Comprehensive Implementation Guide: Secure GCP AI Implementation for Responsible MLOps (PRJ-GCP-AI-083)

Author: Manus AI **Date:** January 26, 2026 **Project ID:** `PRJ-GCP-AI-083`

1. Project Overview

The **Secure GCP AI Implementation for Responsible MLOps (PRJ-GCP-AI-083)** project delivers a robust, production-ready Machine Learning Operations (MLOps) framework on Google Cloud Platform (GCP), anchored by the powerful capabilities of **Vertex AI**. This solution is specifically engineered to address the critical industry need for AI systems that are not only performant but also secure, compliant, and ethically responsible.

The core objective is to provide an end-to-end MLOps pipeline that integrates Governance, Risk, and Compliance (GRC) controls directly into the automated workflow. This “compliance-by-design” approach is essential for organizations handling sensitive data, such as documents for processing, where regulatory adherence and ethical considerations are paramount.

The framework leverages the following key GCP services:

- **Vertex AI:** The unified platform for the entire ML lifecycle, including **Vertex AI Pipelines** for orchestration, **Vertex AI Endpoints** for serving, and **Vertex AI Monitoring** and **Explainable AI** for post-deployment governance.
- **Cloud Storage:** Used for secure, encrypted storage of raw data (e.g., invoices) and model artifacts.
- **Cloud KMS (Key Management Service):** Ensures data encryption at rest uses Customer-Managed Encryption Keys (CMEK), providing an additional layer of control over cryptographic keys.
- **Cloud IAM (Identity and Access Management):** Enforces the principle of least privilege across all components.

- **VPC Service Controls (VPC SC):** Establishes a security perimeter to prevent data exfiltration and unauthorized access to sensitive AI and data services.

This project moves beyond basic MLOps automation by embedding security and responsible AI checks (like bias detection and explainability report generation) directly into the continuous integration/continuous delivery (CI/CD) process, ensuring that every model deployed is auditable and trustworthy.

2. Business Context

The implementation of a secure and responsible MLOps framework on GCP yields significant, quantifiable business value across several dimensions, primarily through risk mitigation, operational efficiency, and regulatory compliance.

Quantified Business Value and ROI

Metric	Before PRJ-GCP-AI-083	After PRJ-GCP-AI-083	Business Value/ROI
Model Deployment Time	Weeks (Manual security review, compliance sign-off)	Hours (Automated compliance checks in pipeline)	**Efficiency Gain:** 90%+ reduction in time-to-market for new models.
Data Exfiltration Risk	High (Reliance on perimeter security only)	Low (VPC SC perimeter, CMEK encryption)	**Risk Mitigation:** Estimated cost avoidance of millions in potential regulatory fines (e.g., GDPR, CCPA) and reputational damage.
Model Monitoring Effort	High (Manual log analysis, reactive fixes)	Low (Automated Vertex AI Monitoring)	**Cost Savings:** Up to 75% reduction in engineering hours spent on reactive model maintenance and troubleshooting.
Audit Preparation Time	Months (Manual artifact collection, log correlation)	Days (Automated Model Cards, audit logs)	**Compliance Efficiency:** Significant reduction in compliance overhead, freeing up GRC personnel.
Ethical Risk (Bias)	Unquantified, High	Quantified, Low	**Reputational Value:** Protects brand integrity by proactively detecting and mitigating model bias using Vertex AI Explainable AI.

Efficiency Gains and Cost Savings

The project's automated MLOps pipeline, orchestrated by Vertex AI Pipelines, replaces manual, error-prone processes. This automation translates directly into efficiency gains:

1. **Reduced Compliance Overhead:** By embedding security and compliance checks (e.g., data lineage tracking, bias evaluation) into the pipeline, the system automatically generates audit evidence (Model Cards, execution logs). This drastically reduces the time and cost associated with preparing for internal and external audits.

2. **Faster Iteration Cycle:** The ability to rapidly retrain, validate, and redeploy models in a secure environment accelerates the pace of innovation. Data scientists can focus on model improvement rather than infrastructure and compliance hurdles.
3. **Optimized Resource Utilization:** Leveraging managed services like Vertex AI and Cloud Storage, combined with strategic cost optimization techniques (detailed in Section 9), ensures that compute resources are only consumed when necessary, leading to significant savings compared to maintaining always-on, self-managed infrastructure.

3. GRC Mapping: Governance, Risk, and Compliance

This project is fundamentally designed to meet stringent GRC requirements, aligning the technical implementation with leading global frameworks for information security and AI governance.

Alignment with AI-Specific Frameworks

The architecture explicitly addresses the principles of the **NIST AI Risk Management Framework (AI RMF)** and **ISO/IEC 42001 (Information Technology — Artificial Intelligence — Management System)**, which are crucial for managing high-risk AI systems.

Framework	Core Principle	GCP/Vertex AI Implementation
NIST AI RMF	**Govern** (Risk Culture, Policy)	Use of Model Cards to document model purpose, limitations, and risk assessment.
NIST AI RMF	**Map** (Context, Risk Sources)	Integration of bias detection and fairness metrics within the Vertex AI Pipeline.
NIST AI RMF	**Measure** (Metrics, Evaluation)	Automated **Vertex AI Model Monitoring** for drift, skew, and performance degradation.
NIST AI RMF	**Manage** (Response, Recovery)	Automated retraining and redeployment via Vertex AI Pipelines upon detection of model drift.
ISO/IEC 42001	**A.5.2 (AI System Design)**	Use of **Vertex AI Explainable AI** to provide transparency into model decisions.
ISO/IEC 42001	**A.6.2 (Data for AI Systems)**	Data encryption using **Cloud KMS** and access control via **VPC Service Controls** and **Cloud IAM**.
ISO/IEC 42001	**A.7.2 (Impact Assessment)**	Mandatory inclusion of a human-in-the-loop review stage before final model deployment.

Mapping to Security and Privacy Frameworks

The technical controls implemented directly map to common security and privacy controls required by major regulatory bodies.

Framework	Control/Criteria	GCP/Vertex AI Implementation
ISO 27001	**A.14.2.1 (Secure Development Policy)**	MLOps pipeline enforces code review and security scanning before deployment.
ISO 27001	**A.18.1.3 (Protection of Records)**	Cloud Audit Logs and Model Cards provide immutable, non-repudiable records of all ML activities.
NIST 800-53	**AC-3 (Access Enforcement)**	Fine-grained **Cloud IAM** roles (Least Privilege) and **VPC Service Controls** perimeter.
NIST 800-53	**SC-13 (Cryptographic Protection)**	Data at rest in Cloud Storage is protected by **Cloud KMS** CMEK.
SOC 2	**CC6.1 (Logical Access)**	Dedicated service accounts (`vertex-pipeline-sa`) with restricted permissions for automated processes.
GDPR	**Art. 22 (Automated Decision-Making)**	**Vertex AI Explainable AI** provides the necessary mechanism to explain decisions to data subjects.
GDPR	**Art. 35 (Data Protection Impact Assessment)**	The comprehensive documentation and GRC mapping serve as a foundation for the DPIA.

4. Prerequisites

Before beginning the deployment, ensure the following prerequisites are met.

4.1. GCP Project and Billing

1. **GCP Project:** A Google Cloud Project must be created.
2. **Billing:** Billing must be enabled for the project.

4.2. Local Environment Setup

1. **Google Cloud SDK (`gcloud` CLI):** The SDK must be installed on your local machine or a secure jump host.

```
# Installation instructions are available on the official Google Cloud
documentation.
# After installation, initialize the gcloud environment:
gcloud init

# Authenticate your user account:
gcloud auth application-default login
```

2. **Python and Dependencies:** Ensure Python 3.8+ is installed, along with necessary libraries for local pipeline development (e.g., `google-cloud-aiplatform`, `kfp`).

4.3. Required GCP APIs

The following APIs must be enabled in your target GCP project. This is a critical step for the deployment scripts to function.

API Name	Service	Purpose
`aiplatform.googleapis.com`	Vertex AI	Core MLOps platform (Pipelines, Endpoints, Monitoring)
`cloudresourcemanager.googleapis.com`	Cloud Resource Manager	Managing project-level resources and IAM policies
`iam.googleapis.com`	Cloud IAM	Creating and managing service accounts and roles
`storage.googleapis.com`	Cloud Storage	Data and artifact storage
`cloudkms.googleapis.com`	Cloud KMS	Customer-Managed Encryption Keys (CMEK)

5. Architecture Overview

The architecture is a secure, layered MLOps system designed for high-stakes AI applications. It follows a hub-and-spoke model where Vertex AI acts as the central hub for ML orchestration, surrounded by a strong security perimeter.

Data Flow and Component Interaction

1. **Secure Data Ingestion:** Raw data (e.g., invoices) is uploaded to a dedicated **Cloud Storage** bucket. This bucket is configured with **Customer-Managed Encryption Keys (CMEK)** via **Cloud KMS**, ensuring that the organization retains control over the encryption key.
2. **Pipeline Orchestration: Vertex AI Pipelines** is the MLOps orchestrator. It is triggered (e.g., on a schedule or data arrival) and executes the end-to-end workflow. The pipeline runs using a dedicated, least-privilege **Service Account** (`vertex-pipeline-sa`).
3. **Training and Validation:** The pipeline components (e.g., data preprocessing, model training) run within the secure Vertex AI environment. The training component generates a model artifact and critical Responsible AI artifacts, such as bias evaluation reports and the **Model Card**.
4. **Deployment and Governance:** Upon successful validation, the model is registered in the **Vertex AI Model Registry** and deployed to a **Vertex AI Endpoint**. The endpoint is configured with **Vertex AI Explainable AI** to generate feature attributions for predictions and **Vertex AI Model Monitoring** to continuously check for data drift, concept drift, and performance anomalies.
5. **Security Perimeter: VPC Service Controls (VPC SC)** form a security perimeter around the Cloud Storage and Vertex AI services. This perimeter prevents data exfiltration by restricting access to only authorized networks and resources, ensuring that sensitive data and models never leave the trusted environment.

Key Architectural Decisions

- **CMEK for Data:** Using Cloud KMS for encryption keys satisfies stringent compliance requirements (e.g., HIPAA, PCI DSS) by separating the data from the key management.
- **Dedicated Service Account:** The use of a dedicated, restricted service account for the pipeline minimizes the blast radius in case of a security compromise.
- **Integrated Responsible AI:** Explainability and monitoring are not add-ons; they are integrated components of the deployment step, ensuring that every deployed model is transparent and governed from day one.

6. Step-by-Step Implementation

The deployment process is broken down into four distinct, sequential steps using the `gcloud` CLI.

Step 1: Set Project Variables and Enable APIs

This step initializes the environment and ensures all necessary services are active.

```
# --- 6.1.1. Environment Setup ---

# Set your project ID and region. The project ID must be the one with billing enabled.
export PROJECT_ID="PRJ-GCP-AI-083"
export REGION="us-central1" # Choose a region geographically close to your users/data
export GCS_BUCKET="gs://${PROJECT_ID}-mlops-data"

# Set the gcloud configuration to the target project
echo "Setting gcloud project to: $PROJECT_ID"
gcloud config set project $PROJECT_ID

# --- 6.1.2. API Activation ---

# Enable all required APIs for Vertex AI, IAM, Storage, and KMS
echo "Enabling required GCP APIs..."
gcloud services enable \
    aiplatform.googleapis.com \
    cloudresourcemanager.googleapis.com \
    iam.googleapis.com \
    storage.googleapis.com \
    cloudkms.googleapis.com

echo "APIs enabled successfully. Proceeding to Step 2."
```

Step 2: Create Secure Storage and KMS Key

This step establishes the secure data foundation using CMEK.

```
# --- 6.2.1. Create GCS Bucket ---

# Create a GCS bucket for data and model artifacts. The '-l' flag sets the
location.
echo "Creating GCS bucket: $GCS_BUCKET"
gsutil mb -l $REGION $GCS_BUCKET

# --- 6.2.2. Create KMS Key Ring and Key ---

# Create a KMS Key Ring (a logical grouping of keys)
echo "Creating KMS Key Ring: vertex-ai-keyring"
gcloud kms keyrings create "vertex-ai-keyring" --location $REGION

# Create a KMS Key for encryption. Purpose must be 'encryption'.
echo "Creating KMS Key: vertex-ai-key"
gcloud kms keys create "vertex-ai-key" \
    --keyring "vertex-ai-keyring" \
    --location $REGION \
    --purpose "encryption"

# --- 6.2.3. Configure GCS Bucket for CMEK ---

# Get the full KMS key resource name
export KMS_KEY_NAME="projects/$PROJECT_ID/locations/$REGION/keyRings/vertex-
ai-keyring/cryptoKeys/vertex-ai-key"

# Grant the Cloud Storage Service Agent the ability to encrypt/decrypt with
the KMS key
# This is crucial for GCS to use the CMEK.
export GCS_SA_EMAIL=$(gsutil kms serviceaccount -project $PROJECT_ID)
gcloud kms keys add-iam-policy-binding $KMS_KEY_NAME \
    --member="serviceAccount:$GCS_SA_EMAIL" \
    --role="roles/cloudkms.cryptoKeyEncrypterDecrypter"

# Set the KMS key as the default encryption key for the GCS bucket
gsutil defstorageclass set STANDARD $GCS_BUCKET
gsutil defenc set -k $KMS_KEY_NAME $GCS_BUCKET

echo "Secure storage and KMS key configured. All data uploaded to $GCS_BUCKET
will be CMEK-encrypted."
```

Step 3: Configure Service Account and IAM (Least Privilege)

A dedicated service account is created and granted the minimum necessary permissions.

```
# --- 6.3.1. Create Dedicated Service Account ---

# Create a dedicated service account for the MLOps pipeline execution
echo "Creating Vertex AI Pipeline Service Account: vertex-pipeline-sa"
gcloud iam service-accounts create vertex-pipeline-sa \
    --display-name "Vertex AI Pipeline Service Account"

# Define the service account email for subsequent commands
export SA_EMAIL="vertex-pipeline-sa@$PROJECT_ID.iam.gserviceaccount.com"

# --- 6.3.2. Grant IAM Roles ---

# 1. Grant Vertex AI User role (required to run pipelines and manage
models/endpoints)
echo "Granting roles/aiplatform.user..."
gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member="serviceAccount:$SA_EMAIL" \
    --role="roles/aiplatform.user"

# 2. Grant Storage Admin role (required to read/write data and artifacts in
the GCS bucket)
echo "Granting roles/storage.admin..."
gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member="serviceAccount:$SA_EMAIL" \
    --role="roles/storage.admin"

# 3. Grant KMS Decrypter/Encrypter role (required to use the CMEK for data
access)
echo "Granting roles/cloudkms.cryptoKeyEncrypterDecrypter..."
gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member="serviceAccount:$SA_EMAIL" \
    --role="roles/cloudkms.cryptoKeyEncrypterDecrypter"

echo "Service account configured with least privilege roles."
```

Step 4: Deploy the MLOps Pipeline

This step assumes the Kubeflow Pipeline definition (`pipeline.json`) has been created using the Kubeflow Pipelines SDK or TFX.

```
# --- 6.4.1. Prepare Pipeline Artifacts ---

# NOTE: The 'pipeline.json' file must be created separately, defining the ML
workflow.
# This file includes steps for data preprocessing, training, model evaluation,
# Model Card generation, bias detection, and conditional deployment.

# Upload the pipeline definition to Cloud Storage
echo "Uploading pipeline definition to GCS..."
gsutil cp ./pipeline.json $GCS_BUCKET/pipelines/

# --- 6.4.2. Submit the Pipeline Job ---

# Submit the pipeline run to Vertex AI.
# The --service-account flag ensures the pipeline runs with the restricted SA.
echo "Submitting MLOps pipeline run to Vertex AI..."
gcloud ai pipelines runs create \
    --display-name="invoice-processing-run-$(date +%Y%m%d%H%M%S)" \
    --pipeline-root="$GCS_BUCKET/pipeline_root" \
    --pipeline-file="./pipeline.json" \
    --project=$PROJECT_ID \
    --region=$REGION \
    --service-account="$SA_EMAIL"

echo "Pipeline submitted. Monitor the run status in the Vertex AI Pipelines
console."
```

Conceptual Step 5: Implementing VPC Service Controls (Post-Deployment)

While not a direct deployment step, setting up VPC Service Controls is a critical post-deployment security measure.

1. **Define the Service Perimeter:** Use the Google Cloud Console or `gcloud access-context-manager` commands to create a new Service Perimeter.
2. **Add Resources:** Include the target project (`PRJ-GCP-AI-083`) within the perimeter.
3. **Restrict Services:** Explicitly restrict access to **Vertex AI API** (`aiplatform.googleapis.com`) and **Cloud Storage API** (`storage.googleapis.com`).

4. **Configure Access Levels:** Define which identities (e.g., specific user groups, on-premise IP ranges) are allowed to access the protected resources from *outside* the perimeter.

7. Validation & Testing

Validation ensures that the deployed system is not only functional but also adheres to the security and responsible AI requirements.

7.1. Pipeline Execution Validation

Verify that the MLOps pipeline completed successfully and produced all expected artifacts.

1. **Check Pipeline Status:** Use the `gcloud` command to confirm the run status.

```
# Check the status of the latest pipeline run
gcloud ai pipelines runs list --project=$PROJECT_ID --region=$REGION --
filter="state=Succeeded" --limit=1
```

2. **Artifact Verification:** Navigate to the `pipeline_root` in the GCS bucket (`$GCS_BUCKET/pipeline_root`). Verify the existence of:
    - Trained model artifact.
    - Evaluation metrics file.
    - **Model Card** JSON/YAML file (proof of governance).
    - Bias detection report.

7.2. Model Prediction Validation

Test the deployed model endpoint to ensure it is serving predictions correctly and generating explainability data.

1. **Get Endpoint ID:** Retrieve the ID of the deployed endpoint from the Vertex AI Endpoints console or programmatically.

```
export ENDPOINT_ID="<YOUR_ENDPOINT_ID>"
```

2. **Prepare Sample Request:** Create a `sample_invoice_request.json` file with a valid input payload.

```
{
  "instances": [
    {
      "feature_1": 12.5,
      "feature_2": "supplier_a",
      "invoice_text_embedding": [0.1, 0.2, 0.3, ...]
    }
  ]
}
```

3. **Request Prediction:**

```
gcloud ai endpoints predict $ENDPOINT_ID \
    --region=$REGION \
    --json-request="./sample_invoice_request.json"
```

4. **Verify Explainability:** The prediction response should include `explanation` fields (feature attributions) if the model was deployed with Explainable AI enabled, confirming the transparency control is active.

7.3. Responsible AI Validation

Confirm that the governance components are actively monitoring the model.

1. **Model Monitoring Dashboard:** In the GCP Console, navigate to **Vertex AI > Model Monitoring**. Verify that the monitoring job is active and configured to check for data skew and drift.
2. **Bias Mitigation Report:** Review the bias evaluation report generated by the pipeline. This report should confirm that the chosen mitigation strategy (e.g.,

reweighing, as per the example `config.yaml`) was applied and reduced bias metrics below the defined threshold.

8. Troubleshooting

A secure MLOps environment introduces complexity, particularly around IAM and network controls.

Issue	Potential Cause	Resolution
Pipeline Fails with Permission Denied	The `vertex-pipeline-sa` is missing a required IAM role (e.g., `storage.objectAdmin` instead of `storage.admin`).	**Resolution:** Verify all roles from Step 3 are assigned. Check **Cloud Audit Logs** for the specific permission failure (e.g., `storage.objects.create`). Grant the missing role.
KMS Key Access Denied	The Cloud Storage Service Agent or the `vertex-pipeline-sa` lacks the `roles/cloudkms.cryptoKeyEncrypterDecrypter` role.	**Resolution:** Re-run Step 2.2.3 and 3.3.2 to ensure the KMS binding is correctly applied to both the GCS Service Agent and the pipeline service account.
VPC Service Controls Blocked Access	A user or service is attempting to access a protected resource (Vertex AI, GCS) from outside the perimeter.	**Resolution:** If the access is legitimate, update the VPC SC **Access Levels** to include the source IP range or service account. If the access is from a non-trusted source, the perimeter is working as intended.
Model Drift Detected	The deployed model's performance has degraded due to changes in production data distribution.	**Resolution:** Review the **Vertex AI Model Monitoring** dashboard. If the drift is confirmed, trigger a new MLOps pipeline run to retrain and redeploy the model with fresh, representative data.
`gcloud` command not found	The Google Cloud SDK is not installed or configured correctly.	**Resolution:** Follow the official Google Cloud documentation to install and initialize the SDK. Ensure the `gcloud` binary is in your system's PATH.

Issue	Potential Cause	Resolution
Pipeline Component Failure (e.g., Training)	Insufficient compute resources (e.g., OOM error) or a bug in the custom training code.	**Resolution:** Review the component logs in the Vertex AI Pipelines console. Increase the machine type or GPU allocation for the failing component. Debug and fix the custom code.

9. Cost Optimization

Managing costs in a production MLOps environment requires continuous optimization across compute, storage, and managed services.

9.1. Compute Optimization (Vertex AI)

1. **Autoscaling Endpoints:** Configure the **Vertex AI Endpoint** with a minimum replica count of 1 (or 0 if possible) and a maximum based on peak load. Use the `min_replica_count` and `max_replica_count` parameters during deployment to ensure you only pay for the compute needed to serve predictions.
2. **Custom Machine Types:** For training jobs, use **Custom Machine Types** instead of pre-defined types to precisely match the CPU/RAM requirements of your workload, avoiding over-provisioning.
3. **Spot/Preemptible VMs:** For non-critical, batch-oriented training or preprocessing steps within the pipeline, utilize **Spot VMs** to achieve up to 91% cost savings compared to standard VMs. The pipeline must be designed to handle preemption gracefully.

9.2. Storage and Data Management

1. **Cloud Storage Lifecycle Management:** Implement **Lifecycle Management policies** on the `$GCS_BUCKET` to automatically transition older, less-frequently accessed data and model artifacts to colder, cheaper storage classes (e.g., from Standard to Nearline after 30 days, and to Coldline after 90 days).
2. **Delete Unused Artifacts:** Implement a cleanup step in the MLOps pipeline (or a separate scheduled job) to delete temporary files, intermediate outputs, and

outdated model versions from Cloud Storage to prevent storage bloat.

9.3. Managed Service Efficiency

1. **Pipeline Frequency:** Optimize the frequency of the MLOps pipeline runs. Instead of running hourly, evaluate if daily or weekly runs are sufficient based on data change rate and model performance requirements.
2. **Cloud KMS Key Rotation:** While key rotation is a security best practice, be aware of the associated operational costs. Optimize the rotation schedule to balance security needs with cost implications.

10. Security Best Practices

Security is the foundational pillar of this MLOps implementation, focusing on data protection, access control, and network isolation.

10.1. Network Isolation with VPC Service Controls

VPC Service Controls (VPC SC) are the most critical security component, creating a security perimeter around the sensitive services.

- **Data Exfiltration Prevention:** VPC SC prevents unauthorized access to the protected services (Vertex AI, Cloud Storage) from outside the perimeter, mitigating the risk of data being copied to external, non-trusted Google Cloud projects or public internet destinations.
- **Perimeter Configuration:** The perimeter should include all projects containing sensitive data and models. All interactions with these services must originate from within the perimeter (e.g., from a secure VPC network).

10.2. Principle of Least Privilege (IAM)

The implementation strictly adheres to the principle of least privilege:

- **Dedicated Service Accounts:** Never use the default Compute Engine Service Account for MLOps workloads. Use the dedicated `vertex-pipeline-sa` with only the roles necessary for its function (`aiplatform.user`, `storage.admin`, `cloudkms.cryptoKeyEncrypterDecrypter`).

- **User Access:** Grant users roles like `roles/aiplatform.viewer` or `roles/aiplatform.metadataViewer` for monitoring, and only grant `roles/aiplatform.admin` to a small, trusted group of MLOps engineers.

10.3. Data Encryption and Key Management (CMEK)

- **Encryption at Rest:** All sensitive data in Cloud Storage is encrypted using **Cloud KMS CMEK**. This ensures that the data is protected even if the storage infrastructure is compromised, and the organization maintains full control over the key lifecycle.
- **Encryption in Transit:** All communication between GCP services (e.g., Vertex AI to Cloud Storage) is automatically encrypted in transit via TLS/SSL, a standard GCP security feature.

10.4. Model Governance and Auditability

- **Model Cards:** Enforce the creation of a **Model Card** for every deployed model. The Model Card acts as a single source of truth, documenting the model's purpose, training data, ethical considerations, evaluation metrics, and lineage. This is a core requirement for GRC and auditability.
- **Cloud Audit Logs:** Ensure that **Cloud Audit Logs** (Admin Activity and Data Access logs) are enabled and exported to a secure, long-term storage solution (e.g., a separate, highly restricted GCS bucket or BigQuery dataset) for non-repudiation and forensic analysis.

Cleanup

To avoid incurring future costs, run the following commands to clean up the deployed resources. **Replace placeholders** (`<YOUR_ENDPOINT_ID>`, `<YOUR_MODEL_ID>`) with the actual values from your deployment.

```
# Set variables again for safety
export PROJECT_ID="PRJ-GCP-AI-083"
export REGION="us-central1"
export GCS_BUCKET="gs://${PROJECT_ID}-mlops-data"
export SA_EMAIL="vertex-pipeline-sa@$PROJECT_ID.iam.gserviceaccount.com"

# 1. Delete the deployed model endpoint
echo "Deleting deployed model endpoint..."
gcloud ai endpoints delete <YOUR_ENDPOINT_ID> --region=$REGION --quiet

# 2. Delete the model from the Model Registry
echo "Deleting model from Model Registry..."
gcloud ai models delete <YOUR_MODEL_ID> --region=$REGION --quiet

# 3. Delete the GCS bucket and all its contents
echo "Deleting GCS bucket and all contents: $GCS_BUCKET"
gsutil rm -r $GCS_BUCKET

# 4. Delete the KMS Key and Key Ring
echo "Deleting KMS Key and Key Ring..."
gcloud kms keys destroy "vertex-ai-key" --keyring "vertex-ai-keyring" --
location $REGION --quiet
gcloud kms keyrings delete "vertex-ai-keyring" --location $REGION --quiet

# 5. Delete the service account
echo "Deleting service account: $SA_EMAIL"
gcloud iam service-accounts delete $SA_EMAIL --quiet

echo "Cleanup complete. All resources for PRJ-GCP-AI-083 have been removed."
```