**方案设计**

1、前端架构

前端开发框架整体采用模块化的设计思想，使用模块化工具webpack构建前端自动化流程，将松散的模块按照依赖和规则打包成符合生产环境部署的前端资源。同时将按需加载的模块进行代码分隔，等到实际需要的时候再异步加载。通过 loader 插件的转换，任何形式的资源都可以视作模块，比如 CommonJs 模块、 AMD 模块、 ES6 模块、CSS、图片、 JSON、 LESS 等；使用npm来管理node\_modules的包依赖；视图层采用**jquery+bootstrap**进行渲染，通过 **artTemplate**提供更加灵活的界面组件定制复用功能；数据层通过**ajax**进行数据的异步请求和储存的更新；同时为了提高用户体验，在视图层通过 **director.js** 路由来实现操作历史的管理；同时保证客户端加载资源时的增量更新，条件允许下，可把静态资源部署在单独的 **cdn**服务器上以提高页面的访问速度，降低数据服务器的压力。

2、缓存方案

鉴于项目要求可支持到IE8+浏览器，所以可使用Html中localstorage将请求数据缓存到浏览器的localstorage中。

3、技术框架

Ø jquery、Bootstrap：实现DOM渲染、数据交互、组件定制

Ø director.js : 在不刷新页面的情况下，利用“#”符号组织不同的URL路径，并根据不同的URL路径来匹配不同的回调方法，提高用户体验。

Ø Requirejs：模块加载器，遵循AMD规范，用于异步加载HTML、css、js代码模块。

Ø Less：css预处理器，用编程的思想编写css样式表，在css开发中引入变量，函数等概念，提升css的开发效率和维护成本。

Ø Webpack：前端自动化构建工具。用于前端工程打包发布，通过配置完成前端资源的压缩合并，以及资源动态添加版本号等繁琐工作。

 

 