

# Introducing Multimodal Llama 3.2


## Multimodal Prompting



# Llama 3.2 vision models




## Pretrained:

- Llama-3.2 1B (text only)
  - Llama-3.2 3B (text only)
- 
- Llama-3.2 11B-Vision (text+image)
  - Llama-3.2 90B-Vision (text+image)



## Instruction-tuned:

- Llama-3.2 1B-Instruct (text only)
  - Llama-3.2 3B-Instruct (text only)
- 
- Llama-3.2 11B-Vision-Instruct (text+image)
  - Llama-3.2 90B-Vision-Instruct (text+image)

# Llama 3.2 text model

- Built on top of Llama 3.1 text-only models.
- The same tokenizer as Llama 3.1
- The same 128k context window.

## Text capabilities:

- Llama 3.2 **11B** ↔ Llama 3.1 **8B**
- Llama 3.2 **90B** ↔ Llama 3.1 **70B**

## Supported languages:

- The same as Llama 3.1.
- **For text only tasks:** English, German, French, Italian, Portuguese, Hindi, Spanish, and Thai.
- **For image+text applications:** Only English.

# Example

`<|begin_of_text|>`

`<|start_header_id|>`

user

`<|end_header_id|>`

`<|image|>`Describe this image in two sentences.

`<|eot_id|>`

`<|start_header_id|>`

assistant

`<|end_header_id|>`

# Use case



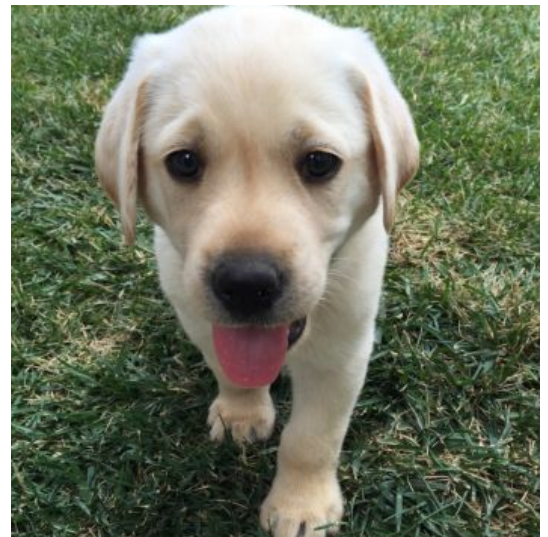
Counting the llamas



Plant recognition



Tire pressure warning



Dog breed recognition



# Use case



## Analyzing multiple receipt images

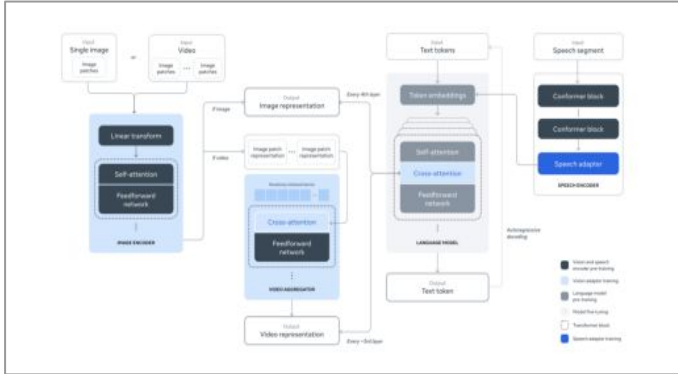


## Interior design assistant

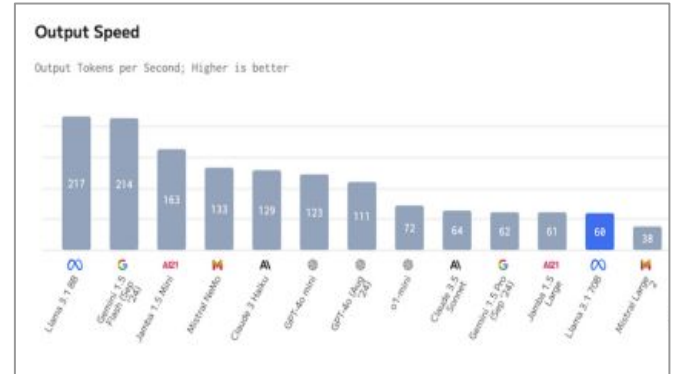


## Nutrition facts to JSON

# Use case



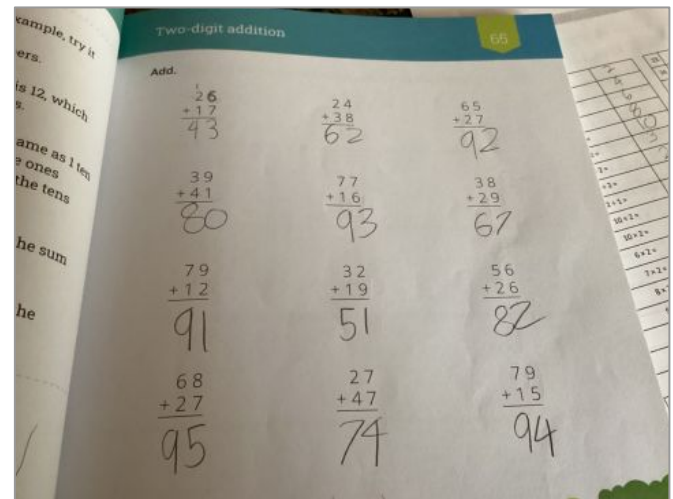
## Convert diagram to code



## Convert plot to table



## Analyze fridge content



# Grade math homework

# Use case: image to tool call



Where is this place?

What is the current weather at this place

```
<|python_tag|>  
brave_search.call(  
  query="current weather in San Francisco")
```



# Introducing Multimodal Llama 3.2

## Prompt Format



# The Llama 3.1 & 3.2 supported roles

**system** – Sets the context in which to interact with the AI model. It typically includes rules, guidelines, or necessary information that helps the model respond effectively.

**user** – Represents the human interacting with the model. It includes the inputs, commands, and questions to the model.

**ipython** – A new role introduced in Llama 3.1. Semantically, this role means "tool". This role is used to mark messages with the output of a tool call when sent back to the model from the executor.

**assistant** – Represents the response generated by the AI model based on the context provided in the 'system', 'ipython' and 'user' prompts.

Each role is set between the special tokens

**<|start\_header\_id|** and **<|end\_header\_id|**.

```
"<|begin_of_text|>",  
"<|end_of_text|>",  
"<|reserved_special_token_0|>",  
"<|reserved_special_token_1|>",  
"<|finetune_right_pad_id|>",  
"<|step_id|>",  
"<|start_header_id|>",  
"<|end_header_id|>",  
"<|eom_id|>",  
"<|eot_id|>",  
"<|python_tag|>",
```

# Special tokens for single-turn and multi-turn chat

1. **<|begin\_of\_text|>**: Start of a prompt.
2. **<|start\_header\_id|>**: Start of a role for a particular message. Possible roles are: system, user, assistant and ipython.
3. **<|end\_header\_id|>**: End of the role for a particular message.
4. **<|eot\_id|>**: End of a turn, which can be the end of the model's interaction with the user or a tool.

# Special tokens for tool calling

5. **<|eom\_id|>**: End of Message. A message represents a possible stopping point where the model can inform the execution environment that a tool call needs to be made.
6. **<|python\_tag|>**: A special tag used in the model's response to signify a tool call.



# Special tokens for fine-tuning and base model

7. **<|finetune\_right\_pad\_id|>**: Used for padding text sequences in a batch to the same length.
8. **<|end\_of\_text|>**: Model will cease to generate more tokens after this. This token is generated only by the base models.

# Tokenization



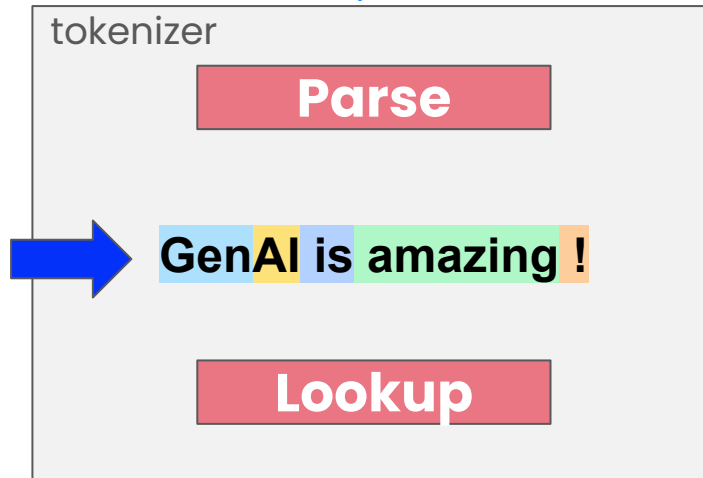
"How many  
'r's in  
Strawberry?"

tokenizer

[4438, 1690,  
3451, 81, 753,  
304, 73700,30]



"GenAI is amazing!"



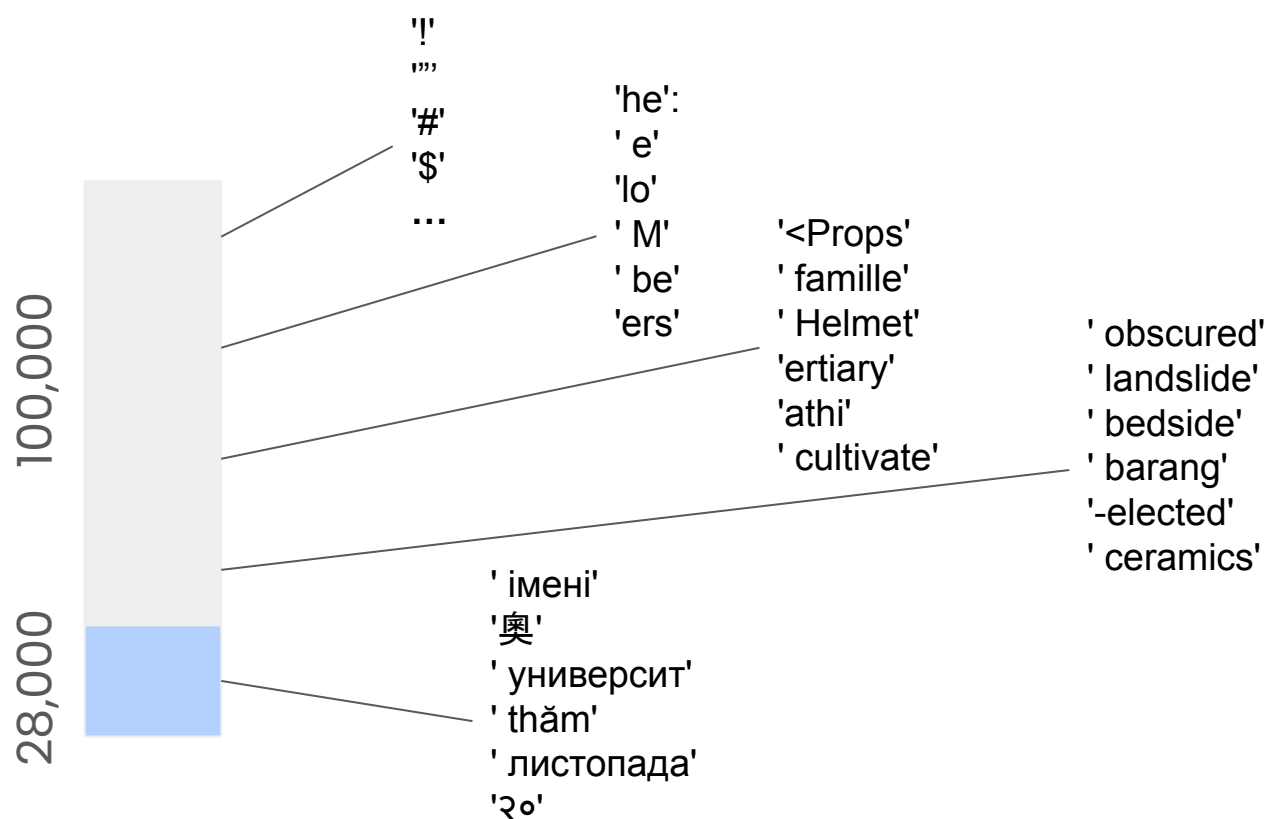
[10172, 15836, 374, 8056, 758]

# Vocabulary



128,000 Token vocabulary

30-50,000 word vocabulary



# Llama 3 family tokenization

The Llama 3 family uses the open source **tiktoken3, Byte Pair Encoding (BPE) tokenizer**.

The family has a **vocabulary of 128K tokens**. This is four times more than Llama 2. and, just for reference, most people use a vocabulary of 30-50 thousand words, so, it is big! Compared to the Llama 2 tokenizer, the new tokenizer improves compression rates on a sample of English data from roughly 3.2 to four characters per token so large prompts require fewer tokens which speeds up inference.

In the 128K tokens:

- 100K tokens are from the tiktoken3 tokenizer
- 28K additional tokens to better support non-English

```
tokenizer.encode("hello world")
```

```
[15339, 1917]
```

```
tokenizer.decode([1917])
```

```
world
```

# Steps below summarize how LLMs Process Input

**Text → Tokens:** The raw input text is split into tokens using a tokenizer.

**Tokens → Integer IDs:** Each token is mapped to an integer ID from the tokenizer model's vocabulary.

**Integer IDs → Embeddings:** These integer IDs are then converted into embeddings, which are the real inputs to LLMs.



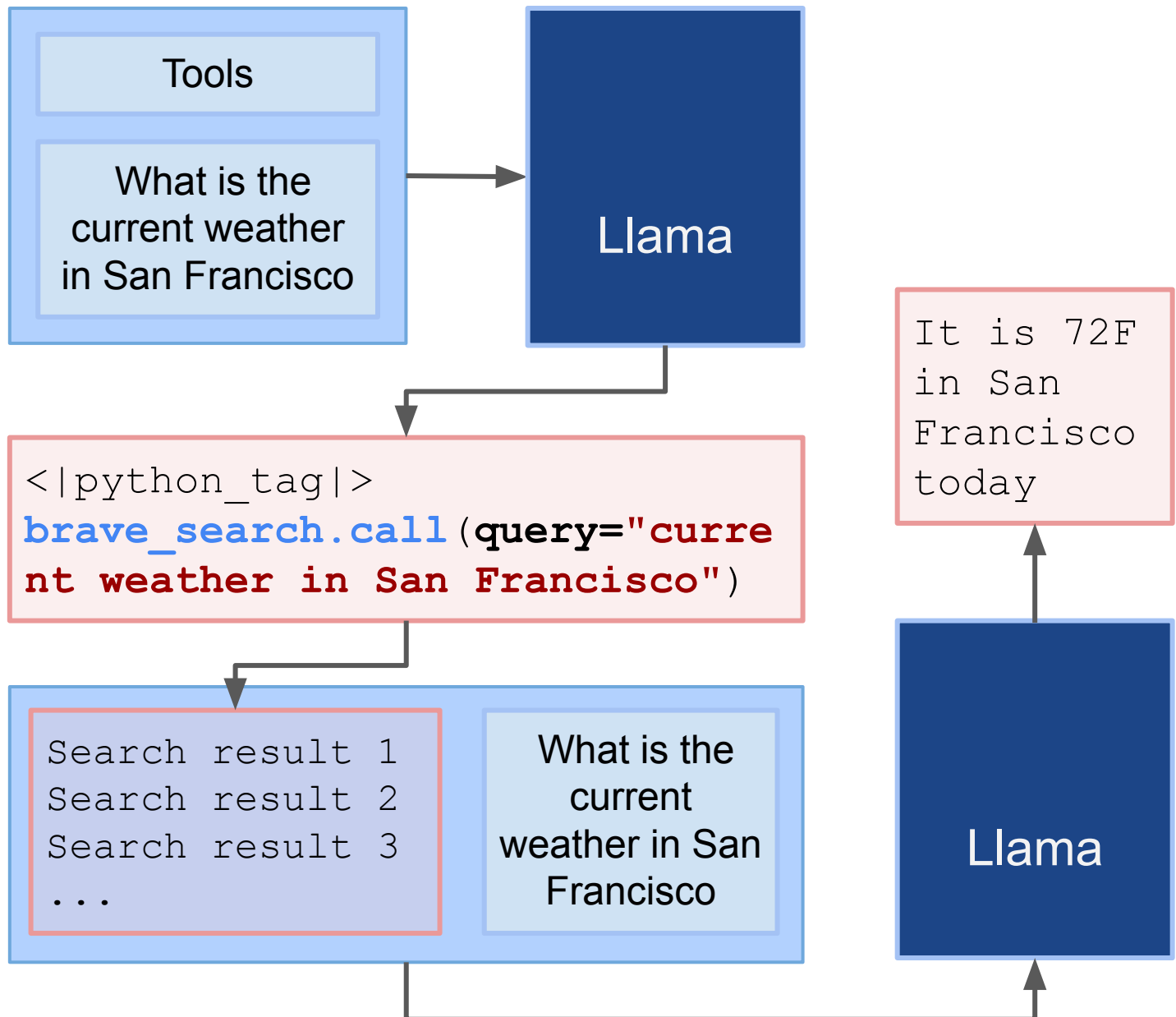
# Tool Calling

- Accessing real time info
- Performing complex math or code tasks
- Interacting with external data and systems
- Building dynamic agents

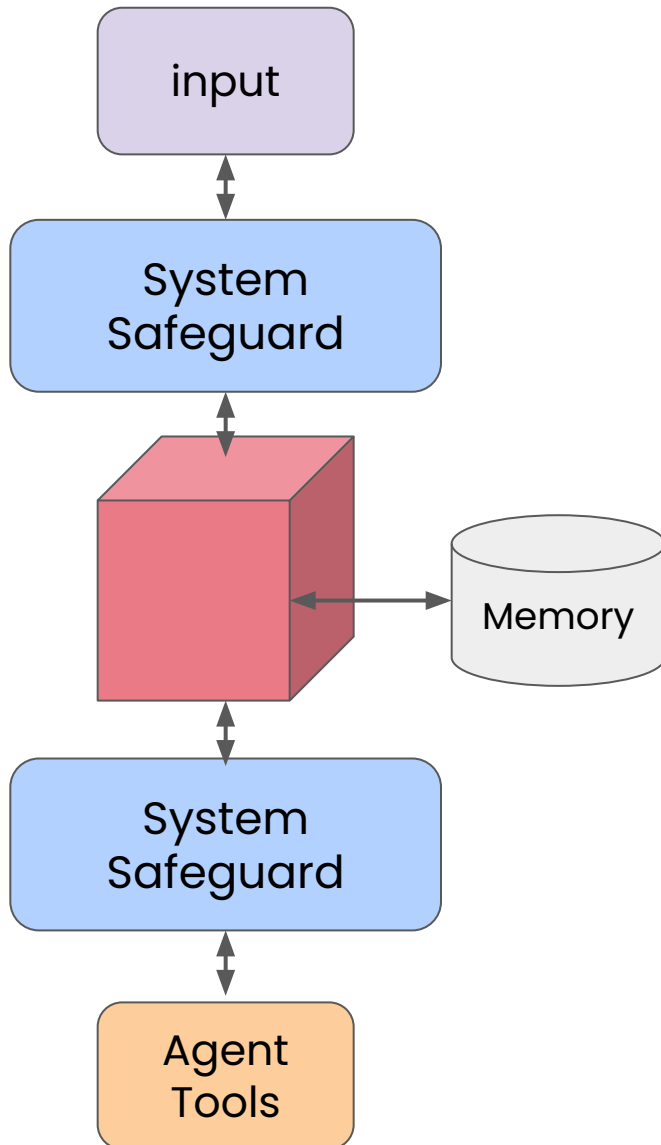
## In this lesson

- The Llama's special tokens and role for tool calling.
- Built-in tool calling: search, wolfram alpha and code interpreter.
- Custom tool calling

# Use case: image to tool call



# Models work in a system



## AI System

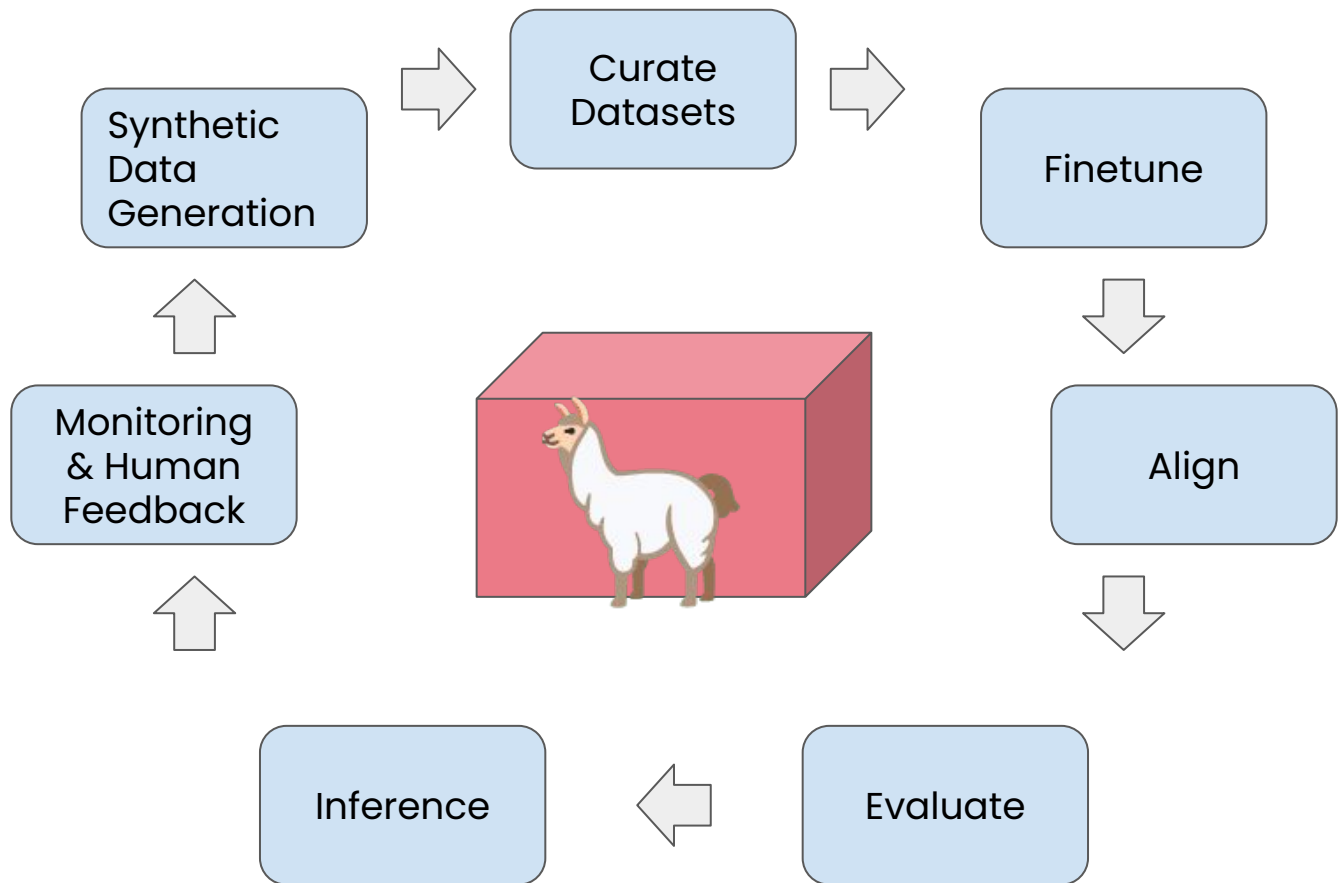
- Multilingual safety models,
- a prompt injection filter
- Cybersecurity Evaluation Suite

## Agentic systems require

- external tool use
- memory

LLM reference System

# Model Lifecycle



# Llama Stack APIs

**Agentic Apps**  
End applications

**Agentic System API**  
System component orchestration

PromptStore

Assistant

Shields

Memory

Orchestrator

**Model Toolchain API**  
Model development & production tools

Batch Inference

Realtime Inference

Quantized Inference

Continual Pretraining

Evals

Fine Tuning

Pretraining

Reward Scoring

Synthetic Data Gen

**Data**  
Pretraining, preference,  
post training

**Models**  
Core, safety, customized

**Hardware**  
GPUs, accelerators, storage



# Llama Stack APIs

**Agentic Apps**  
End applications

**Agentic System API**  
System component orchestration

PromptStore

Assistant

Shields

Memory

Orchestrator

**Model Toolchain API**  
Model development & production tools

Batch Inference

Realtime Inference

Quantized Inference

Continual Pretraining

Evals

Fine Tuning

Pretraining

Reward Scoring

Synthetic Data Gen

**Data**  
Pretraining, preference,  
post training

**Models**  
Core, safety, customized

**Hardware**  
GPUs, accelerators, storage

# Llama Stack APIs

**Agentic Apps**  
End applications

**Agentic System API**  
System component orchestration

PromptStore

Assistant

Shields

Memory

Orchestrator

**Model Toolchain API**  
Model development & production tools

Batch Inference

Realtime Inference

Quantized Inference

Continual Pretraining

Evals

Fine Tuning

Pretraining

Reward Scoring

Synthetic Data Gen

**Data**  
Pretraining, preference,  
post training

**Models**  
Core, safety, customized

**Hardware**  
GPUs, accelerators, storage

# Llama Guard 3 8B

Supports 14 hazard categories:

- \* S1: Violent Crimes.
- \* S2: Non-Violent Crimes.
- \* S3: Sex Crimes.
- \* S4: Child Exploitation.
- \* S5: Defamation.
- \* S6: Specialized Advice.
- \* S7: Privacy.
- \* S8: Intellectual Property.
- \* S9: Indiscriminate Weapons.
- \* S10: Hate.
- \* S11: Self-Harm.
- \* S12: Sexual Content.
- \* S13: Elections.
- \* S14: Code Interpreter Abuse.

Llama Guard 3 8B is multilingual

# Model Tool Chain API

## Model Toolchain API

Model development & production tools

Batch Inference

Realtime Inference

Quantized Inference

Continual Pretraining

**Evaluations**

**Finetuning**

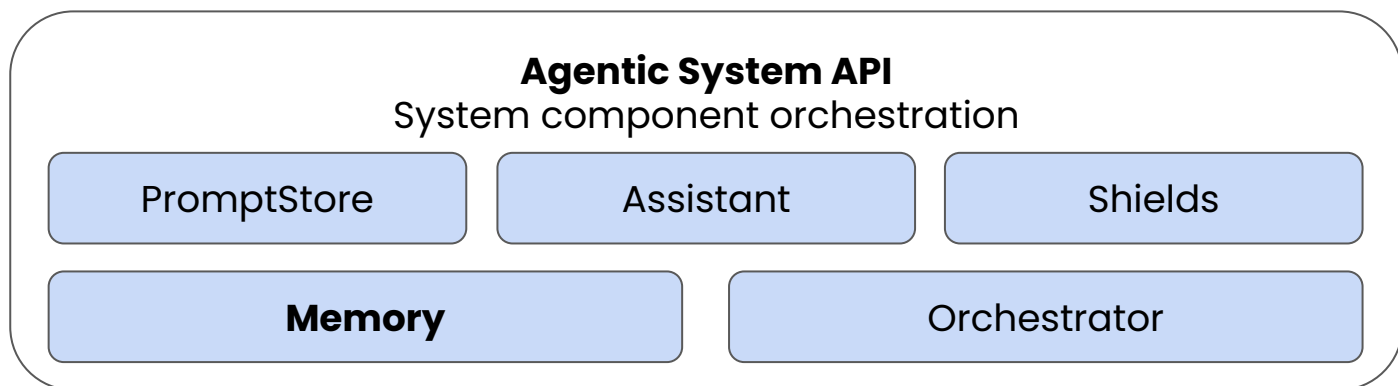
Pretraining

**Reward Scoring**

**Synthetic Data Gen**

- **Inference** – to support serving the models for applications
- **datasets** – to support creating training and evaluation data sets
- **Finetuning** – to support creating and managing supervised finetuning (SFT) or preference optimization jobs
- **Evaluations** – to support creating and managing evaluations for capabilities like question answering, summarization, or text – generation
- **Synthetic Data Generation** – to support generating synthetic data using data generation model and a reward model
- **Reward Scoring** – to support synthetic data generation
-

# Agentic System API



- **memory**- to support creating multiple repositories of data that can be available for agentic systems
- `agentic_system` - to support creating and running agentic systems.
- The sub-APIs support the creation and management of the steps, turns, and sessions within agentic applications.
  - `step` - there can be inference, memory retrieval, tool call, or shield call steps
  - `turn` - each turn begins with a user message and results in a loop consisting of multiple steps, followed by a response back to the user
  - `session` - each session consists of multiple turns that the model is reasoning over
  - `memory_bank` - a memory bank allows for the agentic system to perform retrieval augmented generation



# Llama Guard

## Llama Guard 3 models:

- **8B** – fine-tuned on Llama 3.1 8B. It provides industry leading system-level safety performance and is recommended to be deployed along with **Llama 3.1**.
- **11B Vision**: fine-tuned on **Llama 3.2** vision model and designed to support image reasoning use cases and was optimized to detect harmful multimodal (text and image) prompts and text responses to these prompts.
- **1B** – a lightweight input and output moderation model, optimized for deployment on mobile devices.