Project 1

Building a CI/CD Pipeline for a Retail Company

Table of Content

1. Business Challenge/Requirement

ABC Technologies is a leading online retail store, and it has recently acquired a large retail offline business store. The business store has a large number of stores across the globe but is following the conventional pattern of development and deployment. As a result, it has landed at a great loss and is facing the following challenges.

- Low available
- Low scalable
- Low performance
- Hard to built and maintain
- Developing and deploying are time-consuming

ABC will acquire the data from all these storage systems and plans to use it for analytics and prediction of the firm's growth and sales prospects. In the first phase, ABC has to create the servlets to add a product and display product details. Add servlet dependencies required to compile the servlets. Create an HTML page that will be used to add a product. The team is using Git to keep all the source code.

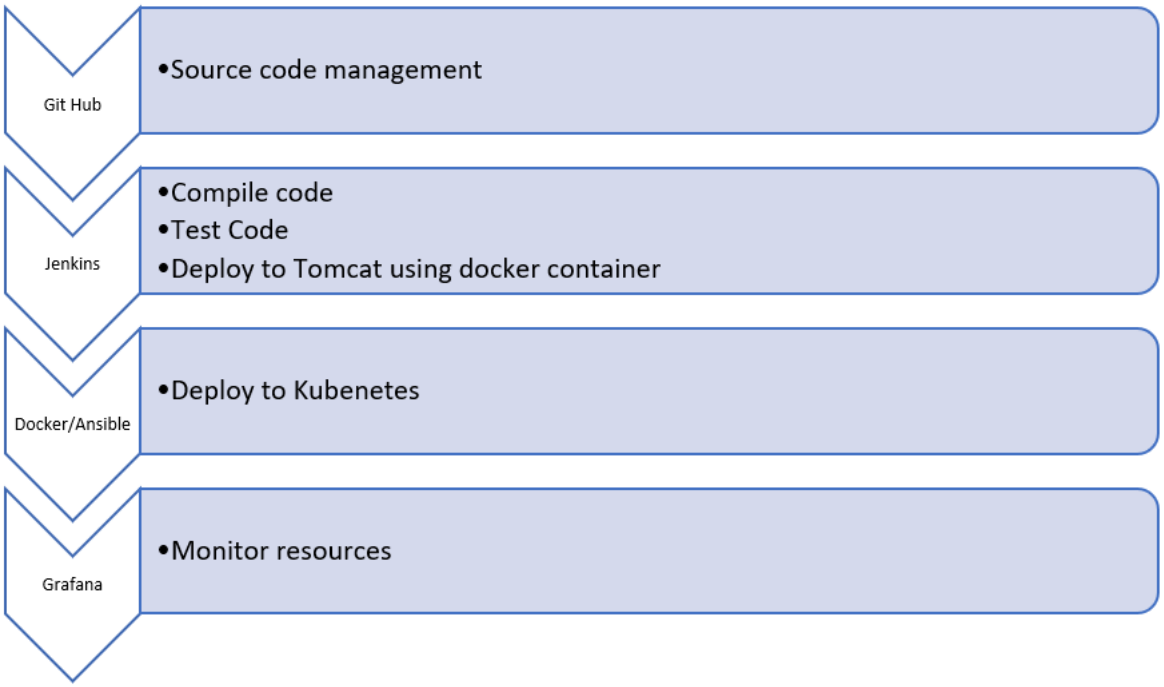
ABC has decided to use the DevOps model. Once source code is available in GitHub, we need to integrate it with Jenkins and provide continuous build generation for continuous delivery as well as integrate with Ansible and Kubernetes for deployment. Use Docker Hub to pull and push images between Ansible and Kubernetes.

2. Goal of the Project

Below are some of the high-level goals of this project:

- Implement CI/CD such that ABC Company is able to be—
  - highly available
  - highly scalable
  - highly performant
  - easily built and maintained
  - developed and deployed quickly

3. Data Flow Architecture/Process Flow



4. Data Explanation and Schema

Sample Java project has been shared for usage. It is a Maven project and has src and test folders created in it. It has a POM.xml file that lists all the needed dependencies to execute this project.

5. Problem Statements/Tasks

We need to develop a CI/CD pipeline to automate the software development, testing, packaging, and deployment, reducing the time to market the app and ensuring good quality service is experienced by end users. In this project, we need to—

- push the code to our GitHub repository.
- create a continuous integration pipeline using Jenkins to compile, test, and package the code present in GitHub.
- Write Dockerfile to push the war file to the Tomcat server.
- Integrate Docker with Ansible and write the playbook.
- Deploy artifacts to the Kubernetes cluster
- Monitor resources using Grafana.

6. Prerequisites

Verify the following software is installed in the working machine.

1. Java
2. Maven
3. Git
4. Jenkins
5. Docker
6. Ansible
7. Kubernetes
8. Grafana
9. Prometheus

7. Approach to Solve

Task 1: Clone the project from the GitHub link shared in resources to your local machine. Build the code using Maven commands.

Task 2: Set up the Git repository and push the source code. Then, log in to Jenkins.

1. Create a build pipeline containing a job for each
    - One for compiling source code
    - Second for testing source code
    - Third for packing the code
2. Execute the CI/CD pipeline to execute the jobs created in step 1
3. Set up a master-slave node to distribute the tasks in the pipeline.

Task 3: Write a Docket file. Create an Image and container on the Docker host. Integrate docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.

1. Enhance the package job created in step 1 of task 2 to create a docker image.
2. In the Docker image, add code to move the war file to the Tomcat server and build the image.

Task 4: Integrate the Docker host with Ansible. Write an Ansible playbook to create an image and create a continuer. Integrate Ansible with Jenkins. Deploy Ansible-playbook. CI/CD job to build code on ansible and deploy it on docker container

1. Deploy Artifacts on Kubernetes
2. Write pod, service, and deployment manifest file
3. Integrate Kubernetes with Ansible
4. Ansible playbook to create deployment and service

Task 5: Using Prometheus, monitor the resources like CPU utilization: Total Usage, Usage per core, usage breakdown, memory, and network on the instance by providing the endpoints on the local host. Install the node exporter and add the URL to the target in Prometheus. Using this data, log in to Grafana and create a dashboard to show the metrics.

8. Considerations/Assumptions

Resources Needed

- An AWS account
- A GitHub account
- MobaXterm/Putty
- Git Bash setup
- Source code

9. Deliverables

- Create a detailed solution document with screenshots for each task.
- Please submit the complete code developed by you, including the Dockerfile, playbook, etc.
- Please submit all the snapshots.

10. Business Benefits

After the solution is built, the business will have the below operational benefits:

1. Highly available
2. Highly scalable
3. Highly performant
4. Easily built and maintained
5. Developed and deployed quickly
6. Lower production bugs
7. Frequent releases
8. Better customer experiences
9. Lesser time to market

11. How to submit the project

- You can even upload all the scripts/files/code into your GitHub repository and share your repository with us.
- Also, share the detailed solution document containing a step-by-step screenshot of the tasks.

12. Marks Allocation

- Creation of CI pipeline in Jenkins [20 Marks]
- Creation of Dockerfile and integration with Ansible [30 Marks]
- Deploy artifacts to Kubernetes [35 Marks]
- Creation of Prometheus to monitor node [15 Marks]