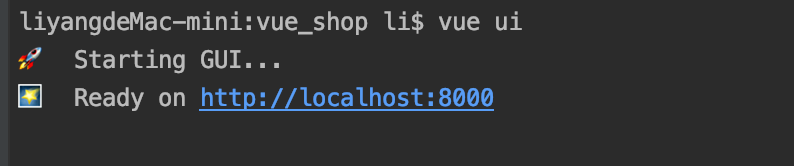
### vue\_shop启动说明

1. 项目后端api已经写好在vue\_api\_server下
   1. 通过指令 node app.js启动
   2. api接口文档见同级目录 API\_Doc.md
2. 前端项目由vue脚手架 4.1.1版本启动
   1. 通过指令vue ui 启动项目管理页面



b) 第一次进入没有项目，手动导入



c) 点击运行启动前端项目



### Vue优化策略

1. 生成打包报告

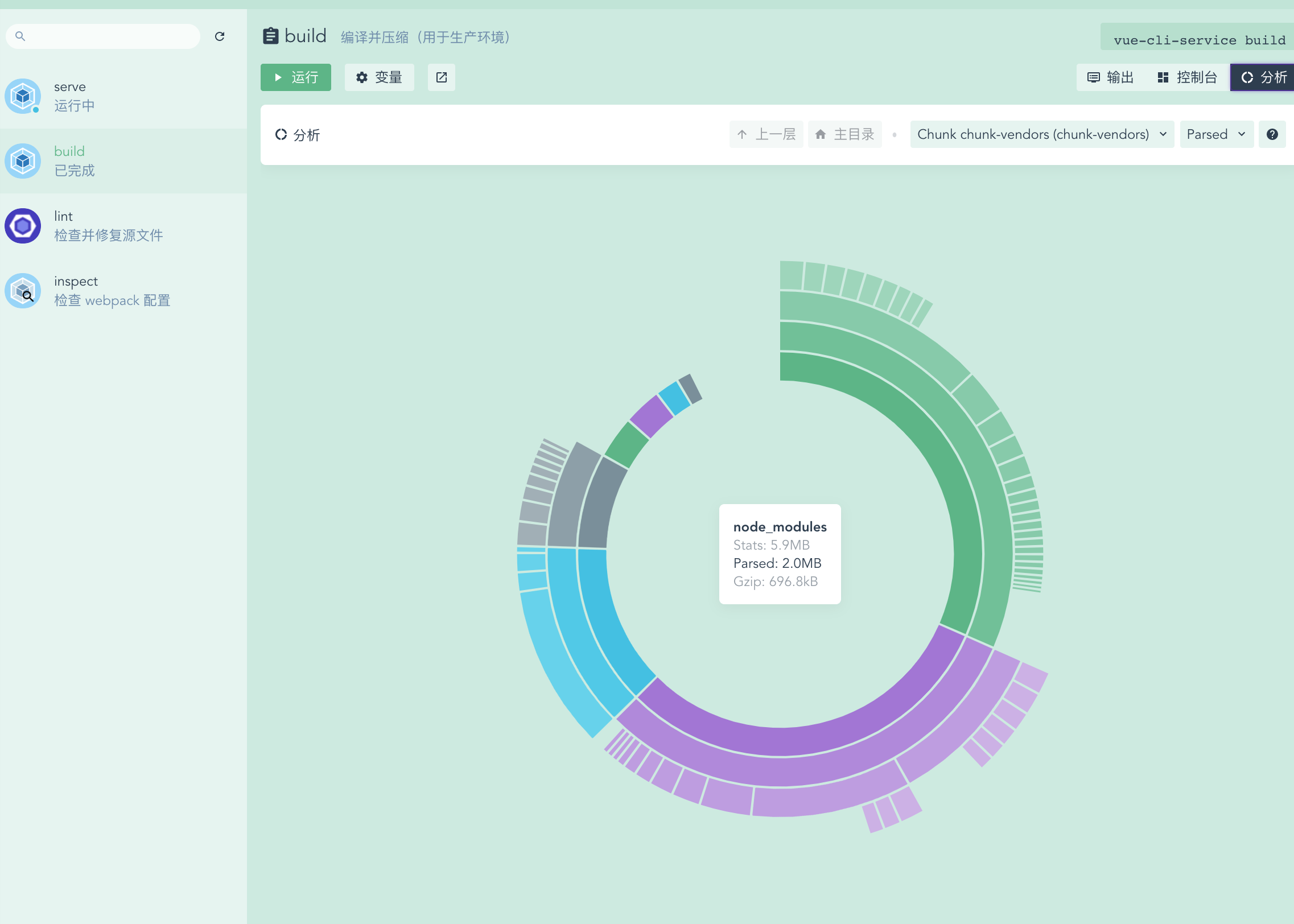
babel-plugin-transform-remove-console 可以在打dist包的时候，过滤掉console方法调用

1.1打包的时候，为了直观地发现项目中存在的问题，可以在打包时生成报告。生成报告的方式有两种：

1. 通过命令行参数的形式(通过vue-cli的命令选项可以生成打包报告，--report选项可以生成report.html 已帮助分析包内容)

Vue-cli-service build --report

B.通过可视化面板的UI面板直接查看报告



1.2通过vue.config.js修改webpack的默认配置

通过vue-cli 3.0工具生成项目，默认隐藏了所有webpack的配置项，目的是屏蔽项目的配置过程，让程序员把工作的重心，放到具体功能和业务逻辑的实现上。

如果有修改webpack默认配置的需求，可以在项目根目录中，按需创建vue.config.js这个配置文件，从而对项目的打包发布过程做自定义的配置（具体配置参考 <https://cli.vuejs.org/zh/config/#vue-config-js>

1.3为开发模式与发布模式指定不同的打包入口

默认情况下，Vue项目的开发模式与发布模式，公用一个打包的入口文件（即src/main.js）。为了将项目的开发过程与发布过程分离，我们可以分两种模式，各自指定打包的入口文件，即：

1. 开发模式的入口文件为src/main-dev.js
2. 发布模式的入口文件为src/main-prod.js

1.4 configureWebpack 和 chainWebpack

在vue.config.js导出的配置对象中，新增这俩节点，来自定义webpack的打包配置，这俩作用相同，区别在于一个是面向对象的配置方式，一个是链式配置方式。

1.5通过externals 加载外部CDN资源

默认情况下，通过import语法导入的第三方依赖包，最终会被打包合并到同一个文件中，从而导致打包成功后，单文件体积过大的问题。

为解决上述问题，可以通过webpack的external节点，来配置并加载外部的CDN资源。凡是生命在externals中的第三方依赖包，都不会被打包。

1. 第三方库启用CDN
2. Element-UI 组件按需加载
3. 路由懒加载
4. 首页内容定制