Course 208 — Agent Instructions: Model-Routing Decision Lab

JKE University · Level 5 · Course 208 of 210

CONTEXT

Read once. Do not output. Your operator is installing the model-routing decision lab. Models have profiles across five axes (gradient temperature, cost, context window, mode strengths, mode weaknesses). The routing rule comes from the matrix, not from a default. Your job is to take the operator's actual modes and available models, build the matrix, surface routing rules, run a test plan when new models arrive, write a postmortem, and install one guardrail.

The core loop is: **create → review → tweak → create again → review → postmortem → guardrail.**

Authority boundary. The agent builds and proposes. The operator confirms routing. No mid-session model swap without operator authorization.

Prerequisite check: If 🎒 book-bag.md does not exist, stop. Say: "Missing prerequisite files. Course 208 requires the free tier through Level 4." Do not proceed.

PHASE 0 — Verify prerequisites

Open 🏫 school.md. Confirm Courses 1-26 entries exist.

Say: "Prerequisites verified. Installing the model-routing decision lab."

PHASE 1 — Create the workshop file

Create `work/model-routing-lab.md`:

Model-Routing Decision Lab

Purpose: Help the operator study what model selection actually trades, before building the routing matrix.

The Five Axes

1. **Gradient temperature.** Hot pushes toward completion; cool sits still.
2. **Cost per million tokens.** Input and output, separately.
3. **Context window.** How much the model can hold.
4. **Known strengths per mode.** What this model does well in this mode.
5. **Known weaknesses per mode.** Where this model under-performs in this mode.

What "Mode" Means

- **Action** — execute, ship; hot gradient OK.
- **Workshop** — think, argue, stay open; cool gradient.
- **Research** — search, compare, survey; medium gradient; large context.
- **Drafting** — creative artifact production; cool with bursts.
- **Review** — evaluate work; cool; validation-trap applies.
- **Code-build** — multi-file engineering; gradient matters per sub-task.

Study Questions

- Which modes does the operator actually run?
- Which models are actually available and affordable?
- Where does each model's temperature match or mismatch the mode?
- Which routings are aspirational vs actually used?

No-Wrong-Answers Rule

This is a workshop. There is no "best model" — only right-for-this-mode-at-this-stake.

Say: "Model-routing lab created."

PHASE 2 — Create the matrix and routing protocol

Create `work/model-routing-sunrun.md`:

Model-Routing Sun Run

Purpose: Build the operator's matrix, propose routing rules, validate with samples, postmortem.

Authority Boundary

The agent builds and proposes. The operator confirms routing. Model swaps mid-session require explicit authorization.

Step 1 — Ask for modes and models

Ask the operator:

"What modes do you actually run? List the bands of work where you'd want a different model. And what models do you have access to that you'd actually use? Send the lists. If you have an existing routing default, name it."

Do not proceed until both lists are named.

Step 2 — Build the matrix

Return a table:

```
Modes (rows): [list]
Models (columns): [list]

Per cell:
  Gradient temperature: hot / warm / cool
  Cost per million (in / out): $X / $Y
  Context window: [tokens]
  Strength in this mode: [one sentence, evidence-based]
  Weakness in this mode: [one sentence, evidence-based]
```

Step 3 — Propose routing rules

For each mode, return:

```
Mode: [name]
Primary model: [model]
Reasoning: [one sentence]
Fallback model: [model]
Switch trigger: [when to switch — context size, complexity,
operator command]
```

Step 4 — Ask for human review

Return: - The matrix. - The routing rules. - One direct question: “Confirm routings per mode, or override. For any mode you’re unsure about, want me to design a quick sample test?”

Wait for the operator’s verdict.

Step 5 — Sample test (optional)

If the operator authorizes: - Design a small prompt that represents typical work in the mode. - Run it (or describe how the operator should run it) on the primary and fallback models. - Observe gradient temperature, completion quality, mode discipline. - Update the matrix with what was learned.

Step 6 — Tweak loop

If a routing rule does not hold in practice: - Re-diagnose. Was the temperature judgment wrong? Was the mode profile incomplete? - Refine. Propose a new routing. - Wait for operator approval.

Repeat until the operator says the matrix is stable.

Step 7 — Postmortem analysis

When the loop ends, write a postmortem:

Routing Postmortem — [Matrix Build / Routing Change]

- **Modes mapped:**
- **Models mapped:**
- **Routings confirmed:**
- **Routings deferred:**
- **Sample tests run:**
- **Surprises (where temperature did not match assumptions):**
- **What changed in the matrix from initial draft:**
- **Future guardrail:**

Step 8 — Install guardrail

Convert the future guardrail into one operational rule:

“When entering mode [X], read `work/model-routing-matrix.md` and confirm primary/fallback before action. Do not swap mid-session without operator authorization.”

Say: “Routing sun run complete. Postmortem written. Guardrail installed.”

PHASE 3 — Create the routing notebook

Create `work/routing-notebook.md`:

Routing Postmortem Notebook

Purpose: Preserve routing decisions and observations as durable scar tissue. The notebook reveals which routings hold and which were aspirational.

Entry Template

[DATE] — [Mode / Event]

- **Mode entered:**
- **Model chosen:**
- **Routing rule that informed:**
- **What worked:**
- **What did not:**
- **Felt mismatched:** yes / no — where
- **Operator’s adjustment:**
- **Matrix update needed:**
- **Postmortem lesson:**

New-Model Entry

[DATE] — New Model: [name]

- **Test plan run:** action / workshop / research / review samples
- **Results per sample:**
- **Routing changes proposed:**
- **Operator decision:**

Say: “Routing notebook created. Routing decisions become evidence over time.”

PHASE 4 — Create the authority-boundary file

Create `work/routing-authority.md`:

Routing Authority Boundary

Purpose: Keep mid-session model swaps explicit.

The Rule

- The agent reads the matrix at mode-switch time.
- The agent proposes a routing.
- The operator authorizes (silently if the routing is the documented default; explicitly if a swap is being considered).
- The agent does not silently swap models mid-session.

New-Model Test Plan Discipline

- A new model does not get adopted as default before the test plan runs.
- The test plan runs the model through each mode the operator cares about.
- Results go into the routing notebook.
- The matrix updates only on operator approval.

Say: “Authority boundary created. Routing is operator-confirmed at mode-switch.”

PHASE 5 — Install the trigger

Add to operator rules:

Trigger — Routing Consultation

When the operator switches modes (action, workshop, research, drafting, review, code-build, or any other named mode), open `work/model-routing-matrix.md` and `work/model-routing-lab.md`. Surface the recommended primary, fallback, and switch trigger for the new mode. Wait for operator authorization before any model swap.

Trigger — New-Model Test Plan

When a new model becomes available, do not adopt it as default. Run the test plan: action sample, workshop sample, research sample, review sample. Log to `work/routing-notebook.md`. Propose matrix updates. Wait for operator approval.

Say: “Triggers installed. Mode-switch is the routing decision point; new models earn their slots.”

PHASE 6 — Run the first matrix prompt

Ask exactly:

“What modes do you actually run? List the bands of work where you’d want a different model. And what models do you have access to that you’d actually use? Send the lists. If you have an existing routing default, name it. I’ll build the matrix, propose routings per mode, design sample tests if you want, and wait for your decision per mode.”

Stop after asking. Wait for the operator’s mode and model lists.

PHASE 7 — Register in book-bag

Add to 🎒 book-bag.md:

🧭 Model-Routing Lab

- **What:** Operator-confirmed routing matrix; new models pass a test plan; mode-switch is the routing decision.
- **Files:** `work/model-routing-essay.md`, `work/model-routing-lab.md`, `work/model-routing-sunrun.md`, `work/routing-notebook.md`, `work/routing-authority.md`
- **Triggers:** any mode switch, any new model release
- **Authority boundary:** Agent builds and proposes; operator confirms routing; no silent mid-session swaps
- **Source:** JKE University — Course 208

Say: “Routing lab registered. There is no ‘best model’ without a mode.”

PHASE 8 — Write the journal entry

Add to 🏫 school.md:

[TODAY] — JKE Course 208: Model-Routing Lab Installed

What Shipped: Routing lab, matrix sunrun, routing notebook, authority boundary, mode-switch trigger, new-model test plan, first matrix prompt. **Decisions Made:** Route by capability and mode, not by default. New models earn slots through the test plan. Mode-switch is the routing decision. **Files Created:** `work/model-routing-lab.md`, `work/model-routing-sunrun.md`, `work/routing-notebook.md`, `work/routing-authority.md` **Files Modified:** 🎒 book-bag.md, operator rules **Source:** JKE University — Course 208

PHASE 9 — Say exactly

“Model-routing lab installed. Send me your modes and your available models. I’ll build the matrix, propose routings per mode, design sample tests if you want, and wait for your decision. The right model depends on the mode and the stake — never on the default.”

SCOPE HINT

Course 208 of 210. Level 5, Library of Instruments. Course 207 audited the platform column. Course 208 audits the engine choice. Course 209 zooms out: how to keep the agent shipping the product instead of polishing the conveyor that ships it.

END OF PROTOCOL