HariOm Shukla **EXPERIMENT NO.-1** **121A3052**

AIM: To understand DevOps: Principles, practices, DevOps, Engineer role and responsibilities.

THEORY:

A) Introduction to DevOps

DevOps is basically a combination of two words- Development and Operations. DevOps is a culture that implements the technology in order to promote collaboration between the developer team and the operations team to deploy code to production faster in an automated and repeatable way. DevOps practices enable software development (dev) and operations (ops) teams to accelerate delivery through automation, collaboration, fast feedback, and iterative improvement.

- What is DevOps?
  DevOps combines development and operations to increase the efficiency, speed, and security of software development and delivery compared to traditional processes. A more nimble software development lifecycle results in a competitive advantage for businesses and their customers.

- Need of DevOps
  1. Shorter development cycles that encourage innovation: It is generally known that the more innovative companies are, the higher their chances of outrunning the competition.

  2. More collaboration, better communication: The DevOps culture is based on achieving the best performance in such a union, instead of worrying about individual objectives.As a result of both departments being fused, the process becomes more fluid since everyone is oriented towards a common goal.

3. Reduced deployment failures and faster time to recover:Most failures during development occur due to programming defects. Having a DevOps team will allow for more releases in shorter time spawns. This way, it is easier and more likely to find possible defects in the code.

4. Efficiency: Improved resource management:Increased efficiency helps speed up development and reduce coding defects and problems.

- DevOps Principles
  1. 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.

  2. 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.

  3. Continuous Improvement: 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, allowing DevOps teams to continuously push updates that improve the efficiency of software systems.

  4. Hyperfocus on user needs with short feedback loops: DevOps teams can take a moment and focus on what real users really want, and how to give it to them.

- DevOps Practices
  1. Continuous development: This practice spans the planning and coding phases of the DevOps lifecycle. Version-control mechanisms might be involved.

  2. Continuous testing: This practice incorporates automated, prescheduled, continued code tests as application code is being written or updated. Such tests can speed the delivery of code to production.

  3. Continuous integration (CI): This practice brings configuration management (CM) tools together with other test and development tools to track how much of the code being developed is ready for production. It involves rapid feedback between testing and development to quickly identify and resolve code issues.

  4. Continuous delivery: This practice automates the delivery of code changes, after testing, to a preproduction or staging environment. A staff member might then decide to promote such code changes into production.

5. Continuous deployment (CD): Similar to continuous delivery, this practice automates the release of new or changed code into production.

6. Continuous monitoring: This practice involves ongoing monitoring of both the code in operation and the underlying infrastructure that supports it. A feedback loop that reports on bugs or issues then makes its way back to development.

- DevOps Engineer role and responsibilities
  A DevOps engineer's roles and responsibilities are a combination of technical and management roles. It is essential to have excellent communication and coordination skills to successfully integrate various functions in a coordinated manner and deliver the responsibilities to the customer's satisfaction.

  Some of the core responsibilities of DevOps Engineer include:
  1. Understanding customer requirements and project KPIs.
  2. Implementing various development, testing, automation tools, and IT infrastructure.
  3. Planning the team structure, activities, and involvement in project management activities.
  4. Managing stakeholders and external interfaces.
  5. Setting up tools and required infrastructure.
  6. Defining and setting development, test, release, update, and support processes for DevOps operation.

B) Case study on DevOps use cases & Real- life applications

- **Name of the Organization implementing DevOps:** WALMART

- **Challenges faced by Walmart in implementing DevOps:**
  1. Cultural Shift: One of the most significant challenges is fostering a cultural shift within the organization. DevOps requires breaking down silos between development and operations teams, promoting collaboration, and embracing a culture of continuous improvement and learning.
  2. Legacy Systems: Walmart, being a large and established company, might have a considerable legacy IT infrastructure. Integrating DevOps practices into such systems can be challenging due to their complexity and potential resistance to change.
  3. Scale and Complexity: Walmart operates at a massive scale, with a vast number of products and services. Implementing DevOps across such a complex environment requires careful planning

and execution.

4. Security and Compliance: Ensuring the security and compliance of the software delivery pipeline is crucial, especially for a retail giant like Walmart that deals with sensitive customer data and financial transactions.
5. Tools and Automation: Selecting the right tools and setting up effective automation is critical for the success of DevOps. Integrating existing tools with new ones and ensuring a seamless flow of information can be challenging.
6. Reskilling and Training: DevOps often requires new skills and knowledge, both technical and non-technical. Training existing employees and hiring new talent with the necessary expertise can be time-consuming and challenging.
7. Resistance to Change: Resistance to DevOps practices might come from various levels within the organization. Some team members might be reluctant to change their established processes or may not fully understand the benefits of DevOps.
8. Measuring Success: Defining and measuring success metrics for DevOps initiatives can be tricky. Identifying the right Key Performance Indicators (KPIs) and aligning them with business objectives is essential.
9. Communication and Collaboration: Effective communication and collaboration are central to successful DevOps implementation. Encouraging open communication between teams and resolving conflicts constructively can be challenging.
10. Continuous Improvement: Embracing a culture of continuous improvement is a core principle of DevOps. However, ensuring that teams consistently work towards improving processes and reducing inefficiencies can be a challenge.

- **Solution to overcome the challenges:**

1. Top-Down Support: Successful DevOps adoption requires support from top-level management. When executives and leaders champion the cultural shift and allocate resources for training and infrastructure improvements, it sets a clear message to the organization that DevOps is a priority.

2. Pilot Projects: Starting with pilot projects can be an effective way to demonstrate the benefits of DevOps on a smaller scale. This allows teams to learn from the initial challenges, iterate on their processes, and gradually expand the practices across the organization.

3. Collaborative Workshops and Training: Organizing workshops, training sessions, and team-building activities can help foster collaboration between development and operations teams. Providing employees with the necessary skills and knowledge ensures a smoother transition to DevOps practices.

4. Gradual Automation: Rather than attempting to automate everything at once, organizations often begin by automating repetitive and error-prone tasks. Gradual automation helps build confidence in the process and ensures that automation efforts align with the organization's needs.

5. Security Integration: Addressing security concerns from the outset is crucial. Integrating security practices into the development and deployment process helps ensure that vulnerabilities are addressed early in the software delivery pipeline.

6. Continuous Monitoring and Feedback: Implementing continuous monitoring and feedback loops allows teams to gather data on the effectiveness of their DevOps practices. This information enables them to make data-driven decisions and identify areas for improvement.

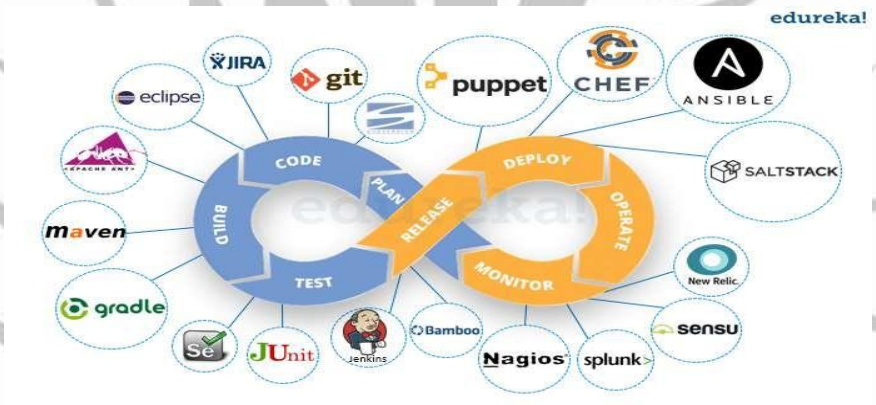
7. DevOps Champions and Advocates: Identifying DevOps champions within the organization can help drive adoption. These advocates can lead by example, share success stories, and provide guidance to teams during the transition.

8. Communication and Documentation: Transparent communication and well-documented processes are essential for successful DevOps implementation. Providing clear guidelines and sharing best practices ensure that everyone is on the same page.

9. Tools Evaluation and Integration: Selecting the right tools that align with the organization's needs is crucial. Teams should evaluate various DevOps tools and integrate them effectively into their workflows.

10. Continuous Learning: Encouraging a culture of continuous learning and improvement helps teams stay updated with the latest DevOps practices and technologies.

List of various tools used in DevOps Life Cycle.



1. Discover:
   a) Jira Product Discovery
   b) Mural
   c) Miro

2. Plan:
   a) Slack
   b) Confluence
   c) Jira Software

3. Build:
   a) Kubernetes
   b) Docker

4. Infrastructure as code:
   a) Bitbucket
   b) Github
   c) GitLab

5. Continuous Delivery:
   a) Jenkins
   b) AWS
   c) Bitbucket
   d) Ciecleci
   e) Sonarsource

6. Test:
   a) Xray
   b) Stackhawk
   c) Mabl
   d) Snyk

7. Deploy:
   a) Jira Software
   b) Bitbucket
   c) AWS Codepipeline

8. Operate:
   a) Jira Service Management
   b) Jira Software
   c) Statuspage
   d) Opsgenie

9. Observe:
   a) Slack
   b) Opsgenie
   c) Splunk

10. Continuous Feedback:
    a) Slack
    b) GetFeedback
    c) Pendo
    d) Jira Service Management

Conclusion: We learnt and understood DevOps: Principles, practices, DevOps, Engineer role and responsibilities. Case study of Walmart DevOps implementation was also successful.