

AI-Powered Research Workshop: From Question to Insight

A Hands-On Journey Through Modern Research Tools

Research Innovation Lab

Table of contents

1	Welcome: Your Research Journey Starts Here	4
1.1	The Hook	4
1.2	Workshop Philosophy	5
1.2.1	How This Workshop Works	5
1.3	Choosing Your AI Model	5
1.3.1	Available FREE Models via OpenRouter (2025)	5
1.3.2	Recommended Models for This Workshop	6
2	Module 1: The Conversation That Understands You	8
2.1	The Hook	8
2.2	The Story	8
2.2.1	Meet Dr. Emma Bakker	8
2.3	The Educational Piece	9
2.3.1	What is OpenWebUI?	9
2.3.2	The Big Idea	9
2.4	The Empowering Piece	9
2.4.1	Your First AI Conversation	9
2.4.2	Test Case #1: First Contact	9
2.4.3	Going Deeper: Why This Matters	11
2.5	The Entertainment Piece	11
2.5.1	Model Comparison Exercise #1: Speed vs Reasoning	11
2.5.2	The “AI Whisperer” Challenge	12
2.6	Key Learnings	12
2.6.1	What You Now Know	12
2.6.2	Questions You Should Answer	12
2.7	The Anecdote	13
2.8	The Quote	13

3	Module 2: Teaching Your AI to Read Your Papers	13
3.1	The Hook	13
3.2	The Story	13
3.2.1	Meet Professor James Okonkwo	13
3.3	The Educational Piece	14
3.3.1	What is RAG (Retrieval Augmented Generation)?	14
3.3.2	Why This Changes Everything	14
3.4	The Empowering Piece	14
3.4.1	Your First Document Upload	14
3.4.2	Test Case #2: Document Intelligence	15
3.4.3	Advanced Challenge	16
3.5	The Entertainment Piece	16
3.5.1	Model Comparison Exercise #2: Citation Accuracy Test	16
3.5.2	The “Citation Scavenger Hunt”	17
3.6	Key Learnings	17
3.6.1	What You Now Know	17
3.6.2	The Technical Magic (Optional Deep Dive)	18
3.7	Questions You Should Answer	18
3.8	The Anecdote	18
3.9	The Quote	19
4	Module 3: Asking the Internet for Help (Without Opening 50 Tabs)	19
4.1	The Hook	19
4.2	The Story	19
4.2.1	Meet Dr. Javier Torres	19
4.3	The Educational Piece	20
4.3.1	What is Web Search Integration?	20
4.3.2	When to Use Web Search	20
4.4	The Empowering Piece	20
4.4.1	Your First Web Search Query	20
4.4.2	Test Case #3: Real-Time Research	21
4.5	The Entertainment Piece	22
4.5.1	The “Time Traveler” Game	22
4.6	Key Learnings	22
4.6.1	What You Now Know	22
4.6.2	Behind the Scenes: SearxNG	23
4.7	Questions You Should Answer	23
4.8	The Anecdote	23
4.9	The Quote	23
5	Module 4: From Question to Graph in 60 Seconds	24
5.1	The Hook	24

5.2	The Story	24
5.2.1	Meet Alex Kowalski	24
5.3	The Educational Piece	25
5.3.1	What is Code Execution?	25
5.3.2	What Can You Do With This?	25
5.4	The Empowering Piece	25
5.4.1	Your First Code Execution	25
5.4.2	Test Case #4: Calculator on Steroids	25
5.4.3	Advanced Challenge	28
5.5	The Entertainment Piece	28
5.5.1	Model Comparison Exercise #3: Code Quality Battle	28
5.5.2	The “Visualization Remix” Game	29
5.6	Key Learnings	30
5.6.1	What You Now Know	30
5.6.2	The Technical Flow	30
5.7	Questions You Should Answer	31
5.8	The Anecdote	31
5.9	The Quote	31
6	Module 5: Bringing It All Together - The Research Power Move	32
6.1	The Hook	32
6.2	The Story	32
6.2.1	Meet Dr. Yuki Tanaka	32
6.3	The Educational Piece	33
6.3.1	The Integrated Research Workflow	33
6.4	The Empowering Piece	33
6.4.1	Your Final Challenge: The Complete Research Cycle	33
6.4.2	Test Case #5: The Grand Finale	34
6.4.3	Real-World Application	35
6.5	The Entertainment Piece	35
6.5.1	ULTIMATE Model Comparison Exercise: The Perfect Workflow	35
6.5.2	The “Research Relay Race”	37
6.6	Key Learnings	38
6.6.1	What You Now Know	38
6.6.2	The Meta-Skills: Prompt Engineering & Model Selection	38
6.7	Questions You Should Answer	38
6.8	The Anecdote	39
6.9	The Quote	39
7	Workshop Conclusion: Your Next Steps	40
7.1	What You’ve Accomplished Today	40
7.1.1	Your Model Arsenal (Quick Reference)	40
7.2	The 3-Day Challenge	40

7.3	Common Pitfalls (And How to Avoid Them)	41
7.3.1	Pitfall 1: Treating AI Like Google	41
7.3.2	Pitfall 2: Not Verifying Citations	41
7.3.3	Pitfall 3: Asking It to Do Everything	41
7.3.4	Pitfall 4: Forgetting It's a Tool, Not Magic	41
7.3.5	Pitfall 5: Using the Wrong Model for the Task NEW!	41
7.3.6	Pitfall 6: Staying with One Model the Whole Time NEW!	41
7.4	Resources for Continued Learning	42
7.5	Final Reflection Questions	42
7.6	The Last Quote	42
8	Appendix: Quick Reference Guide	42
8.1	Checklist for Each Module	42
8.1.1	Module 1: Basic Chat	42
8.1.2	Module 2: RAG	43
8.1.3	Module 3: Web Search	43
8.1.4	Module 4: Code Execution	43
8.1.5	Module 5: Integration	43
8.2	Troubleshooting Guide	43
8.3	Model Selection Cheat Sheet	44
8.3.1	Quick Decision Tree	44
8.3.2	Detailed Model Comparison	44
8.4	Command Cheat Sheet	45
8.4.1	Basic Prompts	45
8.4.2	RAG Prompts	45
8.4.3	Web Search Prompts	45
8.4.4	Code Execution Prompts	45
8.4.5	Integration Prompts	45
8.5	Keyboard Shortcuts	46
8.6	Getting Help	46
9	Acknowledgments	46
10	About This Workshop	47

1 Welcome: Your Research Journey Starts Here

1.1 The Hook

Imagine it's 2 AM. You're staring at 47 open browser tabs, three PDFs you can't find anymore, and a half-written Python script that crashed two hours ago. Sound familiar?

Today, you'll learn to do all of that in one place, in under 5 minutes.

1.2 Workshop Philosophy

“The goal is not to learn everything about AI tools. The goal is to leave here with ONE workflow that saves you 3 hours this week.” - Workshop Design Principle

1.2.1 How This Workshop Works

We follow the “**Learning Loop**” method: 1. **Hook** - Why should you care? 2. **Story** - See it in action 3. **Practice** - Do it yourself 4. **Test** - Prove it works 5. **Reflect** - What did you learn? 6. **Apply** - Use it tomorrow

Time Investment

This workshop is designed for **2.5 hours** with breaks. Each module is 20-30 minutes. You can stop after any module and come back later - each one stands alone.

1.3 Choosing Your AI Model

This workshop uses **FREE models from OpenRouter** - powerful AI models available at no cost. OpenRouter provides access to cutting-edge open-source and free models through a single interface.

1.3.1 Available FREE Models via OpenRouter (2025)

Your OpenWebUI instance has access to these **FREE** models:

Model	Best For	Speed	Quality	Context Window*
DeepSeek Chat V3 (deepseek-chat-v3:free)	Code generation, technical tasks	Fast	Excellent	128K tokens (~300 pages)
Gemini 2.5 Flash (gemini-2.5-flash:free)	Quick reasoning, fast responses	Very Fast	Very Good	1M tokens (~2,500 pages!)
Llama 4 Maverick (llama-4-maverick:free)	Complex reasoning, long context	Good	Excellent	256K tokens (~600 pages)

Model	Best For	Speed	Quality	Context Window*
Mistral Small (mistral-small:free)	Balanced tasks, citations	Fast	Very Good	96K tokens (~240 pages)

*Context Window = How much text the AI can “remember” at once (including your conversation + uploaded documents)

1.3.2 Recommended Models for This Workshop

Different models excel at different tasks! In this workshop, you’ll learn which model to use when:

- **DeepSeek Chat V3** - BEST for Python code generation and technical analysis
- **Gemini 2.5 Flash** - BEST for quick questions and web search synthesis
- **Llama 4 Maverick** - BEST for analyzing long documents and complex reasoning
- **Mistral Small** - BEST for balanced tasks and accurate citations

You’ll switch between models throughout the workshop to see their strengths!

About OpenRouter FREE Models

OpenRouter provides access to multiple FREE AI models through API calls. Your workshop organizer has configured access to free models only.

Benefits: - **100% FREE** - No costs, no credits needed - Access to cutting-edge open-source models - No individual API key setup needed - Works seamlessly through OpenWebUI interface - High-quality outputs comparable to paid models

Important Notes: - **Rate limits:** Free models have usage limits to prevent abuse - **What this means:** You can send 50-1000 messages per day (plenty for this workshop!) - **For this workshop:** Rate limits won’t affect you - we’ll use ~20-40 messages total - **If you hit a limit:** You’ll see a message like “Rate limit exceeded” - just wait a few minutes or try a different model - All free models work through OpenRouter API (your facilitator configured this) - You’ll learn which model is best for which task throughout the workshop!

How to Select Your Model

1. Look for the **model dropdown** at the top of the chat interface
2. Look for FREE models with names ending in **:free**:
 - “deepseek-chat-v3:free” or “DeepSeek Chat V3 Free”

- “gemini-2.5-flash:free” or “Gemini 2.5 Flash Free”
- “llama-4-maverick:free” or “Llama 4 Maverick Free”
- “mistral-small:free” or “Mistral Small Free”

3. **You’ll use DIFFERENT models for different tasks** in this workshop

4. **Avoid local models** (llama3.2, phi3, qwen) - they’re CPU-based and too slow

If you don’t see OpenRouter FREE models: Check with your facilitator - OpenRouter may need to be configured.

⚠ Common Confusions - Read This First!

“**Do I need to know how to code?**” - NO! The AI writes the code for you. You just describe what you want.

“**Will switching models delete my conversation?**” - NO! Your conversation stays when you switch models. The new model can see your history.

“**What if I don’t have any PDFs?**” - We provide sample documents, or you can use any PDF you have.

“**Are these really free?**” - YES! 100% free, no credit card, no tricks. OpenRouter provides these at no cost.

“**What if I make a mistake?**” - Just start a new chat! Click “New Chat” anytime.

“**Will I remember all this?**” - You’ll get reference guides to take home. Focus on understanding the concepts today.

! Quick Model Selection Guide

Use this throughout the workshop:

Task Type	Best Model	Why
Quick questions, simple lookups	Gemini 2.5 Flash	Fastest, great for simple tasks
Web search & current info	Gemini 2.5 Flash	Built for web content, fast synthesis
Document analysis, long texts	Llama 4 Maverick	256K context, best reasoning
Citations & academic accuracy	Mistral Small	Excellent citation precision

Python code & data analysis

DeepSeek Chat V3

Trained
specifically on
code

Complex multi-step reasoning

Llama 4 Maverick

Best logical
reasoning

Pro tip: You can switch models mid-conversation! Start with Gemini for quick exploration, switch to DeepSeek for code, switch to Mistral for citations.

2 Module 1: The Conversation That Understands You

2.1 The Hook

Quick quiz: How many times have you copied the same prompt into ChatGPT today?

What if your AI remembered everything you told it - for the entire project?

2.2 The Story

2.2.1 Meet Dr. Emma Bakker

Dr. Emma Bakker researches circular economy in food systems, specifically bread waste valorization. Last month, she spent 6 hours copying data between ChatGPT (for questions), Jupyter (for analysis), and Google Docs (for notes) while analyzing bakery supply chain data.

Then she discovered OpenWebUI. Now she does everything in one conversation: - Ask questions about bread waste statistics - Analyze data from 50+ bakeries - Compare her findings with Toast Ale's circular economy model - Share the entire workflow with her research partner in Amsterdam

Her time investment to learn: 15 minutes **Her time saved per week:** 4.5 hours **Real impact:** She discovered that 51% of bread waste happens at the supplier-retailer interface - exactly what recent 2024 research confirmed!

2.3 The Educational Piece

2.3.1 What is OpenWebUI?

Think of OpenWebUI as your research lab, but digital:

Traditional Lab	OpenWebUI Lab
Multiple tools on different benches	Everything in one interface
Write down every step manually	Auto-saves your entire process
Can't replay experiments exactly	Perfect reproducibility
Work alone or email files	Real-time collaboration

2.3.2 The Big Idea

Context persistence - The AI remembers everything in your conversation. No more “As I mentioned before...” because you switched tools.

2.4 The Empowering Piece

2.4.1 Your First AI Conversation

Learning Objective: Send a message and get a response in under 2 minutes.

Why this matters: This is your foundation. Every advanced feature builds on this.

2.4.2 Test Case #1: First Contact

Setup checklist:

```
OpenWebUI is open in your browser
You can see the chat interface
Model is set to a FREE OpenRouter model (ending in :free)
```

The Test:

1. **Open your browser** and go to: `https://team1-openwebui.valuechainhackers.xyz`
2. **Create your account** (first signup becomes admin):

Email: [your email]

Password: [strong password]

3. Select your model:

WHERE TO LOOK:

- Look at the **TOP** of the screen in the chat interface
- You'll see a dropdown menu or button showing the current model name
- It might say "Select Model" or show a model name like "llama3.2"

HOW TO SELECT:

- Click on the model name/dropdown
- A list of available models appears
- Scroll through to find "**Gemini 2.5 Flash Free**" (fastest for first contact)
- The exact name might be:
 - "gemini-2.5-flash:free"
 - "Google Gemini 2.5 Flash (Free)"
 - "Gemini Flash Free"
- Click to select it

IMPORTANT:

- Look for models ending in **:free** or with "Free" in the name
- Do NOT select local models like "llama3.2", "phi3", or "qwen" - they're too slow for this workshop

4. Send this exact message:

Hello! Please respond with exactly these words:

"TEST SUCCESSFUL - I am ready to help with your research."

Then tell me which AI model you are and what you're best at.

5. Expected result:

- Response within 3-7 seconds (Gemini Flash is FAST!)
- Contains "TEST SUCCESSFUL" (or very close to it)
- Identifies as Gemini 2.5 Flash
- High-quality, well-formatted response

Success criteria: - [] Message sent without errors - [] Response received in < 10 seconds
- [] Response contains requested text (exactly or very close) - [] Model identifies as a FREE OpenRouter model (NOT llama/phi/qwen)

If it fails: - Check you're logged in (see your email in top right?) - Check model dropdown - should show a model ending in **:free** - If you see a local model name, switch to a FREE OpenRouter model - Try refreshing the page - Ask facilitator if OpenRouter is configured

2.4.3 Going Deeper: Why This Matters

When you sent that message, here's what happened:

```
graph LR
  A[Your Browser] -->|Sends message| B[OpenWebUI Server]
  B -->|Forwards to| C[AI Model]
  C -->|Generates response| B
  B -->|Streams back| A
```

The magic: Your conversation is now saved. Close your browser, come back tomorrow - it's still there.

2.5 The Entertainment Piece

2.5.1 Model Comparison Exercise #1: Speed vs Reasoning

Goal: Discover which model is best for quick questions vs complex analysis.

Part A: Quick Question (Test Gemini 2.5 Flash)

HOW TO SWITCH MODELS: 1. Look at the top of your chat interface 2. Find the **dropdown menu** that shows your current model name 3. Click on it to see all available models 4. Select “**Gemini 2.5 Flash Free**” (or similar name with “Gemini” and “Flash”) 5. The page may refresh - that's normal!

Now ask:

What is the capital of Brazil?

Note the response time (should be 2-4 seconds)

Part B: Complex Reasoning (Test Llama 4 Maverick)

Now switch models using the same steps above, but select “**Llama 4 Maverick Free**”

Important: When you switch models, your conversation history stays! The new model can see your previous messages.

Now ask:

Explain the relationship between circular economy principles and carbon reduction targets in urban planning. Use specific examples.

Compare: - [] Gemini was faster for the simple question - [] Llama provided deeper, more nuanced reasoning for the complex question - [] Both gave correct answers, but different levels of detail

Key Insight: Use **Gemini Flash** for quick lookups, **Llama Maverick** for complex thinking!

2.5.2 The “AI Whisperer” Challenge

Game: See how specific you can make the AI’s response.

Try these with Gemini Flash (for speed): 1. “Respond with a haiku about research.” 2. “List exactly 3 reasons why coffee is essential for research. Number them.” 3. “Explain your last answer, but make it sound like a pirate.”

Point: The more specific your instructions, the better your results. This applies to everything you’ll do today.

2.6 Key Learnings

2.6.1 What You Now Know

1. **OpenWebUI saves context** - No more copying between tools
2. **Specificity matters** - Clear instructions = better results
3. **It’s conversational** - You can refine, ask follow-ups, iterate

2.6.2 Questions You Should Answer

1. Where does your conversation history save? (*Answer: In OpenWebUI’s database, tied to your account*)
2. Can you start multiple conversations? (*Answer: Yes! Click “New Chat” anytime*)
3. What happens if you close your browser? (*Answer: Nothing - log back in and continue*)

2.7 The Anecdote

When we first tested OpenWebUI with grad students, one of them said: “Wait, so I don’t lose my work if Chrome crashes?”

When we said yes, she literally got up and hugged her laptop.

She’d lost a 3-hour ChatGPT session the day before when her browser crashed.

That’s why we’re here.

2.8 The Quote

“The future of sustainability research is not just about finding answers, but about asking better questions, faster.” - Dr. Kate Raworth, Economist and author of “Doughnut Economics”

3 Module 2: Teaching Your AI to Read Your Papers

3.1 The Hook

Thought experiment: You have 50 PDFs about circular economy. You need to find every mention of “reverse logistics.”

- **Old way:** Open each PDF, Ctrl+F, copy excerpts, 2 hours
- **New way:** Upload all 50, ask “What do these papers say about reverse logistics?”, 2 minutes

That’s RAG. Let’s learn it.

3.2 The Story

3.2.1 Meet Professor James Okonkwo

James researches waste management in Lagos. He had a problem:

“I’ve collected 200 interviews, 80 policy documents, and 30 academic papers. I know the answer to my research question is somewhere in there, but I’ll spend 6 months finding it manually.”

He uploaded everything to OpenWebUI. His first question: *“What are the main barriers to waste sorting according to my interview data?”*

Time to answer: 30 seconds **Accuracy:** Cited 7 specific interviews with exact quotes

3.3 The Educational Piece

3.3.1 What is RAG (Retrieval Augmented Generation)?

The problem RAG solves: - AI models only know what they were trained on (data cutoff) - They can't read YOUR specific documents - They "hallucinate" - make up facts confidently

The RAG solution:

```
graph TD
  A[Your Question] --> B[Search your documents]
  B --> C[Find relevant chunks]
  C --> D[Give chunks to AI]
  D --> E[AI answers using YOUR documents]
  E --> F[Response with citations]
```

In plain English: 1. **You upload your PDFs** - Your documents are sent to OpenWebUI 2. **They get processed** - Documents are split into smaller sections (like paragraphs) and converted into a searchable format - Think of it like creating an index in the back of a book - Each section gets a unique "fingerprint" so it can be found quickly 3. **You ask a question** - OpenWebUI searches through all sections to find the most relevant ones 4. **AI reads the relevant sections** - Not the whole document, just the parts that match your question 5. **You get an answer with citations** - The AI tells you where it found the information

Key point: RAG doesn't require the AI to "remember" your entire document - it searches and retrieves the relevant parts when needed!

3.3.2 Why This Changes Everything

Before RAG: - "Tell me about X" → AI uses general knowledge (might be wrong)

With RAG: - "Tell me about X" → AI searches YOUR documents, quotes them, cites them

You're creating a custom AI that knows YOUR research.

3.4 The Empowering Piece

3.4.1 Your First Document Upload

Learning Objective: Upload a PDF and ask it a question it can only answer from that document.

3.4.2 Test Case #2: Document Intelligence

Setup checklist:

```
You have a PDF ready (research paper, report, anything)
You're logged into OpenWebUI
You're in a new chat
Model is set to "Llama 4 Maverick Free" (best for document analysis)
```

Why Llama 4 Maverick for this task? - **256K token context** - Can process up to ~600 pages at once (like 3 entire research papers!) - **Excellent at extracting specific information** - Won't miss details in long documents - **Strong citation accuracy** - Remembers where information came from

What does “256K tokens” mean? - A “token” is roughly 3/4 of a word - 256K tokens 190,000 words 600 pages - This means you can upload a 200-page thesis and it will read the ENTIRE thing!

Don't have a PDF handy? Download this sample:

```
# Sample sustainability report
https://www.ellenmacarthurfoundation.org/circular-economy-diagram
```

The Test:

Step 1: Upload Your Document

1. Look for the **paperclip** or “+” icon in the chat input area (usually bottom left or right)
2. Click it to open the file selector
3. Select “Upload File” or “Choose File”
4. Choose your PDF from your computer
5. **Wait for processing** - You'll see:
 - A progress bar or loading animation
 - The filename appearing in your chat
 - A green checkmark when complete
 - OR a message like “Document processed successfully” or “Ready to query”

How long? Small PDFs (10-20 pages): 5-15 seconds. Large PDFs (100+ pages): 30-60 seconds.

Expected: Green checkmark , filename visible, ready to ask questions!

Step 2: Test RAG is Working

Send this message:

Based on the document I just uploaded, what are the top 3 main points?
Use bullet points and quote specific sections.

Step 3: Verify Citations

Look for: - [] Answer is specific to YOUR document (not general knowledge) - [] You see citations or page references - [] If you uploaded a circular economy doc, it mentions specific frameworks from that doc

Step 4: Test the Limits

Try this:

What does this document say about quantum computing?

Expected result: AI says something like “This document doesn’t contain information about quantum computing” or “I don’t see any references to quantum computing in the uploaded document.”

Why this test matters: You’re verifying RAG stays grounded in your documents and doesn’t hallucinate.

3.4.3 Advanced Challenge

Upload a second document and ask:

What are the similarities and differences between these two documents?

This tests: Multi-document RAG (searching across multiple sources)

3.5 The Entertainment Piece

3.5.1 Model Comparison Exercise #2: Citation Accuracy Test

Goal: See which model is most accurate with citations and document analysis.

Setup: Keep your uploaded document open in a chat.

Part A: Test with Mistral Small (Citation Specialist)

Switch to **Mistral Small Free** and ask:

List 5 specific claims from this document with exact quotes and page numbers.

Part B: Test with Gemini Flash (Speed Reader)

Switch to **Gemini 2.5 Flash Free** and ask the same question:

List 5 specific claims from this document with exact quotes and page numbers.

Compare the results: - [] Which model provided more accurate page numbers? - [] Which model gave longer, more complete quotes? - [] Which model was faster? - [] Which model better understood context?

Expected Results: - **Mistral Small** should give more precise citations - **Gemini Flash** should be faster but may be less precise - Both should correctly avoid making up content

Key Insight: Use **Mistral Small** when citation accuracy is critical, **Gemini Flash** when you need quick document summaries!

3.5.2 The “Citation Scavenger Hunt”

Game: Upload a document and challenge a partner:

1. Each person asks a question about the document
2. The AI answers with citations
3. First person to find the actual quote in the PDF wins
4. **Bonus points** if the AI got it wrong (finding AI mistakes is a skill!)

Why this is fun: You’re learning to trust AND verify AI responses.

3.6 Key Learnings

3.6.1 What You Now Know

1. **RAG grounds AI in YOUR documents** - No more hallucination about your research
2. **Citations are your friend** - Always verify claims against sources
3. **Multiple documents work** - Upload your entire literature review
4. **Embeddings are “searchable math”** - Your docs become queryable knowledge

3.6.2 The Technical Magic (Optional Deep Dive)

When you upload a PDF:

```
# Simplified version of what happens
document = extract_text_from_pdf("your_paper.pdf")
chunks = split_into_paragraphs(document)

for chunk in chunks:
    embedding = create_vector(chunk) # Turns text into numbers
    store_in_database(embedding, chunk) # Saves in Qdrant
```

When you ask a question:

```
question_embedding = create_vector("your question")
similar_chunks = search_database(question_embedding) # Math finds similar vectors
ai_response = generate_answer(question, similar_chunks) # AI reads chunks
```

You don't need to understand this. But knowing it exists helps you understand why: - Large documents take longer to process - More documents = better coverage - Good questions get better chunks

3.7 Questions You Should Answer

1. What does “RAG” stand for? (*Retrieval Augmented Generation*)
2. Why doesn't the AI just “remember” your PDF after uploading? (*It needs to search it each time - that's how it stays accurate*)
3. Can you upload Word docs? Images? (*Depends on configuration - PDF definitely works*)
4. What happens if you ask about something NOT in your documents? (*Good RAG says “I don't know” - bad RAG hallucinates*)

3.8 The Anecdote

During a pilot test, a PhD student uploaded her entire thesis draft (200 pages). She asked: “What's my main argument?”

The AI summarized it in 3 sentences. She stared at the screen, then said: “That's... that's actually what I've been trying to say for 4 years.”

She used that as her elevator pitch for the next conference.

Sometimes the AI sees patterns we're too close to notice.

3.9 The Quote

“We are drowning in information, while starving for wisdom. The world henceforth will be run by synthesizers, people able to put together the right information at the right time, think critically about it, and make important choices wisely.” - E.O. Wilson, Biologist

4 Module 3: Asking the Internet for Help (Without Opening 50 Tabs)

4.1 The Hook

Quick scenario:

You’re writing a paper about sustainable beer brewing and circular economy. You need: - Latest 2024-2025 brewery sustainability initiatives - Current water-to-beer ratios for sustainable breweries - Toast Ale’s verified CO2 savings from using surplus bread - HEINEKEN’s net zero commitments

Your current workflow: 1. Google search “sustainable brewing 2024” 2. Open 15 tabs (company reports, research papers, news articles) 3. Read each one, take notes 4. Lose track of which tab had Toast Ale’s exact CO2 numbers 5. Find the stat, accidentally close the tab 6. Can’t remember which site it was from 7. Start over

New workflow: Ask OpenWebUI with web search. Get answer with sources. Done in 2 minutes.

4.2 The Story

4.2.1 Meet Dr. Javier Torres

Javier studies sustainable cocoa sourcing and deforestation. His research question: “*What progress has Barry Callebaut made on farm-level traceability in 2024?*”

This needs CURRENT data. His literature is from 2022. The Forever Chocolate initiative releases new reports annually.

He enabled web search in OpenWebUI and asked:

What is Barry Callebaut's latest progress on cocoa farm traceability and deforestation-free sourcing in their 2023/24 Forever Chocolate report? Include specific numbers for farm plots mapped and percentage traceable.

Result: - AI searched the web - Found the 2023/24 Forever Chocolate Progress Report - Extracted specific data: 669,174 farm plots mapped, 46.5% deforestation-free - Synthesized answer with citations - Included links to official reports and CFI updates

Time saved: 2 hours of manual googling through corporate sustainability reports

4.3 The Educational Piece

4.3.1 What is Web Search Integration?

The problem it solves: - AI models have a knowledge cutoff (e.g., “I only know data up to April 2024”) - Your research questions need current information - Switching between ChatGPT and Google breaks your flow

The solution:

```
graph TD
  A[Your Question] --> B{Does this need current info?}
  B -->|Yes| C[Search the web via SearxNG]
  B -->|No| D[Use AI knowledge]
  C --> E[Get results]
  E --> F[AI reads search results]
  F --> G[Synthesizes answer]
  G --> H[Response with links]
```

4.3.2 When to Use Web Search

Good use cases: - Current events, policies, statistics - Latest research publications - Real-time data (prices, weather, trends) - Fact-checking recent claims

Bad use cases: - General knowledge questions (“What is photosynthesis?”) - Math calculations (waste of a web search) - Questions about YOUR documents (use RAG instead)

4.4 The Empowering Piece

4.4.1 Your First Web Search Query

Learning Objective: Ask a question requiring current information and get a cited answer.

4.4.2 Test Case #3: Real-Time Research

Setup checklist:

```
You're in OpenWebUI  
You're in a new chat  
Web search is enabled (check settings or toggle)  
Model is set to "Gemini 2.5 Flash Free" (BEST for web search)
```

Why Gemini 2.5 Flash for this task? - Fastest response time for web queries - Excellent at synthesizing multiple sources - 1M token context can handle many search results - Built by Google, optimized for web content

How to enable web search:

Option 1: Toggle Button (Most Common) 1. Look near the message input box at the bottom of the screen 2. Find a **globe icon** or button labeled “Web Search” or “Search” 3. Click it once - it should change color (gray → blue/green = enabled) 4. You’ll see a small badge or text saying “Web search enabled” or the icon will be highlighted

Option 2: Settings Menu 1. Click the **settings icon** (or three dots) near the chat 2. Look for “Enable Web Search” checkbox 3. Check the box 4. Close settings

How to tell it’s working: When you send a message, you’ll see “Searching the web...” or “Searching...” before the response

The Test:

Step 1: Ask a Current Question

Send this:

```
What are the latest (2024-2025) sustainability initiatives in the brewing industry? Include specific companies, their water reduction targets, and verified CO2 savings. Focus on circular economy approaches.
```

Expected behavior: - ☐ You see “Searching the web...” indicator - ☐ Wait 5-15 seconds (web search takes longer) - ☐ Response includes recent dates (2024-2025) - ☐ Response includes URLs or source names

Step 2: Verify It’s Actually Using Web Search

Try this control test - **disable web search** and ask:

```
What happened in sustainability news yesterday?
```

Expected result: “I don’t have access to information about yesterday” or similar.

Now **enable web search** again and ask the same question.

Expected result: Actual recent news with dates and sources.

Step 3: Compare Quality

Ask this WITH web search enabled:

How much CO2 has Toast Ale saved by using surplus bread instead of malted barley? What is their verified 2023 carbon footprint?

You should get: - [] Specific number (5.3 tonnes CO2e avoided in 2023) - [] Their total footprint (150 tCO2e in 2023) - [] A source URL (toastbrewing.com or similar) - [] Context about their circular economy model

This is information the AI cannot know without searching - it’s from Toast Ale’s 2024 sustainability report!

4.5 The Entertainment Piece

4.5.1 The “Time Traveler” Game

Challenge: Ask questions that reveal whether AI is using web search or old knowledge.

Example questions: 1. “Who won the Nobel Prize in Economics this year?” 2. “What’s the latest IPCC report recommendation?” 3. “How much has global temperature risen since 2020?”

How to play: - Disable web search → Note the answer - Enable web search → Note the answer - Compare!

Point: This helps you understand when AI is guessing vs. searching.

4.6 Key Learnings

4.6.1 What You Now Know

1. **Web search extends AI’s knowledge cutoff** - Get current information
2. **Citations are automatic** - No more bookmarking 50 tabs
3. **Synthesis is the value** - AI reads multiple sources and summarizes
4. **Toggle mindfully** - Not every question needs web search

4.6.2 Behind the Scenes: SearxNG

OpenWebUI uses **SearxNG** - a privacy-respecting meta-search engine that: - Searches multiple sources (Google, Bing, DuckDuckGo, etc.) - Doesn't track you - Returns results to the AI - AI synthesizes them into an answer

Why this matters: You get broad search coverage without being tracked across the internet.

4.7 Questions You Should Answer

1. When should you enable web search? (*When you need current info or real-time data*)
2. Why does web search take longer than normal AI responses? (*It has to search the web, fetch results, then read them*)
3. Can you trust web search results? (*Verify sources - AI can misinterpret search results*)
4. What's the difference between web search and RAG? (*RAG = your documents, Web search = the internet*)

4.8 The Anecdote

A food systems researcher was writing a grant proposal about circular economy in brewing. She needed current examples of breweries using waste bread.

She asked OpenWebUI with web search: “*Which breweries are using surplus bread to replace malted barley in 2024? What are their verified environmental impacts?*”

The AI found: - Toast Ale's complete circular economy model - Their verified 2023 data (2.9 million slices, 5.3 tonnes CO2e saved) - Links to their sustainability report and Ellen MacArthur Foundation case study - Similar initiatives in other countries

She got 4 citations with specific numbers for her grant proposal in 5 minutes.

The grant got funded. Her research now analyzes scaling potential for bread-to-beer circular models across Europe.

4.9 The Quote

“In the age of information abundance, the synthesis of knowledge is more valuable than access to data.” - Dr. Johan Rockström, Climate Scientist and Director of the Potsdam Institute

5 Module 4: From Question to Graph in 60 Seconds

5.1 The Hook

Pop quiz: How long does it take you to: 1. Export data from Excel 2. Open Jupyter/Python 3. Load the data 4. Write code to analyze it 5. Generate a graph 6. Interpret results

Average answer: 30-45 minutes

What if it took 60 seconds?

5.2 The Story

5.2.1 Meet Alex Kowalski

Alex is a supply chain researcher studying delivery route efficiency. He has a CSV of 500 delivery routes with distances and times.

His old workflow:

```
# 1. Open Jupyter
# 2. Import libraries (pray they're installed)
import pandas as pd
import matplotlib.pyplot as plt
# 3. Load data (fight with file paths)
df = pd.read_csv('data.csv')
# 4. Clean data (find the error)
# 5. Plot (Google the syntax)
# 6. Interpret
```

Time: 45 minutes (if everything works)

His new workflow in OpenWebUI:

Here's my delivery data [uploads CSV]. Show me the correlation between distance and time, create a scatter plot, and tell me if there are any outliers I should investigate.

Time: 90 seconds

5.3 The Educational Piece

5.3.1 What is Code Execution?

The problem it solves: - You have data but aren't a Python expert - You know what question you want answered, not how to code it - You switch between ChatGPT (for code) and Jupyter (to run it)

The solution:

```
graph LR
  A[Your Question] --> B[AI writes Python code]
  B --> C[Code executes in Jupyter]
  C --> D[Results/Graphs return]
  D --> E[AI interprets results]
  E --> F[You get insights]
```

The magic: Everything happens in one conversation. No copying code. No switching tools.

5.3.2 What Can You Do With This?

Data analysis: - Descriptive statistics - Correlation analysis - Hypothesis testing - Time series analysis

Visualization: - Scatter plots, line graphs, bar charts - Heatmaps, distributions - Custom plots

Data processing: - Cleaning messy data - Merging datasets - Transforming formats - Calculating derived metrics

5.4 The Empowering Piece

5.4.1 Your First Code Execution

Learning Objective: Ask AI to perform a calculation and see the code run automatically.

5.4.2 Test Case #4: Calculator on Steroids

Setup checklist:

```
You're in OpenWebUI
Code execution is enabled (should be by default)
You're in a new chat
Model is set to "DeepSeek Chat V3 Free" (BEST for code generation)
```

Why DeepSeek Chat V3 for this task? - Specifically trained on code - Exceptional at Python - Excellent at debugging and explaining code - Strong mathematical reasoning - Great at data analysis tasks

The Test:

Step 1: Simple Calculation (AI Writes Code)

DeepSeek is excellent at writing code. Let's test it:

Calculate the compound annual growth rate (CAGR) if an investment grows from \$100 to \$250 over 5 years. Show me the formula and write Python code to calculate it, then execute the code.

What you'll see (step by step):

1. **AI explains the formula** - Text explanation of CAGR
2. **AI writes code** - You'll see a Python code block (gray/dark box with colored text)
3. **Code runs automatically** - Look for:
 - A "Running..." or "Executing code..." message
 - The code block may get a green border or checkmark
 - Results appear below the code (either text output or "CAGR: 20.11%")
4. **AI interprets results** - Explains what the answer means

What does code execution look like? - Code appears in a **special box** with syntax highlighting (colors) - You might see a **"Run"** button or it runs automatically - Output appears **directly below the code** (not in a new message) - Result: ~20.11%

Example of what you'll see:

```
initial = 100
final = 250
years = 5
cagr = (final/initial)**(1/years) - 1
print(f"CAGR: {cagr*100:.2f}%")
```

The key insight: With good models, AI can WRITE and RUN code for you!

Step 2: Create a Visualization (AI Writes Code)

Now let's have AI create a graph for us:

Create a line graph showing exponential growth from \$100 to \$250 over 5 years using the CAGR we just calculated. Label the axes and make it look professional. Execute the code to generate the plot.

What you'll see: - [] AI writes matplotlib/plotting code (Python library for graphs) - [] Code executes automatically - [] **A graph appears!** You'll see one of these: - **Best case:** Image displays directly in the chat (inline) - **Alternative:** A message like "Chart created" or "Visualization saved" - **Or:** A clickable link/filename to view the image - [] Graph shows: - Growth curve from \$100 to \$250 - X-axis labeled (years: 0-5) - Y-axis labeled (dollars: \$100-\$250) - Professional appearance (colors, grid, title)

What if I don't see the image? - Don't worry! The code ran successfully - Some setups show images inline, others save them as files - The important learning: AI wrote code that creates professional visualizations - If there's a filename, you can download it

This is the magic moment - you went from question to visualization in ONE request, without knowing any matplotlib syntax!

Step 3: Test With Real Data Analysis (AI Generates Everything)

Now let's test AI's ability to do complete data analysis with REAL sustainability data:

Generate a realistic dataset of 10 breweries with:

- Water usage (liters water per liter beer produced)
- CO2 emissions (kg per hectoliter)
- Percentage of renewable energy used

Then create visualizations showing:

1. Scatter plot: water usage vs CO2 emissions
2. Bar chart: breweries ranked by sustainability score

Calculate correlation and identify which factors matter most.
Execute all the code.

Expected behavior: - [] AI generates sample data - [] Creates visualization - [] Calculates correlation (r value) - [] Interprets the relationship - [] All code executes successfully

This is powerful: You described what you want, AI figured out how to do it, wrote the code, executed it, and interpreted results.

5.4.3 Advanced Challenge

Upload your own data (CSV, Excel) and ask:

Analyze this dataset. Show me:

1. Summary statistics
2. Distribution of the main variables
3. Any interesting patterns or outliers
4. Create appropriate visualizations

Execute all code and explain what you find.

This tests: End-to-end research workflow with your real data.

Expected: AI explores your data, generates multiple analyses, creates visualizations, and provides insights.

5.5 The Entertainment Piece

5.5.1 Model Comparison Exercise #3: Code Quality Battle

Goal: See which model writes better Python code for data analysis.

Part A: Test DeepSeek Chat V3 (Code Specialist)

Switch to **DeepSeek Chat V3 Free** and ask:

Create a Python function that:

1. Generates 100 random data points following a normal distribution
2. Calculates mean, median, and standard deviation
3. Creates a histogram with the distribution curve overlay
4. Returns all statistics in a dictionary

Execute the code and show the results.

Observe: - Code quality and structure - Explanation clarity - Execution success - Visualization quality

Part B: Test Gemini Flash (Generalist)

Switch to **Gemini 2.5 Flash Free** and ask the **exact same question**.

Compare: - [] Which model's code was more readable? - [] Which model added better comments? - [] Which model created a better visualization? - [] Which model executed without errors? - [] Which was faster?

Expected Results: - **DeepSeek** should have cleaner, more professional code structure - **DeepSeek** should handle edge cases better - **Gemini** should be faster but code may be simpler - Both should work, but quality will differ

Key Insight: Use **DeepSeek Chat V3** for all serious coding tasks, **Gemini** for quick one-off calculations!

Part C: The “Impossible” Challenge (DeepSeek’s Specialty)

Now try something complex that really shows DeepSeek’s coding prowess. Use **DeepSeek Chat V3**:

Write a Python function that:

1. Simulates a Monte Carlo analysis for portfolio risk
2. Takes 3 stock tickers, investment amounts, and number of simulations
3. Uses random walk model for price movements
4. Calculates Value at Risk (VaR) at 95% confidence
5. Creates visualization showing distribution of outcomes
6. Returns comprehensive risk metrics

Use realistic parameters and execute the code.

Now try the SAME prompt with Gemini Flash.

What you’ll see: - **DeepSeek** will write sophisticated, production-quality code with proper error handling - **Gemini** will write simpler code that may work but lacks polish - **DeepSeek** will better explain the financial concepts - **DeepSeek** will create more professional visualizations

This is the DRAMATIC difference - for complex technical tasks, specialist models shine!

5.5.2 The “Visualization Remix” Game

Challenge: Take the same data and ask for it in different visualization styles.

Use **DeepSeek Chat V3** for this (best at visualization code)

Starting prompt:

Create sample data of monthly bread waste reduction initiatives at a large bakery over 12 months, showing:

- Surplus bread (kg)
- Bread donated to charity (kg)
- Bread upcycled for brewing (kg)
- Waste sent to landfill (kg)

Then request: 1. “Show this as a stacked area chart to see waste distribution over time” 2. “Show this as a line graph comparing the three circular economy routes” 3. “Create a pie chart showing the final year distribution” 4. “Make a before/after comparison: Month 1 vs Month 12”

Point: Learn what visualization best communicates your circular economy impact story.

5.6 Key Learnings

5.6.1 What You Now Know

1. **Code execution happens in Jupyter** - A real Python environment
2. **AI writes the code for you** - You describe what you want, it codes it
3. **Results stay in the conversation** - No more “where did I save that graph?”
4. **Iteration is fast** - “Now make the bars blue” → done
5. **Reproducibility is built-in** - All code is saved in the chat

5.6.2 The Technical Flow

When you ask for analysis:

```
# 1. AI interprets your question
user_intent = understand_request("show me correlation...")

# 2. AI writes Python code
code = generate_python_code(user_intent)

# 3. Code sent to Jupyter container
result = jupyter_kernel.execute(code)

# 4. Output captured (text, images, errors)
output = capture_output(result)

# 5. Displayed in chat
return_to_user(output)
```

Why Jupyter? It's a standardized Python environment that's: - Isolated (your code can't break the system) - Stateful (variables persist between executions) - Rich (can return plots, dataframes, etc.)

5.7 Questions You Should Answer

1. Where does the Python code actually run? (*In a Jupyter container on the server*)
2. Can you edit the code before it runs? (*Yes! You can copy it, modify it, and ask AI to run the modified version*)
3. What Python libraries are available? (*Common ones: pandas, numpy, matplotlib, scipy, scikit-learn*)
4. What if the code has an error? (*AI sees the error and usually fixes it automatically*)

5.8 The Anecdote

A PhD student in environmental science had 2 years of weather data in Excel. She needed to identify "heat wave events" (5+ consecutive days above 35°C).

She'd been manually highlighting cells for a week.

She asked OpenWebUI: *"Find all heat wave events in this data (5+ days above 35°C). Show me when they occurred and how long each lasted."*

60 seconds later: Complete analysis, a calendar heatmap visualization, and a table of all 17 heat wave events.

She called her supervisor and said, "I need to change my methodology section."

5.9 The Quote

"The real problem is not whether machines think but whether humans do." - B.F. Skinner (though he meant it differently, it applies to data analysis)

"Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom." - Clifford Stoll, Astronomer & Author

6 Module 5: Bringing It All Together - The Research Power Move

6.1 The Hook

The ultimate question: Can you use ALL four skills in ONE research workflow?

Answer: Yes. And it's spectacular.

6.2 The Story

6.2.1 Meet Dr. Yuki Tanaka

Yuki studies urban heat islands in Asian megacities. Her research question:

“How do urban green spaces correlate with temperature reduction, and what do recent policy implementations suggest about effectiveness?”

This requires: - RAG - her uploaded research papers - Web search - current policy implementations - Code execution - correlation analysis - Conversation - synthesis and interpretation

Her single conversation:

I've uploaded 15 papers about urban heat islands. Based on these, what's the theoretical cooling effect of increasing urban green space by 10%?

(AI uses RAG, quotes papers)

Now search the web for cities that implemented urban greening policies in 2023-2024. What were their results?

(AI searches web, finds Seoul, Singapore, Barcelona examples)

Here's temperature data from Singapore [uploads CSV]. Calculate the correlation between green space coverage and temperature reduction. Compare this to the theoretical predictions from the papers.

(AI executes code, creates visualizations, compares empirical vs theoretical)

Synthesize all of this into a 200-word summary I can use in my paper's introduction, with citations.

(AI writes the summary, drawing from papers, web sources, and data analysis)

Time: 15 minutes **Output:** Publication-ready paragraph with citations **Old method:** 2 weeks

6.3 The Educational Piece

6.3.1 The Integrated Research Workflow

What makes this powerful:

```
graph TD
  A[Research Question] --> B[RAG: Literature Review]
  B --> C[Web: Current Context]
  C --> D[Code: Data Analysis]
  D --> E[AI: Synthesis]
  E --> F[Actionable Insight]
  F -.Iterate.-> A
```

Each capability enhances the others:

Capability	Provides	Used By
RAG	Theoretical foundation	Code (variables to test), Web (context for search)
Web Search	Current examples	RAG (validate theories), Code (real-world data)
Code Execution	Empirical evidence	RAG (test theories), Web (inform what to search)
Conversation	Integration layer	Everything connects here

6.4 The Empowering Piece

6.4.1 Your Final Challenge: The Complete Research Cycle

Learning Objective: Use all four capabilities in one conversation to answer a research question.

6.4.2 Test Case #5: The Grand Finale

Setup checklist:

```
You have 1-2 PDFs related to your research area
You have a CSV of data (or can generate sample data)
You have a research question in mind
Web search is enabled
```

The Test:

Your mission: Answer this question using all four capabilities:

“What does the literature say about [YOUR TOPIC], what are current real-world examples, and what does the data show?”

Step-by-step guide:

Step 1: Upload Your Literature (RAG)

I'm researching [YOUR TOPIC]. I've uploaded papers about this.
What are the main findings regarding [SPECIFIC ASPECT]?

Expected: Summary with citations from YOUR papers.

Step 2: Add Current Context (Web Search)

Now search the web for recent (2024-2025) examples of [YOUR TOPIC]
being implemented. What are organizations doing?

Expected: Real-world examples with URLs.

Step 3: Analyze Data (Code Execution)

Here's data related to [YOUR TOPIC] [upload CSV or ask AI to generate sample data].
Analyze it and create a visualization showing [RELATIONSHIP YOU WANT TO EXPLORE].

Expected: Code execution, graph, statistical analysis.

Step 4: Synthesize Everything (Integration)

Based on:

1. The literature I uploaded
2. The current examples you found
3. The data analysis we just did

Write a 3-paragraph synthesis that answers: [YOUR RESEARCH QUESTION]
Include citations and reference the data visualization.

Expected: A coherent synthesis drawing from all three sources.

Success criteria: - [] Literature is cited correctly - [] Web sources are referenced with URLs - [] Data analysis is mentioned with specific numbers - [] Synthesis reads coherently (not just pasted together) - [] You can see the thread connecting all parts

If it fails: - Break it down - do each step separately first - Check each capability works individually before combining - Be more specific in your synthesis request

6.4.3 Real-World Application

Your homework: Use this workflow for an actual research task this week.

Ideas: 1. Literature review + current policy landscape + statistical validation 2. Historical data analysis + recent trends + predictive insights 3. Theoretical framework + case studies + empirical testing

6.5 The Entertainment Piece

6.5.1 ULTIMATE Model Comparison Exercise: The Perfect Workflow

Goal: Use the RIGHT model for each step of a complete research workflow.

The Challenge: Research the topic “Circular economy in beer brewing: using surplus bread to reduce environmental impact” using optimal models for each task.

Step 1: Web Search for Current Examples (Use Gemini 2.5 Flash)

Search for the latest (2024-2025) examples of breweries using surplus bread to replace malted barley. Focus on Toast Ale and their verified environmental impact data. Include specific CO2 savings and waste reduction numbers.

Expected: Fast response with Toast Ale's data (2.9M slices, 5.3t CO2e saved), URLs to sustainability reports (2-3 min)

Step 2: Document Analysis (Switch to Llama 4 Maverick)

Upload 1-2 research papers about circular economy in food systems or bread waste valorization, then ask:

Based on these papers, what are the theoretical CO2 savings from replacing malted barley with surplus bread in brewing? What percentage of barley can bread realistically replace? Provide detailed citations.

Expected: Deep analysis with accurate quotes about 25-30% replacement rate, land/water savings (3-4 min)

Step 3: Citation Verification (Switch to Mistral Small)

Review the previous response. Provide page-specific citations for each major claim about bread replacement percentages and environmental benefits. Cross-reference with the Toast Ale data from our web search.

Expected: Precise citations, confirmation that real-world data (Toast Ale 25% replacement) matches academic predictions (2-3 min)

Step 4: Data Analysis (Switch to DeepSeek Chat V3)

Generate or upload sample data, then ask:

Generate realistic data comparing traditional brewing vs bread-supplemented brewing for 20 breweries over 2 years:

- Water usage (L per L beer)
- CO2 emissions (kg per hL)
- Land use (m² per hL)
- Cost savings (% reduction)

Perform statistical analysis: Is the difference significant?
Create professional visualizations showing the circular economy impact.

Expected: Clean code, professional charts, t-tests showing significant differences (3-4 min)

Step 5: Final Synthesis (Use Llama 4 Maverick for complex reasoning)

Based on:

1. Toast Ale's verified impact (web search: 2.9M slices, 5.3t CO2e saved)
2. Academic literature (uploaded papers: theoretical 25-30% replacement)
3. Statistical analysis (our data: significant reductions across metrics)

Write a 300-word executive summary for policymakers about scaling bread-to-beer circular economy models. Include:

- Environmental benefits (with specific numbers)
- Economic viability
- Policy recommendations
- Citations from all three sources

Expected: Coherent synthesis proving circular economy works, with verified data (3-4 min)

Total Time: 15-20 minutes for complete research cycle!

Compare this to: Using ONE model for everything (would be slower and lower quality) or using the WRONG models for each task.

Success Criteria: - [] Each model performed better at its specialized task - [] Total output is publication-quality - [] All sources properly cited - [] Data analysis is rigorous - [] Synthesis is coherent

Key Insight: The power isn't just in the tools—it's in knowing WHICH tool to use WHEN!

6.5.2 The “Research Relay Race”

For group workshops:

1. **Team 1:** Uploads papers, asks RAG question (choose best model!)
2. **Team 2:** Takes that answer, does web search to expand (choose best model!)
3. **Team 3:** Takes both, analyzes data to validate (choose best model!)
4. **Team 4:** Synthesizes everything (choose best model!)

First team to create a coherent 200-word research summary with the FASTEST time and BEST quality wins.

Twist: Teams are judged on model selection strategy!

Why this is fun: You see how each capability builds on the last, AND you learn optimal model selection.

6.6 Key Learnings

6.6.1 What You Now Know

1. **Integration is the superpower** - Individual tools are good; combined, they're transformative
2. **Model selection matters** - Using the RIGHT model for each task dramatically improves results
3. **Conversation is the glue** - Context persistence makes integration possible
4. **Iteration is cheap** - Refine any part without redoing everything
5. **Reproducibility is automatic** - Share the conversation = share the entire methodology
6. **Free doesn't mean inferior** - Free OpenRouter models rival paid options when used correctly

6.6.2 The Meta-Skills: Prompt Engineering & Model Selection

You've been learning to: - Ask specific questions (Module 1) - Request citations (Module 2) - Enable/disable features strategically (Module 3) - Describe desired outputs (Module 4) - Chain complex requests (Module 5) - **Choose the right model for each task** (All modules)

This is prompt engineering + model selection. These are the literacy skills of the AI age.

Model Selection Summary: - **DeepSeek Chat V3** → Code, technical analysis, data science - **Gemini 2.5 Flash** → Quick questions, web search, speed - **Llama 4 Maverick** → Long documents, complex reasoning, synthesis - **Mistral Small** → Citations, academic accuracy, balanced tasks

6.7 Questions You Should Answer

1. Which capability should you use first? (*Usually RAG or Web Search to gather information, then Code to analyze*)
2. Which model should you use for coding tasks? (*DeepSeek Chat V3 - it's trained specifically on code*)
3. Which model is fastest for simple questions? (*Gemini 2.5 Flash - optimized for speed*)
4. Can you switch models mid-conversation? (*Yes! This is a key strategy for optimal results*)
5. Can you go back and change a previous step? (*Yes! Just ask AI to redo that part*)
6. How do you know if your synthesis is good? (*Check: Are all sources represented? Is the logic clear? Can someone reproduce this?*)
7. When should you NOT use all capabilities? (*When a simple question needs a simple answer*)

8. Are free models really as good as paid ones? *(Yes, when used correctly! DeepSeek rivals GPT-4 for code, Llama 4 rivals Claude for reasoning)*

6.8 The Anecdote

We ran this workshop with a group of sustainability researchers. One participant, Dr. Amara, was skeptical.

“I’ve been doing research for 15 years. I know my workflow.”

We gave her the final challenge. She chose to analyze plastic waste policy effectiveness.

15 minutes later, she had: - Synthesized 8 papers she’d uploaded - Found 6 current policy examples from 4 countries - Analyzed waste reduction data - Generated 3 graphs - Written a draft introduction with 14 citations

She looked up and said, “This would have taken me three days.”

Then she said something profound:

“But the real value isn’t the time saved. It’s that I can now explore three research directions in an afternoon instead of committing to one for a month. This changes how I think about research questions.”

That’s the real transformation.

6.9 The Quote

“The real voyage of discovery consists not in seeking new landscapes, but in having new eyes.” - Marcel Proust

“We live in a society absolutely dependent on science and technology and yet have cleverly arranged things so that almost no one understands science and technology. That’s a clear prescription for disaster.” - Carl Sagan

Today, you’ve gained new eyes. Use them.

7 Workshop Conclusion: Your Next Steps

7.1 What You've Accomplished Today

In 2.5 hours, you learned to:

Have AI conversations that remember context Upload documents and query them with citations Search the web without leaving your conversation Execute code and create visualizations instantly Integrate all four into a research workflow **Select the optimal FREE model for each task type**

More importantly: You've experienced a new way of doing research.

And you did it all with 100% FREE models!

7.1.1 Your Model Arsenal (Quick Reference)

Remember, you now have access to 4 specialized FREE models:

When you need to...	Use this model
Write Python code, analyze data	DeepSeek Chat V3
Get quick answers, search the web	Gemini 2.5 Flash
Analyze long documents, complex reasoning	Llama 4 Maverick
Get precise citations, balanced tasks	Mistral Small

Pro Strategy: Start with Gemini for exploration, switch to specialists when needed!

7.2 The 3-Day Challenge

Your mission: Use OpenWebUI for one research task in the next 3 days.

Why 3 days? Research on learning retention shows: - Use a skill within 24 hours → 65% retention - Use within 3 days → 50% retention - Wait a week → 20% retention

Suggested tasks: 1. Literature review for a section you're writing 2. Analyze a dataset that's been sitting in your folder 3. Fact-check something with current web search 4. Draft a research summary from multiple sources

7.3 Common Pitfalls (And How to Avoid Them)

7.3.1 Pitfall 1: Treating AI Like Google

Problem: Vague questions get vague answers **Solution:** Be specific. “Summarize this paper” → “What does this paper say about [SPECIFIC CLAIM]?”

7.3.2 Pitfall 2: Not Verifying Citations

Problem: AI can misquote or misinterpret **Solution:** Spot-check citations. Click through to sources. Trust, but verify.

7.3.3 Pitfall 3: Asking It to Do Everything

Problem: One massive prompt trying to solve your entire research project **Solution:** Break it down. Iterate. Refine.

7.3.4 Pitfall 4: Forgetting It's a Tool, Not Magic

Problem: Expecting AI to understand unstated context **Solution:** Provide context. If YOU need background to answer the question, so does the AI.

7.3.5 Pitfall 5: Using the Wrong Model for the Task NEW!

Problem: Using Gemini Flash for complex coding, or DeepSeek for quick lookups (slower than needed) **Solution:** Match the model to the task: - **Code?** → DeepSeek Chat V3 - **Quick question?** → Gemini 2.5 Flash - **Long document?** → Llama 4 Maverick - **Need citations?** → Mistral Small

7.3.6 Pitfall 6: Staying with One Model the Whole Time NEW!

Problem: Missing out on specialized capabilities **Solution:** Switch models as your task changes! Start with Gemini for exploration, switch to DeepSeek when you need code, switch to Mistral for citations. It takes 2 seconds!

7.4 Resources for Continued Learning

Within OpenWebUI: - Check the Admin Panel (if you're admin) for configuration options
- Explore model settings (temperature, etc.) for different behaviors - Try different models for different tasks

External Resources: - OpenWebUI Documentation: github.com/open-webui/open-webui - Prompt Engineering Guide: promptingguide.ai - Research Ethics & AI: [guidelines from your institution]

7.5 Final Reflection Questions

Take 5 minutes to write down:

1. What surprised you most today?
2. What's one workflow you'll change this week?
3. What's one thing you're still uncertain about?
4. Who else could benefit from learning this?

7.6 The Last Quote

"The illiterate of the 21st century will not be those who cannot read and write, but those who cannot learn, unlearn, and relearn." - Alvin Toffler

You just learned. Now go use it.

8 Appendix: Quick Reference Guide

8.1 Checklist for Each Module

8.1.1 Module 1: Basic Chat

- ☐ Create account
- ☐ Send first message
- ☐ Receive response < 10 seconds
- ☐ Start new chat
- ☐ Access chat history

8.1.2 Module 2: RAG

- ☐ Upload PDF
- ☐ Wait for processing
- ☐ Ask document-specific question
- ☐ Verify citation
- ☐ Test multi-document query

8.1.3 Module 3: Web Search

- ☐ Enable web search toggle
- ☐ Ask current-info question
- ☐ Receive response with URLs
- ☐ Compare with/without web search
- ☐ Verify source quality


8.1.4 Module 4: Code Execution

- ☐ Request calculation
- ☐ See code execute
- ☐ View result
- ☐ Generate visualization
- ☐ Interpret output

8.1.5 Module 5: Integration

- ☐ Use RAG + Web + Code in sequence
- ☐ Request synthesis
- ☐ Receive coherent output
- ☐ Verify all sources included

8.2 Troubleshooting Guide

Problem	Check	Solution
Can't log in	URL correct?	Try https://team1-openwebui.valuechainhackers.xyz
No response	Model selected?	Check dropdown menu at top
Document won't upload	File size?	Try smaller PDF first
Web search not working	Toggle enabled?	Look for 

Problem	Check	Solution
Code won't execute	Error message?	Share error with AI, it often fixes itself
Everything is slow	Multiple people testing?	Be patient, or try off-peak hours

8.3 Model Selection Cheat Sheet

Print this out and keep it handy!

8.3.1 Quick Decision Tree

Need to write code or analyze data?

→ Use DeepSeek Chat V3

Quick question or web search?

→ Use Gemini 2.5 Flash

Long document or complex reasoning?

→ Use Llama 4 Maverick

Need precise citations?

→ Use Mistral Small

Not sure? Start with Gemini, switch later!

8.3.2 Detailed Model Comparison

Feature	DeepSeek V3	Gemini Flash	Llama 4	Mistral Small
Speed	Fast	Very Fast	Good	Fast
Code Quality				
Reasoning				
Citations				
Context Size	128K	1M	256K	96K
Best For	Python/Code	Speed/Web	Long docs	Accuracy

8.4 Command Cheat Sheet

8.4.1 Basic Prompts

```
# Good question structure
"Based on [SOURCE], what does [SUBJECT] indicate about [TOPIC]?"

# Citation request
"Provide specific quotes and page numbers."

# Refinement
"Make that more concise / more detailed / more technical."

# Model switching (during conversation)
"Let me switch to DeepSeek for the coding part..."
[Change model in dropdown]
"Now write the Python code for..."
```

8.4.2 RAG Prompts

```
"Summarize the main argument of the uploaded document."
"What do these papers agree/disagree on regarding [TOPIC]?"
"Find every mention of [TERM] in the documents."
```

8.4.3 Web Search Prompts

```
"Search for recent developments in [TOPIC] since [DATE]."
```

```
"What are current examples of [CONCEPT] being implemented?"
```

```
"Fact-check this claim: [CLAIM]"
```

8.4.4 Code Execution Prompts

```
"Calculate [FORMULA] where [VARIABLES]."
```

```
"Create a [PLOT TYPE] showing [RELATIONSHIP]."
```

```
"Analyze this data and tell me [WHAT TO LOOK FOR]."
```

8.4.5 Integration Prompts

```
"Based on the uploaded papers, web search for current examples,
and analyze this data, what can we conclude about [QUESTION]?"
```

8.5 Keyboard Shortcuts

(These depend on OpenWebUI version, but commonly:)

- **Ctrl/Cmd + Enter:** Send message
- **Ctrl/Cmd + K:** New chat
- **Ctrl/Cmd + / :** Toggle sidebar
- **↑ Arrow:** Edit last message

8.6 Getting Help

During the workshop: - Raise your hand - Ask your neighbor - Check this guide

After the workshop: - OpenWebUI GitHub Issues - Your institution's support - AI Stack Exchange

9 Acknowledgments

This workshop was designed using principles from:

- **Luma Institute** - Visual thinking and human-centered design
- **Hyper Island** - Experiential learning and reflection
- **Greg Wilson (Teaching Tech Together)** - Evidence-based teaching methods
- **Julie Dirksen (Design for How People Learn)** - Learning science
- **AJ&Smart** - Workshop facilitation and engagement
- **The Art of Hosting** - Community learning practices

Special thanks to: - The Open-WebUI community - Researchers who piloted this workshop
- You, for being here

10 About This Workshop

Version: 1.0 **Last Updated:** 2025 **License:** CC BY-SA 4.0 (use it, adapt it, share it)

Feedback: [your contact method]

If this workshop helped you: Pay it forward. Teach someone else.

“The best way to learn is to teach. The best way to understand is to explain. The best way to master is to share.” - Workshop Philosophy

Now go do research differently.