**Name :** Viven Hotwani
**Roll No :** 17
**Class :** D15C
**Topic No :** 17

Case Study

# Serverless Data Processing with DynamoDB - Case Study Report

## 1. Introduction

### Case Study Overview

This case study demonstrates the implementation of a serverless data processing pipeline using AWS services. The system automatically processes JSON files uploaded to an S3 bucket and stores relevant data in DynamoDB, showcasing event-driven architecture and serverless computing principles.

### Key Features and Applications

- **Event-Driven Processing**: Automatic triggering of Lambda functions on S3 file uploads
- **Serverless Architecture**: No server management required, pay-per-use model
- **Real-Time Data Processing**: Immediate processing of uploaded JSON files
- **Scalable Storage**: Use of DynamoDB for flexible, schema-less data storage

### Practical Applications

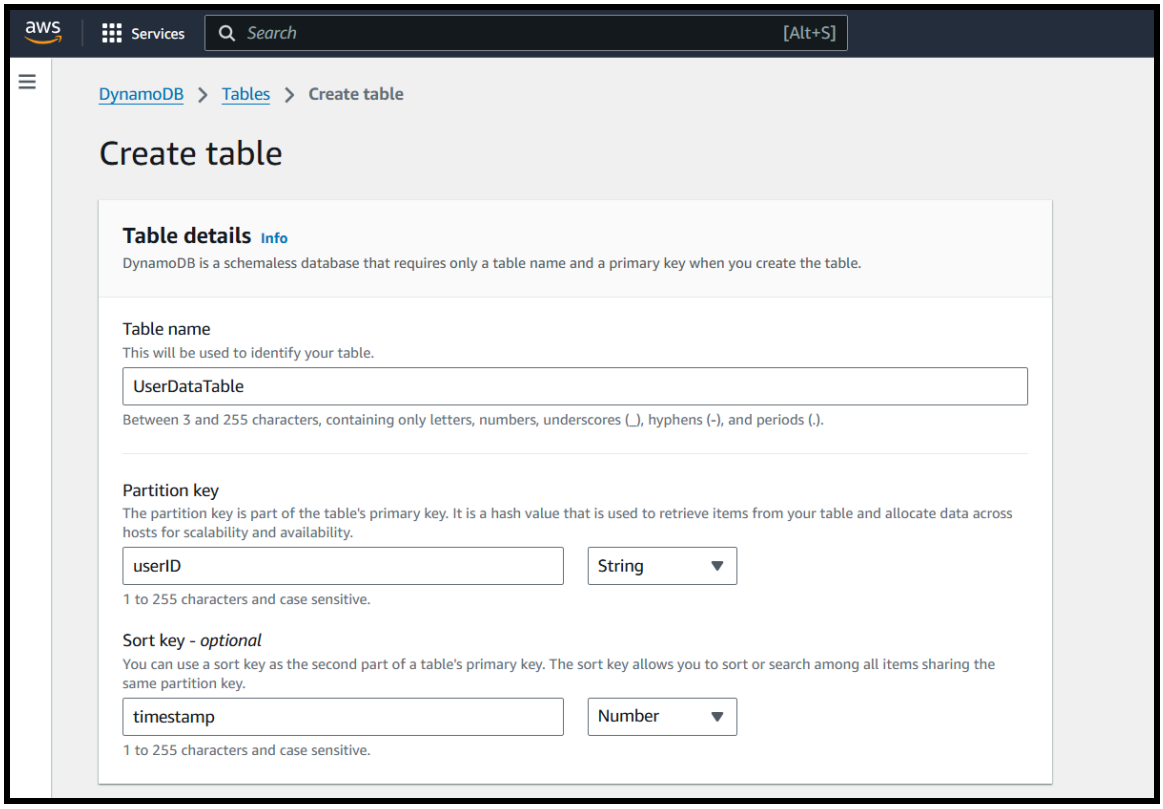
- User activity tracking systems
- E-commerce order processing
- IoT device data collection
- Log file processing and analysis

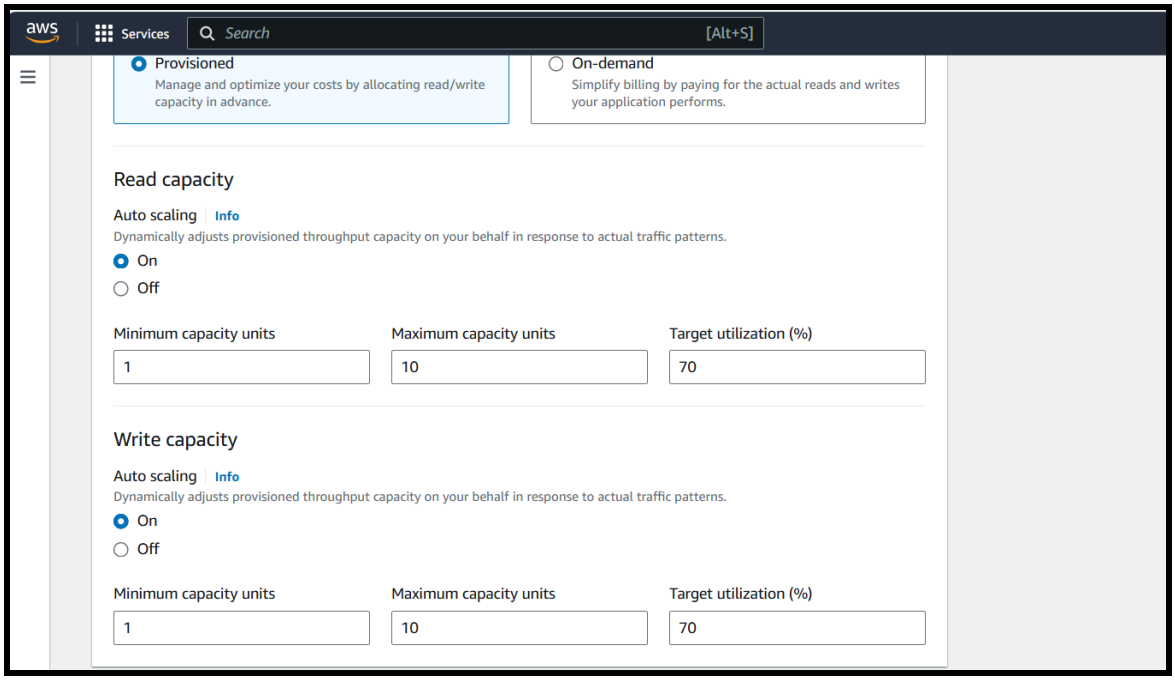
## 2. Step-by-Step Implementation

## Step 1: Creating the DynamoDB Table

1. Navigate to AWS DynamoDB Console
2. Create a new table with the following configuration:
   Table Name: UserDataTable
   Partition Key: userID (String)
   Sort Key: timestamp (Number)

3. Enable auto-scaling for read/write capacity





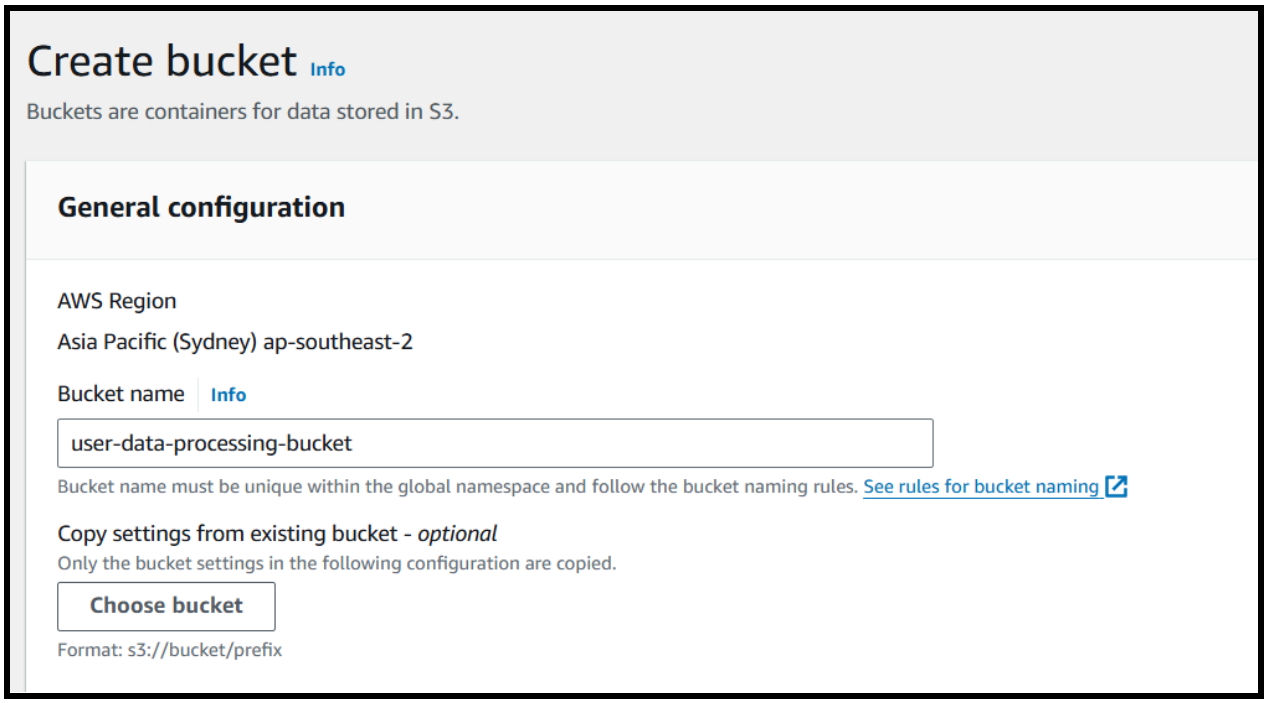

## Step 2: Setting up the S3 Bucket

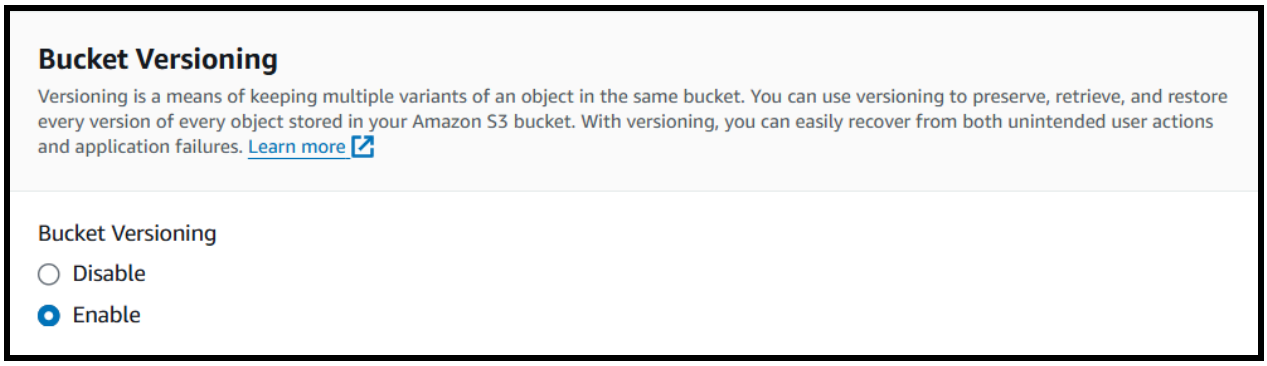
Create a new S3 bucket:

Bucket Name: user-data-processing-bucket
Region: Asia Pacific (Sydney) ap-southeast-2

1. Access: Private

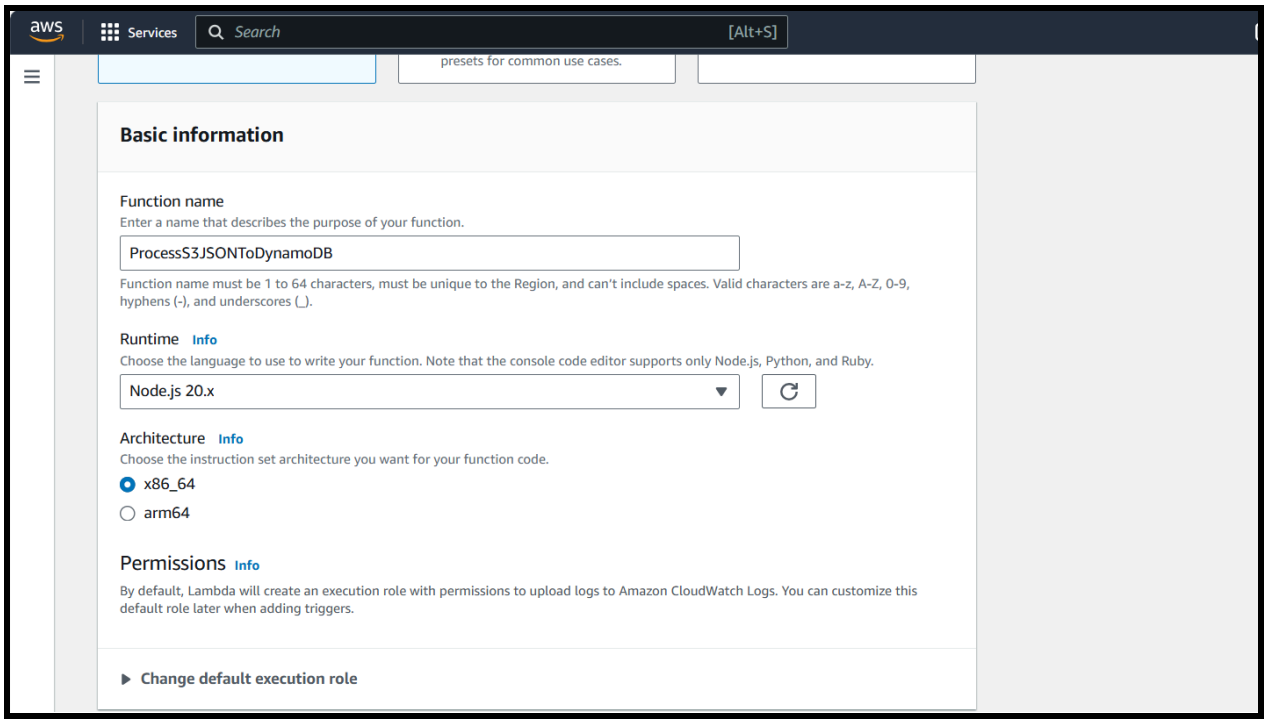
2. Enable versioning for data consistency



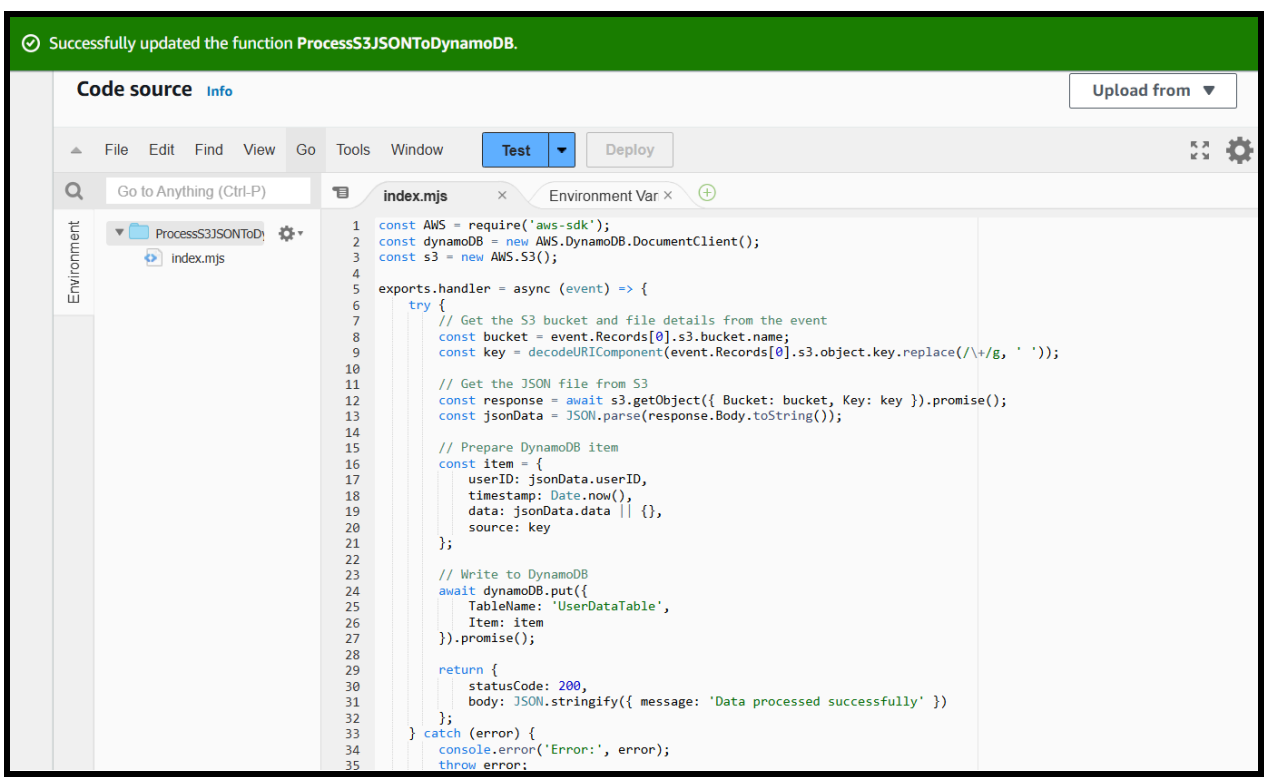




## Step 3: Creating the Lambda Function

- Create a new Lambda function:

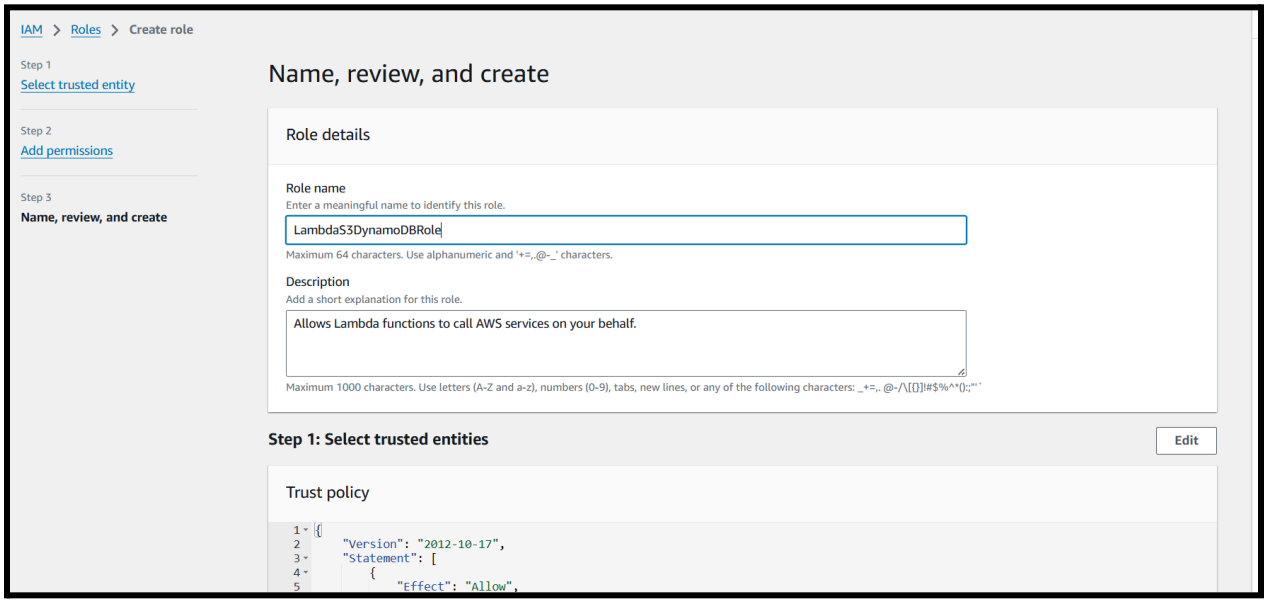


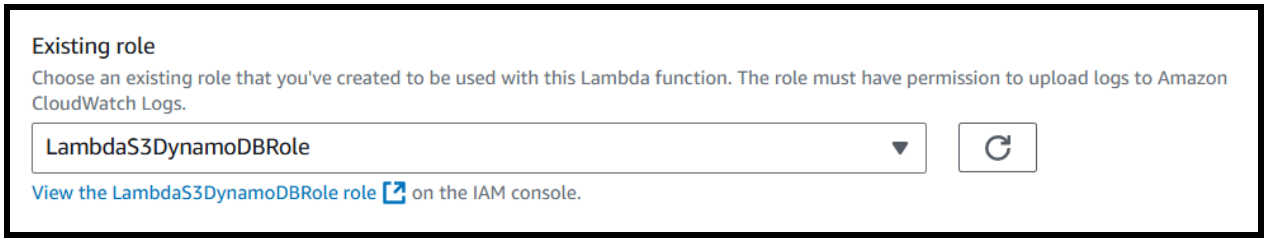

Add the code to Lambda function:



## Step 4: Configuring Permissions

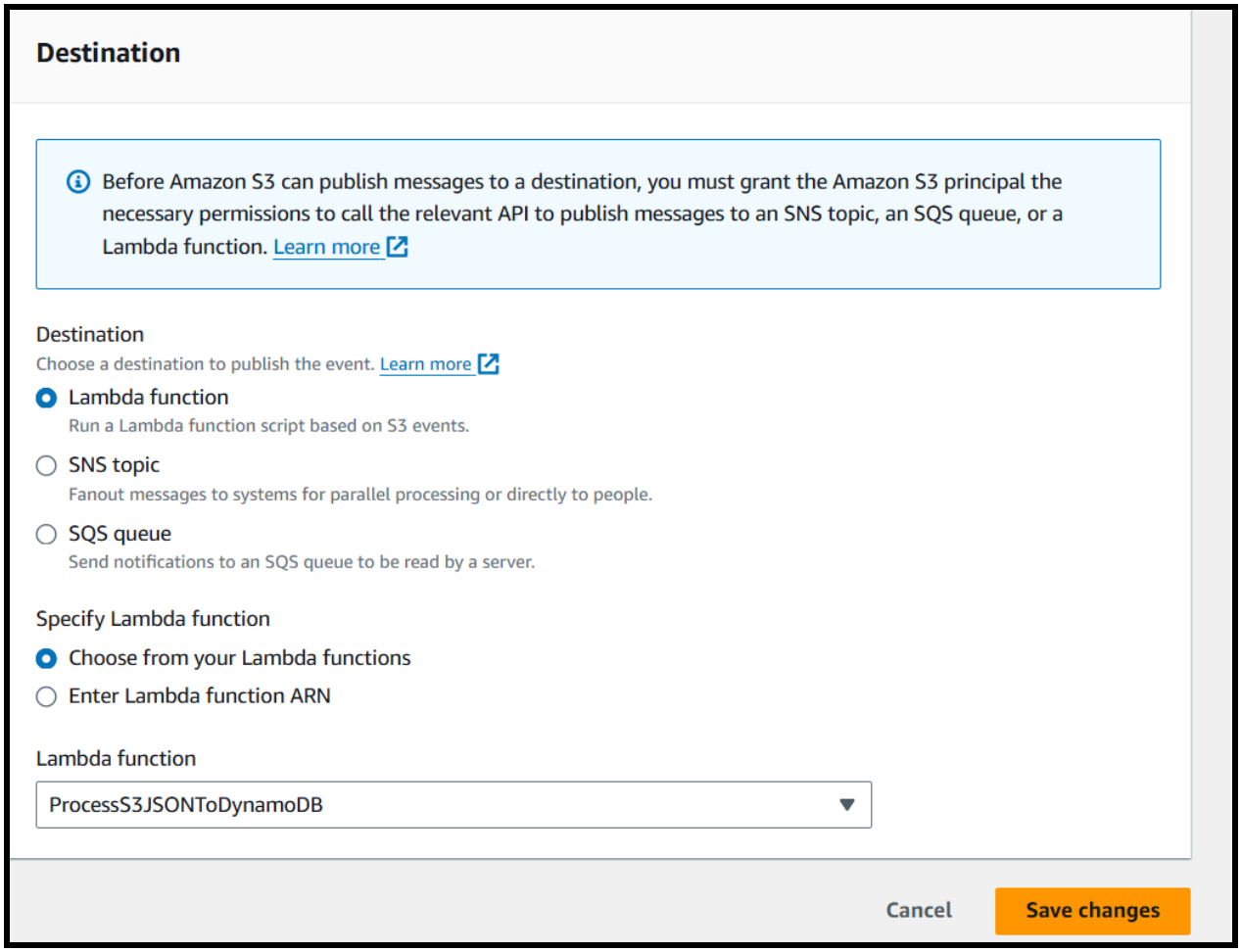
1. Create IAM Role for Lambda with policies:
    - AWSLambdaBasicExecutionRole
    - Custom policy for S3 access
    - Custom policy for DynamoDB access




## Step 5: Setting up S3 Trigger

1. Add S3 trigger to Lambda function
2. Configure for ObjectCreated events
3. Specify .json file suffix filter




# 3. Demonstration Preparation

## Key Points to Cover

1. Architecture Overview
    - Explain serverless benefits
    - Describe event-driven workflow
2. Implementation Details
    - Show DynamoDB table structure
    - Explain Lambda function code
    - Demonstrate S3 trigger configuration
3. Testing Process
    - Upload sample JSON file
    - Show real-time processing
    - Verify DynamoDB entries

### Sample Test Data

Json

```
{
  "userID": "user123",
  "data": {
    "name": "Viven Hotwani",
    "email": "viven.hotwani@example.com",
    "action": "login"
  }
}
```

### Potential Questions and Answers

1. Q: How does the system handle concurrent uploads? A: Lambda automatically scales to handle multiple concurrent executions.
2. Q: What happens if the JSON is malformed? A: Error handling in Lambda catches parsing errors and logs them in CloudWatch.
3. Q: How can we monitor processing failures? A: Through CloudWatch Logs and Metrics, and by setting up CloudWatch Alarms.

## 4. Certification Requirements

### AWS Certification Deadline

- Submission deadline: October 21st, 2024
- Required certification: AWS Certified Developer Associate

### Importance of Certification

- Validates understanding of AWS services
- Demonstrates practical implementation skills
- Required for project evaluation
- Enhances career prospects in cloud computing

### Preparation Tips

1. Complete AWS practice exams
2. Review serverless architecture patterns
3. Understand DynamoDB design principles
4. Practice Lambda function development