CSCI 4145/5409 Summer 2024
Serverless Assignment

Important note

Carefully reading the assignment description is important, as even small oversights could lead to zero marks for your assignments. Take your time to scrutinize every word and ensure you comprehend all the requirements. If you have any doubts, don't hesitate to ask questions in the Teams channel dedicated to this assignment.

Reminder

This is an individual assignment. You are not allowed to collaborate with anyone else when completing this assignment. You can borrow code and configuration snippets from internet sources that are not from students in this class, however that code must be cited and include comments for how you have modified the original code and does not count as code you have written.

While I've mentioned that you're allowed to use ChatGPT or similar tools for your term project, I strongly advise coding the assignments independently. These tasks are simple enough for you to complete without assistance. If you rely on tools to generate code for these relatively straightforward programming assignments, it could impede your ability to qualify for entry level developer positions in the industry. This is an opportunity for you to practice and improve your skills. However, if you still choose to use ChatGPT or similar tools for assignments, that's your decision and acceptable.

I will not accept negotiations regarding marks lost due to your errors.

Introduction

This assignment will measure your understanding of some of the serverless mechanisms of our cloud provider AWS. This assignment assures us that you have attended the tutorials and learned about AWS Lambda and Step Functions, or that you have found some other way to learn these services. In addition, you will have to do some self-learning to study how to use AWS API Gateway to turn your Lambda's and step functions into REST APIs.

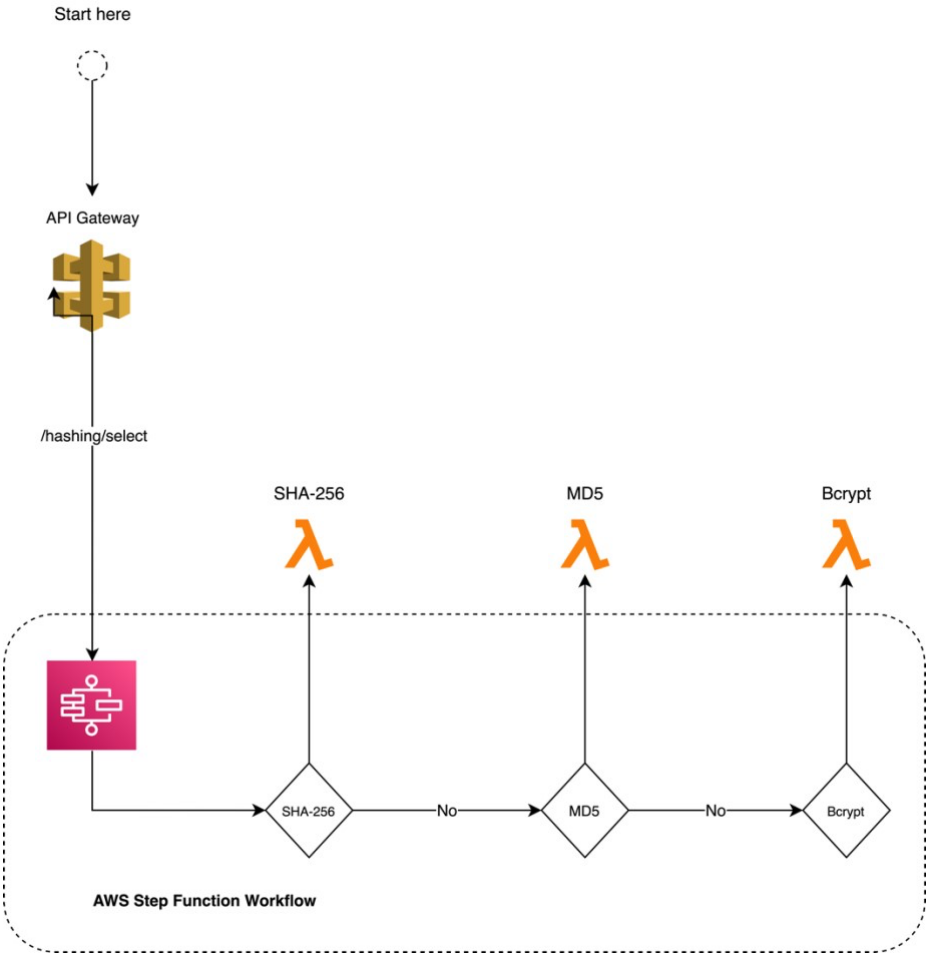
Learning Outcomes

- Learn the benefits of serverless computing and apply that learning to implement a finite state machine using AWS Step Functions and serverless compute mechanisms (Lambda).
- Learn AWS API Gateway to understand how to build serverless APIs in AWS.
- Learn a little bit about common encryption algorithms

Requirements

You will build REST API entry points using serverless compute mechanisms. In this mechanism, you will create a State Machine, configured with API gateway, which will evaluate a choice based on an input, and based on this choice, you will invoke a Lambda function, which will perform a hashing operation on a provided input data.

Here is a rough state diagram for your system:



Endpoint for your state machine - /hashing/select

When performing hashing, explicitly use UTF-8 encoding for all algorithms. For Bcrypt algorithm, use cost factor 12 and make sure the Bcrypt version is '$2b$'.

Your application will "introduce" itself to mine by sending a POST request with some JSON to a URL I provide to you that begins a chain of events. (Note: You can manually send this POST with a tool like PostMan as well).

1. My app will POST to your app's AWS Step Function's endpoint /hashing/select route with some JSON.
2. Your State Machine will process the JSON input and select an option based on this input.
3. Based on this option, you will trigger a Lambda function to perform hashing.
4. Your Lambda Function will perform a specific hashing operation and trigger a POST request with the result to a different endpoint of my app /serverless/end
5. You can query the grade of your recent test submission by calling a POST request to /serverless/results

JSON you will send to my App (/serverless/start)

```
{
        "banner": "<Your Banner ID>",
        "aws_access_key_id": "<Replace with your AWS account's access key id>",
        "aws_secret_access_key": "<Replace with your AWS account's secret access key>",
        "aws_session_token_id": "<Replace with your AWS account's session token>",
        "url_for_step_function": "<Gateway URL>/hashing/select",
        "arn_for_step_function": "<Replace with the ARN of your AWS step Function which has /hashing/select route configured",
        "arn_for_lambda_sha256": "<Replace with the ARN of your Lambda function which performs sha256 hashing",
        "arn_for_lambda_md5": "<Replace with the ARN of your Lambda function which performs md5 hashing",
        "arn_for_lambda_bcrypt": "<Replace with the ARN of your Lambda function which performs bcrypt hashing",
        "api_gateway_id": "<Replace with your API Gateway ID>"
}
```

JSON (POST) you will receive from My App to your Step Function (/hashing/select)

```
{
        "input": "{
                "course_uri": "<URL of My App API>/serverless/end",
                "action": "sha256|md5|bcrypt",
                "value": "<data to perform hash operation on>"
        }",
        "stateMachineArn": "<State Machine ARN that you sent>"
}
```

Please note that the "input" value is a STRING not a JSON object literal and indentation for "input" value is added for explanation purposes. Also, there should not be any body mapping

template configured with your API gateway endpoint and Step Function. Refer to this link to create a Step Functions API using API Gateway.

JSON Request (POST) you will send to My App (/serverless/end)

From Lambda Function which performs SHA-256 hashing

```
{
   "banner": "<Your Banner ID>",
   "result": "<hashed value>",
   "arn": "<ARN of your SHA-256 Lambda function >",
   "action": "sha256",
   "value": "<data to perform hash operation on>"
}
```

From Lambda Function which performs MD5 hashing

```
{
   "banner": "<Your Banner ID>",
   "result": "<hashed value>",
   "arn": "<ARN of your MD5 Lambda function>",
   "action": "md5",
   "value": "<data to perform hash operation on>"
}
```

From Lambda Function which performs Bcrypt hashing

```
{
   "banner": "<Your Banner ID>",
   "result": "<hashed value>",
   "arn": "<ARN of your bcrypt Lambda function >",
   "action": "bcrypt",
   "value": "<data to perform hash operation on>"
}
```

JSON Request (POST) you will send to My App to query results (/serverless/results)

```
{
   "banner": "<Your Banner ID>"
}
```

Note: /serverless/start and /serverless/end will not show the result and feedback

How To Submit

Create a folder in your repository labeled **A3**. Put your lambda source code (no matter what language you wrote it in) in this folder. Also include in the folder either a screenshot of your Step Function implementation or a text export. Push these files to your individual repository on Gitlab **before the assignment deadline.**

Marking Rubric

In this class I'm not very concerned about the quality of the code you write, if you write bad quality code it is you that will suffer in maintaining and supporting it. I care that you can meet the learning objectives defined at the top of this document, and I can verify this by simply verifying the correct behavior of your app's interaction with mine.

Your submission will be marked by the app that I will write, my app will:

- Listen for requests to /serverless/start and /serverless/end and initiate the check process.
- The check process for /serverless/start:
    1. Records the IP and Banner you send to /serverless/start in our database.
    2. Sends POST requests to your IP's /hashing/select API.

The check process for /serverless/end:

1. Retrieves the Banner you previously send to /serverless/start from our database.
2. Validates the output from **SHA-265** Lambda function.
3. Validates the output from **MD5** Lambda function.
4. Validates the output from B**crypt** Lambda function.

You will earn 33.33% per hash test that we execute which you successfully pass by sending the proper post to /end.

You will be marked 0 if we cannot verify any existence of Step functions, Lambda or API Gateway.