Course #: Cloud Developer

Course Description & Syllabus

Name of College or University
Semester Year

1. Overview

Title: Cloud Developer
Credits/Units: 3-credit hours or 4-credit hours with a lab.
Pre-requisites: Introduction to Computing.

Institution LMS access: https://lms.univ-or-college.edu

Recitation:

1. **Tuesday, 8:00 AM – 8:50 AM ET, BLDG XYZ (Videotaped)**

Teaching Staff:

Prof. Best Faculty
faculty-email@xyz.edu
Office Number, +1-555-555-5555
Office hours: Tuesday, 3-4pm (time zone)

TAs typically hold office hours in building/room. The TA office hours are posted on the LMS:

- TA Name<ta-email@xyz.edu>
- TA Name<ta-email@xyz.edu>

2.

2. Course Description

Students will gain knowledge and develop hands-on experience solving real-world problems in the area of cloud administration. This includes the process to provision, orchestrate, scale, manage and monitor cloud services across compute, storage, networking, and security using various cloud interfaces. All projects utilize existing public cloud infrastructure, tools, and services. Specifically, students will be exposed to real-world scenarios, infrastructure, and data. It is our goal that students will develop the skills needed to become a cloud administrator.

This project-based on-line course focuses on skill building across various aspects of cloud computing. We cover conceptual topics and provide hands-on experience through projects utilizing public cloud infrastructures Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP).

Students will utilize MapReduce, interactive programming using Jupyter Notebooks, and data science libraries to clean, prepare and analyze a large data set. Students will orchestrate the deployment of auto-scaled, load-balanced and fault-tolerant applications using virtual machines (VMs), Docker containers and Kubernetes, as well as serverless computing through Functions as a Service. Students will explore and experiment with different distributed cloud-storage abstractions (distributed file systems and databases) and compare their features, capabilities, applicability and consistency models. In addition, students will develop different analytics applications using batch, iterative and stream processing frameworks.

Conceptually, the course will introduce this domain and cover the topics of cloud infrastructures, virtualization, software-defined networks and storage, cloud storage, and programming models (analytics frameworks). As an introduction, we will discuss the motivating factors, benefits and challenges of the cloud, as well as service models, service level agreements (SLAs), security, example Cloud Service Providers and use cases. Modern data centers enable many of the economic and technological benefits of the cloud paradigm; hence, we will describe several concepts behind data center design and management and software deployment. Next, we will focus on virtualization as a key cloud technique for offering software, computation and storage services. Within the same theme of virtualization, students will also be introduced to Software Defined Networks and Storage (SDN and SDS). Subsequently, students will learn about different cloud storage concepts including data distribution, durability, consistency and redundancy. We will discuss distributed file systems, NoSQL databases and object storage. Finally, students will learn the details of the MapReduce programming model and gain a broad overview of the Spark, GraphLab programming models as well as message queues (Kafka) and stream processing (Samza).

3. Course Goals

In this online course, students gain hands-on experience solving real world problems by completing projects in the areas of cloud analytics, compute and elasticity, storage and frameworks, which utilize existing public cloud tools and services. It is our goal that students will develop the skills needed to become a practitioner or carry out research projects in the domain of cloud computing. Specifically, students are exposed to real-world data, scenarios, infrastructure and budgets in order to learn how to:

1) Design, architect, implement, test, deploy, monitor and maintain cloud-based applications;
2) Identify the appropriate tools and architectures to implement a cloud-based design;
3) Analyze the tradeoffs between different tools and cloud offerings to meet real-world constraints;
4) Evaluate performance characteristics of cloud-based services to implement optimizations;

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

4. Learning Outcomes

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

4.1. Project Learning Objectives

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

1. Analyze and visualize small and large data sets on the cloud using interactive computing as well as data manipulation and analysis libraries.
2. Design, implement, test, package, deploy and monitor cloud applications using Virtual Machines (VMs), Containers and Serverless cloud computing services.
3. Explore and experiment with different distributed cloud-storage abstractions and compare their features, capabilities and applicability.
4. Orchestrate, deploy and optimize a unified application that integrates heterogeneous SQL and NoSQL database systems.
5. Implement and compare consistency models to recognize the tradeoff between consistency and performance in replicated and distributed cloud storage systems.
6. Design, implement, test and debug applications using interactive, batch and stream processing frameworks and compare their suitability to different problem domains.
7. Illustrate and explain the execution workflow, overhead, fault-tolerance and logical flow of interactive, batch and stream processing frameworks.
8. Train and deploy a machine learning model using a cloud-based framework.
9. Analyze and identify potential sources of bottlenecks in programming frameworks to optimize their performance.
10. Practice gathering, cleaning and preparing data for analysis on the cloud.
11. Practice Test-driven Development (TDD) in the software development process.
12. Orchestrate and automate the process of managing and provisioning cloud resources through machine-readable definition files.
13. Make informed decisions about choosing an appropriate cloud tool that will satisfy a set of specified requirements.

4.2. Conceptual Learning Objectives

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

1. Explain the ***core concepts*** of the cloud computing paradigm: how and why this paradigm shift came about, the characteristics, advantages and challenges brought about by the various models and services in cloud computing.
2. Apply fundamental concepts in ***cloud infrastructures*** to understand the tradeoffs in power, efficiency and cost, and then study how to leverage and manage single and multiple datacenters to build and deploy cloud applications that are resilient, elastic and cost-efficient.
3. Discuss ***system, network and storage virtualization*** and outline their role in enabling the cloud computing system model.

4. Describe the overall organization of data and the fundamental concepts of ***cloud storage***. Identify the problems of scale and management in big data. Discuss various storage abstractions. Compare and contrast different types of distributed file systems and discuss their design considerations. Compare and contrast different types of databases and discuss their design tradeoffs. Discuss the concepts of cloud object storage. Enumerate the different types of block devices used in data storage.

5. Explain the main execution flow, scheduling and fault tolerance concepts in various ***cloud programming models***. Recall and contrast different cloud programming models (MapReduce, Spark, GraphLab, Spark Streaming and Samza).

5. Course organization

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

1. Reading the conceptual content for each unit on OLI.
2. Completing the unscored inline activities for each unit (Review activities on OLI).
3. Completing the graded checkpoint quizzes after each unit.
4. Complete projects, which are performed on the cloud and submitted through TheProject.Zone.
5. AssessMe assessments, unscored short quizzes to unlock subsequent project sections.

Students should regularly check OLI to see when new content or checkpoint quizzes are made available. Projects must be completed by the due dates posted on TheProject.Zone.

6. Getting help

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

http://Q&A-forum.edu/

7. 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 Prof. Sakr.

8. 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.

There are five units consisting 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 2% 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 four individual project themes. Each project theme consists of several project modules. Every week, a 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
Content Quizzes	10 out of 11	20%
Individual Projects	10 out of 11	80%
Total Grade		**100%**

9. Cheating

We urge each student to carefully read the university policy on academic integrity, 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.

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.

10. Conceptual Topics

The course content will be structured into the following units:

Unit #	Title	Modules and Description
1	**Introduction**	Definition and evolution of Cloud Computing Enabling Technologies, Service and Deployment Models Popular Cloud Stacks and Use Cases Benefits, Risks, and Challenges of Cloud Computing Economic Models and SLAs Topics in Cloud Security
2	**Cloud Infrastructure**	Historical Perspective of Data Centers Datacenter Components: IT Equipment and Facilities Design Considerations: Requirements, Power, Efficiency, & Redundancy Power Calculations, PUE and Challenges in Cloud Data Centers Cloud Management and Cloud Software Deployment Considerations

3	**Virtualization**	Virtualization (CPU, Memory, I/O), Case Study: Amazon EC2 Software Defined Networks (SDN) Software Defined Storage (SDS)
4	**Cloud Storage**	Introduction to Storage Systems Cloud Storage Concepts Distributed File Systems (HDFS, Ceph FS) Cloud Databases (HBase, MongoDB, Cassandra, DynamoDB) Cloud Object Storage (Amazon S3, OpenStack Swift, Ceph)
6	**Programming Models**	Distributed Programming for the Cloud Data-Parallel Analytics with Hadoop MapReduce (YARN); Iterative Data-Parallel Iterative Analytics (Spark); Graph-Parallel Analytics with GraphLab 2.0 (PowerGraph); Stream Processing (Samza)

11. Projects

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

11.1. Project 1: Big Data Analysis

Students will set up cloud accounts, work with provisioning management software and launch instances on Amazon EC2. Students will learn the benefits and tradeoffs of running programs in parallel, using AWS EMR or Azure HDInsight or GCP Dataproc, versus sequential on a large dataset within particular cost constraints. Students will utilize Jupyter Notebooks to solve a set of data analytics problems.

11.2. Project 2: Cloud Elasticity

In this project, students will learn about cloud elasticity through virtual machines, containers and functions using the AWS, Azure and GCP APIs. Students will be tasked with developing their own elastic services for a dynamically changing load scenario. Students will work with the Load Balancing and Auto Scaling services on AWS to mitigate varying loads on the server. Furthermore, students will deploy a web service using Docker Containers and Kubernetes on multiple cloud platforms. Lastly, students will utilize functions (FaaS) to develop a video indexing service.

11.3. Project 3: Cloud Storage

Using cloud resources and services, students will explore cloud storage technologies to evaluate their capabilities, limitations, and applicability to application domains. Students begin by exploring the limitations of traditional filesystems, and then compare them to relational databases (MySQL) and NoSQL databases (HBase). Next, students

build a social network timeline using heterogeneous back-end storage systems. The project will cover several storage systems, including low-latency KV stores, and NoSQL databases, (e.g., Neo4j, DBaaS, MongoDB and others). Finally, students implement strong and eventual consistency for a geo-replicated key-value stores.

11.4. Project 4: Cloud Frameworks

Students will work on developing applications using the Spark, and Samza and Machine Learning frameworks to experience iterative batch, graph and stream processing. Students will implement a social graph analysis algorithm on Apache Spark. Students will perform feature engineering to build and deploy a machine learning model on the cloud. Students will learn to analyze streaming data to perform real-time processing of multiple data streams using Apache Kafka/Samza.

12. Schedule

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

15-week option:

Week	Conceptual Content on OLI	Quiz	Hands-on Projects on TPZ
1	Unit 1, Module 1, 2	Quiz 0	P0
2	Unit 1, Module 1, 2	Quiz 1	P1.1
3	Unit 2, Module 3, 4	Quiz 2	P1.2
4	Unit 2, Module 5, 6	Quiz 3	P2.1
5	Unit 3, Module 7, 8, 9	Quiz 4	P2.2
6	Unit 3, Module 10, 11, 12	Quiz 5	P2.3
7	Unit 3, Module 13	Quiz 6	P3.2
8	Unit 4, Module 14	Quiz 7	
9	Unit 4, Module 15, 16, 17	Quiz 8	P3.3
10	Unit 4, Module 18	Quiz 9	
11			P4.1
12	Unit 5, Module 19, 20	Quiz 10	
13			P4.2
14	Unit 5, Module 21, 22	Quiz 11	
15			P4.3

13. 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@andrew.cmu.edu.

14. 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.