

# CAPSTONE DAY – OVERVIEW

## PURPOSE OF THE CAPSTONE

- Consolidate learning by building a small project.
- Encourage teamwork and creativity.
- Push the **accelerator** closer to production for at least one use case.

# FORMAT & SCHEDULE

Suggested 3–4 hour structure:

- 1. Intro & recap (20–30 min)**
  - Remind everyone of Days 1–3.
  - Clarify expectations and scoring (if any).
- 2. Team formation and project selection (20–30 min)**
  - 2–4 people per team.
  - Pick or adapt a project idea (see project ideas file).
- 3. Build time (2–2.5 h)**
  - Implement a small but end-to-end solution.
- 4. Demos and feedback (30–60 min)**
  - 5–10 min per team.
  - Explain goal, architecture, demo, lessons learned.

# EXPECTATIONS FOR TEAMS

Teams should:

- Use both environments where possible (Ollama + watsonx).
- For a “production-ready” flavour, build on top of:
  - The `accelerator` service (API + tools + UI), and/or
  - RAG & governance notebooks from `labs-src`.

Produce:

- A working notebook or small service.
- A short explanation (slides or markdown) that covers:
  - Problem statement.
  - Architecture (how they used RAG, agents, orchestration).
  - Limitations and next steps.

# ASSESSMENT CRITERIA (OPTIONAL)

If you choose to “score” or rank projects, consider:

- **Clarity of problem statement**
  - Is it clear what the assistant or system is trying to solve?
- **Technical implementation**
  - Does it run end-to-end?
  - Does it use RAG and/or agents in a meaningful way?
- **Use of workshop topics**
  - RAG (Day 2).
  - Orchestration / agents (Day 3).
  - Evaluation / governance (optional but encouraged).
- **Presentation & storytelling**
  - Is the demo easy to follow?
  - Do they reflect on what worked / didn't?

# DELIVERABLES

Each team should provide:

- A repo or folder (or branch) with:
  - Code / notebooks.
  - Instructions to run (README).
- A demo plan:
  - 2–3 main use cases to show.
  - Known limitations.

Optional:

- PRs or patches against `accelerator` (if your org uses a shared repo).

# LINKING TO PROJECT IDEAS

You can reference the **Capstone Project Ideas** page for inspiration.

Highlight projects that explicitly:

- Extend `retriever.py`, `pipeline.py`, `eval_small.py`, `ui/app.py`.
- Combine the two environments (local vs watsonx).
- Include some governance / evaluation flavour.

Encourage teams to adapt scope to fit the available time.

## SUGGESTED FACILITATION TIPS

- Keep teams small (2–4 people) so everyone can contribute.
- Encourage early “thin slice” demos:
  - Simple but working RAG / agent path first.
  - Add polish only if time allows.
- Have a floating “**help desk**” (one or two facilitators) for debugging.

The capstone is about **learning and collaboration**, not perfection.