

Serverless Image / Video Processing using AWS

1. Project Overview

Project Title: Serverless Image & Video Processing Pipeline

Architecture Type: Event-Driven Serverless Architecture

Goal

Automatically process images or videos uploaded to Amazon S3. Processing includes:

- Image resizing (Lambda)
- Label detection (Amazon Rekognition)
- Notification after processing (Amazon SNS)

AWS Services Used

- **Amazon S3** – Storage and event trigger
 - **AWS Lambda** – Image processing & Rekognition calls
 - **AWS Step Functions** – Orchestration of workflow
 - **Amazon Rekognition** – Image/Video label detection
 - **Amazon SNS** – Notifications
 - **IAM** – Permissions and security
-

2. High-Level Architecture Flow

1. User uploads an image/video to S3 bucket
 2. S3 event triggers Step Functions
 3. Step Functions invokes Lambda functions sequentially
 4. Lambda resizes image / processes video
 5. Lambda calls Rekognition for label detection
 6. Results are sent via SNS notification
-

3. Step-by-Step AWS Console Setup

Step 1: Create S3 Buckets

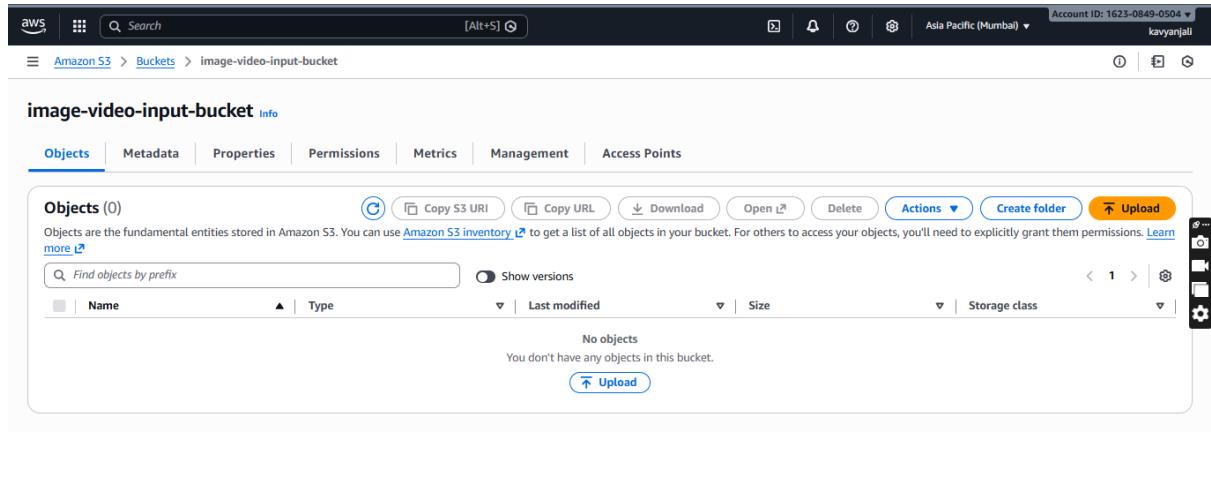
1.1 Create Input Bucket

1. Open **AWS Console** → **S3** → **Create bucket**
2. Bucket name: image-video-input-bucket-<unique>
3. Region: Same region for all services
4. Block all public access → **Enabled**
5. Click **Create bucket**

1.2 Create Output Bucket

1. Click **Create bucket**
2. Bucket name: image-video-output-bucket-<unique>
3. Keep default settings

4. Click Create bucket

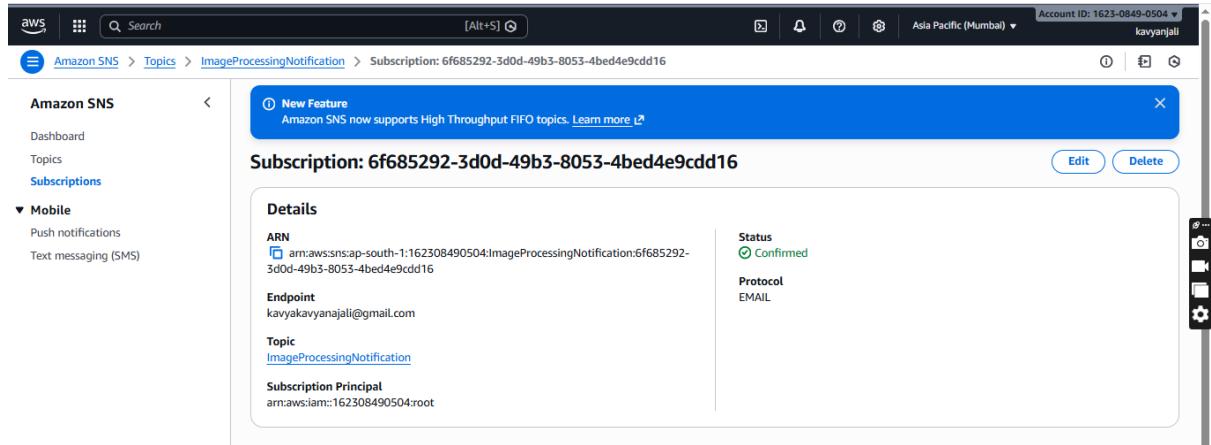


Step 2: Create SNS Topic

1. Go to **SNS** → **Topics** → **Create topic**
2. Type: **Standard**
3. Name: ImageProcessingNotification
4. Click **Create topic**

Subscribe Email

1. Open topic → **Create subscription**
2. Protocol: **Email**
3. Endpoint: Your email address
4. Confirm subscription from email



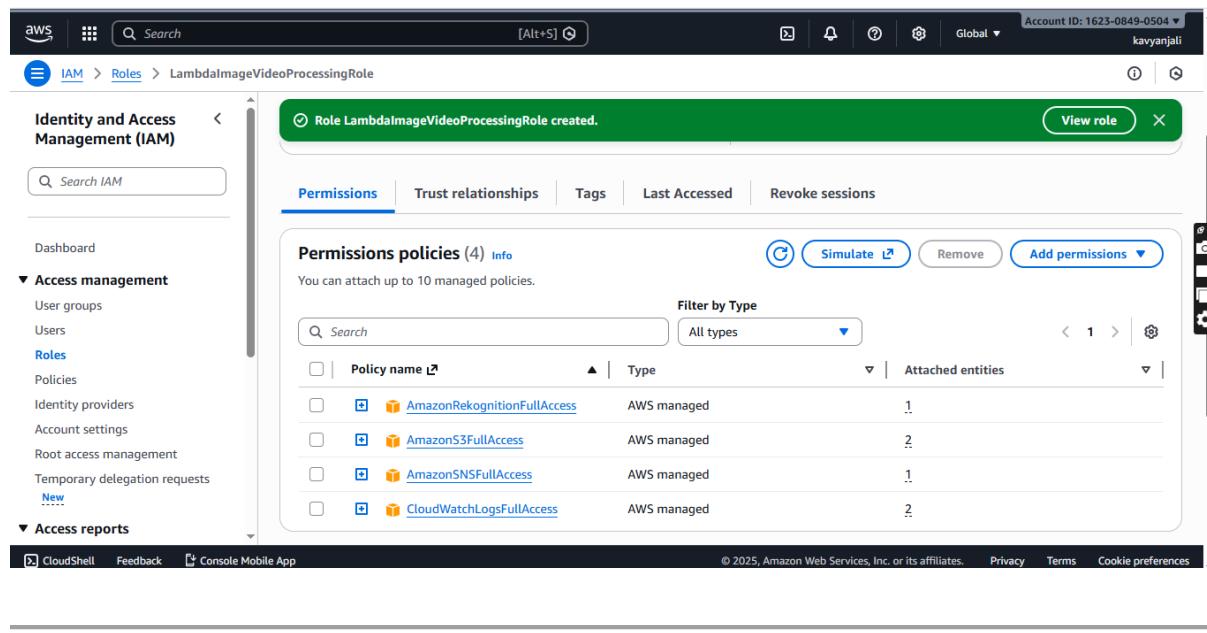
Step 3: Create IAM Role for Lambda

1. Go to **IAM** → **Roles** → **Create role**
2. Trusted entity: **AWS service**
3. Use case: **Lambda**
4. Attach policies:
 - AmazonS3FullAccess

- AmazonRekognitionFullAccess
 - AmazonSNSFullAccess
 - CloudWatchLogsFullAccess

5. Role name: LambdaImageVideoProcessingRole

6. Click **Create role**

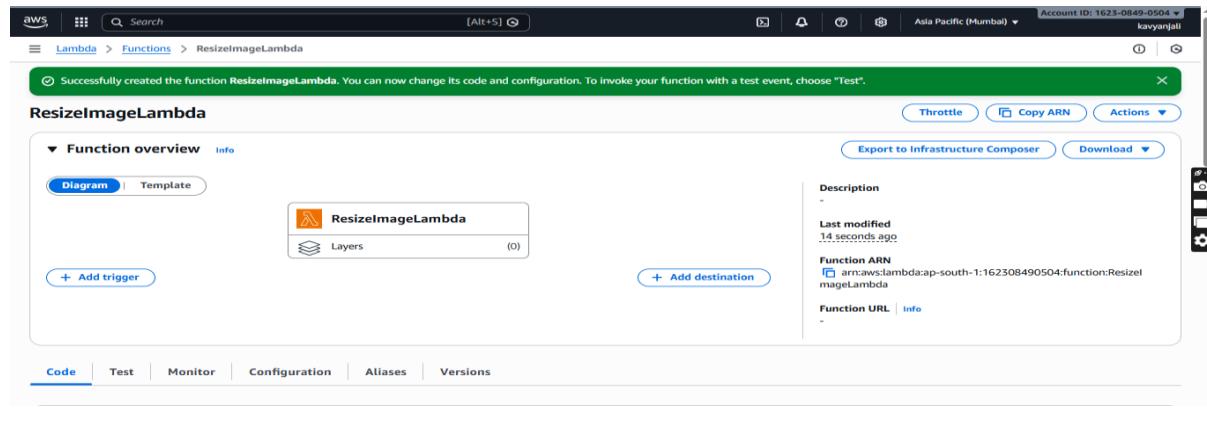


Step 4: Create Lambda Function – Image Resize

1. Go to **Lambda** → **Create function**
 2. Author from scratch
 3. Function name: ResizeImageLambda
 4. Runtime: **Python 3.12**
 5. Execution role: Use existing role
 6. Select: LambdaImageVideoProcessingRole
 7. Click **Create function**

Sample Logic (Concept)

- Read image from input S3 bucket
 - Resize image
 - Save resized image to output bucket



Step 5: Create Lambda Function – Rekognition Labels

1. Create another Lambda function
2. Function name: RekognitionLabelLambda
3. Runtime: Python 3.12
4. Role: LambdaImageVideoProcessingRole

Function Responsibility

- Receive S3 object details
- Call Amazon Rekognition detect_labels
- Store labels in output bucket
- Publish message to SNS

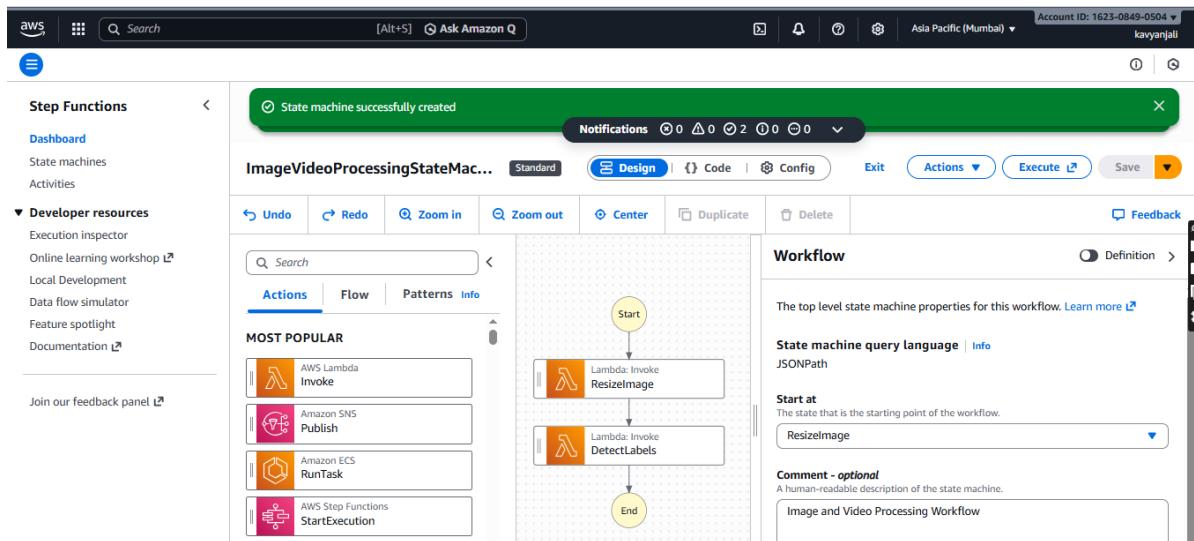
The screenshot shows the AWS Lambda function configuration page for 'RekognitionLabelLambda'. At the top, there's a success message: 'Successfully created the function RekognitionLabelLambda. You can now change its code and configuration. To invoke your function with a test event, choose "Test".' Below this, the function name 'RekognitionLabelLambda' is displayed, along with 'Throttle', 'Copy ARN', and 'Actions' buttons. On the left, under 'Function overview', there are tabs for 'Diagram' (selected) and 'Template'. The 'Diagram' view shows a single Lambda function icon labeled 'RekognitionLabelLambda' with '(0)' layers. There are buttons for '+ Add trigger' and '+ Add destination'. On the right, there are sections for 'Description' (empty), 'Last modified' (3 seconds ago), 'Function ARN' (arn:aws:lambda:ap-south-1:162308490504:function:RekognitionLabelLambda), and 'Function URL' (empty). A sidebar on the right contains icons for more options.

Step 6: Create Step Functions State Machine

1. Go to AWS Step Functions → State machines → **Create state machine**
2. Type: **Standard**
3. Choose **Author with code**

Example Workflow

- Step 1: Resize Image Lambda
 - Step 2: Rekognition Label Detection Lambda
 - Step 3: SNS Notification
4. Execution role: Create new role
 5. Name: ImageVideoProcessingStateMachine
 6. Click **Create state machine**



Step 7: Configure S3 Event Trigger

Option A: Using Lambda Trigger

1. Go to **S3 → Input bucket → Properties**
2. Scroll to **Event notifications**
3. Create event notification
4. Event type: **PUT**
5. Destination: **Step Functions**
6. Select state machine

The screenshot shows the AWS S3 console's event notifications configuration for an input bucket. It lists a single notification rule named 'TriggerImageProcessing' for PUT events, which is triggered by the 'ImageProcessingNotification' Step Function. The interface includes sections for Amazon EventBridge and Transfer acceleration.

Step 8: Test the Workflow

1. Upload an image or video to input bucket
2. Open **Step Functions → Executions**
3. Monitor workflow progress
4. Check output bucket for processed files

5. Verify email notification

The screenshot shows the AWS Step Functions console. A green banner at the top indicates "Execution started successfully". Below it, the execution details for "Execution: 473a9b4d-ec8e-4e50-adf5-0107406e0043" are displayed. The execution status is "Succeeded". Other details include the start time (Dec 13, 2025, 21:48:56.095 UTC+05:30), end time (Dec 13, 2025, 21:48:56.935 UTC+05:30), duration (0:00:00.840), and version (1). The IAM role ARN is listed as arn:aws:iam::162308490504:role/service-role/StepFunctions-ImageVideoProcessingStateMachine-role-98t7vh40f.

The screenshot shows the AWS Lambda S3 File Upload successful summary page. It displays a summary table with one succeeded file (1 file, 19.0 MB) and zero failed files (0 files, 0 B). The destination is s3://image-video-input-bucket. Below this, a table lists the uploaded file: 16.10.2025_13.05.53_REC.mp4, which is a video/mp4 file of size 19.0 MB and status Succeeded.

9. Monitoring & Logging

- CloudWatch Logs for Lambda execution
- Step Functions Execution History
- S3 access logs (optional)
- Check logs for: ResizeImageLambda, RekognitionLabelLambda

The screenshot shows the AWS CloudWatch Log management Log groups page. It lists seven log groups: /aws-glue/crawlers, /aws/lambda/RekognitionLabelLambda, and /aws/lambda/ResizeImageLambda. Each log group is set to Standard log class, Off for Anomaly detection, and Off for Delete on retention. The retention period is set to Never expire. The page also includes filters, actions, and links to view logs insights and start tailing.

10. Security Best Practices

- Use least-privilege IAM policies
 - Keep S3 buckets private
 - Enable encryption at rest
 - Enable CloudTrail for auditing
-

11. Skills Demonstrated

- Serverless Architecture
 - Event-driven systems
 - AWS Lambda orchestration
 - Image & video processing
 - AWS Rekognition integration
 - Monitoring and notifications
-

12. Conclusion

This project demonstrates a complete serverless image/video processing pipeline using AWS managed services. It is scalable, cost-effective, and requires no server management, making it ideal for real-time media processing applications.