MetaQ功能模块

服务器：

服务器操作系统为linux。使用nginx作为服务器代理，接收请求并将请求转发到对应的端口。

使用node.js编译运行vue，运行在某个端口上。

后端打包成jar包，使用java -jar xxx.jar指令运行，持续监听某个端口。

按组件划分模块：

Tabbar组件：

自定义底部导航栏，有更高自由度，分为三个区域，可以更改图标，有不同的效果，主要有两个版本，第一种从左到右分别是我的问卷，添加问卷及主页，第二章时添加问卷的时候从左到右分别是设置，添加问题及保存。

利用props接受父组件传值，动态修改图片，接受tab参数在onload生命周期赋值给activeTab，用来判断当前页面，修改图片颜色并添加圆点用于指示。

Add组件：

为增加从下而上的动态效果，且与其他tabbar按钮样式不同而增加的组件，可以更加灵活定义相关功能。

props接受一个type值（Number类型）用于判断当前需要调用哪种添加，根据Tabbar组件主要分为两种模式，第一种是调用添加问卷的页面，第二种是调用添加问题的页面。

未来或许会有更多不同的功能，同样可以通过添加type的方式实现，维护性与灵活度更高。

QBlock组件：

用于我的问卷页面中我创建的问题板块，包含标题，发布状态以及答卷数量等，为了保持样式美观，添加了动态改变字体颜色的功能，并且x千转换为xk，x万转换为x万，节省空间的同时，可以大致在这个页面就知道答卷数量，点击后会跳转至详细的问卷界面。

QBlock2组件：

用于我的问卷页面中我填写的问题板块，因为不是问卷创建者或拥有者，故没办法查看数量，但在这个板块可以查看问卷是否已经截止，点击查看按钮可以进入详细界面，包含填写情况，或根据权限查看更多内容（如问卷分析等）

tabSwiper组件：

用于我的问卷页面以及查看问卷页面，父组件传入标题的值，可以通过左右滑动切换下面显示的内容，也可以通过点击标题切换。注意：但由于下面显示的内容差异可能较大，所以组件要搭配页面中的值以及内容才可以更好的切换，不然只有标题的样式切换。

以下是待实现组件

analysis组件：

用于问卷详细中的分析模块，传入问卷数据（问卷标题、类型、数据等）利用uni-chart或echart生成图标，下面有一排按钮可以切换不同类型的图标（饼状、环状等）

chart组件（可能）：

可能会用于analysis组件中，生成不同图表并添加一些图标特有功能，比如选中饼状图的一块会放大，或者选中一些数据会浮窗显示等，如果不需要则直接在analysis组件实现。

questions组件：

用于创建问题的界面，为了更好的获取后端数据并在前端渲染，questions组件主要用于判断要生成什么类型的题目（如单选、多选、填空等）也就是判断需要调用什么样的组件。

question组件（包含各种类型题目）：

用于创建题目，每种题型的样式和功能都会有所出入，所以需要不同组件封装，目前主要包含7-8种

answers组件：

用于填写问卷的界面，为了更好的获取后端数据并在前端渲染，answers组件主要用于判断要生成什么类型的题目（如单选、多选、填空等）也就是判断需要调用什么样的组件。

answer组件（包含各种类型题目）：

用于生成题目，每种题型的样式和填写方式都会有所出入，所以需要不同组件封装，目前主要包含7-8种

按页面划分功能模块：  
登录注册页面：

1. 登录页面

1.1正则匹配用户名(邮箱)

1.2前端将用户名密码经过服务器传给后端，验证

1.3可切换至注册页面

1.4 忘记密码，修改密码功能

1.5 使用localstorage api在用户本地存储token

1. 注册页面

2.1正则匹配用户名(邮箱)，判断输入的两次密码是否一致

2.2服务器发送验证码给用户邮箱

2.3验证码倒计时等待再次发送（避免频繁请求）

2.4动态绑定登录注册按钮，只有在所有资料完善会改变样式并可以点击

2.5分配唯一token用于辨识用户

2.6可切换到登录页面(注册成功后自动切换到登录页面)

我的问卷页面：

1. 页面“我创建的”：

1.1搜索功能，可根据问卷题目进行快速检索

1.2展示所有“我”创建的问卷，同时附上答卷数量，以及问卷当前情况(收集中或未收集)

1.3用户可选择按照时间或答卷数量排序问卷

1.4标题过长自动隐藏

1.5答卷数量过多显示会自动变化(如10000显示为1W)

1. 页面“我填写的”：

2.1搜索功能，可根据问卷题目进行快速检索

2.2展示所有“我”填写过的问卷

2.3左下角显示问卷发起人

2.4可选择点击问卷查看，如果发起人公开，可查看答卷汇总后的所有数据

2.5如果此时还允许修改，可修改问卷

查看问卷页面:

1. 分析页面

1.1可选择饼状,环状，折线，条形，图标，柱状来展示问卷结果。

1.2可选择将渲染出的图片单个或者批量保存到本地

1.3 添加一个分享问卷功能，将分析结果分享给他人。

1. 编辑页面

2.1对于填写者，此功能不可用

2.2复制-编辑功能：复制当前问卷的问题和格式，可修改后重新发布，原问卷不受影响

2.3保留-编辑功能：修改问卷的权限(当前是否继续收集，填写者是否可查看问卷结果)

2.4删除当前问卷

2.5点击下一步，才能进入所选择的功能的页面(防止误触)

2.6 添加一个分享问卷的按钮，分享给他人填写。

主页：

1.1头像上传功能：用户点击头像，可上传本地图片，弹出剪裁框，确认剪裁头像

1.2用户名编辑：用户点击用户名框，背景变模糊，用户名框突出显示

1.3个人信息编辑：包括手机号，邮箱，ip地址。点击时，同样会放大突出显示，背景虚化模糊

1.4切换账号功能：切换别的账号，直接回到初始页面

1.5修改密码功能：修改密码，跳转到修改密码页面

添加问卷页面：

1.1输入问卷标题和简介。标题和简介会默认在虚线框的正中间，随着输入字数的增多，标题和简介会自动换行，并且始终整一段位于正中间，当输入字数过多，超出了虚线框，自动产生滑动条

1.2点击“+”号按钮，弹出问题选择页面，包括单选题，多选题，填空题，滑动条,以及模板题目(手机，日期，地点)

1.3可点击相应模块编辑问题和选项

1.4可以点击修改按钮修改已经编辑完成的题目

按接口划分功能模块：

**登录：**

请求路径：/login

请求方式：post

请求参数：

用户邮箱

用户密码

示例：

{

“email”:[123456789@qq.com](mailto:123456789@qq.com),

“password”:”123456789”

}

响应数据：

状态码

状态码描述

服务器返回的属于用户的token

示例：

{

“code”:1, //0表示失败，1代表成功

“msg”:”success”,

“token”:”user”

}

**注册接口：**

请求路径：/register

请求方式：post

请求参数：

邮箱

密码

验证码

示例：

{

“email”:[123456789@qq.com](mailto:123456789@qq.com),

“password”:”123456789”,

“code”:”1a2b3c”

}

响应数据：成功或者失败

示例：  
{

“code”:1, //0表示失败，1代表成功

“msg”:”success”,

}

**验证码接口**

请求路径：/confim

请求方式：get

请求参数：

邮箱

示例：

{

“email”:”123456789@qq.com”

}

响应数据：发送验证码到邮箱是否成功

{

“code”:1, //0表示失败，1代表成功

“msg”:”success”,

}

**“我的问卷”页面：**

概览：获取用户已创建的问卷的基本信息

请求路径：/getCreated

请求方式：GET

请求参数：请求头中携带token信息即可

响应数据：

问卷的名字

问卷发布状态

问卷填写份数

示例：

状态码

状态描述

多个问卷信息对象的数组

{

"code":1,

"msg":"success", //对于上面的那个代码的文本解释，成功或者失败

"data":[ //多个问卷信息组成的数组

{

"title":"问卷", //问卷名字

"state":0, //问卷是否已发布，0为未发布，1为已发布

"num":0 //问卷填写的份数，若未发布，则为0

},

{

"title":"问卷", //问卷名字

"state":0, //问卷是否已发布，0为未发布，1为已发布

"num":0 //问卷填写的份数，若未发布，则为0

}

]

}

**“创建问卷”页面：**

概览：问卷包含若干个问题项，前端应该提交的是多个代表问题项的信息的对象，其中的信息包含问题的描述，类型，以及这个问题可能需要的字符串（比如选择题，需要各个选项的信息）

请求路径：/create

请求方式：post

请求参数：

多个问题项对象的数组

问题项包含：问题，类型，额外需要的字符串（选择题为主）

[

{

“question”:”你的爱好是？”,

“type”:”single”, //代表单选

“choice”:[

“java”,”c++” //代表各个选项的内容，顺序也和原来选项的顺序一致

]

},

{

“question”:”为什么喜欢这个？”,

“type”:”blank”

//choice字符串数组如果不需要可以省略，反正没有的话后端对应的变量赋值也是null

}

…

]

附：各个类型的题目所对应的type值：

单选：single

多选：multiple

填空：blank

滑条：slide

图片：picture

手机：phone

日期：date

位置：address

附：需要额外字符串信息的题目类型以及字符串的含义

单选：需要一个字符串数组，表示多个选项的内容

多选：同上

滑条：需要两个字符串值，一个代表滑条的下限，一个代表上限

响应数据：

仅仅返回成功的信息即可，不需要其他数据

**编辑问卷：**

概览：需要返回用户之前保存的问卷样式

请求路径：/edit

请求方式:Get

请求参数：

请求头：用户的token

请求行：

问卷名

示例：/edit?title=”爱好调查”

响应数据：

1.响应是否成功的状态码

2.解释状态码的文本

3.需要的数据的返回

返回的是问题项对象的数组

问题项对象包含：问题的描述，问题类型，问题的额外字符串（主要是选择）

示例：

{

"code":1,

"msg":"success",

“data”:[ //返回现在已编辑好的问题项

{

“question”:”你的爱好是？”,

“type”:”single”, //代表单选

“choice”:[

“java”,”c++” //代表各个选项的内容，顺序也和原来选项的顺序一致

]

},

{

“question”:”为什么喜欢这个？”,

“type”:”blank”

//choice字符串数组如果不需要可以省略，反正没有的话后端对应的变量赋值也是null

}

]

}

**编辑提交问卷：**

概览：接口传输的数据和创建差不多，后端的操作和创建相比仅仅少了一个初始化问卷对象

请求路径：/editCommit

请求方式：post

请求参数：

请求头：用户的token

请求体：问题项的数组

[

{

“question”:”你的爱好是？”,

“type”:”single”, //代表单选

“choice”:[

“java”,”c++” //代表各个选项的内容，顺序也和原来选项的顺序一致

]

}，

{

“question”:”为什么喜欢这个？”,

“type”:”blank”,

}

…

]

响应数据：

样例：

{

"code":1, //代表成功的代码，其中0表示失败，1表示成功

"msg":"success", //对于上面的那个代码的文本解释，成功或者失败

“data”:”” //成功了，不需要返回什么

}

**填写问卷:**

概览：当打开这个问卷时，需要返回这个问卷包含的问题的信息

请求路径：/fill

请求方式：post

请求参数：

请求头：用户的token

问卷的名字

创建这个问卷的用户名

请求体：无

响应数据：

状态码

状态描述

问题项的数组

示例：

{

“code”:1,

“msg”:”success”,

“data”:

[

{

“question”:”你的爱好是？”,

“type”:”single”, //代表单选

“choice”:[

“java”,”c++” //代表各个选项的内容，顺序也和原来选项的顺序一致

]

}，

{

“question”:”为什么喜欢这个？”,

“type”:”blank”,

}

…

]}

**保存填写：**

概览：把用户目前输入的数据保存，以便下次继续填写

保存不会把数据提交到统计对象，仅仅会创建一个新的字符串数组用于保存填写的数据

当传递字符串数组时每个数组项的顺序要和问题顺序一致，即使用户的某些问题没有填写，也需要主动把这个字符串赋予空值

请求路径：/saveFill

请求方式：post

请求参数：

请求头：用户的token

请求体：

问卷的名字

创建问卷用户的名字

代表用户填写数据的字符串数组

示例：

{

“title”:”爱好调查”,

“username”:”创建者xx”,

Data:[

“1”,””

]

}

附：不同类型的问题的数据形式：

单选：选择第几个选项就用第几个数字表示，比如第一个选项就代表1

多选：字符串内部包含多个数字

文本：字符串就是输入

日期：要满足一定格式的字符串，比如年月日（yyyy-MM-dd）

滑条：一个在区间内的数字

地址：暂定为表示地址的字符串

手机：手机号字符串

图片：暂定

空字符””表示用户暂未填写这个问题

响应数据：

无特殊数据返回，仅仅表示成功或失败

**提交填写：**

概览：大体操作和保存填写一样，但是后端会把数据保存在统计对象内

请求路径：/commitFill

请求方式：post

请求参数：

请求头：用户的token

请求体：

问卷的名字

创建问卷用户的名字

代表用户填写数据的字符串数组

示例：

{

“title”:”爱好调查”,

“username”:”创建者xx”,

Data:[

“1”,””

]

}

响应数据：

无特殊数据返回，仅仅表示成功或失败

**获取填写记录：**

概览：获取填写过的问卷的信息

请求路径：/fillRecord

请求方式：get

请求参数:用户的token

响应数据：

状态码

状态描述

包含：问卷名，问卷作者，问卷状态的对象

示例：

{

“code”:1,

“msg”:”success”,

“data”:{

“爱好调查“,”问卷作者”,”1”

}

}

附：问卷状态有0和1，0表示未截止，1表示已截止

**数据统计：**

概览：数据统计将一个问题一个问题地显示，所以应该一个问题一个问题地请求，在请求路径加上需要问题的序号

请求路径:/statistics/{index}

请求方式：get

请求参数：

请求头：用户的token

请求体：

问卷名

响应数据：

状态码

状态描述

问题（question）

问题类型（type）

一个数组，可以是数值类型，也可以是字符串类型，取决于问题的类型

示例：{

“code”:1,

“msg”:”success”,

“data”:[10,10,10,10]

}

数值类型代表的是选择题每个选项被选择的次数，字符串类型则是其他类型的问题，可能由于填写的用户太多，一次性显示不完所有的字符，所以需要分批显示