

# 自“游”自在--个性化游玩之旅

## 旅游攻略小程序 - 产品需求文档

文档版本：v1.0

最后更新：2025 年 10 月

文档状态：草案

### 目录

旅游攻略小程序 - 产品需求文档	1
1. 文档概述	3
1.1 产品背景	3
1.2 产品定位	3
1.3 目标用户	3
1.4 解决痛点	3
2. 产品功能架构	4
3. 详细功能需求	5
3.1 智能攻略制作器	5
3.2 发帖社区模块	7
3.3 匹配搭子社区	8
4. 非功能需求	9
4.1 性能需求	9
4.2 安全需求	10
4.3 可用性需求 - 学习友好级别	10
4.4 开发效率需求 - 项目完成导向	10
5. 数据需求	10
5.1 核心数据模型	11
5.2 数据分析指标	11
6. 运营需求	11
6.1 内容运营	11

6.2 用户运营 .....	12
7. 项目里程碑 .....	12
7.1 Phase 1 .....	12
7.2 Phase 2 .....	12
7.3 Phase 3 .....	12
8. 风险评估 .....	13
8.1 技术风险 .....	13
8.2 运营风险 .....	13
9. 成功指标 .....	13
9.1 技术实现里程碑指标 .....	13
9.2 代码与工程化指标 .....	14
9.3 个人与团队学习成果指标 .....	15

# 1. 文档概述

## 1.1 产品背景

当前旅游市场信息过载，用户难以快速找到符合个人偏好的旅游方案。本产品旨在通过智能攻略制作、分类社区和搭子匹配，解决用户个性化旅游规划需求。

## 1.2 产品定位

一站式个性化旅游规划与社交平台，整合工具、内容与社交三大功能。

## 1.3 目标用户

- 主要用户：18-35 岁年轻旅行者
- 用户细分：
  - 自由行爱好者
  - 社交旅行需求者
  - 内容分享爱好者

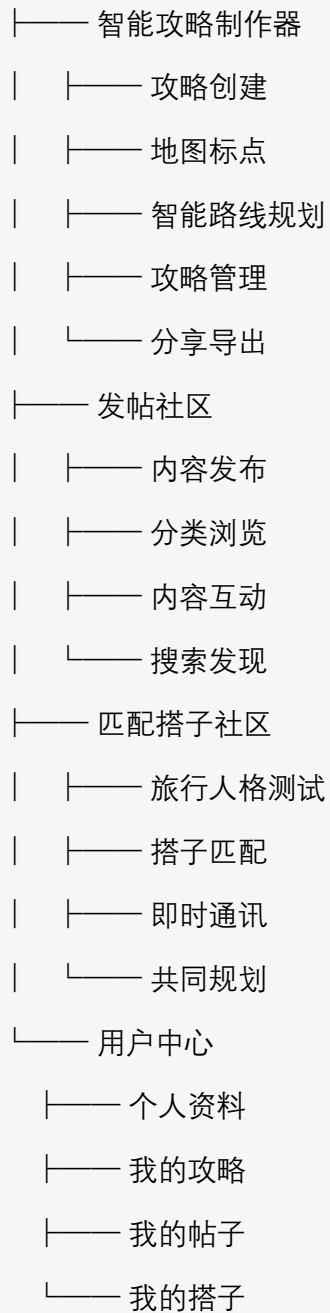
## 1.4 解决痛点

1. 旅游信息筛选困难
2. 行程规划繁琐耗时
3. 难以找到志同道合的旅行伙伴
4. 个性化旅游需求无法满足

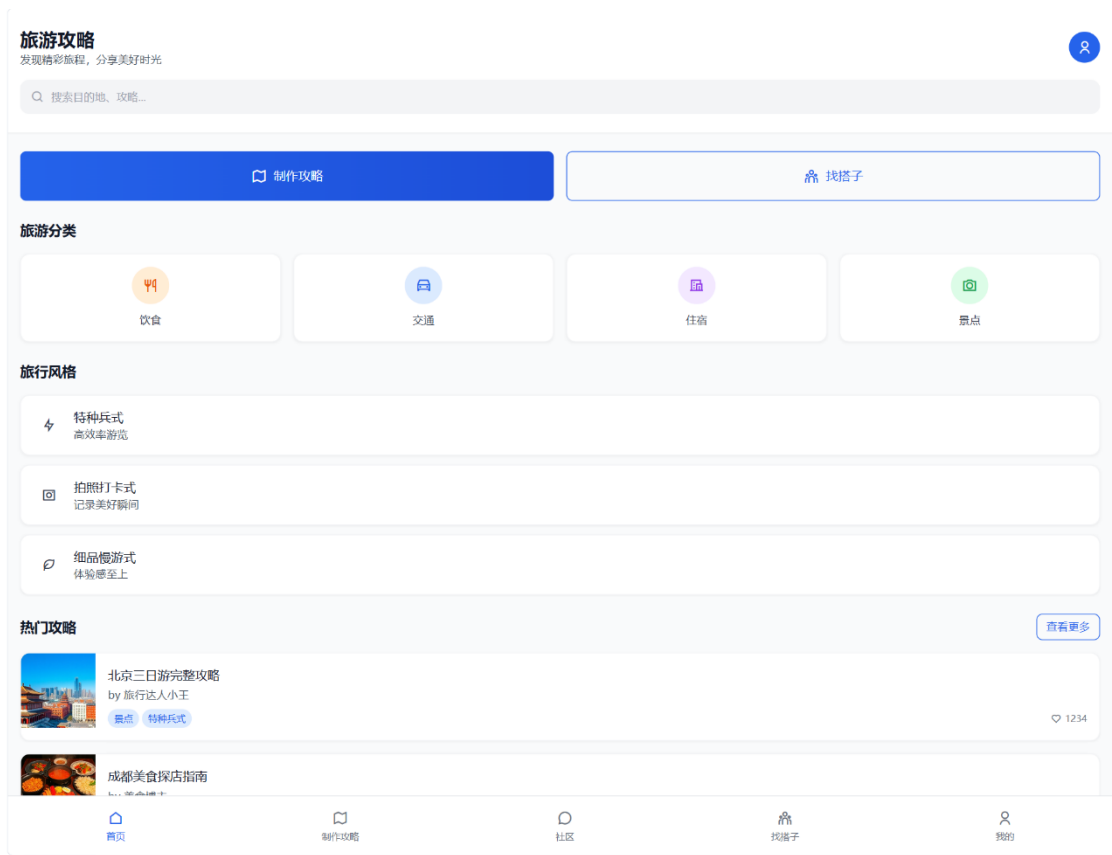
## 2. 产品功能架构

Plain Text

旅游攻略小程序



### 3. 详细功能需求



#### 3.1 智能攻略制作器

##### 3.1.1 攻略创建（FR-001）

- 功能描述：用户创建新的旅游攻略
- 需求详情：
  - 输入攻略基本信息：标题、目的地、旅行日期
  - 选择攻略可见性：公开/私密
  - 自动生成攻略 ID
- 验收标准：
  - 创建成功后在个人攻略列表中显示
  - 基本信息完整保存

### 3.1.2 地图标点功能 (FR-002)

- **功能描述：** 在地图上标记旅游点位
- **需求详情：**
  - a. 集成地图 API (高德/腾讯地图)
  - b. 搜索商家并添加到点位库
  - c. 手动地图点击添加点位
  - d. 点位信息编辑：名称、类型、停留时间、备注、图片
- **验收标准：**
  - a. 搜索响应
  - b. 点位信息完整保存
  - c. 地图操作流畅

### 3.1.3 智能路线规划 (FR-003)

- **功能描述：** 自动生成优化行程路线
- **需求详情：**
  - a. 基于算法自动规划每日行程
  - b. 考虑因素：地理位置、营业时间、交通方式、用户标签
  - c. 手动拖拽调整功能
  - d. 时间轴视图展示
- **验收标准：**
  - a. 规划结果合理可行
  - b. 手动调整实时生效
  - c. 时间估算准确度>85%

### 3.1.4 攻略发布与管理 (FR-004)

- **功能描述：** 攻略的保存、发布和分享
- **需求详情：**
  - a. 选择公开发布需填写完整信息
  - b. 标签系统：第一类+第二类分类
  - c. 导出为 PDF/图片
  - d. 分享到微信好友/朋友圈
- **验收标准：**
  - a. 发布成功后出现在社区对应分类
  - b. 导出文件格式正确
  - c. 分享链接可正常访问

## 3.2 发帖社区模块

### 3.2.1 内容发布 (FR-005)

- **功能描述：** 用户发布旅游相关帖子
- **需求详情：**
  - a. 富媒体编辑器：图文混排、视频
  - b. 强制分类标签选择
  - c. 关联已有攻略功能
  - d. 草稿保存和编辑
- **验收标准：**
  - a. 发布内容格式正确
  - b. 分类标签准确关联
  - c. 内容审核机制有效

### 3.2.2 内容浏览与发现 (FR-006)

- **功能描述：** 用户浏览和发现内容
- **需求详情：**
  - a. 推荐信息流（算法推荐）
  - b. 分类浏览：双层级分类系统
  - c. 搜索功能：关键词+标签搜索
  - d. 互动功能：点赞、收藏、评论、转发
- **验收标准：**
  - a. 推荐内容相关度高
  - b. 分类筛选准确
  - c. 搜索响应快速准确

### 3.3 匹配搭子社区

#### 3.3.1 旅行人格测试（FR-007）

- **功能描述：** 建立用户旅行人格画像
- **需求详情：**
  - a. 简版旅行人格测试（5-8 题）
  - b. 基于 MBTI 的 P/I 分类
  - c. 用户可手动修改标签
  - d. 人格画像可视化展示
- **验收标准：**
  - a. 测试结果合理准确
  - b. 用户可理解人格标签含义

#### 3.3.2 搭子匹配系统（FR-008）



- **功能描述：** 智能匹配旅行伙伴
- **需求详情：**
  - a. 发布搭子需求：目的地、时间、预算、期望人格
  - b. 智能匹配推荐
  - c. 搭子广场浏览
- **验收标准：**
  - a. 匹配推荐相关度高
  - b. 需求发布流程顺畅

### 3.3.3 沟通与协作 (FR-009)

- **功能描述：** 匹配用户间的交流协作
- **需求详情：**
  - a. 专属聊天室
  - b. 共同攻略编辑
  - c. 安全举报机制
  - d. 行程协调工具
- **验收标准：**
  - a. 聊天功能稳定流畅
  - b. 协作编辑实时同步

## 4. 非功能需求

### 4.1 性能需求

1. 核心功能模块不崩溃

2. 数据持久化基本可靠（本地存储+基础云端）
3. 异常情况优雅降级（如地图 API 失败时显示列表）
4. 关键业务流程有完整的状态管理

## 4.2 安全需求

1. 用户数据加密存储
2. 内容审核机制
3. 隐私保护设置
4. 实名认证可选

## 4.3 可用性需求 - 学习友好级别

1. 核心功能操作路径清晰，有基础引导
2. 错误提示明确，便于调试定位问题
3. 关键数据操作有确认提示（防误操作）
4. 网络异常、加载状态有基础反馈

## 4.4 开发效率需求 - 项目完成导向

1. 模块化程度：基础组件复用
2. 代码可读性：关键函数有注释，结构清晰
3. 调试支持：关键流程有日志输出
4. 部署流程：可一键部署到测试环境

## 5. 数据需求

## 5.1 核心数据模型

### Plain Text

用户表：用户 ID、个人信息、旅行人格、创建时间

攻略表：攻略 ID、创建者、目的地、行程数据、标签、状态

点位表：点位 ID、所属攻略、经纬度、类型、详细信息

帖子表：帖子 ID、作者、内容、媒体文件、分类标签、关联攻略

匹配表：匹配 ID、用户 A、用户 B、匹配度、状态、创建时间

## 5.2 数据分析指标

1. 日活跃用户 (DAU)
2. 攻略创建数量
3. 匹配成功率
4. 用户留存率
5. 内容互动率

## 6. 运营需求

### 6.1 内容运营

1. 优质内容推荐机制
2. 用户成长体系
3. 社区规范管理

## 6.2 用户运营

1. 新用户引导流程
2. 活动运营策略
3. 用户反馈收集

## 7. 项目里程碑

### 7.1 Phase 1

1. 核心攻略制作功能
2. 基础社区发帖浏览
3. 用户系统

### 7.2 Phase 2

1. 搭子匹配系统
2. 即时通讯功能
3. 内容分类优化

### 7.3 Phase 3

1. 算法推荐优化
2. 社交功能完善
3. 性能优化

## 8. 风险评估

### 8.1 技术风险

1. 地图 API 性能问题
2. 实时通讯稳定性
3. 数据安全和隐私

### 8.2 运营风险

1. 冷启动内容匮乏
2. 社区内容质量管控
3. 用户匹配体验不佳

## 9. 成功指标

### 9.1 技术实现里程碑指标

#### 1. 环境搭建与基础框架

- 成功创建项目，并配置好基础开发环境
- 完成基础页面框架（Tab Bar、导航）并能正常跳转
- 完成用户登录模块

#### 2. 核心模块完成度

- **攻略制作器：**
  - a. 地图集成：成功调起地图，并实现基本的标点、显示功能
  - b. 点位管理：前端实现点位的添加、编辑、删除列表操作
  - c. 数据持久化：能将创建的攻略和点位数据保存到后端数据库，并能重新读取
- **发帖社区：**
  - a. 发布功能：实现包含文字、图片（或视频）的表单提交和后端接收
  - b. 数据渲染：前端能成功从后端获取帖子列表并正确渲染展示
  - c. 分类筛选：前端实现按分类标签过滤帖子列表的功能
- **匹配搭子：**
  - a. 人格测试：完成测试题目流程，并能将结果保存到用户档案
  - b. 匹配逻辑：后端实现一个简单的匹配算法（如根据标签匹配），并返回匹配列表
  - c. 简易通讯：实现一个最简单的文本聊天功能（无需高并发，能收发消息即可）

### 3. 前后端联调

- 完成至少 3 个核心 API（如：创建攻略、获取帖子列表、用户登录）的前后端成功联调
- 数据传输格式（如 JSON）统一，错误处理机制基本可用

## 9.2 代码与工程化指标

### 1. 代码质量

- 代码结构清晰，有基本的模块化划分（如将页面、组件、工具函数分开）
- 关键函数和复杂逻辑有清晰的注释
- 无明显的安全漏洞（如 SQL 注入风险、敏感信息硬编码）

## 2. 版本控制

- 使用 Git 进行代码管理
- 有合理的 Commit 信息，能反映开发进度
- 建立了主分支和开发分支的基本工作流

## 3. 文档完善度

- 编写了项目的 [README.md](README.md)，说明项目简介、技术栈和如何运行
- 核心模块有简单的设计说明或流程图
- 数据库有基本的结构文档（如 ER 图或表结构说明）

## 9.3 个人与团队学习成果指标

### 1. 技术栈掌握

- 前端：能够熟练使用生命周期、WXML/WXSS 语法、并理解数据绑定和事件处理
- 后端：能够设计数据库表结构、编写 API 接口、处理文件上传等常见任务
- 第三方服务：成功集成并调用至少 1 个第三方服务（如地图 API 或云存储）

### 2. 问题解决能力

- 能够独立查阅官方文档、使用搜索引擎解决大部分以上的开发中遇到的技术报错
- 在遇到复杂问题时，能清晰地描述问题、分析原因，并寻求有效的帮助

### 3. 项目流程体验

- 完整地体验了从“需求分析 -> 技术方案设计 -> 编码实现 -> 测试联调”的完整开发流程
- 对如何将一个产品想法落地为可运行的代码有了第一手的实践经验

| (注：文档部分内容可能由 AI 生成)