возвращающие следующие данные: начальное состояние формы, схему проверки данных на корректность с помощью Yup, разметку формы на JSX. Поскольку любой компонент в пользовательском интерфейсе на React должен возвращать функцию, то была возвращена функция с разметкой формы. Она будет вызвана, когда компонент авторизации будет встречен компилятором в процессе компиляции. После разработки визуальной части была определена функция отправки формы на сервер. Для этого использовался RTK Query и определён в заранее созданном для запросов на сервере файле интерфейс функций авторизации и регистрации, а также соответствующие URL-адреса роутера и специального контроллера. Далее эти функции были экспортированы в файл с формой и их выполнение было связано с действием нажатия на кнопку подтверждения отправки данных на сервер. Наконец, правильность функционирования описанного процесса и выполнения всех намеченных модулем задач были протестированы.

Заключение. Опыт разработки приложения позволил сделать следующие выводы. Строгое следование архитектурным концепциям не всегда может положительно влиять на результат. Некоторые проекты требуют уникальных решений в процессе самой разработки и безоговорочное следование тем или иным концепциям может выступать обоюдоострым мечом на войне за успех проекта. Тем не менее, в данном веб-приложении вышеперечисленные концепции применяются и реализуются в полной мере без какого-либо ущерба.

Литература

1 Документация по React [Электронный ресурс]. – Режим доступа: https://react.dev/reference/react. – Дата доступа: 14.04.2023.

2 Документация по Node.js [Электронный ресурс]. – Режим доступа: https://nodejs.org/en/docs. – Дата доступа: 14.04.2023.

УДК 004.4’2:004.75

А. В. Скибунов

РЕАЛИЗАЦИЯ ETL-ПРОЦЕССОВ С ИСПОЛЬЗОВАНИЕМ СЕРВИСОВ AWS

Статья посвящена описанию реализации ETL-процессов с использованием сервисов Amazon [1]. Проектирование выполнено в среде Amazon Lambda и Amazon Glue, а разработка – на языке программирования Python с использованием его библиотек и модулей. В некоторых процессах задействовано использование OLAP – базы данных Amazon RedShift и OLTP – базы данных PostgreSQL.

AWS Lambda – это сервис вычислений без сервера, который позволяет запускать код в ответ на события и запросы. AWS Glue – это управляемый сервис ETL, который позволяет создавать и запускать ETL-процессы. Glue автоматически масштабируется, чтобы обрабатывать большие объемы данных и обеспечивает простой интерфейс для создания ETL-процессов.

Преимущества использования AWS Lambda и AWS Glue для ETL-процессов:

1) Масштабируемость. AWS Lambda и AWS Glue могут масштабироваться в зависимости от объема данных, что позволяет обрабатывать большие объемы данных.

2) Управляемые сервисы. AWS Lambda и AWS Glue предоставляют управляемые сервисы, которые обеспечивают надежность и безопасность.

3) Простота использования. AWS Lambda и AWS Glue предоставляют простой интерфейс для создания ETL-процессов, что позволяет быстро создавать и обновлять процессы.

4) Интеграция с другими сервисами AWS. AWS Lambda и AWS Glue интегрируются с другими сервисами AWS, такими как Amazon S3, Amazon Redshift, Amazon RDS и другими.

Давайте рассмотрим пример использования AWS Lambda и AWS Glue для создания ETL-процессов. Предположим, что у нас есть набор данных в формате CSV, который мы хотим загрузить в базу данных PostgreSQL. Мы можем использовать AWS Lambda для извлечения данных из CSV-файла и AWS Glue для загрузки данных в базу данных PostgreSQL (рисунок 1).

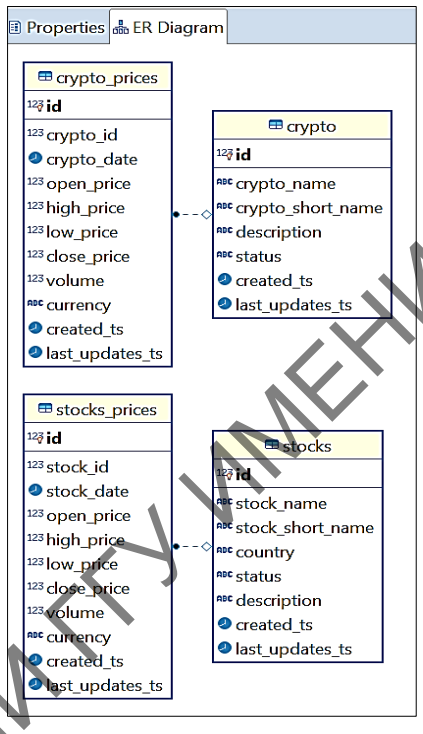

Рисунок 1 – Схема базы данных PostgreSQL

Шаг 1. *Создание функции AWS Lambda для извлечения данных из CSV-файла.*

Мы можем использовать Python и библиотеку Pandas для извлечения данных из CSV-файла. Создадим функцию AWS Lambda, которая будет считывать данные из CSV-файла и возвращать их в формате DataFrame (рисунок 2).

```
import pandas as pd
import boto3
import io

def lambda_handler(event, context):
    s3 = boto3.client('s3')
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']
    obj = s3.get_object(Bucket=bucket, Key=key)
    df = pd.read_csv(io.BytesIO(obj['Body'].read()))
    return df
```

Рисунок 2 – Реализация функции AWS Lambda

Шаг 2. *Создание ETL-процесса в AWS Glue.*

Мы можем использовать AWS Glue для создания ETL-процесса, который загружает данные из DataFrame в базу данных PostgreSQL. Для этого необходимо выполнить следующую последовательность действий:

1) Создайте новый каталог для хранения таблицы в AWS Glue.
2) Создайте новую базу данных в AWS Glue.
3) Создайте новый ETL-процесс в AWS Glue и добавьте источник данных (DataFrame из AWS Lambda) и целевую базу данных PostgreSQL.
4) Настройте преобразование данных (трансформацию) для соответствия структуры данных источника данных и целевой базы данных.
5) Запустите ETL-процесс и проверьте результаты.

Шаг 3. *Проверка результатов.*

После выполнения ETL-процесса мы можем проверить результаты, чтобы убедиться, что данные были успешно загружены в базу данных PostgreSQL.

Шаг 4. *Планирование задач.*

Итак, последний шаг к автоматизации нашего приложения – это, конечно же, планирование наших задач на ежедневном уровне. Запланируем запуск наших задач с 9 утра и отмену в 9:40 утра по белорусскому времени (рисунок 3).

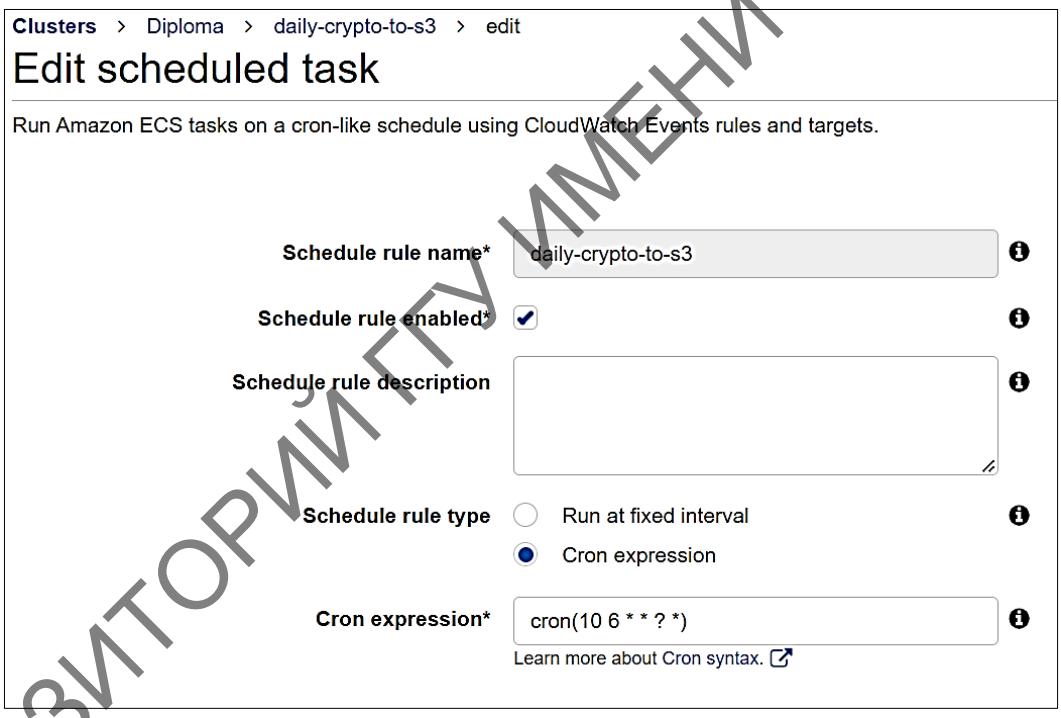

Рисунок 3 – Расписание задачи

AWS Lambda и AWS Glue предоставляют мощные инструменты для создания ETL-процессов. Их простота использования, масштабируемость и управляемость делают их идеальным выбором для предприятий любого размера.

Литература

1 AWS [Electronic resource]. – Mode of access: https://aws.amazon.com/. – Date of access: 01.03.2023.