**Experiment No-1**

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

**Theory**: DevOps is a set of practices, tools, and a cultural philosophy that automates and integrates the processes between software development and IT teams. It emphasizes team empowerment, cross-team communication and collaboration, and technology automation. It is also a methodology in the software development and IT industry. Used as a set of practices and tools, it integrates and automates the work of software development and IT operations as a means for improving and shortening the systems development life cycle.

Under a DevOps model, development and operations teams are no longer “soloed.” Sometimes, these two teams are merged into a single team where the engineers work across the entire application lifecycle, from development and test to deployment to operations, and develop a range of skills not limited to a single function.

In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle. When security is the focus of everyone on a DevOps team, this is sometimes referred to as DevSecOps.

These teams use practices to automate processes that historically have been manual and slow. They use a technology stack and tooling which help them operate and evolve applications quickly and reliably. These tools also help engineers independently accomplish tasks (for example, deploying code or provisioning infrastructure) that normally would have required help from other teams, and this further increases a team’s velocity.

**Principles of DevOps:-**

* **Collaboration: -** The key premise behind DevOps is collaboration. Development and operations teams coalesce into a functional team that communicates, shares feedback, and collaborates throughout the entire development and deployment cycle. Often, this means development and operations teams merge into a single team that works across the entire application lifecycle.
* **Automation**: - An essential practice of DevOps is to automate as much of the software development lifecycle as possible. This gives developers more time to write code and develop new features. Automation is a key element of a [CI/CD pipeline](https://www.atlassian.com/continuous-delivery/principles/pipeline) and helps to reduce human errors and increase team productivity. With automated processes, teams achieve continuous improvement with short iteration times, which allows them to quickly respond to customer feedback.
* **Continuous Improvement: -** Continuous improvement was established as a staple of, as well as lean manufacturing and [Improvement Kata](https://www.atlassian.com/agile/agile-at-scale/using-improvement-kata-to-support-lean). It’s the practice of focusing on experimentation, minimizing waste, and optimizing for speed, cost, and ease of delivery. Continuous improvement is also tied to [continuous delivery](https://www.atlassian.com/continuous-delivery), allowing DevOps teams to continuously push updates that improve the efficiency of software systems. The constant pipeline of new releases means teams consistently push code changes that eliminate waste, improve development efficiency, and bring more customer value.
* **Customer-centric action: -** DevOps teams use short feedback loops with customers and end users to develop products and services centered on user needs. DevOps practices enable rapid collection and response to user feedback through use of real-time live monitoring and rapid deployment. Teams get immediate visibility into how live users interact with a software system and use that insight to develop further improvements.
* **Create with the end in mind: - -**This principle involves understanding the needs of customers and creating products or services that solve real problems. Teams shouldn’t ‘build in a bubble’, or create software based on assumptions about how consumers will use the software. Rather, DevOps teams should have a holistic understanding of the product, from creation to implementation

**Practices of DevOps:-**

**1. Agile project management: -** Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches. Agile teams focus on delivering work in smaller increments, instead of waiting for a single massive release date. Requirements, plans, and results are evaluated continuously, allowing teams to respond to feedback and pivot as necessary.

**2. ​​​​​​​Shift left with CI/CD: -** When teams “shift left”, they bring testing into their code development processes early. Instead of sending multiple changes to a separate test or QA team, a variety of tests are performed throughout the coding process so that developers can fix bugs or improve code quality while they work on the relevant section of the codebase. The practice of continuous integration and continuous delivery (CI/CD), and deployment underpins the ability to shift left. Read more about CI/CD.

**​​​​​​​3. Build with the right tools: -** A DevOps tool chain requires the right tools for each phase of the DevOps lifecycle, with key capabilities to improve software quality and speed of delivery. Read more about how to choose DevOps tools and review functionality for each phase of the DevOps lifecycle.

**4. ​​​​​​​Implement automation: -** Continuous integration and delivery allows developers to merge code regularly into the main repository. Instead of manually checking code, CI/CD automates this process, from batching in a specified window to frequent commits. In addition to CI/CD, automated testing is essential to successful DevOps practices. Automated tests might include end-to-end testing, unit tests, integration tests, and performance tests. Read more about incorporating automation into your software development processes. Read more about automation.

5. ​​​​​​​**Change the culture: -** DevOps requires collaboration, transparency, trust, and empathy. If your organization is one of the rare ones where these qualities are already established, it should be fairly easy for your teams to adopt DevOps practices. If not, some effort will be required to develop these qualities. The most common organizational structures are soloed, meaning different teams have separate areas of ownership and responsibility and there is minimal cross-team communication or collaboration. For DevOps to succeed, these barriers must be eliminated by adopting the "you build it, you run it" practice. This doesn’t mean there aren’t people or teams who specialize, only that the lines of communication and collaboration between teams are open and used. Read more on building your team culture.

**Roles and Responsibilities of DevOps Engineer:-** A DevOps engineer is an IT generalist who should have a wide-ranging knowledge of both development and operations, including coding, infrastructure management, system administration, and DevOps tool chains. DevOps engineers should also possess interpersonal skills since they work across company silos to create a more collaborative environment.

DevOps engineers need to have a strong understanding of common system architecture, provisioning, and administration, but must also have experience with the traditional developer toolset and practices such as using source control, giving and receiving code reviews, writing unit tests, and familiarity with agile principles.

1. **Project planning**: Understanding customer requirements and project KPIs.
2. **Product development**: Implementing various development, testing, automation tools, and IT infrastructure.
3. **Performance management**: Planning the team structure, activities, and involvement in project management activities.
4. **Maintenance and troubleshooting**: Managing stakeholders and external interfaces, setting up tools and required infrastructure, defining and setting development, test, release, update, and support processes for DevOps operation, reviewing, verifying, and validating the software code developed in the project, troubleshooting techniques and fixing the code bugs, monitoring the processes during the entire lifecycle for its adherence and updating or creating new processes for improvement and minimizing the wastage, encouraging and building automated processes wherever possible, identifying and deploying cyber security measures by continuously performing vulnerability assessment and risk management, incidence management and root cause analysis, coordination and communication within the team and with customers.
5. **Product deployment**: Selecting and deploying appropriate CI/CD tools, striving for continuous improvement and building continuous integration, continuous development, and constant deployment pipeline (CI/CD Pipeline).
6. **Mentoring and guiding**: Mentoring and guiding the team members.
7. **Reporting**: Monitoring and measuring customer experience and KPIs, managing periodic reporting on the progress to the management and the customer.

**Conclusion**: Thus we have studied and learned about DevOps. Its principles, its practices and about the roles and responsibilities of the DevOps Engineer.