aufgabe-02.md

Aufgabe 2 - Lambda Trading Notifications

Scenario

Du als Cloud Experte wurdest von einer Finanzfirma beauftragt eine cloudbasierte Loesung zu entwickeln, die das Unternehmen noch agiler machen soll. Es geht um eine Anwendung, die Trader benachrichtigt, wenn gerade gute Marktbedingungen vorhanden sind. Um das System auch in Zukunft noch gut betreiben zu koennen, gibt es folgende Anforderungen:

Anforderungen

- Cloudbasiert(AWS)
- Kein Wartungsaufwand
- Skalierbarkeit / Scalability
- Hochverfuegbarkeit / High Availability

Konzept

Nach ausgiebigen Tests, bist du und dein Team zu dem Entschluss gekommen, einen AWS Serverless Stack zu verwenden um den Wartungsaufwand so gering wie Moeglich zu halten.

!(AWS Infrastruktur)[./stock-infra.png]

Es wird also in regelmaesigen Abstaenden immer wieder von `EventBridge` durch einen Trigger eine `Lambda Function` ausgeloest. Diese ruft dann durch eine API den aktuellen Aktien Preis ab, speichert ihn in die Datenbank und fragt die letzten Werte ab und berechnet ob die Preislage aktuell gut oder schlecht ist. Bei einem positiven Ergebnis, wird durch SNS eine E-Mail versendet.

AWS Ressourcen

SNS

Erstelle ein SNS Topic mit den folgenden Settings:

- Type: `Standard`
- Name: `StockNotifications`
- Dispaly name: `Stock update`

Fuege eine Subscription fuer deine E-Mail Adresse hinzu.

Notiere die Topic ARN oder lasse diesen Tab offen

DynamoDB

Erstelle einen DynamoDb Table mit den folgenden Settings:

- Table name: `StockHistoryTable`
- Partition key: `timestamp`
- Table settings: `Customize settings`
- Table class: `DynamoDb Standard`
- Read/write capacity mode: `On-demand`
- Keine Secondary Indexes

IAM Policy

Erstelle eine IAM Policy mit den folgenden Einstellungen:

- Name: `StockLambdaPolicy`
- Description: `Role for Lambda with access to sns and dynamodb`
- Benutze folgendes Policy Document **ERSETZE ABER DIE SNS TOPIC UND DYNAMODB TABLE ARN**

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:eu-central-1:698449077925:StockNotification"
        },
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:Scan",
                "dynamodb:DescribeTable",
                "dynamodb:Query"
            ],
            "Resource": "arn:aws:dynamodb:eu-central-1:698449077925:table/StockHistory"
        }
    ]
}
```

IAM Rolle

Erstelle eine IAM Rolle, die von der Lambda Function verwendet wird, benutze dabei folgende Einstellungen:

- Trusted entity type: `AWS Service`
- Use case (bei Common use cases): `Lambda`
- Permissions policies: `StockLambdaPolicy`
- Role name: `StockLambdaRole`

Lambda Function

Erstelle eine Lambda Function mit den folgenden Eigenschaften: **Achte darauf nach jeder Aenderung am Code, die Funktion neu zu deployen!**

- Name: `StockLambdaFunction`
- Runtime: `Python 3.10`
- Architecture: `x86_64`
- Execution role: `Use existing role` -> `StockLambdaRole`

Lambda Environment

Nach dem Erstellen, aendere die Environment Variablen: `Configuration` -> `Environment variables` -> `Edit`

1. SNS Topic

- Key: `SNS_TOPIC_ARN`
- Value: Deine SNS Topic ARN

2. DynamoDB ARN

- Key: `DYNAMODB_ARN`
- Value: Deine DynamoDB ARN

Mit diesem Code kannst du die Environment Variablen testen: Siehe file in python/01-env-test.py

SNS Test

Fuege den Code im File python/02-sns-test.py in deine Lambda Funktion ein und deploye erneut. Wurde eine E-Mail and deine registrierte Adresse gesendet?

DynamoDB Test

Fuege den Code im File python/03-dynamodb-test.py in deine Lambda Funktion ein und deploye erneut. Wird ein Eintrag im Table vorgenommen?

API Test

Fuege den Code im File python/04-api-test.py in deine Lambda Funktion ein und deploye erneut. Wird der Bitcoin Preis richtig angezeigt?

The stage is yours!

Nun liegt es an dir die Lambda Function so zu implementieren, dass alle Bediungen erfuellt sind. Die folgenden Funktionen sollten dir helfen, muessen aber noch ausformuliert werden:

```
def get_stock_price():

def save_stock_price(stock_price):

def get_all_stock_prices():

def is_good_price(all_stock_prices):

def sns_alert(data):

stock_price = get_stock_price()
save_stock_price(stock_price)
all_stock_prices = get_all_stock_prices()
if is_good_price(all_stock_prices):
    sns_alert(stock_price)
```