

## Building a Proof of Concept for Data Analytics

In this exercise, you will learn how to do the following:

- Create IAM policies and roles to follow the best practices of working in the AWS Cloud.
- Create the object storage S3 bucket to store clickstream data.
- Create the Lambda function for Amazon Kinesis Data Firehose to transform data.
- Create a Kinesis Data Firehose delivery stream to deliver real-time streaming data to Amazon S3.
- Create a REST API to insert data.
- Create an Amazon Athena table to view the obtained data.
- Create Amazon QuickSight dashboards to visualize data.

### Note:

To complete the instructions in this exercise, choose the US East (N. Virginia) us-east-1 Region in the menu bar the AWS Management Console.

The instructions might prompt you to enter your account ID. Your account ID is a 12-digit account number that appears under your account alias in the top-right corner of the AWS Management Console. When you enter your account number (ID), make sure that you remove hyphens (-).

## Task 1: Setup: Creating the IAM policy and role

In this task, you create a custom IAM policy and role to grant limited permissions to specific AWS services.

### Step 1.1: Creating custom IAM policies

Sign in to the AWS Management Console.

In the search box, enter **IAM** and from the list, choose **IAM**.

In the navigation pane, choose **Policies** and then choose **Create policy**.

The **Create policy** page appears.

In the **JSON** tab, replace the placeholder code by pasting the following policy:

```
{  "Version": "2012-10-17",  "Statement": [    {      "Sid": "VisualEditor0",      "Effect": "Allow",      "Action": "firehose:PutRecord",      "Resource": "*"    }  ]}
```



Choose **Next**.

For the policy name, enter **API-Firehose**.

Choose **Create policy**.

This IAM policy grants permissions to write records to Amazon Kinesis Data Firehose delivery stream.

## Step 1.2: Creating an IAM role and attaching a policy to it

In this step, you create an IAM role that enables API Gateway to send streaming data to Amazon Kinesis Data Firehose. You then add the API-Firehose policy that you created to the role.

In the navigation pane of the IAM dashboard, choose **Roles** and then choose **Create role**.

For **Trusted entity type**, select **AWS service**

In the **Use case** section choose **API Gateway**.

Choose **Next** and then choose **Next** again.

For **Role name**, enter `APIGateway-Firehose`.

Choose **Create role**.

From the roles list, choose the **APIGateway-Firehose** role.

In the **Permissions policies** section, on the **Add permissions** menu, choose **Attach policies**.

From the policies list, select **API-Firehose** and choose **Add permissions**.

In the **Summary** section, copy the **APIGateway-Firehose** ARN and save it for your records. Your APIGateway-Firehose ARN might look similar to the following: `arn:aws:iam::<account ID>:role/APIGateway-Firehose`.

## Task 2: Creating an S3 bucket

In this task, you create an object storage bucket in Amazon S3 to store streaming data in the AWS Cloud.

In the AWS Management Console search box, enter S3 and open the service by choosing **S3**.

Choose **Create bucket**.

For **Bucket name**, enter a unique name.

**Note:** An S3 bucket name must be globally unique. You may name your bucket similar to the following example: `architecting-week2-<your initials>`. Replace `<your initials>` with your own value. Make sure that you also delete the angle brackets (`<>`). For example: `architecting-week2-mw`.

Make sure that **AWS Region** is set to **US East (N. Virginia) us-east-1**

At the bottom of the page, choose **Create bucket**.

Open the bucket details by choosing the name of the bucket that you just created.

Choose the **Properties** tab.

In the **Bucket overview** section, copy your bucket's Amazon Resource Name (ARN) and save it for your records. The ARN of your bucket may look similar to the following: `arn:aws:s3::DOC-EXAMPLE-BUCKET`.

## Task 3: Creating a Lambda function

In this task, you create a Lambda function that transforms data before Amazon Kinesis Data Firehose ingests it into the object storage bucket.

In the AWS Management Console, search for and open the **Lambda** service.

Choose **Create a function**.

Select the **Use a blueprint** card.

In the filter box of the **Blueprints** section, enter `Kinesis`.

In the list of results, you may see two blueprints called **Process records sent to a Kinesis Firehose stream** for Node.js and Python. Select the Python 3.8 blueprint called **Process records sent to a Kinesis Firehose stream** and choose **Configure**.

For **Function name**, enter `transform-data`.

Keep all the other default settings and choose **Create function**.

In the **Code** tab, replace the default code with the following:

```
import json
import boto3
import base64
output = []
def
lambda_handler(event, context):
    for record in event['records']:
```

```

payload = base64.b64decode(record['data']).decode('utf-8')
row_w_newline = payload + "\n"
row_w_newline = base64.b64encode(row_w_newline.encode('utf-8'))
output_record = {
    'recordId': record['recordId'],
    'result': 'Ok',
    'data': row_w_newline
}
output.append(output_record)
return {'records': output}

```

Choose **Deploy**.

Choose the **Configuration** tab.

In the **General configuration** section, choose **Edit** and change the **Timeout** setting to 10 seconds.

Choose **Save**.

In the **Function overview** section, copy the function ARN and save it for your records.

Your ARN might look similar to the following example: `arn:aws:lambda:us-east-1:<account ID>:function:transform-data`.

## Task 4: Creating a Kinesis Data Firehose delivery stream

In this task, you complete two steps. First, you create a Kinesis Data Firehose delivery stream to ingest streaming data. Then, you copy the ARN of the delivery stream's IAM role. Later in the exercise, you use this ARN to connect the delivery stream to your storage bucket.

### Step 4.1: Creating the Kinesis Data Firehose delivery stream

In AWS Management Console, search for and open the **Kinesis** service.

On the **Get started** card, select **Kinesis Data Firehose** and then choose **Create delivery stream**.

1. For **Source and destination**, configure these settings:
  - **Source**: Direct PUT
  - **Destination**: Amazon S3
2. For **Transform and convert records - optional**, configure these settings:
  - **Enable data transformation**: Enabled
  - **AWS Lambda function**: ARN of the function that you created in Lambda
    - **Note**: For example, the ARN might look like the following: `arn:aws:lambda:us-east-1:<account ID>:function:transform-data`
  - **Version and alias**: \$LATEST

For **Destination settings**, choose **Browse**.

Select the S3 bucket that you created for this exercise and then select **Choose**.

At the bottom of the page, choose **Create delivery stream**.

It may take up to 5 minutes to create the Kinesis Data Firehose delivery stream.

#### **Step 4.2: Copying the ARN of the IAM role**

If needed, open the details page for the delivery stream that you created.

Choose the **Configuration** tab.

In the **Service access** section, choose the IAM role.

The role opens in a new IAM window.

Copy the ARN of the IAM role and save it. Your ARN role might look similar to the following: `arn:aws:iam::<account`

```
ID>:role/service-role/KinesisFirehoseServiceRole-PUT-S3-7HMMt-us-east-1-1664893091685.
```

You will need this ARN in the next task.

## Task 5: Adding the Firehose delivery stream ARN to the S3 bucket

In this task, you edit permissions for the S3 bucket so it can store data that's ingested by Kinesis Data Firehose.

Open the Amazon S3 console and open the details page for the bucket you created in this exercise.

Choose the **Permissions** tab.

In the **Bucket policy** section, choose **Edit** and paste the following script:


```
{  "Version": "2012-10-17",  "Id": "PolicyID",  "Statement": [    {      "Sid": "StmtID",      "Effect": "Allow",      "Principal": {        "AWS": "<Enter the ARN for the Kinesis Firehose IAM role>"      },      "Action": [        "s3:AbortMultipartUpload",        "s3:GetBucketLocation",        "s3:GetObject",        "s3:ListBucket",        "s3:ListBucketMultipartUploads",        "s3:PutObject",        "s3:PutObjectAcl"      ],      "Resource": [        "<Enter the ARN of the S3 bucket>",        "<Enter the ARN of the S3 bucket>/*"      ]    }  ] }
```



1. In the script code, replace the following placeholders:

**AWS:** Paste the ARN of the Kinesis Data Firehose IAM role. You saved this ARN in the last step of the previous task. For example:

```
"Principal": {  "AWS": "arn:aws:iam::<account ID>:role/service-  
role/KinesisFirehoseServiceRole-PUT-S3-7Hmt-us-east-1-1664893091685"},
```



**Resource:** Paste the ARN of the S3 bucket for both placeholders. For example:

○

```
"Resource": [  "arn:aws:s3::DOC-EXAMPLE-BUCKET",  
"arn:aws:s3::DOC-EXAMPLE-BUCKET/*"]
```

Save your changes.

## Task 6: Creating an API in API Gateway

In this task, you create a REST API in API Gateway. The API serves as a communication gateway between your application and AWS services. In this exercise, you use API Gateway to insert mock data.

Open the API Gateway service console.

1. On the **REST API** card with an open authentication, choose **Build** and configure these settings:
  - **Choose the protocol:** REST
  - **Create new API:** New API
  - **API name:** clickstream-ingest-poc
  - **Endpoint Type:** Regional

Choose **Create API**.

In the **Resources** pane, on the **Actions** menu, choose **Create Resource**.

On the **New Child Resource** page, name the resource poc and choose **Create Resource**.

On the **Actions** menu, choose **Create Method**.

On the method menu (down arrow), choose **POST**. Choose the checkmark to save your changes.



2. On the **POST - Setup** page, configure the following settings:
  - **Integration type:** AWS Service
  - **AWS Region:** us-east-1
  - **AWS Service:** Firehose
  - **AWS Subdomain:** Keep empty
  - **HTTP method:** POST
  - **Action Type:** Use action name
  - **Action:** PutRecord
  - **Execution role:** Paste the ARN of the APIGateway-Firehose role that you created in task 1
    - **Note:** For example, the ARN might look like the following: `arn:aws:iam::<account ID>:role/APIGateway-Firehose`
  - **Content Handling:** Passthrough
  - **Use Default Timeout:** Keep selected

Save your changes.

Choose the **Integration Request** card.

Expand **Mapping Templates** and for **Request body passthrough**, choose **When there are no templates defined (recommended)**.

Choose **Add mapping template**.

For **Content-Type**, enter `application/json` and save your changes by choosing the check mark.

In the **Generate template** box, paste the following script:

```
{  "DeliveryStreamName": "<Enter the name of your delivery stream>",
  "Record": {    "Data":
    "$util.base64Encode($util.escapeJavaScript($input.json('$')).replace('\'',
    '''))"  } }
```



In the script code, replace the **DeliveryStreamName** placeholder value with the name of the Kinesis Data Firehose delivery stream that you created

**Note:** You can find the name of the Firehose delivery stream in the Amazon Kinesis console, in the **Delivery streams** tab. For example: "DeliveryStreamName": "PUT-S3-AAaAA"

Choose **Save**.

Go back to the **/poc - POST - Method Execution** page.

Choose **Test**.

For the test, you will send several discrete JSON payloads to the API. These payloads simulate how an application frontend sends small bits of data each time a person looks through an item in the menu.

In the **Request Body** box, paste the following JSON:

```
{"element_clicked":"entree_1","time_spent":67,"source_menu":"restaurant_name","created_at":"2022-09-11 23:00:00"}
```

Choose **Test**.

Review the request logs (on the right side of the window) and confirm that you see the following messages: "Successfully completed execution" and "Method completed with status: 200".

These messages indicate that API Gateway processed the data successfully.

Replace the previous JSON by pasting the following JSON payload and then choose **Test**. Again, confirm that API Gateway processed the data successfully.

```
{"element_clicked":"entree_1","time_spent":12,"source_menu":"restaurant_name","created_at":"2022-09-11 23:00:00"}
```



Repeat the steps for pasting the code, testing the API, and confirming success for each of the following JSON payloads

## Entree 4

```
{"element_clicked":"entree_4","time_spent":32,"source_menu":"restaurant_name","created_at":"2022-09-11 23:00:00"}
```

## Drink 1

```
{"element_clicked":"drink_1","time_spent":15,"source_menu":"restaurant_name","created_at":"2022-09-11 23:00:00"}
```

## Drink 3

```
{"element_clicked":"drink_3","time_spent":14,"source_menu":"restaurant_name","created_at":"2022-09-11 23:00:00"}
```

In the next task, you will verify that the solution is working by creating an Amazon Athena table that displays ingested entries.

## Task 7: Creating an Athena table

In this task, you create a table in Athena. You also run a Structured Query Language (SQL) query to view the payloads that you inserted with the REST API.

Open the Athena service console.

At the left choose **Query editor**.

Choose the **Settings** tab and then choose **Manage**.

Choose **Browse S3** and choose the S3 bucket that you created in this exercise.

Select **Choose** and then **Save**.

Choose the **Editor** tab.

In the **Tables and views** section, on the **Create** menu, choose **CREATE TABLE AS SELECT**.

In the **Query** editor, replace the placeholder code with the following script:

```
CREATE EXTERNAL TABLE my_ingested_data (element_clicked STRING, time_spent
INT, source_menu STRING, created_at STRING) PARTITIONED BY (datehour STRING) ROW
FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' with serdeproperties
( 'paths'='element_clicked, time_spent, source_menu, created_at' ) LOCATION
"s3://<Enter your Amazon S3 bucket name>/" TBLPROPERTIES ("projection.enabled"
= "true", "projection.datehour.type" = "date", "projection.datehour.format" =
"yyyy/MM/dd/HH", "projection.datehour.range" =
"2021/01/01/00,NOW", "projection.datehour.interval" =
"1", "projection.datehour.interval.unit" = "HOURS", "storage.location.template"
= "s3://<Enter your Amazon S3 bucket name>/${datehour}/")
```

1. In the script you pasted, replace the following placeholder values:

- **LOCATION:** Replace `<Enter your Amazon S3 bucket name>` with the name of your bucket
  - **Note:** Make sure that you have `s3://` before the bucket name and a slash (/) at the end (for example: `"s3://DOC-EXAMPLE-BUCKET/"`)
- **storage.location.template:** Replace `<Enter your Amazon S3 bucket name>` with the name of your bucket
  - **Note:** For example: `"s3://DOC-EXAMPLE-BUCKET/${datehour}"`

Choose **Run**.

The query creates the **my\_ingested\_data** table.

Create a new query by choosing the plus sign (+) (at the top-right of the query editor).

In the query editor, paste `SELECT * FROM my_ingested_data;` and choose **Run**.

The query should produce results with the entries that you ran in API Gateway.

## Task 8: Visualizing data with QuickSight

After the clickstream data is processed successfully, you can use QuickSight to visualize data. With QuickSight, you can gain better insights into your streaming data by analyzing it and publishing data dashboards.

The instructions for how to visualize data in QuickSight might differ, depending if you are a new user or an existing user.

**Note:** Amazon QuickSight is a subscription service. If you need to delete your QuickSight account after you complete this exercise, follow the instructions in the final task.

### For new Amazon QuickSight users

Open the QuickSight service console.

Choose **Sign up for QuickSight**.  
Choose **Enterprise** and **Continue**.

Set up an account and choose **Finish**.

In the upper-right corner, open the user menu by choosing the user icon and then choose **Manage QuickSight**.

In the navigation pane, choose **Security & permissions** and in **QuickSight access to AWS services**, choose **Manage**.

Under **Amazon S3**, choose **Select S3 buckets**.

Select the bucket that you created in this exercise, and also select **Write permission for Athena Workgroup**.

Choose **Finish** and save your changes.

Return to the QuickSight console.

In the **Analyses** tab, choose **New analysis**.

Choose **New dataset**.

1. Choose **Athena** and configure the following settings:
  - **Name datasource:** poc-clickstream
  - **Select workgroup:** [primary]

Choose **Create data source**.

In the **Choose your table** dialog box, select the **my\_ingested\_data** table, and choose **Select**.

In the **Finish dataset creation** dialog box, make sure that **Import to SPICE for quicker analytics** is selected, and choose **Visualize**.

View your visualization results by selecting field items and visual types for the diagram.

For more information about how to visualize data in Amazon QuickSight, see [Tutorial: Create an AmazonQuickSight analysis](#).

2.

### For existing Amazon QuickSight users

Open the QuickSight service console.

In the upper-right corner, open the user menu by choosing the user icon and then choose **Manage QuickSight**.

In the navigation pane, choose **Security & permissions** and in **QuickSight access to AWS services**, choose **Manage**.

Under **Amazon S3**, choose **Select S3 buckets**.

Select the bucket that you created in this exercise, and also select **Write permission for Athena Workgroup**.

Choose **Finish** and save your changes.

Return to the QuickSight console by choosing the QuickSight icon (in the upper-left area of the webpage).

Choose **New analysis**.

Choose **New dataset**.

1. Choose **Athena** and configure these settings:
  - **Data source name:** poc-clickstream
  - **Athena workgroup:** [primary]

Choose **Create data source**.

In the **Choose your table** dialog box,  
select **my\_ingested\_data** and choose **Select**.

In the **Finish dataset creation** dialog box, keep **Import to SPICE for quicker analytics** selected and choose **Visualize**.

View your visualization results by selecting field items and visual types for the diagram

For more information about how to visualize data in Amazon QuickSight, see [Tutorial: Create an AmazonQuickSight analysis](#).

## Task 9: Deleting all resources

To avoid incurring costs, we recommend that you delete the AWS resources that you created in this exercise.

1. Delete the QuickSight dashboards.
  1. Return to the QuickSight console by choosing the **QuickSight** icon (in the upper-left area of the webpage).
  2. If needed, in the navigation pane, choose **Analyses**.
  3. On the **my\_ingested\_data** card, open the actions menu by choosing the ellipsis icon.
  4. On the actions menu, choose **Delete** and confirm your action.
  5. Confirm your action.
  6. In the navigation pane, choose **Datasets**.
  7. For each dataset, on the actions menu (ellipsis icon), choose **Delete** and confirm your action.

Delete your QuickSight account.

If you will not be using QuickSight in the future, you can delete your account.

**Note:** Amazon QuickSight is a subscription service. For more information about QuickSight pricing, see the “Standard Edition” tab on the [Amazon QuickSight Pricing](#) page.

1. In the QuickSight console, choose the user icon and then choose **Manage QuickSight**.
  2. In the navigation pane, choose **Account settings**.
  3. Choose **Delete account** and confirm your action.
2. Delete the S3 bucket.
  1. Return to the Amazon S3 console.
  2. Choose the bucket that you created for this exercise and delete all files in the bucket.
  3. Select the bucket name, choose **Delete**, and confirm your action.
3. Delete the Athena table and queries.
  1. Return to the Athena console.
  2. Make sure that you are on the **Editor** tab.
  3. In the navigation pane, on the actions menu (ellipsis icon) for **my\_ingested\_data**, choose **Delete table** and confirm your action.
  4. Close the queries that you created.
  5. Choose the **Settings** tab and then choose **Manage**.
  6. Remove the path to the S3 bucket and save your changes.
4. Delete the API Gateway configuration.
  1. Return to the API Gateway console.
  2. Select the API that you created.
  3. On the **Actions** menu, choose **Delete** and confirm your action.
5. Delete the Kinesis Data Firehose delivery stream.
  1. Return to the Kinesis console.
  2. In the navigation pane, choose **Delivery streams**.
  3. Select the delivery stream that you created, choose **Delete**, and confirm your action.
6. Delete the Lambda function.



1. Return to the Lambda console.
2. Select the **transform-data** Lambda function.
3. On the **Actions** menu, choose **Delete**, and confirm your action.

7. Delete the IAM roles and policies.

1. Return to the IAM dashboard.
2. In the navigation pane, choose **Roles**.
3. Select **APIGateway-Firehose**, choose **Delete**, and confirm your action.
4. In the navigation pane, choose **Policies**.
5. Select **API-Firehose**.
6. On the **Actions** menu, choose **Delete** and confirm the deletion.