Thinking With Agents

Webinar 2: AI for Teaching

Emily Beam & **Erkmen G. Aslim**

Department of Economics, University of Vermont

June 2026 · Live Online

thinkingwithagents.github.io

This Hour

1. A Few Ideas

Teaching prep is **production** — agents make it cheap. Your judgement stays the scarce input.

~10 min, slides

This Hour

1. A Few Ideas

Teaching prep is **production** — agents make it cheap. Your judgement stays the scarce input.

~10 min, slides

2. Lecture Builder

Papers in → a ready-to-teach lecture out: notes, slides, discussion prompts, exam questions. Skill + live demo.

~25 min, mostly live

This Hour

1. A Few Ideas

Teaching prep is **production** — agents make it cheap. Your judgement stays the scarce input.

~10 min, slides

2. Lecture Builder

Papers in → a ready-to-teach lecture out: notes, slides, discussion prompts, exam questions. Skill + live demo.

~25 min, mostly live

3. Dashboards

One prompt → an interactive tool your students play with **inside your LMS**. Live demo + embed walkthrough.

~20 min, live

This Hour

1. A Few Ideas

Teaching prep is **production** — agents make it cheap. Your judgement stays the scarce input.

~10 min, slides

2. Lecture Builder

Papers in → a ready-to-teach lecture out: notes, slides, discussion prompts, exam questions. Skill + live demo.

~25 min, mostly live

3. Dashboards

One prompt → an interactive tool your students play with **inside your LMS**. Live demo + embed walkthrough.

~20 min, live

Most of this hour happens **live in the terminal**. Every prompt we run is on a slide so you can copy as we go.

Teaching Prep Is Production — and Production Got Cheap

What agents now produce in minutes

- Lecture notes & slide decks from papers
- Problem sets, exam questions, rubrics
- Interactive tools, simulations, dashboards
- Reading lists — found *and* downloaded

Teaching Prep Is Production — and Production Got Cheap

What agents now produce in minutes

- Lecture notes & slide decks from papers
- Problem sets, exam questions, rubrics
- Interactive tools, simulations, dashboards
- Reading lists — found *and* downloaded

What stayed yours

- Is it **correct**? (numbers, citations, claims)
- Is it **right for your students**? (level, pacing)
- Is it **what you'd stand behind** in class?

Teaching Prep Is Production — and Production Got Cheap

What agents now produce in minutes

- Lecture notes & slide decks from papers
- Problem sets, exam questions, rubrics
- Interactive tools, simulations, dashboards
- Reading lists — found *and* downloaded

What stayed yours

- Is it **correct**? (numbers, citations, claims)
- Is it **right for your students**? (level, pacing)
- Is it **what you'd stand behind** in class?

Webinar 1's laws carry over unchanged: never defer blindly, and **you** — not the agent — are responsible for what reaches your students.

Teaching Prep Is Production — and Production Got Cheap

What agents now produce in minutes

- Lecture notes & slide decks from papers
- Problem sets, exam questions, rubrics
- Interactive tools, simulations, dashboards
- Reading lists — found *and* downloaded

What stayed yours

- Is it **correct**? (numbers, citations, claims)
- Is it **right for your students**? (level, pacing)
- Is it **what you'd stand behind** in class?

Webinar 1's laws carry over unchanged: never defer blindly, and **you** — not the agent — are responsible for what reaches your students.

Review everything **before** the classroom does.

Today's Two Demos = Two Ends of a Spectrum

A skill: `lecture-builder`

A reusable **playbook**: parse the paper in chunks → structured extraction → notes + deck, every time.

- Same file layout, same quality, repeatable
- Has an **audience knob**: undergrad vs. grad
- Worth building for workflows you repeat all semester

Today's Two Demos = Two Ends of a Spectrum

A skill: `lecture-builder`

A reusable **playbook**: parse the paper in chunks → structured extraction → notes + deck, every time.

- Same file layout, same quality, repeatable
- Has an **audience knob**: undergrad vs. grad
- Worth building for workflows you repeat all semester

One good prompt: the dashboard

No skill, no setup — a single well-specified prompt produces a polished interactive tool.

- Right for **one-off artifacts**
- The craft is in the **specification**
- If you start repeating it → promote it to a skill

Today's Two Demos = Two Ends of a Spectrum

A skill: `lecture-builder`

A reusable **playbook**: parse the paper in chunks → structured extraction → notes + deck, every time.

- Same file layout, same quality, repeatable
- Has an **audience knob**: undergrad vs. grad
- Worth building for workflows you repeat all semester

One good prompt: the dashboard

No skill, no setup — a single well-specified prompt produces a polished interactive tool.

- Right for **one-off artifacts**
- The craft is in the **specification**
- If you start repeating it → promote it to a skill

The decision rule: **Repeat it? Skill it. One-off? Prompt it.**

Demo 1

`lecture-builder` — Papers In, a Ready-to-Teach Lecture Out

`lecture-builder` — What It Is

Document in → lecture out

- Takes a paper (PDF/DOCX/URL) — parses it **in chunks**, so a 100-page paper doesn't blow up the context window
- Extracts claims, findings, methods, discussion prompts, exam questions into one structured file
- Writes `lecture_notes.md` + a Beamer deck from that **same** extraction — notes and slides can't drift apart

`lecture-builder` — What It Is

Document in → lecture out

- Takes a paper (PDF/DOCX/URL) — parses it **in chunks**, so a 100-page paper doesn't blow up the context window
- Extracts claims, findings, methods, discussion prompts, exam questions into one structured file
- Writes `lecture_notes.md` + a Beamer deck from that **same** extraction — notes and slides can't drift apart

The audience knob

Undergraduate: intuition first, jargon defined, open-ended discussion.

Graduate: identification at a craft level, critique & replication-style questions.

Flip it with one word in the prompt.

Slide titles are **takeaways**, not topics: "Productivity up 15%" beats "Results."

`lecture-builder` — What It Is

Document in → lecture out

- Takes a paper (PDF/DOCX/URL) — parses it **in chunks**, so a 100-page paper doesn't blow up the context window
- Extracts claims, findings, methods, discussion prompts, exam questions into one structured file
- Writes `lecture_notes.md` + a Beamer deck from that **same** extraction — notes and slides can't drift apart

The audience knob

Undergraduate: intuition first, jargon defined, open-ended discussion.

Graduate: identification at a craft level, critique & replication-style questions.

Flip it with one word in the prompt.

Slide titles are **takeaways**, not topics: "Productivity up 15%" beats "Results."

Today's run: an undergrad lecture on **"AI and the Labor Market"** — a topic your students are already asking about.

Step 1: One Prompt Finds (and Fetches) the Reading List

```
> claude

I'm preparing an undergraduate lecture on how AI is affecting the economy, especially the labor market.

1. Search broadly for the most influential papers (2023-2026): employment, wages, tasks, productivity, inequality.

2. Build an annotated reading list (authors, venue, one-line takeaway) and save it as papers/reading_list.md.

3. Download the open-access PDFs (arXiv, NBER, journal sites) into papers/ and verify each file is a real PDF.
```

Step 1: One Prompt Finds (and Fetches) the Reading List

```
> claude

I'm preparing an undergraduate lecture on how AI is affecting the economy, especially the labor market.

1. Search broadly for the most influential papers (2023-2026): employment, wages, tasks, productivity, inequality.

2. Build an annotated reading list (authors, venue, one-line takeaway) and save it as papers/reading_list.md.

3. Download the open-access PDFs (arXiv, NBER, journal sites) into papers/ and verify each file is a real PDF.
```

Note what the prompt pins down: **audience** (undergrad), **scope** (labor market), **artifacts** (reading list + PDFs on disk), and a **verification step**. Vague in $\rightarrow$ vague out.

Step 1: One Prompt Finds (and Fetches) the Reading List

```
> claude

I'm preparing an undergraduate lecture on how AI is affecting the economy, especially the labor market.

1. Search broadly for the most influential papers (2023-2026): employment, wages, tasks, productivity, inequality.

2. Build an annotated reading list (authors, venue, one-line takeaway) and save it as papers/reading_list.md.

3. Download the open-access PDFs (arXiv, NBER, journal sites) into papers/ and verify each file is a real PDF.
```

Note what the prompt pins down: **audience** (undergrad), **scope** (labor market), **artifacts** (reading list + PDFs on disk), and a **verification step**. Vague in $\rightarrow$ vague out.

In our test run this took $\sim$**3 minutes**: 6 papers found, 6 PDFs downloaded & verified, reading list written.

What Came Back: Six Papers, Annotated, On Disk

Paper	One-line takeaway (as annotated by the agent)
Brynjolfsson, Li & Raymond (*QJE* 2025)	AI assistant for 5,172 support agents: productivity **+15%** — +30%+ for novices, ≈0 for the most experienced
Eloundou et al. (2023, *Science* 2024)	∼80% of workers have ≥10% of tasks LLM-exposed; **high-wage** jobs most exposed
Acemoglu (*Economic Policy* 2024)	Task model + Hulten: 10-year TFP gain ≤**0.66%** — far below the hype
Autor (NBER 2024)	The optimist case: AI can extend expertise and **rebuild middle-class jobs**
Humlum & Vestergaard (2025)	Danish admin data, DiD: **precise null** on earnings & hours two years post-adoption
Handa et al. (2025)	4M Claude conversations: usage is ∼**57% augmentation** vs. 43% automation

What Came Back: Six Papers, Annotated, On Disk

Paper	One-line takeaway (as annotated by the agent)
Brynjolfsson, Li & Raymond (*QJE* 2025)	AI assistant for 5,172 support agents: productivity **+15%** — +30%+ for novices, $\approx$0 for the most experienced
Eloundou et al. (2023, *Science* 2024)	$\sim$80% of workers have $\geq$10% of tasks LLM-exposed; **high-wage** jobs most exposed
Acemoglu (*Economic Policy* 2024)	Task model + Hulten: 10-year TFP gain $\leq$**0.66%** — far below the hype
Autor (NBER 2024)	The optimist case: AI can extend expertise and **rebuild middle-class jobs**
Humlum & Vestergaard (2025)	Danish admin data, DiD: **precise null** on earnings & hours two years post-adoption
Handa et al. (2025)	4M Claude conversations: usage is $\sim$**57% augmentation** vs. 43% automation

A built-in classroom debate: micro gains are large (row 1), exposure is broad (row 2), yet aggregate effects are $\approx$0 so far (row 5).

Step 2: Feed the Papers to `lecture-builder`

```
Now build me a 50-minute undergraduate lecture from the papers in papers/.

Use "Generative AI at Work" (Brynjolfsson, Li & Raymond) as the anchor paper and weave the others in
   as context:  exposure, the macro debate, and the early aggregate evidence.

I want lecture notes, a Beamer slide deck, in-class discussion prompts, and candidate exam questions.

Audience:  principles/intermediate undergraduates - intuition first, define jargon, takeaway-style slide titles.
```

Step 2: Feed the Papers to `lecture-builder`

```
Now build me a 50-minute undergraduate lecture from the papers in papers/.

Use "Generative AI at Work" (Brynjolfsson, Li & Raymond) as the anchor paper and weave the others in

   as context:  exposure, the macro debate, and the early aggregate evidence.

I want lecture notes, a Beamer slide deck, in-class discussion prompts, and candidate exam questions.

Audience:  principles/intermediate undergraduates - intuition first, define jargon, takeaway-style slide titles.
```

What the skill does under the hood

`parse_document.py` chunks each PDF into sections (38 for the anchor) → reads selectively, not wholesale → `extraction.json` (claims, findings + *where in the paper*, methods, questions) → notes + deck generated from that single source of truth.

Step 2: Feed the Papers to `lecture-builder`

```
Now build me a 50-minute undergraduate lecture from the papers in papers/.

Use "Generative AI at Work" (Brynjolfsson, Li & Raymond) as the anchor paper and weave the others in

   as context:  exposure, the macro debate, and the early aggregate evidence.

I want lecture notes, a Beamer slide deck, in-class discussion prompts, and candidate exam questions.

Audience:  principles/intermediate undergraduates - intuition first, define jargon, takeaway-style slide titles.
```

What the skill does under the hood

`parse_document.py` chunks each PDF into sections (38 for the anchor) → reads selectively, not wholesale → `extraction.json` (claims, findings + *where in the paper*, methods, questions) → notes + deck generated from that single source of truth.

Output of our test run: **15-frame compiled Beamer deck + 11-section lecture notes** + 5 discussion prompts + 6 exam questions (short answer, MC, essay) — a TA could run the section from the notes alone.

Live: From a Paper to a Lecture, Start to Finish

Pick a paper. Watch it become
notes + slides + discussion prompts + exam questions.

parse · extract · notes · deck · compile

Watch for: the chunked parse (no 100-page context dump) · claims-not-topics in `extraction.json` · the audience knob doing real work.

Then the 60-second edit: "make slide 8 graduate-level" — regenerate one piece, not the whole deck.

Teaching a paper next week? Drop its title or arXiv link in the chat — if time allows, we'll run yours.

Demo 2

Teaching Dashboards — One Prompt, Any Course, Any LMS

The Idea: Interactive Tools as Single Files

Why one self-contained HTML file

- Everything inside: graph, sliders, quiz — **no internet, no installs, no accounts**
- Opens in any browser, on laptops *and* phones
- Embeds in **every** LMS (Canvas, Blackboard, Brightspace, Moodle) — next slides
- Collects no student data; can't break your course page

The Idea: Interactive Tools as Single Files

Why one self-contained HTML file

- Everything inside: graph, sliders, quiz — **no internet, no installs, no accounts**
- Opens in any browser, on laptops *and* phones
- Embeds in **every** LMS (Canvas, Blackboard, Brightspace, Moodle) — next slides
- Collects no student data; can't break your course page

Why it teaches

Students don't watch the comparative statics — they **cause** them. Predict first (quiz mode), then watch the graph act it out.

Think "a PDF that happens to be interactive" — that's the mental model for your LMS, too.

The Idea: Interactive Tools as Single Files

Why one self-contained HTML file

- Everything inside: graph, sliders, quiz — **no internet, no installs, no accounts**
- Opens in any browser, on laptops *and* phones
- Embeds in **every** LMS (Canvas, Blackboard, Brightspace, Moodle) — next slides
- Collects no student data; can't break your course page

Why it teaches

Students don't watch the comparative statics — they **cause** them. Predict first (quiz mode), then watch the graph act it out.

Think "a PDF that happens to be interactive" — that's the mental model for your LMS, too.

Worked example: a **supply & demand explorer** for principles of micro — but the recipe is course-agnostic.

The Prompt (the Craft Is in the Specification)

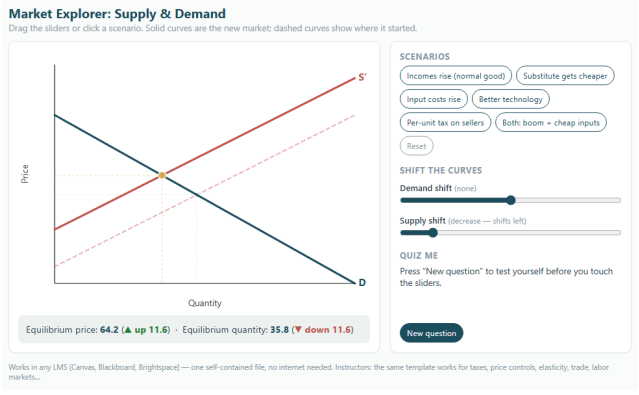
```
> claude

Build an interactive supply-and-demand explorer for my principles of microeconomics class:

- one self-contained HTML file, no external libraries, must work offline and inside an LMS iframe

- sliders to shift demand and supply, plus preset scenario buttons (incomes rise, input costs rise,
   per-unit tax on sellers, better technology); show the original curves dashed after a shift

- live readout of equilibrium price and quantity, with up/down arrows vs. the starting point

- a "quiz me" mode:  ask a shift question, let the student answer, then animate the correct shift

- clean modern look, readable on phones, axis labels, no walls of text

Save it as supply-demand-explorer.html and screenshot it so I can check the design.
```

The Prompt (the Craft Is in the Specification)

```
> claude

Build an interactive supply-and-demand explorer for my principles of microeconomics class:

- one self-contained HTML file, no external libraries, must work offline and inside an LMS iframe

- sliders to shift demand and supply, plus preset scenario buttons (incomes rise, input costs rise,

   per-unit tax on sellers, better technology); show the original curves dashed after a shift

- live readout of equilibrium price and quantity, with up/down arrows vs. the starting point

- a "quiz me" mode:  ask a shift question, let the student answer, then animate the correct shift

- clean modern look, readable on phones, axis labels, no walls of text

Save it as supply-demand-explorer.html and screenshot it so I can check the design.
```

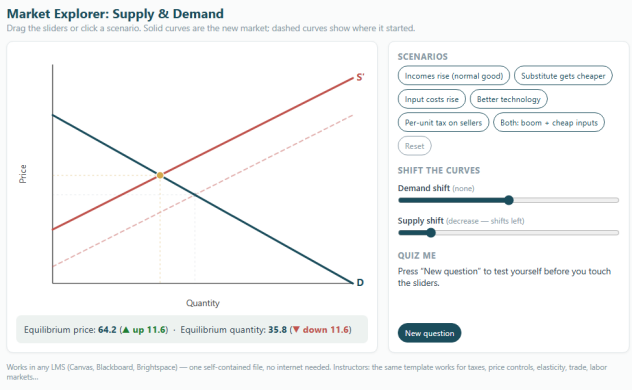
Every line buys robustness: **self-contained** (no library = nothing to break in an iframe) · **offline** · **phone-readable** · **screenshot** so you verify by *seeing*. This is the whole spec — there is no step 2.

What It Built (Our Test Run, Unedited)



"Input costs rise": S shifts left (old curve dashed), P ▲ up 11.6, Q ▼ down 11.6.

What It Built (Our Test Run, Unedited)



"Input costs rise": S shifts left (old curve dashed), P ▲ up 11.6, Q ▼ down 11.6.

Out of the box

- 6 preset scenarios + free sliders
- Quiz mode: predict, then watch
- Equilibrium deltas with arrows
- Deep links: ?s=−22 opens pre-shifted — link it from a homework question

One file, ∼400 lines, zero dependencies. Built and verified in minutes.

Putting It in Your LMS (No Coding, Two Paths)

Path A — host once, embed anywhere (recommended)

Tell the agent: *"Publish this to GitHub Pages and give me the link."* Then paste **one line** into any LMS editor's embed/HTML button:

```
<iframe src="https://you.github.io/econ-dashboards/supply-demand-explorer.html" width="100%" height="700"></iframe>
```

Putting It in Your LMS (No Coding, Two Paths)

Path A — host once, embed anywhere (recommended)

Tell the agent: *"Publish this to GitHub Pages and give me the link."* Then paste **one line** into any LMS editor's embed/HTML button:

```
<iframe src="https://you.github.io/econ-dashboards/supply-demand-explorer.html" width="100%" height="700"></iframe>
```

LMS	Where the embed button lives
Brightspace (D2L)	Content → Create a File → **Insert Stuff** → Enter Embed Code
Canvas	Pages → Edit → **Insert** → **Embed** (or the <> HTML editor)
Blackboard	Build Content → Item → **<> source-code** button (Ultra: Document → Add HTML)

Putting It in Your LMS (No Coding, Two Paths)

Path A — host once, embed anywhere (recommended)

Tell the agent: *"Publish this to GitHub Pages and give me the link."* Then paste **one line** into any LMS editor's embed/HTML button:

```
<iframe src="https://you.github.io/econ-dashboards/supply-demand-explorer.html" width="100%" height="700"></iframe>
```

LMS	Where the embed button lives
Brightspace (D2L)	Content → Create a File → **Insert Stuff** → Enter Embed Code
Canvas	Pages → Edit → **Insert** → **Embed** (or the <> HTML editor)
Blackboard	Build Content → Item → **<> source-code** button (Ultra: Document → Add HTML)

Path B — no hosting at all: upload the `.html` straight into the LMS (Brightspace renders it interactively as a topic; Canvas/Blackboard via Files), or just email it — it works offline by double-clicking.

Why the iframe trick is universal: LMS editors *strip* pasted JavaScript (security), but they all *allow* iframes.

Not Just Intro Micro: The Recipe Is Course-Agnostic

Same prompt skeleton — swap the concept, the controls, and the quiz:

Micro & policy

Tax incidence & DWL · price ceilings/floors · elasticity sandbox · monopoly markup · game-theory payoff matrices

Not Just Intro Micro: The Recipe Is Course-Agnostic

Same prompt skeleton — swap the concept, the controls, and the quiz:

Micro & policy

Tax incidence & DWL · price ceilings/floors · elasticity sandbox · monopoly markup · game-theory payoff matrices

Macro & metrics

AD–AS & policy shocks · Solow growth · Phillips curve · OLS fit-a-line-by-hand · sampling distributions & power

Not Just Intro Micro: The Recipe Is Course-Agnostic

Same prompt skeleton — swap the concept, the controls, and the quiz:

Micro & policy

Tax incidence & DWL · price ceilings/floors · elasticity sandbox · monopoly markup · game-theory payoff matrices

Macro & metrics

AD–AS & policy shocks · Solow growth · Phillips curve · OLS fit-a-line-by-hand · sampling distributions & power

Field courses

Health: insurance & moral hazard · Labor: minimum-wage DiD explorer · Trade: comparative-advantage PPFs · Enviro: cap-and-trade

Not Just Intro Micro: The Recipe Is Course-Agnostic

Same prompt skeleton — swap the concept, the controls, and the quiz:

Micro & policy

Tax incidence & DWL · price ceilings/floors · elasticity sandbox · monopoly markup · game-theory payoff matrices

Macro & metrics

AD–AS & policy shocks · Solow growth · Phillips curve · OLS fit-a-line-by-hand · sampling distributions & power

Field courses

Health: insurance & moral hazard · Labor: minimum-wage DiD explorer · Trade: comparative-advantage PPFs · Enviro: cap-and-trade

The constraint is no longer "can I build it?"
— it's "**what do I want students to *do*?**"

Pedagogy first: predict → manipulate → explain. The tool just makes that loop cheap.

Takeaways

1. Papers $\rightarrow$ reading list $\rightarrow$ notes $+$ slides $+$ exams in **minutes**. You stay the editor.

Takeaways

1. Papers → reading list → notes + slides + exams in **minutes**. You stay the editor.
2. **Repeat it? Skill it. One-off? Prompt it.** The craft is in the specification.

Takeaways

1. Papers → reading list → notes + slides + exams in **minutes**. You stay the editor.
2. **Repeat it? Skill it. One-off? Prompt it.** The craft is in the specification.
3. Interactive tools are **one file away** — and one iframe line from any LMS.

Takeaways

1. Papers → reading list → notes + slides + exams in **minutes**. You stay the editor.
2. **Repeat it? Skill it. One-off? Prompt it.** The craft is in the specification.
3. Interactive tools are **one file away** — and one iframe line from any LMS.
4. **Dry-run everything.** Our test run caught a real bug — before your 9am section did.

Takeaways

1. Papers → reading list → notes + slides + exams in **minutes**. You stay the editor.
2. **Repeat it? Skill it. One-off? Prompt it.** The craft is in the specification.
3. Interactive tools are **one file away** — and one iframe line from any LMS.
4. **Dry-run everything.** Our test run caught a real bug — before your 9am section did.

Slides, skills, prompts & the dashboard: **thinkingwithagents.github.io**

Thank You!

Go build something for your fall course

Emily Beam & Erkmen G. Aslim

Department of Economics, University of Vermont

Slides, skills & demo links:

`thinkingwithagents.github.io`