

Next '24

How ANZ delivers
reliable applications
on Google Cloud using
archetype-based
platforms





Hello Next !

- Joined ANZ in 2015
- 12 years as a hands-on Software Engineer
- 5 years as a leader in Cloud Platforms
- Live in Melbourne, Australia
- Passionate about putting the ‘dev’ in devops
- Avid gardener & sports fan!



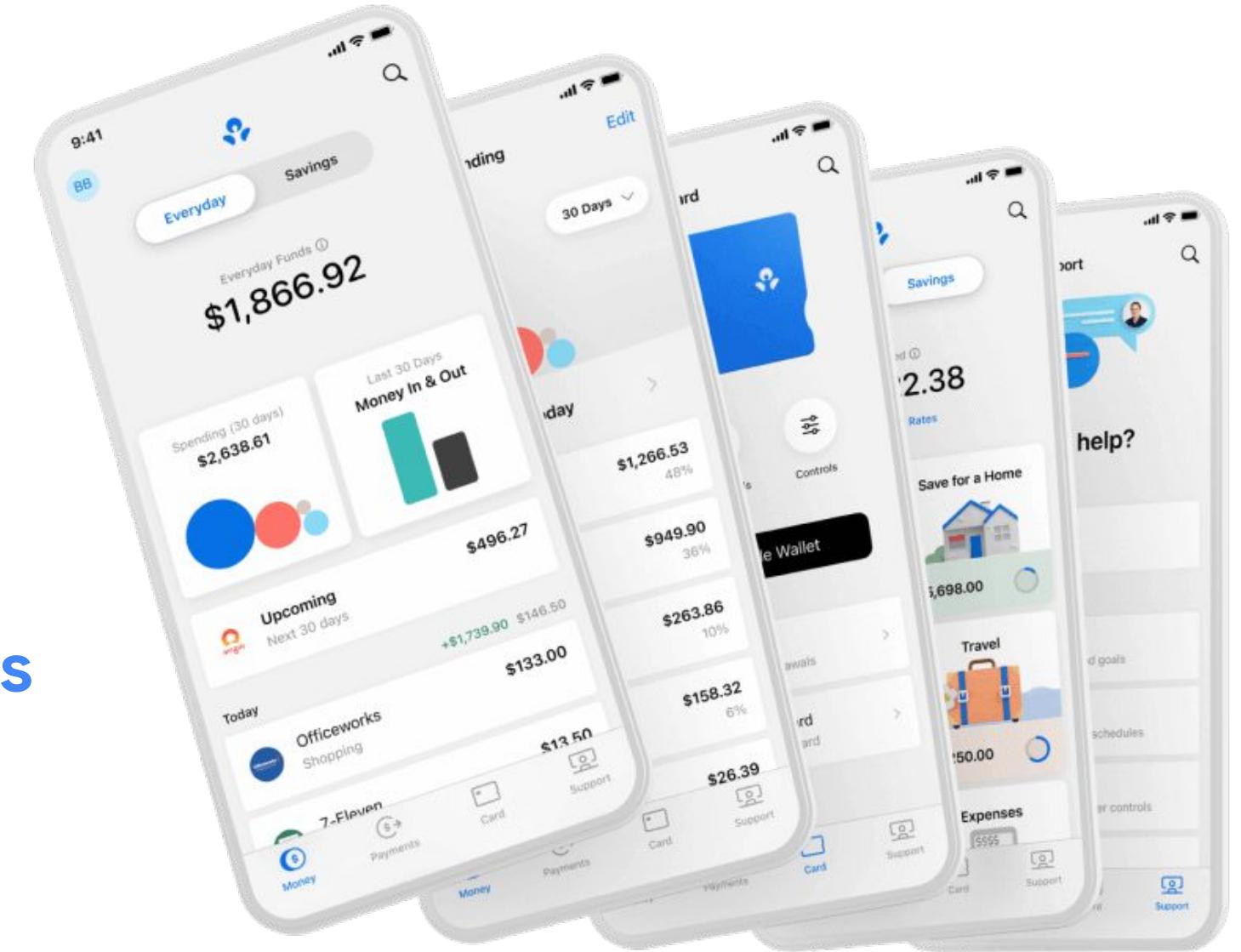
**Daniel
McKeown**
Area Lead Cloud &
Orchestration Platforms,
ANZ Bank

<https://www.linkedin.com/in/danielmckeown/>



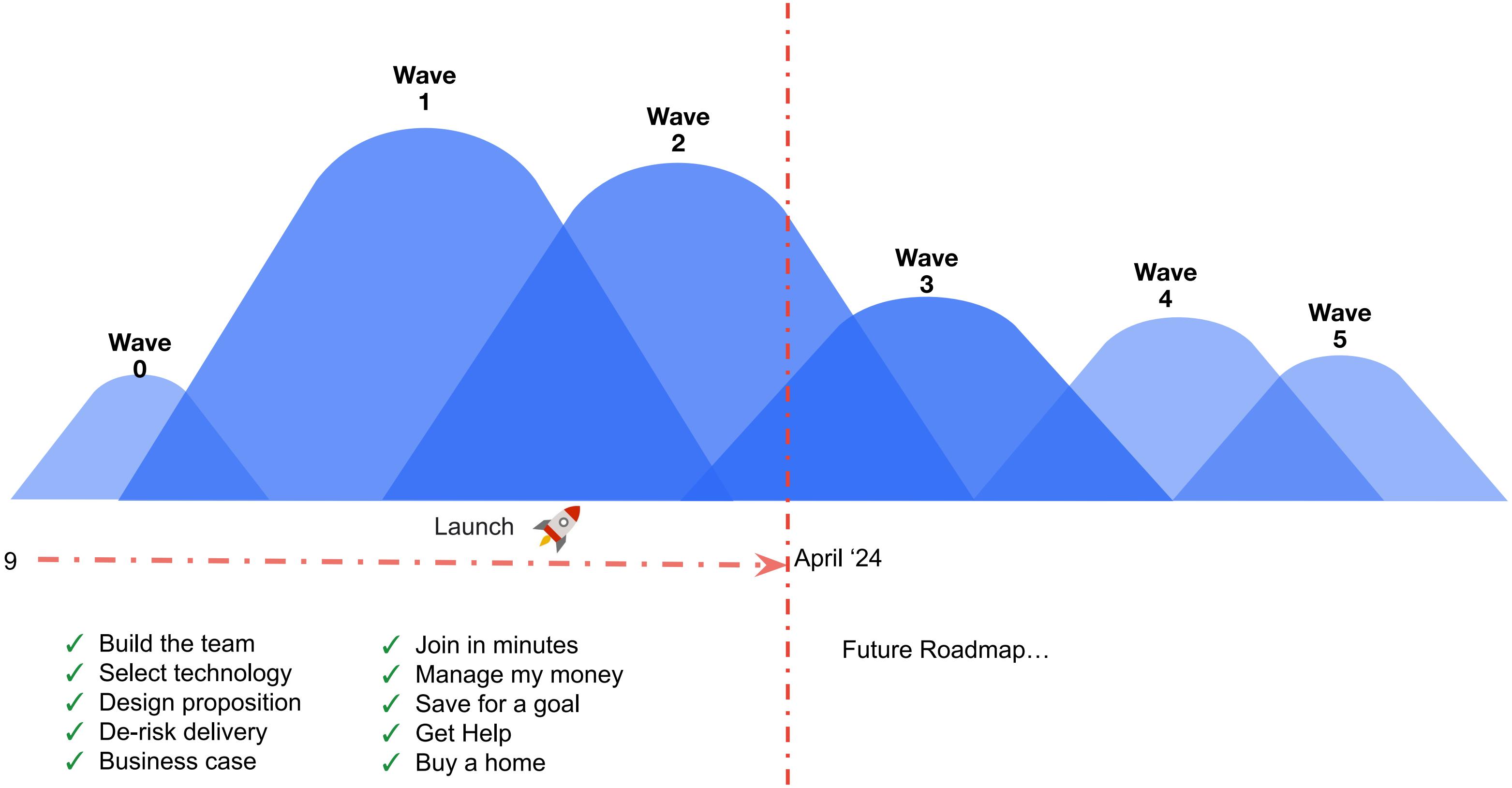
Building a cloud-native digital bank

- ANZ is an 186 year old Australian Bank
 - \$77.1B in market capitalisation
 - 40K employees
 - \$1.1T in funds under management
 - 8.5M customers
 - Operations in 29 countries
- Building a new digital banking proposition called **ANZ Plus**
 - New mobile and web applications
 - Simplified product offering
 - Cloud-native and hosted on Google Cloud





We are part way through a multi-year transformation







ANZ Plus scale on Google Cloud

-  **2500 Deployments**
-  **1700 Namespaces**
-  **1500 Cloud Run Services and Jobs**
-  **1000 Cloud Functions**
-  **1200 Code repositories**
- ...



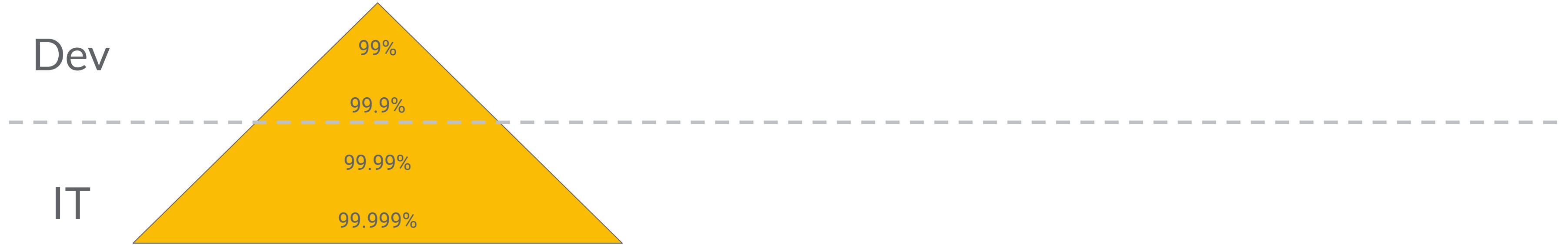


**Steve
McGhee**
**Reliability
Advocate,
Google Cloud**



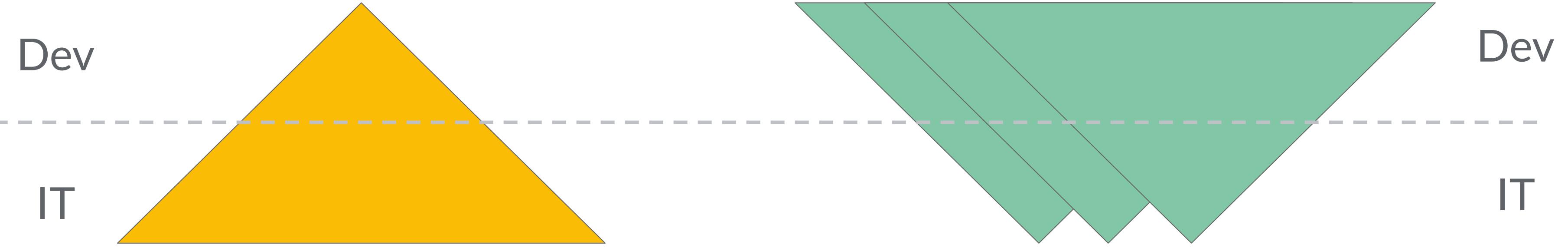
**Ameer
Abbas**
**Outbound Product
Manager,
Google Cloud**

Can you build
99.99 services
on
99.9 infrastructure?



Worked great, for
a long time

Common
mental model



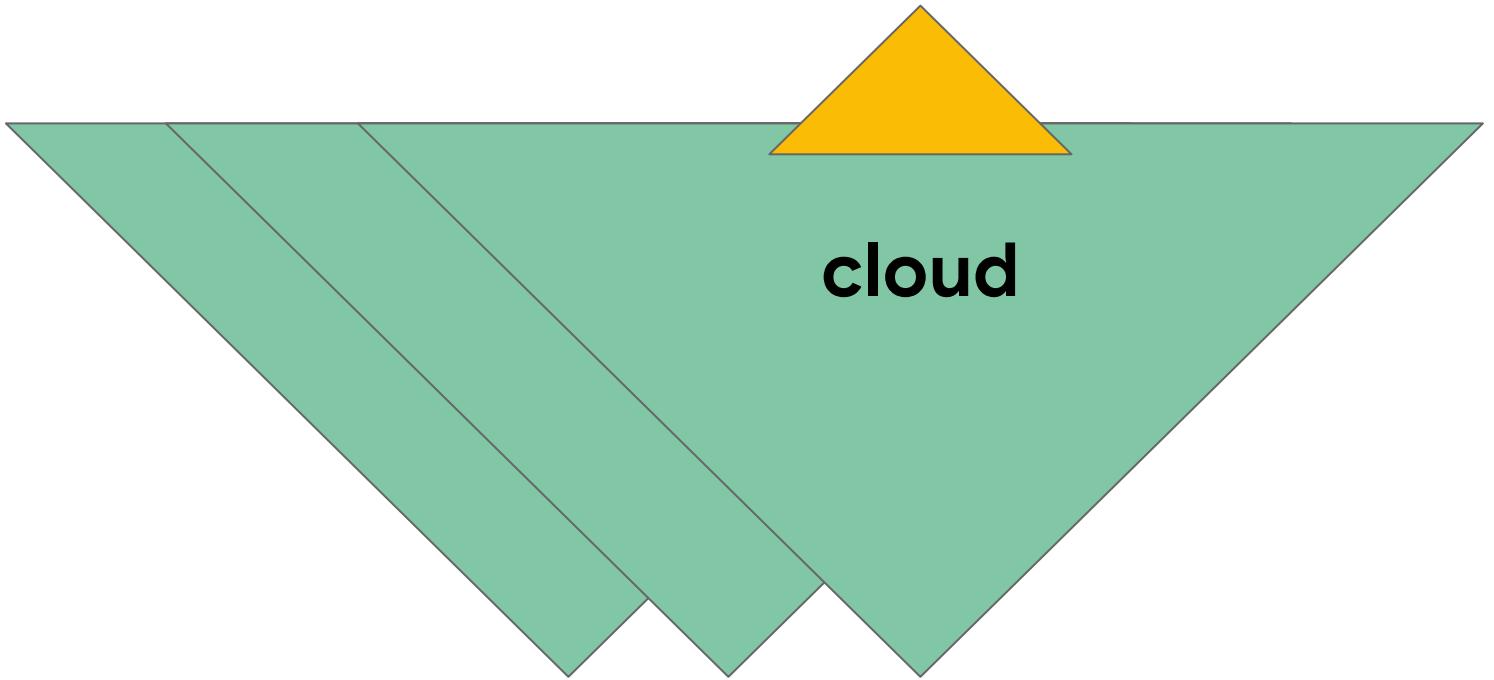
This works!

"Makes sense"

Cloud is here, though.

Because: Scale

A common failure mode we've all seen:



Infrastructure changes can't fix the app.

Platforms, Personas, and Outcomes

"Are we getting better?"



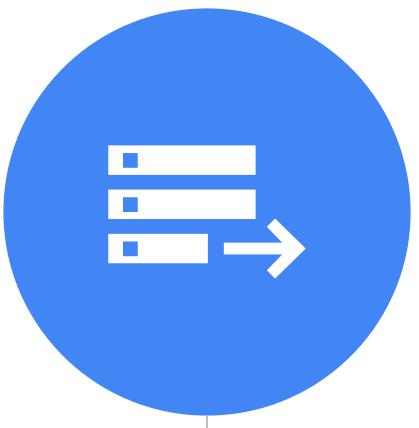
Speed



Stability

"Are we getting better?"

The 4 DORA Metrics



Speed

Deployment
frequency

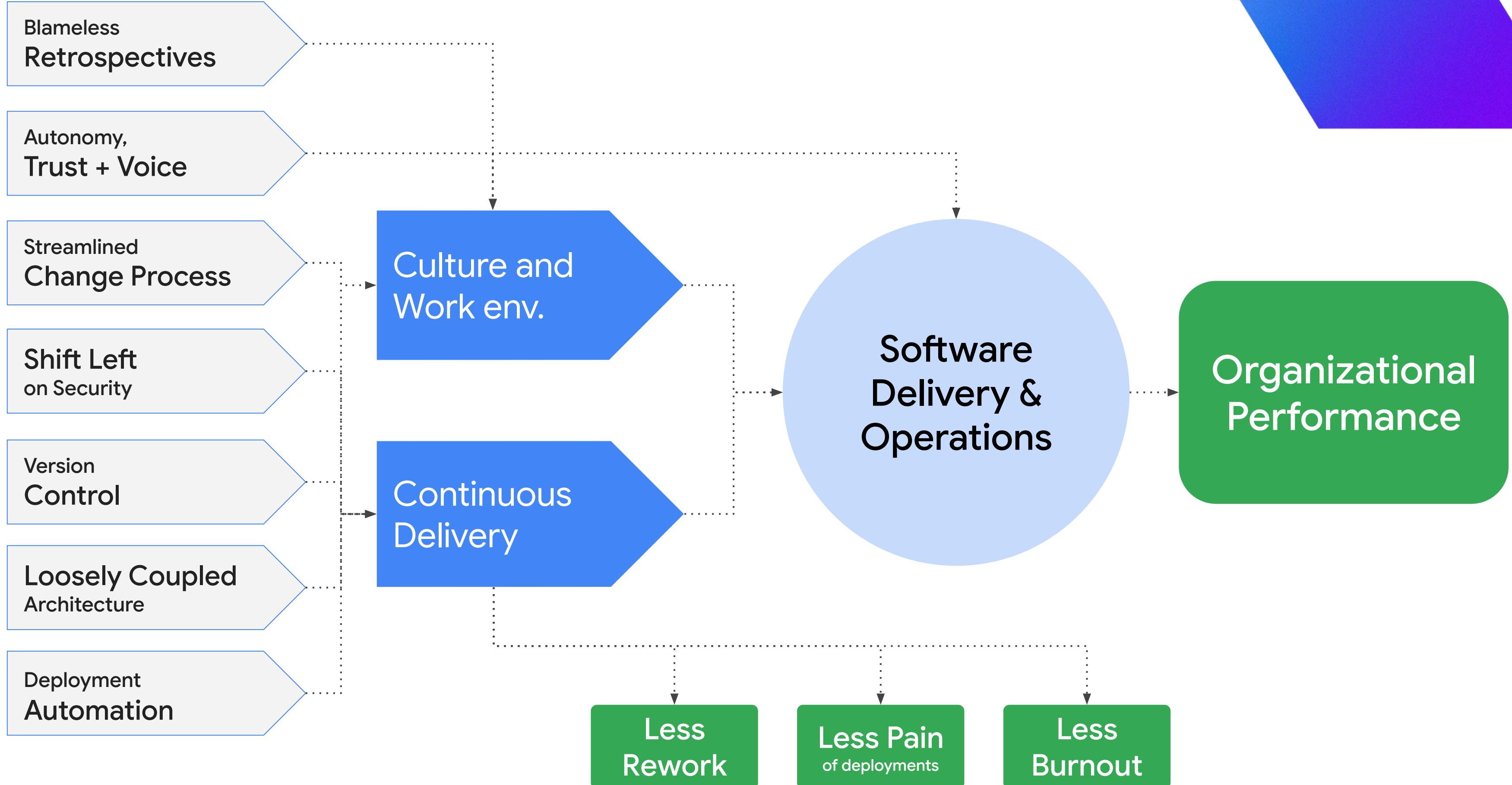
Lead time
for changes

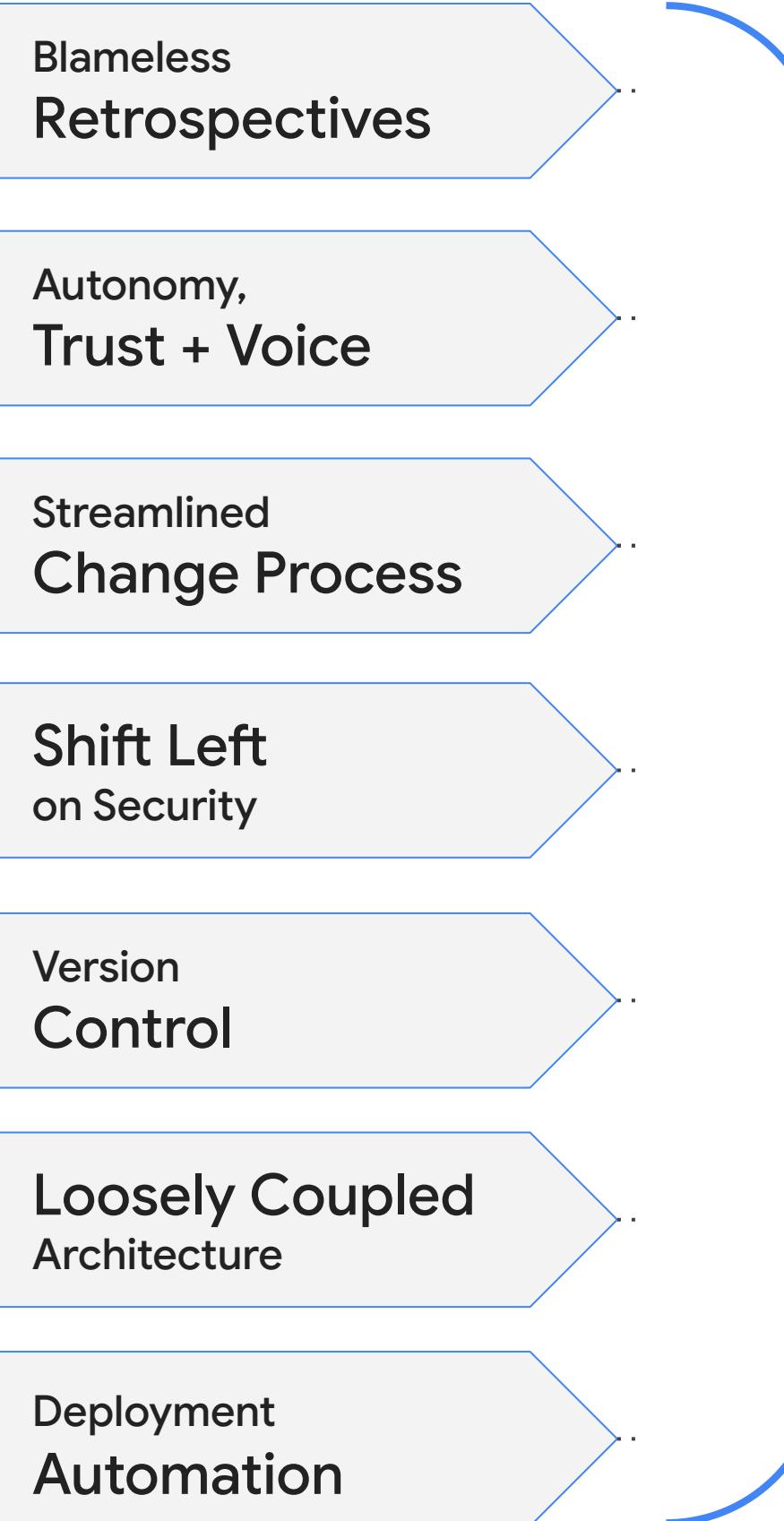


Stability

Change fail
rate

Time to
restore service





Capabilities

- technical
- process
- cultural

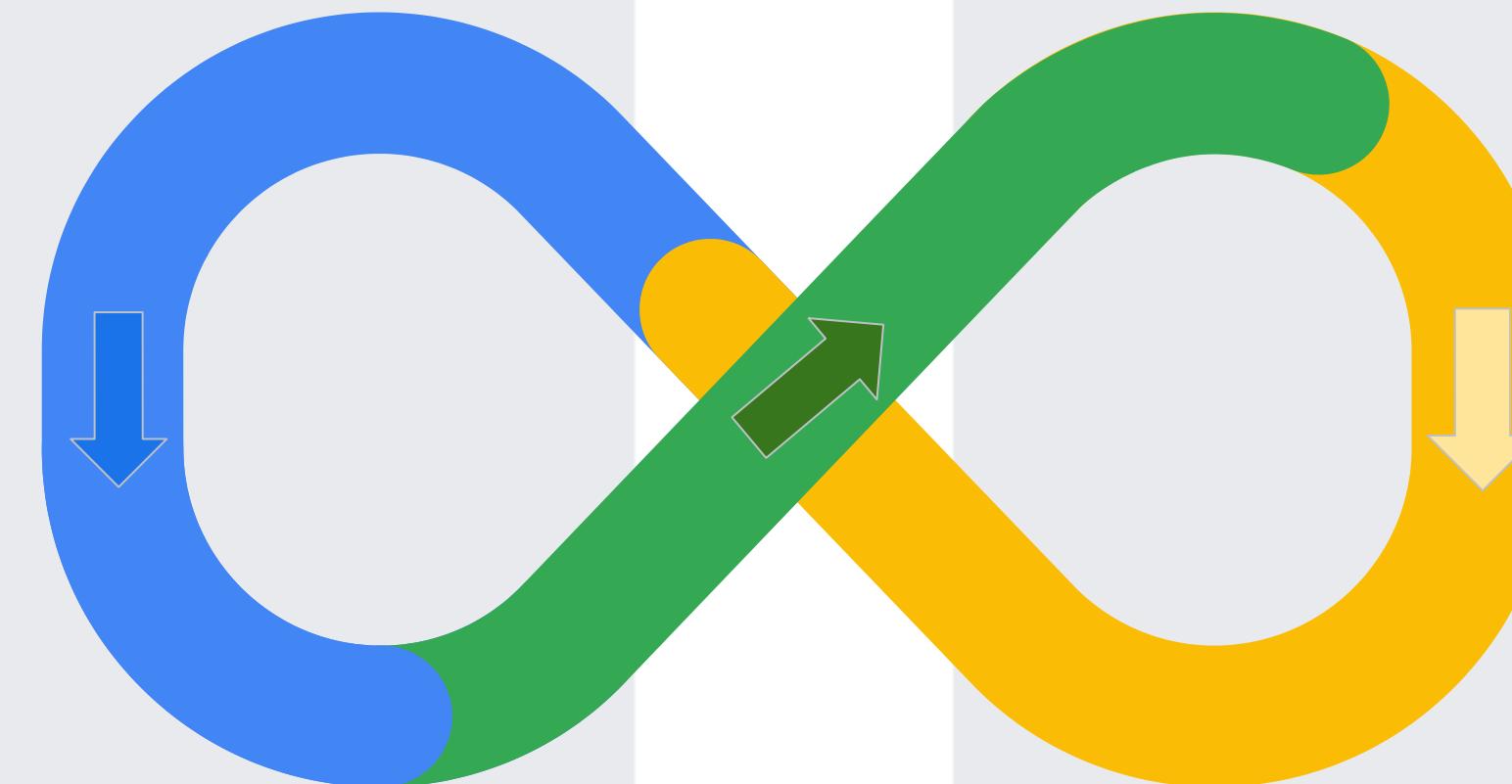
→ build, buy, or adopt

Measure

1. Measure 4 DORA metrics



2. Determine bottleneck(s)



3. Choose **capability** to improve next

Improve

4. Build / buy capabilities



5. Enable capability in platform, document

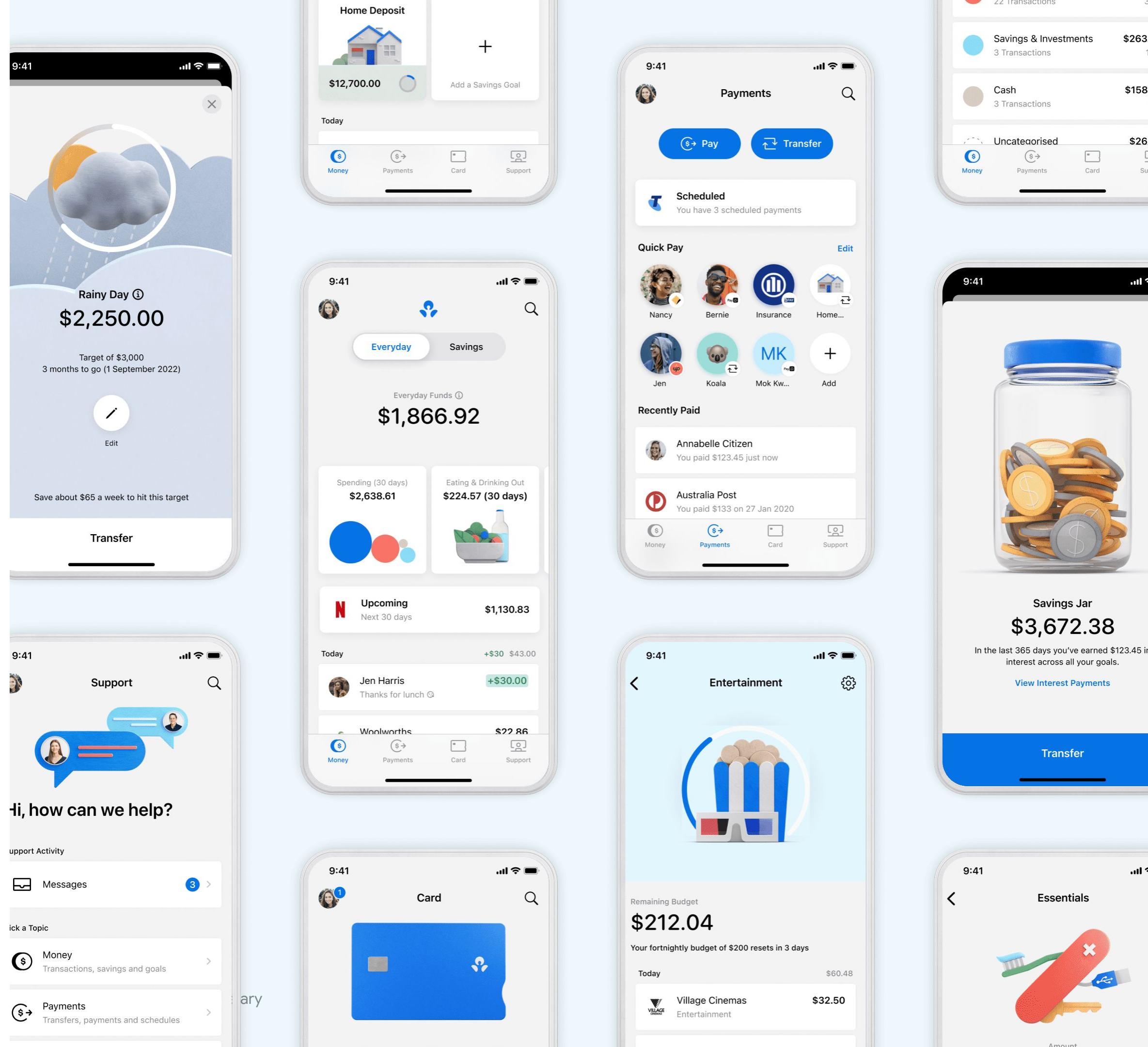
6. Gather early dev feedback

7. Release next version of platform



Measuring success

- To know what good looks like, you must first establish **baseline measures**
- Measures aren't only useful for building **platforms**
- We use the **four key DORA metrics** at ANZ





How have we done it?

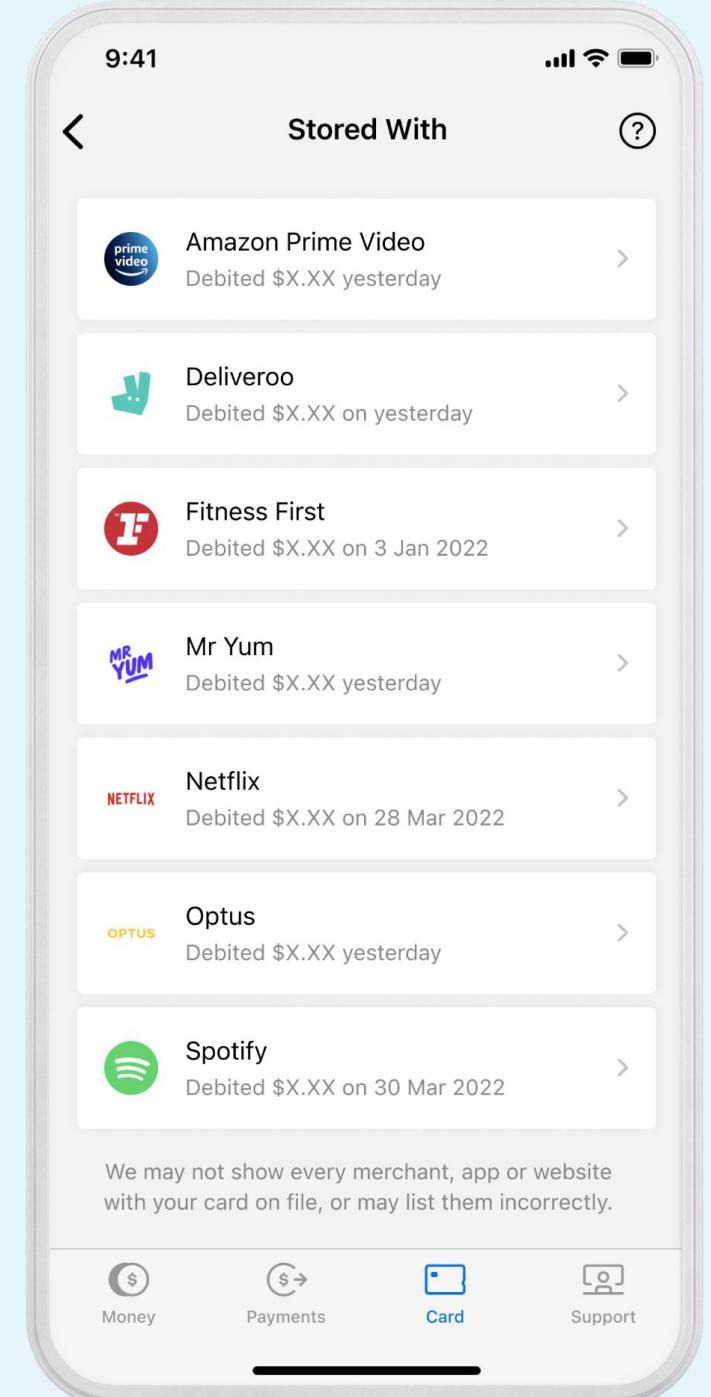
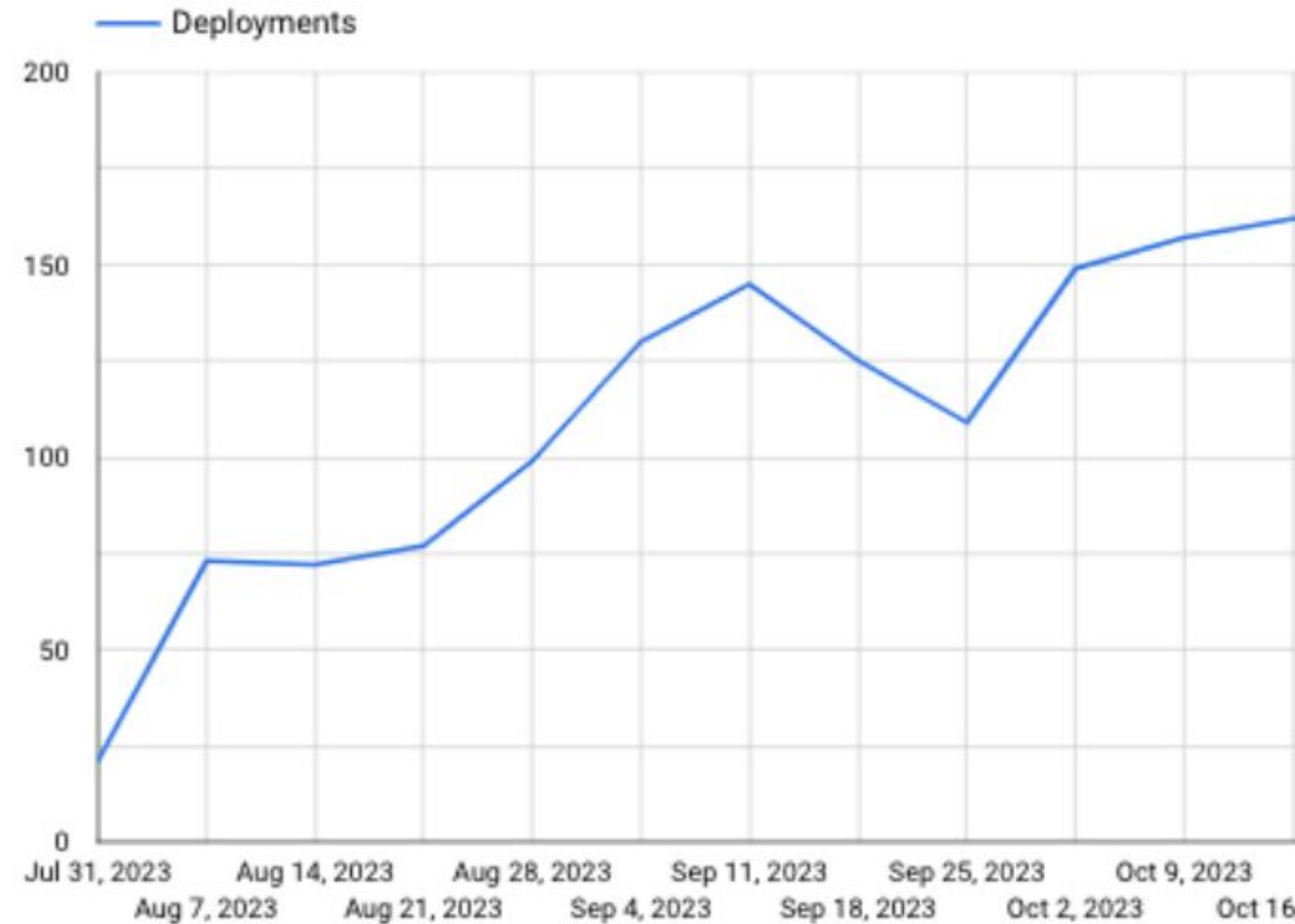
- Scheduled batch loads of historical data from **Jira**, **GitHub** and **Service Now**
- **dbt** for ETL
- **Big Query** as our Data Warehouse
- **Looker Studio** as our Data Presentation layer



Platforms drive devops health

Weekly Deployments

The number of deployments per week



Platform Engineering

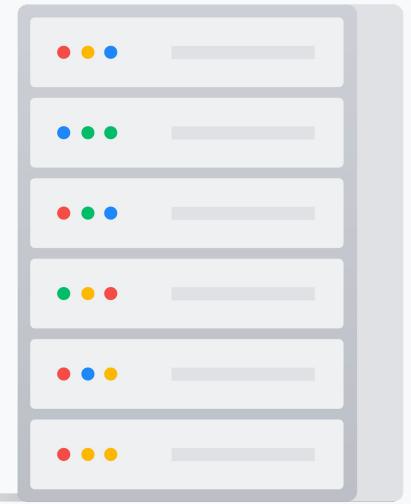
More focus
More creativity
More agility



Software Developer



Platform Engineer

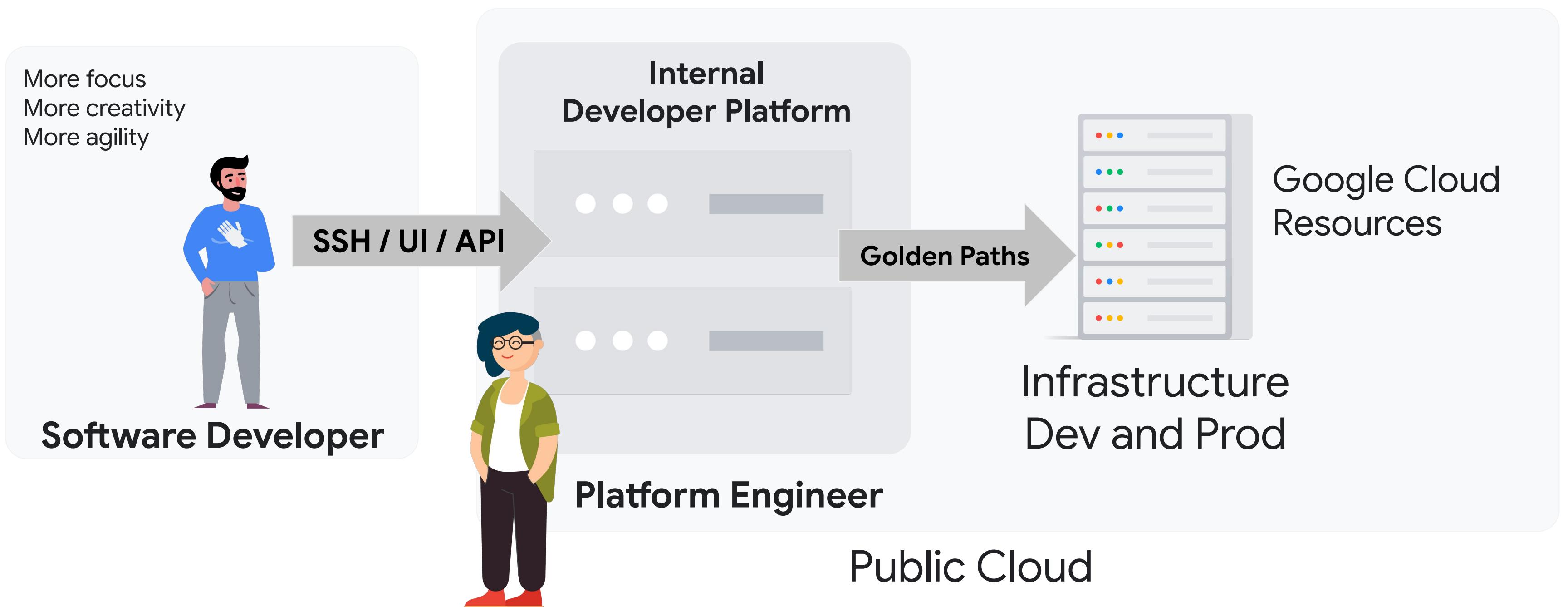


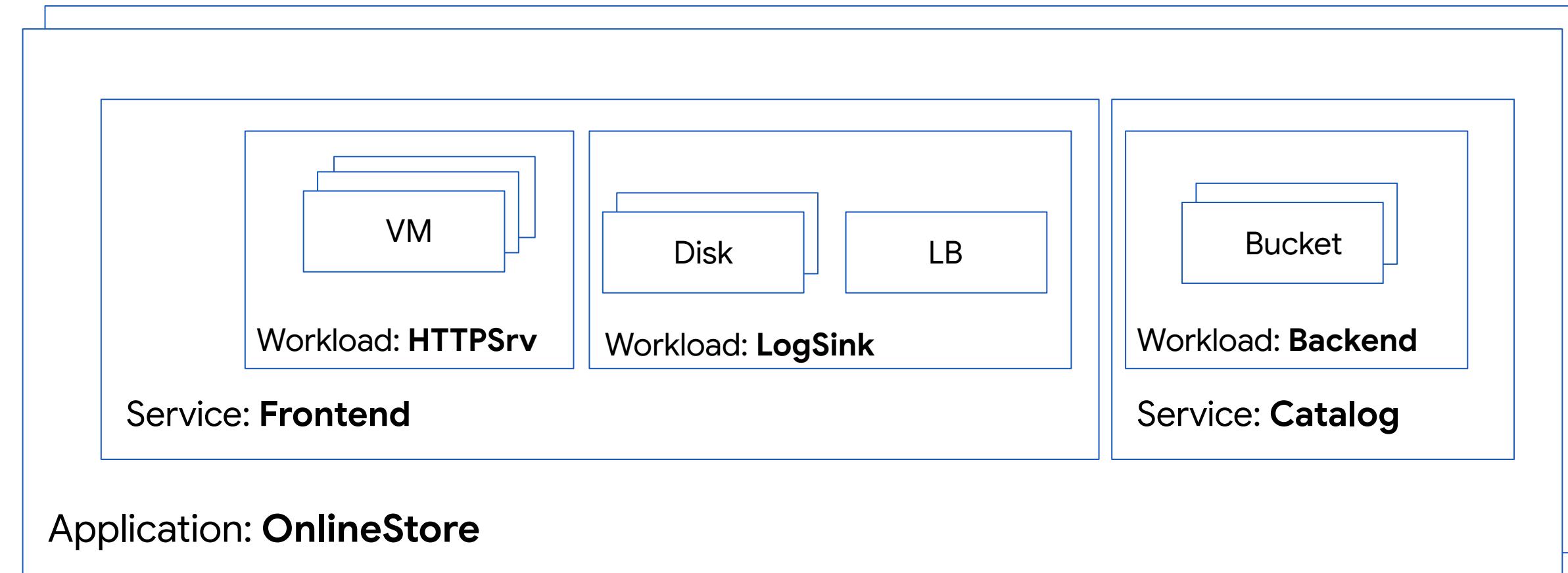
Google Cloud
Resources

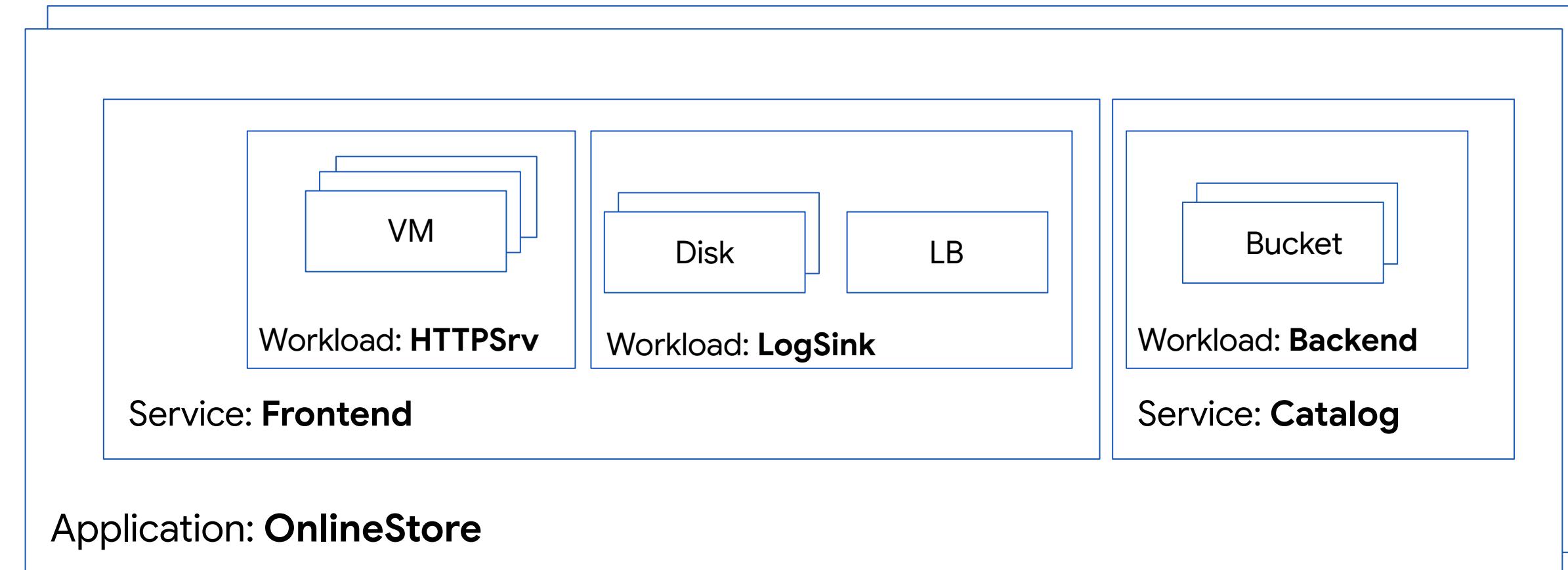
Infrastructure
Dev and Prod

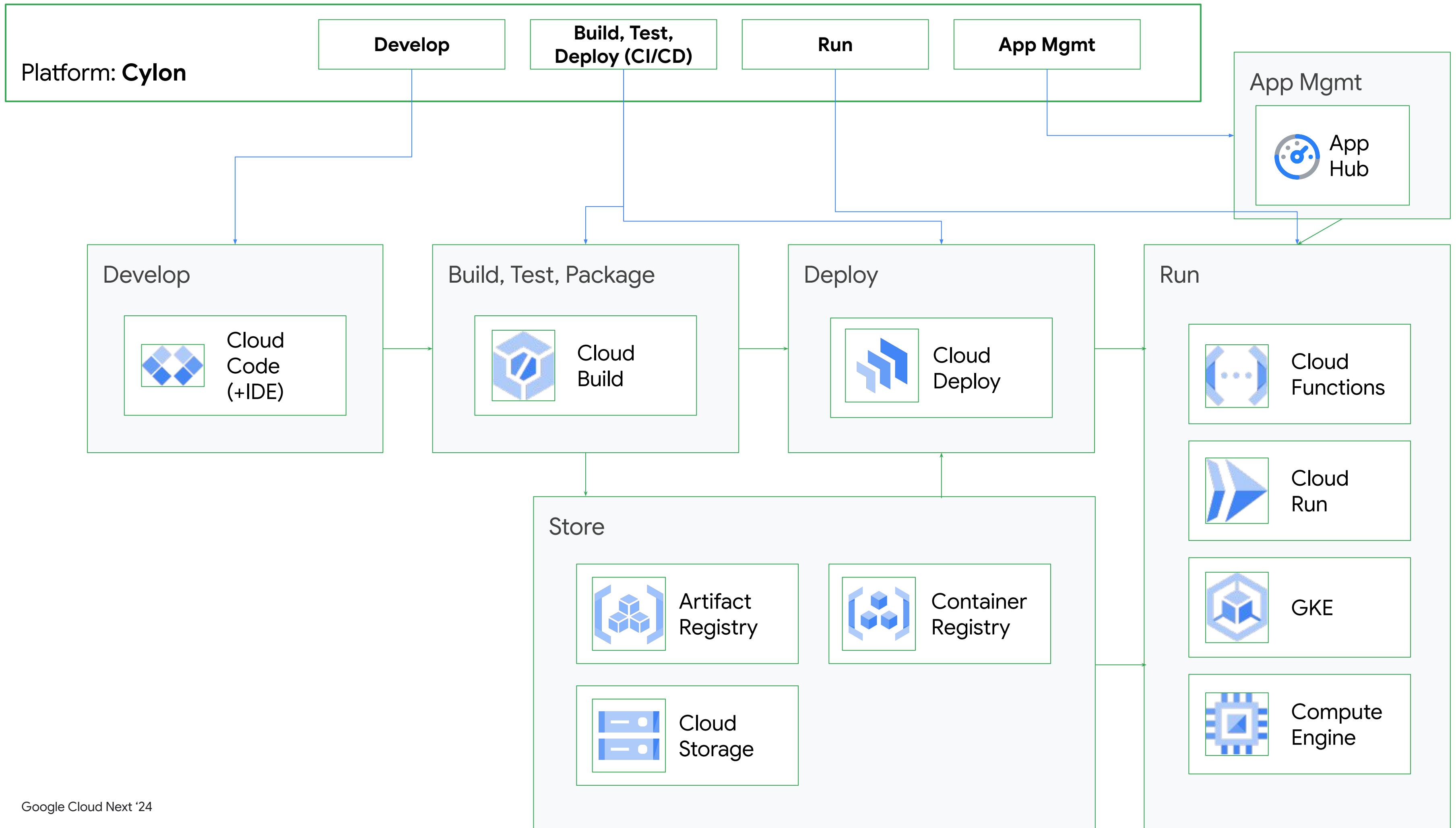
Public Cloud

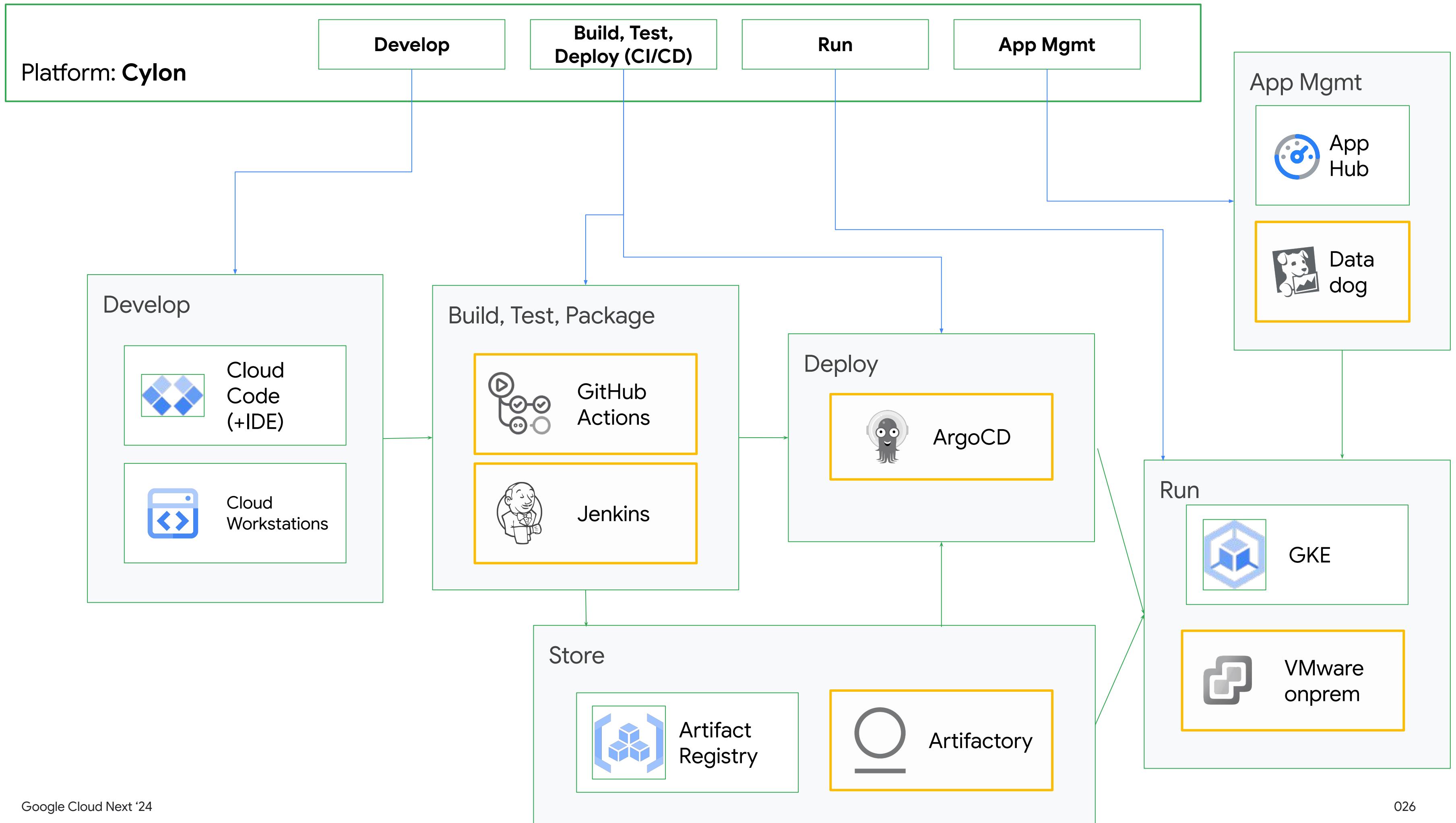
Platform Engineering













We've built an API-driven Platform known internally as the X Framework



Asset
transparency



Reduced
developer
cognitive load

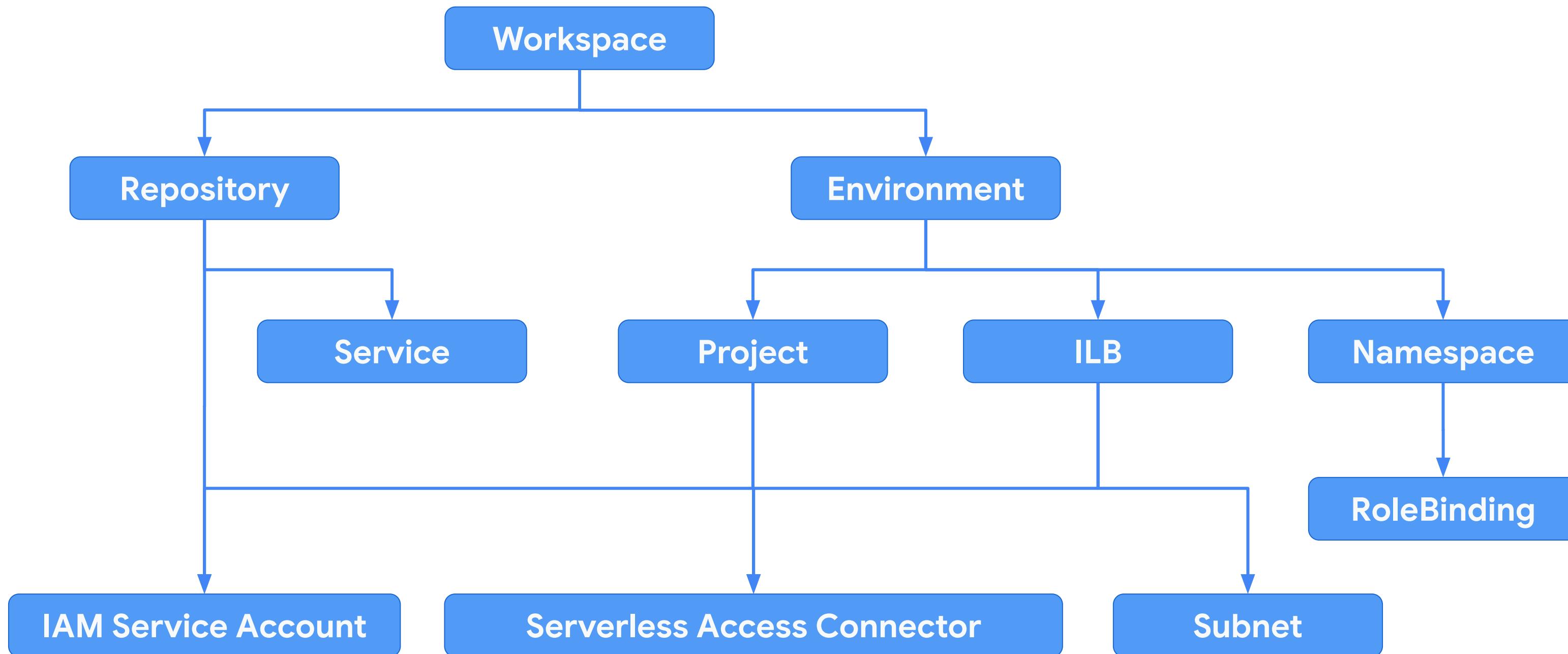


Reduced
operator toil



Community
Governance

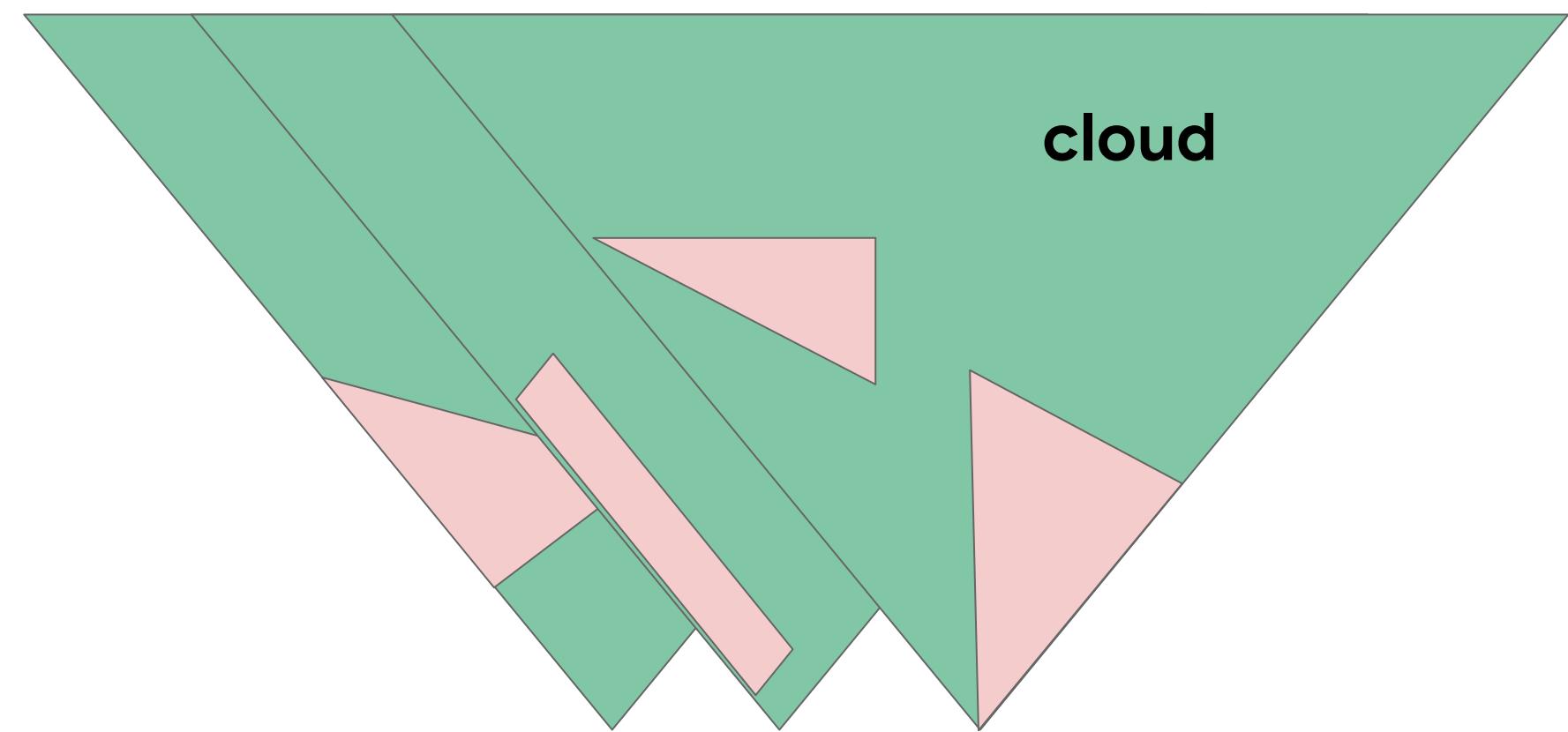
X Framework Domain Model





Platforms enable reliable applications

- **Chaos Engineering**
 - You don't need an off the shelf tool
 - We simulated losing a zone in production
 - Use taints and tolerances move applications between zones
- Non-production clusters use **Spot VMs**
- Stateless **cloud-native** approaches





Platforms enable reliable applications

The screenshot shows the Backstage platform interface. The left sidebar contains navigation links: Home, My Squads, Catalog, APIs, Docs, Adv. Search, Glossary, Playlists, Reports, Tech Radar, Validator, and Add Shortcuts. The main content area has a pink header bar with the title "Google Kubernetes Engine (GKE)" and sub-information: Component (Google Kubernetes Engine (GKE)), Owner (platform-cloud-infra), and Lifecycle (production). Below the header is a search bar labeled "Search Google Kubernetes Engine (GKE) docs". The main content is titled "GKE Reliability Contract". It includes a table of contents on the right side with sections like "Replicate workloads", "Spread workloads", "Specify disruption budgets", "Recommendations", "Setup Probes for your workloads", "Startup Probe", "Liveness Probe", "Readiness Probe", "Use prescribed Persistent Disk/Storage Class", and "Specify Request and Limits for containers". The main text discusses the required configuration for workloads to receive expected service levels and lists notes and bullet points about workloads being restarted or shuffled across nodes.

- Publish a **Reliability Contract**
- Embrace **SRE Principles**
- Keep **competitive tension** between infrastructure **reliability**, **cost** and developer **productivity**

<https://sre.google/workbook/slo-document/>

<https://sre.google/workbook/error-budget-policy/>



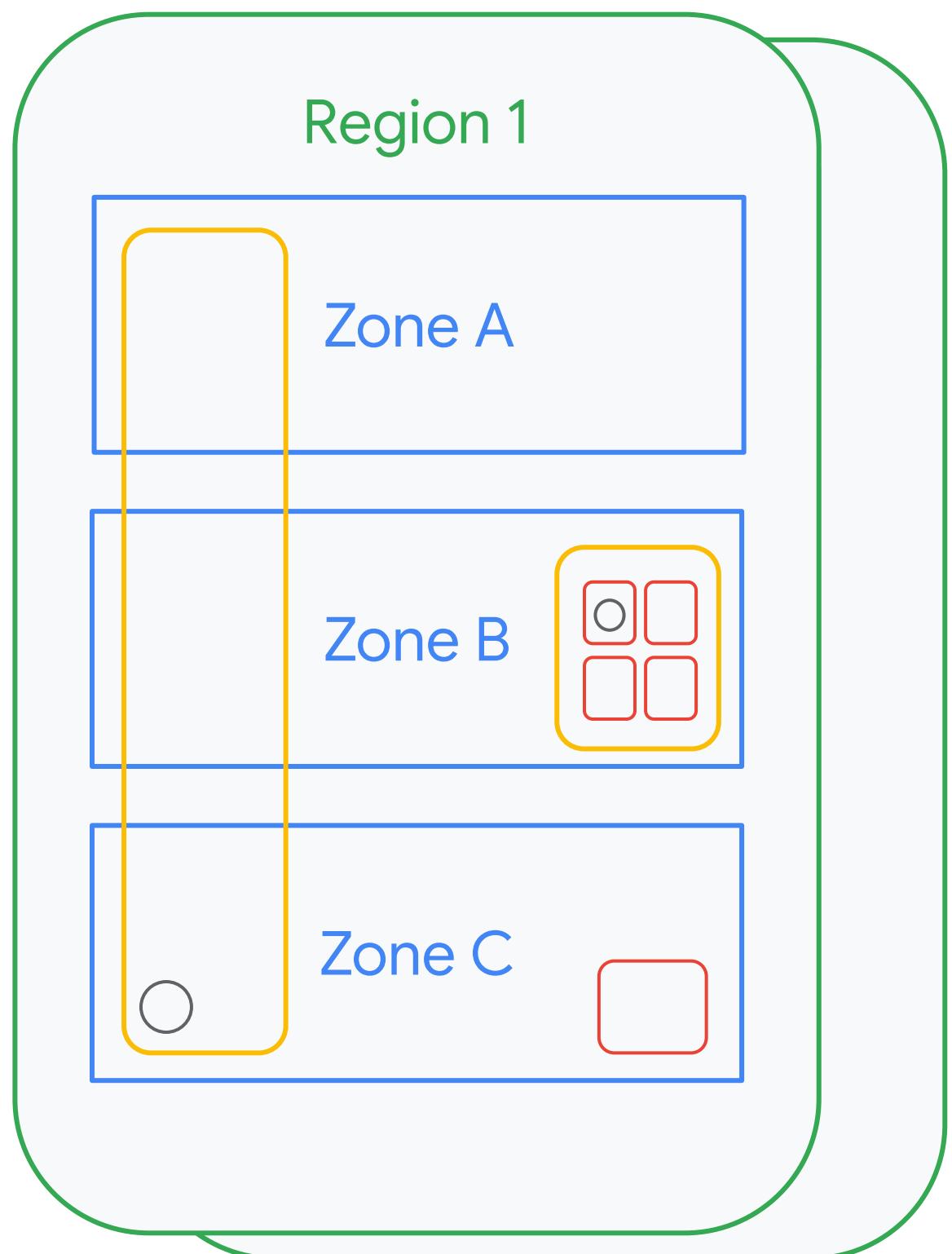
Platforms enable reliable applications

- Only a **single region** in Australia available at launch, second came later.
- Currently finalizing **dual region** design
- Only most critical services will be deployed **active-active**
- We have to **rewrite** some things.
- Extreme reliability is **hard**

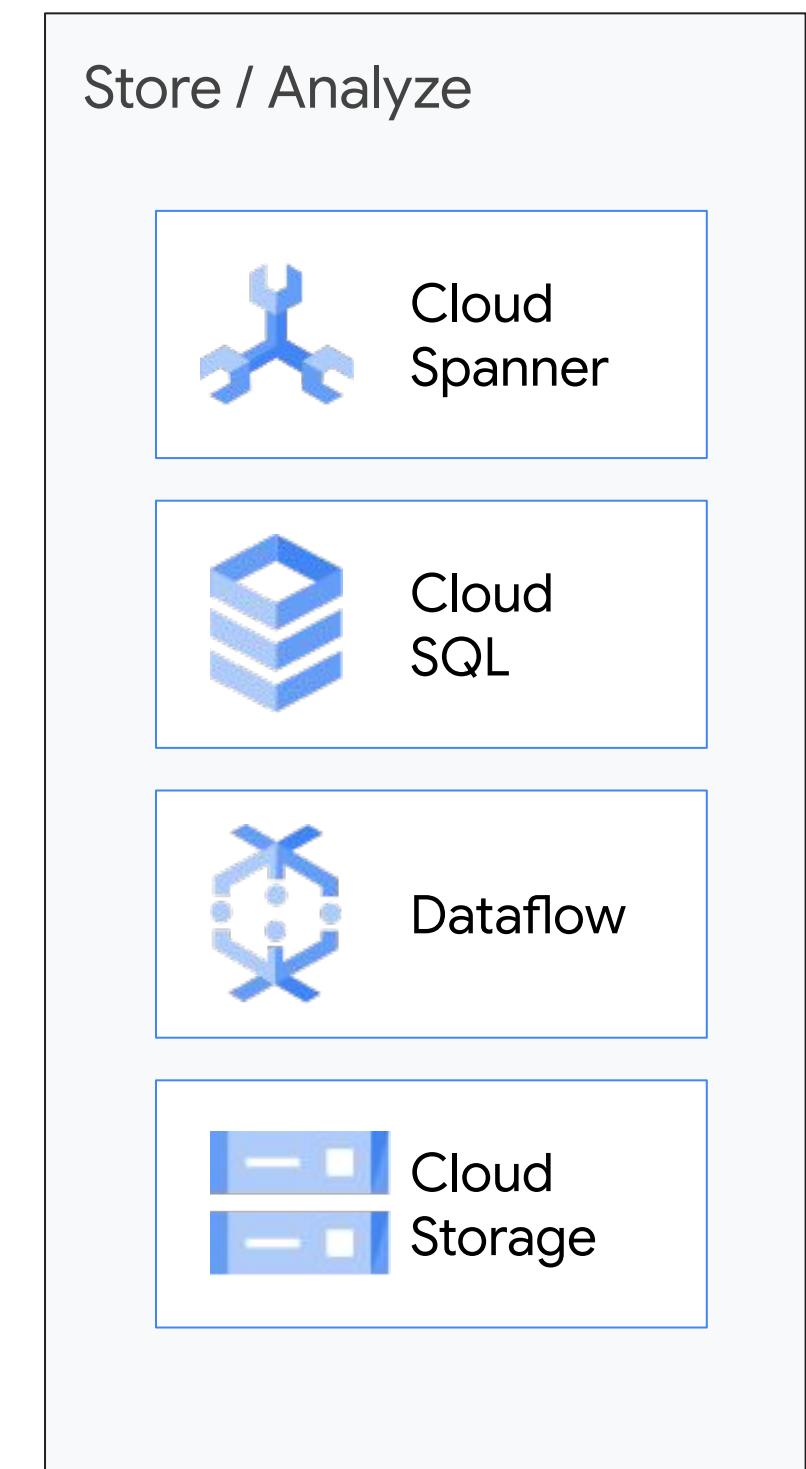
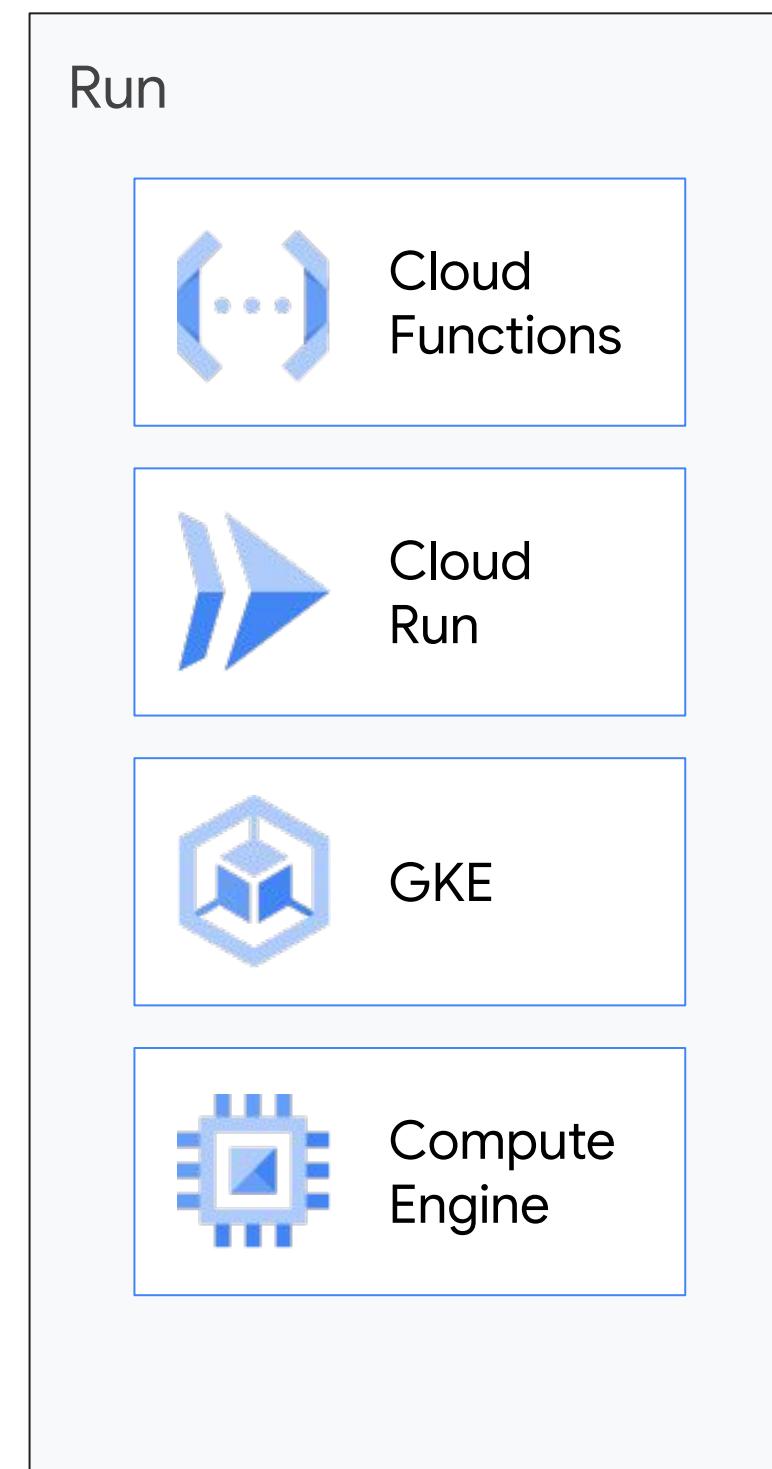


Cloud Failure Domains: Regions, Zones

Clusters, Pods, Apps, Functions, VMs



Products expose their failure domains differently

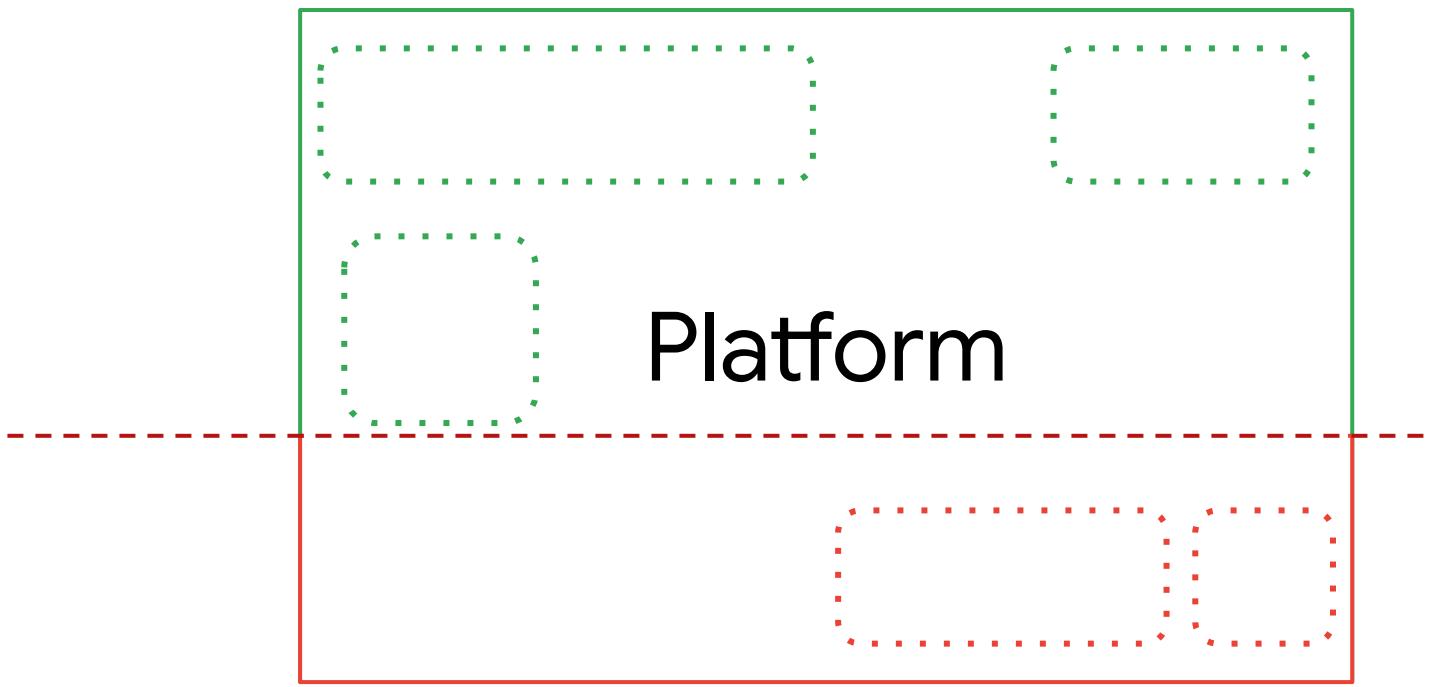


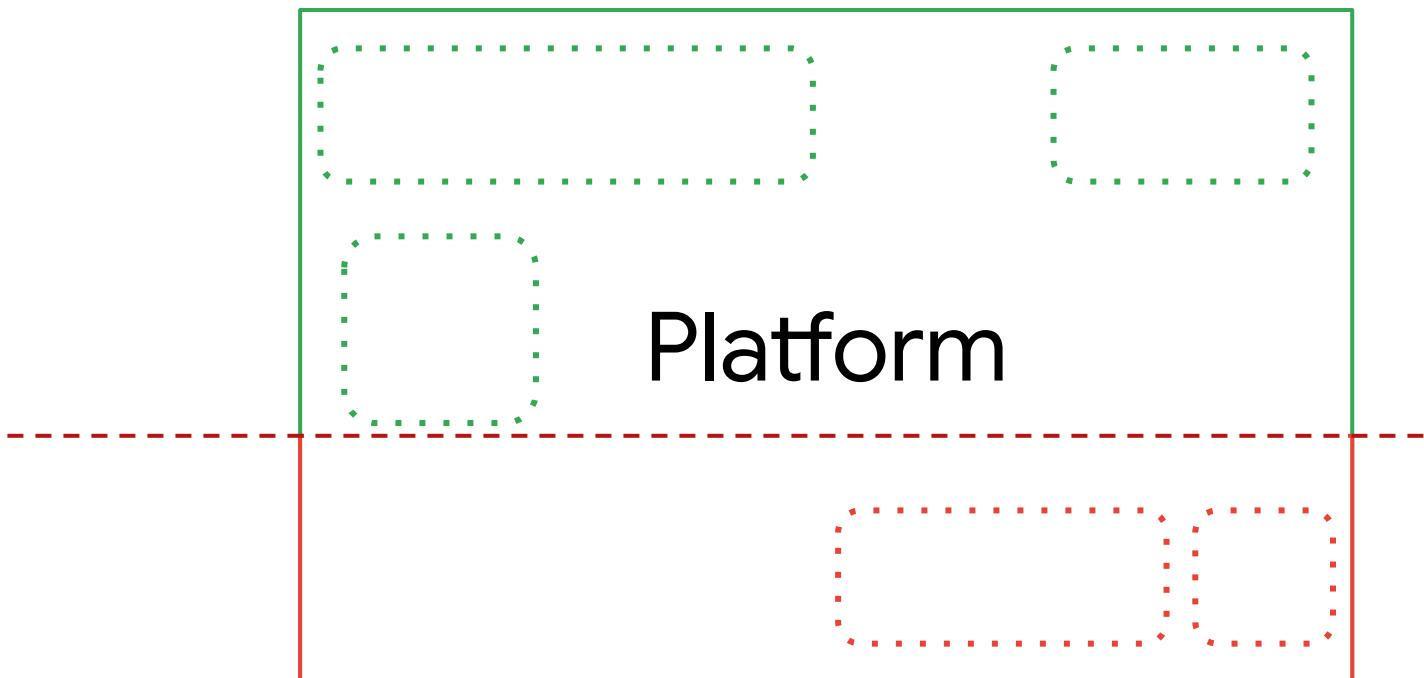
Platform

Product (Internal)

Applications

Product (External)





- **Critical** platform capabilities deployed as active-active, dual region
- **Observability** stack also critical
- **Feature toggling** is critical - for outage tiles and disabling UI components

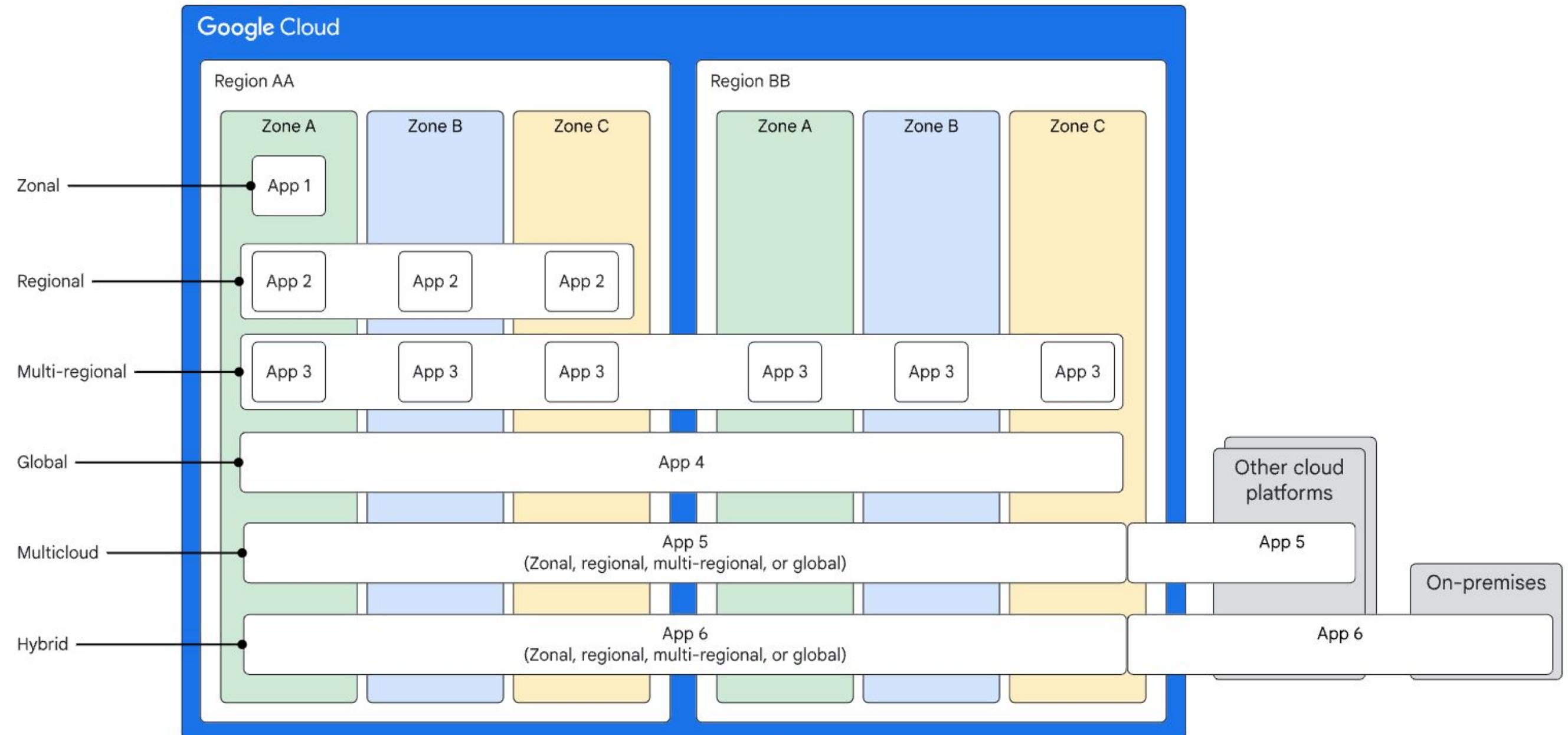
- **CI/CD** not dual-region initially
- **Manual failover** using **break-glass** procedures
- Dual region implementation may come **later**

5 Application Archetypes

5 Application Archetypes

goo.gle/app-archetypes

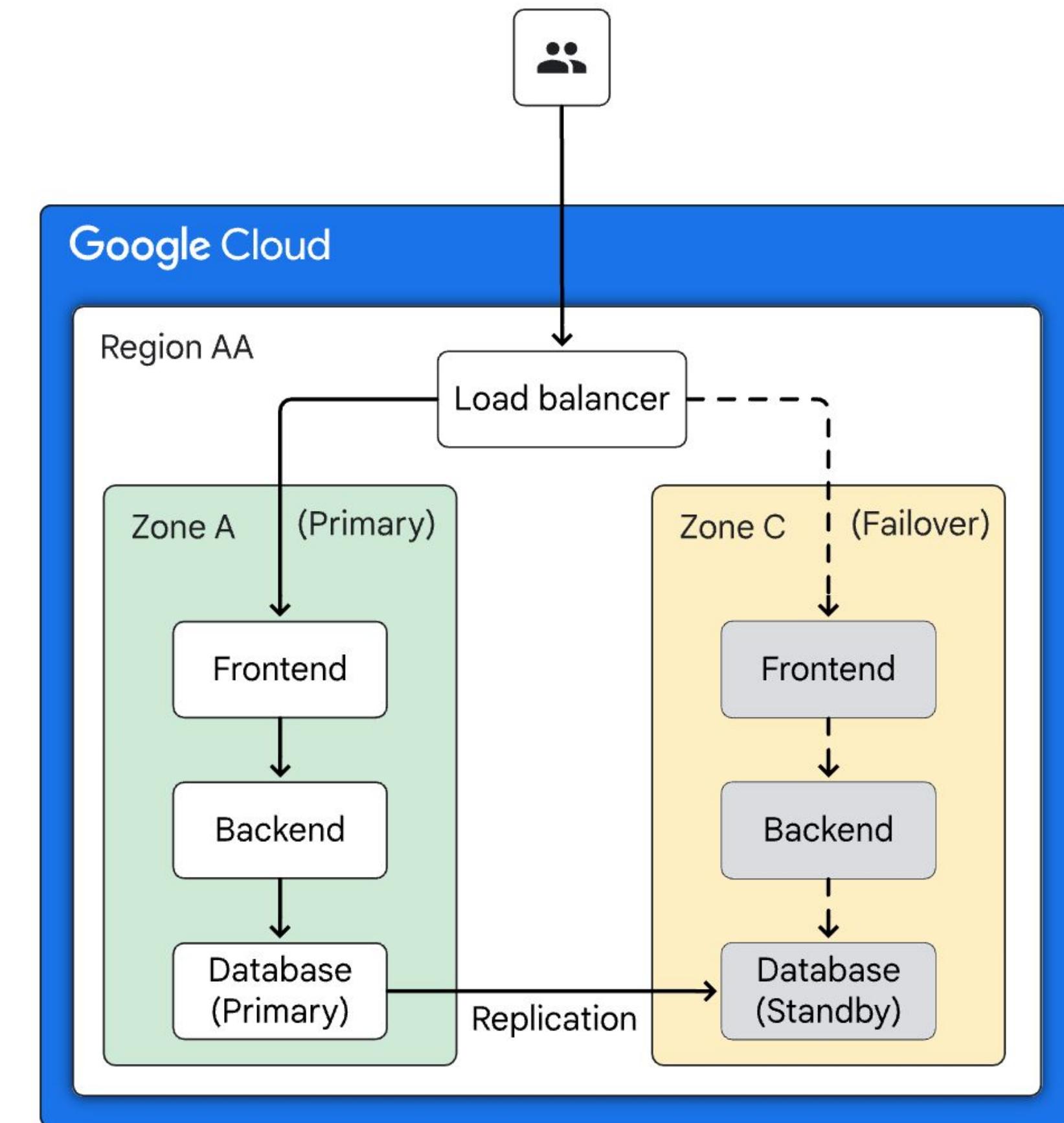
Cloud Architecture Center



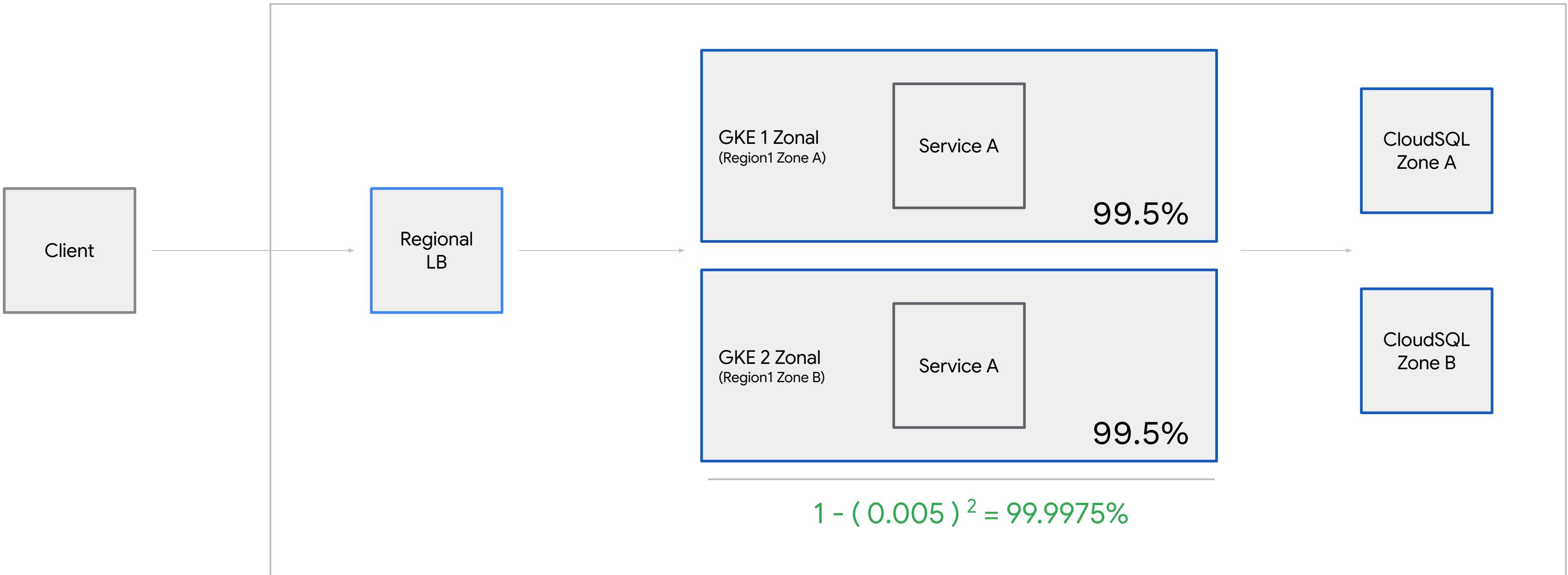
Archetype 1

Active Passive Zones

- **Survives zone failure.**
- **Fail-Ops:** Change LB backend, promote read replica
- **Cost:** 2x serving + 2x data (1 replica)
- **Complexity:** Low
- **App Refactoring:** None (lift and shift)
- **Type:** COTS, licensing



GKE clusters with Cloud SQL HA

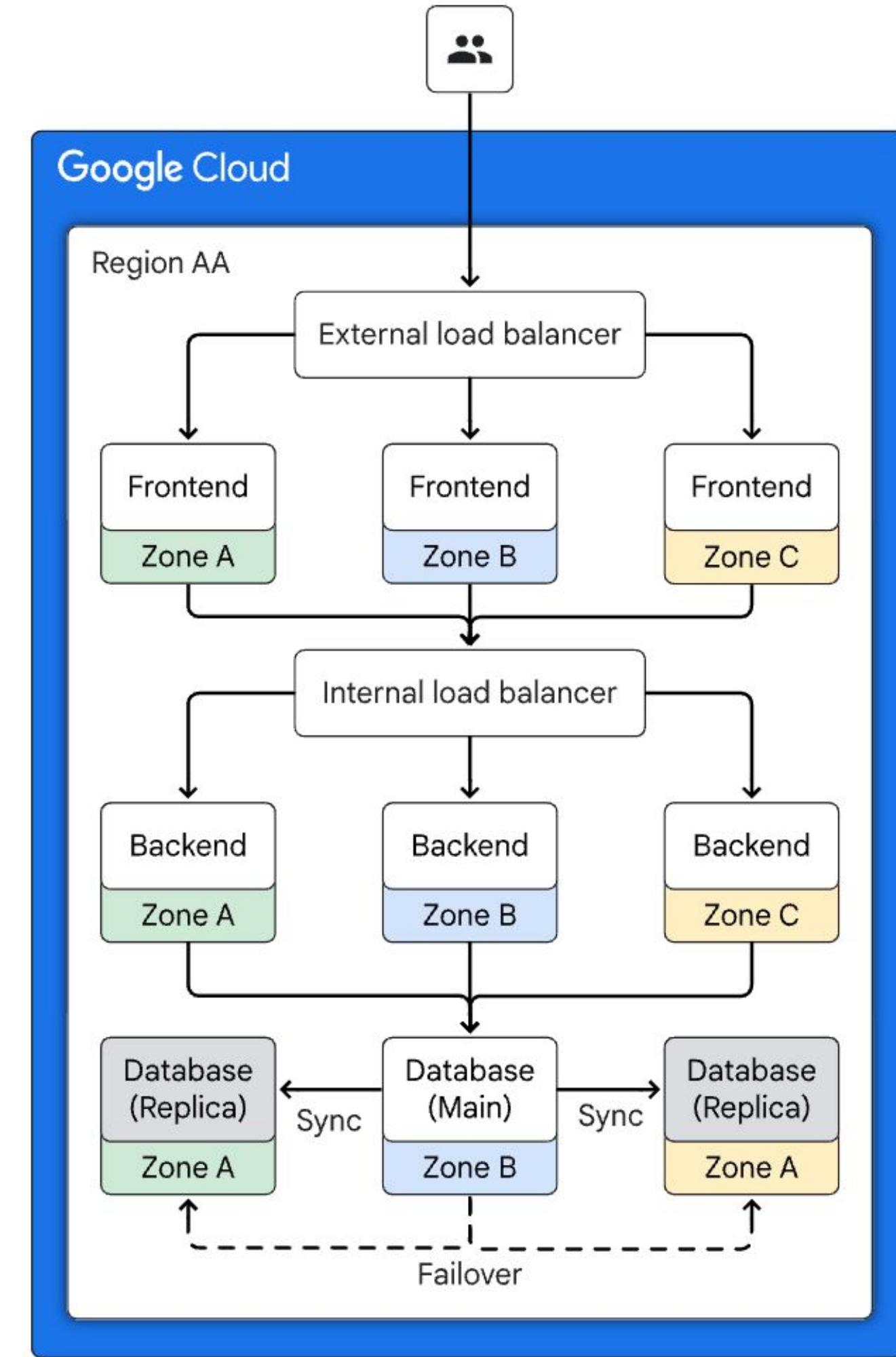


**99.98% SLO
<2h / year**

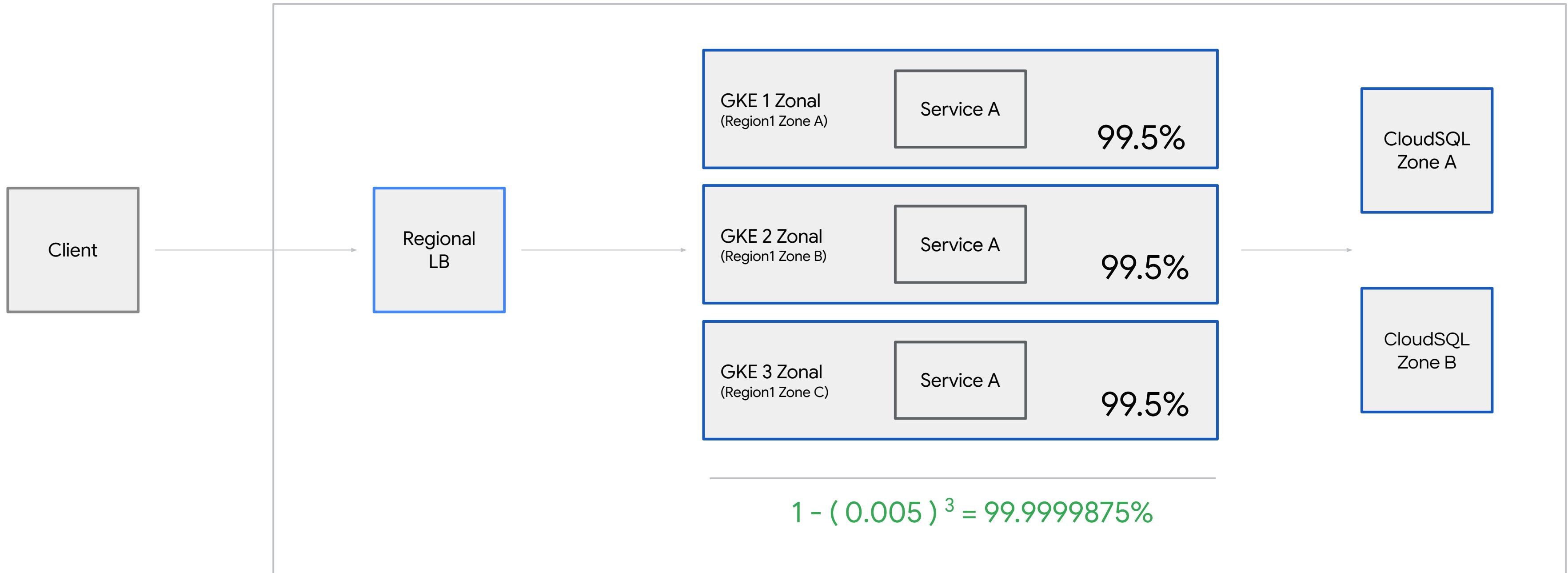
[GCP SLAs](#)

Archetype 2 Multi Zonal

- **Survives zone failure.**
Does not survive region failure.
- **Fail-Ops:** Initiate DB failover
- **Cost:** 1.5x serving + 3x data (HA SQL)
- **Complexity:** Medium
- **App Refactoring:** Low (multi instance)
- **Type:** Web services



GKE clusters with Cloud SQL HA



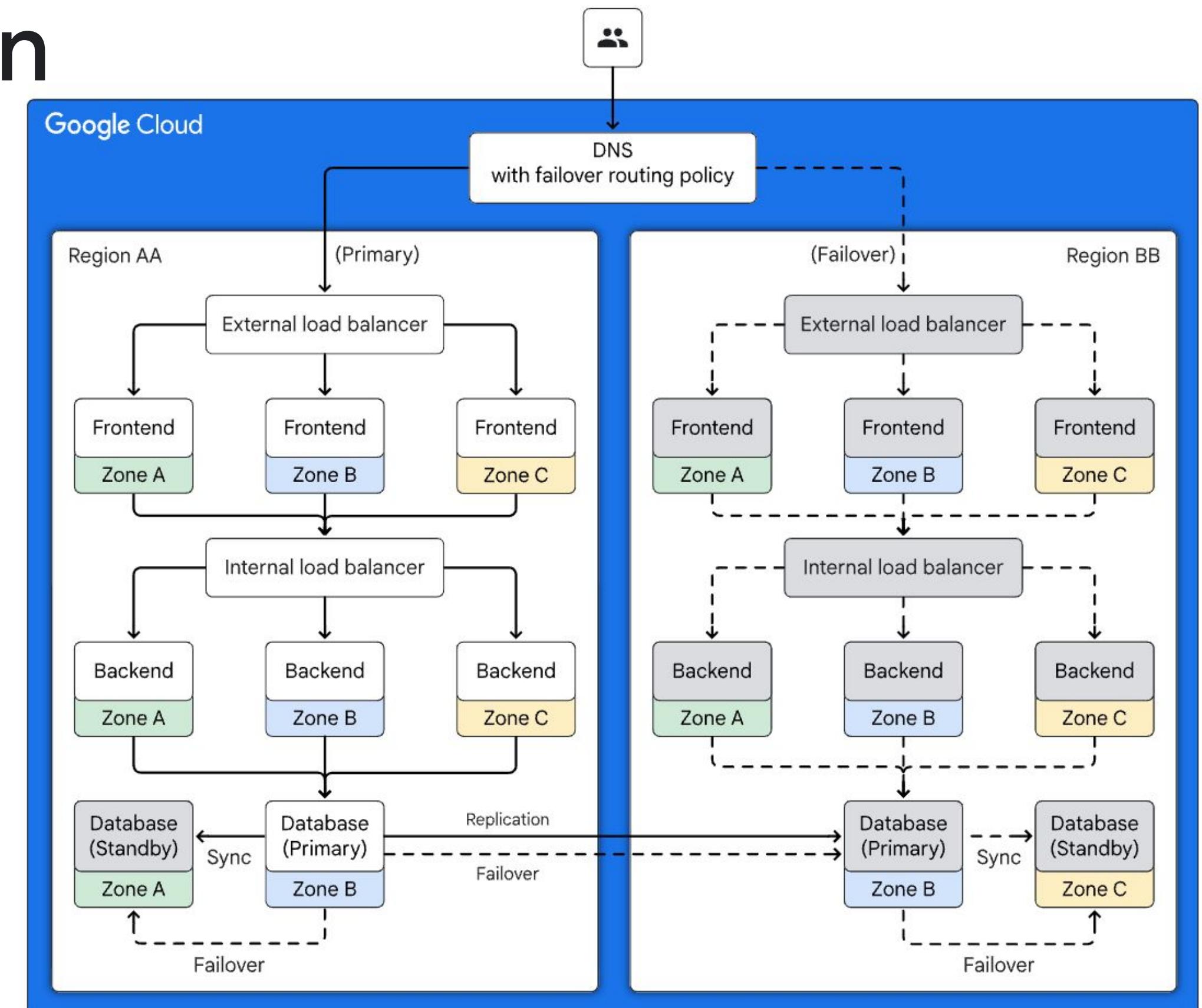
99.99% SLO

[GCP SLAs](#)

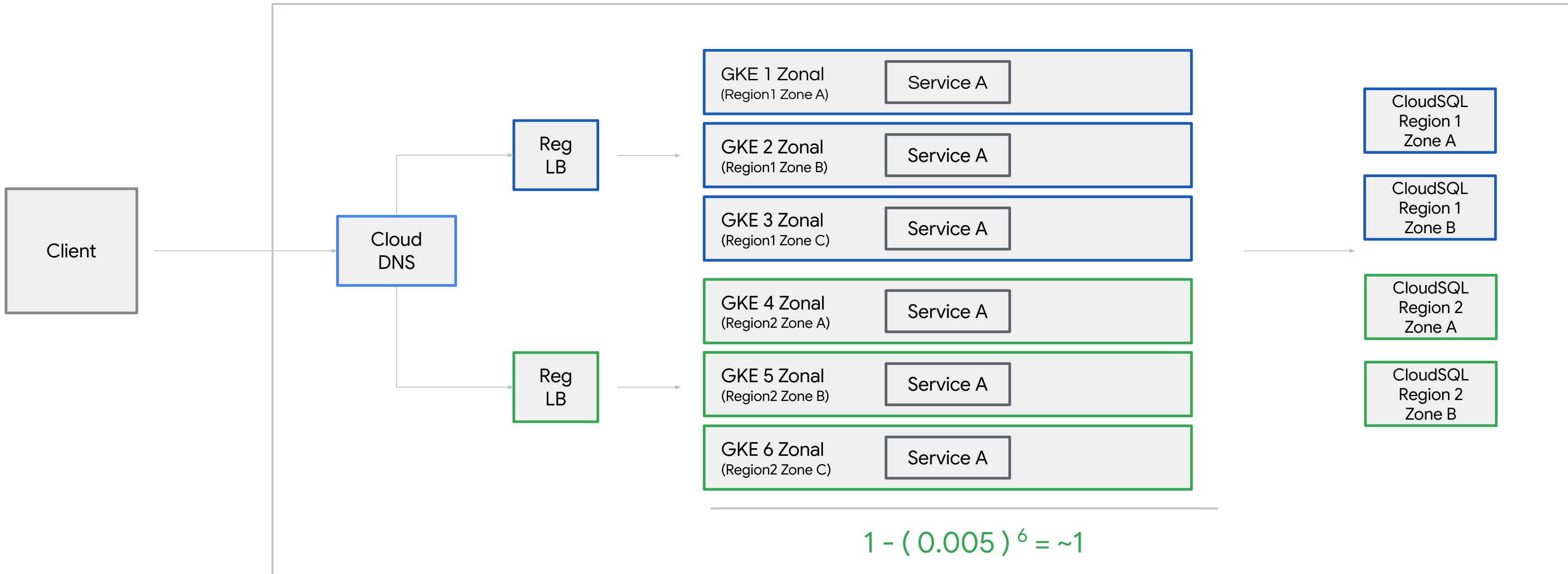
Archetype 3

Active Passive Region

- **Survives zone and region failures**
- **Fail-Ops:** No action for zone failure.
 - Update DNS to point at standby LB
 - DB: Cross region DR failover process
- **Cost:** 3x serving + 6x data (HA SQL)
- **Complexity:** Medium
- **App Refactoring:** Medium (multi instance, multi regional data)
- **Type:** HA web services



GKE clusters in multiple regions with Cloud SQL HA

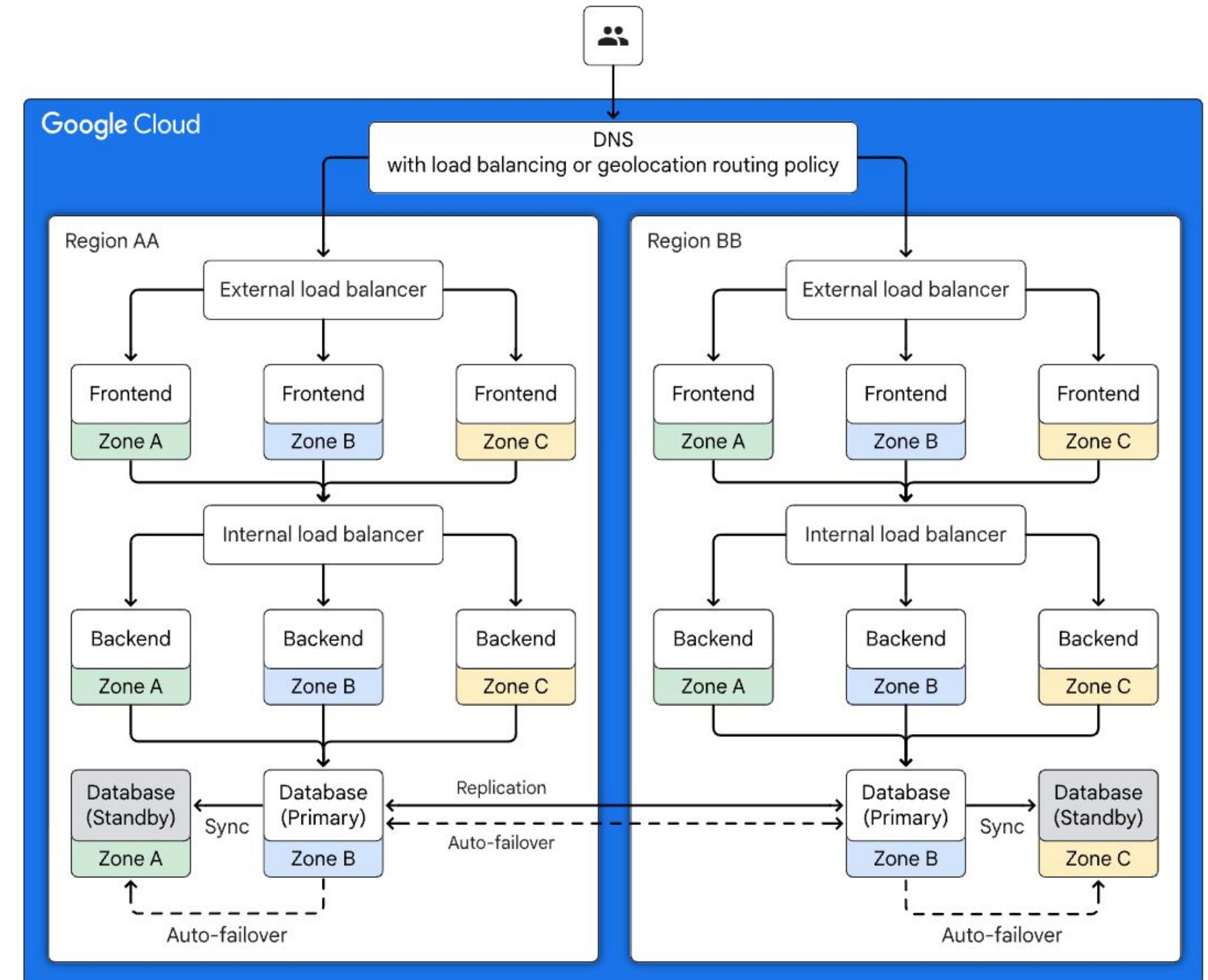


99.999999% SLO

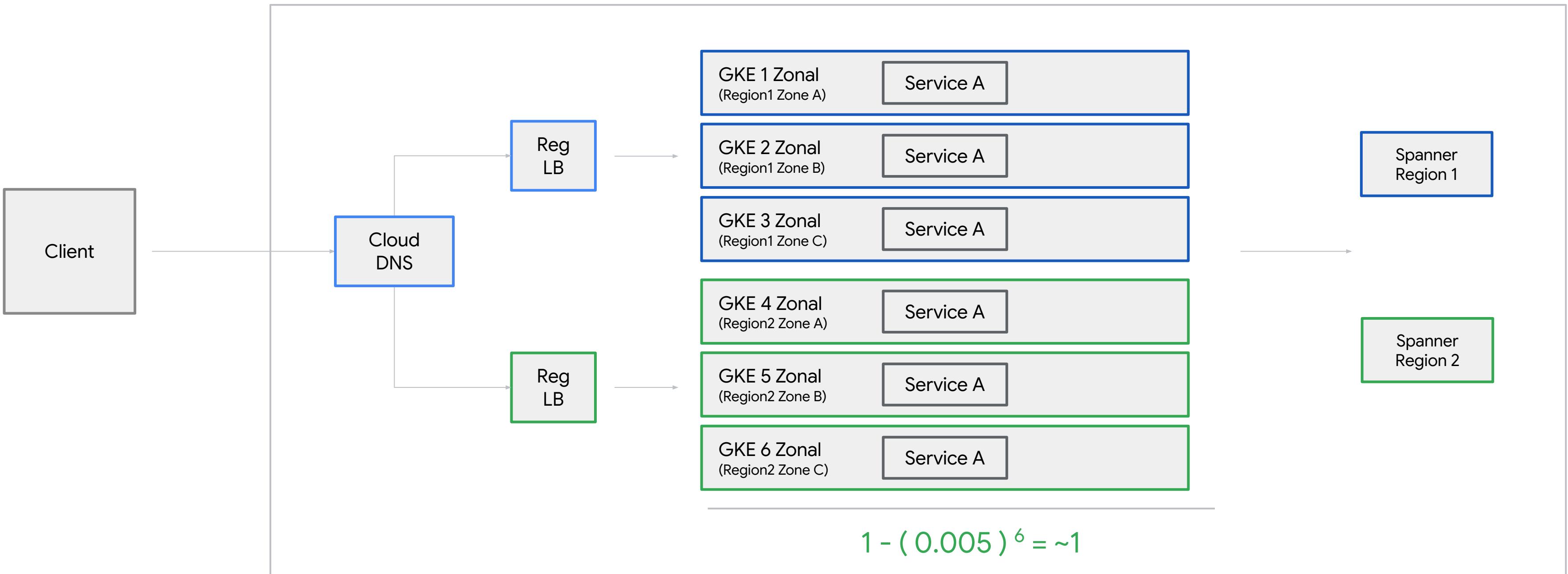
[GCP SLAs](#)

Archetype 4 Isolated Regions

- **Survives zone and region failures.**
No impact for $\frac{1}{2}$ consumers. Possible manual failover
- **Fail-Ops:** No action for zone failure.
Optional regional failover like Arch 3.2
- **Cost:** 1.5 cost per region for zone failure
- **Complexity:** Medium/High
- **App Refactoring:** Medium (multi instance, multi regional data)
- **Type:** Regulated HA services



GKE clusters with Cloud Spanner

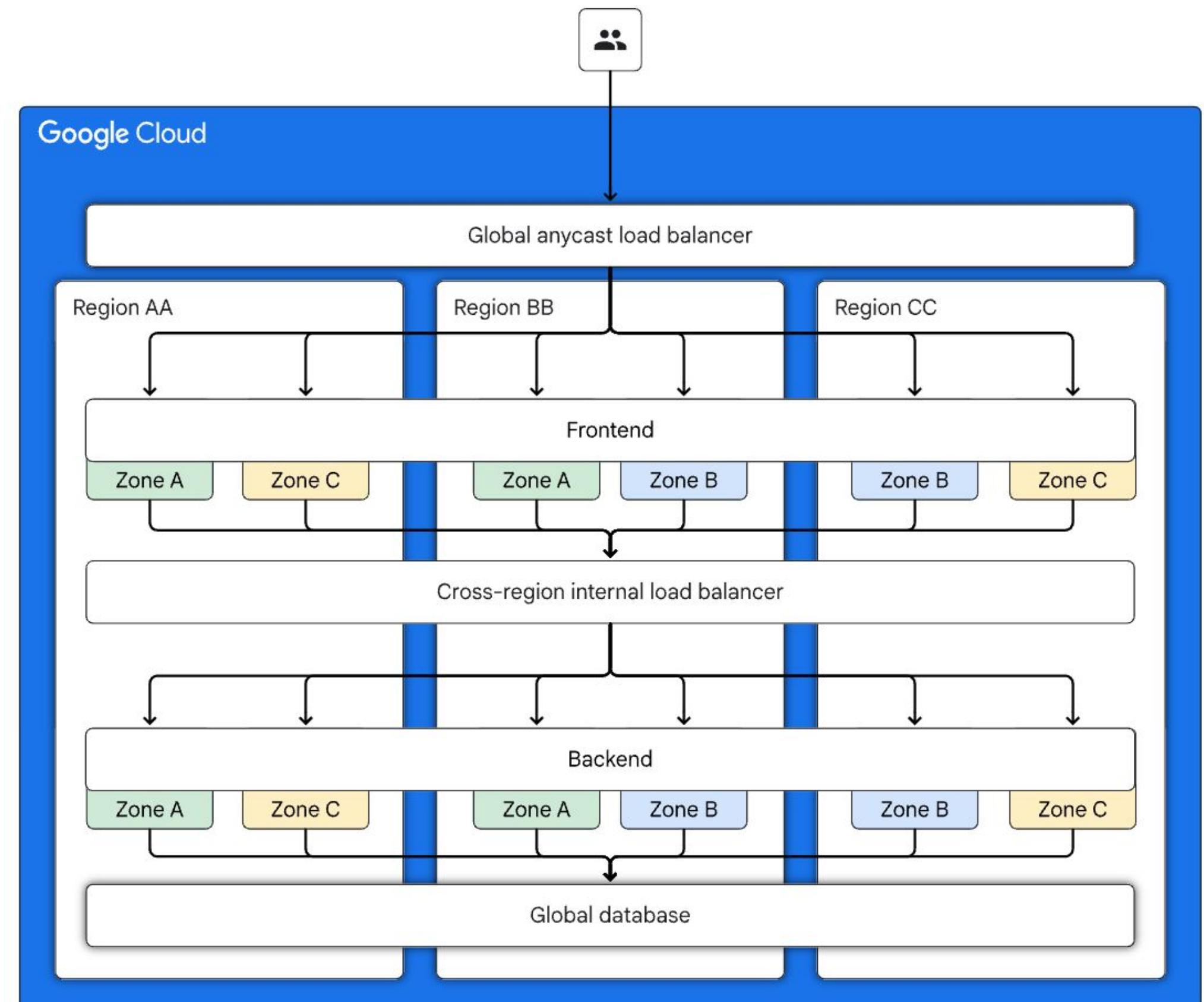


99.999998% SLO

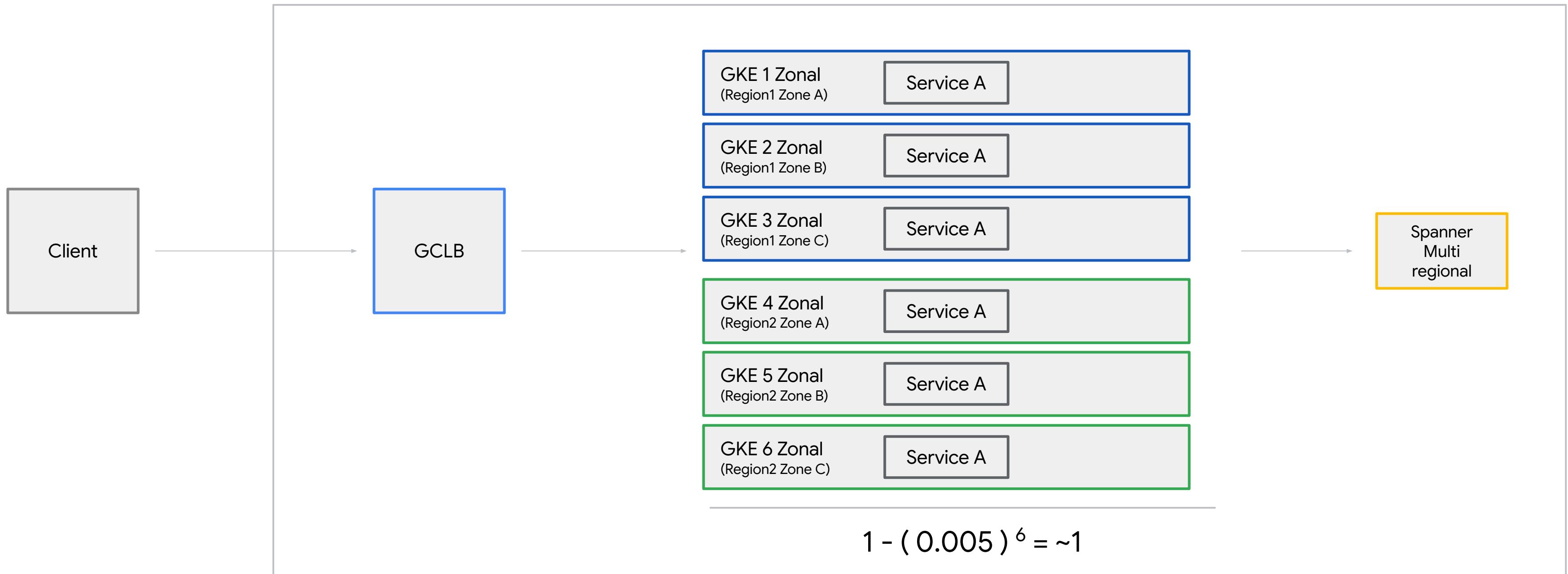
[GCP SLAs](#)

Archetype 5 Global

- **Survives zone and region failures**
- **Fail-Ops:** None
- **Cost:** N+m cost modelling. Global DBs are more expensive
- **Complexity:** High
- **App Refactoring:** High (multi instance, global DBs)
- **Type:** Global consumer services

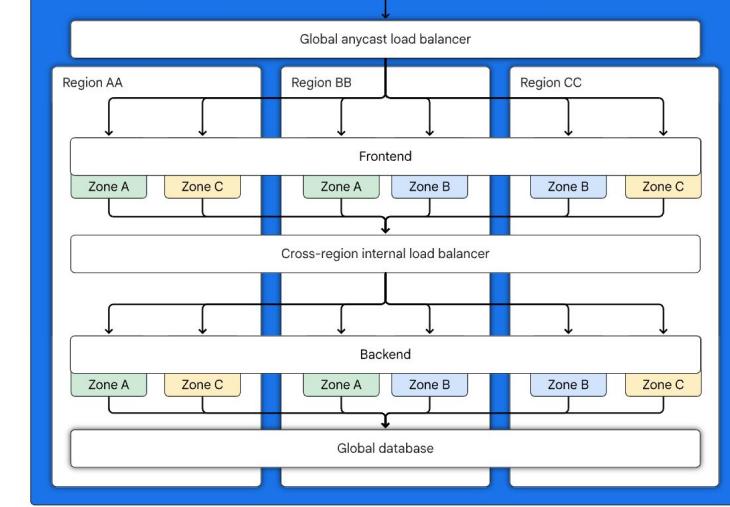
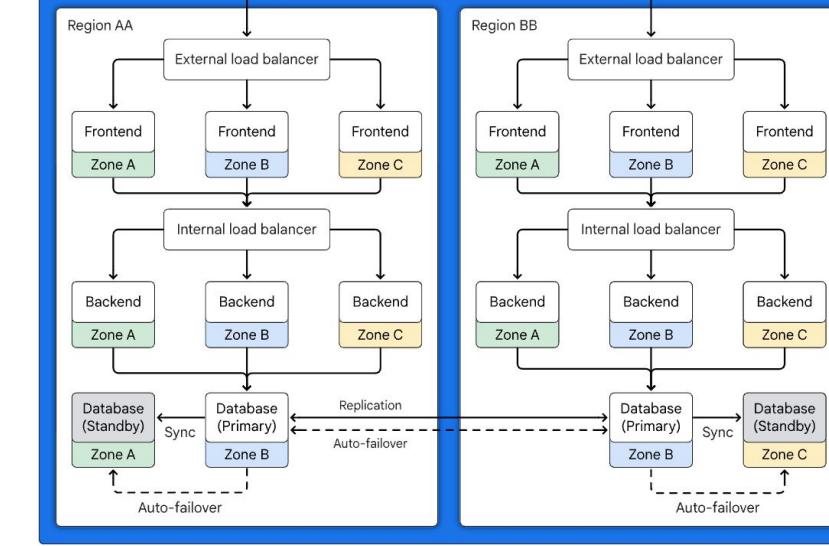
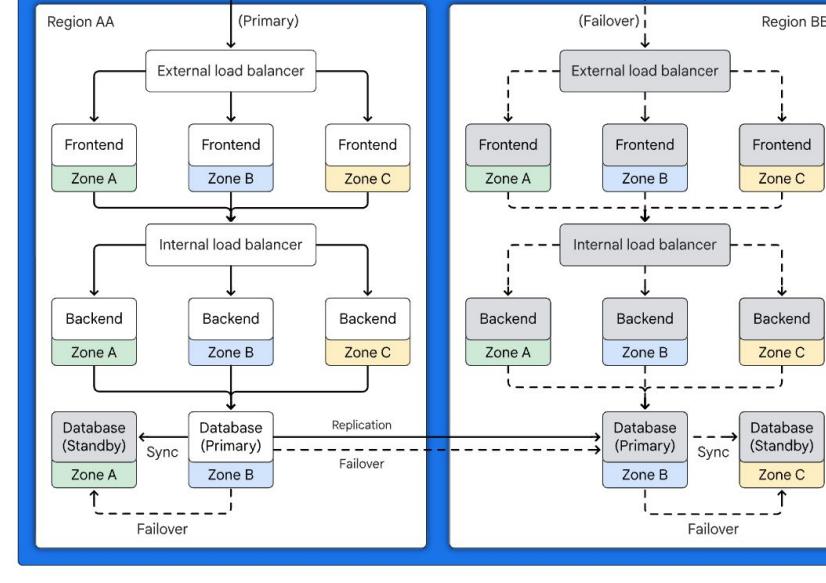
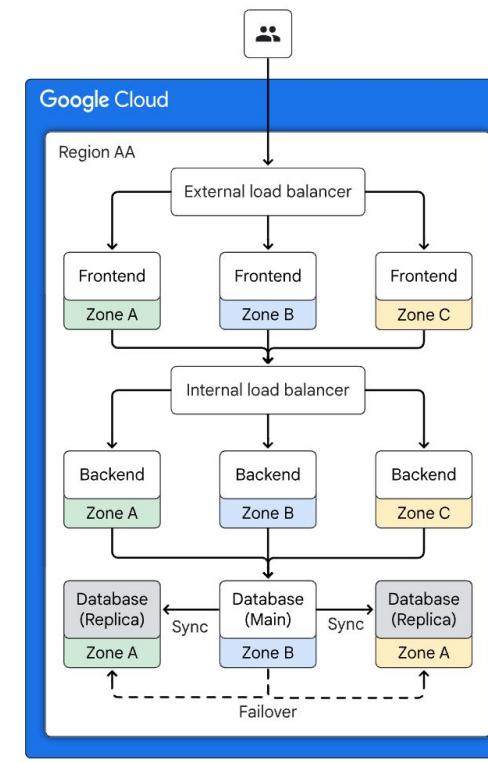
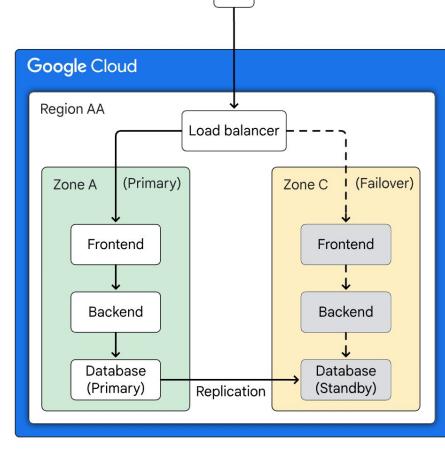


GKE clusters with Cloud Spanner (Multi regional)



99.99% SLO

[GCP SLAs](#)



**Active
Passive
Zones**

SLO: 99.98

Multi-Zonal

SLO: 99.98

**Active Passive
Regions**

SLO: 99.999999

Isolated Regions

SLO: 99.999998

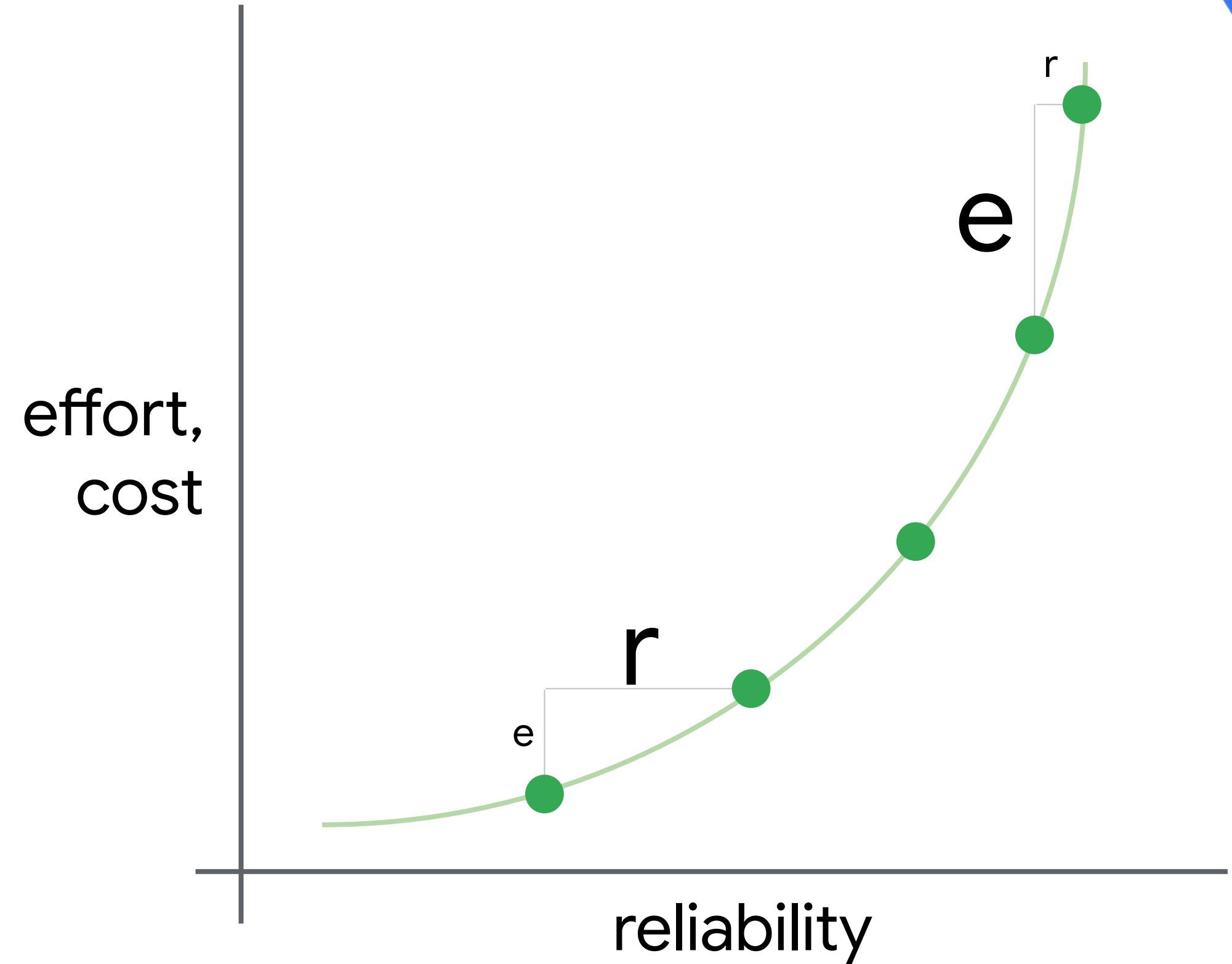
Global

SLO: 99.99



effort,
cost

reliability



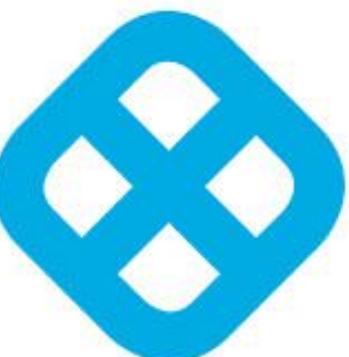


```
1 apiVersion: apps.x.anz/v1beta1
2 kind: CloudRun
3 metadata:
4   isPublic: false
5   name: xxxxx-registration
6   workspace: xxxxx
7   deploymentArchetype: DualRegion
8 pipeline:
9   gha: {}
10  monolith: {}
11  rolloutStrategy: bluegreen
12 spec:
13   application:
14     image:
15       repository: australia-southeast1-docker.pkg.de
16   livenessProbe:
17     failureThreshold: 3
18     httpGet:
19       path: /healthz
20       port: 8080
21     initialDelaySeconds: 30
22     periodSeconds: 30
23     timeoutSeconds: 10
24   ports:
25     - containerPort: 8080
26       name: h2c
27
```

Deployment Archetype can be configured on runtime CRDs



Google Cloud



N O B L⁹



GitHub Actions



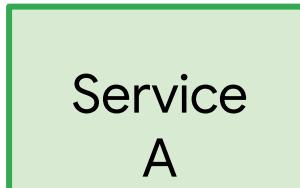
How to use Archetypes?

Services can be
deployed to a single
archetype

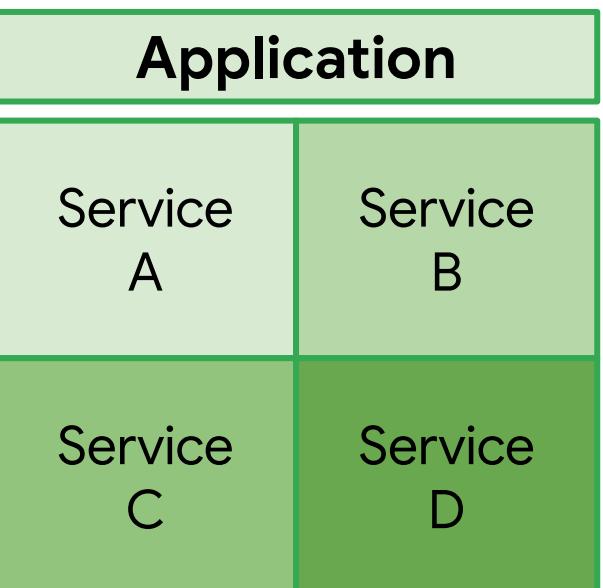


How to use Archetypes?

Services can be deployed to a **single archetype**



Applications can use services across **multiple archetypes**

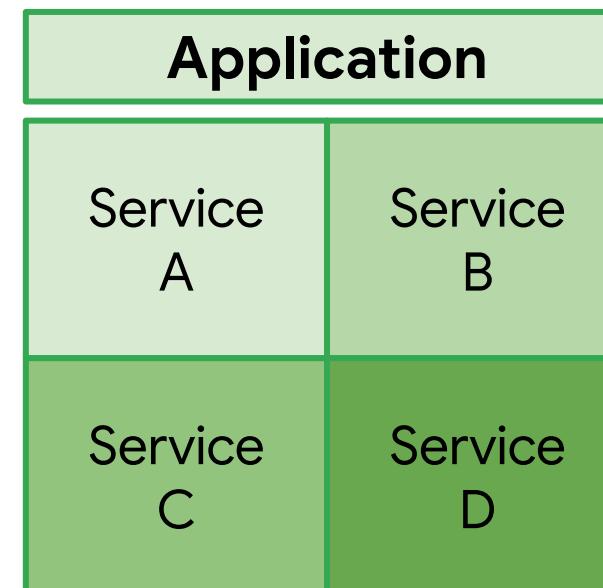


How to use Archetypes?

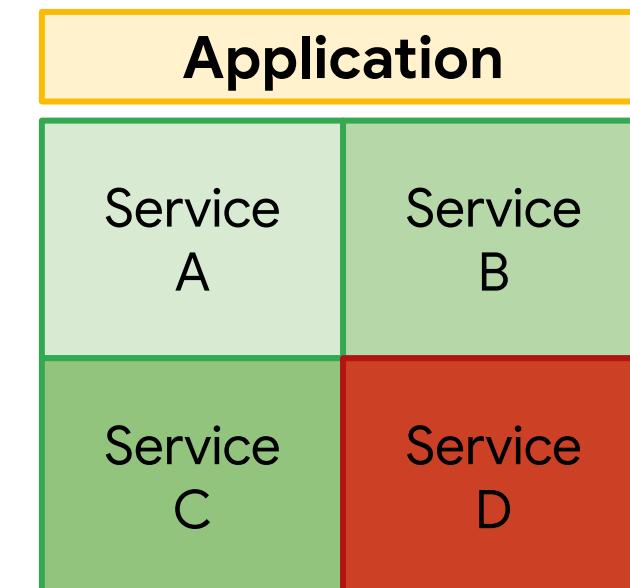
Services can be deployed to a **single archetype**

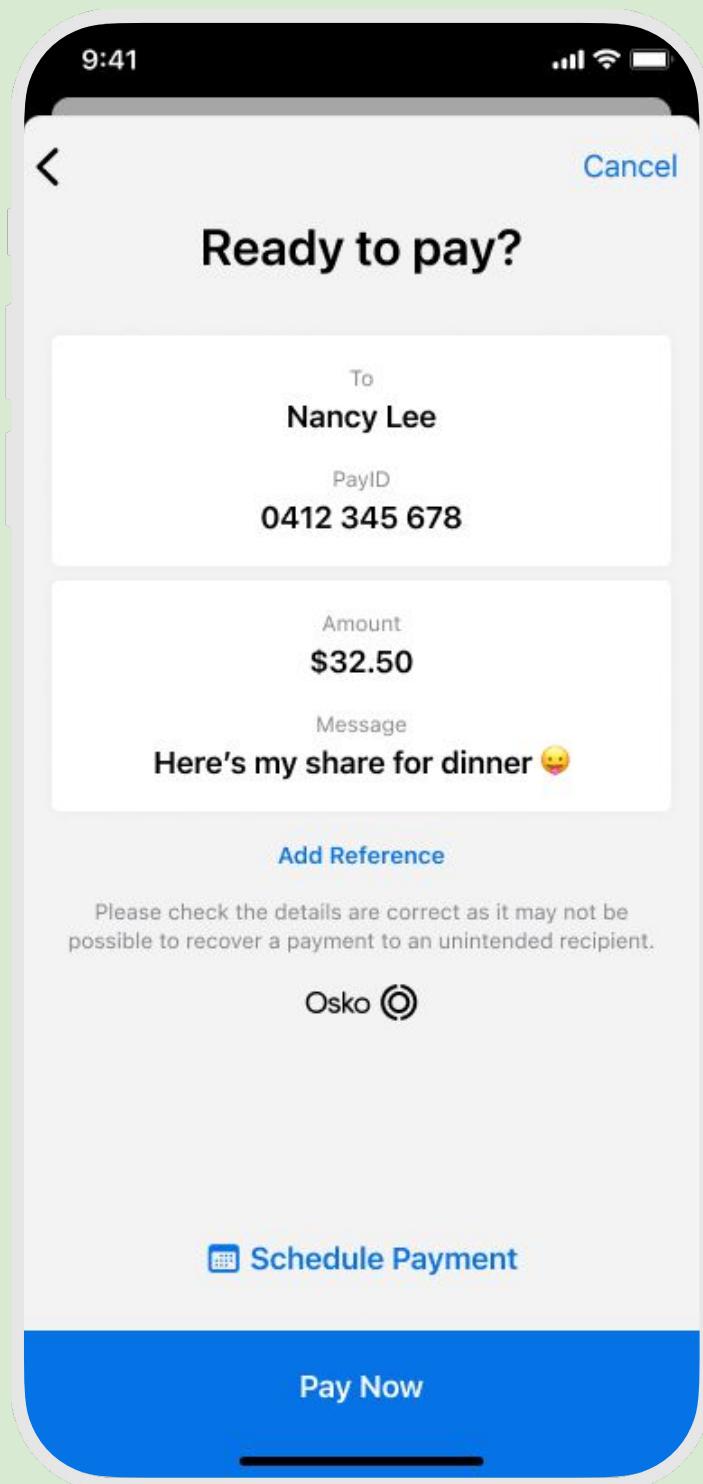
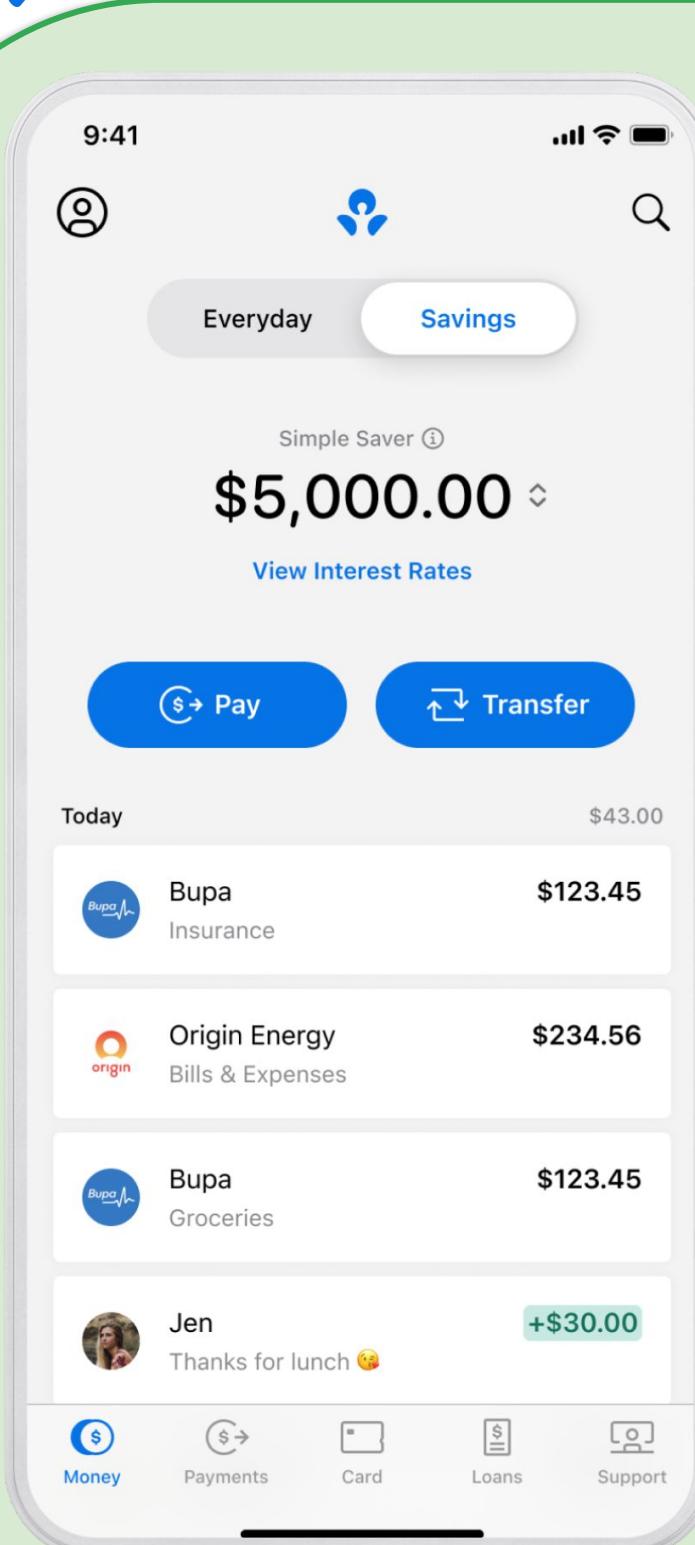


Applications can use services across **multiple archetypes**

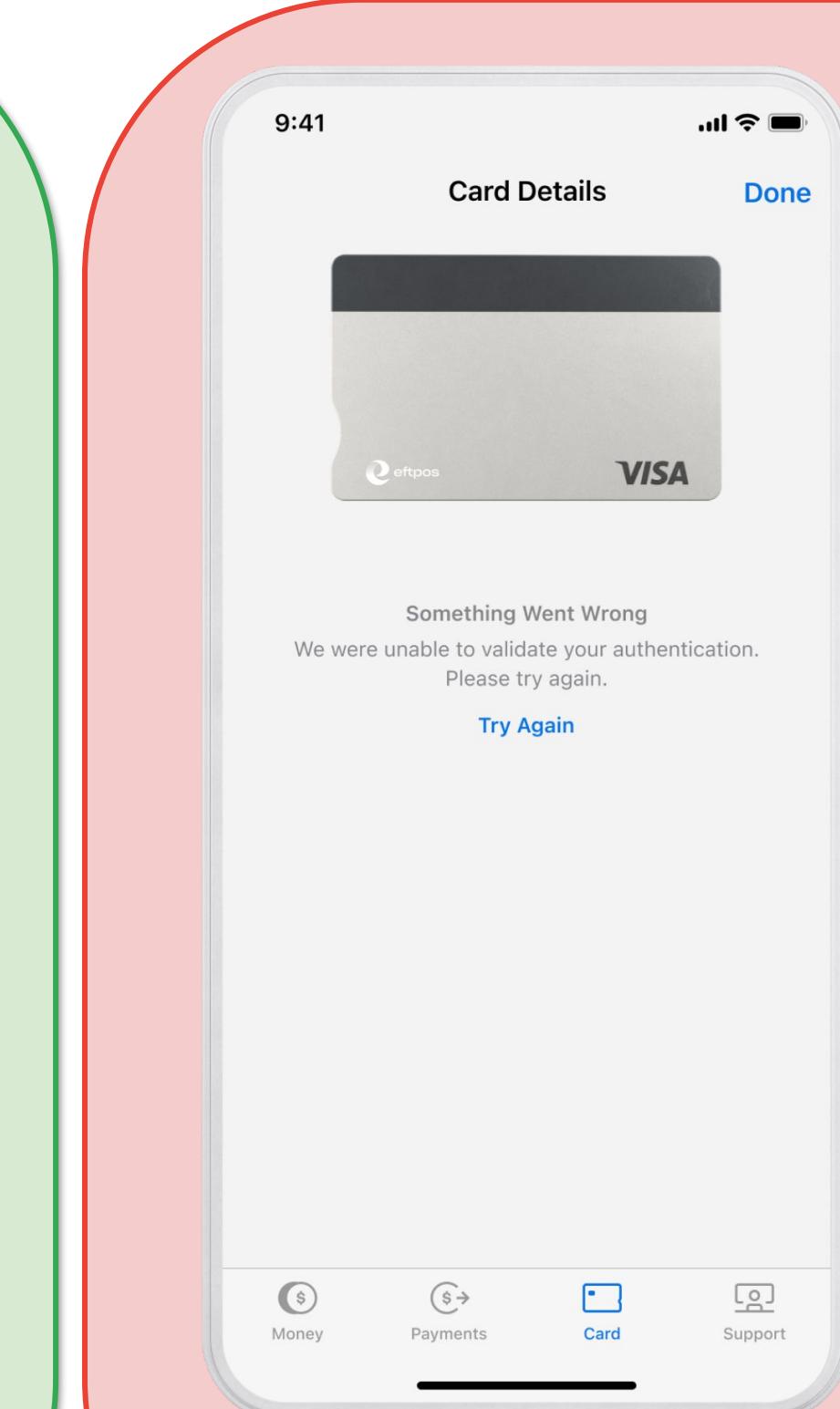


Applications should be designed for **graceful degradation**

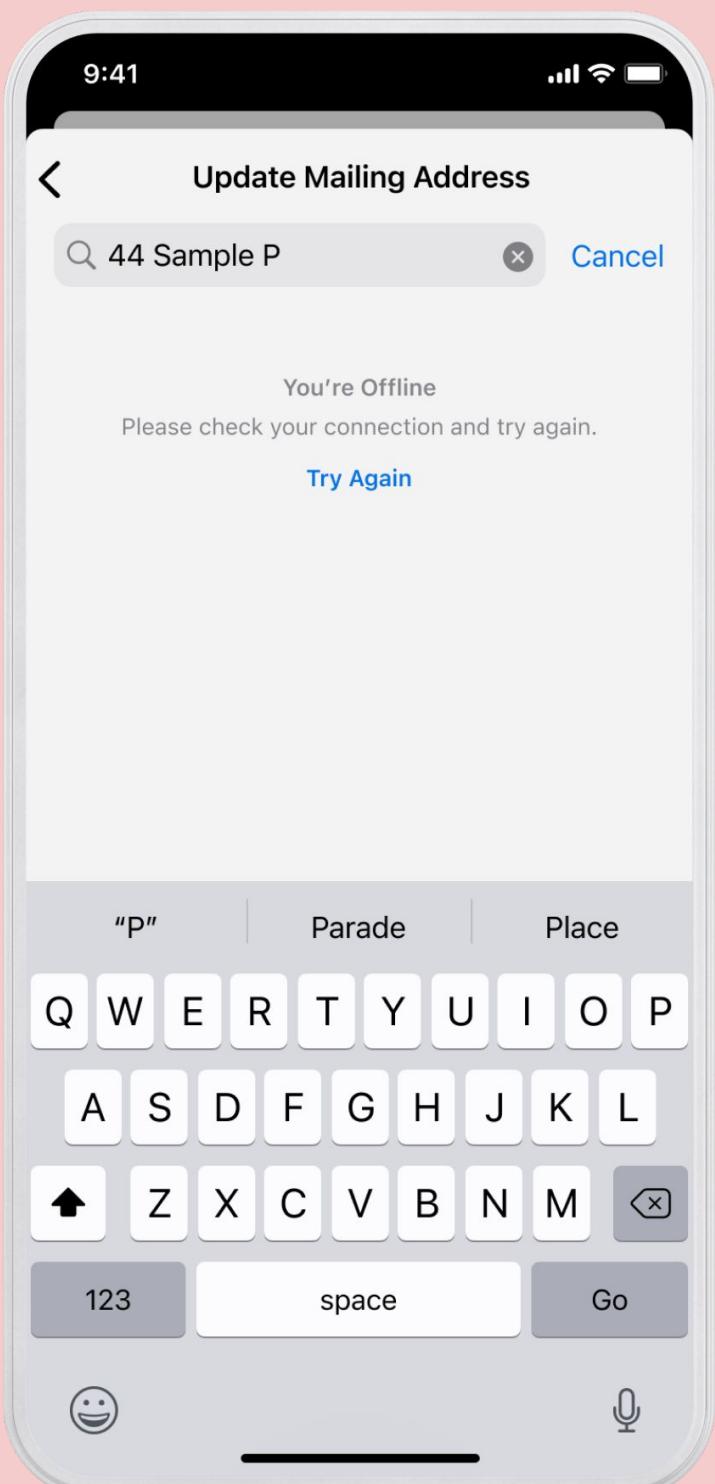




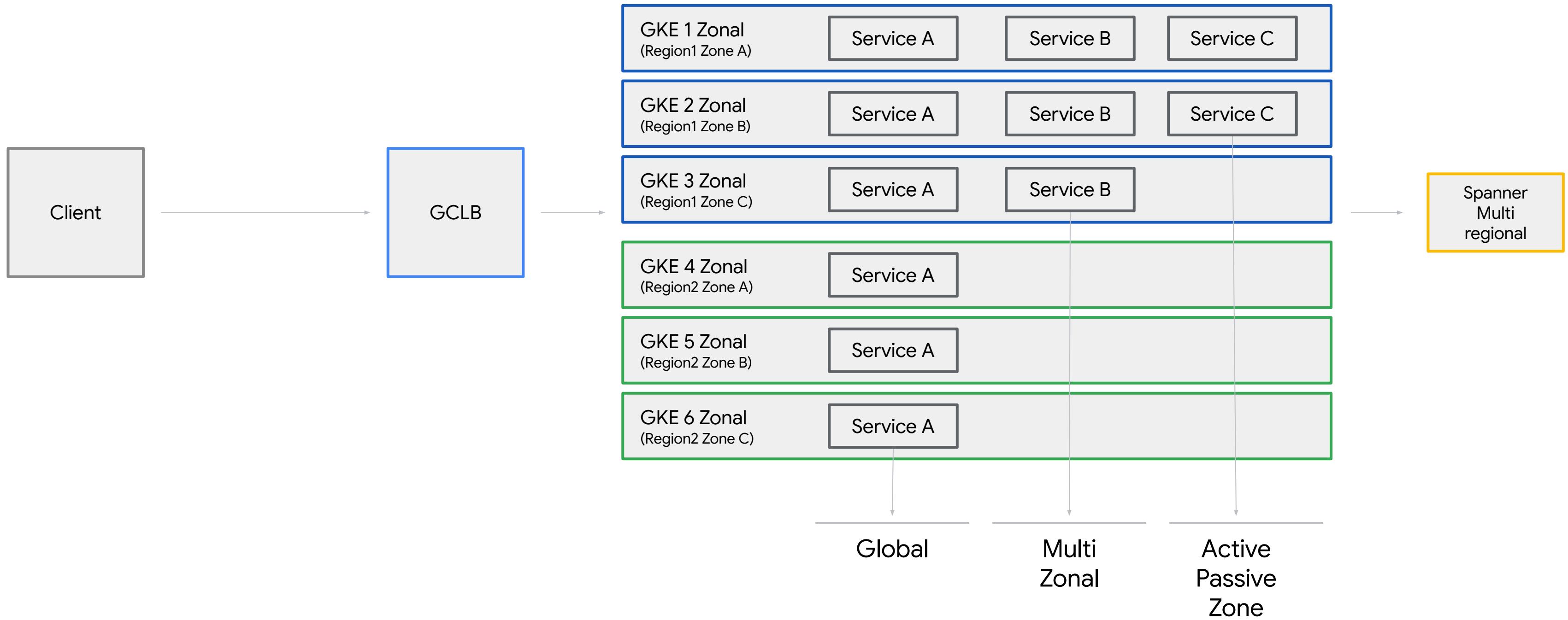
Accounts, Transactions and Payments remain operational



Card and Profile operations can fail



Application with Services using multiple Archetypes



Stepping back...

Cloud is complicated!

Stepping back...

Cloud is complicated!

Teams are swamped.

Stepping back...

Cloud is complicated!
Teams are swamped.

Platforms can help.



Platforms can help, but there are challenges..

- Platforms need to cater for the masses, and allow for graceful **opt-outs**.
- We have had mass adoption quickly, but parts of the platform still **need improvement**
- Production-like **test environments** aren't always possible





Would I do it again?

- **Absolutely!** And our DORA metrics back this up.
- Platforms enable **reliable applications**, improve **developer experience** and **reduce toil** for operators
- **A Platform is software.** Solve problems with data and code



Keep going



Code

[Clone, Fork, Contribute:](#)

goo.gle/reliable-app-platforms



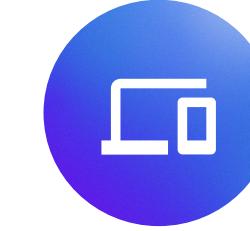
Community

[Monthly Video Discussions](#)

→ r9y.dev/discuss

[Chat on Discord!](#)

→ r9y.dev/chat



Sessions

[ARC103](#) - The reality of reliability:
Big differences between
hyperscalers, explained

[OPS221](#) - What's new in DevOps
tools

[OPS105](#) - The future of platform
engineering is application-centric

Ready to build what's next?

Tap into **special offers** designed to help you **implement what you learned** at Google Cloud Next.

Scan the code to receive personalized guidance from one of our experts.



Or visit g.co/next/24offers

Continue your learning journey!



Sessions

[**Session ID** - Title](#)

[**Session ID** - Title](#)



Sessions

[**Session ID** - Title](#)

[**Session ID** - Title](#)



Sessions

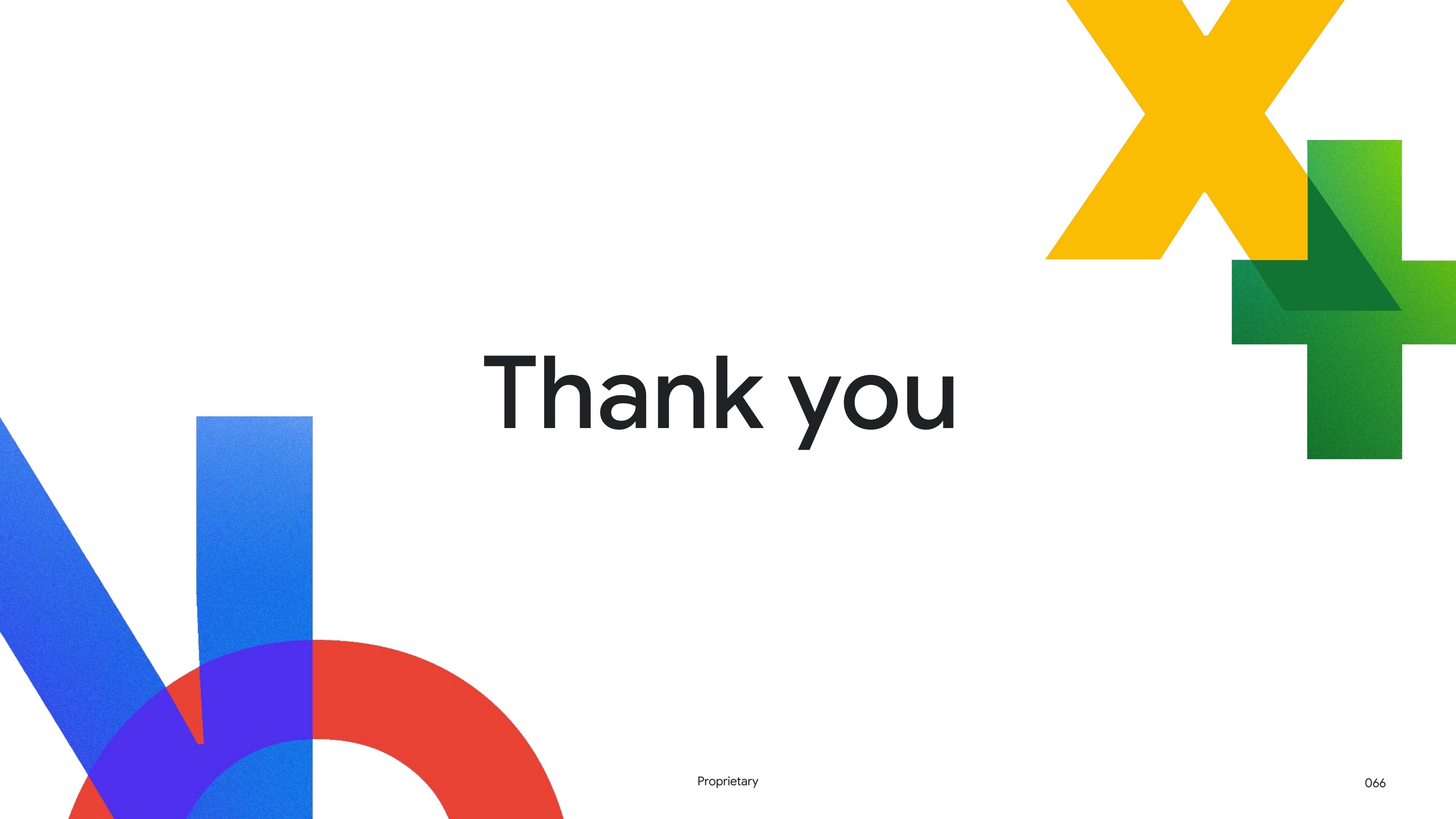
[**Session ID** - Title](#)

[**Session ID** - Title](#)

Your
Feedback
is greatly
appreciated!



Complete the session
survey in the mobile app



Thank you

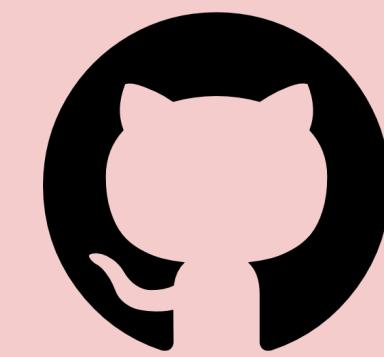


NOBL9

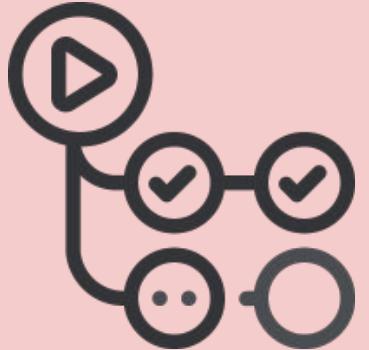


- **Critical services** deployed as Active-active, dual region
- **Observability** stack also dual-region
- **Feature toggling** for outage tiles and disabling UI components

Observability, Feature Toggling remain operational



flux



- **CI/CD not** dual-region initially
- Changes during primary-region-failure: **applied manually** using **break-glass** procedures
- Dual region implementation later

CI/CD can fail



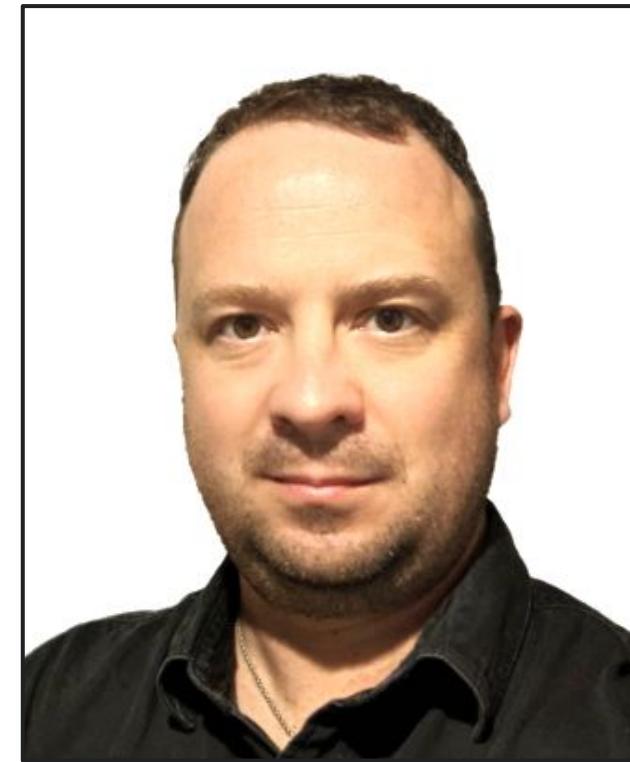
**Steve
McGhee**

Reliability Advocate,
Google Cloud



**Ameer
Abbas**

Outbound Product Manager,
Google Cloud



**Daniel
McKeown**

Area Lead, Cloud and
Orchestration Platform,
ANZ Bank

“

So what, Steve?

What do we actually DO.”



Adopt a platform:

give your team
the means to

run from problems

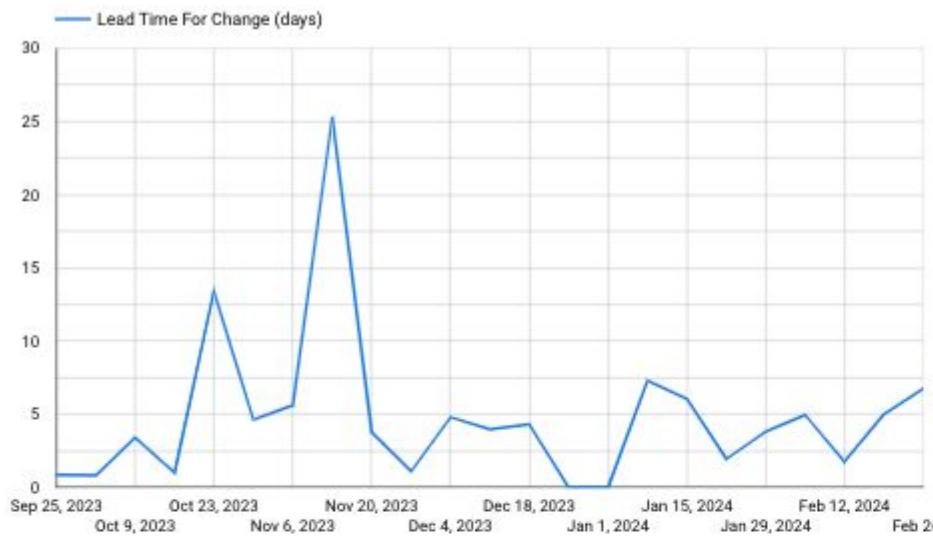
and

run toward opportunities

The Four Key DORA metrics

Weekly Average Lead Time for Changes

The average amount of time for a commit to be deployed into production

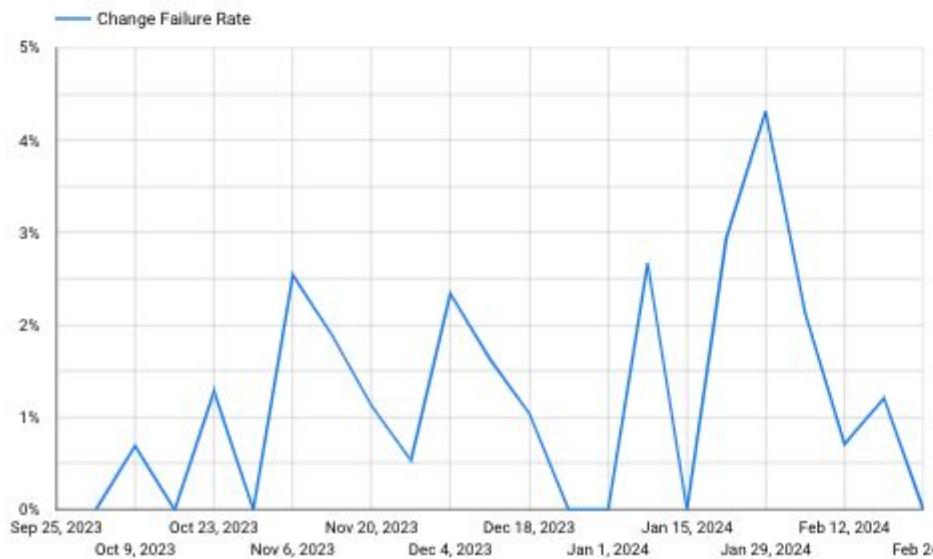


Average Lead Time for Changes
< 1 week (High)

Deployment Frequency
Daily (High)

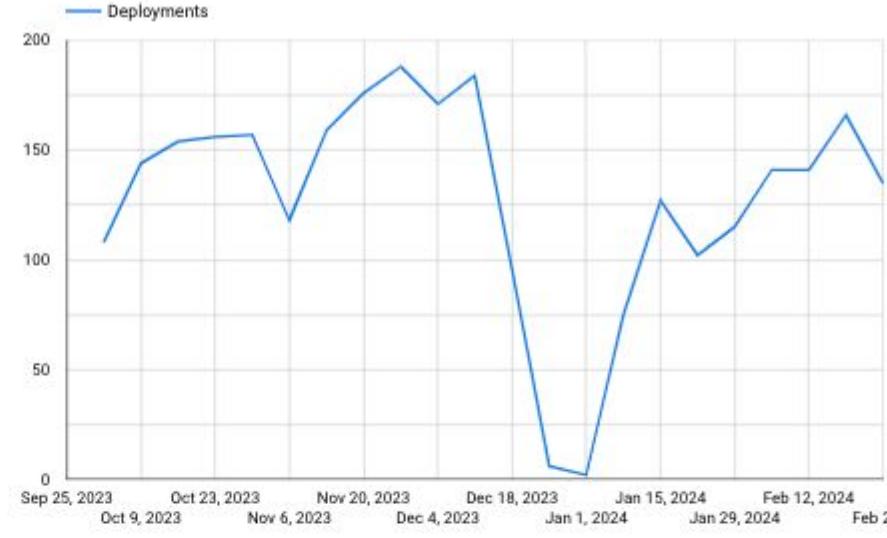
Weekly Change Failure Rate

The number of Failures per the number of deployments. e.g. if there are four deployments in a day and one causes a failure, that would be a 25% change fail.



Weekly Deployments

The number of deployments per week



Deployment Frequency
Daily (High)

Change Failure Rate Rating
0-15% (High)

Time to Restore Service Rating
< 1 day (High)

Lead Time for Changes Only

GitHub Repository

Approved by

Harness Application

Harness Environment

Harness Workflow

IsFirstDeployment

Oct 1, 2023 - Mar 1, 2024

Non-Lead Time for Changes Only

Service Now Assignment Group

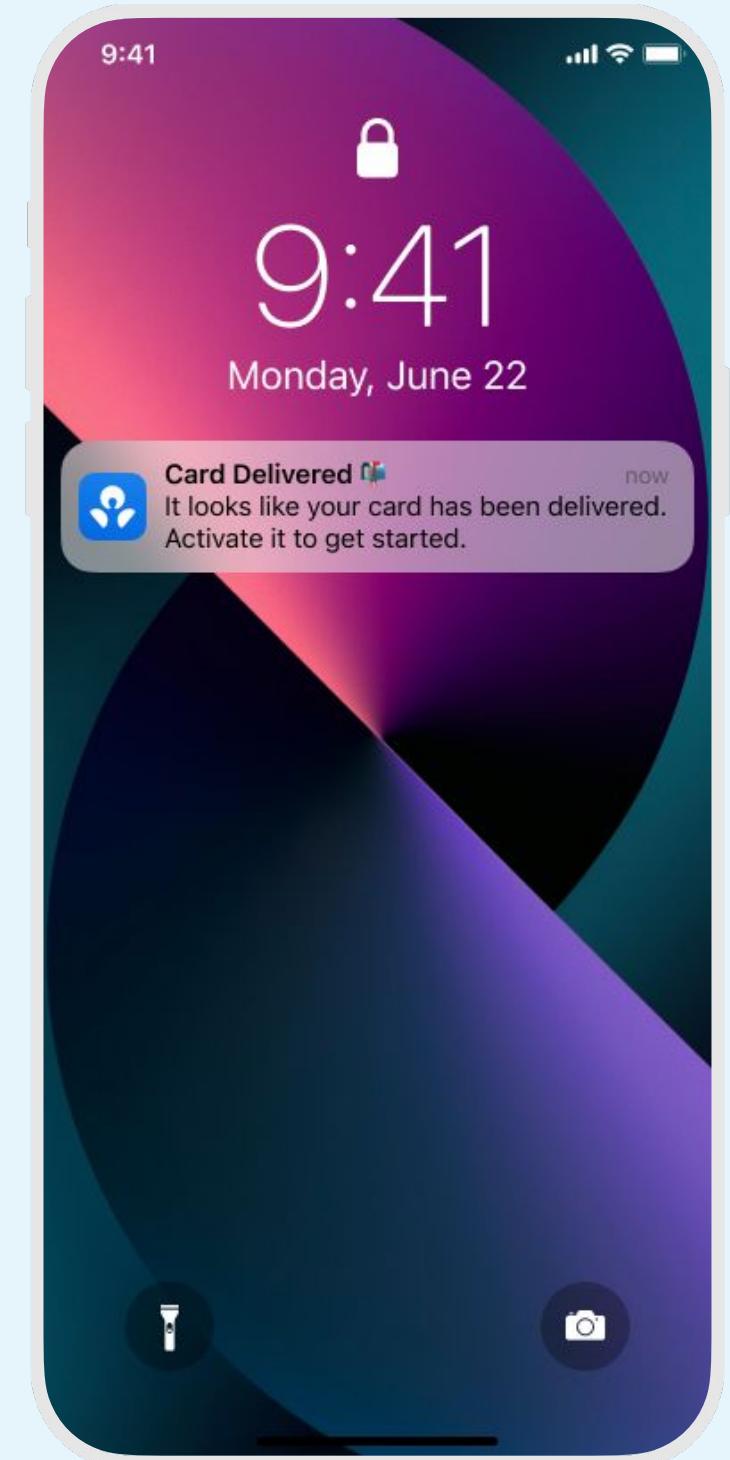
Service Now Business / Tech Area

Service Now Change Type

Service Now Configuration Item

Service Now Change Risk Level

Priority: 1, 2, 3 (3)



Lead Time for Change

Production Deploy Lead Time

0.13

1.22

% diff from previous month

Merge Lead Time

0

-0.51

% diff from previous month

Approval Lead Time

1.43

-0.52

% diff from previous month

First Review Lead Time

0.09

-0.96

% diff from previous month

Build Lead Time

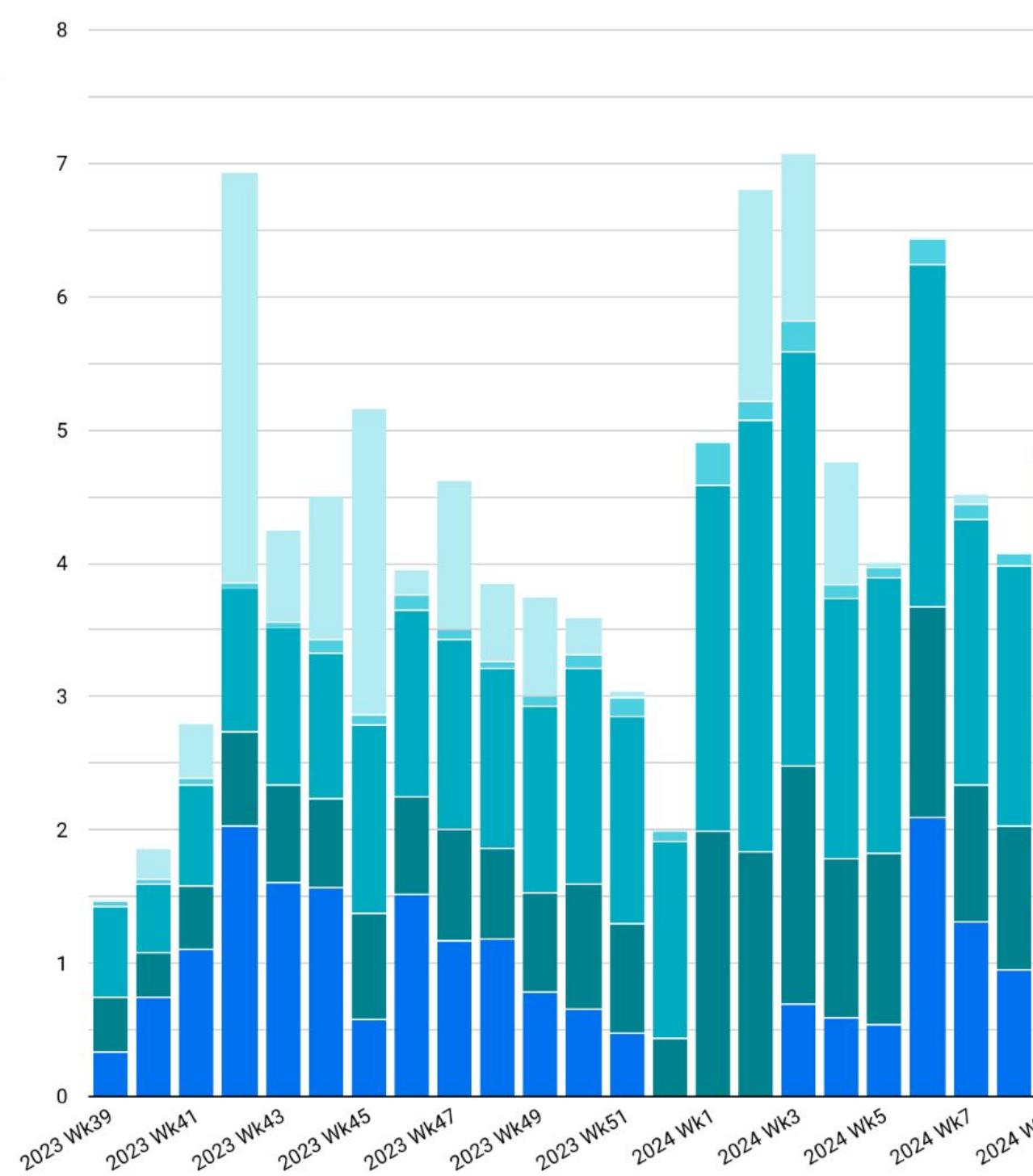
0.01

-1

% diff from previous month

Current Monthly Averages (days)

ANZx Delta Lead Times



GitHub Repository

Harness Application

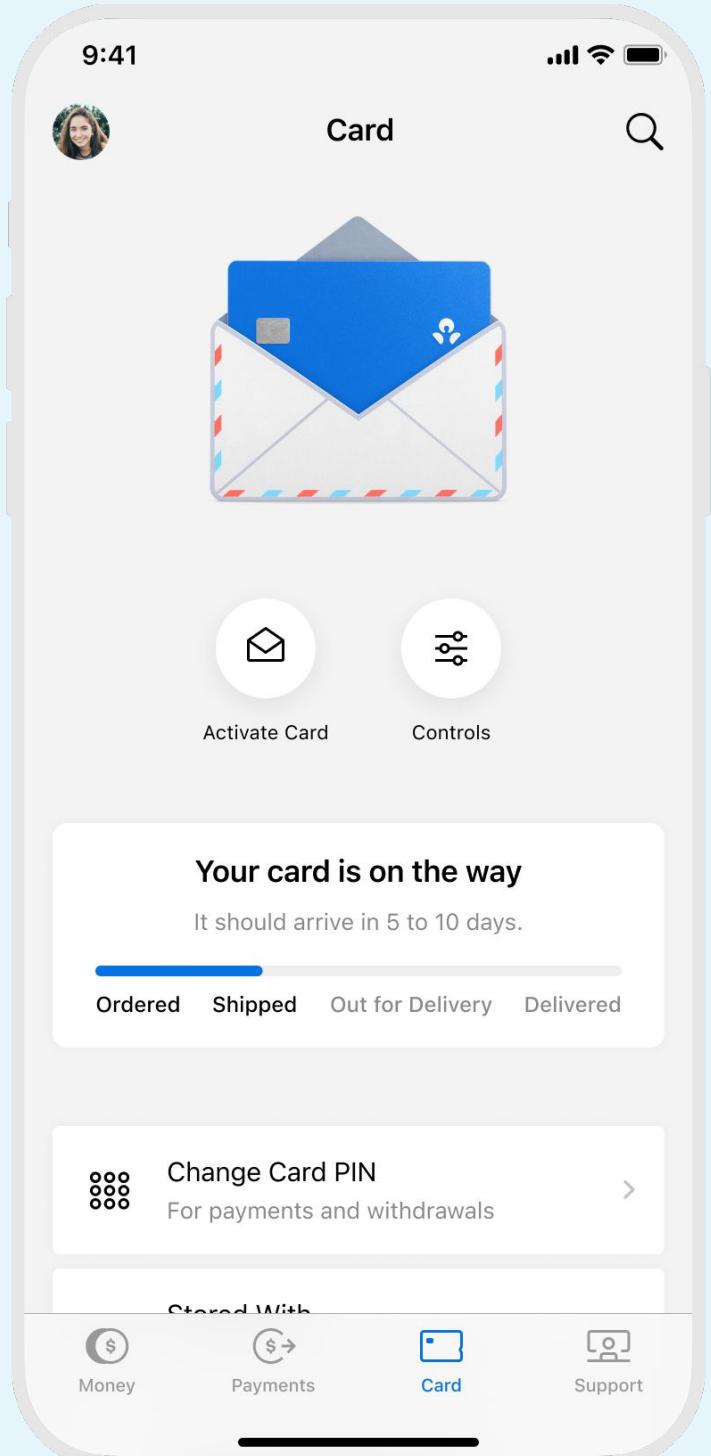
Harness Environment

Code Owners Approvers

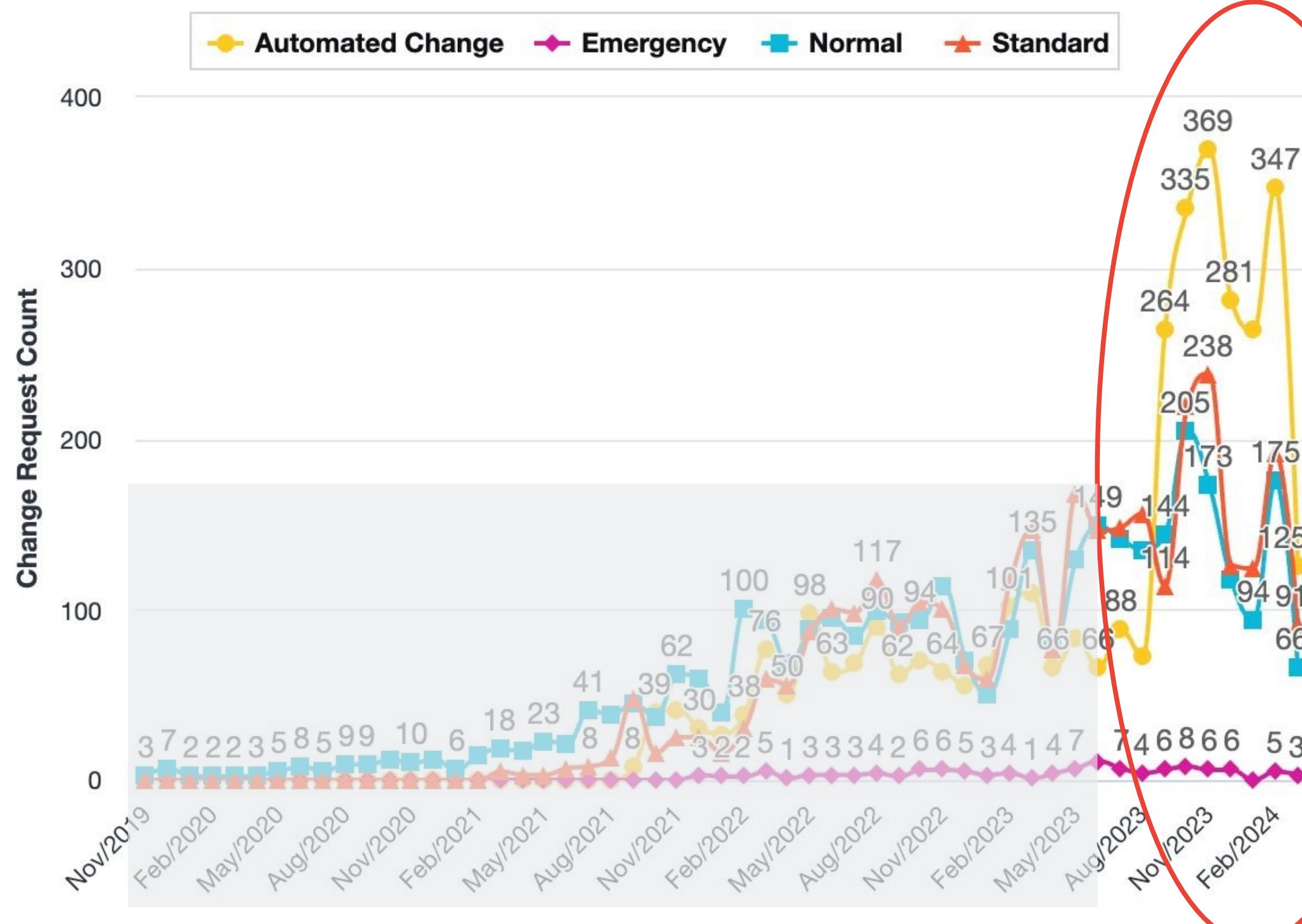
Oct 1, 2023 - Mar 1, 2024

Time Period: Week

- Prod Deploy Lead Time
- Merge Lead Time
- Approval Lead Time
- First Review Lead Time
- Build Lead Time

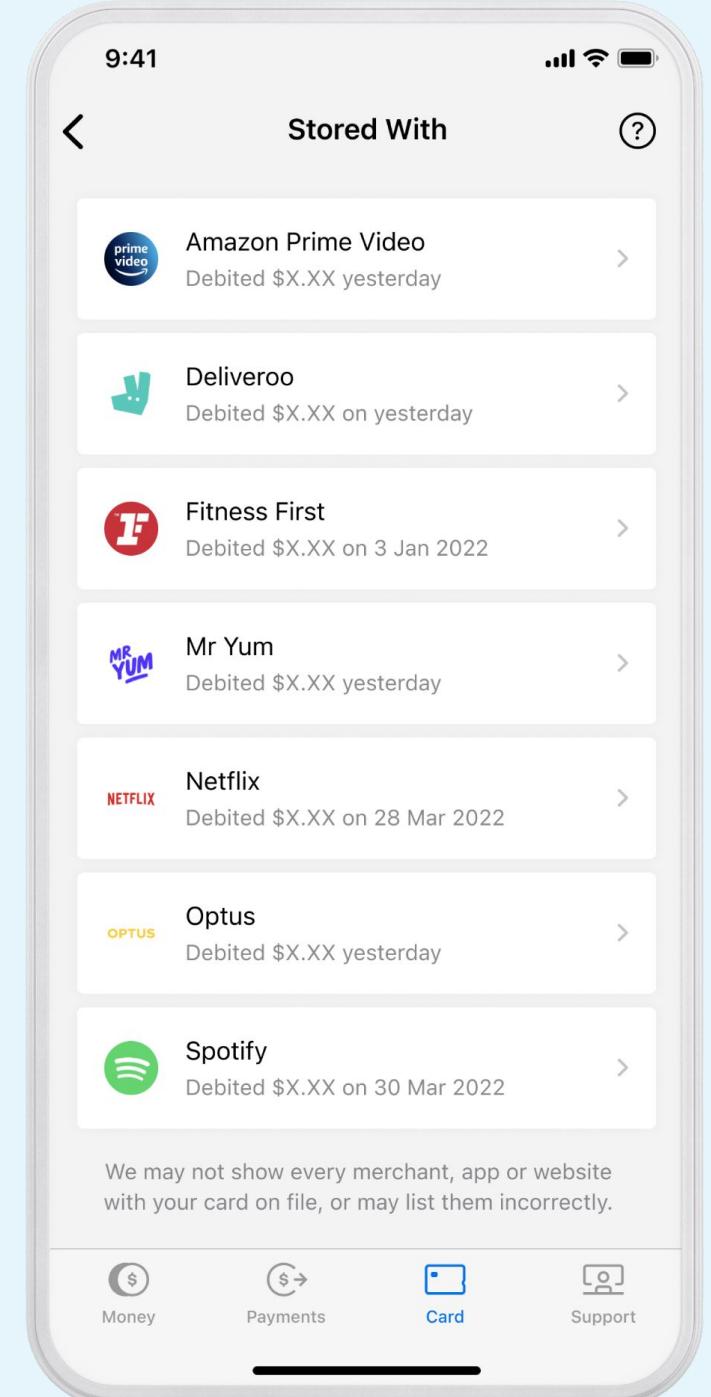


Platform enable Elite teams



Proprietary

Google Cloud Next '24





Benefits of The X Framework



Asset transparency



Reduced developer cognitive load



Reduced operator toil

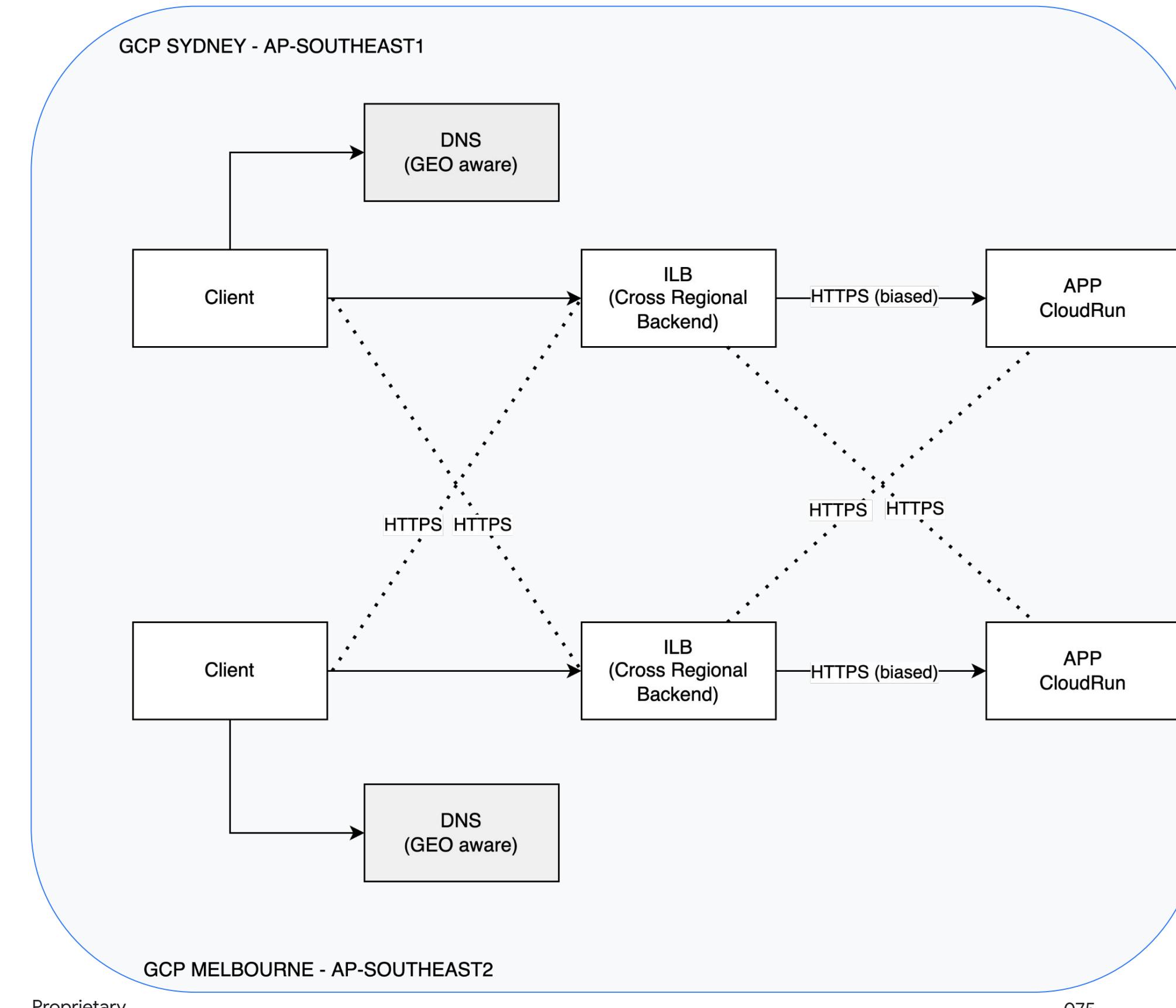


Community Governance



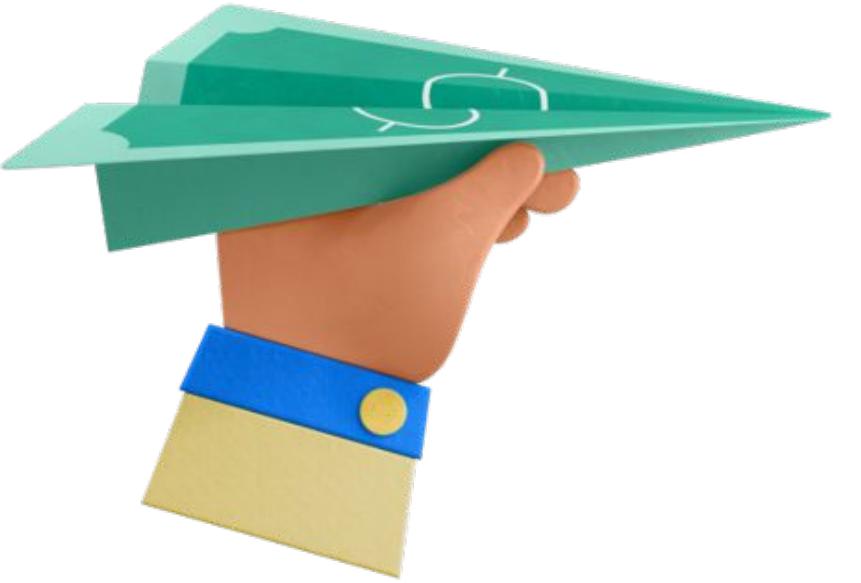
Reliability and failure domains in ANZ Plus

- Achieving multi-region, **active-active** is complex
- Google Services **operate differently** in HA configurations
- Leveraging:
 - dual-region **Cloud Spanner**
 - stateless **Cloud Run** instances
 - region aware **Load Balancing**



X Framework - Adoption

- 95% of KRM is now managed by gitops, drastically improving our **DR posture** and our ability to **operate** the platform
- 85% of repos have **fully generated** CD pipelines
- 80% of workloads are hosted using preferred **runtime patterns**



X Framework - Progress and Plans

Service creation	Cloud features	Governance	Dev tooling	Integration & Ops
 Project creation	 Google Secret Manager	 Azure AD Groups	 GitHub Actions	 Temporal
 Kubernetes Config Connector	 K8s ingress & egress	 Service accounts	 Backstage Component & Techdocs	 Feature Flagging
 GitHub repository template	 Load Balancers and Subnets	 Google Cloud IAM	 Google Artifact Registry	 Identity integration
 Namespace creation	 Cloud Run	 Application Portfolio Management	 sonarqube	 SLO Management
 Service Archetypes	 DataFlow	 Risk & Controls	 BLACKDUCK	 Cloud Monitoring
	 Cloud DNS		 CHECKMARX	
	 Pub/Sub			
	 BigQuery			
	 Cloud KMS			
	 Spanner			
		 Proprietary		



Criticality ↑

- **Critical platform capabilities** deployed as Active-active, dual region
- **Observability stack**
- **Feature toggling** for outage tiles and disabling UI components

Observability,
feature toggling
must remain
operational

- **CI/CD not** dual-region initially
- **Manual failover** using **break-glass** procedures
- Dual region implementation will come later

CI/CD allowed to
fail

Platform

“

Platforms are a means of **centralizing expertise** while **decentralising innovation** to the customer or user. This happens through commoditization of services, but in a way which **relinquishes an amount of control** and is **open to extension**.

This is a **fundamental change** to many organizations core business models, which tend to favor standard unification and work centralization.”

Peter Gillard-Moss
[ThoughtWorks](#)

Platform Engineering

More focus
More creativity
More agility

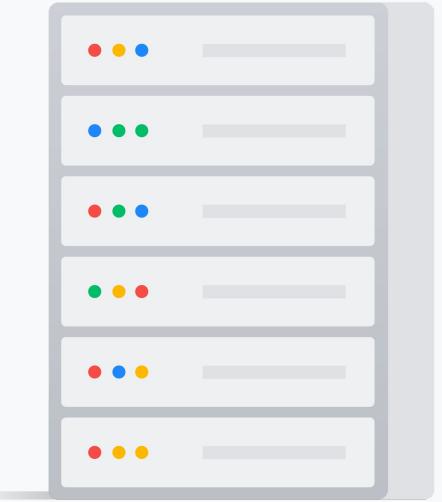


Software Developer

Internal Developer Platform



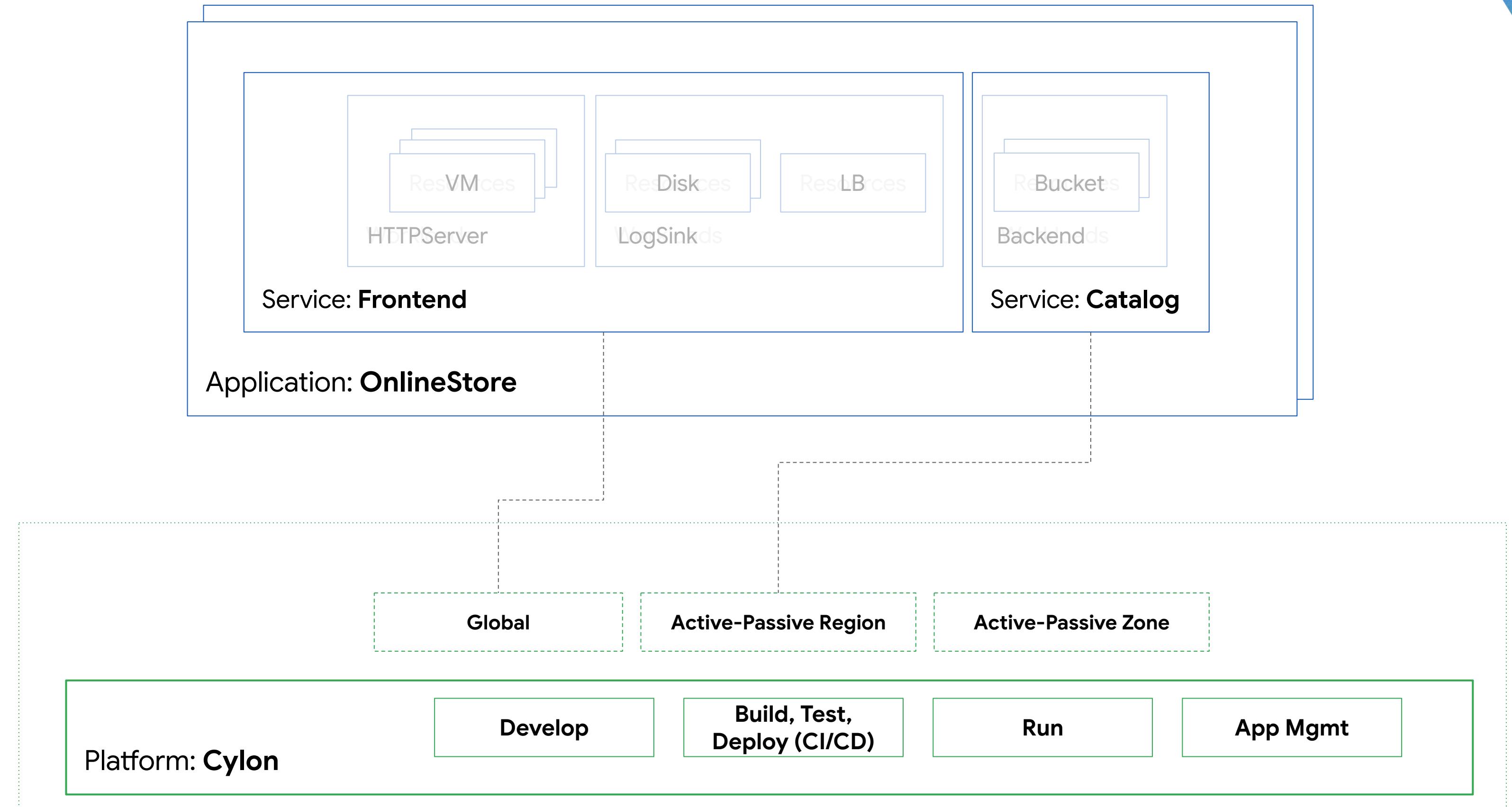
Platform Engineer



Google Cloud
Resources

Infrastructure
Dev and Prod

Public Cloud





Reliability and Failure Domains



Robustness

Scale

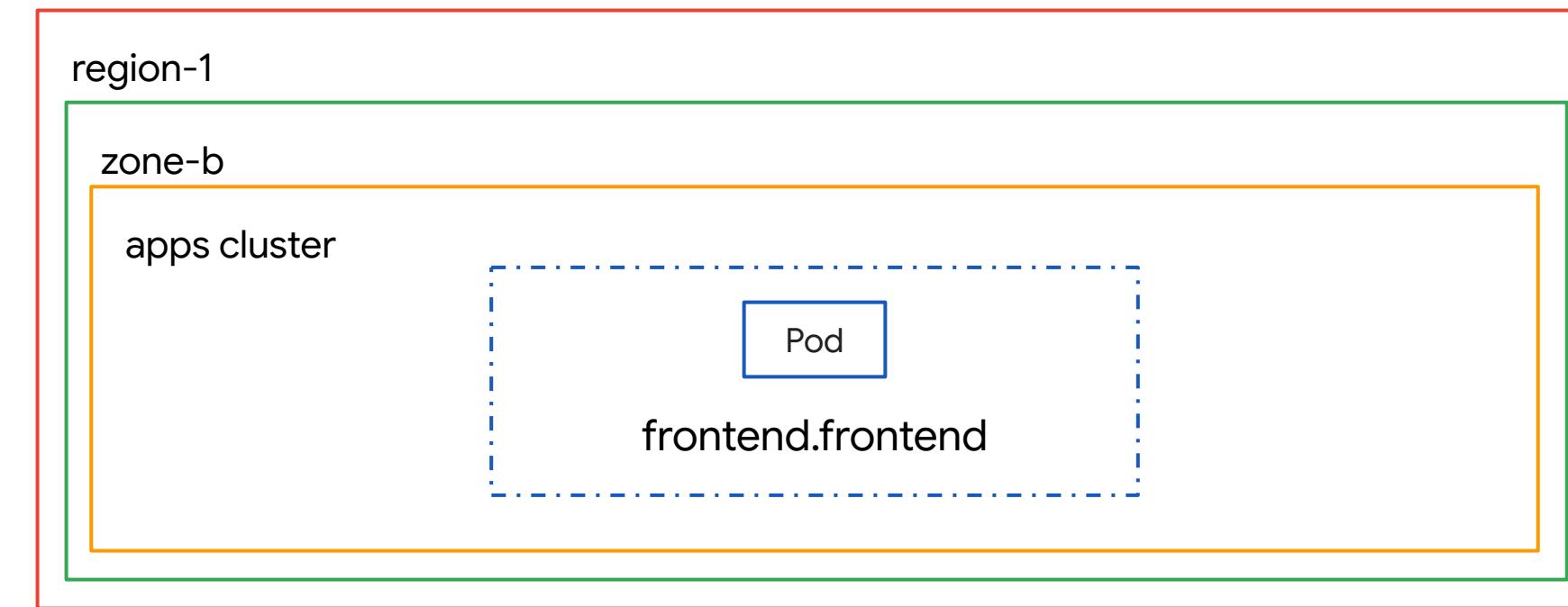


Policy / Security

Manageability

A singleton inside multiple failure domain envelopes

✗ Robustness

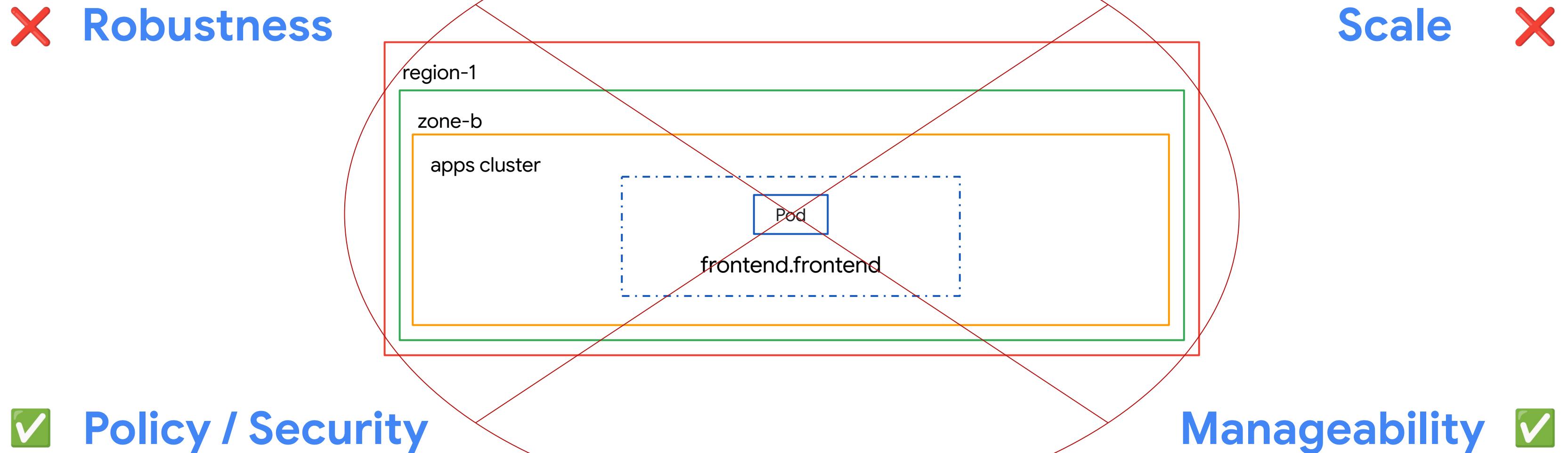


Scale ✗

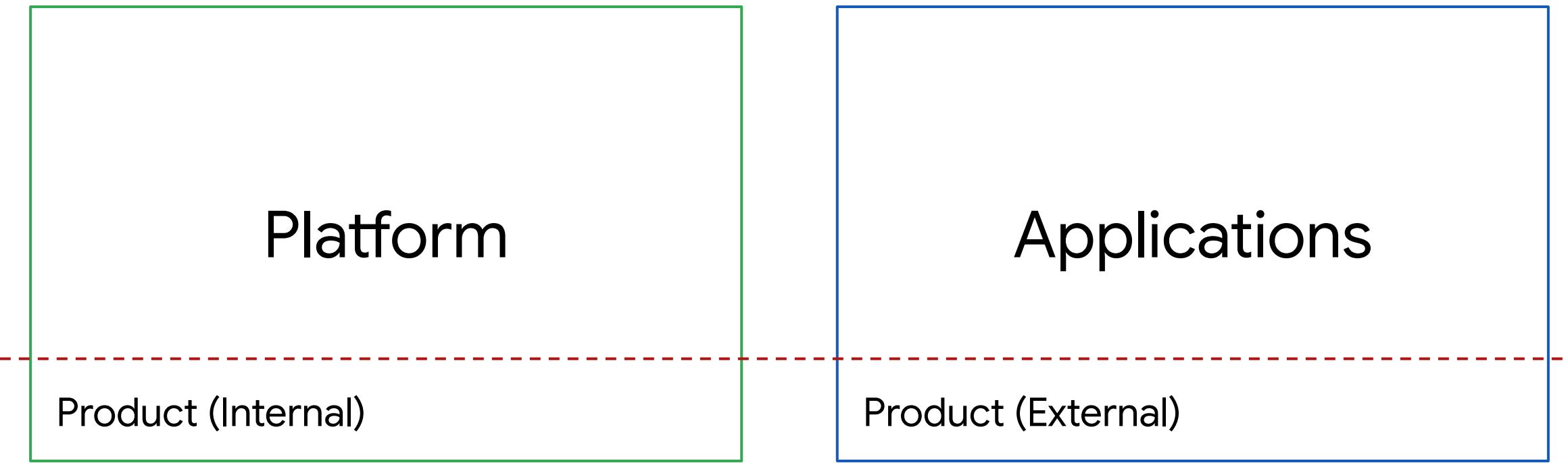
✓ Policy / Security

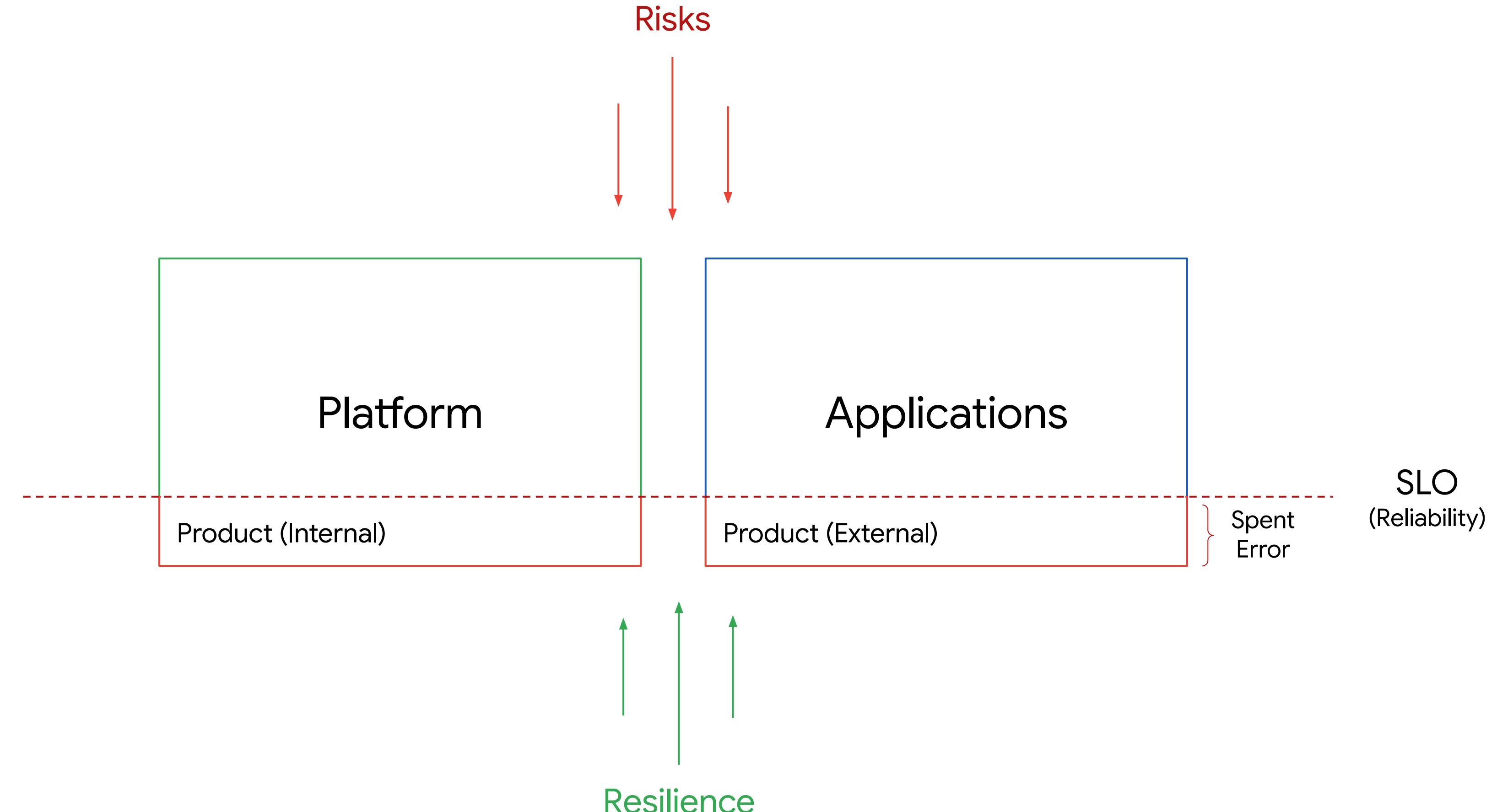
Manageability ✓

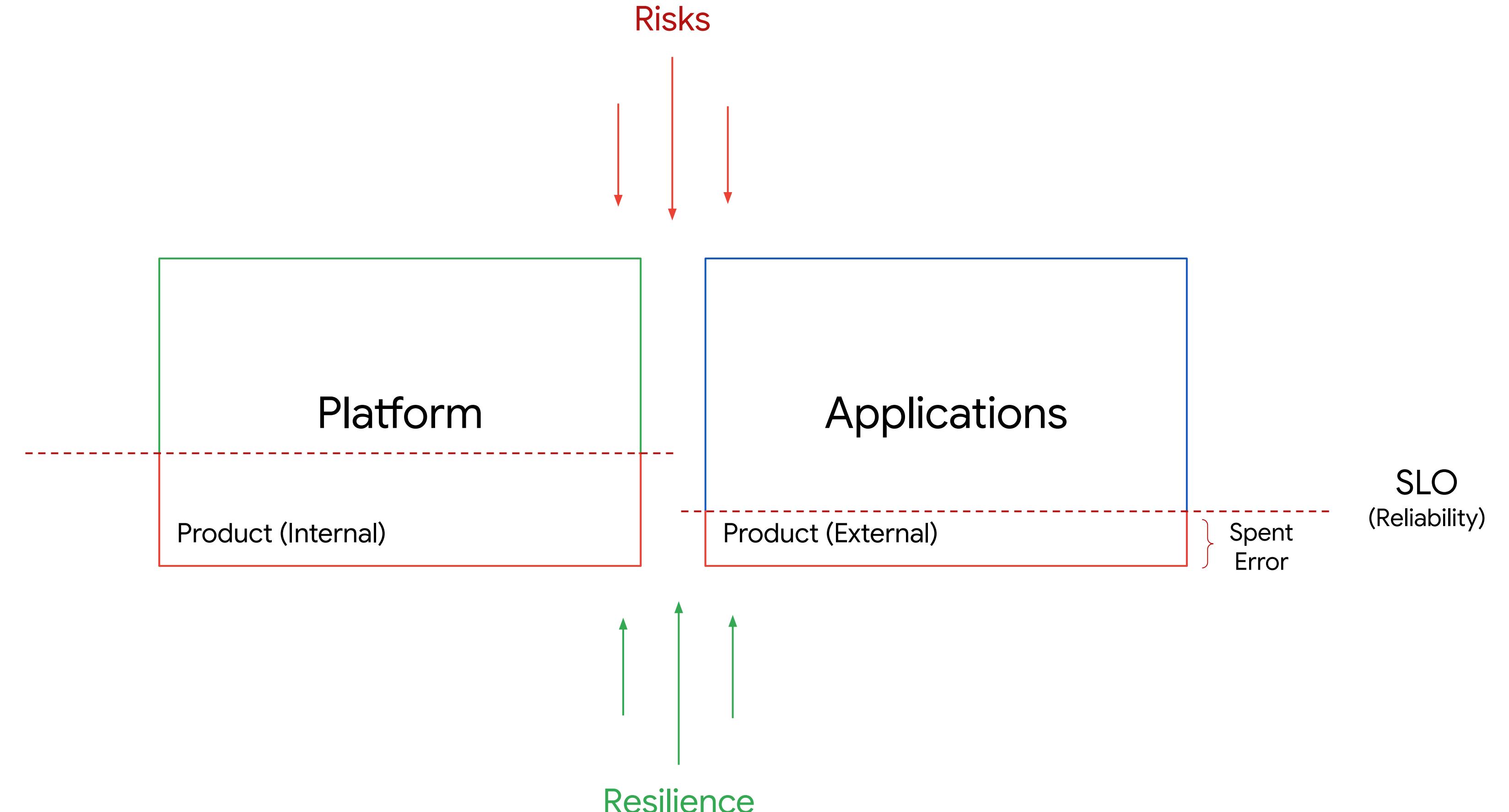
Singletons are **insufficient** for Robustness and Scale



(but may simplify Policy / Security + Manageability)







Your Deck is Under Review



Please refrain from making changes

Application with Services using multiple Archetypes

