**Document Title**: Lambda Setup Document

**Project Title**: Hatching Sparrow CRM

**Date**: 6/30/2024

**Author(s)**: Ryan O’Connor, Full Stack Software Engineer

**Confidentiality:** N/A

### **Setting Up and Writing Your First AWS Lambda Function (Node.js)**

#### **1. Generate Access Keys for the Root User**

1. **Sign in to the AWS Management Console:**
   * Log in using your root account credentials.
2. **Navigate to the IAM Console:**
   * Go to the IAM console by clicking on the services menu and selecting "IAM".
3. **Access the Security Credentials Tab:**
   * Click on your account name at the top right of the page and select "My Security Credentials".
   * Expand the "Access keys (access key ID and secret access key)" section.
4. **Create a New Access Key:**
   * Click on "Create New Access Key".
   * Download the key file or copy the access key ID and secret access key to a secure location.

#### **2. Install and Configure AWS CLI**

**Install AWS CLI:**

* If you haven't installed the AWS CLI yet, download and install it from the [official AWS CLI installation guide](https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html).

**Configure AWS CLI:**

1. **Open a terminal or command prompt.**
2. **Run the AWS configure command:**  
   aws configure
3. **Enter your access key and secret key:**
   * You will be prompted to enter the following details:
     + **AWS Access Key ID:** Enter the access key ID you generated.
     + **AWS Secret Access Key:** Enter the secret access key you generated.
     + **Default region name:** Enter your preferred region (e.g., us-east-1).
     + **Default output format:** Enter your preferred output format (e.g., json).

**Example:**

$ aws configure

AWS Access Key ID [None]: YOUR\_ACCESS\_KEY\_ID

AWS Secret Access Key [None]: YOUR\_SECRET\_ACCESS\_KEY

Default region name [None]: us-east-1

Default output format [None]: json

#### **3. Install AWS SAM CLI**

The AWS Serverless Application Model (SAM) CLI is a tool for building, testing, and deploying serverless applications. It simplifies the development process by providing a local environment to build and test AWS Lambda functions.

**Install AWS SAM CLI:**

Follow the instructions to install the AWS SAM CLI from the [official SAM CLI installation guide](https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/install-sam-cli.html).

#### **Initialize a New AWS SAM Project by running the following command:** sam init

**Prompts Explanation:**

* + **Template source:** Choose 1 (AWS Quick Start Templates).
  + **Runtime:** Choose 1 (nodejs14.x or the latest Node.js runtime).
  + **Project name:** Enter a name for your project, e.g., my-first-lambda.
  + **Package type:** Choose Zip (for simpler deployment).

**Note:** SAM may prompt you to install Docker. Docker is required to simulate the AWS Lambda environment on your local machine. It allows you to build and test your functions locally in a way that mimics the actual Lambda environment.

#### **5. Write Your First Lambda Function**

Navigate to the project directory and open the hello-world directory (or similar, based on the project structure created).

Edit the app.js file to define your Lambda function:

exports.lambdaHandler = async (event, context) => {

try {

const response = {

statusCode: 200,

body: JSON.stringify({

message: 'Hello from Lambda!'

}),

};

return response;

} catch (err) {

console.log(err);

return err;

}

};

#### **6. Test Your Lambda Function Locally**

In the project root directory, run the following command to start the local API using AWS SAM:

sam local start-api

This will start a local API on http://127.0.0.1:3000. You can invoke your Lambda function by sending an HTTP request to this endpoint.

Test the function using curl or any API testing tool (like Postman):

curl http://127.0.0.1:3000/hello

You should see a response similar to:

{

"message": "Hello from Lambda!"

}

**7. Deploy Your Lambda Function**

When you're ready to deploy your Lambda function to AWS, use the following commands:

sh

Copy code

sam build

sam deploy --guided

**Prompts Explanation:**

* **Stack Name:** Enter a stack name (e.g., my-first-lambda-stack). A stack is a collection of AWS resources that you can manage as a single unit.
* **AWS Region:** Enter the region to deploy your stack (e.g., us-east-1).
* **Confirm changes before deploy:** Choose whether to review changes before deployment.
* **Allow SAM CLI IAM role creation:** Allow the SAM CLI to create IAM roles with the necessary permissions.
* **Save arguments to configuration file:** Choose whether to save these options to a configuration file for future deployments.

Once completed, your Lambda function will be deployed to AWS and accessible via the provided API Gateway endpoint.

#### **8. Link Your Lambda Function to Frontend and DynamoDB**

* **API Gateway Endpoint:** Use the API Gateway endpoint returned after deployment to configure your React.js frontend to send requests to this endpoint.
* **DynamoDB Integration:** Update your Lambda function to interact with DynamoDB. You can add AWS SDK calls within your Lambda function to read and write data to DynamoDB.
* **Authentication Setup:** For setting up authentication, you can create additional Lambda functions for user registration and login, which will generate and verify JWT tokens for user sessions.