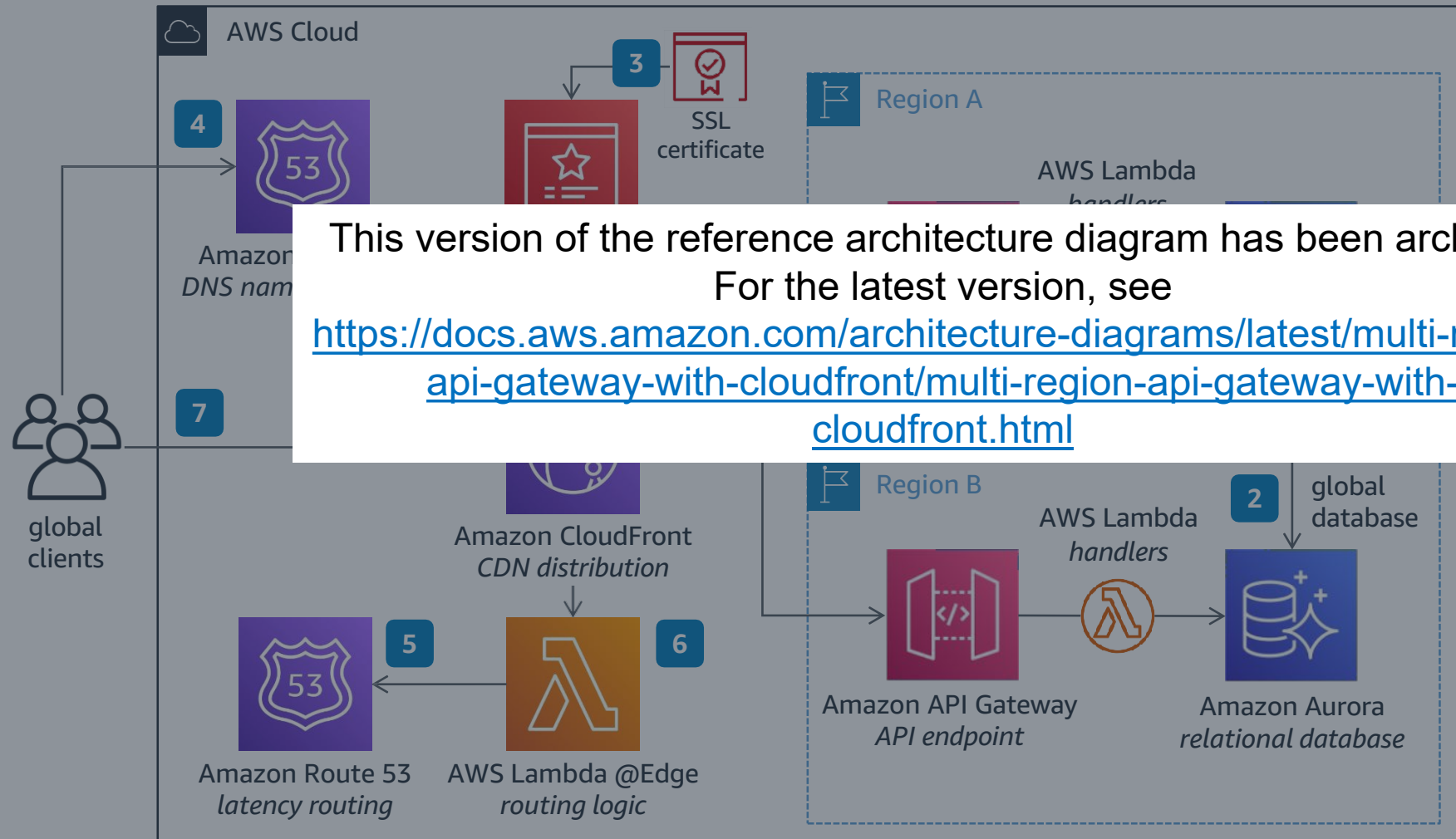


Multi-region API Gateway with CloudFront

Reduce latency for end-users, while increasing an application's availability by providing API Gateway endpoints in multiple AWS Regions. Each endpoint offers read-local write-global data synchronization supported by the Amazon Aurora global database.



- 1 Deploy an API endpoint in two or more **AWS Regions** using **Amazon API Gateway**, then handle requests using **AWS Lambda** connected to an **Amazon Aurora** relational database.
- 2 To keep data in sync across all **AWS Regions**, enable the Aurora Global Database feature. Aurora automatically routes write requests to the primary node to support data transactions, and replicates the changes to all nodes across **AWS Regions**.
- 3 To support custom domains, upload the domain's SSL Certificate into **AWS Certificate Manager (ACM)** and attach it to an **Amazon CloudFront** distribution.
- 4 Point your domain name to **CloudFront** by using **Amazon Route 53** as your DNS name resolution service.
- 5 Set up a routing rule on **Route 53** to route your global clients to the **AWS Region** with least latency to their location.
- 6 Ensure clients can authenticate seamlessly to **API Gateway** endpoints in any Region by using **AWS Lambda @Edge** to query **Route 53** for the best **AWS Region** to forward the request to. This normalizes authorization by abstracting the specifics of each regional endpoint.
- 7 Clients across the globe can then connect to your APIs using a single endpoint available in edge locations.
- 8 **CloudFront** will seamlessly route client requests from edge locations to the API in the **AWS Region** with the lowest latency to the client's location.



Reviewed for technical accuracy April 19, 2022

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Reference Architecture