Reading — Deployment & Monitoring (Conceptual Overview)

1. Why Deployment & Monitoring Matter

A model's value emerges only when it's **used**. Deployment turns notebooks into tools; monitoring keeps those tools healthy. Real systems change: data drifts, user behavior shifts, and upstream pipelines evolve. Without monitoring, failure is often **silent**.

2. Deployment Surfaces (Conceptual)

- **Batch jobs**: periodic scoring, good for non-time-critical use cases.
- **Online APIs**: request/response predictions for apps or services.
- **Streaming**: low-latency, stateful decisions (hardest to operate).

Ownership typically spans **financial engineering**, **data science**, **platform/ML engineering**, and **application** teams. Clear **handoffs** prevent gaps.

3. Monitoring Layers & Example Metrics

- **Data**: freshness minutes, row count, %nulls, schema hash/signature.
- **Model**: rolling MAE/AUC, calibration, population stability.
- **System**: p95/p99 latency, error rate, availability, cost/unit.
- **Business**: approval/reject rate, bad-rate, revenue/decision.

4. Feedback Loops & Drift

Labels can arrive late; users may adapt to the model; feature distributions shift. Plan **retraining triggers** and keep a **change log**.

5. Alerting & Runbooks

Start simple: a few thresholds with owners and a one-page runbook. Aim for **fast detection** and **safe rollback** over exotic metrics.

6. Handoff README (What Good Looks Like)

- Endpoints & auth; data contracts; versioning & rollback; monitoring & alerts; contacts & escalation; audit notes.

7. What This Course Expects

No real deployment. You will **design** monitoring & handoff artifacts conceptually and connect them to your project.

8. Checklist

- ☐ Risks → metrics for all four layers

- [ ] Alert thresholds + owners
- [ ] Retraining trigger defined
- [ ] README items drafted
- [ ] Commit notes updated