






# Teaching Cloud Infrastructure in an undergraduate CS program

-  Aditya Saligrama [saligrama@cs.stanford.edu](mailto:saligrama@cs.stanford.edu)
-  Cody Ho [codyho@cs.stanford.edu](mailto:codyho@cs.stanford.edu)
-  Benjamin Tripp [btripp@cs.stanford.edu](mailto:btripp@cs.stanford.edu)
-  Michael Abbott [michael.abbott@acm.org](mailto:michael.abbott@acm.org)
-  Christos Kozyrakis [christos@cs.stanford.edu](mailto:christos@cs.stanford.edu)





## Aditya Saligrama

Stanford alum (BS & MS CS '24)  
Engineering @ Formal  
ex-Cloudflare, Verkada, Akamai



## Cody Ho

Stanford student (BS Symbolic  
Systems '24, MS CS '26)  
ML @ Apple, ex-OpenAI

# Motivation

# Why teach deployment principles?

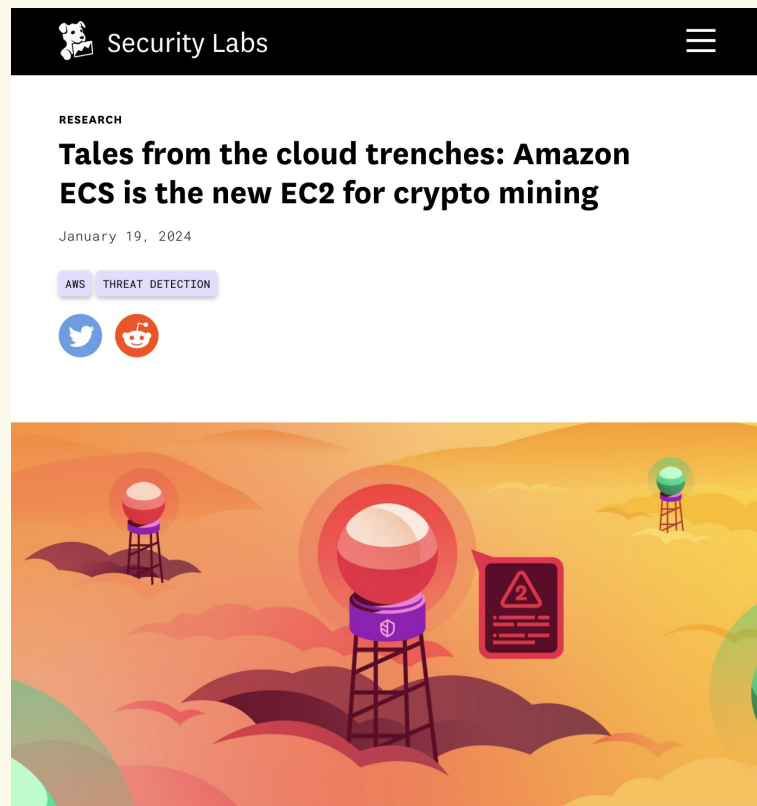
- CS curricula are good at teaching students to build applications
  - ...but, every good application needs to be deployed somewhere!
  - This is a **necessary skill** for any future career path – in academia, industry, and entrepreneurship.
  
- **Knowing where/how an application will run informs how it should be built**
  - Most new applications today run in the cloud, due to **cost, security, and scalability benefits**

# Cloud deployment in 2025

- **Significant maturation** of cloud platform service offerings since ~2019
  - Homogenization of service abstractions offered by “big three” providers (AWS, GCP, Azure)
  - ...allows us to treat deployment in a more principled, systematic way
  
- Broad concepts include:
  - Compute, network, and storage abstractions above hardware
  - Infrastructure as Code (IaC)
  - Containers and container orchestration
  - Functions as a Service (FaaS)
  - Identity and Access Management (IAM)

# The challenges of cloud deployment

- Cloud deployment requires an understanding of **scalability, security, and cost tradeoffs**, as well as application design concepts
- (e.g.) disk storage on container orchestrators (Kubernetes, ECS) is **ephemeral** → implications on application architecture



# Current landscape of cloud courses

- Most cloud courses are either:
  - **Online/self-paced courses** targeted towards reskilling mid-career professionals
  - **Graduate research seminars** on the historical innovations that power cloud computing platforms (like distributed systems)
  
- At the undergraduate level:
  - **DevOps or SRE courses** that incorporate cloud deployment, but lean more towards **software lifecycle management**
  - Courses on **web development** that mention cloud deployment practices at **shallow depth**
  
- **We saw a need for a course that interfaced with modern cloud abstractions, hands-on through IaC.**

# Curriculum



*By the end of the course, students should be able to:*

1. **Design cloud-native applications** to take advantage of the elasticity, cost, and system administration benefits provided by cloud infrastructure.
2. **Architect** a cloud deployment by selecting and combining appropriate **compute, networking, and storage** resources.
3. Leverage **IaC** to systematically and reliably deploy cloud services.
4. Ensure that cloud deployments remain **secure, observable, and continuously updated**.

# Lecture content

Week	Lecture 1	Lecture 2
1	Building Blocks of Cloud Infrastructure	Networking Primer; Web App Primer
2	<i>Holiday, no lecture</i>	Cloud Networking (VPC, Load Balancing, CDN)
3	Cloud Storage (Object Storage, Databases, Caches)	Database Tradeoffs for Cloud Applications*
4	Containerization and Container Orchestration	Infrastructure-as-Code and Cloud Automation
5	Identity and Access Management; Cloud Security Topics	Auditing, Logging, and Observability
6	Serverless Computing and FaaS	Cloud Machine Learning and High-Performance Computing
7	<i>Holiday, no lecture</i>	Continuous Integration and Continuous Deployment
8	Scaling Applications in Industry*	Practical Considerations in Cloud-Native Application Design
9	Ethical Issues in Cloud Computing	Cloud Billing and Cost Management*
10	Evolution of the Cloud Ecosystem and Cloud-Native Design*	Recap and Misc Topics

\* *guest lecture*

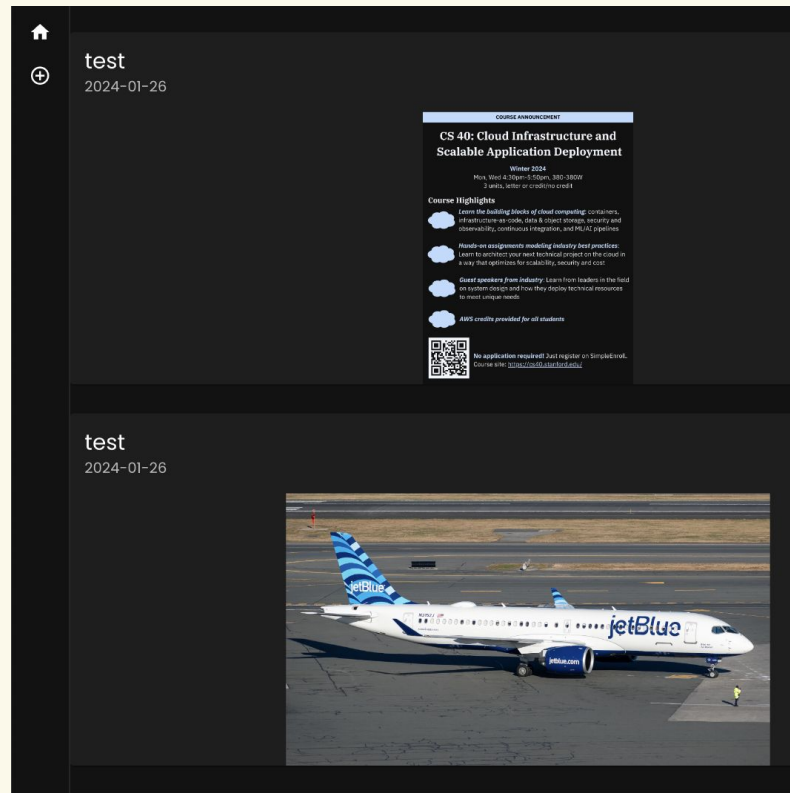
# Guest lectures



# Assignments

# yoctogram

- Custom-built, minimal image-sharing web app
- Designed to be representative of modern applications and easy to deploy at scale



# Technical assignments

- Student deployment to AWS, via **AWS CDK** in Python
  - Idea: provide **gentle introduction** to cloud in a language students are already familiar with
- Goal: expose students hands-on to a variety of commonly used cloud abstractions

## Assignment 2: Bring up Yoctogram

- IaC
- Virtual machines + Containers
- Cloud security & IAM
- Databases and Object storage
- Networking

## Assignment 3: Observability & Event-Driven Image Compression

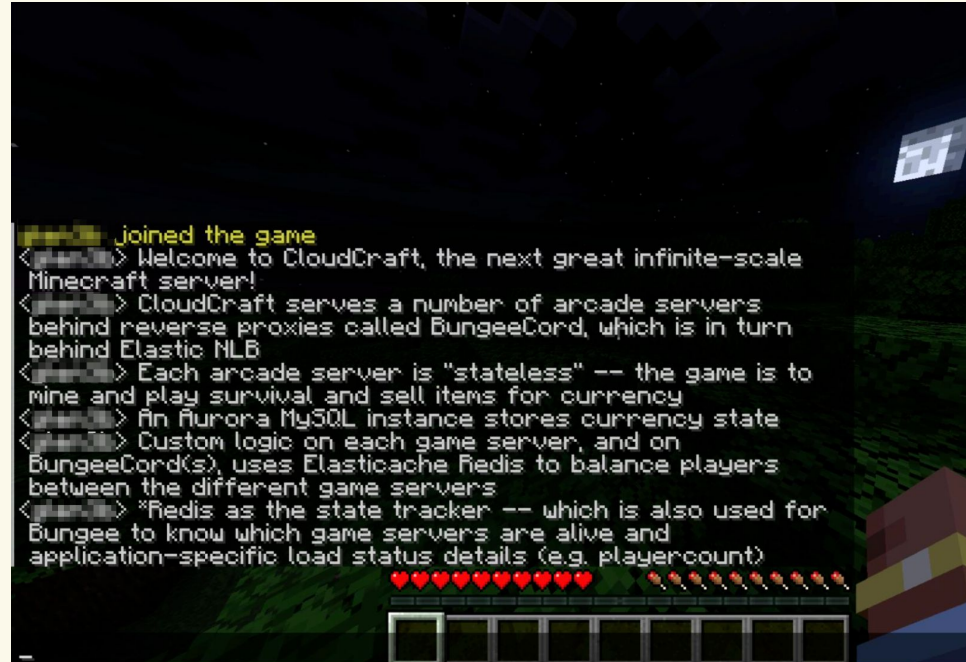
- Functions as a Service
- Logs, traces, and other telemetry
- Debugging using network traces

## Assignment 4: Continuous Integration & Deployment

- Repository workflow automation
- Building and managing container images
- Container registries

# Final project

- Deploy an application of the student's choosing to the cloud
  - Either off-the-shelf (e.g. Wordpress) or custom-developed by the student
- Explicit requirements:
  - Err towards managed services
  - Use IaC
  - Apply cloud security best practices
- Goal: have students demonstrate a general understanding of cloud services applied beyond Yoctogram

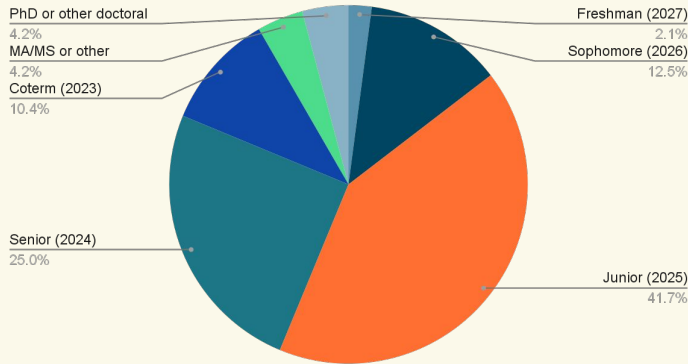


# Reflections



# Course demographics

Class Breakdown



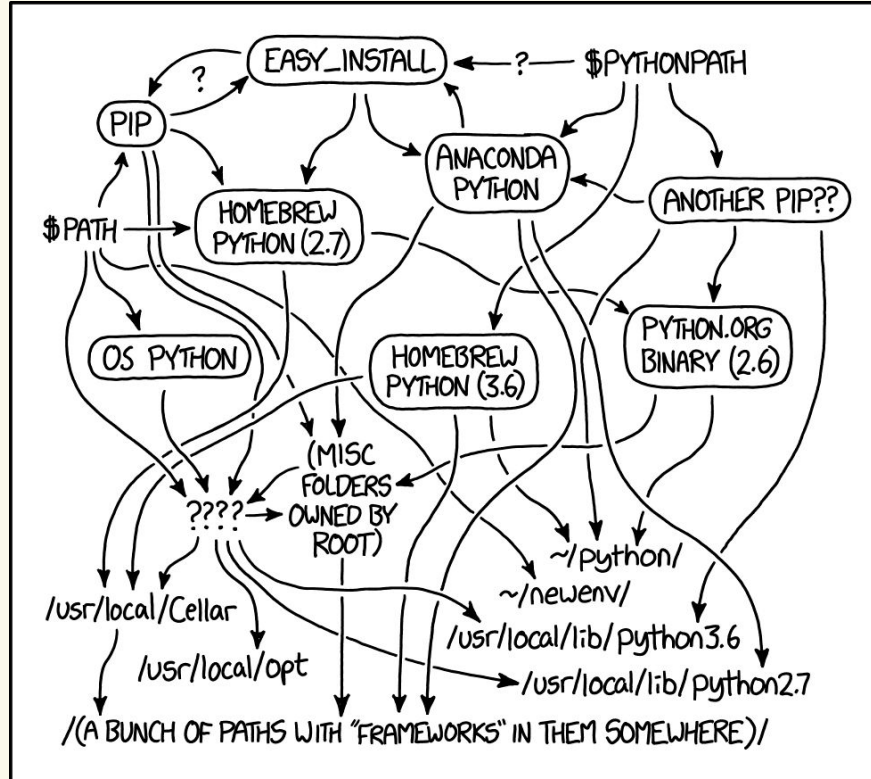
- 50 students total
  - 15 women
  - 7 from underserved minority backgrounds

Open form feedback (of 39 responding students):

- 72% learned "a lot" or "a great deal"
- Many reported "actual confidence [in their] ability to use the cloud intelligently"
- More time consuming and difficult than students expected

# Successful use of custom infrastructure

- **Challenge:**



# Successful use of custom infrastructure

- **Challenge:** Dependency challenges with student development on local laptops
  - **Solution:** Distribute prebuilt development image (for EC2 VMs) containing all required tools

The screenshot displays the AWS Marketplace console interface. At the top, it shows the selected AMI ID: (ami-0823668bf49fe5bae) (Quickstart AMIs). Below this is a search bar containing the query 'ami-02ffb1f43e49576e5'. The console is divided into four tabs: 'Quickstart AMIs (0)', 'My AMIs (0)', 'AWS Marketplace AMIs (9123)', and 'Community AMIs (1)'. The 'Community AMIs (1)' tab is active. On the left, there is a 'Refine results' sidebar with a 'Clear all filters' button and a filter for 'Operating system' expanded to 'Linux/Unix', with sub-options for 'All Linux/Unix', 'Amazon Linux', 'CentOS', 'Debian', and 'Fedora'. The main content area shows the search results for 'ami-02ffb1f43e49576e5 (1 filtered, 1 unfiltered)'. A blue information box explains that 'Community AMIs contain all AMIs that are public, therefore anyone can publish an AMI and it will show in this catalog. This catalog can also contain paid products. When using community AMIs it is best practice to ensure you know and trust the publisher before launching an AMI.' Below this, the specific AMI is listed: 'cs40-assignment2-ubuntu-22.04-lts-with-tools-1706234112' with ID 'ami-02ffb1f43e49576e5'. The AMI is associated with the 'ubuntu' logo. Metadata includes 'OwnerAlias: - Platform: Ubuntu Architecture: arm64 Owner: 315266155408 Publish date: 2024-01-26 Root device type: ebs Virtualization: hvm ENA enabled: Yes'. An orange 'Select' button is positioned to the right of the AMI details.

# Successful use of custom infrastructure

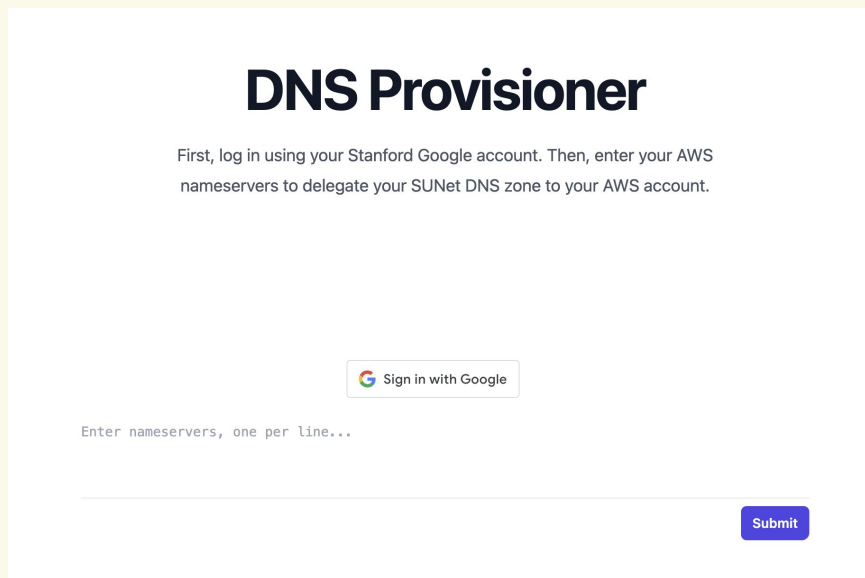
- **Challenge:** No existing solutions to grade an infrastructure deployment
  - **Solution:** Custom (open source) autograder using a mix of static (Open Policy Agent) and dynamic checks

```
# certificate should be issued for domain name SUNET.infracourse.cloud
fail contains msg if {
    certs := [cert | cert := input.Resources[_]; cert.Type == "AWS::CertificateManager::Certificate"]
    domain := certs[_].Properties.DomainName
    not endswith(domain, ".infracourse.cloud")

    msg := sprintf("ACM certificate should be issued for domain name SUNET.infracourse.cloud", [])
}
```

# Successful use of custom infrastructure

- **Challenge:** Students need to publicly deploy applications using TLS/HTTPS, but without paying for a domain name
  - **Solution:** Custom (open source) utility that delegates control of the namespace `studentid.infracourse.cloud` to the student's AWS account



The screenshot shows a web interface titled "DNS Provisioner". The main heading is "DNS Provisioner" in a large, bold, black font. Below the heading, there is a paragraph of text: "First, log in using your Stanford Google account. Then, enter your AWS nameservers to delegate your SUNet DNS zone to your AWS account." In the center of the page, there is a "Sign in with Google" button with the Google logo. Below this button, there is a text input field with the placeholder text "Enter nameservers, one per line...". At the bottom right of the form, there is a blue "Submit" button.

# Building domain-specific debugging skills

- Debugging requires testing **falsifiable hypotheses** against system behavior
  - **Many moving parts** to reason about with cloud-deployed systems
- Cloud providers don't make this easy
  - **Cryptic errors** and **silent failures**
  - Deployment takes time → **slow iteration**
- Some scaffolding might help:
  - Better application-level logging
  - More static testing to catch obvious, non-pedagogically-relevant errors

Timestamp	Logical ID	Status	Status reason
2024-02-13 12:47:21 UTC-0800	yoctogramfrontenddistribution243B0E2D	⊗ CREATE_FAILED	Resource creation cancelled
2024-02-13 12:47:17 UTC-0800	APIGatewayRecord1C7A9AC4	⊗ CREATE_FAILED	Resource creation cancelled
2024-02-13 12:47:17 UTC-0800	yoctogramfargateserviceServiceD B9282DF	⊗ CREATE_FAILED	Resource handler returned message: "Invalid request provided: CreateService error: The target group with targetGroupArn arn:aws:elasticloadbalancing:us-west-2:654654636638:targetgroup/yocto-yocto-JLCAEBXALTMR/e3ff37e4fcc98dd does not have an associated load balancer. (Service: AmazonEC2; Status Code: 400; Error Code: InvalidParameterException; Request ID: 143f10e3-a9ab-4509-8d35-655527099f72; Proxy: null)" (RequestToken: 11004c50-a501-02eb-d714-294169944497, HandlerErrorCode: InvalidRequest)
2024-02-13 12:47:15 UTC-0800	yoctogramfargateserviceServiceD B9282DF	⊙ CREATE_IN_PROGRESS	-

# Course scaling vs architectural freedom

- We kept technical assignments focused on infrastructure deployment, abstracting away application logic
  - Constraints of grading infrastructure at scale removed some opportunities for students to **experiment with architecture-level decisions**
- In the future:
  - Grade based on **performance and reliability** with less specific architecture testing
  - Let students interface with more **application ↔ infrastructure interactions** (e.g., build an application feature using a cloud SDK)

# Questions?

<https://infracourse.cloud>

<https://github.com/infracourse>



Link to paper