MeetingGuru（会开会）项目报告书

项目概述

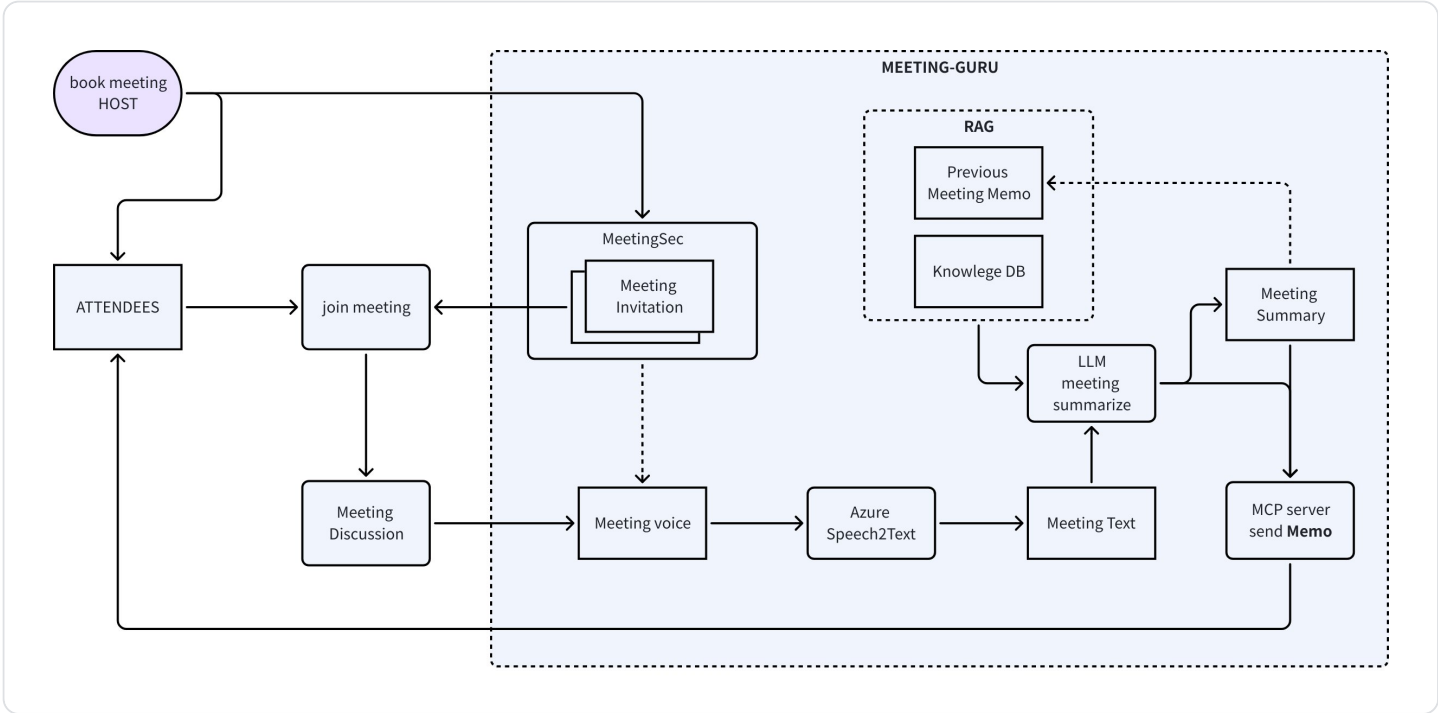
会议成功与否，会议的总结及后续的推进是一个重要方面。日常工作中我们经常会深有感触，如果一次会议没有形成清晰的会议结论并推送给相关责任人，则会议很可能会流于形式，成果难以固化。从另一方面来说，传统的会议总结及推送是一个费时费力、容易出错的工作，同时有一定的门槛。对于开发岗位的绝大多数同事，一般不善于做会议总结并推进会议成果。

基于上面两点现状，一个智能的会议总结与推送应用能够极大的简化会议后的推进工作，将研发同学从会议总结中解放出来。我们的应用名称为**会开会**(**MeetingGuru**)，应用提供自动化会议分析、总结、推送功能，能够高效提炼会议内容，生成会议纪要及待办事项，并将纪要和待办事项发送给参会人员及关联人员。让会议更高效，会议成果更易落地。

团队介绍

我们的团队名称为**2025共商AGI**，团队共有三人：杨瑞健（队长）、苏丽伟、孙飞。三人全部为开发工程师。我们三人目前均就职于商汤科技，队名中的**商**字有呼应商汤之意。我们使用即梦生成了我们团队的队徽如下，队徽包含我们的队名以及较多的AI元素，同时有几个人形剪影，强调了团队合作：

应用整体流程



DEMO视频

【会开会应用演示-20250525-哔哩哔哩】

项目源码

下载链接：https://github.com/user-attachments/files/20430707/MeetingGuru-src-20250525.zip

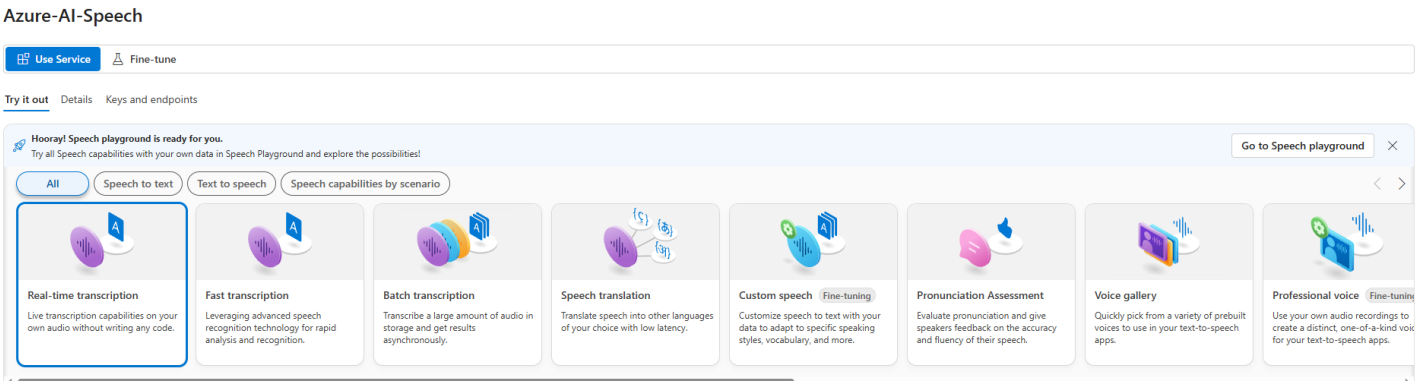
MeetingGuru-src-20250525.zip
11.80MB

关键模块介绍

语音转文字

一般会议的主要输入有两个：1. 参会人员的发言（语音）；2. 参会人员展示的PPT或者其他屏幕分享、板书手写内容等（图像/视频）。本项目因为开发时间原因，目前只接入了会议语音作为输入，整个项目需要先将语音转为文字，再走后续的流程。语音转文字选择了微软Azure-AI-Speech服务，该服务提供多种语言识别方案：流式、快速转译等，同时提供了一定的免费额度供学习和测试。

ai.azure.com



azure语音识别服务接入非常方便，如下一段简单的代码即可实现调用azure语音识别服务并得到识别之后的文字结果：

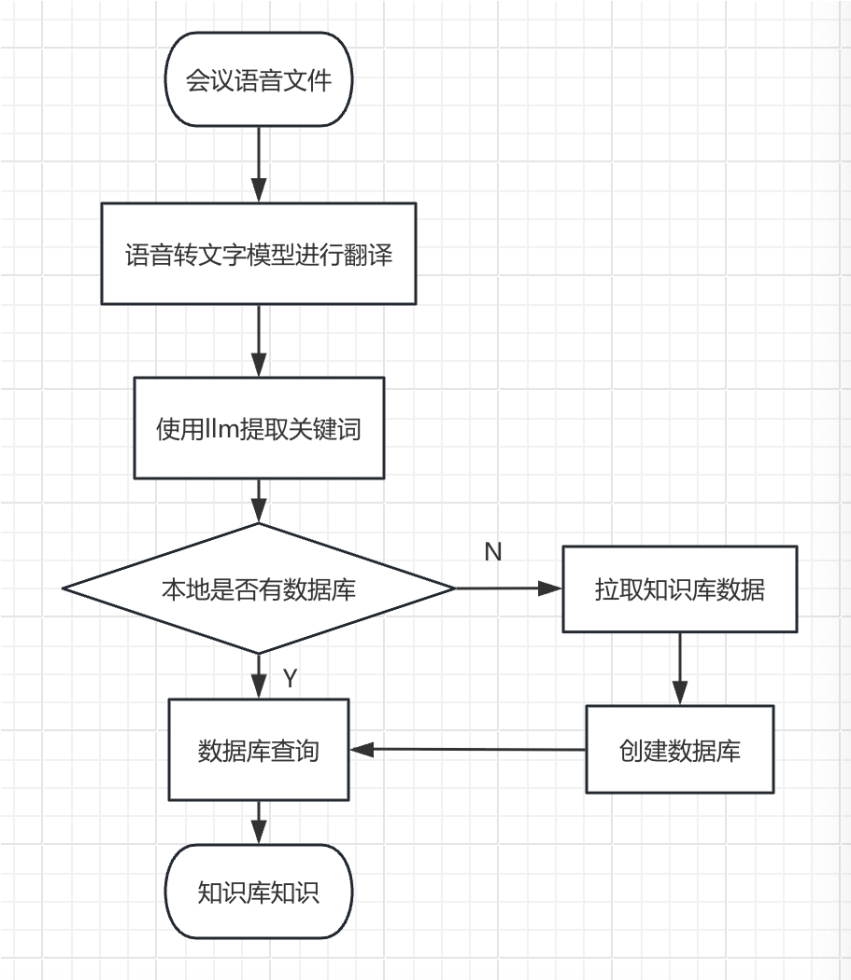
代码块

```
def transcribe_audio():
    try:
        # 创建语音配置对象

        # 创建音频配置（支持 WAV、MP3、OGG 等格式）
        audio_config = AudioConfig(filename=audio_file_path)
        speech_config.speech_recognition_language = "zh-CN"  # 中文普通话

        # 初始化识别器
        recognizer = SpeechRecognizer(speech_config=speech_config,
audio_config=audio_config)

        print("正在识别音频...")

        # 执行语音识别
        result = recognizer.recognize_once_async().get()

        # 处理识别结果
        if result.reason == ResultReason.RecognizedSpeech:
            print(f"识别结果: {result.text}")
        elif result.reason == ResultReason.NoMatch:
            print("未识别到语音")
        elif result.reason == ResultReason.Canceled:
            cancellation_details = result.cancellation_details
            print(f"识别取消: {cancellation_details.reason}")
            if cancellation_details.reason == CancellationReason.Error:
                print(f"错误详情: {cancellation_details.error_details}")

    except Exception as e:
        print(f"发生异常: {str(e)}")
```

RAG

RAG功能主要用于基于行业技术知识，给技术会议提供相关背景知识。我们希望大模型可以基于领域特定知识库对会议中的内容或者问题进行解答或者解释，以便大家可以更深层次的理解会议的内容。

RAG实现的主要流程如下：



主要流程代码如下：

代码块

```
contextStore = VecStore(db_path="docstore_index")
if (not os.path.exists(contextStore.db_path)):

contextStore.loadDocumentsFromURL(["https://docs.unity3d.com/cn/2019.4/Manual/InverseKinematics.html"])

contextStore.loadDocumentsFromURL(["https://docs.unity3d.com/cn/2019.4/Manual/TargetMatching.html"])

contextStore.loadDocumentsFromURL(["https://docs.unity3d.com/cn/2019.4/Manual/RootMotion.html"])

context_prompt = ChatPromptTemplate.from_template(
```

```
        "请阅读下面的会议记录内容，从中提取出恰好 5 个最能概括主题的中文关键词。"
        "会议记录：{speech_text}。\n\n"
        "**要求**:\n"
        "关键词应为 名词或短语，避免长句。\n"
        "按相关性从高到低排序，不得重复。\n"
        "只输出关键词本身，不附加任何解释或空格。 "
    )

    # 3. 创建LangChain处理链
    # prompt = PromptTemplate.from_template(template2)
    llm = ChatNVIDIA(
        model="meta/llama-3.3-70b-instruct",  # 对应NIM部署的模型名称
        temperature=0.3,
        max_tokens=1024,
        nvidia_api_key=API_KEY,
        # base_url=f"{NIM_ENDPOINT}/completions"  # NIM API端点
    )

    context_chain = context_prompt | llm | StrOutputParser() |
    contextStore.generate()
    context = context_chain.invoke({"speech_text" : speech_text})
```

MCP框架

MCP Server 是基于 FastMCP 框架构建的服务，提供了多种功能工具，这些工具可以被客户端应用调用。这种架构设计采用了微服务理念，将不同功能模块化，便于维护和扩展。

FastMCP 是一个专为工具功能封装和暴露设计的框架，它提供了一种机制，使得:

- 工具函数可以被注册为API端点
- 函数的参数和返回类型得到严格检查
- 函数文档可以作为API描述被自动收集

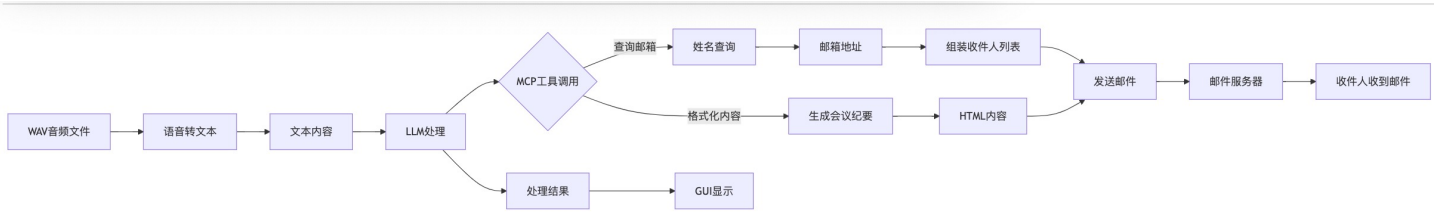
在MeetingGuru系统中，这些MCP工具被用于:

1. 处理会议音频后识别出提到的人名
2. 查询这些人名对应的邮箱地址
3. 将生成的会议纪要发送给相关人员

代码块

```
@mcp.tool()
def send_email(to_mailes: list[str], subject: str, html_body: str) -> bool:
    """Send an email"""
    FROM_EMAIL = os.getenv('FROM_EMAIL')
    SMTP_SERVER = os.getenv('SMTP_SERVER')
```

```
        SMTP_PORT = int(os.getenv('SMTP_PORT', 465))
        SMTP_USER = os.getenv('SMTP_USER')
        SMTP_PASSWORD = os.getenv('SMTP_PASSWORD')
        # ...validation and sending logic
        return send_to(subject, html_body, to_mailes, FROM_EMAIL, SMTP_SERVER,
    int(SMTP_PORT), SMTP_USER, SMTP_PASSWORD)
```



UI

MeetingGuru 的界面层采用了基于 Flet 框架的现代化 UI 实现，结合了多种技术来确保跨平台文件操作能力。整体设计采用了视图切换型的单页应用结构，通过组件化设计提高了代码复用性和可维护性。

MeetingGuru 中GUI相关部分均为AI生成

1. 首页

2. 点击离线会议进入下个页面选择音频

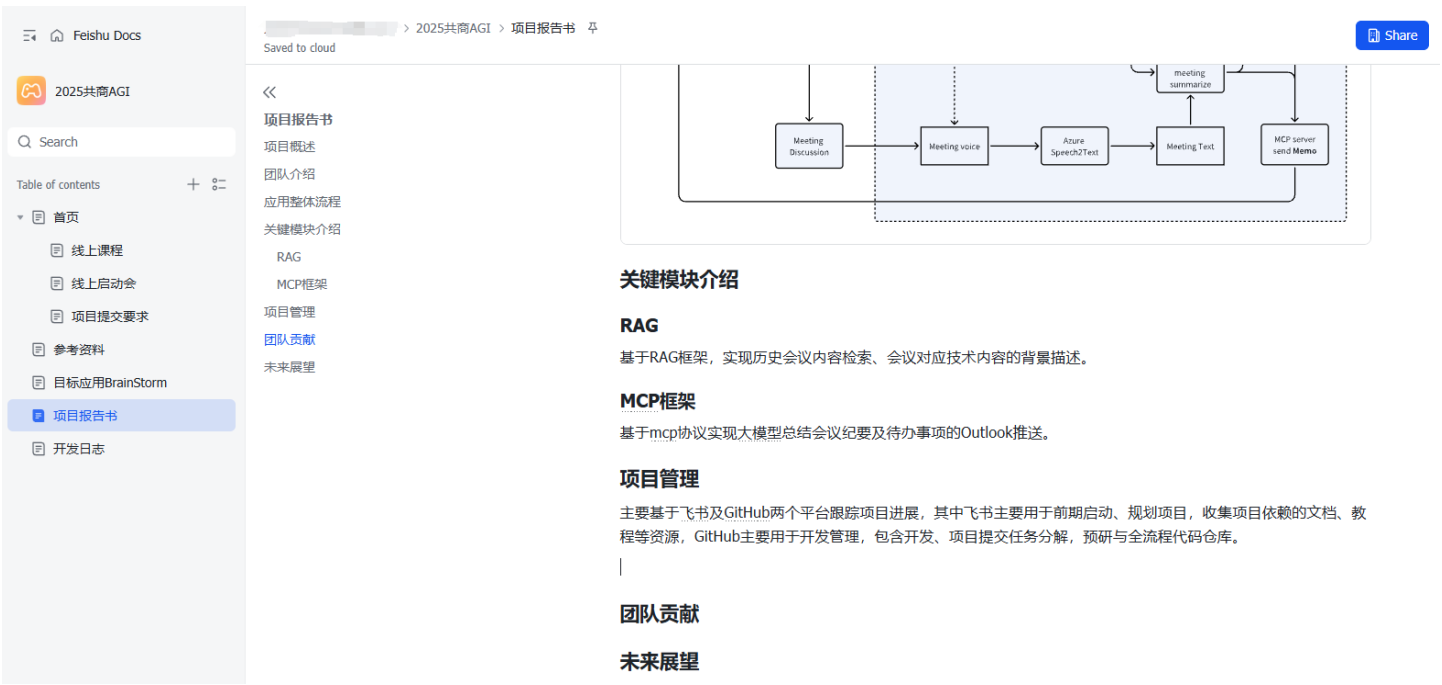
3. 音频选择完后点击开始会议总结

4. 总结完成,llm 通过MCP查询参会人员邮箱，统一发送

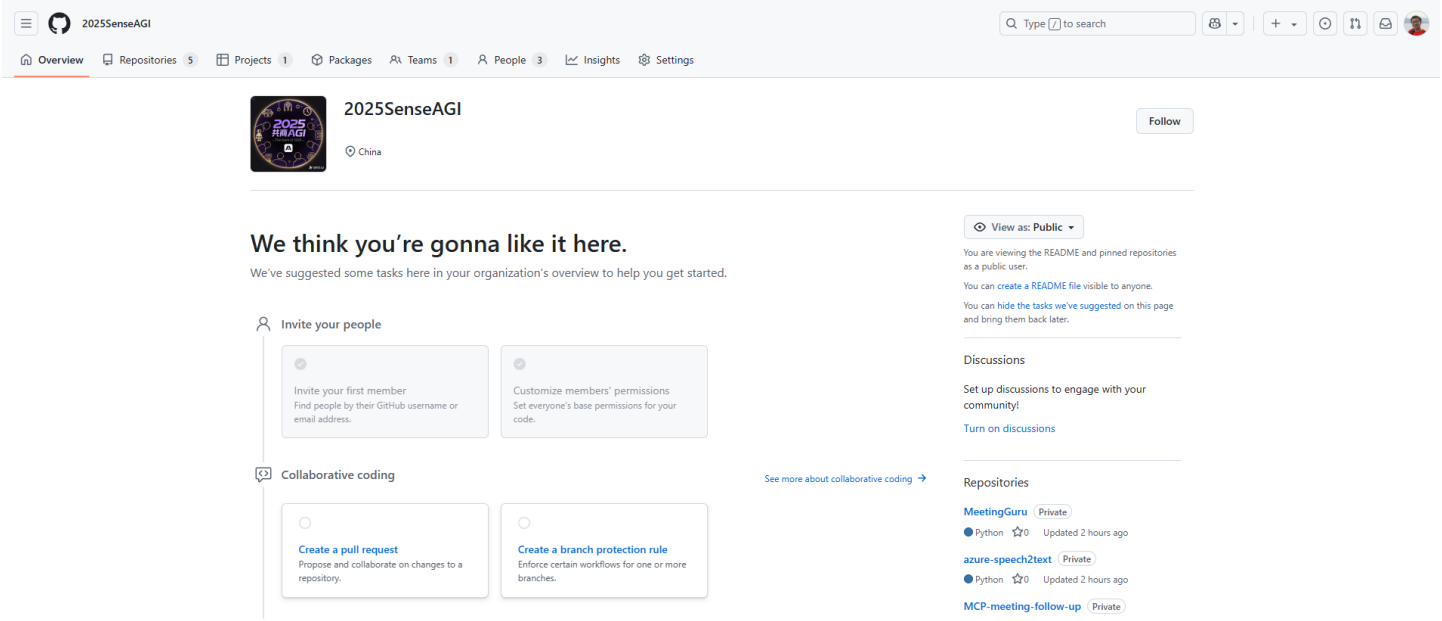
项目管理

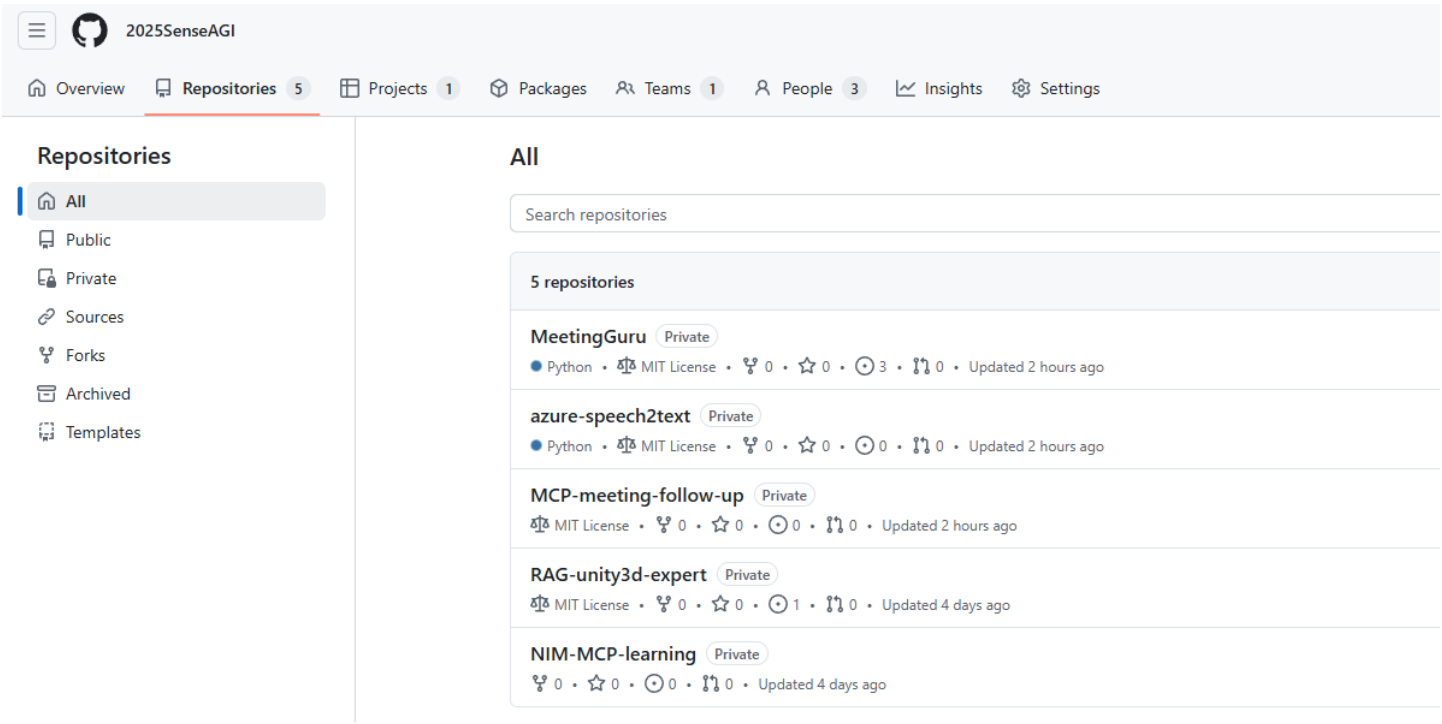
主要基于飞书及GitHub两个平台跟踪项目进展，其中飞书主要用于前期启动、规划项目，收集项目依赖的文档、教程等资源，GitHub主要用于开发管理，包含开发、项目提交任务分解，预研与全流程代码仓库。

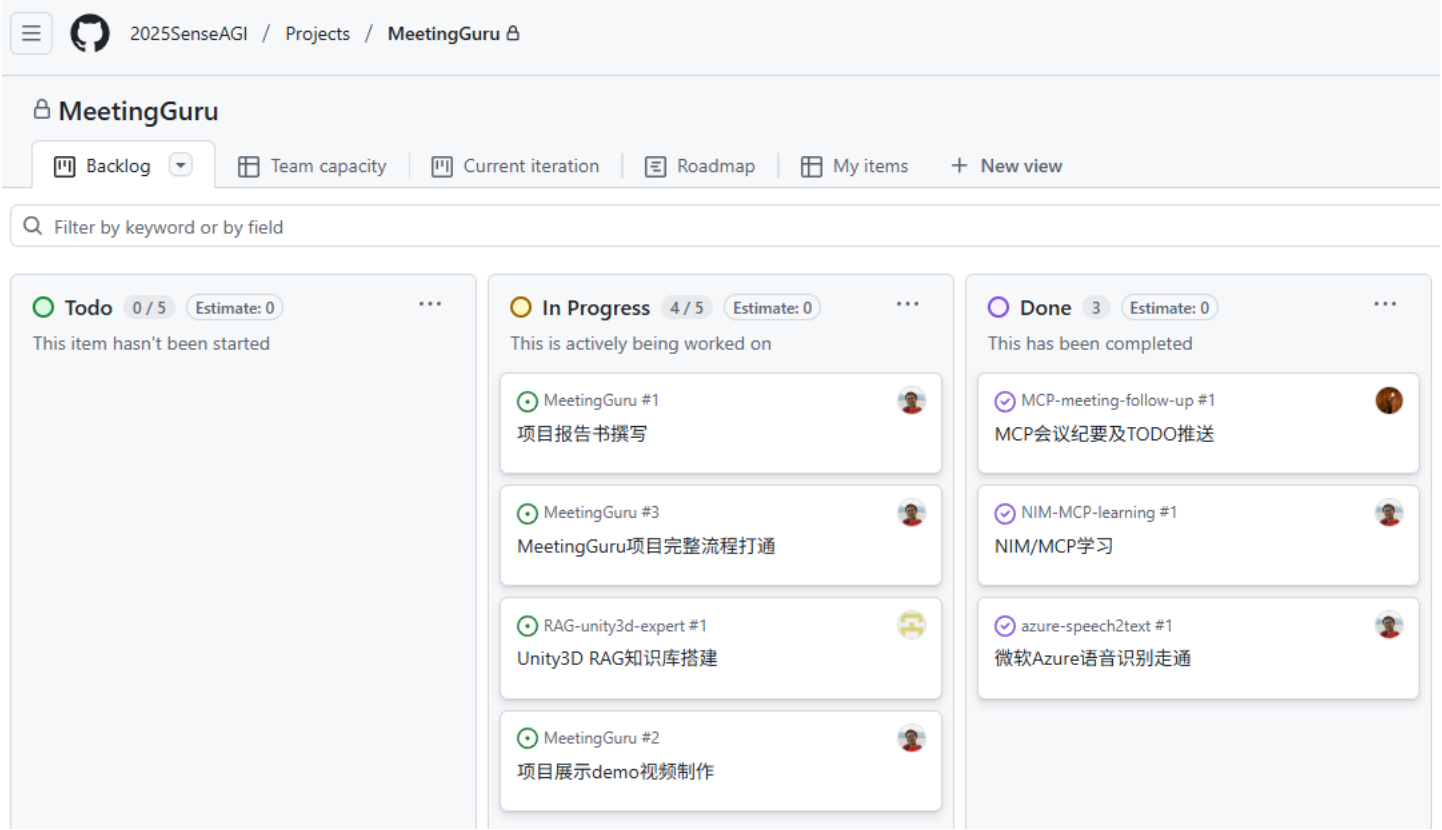
飞书



GitHub


2025SenseAGI
Type / to search
Overview
Repositories 5
Projects 1
Packages
Teams 1
People 3
Insights
Settings
2025SenseAGI
China
Follow
We think you're gonna like it here.
We've suggested some tasks here in your organization's overview to help you get started.
Invite your people
Invite your first member
Find people by their GitHub username or email address.
Customize members' permissions
Set everyone's base permissions for your code.
Collaborative coding
See more about collaborative coding
Create a pull request
Propose and collaborate on changes to a repository.
Create a branch protection rule
Enforce certain workflows for one or more branches.
View as: Public
You are viewing the README and pinned repositories as a public user.
You can create a README file visible to anyone.
You can hide the tasks we've suggested on this page and bring them back later.
Discussions
Set up discussions to engage with your community!
Turn on discussions
Repositories
MeetingGuru Private
Python 0 Updated 2 hours ago
azure-speech2text Private
Python 0 Updated 2 hours ago
MCP-meeting-follow-up Private


2025SenseAGI
Overview
Repositories 5
Projects 1
Packages
Teams 1
People 3
Insights
Settings
Repositories
All
Public
Private
Sources
Forks
Archived
Templates
All
Search repositories
5 repositories
MeetingGuru Private
Python • MIT License • 0 • 0 • 3 • 0 • Updated 2 hours ago
azure-speech2text Private
Python • MIT License • 0 • 0 • 0 • 0 • Updated 2 hours ago
MCP-meeting-follow-up Private
MIT License • 0 • 0 • 0 • 0 • Updated 2 hours ago
RAG-unity3d-expert Private
MIT License • 0 • 0 • 1 • 0 • Updated 4 days ago
NIM-MCP-learning Private
0 • 0 • 0 • 0 • Updated 4 days ago



团队贡献

整个项目是团队通力合作的结果，是团队的统一产出。几乎项目的每个地方都是团队合作的结果。细分一下，团队大致分工如下：

杨瑞健：

1. 队长，整体项目规划、协作平台搭建、推进，主要项目节点把控；
2. Azure语音识别服务走通；
3. NIM LLM会议总结template prompt设计及LLM会议总结调优；
4. 项目报告书撰写；
5. 项目DEMO视频制作；

孙飞：

1. RAG学习及搭建；
2. RAG模块接入到MeetingGuru整体pipeline；

苏丽伟：

1. MCP学习及搭建；
2. MCP模块接入到MeetingGuru整体pipeline；
3. UI交互界面搭建；

未来展望

目前市面上的一些常用会议软件，如腾讯会议、小鱼易连、Zoom等已经提供了类似的会议内容总结、待办事项推送等功能。我们的产品在市场现有龙头上的一个主要改进是增加RAG输入，RAG主要包含两块内容：1. 行业技术知识，能够给技术会议提供相关背景知识；2. 历史会议纪要，通过历史会议纪要将长期的研发工作形成闭环，而不是一个个孤立的会议。基于上面两点，我们的应用能够提供**更有知识含量**，**更闭环**的会议总结。

因为时间原因，我们的应用也有许多尚未完成的工作，主要有如下几点：

1. 图像/视频相关会议内容的输入。会议一般还会有视觉相关内容的输入，这一部分也提供了丰富的信息。可以基于这一部分的属于及目前比较成熟的多模态LLM推理，形成更完整的会议纪要；
2. 会议系统/软件接入。将我们的应用（会议机器人）接入到目前常用的会议软件中，实现自动接收会议邀请，自动接入会议，收集会议信息，会议结束后自动总结并推送这样一个完整的闭环；
3. RAG、LLM prompt等调优，进一步优化会议总结的准确度与质量；
4. 一个包含UI界面的demo，包含会邀接收、自动开启录制、会议完成自动总结并推送、更新历史会议纪要RAG库等完整流程的调起。目前完成的工作需要手工录制会议音频文件给python脚本，并通过python脚本实现后续的整条流程；