AI笔试题

🧩 任务目标

实现一个简化版实时投票系统，满足以下功能：

✅ 基础功能要求

1. 投票问卷
    - 系统预置 1 份问卷，包含 1 道单选题（3~5 个选项 ）。
    - 所有用户可投票，每人一次（不需用户登录，简单前端限制即可）。
2. 数据统计
    - 投票后，系统统计每个选项的当前票数。
    - 所有正在查看页面的用户应实时收到统计结果更新（使用 WebSocket 或 Server-Sent Events 方案之一）。
3. 前端界面
    - 使用 Vue3 或 React 实现：
        - 显示问卷题目和选项；
        - 用户选择并提交投票；
        - 实时展示投票结果（如柱状图、数字）。
4. 后端服务
    - 提供 RESTful API 实现投票和获取当前问卷数据；
    - 支持 WebSocket 或 SSE 接口用于实时推送统计变化。

🧪 技术要求

- 前端：Vue3（可使用 Vite）或 React（nextjs），可选 Chart.js/ECharts 实现图表；
- 后端：Python（FastAPI/Django）或 Go（Gin/Fiber 等）；
- 数据库：使用任意你熟悉的 MySQL/PostgreSQL/MongoDB 之一；
- 通信协议：实现 WebSocket 或 Server-Sent Events，用于实时统计推送；
- 部署与运行：本地可运行即可，不要求线上部署，提供运行说明和 Dockerfile 或 k8s 部署文件；

🔌 API 设计要求（示例）

接口	方法	功能
`/api/poll`	GET	获取当前问卷及实时投票统计
`/api/poll/vote`	POST	提交投票（传入选项 ID）
`/ws/poll` 或 `/sse/poll`	WebSocket / SSE	订阅实时投票结果更新

📄 提交内容

请将以下内容整理并打包为一个压缩包：

1. 前端代码（**Vue3** 或 **React**）
    - 页面展示投票问卷与结果图表；
    - 投票交互与实时结果刷新逻辑；
2. 后端代码
    - API 实现及 WebSocket/SSE 实时推送逻辑；
    - 数据存储与票数更新逻辑；
3. 数据库初始化脚本或说明
4. 项目运行说明
    - 安装依赖、运行步骤、默认端口等；
    - 可选：提供 Dockerfile 和 `docker-compose.yml`（加分项）；
5. 技术文档
    - 简要系统架构图（可手绘/Markdown 描述）；
    - API 接口说明（格式、参数、响应示例）；
    - 实时推送机制说明；
    - 技术选型简要说明；
6. 演示截图/录：展示主要功能实现和实时数据交互效果。
    - 打开两个浏览器窗口，在屏幕中左右两侧各占一半空间，点击左边窗口投票，右边窗口可以看到投票结果变化；点击右边窗口投票，左边窗口也可看到投票变化

💡 加分项

- 使用 TypeScript；
- 提供基础单元测试；
- 使用 Docker + docker-compose 构建运行环境；
- 提供简洁、清晰的代码结构和注释；
- UI 简洁美观，图表动态更新流畅；

🕑 时间限制

建议完成时间：**1** 个工作日（约 **3～4** 小时）

📬 提交方式说明（请务必认真阅读）

为确保您提交的压缩包文件顺利送达，请根据以下方式提交：

✅ 方式一：上传网盘 + 回复分享链接（推荐）

1. 将压缩包上传至网盘（如百度网盘、腾讯微云、阿里云盘、奶牛快传等）；
2. 设置分享链接为“公开可访问”（如有提取码请一并提供）；
3. 将链接和提取码发送至邮箱：`hr@89trillion.com`

示例回复内容：
提交链接：https://xxx.com/xxxxx
提取码：abcd

✅ 方式二：直接发送压缩包附件（请勿使用 **QQ** 邮箱）

1. 使用 Gmail / Outlook 等邮箱发送邮件；
2. 将压缩包作为附件发送，建议设置密码（密码可写在正文中）；

3. 请勿使用 QQ 邮箱发送压缩包附件，容易被 Gmail 系统拦截，导致我们无法收到。

⚠️ 注意事项

- 压缩包中请勿包含 `.exe` / `.bat` 等可执行文件；
- 如包含此类文件，请务必设置密码压缩，并备注密码；
- 请勿转发系统邮件（`.eml` 附件），该格式易被系统拦截；
- 若遇问题，请及时招聘平台HR或联系：**hr@89trillion.com**