Data in Motion

Data in Motion – Intelligent Multi-Cloud Storage Orchestrator

- Hackathon: Data in Motion: Building the Future of Intelligent Cloud Storage
- Goal: Intelligent engine to analyze, tier, and move data across on-prem, private, and public cloud
- Handles both bulk objects and real-time access streams for continuous insight generation
- Deliverables: Working prototype + concise technical presentation (this document)

Problem Understanding

- Hybrid / multi-cloud sprawl → data scattered across object stores and regions
- Need to balance latency, cost/GB, and resilience while keeping hot data close to compute
- Manual tiering & migration does not scale with billions of objects and dynamic access patterns
- Real-time streams (logs, telemetry, events) need to be acted on while data is in motion

Solution Overview & Architecture

- Control plane: FastAPI-based orchestration service exposing REST APIs for data placement & migration
- Data plane: S3-compatible object stores (on-prem + multiple cloud endpoints) accessed via boto3 clients
- Streaming plane: Apache Kafka used to simulate continuous file-access events and heat updates
- ML plane: scikit-learn models for tier classification and hot-soon forecasting
- UX plane: Dash-based web dashboard showing object distribution, activity, and optimization actions

Data Management & Tiering Logic

- Each object tracked in a FileMeta table (size, content type, tier, replicas, access counters, latency SLA, etc.)

- Kafka access events continuously update short-term (1h) and daily (24h) access counts per object
- A decaying heat_score combines recency, frequency, and access window statistics into a single ranking metric
- Placement policy engine maps heat_score + SLA + cost/GB to three tiers: hot, warm, cold
- Optimizer formulates placement as a constrained cost–latency problem for global rebalancing

Streaming & Data in Motion

- Synthetic traffic generator publishes access events to Kafka with realistic skew (Zipf-like distribution)
- Async consumer service aggregates events into access_1h and access_24h counters in the database
- Heat computation job decays historical activity and updates per-file heat_score on a sliding time window
- Simulation API can burst traffic for selected objects to demonstrate dynamic tier upgrades
- All streaming activity is observable via Prometheus metrics (event throughput, consumer lag, etc.)

ML-Driven Predictive Insights

- Tier model: Logistic regression predicts probability of an object belonging to the hot tier
- Forecast model: classifier predicts y_hot_soon (will a non-hot object become hot soon?)
- Features: access_1h, access_24h, size_bytes, recency_s, hour_of_day, day_of_week, and historical tier labels
- Training: Offline parquet dataset → train/test split → ROC-AUC, PR-AUC, F1 metrics → serialized models
- Online inference: FastAPI /ml endpoints load models once and serve placement recommendations

Migration, Consistency & Resilience

- MigrationTask queue persists planned moves (src, dst, key, status, attempts) in the database
- Migrator service ensures buckets on both endpoints, copies data, verifies checksums, then switches primary

- Idempotent design ensures safe retries and eventual consistency
- Security checks enforce encryption-only destinations when required
- Chaos & failure injection tests retry and degradation behaviour

Performance & Observability

- Prometheus metrics: placement evaluations, migration_jobs_total, migration_queue_gauge, access events
- Dashboard shows: per-tier object counts, active migrations, heat_score evolution, and simulated burst impact
- Kafka decouples event ingestion from optimization, allowing independent scaling
- Performance experiments vary event rate, object count, and tier thresholds to explore latency vs. cost trade-offs

Scalability & Future Roadmap

- Horizontal scaling: multiple optimizer and migrator workers per region
- Cloud integration roadmap: adapters for AWS S3, Azure Blob, GCP Storage
- Enterprise features: policy-based auto-tiering, cost and carbon-aware placement
- Alerting on SLA breaches, cost spikes, or abnormal access patterns

User Experience & Demo Flow

- Dashboard to inspect objects, tiers, replicas, and locations
- Trigger synthetic traffic and observe live re-tiering
- CLI / API for automation: simulate access and request explanations
- Explainability endpoint returns reason for tier placement
- Containerized deployment via Docker & docker-compose (API, Kafka, dashboard, Prometheus)

Conclusion

- Data in Motion platform = intelligent control plane for multi-cloud object placement and data in flight
- Unifies streaming, ML-based prediction, optimization, and migration into one system
- Balances cost, latency, and resilience while remaining cloud-agnostic

- Next steps: integrate real cloud APIs, add richer ML models, plug into NetApp ecosystems
- Demo: docker-compose up, open dashboard, start traffic generator, watch re-tiering