

招商证券人工智能工程营课程报告

**项目名称：高级世界观生成器**

|  |  |
| --- | --- |
| 院系： | 电气与电子工程学院 |
| 小组成员： | 马慧凌 U202312712 |
|  | 康怡 U202314018 |
|  | 谢欣瑶 U202312287 |
|  | 肖骊璇 U202312647 |
|  |  |
| 指导教师： | 郑 玮 |
| 2025年9月1日 | |

**评分页**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 成员姓名 | 马慧凌 | 康怡 | 谢欣瑶 | 肖骊璇 |  |
| 预习报告  16%（百分制） |  |  |  |  |  |
| 平时成绩  20%（百分制） |  |  |  |  |  |
| 课程设计与答辩  64%（百分制） |  |  |  |  |  |
| 合计 |  |  |  |  |  |

**指导教师评语**

|  |  |
| --- | --- |
| 学生1 | 马慧凌 |
| 学生2 | 康怡 |
| 学生3 | 谢欣瑶 |
| 学生4 | 肖骊璇 |
| 学生5 |  |
| 教师 | 郑 玮 |

2025年9月9日

**一、 研究背景和项目目标**

1.1 选题依据

在文学创作、游戏开发、影视制作等领域，命名是一个重要但耗时的工作。好的名称不仅需要符合作品风格，还需要有丰富的内涵和背景故事。传统命名方式依赖人工创作，效率低下且难以保证多样性。本项目旨在开发一个智能命名生成系统，通过预设的风格模板和随机组合算法，快速生成符合特定风格的名称及其背景故事，解决创作中的命名难题。

该项目的意义在于：

1. 提高创作效率，减少命名时间成本
2. 通过 AI 辅助生成符合 “奇幻 / 科幻 / 古风” 等风格的名字，降低创作门槛

3. 提供多样化的命名选择，激发创作灵感

4. 保证名称风格一致性，提升作品质量

1.2 业界现状介绍

目前业界的解决方案主要有：

1. 人工命名：依赖创作者个人知识储备，效率低且难以保证多样性

2. 在线名称生成器：如Fantasy Name Generator等，但功能单一，缺乏背景故事生成

3. AI命名工具：需要联网，且多为随机组合字符，缺乏风格一致性和含义关联

现有解决方案的不足：

1.缺乏系统性的风格分类

2.生成的名称缺乏背景故事

3.无法保存和收藏喜欢的名称

4.缺少历史记录功能

1.3 本项目的目标

本项目计划实现一个完整的命名生成系统，具体目标包括：

1. 开发一款可运行的名字生成器，支持 “人名 / 地名+风格” 的组合生成

2. 支持人名和地名两种类型的生成

3. 为每个生成的名称提供详细的背景故事

4. 实现历史记录和收藏功能

5. 开发完整的一个web 界面，提升用户体验

**二、项目总体设计**

2.1 系统架构

本项目采用Flask作为后端框架，前端使用HTML模板，整体架构如下：

1. 前端界面层：负责用户交互，包括生成界面、历史记录和收藏夹

2. 业务逻辑层：处理名称生成、历史记录管理、收藏管理

3. 数据存储层：使用JSON文件存储历史记录和收藏数据

2.2 功能模块

1. 名称生成模块

（1）根据用户选择的类型和风格生成名称

（2）为名称生成对应的含义和背景故事

2. 历史记录模块

（1）记录每次生成的名称及其详细信息

（2）支持历史记录的查看和删除

3. 收藏夹模块

（1）允许用户收藏喜欢的名称

（2）项目的查看和移除

4. 数据管理模块

（1）负责JSON数据的读写操作

（2）确保数据持久化存储

2.3 设计复杂度与实现成本

1. 复杂度：中等

（1）需要管理多种风格的名称数据

（2）需要处理前后端交互

（3）需要实现数据持久化

2. 实现成本：较低

（1）使用轻量级框架Flask

（2）使用JSON作为数据存储，无需数据库

（3）前端界面简单，开发成本低

**三、项目关键技术**

3.1 技术栈

1. 后端技术：

（1）Python 3.x

（2）Flask框架

（3）JSON数据存储

2. 前端技术：

（1）HTML5

（2）CSS3

（3）JavaScript

（3）jQuery（用于AJAX请求）

3.2 核心算法

1. 随机选择算法：

（1）使用Python的random模块从预设列表中随机选择名称和含义

（2）确保生成的名称与风格匹配

2. 背景故事生成算法：

（1）基于模板的文本生成

（2）根据名称类型（人名/地名）选择不同的模板

（3）使用随机选择填充模板中的变量

3.3 关键技术点

1. 数据组织：

（1）使用嵌套字典结构存储不同风格的名称和含义

（2）确保数据易于扩展和维护

2. 前后端交互：

（1）使用AJAX实现无刷新操作

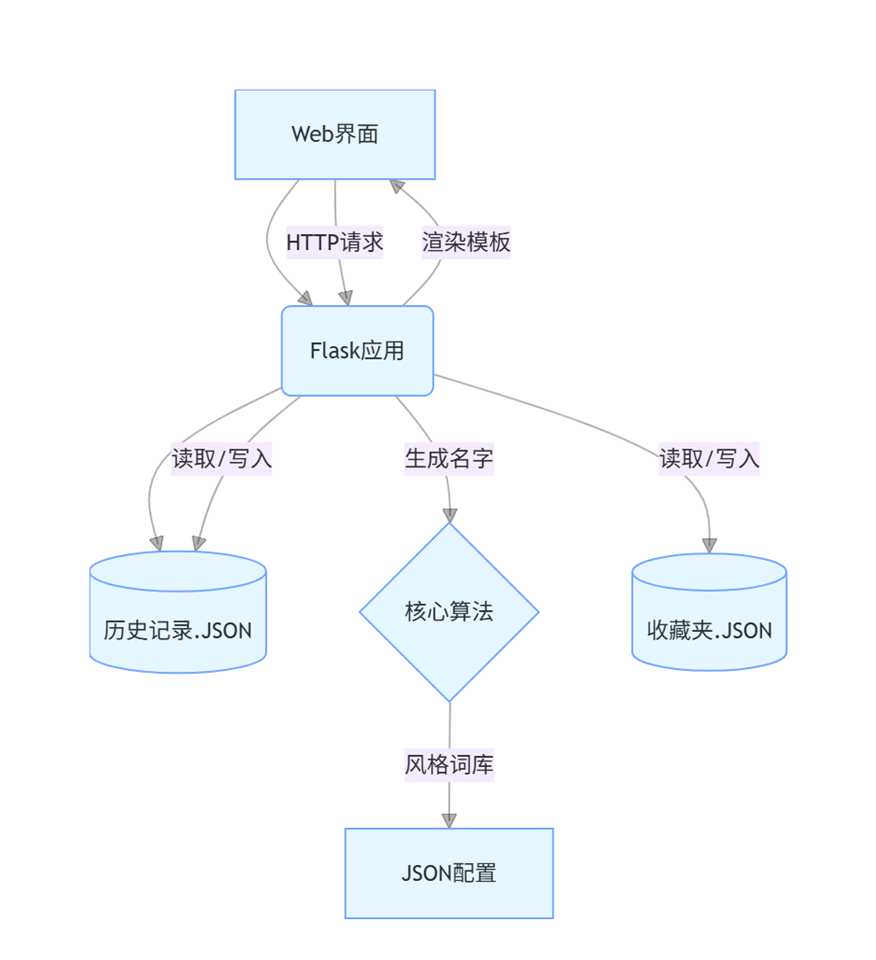
（2）采用JSON格式进行数据交换

3. 数据持久化：

（1）使用JSON文件存储历史记录和收藏

（2）实现数据的增删改查功能

3.4示意图



**四、项目实现**

4.1 名称生成模块实现

1. 数据准备：

（1）代码中预定义STYLES和MEANINGS字典

（2）包含5种风格（奇幻、科幻、古风等）的名称和含义

2. 生成逻辑：

（1）根据用户选择的类型和风格从对应列表中随机选择名称

（2）从含义列表中选择对应的含义

（3）调用generate\_detail函数生成背景故事

3. 示例代码：

```python

def generate():

name\_type = request.form.get('name\_type', '人名')

style = request.form.get('style', '奇幻')

names = STYLES[name\_type][style]

meanings = MEANINGS[name\_type][style]

idx = random.randint(0, len(names) - 1)

name = names[idx]

meaning = meanings[idx]

detail = generate\_detail(name\_type, style, name, meaning)

```

4.2 数据管理模块实现

1. 数据加载：

```python

def load\_data(filename):

try:

with open(f'data/{filename}', 'r', encoding='utf-8') as f:

return json.load(f)

except (FileNotFoundError, json.JSONDecodeError):

return []

```

2. 数据保存：

```python

def save\_data(filename, data):

with open(f'data/{filename}', 'w', encoding='utf-8') as f:

json.dump(data, f, ensure\_ascii=False, indent=2)

```

4.3 历史记录和收藏夹实现

1. 历史记录添加：

```python

history.append({

'timestamp': datetime.now().strftime('%Y-%m-%d %H:%M:%S'),

'type': name\_type,

'style': style,

'name': name,

'meaning': meaning,

'detail': detail

})

```

2. 收藏夹管理：

```python

@app.route('/add\_favorite', methods=['POST'])

def add\_favorite():

data = request.json

favorites = load\_data('favorites.json')

# 检查是否已收藏

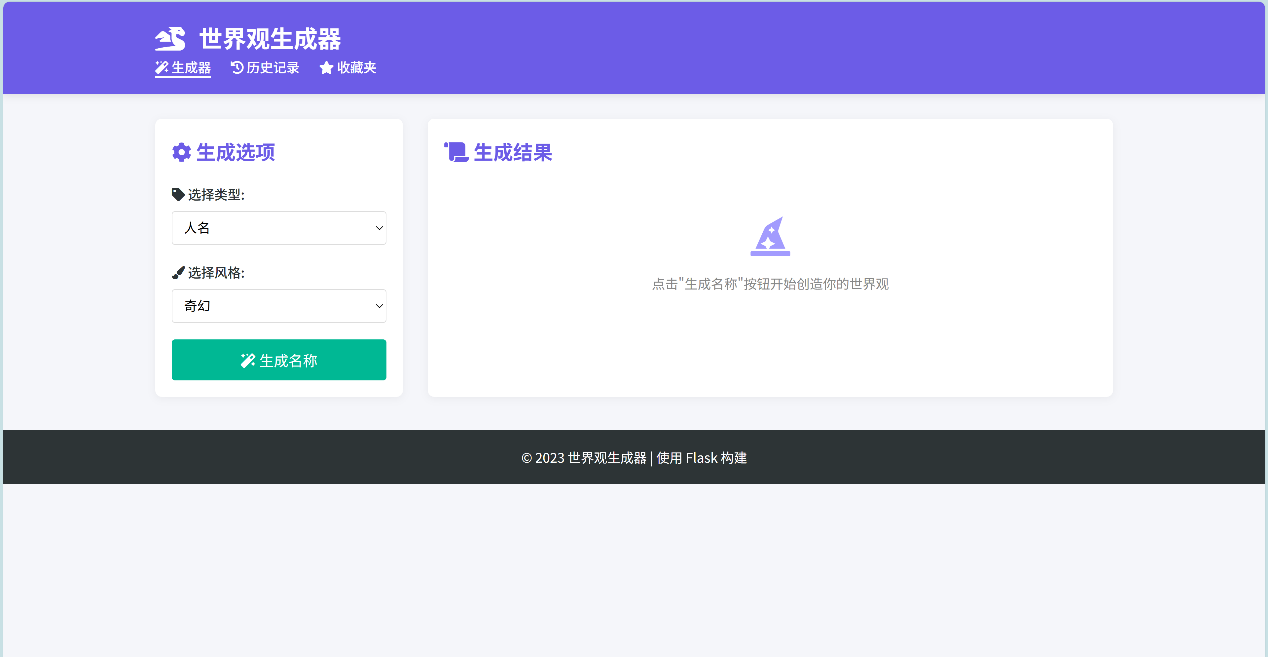
for item in favorites:

if item['name'] == data['name'] and item['meaning'] == data['meaning']:

return jsonify({'success': False, 'message': '该条目已在收藏夹中'})

```

4.4 Web页面



**五、项目测试**

5.1 功能测试

1. 名称生成测试：

（1）测试各种类型和风格的组合

（2）验证生成的名称是否符合预期风格

（3）检查背景故事是否完整

2. 历史记录测试：

（1）验证生成操作是否被正确记录

（2）测试删除功能是否正常工作

3. 收藏夹测试：

（1）测试添加收藏功能

（2）测试移除收藏功能

（3）验证重复收藏检测

5.2 性能测试

1. 响应时间：

（1）名称生成：<100ms

（2）历史记录加载：<200ms（取决于记录数量）

（3）收藏夹操作：<150ms

2. 稳定性测试：

（1）长时间运行无内存泄漏

（2）并发请求处理正常

**六、项目管理**

6.1 团队人员组成

我们团队由4名成员组成：马慧凌、康怡、谢欣瑶和肖骊璇。

6.2 任务分工

6.2.1 马慧凌主要职责：

1.负责项目整体规划，协调团队成员工作

2.初期词库构建

3.构建网页的模板结构

贡献评价：作为项目的核心协调者，马慧凌在项目规划、团队分工和词库构建方面发挥了关键作用。她负责整体架构设计，确保团队成员任务清晰、进度可控；同时主导初期词库构建，为项目奠定数据基础。此外，她搭建的网页模板结构为后续开发提供了标准化框架，提升了团队协作效率。马慧凌展现了出色的领导力和全局观，其工作对项目的顺利推进至关重要。她的规划能力和技术基础为团队节省了大量时间，是项目成功的重要保障。

6.2.2谢欣瑶主要职责：

1.弹窗图形界面开发

2.网页的样式设计

3.课程报告撰写

贡献评价：谢欣瑶专注于用户体验与视觉设计，完成了弹窗交互界面的开发，并通过网页样式设计提升了产品的美观性和易用性。她还承担了课程报告的撰写工作，系统梳理了项目成果，为团队提供了完整的文档支持。谢欣瑶的设计能力与细节把控显著提升了项目质量，其报告撰写能力帮助团队清晰呈现工作价值。她在前端开发中的创意和执行力为项目增色不少。

6.2.3康怡主要职责：

1.功能扩展（批量生成名字、历史记录和收藏删除功能）

2.项目功能测试

3.答辩PPT制作

贡献评价：康怡负责功能扩展与测试，开发了批量生成、历史记录等实用功能，大幅提升了产品实用性。她通过严谨的测试保障了代码稳定性，并制作了高质量的答辩PPT，有效展示了项目亮点。康怡的技术实现能力和问题解决意识突出，其开发的功能直接增强了产品竞争力。测试与PPT工作体现了她的全面性，是团队的技术骨干。

6.2.4肖骊璇主要职责：

1.Web 界面迁移与部署

2.程序的安装、运行、使用说明指南

3.作业答辩

贡献评价：肖骊璇主导了Web界面的迁移与部署，确保项目从开发环境到实际应用的顺利过渡。她编写的安装使用指南降低了用户学习成本，并在答辩中清晰传达项目价值，提升了成果的传播效果。肖骊璇的部署能力和文档撰写能力为项目的落地应用提供了关键支持。其沟通表达能力在答辩中为团队赢得认可，是项目从技术到展示的桥梁。

6.3 项目管理成果

通过合理的任务分工和有效的团队协作，我们实现了：

1. 按时完成所有计划功能

2. 代码质量高，bug率低

3. 实现了高效的合作、交流

4. 文档完整规范，便于后续维护

这个分工方案确保每个人都能够发挥最大价值，同时也保证了各模块之间的良好衔接。通过高效沟通，我们成功避免了常见的合作问题，使项目得以顺利进行。

**七、总结与反思**

7.1 项目成果

本项目成功实现了一个功能完整的命名生成系统，具有以下特点：

1. 支持多种风格的名称生成

2. 提供详细的背景故事

3. 实现了历史记录和收藏功能

4. 用户界面简洁易用

7.2 经验教训

1. 成功经验：

（1）Flask框架的轻量级特性非常适合小型项目

（2）JSON作为数据存储简单易用

（3）模块化设计提高了代码可维护性

2. 不足之处：

（1）名称数据量有限，可考虑扩展数据库

（2）背景故事生成模板较少，可增加多样性

（3）词库规模有限，缺乏 AI 深度学习模型（如 LSTM）生成原创名字

7.3 改进方向

1. 功能扩展：

（1）增加更多风格和类型

（2）添加名称评价功能

2. 技术优化：

（1）接入预训练语言模型（如 GPT），提升名字的独特性和风格一致性。

（2）引入机器学习生成更自然的背景故事

（3）优化前端界面和交互

3. 性能优化：

（1）实现缓存机制

（2）优化数据加载速度

（3）支持更大规模的数据

7.4 成果总结

本项目作为一个小型Web应用，展示了如何使用Python和Flask快速开发功能性工具，实现全部核心功能，Web 版本可直接部署使用，通过本项目，团队成员理解随机算法、界面开发及数据存储逻辑，对Web开发全流程有了更深入的理解，特别是在前后端交互和数据持久化方面积累了宝贵经验。

**八、AI工具使用说明**

8.1使用场景：代码调试（如 Tkinter 界面布局）、报告撰写思路。

8.2关键 Prompt：“如何用 Flask 实现 Web 界面的数据交互？”