1. 引言

1.1 目的

为了明确用户的需求并较好地与开发人员进行沟通，使用户与开发人员双方对软件需求取得共同理解的基础上达成协议，特编写此文档，并作为整个软件开发的基础。

1.2 背景

在当今时代，传统农业正逐步向智能化、精准化转型，以适应资源高效利用与可持续发展的需求。长期以来，农业生产多依赖于经验积累与人力投入，面对复杂多变的病虫害挑战，往往缺乏有效的即时诊断手段。与此同时，随着现代生活条件的不断提升，越来越多的人开始拥有自己的小块土地，用于满足个人的种植爱好和绿色生活的追求。然而，对于这部分种植爱好者而言，缺乏专业的农业知识和经验，常常在面对农作物病虫害等问题时感到束手无策。

在这个背景下，图像识别、传感器等前沿科技为农业种植领域带来了新的曙光。这些技术不仅能够帮助农民实现精准农业，提高生产效率，还能够为种植爱好者提供强有力的支持。通过运用图像识别算法和大模型技术，可以实现对农作物病虫害等问题的快速、准确识别，为种植者提供及时的解决方案。这类农作物识别软件，能够服务于农民、种植爱好者、科研人员和政府机构等多个群体，帮助他们快速识别农作物及其潜在问题，并提出科学合理的解决方案，从而推动农业领域的智能化、精准化发展。

1.3 参考资料

《高级软件测试技术》 -- 杜庆峰

1.4 术语

1. 地块：某个用户种植某一种植物的一片区域，一个地块上有且只有一种植物
2. yolo模型：You Only Look Once，是一种目标检测算法，主要用于在图像中同时定位（检测）和分类多个物体，具有实时性能好、速度快、准确率高的特点。
3. 智能检测：通过yolo模型识别出植物的病害
4. 用户：使用该软件服务的群体
5. 账户：为用户提供服务的最小单位

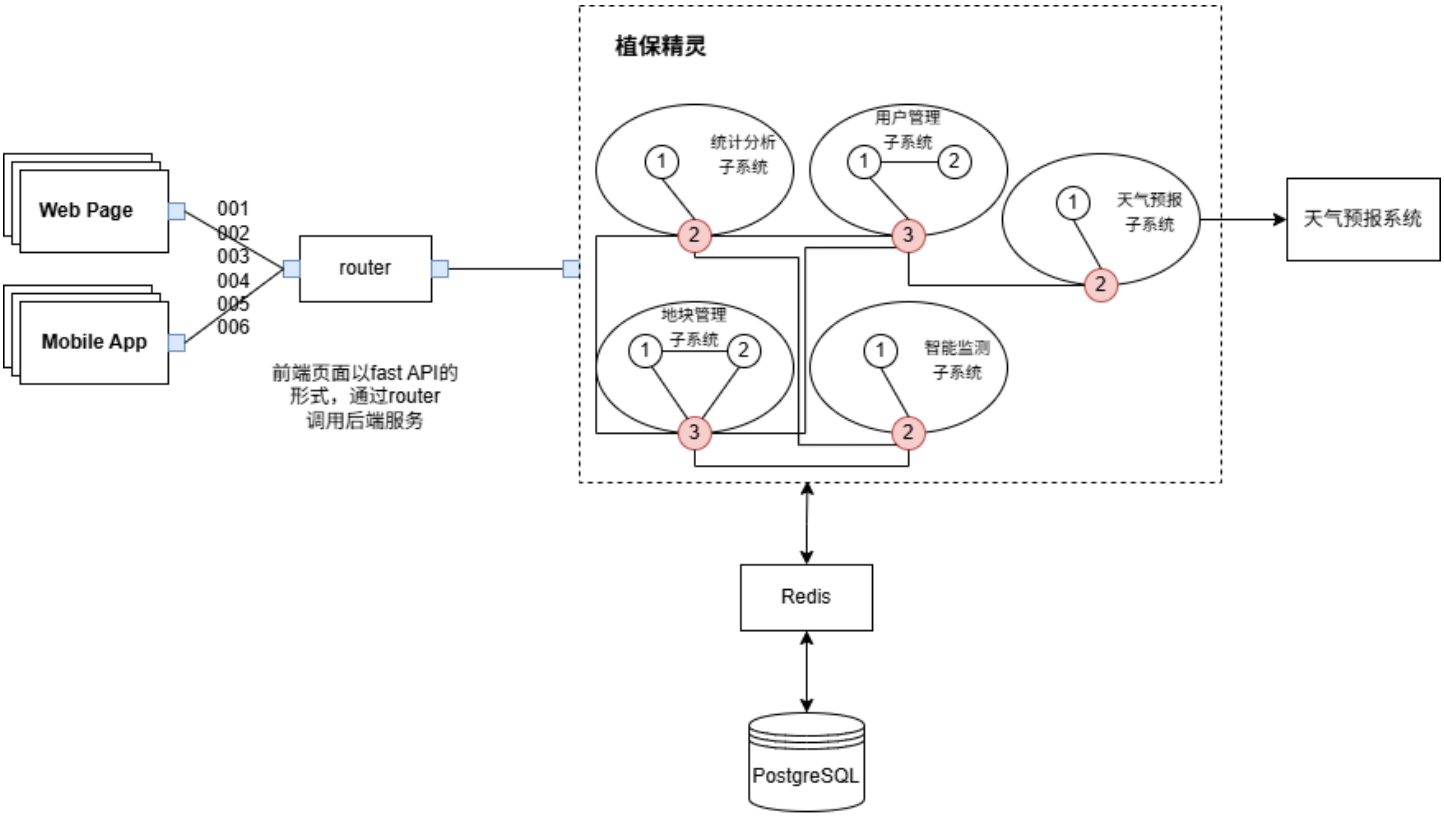
2. 项目概述

2.1 开发软件的一般描述

通过运用图像识别算法和大模型技术，实现对农作物病虫害等问题的快速、准确识别，为种植者提供及时的解决方案。

2.2 开发软件的功能描述

该软件是前后端分离的架构，前端通过为用户提供交互界面，后端提供前端需要的服务。除此之外，使用高德第三方api来获取实时天气信息。



2.3 实现语言

使用python和vue作为主要开发语言。

2.4 用户特点

本项目的用户包括：职业农民、种植爱好者、农业研究人员、农业管理机构。

1. 对很多职业农民来说，文化程度较低，接触到高科技智能设备的机会少，所以本产品应该设计简单的图形化界面，方便用户使用。
2. 对大多数种植爱好者来说，通常能接触到智能设备，但是对病虫害防治的知识了解程度不高，所以本产品应该提供精准而详尽的病虫害识别和防治流程建议。
3. 对农业研究人员和农业管理机构来说，他们更关注某地块植物在几个种植周期内的变化，本系统为他们提供了较为强大的地块跟踪功能。

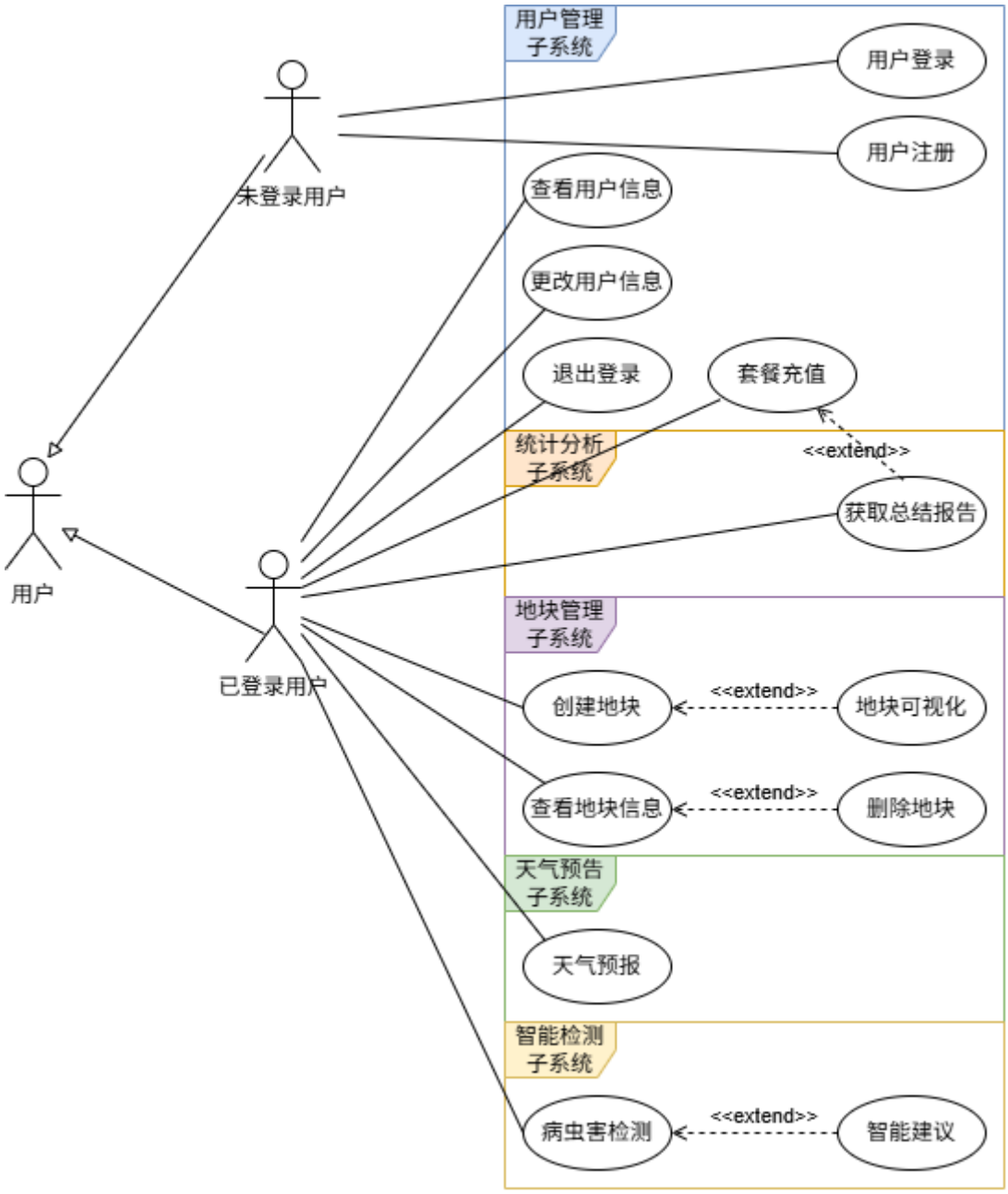
2.5 一般约束

主要约束是时间限制。

3. 需求说明

3.1 基本描述

系统的主要用例图如下所示：



3.2 功能需求

3.2.1 用户管理子系统(UM)

用例	用户注册(Sign Up)
ID	UM01
参与者	未登录用户

前置条件	用户未登录，访问注册登录页面
后置条件	用户成功注册，账户信息存储在系统中
基本流程	1. 用户点击注册选项，进入注册页面 2. 用户提交用户基本信息(用户名，密码，所在地) 3. 系统将用户信息存储在数据库中 4. 系统向用户发送注册成功的确认信息
异常流程	用户取消注册，点击返回，返回注册登陆页面

用例	用户登录(Sign In)
ID	UM02
参与者	未登录用户
前置条件	用户未登录，访问注册登录页面
后置条件	用户成功登录，进入系统主界面
基本流程	1. 用户点击登录选项，进入登录页面 2. 用户输入用户名和密码 3. 系统验证用户名和密码 4. 若验证通过，系统将用户重定向至主界面
异常流程	1. 用户名不存在，提示用户名无效，等待用户重新输入 2. 密码错误，提示密码错误，等待用户重新输入 3. 用户取消登录，点击返回，返回注册登录界面

用例	查看用户信息(Check User Info)
ID	UM03
参与者	已登录用户
前置条件	用户处于登陆状态
后置条件	用户获取个人基本信息和会员信息
基本流程	1. 用户在主界面，点击访问用户信息界面
异常流程	用户信息加载失败

用例	更改用户信息(Change User Info)
ID	UM04
参与者	已登录用户
前置条件	用户处于登陆状态，访问用户信息界面
后置条件	用户信息更改成功，返回用户信息界面并显示新的信息
基本流程	1. 用户在用户信息界面点击更改信息 2. 用户在更改信息界面，直接更改原有信息 3. 改完后点击保存 4. 在弹出的窗口点击确定
异常流程	1. 用户更改信息错误，点击了保存，提示错误信息 2. 用户不点保存直接退出，更改取消，不改变原有信息 3. 用户在弹出的窗口点击取消，取消保存，回到更改信息界面继续更改

用例	退出登录(Log Out)
ID	UM05
参与者	已登录用户
前置条件	用户处于登陆状态
后置条件	用户退出登录，返回注册登录界面
基本流程	1. 用户在主界面，点击访问用户信息界面 2. 在用户信息界面，点击退出登录 3. 在弹出的窗口点击确定
异常流程	用户在弹出的窗口点击取消，取消退出登录，弹窗关闭

用例	套餐充值(Recharge)
ID	UM06
参与者	已登录用户
前置条件	用户已登录，进入用户信息界面
后置条件	成功充值获得总结次数

基本流程	1. 用户在用户信息页面，滑动选择套餐 2. 用户点击想购买的套餐 3. 用户完成付款
异常流程	用户在付款界面点×，取消付款，返回用户信息页面

3.2.2 地块管理子系统(PM)

用例	创建地块(Create Plot)
ID	GM01
参与者	已登录用户
前置条件	用户已登陆
后置条件	成功创建地块
基本流程	1. 用户在主界面点击创建地块按钮，进入创建地块界面 2. 用户在创建地块界面输入地块植物基本信息 3. 用户点击创建地块按钮
异常流程	未输入足够的信息时，点击创建地块，提示缺失信息，继续输入信息

用例	地块可视化(Plot Visual)
ID	GM02
参与者	已登录用户
前置条件	用户已登录，有未删除的地块
后置条件	显示地块植物生长情况的卡通画面
基本流程	1. 系统读取用户上传的地块植物信息，自动生成地块卡通形象 2. 用户在主界面查看，可以看到所有创建地块的卡通形象
异常流程	

用例	查看地块信息(Plot Info)
ID	GM03
参与者	已登录用户

前置条件	用户已登录，有未删除的地块
后置条件	系统返回地块的详细信息
基本流程	1. 用户在主界面点击地块图标，进入地块信息页面 2. 系统在数据库中找到此包括地块日志在内的地块详细信息，呈现给用户
异常流程	

用例	删除地块(Delete Plot)
ID	GM04
参与者	已登录用户
前置条件	用户已登录，进入地块信息页面
后置条件	删除此地块及其所有信息，返回主界面
基本流程	1. 用户在地块信息页面，点击删除地块按钮 2. 在弹窗中选择确定
异常流程	在弹窗中选择取消，弹窗关闭，回到地块信息页面

3.2.3 气候预告子系统(WD)

用例	天气预报(Weather Forecast)
ID	WD01
参与者	已登录用户
前置条件	用户已登录，设备联网
后置条件	显示此地一周的天气和温度信息
基本流程	1. 用户在首页查看天气资料卡 2. 系统通过API获取所在地的气象信息，呈现给用户
异常流程	设备未联网，提示未连接互联网

3.2.4 智能检测子系统(PD)

用例	病虫害检测(Disease Detection)

ID	PD01
参与者	已登录用户
前置条件	用户已登录，有未删除的地块
后置条件	系统返回病虫害识别结果
基本流程	1. 用户在主界面点击要检测的地块图标，进入地块信息页面 2. 用户点击病虫害检测按钮，进入检测页面 3. 用户上传拍摄的视频，点击开始检测 4. 算法对视频进行截图分析，同时将有效截图上传日志 5. 系统使用病害检测算法处理图片，返回病虫害的识别结果
异常流程	

用例	智能建议(Smart Suggest)
ID	PD02
参与者	已登录用户
前置条件	用户已进行病虫害检测
后置条件	系统返回数据库的病虫害防治建议
基本流程	1. 用户在检测完病虫害后系统会根据检测结果给出病虫害的防治建议
异常流程	

3.2.5 统计分析子系统(AS)

用例	获取总结报告(Annual Report)
ID	AS01
参与者	已登录用户
前置条件	用户已登录，且还有总结次数用户已进行病虫害检测
后置条件	显示有记录以来每年的病虫害数据，并给出分析建议
基本流程	1. 用户在用户信息页面，点击获取总结报告 2. 在弹窗选择确认生成 3. 系统扣除用户总结次数，进入总结页面

	4. 显示有记录以来每年的病虫害数据，并给出分析建议 5. 点击保存按钮，可以下载总结报告
异常流程	1. 用户暂无信息，显示暂无病虫害数据，点击返回地块信息界面 2. 用户总结次数耗尽，提示总结次数已用完，充值获得总结次数

3.3 性能及其他需求

1. 用户从上传图片到完成疾病检测的结果应该在5s之内完成。
2. 对于疾病检测以外的操作，应该在1s内完成
3. 系统应该支持至少100人并行访问

3.4 对输入输出的规定

1. 疾病检测时上传的图片应该为常见的图片格式，如jpg、png
2. 用户注册的账户名必须是唯一的，不能重复
3. 用户识别的植物必须已有模型训练支持

3.5 特殊需求

3.5.1 易用性需求

该系统的主要目标用户是农民等农业工作者，农业工作者一般年龄较大，对新技术的熟悉程度较低。为了能让农业工作者快速上手该系统，系统的用户界面必须简单明了，符合常识，易于理解，而无须经过培训或说明。

3.5.2 扩展性需求

该系统的开发采取敏捷开发原则，先从少数几种植物的病虫检测触发，逐步扩展检测的范围并提高检测的精度。该系统应该具有高度可扩展性，使得能够灵活调整系统的组件与功能。系统使用微服务框架，将不同模块之间解耦合。

3.5.3 安全性需求

系统必须要有安全性。具体而言：该系统应该使用加密、匿名化等技术，以及有效的访问控制机制，以保护用户的隐私；该系统应该具有可靠性，使用可靠的服务框架和算法技术，给用户带来稳定的使用体验；该系统应该验证数据的完整性。通过哈希等方法确保数据没有发生损坏，避免带来非预期的影响。

4. 采用的测试方法

- 系统级功能
- 系统业务流

- 系统级别的接口

5. 测试环境

5.1 硬件需求

- 后端服务器一台
- 测试主机一台

5.2 软件需求

- 应用软件测试版
- python环境
- node.js环境
- Linux操作系统
- Windows操作系统
- PostgreSQL数据库

5.3 网络需求

- 8MB以上带宽

6. 测试对象

PGuard系统

应用软件版本：sys_test_tag

7. 测试分析与用例设计

7.1 测试数据准备

账号	密码	地点	拥有地块
Jelly	lyy3366291	上海市	葡萄
testuser	123	北京市	葡萄

7.2 基于系统级功能的测试分析与用例设计

用例编号	Pguard_Sys_Test_case_fun_001

测试覆盖的系统业务	用户进行地块检测					
用例设计方法	场景法					
前置条件	用户已注册账户且至少有一个地块					
输入						
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**		
用户访问网站	输入：用户输入用户密码并登录 输出：跳转到首页	输入：用户打开地块 输出：跳转到地块详细信息	输入：用户点击“病害检测” 输出：弹出选项框，选择图片	输入：用户提交图片		
最后预期输出						
检测成功，返回结果						

用例编号	**Pguard_Sys_Test_case_fun_002**					
测试覆盖的系统业务	用户查看天气信息					
用例设计方法	场景法					
前置条件	用户已有账户					
输入						
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**				
用户访问网站	输入：用户输入用户密码并登录 输出：跳转到首页	输入：无 输出：获取用户当地天气信息				
最后预期输出						
显示用户当地天气信息						

用例编号	Pguard_Sys_Test_case_fun_003					
测试覆盖的系统业务	用户创建地块					
用例设计方法	场景法					
前置条件	用户已有账户					
输入						
初始输入	预期交互输入/输出1	预期交互输入/输出2	预期交互输入/输出3			
用户访问网站	输入：用户输入用户密码并登录 输出：跳转到首页	输入：用户点击“新建地块” 输出：弹出弹窗	输入：用户选择地块植物种类并提交 输出：创建地块成功			
最后预期输出						
创建地块成功						

用例编号	Pguard_Sys_Test_case_fun_004					
测试覆盖的系统业务	用户查看检测日志					
用例设计方法	场景法					
前置条件	用户已有账户、已有地块、至少有一次检测					
输入						
初始输入	预期交互输入/输出1	预期交互输入/输出2				
用户访问网站	输入：用户输入用户密码并登录 输出：跳转到首页	输入：用户打开地块 输出：跳转到地块详细信息				
最后预期输出						
显示检测日志						

用例编号	Pguard_Sys_Test_case_fun_005					
测试覆盖的系统业务	用户注册账户					
用例设计方法	场景法					
前置条件	无					
输入						
初始输入	预期交互输入/输出1	预期交互输入/输出2	预期交互输入/输出3	预期交互输入/输出4		
用户访问网站	输入：用户点击注册 输出：跳转到注册页	输入：用户输入用户名密码 输出：无	输入：用户输入城市 输出：搜索城市列表	输入：用户选择城市并提交 输出：注册成功		
最后预期输出						
注册成功						

7.3 基于系统业务流测试分析与用例设计

用例编号	Pguard_Sys_Test_case_business_001				
测试覆盖的系统业务	未注册用户完成注册并登录系统				
用例设计方法	场景法 + 等价类划分				
前置条件					
输入					
初始输入	预期交互输入/输出1	预期交互输入/输出2	预期交互输入/输出3	预期交互输入/输出4	预期交互输入/输出5
用户访问网站	输入：点击"注册"按钮；	输入：填写有效用户名/密码/所在地；	输入：点击"提交注册"；	输入：自动跳转至登录页面；	输入：用户填写正确的用户名和

	输出：跳转至注册页面	输出：表单验证通过	输出：显示"注册成功"提示	输出：用户可正常填写用户名、密码进行登录	密码并点击登录按钮； 输出：用户登录成功
最后预期输出					
用户成功登录，进入系统主界面					

用例编号	**Pguard_Sys_Test_case_business_002**			
测试覆盖的系统业务	用户完整的病害检测业务流			
用例设计方法	状态迁移法 + 场景法			
前置条件	用户已登录且已创建地块			
输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
用户在主界面查看地块列表	输入：GET /plot 请求获取地块列表；输出：返回地块信息包含plotId、plotName、plantName、plantIconURL	输入：POST /plot/{plotId}/detect 上传图片文件； 输出：文件验证通过，开始检测处理	输入：YOLO模型处理图片； 输出：返回disease、confidence检测结果	输入：系统自动获取防治建议并记录日志； 输出：返回完整检测结果包含diseaseName、advice、percent、imageURL
最后预期输出				
用户获得完整的病害检测结果，包括病害名称、置信度、防治建议，系统自动生成检测日志				

用例编号	**Pguard_Sys_Test_case_business_003**
测试覆盖的系统业务	用户充值套餐到获取统计分析的完整流
用例设计方法	场景法
前置条件	用户已登录，sumCount为0，已有历史检测数据

输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
用户尝试GET /summary获取统计报告	输入：系统检查user.sumCount； 输出：返回400错误"余额不足，请充值"	输入：GET /package获取套餐列表； 输出：返回可购买的套餐信息	输入：POST /recharge/{package_id}购买套餐； 输出：user.sumCount增加，返回充值成功信息	输入：重新GET /summary获取统计报告； 输出：成功返回包含plot_count、monthly_disease_count、prediction等统计信息
最后预期输出				
用户成功充值并获取统计分析报告，包含地块统计、病害趋势、预测建议等完整信息				

用例编号	**Pguard_Sys_Test_case_business_004**
测试覆盖的系统业务	地块完整生命周期管理业务流
用例设计方法	状态迁移法
前置条件	用户已登录

输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
用户查看可用植物类型	输入：GET /plot/plant获取植物列表； 输出：返回系统支持的植物名称列表	输入：POST /plot/add创建地块{plotName, plantName}； 输出：成功创建地块，返回plotId和创建信息	输入：GET /plot/{plotId}查看地块详情； 输出：返回PlotDetails包含logs、plantFeature等详细信息	输入：DELETE /plot/{plotId}删除地块； 输出：地块删除成功，返回确认信息
最后预期输出				
完成地块从创建到删除的完整生命周期，验证地块管理功能的完整性				

用例编号	**Pguard_Sys_Test_case_business_005**

测试覆盖的系统业务	用户城市定位和信息管理业务流
用例设计方法	场景法
前置条件	用户已登录，系统已导入城市数据

输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
用户搜索城市信息	输入：GET /city/{keyword} 搜索城市； 输出：返回匹配的城市列表包含cityName和cityCode	输入：GET /city 获取用户城市； 输出：根据user.location返回对应城市信息	输入：PATCH /update 更新用户信息； 输出：验证新location并更新用户信息	输入：GET /me 获取用户信息； 输出：返回更新后的用户信息包含新的location
最后预期输出				
用户成功搜索城市、更新位置信息，系统正确关联城市数据				

7.4 系统级别的接口的测试分析与用例设计

用例编号	**Pguard_Sys_Test_case_interface_001**
测试覆盖的系统业务	用户认证接口集成测试
用例设计方法	边界值分析 + 等价类
前置条件	系统正常运行，数据库包含测试用户数据

输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
前端发送登录请求	输入：POST /signin {userName:"Jelly",	输入：使用无效用户名登录；	输入：使用错误密码登录；	输入：POST /refresh 刷新token；

	password:"lyy3366291"}; 输出：返回200和access_token、refresh_token	输出：返回400 "用户不存在"	输出：返回400 "密码错误"	输出：返回新的access_token
最后预期输出				
认证接口正确处理各种登录场景，token刷新机制正常工作				

用例编号	**Pguard_Sys_Test_case_interface_002**			
测试覆盖的系统业务	图像检测接口端到端数据流测试			
用例设计方法	场景法			
前置条件	用户已登录，地块已创建，YOLO模型正常加载			
输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
上传.jpg格式图片文件	输入：POST /plot/{plotId}/detect 上传图片；输出：文件格式验证通过，保存到指定路径	输入：调用detect()函数处理图片；输出：YOLO模型返回disease和confidence结果	输入：get_advice()获取防治建议； 输出：从Disease表获取对应advice内容	输入：call_set_log()记录检测日志； 输出：日志成功写入数据库，返回完整检测结果
最后预期输出				
图像从上传到检测结果存储的完整链路正常，返回包含diseaseName、advice、percent、imageURL的结果				

用例编号	**Pguard_Sys_Test_case_interface_003**
测试覆盖的系统业务	管理员数据导入接口测试
用例设计方法	等价类（有效等价类）

前置条件	具有管理员权限，CSV文件格式正确			
输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
管理员上传城市数据CSV	输入：POST /admin/weather/city_input?csvURL=amap.csv； 输出：validate_city_file()验证文件格式	输入：读取CSV文件内容； 输出：解析城市名称和代码数据	输入：City.all().delete() 清空现有数据； 输出：旧数据清理完成	输入：City.bulk_create() 批量插入； 输出：成功导入城市数据，返回导入数量
最后预期输出				
CSV数据成功导入数据库，城市信息可正常被用户搜索和使用				

用例编号	Pguard_Sys_Test_case_interface_004			
测试覆盖的系统业务	跨service层数据一致性测试			
用例设计方法	场景法			
前置条件	用户已登录，系统各service模块正常运行			
输入				
初始输入	**预期交互输入/输出1**	**预期交互输入/输出2**	**预期交互输入/输出3**	**预期交互输入/输出4**
service.plot.add_plot()创建地块	输入：验证Plant.get(plantName)存在； 输出：Plot.create()成功创建地块记录	输入：service.detect.do_detect()检测病害；输出：validate_plot_access()验证用户权限通过	输入：call_set_log()创建检测日志； 输出：Log表中新增记录关联plotId	输入：service.log.get_summary()获取统计；输出：analyze_plot_details()正确统计所有地块数据
最后预期输出				
各service层间数据传递一致，用户操作在所有相关模块中都正确体现				

用例编号	Pguard_Sys_Test_case_interface_005			
测试覆盖的系统业务	文件系统和数据库接口协调测试			
用例设计方法	场景法+错误推测法			
前置条件	文件系统权限正常，数据库连接稳定			
输入				
初始输入	预期交互输入/输出1	预期交互输入/输出2	预期交互输入/输出3	预期交互输入/输出4
上传检测图片文件	输入：生成uuid文件名保存到/resource/log/； 输出：文件物理存储成功	输入：数据库记录imageURL路径； 输出：Log表中imagesURL字段记录正确路径	输入：GET /plot/{plotId}获取地块详情； 输出：logs数组包含正确的图片URL	输入：validate_image_file()验证图片； 输出：文件存在性和格式验证通过
最后预期输出				
文件存储和数据库记录保持一致，图片文件可通过URL正常访问				

8. 测试结果

（仅展示部分）

Running '创建地块'

1. open on /login OK
2. setWindowSize on 1722x1034 OK
3. click on id=ion-input-0 OK
4. click on id=ion-input-0 OK
5. type on id=ion-input-0 with value Jelly OK
6. click on css=.item:nth-child(2) .input-wrapper OK
7. type on id=ion-input-1 with value lyy3366291 OK
8. click on css=.button-solid OK
9. Trying to find css=.ion-activated... OK

Warning Element found with secondary locator xpath=//div[@id='app']/ion-app/ion-router-outlet/div[2]/ion-tabs/ion-router-outlet/div/ion-fab/ion-fab-button. To use it by default, update the test step to use it as the primary locator.

10. click on css=.sc-ion-input-md-h:nth-child(2) .native-wrapper OK
11. click on id=ion-input-2 OK
12. type on id=ion-input-2 with value 土豆地块 OK
13. click on css=.has-placeholder OK
14. click on css=#alert-input-3-1 .alert-radio-label OK
15. Trying to find css=.ion-activated > .alert-button-inner... OK

Warning Element found with secondary locator xpath=//ion-alert[@id='ion-overlay-2']/div[2]/div[4]/button[2]/span. To use it by default, update the test step to use it as the primary locator.

16. click on css=.button-full OK
17. click on css=.alert-button OK

'创建地块' completed successfully

```
Running '查看天气'
1. open on /login OK
2. setWindowSize on 1722x1034 OK
3. click on id=ion-input-0 OK
4. type on id=ion-input-0 with value Jelly OK
5. click on id=ion-input-1 OK
6. type on id=ion-input-1 with value lyy3366291 OK
7. click on css=.button-solid OK
'查看天气' completed successfully
```

9. 性能测试

9.1 测试目的

负载测试在性能测试中是一个重要的环节，其主要目的是通过逐步增加负载，测试系统性能变化，确定系统在正常或峰值负载下的表现，以评估系统的稳定性、可靠性和性能指标。性能测试旨在对P-Guard农业管理系统进行全面的性能评估，主要目标包括：

- **系统稳定性验证**：确保系统在高并发负载下能够稳定运行，不出现系统崩溃或服务中断
- **响应时间评估**：验证核心业务接口在不同负载条件下的响应时间是否满足用户体验要求
- **系统吞吐量测试**：确定系统在单位时间内能够处理的最大请求数量
- **资源利用率分析**：评估系统在高负载下的CPU、内存等资源使用情况
- **性能瓶颈识别**：发现系统中可能存在的性能瓶颈点，为系统优化提供依据
- **容量规划支持**：为系统部署和扩容提供数据支撑

9.2 测试工具

选用Apache JMeter作为主要的性能测试工具，具体配置如下：

工具选择理由：

- 开源免费，功能强大，支持多种协议测试
- 提供直观的GUI界面，便于测试脚本编写和调试
- 支持分布式测试，可模拟大规模并发用户
- 具备丰富的报告生成功能，便于结果分析
- 支持参数化和数据驱动测试

测试环境配置：

- JMeter版本：5.6.3
- Java环境：JDK 23
- 操作系统：Windows 11（客户端）/Ubuntu 22.04（服务器）
- 网络环境：稳定的网络连接，延迟< 50ms

9.3 测试需求分析

系统架构分析： P-Guard系统采用前后端分离架构，后端提供RESTful API服务，前端通过HTTP请求与后端交互。系统核心功能包括用户认证、地块管理。

关键业务流程：

1. 用户登录认证流程
2. 地块信息查询

性能指标要求：

非硬件指标：

- **响应时间要求**：
  - 用户登录接口：< 5秒（涉及加密验证等CPU密集型操作）
  - 查看地块接口：< 0.5秒
- **并发用户数**：支持100-200并发用户
- **事务成功率**>99%
- **错误率**：< 1%

硬件指标：

- **CPU 、内存占用率不超过80%**

测试数据需求：

- 用户账号数据：200个有效测试账号
- 地块数据：每个用户关联5-10个地块信息

9.4 测试方案设计

测试策略： 采用分层递进的测试策略，从基础功能验证到极限压力测试，确保全面覆盖系统性能特征。

测试场景设计：

场景一：基准性能测试

- **目标**：建立系统性能基准线
- **并发用户数**：10-50用户
- **持续时间**：5分钟

场景二：负载压力测试

- **目标**：验证系统在预期负载下的性能表现
- **并发用户数**：100-200用户
- **持续时间**：5分钟

- **负载模式**：阶梯式增长

测试数据配置：

- 用户登录：使用CSV文件配置200个用户账号信息
- 地块查询：为每个用户配置多个地块ID

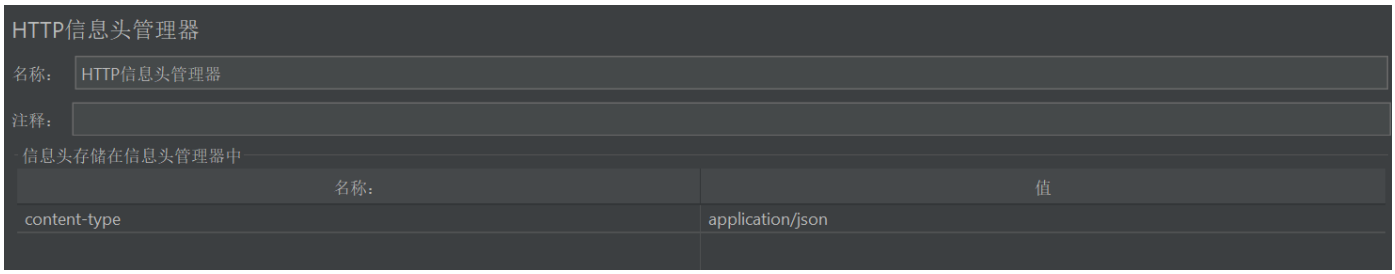
9.5 测试过程

1. **环境准备**：
    - 搭建独立的性能测试环境
    - 部署应用服务，确保服务正常运行
    - 配置监控工具，收集系统资源使用情况

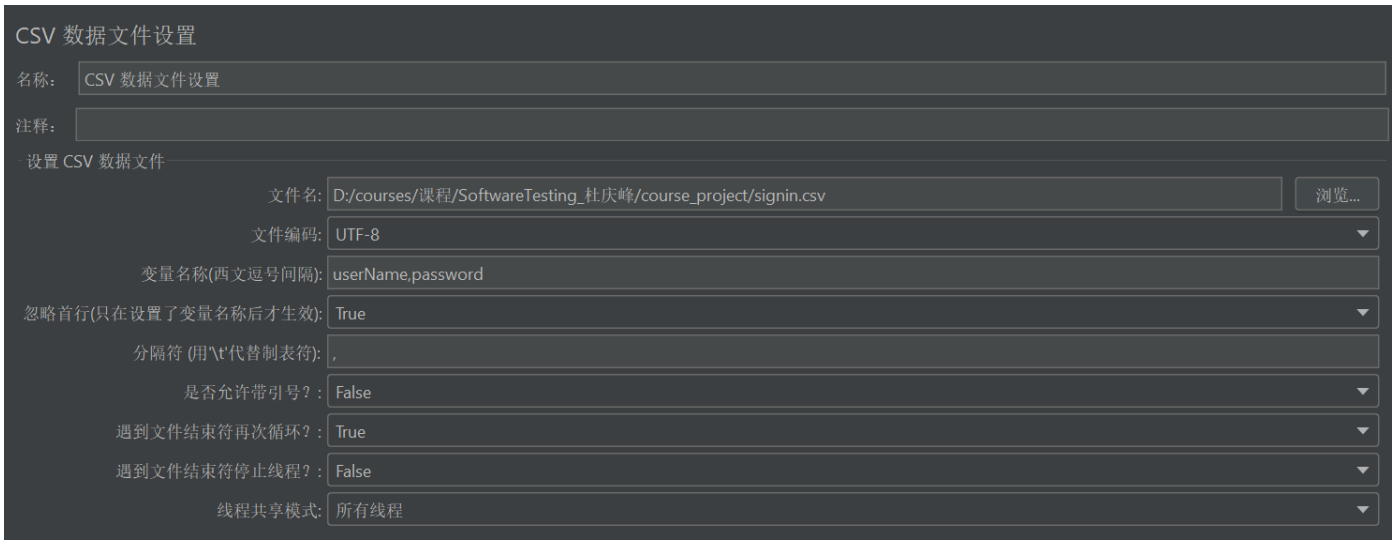
```
E:\apache-jmeter-5.6.3\apache-jmeter-5.6.3\bin>java -jar ApacheJMeter.jar
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
================================================================================
Don't use GUI mode for load testing !, only for Test creation and Test debugging.
For load testing, use CLI Mode (was NON GUI):
   jmeter -n -t [jmx file] -l [results file] -e -o [Path to web report folder]
& increase Java Heap to meet your test requirements:
   Modify current env variable HEAP="-Xms1g -Xmx1g -XX:MaxMetaspaceSize=256m" in the jmeter batch file
Check : https://jmeter.apache.org/usermanual/best-practices.html
================================================================================
```

2. **脚本开发**：
    - 设置线程组

    - 配置HTTP请求



- 配置csv文件并引用



- 配置断言

- 添加监视器及报告生成

3. **测试执行：**

启动测试脚本，同时使用sysstat进行服务器状态监控，包括CPU占用率，内存占用率等。

4. **测试结果收集：**

使用JMeter的HTML报告及响应时间图统计测试结果情况，分析响应时间、吞吐量和错误率等指标，以评估系统的性能表现。

5. **性能优化与改进：**

根据测试结果进行性能优化和调整。根据发现的性能缺陷，定位问题所属的系统业务模块，并采取相应的优化措施，如代码优化、资源调整或系统配置修改等。

9.6 测试结果

9.6.1 登录

基础性能：

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
15	10	300	1063	3.5	51.14%	94.40%	4195.3	64421	100%

负载压力测试：

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
20	10	300	1069	3.51	51.17%	94.45%	5568.5	105362	100%

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
30	10	300	1078	3.5	51.15%	94.77%	8327.01	114688	100%

9.6.2 查看地块

基础性能：

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
15	10	300	64064	213.62	34.26%	92.40%	68.69	820	100%

负载压力测试：

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
50	10	300	92811	309.19	51.53%	94.56%	156.91	611	100%

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
100	10	300	84901	282.53	52.45%	94.35%	353.95	1233	100%

线程数	启动时长(秒)	运行时长(秒)	样本	TPS(T/S)	CPU使用率	内存使用率	平均响应时间(毫秒)	最大响应时间(毫秒)	成功率
100	10	300	84901	282.53	52.45%	94.35%	353.95	1233	100%

9.7 测试结论

1. 登录接口性能分析

基础性能表现（15线程）：

- TPS：3.5
- 平均响应时间：4,195.3ms
- 最大响应时间：64,421ms
- CPU使用率：51.14%
- 成功率：100%

负载压力测试表现：

- **20线程**：TPS保持3.51，响应时间增至5,568.5ms，最大响应时间达105,362ms
- **30线程**：TPS保持3.5，但平均响应时间大幅增至8327ms，最大响应时间激增至114,688ms

登录接口结论：

- **稳定性优秀**：在不同负载下均保持100%成功率
- **响应时间敏感**：随并发增加，响应时间呈显著上升趋势
- **吞吐量瓶颈**：TPS维持在3.5左右，提升空间有限
- **高并发表现不佳**：30线程时最大响应时间过长，用户体验较差

2. 查看地块接口性能分析

基础性能表现（15线程）：

- TPS：213.6
- 平均响应时间：68.7ms
- 最大响应时间：820ms
- CPU使用率：34.26%
- 成功率：100%

负载压力测试表现：

- **50线程**：TPS达309.19，响应时间仅156.91ms，表现优异
- **100线程**：TPS降至282.53，响应时间升至353.95ms，仍在可接受范围
- **150线程**：TPS下降至262.00，平均响应时间维持546.89ms，超出了需求范围

查看地块接口结论：

- **高吞吐量**：最高TPS可达309
- **响应速度快**：即使在高并发下，响应时间仍控制在500ms以内

- **扩展性良好**：150线程以内性能相对稳定
- **极高负载下略有波动**：150线程时响应时间超过0.5s

系统资源利用分析

CPU使用率：

- 登录接口：51-52%（相对稳定）
- 查看地块接口：34-53%（随负载增加而提升）

内存使用率：

- 两个接口均在92-95%之间，内存使用率较高但相对稳定

关键发现

1. **性能差异巨大**：查看地块接口的TPS是登录接口的约85倍，响应速度快30倍以上
2. **登录接口是系统瓶颈**：涉及复杂的认证、加密等耗时操作
3. **查看地块接口表现优异**：适合高并发场景，可承载大量用户同时查询

优化建议

登录接口优化：

- 优化认证算法，减少计算复杂度
- 引入缓存机制，避免重复计算
- 考虑异步处理非关键步骤
- 数据库连接池优化

系统整体优化：

- 内存使用率偏高（94%+），应监控内存泄漏风险
- 针对登录接口进行专项性能调优
- 建立监控告警机制，关注响应时间异常

部署建议：

- 生产环境建议登录接口并发控制在20线程以内
- 查看地块接口可支持100+并发用户
- 考虑负载均衡分散登录压力