Deploying Dockerized Applications on Azure VMs

# Final Project

Name: Lavi Singodiya

Domain: DevOps

Student Id: CT\_CSI\_DV\_554

Contact No: 8769045359

Email ID: [lavisaini322@gmail.com](mailto:lavisaini322@gmail.com)

# 

# Introduction

Project Overview  
- Deploying a Dockerized application on a VM to improve deployment efficiency.  
- Focus on containerization, Docker installation, image pulling, container launching, and networking configuration.

# Business Problem Statement

Challenges in the Insurance Industry  
- Fragmented deployment procedures across VM infrastructure.  
- Hindered scalability and operational complexities.  
- Potential service delivery and regulatory compliance issues.  
  
Objective  
- Adopt Docker containerization to standardize deployment practices.  
- Enhance application management and optimize resource allocation.  
- Streamline deployment processes, improve scalability, and ensure regulatory adherence.

# Prerequisites

Requirements  
- Azure Virtual Machine.  
- Docker Engine installed on the VM.

# High-Level Architecture

Architecture Diagram  
- Developer interaction with Virtual Machine.  
- Applications running within Docker containers.  
- User access to applications through server requests and responses.  
  
Components  
- Docker on Azure VM.  
- Applications 1 & 2 within Docker containers.  
- User interactions and networking.

# Docker Container Deployment on Azure VM

Steps  
- Install Docker: SSH into your VM and install Docker using the appropriate package manager.  
- Pull Docker Image: Use the `docker pull` command to pull the Docker image of your application from a registry like Docker Hub.  
- Run Docker Container: Launch a Docker container using the `docker run` command, specifying necessary options like port mappings or environment variables.  
- Access Your Application: Access the Dockerized application through the specified port or endpoint, ensuring proper networking configurations.  
- Monitoring and Management: Use Docker commands (`docker ps`, `docker logs`, etc.) to monitor and manage running containers.

# Benefits of Dockerization

Standardized Deployment Practices  
- Consistent environment across development, testing, and production.  
  
Enhanced Application Management  
- Simplified updates and rollbacks.  
- Efficient resource utilization.  
  
Optimized Resource Allocation  
- Improved scalability and performance.  
  
Regulatory Compliance  
- Ensured adherence to industry standards and regulations.

# Conclusion

Summary  
- Docker containerization on Azure VM improves deployment efficiency and management.  
- Streamlined processes and enhanced scalability.  
- Better resource utilization and regulatory compliance.  
  
Future Work  
- Explore advanced Docker features (e.g., Docker Compose, Swarm).  
- Implement CI/CD pipelines for automated deployments.