Step by Step process of creating end to end pipeline using AWS:

1. Create the S3 Buckets

You need 4 buckets:

Bucket Name	Purpose	Created / Used By
raw-events-bucket	Stores raw webhook payloads	**Lambda A** (Webhook Receiver → writes raw event)
delay-bucket	Stores pending → ready delayed events	**Lambda A** (writes `pending/`) **Lambda B** (moves to `ready/`)
output-bucket	Stores enriched JSON	**Lambda C** (writes enriched output)
lead-owner-lookup	Stores static JSON files for each lead	**Lambda C** (reads owner lookup JSON via HTTPS)

Go to AWS Console → S3 → Create bucket

a. Create each bucket with default settings
b. Turn Block Public Access ON for all buckets except lookup bucket
c. For `lead-owner-lookup` bucket:

   Go to **Permissions**

   Enable **Static website hosting (optional)**

   Add a bucket policy to allow GET:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "PublicRead",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::lead-owner-lookup/*"
  }]
}
```

2. Create IAM Role for Lambda Functions

AWS Console → **IAM** → **Roles** → **Create Role**

Trusted entity → **AWS Service**

Use case → **Lambda**

Attach policies:

- **AmazonS3FullAccess** (or a reduced version later)
- **AmazonSESFullAccess**
- **AWSLambdaBasicExecutionRole**

Name the role:
lambda-crm-lead-pipeline-role

3. Create Lambda A (Webhook Receiver)

Steps

a. AWS Console → **Lambda** → **Create function**
b. Name: **lambdaA_webhook_receiver**
c. Runtime: **Python 3.11**
d. Execution Role: **Use existing** → **lambda-crm-lead-pipeline-role**

Add environment variables

Go to: Configuration → Environment Variables → Edit

Key	Value
RAW_BUCKET	raw-events-bucket
DELAY_BUCKET	delay-bucket

4. Create Lambda B (Delay Processor)

Steps

1. Create Lambda:
    - Name: **lambdaB_delay**
    - Runtime: Python 3.11
    - Role: **lambda-crm-lead-pipeline-role**

Environment Variables

Key	Value
DELAY_BUCKET	delay-bucket

Configure S3 Trigger

a. Go to **delay-bucket**
b. → **Properties** → **Event notifications** → **Create notification**
c. Name: `pendingToReady`
d. Prefix: `pending/`
e. Event type: **PUT**
f. Destination: **Lambda B**

5. Create Lambda C (Enricher + SES Email Sender)

Steps

1. Create Lambda:
    - Name: **lambdaC_enrich**
    - Runtime: Python 3.11
    - Role: **lambda-crm-lead-pipeline-role**

Environment Variables

Key	Value
RAW_BUCKET	raw-events-bucket
DELAY_BUCKET	delay-bucket
OUTPUT_BUCKET	output-bucket
OWNER_LOOKUP_BUCKET	lead-owner-lookup
SES_REGION	us-east-1
EMAIL_FROM	"..."
EMAIL_TO	"..."

Configure S3 Trigger

a. Go to **delay-bucket**
b. → **Event notifications** → **Create**
c. Prefix = `ready/`
d. Event Type = **PUT**
e. Destination = **LambdaC**

6. Set Up AWS SES (Email Sending)

Steps to enable email sending:

a. Console → **SES**
b. Choose Region: **us-east-1**
c. Go to **Verified identities**
d. Click **Create identity**
e. Choose **Email address** → enter: EMAIL_FROM address
f. Check your email inbox → click verification link
g. (If still in SES sandbox): Also verify **EMAIL_TO**