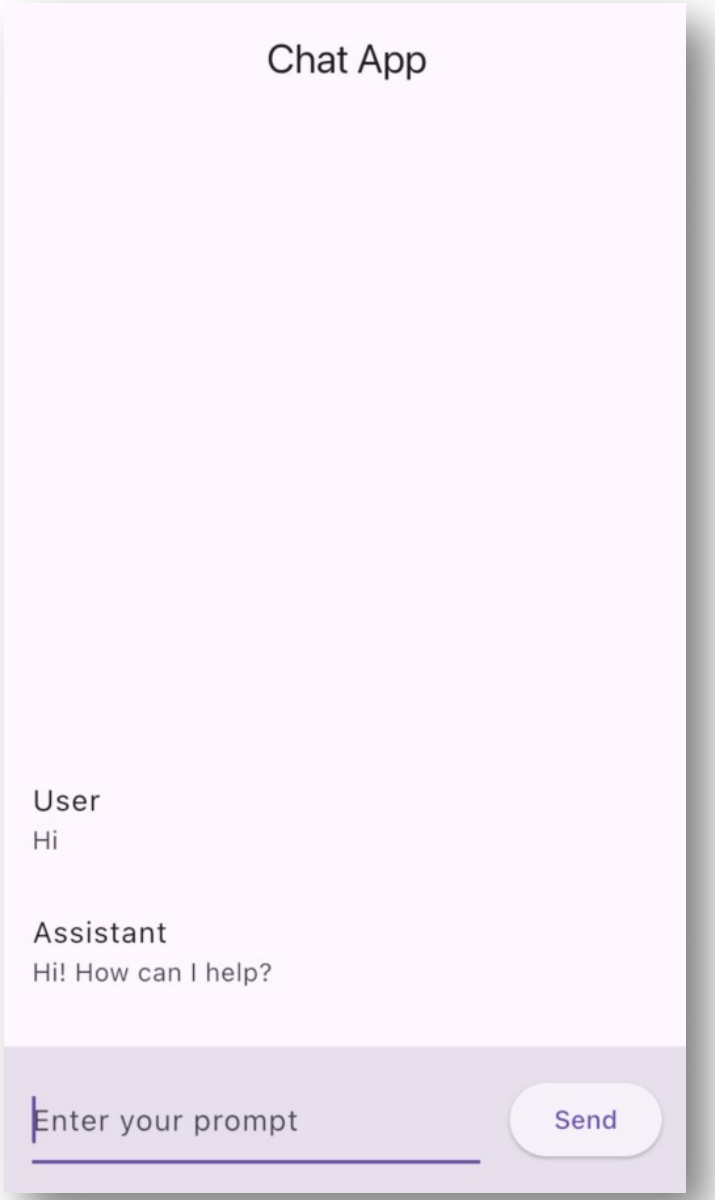
AI & Agents

Shan-Hung Wu
CS, NTHU

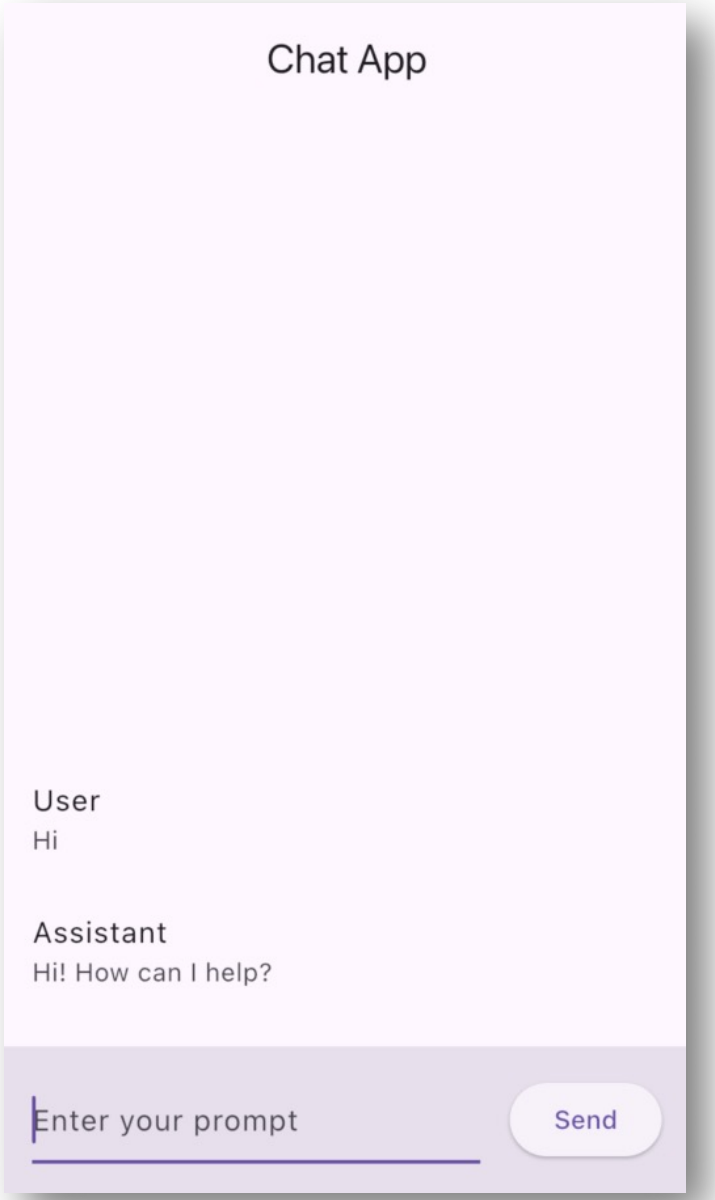
Today's Topics

- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



Today's Topics

- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



Artificial Intelligence

- AI: systems that perform tasks usually associated with human intelligence
  - Image recognition
  - Spam detection
  - Voice assistants
  - Route planning
  - …

Machine Learning

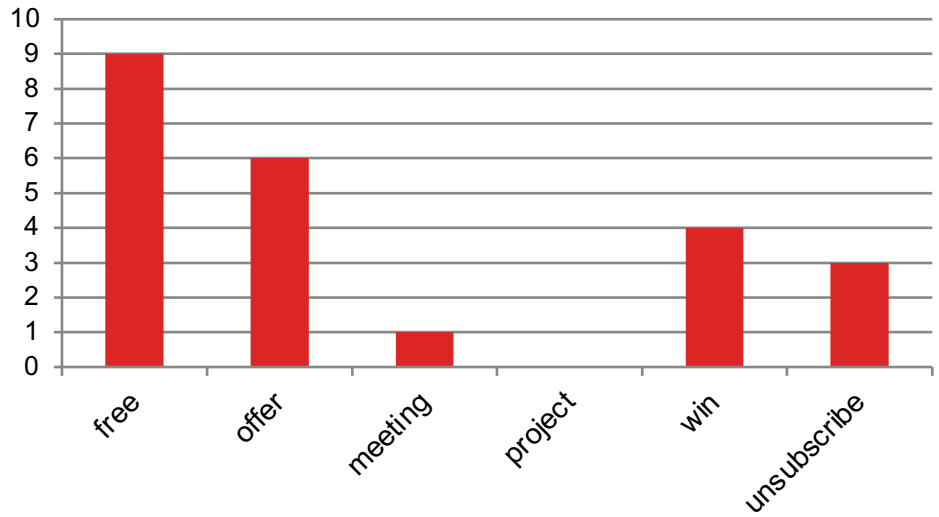
- A branch of AI that acquires intelligence by learning a ***model*** from ***data***
- In particular, ***supervised learning*** aims to get a ***prediction function***, e.g.

$$f(x;\, w) \;=\; w_1 x_1 \;+\; w_2 x_2 \;+\; w_0 \;=\; y$$

  - *x* = input vectors (features)
  - *y* = output / prediction
  - *w* = ***weights*** trained by data
- Training data has many sample pairs (*x*,*y*)
  - *y* in a sample pair is called ***supervision***

Features in *x*

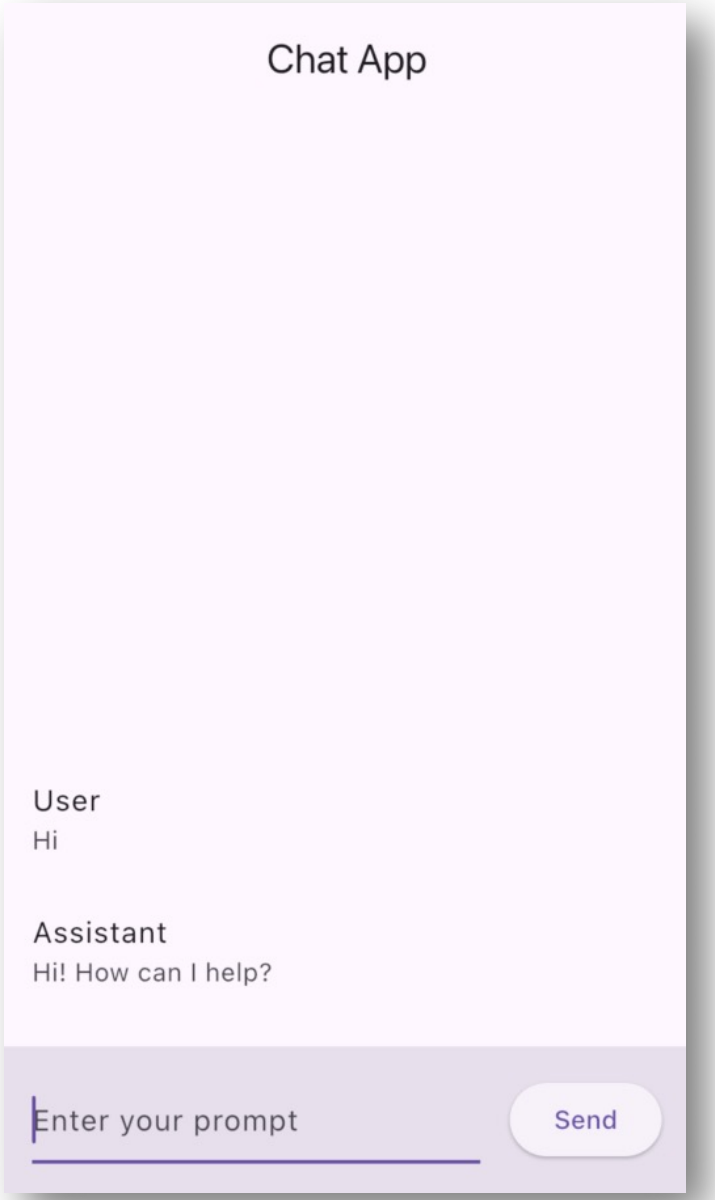
- A ***feature*** is one measurable property of the input
- Example from Spam Detection:
  - *x* could count how often certain words appear in one email



  - x = [9, 6, 1, 0, 4, 3, ...] with dimension = dictionary size
- Classical ML requires humans to design features

Today's Topics

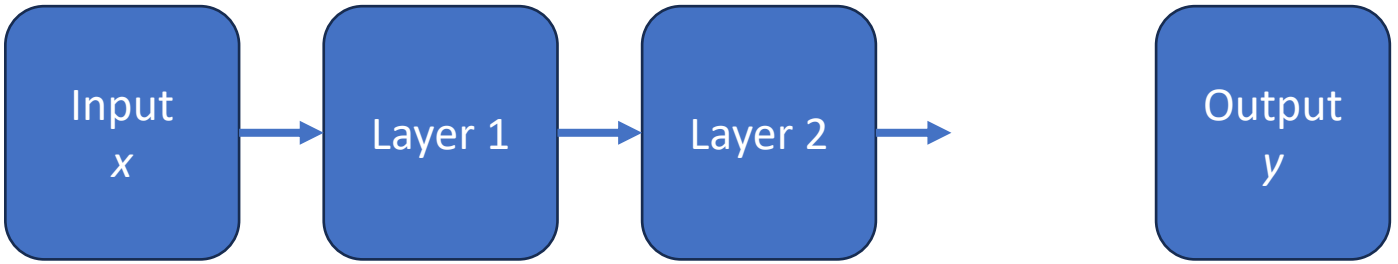
- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



Deep Learning

- Leans a prediction function with many layers:

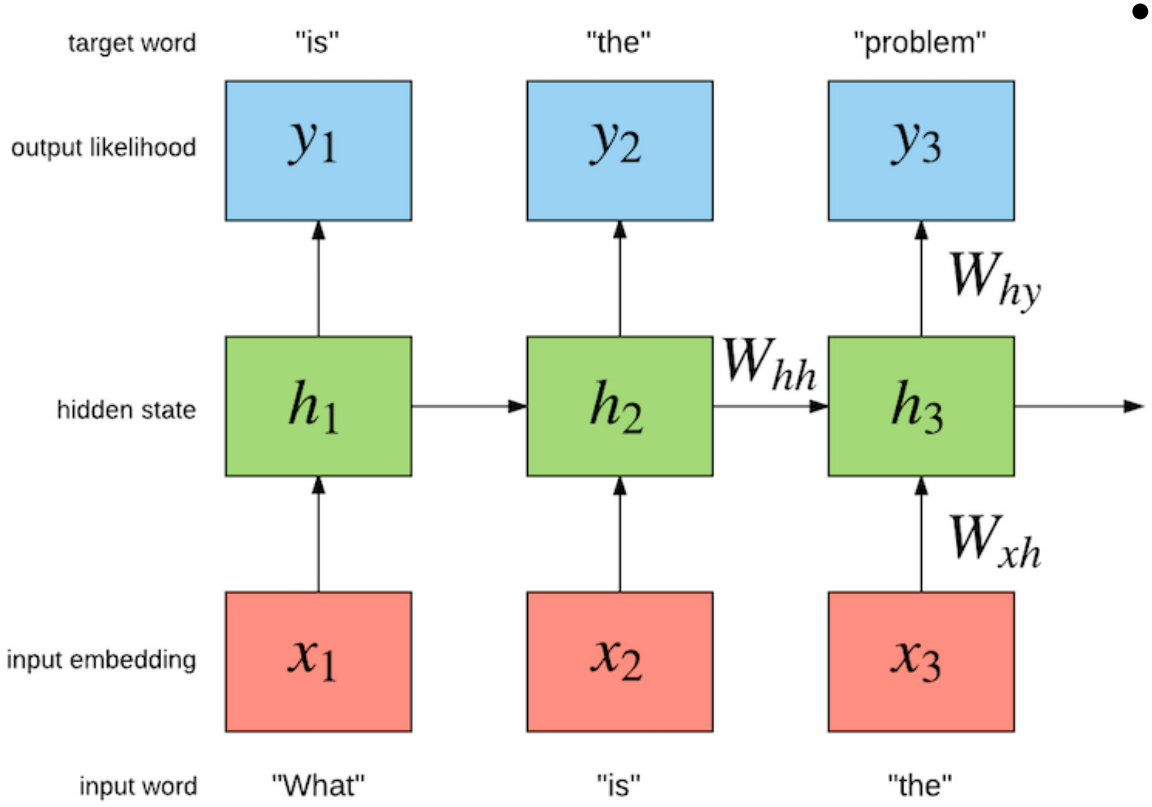
$$f_L(\dots f_2(f_1(x; w_1); w_2) \dots; w_L) = y$$



- Why?
  - ***Automatic*** feature extraction (***representation learning***)
- Good for images, audio, and language
- But training can be data- and compute-intensive

Generative AI & LLMs

- Gen-AI: a model that can output ***non-exhaustive y***
  - E.g., image/video generators, LLMs
- (Large) Language model:



- Trained by large, ***unsupervised*** data $\{x\}$ without sample y

From LLMs to Chat Bots

- By default, an LLM outputs:

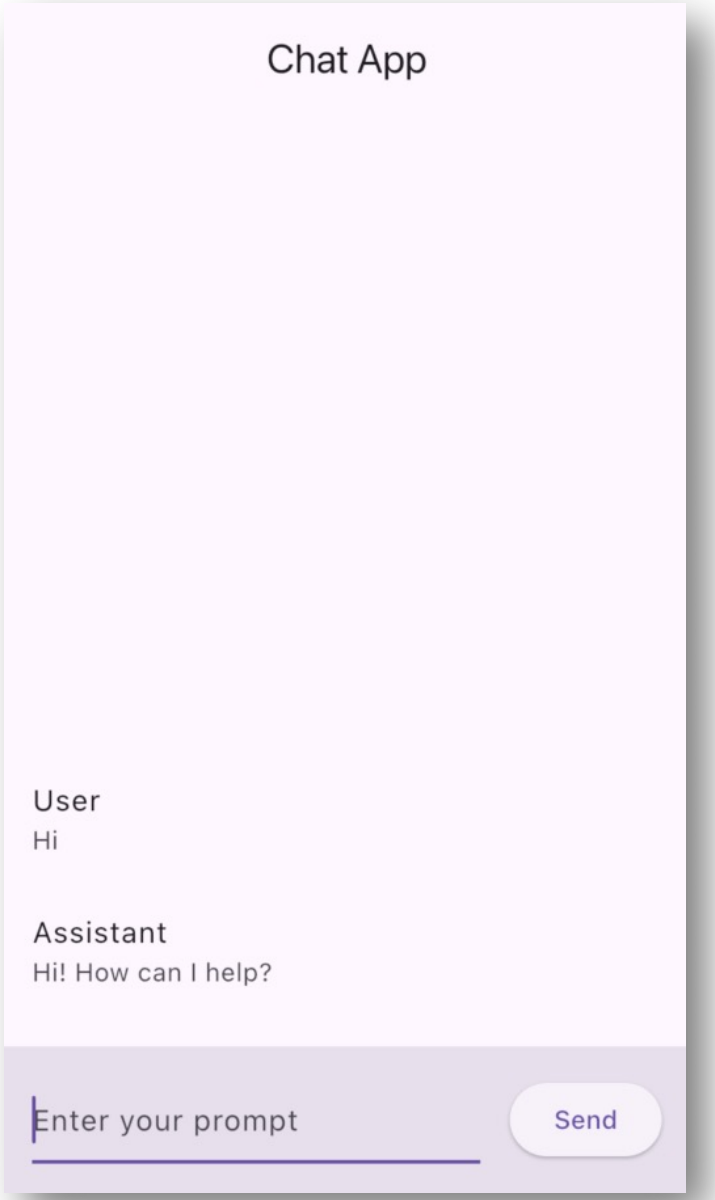
$$P(token_t \mid token_1, token_2, \ldots, token_{t-1})$$

  - Not ideal for chat and problem solving

- ***Alignment*** process letting LLM learn from:
  1. Large text corpus (unsupervised)
  2. Few, high-quality sample (*x*, *y*) sequences for useful tasks
  3. Feedback signals from trials-and-errors
     - Another evaluator model outputting “good” or “bad”

Today's Topics

- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



Demo: Chat App

- `ChatService` emits messages via a `StreamController`
- `Home` listens to the stream and build UI using `StreamBuilder`
- What you sent is not just:

```
userPrompt1 → response1
userPrompt2 → response2
```

- Instead:

```
(userPrompt1) → response1
(userPrompt1,response1,userPrompt2) → response2
```

Context

- The chat history:

  `(`**`context`**`,userPromptN-1) → responseN`

- Usually contains
  - Developer prompts: developer-defined, with role `developer`
    - The very first is called ***system prompt***
  - User prompts: user-defined, with role `user`
  - LLM responses: LLM-defined, with role `assistant`
- Has limited size
  - 1M tokens as of 2026
  - Longer doesn't always give better results

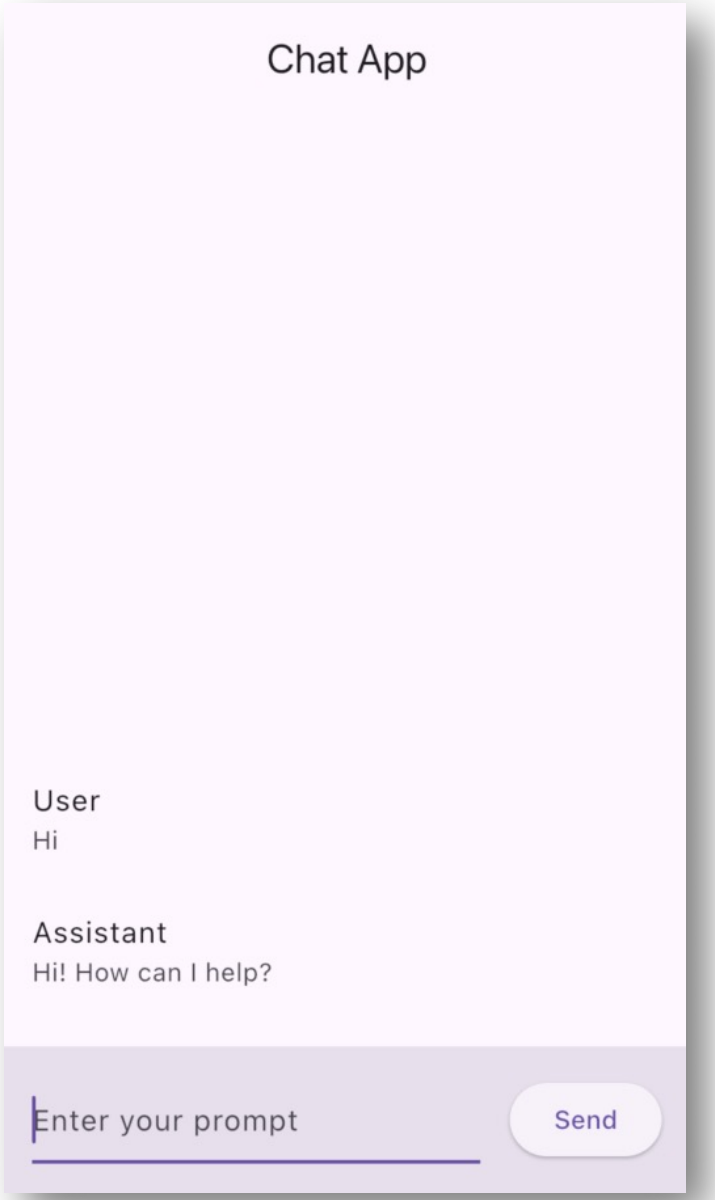
Context Engineering

- As a developer, you have full control of context to be issued to LLM
  - Dynamically modify system/user prompts or responses in the history

- User: “Help me pass this course”
- What would you do as an app developer?

Today's Topics

- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



Flow vs. Agent

Flow:

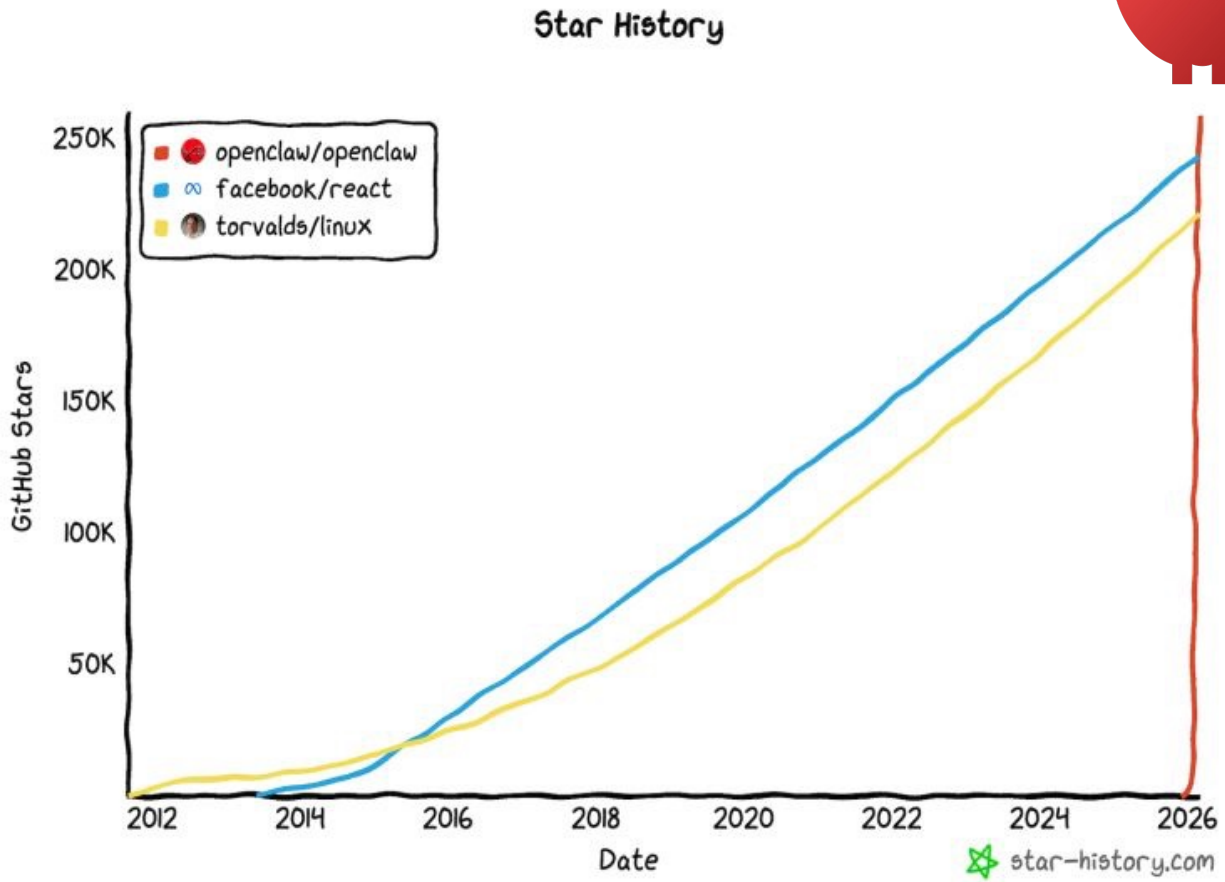
1. Notify user to start learning session at 7AM
2. If user doesn't press "Start," send motivating message
3. Else, answer user's questions
4. On end, show "Ready for quiz?"
5. …

Agentic:

- Check lecture notes
- Check user's calendar
- Create learning plan
- Notify user at each event
- Change plan based on user activities
- Read new lecture notes
- …

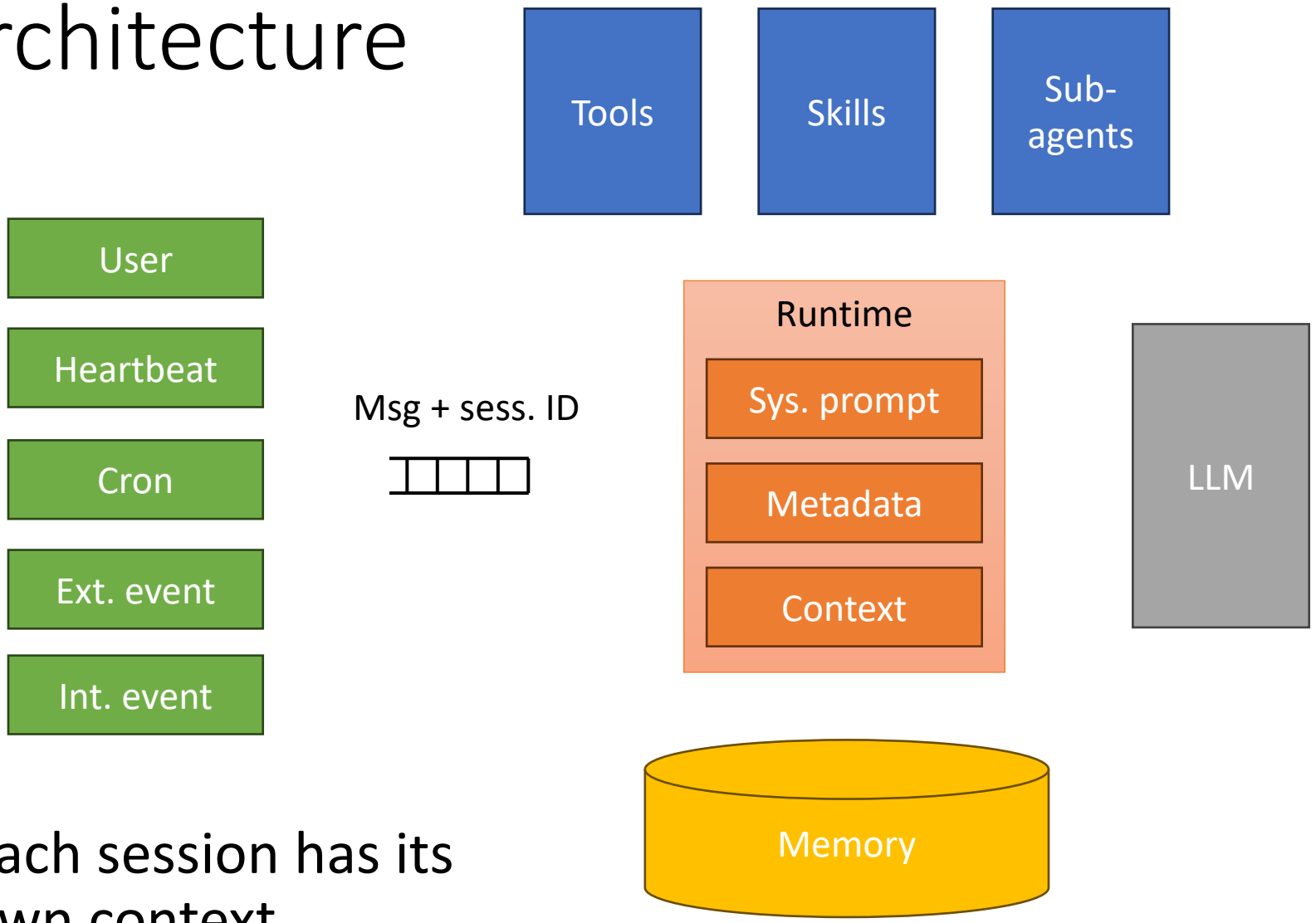
Steps decided by AI, not humans

Case Study: OpenClaw



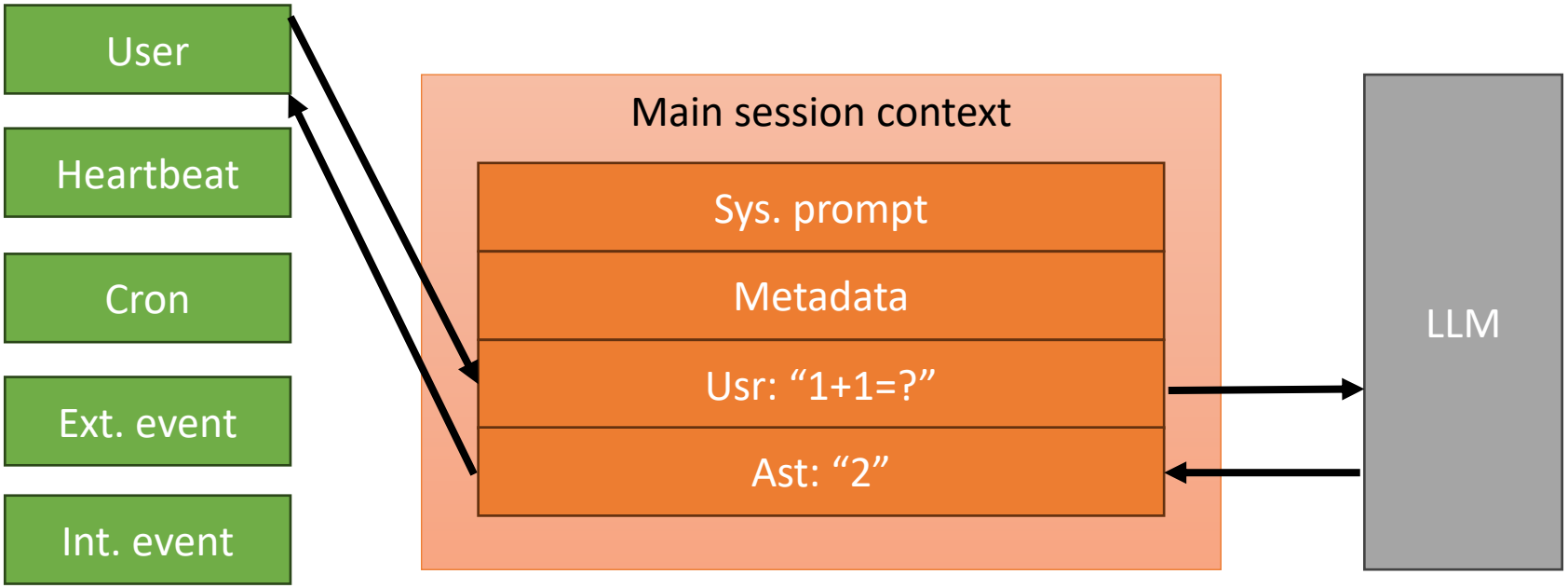
- An agentic context engineering framework

Architecture

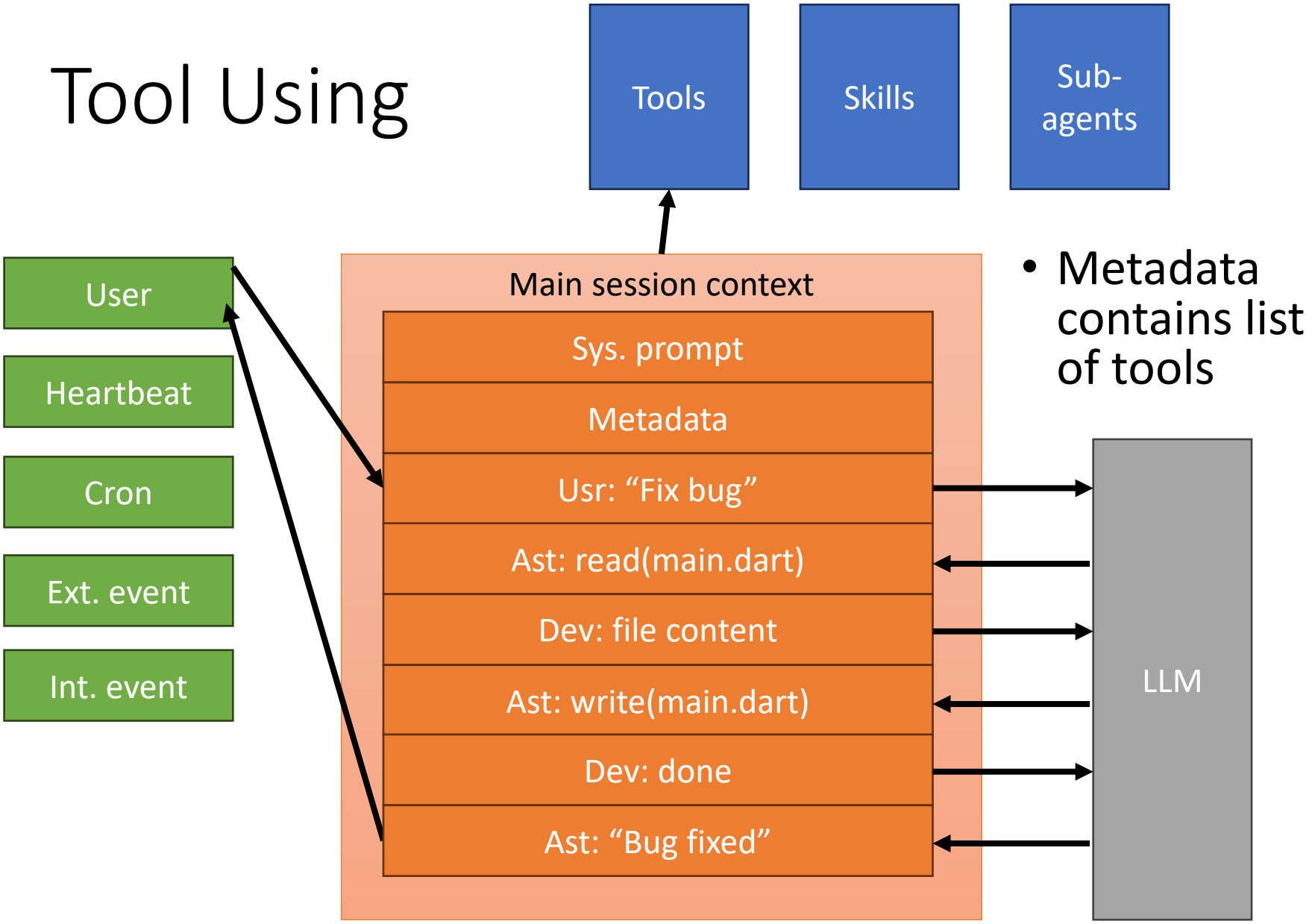


- Each session has its own context

Simple QA

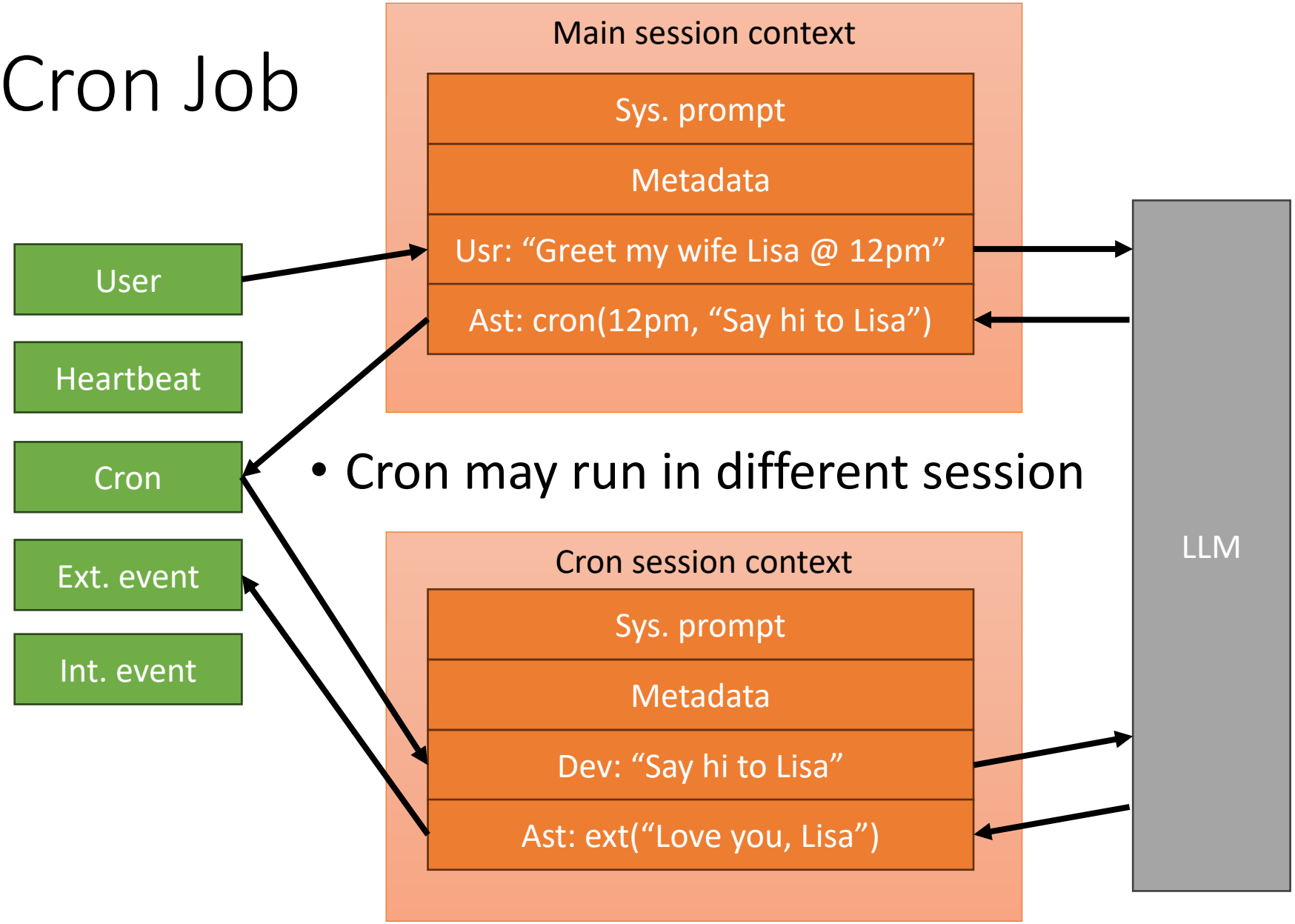

User
Heartbeat
Cron
Ext. event
Int. event
Main session context
Sys. prompt
Metadata
Usr: “1+1=?”
Ast: “2”
LLM

Tool Using

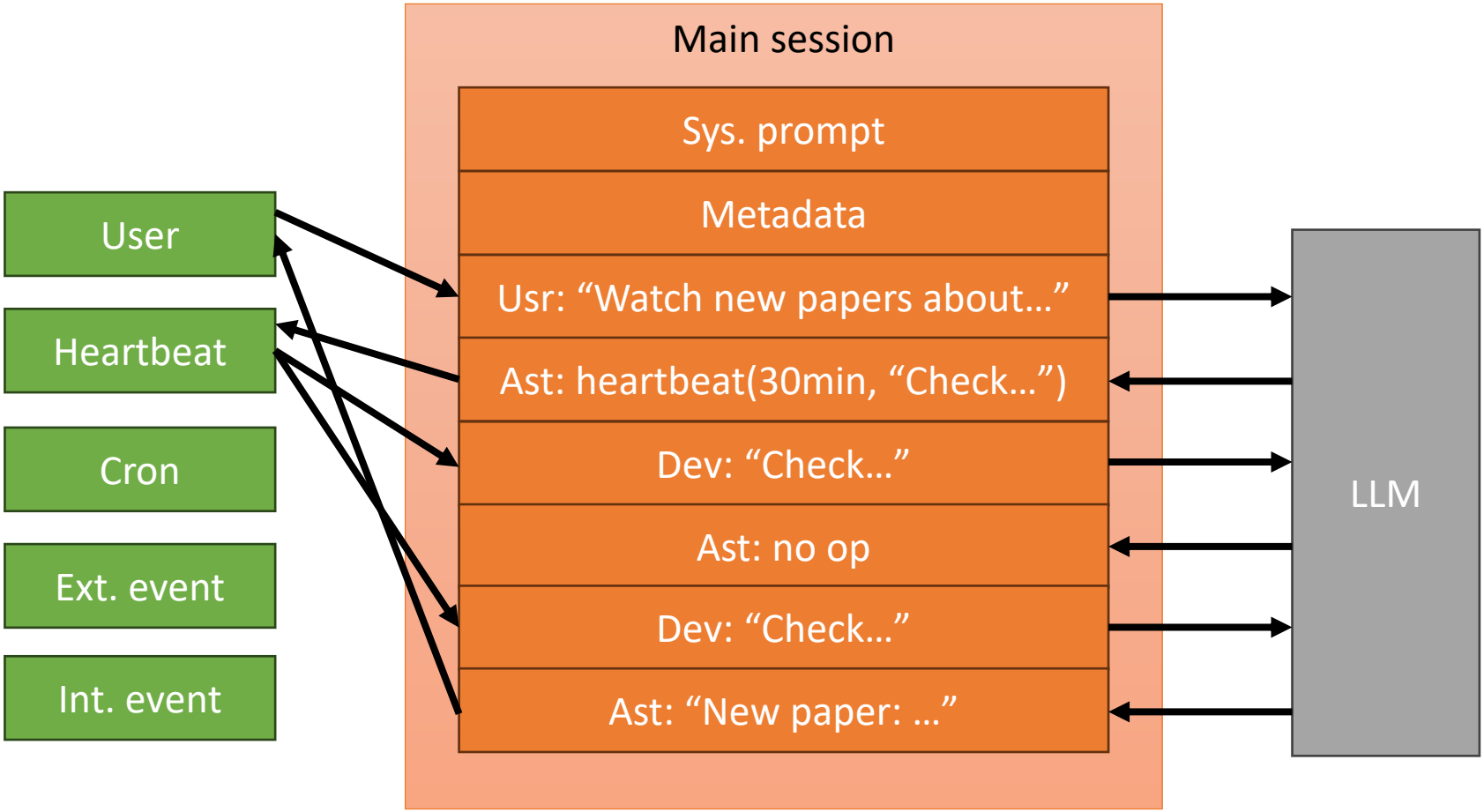


- Metadata contains list of tools

Cron Job

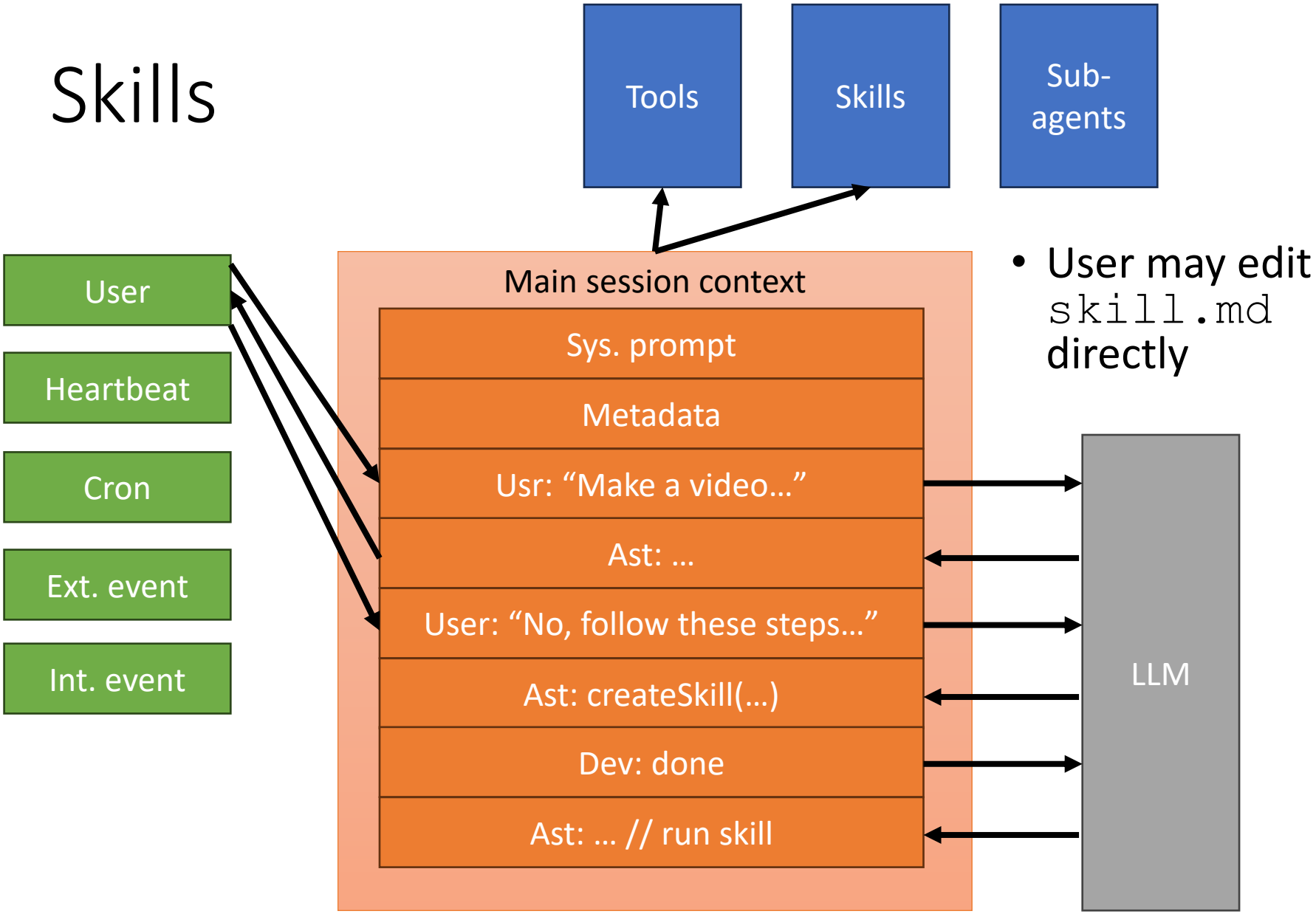

Main session context
Sys. prompt
Metadata
Usr: “Greet my wife Lisa @ 12pm”
Ast: cron(12pm, “Say hi to Lisa”)
User
Heartbeat
Cron
Ext. event
Int. event
• Cron may run in different session
Cron session context
Sys. prompt
Metadata
Dev: “Say hi to Lisa”
Ast: ext(“Love you, Lisa”)
LLM

Heartbeat



- Usually runs in main session

Skills



- User may edit `skill.md` directly

Skills

- A folder containing:
- SKILL.md
  - Step-by-step procedure for the task
  - Trigger conditions (e.g., “Use this when the user asks to generate a video”)
  - Nuances like “Check `.env` file before creating a key”
  - Instructions for running ***scripts***
  - Success criteria (e.g., How to know the task is actually finished)
- Scripts (code)

- Can be easily shared on, e.g., ClawHub

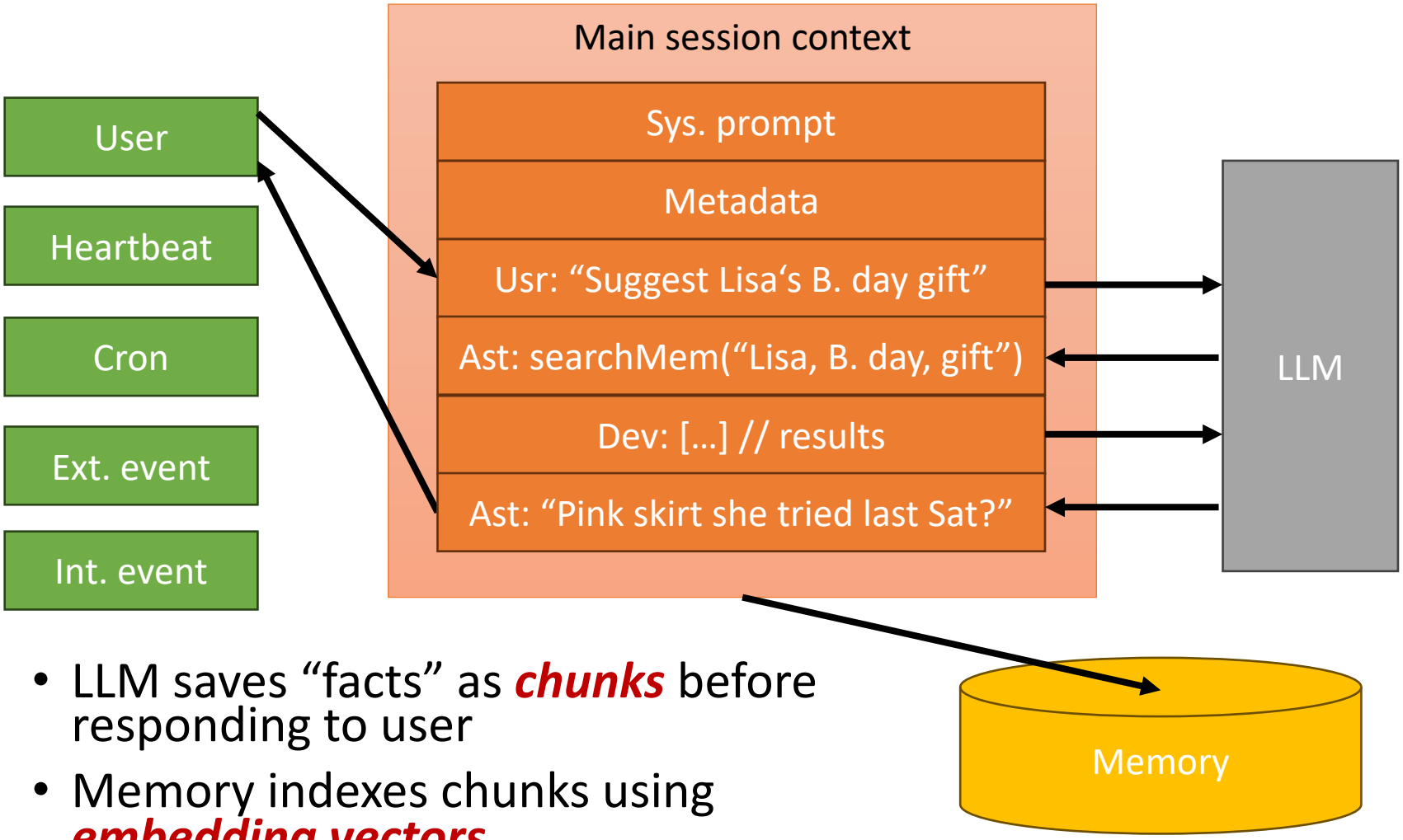
Context Compression & Memory

- Agent's context grows fast
- But context size is limited!

- Solutions:

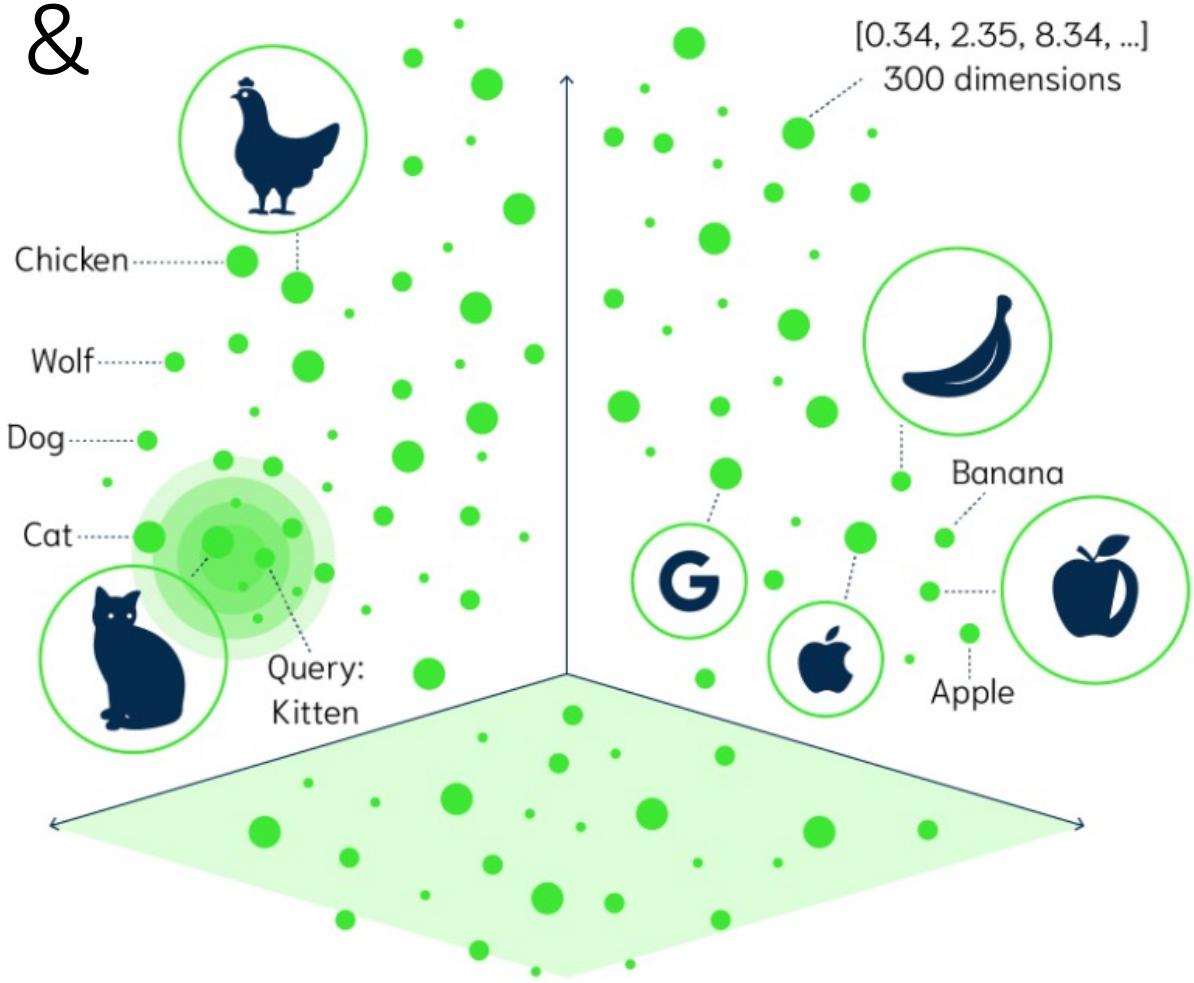
1. Context compression
   - When context length pass certain threshold, ask LLM to summarize context, then use summarization
   - Remove tool using results (entirely or partially)
2. External memory & RAG
3. Subagents

Retrieval Augmented Generation (RAG)



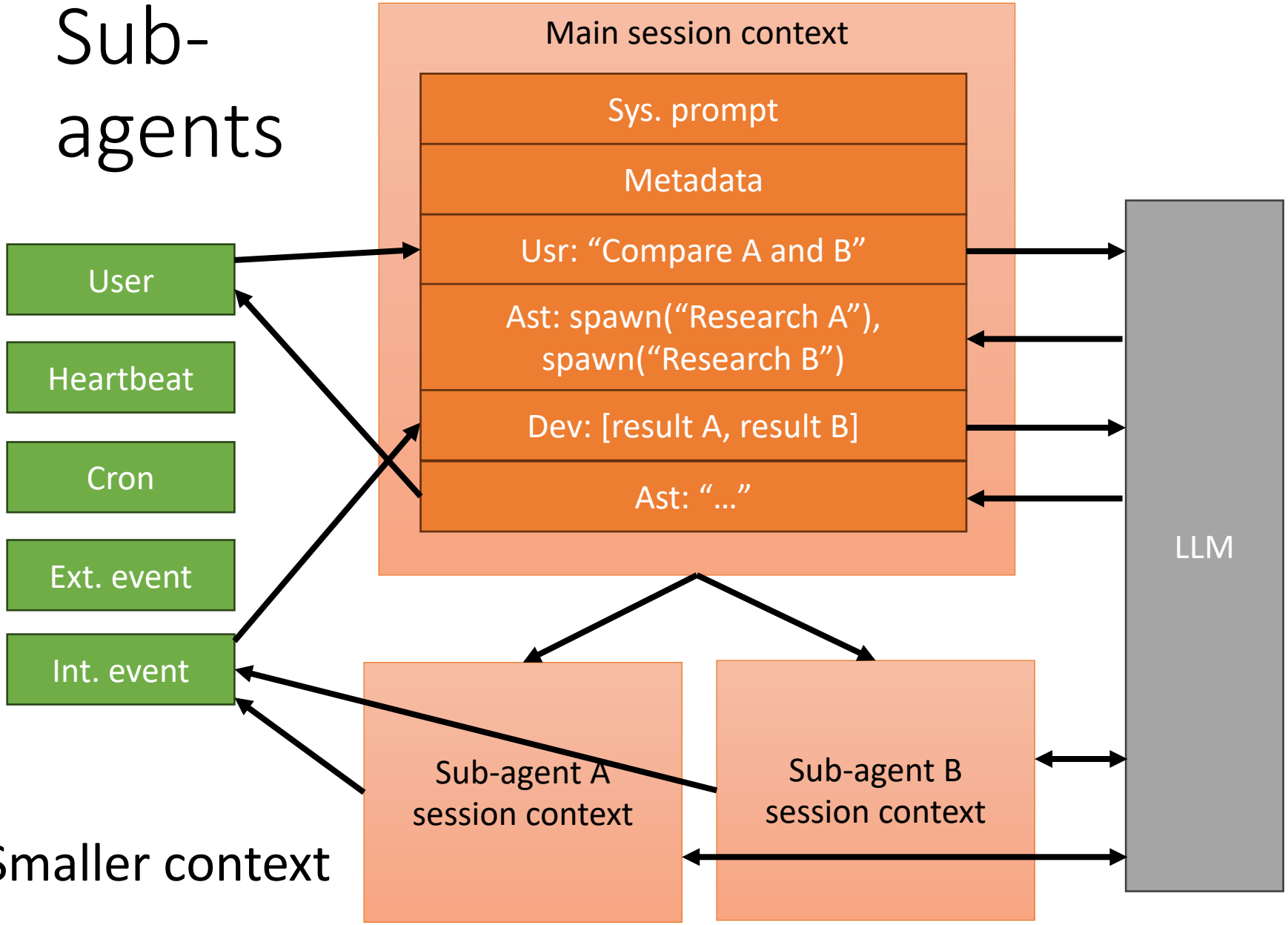
- LLM saves “facts” as ***chunks*** before responding to user
- Memory indexes chunks using ***embedding vectors***

Embeddings & Searching



1. Embed query
2. KNN search by cosine similarity

Sub-agents



Smaller context

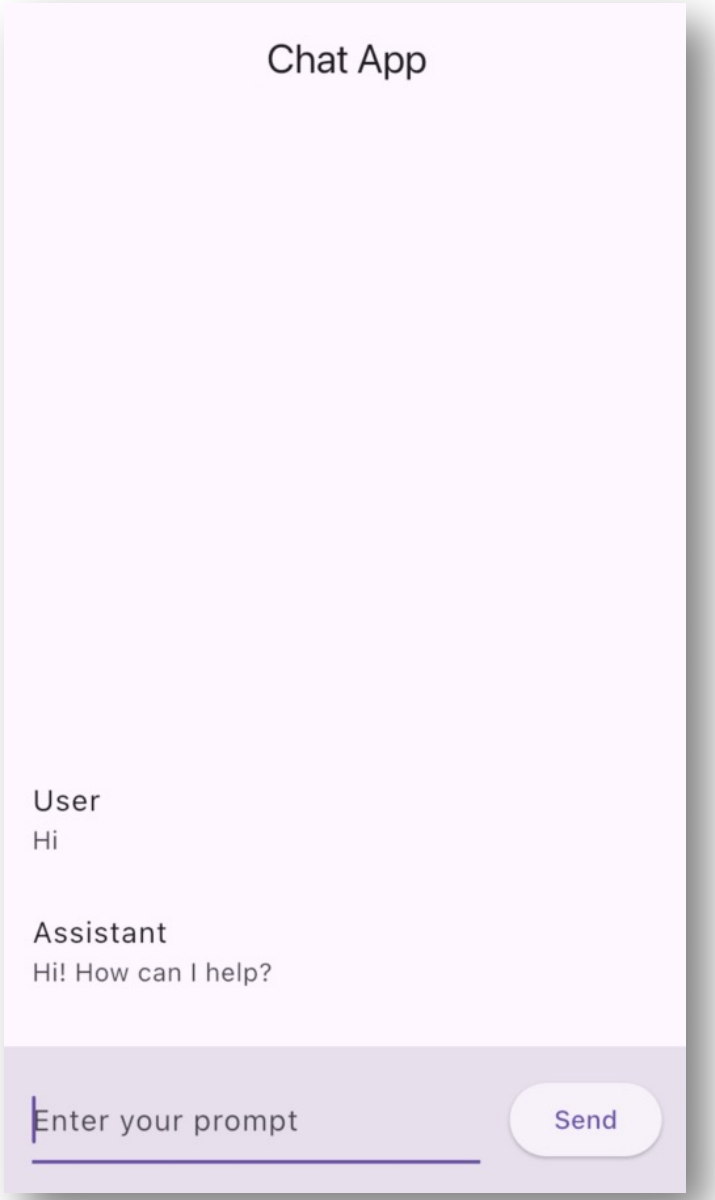
What Makes OpenClaw Special?

- System prompt that enable all above abilities
  - Read the source!
- Additionally, ***updatable*** metadata:
  - SOUL.md
  - IDENTITY.md
  - USER.md
  - MEMORY.md (not a chunk in memory)
  - Tool list, skill list, etc.

- Together, these make the agent feels alive

Today's Topics

- AI & Machine Learning
- Deep Learning & LLMs
- Demo: Chat Bot & Context
- Agents & Context Engineering
- Your Term Project



About Your Term Project

- Design and implement an ***agent app*** that solves a ***real*** problem
  - Framework: Flutter (frontend) + Firebase (backend)

- Your next demo:

1. Problem to solve
2. Agentic flow design
   - Tools, skills, memory strategy, channels, etc.
   - System prompt that glue all together
3. Usability test & lesson learned

Minimum Novelty

- Must be beneficial from user's perspective
- Must clearly beat non-agentic flows
  - Pick a task where agent shine
- Must go through at least one (usability test → improvement) cycle

References

- OpenAI Response API
  - Conversation state management
- OpenClaw system prompt study note