Project 1: Executive Summary & Implementation Guide

S3 → Lambda → SNS Serverless Image Thumbnail Pipeline

By: Islam Zain

PROJECT SUMMARY

Project 1 is a complete **serverless image processing pipeline** that automatically generates thumbnail images when photos are uploaded to Amazon S3. The system sends real-time email notifications using Amazon SNS, providing a production-ready, scalable, and cost-effective solution.

Key Deliverables:

1. **Architecture Diagram** - Visual representation of all components
2. **Complete Lab Guide (11 pages)** - Step-by-step implementation instructions
3. **Terraform Configuration** - Infrastructure as Code for automated deployment
4. **Lambda Function Code** - Production-ready Python code with error handling
5. **Quick Reference Guide** - CLI commands and troubleshooting tips
6. **Process Flowchart** - End-to-end workflow visualization

ARCHITECTURE COMPONENTS

1. Source S3 Bucket (Images)

- Stores original uploaded images
- Triggers Lambda on object creation
- Configured for versioning and encryption
- Public access blocked for security

2. Lambda Function (Processor)

- Triggered by S3 PUT events
- Downloads original image
- Generates 200x200px thumbnail using Python PIL
- Uploads thumbnail to destination bucket
- Publishes notification to SNS

- **Memory:** 512 MB | **Timeout:** 30 seconds

3. Destination S3 Bucket (Thumbnails)

- Stores generated thumbnail images
- Organized in `/thumbnails/` prefix
- Same security settings as source bucket
- Metadata tags for tracking

4. SNS Topic (Notifications)

- Receives processed image events from Lambda
- Distributes notifications to email subscribers
- Sends both success and error messages
- Email subscription with confirmation

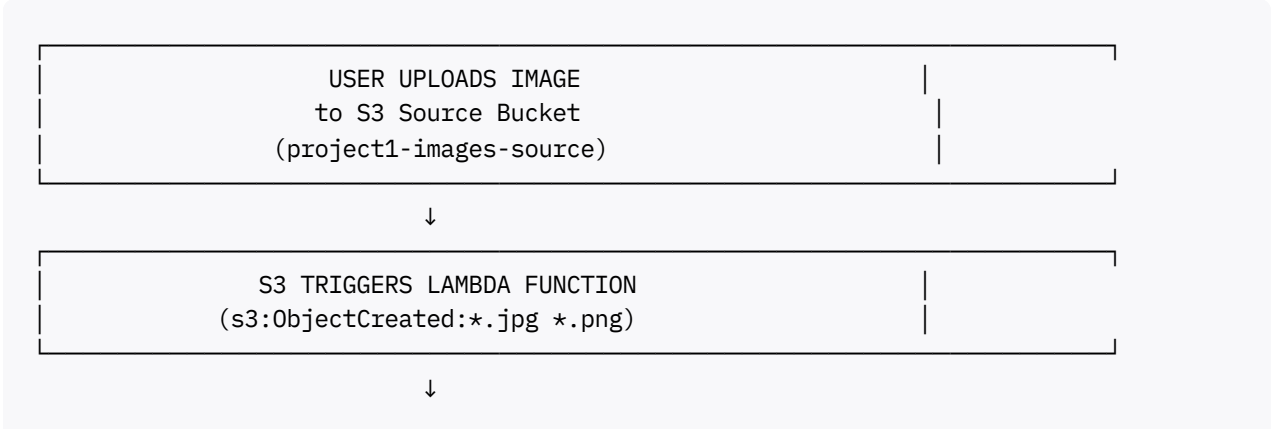
5. IAM Role (Permissions)

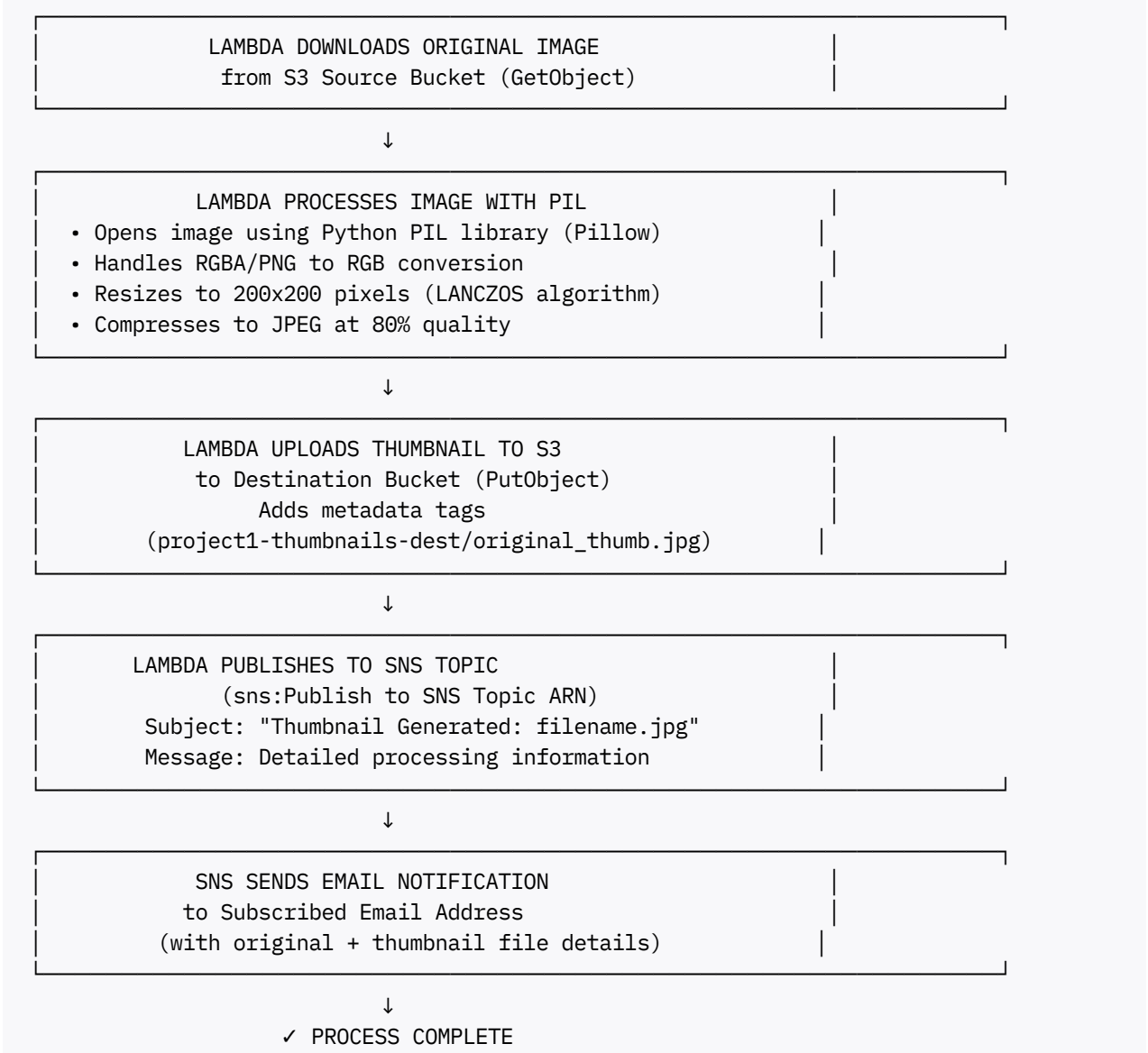
- Least privilege access policy
- S3: GetObject on source bucket only
- S3: PutObject on destination bucket only
- SNS: Publish to specific topic only
- CloudWatch Logs for monitoring

6. CloudWatch (Monitoring)

- Captures Lambda execution logs
- Retention: 14 days
- Alarms for errors and duration
- Integration with SNS for alerting

WORKFLOW PROCESS





FREE TIER ANALYSIS

Monthly Free Tier Allocation:

Service	Limit	10 Images	100 Images	1000 Images
S3 Storage	5 GB	✓ FREE	✓ FREE	~$0.10
S3 Requests	20K	✓ FREE	✓ FREE	✓ FREE
Lambda Calls	1M	✓ FREE	✓ FREE	✓ FREE
Lambda GB-sec	400K	✓ FREE	✓ FREE	~$0.30
SNS Emails	1K	✓ FREE	✓ FREE	✓ FREE
CloudWatch Logs	10 GB	✓ FREE	✓ FREE	~$0.50
TOTAL COST	-	**$0**	**$0**	**~$0.90**

Conclusion: The Free Tier easily covers development, testing, and small production workloads. Costs only accumulate for high-volume scenarios (1000+ images/month).

DEPLOYMENT OPTIONS

Option 1: AWS Console (Manual - 20 minutes)

Pros: Visual interface, no CLI required, good for learning
Cons: More manual steps, error-prone for production

Steps:

1. Create S3 buckets via console
2. Create SNS topic via console
3. Create IAM role via console
4. Write Lambda code in console editor
5. Configure S3 trigger via console UI

Option 2: AWS CLI (Scripts - 15 minutes)

Pros: Reproducible, scriptable, good for CI/CD
Cons: Requires CLI knowledge

Implementation:

```
# Run all 10 steps from Quick Reference Guide
source ./deploy.sh
```

Option 3: Terraform (Infrastructure as Code - 10 minutes)

Pros: Version controlled, reusable, professional
Cons: Requires Terraform knowledge

Implementation:

```
terraform init
terraform apply -var-file=terraform.tfvars
```

Recommendation: Use **Terraform** for production, **AWS Console** for learning, **CLI** for automation.

DEPLOYMENT STEPS (TERRAFORM)

Prerequisites:

```
# Verify tools installed
terraform --version  # v1.0+
aws --version        # AWS CLI v2
python3 --version    # 3.9+
```

Step 1: Prepare Files

```
project1/
├── terraform-config.tf
├── lambda-function.py
├── requirements.txt
└── terraform.tfvars
```

Step 2: Configure Variables

```
# terraform.tfvars
aws_region = "us-east-1"
project_name = "project1"
sns_email = "your-email@example.com"
```

Step 3: Package Lambda

```
mkdir -p lambda_build
cd lambda_build
cp ../lambda-function.py index.py
pip install -r requirements.txt -t .
zip -r ../lambda_function.zip . -x "*.git*"
cd ..
```

Step 4: Validate Configuration

```
terraform fmt          # Format HCL
terraform validate     # Check syntax
terraform plan         # Preview changes
```

Step 5: Deploy

```
terraform apply -auto-approve
```

Step 6: Verify Deployment

```
terraform output       # Show resource details
aws s3 ls             # Check buckets
aws lambda list-functions --query 'Functions[?contains(FunctionName, `project1`)]'
```

TESTING PROCEDURE

Test 1: Console Upload

1. Navigate to S3 source bucket in AWS Console
2. Upload a JPEG or PNG image
3. Wait 30 seconds
4. Navigate to destination bucket
5. Verify thumbnail created (200x200 pixels)
6. Check email for SNS notification

Test 2: CLI Upload

```
aws s3 cp test-image.jpg s3://project1-images-source-ACCOUNT-ID/
sleep 30
aws s3 ls s3://project1-thumbnails-dest-ACCOUNT-ID/
```

Test 3: Verify Logs

```
aws logs tail /aws/lambda/project1-thumbnail-generator --follow --since 5m
```

Test 4: Check Email

- Subject: "Thumbnail Generated: test-image.jpg"
- Contains: Original size, thumbnail size, bucket names, timestamp

Expected Results:

✓ Thumbnail appears in destination bucket within 30 seconds
✓ File size reduces from original (e.g., 2MB → 15KB)
✓ Dimensions are 200x200 pixels
✓ Email notification arrives within 1 minute
✓ CloudWatch logs show successful execution

⚠ ERROR HANDLING & RESILIENCE

Built-in Error Handling:

1. **Image Format Conversion** - Handles RGBA/PNG → RGB automatically
2. **Dimension Preservation** - Maintains aspect ratio with thumbnail() method
3. **File Not Found** - Lambda gracefully handles corrupted uploads
4. **SNS Failures** - Lambda continues even if SNS publish fails
5. **Timeout Protection** - 30-second limit prevents indefinite execution

Monitoring & Alerts:

```
# CloudWatch Alarm: Lambda Errors
Threshold: 5 errors in 5 minutes
Action: Send SNS email alert

# CloudWatch Alarm: High Duration
Threshold: Average &gt; 25 seconds
Action: Send SNS email alert

# CloudWatch Logs Insights:
fields @timestamp, @message, @duration
| filter @message like /ERROR/
| stats count() by @message
```

Troubleshooting Matrix:

Error	Cause	Solution
Lambda timeout	Large image	Reduce image size or increase timeout
AccessDenied	IAM permissions	Review policy, add missing permissions
FileNotFound	Bucket name mismatch	Verify bucket names in env variables
No thumbnail	Trigger not configured	Re-add S3 trigger to Lambda
PIL import error	Missing layer	Add Lambda layer with Pillow

SECURITY CONSIDERATIONS

✓ Implemented Security:

- S3 buckets: Public access blocked
- S3 encryption: AES-256 at rest
- IAM: Least privilege principles
- Lambda: No hardcoded credentials

- SNS: Topic policy restricts publishers
- CloudWatch: Logs retained for audit trail

Additional Hardening:

1. Enable S3 bucket versioning
2. Enable MFA delete on production bucket
3. Use S3 lifecycle policies for cleanup
4. Enable CloudTrail for API auditing
5. Implement VPC endpoints for private S3 access
6. Use KMS keys for encryption
7. Set up AWS Config for compliance

SCALABILITY & PERFORMANCE

Current Capacity:

- **Concurrent Executions:** 1000 (AWS default)
- **Throughput:** 1000+ images/minute
- **Processing Time:** 2-5 seconds per image
- **Storage Cost:** Minimal (thumbnails ~5% of original size)

Scaling Considerations:

1. **Reserved Concurrency:** Lock in capacity for critical workloads
2. **Lambda Memory:** Increase to 1024 MB for faster execution
3. **S3 Bucket Keys:** Enable for request rate optimization
4. **DynamoDB:** Add for logging/tracking processed images
5. **SQS Buffer:** Add for high-volume scenarios
6. **CloudFront:** Add for global distribution

FILES PROVIDED

1. **Project1-S3-Lambda-SNS-Lab.pdf** (11 pages)
    - Complete step-by-step lab guide
    - Prerequisites and setup instructions
    - Troubleshooting guide
    - Best practices and advanced enhancements
2. terraform-config.tf

- Complete infrastructure configuration
- Resource definitions for all AWS services
- Variables and outputs

3. lambda-function.py
   - Production-ready Lambda handler
   - Error handling and logging
   - PIL image processing
   - SNS notification formatting
4. quick-reference.md
   - CLI commands (AWS, Terraform)
   - 10-step quick start
   - Cost monitoring
   - Useful debugging commands
5. **Architecture Diagram** (generated_image:20)
   - Professional AWS service architecture
   - Component relationships
   - Data flow visualization
6. **Workflow Flowchart** (chart:25)
   - Process steps from upload to email
   - Decision points and error paths
   - Service interactions

LEARNING OUTCOMES

After completing Project 1, you will understand:

- ✓ **AWS S3 Event Notifications** - How S3 triggers Lambda functions
- ✓ **Lambda Function Development** - Creating and deploying Python functions
- ✓ **IAM & Access Control** - Implementing least privilege security
- ✓ **SNS Topic Management** - Email notifications and subscriptions
- ✓ **Image Processing** - Using Python PIL/Pillow library
- ✓ **CloudWatch Logging** - Monitoring and debugging Lambda
- ✓ **Serverless Architecture** - Building scalable, event-driven systems
- ✓ **Infrastructure as Code** - Terraform configurations
- ✓ **AWS CLI Automation** - Scripting AWS operations
- ✓ **Production Best Practices** - Error handling, monitoring, security

🚀 NEXT STEPS & EXTENSIONS

Beginner Extensions:

1. Add multiple thumbnail sizes (small, medium, large)
2. Support additional image formats (GIF, WebP)
3. Add image format conversion (to WebP for compression)
4. Create Lambda function for cleanup

Intermediate Extensions:

1. Add DynamoDB table for processing history
2. Create REST API with API Gateway
3. Add CloudWatch dashboards
4. Implement SQS queue for batch processing
5. Add image metadata extraction (EXIF)

Advanced Extensions:

1. Multi-region deployment
2. CloudFront distribution with caching
3. Lambda@Edge for on-demand resizing
4. Advanced image processing (filters, watermarks)
5. ML integration (rekognition, auto-tagging)
6. Automated testing pipeline
7. CI/CD deployment

📚 SUPPORT & RESOURCES

AWS Documentation:

- Lambda S3 Integration: https://docs.aws.amazon.com/lambda/latest/dg/services-s3.html
- S3 Event Notifications: https://docs.aws.amazon.com/AmazonS3/latest/userguide/EventNotifications.html
- SNS Developer Guide: https://docs.aws.amazon.com/sns/latest/dg/

Python Libraries:

- Pillow (PIL): https://pillow.readthedocs.io/
- Boto3: https://boto3.amazonaws.com/v1/documentation/api/latest/index.html

Community:

- AWS Reddit: r/aws, r/cloudcomputing
- Stack Overflow: Tag with aws, lambda, s3
- AWS Workshops: https://workshops.aws/

✅ COMPLETION CHECKLIST

- [ ] Downloaded all project files
- [ ] Read architecture overview
- [ ] Reviewed Terraform configuration
- [ ] Reviewed Lambda function code
- [ ] Set up AWS environment (CLI, credentials)
- [ ] Created S3 buckets
- [ ] Created SNS topic and subscribed
- [ ] Created IAM role with policies
- [ ] Deployed Lambda function
- [ ] Configured S3 trigger
- [ ] Tested with sample image
- [ ] Verified email notification
- [ ] Reviewed CloudWatch logs
- [ ] Cleaned up test resources
- [ ] Documented learnings

Project 1 Ready for Deployment!

Complete Serverless Image Thumbnail Generation Pipeline
Created by: Islam Zain
AWS Solutions Architecture Practice Project
[1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

⁂

1. https://docs.aws.amazon.com/lambda/latest/dg/with-s3-tutorial.html
2. https://stackoverflow.com/questions/54760261/what-is-the-best-approach-for-generating-thumbnails-and-uploading-to-s3-bucket-i

3. https://www.linkedin.com/pulse/automating-aws-lambda-invocation-triggered-s3-event-sns-swati-kanungo-ggw3f
4. https://www.educative.io/cloudlabs/resizing-images-with-s3-batch-operations-and-aws-lambda
5. https://www.norrapscm.com/posts/2021-02-08-generate-thumbnails-in-lambda-from-s3-with-ffmpeg/
6. https://www.jeeviacademy.com/aws-s3-to-sns-simplifying-notification-delivery/
7. https://www.kubeblogs.com/resize-images-at-scale-without-breaking-a-sweat/
8. https://dev.to/nadaahmed/automating-image-thumbnails-with-aws-lambda-and-s3-a-step-by-step-guide-36dd
9. https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/subscribe-a-lambda-function-to-event-notifications-from-s3-buckets-in-different-aws-regions.html
10. https://stackoverflow.com/questions/60486873/resize-and-move-images-using-lambda-function-from-one-s3-to-another-s3