1. Necessary environment for running the project：

(1) Node.js (Additional NPM)

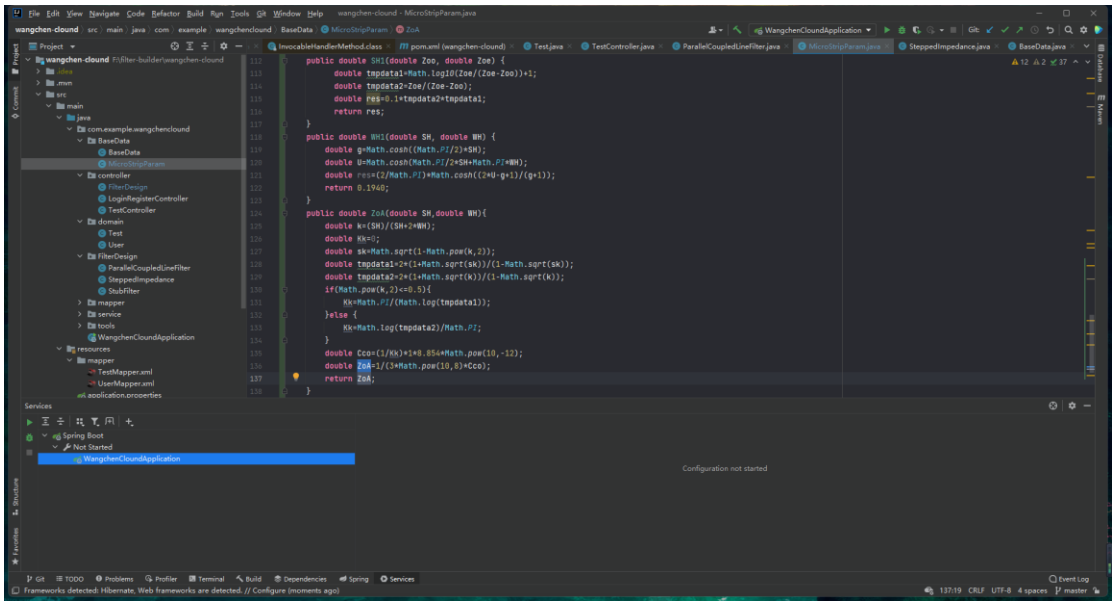
(2) Java1.8.

(3) Vscode

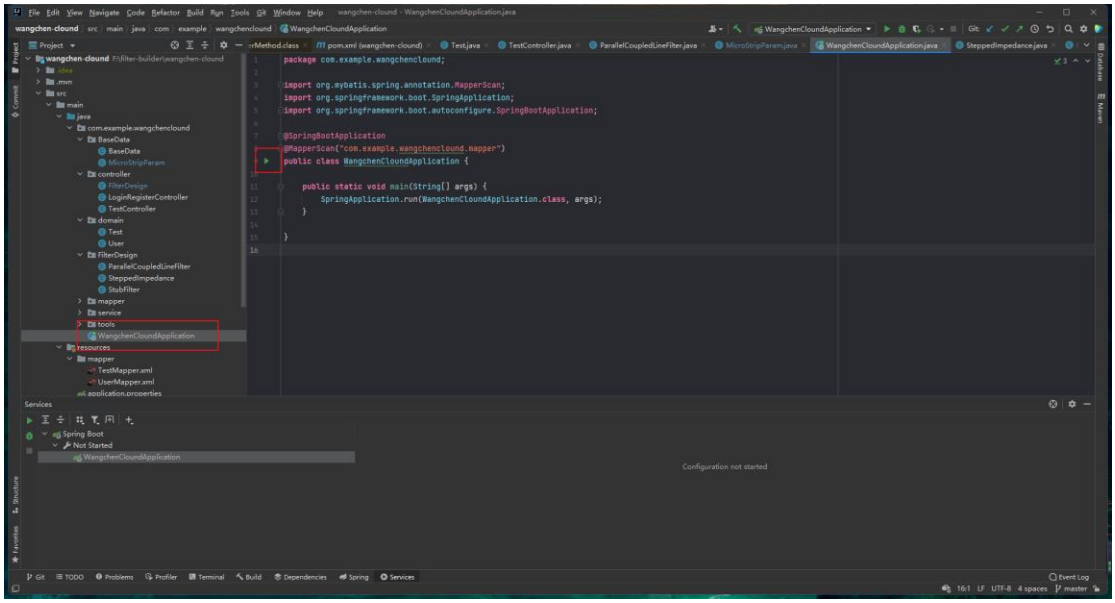
(4) IDEA

2. Start step instance：

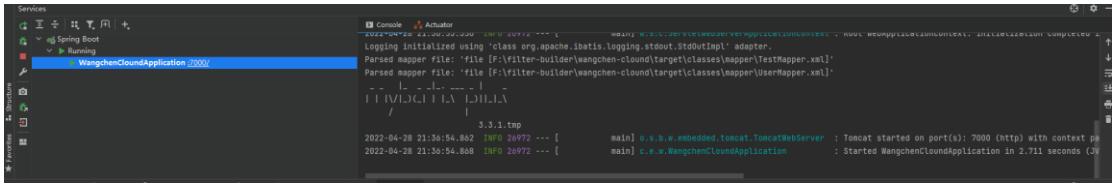
(1) Open sever file using IDEA. Wait for it to automatically download the required files.



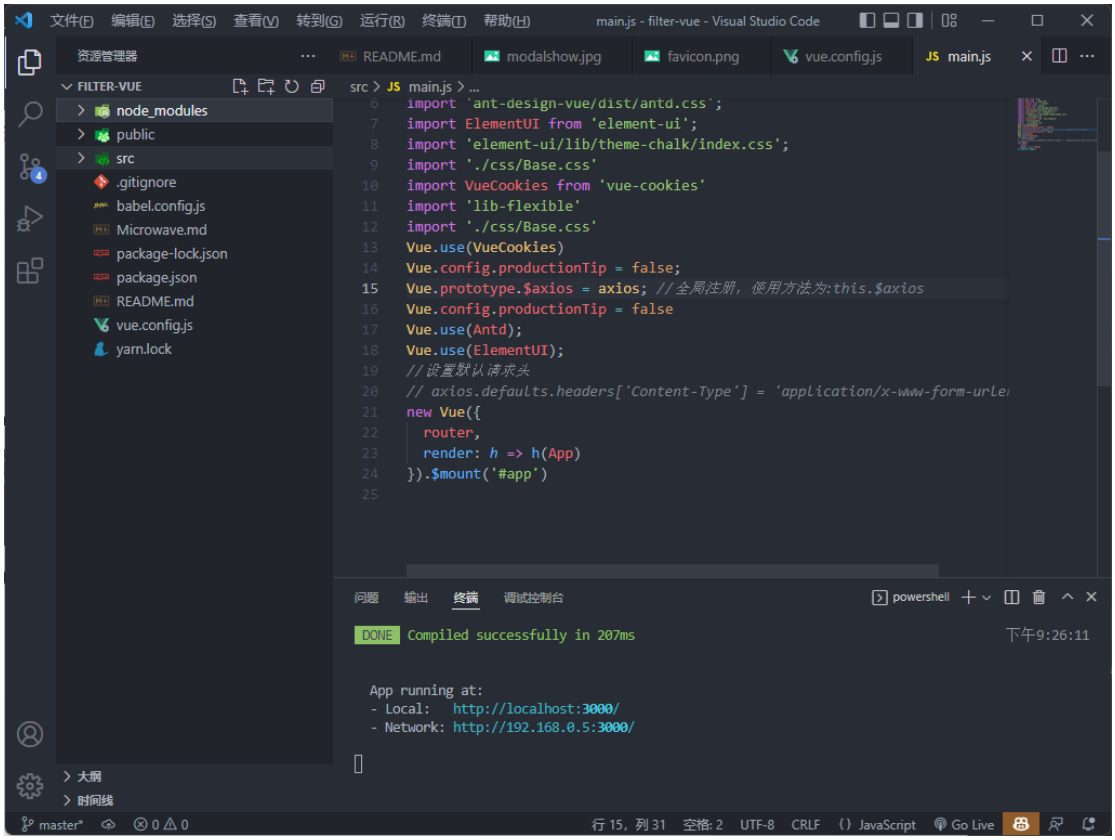
(2) Open the following file and click Run server.



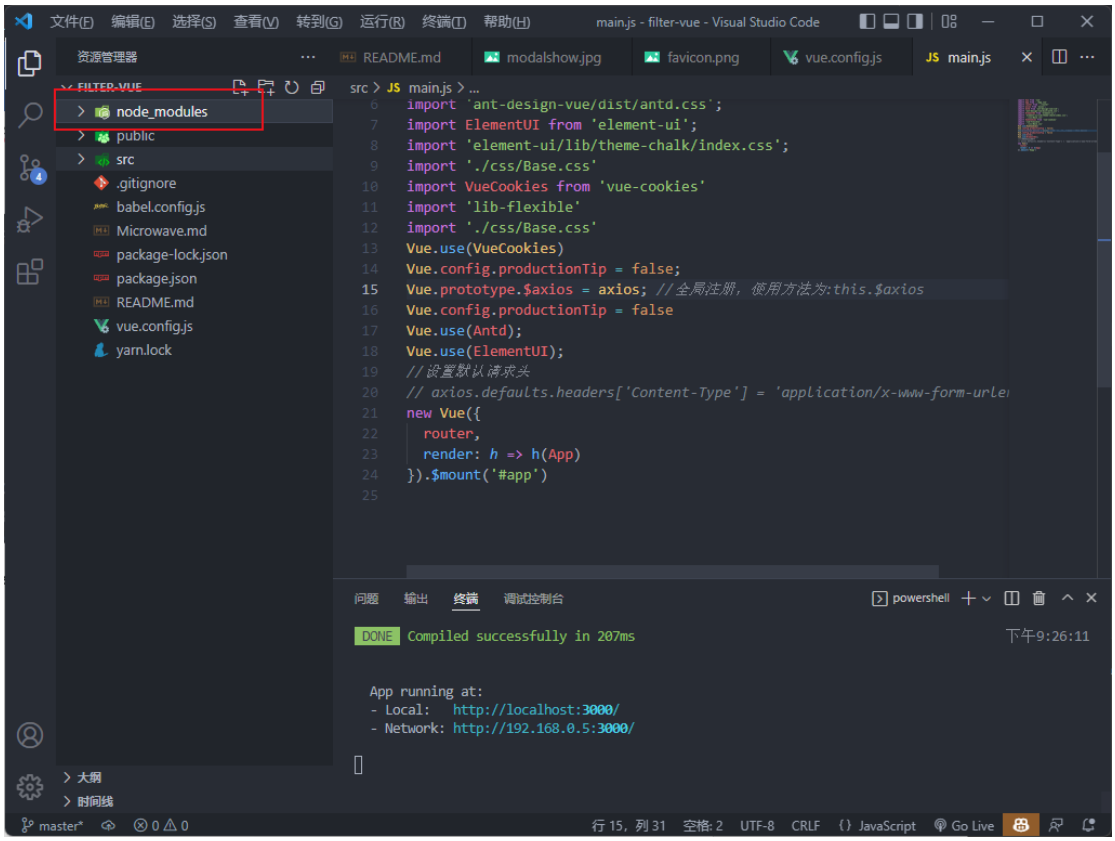
Run successfully



(3) Open the front-end file with vscode.

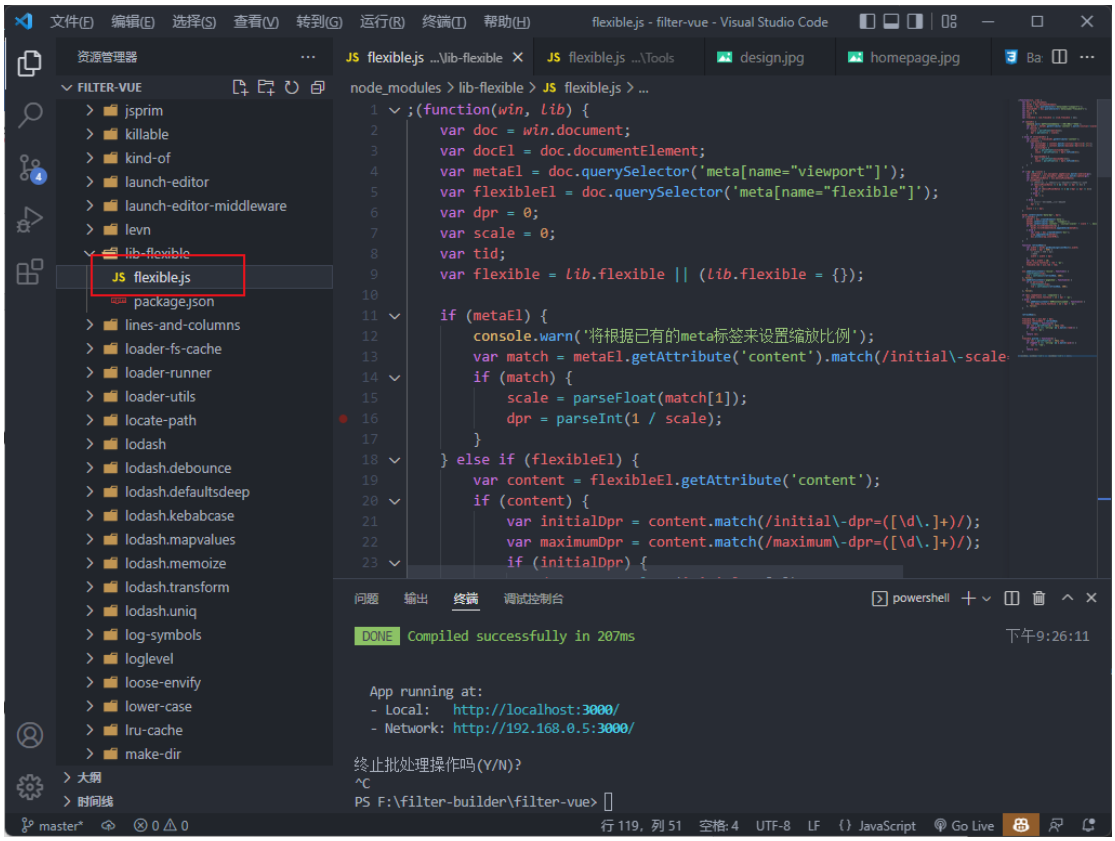


Delete first folder.



(4) Because of the path problem, the project file needs to be regenerated in the terminal. Enter NPM install and wait for the installation to complete.

(5) Modify the file(in node_modules) after installation:



Change to the following code:

```
;(function(win, lib) {
    var doc = win.document;
    var docEl = doc.documentElement;
    var metaEl = doc.querySelector('meta[name="viewport"]');
    var flexibleEl = doc.querySelector('meta[name="flexible"]');
    var dpr = 0;
    var scale = 0;
```

```
    var tid;
    var flexible = lib.flexible || (lib.flexible = {});

    if (metaEl) {
        console.warn('将根据已有的 meta 标签来设置缩放比例');
        var match = metaEl.getAttribute('content').match(/initial\-
scale=([\d\.]+)/);
        if (match) {
            scale = parseFloat(match[1]);
            dpr = parseInt(1 / scale);
        }
    } else if (flexibleEl) {
        var content = flexibleEl.getAttribute('content');
        if (content) {
            var initialDpr = content.match(/initial\-dpr=([\d\.]+)/);
            var maximumDpr = content.match(/maximum\-dpr=([\d\.]+)/);
            if (initialDpr) {
                dpr = parseFloat(initialDpr[1]);
                scale = parseFloat((1 / dpr).toFixed(2));
            }
            if (maximumDpr) {
                dpr = parseFloat(maximumDpr[1]);
                scale = parseFloat((1 / dpr).toFixed(2));
            }
        }
    }

    if (!dpr && !scale) {
        var isAndroid = win.navigator.appVersion.match(/android/gi);
        var isIPhone = win.navigator.appVersion.match(/iphone/gi);
        var devicePixelRatio = win.devicePixelRatio;
        if (isIPhone) {
            // iOS 下，对于 2 和 3 的屏，用 2 倍的方案，其余的用 1 倍方案
            if (devicePixelRatio >= 3 && (!dpr || dpr >= 3)) {
                dpr = 3;
            } else if (devicePixelRatio >= 2 && (!dpr || dpr >= 2)){
                dpr = 2;
            } else {
                dpr = 1;
            }
        } else {
            // 其他设备下，仍旧使用 1 倍的方案
            dpr = 1;
        }
```

```
        scale = 1 / dpr;
    }

    docEl.setAttribute('data-dpr', dpr);
    if (!metaEl) {
        metaEl = doc.createElement('meta');
        metaEl.setAttribute('name', 'viewport');
        metaEl.setAttribute('content', 'initial-scale=' + scale + ',
maximum-scale=' + scale + ', minimum-scale=' + scale + ', user-
scalable=no');
        if (docEl.firstElementChild) {
            docEl.firstElementChild.appendChild(metaEl);
        } else {
            var wrap = doc.createElement('div');
            wrap.appendChild(metaEl);
            doc.write(wrap.innerHTML);
        }
    }

    function refreshRem(){
        var width = docEl.getBoundingClientRect().width;
        if (width / dpr > 540) {
            // width = 540 * dpr;
            //变更
            width = width * dpr;
        }
        var rem = width / 10;
        docEl.style.fontSize = rem + 'px';
        flexible.rem = win.rem = rem;
    }

    win.addEventListener('resize', function() {
        clearTimeout(tid);
        tid = setTimeout(refreshRem, 300);
    }, false);
    win.addEventListener('pageshow', function(e) {
        if (e.persisted) {
            clearTimeout(tid);
            tid = setTimeout(refreshRem, 300);
        }
    }, false);

    if (doc.readyState === 'complete') {
        doc.body.style.fontSize = 12 * dpr + 'px';
```

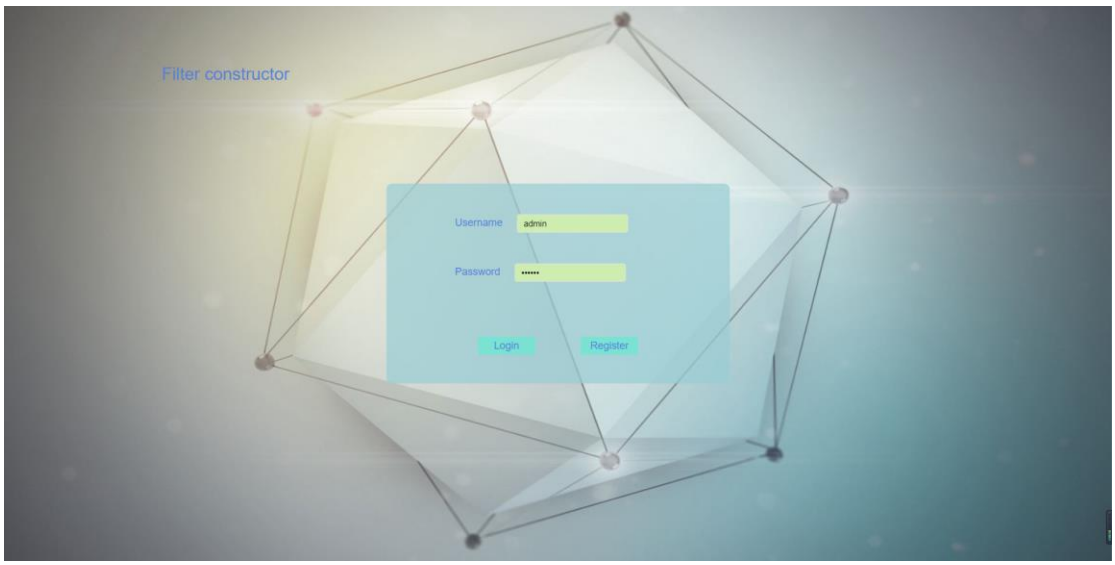
```
    } else {
        doc.addEventListener('DOMContentLoaded', function(e) {
            doc.body.style.fontSize = 12 * dpr + 'px';
        }, false);
    }


    refreshRem();

    flexible.dpr = win.dpr = dpr;
    flexible.refreshRem = refreshRem;
    flexible.rem2px = function(d) {
        var val = parseFloat(d) * this.rem;
        if (typeof d === 'string' && d.match(/rem$/)) {
            val += 'px';
        }
        return val;
    }
    flexible.px2rem = function(d) {
        var val = parseFloat(d) / this.rem;
        if (typeof d === 'string' && d.match(/px$/)) {
            val += 'rem';
        }
        return val;
    }

})(window, window['lib'] || (window['lib'] = {}));
```

(6) Enter npm run serve in the terminal

```
终止批处理操作吗(Y/N)?
^C
PS F:\filter-builder\filter-vue> npm run serve
```

Press enter

(7) Press and hold Ctrl and click the first address to enter the page.



3. Introduction to code

1 . Introduction to language tools

1. For server I use springboot (a special type of java). Its main function is to establish an HTTP connection interface. Process the HTTP request sent by the client and respond.

For example:

GetMapping Indicates the access address of the interface(API). When we run the server we can see the port of the server is

```
plication :7000/
```

And because we run the project on our local host. The address will be

localhost:7000 and because the address of API is

```
@GetMapping("/filterDesign/steppedImpedance")
```

If we want to send data to this API we should sent to the address

localhost:7000/filterDesign/steppedImpedance.

Then the server will give a http response for our webpage.

```
                jsonObject.put("LengthArray", LengthArray);
                jsonObject.put("ZArray", ZArray);
                jsonObject.put("R0", R0);
                jsonObject.put("overflow", "false");
            } else {
                jsonObject.put("overflow", "true");
            }
        }
        return jsonObject;
    }
}
```

2. For front page I use vue2 (a popular frame of javaScript and Html).

For example:

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)

StepImpShow.vue - filter-vue - Visual Studio Code

资源管理器

FILTER-VUE

- node_modules
- public
- src
  - api
  - assets
  - components
    - FilterType
      - StepImpParam.vue U
      - StepImpShow.vue U
      - HomePage.vue U
    - Login.vue U
    - TypeDesign.vue U
  - css
  - media\imags
    - Cp.png U
    - design.jpg U
    - FinalSection.png U
    - homepage.jpg U
    - Id.png U
  - router
  - App.vue M
  - main.js M
- .gitignore
- babel.config.js
- Microwave.md U
- package-lock.json U
- package.json M
- README.md M
- vue.config.js U
- yarn.lock M

TypeDesign.vue U | StepImpParam.vue U | StepImpShow.vue U | FinalSection.png U | Id.png U | Cp.png U | Login.vue U | HomePage.v

src > components > FilterType > StepImpShow.vue > {} "StepImpShow.vue" > style > .table-size:hover

```vue
<template>
  <div>
    <!-- 数据表(归一个元件值) -->
    <div class="clearTopMargin"></div>
    <div class="standard">
      <div class="self-card table-size">
        <div class="card_title">Data Table[N(Steps)={{ Step }}]</div>
        <div>
          <a-table :columns="columns" :data-source="data">
            <a slot="gi" slot-scope="text">{{ text }}</a>
            <p slot="β1" slot-scope="text">{{ text }}*</p>
          </a-table>
        </div>
      </div>
    </div>
    <!-- 原型电路展示 -->
    <div class="self-card circuit-show">
      <div class="card_title">Prototype Circuit</div>
      <div class="circuit-component" v-for="(item, index) in data" :key="index">
        <div v-if="index % 2 == 0">
          <div class="tips">{{ index + 1 }}(C={{ item.ComponentValue }})</div>
          <img class="circuit-C" src="../../media/imags/Cp.png" />
        </div>
        <div v-if="index % 2 == 1">
          <div class="tips">{{ index + 1 }}(L={{ item.ComponentValue }})</div>
          <img class="circuit-L" src="../../media/imags/Id.png" />
        </div>
      </div>
    </div>
    <!-- //最终模型 -->
    <div class="self-card last-modal">
      <div class="card_title">Final Circuit</div>
      <div class="lost-component">
        <div class="last-tips">Z={{ DesignParam.R0 }}Ω</div>
        <img class="final-section" src="../../media/imags/FinalSection.png" />
      </div>
      <div class="lost-component" v-for="(item, index) in data" :key="index">
```

```vue
</template>

<script>
export default {
  name: "StepImpShow",
  props: ["DesignParam"],
  data() {
    return {
      Step: 0,
      columns: [
        {
          title: "gi",
          dataIndex: "gi",
          key: "gi",
          scopedSlots: { customRender: "gi" },
        },
        {
          title: "Component value",
          dataIndex: "ComponentValue",
          key: "ComponentValue",
          width: 110,
        },
        {
          title: "β*l",
          dataIndex: "EleLength",
          key: "EleLength",
          scopedSlots: { customRender: "β1" },
          //   width: 80,
        },
        // {
        //   title: "L",
        //   dataIndex: "Length",
        //   key: "Length",
        //   width: 80,
```

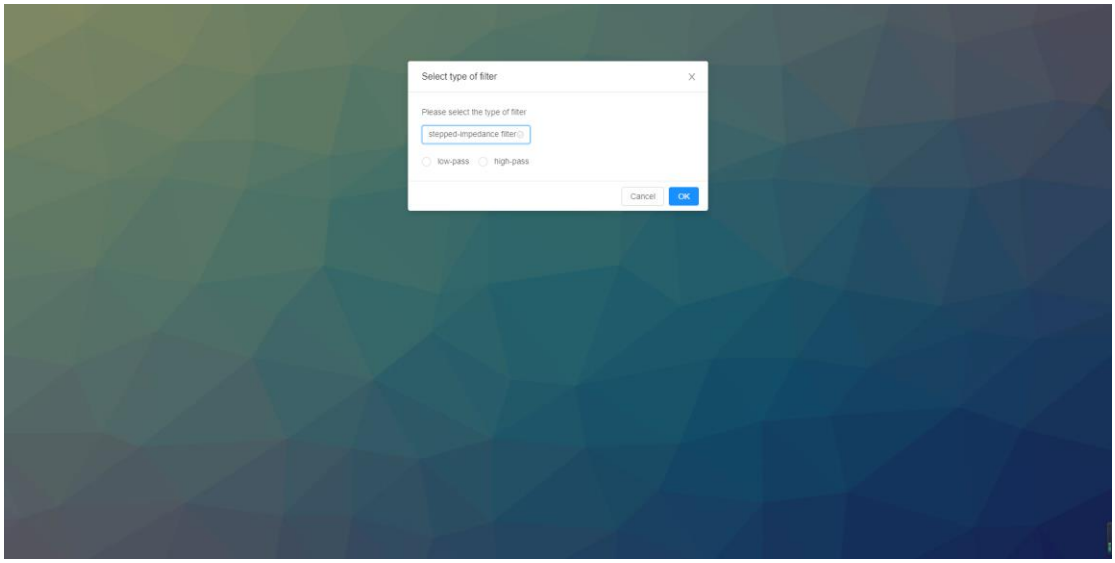
```
</script>

<style scoped>
.standard {
  display: inline-block;
  height: 500px;
}
.clearTopMargin {
  height: 70px;
  /* margin-top: -100px; */
}
.table-size {
  display: inline-block;
  /* margin-top: 20px; */
  margin-left: 50px;
  width: 400px;
  padding-left: 40px;
  padding-right: 40px;
  transition: all 0.7s;
}
.table-size:hover {
  transform: scale(1.2);
}
.circuit-show {
  display: inline-block;
  margin-left: 100px;
  padding: 20px;
  height: 350px;
  transition: all 0.7s;
  vertical-align: top;
}
.circuit-show:hover {
```

Using this frame. We should write html page In <template> label and write javascript in <script> label . Write css code in <style> label. And then I use axios (a good tools for http connection) to send and receive http message. Just like this one:

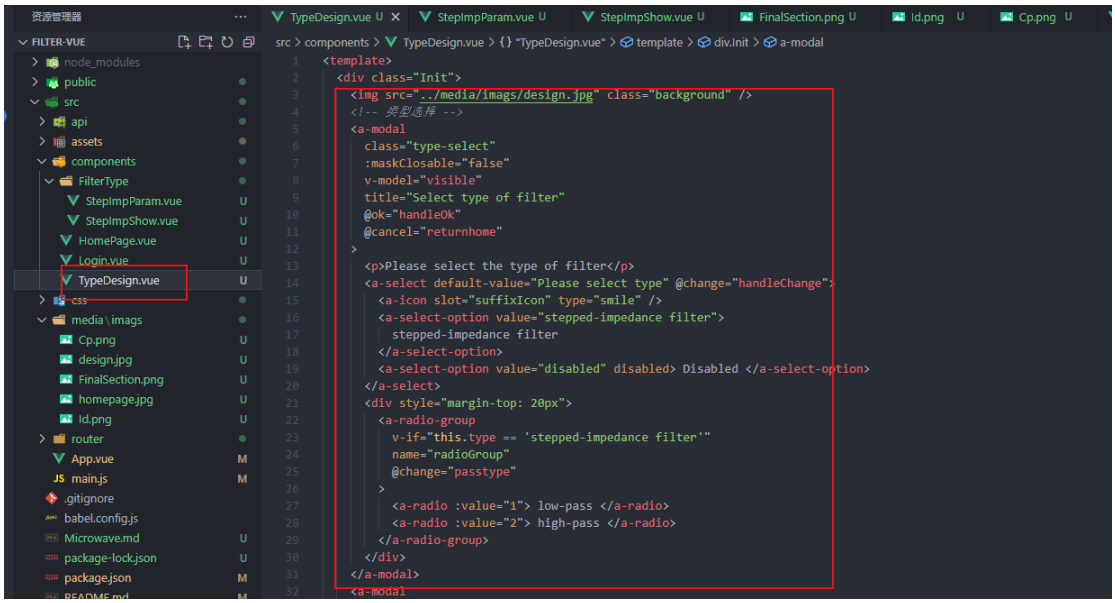
```
    }

    let url = "app/filterDesign/steppedImpedance";
    getAction(url, param)
      .then((res) => {
        console.log(res);
        if (res.data.overflow == "true") {
          this.$message.warning("Out of calculable range.Please try again.
          setTimeout(() => {
            location.reload();
          }, 3000);
        } else {
          console.log(res.data);
          this.$emit("getDesignParam", res.data);
          this.$emit("showFilter");
        }
      })
      .catch((err) => {
        console.log(err);
      });
  },
```

2．Analysis of the whole process of software operation

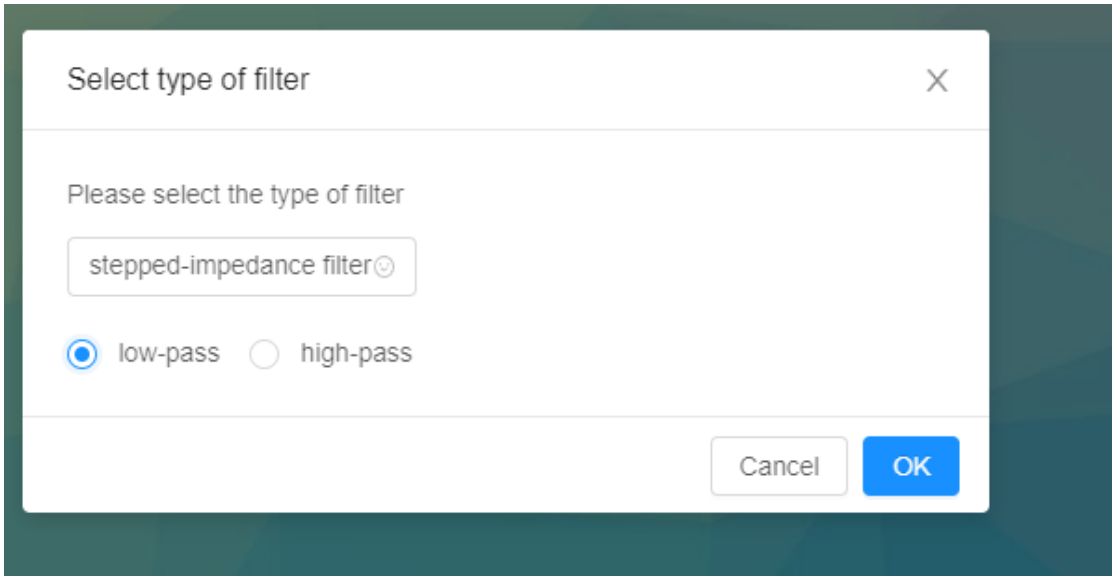
1. Enter the software design page



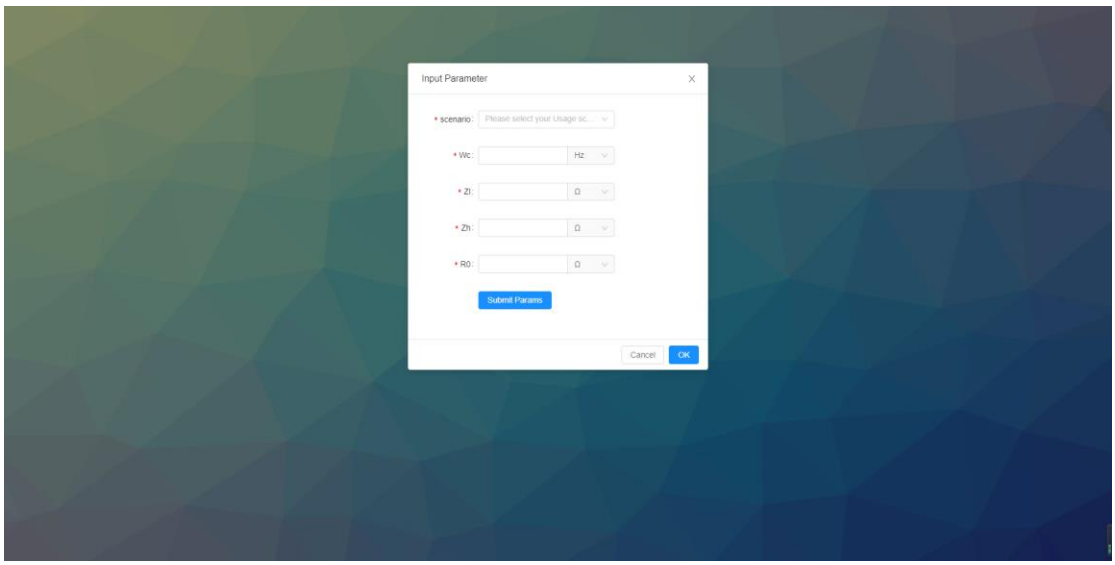
The corresponding vue code is this one:



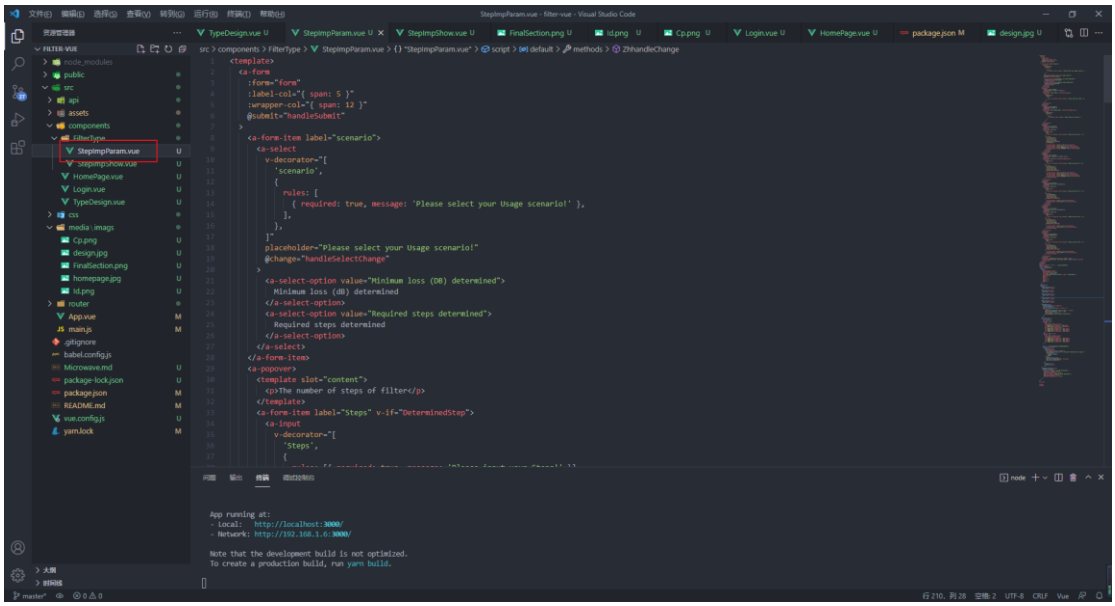
And the we select the type



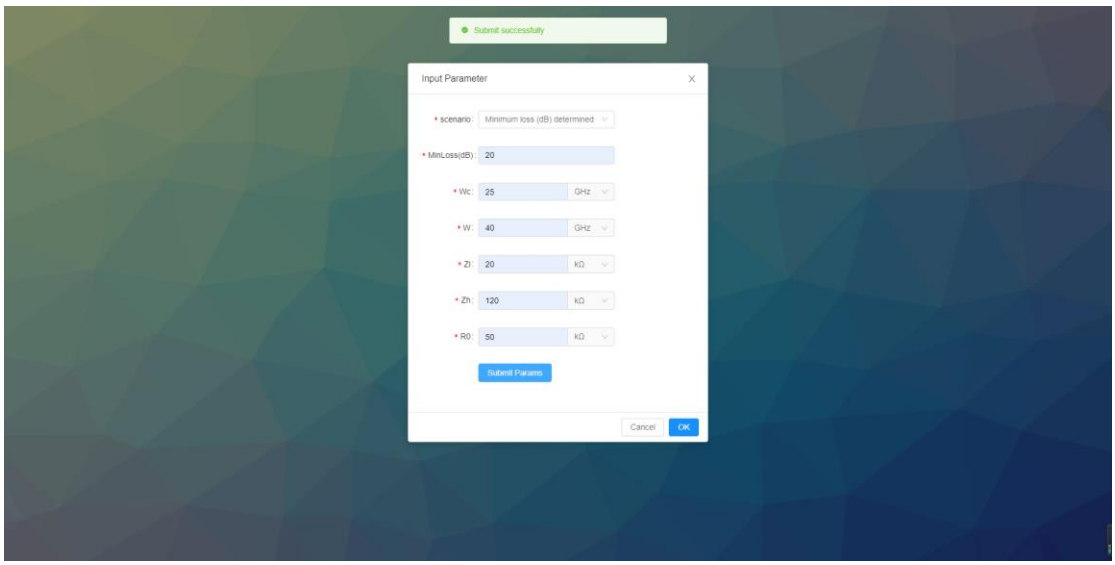
Click Ok go next:



The corresponding vue code is this one:



When we finish the input and click submit



The system will show us we have submitted successful and the we have store the data in our vue code:

```
<a-form-item :wrapper-col="{ span: 12, offset: 5 }">
  <a-button type="primary" html-type="submit"> Submit Params </a-button>
</a-form-item>
```

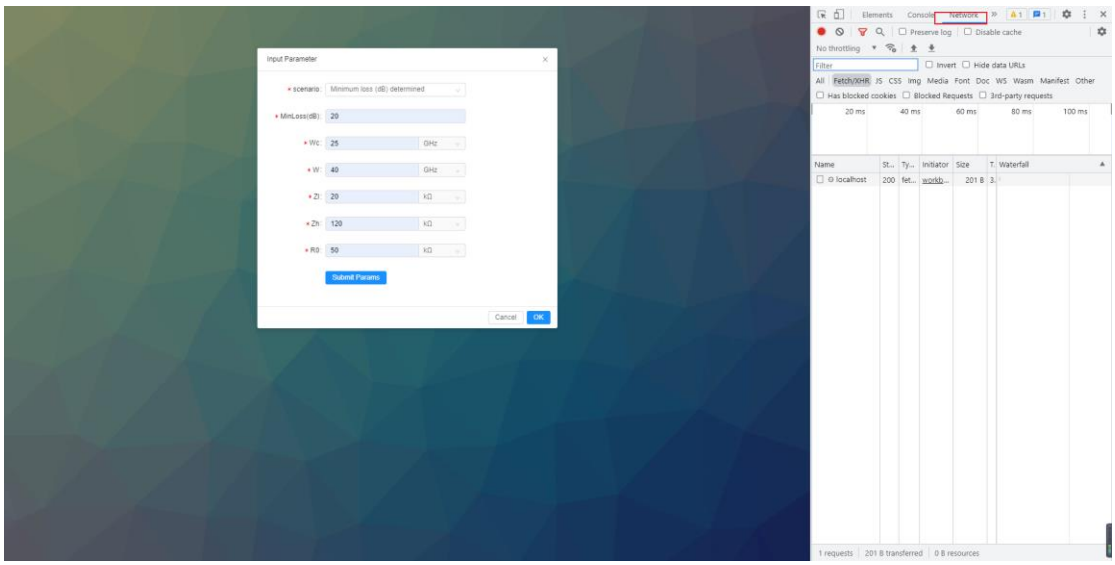
We will click the submit button we will enter the function:

```
},
handleSubmit(e) {
  console.log(this.$refs.WcSelect);
  e.preventDefault();
  this.form.validateFields((err, values) => {
    if (!err) {
      console.log("Received values of form: ", values);
      this.$emit("getParam", values, true);
      this.formData = values;
      this.$message.success("Submit successfully");
    }
  });
},
```

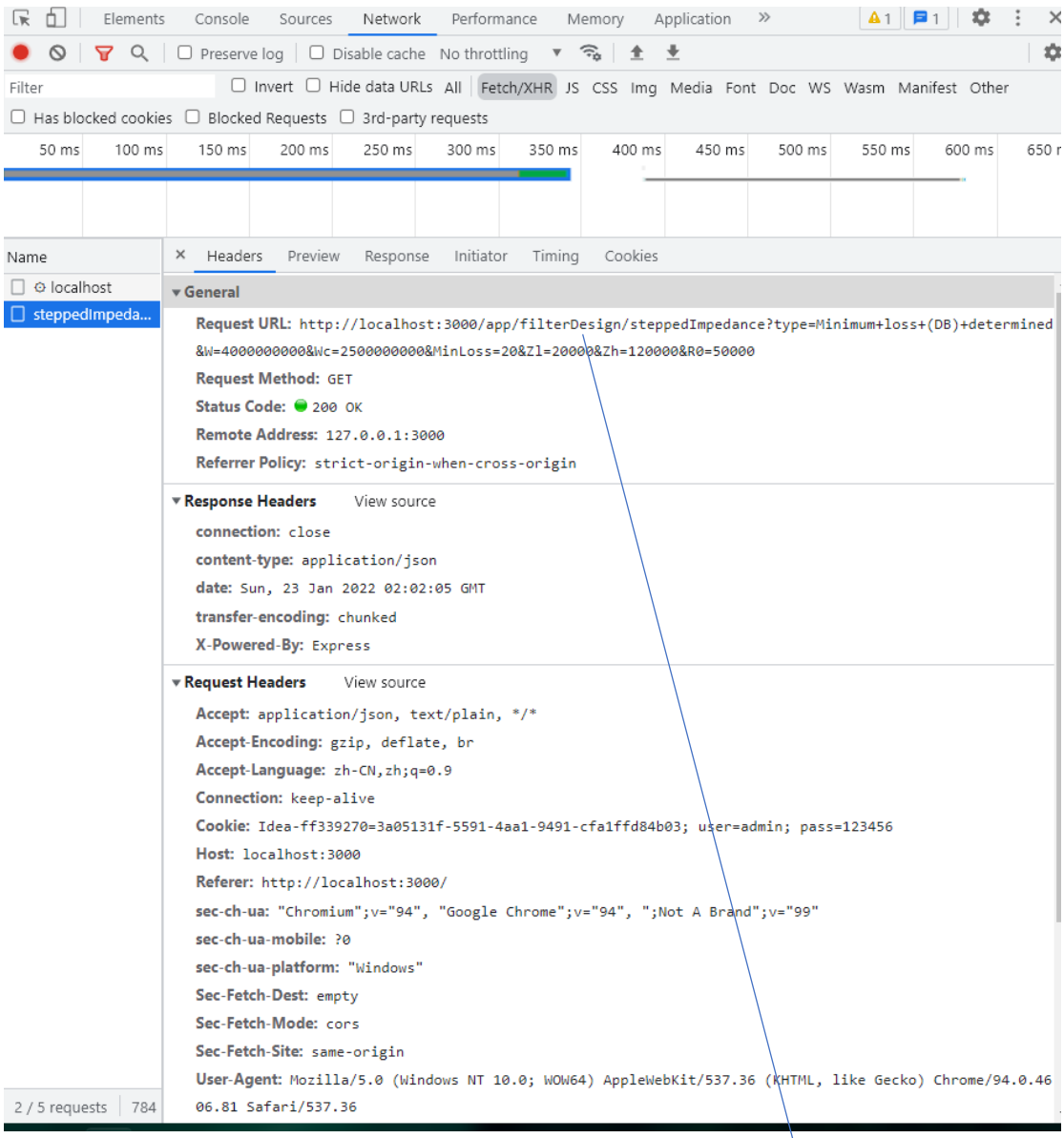
The parameter e means what we have input for filter and the store these data to this.formdata. and give a successful tips.

Then we should click ok button to get the modal of filter, before we click button we can just click F12 in webpage to open the browser console like this

(I use Google Chrome):



And click network to watch the http request we sent.



This is our http request and we can see the address is localhost:3000/…../….

It is different form what I have showed

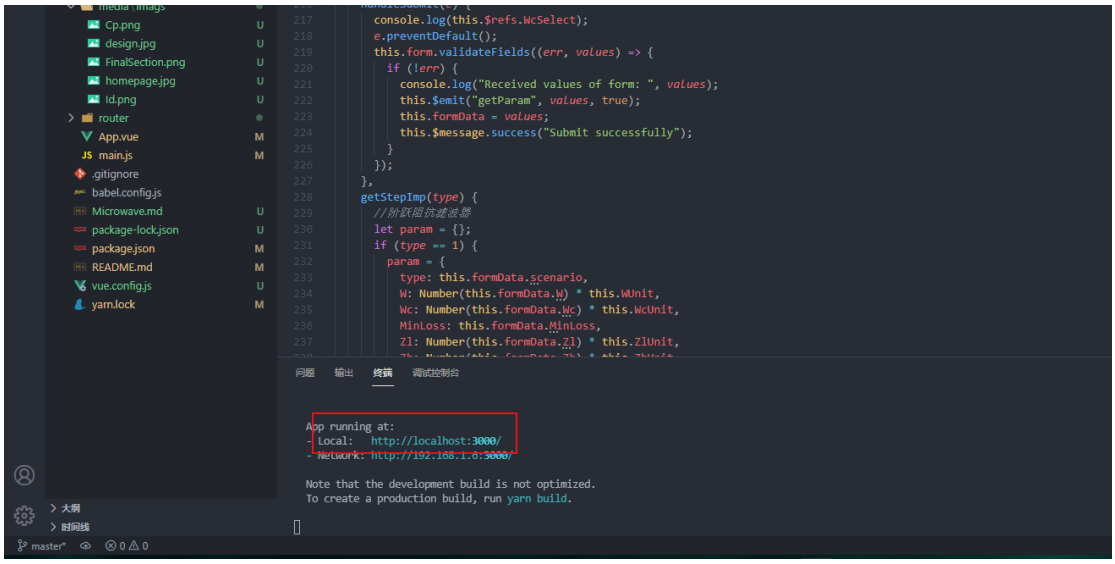
localhost:7000/filterDesign/steppedImpedance.

The port is different. One is 3000 and the other one is 7000. A front-end technology called server proxy is used here.

Tips:

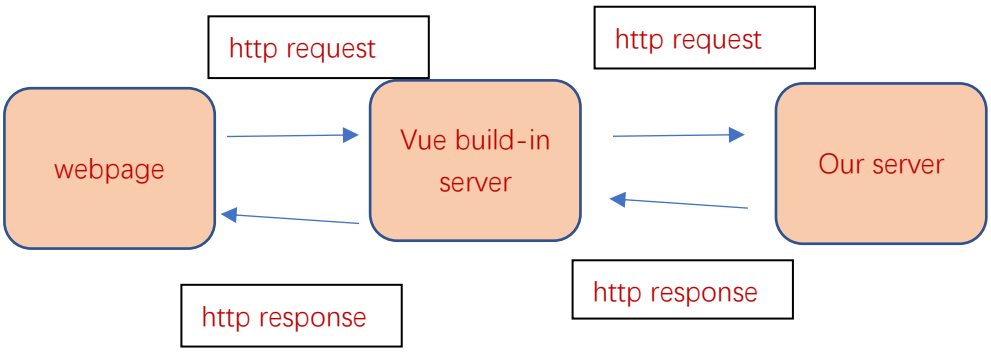
(Here we need to talk about HTTP communication first. Because we use front and rear end separation. For Vue, it has its own built-in server and

its address is localhost of 127.0.0.1 and port is 3000.



For our server, the port is 7000 Therefore, when Vue sends a request to the server, it will be intercepted because we violate the cross domain principle. That is, the domain name and port between the two hosts of HTTP communication must be consistent, so the server proxy technology is used here.)

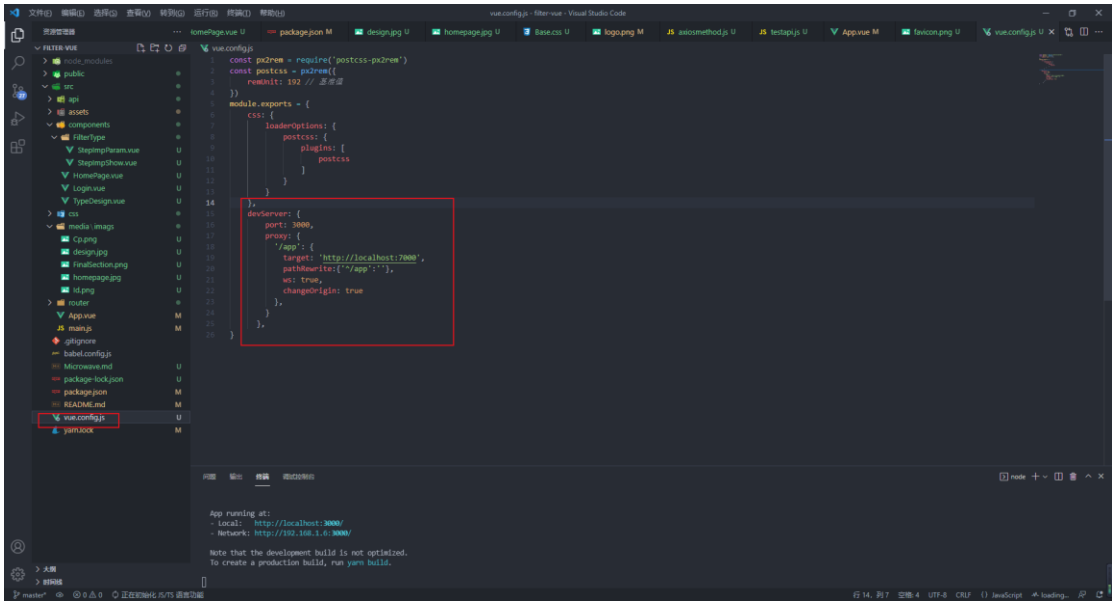
Proxy just like this:



The principle is that there is no cross domain restriction on the communication between servers, but the browser has restrictions on the requests of the server, so we first send our requests to Vue's own built-in

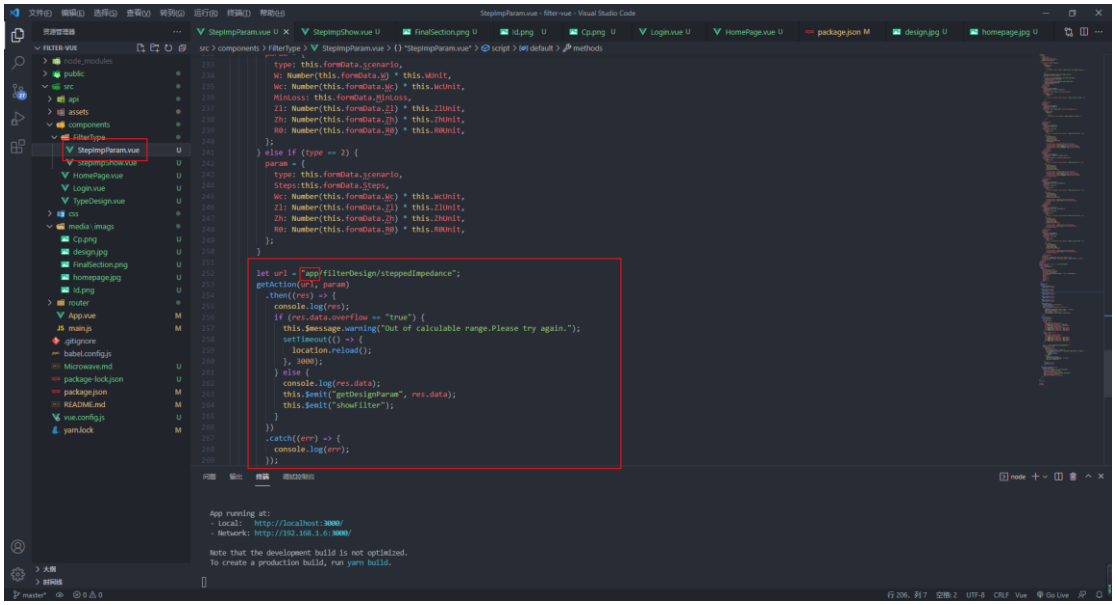
server, and then the server acts as an intermediate medium to assist us in communication.

The code is here:



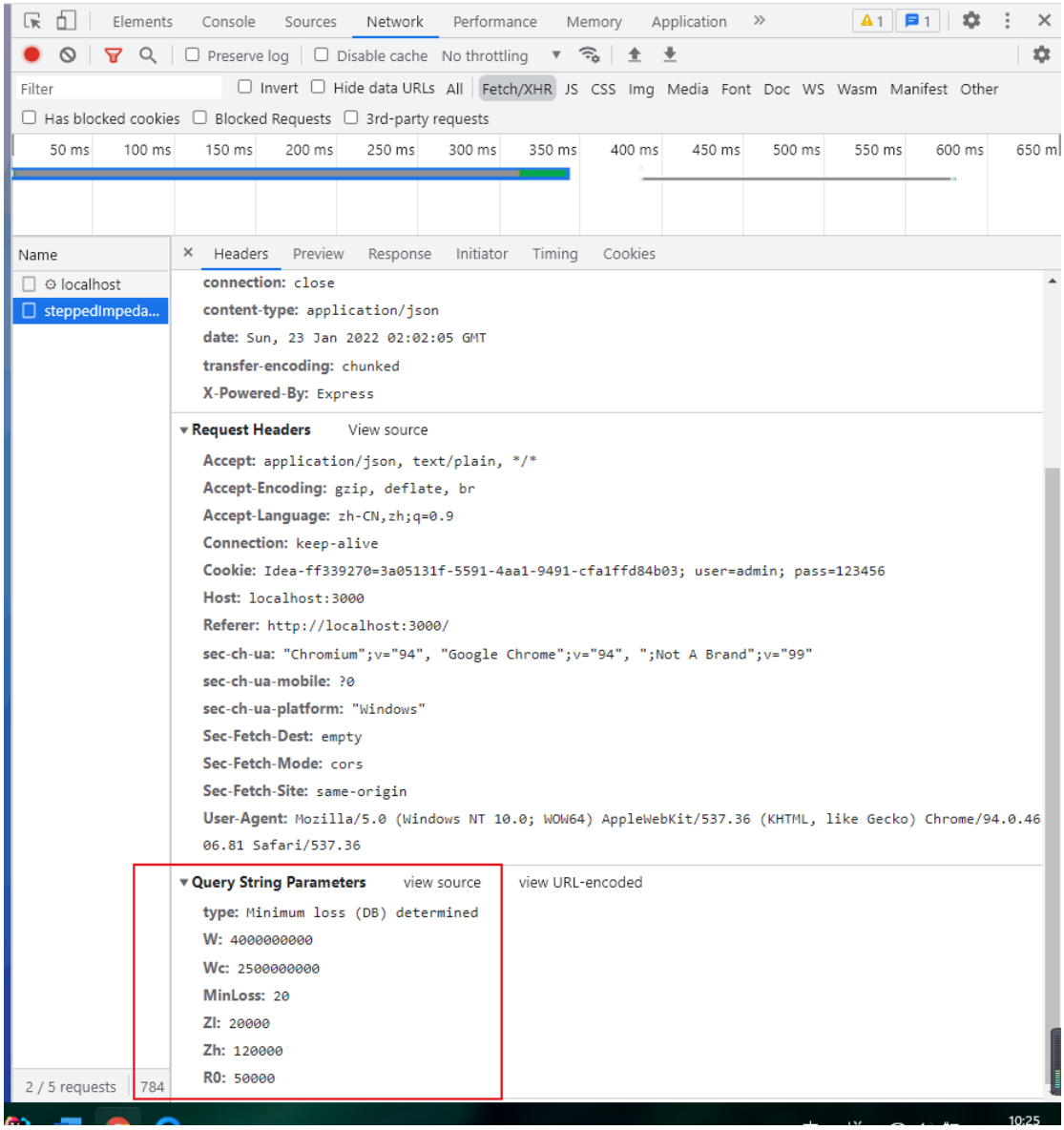
It means if we send http request to address app then it will send to localhost:7000.

So the proxy is finished and the vue code of send data is here:



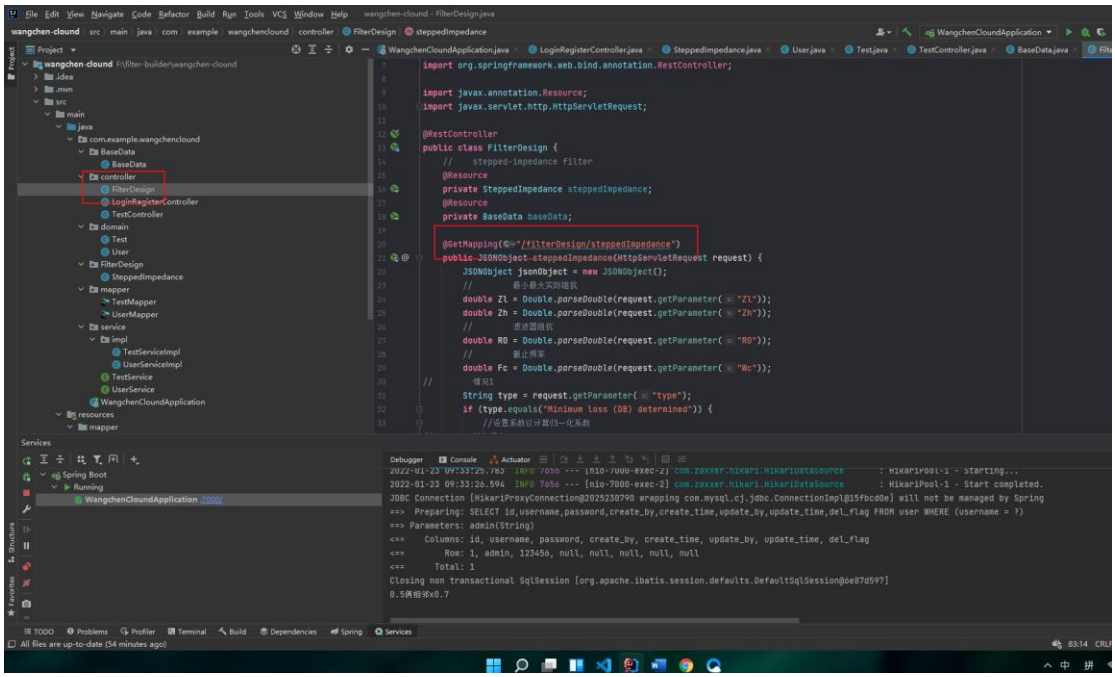
We can see we send data to app/…/…

We return to the request sent again:

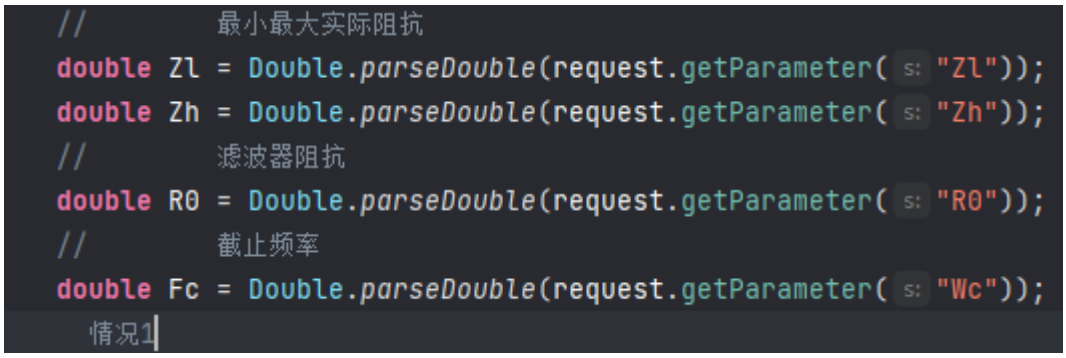


These are the parameter of filter.

The we look at server:



This is the API for this http request.



These codes are just mean we get the parameters from http request and

(Zl: Minimum actual resistance.

Zh: Maximum actual resistance

Fc: Cut-off frequency

R0: Filter impedance

F: Frequency at which insertion loss is minimum

Type: 1 and 2 mean different case of design I have introduced.

)

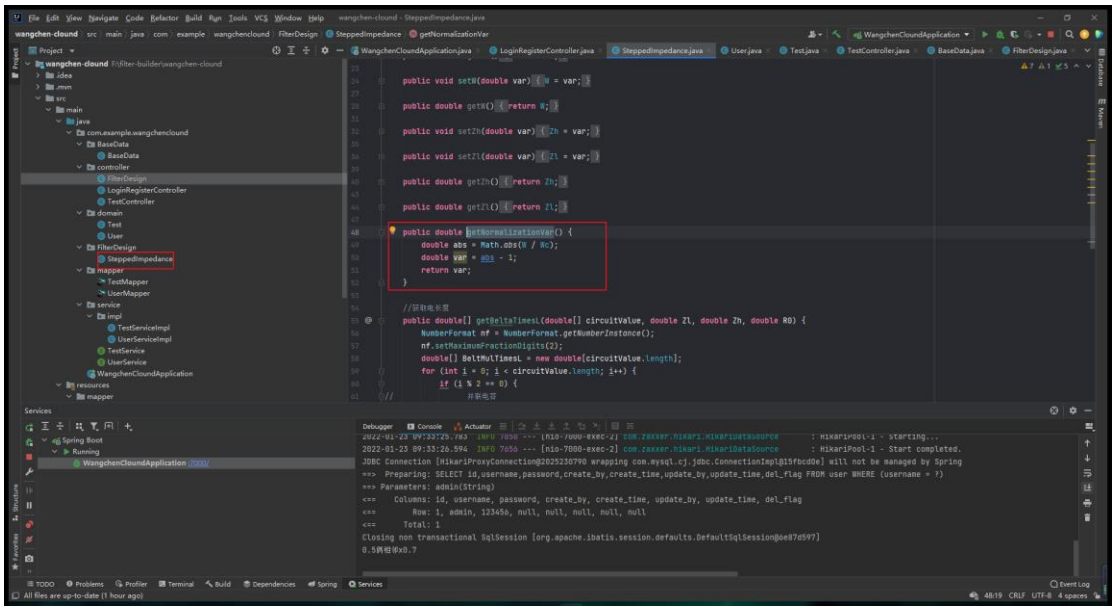
Then the first step we should get the normalization constant.

```
steppedImpedance.setW(F);
steppedImpedance.setWc(Fc);
double nomvar = steppedImpedance.getNormalizationVar();
```

This is just the formular here:

$$\frac{\omega}{\omega_c} - 1 = \frac{4.0}{2.5} - 1 = 0.6$$

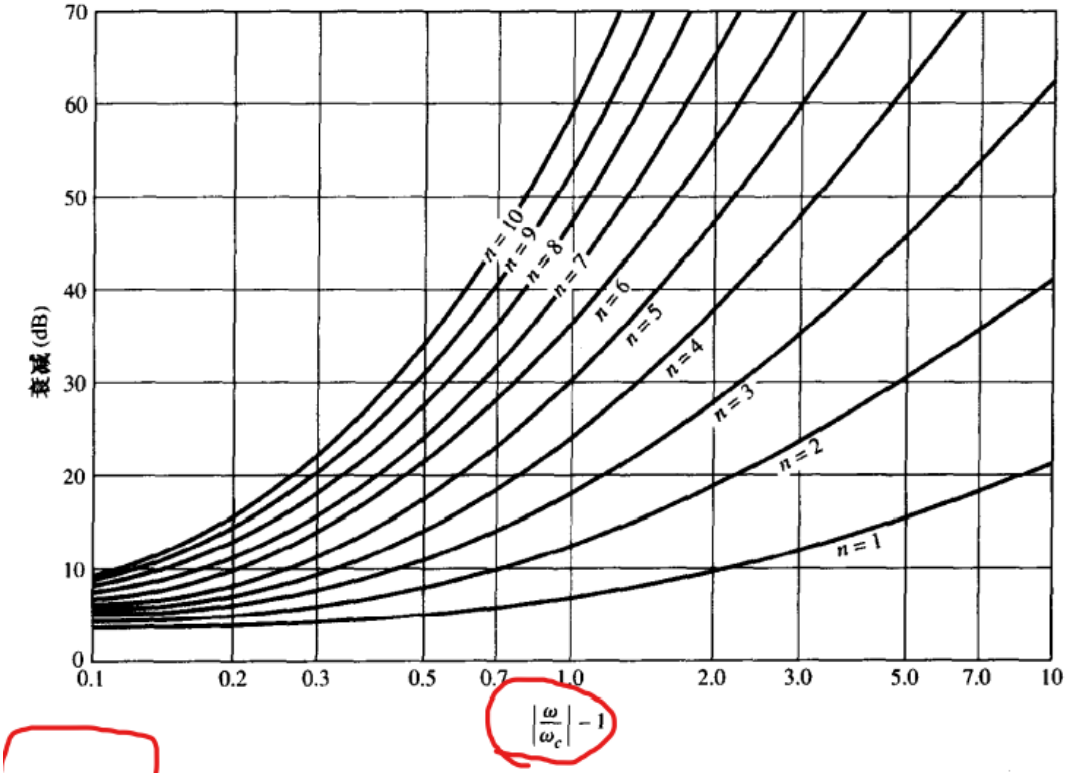
We can see the details about this method here:



It's easy and then we have get the normalization constant nomvar

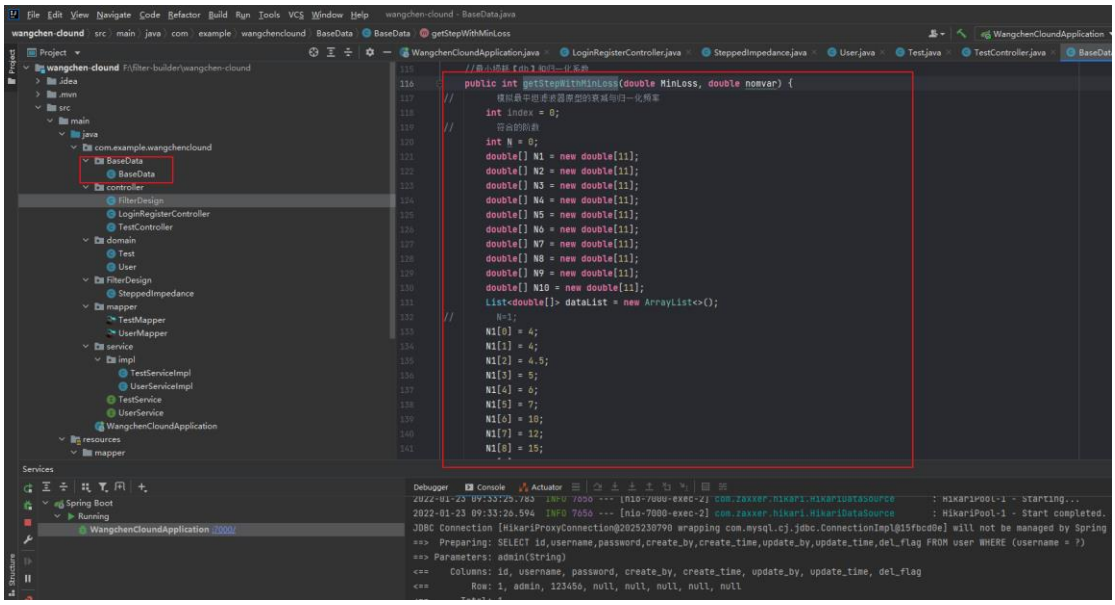
Then we should get the number of steps for our filter:

Like this:



We need 20dB and we get normalization constant equals 0.6. So we go to this function:

```
int N = baseData.getStepWithMinLoss(minLoss, nomvar);
```



In principle, as I said at that time, the curves in different intervals are calculated as straight lines. You can get very accurate values. Just give an

example for 0.6:

```
336                 }
337             }
338             int templength = tmpArray.length;
339             for (int i = 0; i < templength; i++) {
340                 if (tmpArray[i] > MinLoss + 1) {
```

Breakpoint here and debug:

```
ay[i] = lowArray[i];   lowArray: [5.0, 8.0, 11.0, 15.0, 18.0, 22.0, 24.0, 27.0, 31.0, 34.0]

= tmpArray.length;   tmpArray: [5.500000000000001, 9.000000000000002, 12.000000000000002, 16.5, 20.000
; i < templength; i+   tmpArray = {double[10]@7577} [5.500000000000001, 9.000000000000002, 12.000000000... View
ay[i] > MinLoss + 1)      0 = 5.500000000000001
+ 1;                      1 = 9.000000000000002
                          2 = 12.000000000000002
                          3 = 16.5
                          4 = 20.000000000000004
                          5 = 25.000000000000004
                          6 = 28.000000000000004
                          7 = 31.000000000000004
                          8 = 35.50000000000001
                          9 = 39.50000000000001

Set Value  F2   Create Renderer   Add as Inline Watch

Variables
this = {BaseData@7578}
MinLoss = 20.0
```

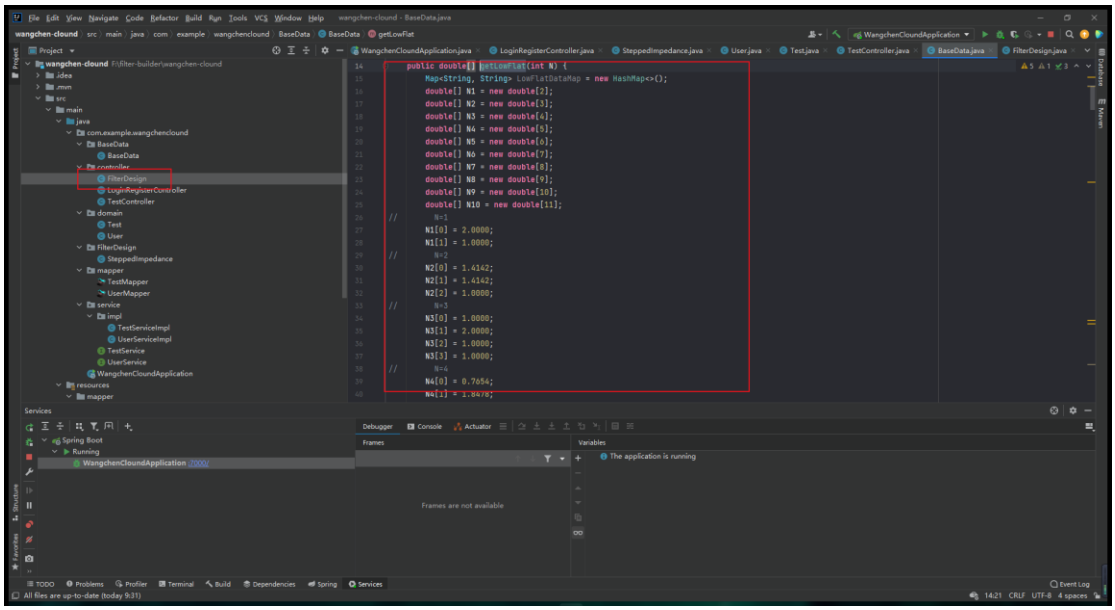
The result calculated by our code. After comparison, we can find that we have obtained very accurate values. And then the code will return the suitable steps for this example is 6 (index+1).

Then we should get the component value like this table:

N	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}	g_{11}
1	2.0000	1.0000									
2	1.4142	1.4142	1.0000								
3	1.0000	2.0000	1.0000	1.0000							
4	0.7654	1.8478	1.8478	0.7654	1.0000						
5	0.6180	1.6180	2.0000	1.6180	0.6180	1.0000					
6	0.5176	1.4142	1.9318	1.9318	1.4142	0.5176	1.0000				
7	0.4450	1.2470	1.8019	2.0000	1.8019	1.2470	0.4450	1.0000			
8	0.3902	1.1111	1.6629	1.9615	1.9615	1.6629	1.1111	0.3902	1.0000		
9	0.3473	1.0000	1.5321	1.8794	2.0000	1.8794	1.5321	1.0000	0.3473	1.0000	
10	0.3129	0.9080	1.4142	1.7820	1.9754	1.9754	1.7820	1.4142	0.9080	0.3129	1.0000

So we use this function

```
double[] LowFlatArray = baseData.getLowFlat(N);
```



This is not difficult, just return the value of the corresponding record.

Then get Electric length (β*l)

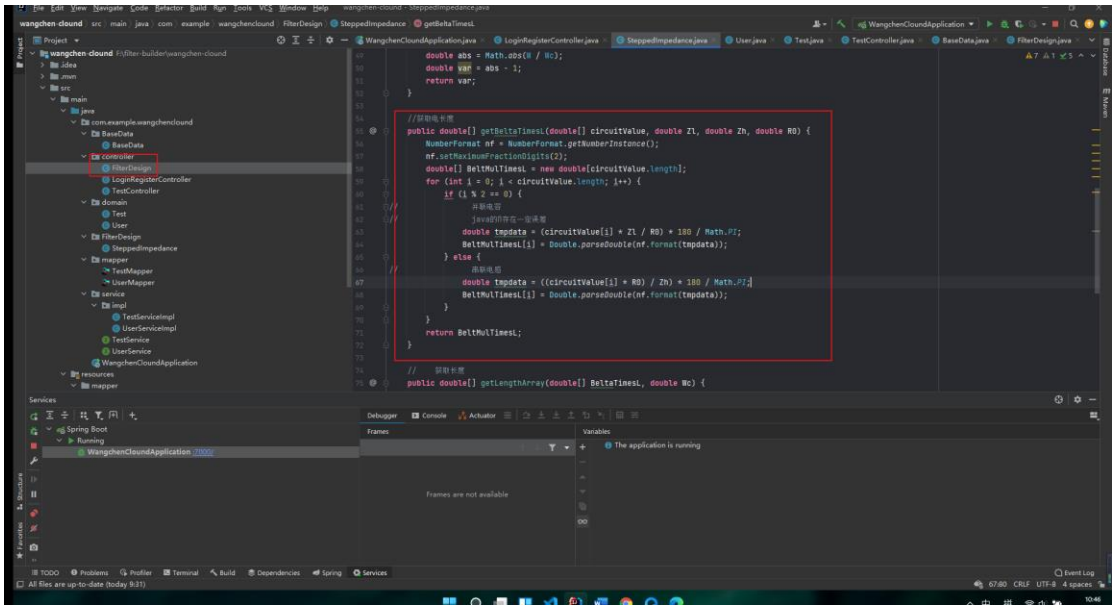
Use formular:

$$\beta\ell = \frac{LR_0}{Z_h} \quad (\text{电感})$$

$$\beta\ell = \frac{CZ_\ell}{R_0} \quad (\text{电容})$$

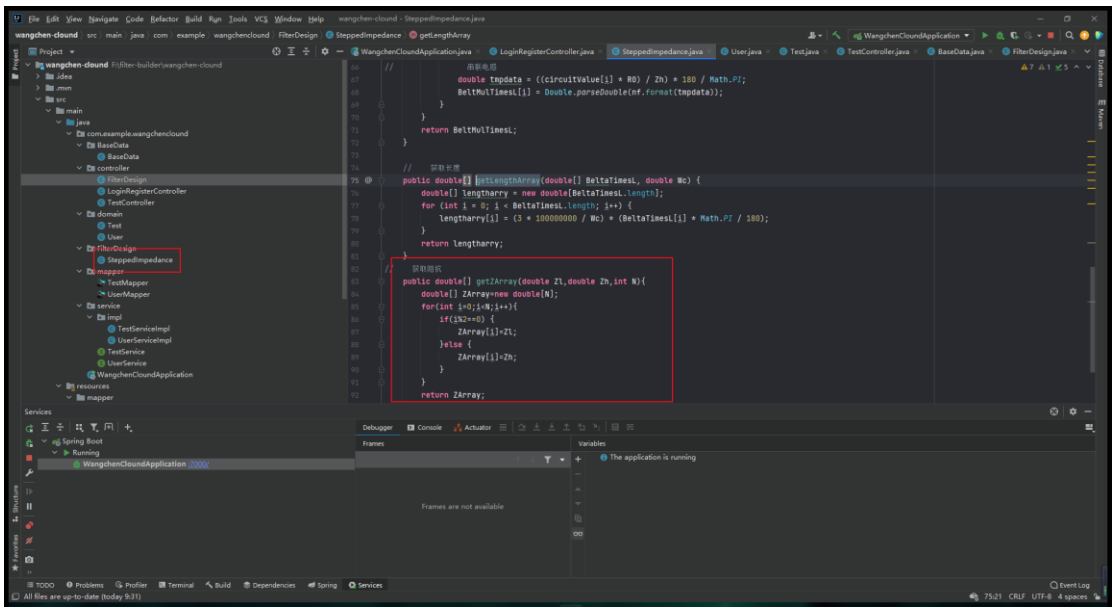
The function is :

```
double[] BeltaTimesL = steppedImpedance.getBeltaTimesL(LowFlatArray, Zl, Zh, R0);
```



Then get the impedance of every section

```
获取电阻
    double[] ZArray = steppedImpedance.getZArray(Zl, Zh, N);
```

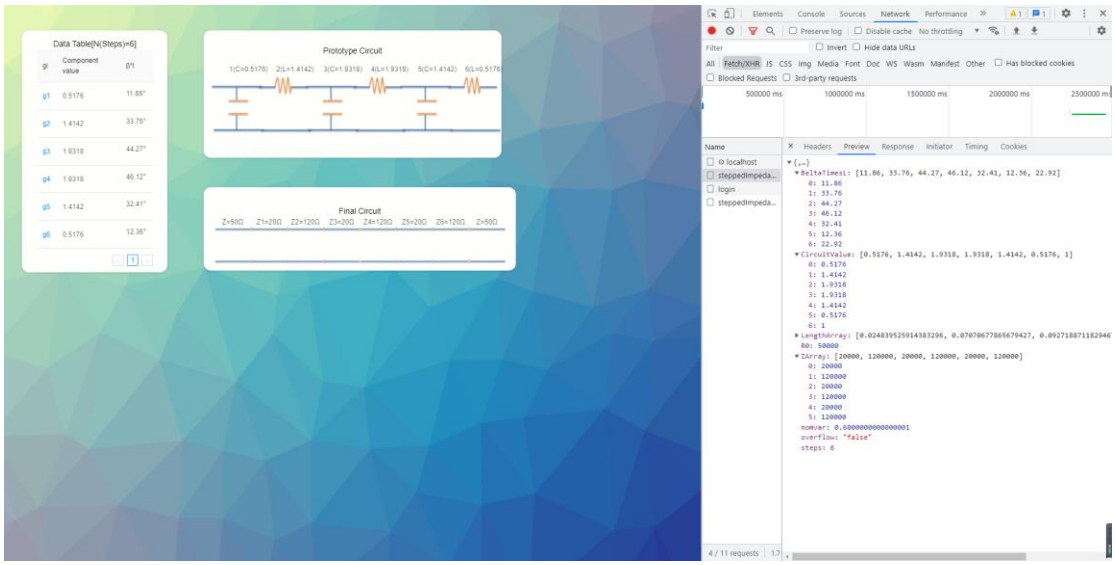


Finally, our interface returns these data.

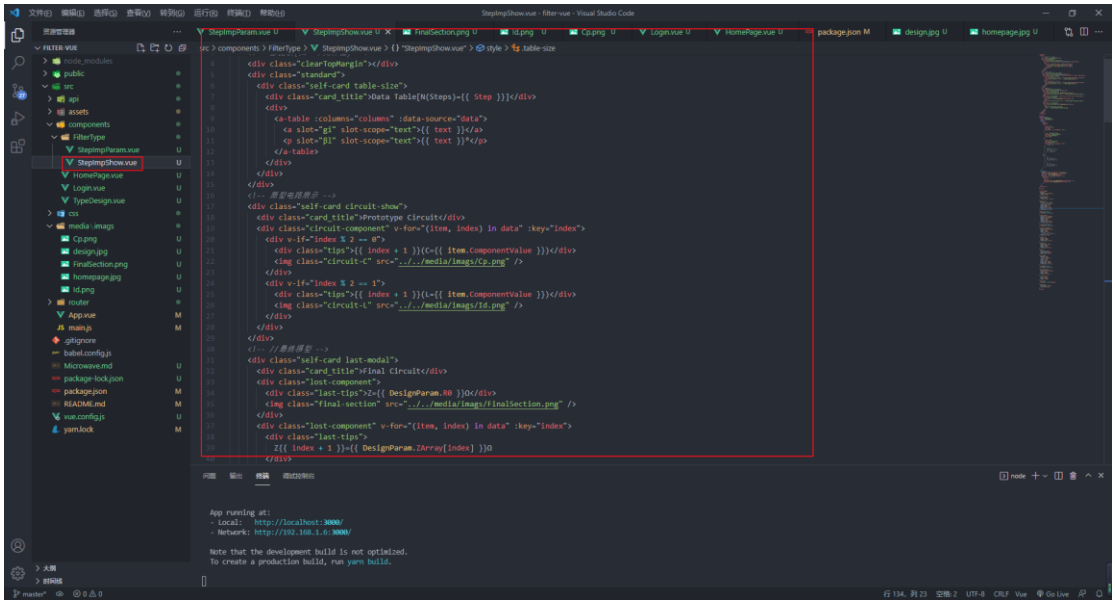
```
        double[] ZArray = steppedImpedance.getZArray(Zl, Zh
        jsonObject.put("CircuitValue", LowFlatArray);
        jsonObject.put("nomvar", nomvar);
        jsonObject.put("steps", N);
        jsonObject.put("BeltaTimesL", BeltaTimesL);
        jsonObject.put("LengthArray", LengthArray);
        jsonObject.put("ZArray", ZArray);
        jsonObject.put("R0", R0);
        jsonObject.put("overflow", "false");
    } else {
        jsonObject.put("overflow", "true");
    }
}
```

```
    return jsonObject;
}
```

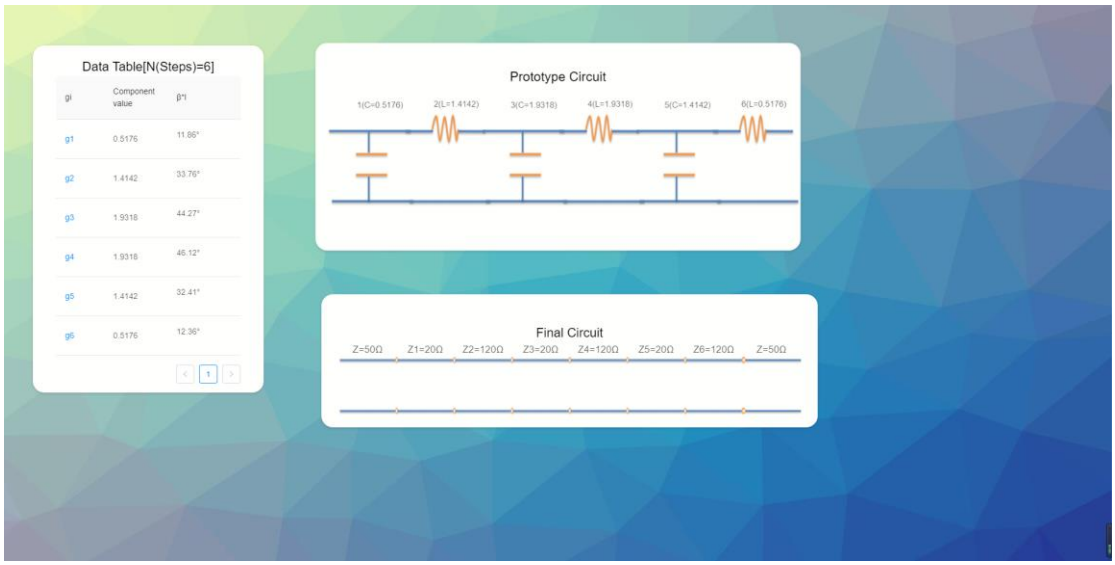
Then just look our http response:



We have obtained the required parameters (except for the microstrip lines), and then Vue render the page according to the obtained parameters.



So we get the modal:



These are some basic process principles.

stub filter