PARSHVANATH CHARITABLE TRUST'S

A. P. SHAH INSTITUTE OF TECHNOLOGY

Department of Information Technology

(NBA Accredited)

Department of Information Technology

Academic Year: 2025-26	**Name of Student:Siddhi Tangsali**
Semester: V	**Student ID:23104069**
Class / Branch: TEIT/Div C	**Date of Performance:9/7/25**
Subject: DevOps Lab	**Date of Submission:9/7/25**
Name of Instructor: Prof. Sujata Oak	

Experiment No. 1

Aim: To understand the principles of DevOps along with its role and responsibilities.

What is DevOps?

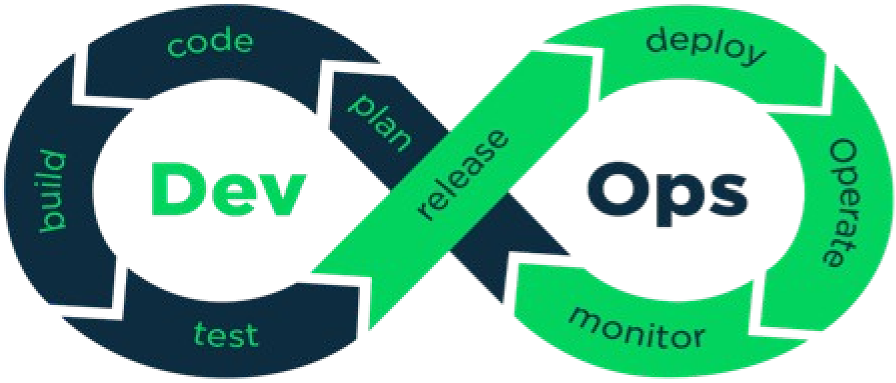
The word DevOps is a combination of two words Development and Operations. The development team is responsible for developing, designing, and building the application. The operation team deals with the deployment and testing of the application. If there are problems with the application, the operation team also provides feedback to the development team.

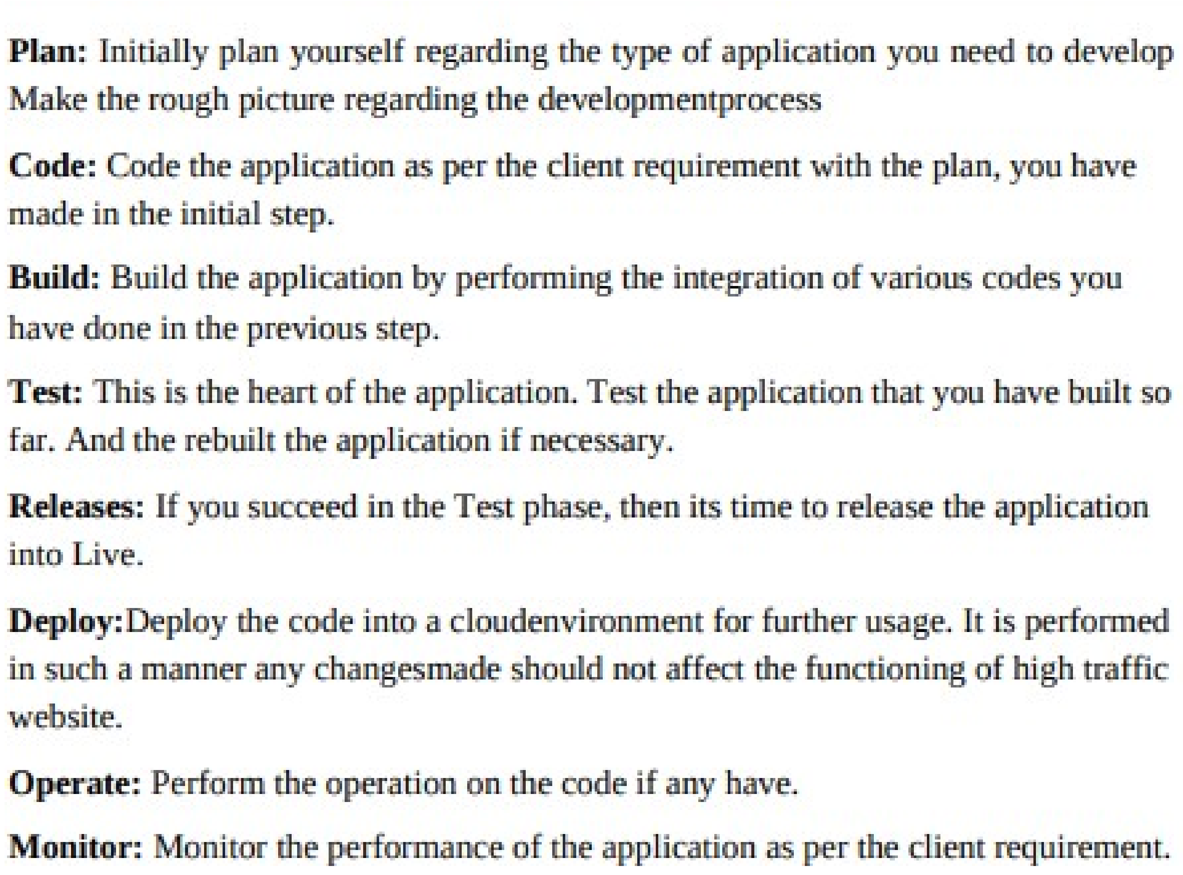
Principal Advantages:

- Quicker updates and releases of software
- Enhanced cooperation among groups
- Early problem identification through ongoing observation

DevOps is used by companies like Netflix, Amazon, and Google to ship code hundreds of times a day, guaranteeing uptime and quick innovation

DevOps Lifecycle: An Overview of the Phases




Plan: Initially plan yourself regarding the type of application you need to develop Make the rough picture regarding the developmentprocess

Code: Code the application as per the client requirement with the plan, you have made in the initial step.

Build: Build the application by performing the integration of various codes you have done in the previous step.

Test: This is the heart of the application. Test the application that you have built so far. And the rebuilt the application if necessary.

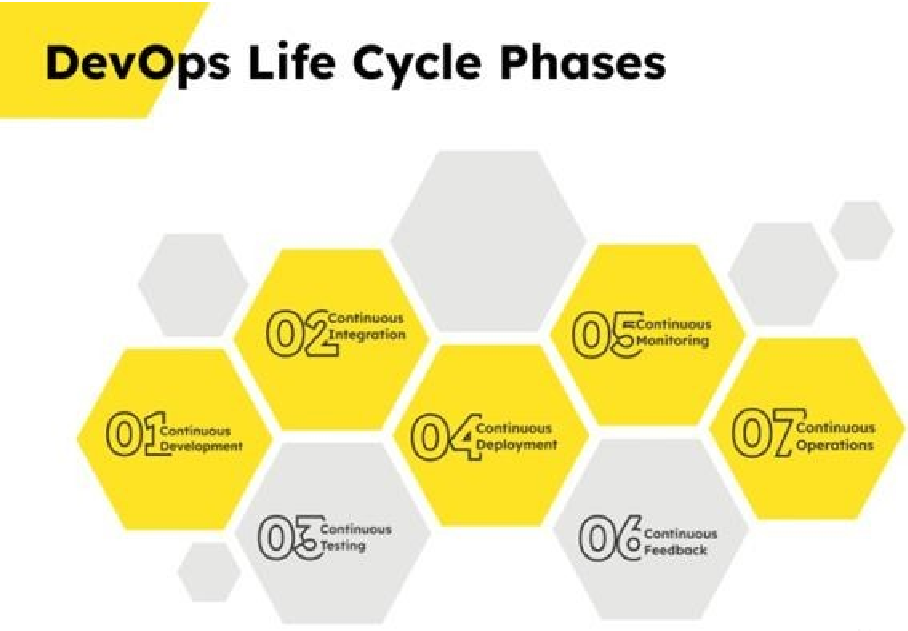
Releases: If you succeed in the Test phase, then its time to release the application into Live.

Deploy: Deploy the code into a cloudenvironment for further usage. It is performed in such a manner any changesmade should not affect the functioning of high traffic website.

Operate: Perform the operation on the code if any have.

Monitor: Monitor the performance of the application as per the client requirement.

Phases of DevOps Lifecycle



Continuous Development

The continuous development phase is a phase in which the program is ‘planned’ and ‘coded .’ The development phase includes all aspects of coding, including team members’ authoring, testing, integrating code, and converting that code into build artifacts deployed in various environments. DevOps teams aspire for rapid innovation while maintaining quality, stability, and productivity. They accomplish this by utilizing highly productive tools, automating repeated and manual tasks, and iterating in smaller increments with the help of automated testing and continuous integration.

The most used tools in this phase are Git, SVN, Mercurial, CVS, and JIRA.

Continuous Testing

The testing step of the DevOps lifecycle examines the created code for errors and faults. This is where qualitative analysis comes in handy for ensuring that the generated software is usable. The Quality Assurance (QA) process must be completed successfully to determine whether the software meets the client’s requirements. This assures that the generated program has no defects in terms of functioning

Continuous Integration

This is the most important step in the DevOps lifecycle. Continuous Integration is a software development approach that requires developers to submit the source code changes more often. It is done once a day or once a week. Code building includes compilation and code review, unit testing, integration testing, and packaging.

Continuous Deployment

The continuous Deployment(CD) stage ensures the products are deployed smoothly without impacting the application’s performance. This technique eliminates the need for planned releases and speeds up the feedback system, allowing developers to respond to concerns faster and more accurately. In addition, through configuration management, containerization solutions help in continuous deployment.

Continuous Monitoring

It is one of the most important stages of the DevOps lifecycle. Monitoring a software product’s performance is critical for determining the overall effectiveness of the product’s output. Through constant monitoring, developers can find general patterns and parts in the program that require more attention. Continuous monitoring is an operational phase aiming to improve the software application’s overall efficiency.

Continuous Operations

All DevOps operations are built on continuity, with complete automation of the release process, allowing the organization to continue reducing overall time to market.

Continuous Feedback

The application development process is constantly improved by analyzing the results of software operations. This is accomplished by establishing a critical phase of constant feedback between operations and developing the next version of the current software application.

DevOps Tools with its functionalities

1. Git (Version Control – Continuous Development)

Git is a distributed version control system that allows multiple developers to work on code simultaneously. It tracks changes, supports branching and merging, and integrates with platforms like GitHub and GitLab for collaboration and code review.

---

2. Jenkins (Automation – Continuous Integration & Delivery)

Jenkins is an open-source automation server used to build, test, and deploy applications. It supports plugins to integrate with almost any DevOps tool and automates the entire CI/CD pipeline, ensuring frequent and reliable software updates.

3. Docker (Containerization – Continuous Deployment)

Docker packages applications and their dependencies into lightweight containers. This ensures consistency across different environments (development, testing, production), making deployment faster and more reliable.

4. Kubernetes (Container Orchestration – Continuous Deployment)

Kubernetes automates the deployment, scaling, and management of containerized applications. It ensures high availability and load balancing by distributing containers across clusters.

5. Selenium (Testing – Continuous Testing)

Selenium is an automation testing tool for web applications. It supports multiple browsers and programming languages, enabling automated test scripts for continuous testing throughout the development lifecycle.

6. Prometheus & Grafana (Monitoring – Continuous Monitoring)

Prometheus is an open-source monitoring tool that collects metrics from applications and systems. Grafana works with Prometheus to visualize these metrics in real-time dashboards, helping teams detect issues quickly.

7. Ansible (Configuration Management – Continuous Deployment)

Ansible is an automation tool used for configuration management, application deployment, and task automation. It uses simple YAML files and requires no agents, making it easy to maintain infrastructure as code.

Conclusion : Hence we implemented the experiment successfully.