

15-319/15-619: CLOUD COMPUTING

COURSE DESCRIPTION & SYLLABUS

CARNEGIE MELLON UNIVERSITY
SPRING 2026

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.

Course on Canvas: <https://canvas.cmu.edu/>

The Sail() Platform: Accessed through [Canvas](#)

Piazza: Accessed through [Canvas](#)

Webpage: <http://www.cs.cmu.edu/~msakr/15619-s26/>

Weekly Overview: Tuesdays (Videotaped) - Accessed through [Canvas](#)

Teaching Staff:

Prof. Majd Sakr

msakr@cs.cmu.edu

GHC 7006, +1-412-268-1161

Office hours: (See Piazza)

Prof. Seth Goldstein

seth@cmu.edu

GHC 7111, +1-412-268-3828

Office hours: (See Piazza)

TA OHs are posted on Piazza.

The course teaching staff:

- Samuel Adeniji <sadeniji@andrew.cmu.edu>
- Halim Boukaram <hboukara@andrew.cmu.edu>
- Shimao (Simon) Du <shimaod@andrew.cmu.edu>
- Oseno Ewaose-Joseph <oe@andrew.cmu.edu>
- Jingxiang (Eric) Gao <jingxia3@andrew.cmu.edu>
- Mark Iraguha <miraguha@andrew.cmu.edu>
- Eric Kumara <ekumara@andrew.cmu.edu>
- Kevin Kuo <kkuo2@andrew.cmu.edu>
- Andy Tang <andyt@andrew.cmu.edu>
- Baljit Singh <baljits@andrew.cmu.edu>
- Yiqing (William) Sun <yiqings2@andrew.cmu.edu>
- Bonheur Tuyikunde <btuyikun@andrew.cmu.edu>
- Yuan (Yara) Yao <yuan Yao3@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 on Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).

Students will utilize cloud programming frameworks, interactive programming with Jupyter Notebooks, and data science libraries to clean, prepare, and analyze a large dataset. 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 semester-long project, which entails designing and implementing a complete web-service solution for querying big data. For the Semester Project, students are evaluated based on both the performance and the cost of their web services.

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 practitioners 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] Develop viable solutions that consider correctness, performance and cost, in 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.
Semester 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 dataset. 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.
2. Completing the unscored inline activities for each unit.
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. In-line activities, unscored short quizzes to unlock subsequent project sections.
6. Complete a semester-long project on building a complete cloud-hosted web service.

Projects and 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 course overview, 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 tools (Sail(), Piazza, and Canvas) as the portals for the class. The course's conceptual content, quizzes, project write-ups, submission, scoreboard, and grades are all on the Sail() Platform, which can be reached through Canvas. Announcements, discussions, and questions are posted on Piazza.

7. POLICIES

WORKING ALONE ON PROJECTS

All projects are assigned to individual students and 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 and use the Sail() Platform.

APPEALING GRADES

After each project is graded, you have seven calendar days to appeal your grade. All your appeals should be posted privately on Piazza and only to the teaching staff.

8. ASSESSMENTS & GRADING POLICY

Inline activities 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.

The course includes a graded weekly quiz that you must complete before the deadline posted on the Sail() Platform. Each weekly 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 quiz.

This course includes five cloud-service projects, one of which is two half-projects (storage services) with two deadlines. 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 semester-long multi-week project in parallel to the cloud services projects.

Type	Number	Weight
Quizzes	10 out of 11	20%
Projects for 15-319	4 out of 5 Cloud Services Projects for 15-319	80%
Projects for 15-619	4 out of 5 Cloud Services Projects + Semester Project for 15-619	50% + 30%
Total Grade		100%

Grace Days

- All students start the semester with two (2) grace days.
- The project titled Getting Started with Cloud Computing is ungraded, but completing it with a 100% project score before the original deadline grants one (1) additional grace day.
- SQL and NoSQL (Storage Services Project - Part 1) and Heterogeneous Storage on the Cloud (Storage Services Project - Part 2) count as one project for final grading, but have different deadlines and require separate grace days.
- Grace days cannot be applied to Machine Learning on the Cloud (AI/ML Services Project).
- Grace days cannot be applied to any Semester Project or Quiz submission.
- At most one (1) grace day can be applied to each project.
- Students should apply a grace day themselves on the Sail() Platform for the project for which they desire a grace day.
- Grace days can only be used within 24 hours of the original deadline and only extend to 24 hours of the original deadline

Bonuses

- Cloud Services projects and quizzes do not have bonus tasks.
- The semester-long project bonuses are detailed in the respective project handouts.
- All students are eligible for a 2% bonus for completing the end-of-semester survey.

Penalties and Penalty Waivers

- All students start the semester with one (1) penalty waiver. Additional penalty waivers may be earned in the Semester Project, as detailed in the Semester Project handouts.
- Penalty waivers apply to penalties specified in project handouts (e.g., missing tags, including credentials, using the wrong instance types, and others).
- Penalties incurred for over-spending (on either a Cloud Services project or the Semester project) cannot be waived, even if additional penalty waivers are earned.
- Multiple penalties incurred in a project/phase are summed up before they are applied.
- Penalty waivers are applied to the highest sums of penalties in a cloud services project (minus over-spending penalties), before the lowest project score is dropped.
- Penalty waivers are applied by the teaching staff, not the students themselves.
- Manual grading deductions are not penalties and cannot be waived.

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 (GenAI tools, stackoverflow.com, 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 (such as GenAI tools).
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 any of the course projects is strictly prohibited.

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 discuss high-level design issues.
4. Guiding others through suggesting potential avenues for debugging code, 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 ability to earn a degree from Carnegie Mellon as well as 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

Here is a list of the topic areas covered by the course's conceptual content:

Topic Area	Modules and Description
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
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
Virtualization	Virtualization (CPU, Memory, I/O), Case Study: Amazon EC2 Software Defined Networks (SDN) Software Defined Storage (SDS)
Cloud Storage	Introduction to Storage Systems Cloud Storage Concepts Distributed File Systems (HDFS, Ceph FS) Cloud Databases (MongoDB, Cassandra, DynamoDB) Cloud Object Storage (Amazon S3, OpenStack Swift, Ceph)
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. CLOUD ELASTICITY SERVICES

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. 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, Kubernetes and Helm 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. 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 exploratory data 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.

11.4. CLOUD STORAGE SERVICES

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 (Redis, MongoDB) to select and deploy suitable and effective databases for certain use cases or applications.

11.5. CLOUD FRAMEWORKS - STREAM PROCESSING SERVICES

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. CLOUD FRAMEWORKS - AI/ML SERVICES

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. Students will utilize LangGraph to orchestrate agentic workflows, and implement a Retrieval-Augmented Generation (RAG) system with prompt engineering to enhance language model performance.

11.7. 15-619 SEMESTER PROJECT: CLOUD NATIVE WEB SERVICE WITH MICROSERVICES

In this semester-long project, students will work individually 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, etc.). Students 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. TENTATIVE SCHEDULE

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

Week	Monday	Cloud Services Projects	Semester Project	Quizzes	Notes
1	1/12/2026	Primers/Started with Cloud		Q0*	
2	1/19/2026	Elasticity Services	Phase 0	Q1 (Jan 23)	MLK Day (1/19)
3	1/26/2026	Elasticity Services	Phase 0	Q2 (Jan 30)	Add Deadline (1/26)
4	2/2/2026	Containers & K8s Services	Phase 0	Q3 (Feb 6)	
5	2/9/2026	Containers & K8s Services	Phase 1	Q4 (Feb 13)	
6	2/16/2026	Storage Services - Part 1	Phase 1	Q5 (Feb 20)	
7	2/23/2026	(No cloud services project)	Phase 1	Q6 (Feb 27)	Drop Deadline (2/23)
8	3/2/2026	Spring Break			
9	3/9/2026	Storage Services - Part 2	Phase 1	Q7 (Mar 13)	Mid-Semester Grades (3/11)
10	3/16/2026	Streaming Services	Phase 2	Q8 (Mar 20)	
11	3/23/2026	Streaming Services	Phase 2	Q9 (Mar 27)	
12	3/30/2026	AI/ML Services**	Phase 2	Q10 (Apr 3)	Pass/Fail Deadline (3/30)
13	4/6/2026	AI/ML Services**	Phase 3		Carnival (4/9-11)
14	4/13/2026		Phase 3	Q11 (Apr 17)	
15	4/20/2026		Phase 3		Last day of classes (4/24)

*Q0 is the academic integrity quiz and is mandatory for all students, due on the Add Deadline (1/26).

**No grace days can be used for P6 AI/ML.

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.