Project: Serverless Image Processing using AWS Lambda

**AIM**

To create a serverless image processing system that:

* Automatically resizes and optimizes images uploaded to an S3 bucket (Bucket A).
* Routes the processed image to another bucket (Bucket A or B) based on size:
  + 5MB → **Bucket A**
  + <2MB → **Bucket B**

**TECHNICAL SKILLS USED**

* AWS S3 – File storage and event triggering
* AWS Lambda – Serverless compute for image processing
* AWS IAM – Permissions management
* AWS SDK (Boto3) – Python AWS interaction
* Python (Pillow library) – Image processing
* CloudWatch – Logging and debugging

**STEPS FOLLOWED**

1. **Create Two S3 Buckets**
   * image-bucket-original (for uploading unprocessed images)
   * image-bucket-large (for images >5MB)
   * image-bucket-small (for images <2MB)
2. **Write Lambda Function**
   * Language: Python 3.9
   * Libraries: boto3, Pillow
   * Functionality:
     + Triggered on ObjectCreated event from image-bucket-original
     + Download image
     + Resize to predefined dimensions (e.g., 1024x1024 max)
     + Optimize and compress
     + Determine size
     + Upload to appropriate bucket
3. **Create IAM Role for Lambda**
   * Allow access to:
     + Read from image-bucket-original
     + Write to image-bucket-large and image-bucket-small
     + Write to CloudWatch Logs
4. **Deploy Lambda Function**
   * Via AWS Console or ZIP deployment
   * Add trigger for image-bucket-original
5. **Test**
   * Upload images to image-bucket-original
   * Validate resizing and routing to correct buckets

**CHALLENGES DURING FOLLOWING STEPS**

* **Handling Different Image Formats**
  + Required conditional processing for .png, .jpg, .jpeg, etc.
* **Lambda Size Limit**
  + Needed to package Pillow library as a layer due to size constraints
* **IAM Permission Errors**
  + Fine-tuning policies to allow S3 actions and logs access
* **Compression vs. Quality Trade-off**
  + Balancing optimization without losing clarity

**TROUBLESHOOTING STEPS**

| **Problem** | **Solution** |
| --- | --- |
| Lambda timeout during processing | Increased timeout duration in settings |
| Permission denied on S3 upload | Modified IAM role to allow s3:PutObject |
| Image not resized properly | Checked image mode (RGB vs. RGBA), converted if needed |
| File size not changing much | Adjusted JPEG compression and quality level (e.g., 85%) |
| Pillow import error | Deployed using Lambda Layers with dependencies packaged |

**FINAL STATEMENT**

This project successfully demonstrates a fully serverless approach to real-time image processing using AWS services. By leveraging S3 and Lambda, images are dynamically resized and routed based on file size without the need for server provisioning. This architecture is scalable, cost-effective, and ideal for applications requiring dynamic media optimization.