**微景观小程序**

**系统详细设计说明书**

**2018年03月**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档修改情况记录 | | | | |
| 版本 | 修改状态 | 修改日期 | 修改摘要 | 修改人 |
| 1.0 | 创建 | 2018-03-20 | 创建详细设计文档1.0 | 陈煦 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**目 录**

[1 引言 1](#_Toc29067)

[1.1 编写目的 1](#_Toc25196)

[1.2 名称解释 1](#_Toc16650)

[2 总体设计 1](#_Toc477)

[2.1 设计方法 1](#_Toc6159)

[2.2 软件结构 2](#_Toc16228)

[2.2.1 总体流程图 2](#_Toc28627)

[2.2.2 总体结构 2](#_Toc32429)

[2.2.3 关注模块 2](#_Toc30558)

[2.2.4 发现模块 3](#_Toc19582)

[2.2.5 消息模块 3](#_Toc17078)

[2.2.6 文章模块 4](#_Toc9656)

[2.2.7 我的模块 4](#_Toc30818)

[3 模块设计 5](#_Toc26442)

[3.1 关注模块 5](#_Toc6156)

[3.1.1 需求概述 5](#_Toc14020)

[3.1.2 流程图 5](#_Toc2902)

[3.1.3 信息展示模块 5](#_Toc28774)

[3.1.4 推荐关注模块 12](#_Toc25980)

[3.2 发现模块 16](#_Toc25178)

[3.2.1 需求概述 16](#_Toc27194)

[3.2.2 流程图 17](#_Toc22175)

[3.2.3 文章推荐模块 17](#_Toc18195)

[3.1.4 专题推荐模块 24](#_Toc20417)

[3.3 消息模块 31](#_Toc30409)

[3.3.1 需求概述 32](#_Toc4332)

[3.3.2 私信模块 32](#_Toc23940)

[3.4.3 评论模块 39](#_Toc20126)

[3.4.4 喜欢模块 43](#_Toc5596)

[3.4.5 关注模块 44](#_Toc18256)

[3.4 文章模块 47](#_Toc29566)

[3.4.1 需求概述 47](#_Toc5026)

[3.4.2 流程图 48](#_Toc2428)

[3.4.3 阅读模块 48](#_Toc4240)

[3.5.4 写作模块 58](#_Toc29737)

[3.5 我的模块 61](#_Toc8665)

[3.5.1 需求概述 62](#_Toc32383)

[3.5.2 流程图 62](#_Toc8605)

[3.5.3 用户管理模块 63](#_Toc3710)

[3.5.4 文章管理 76](#_Toc211)

# **引言**

## 编写目的

此设计说明书是微信小程序开发的主要依据，用以描述程序系统结构，为开发人员提供工作基准文件，并对后续阶段的工作起指导作用。

预期读者为设该系统的系统分析员、程序员、测试员、实施人员等。

## 名称解释

|  |  |
| --- | --- |
| **名称** | **解释** |
|  |  |

# 总体设计

根据功能对系统进行划分，可分为智能设备模块、植物百科模块、社区模块、模块以及我的模块。

## 设计方法

本项目采用传统的软件开发生命周期方法和敏捷开发相结合，采用自顶向下，逐步求精的面向对象软件设计方法。

## 软件结构

### 总体流程图

### 总体结构

### 关注模块

### 发现模块

### 消息模块

### 文章模块

### 我的模块

# 模块设计

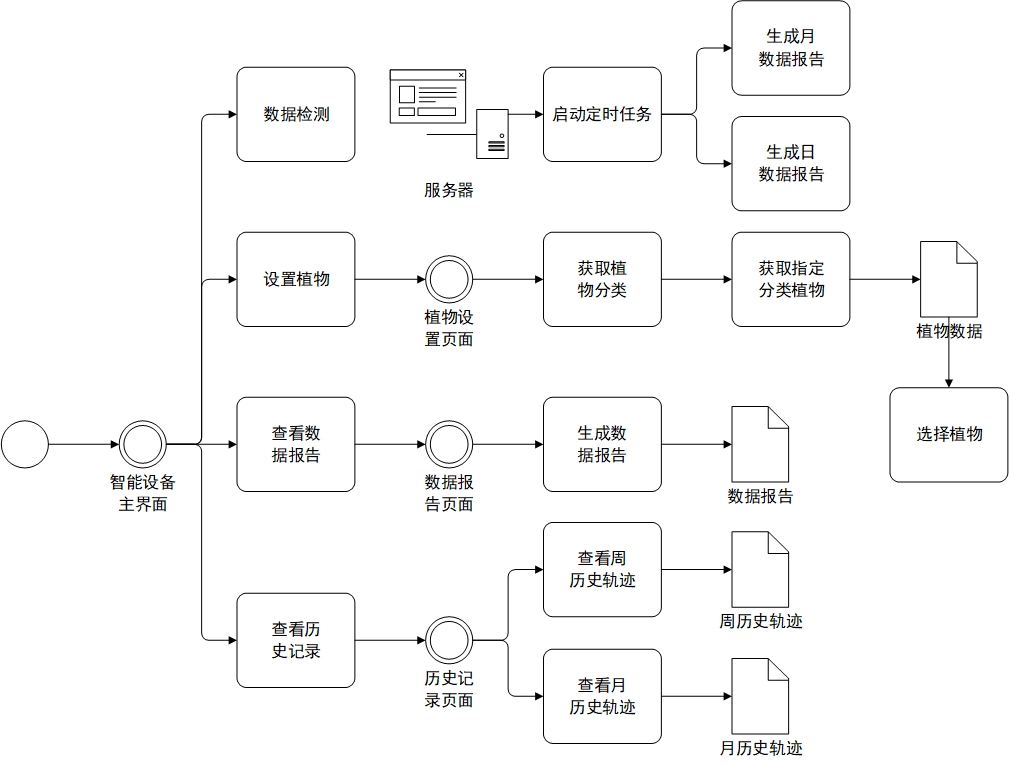
## 智能设备模块

智能设备模块用于呈现用户当前拥有的植物信息以及检测指定植物所得的数据。具体需求可分为植物信息、数据检测、生成数据报告以及查看历史记录。

### **需求概述**

1. 查看设备信息：用户进入智能设备页面即可查看已拥有的植物信息，包括植物的图像以及名称。
2. 数据检测：具体实现过程还需要进行讨论。
3. 设置植物：用户可以修改当前检测的植物种类。
4. 查看数据报告：用户能够主动生成数据报告，数据报告包括今日检测的次数，折线图形式显示的检测数据，以及计算所得的平均数值，相应的总评和养护建议。
5. 查看历史记录：用户可以查看最近一周或最近一月的历史记录，包括历史轨迹（折线图）以及每个时间节点的详细报告。

### 流程图



### **查看设备信息**

1. 具体描述：

用户进入智能设备页面即可查看已拥有的植物信息，包括植物的图像以及名称。若用户拥有多盆植物，则可以通过滑动植物图像来实现切换。

1. 具体流程：
2. 获取植物与设备信息
3. 模块描述

用户进入智能设备界面时，小程序自动获取当前用户拥有的植物信息（列表形式）。

1. 输入、输出

输入：自定义登录态、用户账号。

输出：标识码、植物号、植物图片（自定义图片）、植物名称、植物学名。

1. 模块说明

服务器后台接收到前端传递的参数后，根据用户账号从view\_device视图中检索指定数据，并返回给前端。

1. 数据库设计

### 设备信息表 ndevice

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Ocode | Char | 否 | 主键 | 主键，唯一标识智能设备 | 标识码 |
| Ipid | Char | 否 | 外键 | 外键，植物表外键 | 植物号 |
| Uaccount | Varchar | 否 | 外键 | 外键，用户表外键 | 用户账号 |
| Lpicture | Varchar | 是 | 无 | 用户可以上传植物图片 | 自定义图片 |

1. 视图设计

### 设备信息视图 view\_device

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| code | ndevice的Ocode字段，标识码 | 标识码 | ndevice |
| id | nplant的Pid字段，唯一标志植物信息 | 植物号 | nplant |
| name | nplant的Cname字段，植物名称 | 植物名称 | nplant |
| sname | nplant的Ksname字段，植物学名 | 植物学名 | nplant |
| picture | nplant的Ypicture字段，植物图片 | 植物图片 | nplant |
| alterPic | ndevice的Lpicture字段，自定义图片 | 自定义图片 | ndevice |
| account | ndevice的Uaccount字段，用户账号 | 用户账号 | ndevice |

1. 程序界面



（图：3.1.3.1 智能设备主界面）

1. 接口设计

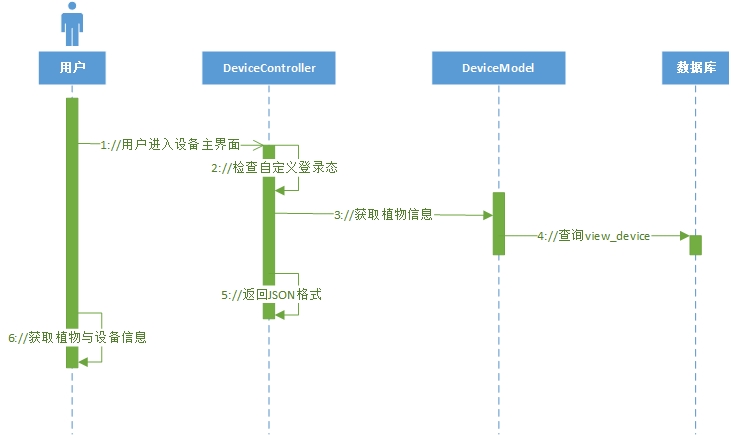
请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/device/list/get

操作数据表：view\_device

1. 流程设计
2. 时序图



1. 获取检测数据
2. 模块描述

暂时还不清楚该如何实现，需和温友建进行沟通后才能确定。

### **3.1.4 设置植物**

1. 具体描述

用户可以通过扫描植物花盆上的二维码来自动设置植物种类。除此之外，用户也可以手动设置植物的种类。

1. 具体实现
2. 获取植物分类
3. 模块描述

用户进入设置植物页面后，小程序会将已有的植物分类都呈现在页面上。用户可以根据已有的分类加快寻找合适的植物种类。该接口的具体实现可参考3.2.5植物检索中的（二）植物分类相关。

1. 获取指定分类植物
2. 模块描述

获取植物分类后，点击不同的植物分类，可以查看指定分类下的植物。该接口的具体实现可参考3.2.5植物检索中的（二）植物分类相关。

1. 检索植物种类

1. 模块描述

用户可以在输入框内输入植物名称，来检索符合输入内容的植物种类。

1. 输入、输出

输入：自定义登录态、检索内容、已获取数据数。

输出：植物号、植物名称。

1. 模块说明

服务器后台收到前端请求后，根据参数“检索内容”和“已获取数据数”进行模糊查询，把符合要求的数据返回给前端。每次暂定返回15条数据。

1. 视图设计

### 植物种类视图 view\_plant\_kind

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | nplant的Pid字段，唯一标志植物信息 | 植物号 | nplant |
| name | nplant的Cname字段，植物名称 | 植物名称 | nplant |

1. 接口设计

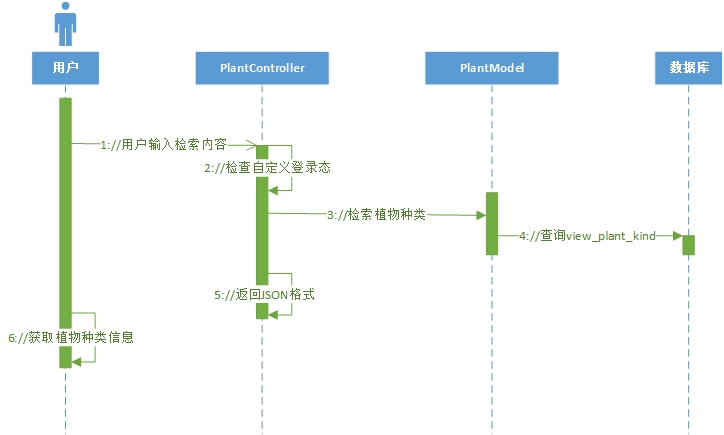
请求方式：GET

请求参数参数：自定义登录态、检索内容、已获取数据数。

请求地址：协议+域名/wjg/plant/list/kind

操作数据表：view\_plant\_kind

1. 流程设计
2. 时序图



1. 设置植物种类

1. 模块描述

用户在设置植物页面找到需要设置的植物种类后，点击植物名称后，在弹出的确定面板中确认即可修改设备的植物种类。

1. 输入、输出

输入：自定义登录态、标识码、植物号。

输出：无。

1. 模块说明

服务器后台收到前端请求后，根据参数“标识码”到ndevice基表中找到相应的数据，并把其中的Ipid字段的值修改为参数“植物号”。

1. 接口设计

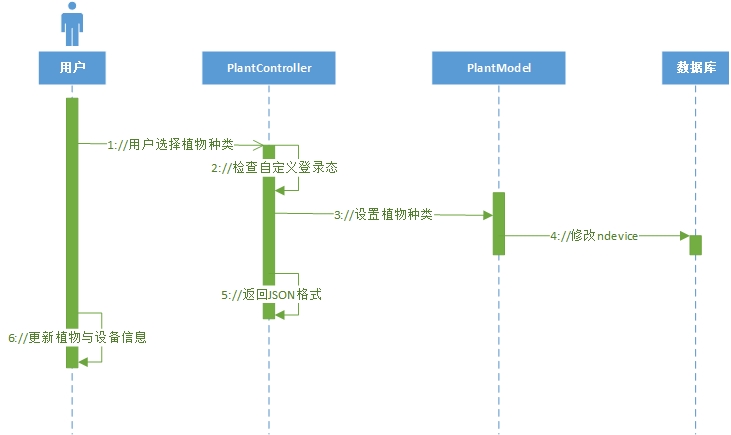
请求方式：PATCH

请求参数参数：自定义登录态、标识码、植物号。

请求地址：协议+域名/wjg/device/detail/modify

操作数据表：ndevice

1. 流程设计
2. 时序图



### **3.1.5 数据检测**

一．具体描述

目前具体的流程还不清楚，需要进行讨论之后方可得出具体的过程。但无论传感器和手机之间的通信过程如何，前后端的交互过程是确定的——前端将检测获得的数据传递给服务器后台。

二．具体实现

一）提交检测数据

1. 模块描述

小程序将检测数据提交到服务器。

1. 输入、输出

输入：自定义登录态、用户账号、检测日期、检测温度、检测水份、检测光照、检测肥力。

输出：无。

1. 模块说明

服务器接收到前端传递的参数后，将参数插入到jrecord基表中。

1. 数据库设计

### 检测数据表 jrecord

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Vdate | Char(19) | 否 | 无 | YYYY-MM-dd HH:mm:ss | 检测日期 |
| Ntemperature | Varchar() | 否 | 无 | 检测所得的温度数据 | 检测温度 |
| Wmoisture | Varchar() | 否 | 无 | 检测所得的水份数据 | 检测水份 |
| Ofertility | Varchar() | 否 | 无 | 检测所得的肥力数据 | 检测肥力 |
| Eillumination | Varchar | 否 | 无 | 检测所得的光照数据 | 检测光照 |
| Haccount | Varchar | 否 | 外键 | 对应用户表的主键 | 用户账号 |

1. 视图设计

### 检测数据视图 view\_record

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| date | nplant的Pid字段，唯一标志植物信息 | 检测日期 | jrecord |
| temperature | jrecord的Ntemperature字段，检测温度 | 检测温度 | jrecord |
| moisture | jrecord的Wmoisture字段，检测水份 | 检测水份 | jrecord |
| fertility | jrecord的Ofertility字段，检测肥力 | 检测肥力 | jrecord |
| illumination | jrecord的Eiilumination字段，检测光照 | 检测光照 | jrecord |
| account | jrecord的Haccount字段，用户账号 | 用户账号 | jrecord |

1. 程序界面

参考3.1.3.1智能设备主界面。

1. 接口设计

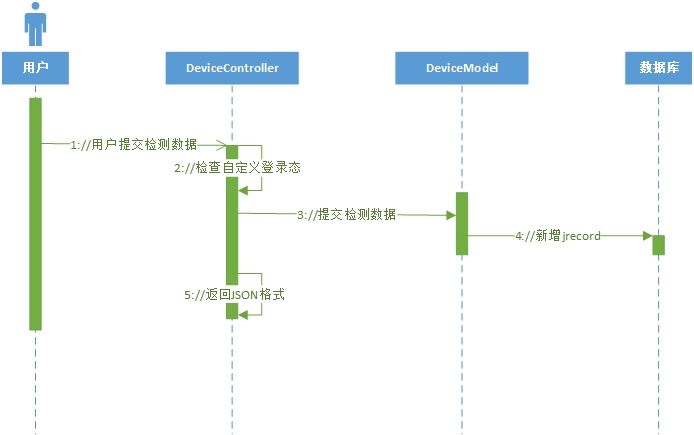
请求方式：POST

请求参数参数：自定义登录态、检测日期、检测温度、检测水份、检测肥力、检测光照、用户账号。

请求地址：协议+域名/wjg/device/record/add

操作数据表：jrecord

1. 流程设计
2. 时序图



### **3.1.6 查看数据报告**

一．具体描述

用户可以主动生成数据报告，来查看今日的数据检测情况。服务器每日（每周）也会主动生成数据报告，为历史记录提供数据依据。

二．具体实现

一）生成数据报告

1. 模块描述

用户点击“数据报告”按钮后，即可进入到“数据报告”页面。在该页面，用户可以查看今日的数据检测情况，包括检测次数、各项检测数据的折线图以及平均数据、总评和养护小贴士。



1. 输入、输出

输入：自定义登录态、用户账号、日期。

输出：检测次数、最近七条检测数据（温度、水份、光照、肥力）、平均温度、平均水份、平均光照、平均肥力、总评、养护建议。

1. 模块说明

服务器收到前端传递的参数后，根据“用户账号”和“日期”从view\_record视图中检索指定用户今日的所有检测数据。然后计算出平均温度、平均水份、平均光照以及平均肥力，再根据计算出的平均数据到view\_summary视图以及view\_advice视图中检索相应的数据。最后整合以上数据并返回给前端。

1. 数据库设计

### 日数据报告表 nreportday

1. 视图设计
2. 程序界面



图3.1.6.1数据报告页面

1. 接口设计

请求方式：GET

请求参数参数：自定义登录态、日期、用户账号。

请求地址：协议+域名/wjg/device/record/get

操作数据表：view\_record、view\_summary、view\_advice

1. 流程设计
2. 时序图

二）定时生成日数据报告

1. 模块描述

服务器每日凌晨启动定时任务，将所有用户昨日的检测数据汇总成一个日数据报告。与“生成数据报告”操作生成的数据报告相似，不过在数据方面只保留了检测次数和各项指标的平均数值。



1. 模块说明

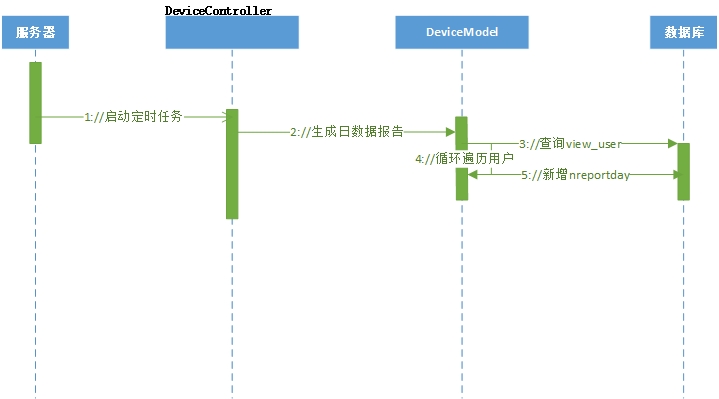
服务器从view\_user视图中获取每个用户的账号，然后依据用户账号循环获取每个用户昨日的所有检测数据，并进行次数统计与平均数值计算。最后将计算结果保存到nreportday基表中。

1. 数据库设计

### 日数据报告表 nreportday

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Hdate | Char(10) | 否 | 无 | 格式：YYYY-MM-dd | 检测日期 |
| Bcount | Int(2) | 否 | 无 | 今日数据检测的次数 | 检测次数 |
| Ctemperature | Varchar() | 否 | 无 | 计算所得的平均温度 | 平均温度 |
| Imoisture | Varchar() | 否 | 无 | 计算所得的平均水份 | 平均水份 |
| Mfertility | Varchar() | 否 | 无 | 计算所得的平均肥力 | 平均肥力 |
| Pillumination | Varchar | 否 | 无 | 计算所得的平均光照 | 平均光照 |
| Kaccount | Varchar | 否 | 外键 | 对应用户表的主键 | 用户账号 |

1. 流程设计
2. 时序图



三）定时生成周数据报告

1. 模块描述

服务器每周一凌晨启动定时任务，将所有用户上一周的日数据报告汇总成一个周数据报告。与“生成数据报告”操作生成的数据报告相似，不过在数据方面只保留了检测次数和各项指标的平均数值。

1. 模块说明

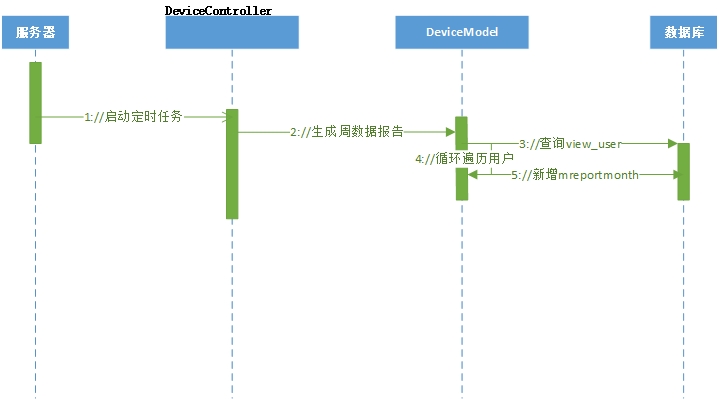
服务器从view\_user视图中获取每个用户的账号，然后依据用户账号循环获取每个用户上一周的所有日数据报告，并进行次数统计与平均数值计算。最后将计算结果保存到mreportmonth基表中。

1. 数据库设计

### 周数据报告表 mreportmonth

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Vdate | Char(10) | 否 | 无 | 格式：YYYY-MM-dd | 起始日期 |
| Ncount | Int(2) | 否 | 无 | 一周数据检测的次数 | 检测次数 |
| Gtemperature | Varchar() | 否 | 无 | 计算所得的平均温度 | 平均温度 |
| Hmoisture | Varchar() | 否 | 无 | 计算所得的平均水份 | 平均水份 |
| Sfertility | Varchar() | 否 | 无 | 计算所得的平均肥力 | 平均肥力 |
| Jillumination | Varchar | 否 | 无 | 计算所得的平均光照 | 平均光照 |
| Qaccount | Varchar | 否 | 外键 | 对应用户表的主键 | 用户账号 |

1. 流程设计
2. 时序图



### **3.1.7 查看历史记录**

一．具体描述

用户可以查看最近一周的数据检测情况，也可以查看最近一月的数据检测情况，包括历史轨迹（折线图）以及每个时间节点的详细报告。

二．具体实现

一）查看周历史轨迹

1. 模块描述

用户点击“历史记录”按钮后，即可进入到“历史记录”页面。在该页面，用户可以查看最近一周的数据检测情况，包括检测次数、各项检测数据的折线图以及平均数据。



1. 输入、输出

输入：自定义登录态、用户账号、日期。

输出：检测次数、检测日期、平均温度、平均水份、平均光照、平均肥力。

1. 模块说明

服务器收到前端传递的参数后，根据“用户账号”和“日期”从view\_report\_day视图中检索指定用户最近七天的检测数据，并返回给前端。

1. 视图设计

### 周历史记录视图 view\_report\_week

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| date | nreportday的Hdate字段，检测日期 | 检测日期 | nreportday |
| count | nreportday的Bcount字段，检测次数 | 检测次数 | nreportday |
| temperature | nreportday的Ctemperature字段 | 检测温度 | nreportday |
| moisture | nreportday的Imoisture字段 | 检测水份 | nreportday |
| fertility | nreportday的Mfertility字段 | 检测肥力 | nreportday |
| illumination | nreportday的Piilumination字段 | 检测光照 | nreportday |
| account | nreportday的Kaccount字段 | 用户账号 | nreportday |

1. 程序界面



图3.1.7.1历史记录页面

1. 接口设计

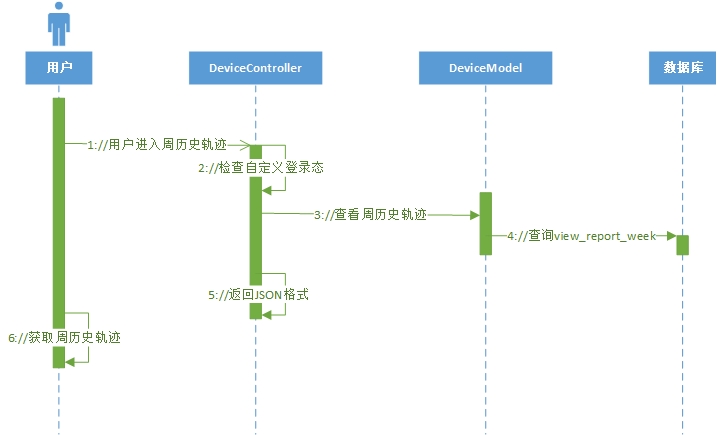
请求方式：GET

请求参数参数：自定义登录态、日期、用户账号。

请求地址：协议+域名/wjg/device/history/week

操作数据表：view\_report\_week

1. 流程设计
2. 时序图



二）查看月历史轨迹

1. 模块描述

用户进入“历史记录”页面后，在页面的上方点击“月历史轨迹”，即可切换到“月历史轨迹”面板。在该面板，用户可以查看最近一月的数据检测情况，包括检测次数、各项检测数据的折线图以及平均数据。



1. 输入、输出

输入：自定义登录态、用户账号、日期。

输出：检测次数、检测日期、平均温度、平均水份、平均光照、平均肥力。

1. 模块说明

相比查看周历史轨迹，查看月历史轨迹要稍微麻烦一些。麻烦的一点主要在于本周的数据报告是不存在的，也就是查看月历史轨迹只能查看从上一周开始的四周记录。这一点问题到时候可以继续讨论。

先不管上述提到的问题，就目前的实现来看：服务器后台接收到前端传递的“用户账号”从view\_report\_month视图中获取最近四条数据，并返回给前端。

1. 视图设计

### 月历史记录视图 view\_report\_month

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| date | mreportmonth的Vdate字段 | 起始日期 | mreportmonth |
| count | mreportmonth的Ncount字段 | 检测次数 | mreportmonth |
| temperature | mreportmonth的Gtemperature字段 | 检测温度 | mreportmonth |
| moisture | mreportmonth的Hmoisture字段 | 检测水份 | mreportmonth |
| fertility | mreportmonth的Sfertility字段 | 检测肥力 | mreportmonth |
| illumination | mreportmonth的Jiilumination字段 | 检测光照 | mreportmonth |
| account | mreportmonth的Qaccount字段 | 用户账号 | mreportmonth |

1. 程序界面

参考3.1.7.1历史记录页面。

1. 接口设计

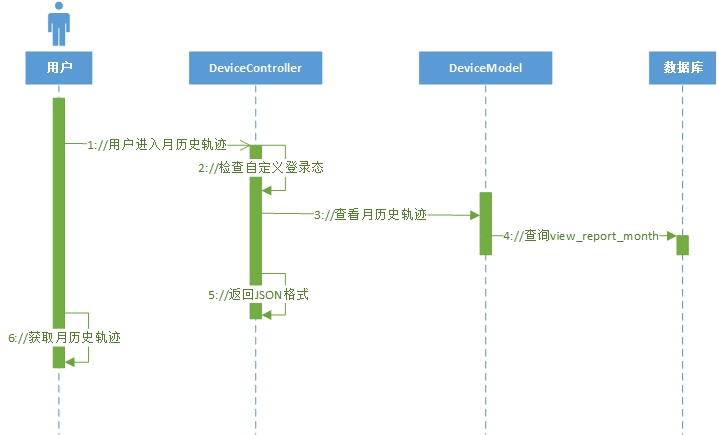
请求方式：GET

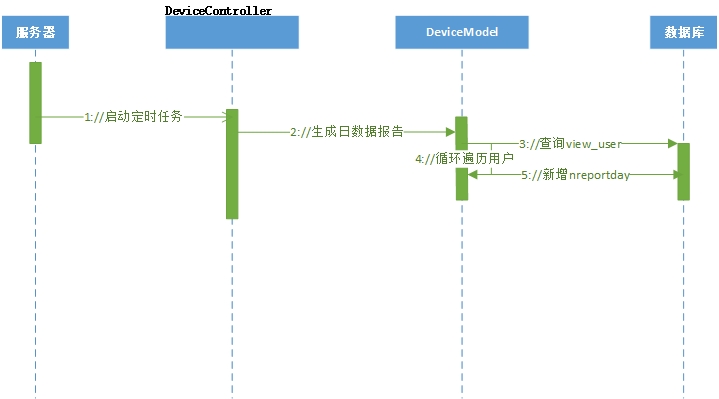
请求参数参数：自定义登录态、日期、用户账号。

请求地址：协议+域名/wjg/device/history/month

操作数据表：view\_report\_month

1. 流程设计
2. 时序图





三）定时生成周数据报告

1. 模块描述

服务器每周一凌晨启动定时任务，将所有用户上一周的日数据报告汇总成一个周数据报告。与“生成数据报告”操作生成的数据报告相似，不过在数据方面只保留了检测次数和各项指标的平均数值。

1. 模块说明

服务器从view\_user视图中获取每个用户的账号，然后依据用户账号循环获取每个用户上一周的所有日数据报告，并进行次数统计与平均数值计算。最后将计算结果保存到mreportmonth基表中。

1. 数据库设计

### 周数据报告表 mreportmonth

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Vdate | Char(10) | 否 | 无 | 格式：YYYY-MM-dd | 起始日期 |
| Ncount | Int(2) | 否 | 无 | 一周数据检测的次数 | 检测次数 |
| Gtemperature | Varchar() | 否 | 无 | 计算所得的平均温度 | 平均温度 |
| Hmoisture | Varchar() | 否 | 无 | 计算所得的平均水份 | 平均水份 |
| Sfertility | Varchar() | 否 | 无 | 计算所得的平均肥力 | 平均肥力 |
| Jillumination | Varchar | 否 | 无 | 计算所得的平均光照 | 平均光照 |
| Qaccount | Varchar | 否 | 外键 | 对应用户表的主键 | 用户账号 |

1. 流程设计
2. 时序图

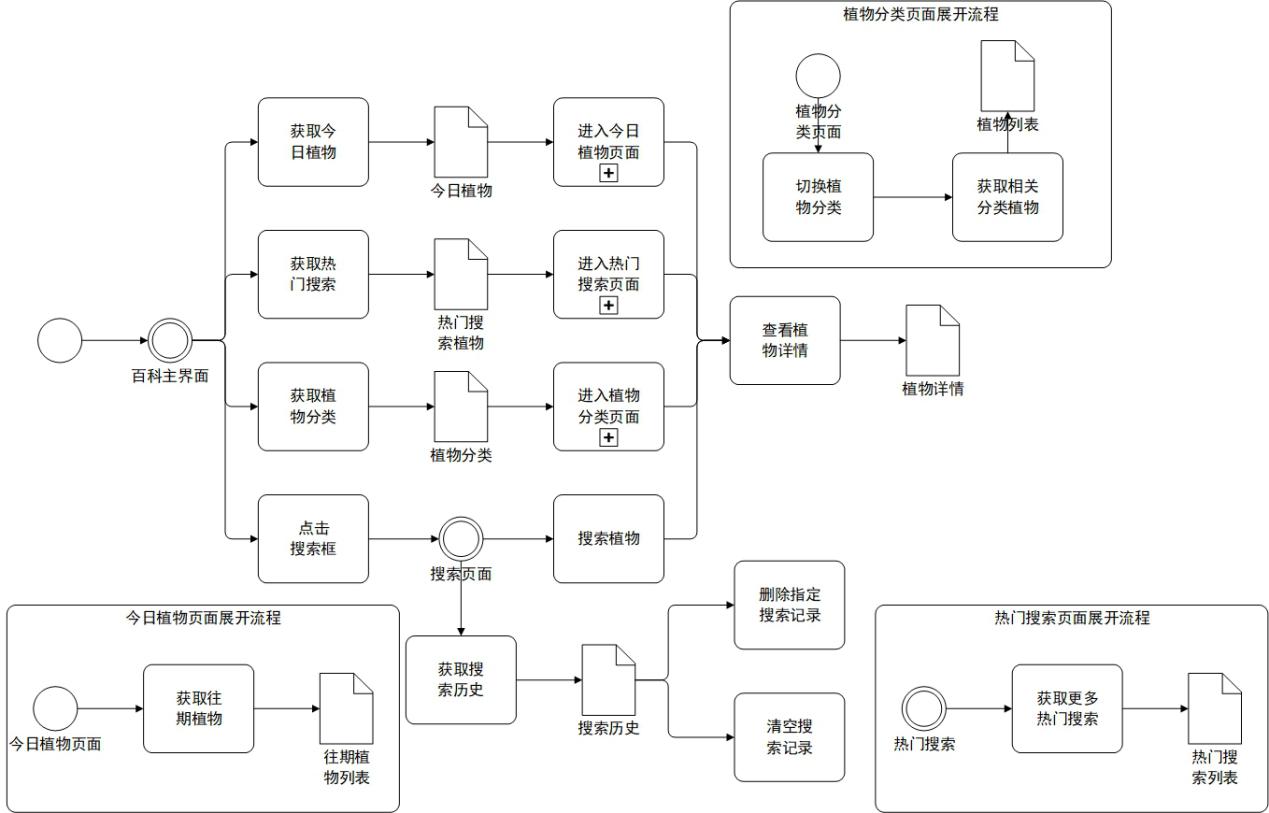
## 百科模块

百科模块相当于一本电子版的植物百科全书，主要用于展示植物相关的百科知识，包括植物的科属、习性以及养护小知识。为了增加百科模块的趣味性，增加了今日植物以及热门搜索模块。

### **需求概述**

1. 今日植物：小程序每天都会为用户推送一类植物，用户也可以查看往期的今日植物。
2. 热门搜索：小程序显示百科中搜索次数最多的植物种类，并呈现在热门搜索页面中。
3. 植物检索：用户可以通过小程序已有的植物分类进行植物检索，也可以通过在搜索框内输入植物名称等方式来获取植物详情。

### 流程图



### **今日植物**

一．具体描述：

微景观小程序每日推荐一类植物，用户可以点击该植物查看该植物的具体数据。也可以查看往期的今日植物。

二．具体流程：

一）定时设置今日植物

1. 模块描述

每日凌晨启动定时任务，自动从植物表中选取一类植物，将其添加到今日植物表中。

1. 模块说明
2. 首先创建基表btoday用以存放今日植物数据，这么做可以极大地方便获取往日植物操作。
3. 基于基表btoday和nplant的数据，建立视图view\_plant\_today\_select。该视图用以显示nplant除去btoday的植物数据，我们自然是挑选没有被选择的植物，因此可以通过 Select \* from nplant where id not in (select id from today)这种方式来实现。
4. 服务器后台需要做的就是每日凌晨启动定时任务，采用某种算法从view\_plant\_today\_select视图中任意挑选一类植物，将id添加到btoday中的Dpid字段即可，Udate字段获取当前系统时间。
5. 数据库设计

### 今日植物表 btoday

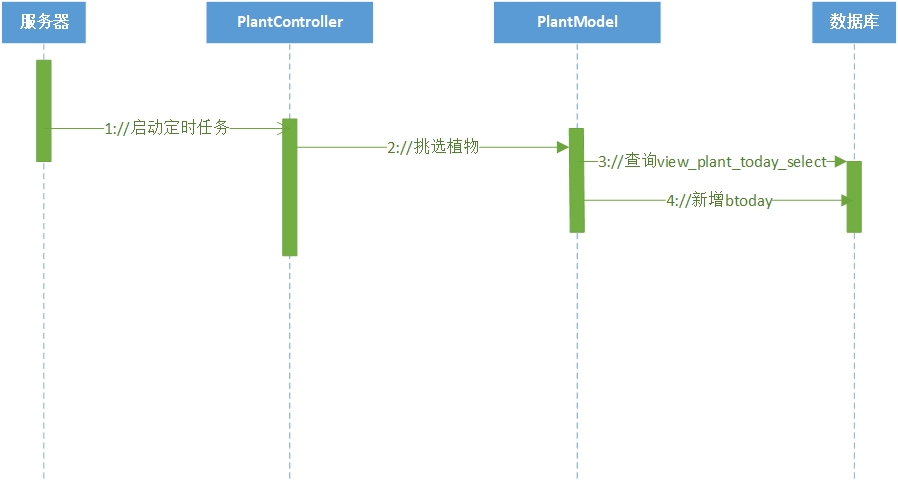
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Tid | Int(6) | 否 | 主键 | 唯一标识今日植物，自增长 | 今日植物号 |
| Dpid | Int(6) | 否 | 外键 | 对应植物表的主键 | 植物号 |
| Udate | Char(16) | 否 | 无 | 格式：YYYY-MM-dd HH:mm | 日期 |

1. 视图设计

### 今日植物挑选视图 view\_plant\_today\_select

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | nplant的Pid字段，唯一标志植物信息 | 植物号 | nplant |

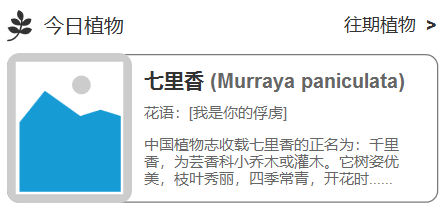
1. 流程设计
2. 时序图



二）获取今日植物

1. 模块描述

微景观小程序每日推荐一颗植物。用户也可以点击“往期植物”进入到“植物历”页面，在该页面可以查看往日推荐的所有植物。



2. 输入、输出

输入：自定义登录态、已获取数据的最晚日期、数据条目。

输出：植物号、植物图片、植物名称、植物学名、【花语】、生物学特性。

3. 模块说明

服务器后台从view\_plant\_today视图中获取最新的一条数据，并返回给前端。该接口同“（三）获取往期植物”，区别在于该接口前端传递的数据条目为1，且最晚日期为2099年，因此后台有且只有一条最新的数据。

1. 视图设计

### 今日植物视图 view\_plant\_today

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | nplant的Pid字段，唯一标志植物信息 | 植物号 | nplant |
| name | nplant的Cname字段，植物名称 | 植物名称 | nplant |
| sname | nplant的Ksname字段，植物学名 | 植物学名 | nplant |
| characteristics | nplant的Hcharacteristics字段 | 生物学特性 | nplant |
| picture | nplant的Ypicture字段，植物图片 | 植物图片 | nplant |
| flang | nplant的Rflang字段，花语 | 花语 | nplant |
| date | btoday的Udate字段，日期 | 日期 | btoday |

1. 程序界面



图3.2.3.1今日植物板块

6. 接口设计

请求方式：GET

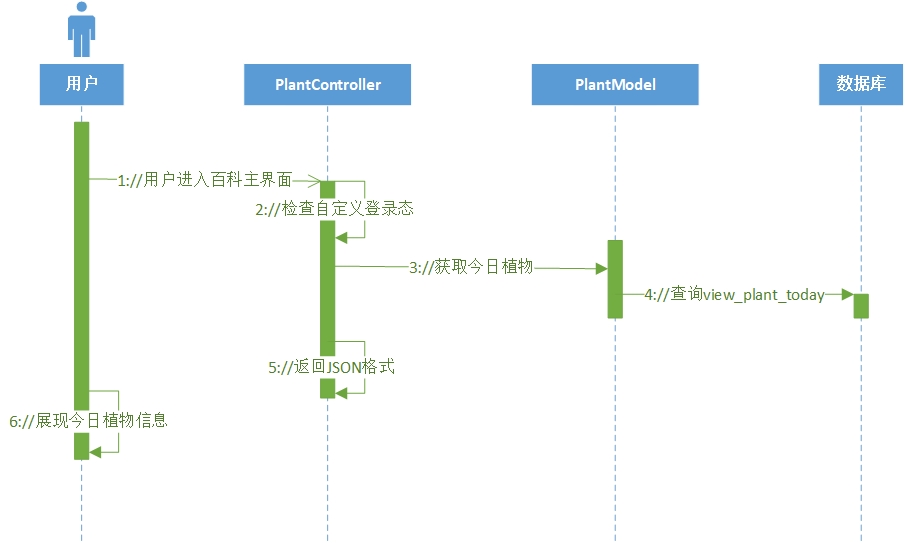
请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/plant/list/today

操作数据表：view\_plant\_today

7. 流程设计

A. 时序图



三）获取往期植物

1. 模块描述

用户点击“往期植物”，即可跳转到今日植物页面。在该页面，用户可以查看过去的今日植物数据。

1. 输入、输出

输入：自定义登录态、已获取数据的最晚日期、数据条目。

输出：植物号、植物图片、植物名称、植物学名、【花语】、生物学特性。

1. 模块说明

服务器后台从view\_plant\_today视图中检索数据，每次检索“数据条目”数量的数据返回给前端。前端会传递已获取到数据的最晚日期，服务器依据该参数来挑选合适的数据进行返回。通过“已获取数据的最晚日期”和“数据条目”这两个参数即可实现下滑加载更多功能。

1. 程序界面



图3.2.3.2今日植物页面

1. 接口设计

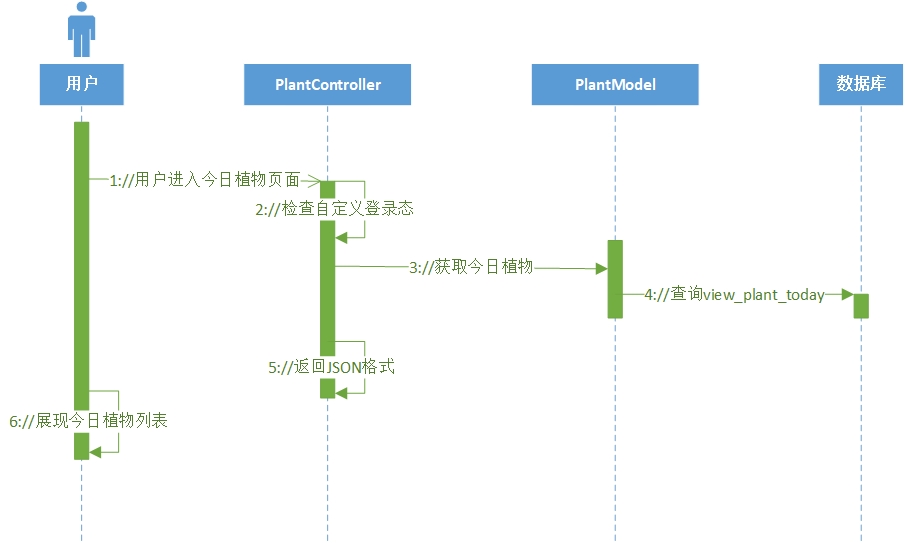
请求方式：GET

请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/plant/list/today

操作数据表：view\_plant\_today

1. 流程设计
2. 时序图



### **3.2.4 热门搜索**

一．具体描述

小程序会将用户每次的搜索都记录下来，搜索次数最多的植物会成为热门搜索植物。用户可以直接在百科主界面看到热门搜索中的几类植物。

二．具体实现

一）获取热门搜索

1. 模块描述

小程序随机展示当前热门的几类植物。

1. 输入、输出

输入：自定义登录态。

输出：植物号、植物图片、植物名称。

1. 模块说明

服务器后台从view\_plant\_popular视图中随机获取三条数据（暂且设置为三条，具体数量需等页面完成后视效果而定），并返回给前端。

1. 数据库设计

### 热门搜索记录表 jpopularsearch

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Psid | Char(32) | 否 | 主键 | 唯一标识热门搜索信息，UUID | 热门搜索号 |
| Epid | Int(6) | 否 | 外键 | 对应植物表的主键 | 植物号 |
| Ycount | Int(6) | 否 | 无 | 当前植物的搜索次数 | 搜索次数 |

1. 视图设计

### 热门搜索视图 view\_plant\_popular

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | nplant的Pid字段，唯一标志植物信息 | 植物号 | nplant |
| name | nplant的Cname字段，植物名称 | 植物名称 | nplant |
| picture | nplant的Ypicture字段，植物图片 | 植物图片 | nplant |

1. 程序界面



图3.2.4.1热门搜索板块

1. 接口设计

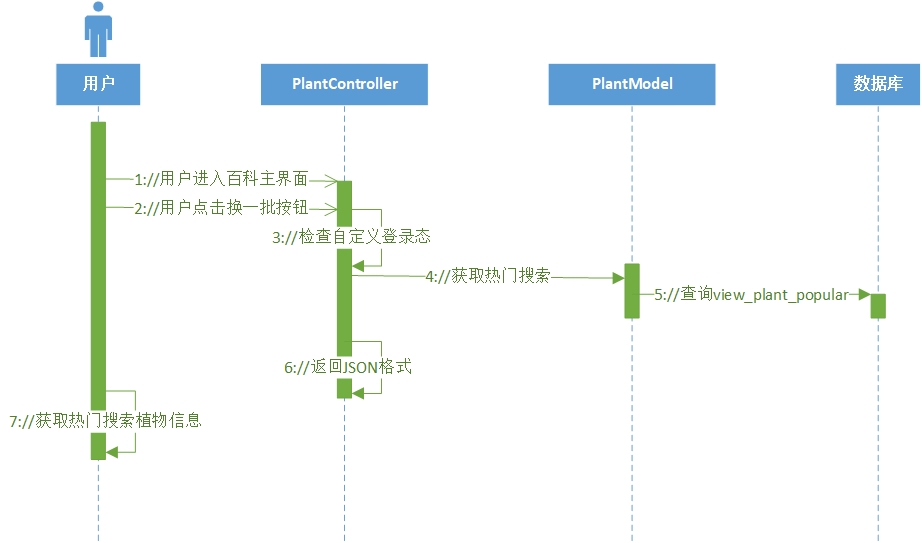
请求方式：GET

请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/plant/popular/get

操作数据表：view\_plant\_popular

1. 流程设计
2. 时序图



二）获取热门搜索列表

1. 模块描述

用户进入热门搜索页面后，在页面上呈现热门top20的植物。

2. 输入、输出

输入：自定义登录态。

输出：植物号、植物名称、植物图片。

1. 模块说明

服务器后台收到前端请求后，将view\_plant\_popular视图中的数据全部返回给前端。

1. 程序界面

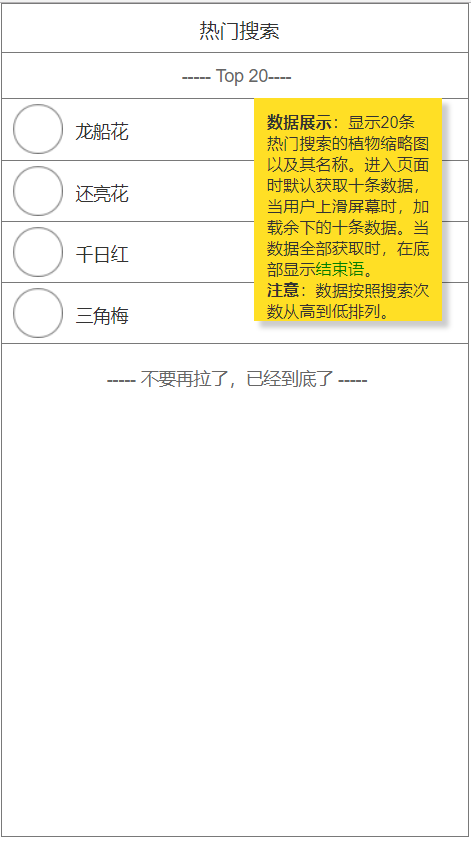


图3.2.4.2热门搜索页面

1. 接口设计

请求方式：GET

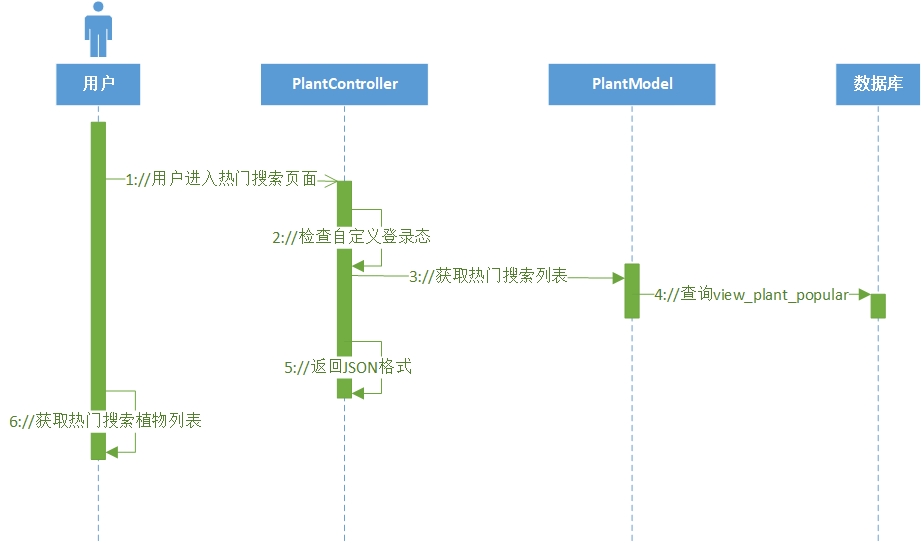
请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/plant/popular/all

操作数据表：view\_plant\_popular

1. 流程设计

A. 时序图



### **3.2.5 植物检索**

一．具体描述

小程序会将用户每次的搜索都记录下来，搜索次数最多的植物会成为热门搜索植物。用户可以直接在百科主界面看到热门搜索中的几类植物。

二．具体实现

一）植物搜索相关

1. 模块描述

用户可以在搜索框内输入植物名称来检索想要检索的植物。每次检索的内容都会被记录下来，用户通过搜索历史来快速检索之前检索的内容。除此之外，用户可以主动清空搜索历史。

1. 输入、输出

（1）获取搜索历史

输入：自定义登录态、用户账号。

输出：植物历史号、搜索内容。

（2）删除指定搜索记录

输入：自定义登录态、植物历史号。

输出：植物历史号、搜索内容。

（3）清空搜索历史

输入：自定义登录态、用户账号。

输出：无。

（4）检索植物

输入：自定义登录态、检索内容、用户账号。

输出：植物号、植物图片、植物名称、植物学名、【花语】、生物学特性。

1. 模块说明
2. 获取搜索历史：当用户点击百科主界面的搜索框时，小程序会自动跳转到植物搜索界面（见图3.5.2.1）。在页面初始化阶段就向后台发起请求以获取当前用户的搜索历史。后台返回当前用户最近的十条搜索记录（若不足十条，则返回全部）。
3. 删除指定搜索记录：用户点击指定搜索记录后方的叉号，即代表删除该搜索记录。此时请求会将该搜索记录的搜索历史号发送到后台，后台根据搜索历史号到数据库中将指定数据删除，并再次返回最近的十条搜索记录（若不足十条，则返回全部）。
4. 清空搜索记录：当用户点击“清空搜索记录”文字按钮之后，后台即把前端传递的用户账号下的所有搜索记录全部删除（delete from rhistoryplant where Raccount = XXX）。
5. 检索植物：服务器后台接收前端传递的参数后，根据检索内容到 view\_plant 视图中检索符合检索内容的植物数据，并将这些植物数据返回给前端。除此之外，还需要执行两项操作。其一，将本次检索内容保存到植物搜索记录表rhistoryplant中。其二，依次修改搜索结果中植物的搜索次数。例如搜索结果返回植物号为1和2的植物，那么服务器后台需要到jpopularsearch热门搜索记录表中去修改植物号为1和2植物的搜索次数，使其分别加一。
6. 数据库设计

### 植物搜索记录表 rhistoryplant

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Hpid | Char(32) | 否 | 主键 | 唯一标识植物搜索信息，UUID | 植物搜索号 |
| Wcontent | Varchar(10) | 否 | 无 | 用户的搜索记录 | 搜索内容 |
| Uaccount | Char(20) | 否 | 外键 | 对应用户表的主键 | 所属用户 |

1. 程序界面



图3.2.5.1植物搜索页面

1. 接口设计
2. 获取搜索历史

请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/plant/history/get

操作数据表：view\_plant\_history

1. 删除指定搜索记录

请求方式：DELETE

请求参数参数：自定义登录态、植物历史号。

请求地址：协议+域名/wjg/plant/history/delete

操作数据表：rhistoryplant

1. 清空搜索记录

请求方式：DELETE

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/plant/history/deleteAll

操作数据表：rhistoryplant

1. 检索植物

请求方式：GET

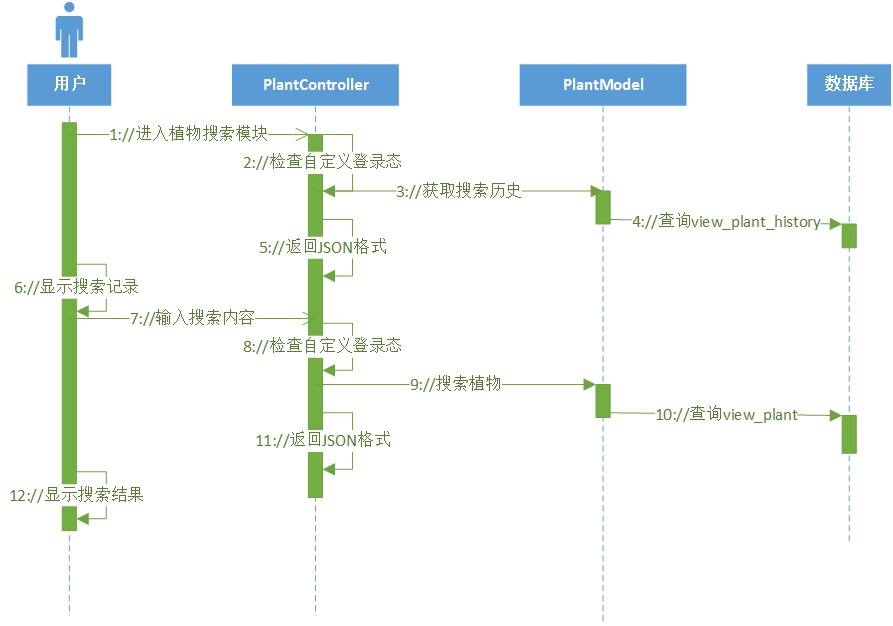
请求参数参数：自定义登录态、检索内容、用户账号。

请求地址：协议+域名/wjg/plant/list/search

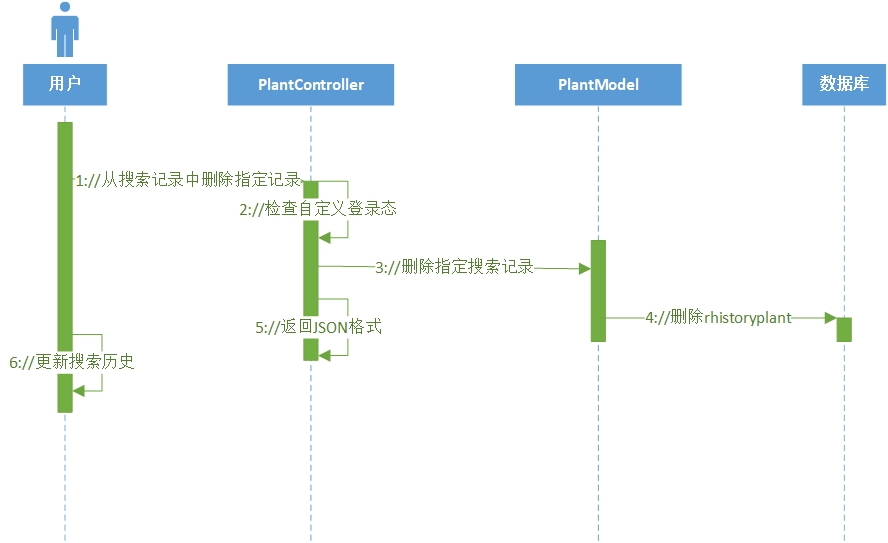
操作数据表：rhistoryplant、view\_plant

1. 流程设计
2. 时序图

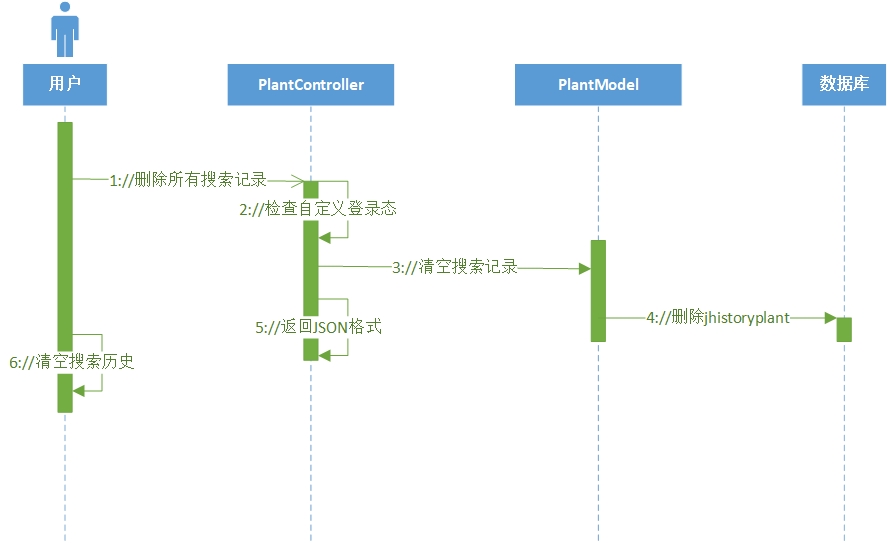
（1）获取搜索历史与检索植物



1. 从搜索记录中删除指定记录



1. 清空搜索记录



二）植物分类相关

1. 模块描述

用户进入百科主界面后，可以在该页面的最下方看到植物的分类，以及每个分两类下的所属植物。

1. 输入、输出

（1）获取植物分类

输入：自定义登录态。

输出：植物分类号、植物分类名。

1. 获取指定分类植物

输入：自定义登录态、植物分类号、已获取植物数

输出：植物号、植物图片、植物名称。

1. 模块说明
2. 获取植物分类：服务器后台收到前端请求后，从view\_plant\_type视图中返回所有的植物分类数据。
3. 获取指定分类植物：服务器后台收到前端请求后，根据参数“植物分类号”以及“已获取植物数”从view\_plant视图中获取指定数量（暂定位15条数据）的植物数据，并返回给前端。
4. 数据库设计

### 植物分类表 kplanttype

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Ptid | Int(2) | 否 | 主键 | 唯一标识植物分类信息，自增长 | 植物分类号 |
| Jname | Varchar(10) | 否 | 唯一 | 植物分类的具体名称 | 植物分类名 |
| Tbrief | Varchar(128) | 否 | 无 | 植物分类的简介 | 植物分类简介 |

1. 视图设计

### 植物分类视图 view\_plant\_type

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | kplanttype的Ptid字段，唯一标志植物分类信息 | 植物分类号 | kplanttype |
| name | kplanttype的Jname字段，植物分类名称 | 植物分类名 | kplanttype |
| brief | kplanttype的Tbrief字段，植物分类简介 | 植物分类简介 | kplanttype |

1. 程序界面

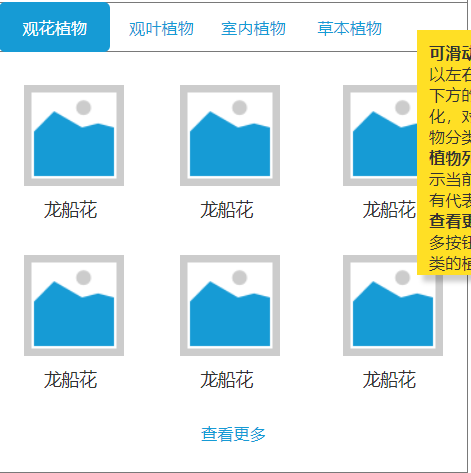


图3.2.5.2植物分类板块



图3.2.5.植物分类页面

1. 接口设计

（1）获取植物分类

请求方式：GET

请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/plant/type/get

操作数据表：view\_plant\_type

（2）获取指定分类植物

请求方式：GET

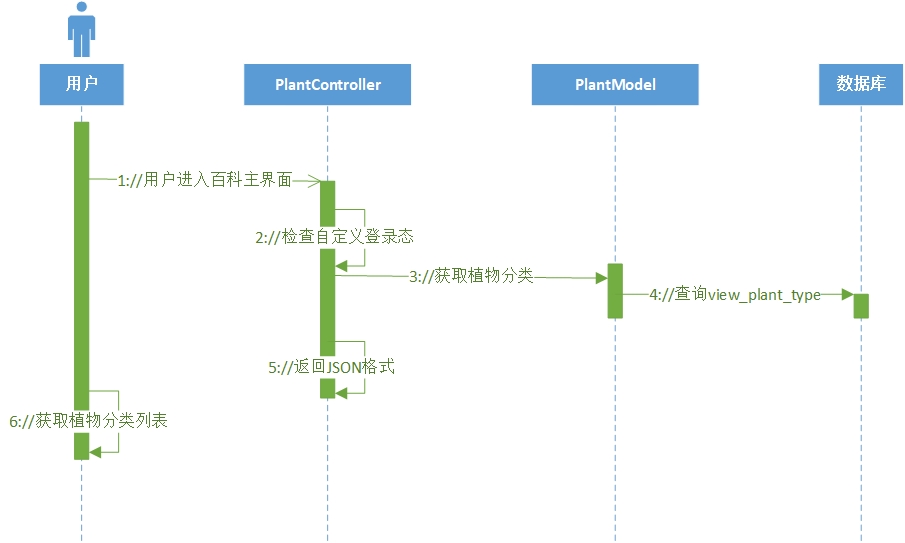
请求参数参数：自定义登录态、植物分类号、已获取植物数。

请求地址：协议+域名/wjg/plant/list/type

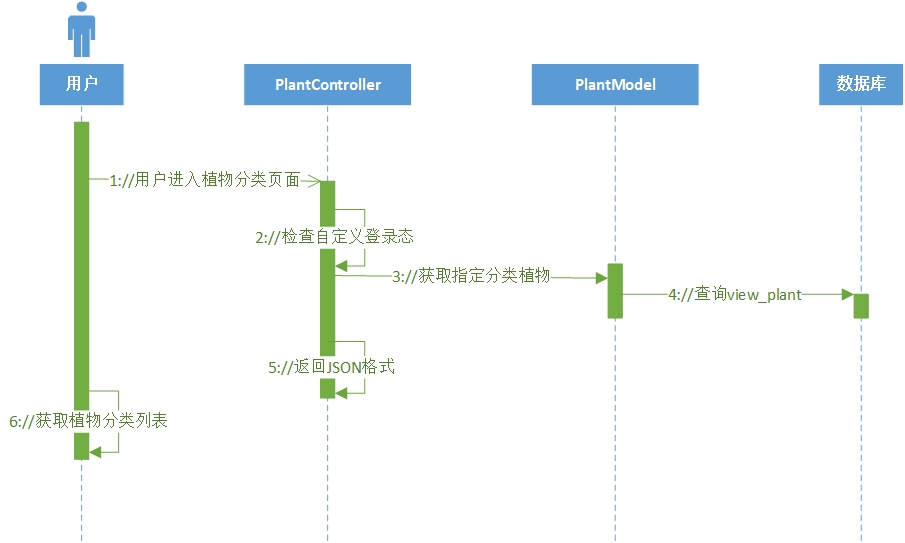
操作数据表：view\_plant

1. 流程设计
2. 时序图

（1）获取植物分类



（2）获取指定分类植物



1. 查看植物详情

1. 模块描述

用户点击植物即可跳转到植物详情页面。在该页面，用户可以查看当前植物的详细信息，包括植物名称、学名、别名、生物学特性以及养护建议等相关内容。今后也可以相应地增加数据。

1. 输入、输出

输入：自定义登录态、植物号。

输出：植物号、植物图片、植物名称、植物学名、植物别名、植物科属、产地与分布、生物学特性、养护建议、【花语】、适宜温度、适宜水份、适宜肥力、适宜光照。

1. 模块说明

服务器后台根据前端传递的植物号，从view\_plant视图中检索相应的植物，并将其信息返回给前端。

1. 程序界面



图3.2.5.4植物详情页面

1. 接口设计

请求方式：GET

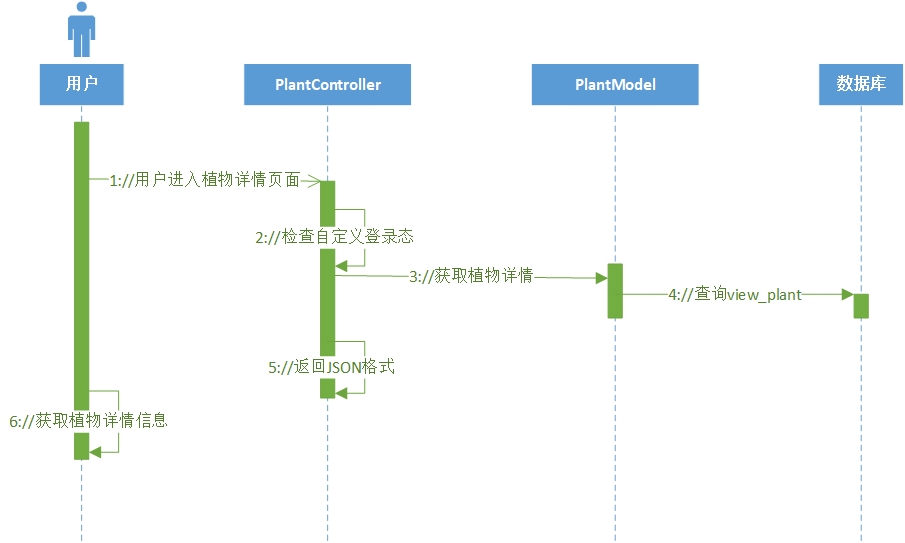
请求参数参数：自定义登录态、植物号。

请求地址：协议+域名/wjg/plant/detail/get

操作数据表：view\_plant

1. 流程设计

A. 时序图



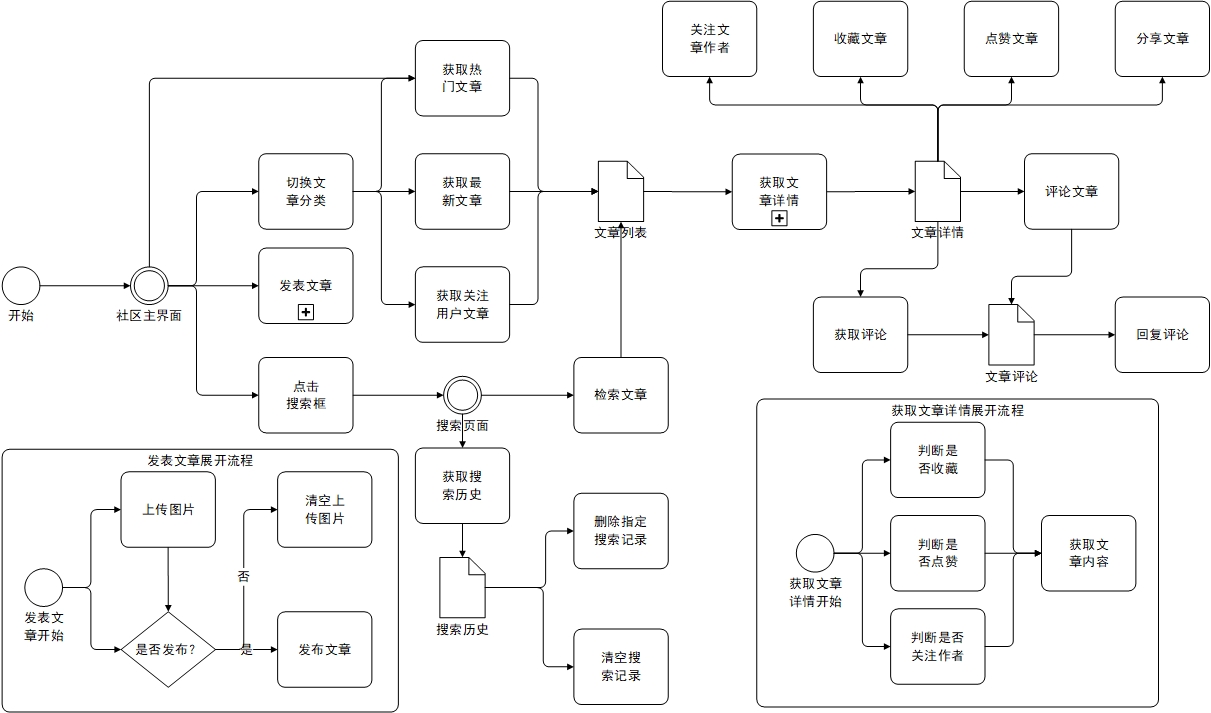
## 社区模块

用户可以在社区中浏览其他用户的文章，并对文章进行评论或回复他人的评论，对感兴趣的文章可以点赞、收藏以及分享，感兴趣的用户可以关注。用户也可以主动发表文章与他人交流，以及检索感兴趣的文章。

### **需求概述**

1. 浏览文章：用户可以浏览其他用户发表的文章。
2. 社区操作：用户可以对喜欢的文章进行收藏、分享以及点赞操作，也可以关注文章的作者。
3. 评论与回复：用户可以评论文章，以及回复文章已有的评论，或者回复他人对自己评论的回复。
4. 发表文章：用户可以主动发布文章，与其他用户进行交流。

### **流程图**



### 浏览文章

一．具体描述：

用户可以通过多种途径浏览文章信息，例如在社区模块中查看最新以及热门的文章，或者通过检索文章标题来查找感兴趣的文章，亦或是关注喜欢的用户，阅读关注用户发表的文章。

二．具体流程：

一）获取最新文章

1. 模块描述

获取最新发表的文章，并以卡片的形式呈现在页面上。

1. 输入、输出

输入：自定义登录态、已获取文章的最晚日期。

输出：文章号、文章标题、内容、日期、图片地址、阅读数、点赞数、用户账号、昵称、头像、话题号、话题名称。

1. 模块说明

服务器后台收到前端请求，先验证自定义登录态是否合理。若自定义登录态验证通过，则根据前端传递的“已获取文章的最晚日期”筛选合适的文章。

已获取文章的最晚日期：view\_article\_new 视图按照字段 date（文章日期）降序排列，假设初始加载前十条数据，第二次加载之后的十条数据。如果此时有其他用户发表了新的文章，那么就会导致view\_article\_new 的数据发生变动，因此仅仅采用获取指定数量的方式就有问题（select \* from view\_article\_new limit n,10）。此时需要结合日期进行判断（select \* from view\_article\_new where date < 已获取文章的最晚日期 limit 10），获取已获取文章的后续文章。

1. 数据库设计

### 文章表 warticle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Aid | Char(32) | 否 | 主键 | 唯一标识文章 | 文章号 |
| Maccount | Char(20) | 否 | 外键 | 用户表外键 | 用户账号 |
| Itopic | Int(2) | 否 |  | 话题表外键 | 话题号 |
| Htitle | Varchar(20) | 否 |  | 文章的标题，最多20个中文 | 文章标题 |
| Dontent | varchar(4096) | 否 |  | 文章的内容，最多4096个中文 | 文章内容 |
| Kdate | Char(16) | 否 |  | 格式：YYYY-MM-dd HH:mm | 文章日期 |
| Opicture | Varchar(128) | 是 |  | 文章所包含图片的路径 | 图片地址 |
| Jread | Int(6) | 是 | 默认 | 默认为0，当前文章的阅读次数 | 阅读数 |
| Gcollect | Int(5) | 是 | 默认 | 默认为0，当前文章的收藏次数 | 收藏数 |
| Ycomment | Int(5) | 是 | 默认 | 默认为0，当前文章的评论次数 | 评论数 |
| Elike | Int(5) | 是 | 默认 | 默认为0，当前文章的点赞次数 | 点赞数 |

注意：对文章的评论以及对评论的回复都计算在“评论数”中。

1. 程序界面



（图：3.3.3.1 社区主界面）

1. 接口设计

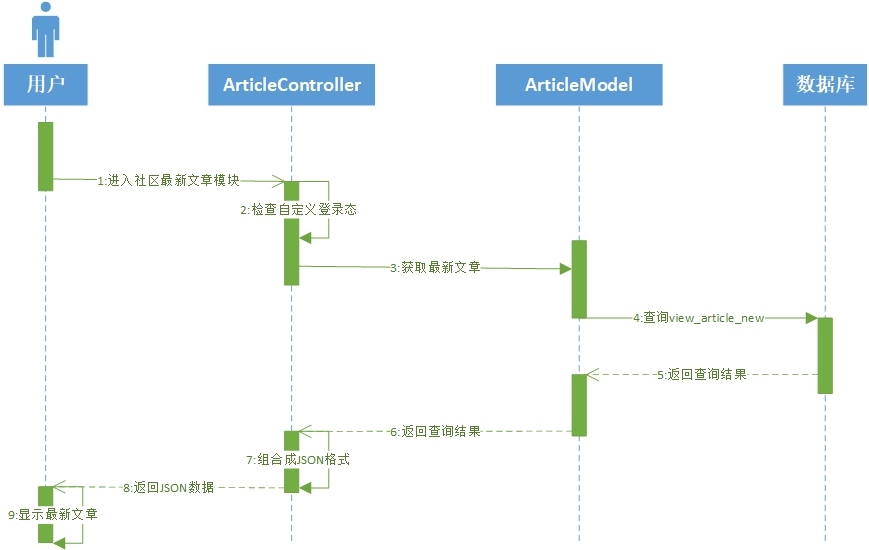
请求方式：GET

请求参数参数：自定义登录态、已获取文章的最晚日期。

请求地址：协议+域名/wjg/article/list/new

操作数据表：view\_article\_new

1. 流程设计
2. 时序图



二）获取热门文章

1. 模块描述

获取最新发表的文章，并以卡片的形式呈现在页面上。进入到社区主界面时，默认获取热门文章。

1. 输入、输出

输入：自定义登录态、已获取文章数目。

输出：文章号、文章标题、内容、日期、图片地址、阅读数、点赞数、用户账号、昵称、头像、话题号、话题名称。

1. 模块说明

服务器后台收到前端请求，先验证自定义登录态是否合理。若自定义登录态验证通过，则根据前端传递的“已获取文章数目”筛选合适的文章（select \* from view\_article\_popular limit n,10）。

与获取最新文章不同，当数据稳定下来之后，热门文章的变动不会很明显，因此可以直接通过 limit n,10 的方式来分批次加载更多文章。

1. 程序界面

参考图3.3.3.1社区主界面。

1. 接口设计

请求方式：GET

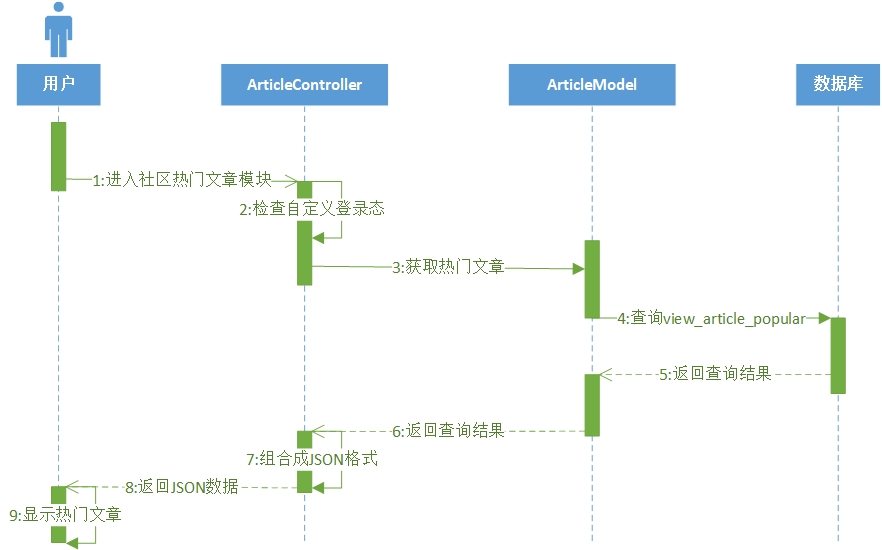
请求参数参数：自定义登录态、已获取文章数目。

请求地址：协议+域名/wjg/article/list/popular

操作数据表：view\_article\_popular

1. 流程设计

A 时序图



三）获取关注用户文章

1. 模块描述

用户可以查看已关注用户的文章。文章按照关注用户的发表时间降序排列。当取消关注指定用户后，该用户的文章就不会出现在关注文章中。

1. 输入、输出

输入：自定义登录态、用户账号、已获取文章数。

输出：文章号、文章标题、内容、日期、图片地址、阅读数、点赞数、用户账号、昵称、头像、话题号、话题名称。

1. 模块说明
2. 程序界面

参考图3.3.3.1社区主界面。

1. 接口设计
2. 流程设计

A 时序图

四）检索文章

1. 模块描述

用户可以在搜索框内输入文章标题来检索想要浏览的文章。每次检索的内容都会被记录下来，用户通过搜索历史来快速检索之前检索的内容。除此之外，用户可以主动清空搜索历史。

1. 输入、输出

（1）获取搜索历史

输入：自定义登录态、用户账号。

输出：搜索历史号、搜索内容。

（2）删除指定搜索记录

输入：自定义登录态、搜索历史号。

输出：搜索历史号、搜索内容。

（3）清空搜索历史

输入：自定义登录态、用户账号。

输出：无。

（4）检索文章

输入：自定义登录态、检索内容、用户账号。

输出：文章号、文章标题、内容、日期、图片地址、阅读数、点赞数、用户账号、昵称、头像、话题号、话题名称。

1. 模块说明
2. 获取搜索历史：当用户点击社区主界面的搜索框时，小程序会自动跳转到文章搜索界面（见图3.4.2.2）。在页面初始化阶段就向后台发起请求以获取当前用户的搜索历史。后台返回当前用户最近的十条搜索记录（若不足十条，则返回全部）。
3. 删除指定搜索记录：用户点击指定搜索记录后方的叉号，即代表删除该搜索记录。此时请求会将该搜索记录的搜索历史号发送到后台，后台根据搜索历史号到数据库中将指定数据删除，并再次返回最近的十条搜索记录（若不足十条，则返回全部）。
4. 清空搜索记录：当用户点击“清空搜索记录”文字按钮之后，后台即把前端传递的用户账号下的所有搜索记录全部删除（delete from chistoryarticle where Raccount = XXX）。
5. 检索文章：服务器后台接收前端传递的参数后，根据检索内容到 view\_article 视图中检索符合检索内容的文章数据，并将这些文章数据返回给前端。除此之外，还需要将本次检索内容保存到文章搜索历史表 chistoryarticle中。
6. 数据库设计

### 文章搜索历史表 chistoryarticle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Thaid | Char(32) | 否 | 主键 | 唯一标识文章搜索历史 | 搜索历史号 |
| Ycontent | Varchar(20) | 否 | 无 | 最多不超过 20 个中文 | 搜索内容 |
| Raccount | Char(20) | 否 | 外键 | 用户表外键 | 所属账号 |

1. 程序界面



图3.3.3.2文章搜索页面

1. 接口设计
2. 获取搜索历史

请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/article/history/get

操作数据表：view\_article\_history

1. 删除指定搜索记录

请求方式：DELETE

请求参数参数：自定义登录态、搜索历史号。

请求地址：协议+域名/wjg/article/history/delete

操作数据表：chistoryarticle

1. 清空搜索记录

请求方式：DELETE

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/article/history/deleteAll

操作数据表：chistoryarticle

1. 检索文章

请求方式：GET

请求参数参数：自定义登录态、检索内容、用户账号。

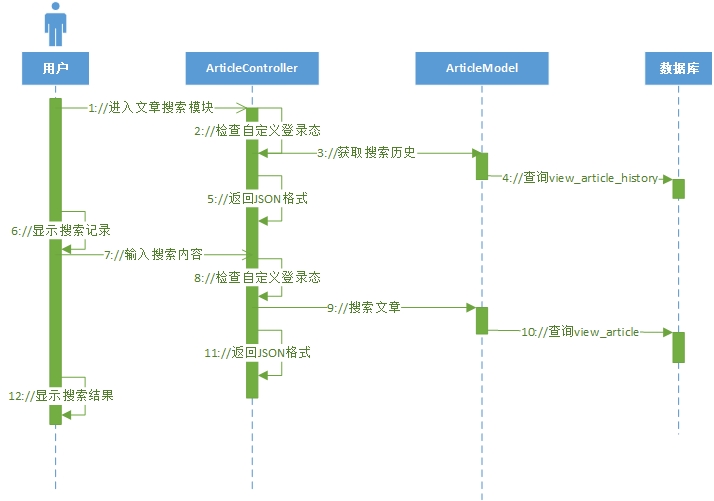
请求地址：协议+域名/wjg/article/list/search

操作数据表：chistoryarticle、view\_article

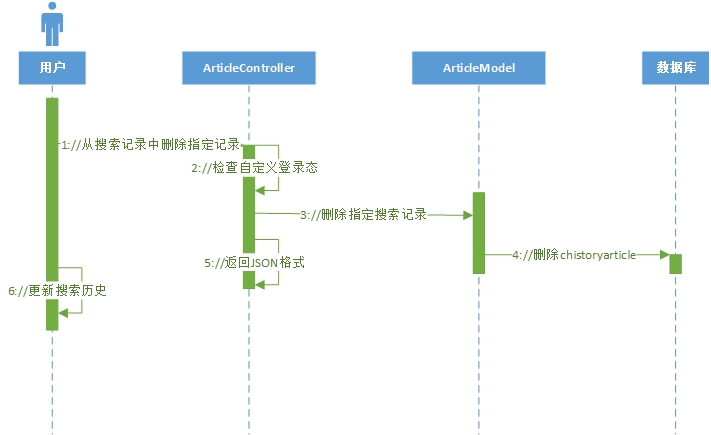
1. 流程设计

A 时序图：

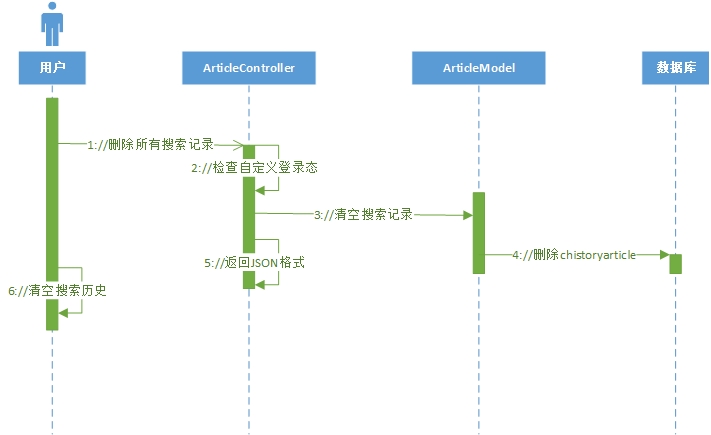
（1）获取搜索历史与检索文章



（2）从搜索记录中删除指定记录。



（3）删除所有搜索记录。



五）获取文章详情

1. 模块描述

用户点击文章卡片，即进入到相应的文章页面。在该页面能够查看文章的全部内容，包括文章作者信息、文章的标题、内容以及配图。下滑可以看到文章的评论。

1. 输入、输出

输入：自定义登录态、文章号、用户账号。

输出：文章号、文章标题、内容、日期、图片地址、阅读数、点赞数、用户账号、昵称、头像、话题号、话题名称、是否收藏、是否关注、是否点赞。

1. 模块说明

服务器后台收到前端传递的文章号以及用户账号后，先判断该文章号对应的文章是否是当前用户的文章。若是用户自己的文章，则不显示“关注”按钮。若不是用户的文章，且没有被关注，则显示“关注”按钮。再判断是否已经被当前用户收藏、点赞，判断完成后，再将这些信息同文章信息一并组合后，返回给前端。最后将该文章的阅读数加一。

1. 程序界面

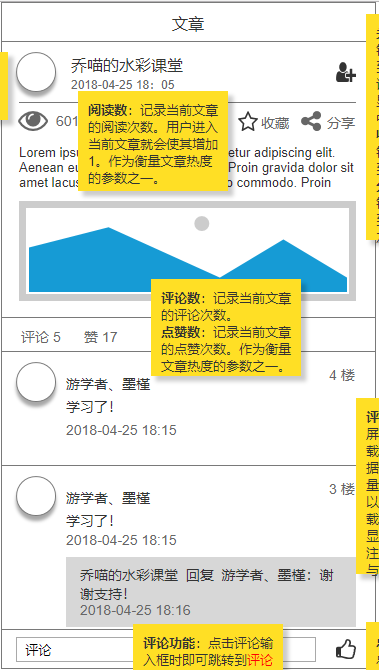


图3.3.3.2文章详情页面

1. 接口设计

请求方式：GET

请求参数参数：自定义登录态、文章号、用户账号。

请求地址：协议+域名/wjg/article/detail/get

操作数据表：view\_article、warticle

1. 流程设计
2. 时序图

### **3.3.4 社区操作**

一．具体描述

用户点击文章卡片后，即跳转到相应的文章详情页面。在该页面，用户可以对喜欢的文章进行收藏、分享以及点赞操作，也可以关注文章的作者。

二．具体实现

一）关注文章作者

1. 模块描述

用户点击文章作者右侧的“添加用户”图标（或在用户主页以及我的关注列表页面中执行关注/取消关注操作）即可将当前文章的作者添加到关注列表中。再次点击该图标，则表示取消关注，将关注的文章作者从关注列表中移除。





2. 输入、输出

（1）关注用户

输入：自定义登录态、文章号、文章作者、用户账号、关注日期、消息类型。

输出：关注号。

（2）取消关注

输入：自定义登录态、关注号、文章作者、用户账号。

输出：无。

3. 模块说明

1. 关注用户：服务器后台接收到前端传递的参数，将文章作者和用户账号插入到 ymyfocus 基表中，Fid则通过UUID生成。然后修改用户的关注数，使其加一，文章作者的粉丝数加一。最后给文章作者发送消息，告知文章作者当前用户关注了他。
2. 取消关注用户：服务器后台接收到前端传递的关注号，并根据关注号从 ymyfocus 基表中删除指定关注号数据。然后修改用户的关注数，使其减一，文章作者的粉丝数减一。

4. 数据库设计

### 关注表 ymyfocus

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Fid | Char(32) | 否 | 主键 | 唯一标识关注信息 | 关注号 |
| Gaccount | Char(20) | 否 | 外键 | 关注的用户（被关注者） | 关注用户 |
| Yuser | Char(20) | 否 | 外键 | 关注者，对应用户表主键 | 当前用户 |

5. 程序界面

参考图3.3.3.2文章详情页面

6. 接口设计

（1）关注用户

请求方式：POST

请求参数参数：自定义登录态、文章号、文章作者、用户账号、关注日期、消息类型。

请求地址：协议+域名/wjg/user/relation/add

操作数据表：ymyfocus、iuser、tmymessage

（2）取消关注

请求方式：DELETE

请求参数参数：自定义登录态、关注号、文章作者、用户账号。

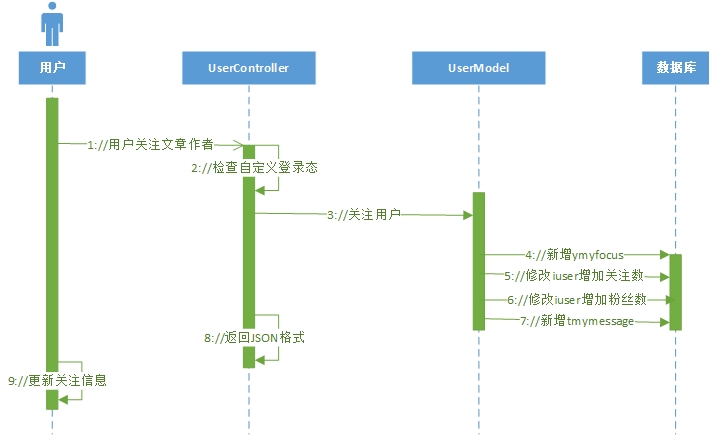
请求地址：协议+域名/wjg/user/relation/delete

操作数据表：ymyfocus、iuse

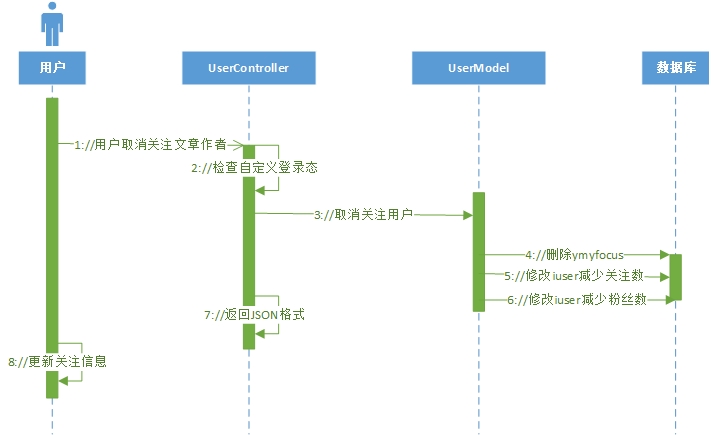
7. 流程设计

A 时序图

1. 关注用户时序图



1. 取消关注用户时序图



二）收藏文章

1. 模块描述

用户可以收藏感兴趣的文章，也可以取消对文章的收藏。



1. 输入、输出
2. 收藏文章

输入：自定义登录态、文章号、用户账号、收藏时间。

输出：无。

1. 取消收藏

输入：自定义登录态、文章号、用户账号。

输出：无。

1. 模块说明
2. 收藏文章：服务器后台收到前端传递的参数，将文章号、收藏时间和用户账号插入到gmycollection基表中。同时，再将当前文章的收藏数加一。最后修改用户的收藏数，使其加一。
3. 取消收藏：服务器后台收到前端传递的参数，根据文章号和用户账号，从gmycollection基表中删除指定数据。同时，再将当前文章的收藏数减一。最后修改用户的收藏数，使其减一。

4. 数据库设计

### 收藏表 gmycollection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Haid | Char(32) | 否 | 主键 | 收藏的文章对应的id | 文章号 |
| Udate | Char(16) | 否 | 无 | 格式：YYYY-MM-dd HH:mm | 收藏日期 |
| Wuser | Char(20) | 否 | 外键 | 收藏者，对应用户表主键 | 用户账号 |

5. 程序界面

参考图：3.3.3.2文章详情页面

6. 接口设计

1. 收藏文章

请求方式：POST

请求参数参数：自定义登录态、文章号、用户账号、收藏时间。

请求地址：协议+域名/wjg/article/detail/collect

操作数据表：gmycollection、iuser、warticle

1. 取消收藏

请求方式：DELETE

请求参数参数：自定义登录态、文章号、用户账号。

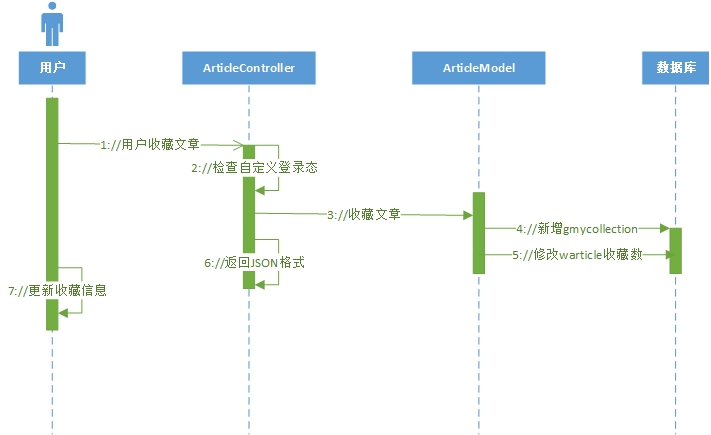
请求地址：协议+域名/wjg/article/detail/cancelCollect

操作数据表：gmycollection、iuser、warticle

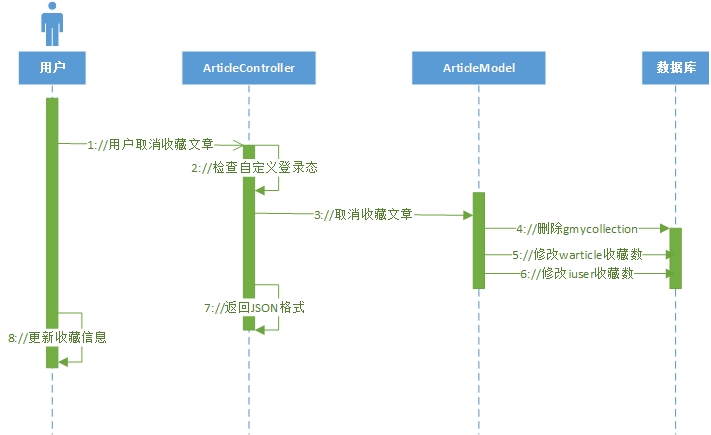
7. 流程设计

A 时序图

1. 收藏文章



1. 取消收藏



三）点赞文章

1. 模块描述

用户可以给感兴趣的文章进行点赞，同样也可以取消点赞。



2. 输入、输出

（1）点赞文章

输入：自定义登录态、文章号、文章作者、用户账号、点赞时间、消息类型。

输出：无。

（2）取消点赞

输入：自定义登录态、文章号、用户账号。

输出：无。

3. 模块说明

（1）点赞文章：服务器后台收到前端传递的参数，将文章号、和用户账号插入到jmythumbsup基表中。同时，再将当前文章的点赞数加一。最后向文章作者发送消息，告知其XXX文章被XXX点赞。

（2）取消点赞：服务器后台收到前端传递的参数，根据文章号和用户账号，从jmythumbsup基表中删除指定数据。同时，再将当前文章的点赞数减一。

4. 数据库设计

### 点赞表 jmythumbsup

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Ntid | Char(32) | 否 | 主键 | 点赞文章对应的id | 文章号 |
| Iuser | Char(20) | 否 | 外键 | 点赞者，对应用户表主键 | 用户账号 |

5. 程序界面

参考图：3.3.3.2文章详情页面

6. 接口设计

（1）点赞文章

请求方式：POST

请求参数参数：自定义登录态、文章号、文章作者、用户账号、点赞时间、消息类型。

请求地址：协议+域名/wjg/article/detail/thumbsup

操作数据表：jmythumbsup、warticle、tmymessage

（2）取消点赞

请求方式：DELETE

请求参数参数：自定义登录态、文章号、用户账号。

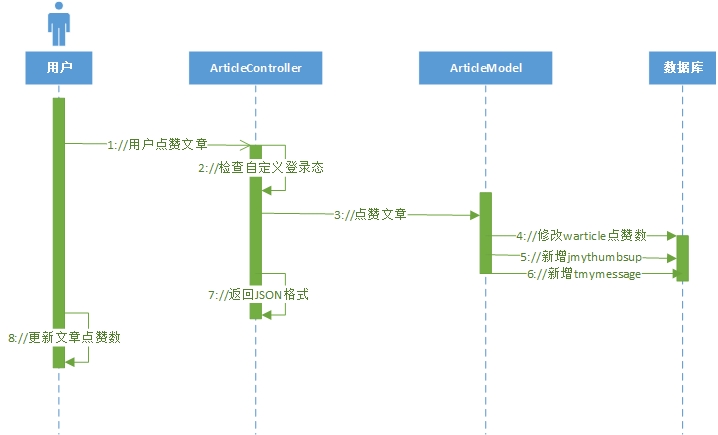
请求地址：协议+域名/wjg/article/detail/cancelThumbsup

操作数据表：jmythumbsup、warticle

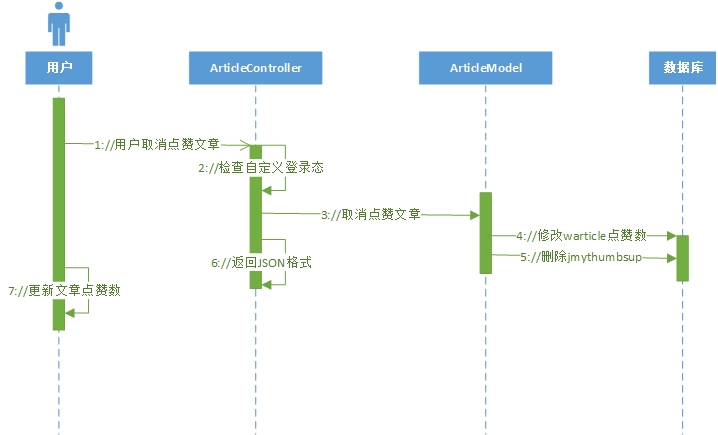
7. 流程设计

A 时序图

（1）点赞文章



（2）取消点赞



四）分享文章

1. 模块描述

用户对感兴趣的文章进行分享操作。

2. 模块说明

请参考[【微信小程序转发API】](https://developers.weixin.qq.com/miniprogram/dev/api/share.html)。

### 3.3.5 评论与回复

一．具体描述

用户进入文章详情页面后，下滑可以查看当前文章的评论。在评论区，用户可以评论文章，以及回复文章已有的评论，或者回复他人对自己评论的回复。

二．具体实现

一）获取文章评论

1. 模块描述

进入文章详情页面，小程序初始加载八条评论。每当用户下滑页面可以加载更多文章评论，每次加载八条。

1. 输入、输出

输入：自定义登录态、文章号、已获取评论的最晚日期。

输出：

1. 评论：评论号、评论内容、评论日期、图片地址、用户账号、用户头像、用户昵称、点赞数。
2. 回复：回复号、回复内容、回复日期、回复用户昵称、用户账号、用户昵称。
3. 模块说明

服务器后台根据前端传递的参数到 view\_comment 视图中先检索八条评论数据，再依据这八条评论的评论号到 view\_reply 视图中分别检索评论号对应的三条回复数据。组装完成后，返回给前端。

已获取评论的最晚日期：同“获取最新文章”的处理方式。

1. 数据库设计

### 评论表 lcomment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Cid | Char(32) | 否 | 主键 | 唯一标识评论信息 | 评论号 |
| Oaid | Char(32) | 否 | 外键 | 文章表的主键 | 文章号 |
| Ycontent | varchar(512) | 否 | 无 | 评论的内容，最多512中文 | 评论内容 |
| Udate | char(16) | 否 | 无 | 格式：YYYY-MM-dd HH:mm | 评论日期 |
| Tpicture | Varchar(32) | 是 | 无 | 评论图片的地址，只能一张 | 图片地址 |
| Xaccount | Char(20) | 否 | 外键 | 用户表的主键 | 用户账号 |
| Klike | Int(3) | 是 | 默认 | 默认0，当前评论的点赞数 | 点赞数 |

### 回复表 oreply

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Rid | Char(32) | 否 | 主键 | 唯一标识回复信息 | 回复号 |
| Hcid | Char(32) | 否 | 外键 | 评论表的主键 | 评论号 |
| Ycontent | varchar(512) | 否 | 无 | 回复的内容，最多512中文 | 回复内容 |
| Adate | char(16) | 否 | 无 | 格式：YYYY-MM-dd HH:mm | 回复日期 |
| Epicture | Varchar(32) | 是 | 无 | 评论图片的地址，只能一张 | 图片地址 |
| Waccount | Char(20) | 否 | 外键 | 用户表的主键 | 用户账号 |
| Qlike | Int(3) | 是 | 默认 | 默认0，当前评论的点赞数 | 点赞数 |
| Jnickname | Varchar(12) | 否 | 无 | 回复的用户的昵称 | 用户昵称 |

1. 程序界面

参考图3.3.3.2文章详情页面

1. 接口设计

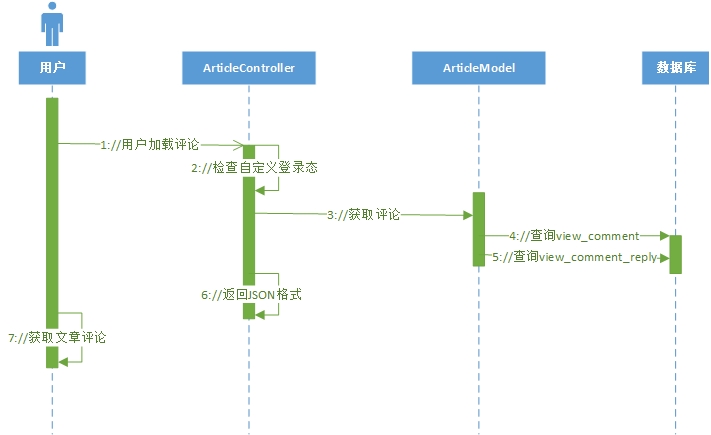
请求方式：GET

请求参数参数：自定义登录态、文章号、已获取评论的最晚日期。

请求地址：协议+域名/wjg/article/comment/get

操作数据表：view\_comment、view\_comment\_reply

1. 流程设计



二）获取文章回复

1. 模块描述

用户点击指定的文章评论即可查看当前评论的详细内容，以及与该评论相关的回复信息。

2. 输入、输出

输入：自定义登录态、评论号、已获取回复的最晚日期。

输出：回复号、回复内容、回复日期、图片地址、点赞数、回复用户昵称、回复者账号、回复者头像、回复者昵称。

3. 模块说明

服务器后台收到前端传递的参数，根据评论号以及已获取回复的最晚日期分批次从 view\_reply 视图中检索回复数据，每次返回八条数据。

4. 视图设计

### 二级评论回复视图 view\_reply

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | oreply的Rid字段，唯一标志回复信息 | 回复号 | oreply |
| cid | oreply的Hcid字段，评论号 | 评论号 | oreply |
| content | oreply的Xcontent字段，回复内容 | 回复内容 | oreply |
| date | oreply的Adate字段 | 回复日期 | oreply |
| picture | oreply的Epicture字段 | 图片地址 | oreply |
| countLike | oreply的Qlike字段 | 点赞数 | oreply |
| nickname | oreply的Jnickname字段 | 回复用户昵称 | oreply |
| account | iuser的Jaccount字段，用户账号 | 回复者账号 | iuser |
| head | iuser的Uhead字段，用户头像 | 回复者头像 | iuser |
| userNickname | iuser的Onickname字段，用户昵称 | 回复者昵称 | iuser |

5. 程序界面

参考图：3.3.3.2文章详情页面

6. 接口设计

请求方式：GET

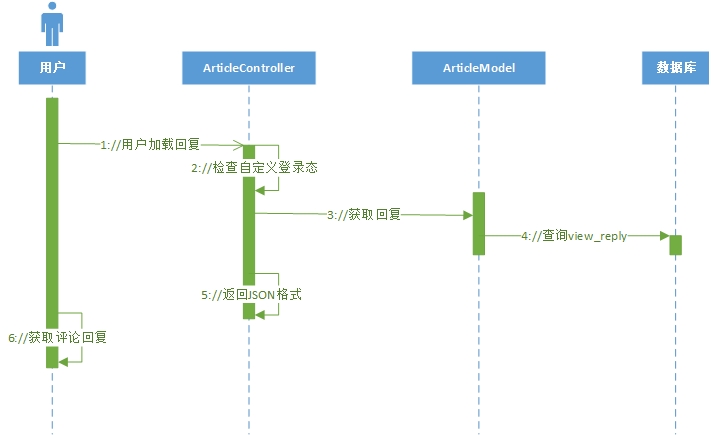
请求参数参数：自定义登录态、评论号、已获取回复的最晚日期。

请求地址：协议+域名/wjg/article/reply/get

操作数据表：view\_reply

7. 流程设计

A 时序图



三）评论和回复

1. 模块描述

用户可以对文章进行评论（一次评论），同时也可以对已有的评论进行二次评论（回复），或者对其他二次评论进行回复。

2. 输入、输出

（1）评论文章

输入：自定义登录态、文章号、评论内容、评论图片、评论时间、用户账号、消息类型、文章作者。

输出：无。

（2）回复评论

输入：自定义登录态、评论号、回复用户账号、回复用户昵称、回复内容、回复时间、图片地址、用户账号、消息类型。

输出：无。

3. 模块说明

（1）评论文章：服务器后台收到前端传递的参数，将参数插入到lcomment 基表中，其中Cid通过UUID生成。同时，再向文章作者发送消息，告知其XXX文章被XXX评论。最后修改文章的评论数，使其加一。

（2）回复评论：服务器后台收到前端传递的参数，将参数插入到 oreply 基表中，其中Rid通过UUID生成。同时，再向评论对象发送消息，告知其XXX评论被XXX评论。最后修改文章的评论数，使其加一。

4. 程序界面

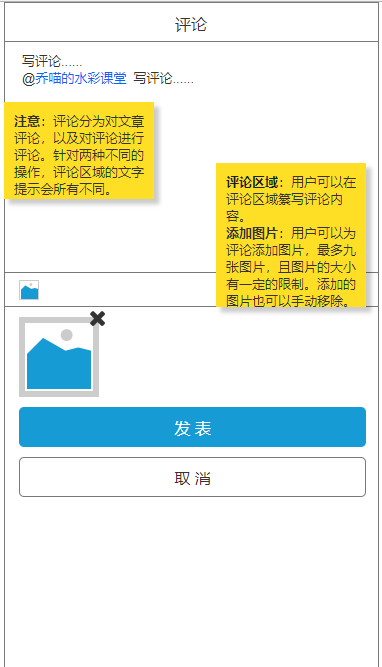


图3.3.5.1评论页面

5. 接口设计

（1）评论文章

请求方式：POST

请求参数参数：自定义登录态、文章号、评论内容、评论图片、评论时间、用户账号、消息类型、文章作者。

请求地址：协议+域名/wjg/article/comment/add

操作数据表：lcomment、warticle、tmymessage

（2）回复评论

请求方式：POST

请求参数参数：自定义登录态、评论号、回复用户账号、回复用户昵称、回复内容、回复时间、图片地址、用户账号、消息类型。

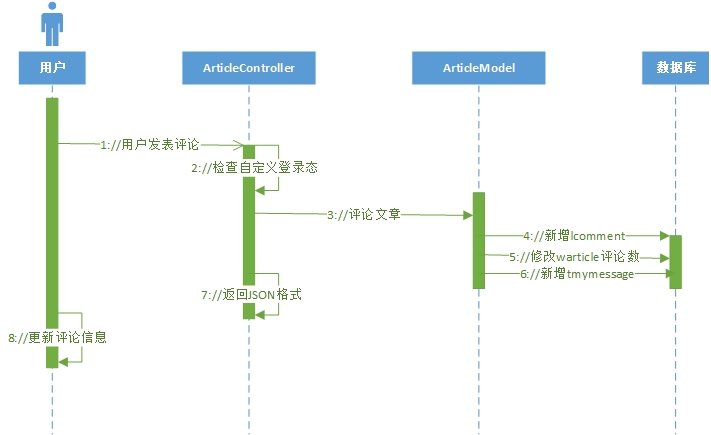
请求地址：协议+域名/wjg/article/reply/add

操作数据表：oreply、warticle、tmymessage

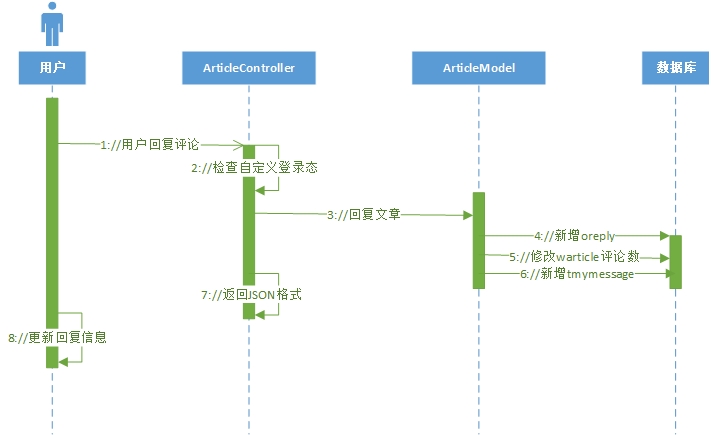
6. 流程设计

A 时序图

（1）评论文章



（2）回复评论



### 3.3.5 发表文章

一．具体描述

用户在社区主界面点击右下方的“加号图标”，即可进入到发表文章页面。在该页面，用户可以主动发布文章，与其他用户进行交流。

二．具体实现

一）上传图片

1. 模块描述

用户可以为文章配上九张图片，图片从相机或相册中选择。

1. 输入、输出

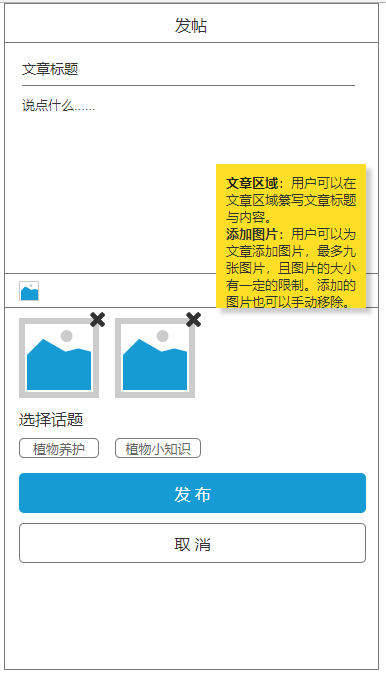
输入：自定义登录态、文章号、图片文件、图片文件名。

输出：图片路径。

1. 模块说明

文章号由前端生成并传递给后台，后台接收到图片文件和文章号之后，到服务器指定的目录下创建文章号同名的文件夹，然后将图片文件保存到该文件夹下。最后返回图片的路径。

1. 程序界面



（图3.3.5.1 发表文章页面）

1. 接口设计

请求方式：POST

请求参数参数：自定义登录态、文章号、图片文件、图片文件名。

请求地址：协议+域名/wjg/article/picture/upload

操作数据表：warticle

1. 流程设计
2. 时序图
3. 获取话题

1. 模块描述

每一篇文章都有相应所属的话题，用户在发表文章前必须选择合适的话题。并且管理员可以时刻修改话题，因此话题需要通过动态的方式获取，而非硬编码在小程序中，故而需要额外编写一个接口用以获取话题列表。

1. 输入、输出

输入：自定义登录态。

输出：话题号、话题名称。

1. 模块说明

服务器后台收到前端的请求后，从view\_topic视图中检索出所有的话题，并返回给前端。

1. 数据库设计

### 话题表 ftopic

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Tid | Int(2) | 否 | 主键 | 唯一标话题信息，自增长 | 话题号 |
| Ytitle | Varchar(10) | 否 | 唯一 | 最多10个中文，话题名称 | 话题名称 |
| Cbrief | varchar(128) | 否 | 无 | 最多128个中文，话题简介 | 话题简介 |

1. 视图设计

### 话题视图 view\_topic

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | ftopic的Tid字段，唯一标志话题信息 | 话题号 | ftopic |
| title | ftopic的Ytitle字段，话题名称 | 话题名称 | ftopic |
| brief | ftopic的Cbrief字段，话题简介 | 话题简介 | ftopic |

1. 程序界面

参考图3.3.5.1发表文章页面。

1. 接口设计

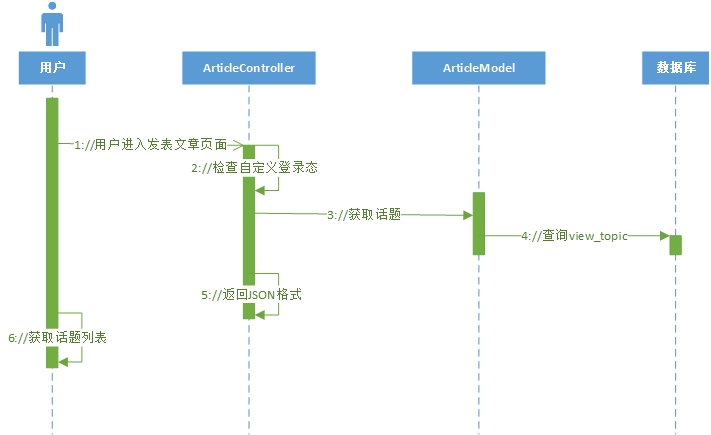
请求方式：GET

请求参数参数：自定义登录态。

请求地址：协议+域名/wjg/article/topic/get

操作数据表：view\_topic

1. 流程设计
2. 时序图



三）发表或取消发表文章

1. 模块描述

用户在编写完文章内容后，即可点击“发布”按钮提交文章。点击“取消”按钮或回退图标都可取消发表文章。



2. 输入、输出

（1）发表文章

输入：自定义登录态、文章号、文章标题、文章内容、文章日期、图片地址、用户账号、话题号。

输出：无。

（2）取消发表

输入：自定义登录态、文章号。

输出：无。

3. 模块说明

（1）发表文章：服务器后台收到前端传递的参数，将参数插入到warticle 基表中。同时，再修改当前用户的文章数（iuser），使其加一。

（2）取消发表：服务器后台收到前端传递的文章号，并根据文章号到服务器指定目录下删除与文章号同名的文件夹。

4. 程序界面

参考图3.3.5.1发表文章页面

5. 接口设计

（1）发表文章

请求方式：POST

请求参数参数：自定义登录态、文章号、文章标题、文章内容、文章日期、图片地址、用户账号、话题号。

请求地址：协议+域名/wjg/article/detail/add

操作数据表：warticle、iuser

（2）取消发表

请求方式：DELETE

请求参数参数：自定义登录态、文章号。

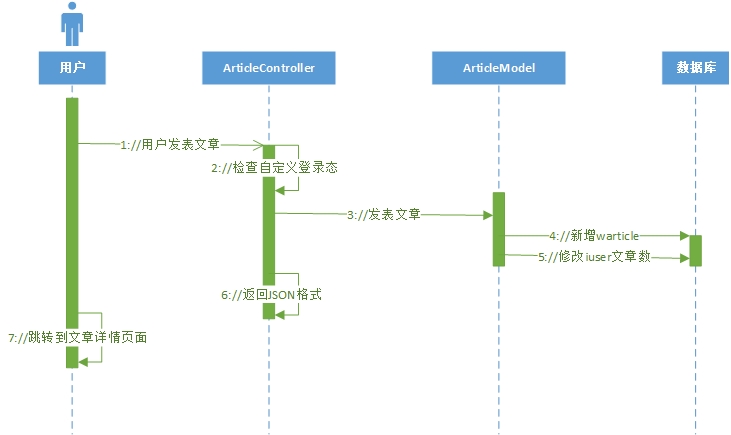
请求地址：协议+域名/wjg/article/picture/deleteAll

操作数据表：无

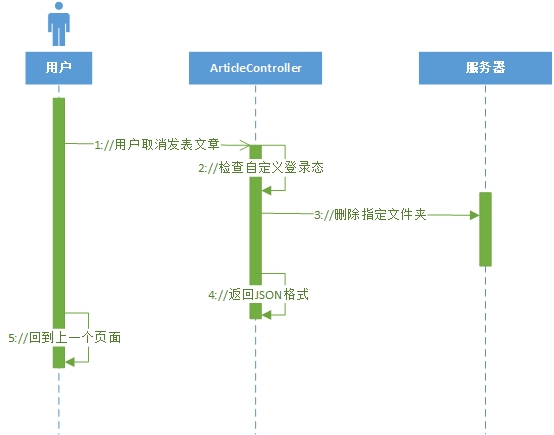
6. 流程设计

A 时序图

（1）发表文章



（2）取消发表



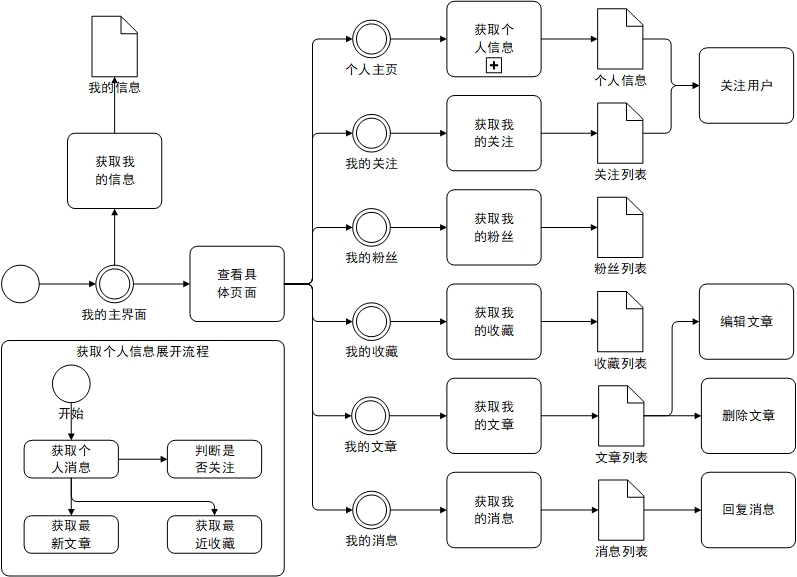
## 我的模块

在我的模块中，用户可以查看个人的相关信息，包括用户头像、昵称、关注的用户、个人粉丝以及收藏的文章和自行发表的文章。除此之外，用户可以查看与自己有关的消息，例如某某关注了你，对你的文章进行点赞或评论了你的文章。我的模块主要分为两部分：个人信息查阅以及信息管理两部分。

### **需求概述**

1. 信息查阅：用户可以查询个人或者其他用户的信息。信息主要为用户的基础信息、关注、粉丝、收藏文章以及编写的文章。第一版的功能中暂未加入帮助中心以及联系我们，在之后版本可以查阅更多的信息。
2. 信息管理：用户可以对已有的信息进行简单的管理，例如取消对已有用户的关注，或者编辑、删除已有的文章。注意，关注用户和回复消息操作可以直接沿用社区模块中的“关注用户”以及“回复评论”。

### 流程图



### 信息查阅

一．具体描述：

用户进入到我的模块主界面或用户个人主页时，小程序会自动从服务器获取当前所查看用户的信息，包括用户头像、昵称等信息。

二．具体流程：

一）获取我的信息

1. 模块描述

用户进入到我的模块主界面时，小程序会自动从服务器获取当前用户的信息，包括用户头像、昵称等信息。

1. 输入、输出

输入：自定义登录态、用户账号。

输出：用户头像、用户昵称、关注数、粉丝数、收藏数、文章数。

1. 模块说明

服务器后台接收到前端传递的参数后，根据用户账号从view\_user视图中检索相应的用户信息，并返回给前端。

1. 数据库设计

### 用户表 iuser

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据格式 | 空 | 约束 | 说明 | 中文解释 |
| Hwxkey | Char(28) | 否 | 主键 | 微信平台用户的唯一标识 | 微信标识号 |
| Jaccount | Char(20) | 否 | 唯一 | 微信标识号对应账号 | 用户账号 |
| Uhead | varchar(32) | 否 | 无 | 用户的头像 | 用户头像 |
| Onickname | varchar(12) | 否 | 无 | 最长12个中文 | 用户昵称 |
| nfocus | Int(3) | 是 | 默认 | 默认：0，关注数 | 关注数 |
| Mfans | Int(3) | 是 | 默认 | 默认：0，粉丝数 | 粉丝数 |
| Hcollect | Int(3) | 是 | 默认 | 默认：0，收藏数 | 收藏数 |
| Darticle | Int(3) | 是 | 默认 | 默认：0，文章数 | 文章数 |
| Tdate | Char(16) | 否 | 无 | 用户开始使用小程序的日期 | 创建日期 |

1. 视图设计

### 用户视图 view\_user

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| Wxkey | iuser的Hwxkey字段 | 微信标识号 | iuser |
| account | iuser的Jaccount字段，用户账号 | 用户账号 | iuser |
| head | iuser的Uhead字段，用户头像 | 用户头像 | iuser |
| nickname | iuser的Onickname字段，用户昵称 | 用户昵称 | iuser |
| countFocus | iuser的Nfocus字段，关注数 | 关注数 | iuser |
| countFans | iuser的Mfans字段，粉丝数 | 粉丝数 | iuser |
| countCollect | iuser的Hcollect字段，收藏数 | 收藏数 | iuser |
| countArticle | iuser的Darticle字段，文章数 | 文章数 | iuser |
| date | iuser的Tdate字段，创建日期 | 创建日期 | iuser |

1. 程序界面



（图：3.4.3.1 我的主界面）

1. 接口设计

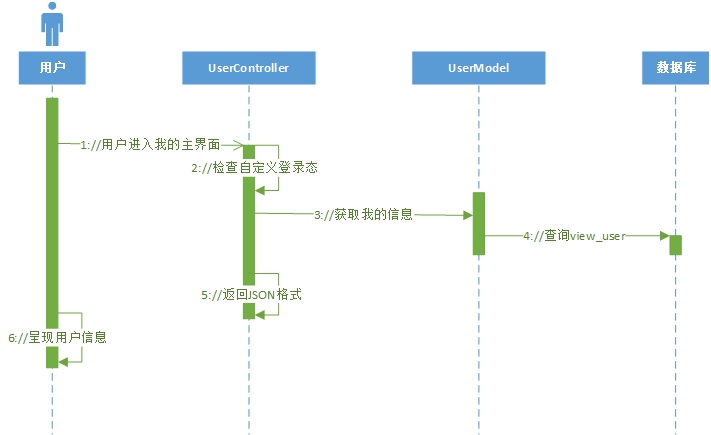
请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/user/detail/get

操作数据表：view\_user

1. 流程设计
2. 时序图



二）获取我的关注

1. 模块描述

用户点击“我的关注”后，跳转到我的关注页面。在该页面，用户可以查看已关注的用户。



1. 输入、输出

输入：自定义登录态、用户账号。

输出：关注用户账号、关注用户头像、关注用户昵称。

1. 模块说明

服务器后台收到前端传递的参数后，根据“用户账号”从view\_user\_focus视图中检索当前用户所关注的其他用户信息，并返回给前端。

1. 视图设计

### 关注视图 view\_user\_focus

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| account | iuser的Jaccount字段，用户账号 | 关注用户账号 | iuser |
| head | iuser的Uhead字段，用户头像 | 关注用户头像 | iuser |
| nickname | iuser的Onickname字段，用户昵称 | 关注用户昵称 | iuser |
| user | ymyfocus的Gaccount字段，用户账号 | 用户账号 | ymyfocus |

1. 程序界面



图3.4.3.2我的关注页面

1. 接口设计

请求方式：GET

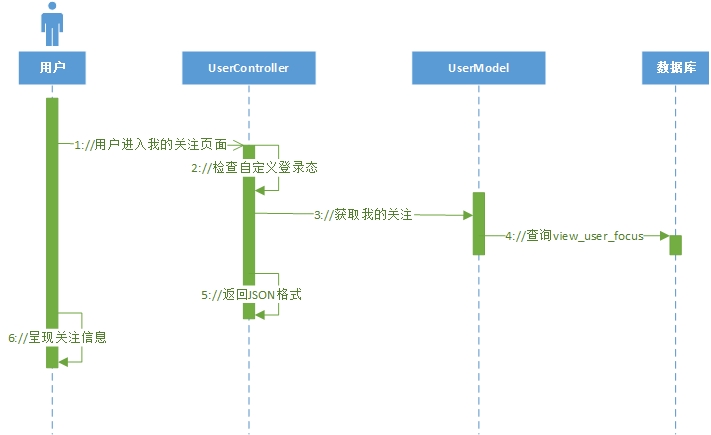
请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/user/relation/get

操作数据表：view\_user\_focus

1. 流程设计

A. 时序图



三）获取我的粉丝

1. 模块描述

用户点击“我的粉丝”后，跳转到我的粉丝页面。在该页面，用户可以查看关注自己的其他用户。



1. 输入、输出

输入：自定义登录态、用户账号。

输出：粉丝账号、粉丝头像、粉丝昵称。

1. 实现过程

服务器后台在收到前端传递的参数后，根据“用户账号”从view\_user\_fans视图中检索相应的粉丝用户信息，并返回给前端。

1. 视图设计

### 粉丝视图 view\_user\_fans

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| account | iuser的Jaccount字段，用户账号 | 关注用户账号 | iuser |
| head | iuser的Uhead字段，用户头像 | 关注用户头像 | iuser |
| nickname | iuser的Onickname字段，用户昵称 | 关注用户昵称 | iuser |
| user | ymyfocus的Gaccount字段，用户账号 | 用户账号 | ymyfocus |

1. 程序界面

参考图3.4.3.2我的关注页面

1. 接口设计

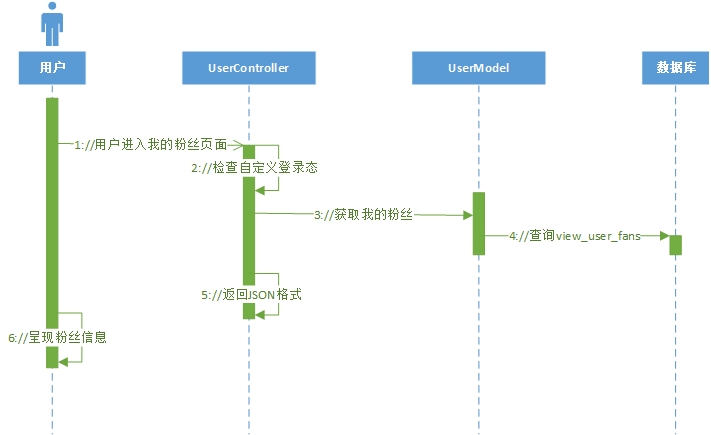
请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/user/relation/getFans

操作数据表：view\_user\_fans

1. 流程设计
2. 时序图



四）获取我的收藏

1. 模块描述

用户点击“我的收藏”后，跳转到我的收藏页面。在该页面，用户可以查看已经收藏的文章。

1. 输入、输出

输入：自定义登录态、用户账号。

输出：文章号、文章标题、收藏日期。

1. 模块说明

服务器后台在收到前端传递的参数后，根据用户账号从view\_user\_collect视图中检索相应的收藏数据，并返回给前端。

1. 视图设计

### 收藏视图 view\_user\_collect

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | warticle的Aid字段，文章号 | 文章号 | warticle |
| title | warticle的Htitle字段，文章标题 | 文章标题 | warticle |
| date | gmycollection的Udate字段，收藏日期 | 收藏日期 | gmycollection |
| account | gmycollection的Wuser字段，用户账号 | 用户账号 | gmycollection |

1. 程序界面



图3.4.3.3我的收藏页面

1. 接口设计

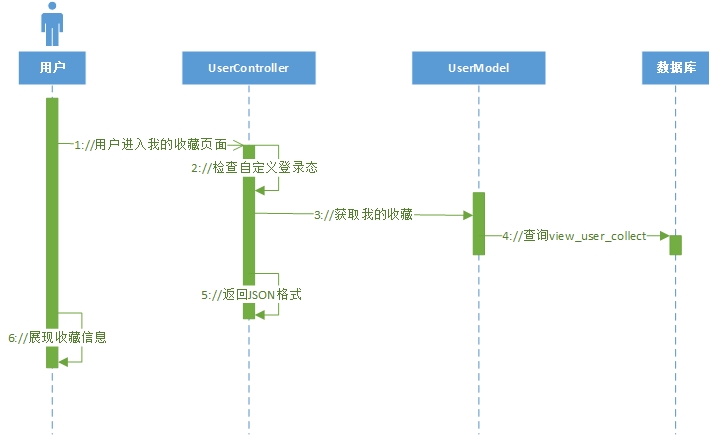
请求方式：GET

请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/user/collection/get

操作数据表：view\_user\_collect

1. 流程设计
2. 时序图



五）获取我的文章

1. 模块描述

用户点击“我的文章”后，跳转到我的文章页面。在该页面，用户可以查看已经发布的个人文章。

2. 输入、输出

输入：自定义登录态、用户账号。

输出：文章号、文章标题、发布日期。

3. 模块说明

服务器后台在收到前端传递的参数后，根据用户账号从view\_user\_article视图中检索相应的文章数据，并返回给前端。

4. 视图设计

### 我的文章视图 view\_user\_article

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| id | warticle的Aid字段，文章号 | 文章号 | warticle |
| title | warticle的Htitle字段，文章标题 | 文章标题 | warticle |
| date | warticle的Kdate字段，发布日期 | 发布日期 | warticle |
| account | warticle的Maccount字段，用户账号 | 用户账号 | warticle |

5. 程序界面

参考图3.4.3.3我的收藏页面

6. 接口设计

请求方式：GET

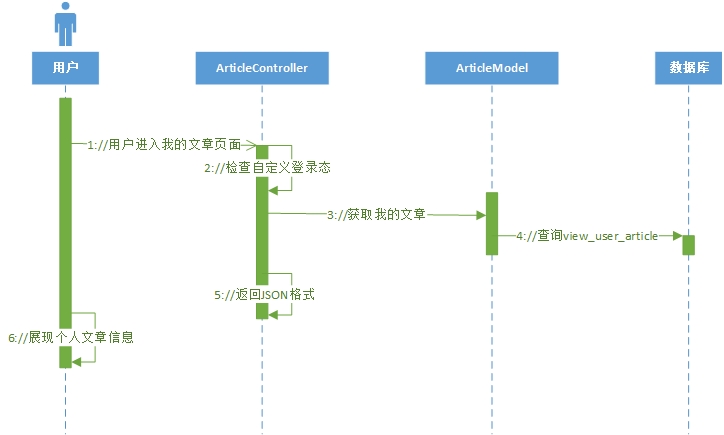
请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/article/list/get

操作数据表：view\_user\_article

7. 流程设计

A. 时序图



六）获取我的消息

1. 模块描述

用户点击“我的消息”后，跳转到我的消息页面。在该页面，用户可以查看接收到的个人消息。

2. 输入、输出

输入：自定义登录态、用户账号。

输出：发送者账号、发送者头像 、发送者昵称、文章号、文章标题、消息发送日期、消息类型、消息内容。

3. 模块说明

服务器后台在收到前端传递的参数后，根据用户账号从view\_message视图中检索相应的消息数据，并返回给前端。

4. 视图设计

### 我的消息视图 view\_message

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 说明 | 中文解释 | 所属基表 |
| sender | iuser的Jaccount字段，发送者账号 | 发送者账号 | iuser |
| head | iuser的Uhead字段，发送者头像 | 发送者头像 | iuser |
| nickname | iuser的Onickname字段，发送者昵称 | 发送者昵称 | iuser |
| id | warticle的Aid字段，文章号 | 文章号 | warticle |
| title | warticle的Htitle字段，文章标题 | 文章标题 | warticle |
| date | tmymessage的Qdate字段，消息发送日期 | 消息发送日期 | tmymessage |
| type | tmymessage的Ptype字段，消息类型 | 消息类型 | tmymessage |
| comment | tmymessage的Lcomment字段，消息内容 | 消息内容 | tmymessage |
| account | tmymessage的Muser字段，用户账号 | 用户账号 | tmymessage |

5. 程序界面



图3.4.3.4我的消息页面

6. 接口设计

请求方式：GET

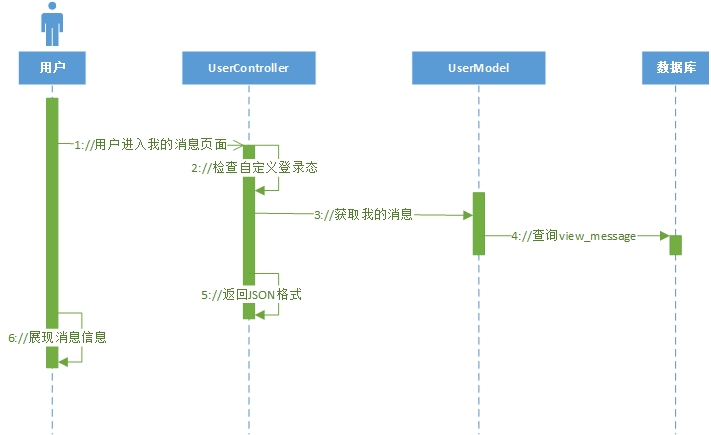
请求参数参数：自定义登录态、用户账号。

请求地址：协议+域名/wjg/user/message/get

操作数据表：view\_message

7. 流程设计

A. 时序图



七）获取个人消息

1. 模块描述

点击用户头像或昵称即可进入到用户的个人主页。在个人主页中，可以查看当前用户的个人基础信息以及最新发表的文章和收藏的文章。

2. 输入、输出

输入：自定义登录态、查看用户账号、用户账号。

输出：用户头像、用户昵称、关注数、粉丝数、收藏数、文章数、最新文章（文章号、文章标题、文章内容、评论数、点赞数、图片地址）、最近收藏（文章号、文章标题、收藏日期）。

3. 模块说明

服务器后台在收到前端传递的参数后，分别执行以下操作：

1. 根据查看用户账号从view\_user视图中检索相应的用户数据（用户头像、用户昵称、关注数、粉丝数、收藏数、文章数）；
2. 根据查看用户账号和用户账号，从view\_judge\_focus视图中检索是否存在数据。若存在，则表示当前用户已经被关注；否则，则表示当前用户没有被关注；
3. 再从view\_article视图中检索一条用户的最新文章数据（文章号、文章标题、文章内容、评论数、点赞数、图片地址）；
4. 然后从view\_user\_collect视图中检索三条收藏数据（文章号、文章标题、收藏日期）；
5. 最后组合成一个完整的JSON对象返回给前端。
6. 程序界面



图3.4.3.5个人主页页面

6. 接口设计

请求方式：GET

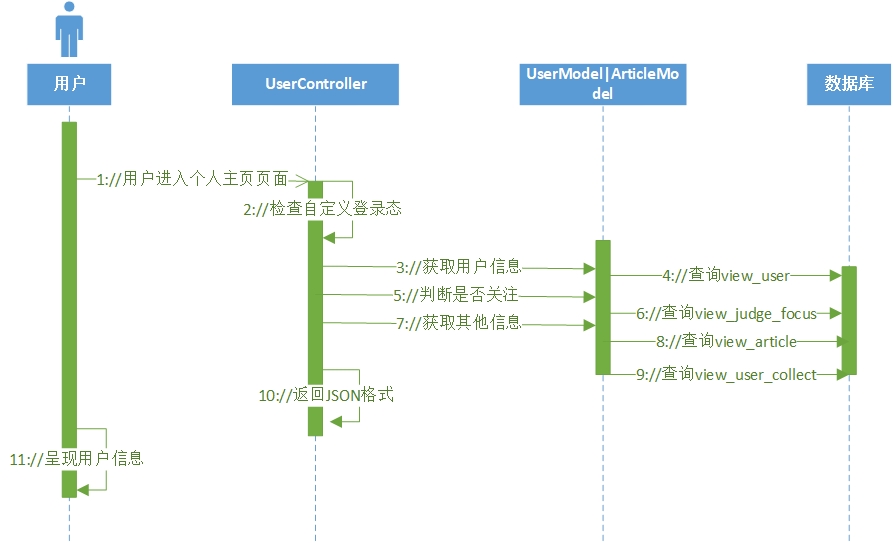
请求参数参数：自定义登录态、查看用户账号、用户账号。

请求地址：协议+域名/wjg/user/detail/getIndex

操作数据表：view\_user、view\_judge\_focus、view\_article、view\_user\_collect

7. 流程设计

A. 时序图



### **3.4.4 信息管理**

一．具体描述

用户可以对已有的个人信息进行管理，目前主要分为对文章的编辑和删除，以及对消息的回复。

二．具体实现

一）编辑文章

1. 模块描述

用户可以对自己的文章进行编辑。

1. 输入、输出

（1）获取文章详情

输入：自定义登录态、文章号。

输出：文章号、文章标题、内容、日期、图片地址、话题号、话题名称。

（2）提交修改

输入：自定义登录态、文章号、文章标题、文章内容、日期、图片地址、话题号。

输出：无。

（3）删除图片：

输入：自定义登录态、文章号、图片地址。

输出：无。

1. 模块说明
2. 获取文章详情：用户编辑文章时，首先获取话题列表（可参考“获取话题”操作流程）。然后再根据前端传递的文章号，从view\_article视图中检索对应文章号的文章，并返回给前端。
3. 提交修改：服务器后台收到前端传递的参数后，将参数update到warticle指定数据中。
4. 图片相关：当用户删除文章的原图片时，前端会传递该图片对应的地址。服务器则将对应的图片从服务器中移除。当用户选择新的图片上传时，操作流程同社区模块中的“图片上传”操作流程。
5. 程序界面

暂时还没有制作。

1. 接口设计

（1）提交修改

请求方式：PATCH

请求参数参数：自定义登录态、文章号、文章标题、文章内容、日期、图片地址、话题号。

请求地址：协议+域名/wjg/article/detail/edit

操作数据表：warticle

（2）删除图片

请求方式：DELETE

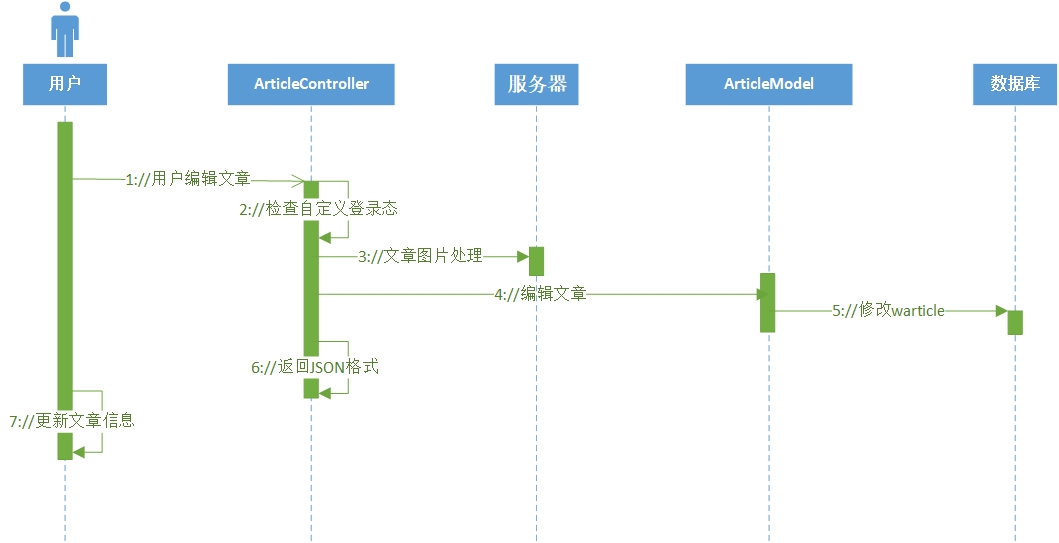
请求参数参数：自定义登录态、文章号、图片地址。

请求地址：协议+域名/wjg/article/picture/delete

操作数据表：无

1. 流程设计

A. 时序图



二）删除文章

1. 模块描述

用户可以删除已有的文章。删除文章之后，若其他用户收藏该文章，则显示为“该文章已被作者删除”。

1. 输入、输出

输入：自定义登录态、文章号。

输出：无。

1. 模块说明

服务器后台根据前端传递的文章号，从warticle基表中删除指定文章号对应的文章。然后再到服务器指定目录下删除与文章相关的图片（与文章号同名的文件夹）。

1. 程序界面

参考图3.4.3.3我的收藏页面。

1. 接口设计

请求方式：GET

请求参数参数：自定义登录态、文章号。

请求地址：协议+域名/wjg/article/detail/delete

操作数据表：warticle

1. 流程设计
2. 时序图

