Music Recommendation System

Technical Design Document

Data Pipeline Integration & Music Recommendation System on AWS with Docker

TABLE OF CONTENTS

1 INTRODUCTION

This document is a Technical Design Document for use Music Recommendation System on AWS with Docker mini-Project. It provides guidance which is intended to assist the relevant technical users. It is also useful background reading for anyone involved in developing or monitoring the Music Recommendation System on AWS with Docker.

1.1 REQUIREMENTS

Develop a Data Pipeline Integration & Music Recommendation System on AWS with Docker

1.2 SCOPE

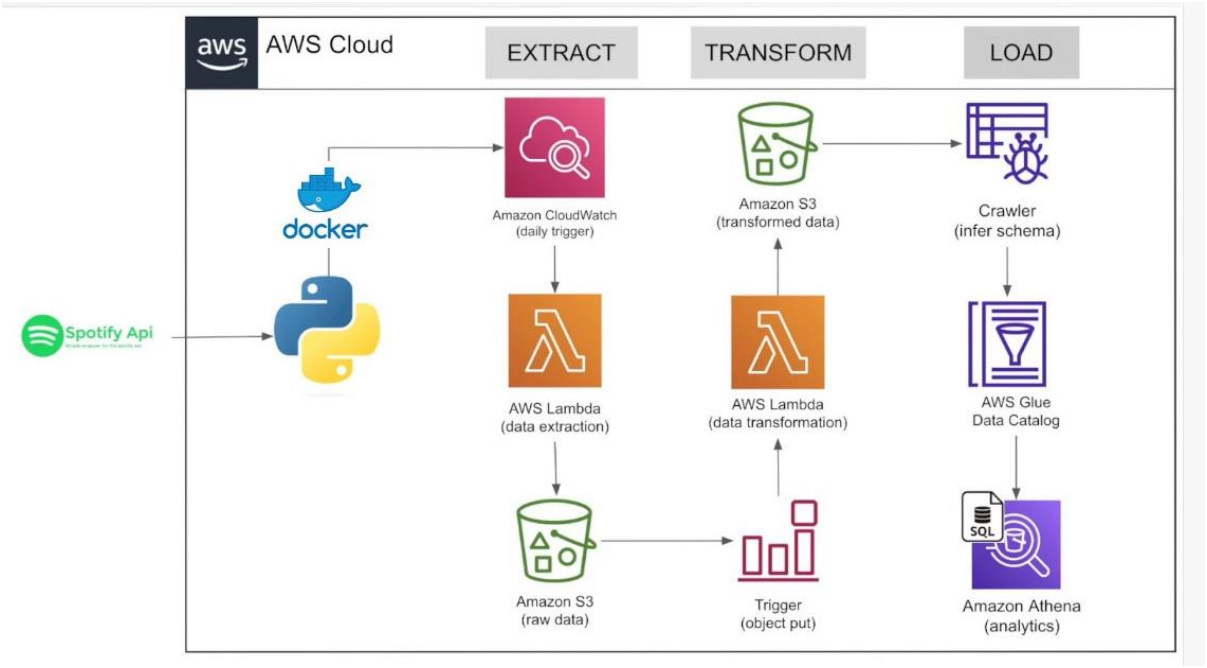
The project involves extracting data from the Spotify API using Dockerized Python scripts, ensuring consistency across environments. The data is transformed using Docker containers and AWS Glue for enrichment and schema management, and stored in Amazon S3 for querying via Athena. A CI/CD pipeline is set up using AWS services like CodePipeline, ECS, and ECR to automate deployment and testing of the ETL pipeline.

2 SYSTEM OVERVIEW

This project aims to construct a robust ETL (Extract, Transform, Load) pipeline using Docker and AWS services. The pipeline facilitates seamless extraction, transformation, and loading of data from the Spotify Web API, while Docker containers ensure portability and consistency across environments. In addition to data processing, the project integrates a music recommendation system that utilizes collaborative filtering techniques to deliver personalized song recommendations.

The project architecture combines the benefits of Docker containerization with AWS services to ensure efficient data ingestion, transformation, and analysis. AWS services such as S3, Glue, and Athena provide the necessary infrastructure to handle large volumes of data. The final deliverable will include insightful analytics and visualizations to better understand user behaviours and preferences, enabling more accurate and personalized recommendations for users.

2.1 SYSTEM ARCHITECTURE



1. **Amazon S3:** S3 bucket to store raw and transformed data extracted from the Spotify API.
2. **AWS Lambda:** Lambda function to handle the extraction of data from the Spotify API and load it into the S3 bucket.
3. **AWS Glue Crawler:** Glue Crawler to crawl the S3 bucket and create a Glue Data Catalog.
4. **AWS Glue Data Catalog:** Use the Glue Data Catalog to manage and maintain metadata about the data stored in S3.
5. **Amazon Athena:** Athena to query the data stored in the Glue Data Catalog.
6. **CloudWatch:** CloudWatch to monitor the AWS resources and Lambda functions.

7. **Spotify API Data:** Containerize the Lambda function for extracting data from the Spotify API using Docker to ensure consistency across development and production environments.
8. **Docker(Container):** Docker to define the environment and dependencies required for running the Spotify API extraction logic.
9. **Amazon ECS (Elastic Container Service):** For larger-scale or batch extractions, deploy the Docker container to Amazon ECS, which allows for scalable container orchestration.