基于Vue的后台管理项目

**一、技术领域**

本项目属于web前端的页面开发领域，是一个基于Vue与elementUI等开源库搭建的脚手架，使用脚手架后可以快速搭建相关的后台管理页面与基础功能，提高开发效率。

1. **背景技术**

随着Web前端工程化与框架化的发展与普及，越来越多的项目采用前端后台分离、组件化与工程化方式的方式进行开发。传统的jquery操作DOM的开发模式性能不高，代码量大且不易复用。此项目正是基于Vue与elementUI等开源库，使用组件化、工程化的方式来开发单页面应用，使代码易复用、易扩展、提高开发效率与高用户体验。

**三、发明的目的及亮点**

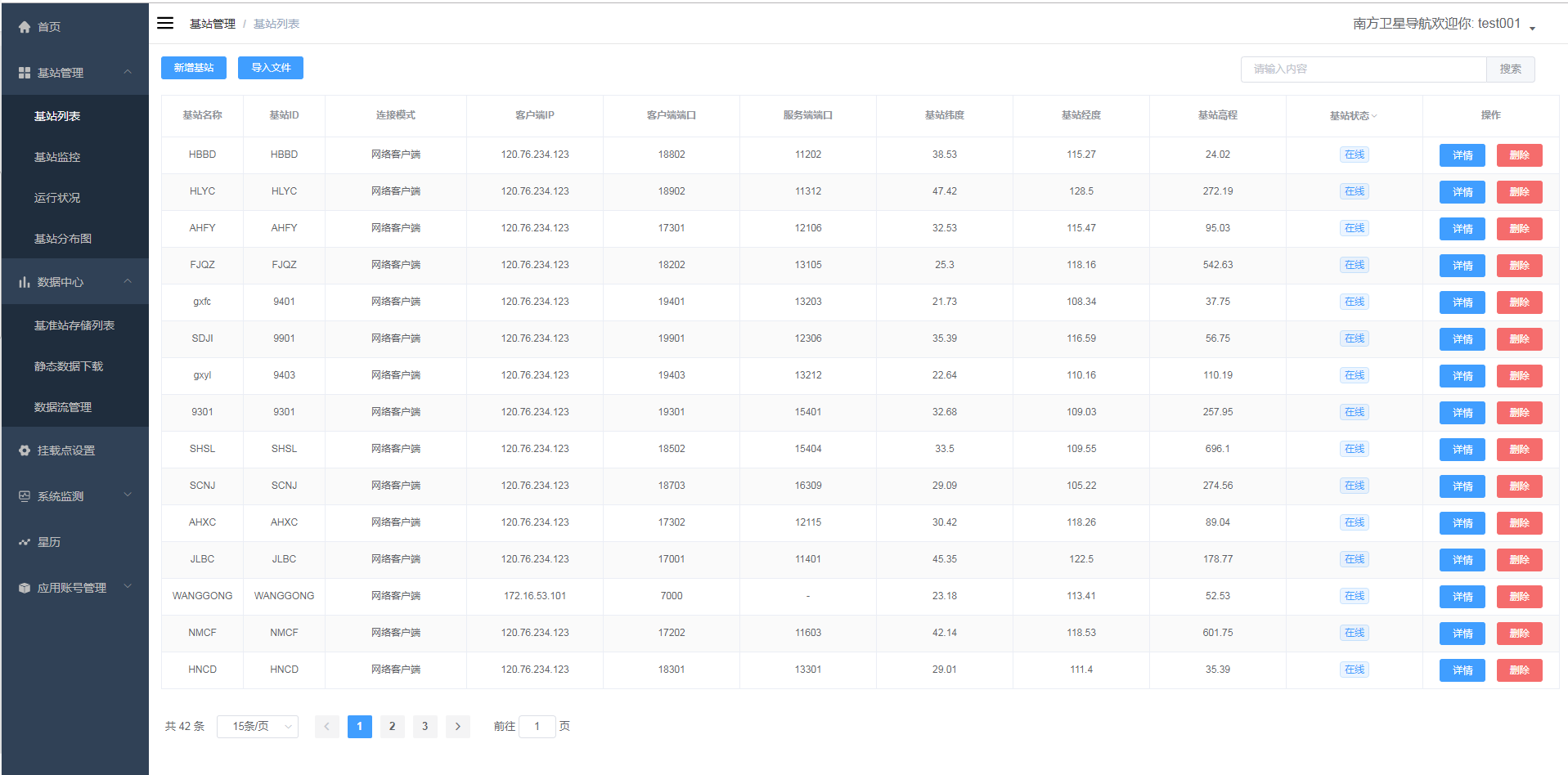
本项目旨在提高前端开发效率，优化前端开发架构，进而提高代码易复用性、易维护性、易扩展性等等。项目包括登录、后台管理等基本页面，封装动态路由、请求拦截器与权限验证判断等相关逻辑功能，并且提供基本的功能组件，能够开箱即用，只需根据约定的代码规范去添加业务逻辑与开发页面组件即可。

**四、技术方案**

使用此项目需要安装node等工具，技术栈基于[ES2015+](http://es6.ruanyifeng.com/" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)、[vue-cli](https://github.com/vuejs/vue-cli" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)、[vue](https://cn.vuejs.org/index.html" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)、[vuex](https://vuex.vuejs.org/zh-cn/" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)、[vue-router](https://router.vuejs.org/zh-cn/" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)、[axios](https://github.com/axios/axios" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)和[element-ui](https://github.com/ElemeFE/element" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/_blank)等。主要包含的内容有页面布局、路由与侧边栏、权限验证、与服务端进行交互、数据存储等等。

布局：

如下图所示，项目页面整体基础布局包含侧边栏、导航栏以及这主体页面的内容展示区等。侧边栏或者导航栏都不会随着主体页面变化而变化的。项目中的页面大都是基于这个布局的，除了个别页面如登录页面与404等页面没有使用该布局。如果想在一个项目中有多种不同的布局也是方便简单的，只要在路由配置中的一级路由设置里选择不同的布局组件即可完成不同布局。



路由和侧边栏：

路由和侧边栏是组织起一个后台应用的关键骨架，本项目的侧边栏和路由是绑定在一起的，所以只需要配置对应的路由，侧边栏就能够根据路由的配置数据动态地生成并显示出来。从而大大减轻了手动重复编辑侧边栏的工作量。当然这样设计就需要在配置路由的时候遵循一些约定的配置规则。

路由分为两种，即常规路由与异步路由。常规路由代表那些不需要动态判断权限的，一直会显示页面的路由，如登录页、404等通用页面的路由。异步路由则代表那些根据动态判断权限动态后而添加的页面路由，比如那些需要用户为管理员权限才能显示操作的页面。

侧边栏主要基于elementUI中的组件el-menu改造而来的，通过读取路由的配置与结合权限判断，进而动态生成的。因为侧边栏需要支持路由无限嵌套即多层菜单显示，所以项目中封装与使用到特定的递归组件，因此侧边栏会根据路由的配置显示菜单具有的层级，是否显示菜单等等。

权限验证：

本项目主要是搭建后台的管理页面，虽然其权限验证主要是针对前端页面的权限验证，但也是项目中的基础核心功能。不同的权限对应着不同的路由，侧边栏也会跟随着不同的权限异步动态生成。实现登录和权限验证的思路如下：

当用户填写完账号和密码登录向服务端验证成功后，服务端会返回一个有效期的token，前端会根据token调用服务端提供的权限接口来获取用户的详细权限或者信息等等。然后动态地根据用户对应的权限对比计算出其对应有权限的路由，并通过异步路由来动态挂载这些路由。

对于按钮级别粒度的权限控制，可以根据获取回来的权限操作列表用Vue中的v-if手动判断来区分不同权限对应的按钮即可。

服务端进行交互：

在本项目中，一个完整的前端界面操作交互到服务端处理流程是这样的：调用统一管理api接口的请求函数；使用封装的拦截器进行发送请求；获取服务端返回的数据；更新或者显示数据等。

拦截器是基本[axios](https://github.com/axios/axios" \t "https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/essentials/_blank)的封装的，它封了全局的请求拦截器、返回拦截器、统一的错误处理、请求超时等处理、请求url设置等等。其中请求拦截器便于统一处理post、get等的请求参数与请求头的信息，根据项目需求可以统一添加必要的参数如token，不必重复在各个请求中添加。返回拦截器只需要处理成功的结果进行逻辑加工，更新数据进行页面显示等，其它的情况则根据判断会自动统一处理，提示错误信息等等，不必针对各个请求的返回手动添加错误处理等，减少代码量并提高开发效率。

数据管理：

项目中的数据一般会存放在每个页面的组件中，当一些全局的数据如登录获取的token,用户权限等等则使用vuex来统一管理，这样会使用各个组件页面能够共享数据的状态，零散分布在各个组件页面的数据就可以方便读取或者进行修改设置，并且vuex能够将数据动态地反馈响应到各个页面组件中显示。如果配合HTML5的本地存储等方式，将vuex的数据state在localStorage、sessionStorage或其它存储方式中取值，就可以实现数据的持久化存储，避免数据在页面刷新后丢失。

最后，本项目的主要目的是搭建一个简单易用的后台管理页面的脚手架，开发者可以在脚手架上进二次开发，根据具体的业务逻辑开发对应的组件与页面，从提高开发效率，快速实现项目的功能。