DevOps

-Ms. Pragati Patil

Course Code	Course Name	Teaching Scheme (Contact Hours)		Credits Assigned		
		Theory	Practical	Theory	Practical	Total
ITL503	DevOPs Lab	--	02	--	01	01

Course Code	Course Name	Examination Scheme							
		Theory							
		Internal Assessment			End Sem Exam	Exam Duration (in Hrs)	Term Work	Pract / Oral	Total
		Test1	Test 2	Avg.					
ITL503	DevOPs Lab	--	--	--	--	--	25	25	50

Syllabus

DETAILED SYLLABUS:

Sr. No.	Module	Detailed Content	Hours	LO Mapping
0	Prerequisite	Knowledge of Linux Operating system, installation and configuration of services and command line basics, Basics of Computer Networks and Software Development Life cycle.	**00**	LO1
I	Introduction to Devops	Understanding of the process to be followed during the development of an application, from the inception of an idea to its final deployment. Learn about the concept of DevOps and the practices and principles followed to implement it in any company's software development life cycle. Learn about the phases of Software Lifecycle. Get familiar with the concept of Minimum Viable Product (MVP) & Cross-functional Teams. Understand why DevOps evolved as a prominent culture in most of the modern-day startups to achieve agility in the software development process **Self-Learning Topics: Scrum, Kanban, Agile**	**04**	LO1

II	Version Control	In this module you will learn: • GIT Installation, Version Control, Working with remote repository • GIT Cheat sheet • Create and fork repositories in GitHub • Apply branching, merging and rebasing concepts. • Implement different Git workflow strategies in real-time scenarios • Understand Git operations in IDE **Self-Learning Topics: AWS Codecommit, Mercurial, Subversion, Bitbucket, CVS**	**04**	LO1 & LO2
III	Continuous Integration using Jenkins	In this module, you will know how to perform Continuous Integration using Jenkins by building and automating test cases using Maven / Gradle / Ant. • Introduction to Jenkins (With Architecture) • Introduction to Maven / Gradle / Ant.	**04**	LO1 & LO3
		• Jenkins Management Adding a slave node to Jenkins • Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to deploy an application over the tomcat server **Self-Learning Topics: Travis CI, Bamboo, GitLab, AWS CodePipeline**		

IV	Continuous Testing with Selenium	In this module, you will learn about selenium and how to automate your test cases for testing web elements. You will also get introduced to X-Path, TestNG and integrate Selenium with Jenkins and Maven. • Introduction to Selenium • Installing Selenium • Creating Test Cases in Selenium WebDriver • Run Selenium Tests in Jenkins Using Maven **Self-Learning Topics: Junit, Cucumber**	**04**	LO1 , LO3 & LO4
V	Continuous Deployment: Containerizatio n with Docker	In this module, you will be introduced to the core concepts and technology behind Docker. Learn in detail about container and various operations performed on it. • Introduction to Docker Architecture and Container Life Cycle • Understanding images and containers • Create and Implement docker images using Dockerfile. • Container Lifecycle and working with containers. • To Build, deploy and manage web or software application on Docker Engine. • Publishing image on Docker Hub. **Self-Learning Topics: Docker Compose, Docker Swarm.**	**05**	LO1 & LO5

VI	Continuous Deployment: Configuration Management with Puppet	In this module, you will learn to Build and operate a scalable automation system. • Puppet Architecture • Puppet Master Slave Communication • Puppet Blocks • Installation and Configuring Puppet Master and Agent on Linux machines • Use exported resources and forge modules to set up Puppet modules • Create efficient manifests to streamline your deployments **Self-Learning Topics: Ansible, Saltstack**	**05**	LO1 & LO6

Books

Text books

1. DevOps Bootcamp, Sybgen Learning
2. Karl Matthias & Sean P. Kane, Docker: Up and Running, O'Reilly Publication.
3. Len Bass,Ingo Weber,Liming Zhu,"DevOps, A Software Architects Perspective", AddisonWesley-Pearson Publication.
4. John Ferguson Smart," Jenkins, The Definitive Guide", O'Reilly Publication.
5. Mastering Puppet 5: Optimize enterprise-grade environment performance with Puppet, by Ryan Russell-

Yates Packt Publishing (September 29, 2018)

References:

1. Sanjeev Sharma and Bernie Coyne," DevOps for Dummies", Wiley Publication
2. Httermann, Michael, "DevOps for Developers", Apress Publication.
3. Joakim Verona, "Practical DevOps", Pack publication
4. Puppet 5 Essentials - Third Edition: A fast-paced guide to automating your infrastructure by Martin Alfke Packt Publishing; 3rd Revised edition (September 13, 2017)

Sr.No	Experiment Title
1.	To understand DevOps: Principles, Practices, and DevOps Engineer Role and Responsibilities.
2.	To understand Version Control System / Source Code Management, install git and create a GitHub account.
3.	To Perform various GIT operations on local and Remote repositories using GIT Cheat-Sheet
4.	To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job.
5.	To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to Test and deploy an application over the tomcat server.
6.	To understand Jenkins Master-Slave Architecture and scale your Jenkins standalone implementation by implementing slave nodes.
7.	To Setup and Run Selenium Tests in Jenkins Using Maven.
8.	To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.
9.	To learn Dockerfile instructions, build an image for a sample web application using Dockerfile.
10.	To install and Configure Pull based Software Configuration Management and provisioning tools using Puppet.
11.	To learn Software Configuration Management and provisioning using Puppet Blocks(Manifest, Modules, Classes, Function)
12	To provision a LAMP/MEAN Stack using Puppet Manifest.

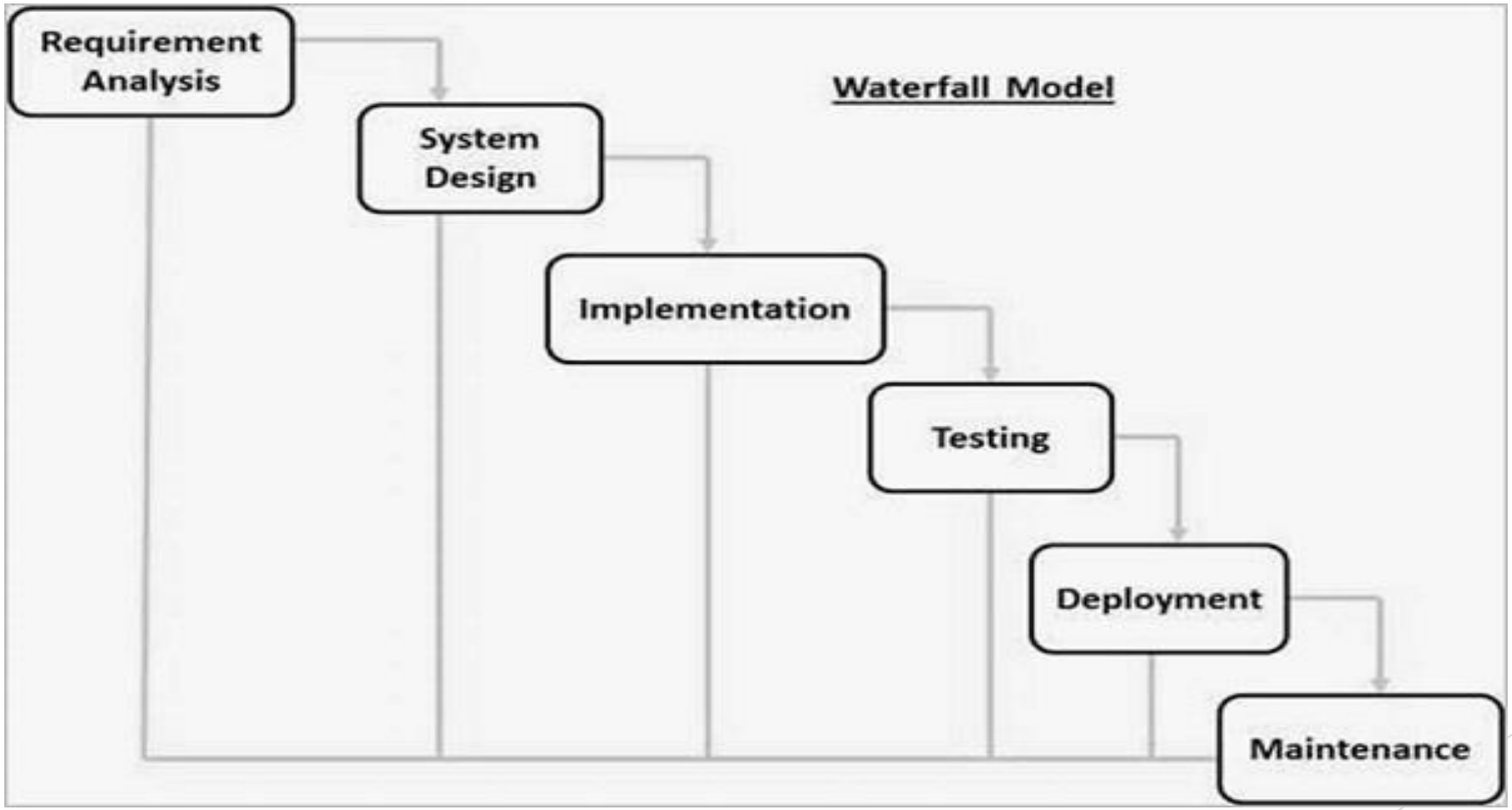
TW

Term Work: Term Work shall consist of at least 12 to 15 practicals based on the above list. Also Term work Journal must include at least 2 assignments, one of which must include a Case study on DevOps Implementation in real world and the other one can be based on the self-learning topics mentioned in syllabus.

Term Work Marks: 25 Marks (Total marks) = 15 Marks (Experiment) + 5 Marks (Assignments) + 5 Marks (Attendance)

Practical & Oral Exam: An Practical & Oral exam will be held based on the above syllabus.

Waterfall model (Iterative Model)



Advantages

Some of the major advantages of the Waterfall Model are as follows −

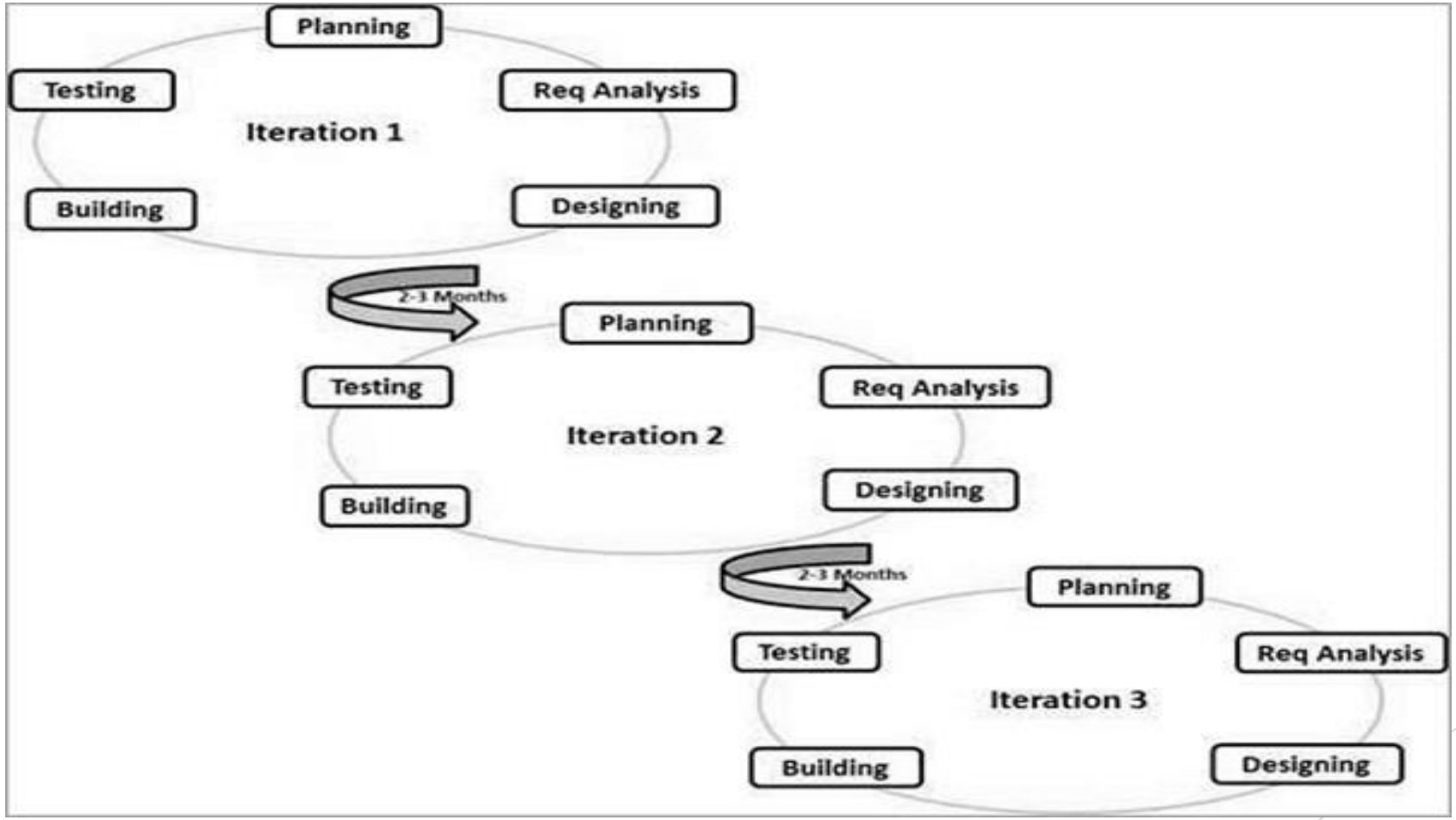
- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

Disadvantages

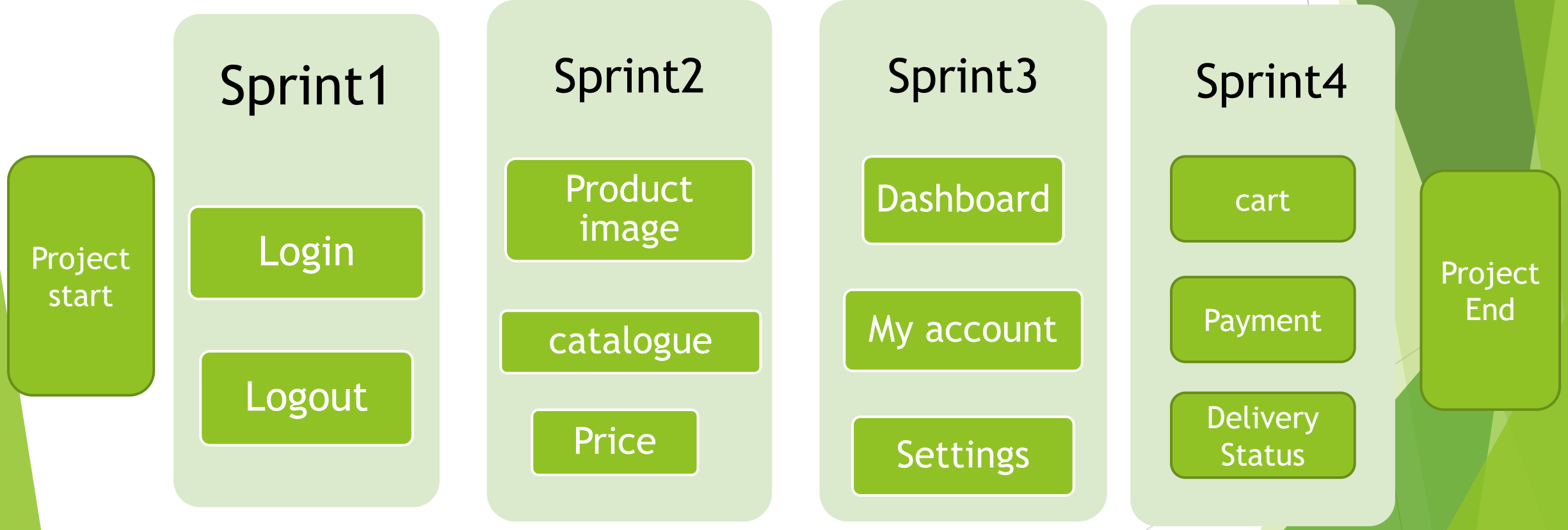
The major disadvantages of the Waterfall Model are as follows −

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.

Agile Methodology



Agile – Build short, build often



Advantages

The advantages of the Agile Model are as follows −

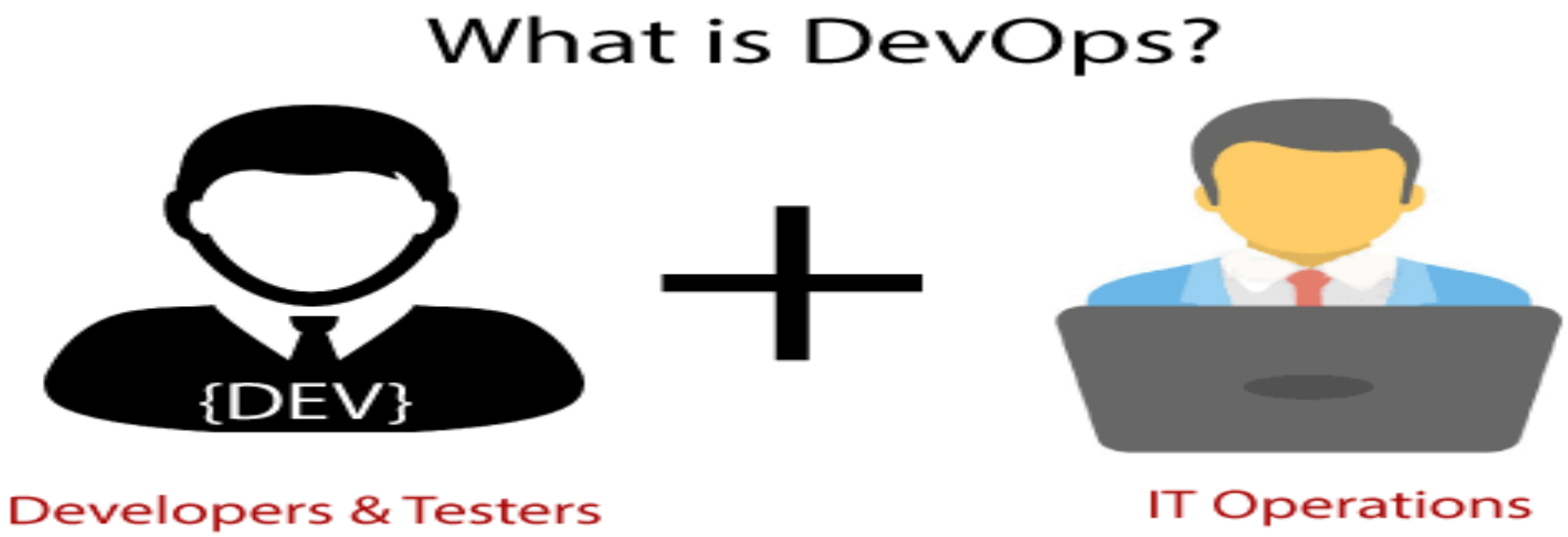
- Is a very realistic approach to software development.
- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Suitable for fixed or changing requirements
- Delivers early partial working solutions.
- Good model for environments that change steadily.
- Minimal rules, documentation easily employed.
- Enables concurrent development and delivery within an overall planned context.
- Easy to manage.
- Gives flexibility to developers.

Disadvantages

- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.
- An overall plan, an agile leader and agile PM practice is a must without which it will not work.
- Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
- There is a very high individual dependency, since there is minimum documentation generated.
- Transfer of technology to new team members may be quite challenging

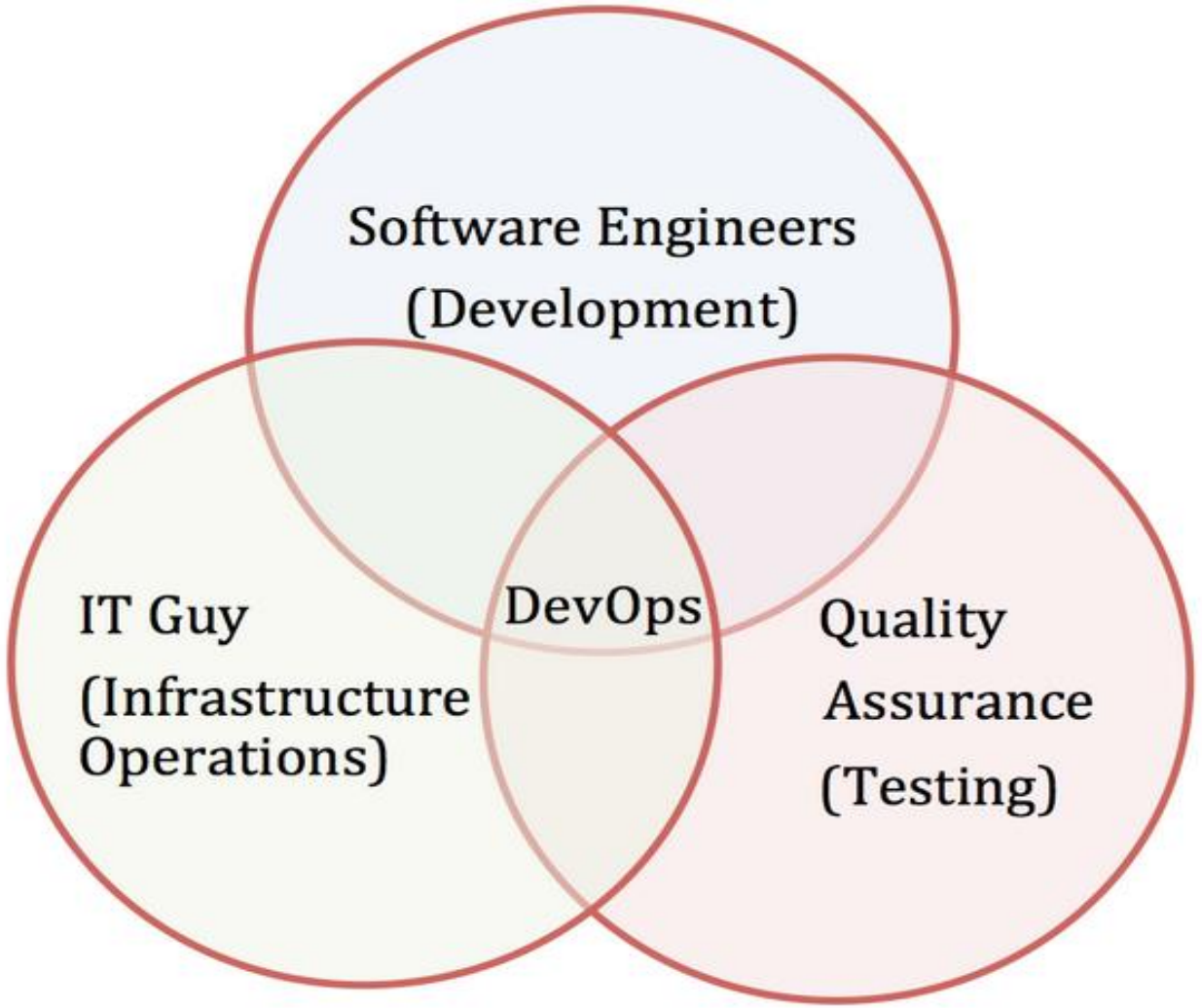
due to lack of documentation.

What is DevOps



What is DevOps?

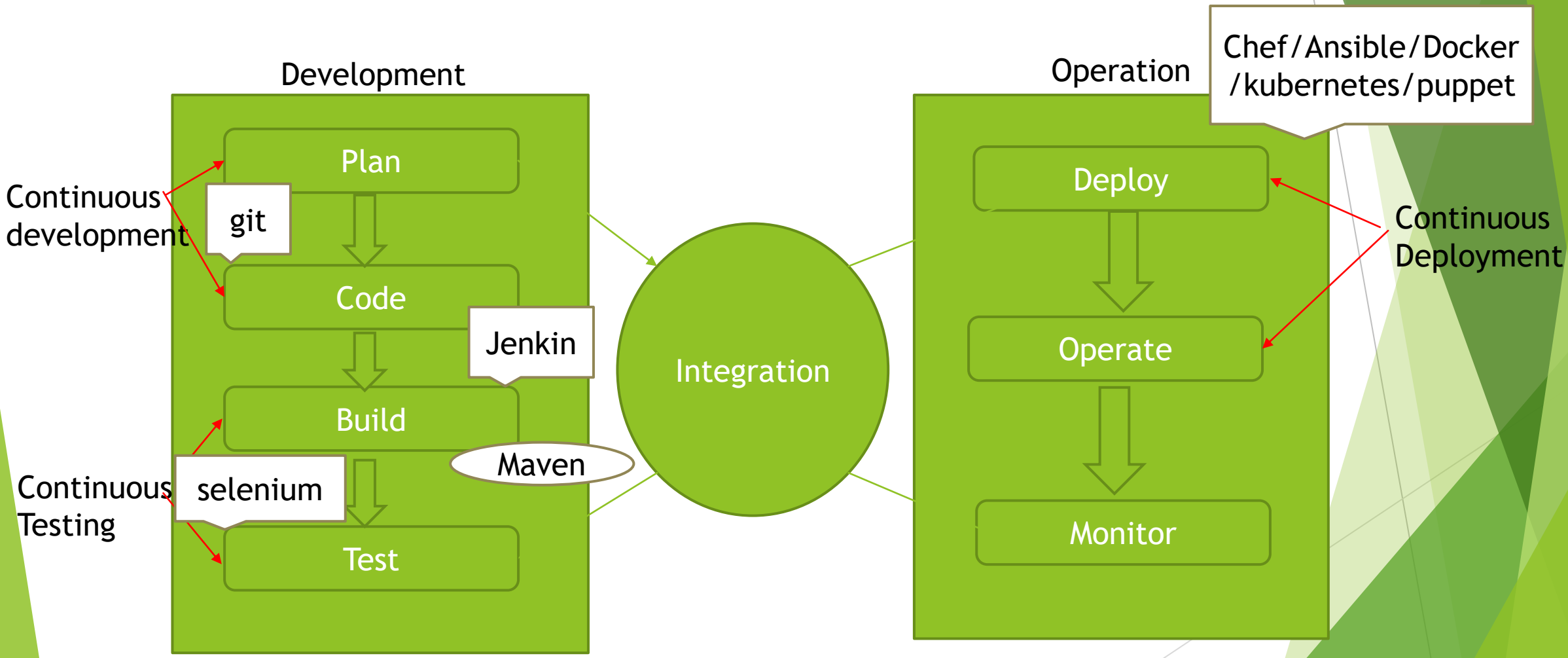
- The DevOps is the combination of two words, one is **Development** and other is **Operations**. It is a culture to promote the development and operation process collectively.
- learn DevOps basics and depth knowledge of various DevOps tools such as **Git, Ansible, Docker, Puppet, Jenkins, Chef** and **Kubernetes**.


Software Engineers
(Development)
IT Guy
(Infrastructure
Operations)
DevOps
Quality
Assurance
(Testing)

Why DevOps ?

We need to understand why we need the DevOps over the other methods.

- The operation and development team worked in complete isolation.
- After the design-build, the testing and deployment are performed respectively. That's why they consumed more time than actual build cycles.
- Without the use of DevOps, the team members are spending a large amount of time on designing, testing, and deploying instead of building the project.
- Manual code deployment leads to human errors in production.
- Coding and operation teams have their separate timelines and are not in synch, causing further delays.


Development
Plan
git
Code
Jenkin
Build
Maven
selenium
Test
Continuous development
Continuous Testing
Integration
Operation
Chef/Ansible/Docker /kubernetes/puppet
Deploy
Operate
Monitor
Continuous Deployment

DevOps Stages

Version control

- Maintain different version of the code

Continuous Integration

- Compile, validate, code review unit testing, Integration testing

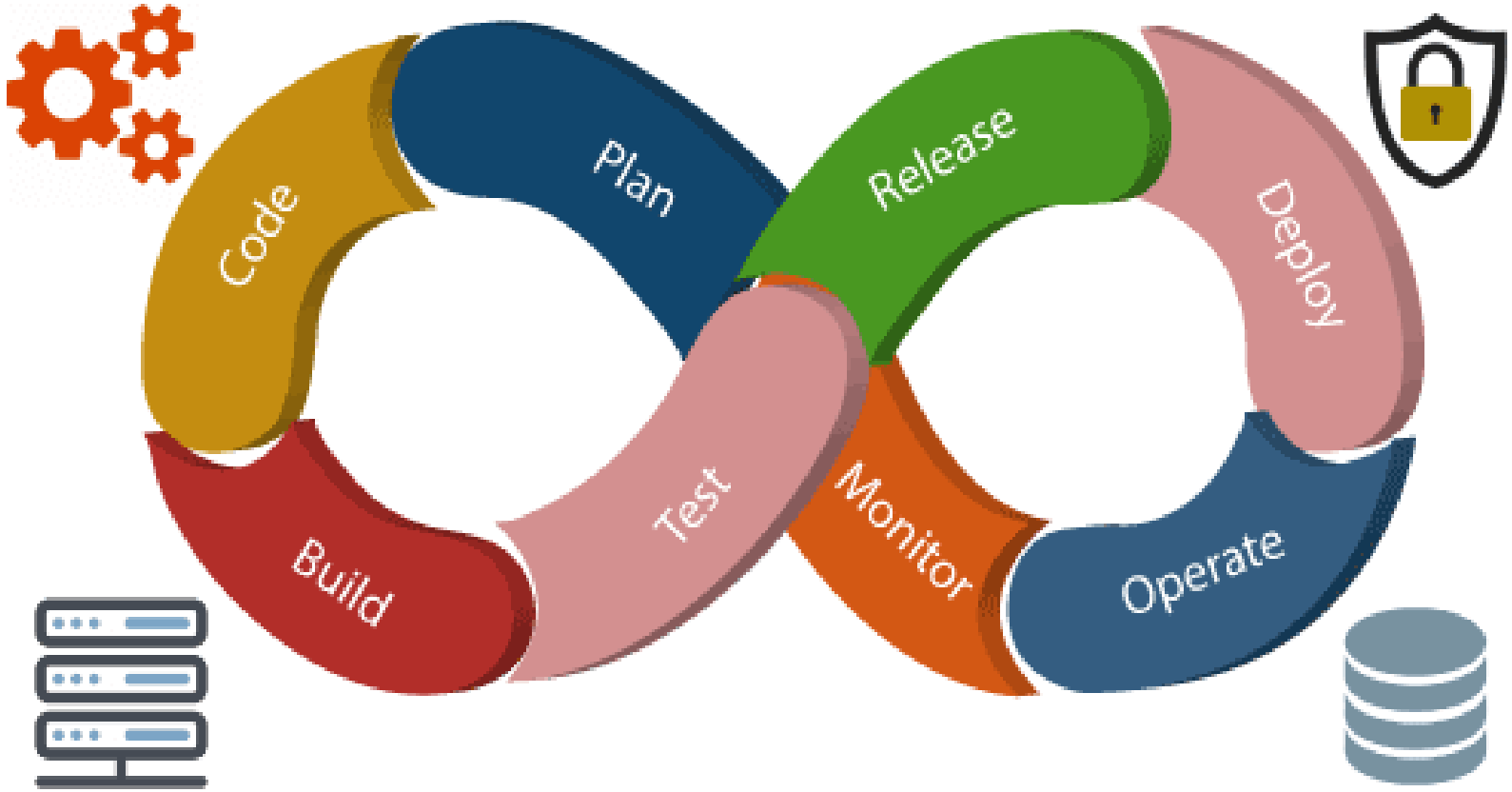
Continuous Delivery

- Deploying the build app to test servers

Continuous Deployment

- Deploying the test app on the production server for release

DevOps Architecture



DevOps Lifecycle

- DevOps defines an agile relationship between operations and Development.
- It is a process that is practiced by the development team and operational engineers together from beginning to the final stage of the product.
- **Continuous Development**

This phase involves the planning and coding of the software. The vision of the project is decided during the planning phase. And the developers begin developing the code for the application. There are no DevOps tools that are required for planning, but there are several tools for maintaining the code.

- **Continuous Integration**

This phase experiences continuous integrations of new code functionalities with the existing source code. Due to continuous development, the updated code seamlessly integrates within the entire system. **Jenkins** is one of the most popular tools for continuous integration. It helps in fetching the updated code and preparing an executable build.

- **Continuous Deployment**

Continuous deployment is guaranteed to benefit your organization once you have a reliable automated testing environment in place. It involves configuring and maintaining consistency in the functional requirement of the app. Popular DevOps tools used for configuration management include **Ansible, Puppet, and Chef**.

- **Continuous Monitoring**

Typical Responsibilities for DevOps engineers include:

 building and setting up new development tools and infrastructure
 understanding the needs of stakeholders and conveying this to developers
 working on ways to automate and improve development and release processes
 testing and examining code written by others and analysing results
 ensuring that systems are safe and secure against cybersecurity threats
 identifying technical problems and developing software updates and “fixes”
 working with software developers and software engineers to ensure that development follows established processes and works as intended
 planning out projects and being involved in project management decisions.

Advantages

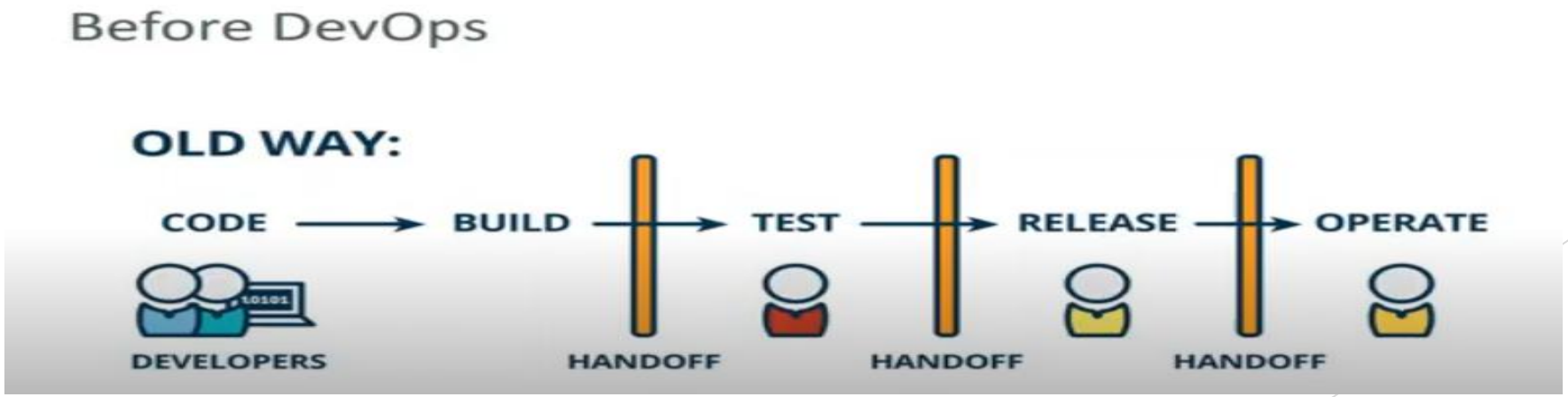
- DevOps is an excellent approach for quick development and deployment of applications.
- It responds faster to the market changes to improve business growth.
- DevOps escalate business profit by decreasing software delivery time and transportation costs.
- DevOps clears the descriptive process, which gives clarity on product development and delivery.
- It improves customer experience and satisfaction.
- DevOps simplifies collaboration and places all tools in the cloud for customers to access.
- DevOps means collective responsibility, which leads to better team engagement and productivity.

Disadvantages

- DevOps professional or expert's developers are less available.
- Developing with DevOps is so expensive.
- Adopting new DevOps technology into the industries is hard to manage in short time.
- Lack of DevOps knowledge can be a problem in the continuous integration of automation projects.

Before devops…. (Recap)

- 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 cycle.
- 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 and operation teams have their separate timeline and are not in sync



Quick Recap

- The term DevOps is a combination of two words i.e. development and operations.
- DevOps is a methodology that allows a single team to manage the entire application development life cycle i.e. development, testing, deployment and operations.
- The objective of DevOps is to shorten the system's development life cycle

  (Reduce the time period).
- DevOps is a software development approach through which superior quality software can be developed quickly and with more reliability.

Improvement using DevOps

- Organizations that have adopted DevOps noticed a 22% improvement in software quality and a 17% improvement in application deployment frequency and achieve a 22% hike in customer satisfaction. 19% of revenue hikes as a result of the successful DevOps implementation.

Thank You