“半日闲”应用设计文档

一、系统概述

“半日闲”是一个本地场景活动规划与执行 Agent，接受一句自然语言目标，在数分钟内输出可执行的完整出行规划方案，并自动完成关键下单与预约动作。用户无需逐一搜索、比价、排队抢位，Agent 完成**从意图解析到订单提交的全链路闭环**。系统由后端 Agent 与前端交互层两部分构成。后端负责规划策略与工具调用，前端通过 SSE 流式接收规划进度并提供确认预约的交互入口。所有工具支持独立 Mock，确保在无真实外部 API 的环境下完整演示规划与预约全流程。

二、Planning 策略

2.1 七阶段规划流程

Agent 的规划过程分为七个有序阶段，每阶段调用对应工具，输出作为下一阶段的输入，如表 1 所示。

表 1 “半日闲”七阶段规划流程

阶段	名称	核心工具	输出
1	意图解析	decompose_goal(text)	时间窗口 / 人数 / 约束 / 偏好关键词
2	历史记忆检索	get_memory(user_id, ctx)	历史偏好向量 + 家庭画像
3	用户画像合成	analyze_user_profile(decomposed, memory)	综合 profile 对象（偏好、禁忌、人群）
4	时间槽规划	plan_time_slots(profile, now)	可用时段列表（含交通缓冲）
5	候选地点检索	search_places(filters)	[{place_id, open_hours, price, score_meta}]（严格来自 places 表）
6	排序与组合求解	rank_places_for_plan() + score_plans()	2-3 套候选方案（含 feasibility、risks）
7	方案呈现与执行	mock_reserve(step)	用户确认后生成 {type:reserve, status:pending} 并提交订单

2.2 评分模型

每套候选方案按五个维度进行加权评分，综合得分决定最终推荐顺序，如表 2 所示。

表 2 评估方案及权重

维度	名称	评分依据	默认权重
T	时间匹配度	营业时间覆盖率 + 排队时间预估	0.25
D	动线合理性	站点间交通时间最小化（地图 API 支持）	0.20
P	人群匹配度	亲子友好 / 年龄适宜 / 饮食偏好（减肥等）	0.30
C	价格合理性	人均消费是否满足预算约束	0.15
S	体验质量	平台评分 + 稀缺性 + 季节性加成	0.10

2.3 候选组合求解

采用贪心算法优先，对约束满足要求较高的场景可替换为整数线性规划。每次输出 2-3 套候选方案，每套方案包含：（1）intro：方案概述（一句话）；（2）steps：活动步骤列表，每步含时间、地点、meta、reason；（3）risks：风险提示（如：该餐厅需提前 1 天预约、节假日可能排队 45 分钟等）；（4）feasibility：可行性评分（0-1），低于 0.6 时自动降级搜索替代方案。

三、工具调用链路

3.1 设计思路

链路的核心设计目标是解耦与可替换。Agent 层不直接依赖任何工具的具体实现，所有调用均通过**工具注册中心（tool_registry.py）统一分发。**工具在系统启动时完成注册，注册信息包含实现函数引用与对应的入参、出参 schema。这一设计使任意工具的实现均可在不修改 Agent 层代码的前提下替换为 Mock 版本，为演示环境的稳定运行提供保障。

3.2 调用规范

每次工具调用经历三个强制阶段：调用前校验入参结构（字段类型、必填项、枚举值），执行工具业务逻辑，调用后校验出参结构。任意校验失败均触发降级流程，不允许将非法数据传入下游规划阶段。所有调用均记录工具名称、入参摘要、出参摘要与执行耗时，通过全局 trace_id 与请求绑定，支持全链路的问题追踪。

3.3 SSE 推送协议

规划过程通过 **SSE 长连接实时推送至前端**，事件类型共五种：status 通知阶段切换，intent 传递意图解析结果或澄清请求，skill_call 告知当前正在调用的工具，observation 推送中间结果与风险预警，done 推送最终方案或致命错误信息。所有事件携带 trace_id，前端依据事件类型驱动 UI 状态机流转，不持有任何规划状态。

四、异常处理机制

4.1 三级降级体系

Level 1（可恢复）：处理单次工具调用超时或返回空集的情况。系统执行最多 2 次指数退避重试，重试失败后放宽搜索条件或使用缓存数据，规划继续推进，用户无感知。

Level 2（功能降级）：在外部服务断路器开路时触发。系统切换至本地只读缓存继续规划，通过 observation(type: warning) 事件告知前端当前部分信息来源于缓存，可能与实时状态存在偏差，由用户决定是否继续。

Level 3（致命错误）：在 LLM 输出连续校验失败或数据库不可达时触发。系统通过 done(type: error) 终止规划流程，推送包含 trace_id 的结构化错误信息，退化为纯对话模式，前端展示重试入口与客服联系渠道。

4.2 断路器机制

对地图、天气、支付等外部服务各自维护独立的断路器实例，状态在 Closed、Open、Half-Open 三态间流转。正常态下请求正常转发；连续失败超过阈值后进入开路态，后续请求直接使用兜底数据响应，避免超时阻塞规划链路；冷却期满后进入探测态，以少量请求验证服务是否恢复，探测成功则回归正常态，失败则重置冷却计时。各服务的失败阈值与冷却时长均可在 config.py 中独立配置。