# MLOps using Azure Devops to Deploy a Classification Model

## Overview

Most of the time, we develop and train the machine learning models in our local Python notebook and hand off the resultant code to an app developer, who must then integrate it into an application and deploy it. Often, the bugs and the performance issue go unnoticed until the code has already been deployed, and it then becomes a cumbersome and frustrating task to resolve the issue. So, to tackle this problem, Azure DevOps came into the picture. You can build Machine learning apps faster through Azure DevOps. The whole process now becomes a part of the Continuous Integration (CI) and Continuous Delivery (CD) pipeline of the Azure DevOps. You can continue to write and train models in your favorite Python environment and rest assured that it is your Python code that is deployed to production without worrying about any conflicts between the app and the model.

This project is focused on how to build the Continuous Integration and Continuous Delivery pipelines using the model developed in Deep Learning Project for Beginners with Source Code Part 1 with Azure DevOps.

## Aim

- To understand Azure DevOps
- To build a classification model which will predict the customer's license status
- To deploy the license status classification model in a scalable way through Azure DevOps

## Data Description

The dataset has information about the customers whose license status, i.e., issued, renewed, canceled, is to be predicted.

## Tech Stack

- ➔ Language: Python
- ➔ Libraries: pandas, TensorFlow, keras, matplotlib,sci-kit learn, numPy, fastapi, unicorn, requests, json, h2o, seaborn

**Prerequisites**

- Docker
- GIT
- Deep Learning Project for Beginners with Source Code Part 1

**Steps to deploy**

1. Clone azure repository
    a. Create a new project after logging to your azure account
    b. Clone the git repository from the HTTP link in repos section and push the necessary changes as per requirement
    c. You can also clone the azure repository through vs-code by clicking on the clone in vs-code and push all the file through it.

2. Define docker file for your project

3. Create your azure resource group
    a. You can create your azure resource group by the portal of azure group and simply create your resource by clicking on the create option by giving it a proper name

4. Create azure container registry to host the docker image
    a. Go to the same resource group you have created and add Azure container Registry to your resource group

5. Define azure pipeline through YAML
    a. Go to your project after logging into your Azure DevOps account
    b. Create your pipeline
    c. Configure your pipeline through docker
    d. Save and run the generated YAML file

6. Customize the YAML file as per your requirement

7. Deploy the Docker image to Azure Web App
    a. Go to the Azure container registry you have created
    b. To deploy the app, add the Web App resource to your resource group
    c. Create your web app by properly naming it
    d. Once you have review and created the web app, your image will be deployed

**Takeaways**

1. What is DevOps?
2. Benefits of DevOps
3. What is Azure DevOps?
4. Features and services of Azure DevOps
5. What are Azure repos?
6. How to clone Azure repos?
7. What is FastAPI?
8. Features of FastAPI
9. What is docker?
10. How to define a docker file?
11. How to create an Azure resource group?
12. How to create an Azure container registry?
13. What are Azure pipelines?
14. Features of Azure pipelines
15. How to configure pipeline using YAML?
16. How to deploy the pipeline on the Azure web application?