**FronEnd – Using Flask**

**Enterprise Integration Technology**

Sarvan Veluppillai

Zikora Orakwe

Ammar Ahmad

Jing Yu

**TABLE OF CONTENTS**

[Revision History 3](#_Toc152182528)

[Executive Summary 4](#_Toc152182529)

[Prerequisites 4](#_Toc152182530)

[Project Timeline 4](#_Toc152182531)

[Stakeholders 5](#_Toc152182532)

[Architecture Diagram 5](#_Toc152182533)

[Conclusion 9](#_Toc152182534)

# **Revision History**

The following revisions identify major changes to this document:

Table 1:

| DATE | REVISED BY | VERSION | REVISION DETAILS & REFERENCE |
| --- | --- | --- | --- |
| Nov 13, 2023 | Sarvan Veluppillai | 001 | Document Creation |
| Nov 14, 2023 | Zikora Orakwe  Ammar  Jing Yu | 001 | Contributors |

# **Executive Summary**

Our team has developed a front-end application that demonstrates the functionality of our streaming infrastructure. We have integrated API gateways into this application, which are connected to Lambda functions to perform specific tasks.

API Gateway is a service that allows developers to create, publish, and manage APIs for their back-end services. It acts as a “front door” for applications to access data, business logic, or functionality from your back-end services. Lambda, on the other hand, is a serverless compute service that lets you run code without provisioning or managing servers. By integrating API Gateway with Lambda, you can build scalable and cost-effective APIs that can handle millions of requests per day.

# **Prerequisites**

**The following development skills are required for this project:**

* Python: This language is needed for backend server programming and Lambda codes.
* Flask: Flask will be used to deploy the web page application.
* jQuery: This is used in the HTML to communicate with the backend server.
* SweetAlert: This is used to control data flow within the application.

**Technologies used**

Flask is a micro web framework written in Python that allows developers to build web applications quickly and easily. It provides tools, libraries, and technologies that allow developers to build scalable and secure web applications.

jQuery is a fast, small, and feature-rich JavaScript library that simplifies HTML document traversing, event handling, and Ajax interactions for rapid web development.

SweetAlert is a beautiful, responsive, customizable, and accessible replacement for JavaScript’s popup boxes that provides a better user experience.

# Project Timeline

* Project Initiation (1 Week)
* Requirement Gathering (1 Week)
* Design and Development (1 Month)
* Testing For Quality Assurance (2 Weeks)

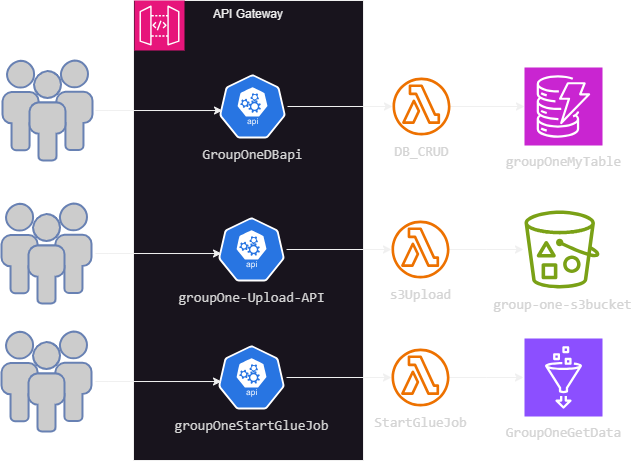
# Stakeholders

* Executive Leadership (CEO, COO, CIO)
* IT Department
* Consultants
* Customers
* Vendors and Partners
* Legal Department

**Implementation Team:**

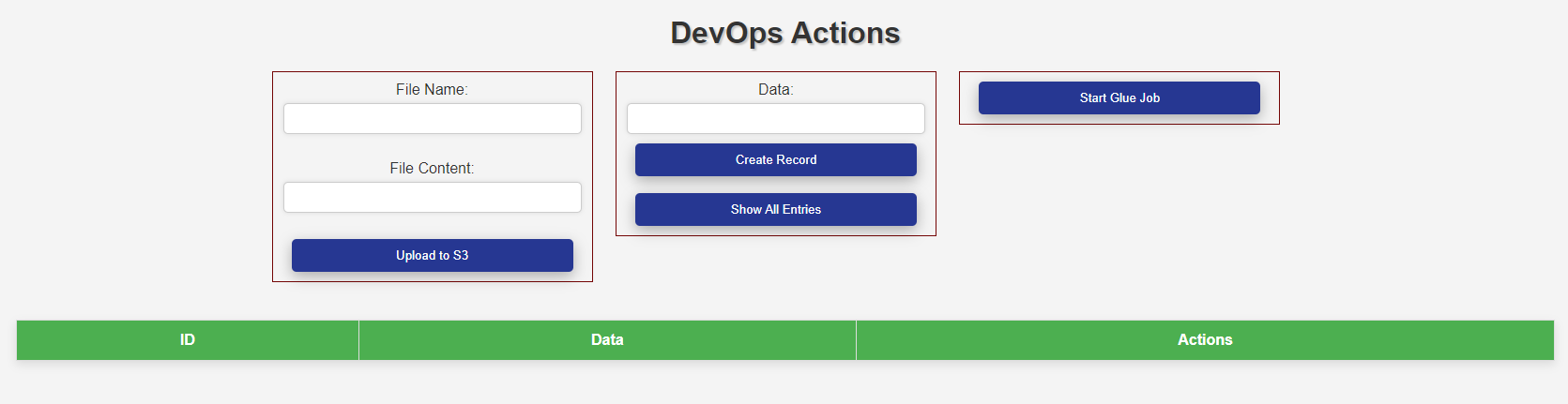
* Project Manager - Sarvan Veluppillai
* Cloud Architect - Zikora Orakwe
* Developers - Ammar Ahmad
* AWS Account Manager - Jing Yu

# Architecture Diagram



Application Flow

The main page

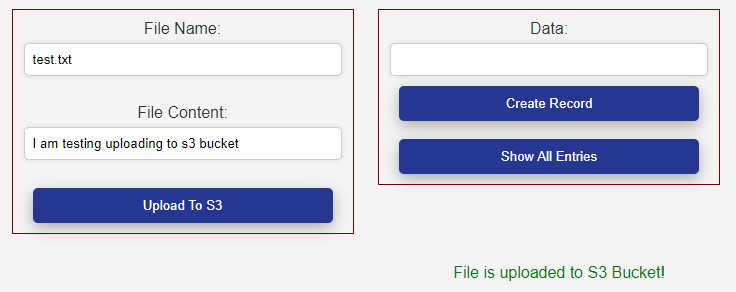


**The application flow consists of three sections:**

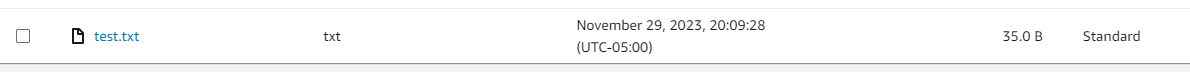
* **Uploading a file to an S3 bucket**: This section deals with uploading a file to an S3 bucket, which is a simple storage service offered by AWS that allows you to store and retrieve data from anywhere on the web.
* **Database operation:** This section interacts with DynamoDB, which is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability.
* **Starting a Glue job:** This section is responsible for starting a Glue job, which is a fully managed extract, transform, and load (ETL) service that makes it easy to move data between data stores.

Each of these tasks has an API connected to a Lambda function that performs an operation on the destination resource. API Gateway is used to create, publish, and manage APIs for these Lambda functions. By integrating API Gateway with Lambda, you can build scalable and cost-effective APIs that can handle millions of requests per day.

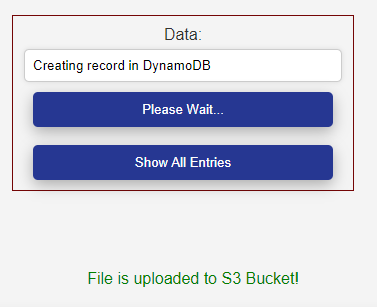
1. Here is the snippet of uploading file to S3 bucket.



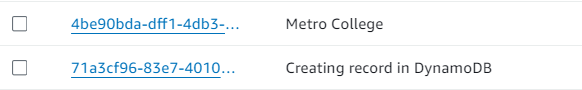
1. S3 Bucket



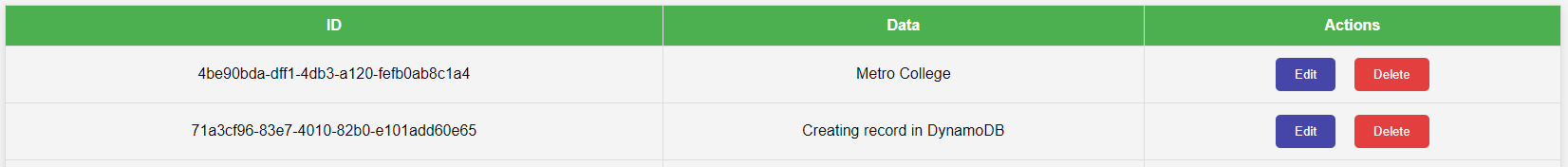
1. Creating record in DynamoDB.



1. Record in DynamoDB

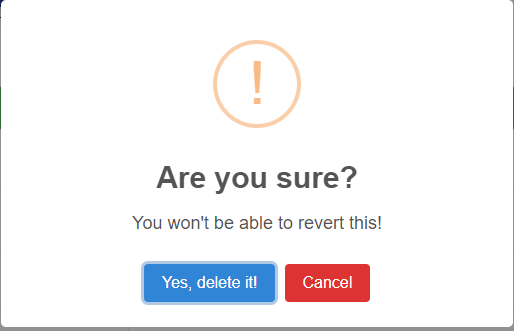
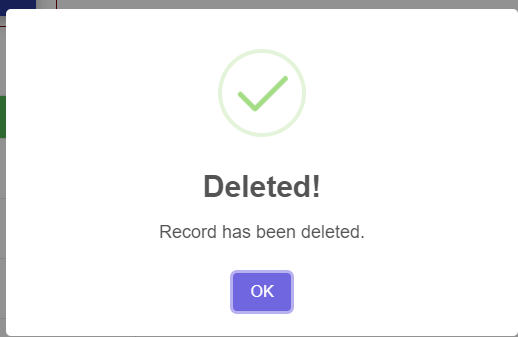


1. Listing all entries



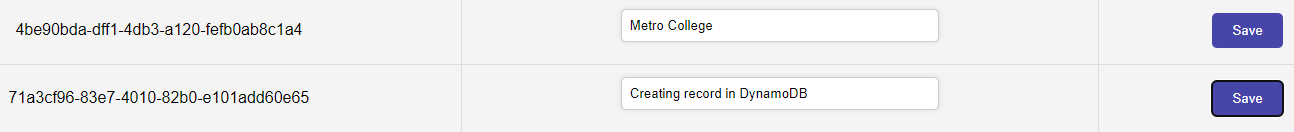
In this grid, “Edit” and “Delete” buttons are associated to each record. Multiple editing can be done.

1. Deletion



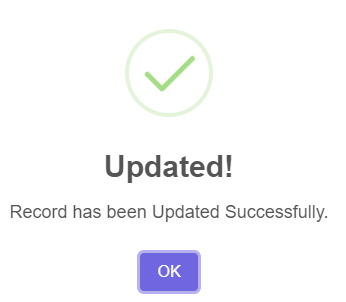
When the “Delete” is chosen, a conformation screen is presented. If “Yes, delete it!” is chosen, then the request to delete that record is sent to the backend to process that request.

1. Edit



Multiple records can be edit at the same time. You can type directly into the input box to change the values and press “Save” to save the record. When the “Save” button is selected, the request is sent to backend through API to process those changes.

Here is the conformation screen



# Conclusion

In this font-end application that demonstrates the functionality of out streaming infrastructure. The application flow consists of three sections: uploading a file to an S3 bucket, database operations that interact with DynamoDB, and starting a Glue job. Each of these tasks has an API connected to a Lambda function that performs an operation on the destination resource. API Gateway is used to create, publish, and manage APIs for these Lambda functions. By integrating API Gateway with Lambda, you can build scalable and cost-effective APIs that can handle millions of requests per day.

In terms of development skills, Python is required for backend server programming and Lambda codes, Flask will be used to deploy the web page application, jQuery is used in the HTML to communicate with the backend server, and SweetAlert is used to control data flow within the application.