# 初始化

## 创建



功能区：点选安装 babel router 配置 不要选择linter



## 安装element



## 安装axios



## 安装less





# 图标



在main.js

import './assets/fonts/iconfont.css'

prefix-icon**="iconfont icon-3702mima"**

# Login

## 居中

position: absolute;  
left: 50%;  
top: 50%;  
transform: translate(-50%, -50%);

## 表单验证重置

ref prop是用来验证数据的指定

rule进行验证具体的规则的

v-model是用来绑定数据的

this.$refs.loginFormRef.resetFields();重置

## 消息机制

this.$message.success("成功");

## 保存机制

保存信息 存储token sessionStorage会话机制 locals是永久性质的

window.sessionStorage.setItem("token", res.data.token);

## 路由跳转

this.$router.push("/home");

# 挂载

## Message

// 这个在element.js里面的 挂在到vue的原型对象上面  
Vue.prototype.$message = Message  
Vue.prototype.$confirm = MessageBox.confirm

// 挂在到vue的原型对象上面  
Vue.prototype.$message = *Message*

## Axios

// 在main.js 导入axios  
import axios from 'axios'  
Vue.prototype.$http = axios  
Vue.config.productionTip = false  
axios.defaults.baseURL = 'http://127.0.0.1:8888/api/private/v1/'  
// 挂载到vue的原型对象上面 通过this直接访问$http 从而发起ajax的请求 挂载token  
axios.interceptors.request.use(config => {  
 //展示进度条  
 NProgress.start()  
 config.headers.Authorization = window.sessionStorage.getItem('token')  
 return config  
})  
  
axios.interceptors.response.use(config => {  
 //隐藏进度条  
 NProgress.done()  
 return config  
})

# 导航守卫：类似于中间件

// 路由导航守卫 在route.js里面的内容  
*router*.beforeEach((*to*, *from*, *next*) => {  
  
 const tokenStr = *window*.sessionStorage.getItem("token")  
   
 if (*to*.path == '/login') {  
 return *next*();  
 }  
 if (!tokenStr) {  
 return *next*('/login');  
 }  
 *next*();  
})

# 路由占位符

## 解释

当访问路由home下面子元素的时候，

会把非home元素页面加载到home组件的Router-view里面去

## 页面占用

<el-main>  
 <!-- 路由占位符 -->  
 <router-view></router-view>  
</el-main>

## 路由

{  
 path: '/home',  
 component: Home,  
 redirect: '/welcome',  
 children: [  
 {path: '/welcome', component: Welcome},  
 {path: '/users', component: Users},  
 {path: '/rights', component: Rights}, ]  
},

# 作用域插槽

<el-table-column label="状态" prop="mg\_state">  
 <!-- 插槽 -->  
 <template slot-scope="scope">  
 <el-switch  
 v-model="scope.row.mg\_state"  
 active-color="#13ce66"  
 inactive-color="#ff4949"  
 @change="userStateChange(scope.row)"  
 ></el-switch>  
 </template>  
</el-table-column>

# 对话框

## 思路

使用visiable进行控制是否显示即可

## 代码

<el-dialog title="添加用户" :visible.sync="addDialogVisible" width="50%" @close="addDialogClosed">  
 <!-- model双向数据绑定 -->  
 <!-- rules验证规则 -->  
 <!-- ref表单的引用名称 -->  
 <el-form :model="addForm" :rules="addFormRules" ref="addFormRef" label-width="70px">  
 <!-- prop验证规则 -->  
 <el-form-item label="用户名" prop="username">  
 <el-input v-model="addForm.username"></el-input>  
 </el-form-item>  
 <el-form-item label="号码" prop="mobile">  
 <el-input v-model="addForm.mobile"></el-input>  
 </el-form-item>  
 </el-form>  
  
 <span slot="footer" class="dialog-footer">  
 <el-button @click="addDialogVisible = false">取 消</el-button>  
 <el-button type="primary" @click="addUser()">确 定</el-button>  
 </span>  
</el-dialog>

# 验证

data() {  
 // 验证手机号  
 var checkMobile = (rule, value, cb) => {  
 const regEmail = /^1[34578]\d{9}$/;  
 if (regEmail.test(value)) {  
 return cb();  
 }  
  
 cb(new ***Error***("请输入合法的号码"));  
 };  
  
 return {  
 addFormRules: {  
 username: [  
 {required: true, message: "请输入邮箱", trigger: "blur"},  
 {min: 0, max: 20, message: "大小不合适", trigger: "blur"}  
 ],  
}

# Treetable

## 运行依赖



## Main.js

import TreeTable from 'vue-table-with-tree-grid'

Vue.component('tree-table', TreeTable)

## Html

<!-- 表格-->  
<tree-table  
 class="treeTable"  
 :data="catelist "  
 :columns="columns"  
 show-index  
 border  
 index-text="ID"  
 :selection-type="false"  
 :expand-type="false"  
>  
</tree-table>

# 验证写法

addCate() {  
 ***console***.log(this.catForm)  
 this.$refs.catFormEsRef.validate(async valid => {  
 if (!valid) {  
 return this.$message.error("验证失败")  
 }  
  
 const {data: res} = await this.$http.post('categories', this.catForm)  
  
 if (res.meta.status !== 201) {  
 return this.$message.error("添加失败")  
 }  
 this.getCateList()  
 this.dialogVisibleCat = false  
 this.$message.success("修改成功");  
 })  
},

# 动态参数静态属性

# 循环

res.data.forEach(item => {  
 item.attr\_vals = item.attr\_vals ? item.attr\_vals.split(',') : [];  
 item.inputVisible = false;  
 item.inputValue = '';  
})

# 计算属性

computed: {  
 isBtnDisabled() {  
 if (this.selectedCateData.length !== 0) {  
 return false  
 } else {  
 return true  
 }  
 }  
}

<el-button @click="dialogVisible = true" type="primary" size="mini" :disabled="isBtnDisabled">添加参数</el-button>

# 时间过滤

<template slot-scope="scope">  
 {{ scope.row.add\_time | dataFormat }}  
</template>

Main.js里面的内容

Vue.filter('dataFormat', function (origiVal) {  
 const dt = new ***Date***(origiVal)  
 const y = dt.getFullYear()  
 const m = (dt.getMonth() + 1 + '').padStart(2, '0')  
 const d = (dt.getDate() + '').padStart(2, '0')  
  
 const hh = (dt.getHours() + '').padStart(2, '0')  
 const mm = (dt.getMinutes() + '').padStart(2, '0')  
 const ss = (dt.getSeconds() + '').padStart(2, '0')  
  
 return `${y}-${m}-${d} ${hh}:${mm}:${ss}`  
  
})

# 自动获取焦点

//当页面上元素被重新渲染后，才会执行回调函数中的代码，否则可能会报错 要等一会  
this.$nextTick(\_ => {  
 this.$refs.saveTagInput.$refs.input.focus();  
});

# 路由跳转

## 方法里的跳转

this.$router.push('/goods/add')

## html的跳转

<el-breadcrumb-item :to="{ path: '/goods' }">理</el-breadcrumb-item>

# 绑定token

## 代码

       <el-upload

                :on-success="handleSuccess"

                :action="uploadUrl"

                :on-preview="handlePreview"

                :on-remove="handleRemove"

                list-type="picture"

                :headers="headObj"

            >

## 获取数据

   headObj: {

        Authorization: window.sessionStorage.getItem('token')

      },

# 编辑工具

## 安装



## 导入与使用

import VueQuillEditor from 'vue-quill-editor'  
import 'quill/dist/quill.core.css' // import styles  
import 'quill/dist/quill.snow.css' // for snow theme  
import 'quill/dist/quill.bubble.css' // for bubble theme  
import 'quill/dist/quill.bubble.css' // for bubble theme  
Vue.use(VueQuillEditor, /\* { default global options } \*/)

## 模板

<template>  
 <quill-editor  
 v-model="addForm.goods\_introduce"  
 >  
 </quill-editor>  
 <el-button  
 @click="add"  
 type="primary"  
 class="btnAdd">添加商品  
 </el-button>  
</template>

# 深拷贝

## 安装



## 导入

import \_ from 'lodash'

## 使用

const form = \_.cloneDeep(this.addForm)

# 三级联动

## 文件

放在同一级

## 导入

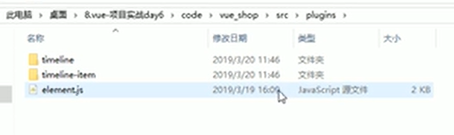
import cityData from "@/components/order/citydata";

## 级联选择器

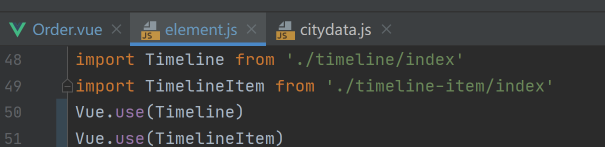
<el-form-item label="省市区县" prop="address1">  
 <!-- v-model是选中数据的-->  
 <!-- options是数据源-->  
 <!-- props：checkStrictly是否只能选中最末尾的元素-->  
 <!-- props：expandTrigger展开触发方式-->  
 <!-- props：value选中的值-->  
 <!-- props：label显示的值-->  
 <!-- props：children子代元素-->  
 <el-cascader  
 v-model="addressForm.address1"  
 :options="cityData"  
 clearable  
 >  
 </el-cascader>

# 导入插件

## 复制文件



## 插件



## 使用

<el-timeline>  
 <el-timeline-item  
 v-for="(activity, index) in processInfo"  
 :key="index"  
 :timestamp="activity.time">  
 {{ activity.context }}  
 </el-timeline-item>  
</el-timeline>

## 导入样式表

@import "../../plugins/timeline/timeline.css";  
@import "../../plugins/timeline-item/timeline-item(1).css";

# Echarts

## 安装



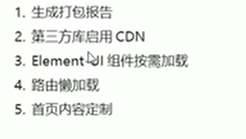
## 导入

import \* as echarts from 'echarts'

## 模板

//dom绘制完之后进行执行mounted  
async mounted() {  
  
 var myChart = echarts.init(***document***.getElementById('main'));  
  
 const {data: res} = await this.$http.get('reports/type/1')  
  
 if (res.meta.status !== 200) {  
 return this.$message.error("失败")  
 }  
  
 const result = \_.merge(res.data, this.options)  
  
 myChart.setOption(result)  
  
},

# 策略



# 进度条

## 安装



## 导入

import NProgress from 'nprogress'  
import 'nprogress/nprogress.css'  
  
Vue.prototype.$http = axios  
  
Vue.config.productionTip = false  
axios.defaults.baseURL = 'http://127.0.0.1:8888/api/private/v1/'  
// 挂载到vue的原型对象上面 通过this直接访问$http 从而发起ajax的请求  
axios.interceptors.request.use(config => {  
 //展示进度条  
 NProgress.start()  
 config.headers.Authorization = ***window***.sessionStorage.getItem('token')  
 return config  
})  
  
axios.interceptors.response.use(config => {  
 //隐藏进度条  
 NProgress.done()  
 return config  
})

# 去除eslint

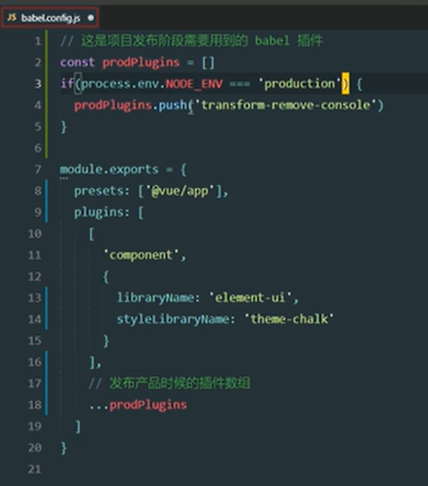


# 去除console

## 安装



## 代码



# 打包报告



# 修改打包配置

## 配置



## Chainwebpack

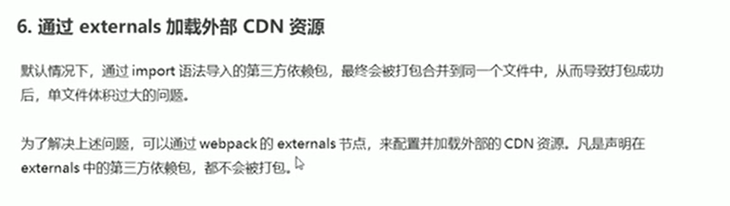


## 使用



# CDN

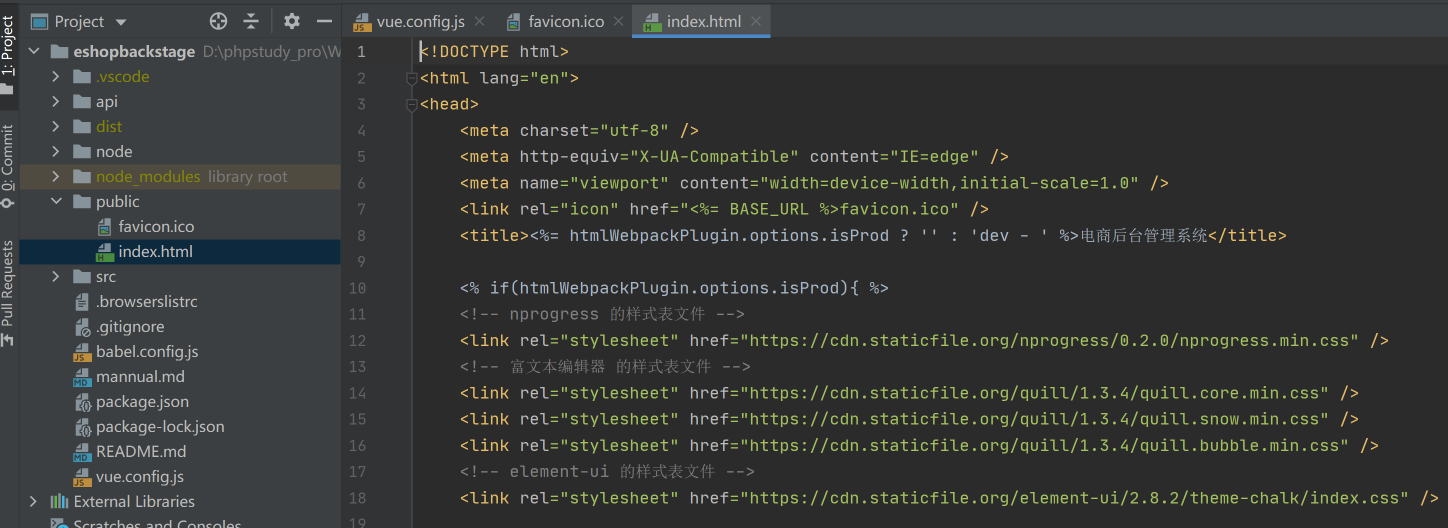
## 外部



## 代码

***module***.***exports*** = {  
 chainWebpack: config => {  
 // 发布模式  
 config.when(process.env.NODE\_ENV === 'production', config => {  
 config  
 .entry('app')  
 .clear()  
 .add('./src/main-prod.js')  
  
 config.set('externals', {  
 vue: 'Vue',  
 'vue-router': 'VueRouter',  
 axios: 'axios',  
 lodash: '\_',  
 echarts: 'echarts',  
 nprogress: 'NProgress',  
 'vue-quill-editor': 'VueQuillEditor'  
 })  
  
 config.plugin('html').tap(args => {  
 args[0].isProd = true  
 return args  
 })  
 })

## cdn



## 思想

不在编译工具里面找vue等信息了 直接去cdn找了 直接去index.html找

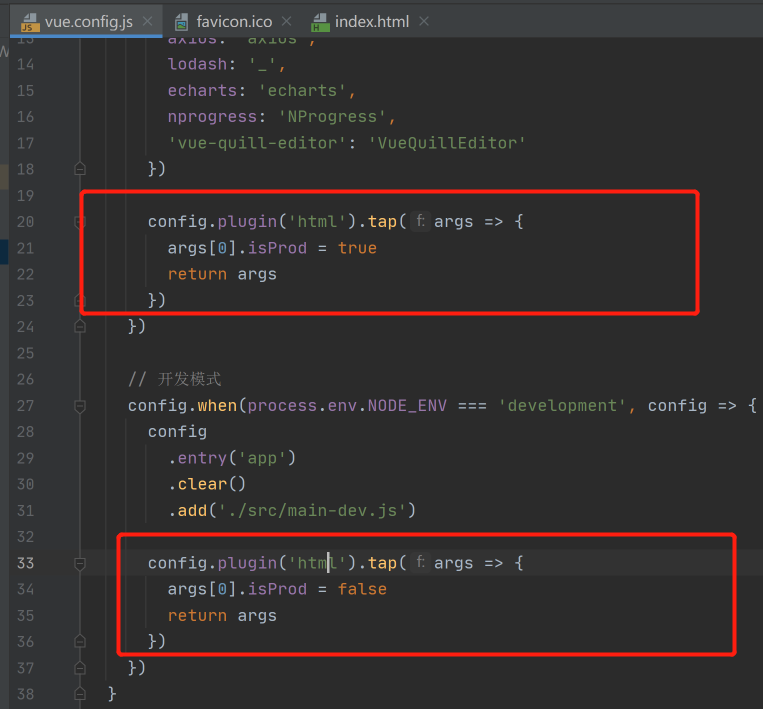
# 优化element



# 环境变量配置

## 配置





## 使用变量



<% if(htmlWebpackPlugin.options.isProd){ %>

<% } %>

# 路由懒加载

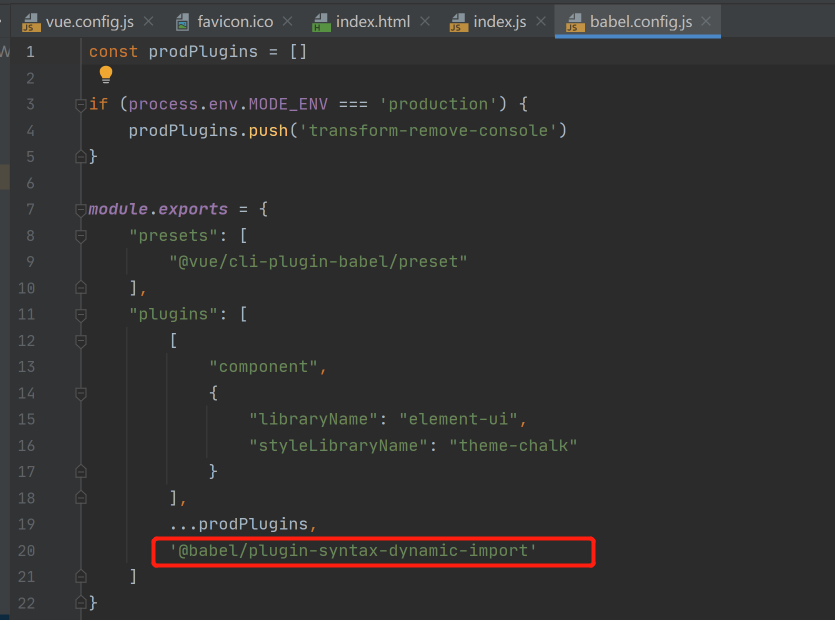
## 步骤



## 安装



## 配置



## 改造

// import Login from './components/Login.vue'  
// import Home from './components/Home.vue'  
// import Welcome from './components/Welcome.vue'  
// import Users from './components/user/Users.vue'  
// import Rights from './components/power/Rights'  
// import Roles from './components/power/Roles.vue'  
const Login = () => import(/\* webpackChunkName: "login\_home\_welcome" \*/ './components/Login.vue')  
const Home = () => import(/\* webpackChunkName: "login\_home\_welcome" \*/ './components/Home.vue')  
const Welcome = () => import(/\* webpackChunkName: "login\_home\_welcome" \*/ './components/Welcome.vue')  
  
const Users = () => import(/\* webpackChunkName: "Users\_Rights\_Roles" \*/ './components/user/Users.vue')  
const Rights = () => import(/\* webpackChunkName: "Users\_Rights\_Roles" \*/ './components/power/Rights')  
const Roles = () => import(/\* webpackChunkName: "Users\_Rights\_Roles" \*/ './components/power/Roles.vue')  
  
const Cate = () => import(/\* webpackChunkName: "Cate\_Params" \*/ './components/goods/Cate.vue')  
const Params = () => import(/\* webpackChunkName: "Cate\_Params" \*/ './components/goods/Params.vue')  
  
const List = () => import(/\* webpackChunkName: "List\_Add" \*/ './components/goods/List.vue')  
const Add = () => import(/\* webpackChunkName: "List\_Add" \*/ './components/goods/Add.vue')  
  
const Order = () => import(/\* webpackChunkName: "Order\_Report" \*/ './components/order/Order.vue')  
const Report = () => import(/\* webpackChunkName: "Order\_Report" \*/ './components/report/Report.vue')

# SSL

## 步骤

