项 目 编 号	200602006
文 档 编 号	１０
密　　级	内部

智能垃圾分类系统需求规格

V1.0

软件工程大作业

评　　审　　日　期：2025 年 12 月 6 日

目录

1 导言

1.1 目的

本需求规格说明文档用于完整描述“智能垃圾分类系统”的功能与非功能需求，明确系统要解决的业务问题、使用边界、角色与用例、数据与接口、性能与安全、实施与交付的约束与验收口径。文档重点依据用户提出的核心要求：通过“物品名称”分类垃圾类型，使用 Python 字典承载规则，提供桌面应用与规则编辑能力，并以 CSV 文件作为规则持久化存储。同时为后续扩展画像识别提供可扩展的结构。
本文档的预期读者包括但不限于：

- 设计人员（架构师、系统分析师）
- 开发人员（客户端、数据处理、UI/UX）
- 测试人员（功能测试、性能测试、验收测试）
- 项目管理人员（PM、Scrum Master）
- 用户与管理员（业务方、运维方）

1.2 范围

本需求聚焦“做什么”，即目标系统的功能与行为，不限定具体实现细节，但明确关键技术约束：

- 使用 Python 字典存储垃圾分类规则（物品名-类型-依据），并从 CSV 加载/保存
- 提供桌面应用，含文本输入与结果展示
- 管理员可添加/修改规则
- 输入项：物品名称
- 输出项：垃圾类型、分类依据
- 数据存储：CSV 文件
- 扩展建议：支持图片识别垃圾类型（集成简单 CV 模型）

1.3 缩写说明

CSV：Comma-Separated Values，逗号分隔值文件格式。
GUI：Graphical User Interface，图形用户界面。
CV：Computer Vision，计算机视觉。
UI/UX：用户界面/用户体验。
OS：操作系统。
ID：标识符。
CRUD：Create、Read、Update、Delete（增删改查）。

1.4 术语定义

- 垃圾类型：指依据城市垃圾分类指引所要求的分类类型，如可回收、有害、厨余、其他（本系统标准分类）。
- 分类依据：对物品归属某垃圾类型的理由简述，如“含重金属，应投有害垃圾”；或引用标准条目摘要。
- 规则：针对一个或一组物品名称的分类定义，包括物品名（或同义词/别名）、垃圾类型、分类依据等。
- 规则冲突：同一物品名存在多个不一致分类定义的情况。
- 同义词：多个名称指同一物品（如“充电宝/移动电源”）。
- 管理员：具有规则编辑、导入导出、备份恢复、系统配置权限的本地用户。
- 普通用户：仅进行查询与结果查看的本地用户。

1.5 引用标准

[1] 《企业文档格式标准》 V1.1（占位）
[2] 《需求规格报告格式标准》 V1.1（占位）
[3] 城市生活垃圾分类相关管理办法与地方标准（占位，具体引用由业务方提供）

1.6 参考资料

[1] 《UML》 V1.1
[2] 《需求规格报告格式标准》 V1.1
[3] CSV 文件格式约定与编码规范（内部约定）
[4] Python 3 标准库文档（字典、CSV、文件 IO）
[5] Tkinter/PyQt GUI 编程基础参考（占位）
[6] OpenCV/Pillow 图像处理基础参考（扩展用）

1.7 版本更新信息

本文档的更新记录如表 A-1。
表 A-1 版本更新记录

修改编号	修改日期	修改后版本	修改位置	修改内容概述
001	2025.10.17	0.1	全部	初始发布版本
002	2025.10.18	0.2	3.1 章节	增加
003	2025.10.19	0.3	4.1 章节	修改
004	2025.10.20	0.4	5.1 章节	修改
005	2025.10.21	1.0	7 章节	增加

2 系统定义

2.1 项目来源及背景

随着各地生活垃圾分类政策落地，居民对“如何正确分类”存在大量即时、碎片化的查询需求。传统解决方案常体现为网页查询、公众号推送或印刷手册，存在以下问题：

1. 获取路径分散，难以离线查询；
2. 术语晦涩，分类依据不直观；
3. 更新不易，规则编辑与追溯难；
4. 对特定地区的分类差异不敏感；
5. 图片识别需求增长，但轻量化方案缺失。

为此，提出“智能垃圾分类系统”，以“物品名称查询”为核心能力，提供：

1. 快速输入与即时反馈；
2. 分类结果清晰且附带依据；
3. 管理员可维护规则，支持版本化、有据可查；
4. CSV 持久化，便于导入导出与与其他系统对接；

后续可扩展至图片识别，满足更多场景（如社区志愿者现场辅助、便民服务站查询等）。

该系统定位为“轻量、离线可用、易维护、可扩展”的桌面工具，适用于：

1. 居民个人查询离线工具；
2. 社区/物业/街道办服务窗口查询台；
3. 学校/公共机构普及教育辅助；
4. 内部培训、知识库编辑与验证环境。

2.2 项目要达到的目标

本项目设定的目标如下：

1. 系统能够提供友好的用户界面，使操作人员的工作量最大限度的减少
2. 系统具有良好的运行效率，能够得到提高生产率的目的
3. 系统应有良好的可扩充性，可以容易的加入其它系统的应用。
4. 平台的设计具有一定的超前性，灵活性，能够适应企业生产配置的变化。
5. 通过这个项目可以锻炼队伍，提高团队的开发能力和项目管理能力

2.3 系统整体结构

系统总体由“用户端（查询）”与“管理端（维护）”构成，共享本地 CSV 数据存储；在扩展模式下包含“图片识别模块”。核心数据在运行时加载为内存 Python 字典以支持快速查询。如图 B-1 所示。

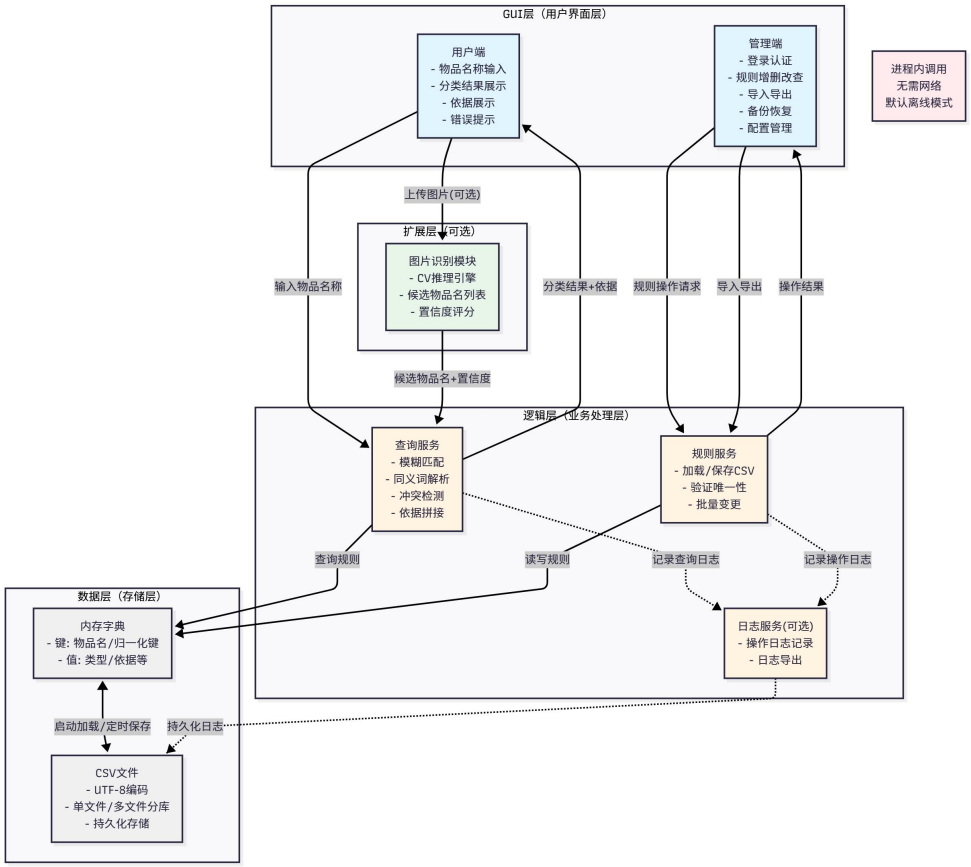


图 B-1：系统总体结构示

3 应用环境

本项目的应用环境可以分硬件环境、软件环境和网络环境来描述。

3.1 系统运行网络环境

默认离线工作：系统设计成“不依赖网络”的桌面应用，启动后即可进行名称查询与规则编辑，CSV 文件保存在本机指定目录。运行拓扑示意图如图 B-2 所示。

说明：

离线模式：

- 优点：响应快、可用性高、数据可控；
- 风险：多机协同可能造成规则版本不一致。

可选联网扩展（非本期范围）：

- 规则云端同步、协同编辑；
- 在线获取官方分类标准更新
- 模型在线更新与差分下载

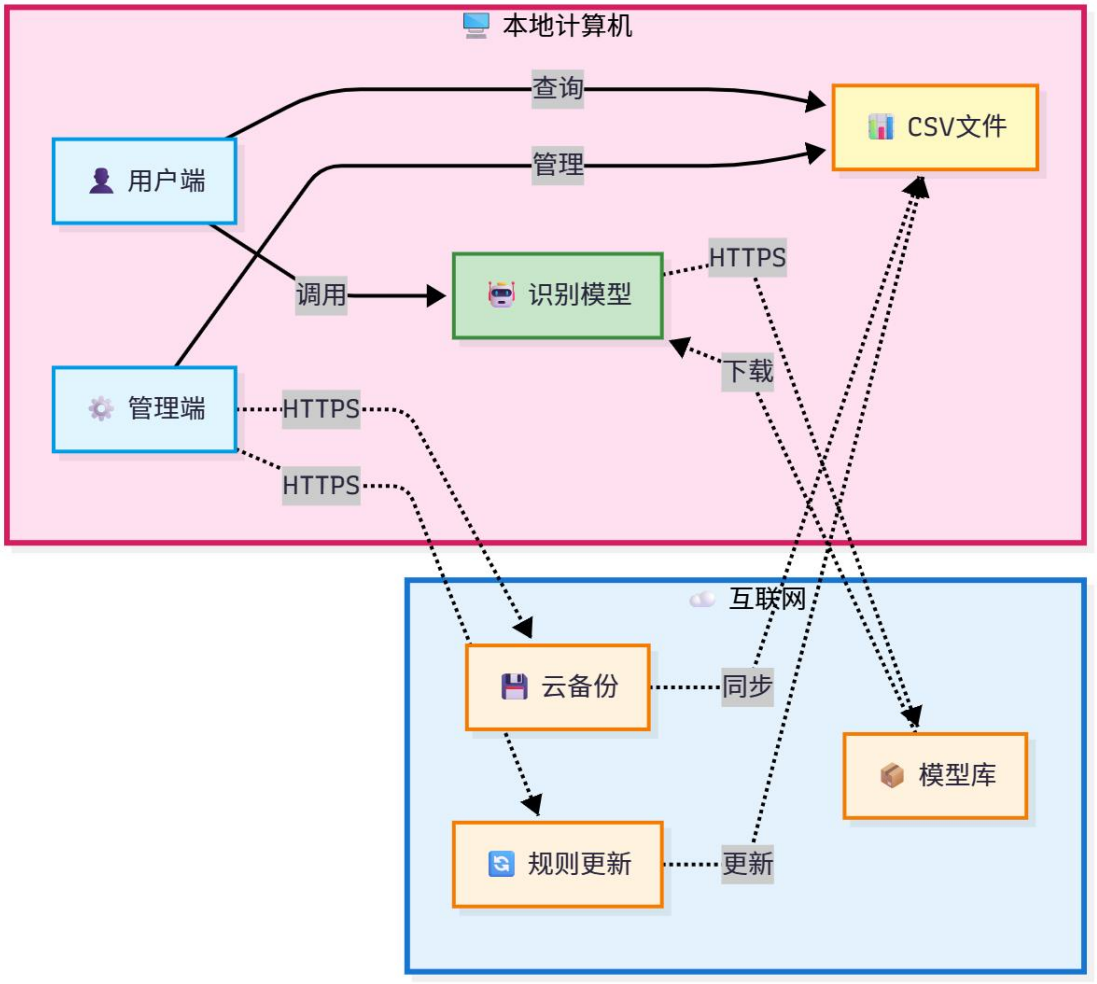


图-2：运行拓扑示意图

3.2 系统运行硬件环境

本系统的硬件环境如下：

- 客户机：普通 PC
  - CPU：双核 2.0GHz 及以上
  - 内存：双核 2.0GHz 及以上
  - 磁盘：1GB 可用空间（规则、日志、备份、模型文件）
  - 显示：分辨率≥1366×768，推荐 1920×1080
  - 外设（可选）：摄像头（若使用图片识别拍照输入）
- 兼容性说明
  - 普通名称查询与规则编辑对硬件要求较低
  - 图片识别若采用 CPU 推理，需适度提升 CPU/内存；若采用 GPU 推理（扩展），需支持相应驱动与 CUDA（不作为本期必需）

3.3 系统运行软件环境

- 操作系统：
  - Windows 10/11（推荐）
  - macOS 12+（验证通过后列入支持）
  - Linux（Ubuntu 20.04+，验证后列入支持）
  - 语言与运行时：
  - Python 3.10+（统一版本以便依赖管理）
- GUI 技术：
  - Tkinter（标准库，轻量优先）或 PyQt5/PySide6（增强 UI，需打包支持）
- 依赖库：
  - csv（标准库）
  - codecs/os/pathlib（标准库）
  - pandas（可选，用于导入导出和数据校验便捷）
  - opencv-python/pillow（扩展：图片读取与基础处理）
- 打包发布：
  - PyInstaller/Briefcase 等（任选其一，目标：免安装运行包）
- 字体与本地化：
  - 中文字体兼容；UTF-8 文件读写；
- 安装与权限：
  - 普通用户权限可运行；规则文件目录需读写权限；
  - 可选将程序安装于用户目录，避免管理员权限。

4 功能规格

我们采用面向对象分析作为主要的系统建模方法，使用 UML(Unified Modeling Language)作为建模语言。UML 为建模活动提供了从不同角度观察和展示系统的各种特征的方法。在 UML 中，从任何一个角度对系统所作的抽象都可能需要几种模型来描述，而这些来自不同角度的模型图最终组成了系统的映像。

Use Case 描述的是“actor”(用户、外部系统以及系统处理)是如何与系统交互来完成工作的。Use Case 模型提供了一个非常重要的方式来界定系统边界以及定义系统功能，同时，该模型将来可以派生出动态对象模型。

设计 Use-case 时，我们遵循下列步骤：

第一步，识别出系统的“actor”。Actor 可以是用户、外部系统，甚至是外部处理，通过某种途径与系统交互。重要的是着重从系统外部执行者的角度来描述系统需要提供哪些功能，并指明这些功能的执行者(Actor)是谁。尽可能地确保所有 Actor 都被完全识别出来。

第二步，描述主要的 Use Case。可以采取不断地问自己“这个 Actor 究竟想通过系统做什么？”来准确地描述 Use Case。

第三步，重新审视每个 Use Case，为它们下个详尽的定义。

功能总览（与用户约束对齐）：

- 名称查询：输入物品名称，系统基于预设规则返回垃圾类型与分类依据；
- - 结果展示：以清晰 UI 展示类型、依据，并可附带示例说明；
- - 规则编辑：管理员可新增、修改、删除规则；支持重名校验与冲突提示；
- - 数据存储：CSV 文件；运行时加载为 Python 字典；
- - 批量导入导出：从 CSV 导入规则，导出规则至 CSV；
- - 备份恢复：规则文件备份与一键恢复；
- - 图片识别（扩展）：加载图片或摄像头拍摄进行物品识别，联动规则返回分类；
- - 系统配置：规则文件路径、自动备份策略、UI 主题（可选）；
- - 日志审计（可选）：记录管理员操作与导入导出事件。

4.1 角色（Actor）定义

角色或者执行者（Actor）指与系统产生交互的外部用户或者外部系统。

4.1.1 普通用户

职责：

- 输入物品名称进行查询
- 查看垃圾类型与分类依据
- 可查看历史查询（可选）
- 可尝试图片识别（扩展）

权限：

- 查询权限
- 不可编辑规则（除非切换管理员登录）

4.1.2 管理用户

职责：

- 登录管理端
- 维护规则（增删改查）；
- 批量导入导出；
- 数据备份与恢复；
- 系统配置；
- 查看操作日志（可选）；

权限：

- 拥有规则编辑与存取的高权限操作；
- 可设置本地口令或简化登录（根据部署策略）

4.1.3 本地数据存储（csv）

作为与系统交互的“外部系统”视角：

- 响应系统的读写请求；
- 是系统外部的持久化实体；

交互内容：

- 加载时机：应用启动/手动刷新；
- 保存时机：规则变更提交/批量导入完成；
- 文件结构：见 4.6。

4.1.4 可选图像识别服务（扩展）

作为外部模块（离线）：

- 输入：图片；
- 输出：候选物品名列表+置信度；
- 供查询服务进一步匹配并给出分类；

部署：

- 本地模型文件+推理代码；
- 不依赖网络。

4.2 系统主Use Case图

网上招聘系统可以分为两个主要的组成部分，一个是客户端子系统，一个是管理端子系统。客户端子系统主要是指使用者通过登录招聘网站进行操作的功能，即垃圾分类功能。管理端子系统是招聘公司的管理人员登录、规则增删改查、批量导入导出、备份恢复、日志查看（可选）、配置管理等功能。系统的主 Use Case 图如图 B-3 所示。

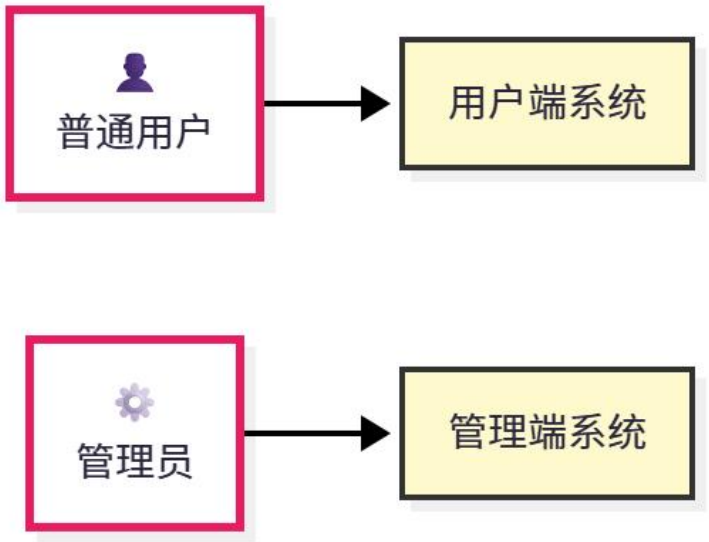


图 B-3：运行拓扑示意图

4.3 客户端子系统

子系统目标：为普通用户提供“尽可能少操作、尽可能快得到结果”的查询体验

功能，具体架构图如图 B-4 所示

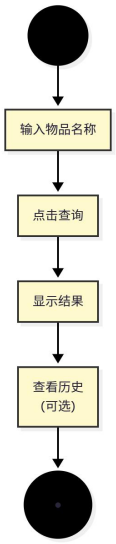


图 B-4：客户端信息架构图

F-C-1：名称查询与结果展示

- 输入项：物品名称（必填）
- 处理：字典查找（支持全称、同义词、大小写/空格/全角半角归一）
- 输出：垃圾类型（可回收/有害/厨余/其他）、分类依据（文本）
- 异常：未命中时提示可能的相近词、建议用户反馈（非在线场景可引导管理员添加）

F-C-2：分类依据说明

- 展现与可读性优化（段落、关键字高亮、示例）
- 可显示标准摘要或规则来源（如“XX 市生活垃圾分类指引条款 x.x”）

F-C-3：历史查询（可选）

- 最近 N 条查询项展示；可点击重查
- 清空历史（用户级）

F-C-4：图片识别（拓展）

- 选择图片/拍照后调用 CV 模块提取候选物品名
- 用户确认候选项或手动改写名称后进行最终查询

4.3.1 名称查询与分类结果展示

用户首先在界面中输入待查询的物品名称，例如"电池"。系统接收到输入后，会自动进行数据归一化处理，包括去除多余空格、统一大小写格式、可选的简繁体转换以及本地化符号标准化等操作，确保输入数据的规范性和一致性。

完成预处理后，系统利用预加载的 Python 字典数据结构进行高效查找，该实现具有 O(1)的平均时间复杂度，能够快速返回查询结果。如果名称在字典中精确命中，系统将渲染展示详细的结果卡片，清晰标明物品类型（如"有害"）及其判定依据（如"含重金属、有化学风险"等关键信息）。

作为可选的辅助功能，系统还会将此次查询记录写入本地历史文件，便于用户追溯查询记录，但需要注意的是这些历史记录不会纳入核心的 CSV 规则库。

当出现查询未命中的情况时，系统会自动启动备选处理机制：首先尝试通过近似匹配算法（包括编辑距离计算、前缀匹配或同义词索引等可选方式）寻找可能的候选项目。如果找到相关候选，系统会呈现候选列表供用户选择，同时引导用户联系管理员进行规则补充；如果经过多轮匹配仍无任何相关候选，系统则会明确提示"该物品未收录，建议联系管理员新增规则"，为用户提供清晰的后续操作指引。

整个用例执行完毕后，系统确保达成相应的后置条件：对于成功命中的查询，完整展示类型判定和依据说明；对于未命中的查询，提供友好的提示信息和后续操作建议。如图 A-1 所示。

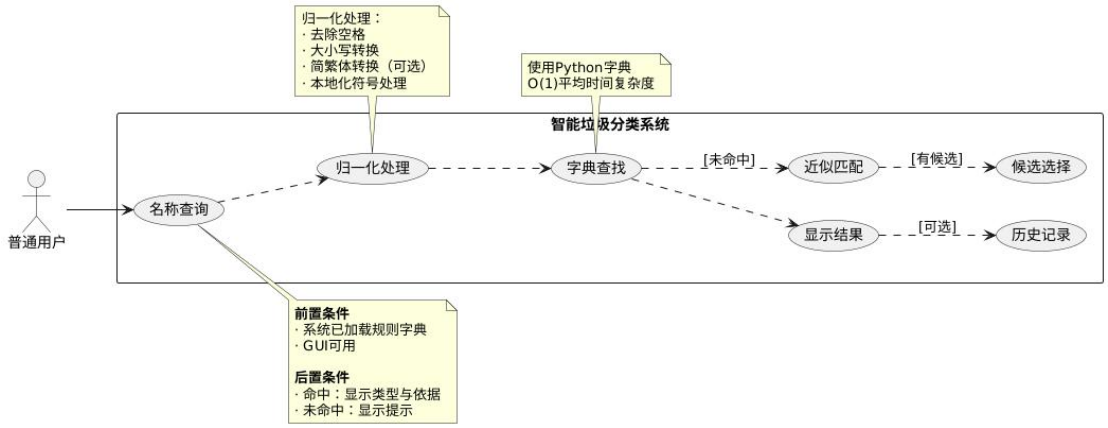


图 A-1：名称查询与分类结果用例图

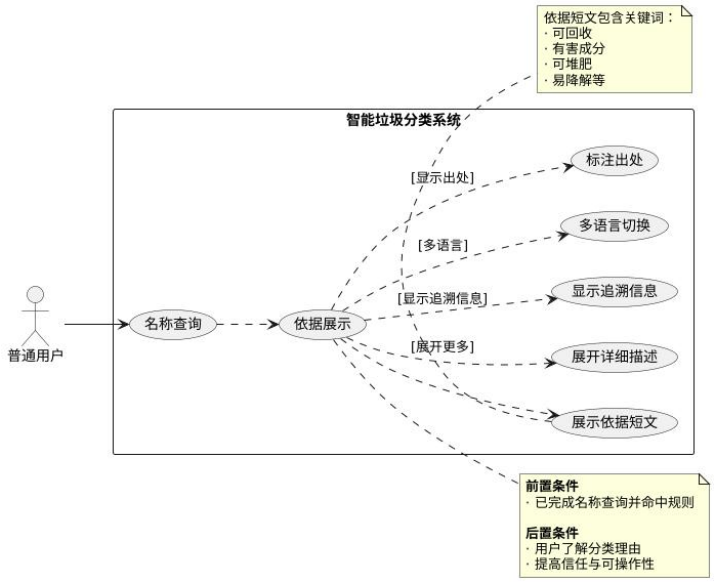
4.3.2 分类依据展示与解释

在用户完成名称查询并成功命中相关规则后，系统将启动依据展示功能，旨在通过清晰的信息呈现帮助用户理解物品分类的具体理由，从而增强用户对系统的信任度和操作体验。

用例的基本执行流程如下：系统首先展示简洁的依据摘要，其中突出显示关键分类标签，如"可回收"、"有害成分"、"可堆肥"等核心关键词，使用户能够快速把握分类要点。为进一步满足用户深入了解的需求，系统提供"展开更多"交互选项，点击后可查看完整的依据详细说明，必要时还可包含相关法规条款的简要引用，确保依据的权威性和完整性。

为保障信息的可追溯性，系统在依据展示区域明确标注"本条依据最后更新日期"和"规则版本号"两项关键元数据，方便用户核实信息的时效性和准确性。

在衍生功能方面，系统支持依据文本的多语言展示（可选功能），能够根据用户偏好切换不同语言版本的说明内容。同时，对于引用的外部法规或标准，系统会标注完整的出处链接信息；考虑到离线使用场景，这些链接以纯文本形式保存，暂不提供直接跳转功能，但仍为用户提供了进一步查证的线索。如图 A-2 所示。



4.3.3 历史查询与快速记录重查

在系统配置已开启历史记录功能的前提下，普通用户可通过历史查询功能快速访问以往的查询记录，有效提升重复查询的效率和操作便捷性。

用例的基本执行流程如下：用户首先打开历史查询面板，系统自动显示最近 N 条查询记录（默认设置为 10 条，支持用户自定义调整显示数量）。用户可通过点击任意一条历史记录，系统将立即触发对该条记录的重新查询，直接展示最新的查询结果，省去重复输入的步骤。同时，为满足用户清理需求，系统提供"清空历史"功能，支持用户一键清除所有历史查询记录。

在数据存储方面，历史记录独立于核心规则库，不会进入 CSV 规则文件，建议采用本地 JSON 格式或应用配置文件进行存储，确保历史数据与核心业务数据的有效分离。

特别注重用户隐私保护，所有历史记录仅保存在本机当前用户空间内，不会上传至任何服务器或云端，充分保障用户查询行为的私密性。这一设计使得用户能够在享受快速重查便利的同时，无需担心个人查询历史的外泄风险。

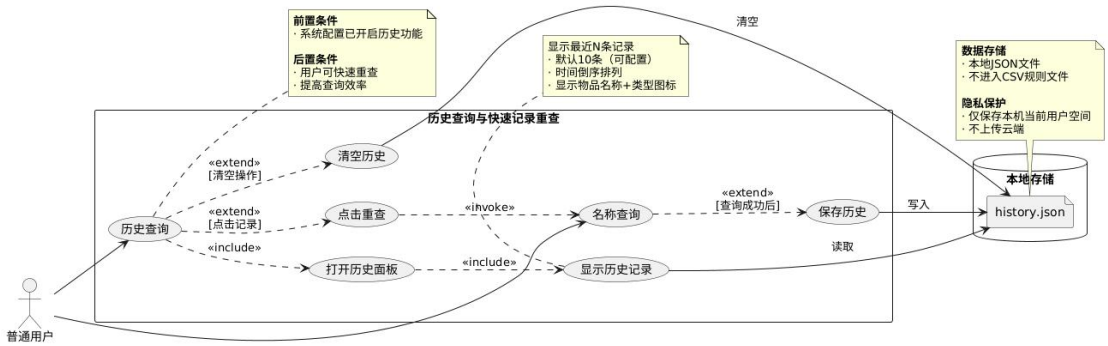


图 A-2：历史查询与快速记录重查用例图

4.3.4 图片识别（扩展）

在系统已启用图片识别模块并成功加载相关模型的技术前提下，普通用户可通过图片识别查询功能，快速获取物品的分类信息，最终达成与文本查询相同的分类判定目标。

用例的基本执行流程如下：用户首先通过系统界面选择本地图片文件或直接开启摄像头进行实时拍摄，获取待识别物品的视觉信息。系统接收到图像输入后，计算机视觉模块将自动执行特征提取与模型推理，返回 Top-K 个最可能的候选物品名称及其对应的置信度评分。用户在候选列表中选择最匹配的物品名称，或对系统推荐的名称进行必要编辑后确认提交。系统随即基于最终确定的物品名称执行标准的名称查询流程，完整展示该物品的分类结果及相关依据。

针对可能出现的异常情况，系统设计了完善的失败处理机制：当模型输出的最高置信度低于预设阈值时，系统会明确提示用户当前识别结果可靠性不足，并建议改用文本查询方式以获得更准确的结果；若系统检测到图片识别模块未正确加载或模型文件缺失，则会提示用户联系管理员在系统配置中启用并加载必要的识别模型。如图 A-3 所示。

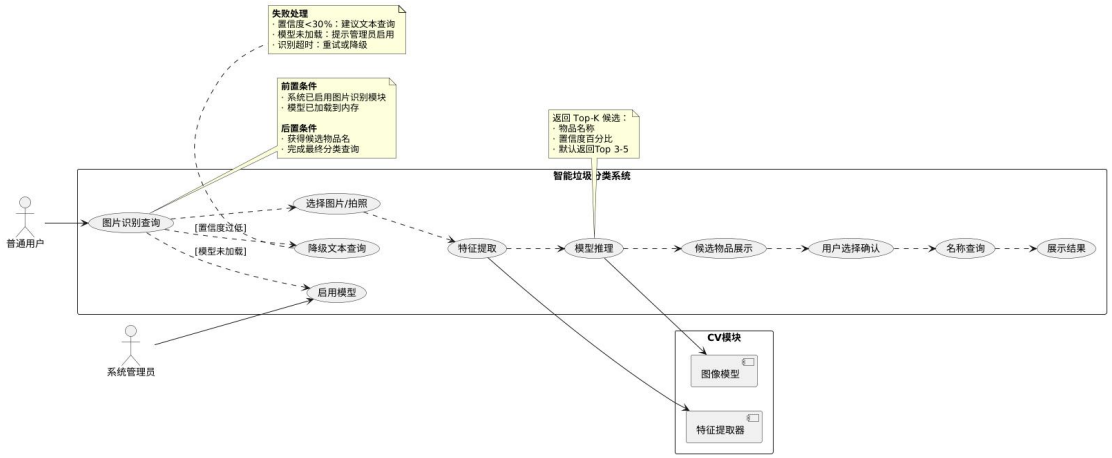


图 A-3：图片识别用例图

4.4 管理端子系统

为了保障规则数据质量与可控性，提供高效的编辑、批量处理、备份与恢复能力。我们构造了管理端子系统，主要架构如图 B-5 所示。

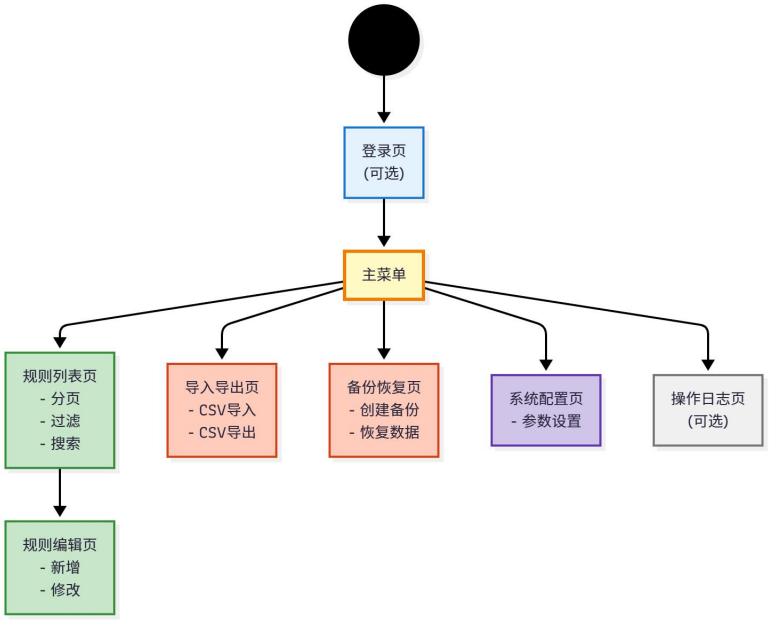


图 B-5：管理端信息架构示意图

4.4.1 登录管理

在系统配置中已启用管理员口令验证机制的前提下（该功能默认可关闭，以简化单机使用场景），管理员需要通过登录流程来获取系统管理权限，成功登录后将进入管理端主界面。

用例的基本执行流程如下：管理员首先在登录界面输入预设的用户名与对应口令，或在单机简化模式下使用本地 PIN 码进行身份验证。系统接收到凭证信息后，执行严格的校验流程，验证通过后立即跳转至管理端主页面，授予相应的系统管理权限。

在安全机制设计方面，系统采用多重防护措施：首先，对于本地存储的口令信息，系统仅保存其不可逆的哈希值，确保即使数据泄露也无法还原原始口令；其次，系统支持可配置的自动锁定策略，当连续认证失败次数达到预设阈值（如 N 次）时，自动锁定账户以防止暴力破解；此外，为兼顾便利性与安全性，系统提供可选的"记住登录"功能，通过本机加密方式保存登录状态，但会明确提示用户谨慎使用此功能。如图 A-4 所示。

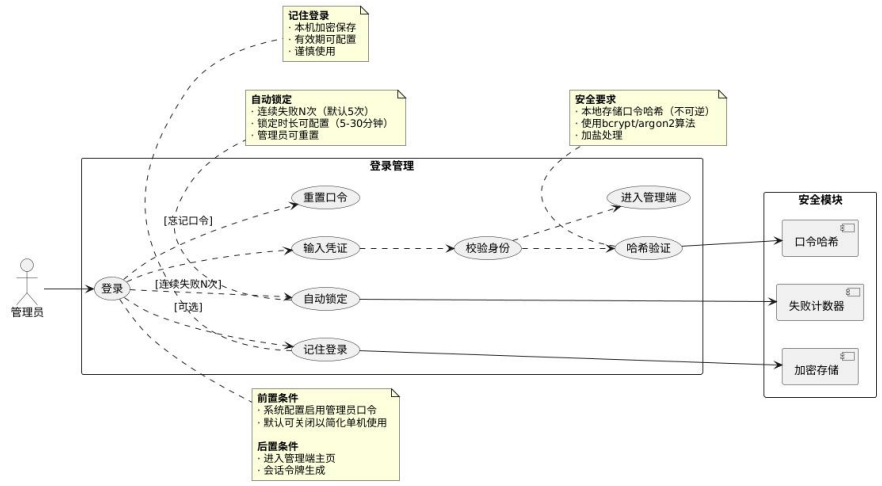


图 A-4：登录管理用例图

4.4.2 规则管理（增/删/改/查）

管理员在成功登录后，可对垃圾分类规则进行全生命周期管理。系统提供分页列表展示所有规则，支持按类型、关键词和更新时间进行筛选。管理员可以新增规则，填写物品名称、选择分类类型、填写依据等必要信息，系统会进行重名检测和完整性校验。规则修改和删除功能允许管理员更新现有条目或进行软删除（可恢复）与硬删除（需确认）操作。

系统具备冲突管理机制，自动检测同名不同类的规则冲突，并提供合并或覆盖策略。规则变更可立即生效或通过可配置的"草稿-发布"流程生效，所有更改都会持久化保存到 CSV 文件，用户端刷新后即可看到更新。整个流程确保规则管理的完整性和数据一致性，具体字段规范参考相关数据结构文档。如图 A-5 所示。

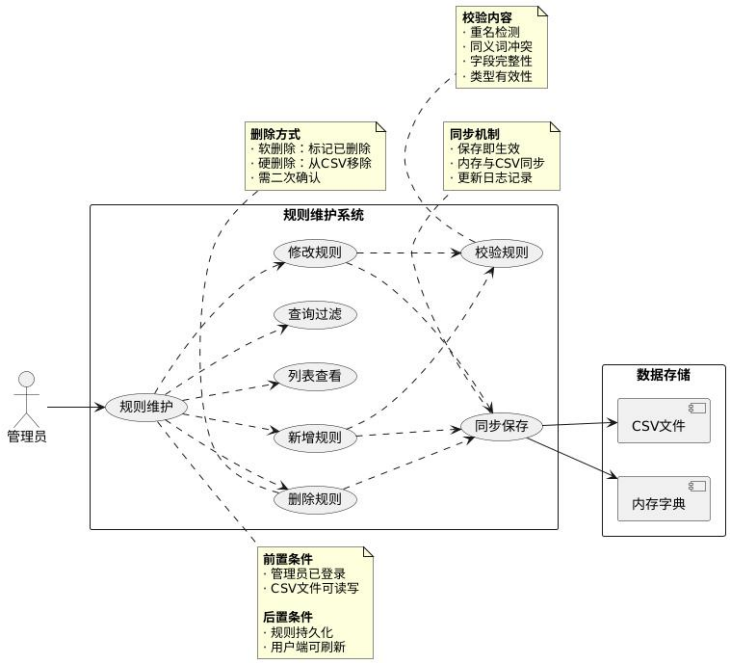


图 A-5：管理员规则管理用例图

4.4.3 批量导入导出（CSV）

管理员通过批量导入导出功能实现规则数据的高效批量操作。在成功登录系统且 CSV 路径可

用的前提下，管理员可以执行数据导出操作，系统会自动从内存字典生成包含完整字段的 CSV 文件，文件名附带时间戳以便区分。

导入功能允许管理员选择 CSV 文件进行批量数据更新。系统会先对文件进行预检查，验证编码格式、标题列和必填字段的完整性，随后提供覆盖和合并两种导入策略。覆盖模式会以导入文件为准替换现有数据，合并模式则只新增和更新差异项，冲突项目会进入人工处理队列。导入完成后系统会生成详细报告，统计成功与失败的条目数量并列明失败原因。

为保障数据安全，系统在导入前会自动创建备份快照，防止格式错误导致数据损坏。同时提供可选的一键撤销功能，能够回滚最近一次的导入操作，确保数据变更的安全可控。如图 A-6 所示。

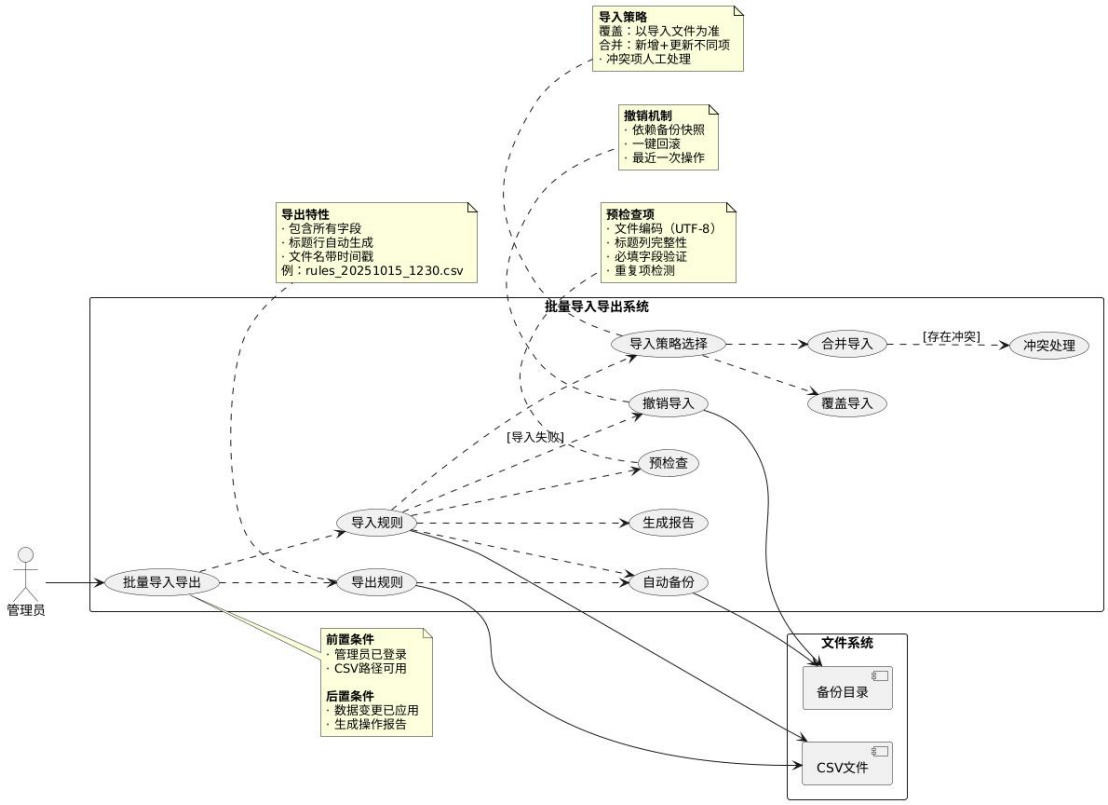


图 A-6：批量导入和导出用例图

4.4.4 数据备份与恢复

管理员通过备份恢复功能确保系统数据安全可追溯。在 CSV 路径和备份目录可用的前提下，管理员可执行手动备份操作，将当前规则文件复制到备份目录并添加版本时间标识。系统在重要操作如批量导入、大量变更或删除前会自动创建备份，防止意外数据丢失。

恢复功能允许管理员选择特定备份版本一键还原规则数据，恢复前系统会提示保存当前未保存的变更。完成恢复后，系统自动重新加载规则字典并执行完整性校验，确保数据恢复的准确性和系统稳定性。整个备份恢复机制为规则管理提供了可靠的数据安全保障。如图 A-7 所示。

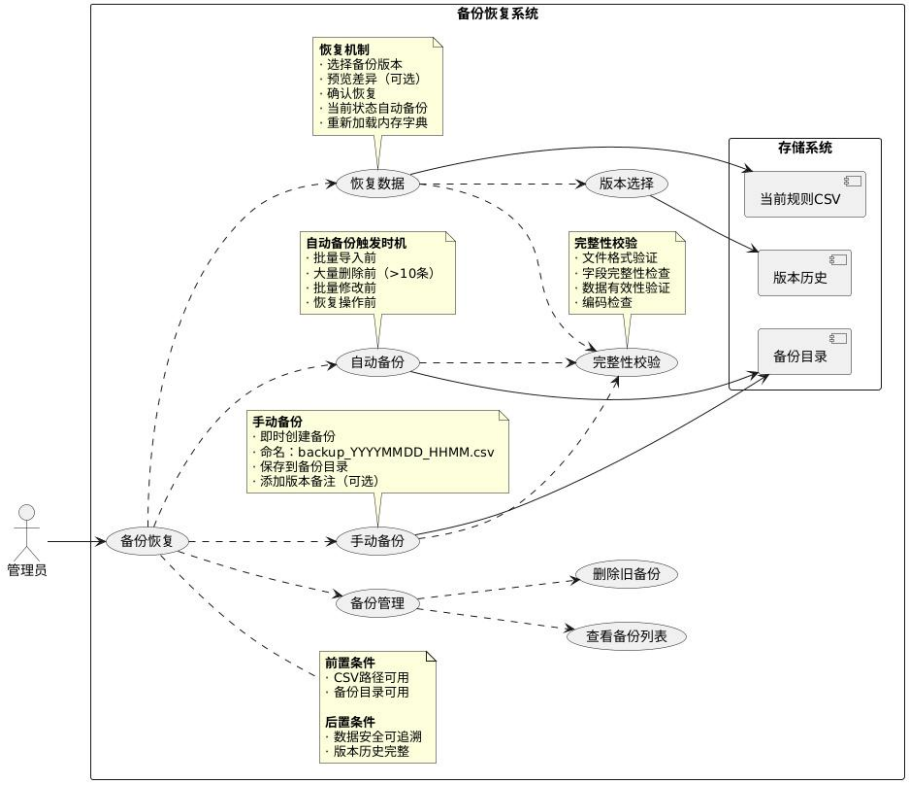


图 A-7：备份恢复系统用例图

4.4.5 操作日志与审计（可选）

管理员通过日志审计功能监控系统操作记录。在已启用日志记录的前提下，系统自动记录关键事件包括登录、规则增删改、导入导出及备份恢复等操作。管理员可按时间、操作人员和操作类型对日志进行筛选查询，并支持将审计结果导出为 CSV 或文本格式报告。

为确保隐私与合规性，日志系统仅记录必要的操作信息，严格避免记录敏感数据如口令原文等。通过完整的日志记录和灵活的查询导出功能，为系统操作提供可靠审计依据，满足运维管理和合规要求。如图 A-8 所示。

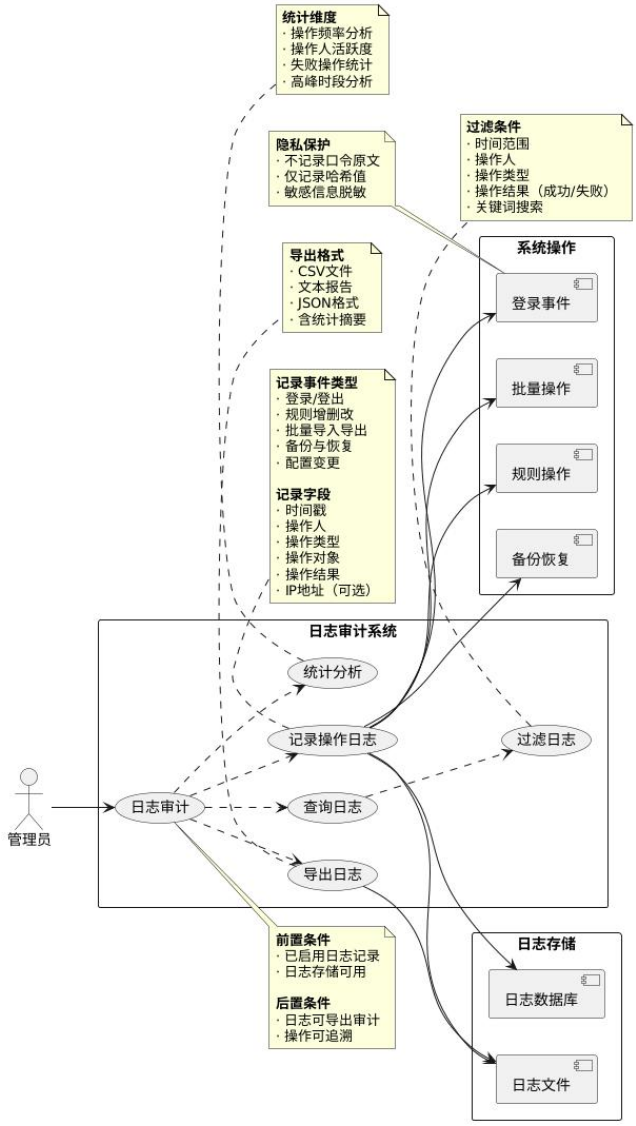
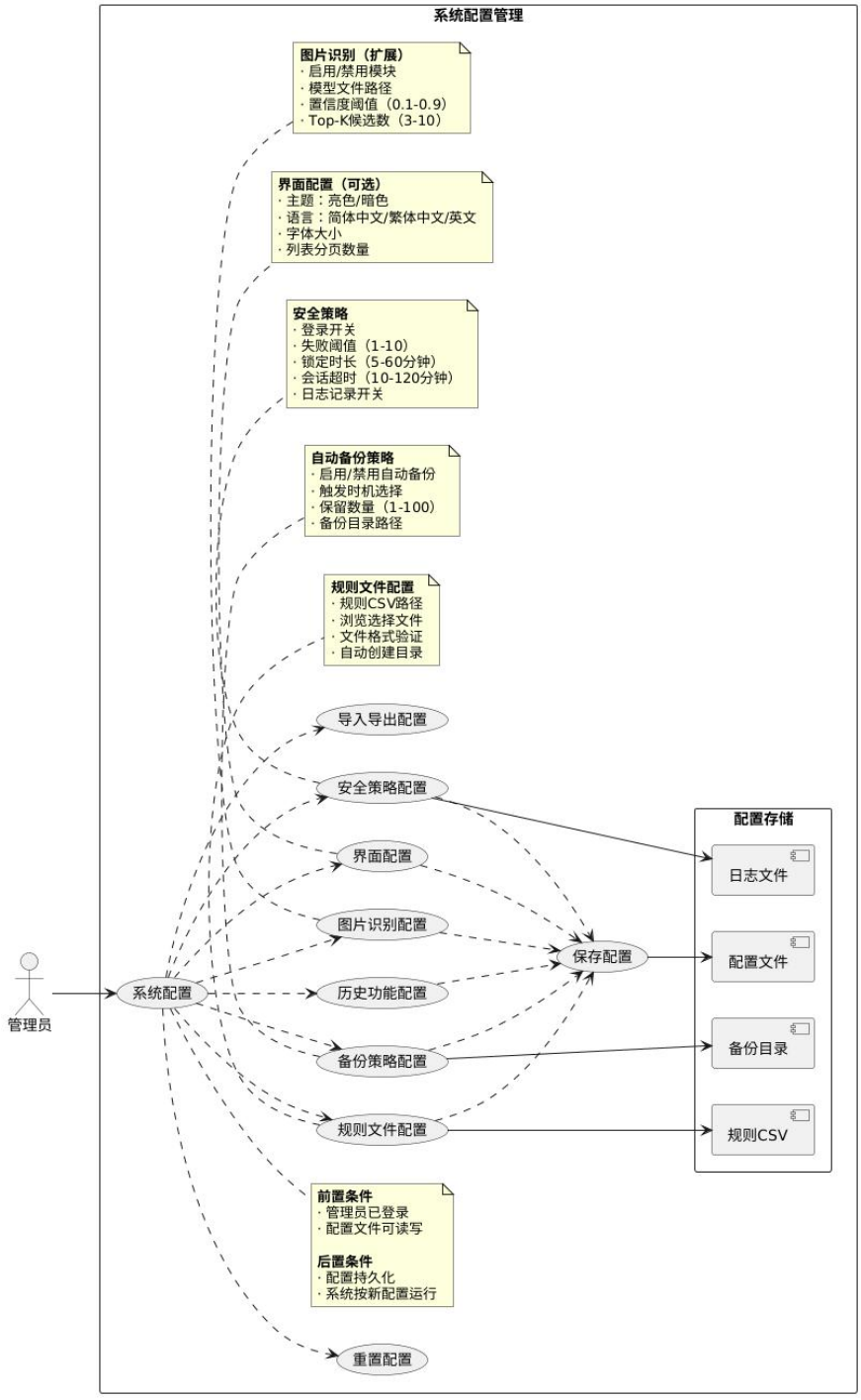


图 A-8：操作日志与审计用例图

4.4.6 系统配置

管理员通过系统配置管理功能对各项系统参数进行个性化设置。系统提供规则文件路径的可视化浏览选择功能，支持自动备份策略配置，包括开关状态、触发时机和备份保留数量设置。管理员可根据需要调整 UI 主题与语言选项，管理图片识别模块的启用状态和模型路径，控制历史查询功能的开



4.5 典型业务流程

普通用户的查询活动始于输入物品名称，系统对输入内容进行归一化处理后执行字典查找。若成功命中规则，系统直接展示分类结果与判定依据，流程结束；若未命中则启动近似匹配寻找候选项，存在候选时由用户选择后展示结果，若无任何候选则提示用户联系管理员添加新规则，流程终止。如图 B-6 所示。

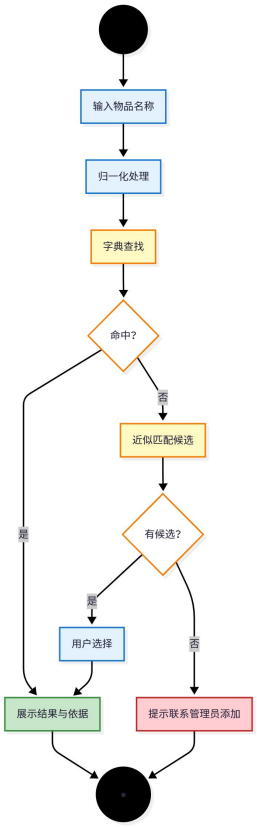


图 B-6：普通用户查询活动示意图

管理员通过登录验证后进入规则管理界面，可执行新增、修改或删除规则操作。系统对变更内容进行冲突校验，通过后保存至 CSV 文件并刷新内存数据，同时记录操作日志。批量导入时系统先进行文件预检，管理员选择导入策略后执行导入操作，生成详细报告并保留撤销选项，确保数据变更的安全可控。

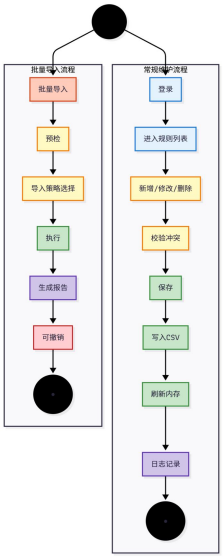


图 B-7：管理员维护活动示意图

用户通过选择图片或实时拍照启动识别流程，系统加载预训练模型进行推理分析，生成候选物品列表。用户从候选中确认或修正物品名称后，系统执行标准名称查询流程，最终展示完整的分类结果。这一扩展功能为用户提供了除文本输入外的另一种便捷查询方式。如图 B-8 所示。

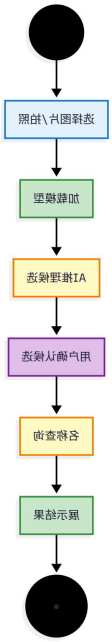


图 B-8：图片识别活动示意图

4.6 数据结构与存储规范（CSV+内存字典）

为保证与“使用 Python 字典保存规则（物品名-类型-依据）”的约束一致，定义如下数据结构。

4.6.1 CSV 字段规范（建议）

a) id：规则唯一 ID（可选，便于追踪）
b) item_name：物品主名称（必填，唯一约束可配置）
c) synonyms：同义词（可选，多值以分号或逗号分隔，需约定统一分隔符）
d) category：垃圾类型（必填，枚举：recyclable/harms/kitchen/other；GUI 显示为中文：可回收/有害/厨余/其他）
e) basis：分类依据（必填，简明扼要）
f) source：依据来源（可选，如“XX 市指引 2023 版 条款 x.x”）
g) region：适用地区（可选，默认“通用”）
h) version：规则版本号（可选）
i) updated_at：最后更新时间（ISO8601）
j) updated_by：最后更新人（可选）

4.6.2 CSV 文件要求

a) 编码：UTF-8（含 BOM 可选，不推荐）
b) 标题行：第一行字段名
c) 分隔符：逗号（如字段中包含逗号，要求加引号；或统一使用分号分隔 synonyms）
d) 空值：使用空字符串表示，不使用 N/A 等特殊字面
e) 文件名：rules.csv（默认），导出/备份带时间戳

4.6.3 内存字典结构

加载策略：启动时读取 CSV→标准化→构建多映射结构
建议结构：
a) 主字典：key=标准化的 item_name；value=对象{category, basis, source, region, version, updated_at...}
b) 同义词索引：key=标准化同义词；value=指向主项 item_name 或直接 value 对象
标准化规则：
a) 去前后空格、统一大小写（必要时）
b) 全角半角转化
c) 可选：简繁转换（若启用）
冲突处理：
a) 加载时发现同名不同类：记录冲突清单并提示管理员；
b) 运行时新增/修改执行同名校验；
持久化：
a) 每次变更提交后写回 CSV；
b) 写入采用临时文件+原子替换（避免中断损坏）。

4.6.4 示例

条目：item_name=电池，category=有害，basis=含重金属及腐蚀性成分，应投有害垃圾专用回收箱
条目：item_name=纸箱，category=可回收，basis=纸制品可回收，再利用价值高
说明：示例仅用于理解字段含义，具体内容由业务管理员维护于 CSV 文件中。

5 性能需求

根据用户对本系统的要求，确定系统在响应时间、可靠性、安全等方面有较高的性能要求。

5.1 界面需求

系统的界面要求如下：
c) - 字体清晰、对比度良好；
d) - 错误提示友好、可定位；
艺术风格：
a) 简洁明快，色彩适度，类型标签配色直观（如可回收=蓝、有害=红、厨余=绿、其他=

灰）；
b) 统一控件样式，动效轻量不干扰操作。
c) 字体清晰、对比度良好；
d) 错误提示友好、可定位；
e) 统一控件样式，动效轻量不干扰操作。

5.2 响应时间需求

a) 启动时间：≤3 秒（不含首次打包后依赖预热）
b) 名称查询：平均≤1 秒（本地字典查询，通常毫秒级）
c) 列表分页切换：≤1 秒
d) 规则保存：≤2 秒（含 CSV 写入）
e) 批量导入：取决于数据量；1 万条以内≤10 秒（含校验与索引重建）
f) 图片识别（扩展）：单张 CPU 推理≤3 秒（以具体模型为准）

5.3 可靠性需求

稳定运行：7×24 小时不当机（桌面应用以用户使用时段计）
数据完整性：
a) 写入采用临时文件与校验，防止中断导致文件损坏；
b) 自动备份：导入、批量删除等操作前强制创建快照；
并发访问：
a) 单机单实例；不支持多实例同时写同一 CSV（如发生，多实例写保护或锁文件提示）
异常处理：
a) 文件不可写：提示权限问题并引导用户修改路径或权限；
b) CSV 格式错误：给出行号与错误类型；
c) 模型未加载：提示配置与路径设置；

5.4 开放性需求

a) 数据开放：CSV 为开放格式，便于与其他系统/流程协作；
b) 模块开放：图片识别模块以接口封装，后续可替换模型；
c) 配置开放：规则路径、自动备份策略、主题等可配；
d) 国际化（可选）：支持多语言文本资源。

5.5 可扩展性需求

业务扩展：
a) 新增垃圾子类/子标签（如“大件、织物”）；
b) 地区化差异覆盖（region 字段与多 CSV 库机制）；

技术扩展：
a) 增加 Web 服务接口（REST，本期不实现）；
b) 远程同步规则库（本期不实现）；
算法扩展：
a) 同义词自动挖掘（基于词向量，本期不实现）；
b) 端侧轻量模型热更新（扩展）。

5.6 系统安全性需求

权限管理：
a) 管理端登录可配；至少在公共场所部署时必须开启；
b) 重要操作要求二次确认；
数据安全：
a) CSV 写入前备份，支持回滚；
b) 日志不记录敏感信息；
防误操作：
a) - 删除/导入前确认并提示影响范围；
b) - 撤销机制（最近一次导入）；
文件安全：
a) - 防止路径穿越（用户仅能在允许的目录树中操作）；
b) - 防止覆盖误写（重名提示与自动改名）；
隐私：
a) - 历史查询仅本机保存，允许一键清除。

5.7 可维护性与可测试性需求

可维护性：
a) 模块化代码结构（GUI/服务/数据）；
b) 明确的数据读写接口与校验流程；
c) 代码与配置分离；
可测试性：
a) 单元测试：查询服务、规则服务、CSV 读写、冲突检测；
b) 集成测试：导入导出、备份恢复、权限控制；
c) 手工测试：GUI 流程、错误提示；
d) 覆盖率目标：核心逻辑≥80%。

6 产品提交

提交产品为：
a) 应用系统软件包（打包后的可执行）

b) 初始规则 CSV 数据（含若干高频物品与依据）
c) 系统使用维护说明文档（用户手册、管理员手册）
d) 测试用例与测试报告（功能/性能/异常）
e) 变更记录与版本说明
f) 交付介质：电子交付；如需物理介质可提供 U 盘/光盘（按合同）

7 实现约束

系统的实现约束如下：
a) 操作系统为Ｗｉｎ２０００
b) 技术要求（硬约束）：
使用 Python 字典存储垃圾分类规则（物品名-类型-依据），运行期数据源来自 CSV；
开发桌面应用，提供文本输入与结果展示；
c) 限定条件：
系统需包含分类规则编辑功能（管理员可添加/修改规则）；
输入项：物品名称；输出项：垃圾类型、分类依据；
数据存储：CSV 文件存储分类规则；
d) 扩展建议：
支持图片识别垃圾类型（集成简单 CV 模型）；
e) 环境与依赖：
Python 3.10+；
GUI 优先 Tkinter（也可选 PyQt/PySide，需考虑打包体积与跨平台）；
不引入服务器端与数据库（本期）；
f) 打包与分发：
使用 PyInstaller 等工具，目标为“零配置运行”；
合规：
分类依据文本需可追溯来源（如有官方条款引用）；
依据与分类标准的地区差异由业务方提供最终判定表。

备注

本文档用词力求与模板一致，章节结构与内容深度覆盖你提出的全部要求，并为图像识别扩展、规则冲突、批量导入导出、备份恢复、日志审计等提供了可落地的需求框架。
- 如你有特定城市/地区的垃圾分类细则与条款编号，可在“basis/source/region”字段中进一步固化，或在“系统配置”中为地区差异预置多个规则集。
- 下一步建议：我可以根据本 SRS 继续输出“概要设计说明书”（模块结构、接口定义、类图/时序描述）与“测试方案/用例集”，并可按你选择的 GUI 框架给出初始项目骨架与 CSV 样例。