# Gen AI-Based Email Classification and OCR Solution Document

## Problem Statement

Commercial Lending arm of Wells Fargo receives a significant volume of servicing requests via emails. These emails contain diverse requests, often with attachments, which must be ingested into the loan servicing platform to create service requests (SRs) for workflow processing.

Currently, the triage process for incoming SRs is manual, performed by a 'Gatekeeper' who

* Reads and interprets email content and attachments.
* Identifies the intent of the email and classifies it under 'Request Type' and 'Sub Request Type'.
* Extracts key attributes for SR population.
* Assigns the request to the appropriate team or individual based on roles and skills.

### Problems with the Manual Process:

* Time-consuming – Requires multiple Gatekeepers for triage.
* Error-prone – Manual interpretation can lead to misclassification.
* Inefficient at scale – High email volume increases workload, slowing response times.

## Solution Designers

* Sethuraman Karthikeyan - Sethurman.karthikeyan@wellsfargo.com
* Azeez Paloor - Azeez.paloor@wellsfargo.com
* Senthamarai Selvi - Senthamarai.Selvi@wellsfargo.com
* Bhavana Sreeravindra - Bhavana.Sreeravindra@wellsfargo.com
* Vivek Shenoy Trasi - VivekShenoy.trasi@wellsfargo.com

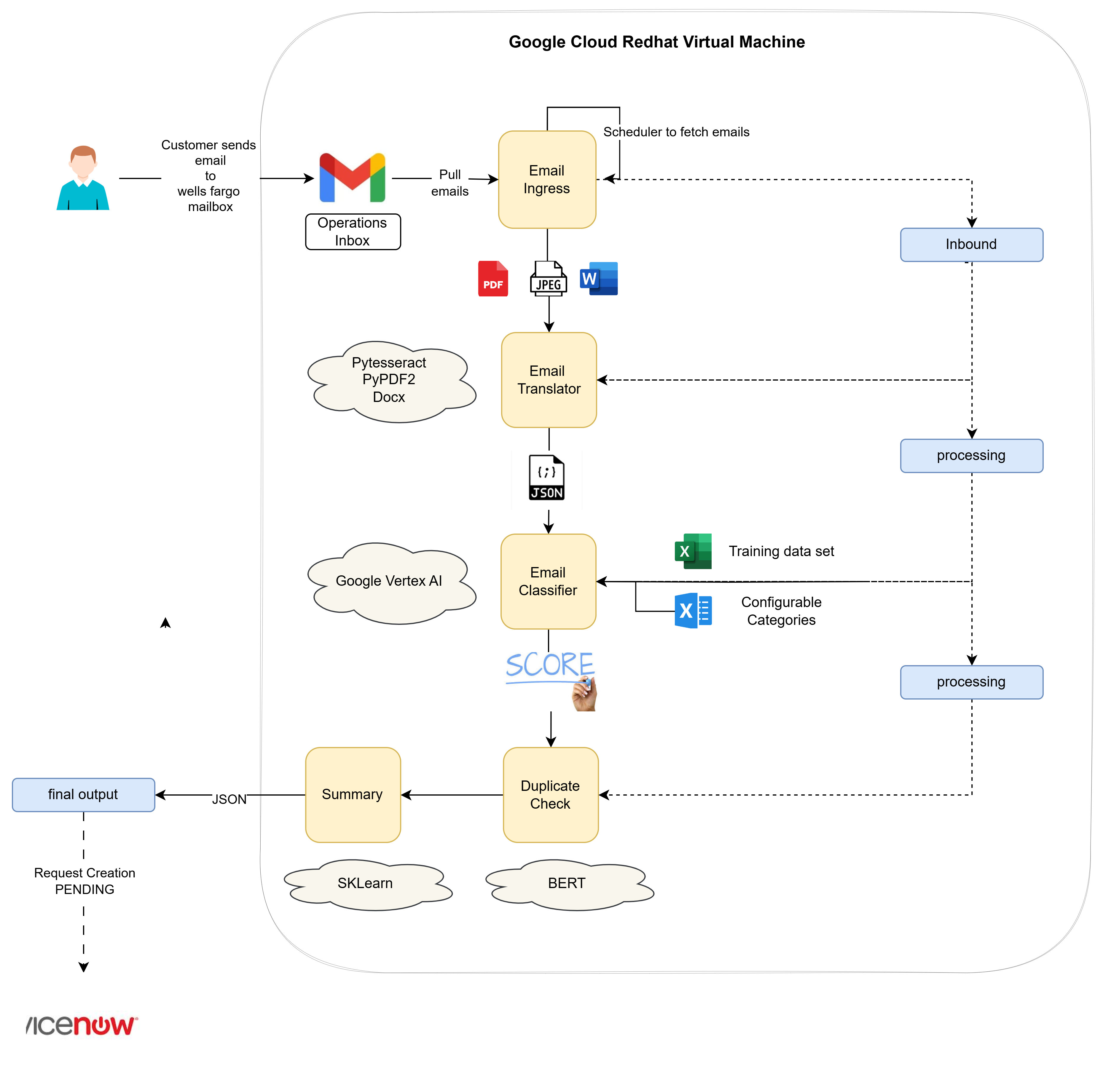
## Solution Approaches

We propose two approaches to solve the problem:

* Google Vertex AI Large Language Model
* Custom ML based classification.

### Approach 1: Google Vertex AI (LLM-based Solution)

Using Google Vertex AI **gemini-1.5-flash-001** model to classify incoming emails with attachments, generate structured output, and perform duplicates detection using the open-source BERT model. To extract content from PDF and images we leverage the **PyTesseract** and **PyPDF2** framework.



* Scheduler to poll the operation mailbox at regular interval.
* Upon arrival of the email, the ingress module to pick up the same , extract attachments and place it on a common data store.
* Translation module to pick the email content , take help of PyTesseract and PyPDF frameworks to extract the content from attached images and PDF and store them as a text on common data store.
* Classifier module will invoke the LLM for classifying the incoming payload. The contents of emails and its attachments are concatenated as a single text file and sent to “**gemini-1.5-flash-001**” model for request type and sub-request type classification. It also provides the summary of analysis along with the confidence score and named entities involved in the transaction .
  + The model type is configurable.
  + The query prompt is also configurable.
  + The System Instructions are also configurable.
* Once content is classified, the payload is fed to BERT model with cosine similarity for duplication classification and the final output is generated as a JSON.
* Ideally we would have a router module here which would then route the instructions to relevant end system.

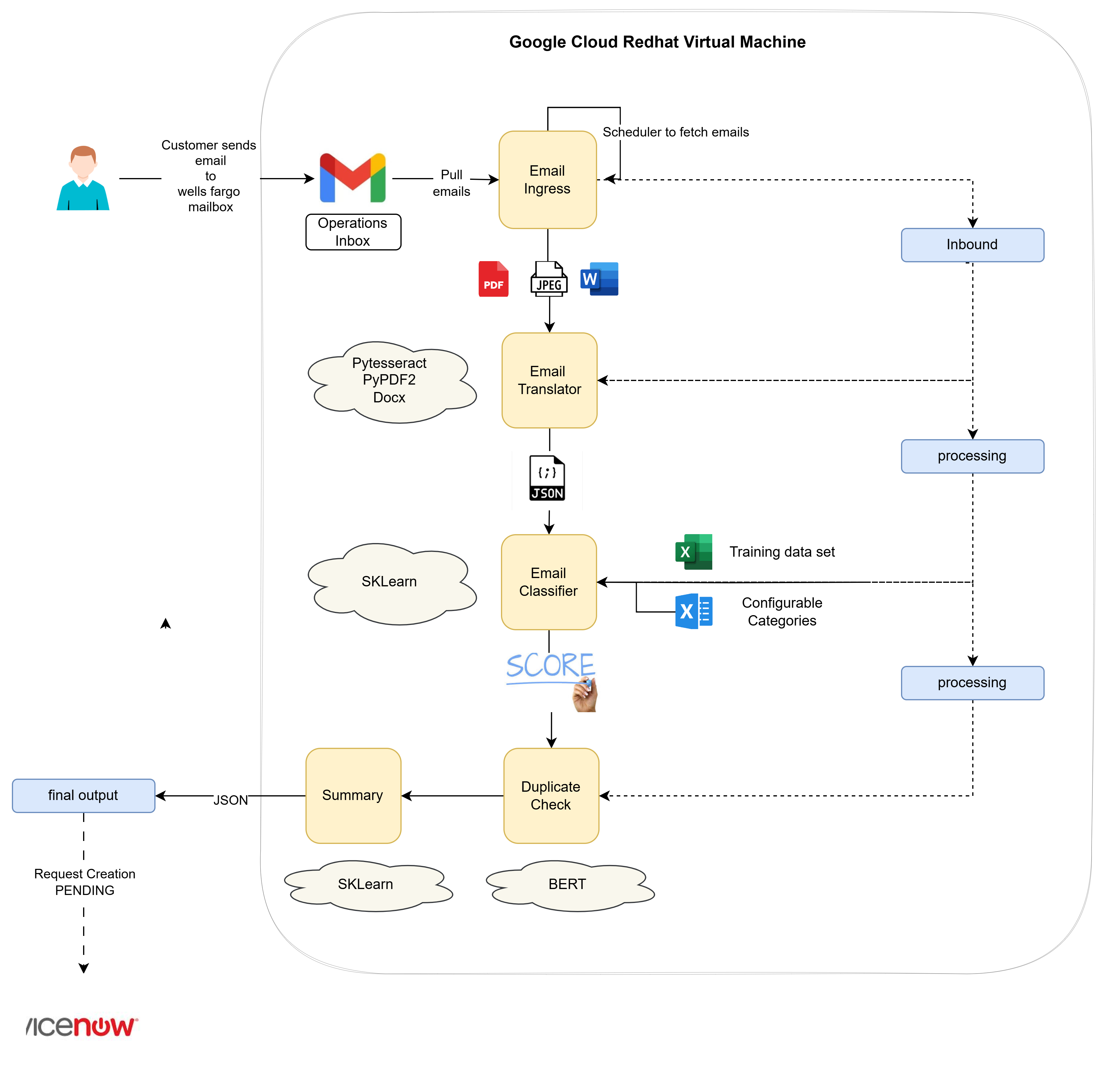
### Tech Stack:

* Programming Language: Python , Java
* ML Libraries: Scikit-Learn, TensorFlow, Hugging Face Transformers

## Approach 2: Custom ML Model Solution

Develop a Machine Learning-based system trained on historical data to classify service requests, extract key attributes, and detect duplicates.

* The approach differs only for Email Classification stage , where we would be using a Machine Learning model to classify an incoming email instead of google vertex AI.



## Comparison: ML Model vs. Vertex AI Solution

|  |  |  |
| --- | --- | --- |
| Feature | ML-Based Approach | Vertex AI-Based Approach |
| Classification Method | Supervised ML (RandomForest, TF-IDF) | LLM-based classification |
| Training Data Requirement | Requires training on labeled data | Pre-trained LLM, minimal training |
| Duplicate Detection | BERT + Cosine Similarity | BERT + Cosine Similarity |
| Customization | Full control over model & logic | Limited customization |
| Deployment Complexity | Requires model training & optimization | Easier to deploy via API |
| Scalability | Requires infrastructure scaling | Cloud-native & scalable |
| Turnaround Time | Moderate | Faster due to pre-trained models |
| Cost | Lower for on-premises | Higher due to API usage |

## Conclusion & Recommendation

When to Use Custom ML Approach:

* If full control over classification logic is required.
* If large historical email data is available for training.
* If cost is a concern (avoiding API costs of cloud-based LLMs).

When to Use Vertex AI-Based Approach:

* If a ready-made, scalable solution is preferred.
* If faster implementation is required.
* If using Google’s cloud ecosystem for AI services.