MCP Dev Summit Nairobi

The Context Window Is the Budget

Running MCP servers on small local models

Brian · Senior Software Engineer, Banda Health

How I got here

Patient summary *Example summary, anonymised test record*

This is a 50-year-old male with multiple active problems including hypertension, Type 2 Diabetes Mellitus, stroke, and LUTS/OAB. Today's visit was for common cold symptoms and follow-up care. Key vitals recorded on 2026-06-05 were a systolic blood pressure of 100 and diastolic of 77 mmHg, with a pulse of 90 BPM. The most recent HbA1c measurement (2026-05-04) was 11.2%. The plan emphasizes medication adherence for hypertension and diabetes, continued daily home BP monitoring, and follow-up on 2026-06-10 to assess progress and repeat the HbA1c on 2026-07-27. He has no recorded allergies.

First run: 1,968 thinking tokens **122 answer tokens**



1. **One headline took 5 minutes**

   A local model summarised patient records. It ran at 8 tokens per second.

2. **1,968 hidden tokens**

   Thinking used 1,968 of 2,090 tokens, for a 73-word answer.

3. **Then I counted the tokens**

   Two settings later: 5 minutes became 29 seconds.

Prototype only. Not in production.

Why run a model locally?



Patient data stays here

Kenya's Data Protection Act treats health data as sensitive personal data.

A cloud with servers outside Kenya counts as a cross-border transfer.

That transfer needs explicit consent and safeguards.

What about cost?

Cloud tokens are cheap now. Cost alone is a weak reason.



A fixed cost after you buy the hardware



No card, no billing account, no exchange-rate risk



Keeps working when the network is down

Also: a fixed model version, offline security tests, model editing (abliteration).

What this talk is, and is not

This talk is

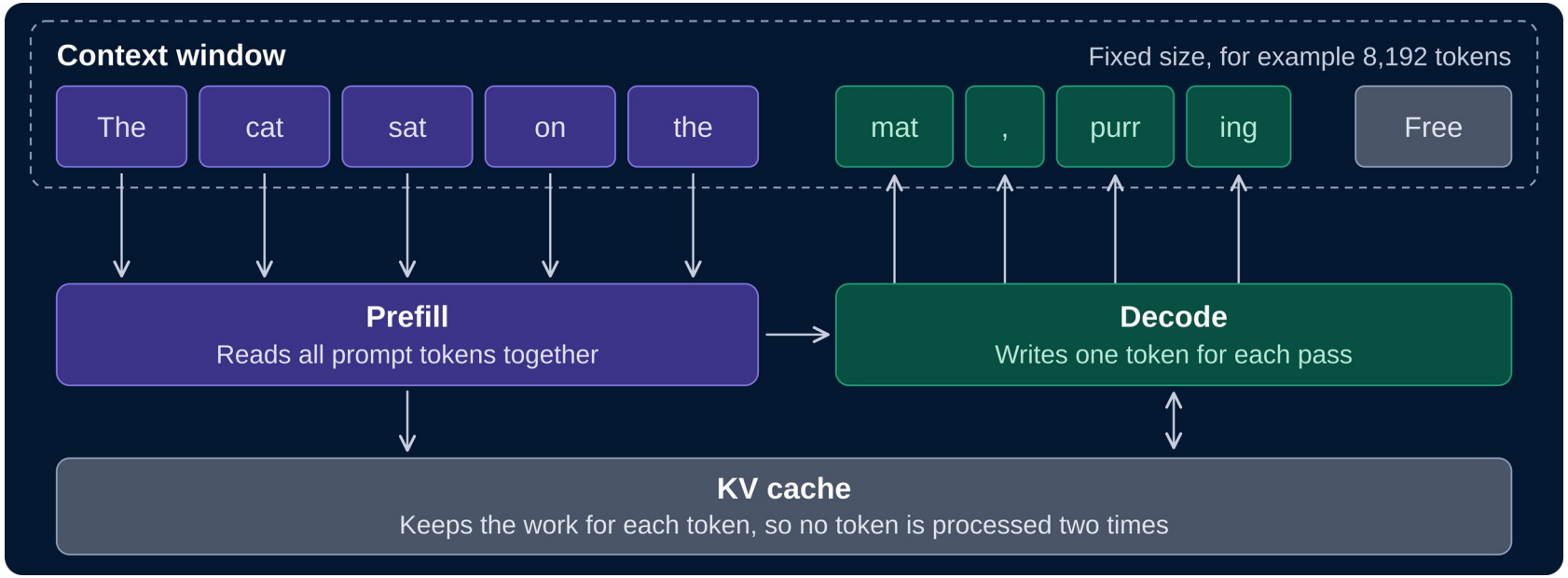
- Lessons from a patient-summary prototype
- An open MCP server for DHIS2, used for Kenya's health data
- Numbers measured on one laptop, RTX 5060, 8GB

This talk is not

- A production deployment story
- A benchmark of every model
- A cloud vs local debate

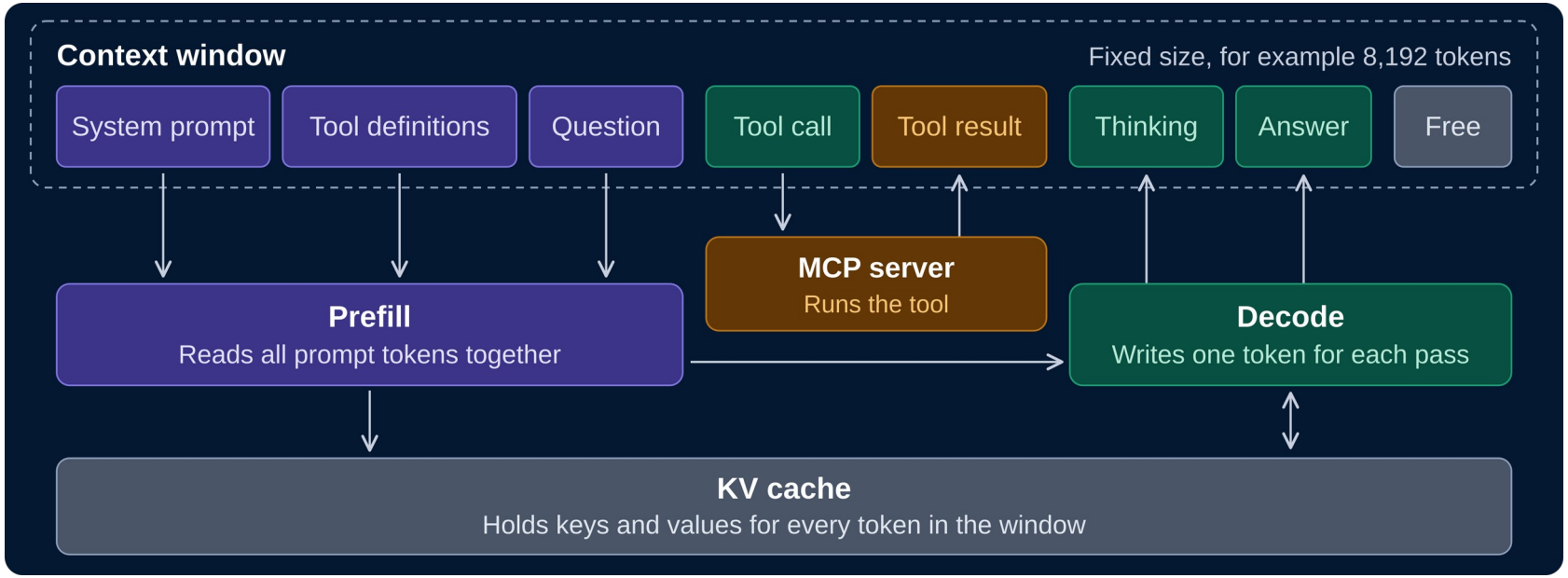
Tokens are the currency

A token is a piece of a word, about 4 characters. 100 tokens are about 75 words.



What fills the context window

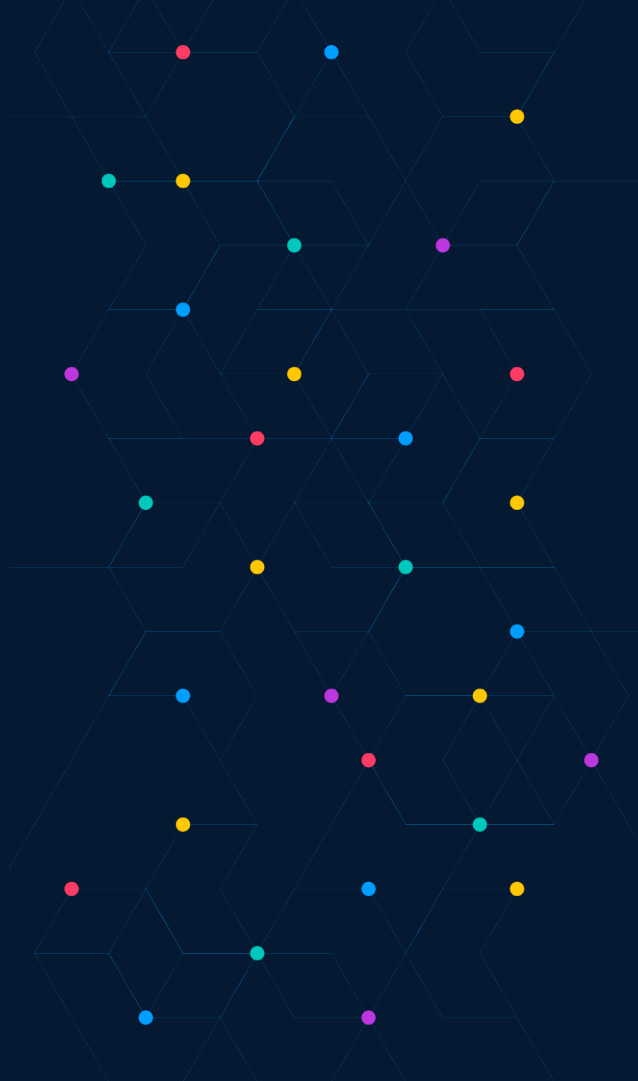
The context window is everything the model can see at once. It is measured in tokens.



604

tokens spent on tool definitions before the user asks anything

7.4% of an 8k context. Two tools.



The 8GB balance sheet

VRAM = model weights + KV cache + overhead



Qwen3.5 9B, Q4_K_M, 8k context, f16 KV cache. 6,480 of 8,151 MiB used.

Weights take three quarters of the card. On this model, an 8k KV cache costs about 0.2 GiB. The context is cheap in memory. It is expensive in time.

Autofit: the context you reserve also costs

Autofit splits the model between GPU and CPU so that a minimum context always fits. My minimum was set to 32k tokens.

Autofit reserved KV cache for 32k

Engine flag --fit-ctx 32000. The headline needed far less.

Weight layers moved to the CPU

The reserved KV cache left no room for the full model.

Decode fell to 8 tokens per second

Prefill stayed at 400–800 tok/s. GPU only 17–28% busy.

Same laptop, three models

Model	Weights	Decode
Gemma 4 12B	6.5 GiB	**~8 tok/s**
Qwen3.5 9B	5.24 GiB	46 tok/s
Gemma 4 E4B	4.97 GiB	69 tok/s

Fix: set the minimum to what you need (8k), then check the real engine flags.

A fuller context, slower answers

Context	VRAM	First token	Speed
4k	6,296 MiB	1.8 s	48 tok/s
8k	6,368 MiB	3.5 s	46 tok/s
16k	6,536 MiB	7.2 s	42 tok/s
32k	6,888 MiB	15.8 s	36 tok/s
64k	7,342 MiB	37.7 s	28 tok/s
96k	Will not load	—	—

Qwen3.5 9B, Q4_K_M, Q8_0 KV cache. Context filled to about 90%.

21× longer

wait for the first token, 4k to 64k

–42% speed

48 to 28 tokens per second

Where the tokens go

Tool definitions

Paid on every turn, before the first question.

604

tokens, 2 tools

Tool results

Paid on every tool call, and again on every later turn.

74

tokens lean, 271 raw

Thinking

Paid on every answer. The user never sees it.

53–119

tokens per answer

Tool results: raw vs lean

Raw DHIS2 analytics

```
rows: [["dy1a1mseGR7","202609",
  "ImspTQPwCqd","58157"]]
metaData: { items: { ... },
  dimensions: { ... } }
```

271 tokens



Lean server result

```
Sierra Leone | Sep 2026 |
ANC visits total | 58157
```

74 tokens

Same answer. 3.7× fewer tokens. Flat rows, with names resolved on the server.

Four runs, one question

	Thinking off	Thinking on
Raw payload	**A** Tokens used: **1,327** First answer token: **0.19 s** Correct: **5 / 5**	**B** Tokens used: **1,455** First answer token: **3.19 s** Correct: **5 / 5**
Lean payload	**C** Tokens used: **861** First answer token: **0.18 s** Correct: **5 / 5**	**D** Tokens used: **921** First answer token: **1.49 s** Correct: **5 / 5**

Qwen3.5 9B Q4_K_M · 8k context · 5 runs per cell · all 20 answers correct

Live demo: the tool loop at 8k

1. Ask the four-indicator question with raw tool results.
2. Ask again with lean tool results.
3. Compare the context used and the answer.

Setup

Terminal scripts, Qwen3.5 9B Q4_K_M

DHIS2 in Docker, on this laptop

Network cable out



Backup: recorded replay

Where it breaks: the tool loop

Question: which of ANC 1, 2, 3 and 4 had the lowest coverage? One search and four analytics calls. Every result stays in the context.

Context	Raw payloads	Lean payloads
4k	**Refused**	3 / 3 correct
8k	**Refused**	3 / 3 correct
16k	3 / 3 correct	3 / 3 correct

4×

the context for the same question. Raw needs 16k. Lean answers in 4k.

Where both fit: 11,551 tokens raw against 2,128 lean (5.4×). Sizes tested: 4k, 8k, 16k.

The failure that matters

search_indicators

```
{"query": "ANC"}
```

Correct call

query_analytics

```
"Uvn6LCg7dVU" = ANC 1 Coverage (%)
```

Wrong indicator

Answer

```
"109.52 ANC visits"
```

Wrong number

True answer

58,157

ANC visits total

No crash. No error. Valid JSON.

The system did not break. It gave a plausible wrong number.

What to do on Monday

- Count the tokens in your tool definitions.
- Return short named rows, not raw API payloads.
- Reserve only the context you need. Check the real engine flags.
- Turn thinking off per request. Turn it on only where it helps.
- Test the full tool loop, and check the answer, not only the call.
- Watch for silent failures: HTTP 200 errors and trimmed history.

Questions

Server: github.com/brianmituka/dhis2-mcp

Brian · Banda Health · X: @brianmituka



MCP
Dev Summit
Nairobi

Backup: the price of saving

KV cache type	VRAM at 8k	Correct runs
F16	6,480 MiB	10 / 10
Q8_0	6,364 MiB	10 / 10
Q4_0	6,300 MiB	9 / 10

Full tool loop against the MCP server, temperature 1.0, 10 runs each.

Q4_0 saves 180 MiB.

One failure in ten does not prove a cost.



MCP Dev Summit Nairobi