# **CSYE 7245 - Big Data Systems & Intelligent Analytics**

## Assignment 2 - Part 1 ML as a Service

### Team 6

|  |  |
| --- | --- |
| **Name** | **NUID** |
| Hardik Thakkar | 11306605 |
| Rugved Gole | 001306940 |
| Vinod Kumar Mullangi | 001057553 |

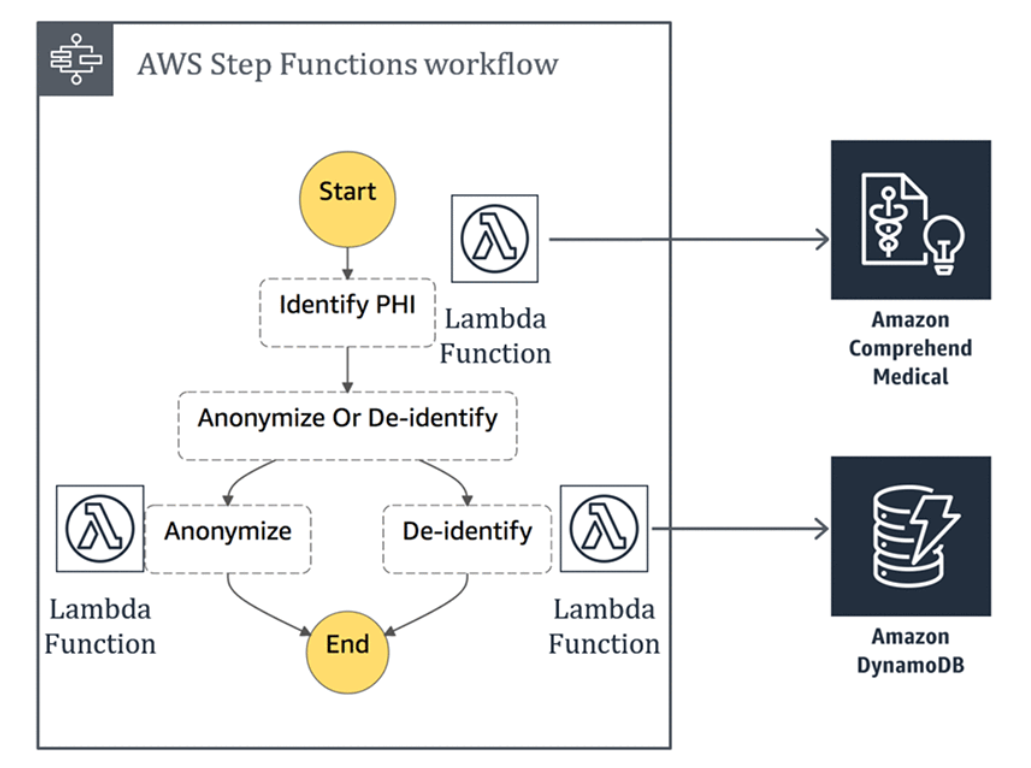
# 

# **Activity 1: Identifying and working with sensitive healthcare data with Amazon Comprehend Medical**

Reference:

<https://aws.amazon.com/blogs/machine-learning/identifying-and-working-with-sensitive-healthcare-data-with-amazon-comprehend-medical/>

Architecture:



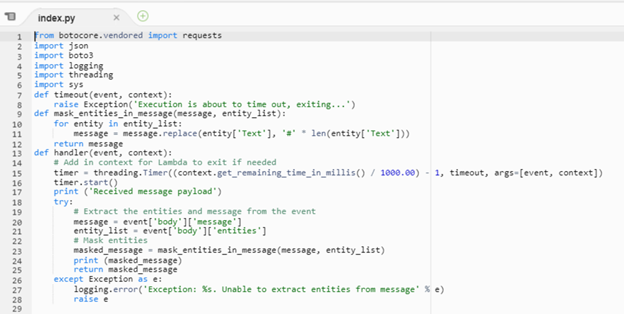
### 

This architecture uses the following services:

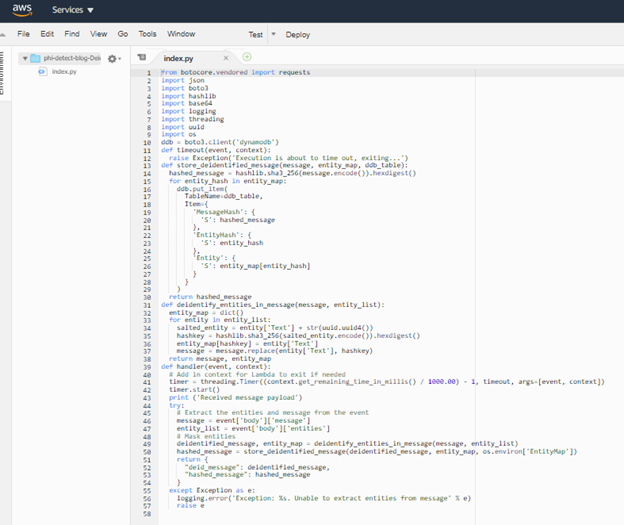
1. Amazon Comprehend Medical to identify entities within a body of text
2. AWS Step Functions and AWS Lambda to coordinate and execute the workflow
3. Amazon DynamoDB to store the de-identified mapping
4. Lambda function to Identify PHI



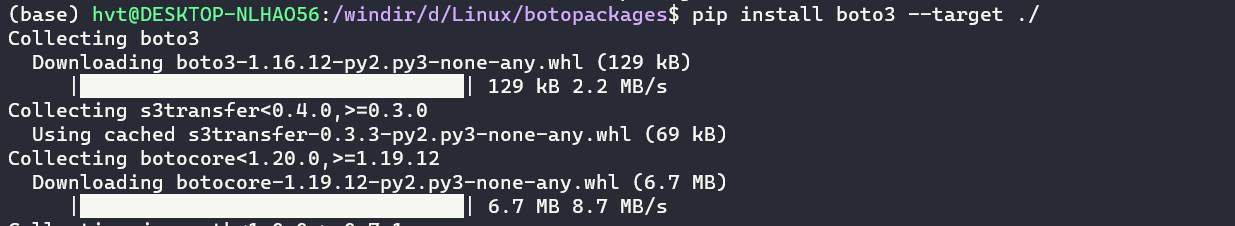
1. Lambda function to Mask entities



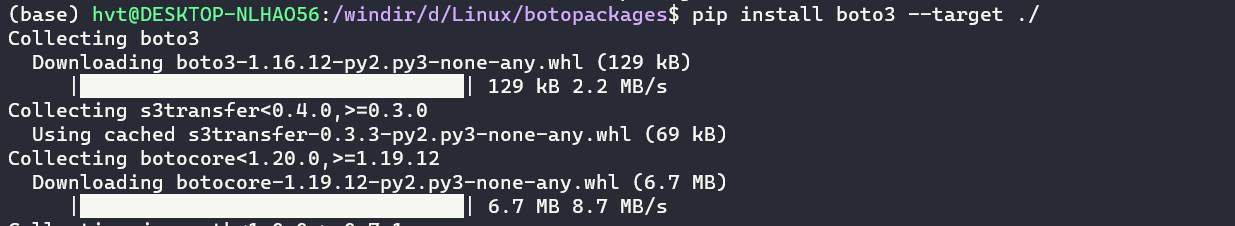
1. Lambda function to De-identify entities



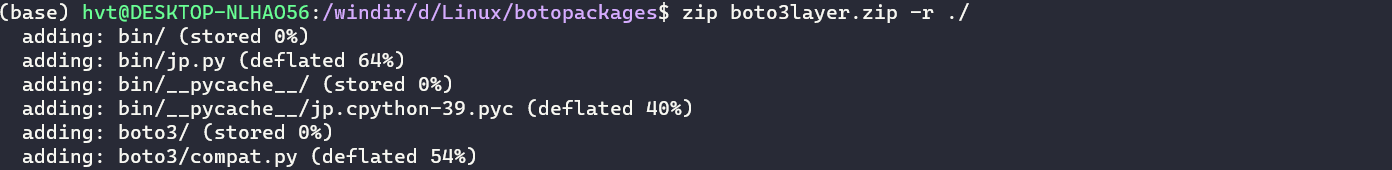
1. Building Boto3 Lambda layer



1. Install boto3



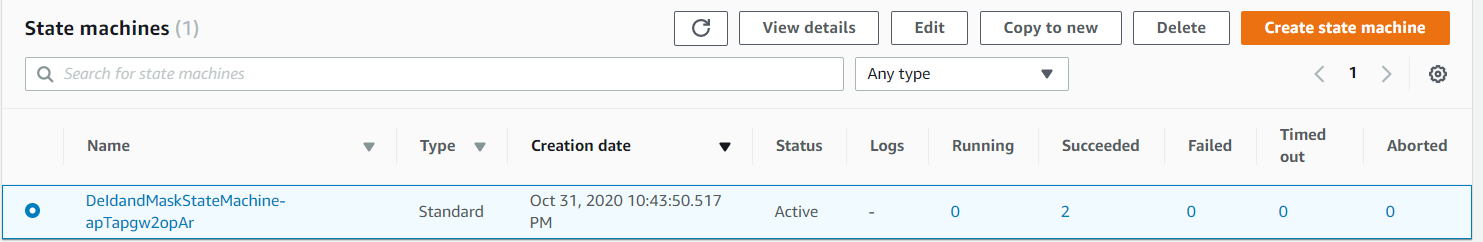
1. Zip to four layer



Layer Version

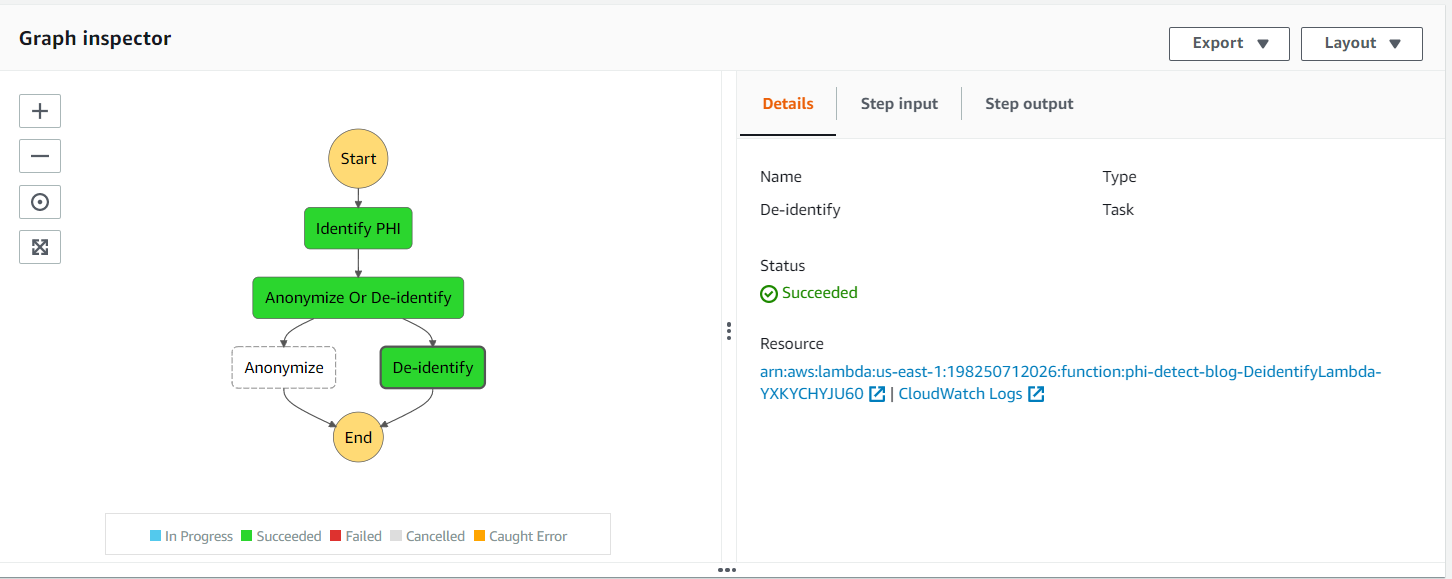


1. Building the state machine



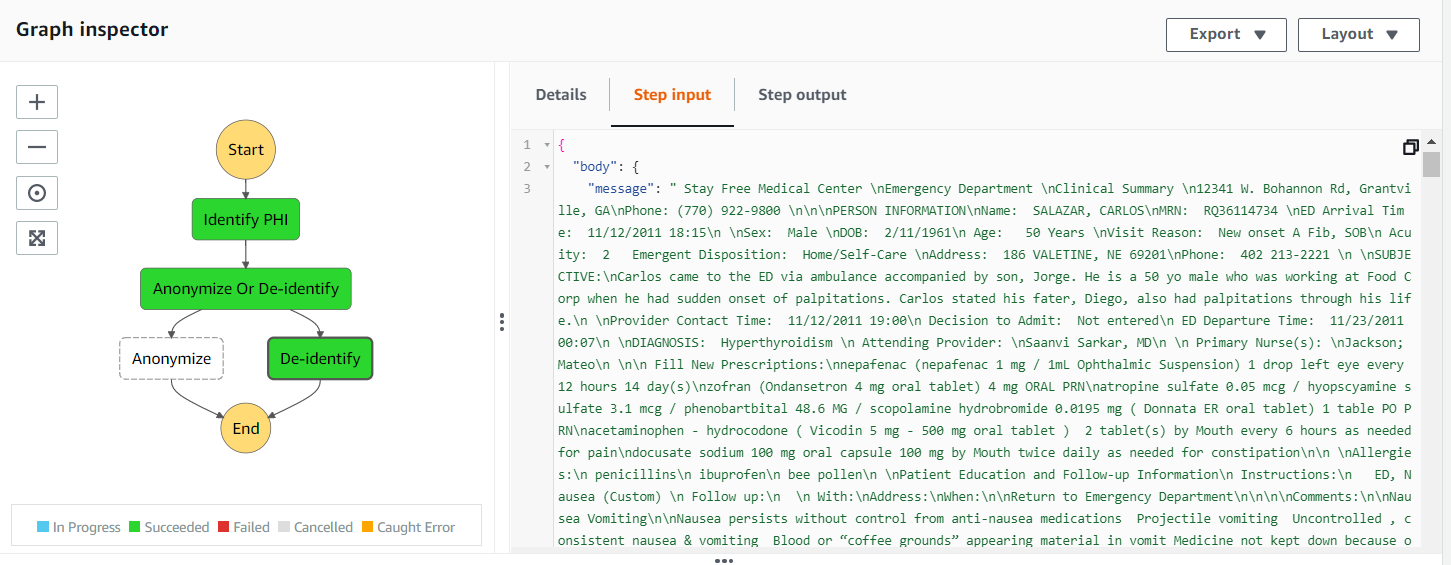
8. Testing the state machine

De-identify :

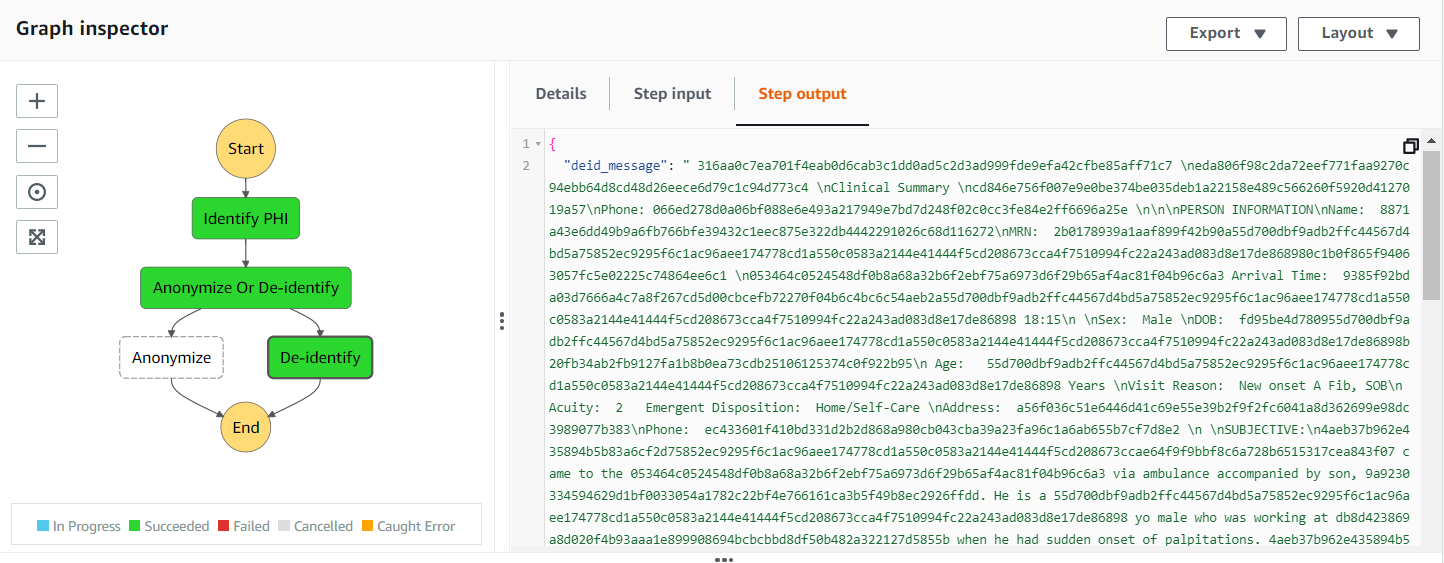


:

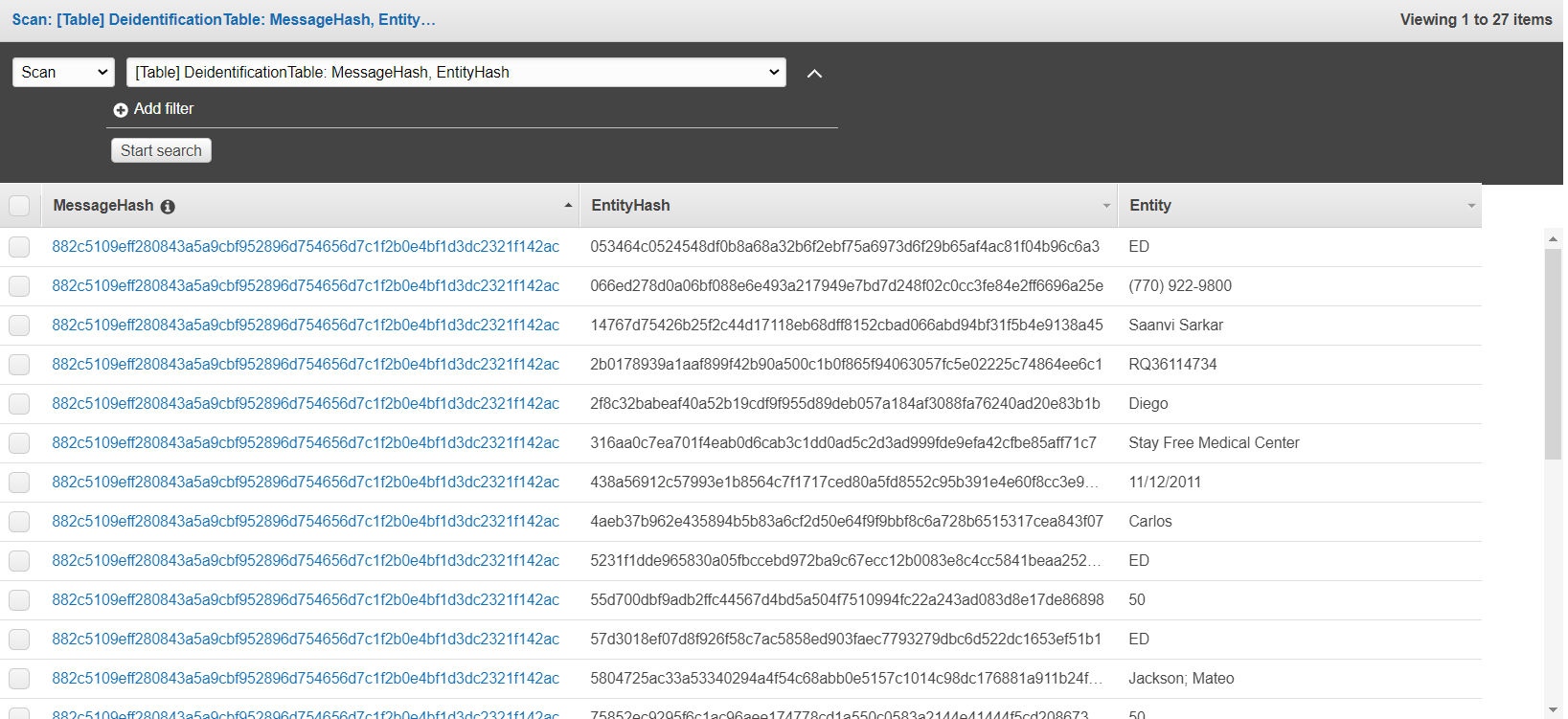
Input :



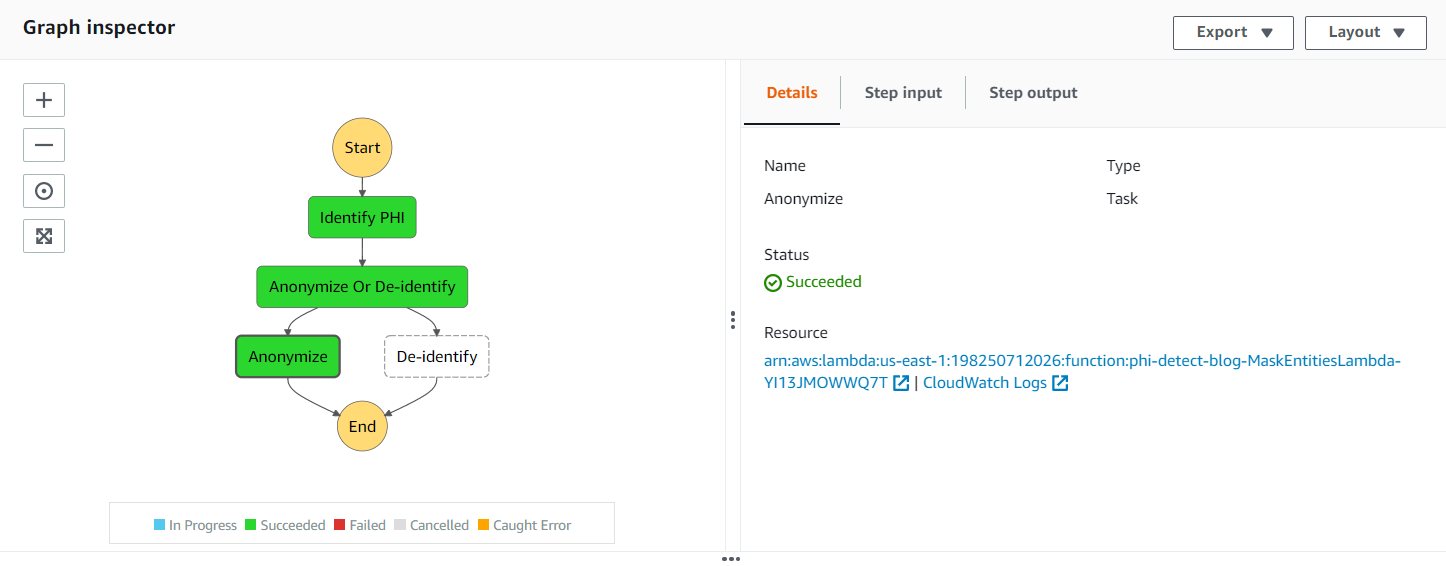
Output :

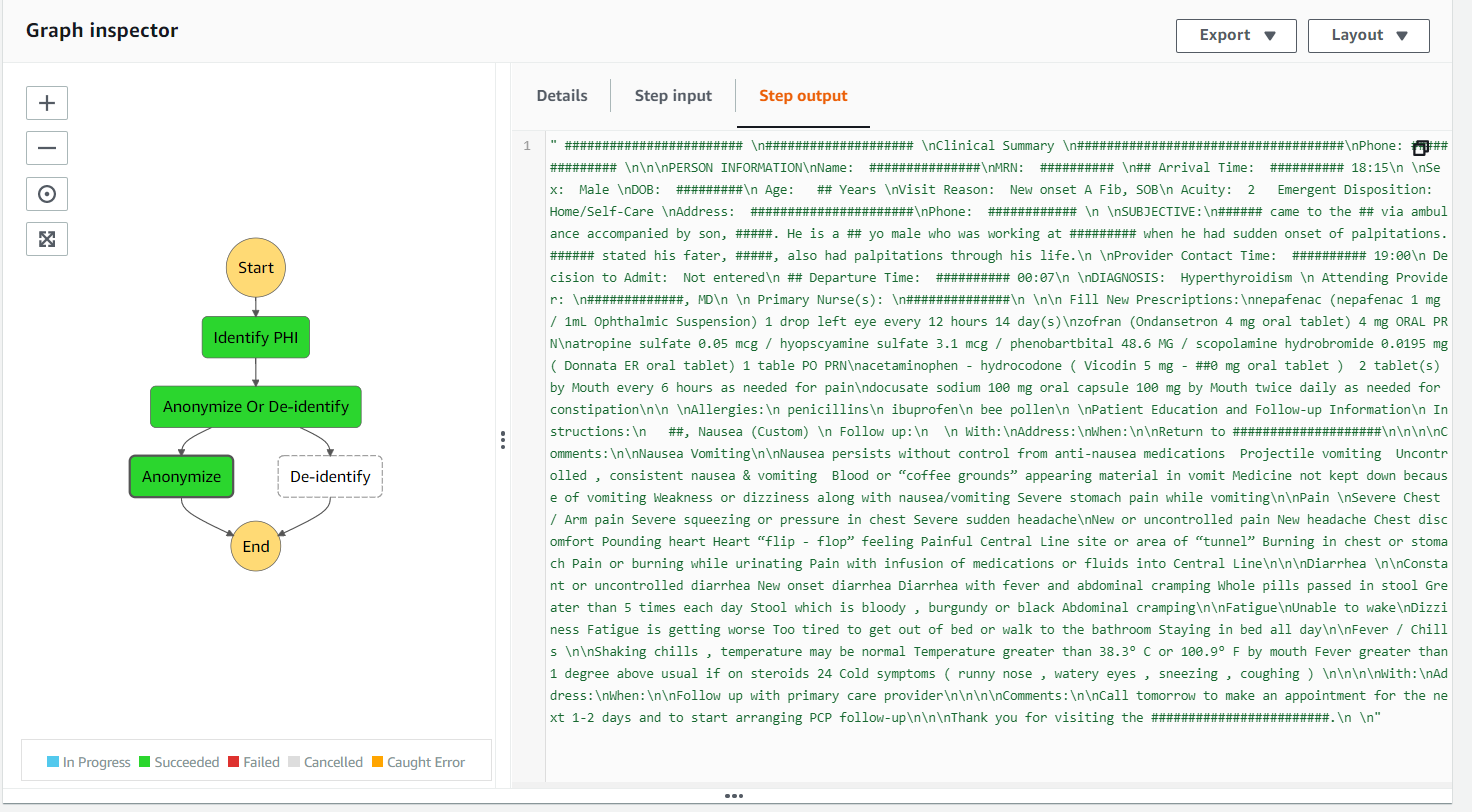


Deidentification Table – Encrypted data



Anonymize :





# **Activity 2: 2020 ComputeFest "Notebook to Cloud" Workshop!**

## 

**Part I: Python Performance**

Pre-requisites:

* Have [Conda](https://docs.conda.io/projects/conda/en/latest/user-guide/install/) installed
* Clone [2020-ComputeFest](https://github.com/Harvard-IACS/2020-ComputeFest.git)

The Jupyter notebook is a commonly used development environment when it comes to writing Python code. It is easy to learn and handy to use. By running a cell, you can interact with your code and see your results. Some pros of developing in a notebook are

* Easy to use: You can break your code into different cells and have immediate outputs. It's good for validating ideas and variable values.
* Visual and Interactive: You can see your changes in real-time.
* Non-linear development: You can go back to a cell and modify variables/functions at any time instead of rerunning the entire notebook.

There is always a tradeoff when it comes to developing large-scale applications. In the first part of the workshop, we assume that participants are familiar with using Jupyter notebooks, and we focus on some advice for how to improve your efficiency when working with a notebook. We will concentrate on two areas for efficiency improvement:

1. How, why, and when to move from a notebook to a script
2. Writing better, more performant code.

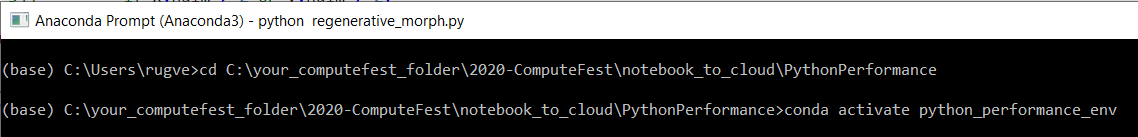
Using Jupyter Notebook:

[Notebook 1](https://colab.research.google.com/drive/1eESxJh5H8GyGQa7027Vf4woPbIMFjY72): Used for initial development(253 KB)

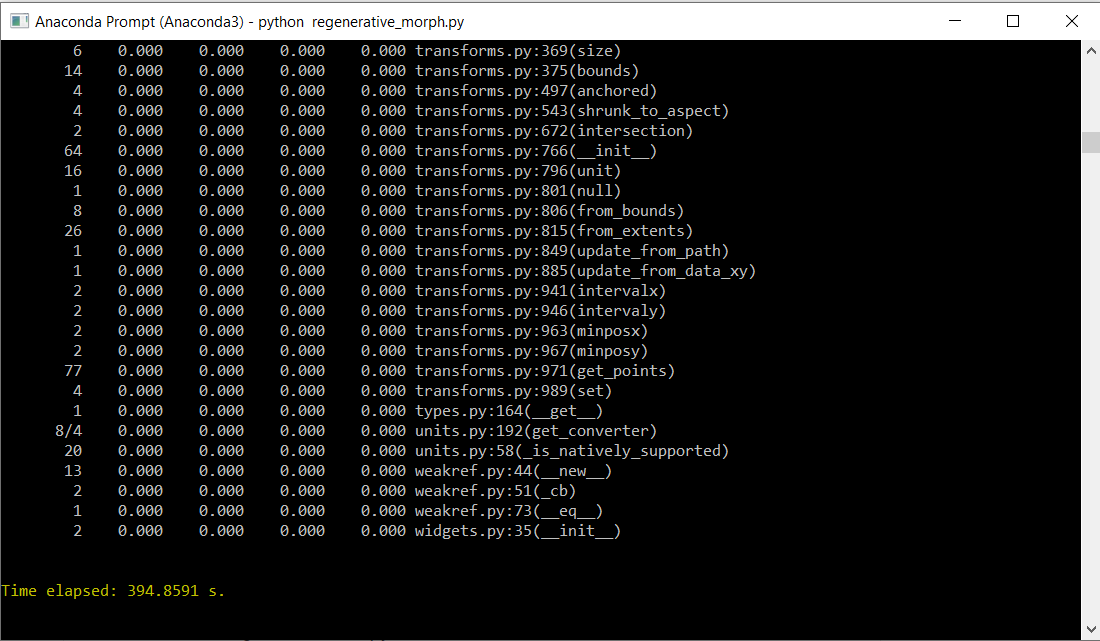
[Notebook 2](https://colab.research.google.com/drive/1lZOynX7lKcJ-Oj0wFBz9P0NeHGhZFE0r): Cleaned up in order to run on cluster(3.3 MB)

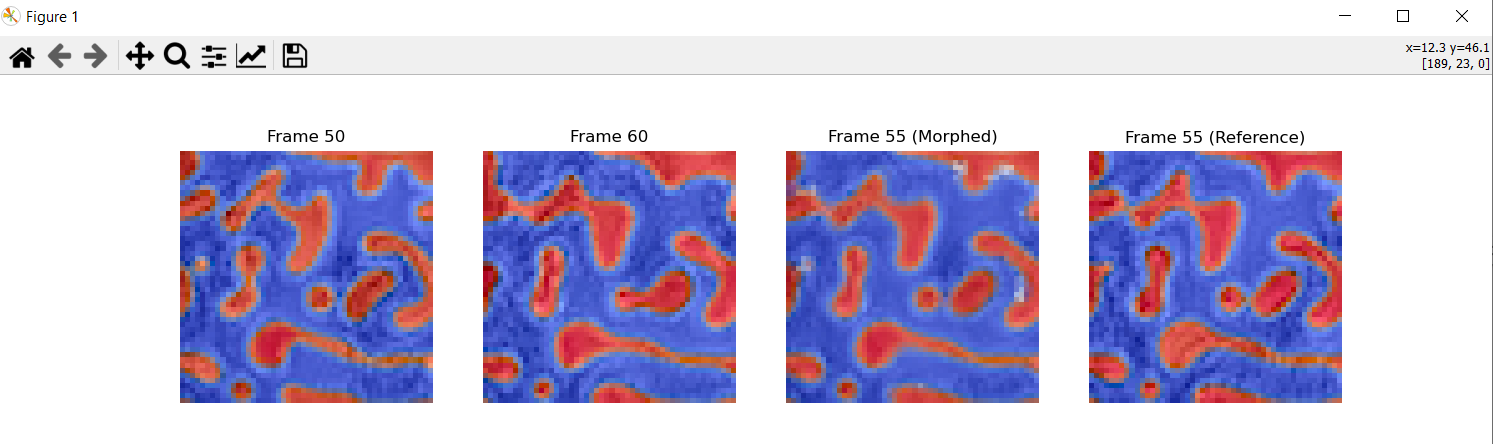
Using Scripts:

To ensure we are all using the same environment (i.e. the same Python version and the same modules) please run the following commands.



Running regenerative\_morph.py as script(11 KB)





Running regenerative\_morph\_slow.py as script

## 

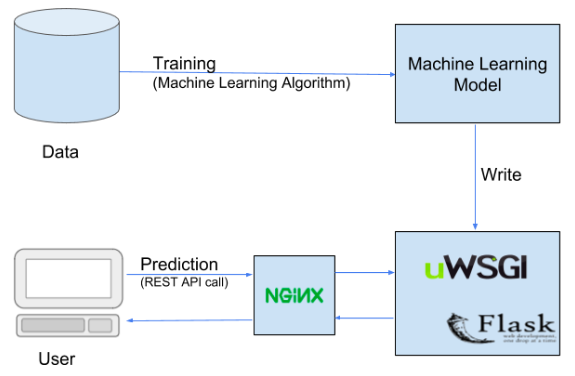
**Conclusion :**

1. Script is more lightweight than Notebook.
2. Script is easy to run on the cloud than Notebook.
3. Script takes less time to execute than Notebook.
4. Notebook is messier than Notebook.

**Part II: Serving a Model with Docker**

### Serving a Model :

Serving a machine learning model means running a server that will accept requests (e.g. HTTP POST) and return model predictions in its responses.

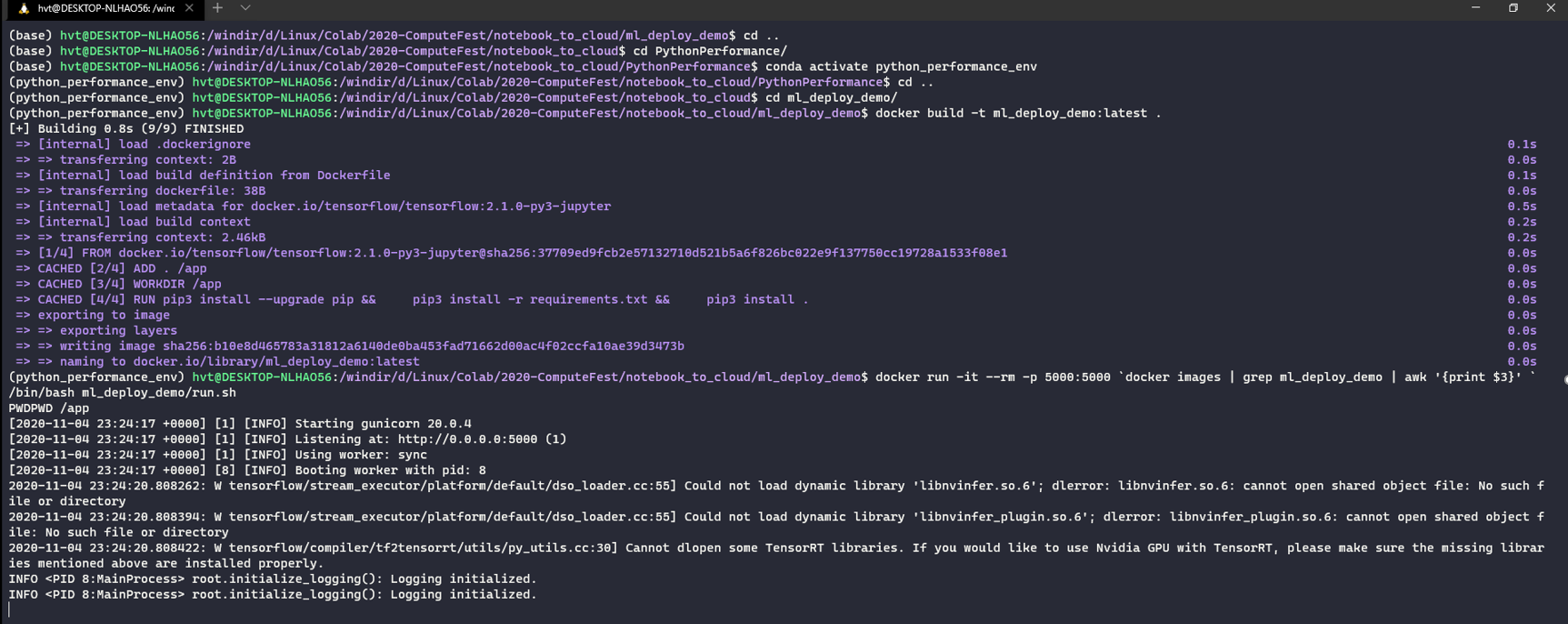
****

## 

Pre-requisites:

* Have [Docker](https://docs.docker.com/get-docker/) & [Postman](https://www.postman.com/downloads/) installed
* Clone [2020-ComputeFest](https://github.com/Harvard-IACS/2020-ComputeFest.git)
* **Have Docker running!**

1. Building a Docker Image & checking it into Docker Images.

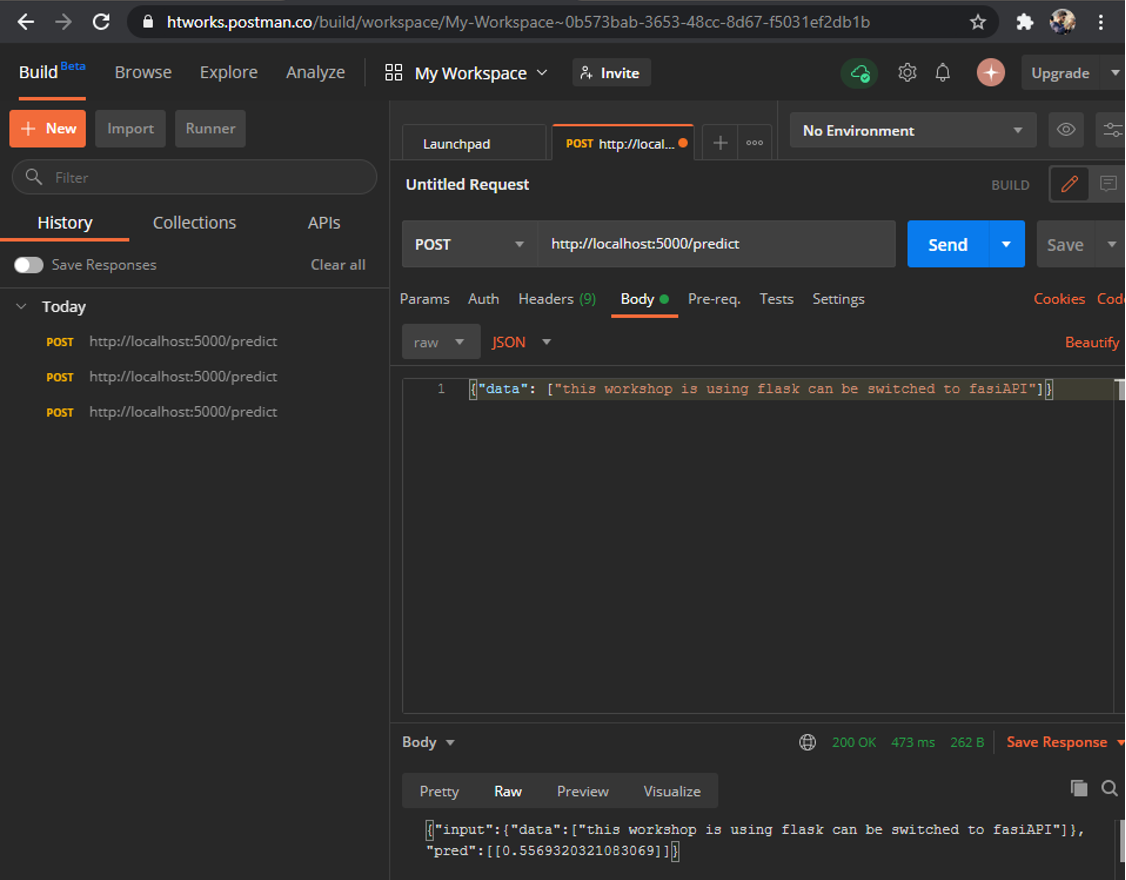




1. Test API :



### Sending Requests to the Model & checking the outputs



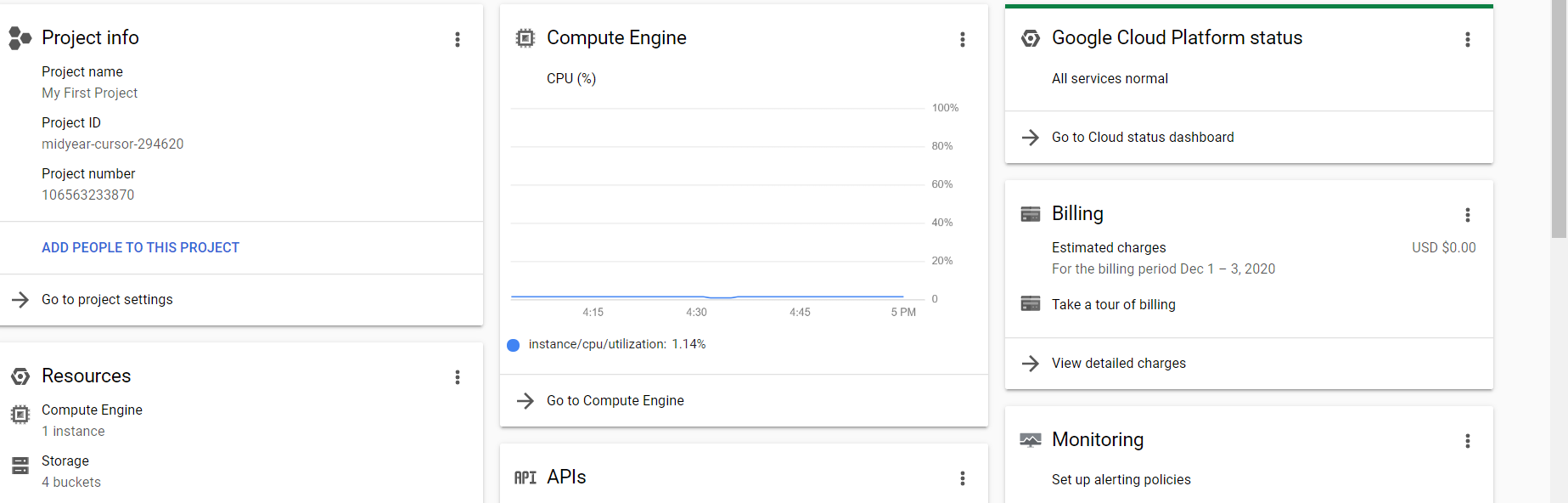
# **Activity 3: DataFlow De-identification pipeline**

Reference:- <https://github.com/GoogleCloudPlatform/dlp-dataflow-deidentification>

Architecture:

## 

Create a Cloud project:



Validate the transfer:



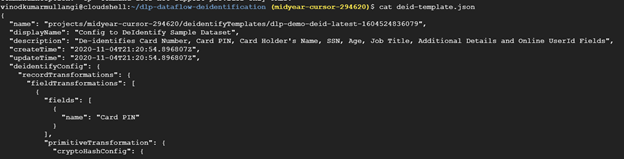
To identify which columns might require DLP de-identification:



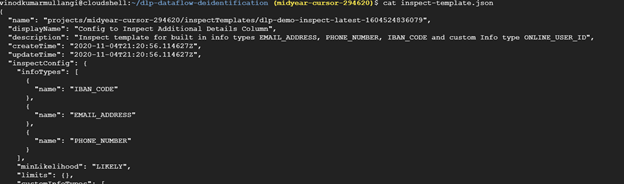
The output is a random key generated in the following format:



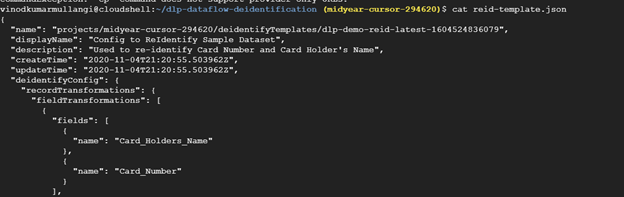
Validate that the de-identification template was successfully created:



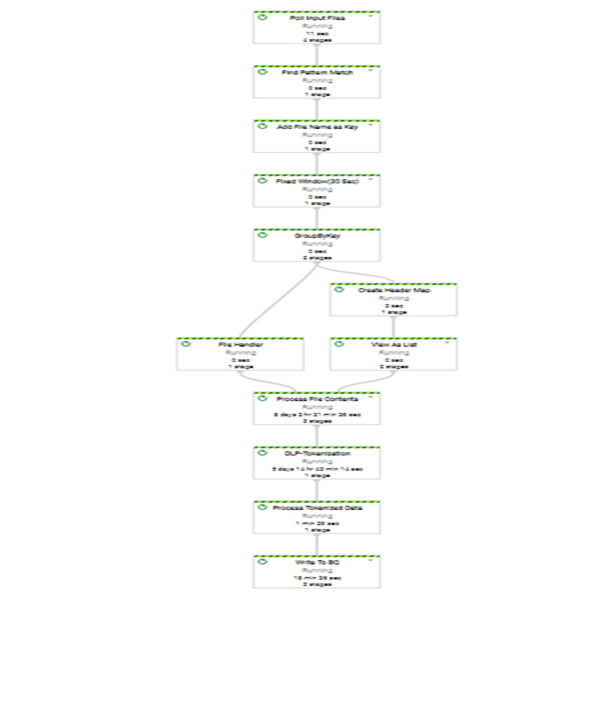
Validate that the inspect template was successfully created:



Validate that the re-identification template was successfully created:

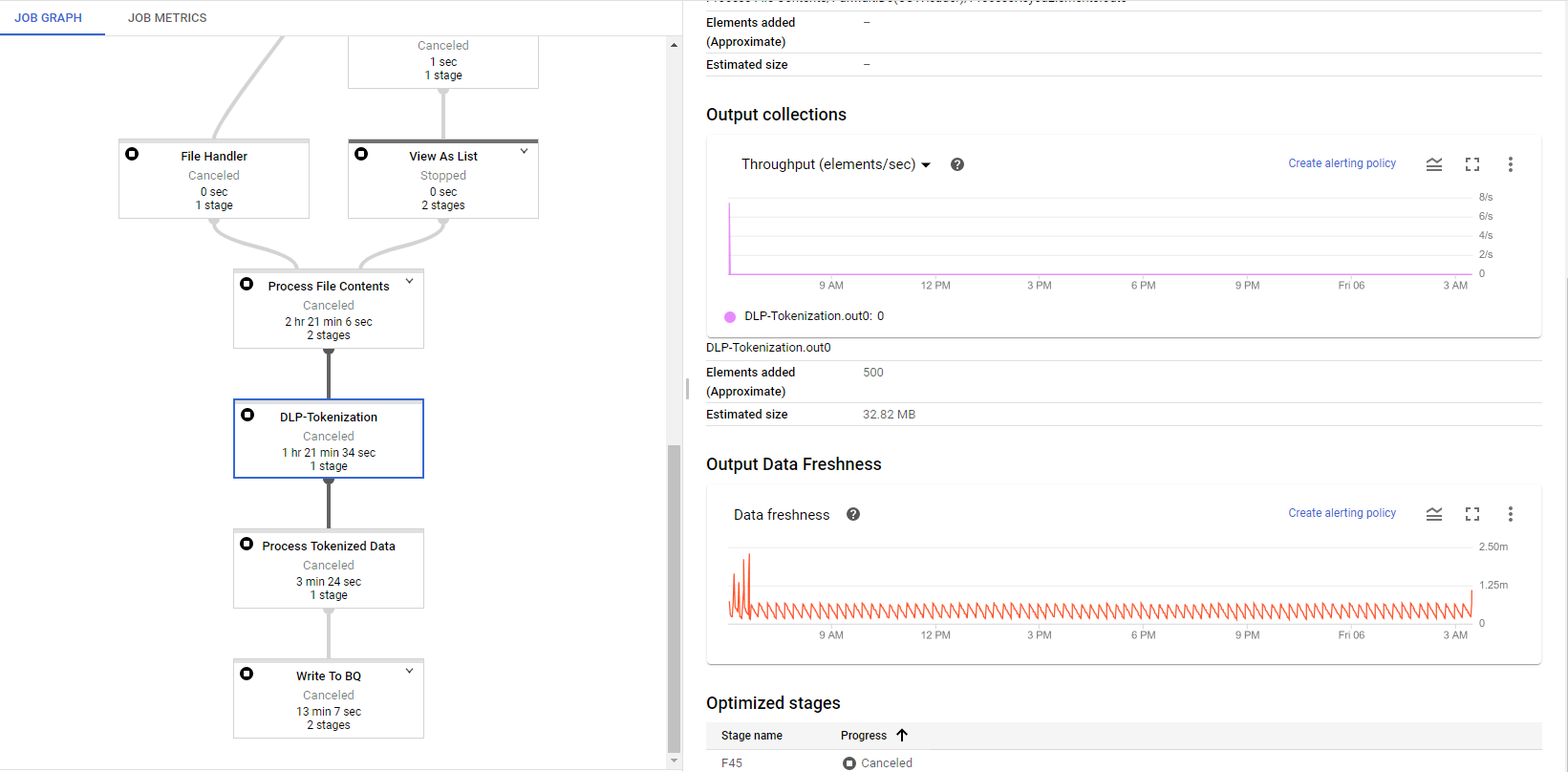


Running the pipeline:

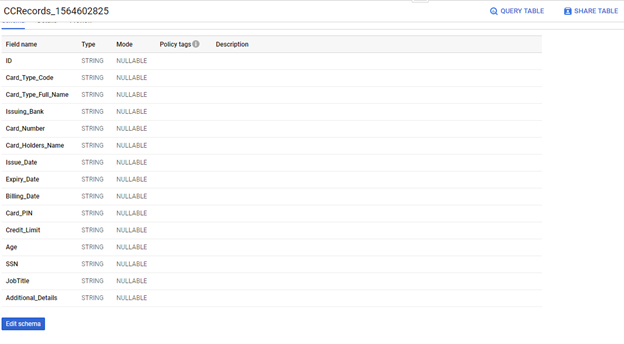




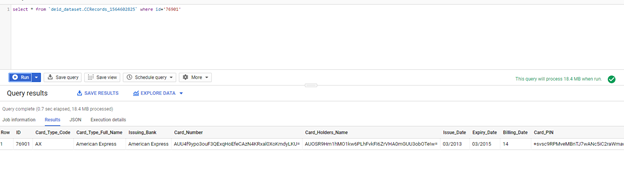
To validate the amount of data processed by the pipeline:



## Validating the de-identified dataset in BigQuery:



Query the de-identified copies of the dataset for the ID 76901:



## 

## 

## Re-identifying the dataset from BigQuery:

