PHASE 4 PROJECT DESCRIPTION

ALL THE THINGS INTEGRATION AND DEPLOYMENT PHASE 4 PROJECT

PROBLEM STATEMENT:

This assignment is designed to help you understand how to plan and develop the back end of a given problem, gain hands-on experience building the CI/CD Pipeline using Jenkins, and containerize the application on the AWS cloud platform. To build an infrastructure to host software on an AWS EC2 instance for a clinic to have online data access for pets, their owners, and consultation fees.

PREREQUISTIES:

- **Spring Boot Project Import**: The Spring Boot project, available on Google Drive, was successfully pushed into Github and need to pull the code from Github.
- **Jenkins**: It is a popular CI/CD pipeline tool that helps automate and integrate multiple technologies and stages of software development.
- **Docker:** Apart from various use cases for Docker, it is mostly used for containing an entire application as a lightweight image, which can then be deployed on multiple servers.
- **AWS AND EC2**: AWS is one of the most popular cloud platforms for managing different applications and projects of any scale. EC2 is one of the simplest ways to get an application deployed.

TASKS NEED TO BE DONE:

1**. Push the Spring Boot project with the generated code in Github**: The Spring Boot project, available on Google Drive, was successfully pushed into Github and need to pull the code from Github.

2. Configure the project with Dockerfile and Jenkinsfile :
Docker Configuration: A Dockerfile was created to package the Spring Boot application into a Docker container. All necessary dependencies and configurations were included to ensure a self-contained and portable deployment unit.
Jenkins Configuration: A Jenkinsfile was implemented to define the CI/CD pipeline stages. Jenkins was configured to connect to the Git repository, enabling automated builds and deployments.

3. Build the project using the Maven package:
Maven Build: The Maven build process was successfully executed locally, ensuring the Spring Boot application was packaged correctly.

4. Create and launch AWS EC2 instance:
Create and Launch EC2 Instance: An EC2 instance was launched on AWS with an appropriate instance type, providing the necessary compute resources.

5. Configure EC2 instance with JDK 11, Docker, and Jenkins:
JDK Configuration: The EC2 instance was configured with both JDK 8 and JDK 11, and environment variables were set to facilitate Java execution.
Docker Installation: Docker was installed on the EC2 instance, with configurations set to start on boot, ensuring the environment is ready for containerized deployments.
Jenkins Installation: Jenkins was installed on the EC2 instance, creating a platform for orchestrating the CI/CD pipeline. The service was configured to start automatically.

6. Sync the given code to the Git repository:
Git Repository: The project code was uploaded to a Git repository, establishing version control for the software solution.

7. Create Jenkins pipeline on EC2 with Git and GitHub:
Jenkins Pipeline: A Jenkins pipeline was configured on the EC2 instance, with the source code management (SCM) set to the Git repository. The pipeline included stages to build the application and containerize it using Docker.

8. Build the pipeline to dockerize the application:
Build and Dockerization: The pipeline successfully executed, automating the build process and containerizing the Spring Boot application using Docker within the Jenkins pipeline.

9. Project Deployment:
Deployment on AWS EC2: The Dockerized Spring Boot application was deployed on the AWS EC2 instance, providing online access to Dr. Shawn's pet clinic software. CI/CD Pipeline Workflow: The CI/CD pipeline workflow was established, triggered

by code changes. The automated deployment process ensured efficient and reliable software updates.

Monitoring and Scaling: Monitoring tools were implemented to keep track of application performance. Potential scaling strategies were considered to handle increased load.

10.Conclusion: The project successfully achieved its goals, hosting the pet clinic application on AWS with a robust CI/CD pipeline. The implemented solution enhances development efficiency, facilitates seamless deployments, and establishes a foundation for future enhancements and support.