Highly Available and Resilient Serverless Application with DR Strategy.

🚀 **What happens when your API goes down in one region?**

In this project, I simulated exactly that scenario by building a **serverless, cross-region disaster recovery (DR) and high availability (HA) architecture** on AWS.

🔹 **The Challenge:**  
APIs deployed in a single region risk downtime if that region experiences issues. For global users, this could mean service interruptions and unhappy customers.

🔹 **My Approach:**

* Designed **DynamoDB Global Tables** to replicate data across us-east-1 and us-west2.
* Built **Lambda functions** for reading/writing data into DynamoDB.
* Exposed them through **API Gateway endpoints** (/read and /write).
* Configured **Amazon Route 53** for DNS-based **Failover**:
  + **Primary Region:** API Gateway in us-east-1.
  + **Secondary (failover) Region:** API Gateway in us-west2 (activated automatically on 5xx errors).

🔹 **Results & Takeaway:**

* The system stayed **resilient and always-on**, when the **primary region simulated a failure**, traffic instantly failed over to the secondary region without users noticing downtime.
* With **DynamoDB Global Tables**, both regions stayed in sync, guaranteeing data consistency during failover.
* I proved that by combining **serverless services (Lambda, API Gateway, DynamoDB, Route 53)** you can achieve **enterprise-level disaster recovery and high availability** without managing servers.

🔗 GitHub: [*Git*](https://github.com/shakilM781)

This project reinforced for me that true cloud architecture success isn’t measured when everything works, it’s proven when systems stay **resilient, available, and seamless for users even in the face of failure**.

A computer screen shot of a computer

AI-generated content may be incorrect.