**DEVOPS**

**ASSIGNMENT -1**

1. **Which way of installing Jenkins would you prefer and why?**

The preferred way of installing Jenkins depends on the use case. However, the **Docker installation** is often the best choice due to its ease of setup, portability, and minimal impact on the host system.

**Docker uses:**

* **Quick and easy setup** with a single command
* **Isolated environment** avoiding conflicts with other applications
* **Portability** across different machines
* **Easy updates** by pulling the latest image
* **Consistent environment** across different systems

If Docker is not an option, installing Jenkins via **Linux package manager (apt/yum)** is a good alternative for production environments due to better system integration and service management.

1. **Write down the steps involved in building a web app, testing it and deployed to QA and Production.**

**1.Development Phase**

**1.1 Requirement Gathering & Planning**

Define project scope, features, and tech stack.

Set up version control (Git, GitHub).

Plan milestones using Agile or Scrum.

**1.2 Setting Up the Project**

Initialize frontend and backend frameworks.

Structure the project for scalability.

Configure environment variables and dependencies.

**1.3 Development (Coding)**

Build frontend UI with React, Angular, or Vue.js.

Develop backend API using Node.js, Django, or Flask.

Set up database (MySQL, MongoDB, Firebase).

Implement authentication (JWT, OAuth).

**2. Testing Phase**

**2.1 Unit Testing**

Test individual components and functions.

Use Jest, Mocha, PyTest, or JUnit.

**2.2 Integration Testing**

Test API and database interactions.

Validate request/response handling.

**2.3 End-to-End (E2E) Testing**

Test the entire user flow (Selenium, Cypress).

Ensure UI and backend work together smoothly.

**2.4 Code Review & Bug Fixing**

Conduct peer reviews.

Fix identified issues before deployment.

**3. Deployment to QA**

**3.1 Merging to QA Branch**

* Push tested code to the QA branch.
* Set up CI/CD pipeline for automated builds.

**3.2 Deploy to QA Server**

* Use Docker/Kubernetes or cloud platforms (AWS, GCP).
* Conduct functional, regression, and UAT testing.

**3.3 Bug Fixes & Approval**

* Identify and fix issues found in QA testing.
* Get approval from testers or stakeholders.

**4. Production Phase (Final Deployment & Maintenance)**

Once the application passes QA testing, it is deployed to production for real users.

**4.1 Merging to Main Branch**

* The finalized, tested code is merged into the production branch.
* Versioning is done to track changes.

**4.2 Deploying to Production Server**

* The application is deployed using **CI/CD pipelines** (Jenkins, GitHub Actions, GitLab CI/CD).
* Hosting services such as **AWS, Azure, GCP, or DigitalOcean** are used.
* Containers (Docker) or orchestration tools (Kubernetes) manage deployment.

**4.3 Database & API Configuration**

* Optimize database connections and indexing.
* Enable caching (Redis, Memcached) for performance.
* Secure API endpoints with HTTPS, CORS, and rate limiting.

**4.4 Monitoring & Performance Optimization**

* **Monitoring tools** (Prometheus, CloudWatch, Grafana) track uptime and server health.
* Logging tools (ELK Stack, Datadog) help debug production issues.
* Performance tuning (CDN, load balancing, server scaling).

**4.5 Security & Backup Strategies**

* Implement **firewalls, authentication, and encryption** for security.
* Perform regular **database backups** and disaster recovery planning.
* Conduct security audits and penetration testing.

**4.6 Rollback & Maintenance**

* Implement **Blue-Green or Canary deployments** for smooth updates.
* Enable rollback strategies in case of failure.
* Provide continuous maintenance with regular updates and patches.