Technical Specifications

# Assumptions:

Following are the assumptions made while developing Payment Information Application.

1. The application receives payment notifications from AWS SNS service.
2. The payment notification message is in json format. Sample format is

{

"paymentId": "12357494",

"paymentMethod": "paypal",

"timeStamp": "13453865",

"transactions": {

"amount": {

"currency": "USD",

"total": "1.00"

},

"description": "This is the payment description."

}

}

1. paymentId attribute is primary key for the dynamodb.
2. To store and lock tfstate file, an s3 bucket named “terraform-state-file-store1838" and dynamodb table named “terraform-state-file-locks" are already configured before provisioning this Payment Information Application.

Terraform modules

The following modules are created in <terraform> directory in github repository “[payment\_application\_notification](https://github.com/umaparvathir/payment_notification_application)” to provision infrastructure.

1. dynamodb – This module contains Terraform scripts dedicated to provisioning the DynamoDB resources.
2. lambda - This module contains terraform scripts to provision two lambda functions.
3. SNS – This module comprises Terraform scripts that handle the provisioning of the SNS (Simple Notification Service) resources.
4. SQS - This module is designed to provide the SQS resources via its Terraform scripts.
5. lambda\_python\_code - This module contains Lambda functions coded in Python 3, used as part of the infrastructure
6. lambda\_python\_code\_tests - This module contains test cases for lambda functions.
7. dev\_env\_tf\_scripts.tfvars – This tfvars file contains variable values shared across multiple modules.

Infrastructure Provisioned List

List of services provisioned through Terraform for the Payment Information Application backend:

1. Established an SNS topic, configured an SNS subscription linked to a Lambda function.
2. Developed Lambda functions, setting essential environment variables.
3. Generated roles and policies, and associated them with the Lambda functions.
4. Produced an SQS Queue, including role and policy definitions with necessary permissions.
5. Implemented an event source mapping connecting SQS to the Lambda function.
6. Created a DynamoDB table with 'paymentID' designated as the hash key.