# Gen AI-Based Email Classification and OCR Solution Document

## Problem Statement

Challenge:

Commercial Bank Lending Services receive 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 Approaches

We propose two approaches to solve the problem:

### Approach 1: Custom ML Model

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

### Approach 2: Google Vertex AI (LLM-based Solution)

Use Google Vertex AI to classify emails and attachments, generate structured output, and detect duplicates using LLM-powered intent recognition.

## Approach 1: Custom ML Model Solution

Develop a tailored ML-based email classification system with priority-based categorization and duplicate detection.

### Workflow

1. Email Ingress

- Connect to a dedicated email inbox.

- A scheduler checks for unread emails every minute.

- If an unread email is found:

- Extract the email and store it in the input folder.

- Mark the email as read.

2. Email Translation

- Parse emails and extract attachments.

- Generate a structured JSON representation.

- Create a job for classification and duplicate detection.

3. Classifier

- Define categories and subcategories.

- Train and load classification models using TF-IDF and RandomForestClassifier.

- Predict request category and subcategory.

4. Duplicate Detection

- Utilize a BERT model with cosine similarity to identify duplicate requests.

5. Output Processing

- Add a duplicate flag and generate a final structured JSON output.

### Tech Stack:

- Programming Language: Python

- ML Libraries: Scikit-Learn, TensorFlow, Hugging Face Transformers

- Storage: PostgreSQL, Apache Kafka (optional)

## Approach 2: Google Vertex AI Solution

This approach leverages Google Vertex AI for LLM-based email classification and duplicate detection.

1. Email Ingress - Similar to Approach 1.

2. Email Translation - Parses emails and extracts attachments.

3. Classifier - Uses Vertex AI for classification.

4. Duplicate Detection - Uses LLM-powered intent recognition.

5. Output Processing - Generates structured JSON for SR creation.

## 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 | LLM-based intent matching |
| 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.