

AWS State, Local, and Education Learning Days

Boston

Cloud Foundations

10:15am – 11:15am

200
level

Cloud architectural patterns:

Master Cloud Architecture: Build Secure, Scalable Solutions with AWS Best Practices and Enterprise-Grade Design Strategies.

11:30am – 12:30pm

200
level

Cultural Transformation Through FinOps:

Go Beyond Cost Management: FinOps unites teams to optimize cloud costs while driving efficient growth

1:30pm – 3:00pm

200
level

Cloud Lab Potpourri

Hands-On Cloud Adventure: Customize Your AWS Learning with Interactive Technical Workshops and Personalized Lab Experiences.

3:15pm – 4:15pm

300
level

Designing modern applications in AWS

Dive into Culture, Processes and Tools to: Reduce Costs, Boost Scalability, and Enhance Security with Cloud-Native Architectures.





Designing modern applications in AWS

A dive into culture, tools, processes

Leo Zhadanovsky

Chief Technologist,
Education/SLG/Elections, WWPS
Amazon Web Services

leozh@amazon.com

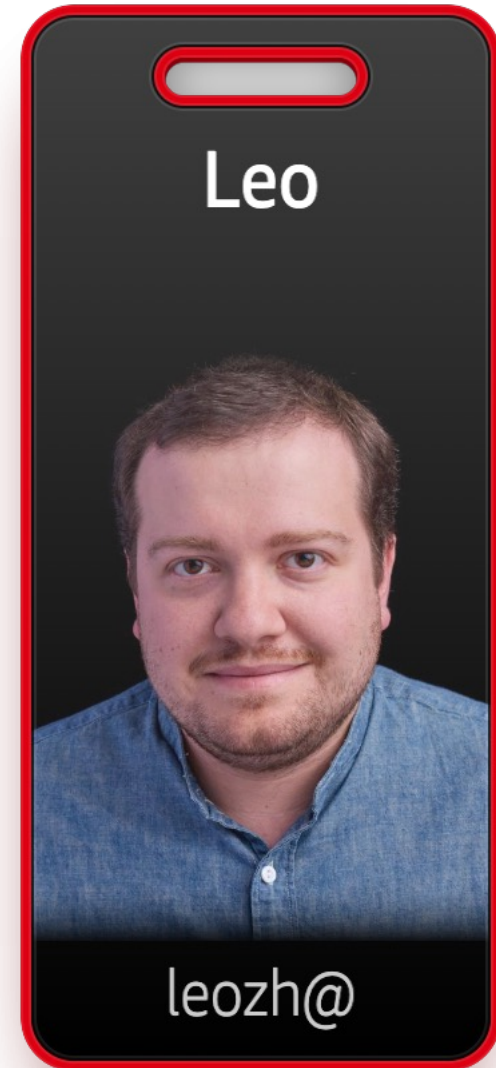
Vlad Fatu

Manager of Solutions Architecture,
Education, WWPS
Amazon Web Services

vladfatu@amazon.com

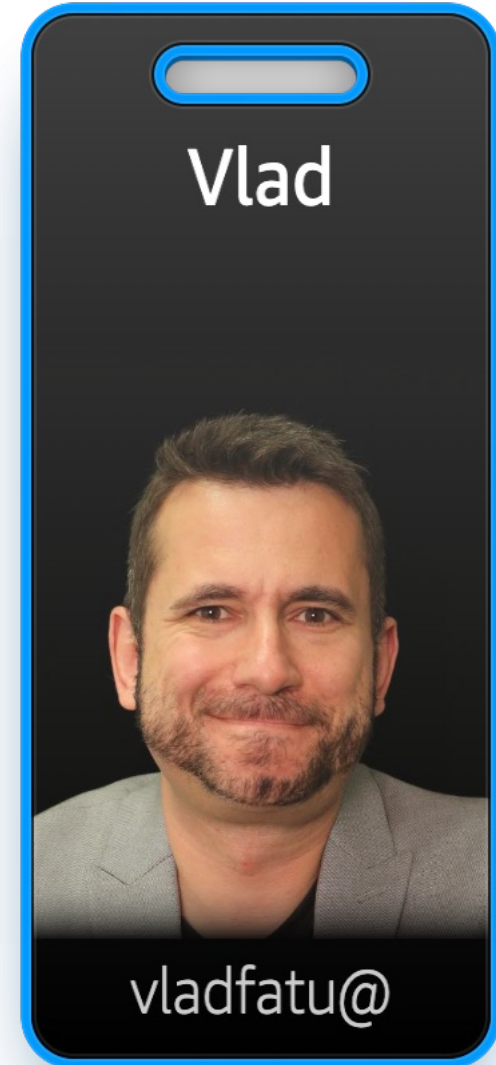
Who am I?

- AWS tenure: 12 years
- Run Critical Impact Engagements for AWS WWPS
- Often delivers talks about many topics, including DevOps at AWS and industry conferences
- Work with Education, SLG and Elections Customers
- Technical Areas of Expertise: DevOps, Information Security
- Enjoys traveling/living on a plane and Star Trek



Who am I?

- AWS tenure: 4.5 years
- WWPS EDU Northeast Leader
- Multidisciplined Geek / T-shaped Engineer
- Led tech teams in HED, HC, DIB
- Technical Areas of Expertise: Governance, Security, Organizational Transformation



Key takeaways

- Aligning culture to support building modern applications is the most important enabler
- Achieving operational excellence is foundational to adopting modern application best practices
- We treat ops as an investment, not a cost
- We align incentives for operational ownership with builders
- We examine our operations together, regularly

Architecting on AWS is different

- It's not just about stringing together services, but about building scalable, elastic, resilient, secure, reliable and cost-efficient solutions using managed cloud-native services
- Leverage cloud-scale, robust infrastructure
- The cost model frees architects up to do more & more cost-efficiently; promotes innovation
- Emphasis on composable architectures of distributed, modular & reusable components; generally service-oriented
- Servers/hosts are no longer the atomic unit of architecture
- Infrastructure-as-code: infrastructure as cattle, not pets
- Hyperautomation as a strategy
- Expanded integration and orchestration pathways (asynchronous, event-driven, "everything-as-an-API" decoupling)

Agenda*

Story: deployment automation at Amazon

How does Amazon do operations?

Mechanisms and Tools

- Help teams learn from each other
- Give teams tools to be successful
- Inform teams proactively about improvements

* We don't need to use any of this content. Ask questions, interrupt, and steer the conversation!

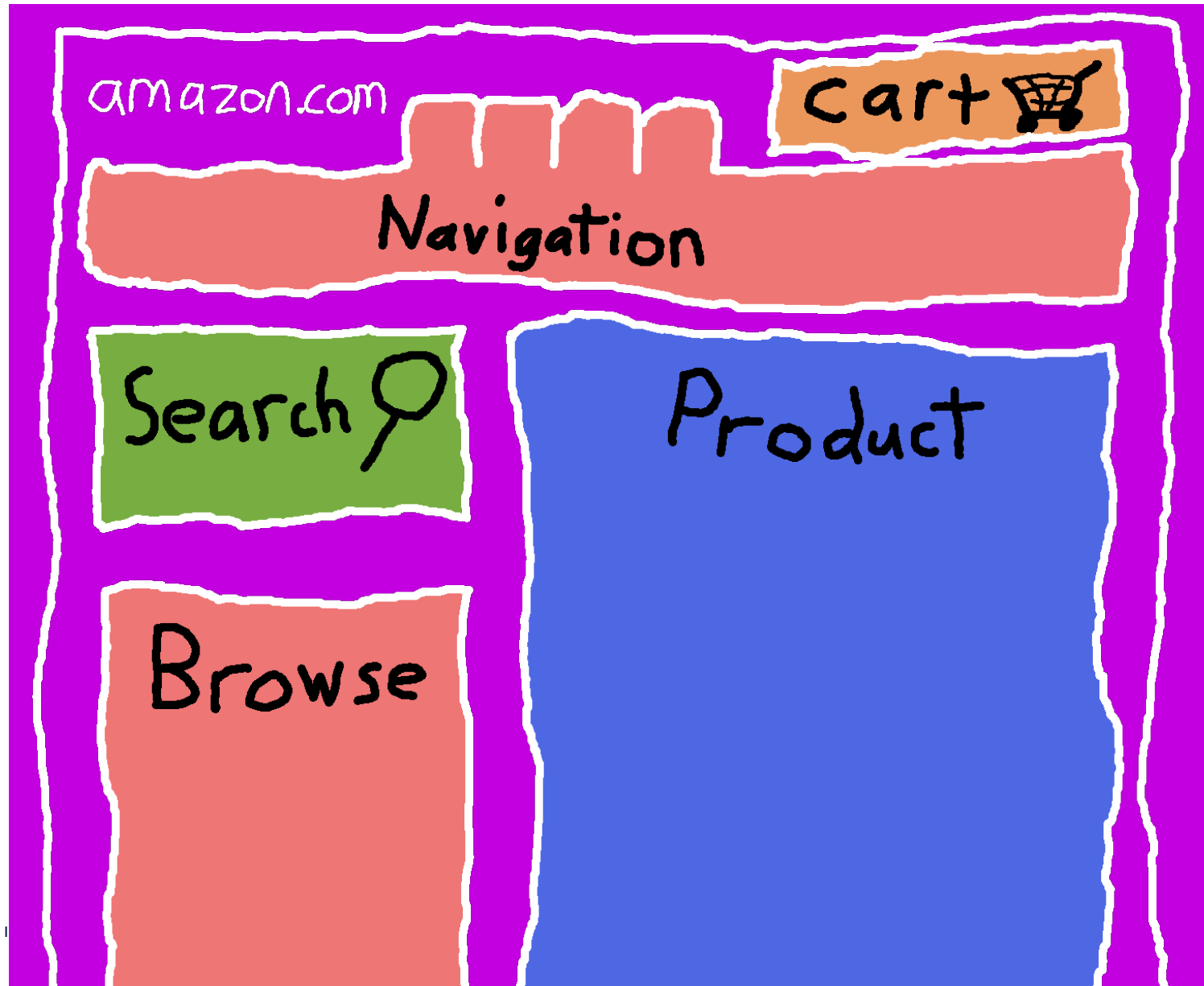
Scan QR code
for live feedback



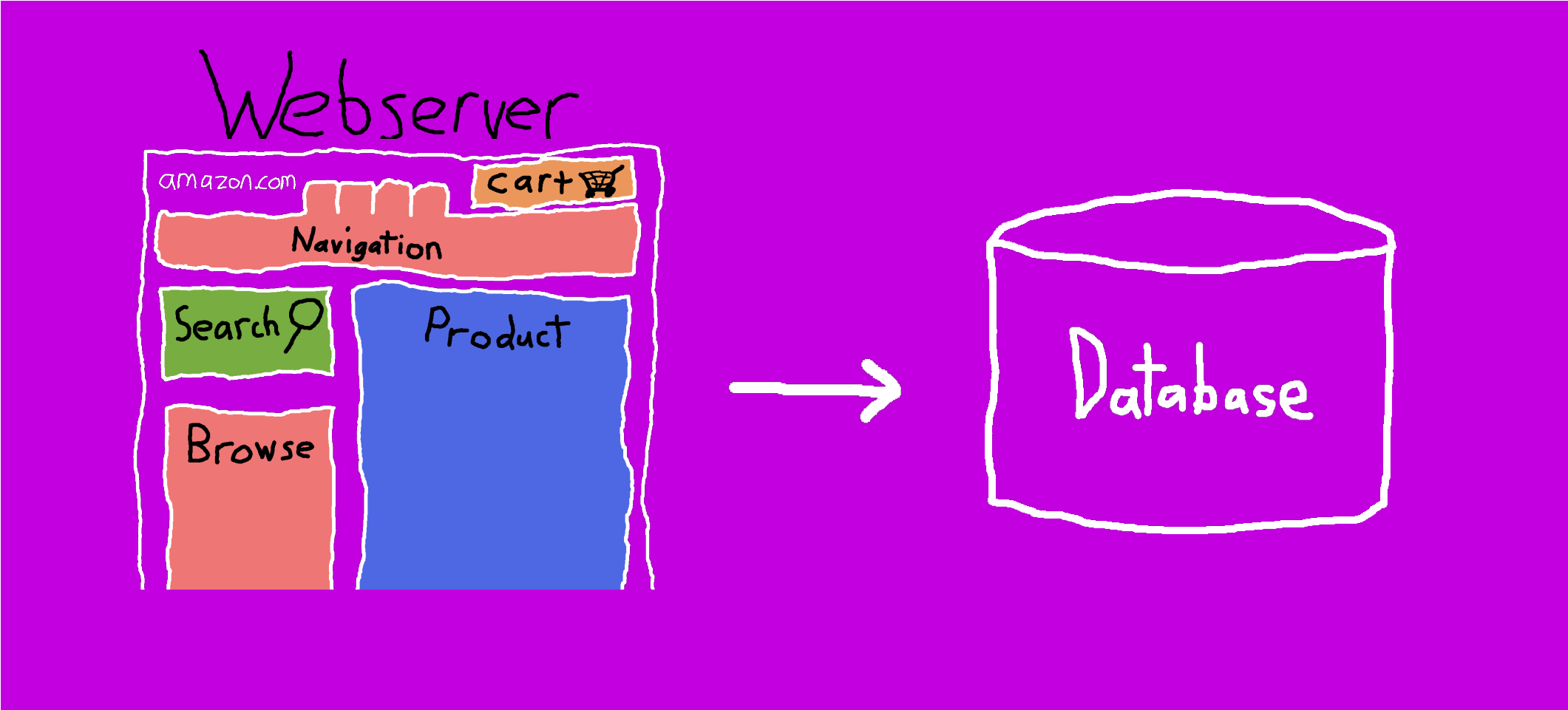
Story: Deployment automation at Amazon



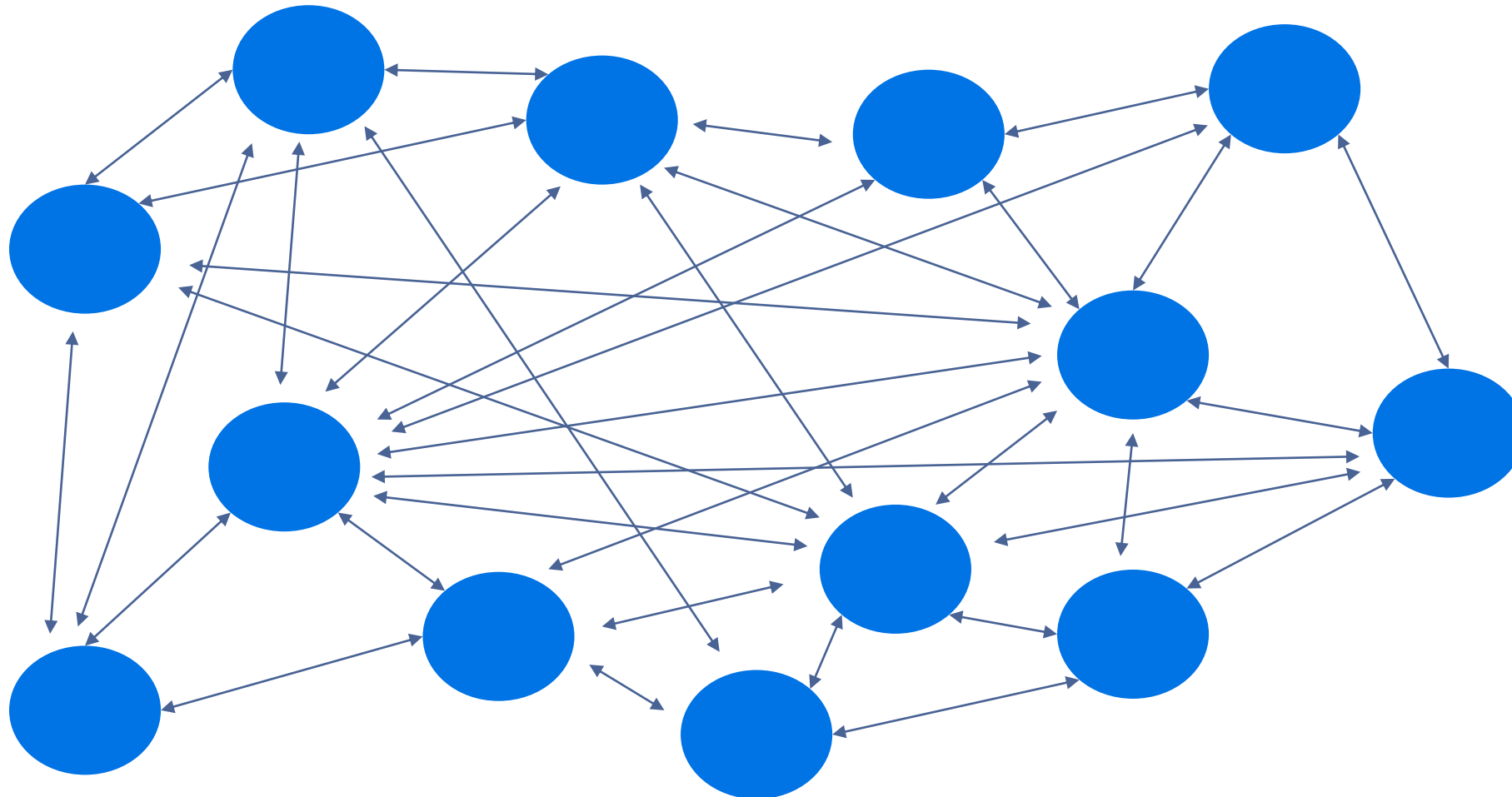
Transformation at Amazon: Early 2000s



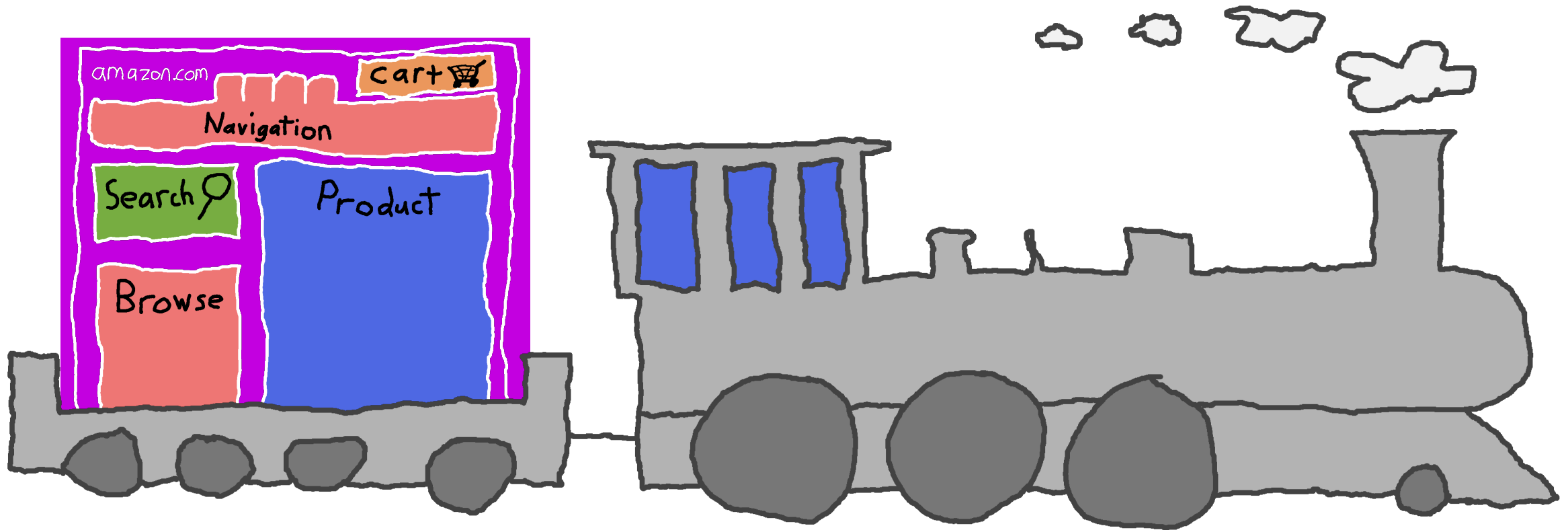
2-tier architecture



The cost of coordination



Release train: build, test, deploy, monitor

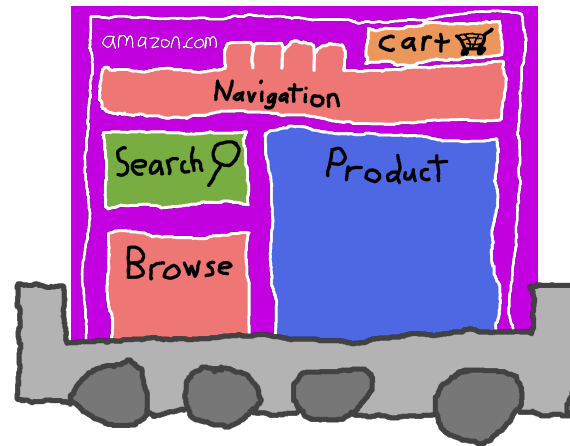


Lots of coordination

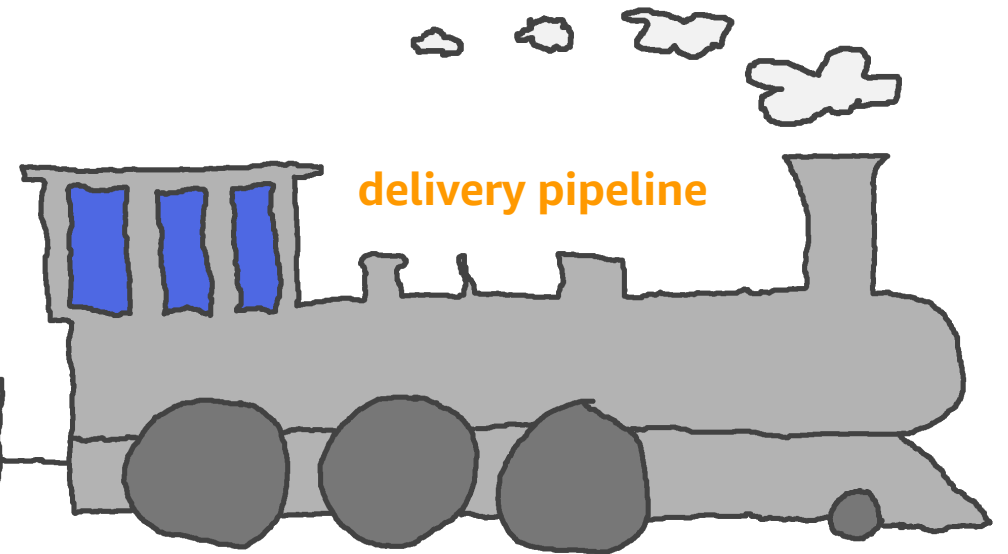
developers



monolith



delivery pipeline

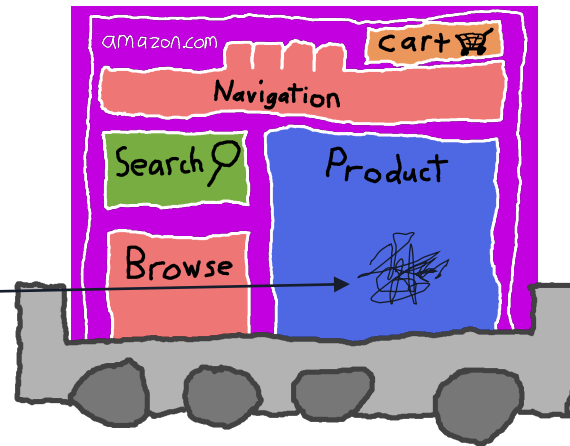


Lots that can go wrong

developers

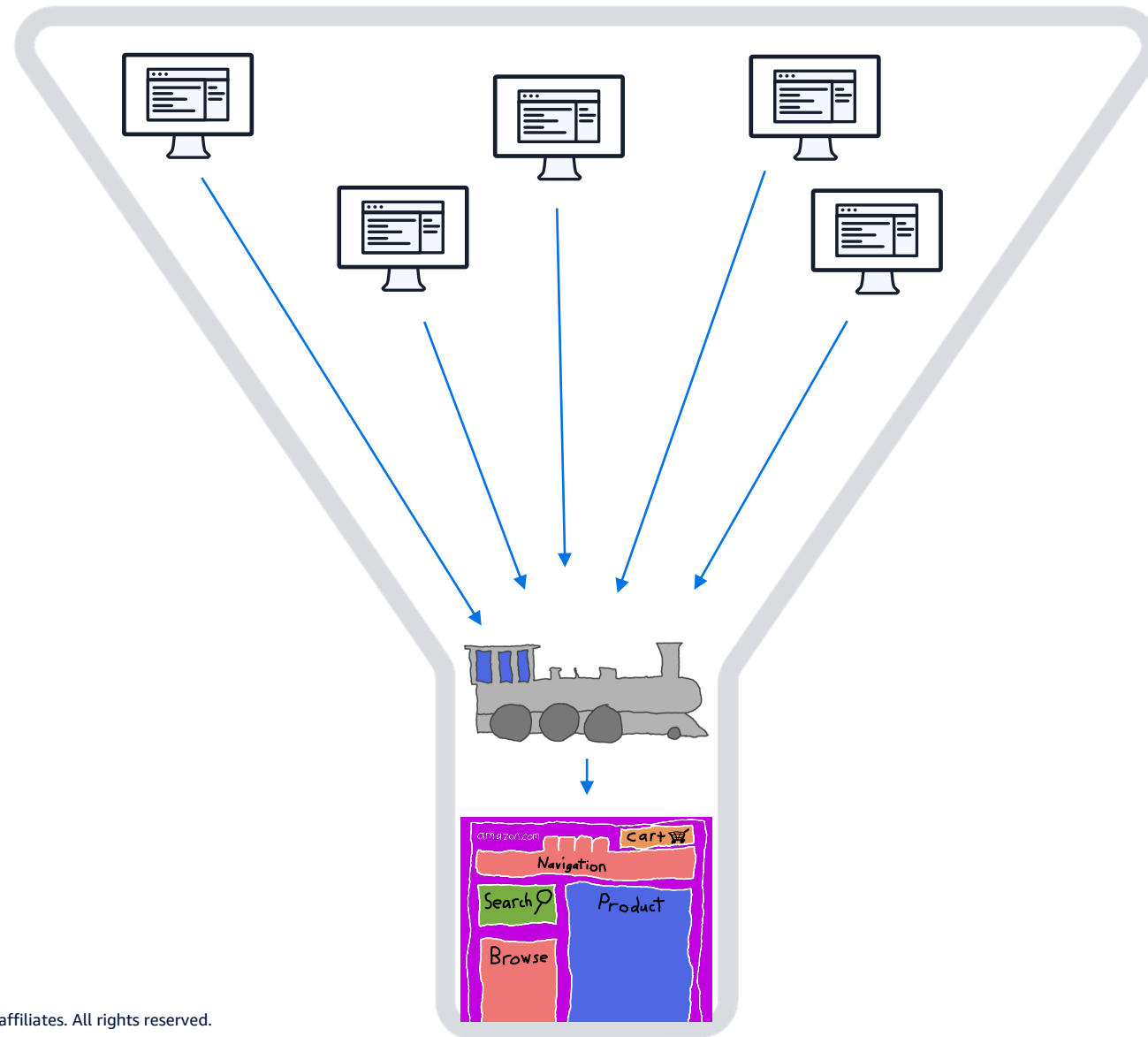


monolith

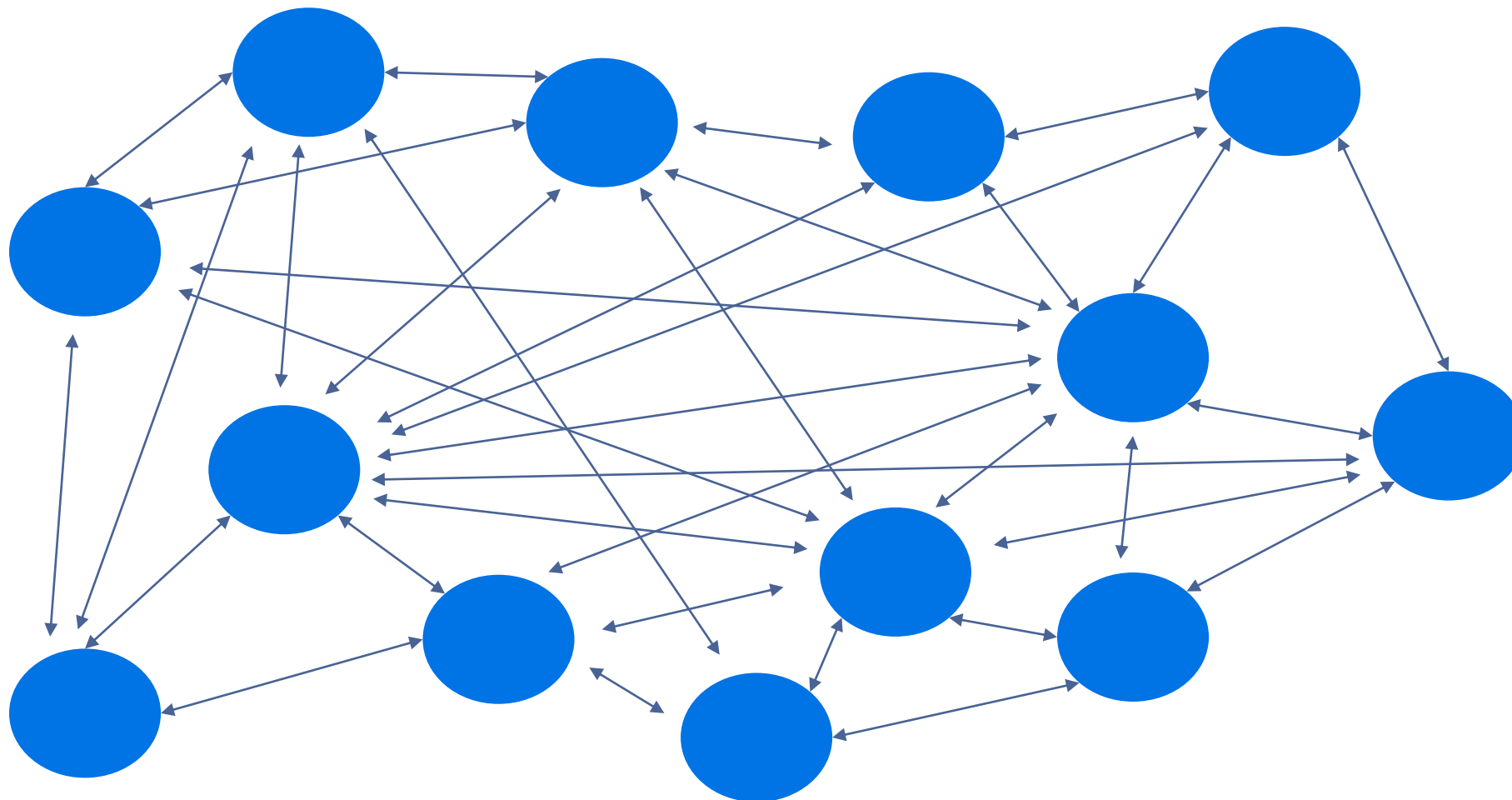


delivery pipeline

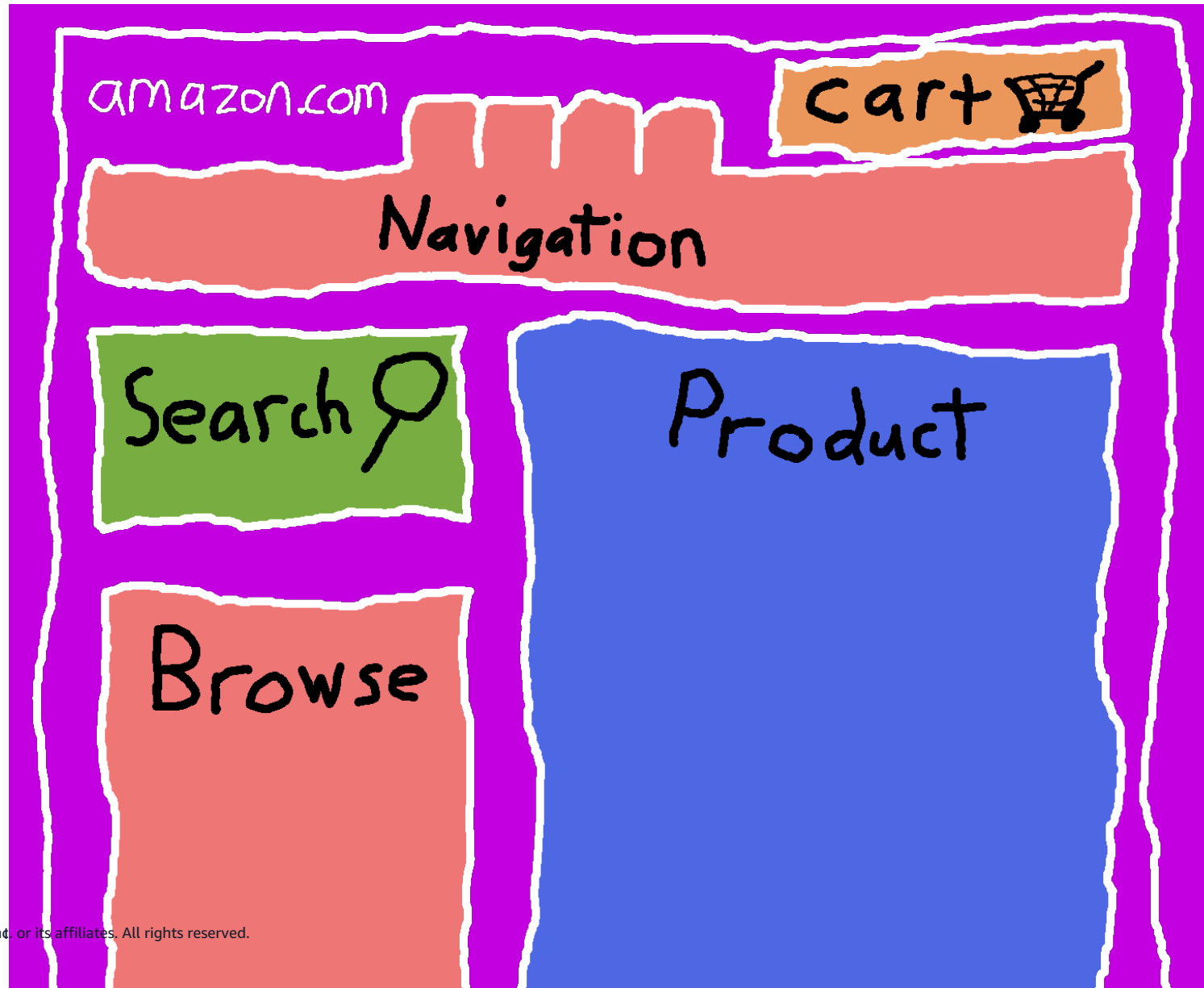
Bottlenecked processes



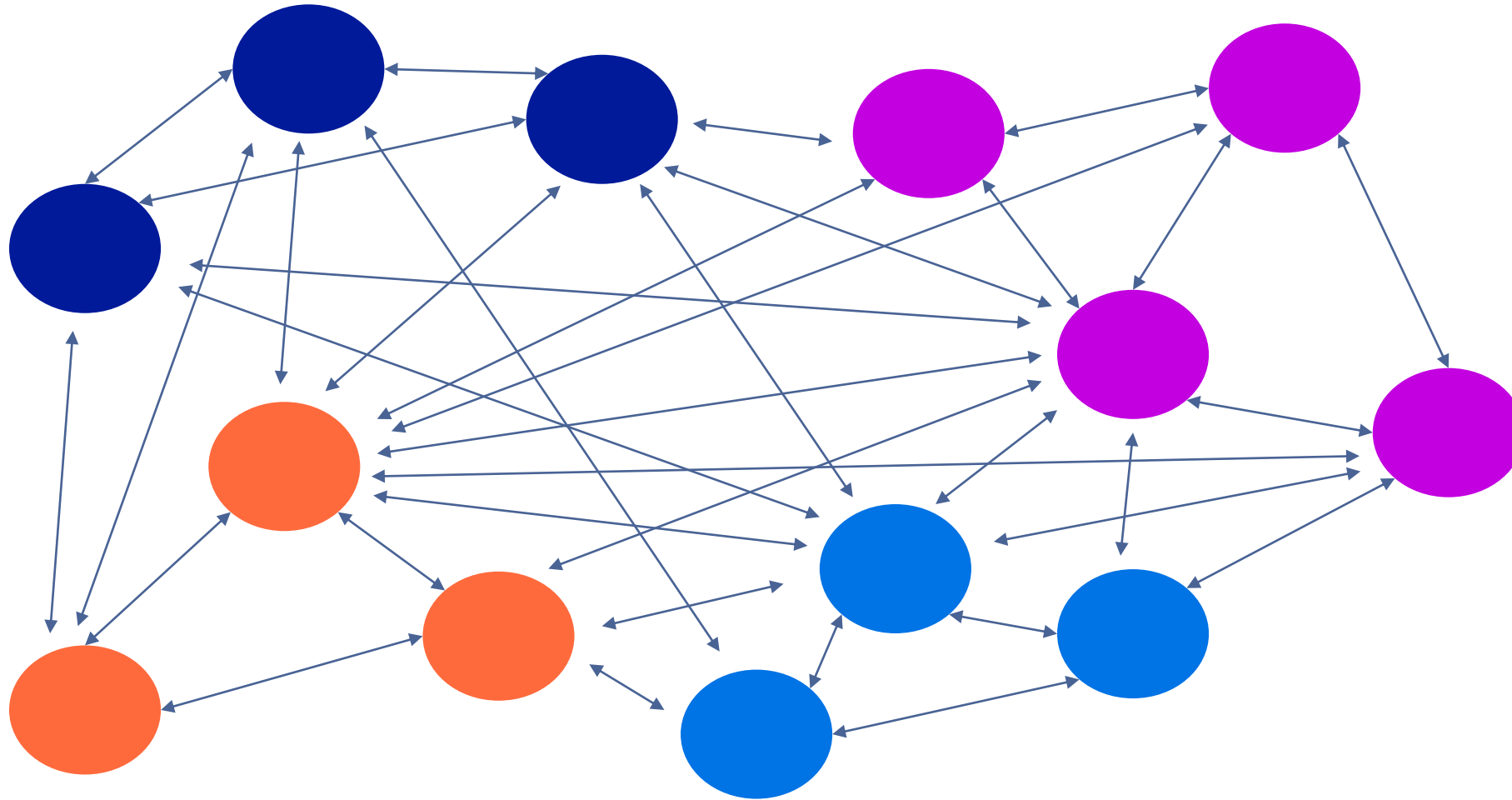
The cost of coordination



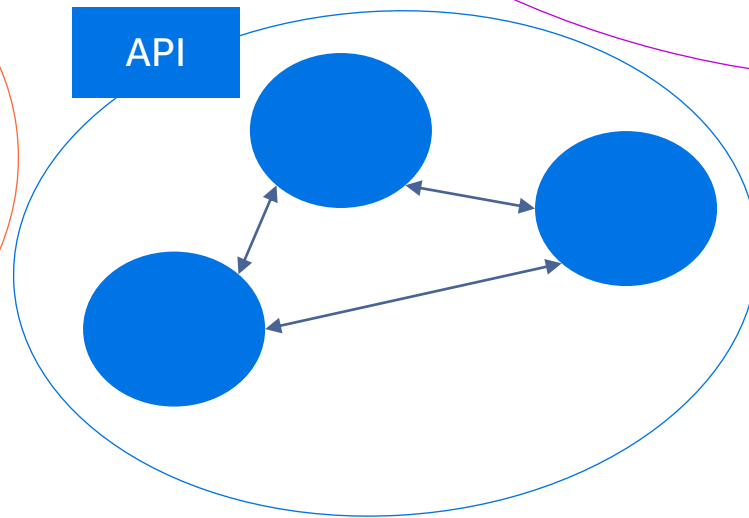
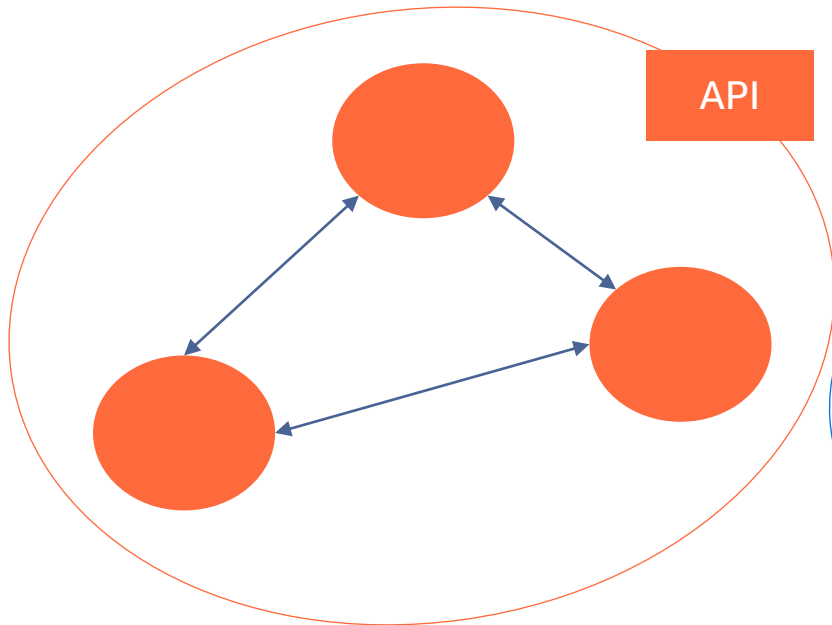
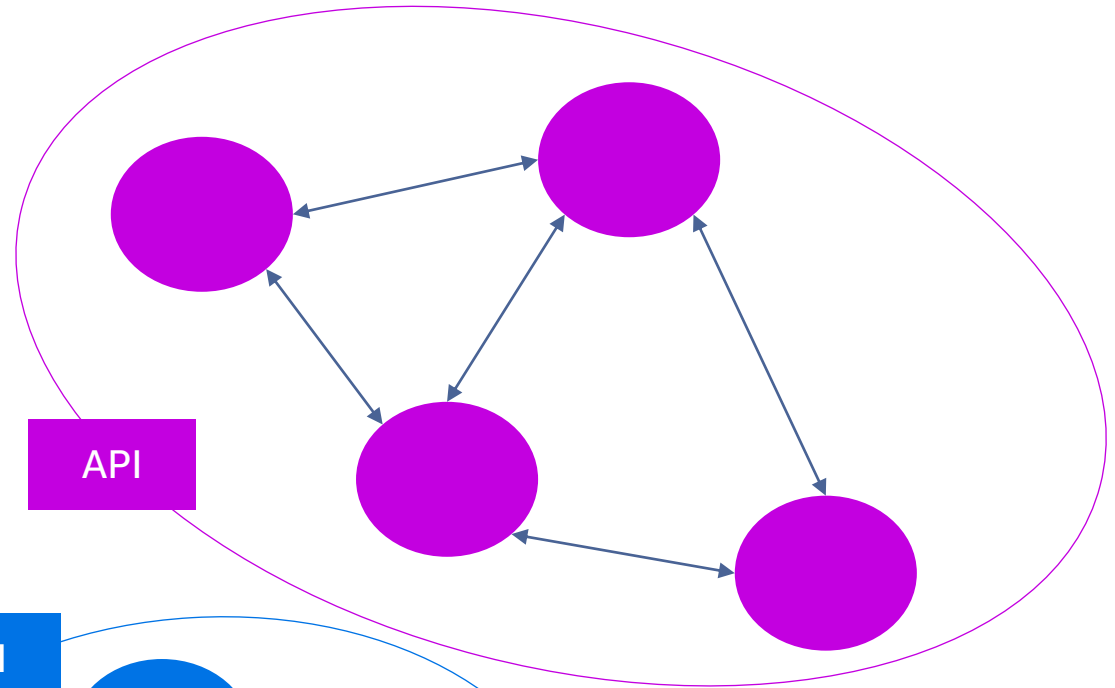
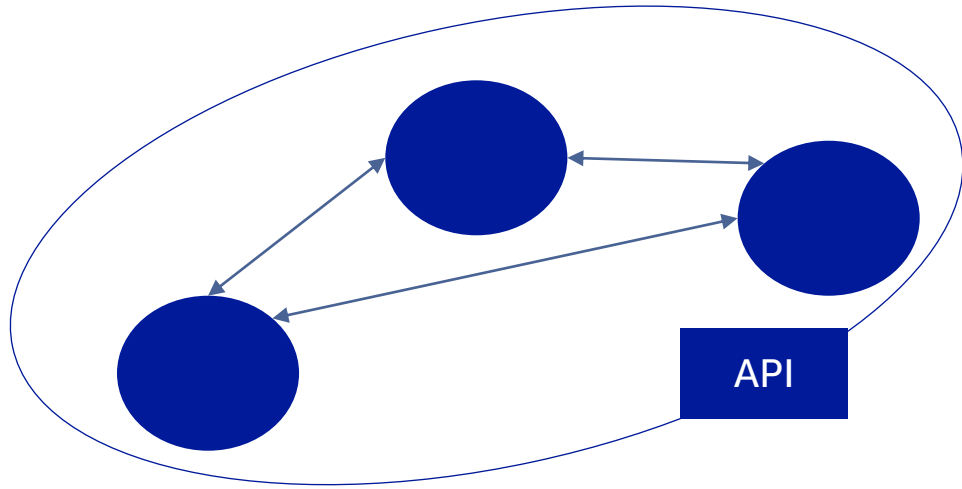
The teams behind the monolith



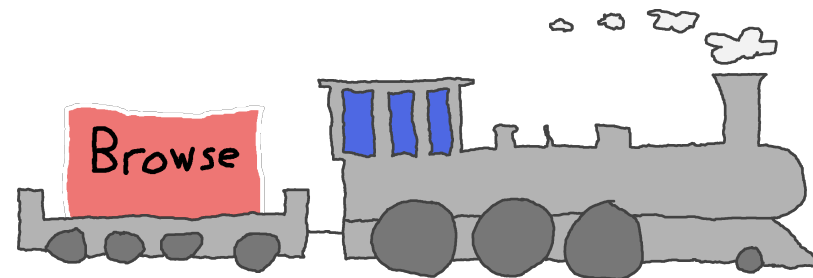
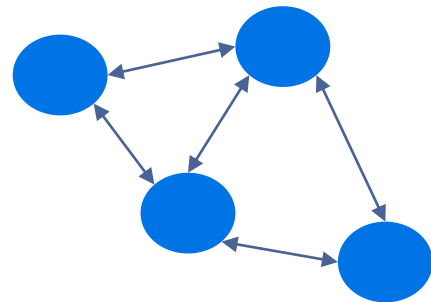
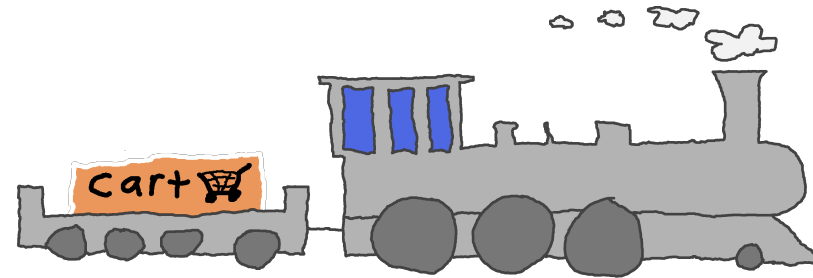
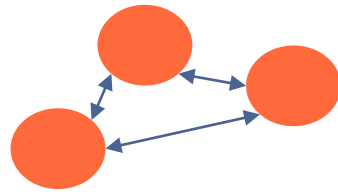
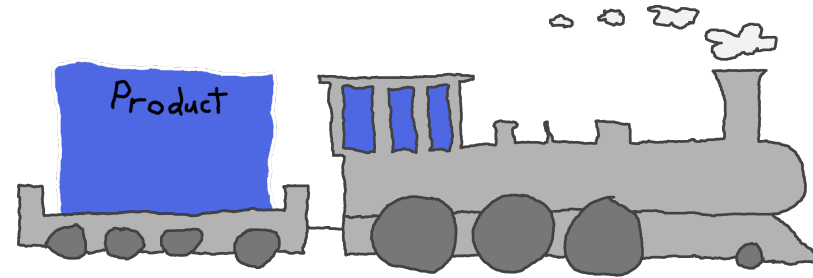
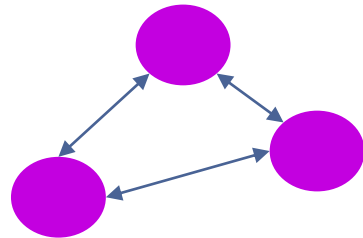
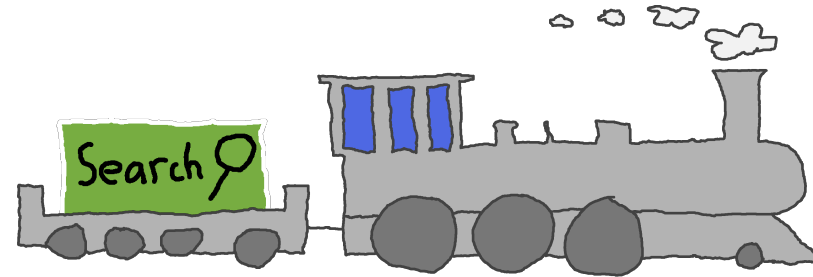
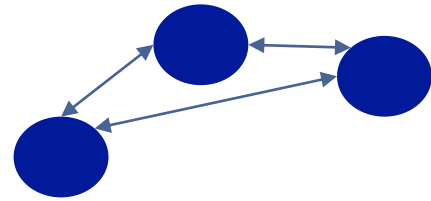
The cost of consensus building



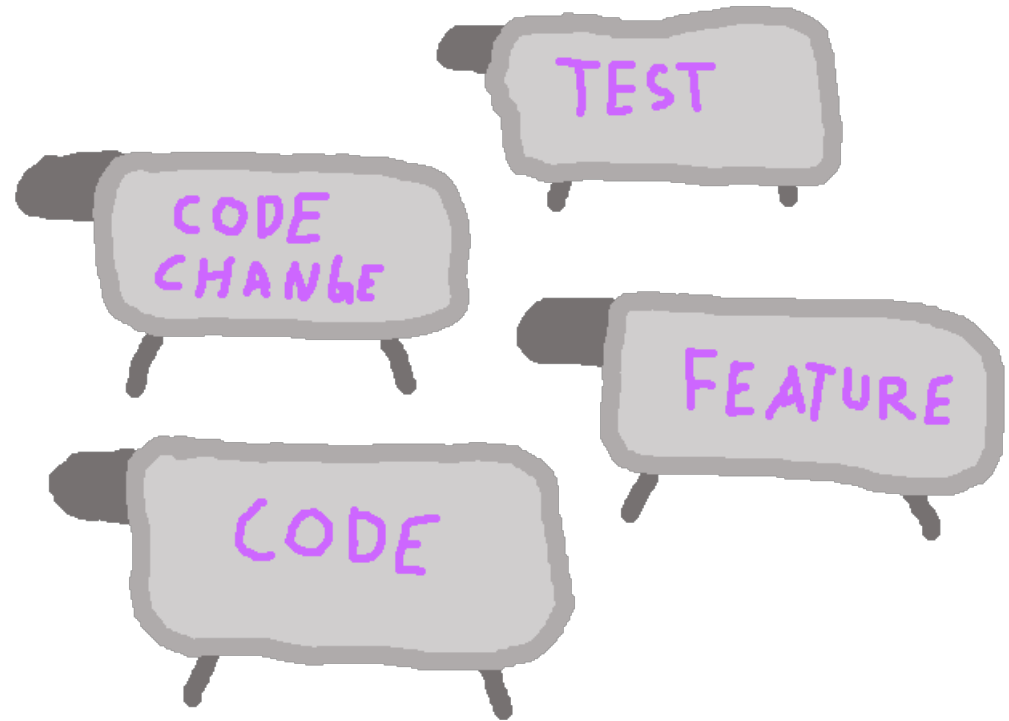
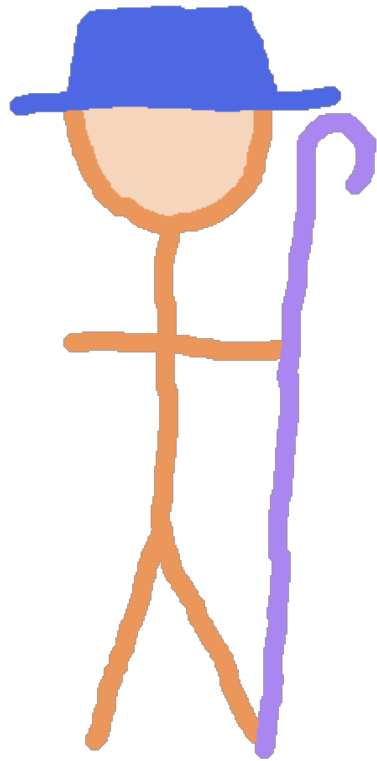
Increased autonomy



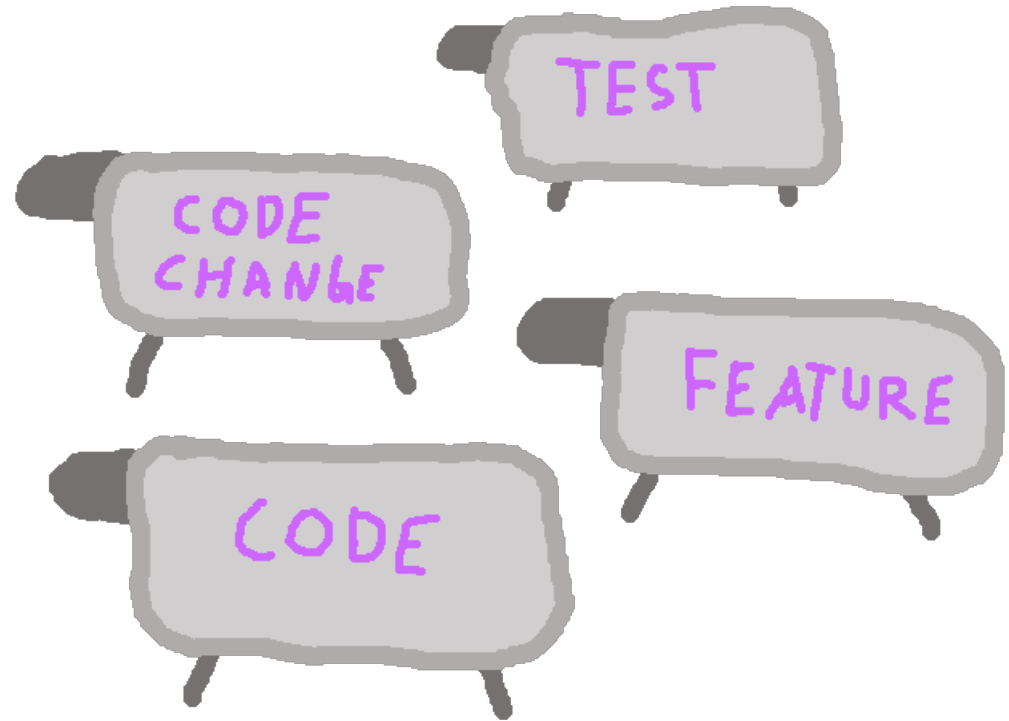
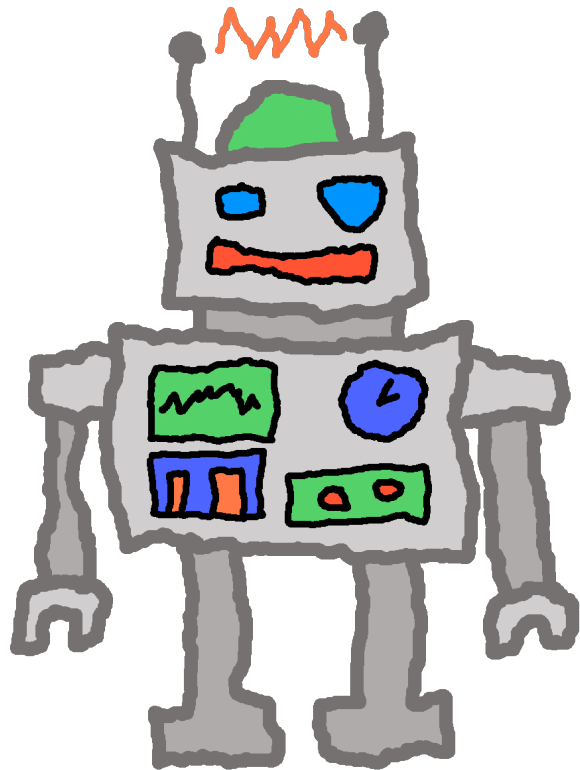
Reduced coordination



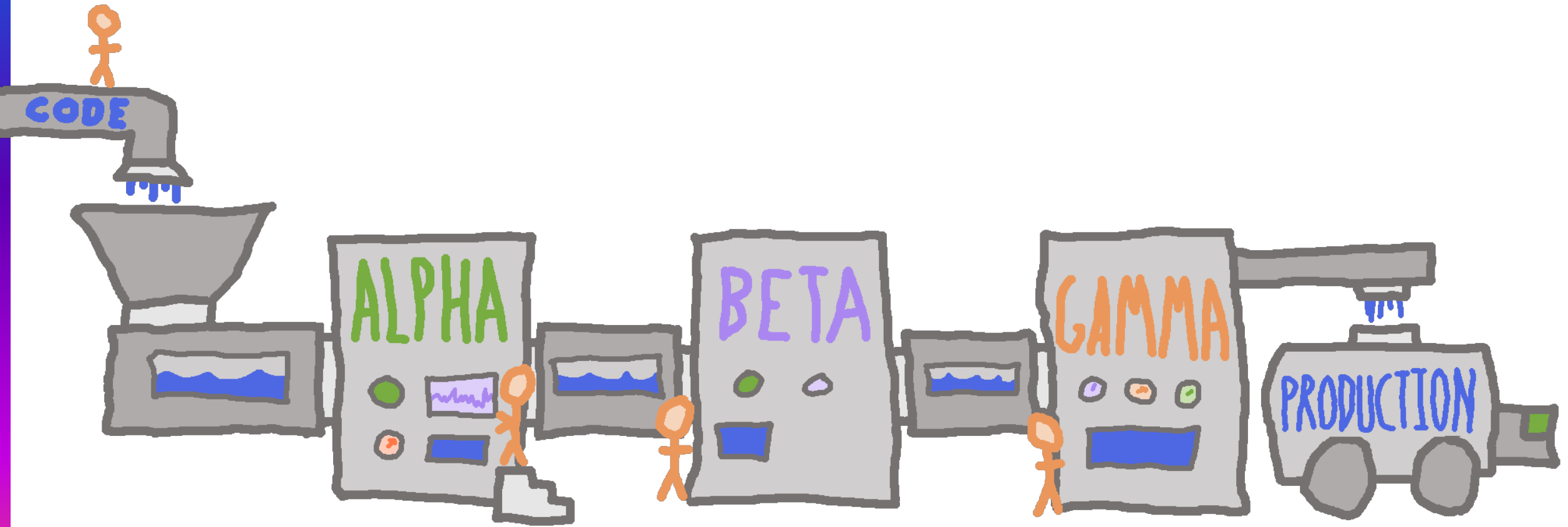
With independence comes responsibility



Deployment automation



Automated deployment pipelines



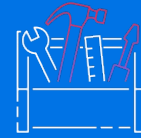
Principles of modern cloud system architecture

Systems

Build flexibly for the future with loosely coupled services and component-based architecture



Require auto-scaling and load balancing



Use purpose-built services



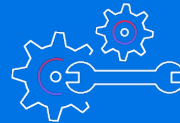
Govern architecture across the enterprise



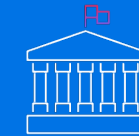
Ensure performance and skill alignment



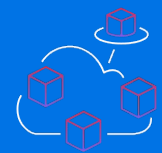
Offer seamless storage functionality



Decouple infrastructure & experience



Work backwards from business needs



Leverage automation and containers

Principles of modern cloud system architecture

Experience

Deliver holistically with user-centered design, accessibility, and reusable components



Require scalable public and worker interfaces



Segment and personalize



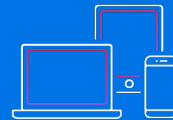
Develop a unified design approach



Ensure performance and skill alignment



Use seamless data to make decisions



Decouple data and channels

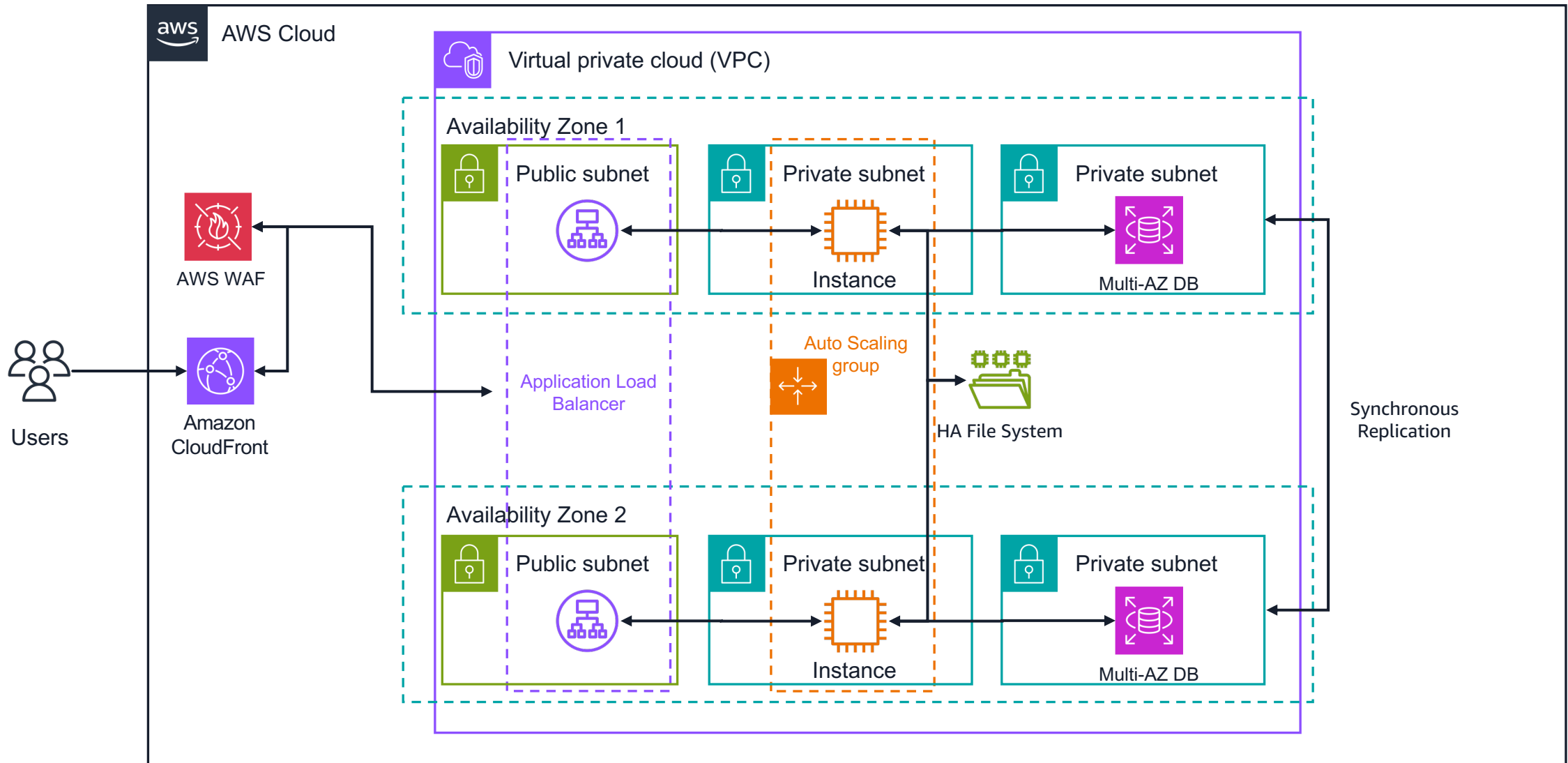


Work backwards from user needs

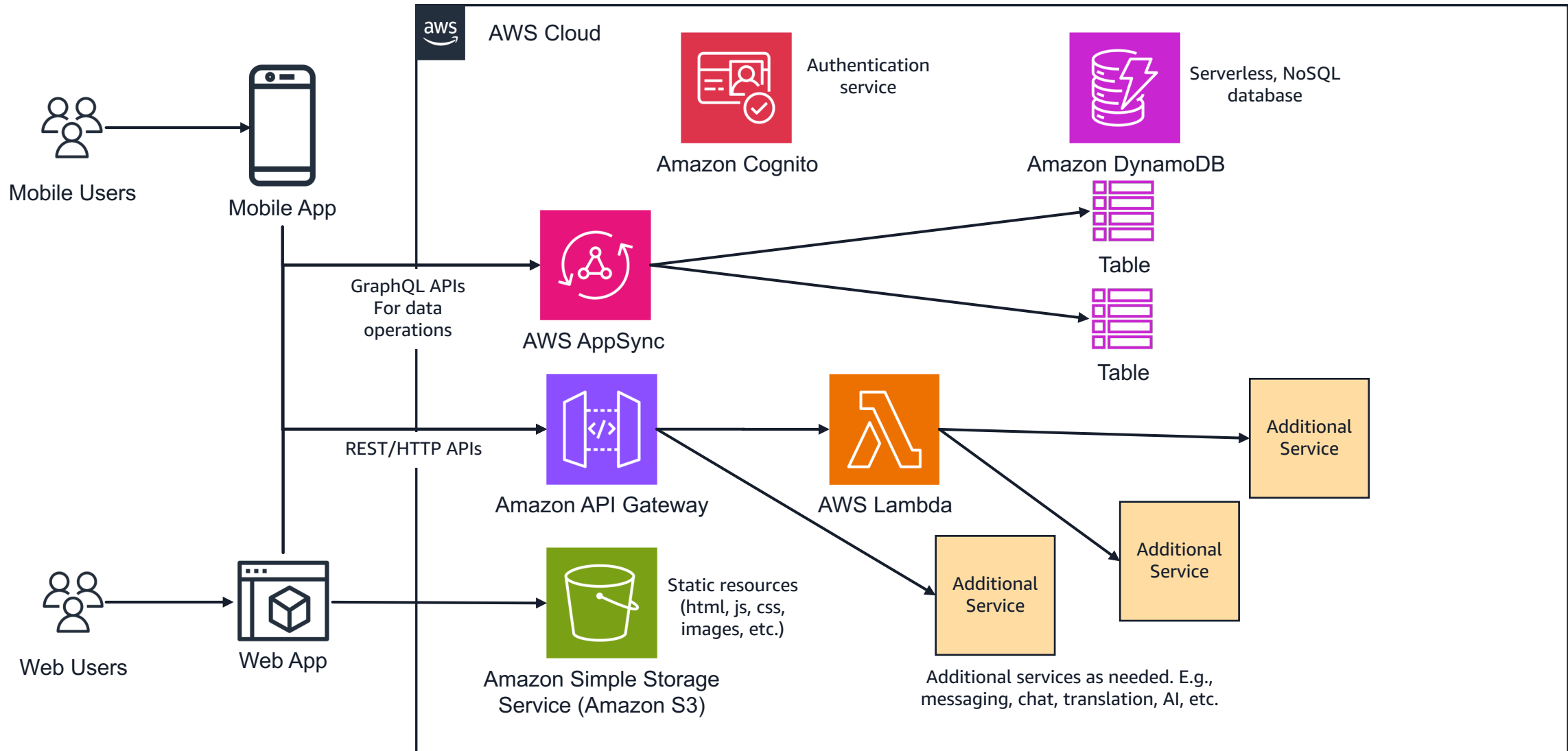


Leverage reusable components

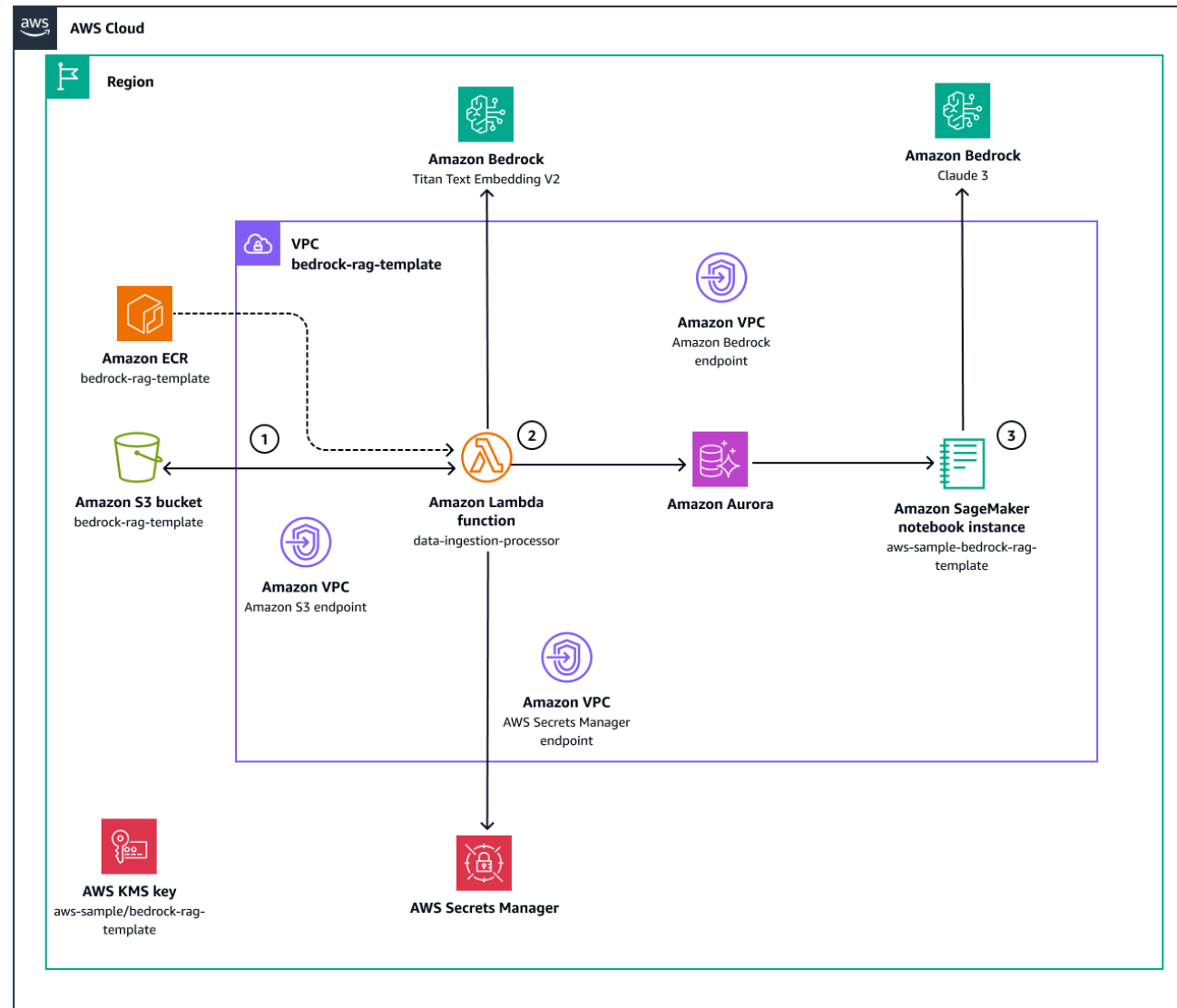
Highly available 3-Tier application



Serverless web/mobile application with APIs



Generative AI RAG Architecture



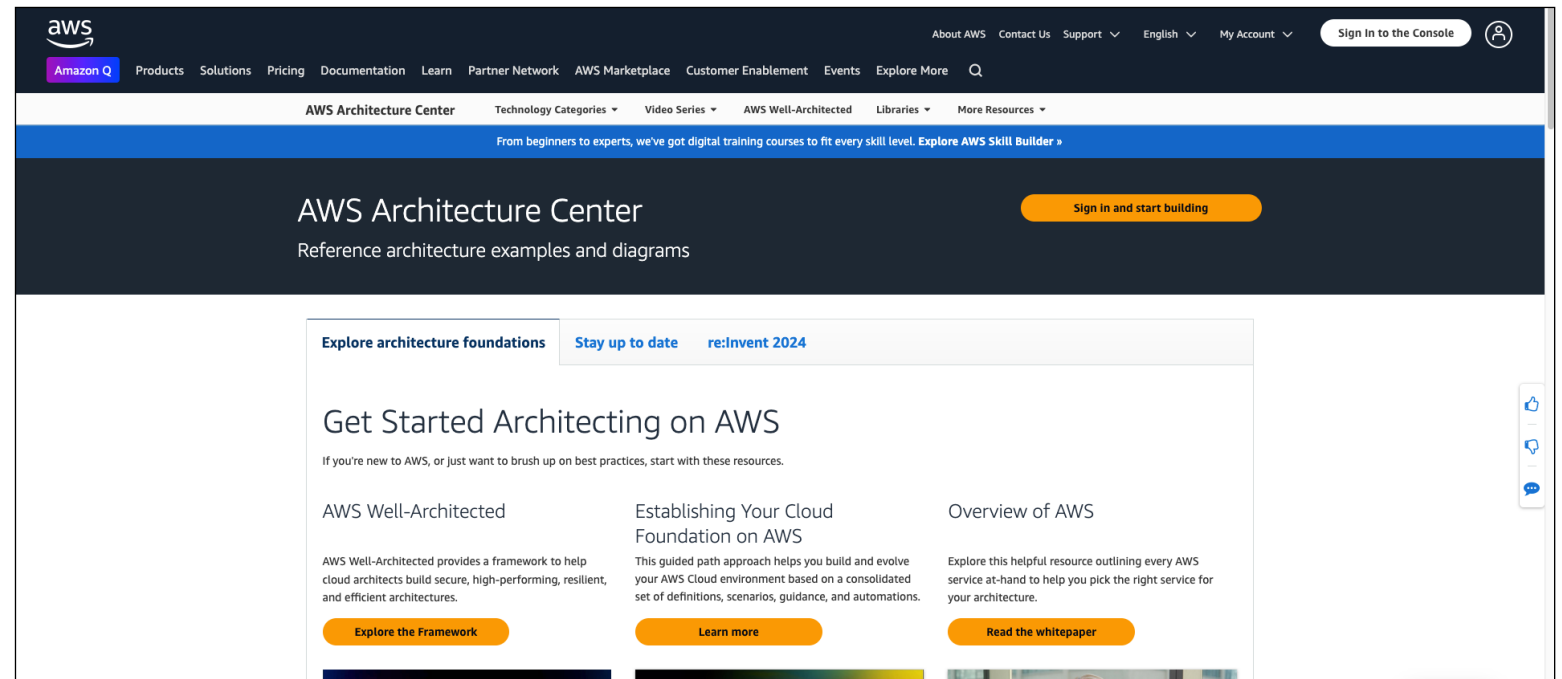
Source: <https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/deploy-rag-use-case-on-aws.html>

AWS Architecture Center

- Library of content including
 - Patterns
 - Reference Architectures
 - Guidance
 - Solutions, and more
- Links and other resources for architecting on AWS
- Video Series like 'This is My Architecture' and 'How to Build This'
- Architecture Best Practices



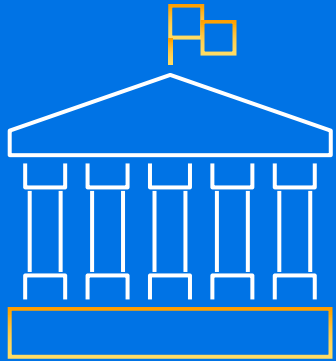
<https://aws.amazon.com/architecture>



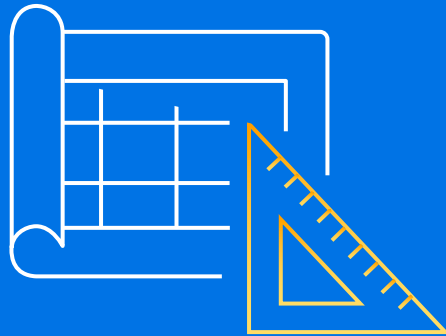
The screenshot shows the AWS Architecture Center homepage. At the top, there's a navigation bar with the AWS logo, 'Amazon Q', and various menu items like 'Products', 'Solutions', 'Pricing', 'Documentation', 'Learn', 'Partner Network', 'AWS Marketplace', 'Customer Enablement', 'Events', and 'Explore More'. A search icon is also present. Below the navigation bar, there's a secondary navigation bar with 'AWS Architecture Center' and several dropdown menus. A blue banner below that reads 'From beginners to experts, we've got digital training courses to fit every skill level. Explore AWS Skill Builder >'. The main heading is 'AWS Architecture Center' with a 'Sign in and start building' button. Below the heading, it says 'Reference architecture examples and diagrams'. The main content area features a section titled 'Get Started Architecting on AWS' with a sub-heading 'Explore architecture foundations' and 'Stay up to date re:Invent 2024'. Below this, there are three cards: 'AWS Well-Architected' with an 'Explore the Framework' button, 'Establishing Your Cloud Foundation on AWS' with a 'Learn more' button, and 'Overview of AWS' with a 'Read the whitepaper' button. A vertical sidebar on the right contains social media icons for Facebook, LinkedIn, and Twitter.



What is the AWS Well-Architected Framework?



Pillars & Lenses



Design principles



Questions



Best Practices

Pillars of the AWS Well-Architected Framework



How does Amazon do operations?



Lessons learned at Amazon for developing and delivering software



Decompose for agility
(microservices, two-pizza teams)



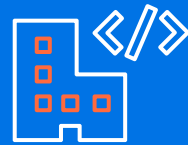
Automate everything



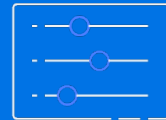
Standardized tools



Belts and suspenders
(governance, templates)



Infrastructure as code (IaC)



Continuous configuration
(feature flags, operational flags)

Who owns operations at Amazon?



Service teams

- Fully own operations
- Innovate and invent



Platform engineering

- Raise standards through tools
- Reduce effort of teams
- Empower and unblock teams



Management

- Reinforce culture
- Set standards
- Hold accountable
- Allocate resources

Help teams learn from each other

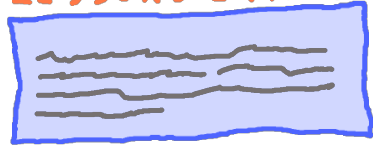


Help teams learn from each other



Ops wins

LESSONS LEARNED



Incident retrospectives



Operational readiness reviews



Weekly ops meetings

Give teams tools to be successful



Give teams tools to be successful

- Deploy software easily and safely
- Manage dependencies and keep software and systems patched
- Respond to incidents and coordinate investigations
- Monitor, alarm, observe, and troubleshoot systems
- Build web services and websites that are secure and reliable
- Operate services efficiently
- Leverage AWS services with guardrails but without red tape

Software quality measurement: prevention

- Raise standards by verifying code commits and deployments
- Formal verification of algorithms
- Scanning codebases for upgradeable dependencies

Software quality measurement: reporting

Software (pipeline health)

- Inventory age, commit-to-deploy time, rollback rate, flaky tests, etc

Builders (productivity and pain)

- Oncall burden (page count, frequency of oncall)
- New builder time-to-onboard

Inform teams proactively about improvements



Inform teams proactively about improvements

- Disable deployment pipelines that don't follow required standards
- Roll back deployments automatically from "golden signal" metric anomalies
- Scan infrastructure and software for known configuration improvements
- Inspect infrastructure utilization metrics and look for wasted resources
- Perform upgrades by pushing "pull requests" to teams
- Set an annually "north star" goal

The Amazon Builders' Library

Firsthand articles from engineers who built AWS services:

- Caching
- Retries
- Fairness
- Safety
- More...



<https://aws.amazon.com/builders-library/>

Thank you!

Leo Zhadanovsky

leozh@amazon.com

Vlad Fatu

vladfatu@amazon.com

**Please complete the survey
for this session**



Track: Cloud Fundamentals

Session: Designing modern applications in AWS



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws Learning Days
State, Local, and Education

Story: The ops win

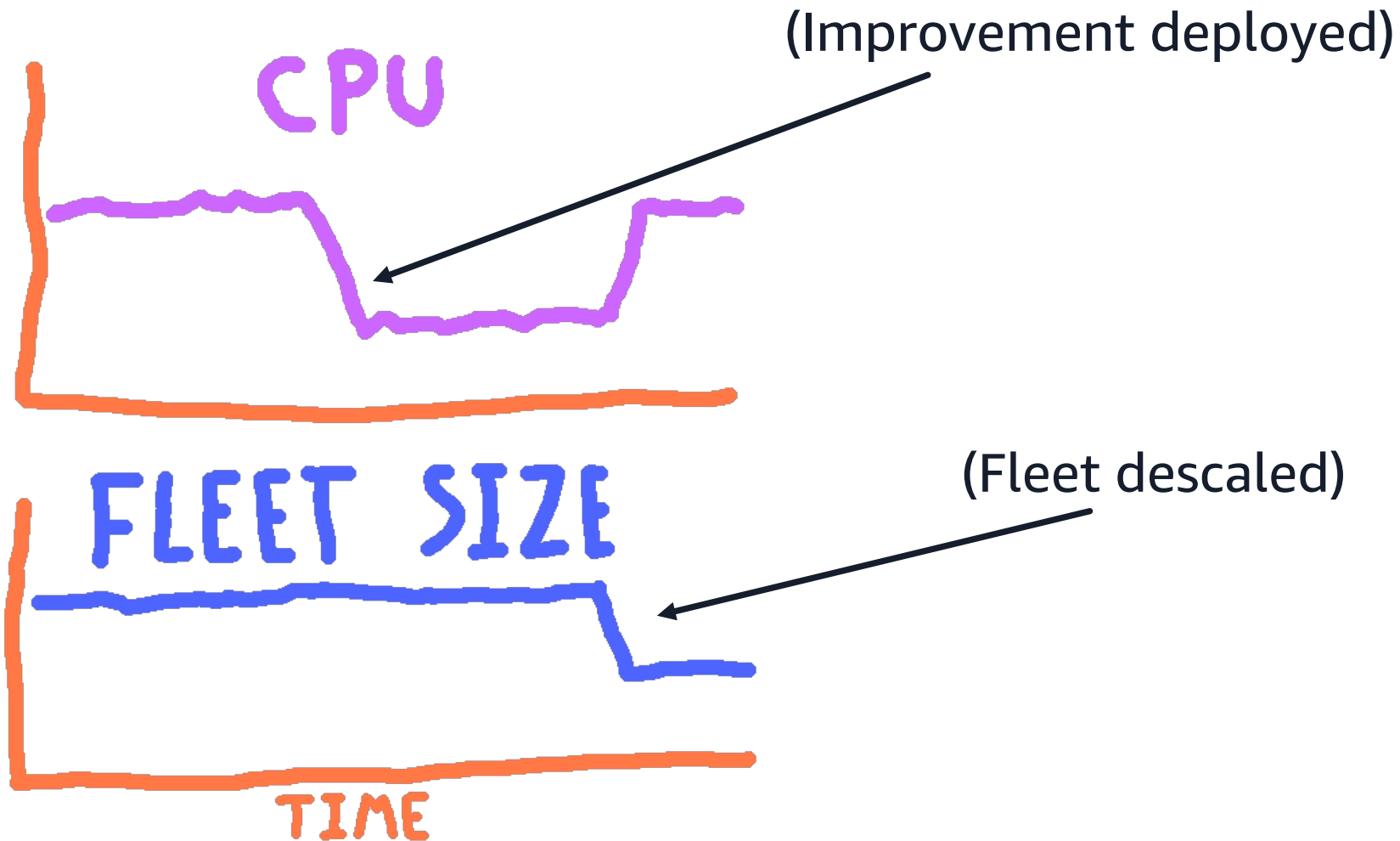




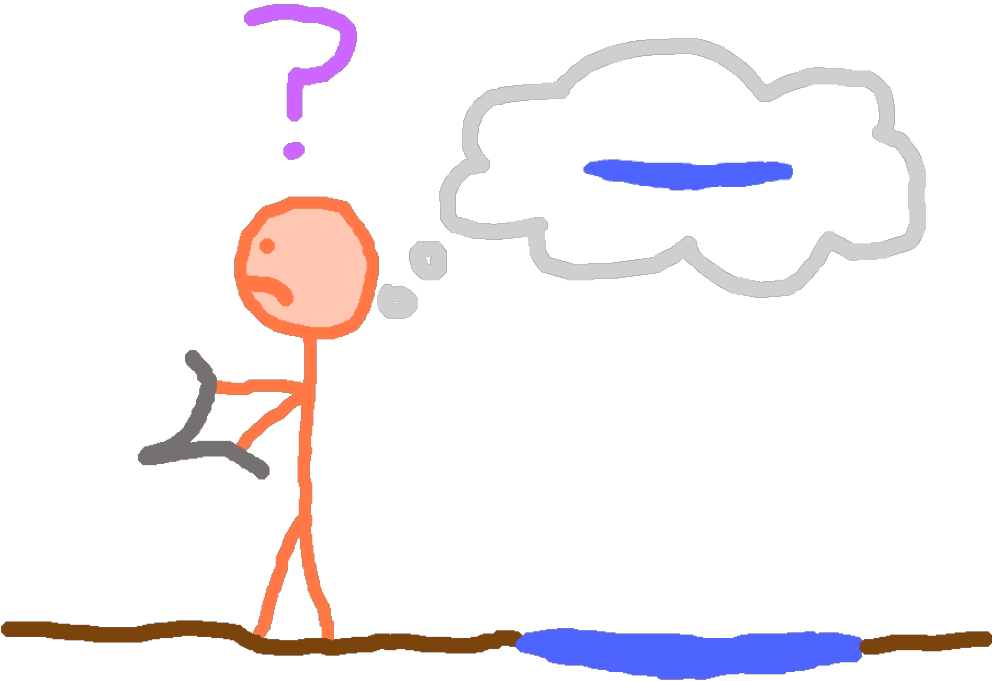




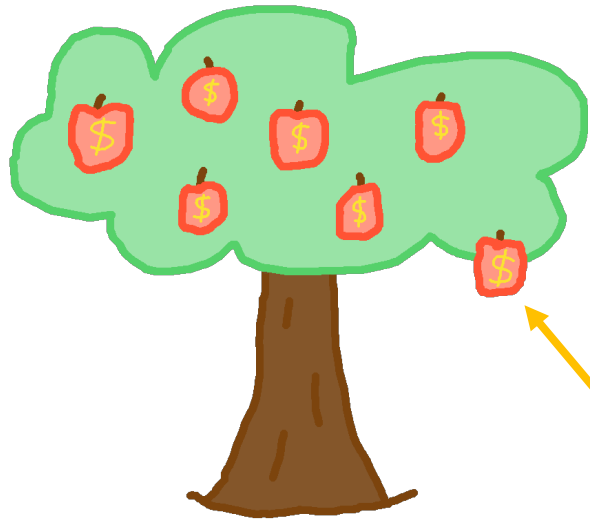
Deploying an efficiency improvement



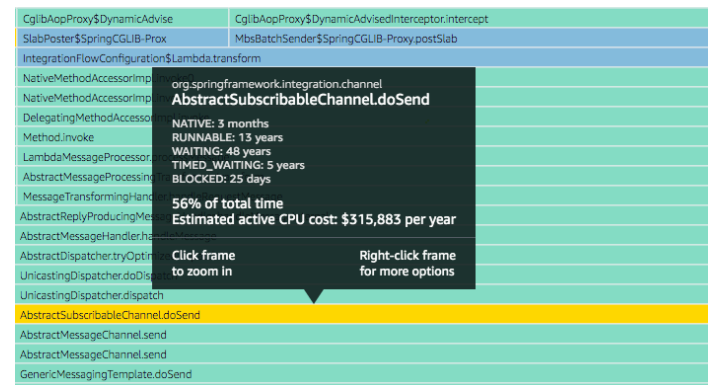
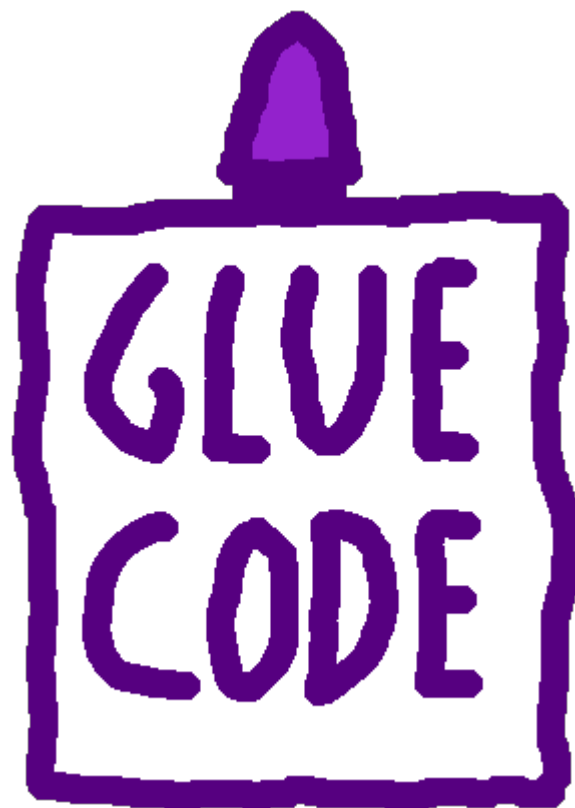
Find easy optimizations in code



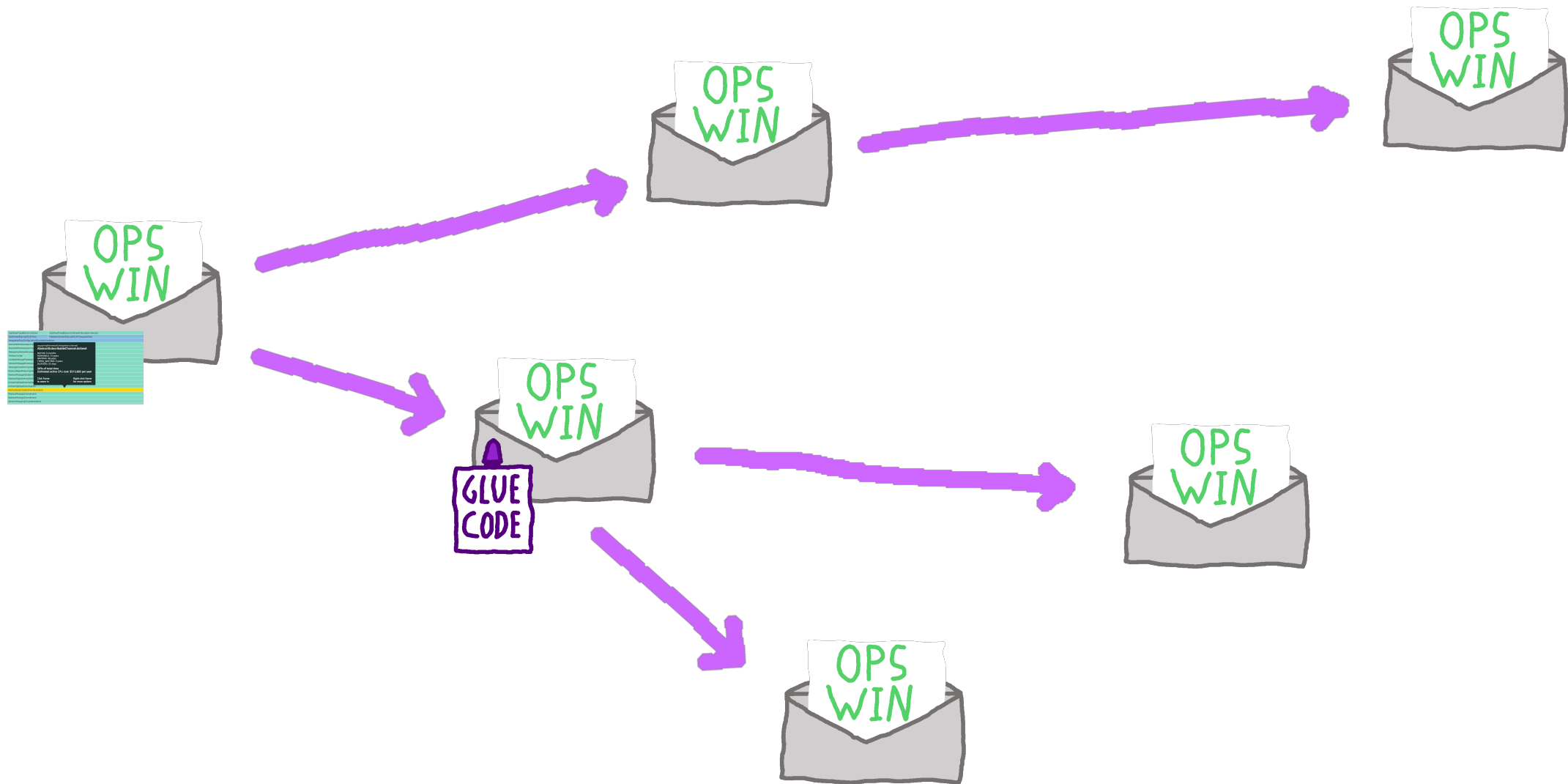
Using a profiler to find low-hanging fruit



CglibAopProxy\$DynamicAdvise	CglibAopProxy\$DynamicAdvisedInterceptor.intercept
SlabPoster\$SpringCGLIB-Proxy	MbsBatchSender\$SpringCGLIB-Proxy.postSlab
IntegrationFlowConfiguration\$Lambda.transform	
NativeMethodAccessorImpl.invoke	org.springframework.integration.channel
NativeMethodAccessorImpl.invoke	AbstractSubscribableChannel.doSend
DelegatingMethodAccessorImpl.invoke	NATIVE: 3 months
Method.invoke	RUNNABLE: 13 years
LambdaMessageProcessor.process	WAITING: 48 years
AbstractMessageProcessingTemplate.doSend	TIMED_WAITING: 5 years
MessageTransformingHandler.handleMessage	BLOCKED: 25 days
AbstractReplyProducingMessageHandler.handleMessage	56% of total time
AbstractMessageHandler.handleMessage	Estimated active CPU cost: \$315,883 per year
AbstractDispatcher.tryOptimize	Click frame to zoom in
UnicastingDispatcher.doDispatch	Right-click frame for more options
UnicastingDispatcher.dispatch	
AbstractSubscribableChannel.doSend	
AbstractMessageChannel.send	
AbstractMessageChannel.send	
GenericMessagingTemplate.doSend	



The ripple effect





[Contact Us](#) [Support](#) [English](#) [My Account](#) [Sign In](#)

[Create an AWS Account](#)

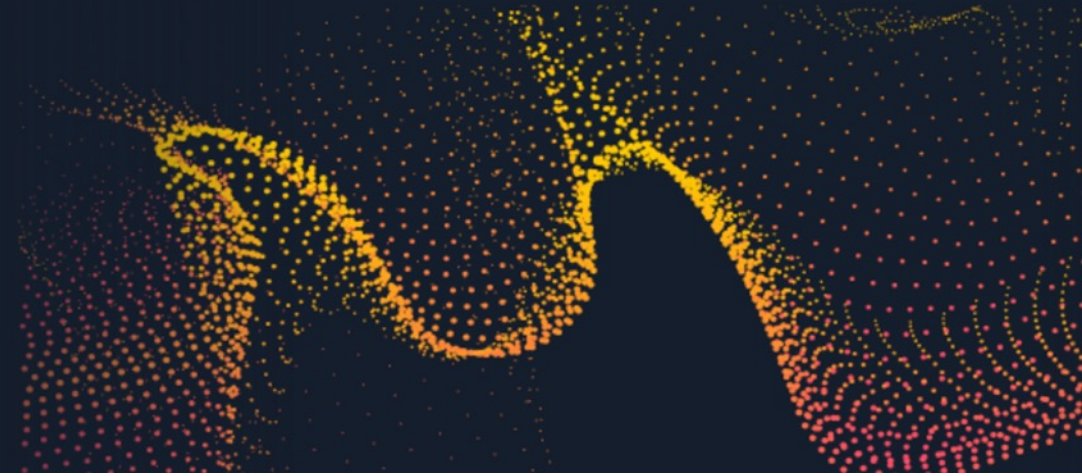
[Products](#) [Solutions](#) [Pricing](#) [Documentation](#) [Learn](#) [Partner Network](#) [AWS Marketplace](#) [Customer Enablement](#) [Events](#) [Explore More](#) [Q](#)

[Amazon CodeGuru](#) [Overview](#) [Features](#) [Pricing](#) [FAQs](#) [Customers](#) [Resources](#)

Amazon CodeGuru

Automate code reviews and optimize application performance with ML-powered recommendations

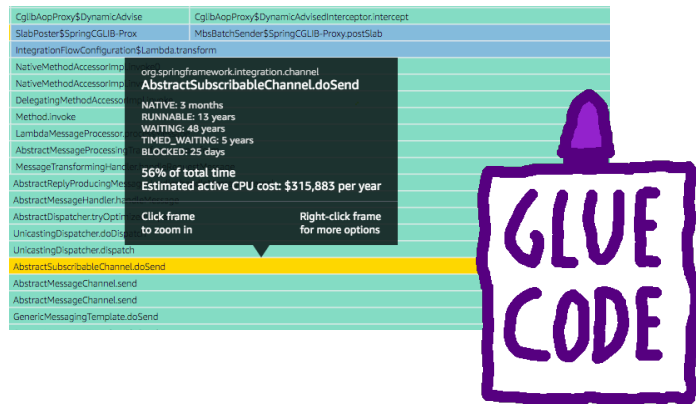
[Get started with Amazon CodeGuru](#)



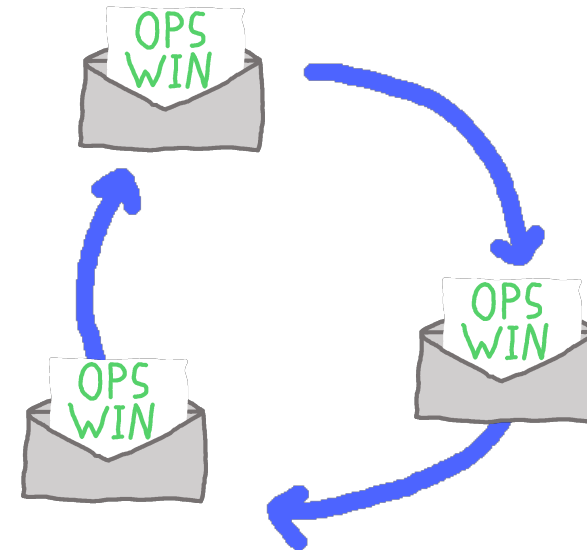
The ripple effect



Takeaways: the ops win



Share best practices



Reinforce ops culture

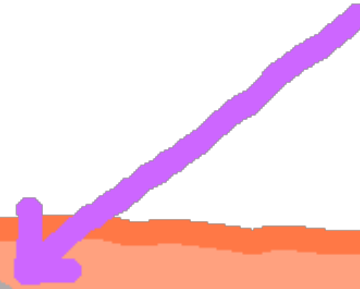
Story: The retrospective



Early one morning . . .



EMPTY



Wrong coffee, wrong pot

DECAF
COFFEE



DECAF
POT



COE: Correction of error

[89458] Decaf coffee brewed in a non-decaf pot

Summary

At 8:45 am Pacific on 4/24/2018 on the 12th floor of Alexandria, an operator inadvertently brewed decaf coffee into the "medium roast, non-decaf" coffee pot. The operator was attempting to brew decaf coffee, but chose the wrong pot to brew it into.

Metrics / Graphs

- Number of pots of coffee brewed incorrectly during incident: 1
- Number of customers who unknowingly took a cup of decaf coffee: 2 (estimated)

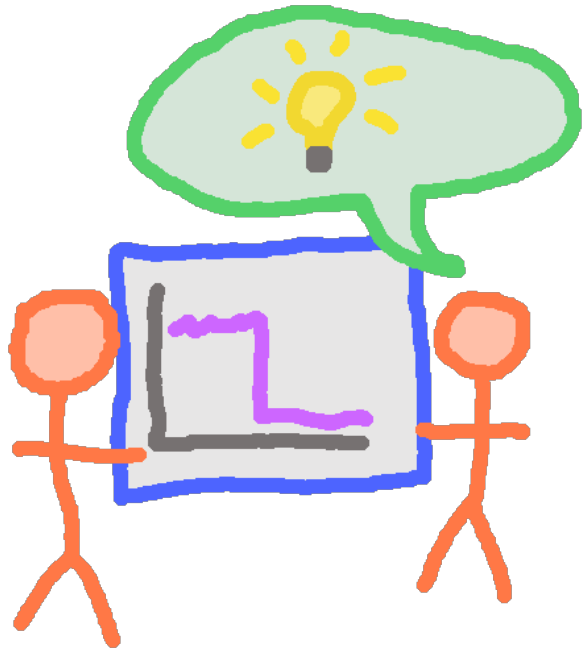
Customer Impact

While there are no metrics available to prove this, we know that there were at least two people who took coffee from the pot. One engineer had a half cup of coffee at his desk. That engineer did not move the pot from the brewing station into its holding spot, so that suggests that at least one other person had coffee. The one confirmed engineer chose to drink the cup of coffee anyway.

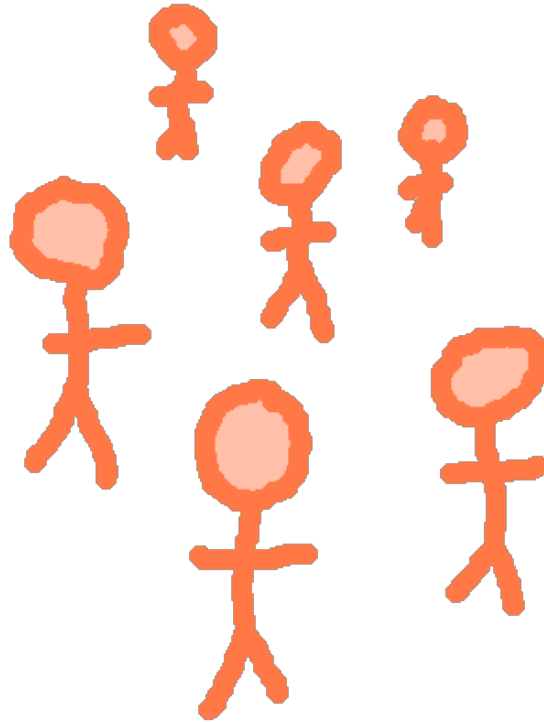


Amazon's approach to failing successfully
<https://www.youtube.com/watch?v=yQiRli2ZPxU>

Goals of retrospectives



Improve systems



Teach others



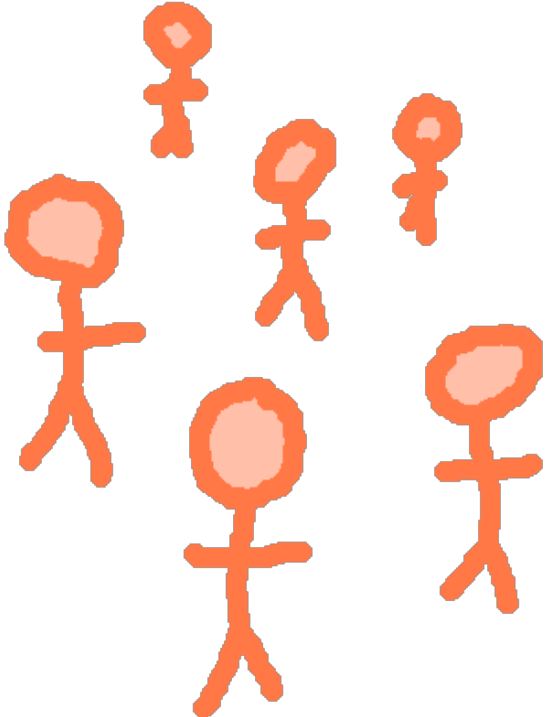
Improve tools

Summary



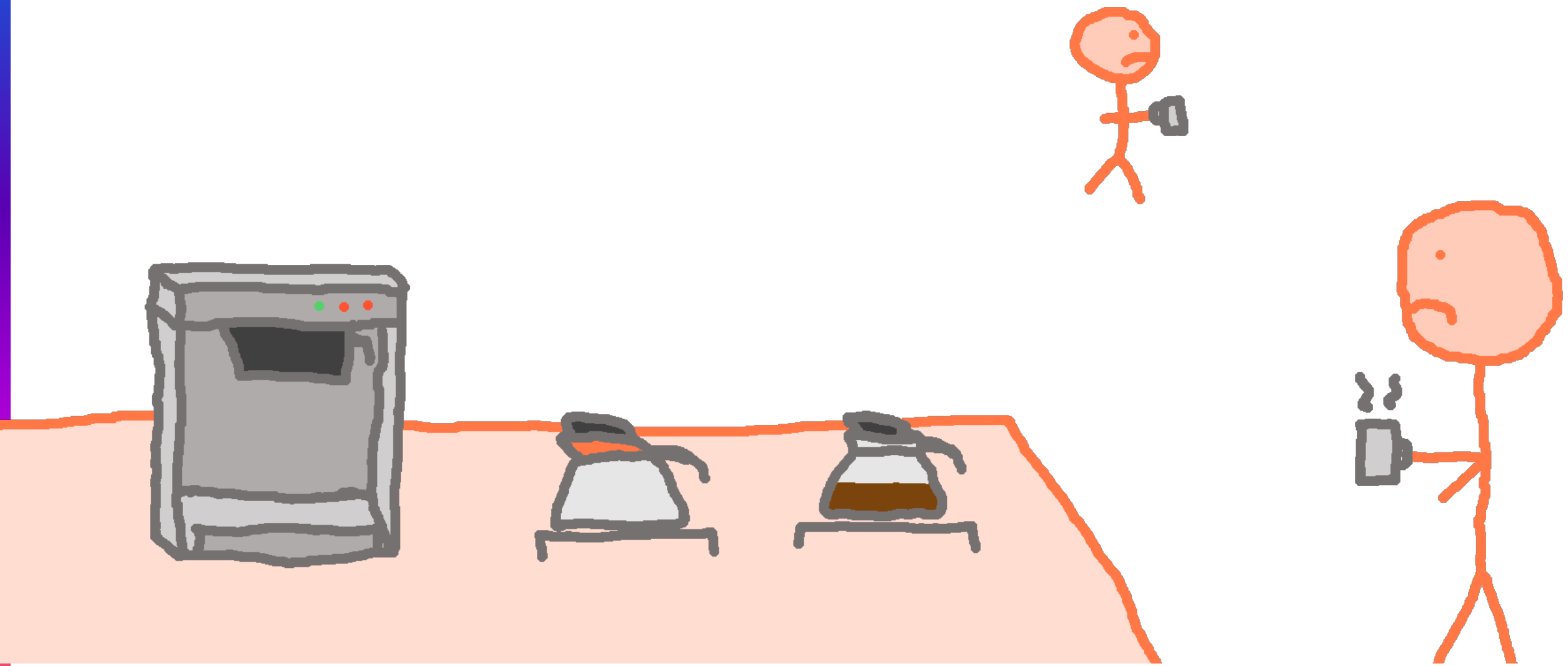
Non-punitive
(no names!)

3 paragraphs

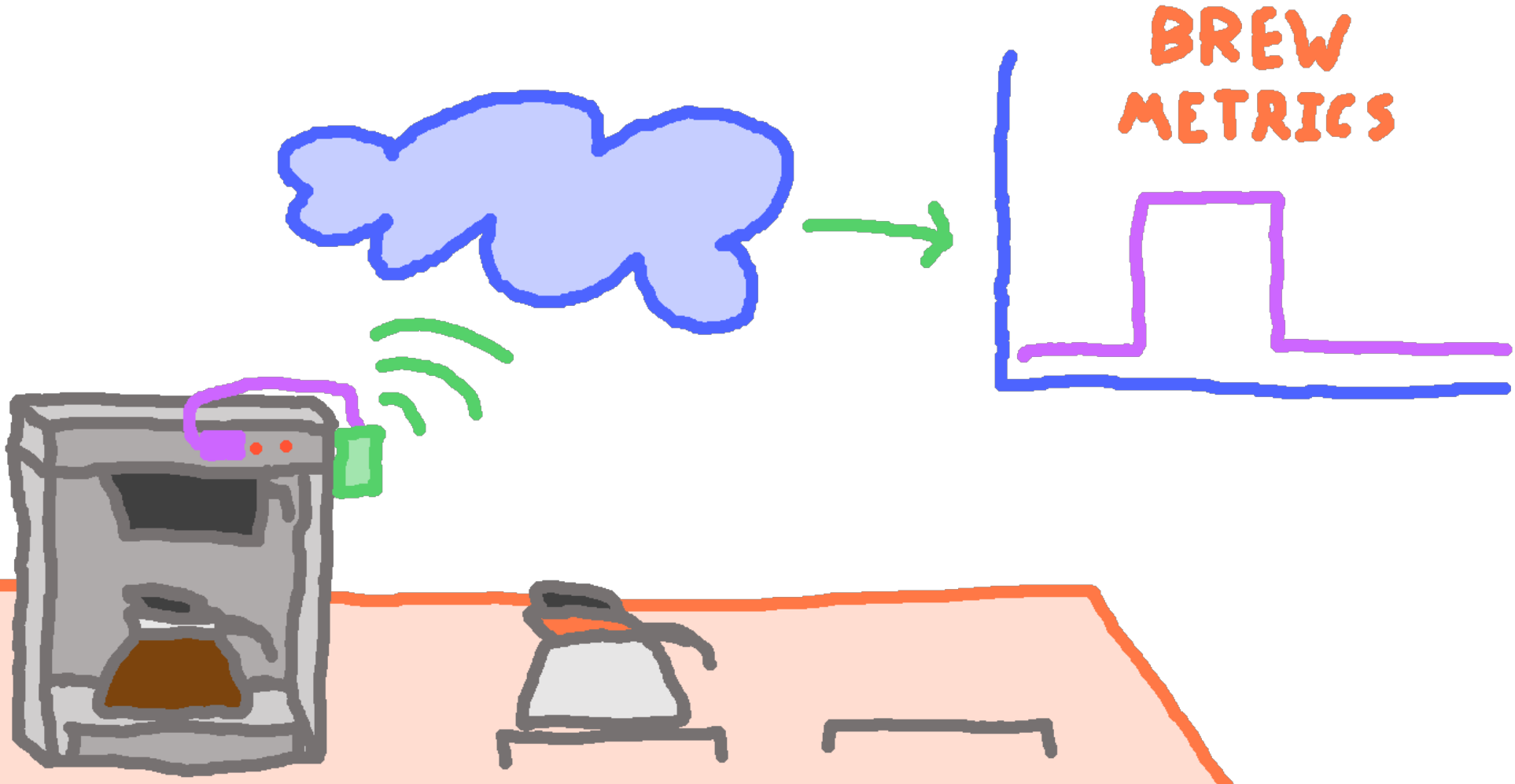


Understandable by anyone
(no jargon!)

Customer impact



Graphs and timeline



Timeline

- 08:29 – Operator accidentally selects “decaf” coffee packet
- 08:30 – Operator selects “regular” coffee pot
- 08:31 – Operator loads coffee and starts brewing
- 08:32 – Operator goes back to desk
- 08:36 – Coffee finishes brewing
- 08:?? – Someone removes carafe from coffee maker and takes a cup
- 08:45 – Operator returns and takes the first cup
- 09:25 – Operator returns for another cup
- 09:26 – Operator realizes that the coffee tastes different than normal
- 09:27 – Investigation begins
- 09:29 – Original coffee packet retrieved, confirmed decaf
- 09:30 – Decaf label transferred to regular carafe

WHY?

WHY?

WHY?

WHY?

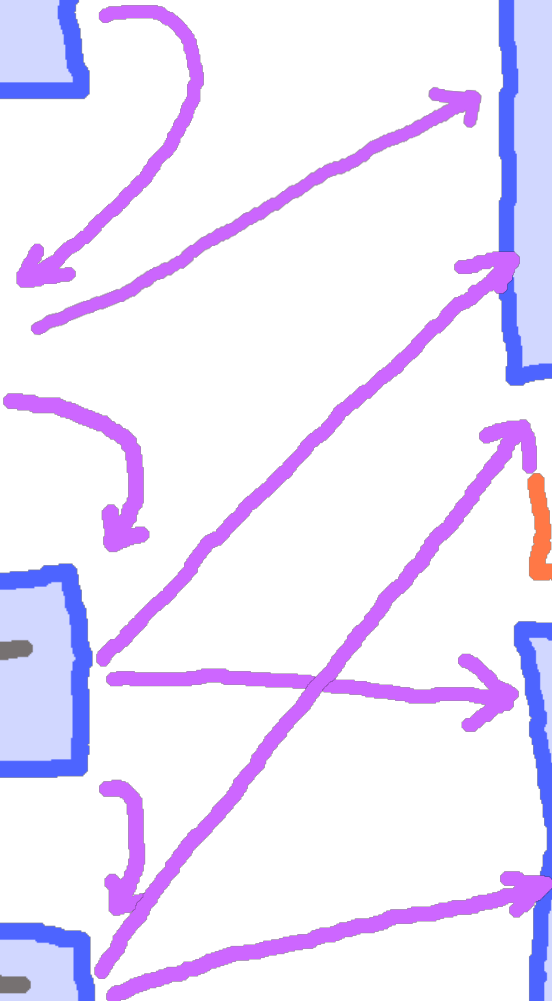
ACTION ITEMS

1) _____

2) _____

3) _____

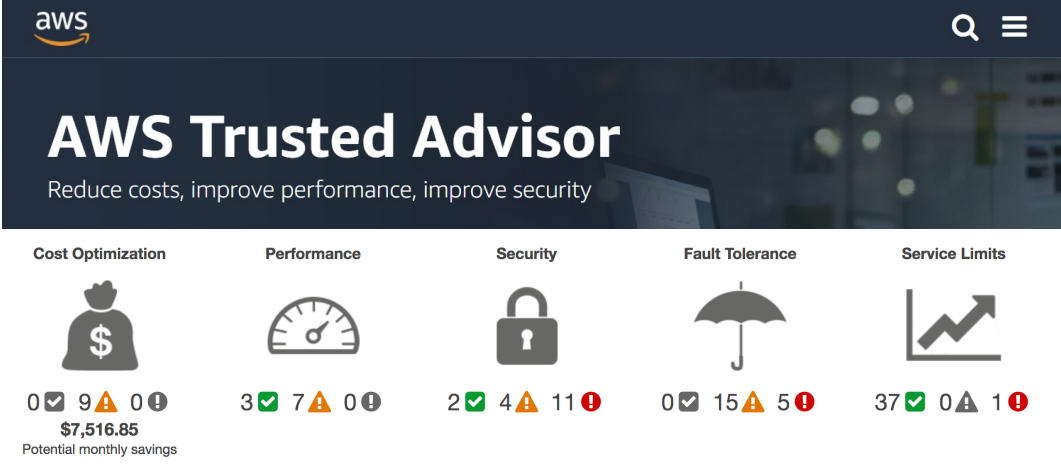
LESSONS LEARNED








Sharing and improving broadly

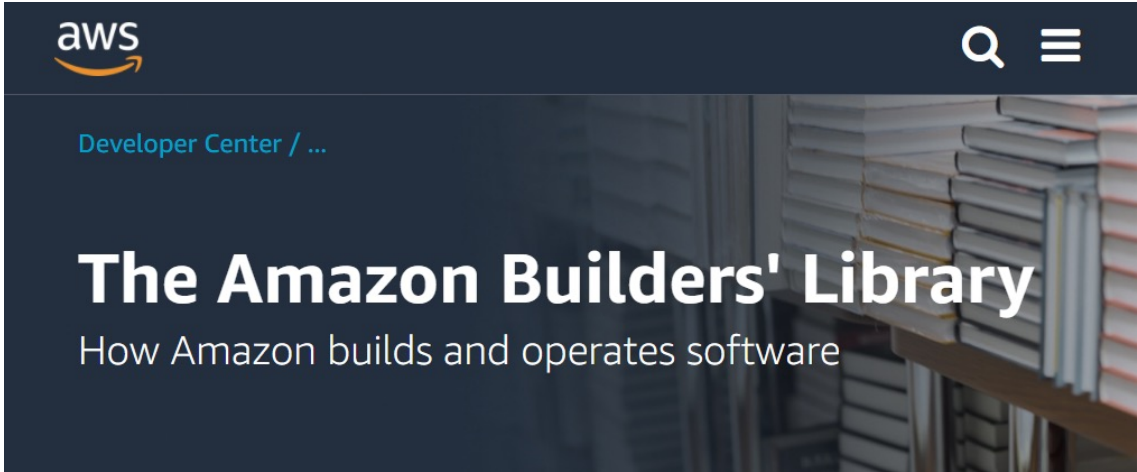


The screenshot shows the AWS Well-Architected landing page. At the top left is the AWS logo. To the right are search and menu icons. The main heading is "AWS Well-Architected" in large white font. Below it is the subtitle "Learn, measure, and build using architectural best practices". At the bottom, there is a yellow button that says "Get started with the AWS Well-Architected Tool". The background features a pattern of interlocking hexagons in shades of brown and blue.



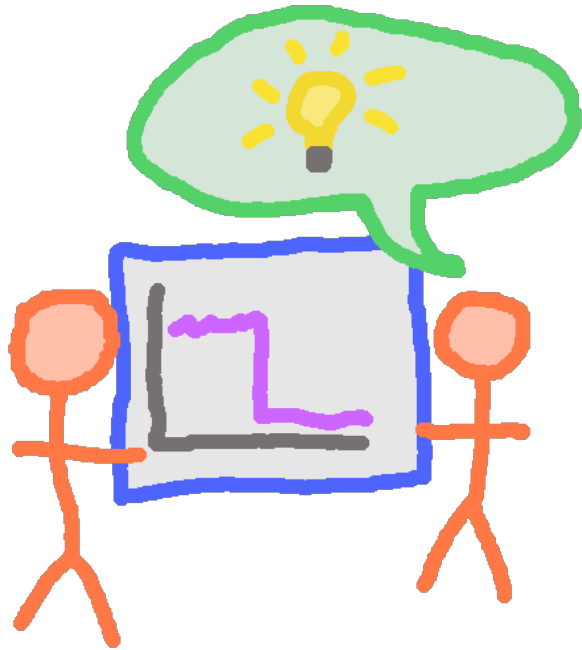
The screenshot shows the AWS Trusted Advisor dashboard. At the top left is the AWS logo. To the right are search and menu icons. The main heading is "AWS Trusted Advisor" in large white font. Below it is the subtitle "Reduce costs, improve performance, improve security". The dashboard is divided into five columns, each with an icon, a title, and a set of status indicators (checkmarks, warning triangles, and exclamation marks).

Cost Optimization	Performance	Security	Fault Tolerance	Service Limits
				
0 ✓ 9 ⚠ 0 ! \$7,516.85 Potential monthly savings	3 ✓ 7 ⚠ 0 !	2 ✓ 4 ⚠ 11 !	0 ✓ 15 ⚠ 5 !	37 ✓ 0 ⚠ 1 !

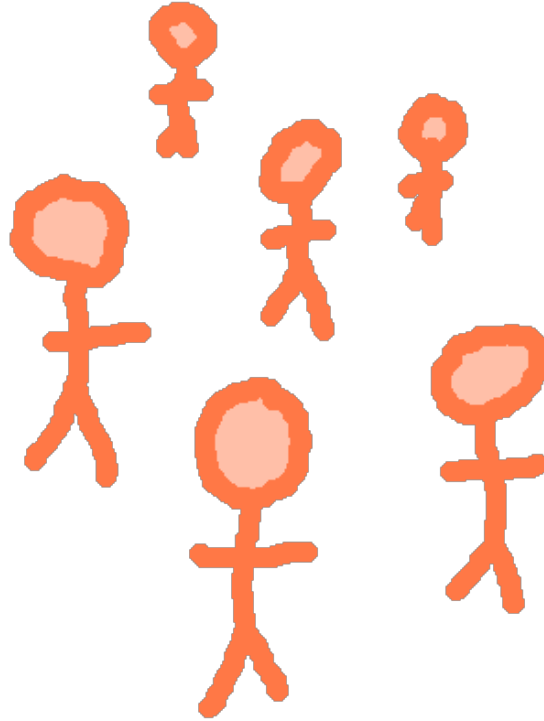


The screenshot shows the Amazon Builders' Library landing page. At the top left is the AWS logo. To the right are search and menu icons. Below the logo is the text "Developer Center / ...". The main heading is "The Amazon Builders' Library" in large white font. Below it is the subtitle "How Amazon builds and operates software". The background features a stack of books on a shelf.

Takeaways: The retrospective



Improve systems



Teach others

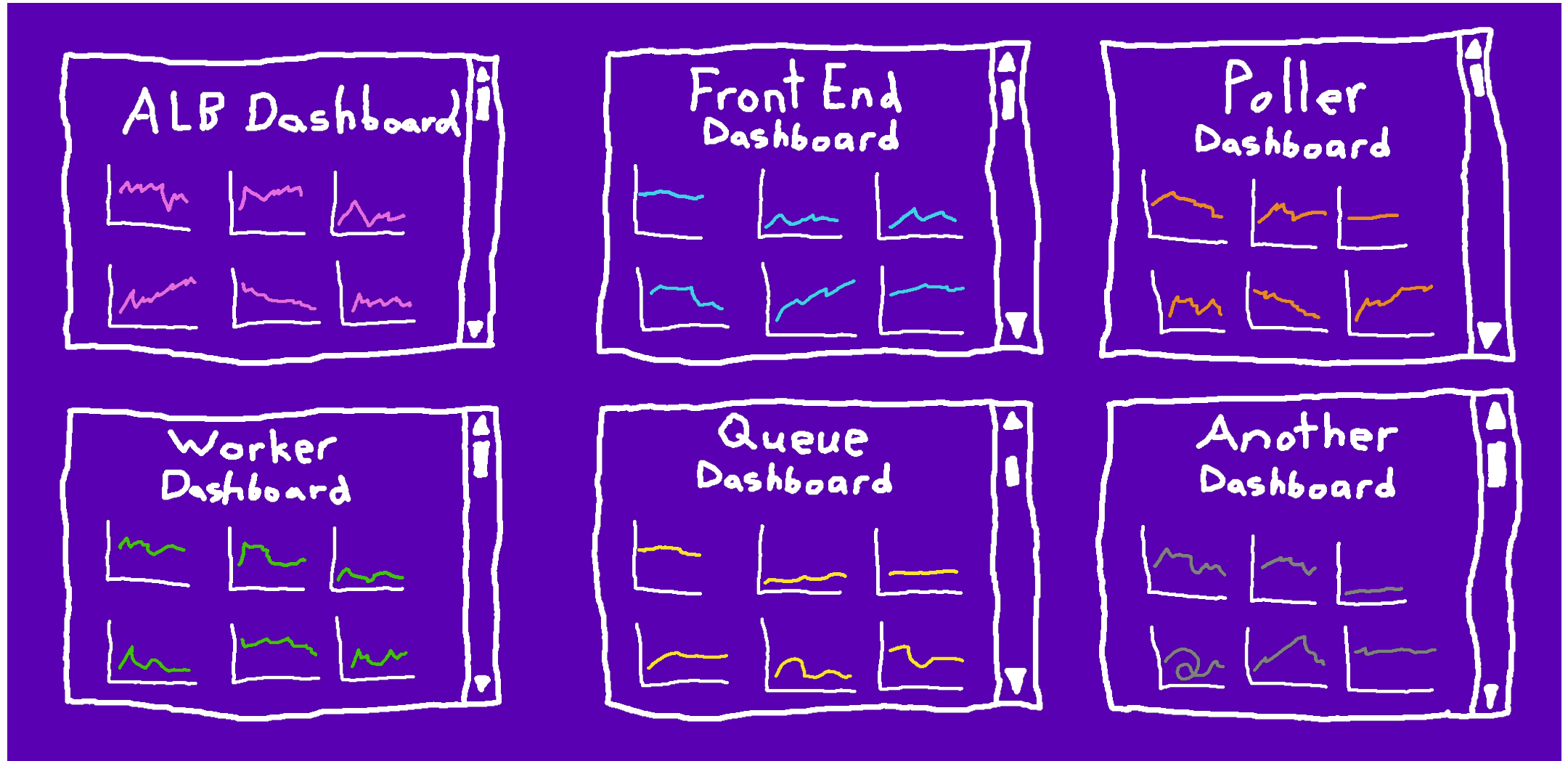


Improve tools

Story: The ops meeting



Dashboards!



<https://aws.amazon.com/builders-library/building-dashboards-for-operational-visibility/>

Morning metrics



Weekly team operations meeting



Weekly ops agenda

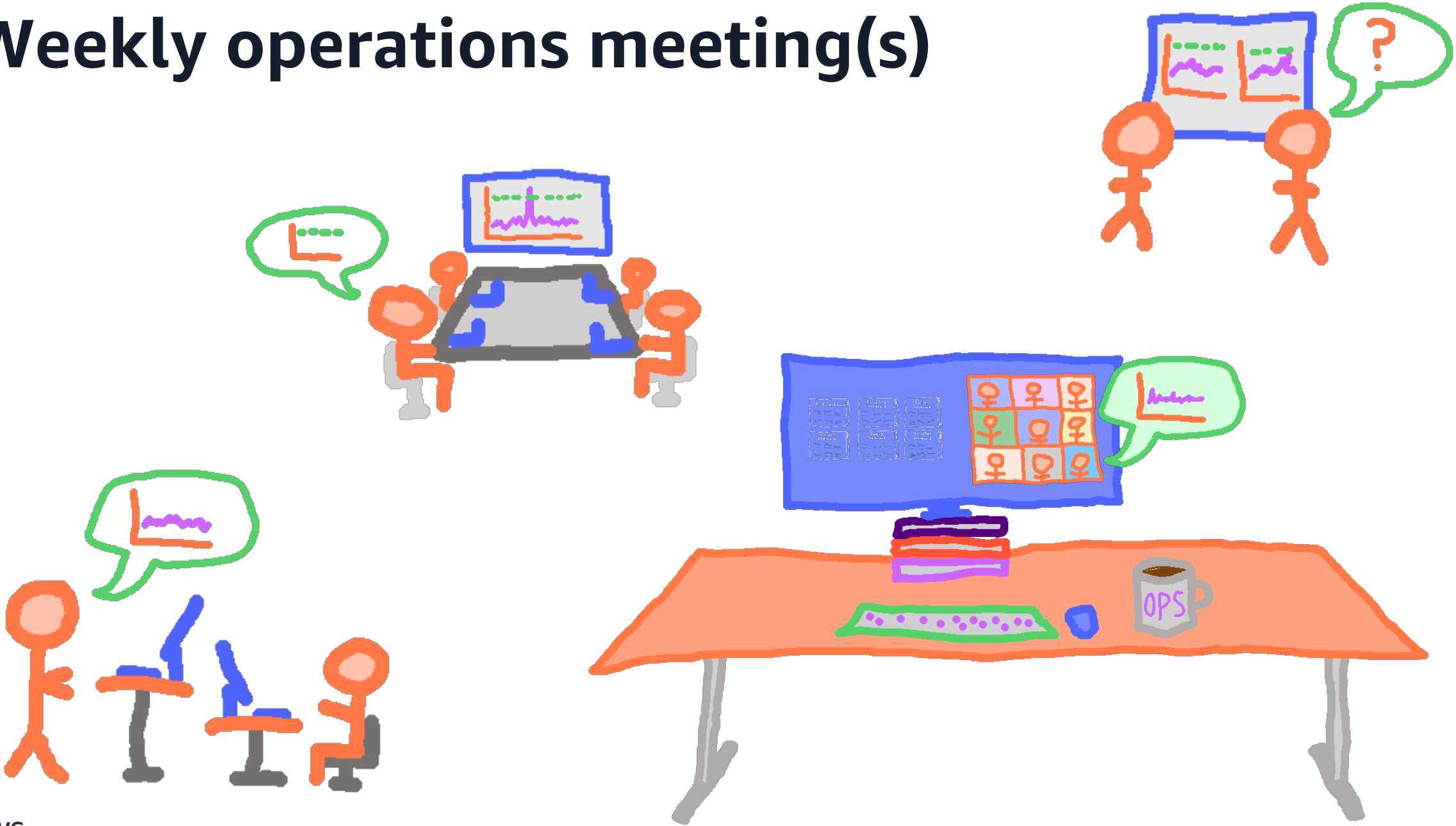
Wins

Retrospectives

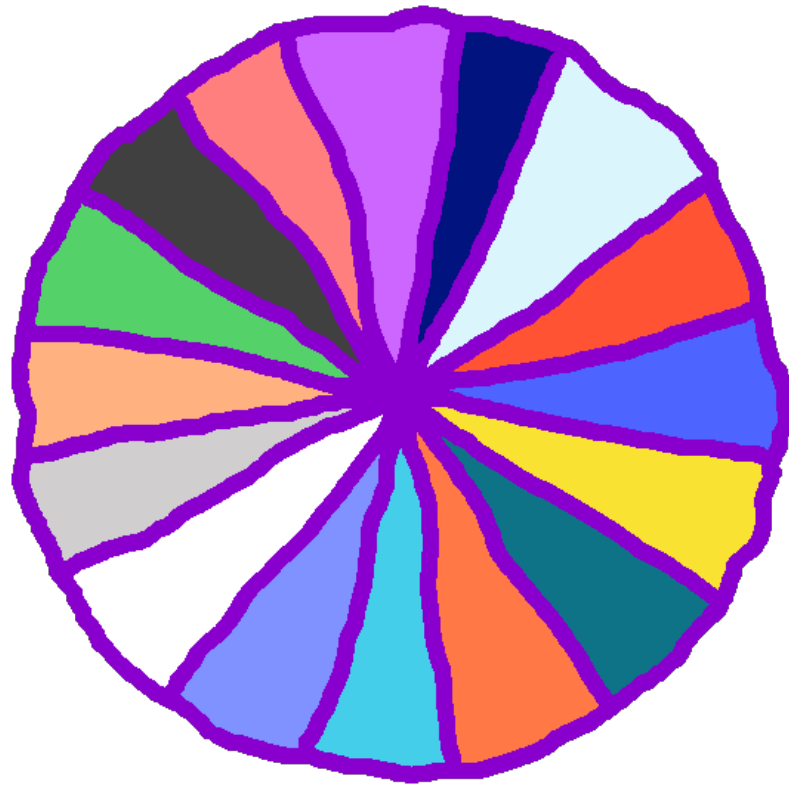
Dashboards



Weekly operations meeting(s)



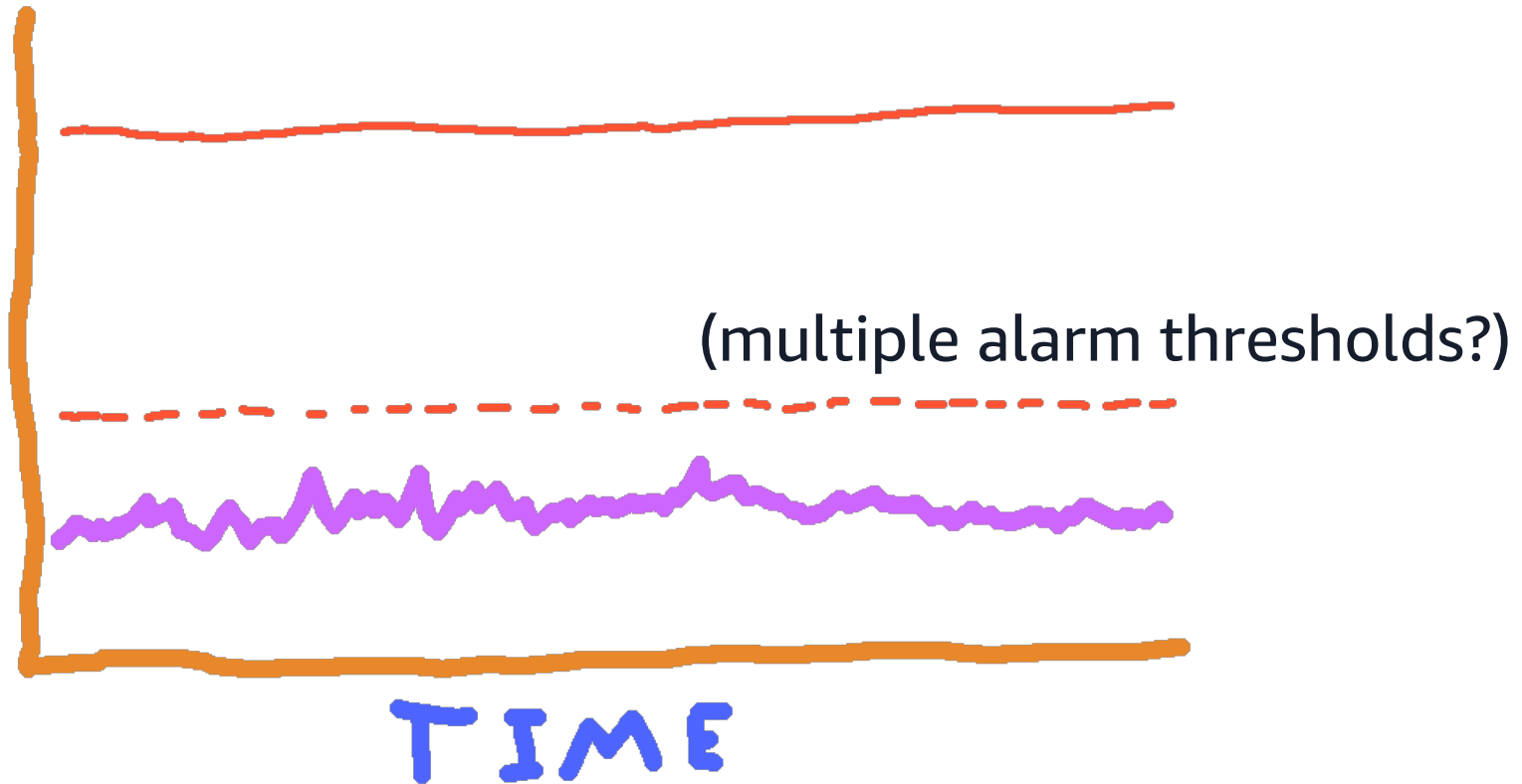
Spinning the wheel



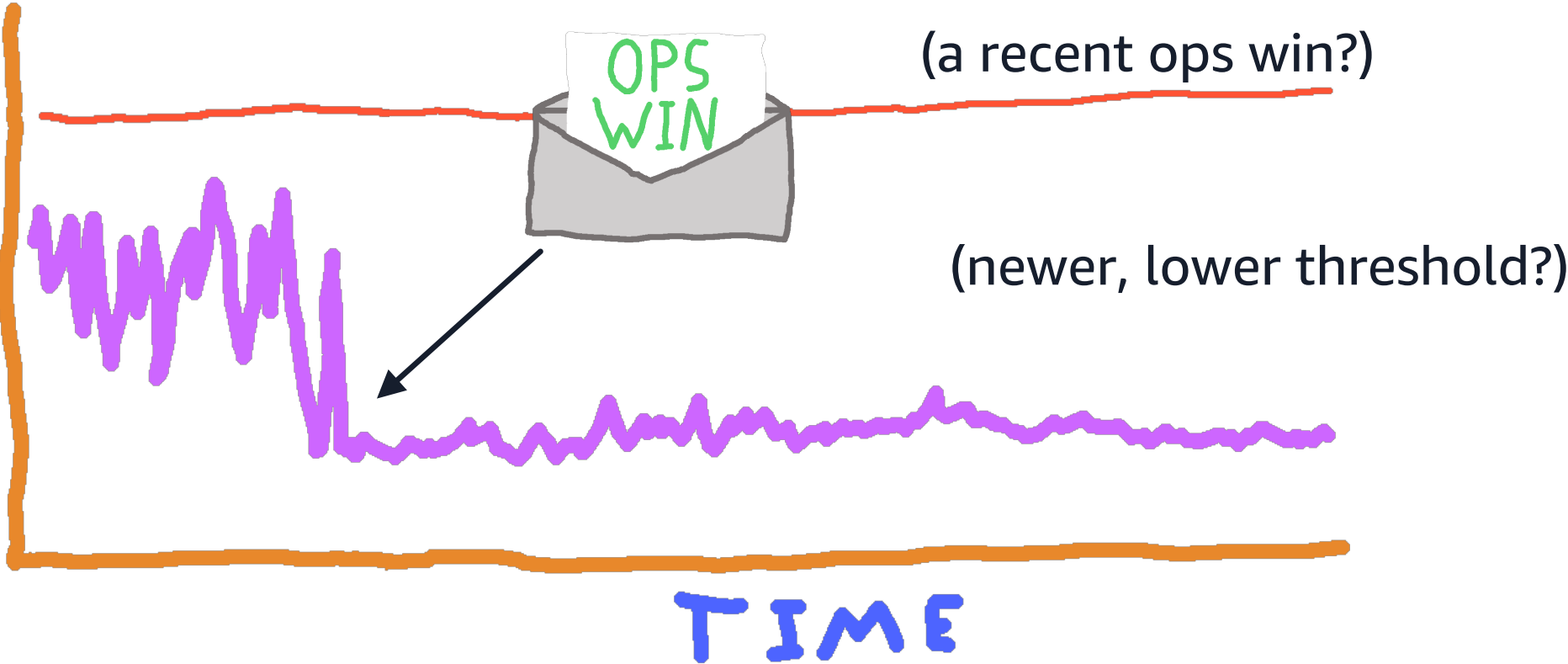
<https://github.com/aws/aws-ops-wheel>

Feedback on dashboard metrics

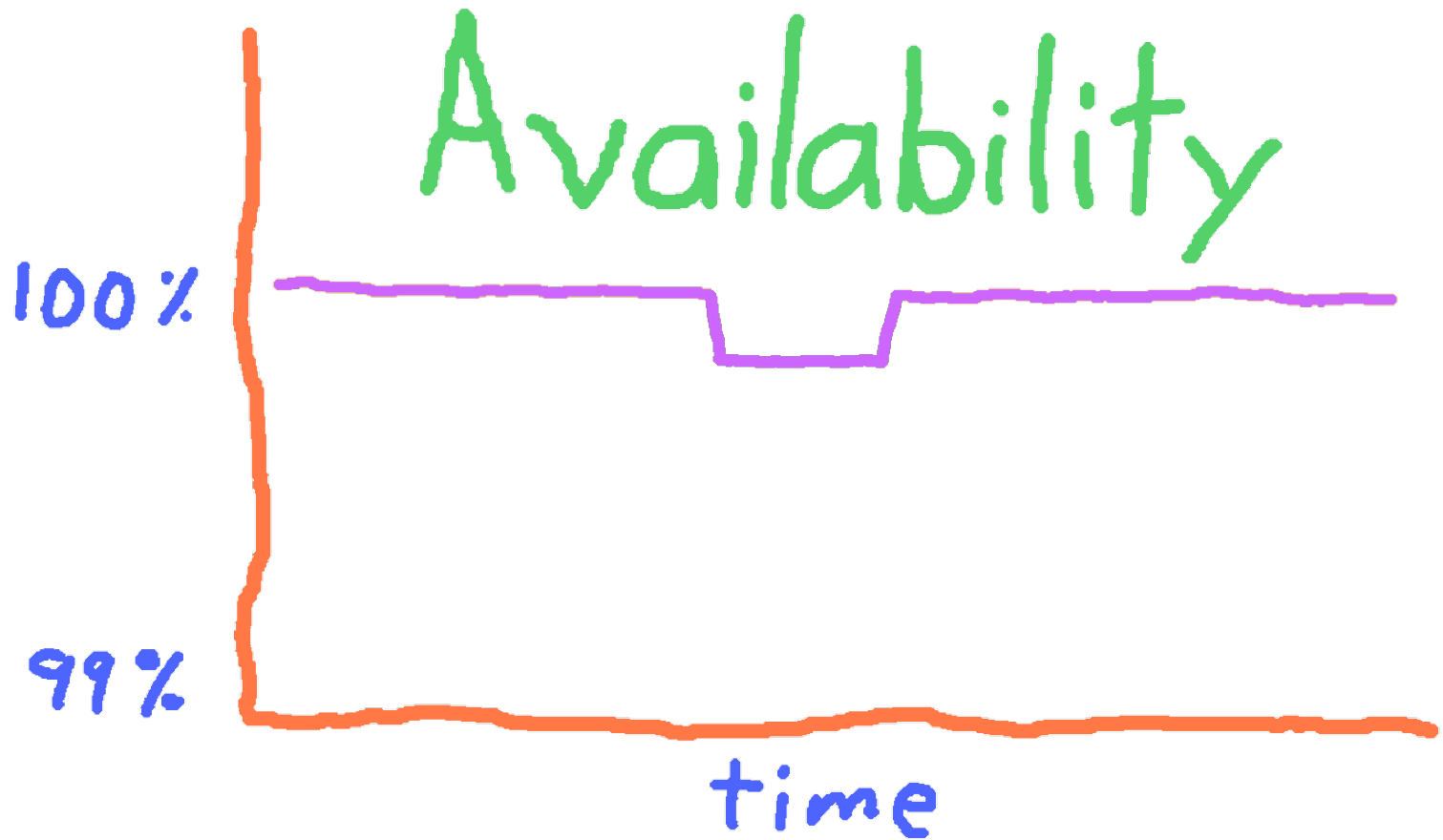
API LATENCY



API LATENCY



Overall availability

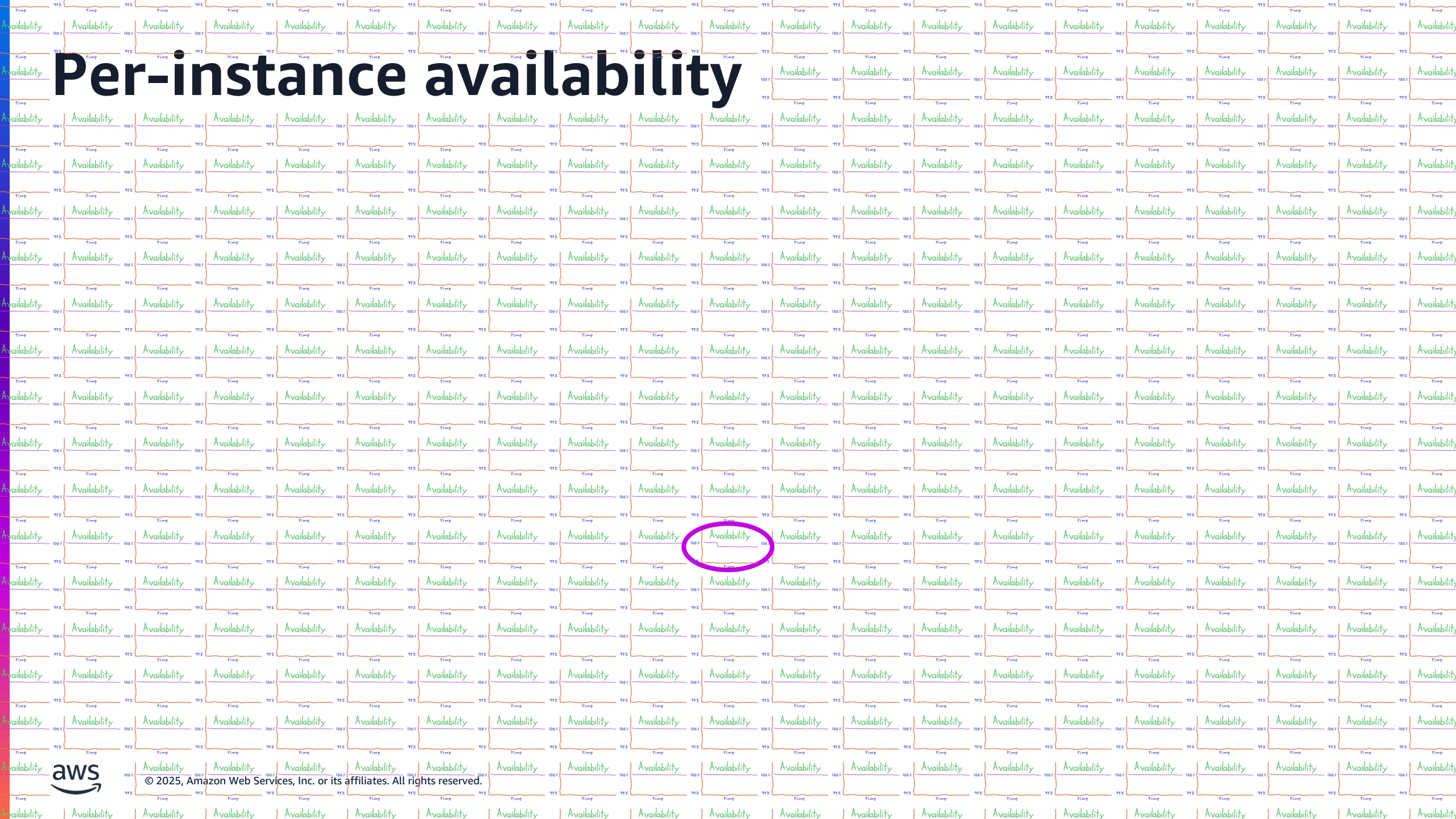


Per-instance availability



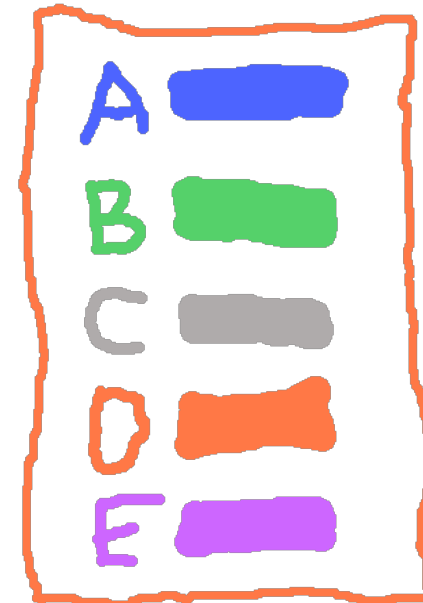
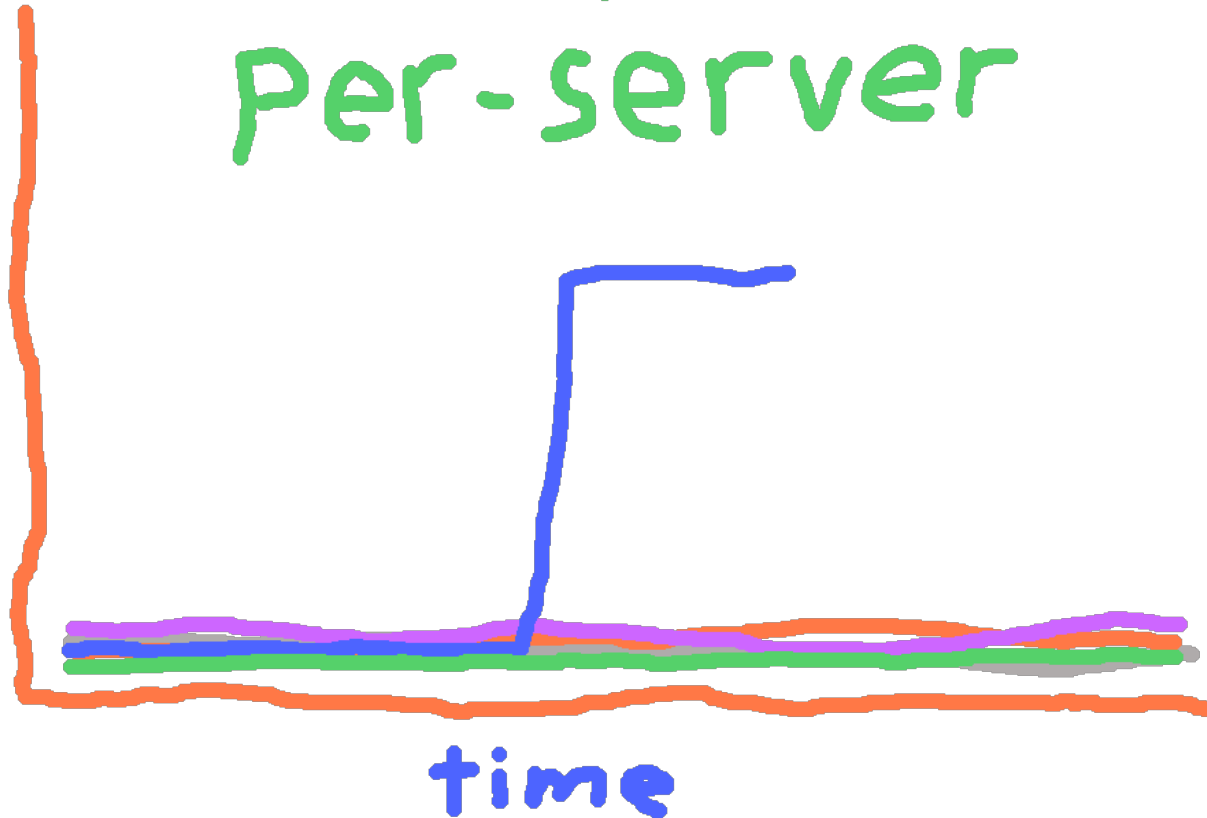
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Per-instance availability



Top-N instances by error count

Errors
Per-server

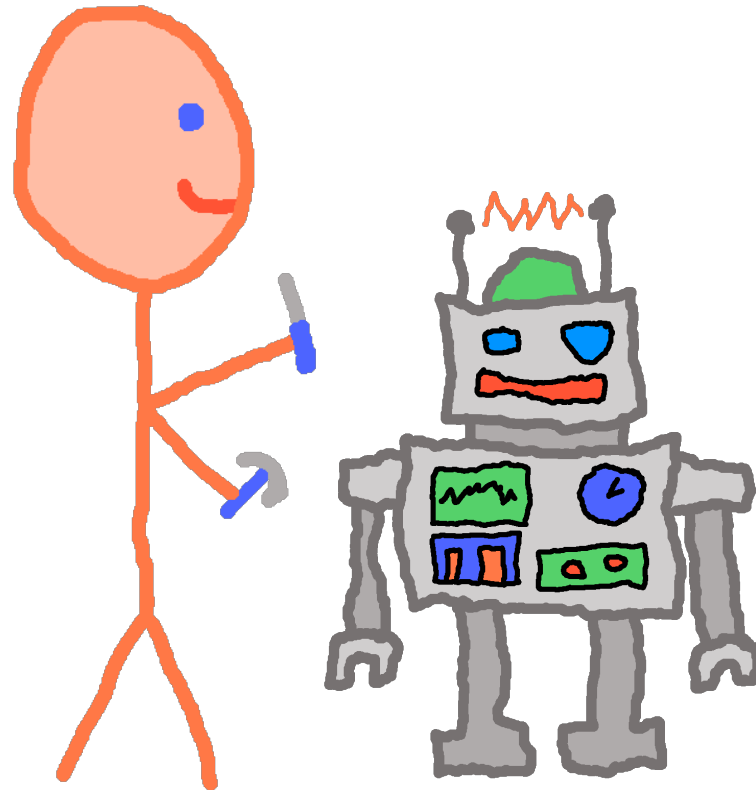




Amazon CloudWatch Contributor Insights

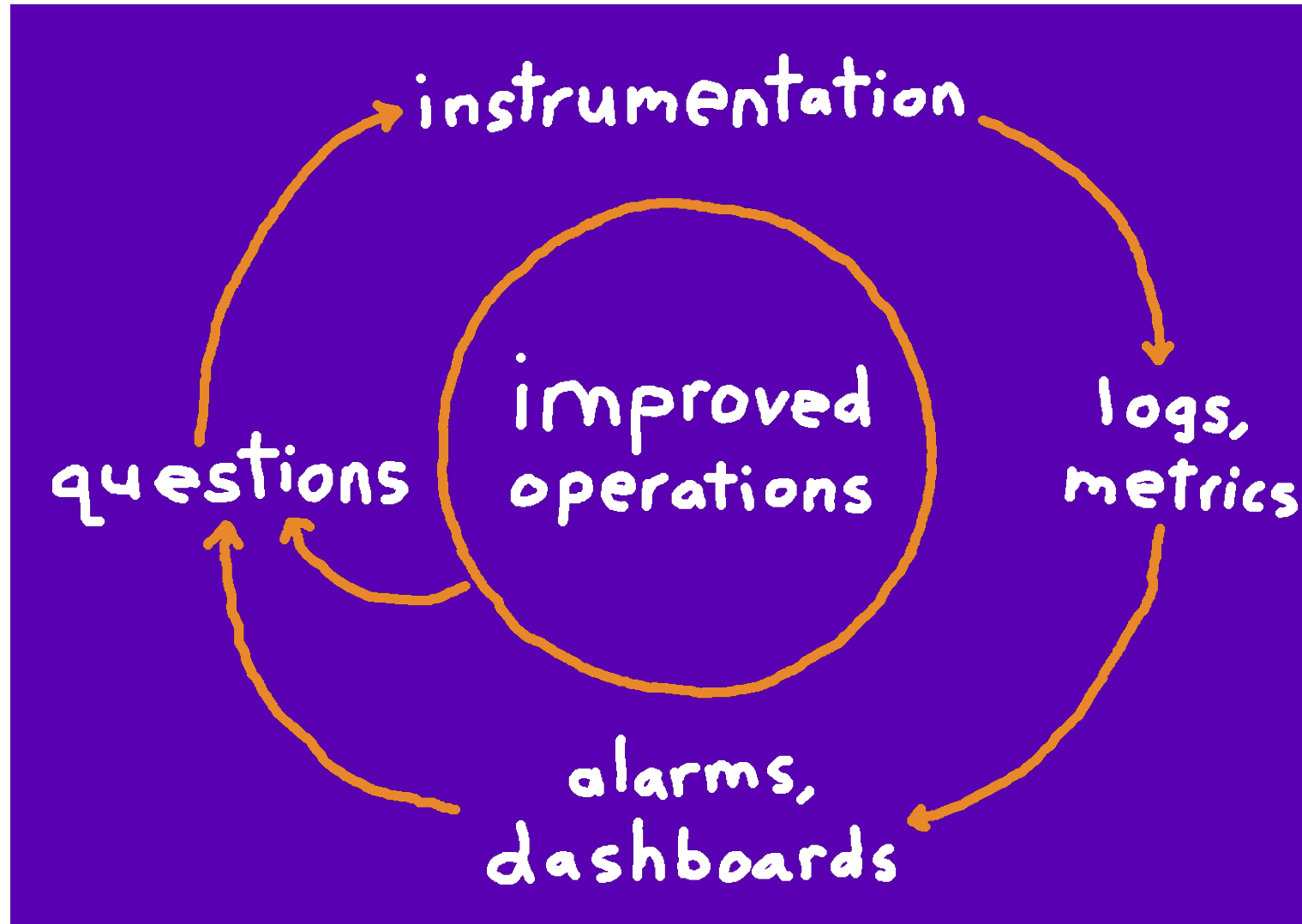
Amazon CloudWatch Contributor Insights allows you to easily view the top contributors impacting the performance of your systems and applications in real-time.

Automation



<https://aws.amazon.com/builders-library/implementing-health-checks/>

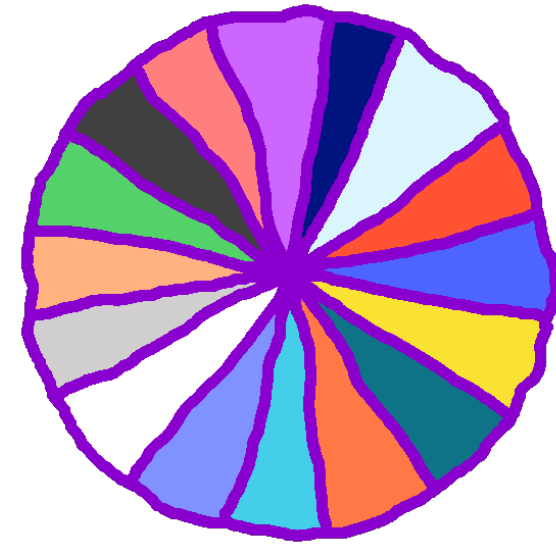
The cycle of monitoring



Takeaways: The ops meeting



Learn from each other



Practice regularly