**照片分类项目**

**概要设计说明文档**

**目录**

**1引言**

1.1编写目的

1.2项目背景

1.3参考资料

**2系统总体设计**

2.1整体架构

2.2模块化设计

**3数据流设计**

**4性能设计**

4.1响应时间

**5外部接口**

5.1 硬件接口

5.2 软件接口

**6程序接口设计**

6.1.导入图片

6.2.按指定顺序排序

6.3.删除指定照片

6.4.分类预览

6.5.搜索

**1引言**

**1.1编写目的**

本文档旨在全面、系统地阐述本项目的系统模块划分、选择的技术路线等，整体说明软件的实现思路。，以确保项目团队成员、利益相关者及潜在用户能够清晰理解项目的核心价值、技术实现路径及预期成果。

**1.2项目背景**

随着数字摄影技术的飞速发展，个人和企业产生的照片数量急剧增加，如何高效地管理、存储和检索这些照片成为了一个亟待解决的问题。照片管理系统应运而生，它不仅解决了传统文件夹管理方式在照片分类、搜索和分享方面的不足，还通过引入自动日期分类和AI分类等先进功能，为用户提供了更加智能和便捷的照片管理体验。随着技术的不断进步和市场的不断扩大，照片管理系统将具有广阔的市场前景和巨大的商业价值。

**1.3参考资料**

（1）华为：OpenGauss数据库官方指南

（2）MDN：Vue开发指南

（3）照片分类项目需求文档

**2系统总体设计**

**2.1整体架构**

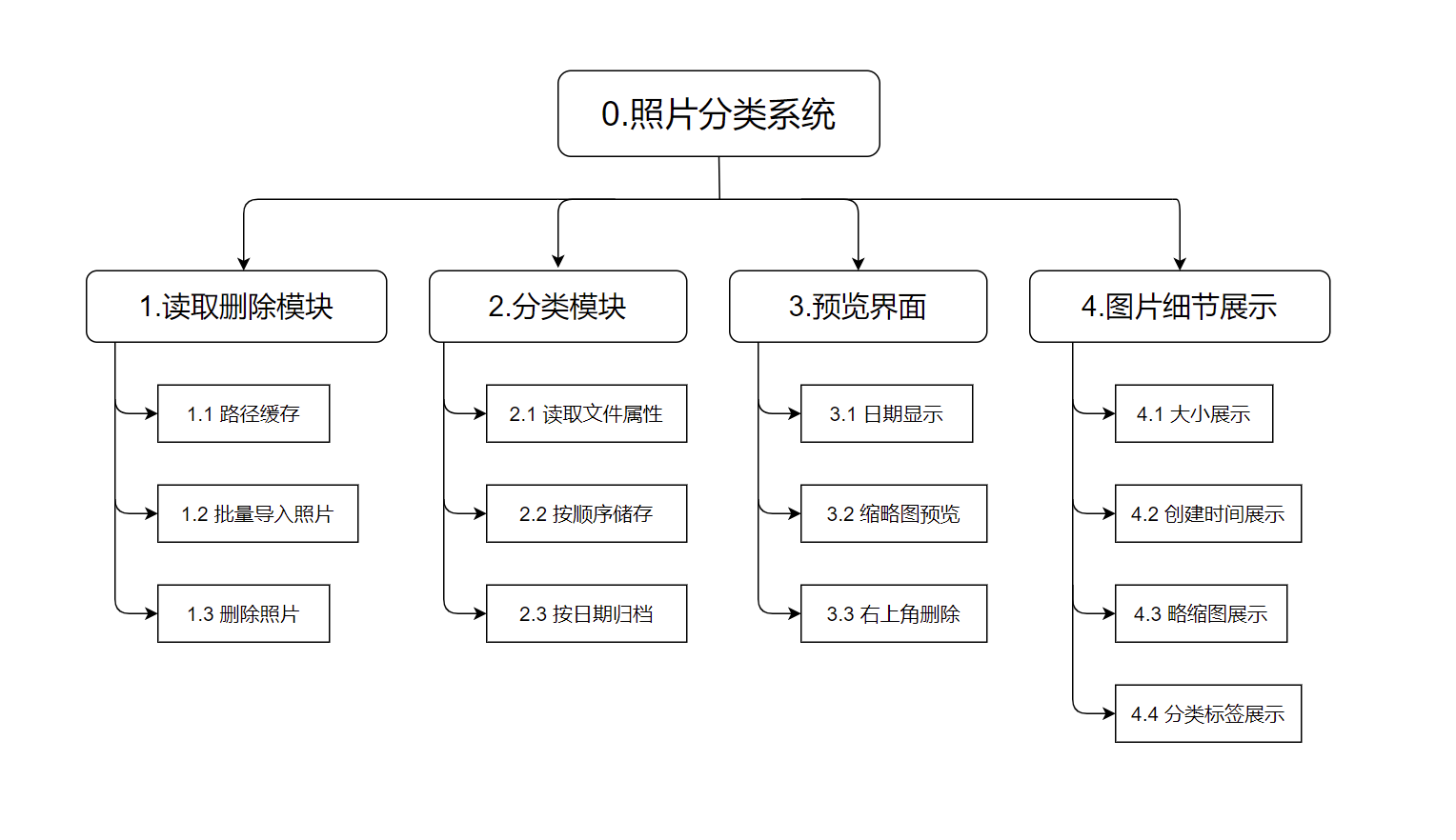
系统分层架构

 表现层：用户交互界面，包括导入界面、预览界面、图片细节展示界面。前端技术上我们小组采用开发vue.js技术用于构建用户界面。

 业务逻辑层：负责处理相片分类、文件操作、缓存管理等核心功能。计划的使用语言是Python，采用微服务架构，将不同模块（如读取删除模块、分类模块）分离成独立服务，方便后续扩展和维护。

 数据层：负责持久化存储和读取文件的元数据及分类信息。计划本地采用MySQL来存储结构化元数据，并采用Gauss数据库进行云存储。

**2.2模块化设计**



 缓存管理模块：管理相片路径的缓存，支持批量导入、删除功能。提供接口与业务逻辑层交互。

 分类模块：实现文件属性读取和日期分类功能。后续设计接入AI分类，支持更多分类方式（如按地点、标签分类）。

**3数据流设计**

 导入流程：用户通过导入接口上传图片，系统存储图片并记录元数据（文件大小、创建时间等），更新缓存管理模块。

 分类流程：当用户请求分类时，分类模块根据元数据执行分类操作，将分类后的结果存储在数据库中，并更新预览界面。

 预览和细节展示流程：预览界面从缓存或数据库中获取图片元数据，按需加载图片并展示。细节展示模块提供单张图片的详细信息，支持删除和重新分类操作。

**4性能设计**

**4.1响应时间**

a.交互和预览无明显卡顿。

b.导入速率 100张导入1s内完成。

c.分类速度 1G或100张以内无明显卡顿。

**5外部接口**

**5.1 硬件接口**

支持win10及以上pc使用。

* 存储设备接口：支持通过USB、SATA等接口导入存储设备中的图片。

**5.2 软件接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 接口描述 | 请求参数 | 错误信息 | 文件操作结果 |
| 图片导入接口 | 1. 用于读取和写入图片文件 2. 支持文件路径的操作，如读取文件、删除文件等 | 字段名称: 文件路径请求 字段来源: 用户操作 示例: C:/Users/Example/Images 逻辑验证规则: 文件路径合法，用户有访问权限 | 字段名称: 文件操作错误 字段描述: 文件系统返回的错误信息 示例: 文件路径错误、权限不足等 | 字段名称: 文件操作结果 字段描述: 文件读取或写入的结果 示例: 文件读取成功、写入成功等 |
| 图像处理库接口 | 1. 用于处理图片文件，如生成缩略图、图像分类等 | 字段名称: 图像处理请求 字段来源: 系统调用 示例: 调用OpenCV库处理图像 逻辑验证规则: 图像文件存在 | 字段名称: 图像处理错误 字段描述: 图像处理库返回的错误信息 示例: 图像格式不支持、处理失败等 | 字段名称: 图像处理结果 字段描述: 图像处理成功与否的结果 示例: 图像处理成功，生成缩略图等 |
| 数据库接口 | 1. 用于保存和读取图片路径、分类标签等信息 2. 支持增删查改操作，维护图片相关数据 | 字段名称: 数据库操作请求 字段来源: 系统调用 示例: 读取或写入图片路径及分类信息 逻辑验证规则: 数据库文件存在 | 字段名称: 数据库操作错误 字段描述: 数据库返回的错误信息 示例: 连接失败、读写错误等 | 字段名称: 数据库操作结果 字段描述: 数据库操作的结果 示例: 数据写入成功、读取成功等 |

**6程序接口设计**

**6.1.导入图片**

**6.1.1基本信息**

请求路径：front\_end\src\views\home

接口描述：该接口用于按当前照片分类方式从手机导入照片到本地，并在网页端显示

**6.1.2请求参数**

请求参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| Option | 展示顺序 | String | 是 | 当前可选项：  1.导入顺序  2.时间顺序  3.其他顺序 |

请求参数示例：

Option=”时间顺序”

**6.1.3响应数据**

响应数据类型：ImageInfo[]（包含图片路径、图片属性等信息的一组类/结构体）

类成员说明（仅作为一个例子）：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| url | 照片路径 | String | 是 |  |
| name | 照片名 | String | 是 |  |
| time | 拍摄时间 | DateTime | 是 |  |
| size | 照片大小 | String | 是 |  |
| place | 拍摄地点 | String | 否 |  |
| device | 设备信息 | String | 否 |  |

响应数据样例：

ImageInfo[5]

其中imageInfo[0]：

{

“url”:…\resources\image1.jpg,

“name”:”DM\_20240328003702\_001”,

“time”:2024/08/24,

“size”:”6.78MB,4000×3000px”,

“place”:”北京理工大学13号楼”,

“device”:”红米K40”

}

**6.2.按指定顺序排序**

**6.2.1基本信息**

请求路径：front\_end\src\views\home

接口描述：该接口在照片导入页右上角选择不同的分类方式时被调用，将导入的照片按照指定方式排序，并记录当前分类方式

**6.2.2请求参数**

请求参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| Filelist | 当前导入的照片路径列表 | String[] | 否 | 后端可访问本地存储获得 |
| Option | 展示顺序 | String | 是 | 当前可选项：  1.导入顺序  2.时间顺序  3.其他顺序 |

请求数据样例：

Filelist=[…\resources\image1.jpg,…\resources\image1.jpg…],Option=”时间顺序”

**6.2.3响应数据**

响应数据类型：ImageInfo[]（包含图片路径、图片属性等信息的一组类/结构体）

类成员说明（仅作为一个例子）：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| url | 照片路径 | String | 是 |  |
| name | 照片名 | String | 是 |  |
| time | 拍摄时间 | DateTime | 是 |  |
| size | 照片大小 | String | 是 |  |
| place | 拍摄地点 | String | 否 |  |
| device | 设备信息 | String | 否 |  |

响应数据样例：

ImageInfo[5]

其中imageInfo[0]：

{

“url”:…\resources\image1.jpg,

“name”:”DM\_20240328003702\_001”,

“time”:2024/08/24,

“size”:”6.78MB,4000×3000px”,

“place”:”北京理工大学13号楼”,

“device”:”红米K40”

}

**6.3.删除指定照片**

**6.3.1基本信息**

请求路径：front\_end\src\views\home

接口描述：该接口在网页端删除指定照片时被调用，后端同步删除本地的存储

**6.3.2请求参数**

请求参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| File.url | 被删除的照片路径 | String | 是 |  |

请求数据样例：

Fileurl=”…\resources\image1.jpg”

**6.3.3响应数据**

响应数据类型：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| Code | 是否删除成功 | Number | 是 | 1成功，0失败 |

**6.4.分类预览**

**6.4.1基本信息**

请求路径：front\_end\src\views\home

接口描述：该接口在网页端点击分类预览时被调用，根据当前分类方式读取本地存储，对已导入的照片进行分类（不止于排序，要显示类别信息）

**6.4.2请求参数**

请求参数说明：无

**6.4.3响应数据**

响应数据类型：

classification[]

{

name:string, //类别名，如2024/08/24

Image[] //这一类中的照片列表

{

“url”:…\resources\image1.jpg,

“name”:”DM\_20240328003702\_001”,

“time”:2024/08/24,

“size”:”6.78MB,4000×3000px”,

“place”:”北京理工大学13号楼”,

“device”:”红米K40”

}

}

**6.5.搜索**

**6.5.1基本信息**

请求路径：front\_end\src\views\home

接口描述：该接口在网页端点击搜索时被调用

**6.5.2请求参数**

请求参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 说明 | 类型 | 是否必须 | 备注 |
| Key | 搜索关键字 | String | 是 | 目前不确定支持在哪些范围内搜索 |

请求参数示例：

Key=”00”

**6.5.3响应数据**

响应数据类型：ImageInfo[]

响应数据样例：

ImageInfo[5]

其中image0：

{

“url”:…\resources\image1.jpg,

“name”:”DM\_20240328003702\_001”,

“time”:2024/08/24,

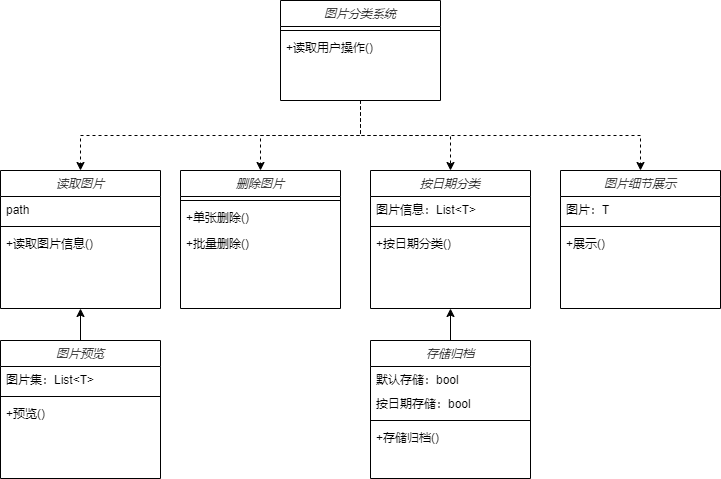
“size”:”6.78MB,4000×3000px”,

“place”:”北京理工大学13号楼”,

“device”:”红米K40”

}

**7 数据结构**



首先是项目后端开发采用ssm架构，也就是mapper层、service层和controller层，具体的开发框架采用的是springboot，并在本地持久化上采用的技术是mybatis及mysql。结合接口文档和实际开发需要，现设计以下关于后端的数据结构：

首先是实体类：

* ImageInfo: 存储图片信息的实体类。
  + 属性:
    - url: 照片路径 (String)
    - name: 照片名 (String)
    - time: 拍摄时间 (Date)
    - size: 照片大小 (String)
    - place: 拍摄地点 (String)
    - device: 设备信息 (String)
* Classification: 存储分类信息的实体类。
  + 属性:
    - name: 类别名 (String)
    - images: 该类别中的照片列表 (List)

然后是关于Mapper层，该层负责与数据库的交互打交道，包括：

* ImageMapper: 与数据库交互的接口，负责图片信息的增删查改。
  + 方法:
    - insertImage(ImageInfo image): 插入图片信息
    - deleteImage(String url): 删除指定路径的图片
    - selectImages(): 查询所有图片信息
    - selectImagesByTime(): 按时间顺序查询图片信息
    - selectImagesByOther(): 按其他方式查询图片信息 (根据实际需求设计)

在Service层上，负责业务逻辑处理。包括：

Service接口

* ImageService: 业务逻辑处理接口，封装对图片信息的操作。
  + 方法:
    - importImages(List<String> urls, String option): 导入图片
    - sortImages(List<String> urls, String option): 排序图片
    - deleteImage(String url): 删除图片
    - getClassifications(String option): 获取分类信息
    - searchImages(String key): 搜索图片

以及Service实现类：

* ImageServiceImpl: 实现 ImageService 接口，完成具体业务逻辑。
  + 需要调用 ImageMapper 接口进行数据库操作。
  + 根据文档描述，需要实现图片导入、排序、删除、分类预览和搜索等功能。

在Controller层上，用于处理前端请求并返回响应，包括有：

* ImageController: 接收前端请求，调用 Service 层方法，返回响应数据。
  + 需要处理不同的请求路径和参数，调用对应的 Service 方法。
  + 需要将响应数据转换为 JSON 格式返回给前端。

这里的不同请求路径便包括了：import导入、delete删除、sort分类、search搜索等不同类型的前端请求。

这里附有关于该后端的总体架构设计导图：

