Table of Contents

[第一步，安装node.js 2](#_Toc167104141)

[第二步，搭建VUE环境 3](#_Toc167104142)

[第三步，创建VUE项目 4](#_Toc167104143)

[第四步，使用vant构建页面 11](#_Toc167104144)

[第五步，PO配置 16](#_Toc167104145)

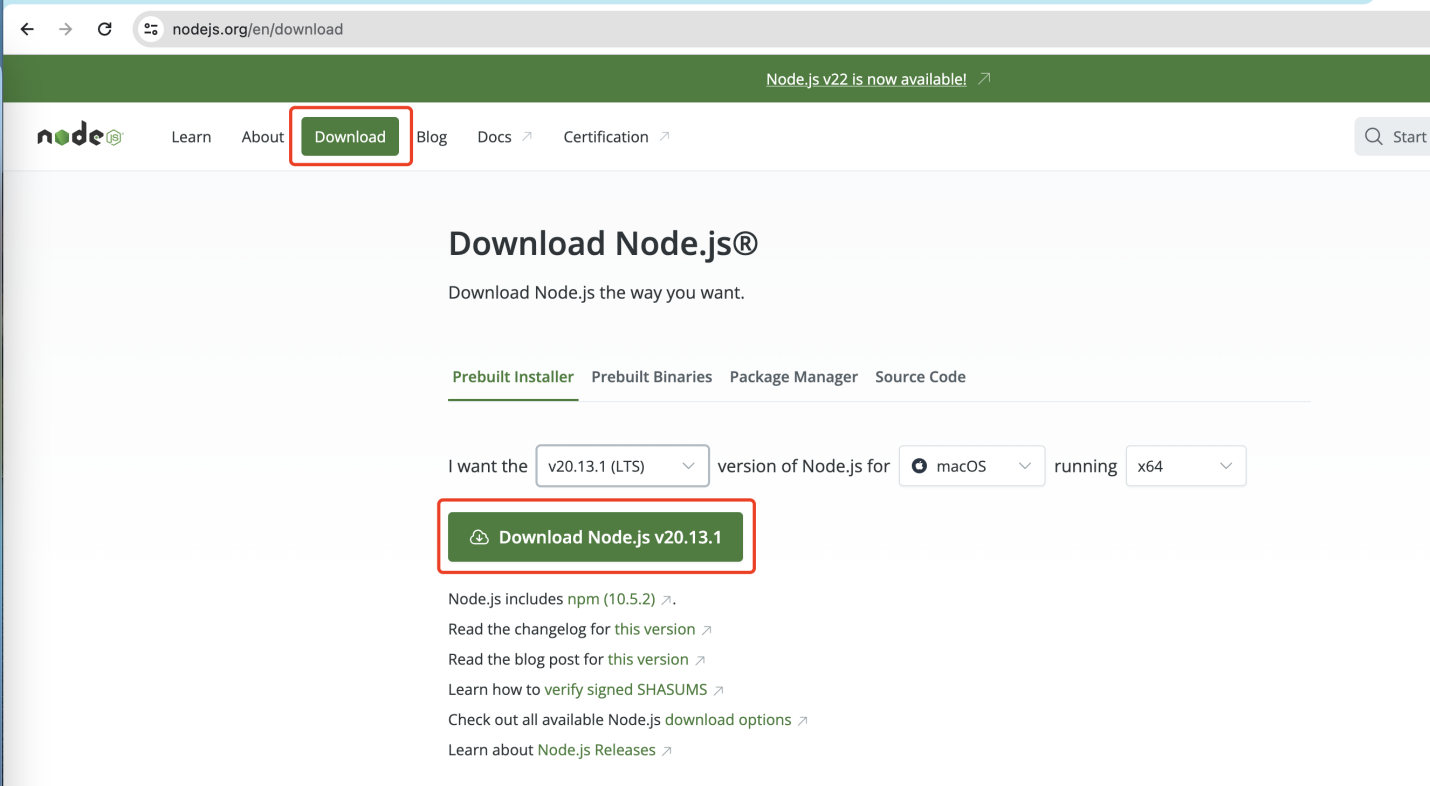
[第六步，ABAP 20](#_Toc167104146)

[第七步 ，部署 20](#_Toc167104147)

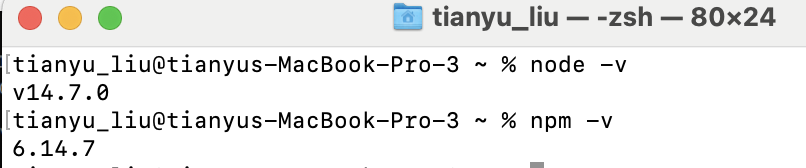
# 第一步，安装node.js

下载地址：<https://nodejs.org/en/download>

版本、OS自己选择，我的版本是node.js v14.7.0。



在cmd中使用node -v检查是否安装成功，如下图



如果NPM的效率低可以安装国内的镜像CNPM，命令如下

使用CMD安装

npm install -g cnpm -registry= [https://registry.npmmirror.com](https://registry.npmmirror.com/" \o "https://registry.npmmirror.com)

1、查看镜像仓库  
npm config get registry  
2、设置镜像仓库  
npm config set registry https://registry.npmmirror.com

# 第二步，搭建VUE环境

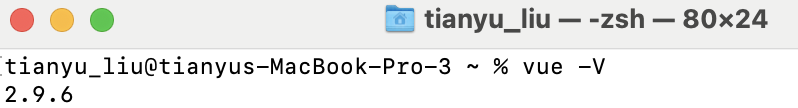
全局安装脚手架 vue-cli

CMD安装命令如下

npm install -g vue-cli (vue-lcli2，使用vue-cli2.x初始化项目安装这个)

npm install -g @vue/cli (vue-cli3，使用vue-cli3.x初始化项目安装这个）

我是用的是2.9.6



全局安装 webpack

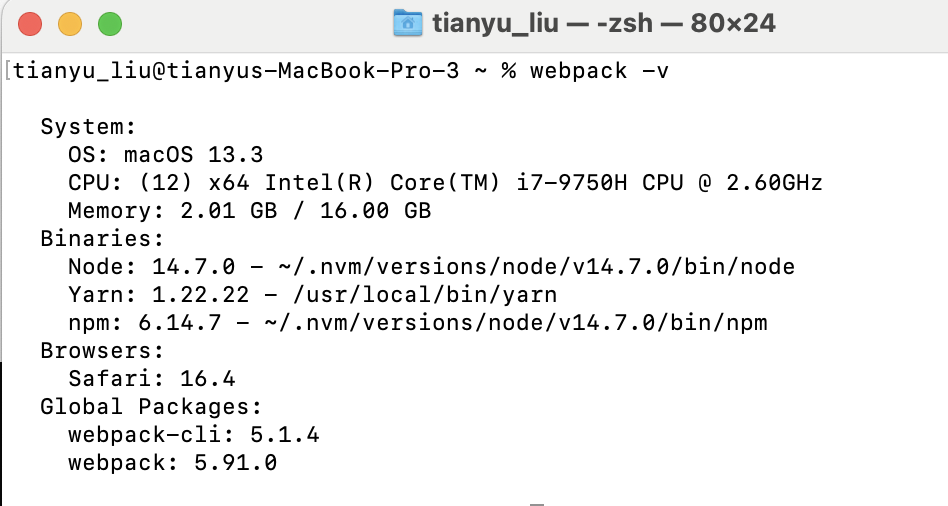
CMD安装命令如下

npm install -g webpack

全局安装 webpack-cli

CMD安装命令如下

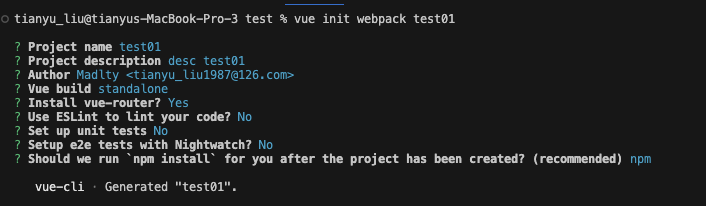
npm install -g webpack-cli



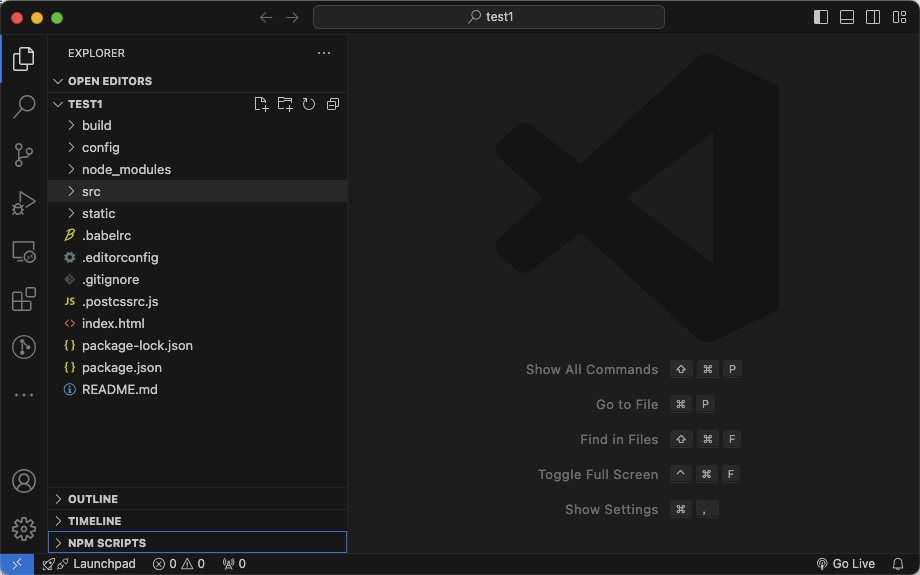
# 第三步，创建VUE项目

使用Visual Studio Code

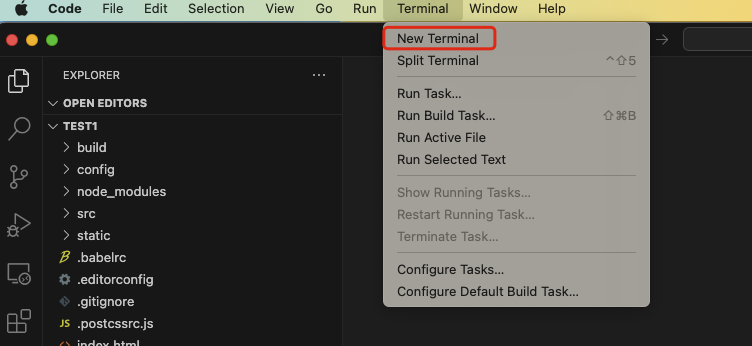
**vue init webpack test01**



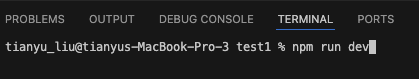
VUE项目创建好后

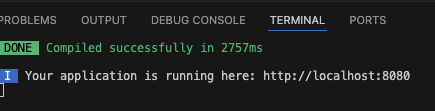


新建一个Terminal，

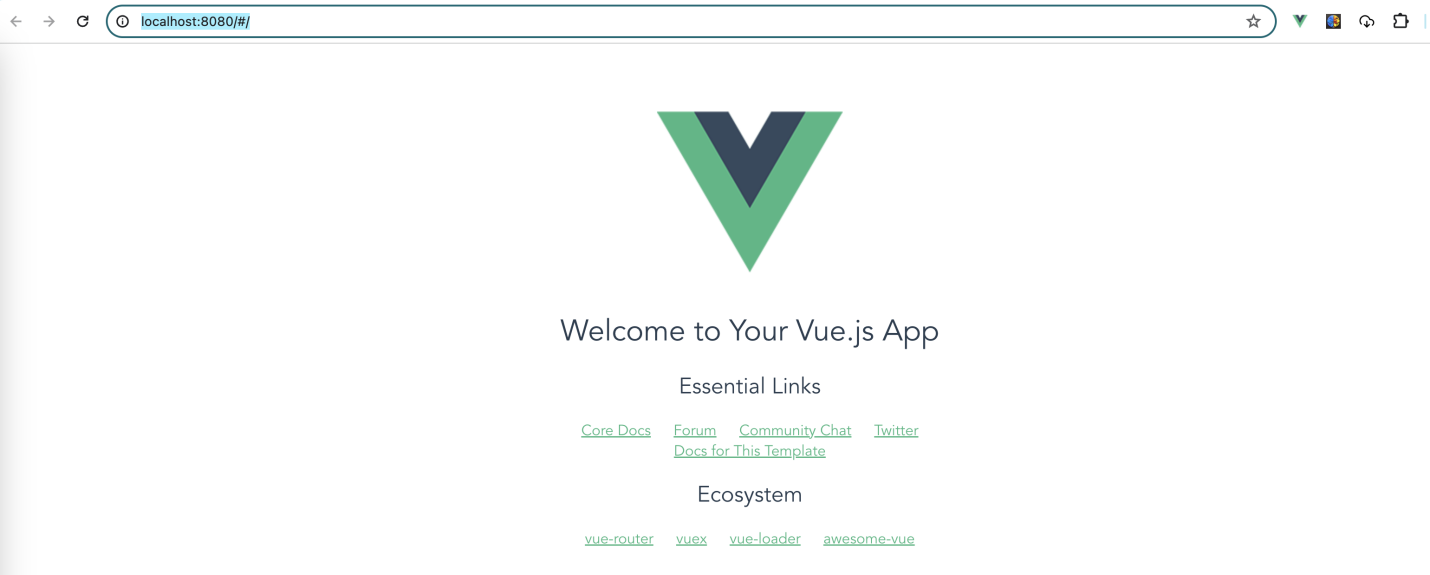


执行npm run dev/ npm run serve





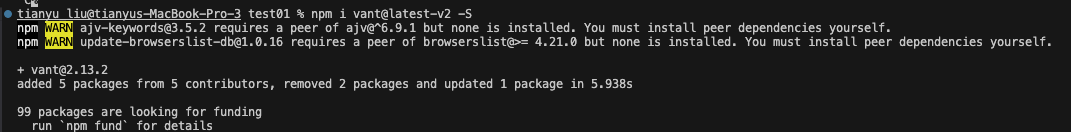
访问<http://localhost:8080>，获得下面的界面证明项目已经创建成功。



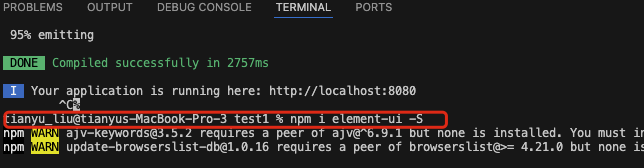
第四步安装必须的组件

Vant 移动端UI npm i vant@latest-v2 -S

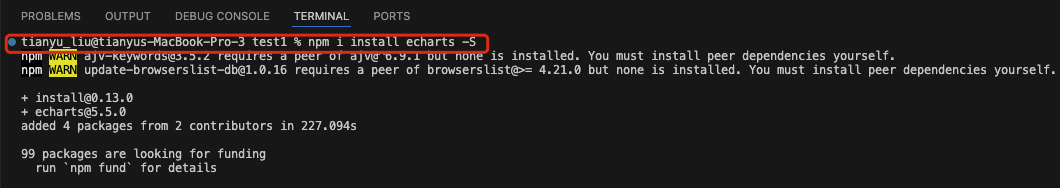
https://youzan.github.io/vant/v2/



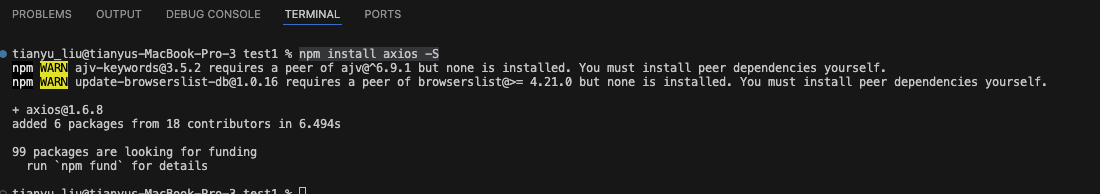
Element UI 电脑UI npm i element-ui -S



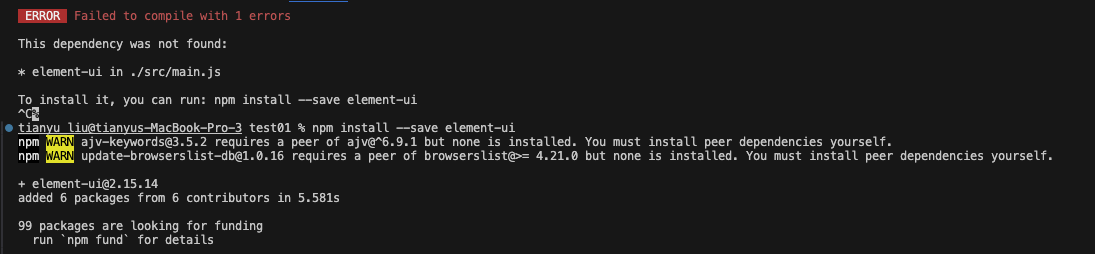
Echart 图表 npm i install echarts -S



安装axios npm install axios@1.5.0 -S



npm install --save element-ui



npm install -S nwabap-ui5uploader

安装上载到sap的组件



组件安装好后需要在main.js中添加引用

// The Vue build version to load with the `import` command

// (runtime-only or standalone) has been set in webpack.base.conf with an alias.

import Vue from 'vue'

import App from './App'

import router from './router'

import vant from 'vant'

import 'vant/lib/index.css'

import ElementUI from 'element-ui'

import \* as echarts from 'echarts'

import utils from './utils'

Vue.config.productionTip = false

Vue.use(vant)

Vue.use(ElementUI)

Vue.prototype.$echarts = echarts;

Vue.use(utils)

/\* eslint-disable no-new \*/

new Vue({

el: '#app',

router,

components: {

App

},

template: '<App/>'

})

添加基础配置

config/dev.env.js

'use strict'

const merge = require('webpack-merge')

const prodEnv = require('./prod.env')

module.exports = merge(prodEnv, {

NODE\_ENV: '"development"',

VUE\_APP\_BASE\_API: '"/api/"',

VUE\_APP\_BASE\_API\_TOKEN: '"/saptoken/"',

VUE\_APP\_TITLE: '"EaspringChart01"',

VUE\_APP\_BASE\_USER: '"liuty"',

VUE\_APP\_BASE\_PWD: '"Init\_123"',

})

config/index.js

'use strict'

// Template version: 1.3.1

// see http://vuejs-templates.github.io/webpack for documentation.

const path = require('path')

module.exports = {

dev: {

// Paths

assetsSubDirectory: 'static',

assetsPublicPath: '/',

proxyTable: {

"/api": {

target: 'http://10.20.60.57:50000/RESTAdapter/',

changeOrigin: true,

pathRewrite: {

'^/api': '/'

}

}

},

// Various Dev Server settings

host: 'localhost', // can be overwritten by process.env.HOST

port: 8080, // can be overwritten by process.env.PORT, if port is in use, a free one will be determined

autoOpenBrowser: false,

errorOverlay: true,

notifyOnErrors: true,

poll: false, // https://webpack.js.org/configuration/dev-server/#devserver-watchoptions-

/\*\*

\* Source Maps

\*/

// https://webpack.js.org/configuration/devtool/#development

devtool: 'cheap-module-eval-source-map',

// If you have problems debugging vue-files in devtools,

// set this to false - it \*may\* help

// https://vue-loader.vuejs.org/en/options.html#cachebusting

cacheBusting: true,

cssSourceMap: true,

productionSourceMap: true,

// https://webpack.js.org/configuration/devtool/#production

devtool: '#source-map',

},

build: {

// Template for index.html

index: path.resolve(\_\_dirname, '../dist/index.html'),

// Paths

assetsRoot: path.resolve(\_\_dirname, '../dist'),

assetsSubDirectory: 'static',

assetsPublicPath: './',

/\*\*

\* Source Maps

\*/

productionSourceMap: true,

// https://webpack.js.org/configuration/devtool/#production

devtool: 'cheap-module-source-map', //'#source-map',

// Gzip off by default as many popular static hosts such as

// Surge or Netlify already gzip all static assets for you.

// Before setting to `true`, make sure to:

// npm install --save-dev compression-webpack-plugin

productionGzip: false,

productionGzipExtensions: ['js', 'css'],

// Run the build command with an extra argument to

// View the bundle analyzer report after build finishes:

// `npm run build --report`

// Set to `true` or `false` to always turn it on or off

bundleAnalyzerReport: process.env.npm\_config\_report

}

}

config/prod.evn.js

'use strict'

module.exports = {

NODE\_ENV: '"production"',

VUE\_APP\_BASE\_API: '"http://10.20.60.57:50000/RESTAdapter/"',

VUE\_APP\_BASE\_API\_TOKEN: '"/saptoken/"',

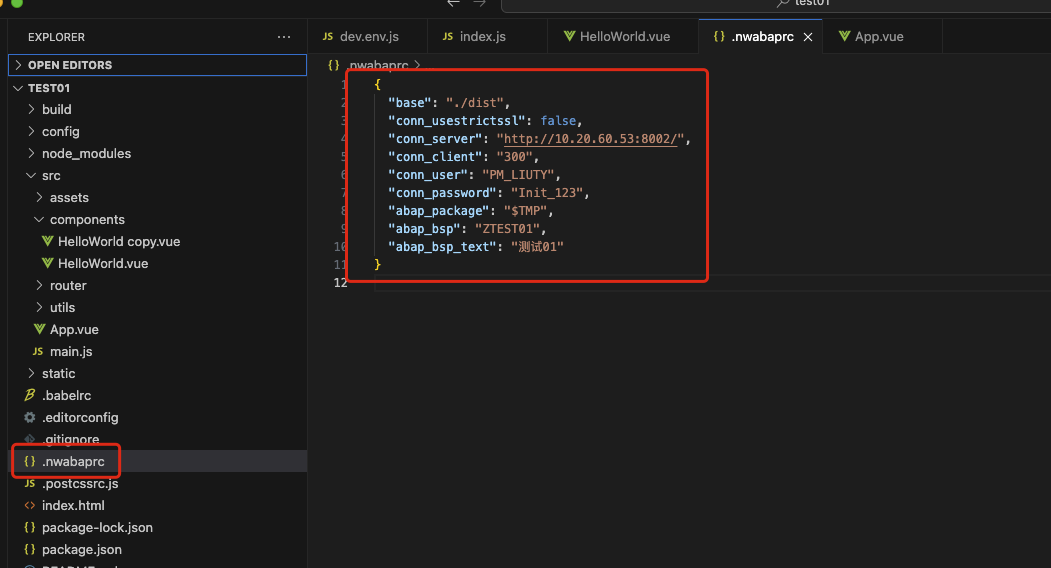
VUE\_APP\_TITLE: '"EaspringChart01"',

VUE\_APP\_BASE\_USER: '"liuty"',

VUE\_APP\_BASE\_PWD: '"Init\_123"',

}

添加后续部署的配置



{

"base": "./dist",

"conn\_usestrictssl": false,

"conn\_server": "http://10.20.60.53:8002/",

"conn\_client": "300",

"conn\_user": "PM\_LIUTY",

"conn\_password": "Init\_123",

"abap\_package": "$TMP",

"abap\_bsp": "ZTEST01",

"abap\_bsp\_text": "测试01"

}

至此我们前期准本工作已经做完，相面我们开始做应用的开发。本次以手机端图表为例子，使用echart的作为图表工具，使用vant做手机端的组件，其中还使用了element-ui的table组件。

# 第四步，使用vant构建页面

我们到src/componetns/HelloWorld.vue下,可以直接修改也可以复制后修改。修改后内容如下

添加一个期间选择

<div class="period">

<van-cell

title="期间"

is-link

:value-class="className"

:value="timeValue"

@click="showPopupPeriod"

/>

<van-popup v-model="showPeriod" position="bottom">

<van-datetime-picker

v-model="currentDate"

type="year-month"

title="期间"

:loading="isLoadingShowPeriod"

:min-date="minDate"

:max-date="maxDate"

:formatter="formatter"

@cancel="showPeriod = false"

@confirm="confirmPicker"

/>

</van-popup>

</div>

添加一个展示表的div

<div id="barChartQuantity" class="barChartQuantity"></div>

添加一个展示图表数据的table

<el-table

:data="barChartQuantityData"

border

style="width: 362px; margin: 0 auto"

>

<el-table-column prop="WEEK" label="周" width="120" align="center">

</el-table-column>

<el-table-column prop="MENGE" label="数量" width="120" align="center">

</el-table-column>

<el-table-column prop="MEINS" label="单位" width="120" align="center">

</el-table-column>

</el-table>

还需要添加样式

<style scoped>

.barChartQuantity {

width: 100%;

height: 300px;

/\* border: 3px solid pink; \*/

margin: 20px auto;

}

</style>

引入调用PO服务的封装函数

import request from "../utils/request";

定义初始化的变量

data() {

return {

className: "",

timeValue: "请选择期间",

showPeriod: false,

currentDate: new Date(),

isLoadingShowPeriod: true,

minDate: new Date(2022, 7, 31),

maxDate: new Date(2030, 12, 31),

periodHaveData: false,

barChartQuantityData: [],

};

},

页面在初始化时使用的方法,

mounted() {

//自适应

window.onresize = () => {

this.$nextTick(() => {

this.barChartQuantity.resize();

});

};

//选择条件初始化

let year = this.currentDate.getFullYear();

let month = this.currentDate.getMonth() + 1;

if (month >= 1 && month <= 9) {

month = `0${month}`;

}

this.timeValue = `${year}${month}`;

this.reqData.SPMON = this.timeValue;

//数据获取

this.getChart();

},

获取图表数据的方法

getChart() {

let that = this;

var resWeekData = {};

var resBrandData = {};

request({

url: "OA/SD220",

method: "post",

data: this.reqData,

}).then((res) => {

if (res.data != "") {

resWeekData = res.data.WEEK\_DATA;

that.periodHaveData = true;

//周

that.barChartQuantityData.length = 0;

resWeekData.data.map((item) => {

let data = { WEEK: "", MENGE: "", MEINS: "" };

data.WEEK = item.WEEK;

data.MENGE = that.formatNum(item.MENGE);

data.MEINS = item.MEINS;

that.barChartQuantityData.push(data);

});

this.$nextTick(() => {

//周

that.getBarChartQuantity(resWeekData);

});

} else {

that.periodHaveData = false;

}

});

},

获取到图表数据后生成图表的方法

//周 数量

getBarChartQuantity(resData) {

let that = this;

var optionQuantity = {

title: [

{

text: "本期按周销售量",

},

],

xAxis: {

type: "category",

data: (() => {

let list = [];

resData.xAxis.data.map((item) => {

list.push(item.WEEK);

});

return list;

})(),

},

yAxis: {

type: "value",

},

series: [

{

label: {

show: true,

position: "inside",

formatter: function (params) {

return (

that.formatNum(resData.series.data[params.dataIndex].MENGE) +

resData.series.data[params.dataIndex].MEINS

);

},

},

data: (() => {

let list = [];

resData.series.data.map((item) => {

list.push(item.MENGE);

});

return list;

})(),

type: "bar",

},

],

};

this.barChartQuantity = this.$echarts.getInstanceByDom(

document.getElementById("barChartQuantity")

);

if (this.barChartQuantity == null) {

this.barChartQuantity = this.$echarts.init(

document.getElementById("barChartQuantity")

);

}

this.barChartQuantity.setOption(optionQuantity);

},

显示格式调用的方法

//数字格式

formatNum(value) {

if (!value && value !== 0) return 0;

let str = value.toString();

let reg =

str.indexOf(".") > -1 ? /(\d)(?=(\d{3})+\.)/g : /(\d)(?=(?:\d{3})+$)/g;

return str.replace(reg, "$1,");

},

// 选项格式化函数

formatter(type, value) {

if (type === "year") {

return `${value}年`;

} else if (type === "month") {

return `${value}月`;

}

return value;

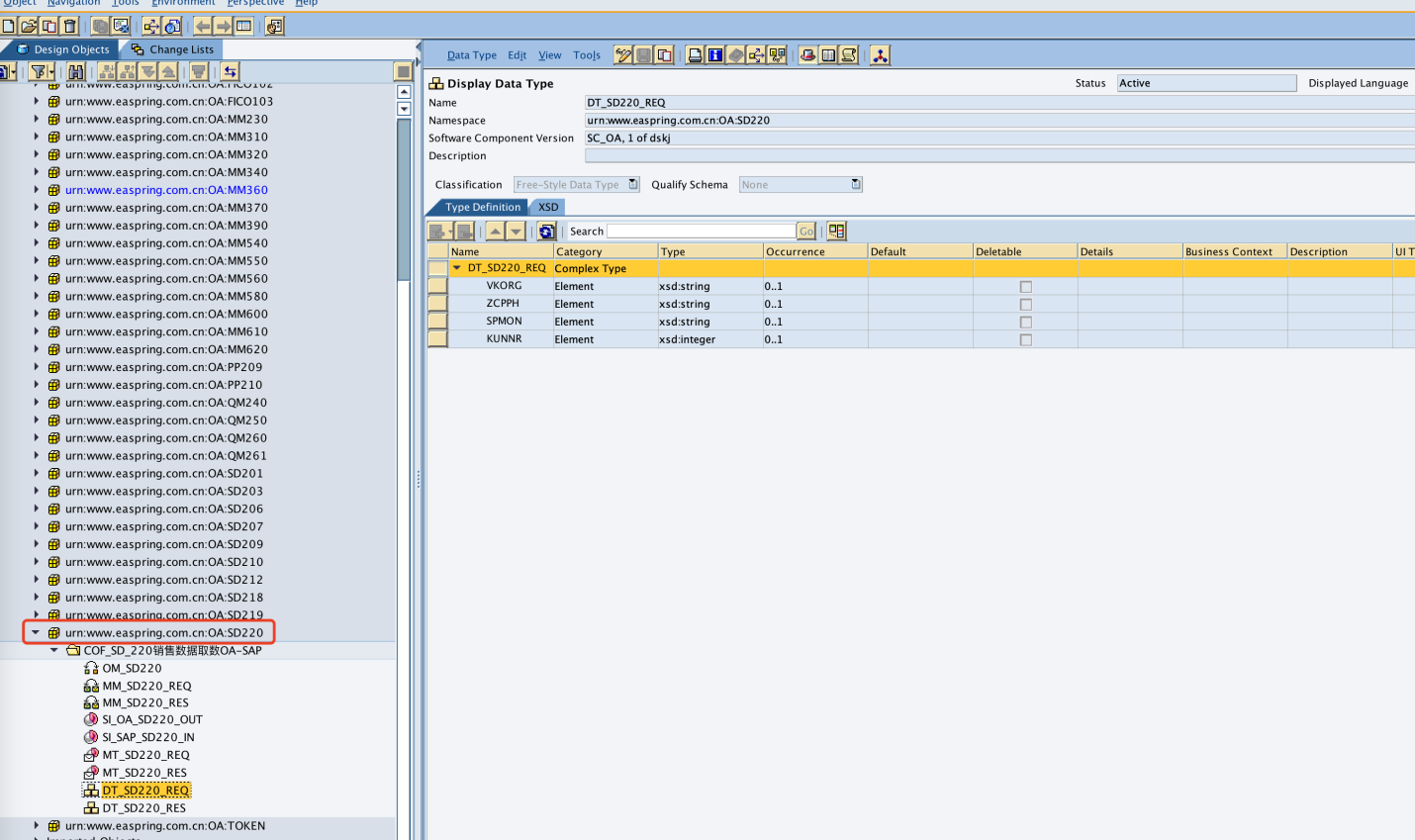
},

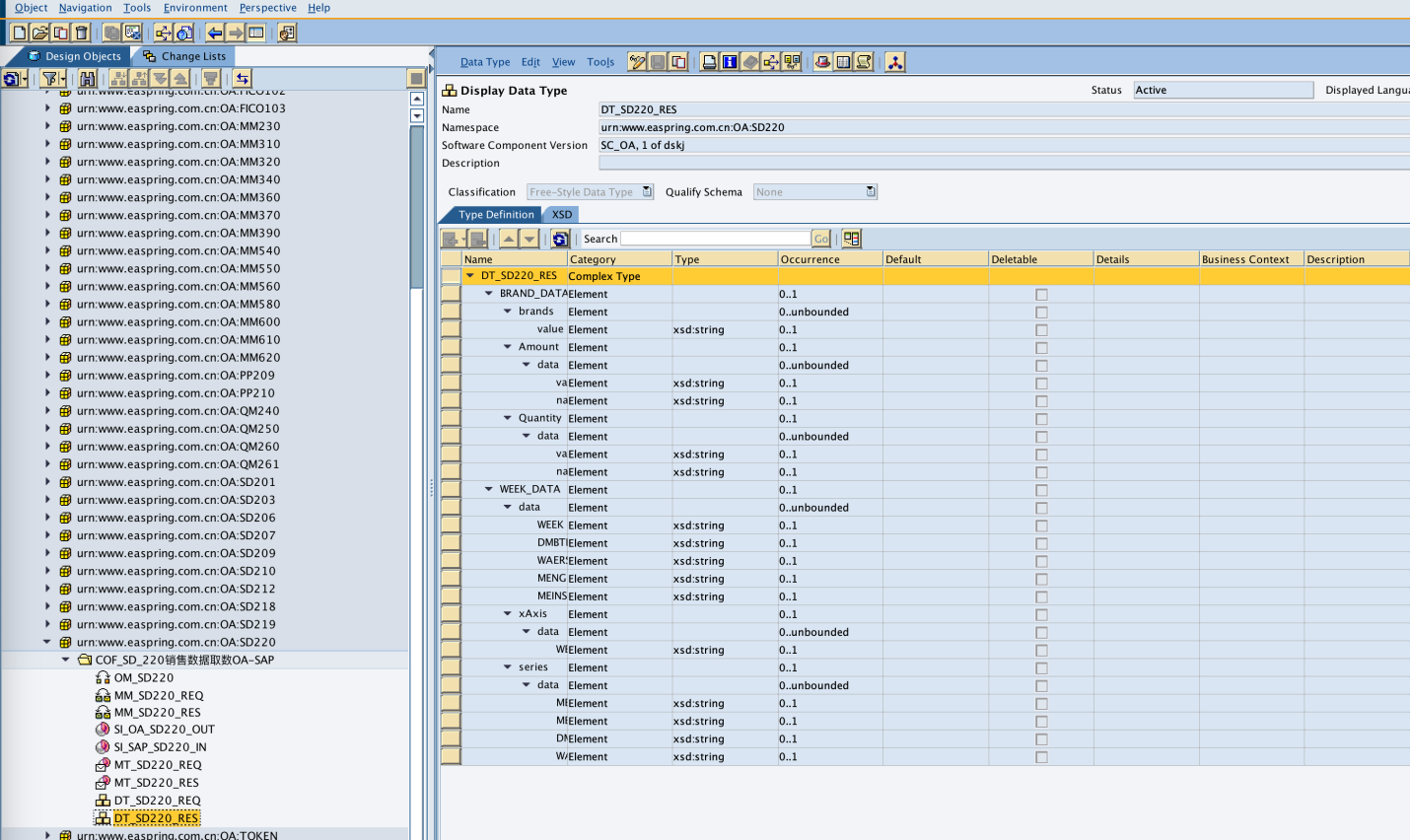
至此我们前段的程序就做完了

# 第五步，PO配置

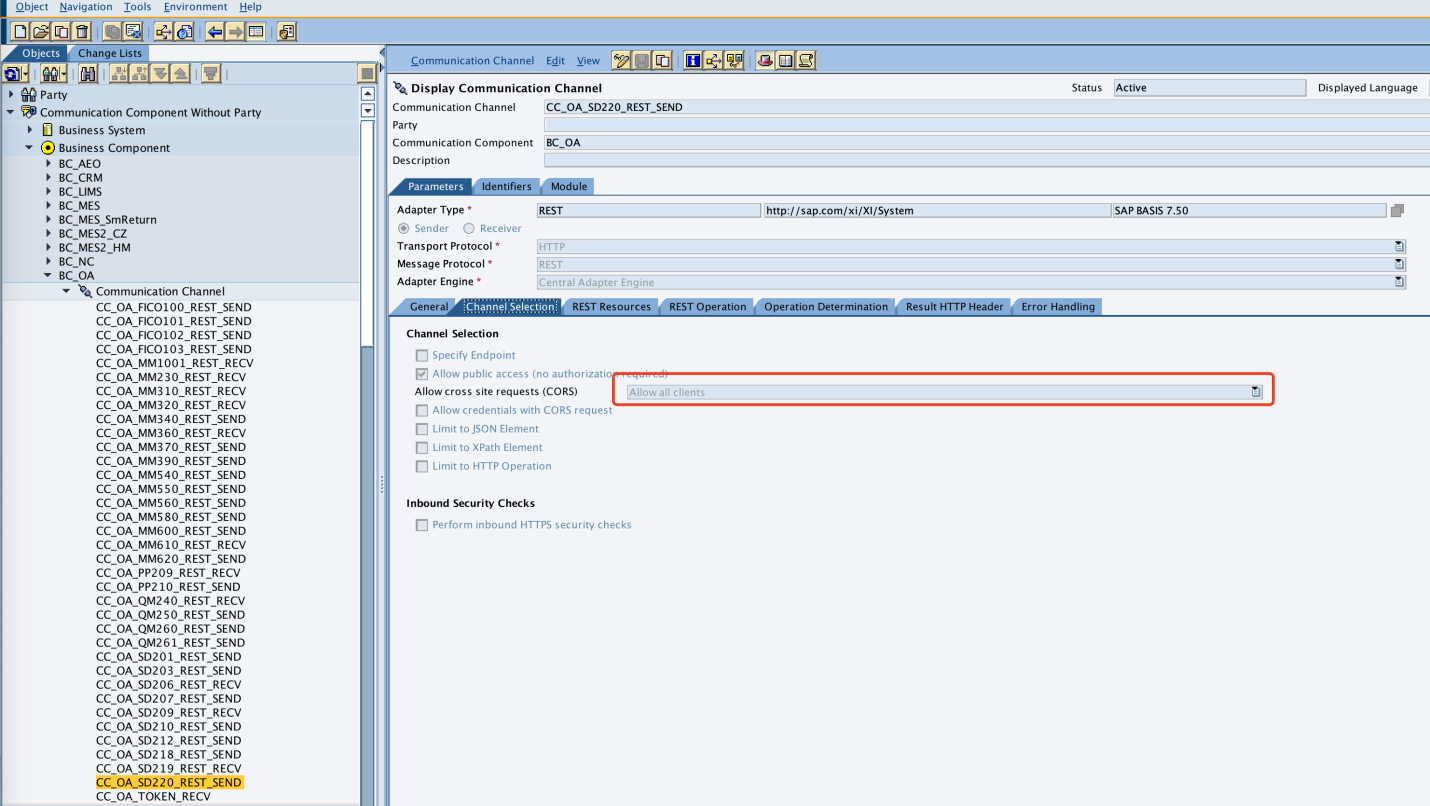
PO配置（SAP是服务端）

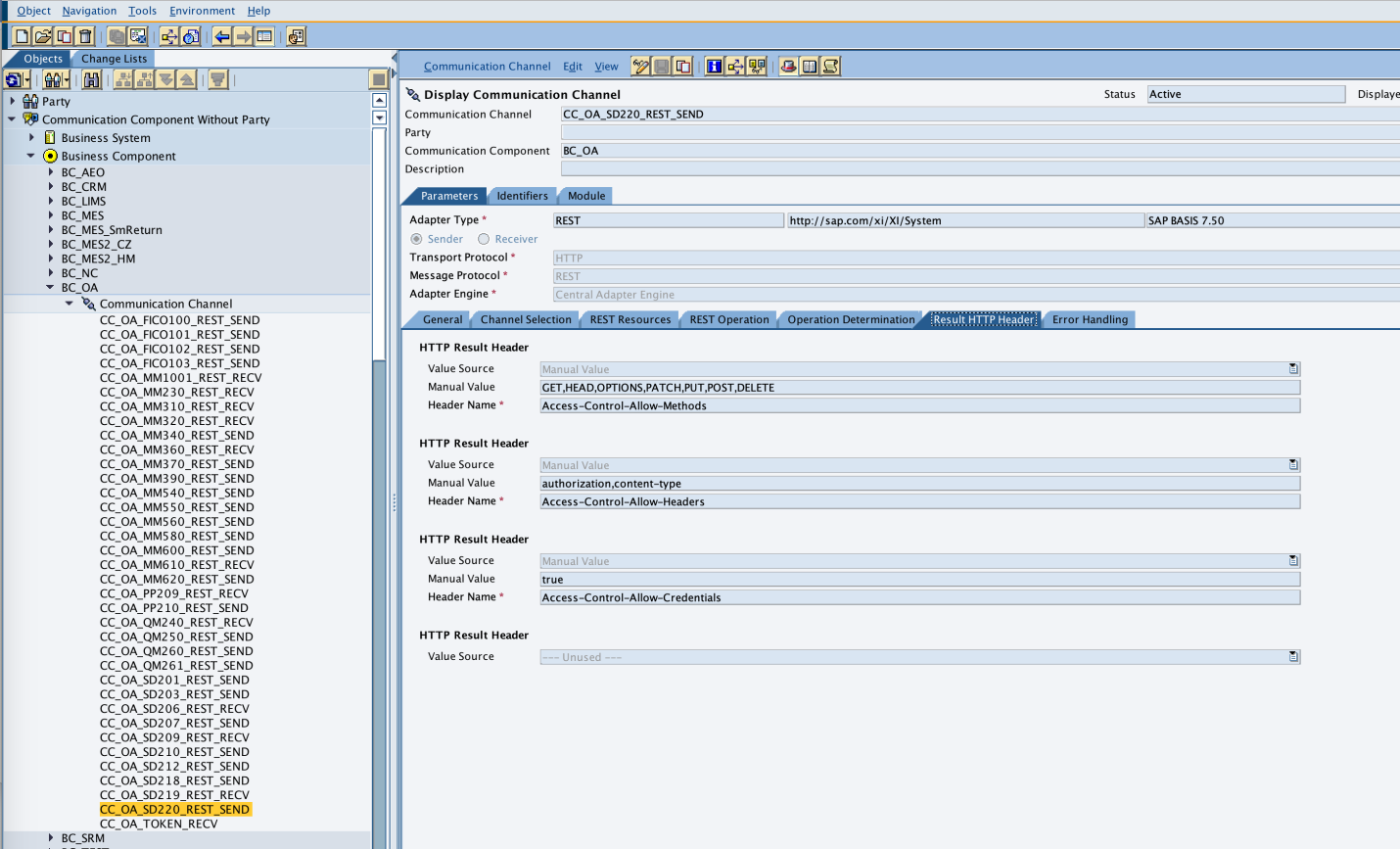
ESR配置



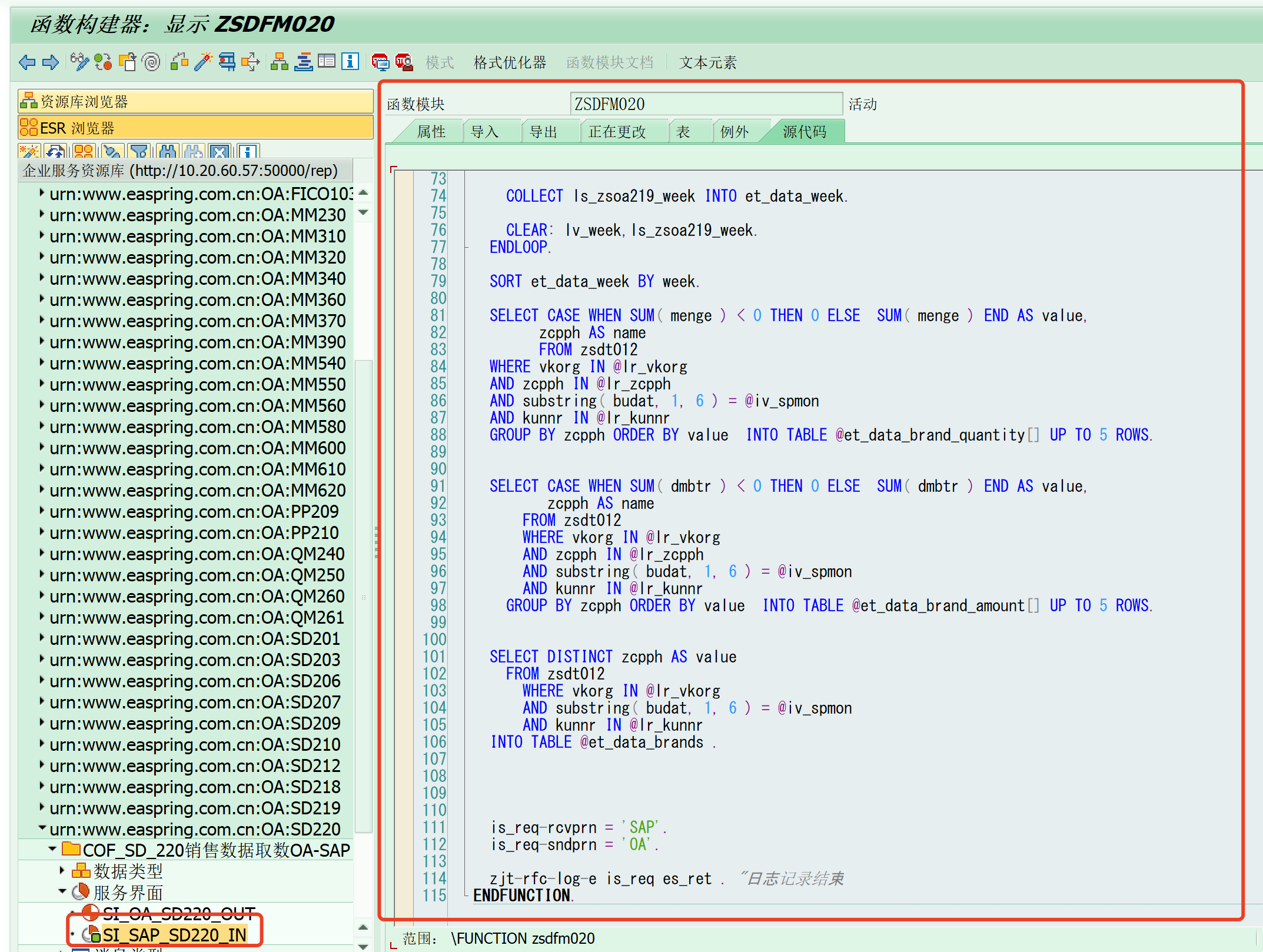


ID配置，这部分配置要注意跨域访问的设置，否则无法在部署后访问





# 第六步，ABAP

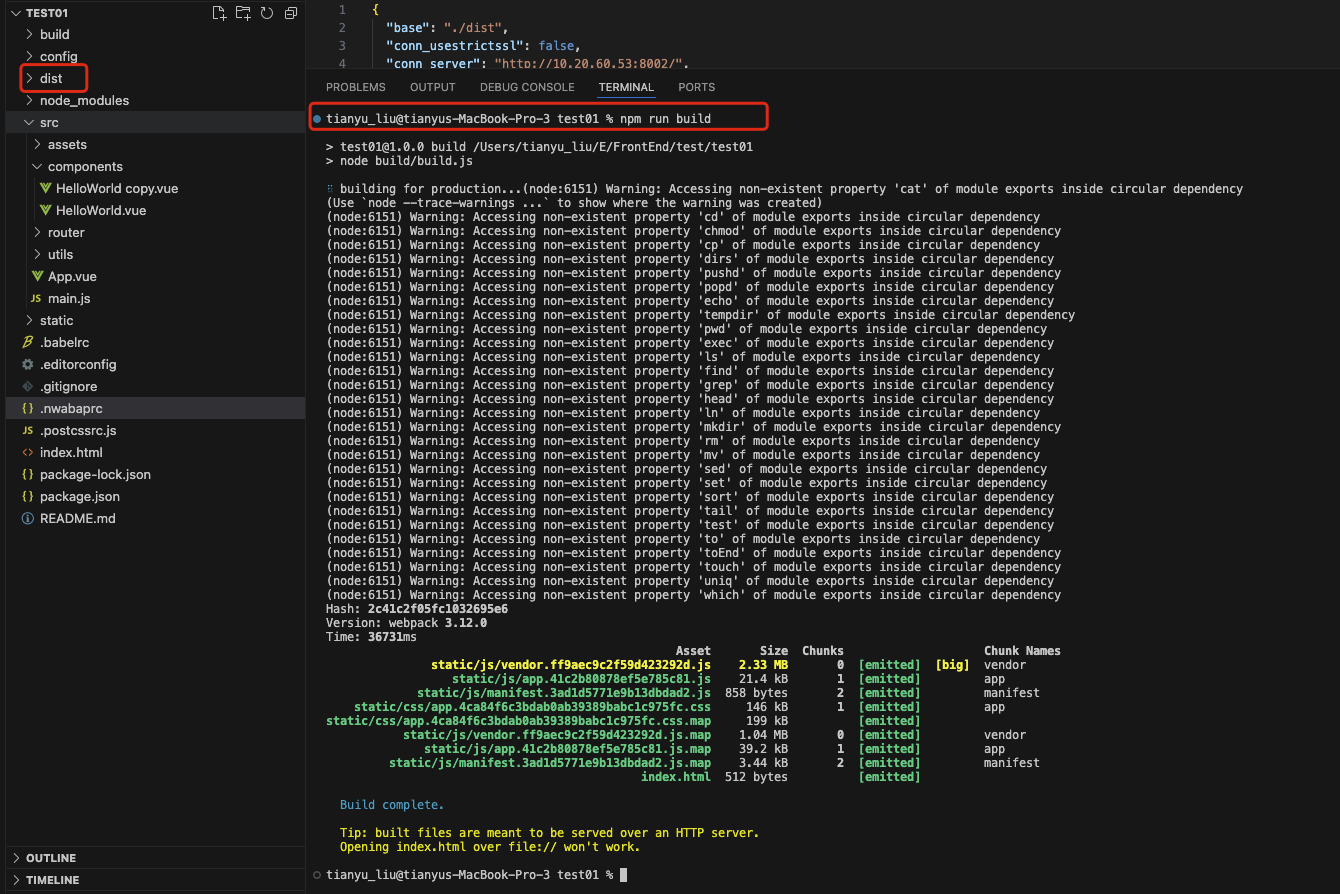


# 第七步 ，部署

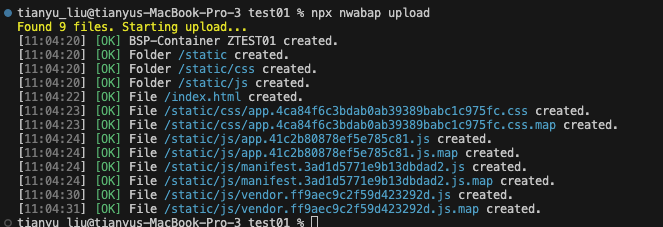
这步我们需要把前端代码部署到SAP的服务器上，体现到SAP的对象是BSP application

首先

首先执行npm run build执行完后目录中会多出来一个dist文件，dist文件就是我们要部署的文件

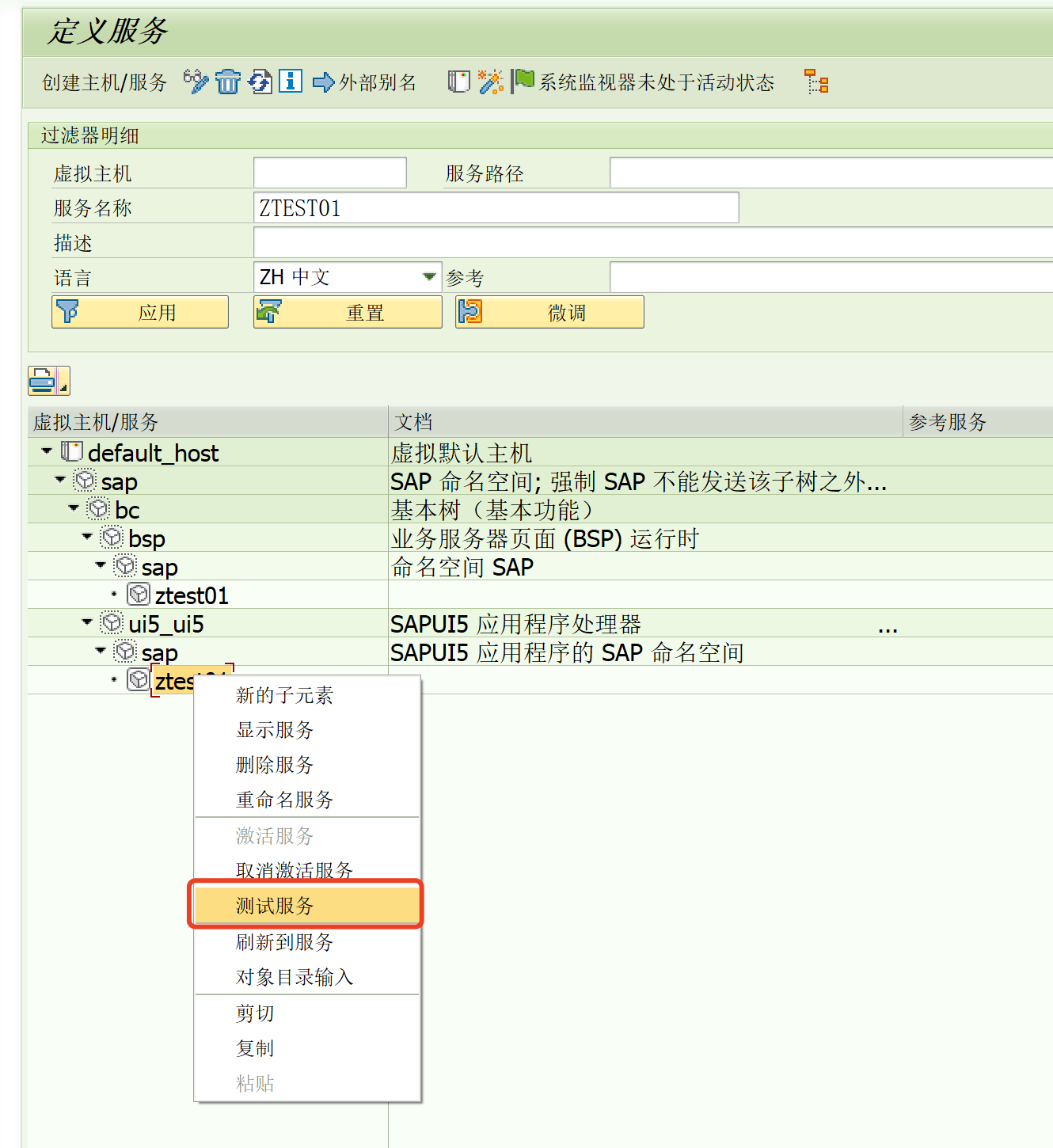


部署到SAP执行命令npx nwabap upload

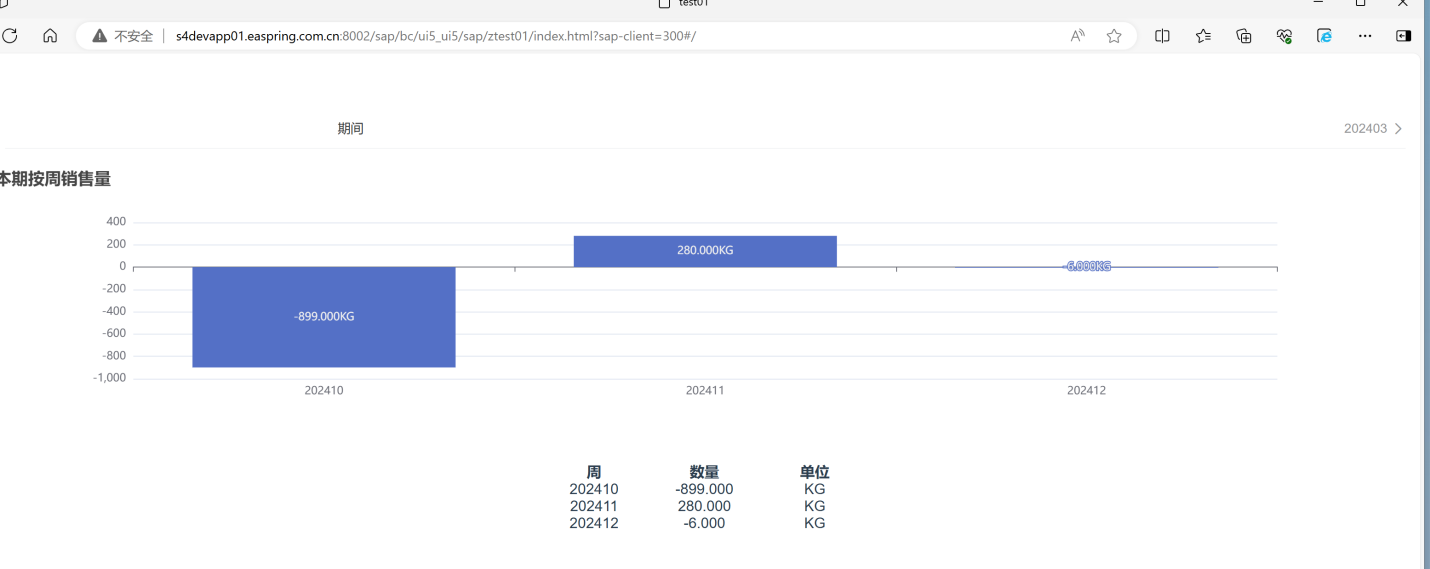


到SAP端激活并测试服务

SICF



测试结果如下





参考链接：

<https://youzan.github.io/vant/v2/#/zh-CN/>

<https://element.eleme.cn/#/zh-CN>

<https://echarts.apache.org/en/index.html>

<https://community.sap.com/t5/technology-blogs-by-members/how-to-create-a-vue-js-app-with-vs-code-and-deploy-to-sap-netweaver-bsp/ba-p/13549578>