File Processing System Documentation – Local cloud bases system

Overview

Guide on setting up a local cloud-based file processing system using LocalStack, AWS S3, Lambda, and DynamoDB in Visual Studio Code (VS Code). The system processes CSV files, extracts metadata, and stores the information in DynamoDB.

Prerequisites

- **Docker**
- **LocalStack CLI**
- **AWS CLI**
- **Python 3.9+**
- **VS Code** (With Python extension)
- **Boto3**

System Architecture

1. **S3 Bucket**: Stores CSV files.
2. **S3 Event Notification**: Triggers a Lambda function when a new file is uploaded.
3. **AWS Lambda Function**: Processes the file and extracts metadata.
4. **DynamoDB Table**: Stores metadata of processed files.

Step 1: Setup LocalStack

Start LocalStack using Docker:

```
localstack start -d
```

Verify LocalStack is running:

```
awslocal s3 ls
```

Step 2: Create S3 Bucket

```
awslocal s3api create-bucket --bucket csv-bucket --region us-east-1 --endpoint-url=http://localhost:4566
```

Verify the bucket:

```
awslocal s3api list-buckets --endpoint-url=http://localhost:4566
```

Step 3: Create DynamoDB Table

aws --endpoint-url=http://localhost:4566 dynamodb create-table ` --table-name csv_metadata ` --attribute-definitions AttributeName=filename,AttributeType=S ` --key-schema AttributeName=filename,KeyType=HASH ` --billing-mode PAY_PER_REQUESTVerify table creation:

awslocal dynamodb list-tables --endpoint-url=http://localhost:4566

Step 4: Deploy Lambda Function

1. **Create lambda_function.py in VS Code:**

```
import json
import boto3
import os
import csv
from datetime import datetime

s3_client = boto3.client('s3', endpoint_url="http://localhost:4566")
dynamodb = boto3.resource('dynamodb',
endpoint_url="http://localhost:4566")

TABLE_NAME = "csv_metadata"

def lambda_handler(event, context):
    for record in event['Records']:
        bucket_name = record['s3']['bucket']['name']
        file_key = record['s3']['object']['key']

        # Download file
        temp_file = f"/tmp/{file_key}"
        s3_client.download_file(bucket_name, file_key, temp_file)

        # Extract metadata
        metadata = extract_metadata(temp_file, file_key)

        # Store metadata in DynamoDB
        store_metadata(metadata)

        return {
            "statusCode": 200,
            "body": json.dumps("Metadata extracted successfully")
        }

def extract_metadata(file_path, filename):
```

```
    with open(file_path, newline='', encoding='utf-8') as csvfile:
        reader = csv.reader(csvfile)
        headers = next(reader)
        row_count = sum(1 for _ in reader)

    file_size = os.path.getsize(file_path)
    upload_timestamp = datetime.utcnow().strftime('%Y-%m-%d %H:%M:%S')

    metadata = {
        "filename": filename,
        "upload_timestamp": upload_timestamp,
        "file_size_bytes": file_size,
        "row_count": row_count,
        "column_count": len(headers),
        "column_names": headers
    }

    return metadata

def store_metadata(metadata):
    table = dynamodb.Table(TABLE_NAME)
    table.put_item(Item=metadata)

```

2. **Create a deployment package:**

```
Compress-Archive -Path "C:\Users\Smriti\OneDrive\Desktop\FPS project\lambda_function\*" `
>>    -DestinationPath "C:\Users\Smriti\OneDrive\Desktop\FPS project\lambda.zip" `
```

Deploy Lambda in LocalStack:

```
    aws --endpoint-url=http://localhost:4566 lambda create-function `
    --function-name process_csv `
    --runtime python3.8 `
    --handler lambda_handler.lambda_handler `
    --role arn:aws:iam::000000000000:role/execution_role `
    --zip-file fileb://lambda.zip
```

Verify Lambda Deployment:

```
awslocal lambda list-functions --endpoint-url=http://localhost:4566
```

Step 5: Configure S3 Event Trigger

```
$notificationConfig = @{
  LambdaFunctionConfigurations = @(
    @{
      LambdaFunctionArn = "arn:aws:lambda:us-east-
1:000000000000:function:process_csv"
      Events = @("s3:ObjectCreated:*")
    }
  )
} | ConvertTo-Json -Depth 3

# Save JSON to a file **without BOM**
$jsonFile = "notification-config.json"
[System.Text.Encoding]::UTF8.GetBytes($notificationConfig) | Set-Content -Path
$jsonFile -Encoding Byte

# Use the file in AWS CLI
aws --endpoint-url=http://localhost:4566 s3api put-bucket-notification-configuration `
  --bucket csv-bucket `
  --notification-configuration file://$jsonFile
```

Step 6: Upload and Process a File

1. **Create a Sample CSV File (data.csv):**

   ```
   id,name,age,city,date
   1,John,30,New York,2025-01-14
   2,Jane,25,Los Angeles,2025-02-14
   3,Smriti,26,New Delhi,2025-03-09
   ```

2. **Upload File to S3:**

```
awslocal s3 cp data.csv s3://csv-bucket/ --endpoint-url=http://localhost:4566
```

3. **Verify Processing in DynamoDB:**

```
awslocal dynamodb scan --table-name FileMetadata --endpoint-
url=http://localhost:4566
```