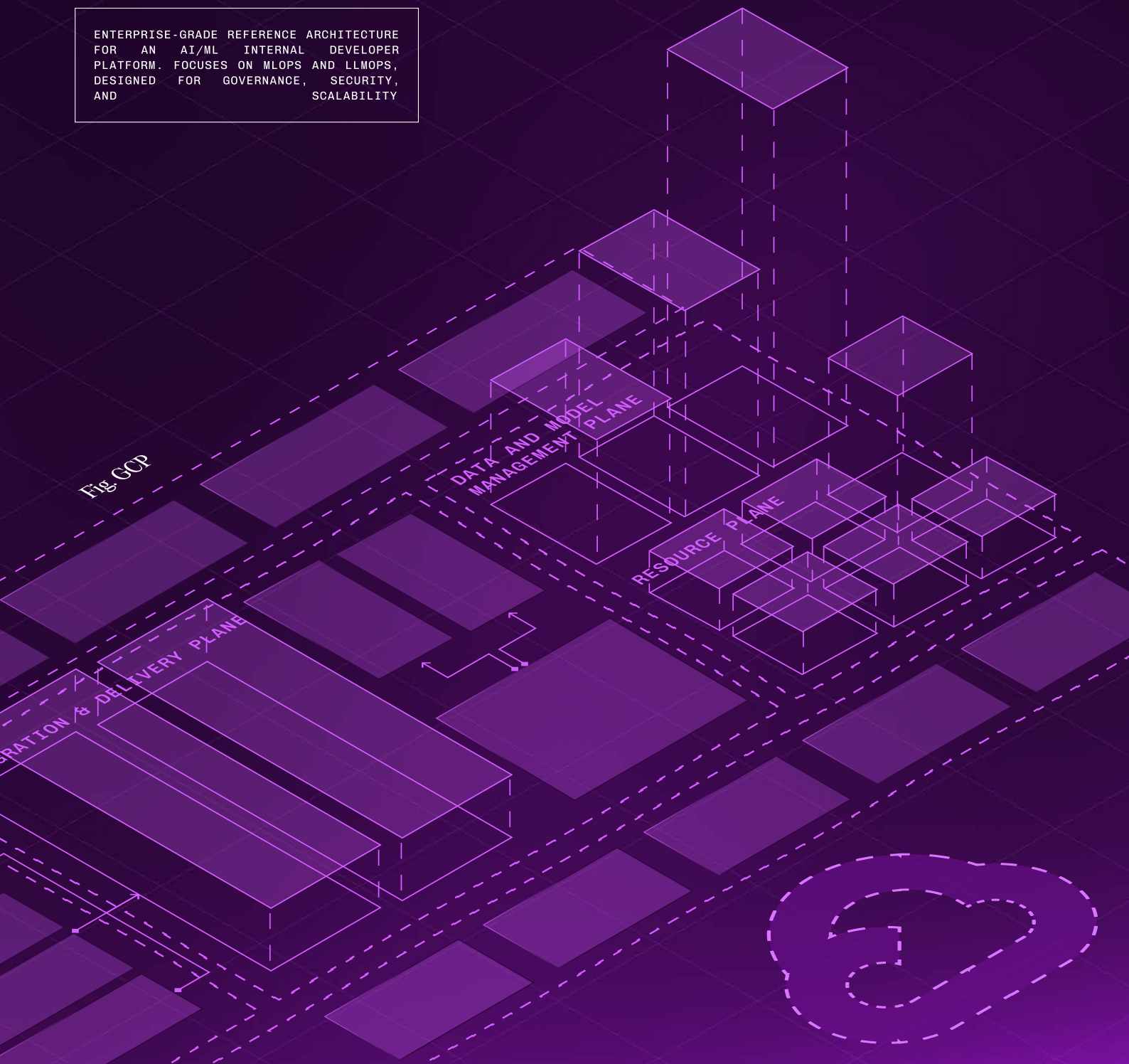




# REFERENCE ARCHITECTURE FOR AN AI/ML INTERNAL DEVELOPER PLATFORM ON GCP

ENTERPRISE-GRADE REFERENCE ARCHITECTURE  
FOR AN AI/ML INTERNAL DEVELOPER  
PLATFORM. FOCUSES ON MLOPS AND LLMOPS,  
DESIGNED FOR GOVERNANCE, SECURITY,  
AND SCALABILITY



# Reference architecture for an AI/ML Internal Developer Platform on GCP

## Table of contents

<b>Executive summary</b>	<b>03</b>		
<hr/>			
<b>Introduction</b>	<b>04</b>		
The enterprise AI/ML challenge	05		
Why IDPs matter for data/AI/ML	08		
<hr/>			
<b>The reference architecture</b>	<b>10</b>		
<hr/>			
<b>Architectural planes in detail</b>	<b>12</b>		
Why planes instead of layers?	13		
The Developer Control Plane	14		
Integration and Delivery Plane	16		
Data and Model Management Plane	18		
Resource Plane	20		
Observability Plane	22		
Security Plane	24		
<hr/>			
<b>Sample use cases &amp; golden paths</b>	<b>26</b>		
Role: Data scientist	27		
Golden path 1: Launching a notebook workspace for exploration	27		
Golden path 2: Log experiment to tracking store	28		
Role: ML engineer	29		
		Golden path 1: Define and deploying a model training pipeline	29
		Golden path 2: Deploying a real-time serving endpoint	30
		Role: Data engineer	31
		Golden path 1: Publish dataset to catalog	31
		Golden path 2: Setting up data transformation workflows	32
		Role: Platform engineer	33
		Golden path 1: Onboarding a new compute resource (e.g., GPU node pool)	33
		Golden path 2: Creating a golden path for training workloads	34
<hr/>			
		<b>Best practices for adoption</b>	<b>35</b>
<hr/>			
		<b>Technology mapping</b>	<b>37</b>
<hr/>			
		<b>Organizational &amp; operational considerations</b>	<b>40</b>
<hr/>			
		<b>Conclusion</b>	<b>42</b>
<hr/>			
		References	44
		About the authors	45



# Executive Summary

We have reached a critical moment in enterprise AI adoption. While the promise of AI and machine learning to transform decision-making, boost efficiency, and create competitive advantage has never been clearer, most organizations find themselves stuck in an all-too-familiar place: unable to scale beyond promising pilots to production-ready, enterprise-wide solutions.

The culprits? A tangled web of challenges spanning data management complexity, evolving model requirements, deployment headaches, monitoring blind spots, and runaway costs. These are all compounded by fragmented adoption patterns, security vulnerabilities, and reproducibility nightmares.

Add to this a persistent talent crunch and the breakneck pace of change in AI/ML infrastructure, and you have a recipe for stalled innovation.

The answer lies in building a purpose-built Internal Developer Platform (IDP) specifically tailored for Data, AI, and ML workloads on a consistent cloud foundation.

This document presents a proposed reference architecture based on current industry best practices. It is crucial to understand that, given the rapid pace of innovation in the AI/ML

domain and the constantly evolving landscape of tools and application domains, this architecture should be viewed as a living foundation rather than a final state.

Think of it as creating a foundation that provides standardized environments, automated workflows, and self-service capabilities, all designed to accelerate innovation while maintaining iron-clad governance and operational efficiency. By standardizing tooling, improving visibility across the development lifecycle, and protecting organizational intellectual property, a specialized IDP aims to tackle the pain points that keep AI initiatives from reaching their full potential.

The payoff for getting this right is substantial. We are talking about dramatic improvements in developer productivity through reduced cognitive load, operational efficiency gains via extensive automation, enhanced scalability with fewer errors, and significantly faster deployment cycles that slash time to market. Most importantly, this specialized IDP creates the bedrock for effective Machine Learning Operations (MLOps) practices. This is the essential ingredient for achieving real-world scale and extracting lasting value from AI investments.





# Introduction

Enterprises are racing to harness AI/ML, yet most struggle to move from experimentation to scalable, secure production. The path is riddled with technical, operational, and organizational hurdles. From governance and cost control to talent gaps and infrastructure complexity, these challenges reveal why many AI ambitions stall before delivering real impact.





# The enterprise AI/ML challenge

Let's face it: enterprises worldwide are struggling to make the leap from AI experimentation to production reality. The journey from proof-of-concept to fully integrated, governed, and scalable enterprise platform is littered with obstacles that many organizations underestimate. Those early POCs, designed for limited experimentation, often buckle under real-world operational demands. What starts as a technical challenge quickly morphs into a complex puzzle involving data management, model complexity, deployment strategies, continuous monitoring, and cost control. Each piece is interdependent and equally critical.

Security, reproducibility, and compliance present particularly thorny challenges. How do you ensure AI/ML initiatives meet stringent security standards, can be consistently reproduced for validation, and comply with ever-evolving regulations? In the rush to innovate, security has too often taken a back seat. This results in weak controls and direct (sometimes unauthorized) access to sensitive data lakes. For large enterprises, regulations like HIPAA and GDPR add another layer of complexity, imposing strict limits on data collection, storage, and processing that require constant vigilance as models evolve and new data emerges.

Then there is the financial reality check. AI/ML workloads come with a hefty price tag, forcing organizations to grapple with transparent cost management, optimize cost ownership, and efficiently manage infrastructure expenses. Balancing investments in computational resources, model training costs, and performance optimization becomes a delicate dance of resource allocation. Core MLOps components, such as cloud storage and high-performance computing, do not come cheap, making meticulous financial stewardship non-negotiable.

The infrastructure landscape itself presents its own challenges. The rapid pace of change and varying maturity levels across tools demand continuous adoption of best practices and harmonization efforts. We see this play out as teams across organizations independently wrestle with similar technology decisions, leading to analysis paralysis and fragmented, siloed adoption patterns. Each team reinvents the wheel, creating inefficiencies and lacking coherence. This unmanaged proliferation of tools and bespoke infrastructure creates a sprawling, inconsistent technology landscape that accumulates AI/ML-specific technical debt. This includes increased maintenance overhead, collaboration friction, and absent standardized security

and compliance. This mounting technical debt does not just slow innovation; it actively impedes it while driving up operational costs. Organizations need a blueprint for dynamic cloud solutions that enable ML workload optimization and on-demand scalability. A hybrid MLOps approach, for instance, has proven effective for cost optimization in large enterprises.

But perhaps the most critical challenge is the human element, which acts as both a bottleneck and an opportunity. The persistent talent shortage demands creative solutions and roadmaps for upskilling existing teams in AI/ML topics. A major stumbling block in AI projects often comes from misalignment among data scientists, IT teams, and business stakeholders regarding objectives and expectations. Employee turnover and scarce specialized expertise can derail machine learning lifecycle delivery. These human factors, including skill gaps, collaboration friction, and misaligned expectations, can stop progress cold. Yet by proactively

addressing them through targeted upskilling and structured frameworks, organizations can unlock tremendous opportunities.

Success in AI/ML is not just about technology; it equally depends on organizational maturity, cross-functional alignment, and continuous talent development. The platform must therefore reduce human-induced friction and foster enhanced collaboration.

The interconnected nature of cost, scalability, and governance becomes clear when you step back. Inefficient resource allocation, often stemming from inadequate governance, directly inflates costs and hampers scalability. Without clear resource usage policies, expenses spiral. Without standardized infrastructure, scaling becomes inefficient. Conversely, inability to scale efficiently drives up unit costs for AI/ML workloads. This dynamic reveals how demands for cost optimization and scalability create powerful incentives for improved governance and standardized platform solutions.

The following table summarizes the key challenges enterprises face in their AI/ML adoption journey:

CHALLENGE CATEGORY	SPECIFIC CHALLENGE DESCRIPTION	SUPPORTING EVIDENCE
<b>Scalability</b>	Transitioning from POC to production	POCs are typically designed for small scale experimentation. May not possess the scalability required for real world operations.
<b>Security &amp; compliance</b>	Security, reproducibility, compliance gaps	Security has been often overlooked in the name of progress, resulting in weak controls and direct access to sensitive data lakes.
<b>Cost management</b>	High cost impact of ML/ AI workloads	Cost management emerges as a critical consideration, with organizations grappling with the financial implications of scaling AI/ ML pipelines.
<b>Infrastructure volatility</b>	High cost impact of ML/ AI workloads	High pace of changes in the ML/Ai infra space, with varying maturity drive need for best practices and harmonization
<b>Talent &amp; alignment</b>	Talent shortage; lack of alignment among stakeholders	A major challenge, in AI projects is the lack of alignment among data scientists IT teams and business stakeholders regarding objectives and expectations.

# Why IDPs matter for data/AI/ML

The explosive growth of AI and ML in enterprises demands we rethink traditional platform approaches. While ML/AI workloads share fundamental challenges with cloud-native development, such as automated deployment, efficient scaling, and reliable operation, they bring their own unique characteristics to the table.

The key differences include specialized infrastructure configurations, a distinct tooling and services landscape, and a much broader user base. This base extends well beyond traditional developers to include data engineers, data scientists, MLOps specialists, platform engineers, and BI analysts. This diverse audience demands a tailored platform experience.

Given these distinctions, Internal Developer Platforms emerge as a compelling solution. However, there is a catch: they must be specifically designed for AI/ML workloads. A generic SDLC-focused IDP simply will not cut it. Traditional IDPs fall short when it comes to complex data pipelines, interactive notebook environments, and robust model governance, all of which are central to the AI/ML lifecycle.

An IDP's core mission is streamlining workflows, reducing cognitive load, and boosting productivity. The fundamental problem IDPs solve

is cognitive overload. This is the mental exhaustion developers face when navigating complex, disparate toolchains and infrastructure. For AI/ML practitioners, this burden multiplies due to the diverse tool landscape, specialized infrastructure requirements (hello, GPUs), and the broader user base we mentioned.

A Data/AI/ML-specific IDP acts as a unifying force. It abstracts away complexity so practitioners can focus on what they do best, such as developing models and deriving insights, rather than wrestling with infrastructure details. We can measure an AI/ML IDP's success directly by how much it reduces cognitive load, which translates to faster iteration cycles and increased innovation velocity. This focus on user experience sets it apart from generic IT platforms.

A well-architected ML IDP becomes instrumental in establishing effective MLOps practices across the organization. MLOps, defined as the practices for deploying and maintaining ML models reliably and efficiently, cannot scale without a robust platform underneath. The IDP provides essential infrastructure, standardized tooling, CI/CD pipelines, model versioning, and monitoring frameworks that enable MLOps principles to work at scale. Without the platform, MLOps remains aspirational; with it,





MLOps becomes operational reality, delivering scalable and reliable ML in production.

One of the most powerful features is the provision of curated “golden paths.” These are opinionated, well-documented, and supported methods for building and deploying software that consistently meet organizational standards. Golden paths dramatically reduce cognitive load and accelerate development by offering clear, templated workflows. In AI/ML, where rapid iteration and experimentation are critical, golden paths provide “paved roads” with built-in security, compliance, and observability. This significantly accelerates experimentation by removing boilerplate setup, while offering the flexibility to “break out”

at any time for novel approaches. Golden paths ensure disparate teams operate within established guardrails, preventing technical debt accumulation and IP fragmentation. They are not merely about efficiency; they are strategic tools for enforcing standards, mitigating risk, and democratizing access to complex AI/ML capabilities, fostering a consistent and secure ecosystem.

Additionally, the IDP empowers developers with self-service tooling, reducing their dependence on operations teams and improving cross-team collaboration. The platform must also provide secure compute environments and robust lineage tracking for both data and models.

# The *reference* architecture

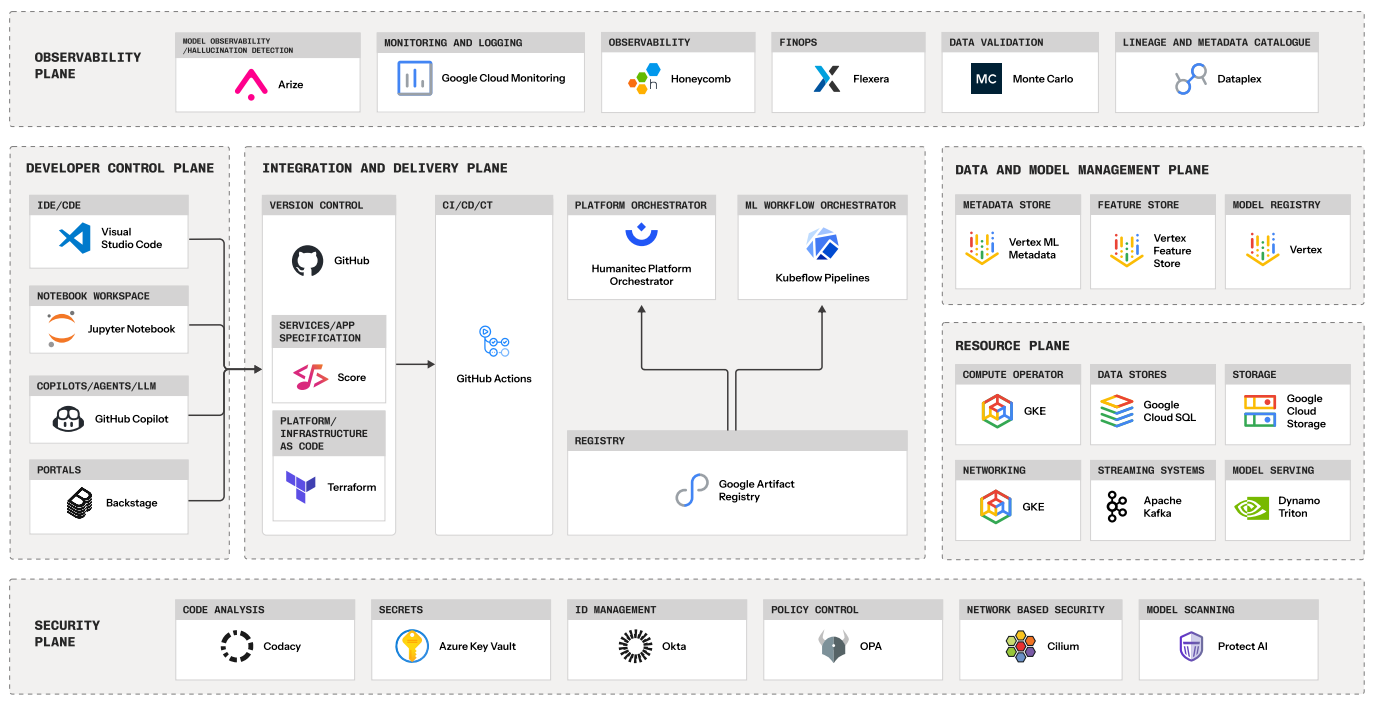
Our proposed reference architecture for an Internal Developer Platform supporting Data, AI, and ML workloads revolves around six distinct planes.

These planes represent logical separations of concerns, each specializing in a critical aspect of the platform's functionality. This design transcends traditional layered approaches, emphasizing a horizontal, cross-cutting perspective where each plane addresses specific concerns impacting the entire data and ML lifecycle.



This document presents a proposed reference architecture for an Internal Developer Platform for Data and AI, specifically leveraging Google Cloud technologies. It is important to note that the architecture is flexible, allowing for the substitution of featured tools with alternatives in the same category, including open-source options.

## REFERENCE ARCHITECTURE FOR DATA/AI INTERNAL DEVELOPER PLATFORM ON GCP



The architecture can be adapted to integrate setups from different cloud vendors as required.

Further reference architectures will be released in the near future.





# Architectural *planes* in detail

Each architectural plane serves a specialized function, contributing to overall efficiency, governance, and scalability of AI/ML workloads. Let's dive into the responsibilities and specific components for each plane as defined in this reference architecture.



# Why planes instead of layers?

Why “planes” instead of conventional “layers”? This distinction matters. Layers often imply strict hierarchy and sequential dependencies, introducing rigidity and bottlenecks in complex, rapidly evolving systems. Planes, by contrast, suggest parallel concerns that intersect and interact dynamically.

This architectural choice inherently addresses the challenges of intertwined functionalities and siloed approaches that plague complex data platforms. By abstracting concerns into distinct planes, we provide a conceptual model for managing the inherent complexity of integrating diverse AI/ML tools and workflows.

This shift from layers to planes is a proactive strategy to prevent future architectural rigidity and technical debt. It acknowledges that AI/ML systems are not linear software stacks but complex, interconnected ecosystems requiring flexible, cross-cutting concerns.

This modular architecture ensures flexibility, scalability, and maintainability. In a modular setup,

components such as data ingestion, storage, processing, and analytics are decoupled. This separation allows independent development, updates, or replacement without disrupting the entire system. The approach promotes component reusability and significantly reduces development time.

The benefits of modular architecture such as separation of concerns, loose coupling between components, high cohesion within components, and standardized interfaces, directly enable the IDP to adapt to rapid changes in the ML/AI infrastructure space. When each plane functions as a cohesive, loosely coupled module, the platform can evolve, integrate new technologies, and respond to changing business needs far more rapidly than monolithic or tightly coupled systems. This enhanced flexibility facilitates faster integration of new tools, increasing organizational agility in the dynamic AI/ML landscape. Plus, this “planes” concept offers seamless extensibility, allowing new planes to be incorporated as organizational needs evolve.

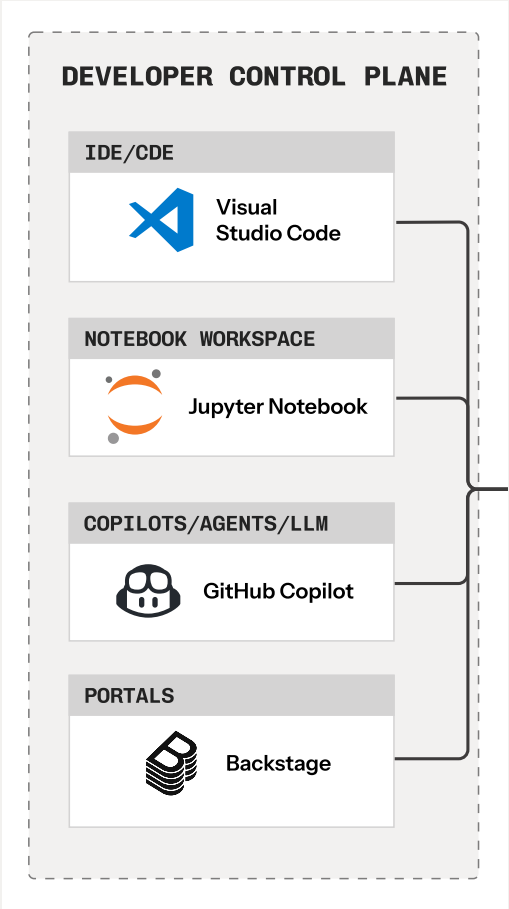




# Developer Control Plane

This plane serves as the front door for all platform users, including Data Scientists, ML Engineers, Data Engineers, and Platform Engineers. It provides intuitive, self-service interfaces for development, experimentation, and platform interaction.

By abstracting underlying infrastructure complexities, this plane dramatically reduces cognitive load on developers and data specialists. Rather than requiring deep infrastructure knowledge or complex manual configurations, users interact with simplified, opinionated interfaces. This directly enables rapid experimentation, reproducibility, and self-service access to compliant resources without manual infrastructure management for Data Scientists.

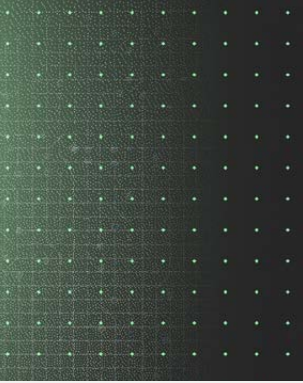


## Developer Control Plane overview

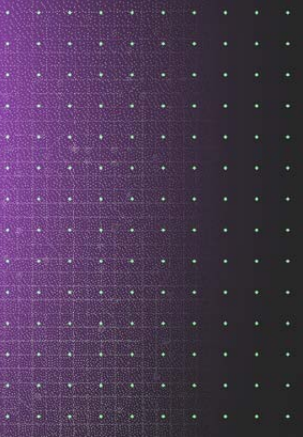
DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Provide user-friendly access points for diverse personas; Enable self-service provisioning and management of resources; Offer integrated development environments, including interactive notebooks and AI agents.	IDE/CDE: Visual Studio Code  Notebook Workspace: Jupyter Notebook  CoPilots/Agents/LLM: Claude Code  Portal: Backstage	Ensuring flexibility for experimentation while accelerating through standardization and governance; Ensuring consistent user experience across multiple interfaces; Managing access control for different tools.







Key components include standard Integrated Development Environments (IDEs) like Visual Studio Code, interactive Notebook Workspaces such as Jupyter Notebook, and AI-powered assistants like Claude Code to accelerate the inner-loop development process.

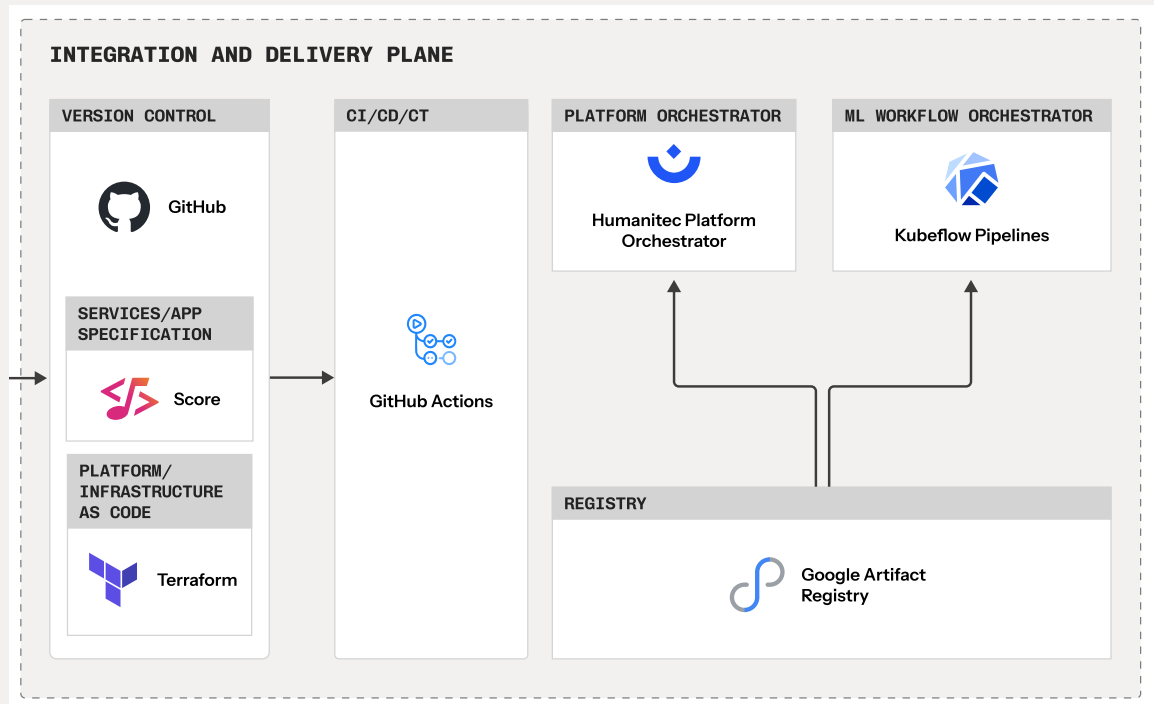


A central Portal, here built on Backstage, provides a “single pane of glass” for resource discovery, documentation, and self-service provisioning. This plane is not just about providing tools. It is about shifting the mental capacity from “how to build and run” to “what to build,” driving developer velocity and satisfaction within the IDP.



# Integration and Delivery Plane

This plane automates the entire lifecycle of data and ML workloads, from code commit to deployment and updates. It encompasses CI, CD, CT (Continuous Testing), and specialized ML pipelines, ensuring changes are validated, artifacts managed, and workloads orchestrated efficiently across the platform.



This architecture specifies a highly-opinionated, modern stack. The flow begins with GitHub as the single source of truth for version control. Score is used to create a platform-agnostic Services/App Specification, defining what the workload is, while Terraform defines the Platform/Infrastructure as Code, or where it will run. GitHub Actions serves as the CI/CD/CT engine, automating the building, testing, and packaging of applications and models.



These workflows feed into two distinct orchestrators. The Humanitec Platform Orchestrator handles platform-level configuration and environment management, translating the developer's intent (from Score) into a running application. Kubeflow Pipelines is used as the specialized ML Workflow Orchestrator, managing the complex, multi-step graphs of training, evaluation, and validation. All resulting container images, models, and packages are stored in Google Artifact Registry as the central, secure Registry.

This plane, particularly the dual-orchestrator model, is where platform governance and automation truly come alive. It bridges the gap between user intent and infrastructure reality while ensuring compliance and efficiency at scale.

### Integration and Delivery Plane overview

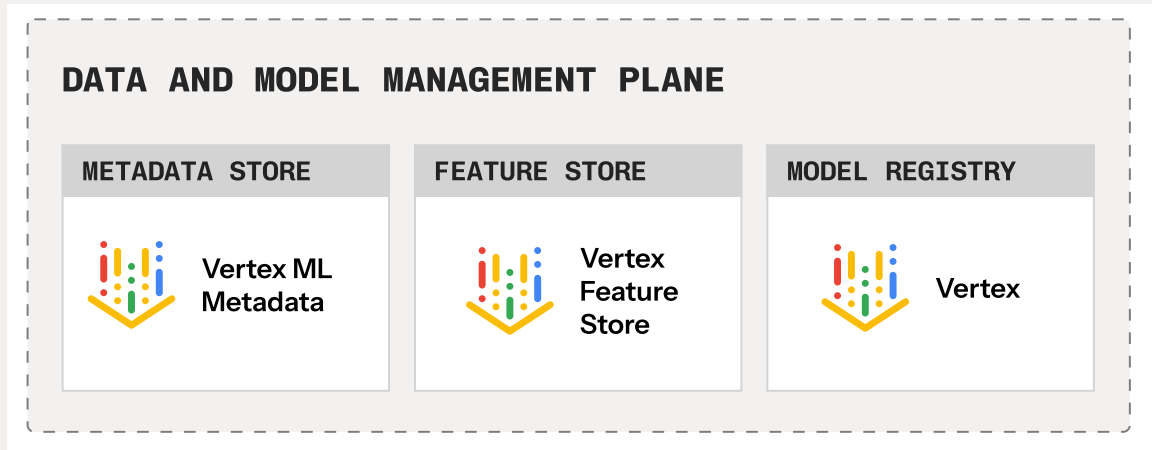
DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Automate build, test, and deployment; Orchestrate complex data, ML, and app workflows; Manage artifacts; Enforce quality gates.	Version Control: GitHub  Services/App Specification: Score  Platform/Infrastructure as Code: Terraform  CI/CD/CT: GitHub Actions  Platform Orchestrator: Humanitec  ML Workflow Orchestrator: Kubeflow Pipelines  Registry: Google Artifact Registry	Integrating diverse ML-specific tools into a unified CI/CD framework; Managing complex dependencies across data, code, and models; Ensuring reproducibility of pipeline runs.





# Data and Model Management Plane

This plane is the heart of data and ML model lifecycle management, spanning from raw ingestion to model serving. It provides managed services for data preparation, feature engineering, experiment tracking, and model versioning.



Vertex AI Metadata (formerly ML Metadata) functions as the central Metadata Store, capturing the lineage and artifacts from all Kubeflow Pipeline runs and other platform activities. Vertex Feature Store serves as the Feature Store, providing a unified source for ML features. This managed service helps address the train-serve skew problem by facilitating the use of consistent feature definitions for both batch training and real-time inference, which is a common challenge with tools like Feast.

For model management, Vertex AI Model Registry is the central repository for all trained models. It goes beyond simple storage, providing deep integration with the GCP ecosystem for versioning, aliasing (e.g., “staging” vs. “production”), and storing “model cards.”



Model cards provide comprehensive information about models, including purpose, performance metrics, training data, ethical considerations, and usage guidelines, offering rich context for governance and reproducibility. This plane also encompasses the training and serving infrastructure, which are dynamically provisioned by the orchestrators and run on the Resource Plane.

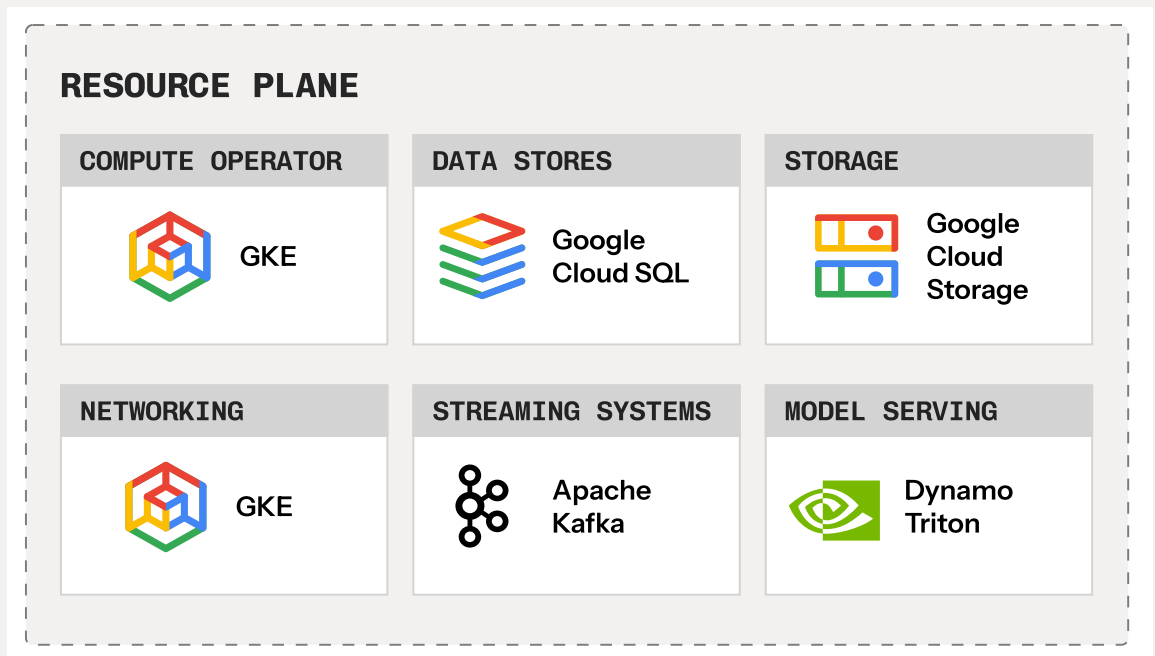
## Data and Model Management Plane overview

DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Provide scalable training infrastructure; Enable comprehensive experiment tracking; Centralize features for consistent training/ inference; Version, store, and govern machine learning models.	Metadata Store: Vertex AI Metadata  Feature Store: Vertex Feature Store  Model Registry: Vertex AI Model Registry	Ensuring data consistency between training and serving; Managing feature versioning and backfilling; Implementing robust data governance and access controls; Maintaining comprehensive model metadata.



# Resource Plane

The Resource Plane provides the computational and storage infrastructure powering all data, AI, and ML workloads. This plane abstracts infrastructure provisioning and management complexities, offering self-service access to diverse compute types and storage solutions.



The core Compute Operator and Networking layer is GKE (Google Kubernetes Engine). GKE provisions and manages all containerized workloads, from data processing pipelines to model serving endpoints, and handles all service-to-service communication and network policies. Google Cloud SQL is designated as the primary relational Data Store for structured data, metadata, and application backends.

For large-scale unstructured data, such as training datasets and model artifacts, Google Cloud Storage is the scalable Storage solution. For real-time data, Apache Kafka is included as the Streaming System for high-throughput, low-latency data ingestion.





Finally, a dedicated Model Serving layer is specified, consisting of Nvidia Triton for high-performance, multi-framework inference and DynamoDB as a low-latency key-value store, potentially for feature lookups or prediction caching at the edge.

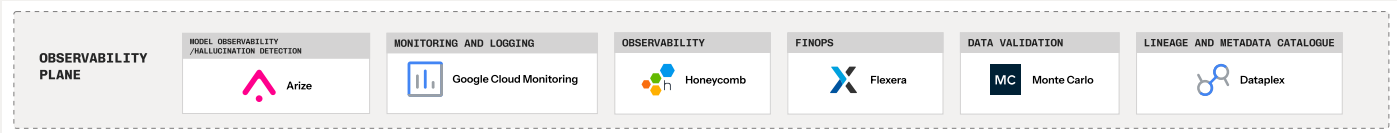
## Resource Plane overview

DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Dynamically provision and manage compute (CPU, GPU); Provide scalable and durable storage; Support real-time data ingestion; Offer high-performance model serving.	Compute Operator: GKE  Data Stores: Google Cloud SQL  Storage: Google Cloud Storage  Networking: GKE  Streaming Systems: Apache Kafka  Model Serving: DynamoDB, Nvidia Triton	Optimizing resource utilization and cost across diverse workloads; Managing heterogeneous compute environments; Ensuring high-throughput access to storage; Scaling streaming and inference infrastructure on demand.



# Observability Plane

This plane provides comprehensive visibility into the health, performance, and behavior of all data, AI, and ML workloads and the underlying platform. It encompasses monitoring, logging, tracing, and cost tracking.



Google Cloud Monitoring is the foundational tool for Monitoring and Logging, collecting logs, metrics, and traces from GKE, Vertex AI, and all other GCP services. For deep, business-context-aware Observability and distributed tracing, Honeycomb is specified, as well as Arize AI for Model Observability. A critical component of any modern platform is financial governance. Flexera is designated as the FinOps solution, providing cost visibility, optimization, and chargeback capabilities.

Beyond traditional system metrics, this plane includes specialized capabilities crucial for AI/ML, though specific tools are not prescribed in this architecture. These include model observability, hallucination detection, and data validation. A critical addition is drift detection.

Mechanisms are needed to identify when models degrade or “drift” from expected behavior as data distributions change. Data quality monitoring with Monte Carlo is equally vital to prevent “garbage in, garbage out” scenarios.

Finally, a Lineage and Metadata Catalogue is required to capture execution logs, metrics, and results, linking them to source models and data lineage for full traceability and auditability.



# Observability Plane overview

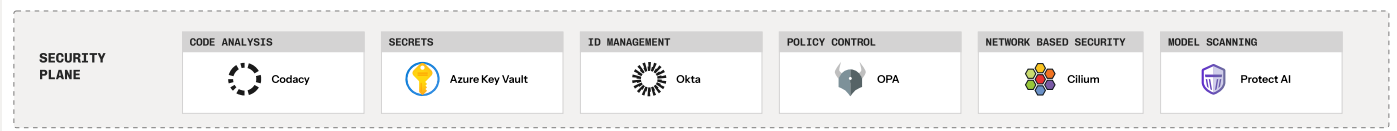
DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Collect and visualize logs, metrics, traces; Monitor system health; Track costs; Detect model drift and data quality issues; Provide data and model lineage.	Monitoring and Logging: Google Cloud Monitoring  Observability: Honeycomb  FinOps: Flexera  Model Observability: Arize AI  Data Validation: Monte Carlo  Lineage and Metadata Catalogue: Dataplex	Optimizing resource utilization and cost across diverse workloads; Managing heterogeneous compute environments; Ensuring high-throughput access to storage; Scaling streaming and inference infrastructure on demand.





# Security Plane

The Security Plane is fundamental to ensuring the integrity, confidentiality, and availability of data, models, and the platform itself. It enforces access controls, manages sensitive credentials, and implements policies protecting against vulnerabilities. This plane integrates security into the CI/CD pipeline and at runtime.



SonarQube is specified for static Code Analysis, scanning code for bugs and vulnerabilities before it is deployed. Google Secrets Manager is the central Secrets store, securely storing and injecting sensitive information, such as API keys, database credentials, and model weights, into workloads at runtime. This prevents hardcoding sensitive data and reduces exposure risk.

Google IAM provides the core ID Management, controlling user and service account permissions across all GCP resources. For fine-grained, in-cluster Policy Control, OPA (Open Policy Agent) is used enabling fine-grained access control and admission policies across the GKE environment. Cilium is used for Network Based Security, providing eBPF-based networking, observability, and security, enforcing network policies at the kernel level.

Finally, Model Scanning capabilities are desirable. This involves automated checks for licensing compliance, potential biases, PII leakage, and security vulnerabilities within model artifacts before production deployment. These automated gates ensure models only proceed to production after passing rigorous security requirements, preventing manual bottlenecks while ensuring governance.



# Security Plane overview

CATEGORY	DESCRIPTION	SPECIFIC TOOL EXAMPLES	KEY CHALLENGES
Responsibilities	Securely manage and inject secrets; Enforce granular access control; Provide secure workload identities; Automate model and code scanning.	Code Analysis: SonarQube  Secrets: Google Secrets Manager  ID Management: Google IAM  Policy Control: OPA (Open Policy Agent)  Network Based Security: Cilium  Model Scanning: Protect AI	Implementing consistent security policies; Managing secrets at scale; Ensuring data privacy and compliance; Detecting and mitigating model-specific security risks (e.g., adversarial attacks).





# Sample use cases & *golden* paths

Golden paths are the secret sauce of an effective Internal Developer Platform. They are opinionated, pre-configured, and automated workflows that guide users through common tasks while reducing cognitive load and ensuring adherence to organizational best practices and policies.

They provide those “paved roads” for data scientists, ML engineers, data engineers, developers and platform engineers, streamlining daily operations.

Let’s explore key golden paths for each persona to see how this IDP transforms their work.





# Data scientist

Data scientists explore datasets, develop statistical and machine learning models, and run experiments.

They rely on curated data access, feature stores, and scalable compute.

The platform empowers them with rapid experimentation, reproducibility, and self-service access to compliant resources.

## Golden path 1: Launching a notebook workspace for exploration

The data scientist needs a secure, GPU-enabled workspace. This flow provisions a Jupyter Notebook with persistent storage and GPU resources - no IT tickets required.

PLANE	WHAT HAPPENS
Developer control	User requests a Jupyter Notebook environment via Backstage portal.
Integration & delivery	Humanitec orchestrator triggers provisioning with standard templates.
Resource	Notebook instance (GKE pod) is created; Google Cloud Storage volumes are attached.
Security	Google Secrets Manager injects credentials; OPA policies enforce isolation.
Observability	Logs and metrics flow to Google Cloud Monitoring for usage tracking.

### Outcome

A ready-to-use notebook spins up within minutes, fully compliant and observable.

### Example tools & components:

Backstage, Humanitec, GKE, Jupyter Notebook, Google Secrets Manager, Google Cloud Monitoring, Google Cloud Storage.



## Golden path 2: Log experiment to tracking store

The data scientist wants to run experiments and log parameters, metrics, and artifacts to the central model registry.

PLANE	WHAT HAPPENS
Developer control	Experiment is launched via notebook, using Vertex AI SDK to log metadata.
Data & model mgt	Metadata (params, metrics) is sent to Vertex AI Metadata. Model artifacts are sent to Vertex AI Model Registry.
Resource	Artifacts are stored in Google Cloud Storage; metadata in Google Cloud SQL (managed by Vertex).
Security	Google IAM permissions restrict access to tracking data per project.
Observability	Experiment runs are visible in the Vertex AI UI, linked to pipeline runs.

### Outcome

All experiments are consistently tracked, enabling reproducibility and model comparison.

### Example tools & components:

Jupyter Notebook, Vertex AI Metadata, Vertex AI Model Registry, Google Cloud Storage, Google Cloud SQL, Google IAM.



# ML engineer

ML engineers scale, optimize, and automate machine learning workflows. They transform experimental code into production-ready pipelines, deploy models, and ensure reliable serving.

## Golden path 1: Define and deploying a model training pipeline

The ML engineer automates the training process using Kubeflow Pipelines.

PLANE	WHAT HAPPENS
Integration & delivery	Pipeline spec (Python) is committed to GitHub. GitHub Actions triggers Kubeflow Pipelines to compile and run the pipeline.
Data & model mgt	The pipeline pulls features from Vertex Feature Store and logs all outputs (metrics, artifacts) to Vertex AI Metadata and Vertex AI Model Registry.
Resource	Kubeflow Pipelines provisions GKE nodes (with GPUs) for each training step.
Security	Google Secrets Manager injects credentials for data access.
Observability	Pipeline execution, logs, and resource usage are monitored in Google Cloud Monitoring and the Kubeflow UI.

### Outcome

A standardized, reproducible training pipeline is executed with full visibility and governance.

### Example tools & components:

Kubeflow Pipelines, GitHub Actions, GKE, Vertex AI (Feature Store, Metadata, Model Registry), Google Secrets Manager, Google Cloud Monitoring.





# Golden path 2: Deploying a real-time serving endpoint

The ML engineer deploys a validated model from the registry as a scalable, real-time API.

PLANE	WHAT HAPPENS
Integration & delivery	A commit to GitHub (e.g., updating a manifest) triggers a GitHub Actions workflow to deploy a model. The model is pulled from Google Artifact Registry.
Resource	GKE provisions a Nvidia Triton inference server. The model artifact is pulled from Vertex AI Model Registry.
Security	The API endpoint is protected by Google IAM. Cilium policies restrict network access to the pod.
Observability	Google Cloud Monitoring tracks latency, throughput, and error rates. Honeycomb traces requests.

### Outcome

A scalable, secure, real-time inference endpoint is deployed, observable, and production-ready.

### Example tools & components:

Nvidia Triton, Vertex AI Model Registry, GKE, GitHub Actions, Google IAM, Cilium, Google Cloud Monitoring, Honeycomb.

# Data engineer

Data engineers design, build, and maintain the pipelines and infrastructure powering analytics and machine learning. They ingest raw data, transform it, and ensure it's accessible and secure.

## Golden path 1: Publish dataset to catalog

The data engineer registers a new dataset in the central metadata store.

PLANE	WHAT HAPPENS
Integration & delivery	A GitHub Actions pipeline is triggered, which executes a script to register the dataset.
Data & model mgt	The dataset's schema, location, and metadata are registered in Vertex AI Metadata.
Resource	The underlying data resides in Google Cloud Storage or Google Cloud SQL.
Security	Google IAM policies are attached to the dataset's metadata definition to govern access.
Observability	The dataset becomes discoverable in the platform's lineage view (via Vertex AI).

### Outcome

The dataset is now discoverable, documented, and governed within the platform.

### Example tools & components:

Vertex AI Metadata, GitHub Actions, Google Cloud Storage, Google Cloud SQL, Google IAM.



# Golden path 2: Setting up data transformation workflows

The data engineer needs to transform raw data from Kafka into an analytics-ready format in Google Cloud SQL.

PLANE	WHAT HAPPENS
Integration & delivery	Transformation logic (e.g., a containerized Python app) is stored in GitHub and registered in Google Artifact Registry. A GitHub Actions workflow deploys it.
Resource	A GKE job is provisioned. It reads from Apache Kafka, performs transformations, and writes the structured data to Google Cloud SQL.
Security	Google Secrets Manager injects credentials for both Kafka and Cloud SQL.
Observability	Job execution logs and run status are captured in Google Cloud Monitoring.

### Outcome

Structured, validated datasets are produced from raw data with repeatable, policy-compliant workflows.

### Example tools & components:

GKE, Apache Kafka, Google Cloud SQL, GitHub Actions, Google Artifact Registry, Google Secrets Manager, Google Cloud Monitoring.





# Platform engineer

Platform engineers build and maintain the Internal Developer Platform, define golden paths, provision environments, and enforce policies.

Their mission is to reduce cognitive load for all other personas.

## Golden path 1: Onboarding a new compute resource (e.g., GPU node pool)

A new GPU-enabled node pool is needed in GKE. The platform engineer integrates it into the IDP.

PLANE	WHAT HAPPENS
Integration & delivery	The platform engineer defines the new node pool using Terraform and commits to GitHub. GitHub Actions applies the change.
Data & model mgt	The Humanitec Platform Orchestrator is updated with rules to allow ML workloads to target this new GPU pool.
Resource	GKE provisions the new GPU-enabled nodes.
Security	OPA policies are updated to define who can consume these expensive GPU resources.
Observability	GPU metrics flow to Google Cloud Monitoring. Flexera begins tracking the cost of the new pool.

### Outcome

GPU-enabled compute becomes available as a self-service resource, governed by policy and tracked for FinOps.

### Example tools & components:

GKE, Terraform, GitHub Actions, Humanitec, OPA, Google Cloud Monitoring, Flexera.



# Golden path 2: Creating a golden path for training workloads

The platform engineer wants to standardize how all teams run training jobs.

PLANE	WHAT HAPPENS
Developer control	A golden path template for Kubeflow Pipelines is published to the Backstage portal.
Integration & delivery	The template uses Score to define the workload and Terraform to define the infrastructure. The Humanitec Orchestrator is configured to wire it all together.
Resource	The template declares required bindings (e.g., GKE compute, Cloud Storage).
Security	Google IAM roles and Google Secrets Manager paths are embedded in the template logic.
Observability	Google Cloud Monitoring dashboards are pre-configured by default for all jobs using this path.

## Outcome

Teams can launch compliant training jobs using a pre-defined, observable, and scalable golden path.

## Example tools & components:

Backstage, Score, Humanitec, Terraform, GKE, Google Secrets Manager, Google Cloud Monitoring, GitHub Actions.



# Best practices *for* adoption

Successfully adopting an Internal Developer Platform for Data/AI/ML workloads requires intentional sequencing and strong foundations. The most successful implementations share several interconnected practices. It starts with golden paths, not platform complexity. Rather than attempting to build an all-purpose platform from day one, organizations should first define and automate their most common, high-impact workflows. These “paved roads” immediately reduce cognitive load, deliver visible value, and build user confidence, creating the momentum needed for broader adoption.



With these early foundations in place, the platform can then expand gradually. Beginning with simple training workloads before progressing to serving and more sophisticated pipelines allows teams to iterate safely and incorporate feedback. This phased rollout mirrors how ML systems evolve in production from training and evaluation to model registration and finally serving, which helps the platform mature organically alongside real user needs.

Security and observability must support this evolution from the very beginning. Treating them as Day 1 priorities ensures compliance, protects sensitive data, and establishes trust across the organization. At the same time, robust observability including logging, monitoring, tracing,

and ML-specific signals like drift detection provides the visibility needed to diagnose issues early and continuously improve. Retrofitting these capabilities later almost always leads to rework, higher risk, and slower progress.

Finally, early alignment on workload definitions ties everything together. Clearly defining what constitutes a workload, whether a training job, inference endpoint, or data transformation pipeline, gives teams a shared mental model. This clarity enables consistent interfaces, policy application, and cost attribution, ensuring that all AI/ML activities integrate cleanly into the platform. Together, these practices form a cohesive blueprint that sets organizations up for long-term platform success.





# Technology *mapping*

The Internal Developer Platform for Data/AI/ML, as specified in this reference architecture, leverages a specific, curated set of technologies from Google Cloud, open-source projects, and best-in-class commercial vendors.





This section maps the prescribed tools to each architectural plane, providing a clear summary of the technology stack.

## Technology mapping for the GCP data/AI IDP

ARCHITECTURAL PLANE	CATEGORY	PRESCRIBED TOOL (S)
Observability Plane	Monitoring and Logging	Google Cloud Monitoring
	Observability	Honeycomb
	FinOps	Flexera
	Model observability	Arize AI
	Data Validation	Monte Carlo
	Lineage and Metadata Catalogue	Dataplex
Developer Control Plane	IDE/CDE	Visual Studio Code
	Notebook Workspace	Jupyter Notebook
	CoPilots/Agents/LLM	Claude Code
	Portal	Backstage
Integration & Delivery Plane	Version Control	GitHub
	Services/App Specification	Score
	Platform/Infrastructure as Code	Terraform
	CI/CD/CT	GitHub Actions
	Platform Orchestrator	Humanitec Platform Orchestrator
	ML Workflow Orchestrator	Kubeflow Pipelines
	Registry	Google Artifact Registry



ARCHITECTURAL PLANE	CATEGORY	PRESCRIBED TOOL (S)
Data and Model Management Plane	Metadata Store	Vertex AI Metadata
	Feature Store	Vertex Feature Store
	Model Registry	Vertex AI Model Registry
Resource Plane	Compute Operator	GKE
	Data Stores	Google Cloud SQL
	Storage	Google Cloud Storage
	Networking	GKE
	Streaming Systems	Apache Kafka
	Model Serving	DynamoDB, Nvidia Triton
Security Plane	Code Analysis	SonarQube
	Secrets	Google Secrets Manager
	ID Management	Google IAM
	Policy Control	OPA (Open Policy Agent)
	Network Based Security	Cilium
	Model Scanning	Protect AI





# Organizational & *operational* considerations

Implementing a comprehensive Internal Developer Platform for Data/AI/ML is not just a technical endeavor; it fundamentally reshapes organizational structures and operational models. Success hinges on clear ownership definitions, a product-centric mindset, and seamless integration with existing enterprise functions. Defining who owns what proves crucial.





Typically, the platform team builds and maintains core IDP infrastructure, defines reusable components, and curates golden paths. The data team focuses on data pipelines, quality, governance, and ensuring accessibility.

The ML Ops team bridges data science and operations, handling model operationalization including deployment, monitoring, and production maintenance. This clear delineation minimizes overlap and fosters specialized expertise.

Adopting a “Platform as a Product” mindset is essential. This means treating the IDP as a product with internal customers: the data scientists, ML engineers, and data engineers. This approach involves continuous feedback loops, iterative development, and focusing on delivering value that reduces cognitive load and accelerates development cycles.

Extending this to a ‘Data as a Product’ mindset further shapes platform capabilities needed to

reduce cognitive load in delivering and managing data products. When data is treated as a product, it becomes discoverable, addressable, trustworthy, and self-describing. This requires platform capabilities that automate these aspects. Finally, integrating with InfoSec, CloudOps, and Compliance is non-negotiable. The IDP cannot operate in isolation.

InfoSec must be involved from design phase to embed security by design, ensuring policy enforcement and vulnerability mitigation. CloudOps teams prove critical for managing underlying cloud infrastructure (in this case, GCP), optimizing costs, and ensuring operational reliability.

Compliance teams ensure all data and model operations adhere to regulatory requirements and internal governance standards. This cross-functional collaboration ensures the IDP is not only technically sound but also secure, compliant, and operationally efficient within the broader enterprise ecosystem.





# Conclusion

The journey toward enterprise-grade AI, Data, and ML capabilities is challenging. Fragmented tooling, high costs, reproducibility gaps, and compliance issues create significant obstacles. This whitepaper has articulated why a tailored Internal Developer Platform, specifically this reference architecture on GCP, is critical for addressing these unique demands.





Our proposed reference architecture, built on distinct functional “planes,” offers a structured and modular approach to building a scalable, secure, and efficient foundation for AI/ML workloads. The “why” is clear: traditional IDPs fall short in supporting the iterative, data-intensive, and specialized nature of AI/ML development. The “what” is a plane-based architecture providing separation of concerns, enabling specialized capabilities from the Developer Control Plane to the fully integrated Vertex AI Data & Model Management Plane, all underpinned by robust Security and Observability. The “how” lies in implementing curated “golden paths” that empower diverse personas like data Scientists, ML engineers, data engineers, and platform engineers through self-service, automated workflows.

This approach accelerates innovation while safeguarding organizational intellectual property and ensuring regulatory adherence. To embark on this transformative journey, start by piloting the reference model, adapting it to your specific enterprise context.

Identify early, high-impact use cases that demonstrate immediate platform value, such as launching secure notebook environments or automating model training pipelines. Simultaneously, prioritize developing initial golden paths for these identified use cases. By focusing on practical, incremental steps, you can progressively build a robust, future-ready IDP that truly unlocks the full potential of your AI, Data, and ML initiatives on Google Cloud Platform.



# References

Manjunath Bhat. [How platform teams can help scale generative AI application delivery](#). PlatformCon 2025.

Kevin Cochrane. [Building AI-native infrastructure with platform engineering](#). Platform Engineering Blog.

Patrick Debois. [Why AI needs a platform team](#). PlatformCon 2025.

Luca Galante. [AI and platform engineering](#). Platform Engineering Blog.

Google Cloud. [What is LLMOps \(large language model operations\)?](#)

IBM. [Abusing MLOps platforms to compromise ML models and enterprise data lakes](#).

MIT CISR. [A product mindset for data](#).

Rakuten SixthSense. [The impact of data observability on machine learning and AI models](#).

S. Sagi. [Scaling generative AI in enterprise IT operations: challenges and opportunities](#). ResearchGate.

F. M. Stürmer. [The developer control plane](#). arXiv.org.

Kaspar von Grünberg. [Why platform engineering will eat the world](#). Platform Engineering Blog.





# About the authors



## *Luca Galante*

Luca Galante is the Core Contributor to the Platform Engineering community, the world's largest platform engineering community with over 200,000 members. He routinely speaks to dozens of engineering teams every month, and summarizes his learnings and takeaways from hundreds of setups into crisp, insightful content for everyone in the industry, from beginner-Ops to cloud experts. He is the host of PlatformCon, the world's largest platform engineering event, and writes to over 100,000 engineers every Friday in his newsletter, Platform Weekly.



## *Dilek Altin*

Dilek is a recognized leader in the European deep-tech ecosystem, specializing in the intersection of Artificial Intelligence, robotics, and secure cloud architectures. With experience ranging from high-performance compute to enterprise SaaS, he develops strategies that connect advanced hardware capabilities with modern cloud-native software stacks in the age of AI.

He has championed platform-engineering principles that position AI as a core part of cloud-native infrastructure, defining reference architectures that secure the software supply chain for machine-learning models and enable engineering teams to run heavy-compute AI workloads on Kubernetes with strong governance, security, and speed.

Dilek previously held leadership roles across strategy, product, and operations in the deep-tech sector. His perspective is informed by earlier work at Intel and McKinsey, where he managed complex technology portfolios. He currently serves on the boards of several deep-tech startups, advising on scaling, fundraising, and market development.





## *Dr. Kessie Francis Kwasi*

Dr. Francis Kessie is a Principal Data Scientist at Fortescue, where he leads a portfolio of AI initiatives encompassing large-scale machine learning development, operational analytics, and enterprise adoption of large language models (LLMs). He has extensive experience architecting and productionising industrial ML and data platforms across mining, transportation, and other heavy-asset industries.

Previously, Dr. Kessie led the development of IMDEX's AiSwyft™ and Blastdog™ platforms—cloud-native hyperspectral and geoscience analytics systems built on distributed data pipelines and Azure-based MLOps. He also served as Big Data Technical Lead at Hitachi Rail, contributing to the core architecture of the Hitachi HMAX™ platform for high-frequency rail telemetry and predictive analytics.

His technical expertise spans deep learning, classical ML, IoT and sensor analytics, LLM integration, MLOps/LLMOps, and cloud-native data engineering. Holding a PhD in computational genomics, he combines scientific rigor with scalable engineering to deliver high-performance, production-ready AI systems.



## *Muhammad Nouman Shahzad*

Muhammad Nouman Shahzad is a data architecture leader with deep expertise in designing scalable, resilient, and AI-ready data ecosystems. He creates cohesive, interoperable reference architectures that integrate data quality, governance, lineage, observability, and a strong developer experience. Drawing on platform engineering principles, he enables organizations to operate efficient, secure, and developer-friendly data platforms that accelerate ML and AI adoption and support the high-velocity delivery of trustworthy data products.

