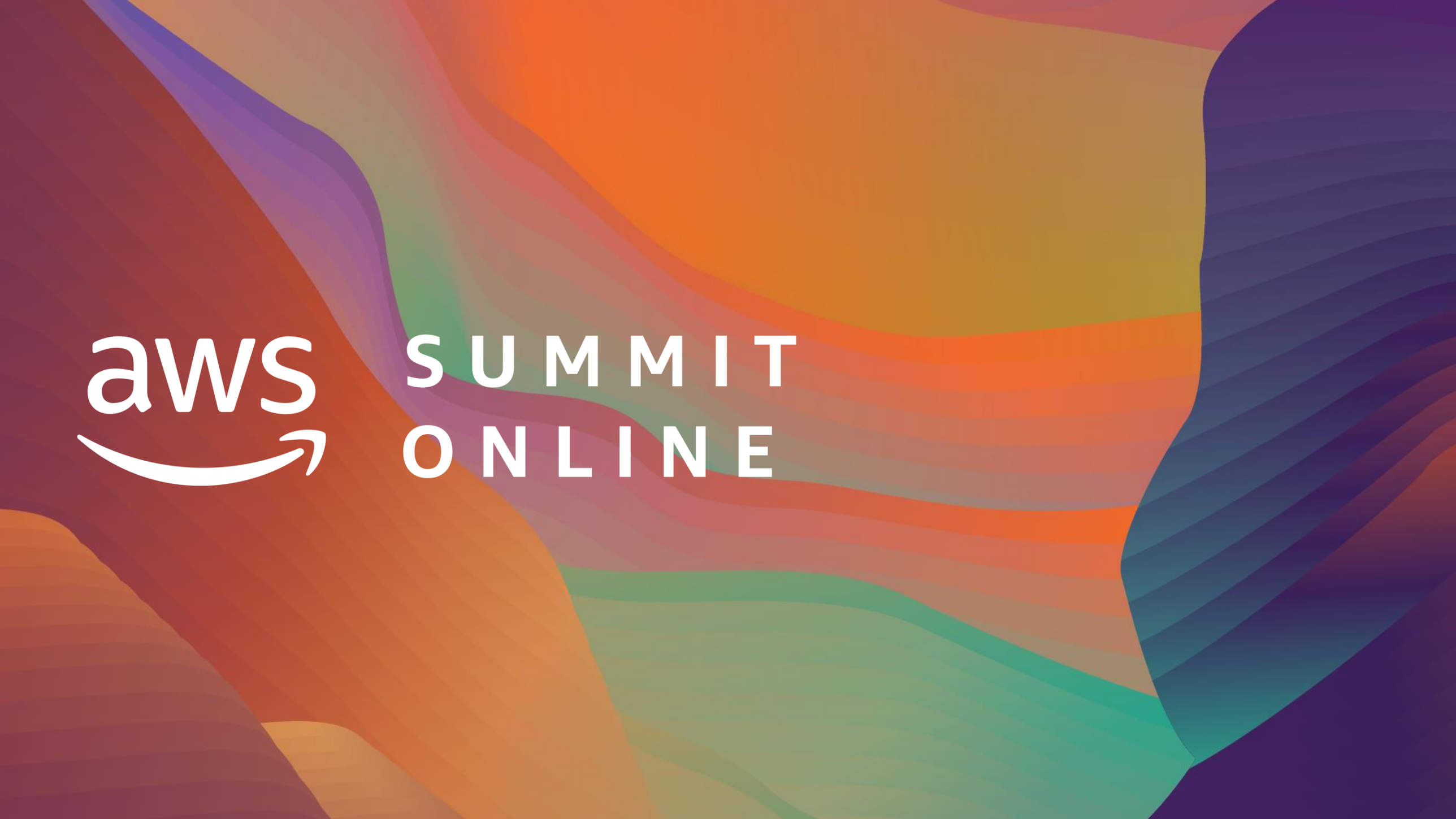

aws
SUMMIT
ONLINE

BUI09

Application integration patterns for microservices: Fan-out strategies

Chris Modica

Enterprise Solutions Architect
Amazon Web Services

aws SUMMIT ONLINE

What we'll cover in this session

Common integration anti-patterns

Advanced integration patterns:

- Publisher/Subscriber
- Message filter pattern
- Topic-queue-chaining pattern

Coding - show you how to implement these patterns

Where to learn more

"If your application is cloud-native or large-scale, or distributed, and doesn't include a messaging component, that's probably a bug".

Tim Bray

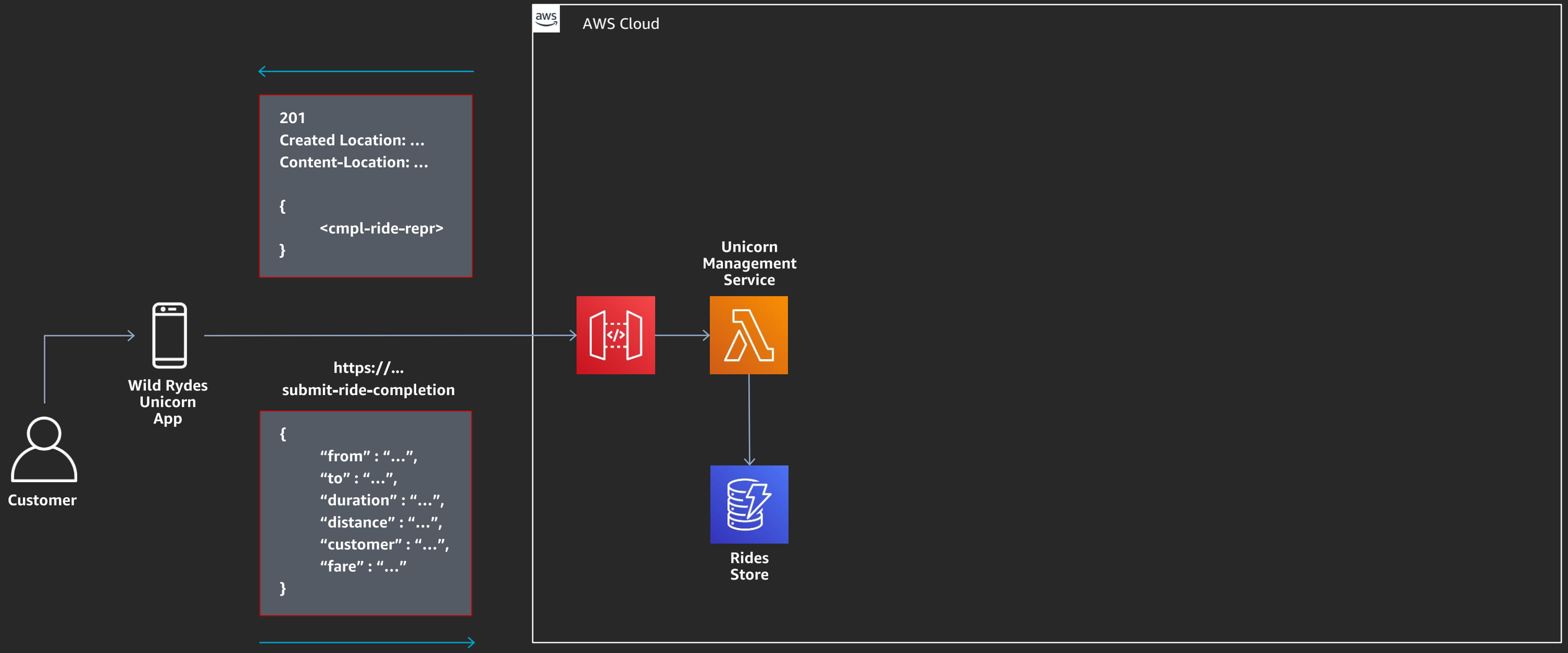
Distinguished Engineer

AWS Messaging, Workflow Management

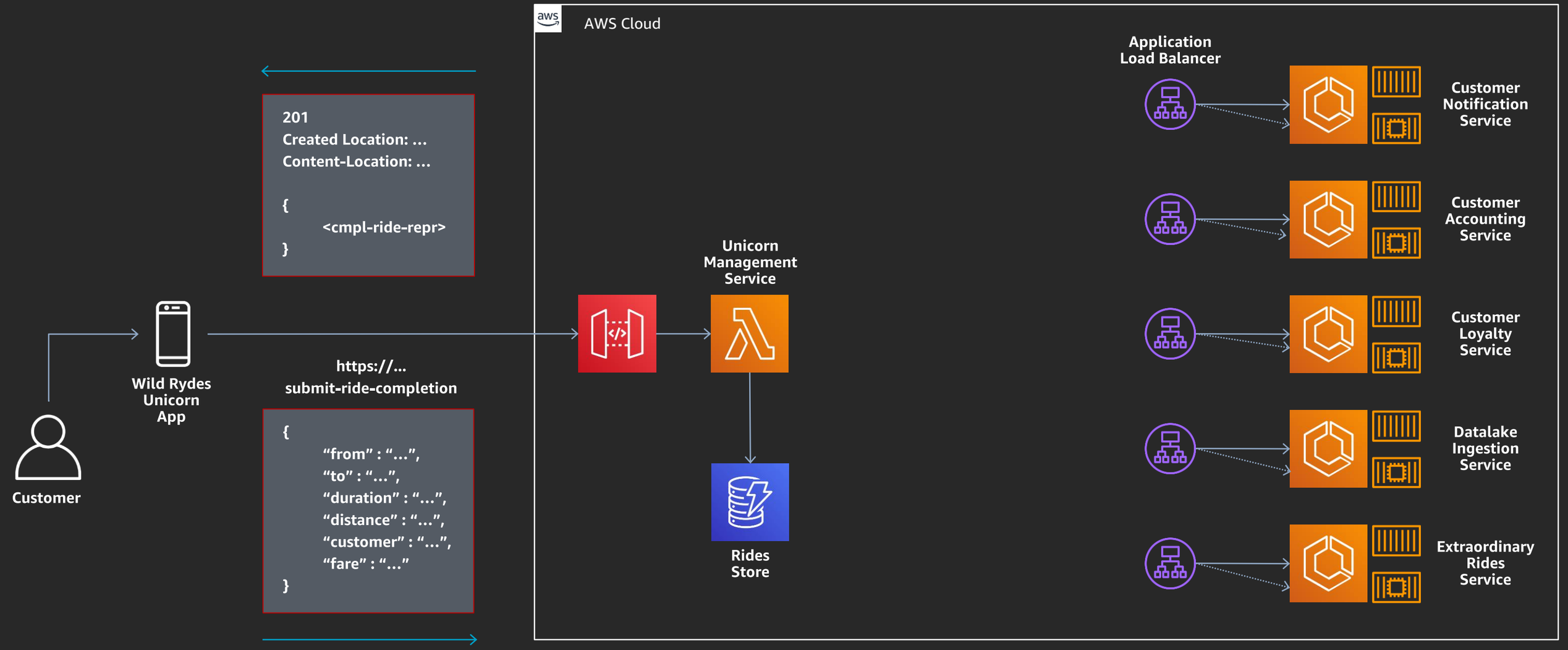
Example application: Wild Rydes

www.wildrydes.com

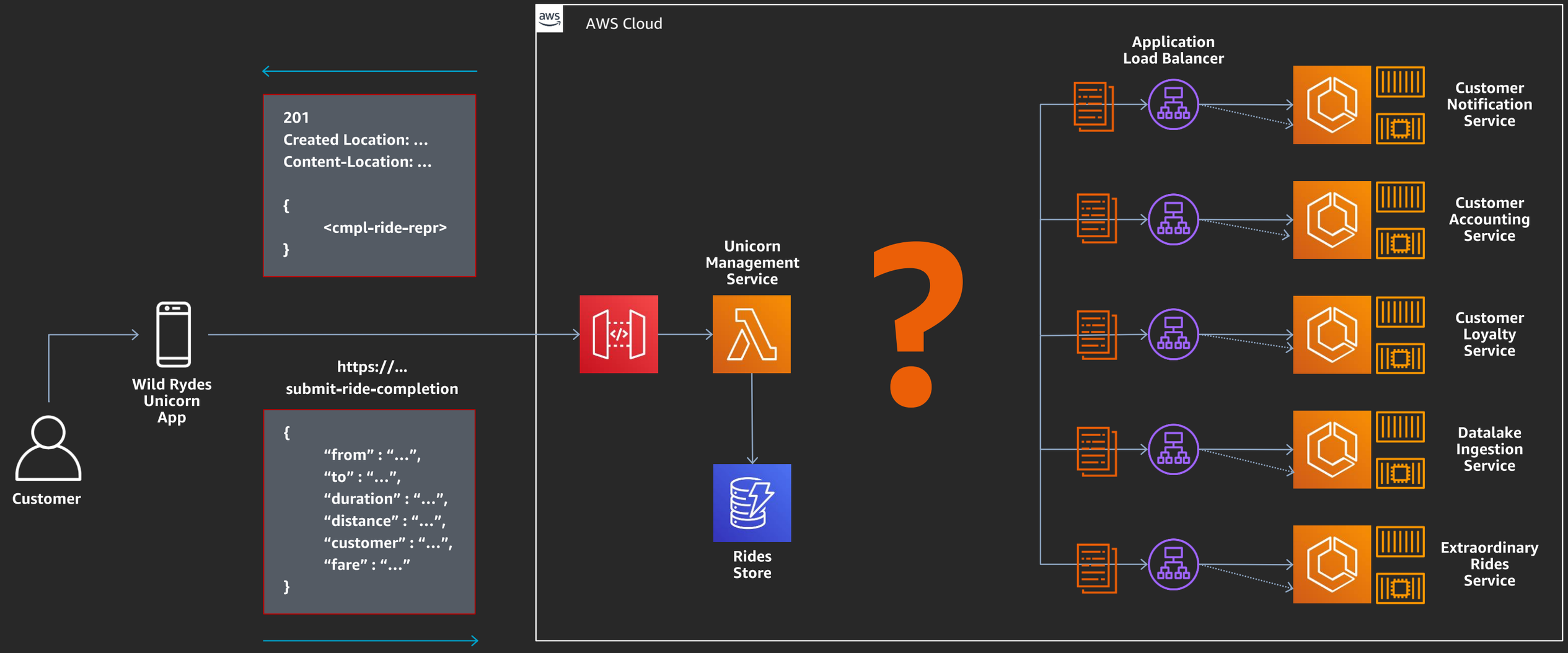
Logical architecture for the use case



Logical architecture for the use case

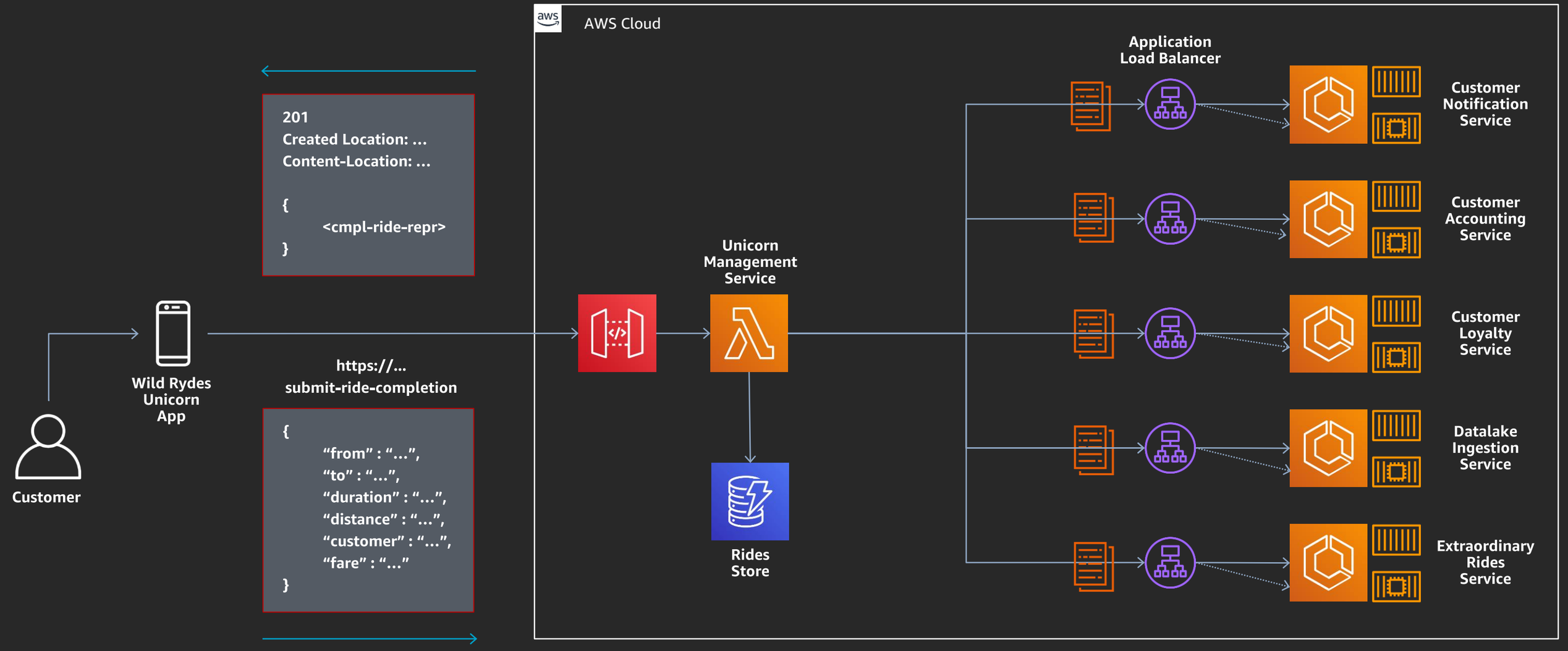


How can we integrate these services?

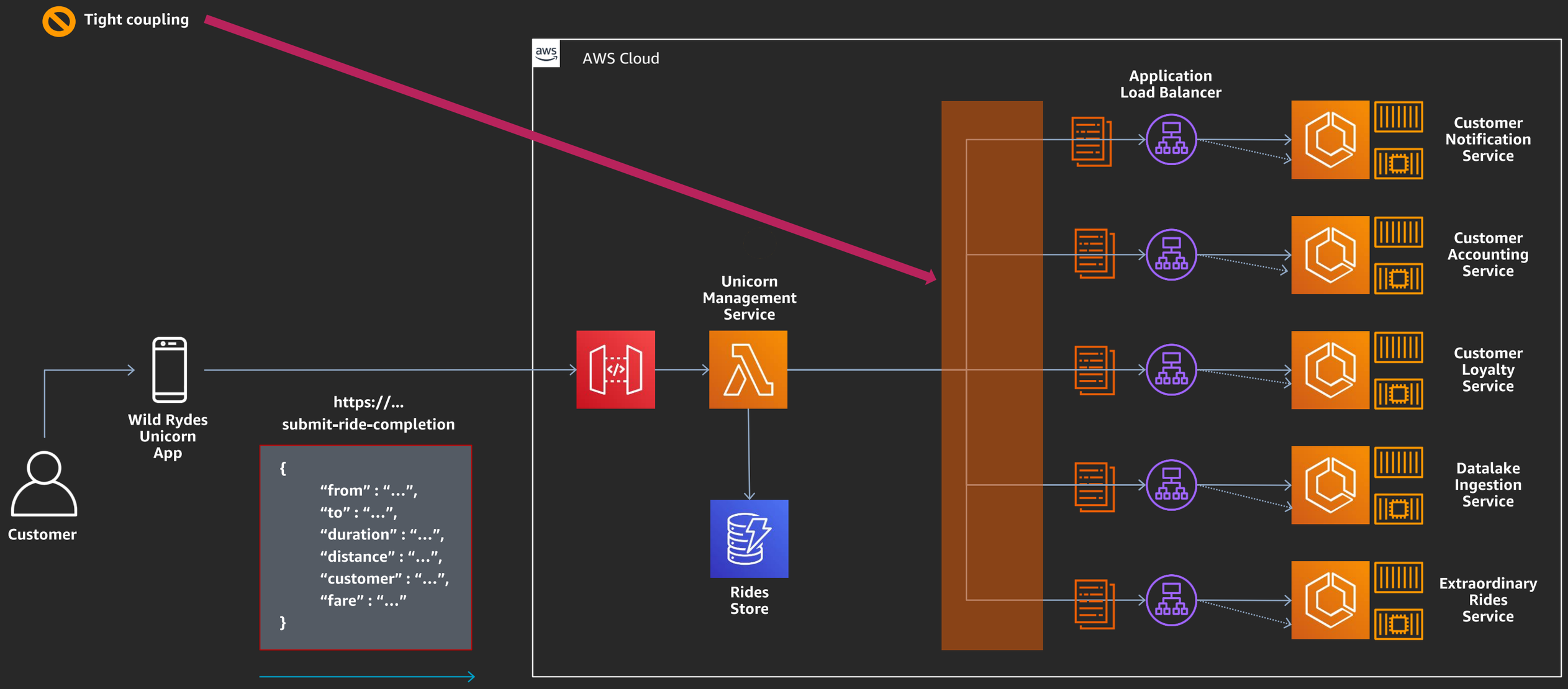


Option 1:
Integration via a database

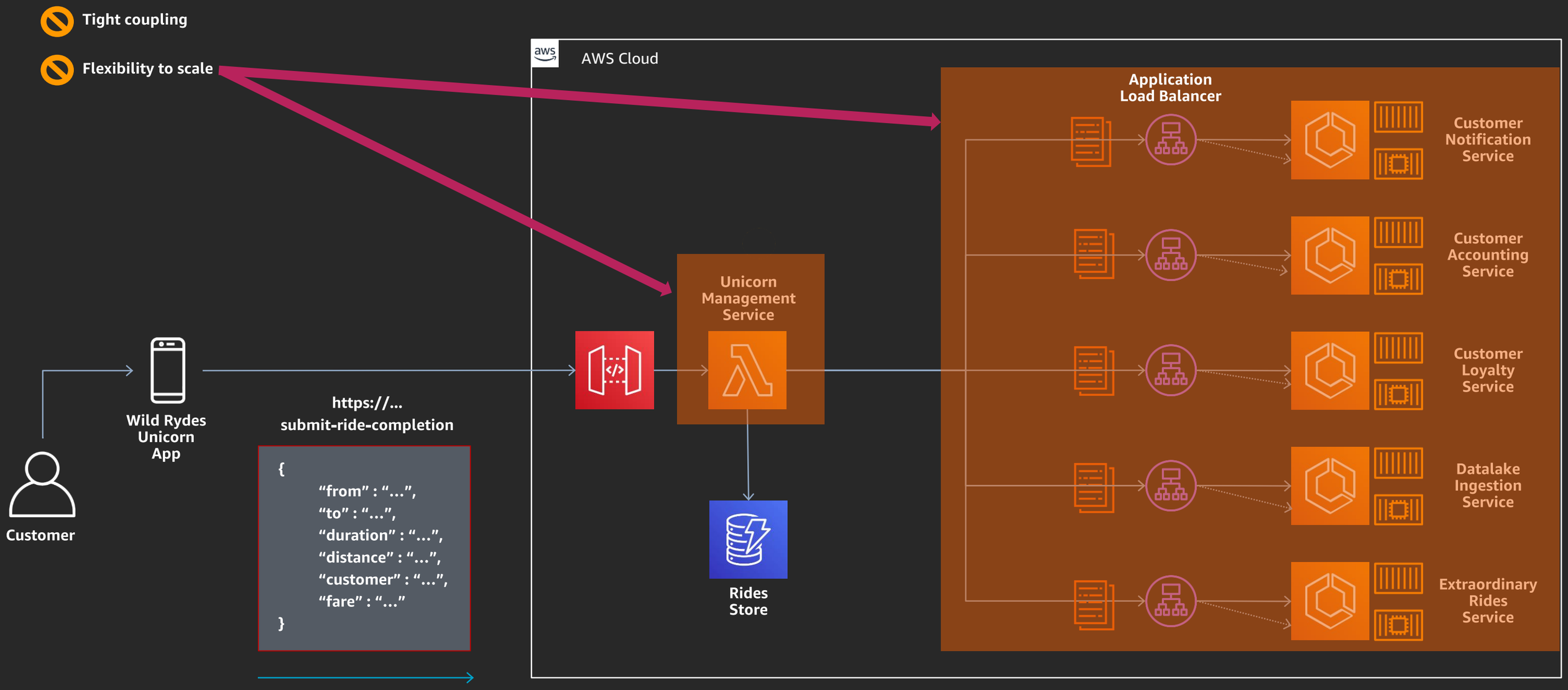
Logical architecture: Integration via a database



Logical architecture: Integration via a database

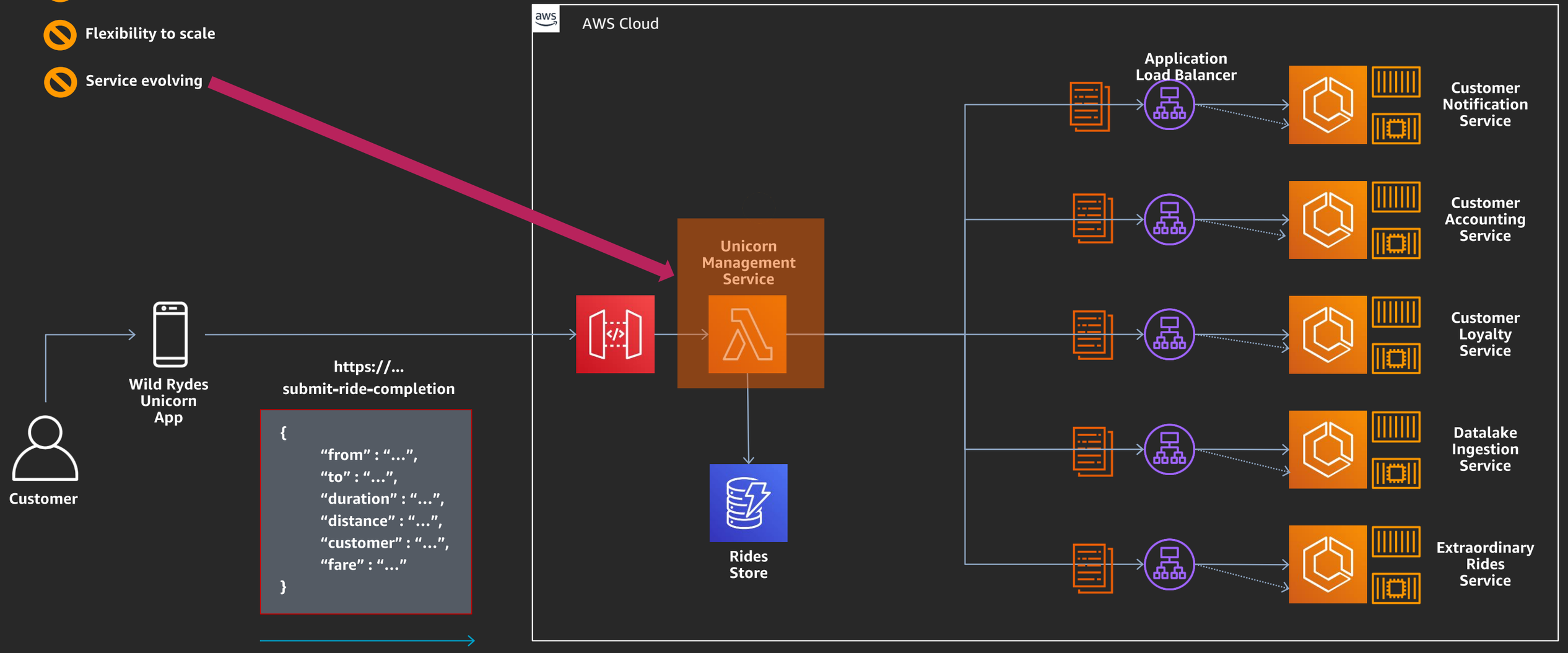


Logical architecture: Integration via a database



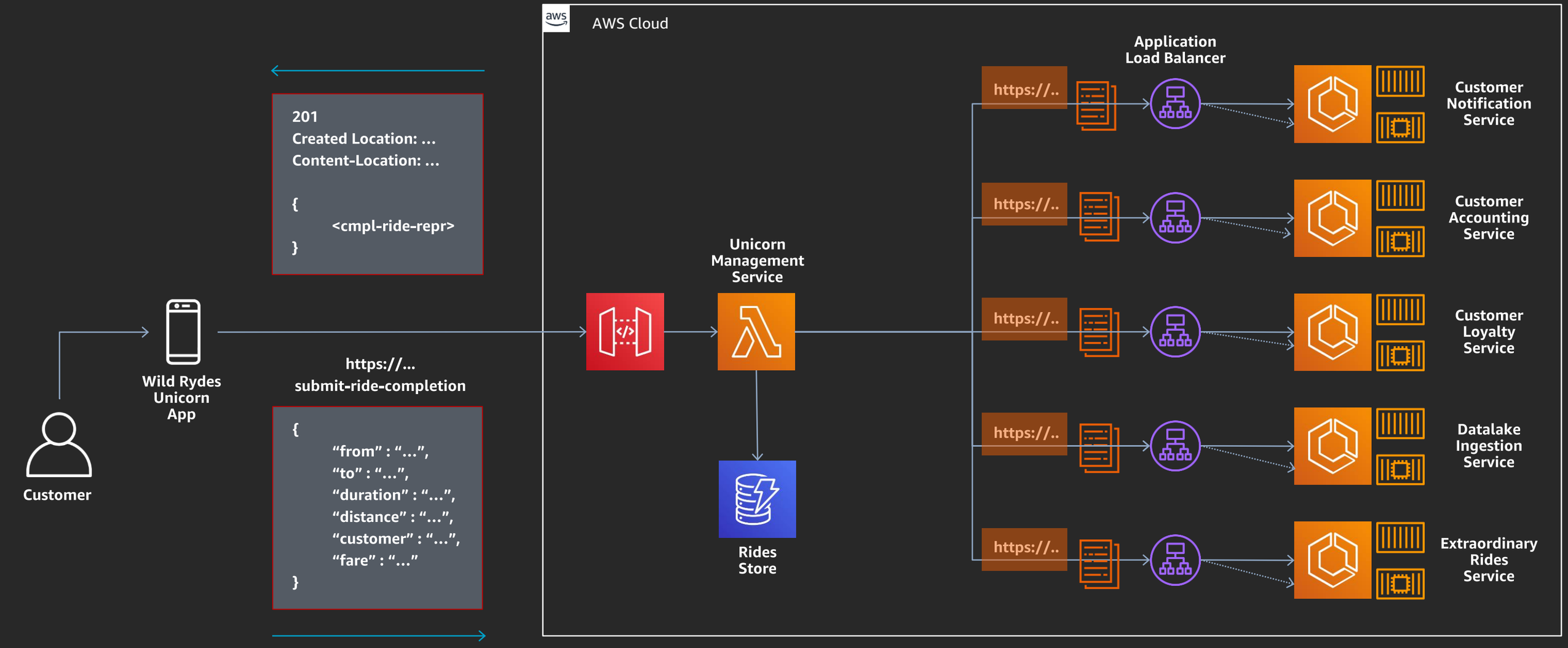
Logical architecture: Integration via a database

- Tight coupling
- Flexibility to scale
- Service evolving

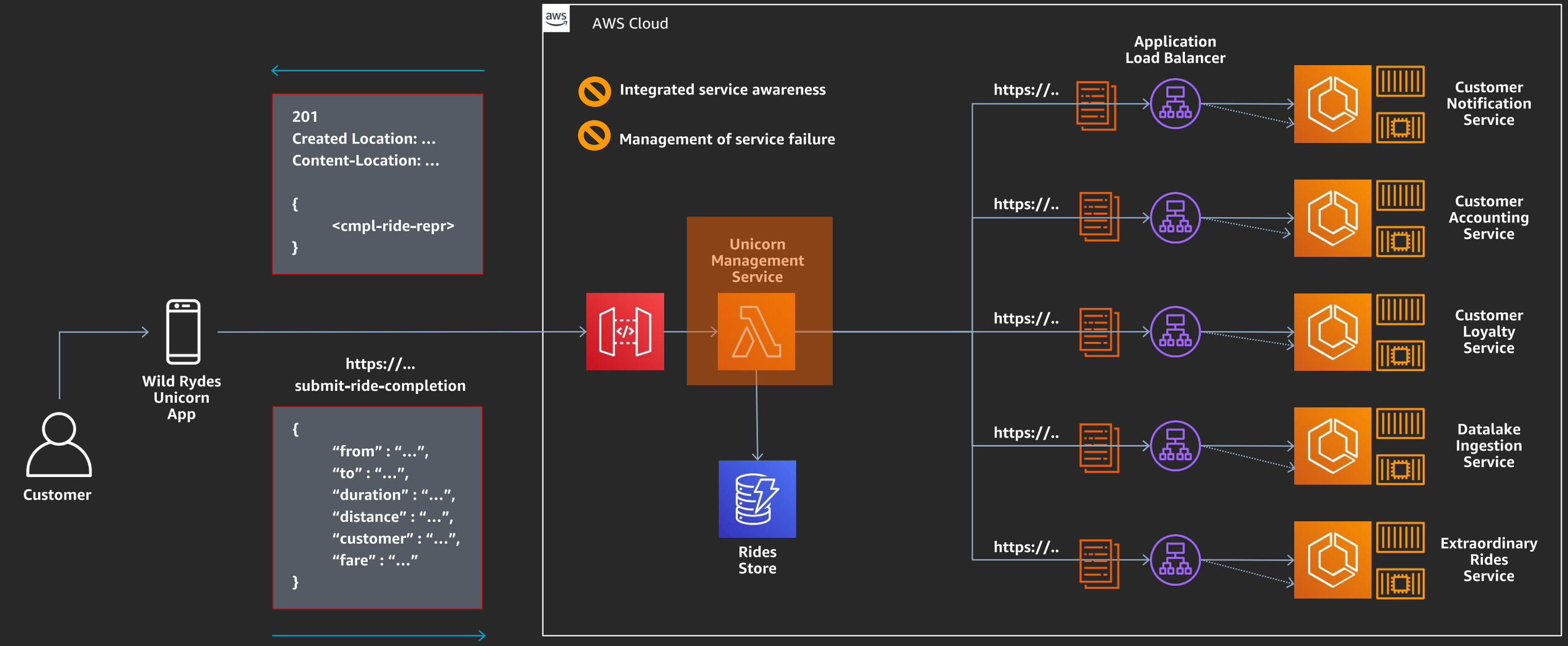


Option 2:
Integration via REST APIs

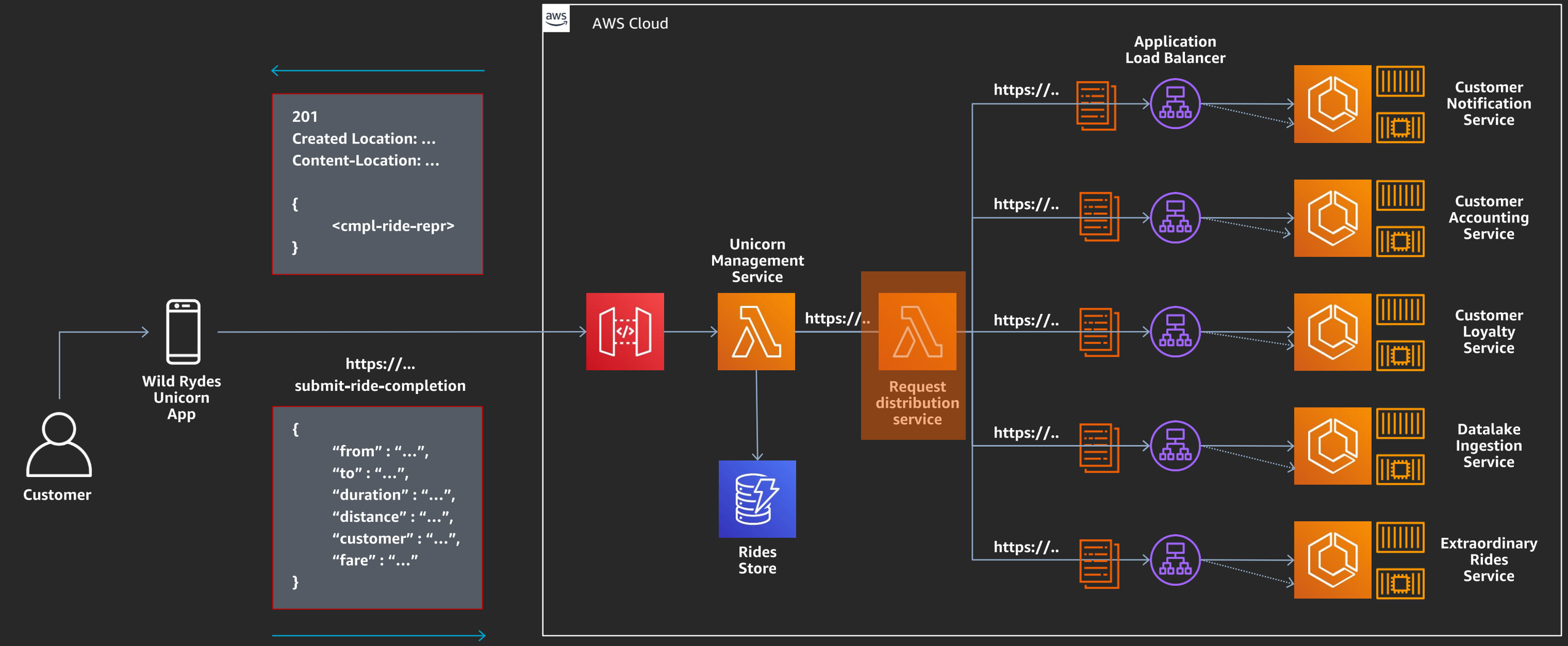
Logical architecture: Integration via REST APIs



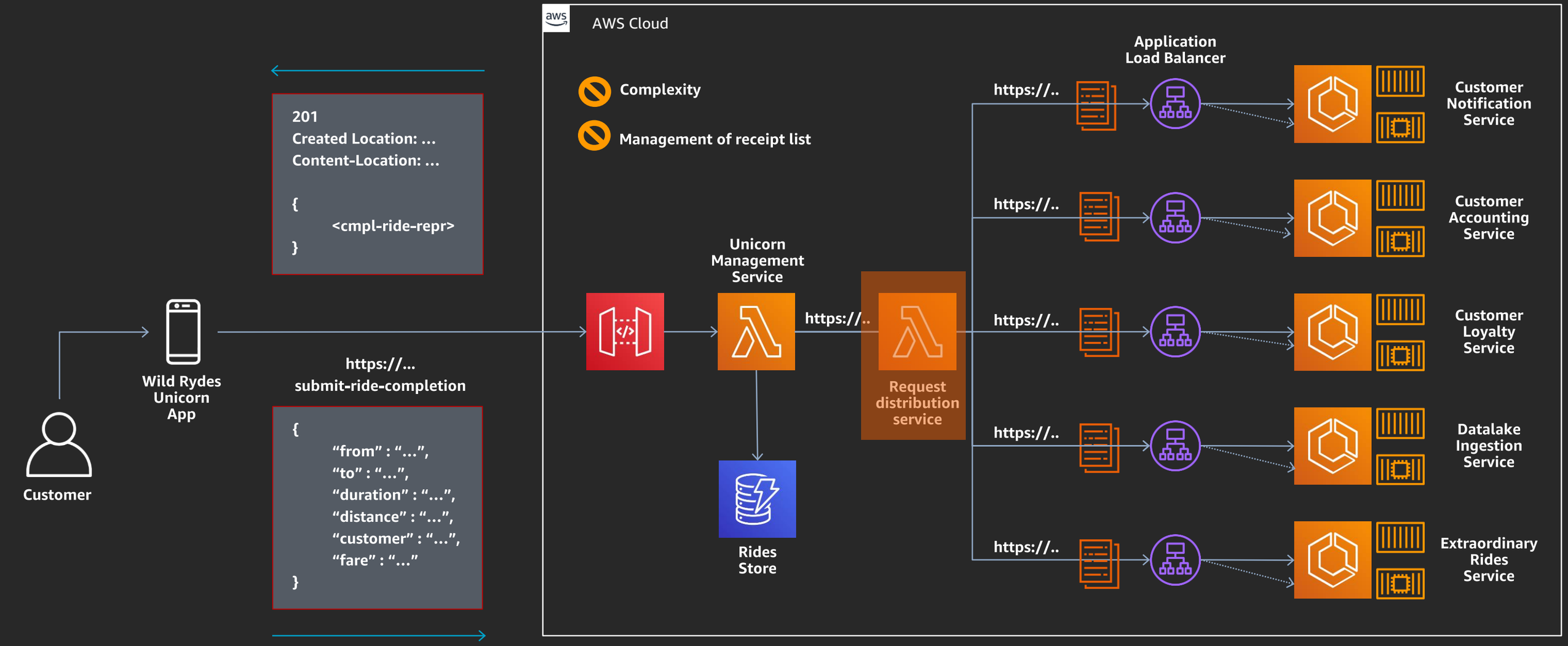
Logical architecture: Integration via REST APIs



Logical architecture: Integration via REST APIs



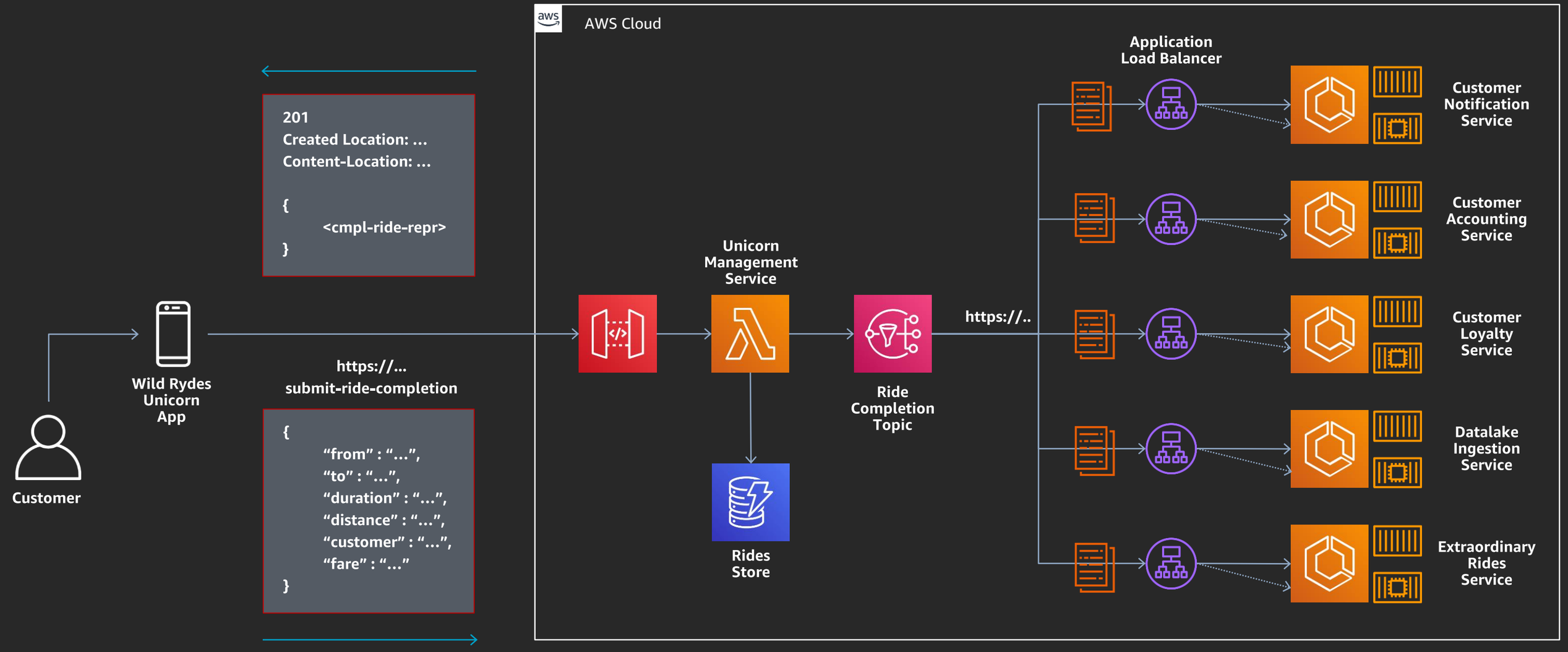
Logical architecture: Integration via REST APIs



Option 3:
Integration via messaging

aws SUMMIT ONLINE

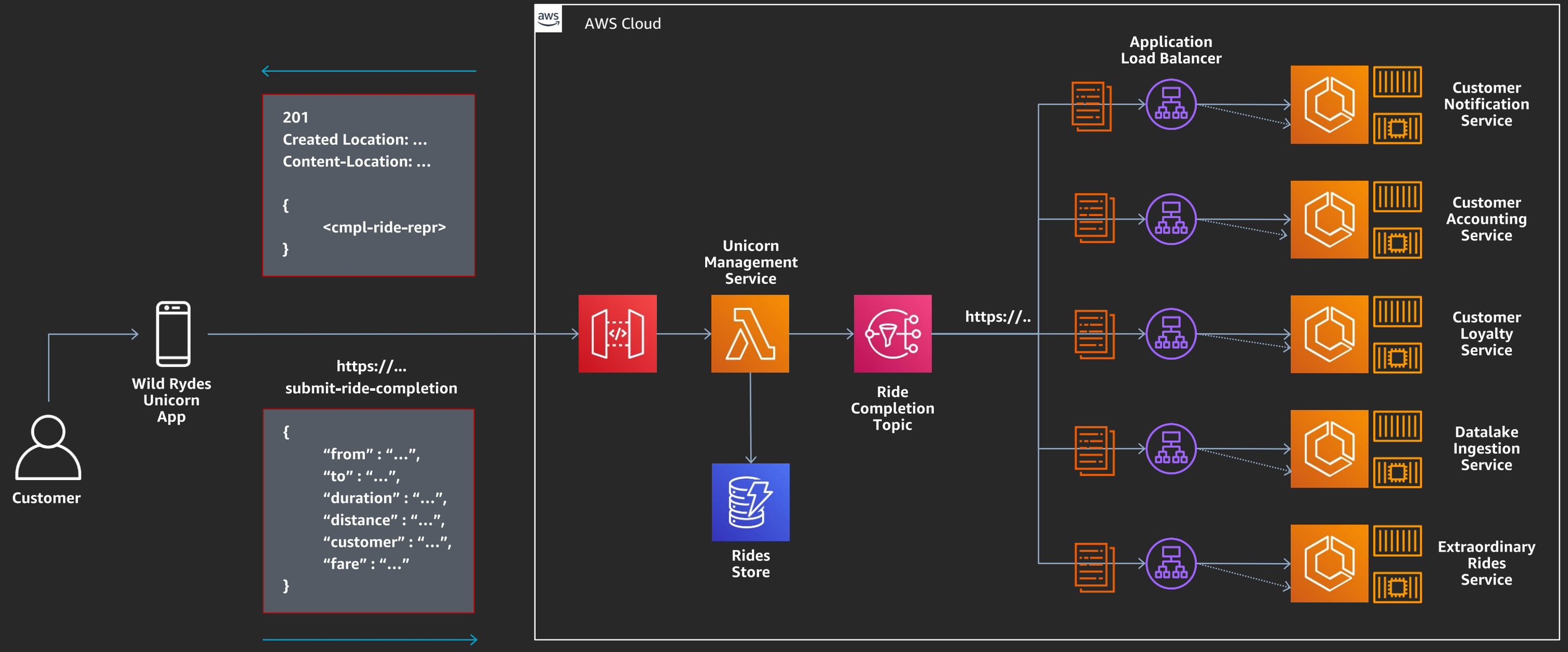
Logical architecture: Pub/Sub (Fan-out) pattern



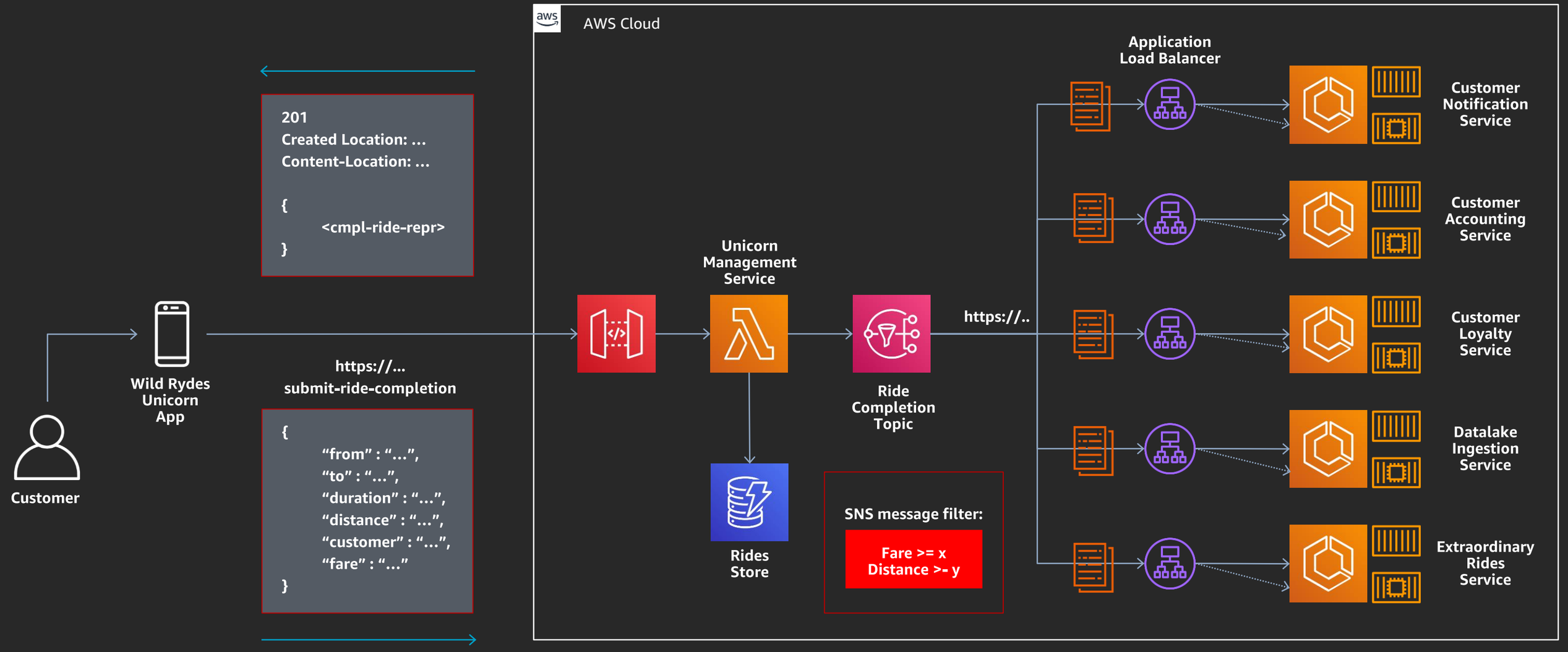
Live Demo: 1

aws SUMMIT ONLINE

Logical architecture: Pub/Sub (Fan-out) pattern



Logical architecture: Fan-out and Message Filtering



Integration via messaging: Message filter pattern

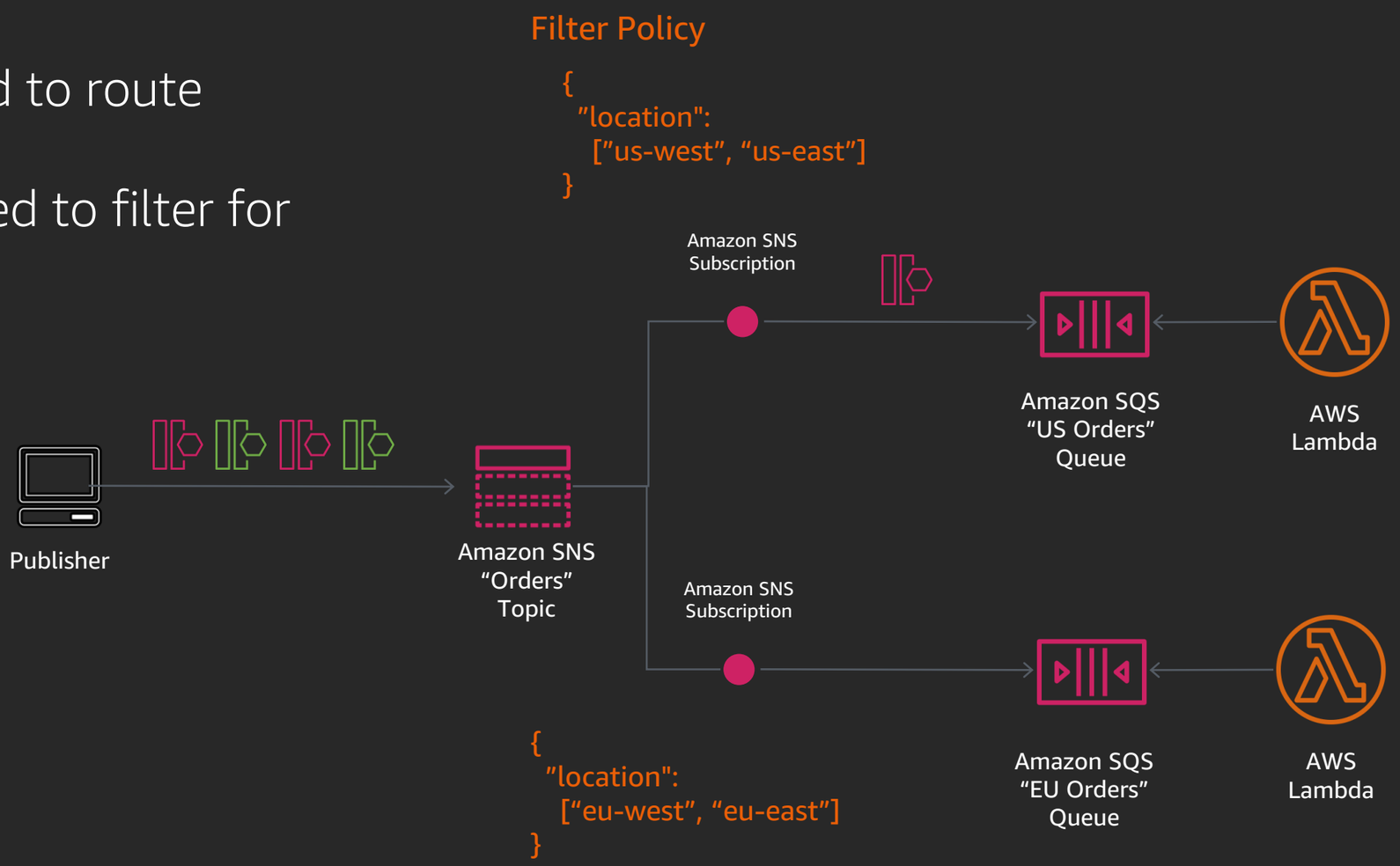
aws SUMMIT ONLINE

Amazon SNS message filters

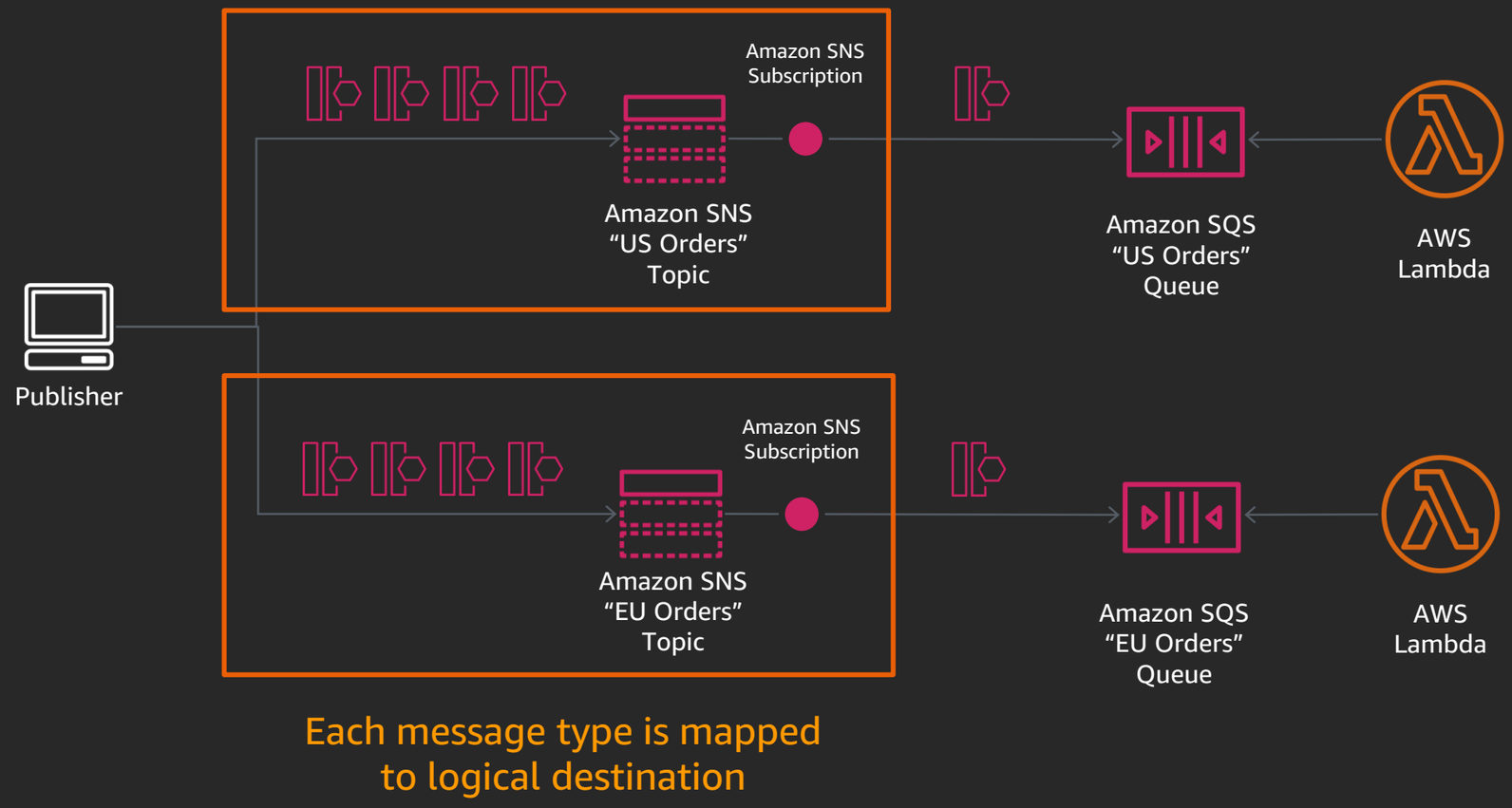
- Publishers do not need to route message
- Subscribers do not need to filter for message of interest
- Lowers cost

Message Attributes

```
{
  "location": "eu-west"]
}
```

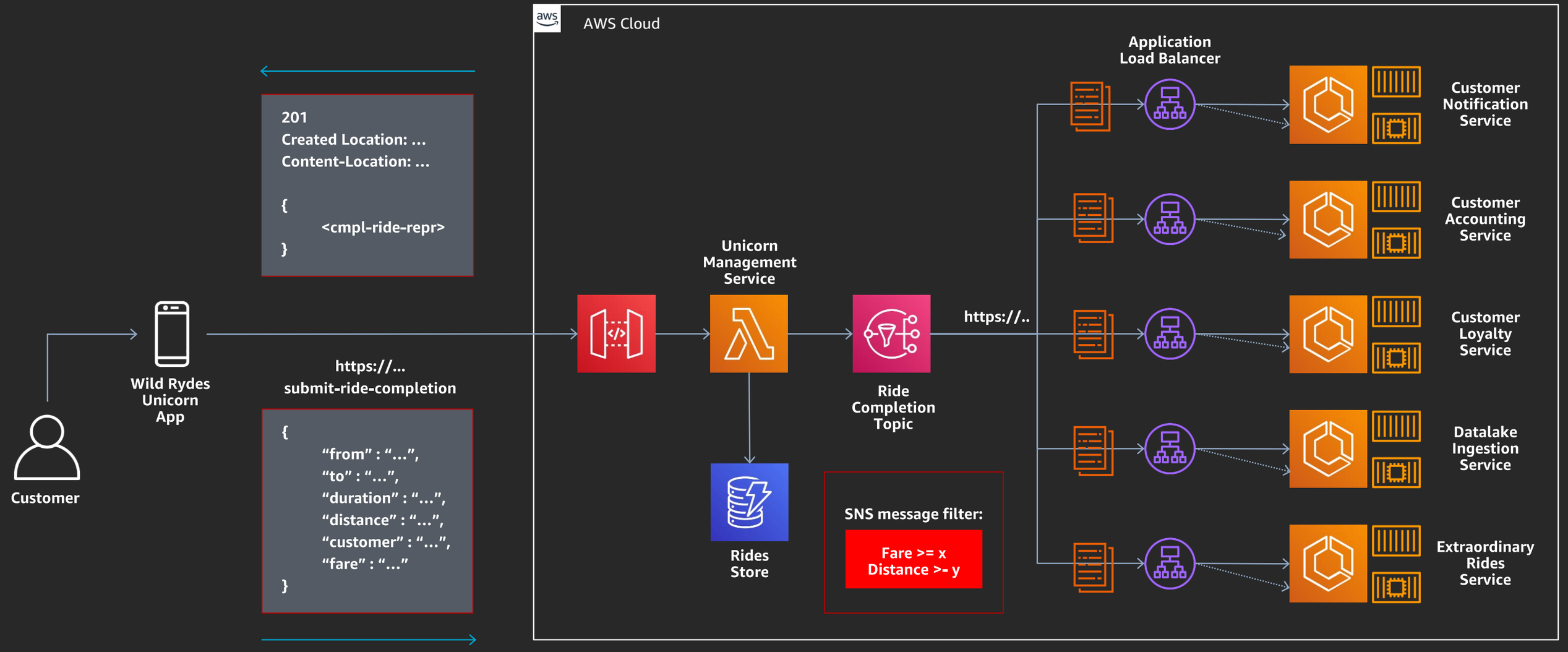


Mapping events to topics



Each message type is mapped to logical destination

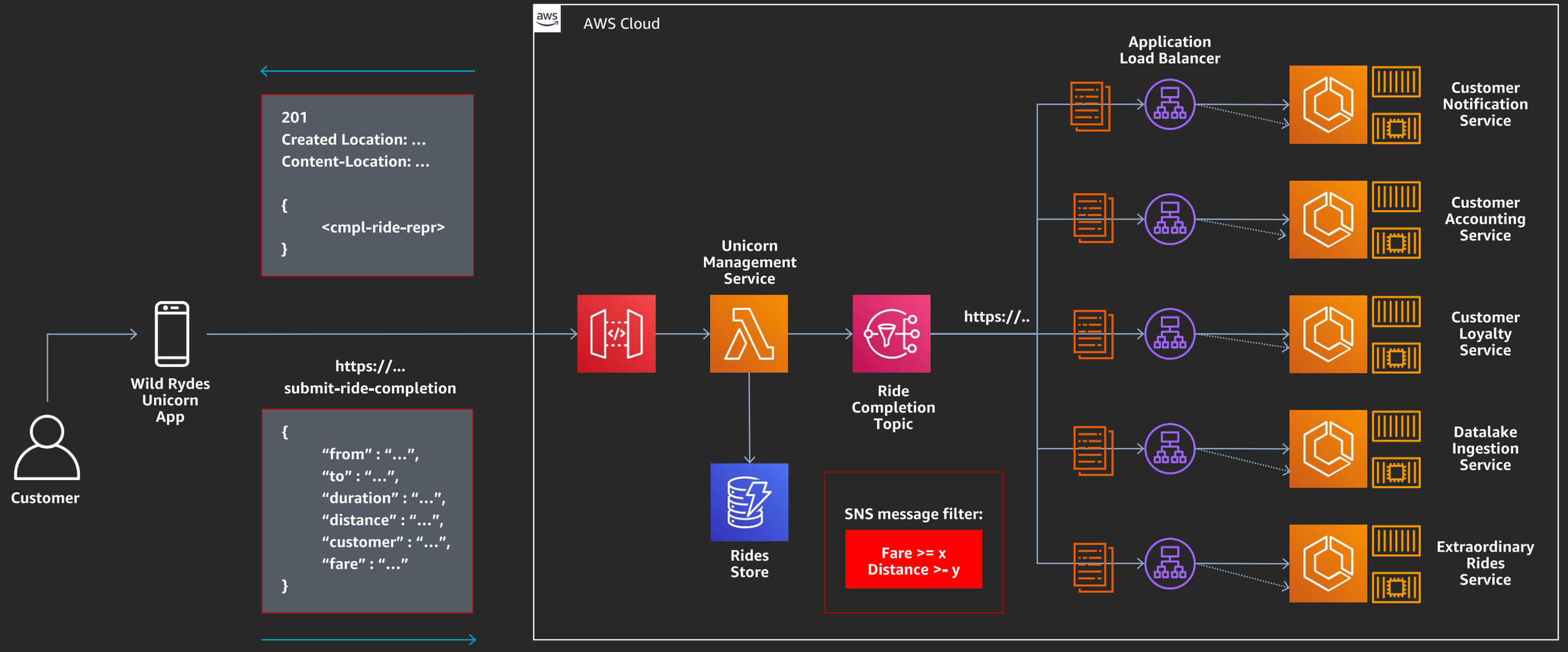
Logical architecture: Fan-out and Message Filtering



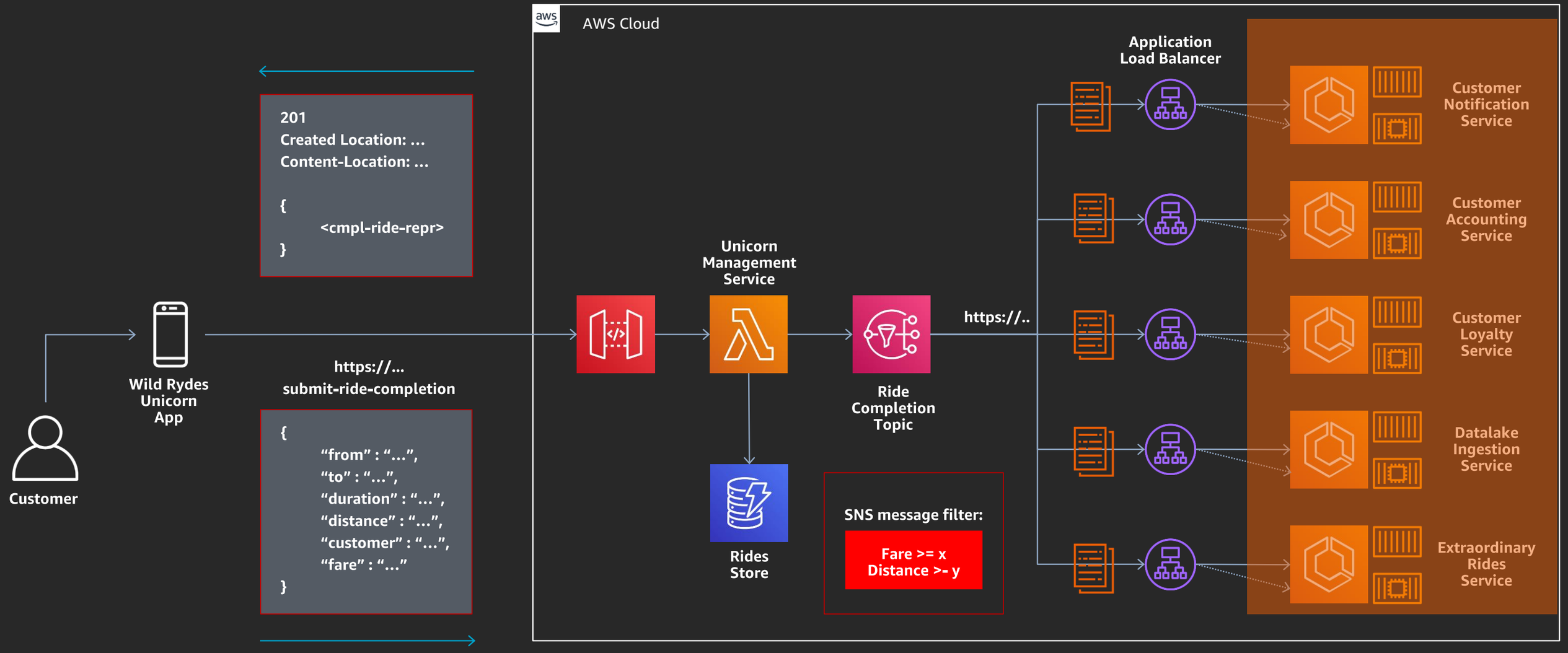
Live Demo: 2

aws SUMMIT ONLINE

Logical architecture: Fan-out and Message Filtering



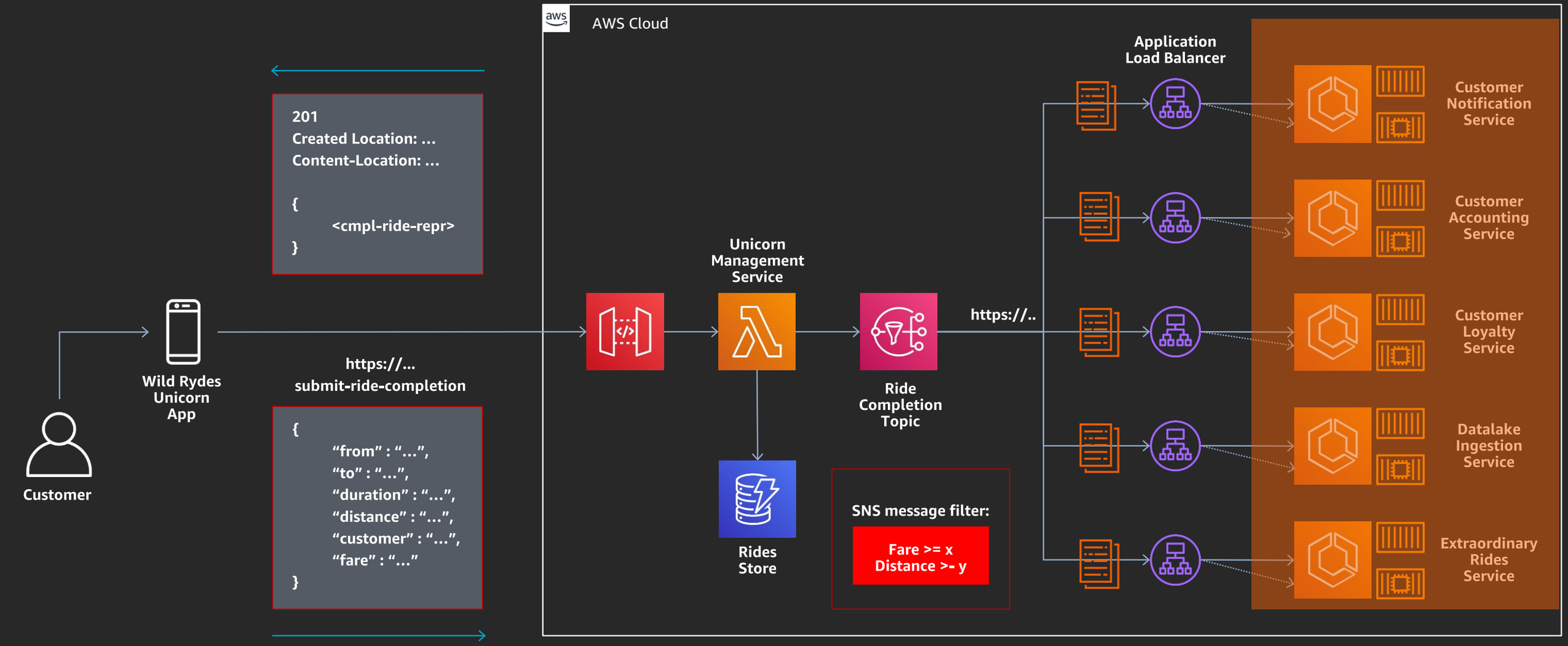
What if a service is offline ?



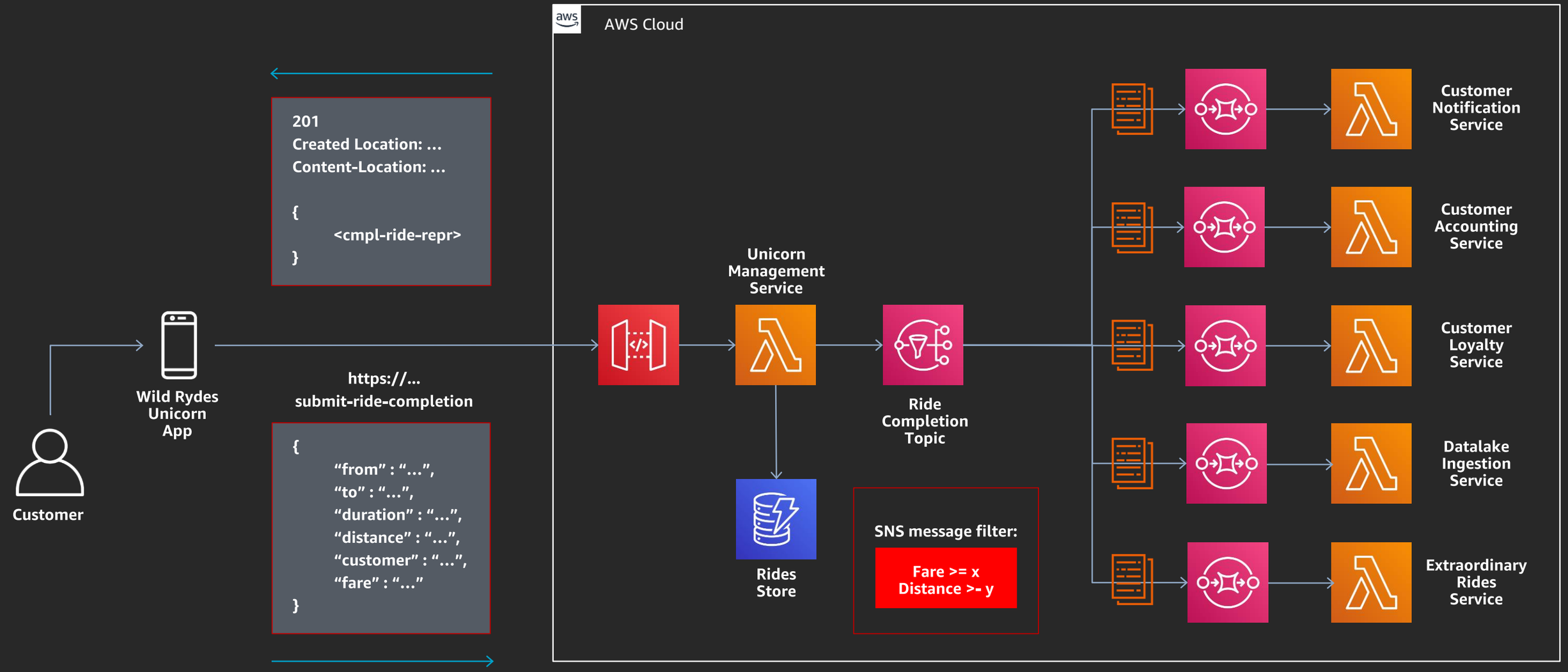
Integration via Messaging: Topic-queue-chaining pattern

aws SUMMIT ONLINE

Logical architecture: Fan-out and Message Filtering



Logical architecture: Topic-Queue Chaining & Load balancing



Quick recap

Common integration anti-patterns

Advanced integration patterns:

- Publisher/Subscriber
- Message filter pattern
- Topic-queue-chaining pattern

Coding – How to implement these patterns

Where to learn more

Where to learn more ?

References

https://rebrand.ly/mospvqd

Additional resources

AWS Compute Blog:

Understanding asynchronous messaging for microservices
https://aws.amazon.com/blogs/compute/understanding-asynchronous-messaging-for-microservices/

AWS re:Invent 2019:

Application integration patterns for microservices (API315-R3)

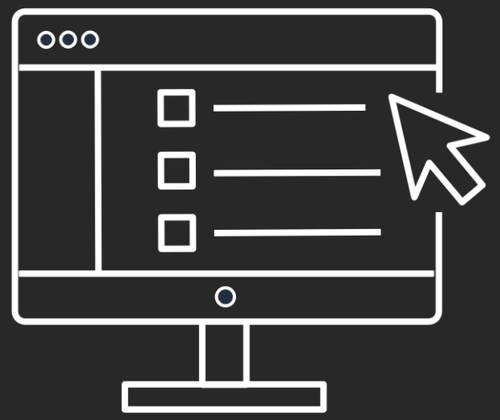
https://www.youtube.com/watch?v=K6Ehvt656Ss

Pub/Sub Messaging: Resources

https://aws.amazon.com/pub-sub-messaging/

AWS Event-Driven Architecture

https://aws.amazon.com/event-driven-architecture/



Additional resources

Amazon SQS

https://aws.amazon.com/sqs/getting-started/

Amazon SNS

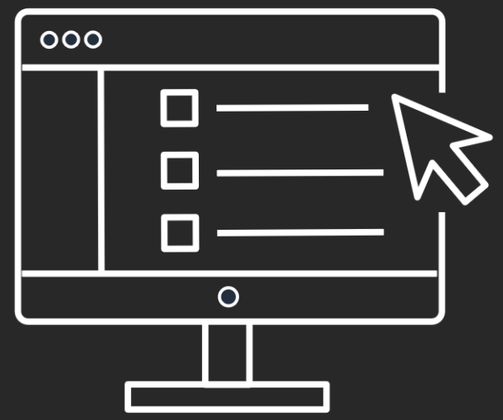
https://aws.amazon.com/sns/getting-started/

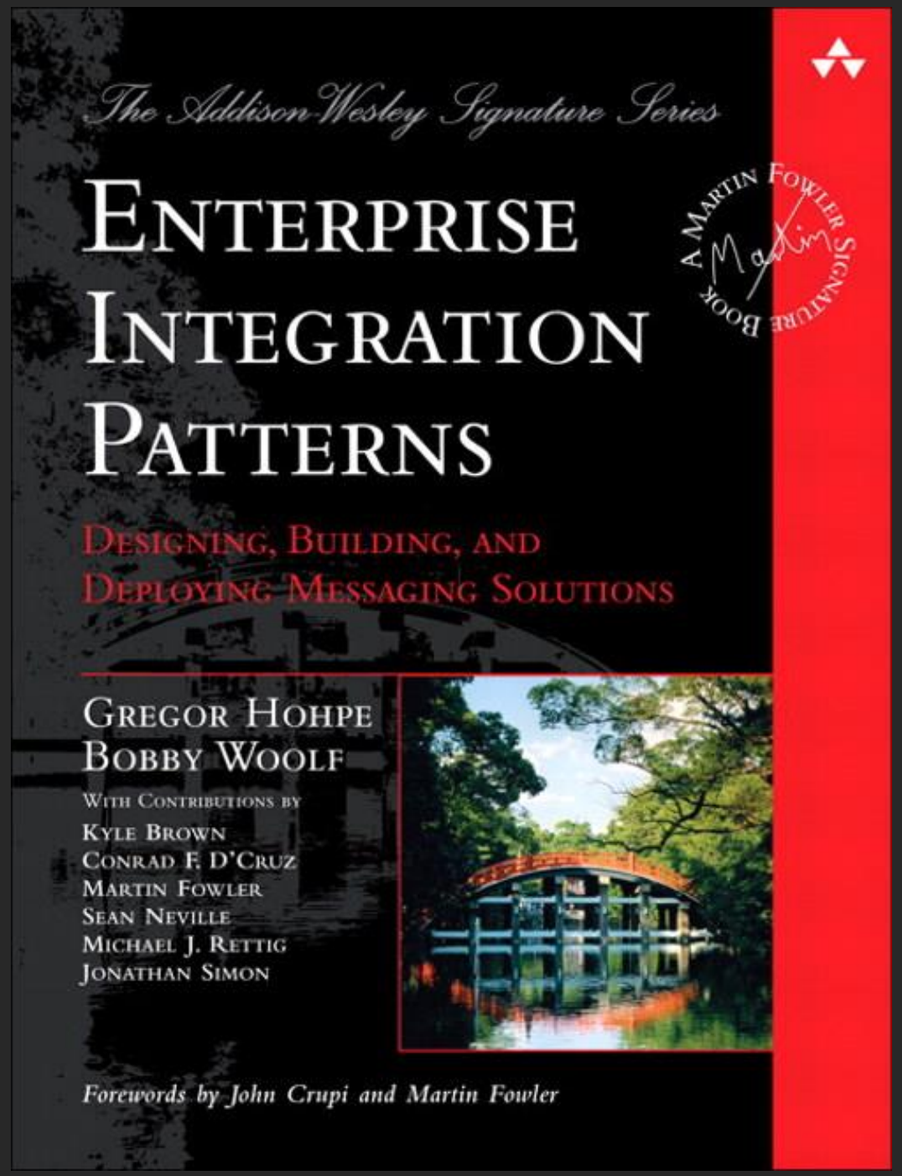
Amazon API Gateway

https://aws.amazon.com/api-gateway/

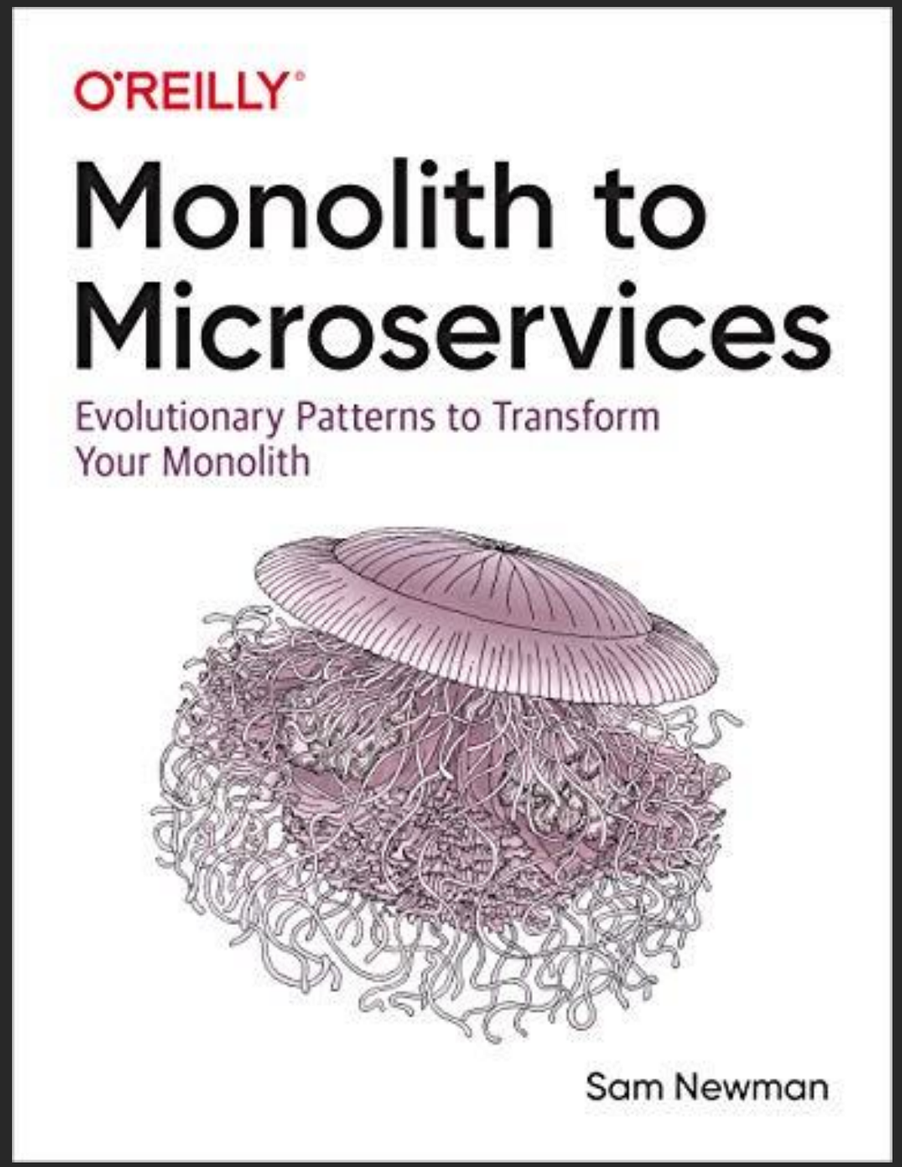
AWS Serverless Application Model (SAM)

https://aws.amazon.com/serverless/sam/





amazon.com.au/dp/B007MQLL4E



amazon.com.au/dp/B081TKSSNN

Thank you!

Chris Modica

@chris_modica_

aws SUMMIT ONLINE