Guten Tag

Mein Name ist Selahattin Karakaya, ich bin 44 Jahre alt, verheiratet und habe drei Kinder.

Ich präsentiere Euch heute meine Abschlussprojects mit dem Thema „Kontinuierliche Integration und Verteilung“ CICD Pipeline.

Hier haben wir die Agenda die heutige :

Projektumfeld

Die Nexus / IPS GmbH ist eine Tochterfirma der Nexus/ag.

Ist Anbieter für die Dokumentation der Medizinprodukte-Aufbereitung und Prozesskommunikation mit seinen Softwarelösungen

Nexus / IPS beschäftigt 33 Mitarbeiter mit im Standort Kassel

Dieses Diagramm zeigt das Zielsystem

Das Projekt wurde unter diesen 4 Haupttiteln geplant.

Die Zeit unter Planung- Durchführung- Validierung- und Abschluss Phasen eingeteilt.

die maximale Zeit für die Realisierungsphase vorgesehen.

Denn es geht darum, mehrere Systeme gemeinsam aufzubauen und zu verwalten.

**Projektplanung**

* **Ich war für die Umsetzung des Projektes verantwortlich.**
* **Her Michael Sieber is als Softwareentwickler verantwortlich für die Projektdefinition und Unterstützung**
* **Herr Andre Hosbach ist als Produktmanager verantwortlich für die Projektübergabe; weil als Produktmanager diese CICD-Pipeline von ihm benutzt würde.**

Projektplanung

* Sachmittelplanung
* Drei Virtuelle Maschinen sind zu nutzen für Isolierung.Eine Dafür wird ein Notebook mit SSD-Festplatte verwendet. Dieses Notebook muss mit dem LAN-Netzwerk des Unternehmens verbunden sein.
* Dieses Projekt kann mit vom bereits gekauft oder Freeware Software abgeschlossen werden.

Die im Projekt benutzte Software sind : Git, Docker, Docker-Registry, Jenkin, Gradle, Minikube, MobaXterm, VMWare Workstation.

Die Kosten der Hardware und Software sind nicht zu berücksichtigen, die Ausgewählte Hard- und Software wurde zuvor vom Unternehmen erworben oder sind Freeware.

Im Rahmen des Projektes wurde Räumlichkeit und Arbeitskosten des Personals berechnet.

Für ein Software Produkt Manager 30,93 €,

für ein Senior Software Developern 25,83 € zu berücksichtigen.

Räumlichkeit kosten für jede Mitarbeiter wird 35 € berechnet. (für die Nutzung der Räumlichkeiten, Strom und Arbeitsmaterialien berücksichtigt. )

In diesem Swim-Lane Diagram sehen wir die Arbeitsabläufe vor dem Projekt.

Jenkins ist schon verwendet als Automationsserver. wurde das Jenkins-Tool im Unternehmen verwendet.

Der vom Entwickler vorbereitete Quellcode wird in das GitHub-Repository gepusht.

Nach diesem Push-Verfahren wird ein Jenkins\_job triggert und

Der Quellcode wurde von Jenkins aus Github abgerufen und die folgende Vorgänge (kompile, unit-test, build in .jar form Artefakt und dockerization) werden automatisch von Jenkins durchgeführt.

Das resultierende Image von Artefakt im .jar-Format wird manuell extrahiert und die Sicherungs der Docker\_Image - und Bereitstellungsvorgänge wurden manuell durchgeführt.

Die Prozesse der Speicherung von Docker\_Images in einem Repository und der Bereitstellung in einer Kubernetes-Umgebung werden ebenfalls automatisiert.

Durch den Einsatz der mit dem Projekt zu beschaffenden CI/CD Pipeline werden Zeit und damit Personalkosten eingespart.

Die manuellen Prozesse des Produktmanagers dauern mindestens 30 Minuten pro Tag.

Nehmen wir an, dass das Jahr 220 Arbeitstage hat.

Durch die Automatisierung dieser Prozesse mit dem Projekt werden jährlich insgesamt 3402,3 Euro an überschüssigen Personallöhnen eingespart.

* + Hardware Voraussetzungen für Jenkins, und Minikube wurde aus offiziellen Internetseite ermittelt
  + Auf der Docker-Registry-Website konnte keine Empfehlung für die erforderlichen Hardware-Voraussetzungen gefunden werden.
  + Als Betriebssystem wird aufgrund der Eigenschaften wie Stabilität, Benutzerfreundlichkeit und weit verbreitenden Community-Unterstützung Ubuntu 20.04.LTS ausgewählt.
  + Es ist erforderlich, dass alle zu verwendete Virtuelle Maschinen sicher und isoliert miteinander kommunizieren können. Als beste Praktik werden sich die Maschinen in denselben LAN befinden. Wie im Abschnitt „Projektabgrenzung“ erklärt, werden nach der Installation des Betriebssystems „Ubuntu“ auf den Servern, werden statische IP-Adressen zugewiesen. Außerdem werden die Konfigurationen des im LAN verwendeten Switches gelten.

Realisierung

* Jenkins ist ein eigenständiger Open-Source-Automatisierung-Server,
* Für die Jenkins-Serverinstallation müssen wir Git, Docker, Docker-Compose, Open-JDK.11, Gradle auf dem Rechnern installieren.
* Da Push- und Pull-Verfahren mit GitHub Repo durchgeführt werden, installieren wir die Applikation „git“. Außerdem muss Gradle installiert werden, um den von GitHub abgerufenen Quellcode zu testen und ihn als Artefakt im .jar-Format zu packen. Docker muss installiert sein, um das gepackte Binär-Image zu erhalten. Da Jenkins eine Java-basierte Anwendung ist, benötigen wir auch eine Java-Laufzeit-Umgebung. Bei Bedarf wird vorzugsweise das Java Development Kit verwendet, um das mögliche Testen zu erleichtern. OpenJDK-11, das sowohl Open Source als auch einfach zu installieren ist, wird bevorzugt.
* Schreiben von Dockerfile:

Realisierung

* Installation benötigter Software

Docker Registry ist kostenlos, einfach zu installieren und verfügt über die Funktionalität zum Speichern von Docker-Image-Dateien. Aufgrund dieser Funktionen wird Docker-Registry als privates Repository verwendet, um genau zu steuern, wo Docker-Images gespeichert werden, und um die Verteilung von Docker-Images in Ihrer internen Entwicklungsumgebung sicherzustellen.

Die Docker-Registry-Anwendung wird als Multi-Container Applikation ausgeführt. Dazu müssen docker und docker-compose auf dem Server installiert werden.

Realisierung

* Kubernetes bietet automatisierte Funktionen zum Verwalten und Orchestrieren mehrerer Container.
* Installation des Minikube und andere benötigte Software:
  + - Ein Kubernetes-Cluster mit einem einzelnen Knoten reicht aus, um containerisierte Anwendungen in der Bereitstellungsumgebung im Projekt zu sehen. Minikube ist eine leichtgewichtige Kubernetes-Anwendung, die eine virtuelle Maschine auf Ihrer lokalen Maschine erstellt und ein einfaches Cluster mit einem einzigen Knoten bereitstellt. Und Minikube ist einfach zu installieren, verbraucht weniger Ressourcen, ist Open Source und kostenlos. Aus diesen Gründen wird Minikube zum Bereitstellen von Kubernetes-Clustern verwendet. Die IP-Adresse des Lieferungs-/Verteilungsservers lautet 192.168.22.134.

Kubectl ist das Terminalprogramm, das zum Verwalten von Minikube erforderlich ist.

- Schreiben von Deployment und Service Files

- Ein Objekttyp namens Deployment wird verwendet, um eine Anwendung in einer Kubernetes-Umgebung auszuführen und nach Bedarf zu skalieren, zurückzusetzen und bereitzustellen. Deployment ist ein API-Objekt in Kubernetes. Ein Service-Objekt wird auch verwendet, um die Anwendung im Internet zu veröffentlichen und zu übertragen. Für Kubernetes-Objekte wird eine deklarative Methode verwendet, um dem Benutzer die Bereitstellung und Verwaltung des Objekts zu erleichtern. Eine Datei im .yaml-Format wird erstellt und das Verteilungsobjekt und das Dienstobjekt werden in dieser Datei definiert.

Realisierung

* Pipeline
  + - Global Tool Konfiguration

Programme, die automatisch von Jenkins in der Pipeline verwendet werden sollen, müssen auf Jenkins im Abschnitt globale Toolkonfigurationen angegeben werden.

In diesem Zusammenhang sollte der Pfad zu Git, OpenJDK11, Gradle, Docker Executable eingegeben werden..

* + - Agent-Knoten Konfigurieren
    - Um Prozesse auf einem Jenkins-Server und anderen Knoten als Teil der CI/CD-Pipeline auszuführen, müssen diese Knoten natürlich als Agenten für Jenkins-Server definiert werden. Dazu müssen wir eine ssh-Verbindung zwischen Jenkins-Server und Agent-Knoten herstellen und die „Agent“-Datei vom Jenkins-Server auf den Agent-Knoten kopieren.
  + - Schreiben von Jenkinsfile

Wir klicken auf die Menüoption „Neues Element“, um sich bei Jenkins anzumelden. Wir geben den Namen der Jenkins-Pipeline ein und klicken auf Pipeline. Danach drücken wir die „OK“-Taste. Wir gelangen direkt zur Seite "Konfiguration" des Projekts, wählen den Abschnitt "Pipeline" aus und fügen wir den Code ein, die im Anhang A14 befinder Jenkinsfile steht.

Innerhalb der Pipeline kann es „stages“ eingegeben werden. Innerhalb der „stages“ kann es mehrere „stage elements“ geben. Innerhalb jeder „stage“ müssen „steps“ vorhanden sein. Es kann hilfreich sein, Werte im Verlauf der Pipeline anzuzeigen. Pipelines bestehen aus mehreren Schritten, mit denen wir Anwendungen erstellen, testen und bereitstellen können. Jenkins Pipeline macht es einfach, mehrere Schritte zu kombinieren, die verwendet werden können, um jeden Automatisierungsprozess zu modellieren. Wenn ein Schritt erfolgreich ist, wird der nächste Schritt gestartet. Pipeline schlägt fehl, wenn ein Schritt nicht korrekt ausgeführt wird. Wenn alle Schritte in der Pipeline erfolgreich abgeschlossen sind, erhalten Sie die Meldung, dass die Pipeline erfolgreich ausgeführt wird..

Quellcodebeispiel, das die Programmiersprache Java beschreibt, wurde dem GitHub-Repository entnommen.

Alle Stufen der Pipeline werden in Jenkinsfile deklariert und Jenkinsfile wird in Pipeline eingegeben.

Zum Ausführen der Pipeline wird die Jenkins-Konsole in der GUI von Broser verwendet. Sobald wird auf dem Knopf „Buildnow“ gedruckt, startet die Pipeline zu laufen.

Hier sehen wir die Stages.

Zunächst wird der Quellcode von GitHub übernommen.

Dann werden Docker-Image-Generierung und Tagging-Operationen durchgeführt.

Endlich wird die Applikation als Deployment-Objekt im Delivery-Server durchgeführt.

Hier können wir das laufende Programm sowohl auf dem GUI und als auch im Terminal sehen.

Fazit

- Soll-/Ist Vergleich

Die Projektziele wurden erreicht: alle Verfahren können (von GitHub pullen, Kompilieren, Testen, Bauen, Containerisieren und Bereitstellung ) automatisch durchgeführt werden.

Wie geplant, das Projekt konnte in 35 Stunden abgeschlossen werden. Dabei dauert Durchführungsphase eine Stunde mehr und Validierungsphase eine Stunde weniger.

Fazit

- Lessons-Learned

Im Rahmen dieses Projekts hat Herr Karakaya sowohl das Server-Setup als auch die Anwendung von Technologien wie Docker und Kubernetes praktisch ausgeübt.

Im Zuge dessen erlebte Herr Karakaya theoretisch und praktisch die ganz entscheidenden Vorteile, die das deklarative Verfahren und die Automatisierungsteknologien bieten,( sowohl in Bezug auf den Hardwareverbrauch als auch auf die Zeitersparnis.)

Fazit

- Ausblick

Das etablierte System kann sowohl in der Testumgebung als auch in der Stage- und Produktivumgebung eingesetzt

und mit anderen Technologien wie Vagrant, Ansible, Terraform, GitLab und Helm weiterentwickelt werden.