**DEVOPS**

**ASSIGNMENT -1**

1. **Installing Jenkins:** I prefer installing Jenkins with Docker because it is quick to set up and easy to manage. Docker avoids dependency problems and runs Jenkins in its own container, making it simple to upgrade or roll back changes.

* Quick setup – No need to install Java or other software by hand.
* Easy cleanup – You can remove the container when you no longer need it.
* Portability – It works the same on any operating system.
* No system pollution – It does not install extra packages on your computer.
* Easy upgrades – You just pull the latest Jenkins image.

1. **Steps to Build, Test, and Deploy a Web App:**

**Development Phase – Building the Web App**

* **Requirement Gathering & Planning:**
  + Decide the project scope, features, and technology (like MERN or Django).
  + Set up a code repository (for example, on GitHub).
  + Choose a workflow (such as Kanban, Agile, or Scrum).
* **Setting Up the Development Environment:**
  + Install the necessary software (like Node.js, Python, Docker, and a database).
  + Start the project with a package manager.
  + Choose a code editor and frameworks (like React or Express).
* **Writing Code:**
  + Create the frontend using technologies like React, Angular, or Vue.
  + Develop the backend with Node.js, Django, Flask, or Spring Boot.
  + Connect the app to a database (such as MongoDB, PostgreSQL, or MySQL).
  + Add user authentication (using methods like JWT or OAuth).
* **Version Control:**
  + Create a Git repository and save your changes regularly.

**Testing Phase – Ensuring Quality**

* **Unit Testing:**
  + Write tests for each component using tools like Jest, Mocha, or PyTest.
* **Integration and API Testing:**
  + Test your API endpoints with tools like Postman.
  + Automate these tests using tools like Cypress or Selenium.
* **UI/UX Testing:**
  + Check that the app works in different browsers and on mobile devices.
* **Security Testing:**
  + Scan for vulnerabilities using tools like OWASP ZAP.
  + Ensure proper use of SSL, CORS, and authentication.

**Deployment Phase – QA and Production**

* **Deploying to QA:**
  + Package the app in a container.
  + Deploy it to a QA server (on AWS, DigitalOcean, etc.).
  + Let the QA team test the app manually and automatically.
* **Deploying to Production:**
  + Use CI/CD pipelines (like Jenkins or GitHub Actions) to deploy the app.
  + Use tools like Kubernetes for managing the deployment.
  + Run load tests before launching.
  + Deploy with methods that ensure no downtime (for example, blue-green deployment).

**Post-Deployment – Monitoring and Maintenance**

* **Monitoring:**
  + Keep an eye on the app with tools like Prometheus or Grafana.
  + Set up logging with systems like the ELK stack.
* **Bug Fixes and Updates:**
  + Listen to user feedback and fix any problems.
  + Roll out new features gradually using feature flags.
  + Be ready to roll back if something goes wrong.