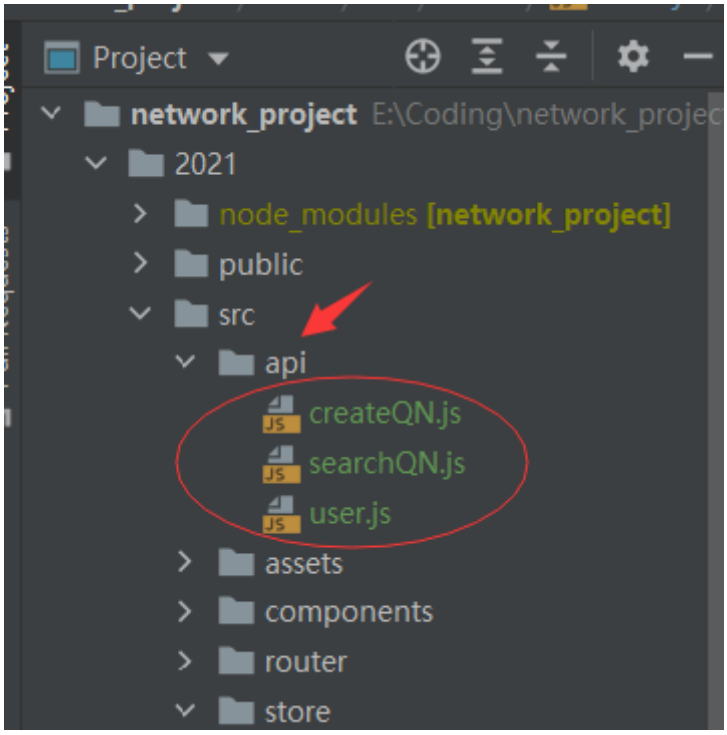
前后端接口说明

一、接口目录



在src文件夹下新建了一个名api的文件夹，里面用于存放和后端进行数据交互的接口文件

二、调用说明

（一）用户相关

集中于api文件夹下的user.js中，其中方法见下图

```
import axios from 'axios'

export const login = payload => {
  console.log(payload)
  const { accountNumber, password } = payload
  return axios.post( url: '后端提供的Controller', data: { accountNumber, password }).then(res => {
    return res.data
  })
}

export const signup = payload => {
  console.log(payload)
  const { accountNumber, nickname, password } = payload
  return axios.post( url: '后端提供的Controller', data: { accountNumber, nickname, password }).then(res => {
    return res.data
  })
}
```

1、用户登录

在登录界面，用户点击"登录"后，通过Login.vue中的login()方法来触发user.js中的login方法，如下：

在Login.vue中，先触发其中的login()方法

```
login() {
  // TODO: 点击登录按钮后实现登录功能，首先检查是否输入，然后检查输入是否有误，然后检查数据库中是否有该用户，无误后登录
```

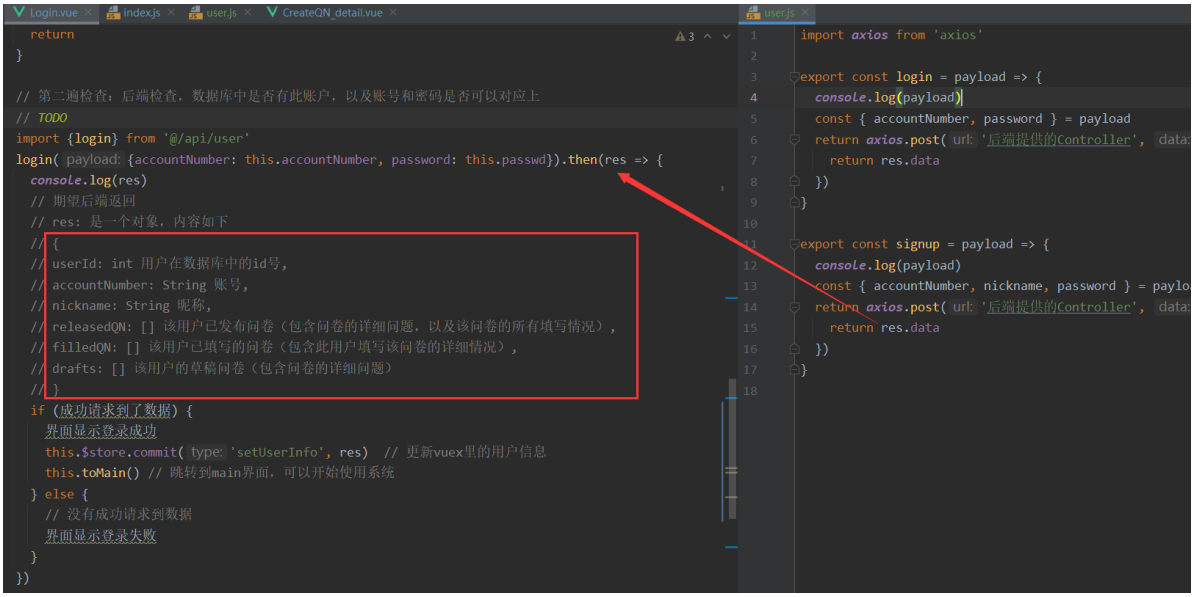
然后通过import来引入src/api/user.js中的login方法，并调用该方法，传入参数payload为一个对象，包含两个成员accountNumber（账号）和password（密码）

```
import {login} from '@/api/user'
login( payload: {accountNumber: this.accountNumber, password: this.passwd}).then(res => {
  console.log(res)
  // 期望后端返回
  // res: 是一个对象，内容如下
  // {
  // userId: int 用户在数据库中的id号，
  // accountNumber: String 账号，
  // nickname: String 昵称，
  // releasedQN: [] 该用户已发布问卷（包含问卷的详细问题，以及该问卷的所有填写情况），
  // filledQN: [] 该用户已填写的问卷（包含此用户填写该问卷的详细情况），
  // drafts: [] 该用户的草稿问卷（包含问卷的详细问题）
  // }
  if (成功请求到了数据) {
    界面显示登录成功
    this.$store.commit( type: 'setUserInfo', res)  // 更新vuex里的用户信息
    this.toMain() // 跳转到main界面，可以开始使用系统
  } else {
    // 没有成功请求到数据
    界面显示登录失败
  }
})
```

先执行user.js中的login方法，该方法就是后端匹配的接口，见下图。通过axios.post调用，第一个参数为url，也就是接收前端数据作为参数的后端方法，第二个参数为data，也就是前端要往后端传的数据，这里就是一个有两个成员的对象。

```
export const login = payload => {
  console.log(payload)
  const { accountNumber, password } = payload
  return axios.post( url: '后端提供的Controller',  data: { accountNumber, password }).then(res => {
    return res.data
  })
}
```

获取到了后端传回的数据后，随后执行then中方法，将后端传回数据作为参数res，并将res.data返回给Login.vue中的login方法，随后调用then中的方法，见下图，分为成功登录和不成功登录两种情况。



2、用户注册

注：相关数据库须有一个userid作为主键

基本原理和用户登录一样，在注册界面点击注册按钮后，先调用Signup.vue中的signup方法

```
signup() {
  // TODO: 用户点击注册按钮后，先检查输入是否合格，再检查是否与现有数据存在冲突，如果都正确，将注册信息存入数据库
```

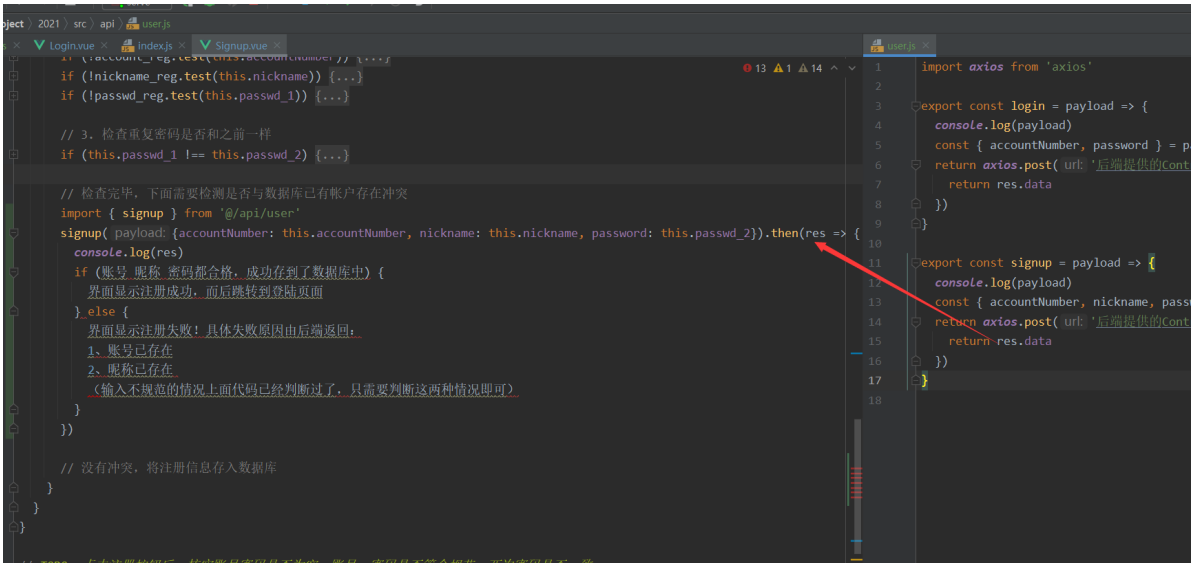
先通过前端检查输入不规范的情况，以保证传入数据库的信息是正确的，然后从api/user.js中引入signup方法，随后调用之，传入参数payload为一个包含三个成员的对象，accountNumber、nickname和password，随后正式调用user.js中的signup方法

```
  // 检查完毕，下面需要检测是否与数据库已有帐户存在冲突
  import { signup } from '@/api/user'
  signup( payload: {accountNumber: this.accountNumber, nickname: this.nickname, password: this.passwd_2}).then(res => {
    console.log(res)
    if (账号 昵称 密码都合格，成功存到了数据库中) {
      界面显示注册成功，而后跳转到登陆页面
    } else {
      界面显示注册失败！具体失败原因由后端返回：
      1、账号已存在
      2、昵称已存在
      （输入不规范的情况上面代码已经判断过了，只需要判断这两种情况即可）
    }
  })
```

还是通过axios.post请求后端的方法，第一个参数url就是接收前端数据作为参数的后端方法，第二个参数为data，也就是前端要往后端传的数据，这里就是一个有三个成员的对象。获取到了后端传回的数据后，随后执行then中方法，将后端返回的数据作为res，并将res.data返回给Signup.vue中的signup方法中

```
export const signup = payload => {
  console.log(payload)
  const { accountNumber, nickname, password } = payload
  return axios.post( url: '后端提供的Controller', data: { accountNumber, nickname, password }).then(res => {
    return res.data
  })
}
```

而后调用Signup.vue中的signup里面then中的过程，分为注册成功和不成功两种情况，注册不成功的情况需要后端判断，一是账号已存在、二是昵称已存在，把错误信息的字符串也要一并返回

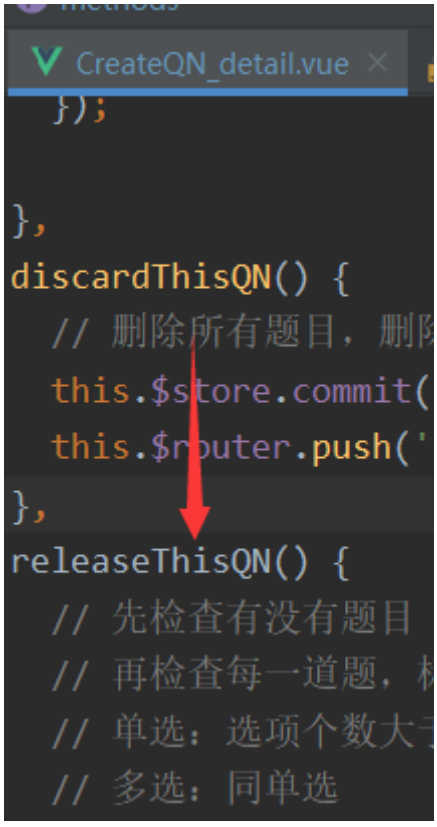


（二）问卷相关

1、发布问卷（src/api/createQN.js）

注：相关数据库须有一个releaseid作为主键

点击发布问卷按钮、确定后，调用CreateQN_detail.vue 中的releaseThisQN()方法



然后引用api/createQN.js中的release方法，并调用之，传入的参数为三个成员的对象，包括用户id（用于后端数据库作insert用）、问卷题目titleOfQN，和题目数组questions，题目数组questions中包含了一些对象，每道题对象格式为

title: String,

type: String(题目类型，有三种情况：single、multiple和gap-fill),

must: String(该题必答与否，有两种情况：must、optional),

options[ ] (如果是选择题的话)

```
// TODO: 检查结束，每个问题都符合要求后，下面要把问卷传到数据库，数据库存储完毕后，再将该问卷信息从$store中删除
import {release} from '@/api/createQN'
release( payload: {
  userID: this.$store.state.userID,
  titleOfQN: this.$store.state.titleOfQN,
  questions: this.$store.state.questions
}).then(res => {
  console.log(res)
  if (发布成功，即存储到数据库成功) {
    页面显示发布成功
  } else {
    页面显示发布失败
  }
})
```

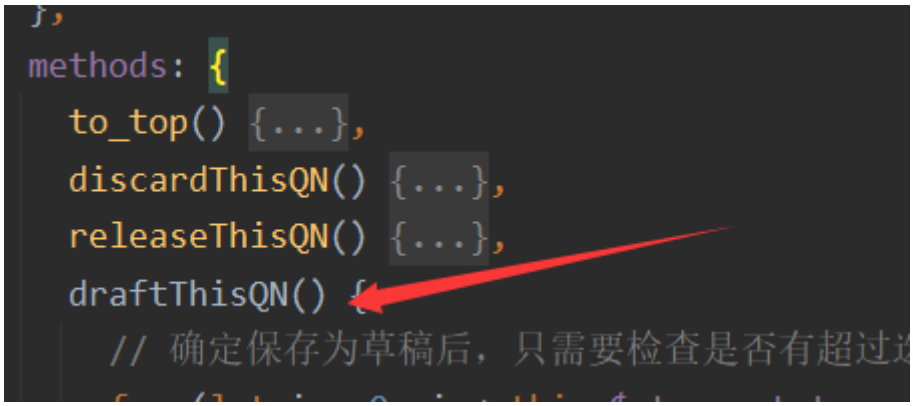
调用api/createQN.js中的release方法，调用后回到releaseThisQN方法

```
export const release = payload => {
  console.log(payload)
  const { userID, titleOfQN, questions } = payload
  return axios.post( url: '后端提供的接收问卷信息的接口', data: { userID, titleOfQN, questions }).then(res => {
    return res.data
  })
}
```

2、存为草稿（src/api/createQN.js）

注：相关数据库须有一个draftid作为主键

点击存为草稿按钮、确定后，调用CreateQN_detail.vue 中的draftThisQN()方法



然后引用api/createQN.js中的draft方法，并调用之，传入的参数为三个成员的对象，包括用户id（用于后端数据库作insert用）、问卷题目titleOfQN，和题目数组questions，题目数组questions中包含了一些对象，每道题对象格式为

title: String,

type: String(题目类型，有三种情况：single、multiple和gap-fill),

must: String(该题必答与否，有两种情况：must、optional),

options[ ] (如果是选择题的话)

```
// TODO: 检查结束，下面要把问卷草稿传到数据库，数据库存储完毕后，再将该问卷信息从$store中删除
import {draft} from '@/api/createQN'
draft( payload: {
  userID: this.$store.state.userID,
  titleOfQN: this.$store.state.titleOfQN,
  questions: this.$store.state.questions
}).then(res => {
  console.log(res)
  if (保存成功，即存储到数据库成功) {
    页面显示保存成功
  } else {
    页面显示保存失败
  }
})
```

调用api/createQN.js中的draft方法，调用后回到DraftThisQN方法

```
export const draft = payload => {
  console.log(payload)
  const { userID, titleOfQN, questions } = payload
  return axios.post( url: '后端提供的接收问卷信息的接口', data: { userID, titleOfQN, questions }).then(res => {
    return res.data
  })
}
```

3、查找问卷（src/api/searchQN.js）

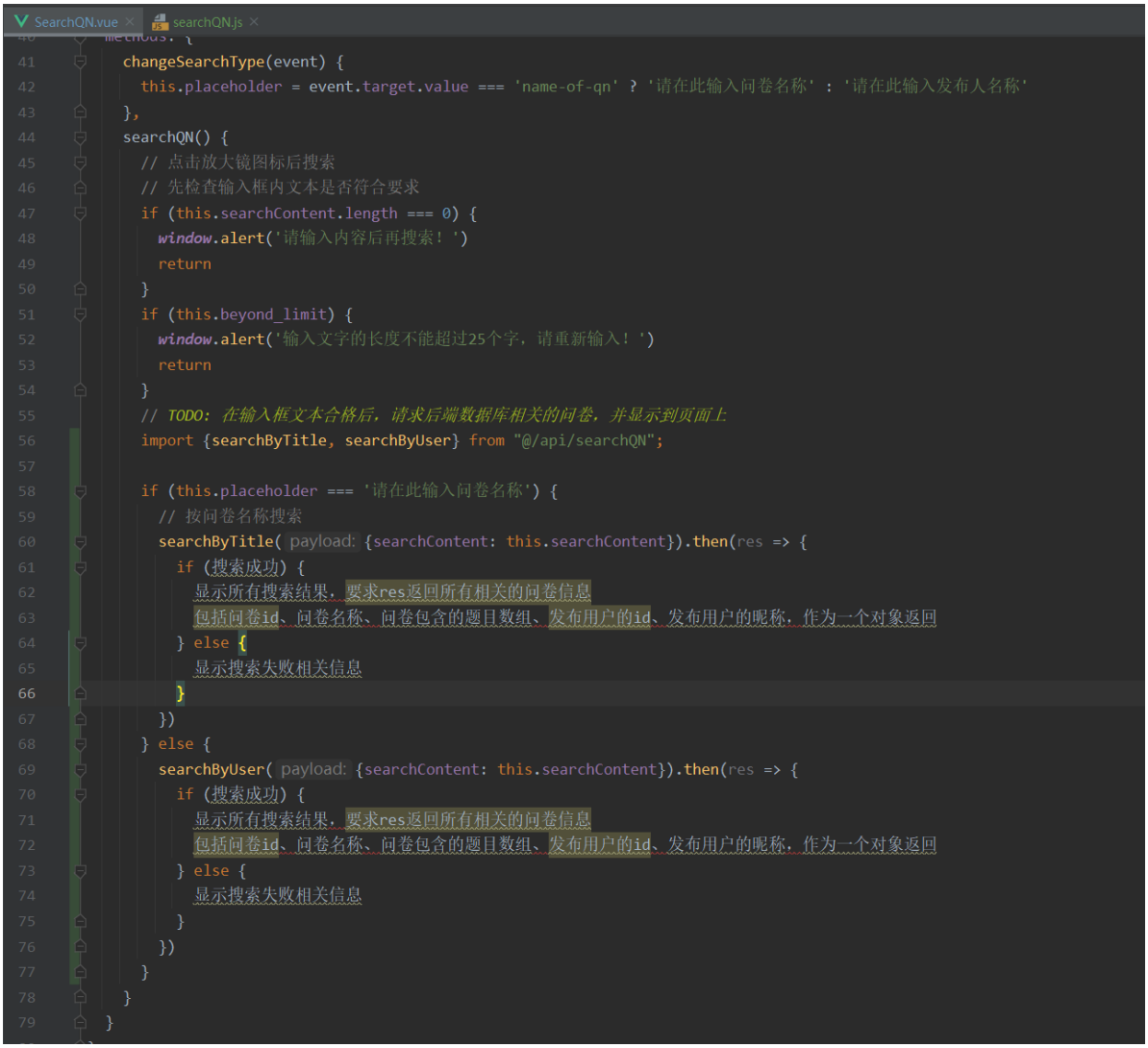
分为按用户查找、按问卷题目查找

SearchQN.vue | searchQN.js

```
import axios from "axios";

export const searchByTitle = payload => {
  console.log(payload)
  const { searchContent } = payload
  return axios.post( url: '后端返回问卷信息的接口', data: { searchContent }).then(res => {
    return res.data
  })
}

export const searchByUser = payload => {
  console.log(payload)
  const { searchContent } = payload
  return axios.post( url: '后端返回问卷信息的接口', data: { searchContent }).then(res => {
    return res.data
  })
}
```



4、填写问卷

目前前端还没有相应的api，需要在查找问卷页面点击查找到的问卷下方的"填写问卷"来触发，期望后端有这样一个方法，可以根据releaseid（数据库中存储已发布问卷的表的主键）和userid（当前填写这张问卷的用户id）将传入的填写内容保存到数据库中。

可以单独建立一个"填写情况表"，属性包括fillid（该表的主键），releaseid（被填写的问卷的id），userid（填写者的id），content（每道题的填写情况，初步想法是一个数组，对应索引的值为对应问题的回答）

5、管理问卷

管理问卷页面可以查看该用户已填写、已发布的问卷，还有草稿问卷。

这些数据希望在用户成功登录时就请求到，并存到vuex中，只需直接取用即可。

因此期望在**用户登录**成功后，前端已经得到了后端传过来的用户的已填写、已发布、草稿箱问卷，

```
releasedQN: [],
filledQN: [],
drafts: []

都是数组，内含0到多个问卷对象，每个问卷对象包含：
```

1、如果是releasedQN[]，为releaseid；如果是filledQN[]，为fillid；如果是drafts[]，为draftid。整型
2、问卷题目titleOfQN，字符串
3、题目数组questions，而题目数组questions中又包含了一些对象，每道题对象格式为

title: String,

type: String(题目类型，有三种情况：single、multiple和gap-fill),

must: String(该题必答与否，有两种情况：must、optional),

options[] (如果是选择题的话)