

学号:

2024303053

# 武汉理工大学

## 《企业级应用软件开发与开发》 课程大作业

学 院

计算机与人工智能学院

专 业

软件工程

班 级

专硕 2405

姓 名

胡珊

指导教师

戚欣

2025 年 6 月 15 日

选题说明	<p>题目：基于 Spring AI 和 Langchain4j 的旅游行程规划智能体</p>
作业内容	<p>一、 简要叙述设计思想和技术路线（不少于 300 字）（20 分）。</p> <p>二、 详细介绍分析与设计（30 分）。</p> <p>请尽量运用 UML 建模，少量文字说明，一图胜千言。用例图、类图、包图、活动图、部署图、ER 图等模型。做到流程清晰，设计规范，编程规约。</p> <p>三、 谈谈这个系统升级扩展的设想。（不少于 300 字）（10 分）。</p> <p>四、 成果截图（20 分）。</p> <p>IDE 中 project 窗口截图；最重要的配置文件和 Java 代码截图；Postman 测试截图；JMeter 测试截图。少量文字说明，图文并茂。</p> <p>五、 课程总结（20 分）。</p> <p>个人收获。对课程的意见和建议。</p> <p>(关于格式：整个作业必须用小四字体，不得改变封面及题目页的格式，必须有封面及题目页，必须完整填写封面页信息。</p> <p>关于纪律：不得抄袭，不得雷同。否则，按零分处理。</p> <p>关于提交时间：大作业压缩包必须 2025 年 6 月 22 日 23 点前线上提交，逾期不予受理。大作业文档需打印后集中存档。</p> <p>打印版集中在 6 月 23 日交给各个班的学习委员 (***)。</p> <p>提交至邮箱 <a href="mailto:8854655@qq.com">8854655@qq.com</a></p> <p>邮件标题：企业级应用软件设计与开发_1049731901953_李工大</p> <p>附件：一个压缩文件</p> <p>压缩文件命名规则：学号_姓名.zip，例：1049731901952_李工大.zip</p> <p>压缩文件中须包含：</p> <ol style="list-style-type: none"><li>1. 【强制】大作业文档，为便于评阅请确保文档的“导航窗格”可以正常使用。</li><li>2. 【强制】README.md，含所用集成开发环境，基础设施搭建方法 等。</li><li>3. 【强制】项目文件夹，确保按照 README.md 可在 IntelliJ IDEA 中正常启动及使用。</li><li>4. 【可选】其他文件夹</li></ol> <p>备注：</p> <ol style="list-style-type: none"><li>1. 功能小而美，技术有亮点。</li><li>2. 加分项：融合 AI；云服务器部署；代码共享至 git 托管平台；课堂分享。</li></ol>

# 基于 Spring AI 和 Langchain4j 的旅游行程规划智能体

软件工程 专硕 2405 2024303053 胡姗

## 〇、快速体验

- 代码仓库地址: <https://github.com/kmoonn/AI-Tourism-Assistant-Agent>
- 线上预览地址: <https://tourism.kmoon.fun>

## 一、设计思想与技术路线

### 1.1 设计思想

本项目旨在构建一个基于 Spring AI 和 LangChain4j 的旅游行程规划智能体,能够通过自然语言理解用户需求,自动生成个性化的旅游计划。整体设计思路以模块化、可扩展性和实用性为核心原则,结合当下热门潮流的 AI 技术与企业级 Java 开发技术栈。

### 1.2 技术路线

在技术路线方面,系统采用 Spring Boot 框架作为后端服务基础,利用其强大的依赖注入和模块组织能力,保障项目结构清晰与易于维护。通过集成 Spring AI 实现对大语言模型的调用,使用 LangChain4j 封装提示模板、管理上下文,提升自然语言交互体验。

系统前端采用 Vue 3 进行开发,采用 Element-plus UI 组件库,让用户可以通过简单的 Web 页面进行交互。核心功能模块包括:用户需求识别模块、旅游行程规划模块、工具调用模块(集成高德地图 MCP 服务、自建和风天气 MCP 服务等),以及响应生成与展示模块。

此外,借助 MCP(Model Context Protocol)协议与工具能力,系统可扩展集成如高德地图、和风天气、小红书笔记推荐等服务,极大提升智能体生成旅游计划的能力与准确性。

最终系统部署至腾讯云轻量服务器,配置 DNS 域名解析,形成从用户输入意图到自动输出旅游日程的一条龙服务流程,体现 AI Agent 在实际应用中的可行性与先进性。

## 二、分析与设计

系统的核心功能是根据用户的需求规划个性化的旅游行程。用户通过对话接口输入目的地、天数、偏好等信息,系统通过调用多种服务(如天气查询、地图查询等)生成最佳的旅游行程。系统应支持用户进行定制化查询和推荐。在设计阶段使用 UML 进行系统建模,确保系统流程清晰、设计规范。

2.1 用例图

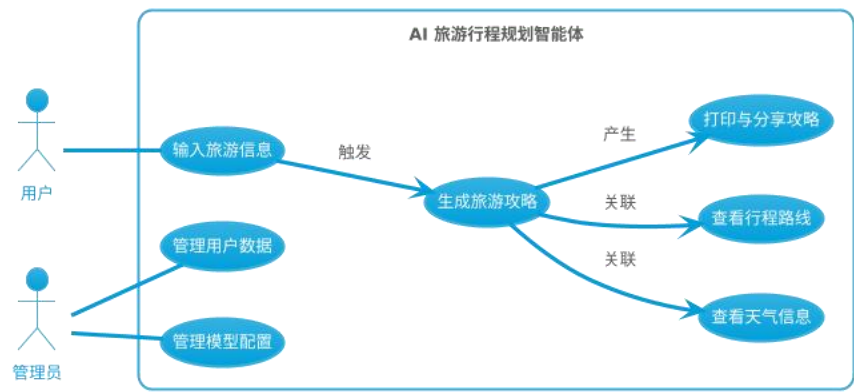


图 1. 用例图

2.2 类图

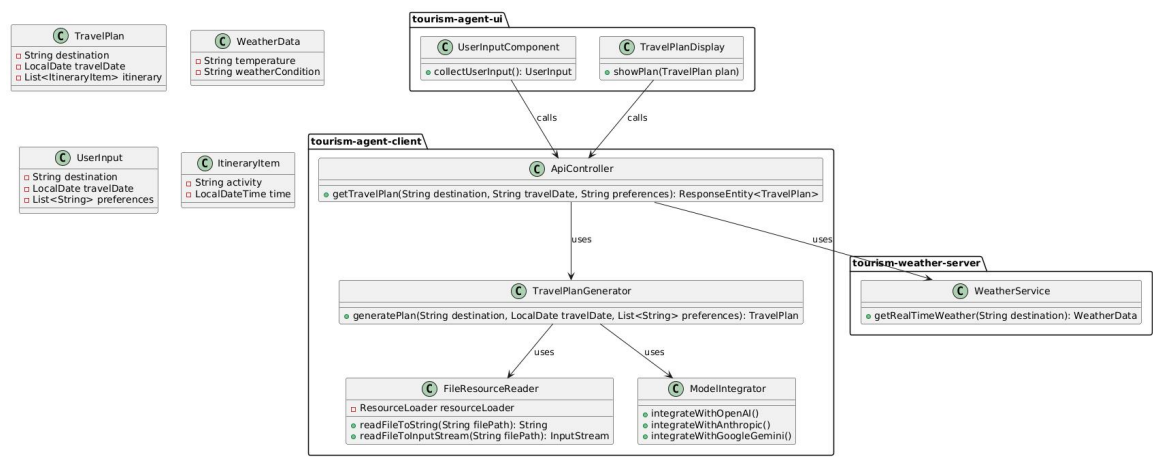


图 2. 类图

(1) tourism-agent-client 包：

- FileResourceReader：负责从类路径读取文件内容。
- TravelPlanGenerator：依据用户输入生成旅游计划。
- ApiController：提供 RESTful API 接口，供前端调用。
- ModelIntegrator：集成多种大语言模型。

(2) tourism-weather-server 包：

- WeatherService：获取目的地实时天气数据。

(3) tourism-agent-ui 包：

- UserInputComponent：收集用户输入信息。
- TravelPlanDisplay：展示生成的旅游计划。

(4) 实体类：

- TravelPlan：表示生成的旅游计划。
- WeatherData：封装天气数据。
- UserInput：存储用户输入信息。
- ItineraryItem：表示旅游计划中的行程项。

2.3 包图

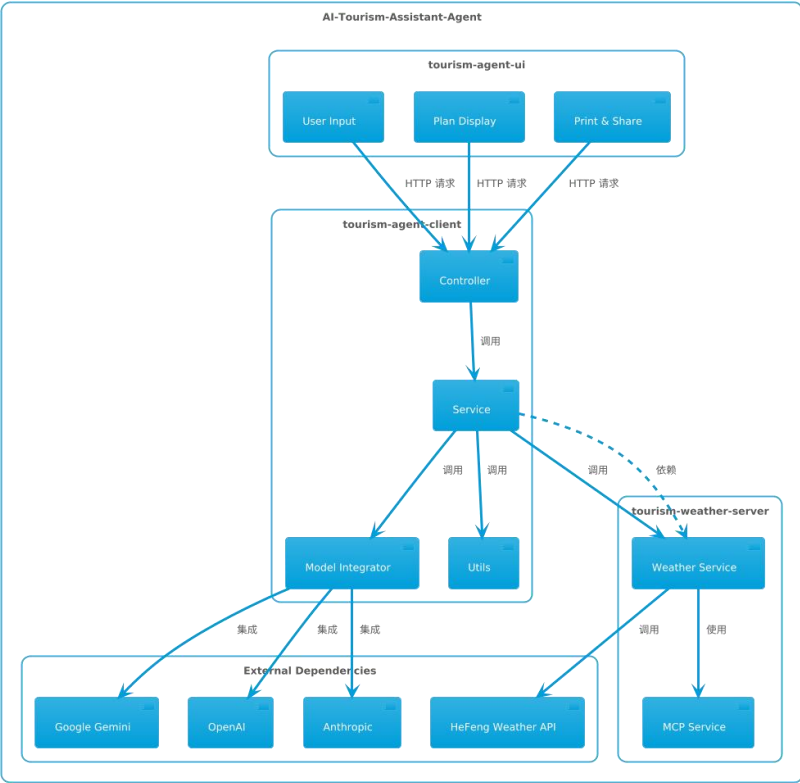


图 3. 包图

2.4 活动图

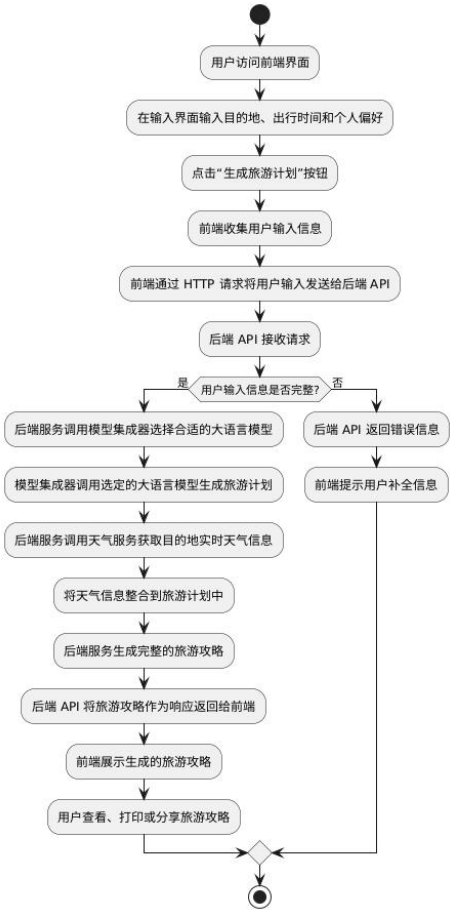


图 4. 活动图

2.5 部署图

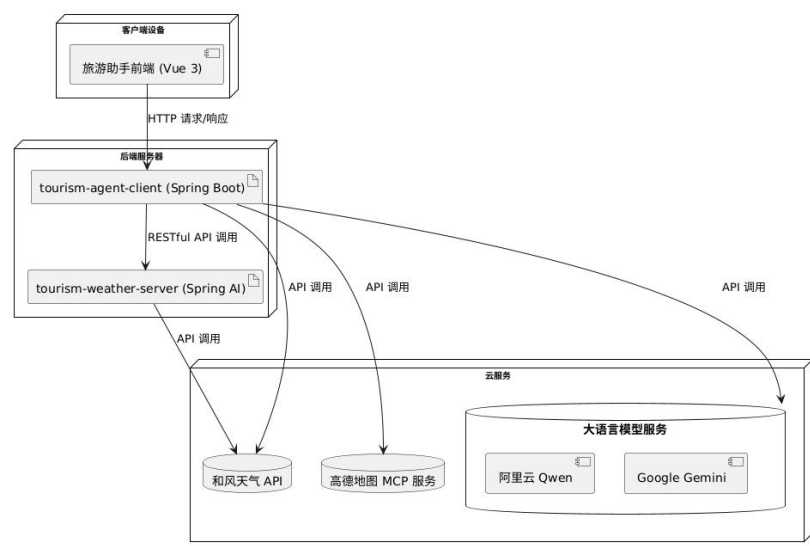


图 5. 部署图

三、扩展设想

当前系统已具备基本的旅游行程规划生成能力，但仍有大量升级空间。未来可考虑对如下几个方面进行扩展优化：

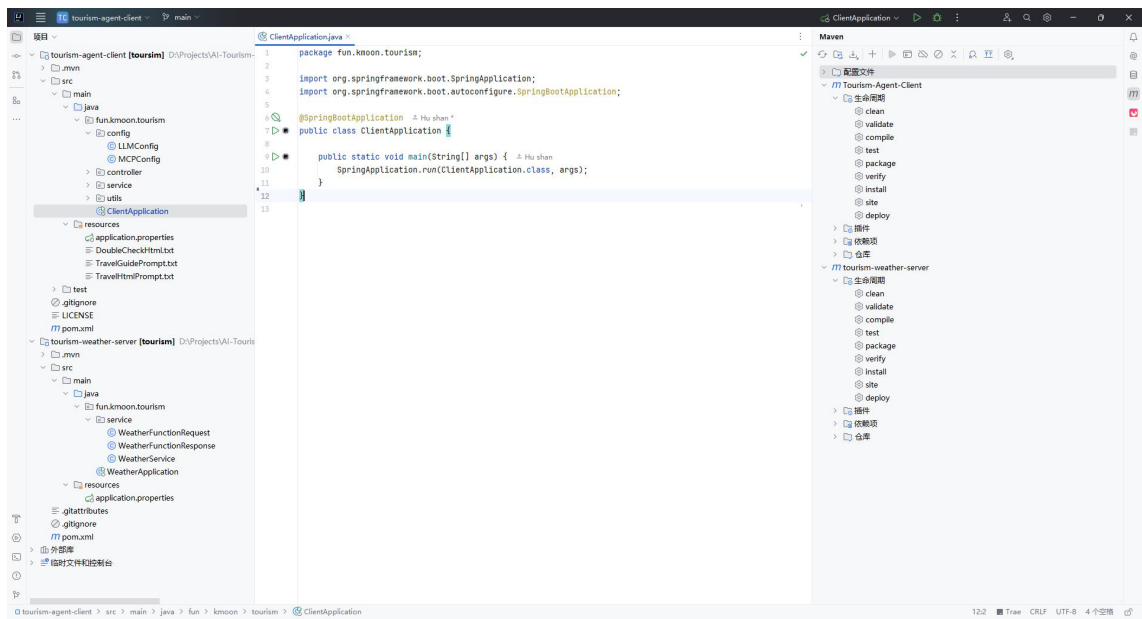
- 1. **多模态输入支持：**集成成熟的语音识别 SDK，如百度语音识别、科大讯飞语音识别等。在前端添加语音输入按钮，用户点击后可通过麦克风输入语音指令，系统将语音转换为文本，解析其中的行程查询需求，如“我想在暑假去北京旅游，帮我规划行程”；利用深度学习框架，如 TensorFlow 或 PyTorch，训练图像识别模型，识别地图中的地理位置、景点照片中的景点名称等信息。用户上传地图或景点照片后，系统提取关键信息，结合用户的其他输入进行行程规划。
- 2. **用户画像系统：**记录用户每次的输入信息、生成的旅游行程、点击的景点推荐、收藏的旅游攻略等行为数据。同时，在用户注册时收集一些基本信息，如年龄、性别、职业等，为每位用户建立偏好档案，根据历史行为自动推荐目的地与行程，更加个性化。
- 3. **支持移动端小程序：**考虑到现代社会用户使用手机端 APP 应用或服务占比显著，后续可以考虑将该项目封装为微信小程序，提升用户可达性。
- 4. **多语言支持：**考虑到全球用户的需求，未来可以加入多语言支持，使得系统能够为不同国家的用户提供本地化的旅游规划服务，同时集成专业的机器翻译服务，如 Google Translate API、百度翻译 API 等。在前端添加语言选择功能，用户可选择自己熟悉的语言进行操作。。
- 5. **多人协同旅行规划：**引入群组功能，支持多人对同一行程进行修改和投票，适配家庭/朋友出行。

## 四、成果截图

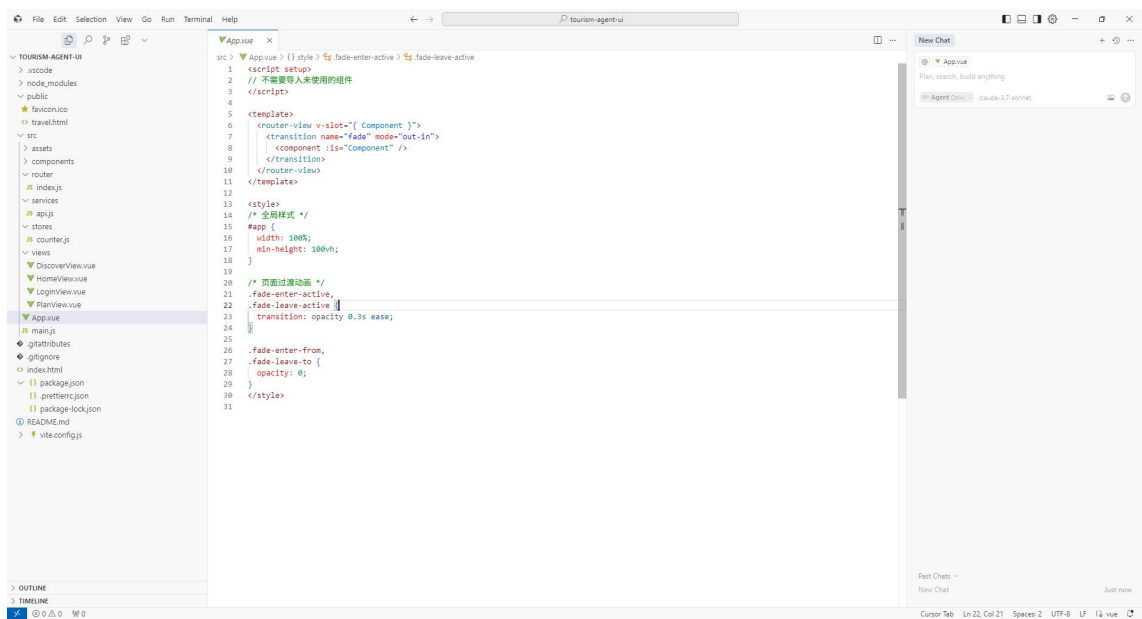
为提升文档阅读体验，将 IDE 显示切换至 Light UI 进行截图。

### 4.1 IDE 中项目窗口截图

#### (1) 后端服务（IDEA）

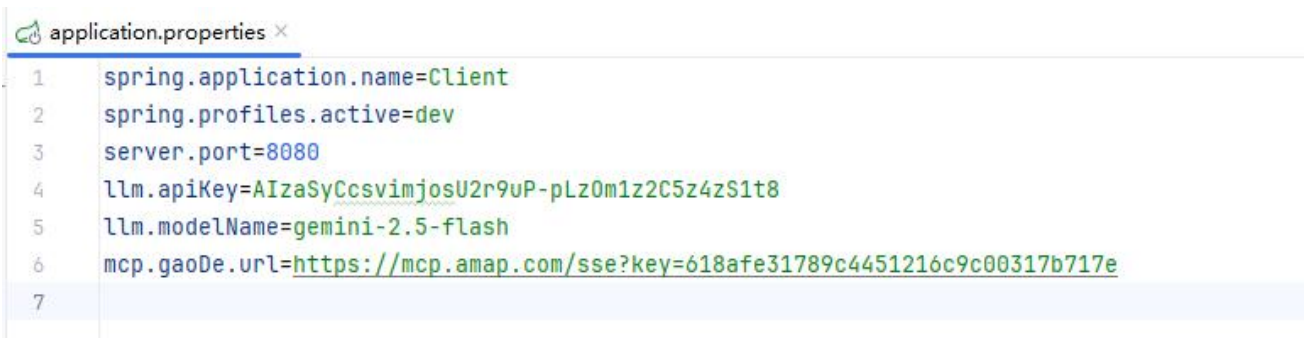


#### (2) 前端项目（Cursor）



### 4.2 配置文件截图

#### (1) tourism-agent-client



(2) tourism-weather-server

```

D:\...application.properties
1  spring.application.name=weather
2  spring.main.banner-mode=off
3  server.port=9000
4  spring.ai.mcp.server.name=my-weather-server
5  spring.ai.mcp.server.version=0.0.1
6  hefeng.weather.host=https://q86yvtwakk.re.qweatherapi.com
7  hefeng.api.key=217fec459a9e4d65afba0fe585f1fd81
8  spring.ai.mcp.server.request-timeout=1000000

```

(3) tourism-agent-ui

```

vite.config.js
vite.config.js > default > server > proxy > /travel/chat
1  import { fileURLToPath, URL } from 'node:url'
2  import { defineConfig } from 'vite'
3  import vue from '@vitejs/plugin-vue'
4
5  export default defineConfig({
6    plugins: [
7      vue(),
8    ],
9    resolve: {
10     alias: {
11       '@': fileURLToPath(new URL('./src', import.meta.url))
12     },
13   },
14   server: {
15     proxy: {
16       '/travel/chat': {
17         target: 'http://localhost:8080',
18         changeOrigin: true
19       },
20     },
21     fs: {
22       allow: ['..']
23     },
24   },
25   publicDir: 'public'
26 })
27

```

(4) Prompt 截图

```

TravelGuidePrompt.txt
1  # Role: 资深旅行策划AI助手
2
3  ## 主要任务
4
5  根据用户提供的具体目的地`[目的地具体名称]`、期望出行时间`[期望出行时间]`以及个人兴趣偏好`[兴趣偏好]`，策划并生成一份可以直接采纳和使用的、完整的、高质量
6
7  ## 工作流程
8
9  1. **深度理解用户需求与偏好**：
10     - 准确记录用户指定的`[目的地具体名称]`。
11     - 明确用户计划的`[期望出行时间]`（例如，X年X月X日到X月X日，或X月份）。
12     - 全面理解用户在`[兴趣偏好]`中表达的各项兴趣点（例如：自然风光、历史遗迹、美食探索、购物、户外活动、艺术文化、亲子等）。
13
14  2. **信息搜集与分析**：
15     - **核心信息来源**：参考“小红书”平台上的相关旅行攻略、笔记和用户评论内容。在引用具体攻略内容时，尽可能尝试保留或注明原文的引用来源链接，以使用户查阅。
16     - **天气状况查询**：查询分析`[目的地具体名称]`在用户`[期望出行时间]`内的大致天气情况，包括平均气温、降雨概率、日照时长等。
17
18  3. **行程规划与内容撰写**：
19     - **兴趣点与目的地融合**：
20       - 巧妙地将用户的`[兴趣偏好]`与`[目的地具体名称]`的特色和魅力点相结合进行规划。
21       - 积极寻找目的地当地与用户兴趣点相关的体验项目或活动（即使不是完美匹配，也可推荐相似或可替代的优质选项）。
22       - 核心是提供一个符合用户期望、内容充实、切实可行且具有高质量体验的旅行方案。
23     - 路线规划与交通建议：
24       - 使用“高德地图”地图工具进行路线规划逻辑，为每日行程中的景点/活动点设计合理的游览顺序。
25       - 在每日行程中，清晰列出各景点/活动点之间的推荐交通方式，简述即可（例如：“可乘坐公交X路直达”、“打车约X分钟”或“步行约X分钟”），无需提供复杂的导航步骤。
26     - 个性化实用建议：
27       - 根据查询到的天气情况，给出具体的穿衣建议。
28       - 提供详细的携带物品提示，例如是否需要雨具、防晒用品、特殊户外装备、转换插头等。
29       - 餐饮住宿建议（可选，若用户提及或为多日行程）：
30       - 可根据用户兴趣（如美食探索）或行程需要，推荐当地特色餐厅或住宿区域。
31
32  4. **攻略整理与优化**：
33     - 确保攻略内容完整，覆盖行程、交通、天气、穿衣、必备品等关键要素。
34     - 核查所有信息的准确性（如景点开放时间、大致票价范围等，提醒用户出行前再次确认）。
35     - 以友好、贴心的口吻撰写攻略，语言生动，易于阅读和理解。
36     - 整体排版清晰，结构合理，方便用户直接采纳使用。

```



4.3 核心类代码截图

(1) WeatherFunctionResponse 类

```
1 package fun.kmoon.tourism.service;
2
3 import com.fasterxml.jackson.annotation.JsonProperty;
4 import com.fasterxml.jackson.annotation.JsonPropertyDescription;
5 import lombok.Data;
6
7 @Data 2 个用法  ± Hu shan
8 public class WeatherFunctionResponse {
9
10     @JsonProperty(value = "obsTime")
11     @JsonPropertyDescription("数据观测时间")
12     private String obsTime;
13
14     @JsonProperty(value = "temp")
15     @JsonPropertyDescription("温度 (摄氏度)")
16     private String temp;
17
18     @JsonProperty(value = "feelsLike")
19     @JsonPropertyDescription("体感温度 (摄氏度)")
20     private String feelsLike;
21
22     @JsonProperty(value = "icon")
23     @JsonPropertyDescription("天气状况图标代码")
24     private String icon;
25
26     @JsonProperty(value = "text")
27     @JsonPropertyDescription("天气状况描述")
28     private String text;
29
30     @JsonProperty(value = "wind360")
31     @JsonPropertyDescription("风向360角度")
32     private String wind360;
33 }
```

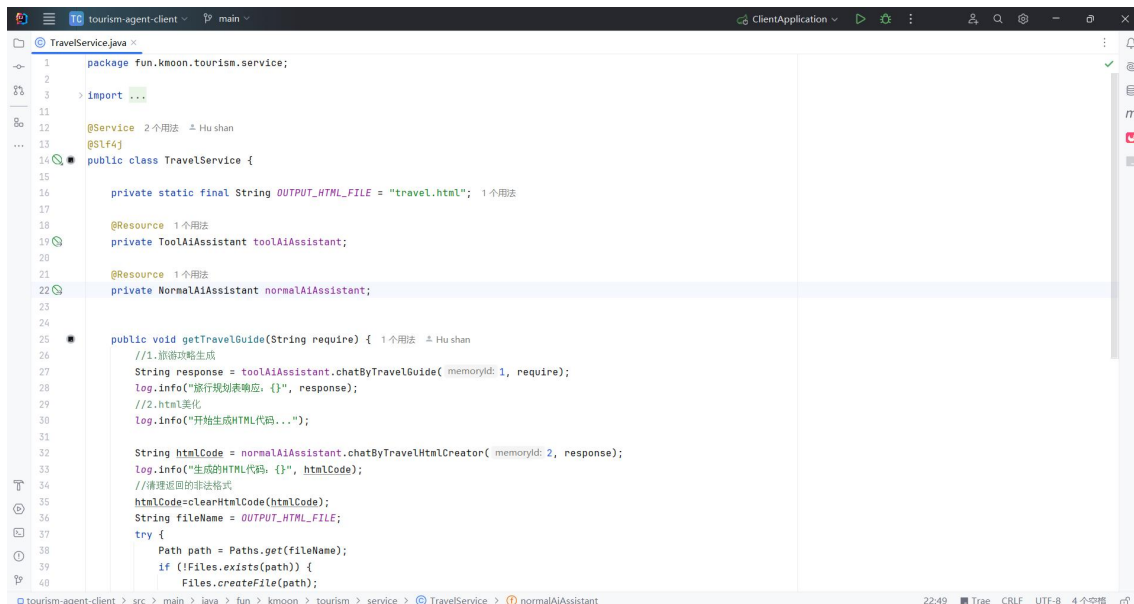
(2) LLM 模型配置类

```
22 @Configuration  ± Hu shan
23 @Slf4j
24 public class LLMConfig {
25
26     @Value("${AlzaSyCcsvInjosU2r9uP-pLz0mlz2C5z4zS1t8}")
27     private String apiKey;
28
29     @Value("${gemini-2.5-flash}")
30     private String modelName;
31
32     /**
33      * 阿里的模型
34      *
35      * @return
36      */
37     @Bean  ± Hu shan
38     public ChatModel chatLanguageModel() {
39         return OpenAiChatModel.builder()
40             .apiKey("sk-b088ddeb3274884a92e57df84f3967a") //sk-897f603d6cd341d8e2aa1d1a53ffac9
41             .modelName("qwen-plus-latest")
42             .maxRetries(1)
43             .maxTokens(8192)
44             .timeout(Duration.ofMinutes(40))
45             .logRequests(true)
46             .logResponses(true)
47             .baseUrl("https://dashscope.aliyuncs.com/compatible-mode/v1")
48             .build();
49
50     /**
51      * 使用LangChain4J的高级API来构建一个AI助手，注入 MCP Client
52      * 构建可以使用工具的AI助手
53      */
54 }
```

(3) MCP 服务配置类

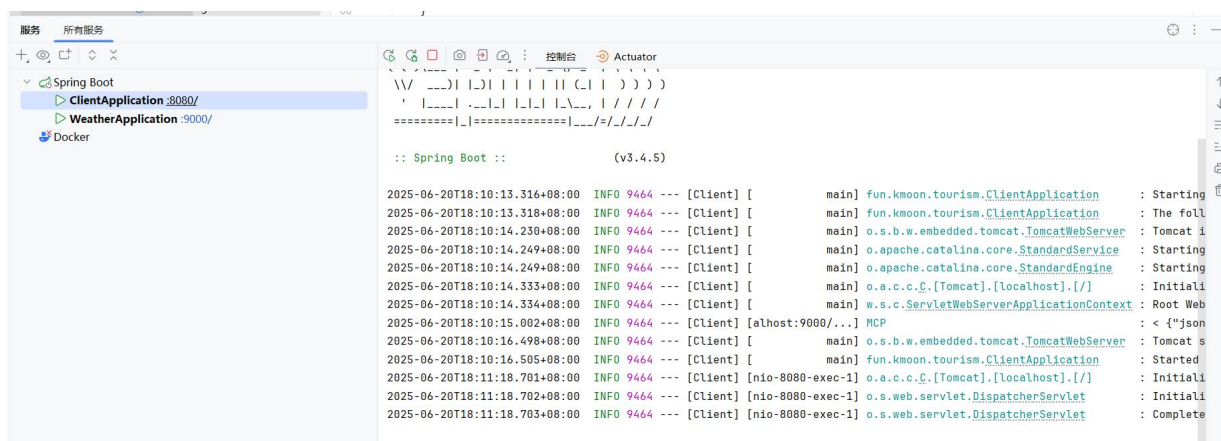
```
1 package fun.kmoon.tourism.config;
2
3 > import ...
4
14
15 @Configuration  ± Hu shan
16 public class MCPConfig {
17
18     @Value("${https://mcp.amap.com/sse?key=618afe31789c4451216c9c00317b...}")
19     private String gaoDeUrl;
20
21     /**
22      * 自建天气 SSE MCP服务
23      *
24      * @return
25      */
26     @Bean  ± Hu shan
27     public McpClient mcpClientWeather() {
28         McpTransport transport = new HttpMcpTransport.Builder()
29             .sseUrl("http://localhost:9000/sse")
30             .timeout(Duration.ofMinutes(5))
31             .logRequests(true) // if you want to see the traffic in the log
32             .logResponses(true)
33             .build();
34         return new DefaultMcpClient.Builder()
35             .transport(transport)
36             .build();
37
38     /**
39      * 高德地图 SSE MCP服务
40      *
41      * @return
42      */
43     @Bean  ± Hu shan
```

## (4) TravelService 类

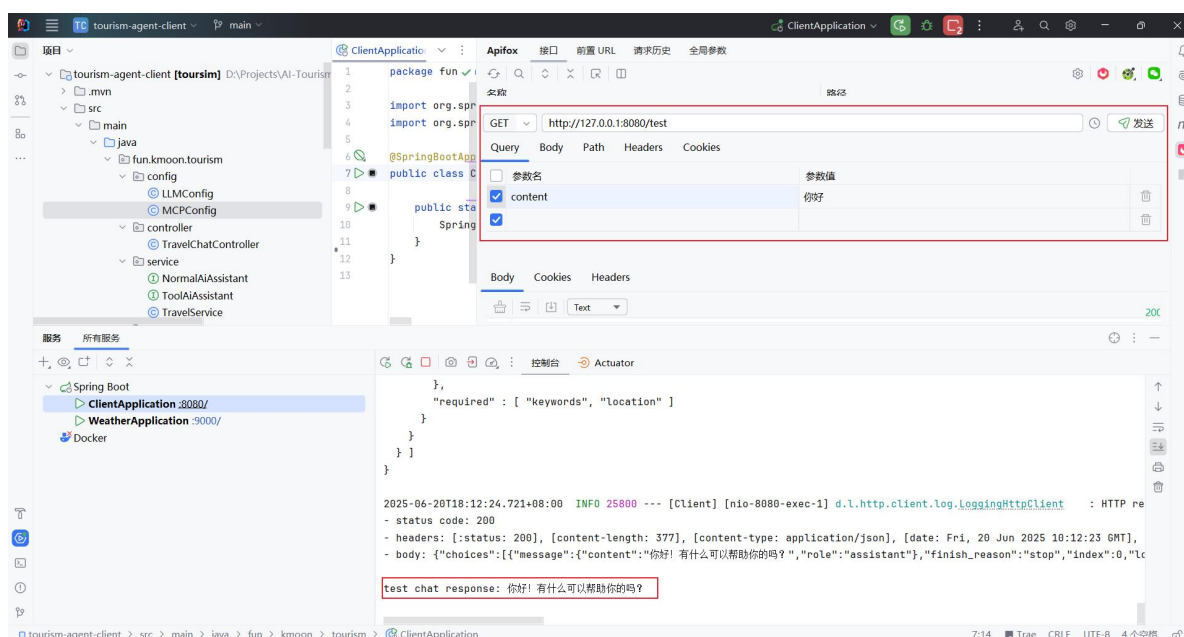


## 4.4 ApiFox 请求截图

首先启动后端项目：

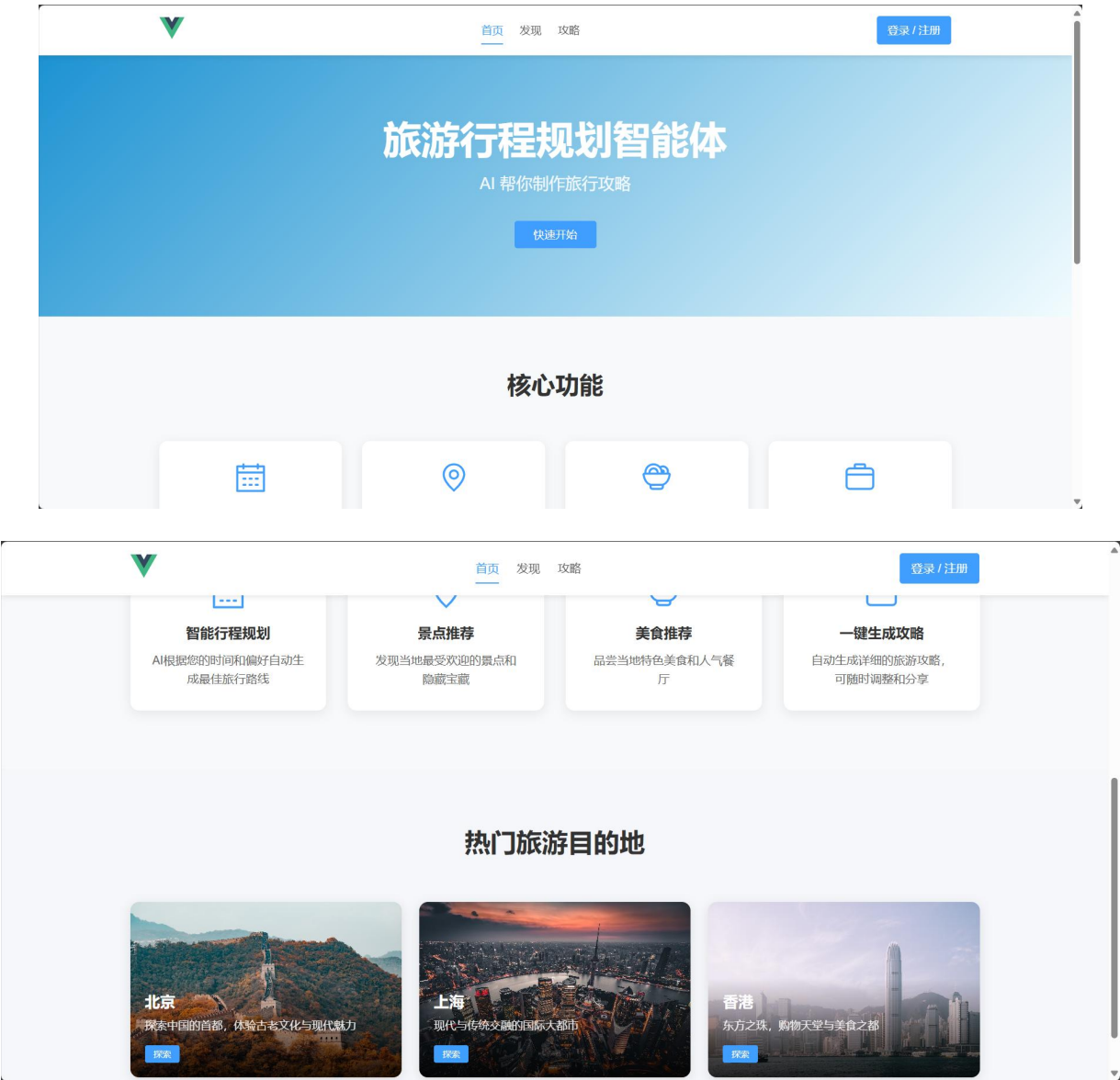


使用 Apifox 插件测试项目：



4.5 前端页面截图

首页：



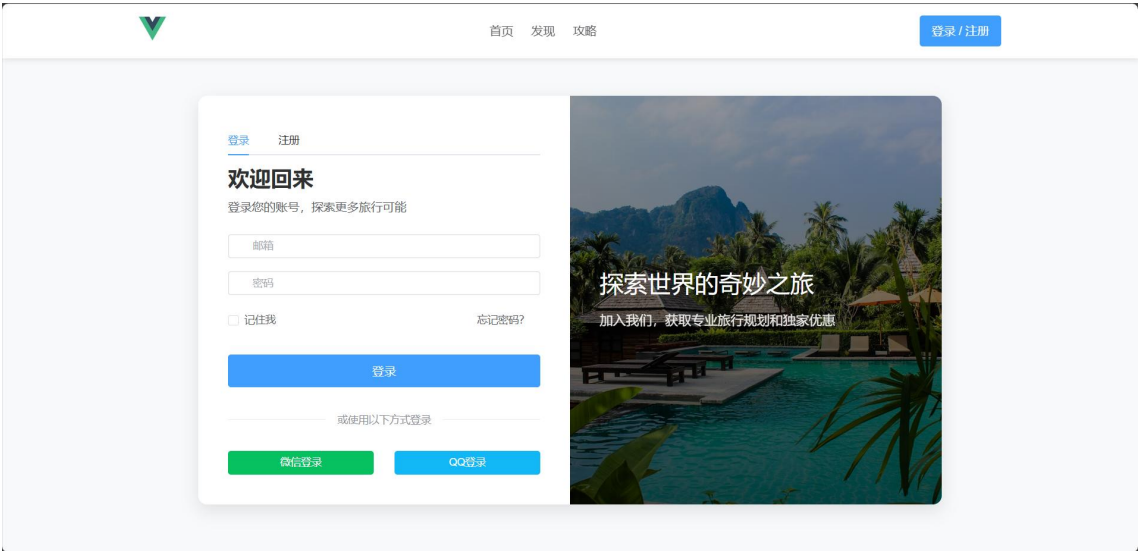
发现页：



Agent 功能页：



登录页：

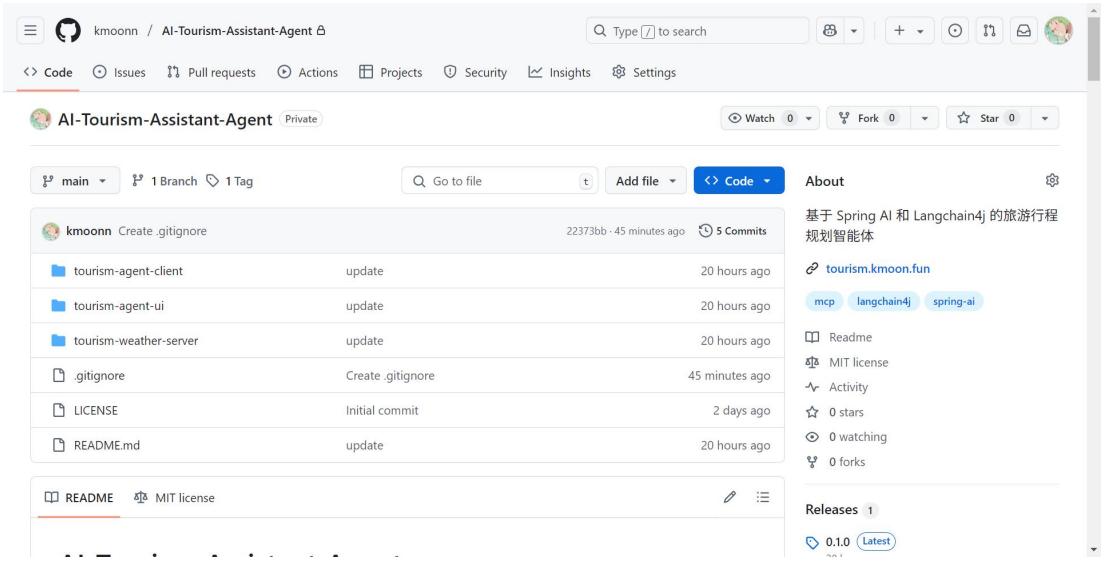


4.6 生成旅游规划 HTML 示例截图





4.7 Github 代码仓库截图



五、课程总结

通过本次课程的学习以及独立完成该结课作业，我深入了解了 Spring Boot、LangChain4j 与 AI Agent 在实际应用中的结合方式，掌握了如何通过 LangChain4j 封装多轮提示、管理上下文，并结合 Spring AI 实现 AI 应用的企业级整合。同时，也对 MCP 协议的使用及其在调用如高德地图、和风天气 API 等工具中的作用有了实践体验，了解其在调用高德地图、和风天气 API 等工具时的关键作用。这不仅拓宽了技术视野，还增强了利用外部服务丰富项目功能的能力。

本次项目让我从传统后端开发中跨入了 AI 应用开发领域，真正体验了"Agent 智能体"的落地过程。整体而言，对本次课程教学非常满意。实践环节的设计十分合理，是在学校里为数不多能接近企业、贴近市场、模拟真实工作环境的课程。这种教学模式让学生能更好地将理论知识应用到实际项目中，提前适应企业的工作节奏和需求。建议后续课程可增加对 AI 工具链（如 LangChain、Flowise、Function Call 机制等）的讲解，帮助学生更好地实现智能化项目。对课程教学非常满意，实践环节设计合理，是我在学校接受到的为数不多接近企业、贴近市场、真实工作环境的课程学校，希望未来能更多结合真实业务场景进行演练与模拟部署。