

**wink需求分析规约**

软件工程课程设计



**指导教师：**

**杜庆峰**

**小组成员：**

**1752026 李航**

**1652977 王星洲**

**1650350 乔宇**

**1753499 潘小逸**

**2019-11-9**

目录

[修订历史 2](#_Toc29325318)

[1. 引言 3](#_Toc29325319)

[1.1 项目背景 3](#_Toc29325320)

[1.1.1 背景介绍 3](#_Toc29325321)

[1.1.2 必要性 3](#_Toc29325322)

[1.2 参考资料 3](#_Toc29325323)

[1.3 假定和约束 4](#_Toc29325324)

[1.4 用户特点 4](#_Toc29325325)

[2. 系统概述 4](#_Toc29325326)

[2.1 设备 4](#_Toc29325327)

[2.2 与第三方平台的接口 4](#_Toc29325328)

[2.3 普通接口 4](#_Toc29325329)

[3. 功能需求 4](#_Toc29325330)

[3.1系统范围 4](#_Toc29325331)

[3.2系统总体流程 4](#_Toc29325332)

[3.3系统体系结构 4](#_Toc29325333)

[3.4需求分析 5](#_Toc29325334)

[3.4.1 内容建模 5](#_Toc29325335)

[3.4.2 功能建模 10](#_Toc29325336)

[3.4.3 交互建模 17](#_Toc29325337)

[3.4.4 配置建模 20](#_Toc29325338)

[3.4.5 导航建模 20](#_Toc29325339)

[4. 非功能需求 21](#_Toc29325340)

[4.1 性能要求 21](#_Toc29325341)

[4.1.1 用户体验 21](#_Toc29325342)

[4.1.2 时间特性要求 21](#_Toc29325343)

[4.1.3 输入输出要求 22](#_Toc29325344)

[4.2 数据管理能力要求 22](#_Toc29325345)

[4.2.1 常量约定 22](#_Toc29325346)

[4.2.2 数据存储要求 22](#_Toc29325347)

[4.2.3 负载 22](#_Toc29325348)

[4.3 安全及保密性要求 22](#_Toc29325349)

[4.3.1 软件使用数据 22](#_Toc29325350)

[4.3.2 用户账号数据 22](#_Toc29325351)

[4.4 灵活性要求 22](#_Toc29325352)

[4.5 其他要求 22](#_Toc29325353)

[4.5.1 可维护性 22](#_Toc29325354)

[4.5.2 灾难恢复 23](#_Toc29325355)

[4.5.3 法律限制 23](#_Toc29325356)

# 修订历史

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编写日期 | SEPG | 版本 | 说明 | 作者 | 评审时间 | 评审人员 | 批准日期 | 签字人员 |
| 2019/11/12 | wink | 1.0.01112\_alpha | 初稿 | 李航 | 2019/11/12 | 乔宇 | 2019/11/12 | 乔宇 |
| 2019/11/19 | wink | 1.0.01119\_alpha | 数据建模 | 乔宇 | 2019/11/19 | 李航 | 2019/11/19 | 李航 |
| 2019/11/26 | wink | 1.0.11126\_alpha | 数据建模修改，数据存储要求 | 乔宇 | 2019/11/26 | 李航 | 2019/11/26 | 李航 |
| 2019/12/17 | wink | 1.2.01217.alpha | 修改数据建模、添加功能建模和行为建模 | 王星洲 | 2019/12/17 | 李航 | 2019/12/17 | 李航 |
| 2019/12/20 | wink | 1.3.01220\_alpha | 非功能需求 | 李航 | 2019/12/20 | 潘小逸 | 2019/12/20 | 潘小逸 |
| 2019/12/23 | wink | 1.4.21223\_alpha | 修改需求 | 乔宇 | 2019/12/23 | 李航 | 2019/12/23 | 李航 |
| 2019/12/27 | wink | 1.5.01227\_alpha | 改通用3阶段建模为移动APP5阶段建模 | 李航 | 2019/12/27 | 王星洲 | 2019/12/27 | 王星洲 |
| 2020/01/07 | wink | 2.0.20107\_beta | 完成文档整合 | 潘小逸 | 2020/01/07 | 李航 | 2020/01/07 | 李航 |

# 1. 引言

## 1.1 项目背景

### 1.1.1 背景介绍

互联网技术的不断发展，使得国内拥有的移动终端的用户数量已经远远超过了个人电脑的拥有量，并且目前移动互联网用户数已经超过了宽带上网用户数，据调查，目前有超过60%的人通过手机接入互联网，因此移动互联网这个信息平台也成为各个媒体抢占数据流量的重要入口，而对于新闻媒体来说，这种便捷的信息传输平台为保证新闻时效性提供了更好的技术保障，所以随着移动互联平台成为信息传输和接收的主流渠道平台，各个新闻媒体也开始争抢移动互联网这块蛋糕。所以，在智能手机普及之后，各个新闻网站首先开始推出自己网站的 APP 服务 （Application 第三方智能手机应用程序），随后其他的传统媒体也及时跟进，三联生活周刊等纸媒也迅速推出了自身的 APP 软件。我们正是参考这些软件，制作一款具有特色的全新的新闻APP。

### 1.1.2 必要性

市场上的新闻APP已经五花八门，他们都能看实时新闻、热点消息等。但是有些APP不免具有一些缺点：如有大量的广告，还有一些以“多刷赚钱”为噱头吸引顾客长时间停留在他们的APP中。我们这款APP不仅可以查看实时新闻、评论新闻、查看视频等，还可以就看过的系列新闻、经历的事情或此刻的心情发表专属于自己的动态。在动态圈中，用户可以寻找志同道合的好友，共同探讨新闻。

## 1.2 参考资料

1. Pressman R S. Software Engineering A Practitioner's Approach[M]. Seventh Edition. McGraw- Hill Higher Education, 2017-1.

## 1.3 假定和约束

**发布时间：**2019年12月27日发布beta释放版

**android版本要求：**android SDK 16及以上

**开发条件：**成员自备电脑与移动应用，配置开发环境

**经费支持：**无经费支持

**设计成本：**自行承担设计成本

**开发效率：**使用SCRUM敏捷开发模型

## 1.4 用户特点

1. 经常喜欢看新闻的用户

2. 喜欢分享见闻的用户

3. 乐于交友的用户

4. 用户间互动性比较强，会互相评论动态进行讨论

# 2. 系统概述

## 2.1 设备

在项目调试过程中，我们分别在Pixel 2 API 28、HUAWEI HUAWEI NXT-AL10、HUAWEI YAL-AL00等一系列实机或仿真器上运行正常，项目最低支持Android SDK 16，基本覆盖100%的安卓设备。

## 2.2 与第三方平台的接口

1. 聚合数据-新闻头条接口

该接口提供最新新闻头条，各类社会、国内、国际、体育、娱乐、科技等资讯，更新周期为5-30分钟。通过周期性访问接口获取新闻头条的题目，日期，分类，作者，链接等信息，并写入数据库作为数据库所提供的新闻。

接口地址：<http://v.juhe.cn/toutiao/index>

返回格式：json

请求方式：get/post

请求示例：http://v.juhe.cn/toutiao/index?type=top&key=APPKEY

# 3. 功能需求

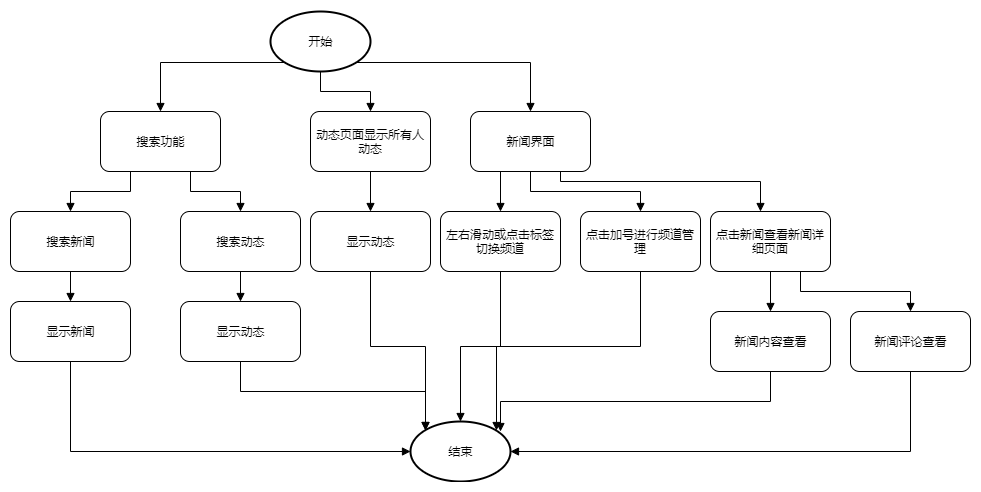
## 3.1系统范围

喜欢并常看新闻的人对于一些新闻必定有自己的看法。我们提供了这样一个平台：让看新闻的人能够发表自己的见解，并且，他可以选择关注一些与自己看法相近的人，并对他们感兴趣的话题评论，或者对他们的动态进一步讨论。这样，很多志趣相近的人便走到一起，并有聊不完的新鲜话题。

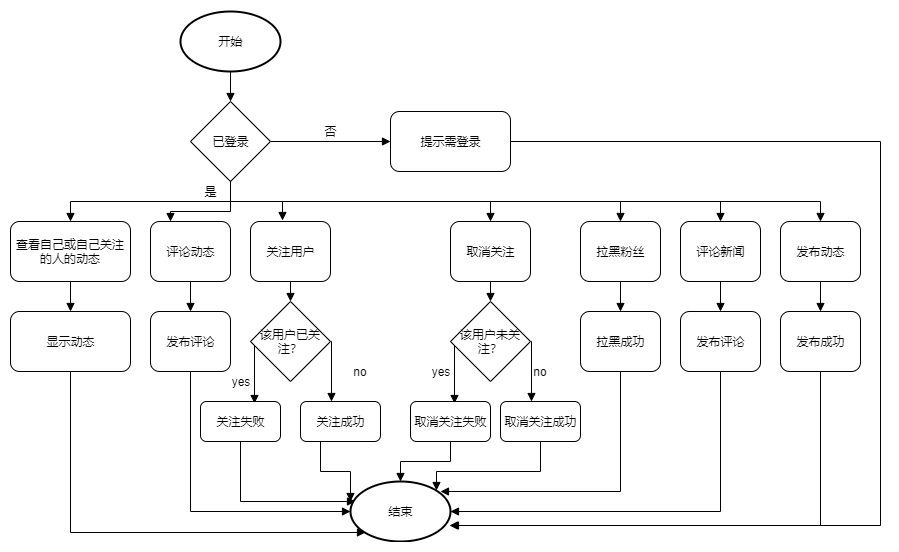
## 3.2系统总体流程

对于本系统总体的四大模块：浏览模块、社交模块、个人收藏模块和账户管理模块，分别构建了他们的整体流程图，如下：

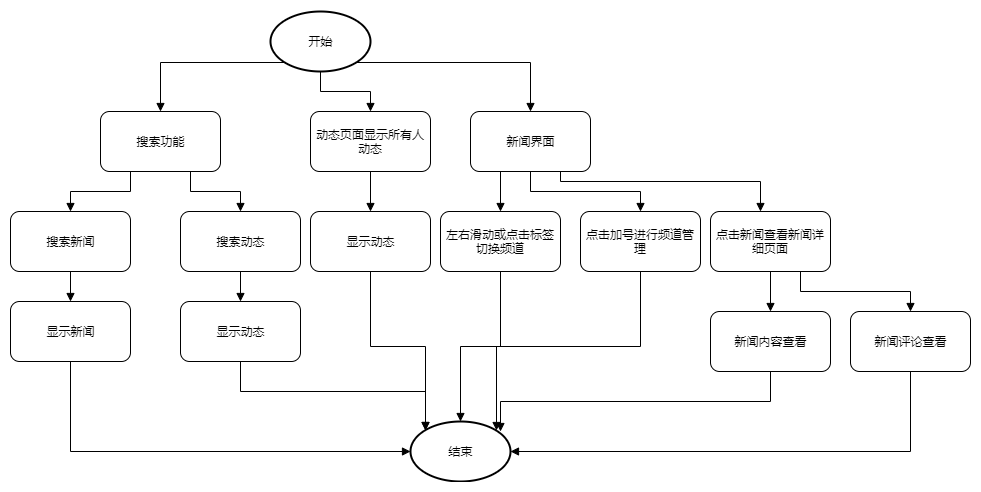
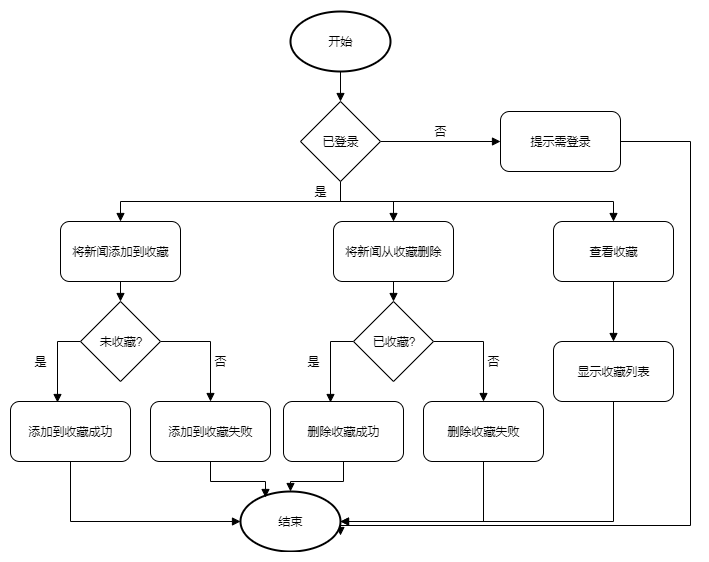
### 3.2.1 浏览模块



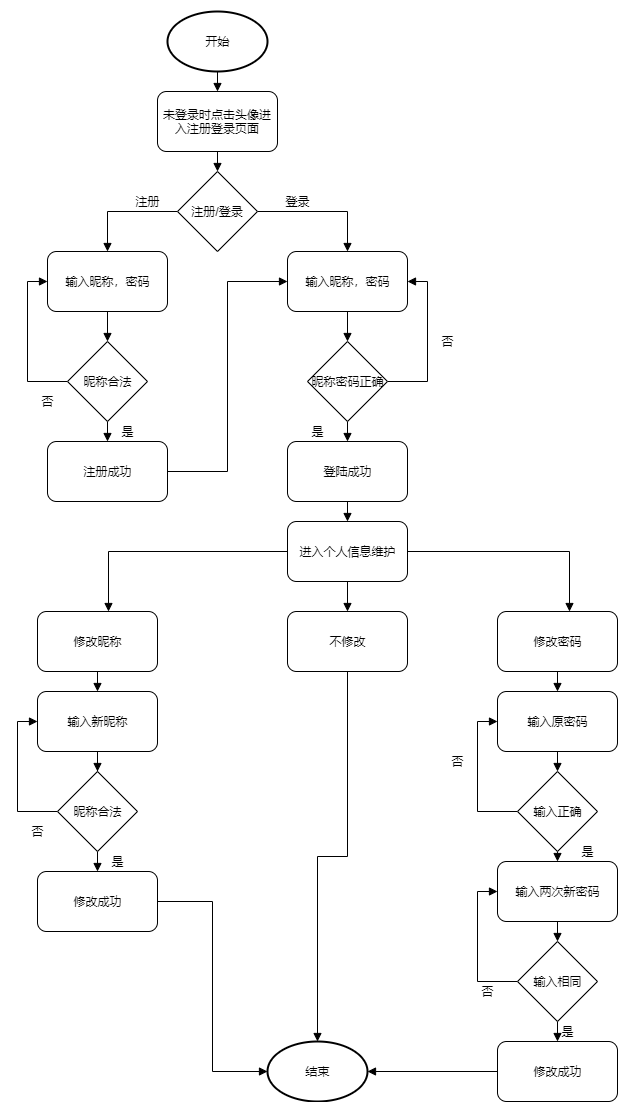
### 3.2.2 社交模块



### 3.2.3 个人收藏模块



### 3.2.4 账户管理模块



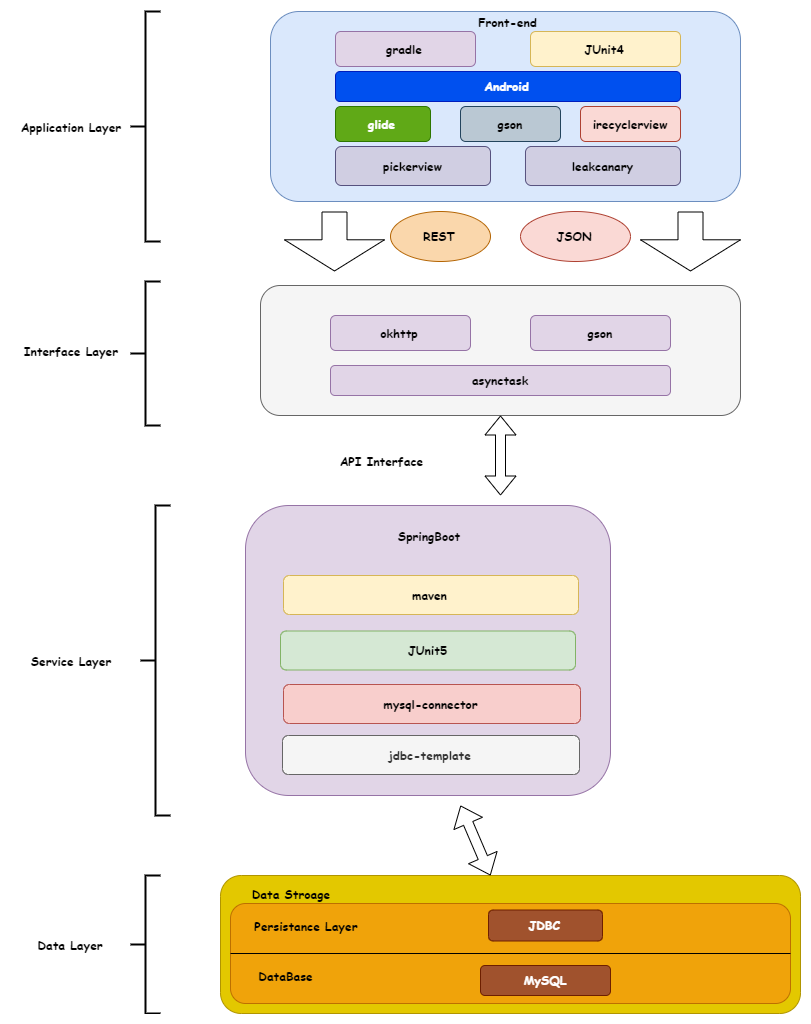
## 3.3系统体系结构

应用层(前端)由于本应用是新闻类应用，用户希望随时随地使用的原因而采用了移动端开发，这里我们选择了安卓平台开发。采用了JUnit4和leakcanary进行测试以及内存检测，同时我们使用了irecyclerview，pickerview等用于组件设计。

接口层采用了okhttp发送http请求，用gson进行json与类之间的转换，asynctask用来实现异步访问。前后端接口设计采用了RESTful API

服务层(后端)我们采用了SpringBoot框架，用maven安装了依赖，并进行依赖注入，采用Junit5进行单元测试并用mysql-connector和jdbc-template实现与MySQL数据库的连接。

数据层我们选择了MySQL数据库，这是一个关系型数据库。并实现了它与后端的连接。具体架构图如下：



## 3.4需求分析

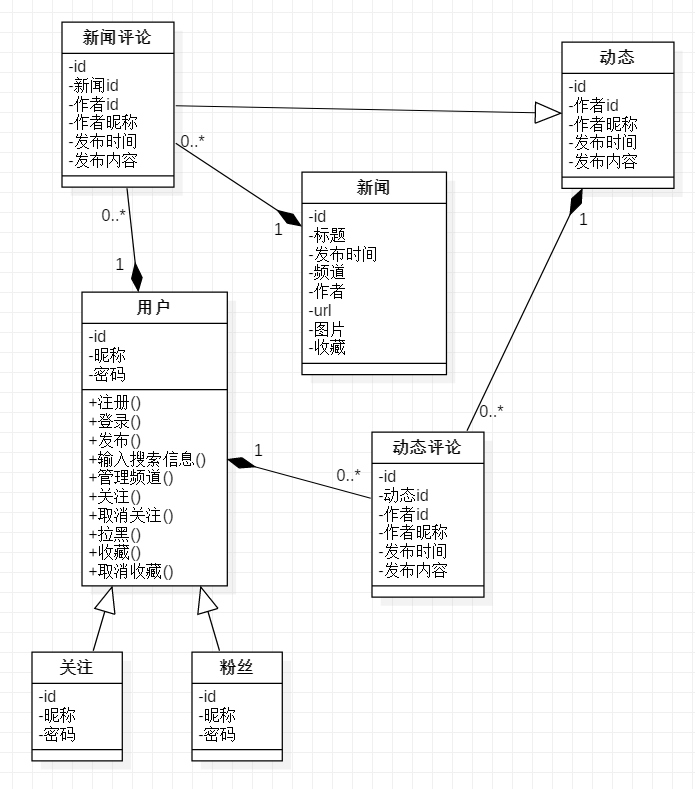
在这一部分，我们将进一步细化需求。区别于普通的应用，移动应用的需求分析在建模过程中增加了许多独特的内容，比如组件建模、导航建模等等。完整一些讲，移动APP的需求分析模型产出有5个：内容建模、功能建模、交互建模、配置建模、导航建模。下面我展示建模过程及模型产品。

### 3.4.1 内容建模

* 根据对需求规约中的主要功能需求文字描述的名词/名词词组的提取，我们提取出以下潜在类，并对潜在类接受或拒绝六大特征进行进一步的分析讨论。

|  |  |  |
| --- | --- | --- |
| 潜在类 | 一般分类 | 处理 |
| 用户 | 角色 | 接受 |
| 关键词 | 事物 | 拒绝 |
| APP | 场地 | 拒绝 |
| 标题 | 事物🡪属性 | 拒绝 |
| 新闻 | 事物 | 接受 |
| 评论 | 事物 | 接受 |
| 页面 | 组织单元 | 拒绝 |
| 频道/频道栏 | 事物🡪属性/组织单元 | 接受🡪拒绝 |
| 动态 | 事物 | 接受 |
| 关注/粉丝 | 角色 | 接受 |
| 链接 | 事物🡪属性 | 拒绝 |
| 收藏 | 事件🡪属性 | 接受🡪拒绝 |
| 账号 | 事物 | 接受🡪拒绝 |
| 凭证 | 事物 | 拒绝 |
| 昵称 | 属性 | 拒绝 |
| 密码 | 属性 | 拒绝 |

在反复推敲后，我们确认了本应用的分析类：用户、关注、粉丝、新闻、评论、动态等。构建出如下分析类图：



* 接下来我们对于每个分析类做了详细的介绍：

#### 3.4.2.1 用户

属性：用户ID，昵称，密码。

操作：输入昵称和密码，进入页面，发布，输入搜索信息，管理频道，关注，取消关注，拉黑，收藏，取消收藏。

说明：用户的基本信息，以及一系列基本活动。被关注与粉丝继承，是最基础的角色类。用户的所有属性都是持久的，它存储了注册用户的基本信息。

#### 3.4.2.2 关注

属性：用户ID，昵称，密码。

说明：用户的一种，关注了之后便成为用户的关注。

#### 3.4.2.3 粉丝

属性：用户ID，昵称，密码。

说明：用户的一种，关注了用户之后便成为用户的粉丝。

#### 3.4.2.4 新闻

属性：新闻ID，新闻标题，新闻发布时间，频道，作者，新闻URL，图片地址。

说明：新闻的基本信息。新闻类中所有属性都是持久化的，它负责存储新闻的基本信息。可能要与系统进行一定的交互。

#### 3.4.2.5 新闻评论

属性：评论ID，新闻ID，作者ID，作者昵称，发布时间，发布内容。

说明：用户对于某一新闻的评论，当然，它也是一条动态。用户对于新闻的评论会自动作为一条动态发表。可能要与系统进行一定的交互。

#### 3.4.2.6 动态

属性：动态ID，作者ID，作者昵称，发布时间，发布内容。

说明：用户发表的一条动态，用来分享自己的一些看法与总结，并不一定与新闻时事相关。可能要与系统进行一定的交互。

#### 3.4.2.7 动态评论

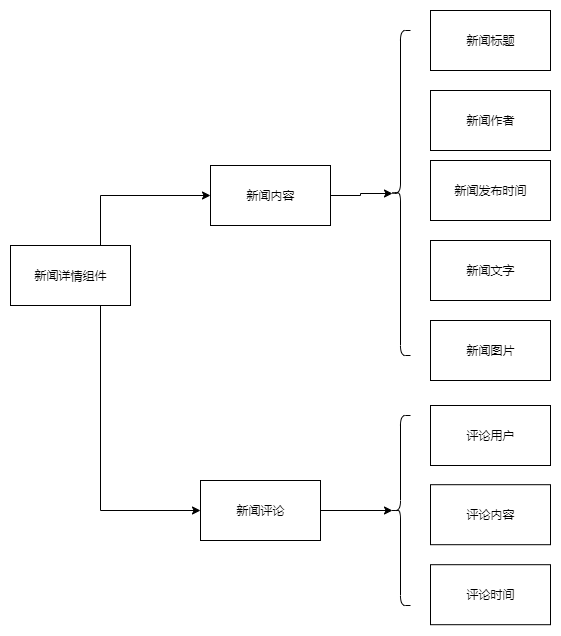
属性：评论ID，动态ID，作者ID，作者昵称，发布时间，发布内容。

说明：用户对于某一动态的评论。这与新闻评论不同，动态的评论就只是评论而已。

* 接下来我们将对系统部分组件进行建模：

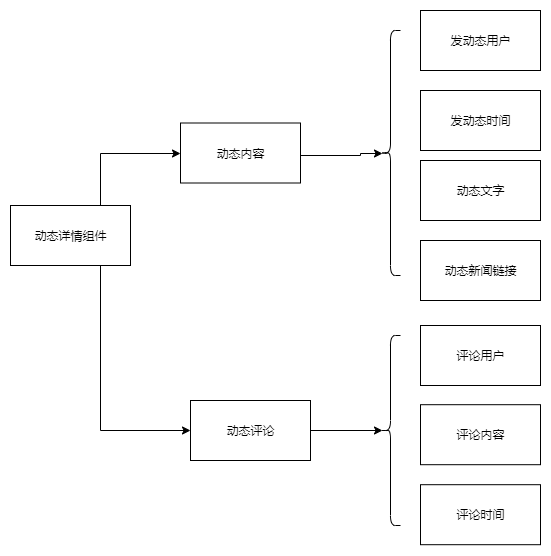
#### 3.4.2.9 新闻组件图

对于新闻组件的内容，我们做了如下建模：



#### 3.4.2.10 动态组件图

对于动态组件的内容，我们做了如下建模：

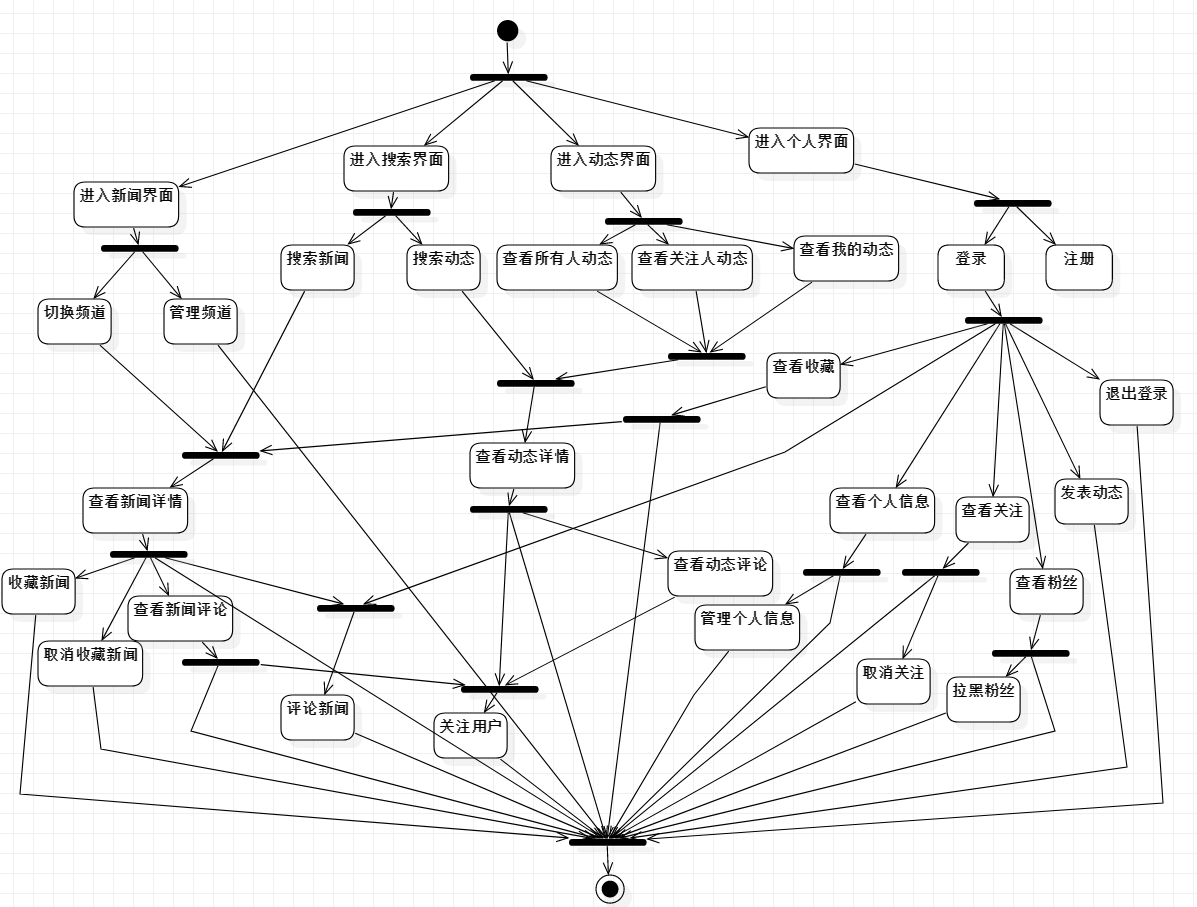


### 3.4.2 功能建模

基于用例图，对于每一个用例都有一些步骤来实现它。对于这些步骤，我们抽象出一些用例的活动图，对应用的功能进行建模。

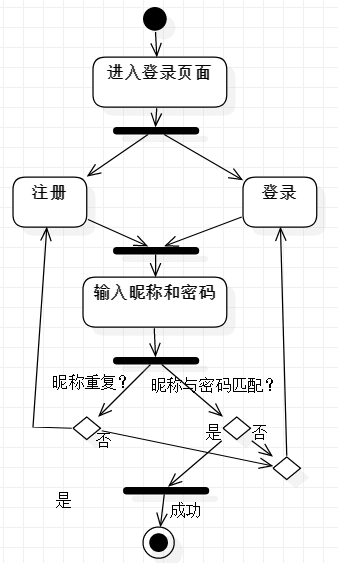
#### 3.4.1.1 系统级活动图

从进入app开始，用户可以进行一系列活动。

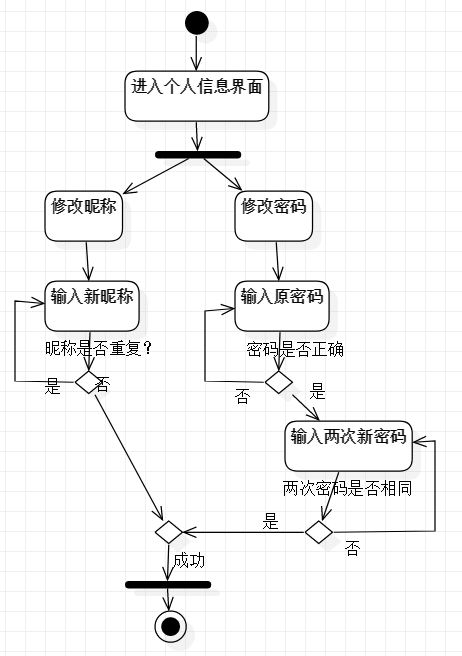
4

#### 3.4.1.2 账户管理模块

注册/登录：

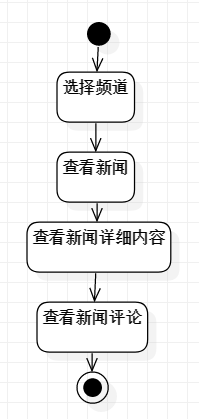


管理个人信息：

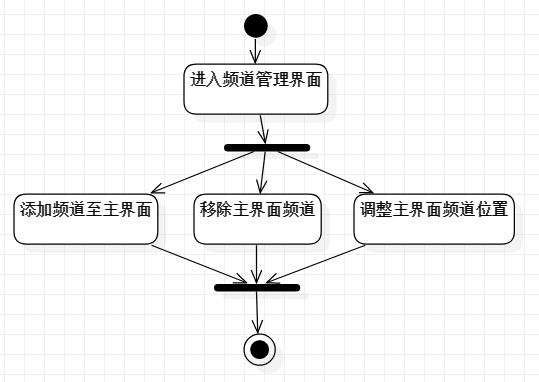


#### 3.4.1.3 浏览模块

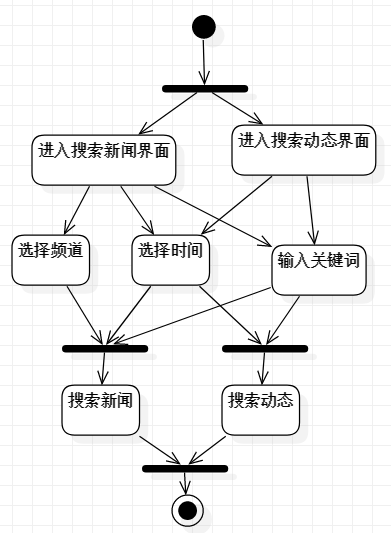
查看新闻/切换频道：



频道管理：

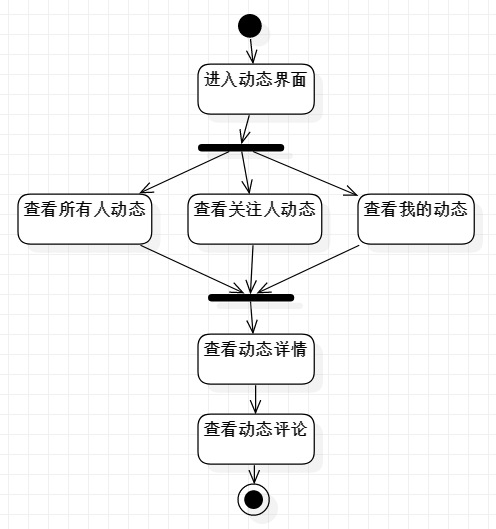


搜索新闻/动态（不可以按照好友搜索动态）：

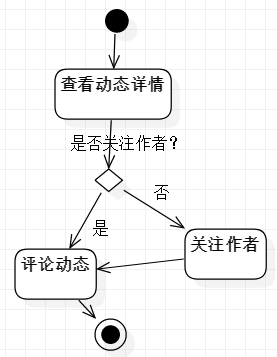


#### 3.4.1.4 社交模块

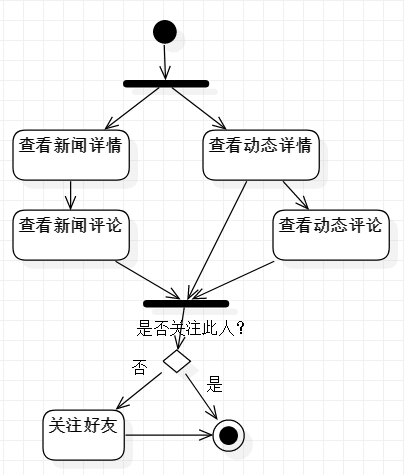
查看动态：



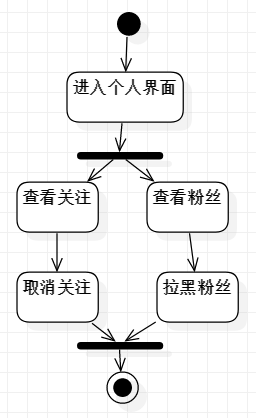
评论动态：



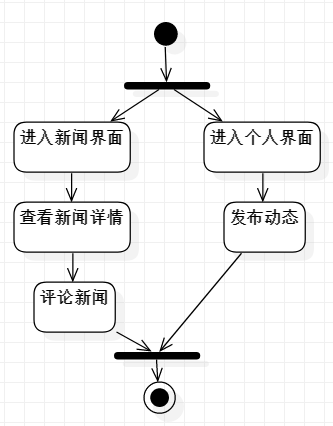
关注好友：



取消关注好友/拉黑粉丝：

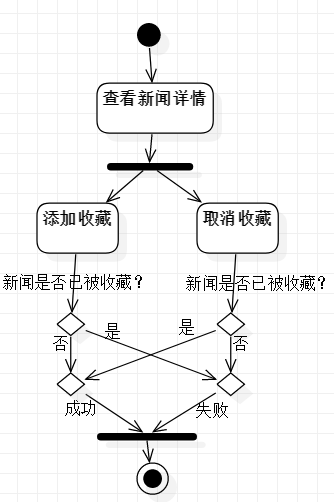


发布动态（评论新闻）：

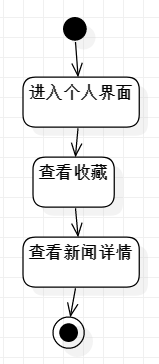


#### 3.4.1.5 个人收藏模块

收藏新闻/取消收藏新闻：



查看收藏：

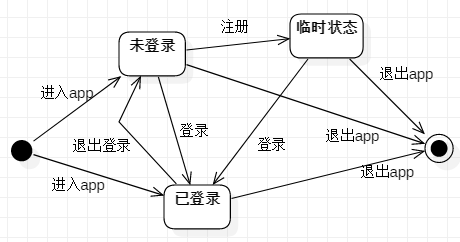


### 3.4.3 交互建模

* 根据用例图和分析类图，我们抽象出一些分析类的状态图和一些用例图的顺序图，用来进一步对于应用的行为进行建模。

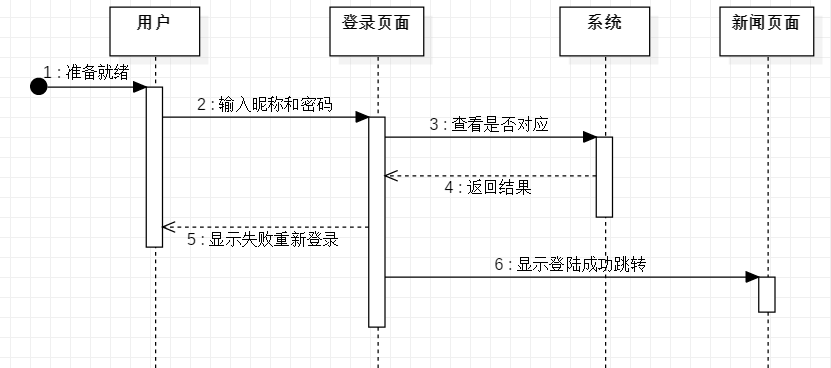
#### 3.4.3.1 用户分析类状态图：

这是用户分析类的状态图，用户有登录，未登录等状态，使用不同的方法后，状态发生变化。



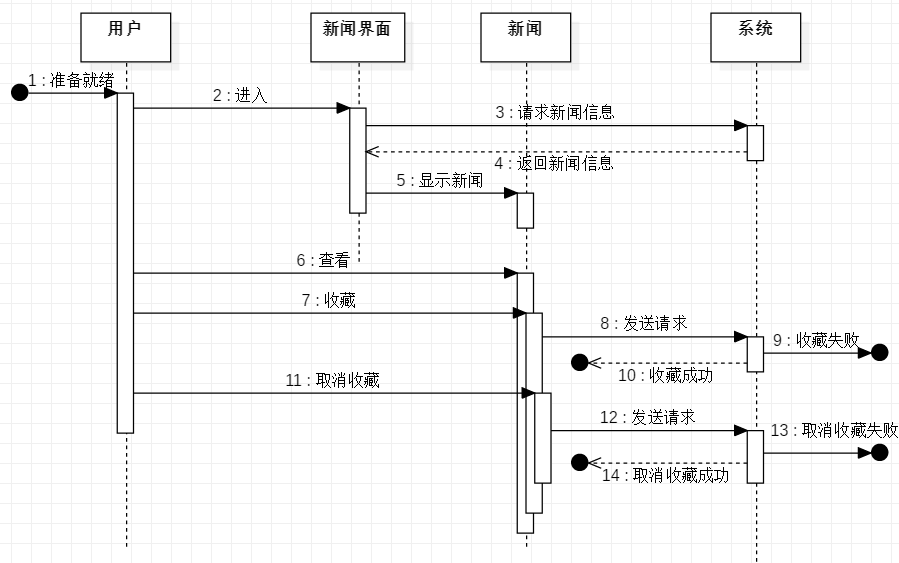
#### 3.4.3.2 注册/登录用例时序图：

这是注册/登录的时序图，展示了用户注册登录的全过程。



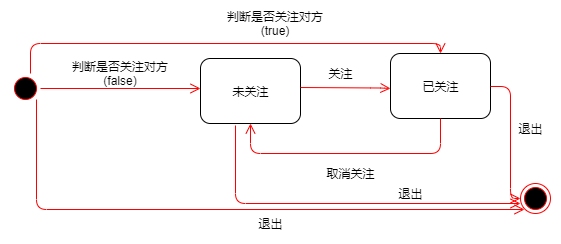
#### 3.4.3.3 收藏/取消收藏用例时序图

这是收藏/取消收藏的时序图，展示了如何在新闻详情界面收藏一条未收藏的新闻或者取消收藏一条已经收藏的新闻。



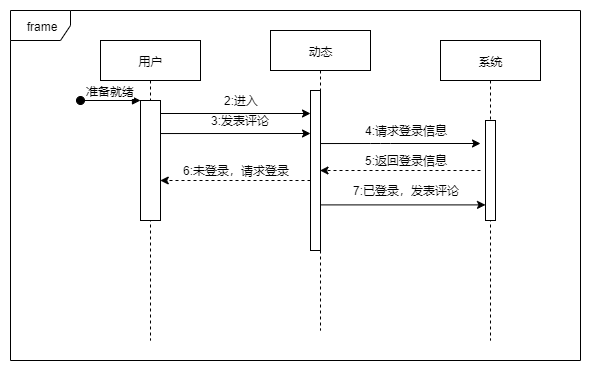
#### 3.4.3.4 关注分析类的状态图

这是关注的状态图，展示了一个关注好友的状态及其转换。



#### 3.4.3.5 评论动态用例时序图

这是评论动态的时序图，展示了如何在动态详细界面发表评论。

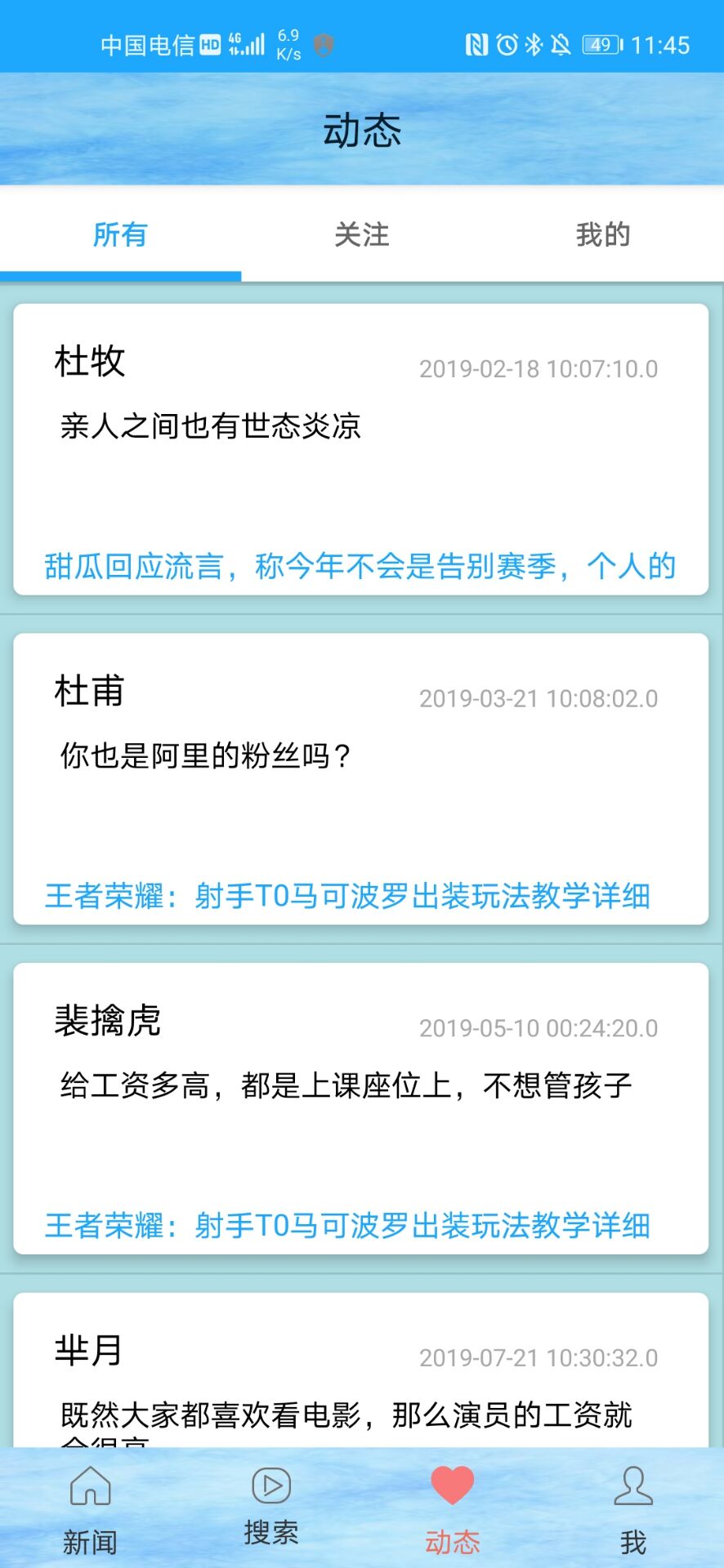


* 同时，对于四个主界面，我们做了界面原型设计

#### 3.4.3.6 新闻主界面



#### 3.4.3.7 动态主界面



#### 3.4.3.8 搜索主界面



#### 3.4.3.9 我的主界面



### 3.4.4配置建模

#### 3.4.4.1 客户端配置

|  |  |
| --- | --- |
| **org.gradle.jvmargs** | **-Xmx1536m** |
| android.useAndroidX | false |
| android.enableJetifier | false |

#### 3.4.4.2服务端配置

spring:

​ datasource:

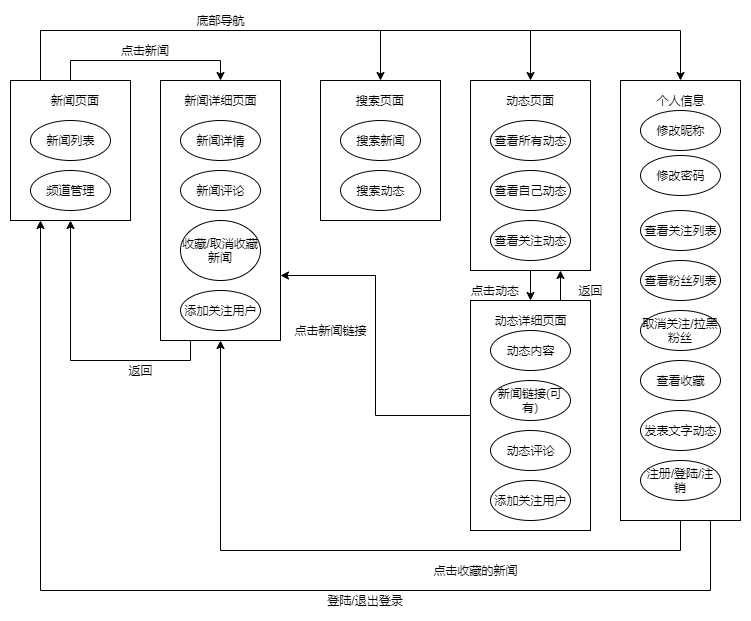
|  |  |
| --- | --- |
| **driver-class-name** | **com.mysql.cj.jdbc.Driver** |
| url | jdbc:mysql://47.100.79.111:3396/newsApp |
| username | root |
| password | \* |

server:

​ port: 8080

### 3.4.5 导航建模

我们为本APP的主要几个页面进行导航建模，明确他们之间的跳转关系，为用户使用某些功能提供便利。



# 4. 非功能需求

## 4.1 性能要求

### 4.1.1 用户体验

本应用是新闻类APP，希望能够获取到最新的新闻，并可以读取新闻的具体内容。对实时性要求比较高，同时在获取新闻时要尽可能快。同时，此APP还要在动态方面做到实时刷新，以便用户在第一时刻了解好友动态。

### 4.1.2 时间特性要求

1. 响应时间：

在网络状况良好的情况下，对于页面的加载，响应时间不超过1s。在并发用户少于100人时，各种异步访问加载的响应时间也应小于1s。

2. 更新处理时间

数据库的更新涉及网络数据传输，与网络传输速度有关。但正常情况下，数据库更新 时间不应超过 1s，网络状况较差的情况下，不应超过 4s。数据库也是异步更新，不影响主进程。

3. 数据转换时间

对于文字的加载，数据转换较快，不应大于1s。对于图片的加载，数据转换速度一般，在网络情况较好时，不应超过2s。对于网页的加载就会更加耗时，在网络情况较好时，不应超过3s。

### 4.1.3 输入输出要求

1. 新闻详情

2. 视频详情

3. 朋友圈详情

4. 个人详情

## 4.2 数据管理能力要求

### 4.2.1 常量约定

1. 新闻界面新闻 <= 100条

2. 动态界面动态 <= 100条

3. 动态字数 <= 100 字

3. 新闻详情、动态详情评论数无常量约定

### 4.2.2 数据存储要求

本应用采用MySQL数据库进行数据的存储，对于应对比较复杂的查询具有优势，但是负载能力不强，为了应对高负载状态台下的运行，本系统数据存储要求如下：

* 数据库采用主从复制，读写分离的方式应对读频率远高于写频率的数据单独存储情况。
* 对于系统中高并发、访问频繁的数据，采用缓存数据库对MySQL进行隔离，防止其崩溃。
* 数据库的备份采用单服务器集中备份的方法，保证数据的安全性和可靠性。

### 4.2.3 负载

## 4.3 安全及保密性要求

### 4.3.1 软件使用数据

本应用存储的数据并没有用户的一些真实信息，如手机号、身份证号、银行号、真实姓名等等。所以用户在使用软件时，一定要注意保护隐私，不要随意泄漏私密信息，保护自身安全。对于评论、好友、粉丝、收藏等信息则存储在

### 4.3.2 用户账号数据

本APP只对用户在本系统的用户名、密码等进行保存，个人信息仅与本APP功能相关，不会对用户隐私造成任何不利影响。

## 4.4 灵活性要求

1. 项目采用SCRUM模型，一般为2周一个sprint，每周二开例会，总结本周工作进度，并进行头脑风暴提出意见完善系统。

2. 项目使用Leagoo作为项目过程管理工具，定期更新每个sprint的具体任务，并由任务负责人对已完成项目的状态进行更改。

3. 使用Git进行版本控制，采取前后端完全分离的政策，前端一个库，后端一个库，文档一个库，分工明确，灵活度高。

4. 软件采用MVC模式进行设计，界面和逻辑分离，代码维护、扩展变得容易，大大提高了灵活性。

5. 项目采用原型开发，迅速策划并迭代建模，灵活性好。

## 4.5 其他要求

### 4.5.1 可维护性

可维护性是指在不影响系统其他部分的情况下修改现有系统功能中问题或缺陷的能力。开发人员创建和设计系统架构时，为了提高系统的可维护性，必须考虑以下几个方面的要素：低 耦合、高内聚合系统文档记录。本系统将采用严格的软件工程的规范进行开发，并采用良好的设计模式保证系统各模块之间的低耦合及模块之间的高内聚。

本系统的所有代码将会被详细注释，对于系统所有代码，我们会生成详尽的技术文档。对于系统开发过程可能出现的报错，我们将以文档的方式详细罗列报错码及对应的报错信息。

### 4.5.2 灾难恢复

本应用为服务器-客户端模式的APP应用，数据一部分保存在客户端（本地），但大多重要的信息都保存在服务端。当客户端发生异常闪退或者突然断电等事件，重新进入应用时，可以向服务端重新请求这些数据，完成灾难恢复。

### 4.5.3 法律限制

#### 4.5.3.1 合法内容限制

用户在使用本应用提供的服务过程中，必须遵循国家的相关法律法规，不得利用本平台 发布危害国家安全、色情、暴力、凶杀、恐怖等非法内容；不得利用本平台发布含有虚假、有 害、胁迫、侵害他人隐私、骚扰、侵害、中伤、粗俗、或其它违背道德的内容。

用户使用本APP的行为若有任何违反上述条款的情形时，本应用有权直接删除该等违反条 款的内容。

#### 4.5.3.2 责任限制

互联网是一个开放平台，用户在此APP上发表的评论、动态有可能会被其他组织或个人 复制、转载、擅改或做其它非法用途，用户必须充分意识此类风险的存在。用户明确同意其使 用本应用服务所存在的风险将完全由其自己承担；因其使用本应用服务而产生的一切后果也由 其自己承担，本应用对用户不承担任何责任。