PROBLEM STATEMENT 6

Problem Statement: Building an E-commerce Microservices Application on Cloud using Docker, Kubernetes, Jenkins and Git.

Overview :

The aim of this project is to develop an e-commerce microservices application that can be deployed on the cloud using Docker, Kubernetes, Jenkins, and Git. The application will consist of several microservices that will be deployed as Docker containers on a Kubernetes cluster. Jenkins will be used for continuous integration and deployment, while Git will be used for version control.

Objectives:

- Design and implement the microservices architecture for the application.
- Create Docker containers for each microservice.
- Use Kubernetes to orchestrate the containers locally.
- Implement a Jenkins pipeline to automate the deployment process.
- Integrate Git with Jenkins to trigger the pipeline on code changes.

Week 1: Microservices Architecture Design and Development

- Define microservices that will be part of the application.
- Determine communication protocols between microservices.
- Plan data models and schemas for each microservice.
- Develop microservices using appropriate programming languages and frameworks.
- Implement REST APIs to allow communication between the microservices.
- The app should contain different modules connected to a database to store data
- For instance, a user page, product page and order page
    - User Management: This module handles the registration, authentication, and authorization of users. It allows users to create accounts, login, and manage their profiles.
    - Product Management: This module handles the management of products. It allows admins to add, edit, and delete products. It also allows users to view and search for products.
    - Order Management: This module handles the management of orders. It allows users to view their order history, track their orders, and manage their orders.

- Review Management (optional): This module handles the management of product reviews. It allows users to view and add reviews for products.

Deliverables:

- Microservices architecture document.
- Codebase for implemented microservices with REST APIs.

Week 2: Containerization and Orchestration:

- Write Dockerfiles for each microservice.
- Build and test Docker images for each microservice.
- Create Kubernetes deployment manifests for each microservice.
- Set up Kubernetes services for each microservice.
- Test and validate the Kubernetes deployment.

Deliverables:

- Docker images for each microservice.
- Kubernetes deployment manifests and services.

Week 3: Continuous Integration and Deployment:

- Setup Jenkins on a server.
- Create Jenkins jobs and corresponding Jenkinsfiles for building, testing, and deploying microservices.
- Configure Jenkins to monitor the Git repository for changes and trigger automatic builds and deployments.
- Create a Git repository for the microservices code.
- Commit and push code changes to the Git repository.
- Utilise Git for version control, managing different versions and branches of the codebase.

Deliverables:

- Jenkins jobs and configuration files.
- Automated builds and deployments triggered by Git commits.

GENERAL GUIDELINES FOR THE PROJECT

- **Create a GitHub Repository.**
- **Weekly progress according to the problem statement assigned must be pushed to the repo and this will be considered while evaluating.**
- **Each team's weekly progress update in the repository must be shown to the teachers in class.**
- **Name the GitHub repository with the four SRN's in order and the Project title.**