3.2 BRIDGE TO WATSONX ORCHESTRATE & GOVERNANCE

LEARNING OBJECTIVES

- Understand how notebook patterns translate to watsonx Orchestrate & governance tools.
- Recognize where policies and monitoring fit.

MAPPING ACCELERATOR + AGENT TO ORCHESTRATE

Think of watsonx Orchestrate as a **platform** that can host the patterns you built in notebooks.

- Conceptual mapping:
  - `accelerator/service/api.py` → a reusable “RAG answer” action (tool / service).
  - Agent notebook → blueprint for a multi-step orchestrated workflow.

EXAMPLE ORCHESTRATE FLOW

1. Receive a user request (chat, API, UI).
2. Decide whether to:
   - Call the **RAG service** (accelerator `/ask` endpoint).
   - Call a calculator or other utility tool.
   - Escalate to a **human** (handoff / ticket).
3. Log all steps and outcomes for governance.

In Orchestrate terms:

- **Agent** = top-level orchestrator that plans and delegates.
- **Tools** = your APIs (RAG, calculator, external systems).
- **Connections** = how tools authenticate to external services.
- **Knowledge bases** = RAG-ready corpora defined in Orchestrate.

GOVERNANCE & EVALUATION

watsonx.governance adds a layer of **control and insight** over your agents and models:

- **Model & asset catalog**
  - Track which models, prompts, tools, and agents exist.
  - Versioning and metadata.
- **Policies**
  - Allowed models / endpoints.
  - Risk profiles and approval workflows.
  - For example:
    - “Customer-facing agents may only use models with a given risk rating.”
- **Evaluation Studio**
  - Run experiments on your agents / RAG flows.
  - Compare LLMs, prompts, retrieval strategies.
  - Track metrics like faithfulness, answer relevance, content safety.
- **Monitoring**
  - Track runs and metrics over time.
  - Detect drift or degradation.
  - Investigate failures and edge-cases.

USING EXISTING NOTEBOOKS AS A BRIDGE

You already have governance-related notebooks in `labs-src` and `accelerator`:

- `labs-src/ibm-watsonx-governance-governed-agentic-catalog.ipynb`:
  - Shows how to:
    - Register models & tools in a governed catalog.
    - Create and use governed tools (e.g. PII detectors, jailbreak detectors).
- `labs-src/ibm-watsonx-governance-evaluation-studio-getting-started.ipynb`:
  - Shows how to:
    - Define evaluation datasets.
    - Compute metrics like context relevance, faithfulness, answer relevance.
- `accelerator/assets/notebook/notebook:Analyze_Log_and_Feedback.ipynb`:
  - Concrete example of:
    - Pulling logs from a deployed service.
    - Analyzing quality and user feedback.
    - Bridging logs → evaluation datasets.

Your Day 3 agent logs can be fed into these notebooks as a **first step** towards full governance.

FROM LOGS TO GOVERNANCE

Where do logs come from?

- `rag/pipeline.py`:
  - Can log retrievals and model calls (question, chunks, model, latency).
- `service/api.py`:
  - Can log user requests, status codes, errors.
- Agent notebook (Lab 3.1):
  - Can log tool choices, planner outputs, tool responses, final answers.

These logs can be:

1. Exported periodically as CSV/JSON.
2. Loaded into:
   - Evaluation Studio datasets.
   - Analysis notebooks (e.g. `Analyze_Log_and_Feedback.ipynb`).

Governance workflows can then:

- Detect drift in answer quality.
- Enforce thresholds:
  - "If faithfulness < 0.8, flag for review."
- Support audit trails:

EXAMPLE USE CASES

Some concrete patterns where orchestrate + governance shine:

- **Governed HR assistant**
  - Uses RAG over HR policies.
  - Tools for detecting PII and harmful content.
  - Governed model + tool catalog and metrics.
- **Customer support bot with monitored outputs**
  - Agent routes between FAQ RAG, ticket creation, and human handoff.
  - Governance monitors answer relevance and safety.
- **Internal knowledge bot with evaluation cycles**
  - Weekly evaluation runs on a curated question set.
  - Results feed into prompt/model/index improvements.

HOW TO GO FROM LAB TO PRODUCTION

From Day 3 notebooks to real systems:

- Extract the best patterns from:
  - `agent_watsonx.ipynb` (agent loop + tools).
  - `accelerator` service (RAG microservice).
  - Governance notebooks (evaluation + catalog).
- Embed them into your delivery pipeline:
  - Use `Makefile` / `pyproject.toml` / Dockerfiles to package the accelerator.
  - Deploy accelerator API behind a stable endpoint.
  - Register it as a tool / connection in Orchestrate.
- Connect Orchestrate deployments to governance:
  - Ensure models and agents are registered in the catalog.
  - Route logs to Evaluation Studio or custom analysis.

DISCUSSION PROMPTS

To close the session, consider:

- Where would you use **orchestration** and **governance** in your organization?
- Which components from this workshop are closest to your production needs?
- What would you need to:
  - Onboard your own data?
  - Apply your internal policies?
  - Connect to your existing IT systems?

Capture these ideas — they are great seeds for the **capstone day** and follow-up projects.