Vant组件van-uploader表单提交使用

1.需要在src/http/axios.js下新增 post_formData 方法

```
export function post_formData(url, data) {
    return axios({
        method: "post",
        url,
        data,
        headers: {
            "Content-Type": "multipart/form-data"
        },
        timeout: 10000
    })
}
```

2.使用步骤

2.1根据自己的需求，在表单中使用van-uploader组件

```
<van-field name="uploader" label="身份证正面照">
  <template #input>
    <van-uploader :after-read='afterRead1'
    v-model="fileList1" :max-count="1" />
  </template>
</van-field>
<van-field name="uploader" label="身份证反面照" >
  <template #input>
    <van-uploader :after-read='afterRead2'
    v-model="fileList2" :max-count="1" />
  </template>
</van-field>
<van-field name="uploader" label="银行卡照片">
  <template #input>
    <van-uploader :after-read='afterRead3'
    v-model="fileList3" :max-count="1" />
  </template>
</van-field>
```

2.2在data中声明需要双向数据绑定的变量

```
data () {
 return {
   form:{

   },
   fileList1:[],
   fileList2:[],
   fileList3:[]
 }
},
```

2.3在methods中声明:after-read='afterRead1'绑定的方法并实现

```
//file 对象
//detail 额外信息，包含 name 和 index 字段
afterRead1(file, detail) {
  console.log(this.dataURLtoFileFun(file.content));
  //定义一个formData表单对象
  let formData = new FormData();
  // 调用base64转文件对象方法进行格式转换
  formData.append('file',this.dataURLtoFileFun(file.conten
t,'file.jpg'))
  // 使用post_formData方法将包含文件的formData上传到文件服务器
文件服务器路径见swagger
  post_formData('http://121.199.29.84:5588/file/upload',fo
rmData).then(res=>{
    console.log(res.data);
    if(res.data.status === 200){
        //上传成功 将对应的图片路径保存到需要提交的表单对象中
        this.form.idcardPhotoPositive
='http://121.199.29.84:8888/'+res.data.data.groupname+'/'+
res.data.data.id
        // 保存已上传的文件路径
        this.fileList1 =
[{url:'http://121.199.29.84:8888/'+res.data.data.groupname
+'/'+res.data.data.id}]
    }
  })
}
```

base64转文件方法

```
// bae64转文件对象
dataURLtoFileFun (dataurl, filename) {
  // 将base64转换为文件，dataurl为base64字符串，filename为文件名（必须带后缀名，如.jpg,.png）
  let arr = dataurl.split(",");
  let mime = arr[0].match(/:(.*?);/)[1];
  let bstr = atob(arr[1]);
  let n = bstr.length;
  let u8arr = new Uint8Array(n);
  while (n--) {
    u8arr[n] = bstr.charCodeAt(n);
  }
  return new File([u8arr], filename, { type: mime });
},
```

2.4最终结合form表单提交，将对应的表单对象提交到对应接口即可

```
//提交审核
submitHandler(){
  this.form.userId = this.$route.query.id
  console.log(this.form);
  post('/certification/submitCertificationApply',this.form).then(res=>{
    console.log(res.data.status);
    if(res.data.status === 200){
      Toast(res.data.message)
      this.$router.push('/writer/mine')
    }else {
      Toast(res.data.message)
    }
  })
}
```

3.实名认证源码

```
<template>
  <div class='authentication'>
    <div class="title">
      <span><img @click="toBack" width="22px"
height="22px" src="../../../public/返回.png" alt=""></span>
      实名认证
    </div>
    <div class="tip">
      <div class="icon">
        <img src="../../../public/安全.png" alt="">
      </div>
      <p class="tip1">进行实名认证</p>
      <p class="tip2">用于保障账号的安全及快速接收订单信息</p>
    </div>
    <div class="form">
      <van-form>
        <van-field
            v-model="form.realname"
            name="真实姓名"
            label="真实姓名"
            placeholder="真实姓名"
            :rules="[{ required: true, message: '请填写真实
姓名' }]"
          />
        <van-field
          v-model="form.idCard"
          name="身份证号"
          label="身份证号"
          placeholder="身份证号"
          :rules="[{ required: true, message: '请填写身份证
号' }]"
```

```
        />
        <van-field
          v-model="form.bankCard"
          name="银行卡号"
          label="银行卡号"
          placeholder="银行卡号"
          :rules="[{ required: true, message: '请填写银行卡
号' }]"
        />
        <van-field name="uploader" label="身份证正面照">
          <template #input>
            <van-uploader :after-read='afterRead1' v-
model="fileList1" :max-count="1" />
          </template>
        </van-field>
        <van-field name="uploader" label="身份证反面照" >
          <template #input>
            <van-uploader :after-read='afterRead2' v-
model="fileList2" :max-count="1" />
          </template>
        </van-field>
        <van-field name="uploader" label="银行卡照片">
          <template #input>
            <van-uploader :after-read='afterRead3' v-
model="fileList3" :max-count="1" />
          </template>
        </van-field>
        <div class="btn" @click="submitHandler">
          提交
        </div>
      </van-form>
    </div>
  </div>
```

```
</template>

<script>
import {post_formData,post} from '../../http/axios'
import {Toast} from 'vant'
export default {
  // 组件名称
  name: 'demo',
  // 组件参数 接收来自父组件的数据
  props: {},
  // 组件状态值
  data () {
   return {
     form:{

     },
     fileList1:[],
     fileList2:[],
     fileList3:[]
   }
  },
  // 计算属性
  computed: {},
  // 侦听器
  watch: {},
  // 组件方法
  methods: {
    //返回上一页面
    toBack(){
      this.$router.push({path:'/writer/mine'})
    },
    afterRead1(file, detail) {
      console.log(this.dataURLtoFileFun(file.content));
```

```
      let formData = new FormData();

 formData.append('file',this.dataURLtoFileFun(file.content
,'file.jpg'))

 post_formData('http://121.199.29.84:5588/file/upload',for
mData).then(res=>{
        console.log(res.data);
        if(res.data.status === 200){
            this.form.idcardPhotoPositive
='http://121.199.29.84:8888/'+res.data.data.groupname+'/'+
res.data.data.id
            this.fileList1 =
[{url:'http://121.199.29.84:8888/'+res.data.data.groupname
+'/'+res.data.data.id}]
        }
      })
    },
    afterRead2(file, detail) {
      console.log(this.dataURLtoFileFun(file.content));
      let formData = new FormData();

 formData.append('file',this.dataURLtoFileFun(file.content
,'file.jpg'))

 post_formData('http://121.199.29.84:5588/file/upload',for
mData).then(res=>{
        console.log(res.data);
        if(res.data.status === 200){
            this.form.idcardPhotoNegative
='http://121.199.29.84:8888/'+res.data.data.groupname+'/'+
res.data.data.id
```

```
            this.fileList2 =
[{url:'http://121.199.29.84:8888/'+res.data.data.groupname
+'/'+res.data.data.id}]
          }
        })
      },
      afterRead3(file, detail) {
        console.log(file);
        let formData = new FormData();

 formData.append('file',this.dataURLtoFileFun(file.content
,'file.jpg'))

 post_formData('http://121.199.29.84:5588/file/upload',for
mData).then(res=>{
          console.log(res.data);
          if(res.data.status === 200){
              this.form.bankCardPhoto
='http://121.199.29.84:8888/'+res.data.data.groupname+'/'+
res.data.data.id
              this.fileList3 =
[{url:'http://121.199.29.84:8888/'+res.data.data.groupname
+'/'+res.data.data.id}]
          }
        })
      },
      // bae64转文件对象
      dataURLtoFileFun (dataurl, filename) {
        // 将base64转换为文件，dataurl为base64字符串，filename为
文件名（必须带后缀名，如.jpg,.png）
        let arr = dataurl.split(",");
        let mime = arr[0].match(/:(.*?);/)[1];
        let bstr = atob(arr[1]);
```

```
      let n = bstr.length;
      let u8arr = new Uint8Array(n);
      while (n--) {
        u8arr[n] = bstr.charCodeAt(n);
      }
      return new File([u8arr], filename, { type: mime });
    },
    //提交审核
    submitHandler(){
      this.form.userId = this.$route.query.id
      console.log(this.form);

 post('/certification/submitCertificationApply',this.form)
.then(res=>{
        console.log(res.data.status);
        if(res.data.status === 200){
          Toast(res.data.message)
          this.$router.push('/writer/mine')
        }else {
          Toast(res.data.message)
        }
      })
    }
  },
  // 以下是生命周期钩子
  /**
  * 组件实例创建完成，属性已绑定，但DOM还未生成，$ el属性还不存在
  */
  created () {
  },
  /**
  * el 被新创建的 vm.$ el 替换，并挂载到实例上去之后调用该钩子。
```

```
  * 如果 root 实例挂载了一个文档内元素，当 mounted 被调用时 vm.$
el 也在文档内。
  */
  mounted () {
  },
  /**
  * 实例销毁之前调用。在这一步，实例仍然完全可用。
  */
  beforeDestroy () {
  },
  /**
  * Vue 实例销毁后调用。调用后，Vue 实例指示的所有东西都会解绑定，
  * 所有的事件监听器会被移除，所有的子实例也会被销毁。
  */
  destroyed () {
  }
}
</script>

<!-- Add "scoped" attribute to limit CSS to this component
only -->
<!--使用了scoped属性之后，父组件的style样式将不会渗透到子组件中，-
->
<!--然而子组件的根节点元素会同时被设置了scoped的父css样式和设置了
scoped的子css样式影响，-->
<!--这么设计的目的是父组件可以对子组件根元素进行布局。-->
<style scoped>
  .authentication .title {
      height: 58px;
      background-image: linear-gradient(to
right,#BF73FF,#7579FF);
      text-align: center;
      line-height: 58px;
```

```
    color: #fff;
    letter-spacing: .2em;
}
.authentication .title span {
    float:left;
    margin-left:1em;
    margin-top:5px;
    cursor: pointer;
}
.tip .icon {
    width: 100px;
    height: 100px;
    margin: 0 auto;
    margin-top: 1em;
    border-radius: 50%;
    background-color: #ddfae7;
}
.tip .icon img {
  width: 98%;
  height: 98%;
}
.tip .tip1 {
  width: 50%;
  margin: 0 auto;
  text-align: center;
  margin-top: 1em;
  color: black;
  font-weight: 300;
}
.tip .tip2 {
  width: 80%;
  margin: 0 auto;
  text-align: center;
```

```
    margin-top: 1em;
    color: #999;
    font-weight: 300;
    font-size: 12px;
    letter-spacing: .1em;
  }
  .form {
    margin-top: 2em;
    padding: 0 .5em;
  }
  .form .btn {
    height: 42px;
    background-image: linear-gradient(to
right,#BF73FF,#7579FF);
    margin: 16px;
    border-radius: 5px;
    text-align: center;
    line-height: 42px;
    color: #fff;
    font-size: 14px;
    letter-spacing: .2em;
    cursor: pointer;
  }
</style>
```