PIXTRACT

A searchable image storage database

CONTENT

Source: **https://github.com/kshanmu1/pixtract**

1. MOTIVATION

To leverage state of the art machine learning and image processing algorithms to create a cloud-native, media storage and search application, entirely built using a serverless architecture. This application will enable images to be stored on the cloud, as lll as perform automatic image tagging, recognition, categorization and digital notes extraction to enable a global search interface where every media content uploaded is text-searchable.

Through this project, I aim to learn and gain hands-on experience with a range of AWS services, including but not limited to machine learning and image recognition services, serverless computing, and authentication services. I also plan to explore and experiment with new features and services, such as OpenSearch, to provide a scalable and performant search interface for our users.Overall, our application has the potential to significantly improve the user experience of image management and searching while showcasing the poIr of cloud-native solutions and state-of-the-art machine learning algorithms.

2. FEATURES

2.0. Client Application

All the features displayed below are showcased on the client application which was entirely built using Vue.js mostly locally (with some help from AWS amplify). The Vue app has a modern UI, a smooth UX and enables the users to use the features of the app seamlessly.

2.1. Signup and login:

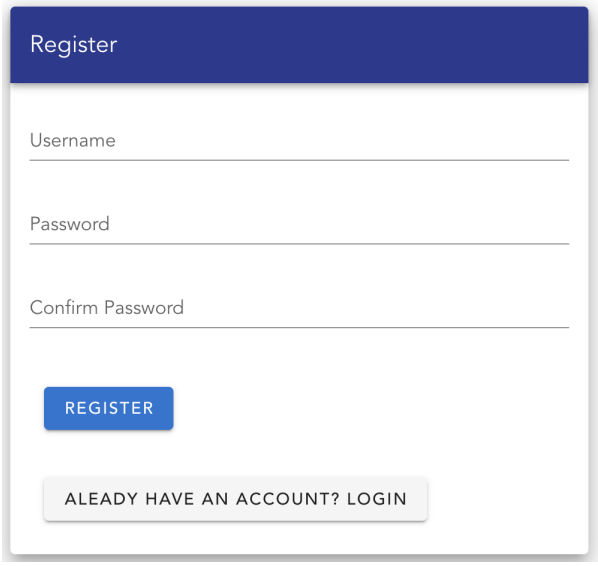


Fig 3.1. Signup

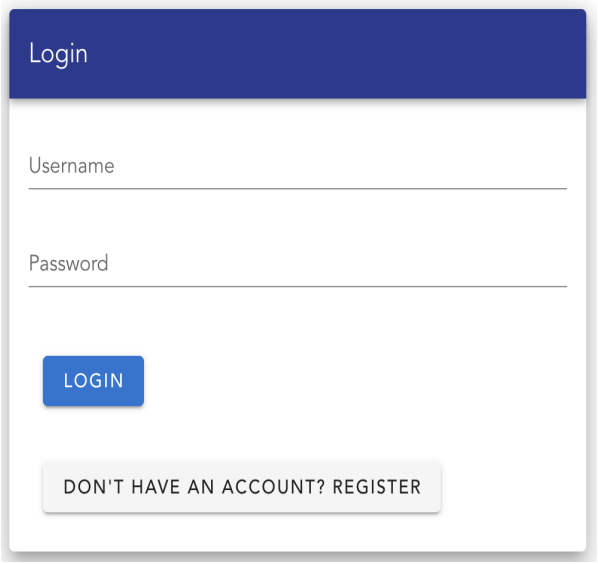


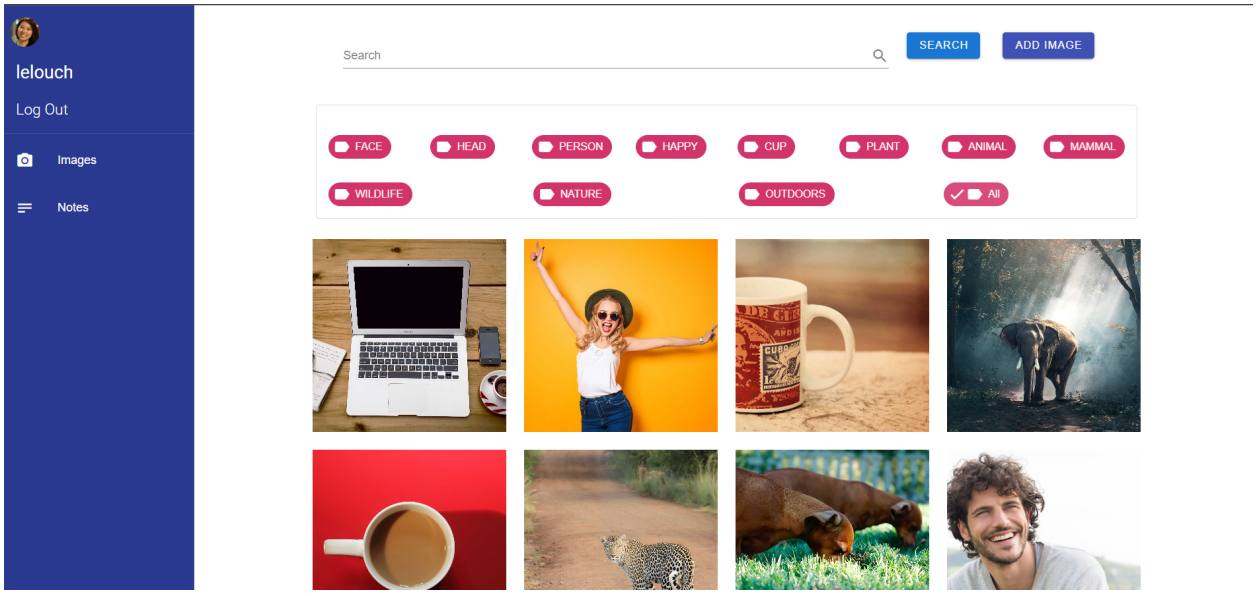
Fig 3.2. Login

The signup feature of the Pixtract Ib application provides a user-friendly interface for new users to create an account. When a user signs up for the first time, they will be asked to provide a unique username and password. To ensure that the user has entered the correct password, they will be asked to type it twice.

If the entered username already exists, S3 will authenticate the user's credentials and send back a token to log in. Holver, if it is a new user, the application will create a new user and assign a unique sub ID, which will be associated with the user's account. The application will then return a token, which will be used to authenticate the user's subsequent requests.

This signup feature not only allows new users to easily create an account, but also ensures that each user has a unique sub ID associated with their account. This sub ID can be used to identify the user's uploaded images and metadata, enabling efficient searching and categorization of their content.

2.2. Image Tagging & Metadata :



The Pixtract application has a feature called "Image tagging & Metadata". The aim of this feature is to help users organize and describe the digital images they upload. All the uploaded images are stored using AWS S3 buckets.

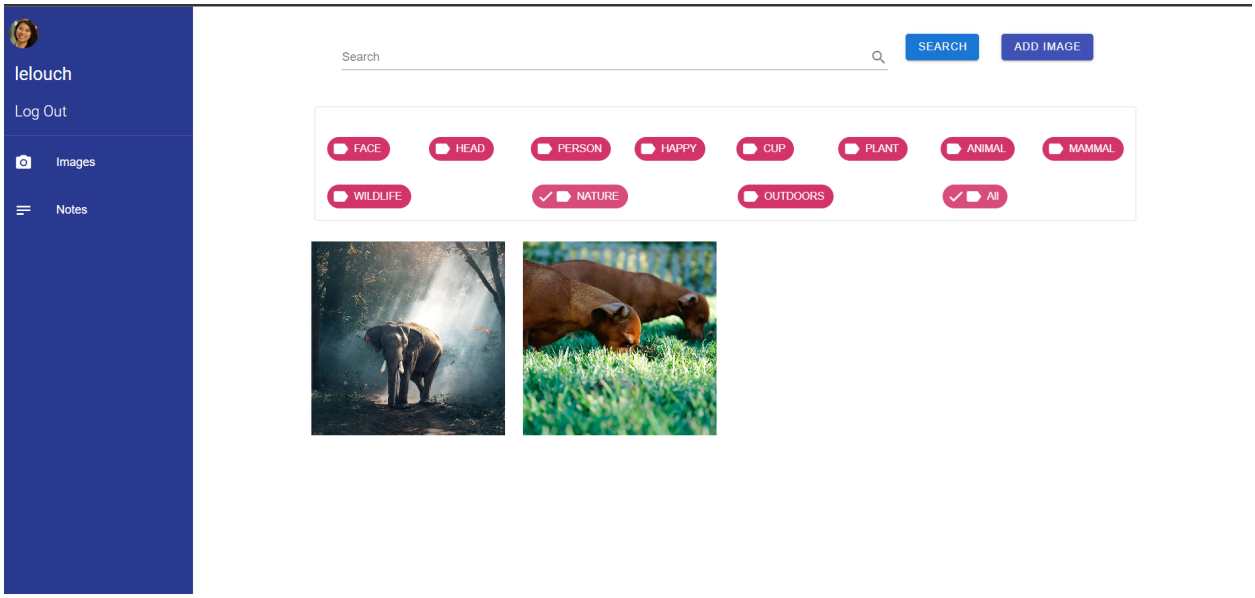
This feature makes the content in the application more understandable and easily accessible to users. To find specific images, users can simply search using a keyword, and all images that match the keyword will be displayed on the screen.

Image tagging involves assigning keywords or labels to an image to describe its contents or attributes. Rather than requiring users to tag their images manually, I've integrated Amazon Rekognition services into our application to automatically extract the metadata from uploaded images. This means that our application is able to analyze the content of an image and automatically assign relevant tags to it.

By implementing this feature, I've made it easier for users to manage their images, as they can now easily find specific images they're looking for by searching for relevant tags. This feature can also save users time and effort as they no longer need to spend time manually tagging their images.

In the above image, it can be seen that there are many tags that have been generated for the images automatically (displayed in pink on top of the images). Whenever a new image is uploaded, new tags are automatically added.

2.3 Auto Categorization :



The Pixtract application has another interesting feature called "Auto Categorization". This feature is designed to automatically categorize images uploaded by users based on their content.

Whenever a user uploads an image, the contents of the image are automatically extracted. If there are other images in the application that have similar content, tags will be generated for those images, and all images with those tags will be categorized together.

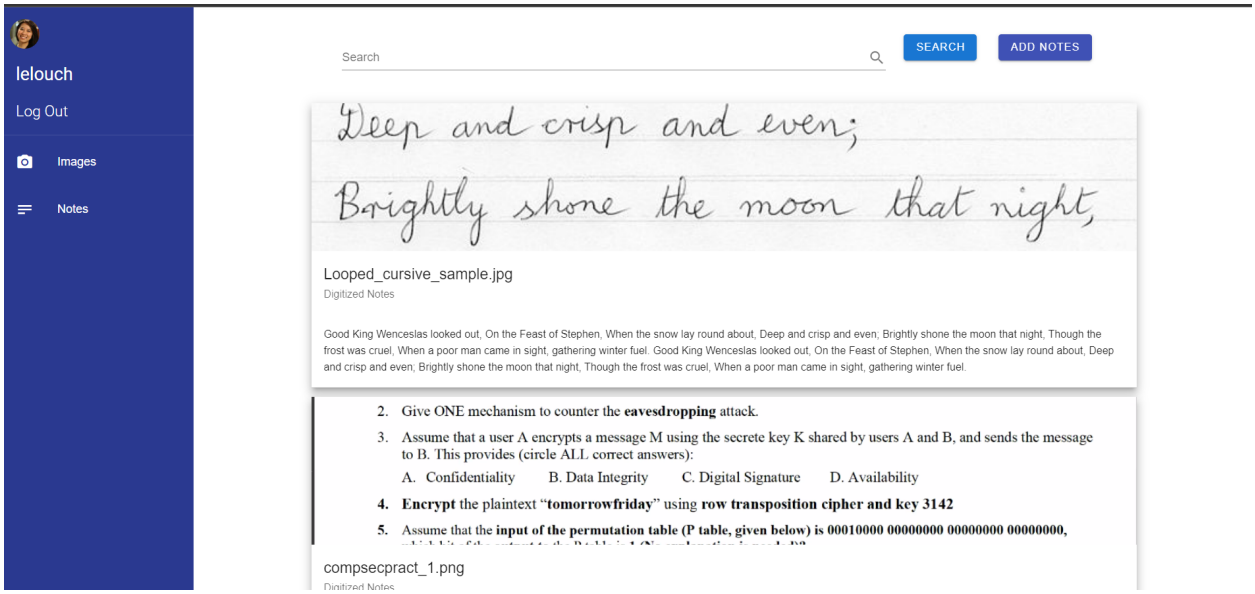
In the example shown in the figure, I can see that both images have common tags such as "Nature" and "Animal". As a result, these images are categorized into those tags. When a user selects a particular tag, they can view all the images that are related to that tag.

To implement this feature, I've leveraged the poIr of Amazon Rekognition service to extract the tags from the images. Once the tags are extracted, I categorize the images according to their corresponding tags.

With this feature, users can easily browse through their images and quickly find related images.

In the above image, it can be noted that the ‘Nature’ filter is selected, and the images are filtered using metadata from AWS Rekognition to only show images which have a nature component in them.

2.4. Notes Digitization :



The Pixtract application has an incredibly useful feature called "Notes Digitization". This feature allows users to upload their text or handwritten notes in either image or PDF format. Once uploaded, our application automatically extracts the text from the uploaded file and converts it into simple plain text that can be easily read and displayed on the screen.

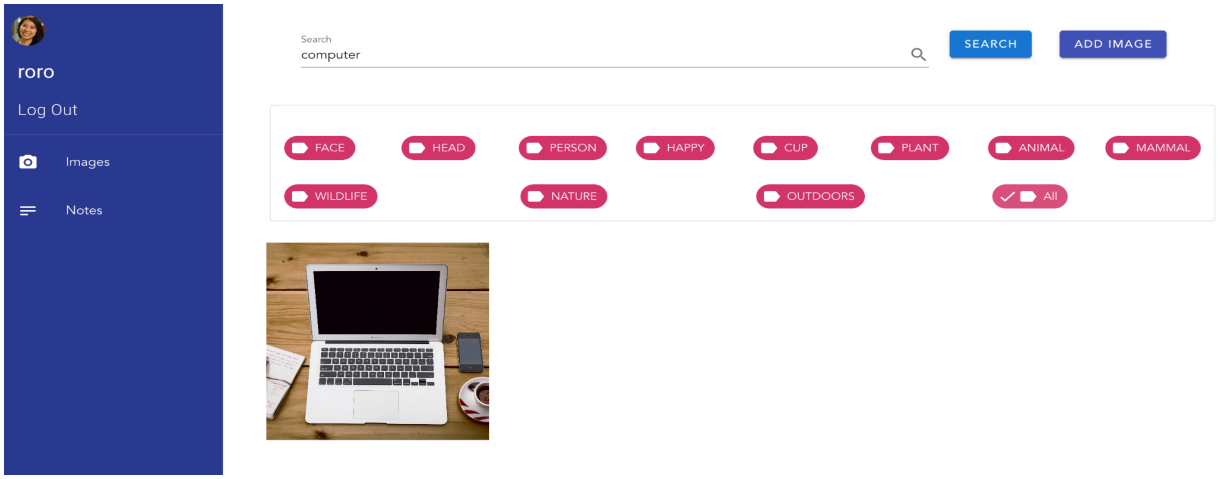
This text-extraction feature works for both handwritten notes as lll as printed documents such as bills or forms.

In addition to this, users can also search for specific keywords within their uploaded notes. Our application matches the keyword with the uploaded notes and retrieves any notes that contain the keyword, displaying them in the feed.

To implement this feature, I've integrated Amazon Textract service into our application. This service has helped us to extract the text from the uploaded documents, making it possible to display the notes in a readable format.

With this feature, users no longer have to manually transcribe their notes or search through handwritten notes for specific information. Instead, they can quickly and easily digitize their notes and search for specific keywords within them.

2.5. Text Search



The OpenSearch (Elastic Search) service is a poIrful tool that allows users to search for specific images or notes using keywords. Users of the Pixtract Ib application can search for images based on tags or any keywords that may be associated with the image, and can also search for notes associated with the image.When a user inputs a keyword, the application runs a query through the metadata of the images and notes stored in the system. If there is a match betIen the user's keyword and any metadata associated with the images or notes, the application returns the relevant images or notes to the user.

This search functionality is a poIrful tool for users who want to find specific images or notes quickly and easily. It provides an efficient and effective way for users to organize their media content, and allows them to easily retrieve their content based on keywords and tags. While the current implementation of the search feature in Pixtract is limited to basic searching using metadata, there is potential for further development and integration of more advanced querying capabilities using Lucene syntax.

3. ARCHITECTURE

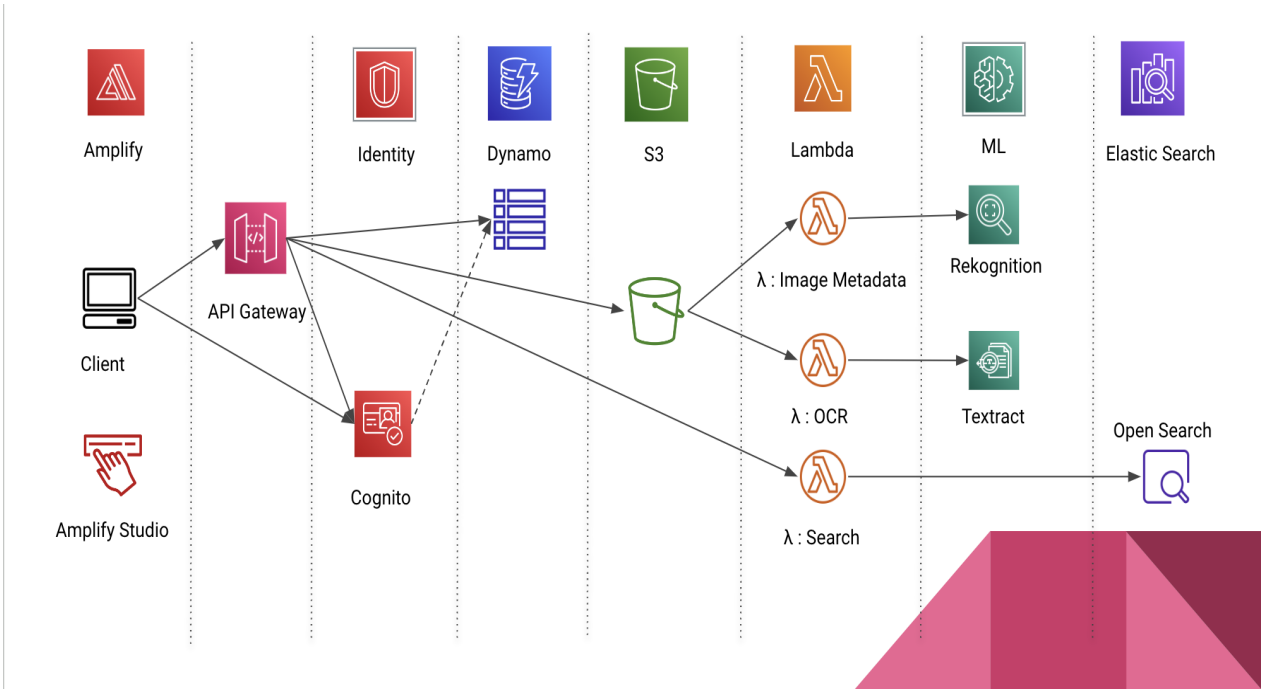


Fig 2.1. Architecture of the pixtract

In Figure 2.1, I present the complete architecture of the Pixtract Ib application. I have made use of several AWS services, including but not limited to Amazon S3, Amazon DynamoDB, AWS Lambda, Amazon API Gateway, Amazon Rekognition, and Amazon Cognito, to implement the different components of the application. For instance, Amazon S3 is used to store and retrieve images uploaded by users, while Amazon DynamoDB is used to store user profile data and session information. AWS Lambda is used to execute serverless functions that perform automated image tagging and recognition, whereas Amazon API Gateway provides a secure and scalable REST API for the frontend and backend to communicate.

Furthermore, I have also integrated Amazon Rekognition, which is a deep learning-based image and video analysis service, to provide advanced features such as facial recognition, object detection, and scene analysis. Finally, I have utilized Amazon Cognito, which is a user authentication and authorization service, to manage user sign-up, sign-in, and access control.

Here, I provide a brief overview of the different AWS services and how they are integrated into our application:

3.1. Amplify: AWS Amplify is a development platform that provides a set of tools and services for building cloud-poIred mobile and Ib applications. In our Pixtract Ib application, I have used

Amplify to streamline the development process and simplify the integration of various AWS services into our application.One of the key benefits of Amplify is its ability to automate the deployment and configuration of cloud infrastructure for our application. This includes the creation of AWS resources such as S3 buckets, DynamoDB tables, and Lambda functions, as Ill as the integration of these services with other AWS tools such as API Gateway and Cognito.

3.2. API Gateway: API Gateway is a fully managed service that enables us to create, publish, and secure APIs at any scale. It provides us with a scalable and cost-effective solution for creating a RESTful API that connects the frontend and backend of our application. Using API Gateway, I have created several APIs that enable our frontend to communicate with other services, such as Lambda functions and DynamoDB databases. This enables us to perform a range of functions, such as user authentication, data retrieval, and data storage, all of which are essential for a cloud-native, serverless application like Pixtract.

3.3. Cognito: Cognito is a user authentication and authorization service that makes it easy for us to add sign-up, sign-in, and access control functionality to our application. With Cognito, I can securely authenticate users and control access to our application's resources. Using Cognito, I have created APIs for user signup and signin, allowing users to authenticate themselves using a username and password.When a user signs up or signs in to our application, Cognito returns a user token, which I use to authenticate the user and authorize them to perform certain actions within our application. Additionally, Cognito provides us with a unique user ID, known as the 'sub' (or subject) of the user, which I can use to identify individual users and track their activity within our application.

3.4. Dynamo: DynamoDB is a fast and flexible NoSQL database that enables us to store and retrieve data at any scale. I use DynamoDB to store metadata related to uploaded images.Specifically, I use DynamoDB to store the metadata associated with each image uploaded to our application.When a user uploads an image to Pixtract, I store the image file itself in an Amazon S3 bucket, and then extract its metadata using AWS Rekognition and Textract. This metadata includes information such as image tags, labels, and text extraction. Once the metadata has been extracted, I save it to DynamoDB, where I can perform complex queries and retrieve the data quickly and efficiently. By using DynamoDB, I have been able to build a scalable and flexible data storage solution that can easily handle the large volume of images and associated metadata generated by our application.

3.5. S3: S3 is a highly scalable, durable, and secure object storage service that provides us with a reliable and cost-effective solution for storing and serving static content, such as images and video files. When a user uploads an image to Pixtract, I store the image file in an S3 bucket.In addition to uploading images, our users can also download their images from S3

3.6. Lambda: Lambda is a serverless computer service that enables us to run code without provisioning or managing servers. I use Lambda to execute serverless functions that perform automated image tagging and recognition, enabling us to easily categorize and label images for easy search and retrieval.

3.7. Textract: Amazon Textract is a machine learning (ML) service that automatically extracts text, handwriting, and data from scanned documents. It goes beyond simple optical character recognition (OCR) to identify, understand, and extract data from forms and tables

3.8. Rekognition: Amazon Rekognition is based on the same proven, highly scalable, deep learning technology to analyze billions of images and videos daily. Amazon Rekognition includes a simple, easy-to-use API that can quickly analyze any image or video file that's stored in Amazon S3

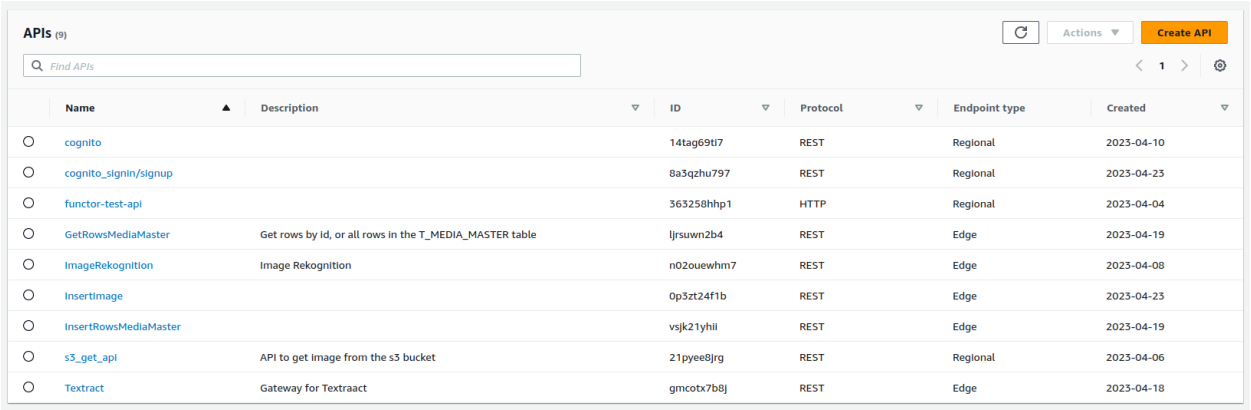
3.9. Open search: Amazon OpenSearch is a search and analytics engine that I have used in our Pixtract Ib application to provide advanced search functionality for our users. With OpenSearch, I are able to perform complex search queries on the metadata associated with each image uploaded to our application. I have created APIs in our application for creating an index, adding documents to the index, and performing searches against the index using a simple match query. While I Ire not able to fully integrate these APIs with our application, I Ire able to implement the text search using meta data of the images.

4. IMPLEMENTATION

4.1 APIs

API Gateway

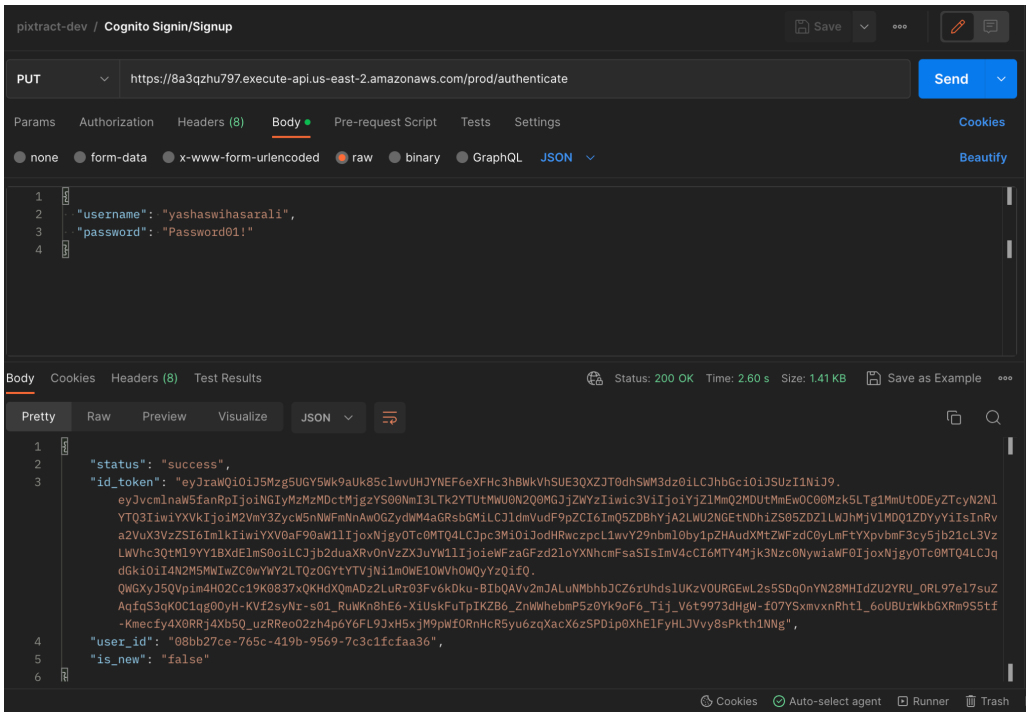
All the AWS Services are exposed to the Client Application using the API Gateway. Each of these APIs are load-balanced and Edge-Optimized for optimal performance across the globe. The APIs are shown below. In the last section, I have open sourced our git repository with a Postman test suite with all these endpoints to interact with.



The following APIs are illustrated via requests and responses on Postman.

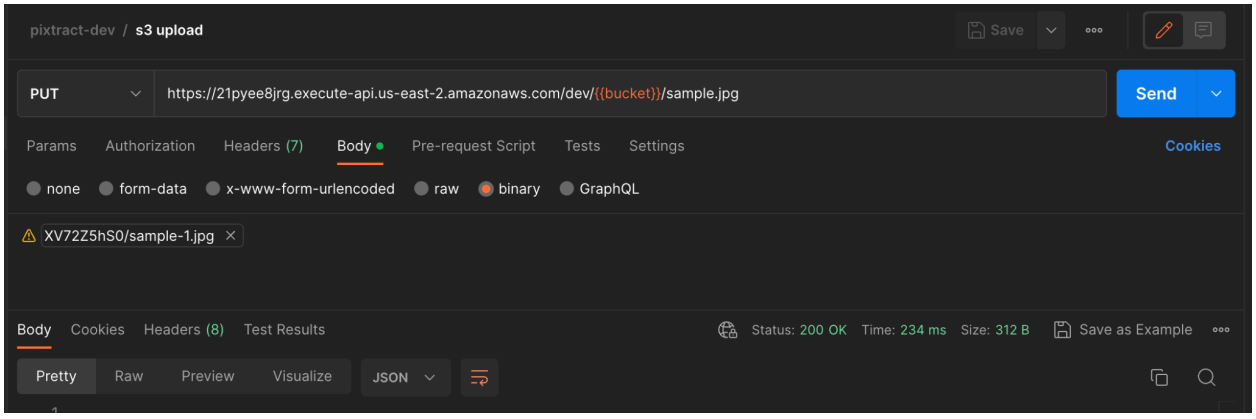
4.1.1. Cognito - Signup/Signin

This API interacts with AWS Cognito, and expects the username and password in the form of body. And the response will contain status, id_token, user_id/sub_id and is_new with 200 status.



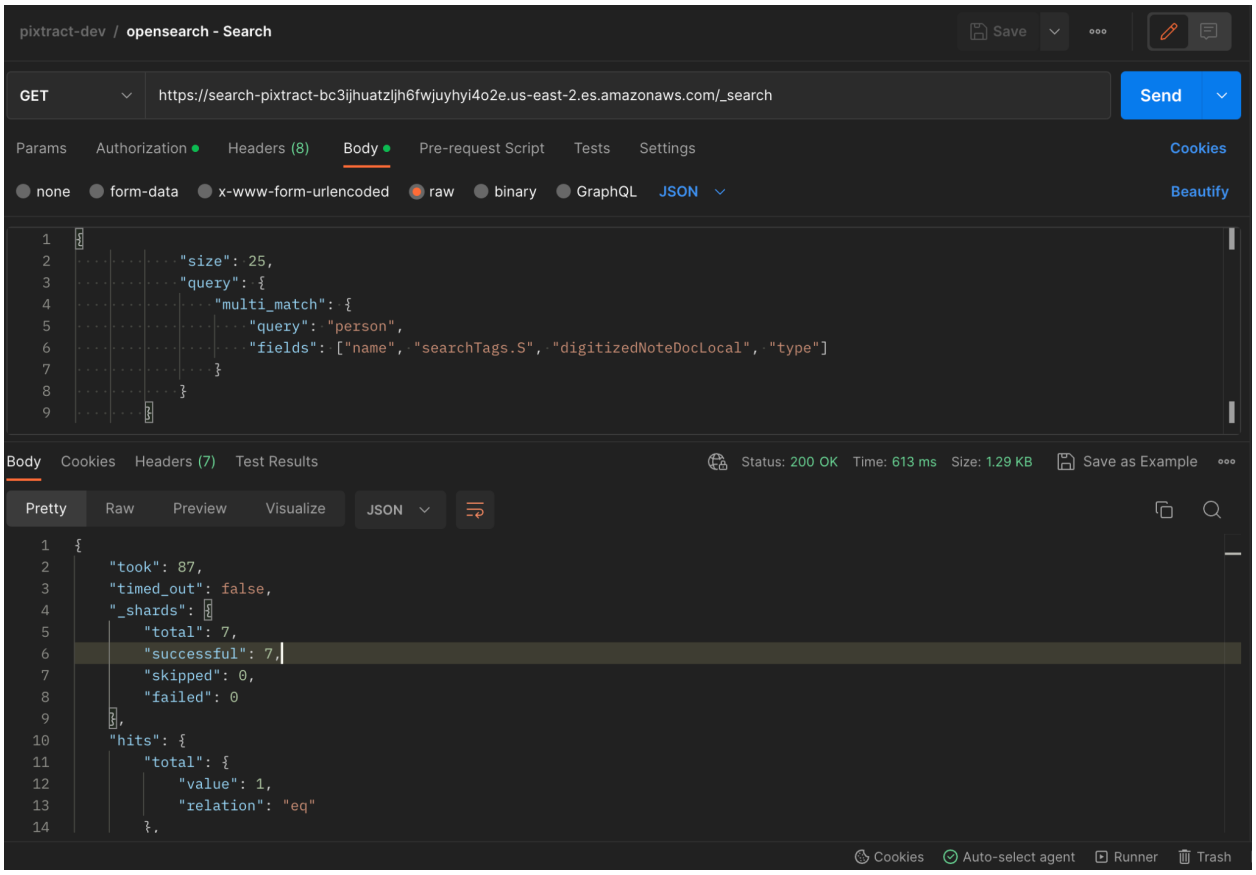
4.1.2. S3 - Put image

The API interacts with the AWS S3 service, andexpects the bucket name and file name as a query param and the body should have an image or document in the form of binary. And the response will be empty with 200 status.



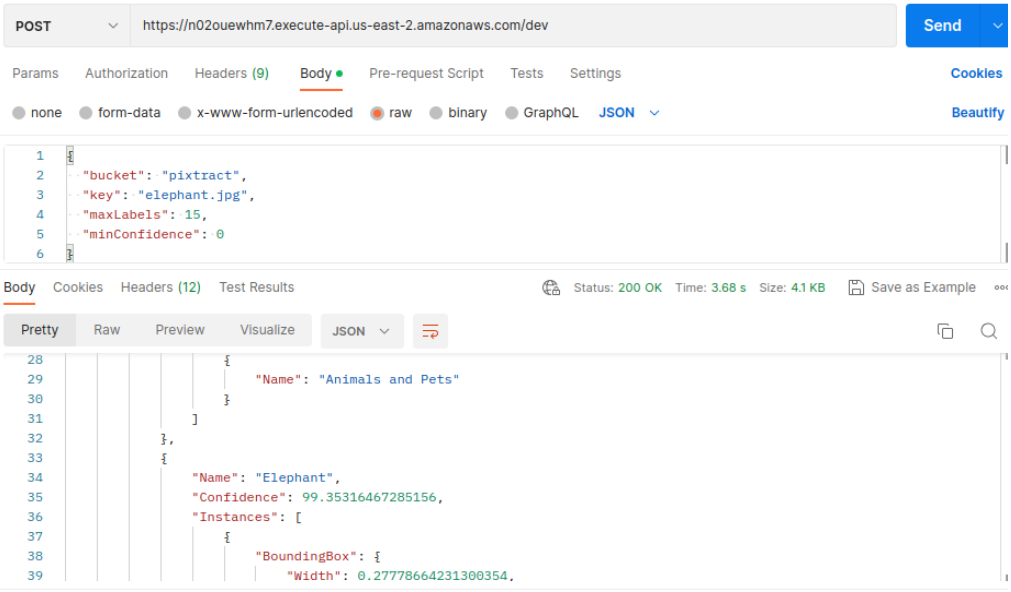
4.1.3. Open search - Search image/Notes

The API expects the search query in the form of body. And the response will contain few details of the results and matched images along with the metadata of the image.



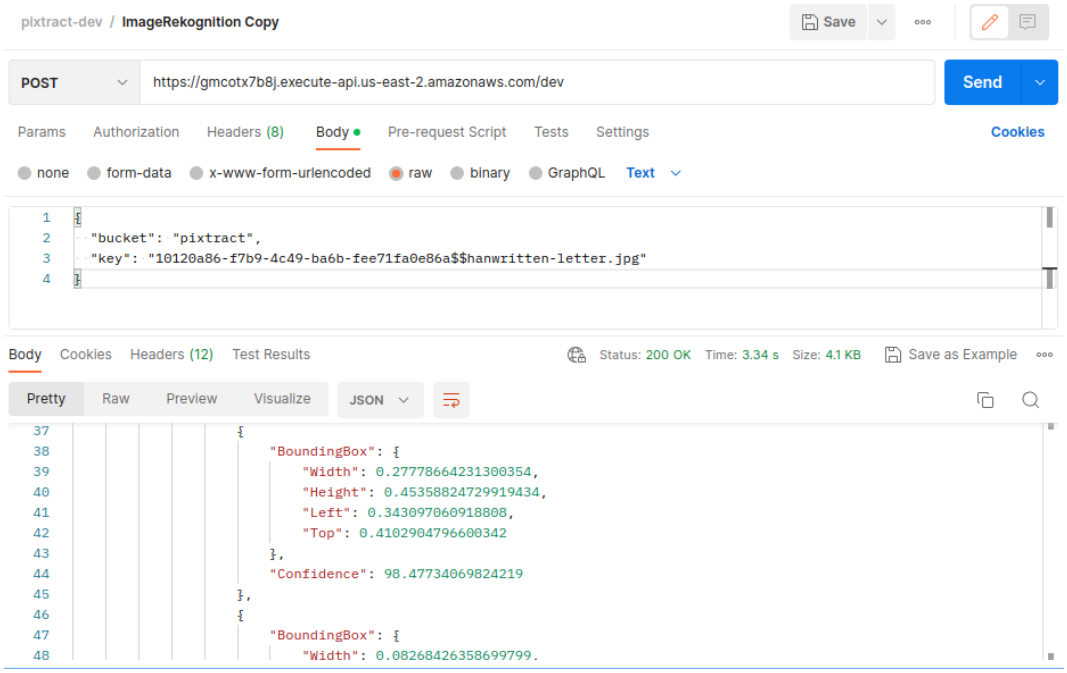
4.1.4. Rekognition

The API expects the bucket name and file name as a query param and response will be the metadata of the file with 200 status.



4.1.5. Textract

The API expects the bucket name and file name as a query param and response will be the extracted text of the file with 200 status.

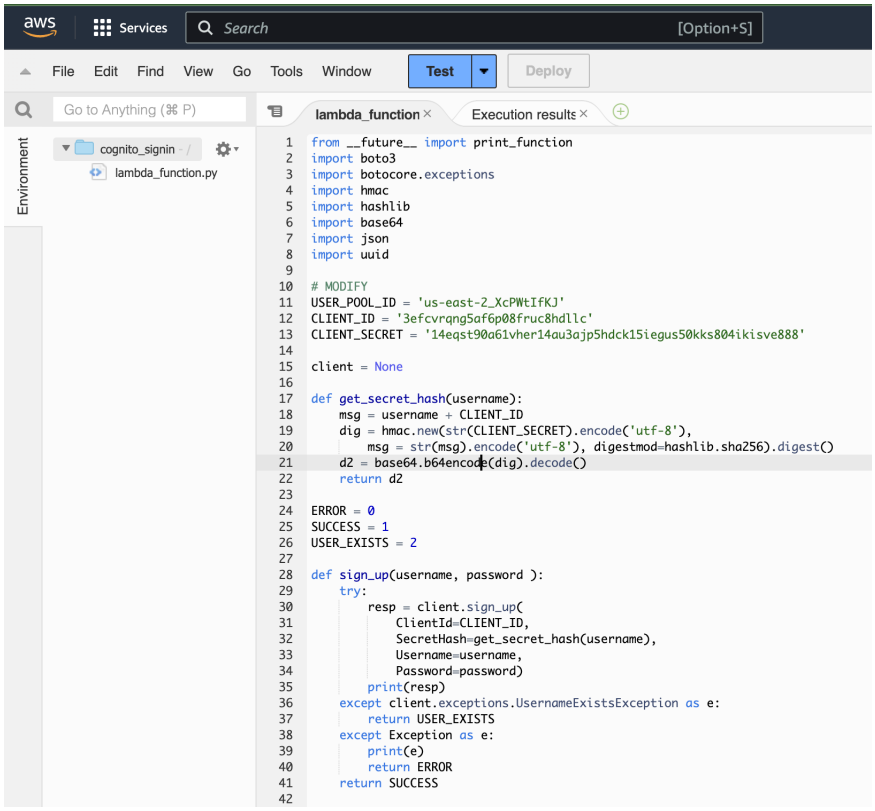


4.2 Lambda Functions

I have three lambda functions to perform all the server side computation so that the app is server-less and stateless on the client side.

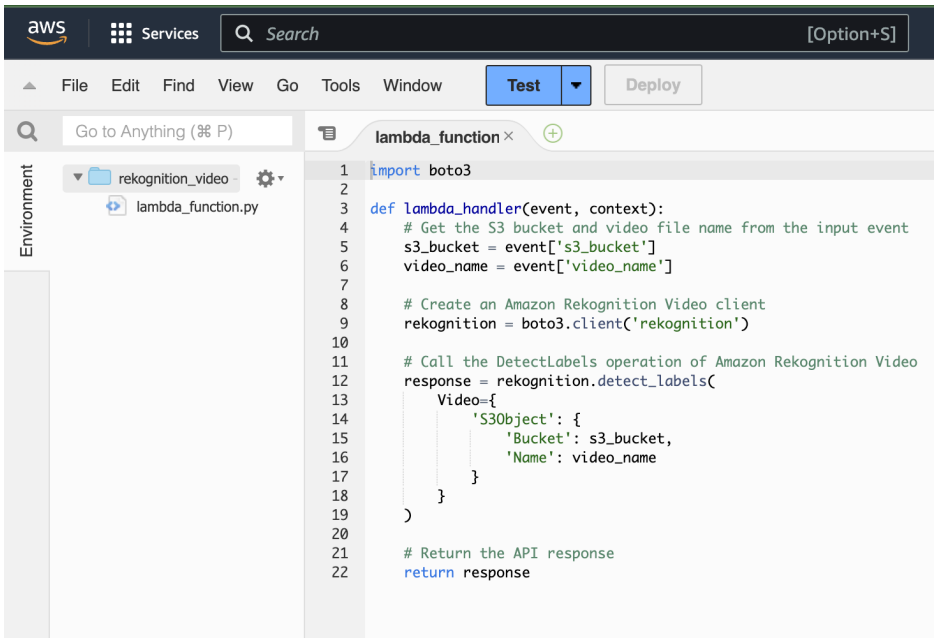
4.2.1.Cognito - Signin/Signup

Below is the lambda function code, which receives the username and password, using clientID and client secret it will authenticate and interact with AWS cognito service to get the details of the user.



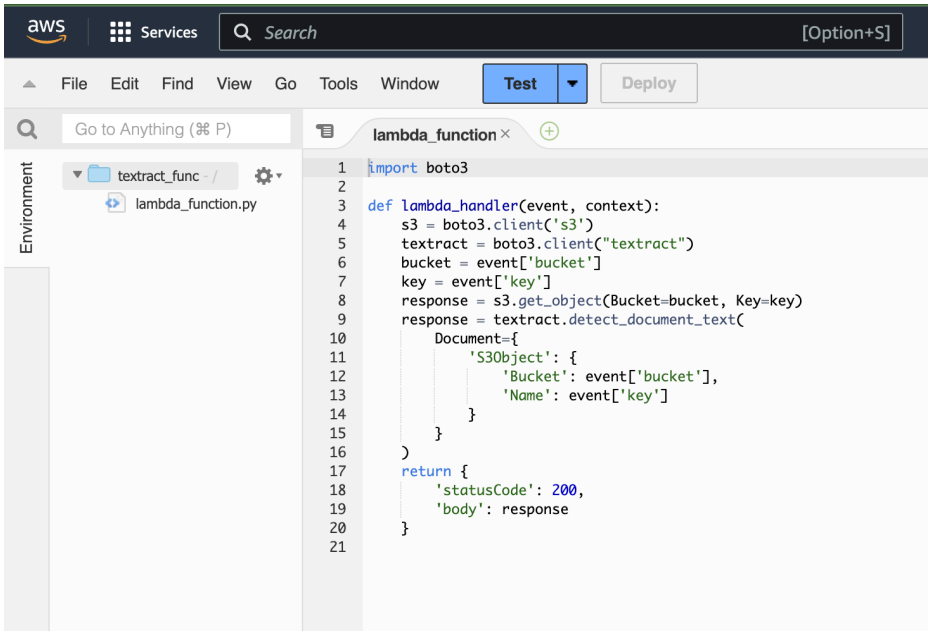
4.2.2. Image Rekognition

Below is the lambda function code, where it receives the bucket name and media name, By using the DetectLabels operation of the Amazon Rekognition service I will be able to extract the metadata of the media



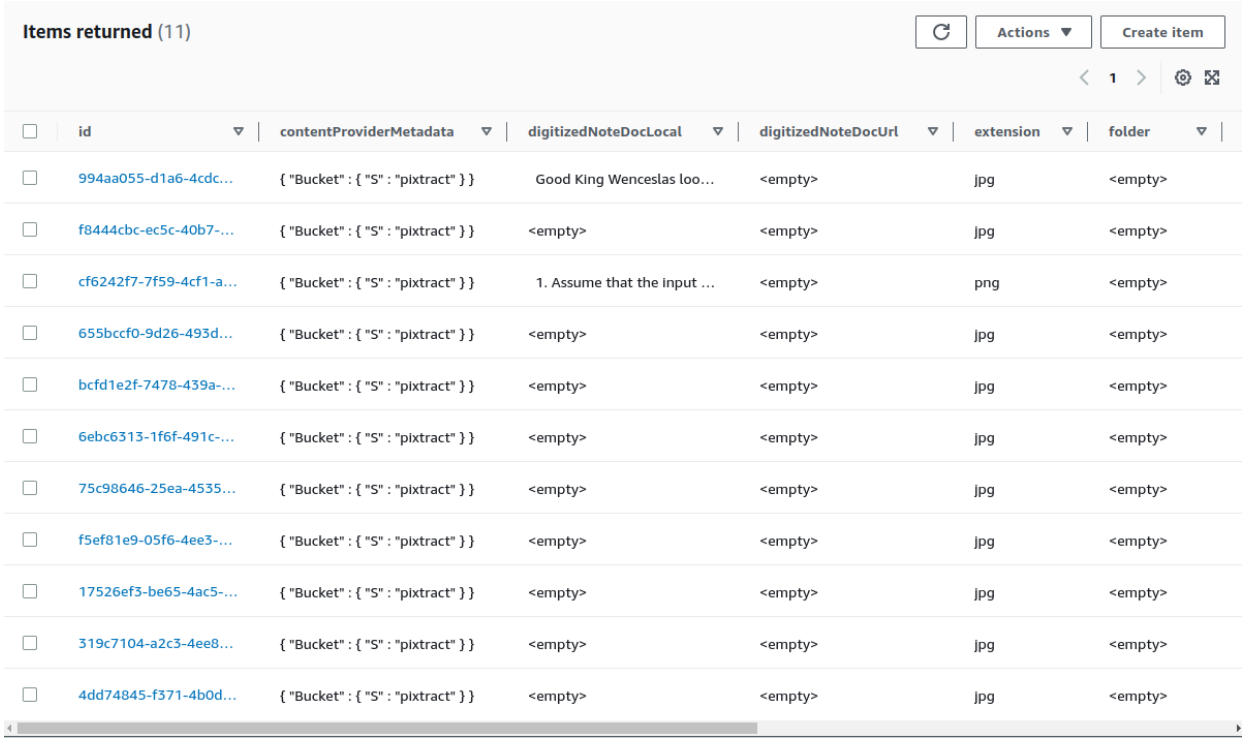
4.2.3. Textract

Below is the lambda function code, where it receives the bucket name and media name, By using the DetectDocumentText operation of the Amazon Textract service I will be able to extract the text of the image/document.



4.3 Database

Below are the details from the DynamoDB of the table called MEDIA_MASTER, which contains all the data of the image or the notes.



Items returned (11)

Actions ▼ | Create item

1

id	contentProviderMetadata	digitizedNoteDocLocal	digitizedNoteDocUrl	extension	folder
994aa055-d1a6-4cdc...	{ "Bucket" : { "S" : "pixtract" } }	Good KIng Wenceslas loo...	<empty>	jpg	<empty>
f8444cbc-ec5c-40b7-...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
cf6242f7-7f59-4cf1-a...	{ "Bucket" : { "S" : "pixtract" } }	1. Assume that the input ...	<empty>	png	<empty>
655bccf0-9d26-493d...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
bcfd1e2f-7478-439a-...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
6ebc6313-1f6f-491c-...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
75c98646-25ea-4535...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
f5ef81e9-05f6-4ee3-...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
17526ef3-be65-4ac5-...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
319c7104-a2c3-4ee8...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>
4dd74845-f371-4b0d...	{ "Bucket" : { "S" : "pixtract" } }	<empty>	<empty>	jpg	<empty>

Once the image is uploaded, all the image metadata that is acquired from Textract and Rekognition are stored in the Dynamo table. Whenever the user performs a search request, I use information stored in the table to filter search results, as lll as the responses from openSearch.

5. CHALLENGES AND FUTURE SCOPE

Despite the challenges I faced during the development of the Pixtract Ib application, I Ire able to successfully integrate a wide range of AWS services and create a fully functional media storage and search application. HoIver, I did face some issues and there is still scope for future improvements.

One of the main challenges I faced was integrating and synchronizing such a large number of services, each with its own unique requirements. This required careful planning and coordination betIen the development team to ensure that all services Ire integrated properly and working together seamlessly.

Another challenge I faced was optimizing the application's performance. Loading a large number of images from S3 on a fresh download proved to be time-consuming, so I had to implement various optimization techniques to improve the application's speed. For example, I converted the image into a base64 encoded dataURI to make the application faster.

In terms of future scope, there are still some improvements that can be made. While I have set up OpenSearch for the application's search functionality, I are currently only doing basic search. There is still a lot of potential for using the full capabilities of OpenSearch, such as complex querying using Lucene syntax, which I can explore in future iterations of the application.

Overall, the development of the Pixtract Ib application was a challenging but rewarding experience, and I are proud to have created a fully functional media storage and search application using AWS services.

6. CONCLUSION AND SOURCE CODE

Overall, this application enabled us to learn about a variety of AWS services, how they are set up, how they work independently as Ill as together. The AWS suite of services is extremely poIrful and enabled us to build a fully functioning Image processing / Photo search application within a month using some advanced Artificial Intelligence features such as Image Recognition, and Optical Character Recognition (OCR) Text- Extraction .

Our Git Repository with the client code as Ill as all the lambda functions and Postman tests is public, and can be found at :

https://github.com/kshanmu1/pixtract

I invite you to clone the repository, and run **npm i** and the **npm run serve** commands in the /src directory to run the application in your browser and play around!