Image Upload & Management API Documentation

# Overview

This API allows users to upload, list, view, and delete images.

The images are stored in AWS S3, and metadata is saved in AWS DynamoDB. The API is protected using AWS API Gateway.

# Base URL

If deployed using AWS API Gateway, the API base URL will be:

https://image-api-gateway-102545.execute-api.us-east-1.amazonaws.com/prod

# Endpoints & Usage

1. **Upload Image**

URL: POST /upload

Description: Upload an image along with metadata. Request Format: multipart/form-data

Parameters:

* file (required) - Image file
* user\_id (string, required) - User ID
* description (string, optional) - Image description

Response:

* 201 Created: { "file\_id": "<generated-file-id>" }
* 400 Bad Request: { "error": "Missing file" }

Example (cURL):

curl -X POST "https://image-api-gateway-102545.execute-api.us-east-1.amazonaws.com/prod

/upload" \

-H "Content-Type: multipart/form-data" \

-F ["file=@test.jpg"](mailto:file%3D@test.jpg) \

-F "user\_id=123" \

-F "description=Sample Image"

# List All Images

URL: GET /images

Description: Retrieve a list of all uploaded images with optional filters.

Query Parameters (optional):

* user\_id - Filter by user ID
* description - Filter by description keyword

Response: [

{"id": "file1", "user\_id": "123", "description": "Test image", "s3\_url": "https://s3-url"},

{"id": "file2", "user\_id": "124", "description": "Another image", "s3\_url": "https://s3-url"}

]

Example (cURL):

curl -X GET "https://image-api-gateway-102545.execute-api.us-east-1.amazonaws.com/prod

/images?user\_id=123"

# View/Download Image

URL: GET /image/<file\_id>

Description: Retrieve an image URL for viewing or downloading. Response:

* + 200 OK: { "s3\_url": "https://s3-url-of-image" }
  + 404 Not Found: { "error": "Image not found" }

Example (cURL):

curl -X GET "https://image-api-gateway-102545.execute-api.us-east-1.amazonaws.com/prod

/image/file1"

# Delete an Image

URL: DELETE /delete/<file\_id>

Description: Delete an image from S3 and its metadata from DynamoDB. Response:

* + 200 OK: { "message": "Image deleted successfully" }
  + 404 Not Found: { "error": "Image not found" } Example (cURL):

curl -X DELETE "https://image-api-gateway-102545.execute-api.us-east-1.amazonaws.com/prod

delete/file1"

# Deployment Steps

1. Deploy the Flask App to AWS Lambda using Zappa or AWS Serverless Framework.
2. Set up API Gateway to route requests to the Lambda function.
3. Secure API Gateway with IAM roles, API keys, or JWT authentication.
4. Test APIs using cURL, Postman, or any API testing tool.

# Development Environment

Use LocalStack to create a local development environment.

Setting up LocalStack:

1. Install LocalStack using pip: pip install localstack
2. Start LocalStack: localstack start
3. Configure AWS CLI to use LocalStack: aws configure set aws\_access\_key\_id test

aws configure set aws\_secret\_access\_key test aws configure set default.region us-east-1

Using LocalStack with S3 and DynamoDB:

1. Create an S3 bucket:

aws [--endpoint-url=http://localhost:4566](http://localhost:4566/) s3 mb s3://my-bucket

1. Create a DynamoDB table:

aws [--endpoint-url=http://localhost:4566](http://localhost:4566/) dynamodb create-table --table-name Images

--attribute-definitions AttributeName=id,AttributeType=S --key-schema AttributeName=id,KeyType=HASH --billing-mode PAY\_PER\_REQUEST

Now, your API can interact with the LocalStack environment as if it were AWS.

# Testing Instructions

Run Locally: python app.py

Run Unit Tests: python test\_app.py

Deploy using Zappa: zappa deploy