Using AWS for Image Recognition

Daniel Garza
Michael Garcia

Professor:
Dr. Gonzalo D. Parra

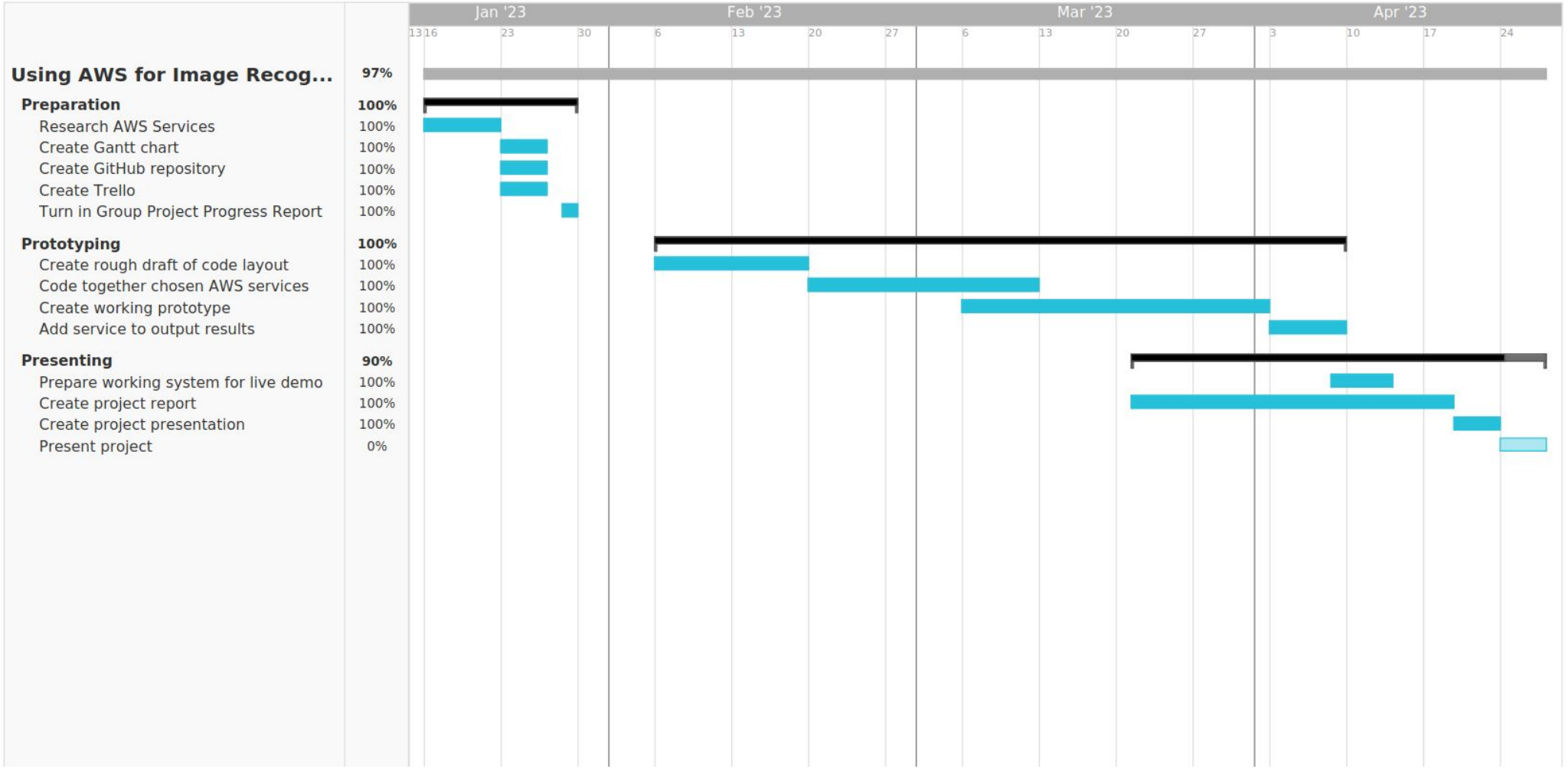
Why this project?

- Image generation/recognition is becoming increasingly popular with AI models such as ChatGPT/DALL-E/HuggingGPT.
- We wanted to attempt to create a pseudo-build of an AI image recognition model solely utilizing AWS.
  - AWS has its own model that can recognize images and characteristics.

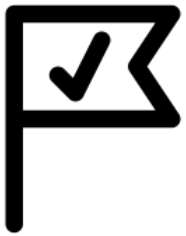
Objective

- Our goal is to create a system that can accurately generate a series of tags about an image that it is given, and send it to an email address.
  - Additionally, we would like to use this project to explore the various services that AWS has to offer.

Timeline


Jan '23
Feb '23
Mar '23
Apr '23
13 16 23 30 6 13 20 27 6 13 20 27 3 10 17 24
Using AWS for Image Recog... 97%
Preparation 100%
Research AWS Services 100%
Create Gantt chart 100%
Create GitHub repository 100%
Create Trello 100%
Turn in Group Project Progress Report 100%
Prototyping 100%
Create rough draft of code layout 100%
Code together chosen AWS services 100%
Create working prototype 100%
Add service to output results 100%
Presenting 90%
Prepare working system for live demo 100%
Create project report 100%
Create project presentation 100%
Present project 0%

Project Milestones

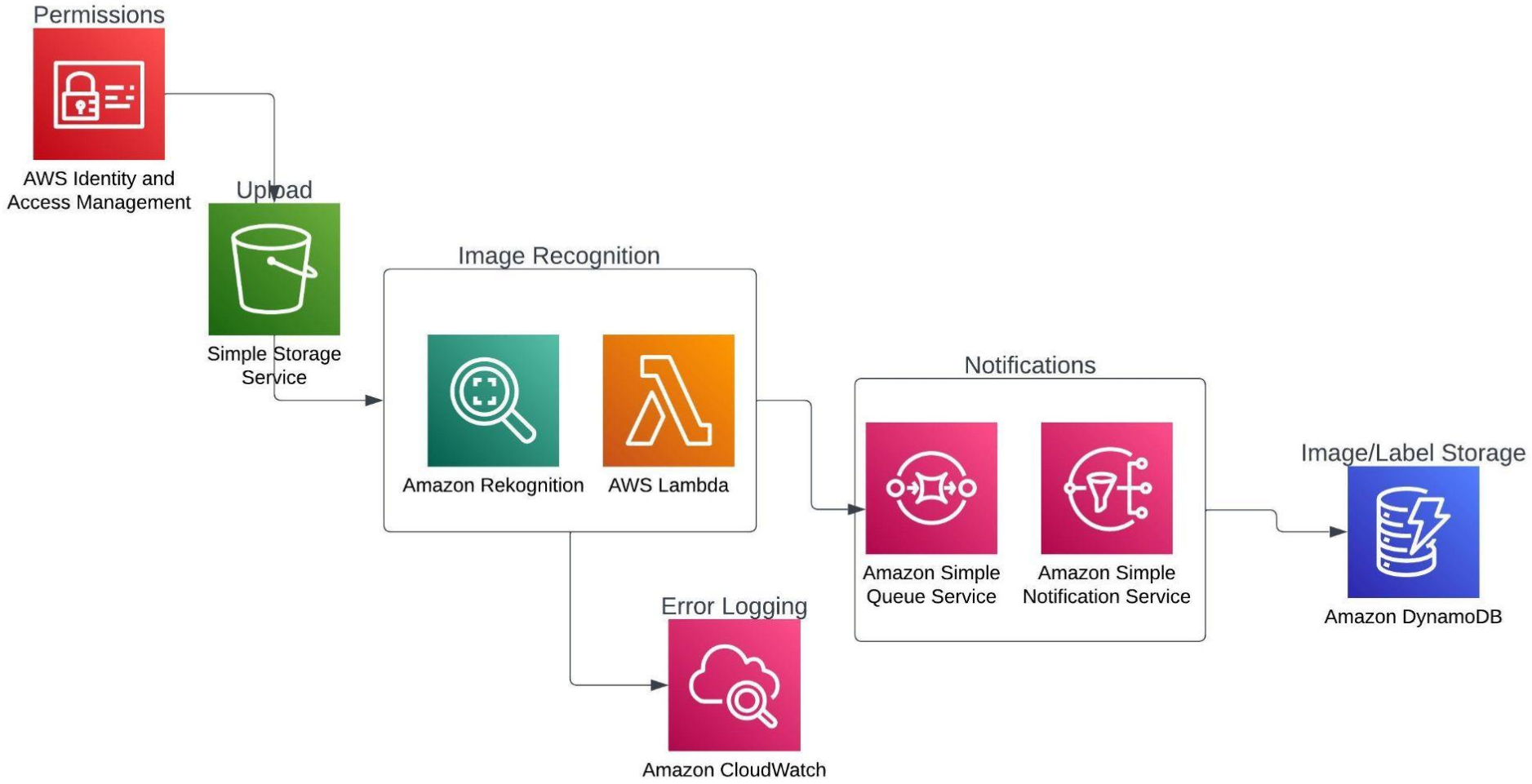


1. Research various AWS services to incorporate into the image recognition system.
2. Link the various services to create a rough draft of the system through coding them together.
3. Create a working prototype that can recognize images and return descriptors.
4. Employ the Amazon SQS service to send the descriptors to a mobile device.

AWS Services

1. IAM - sets permissions for Lambda
2. S3 - stores images to be labeled by Rekognition
3. Lambda - hosts core python code
4. Rekognition - identifies labels to associate to image
5. SNS - sends notification with labels via email
6. SQS - works in conjunction with SNS to deliver notifications
7. DynamoDB - stores labels produced by system

Cloud Architecture



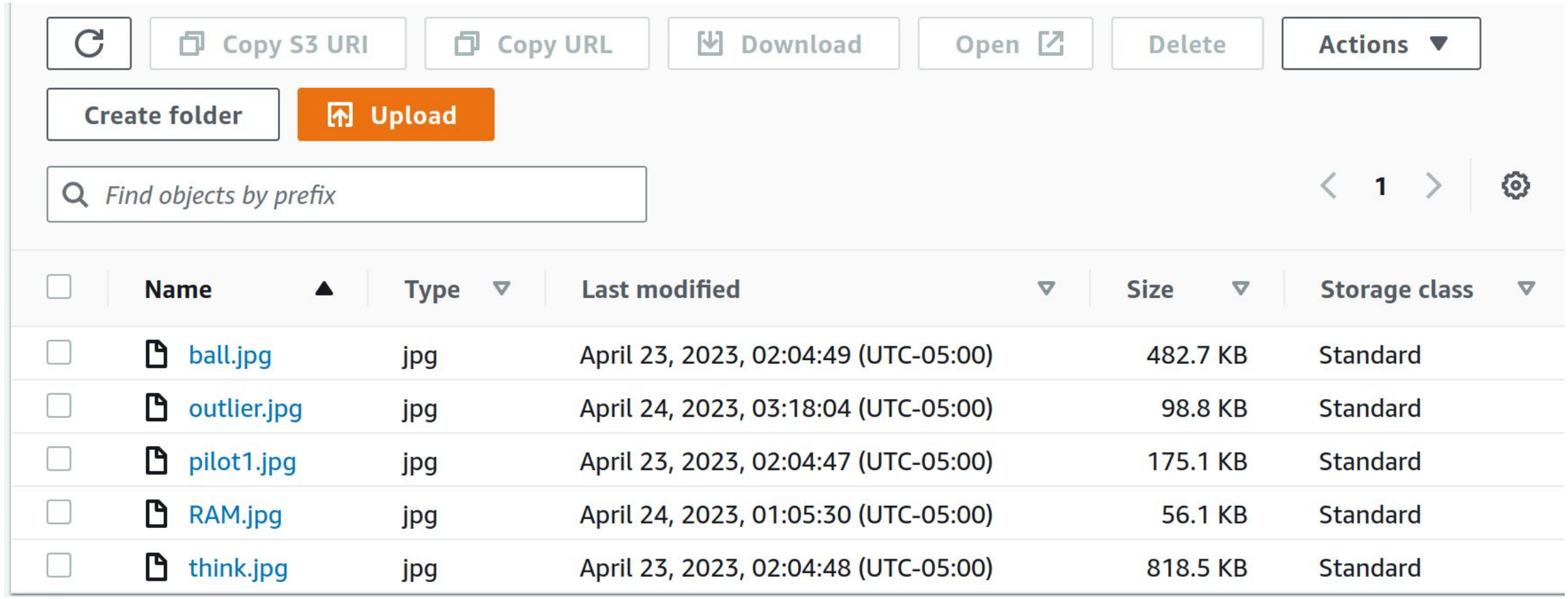
IAM

Policy name	Type	Description
AmazonSQSFullAccess	AWS managed	Provides full access to Amazon SQS via the AWS Ma…
AmazonS3FullAccess	AWS managed	Provides full access to all buckets via the AWS Mana…
CloudWatchFullAccess	AWS managed	Provides full access to CloudWatch.
AmazonDynamoDBFullAccess	AWS managed	Provides full access to Amazon DynamoDB via the A…
AmazonRekognitionFullAccess	AWS managed	Access to all Amazon Rekognition APIs
AWSLambdaBasicExecutionR…	AWS managed	Provides write permissions to CloudWatch Logs.
AmazonSNSFullAccess	AWS managed	Provides full access to Amazon SNS via the AWS Ma…

IAM (Updated for Security)

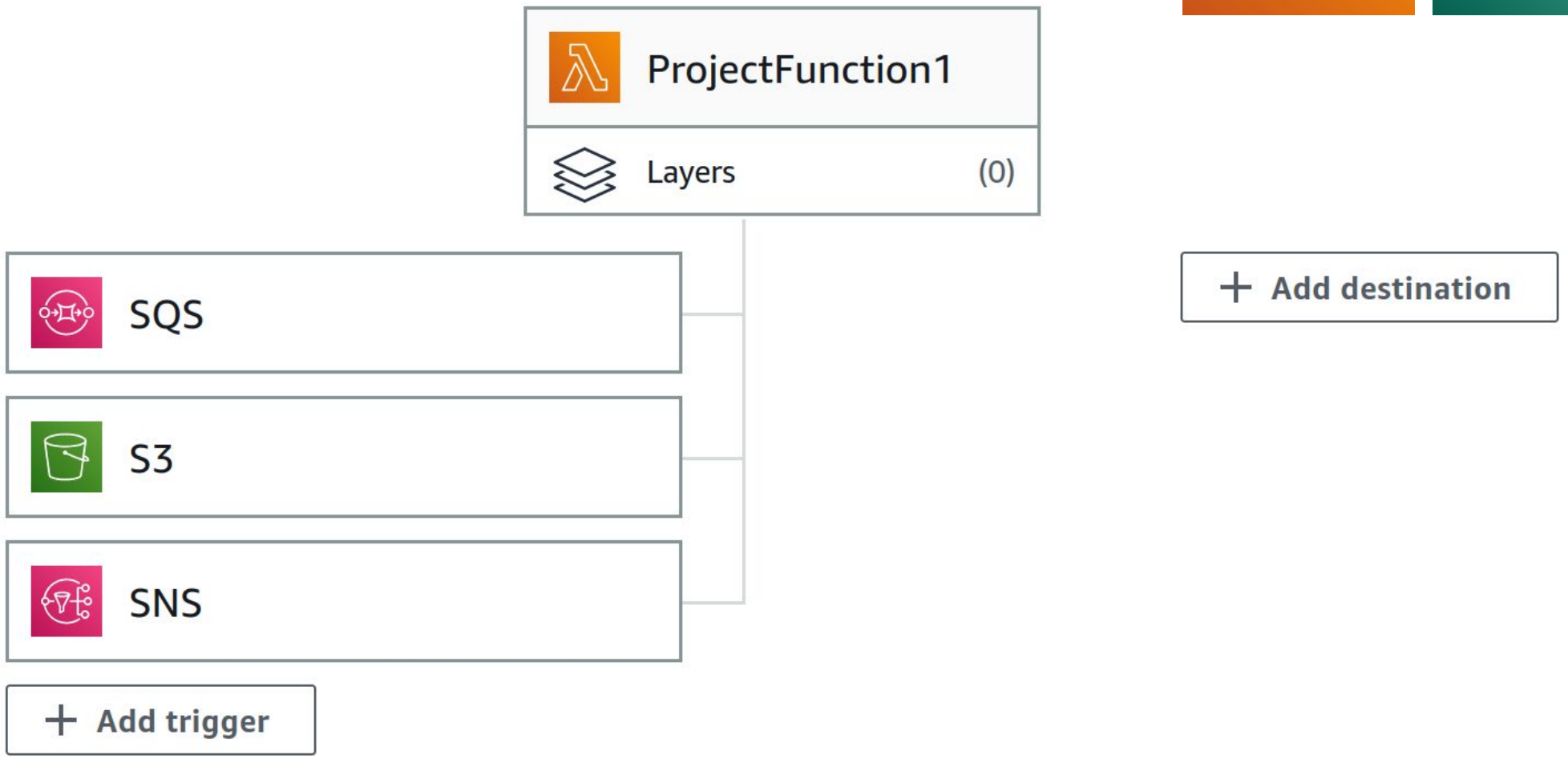
Policy name	Type	Description
AmazonRekognitionReadOnlyAccess	AWS managed	Access to all Read rekognition
AmazonSQSReadOnlyAccess	AWS managed	Provides read only access to A
AmazonS3ReadOnlyAccess	AWS managed	Provides read only access to a
AWSLambdaSQSQueueExecutionRole	AWS managed	Provides receive message, del
AWSLambdaBasicExecutionRole	AWS managed	Provides write permissions to C
DynamoProjectPolicy	Customer inline	
SNSProjectPolicy	Customer inline	

S3



Name	Type	Last modified	Size	Storage class
ball.jpg	jpg	April 23, 2023, 02:04:49 (UTC-05:00)	482.7 KB	Standard
outlier.jpg	jpg	April 24, 2023, 03:18:04 (UTC-05:00)	98.8 KB	Standard
pilot1.jpg	jpg	April 23, 2023, 02:04:47 (UTC-05:00)	175.1 KB	Standard
RAM.jpg	jpg	April 24, 2023, 01:05:30 (UTC-05:00)	56.1 KB	Standard
think.jpg	jpg	April 23, 2023, 02:04:48 (UTC-05:00)	818.5 KB	Standard

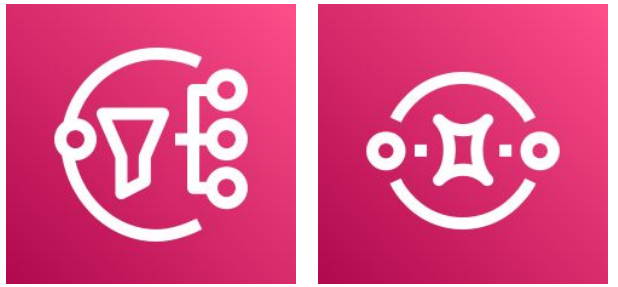
Lambda & Rekognition



Lambda & Rekognition

```python
import boto3
import uuid

s3_client = boto3.client('s3')
rekognition_client = boto3.client('rekognition')
sqs_client = boto3.client('sqs')
sns_client = boto3.client('sns')
dynamodb_client = boto3.client('dynamodb')

def lambda_handler(event, context):
    # Retrieve S3 bucket and object information from the event
    s3_bucket = event['Records'][0]['s3']['bucket']['name']
    s3_object_key = event['Records'][0]['s3']['object']['key']

    try:
        image = get_s3_image(s3_bucket, s3_object_key)

        labels = detect_labels(image)

        process_labels(labels, s3_object_key)

        message = '\n'
        for label in labels:
            message += f'Label: {label["Name"]}, Confidence: {label["Confidence"]}%\n'
```

SNS & SQS



projectSNS

Edit | Delete | Publish message

Details

Name
projectSNS

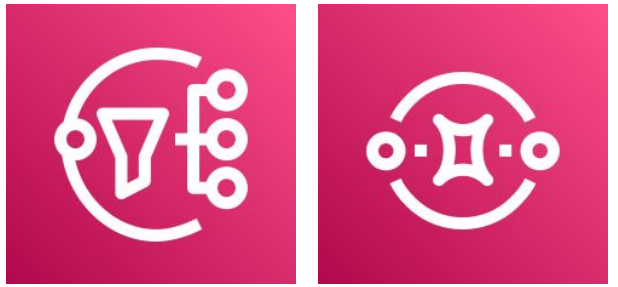
ARN

Type
Standard

Display name
Image Recognition Status

Topic owner

SNS & SQS



Subscriptions (4)

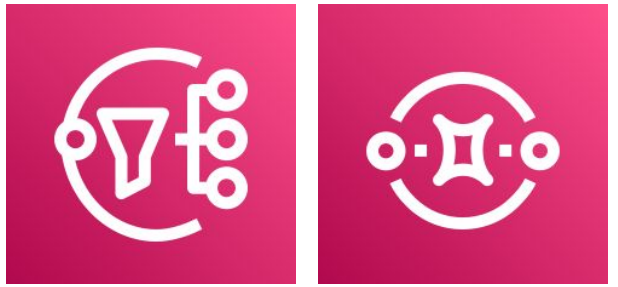
Edit | Delete | Request confirmation | Confirm subscription | **Create subscription**

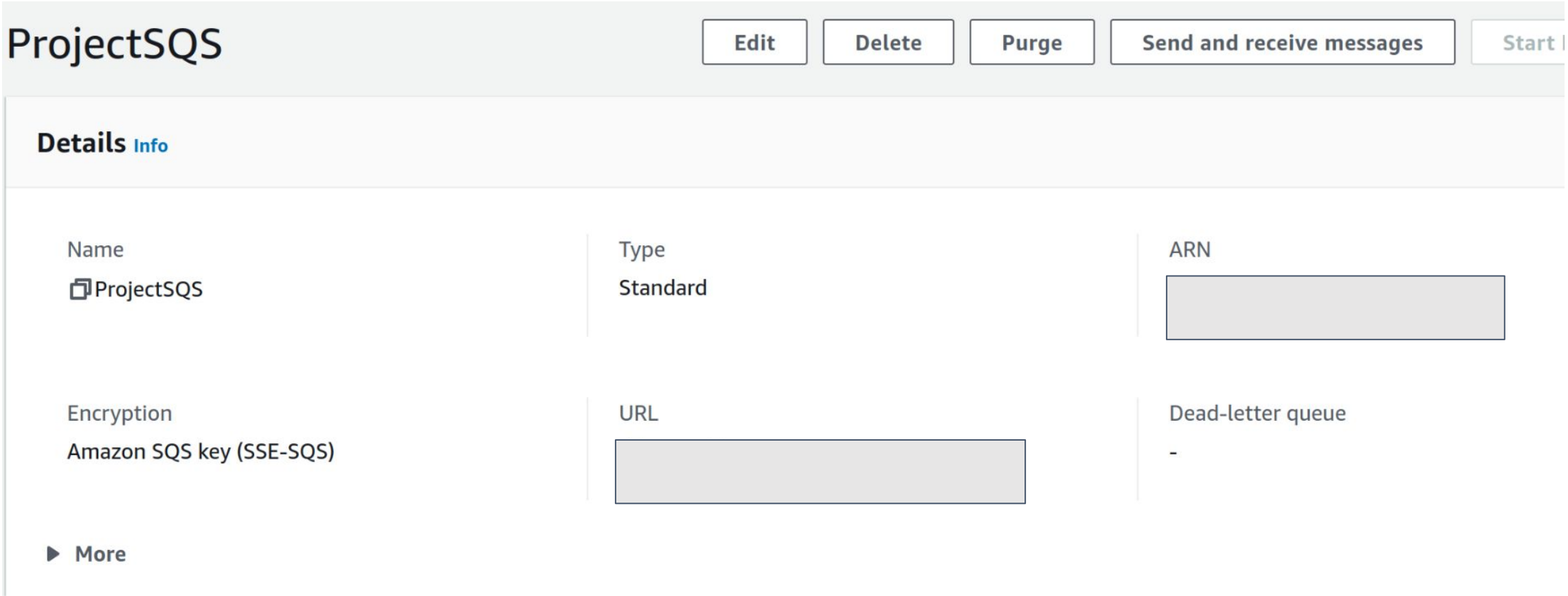
Search

1

ID	Endpoint	Status	Protocol	Topic
483f8313-9fb2-...		Confirmed	EMAIL	Default_CloudWa...
65cd3120-1fde-...		Confirmed	EMAIL	projectSNS
bf2e0c6e-f87e-4...		Confirmed	SQS	projectSNS
d937cc7a-a7de-...		Confirmed	LAMBDA	projectSNS

SNS & SQS




ProjectSQS
Edit
Delete
Purge
Send and receive messages
Start
Details Info
Name
ProjectSQS
Type
Standard
ARN
Encryption
Amazon SQS key (SSE-SQS)
URL
Dead-letter queue
-
More

DynamoDB

image_id	confidence	label	object_key
d64cb3fb-9af3-47ab-...	99.9999847...	Truck	RAM.jpg
62ab538e-e4fb-4383...	99.9999847...	Pickup Truck	RAM.jpg
7439524a-5d35-408c...	95.6109008...	American F...	ball.jpg
4848d1ae-0c84-4b1a...	95.6109008...	Sport	ball.jpg
a2afaa7a-40ad-4cea-...	95.6109008...	Football	ball.jpg
48eae29a-bf83-46f1-...	99.9999847...	Vehicle	RAM.jpg

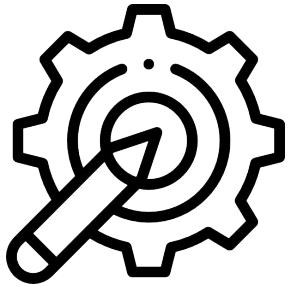
Challenges & Limitations

- Learning to integrate multiple AWS services into python code
- Unable to use SNS for text messaging due to law requiring usage of toll-free number
- Code deployment required before testing, resulting in constant rollbacks
- DynamoDB caused many issues with integrating into our system
  - Lack of experience with NoSQL

Conclusion

- While our image recognition system can only provide basic labels from the images it is given, it could be improved to recognize more specific details with AutoML, such as the make and model of a car.
- It was somewhat difficult to work within the bounds of the AWS Free Tier, as a lot of the services required payment for resource use/extra features.

Resources



Amazon DynamoDB. (n.d.). Retrieved March 28, 2023, from https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStartedDynamoDB.html

Marquez, E. (2021, November 10). *How to create an AWS lambda function*. Cloud Computing. Retrieved February 6, 2023, from https://www.techtarget.com/searchcloudcomputing/tip/How-to-create-an-AWS-Lambda-function

Musgrave, D. (2022). *Lambda*. Amazon. Retrieved February 8, 2023, from https://docs.aws.amazon.com/lambda/latest/dg/API_Reference.html

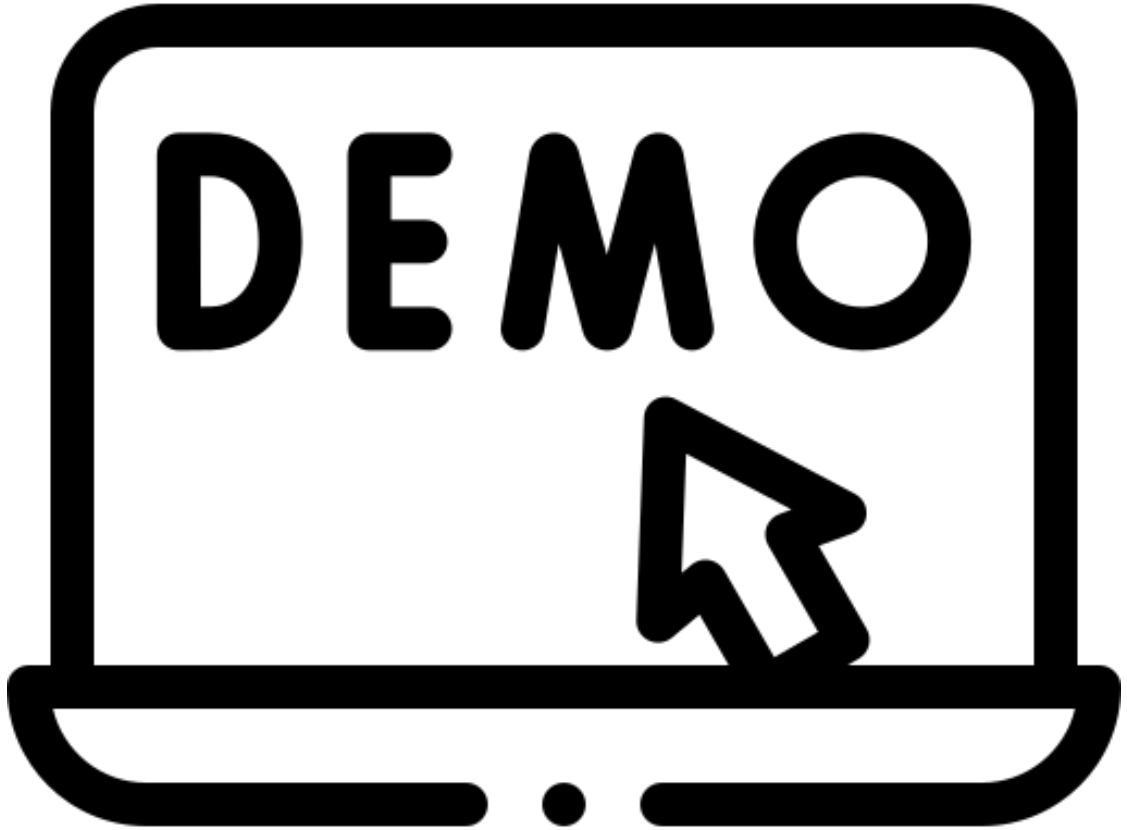
North, F. (1998). *SNS*. Amazon. Retrieved April 4, 2023, from https://aws.amazon.com/getting-started/hands-on/send-fanout-event-notifications/

North, F. (1998). *SQS*. Amazon. Retrieved April 9, 2023, from https://aws.amazon.com/getting-started/hands-on/send-messages-distributed-applications/

Prerequisite: Setting up Amazon S3 - Amazon Simple Storage Service. (n.d.). Retrieved March 2, 2023, from https://docs.aws.amazon.com/AmazonS3/latest/userguide/setting-up-s3.html

Schütz Julia. (2011). *Amazon Rekognition: Detecting Labels*. Amazon. Retrieved March 17, 2023, from https://docs.aws.amazon.com/rekognition/latest/dg/labels.html?pg=ln&sec=ft

Whitehouse-Grant-Christ, I. H. V. (2011). *IAM*. Amazon. Retrieved March 8, 2023, from https://docs.aws.amazon.com/IAM/latest/UserGuide/access.html


DEMO