15-319/15619: Cloud Computing

Course Description & Syllabus

Carnegie Mellon University
Spring 2024

1. Overview

Title: Cloud Computing
Units: 15-319 is 12 units, and 15-619 is 15 units.
Pre-requisites for undergraduate students: A "C" or better in 15-213.
Pre-requisites for graduate students: Knowledge of computer systems, programming and debugging, with a strong competency in at least one language (such as Java/Python), and the ability to pick up other languages as needed.

Canvas Course: https://canvas.cmu.edu/
OLI Course: Accessed through Canvas
The Sail() Platform: Accessed through Canvas
Piazza: Accessed through Canvas
Webpage: http://www.cs.cmu.edu/~msakr/15619-s24/
Weekly Overview: Tuesdays (Videotaped) - Accessed through Canvas

Teaching Staff:

Prof. Majd Sakr
msakr@cs.cmu.edu
GHC 7006, +1-412-268-1161
Office hours: Tuesday, 3-4 pm ET (Pittsburgh)

Prof. Seth Goldstein
seth@cmu.edu
GHC 7111, +1-412-268-3828
Office hours: TBD, TBD pm ET (Pittsburgh)

TA OHs are posted on Piazza.

The course teaching staff:

- Halim Boukaram <hboukara@andrew.cmu.edu>
- Johnny Chang <chiamin2@andrew.cmu.edu>
- Mateo Correa <jmcorrea@andrew.cmu.edu>
- Tom Dai <hanqid@andrew.cmu.edu>
- Yu Gu <ygu3@andrew.cmu.edu>
- Yashi Gupta <yashig@andrew.cmu.edu>
- Junxi He <junxih@andrew.cmu.edu>
- Robert Hu <zhaohanh@andrew.cmu.edu>
- Joyce Hong <joyceh@andrew.cmu.edu>
- Hanzhe Lu <hanzhel@andrew.cmu.edu>
- Naren Omprakash <nompraka@andrew.cmu.edu>
- Baljit Singh <baljits@andrew.cmu.edu>
- Divyaansh Sinha <divyaans@andrew.cmu.edu>
- Ashwin Nayak Ullal <anayakul@andrew.cmu.edu>
- Leo Wang <hongshuw@andrew.cmu.edu>
- Ruby Wu <yunjiew@andrew.cmu.edu>
- Jessica Yin <ruhany@andrew.cmu.edu>
- William Zheng <yuxiaoz2@andrew.cmu.edu>

2. Course Description

This project-based online 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), as well as Docker containers and Kubernetes. Students will explore and experiment with different distributed cloud storage abstractions (distributed file systems and databases) and compare their features, capabilities, and applicability. In addition, students will develop different analytics applications using batch, iterative, and stream processing frameworks. The 15-619 [graduate] students will participate in a team project, which entails designing and implementing a complete web-service solution for querying big data. For the team project, the student teams are evaluated based on the cost and performance of their web service.

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 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, and 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 **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;
5) [15-619 only] Collaborate with a team on an open-ended project to incrementally realize an optimized end-to-end cloud-based solution.

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 common to the undergraduate and graduate offerings of the course. The graduate-only (15-619) LOs are designated below. Students will be able to:

Compute & Elasticity	1) Design, implement, test, package, deploy and monitor cloud applications using Virtual Machines (VMs), and Containers cloud computing services.
Cloud Storage	2) Explore and experiment with different distributed cloud-storage abstractions and compare their features, capabilities and applicability. 3) Orchestrate, deploy and optimize a unified application that integrates heterogeneous SQL and NoSQL database systems.
Frameworks	4) Design, implement, test and debug applications using interactive, batch and stream processing frameworks and compare their suitability to different problem domains. 5) Illustrate and explain the execution workflow, overhead, fault-tolerance and logical flow of interactive, batch and stream processing frameworks. 6) Train and deploy a machine learning model using a cloud-based framework. 7) Analyze and identify potential sources of bottlenecks in programming frameworks to optimize their performance.
Team Project [15-619 only]	8) Design, build, and deploy a performant, reliable, scalable and fault-tolerant cloud native microservice based web service on the cloud within a specified budget. 9) Perform extract, transform and load (ETL) on a large data set. 10) Design schema as well as configure and optimize cloud-based databases to deal with scale and improve the throughput of a web service. 11) Explore methods to identify the potential bottlenecks in a cloud native web service and implement methods to improve system performance.
Overall	12) Practice gathering, cleaning and preparing data for analysis on the cloud. 13) Practice Test-driven Development (TDD) in the software development process. 14) Orchestrate and automate the process of managing and provisioning cloud resources through machine-readable definition files. 15) 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 online coursework content for each unit on OLI.
2. Completing the unscored inline activities for each unit (Review activities on OLI).
3. Completing the graded checkpoint weekly quizzes after each unit.
4. Complete projects, which are performed on the cloud and submitted through the Sail() Platform.
5. AssessMe assessments, unscored short quizzes to unlock subsequent project sections.
6. Complete a team project on building a complete web service.

Students should regularly check OLI to see when new content or checkpoint quizzes are made available. Projects and Checkpoint quizzes must be completed by the due dates posted on the Sail() Platform.

6. Getting Help

Students are encouraged to ask questions about content and projects through **Piazza**, where an online class portal has been created for this course. Students can access Piazza through Canvas.

There is a video recording of a weekly recitation which is made available to all students. The teaching staff will discuss any major questions posted to Piazza. It is best to post on Piazza for urgent communication with the teaching staff.

We will use the course website as the basic portal for the class. The course's conceptual content is on OLI. The project write-ups, submission, scoreboard, and grades are on the Sail() Platform. The checkpoint quizzes are on OLI. OLI can be reached through Canvas. Announcements, discussions, and questions are posted on Piazza.

7. Policies

Working Alone on Projects

Projects assigned to individual students should be performed individually.

Handing in Projects

All assessments are due at 11:59 PM EST (one minute before midnight) on the due dates specified on OLI or the Sail() Platform. All hand-ins are electronic and use the OLI Checkpoint system and the Sail() Platform.

Appealing Grades

After each project 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 on 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 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, and 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. You are responsible for ensuring that the quiz is submitted before the deadline. You will have only a single attempt to complete each Checkpoint Quiz on OLI.

This course includes six individual projects. Each project consists of several tasks. A project has to be completed based on the deadlines posted on the Sail() Platform. The write-up required to complete each project is available on the Sail() Platform. Each project has a submission process that is specific to the project that is due. It is the students' responsibility to make sure that all project work is completed and that the project is submitted prior to the deadline. Students have multiple attempts to submit the project on the Sail() Platform.

15-619 students have to complete a team-based multi-week project in parallel to the individual projects.

Type	Number	Weight
Content Checkpoint Quizzes	10 out of 11	20%
Projects 15-319 Projects 15-619	5 out of 6 for 15-319 5 out of 6 + Team for 15-619	80% 60% + 20%
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 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 projects.
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. You are cheating yourself by cheating in the course; 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 conceptual 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. Project Areas

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: Cloud Elasticity

In this project, students will learn about cloud elasticity through virtual machines using the AWS 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.

11.2. Project 2: Microservices with Containers and Kubernetes

In this project, students will learn about microservices and elasticity through Docker containers and Kubernetes using the Azure and GCP APIs. Students will deploy a web service using Docker Containers and Kubernetes on multiple cloud platforms. Students will also implement a CI/CD pipeline using GitHub Actions which automate the building and deploying of containerized microservices.

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 will explore and experiment with relational databases (MySQL) and NoSQL databases (HBase, MongoDB) to select and deploy suitable and effective databases for certain use cases or applications.

11.4. Project 4: Cloud Frameworks - Batch Processing with Spark

Students analyze a Twitter social graph using Apache Spark and perform a number of different graph computations in order to understand the nature of the Twitter social graph. Students perform data exploratory analysis on a Twitter social graph dataset, run the PageRank algorithm on the graph to find the most influential users on Twitter, deploy their Spark programs on Azure HDInsight and optimize their programs to meet certain performance objectives. In the bonus task, students run their Spark program on Azure Databricks and compare the UX and performance between HDInsight and Databricks.

11.5. Project 5: Cloud Frameworks - Stream Processing

Students will work on developing applications using the Apache Samza framework to experience stream processing. Students will develop solutions to analyze streaming data to perform real-time processing of multiple data streams using Apache Kafka/Samza.

11.6. Project 6: Cloud Frameworks - ML on the Cloud

Students will work on developing applications using Machine Learning frameworks. Students will perform feature engineering to build and deploy a machine learning model on the cloud.

11.7. 15-619 Team Project: Cloud Native Web Service with Microservices

Students will work in teams to design and implement a complete cloud native web service that uses the microservice model and the REST interface to respond to queries that require running an analytics job on a large (1.2TB) Twitter data set which is stored in a database (MySQL, HBase, etc.). Student teams are expected to use different tools and services to build a performing web service that meets the requirements. The web services should be automatically deployed to the cloud using CI/CD practices. The web services are evaluated through a load generator for a fixed time period (several hours) by measuring the cost of cloud resources used and their system's performance (throughput). There is an upper bound on the budget, which could cause students to be disqualified. Students are evaluated based on how their service performs compared to a baseline.

12. Schedule

The tentative schedule is as follows (specific deadlines are posted on OLI and the Sail() Platform):

Week	Monday	OLI Content	Individual Projects	Team Project	Quizzes
1	1/15/2024	Unit 1, Module 1, 2	Primers/P0 (Jan 21)		Q0 (Ac. Integ.)
2	1/22/2024	Unit 1, Module 1, 2	P1 Elasticity (Feb 4)		Q1 (Jan 26)
3	1/29/2024	Unit 2, Module 3, 4			Q2 (Feb 02)
4	2/5/2024	Unit 2, Module 5, 6	P2 Cont. & K8s (Feb 18)		Q3 (Feb 09)
5	2/12/2024	Unit 3, Module 7		Project Out (Feb 12)	Q4 (Feb 16)
6	2/19/2024	Unit 3, Module 8, 9, 10	P3 Storage Part 1 (Feb 25)		Q5 (Feb 23)
7	2/26/2024	Unit 3, Module 11	P3 Storage Part 2 (Mar 03)		Q6 (Mar 01)
8	3/4/2024	Spring Break			
9	3/11/2024	Unit 4, Module 12, 13	P4 Spark (Mar 24)	Phase 1 Due (Mar 17)	Q7 (Mar 15)
10	3/18/2024	Unit 4, Module 14, 15		Phase 2 Out (Mar 18)	Q8 (Mar 22)
11	3/25/2024	Unit 4, Module 16, 17, 18	(No individual project)		Q9 (Mar 29)
12	4/1/2024	Unit 5, Module 19, 20, 21	P5 Streaming (Apr 14)	Phase 2 Due (Apr 07)	Q10 (Apr 05)
13	4/8/2024	Unit 5, Module 22		Phase 3 Out (Apr 08)	Q11 (Apr 12)
14	4/15/2024		P6 AI/ML (Apr 28)	Phase 3 Due (Apr 21)	
15	4/22/2024				

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.

If you or anyone you know experiences any academic stress, difficult life events, or feelings like anxiety or depression, we strongly encourage you to seek support. Counseling and Psychological Services (CaPS) is here to help: call 412-268-2922 and visit their website at http://www.cmu.edu/counseling/. Consider reaching out to a friend, faculty or family member you trust for help getting connected to the support that can help.

If you or someone you know is feeling suicidal or in danger of self-harm, call someone immediately, day or night:

CaPS: 412-268-2922

Resolve Crisis Network: 888-796-8226

If the situation is life threatening, call the police:

- **On campus**: CMU Police: 412-268-2323
- **Off campus**: 911

If you have questions about this or your coursework, please let me know.