JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY

BTECH CSE 5TH SEMESTER

MINOR PROJECT-1 (15B19CI591)

WORK SUMMARY SHEET

DOCKIFY

(Deploying an application on Docker and Kubernetes with Jenkins CI/CD)

Mentor:

Dr. Amarjeet Prajapati

Submitted By:

Mukund Sarda (21103105)

Karanjot Singh (21103096)

Priyanshu Jain (21103102)

1. Motivation Behind the Project:

We want to make it easier and smoother to launch software applications. Our project focuses on using Docker, Kubernetes, and Jenkins CI/CD to simplify the deployment process, improve teamwork, and make the overall software delivery pipeline more efficient. We believe that by exploring and implementing these technologies, we can help teams deploy their applications faster and with fewer hiccups, ultimately making the development process more effective.

2. Type of Project:

Research cum Development Project

3. Details of The New Technologies, Tools, And Software Including Programming Languages Learned While Developing the Project:

We will be using Jenkins as a CICD tool and deploying our application on a Docker container and Kubernetes Cluster. We learned various Linux commands along with docker commands through the deployment part and JavaScript, ReactJs for the development part.

Jenkins: Jenkins is an open-source CI/CD tool automating software development processes.

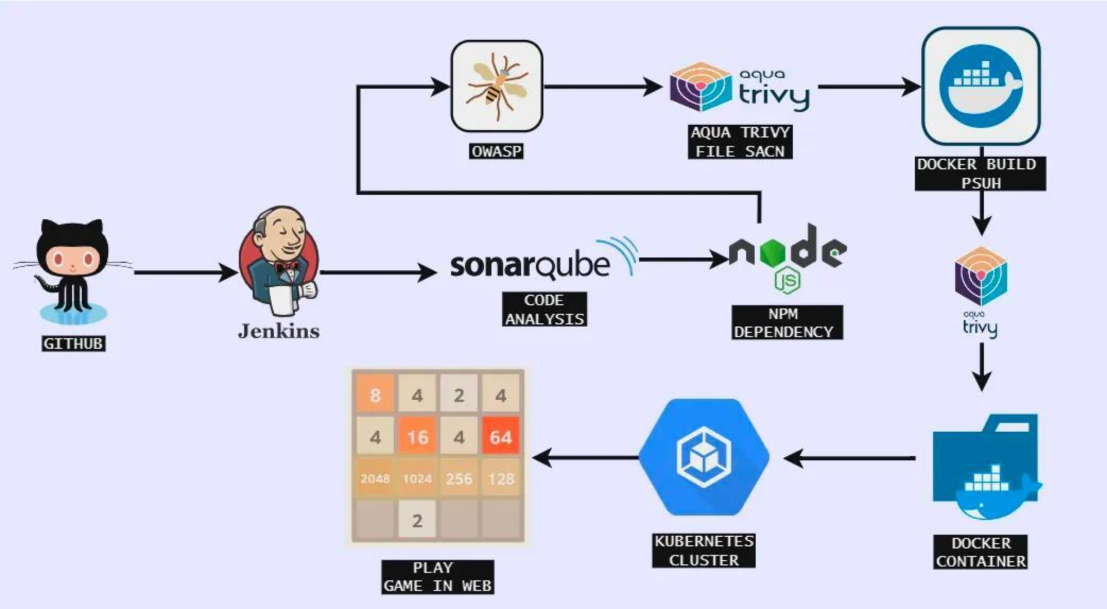
Docker: Docker is a containerization platform streamlining application deployment and ensuring consistency.

Kubernetes: Kubernetes is an open-source container orchestration tool automating deployment, scaling, and management.

Linux Commands: Linux commands are instructions used in Linux for system interaction and task execution.

JavaScript and React: JavaScript is a versatile programming language, and React is a JavaScript library, both used in web development for creating interactive and dynamic user interfaces.

4. Overall Design of Project:



5. Proposed Methodology:

At first, we critically analysed the problems that were solved by using modern techniques like docker, Kubernetes over traditional client server deployments.

Then we created a sample project for testing these tools, particularly a react.js web-based application performing some basic operations, with the main goal being optimizing the deployment.

Then we move on to deployment section:

1. Infrastructure Setup

Step 1: Launch an Ubuntu Instance

2. Jenkins, Docker, and Trivy Installation

Step 2: Install Jenkins, Docker, and Trivy

Step 3: Create a Sonarqube Container using Docker

A Docker container running Sonarqube is created to facilitate code quality analysis during the CI/CD pipeline.

3. Jenkins Configuration

Step 4: Install Jenkins Plugins

Step 5: Create a Pipeline Project

Step 6: Install OWASP Dependency Check Plugins

4. CI/CD Pipeline Execution

Step 7: Docker Image Build and Push The CI/CD pipeline is triggered

5. Kubernetes Setup

Step 8: Kubernetes Master and Slave Setup Kubernetes is set up on an Ubuntu environment

6. Description of The Work:

This project focuses on deploying the popular 2048 Game onto a dynamic and scalable infrastructure through a comprehensive DevOps pipeline. Beginning with the setup of an Ubuntu T2 Large Instance on AWS, the deployment leverages Jenkins CI/CD pipelines to automate the build, testing, and deployment phases. Docker containers encapsulate the game, ensuring portability and consistency, while Kubernetes orchestrates containerized applications for scalability. The project integrates security measures, utilizing Trivy for vulnerability scanning. The deployment culminates with the accessible 2048 Game, demonstrating the synergy between modern DevOps practices and containerization technologies.

7. Division of The Work Among Students

Karanjot:

- **Infrastructure Setup:** Responsible for launching Ubuntu. This involves configuring the instance and ensuring it meets the necessary requirements for hosting the application.
- **Jenkins, Docker, and Trivy Installation:** Responsible for handling the installation of Jenkins, Docker, and Trivy on the Ubuntu instance. This includes configuring Docker and setting up Trivy for vulnerability scanning.
- **Jenkins Configuration:** Installed and configured the required Jenkins plugins (JDK, Sonarqube Scanner, Nodejs, and OWASP Dependency Check).

Priyanshu:

- **CI/CD Pipeline Execution**: Took charge of the CI/CD pipeline execution. This involves configuring the Jenkins pipeline to build and push the Docker image, as well as deploying the image using Docker.
- **Kubernetes Setup:** Responsible for setting up Kubernetes on the Ubuntu environment. This includes configuring the master and slave nodes to ensure a robust and scalable Kubernetes cluster.

Mukund:

- **Frontend application -** Responsible for creating the sample application in react for further deployment.
- **Documentation** -Document the deployment process, detailing the steps followed in each phase. Create a user guide documenting how end-users can access and interact with the deployed 2048 Game.
- **Game Access and Cleanup:** Focus on verifying the accessibility of the application through a web browser after deployment. Additionally, handled the cleanup phase, ensuring that AWS EC2 instances are terminated once the testing and deployment are successful.

9. Results:

The successful execution of this project has yielded significant results in terms of automation, scalability, and reliability in the deployment of the 2048 Game. Leveraging Docker for containerization and Jenkins for CI/CD, the project has streamlined the software delivery process, reduced manual errors and enhanced code quality through automated testing.

The integration of Kubernetes provides a robust platform for orchestrating containerized applications, ensuring scalability and fault tolerance. Security measures, including vulnerability scanning with Trivy, contribute to a heightened security posture for the deployed infrastructure.

The accessible 2048 Game marks the culmination of a well-coordinated deployment pipeline, showcasing the synergy between DevOps practices and containerization technologies. Documentation efforts have encapsulated valuable insights, serving as a knowledge base for future reference, troubleshooting, and continuous improvement. Overall, the project results in a deployable and scalable 2048 Game, emphasizing the effectiveness of modern DevOps methodologies and container orchestration in software deployment.

10. Conclusion:

The deployment of the application on Docker and Kubernetes with Jenkins CI/CD addresses several critical challenges in modern software development and deployment. One of the primary issues it resolves is the streamlining of the software delivery process through automated and consistent workflows.

By integrating Jenkins CI/CD pipelines, developers can ensure that the application undergoes systematic testing, builds, and deployments, minimizing manual errors and enhancing overall code quality.

The use of Docker containers facilitates a standardized and isolated runtime environment, promoting portability and scalability across different computing environments.