**Serverless Image Recognition using AWS Rekognition**

*A Course Project Report Submitted in partial fulfillment of the course requirements for the award of grades in the subject of*

**CLOUD BASED AIML SPECIALITY**

**(22SDCS07A)**

by

**Hasritha Reddy Cheruku**

**2210030012**

*Under the esteemed guidance of*

**Ms. P. Sree Lakshmi**

Assistant Professor,

Department of Computer Science and Engineering



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**K L Deemed to be UNIVERSITY**

*Aziznagar, Moinabad, Hyderabad,*

*Telangana, Pincode: 500075*

April 2025

**K L Deemed to be UNIVERSITY**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

*Certificate*

This is Certified that the project entitled **“Serversless image recognition using AWS rekognition”** which is a experimental &/ theoretical work carried out by Hasritha Reddy Cheruku 2210030012, in partial fulfillment of the course requirements for the award of grades in the subject of  **CLOUD BASED AIML SPECIALITY**, during the year **2024-2025**. The project has been approved as it satisfies the academic requirements.

**Ms.P.Sree Lakshmi Dr. Arpita Gupta**

**Course Coordinator Head of the Department**

**Ms. P. Sree Lakshmi**

**Course Instructor**

**CONTENTS**

Page No.

1. Introduction 4

2. AWS Services Used as part of the project 5

3. Steps involved in solving project problem statement 7

4. Stepwise Screenshots with brief description 9

5. Learning Outcomes 12

6. Conclusion 13

7.Future Scope 14

7. References 15

1. **INTRODUCTION**

*Mini Project Title*

In the digital age, images and videos are everywhere—from social media and websites to surveillance and security systems. With this massive increase in visual data, there is a growing need for automated systems that can quickly and accurately analyze this content. Manual analysis is slow and impractical, especially at scale. This is where cloud-based image analysis services like **Amazon Rekognition** come into play.

**Amazon Rekognition** is a fully managed service from AWS that provides deep learning–based image and video analysis. It can recognize thousands of objects such as people, vehicles, animals, and scenes. It also offers advanced features like facial analysis, facial comparison, text detection in images, and inappropriate content detection. This makes it useful for a wide range of applications including content moderation, face-based user verification, and even public safety solutions.

To handle large-scale tasks like this without managing servers, cloud platforms offer **serverless computing**. In a serverless setup, code is executed in response to specific events and automatically scales as needed. **AWS Lambda** is one such service that runs backend code in response to events like HTTP requests or file uploads. It doesn't require provisioning or managing servers, which helps reduce operational complexity and cost.

In this project, a **serverless image analysis system** is implemented using Amazon Rekognition, **Amazon S3**, **AWS Lambda**, **IAM roles**, and **Amazon CloudWatch**. The flow begins with uploading an image to an S3 bucket. This action triggers a Lambda function, which then sends the image to Amazon Rekognition for analysis. The results, such as detected labels or facial features, are logged in Amazon CloudWatch.

An **IAM role** is configured with permissions to allow the Lambda function to access S3, Rekognition, and CloudWatch. This ensures secure communication between services. Since the entire solution runs on serverless architecture, it can handle multiple requests automatically without any performance issues or the need for infrastructure maintenance.

This project demonstrates a practical and efficient way to automate image processing using modern cloud tools. It highlights how serverless technology and AI services can be combined to create smart, scalable, and cost-effective applications—making it ideal for both academic learning and real-world use cases.

1. **AWS Services used as part of the project**

This project leverages several powerful AWS services that work together seamlessly to provide a serverless image analysis system. These services handle everything from storing images to processing and analyzing them. Here’s a closer look at each AWS service used in the project:

### ****Amazon S3 (Simple Storage Service)****

**Amazon S3** is used to store images in this project. S3 is a scalable and highly durable object storage service that allows users to store any amount of data. In this project, when an image is uploaded to the designated S3 bucket, it automatically triggers a series of actions in the system. S3 is the starting point of the workflow, where images are stored and event notifications are sent to initiate processing by the Lambda function.

S3 provides several benefits such as built-in redundancy, ease of integration with other AWS services, and minimal cost, making it a perfect choice for storing large volumes of data like images.

### ****AWS Lambda****

**AWS Lambda** is at the heart of the serverless architecture in this project. Lambda runs code in response to events, such as the upload of an image to an S3 bucket. The Lambda function is responsible for processing the image and calling the **Amazon Rekognition** service to analyze the image’s content. With Lambda, there is no need to provision or manage servers, as the service automatically scales based on the workload.

Lambda allows the code to run only when triggered, which is cost-effective since users only pay for the time the function is running, rather than keeping a server running continuously.

### ****Amazon Rekognition****

**Amazon Rekognition** is the main service for image and video analysis in this project. It leverages deep learning models to detect objects, scenes, text, faces, and other elements within an image. When an image is uploaded to the S3 bucket, the Lambda function calls Rekognition to perform analysis on the image. The Rekognition API returns detailed results, such as labels for objects found in the image or facial analysis results, which are then logged in **Amazon CloudWatch**.

Rekognition provides a fully managed, highly accurate service that requires no machine learning expertise to use. It is ideal for quickly adding advanced image analysis features to applications.

### ****Amazon CloudWatch****

**Amazon CloudWatch** is used for monitoring and logging Lambda execution. Every time the Lambda function processes an image, the results (such as detected labels or facial analysis data) are logged in CloudWatch. This service helps track function performance, debug issues, and maintain system health by providing logs and metrics in real-time.

CloudWatch allows developers to monitor the entire flow of the application, providing insight into Lambda function executions and Rekognition API responses. It also facilitates easy troubleshooting by displaying detailed logs of what happens during each Lambda invocation.

### ****AWS IAM (Identity and Access Management)****

**AWS IAM** plays a crucial role in securing the communication between services. In this project, an IAM role is created and assigned to the Lambda function. This role grants the necessary permissions for Lambda to interact with other AWS services, such as **S3**, **Rekognition**, and **CloudWatch**. IAM roles ensure that each service only has access to the resources it needs, improving security and reducing the risk of unauthorized access.

IAM helps define who can access what resources and under what conditions, providing fine-grained control over permissions.

1. **Steps involved in solving project problem statement**

To solve the project’s problem statement, which is to automate the image analysis process using AWS services, several steps are involved. These steps include setting up the necessary services, creating an image processing pipeline, and configuring security. The following is an outline of the entire workflow:

Step 1: Set Up Amazon S3 Bucket for Image Storage

The first step is to create an Amazon S3 bucket, which will serve as the storage location for the images that need to be analyzed. Amazon S3 is highly durable and scalable, making it an ideal choice for storing images.

Create an S3 bucket via the AWS Management Console.

Set the appropriate permissions on the bucket to control who can upload and access files.

Enable event notifications to trigger actions when new files are uploaded to the bucket.

Step 2: Create AWS Lambda Function

Once the S3 bucket is set up, the next step is to create an AWS Lambda function that will automatically be triggered whenever a new image is uploaded to the S3 bucket.

Define the function that will process images. This involves writing the code that will be executed when an event (e.g., image upload) occurs.

Configure the Lambda function to take the uploaded image, call Amazon Rekognition to analyze it, and log the results to Amazon CloudWatch.

Step 3: Configure Event Trigger on S3 Bucket

To ensure the Lambda function runs automatically when an image is uploaded, an event notification must be set up on the S3 bucket.

In the S3 bucket settings, configure an event notification to trigger the Lambda function upon the ObjectCreated event.

This ensures that whenever a new image is uploaded to the S3 bucket, it will automatically invoke the Lambda function for processing.

Step 4: Set Up Permissions Using AWS IAM

AWS Identity and Access Management (IAM) is crucial to secure the communication between AWS services. An IAM role is needed to allow Lambda to interact with other services like S3, Rekognition, and CloudWatch.

Create an IAM role with the appropriate permissions for accessing S3, Rekognition, and CloudWatch.

Attach this role to the Lambda function, ensuring it has the necessary access rights to process images and log results.

Step 5: Integrate Amazon Rekognition for Image Analysis

The next step is to integrate Amazon Rekognition, which will analyze the images uploaded to S3.

Inside the Lambda function code, use the Rekognition API to analyze the uploaded image.

The analysis can include detecting labels (objects and scenes), facial analysis, or even text in the image, depending on the project’s requirements.

Rekognition will return the analysis results as a structured response, which will then be processed by the Lambda function.

Step 6: Log Results in Amazon CloudWatch

To track the results and performance of the Lambda function, Amazon CloudWatch is used for logging.

In the Lambda function code, ensure that the Rekognition results (e.g., detected labels, objects, faces) are logged to CloudWatch.

CloudWatch allows for monitoring the Lambda function’s execution and helps identify any issues or errors in the image processing pipeline.

1. **Stepwise Screenshots with brief description**

**Step 1: Creating A IAM Role**

First create a IAM Role with AmazonRekognitionFullAccess, AmazonS3FullAccess and CloudWatchFullAccess policies.

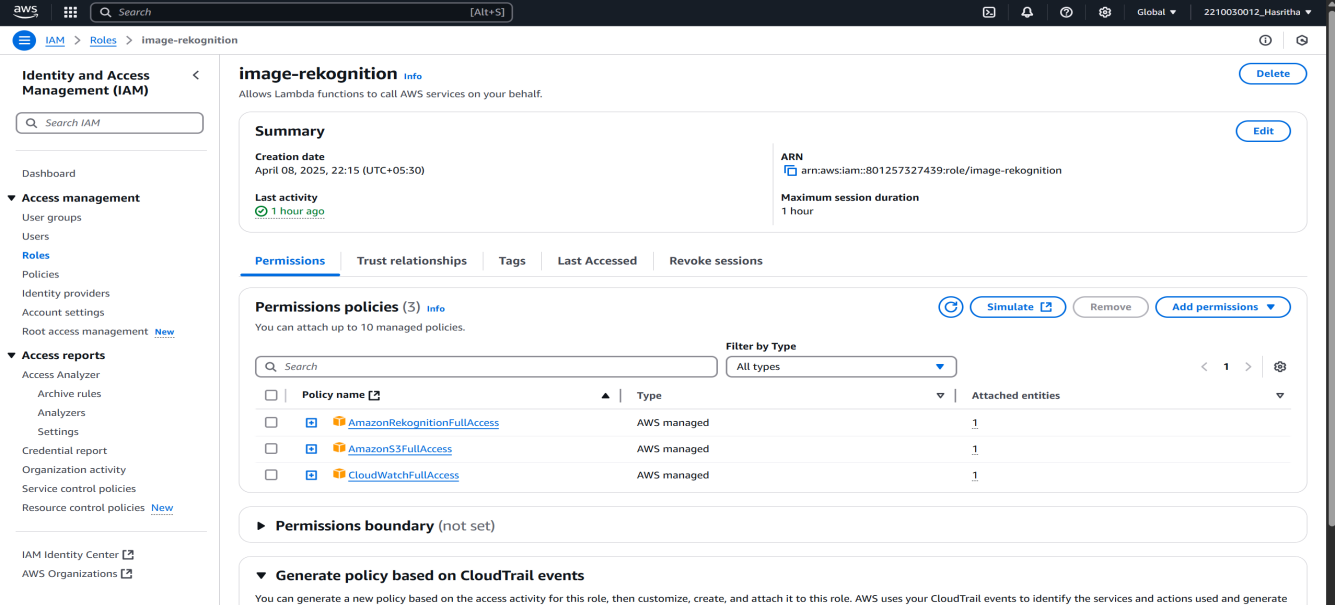


Fig 4.1: Creating A IAM Role

**Step 2: Create a S3 Bucket and upload objects**

Then Navigate to amazon S3 console and create a S3 bucket and upload any jpeg image for further analysis.

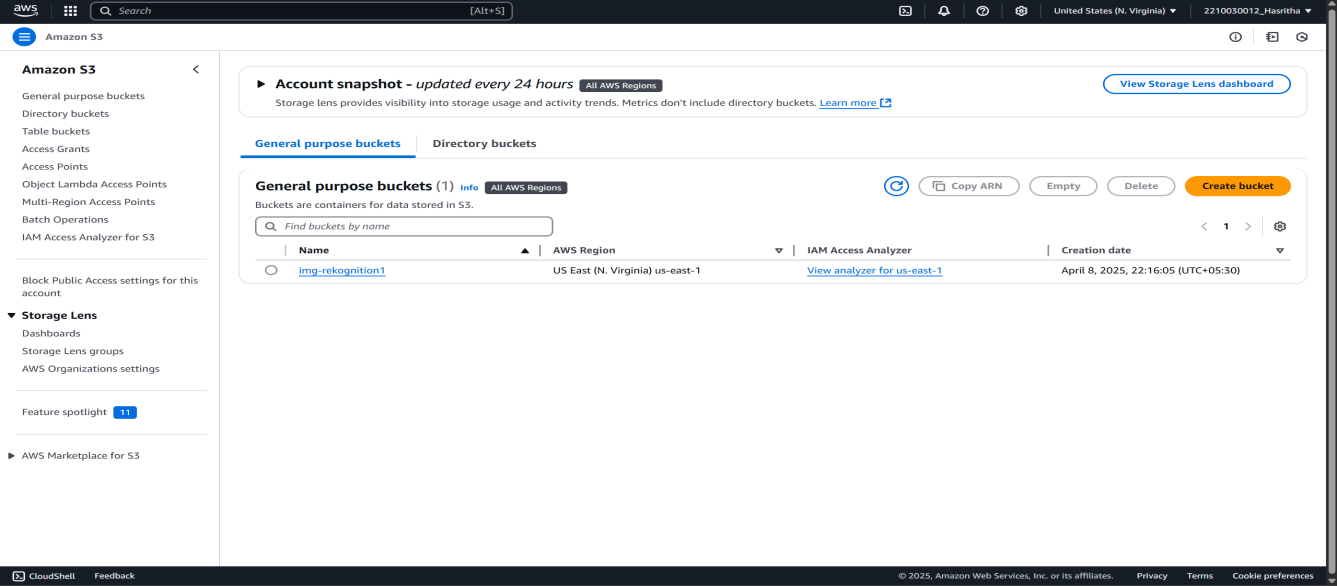


Fig 4.2: Create a S3 Bucket and upload objects

**Step 3:Creating a lambda function**

After creating the S3 bucket and uploading the image, create a lambda function and add the role created earlier and add the aws image rekognition template to the function and create it.

After the creation of the function at code section we can see the code which is displayed for the testing of image analysis.

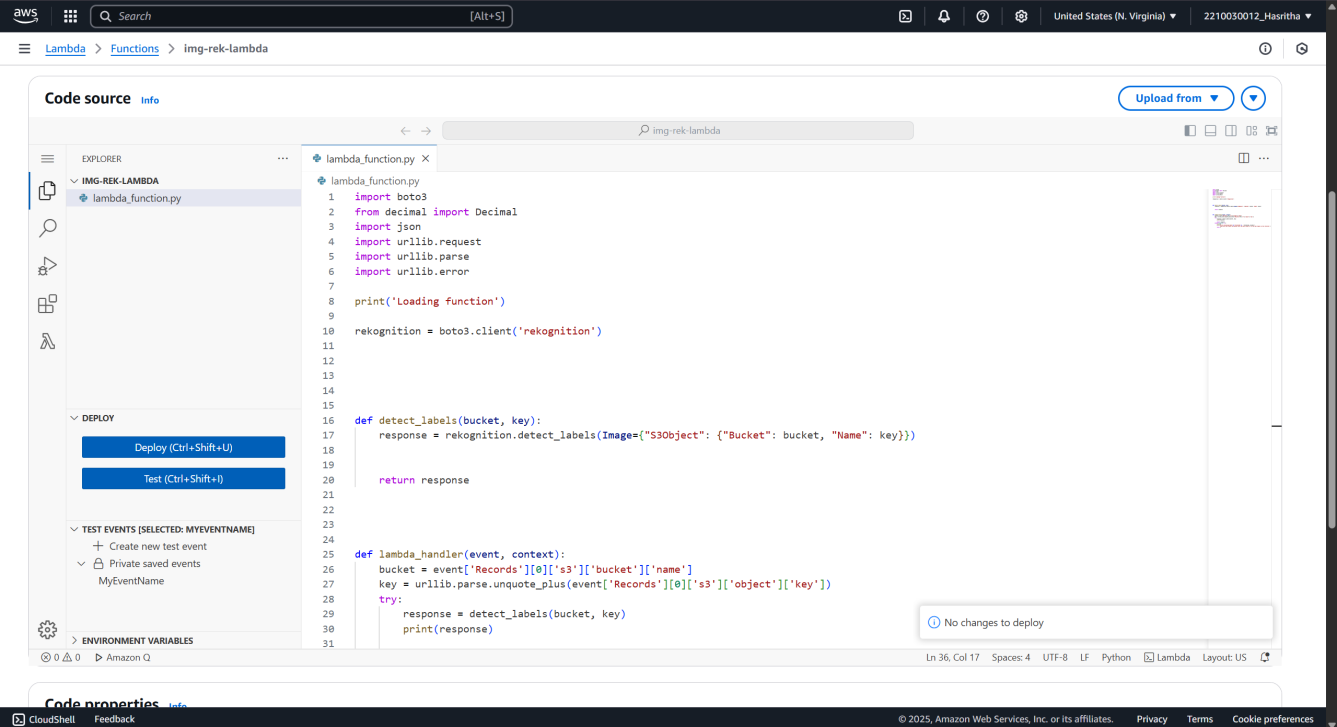


Fig 4.3: Creating a lambda function

**Step 4: Montioring the analysis using cloudwatch console**

After creating the lambda function and saving it. It is time to view the analysis of the image created. On upside of the code editor, go to the monitor section. There, it has multiple monitoring options, choose the cloudwatch console and we can find multiple logs in the log stream choose the recent log stream and view the output of the analysis.

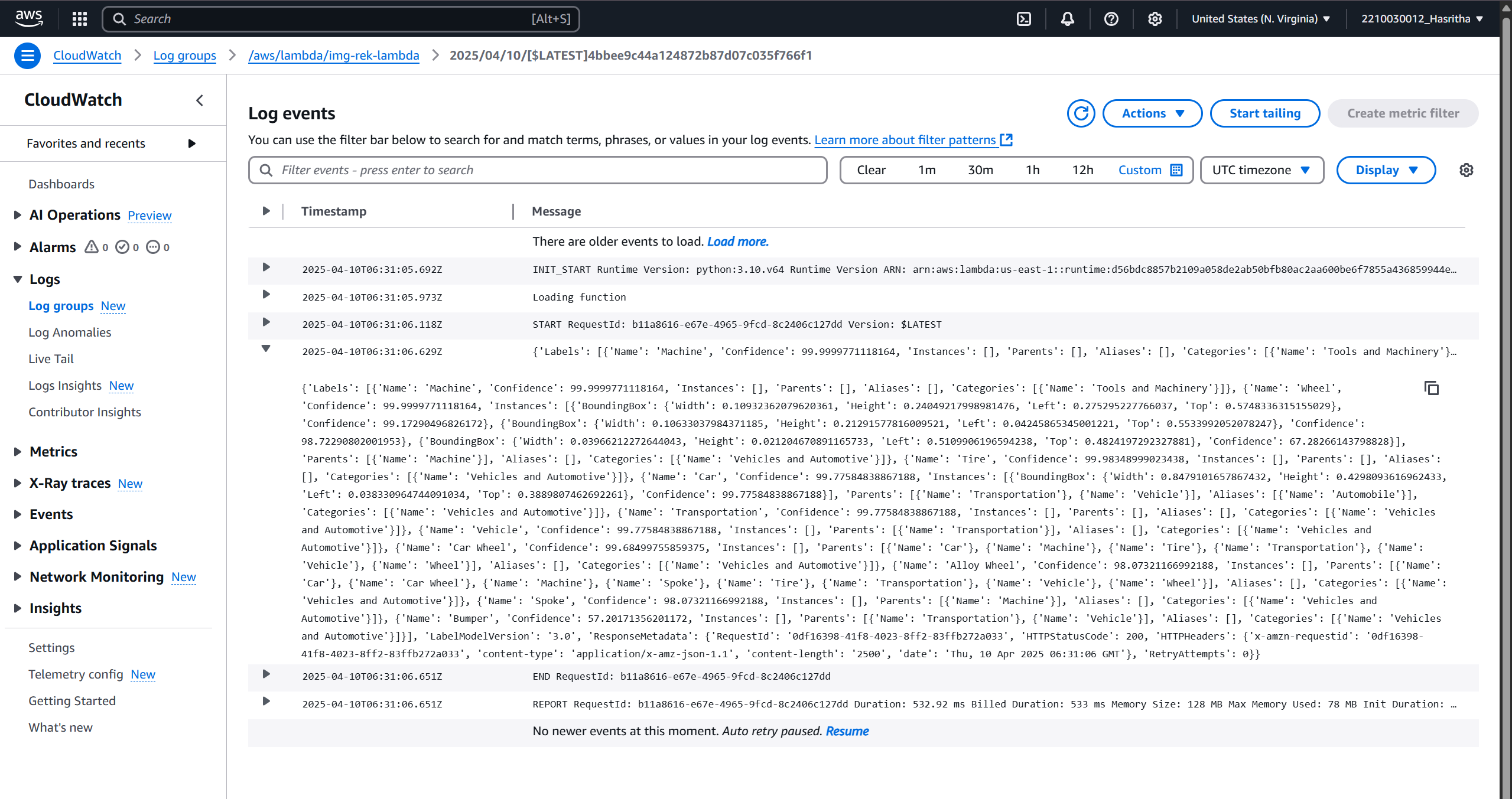


Fig 4.4: Montioring the analysis using cloudwatch console

**5. Learning Outcomes**

This project offered significant insights into cloud computing and serverless architecture by utilizing various AWS services. A major takeaway was understanding how serverless computing works, particularly through AWS Lambda. It became clear how Lambda functions can be triggered automatically without the need to manage any servers, making application development more efficient and scalable.

Additionally, working with AWS services such as Amazon S3, Rekognition, IAM, and CloudWatch provided hands-on experience in setting up a complete cloud-based workflow. The use of Amazon Rekognition introduced the process of image analysis using pre-trained machine learning models accessible via API. This exposure is valuable for developing applications involving facial recognition, object detection, or text extraction from images.

The implementation of CloudWatch for logging and monitoring helped reinforce the importance of observability in cloud applications. By tracking logs and metrics in real-time, it becomes easier to debug and optimize performance. Furthermore, configuring IAM roles and permissions ensured that each AWS service had only the necessary access, emphasizing best practices in cloud security.

Finally, the project highlighted the cost-effective nature of serverless services. Since billing is based only on usage, this architecture is suitable for both small-scale and large-scale applications. The flexibility and automation of AWS services allowed for the development of a fully functional image analysis system without requiring extensive infrastructure or manual maintenance.

**6. Conclusion**

This project successfully demonstrated the power and practicality of serverless architecture by integrating AWS services such as Amazon S3 for image storage, AWS Lambda for event-driven execution, Amazon Rekognition for image analysis, and Amazon CloudWatch for monitoring. By eliminating the need for server management, it significantly reduced operational complexity and overhead while enabling real-time image processing with automated result logging. The resulting system is well-suited for applications like security monitoring, content tagging, or automated image moderation. Overall, it exemplifies how cloud-based serverless solutions can effectively address real-world challenges with intelligent, scalable, and cost-efficient automation.

1. **Future Scope**

The project lays a strong foundation for several future enhancements and real-world applications. One potential advancement is integrating a real-time alert system using AWS SNS (Simple Notification Service) or AWS Step Functions for more complex workflows. The image recognition capabilities can be extended to include facial recognition, label detection, or unsafe content moderation at a larger scale, which is particularly useful in surveillance, social media platforms, or enterprise content filtering. Additionally, incorporating Amazon DynamoDB for structured result storage could support more advanced analytics and querying capabilities. Enhancing the system with API Gateway would allow external applications or mobile clients to interact with the recognition pipeline. Furthermore, the solution can be scaled to handle video input using Amazon Kinesis Video Streams, enabling real-time video analysis in a fully serverless environment. These improvements will make the system more versatile, intelligent, and production-ready for enterprise-grade deployments.

**8. References**

1. What is Amazon Rekognition?. [https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html](https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html" \t "_new)

2. AWS Lambda Developer Guide. [https://docs.aws.amazon.com/lambda/latest/dg/welcome.html](https://docs.aws.amazon.com/lambda/latest/dg/welcome.html" \t "_new)

3. Amazon S3 Documentation. [https://docs.aws.amazon.com/s3/index.html](https://docs.aws.amazon.com/s3/index.html" \t "_new)

4. Amazon CloudWatch Documentation. [https://docs.aws.amazon.com/cloudwatch/](https://docs.aws.amazon.com/cloudwatch/" \t "_new)

5. IAM Roles and Policies. [https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html" \t "_new)