Course outline

Module 1: Implementing Continuous Integration in an Azure DevOps Pipeline

In this module, you'll be introduced to continuous integration principles, including: benefits, challenges, build best practices, and implementation steps. You will also learn about implementing a build strategy with workflows, triggers, agents, and tools.

Lessons

- Continuous Integration Overview
- Implementing a Build Strategy

Lab : Enabling Continuous Integration with Azure Pipelines
Lab : Creating a Jenkins Build Job and Triggering CI

After completing this module, students will:

- Be able to explain why continuous integration matters
- Implement continuous integration using Azure DevOps

Module 2: Managing Code Quality and Security Policies

In this module, you will learn how to manage code quality, including: technical debt, SonarCloud, and other tooling solutions. You will also learn how to manage security policies with open source, OWASP, and WhiteSource Bolt.

Lessons

- Managing Code Quality
- Managing Security Policies

Lab : Managing Technical Debt with Azure DevOps and SonarCloud
Lab : Checking Vulnerabilities using WhiteSource Bolt and Azure DevOps

After completing this module, students will be able to:

- Manage code quality including: technical debt SonarCloud, and other tooling solutions.
- Manage security policies with open source, OWASP, and WhiteSource Bolt.
- Manage code quality including: technical debt, SonarCloud, and other tooling solutions.

Module 3: Implementing a Container Build Strategy

In this module, you will learn how to implement a container strategy including how containers are different from virtual machines and how microservices use containers. You will also learn how to implement containers using Docker.

Lessons

- Implementing a Container Build Strategy

Lab : Existing .NET Applications with Azure and Docker Images

After completing this module, students will be able to:

- Implement a container strategy including how containers are different from virtual machines and how microservices use containers.
- Implement containers using Docker.

Course outline

Module 1: Design a Release Strategy
Lessons

- Introduction to Continuous Delivery
- Release strategy recommendations
- Building a High Quality Release pipeline
- Choosing a deployment pattern
- Choosing the right release management tool

Lab : Building a release strategy
After completing this module, students will be able to:

- Differentiate between a release and a deployment
- Define the components of a release pipeline
- Explain things to consider when designing your release strategy
- Classify a release versus a release process and outline how to control the quality of both
- Describe the principle of release gates and how to deal with release notes and documentation
- Explain deployment patterns, both in the traditional sense and in the modern sense
- Choose a release management tool

Module 2: Set up a Release Management Workflow
Lessons

- Create a Release Pipeline
- Provision and Configure Environments
- Manage and Modularize Tasks and Templates
- Integrate Secrets with the release pipeline
- Configure Automated Integration and Functional Test Automation
- Automate Inspection of Health

Lab : Automating your infrastructure deployments in the Cloud with Terraform and Azure Pipelines
Lab : Setting up secrets in the pipeline with Azure Key vault
Lab : Setting up and Running Load Tests
Lab : Setting up and Running Functional Tests
Lab : Using Azure Monitor as release gate
Lab : Creating a Release Dashboard

After completing this module, students will be able to:

- Explain the terminology used in Azure DevOps and other Release Management Tooling
- Describe what a Build and Release task is, what it can do, and some available deployment tasks
- Classify an Agent, Agent Queue, and Agent Pool
- Explain why you sometimes need multiple release jobs in one release pipeline
- Differentiate between multi-agent and multi-configuration release job
- Use release variables and stage variables in your release pipeline
- Deploy to an environment securely using a service connection
- Embed testing in the pipeline
- List the different ways to inspect the health of your pipeline and release by using alerts, service hooks, and reports
- Create a release gate

Module 3: Implement an appropriate deployment pattern

Lessons

- Introduction to Deployment Patterns
- Implement Blue Green Deployment
- Feature Toggles
- Canary Releases
- Dark Launching
- AB Testing
- Progressive Exposure Deployment

Lab : Blue-Green Deployments

Lab : Traffic Manager

After completing this module, students will be able to:

- Describe deployment patterns
- Implement Blue Green Deployment
- Implement Canary Release
- Implement Progressive Exposure Deployment