SYLLABUS – EE 547: Spring 2026 (2 units)

Applied and Cloud Computing for Electrical Engineers

Instructor:	Brandon Franzke	Office:	EEB 504B	
Email:	franzke@usc.edu	Hours:	Monday:	13:00 - 14:45
			Thursday:	12:00 - 13:30

This course introduces tools and concepts to build and deploy machine learning systems in modern computing environments. It is a project-driven course that develops from concept to production deployment. The course is intended for graduate electrical engineering students with prior programming and machine learning experience. Students will learn about technologies and practices essential for scaling machine learning from experimental notebooks to production systems. The course covers three main areas: (1) cloud technologies and distributed computing for ML workloads, (2) system architecture and infrastructure programming, and (3) deployment and orchestration in global computing infrastructure. Students gain hands-on experience with GPU clusters, containerization, and cloud environments while learning concepts that apply across modern ML platforms.

Lecture	Wednesday (section: 31250)	14:00 – 15:50
Discussion	Friday (section: 30404)	14:00 – 14:50

Enrollment is in-person ONLY. Attendance is mandatory to all lectures. Taping or recording lectures or discussions is strictly forbidden without the instructor's explicit written permission.

Teaching assistants

TA:	Zijing Chen
Email:	zijingch@usc.edu
Office:	*see Course Website*

Course resources

Course website https://ee547.usc-ece.com.

Gradescope https://gradescope.com. Electronically submit all homeworks. You will be automatically added to Gradescope during the first week of classes. Contact Dr. Franzke with technical issues.

Course materials

[1] *Designing Machine Learning Systems*, Huyen, C., O'Reilly Media, 2022. online, USC libraries.

[2] *Cloud Native Patterns: Designing change-tolerant software*, Davis, C., Manning, 2019. online, USC libraries.

[3] *The Good Parts of AWS*, Vassallo, D., Pschorr, J., 2020. (optional).

[4] *High Performance Python: Practical Performant Programming for Humans*, 3rd edition, Gorelick, M., Ozsvald, I., O'Reilly Media, 2020. online, USC libraries.

[5] *Streaming Systems: The What, Where, When, and How of Large-Scale Data Processing*, Akidau, T., Chernyshev, S., Lax, R., O'Reilly Media, 2018. online, USC libraries.

"AI" policy. You may use AI-powered tools in this course to enhance your learning and productivity. Use AI as a collaborative tool for understanding concepts, generating ideas, and troubleshooting. Approach AI-generated content critically and use it responsibly. Engage with AI as you would with a knowledgeable peer or tutor, using iterative conversations to deepen your understanding. You must attribute all AI-generated content in your work, including the prompts you used. You are fully accountable for the accuracy and appropriateness of any AI-assisted work. AI should supplement, not substitute, your own critical thinking and problem-solving. For assignments, you may use AI to clarify concepts or resolve issues, but submitted work must be your own. Submitting AI-generated work as your own without proper attribution or understanding is academic misconduct and will be treated as such.

You must develop complete mastery of all course material independent of AI assistance. Your knowledge and skills will be evaluated in contexts where AI tools are not accessible, mirroring real-world scenarios where you must rely solely on your own expertise. This ensures you can perform effectively in any situation, with or without AI support. Violations of this policy will result in severe academic penalties. The goal is to prepare you to use AI effectively in your future work while ensuring you develop a strong, self-reliant foundation in the course material.

Learning objectives

Upon completion of this course, a student will be able to:

- Design and implement distributed systems for machine learning workloads, understanding asynchronous coordination and scaling limits.
- Apply containerization and orchestration technologies to manage ML training and inference.
- Optimize cloud resource utilization through cloud instances and address data locality.
- Implement fault-tolerant ML systems using service discovery and recovery mechanisms.
- Build production ML services with proper monitoring, versioning, and rollback capabilities.
- Navigate the transition from experimental notebooks to production-ready ML systems.

Course Outline

	Topics	Notes
Week 1 14 Jan	Distributed systems for ML. Coordination, Components, and Cloud.	
Week 2 21 Jan	Virtualization. Containers and orchestration. Fault tolerance	
Week 3 28 Jan	Cloud Computing Fundamentals.	
Week 4 04 Feb	Practical Cloud Computing.	
Week 5 11 Feb	Structured Storage and Query Systems.	
Week 6 18 Feb	SQL and Relational Databases.	
Week 7 25 Feb	NoSQL and Distributed Databases.	
Week 8 04 Mar	API Design. REST. Access control and data privacy.	
Week 9 11 Mar	**Exam.**	
(18 Mar)	**No class, Spring Break.**	
Week 10 25 Mar	Cloud Patterns: Storage, Serverless, and Coordination.	**Draft proposal due (27 Mar)**.
Week 11 01 Apr	**Project proposal meetings**.	
Week 12 08 Apr	Web Interfaces: HTML, CSS, and JavaScript	**Revised proposal due (05 Apr)**.
Week 13 15 Apr	Dynamic Web Applications	**Status report due (19 Apr)**.
Week 14 22 Apr	**Project meetings and wrap-up**.	
Week 15 29 Apr	**Technical review and demos, 14:00 $\sim$ 17:00.**	
Sunday **03 May**	**Project deliverables, due 23:59.**	

Grading Procedure

Homework (45%). Assignments include a mix of applied and programmatic problems. You may discuss homework problems with classmates but each student must submit their own original work. Cheating warrants an "F" on the assignment. Turning in substantively identical homework solutions counts as cheating.

Late homework is accepted with a 0.5% deduction per hour, up to 48-hours – **no exceptions**. Technical issues while submitting are not grounds for extension. No submissions will be accepted 48-hours after the due date. Graders score what is submitted and will not follow up if the file is incorrect, incomplete, or corrupt. It is your responsibility to ensure you submit the correct files and that they are accessible.

Exam (20%). The exam tests your ability to apply major principles, demonstrate conceptual understanding, and requires writing code. It occurs during week 9 (tentative). You are expected to bring a scientific (non-graphing) calculator. You may use both-sides of one 3.5" $\times$ 5" reference card. You may not use any additional resources.

The exam includes multiple-choice and short answer questions. It also includes free-response or open-ended questions to demonstrate conceptual understanding. You are expected to write reasonably correct code as well as determine expected behavior of novel computer code. Grading primarily follows correct reasoning but may include deductions for major syntax errors, algorithmic inefficiency, or poor implementation.

Final project (35%). This course culminates with a final project in lieu of a final exam. Teams of three students (in rare cases, teams of two with instructor approval) design and implement a complete software product that connects two or more independent asynchronous components (often *frontend* and *backend*). The instructor will guide teams having difficulty identifying a suitable application. Teams are encouraged to devise solutions to novel problems of personal interest to their background or research. But teams may build an application similar to existing services or tools provided their efforts demonstrate understanding of the development stack and the product lifecycle — from idea to deployment to maintenance. All projects must obtain the instructor's written approval. Teams will prepare and present/demo their approved project and show how it applies course material, concepts, and best-practices.

Course Grade

A if 90 - 100 points, **B** if 80 - 89 points, **C** if 70 - 79 points, **D** if 60 - 69 points, **F** if 0 - 59 points.
("+" and "–" at $\approx$ 1.5% of grade boundary).

Cheating

Cheating is not tolerated on homework or exams. Penalty ranges from F on exam to F in course to recommended expulsion.

Final Project

Project Requirements

Project topics must include sufficient scope and apply course knowledge to a useful end. The project must implement a complete ML system that includes training infrastructure and deployment serving, demonstrating the transition from experimental development to production operation. It must compose at least two distinct computational components that demonstrate distributed system principles (examples: training and inference pipelines, data processing and model serving, or experimentation platform and production deployment). The project must demonstrate comprehensive understanding of the entire development stack and the product lifecycle from idea to deployment to maintenance. Additional requirements and guidelines will be discussed closer to the commencement of the project.

All projects must use Python as the primary language unless approved explicitly in writing by the instructor. But projects may use additional languages for tooling and support. Projects must implement and expose some API or service to consumers. The instructor may provide additional requirements when introducing the final project assignment.

Grading and Milestones

Topic proposal (initial and revised)	week 10 & 12	3% + 7%
Status report - Design, components, integration	week 14	7%
Technical review and demo	final	25%
Project report		20%
Design and source code		35%
Video		3%

Deliverables

Topic proposal: describe the problem, proposed technical approach, and expected outcomes. It should communicate that your topic is adequately prepared and it should outline immediate next steps. But the proposal is merely a guidepost and reasonable deviations in method, approach, and scope are expected.

Written report: summarize the topic, provide relevant background (theoretical or applied), timeline and contributions, and document challenges and extensions. It should provide discussion sufficient that an uninformed expert can understand the models, analytic decisions, outcomes, and implementation. Teams should provide quantifiable metrics to justify engineering tradeoffs, including performance benchmarks, scalability analysis, and resource utilization.

Technical review and demo: Approximately 15 minutes (depends on class size) to describe the topic problem and solution. It should provide only what is necessary to understand the what and why and include minimal theoretical background. The instructor *may* provide a *technical reference* slide-deck template that must be completed in advance of the demo session.

Source code: submitted as a GitHub repository archive file (zip). It must include README file(s) that describe the repository structure, execution instructions, deployment configuration, and infrastructure requirements.

Video: a 4-minute video that describes the topic, your implementation, and your results. You may choose to upload this to a video sharing site such as YouTube but that is not required.

Academic Accommodations

Any student requesting academic accommodations based on a disability is required to register with to Office of Student Accessibility Services (OSAS) each semester. A letter of verification for approved accommodations can be obtained from OSAS. Please be sure the letter is delivered to me as early in the semester as possible. OSAS is located in GFS 120 and is open 08:30 - 17:00, Monday through Friday. The phone number for DSP is (213) 740-0776.

Support Systems

A number of USC's schools provide support for students who need help with scholarly writing. Check with your advisor or program staff to find out more. Students whose primary language is not English should check with the *American Language Institute* http://dornsife.usc.edu/ali, which sponsors courses and workshops specifically for international graduate students. *The Office of Disability Services and Programs* http://sait.usc.edu/academicsupport/centerprograms/dsp/home_index.html provides certification for students with disabilities and helps arrange the relevant accommodations. If an officially declared emergency makes travel to campus infeasible, *USC Emergency Information* http://emergency.usc.edu will provide safety and other updates, including ways in which instruction will be continued by means of brightspace, teleconferencing, and other technology.

Discrimination, sexual assault, and harassment are not tolerated by the university. You are encouraged to report any incidents to the *Office of Civil Rights Compliance* http://ocrc.usc.edu or to the *Department of Public Safety* https://dps.usc.edu/contact/feedback/. This is important for the safety of the whole USC community. Another member of the university community - such as a friend, classmate, advisor, or faculty member - can help initiate the report, or can initiate the report on behalf of another person.

Academic Conduct

The University of Southern California is foremost a learning community committed to fostering successful scholars and researchers dedicated to the pursuit of knowledge and the transmission of ideas. Academic misconduct is in contrast to the university's mission to educate students through a broad array of first-rank academic, professional, and extracurricular programs and includes any act of dishonesty in the submission of academic work (either in draft or final form).

This course will follow the expectations for academic integrity as stated in the *USC Student Handbook*. All students are expected to submit assignments that are original work and prepared specifically for the course/section in this academic term. You may not submit work written by others or "recycle" work prepared for other courses without obtaining written permission from the instructor(s). Students suspected of engaging in academic misconduct will be reported to the *Office of Academic Integrity*.

Other violations of academic misconduct include, but are not limited to, cheating, plagiarism, fabrication (*e.g.*, falsifying data), knowingly assisting others in acts of academic dishonesty, and any act that gains or is intended to gain an unfair academic advantage.

Academic dishonesty has a far-reaching impact and is considered a serious offense against the university. Violations will result in a grade penalty, such as a failing grade on the assignment or in the course, and disciplinary action from the university itself, such as suspension or even expulsion.

For more information about academic integrity see the student handbook https://policy.usc.edu/studenthandbook/, or the Office of Academic Integrity's website https://academicintegrity.usc.edu/, and university policies on Research and Scholarship Misconduct https://policy.usc.edu/research-and-scholarship-misconduct/.