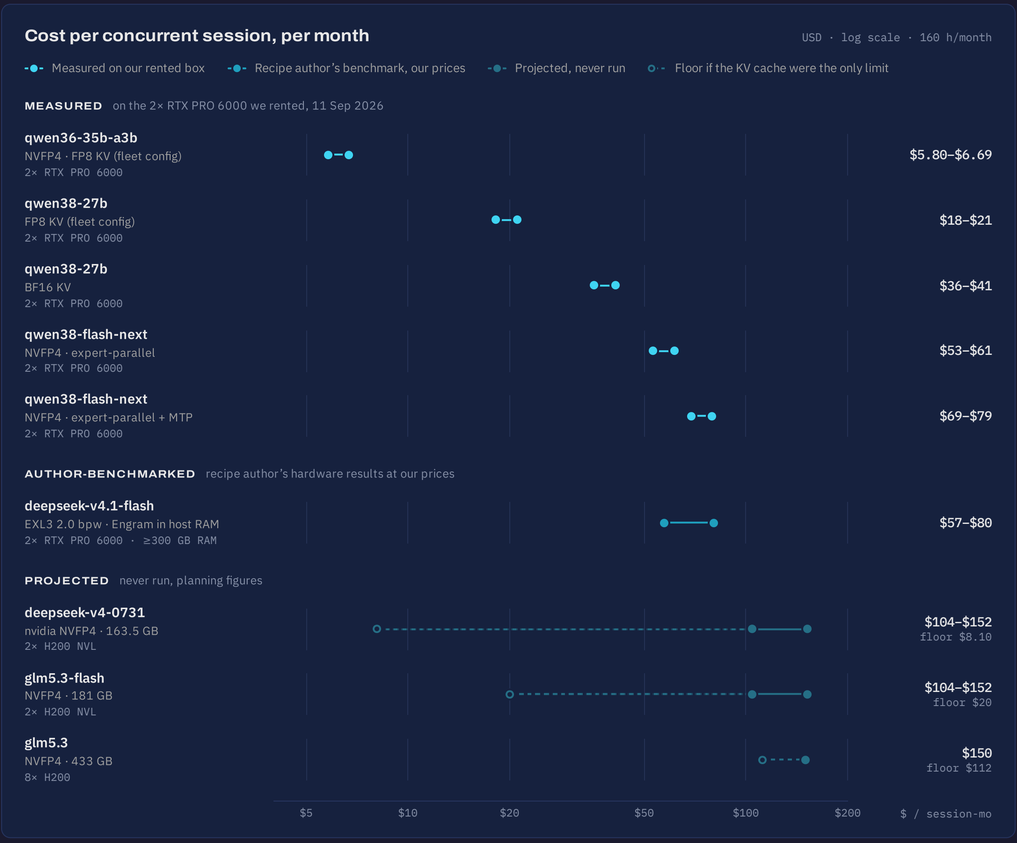
Flotilla Session Rate Card

What one concurrent session with a 262K-token context costs per month at 160 rented hours, for every model in scope. Each range spans the vast.ai offers we actually saw for a box that can run the model. Measured rows come from our own rented hardware; projected rows lean on a concurrency assumption.

`box $/h × 160 h ÷ concurrent 262K sessions per box = $/session-month`

Concurrent sessions are the lower of two limits: what fits in the KV cache, and what the engine will run at once.



Inputs behind every row

Hourly prices are the range of real offers seen on 11–12 Sep 2026 for a box that meets each model's requirements. Box cost is the hourly price times 160 hours.

Evidence	Model and configuration	Box	$/h	Box, 160 h	Concurrent sessions	$/session-month	Engine	Basis
Measured	**qwen36-35b-a3b** NVFP4 · FP8 KV (fleet config)	2× RTX PRO 6000	$1.97–$2.27	$315–$363	54.27	$5.80–$6.69	vLLM nightly e7edf17c	Measured KV ledger on the rented box; the cache fills first
Measured	**qwen38-27b** FP8 KV (fleet config)	2× RTX PRO 6000	$1.97–$2.27	$315–$363	17.28	$18–$21	vLLM nightly e7edf17c	Measured KV ledger; the cache fills first
Measured	**qwen38-27b** BF16 KV	2× RTX PRO 6000	$1.97–$2.27	$315–$363	8.86	$36–$41	vLLM nightly e7edf17c	Measured KV ledger; the cache fills first
Measured	**qwen38-flash-next** NVFP4 · expert-parallel	2× RTX PRO 6000	$1.97–$2.27	$315–$363	5.92	$53–$61	vLLM nightly e7edf17c	Measured KV ledger; its attention forces a BF16 cache
Measured	**qwen38-flash-next** NVFP4 · expert-parallel + MTP	2× RTX PRO 6000	$1.97–$2.27	$315–$363	4.58	$69–$79	vLLM nightly e7edf17c	Measured KV ledger; the MTP draft uses cache too
Author-benchmarked	**deepseek-v4.1-flash** EXL3 2.0 bpw · Engram in host RAM	2× RTX PRO 6000 · ≥300 GB RAM	$2.86–$4.00	$457–$640	8 cache alone ≈ 14.5	$57–$80	vLLM day-0 image + EXL3	The recipe runs at most 8 sequences; its author benchmarked up to 4 streams
Projected	**deepseek-v4-0731** nvidia NVFP4 · 163.5 GB	2× H200 NVL	$5.21–$7.58	$834–$1,212	8 planned cache alone ≈ 41–103	$104–$152 floor $8.10	vLLM or SGLang	Assumes 8 per 2 GPUs, borrowed from V4.1-Flash; cache size per token inferred from the V4.1 card
Projected	**glm5.3-flash** NVFP4 · 181 GB	2× H200 NVL	$5.21–$7.58	$834–$1,212	8 planned cache alone ≈ 42	$104–$152 floor $20	vLLM per-model image	Assumes 8 per 2 GPUs; only 11 of its 45 layers hold KV cache
Projected	**glm5.3** NVFP4 · 433 GB	8× H200	$30.04	$4,806	32 planned cache alone ≈ 43	$150 floor $112	vLLM or SGLang	Assumes 8 per 2 GPUs; cache size per token derived from its config

Reading it

Two different limits fill a box.

The qwen rows run out of KV cache first, so their session counts come straight from the KV ledger we measured. DeepSeek-V4.1-Flash runs out of compute first: its recipe runs at most 8 sequences, though its cache could hold about 14.5. Its author benchmarked only up to 4

streams, and at 4 a session would cost $114–$160. For contrast, qwen36-35b-a3b held 64 concurrent 27K-token agent sessions at 0.37 s p95 time-to-first-token or better, which is $4.92–$5.68 each.

Projected rows are planning figures, not quotes.

deepseek-v4-0731, glm5.3-flash and glm5.3 have never run for us. They assume 8 concurrent sessions per 2 GPUs, borrowed from V4.1-Flash's recipe. The dashed line runs down to what they would cost if the KV cache were the only limit, which is where the lower figures quoted earlier came from. Quote the solid range and treat the dashed end as a best case.

Prices moved all day, and some costs are left out.

2× RTX PRO 6000 went for $1.97–$2.27/h. A V4.1-Flash host with at least 300 GB of RAM and CUDA 13 ran $2.86–$4.00/h, and the cheapest of those fails our 0.95 reliability floor. 2× H200 NVL ran $5.21–$7.58/h, and 8× H200 $30.04/h. Bandwidth, disk and cold-start time are not included.

Sources: `FLOTILLA-GUIDANCE.md` §0 and §1.10b, `flotilla/runs/dsv41.sh`, and lane commits `a88aed6a` and `78cf47fa`, plus the vast.ai market on 11–12 Sep 2026.

Every performance figure here is self-reported by hardware we do not control. DeepSeek-V4.1-Flash's numbers belong to its recipe author until our own run reproduces them.