

## Exercise Lab - Self-Learning User Guide

### 1. What this lab is for

This Exercise Lab is for **knowledge workers**, not software engineers.

You'll use intuitive controls (dropdowns, tables, simple visual bars) to practice **design and judgment skills** related to:

Designing **schemas and labels** that drive dashboards

Building **reasoning blueprints** that prevent error cascades

Auditing **claims vs sources** to catch hallucinations

Designing **AI team roles and boundaries**

Creating **summarization recipes** that respect audience and risk

You don't need to write code. You just bring your **domain knowledge** and curiosity about how AI should behave in your real work.

### 2. Getting started

#### 1. Open the HTML file

Open Chapter2\_Exercise\_Lab.html in your browser (Chrome, Edge, etc.).

#### 2. Understand the layout

**Left side:** Navigation buttons for **Exercise 1–5**.

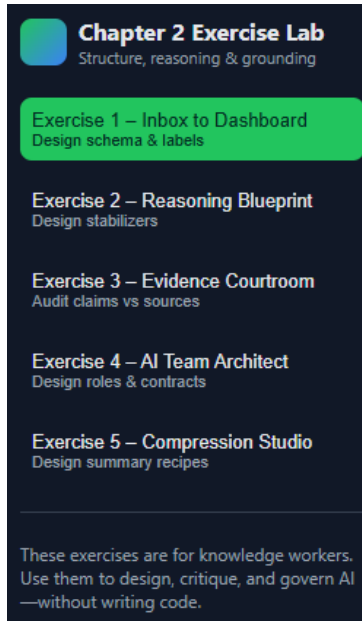
**Right side:** The currently selected exercise, shown as cards with controls and reflection areas.

#### 3. How to move around

Click any exercise on the left (e.g., “Exercise 1 – Inbox to Dashboard”) to switch views.

Only one exercise is visible at a time.

You can go **in order (1 → 5)** or jump directly to the exercises that match your current responsibilities (e.g., if you work with reports, start with Evidence Courtroom or Compression Studio).



## Exercise 1 - Inbox to Dashboard: Designing a Schema That Doesn't Lie

### In the app:

Exercise 1 - Inbox to Dashboard

## Exercise 1 – Inbox to Dashboard

Objective: Design a schema that drives honest dashboards

Choose fields and label sets for a business scenario, then see how your schema changes the story your dashboard tells.

### Step 1 – Choose a scenario

Scenario

Customer feedback triage

Classify incoming customer feedback so you can triage issues and spot themes.

### Step 2 – Pick your schema fields

Select which fields your AI assistant should extract and structure for this scenario.

☒ Sentiment label

☒ Topic/category

☒ Impact severity

☐ Channel (email / app / social)

☐ Owner / team

### Label sets for key fields

Severity labels

Critical / High / Medium / Low / Info

Sentiment labels

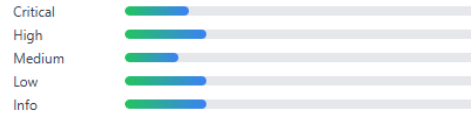
Positive / Neutral / Negative

### Step 3 – See the dashboard effect

Generate mocked dashboard

#### Label distribution

#### Severity / risk distribution



#### Sentiment distribution



#### Ops summary card

Your dashboard suggests about 9 items at 'High' level out of 40. In customer feedback triage, this might drive where support teams focus first, or it might hide slow-burning issues if labels are too coarse.

#### Reflection

How might your schema design mislead or help leadership if turned into a real dashboard?

## What you'll practice

You'll practice designing **simple schemas and label sets** for real-world scenarios and seeing how those choices change the story your **dashboard** tells.

You're moving from:

"I know what a schema is"  
to

"I understand how my schema and labels can mislead or clarify what leaders see."

## How to use it

### 1. Choose a scenario

Use the **Scenario** dropdown:

Customer feedback triage

Risk extraction from analyst reports

Internal incident reports

Read the short description below the dropdown. Picture how this fits your real work.

## 2. Pick your schema fields

In **Step 2 – Pick your schema fields**, you’ll see a list of possible fields (e.g., sentiment, severity, owner, location).

Check the boxes for the fields you want your AI assistant to extract.

These become the “columns” in your structured dataset.

## 3. Choose label sets

For **Severity labels**, choose a scheme (e.g., High/Medium/Low vs Critical/High/Medium/Low/Info).

For **Sentiment labels**, pick how fine-grained you want it (e.g., Positive/Neutral/Negative vs Strongly positive → Strongly negative).

## 4. Generate the mocked dashboard

Click **Generate mocked dashboard**.

The lab will show:

**Label distribution bars** (e.g., how many items are High vs Low)

An **Ops summary card** (“You have X high-risk items out of 40...”)

## 5. Reflect

In the **Reflection** area, answer:

“How might your schema design mislead or help leadership if turned into a real dashboard?”

### How to get value from it

Try a **minimal schema** (few fields) vs a **rich schema** (more fields) and see how the story changes.

Change severity label sets (binary vs 3 vs 5 levels) and ask:

*“Would this make leadership more likely to overreact or underreact?”*

Use this to start drafting **ground rules** in your team for how you structure feedback, incidents, or risk data when AI is involved.

## Exercise 2 - Reasoning Blueprint: Stop the Error Cascade Before It Starts

### In the app:

#### Exercise 2 - Reasoning Blueprint

### Exercise 2 – Reasoning Blueprint

Objective: Design a reasoning flow that reduces error cascades

Build a 3-step reasoning blueprint for a real task using stabilizers like anchor prompts and critique passes, then see likely failure and success stories.

#### Step 1 – Pick a task

Scenario

Summarize a 10-page policy for managers

Produce a manager-friendly summary of a long policy while preserving key rules and exceptions.

#### Step 2 – Stabilizer toolbox

You can use these stabilizers to strengthen each step:

- Anchor prompt
- Worked example
- Verification question
- Critique pass
- Grounding check (RAG)

#### Step 3 – Build your 3-step blueprint

Step	Stabilizer 1	Stabilizer 2 (optional)
Step 1	Anchor prompt	(none)
Step 2	Worked example	Grounding check (RA)
Step 3	Grounding check (RA)	(none)

Show likely outcomes

#### Likely failure mode

Without strong stabilizers, the policy summary may sound polished but quietly drop key exceptions, eligibility conditions, or dates. Managers walk away believing 'everyone is eligible' when the fine print says otherwise.

#### Likely robust outcome

With grounding checks and at least one verification question, your blueprint encourages the AI to cross-check rules against the source text and surface exceptions explicitly (e.g., 'Only employees with 12+ months of service qualify'). Your blueprint includes at least one stabilizer that forces the AI to either look back at the source (grounding) or question its own output (verification/critique), which directly reduces the chance of error cascades. Including worked examples helps the AI mirror the exact format and level of detail you want.

Reflection

Where would you add stabilizers in your real workflows to prevent error cascades?

## What you'll practice

You'll design a **3-step reasoning blueprint** for a task and choose **stabilizers** (anchor prompts, verification questions, critique passes, grounding checks, etc.) to reduce **error cascades**.

You're moving from:

"I know error cascades are bad"  
to

"I can design a reasoning flow that makes those cascades less likely."

## How to use it

### 1. Pick a task

Use the **Scenario** dropdown:

Summarize a 10-page policy for managers

Draft a compliance checklist from a regulation

Review a partner contract for red flags

Read the description and think of a similar task in your own job.

### 2. Review the stabilizer toolbox

You'll see a list of stabilizers:

Anchor prompt

Worked example

Verification question

Critique pass

Grounding check (RAG)

### 3. Build your 3-step blueprint

In the table (Step 1, Step 2, Step 3):

For each step, choose **Stabilizer 1** and optionally **Stabilizer 2** from the dropdowns.

Example:

Step 1: Anchor prompt + Worked example

Step 2: Grounding check (RAG)

Step 3: Critique pass + Verification question

#### 4. **Show likely outcomes**

Click **Show likely outcomes**.

The lab generates:

A **Likely failure mode** story if stabilizers are weak or missing

A **Likely robust outcome** story highlighting what your design helps prevent

#### 5. **Reflect**

In the **Reflection** area:

“Where would you add stabilizers in your real workflows to prevent error cascades?”

#### **How to get value from it**

- Compare a **bare-bones blueprint** (almost no stabilizers) vs a **robust one**.

Think of yourself as a **process designer**, not a prompt engineer:

*“Where in this flow should the AI be forced to slow down, check the source, or question its own answer?”*

Use your blueprint as a template when drafting real prompts for policies, checklists, or contract triage.

### **Exercise 3 - Evidence Courtroom: Judge the Claims, Not the Vibes**

**In the app:**

Exercise 3 – Evidence Courtroom

### Exercise 3 – Evidence Courtroom

Objective: Decide which claims are truly grounded in the retrieved evidence

Act as an evidence judge: label each claim in the model's answer as supported, partially supported, or unsupported by the retrieved snippets.

#### Step 1 – Choose a scenario

Scenario

HR policy helpdesk

You are answering employees' HR questions based on an internal policy manual.

#### Question

"Am I eligible for paid parental leave if I've worked here for 9 months?"

#### Model answer

Yes, you are eligible for 8 weeks of paid parental leave, because anyone who has worked more than 6 months at the company qualifies. The benefit applies to both full-time and part-time employees.

#### Step 2 – Compare answer with retrieved snippets

##### Retrieved snippets

- Parental Leave Policy, section 2.1: Employees with at least 12 months of continuous service are eligible for paid parental leave.
- Parental Leave Policy, section 2.2: The standard paid parental leave duration is 8 weeks.
- Parental Leave Policy, section 2.3: Paid parental leave applies to full-time employees only.

##### Individual claims

Claim	Your judgment
Employees with more than 6 months of service qualify for paid parental leave.	Unsupported
The standard paid parental leave duration is 8 weeks.	Supported
Paid parental leave applies to both full-time and part-time employees.	Supported
The answer given to the employee is incorrect about eligibility timing.	Unsupported

#### Check my judgments

You matched 2 out of 4 claim judgments. Pay special attention to borderline-sounding statements that feel plausible but aren't actually stated in the snippets. Claim 3 is best seen as 'unsupported'. Re-read the snippets to see whether they actually say what the claim asserts. Claim 4 is best seen as 'partial'. Re-read the snippets to see whether they actually say what the claim asserts.

#### Reflection

How could you use this claim-by-claim thinking when reading AI answers at work?

## What you'll practice

You'll act as an **evidence judge**:

- Look at an AI-generated answer,
- Read the retrieved **snippets** behind it,
- Decide for each sentence whether it's **supported**, **partially supported**, or **unsupported**.

You're moving from:

- "Grounding sounds important"
- to



- “I can systematically audit whether an AI answer is actually grounded in its sources.”

## How to use it

### 1. Choose a scenario

- Use the **Scenario** dropdown:
  - HR policy helpdesk
  - Product feature FAQ
  - Regulatory compliance helpdesk
- Read the short description.

### 2. Read the question and answer

- Under **Question**, read what the user asked.
- Under **Model answer**, read the AI’s response as if you were the end user.

### 3. Review the retrieved snippets

- Under **Retrieved snippets**, read the passages the system supposedly used.
- These represent the “evidence” behind the answer.

### 4. Judge each claim

- Under **Individual claims**, each claim is a single sentence pulled out of the answer.
- For each claim, use the dropdown to choose:
  - Supported
  - Partially supported
  - Unsupported
- Think: *“If I were a regulator, could I point to a sentence in the snippets that backs this up?”*

### 5. Check your judgments

- Click **Check my judgments**.
- The lab will:

- Tell you how many claims you matched with the “expert” answer key
- Give coaching comments where your call differed

## 6. Reflect

- In the **Reflection** area:

“How could you use this claim-by-claim thinking when reading AI answers at work?”

### How to get value from it

- Notice how **plausible but unsupported** claims feel when you see them. That “this sounds right” instinct is exactly how hallucinations slip through in real life.
- Practice this with your own AI outputs (e.g., policy summaries, risk notes) by mentally breaking them into claims and checking each against the source.
- Use this exercise as the basis for a “**grounded answer checklist**” in your team.

## Exercise 4 – AI Team Architect: Designing Roles, Boundaries, and Handoffs

### In the app:

*Exercise 4 – AI Team Architect*

## Exercise 4 – AI Team Architect

Objective: Define clear roles, permissions, and handoffs for an AI agent team

Treat agents like teammates: define what each is allowed to do, what they focus on, and where collisions or safety issues might appear.

### Step 1 – Choose a workflow

Workflow

Marketing content pipeline

Design an AI-assisted marketing content pipeline (from research to legally safe final copy).

### Step 2 – Configure your AI team

Agent	Job focus	Allowed actions
Research agent	Generate ide	<input type="checkbox"/> Change structure <input type="checkbox"/> Change meaning <input checked="" type="checkbox"/> Introduce new facts
Writer agent	Draft main c	<input checked="" type="checkbox"/> Change structure <input checked="" type="checkbox"/> Change meaning <input checked="" type="checkbox"/> Introduce new facts
Editor agent	Enforce style	<input checked="" type="checkbox"/> Change structure <input checked="" type="checkbox"/> Change meaning <input type="checkbox"/> Introduce new facts
Legal reviewer agent	Check facts /	<input checked="" type="checkbox"/> Change structure <input checked="" type="checkbox"/> Change meaning <input checked="" type="checkbox"/> Introduce new facts

### Step 3 – Check for collisions and risks

Analyze my design

#### Role clarity score

Estimated role clarity: 90/100.

#### Potential collisions

- No major collisions detected. Roles appear reasonably distinct.

#### Safety & injection hints

- Legal reviewer agent is focused on checking facts but is also allowed to introduce new facts. This weakens the checker role.

#### Reflection

If these agents were real teammates, would they know what they are allowed to do? What would you change?

## What you'll practice

You'll design an **AI team**: multiple agents with different roles.

You'll define:

Job focus (e.g., generate ideas, check facts, route cases)

Allowed actions (e.g., change meaning, introduce new facts)

Then see whether your design has **role collisions** or **safety issues**.

You're moving from:

"I know what an agent is"

to

"I can design clear, safe roles for a team of AI agents."

## How to use it

### 1. Choose a workflow

Use the **Workflow** dropdown:

Marketing content pipeline

Risk report pipeline

Customer support assistant

Read the description to understand the scenario.

### 2. Configure your AI team

In the table, you'll see agents like: Research agent, Writer agent, Editor agent, Legal reviewer agent, etc.

For each agent:

Choose a **Job focus** from the dropdown (e.g., Generate ideas, Draft main content, Check facts, Enforce style, Route/escalate).

Check the **Allowed actions** that apply:

Change structure

Change meaning

Introduce new facts

### 3. Analyze your design

Click **Analyze my design**.

The lab will:

Show a **Role clarity score** (0–100)

List **Potential collisions** (e.g., two agents both changing meaning for the same role)

List **Safety & injection hints** (e.g., multiple agents introducing facts, but no checker role)

### 4. Reflect

In the **Reflection** area:

“If these agents were real teammates, would they know what they are allowed to do? What would you change?”

## **How to get value from it**

Think of this as designing a **human org chart**, but for AI assistants.

Watch for patterns like:

No one owns **checking facts**

Too many agents can **introduce new facts**

A “checker” that is also allowed to introduce new facts (not ideal)

Use your insights to propose **real-world AI role charters** in your organization (e.g., “This assistant may change wording and structure, but may not introduce new, unverified facts.”).

## Exercise 5 – Compression Studio: From 50 Pages to the Right Summary

### In the app:

#### Exercise 5 – Compression Studio

**Exercise 5 – Compression Studio**

Objective: Design the right summarization path for your audience

Choose an audience and use case, then design a compression path that turns a long document into the right kind of summary—with visibility into risks.

**Step 1 – Choose a document and audience**

Document type

50-page vendor contract

Audience

CFO / executive

Use case

Decision briefing

You are summarizing a 50-page vendor contract for a CFO/executive to support a decision briefing.

**Step 2 – Build your compression path**

Chunking strategy	By sections
Section summary style	Bulleted risks
Final merge format	Executive summary
Verbosity control	2 bullets per section

**Step 3 – Preview and risk check**

Generate sample summary

**Sample summary snippets**

Executive summary: Key decisions and risks distilled for quick review.

- Top 3 risks highlighted with a brief explanation for each.
- Additional lower-level items grouped for later review.

The document is processed section-by-section, so structure mostly follows the original layout.

**Compression overview**

50 pages → approximately 300 words of summary in the form of a single executive summary.

**Potential risks**

- No obvious red flags from this simple analysis, but always have a subject-matter expert review critical summaries.

Reflection

Would your target audience accept this level of compression? What would you adjust?

## What you'll practice

You'll design a **summarization recipe**:

Document type (contract, clinical paper, policy manual)

Audience (CFO, lawyer, clinician, manager, new hire)

Use case (decision brief, risk review, orientation)

Chunking, summary style, merge format, and verbosity

You're moving from:

"AI can summarize stuff"

to

"I can design a summarization strategy that fits my audience and risk level."

## How to use it

### 1. Choose document and audience

Select **Document type**:

50-page vendor contract

Clinical research paper

Policy manual

Select **Audience** (e.g., CFO, legal counsel, busy clinician, front-line manager, new hire).

Select **Use case**:

Decision briefing

Risk review

Quick orientation

Read the combined description under **Step 1** — it describes who is reading what, and for which purpose.

### 2. Build your compression path

In **Step 2 – Build your compression path**:

**Chunking strategy**: By sections, topics, or pages

**Section summary style:** Bulleted risks, narrative paragraphs, or Q&A

**Final merge format:** Executive summary, FAQ, or checklist

**Verbosity control:** 2 bullets per section, 5 key points total, or 200-word cap

### 3. **Generate sample summary**

Click **Generate sample summary**.

The lab will show:

**Sample summary snippets** that illustrate your choices (style, format, chunking)

A **Compression overview** (e.g., “50 pages → ~250 words in FAQ format”)

A **Potential risks** list (e.g., lost nuance for legal; missing limitations in clinical summaries; confusing structure for new hires)

### 4. **Reflect**

In the **Reflection** area:

“Would your target audience accept this level of compression? What would you adjust?”

#### **How to get value from it**

Try different combinations for the **same audience** and watch how risks change.

Ask:

*“If I were in this audience, would I feel over-briefed or under-briefed?”*

*“What absolutely cannot be compressed away for this domain (e.g., legal, medical, safety)?*

Use this to frame real guidelines like:

“For legal, always add a ‘must-read clauses’ annex.”

“For clinicians, always include a dedicated section on limitations and adverse effects.”

## 8. **Putting it all together**

As you complete the Chapter 2 Exercise Lab:

### 1. **Collect your reflections**

Export or copy your written reflections from each exercise.



Combine them in one place as your **Chapter 2 AI Playbook**.

## 2. Look for patterns

How do your schema choices influence dashboards and decisions?

Where do you tend to **overtrust** model answers?

How do you like to balance **brevity vs safety** in summaries?

## 3. Turn insights into practice

Draft 2–3 concrete rules for your own AI use, e.g.:

“I will never trust a summary about regulations without scanning a few key snippets.”

“For executive briefings, I will use a two-layer summary: 1-page exec summary plus a short risk annex.”

“Any multi-step AI process I design will include at least one explicit check or critique step.”

This lab is about helping you think like a **designer and governor** of AI systems in your day job—not a coder. If you revisit these exercises after using AI more heavily, you’ll likely make sharper, more confident choices, which is exactly the point.