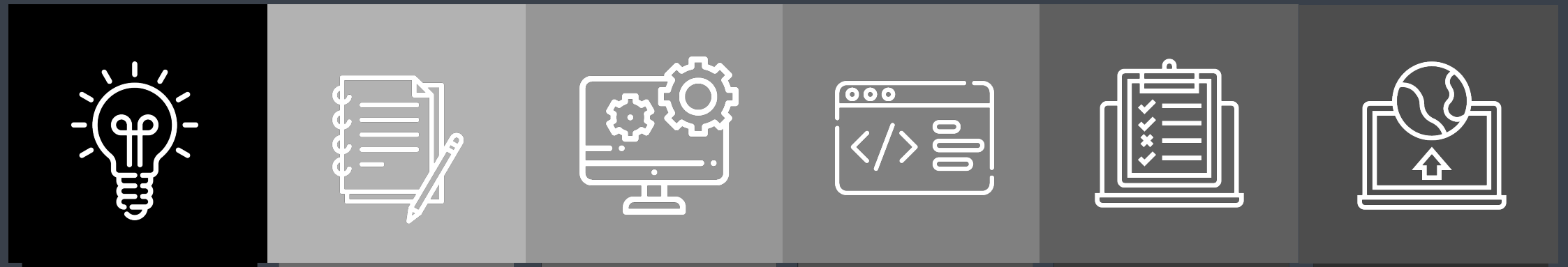
# DevOps

# Software Development Lifecycle



**PLANNING**

Talk to customer and understand the requirements

Requirement Gathering & Analysis

**DEFINING**

Define the requirements and stick to them

SRS

devs testers

**DESIGNING**

Design the solution with right approach

DDS

architecture(s), screens / workflow

**BUILDING**

Development following guidelines

languages / IDEs / SDKs / SCM

TDD

**TESTING**

Make sure that your code is working

testers

**DEPLOYMENT**

Make your app available for rest of the world

operations

prod

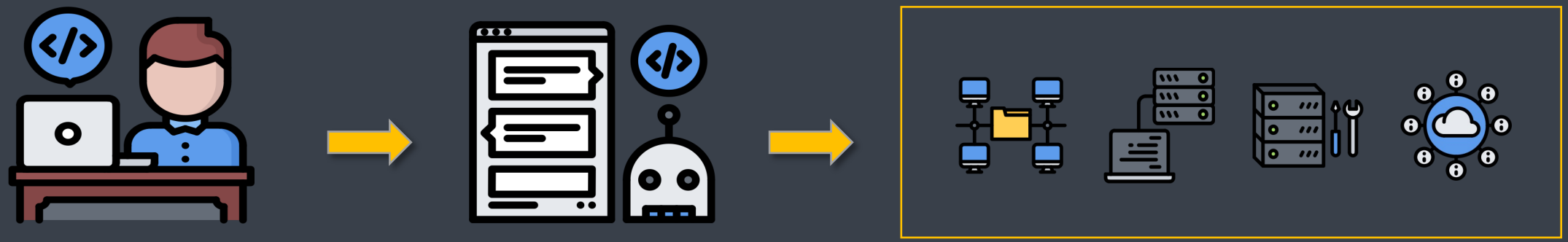
# Waterfall Model

- Requirement Specification
- System Design
- Design Implementation
- Verification & Testing
- System Deployment
- Software Maintenance

Iterative → modules

agile → stories → sprint → Scrum

# Entities involved



Developer

Testers

Operations Team

Dev team

operations

# Responsibilities

Dev team

## Developers and Testers

- Developers → SRS + DDS
  - Develop the application (programming)
  - Package the application (building)
  - Fix the bugs
  - Maintain the application
- Testers
  - Thoroughly test the application manually or using test automation
  - Report the bugs to the developer

## Operations Team

- Make all the necessary resources ready
  - licenses
  - machine
    - → RAM
    - → storage
  - Network / internet
  - other devices → printer
- Deploy the application
- Maintain multiple environments
  - → dev
  - → staging
  - → pre-prod
  - → prod
- Continuously monitor the application
- Manage the resources
- maintain uptime

Deployment

- Desktop → setup / msi
- Web → cloud
  - strategy
  - resources
  - providers
  - services
  - model
- Mobile → respective market
  - ios → app store
  - android → play store

## Dev team

**Developer**

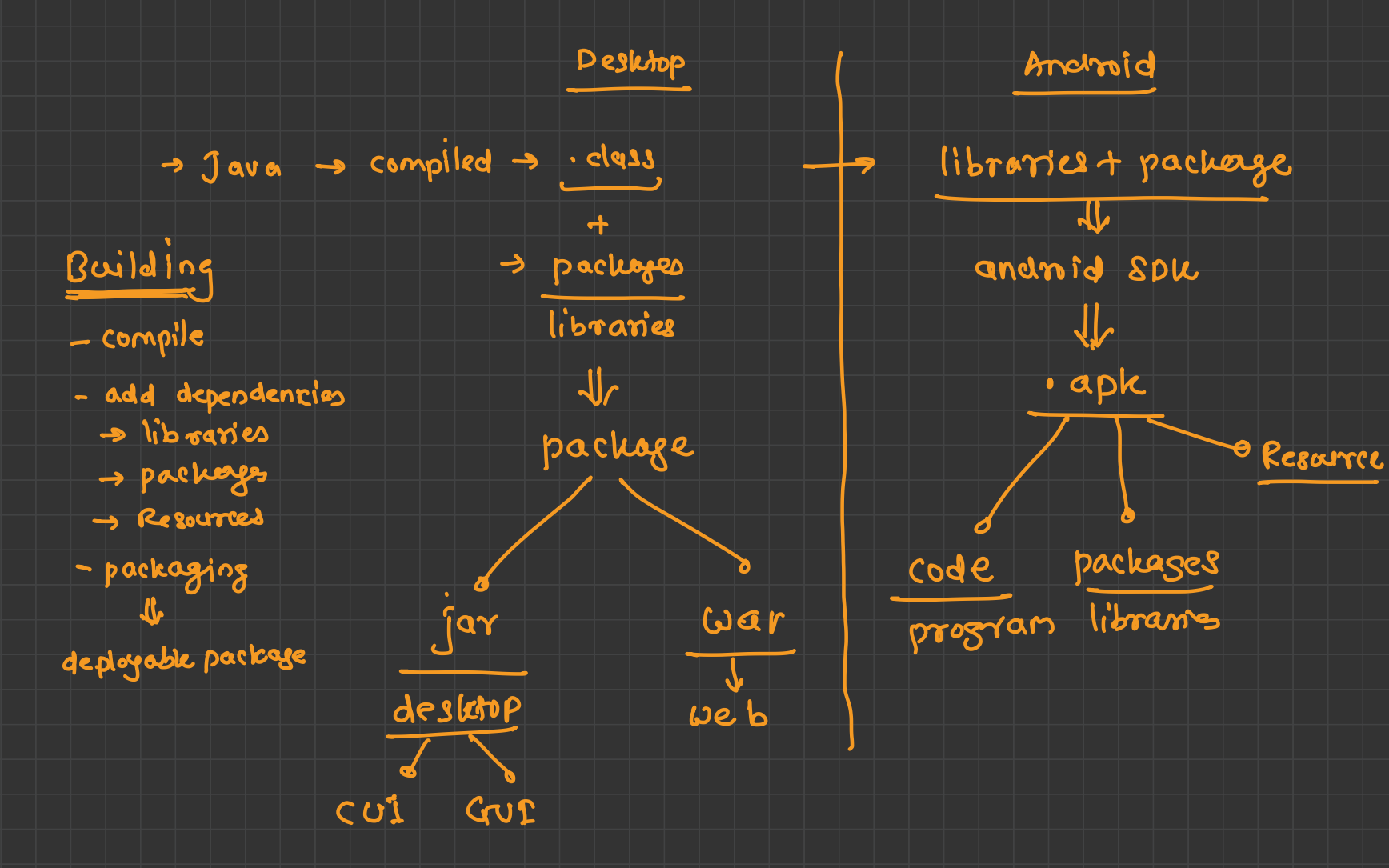
- programming skill
- soft skill

**Tester**

- testing skill

## operations

- administrator of OS
  - windows
  - Linux
- networking skill
- diagnostics skill
- HW knowledge


Desktop
Android
→ Java → compiled → .class
+
→ packages
libraries
libraries + package
android SDK
.apk
Resource
code
program
packages
libraries
Building
- compile
- add dependencies
→ libraries
→ packages
→ Resources
- packaging
deployable package
package
jar
desktop
CUI
GUI
war
web

# Challenges

## Developers and Testers

- The process is slow
- The pressure to work on the newer features and fix the older code
- Not flexible

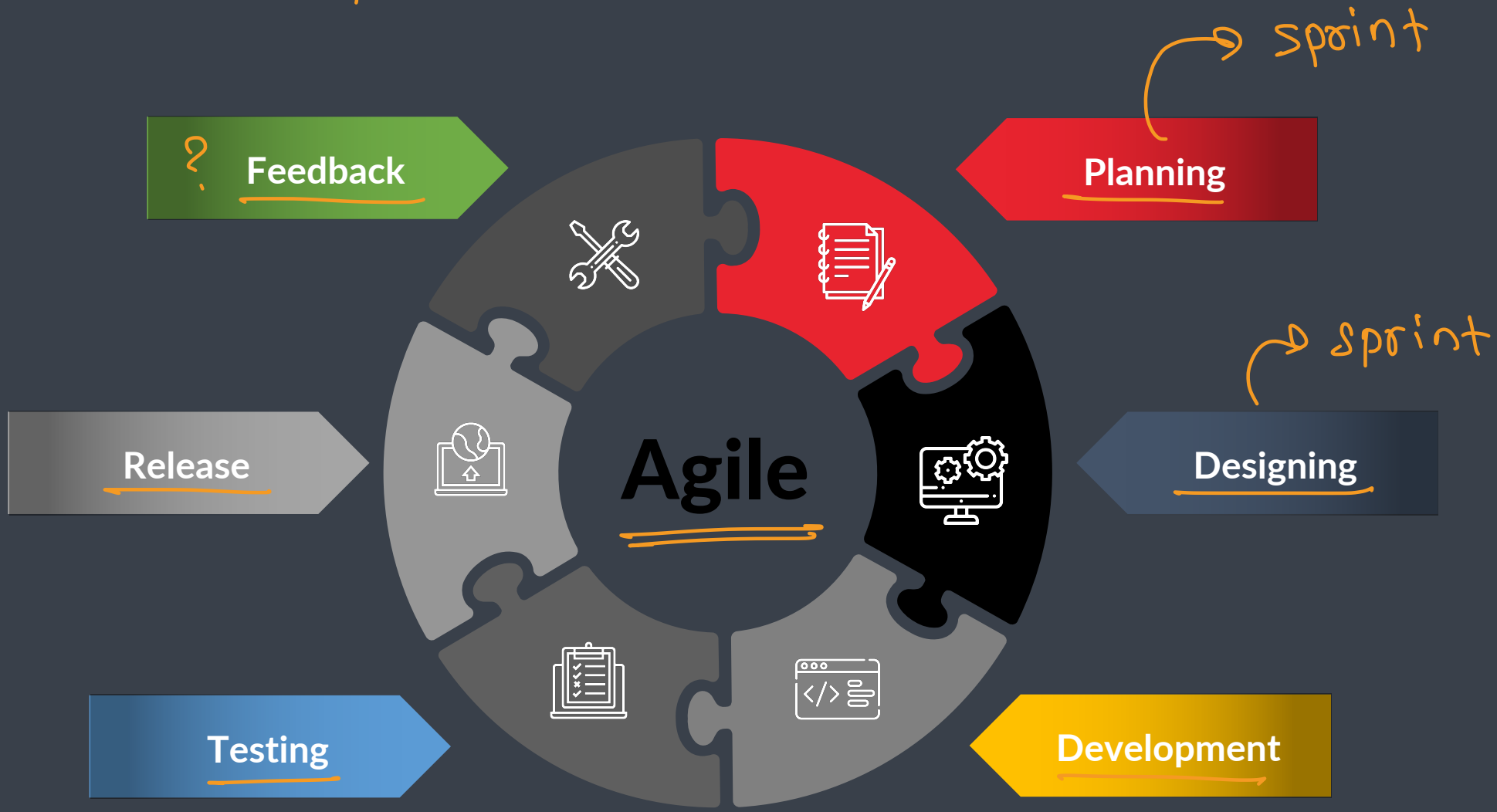
## Operations Team

- Uptime → monitoring app
- Configure the huge infrastructure
- Diagnose and fix the issue

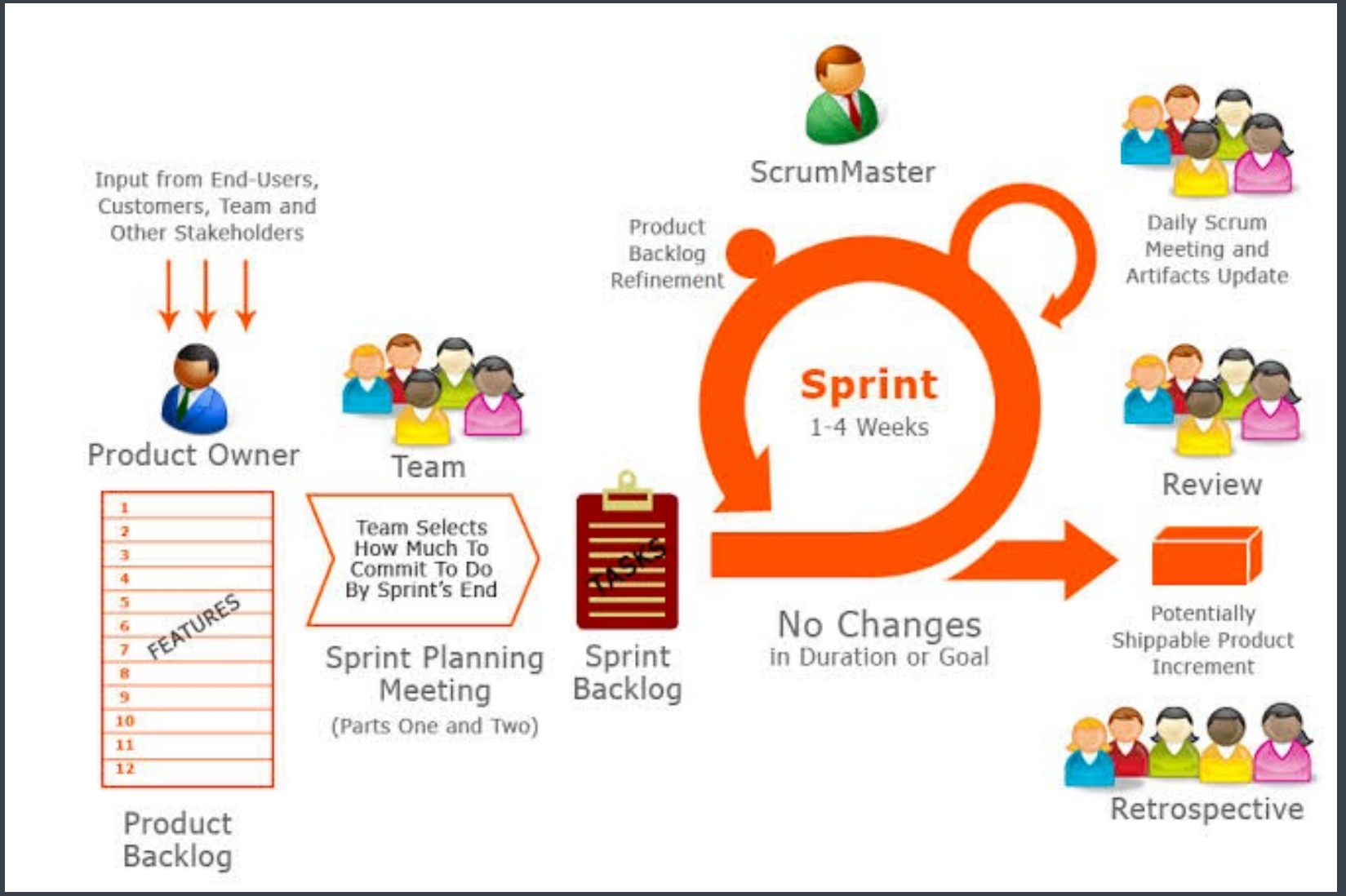
infrastructure
- machines (physical, virtual)
- network
- other device
- security equipments

# Agile Development

Sprint - based



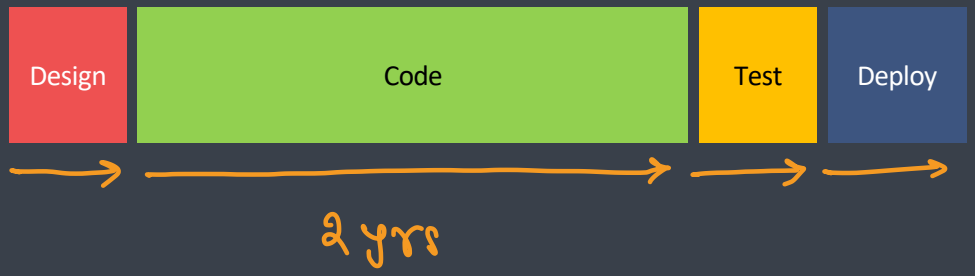
# Scrum Process



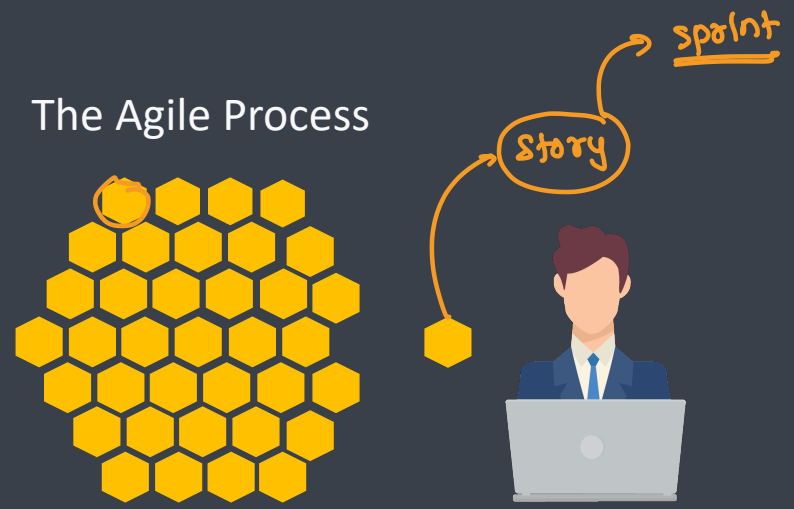
# Waterfall Vs Agile

## The Waterfall Process

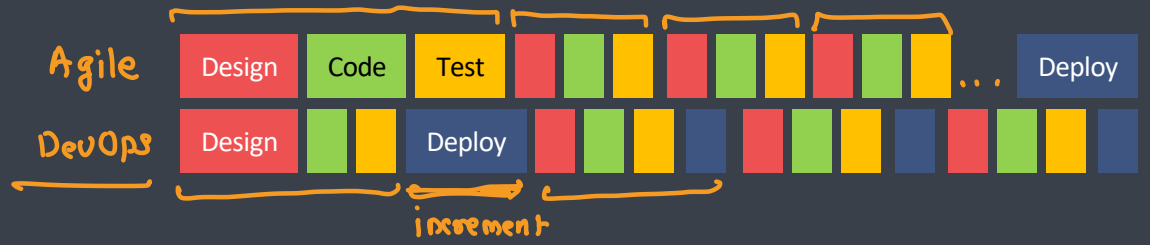

This project has got so big.
I am not sure I will be able to deliver it!




## The Agile Process




It is so much better delivering
this project in bite-sized sections



# Problems

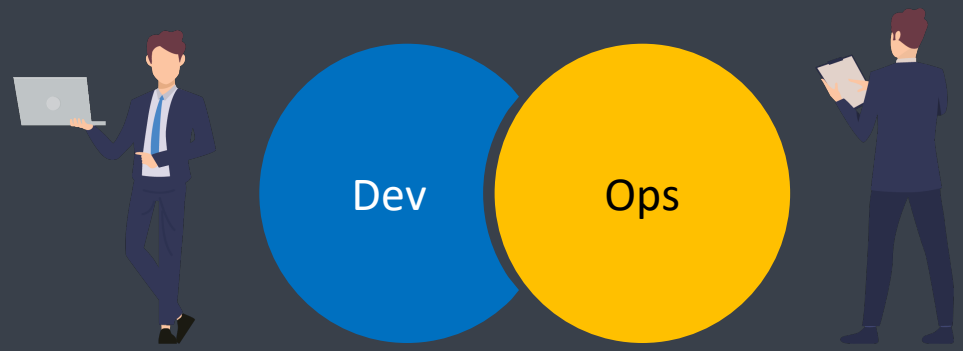
- Managing and tracking changes in the code is difficult
- Incremental builds are difficult to manage, test and deploy
- Manual testing and deployment of various components/modules takes a lot of time
- Ensuring consistency, adaptability and scalability across environments is very difficult task
- Environment dependencies makes the project behave differently in different environments

# Solutions to the problem → devops

- Managing and tracking changes in the code is difficult: **SCM tools** → git
- Incremental builds are difficult to manage, test and deploy: **Jenkins** → CI/CD
- Manual testing and deployment of various components/modules takes a lot of time: **Selenium** : Test Automation
- Ensuring consistency, adaptability and scalability across environments is very difficult task: **Puppet** : configuration tools
- Environment dependencies makes the project behave differently in different environments: **Docker** → deployment tools

# What is DevOps ?

- DevOps is a combination of two words development and operations
- Promotes collaboration between Development and Operations Team to deploy code to production faster in an automated & repeatable way
- DevOps helps to increases an organization's speed to deliver applications and services
- It allows organizations to serve their customers better and compete more strongly in the market
- Can be defined as an alignment of development and IT operations with better communication and collaboration
- DevOps is not a goal but a never-ending process of continuous improvement
- It integrates Development and Operations teams
- It improves collaboration and productivity by
  - Automating infrastructure
  - Automating workflow
  - Continuously measuring application performance



# Why DevOps is Needed?

- Before DevOps, the development and operation team worked in complete isolation
- Testing and Deployment were isolated activities done after design-build. Hence they consumed more time than actual build cycles.
- Without using DevOps, team members are spending a large amount of their time in testing, deploying, and designing instead of building the project.
- Manual code deployment leads to human errors in production
- Coding & operation teams have their separate timelines and are not in synch causing further delays

# Common misunderstanding

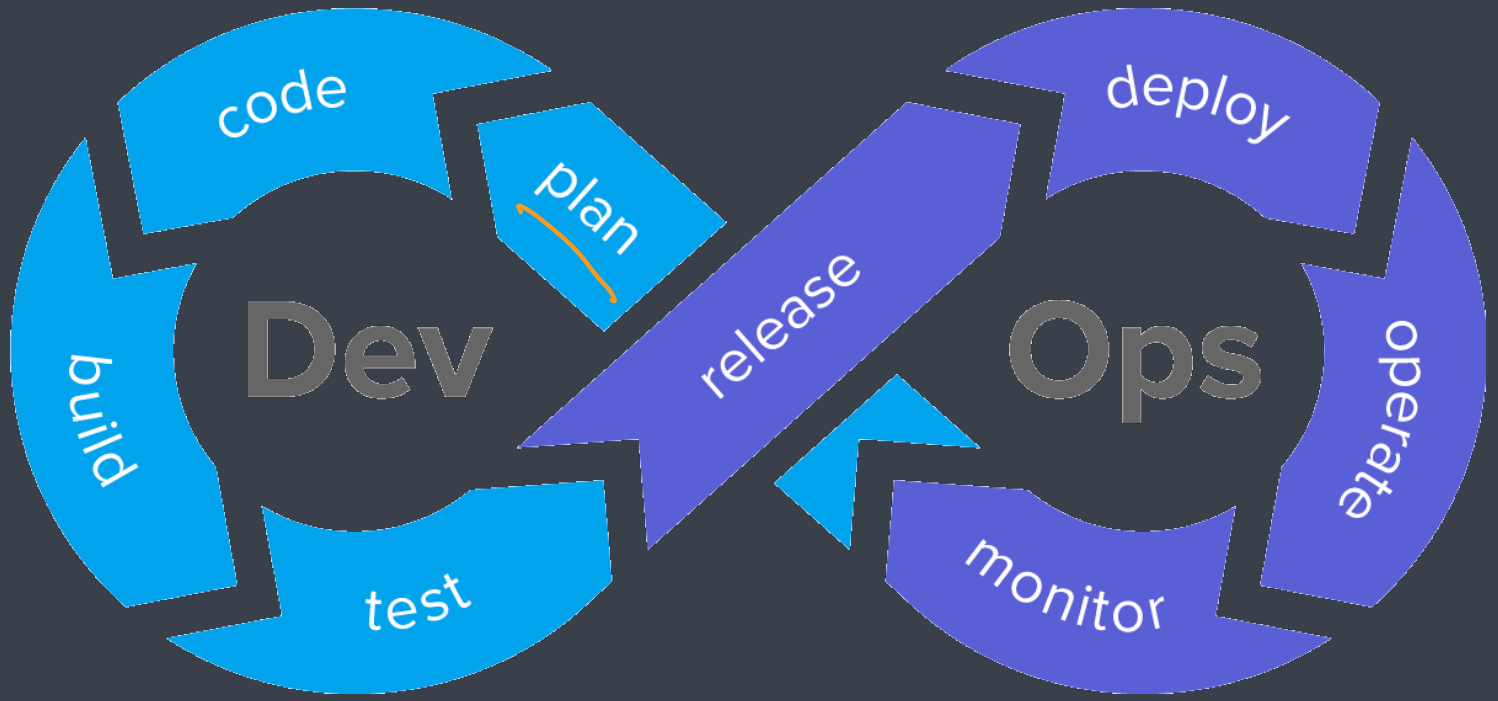
- DevOps is not a role, person or organization
- DevOps is not a separate team
- DevOps is not a product or a tool
- DevOps is not just writing scripts or implementing tools

# Reasons to use DevOps

- **Predictability**
  - DevOps offers significantly lower failure rate of new releases
- **Reproducibility**
  - Version everything so that earlier version can be restored anytime
- **Maintainability**
  - Effortless process of recovery in the event of a new release crashing or disabling the current system
- **Time to market**
  - DevOps reduces the time to market up to 50% through streamlined software delivery
  - This is particularly the case for digital and mobile applications
- **Greater Quality**
  - DevOps helps the team to provide improved quality of application development as it incorporates infrastructure issues
- **Reduced Risk**
  - DevOps incorporates security aspects in the software delivery lifecycle. It helps in reduction of defects across the lifecycle
- **Resiliency**
  - The Operational state of the software system is more stable, secure, and changes are auditable

# DevOps Lifecycle

sprint based



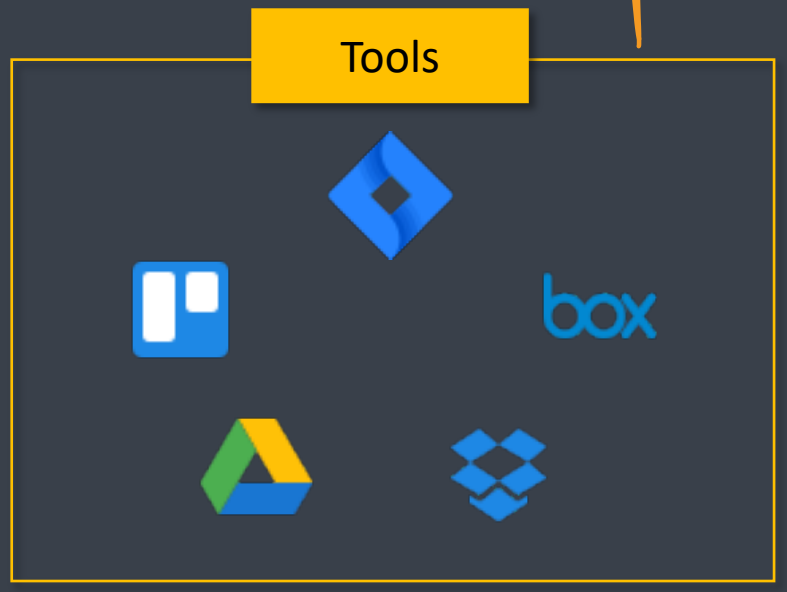
# DevOps Lifecycle - Plan : plan the sprint

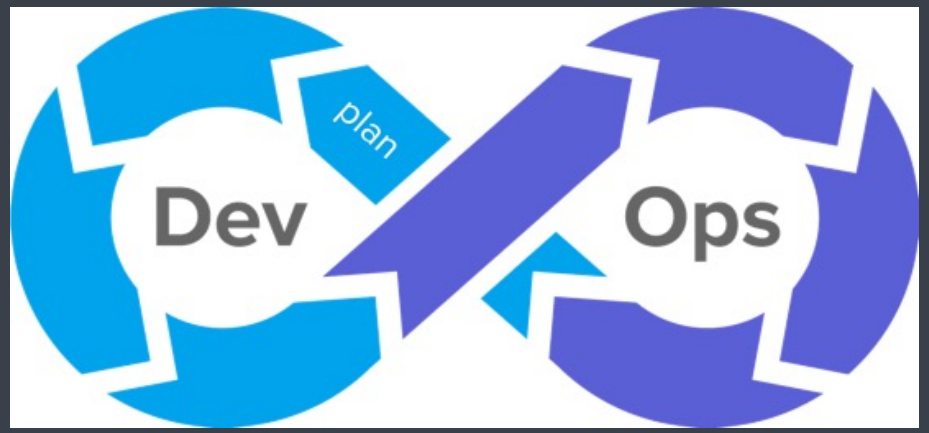
- First stage of DevOps lifecycle where you plan, track, visualize and summarize your project before you start working on it

- text file(s)
- Excel file(s)
- cloud based
  - drop box
  - one drive , box

planning tools
- jira
- trello
- GitScrum

planning
- stories → creation & selection
- resources → availability
- deployment



# DevOps Lifecycle - Code

- Second stage where developer writes the code using favorite programming language

**SCM tools**
- SVN
- CVS
- bazaar
- bitkeeper
- git

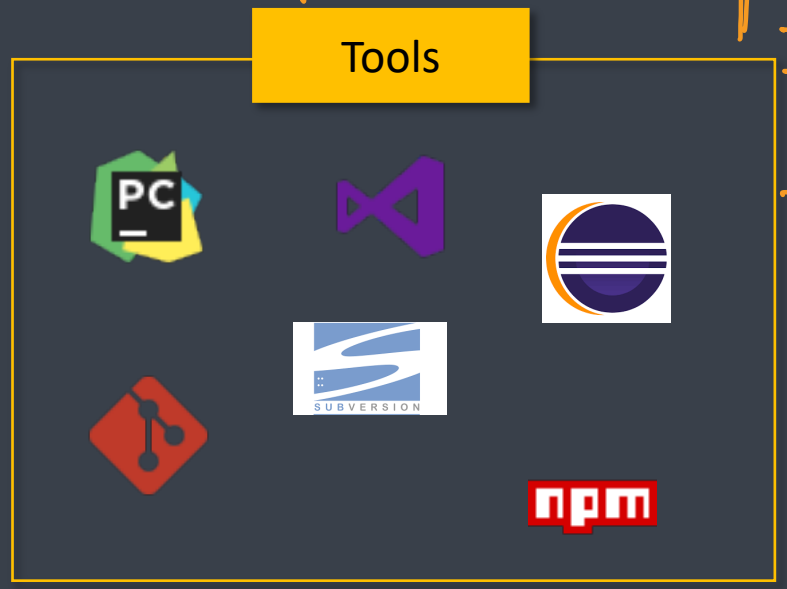
**language**
- C, C++, go
- python, perl, php
- html / css, JS
- JS, Java, C#, ruby,

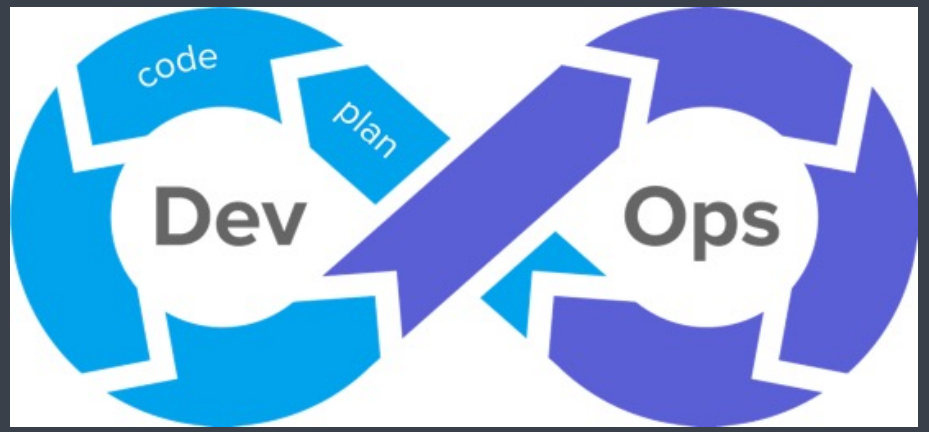
**SDK**
- tool chain
  - compiler
  - linker
  - interpreter
  - debugger
  - assembler
  - disassembler
- documentation
- libraries / package / namespaces
- headers
- eg.
  - C/C++ SDK
  - .Net SDK
  - Java SDK
  - android SDK
  - iOS SDK

**Editors**
- vim
- vscode
- atom
- sublime

**IDEs**
- C/C++ → VS, CLion, Dev C, CodeBlock
- Java → Eclipse / netbeans / IDEA
- JS/TS → Webstorm
- Java / kotline (android) - Android IDE
- swift / objective-C → xcode
- PHP - PHPstorm
- C# - VS

Tools



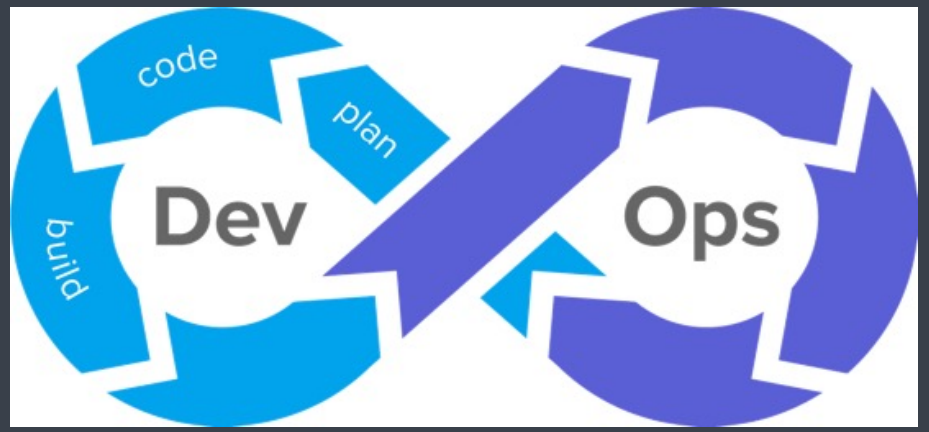
# DevOps Lifecycle -Build

= compile + add dependencies
⇒ package the application

- Integrating the required libraries
- Compiling the source code
- Create deployable packages

tools
- ant → deprecated
- maven
- gradle - ***
- xcodebuild
- npm - build
  - webpack
  - bundler

package
- android : .apk
- ios : .ipa
- web : bundle/webpack
- java : .jar / .war
- .net : .xaml

# DevOps Lifecycle - Test

- Process of executing automated tests
- The goal here is to get the feedback about the changes as quickly as possible

**unit testing**

- JS/TS → Jest / Jasmine
- python - PyUnit
- Java - JUnit
- C# - NUnit

**UI testing**

web - cypress / selenium
mobile - apium
native - WinRunner

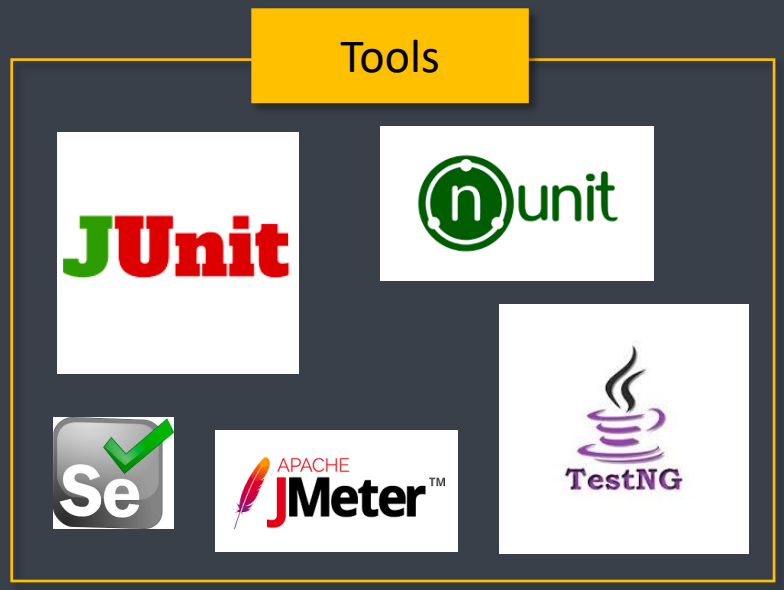
**performance testing**

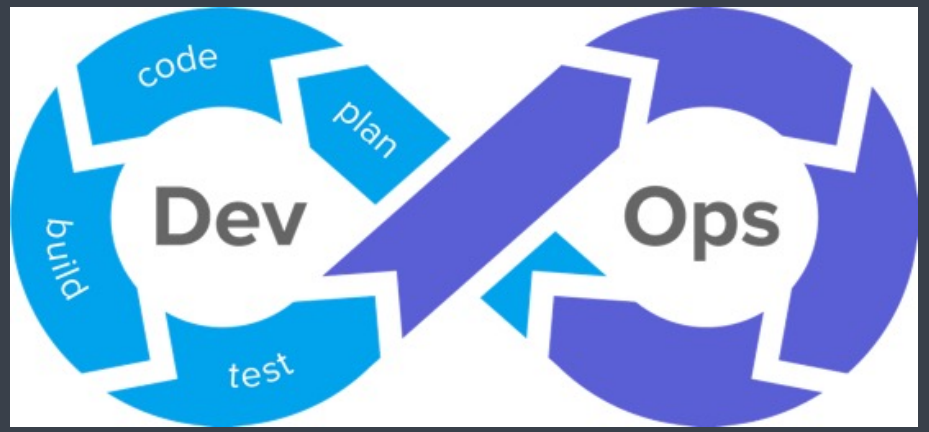
- JMeter
- LoadRunner

**Stress testing**

- Stress tester

Tools



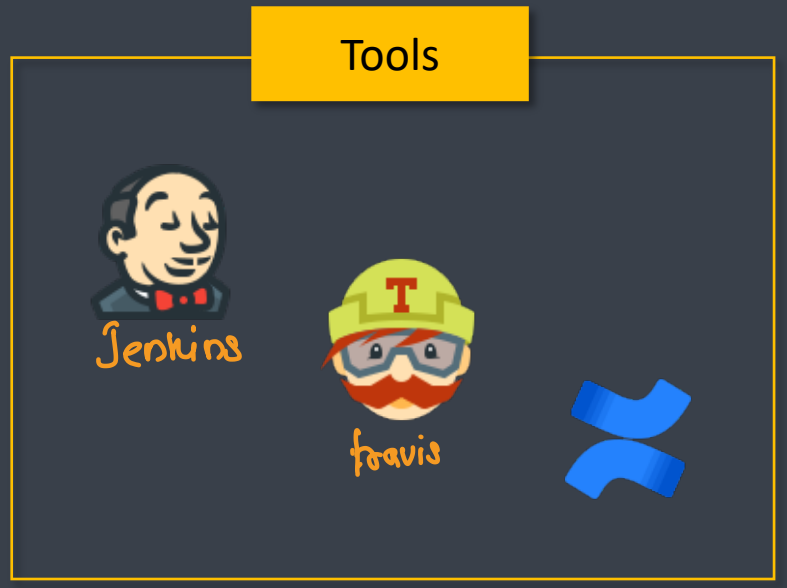



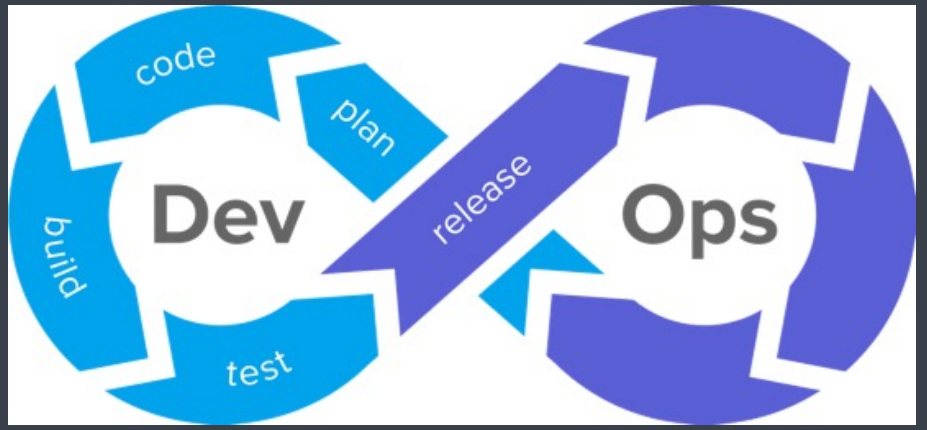
# DevOps Lifecycle - Release

- This phase helps to integrate code into a shared repository using which you can detect and locate errors quickly and easily

CI/CD pipeline → sequence of stages (scm, build, test, env creation, env config, deployment)

tools
- CI tools → Travis CI, CircleCI
- DI tools → ArgoCD
- CI/CD tools → Jenkins, Bamboo



# DevOps Lifecycle - Deploy

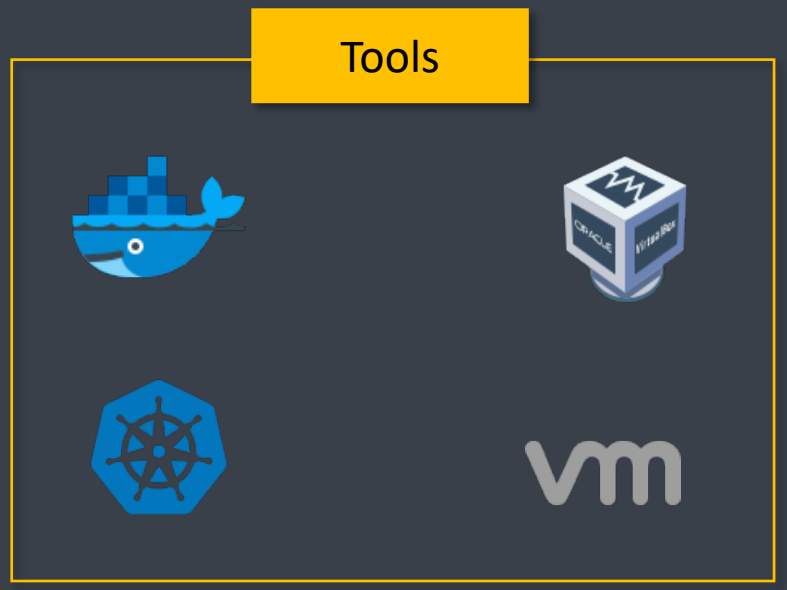
- Manage and maintain development and deployment of software systems and server in any computational environment

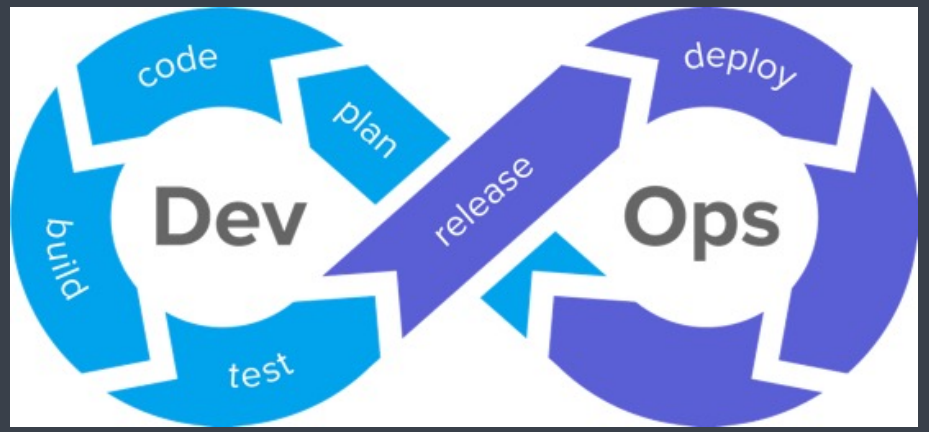
Deployment strategies

1. traditional deployment - using physical machines
2. virtualized deployment - using VM
   - on-prem → VMware, virtualbox, parallels, bosh
   - cloud → AWS EC2, Azure VM, GCP VM
3. containerized deployment → using containers
   - containers → docker, podman, cri-o, rkt, lmctfy
   - orchestration → docker swarm, kuberbetes, mesos, marathon

horizontally scaled

Tools



# DevOps Lifecycle - Operate

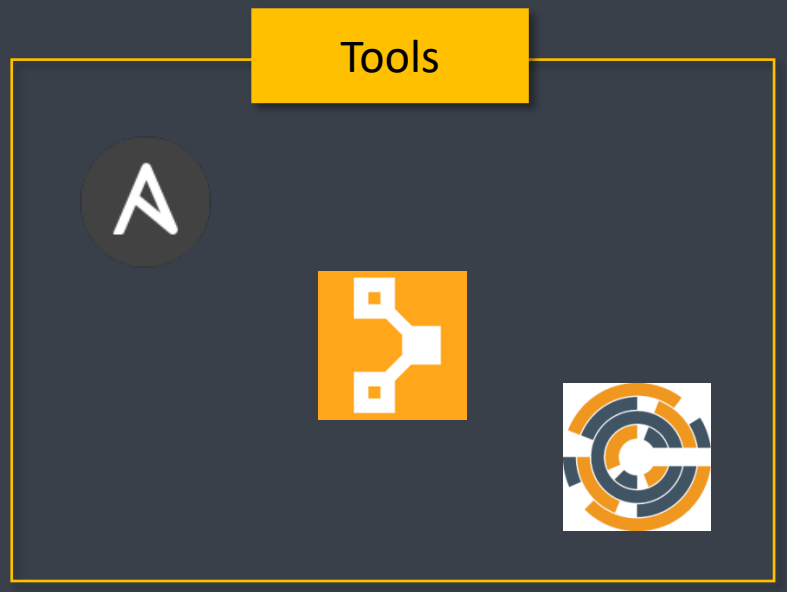
- This stage where the updated system gets operated

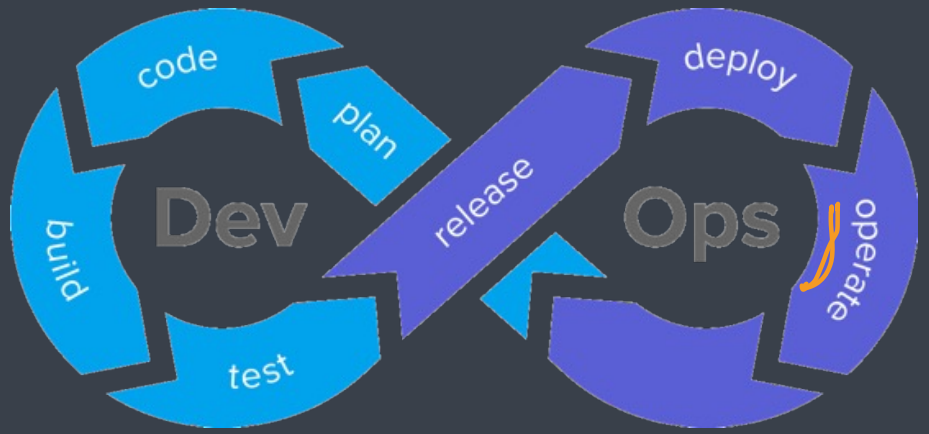
Env creation tools - infra

- terraform <- AWS, Azure, GCP
- AWS - cloud formation
- local - vagrant

Env configurations tools

- ansible
- puppet
- chef
- SaltStack



# DevOps Lifecycle - Monitor

- It ensures that the application is performing as expected and the environment is stable
- It quickly determines when a service is unavailable and understand the underlying causes

tools

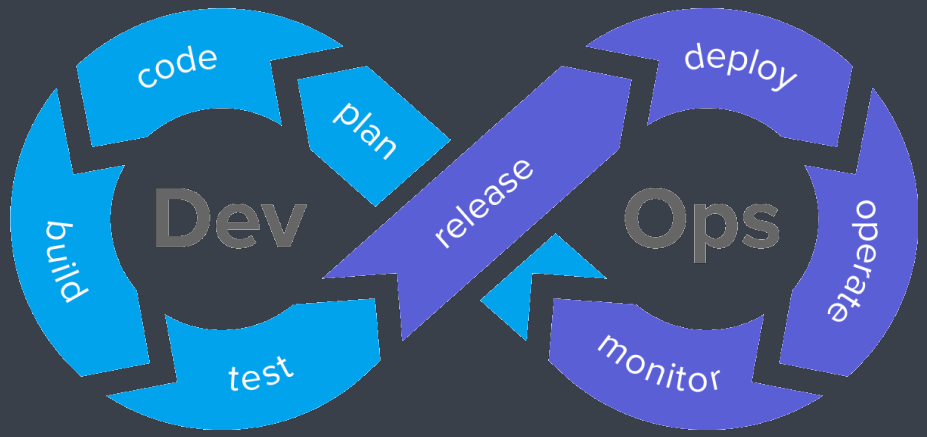
- splunk
- logstash
- DataDog
- Nagios

Tools

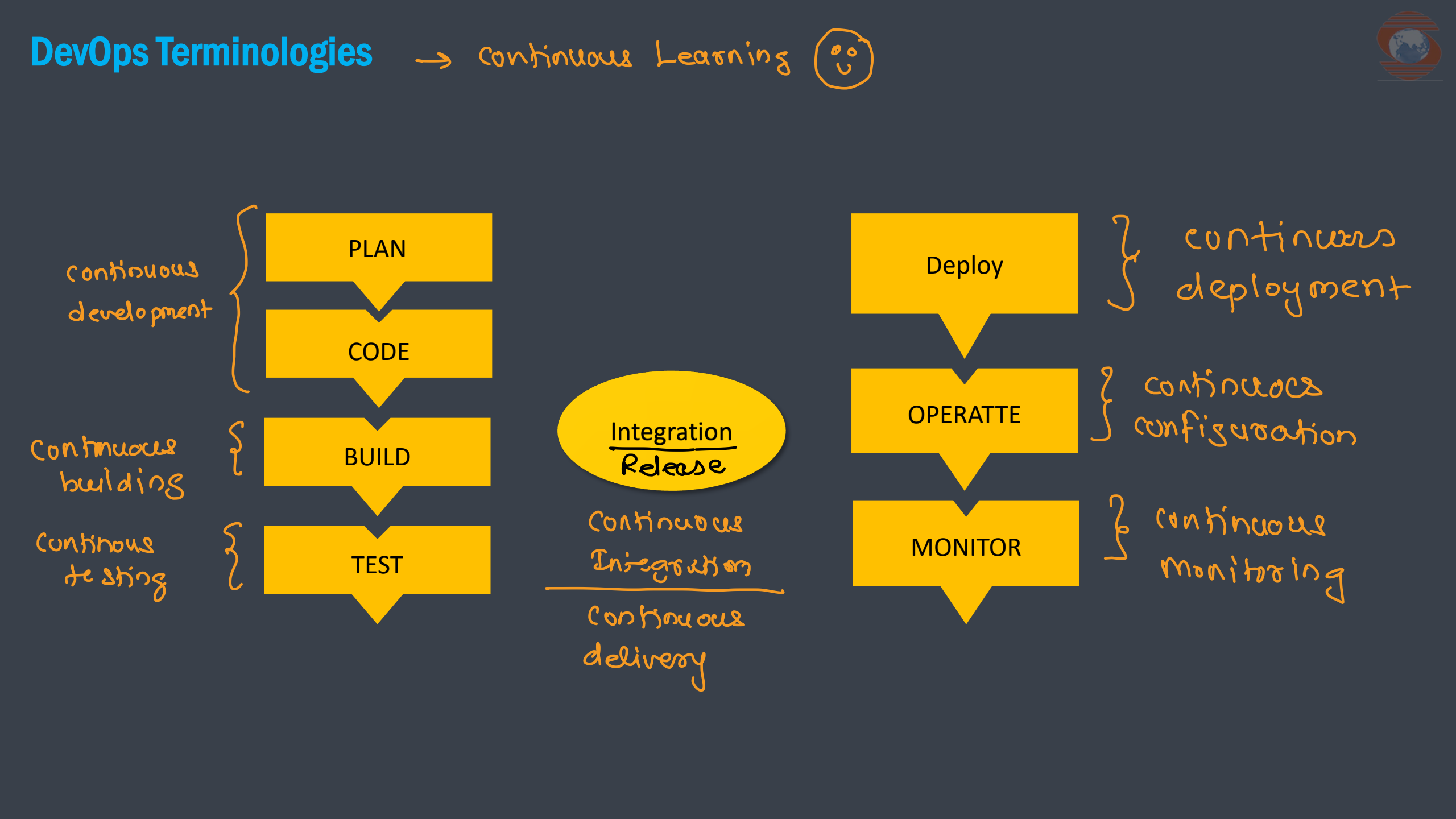
splunk>

DATADOG

Nagios®

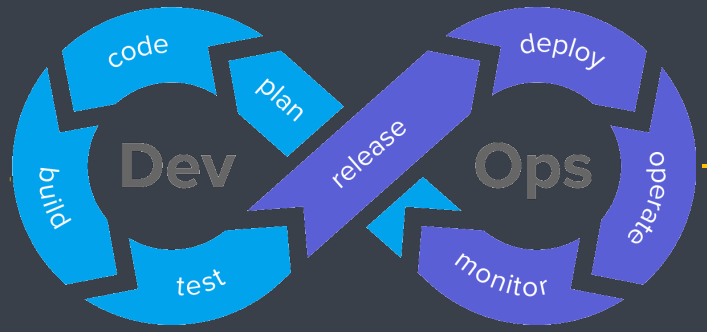


# DevOps Terminologies → Continuous Learning



# Responsibilities of DevOps Engineer

- Be an excellent sysadmin
- Deploy Virtualization
- Hands-on experience in network and storage
- Introduction to coding




- Soft skills
- Automation tools
- Software Testing knowledge
- IT security

# Skills of a DevOps Engineer

| Skills | Description |
| --- | --- |
| Tools | • Version Control – Git/SVN<br>• Continuous Integration – Jenkins<br>• Virtualization / Containerization – Docker/Kubernetes<br>• Configuration Management – Puppet/Chef/Ansible<br>• Monitoring – Nagios/Splunk |
| Network Skills | • General Networking Skills<br>• Maintaining connections/Port Forwarding |
| Other Skills | • Cloud: AWS/Azure/GCP<br>• Soft Skills<br>• People management skill |