Fall 2021 CS 687 Capstone Project
Progress Report
OneUI Kitchen Remodeling

Sravya Karnati
Advisor: Sam Chung, Ph.D, Ahereum Ju, Ph.D.
MS in Computer Science
School of Technology and Computing (STC)
City University of Seattle (CityU)
karnatisravya@cityuniversity.edu, chungsam@cityuniversity.edu

Abstract

The OneUI Kitchen remodeling website will provide seamless experiences to personalize their Kitchen in various categories at less cost that works for their family. This is also going to help the publishers, individual contractors, suppliers, or specialists in different categories who offer their services to reach the customer. This paper includes developing the web application using serverless framework architecture and deployed on AWS cloud services. This paper also addresses the scaling, recovery, and management of each component and the REST API service deployment. The application is hosted using the AWS S3 with high availability.

Keywords: OneUI Kitchen, Web Applications, HTML5, CSS, Javascript, Amazon Cloud Services, Design, Implementation, Web Server, Amazon DynamoDB, AmazonS3, AWS Lambda, RestAPI, AWS Cloud Services

1. INTRODUCTION

OneUI Kitchen Remodeling Project. We are undertaking this project to benefit people with various categories that are involved in kitchen remodeling and personalize each category, showing everything in one neatly designed UI page. People can choose each category specialist and look at their profile, collect the feedbacks of previous work, and use coupons. The Publishers of these various categories can get registered and offer their services, coupons and upload their work.

Problem Statement

Typically, the online search engines or applications like YellowPages or Yelp list the Kitchen remodeling contractors as a whole project. Since it's a complex project that requires lots of upfront research to stay within budget and avoid common pitfalls. Kitchen Remodeling requires various professional categories like General contractors, plumbers, floors, countertops, etc. Each of these individual publishers offers their services to utilize their coupons and get personalized designs or read reviews. Showing everything on One-UI is the Vision of the project and helps customers.

Motivation

The kitchen is considered the heart and hub of the house. The OneUI Kitchen remodeling website will provide seamless experiences to personalize their Kitchen in various categories at less cost that works for their family. This is also going to help the publishers, individual contractors, suppliers, or specialists in different categories who offer their services to reach the customer.

Approach

A comparison of Houzz and Angie's List has been made regarding the usability and reviews in the previous works. It will evaluate how different to find the different professionals of each category of kitchen remodeling are in various SERP pages.

Conclusions

This paper included a way to showcase the easy access to any kitchen remodeling resources with plenty of information through an OneUI Kitchen Remodeling web application, which has the nice

features to make it easy to exchange the ideas for a better customize view. The ideal goal of the project is to create a web application for the user who can manage the project of the kitchen remodeling in one view and which intern benefits the local professionals in each category, involved in this inside.

2. BACKGROUND

One UI Kitchen Remodeling project requires a good understanding of web application development with various skills. Technical skills like HTML5, CSS, and Javascript language and application as infrastructure setup to get it live which is highly performant, reliable, and scalable with less operational maintenance.

The infrastructure service of Amazon can be used for a better understanding of serverless techniques and deploying the application.

To increase the availability, strategies like blue-green are used.

3. RELATED WORK

Literature Review

One UI Kitchen Remodeling Project is to benefit people with various categories that are involved in kitchen remodeling and personalize each category, showing everything in one neatly designed UI page. People can choose each category specialist and look at their profile, previous work, and use coupons. The Publishers of these various categories can get registered and offer their services, coupons and upload their work. Through this local clients, specialists, suppliers can be benefited. We will essentially map out all of the local businesses and professionals needed to finish a Kitchen Remodel, basically bringing 10 different category searches in one view which makes the monetization potential.
The big advantage we will provide is that we are directing ready-to-buy customers to local businesses where they can look, touch and negotiate that backsplash that they want. No other company is doing this right now which is a major win. In the future, we will also provide personalizing the results based on budget or the project stage.
Coming to technical wise, for storage purposes, all the information is in a key-value store kind of database in Amazon Dynamo DB. For building Serverless Rest API service which will get the data from the DB and serve in AWS Lambda. And for building a website that will be hosted through Amazon S3 and takes the data and displays to the user and will be creating few dashboards.

One UI Kitchen Remodeling web application typically can be built using AWS Cloud services. It consists of multiple services that need to be coupled and hosted on the cloud. Making the application highly available and performant for each service hosting on a cloud will be the ideal and optimal solution.

For serverless web development (Sbarski & Kroonenburg, S. 2017) suggested how to design and architect the Rest API service and integrate with the frontend static webpages. (Hendrix. 2019) included creating the CRUD operations, supporting HTTPS, handling the cors issue, and other security concerns about Authentication and Authorization for building the API. Regarding the application deployment using a serverless framework that is compatible with AWS Lambda and deploys the Functions. As a part of the deployment process (Hendrix, R. 1983) the Cloud Formation templates will be generated and pushed from the command line and we don't need to manage or provision any servers.

For a highly performant and reliable application, we need to access the data with minimal latency and for a One UI Kitchen Remodeling application with minimal structured data, a NoSQL database is the ideal option. Dynamo DB is a NoSQL database (Rangel, D. 2015) that is a fully managed, serverless, scalable, and cost-effective option for the application. It supports all the major factors like data replication or fine-grain access control to the data with the support of schema-less which means neither the attributes nor the data types need to be defined beforehand.

The author Tavis, M., & Fitzsimons, P. (2012) added hosting a static website on AWS S3 service is reliable and very cost-effective with zero server-side code and consists of only HTML, CSS, and JavaScript. This has a huge benefit in terms of scale, performance, security, and cost. Since to reduce the operational maintenance and cost AWS S3 is one of the ideal solutions to host the website and also has a great community for help online. Configuring the website hosting can be done with AWS GUI's console and policies to the bucket that is created. A Cname record DNS mapping from the application domain name to the s3 website URL can also be done to access the website. AWS CloudFront can be used to deliver a low latency delivery of the application website.

4. APPROACH

A comparison of Houzz and Angie's List has been made regarding the usability and reviews in the previous works. It will evaluate how different to find the different professionals of each category of kitchen remodeling in various SERP pages, and how they handle people searching kitchen/bathroom remodels, and they land them on a dedicated project page. Where it's full of information but skewed in a way that makes it incomplete.

- User Requirement

In the User, Requirements face the project is about to concentrate on system-wise and business-wise. And concentrate on the User's point of view, their goal to achieve, and Outputs & Inputs of the User.
And also covers a few objectives and main goals which allow to help and achieve the Requirements. Implementing Agile using Sprint Methodology, and a few functional requirements could be User friendly for the Publisher.
And more efficiently should start developing about the few high-quality requirements like software product side that involves few jobs likes QA, Developers engineers that can handle the various scenarios who can handle the end-user requirements. And the main Roles and responsibilities to cover in the requirements face are Stakeholders, customers, Product Owners, development team.

- Design

One UI Kitchen Remodeling is a simple serverless web application where develops, management and deployment using AWS Cloud serverless. The application architecture uses AWS Lambda, Amazon API Gateway, Amazon DynamoDB, and AWS console to manage. Hosting static web resources including HTML, CSS, Javascript, and image files. The Javascript makes the calls from the browser and receives data from the backend API which is using AWS Lambda and API gateway. The AWS DynamoDB stores the data from the Lambda function and provides the persistence layer.

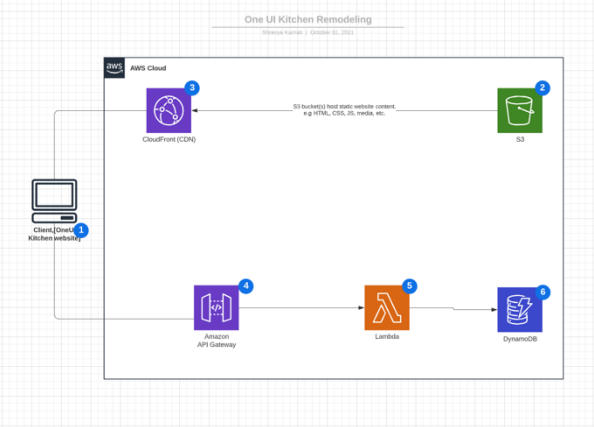

Figure 1. Design Architecture OneUI

- Implementation

Building a serverless HTTP API using Serverless Framework, easy to build the API using AWS Lambda and integrate with AWS DynamoDB. Since serverless depends on CloudFormation for provisioning the microservice and scaffolds the complete project structure. It will also take care of deploying the functions.

For the local development, prerequisites need to be installed like brew, npm, UUID, bluebird, serverless, etc. Creating a Node.js serverless project directory and executing the serverless command to create the project and multiple initial files, like. npmignore, handler.js and serverless. As an example, to go over one of the CRUD operations APIS, REST resources for creating the local publishers. Defined publisher Create function when the HTTP POST request is made to /publisher/create then API/publisher. Create handler function will be invoked and create a publisher of the specific category with unique UUID and store in DynamoDB.

For securing the connection, iamRoleStatements are added to define which actions are permissible. A lot of metadata can be configured in the serverless like specifying the memory to allocate to the function. Created the function to list all the publishers, get a specific publisher or update the data of a publisher. Serverless Framework helps to invoke the functions locally, remotely, and tail logs for each function's

- Technologies Used

In web application for storing all the information in a key0value store kind of database is done by Amazon DynamoDb.

For building a serverless RestAPI service which will get data from the DB and server in AWS Lambda, for a website hosting with Amazon S3 and takes the data and display to the user.

5. DATA COLLECTION

One UI Kitchen Remodeling is specifically concentrating on 10 different categories that are related to Kitchen Remodeling. The categories are General Contractor, Plumber, Electrician, Cabinets, Appliances, Flooring, Countertops, Painters, Faucets/Sink, Light Fixtures. The dataset for this application is provided by Yelp. This data is provided in JSON format of each listing which includes location data, attributes of listing, and categories which is an array of strings of business categories that the listing belongs to. Since the primary assumption is that the application supports only 1 location for simplicity, we have to extract the data of that location listings from the dataset. So, we run an ETL job to extract the listings that are related to the above categories as mentioned and also filter on the location Bellevue, postal code 98005. All this data is inserted into the DynamoDb tables and served using the Lambda functions rest API.

And adding the data for the project, Here I'm concentrating to add a few steps to prep the better outcome, like Adding Data exploration, data cleansing, data blending data profiling, data wrangling, and Extract transfer load, In the Yelp data set, they have subsets for the help and easy access to the individual business, Here I'm going to concentrate on few subsets for my application deployment. And I'm going to concentrate on the particular city of Bellevue and using to extract the data for pin:98005.

Reading every business listing file of the yelp dataset and uploading the data to Amazon DynamoDB.

```
import json
import pandas as pd
data_file = open("yelp_academic_dataset_checkin.json")
data = []
for line in data_file:
&amp;nbsp;data.append(json.loads(line))
checkin_df = pd.DataFrame(data)
data_file.close()
```

Figure 2

6. DATA ANALYSIS

As part of data analysis, the dataset from yelp is a local business directory service with different attributes for each business listing. It allows users to give ratings, reviews and provide the opening hours and closing hours. Kitchen Remodeling is divided into 10 different categories and from the dataset, filtered the listings which belong to these categories and uploaded to the database.

S.No	Category
1	Plumbing
2	General Contractor
3	Electrician
4	Cabinets
5	Appliances
6	Floors
7	Countertops
8	Painters
9	Faucets/sink
10	Light fixtures

Figure 3. Category table

The business JSON file has attributes such as business ID, business name, business location, rating stars, count of reviews, and the information whether the listing is open or not. For each listing, the dataset has the mapping to a set of different categories the listing belongs to or the information of different types.

For data clean up and pre-processing, which is the important task for us to be performed, as to load the business listings data which are mapped to the Kitchen Remodeling categories. Also removed unnecessary column information which is not required for the current project scope and deleting the null value records for the above categories. Analyzed all the given business listings are unique based on the business_id uniqueness.

```
table = dynamodb.create_table(
    TableName='BusinessListing',
    KeySchema=[
        {
            'AttributeName': 'business_id',
            'KeyType': 'HASH'  # Partition key
        },
        {
            'AttributeName': 'name',
            'KeyType': 'RANGE'  # Sort key
        }
    ],
    AttributeDefinitions=[
        {
            'AttributeName': 'business_id',
            'AttributeType': 'S'
        },
        {
            'AttributeName': 'name',
            'AttributeType': 'S'
        },
        # Below attributes will be added.
        # {
        #     'AttributeName': 'stars',
        #     'AttributeType': 'N'
        # },
        # {
        #     'AttributeName': 'address',
        #     'AttributeType': 'S'
        # },
        # {
        #     'AttributeName': 'city',
        #     'AttributeType': 'S'
        # },
        # {
        #     'AttributeName': 'postal_code',
        #     'AttributeType': 'S'
        # },
        # {
        #     'AttributeName': 'review_count',
        #     'AttributeType': 'N'
        # },
        # {
        #     'AttributeName': 'is_open',
        #     'AttributeType': 'N'
        # },
        # {
        #     'AttributeName': 'categories',
        #     'AttributeType': 'S'
        # },

    ],
    ProvisionedThroughput={
```

Figure 4

S.No	Attribute Name	Data Type
1	business_id	String
2	name	String
3	address	String
4	city	String
5	postal_code	String
6	stars	float
7	review_count	Integer
8	is_open	Integer (1 or0)
9	hours	Integer
10	categories	Object

Figure 5

Several listings are associated with kitchen remodeling categories for each state in the given dataset and plot the graph for the Top 50 cities by business in the dataset. Analyzed the distribution of stars attribute for the business listings and corrected the null values with a default value. Several listings are still open and that is closed.

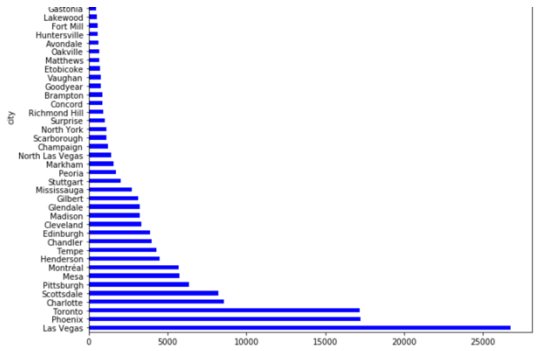

Figure 6. Kitchen Remodeling Search plot

After analyzing and preprocessing the dataset for all the category listings and then uploading the data to the tables created in the database as a one-time job. (Amazon, region)

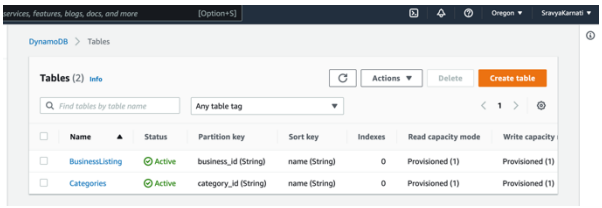

Figure 7. Dynamo DB Table

7. FINDINGS

Serverless Framework: - The One UI Kitchen Remodeling web application uses the serverless framework to deploy and not to worry about the uptime or manage any servers. This helps to automatically scale depending on the load and manage the AWS resources to create or update.

Sequence Diagram: - The UML flow of the OneUI kitchen remodeling web application.

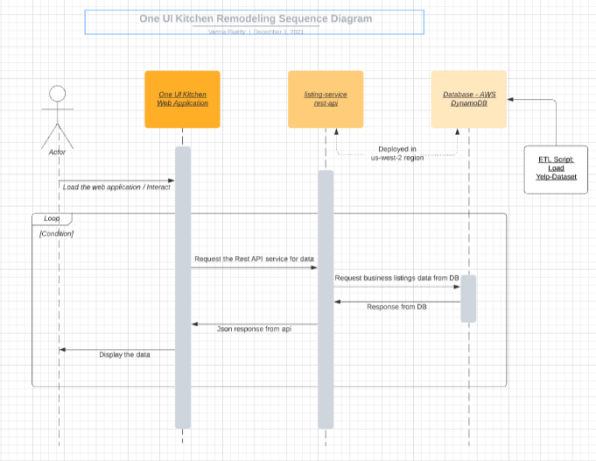

Figure 8. Data Flow Diagram

Rest API service: - The listing service REST API service supports 5 different lambda functions which are get-listing, create-listing, update-listing, delete-listing, get-all-listings. These APIs are developed using the AWS SDK for JavaScript DynamoDB Client for Node.js. This is modelized by clients and commands. To send a request, you only need to import the DynamoDB Client and the commands you need. The response of each API has the marshal data - DynamoDB record data and unmarshal data - JavaScript Object (JSON).

For running the service locally, it's a simple step process of installing all the dependencies and executing a serverless offline start command to test the APIs.

```
~/api-service/listing-service git:(main) ✗ sls offline start --skipCacheInvalidation -s dev
Serverless: Running "serverless" installed locally (in service node_modules)
Serverless: Deprecation warning: Detected unrecognized CLI options: "--skipCacheInvalidation".
            Starting with the next major, Serverless Framework will report them with a thrown error
            More Info: https://www.serverless.com/framework/docs/deprecations/#UNSUPPORTED_CLI_OPTIONS
offline: Starting Offline: dev us-west-2.
offline: Offline [http for lambda] listening on http://localhost:3002
offline: Function names exposed for local invocation by aws-sdk:
           * getListing: get-listing
           * createListing: create-listing
           * updateListing: update-listing
           * deleteListing: delete-listing
           * getAllListings: get-all-listings

     GET     | http://localhost:3000/dev/listing/{id}
     POST    | http://localhost:3000/2015-03-31/functions/getListing/invocations
     POST    | http://localhost:3000/dev/listing
     POST    | http://localhost:3000/2015-03-31/functions/createListing/invocations
     PUT     | http://localhost:3000/dev/listing/{id}
     POST    | http://localhost:3000/2015-03-31/functions/updateListing/invocations
     DELETE  | http://localhost:3000/dev/listing/{id}
     POST    | http://localhost:3000/2015-03-31/functions/deleteListing/invocations
     GET     | http://localhost:3000/dev/listings/{categoryId}
     POST    | http://localhost:3000/2015-03-31/functions/getAllListings/invocations

offline: [HTTP] server ready: http://localhost:3000 🚀
offline:
offline: Enter "rp" to replay the last request
```

Figure 9

Deployment: - The web application is hosted on AWS S3, and the Rest API service is an AWS serverless lambda function. The rest API service is managed using the serverless framework and hosted on AWS. All the resources can be created or updated in the file serverless.yml file. For the deployment of the service, there are 2 different approaches. First manually, executing the serverless deploy command from the root folder.

```
~/api-service/listing-service git:(main) ✗ sls deploy
Serverless: Packaging service...
Serverless: Excluding development dependencies...
Serverless: Uploading CloudFormation file to S3...
Serverless: Uploading artifacts...
Serverless: Uploading service listing-service-api.zip file to S3 (2.37 MB)...
Serverless: Validating template...
Serverless: Updating Stack...
Serverless: Checking Stack update progress...
.....................................
Serverless: Stack update finished...
Service Information
service: listing-service-api
stage: dev
region: us-west-2
stack: listing-service-api-dev
resources: 40
api keys:
  None
endpoints:
  GET - https://cqdqvaqgw0.execute-api.us-west-2.amazonaws.com/dev/listing/{id}
  POST - https://cqdqvaqgw0.execute-api.us-west-2.amazonaws.com/dev/listing
  PUT - https://cqdqvaqgw0.execute-api.us-west-2.amazonaws.com/dev/listing/{id}
  DELETE - https://cqdqvaqgw0.execute-api.us-west-2.amazonaws.com/dev/listing/{id}
  GET - https://cqdqvaqgw0.execute-api.us-west-2.amazonaws.com/dev/listings/{categoryId}
functions:
  getListing: get-listing
  createListing: create-listing
  updateListing: update-listing
  deleteListing: delete-listing
  getAllListings: get-all-listings
layers:
  None

Improve API performance - monitor it with the Serverless Dashboard: run "serverless"
```

Figure 10

The second approach is automated using the GitHub workflow actions pipeline. This way the rest API service is Automated, customized, and executed in the repository with GitHub Actions.

Once the new branch is merged to the main branch for the rest API listing service, there is a workflow pipeline that gets triggered for every new commit on the main branch. This workflow executes a series of steps that will install decencies, run the prerequisite steps and do a serverless deploy and create or update all the necessary resources. This requires creating a generic IAM user and copying the AWS secret key and AWS secret access id to the GitHub secrets.

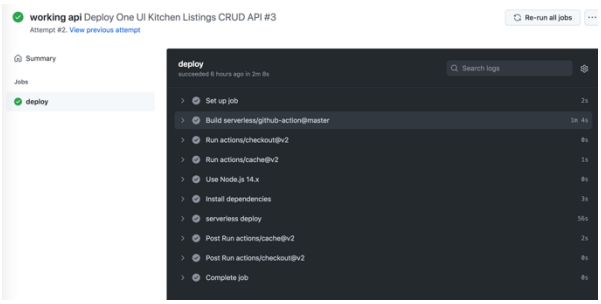

Figure 11

8. CONCLUSION

The One UI Kitchen Remodeling web application has all the information in a one-page view and is organized for better understanding, navigation of information users. Also, it helps the small

local business and professionals with more visibility and a medium to reach the users. This way the customers who are interested in the professional of a specific category can use their coupons, contact them, read the reviews, cost analyze and review their past work photos. Many features can be added and extend the use cases of the One UI Kitchen Remodeling idea.

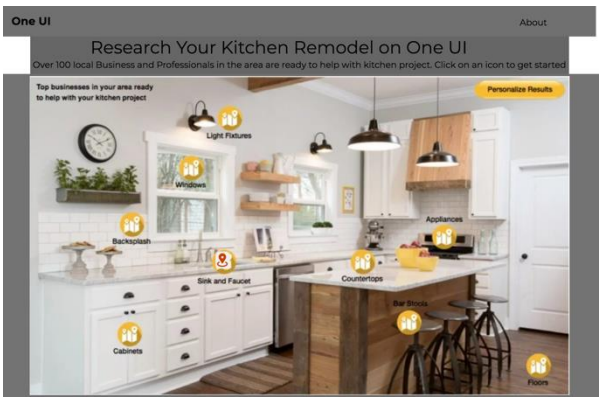

Figure 12

Future Work

One UI Kitchen Remodeling can be extended extensively by adding multiple features for the web application. Firstly, the functional requirements would be, extending the support to search for local professionals for each category in most of the cities in the U.S. Also, support for providing the time slots for the local professionals to come home. Create a view to adding reviews or ratings to any business listing. Secondly, the nonfunctional requirements would be creating the sub-domain namespace for the rest of the API service. Adding authentication for the API and support of sign-up/sign-in for the customers.

REFERENCE

Mitchell, T. (2017, September 21). Someone's in the kitchen The Ergonomics of Cooking and Kitchen Design. http://www.working-well.org/articles/pdf/Cooking_2017.pdf. Retrieved October 16, 2021, from http://www.workingwell.org/articles/pdf/Cooking_2017.pdf

kritikos IEEE Xplore, K. (2018, December 20). A review of serverless frameworks. IEEE Xplore. Retrieved October 17, 2021, from https://ieeexplore.ieee.org/abstract/document/8605774.

Sbarski, Kroonenburg, (2017). Serverless architectures on Aws. Manning.

Hendrix, R. (1983). Lambda. Retrieved October 18, 2021, from https://aws.amazon.com/lambda/

Rangel, D. (2015). DynamoDB: Everything you need to know about Amazon Web Service's NoSQL database. Retrieved October 18, 2021, from

Hendrix, R. (2019). Lambda. Retrieved October 18, 2021, from https://docs.aws.amazon.com/lambda/latest/dg/nodejs-handler.html

Li, X., & Kothurkar, A. (2002). Amazon S3. Retrieved October 18, 2021, from https://aws.amazon.com/s3/

Web application hosting in the AWS cloud. Site Point.

Doglio, F. (2015). Pro rest API development with node.js. Google Books. Retrieved October 18, 2021, from https://www.google.com/books/edition/Pro_REST_API_Development_with_Node_js/kjUwCgAAQBAJ?hl=en&gbpv=1&dq=rest%2Bapi%2Bnode%2Bjs&pg=PR7&printsec=frontcover.

Jacyntho, M. D., Schwabe, D., & Rossi, G. (2002). A software architecture for structuring complex web applications. J. Web Eng., 1(1), 37-60.

Pore, A. (2019, June 27). How to design schema for your NoSQL database? Retrieved October 18, 2021, from https://www.dataversity.net/how-to-design-schema-for-your-nosql-database/

Houzz [Advertisement]. (2021). Retrieved October 17, 2021, from https://www.houzz.com/aboutUs

Mather, L., & Weiss, A. (2016, March 21). 10 questions to ask yourself before your kitchen remodel. Retrieved October 28, 2021, from https://www.architecturaldigest.com/story/questions-ask-yourself-before-kitchen-remodel

US-west-2.Console.aws.amazon.com. (n.d.). Retrieved November 22, 2021, from https://us-west-2.console.aws.amazon.com/secretsmanager/home?region=us-west-2#!/secret?name=

DynamoDB client - AWS SDK for JavaScript V3. (n.d.). Retrieved December 2, 2021, from https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/clients/client-dynamodb/index.html.

Workflow syntax for GitHub actions. GitHub Docs. (n.d.). Retrieved December 2, 2021, from https://docs.github.com/en/actions/learn-github-actions/workflow-syntax-for-github-actions#onpushpull_requestbranchestags.

Serverless framework - AWS lambda guide - credentials. Serverless Framework - AWS Lambda Guide - Credentials. (n.d.). Retrieved December 2, 2021, from https://www.serverless.com/framework/docs/providers/aws/guide/credentials/.

Serverless framework - AWS lambda guide - functions. Serverless Framework - AWS Lambda Guide - Functions. (n.d.). Retrieved December 2, 2021, from

https://www.serverless.com/framework/docs/providers/aws/guide/functions.

Github link: -
https://github.com/karnatisravya/listing-service
https://github.com/karnatisravya/ETL-script

Demo

Google Drive: -
https://drive.google.com/file/d/18SHaw-QPyslL0rZ1Eu_av-_GYqZw_bbZ/view?usp=sharing

YouTube Link: -https://youtu.be/JBRIDC4r-TA