“校园咸鱼” 桌面端 V1.0 产品需求文档 (PRD)

文档版本	V1.0	创建日期	2025年10月19日
产品名称	校园咸鱼	创建人	李宣廷
文档状态	定稿	目标平台	桌面端 Windows

1. 产品背景与目标

1.1. 项目简介

“校园咸鱼”是一个轻量级的桌面应用程序，旨在帮助大学生处理他们不再需要但仍有价值的物品。它提供了一个简单的平台，让学生可以方便地发布闲置物品信息，以便其他同学可以获取这些物品。

1.2. 问题陈述 (用户痛点)

大学生（尤其是临近毕业季）拥有大量物品（如教科书、生活用品、小型电器等）。这些物品“扔掉可惜，不处理又占据空间”。目前缺乏一个简单、快捷、集中的校内信息发布渠道来解决这个问题。

1.3. 目标用户

在校大学生，包括：

- **发布者**：希望快速清空闲置物品，减少浪费的学生（如毕业生、换宿舍学生）。
- **浏览者**：希望以低成本（甚至免费）获取所需物品的学生（如新生、校内居住学生）。

1.4. V1.0 核心目标

验证核心功能闭环：**发布 -> 浏览 -> 查找 -> 联系**。V1.0 的重点是提供一个极其简单易用的本地工具，实现最基本的信息发布与检索功能。

2. 功能需求 (Functionl Requirements)

V1.0 必须包含以下四个核心功能模块。

FR-001: 物品列表展示 (主界面)

功能描述：

1. 应用程序的主窗口应默认显示所有已添加物品的列表。
2. 此列表应以表格或列表视图（List View）形式展示。
3. 列表应至少显示核心信息：**物品名称**、**物品描述（摘要）**、**联系人信息**。
4. 当物品数量过多时，列表必须支持垂直滚动。
5. 列表应在添加或删除物品后自动刷新，实时反映最新数据。

FR-002: 添加物品信息

功能描述：

1. 主界面必须有一个清晰的“添加物品”按钮。
2. 点击该按钮后，弹出一个模式对话框（Modal Dialog）用于信息录入。
3. **录入表单** 必须包含以下字段：
    - **物品名称**：[文本输入框]（必填项）。
    - **物品描述**：[多行文本区域]（选填项，建议限制最大字数，如500字）。
    - **联系人信息**：[文本输入框]（必填项，例如：微信/QQ/手机号/宿舍号）。
4. **表单操作**：
    - “确认/保存”按钮：点击后，执行校验（检查必填项）。校验通过，则保存数据，关闭对话框，并刷新主界面的物品列表（FR-001）。
    - “取消”按钮：点击后，关闭对话框，不保存任何信息。

FR-003: 删除物品信息

功能描述：

1. 用户必须能够在主界面的物品列表（FR-001）中选中一个物品。
2. 主界面必须有一个“删除物品”按钮。
3. 用户选中一个物品后，点击“删除物品”按钮。
4. **安全确认**：系统必须弹出一个确认对话框（例如：“您确定要删除「物品名称」吗？此操作不可撤销。”）。
5. 用户点击“确认”后，该物品信息从系统中被永久删除，并立即刷新主界面的物品列表。

FR-004: 查找物品信息

功能描述：

1. 主界面（FR-001）必须提供一个“搜索框”（文本输入框）。
2. 主界面必须提供一个“搜索”按钮（或支持输入后按回车键触发搜索）。
3. 用户输入关键词后点击“搜索”。
4. 系统应根据关键词，在所有物品的“物品名称”和“物品描述”字段中进行模糊匹配。
5. 物品列表（FR-001）应刷新，仅显示符合搜索条件的物品。
6. 必须提供一个“清除/重置”按钮，点击后清除搜索条件，物品列表恢复显示所有物品。

3. UI/UX 设计规范 (界面草图描述)

我们追求简洁、直观、原生的PyQt风格。

3.1. 主窗口 (Main Window)

- **布局**：垂直流式布局。

- **顶部区域（工具栏）**：
  - [标签] "搜索："
  - [文本输入框 - `QLineEdit`] (用于 FR-004)
  - [按钮 - `QPushButton`] "搜索"
  - [按钮 - `QPushButton`] "清除搜索"
- **中部区域（内容区）**：
  - [表格视图 - `QTableWidget` 或 列表视图 - `QListView`] (用于 FR-001)
  - 表格列：`| 物品名称 | 物品描述 | 联系人信息 |`
  - 该区域应占据窗口的主要空间，且可滚动。
- **底部区域（操作栏）**：
  - [按钮 - `QPushButton`] "添加新物品..." (用于 FR-002)
  - [按钮 - `QPushButton`] "删除选中物品" (用于 FR-003，默认禁用，选中列表项后激活)

3.2. 添加物品对话框 (Add Item Dialog)

- **类型**：模式对话框 (`QDialog`)。
- **布局**：表单布局 (`QFormLayout`)。
- **控件**：
  - "物品名称:" [`QLineEdit`] (带星号* 标记必填)
  - "物品描述:" [`QTextEdit`] (多行文本)
  - "联系信息:" [`QLineEdit`] (带星号* 标记必填)
- **底部按钮**：
  - [`QPushButton`] "保存"
  - [`QPushButton`] "取消"

3.3. 确认对话框 (Confirmation Dialog)

- 使用PyQt标准的 `QMessageBox.warning` 或 `QMessageBox.question` 来实现删除确认。
- 标题：“确认删除”
- 内容：“您确定要删除 [选中的物品名称] 吗？”
- 按钮：“是”、“否”。

4. 技术文档与规范 (V1.0)

4.1. 技术栈

- **语言**：Python 3.x
- **GUI框架**：PyQt (建议 PyQt5 或 PyQt6)
- **数据存储**：**CSV 文件**

4.2. 数据存储 (V1.0 核心 - CSV方案)

- **方案**：本地CSV文件。
- **理由**：实现简单，无需数据库依赖，数据直观易读，符合V1.0快速原型的需求。
- **数据文件**：`items.csv` (应存储在用户应用数据目录或程序根目录)。
- **CSV文件规范**：
  - **编码**：必须使用 `UTF-8` （以支持中文描述）。
  - **分隔符**：逗号 (`,`)。
  - **数据列 (Headers)**：
    1. `id` (唯一标识符)
    2. `name` (物品名称)
    3. `description` (物品描述)
    4. `contact_info` (联系人信息)
- **核心技术关注点**：

4.2.1. ID的生成 (FR-002: 添加)

- **问题**：CSV没有自增主键。
- **V1.0决策（建议）**：使用 **UUID** (Universally Unique Identifier)。
- **实现**：在添加物品时，使用 Python 的 `uuid` 库（如 `uuid.uuid4().hex`）生成一个唯一的字符串ID。
- **理由**：避免了在删除物品后ID重复的问题。

4.2.2. 数据读写 (FR-001, 002, 003)

- **规范**：
  - **读取 (FR-001, FR-004)**：程序启动时，一次性将 `items.csv` 读入内存中的一个列表（List of Dictionaries）。所有的“显示”和“搜索”操作都只针对此内存列表进行，**避免频繁IO**。
  - **写入 (FR-002: 添加, FR-003: 删除)**：当用户执行“添加”或“删除”操作后，程序应：
    1. 更新内存中的列表。
    2. **立即将完整的内存列表覆盖写回 (Overwrite)** `items.csv` 文件。
  - **模块**：必须使用 Python 内置的 `csv` 模块 (`csv.DictReader` 和 `csv.DictWriter`) 来处理读写，确保对CSV中的特殊字符（如描述中包含的逗号或换行符）进行正确转义。

4.2.3. 删除操作 (FR-003: 删除)

- **实现**：
  1. 用户在UI上选中一行，程序获取该行的 `id`。
  2. 从内存列表中移除具有该 `id` 的字典项。
  3. 调用4.2.2中的“写入”规范，将更新后的内存列表（已删除该项）完整写回 `items.csv` 文件。

4.2.4. 搜索操作 (FR-004: 查找)

- **实现**：
    1. 用户输入关键词。
    2. 程序遍历内存中的列表数据（*不是* 遍历CSV文件）。
    3. 在内存中筛选出 `name` 或 `description` 字段包含关键词的项。
    4. 将筛选结果更新到UI列表（FR-001）中。

4.3. 架构要求

- 建议采用 **MVC** (Model-View-Controller) 模式。
    - **Model (数据模型层)**：
        - 必须封装一个 `CsvDataManager` 类（或类似名称）。
        - **职责**：负责 `items.csv` 的初始化、 `load_data()` (加载CSV到内存), `save_data()` (保存内存到CSV), `add_item()`, `delete_item()`, `search_items()` 等。
    - **View (视图层)**：PyQt的窗口（主窗口, 添加对话框）。
    - **Controller (控制层)**：处理按钮点击事件，调用 `CsvDataManager` 的方法，并更新View。