DevFest

GDG Bandung

Observability for GenAI Apps: Logging, Metrics, & Tracing on GCP

Ananda Dwi Rahmawati
Google Developer Expert - Cloud

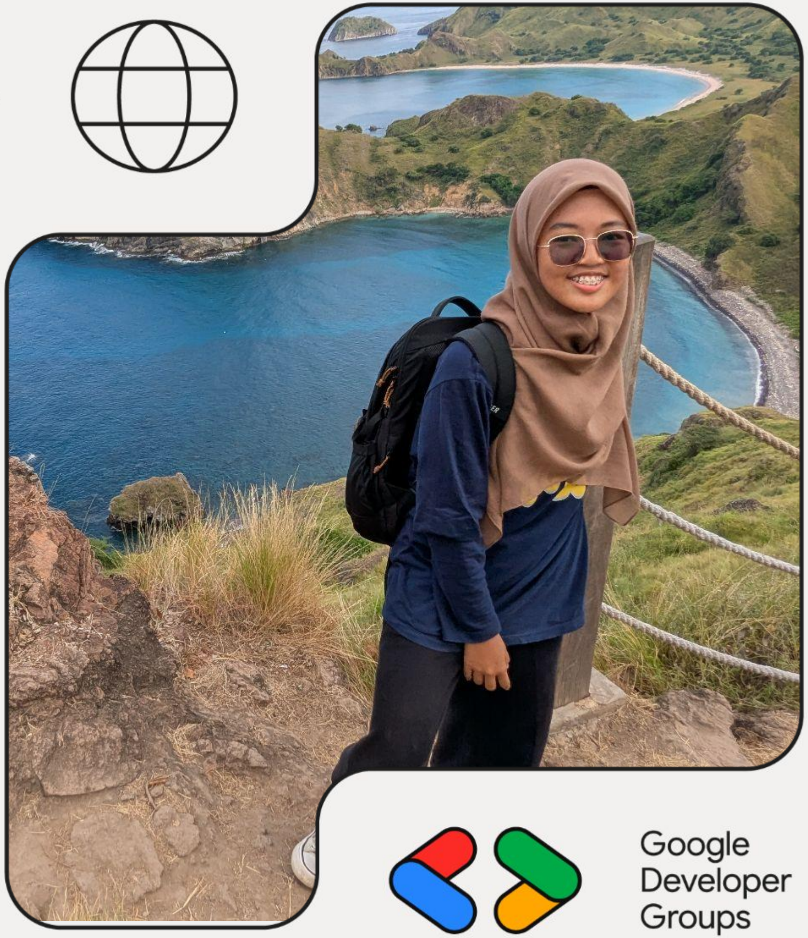
Google Developer Groups

{ DevFest }

GDG Bandung

Ananda Dwi Rahmawati

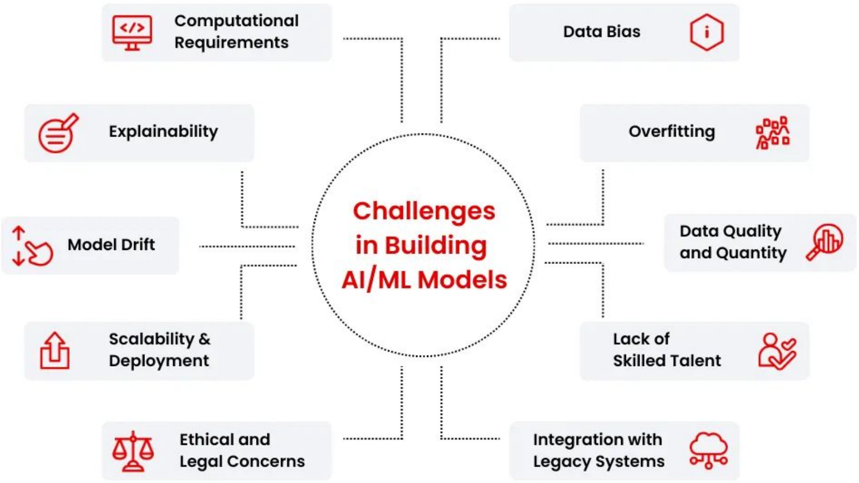
- Cloud & DevOps Engineer, Singapore
- Google Developer Expert Cloud - Modern Architecture
- Master of Computer Science - University of Texas at Austin
- https://linktr.ee/misskecupbung



Google Developer Groups

Operating ML models, presents several challenges:

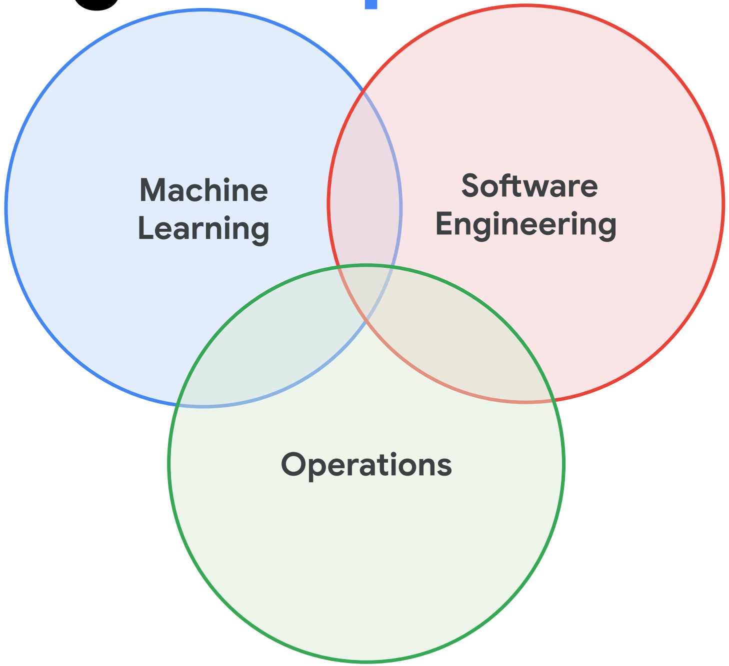
- **Model drift**: As real-world data changes, models become less accurate, requiring frequent retraining.
- **Resource management**: ML workloads have varying demands, making efficient allocation crucial.
- **Data quality**: Consistent, reliable input data is essential for model performance.
- **Compliance**: Meeting governance and regulatory requirements is challenging.
- **Versioning**: Tracking models, datasets, and experiments is difficult at scale.



Google Developer Groups

Introducing MLOps

- Model development
- Model evaluation
- Parameter tuning

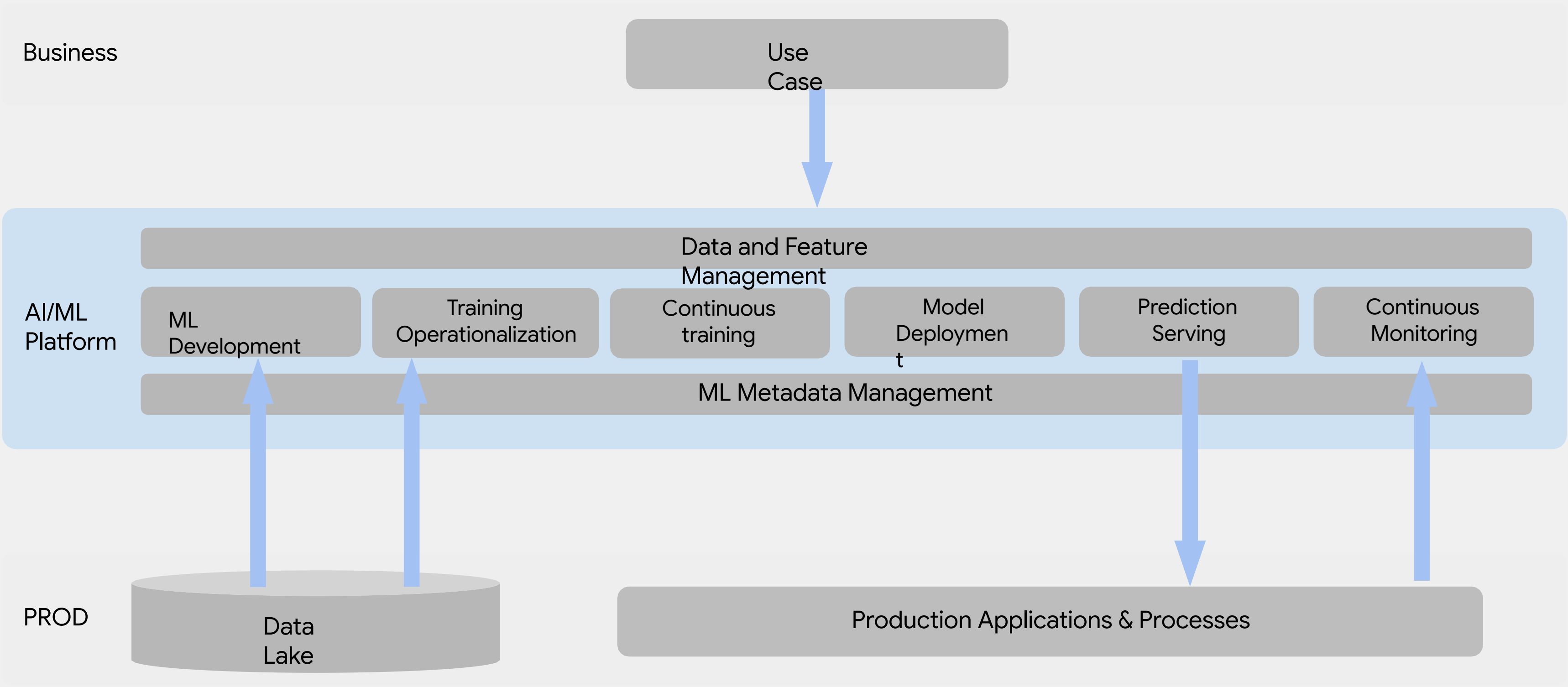


- Data engineering
- Pipeline development
- Integration of model into business application

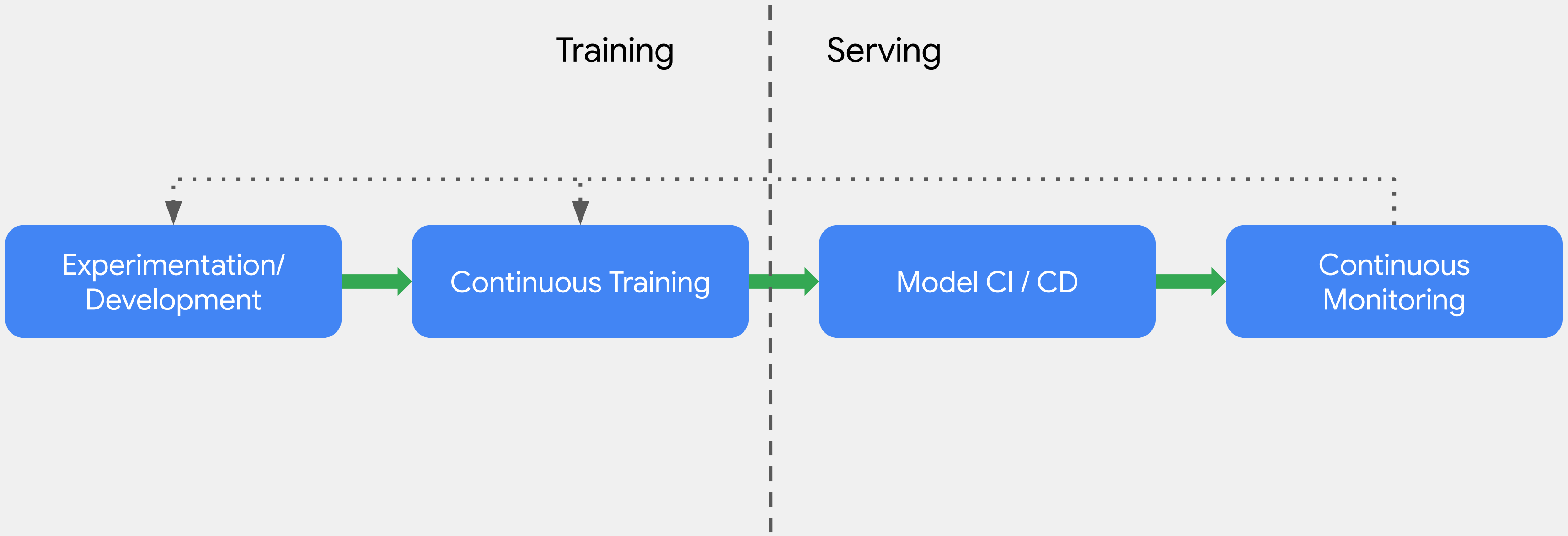
- Model deployment
- Metadata management
- Logging and monitoring

Google Developer Groups

MLOps: quick recap

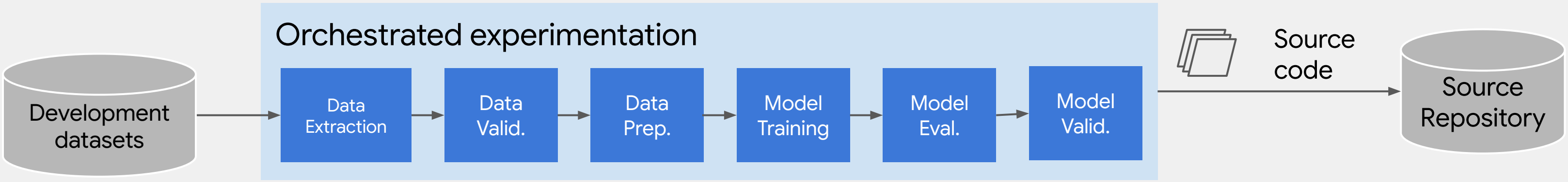


ML Solution Lifecycle



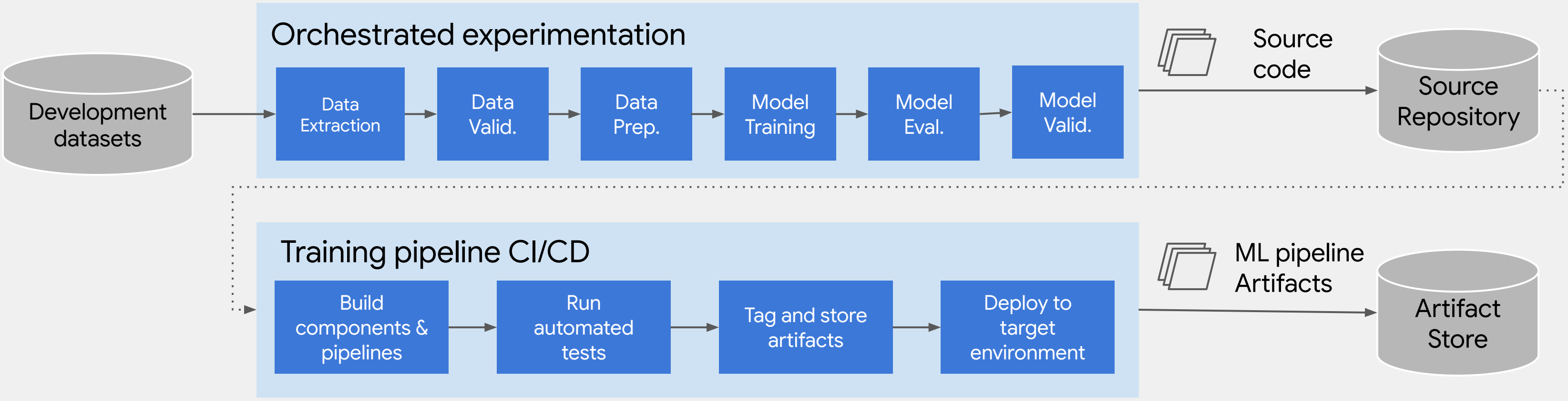
Reliable and repeatable training

Automated E2E Pipelines



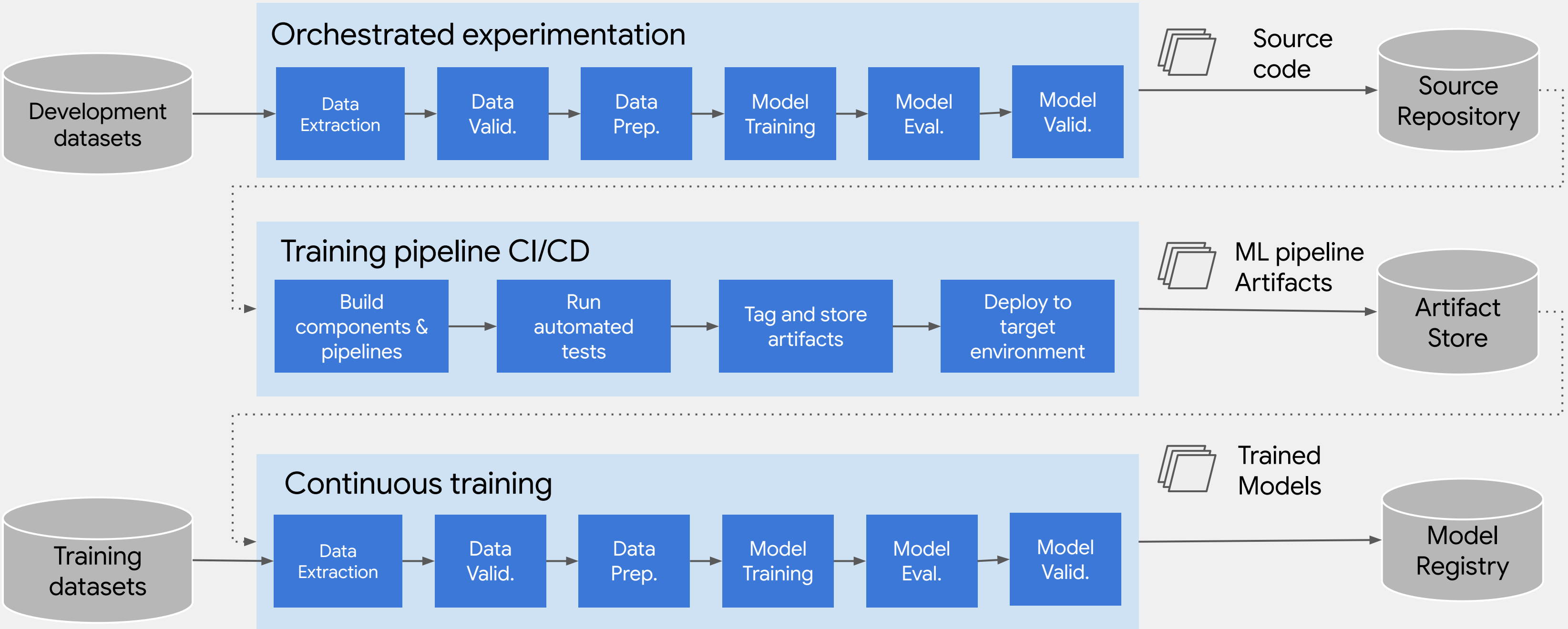
Reliable and repeatable training

Automated E2E Pipelines



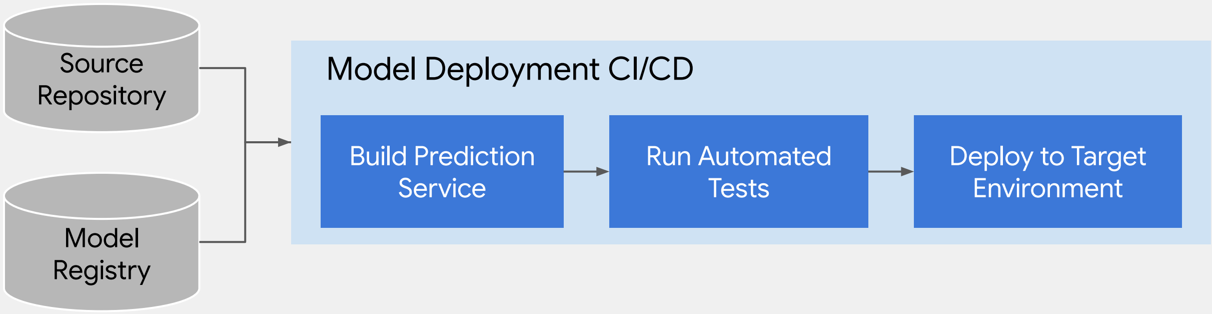
Reliable and repeatable training

Automated E2E Pipelines



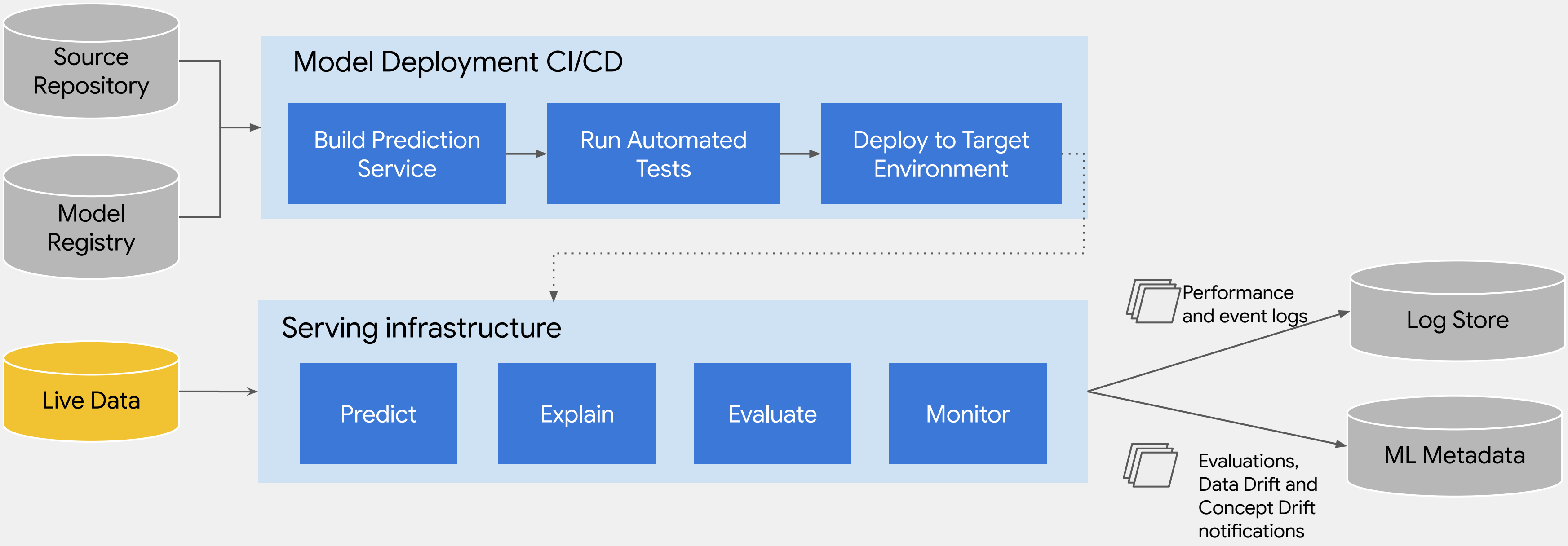
Reliable and monitored serving

Automated E2E Pipelines



Reliable and monitored serving

Automated E2E Pipelines



DevFest

GDG Bandung

"In control theory, **observability** is a measure of **how well** internal states of a system can be inferred from knowledge of its external outputs."

{ DevFest }
GDG Bandung

"**Monitoring** tells you whether a system is working; **Observability** lets you understand why isn't working."

Unique ML Characteristics

Resource Patterns:

- Sustained high GPU usage during training vs consistent CPU usage in traditional apps
- Specialized GPU node scheduling vs typical short-lived batch jobs
- Variable computational demands requiring dynamic resource allocation

Monitoring Focus:

- **Model-specific metrics**: Accuracy, F1 scores (irrelevant for standard applications)
- **Data drift monitoring:** Track shifts in user preferences and data patterns
- **Continuous feedback loops:** Analyze interactions for targeted improvements
- **Granular observations**: Sometimes per-prediction monitoring vs standard application metrics

Google Developer Groups

Observability

Observability = **gaining insights into ML model behavior & infrastructure**.

Enables Teams to:

- Quickly identify and diagnose issues
- Optimize resource usage
- Ensure compliance
- Monitor model performance and detect drift
- Track data quality and integrity

Feedback Loop:

- Continuous monitoring and retraining using real-world data
- Helps models adapt to user behavior, new data patterns, and emerging trends
- Drives better decision-making, user experience, and business value

Google
Developer
Groups

Unique ML Characteristics

Resource Patterns:

- Sustained high GPU usage during training vs consistent CPU usage in traditional apps
- Specialized GPU node scheduling vs typical short-lived batch jobs
- Variable computational demands requiring dynamic resource allocation

Monitoring Focus:

- **Model-specific metrics**: Accuracy, F1 scores (irrelevant for standard applications)
- **Data drift monitoring:** Track shifts in user preferences and data patterns
- **Continuous feedback loops:** Analyze interactions for targeted improvements
- **Granular observations**: Sometimes per-prediction monitoring vs standard application metrics

The Need for ML Observability in MLOps

Pillar	What It Covers	GCP Services / Methods
Data Quality	Detect schema mismatches, cardinality shifts, out-of-range values; track distribution drift in features.	**Vertex AI Feature Monitoring** for feature drift & anomalies; batch & streaming ingestion via **BigQuery**, **Dataflow**, or **Cloud Storage**; establish baseline datasets with **Vertex AI Data Labeling** and **Data Quality checks**.
Fairness / Bias	Pre- and post-training bias detection; monitor predictions' distribution across sensitive groups.	**Vertex AI Fairness Indicators** to compute fairness metrics across facets; integrate with model evaluation pipelines in **Vertex AI Experiments**.
Explainability	Understand which features drive predictions (global & local); detect unjustified dependencies.	**Vertex AI Explainable AI** using SHAP/LIME-like methods; feature attribution for global & local explanations; visualize top features, heatmaps, and partial dependence plots.
Model Performance / Drift	Monitor model accuracy, recall, F1, etc.; detect concept drift; compare predictions vs ground truth.	**Vertex AI Model Monitoring** for drift detection and performance metrics; optionally use **TensorFlow Data Validation (TFDV)** or open-source libs like **nannyML**.

Google Developer Groups

Observability checks

Concept drift

Monitor changes in Performance metrics

- Model accuracy
- Model score drifts
- Monitor drift between train baseline and production scoring

Bias metrics drift

Monitor changes in bias metrics

- Model fairness

Data drift

Monitor changes in data distribution

Monitor changes in embeddings distribution for images or text

- Monitor feature drifts across tabular (structured) and unstructured (images, video) data

Feature attribution drift

Track changes in feature contribution

- Explainability

Let’s Demo

Google
Developer
Groups

Thank you!

{ DevFest }
GDG Bandung