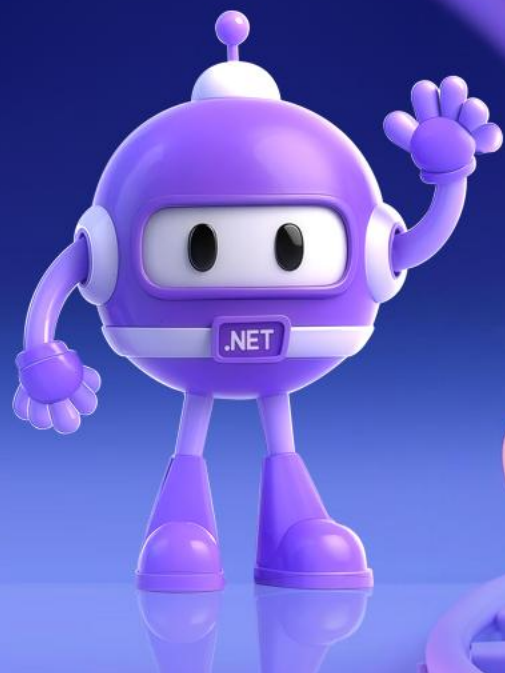


.NET Conf China 2025

改变世界 改变自己

2025 年 11 月 30 日 | 中国 上海





Microsoft®
Most Valuable
Professional

.NET Conf China 2025

改变世界 改变自己



为什么说 Blazor是 AI应用全栈开发利器？

分享人：杨舜杰



has been added. No need to wait for a build.

ant-design-blazor

Edit profile

Sponsors dashboard

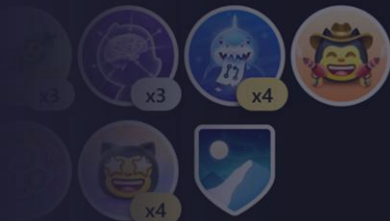
157 followers · 160 following

ant-design-blazor

https://jamesyeung.cn

@chunjiey

Achievements

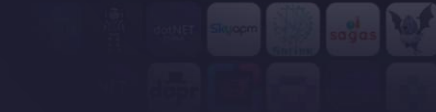


Highlights

Developer Program Member

2021

Organizations



ant-design-blazor/ant-design-blazor · Public

A rich set of enterprise-class UI components based on Ant Design and Blazor.

C# 6.1k 1.2k

SkyAPM/SkyAPM-dotnet · Public

The .NET/.NET Core instrument agent for Apache SkyWalking

C# 1.6k 336

NCaptcha · Public

Captcha in .NET Core

C# 45 4



杨舜杰

James Yeung

微软MVP (AI & DT)

Ant Design Blazor 作者

1,204 contributions in the last year

Contribution settings



@ant-design-blazor @SkyAPM @BeyondDimension More

Activity overview

dotnet/aspnetcore · Public

ASP.NET Core is a cross-platform .NET framework for building modern cloud-based web applications on Windows, Mac, or Linux.

C# 37.4k 10.5k

vue-router-storage · Public

A vue router storage solution. -If your vue application needs to jump to a third party page, and then jump back, want to restore to the original state and continue to operate, the use of this plu...

JavaScript 92 10

CoreProxy · Public

Implement a simple Aop using .Net Core library System.Reflection.DispatchProxy

C# 44 16

2025

2024

2023

2022

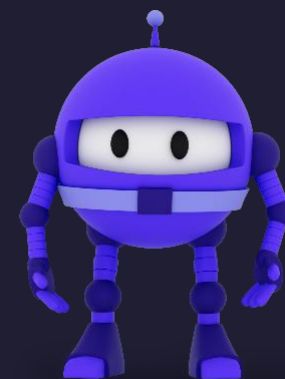
2021

2020

Table of contents

- 01** Blazor 的进化之路 (.NET 8 - 10)
- 02** 传统 AI 应用开发的局限
- 03** Blazor 技术栈的核心优势
- 04** .NET AI 生态与 UI 组件
- 05** 实战案例分享

01 Blazor 的进化之路 (.NET 8 - 10)



Blazor 成为 .NET 主推 Web 技术框架

版本	核心特性
.NET 8	全栈统一：Auto 渲染模式，Static SSR，Identity UI
.NET 9	性能与体验：静态资源优化，运行时检测，重连体验升级
.NET 10	极致体验：Circuit 状态持久化，PersistentState，资源预加载

Blazor in .NET 8

全栈统一的里程碑

Auto 渲染模式

首屏秒开 (Server) + 本地运行 (WASM)

完美解决 WebAssembly 首次加载慢的问题

Full-stack Web UI

支持静态服务端渲染 (Static SSR)

SEO 友好, 适用于官网、博客等内容型页面

Identity UI

一键生成登录、注册、用户管理功能

Blazor in .NET 9

性能优化与体验升级

静态资源传递优化

构建时自动压缩 (gzip/brotli) 与指纹处理
替代 UseStaticFiles, 显著减少传输体积

运行时渲染模式检测

新 API 检测组件运行位置及交互性
简化跨模式组件开发与调试

体验增强

重连优化 断连后自动刷新, 提升 Server 模式体验
Auth 序列化 简化服务端到客户端的身份状态传递

Blazor in .NET 10

极致体验与状态管理

Circuit 状态持久化

彻底解决 Server 模式断连后状态丢失的问题
网络波动重连后，用户输入内容依然保留

[PersistentState]

简化预渲染 (Prerendering) 时的状态持久化
避免页面“闪烁”或数据重复加载

资源预加载

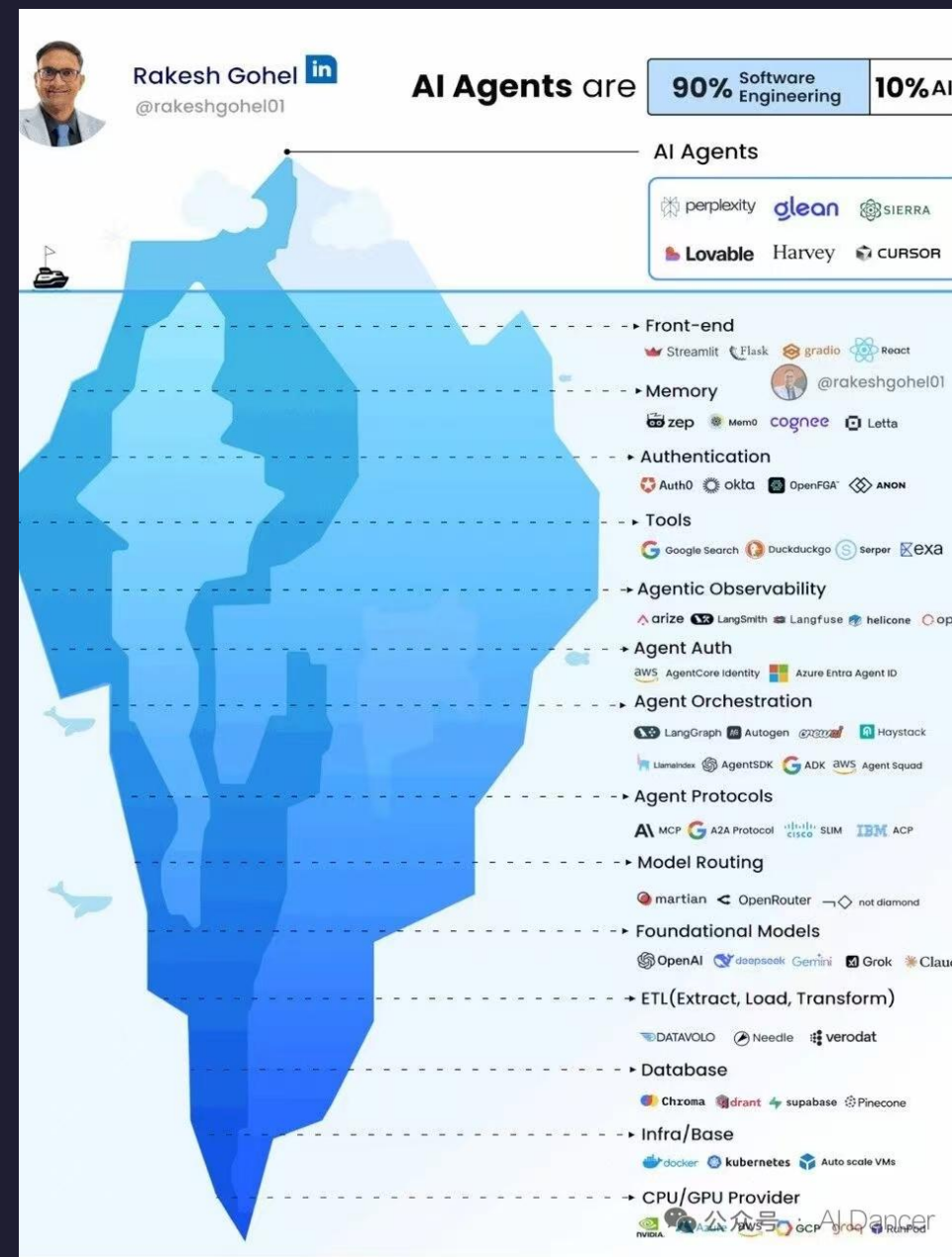
预加载 Blazor 静态资源文件
显著提高 WebAssembly 模式的加载速度

02 AI 应用开发的局限



局限一 技术栈割裂

- 前端 (React/Vue): 需理解后端 API 逻辑, 类型定义重复
- 后端 (Python/Node): 需维护 API 文档, 处理跨域与序列化
- 全栈/独立: 认知负荷过载, 上下文切换频繁



局限二

复杂的交互逻辑与状态维护

- **上下文 (Context) 维护难**：AI 对话依赖连续上下文，HTTP 无状态特性导致每次请求需搬运大量历史数据，或依赖复杂的客户端状态库。
- **逻辑割裂**：Agent 编排（工具调用、多步推理）逻辑分散在前后端，修改业务需同时调整 Python 后端与 JS 前端。
- **异常状态处理**：流式生成中途失败或网络波动时，前端状态回滚与 UI 恢复极其繁琐。

局限二

流式响应实现的“高门槛”

- **流式是标配**：AI 应用必须支持“打字机”效果，但实现健壮的 SSE/WebSocket 通信（鉴权、心跳、重连）耗时耗力。
- **渲染性能挑战**：高频 Token 推送导致前端频繁重绘，渲染复杂 Markdown/代码块时容易造成页面卡顿。
- **调试链路长**：出现乱码或截断时，需排查 Python 生成 -> 网关代理 -> JS 解析整个链路，定位困难。

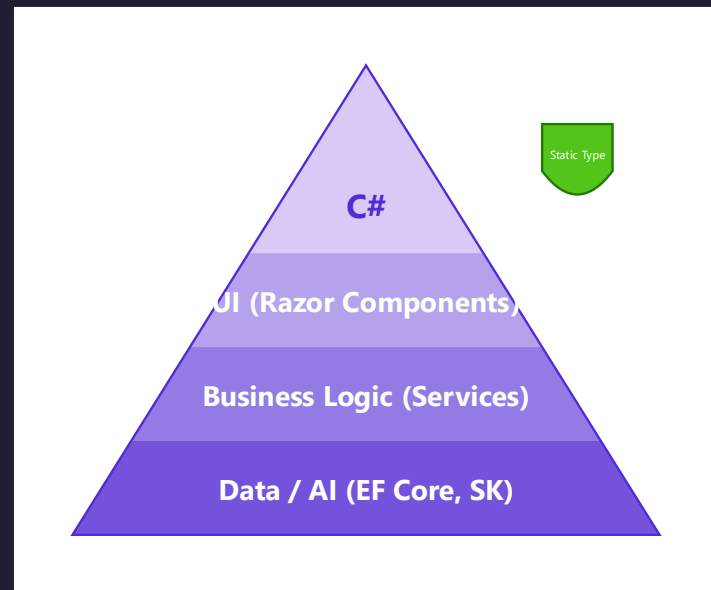
03 Blazor 技术栈的核心优势



优势一

C# 全栈统一与 AI 友好性

- **AI 编程基石**：静态类型约束 —— 让 AI 生成的代码可控、可维护
- **UI 渲染**：C# (Razor Components) —— 类型安全，智能提示
- **业务逻辑**：C# (Services) —— 复用后端逻辑
- **数据/AI**：C# (EF Core / SK) —— 统一模型定义 (Shared Project)

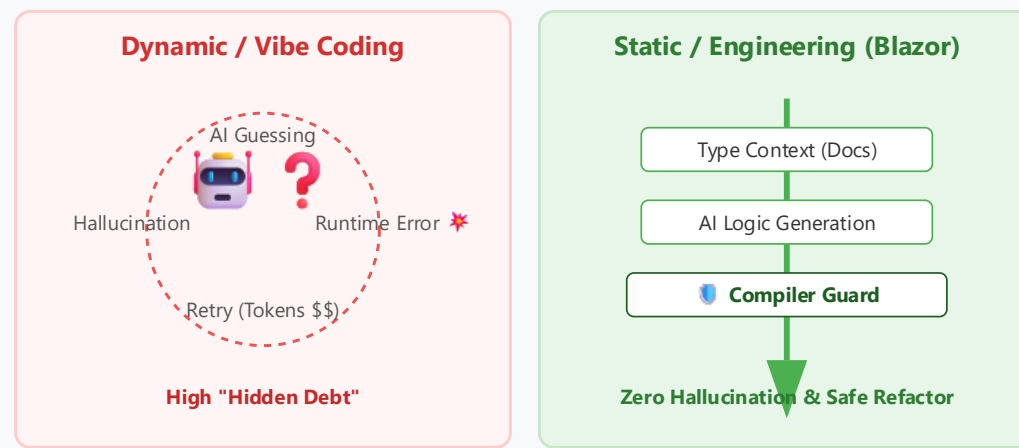


优势二

AI 辅助编程的最佳拍档

- 拒绝幻觉：
 - AI 编造的 API 直接被拦截
 - 编译期识别错误，节省 Token
- 蕴含高质量上下文信息：
 - 类型包含高密度语义信息，减少 AI "瞎猜"
 - 有效帮助 AI 在有限窗口内保持逻辑一致
- 从 Vibe Coding 到工程化：
 - 重构安全，避免 AI 引入"隐性债务"

AI Assisted Programming: Vibe Coding vs. Engineering



优势三

零 API 开销与高效流式渲染

维度	传统 SPA (REST/SSE)	Blazor Server
数据链路	后端 -> JSON -> 网络 -> 前端 JS -> DOM	后端内存 -> 组件状态 -> SignalR Diff -> DOM
流式实现	复杂的 SSE/WebSocket 协议处理	C# <code>await foreach</code> (原生异步流)
网络传输	传输完整业务数据 (JSON 文本)	仅传输 UI 变动 (二进制 DOM Diff)
开发体验	需维护 API 定义与数据同步	像写本地应用一样调用 Service

优势四

有状态架构与原生安全

会话状态常驻

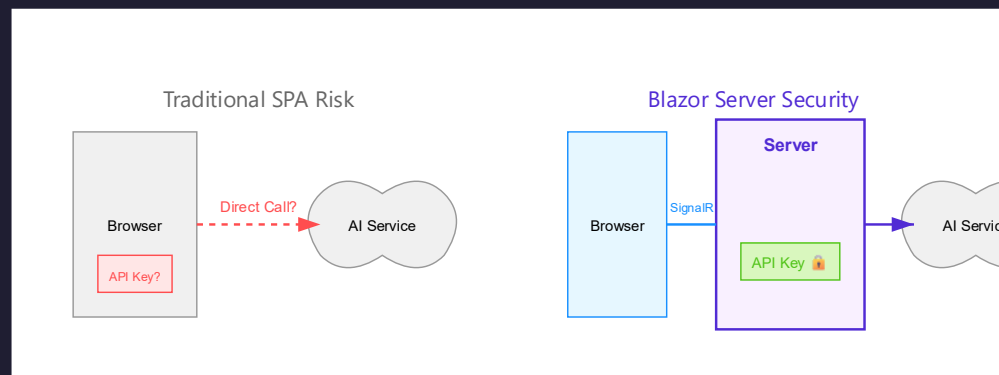
内存级上下文：对话历史 (History) 驻留内存，无需每次请求读写数据库。

极简多轮对话：可像写 Console App 一样通过 List<Message> 维护状态。

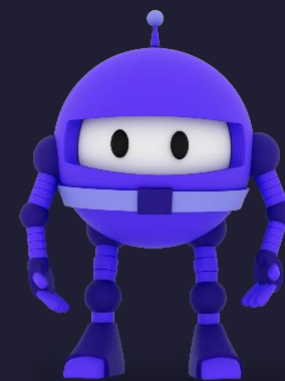
原生安全

API Key 仅在服务端使用，永不发送至客户端浏览器。

零 BFF 成本：无需搭建额外后端代理层 (Proxy) 来隐藏敏感配置。



04.NET AI 生态与 UI 组件



庞大的 .NET AI 生态概览

Microsoft Agent Framework

最新推出，用于构建、编排和部署人工智能代理及多代理工作流的框架。

Microsoft.Extensions.AI

为生成式人工智能组件提供一种统一的表示方法，并实现与各种人工智能服务的无缝集成和互操作性。

Microsoft.Extensions.VectorData

提供一个统一的C#抽象层，用于与向量存储进行交互。

Semantic Kernel

一个与模型无关的软件开发工具包（SDK），它使开发人员能够构建、编排和部署人工智能代理以及多代理系统。无论你是要构建一个简单的聊天机器人，还是一个复杂的多代理工作流，Semantic Kernel 都能为你提供所需的工具，并且具备企业级的可靠性和灵活性。

官方SDK

OpenAI SDK
MCP SDK
A2A SDK

ML.NET

可以使用现有的 .NET 技能轻松地将 ML 集成到 .NET 应用中，而无需任何以前的 ML 体验。

AI 编程工具链

Visual Studio
Visual Studio Code
Github Copilot
C# dev kit

社区项目

BotSharp
LLamaSharp
AntSK
AntDesign X Blazor

Azure AI Foundry

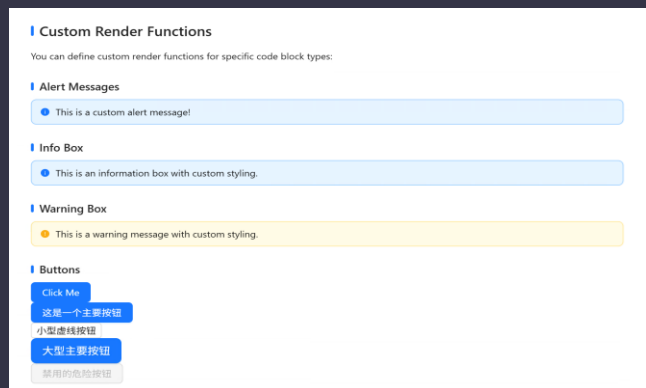
使用来自 Microsoft、OpenAI、Anthropic、Cohere、DeepSeek、Mistral AI、Meta 等的热门模型加速创新。

AntDesign X Blazor

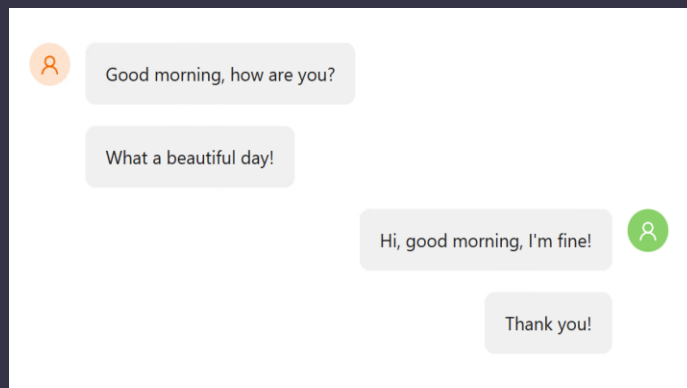
UI 组件库

基于 Ant Design RICH 设计范式，深度整合微软全栈 AI 生态，为全球企业及开发者打造卓越 AI 界面解决方案，引领智能新体验。

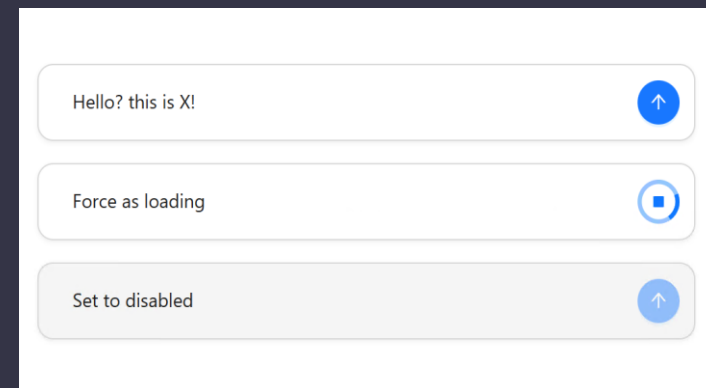
Markdown 自定义组件渲染



对话气泡



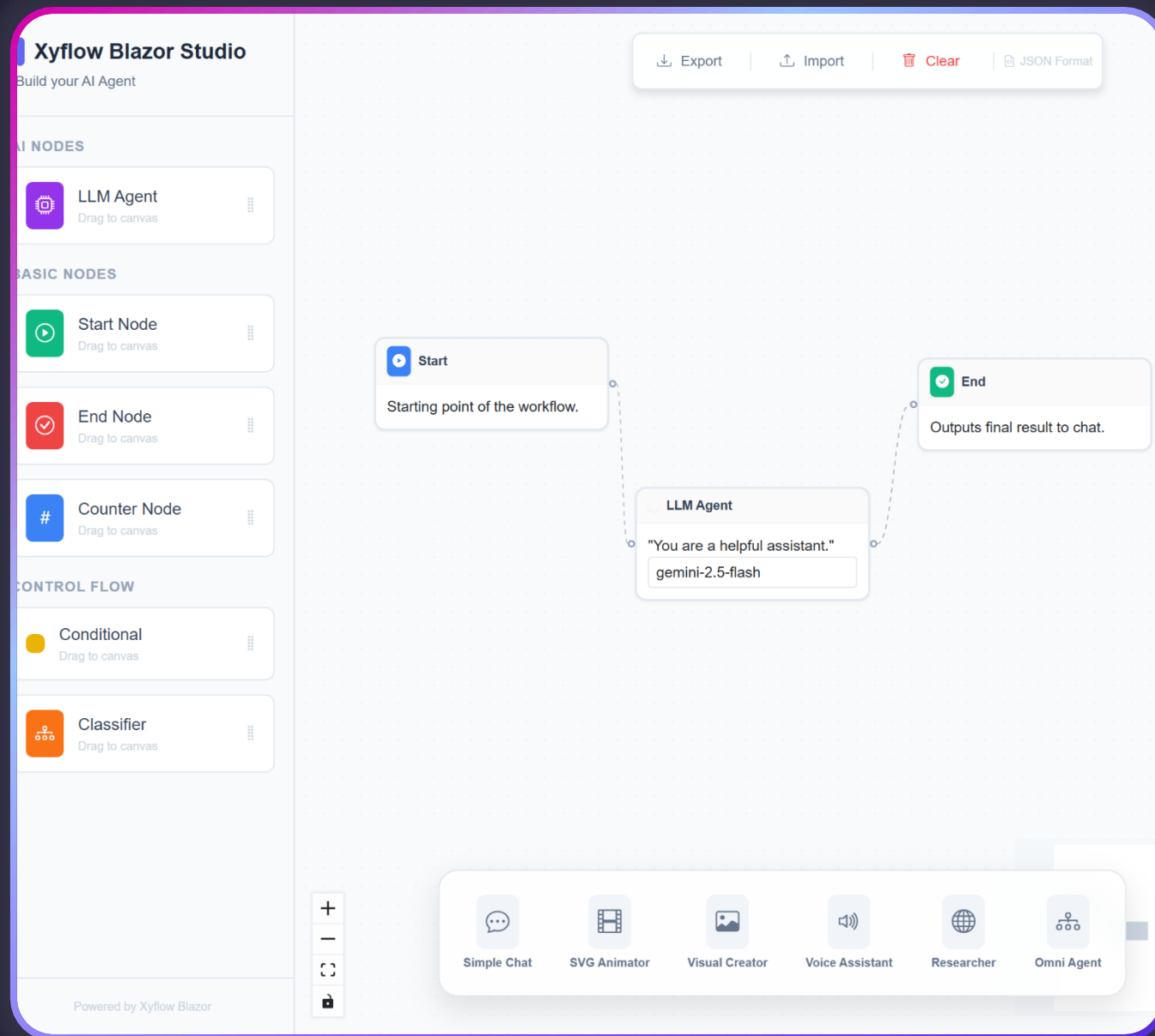
输入框



Xyflow Blazor

新坑

利用 Blazor 封装 ReactFlow，提供一致的用于构建基于节点的编辑器和交互式图表、工作流设计器，并实现工作流引擎。



05 实战案例分享



产品主页和登录页

使用Blazor Static SSR

综合加载速度、SEO、轻量交互



批改交互 使用Blazor Server

批改结果人工修改等重交互场景

三月三十日至四月五日

我這個星期過得十分有意義，令我獲益良多，在我的小學生活中留下美好的回憶。

星期一，老師宣佈這次全方位學習日的地點——對手農舍去參加歷奇考驗活動。同時也宣佈了分組，我跟小俞算同一組。一知道名單後，我們便分配工作而我被分配到去摘菜。

當天，我們一群人浩浩蕩蕩地到達目的地。映入眼簾是一片鬱鬱蔥蔥的草地，還帶着淡淡香味，令人精神為之一振。我們做的第一件時的事就是製作木筷子，然後便開始烹飪了。我最喜歡就是去摘香草了，因為香草的草香我十分喜歡，也能做調味料。負責人說明可摘的香草後，便讓我們親自嘗試，我也摘了很多不同的草。但我的組把所有都塞進魚的肚子裏面。烤完取出來，肚子也被脹爆了，讓大家哭笑不得。

星期六，我看著那天拍的照片，發現我們快畢業了。希望之後大家也要繼續聯絡，當一輩子的朋友。

正所謂天下無不散之筵席，這次到對手農舍的經歷，將在我小學生活留下美好的回憶。

評語列表

+ 新增評語

☐ 顯示已刪除的評語

我這個星期過得十分有意義，令我獲益良多，在我的小學生活中留下美好的回憶。

你已掌握開頭點題技巧，若加入具體事例會更生動！

一知道名單後，我們便分配工作而我被分配到去摘菜。

『而』用得不太順，試試改為『然後』？會更通順喔！

當天，我們一群人浩浩蕩蕩地到達目的地，映入眼簾是一片鬱鬱蔥蔥的草地，還帶着淡淡香味，令人精神為之一振...

『鬱鬱蔥蔥』『精神為之一振』描寫太棒了，畫面感十足！

我們做的第一件時的事就是製作木筷子，然後便開始烹飪了。

『時』是錯別字，應為『事』，記得以後多檢查喔！

我最喜歡就是去摘香草了，因為香草的草香我十分喜歡，也能做調味料。

你很喜歡摘香草呢！試試用不同詞形容香味吧？

但我的組把所有都塞進魚的肚子裏面，

.NET Conf China 2025

改变世界 改变自己



THANK YOU

