Experiment 01 - Case Study

Roll No.	70
Name	MAYURI SHRIDATTA YERANDE
Class	D15-B
Subject	DevOps Lab
LO Mapped	LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements

Aim: To understand DevOps: Principles, Practices, and DevOps Engineer Role and Responsibilities.

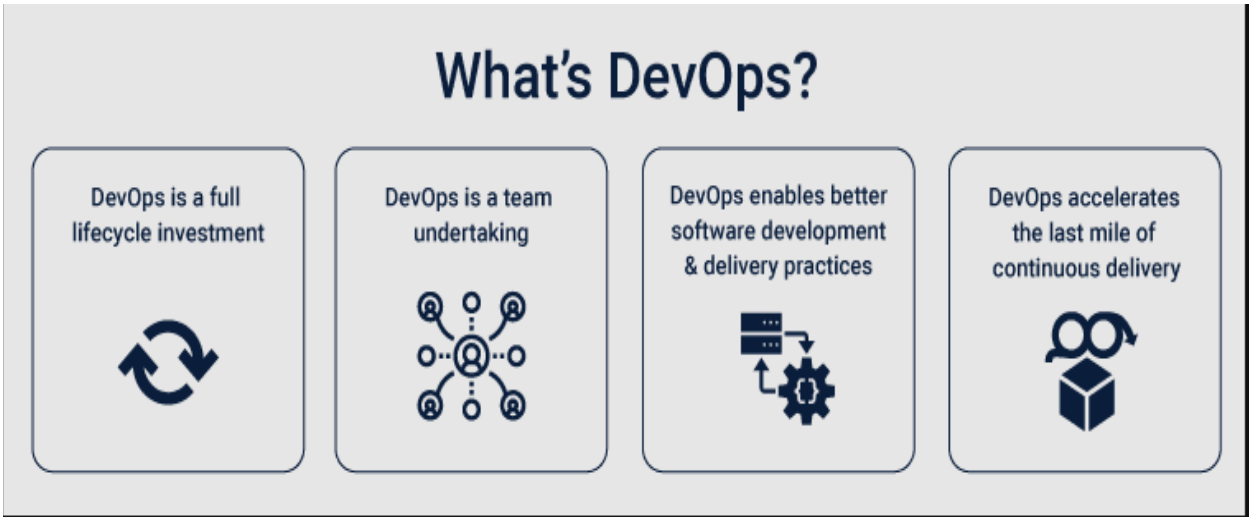
Introduction:

DevOps is, by definition, a field that spans several disciplines. It is a field that is very practical and hands-on, but at the same time, you must understand both the technical background and the nontechnical cultural aspects.
The word "DevOps" is a combination of the words **"development" and "operation"**. This wordplay already serves to give us a hint of the basic nature of the idea behind DevOps. It is a practice where collaboration between different disciplines of software development is encouraged.
The DevOps movement has its roots in Agile software development principles..

Case Study

What is DevOps?

DevOps is the combination of cultural philosophies, practices, and tools that increase an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.



Best Practices for Effective DevOps?

While DevOps still means different things to different people, there has emerged a core of best practices that should be incorporated by companies looking at adopting DevOps.

- Active Stakeholder Participation

This is the fundamental guiding principle of DevOps. DevOps can succeed only if both the developers and the operations and support staff are truly committed to collaborating and using an integrated approach to achieve goals.

- Automated Testing

Automated regression testing is something agile teams adopt very often as it helps to fix problems right away and ship higher quality code. This works well in DevOps too as a dire need of operations staff is that the code shipped should meet a certain quality standard.

- Integrated Configuration Management

In a DevOps environment, configuration management applies not only to the current solution being worked on but also to the configuration issues between the solution and the rest of the organization infrastructure. Integrated Configuration management helps operations teams see the potential impact of a new release more clearly which helps in making better decisions regarding when the release should be made.

- Integrated Change Management

With integrated change management, operations and development teams work together to understand how using different technologies will impact the organization as a whole and then work toward managing that.

- Continuous Integration

With continuous integration, the code is tested and analyzed whenever updated code is checked into the version control system. This provides immediate feedback on code defects which allows developers to build a high-quality solution with little risk.

- Integrated Deployment Planning

A DevOps approach means operations engineers will be closely involved with the developers when it comes to planning the deployment of products as per an organisational deployment schedule.

- Continuous Deployment

With continuous deployment, when integration is successful in one sandbox it is automatically promoted to the next sandbox and integration begins there. This continues until it reaches the point where it requires human verification. This usually occurs at the point of transition from dev to operations.

- Production Support

With DevOps, not only do developers work on new releases, but they also work on addressing critical problems with a solution that is already in production. Although they are the third and last team to get involved in solving production issues, it is a fairly common occurrence and gives them insights on production problems that help them design better solutions in the first place.

- Application Monitoring

This refers to the practice of monitoring and logging solutions real-time once they are in production. This gives us performance metrics that improve the reliability of the solution and prevent failures.

What are the DevOps Tools?

In order to implement DevOps best practices described above, certain tools have been developed to automate and facilitate different DevOps processes. While the right tools play a key role in effective DevOps implementation, simply using the tools does not mean DevOps adoption. Tools are only relevant when they are used as the last stage- after the organization has already adopted the philosophy of DevOps and there is a commitment to execute its best practices.
Here are just a few examples.

Release Tools

- Jenkins
- Travis
- TeamCity

Configuration Management Tools

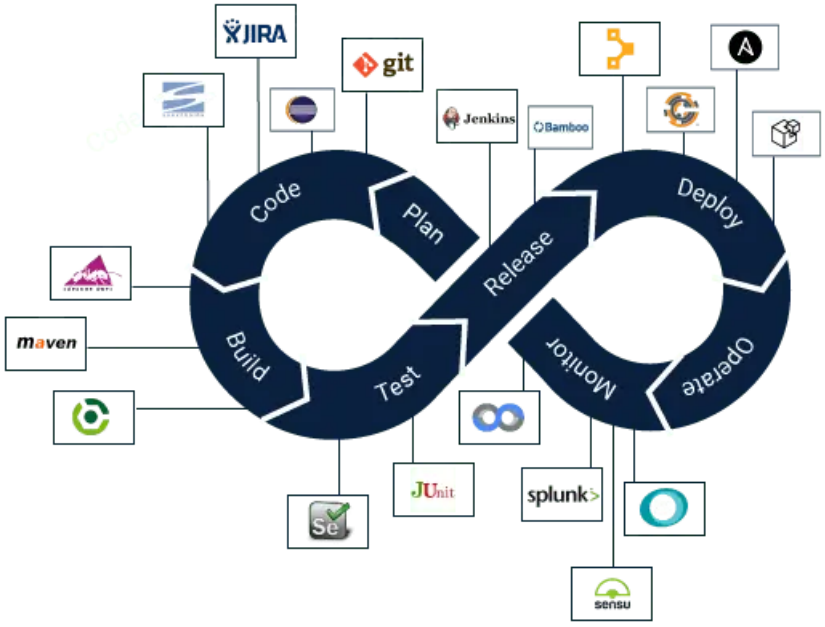
- Puppet
- Chef
- Ansible
- Cfengine

Monitoring, Virtualization and Containerization Tools

- AWS
- OpenStack
- Vagrant
- Docker
- New Relic

Coding Tools

- Jira
- Git
- Eclipse



How is DevOps beneficial?

DevOps often provides cloud-based services to easily associate the tools. It provides separate operations, which means each developer focuses on a single tool without understanding the entire toolchain. DevOps makes better collaboration and faster turnaround.

- Faster product release and faster time to market
- Higher productivity
- Process efficiency
- Shortened production cycles
- Better operational support
- Engaged and motivated staff
- Better Customer experience management
- Clear product vision within the team
- Increased deployment success rates
- Increased product quality
- Better team efficiency
- Improved flexibility and support
- Reduced chance of product failure
- Cross-skilling and self-improvement

The Future of DevOps

The future of DevOps will likely bring changes in tooling and organizational strategies, but its core mission will remain the same

- Automation Will Play a Major Role

Automation will continue to play a major role in DevOps transformation, and artificial intelligence for IT operations—AIOps—will help organizations achieve their DevOps goals. The core elements of AIOps—machine learning, performance baselining, anomaly detection, automated root cause analysis (RCA) and predictive insights—work together to accelerate

routine operational tasks. This emerging technology, which can transform how IT operations teams manage alerts and resolve issues, will be a crucial component of the future of DevOps.

- AIOps Will Make Service Uptime Easier to Achieve

In addition to using data science and computational techniques to automate mundane tasks, AIOps also ingests metrics and uses inference models to pull actionable insights from data, notes data science architect Jiayi Hoffman. AIOps' automation capabilities can make service uptime much easier to achieve, from monitoring to alerting to remediation. And AIOps is a boon for DevOps teams, who can use AIOps tools for real-time analysis of event streams, proactive detection to reduce downtime, improved collaboration, faster deployments, and more.

- Will Sharpen Focus on Cloud Optimization

The future of DevOps will also bring a greater focus on optimizing the use of cloud technologies. The centralized nature of the cloud provides DevOps automation with a standard platform for testing, deployment, and production notes Deloitte Consulting analyst David Linthicum.

DevOps Engineer Roles and Responsibilities:

The role of a devOps engineer combines aspects of a technical role and an IT operations role. While engineers are likely to be involved with coding, and a knowledge of coding languages and principles is required to find problems and build solutions, this is not the primary role of a devOps engineer. More senior engineers are likely to be more involved in the project management and planning side of development.

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 analyzing 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.

Graduates may start out in a ‘graduate devOps engineer’ or ‘associate devOps engineer’ role, before progressing to ‘senior devOps engineer’ and ‘principal devOps engineer’ roles with experience.

Useful skills for devOps engineers include:

- excellent team working and communication skills
- knowledge of programming languages
- strong problem-solving skills
- good attention to detail
- excellent organizational and time management skills, and the ability to work on multiple projects at the same time
- awareness of devOps and Agile principle

Conclusion: DevOps is helping businesses in a tremendous way. It's bridging the gap between developers' need for change and operations' resistance to change and thus creates a smooth path for Continuous Development and Continuous Integration.