Jerryn Joeboy
501837
IT sem 8

Experiment 1

Aim: Introduction to DevOps

Lab Outcome (LO1): Remember the importance of DevOps tools used in software development life cycle

Theory:

What is DevOps?
DevOps is a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.

How does DevOps work?
The DevOps process flow is all about agility and automation. Each phase in the DevOps lifecycle focuses on closing the loop between development and operations and driving production through continuous development, integration, testing, monitoring and feedback, delivery, and deployment.

What is the difference between Development and Operations?
The Development team works on code which is then sent to the testing team for validation against requirements. Operation team comes in toward the end of the process, where handover of release is given.

What are the stages in a DevOps Lifecycle?
It consists of various stages such as continuous development, continuous integration, continuous testing, continuous deployment, and continuous monitoring.

- ***Continuous Development –***

  This is the phase that involves ‘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

- ***Continuous Testing –***

  This is the stage where the developed software is continuously tested for bugs. For Continuous testing, automation testing tools like Selenium, TestNG, JUnit, etc are used. These tools allow QAs to test multiple code-bases thoroughly in parallel to ensure that there are no flaws in the functionality. In this phase, Docker Containers can be used for simulating the test environment

- ***Continuous Integration –***
  This stage is the heart of the entire DevOps life cycle. It is a software development practice in which the developers are required to commit changes to the source code more frequently. This may be on a daily or a weekly basis. Every commit is then built and this allows early detection of problems if they are present. Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging

- ***Continuous Deployment –***
  This is the stage where the code is deployed to the production servers. It is also important to ensure that the code is correctly deployed on all the servers

- ***Continuous Monitoring –***
  This is a very crucial stage of the DevOps lifecycle where you continuously monitor the performance of your application. Here vital information about the use of the software is recorded. This information is processed to recognize the proper functionality of the application. The system errors such as low memory, server not reachable, etc are resolved in this phase

What is the Software Development Lifecycle?

SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use. Its stages are as follows:

- **Identify the Current Problems**
  This stage of the SDLC means getting input from all stakeholders, including customers, salespeople, industry experts, and programmers. Learn the strengths and weaknesses of the current system with improvement as the goal

- **Plan**
  In this stage of the SDLC, the team determines the cost and resources required for implementing the analyzed requirements. It also details the risks involved and provides sub-plans for softening those risks

- **Design**
  This phase of the SDLC starts by turning the software specifications into a design plan called the Design Specification. All stakeholders then review this plan and offer feedback

and suggestions. It’s crucial to have a plan for collecting and incorporating stakeholder input into this document. Failure at this stage will almost certainly result in cost overruns at best and the total collapse of the project at worst

- **Build**
  At this stage, the actual development starts. It’s important that every developer sticks to the agreed blueprint. Also, make sure you have proper guidelines in place about the code style and practices

- **Test**
  In this stage, we test for defects and deficiencies. We fix those issues until the product meets the original specifications. In short, we want to verify if the code meets the defined requirements

- **Software Deployment**
  At this stage, the goal is to deploy the software to the production environment so users can start using the product. However, many organizations choose to move the product through different deployment environments such as a testing or staging environment

What are the tools used in DevOps?

DevOps tools help simplify and accelerate testing, configuration, deployment, and other software-related tasks required to implement DevOps processes. A few tools are as follows:

- **Git:** It is a widely used DevOps tool across the software industry. It’s a distributed SCM (source code management) tool known for its free open source collaboration and planning that is extensively used for tracking the progress of development work by remote teams and open source contributors. It supports most of the version control features including check-in, commits, branches, merging, labels, push and pull to/from GitHub and more
- **Jenkins:** It is an open source solution for continuous integration that orchestrates and automates sequence of actions enabling developers to reliably build, test, and deploy their software. Jenkins is used by DevOps teams for accelerating production rollouts by benefiting from its power of automation. With a large pool of plug-ins available in the Jenkins ecosystem, its capabilities can be expanded to various stages in the DevOps lifecycle
- **Puppet:** It is an open source configuration management tool to automate inspecting, delivering and managing the software across the complete development lifecycle with platform independence. It automates infrastructure management to deliver software quickly and securely
- **Chef:** It is another configuration management tool used to automate and simplify deployment, repair, update and management of application infrastructures. Avoiding

manual scrip-based changes, Chef provides a seamless orchestration engine to allow DevOps engineers to ensure continuous delivery of code releases. By treating infrastructure as code, Chef uses pre-built and customizable policies to automatically effect changes to the deployment infrastructure

- **eG Enterprise:** Monitoring is a critical part of software development and deployment. Through all the stages of DevOps, from code build to test and commit to deploy DevOps teams need to understand the impact that their code will have on pre-production and production environments. eG Enterprise is a continuous monitoring tool allows tracking application performance in the context of code changes to understand how they impact performance

Conclusion: Therefore, we have observed an overview of DevOps.