TABLE OF CONTENTS

PROBLEM STATEMENT

Modern software development practices require efficient and automated deployment processes to accelerate release cycles and enhance application reliability. However, manual deployment procedures, inconsistent infrastructure management and limited observability hinder the development and operational efficiency of Azure Kubernetes Service (AKS) workloads. Teams face challenges in maintaining a streamlined, consistent and automated deployment workflow while ensuring effective monitoring and observability for containerized applications running on AKS.

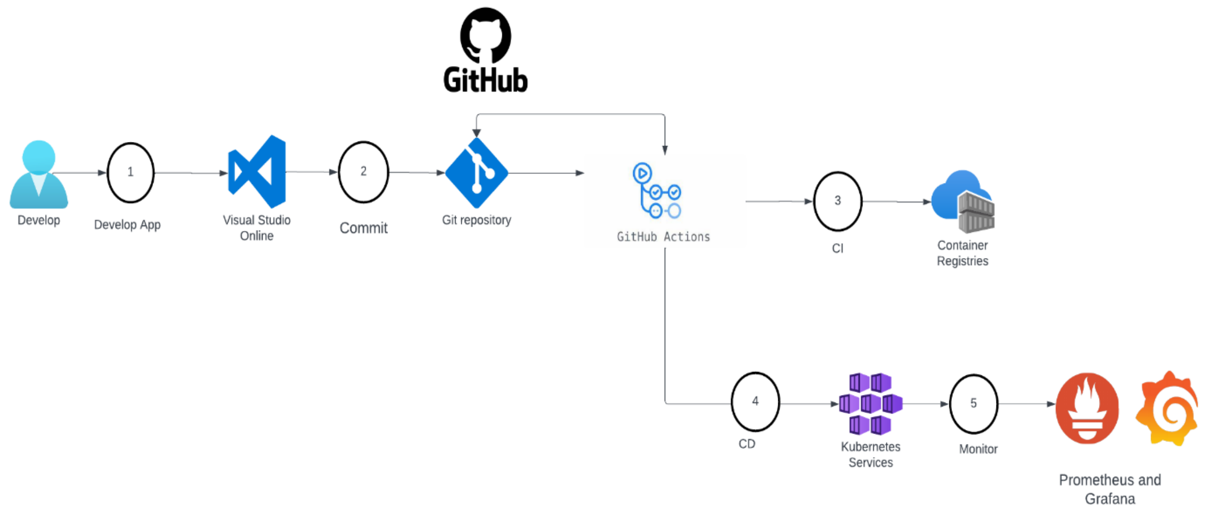
SOLUTION

Minfy has come up with a solution that involves implementing an end-to-end automated deployment workflow for Azure Kubernetes Service (AKS) using GitHub Actions, coupled with the integration of Prometheus and Grafana for monitoring and observability. This solution addresses the identified challenges and provides a streamlined, consistent and efficient approach to AKS deployment and management. When a change is pushed to the source code repository the Github action workflow gets executed. It logs in to Azure , builds the docker image , pushes to the Azure Container Registry (ACR) repo and deploys the application in AKS cluster.

- Using Infrastructure as Code principles to define AKS cluster configurations , Azure Container Repository and associated resources using tools like Azure Resource Manager (ARM) templates
- Source code stored in github repository facilitates version control for the application code whenever made changes.
- Github actions provides the automated execution of workflow run whenever a push or merge is happened on main branch.
- Azure Container Registry (ACR) stores the built docker images for the application.
- Azure Kubernetes Services (AKS) hosts the deployment of the applications done through the images stored in ACR

- Utilizing Helm charts to automate the deployment of Prometheus and Grafana to the AKS cluster.
- Configure Prometheus to scrape metrics from applications, providing comprehensive monitoring coverage.
- Configuring Grafana to connect to Prometheus as a data source and create customizable dashboards and develop Grafana dashboards tailored to the specific metrics and key performance indicators relevant to the AKS applications.

ARCHITECTURE



OUTCOME

Implementing Azure Kubernetes Service (AKS) deployment through GitHub Actions provides a streamlined and automated way to deploy and manage containerized applications on Azure Kubernetes Service. CI/CD(GitHub Actions) allows to automate the build, test and deployment processes for the applications. AKS enables a continuous integration/continuous deployment (CI/CD) pipeline to automatically deploy the containerized applications to the Kubernetes cluster whenever changes are pushed to the repository. Through github actions, every code change triggers the same set of actions, reducing the risk of configuration drift and ensuring that applications behave consistently in various stages of the deployment pipeline. As GitHub Actions is tightly integrated with GitHub repositories, using it for AKS deployment, we had leveraged version control features to track changes, rollbacks, and collaborate with team members effectively. Integrating Prometheus and Grafana into the AKS deployment provides powerful monitoring and observability. We can visualize key metrics, set up alerts, and troubleshoot issues more effectively. Prometheus allows to define alerting rules based on predefined thresholds. GitHub Actions is configured to notify or take corrective actions when alerts are triggered, enabling proactive issue identification and resolution.