Vant

Vant 是**有赞前端团队**开源的移动端组件库，于 2017 年开源，已持续维护 4 年时间。Vant 对内承载了有赞所有核心业务，对外服务十多万开发者，是业界主流的移动端组件库之一。

https://vant-contrib.gitee.io/vant/#/zh-CN/home

安装使用

1.可以直接通过 vue add vant指令安装配置，过程类似vue add element

2.通过npm i vant -S指令安装，安装后再main.js中进行配置

```
import Vue from 'vue';
import Vant from 'vant';
import 'vant/lib/index.css';

Vue.use(Vant);
```

安装成功后可直接调用相应的组件

App基本架构地址：https://gitee.com/zachgmytea/briup-ej-app.git

项目基本结构

src/App.vue

项目全局根组件，只显示即可，其余路由跳转配置在Manager.vue中。

```
<template>
  <div id="app">
    <!-- 全局根组件 -->
    <!-- 只显示一个routerview 即可 -->
    <router-view></router-view>
  </div>
</template>

<script>
export default {
  name: "app",
};
</script>

<style>
/* 获取父元素高度 */
html,body,#app{
 height: 100%;
}
</style>
```

src/router/index.js

路由配置文件，将基本的页面路由进行配置

```
import Vue from 'vue'
import VueRouter from 'vue-router'
```

```
Vue.use(VueRouter)
// 防止路由重复点击报错
const originalPush = VueRouter.prototype.push
VueRouter.prototype.push = function push(location) {
  return originalPush.call(this, location).catch(err =>
err)
}

const routes = [
  // 默认加载界面 ，  重定向页面
  {
    path: '/',
    redirect: '/login'
  },
  // 登录页面界面
  {
    path: '/login',
    name: 'Login',
    component: () => import('@/views/Login.vue')
  },
  // 注册页面路由
  {
    path: '/register',
    name: 'Register',
    component: () => import('@/views/Register.vue')
  },
  // 总路由
  {
    path: '/manager',
    name: 'manager',
    component: () => import('@/views/Manager.vue'),
  },
]
```

```
const router = new VueRouter({
  // mode: 'history',
  base: process.env.BASE_URL,
  routes
})

// 路由守卫 判断有无token
// 使用 router.beforeEach 注册一个全局前置守卫，判断用户是否登陆
router.beforeEach((to, from, next) => {
  //若要跳转的页面是登录界面
  if (to.path === '/login') {
    //直接跳转
    next();
  } else { //若想要跳转其他页面
    //获取本地存储的token值
    let token = localStorage.getItem('token');
    //若token为空则验证不成功，跳转到登录页面
    if (token === 'null' || token === '') {
      next('/login');
    } else {
      next();
    }
  }
});

export default router
```

src/views/Manager.vue

管理者页面，在此配置App的tabbar，控制底部导航跳转

```
<template>
```

```
  <div class="manager">
    <!-- 用于接受管理页面，首页，订单页，我的页面 -->
    我是管理页面
  </div>
</template>

<script>
export default {
  data() {
    return {
      active: 0,
    };
  },
};
</script>

<style scoped>
</style>
```

src/views/Login.vue

登录页面，登录页面样式以及登录操作的实现

```
<template>
  <div class="login">
    <!-- 头部信息展示区 -->
    <div class="header">
      <div class="title">
        易洁家政
      </div>
    </div>
    <!-- 登录区域 -->
    <div class="loginArea">
```

```
      <div class="loginForm">
        <van-form @submit="onSubmit">
          <van-field
            v-model="username"
            name="用户名"
            label="用户名"
            placeholder="用户名"
            :rules="[{ required: true, message: '请填写用户
名' }]"
            size="large"
          />
          <van-field
            v-model="password"
            type="password"
            name="密码"
            label="密码"
            placeholder="密码"
            :rules="[{ required: true, message: '请填写密码'
}]"
            size="large"
          />
          <div style="margin: 16px">
            <van-button color="linear-gradient(to right,
#5098fa, #265aca)" round block type="info" native-
type="submit">
            提交
            </van-button>
          </div>
          <div class="signUp">
            <span @click="toRegister">没有账号？点击注册
</span>
          </div>
        </van-form>
```

```
      </div>
    </div>
  </div>
</template>

<script>
// 引入辅助函数
import { mapMutations } from 'vuex'
// 异步加载  引入axios
import { post_json } from '@/http/axios'
import { Toast } from 'vant';

export default {
  data() {
    return {
      username: "",
      password: "",
    };
  },
  methods: {
    // 引入vuex中的mutations
    ...mapMutations('user',['setToken']),
    // 登录
    async onSubmit(values) {
      // 配置参数
      let params = {
        username: this.username,
        password: this.password,
      }
      // 发送登录验证请求
      let res = await post_json('/user/login', params)
      // console.log(res);
      if (res.data.status == 200) {
```

```
          // 保存token
          this.setToken({token: res.data.data.token});
          // 跳转到首页
          this.$router.push('/manager')
        }else{
          // 提示错误信息
          Toast(res.data.message);
        }
      },
      // 跳转到注册
      toRegister(){
        this.$router.push('/register')
      }
    },
};
</script>

<style scoped>
/* 整体样式 */
.login {
  width: 100%;
  height: 100%;
  background-image: linear-gradient(#5098fa, #265aca);
  overflow: hidden;
}
/* 头部样式 */
.header{
  width: 100%;
  position: absolute;
  text-align: center;
  top: 120px;
}
/* 标题样式 */
```

```
.header .title{
  font-family: 'webfont';
  font-size: 40px;
  color: white;
}
@font-face {
  font-family: 'webfont';
  font-display: swap;
  src: url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.eot');
/* IE9*/
  src: url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.eot?
#iefix') format('embedded-opentype'), /* IE6-IE8 */
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.woff2')
format('woff2'),
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.woff')
format('woff'), /* chrome、firefox */
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.ttf')
format('truetype'), /* chrome、firefox、opera、Safari,
Android, iOS 4.2+*/
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.svg#AlibabaPu
HuiTiH') format('svg'); /* iOS 4.1- */
}
/* 中间登录输入区域 卡片样式 */
.loginArea {
  box-shadow: 0 4px 8px 0 rgba(255, 255, 255, 0.2);
  background-color: white;
  transition: 0.3s;
  width: 90%;
  border-radius: 5px;
  margin: 265px auto;
  height: 250px;
  padding: 10px;
}
```

```
.loginArea:hover {
  box-shadow: 0 8px 16px 0 rgba(0, 0, 0, 0.2);
}
/* 输入框整体样式 */
.loginForm{
  padding-top: 20px;
}
/* 注册样式 */
.signUp{
  color: #4E95F7;
  text-align: center;
}
</style>
```

src/views/Register.vue

注册页面，注册页面样式以及注册操作的实现

```
<template>
  <div class="login">
    <!-- 头部信息展示区 -->
    <div class="header">
      <div class="title">
        易洁家政
      </div>
    </div>
    <!-- 登录区域 -->
    <div class="loginArea">
      <div class="loginForm">
        <van-form @submit="onSubmit">
          <van-field
            v-model="username"
```

```
            name="用户名"
            label="用户名"
            placeholder="用户名"
            :rules="[{ required: true, message: '请填写用户
名' }]"
            size="large"
          />
          <van-field
            v-model="password"
            type="password"
            name="密码"
            label="密码"
            placeholder="密码"
            :rules="[{ required: true, message: '请填写密码'
}]"
            size="large"
          />
          <van-field
            v-model="confirmPwd"
            type="password"
            name="确认密码"
            label="确认密码"
            placeholder="确认密码"
            :rules="[{ required: true, message: '请再次输入
密码' }]"
            size="large"
          />
          <div style="margin: 16px">
            <van-button color="linear-gradient(to right,
#5098fa, #265aca)" round block type="info" native-
type="submit">
            注册
            </van-button>
```

```
          </div>
          <div class="signUp">
            <span @click="toLogin">已有账号，去登录</span>
          </div>
        </van-form>
      </div>
    </div>
  </div>
</template>

<script>
// 引入辅助函数
import { mapMutations, mapActions } from 'vuex'
// 异步加载  引入axios
import { post } from '@/http/axios'
import { Toast } from 'vant';

export default {
  data() {
    return {
      username: "",
      password: "",
      confirmPwd:''
    };
  },
  methods: {
    ...mapActions('register', ['toRegister']),
    onClickLeft() {
      this.$router.go(-1);
    },
    onSubmit(){
      if (this.password != this.confirmPwd) {
        Toast('两次密码输入不一致,请重新输入! ');
```

```
      }else{
        // 获取注册参数
        let params = {
          username: this.username,
          password: this.password,
        }
        // 调用注册请求
        this.toRegister(params)
        Toast('注册成功')
        setTimeout(() => {
          this.$router.go(-1)
        }, 1500);
      }
    },
    // 跳转到登录
    toLogin(){
      this.$router.push('/login')
    }
  },
};
</script>

<style scoped>
/* 整体样式 */
.login {
  width: 100%;
  height: 100%;
  background-image: linear-gradient(#5098fa, #265aca);
  overflow: hidden;
}
/* 头部样式 */
.header{
  width: 100%;
```

```
  position: absolute;
  text-align: center;
  top: 120px;
}
/* 标题样式 */
.header .title{
  font-family: 'webfont';
  font-size: 40px;
  color: white;
}
.header .title_bottom {
  color: #fff;
  font-family: 'webfont';
  font-size: 20px;
}
@font-face {
  font-family: 'webfont';
  font-display: swap;
  src: url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.eot');
/* IE9*/
  src: url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.eot?
#iefix') format('embedded-opentype'), /* IE6-IE8 */
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.woff2')
format('woff2'),
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.woff')
format('woff'), /* chrome、firefox */
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.ttf')
format('truetype'), /* chrome、firefox、opera、Safari,
Android, iOS 4.2+*/
  url('//at.alicdn.com/t/webfont_mtjyk0oq5ef.svg#AlibabaPu
HuiTiH') format('svg'); /* iOS 4.1- */
}
/* 中间登录输入区域 卡片样式 */
```

```
.loginArea {
  box-shadow: 0 4px 8px 0 rgba(255, 255, 255, 0.2);
  background-color: white;
  transition: 0.3s;
  width: 90%;
  border-radius: 5px;
  margin: 265px auto;
  height: 250px;
  padding: 10px;
}
.loginArea:hover {
  box-shadow: 0 8px 16px 0 rgba(0, 0, 0, 0.2);
}
/* 输入框整体样式 */
.loginForm{
  padding-top: 20px;
}
/* 注册样式 */
.signUp{
  color: #4E95F7;
  text-align: center;
  font-size: 12px;
  letter-spacing: .05em;
  cursor: pointer;
}
</style>
```

src/http/axios.js

项目封装axios文件，将接口地址更改为自己的可直接使用

```
import axios from 'axios'
```

```
import router from '../router/index'
import Vue from 'vue';
import { Toast } from 'vant';
import qs from 'qs'

Vue.use(Toast);

let token = '';
// 1. 基础路径 配置为自己的服务器地址
axios.defaults.baseURL = 'http://8.134.121.225:8002/'
//配置请求头
axios.defaults.headers.post['Content-Type'] =
'application/json';

// http request拦截器 添加一个请求拦截器
axios.interceptors.request.use(function (config) {
  let token = localStorage.getItem("token")
  if (token) {
    //将token放到请求头发送给服务器,将token放在请求头中
    config.headers['Authorization'] = token
  }
  return config;
}, function (error) {
  Toast.fail('请求超时');
  // Do something with request error
  return Promise.reject(error);
});

// 添加一个响应拦截器
axios.interceptors.response.use(function (response) {
  return response;
}, function (error) {
  Toast.fail("服务器连接失败");
```

```
  return Promise.reject(error);
})

/**
  get方式请求
*/
export function get(url, params) {
  return axios({
    method: 'get',
    url,
    params, // get 请求时带的参数
    timeout: 5000
  })
}

/**
 * 提交post请求 发送的数据为查询字符串，key=val&key=val
*/
export function post(url, data) {
  return axios({
    method: "post",
    url,
    data: qs.stringify(data),
    timeout: 5000,
    headers: {
      'Content-Type': 'application/x-www-form-urlencoded'
    }
  })
}

/**
 * 提交post请求 ,查询字符串，对象中嵌套数组的格式
*/
```

```
export function post_obj_array(url, data) {
  return axios({
    method: "post",
    url,
    data: qs.stringify(data, { allowDots: true }),
    timeout: 5000
  })
}

/**
 * 提交post请求 发送的数据为查询字符串，当参数为数组的时候适用该方法
 * ids=1&ids=2
*/
export function post_array(url, data) {
  return axios({
    method: "post",
    url,
    data: qs.stringify(data, { arrayFormat: "repeat" }),
    timeout: 5000
  })
}
/**
 * 提交post请求 发送的数据为json字符串
*/
export function post_json(url, data) {
  return axios({
    method: "post",
    url,
    data,
    timeout: 5000
  })
}
```

```
export default axios
```

可能会用到的

渐变色使用方式

background-image: linear-gradient(#5098fa, #265aca);

卡片阴影样式

box-shadow: 0 0 0 0 rgba(255, 255, 255, 0.2);

渐变色网站

https://uigradients.com/#BlurryBeach

https://mycolor.space/

https://webgradients.com

接口

app-顾客端

登录，注册，退出接口

user-controller : 登录相关接口

Show/Hide | List Operations | Expand Operations

GET	/user/info	通过token获取用户的基本信息
POST	/user/login	login
POST	/user/logout	logout
POST	/user/register	顾客注册
POST	/user/registerEmployee	员工注册

登录接口

退出接口

注册接口

首页

轮播图

carousel-controller : 轮播图相关接口

Show/Hide | List Operations | Expand Operations

GET	/carousel/deleteById	通过id删除
GET	/carousel/findAll	查询所有轮播图
GET	/carousel/query	查询所有轮播图
POST	/carousel/saveOrUpdate	保存或更新

传 正常 参数

栏目菜单

product-category-controller : jz-产品分类相关接口

Show/Hide | List Operations | Expand Operations

GET	/productCategory/deleteById	通过id删除产品分类
GET	/productCategory/pageQuery	分页查询产品分类相关信息
POST	/productCategory/saveOrUpdate	保存或更新产品分类信息

产品展示

product-controller : jz-产品相关接口

Show/Hide | List Operations | Expand Operations

GET	~~/product/deleteById~~	通过id删除
GET	/product/offline	下架
GET	/product/online	上架
GET	/product/pageQuery	分页查询产品相关信息
POST	/product/saveOrUpdate	保存或更新产品信息

服务产品页面

侧边导航栏

product-category-controller : jz-产品分类相关接口 Show/Hide | List Operations | Expand Operations

GET	/productCategory/deleteById	通过id删除产品分类
GET	/productCategory/pageQuery	分页查询产品分类相关信息
POST	/productCategory/saveOrUpdate	保存或更新产品分类信息

根据点击导航栏获取产品数据

product-controller : jz-产品相关接口 Show/Hide | List Operations | Expand Operations

GET	~~/product/deleteById~~	通过id删除
GET	/product/offline	下架
GET	/product/online	上架
GET	/product/pageQuery 需要传递栏目id	分页查询产品相关信息
POST	/product/saveOrUpdate	保存或更新产品信息

订单确认页面

地址相关信息

address-controller : jz-地址相关接口 Show/Hide | List Operations | Expand Operations

GET	/address/deleteById	通过id删除
GET	/address/pageQuery	分页查询地址相关信息
POST	/address/saveOrUpdate	保存或更新地址信息

用户相关信息

user-controller : 登录相关接口 Show/Hide | List Operations | Expand Operations

GET	/user/info	通过token获取用户的基本信息
POST	/user/login	login
POST	/user/logout	logout
POST	/user/register	顾客注册
POST	/user/registerEmployee	员工注册

提交订单

order-controller : jz-订单相关接口 Show/Hide | List Operations | Expand Operations

GET	/order/cancelSendOrder	取消派单
GET	/order/confirmOrder	确认订单
GET	/order/deleteById	通过id删除
GET	/order/findById	通过id查询订单详情
GET	/order/pageQuery	分页查询订单相关信息
GET	/order/rejectOrder	拒绝订单
GET	/order/sendOrder	派单
GET	/order/serviceCompleteOrder	员工服务结束
POST	/order/submitOrder	提交订单
GET	/order/takeOrder	接单

余额充值

account-controller : jz-账户相关接口 Show/Hide | List Operations | Expand Operations

GET	/account/pageQueryCustomerAccount	分页查询顾客账单
GET	/account/pageQueryEmployeeAccount	分页查询员工账单
GET	/account/pageQuerySystemAccount	分页查询系统账单
GET	/account/recharge	顾客充值

订单界面

所有状态订单查询，订单详情等接口

order-controller : jz-订单相关接口 Show/Hide | List Operations | Expand Operations

GET	/order/cancelSendOrder	取消派单
GET	/order/confirmOrder	确认订单
GET	/order/deleteById	通过id删除
GET	/order/findById	通过id查询订单详情
GET	/order/pageQuery	分页查询订单相关信息
GET	/order/rejectOrder	拒绝订单
GET	/order/sendOrder	派单
GET	/order/serviceCompleteOrder	员工服务结束
POST	/order/submitOrder	提交订单
GET	/order/takeOrder	接单

帮助页面

此页面目前只是一个假数据展示页面，可自定义

我的页面

用户个人信息

user-controller : 登录相关接口 Show/Hide | List Operations | Expand Operations

GET	/user/info	通过token获取用户的基本信息
POST	/user/login	login
POST	/user/logout	logout
POST	/user/register	顾客注册
POST	/user/registerEmployee	员工注册

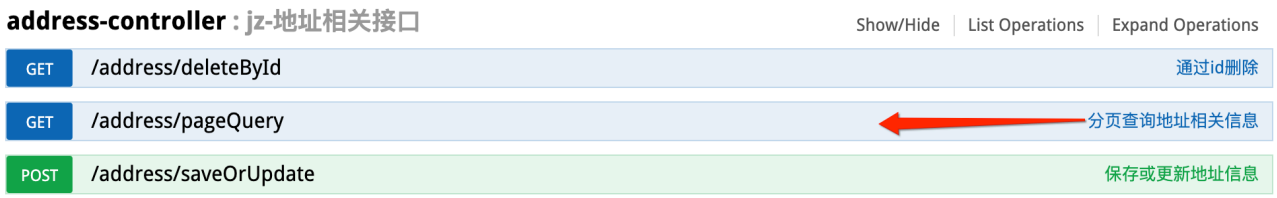
充值

account-controller : jz-账户相关接口 Show/Hide | List Operations | Expand Operations

GET	/account/pageQueryCustomerAccount	分页查询顾客账单
GET	/account/pageQueryEmployeeAccount	分页查询员工账单
GET	/account/pageQuerySystemAccount	分页查询系统账单
GET	/account/recharge	顾客充值

常用地址页面

地址查询



新增地址