

# Azure DevOps Capstone Project

## Product Management System

### Abstract

This project delivers a multi-region, production-grade infrastructure on Microsoft Azure with high availability, disaster recovery (DR), and robust CI/CD practices using Infrastructure-as-Code (IaC).

Pratheek U B  
289226

## Agenda

---

### 1. Project Introduction & Architecture

- Goals: High Availability, Disaster Recovery (DR), Scalability
- Multi-Region Strategy (Central Australia & Japan West)
- Traffic Manager for Global Routing

### 2. Infrastructure Provisioning (IaC)

- Terraform for Region 1
- Bicep for Region 2

### 3. Core Azure Resources

- AKS Cluster Setup with NGINX / Web App Routing
- Failover Group (FOG)
- Azure Container Registry (ACR)
- Azure Key Vault for Secret Management

### 4. CI/CD Automation (Azure DevOps)

- Trigger from GitHub
- Build: Frontend (React) & Backend (Spring Boot)
- Security Scans: SonarCloud, Snyk, Trivy
- Docker Build & Push to ACR
- Deployment to Primary & DR AKS Clusters

### 5. Disaster Recovery & Failover

- DR Region Setup via Bicep
- Traffic Manager Routing Policies
- Simultaneous Deployment to Both Clusters

### 6. Monitoring & Observability

- Prometheus for Metrics
- Grafana Dashboards (Node, Pod, API Server)

### 7. Security & Compliance

- NSGs
- Trivy, Snyk for vulnerability scanning tools
- Key Vault + Secret Rotation

### 8. Site Reliability Engineering (SRE) Practices

- SLA, SLO, SLI

## Project Overview

This project delivers a multi-region, production-grade infrastructure on Microsoft Azure with high availability, disaster recovery (DR), and robust CI/CD practices using Infrastructure-as-Code (IaC).

## Key Features:

**Global Traffic Management:** Azure Traffic Manager routes user requests to the most optimal regional deployment.

**Region 1** (Central Australia): Provisioned via Terraform.

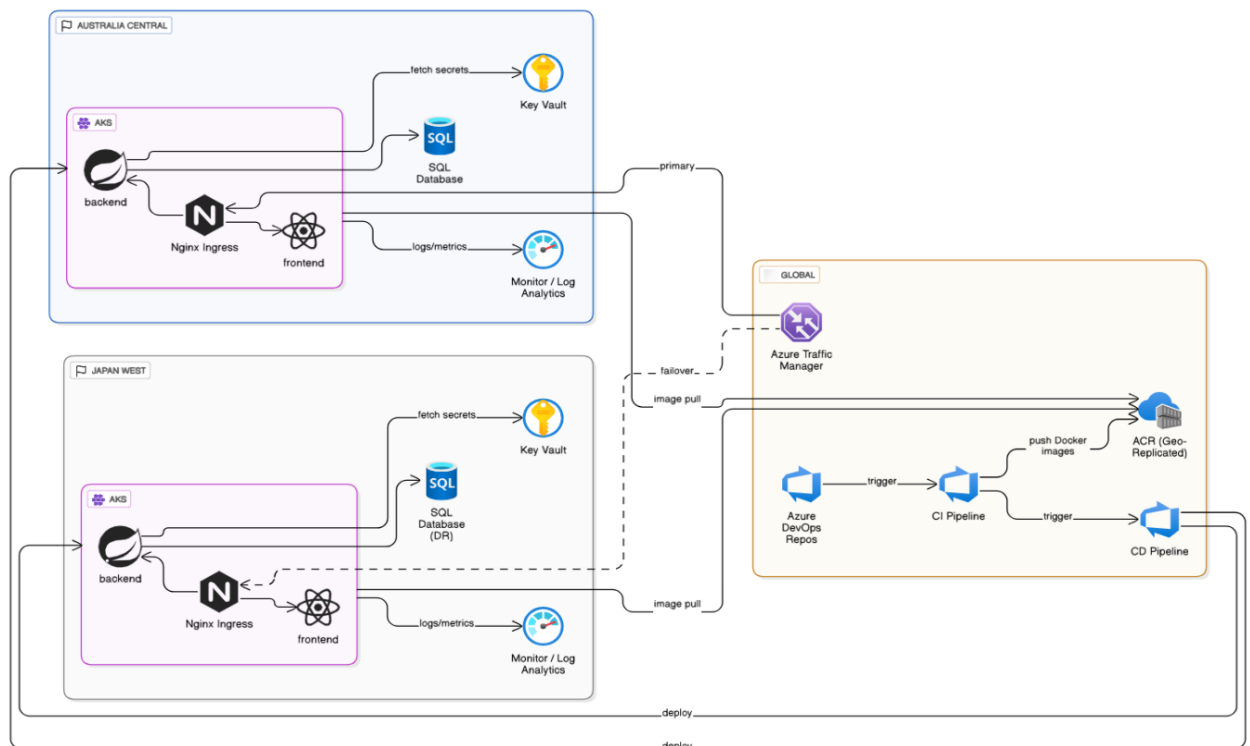
**Region 2** (Japan West).

**Kubernetes Deployments:** Frontend and backend apps deployed using AKS with different ingress strategies.

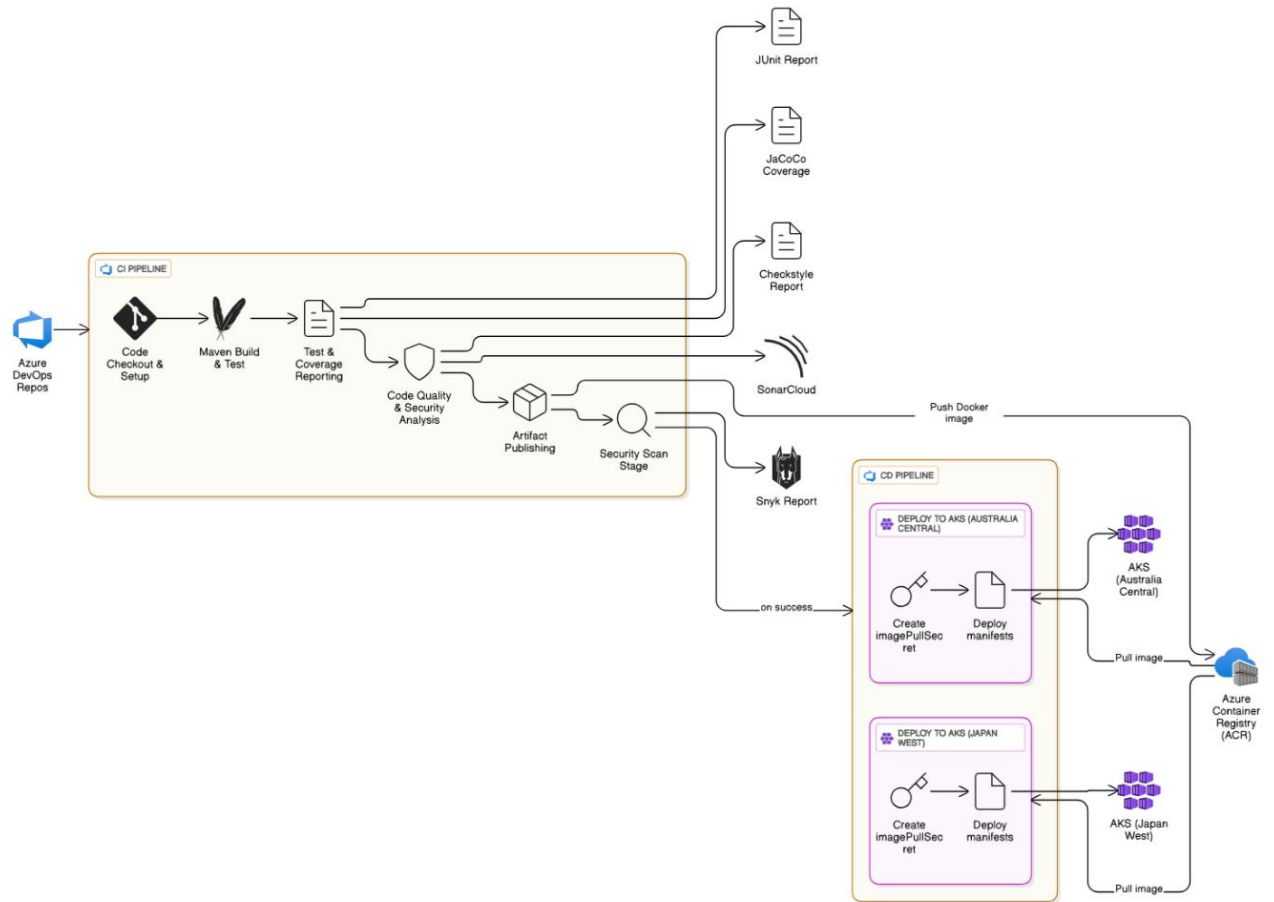
**CI/CD Pipelines:** Azure DevOps automates provisioning and deployments.

**Database Resiliency:** Azure SQL Failover Group ensures geo-redundancy.

## Architecture:



CI/CD:



## 1. Created a project in azure devops -

The screenshot shows the Azure DevOps interface for a project named 'Capstone'. The left sidebar contains navigation links: Capstone, Overview, Summary, Dashboards, Wiki, Boards, Repos, Pipelines, Test Plans, and Artifacts. The main content area has a search bar and a 'Private' label. Below the project name, there's a section 'About this project' with a description: 'This project delivers a multi-region, production-grade infrastructure on Microsoft Azure with high availability, disaster recovery (DR), and robust CI/CD practices using Infrastructure-as-Code (IaC)'. Below this is the 'Project stats' section, which shows 'Repos' with 0 pull requests opened and 'Pipelines' with a 31% success rate. The URL at the bottom is 'https://dev.azure.com/Pratheek008'.

## 2. created an agent pool

### a. projectsettings-> agent pools -> add pool

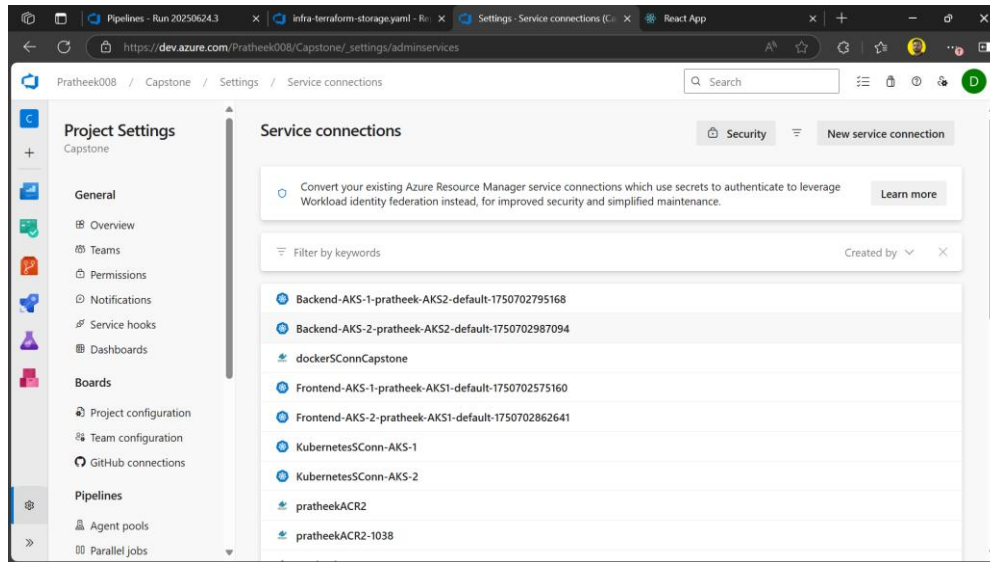
The screenshot shows the 'Agent pools' settings page in Azure DevOps. The left sidebar is expanded to 'Project Settings' > 'Agent pools'. The main content area shows the 'Default' agent pool. A table lists the agents:

Name	Last run	Current status	Agent version	Enabled
pratheekVM Online	Yesterday	Idle	4.258.1	<input checked="" type="checkbox"/> On

Buttons for 'Update all agents' and 'New agent' are visible at the top right. The URL at the bottom is 'https://dev.azure.com/Pratheek008'.

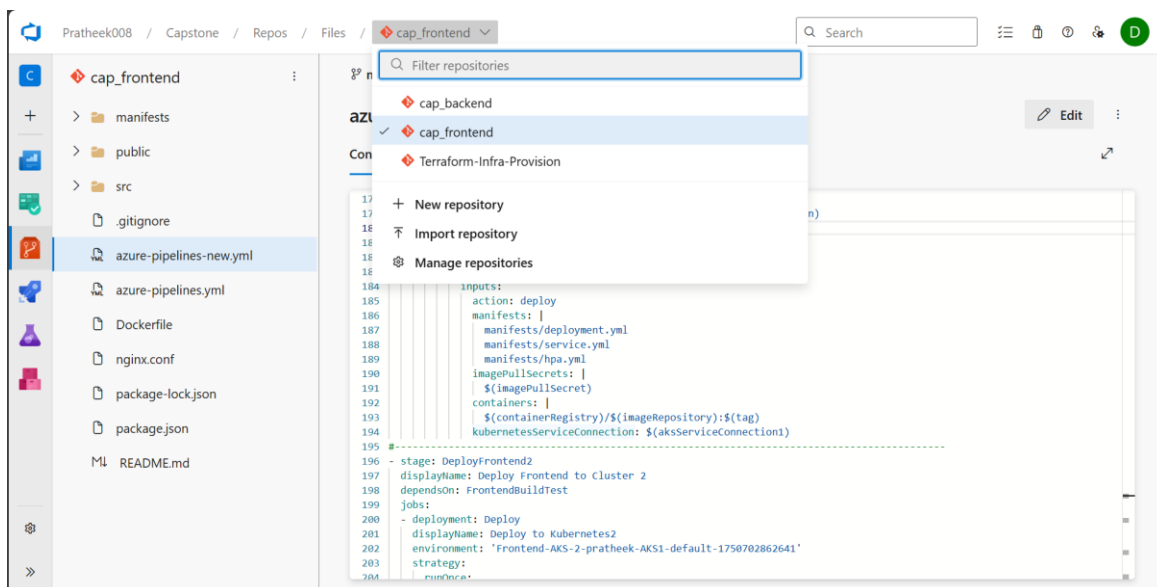
### 3. Service connections:

- a. Azure connection
- b. Docker registry connection (ACR)
- c. Kubernetes registry connection
- d. Sonarqube connection

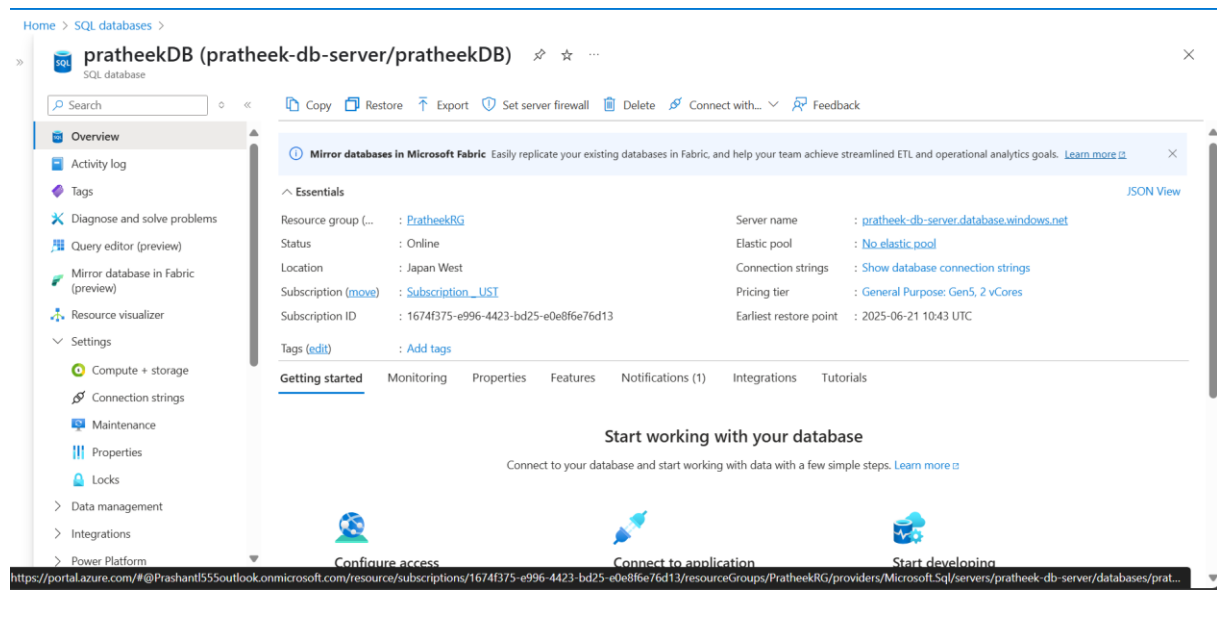


### 4. Different repository for each services in azure devops

- Cap\_frontend
- Cap\_backend
- Terraform-infra-provision



## 5. Created the database:



## 6. My infrastructure:

### Region 1: Central Australia (Provisioned via Terraform)

### Terraform Backend Configuration

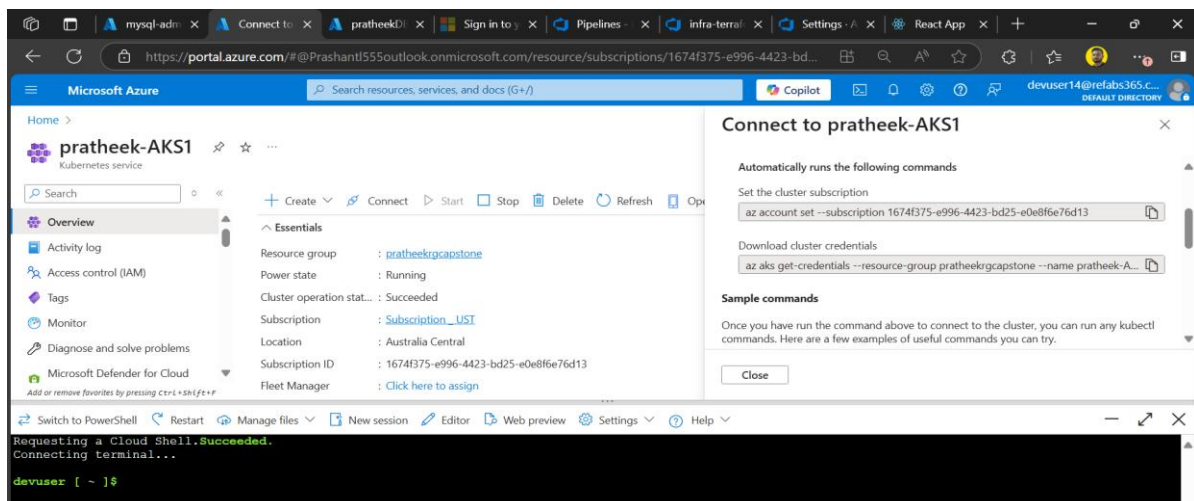
#### infra-terraform-storage.yaml

Contents History Compare Blame

```
1 trigger: none # Run manually only
2
3 variables:
4   azureServiceConnection: 'pratheekConnNew'
5
6 stages:
7   - stage: CreateTerraformBackend
8     displayName: "Create Terraform Backend Resources"
9     jobs:
10      - job: CreateBackend
11        displayName: "Provision Resource Group, Storage Account, and Container"
12        pool:
13          name: Default
14        steps:
15          - task: AzureCLI@2
16            displayName: "Create Resource Group"
17            inputs:
18              azureSubscription: $(azureServiceConnection)
19              scriptType: 'bash'
20              scriptLocation: 'inlineScript'
21              inlineScript: |
22                az group create \
23                  --name pratheekSTORAGE-rg \
24                  --location japanwest
25
26          - task: AzureCLI@2
27            displayName: "Create Storage Account"
28            inputs:
29              azureSubscription: $(azureServiceConnection)
30              scriptType: 'bash'
31              scriptLocation: 'inlineScript'
32              inlineScript: |
33                az storage account create \
34                  --name pratheekstorage0008 \
35                  --resource-group pratheekSTORAGE-rg \
36                  --location japanwest \
37                  --sku Standard_LRS
38
39          - task: AzureCLI@2
40            name: GetKey
41            displayName: "Get Storage Account Key"
42            inputs:
43              azureSubscription: $(azureServiceConnection)
44              scriptType: 'bash'
45              scriptLocation: 'inlineScript'
46              inlineScript: |
47                ACCOUNT_KEY=$(az storage account keys list \
48                  --resource-group pratheekSTORAGE-rg \
49                  --account-name pratheekstorage0008 \
50                  --query '[0].value' \
51                  --output tsv)
52                echo "##vso[task.setvariable variable=ACCOUNT_KEY;isecret=true]$ACCOUNT_KEY"
53
54          - task: AzureCLI@2
55            displayName: "Create Blob Container for Terraform State"
56            inputs:
57              azureSubscription: $(azureServiceConnection)
58              scriptType: 'bash'
59              scriptLocation: 'inlineScript'
60              inlineScript: |
61                az storage container create \
62                  --name tfstate \
63                  --account-name pratheekstorage0008 \
64                  --account-key $ACCOUNT_KEY
```

## Core Infrastructure Resources

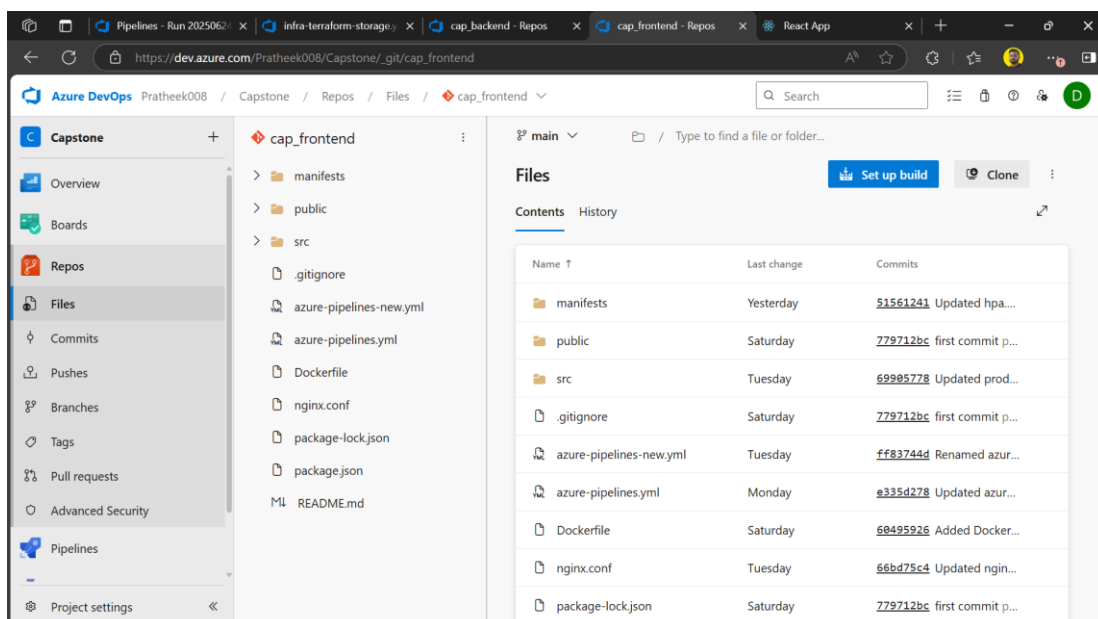
- **Resource Group:** pratheekRGcapstone
- **Virtual Network:** aks-vnet-1
  - **Subnet:** aks-subnet-1 (used by AKS)
- **Azure SQL**
  - **Server:** pratheekDB
  - **Database:** springbootdb
  - **Private Endpoint:** Enabled with Private DNS zone integration
- **AKS Cluster:** pratheekAKS1
  - **Node Pool:** 3 nodes (Standard\_D2s\_v3)
  - **Network Plugin:** Azure CNI
  - **Subnet Integration:** Connected to aks-subnet-1
- **Ingress Controller:** NGINX
  - Installed via Helm
  - Static Public IP provisioned using Terraform
- **Azure Container Registry (ACR):** pratheekACRcapstone
  - AcrPull role assigned to AKS Managed Identity (MSI)
- **Azure Traffic Manager**
  - **Profile Name:** pratheek-tm
  - **Routing Method:** Priority
  - **Endpoint:** Region 1 Ingress Public IP

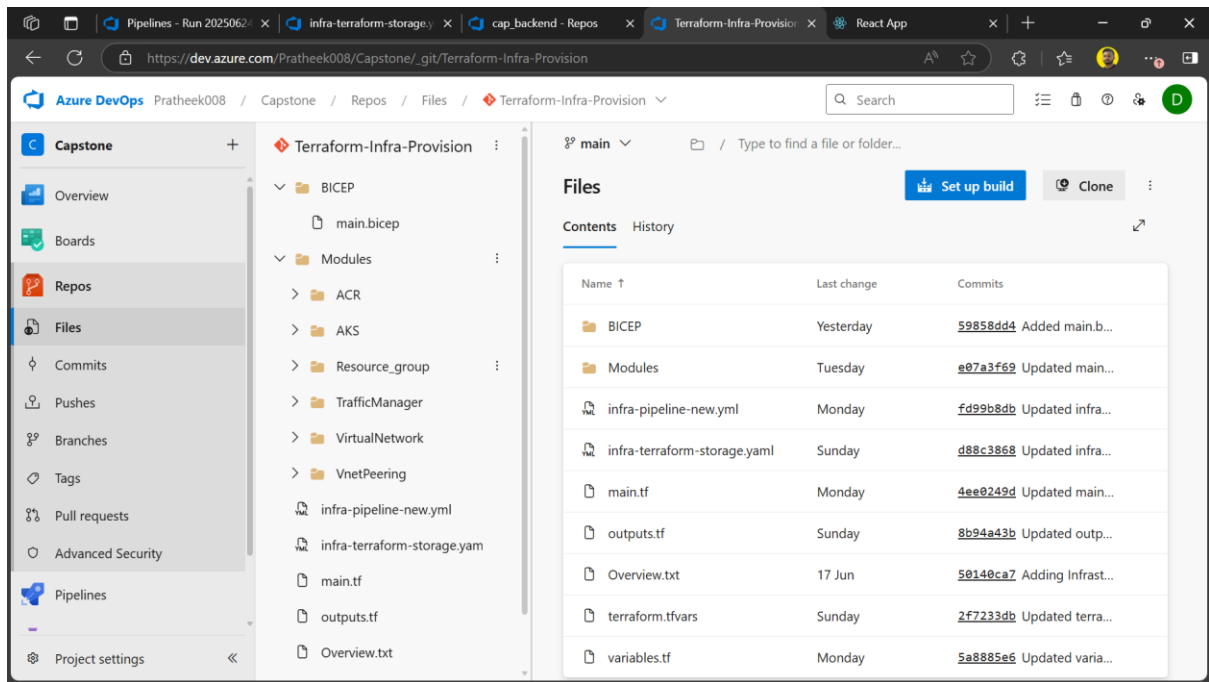




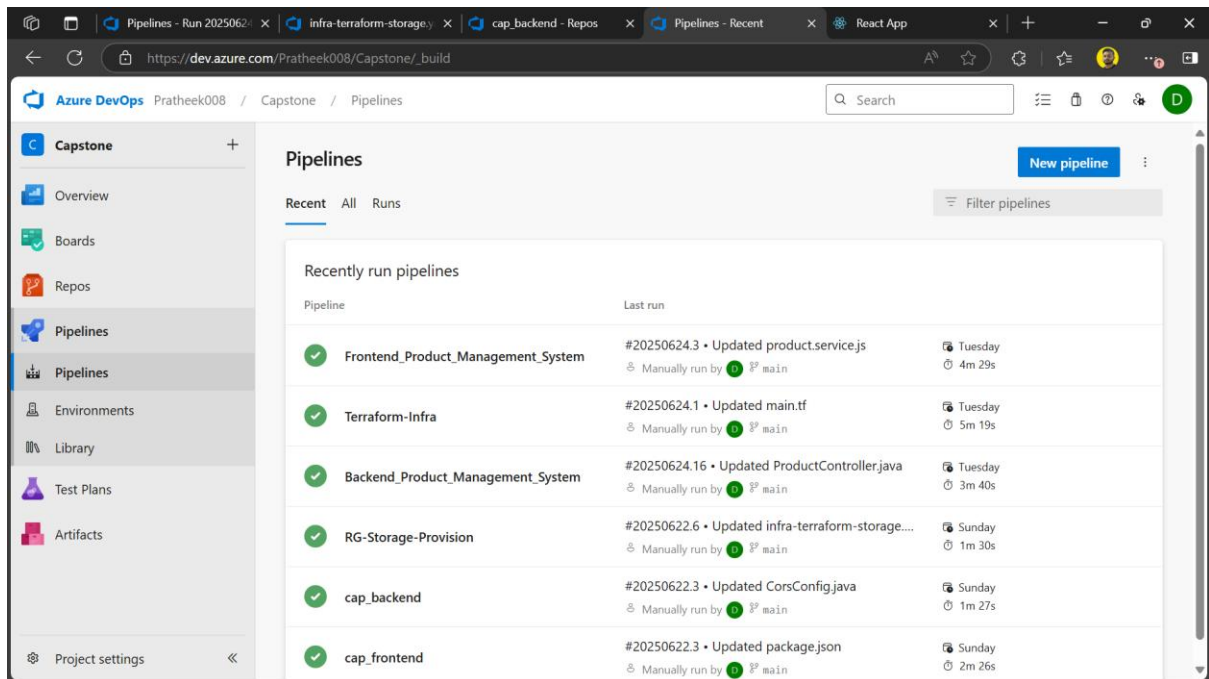
## Region 2: Japan West (Provisioned via Azure Bicep)

- **AKS Cluster**
  - **Name:** pratheekAKS2
  - **Region:** japanwest
  - **VNet Integration:** Connected to pratheek-vnet1
  - **RBAC:** Enabled
  - **Identity:** System-assigned managed identity
  - **Agent Pool:** systempool with 3 nodes
- **Azure SQL Database with Geo-Redundancy**
  - **Primary Region (Central Australia)**
    - SQL Server and Database
    - Private Endpoint configured
  - **Secondary Region (Japan West)**
    - Geo-replica (read-only)
    - **Failover Group (FOG):** Provides automatic regional failover
    - Uses public endpoint DNS for read access





## 7. Pipelines



## 8. Azure DevOps CI/CD Pipeline

### Trigger

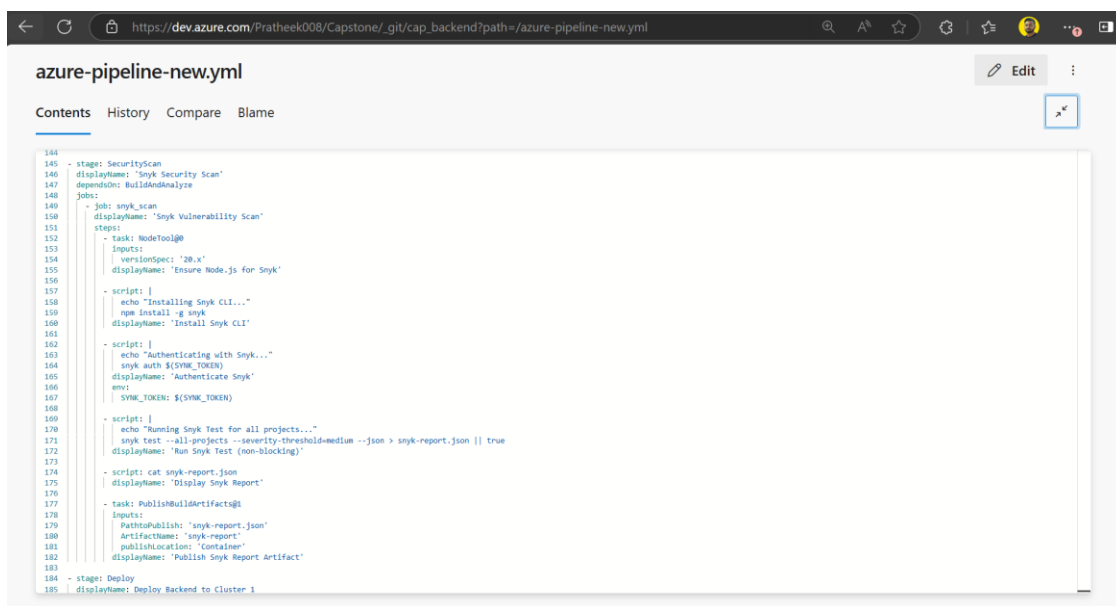
- Push to main branch on GitHub

### Setup

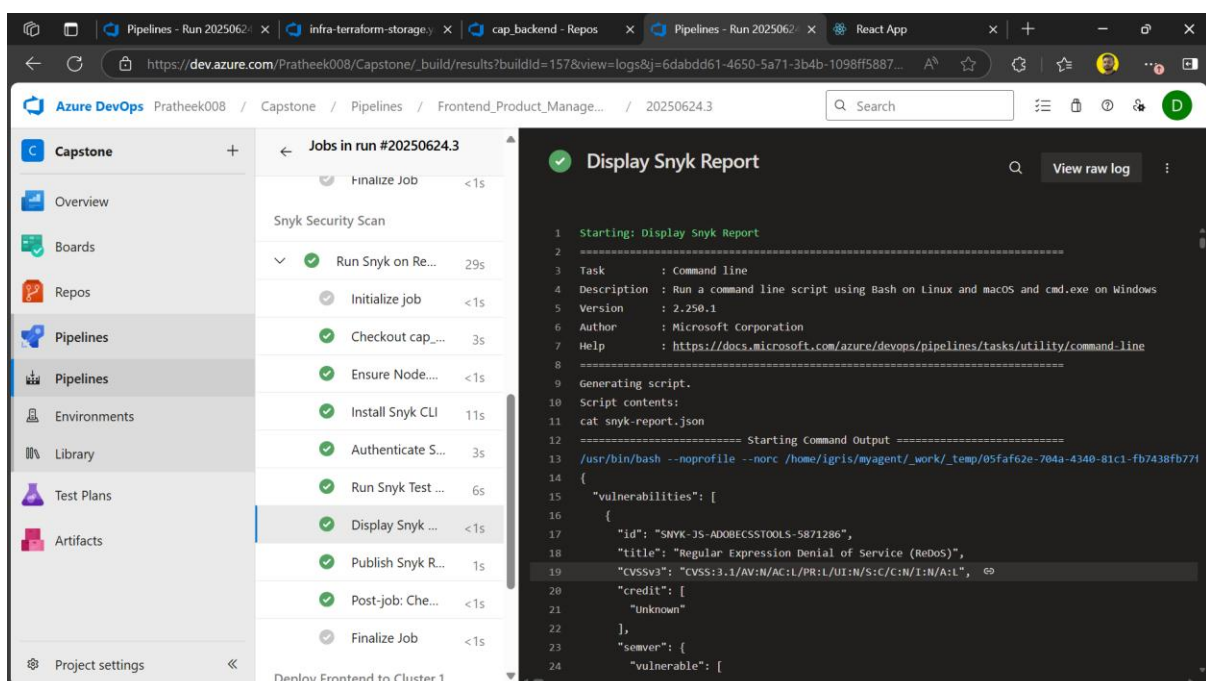
- Install Node.js and dependencies for React frontend

### Testing & Code Analysis

- **SonarCloud**: For static code analysis
- **Snyk**: For scanning NPM vulnerabilities



```
144 - stage: SecurityScan
145   displayName: 'Snyk Security Scan'
146   dependsOn: BuildAndAnalyze
147   jobs:
148     - job: snyk_scan
149       displayName: 'Snyk Vulnerability Scan'
150       steps:
151         - task: NodeTool@0
152           inputs:
153             versionSpec: '20.x'
154           displayName: 'Ensure Node.js for Snyk'
155         - script: |
156             echo "Installing Snyk CLI..."
157             npm install -g snyk
158             displayname: 'Install Snyk CLI'
159         - script: |
160             echo "Authenticating with Snyk..."
161             snyk auth $(SYNK_TOKEN)
162             displayname: 'Authenticate Snyk'
163             echo: |
164               SYNK_TOKEN: $(SYNK_TOKEN)
165         - script: |
166             echo "Running Snyk Test for all projects..."
167             snyk test --all-projects --severity-threshold=medium --json > snyk-report.json || true
168             displayname: 'Run Snyk Test (non-blocking)'
169         - script: cat snyk-report.json
170           displayName: 'Display Snyk Report'
171         - task: PublishBuildArtifacts@1
172           inputs:
173             PathToPublish: 'snyk-report.json'
174             ArtifactName: 'snyk-report'
175             publishLocation: 'Container'
176           displayName: 'Publish Snyk Report Artifact'
177 - stage: Deploy
178   displayName: 'Deploy Backend to Cluster 1'
```



**Azure DevOps** Pratheek008 / Capstone / Pipelines / Frontend\_Product\_Manage... / 20250624.3

**Jobs in run #20250624.3**

Job	Status	Duration
Finalize Job	Completed	<1s
Snyk Security Scan		
Run Snyk on Re...	Completed	29s
Initialize job	Completed	<1s
Checkout cap...	Completed	3s
Ensure Node...	Completed	<1s
Install Snyk CLI	Completed	11s
Authenticate S...	Completed	3s
Run Snyk Test ...	Completed	6s
Display Snyk ...	Completed	<1s
Publish Snyk R...	Completed	1s
Post-job: Che...	Completed	<1s
Finalize Job	Completed	<1s

**Display Snyk Report** View raw log

```
1 Starting: Display Snyk Report
2 =====
3 Task      : Command line
4 Description : Run a command line script using Bash on Linux and macOS and cmd.exe on Windows
5 Version    : 2.250.1
6 Author     : Microsoft Corporation
7 Help       : https://docs.microsoft.com/azure/devops/pipelines/tasks/utility/command-line
8 =====
9 Generating script.
10 Script contents:
11 cat snyk-report.json
12 ===== Starting Command Output =====
13 /usr/bin/bash --noprofile --norc /home/igrls/myagent/_work/_temp/05f462e-704a-4340-81c1-fb7438fb771
14 {
15   "vulnerabilities": [
16     {
17       "id": "SNYK-JS-ADOBECSSTOOLS-5871286",
18       "title": "Regular Expression Denial of Service (ReDoS)",
19       "CVSSv3": "CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:N/I:N/A:L",
20       "credit": [
21         "unknown"
22       ],
23       "severer": {
24         "vulnerable": [
```

## Build Stage

- Compile Spring Boot backend using Maven
- Run unit tests and calculate code coverage
- Perform **Snyk** and **Trivy** scans for vulnerabilities
- Build Docker images and push to **ACR**

The screenshot shows the Azure DevOps Pipelines interface for a project named 'Capstone'. The left sidebar contains navigation links: Overview, Boards, Repos, Pipelines, Environments, Library, Test Plans, and Artifacts. The main area displays a list of jobs in run #20250624.3. The 'Run Trivy Scan' job is highlighted, showing a duration of 5s. The right pane shows the job's output, including a table of vulnerabilities found in the 'libxml2' library.

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version
libxml2	CVE-2025-32414	HIGH	fixed	2.13.4-r5	2.13.4-r6
libxml2	CVE-2025-32415	HIGH	fixed	2.13.4-r5	2.13.4-r6

## 9. Deployment

### Primary Region (pratheeAKS1)

- Retrieve SQL credentials from **Azure Key Vault**
- Create Kubernetes secrets for sensitive data
- Deploy backend and frontend using **Helm/manifests**

The screenshot shows the Azure DevOps Pipelines interface for a project named 'Capstone'. The left sidebar contains navigation links: Overview, Boards, Repos, Pipelines, Environments, Library, Test Plans, and Artifacts. The main area displays a list of jobs in run #20250624.16. The 'Build and Analyze Spring Boot App' job is highlighted, showing a duration of 2m 0s. The right pane shows the job's output, including a table of vulnerabilities found in the 'libxml2' library.

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version
libxml2	CVE-2025-32414	HIGH	fixed	2.13.4-r5	2.13.4-r6
libxml2	CVE-2025-32415	HIGH	fixed	2.13.4-r5	2.13.4-r6

Azure DevOps interface showing the 'Environments' tab for build #20250624.16. The page displays two environments: Backend-AKS-1 and Backend-AKS-2. Backend-AKS-1 has a job 'Deploy Backend to AKS Cluster 1' with a duration of 16s. Backend-AKS-2 has a job 'Deploy Backend to AKS Cluster 2' with a duration of 10s. The left sidebar shows the project structure with 'Pipelines' selected. The top navigation bar includes 'Capstone', 'Overview', 'Boards', 'Repos', 'Pipelines', 'Environments', 'Library', 'Test Plans', and 'Artifacts'.

## Backup Region (pratheekAKS2)

- Use capstone\_backup agent pool
- Connect to pratheekAKS2 cluster
- Reuse the same Docker image and manifests for deployment

Azure DevOps interface showing the 'Jobs in run #20250624.3' tab. The page displays a list of jobs: Deploy to Kubernetes cluster2, Initialize job, Create imageP..., Deploy to Kub..., and Finalize Job. The 'Deploy to Kubernetes cluster2' job is selected, and its details are shown on the right. The details include the task 'Deploy to Kubernetes', the description 'Use Kubernetes manifest files to deploy to clusters or even bake the manifest files', and the version '0.246.3'. The logs show the deployment process, including downloading the Kubernetes release, applying the manifest, and rolling out the deployment.

This screenshot shows the 'Build, Test, Analyze R...' stage of a pipeline. It includes a summary of stages and jobs, with a 'Snyk Security Scan' job showing 1 artifact.

Stage	Job	Status	Duration
Build, Test, Analyze R...	1 job completed	Completed	2m 50s
Snyk Security Scan	1 job completed	Completed	38s
Deploy Frontend to C...	1 job completed	Completed	14s
Deploy Frontend to C...	1 job completed	Completed	10s

This screenshot shows the 'Tests' tab of a pipeline. It displays a summary of test results, including a 'Hooray! There are no test failures.' message.

Summary
1 Run(s) Completed (1 Passed, 0 Failed)
Total tests: 1 (1 Passed, 0 Failed, 0 Others)
Pass percentage: 100%
Run duration: 12s 533ms
Tests not reported: 0

This screenshot shows the 'Analytics' tab of a pipeline. It displays three metrics: Pipeline pass rate (100%), Test pass rate (No test runs completed in the last 14 days), and Pipeline duration (7m 18s).

Metric	Value
Pipeline pass rate	100%
Test pass rate	No test runs completed in the last 14 days
Pipeline duration	7m 18s

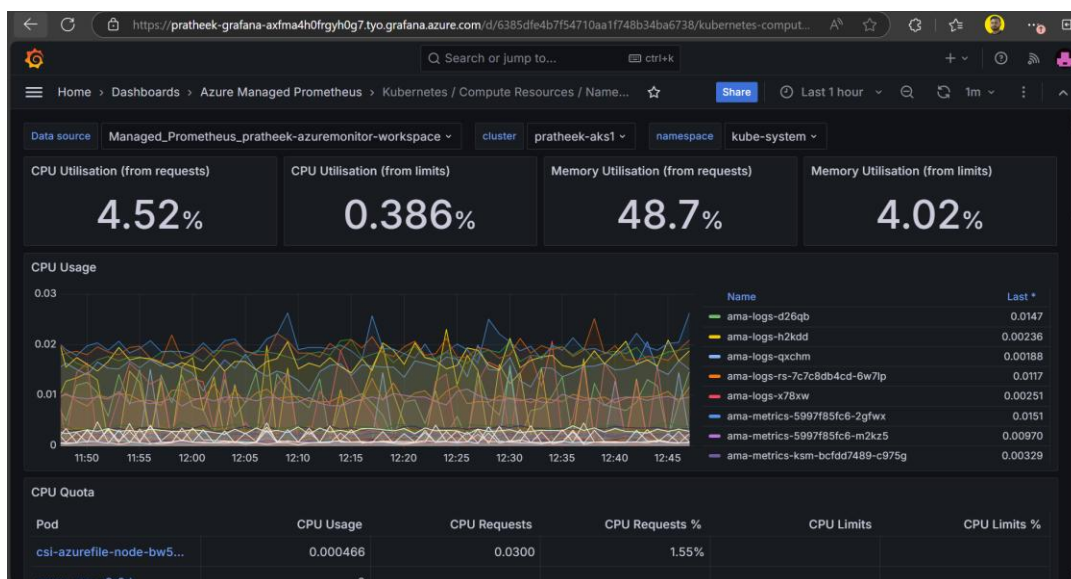
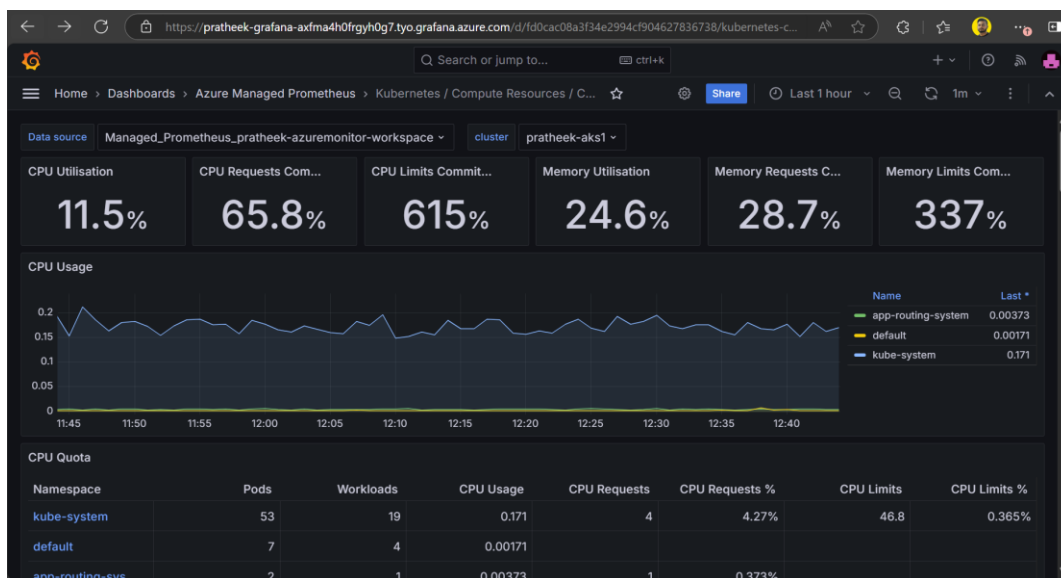
## 10. Monitoring Setup (Both AKS Clusters)

- **Prometheus:** Metrics collection and alerting
- **Grafana:** Visualization dashboards

Configured Grafana for aks1

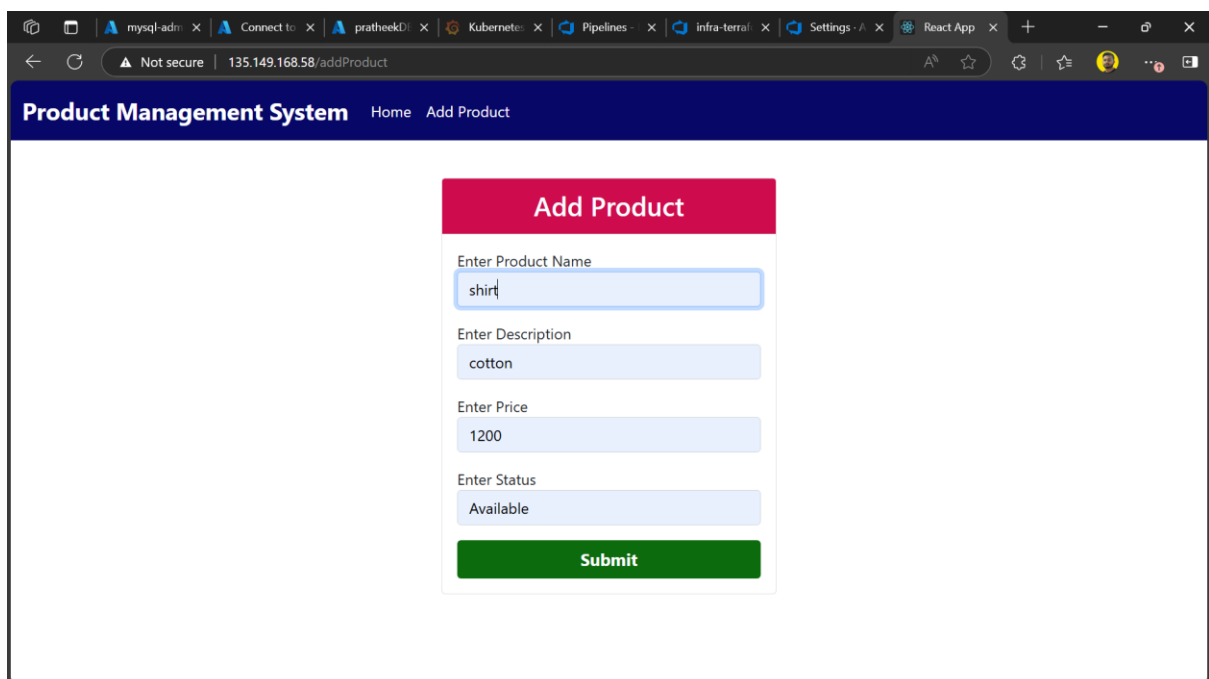
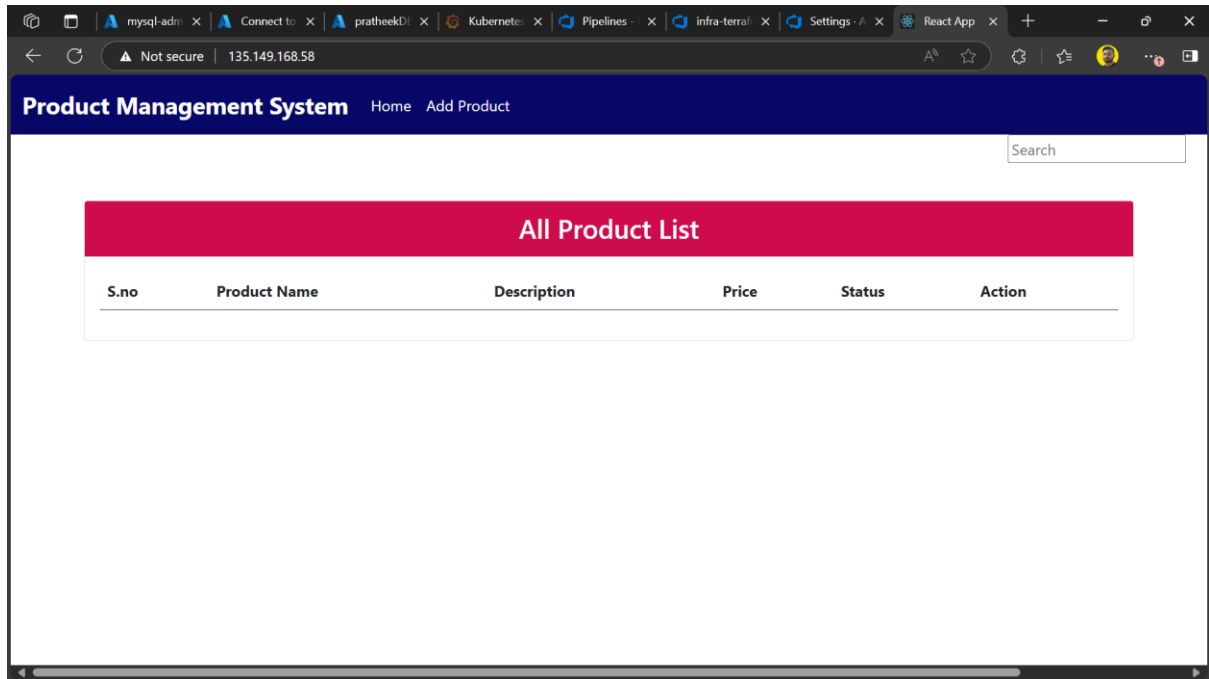
Steps:

- Azure Managed Grafana
- Azure Monitor workspaces (configure Grafana with workspace)
- Kubernetes services (configure workspace with k8s service)
- [Grafana](#)



## 11. Accessing application:

Testing failover using browser or curl after simulating failure in one region.so using cluster 2 in another region:





mysql: xConnect xpratheek: xSign in t: xPipeline: xinfra-ter: xSetting: xReact A: xHTML b: x

Not secure | 135.149.168.58

Product Management SystemHomeAdd Product

Search

All Product List

S.no	Product Name	Description	Price	Status	Action
1	Shirt	Cotton	1200	Available	<div>EditDelete</div>

## 12. Conclusion

This Capstone project showcases a comprehensive, real-world implementation of DevOps and Cloud Engineering principles on the Azure platform. The solution follows industry best practices across the entire software delivery lifecycle — from infrastructure provisioning to deployment, security, and observability.

Through this project, the following outcomes were achieved:

- **High Availability & Resilience:**  
Multi-region deployment of Azure Kubernetes Service (AKS) clusters with Active-Passive failover using Azure Traffic Manager ensures business continuity and disaster recovery readiness.
- **Scalable & Modular Architecture:**  
Microservices are containerized and deployed on Kubernetes, allowing horizontal scaling and easier maintainability.
- **Automated Infrastructure Provisioning:**  
Terraform is used to define and manage cloud infrastructure as code, enabling repeatable, version-controlled deployments across environments.
- **Robust CI/CD Pipelines:**  
Azure DevOps Pipelines automate build, test, security scanning, and multi-region deployment, enabling faster, safer delivery of application changes.
- **Security & Compliance:**  
Azure Defender, Key Vault, NSGs, and Azure Policy enforce strong security postures, including secret management, image scanning, and port hardening.
- **Comprehensive Monitoring & Logging:**  
Application Insights and Azure Monitor deliver full-stack observability — capturing logs, metrics, dependencies, crash analytics, and SLO breaches.

**Github:**

[pratheek08/Capstone-Frontend](#)

[pratheek08/Capstone-Backend](#)

[pratheek08/Capstone-Infra](#)