Member-only story

Reference Architecture for Team AI Productivity

Patterns for building enterprise grade AI web chat portals

10 min read · Jun 19, 2025

Matt Eland Follow

Listen Share More

Let's discuss a sample reference architecture for providing a secure and convenient way for your organization to chat with pr approved AI capabilities.

Previously in this series we discussed Website RAG Chat and Developer AI Productivity reference architectures. Those architectures are valid and helpful for delivering rich AI capabilities to your customers and developer team, but what about t rest of your organization?

In this article we'll lay out a reference architecture that allows different members of your organization to safely enhance the workflows through AI, and do so with the knowledge that organizational data is being handled securely and intellectual property is being respected.

While this architecture could work using a variety of different technologies, specific examples and screenshots will feature the popular Open WebUI conversational AI platform.

Use cases

You may not immediately think that providing a team-wide AI system would be extremely helpful, but when we unveiled our "Chat EDJE" conversational AI solution at Leading EDJE we noticed some immediate and profound impacts on our teams for all types of users.

While developers were excited about these capabilities and took advantage of them for complex tasks like working on improving SQL performance by comparing a complex query with an execution plan, we also saw some tremendous benefits for non-developers.

We saw project managers gain access to ways to generate new ideas relevant for the teams they were on, web designers able to gain access to specialized insights for technical search engine optimization (SEO) considerations, analysts able to help quickly find and fix anomalies in data, and all team members benefitted from summarization and email drafting / proof checking capabilities.

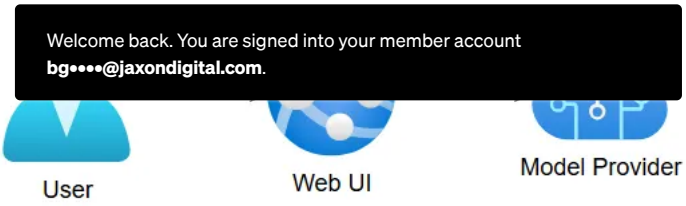
While we weren't initially sure that AI tooling would truly help all team members, we continue to be blown away by the impact of secure, reliable, AI tooling applied to the entire organization and governed by the organization's IT staff.

Let's talk about how this works.

A sample architecture

A team productivity conversational web AI chat architecture consists of the following required components:

- A **web chat portal** for hosting the conversations
- One or more registered **model provider** that makes public or private conversational AI models (typically LLMs) available to the team
- A **management layer** allowing administrators to configure and control the model



A team AI productivity architecture with only the required components

Conversational web AI chat architectures may also include the following optional components:

- A **persistence layer** for storing past conversation sessions
- A **context layer** that provides additional documents, tools, or capabilities that can augment the conversation

A team AI productivity architecture supported by additional context

Let's walk through each of these required and optional components, talk about what they are and the various choices you might need to make with each one.

Web Chat Portal

The first component is the most obvious: you need a centralized **web chat portal** where users can go to ask questions of your system. This typically looks like a web page hosted on your organization's intranet or on the internet and is secured via your organization's preferred authentication options.

The web chat portal allows you to select a prior conversation to continue (if a persistence layer is present) or start a new conversation.

Welcome back. You are signed into your member account
bg••••@jaxondigital.com.

Starting or resuming a conversation in Open WebUI

When starting a new conversation, users must select a model to interact with from the approved list of models, specify a textual query / prompt, and optionally include documents, files, or web links to act as additional sources of context before sending the message.

Your web chat portal *may* stream the completions your model provides so the user can see the response in real-time, but once the response is fully complete it should show up in your portal and include additional context, links, and feedback mechanisms.

A chat response in Open WebUI showing additional actions

Some web chat portals may also support asking the same question of multiple models to compare their responses or may give you the ability to edit your existing messages and resend them to regenerate a response.

It's important to know that when a persistence layer is present, entire interactions with the system may be stored temporarily or permanently for auditing or quality control purposes. This is particularly true when users provide positive or negative feedback. As a result, users should be informed that their conversations may be stored and retrieved by your organization's IT staff or potentially other personnel and the system should therefore be treated with the same level of professionalism as you would expect from a communications platform like Slack, Teams, or Discord.

Model Provider

A **model provider** is a connector that connects your web chat portal to organizationally approved large language models (LLMs) that users are allowed to interact with.

Model Providers allow the user to select from pre-approved models

Your organization may use public models like those deployed on OpenAI, instanced / dedicated models hosted on a service like Azure, or your team may self-host models using something like Ollama or LM Studio. Your team could even use a combination of these by specifying multiple model providers.

Your list of approved models will grow and shrink over time as new models arrive, are reviewed and approved, and as older models get retired and replaced with newer ones. The most important thing to remember with your model selection is that you **only list models your organization is comfortable with people using**. If a model does not meet your organization's IP security needs (for example, it retains logs for training future models), it should not appear in your list of models to your end users. In this way, users working with your solution know that they are in compliance with organizational AI policies.

You may wonder "why don't organization's just use a single approved model? Why give user choices?". While providing choices to users may raise the barrier to entry slightly for some users, the overall benefits of having different models is usually worth it. Because different models are good at different tasks and have different basic characteristics in terms of speed, accuracy, and cost, it can be helpful to allow your users to choose.

Additionally, you may find that some models temporarily go offline — particularly if you're using multiple model providers — and it can be helpful to have backup resources for people to consider.

Management Layer

Most AI chat solutions have some form of management or configuration associated with them. The **management layer** allows your IT admin team to configure your web chat system and connect it to various models and other providers.

Connecting Open WebUI to different model providers

Once model providers are configu roviders that should be available to your users:

Configuring available models

Organizations using pay-per-usage models can sometimes use the management layer to limit the budget of individual users in order to ensure a predictable maximum expense per week per user limit.

Persistence Layer

Most web portals with a management layer will also have some form of a **persistence layer** that allows storing past conversations. This is done for convenience for users who which to refer to past conversations or resume them and can also help your organization's IT team manage and monitor its AI infrastructure.

In evaluating models and compliance, admins may be able to see some or all of the private interactions with users, depending on how the persistence layer is configured and if any rolling delete or anonymization capabilities are present. While this helps evaluate which models are actively being used and how they're performing, this capability and how it may be used should be disclosed to your employees as some employees may include context they intended to be private in even legitimate interactions with the system. For example, an employee brainstorming a presentation with an LLM may choose to disclose private medical information about physical or mental conditions that might impact their performance in order to perform their company-assigned tasks more effectively.

Your persistence layer could be as simple as a series of configuration files, or it could be a relational or document database. Some persistence layers may even use a vector store to store text embeddings allowing for searching past conversations or indexed documents. The capabilities of your persistence layer vary based on the overall solution you're using and will be strongly tied to whatever solution you choose.

Context Layer

Perhaps the most exciting of all the parts of an AI solution is the **context layer**. The context layer is able to provide your LLM with additional knowledge and capabilities including:

- **Tools / functions** that can be called to produce a result. Some tool examples might involve checking current weather in an area, tracking a package that's out for delivery, or searching the internet.
- **Prompts** define common text instructions for carrying out a task in a way that helps multiple people on your team
- **Resources** such as documents, web pages, and additional pieces of information that can help provide additional context.

These capabilities are typically integrated using a standardized format like Model Context Protocol or OpenAPI endpoints.

When a user sends a request to the LLM, these additional capabilities will also be sent along to the LLM and it *may* choose to take advantage of them in order to fulfill the user's request. This makes these additional capabilities a form of a retrieval-augmented generation (RAG) data source.

By integrating additional tools and capabilities into your AI systems, you are offering unique value for your organization that they cannot find in another tool. These capabilities are your unique way of adding in additional context to your organization that will help employees do their jobs more effectively. This context can include:

- **Tools** of looking up the status of different work items, orders, or customers

- **Resources** documenting standard definitions, systems, and workflows
- **Prompts** that help generate ou ndards

In short, the context layer is something that is uniquely yours and can be uniquely controlled by your organization.

These capabilities can be so valuable that some organizations even offer a shared architecture that encapsulates these tools into a MCP server that is shared between the web AI chat tooling and individual developer productivity solutions as shown here:

Using the same shared resources between developer teams and web application users

In this way all employees can take advantage of organizational knowledge, standards, and capabilities when performing their work, regardless of what that work entails.

Additional Integrations

Some web AI chat systems may include additional integrations including:

- Text to speech capabilities that read aloud responses from the system
- Speech to text capabilities that allow you to talk to your LLM
- Image generation via ComfyUI, OpenAI, Gemini, or other providers
- Web search capabilities (essentially a built-in tool provided by your platform)
- Direct code execution capabilities in sandboxed environments

This list of capabilities will vary depending on what web chat provider you selected and will change over time as industry trends evolve.

Securing your chat provider

While providing your users with a curated list of models is fantastic for helping users interact with AI in approved ways, these same capabilities can be a target for attackers as well.

If you do not properly secure your AI web chat capabilities it is possible that an attacker can discover your endpoint and use it to cause damage such as:

- Incurring charges against pay-per-use AI models
- Conduct a denial of service attack against your AI models by attempting to exhaust your rate limit capabilities for certain LLMs, denying legitimate users access to these resources
- Access sensitive information stored in prompts or resources

- Exploit tools to perform additional attacks such as searching your knowledgebase, querying data stores, or other actions dependent on the exact natur

There are a number of ways of remedying these vulnerabilities including:

- Properly researching the various web AI chat providers to ensure they meet your security and administration needs
- Requiring users to log in via an API key, LDAP, or some other form of authentication
- Configuring firewall rules to require a VPN to access AI tooling
- Setting sensible rate limiting or access permission on groups of users so a single compromised user cannot inflict massive damage to the organization

While any new system carries new attack vectors for malicious users, one of the realities of a world where AI tools are ubiquitous is that your users will find AI tooling that meets their needs. Your goal as an organization should be to make sure that when they do this, they do it in an approved way that also meets the organization's data stewardship and security needs.

Conclusion

Conversational AI systems are powerful ways of augmenting your entire team's capabilities, and a web AI chat portal is an effective way to provide a secure means for your organization to innovate with AI in approved and cost-effective ways. What's more, the ability to integrate your organization's context through resources, prompts, and tools is an offering that no other AI chat toolset will provide — and it can be easily integrated into other solutions such as developer AI productivity architectures.

We've been amazed at the things our team at Leading EDJE has been able to do with properly governed AI — both internally and for our clients — and we'd love to discuss how you can move forward with AI.

AI | Software Architecture | Productivity | Software Development | Security

I. N. Palacios
Jun 23

There's no reference architecture in your post

5 Reply

Md Nabil Hossain | Marketing Specialist
Aug 17

Matt Eland, your article is an excellent roadmap for implementing AI across an organization. I appreciate how you break down the architecture, from web portals to context layers, showing practical ways teams can safely and effectively leverage AI for productivity.

Reply

More from Matt Eland and Leading EDJE

In Leading EDJE by Matt Eland

Tracking AI system performance using AI Evaluation Reports

Measure, explore, and track your AI system's performance over time using .NET

Sep 9 42

In Leading EDJE by Matt Eland

Reference Architecture for AI Developer Productivity

Reference Architecture for AI Developer Productivity

May 6 242 3

In Leading EDJE by Matt Eland

Document Search in .NET with Kernel Memory

Simple web scraping, document indexing, RAG search, and chat

May 20 246 3

Matt Eland

OneOf<Us> Discriminated Unions in C#

Let's look at a C# library for Discriminated Unions that gives a functional way of differing behavior based on different types of objects.

Sep 22, 2019 29 1

See all from Matt Eland

See all from Leading EDJE

Recommended from Medium

In ITNEXT by Animesh Gaitonde

Solving Double Booking at Scale: System Design Patterns from Top Tech Companies

Learn how Airbnb, Ticketmaster, and booking platforms handle millions of concurrent reservations without conflicts

Oct 7 1.3K 19

Tosny

7 Websites I Visit Every Day in 2025

If there is one thing I am addicted to, besides coffee, it is the internet.

Sep 23 4.2K 166

Abhinav

Docker Is Dead—And It's About Time

Docker changed the game when it launched in 2013, making containers accessible and turning "Dockerize it" into a developer catchphrase.

Jun 8 6.9K 192

Himanshu Singour

How Shopify Handles 30TB of Data Every Minute with a Monolithic Architecture

If you've ever worked on a web app that slows down with a few thousand users, try imagining one that handles billions.

Oct 9 403 9

In Level Up Coding by Fareed Khan

Building 17 Agentic AI Patterns and Their Role in Large-Scale AI Systems

Ensembling, Meta-Control, ToT, Reflexive, PEV and more

Sep 25 2K 42

Nivetha Thangaraj

Kubernetes Is Dead: Why Tech Giants Are Secretly Moving to These 5 Orchestration Alternatives

By Nivetha Thangaraj

Jun 30 708 90

See more recommendations