**DEVOPS**

**Name : SOWMIYA R**

**Roll no : 22CDR099**

**Class : CSD -B**

**ASSIGNMENT -1**

**1. Preferred Method for Installing Jenkins**

The recommended way to install Jenkins is by using Docker, as it offers a quick and efficient setup. This method eliminates dependency conflicts and ensures an isolated environment, making upgrades and rollbacks seamless.

✅ **Rapid Installation** – No need to manually configure Java or other dependencies.  
✅ **Simplified Cleanup** – Containers can be removed easily when no longer needed.  
✅ **Cross-Platform Compatibility** – Runs uniformly across different operating systems.  
✅ **Minimal System Impact** – Prevents unnecessary software installations.  
✅ **Effortless Upgrades** – Simply update by pulling the latest Jenkins image.

**2. Steps for Building, Testing, and Deploying a Web Application**

**Phase 1: Development – Building the Web App**

**Step 1: Planning & Requirement Analysis**

* Identify the project's scope, features, and preferred technology stack (e.g., MERN, Django).
* Establish a repository on platforms like GitHub, GitLab, or Bitbucket.
* Define a workflow strategy (Kanban, Agile, Scrum).

**Step 2: Setting Up the Development Environment**

* Install necessary tools (Node.js, Python, Docker, database systems).
* Initialize the project (npm init, pip install).
* Configure an appropriate code editor (VS Code, WebStorm) and frameworks (React, Express).

**Step 3: Development**

* Create the frontend using React, Angular, Vue, or HTML/CSS.
* Develop the backend with technologies such as Node.js, Django, Flask, or Spring Boot.
* Implement database support (MongoDB, PostgreSQL, MySQL).
* Secure authentication using JWT, OAuth, or Firebase.

**Step 4: Version Control**  
Push the project to a Git repository:

git init

git add .

git commit -m "Initial commit"

git push origin main

**Phase 2: Testing – Ensuring Quality**

**Step 5: Unit Testing**

* Conduct unit tests on individual components using Jest, Mocha, or PyTest.

test('adds 1 + 2 to equal 3', () => {

expect(1 + 2).toBe(3);

});

**Step 6: Integration & API Testing**

* Validate API functionality using Postman, Newman, or Supertest.
* Automate testing with Cypress or Selenium.

**Step 7: UI/UX Testing**

* Verify compatibility across multiple browsers (Chrome, Firefox, Edge).
* Assess responsiveness on mobile devices.

**Step 8: Security Testing**

* Perform vulnerability assessments with OWASP ZAP or Burp Suite.
* Ensure security measures like SSL, CORS, and authentication mechanisms.

**Phase 3: Deployment – QA & Production**

**Step 9: Deployment to QA Environment**  
Containerize the application with Docker:

docker build -t myapp .

docker run -d -p 3000:3000 myapp

Deploy to QA infrastructure (AWS, DigitalOcean, Azure).  
Conduct manual and automated testing.

**Step 10: Production Deployment**

* Implement CI/CD pipelines using GitHub Actions, Jenkins, or GitLab CI/CD.
* Deploy with Kubernetes, Docker Swarm, or AWS Elastic Beanstalk.
* Execute performance tests (e.g., JMeter) before launching.
* Utilize Blue-Green Deployment for zero downtime.

**Phase 4: Post-Deployment – Monitoring & Maintenance**

**Step 11: Monitoring**

* Track application performance with Prometheus, Grafana, or Datadog.
* Implement logging using the ELK Stack (Elasticsearch, Logstash, Kibana).

**Step 12: Bug Fixes & Updates**

* Gather user feedback and resolve reported issues.
* Introduce new features via feature flagging.
* Maintain a rollback strategy for rapid recovery in case of failures.