快享讲座管理系统

软件设计规格说明

Team of Quick-share

2016/4/25

修订记录：

版本号	修订人	修订日期	修订描述
v1.0	许瑞嘉、江剑锋	2016/4/25	初始草案，概要式软件设计说明
v1.1	廖卢神 黄钦开 许瑞嘉	2016/7/2	完善文档，进行详细设计说明

一、 引言

1. 目的

该软件设计规格说明文档旨在根据软件需求规格说明书中的任务设计出一套可执行软件的结构模型。在本次初始草案中将概要式给出软件总体架构、技术选型及理由等。在后续版本中将进一步明确模块划分、数据库表设计、设计模式、性能分析等。该文档将指导软件的实际开发阶段，也为项目的可扩展性与可维护性提供重要依据。

2. 文档约定

该文档基本沿用 IEEE 1016 标准-软件设计规格说明模板(SDS)，其中对部分章节的内容进行了调整。

以下是对该文档所涉及术语的定义：

术语	定义
vue.js	一个构建数据驱动的 Web 界面的库，通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。
python	一种面向对象的解释型计算机程序设计语言，在设计中注重代码的可读性，同时也是一种功能强大的通用型语言。
javascript	一种弱类型的动态脚本语言，支持多种编程范式，包括面向对象和函数式编程。

3. 参考文献

IEEE STANDARD 1016：Software Design Specification

二、 总体架构

1. 硬件架构

用户可以通过台式电脑、笔电、手机、平板连接 Web 服务器使用快享讲座管理应用程序。Web 服务器连接数据库以拉取文件、查询记录等。（措辞还不够具体严谨）

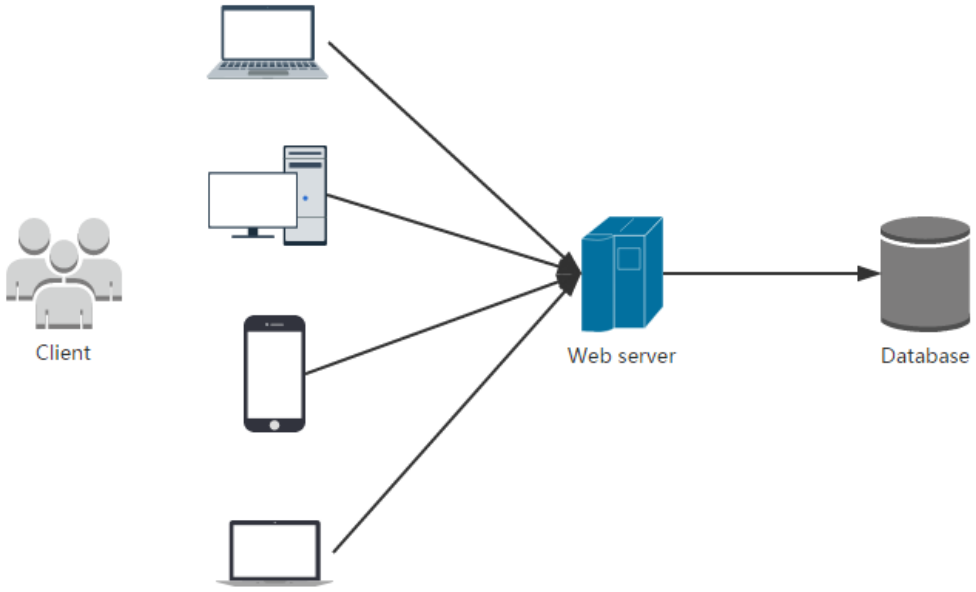


Figure 1 硬件架构

2. 软件架构

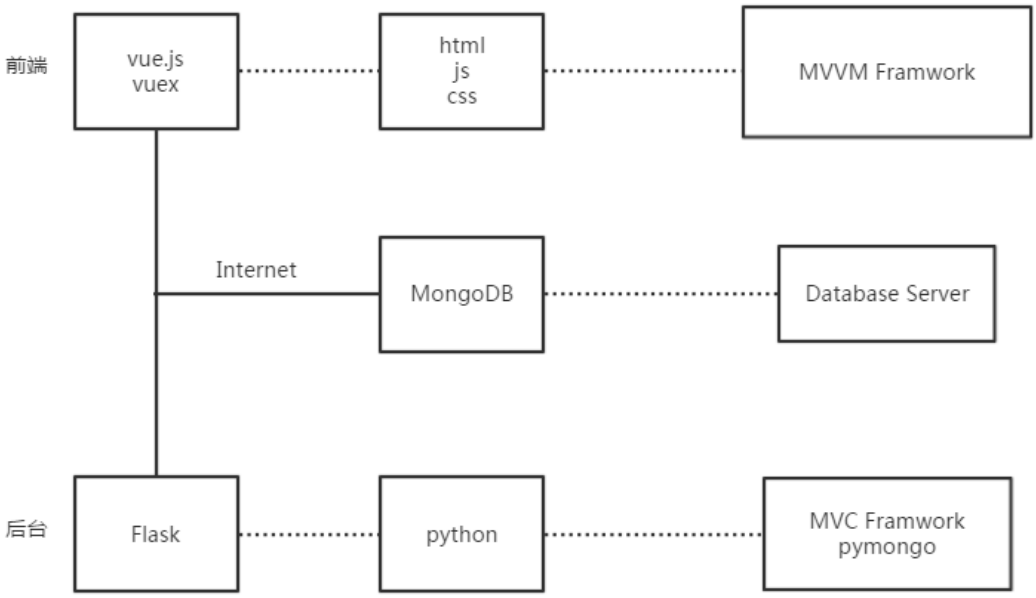
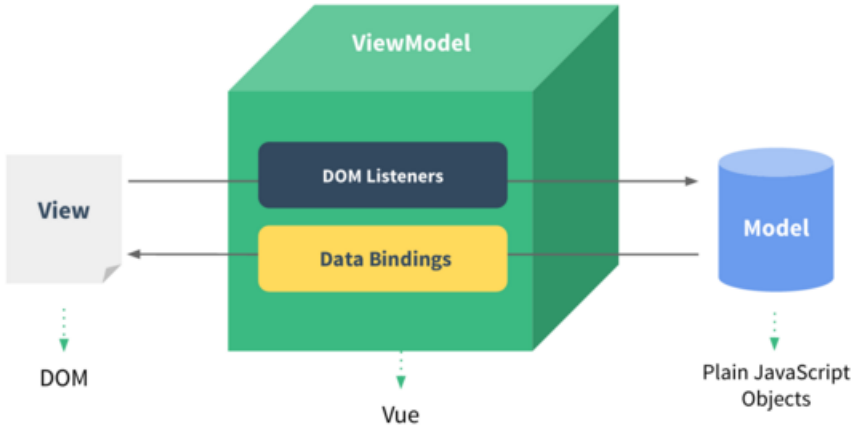


Figure 2 软件架构

三、 技术选型及理由

Vue.js



Vue.js 的核心是一个响应的数据绑定系统，它让数据与 DOM 保持同步非常简单。在使用 JQUERY 手工操作 DOM 时，我们的代码常常是命令式的、重复的与易错的。Vue.js 拥抱数据驱动的视图概念。通俗地讲，它意味着我们在普通 HTML 模板中使用特殊的语法将 DOM“绑定”到底层数据。一旦创建了绑定，DOM 将与数据保持同步。每当修改了数据, DOM 便相应地更新。这样我们应用中的逻辑就几乎都是直接修改数据了，不必与 DOM 更新搅在一起。这让我们的代码更容易撰写、理解与维护。

Flask

Flask 旨在保持代码简洁且易于扩展。Flask 本身不会做太多的选择，而它所做的选择往往易于更改。默认情况下，Flask 并不包含数据库抽象层，表单验证或者任何其它现有的库能够处理的。Flask 支持扩展，这些扩展能够添加功能到你的应用，像是 Flask 本身实现的一样。众多的扩展提供了数据库集成，表单验证，上传处理，多种开放的认证技术等功能。

MongoDB

MongoDB 是一个基于分布式文件存储的数据库。由 C++语言编写。旨在为 Web 应用提供可扩展的高性能数据存储解决方案。

快享讲座管理系统案例分析：

✧ 用户数据需要持久化存储，不能用内存数据库；

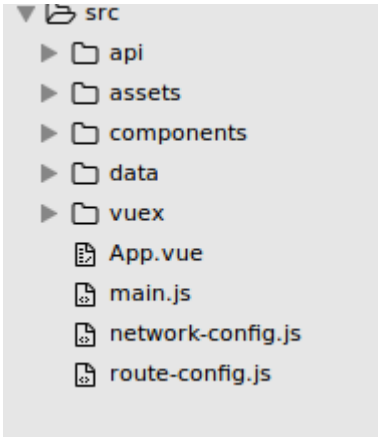
- ✧ 用户数据安全性不高，不要求事务性，可以使用 Mongo；
- ✧ 大概支持查询并发量 1000，mysql(以 2000 估计)与 mongo(以 5000 估计)均可使用；
- ✧ 业务较复杂，业务模型需要可扩展，mongo 修改更方便；
- ✧ 考虑未来支持分布式。

综合以上几点，我们最终使用 Mongo 作为数据库。

四、 架构设计

网页端：

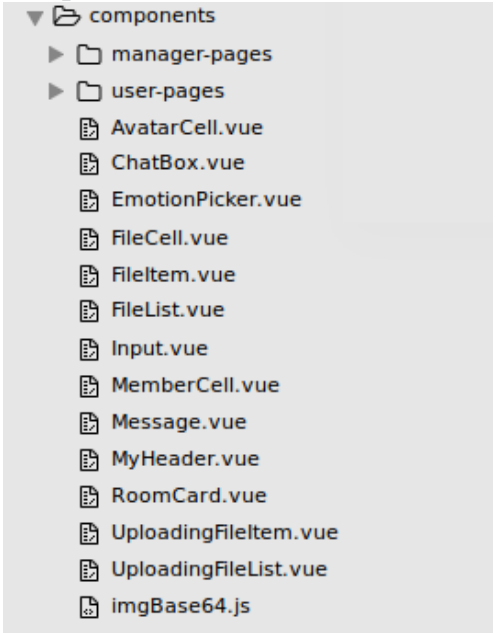
网页端的代码结构如下图：



api：代码文件主要是用于封装一些业务逻辑的 api，诸如登录验证之类的，同时还封装了基于 fetch 这一异步网络请求的 api 从而实现了代码里面复用

assert：放置一些图片相关的静态资源

components：放置了所有组件的组件代码，如下：

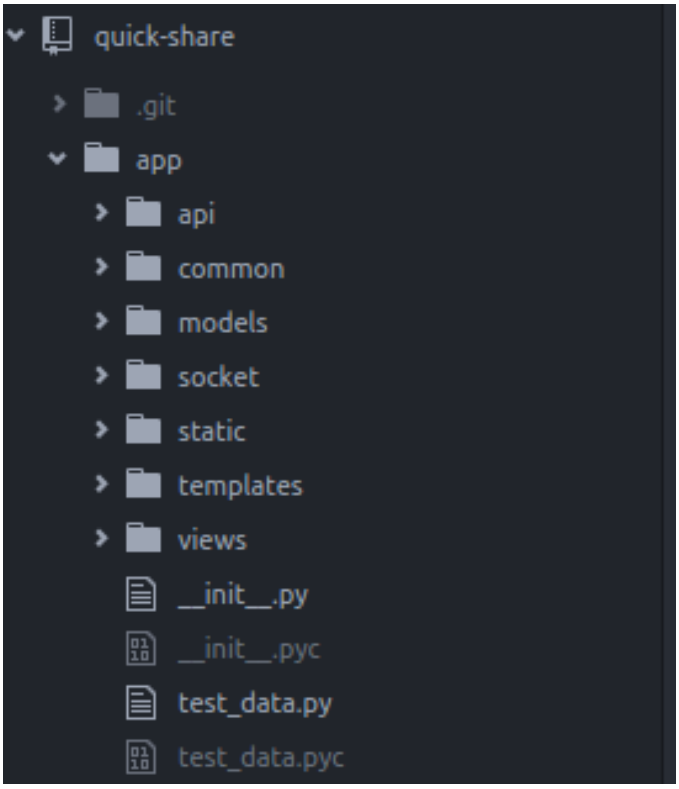


其中为了方便区分用户级别和管理员级别的代码将用户的页面组件都放于 user-pages，管理员的页面组件都放于 manager-pages;其余些可以共用的组件的都放于 components 目录下。

Vuex: 用于放置共享状态变量处理的代码文件

服务器端:

服务端的代码结构如下图:



属于典型的 MVC 结构，其中 models 里面放的是数据库的结构，static 和 templates 里面放的是静态页面。剩下的用于 controller 的内容比较庞大，我根据接口类型的不同，将 controller 的才分成 api，socket 和 views 三个部分。其中 api 返回的是 json 类型的数据，socket 返回的是 socket 类型的数据，views 则返回的主要是页面数据。

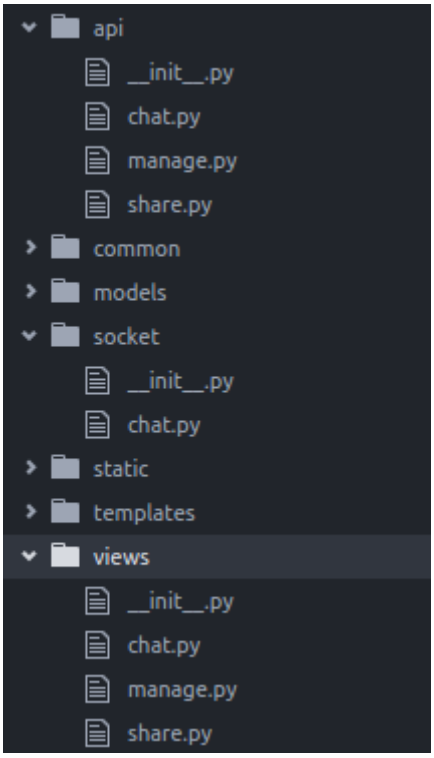
五、 模块划分

在整体的模块划分上，我采用的是 MVC 的这种竖直的结构，但是由于 MVC 结构的特性，controller 的部分难免变得比较臃肿，为了能够减少 controller 部分的臃肿的状态，我将不同类型的结构分门别类，在各个类型的接口中，还是按照各个业务模块继续分类。

通过采用这样一种扁平化分类的方法，降低了 controller 部分的臃肿。

在各个接口类型中，我根据每个类型的业务需求，把他们各个独立分类，虽然可能会有些许冗余，但是确实便于了代码的管理。

业务模块如下：



六、 软件设计技术

网页端：

使用的主要技术如下：

Vue：用于构建交互式的 Web 界面的库。它提供了 MVVM 数据绑定和一个可组合的组件系统，具有简单、灵活的 API。

Socket.io：将 Websocket 和轮询(Polling)机制以及其它的实时通信方式封装成了通用的接口，极大方便了实时通讯的编程

Vuex：Vuex 是一个专门为 Vue.js 应用所设计的集中式状态管理架构,使用它可以更好地组织代码，以及把应用内的的状态保持在可维护、可理解的状态。

服务器端：

使用的主要技术如下：

Flask：Python 的微型框架，简单易用，有非常优雅的路由函数编写方式，同时可以非常便捷的使用插件，也有数量丰富的插件库。

Socket_IO：支持多种 socket 引擎，极大方便 socket 的程序开发，但是也有调试不便等缺点。

Sqlalchemy：数据库访问插件，能够直接将数据库表自动转换成 python 数据结构。

Redis：缓存数据库，能够方便的存储一些临时数据于内存中，加快了数据访问，同时也减小了真正数据库的存储负担，但是也加大了数据同步和保存的负担。

附：UML 图

领域模型