Basics of DevOps

DevOps Objective

- To deliver Software/Software features to the user quickly and efficiently.
- Also known as the contact delivery to the business.

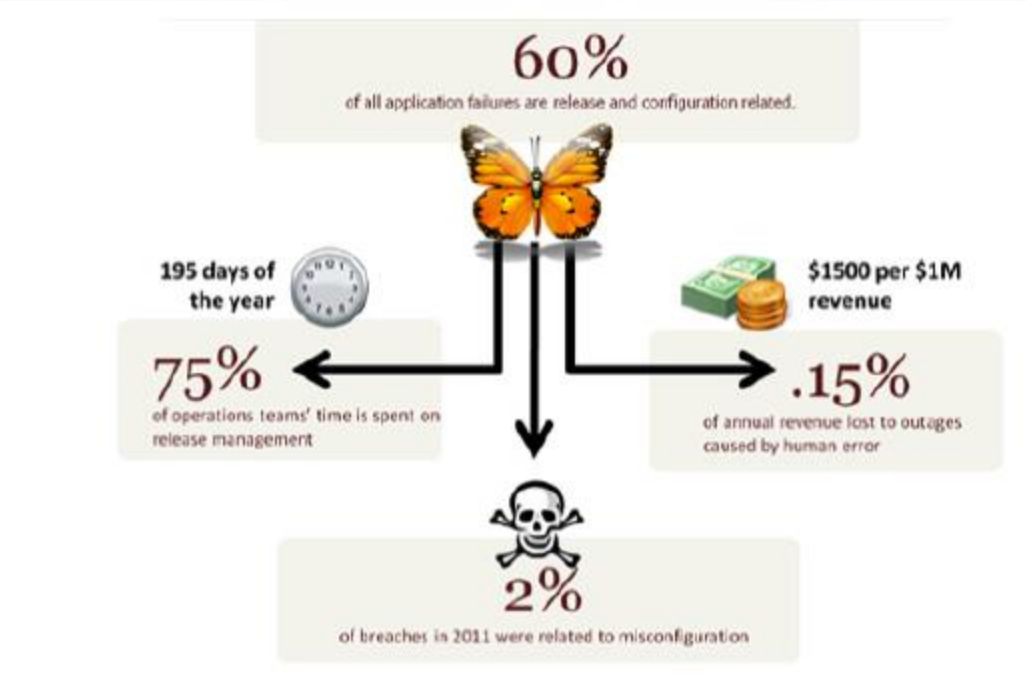
Traditional Software Development Life Cycle

- Gather the requirement
- Code the Software also knows as Developement
- Test the code
- Build the code, deploy to Developement servers and test it
- Deploy the software to QA servers for testing.
- Testers/QA test the software/application
- Operations team deploy it to Production servers
- Maintain the Softwares & Servers

Prod Deployment Issues

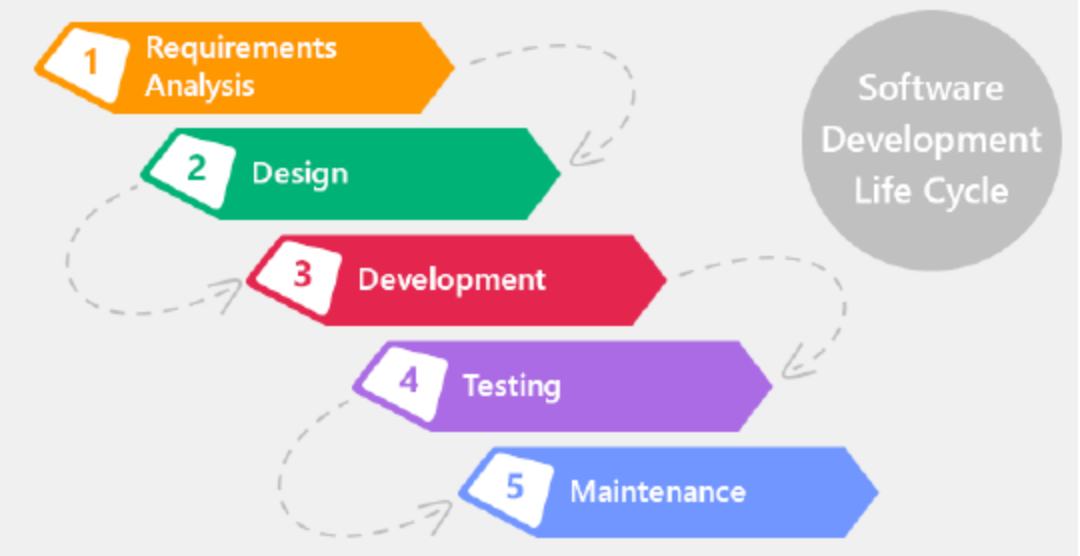
- Operations team deploys it but with a possibility that whatever was working in Development environment does not work in Production environment. The response from the development team usually is, “It works just fine in development…” The reason for the failure is that these two environments are different or are not periodically synchronised.
- New development tools and processes make coding faster but the operations team is not able to cope with frequent changes and releases.
- Production servers may need some tweaking or fine tuning at the database or OS level and also network security level.
- Developers usually do not have access to Production servers to check how the application is behaving so there is a need of feedback from end users, which, is not often received by the Developers for obvious reasons. In some instances there are no clear instructions / details of deployment. Ops team have to figure out some things based on their experience / skill – poor transition.

Traditional Deployment Stats

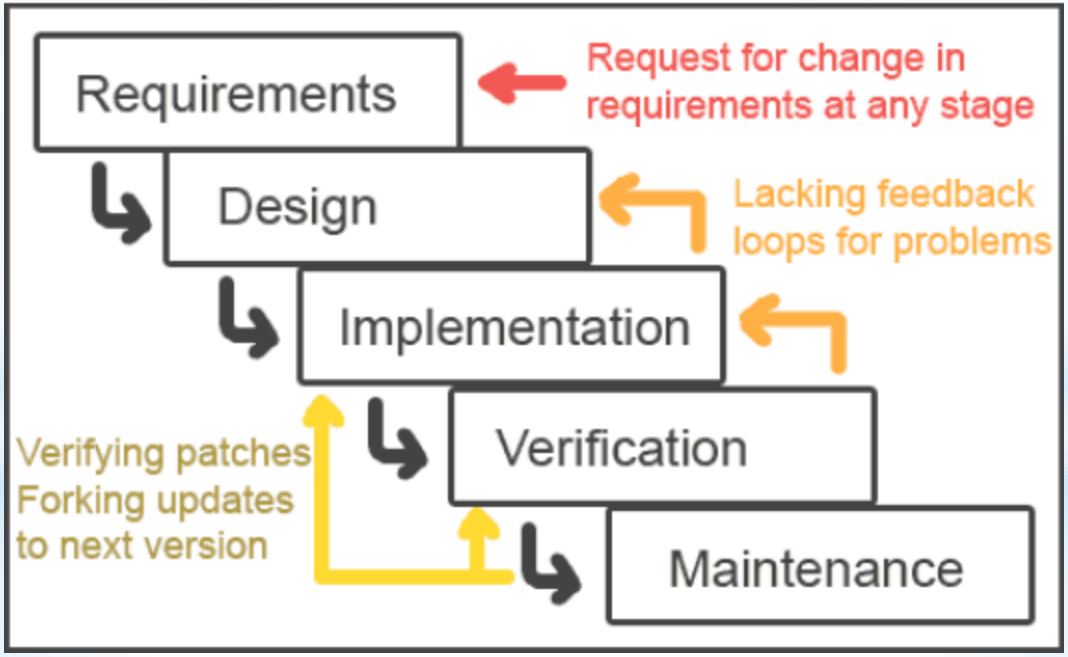


Traditional Development SDLC

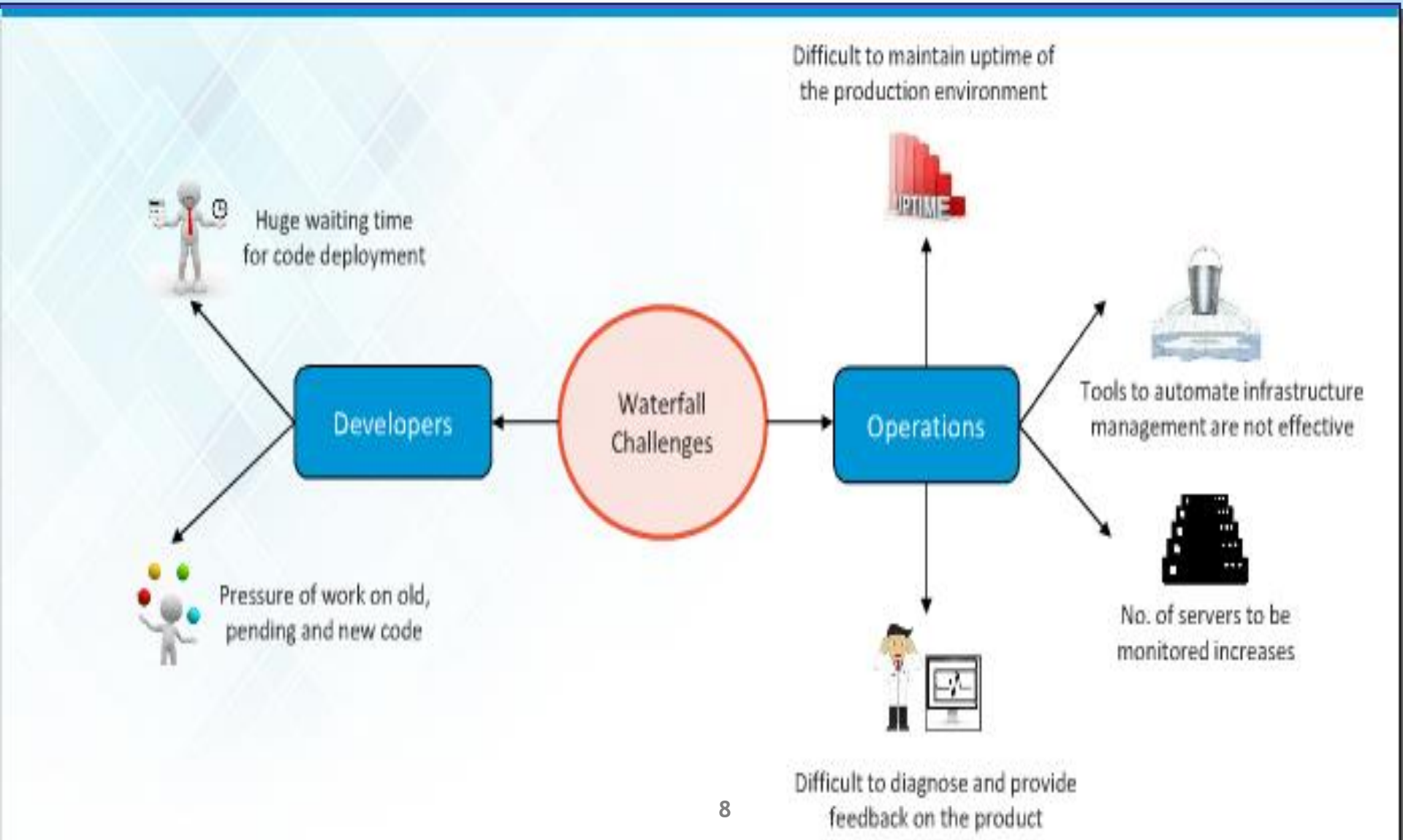
Waterfall



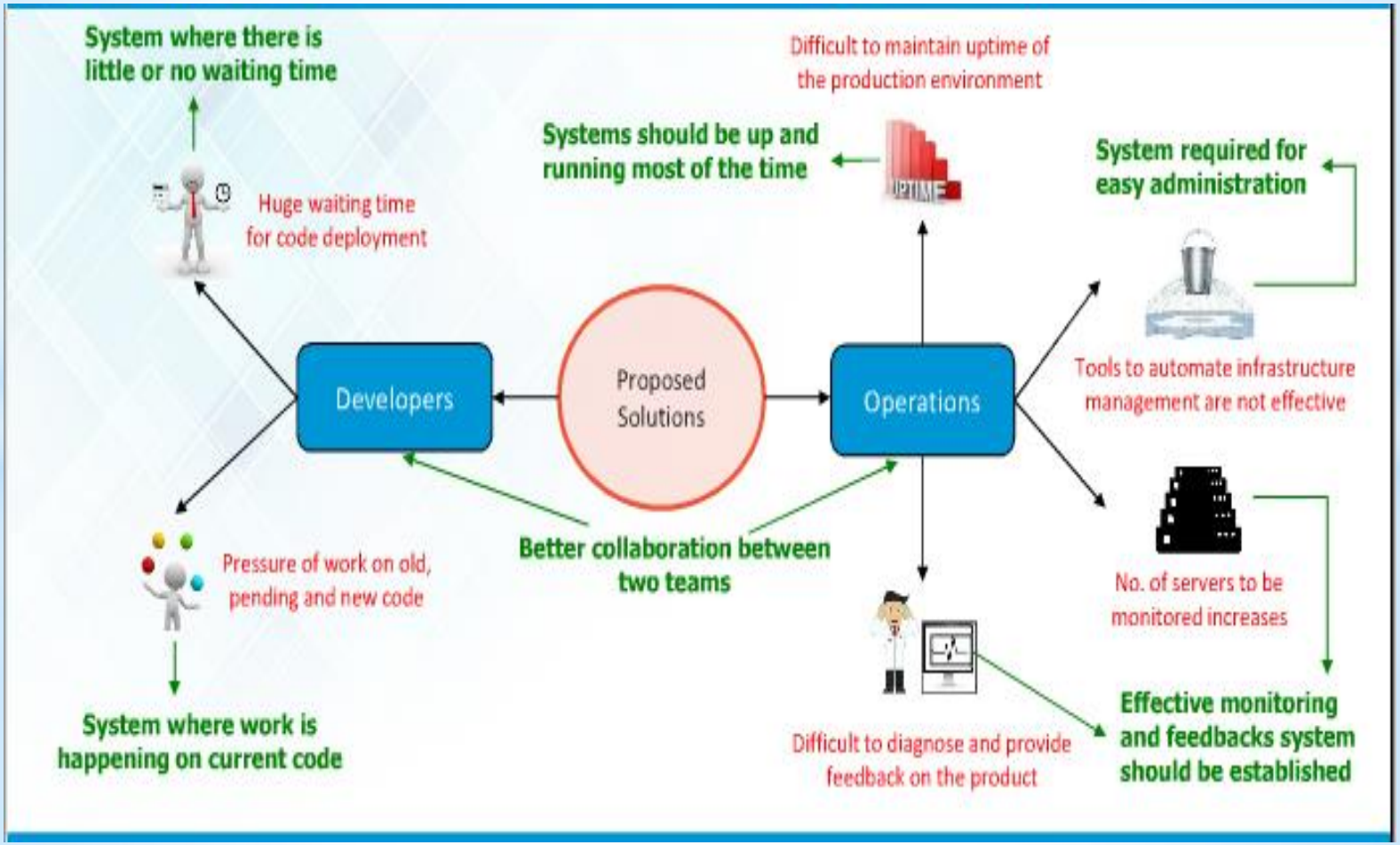
Issues with WaterFall

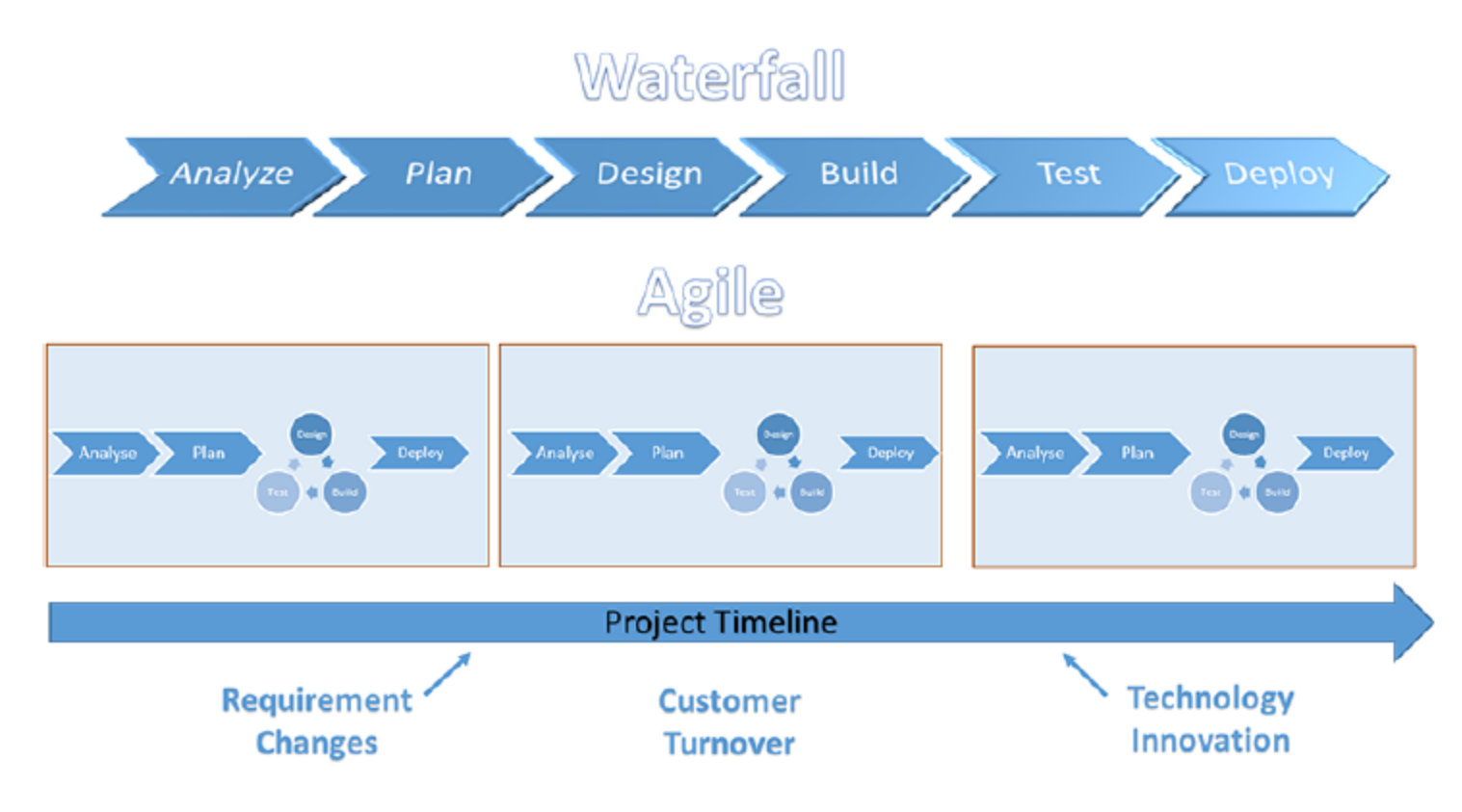


Waterfall Model Challenges

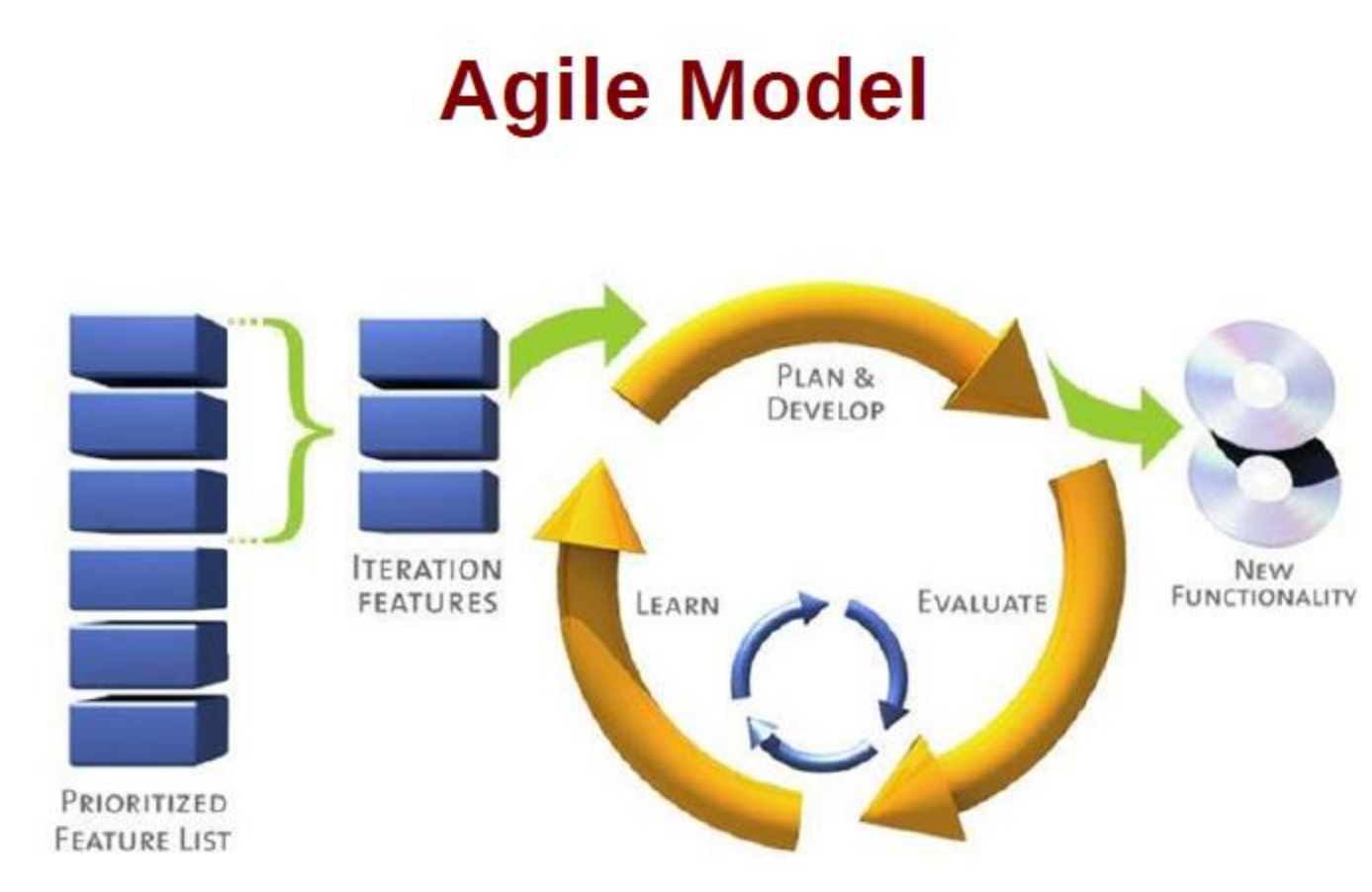


What need to be done ?

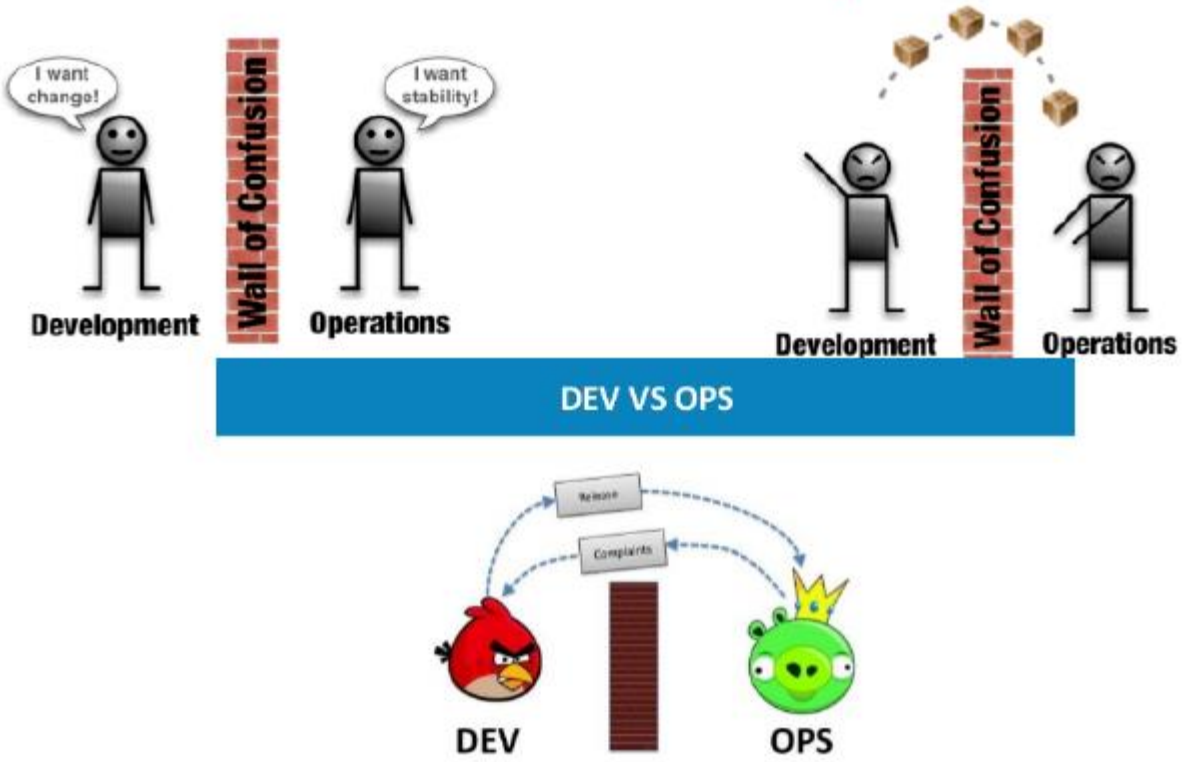



Waterfall
Analyze
Plan
Design
Build
Test
Deploy
Agile
Analyse
Plan
Design
Deploy
Test
Build
Project Timeline
Requirement Changes
Customer Turnover
Technology Innovation

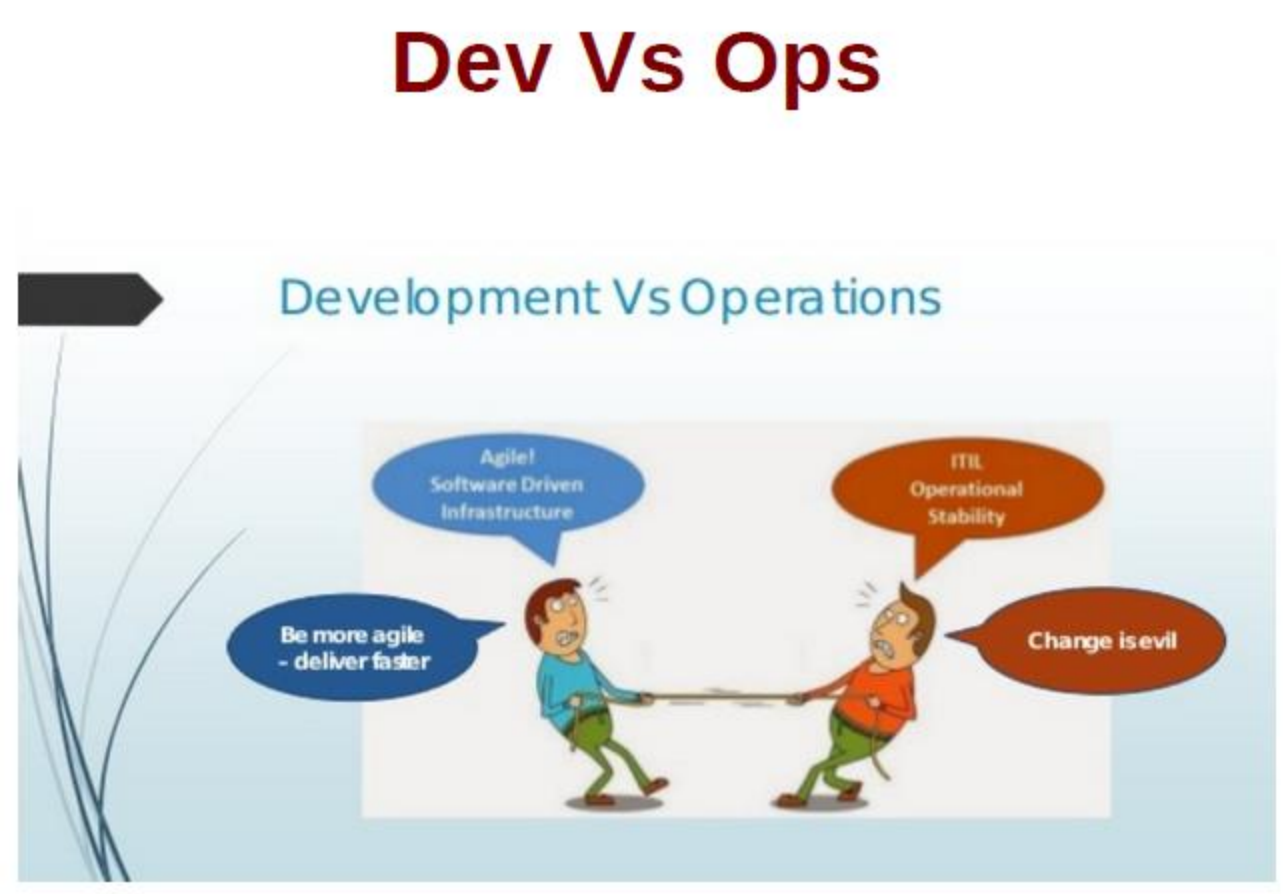
Agile Model


Agile Model
Prioritized Feature List
Iteration Features
Plan & Develop
Learn
Evaluate
New Functionality

Dev Vs Ops


I want change!
Wall of Confusion
I want stability!
Development
Operations
Development
Wall of Confusion
Operations
DEV VS OPS
Release
Complaints
DEV
OPS

Traditional Development Model



A Short History of DevOps

2008

- Software developer Patrick Debois - —developer, network specialist, system administrator, tester and project manager.
- Debois helps plant the seeds of the DevOps movement at the Agile conference in Toronto, resolve the conflict between the software developers and the operations teams when it comes to getting great work done quickly.

2009

- At the O'Reilly Velocity Conference, two Flickr employees—John Allspaw, senior vice president of technical operations, and Paul Hammond, director of engineering—deliver a seminal talk known as "10+ Deploys per Day: Dev and Ops Cooperation at Flickr."

- Debois launches the first Devopsdays event, in Ghent, Belgium. Early supporters include John Willis, an enterprise system management expert, and Kris Buytaert, a Linux and open source consultant.

2010

- The first US Devopsdays is organized, with the help of Willis.The events soon become a regular global series of community-organized conferences and a major force driving the DevOps community forward.

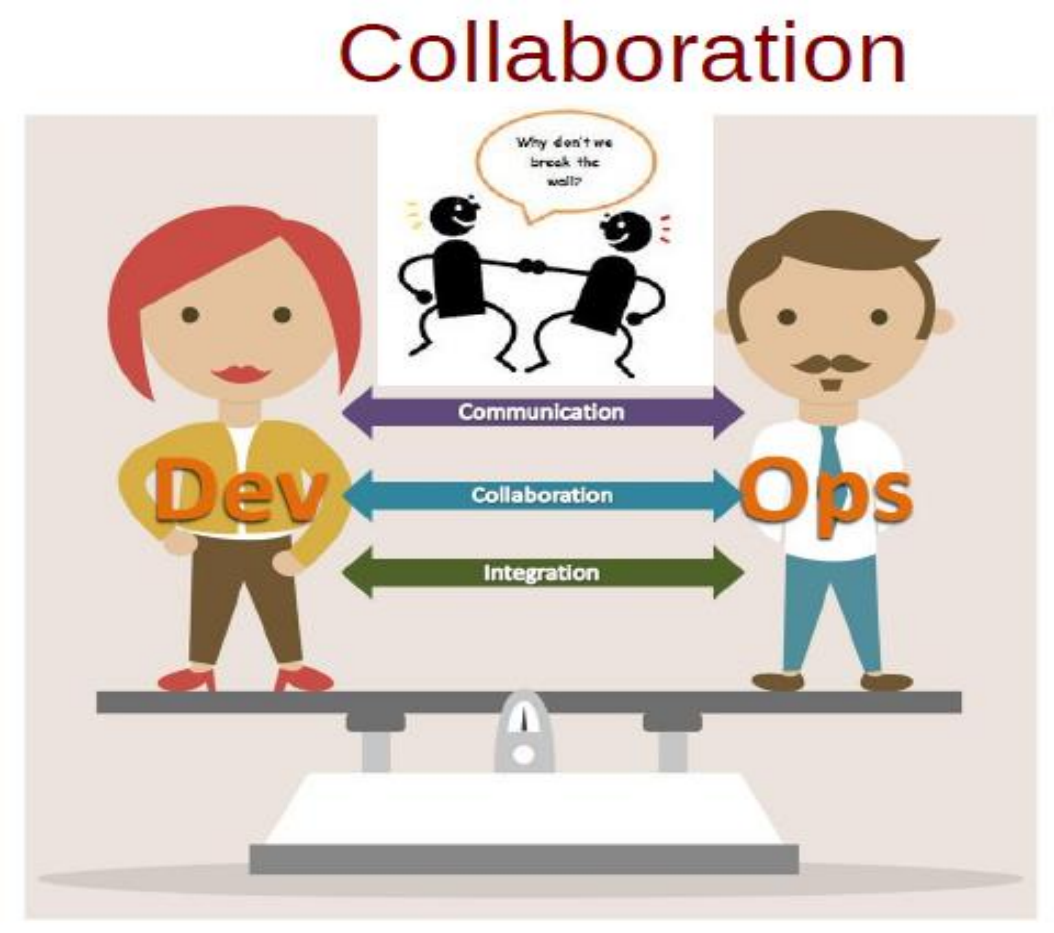
2011

- The DevOps community starts to build open source tools like Vagrant (for creating and configuring virtual development environments) that work with existing configuration management tools like Puppet and Chef.

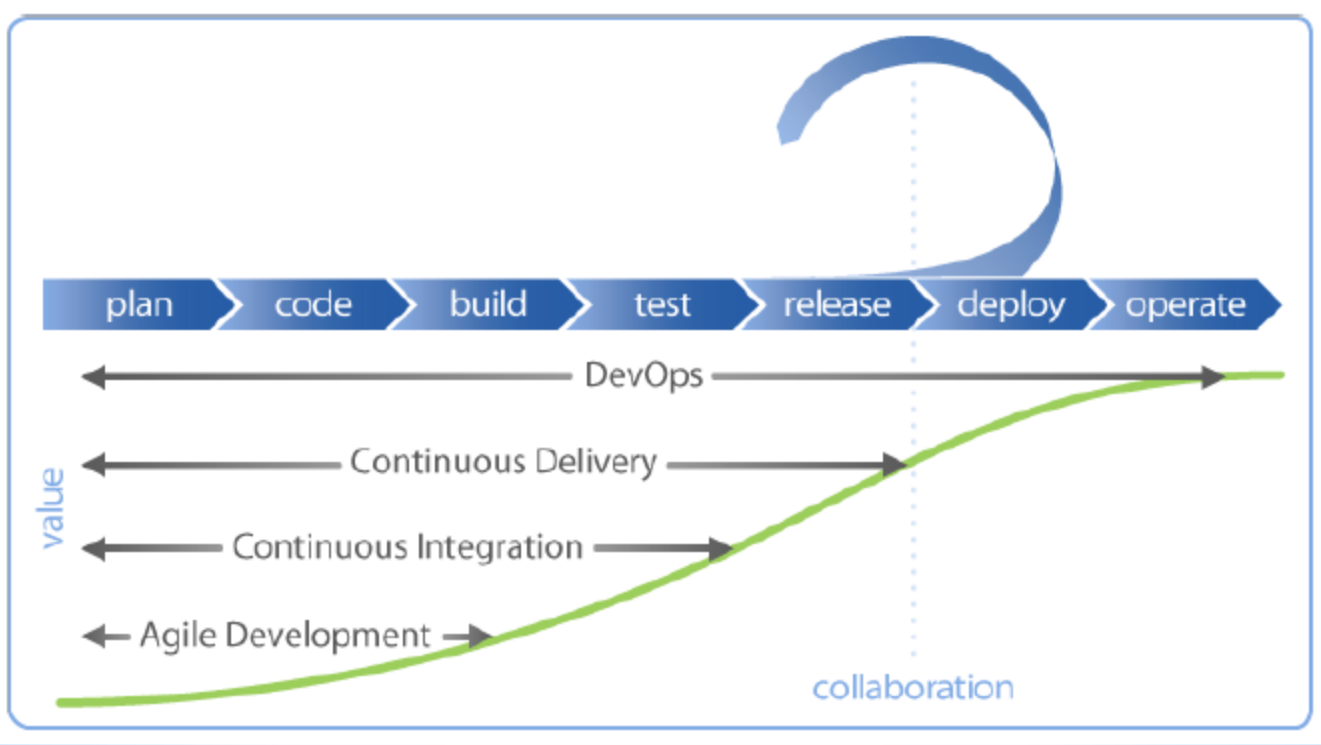
DevOps Philosophy

- DevOps is a philosophy to bring in cultural change aiming to deliver functionalities faster at a higher rate of quality.
- A way to bridge the gap between Developers and Operations team for frequent deployments.
- Human intervention is minimized wherever possible.
- Automation throughout the development life cycle, continuous feedback and process improvement is the key for adopting DevOps.

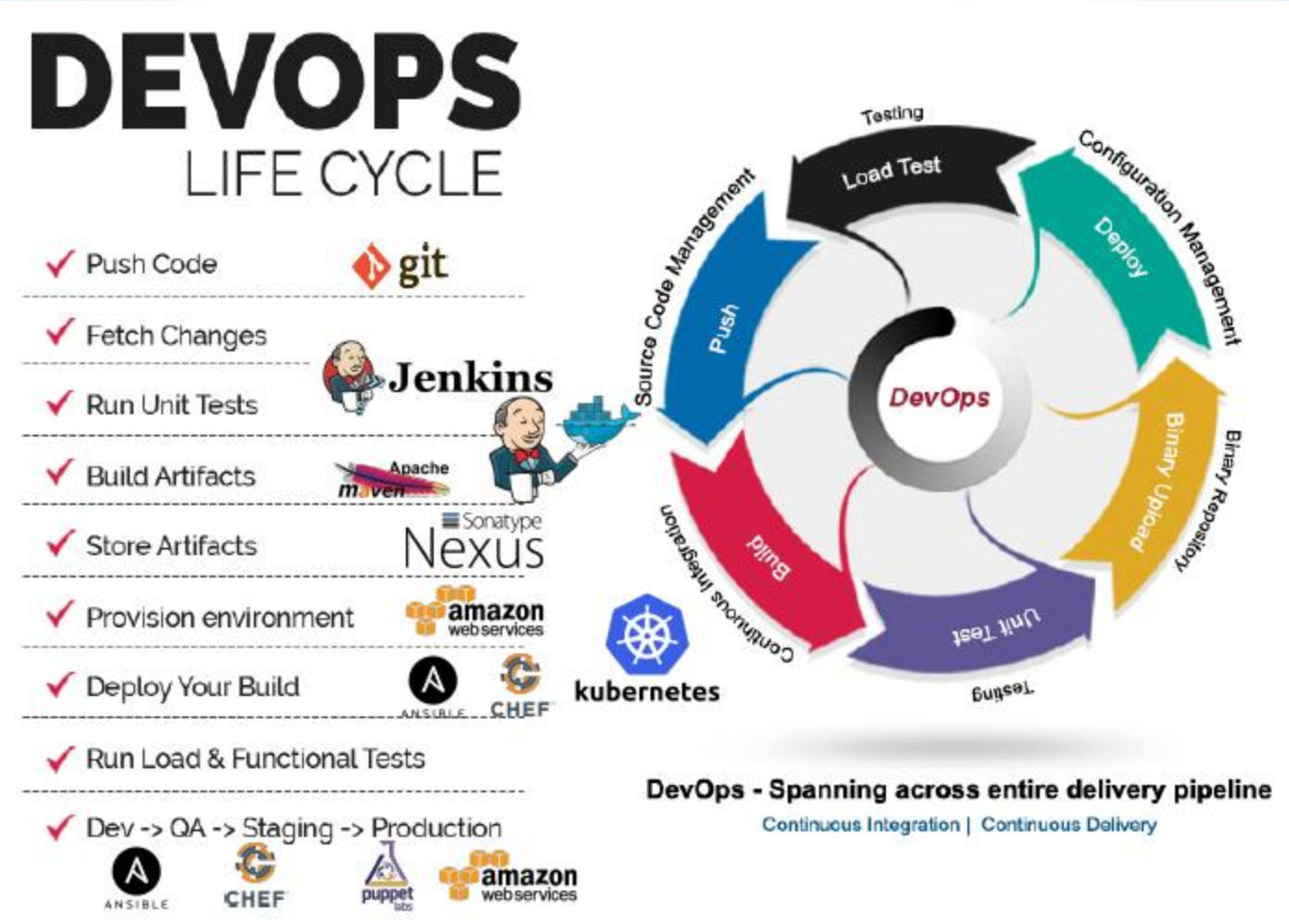
DevOps


Collaboration
Why don't we break the wall?
Communication
Collaboration
Integration
Dev
Ops

DevOps SDLC


plan
code
build
test
release
deploy
operate
DevOps
Continuous Delivery
Continuous Integration
Agile Development
value
collaboration

SDLC Process Overview



DevOps Definition

DevOps is the philosophy of unifying Development and Operations at the culture, practice, and tool levels, to achieve accelerated and more frequent deployment of changes to Production.

Culture=behaviour, teamwork, responsibility/accountability, trust...
Practice=policy, roles, processes/procedures, metrics/reporting...
Tools=shared skills, toolmaking for each other, common technology platforms...

DevOps Skillsets

- Systems Knowledge
- Infrastruture Knowledge
- Cloud Computing skills
- Development SDLC Knowledge
- Build & Release automation / Continous Integration
- Automation of OS tasks
- Orchestration of Continous Delivery process
- Network & Security knowledge
- Containerization (Dockers)

About the Course

- Training from ground zero
- In depth training on every tool
- Document for practicising every tool
- Real Time use cases
- Real Time Project deployment process
- Exercises
- Interview Questions

About Tools

- Linux
- Vagrant & Virtualization
- AWS
- Jankins, GIT & Nexus
- Ansible
- Puppet & Chef
- Docker & Kubernetes
- OS and Cloud security

Step 1 : Systems & Labs

- Linux basics & server management
- Networking
- Virtualization
- Vagrant & VirtualBox

Step 2 : Cloud Computing

- AWS
  - EC2
  - VPC
  - RDS
  - S3
  - Cloudwatch
  - Route53
  - Beanstalk
  - OpsWorks

Step 3 : Automation & Config Management

- Jenkins
  - Build & Release, Build pipelnes, Integration with config tools

- Ansible, Puppet & Chef
  - Complete automation & orchestration
  - Multistack deployment
  - Integration with all the tools/services
  - Troubleshooting

Step 4 : Containerization

- Docker
  - Docker Images & Docker Hub
  - Docker Containers
  - Docker build, Dockerfile

- Docker compose
- Microservices architecture
- Kubernetes

Few more tools

- Git
- Nagios
- Monit
- Nexus
- Sonarqube

Step 5 : Security & High Availability

- Securing OS & Network
- Firewalls & NACL
- Designing & Implementing HA network over cloud
- Offensive security

Project

- Complete Continous delivery web project
  - AWS
  - Git
  - Jenkins
  - Nexus
  - Ansible
  - Tomcat

QUESTIONS