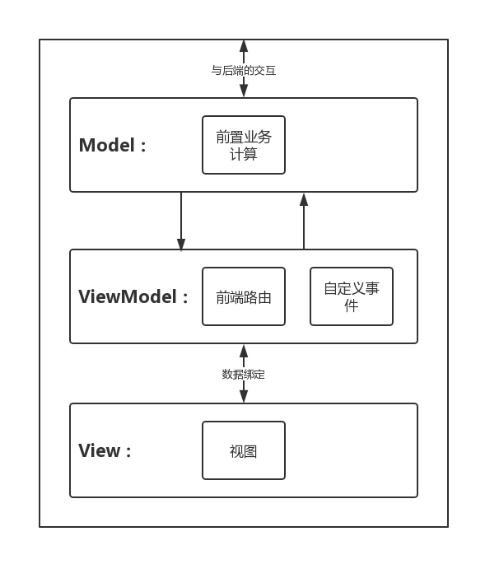
**Web前端架构**

前端主要采用MVVM（Model-View-ViewModel）架构，分离视图和模型，将其中的View 的状态和行为抽象化，与业务逻辑分开。Web前端设计为SPA（Single Page web Application ，单页web应用），总共包括普通用户和系统管理员两个场景的SPA，均需满足MVVM架构。



Web前端架构图

1. SPA

单页web应用只加载单个HTML页面，并在用户与应用程序交互时动态更新该页面的Web应用程序。浏览器一开始会加载必需的HTML、CSS和JavaScript，所有的操作都在这张页面上完成，都由JavaScript来控制。随着用户点击链接，SPA无需进行页面跳转，而是根据目标的url，计算需要显示和隐藏的元素。在SPA中，点击链接后浏览器无需向服务器请求新的页面文件，只需要向服务器请求必要的数据。这样可以使得整体的请求更少，更快的响应用户的操作。

SPA需要前端路由的支持，前端路由框架采用vue-router；同时，采用HTML5的history API进行url控制，以完成相应用户请求功能。

1. MVVM

MVVM架构包括模型层(Model)、视图层(View)、视图模型层(ViewModel)，利用 数据绑定，通过视图层实现GUI部分的代码隐藏。

模型层采用vue-mc实现，负责通过ajax与web后端部分进行数据传输交互。模型层将数据操作进行封装，将视图层、视图模型层与后端隔离开，便于web后端的改动和升级，也更有效地进行访问控制和缓存控制。

视图层和视图模型层采用vue.js实现。视图层由HTML5、JavaScript、CSS组成，实现可复用视图，完成主要GUI实现。 视图模型层连接模型层和视图层，与视图层显示的数据进行绑定。试图模型是存在公共属性和命令的视图的抽象。

1. 组件分离

实现过程中，以组件为单位，将每个内部耦合的组件合并在一起，而不同组件分离以避免互相影响。即一个组件（.vue单文件组件）包括它的HTML结构，JavaScript数据行为，和对应的CSS样式。

1. 组件间通信

虽然采用vue.js组件分离视图层和视图模型后，不同组件间耦合度较低，但是组件间仍存在通信需求。通信过程采用自定义事件的方式，自定义通信的数据格式，完成组件之间的的通信，以保持组件的独立性。

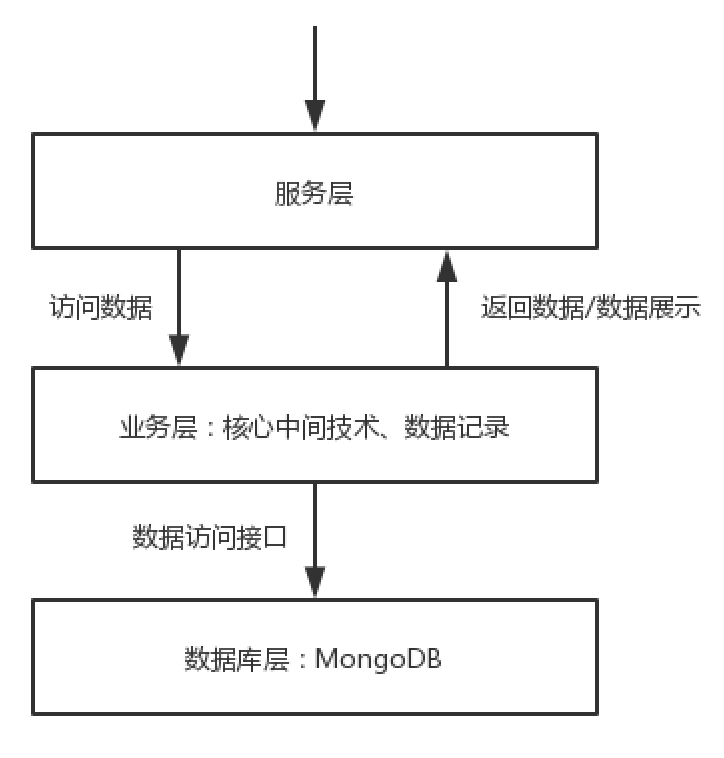
**系统后端架构**

后端使用较为简单的三层架构模式，基本上分为服务层，业务层和数据库层三个层面。后端采用flask框架，使用Python来完成和前端vue.js的交互。数据库方面选择了应用范围较广的MongoDB。

服务层主要负责前后端数据交互，前端的访问数据通过服务层的解析来传递给接下来的业务层，同时也将业务层的返回数据经过转化后交付给前端。

业务层主要目的是将服务层和数据库层分隔开，向上提供数据展示的服务，向下提供数据访问的接口。在业务层还包括了一些核心的中间技术，如第三方的开源组件等。另外，日志之类的数据记录也通过业务层进行处理。

数据库层使用MongoDB数据库，完成对数据库的各种操作，主要目的是封装一系列数据库操作，保证并发控制和各种事务处理等。



系统后端架构图

**子系统架构**

1. 数据更新系统

数据更新系统每日定期向合法数据源获取每日科研成果更新信息，以获取论文为主体，丰富系统数据信息，以便提供最新科研成果展示给用户。

数据更新系统以Python完成主要爬虫工作，获取科研成果的关键信息，包括题目、作者、关键字、主题等信息。

1. 日志系统

日志系统从消息队列中获取信息传递内容，包括各子系统信息、调试信息、用户浏览信息，便于管理员和维护人员了解系统状况，有利于跟踪调试、运行状态记录、系统崩溃恢复。

日志内容以时间和内容为划分，存为文本格式并持久化到数据库中。由于日志系统需要记录系统操作记录，数据量较大，需要根据时间划分数据表，便于数据库存储、管理、查询。考虑到项目规模，目前决定以星期为单位创建日志表。

1. 数据分析子系统

数据分析子系统主要作用是给平台管理员展示后台日志子系统的结果以及数据库的分析。目的是让管理人员更好更加直观地了解平台的各项数据（如访问量，下载量，用户数等）。

数据分析子系统作为一个辅助后台管理的子系统，不需要像前端那样注重于UI等展示方式，因为本身使用者仅仅只有管理员。因此可以使用较为简单的设计模式，专注于数据的处理和展示即可。不过需要注意的是在数据分析子系统访问后台数据的同时不能对其他用户的访问造成影响。

1. 推荐列表子系统

根据用户的搜索记录等信息进行每日推荐，推荐用户可能感兴趣的文献列表。

用户感兴趣的文献列表是在前端的显示内容，但是一系列推荐信息的计算显然不能在前端完成，因此会通过本子系统的计算之后，由后端交付给前端进行结果的展示。

1. 学者认证子系统

学者的身份认证有很大一部分需要交给人工审核，因此需要一个独立的子系统来完成从申请到审核最后批复的流程。

用户申请学者身份认证大致分为8个步骤：

1. 用户申请学者身份认证
2. 系统返回认证页面
3. 用户填写认证信息并提交
4. 系统生成认证申请并交由平台管理员审核
5. 平台管理员审核申请
6. 认证申请通过（或者不通过然后就此结束）
7. 系统修改用户信息权限
8. 8. 学者身份认证成功

因此，本子系统需要统计各项审核信息并且做初步筛选，然后将剩余的完整信息提取出必要内容，交给审核人员进行处理。