

# **Cloud Infrastructure and Scalable Application Deployment**

“If you’re at Stanford touting yourself an entrepreneurial genius with your app about antagonizing your classmates, you had better actually bring the bacon to your supposed technology. Because you have more than your founding team’s worth of classmates who hate your guts, but can actually program.”

– commenter “doctorpangloss” on Hacker News  
August 28, 2023

# **Course Information**

# About Us: Teaching Team



**Aditya Saligrama**

**Instructor** || [saligrama@stanford.edu](mailto:saligrama@stanford.edu)

BSCS '24, MSCS '25 (Systems/Security)  
SWE @ Verkada, Lacework, Uptycs, Akamai  
Applied Cyber President



**Cody Ho**

**Instructor** || [codyho@stanford.edu](mailto:codyho@stanford.edu)

Symsys '24, MSCS '25 (AI)  
ML @ OpenAI, Tatsui Lab, Infosys  
Applied Cyber Vice President



**Ben Tripp**

**Teaching Assistant** || [btripp@stanford.edu](mailto:btripp@stanford.edu)

Symsys '25 (AI)  
SWE @ Azure, CISA, DevOps & cloud consulting  
Applied Cyber Workshops Lead

# About Us: Advisors



**Mike Abbott**

**Advisor**

EVP Software & Services @ GM  
fmr VP Eng (Cloud Services) @ Apple, Twitter  
Instructor CS153, CS349D



**Christos Kozyrakis**

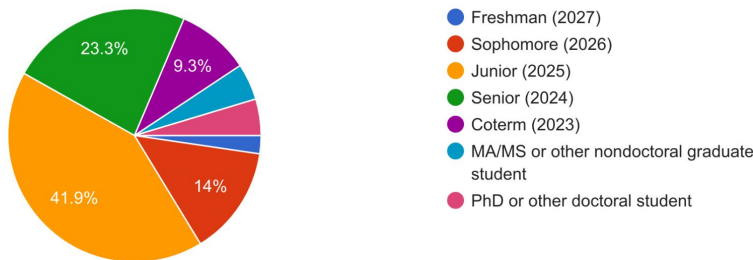
**Faculty Sponsor**

Professor EE/CS @ Stanford  
Cloud computing research  
Instructor EE180, CS349D

# About You

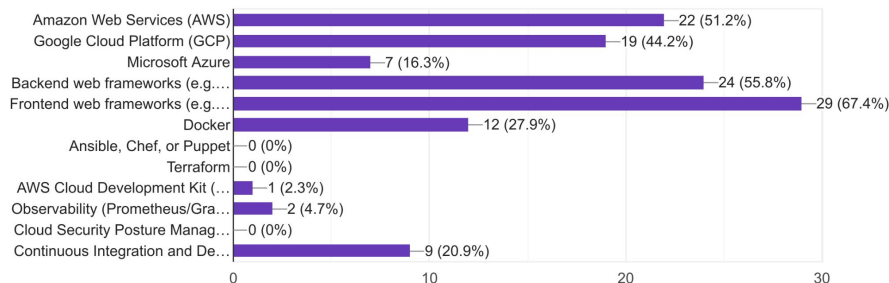
What class year / degree program are you?

43 responses



Which of the following technologies or frameworks have you used in the past? Again, none of these are required – don't worry if you haven't used any of these before.

43 responses



## Topics you're interested in:

- Kubernetes
- ML/AI applications
- Evaluating cost/performance across cloud environments
- Security, privacy, and observability

# Prerequisites

- Programming maturity up to CS107
- Familiarity with the **command line, version control, and basic development tools** to the level of CS45/CS104. Tools include:
  - TUI editors: **vim**, **emacs**, or **nano**
  - **Git/GitHub**, including branching and pull requests
  - Language and OS package managers: e.g., **pip**, **apt**, **homebrew**
- Prior web development or networking experience helpful but not required
  - We'll run a web basics section on **Wednesday 1/10 at 8pm (Zoom/recorded)**

**Please reach out if you have any questions about your readiness for CS40!**

**What you'll learn**



# Why CS40

- Stanford CS courses teach foundational technical skills, but less on building projects from scratch **with robust and scalable architecture**
  - Course final projects usually involve building web or mobile apps or Jupyter Notebooks
- Before CS40: “I built an app with a frontend and a backend”
- After CS40: “My app can handle lots of users at minimal cost”
- These skills are **broadly applicable to all technical careers** (academic, industrial, entrepreneurial) and even to hobby projects!

# Why Cloud

- Before ~2004, deploying web apps required buying **physical servers** or renting **datacenter** server capacity
  - Inflexible (what if you suddenly need more capacity?), and not very redundant
  - Needs a lot of **manual infrastructure management**: installing required software, keeping base software up to date, updating the application itself
- Today: pay-as-you-go for cloud resources
  - **Automatically adjust required capacity** in a very fine-grained way → cost reduction
  - Letting cloud providers **manage most of your resources** has security benefits, plus reduces engineering mindshare
  - Easy redundancy and availability close to customers → performance improvements

# Course Goals

You'll learn about:

1. What **resources** are made available by cloud providers that help in deploying applications
2. Architecting a cloud deployment by selecting resources for **optimal scaling** (performance) and **cost efficiency**
3. Systematically deploying cloud resources using **Infrastructure as Code** (IaC)
4. Ensuring your deployment remains **secure, observable, and continuously updated**

# **Building Blocks of Cloud Infrastructure**

# Types of Cloud Product Offerings

## Infrastructure as a Service (IaaS) (CS40 focus)





























































































































Lowest level of abstraction  
is directly above hardware  
(e.g., networks, storage, servers)

## Platform as a Service (PaaS)



Abstraction: all hosting  
infrastructure managed for you  
Just bring the code!

## What services do cloud providers offer?

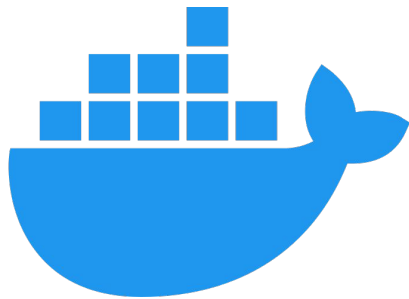
			
AI & ML	 Augmented AI  Forecast  Neuron  Fraud Detector  Comprehend  Elastic Inference  Lex  Personalize  Recommender  Lookout for Metrics  Polly  SageMaker  Transcribe  TensorFlow  PyTorch  Amazon Rekognition  Language Service  Personaliser  Form Recognizer  Cognitive Search  File API  Speech-to-Text  Computer Vision  Translator  AutoML  DialogFlow  Natural Language AI  Text-to-Speech  Vision AI  Video AI	 Bot Service  Amazon Rekognition  Language Service  Personaliser  Form Recognizer  Cognitive Search  File API  Speech-to-Text  Computer Vision  Translator	 App Engine  Cloud APIs  Apigee
Application – Mgmt	 Amplify  App Runner  Application Repository  API Gateway  AppSync  Elastic Beanstalk  App Mesh	 API Apps  Visual Studio App Centre  Service Fabric  API Management  App Service  Web Apps  Azure Maps  Cloud Services	 App Engine  Cloud APIs  Apigee
Application – Mobile	 Mobile Hub  Publisher Services	 Mobile Apps  Xamarin with Azure	 App Engine  Firebase
Automation	 CloudWorks	 Automation	
Compliance	 Artifact  Adapt Manager  Compliance Center	 Microsoft Compliance Manager  Trust Center  Container Instances  Container Registry  Azure Kubernetes Service (AKS)  Container Apps	 Amazon WorkSpaces  Compliance Manager  Kubernetes Engine
Compute – Container	 App2Container  Elastic Kubernetes Service (EKS)  Basic Container Engine (BCE)  Fargate	 Azure Functions  AutoScale  Batch  Virtual Machines  Compute Engine  VMware Engine	 Cloud Run  Kubernetes Engine
Compute – FaaS	 Lambda	 Azure Functions	 Cloud Functions
Compute – Server	 Auto Scaling  LightSail  Basic Container Engine (BCE)	 AutoScale  Batch  Virtual Machines  Compute Engine  VMware Engine	 Cloud Run  Kubernetes Engine
Cost	 Application Cost Profiler  Cost Anomaly Detection  Pricing Calculator  Billing  Budgets  Cost Explorer	 Advisor  Cost Management and Billing  Pricing Calculator  TCO Calculator	 Cost Management  Recommender  Pricing Calculator

# Cloud Concepts in CS40

- Building blocks
  - Compute (EC2, ECS)
  - Network (VPC, ACM)
  - Storage (S3, databases)
- Structuring cloud deployments
  - Infrastructure-as-Code (CDK)
  - Continuous Integration and Continuous Deployment (GitHub Actions)
  - Auditing, Security, and Observability (CloudWatch, CloudTrail, Datadog)
- Applications of cloud computing
  - Serverless compute (Lambda)
  - AI/ML pipelines (GPUs on EC2, Sagemaker)

# Containers

- **Definition:** a *container* is a portable software package containing all resources needed to run it, providing:
  - **Isolation:** processes of container A don't interfere with those of container B
  - **Replicability:** same process from same container image should execute the same on any host machine/OS/configuration
- A container behaves like its own isolated machine, but **shares its kernel with the host machine**
  - Cannot run directly on hardware: **needs to run on a real OS**
  - Manage the OS yourself, or cloud providers can abstract the OS layer for you, e.g. *AWS Elastic Container Service, Google Cloud Run*





# ***Demo:* Containers**

# Virtual Machines

- **Definition:** a *virtual machine* is a computer system created using software on one physical computer in order to emulate the functionality of another separate physical computer.
  - Virtual machines cannot interfere with the host or other virtual machines
  - Run their own kernel
  - Managed by a *hypervisor*
- Primary service offered by cloud providers:
  - AWS Elastic Compute Cloud (EC2)
  - Google Compute Engine
  - Azure VMs

# ***Demo:* Virtual Machines**

# Storage

- Object storage
  - Store arbitrary blobs: great for media, but also process inputs/outputs, logs, etc.
  - e.g. AWS S3, GCP Cloud Storage Buckets, Azure Blob Storage
- Block storage
  - Only get the base storage device, with no file abstraction
  - e.g. AWS EBS, GCP Persistent Disk, Azure Disk Storage
- Databases
  - Traditional SQL DBs, plus newer NoSQL DBs (e.g. document, KV, graph models)
  - AWS RDS/Aurora, AWS Neptune, GCP Database, Azure SQL Database, Azure Cosmos
  - Vector DBs, ChromaDB, Pinecone (PaaS)

# Network

- **Virtual Private Cloud (VPC)**: a logically isolated virtual network with controls for resource placement, connectivity, and security
- A VPC is located in one geographical region, but can span multiple availability zones (datacenters) in that region for redundancy and scalability
- We'll talk more about cloud networking in the next couple of lectures

# **Course Logistics**

# Course Structure

- Lectures: Mon, Wed 4:30pm-5:50pm, 530-127
  - Lecture attendance optional but encouraged; **must attend guest speakers' lectures**
  - Recordings are best-effort and only available in extenuating circumstances
- Four Assignments: hands-on, deployments to AWS
- Final project
  - More details to come

# Grading

**60%:** Assignments (15% each)

**40%:** Final project

**Guest Speaker Attendance is Mandatory**

**Don't stress about this!** Assignments are intended to be straightforward, and we'll release autograders so you can check your assignment score as you work



# Assignments (60%)

- Four assignments (15% each)
  - A1: Hello World, AWS account setup, CLI refresh, EC2/Nginx static webpage deployment (1/23)
  - A2: Intro to Infrastructure-as-Code; deploy a complex webapp to AWS (2/13)
  - A3: Lambdas, Observability, and AI/ML pipelines (2/27)
  - A4: Continuous Integration & Continuous Deployment (3/7)
- Mostly in Python, but significant shell usage + configuration files (JSON, YAML)
- Can work in pairs for assignments 2-4
- **Credits provided for all students after add/drop deadline**, look out for communication on Ed on how to access

# Final Project (40%)

- Open-ended (with suggested ideas you're free to use)
  - Deploy an application of your choosing to AWS!
- Due **3/17**, can be completed in pairs
- More details to come

# Participation

Required attendance at the following guest lectures:

- Benjamin Bercovitz (Co-Founder of Verkada), **Wed 1/24**
- Corey Quinn (Chief Cloud Economist, Duckbill Group), **Wed 3/6**
- Mark Russinovich (CTO of Microsoft Azure), **TBD**
- Bill Jia (Google Cloud, fmr VP of Engineering @ Meta), **TBD**

Every unexcused absence will drop you a letter grade. Absences must be excused before lecture or have extenuating circumstances.

# A few quick policy notes

- You MUST follow the letter and spirit of the Honor Code
  - Please reach out if you have any questions about our Honor Code policy
- You can use ChatGPT and other LLM's for reference, preferably as a last resort
  - Don't ask for entire assignment solutions
  - We use modern (2024) best practice constructs which may not be in LLM training data
  - **Detail LLM usage in your assignment writeups**
- If you find bugs in course infrastructure, please tell us!
  - At our discretion, we might give you some extra credit
  - But you still need to complete the assignments as specified

# Course Resources

- Course website: <https://cs40.stanford.edu>
- Ed discussion: <https://edstem.org/us/courses/50007/discussion/>
  - This is where all course communication will be handled
- Gradescope: **4GYR7J**
- GitHub:
  - Public resources: <https://github.com/infracourse>
  - Assignments 2-4 will be on a GitHub classroom within <https://github.com/cs40-24win>
- Office hours:
  - **Aditya**: Mon, Wed 10:00am-11:00am
  - **Cody**: Wed, Fri 12:00pm-1:00pm
  - **Ben**: Tue, Thu 1:00pm-2:00pm
  - Starting next Tuesday 1/16