# 开发文档

这份文档将记录我都做了些什么…

目录

[开发文档 1](#_Toc23599477)

[总体目录（概览） 1](#_Toc23599478)

[1. 前端 1](#_Toc23599479)

[2. 后端 3](#_Toc23599480)

[3. 数据库 3](#_Toc23599481)

[功能详情 4](#_Toc23599482)

[后端 4](#_Toc23599483)

[目录结构说明： 4](#_Toc23599484)

[接口详情： 4](#_Toc23599485)

[后台图像处理接口 7](#_Toc23599486)

[数据持久化(mysql、Sequelize对象)： 7](#_Toc23599487)

[前端 8](#_Toc23599488)

[目录说明 8](#_Toc23599489)

[页面详情及实现： 9](#_Toc23599490)

[使用说明 12](#_Toc23599491)

# 总体目录（概览）

## 前端

前端采用vue-cli 3.0脚手架开发。

|  |  |
| --- | --- |
| 技术栈： |  |
| http拦截 | axios |
| 身份验证 | JSON Web Token (JWT) |
| 状态管理 | Vuex |
| UI界面 | Element-UI |

目前功能。

|  |  |
| --- | --- |
| 地址 | 描述 |
| /register | 注册页面 |
| /login | 登录页面 |
| /index | 主页 |
| /home | 欢迎页，由右侧导航访问 |
| /InfoShow | 显示个人信息 |
| /expaperlist | 试卷列表 |
| /mypaper | 创建答题卡 |
| /expapercreate | 创建试卷 |

## 后端

后端使用node.js开发。

|  |  |
| --- | --- |
| 技术栈： |  |
| 框架 | Express框架 |
| 身份验证 | Passport、passport-jwt、JSON Web Token (JWT)、bcryptjs |
| url中间件 | body-parser |
| UI界面 | Element-UI |
| 数据持久化 | Mysql2、sequelize |

接口

|  |  |  |
| --- | --- | --- |
| 已提供API列表 | Method |  |
| /api/getscript | GET | 获取试卷列表 |
| /api/del\_expaper | GET | 删除列表项 |
| /api/add\_expaper（post） | POST | 添加答题卡 |
| /api/add\_expaper（get） | GET | 添加答题卡 |
| /api/user/register | POST | 注册 |
| /api/user/login | POST | 登录，拿到token |

## 数据库

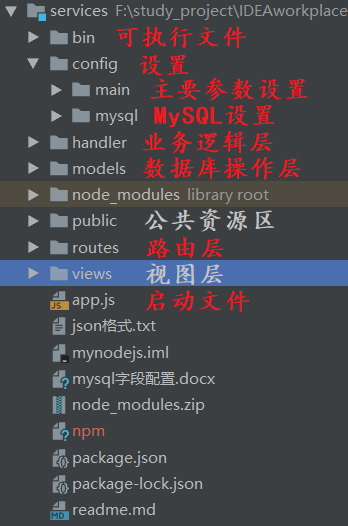
设计数据库字段。

设计expaper、user表

# 功能详情

## 后端

### 目录结构说明：



### 接口详情：

***整体处理流程为：***

业务层

Handler/

数据库交互层

Models/

路由层

Router/

主要负责把数据写入数据库，同步数据表。

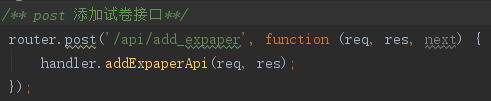
负责解析请求，文件IO，然后将需要写入的数据提交给数据库交互层

接收到请求之后将请求提交给业务层

#### 添加试卷接口（典例）

POST /api/add\_expaper（post）

***Router/expaper.js***



***Handler/expaperhandler.js***

使用了Promise规范。避免地狱回调





***\models\db\sqlhandler.js***

***依然使用promise***



其他接口也类似。

### 后台图像处理接口

**基于GraphicsMagick**

需要用到的技术栈：安装GraphicsMagick并重启电脑。

依赖包：gm 裁剪图片，connect-multiparty 处理请求的base64图片。

前端使用html2canvas发送图片。

具体操作流程为：接收前端图片储存到temp，切割temp图片到paperpoint。

**基于OpenVC图像识别：**

Node需要python2.7支持

//需要构建VC：<https://www.jianshu.com/p/b4d739b43a23>参考此文

NodeJS的OpenCV安装：https://github.com/peterbraden/node-opencv

### 数据持久化(mysql、Sequelize对象)：

Sequelize中间件

Sequelize对象

用户表

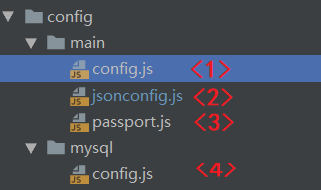
答题卡表

config设置，用户，密码，连接池…

***Models/db/db.js 包装Sequelize中间件***

***Models/db/tables 声明表，同步表。***

***Config/mysql 设置mysql用户密码连接池***

******

***<1> 存放设置app端口，数据存放路径等app相关设置。***

***<2>存放即将被写入的json数据和sql数据对象的格式。***

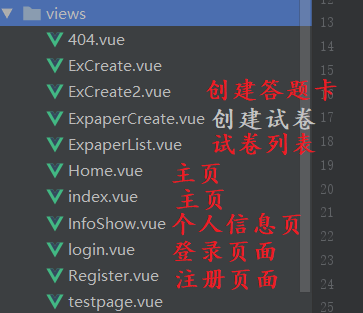
***<3>存放jwt相关设置***

***<4>mysql用户密码设置。***

## 前端

### 目录说明



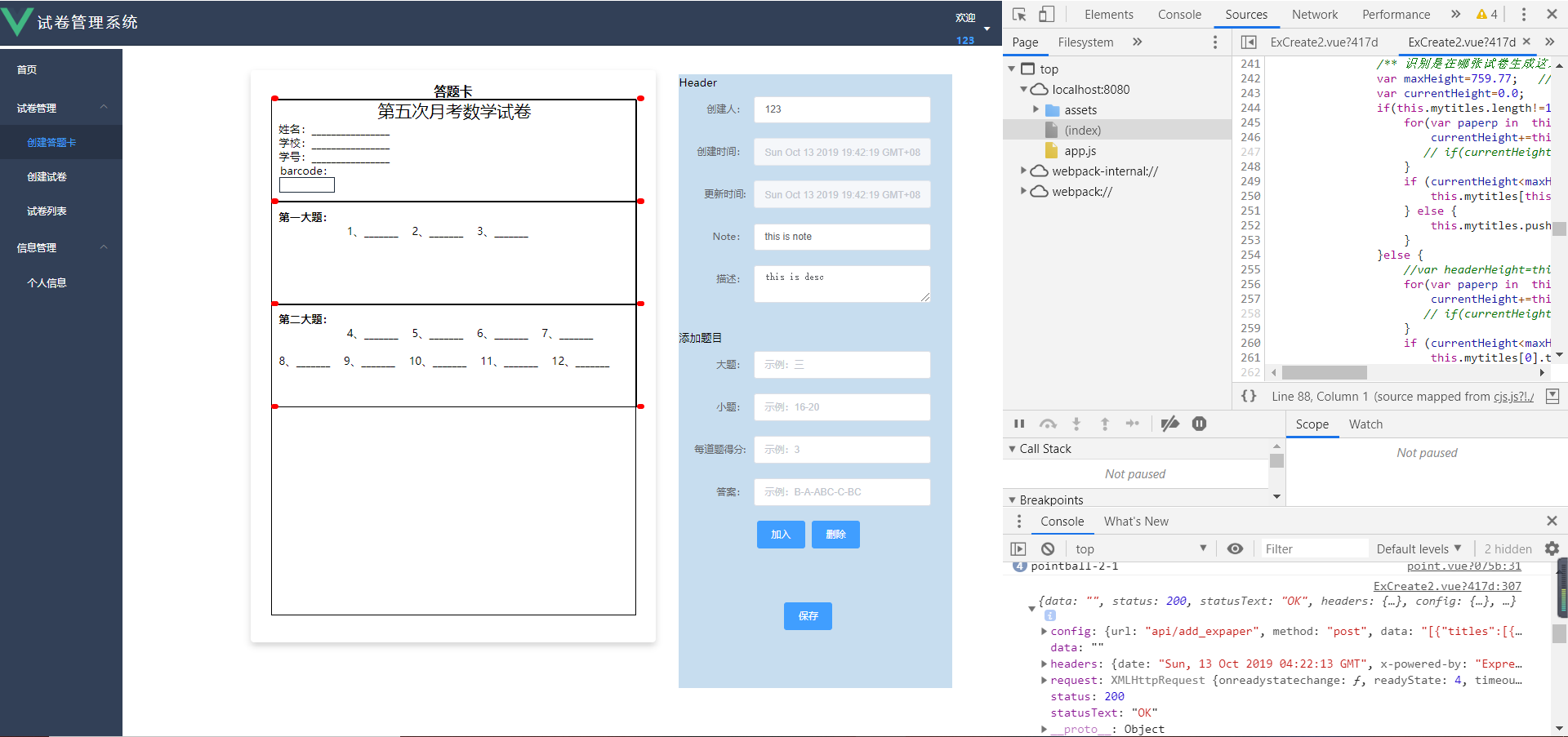


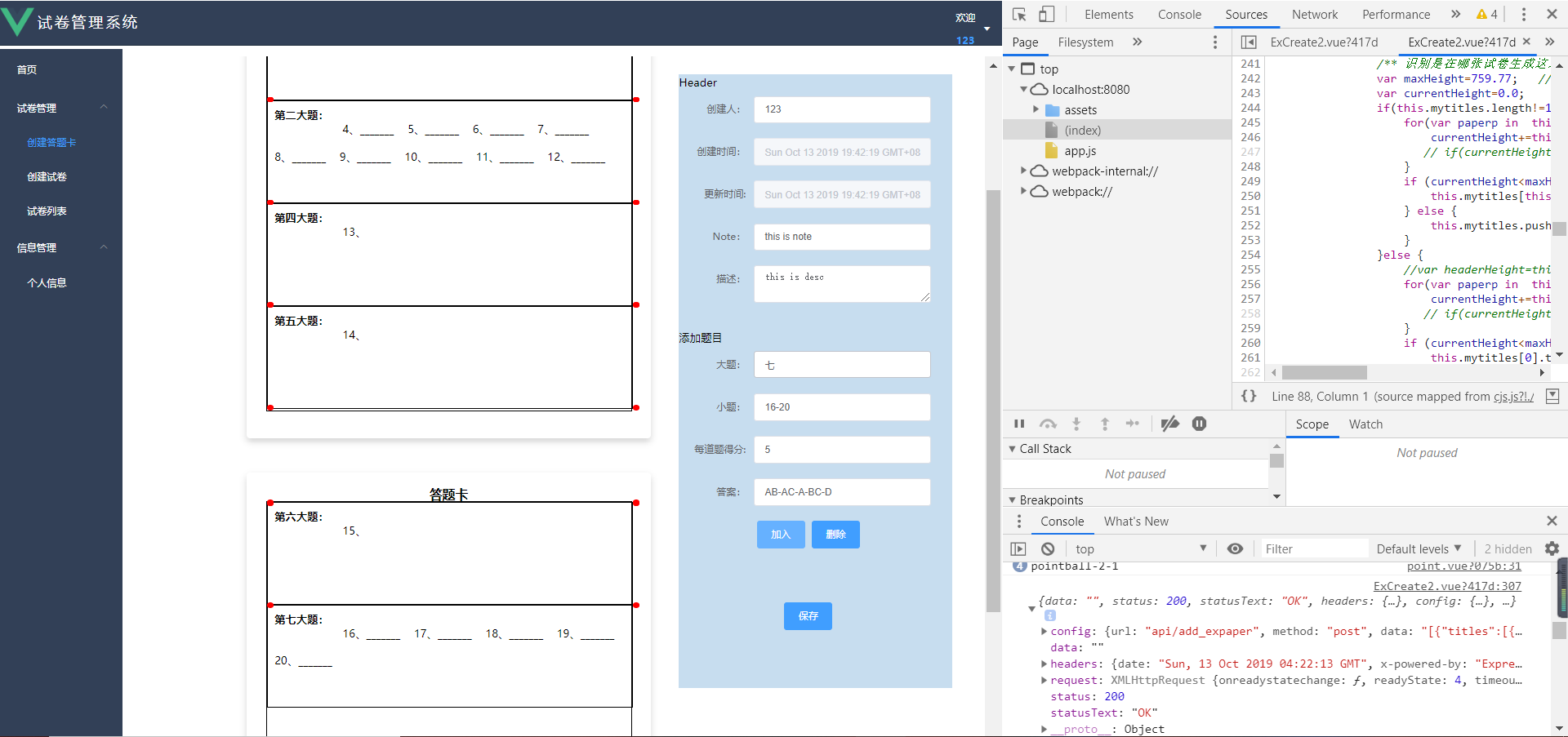
### 页面详情及实现：

#### /mypaper 创建答题卡页面

***源码路径：/views/excreate2.vue***

视图如下：





原理：将答题卡全部信息以json数组形式存放形式。添加试卷时向数组加入json。

***页面加载：***

1.当页面加载时，mounted()函数将token中获取的用户信息加入答题卡header，

2.开始循环数组加入A4试卷纸。

3.把数据交给子组件questions.vue，在试卷纸上循环数组加入题目，并且返回题目坐标

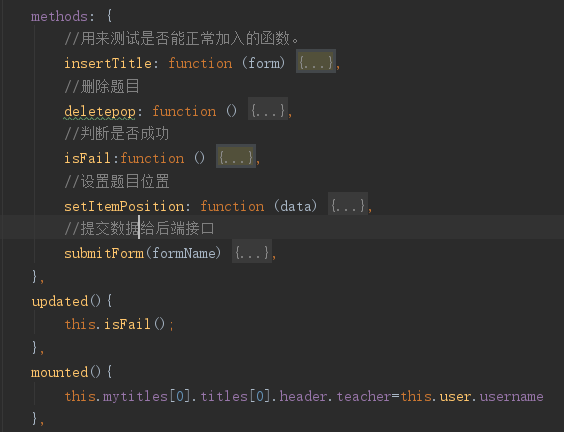
4.通过坐标为每一道题目进行定位。

5.updated()函数判断加入试卷的正确性来进行相关操作。

***“加入”操作（为答题卡添加题目）：***

1. 创建题目json数组。
2. 将数组append到data
3. 判断应该在哪张试卷的哪个位置生成题目div

***此vue实例方法：***





#### /login 登录页面

其他页面逻辑基本类似

# 使用说明

1. mysql：services/config/mysql/config.js里面配置用户密码，然后导入sql脚本storage/expaper.sql  
2. 后端：在services下，npm install然后启动或者配置idea环境 启动命令“node bin/www”  
3. 前端：在gui/expaper-client下，npm install 后启动命令“npm serve”