浙江大学

本科实验报告

课程名称：	B/S 体系软件设计
姓　　名：	郑博文
学　　院：	计算机科学与技术学院
系：	计算机系
专　　业：	软件工程
学　　号：	3190103037
指导教师：	胡晓军

2021 年　1 月　3 日

浙江大学实验报告

课程名称：B/S 体系软件设计 实验类型：综合型

实验项目名称：图像数据标注网站开发

学生姓名：郑博文 专业：软件工程 学号：3190103037

同组学生姓名：无 指导老师：胡晓军

实验地点：玉泉 32 舍 403 实验日期：2021 年 11 月 1 日

1.项目背景

本项目是浙江大学《B/S 体系软件设计》课程的课程设计，旨在设计实现一个用于机器学习的图像数据标注网站，允许用户登录后，上传图片、视频等图像信息并发布图像数据标注任务，其他用户可以在登陆后进行标注完成任务，标注结果可以导出为多种数据集格式。同时要求网站页面对用户友好，并提供必要的软件项目文档。本课程项目设计的目的在于帮助同学独立理解并掌握一套 B/S 体系的 web 应用开发技术和开发的总体流程。

本文档是该项目的中期报告兼系统设计文档，主要包括了系统的需求分析、总体架构设计、数据库设计、系统接口设计以及项目当前进度等内容。

2.系统需求分析

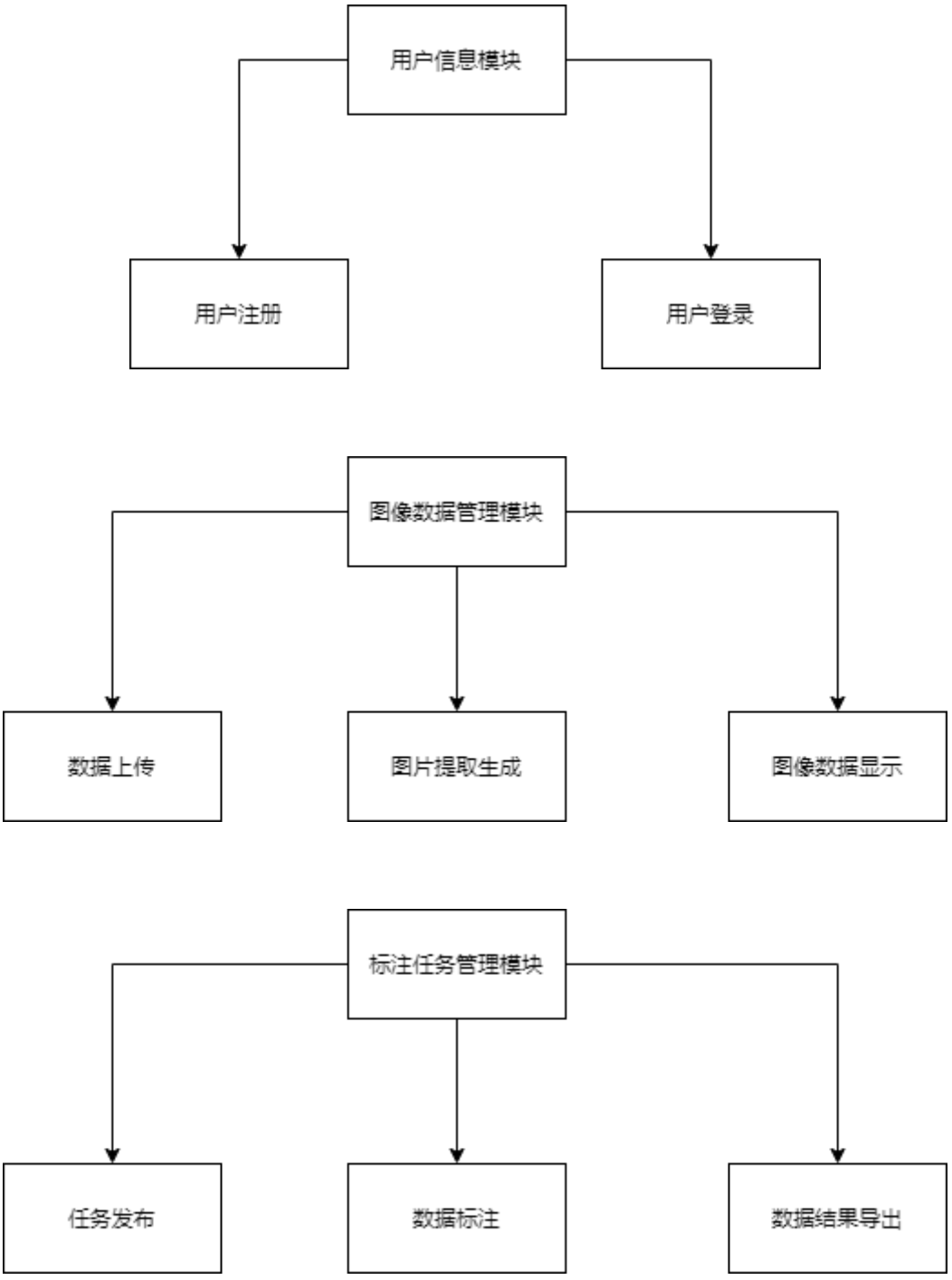
2.1. 功能性需求分析

该项目要求对于每个用户实现如下功能性需求：

- 用户注册（需要用户名、密码、手机号、邮箱等关键信息）
- 用户登录
- 批量上传图片、视频等图像信息
- 将上传的视频转换为多张图片，并可以选择有效图片或按照一定规则自动提取图片。

- 用户可选择若干图片创建标注任务
- 用户可领取他人发布的任务并进行数据标注
- 标注结果支持到处多种数据集格式（如 PASCAL VOC， COCO 等）

由以上功能性需求，可将系统分为三个模块，分别是用户信息模块、图像数据管理模块和标注任务管理模块。这些功能性需求的关系和模块划分可用一下图表表示。



其中每个功能下有若干子功能，例如数据标注功能下包括在图片上绘制选框和输入标签文字，数据上传包括上传图片和上传视频等等，任务发布包括任务内容设置和发布上线等等。

2.2. 非功能性需求分析

该系统的非功能性需求包括性能需求、输入输出需求、数据管理需求等，具体的要求如下。

2.2.1. 性能需求

- 系统应保证正常运行，避免出现崩溃；
- 保证当前主流浏览器均能访问本系统；
- 应至少保证支持 100 人的并发访问；
- 在用户进行操作时，系统应能在 1 秒内做出响应；
- 每个页面应在较快速度内完成加载，正常情况下不超过 1 秒，高峰时期不超过 5 秒；
- 系统应能检测出异常状况，如无法连接数据库服务器等情况；

2.2.2. 输入输出需求

- 用户注册时，应该对用户名、密码进行验证，长度在 6 个字符以上。
- 输入的 email 地址需要进行格式验证。
- 对于注册时输入的用户名和 email，应保证其在系统中是唯一的。

2.2.3. 数据管理需求

- 系统服务器软件必须提供可靠的数据备份和恢复手段，在服务器软件或硬件出现严重故障 时，能够根据备份的数据和账户信息迅速恢复正常运行环境。同时，软件开发者不得在系 统中预留任何特殊账户和密码，保证其安全性。
- 系统应具备加密登录、数据加密传输等安全方面的保障，保证数据在不用系统间 传输过程中的保密性与安全性。

3.系统技术选型与架构设计

3.1. 项目技术选型

本项目采用前后端分离的 B/S 体系的 Web 开发技术。整个项目各环节结束选型如下：

- 前端：React + Ant Design 组件库 + react-router
- 后端：Django + Django Rest Framework
- 数据库：SQLite3

由于条件受限，本项目预计在本地 PC 运行，将不会部署在远程服务器上。

3.2. 项目主要技术介绍

3.2.1. 前端技术框架

项目主要使用了 React + Ant Design 组件库作为前端开发的技术栈。

React 是 FaceBook 公司开发的一套 Web 开发框架，基于 Javascript 来构建用户界面，可构建管理自身状态的封装组件，然后对其组合产生复杂的 UI。React 采用虚拟 DOM 的技术和声明式设计，可以非常高效而灵活地开发 Web 应用。

Ant Design 是蚂蚁集团开发的开源的企业级产品 UI 框架，提供了大量美观而功能性强的 UI 组件。

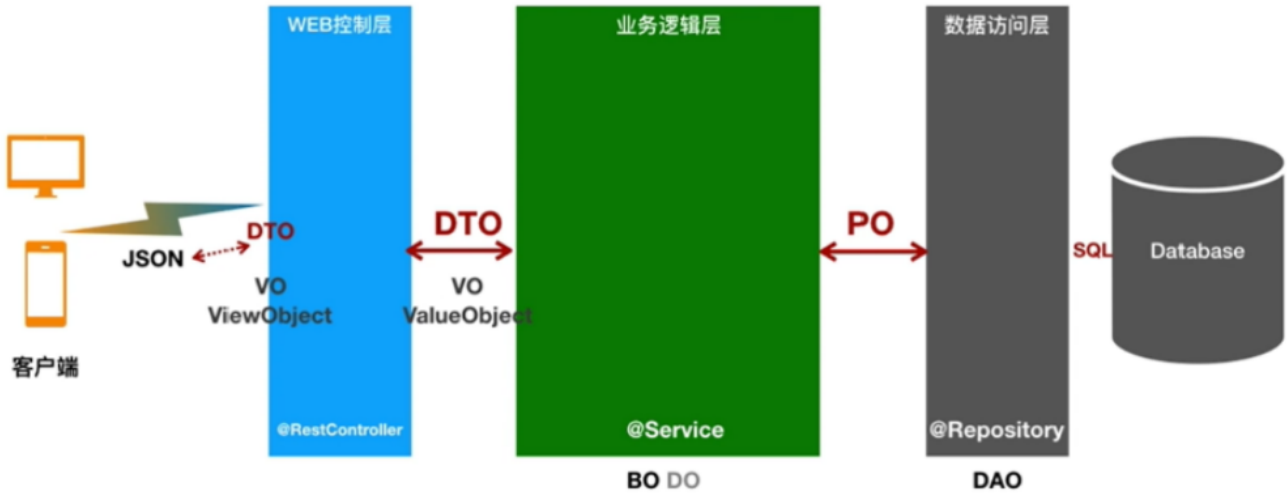
3.2.2. 后端技术框架

本项目后端技术主要采用 Django 框架。

Django 是一个由 Python 编写的开源 Web 应用框架。使用 Django，只需少量代码，程序开发人员就可以完成一个正是网站所需要的大部分内容。Django 本身基于 MVC 模型，即 Model（模型）+ View（视图）+ Controller（控制器）设计模式，MVC 模式是后续对程序的修改和扩展简化，具有低耦合、开发快捷、部署方便、可重用性高等优势。且 Django 自带了强大的后台管理功能，便于网站调试和管理。

3.3. 前端架构图

3.4. 后端架构图



3.5. 系统运行环境

系统运行环境包括硬件环境和软件环境，具体要求如下。

3.5.1. 软件要求

项目	软件名称	最低要求版本
操作系统	Windows7 及以上、Linux	
编程语言环境	Python	3.7.0
数据库	SQLite	
Django 包	Django	2.2
浏览器	Chrome	

3.5.2. 硬件要求

硬件项目	名称
CPU	CORE i5 及以上
内存	4G 及以上
硬盘	剩余空间 200G 及以上

4.数据库设计与 ER 图

4.1. 数据表设计

本项目的数据库共涉及了三个数据表，分别用于存储用户信息、图像数据、任务信息、任务与图像数据关系以及标注结果等信息，具体表结构设计情况如下。

4.1.1. User 表

该表用于保存用户信息。

字段名	类型	描述	备注
id	int	用户的 id	主键，非空，自增
name	varchar(128)	用户名	非空，unique
password	varchar(128)	账号密码	非空
email	varchar(128)	用户邮箱	非空，unique

4.1.2. 图像数据表

该表用于保存图像数据信息。

字段名	类型	描述	备注
id	int	文件的 id	主键，非空，自增
uploader	Int	文件上传者的 id	外键，参照 User 表
filename	varchar(256)	文件名	非空，包含路径，实际文件存储在硬盘当中，通过路径访问

4.1.3. 标注任务表

标注任务表用于保存发布给用户的信息，标注结果和标注对象图像不存在该表，因为它们是一对多或多对多关系。

字段名	类型	描述	备注
id	int	任务的 id	主键，非空，自增
Uploader	Int	任务发布者的 id	外键，参照 User 表
description	text	任务描述	记录标注任务的目标

			等说明信息
name	varchar(64)	任务名称	非空
state	varchar(8)	任务状态	非空，已收到回答、未收到回答、正在进行

4.1.4. 标注结果 VOC 表

标注结果表用于保存用户的标注结果的 VOC 格式。VOC 格式的数据集每一个 xml 文件将对应一个

字段名	类型	描述	备注
Task	int	任务的 id	外键，参照标注任务表
image	Int	该标注数据对应图片的 id	外键，参照图像数据表
Annotation_file	mediumblob	数据集文件	非空

4.1.5. 标注结果 COCO 表

标注结果表用于保存用户的标注结果的 COCO 格式。COCO 格式的数据集每一格标注任务对应一个 json 文件。

字段名	类型	描述	备注
Task	int	任务的 id	外键，参照标注任务表
dataseet_file	mediumblob	数据集文件	非空

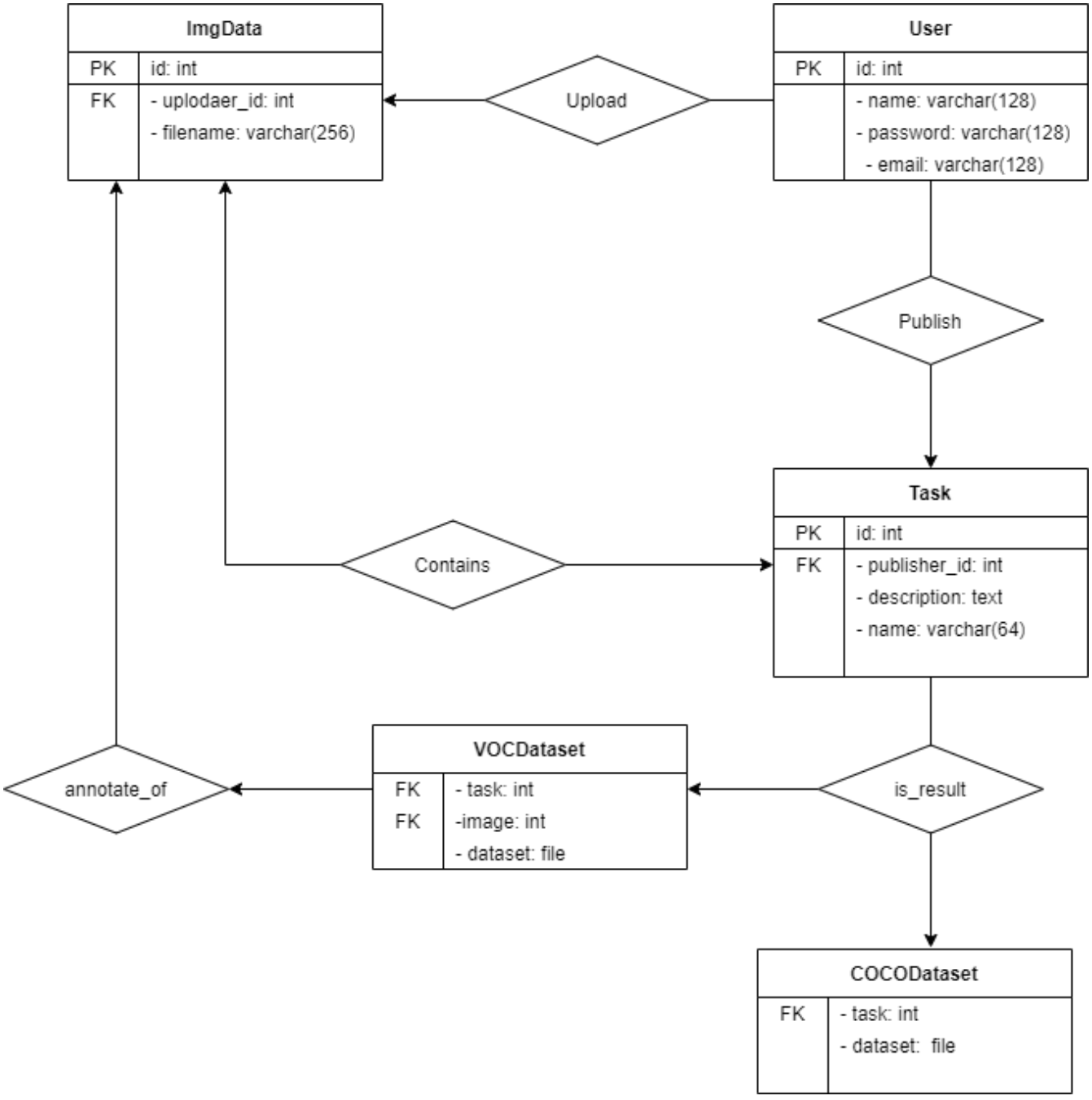
4.1.6. 任务图像表

任务图像表是图像数据表与标注任务表的关系，用于表示两者之间的多对多关系，因为一个任务包含多个图像，一个图像也可以被多个任务包含，在 ER 图中不作为一张表的形式出现。

字段名	类型	描述	备注
task_id	int	任务的 id	外键，参照标注任务

			表
img_id	int	图像数据的 id	外键，参照图像数据表

4.2. ER 图



5.接口设计

注：一下所有带有 /api 的 url，都需要在请求头带上 token 令牌以验证用户身份。

5.1. 账户管理相关接口

5.1.1. 用户登录

URL	/api/users/login
请求方法	POST

请求格式	{
	“username”: “xxx”,
	“password”: “xxx”
	}

响应格式	{
	"code":201,
	"msg":"login success!",
	"data":{
	"token":"xxxxx..."
	}
	}

备注	返回 token 为校验令牌，需要前端存在本地，此后请求需要在请求头带上 token，token 有效期由服务端决定。

5.1.2. 用户注册

URL	/api/users/register
请求方法	PUT

请求格式	{
	“username”: “xxx”,
	“password”: “xxx”,
	“email”: “xxxx@xx.com”
	}

响应格式	{
	"code":200,
	"msg": "register success!",
	"data":{}
	}

备注	注册后不会返回 token，需要自行登录

5.2. 标注任务相关接口

5.2.1. 创建发布任务

URL	/api/tasks/publish
请求方法	PUT

请求格式	{
	"name": "task name",
	"descriprion": "task descrpition",
	"end_time": "xxxx-xx-xx",
	"publisher_name": "xxx",
	"questions":[
	{
	"img_id": "id of the img",
	}
	]
	}

响应格式	{
	"code":200,
	"msg": "publish success!",
	"data":{}
	}

备注	若返回 500 状态码，则表示非法数据

5.2.2. 删除任务

URL	/api/task/#{task_ID}/owner/delete
请求方法	DELETE
请求格式	{}

响应格式	{
	"code": 200,
	"msg": "delete success!",
	"data": {}
	}

备注	

5.2.3. 查看待标注任务列表

URL	/api/tasks/
请求方法	GET
请求格式	{}

响应格式	{
	"tasks": [
	{
	"name": "task name",
	"descriprion": "task descrpition",
	"end_time": "xxxx-xx-xx",
	}
	]
	}

备注	

5.2.4. 查看个人发布任务列表

URL	/api/tasks/list_mine
请求方法	GET

请求格式	{}

响应格式	{
	"tasks": [
	{
	"name": "task name",
	"descriprion": "task descrpition",
	"end_time": "xxxx-xx-xx",
	}
	]
	}

备注	

5.2.5. 查看任务详情

URL	/api/task/#{task_ID}/owner /api/task/#{task_ID}/all
请求方法	GET
请求格式	{}

响应格式	{
	"name": "task name",
	"descriprion": "task descrpition",
	"end_time": "xxxx-xx-xx",
	"publisher_name": "xxx",
	"questions":[
	{
	"img_id": "id of the img",
	"img": "base64 img",
	}
	]
	}

备注	

5.2.6. 提交标注结果

URL	/api/datasets/COCO /api/datasets/VOC
请求方法	POST

请求格式	{
	"answers":[
	{
	"img_id": "id of the img",
	"annotations": {
	"category": "category",
	"x": "coordinate_x",
	"y": "coordinate_y",
	"width": "int",
	"height": "int"
	}
	}
	]
	}

响应格式	{
	"code": 200,
	"msg": "submit success!",
	"data": {}
	}

备注	

5.2.7. 导出标注结果

URL	/api/datasets/VOC/download

	/api/datasets/downloads/COCO
请求方法	GET
请求格式	{}

响应格式	{
	"dataset": "base64 dataset file"
	}

备注	URL 中的 type 参数表示需要导出的数据集格式

5.3. 图片管理相关接口

5.3.1. 上传图片

URL	/api/images/upload
请求方法	POST

请求格式	{
	"images": [
	{
	"filename": "filename",
	"img": "base64 data"
	}
	]
	}

响应格式	{
	"code": 200,
	"msg": "submit success!",
	"data": {}
	}

备注	可一次性上传多张图片

5.3.2. 查看图片

URL	/api/images
请求方法	GET
请求格式	{}

响应格式	{
	"images": [
	{
	"filename": "filename",
	"img": "base64 data"
	}
	]
	}

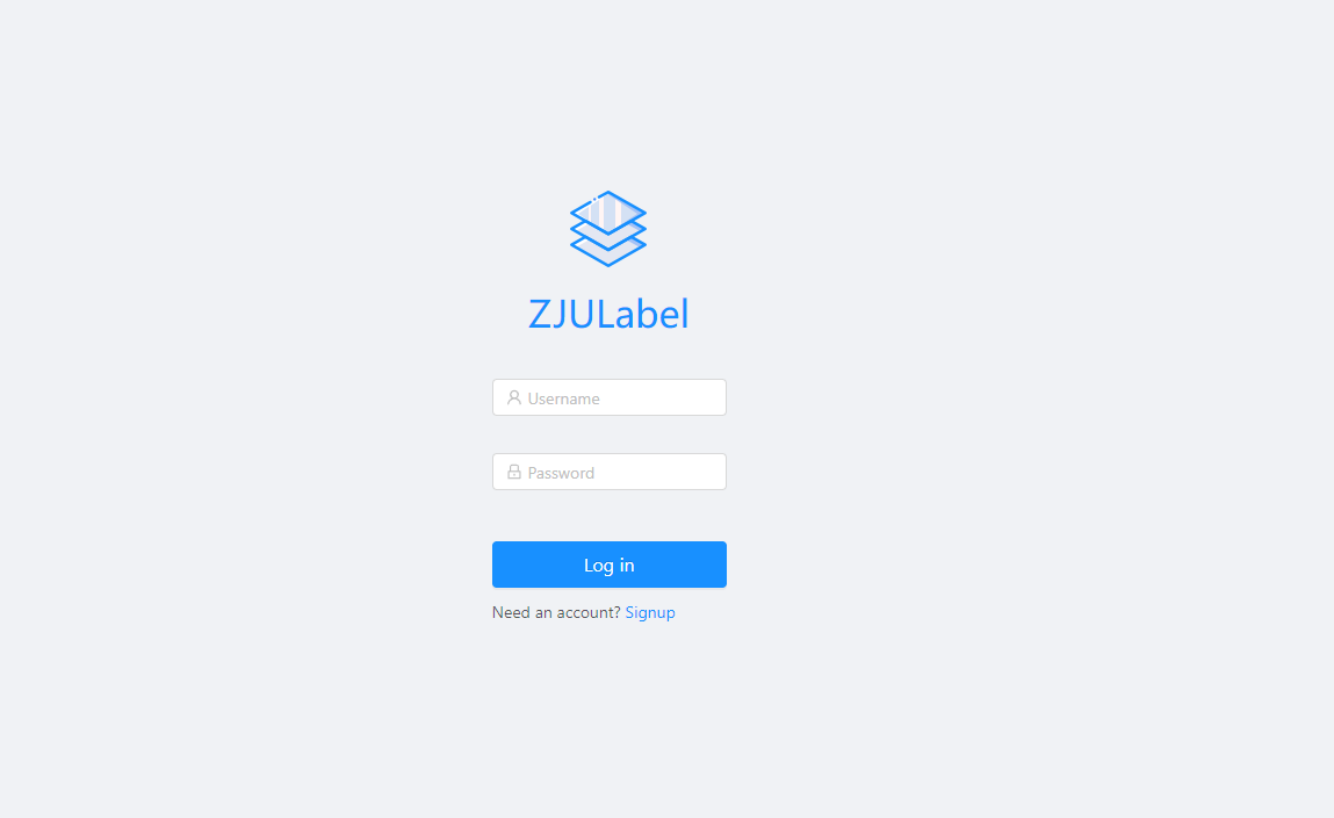
备注	

6. 系统界面原型

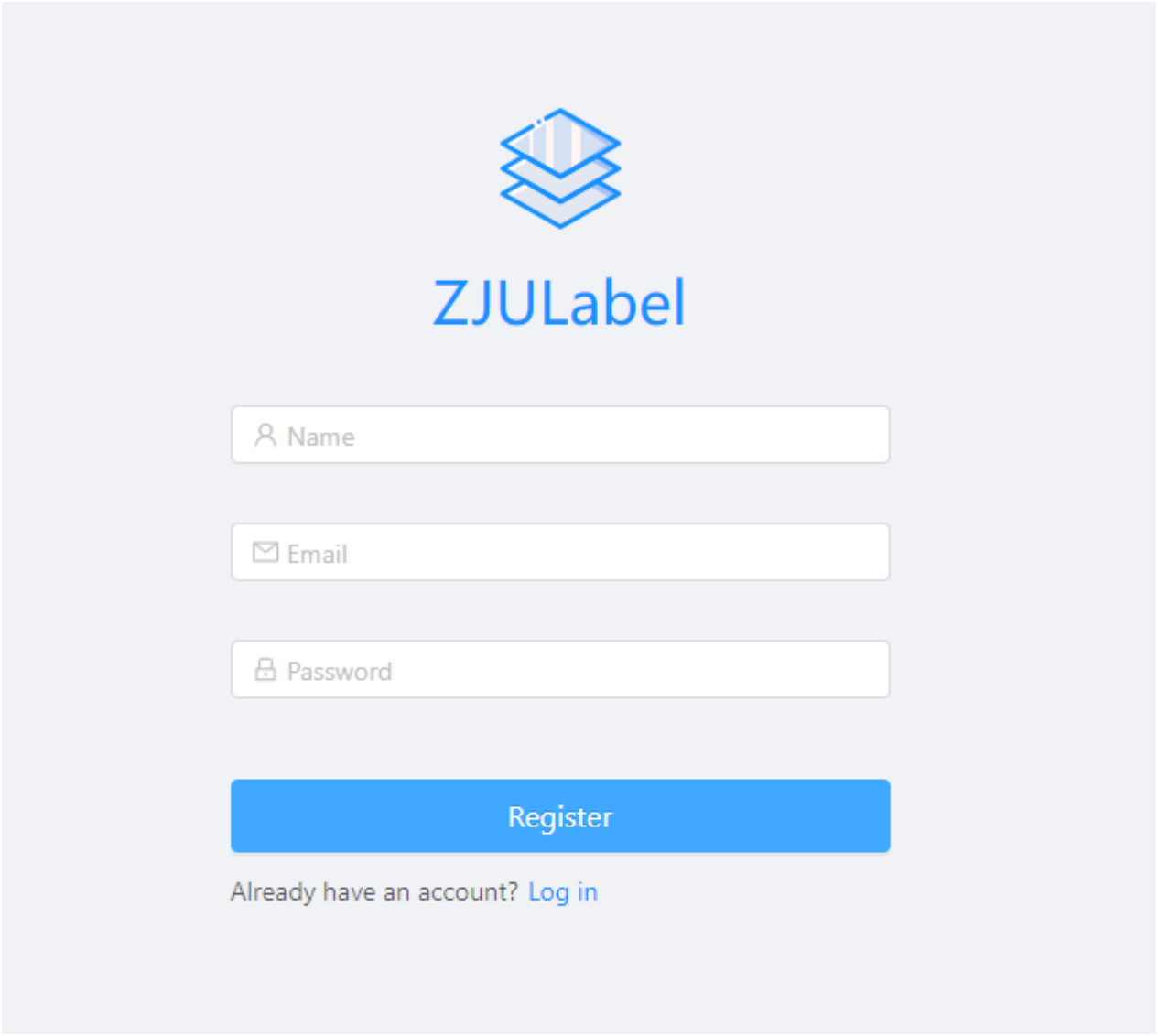
系统的界面原型设计如下所示。（大部分页面还未设计完成，本文档将在大程提交时修改完成）

注：最终成品和界面原型可能存在较大差异。以下使用原型仅为界面布局参考，原型网站实际内容与本系统内容可能无关，请忽略实际内容，这里仅展示界面参考。

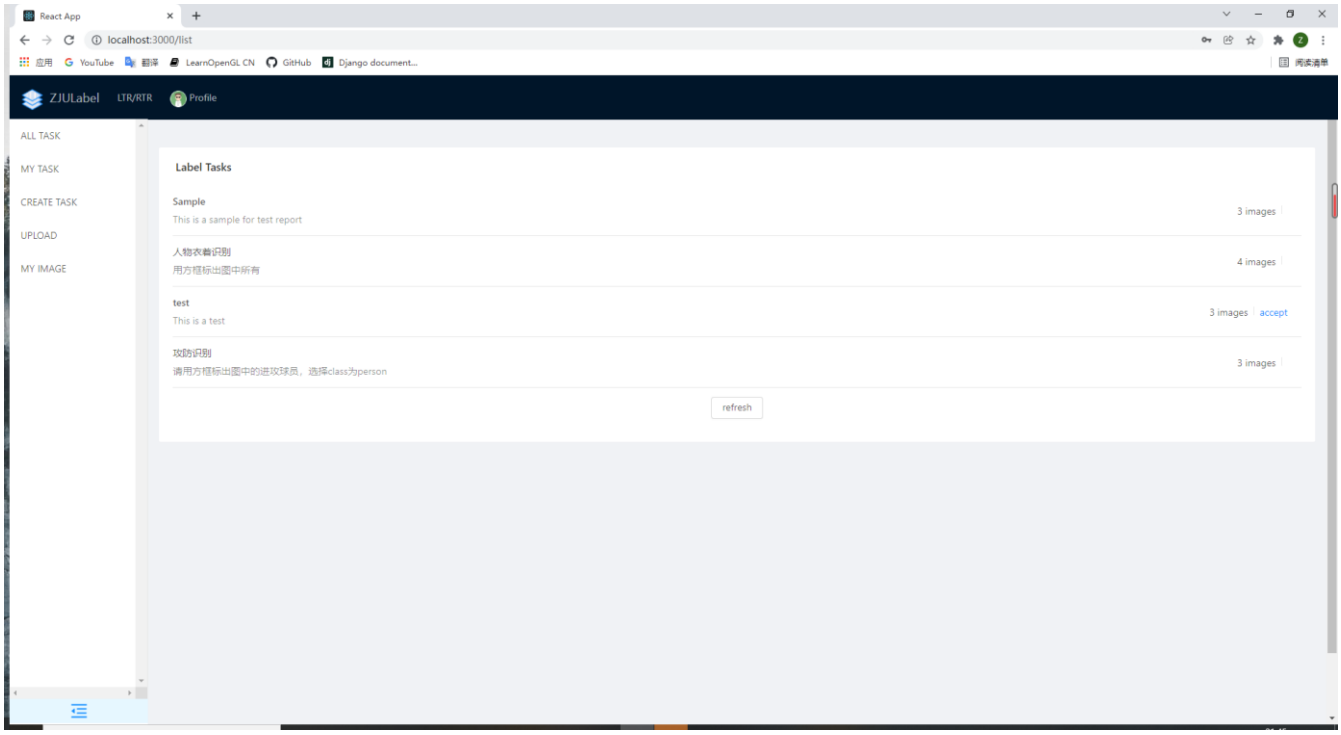
6.1. 登陆界面



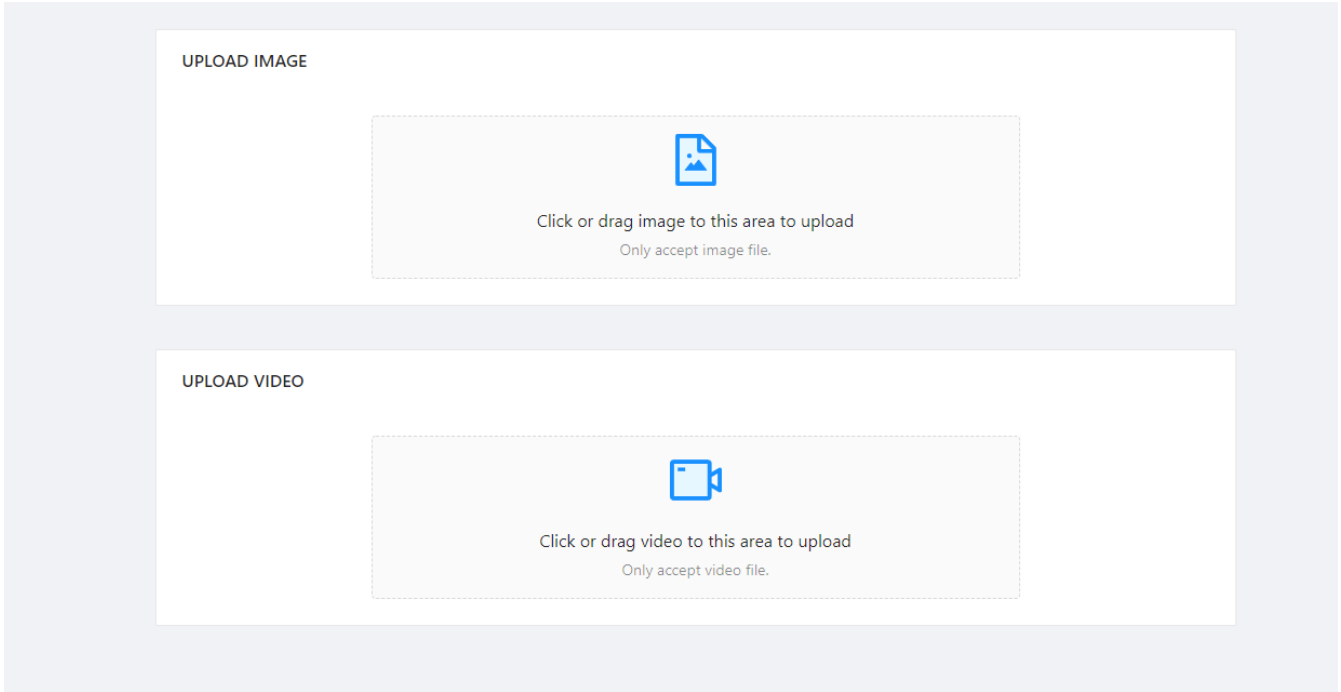
6.2. 注册界面


ZJULabel
Name
Email
Password
Register
Already have an account? Log in

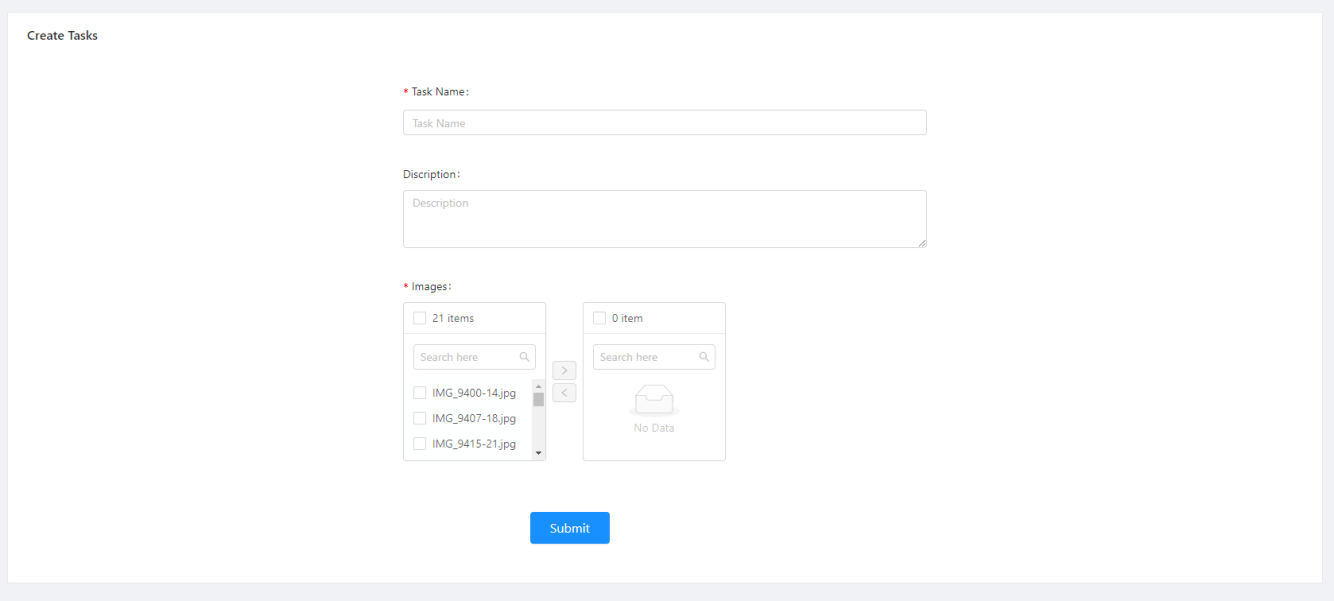
6.3. 主体界面



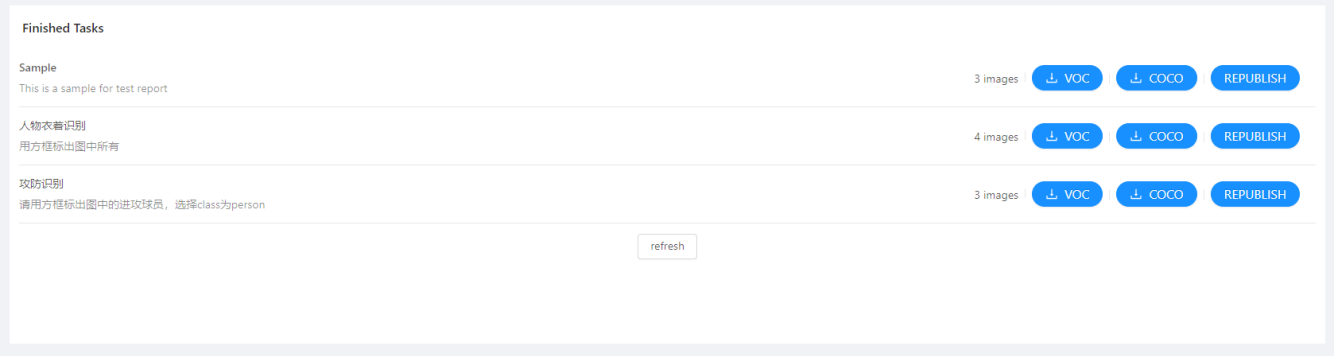
6.4. 图片上传界面



6.5. 图片发布界面


Create Tasks
* Task Name:
Task Name
Discription:
Description
* Images:
21 items
Search here
IMG_9400-14.jpg
IMG_9407-18.jpg
IMG_9415-21.jpg
0 item
Search here
No Data
Submit

6.6. 数据集下载界面


Finished Tasks
Sample
This is a sample for test report
3 images
VOC
COCO
REPUBLISH
人物衣着识别
用方框标出图中所有
4 images
VOC
COCO
REPUBLISH
攻防识别
请用方框标出图中的进攻球员，选择class为person
3 images
VOC
COCO
REPUBLISH
refresh

7.附录

7.1. 项目进度安排

时间	计划进度
2021 年 11 月	完成设计和技术栈学习，完成基本框架搭建
2021 年 12 月 1 日至 20 日	完成系统的前后端代码开发
2021 年 12 月 21 日至 27 日	完成系统集成与测试
2021 年 12 月 28 日至 2022 年 1 月 5 日	完成测试报告、用户手册等各个文档编写

7.2. 附录

本设计文档中并非定稿，应以最终提交的系统源代码和文档为准。