[DISTRIBUTIONAL]

Distributional's Platform

Intro

Distributional has updated and continues to update our core functionality.

See docs.dbnl.com for the latest on our data pipeline, analytics workflow, and enterprise platform.

Production AI behavior is a black box, which leaves AI teams struggling to improve, fix, and scale their products. Distributional's adaptive analytics platform empowers these teams to have confidence in AI behavior—the interplay and correlations between users, context, tools, models, and metrics. In short, the product (DBNL) transforms traces into insights. To make these insights actionable, DBNL also provides a detailed snapshot of aggregate AI product behavior, regularly surfaces behavioral signals, links evidence to facilitate deep root cause analysis, and enables teams to track relevant signals over time. This empowers AI teams to better understand the behavior of their users and AI products so that they can fix and improve those products with confidence.

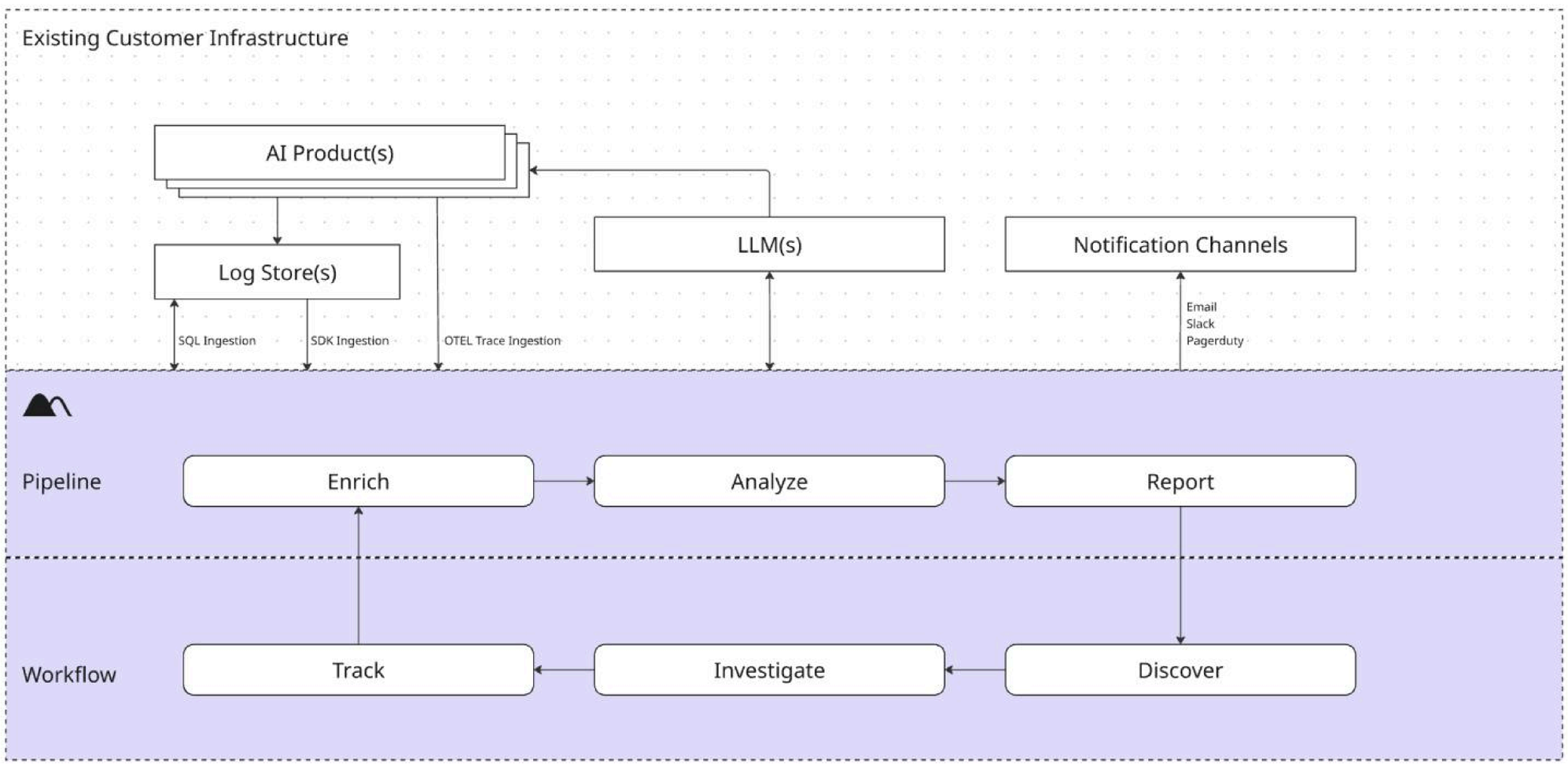
Distributional's adaptive analytics product includes an analytics workflow that is supported by a data pipeline and powered by an enterprise platform. The focus of this paper is the enterprise platform that enables you to deploy, manage, scale, and integrate our product seamlessly with your stack.

Table Of Contents

Enterprise Platform

Distributional's platform includes deployment, architecture, integration, and management features designed for enterprise requirements and production AI scale. Distributional's platform was designed to easily integrate with existing tools and workflows, and to scale alongside AI teams and their projects. The architecture is intentionally streamlined to result in efficiencies and simplified management in the long run.

If you are a team running AI in production at scale that is struggling to understand usage patterns and has strict requirements on data privacy and security, Distributional is designed for your circumstances. This platform makes it possible for any team to use Distributional's Data Pipeline and analytics workflow in a secure, safe, and scalable environment.

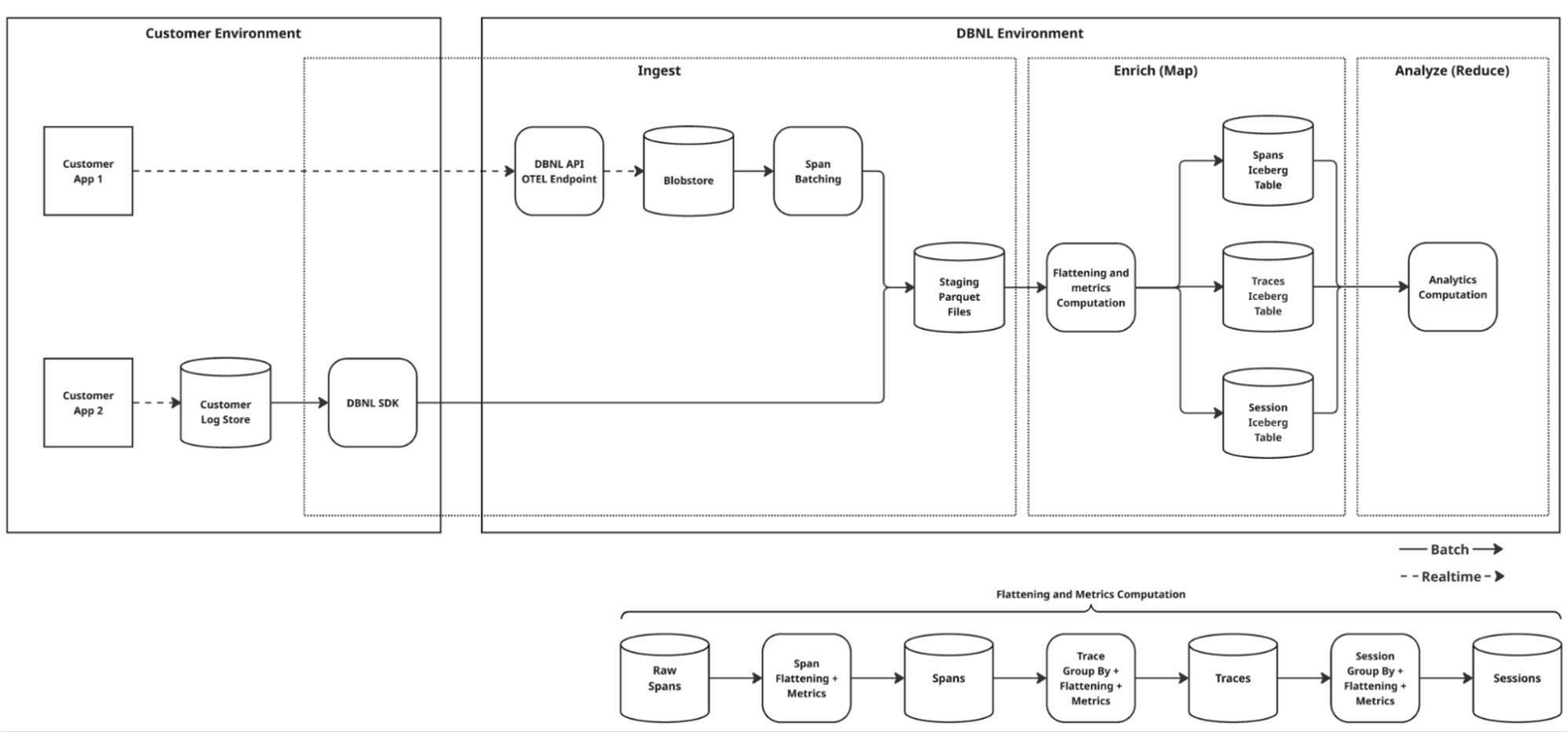


Data Model

Before diving into how the Distributional platform is designed, it's important to understand some key terms related to how Distributional implements adaptive analytics for production AI. These help to break down the types of computation in the system done to create a statistical representation of behavior for production AI.

SETUP

Distributional deploys a Data Pipeline that analyzes your production AI logs. First, Distributional ingests production AI logs with one of three methods: SDK, SQL, or OTEL. Second, Distributional enriches production AI logs with LLM-as-judge, NLP, and other behavioral Metrics. Third, Distributional analyzes these enriched production AI logs with various unsupervised learning and statistical techniques to uncover behavioral signals. Finally, this pipeline publishes insights, signals, and results to dashboards and channels.



CONCEPTS

Columns

A single Log represents production AI behavior. As part of the ingestion step, Distributional requires you to flatten each log into multiple Columns. The only required Columns for a given Log are **INPUT**, **OUTPUT**, and **TIMESTAMP**, but we recommend you include the full span from a trace and apply the DBNL semantic convention.

Insights

A DBNL Insight is a human readable explanation and quantification of behavioral signals generated from unsupervised analysis of enriched Logs as part of the Data Pipeline. Insights represent signals to investigate with the Explorer and a curated sample of Logs. Insights can relate to temporal change, interesting Segments, or newly discovered outliers. DBNL is designed to make it easy to track relevant insights as Metrics or Segments to confirm improvements and fixes over time.

Metrics

A Metric is a mapping from Columns into meaningful numeric values representing cost, quality, performance, or other behavioral characteristics. DBNL computes Metrics daily as part of an enrichment step that makes deeper analysis of these Logs possible. DBNL comes with pre-built LLM-as-judge and standard Metrics. DBNL also makes it easy to edit one of these templates to produce a custom Metric, or apply a function with our query language. These Metrics appear in the Logs, Explorer, and Dashboard in DBNL.

Segments

Segments are saved filters on Logs corresponding to a specific behavioral signal discovered manually or from an DBNL Insight. All Segments are computed and published to the Segments Dashboard. When a new Segment is saved, the DBNL Data Pipeline incorporates that into future analysis as a signal this is a meaningful bifurcation of the data.

Semantic convention

DBNL ingests data using traces produced using telemetry frameworks with different semantic conventions as well as tabular Logs with a user defined format. To compute Metrics and derive Insights consistently across different data ingestion formats, we define a semantic convention for the data as stored within DBNL. If you are using OTEL Trace Ingestion and the OpenInference semantic convention, this happens automatically. If you are using SDK Log Ingestion or SQL Integration Ingestion, you need to ensure that your Column names adhere to our semantic convention for best results.

Deployment

The full Distributional service is available for free with an open distribution model to facilitate adoption by any team that needs AI product analytics. There are a few ways to deploy Distributional:

- SaaS: DBNL offers free, hosted single-user SaaS with a free LLM endpoint to power analysis of production AI traces. This service is best for teams comfortable sending their data to our multi-tenant SaaS and wanting to get started quickly and cheaply with DBNL.
- Sandbox: The DBNL sandbox deployment bundles all of the DBNL services and dependencies into a single self-contained Docker container for quick proof of concepts.
- Helm Chart: The full DBNL platform can be deployed using a Helm chart to existing infrastructure provisioned by the customer.
- Terraform Module: The full DBNL platform can be deployed using a Terraform module on infrastructure provisioned by the module alongside the platform. This option is supported on AWS, GCP, and Azure.

And here is a comparison of these options:

Deployment Type	Pros	Cons
SaaS ↗ Quick, cost effective access to hosted SaaS	• Fast and easy way to use DBNL • Free access to DBNL and an LLM endpoint for analysis • Add a data connection to get started	• Passes data to DBNL servers • Single user only, contact us ↗ for user permission management • Access to the free LLM endpoint for analysis may be limited at certain scale
Sandbox Quick, self contained proof of concept deployments	• Fast and easy way to start exploring platform • Self contained single Docker container • Can be deployed locally on a laptop	• No enterprise Authentication or Administration • Not designed for production scale
Helm Chart Fully customizable deployments within current infrastructure	• Full, scalable deployment • Most customizable • Reuse existing infrastructure	• Requires more configuration
Terraform Module Independent, full deployments in AWS, GCP, or Azure VPCs	• Full, scalable deployment • Automatically provisions infrastructure with a single Terraform command	• Only currently supported in AWS, GCP, and Azure. • Requires permissions to provision infrastructure

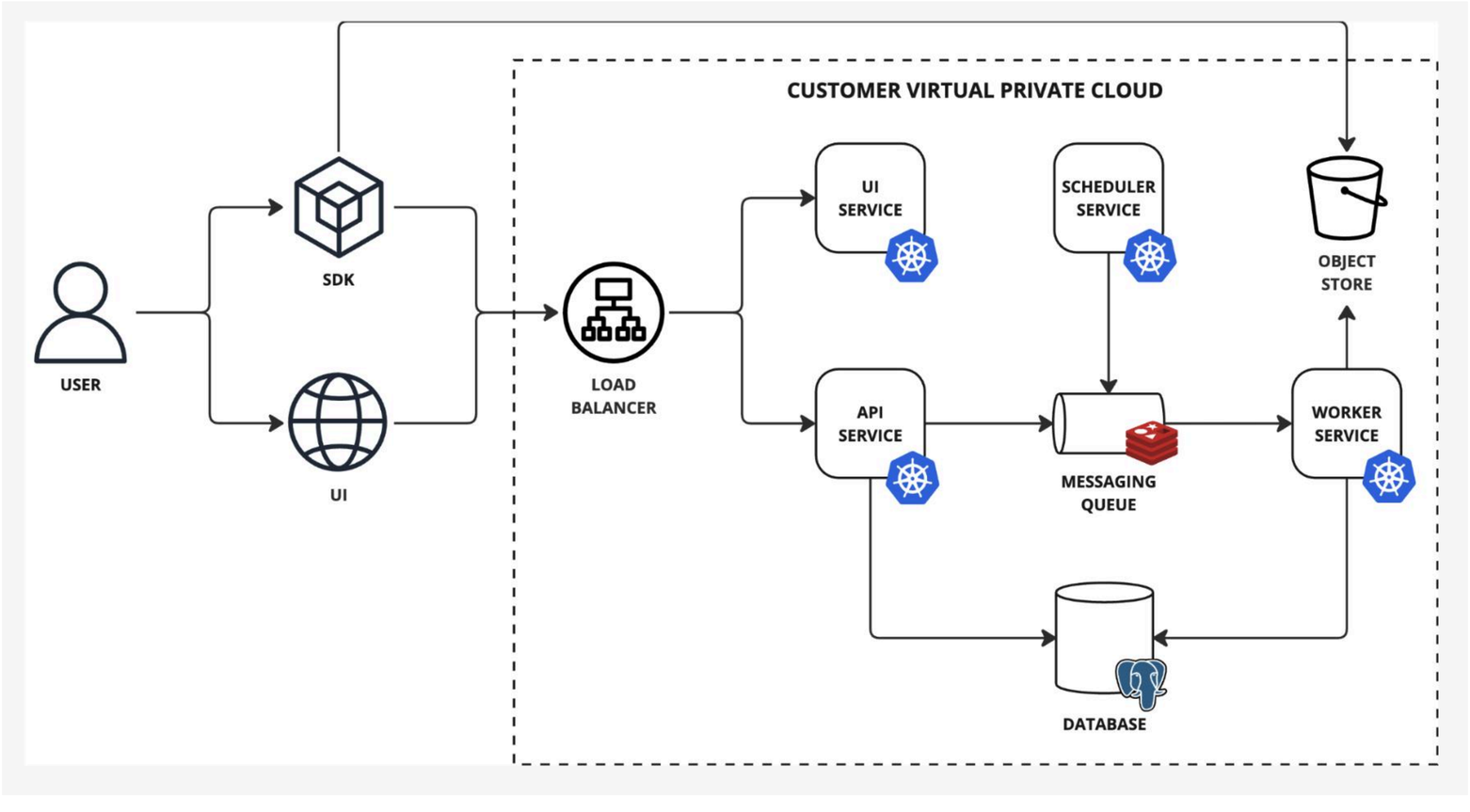
Architecture

To produce relevant Insights, Distributional's platform must be able to handle large, context-rich datasets as well as perform fast, analytic-style processing of that data. Importantly, Distributional's platform does not try to replace customers' existing data storage systems. It is purposefully designed to integrate with existing data infrastructure, where all the raw logs for the AI applications are already stored. Once you establish ingestion based on one of our supported methods, Distributional is then able to efficiently process this data in batch, with inherently no limits on scale. This enables the platform to run much more complex analytics against the data to derive a comprehensive set of insights about AI product behavior as it evolves over time.

Because access to this data is critical, customers deploy and manage the Distributional platform in their private cloud environment. This prevents Distributional from becoming yet another technology silo within a customer's overall architecture and allows the platform to live where the data already resides, preventing duplicate systems of record or concerns about moving data outside of the existing secure environment.

To support this deployment model, we purposefully architected it with as few dependencies as possible and leveraged industry-standard systems to make it as easy to deploy and manage as possible. Distributional's architecture consists of a set of services packaged as Docker images and a set of standard infrastructure components that are deployed into your infrastructure (e.g., a VPC in AWS or on premises on your own infrastructure).

The architecture is designed to be scalable, modular, and self contained. It does not require an external connection to hosted Distributional services to operate.



Integration

The Distributional platform is designed to sit within a customer's existing data infrastructure, rather than be a separate silo to manage and sync. It can easily be deployed in your existing cloud environment and integrate with your existing storage system. This makes it easy to fit seamlessly within your current environment and production AI stack. By leveraging industry-standard components, this also ensures our customers can take advantage of the managed cloud service versions of each for even easier management.

Distributional has few dependencies, runs as a fully contained service, connects with your existing stack, and is designed to produce analysis on your data wherever it is rather than force you to adopt an entirely new data workflow.

Distributional's Data Connections include support for three approaches for DBNL to ingest production AI logs. OTEL Trace Ingestion publishes Open Telemetry traces directly to DBNL as the product runs. SDK Log Ingestion pushes data manually or as part of a daily orchestration job using the Python SDK. SQL Integration Ingestion pulls data from a SQL table into DBNL on a schedule.

Distributional's Model Connections are how DBNL interfaces with the LLMs required for the DBNL Data Pipeline to compute LLM-as-judge Metrics, perform unsupervised analytics, and translate behavioral signals into human readable insights. DBNL supports externally managed LLM endpoints, cloud managed LLM services that are part of your VPC, or locally managed LLMs that are deployed on a cluster. This includes AWS, Azure, Google, and NVIDIA options, as well as any LLM provider compatible with the OpenAI API convention.

Here are the pros and cons of each model connection:

Model Connection Type	Pros	Cons
Externally managed service (together.ai, OpenAI, etc)	• Fast and easy to set up (just provide keys) • Model and scaling flexibility	• Requires sending data outside of your cloud environment • Higher cost, on demand model
Cloud managed service (Bedrock, Vertex, etc)	• Data stays within your cloud provider • Often managed by another team within the organization	• Can be higher cost than locally running models • Usage, rate limits are typically shared across organization
Locally managed deployment (NVIDIA NIMs in k8s cluster)	• Data stays within your local deployment • Cheaper than a managed service • Maximum control of cost vs timing tradeoffs	• Requires access to GPU resources • Can require local admin and debugging

Management

Distributional's full service includes management features to keep data private, secure, and organized to meet strict enterprise standards.

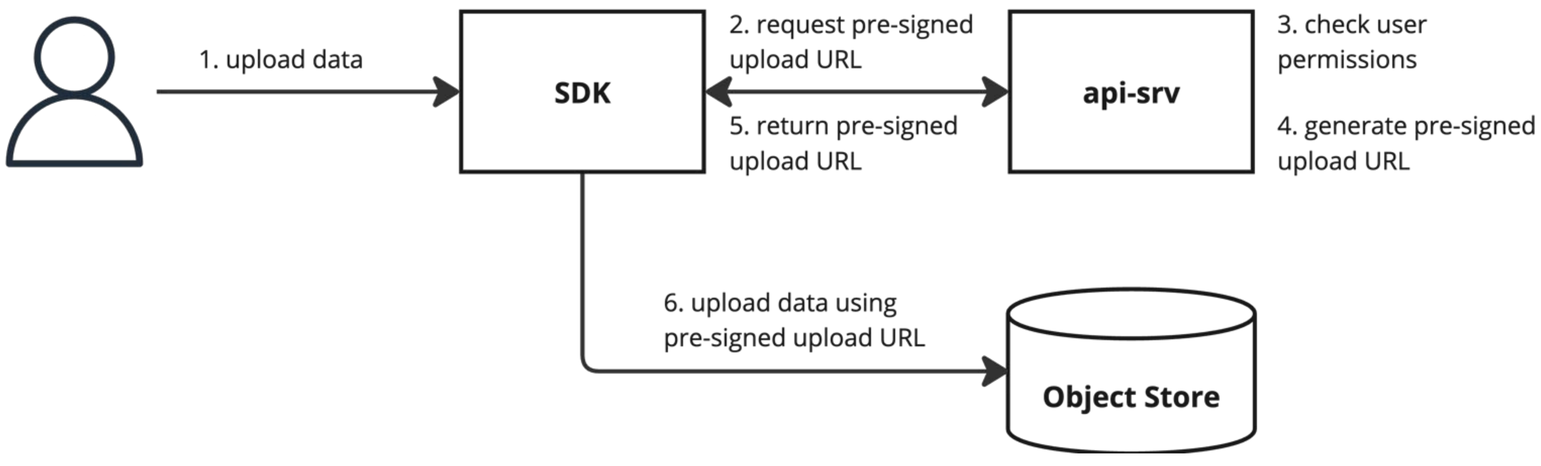
PROJECTS

Projects are the main organizational tool in DBNL. Generally, you'll create one Project for every AI application that you'd like to analyze with DBNL. After a Project is created, you can start analyzing signals from your production AI application. Projects must include a Data Connection and Model Connection (Notification Connections are optional).

DATA SECURITY

Data is secure and does not leave your deployment. A DBNL deployment is self-contained and does not "call home" or send your data back to a hosted cloud service keeping your data safe, secure, and always under your control.

Data is split between databases (e.g., postgres, redis, clickhouse) and an object store (e.g., S3, GCS). Databases contain metadata (e.g. name, schema), aggregate data (e.g., summary statistics, histograms), and raw traces (e.g., for OTEL Trace Ingestion). Obje\ct stores contain raw data (e.g., enriched logs). All data accesses are mediated by the API ensuring the enforcement of access controls.



NETWORKING

A DBNL deployment does not connect back to a hosted external Distributional cloud service. It is designed for enterprise use on potentially sensitive log data that cannot leave the enterprise environment. This comes with a few networking requirements. It needs to be hosted on a customer managed domain or subdomain (e.g. dbnl-example.com or dbnl.example.com). It is recommended that the DBNL platform be served over HTTPS. Support for SSL termination at the load balancer is included. It cannot run in an air-gapped environment and requires a few URLs to be accessible via egress: an artifacts registry, object store, and OIDC token validation (if using a third party provider).

Distributional uses personal access tokens for API authentication, OpenID connect or OIDC for user authentication, and configuration options depending on OIDC provider. DBNL also includes namespaces and user permissions that can be centrally managed by organization admins from an admin dashboard. And networking requirements are designed to fit best practices for running these services in your environment.

These features ensure data stays secure and private as you scale the number of AI products and AI product teams that use DBNL.

AUTHENTICATION

Distributional uses personal access tokens for API authentication. The DBNL platform uses OpenID Connect or OIDC for user authentication. OIDC providers that are known to work with DBNL include: Auth0, Microsoft Entra ID, and Okta. Configuration options depend on the OIDC provider, but generally include AUDIENCE, CLIENTID, ISSUER, and SCOPES.

ADMINISTRATION

Each DBNL deployment corresponds to a single Organization containing all Namespaces and all Users. A Namespace is a unit of isolation within an Organization containing Projects, Data Connections, Model Connections, and Notification Connections. Users are individuals with a login to an Organization and are defined by Roles related to the Organization and one or more Namespaces. Users can be created from the Organization or Namespace Admin Dashboard. Users can have Organization Admin, Namespace Admin, or Namespace Writer roles. Namespaces, Users, and Roles can be organized by Organization Admins from the Admin Dashboard.

Scale With Distributional

With Distributional's adaptive analytics platform, customers can complete the AI product feedback cycle. Empowered by continuous insights from DBNL, teams can understand usage patterns of their AI products, and use this information to improve and fix their products over time. This analytics workflow is powered by a Data Pipeline, which is supported by ourt enterprise platform. This platform deploys DBNLl into your environment and integrates with your stack, ensuring that analytics fits cleanly into the existing production AI stack.

Learn more

- docs.dbnl.com
- distributional.com/blog

[DISTRIBUTIONAL]

About Distributional

Distributional's platform is designed to give customers complete control and optionality over their data and integrations. The platform can adapt to specific security, privacy, and regulatory needs, while still providing the quality and insights necessary to test AI applications.

If you'd like to learn more, reach out to the Distributional team. We're here to help!

Learn more at distributional.com

Follow us on:

- LinkedIn
- YouTube