武汉大学

计算机学院实训

photoX 项目需求说明书

项目成员： _____

撰 写 人： _____ _____年___月__日

评 审 人： _____ _____年___月__日

武汉凡诺软件技术有限公司

2025 年

修订记录

日期	修订版本	修改章节	修改描述	修订人

目 录

关键词：

摘　要：

缩略语清单：<对本文所用缩略语进行说明。>

缩略语	英文全名	中文解释

1. 简介

1.1 编写目的

对软件需求的充分理解对于软件开发工作的成功至关重要。需求说明文档的任务是确立和规范软件开发的全过程，有助于提高开发过程中的可见性，便于实施过程控制与项目管理，推动软件工程方法的有效应用，从而提升软件质量。此外，它还能促进开发人员、维护人员、项目管理者之间的交流与协作，并作为整个项目成果的基础依据，同时也向潜在用户传递软件的功能与性能需求，使其能够判断该软件是否符合自身的业务需求。

本文档作为所有参与本项目人员协同工作的基础，详细描述了《Photox 图片管理与社区分享平台》(以下简称 Photox 系统) 的软件需求。通过本需求文档，项目相关方可以了解开发团队对 Photox 系统的功能目标和性能要求的理解是否准确。

1.本文档作为 Photox 系统的系统设计依据，供架构师和开发人员参考使用。

2.作为项目验收评审的重要标准之一，用于确认开发成果是否符合需求。

3.为将来的软件维护与迭代开发提供参考依据。

1.2 范围

项目目标：

1、界面友好，体验良好，符合图片管理与社区分享的业务场景，满足普通用户与管理员等不同角色的使用需求。

2、基于 WEB 架构设计，确保系统具备良好的访问性能与可用性。

3、跨平台、通用性强，系统应具备良好的兼容性、扩展性与安全性，适用于多种终端与浏览器。

4、支持资源共享与协同管理，实现多用户间对图片资源的上传、管理、分析等功能，提升用户参与度和平台活跃性。

5、提供便捷的后台管理与数据维护接口，便于运营人员进行用户管理、内容审查与系统配置等操作。

6、支持图片智能分析功能（如标签识别、分类），提升用户图片组织效率和搜索体验。

2. 总体概述

2.1 软件概述

Photox 是一个集成人工智能技术的专业照片管理与分享平台，旨在为用户提供智能化的照片

存储、自动分类、内容分析和社交分享功能。系统通过深度学习算法和计算机视觉技术，能够自动识别照片中的对象、场景、人物等元素，生成智能标签，实现照片的自动分类和相册生成，极大提升用户的照片管理效率。

2.2 项目介绍

本项目是一个新的独立项目，命名为 Photox，旨在为广大用户提供一个集 图片上传、智能管理、相册整理、社区共享 于一体的综合图像平台。该系统从零开始设计与开发，面向互联网用户，支持高并发访问和多终端使用。

Photox 通过集成人工智能图像识别能力与云存储服务，显著提升了图片管理效率和用户体验。相比传统图床工具，Photox 更强调用户之间的交互和内容价值的发掘，支持社区相册分享与个性化推荐，是一个融合了“图片社交”与“AI 内容标注”的现代化图像平台。

2.3 产品环境介绍

Photox 项目部署和运行于现代 Web 应用环境中，由前端 SPA 页面、后端 API 服务和云端资源组成，支持弹性扩展与云部署，主要环境配置如下：

1. 硬件环境：

服务器平台：云服务器（阿里云 ）

CPU：4 核心以上

内存：8 GB 及以上

存储：支持 SSD 磁盘，最低 50GB，可挂载对象存储

网络：公网 IP，带宽 ≥ 10 Mbps，支持 HTTPS 与防火墙策略

GPU（可选）：用于后续扩展 AI 模型推理优化

2. 软件环境：

操作系统：Linux（

后端框架：Python 3.10 + Django 4.x + Django REST Framework (DRF)

前端框架：Vue 3 + Vite + ElementPlus / TailwindCSS

数据库：MySQL 8.0

缓存与中间件：Redis 6.x（用于缓存、Token 黑名单、消息队列等）

对象存储：七牛云（用于图片存储与 CDN 加速访问）

Web 服务器：Nginx（作为反向代理与静态资源服务器）

应用服务器：Gunicorn（WSGI Server）

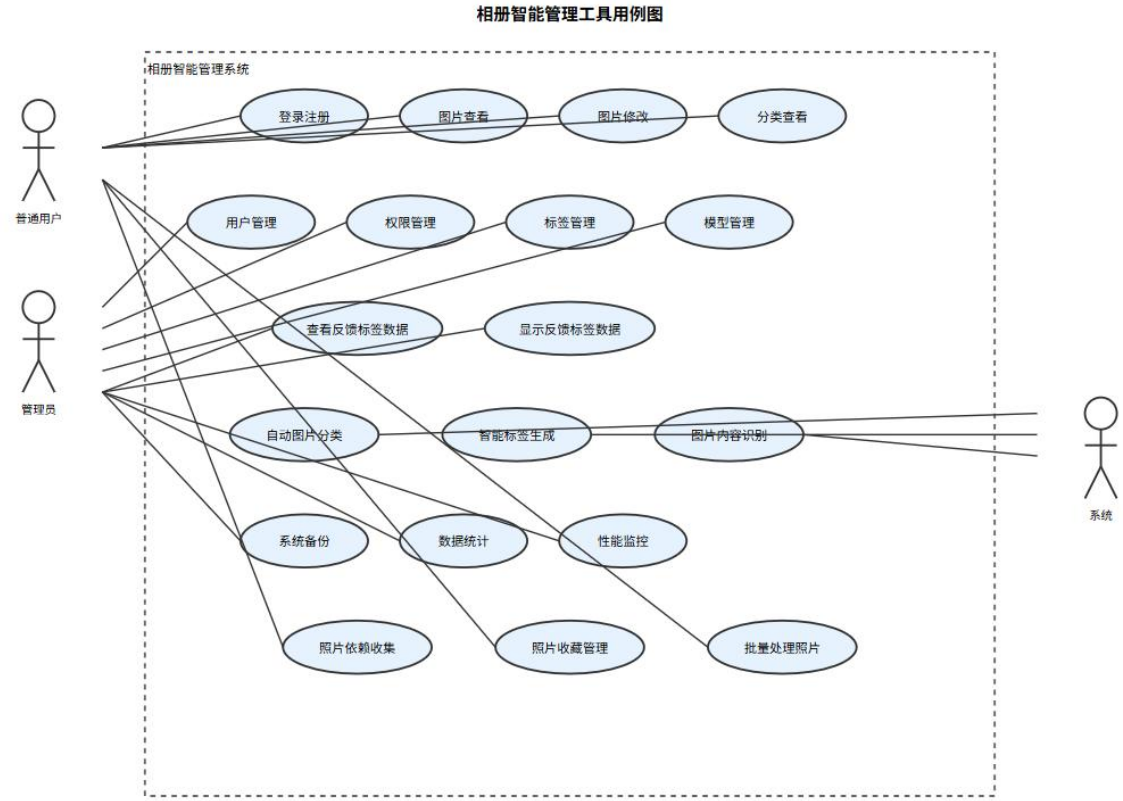
安全机制：HTTPS、JWT 身份认证、CORS、CSRF 保护、API 限流

部署容器：Docker

2.4 软件功能

Photox 是一个基于云存储与AI分析技术的照片管理平台，提供照片的智能分析、内容标签生成、个性化整理与社交分享等功能，致力于为摄影师、摄影爱好者及普通用户打造一个安全、高效、智能的图片管理环境。本系统为 跨平台WEB应用，采用前后端分离架构，兼容PC端与移动端访问，后端部署于云服务器，支持大并发访问，前端界面响应灵活、用户友好。功能需求

3.1 用例图



Use Case 1：用户登录

Goal in Context 简要说明 用户通过输入用户名和密码登录系统，获得访问相册管理功能的权限。此用例确保系统安全性，只有经过身份验证的用户才能访问个人照片数据。

Preconditions 前置条件

- 系统正常运行
- 用户已拥有有效的注册账户
- 数据库连接正常

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 用户成功登录系统
    - 用户会话建立
    - 用户可以访问个人相册功能
    - 系统记录用户登录日志

- **Failed End Condition 失败后置条件**
    - 用户未能登录系统
    - 显示相应错误信息
    - 用户停留在登录页面
    - 记录登录失败日志

Actors 普通用户 - 需要访问相册管理功能的注册用户

Trigger 触发条件 用户访问系统首页并点击"登录"按钮

Description 基本事件流描述 用户需要通过身份验证才能访问个人相册数据，系统通过用户名密码验证用户身份，成功后为用户建立会话。

Step 步骤

1. 用户在浏览器中访问系统 URL
2. 系统显示登录页面
3. 用户输入用户名和密码
4. 用户点击"登录"按钮
5. 系统验证用户凭据
6. 系统检查用户账户状态（是否被禁用）
7. 系统创建用户会话
8. 系统跳转到用户主页面
9. 系统记录登录成功日志

Use Case 2: 图片上传

Goal in Context 简要说明 用户将本地存储的照片文件上传到系统中，系统自动处理图片格式、生成缩略图，并为图片分配唯一标识符进行存储管理。

Preconditions 前置条件

- 用户已成功登录系统
- 用户具有上传权限
- 系统存储空间充足
- 用户本地有待上传的图片文件

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 图片成功上传到服务器
    - 图片信息存储到数据库
    - 生成图片缩略图
    - 用户可以在相册中查看新上传的图片
    - 系统为图片分配唯一 ID
- **Failed End Condition 失败后置条件**
    - 图片上传失败
    - 显示具体错误信息（文件过大、格式不支持等）
    - 系统状态保持不变
    - 记录上传失败日志

Actors 普通用户 - 需要上传照片的已登录用户

Trigger 触发条件 用户点击"上传照片"按钮或拖拽文件到上传区域

Description 基本事件流描述 用户选择本地图片文件进行上传，系统接收文件并进行格式验证、大小检查，成功后存储文件并更新数据库记录。

Step 步骤

1. 用户进入相册管理页面
2. 用户点击"上传照片"按钮
3. 系统打开文件选择对话框
4. 用户选择一个或多个图片文件
5. 系统验证文件格式（JPG、PNG、GIF 等）
6. 系统检查文件大小是否超过限制
7. 系统开始上传文件到服务器
8. 系统显示上传进度条
9. 系统生成图片缩略图
10. 系统将图片信息保存到数据库
11. 系统显示上传成功提示
12. 系统刷新相册列表显示新图片

Use Case 3: 智能图片分类

Goal in Context 简要说明 系统使用人工智能技术自动分析图片内容，识别图片中的对象、场景、人物等元素，将图片自动归类到相应的分类中，提高用户照片管理效率。

Preconditions 前置条件

- 系统中存在待分类的图片
- AI 分类模型已加载并正常运行
- 图片文件完整且可读取
- 分类体系已建立

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 图片被成功分类到相应类别
    - 生成分类标签并存储
    - 用户可以通过分类浏览图片
    - 系统记录分类结果和置信度
- **Failed End Condition 失败后置条件**
    - 分类过程失败
    - 图片保持未分类状态
    - 记录分类失败原因
    - 通知用户分类失败

Actors

- 系统 - 执行自动分类的 AI 模块
- 普通用户 - 查看分类结果的用户

Trigger 触发条件

- 新图片上传完成后自动触发
- 用户手动请求重新分类
- 系统定时批量分类任务

Description 基本事件流描述 系统自动分析图片内容特征，使用预训练的深度学习模型识别图片中的内容，根据识别结果将图片分配到最合适的分类中。

Step 步骤

1. 系统检测到新上传的图片或分类请求
2. 系统加载图片文件进行预处理
3. 系统调用图像识别 AI 模型
4. AI 模型分析图片内容特征

5. AI 模型识别图片中的对象、场景、人物
6. 系统根据识别结果计算分类置信度
7. 系统选择置信度最高的分类
8. 系统将图片归类到相应分类中
9. 系统生成并保存分类标签
10. 系统更新分类统计信息
11. 系统通知用户分类完成

Use Case 4: 图片搜索

Goal in Context 简要说明 用户通过输入关键词、标签或描述信息，快速定位和查找相册中的特定图片。系统支持多种搜索方式，包括文本搜索、标签搜索和智能语义搜索。

Preconditions 前置条件

- 用户已登录系统
- 相册中存在图片数据
- 图片已建立索引
- 搜索引擎服务正常运行

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 系统返回匹配的图片结果列表
    - 图片按相关度排序显示
    - 用户可以预览和选择图片
    - 记录用户搜索历史
- **Failed End Condition 失败后置条件**
    - 未找到匹配的图片
    - 显示"无搜索结果"提示
    - 提供搜索建议或相关推荐
    - 记录搜索失败信息

Actors 普通用户 - 需要查找特定图片的已登录用户

Trigger 触发条件 用户在搜索框中输入关键词并点击搜索按钮或按回车键

Description 基本事件流描述 用户输入搜索条件，系统解析搜索词并在图片数据库中查找匹配的结果，根据相关度排序后返回给用户。

Step 步骤

1. 用户在相册页面定位到搜索框
2. 用户输入搜索关键词或描述
3. 用户点击搜索按钮或按回车键
4. 系统解析搜索词并进行分词处理
5. 系统在图片标签、文件名、描述中搜索匹配项
6. 系统计算搜索结果相关度得分
7. 系统按相关度对结果进行排序
8. 系统返回搜索结果列表
9. 系统显示结果数量和用时
10. 用户浏览搜索结果
11. 系统记录搜索历史

Use Case 5: 用户权限管理

Goal in Context 简要说明 管理员对系统中的用户账户进行权限分配和管理，包括创建、修改、删除用户账户，设置用户角色和访问权限，确保系统安全性和数据保护。

Preconditions 前置条件

- 管理员已登录系统
- 管理员具有用户管理权限
- 用户管理模块正常运行
- 权限体系已定义完善

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 用户权限成功更新
    - 权限变更立即生效
    - 系统记录权限变更日志
    - 相关用户收到权限变更通知
- **Failed End Condition 失败后置条件**
    - 权限变更操作失败
    - 用户权限保持原状
    - 显示具体错误信息
    - 记录操作失败日志

Actors 管理员 - 具有用户管理权限的系统管理人员

Trigger 触发条件 管理员在用户管理页面对特定用户执行权限变更操作

Description 基本事件流描述 管理员查看用户列表，选择需要修改权限的用户，设置新的权限级别和访问范围，系统验证权限变更的合法性并应用新权限。

Step 步骤

1. 管理员进入用户管理页面
2. 系统显示所有用户列表和当前权限状态
3. 管理员选择要修改权限的目标用户
4. 系统显示该用户的详细权限信息
5. 管理员修改用户角色或具体权限设置
6. 管理员确认权限变更操作
7. 系统验证权限变更的合法性
8. 系统更新用户权限数据
9. 系统使新权限设置立即生效
10. 系统记录权限变更日志
11. 系统发送权限变更通知给相关用户
12. 系统显示操作成功提示

Use Case 6: 图片批量处理

Goal in Context 简要说明 用户可以选择多张图片进行批量操作，包括批量删除、批量移动分类、批量添加标签、批量下载等，提高用户处理大量图片的效率。

Preconditions 前置条件

- 用户已登录系统
- 相册中存在多张图片
- 用户具有相应操作权限
- 系统资源充足支持批量操作

End Condition 后置条件

- **Success End Condition 成功后置条件**
    - 所有选中图片完成指定操作
    - 图片状态或属性成功更新
    - 系统显示批量操作结果摘要
    - 用户界面刷新显示最新状态
- **Failed End Condition 失败后置条件**
    - 批量操作部分或全部失败
    - 显示详细的错误报告
    - 成功处理的图片保持变更状态

- 失败的图片保持原状态

Actors 普通用户 - 需要进行批量图片操作的已登录用户

Trigger 触发条件 用户选择多张图片后点击批量操作按钮（如"批量删除"、"批量移动"等）

Description 基本事件流描述 用户通过复选框选择多张图片，选择要执行的批量操作类型，系统逐一处理每张图片并反馈处理结果。

Step 步骤

1. 用户进入相册管理页面
2. 用户通过复选框选择多张目标图片
3. 系统显示已选中图片数量
4. 用户点击批量操作菜单
5. 用户选择具体的批量操作类型
6. 系统显示操作确认对话框
7. 用户确认批量操作
8. 系统开始逐一处理选中的图片
9. 系统显示批量处理进度
10. 系统处理完成后生成结果报告
11. 系统显示成功和失败的图片统计
12. 系统刷新页面显示最新状态

Use Case 7: 系统数据备份

Goal in Context 简要说明 管理员执行系统数据备份操作，包括用户数据、图片文件、数据库信息等的完整备份，确保数据安全和系统可恢复性。

Preconditions 前置条件

- 管理员已登录系统
- 管理员具有系统管理权限
- 备份存储空间充足
- 数据库和文件系统访问正常
- 备份服务模块运行正常

End Condition 后置条件

- **Success End Condition 成功后置条件**
  - 系统数据完整备份完成
  - 生成备份文件清单和校验码
  - 备份文件存储在指定位置

- 记录备份操作日志和时间戳

- **Failed End Condition 失败后置条件**

  - 备份操作失败或不完整
  - 生成详细的错误报告
  - 清理不完整的备份文件
  - 通知管理员备份失败

Actors 管理员 - 负责系统维护的管理人员

Trigger 触发条件

- 管理员手动启动备份操作
- 系统定时自动备份任务触发
- 系统重大更新前的预备份

Description 基本事件流描述 管理员启动备份流程，系统按序备份数据库、用户文件、配置信息等关键数据，生成完整的系统备份包。

Step 步骤

1. 管理员进入系统管理页面
2. 管理员选择"数据备份"功能
3. 系统显示备份配置选项
4. 管理员设置备份范围和存储位置
5. 管理员启动备份操作
6. 系统检查备份存储空间
7. 系统开始备份数据库数据
8. 系统备份用户上传的图片文件
9. 系统备份系统配置文件
10. 系统生成备份文件校验码
11. 系统创建备份清单文件
12. 系统记录备份操作日志
13. 系统通知管理员备份完成状态

3.2 系统模块

本章从用户视角描述 Photox 系统的核心功能需求，通过用例方式详细说明系统应支持的用户行为和系统响应。

3.1 用户管理模块

3.1.1 用户注册

前置条件：用户未登录，访问注册页面。
主要事件流：

1. 用户输入用户名、邮箱、密码、确认密码。
2. 系统检查格式合法性、用户名/邮箱唯一性。
3. 发送激活邮件至邮箱（选配）。
4. 注册成功并提示用户登录。

后置条件：用户信息记录在数据库中，状态为“未激活”或“正常”。

3.1.2 用户登录

输入：用户名/邮箱 + 密码
处理逻辑：

验证身份；
使用 JWT 返回令牌；
登录状态写入 Cookie/本地存储。

异常处理：

用户名或密码错误提示；
用户被封禁或未激活提示。

3.1.3 用户信息修改

用户可在个人中心修改昵称、邮箱、密码。

3.2 图片管理模块

3.2.1 上传图片

功能说明：

支持单张上传；
后端生成缩略图；
自动记录上传时间、来源设备信息；
存储至云端对象存储系统；
将图片路径和元数据写入数据库。

3.2.2 删除图片

用户对自己上传的图片有删除权限；
删除操作同步清理数据库记录和云端资源（延迟删除机制）。

3.2.3 编辑图片标签与描述

用户可为每张图片添加自定义标签；
可修改由 AI 自动生成的标签。

3.3 相册管理模块

3.3.1 创建相册

用户可填写相册名称、描述、封面；
可设定相册权限（私有/公开/密码访问）；
支持添加图片至相册。

3.3.2 编辑相册信息

支持修改相册标题、描述、封面；
支持图片排序、重新分组等操作。

3.3.3 删除相册

删除相册后，图片本身不被删除，只移出该相册。

3.4 AI 智能分析模块

3.4.1 图像内容识别与打标签

上传完成后自动调用 AI 服务；
返回标签（如：猫、阳光、人物、城市夜景等）、场景类别、拍摄风格等；
标签保存在数据库中供后续检索使用。

3.4.2 图像质量评分（后续版本）

分析照片构图、光照、清晰度、噪点等指标；
给出综合评分与改进建议。

3.5 社区模块

3.5.1 浏览社区相册

用户可浏览社区中公开相册；

支持分页、筛选、搜索。

3.5.2 点赞与评论

登录用户可点赞相册；
可对相册进行文字评论；
评论支持删除、举报等操作。

3.6 搜索与推荐模块

3.6.1 标签搜索

支持关键词搜索标签、相册标题；
可联合筛选标签+时间+上传用户等条件。

3.6.2 相似图推荐（后续版本）

基于 AI 提取图像特征；
返回相似图像列表供用户浏览。

3.7 安全与权限控制模块

所有图片/相册默认私有，除非用户设为公开；
用户可设定相册访问权限（公开、私密、密码访问）；
所有用户数据（密码等）均加密存储；
后端接口权限校验严格（基于 JWT 与 RBAC 模型）。

【项目需求描述】

前台功能

功能名称	子功能	描述
“搜索”		
用户中心	登陆信息	查看自己的登录信息
	图片查看	查看分类好的图片信息
	图片修改	对所选择的图片信息进行修改
	分类查看	对照片分类进行查看

后台功能

功能名称	子功能	描述
用户管理	用户管理	对用户信进行管理
	权限管理	对用户权限进行设置
	权限组管理	对不同类型用户的增删改查操作
标签管理	标签添加	添加分类标签
	标签修改	修改分类标签
	标签查看	查看分类标签
	标签删除	删除分类标签
模型管理	模型添加	添加模型标签
	模型修改	修改模型标签
	模型查看	查看模型标签
	模型删除	删除模型标签
数据管理	查看反馈标签数据	对用户的标签数据及进行统计
	显示反馈标签数据	利用图表形式显示标签数据

【项目基础功能】

1.登录注册功能； 软件工程 2023 级专业实习实训项目介绍

2.用户能够搜索和浏览照片；

3.用户能够管理自己的照片；

4.系统能够自动整理照片，进行分类，生成相册等；

5.后台能够管理用户、标签、模型等数据；

【项目拓展功能】

1.系统能够依据相册内容，用一段话总结相册内容；

2.系统能够根据用户的描述，检索相关的照片和相册；
3.系统能够根据照片信息，按时间生成用户的足迹历程

3. 性能需求

3.1 静态量化需求

表 1 静态量化需求表

性能指标	需求规格	说明
最大终端支持	5,000 终端	兼容 Windows/macOS PC 端
同时在线用户数	500 用户	支持中小型社区高峰期在线访问
图片存储容量	1,000 万张	基于 1 万用户 × 1,000 张图片/人
单用户图片存储量	≤1,000 张	单个用户账户可存储上限
数据库记录规模	5,000 万条	包含图片元数据、用户信息、标签数据等

3.2 动态量化需求

A. 正常工作量（每小时）

表 2 正常工作量性能表

操作类型	处理能力需求	数据量需求
图片上传	500 次/小时	25GB/小时（平均 50MB/张）
AI 分析处理	500 次/小时	持续分析能力
图片搜索	5,000 次/小时	每次返回≤50 条结果
社区互动	3,000 次/小时	含评论、点赞等操作

B. 峰值工作量（每小时）

表 3 峰值工作量性能表

操作类型	峰值处理能力需求	数据量需求
图片上传	1,500 次/小时	75GB/小时
AI 分析处理	1,500 次/小时	支持突发任务处理
图片搜索	15,000 次/小时	99%请求响应≤3 秒
社区互动	9,000 次/小时	含热门相册高并发访问

3.3 响应时间需求

为保障用户体验与系统流畅性，各类操作需满足下列响应时限。

操作类型	响应时间要求	特别说明
图片上传	≤5 秒	指上传接口接收并返回成功状态时间
AI 分析处理	≤5 秒	从任务触发到返回 AI 结果时间
缩略图加载	≤1 秒	CDN 缓存命中下快速显示
原图加载	≤3 秒	原图 50MB 以下，带宽 100Mbps
关键词搜索	≤2 秒	返回前 50 条匹配结果
高级多条件搜索	≤5 秒	多标签+时间区间联合查询

4. 接口需求

- Photox 后端软件环境：Linux Ubuntu 20.04 操作系统，Python 3.11 环境，Django 4.x + DRF 框架，数据库使用 MySQL 8.0，采用 Gunicorn + Nginx 部署，容器环境使用 Docker。
- Photox 前端软件环境：基于 Node.js 18 及 Vite 构建，使用 Vue 3 框架，运行于各类主流浏览器环境。
- 云存储服务接口环境：对接七牛云对象存储平台，基于 HTTPS 及 RESTful API 接口方式上传与访问图片资源，使用 Access Key + Secret Key 认证机制。
- 用户登录接口环境：认证方式采用 JWT 标准，前后端通过 HTTP 请求头传递认证 Token，实现用户登录与权限控制功能。

5. 用户接口需求

详细描述用户操作和系统反馈
可以通过草图形式展示

Photox 提供用户友好的图像管理与社交相册平台，用户接口主要通过网页前端（Vue 3）与后端 RESTful API 交互，实现登录、图片上传、相册管理、社区浏览等功能。以下详细描述用户操作流程和界面反馈，包含典型页面结构和交互草图说明。

1. 用户模块 (Users)

●功能需求：

○用户注册（用户名、密码、邮箱）。

○用户登录（用户名、密码），返回 JWT Token 。

○JWT Token 刷新。

○获取用户基本信息（扩展 CustomUser 模型以包含头像、简介等）。

○修改用户基本信息。

●非功能需求：

○密码需安全存储（例如哈希处理）。

○接口需进行身份认证（除注册、登录外） 。

2. 图片模块 (Images)

●功能需求：

○图片上传（支持 JPEG, PNG 格式，最大 50MB）。

○图片存储至云存储 。

○图片上传后触发 AI 智能分析（对象、场景、特征识别） 。

○存储 AI 分析结果（标签）并与图片关联 。

○获取单张图片信息（URL、标题、标签、上传者、创建时间）。

○修改图片信息（如标题）。

○删除图片（需要处理云存储上的文件）。

○获取用户上传的图片列表（支持分页、按时间排序等）。

○（可选）图片搜索（基于标签或其他元数据），响应时间 ≤ 2 秒 。

●非功能需求：

○上传接口需进行身份认证。
○AI 分析接口为内部调用或异步处理，需考虑其响应时间（如 ＜5 秒）。
○个人图片默认私有 。
3. 相册模块 (Albums)
●功能需求：
○创建相册（标题、描述、可见性） 。
○将图片添加到相册。
○从相册移除图片。
○修改相册信息（标题、描述、可见性）。
○删除相册（需确认是否删除相册内图片）。
○获取用户创建的相册列表。
○获取单个相册信息及其包含的图片列表。
●非功能需求：
○相关接口需进行身份认证。
4. 社区模块 (Community)
●功能需求：
○将相册分享到社区（修改相册可见性为公开） 。
○获取社区公开的相册列表（支持排序、分页）。

6. 总体设计约束

Photox 图片管理与社区分享平台在开发过程中存在若干设计层面的限制，需在系统架构、技术选型、安全机制与部署方式等方面遵循既定的约束。后端必须基于 Django 及 Django REST Framework（DRF）实现，前端采用 Vue 3 和 Vite 构建，统一技术栈便于团队协作与维护。图片资源统一存储于七牛云平台，需通过其提供的 API 完成上传、访问与权限控制，数据库使用 MySQL ，并要求在数据建模时考虑索引优化和规范命名以确保高性能。

系统所有 API 接口需采用 JWT 认证机制保障安全性，并实现图片访问权限的分级控制，防止未授权访问。平台需支持 Docker 容器化部署，便于在不同环境中迁移和部署，同时敏感配置信息需通过 .env 文件进行统一管理，确保开发与生产环境分离。系统中集成的图像智能分析功能需通过调用图像识别模型，以提升响应效率并降低资源占用。

此外，Photox 所处理的图片及用户信息需遵循中国大陆的《个人信息保护法》《网络安全法》等相关法律法规，系统必须提供用户数据的隐私设置和删除接口，确保用户对自身数据拥有充分的控制权。上述所有设计约束都是系统开发的基础前提，将直接影响后续的设计决策与开发实现过程。

7. 其他需求

7.1 数据库需求

Photox 采用关系型数据库（MySQL），数据库设计需遵循规范化原则，确保数据一致性和完整性。表结构设计应合理建立索引以优化查询性能。对图片元数据、用户信息、相册分别建表，支持必要的外键约束和事务管理。数据备份机制需完善，定期导出并保存备份文件，确保数据安全。

7.2 编码规范

项目开发应遵循团队统一的编码规范和风格指南。后端代码使用 Python 语言，遵循 PEP8 标准，变量命名应具有描述性且统一；前端代码采用 Vue 3 框架，遵守官方风格指南，使用 ESLint 进行代码检查。所有提交代码需经过代码审查，保证代码质量与可读性。

7.3 错误处理

系统应实现完善的异常捕获机制，保证在出现异常时能返回明确的错误信息给前端，避免程序崩溃。API 返回格式需统一，包含错误码、错误描述及必要的调试信息。前端应根据错误码做出对应提示，如网络异常、权限不足、文件大小超限等。日志系统需记录关键异常，便于后期排查。

7.4 测试需求

项目需覆盖单元测试、集成测试和接口测试，确保各模块功能的正确性和稳定性。后端使用 postman 等测试，前端使用 Jest 进行组件测试。测试用例应覆盖正常流程和边界情况，错误输入和异常场景。发布前需进行压力测试，保证系统在高并发环境下的性能表现符合预期。

7.5 性能需求

数据库访问需优化查询效率，保证响应时间在可接受范围内。对大批量图片访问需支持分页加载及缓存机制。

7.6 安全需求

必须保护用户数据安全，采取 HTTPS 传输、密码加密存储、接口鉴权等措施。系统应防范常见安全威胁，如 SQL 注入、跨站脚本攻击（XSS）、跨站请求伪造（CSRF）等。用户权限管理应细化，防止越权操作。

8. 需求分级

需求ID	需求名称	需求分级
1	用户注册与登录	A
2	图片上传	A
3	用户权限管理	A
4	相册管理	B
5	图片智能分析	B

6	图片下载	B
7	社区动态浏览	C
8	搜索与筛选功能	C

9. 待确定问题

需求ID	问题描述	影响 (H/M/L)	风险	责任人	解决日期	状态 (Open/Close)