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.