# Building a Serverless Application Using Step Functions, API Gateway, Lambda, and S3 in AWS

# **Description**

# In this AWS hands-on lab, we will create a fully working serverless reminder application using S3, Lambda, API Gateway, Step Functions, Simple Email Service, and Simple Notification Service.

# While the lab does use Python and JavaScript, you don't need to be able to code to understand and implement the solution. By the end of the lab, you will feel more comfortable architecting and implementing serverless solutions within AWS.

# Let's get started!

# **Objectives**

# Successfully complete this lab by achieving the following learning objectives:

# ***Create the email Lambda Function***

# Open the Simple Email Service console.

# Click Create identity, Email address

# Click the link in the email you received to verify your email address.

# Refresh, and make sure the verification state is *verified*.

# Go to the Lambda console.

# Create a function.

# Choose **Author from scratch**, name the function **email**, and the runtime can be either **python 3.6** or **python 3.7**.

# Expand **Choose or create an execution role**, select **Use an existing role**, and select the existing role from the dropdown, called **LambdaRuntimeRole**.

# Click **Create function**, and scroll down to the function code box.

# Replace all the code with the code from email\_reminder.py.

# Change YOUR\_SES\_VERIFIED\_EMAIL to the email address you verified earlier.

# Click **Save**.

# ***Create the sms Lambda Function***

# From the Lambda console:

# Create a function.

# Make sure **Author from scratch** is selected, call it **sms**, and select the **python 3.6** or **python 3.7** runtime.

# Expand **Choose or create an execution role**, pick **Use an existing role**, and select **LambdaRuntimeRole**.

# Click **Create Function**.

# Scroll down to the function code box.

# Delete the existing code, and replace it with the contents of sms\_reminder.py from GitHub.

# Click **Save**.

# ***Create a Step Function State Machine***

# Navigate to the Step Functions console.

# Delete any existing state machines.

# Note any functions that don't delete. You need to pick a unique name.

# Click **Create state machine**.

# Select **Author with code snippets**.

# For name, enter **reminder** (ideally) or **reminder-** with a unique string on the end if **reminder** is already taken.

# Delete the existing snippet, and replace it with the contents of step-function-template.json from GitHub.

# You may see red error indicators, and those are fine.

# Replace any occurrences of EMAIL\_REMINDER\_ARN with the ARN of the email Lambda function.

# Replace any occurrences of TEXT\_REMINDER\_ARN with the ARN of the sms Lambda function.

# Click **Refresh** next to the diagram, and ensure it updates to include the flow of steps.

# Click **Next**, select **Choose an existing IAM role**, select **I will use an existing role**, and select **RoleForStepFunction**.

# Click **Create state machine**.

# ***Create the api\_handler Lambda Function***

# From the Lambda console:

# Create function.

# Set **Author from scratch**.

# Set the function name to **api\_handler** and the runtime to **python 3.6** or **python 3.7**.

# Expand **Choose or create an execution role**, select **Use an existing role**, and select **LambdaRuntimeRole**.

# Click **Create function**.

# Scroll down to the function code box, delete the existing code, and replace it with the contents of api\_handler.py from GitHub.

# At the top of the code, replace STEP\_FUNCTION\_ARN with the ARN of the state machine you just created.

# Ensure you pick the state machine ARN, *not* the IAM role ARN.

# Click **Save**.

# ***Create the API Gateway***

# From the API Gateway endpoint:

# Click **Get Started** and **OK** on any example dialog box.

# Select **REST** and **New API**.

# Set the API name to **reminders**, and configure it to be a regional API.

# Click **Create API**.

# Click **Actions** and **Create Resource**.

# Call it **reminders**, and enable **Enable API Gateway CORS**.

# Click **Create Resource**.

# Select **/reminders**, and click **Actions** > **Create method**.

# Select **POST** in the dropdown.

# Click the check mark next to *POST*.

# Select **POST**, ensure **Lambda Function** is selected and *Use Lambda Proxy Integration* is checked.

# In the Lambda function box, type **api\_handler** and click **Save**.

# Click **OK** to allow API Gateway permission to invoke the api\_handler function.

# Select **Actions** and **Deploy API**.

# Set *Deployment stage* to **New Stage**.

# Type **prod** for name and description.

# Click **Deploy**.

# You can ignore any errors involving WAF. They don't impact the lab.

# Take note of the **Invoke URL**. You will need this next.

# ***Create the Static S3 Website***

# Locate the Local *static\_website* Folder

# Open *formlogic.js*.

# Replace the YOUR\_API\_ENDPOINT\_URL placeholder with the invoke URL for the *prod* API with **/reminders** appended.

# ***Configure the Static Website***

# Open the S3 console.

# Click **Create bucket**.

# The bucket name will need to be unique, and ensure the region is **US East (N. Virginia)**.

# Click **Next**.

# Click **Next** again.

# Ensure *no* checkmarks are selected on the block public access screen. This includes individual ones and any group check mark.

# Click **Next** and then **Create bucket**.

# Open the bucket.

# Click **Upload**, select the objects from the static\_website folder, and click **Next**.

# Change the **Manage public permissions** dropdown to **Grant public read access to this object(s)**.

# Click **Next**, **Next** again, and **Upload**.

# If you get any errors here, verify the files are public by clicking one, selecting **Permissions**, and ensuring public access is set so everyone has read object permissions.

# Head back to your bucket, and click the **Properties** tab.

# Click **Static website hosting**.

# Select **Use this bucket to host a website**.

# Enter index.html in the Index Document box, and error.html in the Error Document box.

# Click **Save**.

# Browse to the Static Website Hosting Endpoint URL for the bucket (the link is on the static website hosting option screen).

# Test it out.