# 140509\_45.md — AI Cost Optimization & FinOps Platform

**Theme:** AI Observability & FinOps for AI  
**Mission:** Provide real-time cost visibility, forecasting, optimization, budgets, and chargebacks for AI workloads across cloud and on-prem infrastructure.

## README (Problem Statement)

**Summary:** Develop a platform that tracks, analyzes, and optimizes costs of AI model development, training, and deployment.  
**Problem Statement:** AI workloads are resource-intensive and costly. The platform should track costs, forecast usage, optimize resources, and provide recommendations while maintaining model performance.

**Steps:**  
- Integrate cloud billing + on-prem metrics  
- Analyze resource utilization  
- Forecast costs  
- Build budget controls + alerts  
- Enable chargeback per team/project  
- Provide ROI analysis

**Suggested Data:** Cloud billing APIs, GPU/CPU usage logs, project metadata, cost-performance benchmarks.

## 1) Vision, Scope, KPIs

**Vision:** Deliver a FinOps platform for AI that gives cost transparency, optimization, and control.  
**Scope:**  
- v1: cost ingestion, dashboards, alerts.  
- v2: forecasting, optimization recs, chargeback.  
- v3: ROI & benchmarking, auto-scaling integration.

**KPIs:**  
- Forecast error <10% MAPE.  
- Cost anomaly detection recall ≥0.9.  
- Identify ≥20% cost savings opportunities.

## 2) Personas & User Stories

* **Data Scientist:** “I need to know if I’m exceeding GPU budgets.”
* **FinOps Manager:** “I need team-level cost reports & chargeback.”
* **CTO:** “I want ROI and efficiency metrics across AI projects.”

**User Stories:**  
- US-01: “As a DS, I want alerts if my training exceeds budget.”  
- US-05: “As FinOps, I want per-team dashboards.”  
- US-10: “As CTO, I want ROI metrics across projects.”

## 3) PRD

**Capabilities:**  
1. **Ingestion:** billing APIs (AWS CUR, GCP BigQuery billing, Azure), k8s usage, on-prem metrics.  
2. **Normalization:** map to unified schema {project, team, resource, usage, cost}.  
3. **Dashboards:** by team, project, service, region.  
4. **Forecasting:** time-series models (Prophet, LSTM).  
5. **Optimization:** right-sizing, spot/preemptible recs, scheduling, quantization, checkpointing.  
6. **Budget Mgmt:** set thresholds, alerting.  
7. **Chargeback:** allocate costs to teams/projects.  
8. **ROI:** efficiency metrics (cost per model/performance).

## 4) FRD

* **Collectors:** billing API connectors, k8s metrics, Prometheus exporters.
* **ETL:** batch + streaming, store in cost lake.
* **Forecast Engine:** Prophet, LSTM; expose via API.
* **Optimizer:** heuristics + ML; e.g., “GPU idle >30% for 2h → downsize.”
* **Budget/Alerts:** rules in Prometheus Alertmanager/CloudWatch.
* **Dashboards:** Grafana, cost explorer UI.
* **Chargeback:** cost allocation engine; export CSV/JSON.

## 5) NFRD

* **Scale:** 10k+ resources.
* **Accuracy:** ±2% reconciliation vs cloud bills.
* **Availability:** 99.9%.
* **Security:** encrypt cost data; role-based views.
* **Compliance:** SOX, ISO 27001.

## 6) Architecture (Logical)

[Cloud Billing APIs] [On-Prem Metrics]  
 | |  
 v v  
 [Collectors] -> [ETL/Normalization] -> [Cost Data Lake]  
 |  
 -----------------------------  
 | | |  
 [Forecast] [Optimizer] [Chargeback]  
 | | |  
 [Dashboards / Alerts / Reports]

## 7) HLD

* **Data Lake:** S3/GCS + Parquet.
* **Forecast:** Prophet (daily/weekly); LSTM (seasonal).
* **Optimizer:** heuristics + ML classification.
* **Dashboards:** Grafana/Kibana.
* **APIs:** REST/GraphQL.

## 8) LLD Examples

**Forecasting:** Prophet on GPU-hours per week.  
**Optimization Rule:** if GPU\_util <30% for >2h → recommend downsize.  
**Chargeback:** map costs by team\_id.

## 9) Pseudocode

costs = ingest(billing, metrics)  
forecast = model.predict(costs)  
alerts = check\_budgets(costs, forecast)  
recs = optimize(costs)  
report = allocate(costs, by\_team)  
return dashboard(forecast, alerts, recs, report)

## 10) Data & Evaluation

* **Data:** billing exports, usage logs, infra metrics.
* **Evaluation:** forecast MAPE, optimization savings identified vs realized.

## 11) Security & Governance

* Role-based access (FinOps vs DS).
* Data encrypted at rest + transit.
* Immutable logs of allocations.

## 12) Observability & Cost

* Metrics: ingestion lag, forecast error, anomaly recalls.
* Cost: serverless ETL, autoscale compute.

## 13) Roadmap

* **M1 (4w):** Ingestion + dashboards.
* **M2 (8w):** Forecast + budget alerts.
* **M3 (12w):** Optimizer + chargeback.
* **M4 (16w):** ROI metrics + automation.

## 14) Risks & Mitigations

* **Forecast errors:** ensembles, recalibration.
* **Data gaps:** reconciliation jobs.
* **Resistance to chargeback:** reports with transparency.