contextbridge_hackathon_challenge_en

Hackathon Challenge: ContextBridge – The Missing Link Between AI and Enterprise Systems

The Problem

Imagine the following real-world scenario:

A global company wants to implement an internal AI assistant to support teams in tasks such as:

- Responding to clients based on previous emails, contracts, and CRM data
- Generating real-time reports from internal systems (SAP, SQL databases, SharePoint)
- Assisting developers and analysts in finding technical context scattered across documentation, incidents, code, etc.

There are dozens of tools on the market: copilots, RAG wrappers, autonomous agents.
But **none of them solve this** in a **private, controllable, and extensible** way:

- There's no **context orchestrator** capable of connecting real sources, dynamically building what the AI needs to know, and delivering it with traceability.
- Existing solutions (LangChain, LlamaIndex, Flowise, Semantic Kernel) **handle parts of the problem**, but don’t fully address it for real enterprise use.
- Most importantly, none allow this to be done in a **private, open-source, on-premises environment**.

The Challenge

Develop an **open-source** solution named `ContextBridge` that acts as a **semantic middleware** between LLMs and enterprise systems. This tool should be capable of:

Connecting to enterprise sources

- Mocked or real: SharePoint, SAP, databases, CRMs, document directories, REST APIs, etc.

Orchestrating the context

- Through **rules and triggers** (e.g., request type, source, user, topic)
- With **reusable context profiles** (e.g., commercial responses, internal reports, log analysis)

Injecting the context into the LLM

- In real-time, with logs of every interaction
- Compatible with OpenAI, Claude, Ollama, etc.

- Avoiding hardcoding or fixed prompts

Ensuring privacy and traceability

- Local or Docker-based deploy with no cloud dependency
- Logging of everything retrieved, injected, and responded
- Access control per source/context

Integration with Frontends and Existing Apps

`ContextBridge` must allow **zero-refactor integration** with existing applications, such as:

- Websites with forms
- Enterprise backoffices in React, Angular, or Vue
- Legacy systems with minimal integration effort

The solution must provide:

- **Lightweight SDK for front-end** (JS/TS)
- **REST or GraphQL API** for sending metadata (user, intent, page, etc.)
- **Simple hooks or wrappers** (e.g., `useContextBridge()`)

Goal: allow any app to trigger contextual enrichments with **minimal coupling**, and receive back the necessary context to enhance the LLM interaction — without changing the app's business logic.

What You Can Build

- **Context engine**: selects and assembles the best data set for a given task
- **Integration pipeline** with at least two different data sources
- **LLM proxy** that intercepts prompts, enriches with context, and routes to the model
- **Logging system** and simple dashboard (CLI or UI)
- **SDK/API for front-end and backend** to integrate with real-world applications

---

Suggested Stack (Optional)

- Backend: Python or Node.js
- RAG Orchestration: LlamaIndex, LangChain, or custom
- Vector DB: Weaviate, Qdrant, or SQLite+FAISS
- LLM: OpenAI API, Ollama, Azure OpenAI, Claude
- Interface: CLI, Streamlit, or React
- Protocols: **Compatibility with MCP (Model Context Protocol)** is a big plus

Bonus for Advanced Teams

- Implement a **semantic rules engine** (e.g., if client = X, fetch Y)
- Create a **context journey visualizer** (audit trail)
- Support for **multi-user and multi-model**
- Direct **MCP (Model Context Protocol)** compatibility

Notes

This challenge is **open-source by design**.
We believe “context engineering” will be as important as the models themselves.
Help us build the bridge — **not just another stack, but a new layer for the corporate internet**.