

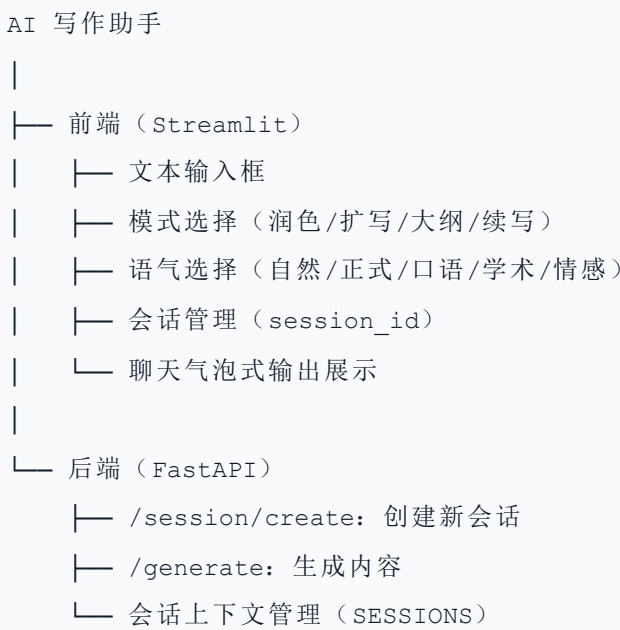
# 写作生成器 PRD（产品需求文档）

## 1. 产品背景与目标

随着内容创作需求不断增长，用户希望拥有一个 轻量、高效、可本地运行的写作助手。  
本项目基于：

- FastAPI 后端：提供 AI 写作能力（润色/扩写/续写/大纲）
- Streamlit 前端：提供可视化操作界面，支持会话管理、结果展示
- 目标：构建一个简单、易用、本地可运行的 AI 写作助手，帮助用户提升文本质量与创作效率。

## 2. 整体功能结构



## 3. 功能详情

### 3.1 会话管理

功能	描述
创建新会话	后端生成 <code>UUID</code> ，用于区分对话上下文

功能	描述
保存历史消息	SESSIONS 字典记录 prompt 历史
持续对话	前后文关联，生成更连贯的内容

接口：

后端：POST /session/create

前端：启动时自动请求一次，存入 st.session\_state.session\_id

### 3.2 文本生成（核心功能）

前端发送参数：

字段	类型	说明
session_id	str	用于定位对话历史
text	str	用户输入文本
mode	str	revise / expand / outline / continue
tone	str	自然、正式、口语、学术、情感

后端行为：

mode	行为描述
revise	润色表达，使语言更自然清晰
expand	扩写内容，丰富细节
outline	生成结构化大纲
continue	续写下文

## 前端 UI 交互流程 (Streamlit)

### 页面组成

- 左侧参数栏
  - 模式下拉框
  - 语气下拉框
  - 当前会话 ID
- 右侧写作区
  - 输入文本框
  - “生成”按钮
  - 聊天气泡展示历史内容

### 用户流程

1. 打开页面自动创建新的 session\_id
2. 用户选择模式 + 语气
3. 输入文本点击“生成”
4. 前端发送 POST 请求到 `/generate``
5. 后端返回 AI 生成结果
6. 前端追加到气泡消息区域展示

---

## 5.后端 API 设计 (FastAPI)

### 5.1 POST /session/create

用于创建独立会话。

响应：

```
{  
  "session_id": "xxxx-xxxx-xxxx"  
}
```

## 5.2 POST /generate

输入文本 → 输出 AI 生成内容

请求示例：

```
{  
  "session_id": "xxx",  
  "text": "原文内容.....",  
  "mode": "revise",  
  "tone": "自然、清晰"  
}
```

响应示例：

```
{  
  "output": "AI 生成后的结果....."  
}
```

---

## 6. 模型能力（基于 Qwen API）

系统 Prompt：

- 不编造信息
- 输出简洁自然
- 根据 mode 不同行为
- 语气由 `tone` 控制

模型：`qwen-plus`

---

## 7. 系统架构图

Browser

|

▼

Streamlit 前端 (文件6)

|

HTTP/JSON

▼

FastAPI 后端 (文件5)

|

调用

▼

Qwen API (云端)

## 8. 错误处理

前端需处理：

- 无文本输入 → 禁止提交
- 后端返回 500 → 弹出错误提示
- 网络错误提醒

后端需处理：

- 空文本
- Qwen API 失败
- session\_id 不存在

## 9. 运行方式

后端

```
uvicorn main:app --reload
```

前端

```
streamlit run app.py
```

## 10. 后续可扩展功能

- 多会话管理（可切换）
  - Markdown 导出
  - Prompt 模板管理
  - 本地 LLM 支持（如 Ollama）
  - 文本差异对比（Diff）
-