Questions and Answers for Viva

Experiment 1: DevOps Basics

1. **Q:** What are the key principles of DevOps?
    - **A:** DevOps emphasizes collaboration, automation, continuous integration, deployment, and monitoring between development and operations teams.
2. **Q:** Describe the DevOps lifecycle and its phases.
    - **A:** The lifecycle includes phases like planning, coding, building, testing, deployment, operations, monitoring, and continuous improvement.
3. **Q:** Why is continuous feedback important in DevOps?
    - **A:** Continuous feedback allows teams to make iterative improvements based on user experience and performance insights, ensuring ongoing product quality.

Experiment 2 (Git):

1. **What is Git?**
    - Git is a distributed version control system used to track changes in source code during software development.
2. **What is the purpose of the `git add` and `git commit` commands?**
    - `git add` stages the file for commit, and `git commit` permanently saves the staged changes in the repository.
3. **How do you push changes from a local repository to a remote one like GitHub?**
    - You use the `git push` command after committing the changes locally.

Experiment 3: Version Control with Git

1. **Q:** What is Git, and why is it used in software development?
    - **A:** Git is a distributed version control system used for tracking code changes, collaboration, and managing different versions of a project effectively.
2. **Q:** Explain the difference between `git add` and `git commit`.
    - **A:** `git add` stages changes to be committed, while `git commit` saves those staged changes to the local repository with a message.
3. **Q:** How does branching help in a collaborative environment?
    - **A:** Branching allows developers to work on separate features or fixes without impacting the main codebase, making it easier to manage and integrate changes.

Experiment 4 (Jenkins with Maven):

1. **What is Jenkins used for in DevOps?**
    - Jenkins is used for Continuous Integration (CI) to automate the process of building, testing, and deploying software applications.
2. **What role does Maven play in Jenkins?**
    - Maven is a build automation tool integrated with Jenkins to manage project dependencies and automate the build process.
3. **How does Jenkins improve software development workflows?**
    - Jenkins automates repetitive tasks such as testing and building software, ensuring that code changes are integrated continuously without manual intervention.

Experiment 5 (Jenkins CI/CD with Tomcat):

1. **What is a CI/CD pipeline in Jenkins?**
    - A CI/CD pipeline automates the process of building, testing, and deploying code changes to production.
2. **How does Jenkins deploy an application to Tomcat?**
    - Jenkins builds the application into a `.war` file, and using the "Deploy to Container" plugin, it deploys the `.war` file to the Tomcat server.
3. **What is the significance of using Tomcat with Jenkins?**
    - Tomcat serves as a web server for deploying Java-based web applications, and Jenkins automates the deployment process to Tomcat, ensuring continuous delivery.

Experiment 6: Jenkins Master-Slave Architecture with Scaling

1. **Q**: Why is the Master-Slave architecture beneficial in Jenkins?
    - **A**: It allows the distribution of workload across multiple nodes, enabling parallel execution and better resource management.
2. **Q**: How do you add a new Slave node in Jenkins?
    - **A**: By going to `Manage Jenkins -> Manage Nodes -> New Node`, configuring the node properties, and connecting via an agent.
3. **Q**: What command would you use to allow network access to a specific port for Jenkins agent communication?
    - **A**: `sudo ufw allow 50000/tcp` to allow communication on port 50000, which is required for agent communication.

Experiment 7: Selenium Automation

1. **Q:** What is Selenium, and what are its main components?

- **A:** Selenium is a web automation testing suite that includes components like Selenium IDE, Selenium WebDriver, and Selenium Grid for automating browser actions.

2. **Q:** Describe the steps for creating a basic test case in Selenium IDE.
    - **A:** The steps include recording user interactions, playing back the recorded script, and saving the test suite for future use.
3. **Q:** What is the role of continuous testing in DevOps?
    - **A:** Continuous testing ensures code quality at every development stage, reducing risks and allowing for rapid feedback by automating tests throughout the lifecycle.

Experiment 8: Container Lifecycle in Docker

1. **Q**: Describe the different states in a Docker container lifecycle.
    - **A**: The main states are Created, Running, Paused, Stopped, and Deleted, representing the various stages of a container's lifecycle.
2. **Q**: How do you check if Docker is correctly installed on your system?
    - **A**: By running the `docker version` command, which will output Docker's client and server versions if installed correctly.
3. **Q**: Which command would you use to run a new container with an interactive terminal session?
    - **A**: `docker run -it <image-name> bash`, which allows you to start a container and access its terminal.

Experiment 9: Building a Docker Image for a Web Application

1. **Q**: What is a Dockerfile, and how does it function in containerization?
    - **A**: A Dockerfile is a script containing instructions to create a Docker image. It automates the setup by defining all dependencies and configurations needed for an application within a portable image.
2. **Q**: Explain the role of the `COPY` and `RUN` commands in a Dockerfile.
    - **A**: The `COPY` command moves files from the host system to the Docker image, while `RUN` executes commands needed during the image build, such as installing software or configuring environments.
3. **Q**: How does a Docker container differ from a Docker image?
    - **A**: A Docker image is a read-only template with the application and environment setup, while a container is a runtime instance of the image, allowing for execution in an isolated environment.

Experiment 10: Ansible Installation on AWS Instances

1. **Q**: What is the purpose of creating an SSH key when setting up Ansible?
    - **A**: SSH keys provide secure, password-less authentication, allowing Ansible to connect to managed nodes automatically.
2. **Q**: How can you test the connection between Ansible's Controller and Slave nodes?
    - **A**: By using the command `ansible -m ping all`, which sends a ping request to all nodes listed in the inventory file.
3. **Q**: What role does the inventory file play in Ansible?
    - **A**: It defines the hosts or groups of hosts that Ansible will manage, enabling it to target specific machines for configuration and deployment tasks.

Experiment 11: Deploying LAMP Stack Using Ansible

1 Q: What is a LAMP stack, and what are its components?

- **A:** A LAMP stack consists of Linux (OS), Apache (Web Server), MySQL (Database), and PHP (Programming Language), commonly used for hosting websites and web applications.

2 Q: Explain the purpose of Ansible playbooks.

- **A:** Ansible playbooks automate tasks like configuration management and application deployment, streamlining the setup and scaling of infrastructure.

3 Q: How does Ansible benefit DevOps processes?

- **A:** Ansible automates repetitive tasks, improves consistency, and reduces human error, making it ideal for scalable and efficient DevOps operations.

Experiment 12: Deploying a Website Using Ansible and MySQL

1. **Q**: What are Ansible modules, and how are they used in a playbook?
    - **A**: Ansible modules are small scripts used to execute tasks on target machines, such as software installation or configuration. Playbooks use these modules to automate complex workflows across multiple machines.
2. **Q**: Describe the process of deploying a MySQL database using an Ansible playbook.
    - **A**: First, we write a playbook (YAML file) with tasks to install and configure MySQL. When executed, Ansible provisions the database on the target node, setting up the server as specified.

3. **Q**: What are the benefits of using Ansible playbooks in deployment automation?
    - **A**: Ansible playbooks simplify repetitive tasks, reduce errors, and ensure consistency in configuration across environments by defining the desired state in a single YAML file.