需求分析报告

1. 项目概述

1.1项目背景

随着二次元文化的普及，手办作为一种具备收藏价值和艺术价值的产品，受到了广大二次元爱好者和收藏家的追捧。据统计，2023年全球手办市场规模已达到数十亿美元，且仍在持续增长。然而，目前针对手办领域的垂直社区较少，现有平台多以综合性内容为主，无法满足手办爱好者对深度测评、高清图片分享、知识学习以及交流互动的强烈需求。在这样的市场环境下，我们希望打造一个以二次元手办为核心主题的垂直社区，专注于为用户提供高质量的测评内容、丰富的交流渠道和系统化的学习资源。

1.2项目目标

本项目是一个以二次元手办为主题的垂直社区网站，目标是为手办爱好者提供一个集交流、测评、教学于一体的综合平台。主要用户为手办爱好者、收藏家及相关从业者。平台旨在通过提供丰富的内容和互动功能，打造手办领域的领先社区。

1. 用户需求

2.1用户类型

* 普通用户：对手办感兴趣、主要以浏览、学习为主，喜欢分享周边图片、动态、与其它用户互动。
* 内容创作者：有一定影响力和专业知识的用户，致力于创建高质量内容，热衷于分享手办评测、教程或制作过程。
* 社区管理员：平台的运营人员，负责社区管理和内容审查，保障社区的和谐氛围，维护内容质量。
* 潜在合作伙伴：手办品牌方、零售商、活动主办方，希望通过平台推广产品或合作。

2.2用户功能需求

2.2.1普通用户

1. 浏览首页推荐内容
2. 参与社区讨论，发布动态或提问
3. 学习手办相关知识

2.2.2内容创作者

1. 上传视频、图片、发布专业测评内容
2. 管理自己的内容，如编辑、删除作品
3. 维护个人主页，与粉丝互动

2.2.3社区管理员

1. 审核用户发布的动态、视频、图片等内容
2. 管理违规信息，如删除不良内容、封禁用防护
3. 设置平台规则和公告
4. 功能需求

3.1主页

* + 热门内容推荐：系统根据用户兴趣，浏览历史推荐热门视频、测评、动态等内容。
  + 搜索功能：用户可通过关键词搜索手办测评、图片、视频或用户动态。支持过滤条件，如按类别、品牌、发布时间排序等。

3.2视频/图片

* + 视频播放与图片展示：用户可查看手办图片或播放手办测评视频。视频支持弹幕功能，图片支持多图滑动预览。
  + 点赞与收藏：用户可对喜欢的额视频或图片点赞，并收藏到个人收藏夹中。
  + 评论与回复：视频或图片下方有评论区，永魂可在该区域进行评论与回复。

3.3测评

* + 手办测评发布：内容创作者可发布手办开箱、测评文章或视频，支持图文混排。
  + 评分系统：用户可以为手办测评分数，如外观、质量、性价比等，评分结果已平均分形式展示在测评详情页。

3.4新手课堂

* + 新手知识学习：体功手办入坑指南、品牌介绍、保养技巧等内容，内容以图文或短视频形式提供。
  + 收藏与进度管理：用户可收藏学习内容，并记录学习进度，如下次回来时跳转到上次观看处。

3.5社区

* + - 动态发布：用户可发布图文动态，分享手办照片、心得等。
    - 话题讨论：支持话题标签，（如#开箱心得、#新品发布），方便用户围绕话题展开讨论。
    - 用户私信：用户之间可发送私信进行一对一交流。

3.6管理模块（管理员专属）

* + 内容管理：审核用户发布的动态、测评、视频、评论等内容。
  + 公告管理：发布平台公告，并设置首页弹窗展示。

1. 非功能需求

4.1性能需求

1. 网站的首页响应时间应小于2秒
2. 支持至少1000名用户同时在线访问

4.2可用性需求

* + 1. 界面设计符合手办爱好者审美习惯
    2. 系统在操作错误是提供详细的错误提示信息

4.3安全性需求

用户敏感数据如密码可加密存储

4.4可扩展性需求

系统支持后续增加新模块时不影响现有功能。

1. 技术需求

5.1前端

* 1. **Vue 3**：作为前端框架，使用其响应式特性、组件化开发、Vue Router、Vuex等功能来构建单页面应用。
  2. **Vue Router**：用于处理单页应用的路由管理，确保用户可以在不同模块（如主页、视频/图片、测评等）之间进行导航。
  3. **Vuex**：用于管理应用的状态，尤其在处理用户登录、用户数据、社区帖子等时，可以提供全局状态管理。
  4. **Element UI**：用于开发符合设计需求的UI组件，提高开发效率。

5.2后端

* 1. **Django**：作为后端框架，提供可靠的数据库管理、用户认证、RESTful API、后台管理界面等功能。
  2. **Django REST Framework (DRF)**：用于构建后端API，支持前端Vue应用的数据交互。

5.3数据库

**数据库管理系统：**sqlite

**具体设计：**

5.4接口和数据

5.5缓存机制

**Redis**：用于实现数据缓存，提升高并发时系统的响应速度，避免频繁访问数据库。可用于缓存用户登录状态、热门帖子、评论等数据。

5.6前后端通信

1. 项目风险分析

6.1技术风险

1. Redis或SQLite性能不足，无法满足高并发需求。
2. 系统性能优化不足，用户访问高峰时页面加载缓慢。

6.2资源风险

开发设备或工具（如服务器）配置不足，影响项目运行。