module_2.md

Module 2: API-based solution

Resources

- AWS Solutions Library: AI-Powered Health Data Masking
- Deployment Guide: HTML, PDF
- Source code: GitHub: awslabs/ai-powered-health-data-masking

Process

Step 1: Launch CloudFormation Stack

- Ensure you have launched the CloudFormation stack as described in Module 1, Step 1.
- From the CloudFormation console, open the **Outputs** tab of your stack and note the value of **ApiGatewayId**

Step 2: Create an IAM Policy

- Reference: Implementation Guide, Step 2
- Open the AWS IAM console
- Select **Policies** then **Create Policy**
- Select the **JSON** tab and replace the policy contents with the text copied from below. Note, if you are having formatting problems with copy/paste, this text is in the file `iam/api-policy.txt` in the downloaded course materials.
  - Replace `ACCOUNTID` with your 12-digit account ID
  - Replace `APIGATEWAYID` with the API Gateway ID from above
  - Optionally, replace **us-east-1** with your current region, and prod with your staging environment name, if modified

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "execute-api:Invoke",
        "apigateway:PUT",
        "apigateway:POST",
        "apigateway:GET"
      ],
      "Resource": [
        "arn:aws:execute-api:us-east-1:ACCOUNTID:APIGATEWAYID/prod/*",
        "arn:aws:apigateway:us-east-1::/restapis/APIGATEWAYID/resources/*"
      ]
    }
  ]
}
```

- Click **Review Policy**
- Give your policy a memorable name such as **workshop-api-policy** then click **Create policy**

Step 3: Attach the policy to the SageMaker role

- In the IAM console, select **Roles** and locate the customer-managed role for SageMaker created in Module 1, Step 3 (it will start with **AmazonSageMaker-ExecutionRole-**)
- Select **Attach policies** and select the policy just created in Step 2, above
- Click **Attach policy** and verify that the role now has 5 policies attached

Step 4: Test the API

- Reference: Implementation Guide, Appendix B
- Return to the Jupyter notebook of the SageMaker instance from Solution 1
- Ensure you have uploaded the files `api_imagemask.py` and `api_textmask.py` as described in Module 1, Step 5.
- We will be editing these files; either select the file in Jupyter and click 'Edit', or else use **vi** or **emacs** in a terminal window
- Open a new Terminal by selecting **New** → **Terminal**
- In the terminal, run **bash**, and **cd** to the `SageMaker` directory

Text Masking

- Edit the file `api_textmask.py` , entering the values from the CloudFormation outputs:
  - `api_id` : **ApiGatewayId**
  - `resource_id` : **TextMaskResourceId**
- In the Terminal tab, run the code: `python api_textmask.py`
- Notes:
  - Try adjusting the value of **phiDetectionThreshold**
  - This calls the **POST** method on the `/text/mask` resource, which invokes the Lambda method in `/mask_text/lambda_function.py`
  - Lambda invocations are logged in CloudWatch Logs log group **/aws/lambda/<stack-name>-<lambda-name>**
  - Lambda logging can be configured with the environment variable **LOG_LEVEL** in the Lambda console (INFO, ERROR etc)
  - API Gateway logging may be configured in **Stages** → **Logs/Tracing**

Image Masking

- Edit the file `api_imagemask.py` , entering the values from the CloudFormation outputs:
  - `api_id` : **ApiGatewayId**
  - `resource_id` : **ImageMaskResourceId**
  - `s3_bucket` : Bucket containing your image
  - `s3_key` : Name of image file
- In the Terminal tab, run the code: `python api_imagemask.py`
- The script will return the S3 URL of the masked image file, which you can download:
  - From the S3 console
  - From your local command line: `aws s3 cp s3://YOUR-IMAGE-BUCKET/masked/chest.jpg .`
- Notes:
  - Try adjusting the value of **phiDetectionThreshold**
  - This calls the **POST** method on the `/image/mask` resource, which invokes the Lambda method in `/mask_image/lambda_function.py`