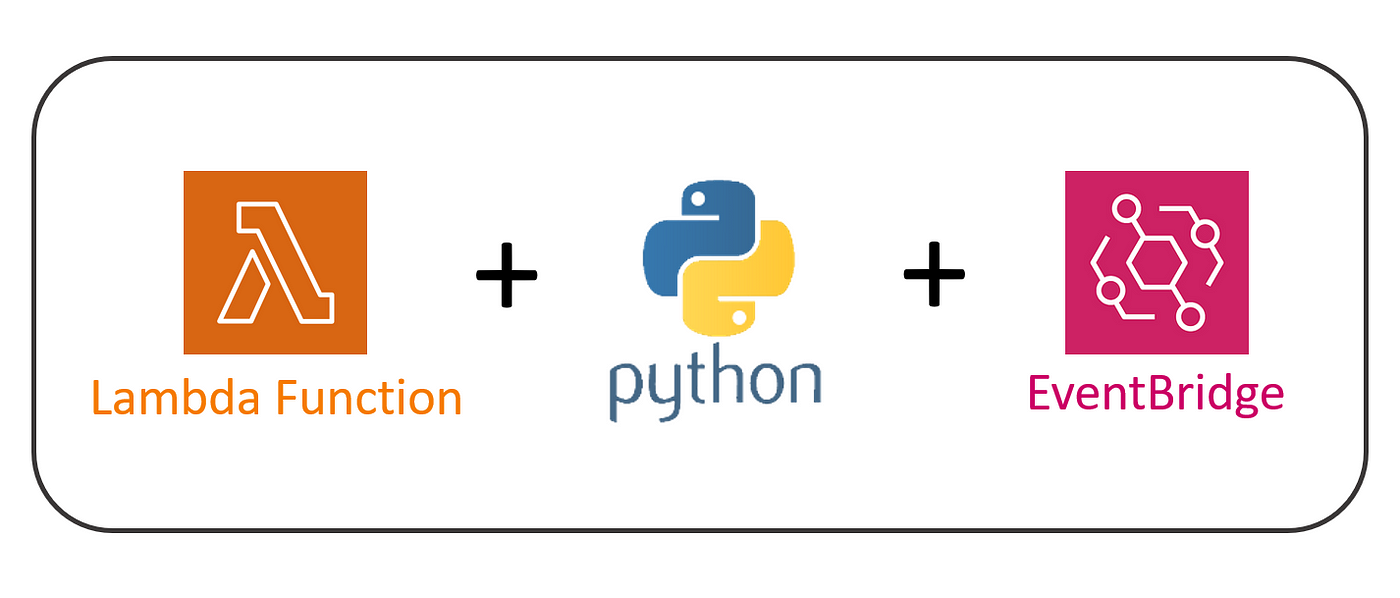
**AUTOMATING AWS WITH PYTHON | PART 5 : EC2 DAILY SNAPSHOTS**



GitHub repository: <https://github.com/Silas-cloudspace/python-for-aws/tree/main/3%20-%20Automating%20AWS%20with%20Lambda/1.%20Lambda_EC2_Daily_Snapshots>

Today we are setting up an AWS infrastructure to automatically create EC2 snapshots using Python, AWS Lambda and EventBridge. The setup involves three main components:

1. **LAMBDA FUNCTION**:

**Purpose**: This function will be triggered to create a snapshot of an EC2 instance. It finds the volume ID of an EC2 instance based on a tag and then creates a snapshot of that volume.

**Code**:

* Imports necessary libraries (boto3, logging, etc.).
* Retrieves the EC2 instance volume ID based on a tag.
* Creates a snapshot of the volume.
* Logs the success or failure of the snapshot creation.

1. **CLOUDFORMATION TEMPLATE**:

**Purpose**: Defines the AWS resources and configurations required for the Lambda function, EC2 instance, EventBridge rule, and necessary permissions.

**Components**:

* **IAM Role (LambdaExecutionRole)**: Grants permissions to the Lambda function to interact with EC2 and CloudWatch Logs.
* **Lambda Function (LambdaEC2DailySnapshot)**: The function created from file 1, which is scheduled to run automatically.
* **EC2 Instance (EC2Instance)**: An example EC2 instance used for snapshot creation. You can update the AMI ID and other properties as needed.
* **EventBridge Rule (DailyEC2SnapshotRule)**: Schedules the Lambda function to run periodically (every 5 minutes, which you can adjust to 24 hours).
* **Lambda Permission (LambdaInvokePermission)**: Allows EventBridge to invoke the Lambda function.

1. **IAM POLICY DOCUMENT**:

**Purpose**: Provides the permissions required by the Lambda function to create and manage EC2 snapshots and interact with CloudWatch Logs.

**Permissions**:

Allows logging actions (logs:CreateLogGroup, logs:CreateLogStream, etc.).

Allows EC2 actions (ec2:CreateSnapshot, ec2:DescribeSnapshots, etc.).

1. **PREREQUISITES**

* **AWS CLI and SAM CLI Installed**:

Ensure that you have the AWS CLI and SAM CLI installed on your local machine. These tools are used for interacting with AWS services and deploying SAM applications.

* **AWS CLI**

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

* **SAM CLI**

<https://github.com/aws/aws-sam-cli/releases>

* **Docker**:

Docker is required for local testing of Lambda functions using SAM CLI. Install Docker and ensure it’s running.

<https://www.docker.com/products/docker-desktop/>

* **AWS Account**:

You need an AWS account to deploy resources. Ensure you have appropriate permissions to create IAM roles, Lambda functions, EC2 instances, and EventBridge rules.

* **IAM Permissions**:

Ensure you have sufficient IAM permissions to deploy and manage the resources defined in your template. This includes permissions for CloudFormation, Lambda, EC2, EventBridge, and IAM.

1. **PREPARING THE ENVIRONMENT**

Create 3 new files on VS Code

* touch lambda\_function.py template.yaml event.json
* Copy the code from the GitHub repository into them

1. **RUNNING THE CODE**

In command prompt run: “sam deploy --guided”