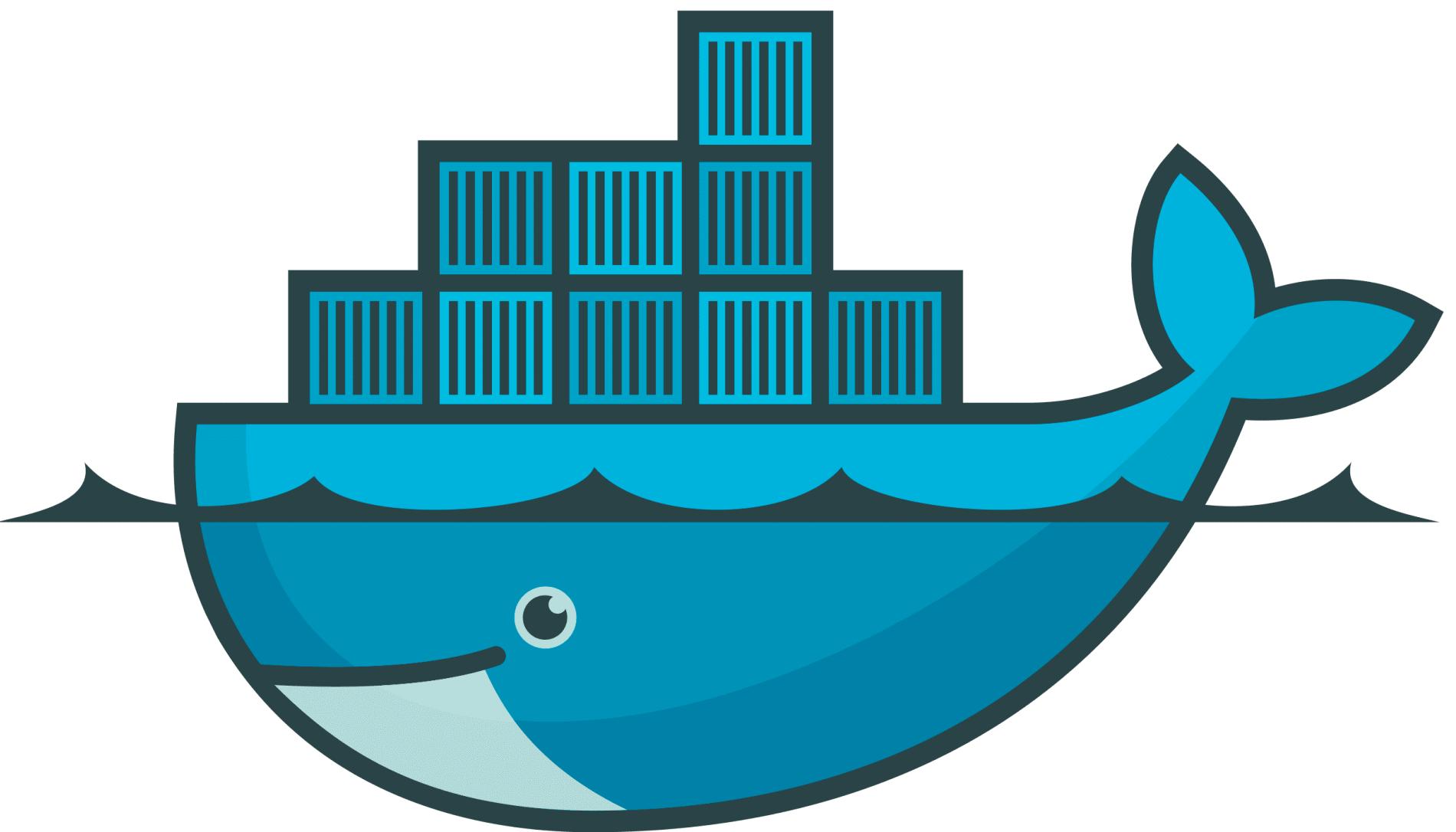
Deployment Examples

- App deployment to WebAPP
- ARM template Deployment

Exercises for Section 4

- Create ARM deployment stage with ADO pipeline
- Create resources using terraform with ADO pipeline
- Include security scanning for Resources using Code
- DependaBot, security Enablement in Github Repos

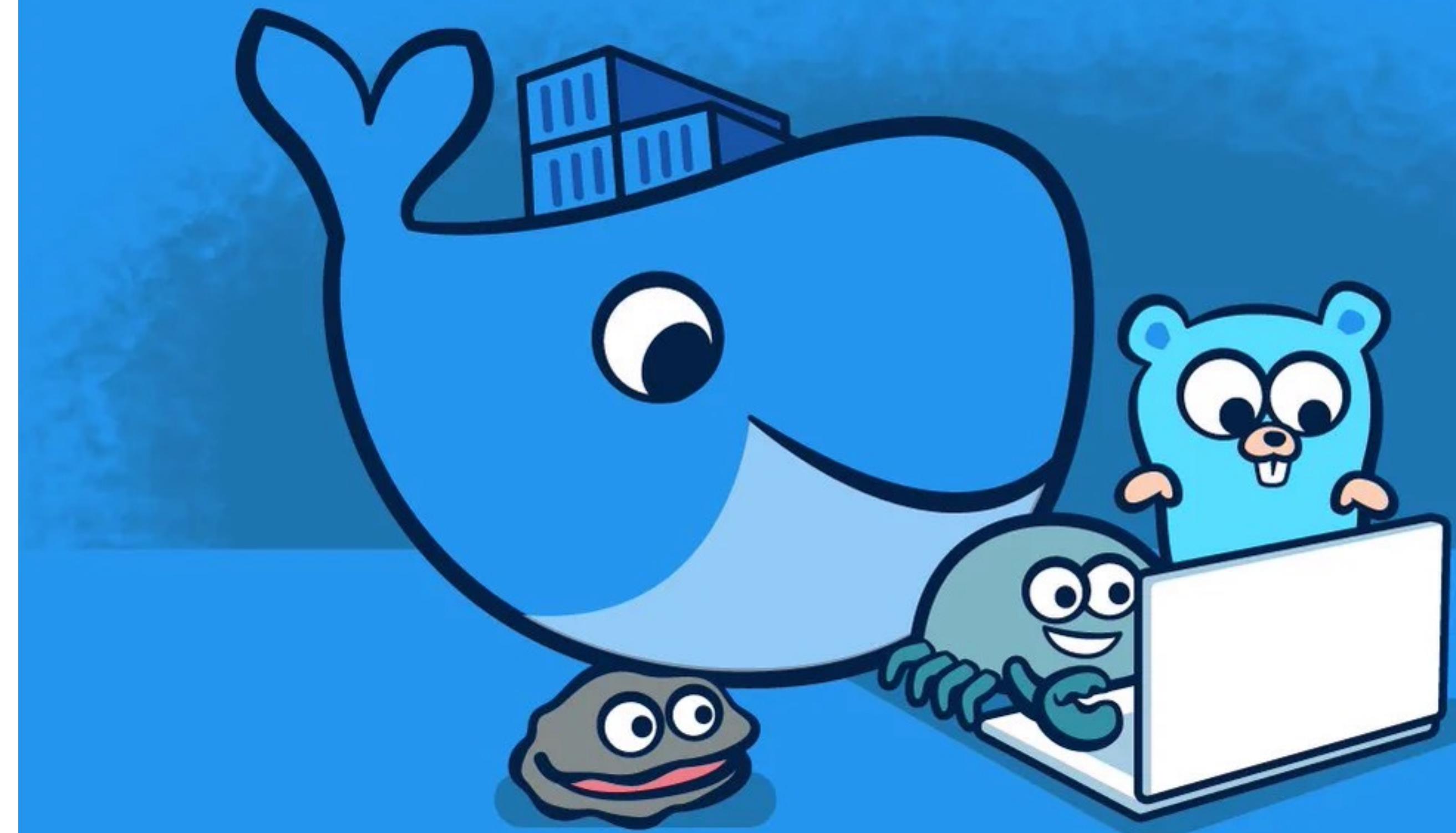
Section 5



docker

Docker setup

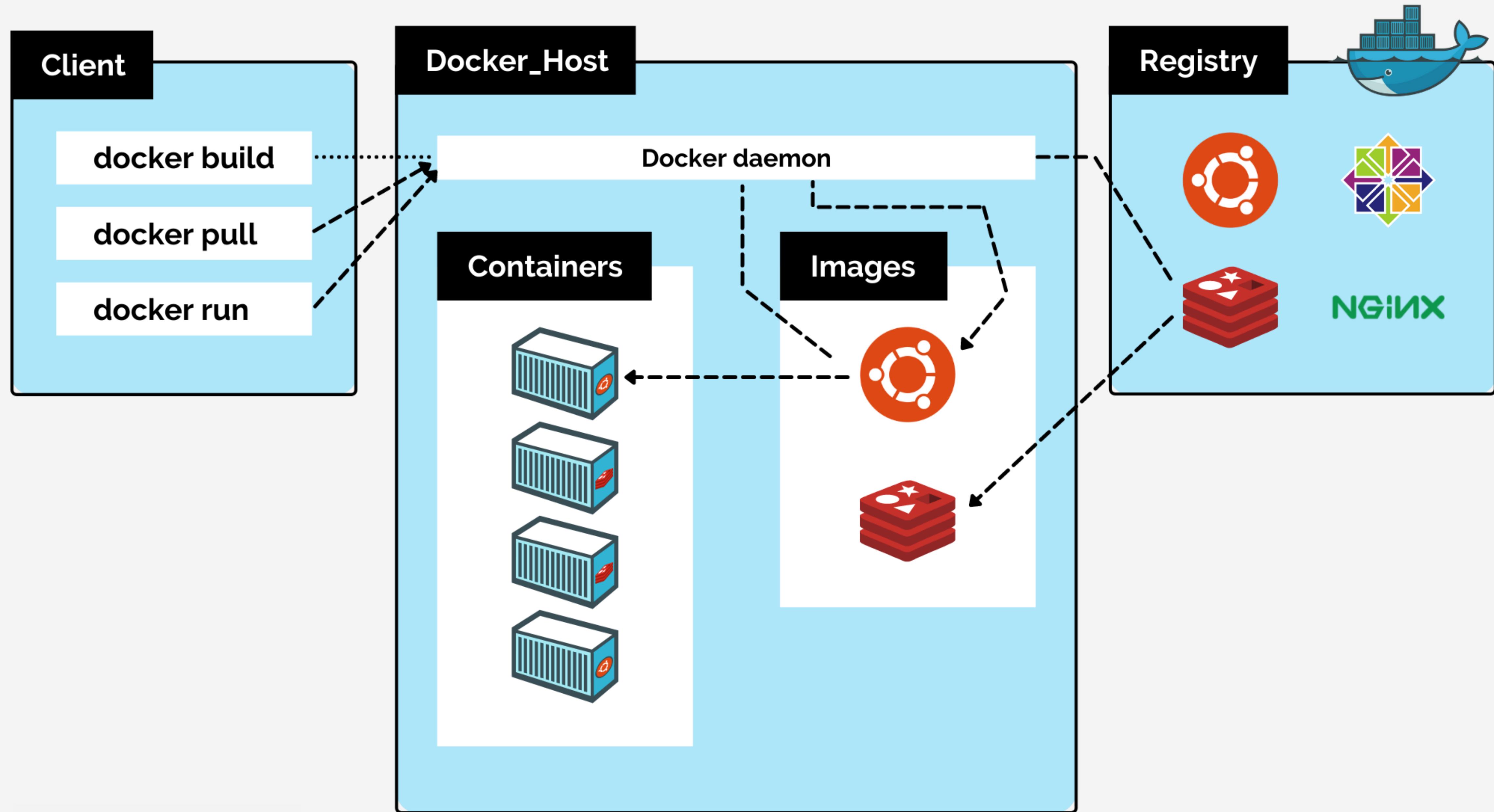
- Docker Desktop in Local machine
- Docker services in Azure Cloud
- ACR, ACI
- PlaywithDocker



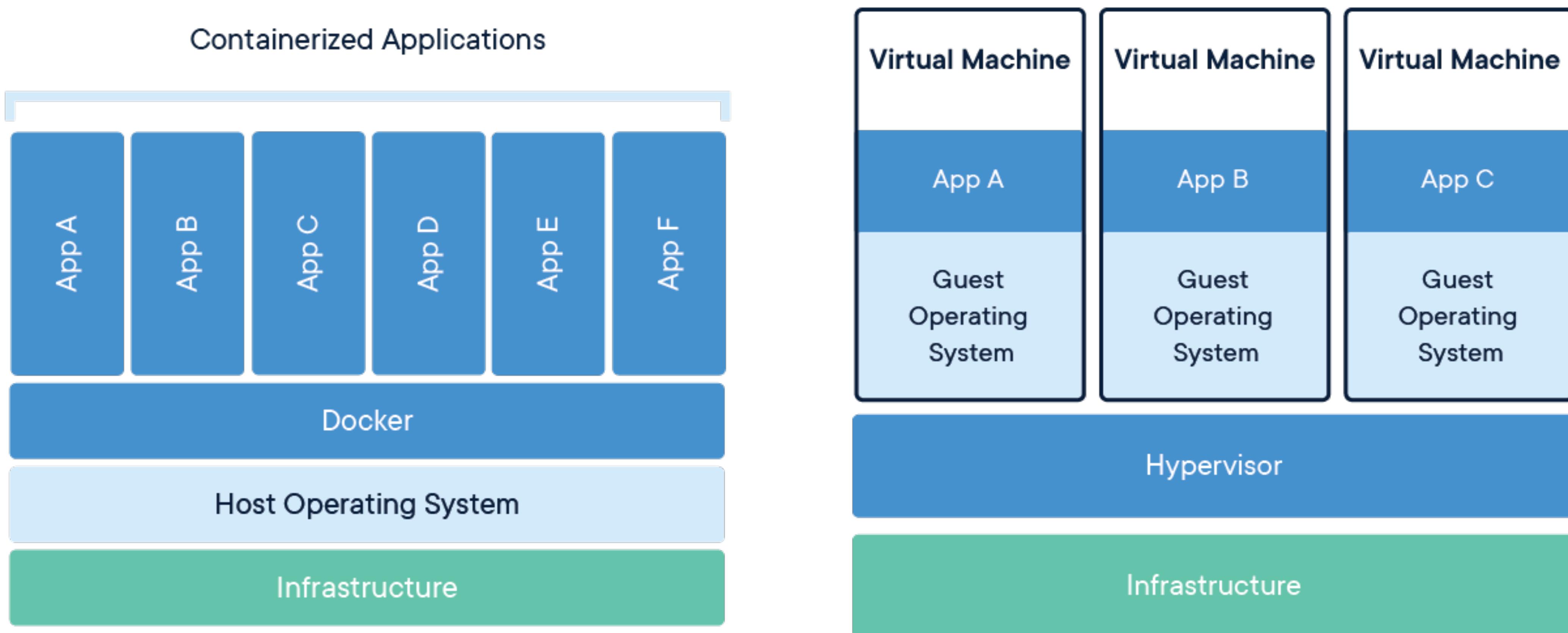
Docker Key Concepts

- DockerFile
- Image
- Container
- DockerRegistry
- Docker Filesystem
- Docker Commands
- Docker Compose
- Docker Networking
- Docker Swarm
- Kubernetes

Docker Architecture



Container vs VM architecture

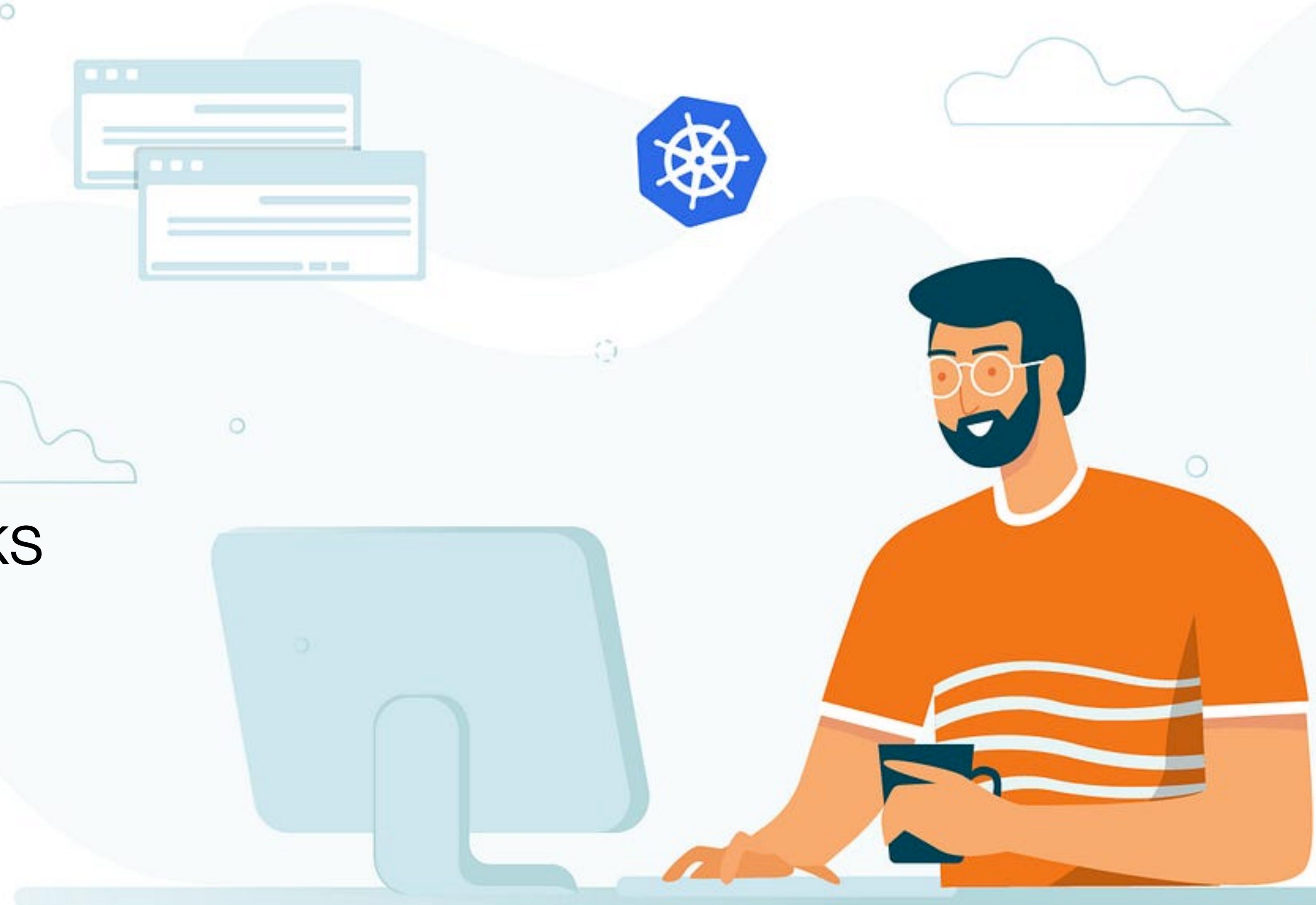




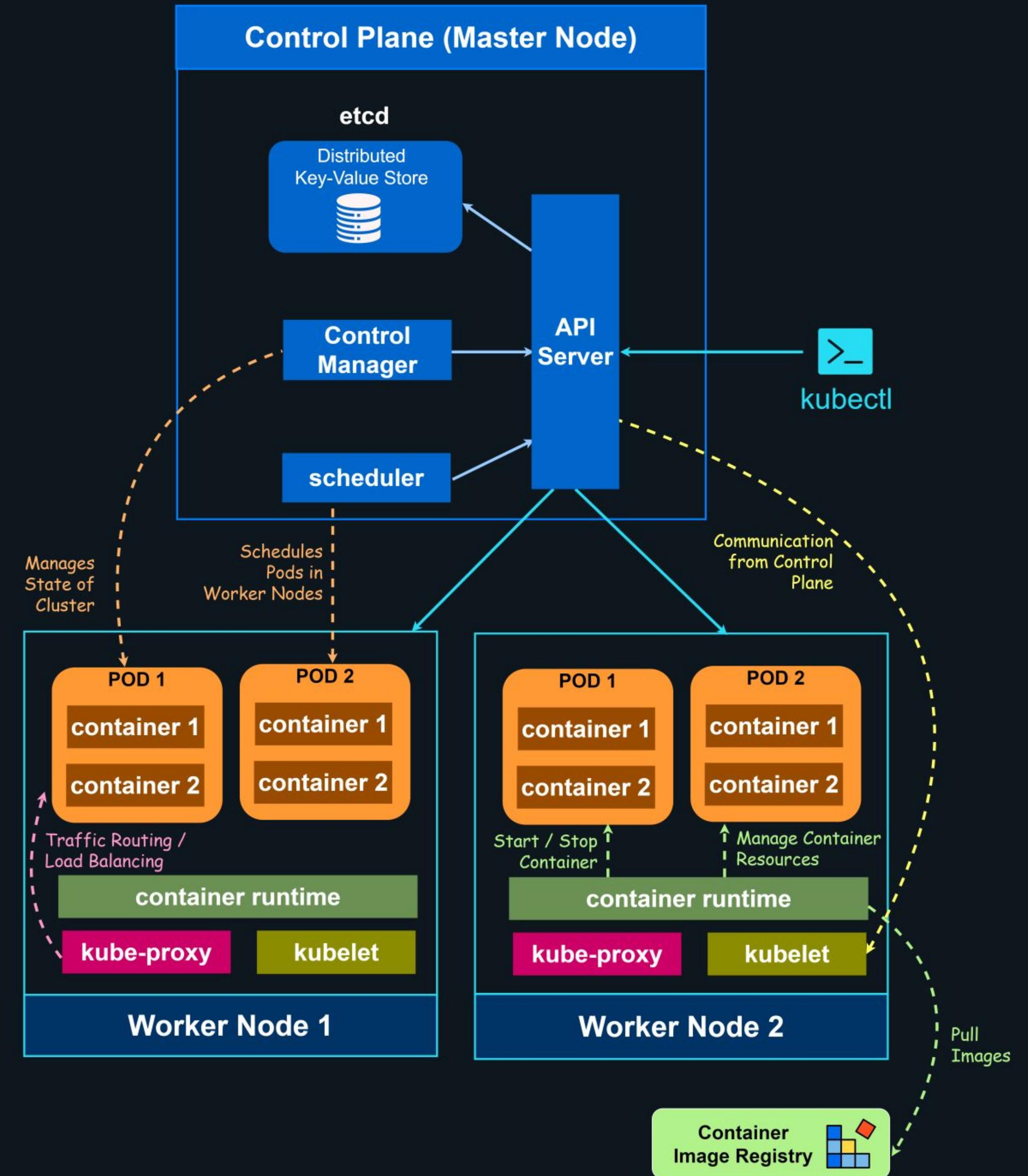
kubernetes

Kubernetes setup

- miniKube
- Docker Desktop
- Kubernetes extension
- Play with k8s
- Azure cloud service - AKS



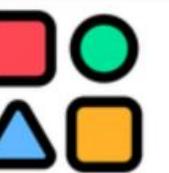
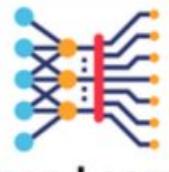
Kubernetes Architecture Knowledge Map

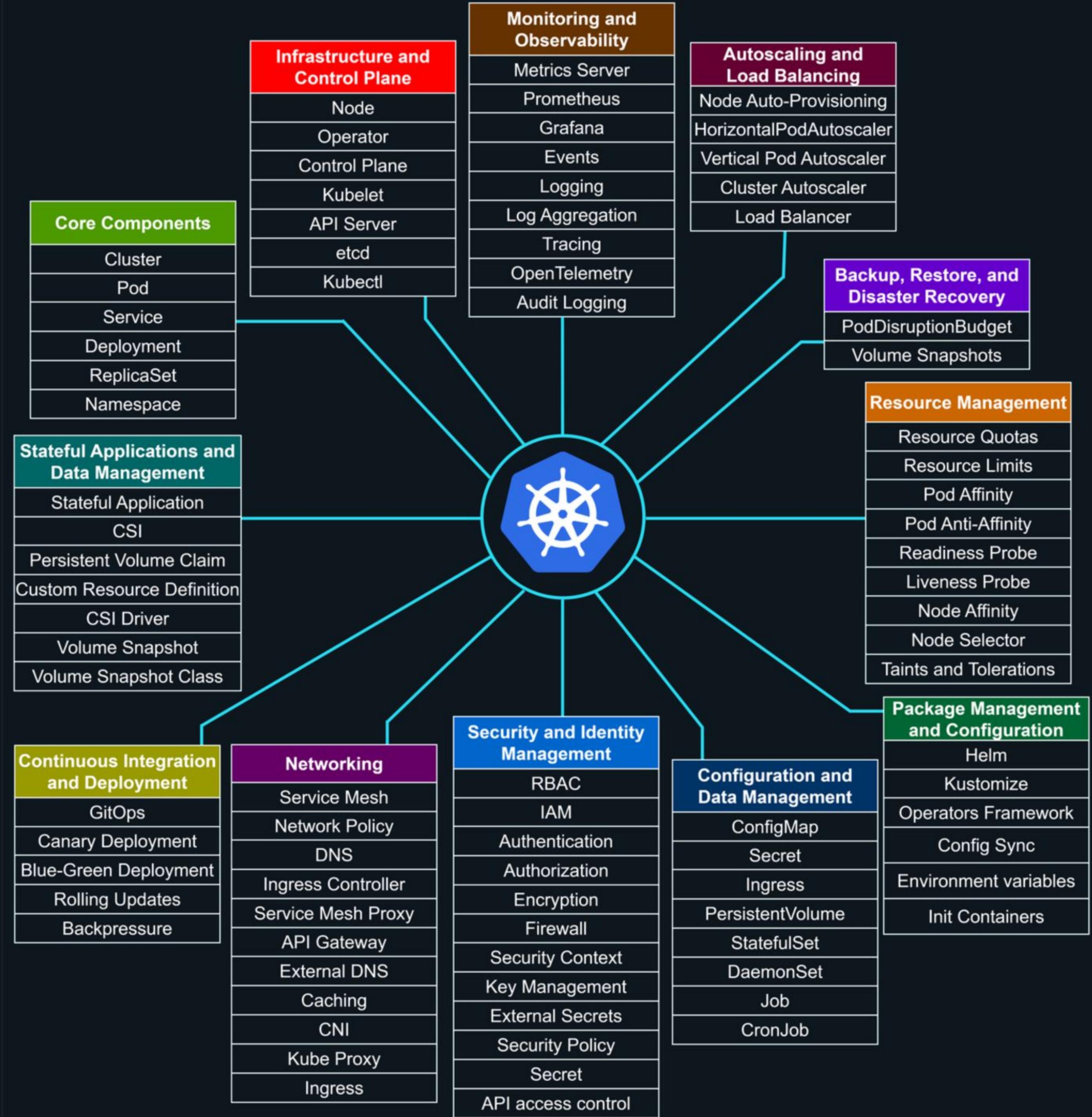


Kubernetes Key Concepts

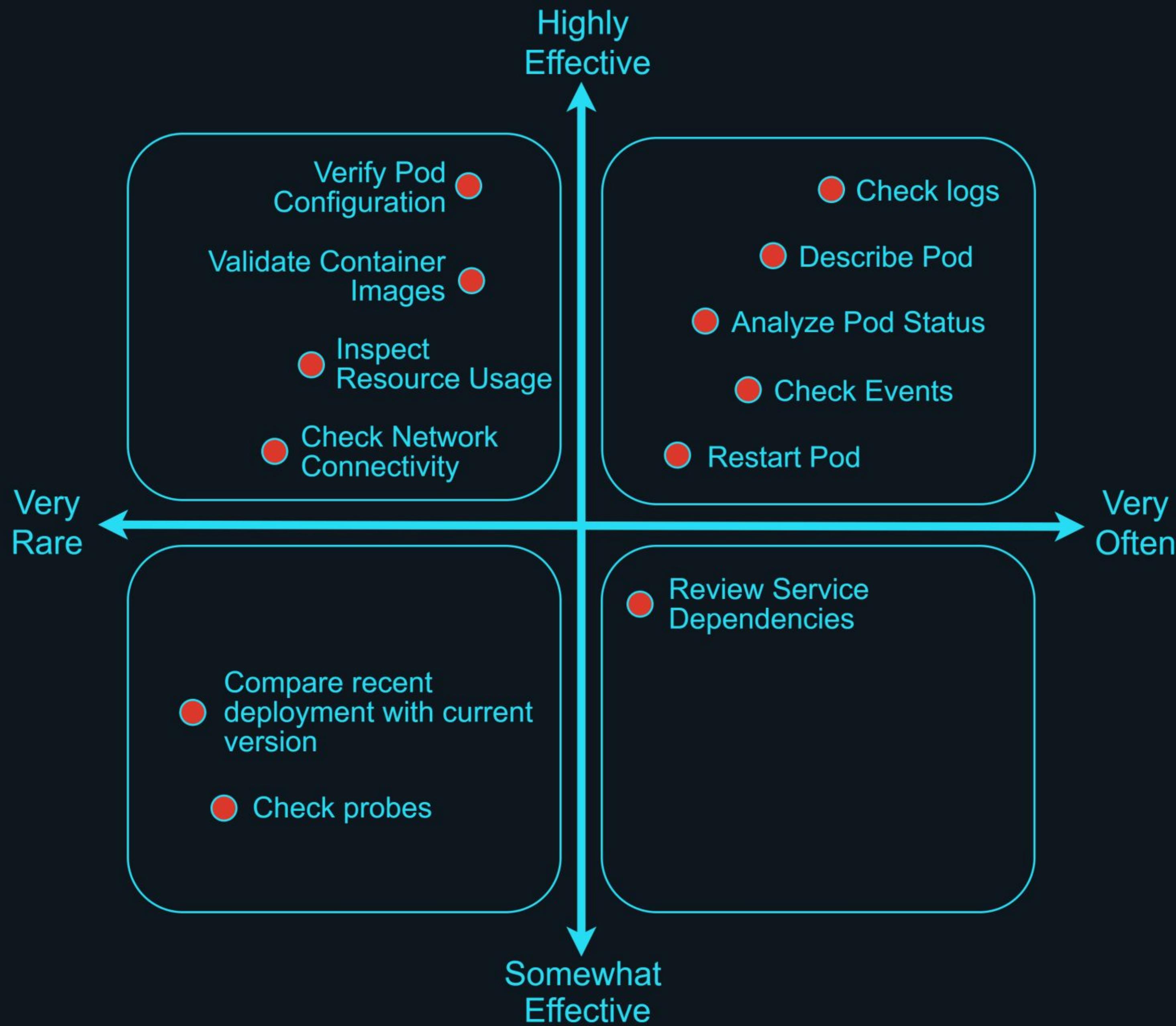
- Pods
- ReplicaSets
- Deployments
- Services
- Imperative way
- Declarative way
- Persistent storage
- Secrets
- Taints
- NodeAffinity
- Cordon
- Ingress

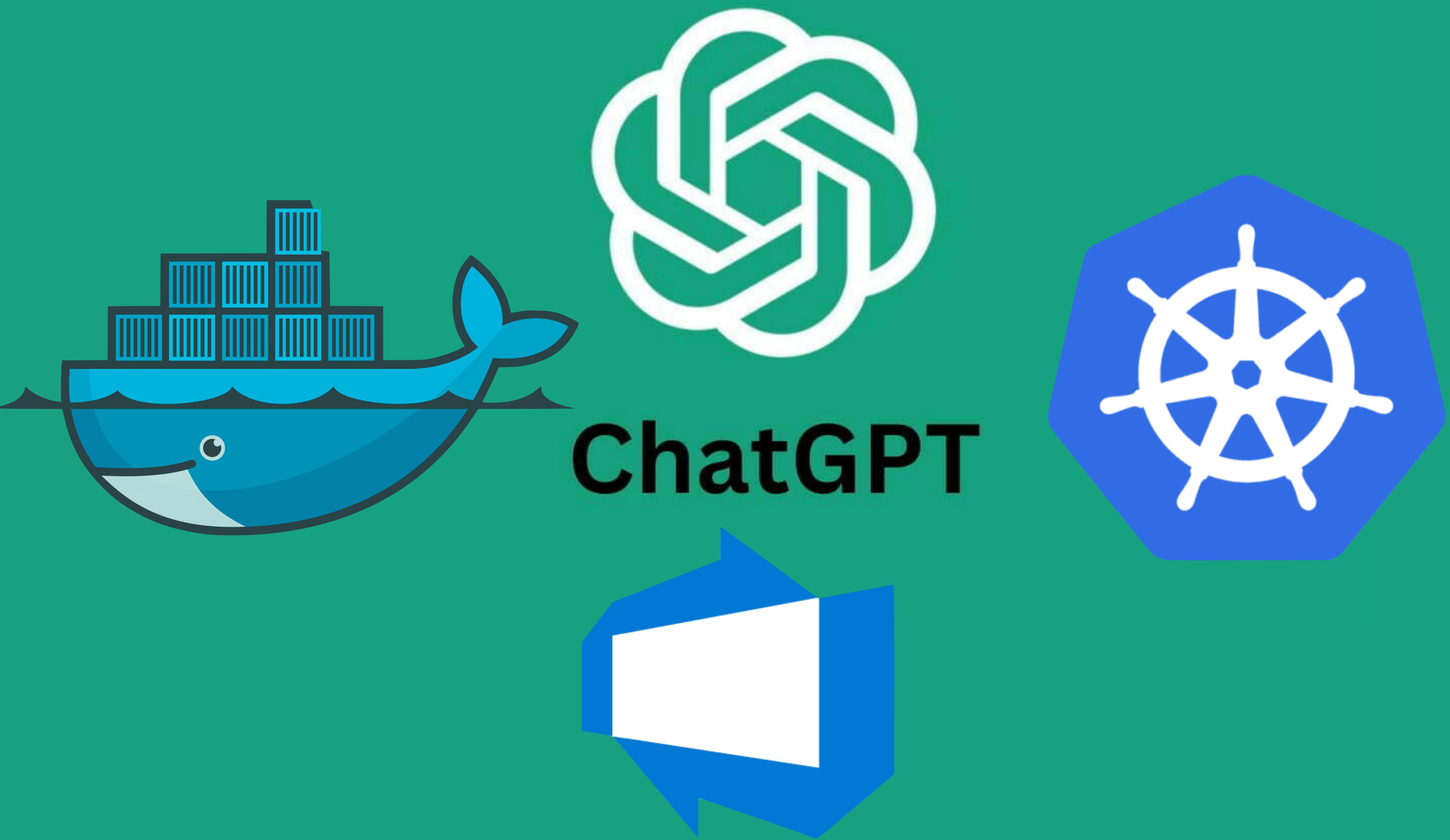
10-STEP KUBERNETES LEARNING PATH

Step	Coverage	Concepts
 Understand the Basics	<ul style="list-style-type: none"> • Concepts and Terminology • Kubernetes Architecture 	<ul style="list-style-type: none"> • Pods, Services, Nodes, Clusters, Deployments, and more • Architecture of Kubernetes, including the control plane and worker nodes
 Setting Up Your Environment	<ul style="list-style-type: none"> • Local Setup • Cloud Setup 	<ul style="list-style-type: none"> • Minikube, Docker Desktop with Kubernetes • EKS, GKE, AKS etc
 Deploy & Manage Applications	<ul style="list-style-type: none"> • Pods • ReplicaSets and Deployments • Services 	<ul style="list-style-type: none"> • Create and Manage Pods • Create ReplicaSets and Deployments to manage scaling and apps updates • Explore how Services expose your apps in-cluster or externally
 Storage and Configurations	<ul style="list-style-type: none"> • Persistent Storage • Configuration Management 	<ul style="list-style-type: none"> • Master PVs and PVCs for app storage • Explore ConfigMaps and Secrets for managing app configuration data
 Advanced Topics	<ul style="list-style-type: none"> • Networking • Security • Monitoring and Logging 	<ul style="list-style-type: none"> • Explore networking, including Ingress, policies, and Service meshes • Study RBAC and security policies • Explore tools like Prometheus, Grafana, Elasticsearch etc.,
 Helm and Operators	<ul style="list-style-type: none"> • Helm • Operators 	<ul style="list-style-type: none"> • Learn Helm to package, share, and deploy applications • Understand Kubernetes Operators for automating complex , stateful apps
 Practice and Projects	<ul style="list-style-type: none"> • Hands-On Projects • Troubleshooting 	<ul style="list-style-type: none"> • Create apps on Kubernetes, starting simple and going bigger • Learn how to troubleshoot common issues and problems
 Community and Resources	<ul style="list-style-type: none"> • Join the Community • Documentation and Tutorials 	<ul style="list-style-type: none"> • Participate in Kubernetes forums, mailing lists, and meetups to learn from others • Stay updated with Kubernetes official documentation, blogs, and tutorials
 Validation	Certifications	Consider taking the Certified Kubernetes Administrator (CKA) or Certified Kubernetes Application Developer (CKAD) exams
 Keep Learning	Stay Informed	Kubernetes is rapidly evolving, so keep learning and experimenting with new features and best practices



Kubernetes POD Troubleshooting Techniques





Exercises for Section 5

- Installation of Docker, Minikube , Docker Kubernetes extension
- Create docker image using DockerFile
- Store container data in volumes
- Deploy kubernetes yaml files in minikube and AKS
- Expose application to access outside
- Use ChatGPT to generate snippets for various components - Azure Devops, Docker, Kubernetes