CAPSTONE PROJECT IDEAS

HOW TO USE THIS LIST

- Pick one idea or merge aspects from several.
- Adapt scope to fit 3–4 hours of work.
- Feel free to leverage code from:
  - Day 1–3 labs.
  - `labs-src` notebooks.
  - `accelerator` Python modules & notebooks.

1. WORKSHOP FAQ ASSISTANT (RAG OVER THE COURSE)

- **Description**
  - Build an assistant that can answer questions about the workshop itself:
    - Agenda, labs, files, concepts.
- **Required pieces**
  - Corpus = workshop docs, READMEs, and lab guides.
  - RAG in both:
    - Ollama env.
    - watsonx env.
  - Optional: deploy via `accelerator` service.
- **Helpful accelerator files**
  - `rag/retriever.py`, `rag/pipeline.py`, `rag/prompt.py`
  - `tools/chunk.py`, `tools/embed_index.py`
  - `service/api.py`, `ui/app.py`
- **Suggested stretch goals**
  - Side-by-side comparison (Ollama vs watsonx).
  - Evaluation with `tools/eval_small.py`.

2. INTERNAL POLICY / HR HELPER

- **Domain**
  - Synthetic HR/policy docs.
- **Build**
  - RAG with safety-aware prompting and refusal patterns.
- **Use accelerator as**
  - A properly configured microservice for HR policy queries.
- **Helpful reference notebooks**
  - RAG examples in `labs-src`.
  - Governance notebooks for policy checks.
- **Stretch goals**
  - Confidence indicator (e.g. high/medium/low).
  - Refusal or handoff for out-of-scope questions.
  - Governance metrics via Evaluation Studio.

3. RAG DEBUGGER & EVALUATOR

- **Focus**
  - Evaluation rather than a new use case.
- **Build**
  - Harness that compares different RAG settings/backends:
    - `k` values.
    - Embedding models.
    - Retriever strategies.
- **Accelerator integration**
  - Implement `tools/eval_small.py` to call `/ask`.
  - Use `notebook:Analyze_Log_and_Feedback.ipynb` to explore logs.
- **Helpful reference notebooks**
  - RAG notebooks in `labs-src`.
  - Governance evaluation notebook.
- **Stretch goals**
  - Simple dashboards (e.g. Streamlit or notebooks).
  - Export to governance as evaluation datasets.

4. TEAM KNOWLEDGE HUB BOT

- **Corpus**
  - Short articles written by team members about their domains.
- **Build**
  - RAG assistant that attributes answers to authors and links to sources.
- **Use accelerator to**
  - Host the service with a tailored `ui/app.py` Streamlit interface.
- **Helpful reference notebooks**
  - `labs-src/use-watsonx-chroma-and-langchain-to-answer-questions-rag.ipynb`
  - Accelerator ingestion notebooks.
- **Stretch goals**
  - User authentication hooks.
  - Session-aware UI and feedback collection.

5. ORCHESTRATED ASSISTANT WITH TOOLS

- **Build**
  - Agent in watsonx env that chooses between tools:
    - Accelerator RAG API.
    - Calculator.
    - Possibly others (ticket creation, CRM, etc.).
- **Helpful files**
  - `service/api.py`, `service/deps.py` (as the RAG microservice).
  - Agent notebook from Day 3.
  - Governance notebooks (`labs-src`).
- **Stretch goals**
  - Logging suitable for governance.
  - Simple “session view” for conversations and tool calls.

CUSTOM PROJECT IDEAS

Encourage participants to propose:

- Domain-specific assistants (e.g., finance, support, internal engineering docs).
- Integrations with existing datasets or APIs.
- Extensions to accelerator:
  - Additional endpoints (e.g. `/ingest`, `/batch-ask`).
  - Batch scoring APIs.
  - Admin UI for corpus management.

TIPS FOR SUCCESS

- Start from existing lab notebooks and accelerator scripts.
- Aim for a clear end-to-end story: **data → RAG → API/UI → evaluation**.
- Limit scope to something demoable within the time.
- Focus on one or two *new* ideas, not everything at once.