AWS re:Invent

FSI302

Migrating monolithic applications with the strangler pattern

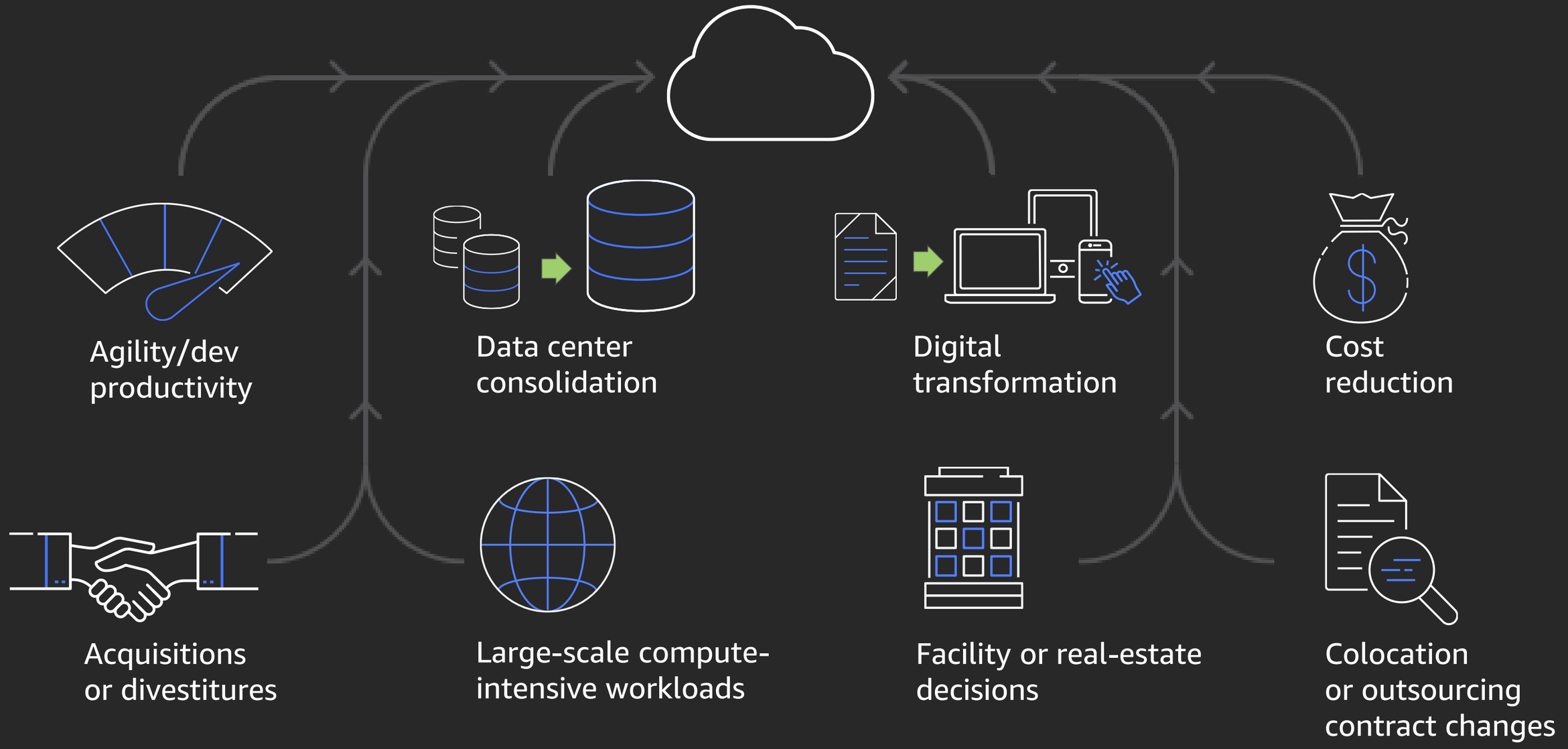
Christopher Marsh-Bourdon
CIO of Development and Enablement
Wells Fargo

Kenneth Jackson
Principal Solutions Architect
Global Financial Services
Amazon Web Services

Agenda

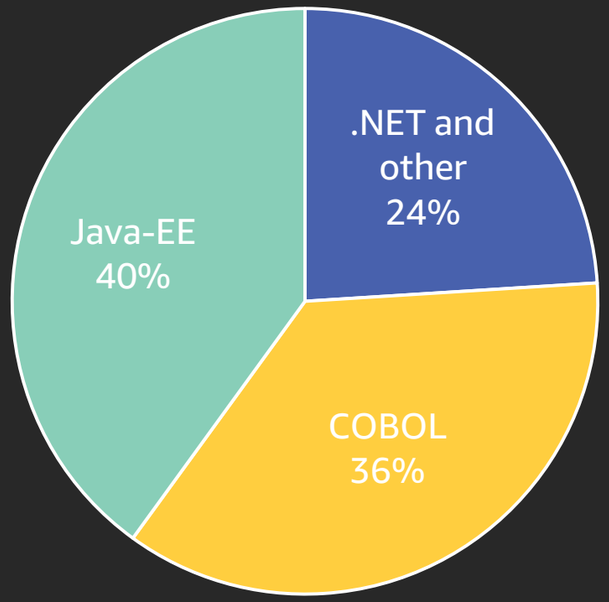
- Common migration drivers
- Problem space: Monoliths in financial services
- Current strategies
- Strangler pattern strategy
- Takeaways

Common migration drivers

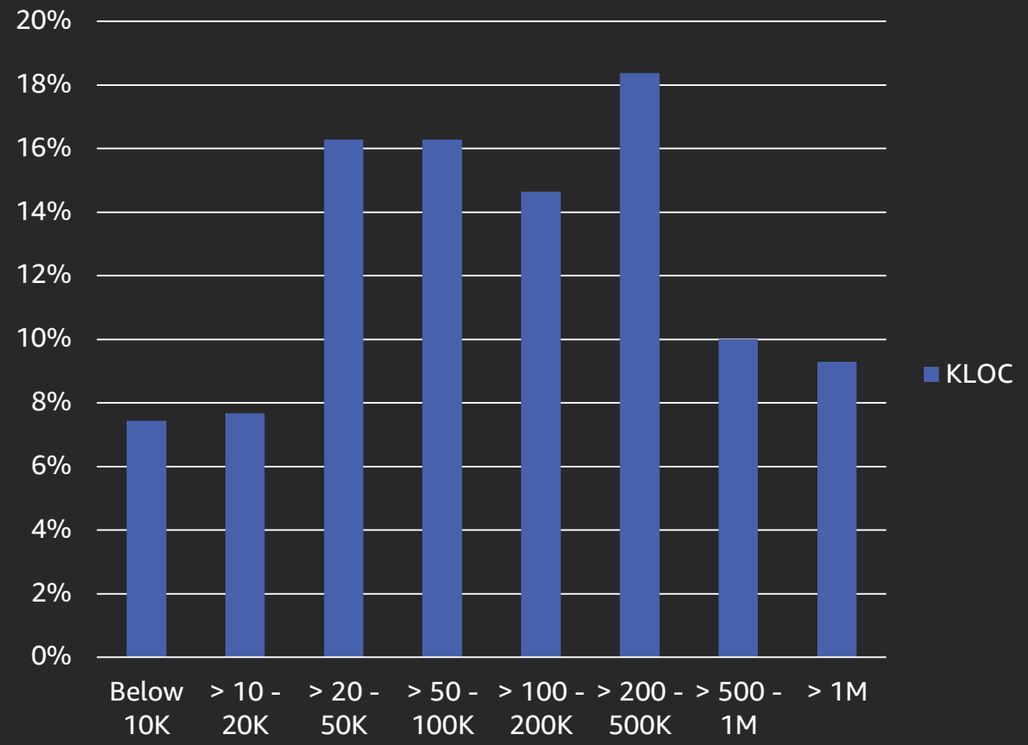


Monoliths in financial services

Technology distribution



Application size



Sample size: 430 applications from 53 firms totaling 241 MLOC

Source: CAST Software – CRASH Report Financial Services 2016

Current strategies

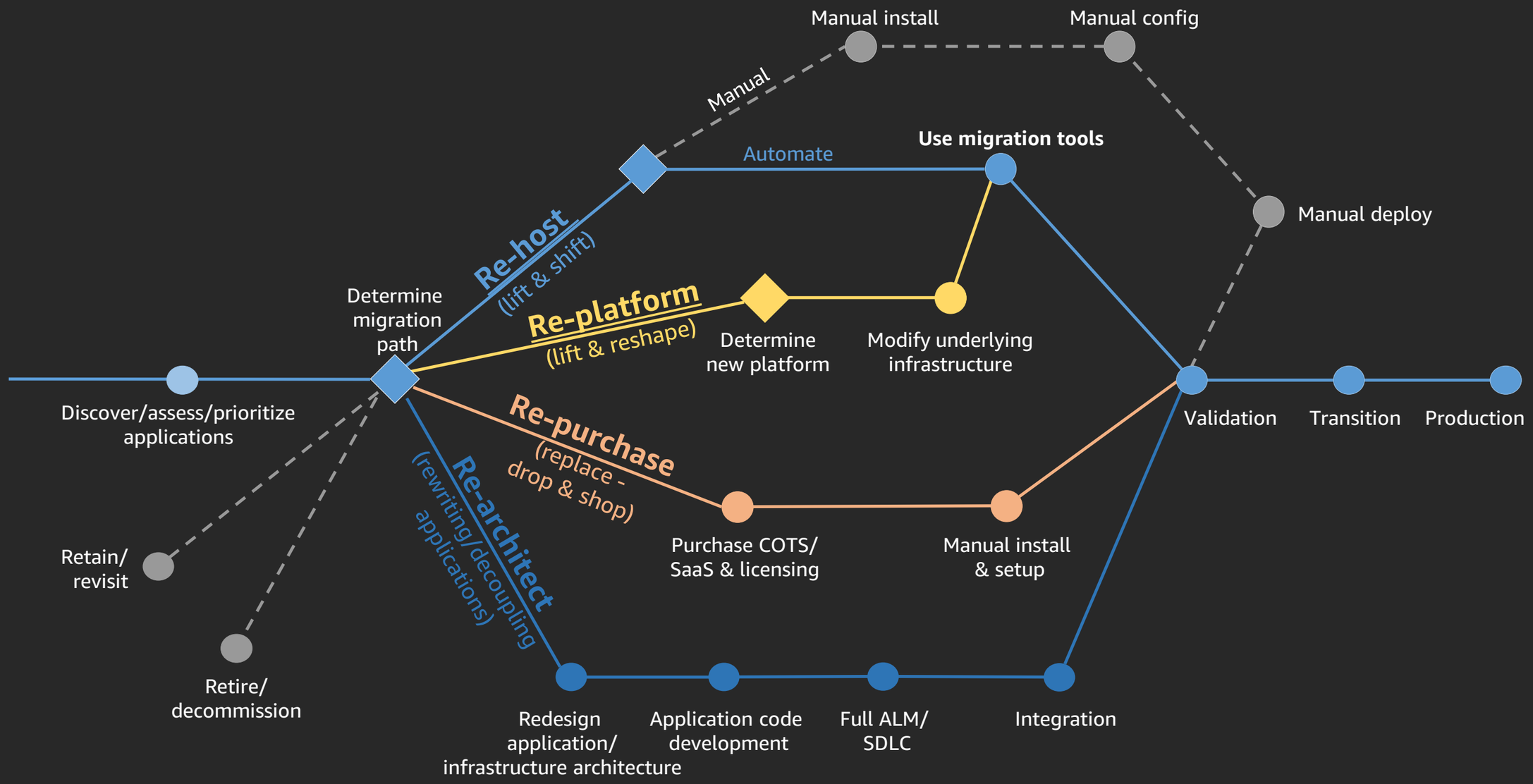
"We find that many early cloud projects gravitate toward **net new** development using **cloud-native** capabilities, but in a large legacy migration scenario where the organization is looking to scale its migration **quickly** to meet a business case, we find that the majority of applications are **rehosted**."

Stephen Orban

Head of Enterprise Strategy, AWS

Author, *Ahead in the Cloud*

The 6 R's: Application migration strategies



Comparison of migration strategies

	Effort (time & cost)	Opportunity to optimize
Retire	N/A	N/A
Retain		N/A
Re-host		
Re-purchase		
Re-platform		
Re-architect		

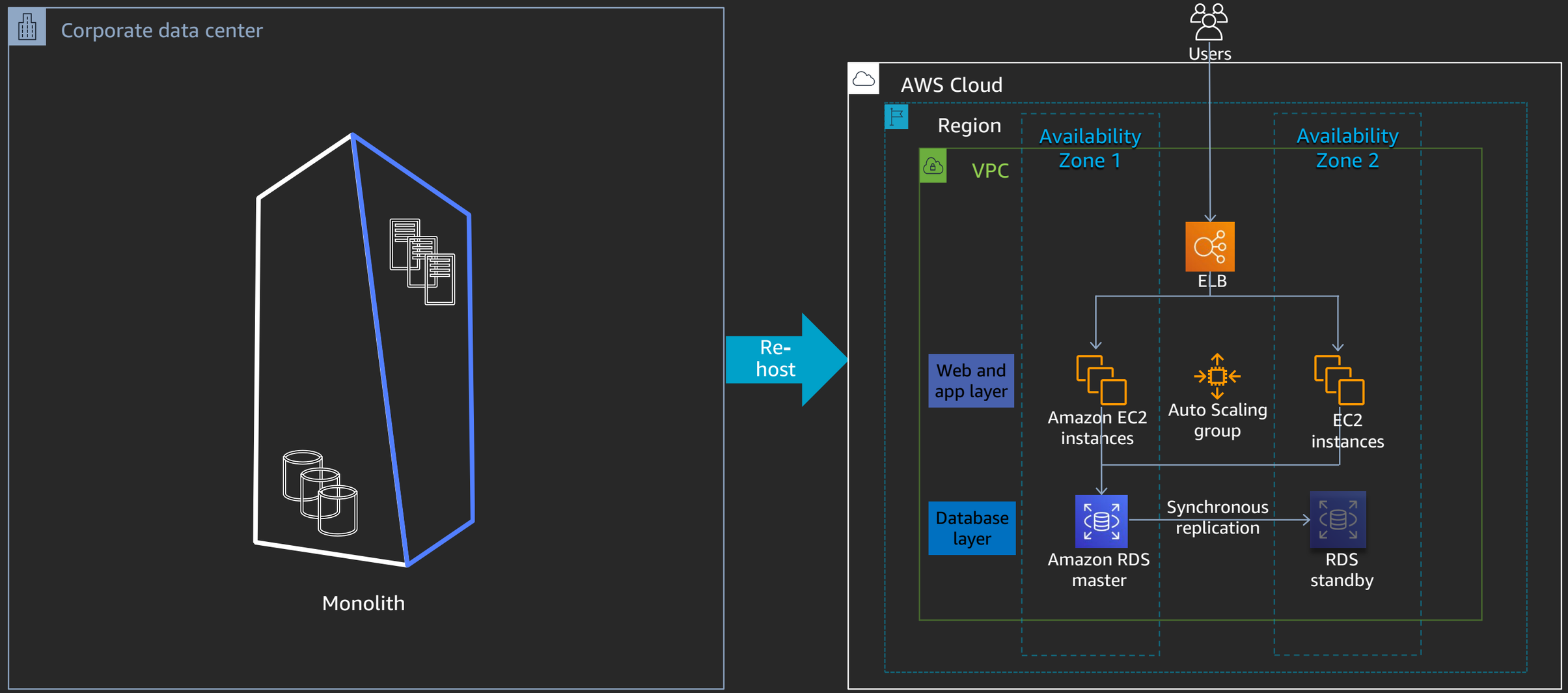
Strangler pattern strategy

"An alternative route is to gradually create a new system around the edges of the old, letting it grow slowly over several years until the old system is strangled."

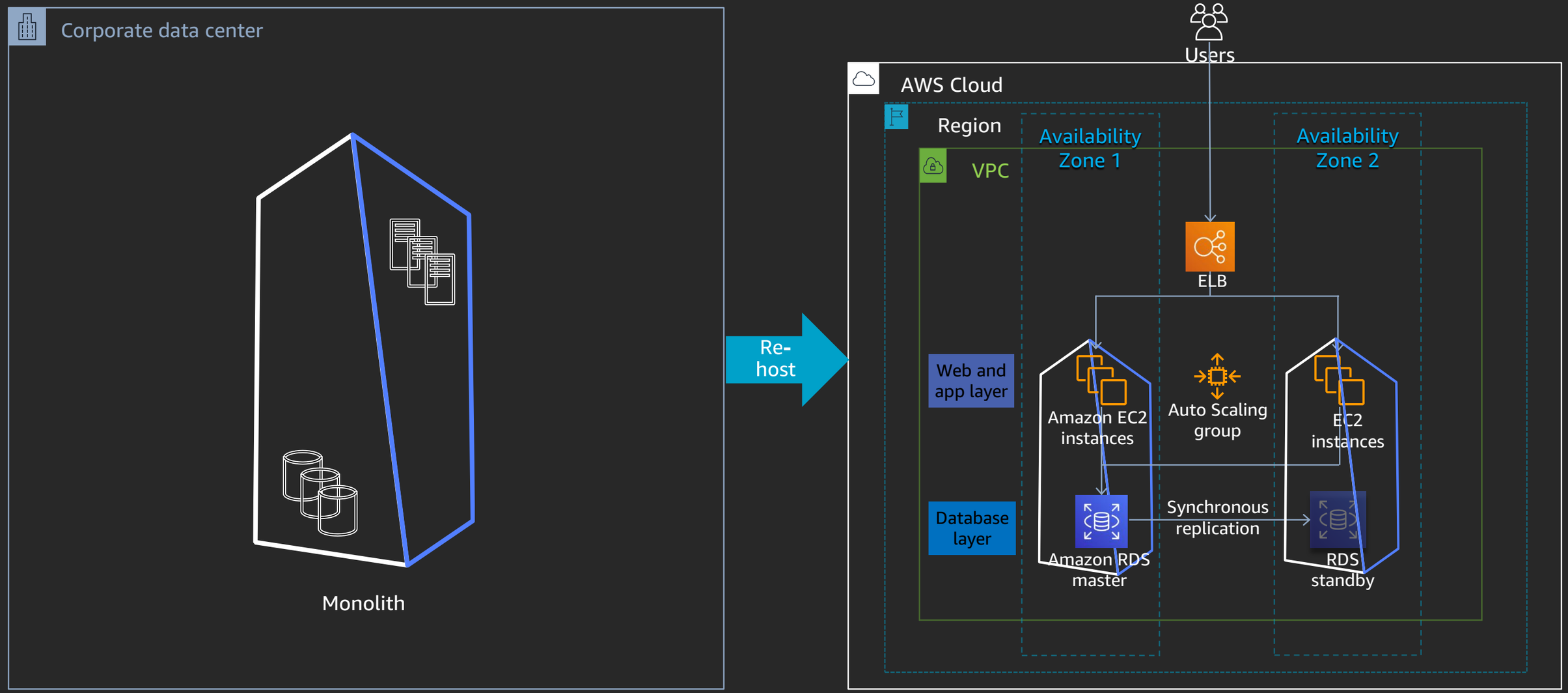
Martin Fowler

Chief Scientist, ThoughtWorks

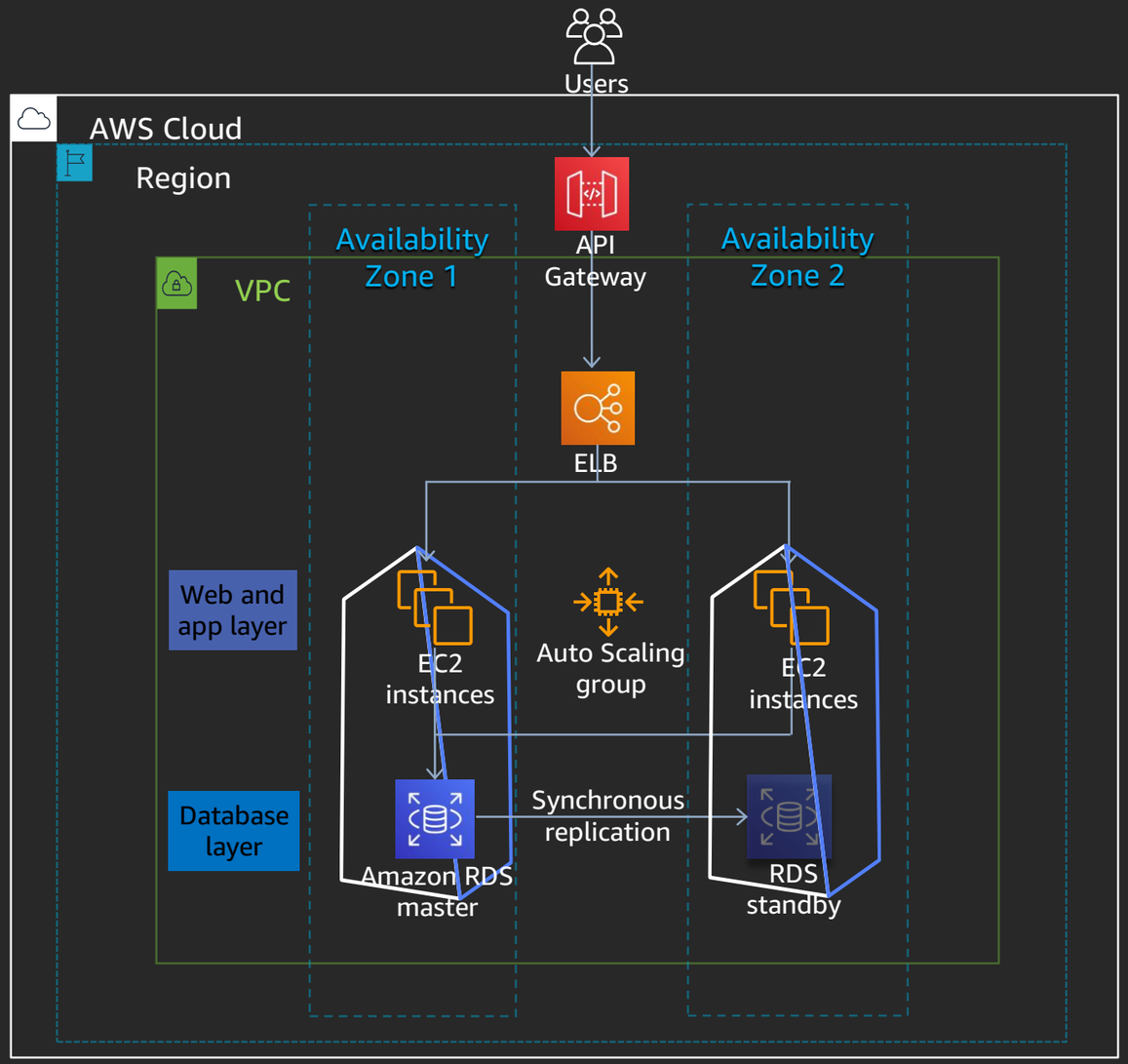
Step 1: Re-host application



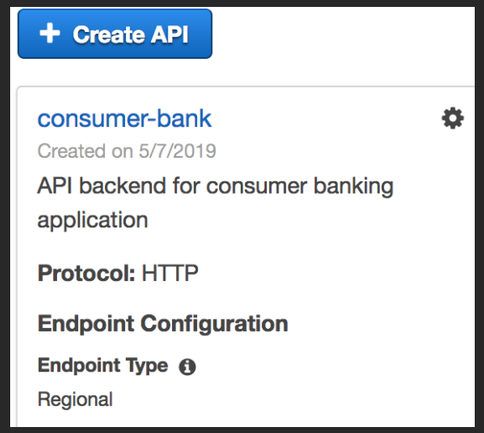
Step 1: Re-host application

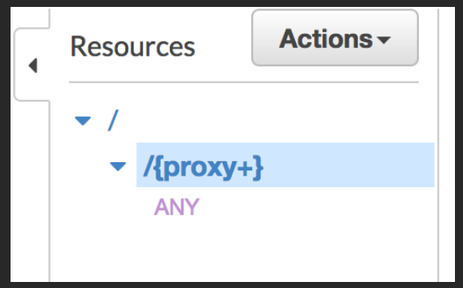


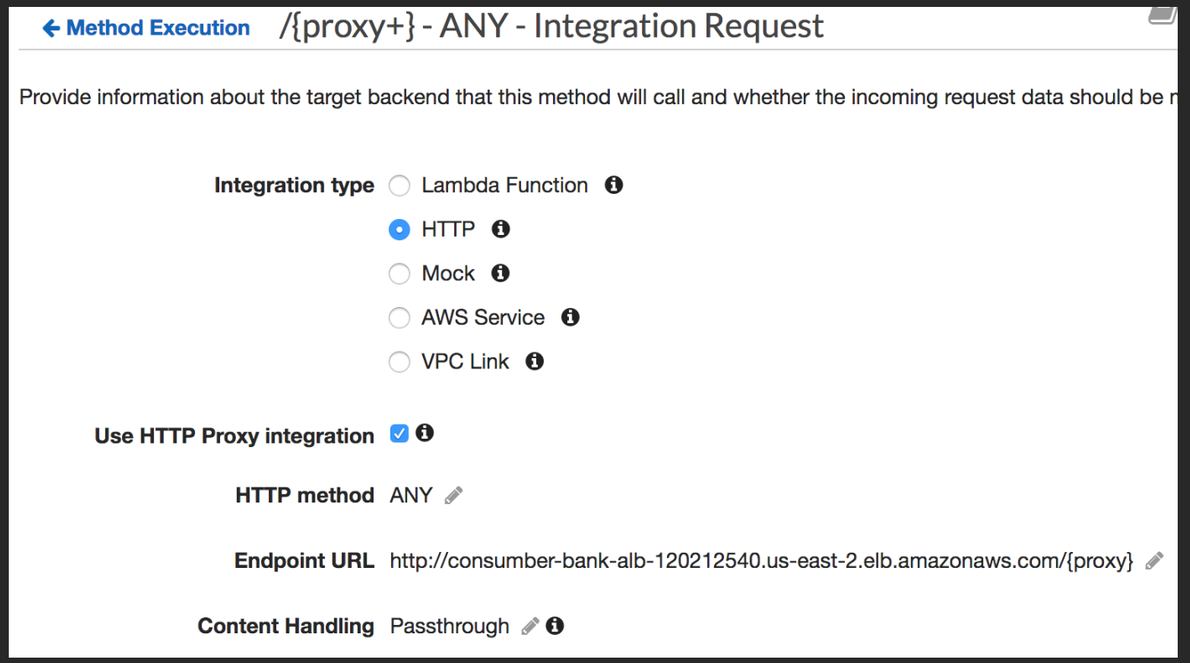
Step 2: Facade with Amazon API Gateway



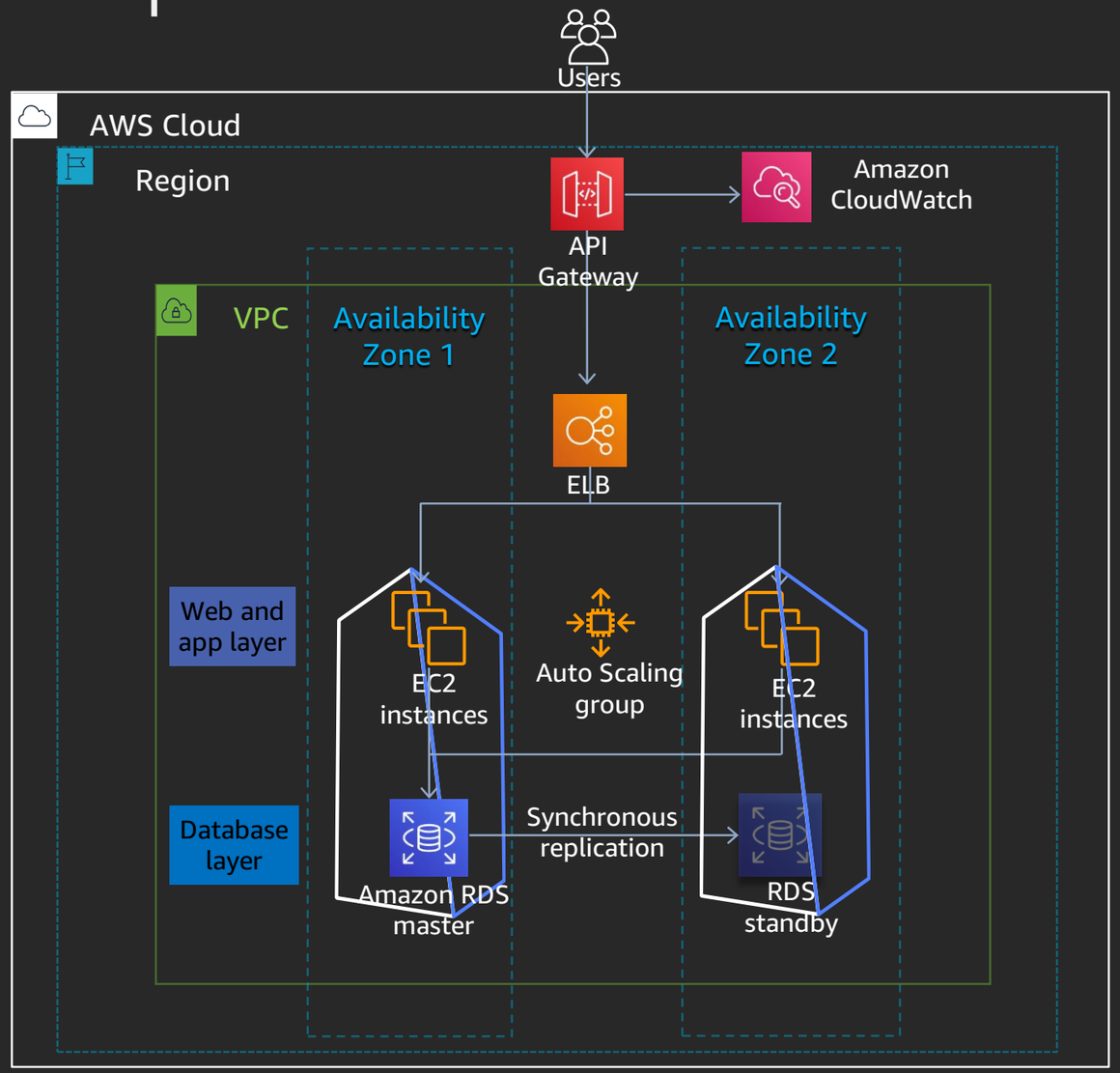
Step 2: Facade with Amazon API Gateway – Setup

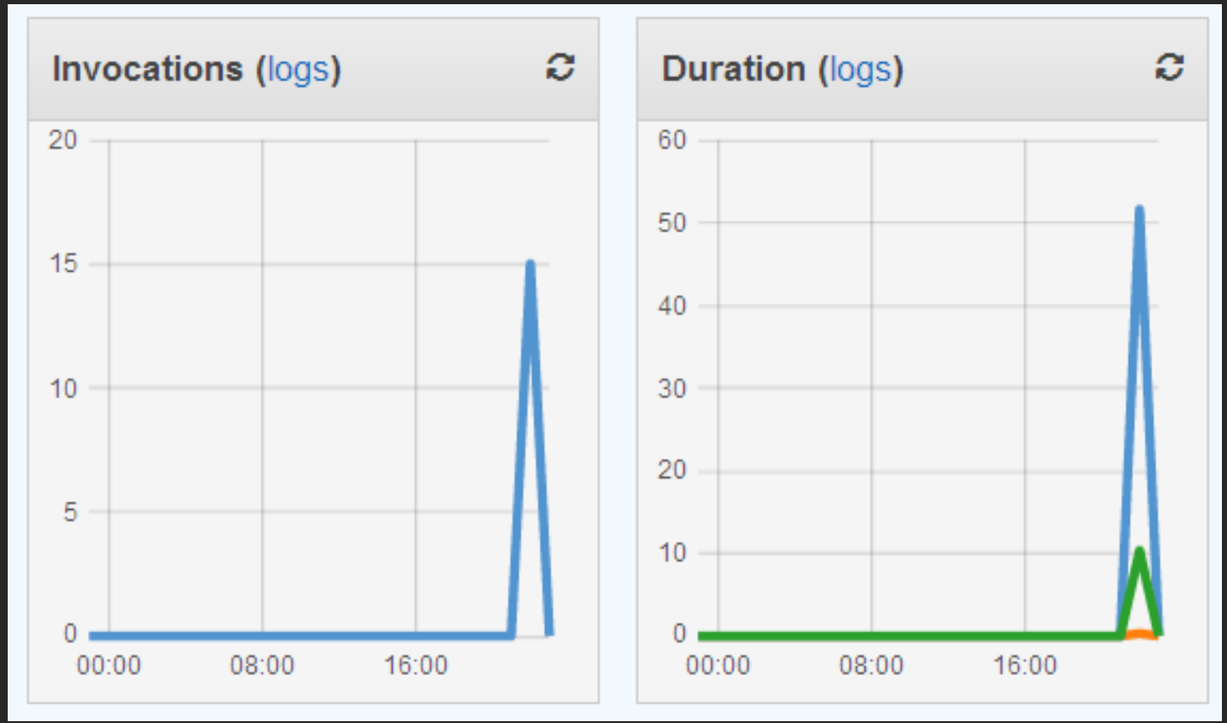




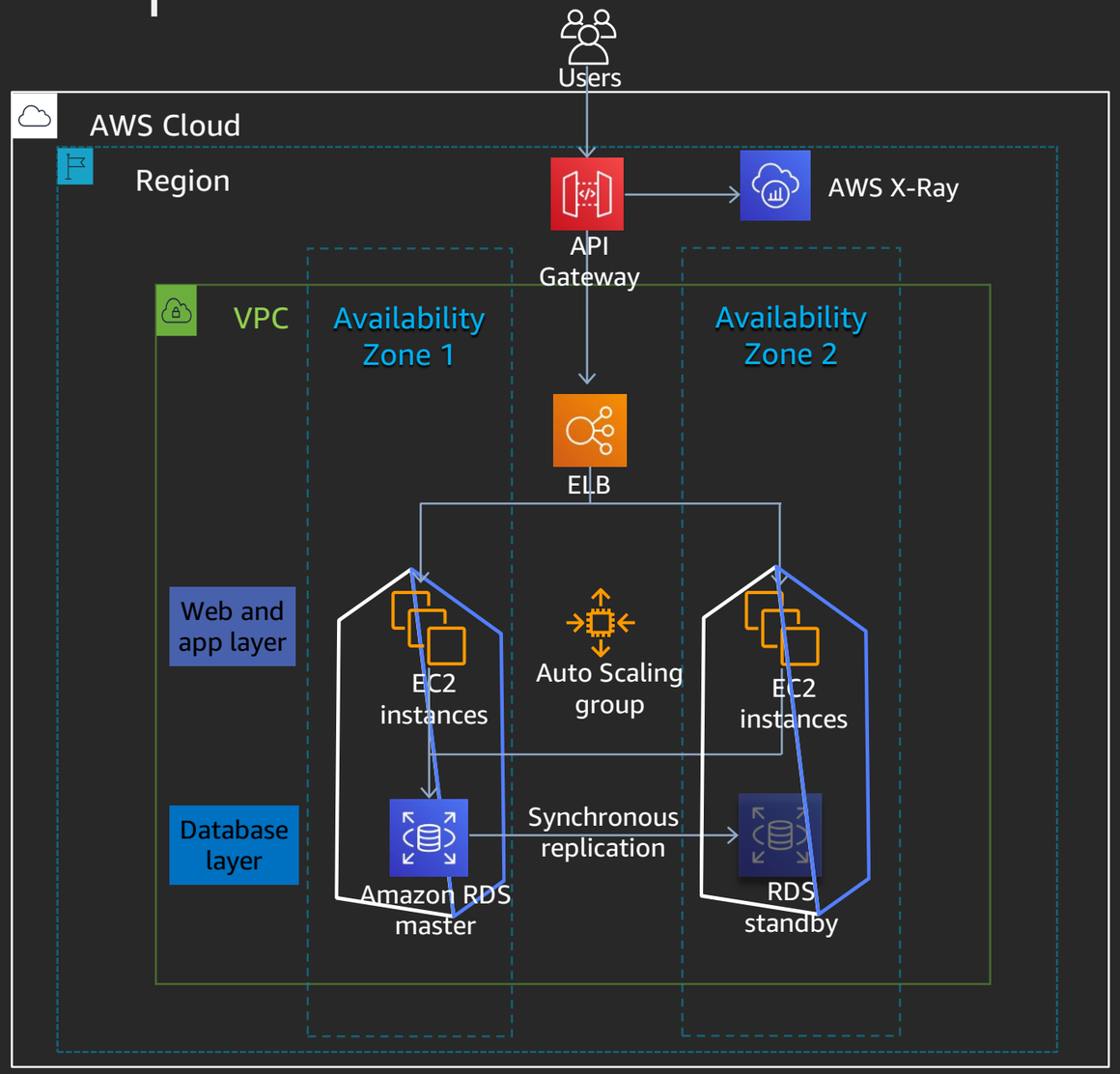


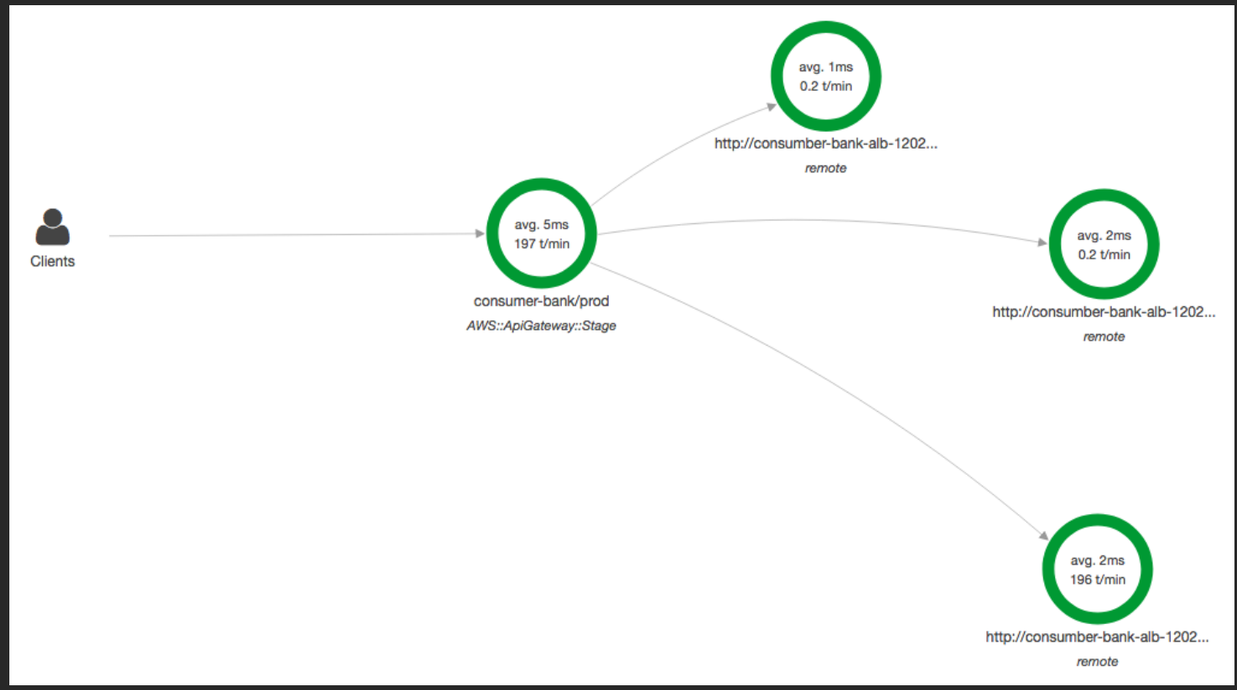
Step 3: Detect & measure hot spots



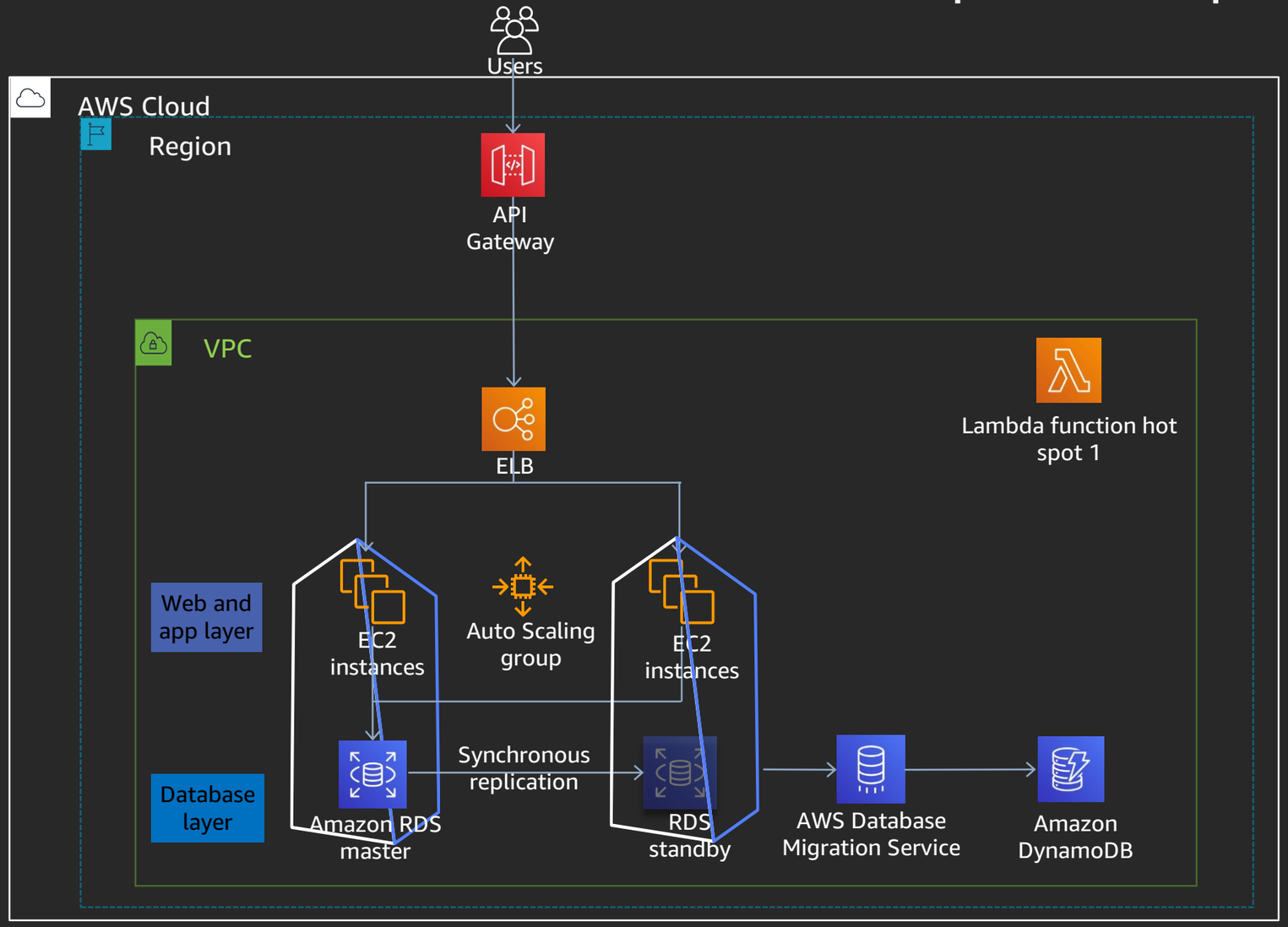


Step 3: Detect & measure hot spots





Step 4: Create AWS Lambda for top hot spot



Data access patterns

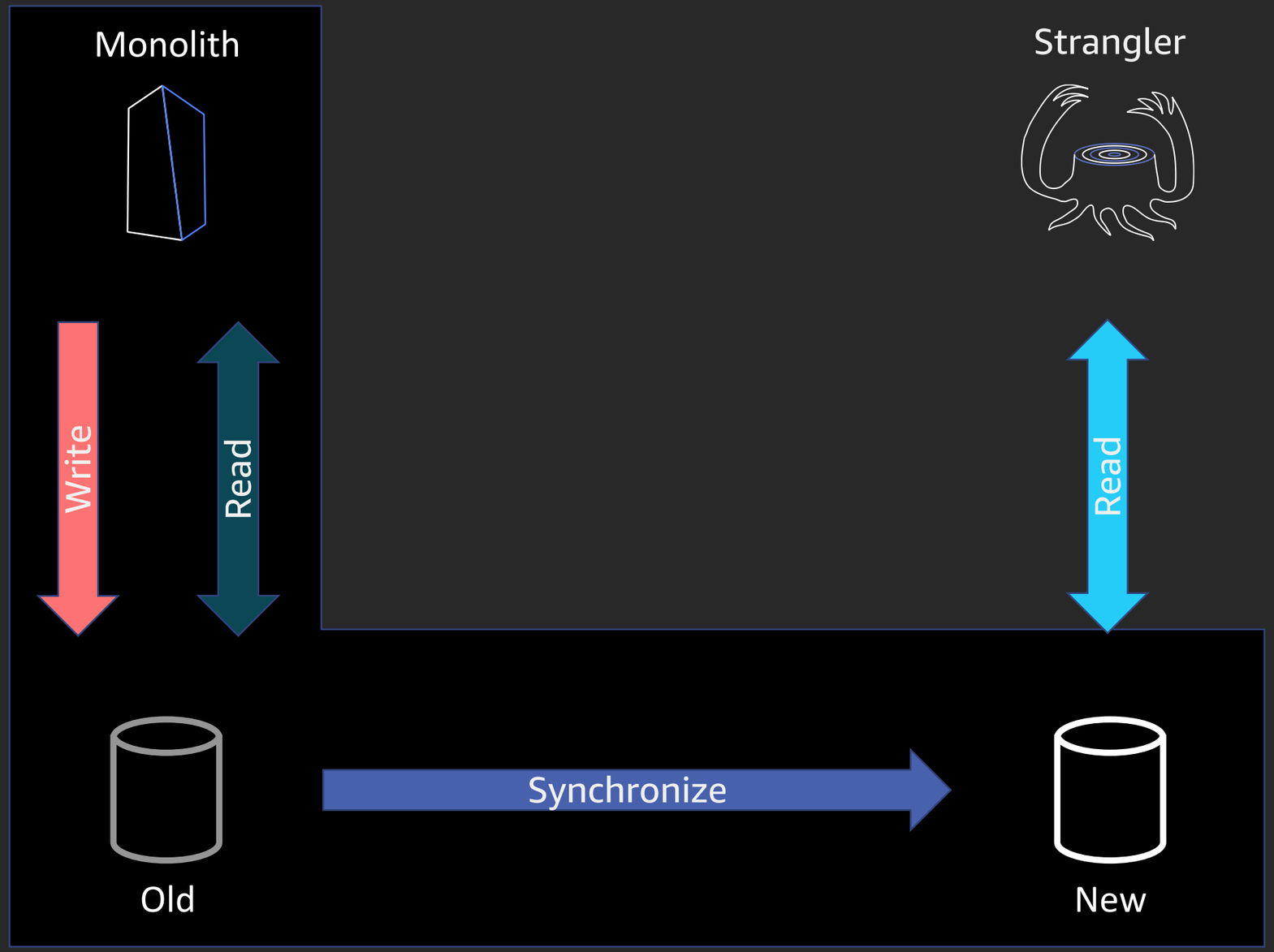
L-shaped pattern

Use case(s)

- Data structure stays consistent
- Data content stays consistent
- Eventual consistency acceptable
  - Batch synchronize
- Golden copy remains with old database

Synchronization technology(s)

- Triggers
- AWS Database Migration Service
- Generic batch process



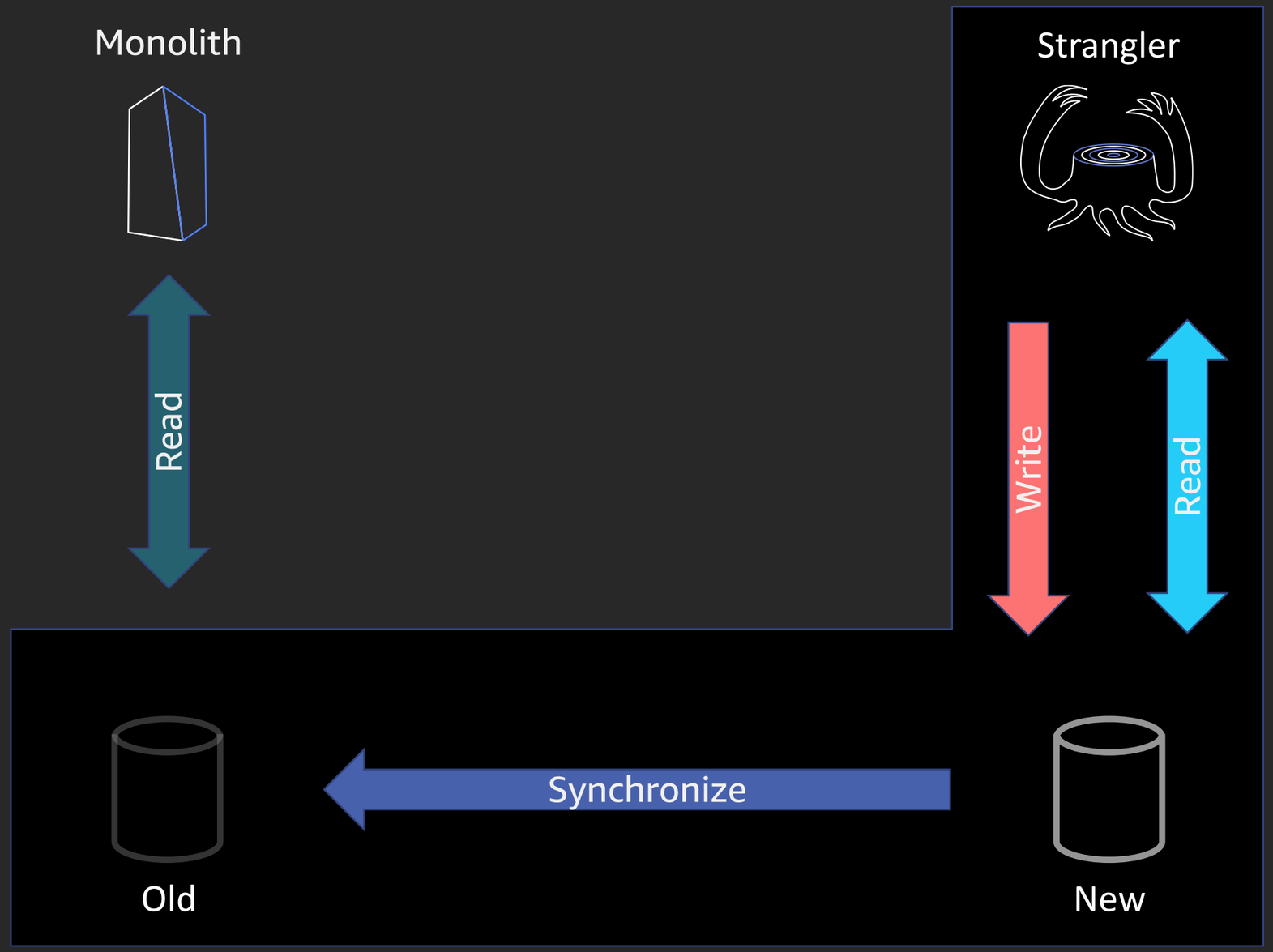
Backwards L-shaped pattern

Use case(s)

- Data structure differs
- Data content is expanded upon
- Eventual consistency acceptable
  - Batch synchronize
- Golden copy for expanded schema to new database and remains with existing data in the old database

Synchronization technology(s)

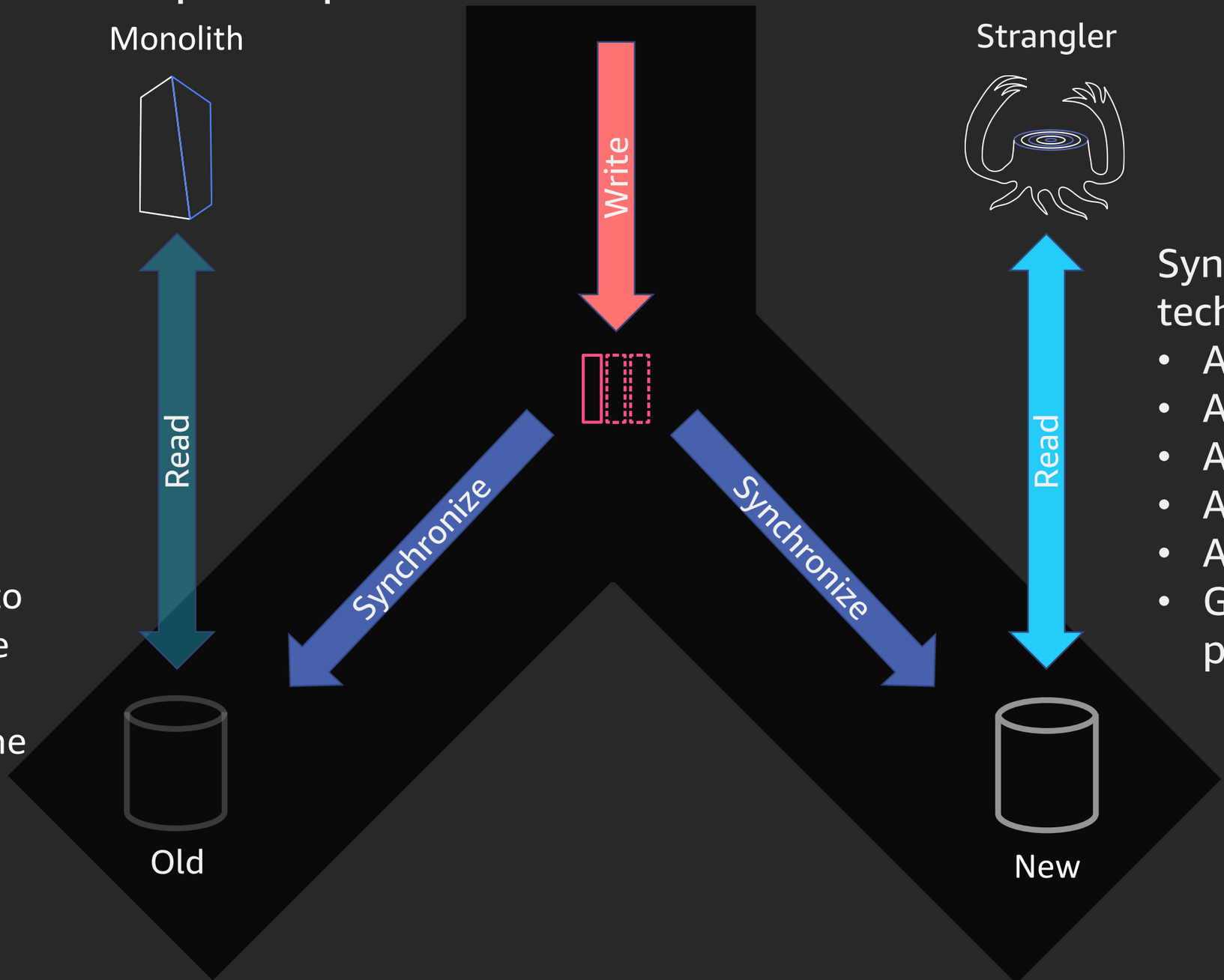
- Events
- AWS Database Migration Services
- Generic batch process



Inverted Y-shaped pattern

Use case(s)

- Data structure differs
- Data content is expanded upon
- Immediate consistency required
- Golden copy moves for the expanded data to the new database and remains with existing data in the old database



Synchronization technology(s)

- Amazon SQS
- Amazon Kinesis
- Amazon MQ
- Amazon MSK
- AWS Lambda
- Generic queue process

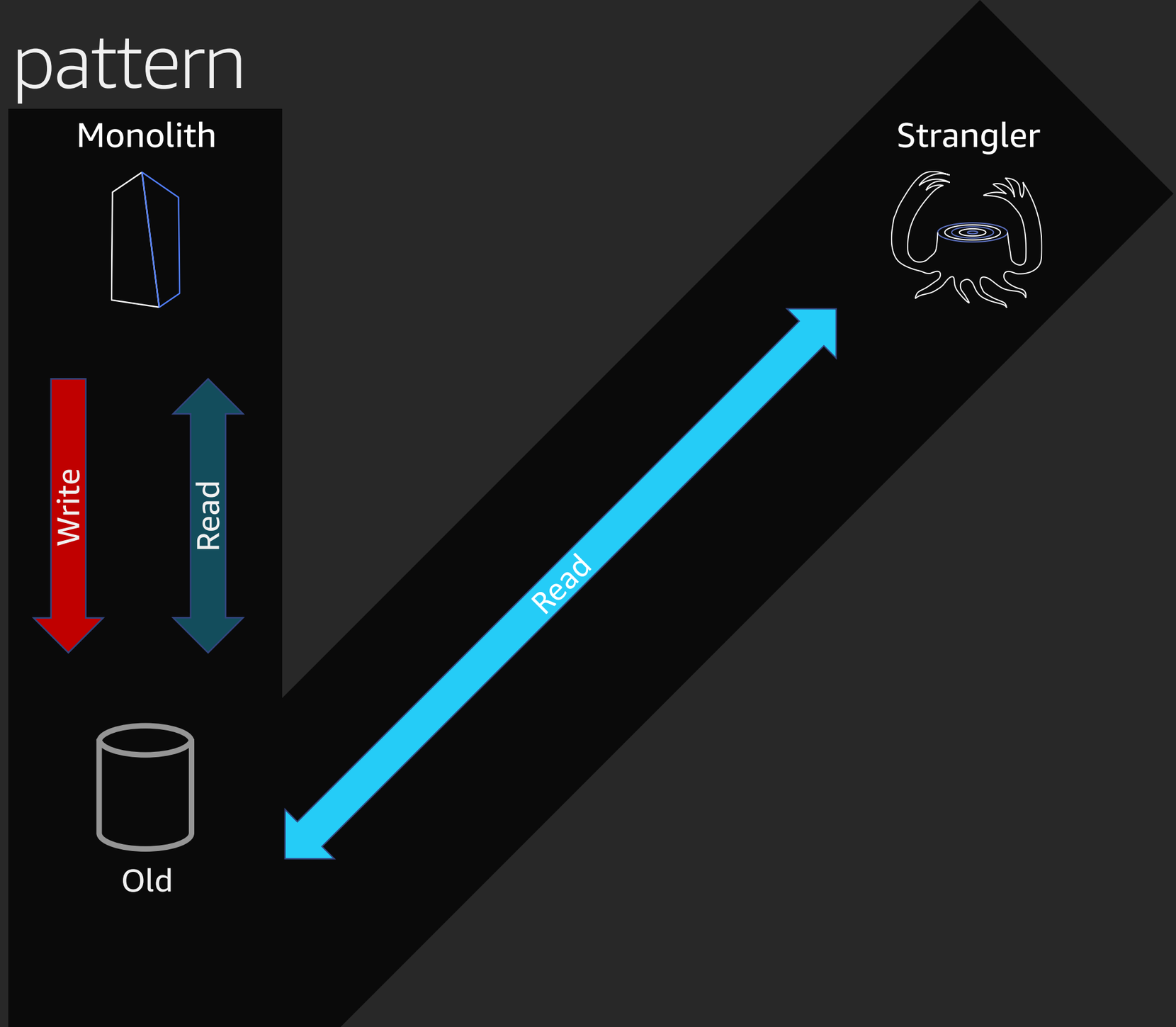
Slanted V-shaped pattern

Use case(s)

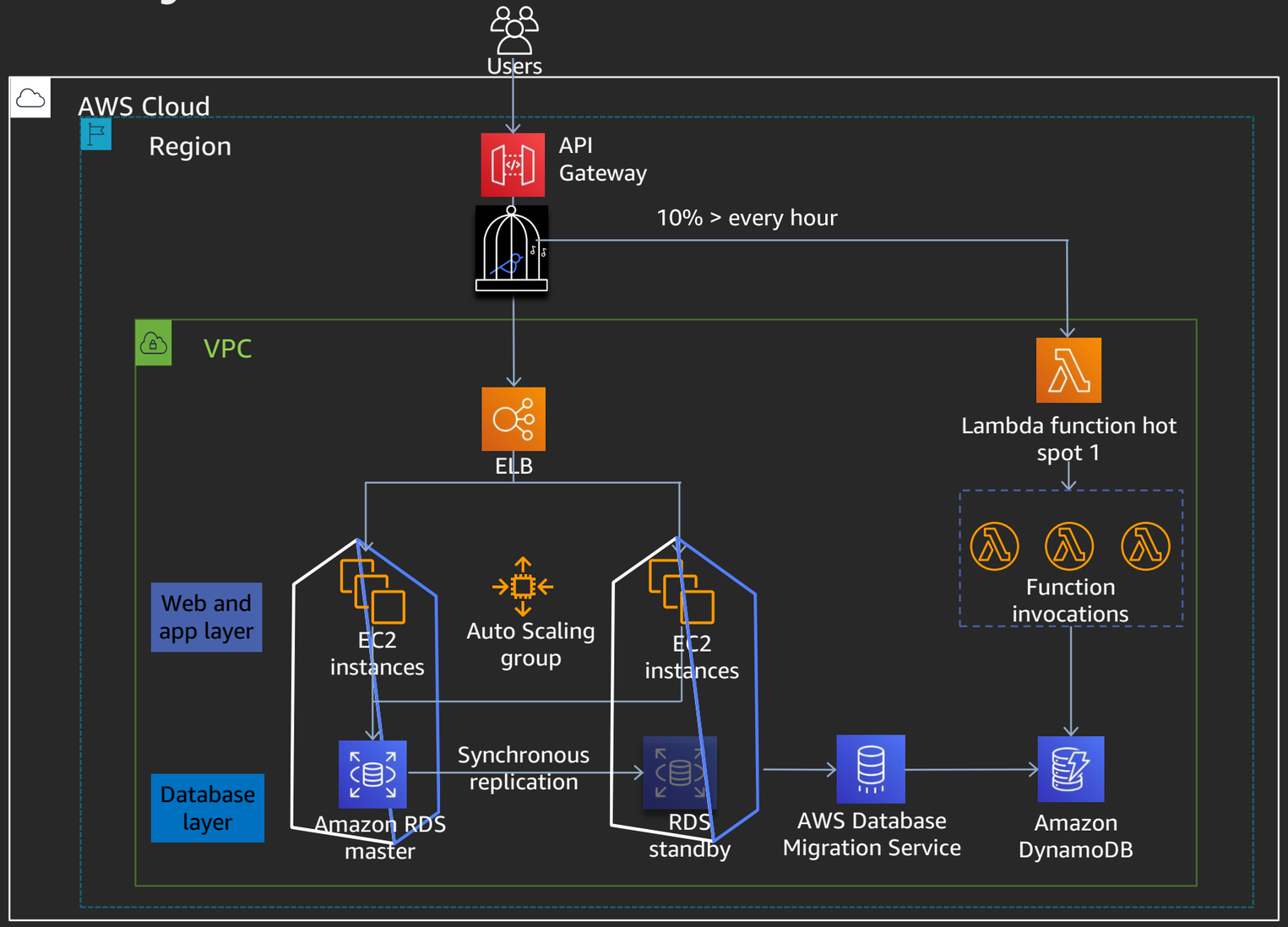
- Data structure remains consistent
- Data content remains consistent
- Immediate consistency required
- Golden copy remains with old, and only, database

Synchronize technology(s)

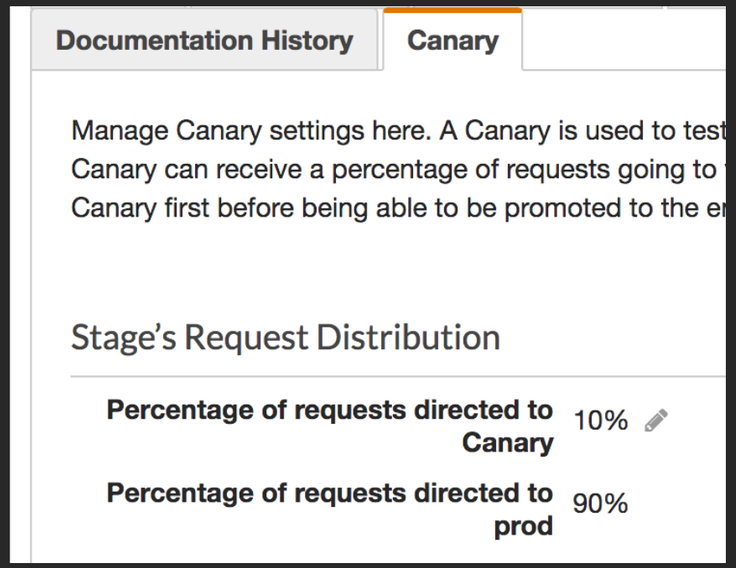
- Amazon ElastiCache
- Generic caching

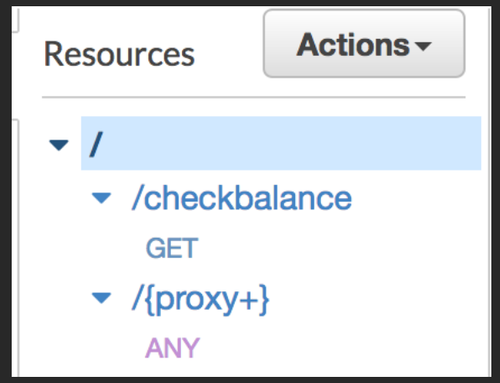


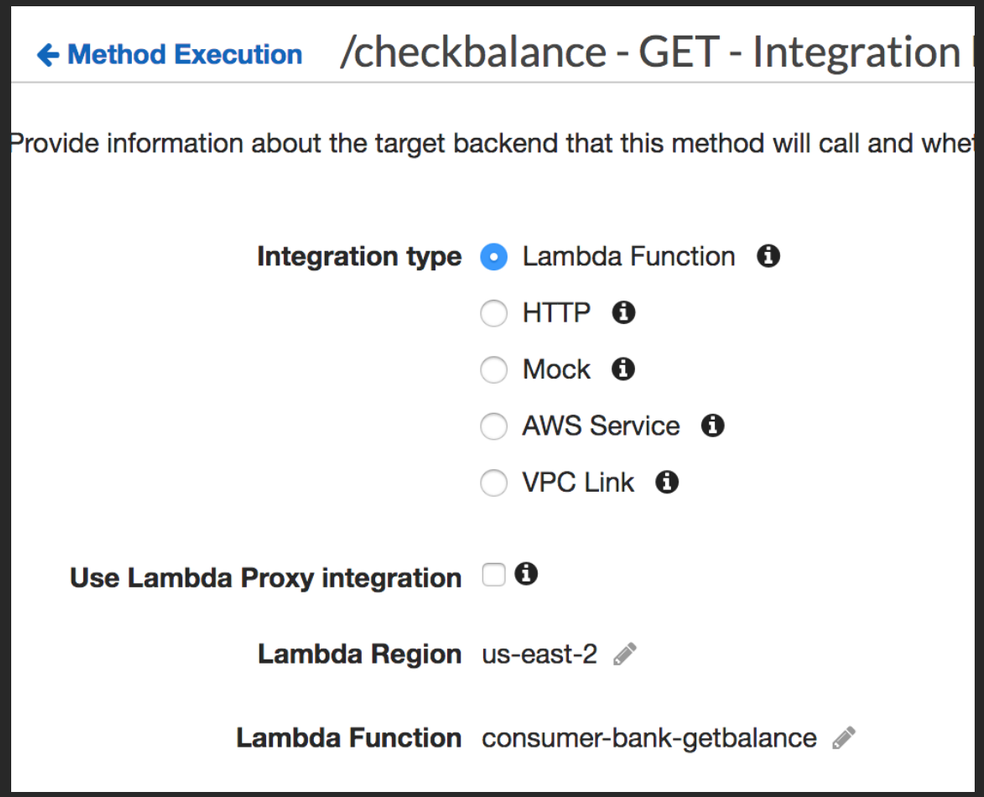
Step 5: Canary release



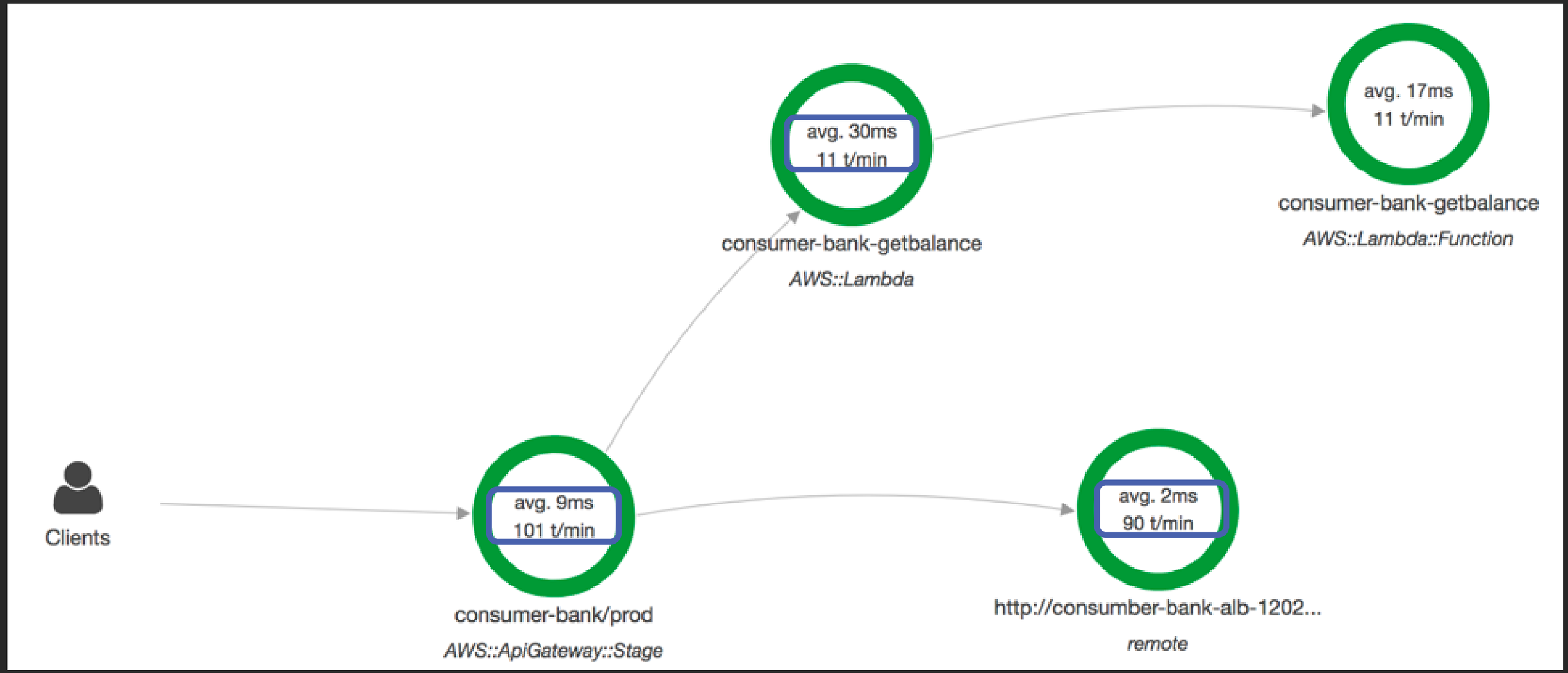
Step 5: Canary release – Setup



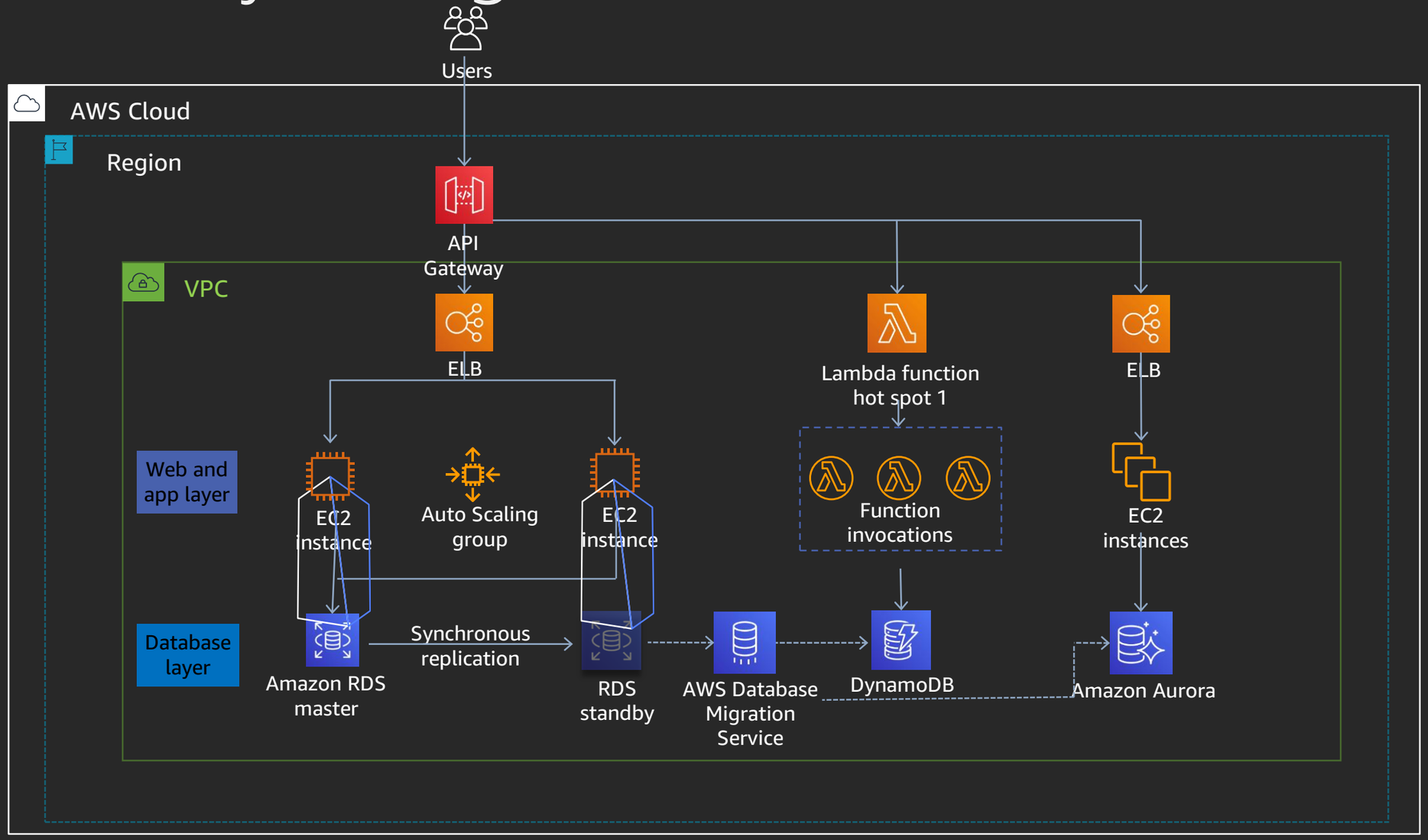




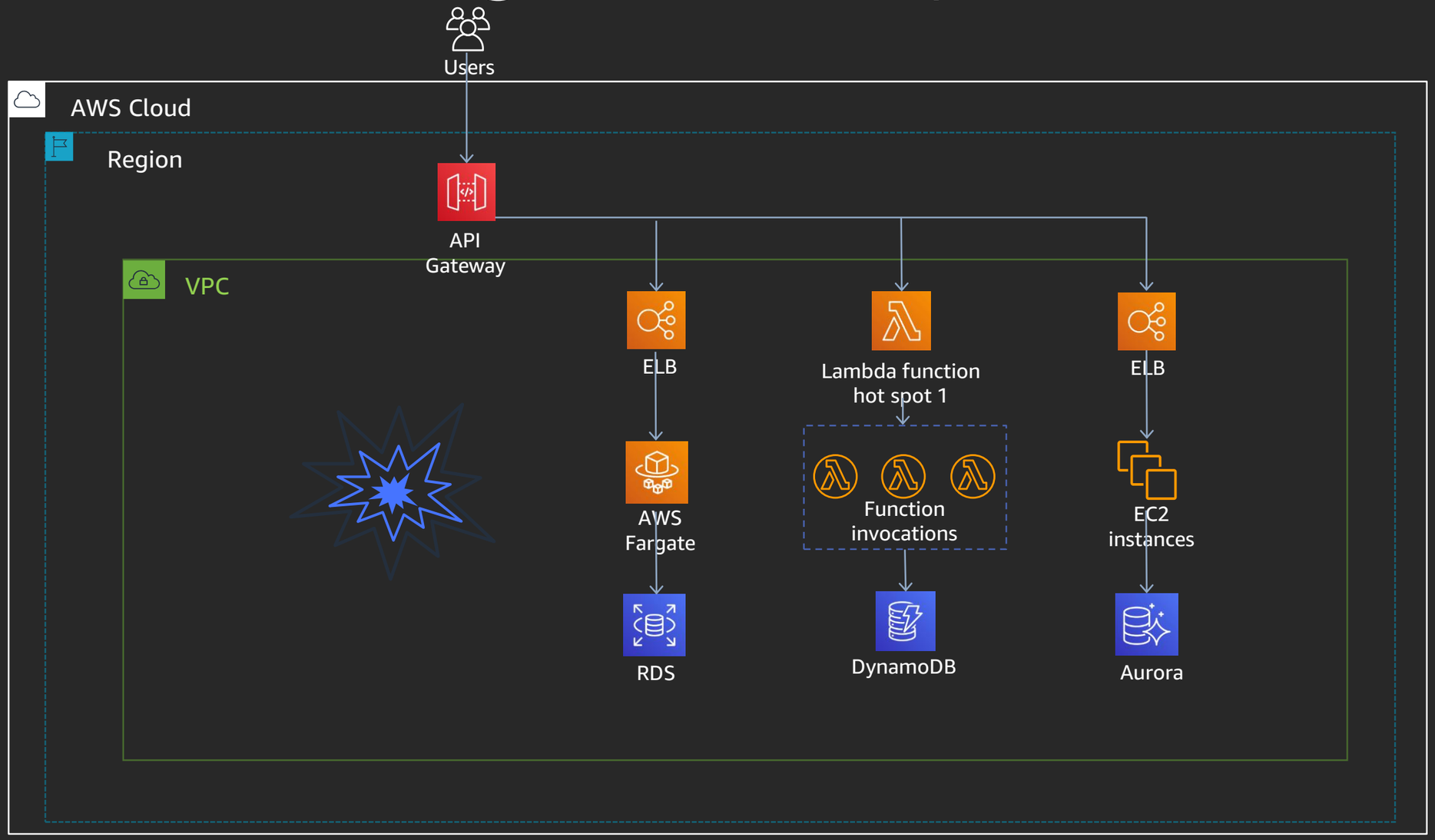
X-Ray canary service map



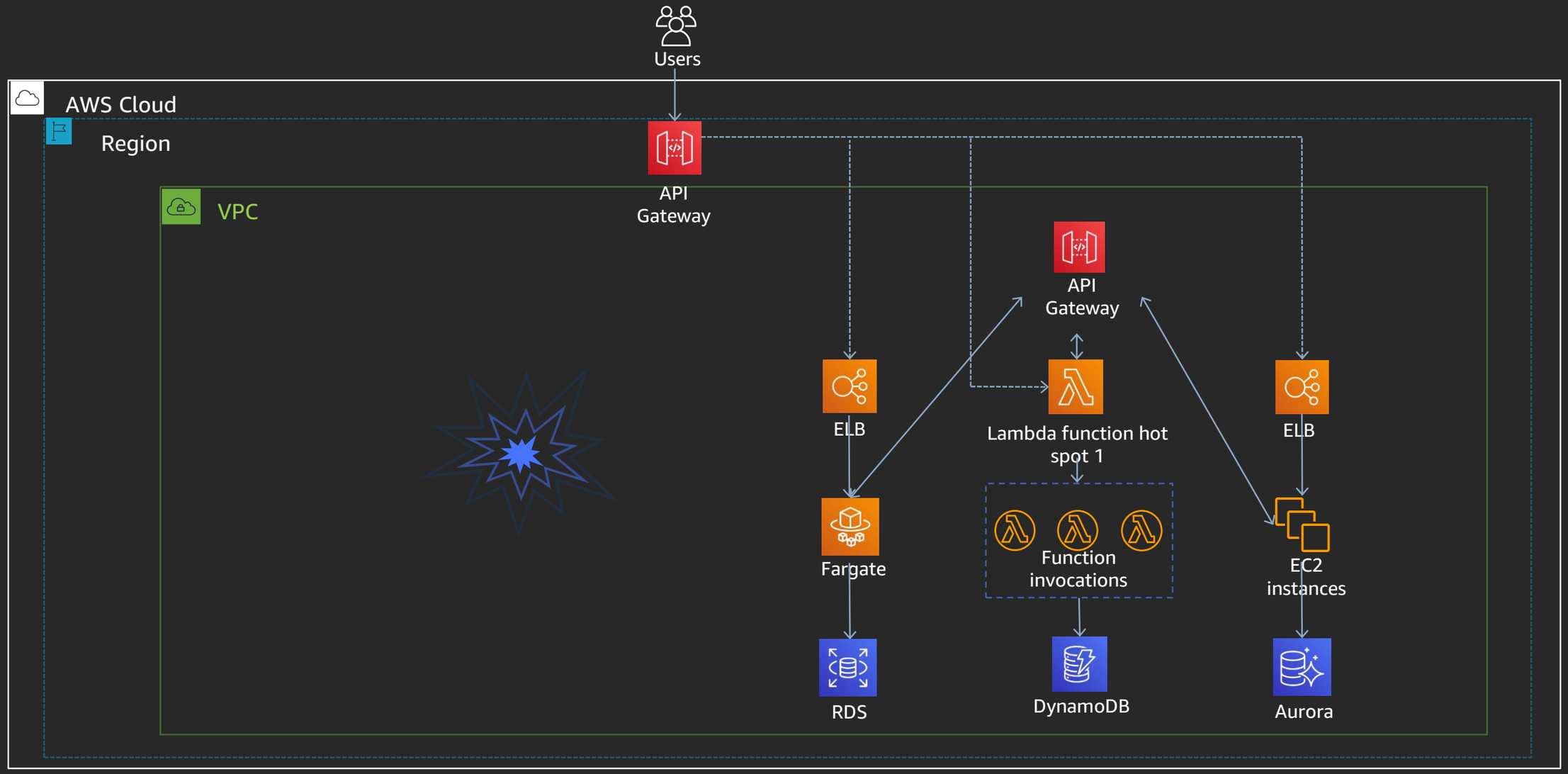
Step 6: Iteratively strangulate



Step 7: “Not with a bang but a whimper”



Step 7: "Not with a bang but a whimper"



Comparison of migration strategies (updated)

	Effort (time & cost)	Opportunity to optimize
Retire	N/A	N/A
Retain		N/A
Re-host		
Re-purchase		
Re-platform		
Re-architect		
Strangler		

Takeaways

- “Just move it”[1] (Jim Scholefield, Nike Global CIO) is often good advice
- Customers have seen 25%-30% saves without going “cloud native”
- Re-hosting offers lowest time to value and is a platform for more value
- Re-architecting might be appropriate for applications with a bright future
- Consider the strangler pattern to realize that value incrementally
- AWS services like API Gateway and Lambda are accelerators
- And, they don’t require you to “spend to save” – no extra infrastructure
- Leverage canaries and other modern techniques to validate new features

Additional resources

AWS Migration Whitepaper

https://d1.awsstatic.com/whitepapers/Migration/aws-migration-whitepaper.pdf

AWS Training: Advanced Developing on AWS
https://www.aws.training/training/schedule?courseId=36896

AWS re:Invent 2016: From Monolithic to Microservices: Evolving Architecture Patterns (ARC305)

https://www.youtube.com/watch?v=oRIYtOsAlzk

AWS re:Invent 2018: Mainframe Modernization with AWS: Patterns and Best Practices (GPSTEC305)

https://www.youtube.com/watch?v=AJ88gY1w9NA

Strangling the Monolith With a Data-Driven Approach –
David Julia, Simon Duffy (Pivotal Software, Inc.)

https://www.youtube.com/watch?v=P7mM6TA75x4

Thank you!

AWS re:Invent

aws

Please complete the session survey in the mobile app.