软件工程项目个人总结报告

学院：软件学院

姓名：谢志伟

班级：卓越工程师班

学号：55201026

指导老师：朱晓冬

撰写时间：2023.2

一、 工作概述

项目组在经过总体的需求分析、用例分析之后开始对页面进行设计与布局。

在本项目中，本人主要工作为前端页面的编写，图 1 是项目前端页面总图。

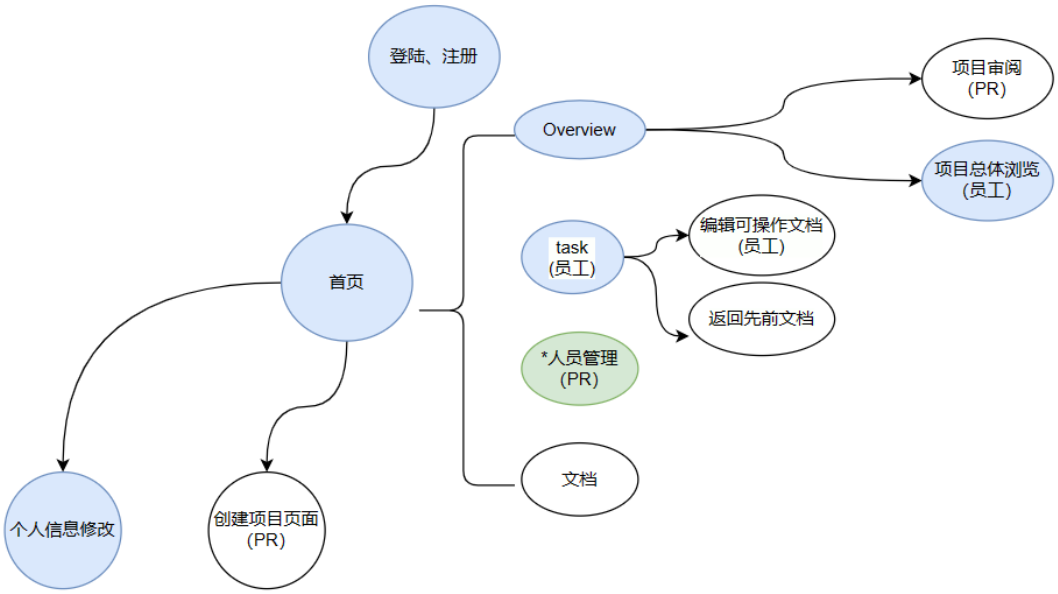


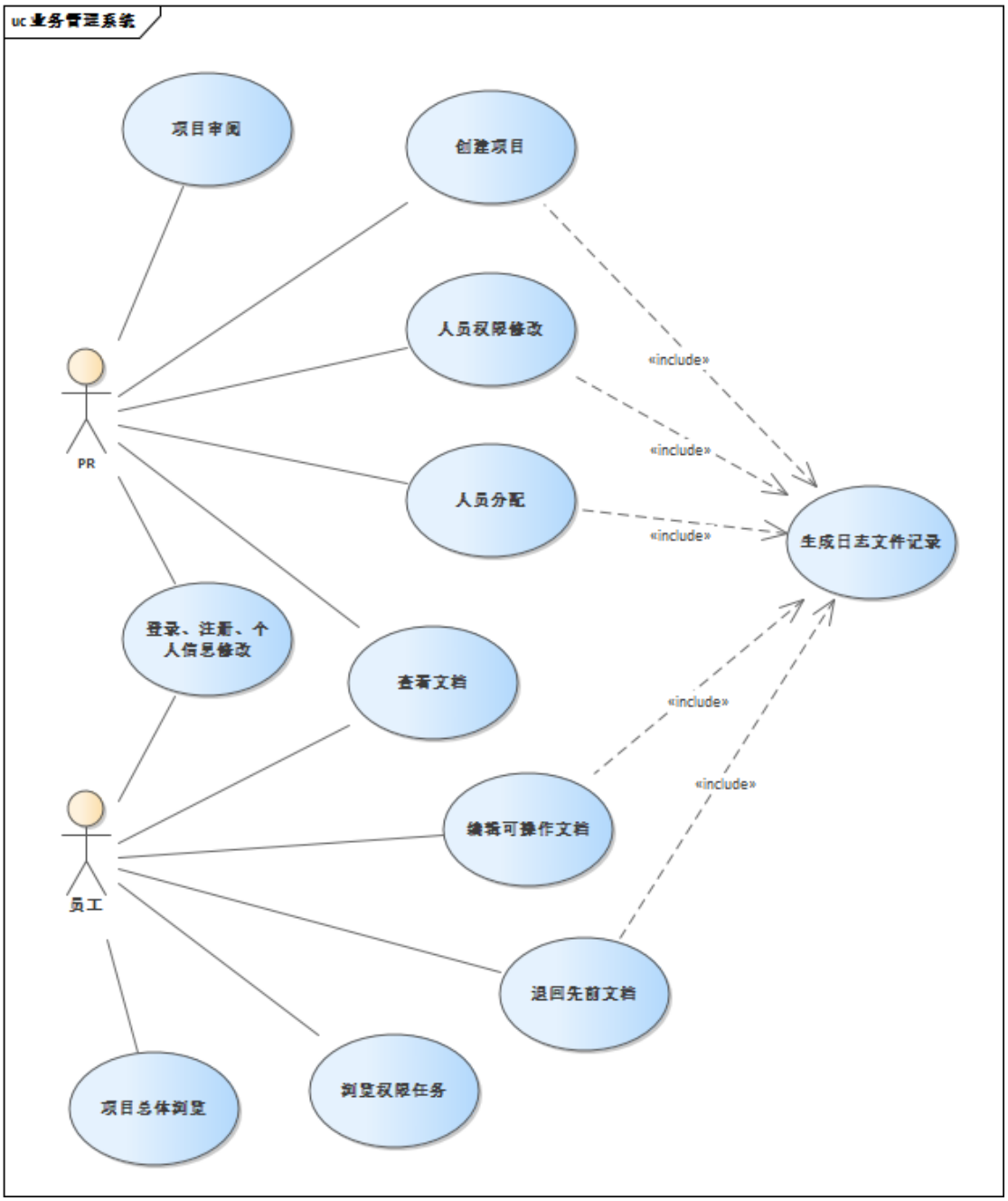
图 1 项目前端页面模块总图

其中蓝色部分为本人所负责模块，包括登陆注册、首页菜单栏、个人信息、项目总体浏览、员工个人任务管理等功能模块的前端部分，绿色部分为与项目成员单嘉澳沟通实现。此外还有路由配置、接口分层等整合工作。

项目组前端所用框架为 Vue2，本人使用了 ElementUI 组件库、less 及其它 js 插件用于辅助开发。

二、 需求分析

项目主要功能模块用例图：



对于登陆注册页面，界面应该简单直观，此外，需要提供必要的帮助和支持，如登陆注册页面切换等选项，验证用户的用户名和密码是否匹配，并且需要提供必要的错误消息和提示，以指导用户输入正确的信息。在注册页面，需要根据用户在身份选项中的选择，获取现有组名列表供用户选择。

对于项目首页，首先需要根据用户权限提供菜单导航栏，对于创建项目页面和人员管理页面，只有权限身份为项目经理时才会有相应的导航选项。其次在项目头部中，提供面包屑、用户头像等优化用户体验的组件。最后，项目首页提供项目总体浏览功能，可以总体浏览当前用户活跃项目，并提供进入审阅该项目的入口。

对于任务管理页面，提供所有对用户当前活跃的任务清单展示，并提供自定义拖拽排序功能，并提供进入该任务文档的编辑入口。

对于个人信息修改页面，提供对用户个人信息的展示与修改，并支持头像图片上传等其它优化体验的功能。

三、 组件划分与层次划分

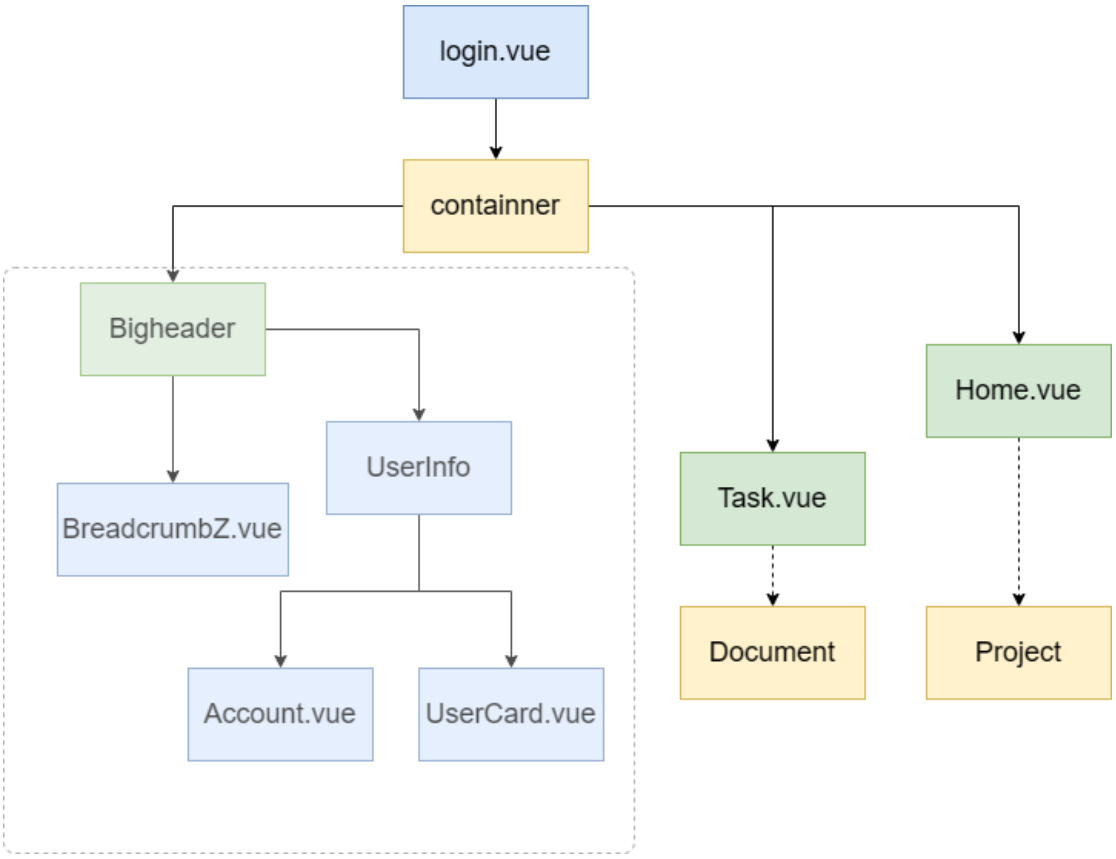


图 2 部分组件关系图

图 2 中蓝色模块与绿色模块是本人负责模块部分的组件关系图，用户通过 login.vue 组件完成注册与登陆操作，进入系统，初始首页为 Home.vue，通过顶部菜单导航栏可跳转到个人任务管理 Task.vue 及其它项目页面。顶部导航栏中的面包屑组件可实现页面路径的记忆与跳转。通过顶部个人头像可进入个人信息管理页面，其含 Account.vue 与 Usercard.vue 两子组件。

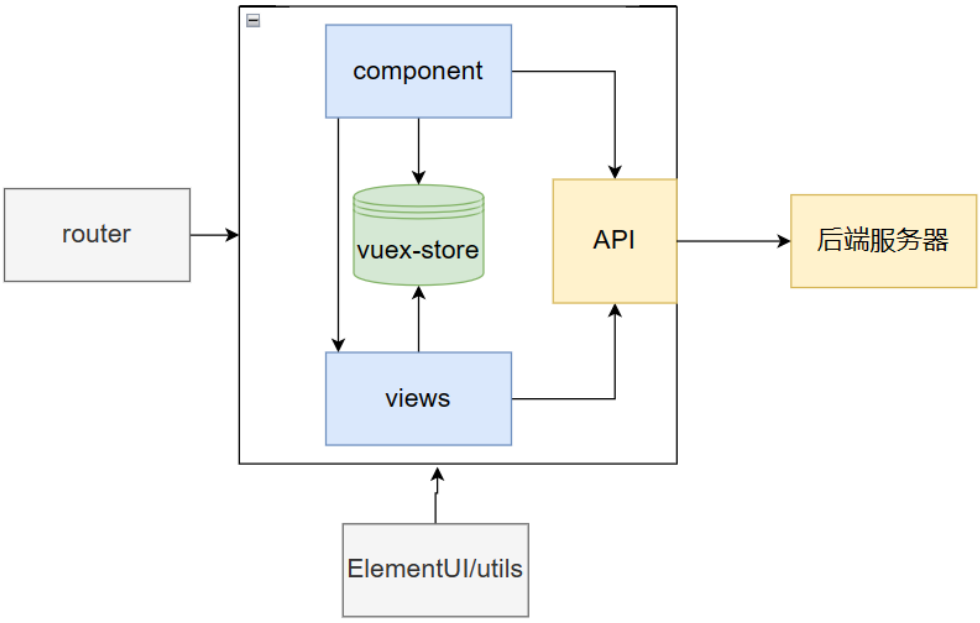
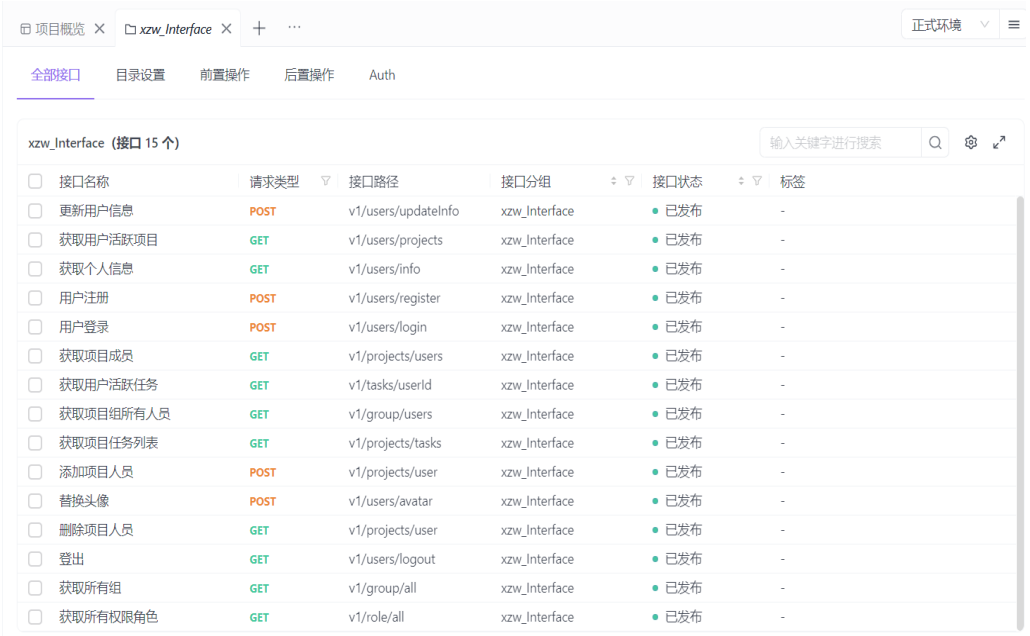


图 3 项目层次划分

- 在实现的过程中，为了便于分层管理，将整个项目划分了 api 层、router 层、store 层、 Component 层、views 层和 Util 层。
- 其中 api 层封装了 axios 示例，提供了一个公共接口，统一编写异常处理。api 层在此基础上有进行一次封装，提供了 Restful API 风格的接口。最后 api 层将所有的接口写在 api/api.js 中，为 Component 层与 views 层组件所服务。
- router 层封装了所有的路由，系统想要访问组件时需要通过路由层进行统一的访问。
- store 层使用 Vuex 技术封装了一个全局总线对象，提供了 Mutations 和 Getters 的方法，为组件提供组件间通信服务。
- views 层负责实现业务流程，展示视图。
- Util 层封装一些工具类和工具函数，提供给其他层，便于复用。

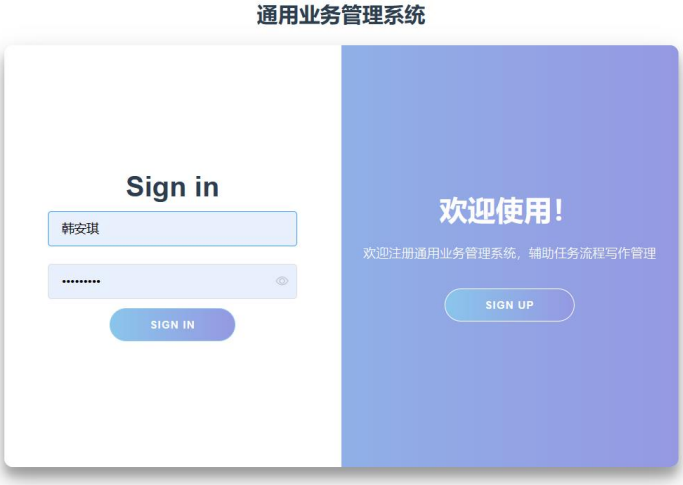
四、 重点功能与代码实现

相关接口：



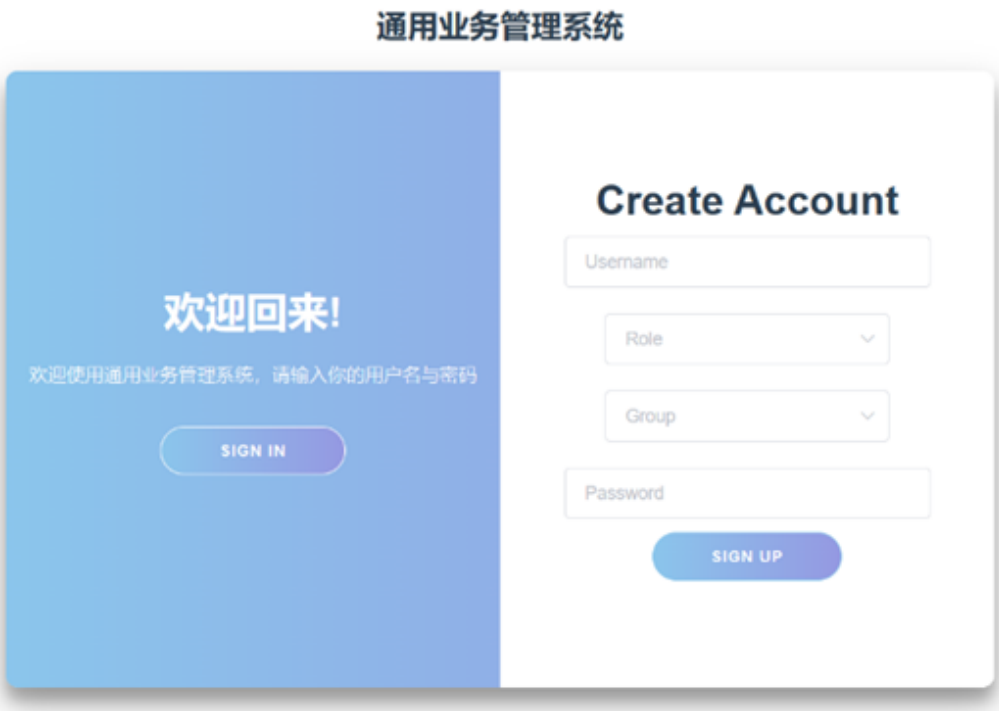
接口名称	请求类型	接口路径	接口分组	接口状态	标签
更新用户信息	POST	v1/users/updateInfo	xzw_Interface	已发布	-
获取用户活跃项目	GET	v1/users/projects	xzw_Interface	已发布	-
获取个人信息	GET	v1/users/info	xzw_Interface	已发布	-
用户注册	POST	v1/users/register	xzw_Interface	已发布	-
用户登录	POST	v1/users/login	xzw_Interface	已发布	-
获取项目成员	GET	v1/projects/users	xzw_Interface	已发布	-
获取用户活跃任务	GET	v1/tasks/userId	xzw_Interface	已发布	-
获取项目组所有人员	GET	v1/group/users	xzw_Interface	已发布	-
获取项目任务列表	GET	v1/projects/tasks	xzw_Interface	已发布	-
添加项目人员	POST	v1/projects/user	xzw_Interface	已发布	-
替换头像	POST	v1/users/avatar	xzw_Interface	已发布	-
删除项目人员	GET	v1/projects/user	xzw_Interface	已发布	-
登出	GET	v1/users/logout	xzw_Interface	已发布	-
获取所有组	GET	v1/group/all	xzw_Interface	已发布	-
获取所有权限角色	GET	v1/role/all	xzw_Interface	已发布	-

功能一：项目登录



登录页面，两个输入框为用户名与密码输入框，点击 sign in 登录，对输入框进行格式检查后发起登录请求并根据后端处理结果给出相应提示，点击 sign up 按钮可切换注册页面

功能二：项目注册



注册页面，其中 role 为下拉选项，有项目经理与普通员工两个选项，group 聚焦时会请求项目组数据，并展示为下拉选项，点击 sign up 注册，对输入框进行格式检查后发起注册请求并根据后端处理结果给出相应提示，点击 sign in 切换登录页面

关键代码：页面切换与登录事件

```
        addNewCss() {
            console.log("add")
            const container = document.getElementById('container')
            container.classList.add("right-panel-active")
        },
        removeNewCss() {
            console.log("remove")
            const container = document.getElementById('container')
```

```
            container.classList.remove("right-panel-active")
        },
        loginHandle() {
            // 登录事件
            if (!this.loginData.name || !this.loginData.password) {
                this.$message.error('用户名或密码不能为空')
                return
            }
            // let that = this
            login(this.loginData).then(res => {
                //Cookie.set('token', res.data.data.token)
                console.log("login.res:", res.data)
                localStorage.setItem("token", res.data.data.token)

                this.$store.commit('xzwxzw/UPDATEUSERID',
res.data.data.userId)

                console.log("userId::::",
this.$store.state.xzwxzw.userInfo.userId)


                getInfo((res.data.data.userId)).then(res => {
                    console.log("getInfo", res.data)
                    this.$store.commit('xzwxzw/updateUserInfo',
res.data.data)
                })

                // 更新
                console.log("userInfo",
this.$store.state.xzwxzw.userInfo)
                this.$router.push('/')
            }).catch(res => {
                this.$message.error(res.response.data.message)
            })
            // this.$store.commit('xzwxzw/updateUserId', this.userId)
            // 获取个人信息
            console.log("userId",
this.$store.state.xzwxzw.userInfo.userId)
        },
```

关键代码：注册页组数据获取

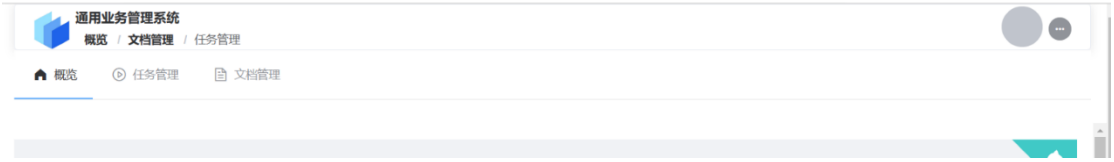
```
getGroups() {
            console.log("groupSearching...")
            // 向后端请求 group 信息
```

```
            getgroups().then(res => {
                // 将后端传回的 group 信息存储到 GroupOptions 中
                this.GroupOptions = []
                for (let i = 0; i < res.data.data.groups.length; i++) {
                    this.GroupOptions.push({
                        groupId: res.data.data[i].groupsId,
                        name: res.data.data[i].name,
                        memberCount: res.data.data[i].memberCount
                    })
                }
                console.log("group")
            }).catch(err => {
                // 处理错误信息
            })
        }
```

功能三：权限菜单栏

项目经理：

普通员工：



对于创建项目页面和人员管理页面，只有权限身份为项目经理时才会有相应的导航选项，通过 vue 的页面选择渲染与路由表的动态注册实现。在项目头部中，提供面包屑、用户头像等优化用户体验的组件。通过面包屑可跳转到对应页面，通过头像处可跳转到个人信息页面。

关键代码：导航栏选择渲染

```
<el-menu-item @click="clickMenu(item)" v-for="item in noChildren"
                v-if="user.role >=
item.role" :key="item.name" :index="item.name">
```

```
                <i :class="`el-icon-${item.icon}`"></i>
                <span slot="title">{{ item.label }}</span>
            </el-menu-item>
```

关键代码：动态注册路由表

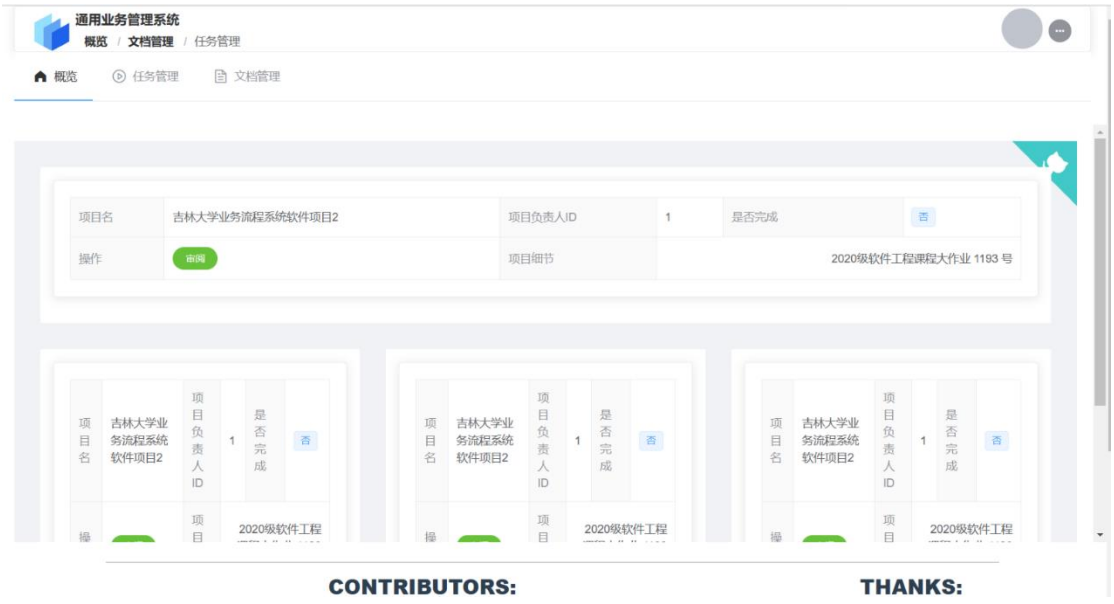
```
const router = new VueRouter({
    routes: [
        // 初始路由表
        { path: '/home', name: 'Home', component: Home },
        { path: '/login', name: 'login', component: Login },
        { path: '*', name: 'Error404', component: () =>
import('@/views/Error404.vue') }
    ]
})

// 项目经理路由表数据
const PRRoutes = [
    { path: '/user', component: User }, // 人员管理
    { path: '/document', component: Document }, // 文档管理
    { path: '/task', component: Task }, // 任务管理
    { path: '/cerater', component: Creater },
    { path: '/userinfo', name: 'userInfor', component: UserInfo },// 个
人信息
    { path: '/create', component: ProjectCreate },
    { path: '/preview', component: ProjectPreview },
    { path: '/documentShow', component: DocumentShow }, // 文档展示
    { path: '/useredit', component: UserEdit }, // 人员编辑
]
// 普通员工路由表数据
const comRoutes = [
    { path: '/document', component: Document }, // 文档管理
    { path: '/documentShow', component: DocumentShow }, // 文档展示
    { path: '/userinfo', name: 'userInfor', component: UserInfo },// 个
人信息
    { path: '/task', component: Task }, // 任务管理
]
```

关键代码：面包屑

```
<el-breadcrumb style="margin-top : -10px;" separator="/">
    <el-breadcrumb-item v-for="item in tablist" :key="item.path"
                                    :to="{ path: item.path }">
                                    {{ item.label }}
                                </el-breadcrumb-item>
                            </el-breadcrumb>
```

```
clickMenu(item) {
            console.log("clickmenu",item)
            if (this.$route.path !== item.path && !(this.$route.path ===
'\home' && (item.path === '/'))) {
                this.$router.push(item.path)
            }
            this.$store.commit('xzwxzw/selectMenu', item)
        }
// 更新面包屑数据
        selectMenu(state, val) {
            console.log("selectMenu",val)
            if (val.name !== 'home') {
                const index = state.tablist.findIndex(item => item.name
=== val.name)
                if (index === -1) {
                    state.tablist.push(val)
                }
            }
        },
```

功能四：项目浏览



主页 main 部分为项目卡片，可通过审阅按钮跳转到项目审阅部分，右上角为

github 角标，可直接跳转到本项目的 github 库

关键代码：角标

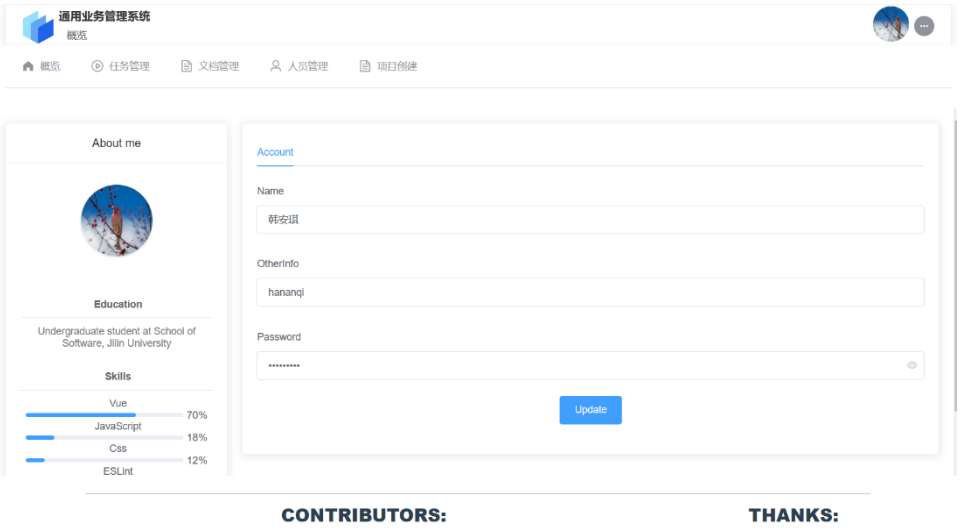
```
<template>
    <a href="https://github.com/RelaxDegree/Business_Process_System"
target="_blank" class="github-corner"
        aria-label="View source on Github">
        <svg width="80" height="80" viewBox="0 0 250 250"
style="fill:#40c9c6; color:#fff;" aria-hidden="true">
            <path d="M0,0 L115,115 L130,115 L142,142 L250,250 L250,0 Z"
/>
            <path
                d="M128.3,109.0 C113.8,99.7 119.0,89.6 119.0,89.6
C122.0,82.7 120.5,78.6 120.5,78.6 C119.2,72.0 123.4,76.3 123.4,76.3
C127.3,80.9 125.5,87.3 125.5,87.3 C122.9,97.6 130.6,101.9 134.4,103.2"
                fill="currentColor" style="transform-origin: 130px
106px;" class="octo-arm" />
            <path
                d="M115.0,115.0 C114.9,115.1 118.7,116.5 119.8,115.4
L133.7,101.6 C136.9,99.2 139.9,98.4 142.2,98.6 C133.8,88.0 127.5,74.4
143.8,58.0 C148.5,53.4 154.0,51.2 159.7,51.0 C160.3,49.4 163.2,43.6
171.4,40.1 C171.4,40.1 176.1,42.5 178.8,56.2 C183.1,58.6 187.2,61.8
190.9,65.4 C194.5,69.0 197.7,73.2 200.1,77.6 C213.8,80.2 216.3,84.9
216.3,84.9 C212.7,93.1 206.9,96.0 205.4,96.6 C205.1,102.4 203.0,107.8
198.3,112.5 C181.9,128.9 168.3,122.5 157.7,114.1 C157.9,116.9
156.7,120.9 152.7,124.9 L141.0,136.5 C139.8,137.7 141.6,141.9
141.8,141.8 Z"
                fill="currentColor" class="octo-body" />
        </svg>
    </a>
</template>
```

关键代码：项目卡片

```
<template>
    <el-card class="box-card" shadow="always" body-style="padding: '0px
0px 0px'">
    <el-descriptions :column="3" border class="box-table">
        <el-descriptions-item label="项目名" label-class-name="my-label"
            content-class-name="my-content"></el-descriptions-item>
        <el-descriptions-item label="项目负责人 ID"></el-descriptions-
item>
        <el-descriptions-item label="是否完成">
            <el-tag size="small"></el-tag>
```

```
        </el-descriptions-item>
        <el-descriptions-item label="操作">
                <el-button size="mini" type="success" round
@click="handleDetail()"></el-button>
        </el-descriptions-item>
        <el-descriptions-item label="项目细节" :contentStyle="{ 'text-
align': 'right' }"> </el-descriptions-item>
    </el-descriptions>
    </el-card>
</template>
```

功能五：个人信息修改



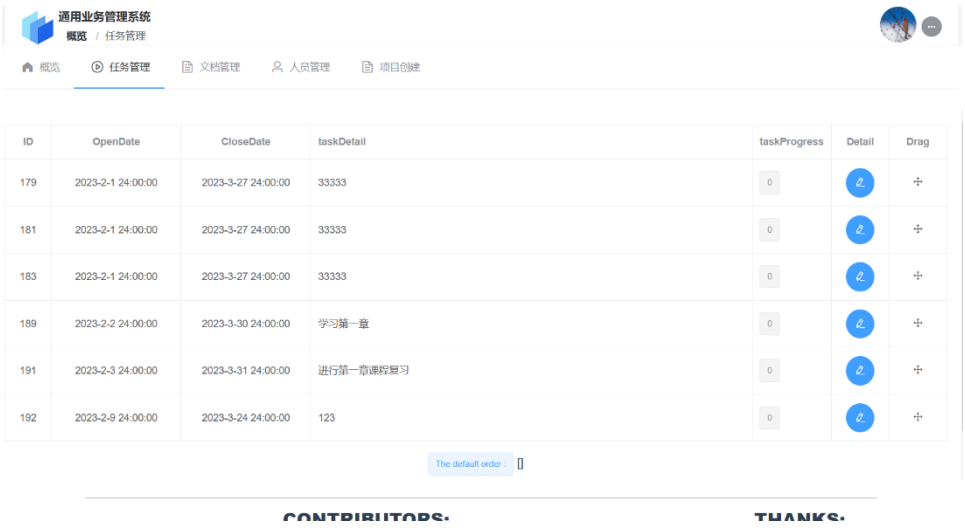
个人信息页面，可在此对自身个人信息进行修改

关键代码：

```
<template>
    <el-form>
        <el-form-item label="Name">
            <el-input v-model.trim="userdata.name" />
        </el-form-item>
        <el-form-item label="OtherInfo">
            <el-input v-model.trim="userdata.otherInfo" />
        </el-form-item>
        <el-form-item label="Password">
            <el-input placeholder="请输入密码" v-
model.trim="userdata.password" show-password></el-input>
```

```
        </el-form-item>
        <el-form-item>
            <el-button type="primary" @click="submit">Update</el-button>
        </el-form-item>
    </el-form>
</template>
```

功能六：任务管理页面



该页面会展示对当前用户所有活跃的任务，可通过右侧 drag 图标自定义拖拽排序，点击 Detail 按钮会进入当前任务的文档编辑页面

关键代码：列表自定义排序

```
created() {
        this.getList()
    },
    computed: {
        ...mapState({
            userId: state => state.xzwxzw.userInfo.userId
        })
    },
    methods: {
        async getList() {
            //this.listLoading = true
            // const { data } = await fetchList(this.listQuery)
            // this.list = data.items
            getActTask(this.userId).then(res => {
```

```
                this.list = res.data.data
                console.log("listdata:",res.data.data)
            })
            // this.total = data.total
             this.listLoading = false
            this.oldList = this.list.map(v => v.id)
            this.newList = this.oldList.slice()
            this.$nextTick(() => {
                this.setSort()
            })

        },
        setSort() {
            const el = this.$refs.dragTable.$el.querySelectorAll('.el-
table__body-wrapper > table > tbody')[0]
            this.sortable = Sortable.create(el, {
                ghostClass: 'sortable-ghost', // Class name for the drop
placeholder,
                setData: function (dataTransfer) {
                    // to avoid Firefox bug
                    dataTransfer.setData('Text', '')
                },
                onEnd: evt => {
                    const targetRow = this.list.splice(evt.oldIndex,
1)[0]
                    this.list.splice(evt.newIndex, 0, targetRow)
                }
            })
        },
```

关键代码：拖拽列表

```
        <el-table ref="dragTable" v-loading="listLoading" :data="list"
row-key="id" border fit highlight-current-row
            style="width: 100%">
            <el-table-column align="center" label="ID" width="65">
                <template slot-scope="{row}">
                    <span>{{ row.taskId }}</span>
                </template>
            </el-table-column>

            <el-table-column width="180px" align="center"
label="OpenDate">
                <template slot-scope="{row}">
```

```
                    <span>{{ row.taskOpenTime | parseTime('{y}-{m}-{d}
{h}:{i}') }}</span>
                </template>
            </el-table-column>

            <el-table-column width="180px" align="center"
label="CloseDate">
                <template slot-scope="{row}">
                    <span>{{ row.taskCloseTime | parseTime('{y}-{m}-{d}
{h}:{i}') }}</span>
                </template>
            </el-table-column>

            <el-table-column min-width="300px" label="taskDetail">
                <template slot-scope="{row}">
                    <span>{{ row.taskDetail }}</span>
                </template>
            </el-table-column>

            <el-table-column class-name="status-col"
label="taskProgress" width="110">
                <template slot-scope="{row}">
                    <el-tag :type="row.taskProgress | statusFilter">
                        {{ row.taskProgress }}
                    </el-tag>
                </template>
            </el-table-column>

            <el-table-column align="center" label="Detail" width="80">
                <template slot-scope="{}">
                    <el-button type="primary" icon="el-icon-edit"
@click="editor()" circle></el-button>
                </template>
            </el-table-column>

            <el-table-column align="center" label="Drag" width="80">
                <template slot-scope="{}">
                    <i class="el-icon-rank"></i>
                </template>
            </el-table-column>
        </el-table>
```

相关 Store 层代码：xzwxzw 模块

```
export default {
    namespaced : true,
    state: {
        isCollapse: false, // 菜单展开
        tablist: [
            {
                path: '/',
                name: 'home',
                label: '概览',
                icon: 's-home',
                url: 'Home/Home'
            }
        ], // 面包屑数据
        userInfo: {
            userId: 1,
            name:'zzxx',
            password: 'wqqwweq',
            otherInfo: 'qwqwwwwwwwwwwwww',
            groupId: 1,
            headPic: ''
        },
        menu: []
    },
    mutations: {
        // 修改菜单展开状态
        CollapseMenu(state) {
            state.isCollapse = !state.isCollapse
        },
        // 更新面包屑数据
        selectMenu(state, val) {
            console.log("selectMenu",val)
            if (val.name !== 'home') {
                const index = state.tablist.findIndex(item => item.name
=== val.name)
                if (index === -1) {
                    state.tablist.push(val)
                }
            }
        },
        // 更新个人信息
        updateUserInfo(state,userinfo) {
            state.userInfo.name = userinfo.name
            state.userInfo.password = userinfo.password
            state.userInfo.otherInfo = userinfo.otherInfo
```

```
            state.userInfo.groupId = userinfo.groupId
            //state.userInfo.headPic = userinfo.headPic
        },
        // 头像
        updateAvatar(state,avatar) {
            state.userInfo.avatar = avatar
        },
        UPDATEUSERID(state,userId) {
            console.log("update 调用",userId)
            state.userInfo.userId = userId
            console.log("-----------",state.userInfo.userId)

        }

    },
    actions:{

    }
}
```

相关 api 层代码：login.js

```
// 用户登录
export function login(data) {
    console.log(data)
    return request({
        url: 'api/v1/users/login',
        method: 'post',
        data: data
    })
}

// 获取个人信息
export function getInfo(userId) {
    return request({
        url: 'api/v1/users/info',
        method: 'get',
        params: {userId}
    })
}

// 用户登出
export function logout() {
    return request({
```

```
        url: 'api/v1/users/logout',
        method: 'get'
    })
}

// 获取所有组
export function getgroups() {
    return request({
        url: 'api/v1/group/all',
        method: 'get'
    })
}
// 用户注册
export function register(data) {
    return request({
        url: 'api/v1/users/register',
        method: 'post',
        data: data
    })
}
```

相关 api 层代码：user.js

```
// 获取项目组成员
export function getUser(groupName) {
    return request({
        url: 'api/v1/group/users',
        method: 'get',
        params : { groupName }
    })
}

// 获取项目成员
export function getProUser(id) {
    return request({
        url: 'api/v1/projects/users',
        method: 'get',
        params : { id }
    })
}

// 获取项目任务列表
export function getProTask( id ) {
    return request({
```

```
        url: 'api/v1/projects/tasks',
        method: 'get',
        params : { id }
    })
}

// 添加项目人员
export function addProUser(data) {
    return request({
        url: 'api/v1/projects/user',
        method: 'post',
        data
    })
}

// 删除项目人员
export function delProUser(userId, taskId) {
    return request({
        url: `api/v1/projects/user?userId=${userId}&taskId=${taskId}`,
        method: 'post',
    })
}
```

相关 api 层代码：userInfo.js

```
// 更新用户信息
export function updateUserInfo(data) {
    return request({
        url: 'api/v1/users/updateInfo',
        method: 'post',
        data
    })
}

// 获取用户活跃项目
export function getActPro(userId) {
    return request({
        url: `api/v1/users/projects?userId=${userId}`,
        method: 'get',

    })
}

// 获取用户活跃任务
```

```
export function getActTask(userId) {
    return request({
        url: `api/v1/tasks/userId?userId=${userId}`,
        method: 'get',

    })
}

// 替换头像
export function updateAvatar(data) {
    return request({
        url: 'api/v1/users/avatar',
        method: 'post',
        data
    })
}
```

五、 个人总结

在本次项目中，我负责登陆注册、首页菜单栏、个人信息、项目总体浏览、员工个人任务管理等功能模块的前端部分。在项目开发过程中，我们进行了需求分析、用例分析，并进行了统一的框架技术选择、数据库模型设计、数据结构设计、统一的页面设计等等。同时，我们也使用了统一的接口管理工具 Apifox 来加快前后端协作进度。

在组件代码编写的过程中，我参考了 Vue、Element UI、less 以及其它 js 插件、Vue-Cli 的官方文档，并从网络平台学习了一些框架和组件知识。这令我感受到了完善的框架和良好的生态对于一个项目构成的帮助与便利。

在此次项目开发过程中，我深切的意识到了合理的协作方式对于项目开发的重要性。接口管理工具 Apifox、代码托管库 github、共享文档等，无一不对我们项目组的开发起到了极大的帮助，也更让我意识到了软件工程于现代软件开发的重要性。

在本次项目开发中，我也进一步体验了完整流程的项目开发，从需求分析、设计、开发到测试和部署的整个过程。通过与团队成员的密切协作，我更加深入地理解了软件开发中的各个环节，也对如何协调团队成员以及如何在时间和质量上平衡得到了更深刻的认识。

同时，本次项目也让我认识到了自己的不足之处，如项目内代码复用性、可扩展性等方面还需加强。因此，我会继续学习和探索软件工程技术，提升自己的能力和水平。在未来的项目中，我也将更加注重代码规范性和可维护性，以便于团队成员协作和项目后期的维护。

这次项目经历让我受益匪浅，不仅提高了自己的技术能力与项目协作经验，也让我更加深入地了解了软件工程的重要性和必要性，我相信这对我今后的学习和职业发展都将产生积极的影响。