Task : Employee Management Application Using AWS Serverless Technologies

Project Overview

The goal of this task was to develop a full-stack employee management system that performs CRUD (Create, Read, Update, Delete) operations. The backend was built entirely using AWS serverless services — DynamoDB, Lambda, and AppSync — while the frontend provided a user-friendly interface to interact with employee data.

Implementation Details

Data Storage with DynamoDB:
Employee records were stored in a DynamoDB table, leveraging its high scalability and flexible schema to accommodate the application's data needs. DynamoDB's managed service features ensured automatic backups, multi-region replication, and low-latency access.

Business Logic Using Lambda:
Update operations and other complex business logic were handled via Lambda functions. These functions were triggered by API requests routed through AppSync and performed validation, processing, and updates to the DynamoDB table. This approach enabled a clean separation of concerns and facilitated easy maintenance and scalability.

GraphQL API with AppSync:
AWS AppSync was configured to serve as the application's API layer. It allowed the frontend to perform queries to fetch employee data, mutations to add or update records, and subscriptions for real-time updates. AppSync efficiently orchestrated communication between the frontend and backend Lambda/DynamoDB services.

Frontend Integration:
The frontend featured an employee table UI where users could create, view, update, and delete employee records. It integrated directly with the AppSync API, enabling seamless data flow and real-time interface updates. The frontend was built using modern web technologies and connected via GraphQL queries and mutations.

Key Features

- Create: Users could add new employee records which were saved in DynamoDB through AppSync and Lambda.

- Read: The employee list was dynamically fetched using GraphQL queries with support for filtering.

- Update: Employee details were updated through Lambda functions invoked by GraphQL mutations, ensuring centralized business logic.

- Delete: Records could be deleted with real-time UI updates reflecting changes via GraphQL subscriptions.

CRUD OPERATIONS:

UI: READ FROM BACKEND:

Id	Name	Email Id	Phone number	Actions	
1	arun	arun364@gmail.com	8362546273	EDIT	DELETE
2	ravi	ravi123@gmail.com	9963426574	EDIT	DELETE
3	akilesh	akilesh@gmail.com	8735274673	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

ADD NEW

CREATE:

Search [   ]

Id	Name	Email Id	Phone number	Actions	
1	arun	arun364@gmail.com	8362546273	EDIT	DELETE
2	ravi	ravi123@gmail.com	9963426574	EDIT	DELETE
3	akilesh	akilesh@gmail.com	8735274673	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

Add New Data

Id	Name	Email Id	Phone number	Action	
5	Naveen	naveen@gmail.com	8284727842	Save	Cancel

AFTER CREATE OPERATION:

Search [   ]

Id	Name	Email Id	Phone number	Actions	
1	arun	arun364@gmail.com	8362546273	EDIT	DELETE
2	ravi	ravi123@gmail.com	9963426574	EDIT	DELETE
3	akilesh	akilesh@gmail.com	8735274673	EDIT	DELETE
5	Naveen	naveen@gmail.com	8284727842	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

ADD NEW

UPDATE:

Search				

Id	Name	Email Id	Phone number	Actions	
1	arun	arun364@gmail.com	8362546273	EDIT	DELETE
2	ravi	ravi123@gmail.com	9963426574	EDIT	DELETE
3	akilesh	akilesh@gmail.com	8735274673	EDIT	DELETE
5	Naveen	naveen@gmail.com	8284727842	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

Update the Data

Id	Name	Email Id	Phone number	Action	
1	arunkumar	arun364@gmail.com	8362546273	Update	Cancel

BEFORE UPDATE:

Search				

Id	Name	Email Id	Phone number	Actions	
1	arun	arun364@gmail.com	8362546273	EDIT	DELETE

AFTER UPDATE:

Search				

Id	Name	Email Id	Phone number	Actions	
1	arunkumar	arun364@gmail.com	8362546273	EDIT	DELETE

DELETE:

BEFORE DELETE:

| Search | | | | | |

Id	Name	Email Id	Phone number	Actions	
1	arunkumar	arun364@gmail.com	8362546273	EDIT	DELETE
2	ravi	ravi123@gmail.com	9963426574	EDIT	DELETE
3	akilesh	akilesh@gmail.com	8735274673	EDIT	DELETE
5	Naveen	naveen@gmail.com	8284727842	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

ADD NEW

AFTER DELETE:

| Search | | | | | |

Id	Name	Email Id	Phone number	Actions	
1	arunkumar	arun364@gmail.com	8362546273	EDIT	DELETE
5	Naveen	naveen@gmail.com	8284727842	EDIT	DELETE
7	lokesh	lokesh5667@gmail.com	9284238482	EDIT	DELETE

ADD NEW

FILTER:

Search	arun|

Id	Name	Email Id	Phone number	Actions	
1	arunkumar	arun364@gmail.com	8362546273	EDIT	DELETE

ADD NEW

ERROR HANDLING:

Duplicate ID Error:

Add New Data

Id	Name	Email Id	Phone number	Action	
1	karthik			Save	Cancel

ID already exists.

Empty Input Field Validation:

Add New Data

Id	Name	Email Id	Phone number	Action	
12	karthik	karthik@gmail.co		Save	Cancel

Enter all the fields Correctly!

No Data Found:

If Table is Empty:

Search	

No Data Found

ADD NEW

DynamoDB Table:

Table: Employee - Items returned (3)
Scan started on May 30, 2025, 11:53:37

‹ 1 › ⚙

	id (Number) ▼	emailId ▼	name ▼	phoneNumber ▼
☐	7	lokesh5667...	lokesh	9284238482
☐	1	arun364@...	arunkumar	8362546273
☐	5	naveen@g...	Naveen	8284727842