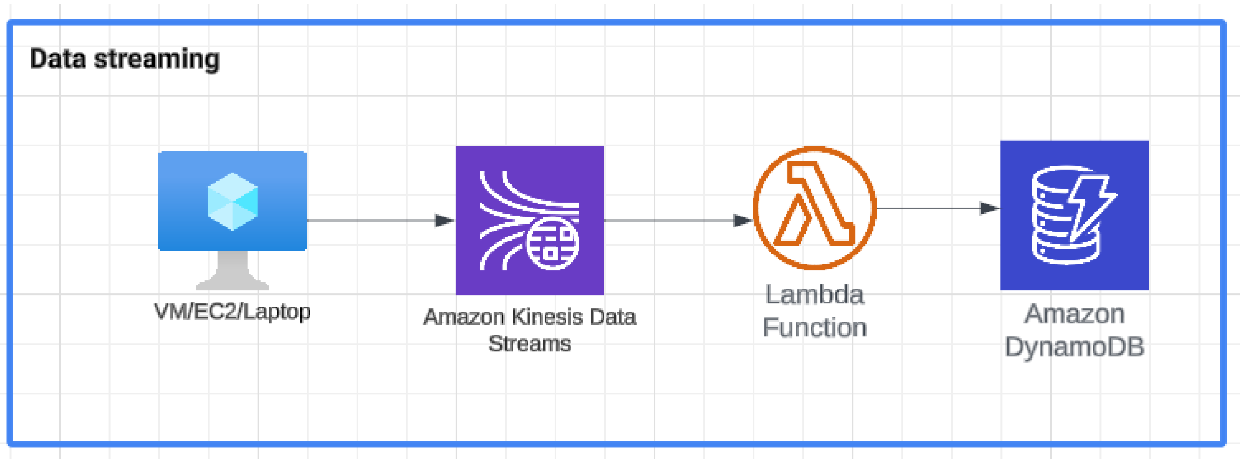
Project 1 - Clickstream (KDS, Lambda, DynamoDB)

Architecture



1. Create Kinesis Data Stream (KDS) named *events*
    a. Capacity mode: Provisioned
    b. Provisioned shards: 1
2. Create a DynamoDb table named *events*
    a. Partition key: pk
3. Create a Lambda function
    a. Function name: event-processor
    b. Runtime: Python (latest version)
    c. Use an existing IAM role - create Lambda role with below policies
        - AmazonKinesisReadOnlyAccess
        - AmazonDynamoDBFullAccess
    d. Code source (see below - intentionally made error on line 11, typ0 -> based instead of base)
    e. Deploy the function
    f. Add trigger
        - Select a source: kinesis
        - Kinesis stream: events
        - Batch size: 10

4. Execute the AWS CLI command to put records into KDS (see command below)
5. Check Lambda trigger
6. Check DynamoDB table for records

Note:

1. After running aws kinesis put-record, if you check the Lambda-> Configure-> Kinesis-> Last processing result, it will show failure
2. To know why we got the failure, if you try looking at CloudWatch logs, you don't see the CloudWatch logs
3. Reason why CloudWatch logs doesn't work is, the Lambda role is missing permissions to publish to CloudWatch
4. Update the Lambda role giving CloudWatch full access
5. Now run the put-record command, and you will see failure again (same as in step1)
6. However, now you can review CloudWatch for errors, you'll notice issue in line 11 of the Lambda code
7. Update the Lambda code typo (based to base) and rerun, everything should run fine.

AWS CLI command to check region you're logged in

```
aws configure get region
```

AWS CLI command to change region

```
aws configure set region us-east-1
```

AWS CLI command to put records into KDS

```
aws kinesis put-record --stream-name events --partition-key "101" --data 'this is
first KDS entry' --cli-binary-format raw-in-base64-out
```

Lambda function code:

```
import json
import boto3
import base64

dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('events')

def lambda_handler(event, context):
   for record in event["Records"]:
       pk = record["kinesis"]["partitionKey"]
```

```
        msg = based64.b64decode(record["kinesis"]["data"])

        table.put_item(Item={
            "pk": pk,
            "data": msg.decode("utf-8")
        })

    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

Additional add on: To mimic clickstream data, you can automate put records (for example, insert 100 records). You can do this on your own laptop or on EC2

```
#!/bin/bash

stream_name="kds-demo"
num_records=100
partition_key=101

i=1
until [ $i -gt $num_records ]; do
  data="this is KDS entry number $i"
  aws kinesis put-record --stream-name $stream_name --partition-key "$partition_key"
--data "$data" --cli-binary-format raw-in-base64-out
  echo "Successfully sent record number $i with partition key $partition_key"
  ((i++))
  ((partition_key++))
done
```

```
Install AWS CLI on EC2 machine

Step-1
Sudo apt install unzip

Step-2
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```