

The Modern Software Developer

CS146S
Stanford University, Fall 2025
Mihail Eric

Guest Lecture - 11/21/25



CTO of [Resolve](#)
Mayank Agarwal

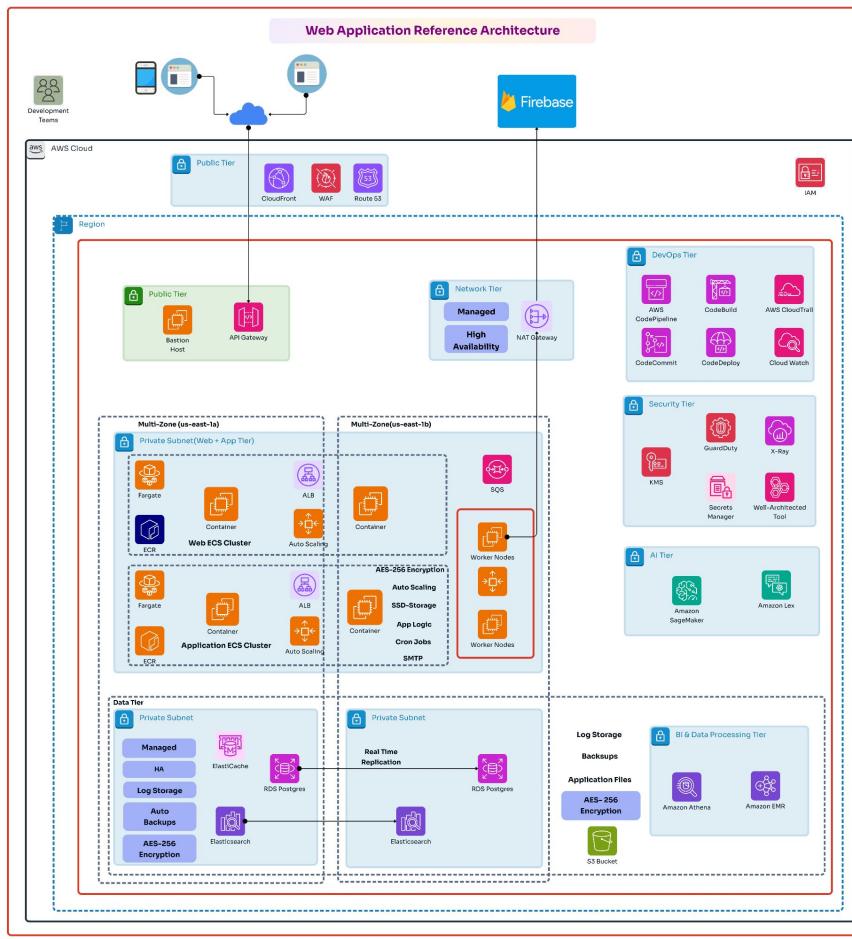


Member of Technical Staff [Resolve](#)
Milind Ganjoo

AI DevOps

Why

- Monitoring software systems in production remains a mission-critical task
 - Coding represents just 30 percent of engineering time
 - Harder 70 percent is running that code in production where complexity, tool silos, knowledge gaps, and interdependencies all collide
- Managing production often requires laborious software triaging handled by SREs



The Old World

- Responsibilities of SRE
 - Operational monitoring
 - on-call, troubleshooting, infrastructure management and security
- Incident resolution involves piecing together info from many different sources and teams
- Maintain oftentimes outdated runbooks for how to resolve issues
- Shift to cloud-native architectures with containerized workloads and Kubernetes has introduced more data, dependencies, and complexity across systems
- SREs often get burned out due to on-call shifts

Principles of Infrastructure and DevOps

- Four golden signals of monitoring
 - Latency
 - Errors
 - Traffic
 - Requests/sec
 - Saturation
- Monitor production traces

**At 3:12 am you get a ping from PagerDuty
that you're seeing a spike in 500s on your
database queries. What do we do?**

A Potential Playbook

- Acknowledge and assess
- Check DB + app
- Identify recent changes
- Localize blast radius
- Execute mitigations
- Stabilize + monitor
- Communicate
- Document

GETTING ON CALL PING AT 3AM



Metrics Tracked

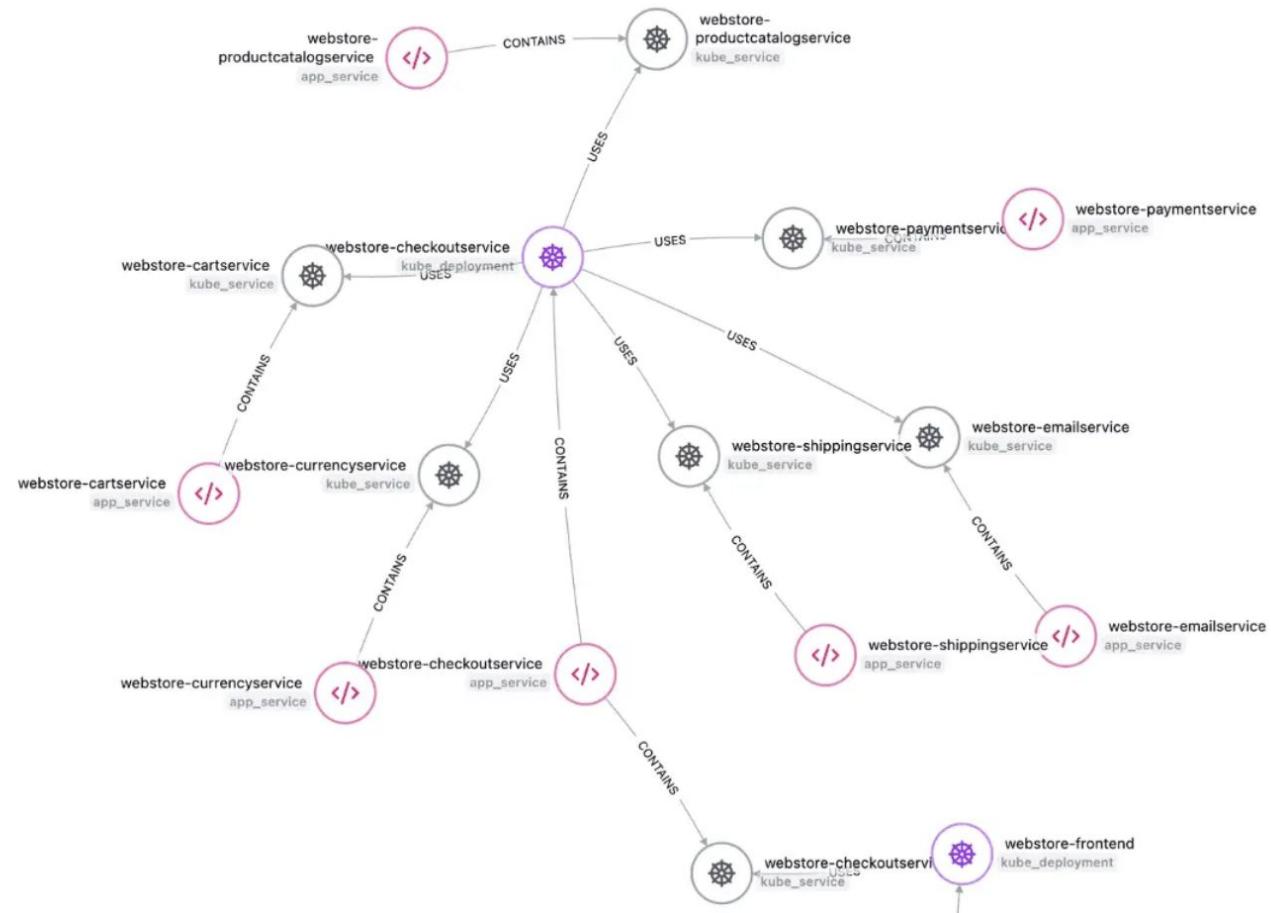
- Mean-time-to-repair (MTTR)
- How many engineers are pulled into incident
- Reported SLA for customers

The New AI World

- **Resolve AI** (guest lecture 11/21)
- DataDog Bits AI Agent
- Splunk Observability Assistant

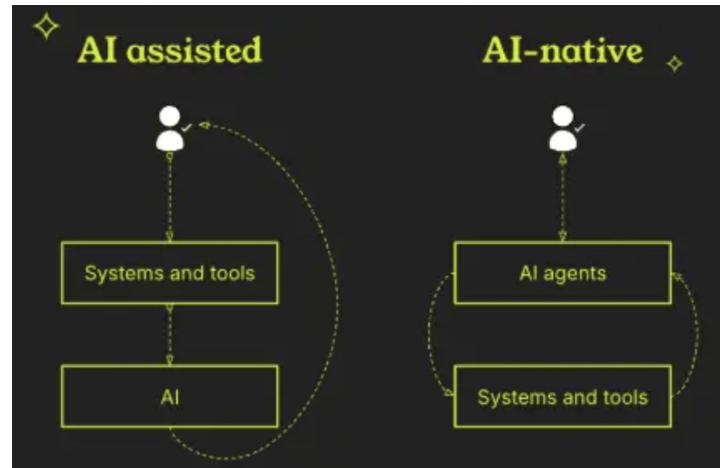
Characteristics of an AI SRE

- Dynamic mapping of a knowledge graph
- Agentic system across observability stack and clouds
- Generates real-time narratives of what is happening, pinpoints likely root causes with supporting evidence, and recommends prescriptive remediation steps
- Heavy emphasis on explainability and auditability of predictions/reasoning



What Has Changed

- AI promises to scale out organizational and service level knowledge
 - Information is not siloed to the only engineers who know the undocumented dependencies, brittle legacy services, and quirks that only surface during high-stakes incidents.



What Has Changed

- Automation reduces review time and catches issues early
- Developers learn best practices through AI suggestions
- AI applies the same standards across all code reviews
- AI handles routine checks, letting humans focus on complex logic
- AI systems improve over time with more data
- modern AI code review tools go deeper, offering contextual analysis and pattern recognition

Let's See AI SRE in Action

Limitations

- Complexity of incidents that can be resolved
- Heterogeneity of modern production stacks
- Ability to remediate actual code based on what has been detected
 - Eventual goal though all providers are starting with root cause analysis
- Good root cause analysis requires good monitoring gardening
- Security could be a new attack vector

Questions?