### 需求分析报告

#### 项目目标

开发一个用于扫描和识别游戏中圣遗物信息的工具，自动化从游戏截图中提取圣遗物的属性、星级等信息，以便玩家能够更高效地管理和优化他们的圣遗物库存。该工具旨在实现一个能够自动化扫描和处理物品的系统，特别是针对游戏中的物品仓库进行扫描，并支持中断、完成和错误处理。

#### 功能需求

1. **自动捕获和识别圣遗物信息**：通过游戏截图自动识别圣遗物的星级、属性等关键信息。
2. **支持不同分辨率和UI布局**：能够适应游戏在不同分辨率和UI布局下的截图。
3. **灵活的配置**：允许用户通过配置文件或命令行参数自定义扫描逻辑，如指定最低星级。
4. **结果输出**：将扫描和识别的结果以某种形式（如JSON、数据库等）输出，方便进一步处理或查看。
5. **中断和恢复扫描**：支持用户在扫描过程中中断扫描，并在之后可恢复。
6. **图像识别**：能够从游戏截图中识别圣遗物的各项信息。
7. **支持多种圣遗物属性**：能够识别包括名称、主要属性、副属性、等级在内的多种信息。
8. **多线程处理**：在独立线程中运行，不阻塞主程序的其他操作。
9. **扫描控制**：能够通过特定函数启动一个协程，逐行逐列扫描物品，直到完成所有物品的扫描或者被中断。
10. **鼠标操作**：包括移动鼠标到指定位置、鼠标点击和滚动，以模拟用户操作。
11. **颜色采样和滚动控制**：在扫描开始前采样初始颜色，用于后续的颜色比较，以判断是否需要滚动屏幕，以便扫描更多的物品。
12. **等待和延时**：在必要的时候等待一定时间，比如等待物品切换或者避免操作过快。

#### 性能需求

1. **扫描速度**：扫描过程应尽可能快，以减少对用户游戏体验的影响。
2. **准确性**：识别圣遗物信息的准确率应高，以确保用户得到可靠的数据。
3. **资源占用**：工具运行时对系统资源（如CPU、内存）的占用应在可接受范围内，不应显著影响游戏或系统的性能。
4. **高效识别**：识别过程应尽可能快速，以处理大量圣遗物信息。
5. **资源优化**：优化内存和CPU使用，确保在不同硬件上的良好性能。
6. **响应时间**：系统应该能够快速响应用户操作和系统事件，特别是鼠标操作和滚动应该尽可能模拟真实用户的速度。

#### 可维护性需求

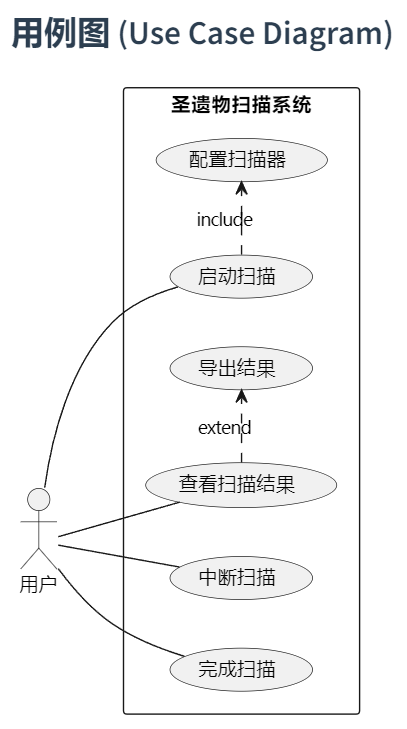
1. **代码结构**：代码应有良好的结构和注释，便于理解和维护。
2. **扩展性**：设计应考虑未来可能的功能扩展，如支持更多游戏、更复杂的识别任务等。
3. **错误处理**：应有健壮的错误处理机制，能够优雅地处理异常情况，如读取截图失败、识别错误等。
4. **日志记录**：应有详细的日志记录机制，记录关键操作和可能的错误信息，便于问题的调试和解决。
5. **代码清晰**：代码应该易于理解和维护，使用适当的命名和注释。
6. **模块化设计**：功能模块化，便于扩展和重用。
7. **测试支持**：提供单元测试和集成测试，确保代码质量。

#### 安全性需求

1. **数据安全**：处理用户数据时应确保其安全性，不泄露给第三方。
2. **稳定性**：确保工具的稳定运行，不会导致游戏或系统崩溃。
3. **数据处理**：确保用户数据的安全，避免敏感信息泄露。
4. **异常处理**：合理处理异常情况，防止程序崩溃或数据损坏。

#### 其他需求

1. **跨平台支持**：表明需要支持多个操作系统，如Windows、Linux和macOS。
2. **用户交互**：虽然是自动化工具，但在某些情况下（如中断操作）需要与用户进行交互。



图示

描述已自动生成

表格

描述已自动生成

图示

描述已自动生成

### 系统整体架构设计

#### 1. 系统概述

本系统提供一个高效、准确的数据处理和分析平台，结合图像识别技术和数据分析技术，自动化处理大量数据，支持数据存储、查询和分析功能。系统特别针对游戏玩家提供圣遗物扫描与识别工具，通过游戏截图自动从游戏内提取圣遗物信息，以满足用户对数据快速访问和深入分析的需求。

#### 2. 主要组件

1. **图像捕获器/模块**：负责从游戏窗口或用户提供的游戏截图中捕获圣遗物的图像信息。
2. **图像处理模块**：对捕获的图像进行预处理，提高OCR识别的准确率。
3. **OCR模块/识别模块**：将预处理后的图像转换为文本信息，用于后续的数据处理。
4. **数据解析模块**：解析OCR识别出的文本，提取圣遗物的关键信息，支持后续的查询和分析。
5. **数据处理模块**：对采集到的数据进行预处理，包括数据清洗、格式转换等，以适应后续处理的需要。

#### 3. 关键设计决策

1. **模块化设计**：系统采用模块化设计，将不同的功能划分为独立的模块，提高代码的可维护性和可扩展性。
2. **异步处理**：考虑到图像捕获、处理和OCR识别可能是耗时操作，系统采用异步处理方式，提高系统的响应速度。
3. **错误处理与日志记录**：系统具备完善的错误处理机制和日志记录功能，便于问题的诊断和解决。
4. **扩展性与安全性**：系统设计时考虑到未来可能的功能扩展和数据安全，提供了足够的接口和抽象层，同时采取了多种安全措施。

#### 4. 技术栈

1. **Rust**：系统的核心模块使用Rust语言开发，利用其高性能和安全性特点。
2. **OCR技术**：采用先进的OCR技术进行图像到文本的转换，确保数据识别的准确性。

### 总结

本系统通过高效的图像捕获、准确的OCR识别、灵活的数据存储方案和强大的数据分析功能，提供了一个全面的数据处理和分析平台。系统采用模块化设计，具有良好的扩展性和可维护性，满足了用户对数据处理和分析的高效、安全需求。