第一章 产品定位

1.1 产品介绍

sbsspro 是一个在线数据决策平台，提供数据处理，制作图表的功能，帮助用户更好分析数据特征及其相关性等数学特征。本产品目前实现了制作柱状图、折线图、饼图功能。

1.2 产品优势

对比 office 图表功能，本产品实现了导入 excel 一键作图的功能，大大优化了制作图标的效率，也降低了制作图表使用门槛，受众面更广。此外本产品还实现了处理了含有 1 个分类字段+多个数值字段的 excel 表格，导入数据不需要特定的模板，一键导入即可作图。

第二章 交互设计

本产品网站布局分为数据导入区、数据处理区、作图区、图标选择区四个板块。用户通过在数据导入区点击按钮选择本地的 excel 文件进行导入。在通过设定的代码进行文件格式验证后，数据处理区会以 JSON 格式呈现导入的数据。用户通过在图表选择区选择需要的图标类型，作图区就会生成相应的图标进行展示。在数据处理区下方的下拉链表可以查询到当前导入的数据字段，并可通过选择来删除数据。

图1 页面布局

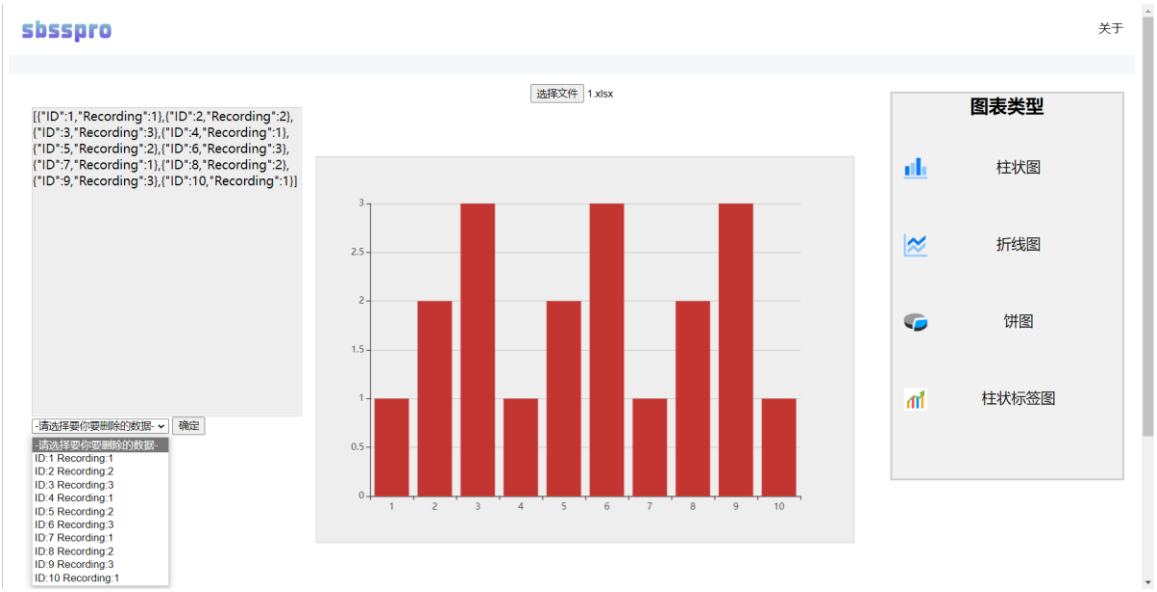

图2 柱状图实例

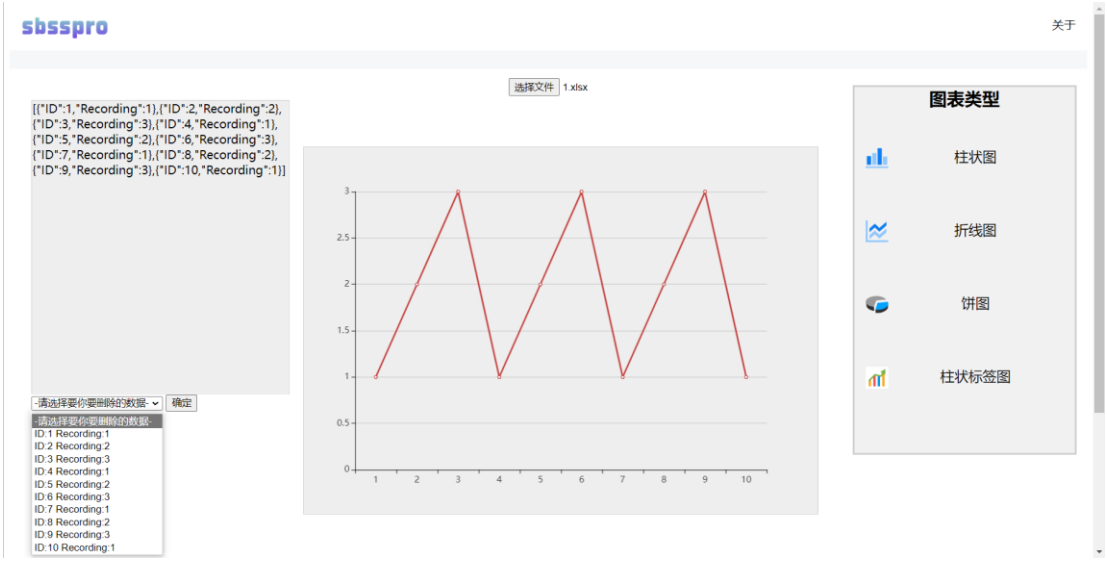

图3 折线图实例

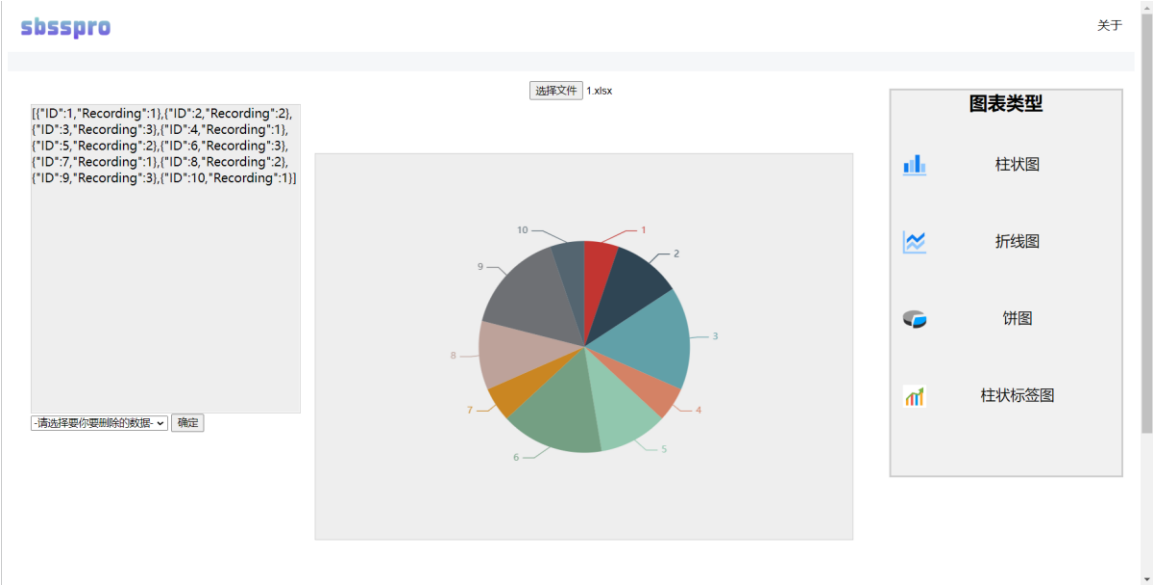

图4 饼图实例

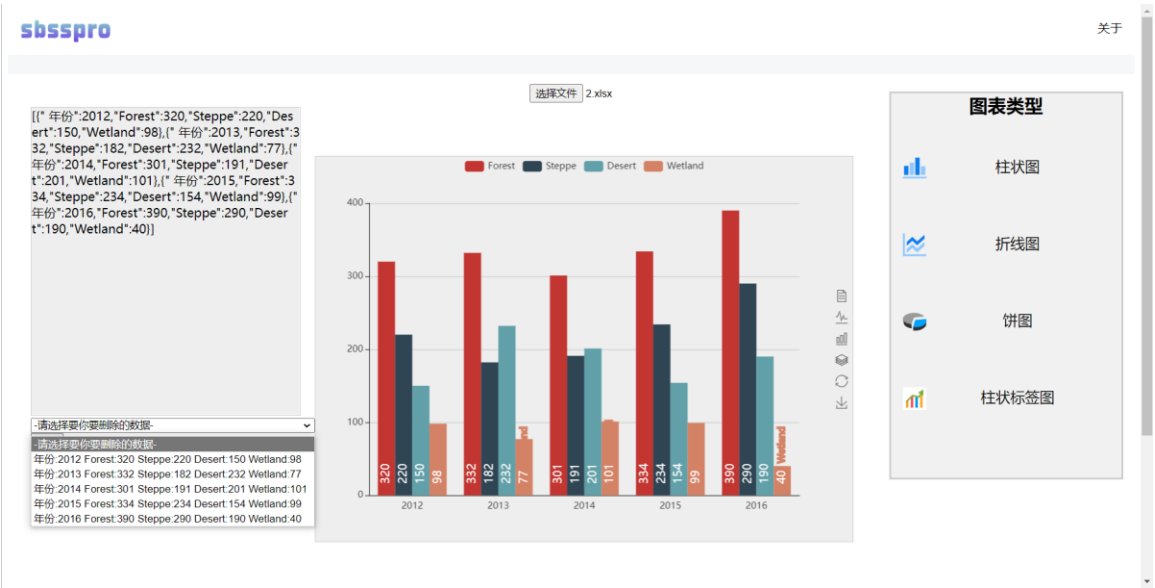

图5 柱状标签图实例

第三章 技术方案

本系统技术层面分为数据解析模块、数据处理模块和数据可视化模块。数据解析模块用于接收用户导入的 excel 并将数据解析为本地的 json 数据和本地数组或对象数据。数据处理模块获取到解析后的数据即可在数据处理区呈现导入的数据。

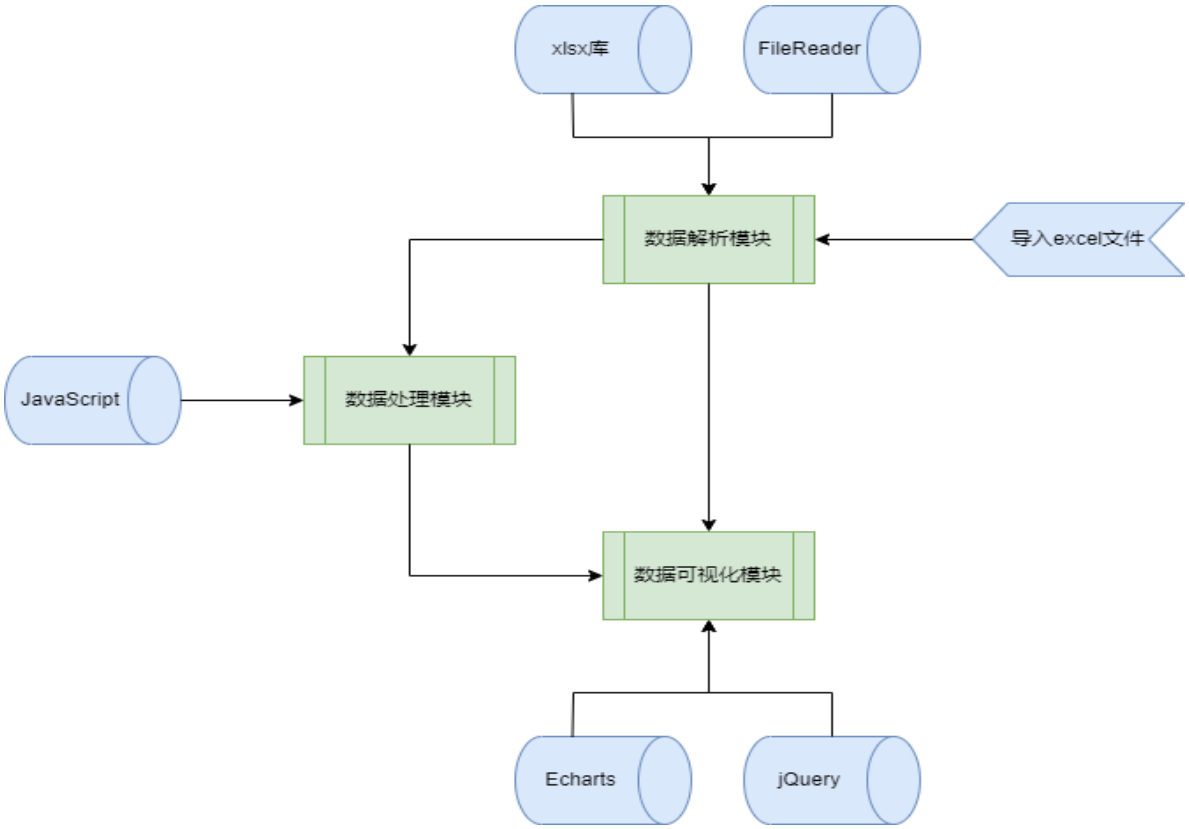

图6 系统总体设计

3.1 数据解析模块

对于 Excel 文件的内容分析转换，本系统利用了 xlsx 库和 FileReader 两个工具。部分代码如下。

```
if (!obj.files) {
      return;
}
const IMPORTFILE_MAXSIZE = 1 * 2048; //这里可以自定义控制导入文件大小
var suffix = obj.files[0].name.split(".")[1]
if (suffix != 'xls' && suffix != 'xlsx') {
      alert('导入的文件格式不正确!')
      return
}
if (obj.files[0].size / 1024 > IMPORTFILE_MAXSIZE) {
      alert('导入的表格文件不能大于 2M')
      return
}

var f = obj.files[0];
var reader = new FileReader();
```

```
reader.onload = e => {
            var data = e.target.result;
            if (rABS) {
                  wb = XLSX.read(btoa(fixdata(data)), { //手动转化
                        type: 'base64'
                  });
            } else {
                  wb = XLSX.read(data, {
                        type: 'binary'
                  });
            }
    dd=wb.Sheets[wb.SheetNames[0]];
    aa = JSON.stringify(XLSX.utils.sheet_to_json(dd));
    ...
```

用 XLSX.read 读取获取到的 Excel 数据，返回 workbook，而 workbook 对象，指的是整份 Excel 文档。我们在使用 js-xlsx 读取 Excel 文档之后就会获得 workbook 对象。用 `workbook.Sheets[xxx]` 通过表名获取表格，这里的 `wb.Sheets[wb.SheetNames[0]]`是获取第一个 Sheet 的数据。用 `XLSX.utils.sheet_to_json` 针对单个表获取表格数据转换为 json 格式。

3.2 数据处理模块

数据处理模块利用 JavaScript 实现基本的数据查询和删除。部分代码如下。

```
//渲染选择数据框
dataArray = JSON.parse(pp); //json 转为数组
console.log(dataArray)
var deledata = document.getElementById("deledata");
deledata.options.length = 1;
for (var i = 0; i < dataArray.length; i++) {
      var dataValue = "";
      for (var x in dataArray[i]) {
            dataValue += x + ":" + dataArray[i][x] + " ";
      }
      var option = new Option(dataValue, i);
      deledata.options.add(option);
}
//删除数据
function deledata() {
      var index = document.getElementById('deledata').value;
      console.log(index)
      deleteRow(index)
      aa = JSON.stringify(XLSX.utils.sheet_to_json(wbsheet));
      u = eval('(' + aa + ')');
      console.log(u)
      //渲染选择框
      dataArray = JSON.parse(aa); //json 转为数组
      console.log(dataArray)
```

```
        var deledata = document.getElementById("deledata");
        deledata.options.length = 1;
        for (var i = 0; i < dataArray.length; i++) {
              var dataValue = "";
              for (var x in dataArray[i]) {
                    dataValue += x + ":" + dataArray[i][x] + " ";
              }
              var option = new Option(dataValue, i);
              deledata.options.add(option);
        }
        document.getElementById("databoard").innerHTML = aa;
}
//按行删除
function deleteRow(index) {
        const range = XLSX.utils.decode_range(wbsheet['!ref']);
        for (let row = index; row < range.e.r; row++) {
              for (let col = range.s.c; col <= range.e.c; col++) {
                    wbsheet[encodeCell(row, col)] = wbsheet[encodeCell(ro
w + 1, col)];
              }
        }
        range.e.r--;
        wbsheet['!ref'] = XLSX.utils.encode_range(range.s, range.e);

}
```

3.3 数据可视化模块

数据可视化模块借助了 Echarts 以及 jQuery 两个工具辅助实现。Echarts 是一个使用 JavaScript 实现的开源可视化库，提供了画柱状图、折线图、饼图的功能。但是在数据集的设置这一块遇到了较大问题，也是我在这个系统中耗费大量时间重点解决的问题。目前已知 Echarts 提供的代码可以以数组或者是对象为数据集进行画图。然而这个数据集具有严格的格式要求，例如“对象数组”的形式。而由于用户导入的 excel 内容可以是自定义的，因此通过数据解析模块获取到的数据无法直接应用于 Echarts 提供的数据集中，需要对获取的 json 数据进行修改属性名或者是转换成数组的形式，而这一过程就无法避免对数据进行查询遍历修改的重新整合。因此这也是本系统在开发过程中的一个重点问题。

目前本系统已经实现对于一个分类字段+一个数值字段的数据可视化功能和查询删除功能，对于一个分类字段+多个数值字段的数据目前实现了可视化功能。图表类型选择的“柱状图”、“饼图”、“条形图”都是针对一个分类字段+一个数值字段的数据，而“柱状标签图”是针对一个分类字段+多个数值字段的数据的数据的。

```
//画柱状图
function drawbar() {
        picType = "bar";
```

```
        echarts.dispose(document.getElementById('main'));
        myChart = echarts.init(document.getElementById('main'));
        var xDataArr = [];
        var yDataArr = [];
        console.log(u)
        for (var i = 0; i < u.length; i++) { //数据重新整合
              xDataArr.push(u[i].name)
              yDataArr.push(u[i].value)
        }
        myChart.setOption({
              xAxis: {
                    type: 'category',
                    data: xDataArr
              },
              yAxis: {
                    type: 'value'
              },
              series: [{
                    type: 'bar', // 设置图表类型为饼图
                    data: yDataArr
              }]
        })
}
//折线图、饼图及柱状标签图代码过于冗长，详见源代码
```