# Student Name : Ayandah Dube

Pin : P1882196 T

Program : BSEH

Module : BSEH 480

# CHAPTER 1: Research Proposal

## ABSTRACT

CI/CD, which stands for continuous integration (CI) and continuous delivery (CD), creates a faster and more precise way of combining the work of different people into one cohesive product. In application development and operations (DevOps), CI/CD streamlines application coding, testing and deployment by giving teams a single repository for storing work and automation tools to consistently combine and test the code to ensure it works.

## BACKGROUND

The continuous integration/continuous delivery (CI/CD) pipeline is an agile DevOps workflow focused on a frequent and reliable software delivery process. The methodology is iterative, rather than linear, which allows DevOps teams to write code, integrate it, run tests, deliver releases and deploy changes to the software collaboratively and in real-time.

A key characteristic of the CI/CD pipeline is the use of automation to ensure code quality. As the software changes progress through the pipeline, test automation is used to identify dependencies and other issues earlier, push code changes to different environments and deliver applications to production environments. Here, the automation’s job is to perform quality control, assessing everything from performance to API usage and security. This ensures the changes made by all team members are integrated comprehensively and perform as intended.

The ability to automate various phases of the CI/CD pipeline helps development teams improve quality, work faster and improve other DevOps metrics.

## PROBLEM STATEMENT

The unavailability of a DevOps CI/CD pipeline framework that automates the software delivery process by using continuous integration, testing, delivery, and deployment practices for a technology company.

## PROJECT AIM

This setup is going to take me through a DevOps journey using Azure DevOps. From setting up a pipeline to deploying an application to Azure Kubernetes cluster!

## OBJECTIVES

* Initial setup of Azure DevOps to begin deploying to Azure using Pipelines as code
* Deploy Azure resources using Terraform modules
* Deploy a test application to Azure Kubernetes Service
* Perform CI/CD with automated application deployments
* Test your deployed Azure resources using automated testing
* Reviewing monitoring and alerting using Application & Container Insights

## MATERIALS

The major materials used in this Study research will be MICROSOFT AZURE Platform

## METHODOLOGY

## PROPOSED TIME ALLOCATION

This project will take 8 Weeks to complete as each stage will require a week to complete and deploy