Course #: Cloud DevOps

**course description & Syllabus**

Name of College or University

Semester 2020

# Overview

**Title:** Cloud DevOps
**Credits/Units:** 6-credit hours
**Pre-requisites:** Cloud Administrator

**Institution LMS access:** [https://lms.univ-or-college.edu](https://canvas.cmu.edu)

**Recitation:**

1. **Tuesday, 8:00 AM – 8:50 AM ET, GHC 4307 (Videotaped)**

**Teaching Staff:**

|  |  |
| --- | --- |
| [**Prof. Majd Sakr**](http://www.cs.cmu.edu/~msakr)msakr@cs.cmu.eduOffice Number, +1-412-268-1161*Office hours:*Tuesday, 3-4pm (time zone) |  |
| TAs typically hold office hours in the GHC 5th floor. The TA office hours are posted on the LMS:* TA Name<ta-email@xyz.edu>
 | * TA Name<ta-email@xyz.edu>
 |

# Course Description

Students will gain knowledge and develop hands-on experience solving real-world problems in the area of Cloud DevOps. Cloud DevOps combines people, processes, and technologies in order to increase software delivery velocity and improve service reliability. Cloud DevOps connects formerly siloed roles—development, IT operations, quality engineering, and security. Students will design and implement strategies for application and infrastructure that enable continuous integration, continuous testing, continuous delivery, infrastructure as code as well as monitoring. All projects utilize existing public cloud infrastructure, tools, and services. Students will leverage cloud technologies to design and implement DevOps solutions to version control, building, testing, release, provisioning, configuration, deployment, and monitoring. It is our goal that students will develop the skills needed to become a cloud DevOps engineer.

# Course Goals

At the conclusion of the course, the successful student should be able to:

1. Manage source control, adopt source control strategy and integrate source control with Continuous Integration/Continuous Delivery pipelines
2. Identify and implement appropriate end-to-end CI/CD solution to deploy cloud-based applications with various deployment models (e.g., VM-hosted, containerized microservices, serverless)
3. Integrate monitoring solutions, identify metrics and implement alerts for cloud-based applications
4. Explain and recommend best practices of Site Reliability Engineering (SRE), Security and Compliance

Through this process, we aspire for our students to become sophisticated, independent, and resilient problem solvers who are able to overcome challenges and learn.

# Learning Outcomes

In this project-based course, we have project and conceptual learning objectives.

## Project Learning Objectives

The **project learning objectives** (LOs) are the following. Students will be able to:

1. Implement Continuous Integration solutions to automate test and build
2. Implement Continuous Delivery solutions to automate deployment
3. Implement end-to-end CI/CD pipelines for VM-hosted applications
4. Implement end-to-end CI/CD pipelines for containerized microservices
5. Implement end-to-end CI/CD pipelines for serverless architecture
6. Provision and manage cloud infrastructure using Infrastructure as Code
7. Utilize monitoring telemetry, metrics, and alerts for cloud resources to improve the reliability and availability of cloud applications.

## Conceptual Learning Objectives

The **conceptual learning objectives** are below. Students will be able to:

1. Define DevOps and describe the benefits and key metrics of DevOps
2. Describe Systems development life cycle (SDLC), compare Agile v.s. Waterfall and explain how Agile and DevOps work in tandem
3. Describe the definition, benefits, stages and best practices of Continuous Integration/Continuous Delivery
4. Describe the definition, benefits, and best practices of Continuous testing as well as the types of tests
5. Define version control systems and explain how a source control strategy fosters collaboration and enables automation
6. Define infrastructure and configuration management and compare solutions
7. Define Site Reliability Engineering (SRE) and describe the benefits and key metrics of SRE
8. Define and compare logging and monitoring
9. Describe the best practices to design logging, monitoring and actionable alerts
10. Describe the security and compliance requirements in DevOps

# Course organization

Your participation in the course will involve several forms of activity:

1. Reading the conceptual content for each unit.
2. Completing the graded weekly assessments after each unit.
3. Complete projects, which are hands-on training and automated feedback.

# Getting help

Students are encouraged to ask questions about content and projects through **Q&A forum**. The course link for forum is:

[http://Q&A-forum.edu/](http://piazza.com/cmu/spring2020/1531915619/home)

# Policies

###### Working Alone on Projects

Projects that are assigned to single students should be performed individually.

###### Handing in Projects

All assessments are due at 11:59 PM ET (one minute before midnight) on the due dates specified on the Sail() Platform. All hand-ins are electronic.

###### Appealing Grades

After each project module is graded, you have seven calendar days to appeal your grade. All your appeals should be provided by email to the professor.

# Assessment

Inline activities (“Learn by Doing” and “Did I Get This”), which are available in most pages in the OLI course, are simple, non-graded activities to assess your comprehension of the material as you read through the course material. You are advised to complete all of the inline activities before proceeding through to the next page or module. If you missed many of the activities, it is recommended that you review the material again.

The conceptual units consist of modules of content on OLI, each week has a Checkpoint Quiz that you must complete before the deadline posted on OLI. Each weekly Checkpoint Quiz will be worth 5% of your total grade. It is your responsibility to ensure that the quiz is submitted prior to the deadline. You will have only a single attempt to complete each Checkpoint Quiz on OLI.

This course includes several individual projects. Each project module has to be completed based on the deadlines posted on TheProject.Zone. The write-up required to complete each project module is available on TheProject.Zone. Each module has a submission process that is specific to the project module that is due. It is the students’ responsibility to make sure that all project work is completed and that the project module is submitted prior to the deadline. Students typically have multiple attempts to submit the project module on TheProject.Zone.

|  |  |  |
| --- | --- | --- |
| Type | Number | Weight |
| Conceptual Content Quizzes | 8 | 40% |
| Projects | 8 | 60% |
| Total Grade |  | **100%** |

# Cheating

We urge each student to carefully read the [university policy on academic integrity](http://www.xyz.edu/academic-integrity.html), which outlines the policy on cheating, plagiarism or unauthorized assistance. It is the responsibility of each student to produce her/his own original academic work. Collaboration or assistance on academic work to be graded is not permitted unless explicitly authorized by the course instructor. Each unit checkpoint quiz or project module submitted must be the sole work of the student turning it in. Student work on the cloud is logged, submitted work will be closely monitored by automatic cheat checkers, and students may be asked to explain any suspicious similarities with any piece of code available. The following are guidelines on what collaboration is authorized and what is not:

###### What is cheating?

1. Sharing code or other electronic files by either copying, retyping, looking at, or supplying a copy of any file. Copying any code from the internet (stackoverflow.com or github or others). No code can be used to “test” the auto-grader. Anything you submit to the auto-grader must be your code.
2. Copying answers to any checkpoint quiz from another individual, published or unpublished written sources, and electronic sources.
3. Collaborating with another student or another individual on checkpoint quizzes or project modules.
4. Sharing written work, looking at, copying, or supplying work from another individual, published or unpublished written sources, and electronic sources.
5. Collaboration in team projects is strictly limited to the members of the team.

###### What is **not** cheating?

1. Clarifying ambiguities or vague points in class handouts.
2. Helping others use computer systems, networks, compilers, debuggers, profilers, or system facilities.
3. Helping others with high-level design issues.
4. Guiding others through code debugging but not debugging for them.

Cheating in projects will also be strictly monitored and penalized. Be aware of what constitutes cheating (and what does not) while interacting with students. You cannot share or use written code, and other electronic files from students. If you are unsure, ask the teaching staff.

Be sure to store your work in protected directories. The penalty for cheating is severe, and might jeopardize your career – cheating is simply not worth the trouble. By cheating in the course, you are cheating yourself; the worst outcome of cheating is missing an opportunity to learn. In addition, you will be removed from the course with a failing grade. We also place a record of the incident in the student’s permanent record.

# Conceptual topics

The course content will be structured into the following units:

|  |  |  |
| --- | --- | --- |
| Module | Title | Modules and Description |
| 1 | **Introduction to DevOps** | * Define DevOps from organizational, cultural and technical perspectives
* Describe how DevOps helps accelerate development, testing and deployment
* Describe the key metrics of DevOps
* Describe Systems development life cycle
* Compare Agile v.s. Waterfall
* Explain how Agile and DevOps work in tandem
* Describe the job responsibilities of DevOps engineer, what are they responsible for, what skills do they need to have, career path.
* Describe the future outlook of DevOps
 |
| 2 | **An Overview of CI/CD** | * Describe the definition and the benefits of CI/CD
* Describe the stages of CI/CD Pipelines
* Describe Continuous Testing and best practices for Continuous Testing and the types of tests
 |
| 3 | **Source Control** | * Define version control systems (VCS)
* Describe how a source control strategy fosters collaboration and enables automation
* Describe branching strategies (master/feature/release branches, forks, pull requests policies)
* Describe pull request workflow (e.g., CI, code reviews, sign-off)
* Describe trunk-based development and compare trunk-based v.s. mainline
 |
| 4 | **Continuous Integration** | * Describe the motivation for CI and what benefits it provides to the development cycle.
* Describe the concept of Continuous Integration (CI).
* Identify the components of a CI pipeline and how they integrate into the overall workflow.
* Discuss the types and coverage of testing methods in CI.
* Discuss the tradeoff of adopting a managed or hosted agent.
* Present industry best practices of CI and how they improve the efficiency and reliability of the software process.
* Define package management and how to access package dependencies
 |
| 5 | **Continuous Delivery and Continuous Deployment** | * Describe the motivation for CD and what benefits it provides to the development cycle.
* Describe the concept of Continuous Delivery (CD)
* Compare Continuous Delivery and Continuous Deployment
* Discuss the types of testing methods in CD.
* Define and compare different deployment environments
* Define release strategies, e.g., release approvals and gates
* Define and compare Deployment patterns: blue/green, canary, ring release strategy
* Describe strategies to minimize downtime during deployments (Load balancer, rolling deployments, etc.)
 |
| 6 | **Infrastructure and configuration management** | * Define Infrastructure as Code
* Define Configuration management
* Describe the primary responsibility of configuration management: identify, control, and audit configurations
 |
| 7 | **Site Reliability Engineering (SRE)** | * Define Site Reliability Engineering (SRE) and the key metrics of SRE
* Define and compare logging and monitoring
* Describe the best practices to design logging, monitoring
* Describe best practices to design actionable alerts
* Define continuous monitoring
* Define Business Continuity and Disaster Recovery (BCDR)
* Define replication, failover, and recovery
* Define Disaster Recovery Testing (DiRT)
 |
| 8 | **Security and Compliance** | * Describe Identity and access management (IAM) and Role-based Access Control (RBAC)
* Define secrets management
* Describe how to inspect source code for compliance Describe how to Inspect infrastructure for compliance
 |

# Projects

The projects are geared towards providing hands-on experience. Students will learn to develop all projects using various public cloud services. For each project, students are expected to work within a specified budget otherwise they risk being penalized.

## Project 1: Continuous Integration

* Explain Continuous Integration (CI)
* Identify components of a CI pipeline
* Describe the structure of pipelines and compare Jobs, Stages, and Steps in Azure Pipelines
* Experiment building CI pipeline using Azure Pipelines with prepared worked examples
* Design pipeline to build, test and publish the artifacts

## Project 2: Continuous Deployment

* Explain the concept of continuous deployment.
* Explain the concept of the software lifecycle, especially the Agile methodology.
* Explain different types of testing used in the software lifecycle.
* Describe and compare the two approaches to define pipelines in Azure DevOps: using YAML syntax and using the Classic interface
* Implement a CD pipeline using YAML in Azure DevOps.
* Experiment with customized deployment tasks in YAML.
* Configure the CD pipeline to deploy generated artifacts between multiple stages.

## Project 3: CI/CD Pipelines for VM-hosted Webservice

* Design and implement Continuous Integration and Continuous Delivery for a Node.JS application
* Create Deployment Groups and register VMs to deployment groups
* Implement a CD pipeline to deploy the application to appropriate groups of VMs at each stage of release
* Demonstrate a complete iteration of a software development lifecycle
* Monitor the states of virtual machines by using Azure Traffic Manager and configuring alerts.

## Project 4: Introduction to GitHub Actions

* Identify the characteristics and benefits of GitHub Actions
* Identify the components of CI/CD workflow on GitHub Actions, such as Workflows, Jobs, Actions, and Secrets
* Describe the usage of self-hosted runners
* Design and implement a CI/CD workflow with GitHub Actions
* Implement a CI/CD workflow triggered by multiple workflow events
* Implement a CI/CD workflow with conditional steps

## Project 5: CI/CD Pipelines for Containerized Microservices

* Describe Docker and Kubernetes and how a developer uses them to implement a microservices architecture (MSA).
* Build docker images in order to containerize microservices and push the images into Azure Container Registry.
* Provision, configure and manage Kubernetes clusters.
* Deploy multiple microservices in Kubernetes clusters.
* Implement a CI/CD pipeline to automate the building and deploying of containerized microservices using Github Actions.
* Demonstrate a typical systems development life cycle of a microservices architecture.

## Project 6: CI/CD Pipelines for Functions

* Illustrate the definition, advantages and disadvantages of Serverless applications.
* Develop and deploy Serverless applications.
* Implement a CI/CD pipeline for the Serverless application to automate the build, test and deployment.
* Interpret the definition of deployment strategies.
* Explain and adopt the Blue-Green deployment technique.

## Project 7: Infrastructure as Code for DevOps

* Provision cloud resources using Terraform CLI
* Configure multiple interdependent cloud resources
* Configure a resource to perform additional actions upon initialization using provisioners
* Create and manage replicated copies of a working cloud application infrastructure using Terraform workspaces
* Configure Terraform backend for remote state storage in Azure storage account
* Implement a CI/CD pipeline to automate cloud resource provisioning and management

## Project 8: Monitoring Cloud Services

* Explain the concept and the benefits of cloud monitoring
* Practice configuring autoscale settings of Azure Web App plan using both the Azure portal and Terraform
* Configure autoscale setting to collect resource metrics and trigger scale actions
* Explain the workflow of an alert to trigger an action group
* Practice configuring an availability test of a web app on Azure and set up an alert for test fails using both the Azure portal and Terraform

#  Schedule

The tentative schedule is as follows (specific deadlines are posted on OLI and TheProject.Zone):

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Conceptual Content on OLI** | **Quiz** | **Hands-on Projects on TPZ** |
| 1 | M1: Introduction to DevOps | Quiz 1 | P1: Continuous Integration |
| 2 | M2: An Overview of CI/CD | Quiz 2 | P2: Continuous Deployment |
| 3 | M3: Source Control | Quiz 3 | P3: CI/CD Pipelines for VM-hosted Webservice |
| 4 | M4: Continuous Integration | Quiz 4 | P4: Introduction to GitHub Actions |
| 5 | M5: Continuous Delivery and Continuous Deployment | Quiz 5 | P5: CI/CD Pipelines for Containerized Microservices |
| 6 | M6: Infrastructure and configuration management | Quiz 6 | P6: CI/CD Pipelines for Functions |
| 7 | M7: Site Reliability Engineering (SRE) | Quiz 7 | P7: Infrastructure as Code for DevOps |
| 8 | M8: Security and Compliance | Quiz 8 | P8: Monitoring Cloud Services |

# Accommodations for Students with Disabilities

If you have a disability and have an accommodations letter from the Disability Resources office, I encourage you to discuss your accommodations and needs with me as early in the semester as possible. I will work with you to ensure that accommodations are provided as appropriate. If you suspect that you may have a disability and would benefit from accommodations but are not yet registered with the Office of Disability Resources, I encourage you to contact them at access@xyz.edu.

#  Take care of yourself

Do your best to maintain a healthy lifestyle this semester by eating well, exercising, avoiding drugs and alcohol, getting enough sleep and taking some time to relax. This will help you achieve your goals and cope with stress.

All of us benefit from support during times of struggle. You are not alone. There are many helpful resources available on campus and an important part of the college experience is learning how to ask for help. Asking for support sooner rather than later is often helpful.