## Project - Serverless IoT Data Processing

# Project Title: Serverless IoT Data Processing for Smart Home Automation

**CODING:**

import json

def lambda\_handler(event, context):

try:

# Extract IoT data from the event

iot\_data = json.loads(event['body'])

# Perform data preprocessing (customize as needed)

preprocessed\_data = preprocess\_data(iot\_data)

# Store the preprocessed data (e.g., in an S3 bucket)

store\_preprocessed\_data(preprocessed\_data)

# Return a response if needed

response = {

"statusCode": 200,

"body": "Data loaded and preprocessed successfully"

}

return response

except Exception as e:

# Handle errors and exceptions

error\_message = "Error processing IoT data: " + str(e)

response = {

"statusCode": 500,

"body": json.dumps({"error": error\_message})

}

return response

def preprocess\_data(iot\_data):

# Example preprocessing function

# You can perform data cleaning, transformation, or feature extraction here

# For simplicity, this example just returns the input data as-is

return iot\_data

def store\_preprocessed\_data(data):

# Example function to store data (e.g., in an S3 bucket)

# Replace with the actual code for storing data in your chosen storage service

Pass

**IMPLEMENTATION:**

import json

import boto3

# Initialize AWS clients

s3 = boto3.client('s3')

def lambda\_handler(event, context):

try:

# Extract IoT data from the event

iot\_data = json.loads(event['body'])

# Perform data preprocessing

preprocessed\_data = preprocess\_data(iot\_data)

# Store the preprocessed data in an S3 bucket

store\_preprocessed\_data(preprocessed\_data)

# Return a successful response

response = {

"statusCode": 200,

"body": "Data loaded and preprocessed successfully"

}

return response

except Exception as e:

# Handle errors and exceptions

error\_message = "Error processing IoT data: " + str(e)

response = {

"statusCode": 500,

"body": json.dumps({"error": error\_message})

}

return response

def preprocess\_data(iot\_data):

# Example preprocessing function

# You can perform data cleaning, transformation, or feature extraction here

# For simplicity, this example just returns the input data as-is

return iot\_data

def store\_preprocessed\_data(data):

# Example function to store data in an S3 bucket

bucket\_name = "your-s3-bucket-name"

file\_name = "your-prefix/data.json"

s3.put\_object(Bucket=bucket\_name, Key=file\_name, Body=json.dumps(data))