Serverless Web Application using AWS

A Full-Stack, Scalable, and Secure Web App without Server Management

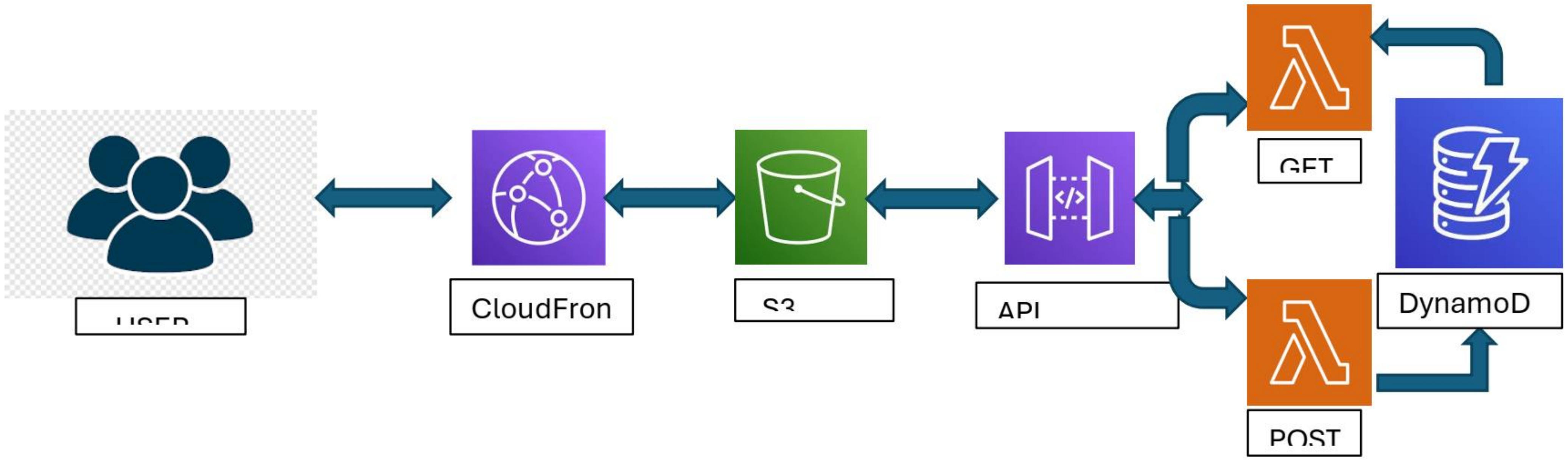
Presented by: Pravin Balaji K N

Introduction

This project demonstrates the design and deployment of a fully serverless web application using AWS services. By leveraging S3, Lambda, API Gateway, and DynamoDB, we built a scalable and cost-effective solution that processes and stores student data through a responsive frontend interface.

The solution ensures secure and fast access through CloudFront and eliminates the need for any server provisioning or maintenance.

Architecture Overview



Services Involved:

- **Amazon S3:** Hosts the frontend HTML/JavaScript.
- **Amazon CloudFront:** Distributes content securely over HTTPS.
- **API Gateway:** Interfaces between frontend and backend.
- **AWS Lambda:** Runs GET and POST logic.
- **Amazon DynamoDB:** NoSQL data store for student records.

Flow Summary:

- User interacts with a static frontend hosted on S3.
- JavaScript sends requests to API Gateway.
- API Gateway routes to Lambda (GET/POST)
- Lambda processes data and interacts with DynamoDB
- CloudFront serves the frontend over HTTPS

Work Partition

This serverless web application project highlights the efficiency of using AWS cloud-native services to build responsive, scalable, and secure systems. It proves that high-performance applications can be built without managing a single server, allowing developers to focus on features and experience.

- Partition 1: **Backend Configuration**
- Partition 2: **API Gateway & Static Website**
- Partition 3: **CloudFront Distribution**

Problem-Solution Summary

The Problem:

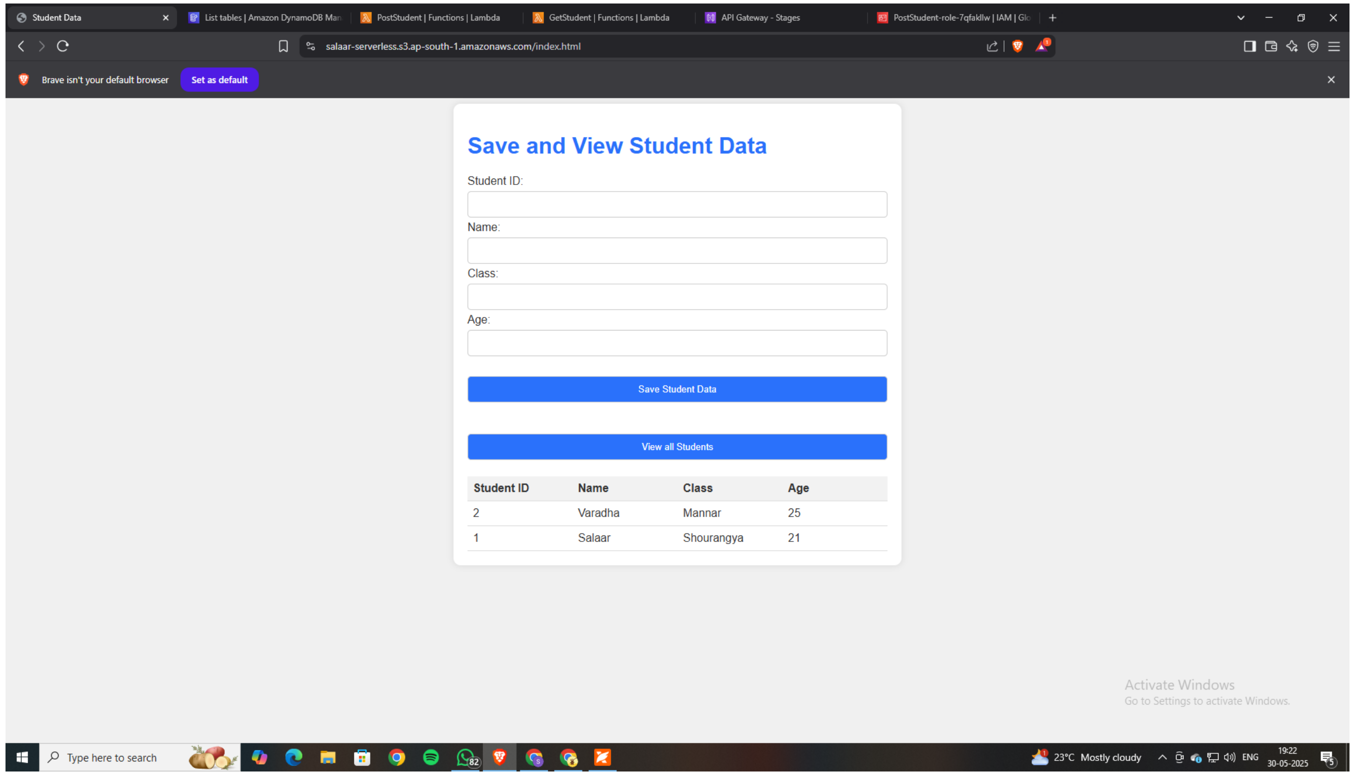
- Traditional web hosting involves provisioning, scaling, and securing backend servers.
- Increased complexity and cost for lightweight apps.

Our Solution:

- A 100% serverless architecture with no servers to manage.
- Uses AWS managed services to automate scaling, security, and availability.
- Rapid deployment of full-stack apps with minimal code.

Primary Use Cases

- To-Do-List Static website can be hosted.
- Online registration and student forms.
- Contact submission systems.
- Lightweight SaaS product dashboards.
- Event signup or feedback applications.


Student Data
List tables | Amazon DynamoDB Man
PostStudent | Functions | Lambda
GetStudent | Functions | Lambda
API Gateway - Stages
PostStudent-role-7qfakllw | IAM | Gl
salaar-serverless.s3.ap-south-1.amazonaws.com/index.html
Brave isn't your default browser
Set as default
Save and View Student Data
Student ID:
Name:
Class:
Age:
Save Student Data
View all Students
Activate Windows
Go to Settings to activate Windows.
Type here to search
23°C Mostly cloudy
ENG
19:22
30-05-2025

Student ID	Name	Class	Age
2	Varadha	Mannar	25
1	Salaar	Shourangya	21

Benefits & Value Proposition

- **Serverless Design:** No infrastructure to manage.
- **High Scalability:** Seamless scaling with user traffic.
- **Real-Time Processing:** Instant data storage and retrieval via Lambda & DynamoDB.
- **Low Cost:** Pay only for what you use.
- **Secure Access:** HTTPS and role-based security.

Conclusion

This serverless web application project highlights the efficiency of using AWS cloud-native services to build responsive, scalable, and secure systems. It proves that high-performance applications can be built without managing a single server, allowing developers to focus on features and experience.

Thank You