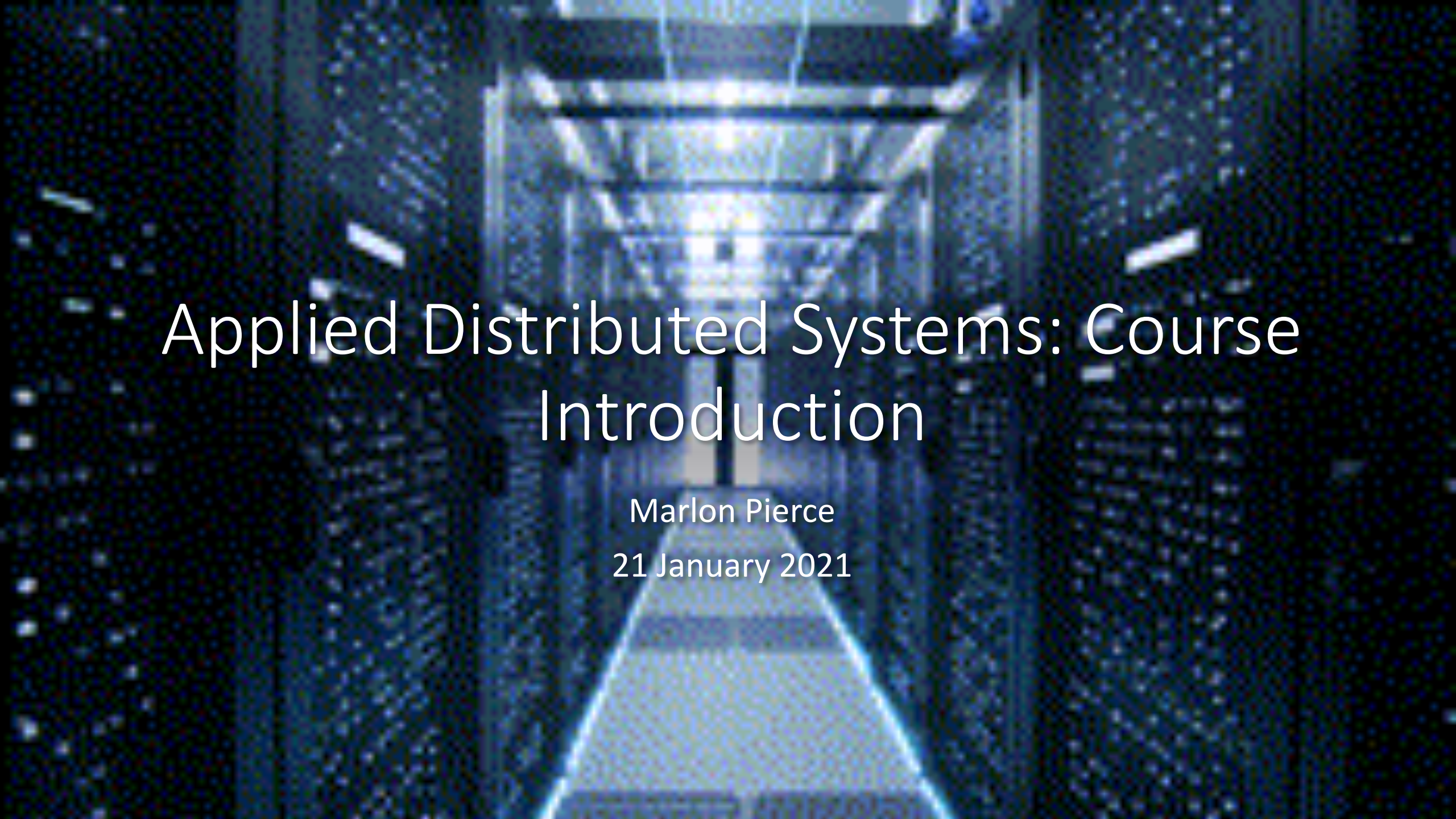


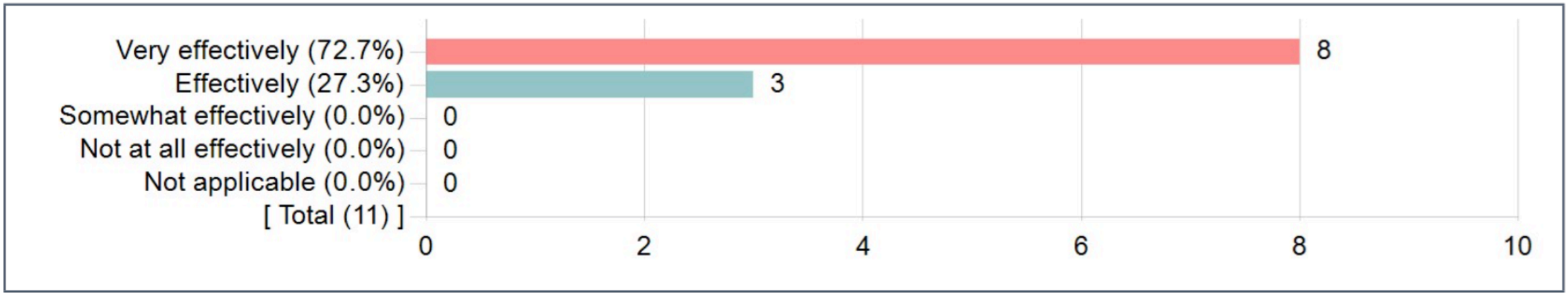
Applied Distributed Systems: Course Introduction

Marlon Pierce

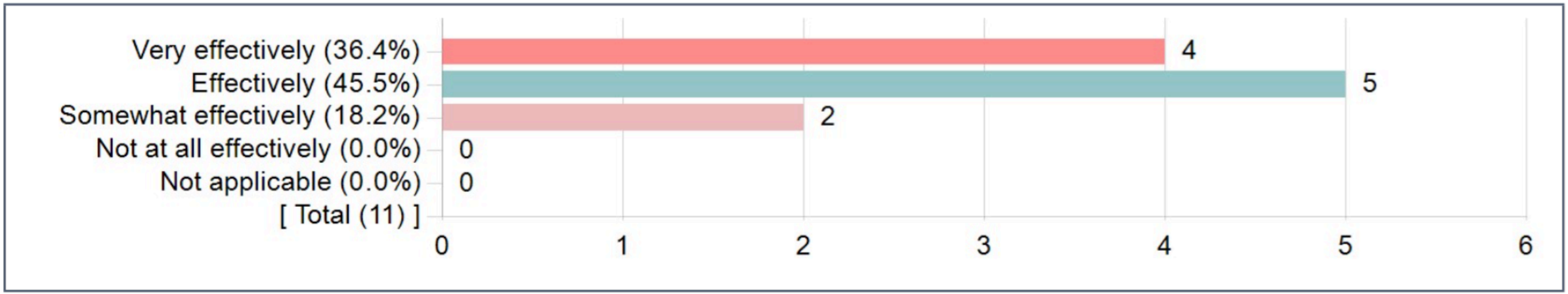
21 January 2021

What to Expect

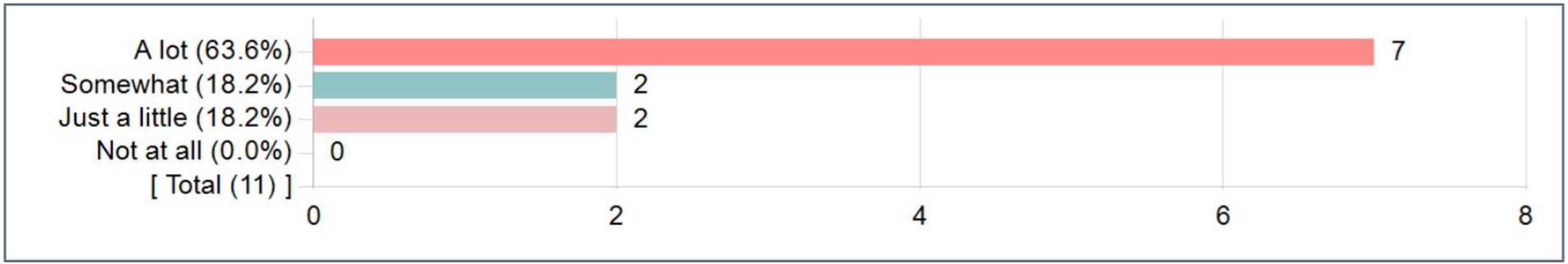
How effectively did out-of-class work (assignments, readings, practice, etc.) help you learn?



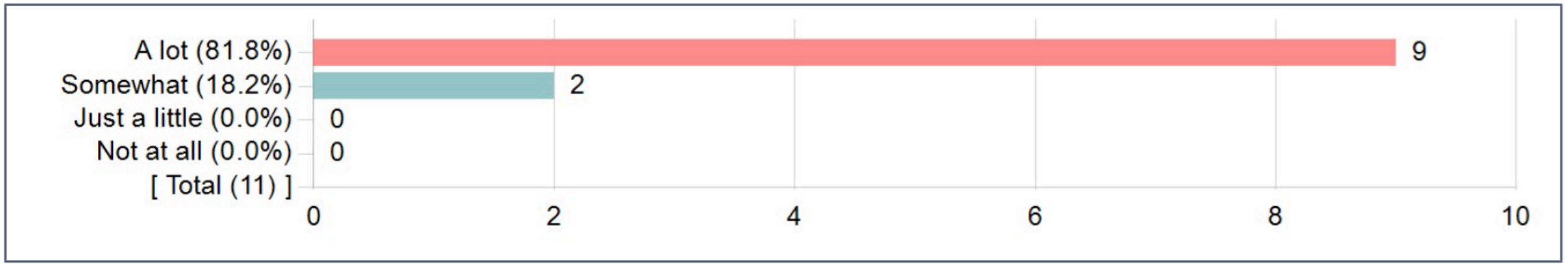
How effectively was class time used to help you learn?



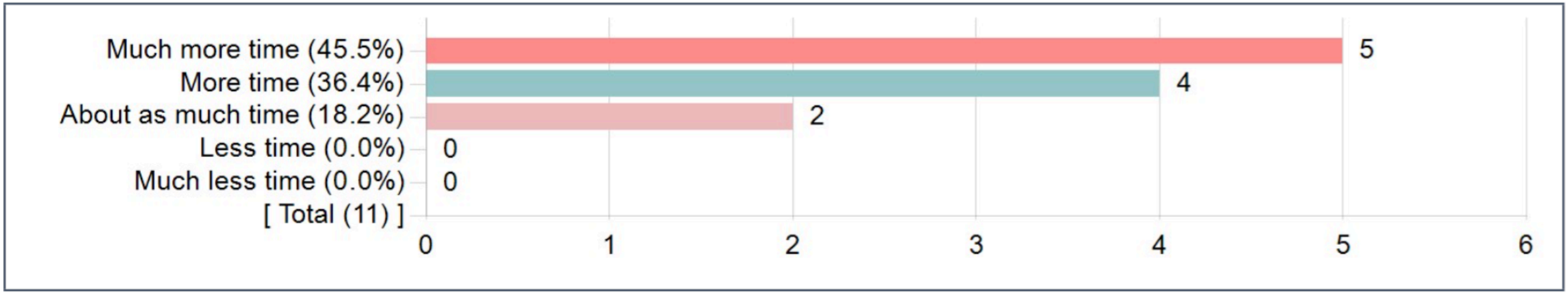
How much did the instructor motivate you to do your best work?



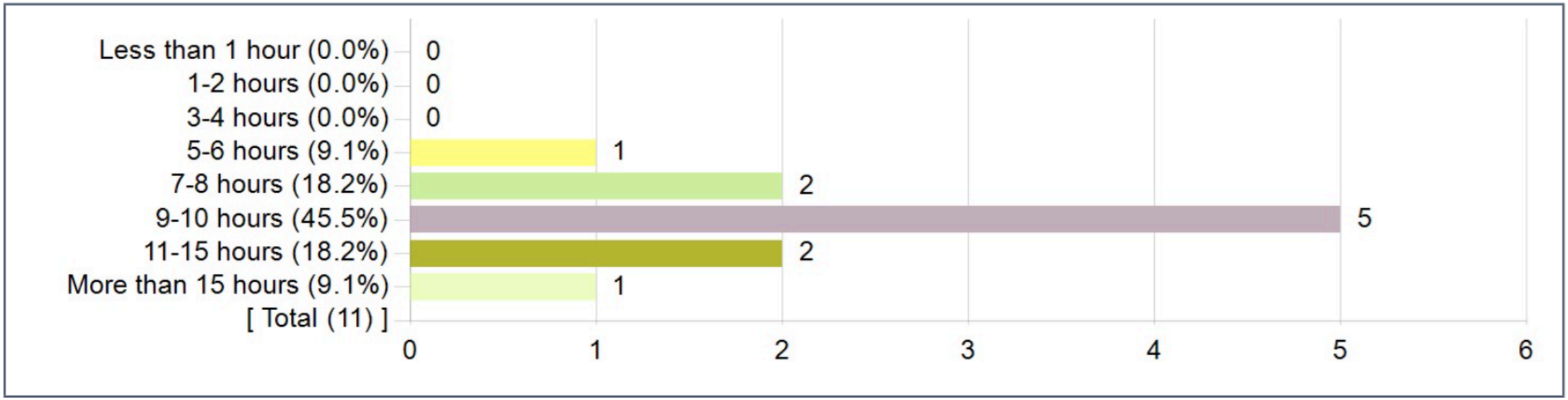
How much did the instructor emphasize student learning and development?



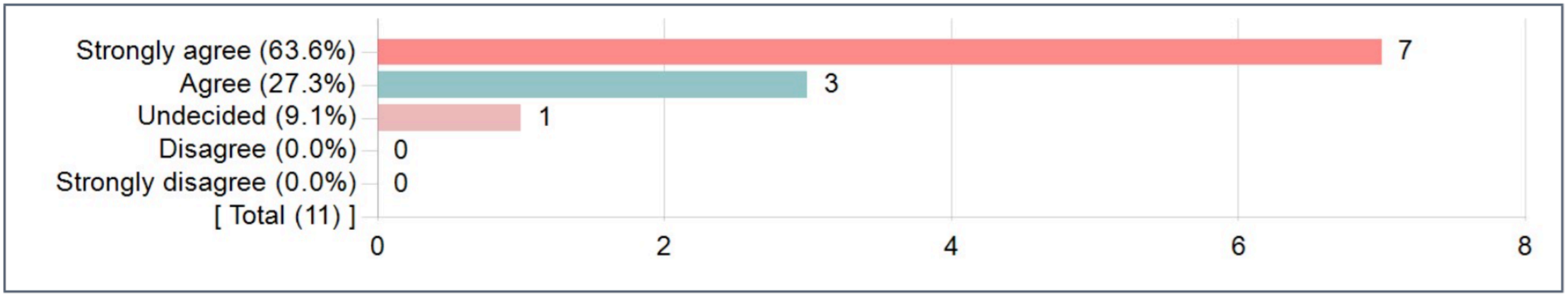
Compared to other courses you've taken, how much time did this course require?



In a typical week, about how much time did you devote to this course? (Do not count scheduled class time, labs, etc.)



I developed skill in critical thinking in this course.



What did you like most about this course and instructor?

Comments
Learning so many tools.
Best thing about projects is that I got to play around with instances on jetstream and there is no spoon feeding on how to go about setting it up.I learnt a lot in the process.
Explains in comprehensive manner
Topic Explainations were clear
Was able to learn new tools and technologies. The course was flexible with the project ideas being chosen. Ample time was given to work on assignments

What Is This Class About?

This Class Is Not About Building a Photo App

Read the Course Description Carefully

- https://courses.airavata.org/

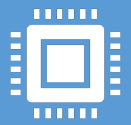
What Does “Cloud Native” Mean?

Cloud Native *X*

Architecture: How do you design a scalable distributed system?

Development: How do build a distributed system when you still do most of your development on a laptop?

Engineering: How do you deploy and operate at scale?

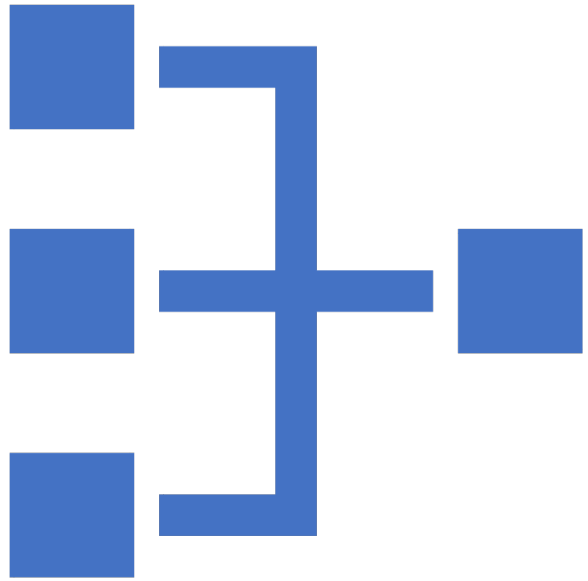


User Environments: How do you design user environments that are as scalable and flexible as the underlying systems?

Distributed Systems <--> Cloud Native

- Many “cloud native” services are pushing distributed systems to new scales
- Companies like Netflix have pioneered the use of 100’s of services and 1000’s of service instances
- Internet of Things systems will push this higher by orders of magnitude

What Are Distributed Systems?



Properties of Distributed Systems

Workloads are distributed over multiple independent entities (processes, servers, agents, ...)

Entities communicate with each other using messages sent over network connections

Why Do This? Compared to Monolithic Systems, Distributed Systems Are...

Potentially better performing

Potentially more scalable

Potentially more fault tolerant

Potentially built by integrating standalone or reusable capabilities

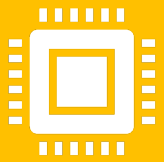
Potentially built using a range of programming languages, frameworks, etc

Potentially evolvable

Distributed Systems Are Hard to Build

Network disruptions will happen, knocking parts of your system offline

Distributed transactions and other coordinated activities are difficult



Security of and between components must be established

Fallacies of Distributed Systems

- The network is reliable
- Latency is zero
- Bandwidth is infinite
- The network is secure
- Network topology doesn't change
- There is one network administrator
- Transport cost is zero
- The network is homogeneous

https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing

What are we to do?

Answer: Build on Foundations

The order is important

Network Messaging, Messaging Patterns, Protocols

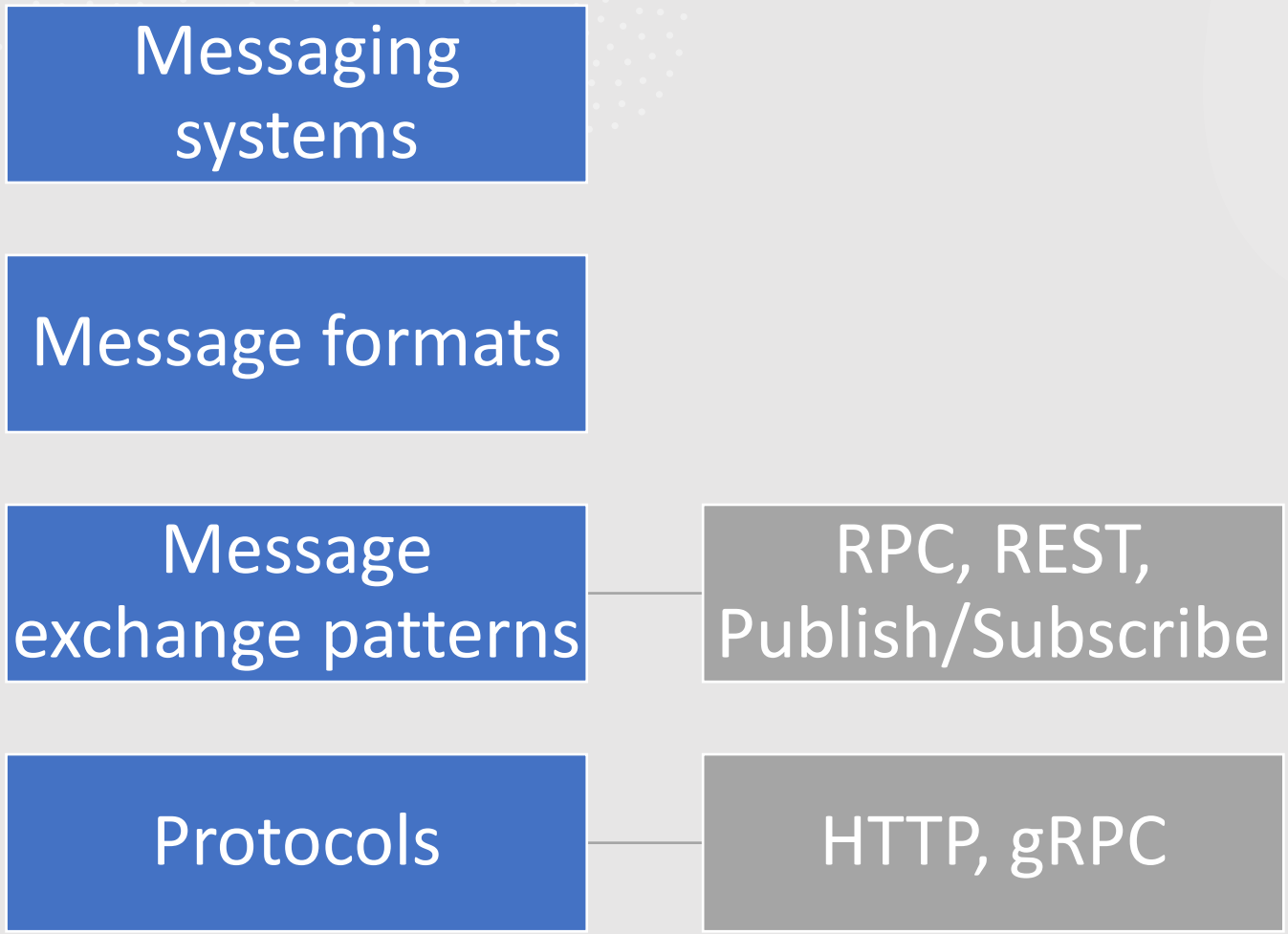
Algorithms

Design Patterns

Engineering Practices

Tools

Protocols, Messaging, Message Exchange Patterns



Algorithms and Protocols

RAFT

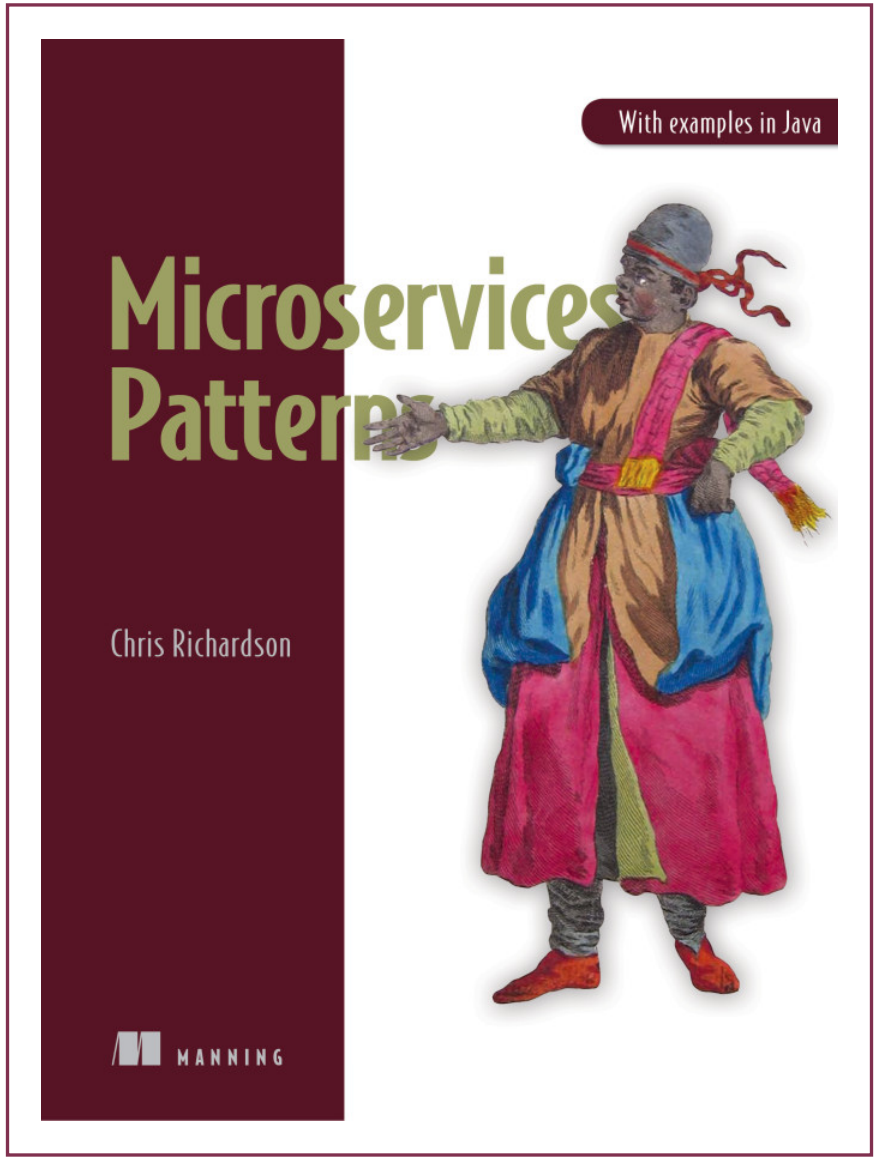
SWIM

Blockchain

OAuth2, OpenID Connect

Distributed System Design Patterns

- Microservices
  - https://microservices.io/
- Service Meshes
  - Sidecar Proxies
  - Data plane
  - Control plane



Engineering Processes and Practices

Open-source practices: GitHub to Governance

Developing telescoping applications

Deploying at scale: continuous integration and deployment

Operating at scale

Tools for Building Distributed Systems

Apache Kafka: Reliable messaging

HashiCorp Consul: Registries, control plane services

Docker: Developing portable application components

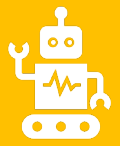
Kubernetes: Container runtime management

Istio, Envoy, LinkerD: Service Mesh

Why Do We Teach This Class?

Suresh and I are not computer scientists.

I’m personally not professionally interested in e-commerce, financial technology, streaming entertainment, social network platforms, etc.



We like to build systems that support scientific research at scale: cyberinfrastructure or e-science

To build cyberinfrastructure system, we need to understand how modern systems are built outside of academia.

Opportunities

- We build lots of systems called science gateways
- We do this by operating a platform called SciGaP
- We base the platform on Apache Airavata
- We work with outstanding students
  - Google Summer of Code via Apache Software Foundation
  - Advance class
  - Hourly employees
  - Graduate research assistants