Serverless Architecture on AWS

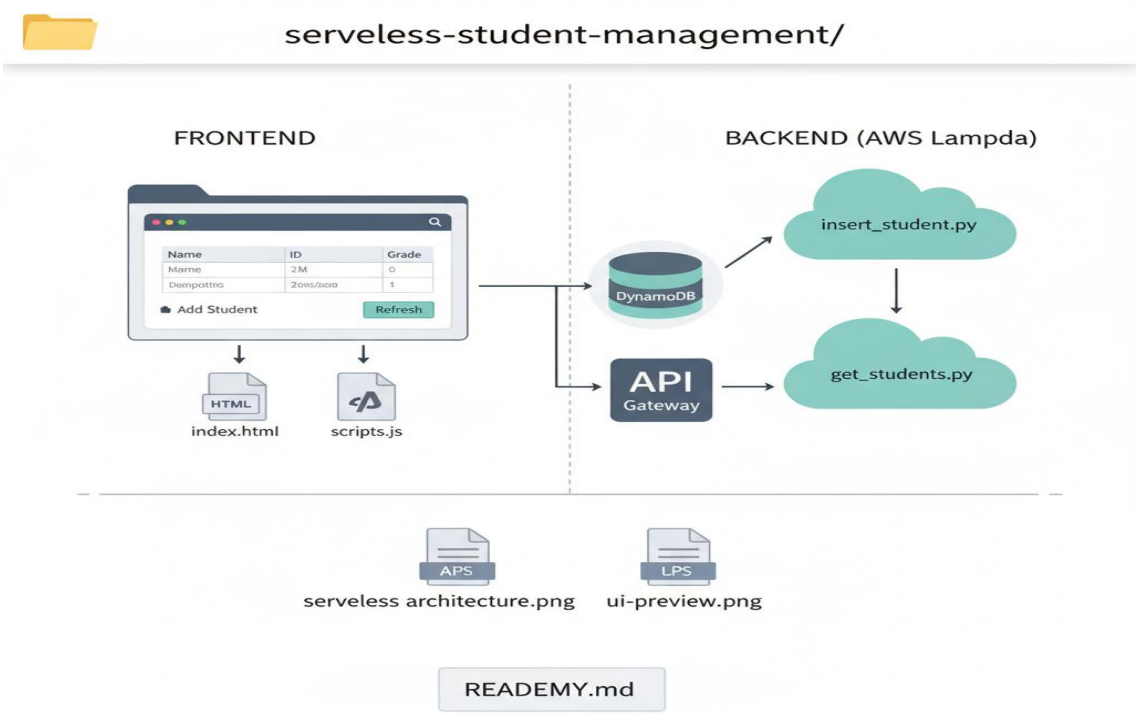
Author: Shaikh Mubashir Ahmed
Email: shaikhmubashir215@gmail.com
Contact: 9701223004
GitHub: https://github.com/Mubashir970/
LinkedIn: https://www.linkedin.com/in/shaikh-mubashir2000/

Project Overview

The Serverless Student Management System is a cloud-native web application built using AWS serverless services. The project demonstrates how to design, deploy, and operate a scalable application without managing servers. Users can add and retrieve student records through a web interface hosted on Amazon S3.

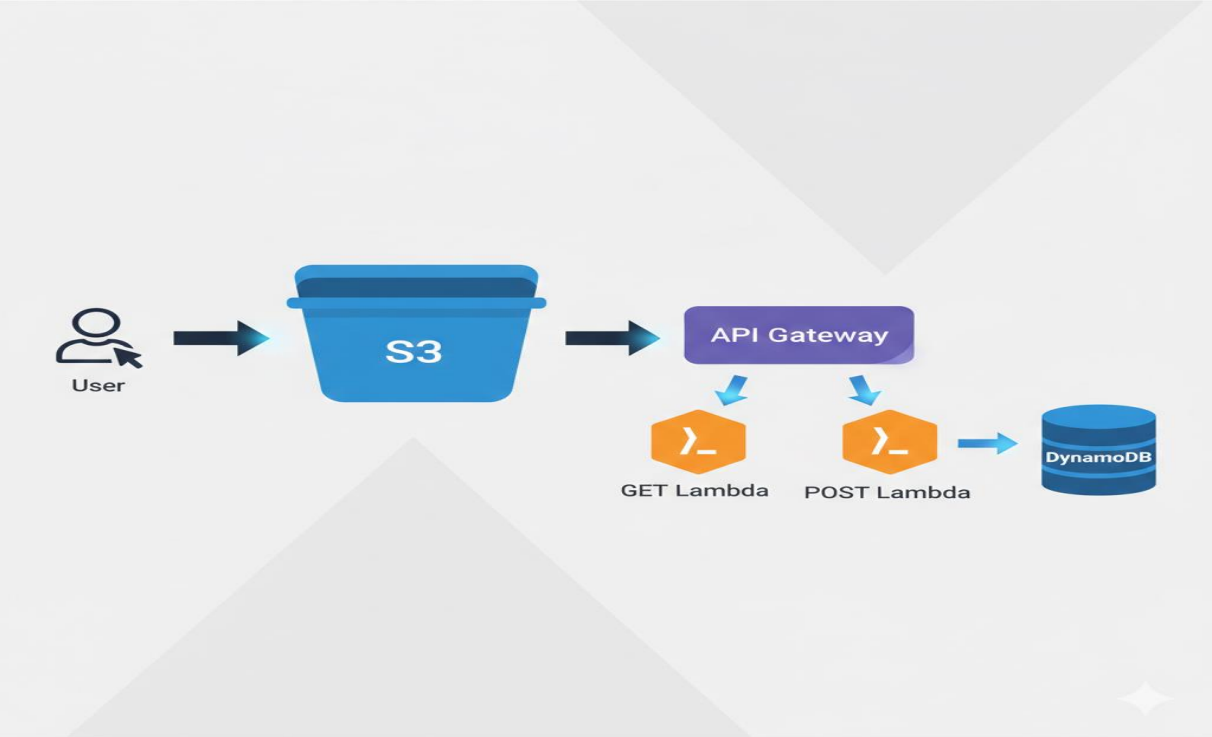
Architecture Overview

The application follows a fully serverless architecture. The frontend is hosted on Amazon S3 as a static website. User requests are routed through Amazon API Gateway, which invokes AWS Lambda functions. These Lambda functions interact with Amazon DynamoDB to store and retrieve student data.



Architecture Flow

1. User accesses the web application via S3 static website URL.
2. Frontend sends HTTP requests using JavaScript (AJAX).
3. Requests reach Amazon API Gateway.
4. API Gateway triggers AWS Lambda functions.
5. Lambda functions perform operations on DynamoDB.
6. Response is returned to the user interface.



Technologies Used

Frontend: HTML, CSS, JavaScript, jQuery
Cloud Services: Amazon S3, API Gateway, AWS Lambda, DynamoDB, IAM

DynamoDB Table Design

Table Name: Student_Details
Partition Key: studentid (String)
Attributes: name (String), class (Number), age (Number)

Security Implementation

IAM roles are configured using the principle of least privilege. The frontend does not expose any AWS credentials. CORS is enabled only for required HTTP methods.

Learning Outcomes

This project provided hands-on experience with AWS serverless architecture, API Gateway and Lambda integration, DynamoDB data modeling, and deploying a production-style static web application.

Conclusion

The Serverless Student Management System demonstrates how modern cloud-native applications can be built efficiently using AWS managed services. The architecture ensures scalability, security, and cost-effectiveness while eliminating server management overhead.