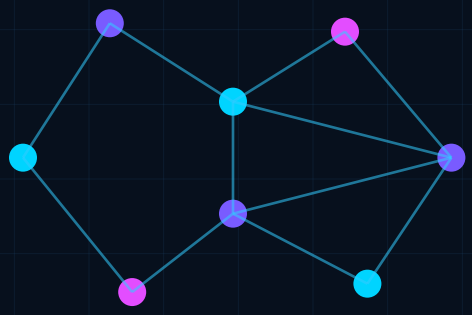
BROWSER-LOCAL AI

Browser-Local WebLLM (Architecture, Constraints, and WebGPU bindings)

Current-source review, evidence-bounded adoption decision, explicit limitations, reproducibility controls, and preserved source research note.



PERMANENT ID	EVIDENCE	ADOPTION	FRESHNESS
RES-006	A-	PILOT	ACTIVE WATCH

Freshness review: 2026-09-17 / Next scheduled review: 2026-12-16

RESEARCH CONTROL SURFACE

Required Research Fields

Each field remains independently reviewable; evidence strength does not imply production authority.

EVIDENCE GRADE — **A-**

Strength of the retained source set and technical basis.

SOURCE AUTHORITY — **3 registered authorities**

W3C WebGPU Specification; WebLLM Documentation; WebLLM: A High-Performance In-Browser LLM Inference Engine

FRESHNESS — **ACTIVE WATCH**

Reviewed 2026-09-17; dynamic sources require event-driven revalidation.

CONFLICTS — **VISIBLE / NOT SILENT**

Differences between specification, vendor behavior, research claims, and reproduced evidence must remain explicit.

LIMITATIONS — **EXPLICIT**

No certification, procurement approval, legal conclusion, or unbounded production claim is created by this report.

REPRODUCIBILITY — **REQUIRED**

Material claims require exact versions, representative data, failure cases, artifacts, and repeatable acceptance criteria.

ADOPTION POSTURE — **PILOT**

Pilot browser-local inference for privacy-sensitive, latency-tolerant, bounded workloads. Gate deployment on WebGPU support, memory pressure, model size, caching, browser isolation, and measured device compatibility.

REVIEW TRIGGER — **2026-12-16**

Scheduled at 90 days; earlier on material source, vulnerability, implementation, or contradictory-evidence change.

SOURCE AUTHORITY / FRESHNESS

Authority Register and Freshness Delta

Primary-source lineage is preserved; current-source changes are separated from unperformed implementation claims.

AUTHORITY 01

W3C WebGPU Specification

https://www.w3.org/TR/webgpu/

AUTHORITY 02

WebLLM Documentation

https://webllm.mlc.ai/docs/index.html

AUTHORITY 03

WebLLM: A High-Performance In-Browser LLM Inference Engine

https://arxiv.org/abs/2412.15803

2026-09-17 FRESHNESS DELTA

W3C WebGPU remains on the Candidate Recommendation Draft track; the publication history includes an August 20, 2026 draft.

Freshness policy: scheduled review + immediate review on source supersession, material release, vulnerability, retraction, or contradictory evidence.

Evidence Must Survive Contact With Reality

- Source authority is not implementation evidence. Product behavior, scale, security, interoperability, and operator fit remain environment-specific.
- Conflicting evidence is recorded rather than averaged away. Specification behavior, vendor implementation, preprint results, and production observations are separate evidence classes.
- Quantitative claims are informative until reproduced under documented hardware, software, data, topology, load, and failure conditions.
- Adoption posture is reversible. A stronger source set can coexist with a pilot-only or research-only recommendation when maturity, safety, economics, or operability remain unresolved.

ADOPTION / VALIDATION / REVIEW

Decision Gate and Validation Plan

ADOPTION POSTURE

PILOT

Pilot browser-local inference for privacy-sensitive, latency-tolerant, bounded workloads. Gate deployment on WebGPU support, memory pressure, model size, caching, browser isolation, and measured device compatibility.

1 / SOURCE	Pin exact versions, publication state, retrieved artifacts, and source hashes.
2 / PROTOTYPE	Demonstrate the core mechanism with representative data and instrumented execution.
3 / FAILURE	Exercise malformed input, dependency loss, stale state, overload, recovery, rollback, and misuse.
4 / BASELINE	Compare against an incumbent, classical, or simpler architecture using the same workload.
5 / ACCEPT	Record acceptance criteria, residual limitations, owner, expiration, and revalidation triggers.

EXPIRATION / REVIEW TRIGGER

Next scheduled review: 2026-12-16 (90-day cadence). Review sooner after material source revision, breaking implementation release, critical vulnerability, retraction, benchmark contradiction, changed workload, or changed legal/safety boundary.

8. Complete Source Research Note

SOURCE-LINEAGE NOTICE

The following material preserves the complete original source note, excluding only the conversational wrapper and outer Markdown fence. Legacy status labels and absolute language are retained for traceability and do not override the candidate status, limitations, or adoption decision in this edition.

Document ID: RES-006 Version: 1.0.0 (Definitive Registry Edition) Status: Published Research Note Classification: Public Organization: Mythos Systems (MYTHOS AI) Owner: Aaron Stovall Effective Date: 2026-07-16 Applies To: Front-end architects, AI engineers building local-first web clients (CALLISTO), and platform engineers evaluating browser-based inference capabilities Companion Standards: MYTHOS-WEB-0001 (WebGPU, WebLLM & Browser Isolation), MYTHOS-EMT-0001 (WASM & Edge Sandbox), MYTHOS-DAT-0001 (Vector DB & Local-First Sync), MYTHOS-AI-0014 (Inference Serving)

The architecture of web-based AI has historically been a lie. Applications claim to be "local-first" while silently routing every keystroke to a third-party cloud LLM API for inference. This introduces crippling latency (200ms+ per token) and violates basic data sovereignty.

The introduction of WebGPU and the WebLLM runtime (e.g., `mlc-ai/web-llm`) shatters this limitation. It is now architecturally possible to download a quantized LLM directly to the browser, cache it in the Origin Private File System (OPFS), and execute tensor operations natively on the user's GPU via compute shaders.

This research note defines the mechanics of the Browser-Local WebLLM architecture. It dissects the WebGPU bindings (how LLM math maps to GPU workgroups), the browser storage lifecycle (OPFS), and the hard physical constraints (VRAM ceilings, tab eviction) that govern what is and is not possible in a purely client-side sovereign AI application.

To execute an LLM in the browser without a backend server, the architecture must solve three problems: computation, memory, and storage.

- Compute (WebGPU): The browser does not have access to CUDA. It utilizes the WebGPU API, which provides a low-overhead, cross-platform abstraction layer to the underlying graphics driver (Vulkan, Metal, Direct3D 12).
- Memory (VRAM): The model weights and the KV-cache must reside in the GPU's VRAM. WebGPU allocates `GPUBuffer` objects for this purpose.
- Storage (OPFS): A 4GB model cannot be downloaded on every page load. The browser utilizes the Origin Private File System (OPFS) to permanently cache the model weights on the user's disk.

The WebLLM runtime (like MLC-LLM or `transformers.js`) acts as the orchestrator. It fetches the model from OPFS, loads it into WebGPU buffers, compiles the LLM architecture into WebGPU Shading Language (WGSL) compute pipelines, and executes the prefill and decode loops entirely within the browser tab's sandbox.

Traditional native inference engines (vLLM, llama.cpp) use C++ and CUDA. WebLLM must map these same matrix multiplications to WebGPU compute shaders.

2.1 The WGSL Compute Pipeline

In WebGPU, the LLM's attention heads and Multi-Layer Perceptron (MLP) blocks are compiled into WGSL compute shaders.

- The Mechanic: The WebLLM runtime creates a `GPUComputePipeline` for the matrix multiplication. It binds the weight buffers and the activation buffers to `@group(0)` binding indices.

- Execution: The GPU executes the shader in parallel across thousands of workgroups. Each workgroup computes a tile of the output matrix.
- Constraint: WebGPU lacks the fine-grained hardware intrinsics (like Tensor Cores via PTX) that native CUDA utilizes. WebGPU relies on standard 32-bit floating point or emerging 16-bit (f16) support. This means browser inference is currently slower per-FLOP than native CUDA, though still highly usable for sub-7B models.

2.2 The KV-Cache in Browser VRAM

During the decode phase, the model must store the Key and Value tensors for previous tokens.

- The Mechanic: WebLLM allocates a `GPUBuffer` mapped as `STORAGE` to hold the KV-cache. As the context window grows, this buffer expands.
- Constraint: Unlike native vLLM, which utilizes PagedAttention to manage VRAM fragmentation, browser WebLLM runtimes often pre-allocate a contiguous buffer for the max context length. If the context exceeds the buffer, the application crashes (Out of Memory).

Executing AI in a browser tab is an act of brute force against a highly constrained environment. Engineers must understand these hard limits.

3.1 The VRAM Ceiling

A browser tab does not get access to the entire GPU. The OS and the browser partition GPU memory. If a user has an 8GB GPU, the browser might only allow WebGPU to allocate 2-4GB of VRAM before throwing a `GPUOutOfMemoryError`.

- Impact: You cannot run a 70B model in the browser. You cannot even run a 13B FP16 model. WebLLM is strictly constrained to heavily quantized models (INT4 AWQ or GGUF), typically in the 1B to 7B parameter range (e.g., Llama-3-8B-Instruct-Q4, Phi-3-Mini).

3.2 The Cold-Start Penalty

Loading a 4GB model from OPFS into VRAM is an I/O-intensive process.

- Impact: The first prompt a user sends will take 5 to 15 seconds to process (depending on disk speed and GPU bandwidth) as the weights are transferred. If the user navigates away or the tab is evicted, the process must restart.
- Mitigation: The model loading MUST occur inside a Web Worker. If it runs on the main thread, the browser UI will freeze, and the OS will display a "Page Unresponsive" warning.

3.3 Tab Eviction and Lifecycle

Browsers aggressively reclaim memory. If a user switches to another memory-intensive application, the OS may suspend the browser tab. When the tab is suspended, the WebGPU context is lost, and the VRAM is zeroed.

- Impact: The user's in-progress conversation (the KV-cache) is destroyed. When they return, the application must either restart the session or reload the KV-cache from memory, causing latency spikes.

To make WebLLM production-ready (as mandated by MYTHOS-WEB-0001), the architecture must actively manage the browser's constraints.

4.1 Dynamic Model Routing

Do not force the browser to run a model it cannot handle. The CALLISTO architecture requires hardware discovery.

- The app queries `navigator.gpu` and `GPUAdapter` to estimate available VRAM.
- If VRAM is high (e.g., > 6GB available), load `Llama-3-8B-Q4`.

- If VRAM is low (e.g., < 2GB available), load `Phi-3-Mini-Q4` or route the request to a local edge node (RA-006) via WebRTC.

4.2 OPFS Caching and Service Workers

A 4GB download is unacceptable on every page load.

- The app must use a Service Worker to intercept the model download request.
- The Service Worker fetches the model from the CDN the first time, writes it to OPFS, and serves all subsequent requests from the local disk. This guarantees zero-latency model loading on return visits.

4.3 Context Compaction (KV-Cache Eviction)

Because VRAM is scarce, the WebLLM runtime cannot hold an infinite context window.

- The architecture MUST implement context compaction. When the KV-cache approaches 80% of the allocated VRAM, the runtime must summarize the earliest parts of the conversation, drop those KV blocks, and inject the summary as a system prompt. This prevents OOM crashes during long chat sessions.

Browser-Local WebLLM is not a replacement for datacenter-scale inference. It is a tool for sovereign, zero-latency AI. By mapping LLM tensor operations to WebGPU compute pipelines and caching weights in OPFS, we transform the browser from a dumb terminal into a self-sufficient AI compute node. While hard VRAM constraints limit the model size, the CALLISTO architecture's dynamic routing and context compaction ensure that the user retains absolute cognitive privacy and offline resilience without sacrificing usability.

```
A cloud API is a leash.
A browser tab is a sandbox, but it is a sovereign sandbox.

WebGPU binds the math to the silicon.
OPFS binds the weights to the disk.

The VRAM is small, but the privacy is absolute.

> The web may be distributed.
> Local-first execution must be absolute.
```

End of Research Note RES-006

9. Source Register

Source	Authority and Status	Freshness Control
W3C WebGPU Specification https://www.w3.org/TR/webgpu/	Primary web standard Living W3C specification Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
WebLLM Documentation https://webllm.mlc.ai/docs/index.html	Primary project documentation Living Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
WebLLM: A High-Performance In-Browser LLM Inference Engine https://arxiv.org/abs/2412.15803	Primary research preprint Preprint Published: 2024	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.

10. Lineage, Review, and Publication Record

Record	Value
Original source file	RES-006_Browser_Local_WebLLM_Architecture_Constraints_WebGPU_Bindings.md
Original source words	1272
Upgrade type	Enterprise research report; source and freshness validation; limitations; adoption decision; reproducibility plan
Generated on	2026-07-22
Current status	Candidate Research Report
Publication authority	Formal Mythos approval not yet recorded
Next review	90 days or event-driven trigger, whichever occurs first