# Project Overview

This project encompasses the development and deployment of a dynamic web application

utilizing a combination of AWS Elastic Beanstalk for application hosting, DynamoDB for

database services, and Python Flask as the web framework. The goal is to demonstrate the

effective use of cloud infrastructure to deploy a scalable, reliable web application that interacts

with a NoSQL database. The end product should allows users to query a map and its data.

## Key Components:

**AWS Elastic Beanstalk**: Automates the deployment process, from capacity provisioning, load

balancing, autoscaling to application health monitoring. It significantly reduces the management complexity without sacrificing control and flexibility.

**DynamoDB**: A fully managed NoSQL database service that provides fast and predictable

performance with seamless scalability. It's used to store and retrieve the application data,

ensuring high availability and data durability.

**Python Flask**: A lightweight WSGI web application framework designed to make getting started quick and easy, with the ability to scale up to complex applications. It serves as the backbone of the application, handling routing, requests, and responses.

**JIRA/Github:** Project was tracked and planned with Atlassian JIRA and shared on Github <https://quickmapfacts.atlassian.net/jira/software/projects/QMF/boards/1>

[**https://github.com/jaqkk/QuickMapFacts**](https://github.com/jaqkk/QuickMapFacts)

## Project Steps:

1. **Initial Setup**: Configuration of the AWS account, AWS CLI, local Python environment, and Elastic Beanstalk environment to align with project requirements.

2. **Application Deployment**: Utilization of AWS Elastic Beanstalk for deploying the Flask

application, leveraging the provided eb-flask.zip for a streamlined deployment process.

3. **Database Integration**: Creation and configuration of a DynamoDB table named

QMFS\_DATA to store application data, ensuring it's in the same region as the Elastic Beanstalk instance for optimal performance.

4. **Data Management**: Implementation of a Python script, QMF\_json\_to\_DynamoDB.py, to

import data from QMF\_CursorAI.json into DynamoDB, demonstrating data ingestion and

interaction within the application.

5. **Iterative Development and Troubleshooting**: Continuous monitoring and logging through

Elastic Beanstalk to identify and resolve issues, with iterative improvements based on GPT assistance for troubleshooting and optimization.

## Objective:

The project aims to showcase the integration of AWS services with a Python Flask application tocreate a robust web application. It highlights the practical application of cloud services for web

hosting and database management, emphasizing the ease of deployment, scalability, and the

ability to handle complex data interactions.