AWS CLOUD AUTOMATION PROJECT: INTELLIGENT IMAGE RESIZING & STORAGE OPTIMIZATION USING LAMBDA AND S3

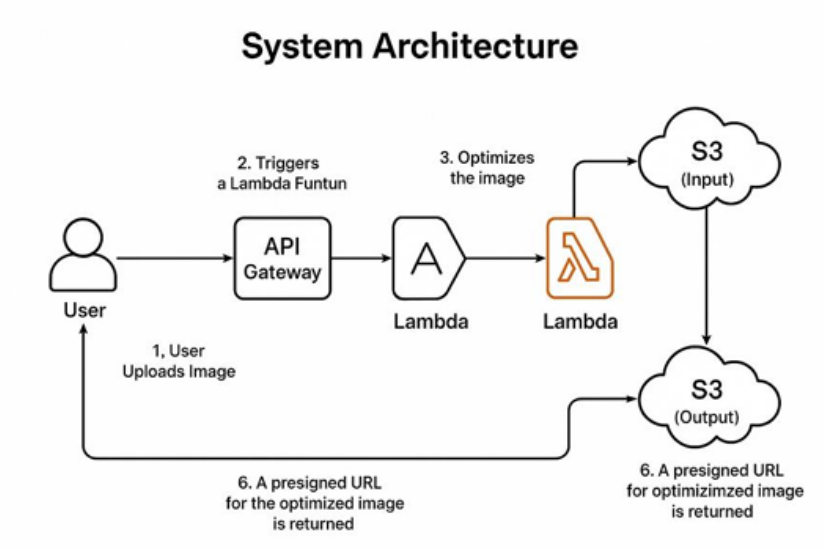
INTERMEDIATE PROGRESS REPORT

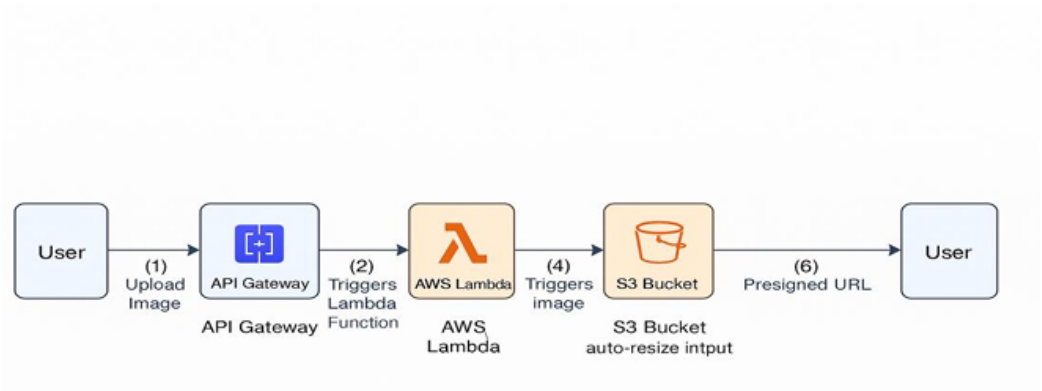
Team Members & Responsibilities:

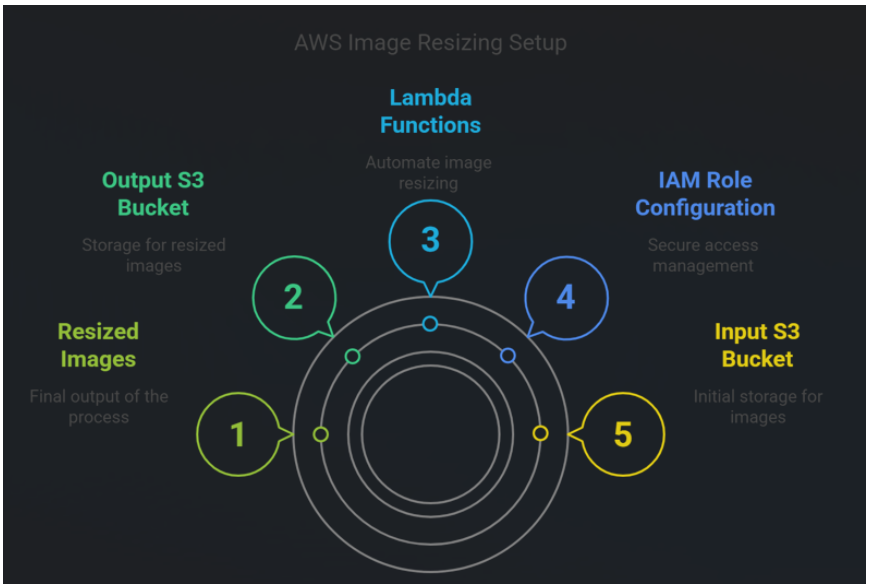
- **Shreya Saraf** (16369800) – Compute & API Development
  Implement Lambda functions for image processing. Configure API Gateway to handle image uploads securely.
- **Sreevatsava Reddy Musani** (16371299) – Storage & Access Management
  Design and configure S3 storage for input and output images. Implement secure access via presigned URLs.
- **Sudhakar Reddy Jerribanda** (16371361) – Image Optimization & Processing
  Integrate Sharp/Pillow for resizing and format conversion. Optimize image quality for efficient storage and retrieval.
- **Hariram Sabari Kriesh** (16371298) – Monitoring & Visualization
  Set up CloudWatch for logging and performance tracking. Develop a simple frontend for testing and demonstrating uploads.

PROJECT OVERVIEW This project demonstrates how to automate image resizing and optimize cloud storage using AWS services. The system automatically resizes images uploaded to a source S3 bucket and stores the optimized version in a destination bucket. This is highly beneficial for managing large image files in web or mobile applications, while reducing storage costs and improving performance. The automation is achieved through AWS Lambda triggered by S3 events.

ARCHITECTURE DIAGRAM







PROGRESS SUMMARY (WEEKS 1–4)

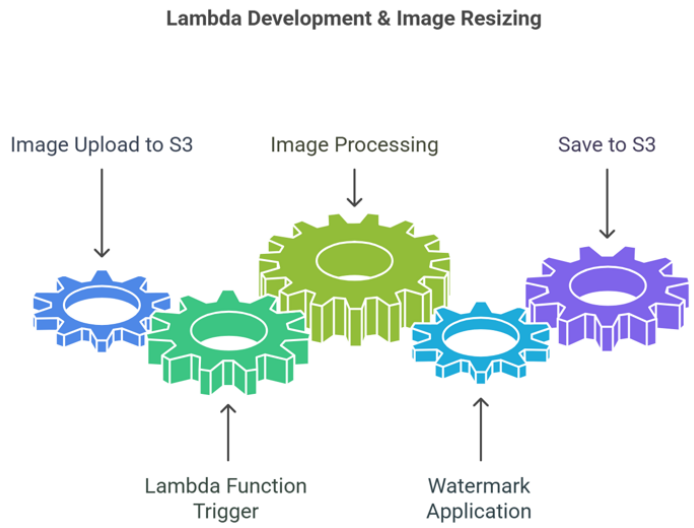
WEEKS 1–2: AWS INFRASTRUCTURE SETUP

- Created input and output S3 buckets with lifecycle configurations:
    - auto-resize-input: 7-day retention for raw uploads
    - auto-resize-output: 30-day retention for processed files
- Assigned IAM roles with scoped permissions for Lambda to access S3

WEEK 3: LAMBDA FUNCTIONS IMPLEMENTATION

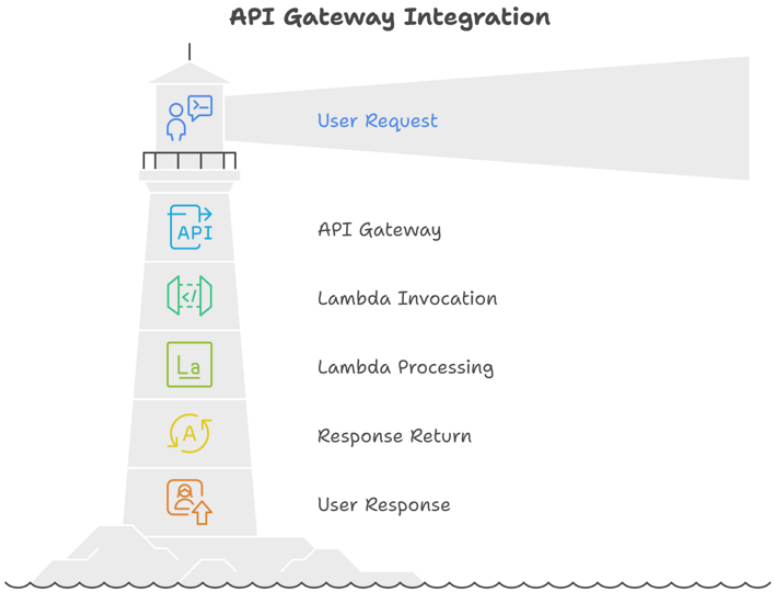
- Developed Lambda function to:
    - Automatically resize image using Sharp
    - Add watermark text overlay on image
    - Store resized image in target S3 bucket
- Performance benchmarking:

Function	Memory	Avg Execution Time
Resize & Watermark	1024MB	700ms



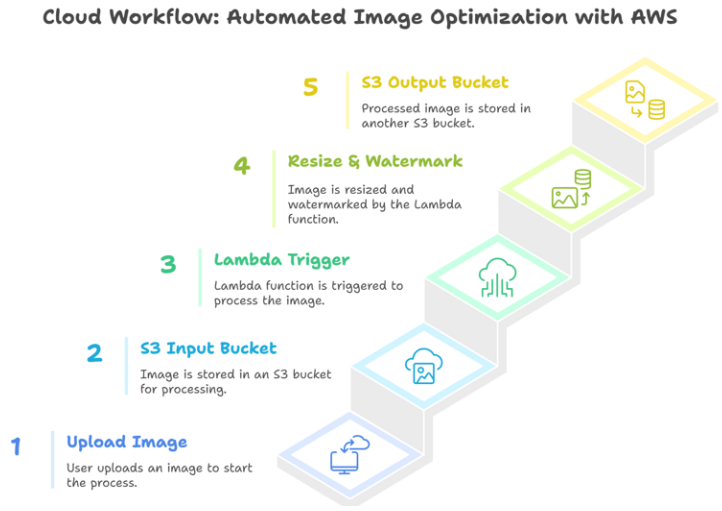
WEEK 4: API GATEWAY INTEGRATION

- Created basic REST API endpoint to test Lambda manually
- Verified functionality using Postman/cURL
- Input validation handled through Lambda code (MIME types, file size)



WORK IN PROGRESS

- Refactoring Lambda to support more file types and improved logging
- Enhancing error handling and retry logic
- Adding configurable watermark position and font styling



CHALLENGES ENCOUNTERED

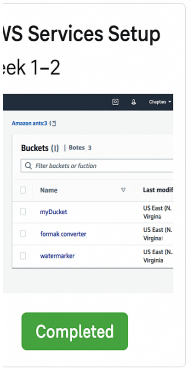
- Temporary permission errors during S3-Lambda access resolved via IAM policy debugging
- Delay in image loading due to file size mitigated through quality adjustment

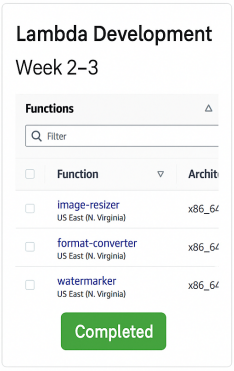
PERFORMANCE SNAPSHOT

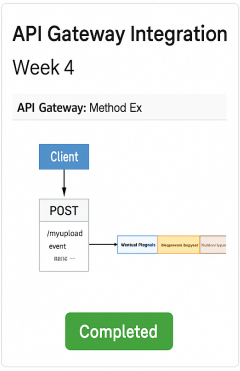
Operation	Success Rate	Avg Duration	Est. Cost per 1000 Executions
Resize & Watermark	99.7%	700ms	$0.020

TIMELINE (PLANNED VS ACTUAL)

Task	Planned Timeline	Status
AWS Services Setup	Week 1-2	Completed
Lambda Development	Week 2-3	Completed
API Gateway Integration	Week 4	Completed
Extended Format Support	Week 5	Ongoing
Logging & Config Enhancements	Week 6	Upcoming







NEXT STEPS

- Finalize support for additional file types and extensions
- Complete enhancements for watermark configuration
- Perform load testing with various image sizes
- Prepare demo walkthrough and documentation

REFERENCES

- Project plan document: CC_Project_Plan_Cloud-Based Image Optimization Pipeline.pdf
- Video tutorial reference: YouTube - Automate Image Resizing with AWS Lambda and S3
- AWS Documentation: Lambda, API Gateway, IAM, S3