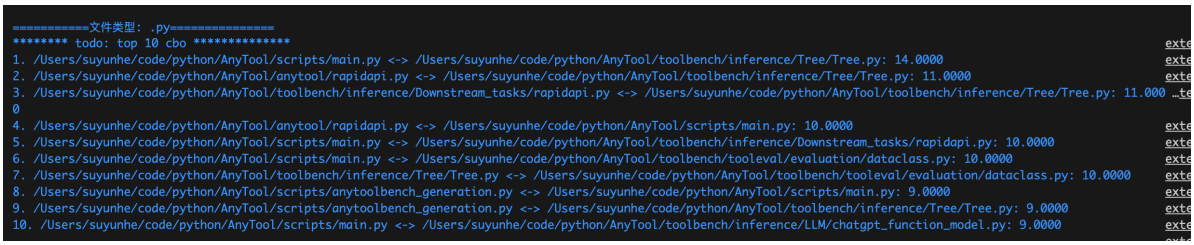
插件开发

在前端添加针对 `python` 和 `java` 的 CBO 计算、单文件的 RFC 计算。

> CBO（Coupling Between Objects）是面向对象编程中衡量类之间耦合程度的一个指标，为两个类 A 和 B 之间相互访问的成员数量之和，即 $|E_{A\to B}|+|E_{B\to A}|$。CBO值越高，说明两个类之间的耦合度越高。
>
> RFC（Response For a Class）是一个软件度量指标，用于衡量一个类的响应能力。它表示一个类的方法数量加上该类调用的外部方法数量。RFC 值越高，通常意味着类的复杂度越高，因为它需要响应更多的外部调用。



这部分功能使用 `python` 程序实现，首次将 `python` 程序集成到的插件中，之后会将更多的特定功能模型集成到插件，这样能获取到实时数据，有利于获得更准确和即时的结果。

- 考虑计算语义相似度

模型设计

初步编写**行为总结**功能的原型

输入：日志文件（日志条目序列）

功能：

1. 统计日志数据中各类型操作的数量
2. 可视化文件的变化（文件的创建和删除）
3. 可视化文本操作涉及的所有工件
4. 提取日志数据关键内容，调用大模型总结日记序列中开发者进行的操作

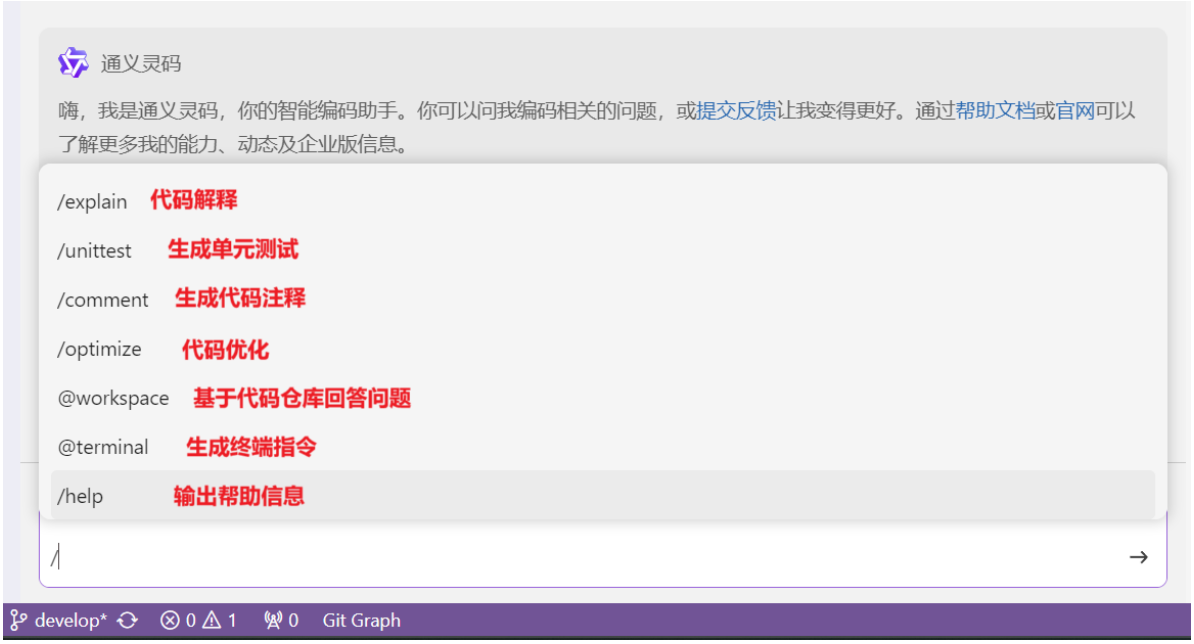
问题：

1. 还没做完，只处理了文件类操作和文本类操作，终端类操作和指令执行操作还没处理
2. 提取关键内容的方案还比较初级，有信息丢失，而且对内容的压缩还不够

`VirtualMe` 研发问答QA设计

`@VirtualMe` 提供若干固定的问题选项，可以在 Chat 中提问，也可以在内部通过 API 调用。

通义灵码的 QA 设计：



原始方案：

只提供三个问题：行为总结、操作件预测、能力分析

拿到问题后进行分解，输出全部可以获取的信息。例如行为总结包含了代码变更摘要、终端命令摘要、编辑强度区域、包管理类型等…，操作件预测包含了识别出的模式下新操作件预测结果+旧操作件检索结果…

新方案：

每个模块下可以细分更多子问题，上层智能体根据实际情况选择提问对应的内容。

模块1：行为总结

这部分可能没有涉及需要训练的模型。

需求：如何提供接口

问题/API	输出
代码变更的摘要，可以指定文件/文件夹范围	范围内涉及的文件的编辑历史总结，通过代码快照+LLM实现
终端命令的摘要，可以指定命令类型	对应类型的命令的执行历史总结，成功与否，控制台输出内容总结，通过匹配+LLM实现
开发区域可视化	一张repo的树形图，节点精确到artifact或file，节点颜色或属性表明编辑强度或其他编辑细节
获取宏观总结	与原方案一致，先模式识别，根据模式，总结对应的内容。

模块2：操作件预测

这部分的模型是模式识别和预测检索模型

需求：对标 Cursor，行级预测

方案：1.基于基础模型微调 2.从零开始训练

问题/API	输出
操作过的工件中，哪些最可能被下一步操作	工件路径+行号范围
未操作过的工件中，哪些与当前开发最相关	工件路径+行号范围
获取宏观预测	与原方案一致，先模式识别+新旧件预测，再检索+计算，输出工件路径+行号范围

模块3：能力分析

这部分尚未形成详细的指标清单和计算方案，可能将能力指标分类，按类别提供问题选项。

- 生产率（指标：编码效率、Debug效率）
- 方法：论文、GitHub

问题	输出
我的编码能力	与该能力相关的指标，提供可视化图表，下同
我的项目熟悉程度	
我解决问题的能力	
对我有什么建议