МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Навчально-науковий інститут фізико-технічних та комп’ютерних наук

Кафедра комп’ютерних програмного забезпечення комп’ютерних систем

**ЗВІТ  
про виконання лабораторної роботи № 5**

**з дисципліни**

**«Безсерверні обчислення у хмарних середовищах»**

**на тему: «Використання AWS SQS з Serverless Framework»**

Виконав

студент 5-го курсу

543 групи

Борсук Андрій Юрійович

П.І.Б

Чернівці, 2025

1. **За основу було взято лабораторну роботу №5**
2. **Підняття брокера**

Для коректної роботи serverless sqs нам потрібен брокер, який буде отримувати повідомлення та надсилати їх нашому застосунку. Було обрано брокер ElasticMQ. Для налаштування брокера було створено docker compose файл:

version: '3.8'

services:

  elasticmq:

    image: softwaremill/elasticmq:latest

    container\_name: elasticmq

    ports:

      - "9324:9324"   # REST interface (SQS-compatible)

      - "9325:9325"   # Web UI (if using custom UI configuration)

Та піднято за допомогою команди:

*docker compose up -d*

1. **Налаштування проєкту**

Для підтримки проєктом sqs було встановлено пакет serverless-offline-sqs:

*npm install serverless-offline-sqs*

Також було змінено конфігурацію самого serverless, де ми вказали, що замість http gateway ми будемо використовувати певні повідомлення для виконання тієї чи іншої команди. Зараз конфігурація має наступний вигляд:

service: serverless-lab

frameworkVersion: "3"

provider:

  name: aws

  runtime: nodejs18.x

plugins:

  - serverless-offline

  - serverless-offline-sqs

  - serverless-dynamodb

custom:

  serverless-dynamodb:

    stages:

      - dev

    start:

      port: 8000

      region: eu-west-1

      inMemory: true

      migrate: true

  serverless-offline-sqs:

    autoCreate: true

    apiVersion: '2012-11-05'

    endpoint: http://localhost:9324

    region: us-east-1

    accessKeyId: root

    secretAccessKey: root

    skipCacheInvalidation: false

resources:

  Resources:

    usersTable:

      Type: AWS::DynamoDB::Table

      Properties:

        TableName: Users

        AttributeDefinitions:

          - AttributeName: userId

            AttributeType: S

          - AttributeName: email

            AttributeType: S

        KeySchema:

          - AttributeName: userId

            KeyType: HASH

        GlobalSecondaryIndexes:

          - IndexName: email-index

            KeySchema:

              - AttributeName: email

                KeyType: HASH

            Projection:

              ProjectionType: ALL

    organizationsTable:

      Type: AWS::DynamoDB::Table

      Properties:

        TableName: Organizations

        AttributeDefinitions:

          - AttributeName: organizationId

            AttributeType: S

          - AttributeName: name

            AttributeType: S

        KeySchema:

          - AttributeName: organizationId

            KeyType: HASH

        GlobalSecondaryIndexes:

          - IndexName: name-index

            KeySchema:

              - AttributeName: name

                KeyType: HASH

            Projection:

              ProjectionType: ALL

    CreateOrganizationQueue:

      Type: AWS::SQS::Queue

      Properties:

        QueueName: create-organization-queue

    CreateUserQueue:

      Type: AWS::SQS::Queue

      Properties:

        QueueName: create-user-queue

    UpdateOrganizationQueue:

      Type: AWS::SQS::Queue

      Properties:

        QueueName: update-organization-queue

    UpdateUserQueue:

      Type: AWS::SQS::Queue

      Properties:

        QueueName: update-user-queue

functions:

  sqsHandler:

    handler: handler.sqsHandler

    events:

      - sqs:

          arn:

            Fn::GetAtt: [CreateOrganizationQueue, Arn]

      - sqs:

          arn:

            Fn::GetAtt: [CreateUserQueue, Arn]

      - sqs:

          arn:

            Fn::GetAtt: [UpdateOrganizationQueue, Arn]

      - sqs:

          arn:

            Fn::GetAtt: [UpdateUserQueue, Arn]

Трохи змінено було й код, де тепер експортується не кожен метод як окремий хендлер, а спільний хендлер, який по типу івента викликає необхідний метод:

module.exports.sqsHandler = async (event) => {

  try {

    for (const record of event.Records) {

      const { eventType, ...payload } = JSON.parse(record.body);

      switch (eventType) {

        case 'createOrganization':

          await createOrganization(payload);

          break;

        case 'createUser':

          await createUser(payload);

          break;

        case 'updateOrganization':

          await updateOrganization(payload);

          break;

        case 'updateUser':

          await updateUser(payload);

          break;

        default:

          console.warn('Unknown eventType:', eventType);

      }

    }

    return { statusCode: 200, body: JSON.stringify({ message: 'Processed' }) };

  } catch (err) {

    throw err;

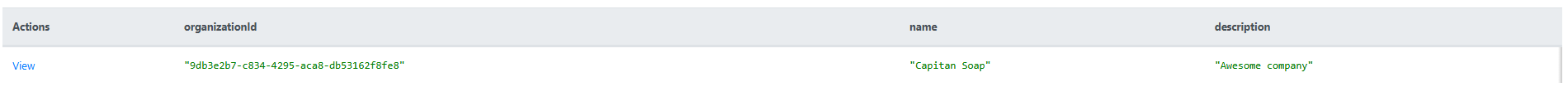
  }

};

1. **Тестування**

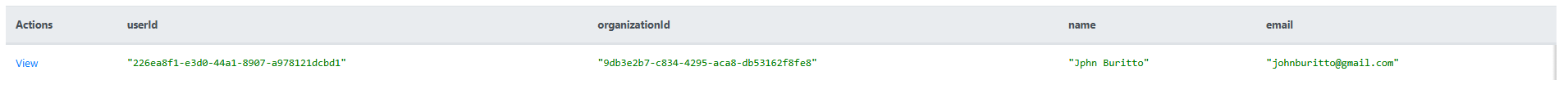
* *Створення організації*

*aws --endpoint-url=http://localhost:9324 sqs send-message --queue-url http://localhost:9324/queue/create-organization-queue --message-body '{"eventType":"createOrganization","name":"Capitan Soap","description":"Awesome company"}'*



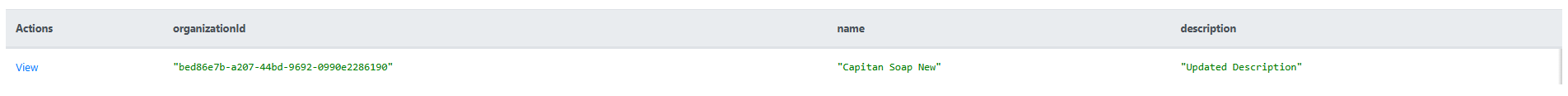
* *Створення користувача*

*aws --endpoint-url=http://localhost:9324 sqs send-message --queue-url http://localhost:9324/queue/create-user-queue --message-body '{"eventType":"createUser","organizationId":"9db3e2b7-c834-4295-aca8-db53162f8fe8","name":"Jphn Buritto","email":"johnburitto@gmail.com"}'*



* *Оновлення організації*

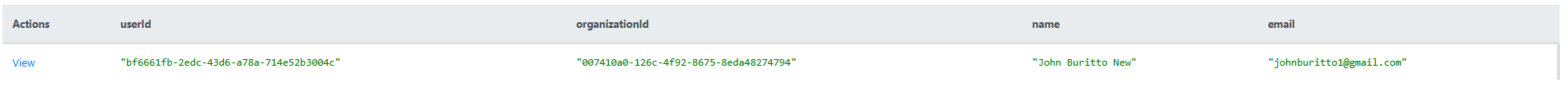
*aws --endpoint-url=http://localhost:9324 sqs send-message --queue-url http://localhost:9324/queue/update-organization-queue --message-body '{"eventType":"updateOrganization","organizationId":"bed86e7b-a207-44bd-9692-0990e2286190","name":"Capitan Soap New","description":"Updated Description"}'*



* *Оновлення користувача*

*aws --endpoint-url=http://localhost:9324 sqs send-message --queue-url http://localhost:9324/queue/update-user-queue --message-body '{"eventType":"updateUser","userId":"bf6661fb-2edc-43d6-a78a-714e52b3004c",*

*"organizationId":"007410a0-126c-4f92-8675-8eda48274794","name":"John Buritto New","email":"johnburitto1@gmail.com"}'*



**Додатки**

Код проєкту – <https://github.com/johnburitto/serverless-lab5/tree/lab-6>