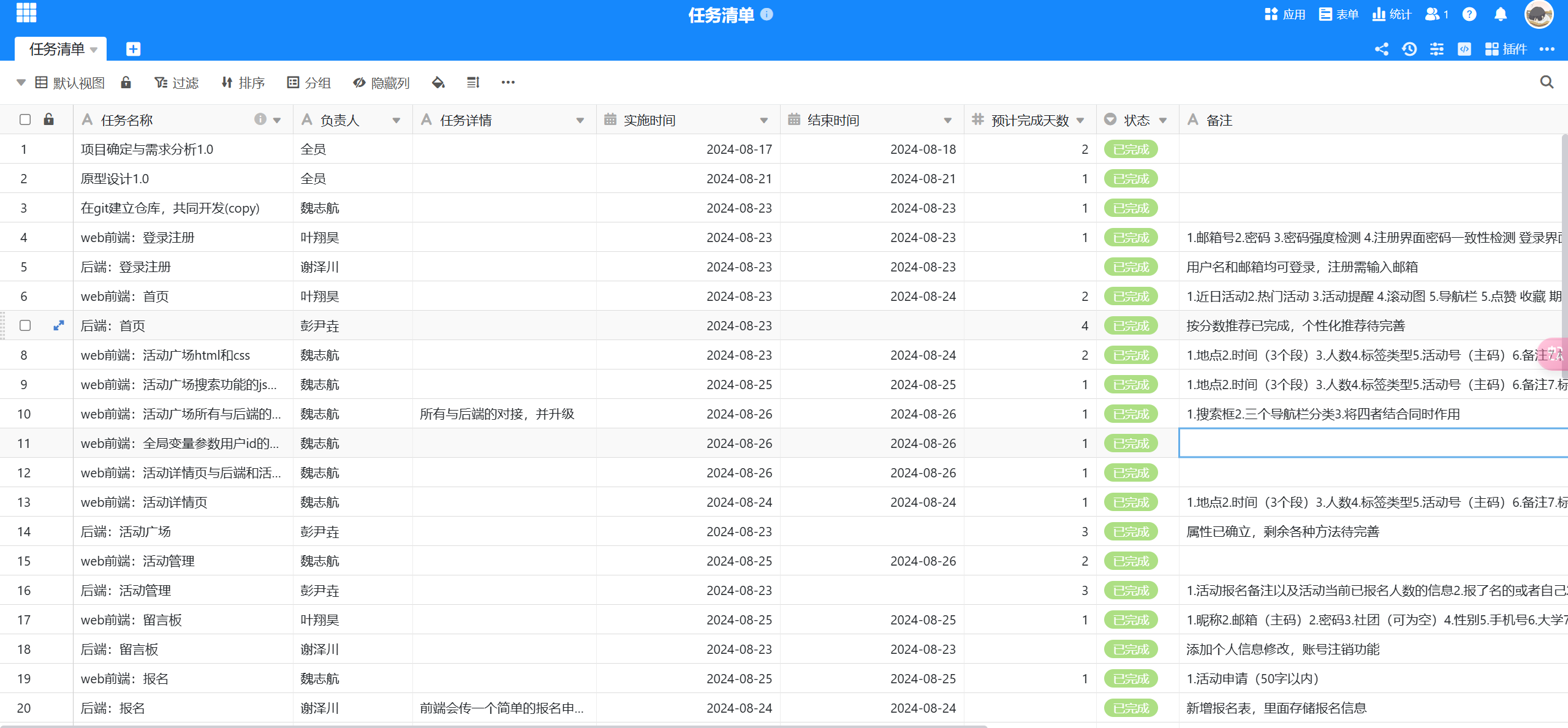
组长:魏志航

组员:兰天阳，彭尹垚，张世纪，叶翔昊，谢泽川

总体情况：

整体上进度和计划接近，完成近一半的页面与后端的对接。明天完成剩下的对接，后天进行整体的优化与

**魏志航**

****

**1.完成活动管理到活动发布的跳转**

<!-- 发布活动按钮 -->

            <button class="c-button" id="releaseButton">

                <span class="c-main">

                  <span class="c-ico"><span class="c-blur"></span> <span class="ico-text">+</span></span>

                  发布

                </span>

              </button>

//活动发布按钮

                document.getElementById('releaseButton').onclick = function() {

                    window.location.href = '../../ActivityRelease/ActivityRelease.html';

                };

/\* 发布活动 \*/

.c-button {

    position: fixed;

    bottom: 80px; /\* 与返回顶部按钮留出空间 \*/

    right: 100px;

    border: none;

    background: linear-gradient(140deg, rgba(249, 231, 30, 1) 0%, rgba(224, 204, 24, 1) 100%); /\* 使用主题色 \*/

    border-radius: 45px;

    font-size: 20px;

    text-align: left;

    padding: 8px 0 10px 0; /\* 上下内边距保持一致 \*/

    border: 2px solid rgb(32, 32, 31) !important; /\* 边框颜色 \*/

    border-style: outset;

    box-shadow: -5px -5px 18px rgba(88, 80, 80, 0.1);

    cursor: pointer;

    transition: background 0.5s ease-in-out,colorcolor 0.5s ease-in-out;; /\* 添加渐变过渡 \*/

}

.c-button .c-main {

    border-radius: 45px;

    color: rgba(0, 0, 0, 1); /\* 黑色文字 \*/

    padding: 11px 25px 11px 2px;

    box-shadow: inset 0px 0px 5px rgba(17, 17, 17, 0);

    transition: all 0.5s ease-in-out;

  border: 1px solid transparent;

}

.c-ico .c-blur {

    background: linear-gradient(318deg, rgba(249, 231, 30, 1) 0%, rgba(224, 204, 24, 1) 100%);

    border-radius: 100%;

    margin-left: 0;

    padding: 8px 23px;

    filter: blur(1px);

    text-align: center;

}

.c-ico {

    position: relative;

    margin-right: 20px;

}

.c-ico .ico-text {

    position: absolute;

    left: 50%;

    top: 50%;

    transform: translate(-50%, -50%);

}

.ico-text

{

    font-size: 35px;

}

/\* Hovering \*/

.c-button .c-main:hover {

    box-shadow: inset 0px 0px 5px rgba(17, 17, 17, 0.6);

    border: 1px solid rgba(26, 26, 26, 0.5);

    color: rgba(0, 0, 0, 0.7); /\* 悬浮时改变文字颜色 \*/

    background: linear-gradient(140deg, rgba(224, 204, 24, 1) 0%, rgba(249, 231, 30, 1) 100%); /\* 悬浮时的背景色变更 \*/

}

**2.完成活动广场与后端的所有对接，主要是搜索功能和三个导航栏分类功能，并进行升级。**

**先完成单个搜索和单个导航栏分类的功能，而后升级实现搜索与导航栏分类同时作用，对整个活动广场的html代码做了不少升级和更改。**

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>活动 | 校园活动平台</title>

        <link rel="icon" type="image/png" href="../../images/logo.png" sizes="32x32">

        <link rel="stylesheet" href="activity.css">

    </head>

    <body>

        <div>

            <nav>

                <div class="logo">

                    <img src="../../images/logo.png" alt="Logo" /> <!-- 替换为您的 logo 图片路径 -->

                </div>

                    <ul>

                        <li><a href="../../homepage/homepage.html">首页</a></li>

                        <li><a href="../../activity/activitySquare/activity.html"class="active">活动广场</a></li>

                        <li><a href="../../activity/activityManagement/activityManagement.html">活动管理</a></li>

                        <li><a href="../../Message/message.html">留言板</a></li>

                        <li><a href="../../PHP/personal page.html">个人中心</a></li>

                    </ul>

            </nav>

        </div>

        <div class="mainbody-area">

            <div class="select-area">

                <!-- 搜索框 -->

                <div class="search-area"> <!-- 添加搜索框的容器 -->

                    <input type="text" placeholder="搜索活动..." class="search-input"> <!-- 搜索框输入框 -->

                    <button class="search-button">搜索</button> <!-- 搜索按钮 -->

                </div>

                <!-- 分类列表 -->

                <div class="breadcrumb-area">

                    <link href="//unpkg.com/layui@2.9.16/dist/css/layui.css" rel="stylesheet">

                    <h3>分类</h3>

                    <span class="layui-breadcrumb" lay-separator="|" id="category-link">

                        <a href="" data-category="全部"><cite>全部</cite></a>

                        <a href="" data-category="体育">体育</a>

                        <a href="" data-category="学术">学术</a>

                        <a href="" data-category="艺术">艺术</a>

                        <a href="" data-category="桌游">桌游</a>

                        <a href="" data-category="出行">出行</a>

                        <a href="" data-category="手游">手游</a>

                        <a href="" data-category="端游">端游</a>

                        <a href="" data-category="其他">其他</a>

                    </span>

                    <h3>状态</h3>

                    <span class="layui-breadcrumb" lay-separator="|" id="status-link">

                        <a href="" data-status="全部"><cite>全部</cite></a>

                        <a href="" data-status="审核中">审核中</a>

                        <a href="" data-status="报名中">报名中</a>

                        <a href="" data-status="进行中">进行中</a>

                        <a href="" data-status="已结束">已结束</a>

                    </span>

                    <h3>人数</h3>

                    <span class="layui-breadcrumb" lay-separator="|" id="people-link">

                        <a href="" data-people="全部"><cite>全部</cite></a>

                        <a href="" data-people="2">双人</a>

                        <a href="" data-people="3">3~5人</a>

                        <a href="" data-people="6">6~10人</a>

                        <a href="" data-people="11">大型多人</a>

                    </span>

                    <style>

                        .highlight {

                            font-weight: bold; /\* 让文本加粗 \*/

                            color: #007bff;    /\* 选择您想要的高亮颜色 \*/

                        }

                        .breadcrumb-area a {

                            text-decoration: none; /\* 去掉链接下划线 \*/

                            padding: 0 5px; /\* 为链接添加内边距 \*/

                        }

                        .breadcrumb-area a:hover {

                            text-decoration: underline; /\* 悬停时下划线 \*/

                        }

                    </style>

                    <script src="//unpkg.com/layui@2.9.16/dist/layui.js"></script>

                    <script>

                        document.addEventListener("DOMContentLoaded", function () {

                            // 默认高亮“全部”链接

                            const attributes = ['category', 'status', 'people'];

                            attributes.forEach(attr => {

                                const highlightLink = document.querySelector(`.breadcrumb-area a[data-${attr}="全部"]`);

                                if (highlightLink) {

                                    highlightLink.classList.add('highlight');

                                }

                            });

                            // 添加点击事件

                            const categoryLinks = document.querySelectorAll('#category-link a');

                            const statusLinks = document.querySelectorAll('#status-link a');

                            const peopleLinks = document.querySelectorAll('#people-link a');

                            // 处理类别链接

                            categoryLinks.forEach(link => {

                                link.addEventListener('click', function (event) {

                                    event.preventDefault(); // 防止链接的默认行为

                                    // 清除其他链接的高亮样式

                                    categoryLinks.forEach(l => l.classList.remove('highlight'));

                                    // 添加高亮样式到当前链接

                                    this.classList.add('highlight');

                                });

                            });

                            // 处理状态链接

                            statusLinks.forEach(link => {

                                link.addEventListener('click', function (event) {

                                    event.preventDefault(); // 防止链接的默认行为

                                    // 清除其他链接的高亮样式

                                    statusLinks.forEach(l => l.classList.remove('highlight'));

                                    // 添加高亮样式到当前链接

                                    this.classList.add('highlight');

                                });

                            });

                            // 处理人数链接

                            peopleLinks.forEach(link => {

                                link.addEventListener('click', function (event) {

                                    event.preventDefault(); // 防止链接的默认行为

                                    // 清除其他链接的高亮样式

                                    peopleLinks.forEach(l => l.classList.remove('highlight'));

                                    // 添加高亮样式到当前链接

                                    this.classList.add('highlight');

                                });

                            });

                        });

                    </script>

                </div>

            </div>

            <!-- 活动列表 -->

            <div class="activity-list">

                <h2>无搜索结果</h2>

                <div class="activity-item" data-url="../activityDetails/activityDetails.html">

                    <div class="activity-img">

                        <img src="../activityImage/狼人杀.jpg" alt="狼人杀" />

                    </div>

                    <div class="activity-details" >

                        <h2>狼人杀</h2>

                        <p>发布人: 不吃香菜</p>

                        <p>时间: 2023年10月1日</p>

                        <p>人数: 8人</p>

                        <p>地点: 活动室</p>

                        <p>类型: 桌游</p>

                    </div>

                </div>

            </div>

            <!-- 返回顶部按钮 -->

            <button class="back-to-top" id="backToTop">返回顶部</button>

        </div>

        <script>

            // 回到顶部按钮

            const backToTopButton = document.getElementById('backToTop');

            window.onscroll = function() {

                if (document.body.scrollTop > 100 || document.documentElement.scrollTop > 100) {

                    backToTopButton.style.display = 'block'; // 显示按钮

                } else {

                    backToTopButton.style.display = 'none'; // 隐藏按钮

                }

            };

            backToTopButton.addEventListener('click', function() {

                window.scrollTo({

                    top: 0,

                    behavior: 'smooth' // 平滑滚动

                });

            });

            // 函数：格式化日期以去掉秒数

            function formatDate(dateString) {

                const date = new Date(dateString);

                if (isNaN(date)) {

                    return '未设置'; // 如果日期无效，返回“未设置”

                }

                // 格式化为 YYYY-MM-DD HH:MM

                const year = date.getFullYear();

                const month = (date.getMonth() + 1).toString().padStart(2, '0'); // 月份从 0 开始，所以需要加 1

                const day = date.getDate().toString().padStart(2, '0');

                const hours = date.getHours().toString().padStart(2, '0');

                const minutes = date.getMinutes().toString().padStart(2, '0');

                return `${year}-${month}-${day} ${hours}:${minutes}`; // 返回格式化的字符串

            }

            //活动搜索

            document.addEventListener("DOMContentLoaded", function () {

                const activityList = document.querySelector('.activity-list');

                const searchInput = document.querySelector('.search-input'); // 选择搜索框

                const searchButton = document.querySelector('.search-button'); // 选择搜索按钮

                // 初始化存储当前选择的分类、状态和人数

                let currentCategory = '全部';

                let currentStatus = '全部';

                let currentPeople = '全部';

                // 加载活动数据的函数

                function loadActivities(keyword = '', category = '', status = '', people = '') {

                    activityList.innerHTML = ''; // 清空活动列表内容

                    // 根据是否有分类，选择合适的URL

                    let url = 'http://192.168.1.43:8080/activities/searchbyfour';

                    const queryParams = [];

                    // 添加搜索关键词

                    if (keyword) {

                        queryParams.push(`keyword=${encodeURIComponent(keyword)}`); // 添加搜索关键词参数

                    }

                    // 添加分类参数，确保不传递"全部"

                    if (category && category !== '全部') {

                        queryParams.push(`type=${encodeURIComponent(category)}`); // 添加分类参数

                    }

                    // 添加状态参数，确保不传递"全部"

                    if (status && status !== '全部') {

                        queryParams.push(`status=${encodeURIComponent(status)}`); // 添加状态参数

                    }

                    // 添加人数参数，确保不传递"全部"

                    if (people && people !== '全部') {

                        if (people === '2') {

                            queryParams.push(`minCeiling=2`);

                            queryParams.push(`maxCeiling=2`);

                        } else if (people === '3') {

                            queryParams.push(`minCeiling=3`);

                            queryParams.push(`maxCeiling=5`);

                        } else if (people === '6') {

                            queryParams.push(`minCeiling=6`);

                            queryParams.push(`maxCeiling=10`);

                        } else if (people === '11') {

                            queryParams.push(`minCeiling=11`);

                            queryParams.push(`maxCeiling=9999`);

                        }

                    }

                    // 检查查询参数，决定是否修改 URL

                    if (queryParams.length > 0) {

                        url += '?' + queryParams.join('&'); // 构建完整的请求 URL

                    }

                    // 发起请求活动的 API

                    fetch(url)

                        .then(response => {

                            if (!response.ok) {

                                throw new Error('网络响应错误');

                            }

                            return response.json();

                        })

                        .then(result => {

                            // 检查返回的状态码

                            if (result.code === 200) {

                                const activities = result.data; // 获取活动列表

                                // 检查是否有活动数据

                                if (Array.isArray(activities) && activities.length > 0) {

                                    activities.forEach(activity => {

                                        const activityItem = document.createElement('div');

                                        activityItem.className = 'activity-item';

                                        activityItem.setAttribute('data-url', `../activityDetails/activityDetails.html?activityid=${activity.activityid}`); // 每个项的点击链接

                                        // 使用响应中的数据填充活动项

                                        activityItem.innerHTML = `

                                            <div class="activity-img">

                                                <img src="${activity.profileimageurl || 'default\_image.png'}" alt="${activity.activityname}" />

                                            </div>

                                            <div class="activity-details">

                                                <h2>${activity.activityname}</h2>

                                                <p>发布人: ${activity.headusername}</p>

                                                <p>时间: ${formatDate(activity.timesubmit || '未设置')}</p>

                                                <p>人数: ${activity.maxceiling}人</p>

                                                <p>地点: ${activity.location || '未设置'}</p>

                                                <p>类型: ${activity.type}</p>

                                            </div>

                                        `;

                                        activityList.appendChild(activityItem);

                                    });

                                    // 处理每个活动项的点击事件

                                    document.querySelectorAll('.activity-item').forEach(function (item) {

                                        item.addEventListener('click', function () {

                                            const url = this.getAttribute('data-url');

                                            console.log("点击了activity项，目标URL是:", url);

                                            // 点击后跳转到活动详情页（基于实际情况更新）

                                            if (url) {

                                                window.location.href = url;

                                            }

                                        });

                                    });

                                } else {

                                    // 如果没有活动数据

                                    activityList.innerHTML = '<p>当前没有活动报名。</p>';

                                }

                            } else {

                                // 处理请求失败的情况

                                console.error('获取活动数据失败:', result.message);

                                activityList.innerHTML = `<h2>暂无结果。</h2>`; // 显示错误消息

                            }

                        })

                        .catch(error => {

                            console.error('获取活动数据时发生错误:', error);

                            activityList.innerHTML = '<p>网络错误，请稍后再试。</p>'; // 显示网络错误

                        });

                }

                // 初始化加载所有活动

                loadActivities(); // 加载时显示所有活动

                // 绑定搜索按钮事件

                searchButton.addEventListener('click', () => {

                    loadActivities(searchInput.value.trim(), currentCategory, currentStatus, currentPeople); // 传递搜索关键词和当前分类、状态、人数

                });

                // 监听搜索框的键盘事件

                searchInput.addEventListener('keypress', (event) => {

                    if (event.key === 'Enter') {

                        loadActivities(searchInput.value.trim(), currentCategory, currentStatus, currentPeople); // 传递搜索关键词和当前分类、状态、人数

                    }

                });

                // 监听分类链接点击事件

                const categoryLinks = document.querySelectorAll('#category-link a');

                categoryLinks.forEach(link => {

                    link.addEventListener('click', function (event) {

                        event.preventDefault(); // 防止链接的默认行为

                        // 设置当前分类

                        currentCategory = this.getAttribute('data-category'); // 获取被点击的分类

                        // 高亮当前分类

                        categoryLinks.forEach(l => l.classList.remove('highlight'));

                        this.classList.add('highlight');

                        // 发起活动加载请求

                        loadActivities(searchInput.value.trim(), currentCategory, currentStatus, currentPeople);

                    });

                });

                // 监听状态链接点击事件

                const statusLinks = document.querySelectorAll('#status-link a');

                statusLinks.forEach(link => {

                    link.addEventListener('click', function (event) {

                        event.preventDefault(); // 防止链接的默认行为

                        // 设置当前状态

                        currentStatus = this.getAttribute('data-status'); // 获取被点击的状态

                        // 高亮当前状态

                        statusLinks.forEach(l => l.classList.remove('highlight'));

                        this.classList.add('highlight');

                        // 发起活动加载请求

                        loadActivities(searchInput.value.trim(), currentCategory, currentStatus, currentPeople);

                    });

                });

                // 监听人数链接点击事件

                const peopleLinks = document.querySelectorAll('#people-link a');

                peopleLinks.forEach(link => {

                    link.addEventListener('click', function (event) {

                        event.preventDefault(); // 防止链接的默认行为

                        // 设置当前人数

                        currentPeople = this.getAttribute('data-people'); // 获取被点击的人数

                        // 高亮当前人数

                        peopleLinks.forEach(l => l.classList.remove('highlight'));

                        this.classList.add('highlight');

                        // 发起活动加载请求

                        loadActivities(searchInput.value.trim(), currentCategory, currentStatus, currentPeople);

                    });

                });

            });

        </script>

    </body>

</html>

**3.活动详情页与后端和活动广场的对接，完成从活动广场接收到活动id，并从后端得到活动的详细信息**

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>活动详情 | 校园活动平台</title>

    <link rel="icon" type="image/png" href="../../images/logo.png" sizes="32x32">

    <link rel="stylesheet" href="activityDetails.css">

</head>

<body>

    <div>

        <nav>

            <div class="logo">

                <img src="../../images/logo.png" alt="Logo" />

            </div>

            <ul>

                <li><a href="../../homepage/homepage.html">首页</a></li>

                <li><a href="../../activity/activitySquare/activity.html" class="active">活动广场</a></li>

                <li><a href="../../activity/activityManagement/activityManagement.html">活动管理</a></li>

                <li><a href="../../Message/message.html">留言板</a></li>

                <li><a href="../../PHP/personal page.html">个人中心</a></li>

            </ul>

        </nav>

    </div>

    <div class="mainbody-area">

        <div class="activityDetails">

            <img id="coverImage" src="" alt="活动封面" class="cover-image">

            <div class="details-content">

                <h2 id="activityName"></h2>

                <p><strong>发布人:</strong> <span id="publisher"></span></p>

                <p><strong>时间:</strong> <span id="eventTime"></span></p>

                <p><strong>人数:</strong> <span id="maxParticipants"></span></p>

                <p><strong>地点:</strong> <span id="location"></span></p>

                <p><strong>类型:</strong> <span id="activityType"></span></p>

                <p><strong>报名要求:</strong> <span id="registrationRequirements"></span></p>

                <p><strong>联系方式:</strong> <span id="contact"></span></p>

                <div class="register-section">

                    <input type="text" placeholder="请填写简单的备注, 建议30字以内" />

                    <button class="register-button"><strong>报名</strong></button>

                </div>

                <div class="rate-section">

                    <p><strong>发布人评分:</strong> <span id="rating"></span></p>

                </div>

            </div>

        </div>

        <div class="more-detail">

            <h1 style="text-align: center;">其他说明</h1>

            <p id="additionalInfo"></p>

        </div>

        <button class="back-to-top" id="backToTop">返回顶部</button>

        <script>

            const backToTopButton = document.getElementById('backToTop');

            window.onscroll = function() {

                if (document.body.scrollTop > 100 || document.documentElement.scrollTop > 100) {

                    backToTopButton.style.display = 'block'; // 显示按钮

                } else {

                    backToTopButton.style.display = 'none'; // 隐藏按钮

                }

            };

            backToTopButton.addEventListener('click', function() {

                window.scrollTo({

                    top: 0,

                    behavior: 'smooth' // 平滑滚动

                });

            });

            // 获取活动ID并加载活动详细信息

            // 获取活动ID并加载活动详细信息

            function loadActivityDetails() {

                const urlParams = new URLSearchParams(window.location.search);

                const activityId = urlParams.get('activityid');

                if (activityId)

                {

                    // 首先请求活动详细信息

                    fetch(`http://192.168.1.43:8080/activities/showactivity/${activityId}`)

                        .then(response => {

                            if (!response.ok) {

                                throw new Error('网络响应错误');

                            }

                            return response.json();

                        })

                        .then(data => {

                            if (data.code === 200) {

                                const activity = data.data; // 获取活动的具体信息

                                // 更新界面内容

                                document.getElementById('coverImage').src = activity.profileimageurl || 'default\_image.png'; // 使用活动封面图片

                                document.getElementById('activityName').innerText = activity.activityname || '未命名活动';

                                document.getElementById('eventTime').innerText = formatDate(activity.timesubmit) || '未设置'; // 格式化时间

                                document.getElementById('maxParticipants').innerText = activity.maxceiling ? `${activity.maxceiling}人` : '未设置';

                                document.getElementById('location').innerText = activity.location || '未设置';

                                document.getElementById('activityType').innerText = activity.type || '未指定类型';

                                document.getElementById('registrationRequirements').innerText = activity.registrationRequirements || '未指定要求';

                                document.getElementById('contact').innerText = activity.contact || '未提供';

                                document.getElementById('additionalInfo').innerText = activity.additionalInfo || '无其他说明';

                                // 获取发布人的声誉信息

                                return fetch(`http://192.168.1.43:8080/auth/showuser/${activity.headid}`);

                            } else {

                                console.error('获取活动详细信息失败:', data.message);

                                alert('加载活动详情失败，请重试。');

                            }

                        })

                        .then(response => {

                            if (response) {

                                if (!response.ok) {

                                    throw new Error('网络响应错误');

                                }

                                return response.json();

                            }

                        })

                        .then(userData => {

                            if (userData && userData.code === 200) {

                                const reputation = userData.data.reputation || '暂无评分';

                                const username = userData.data.username || '未知发布人';

                                document.getElementById('rating').innerText = reputation; // 更新声誉信息

                                document.getElementById('publisher').innerText = username || '未知发布人';

                            } else {

                                console.error('获取用户声誉失败:', userData.message);

                                document.getElementById('rating').innerText = '暂无评分'; // 默认值或错误信息

                            }

                        })

                        .catch(error => {

                            console.error('获取活动详情或用户声誉时发生错误:', error);

                            alert('网络错误，请稍后再试。');

                        });

                } else {

                    alert('活动ID未找到。');

                }

            }

            // 时间格式化函数

            function formatDate(dateString) {

                const date = new Date(dateString);

                if (isNaN(date)) {

                    return '未设置'; // 如果日期无效，返回“未设置”

                }

                // 格式化为 YYYY-MM-DD HH:MM

                const year = date.getFullYear();

                const month = (date.getMonth() + 1).toString().padStart(2, '0'); // 月份从 0 开始，所以需要加 1

                const day = date.getDate().toString().padStart(2, '0');

                const hours = date.getHours().toString().padStart(2, '0');

                const minutes = date.getMinutes().toString().padStart(2, '0');

                return `${year}-${month}-${day} ${hours}:${minutes}`; // 返回格式化的字符串

            }

            // 页面加载时调用函数

            loadActivityDetails();

        </script>

    </div>

</body>

</html>

**4.登陆成功后全局变量参数用户id的设置和部分网页的接收**

const headid= result.data.id; // 从服务器获取的用户 ID

                    // 保存用户 ID 到 localStorage

                    localStorage.setItem('headid', headid);

                    console.log(headid);

// 读取本地存储的用户 ID

                document.addEventListener('DOMContentLoaded', (event) => {

                    const savedUserId = localStorage.getItem('headid'); // 在这里读取

                    if (savedUserId) {

                        console.log(`欢迎回来，用户ID: ${savedUserId}`);

                    } else {

                        console.log('用户未登录或 session 已过期');

                    }

                });

**5.继续统筹规划项目计划，根据进度和实现难度不断动态更改**

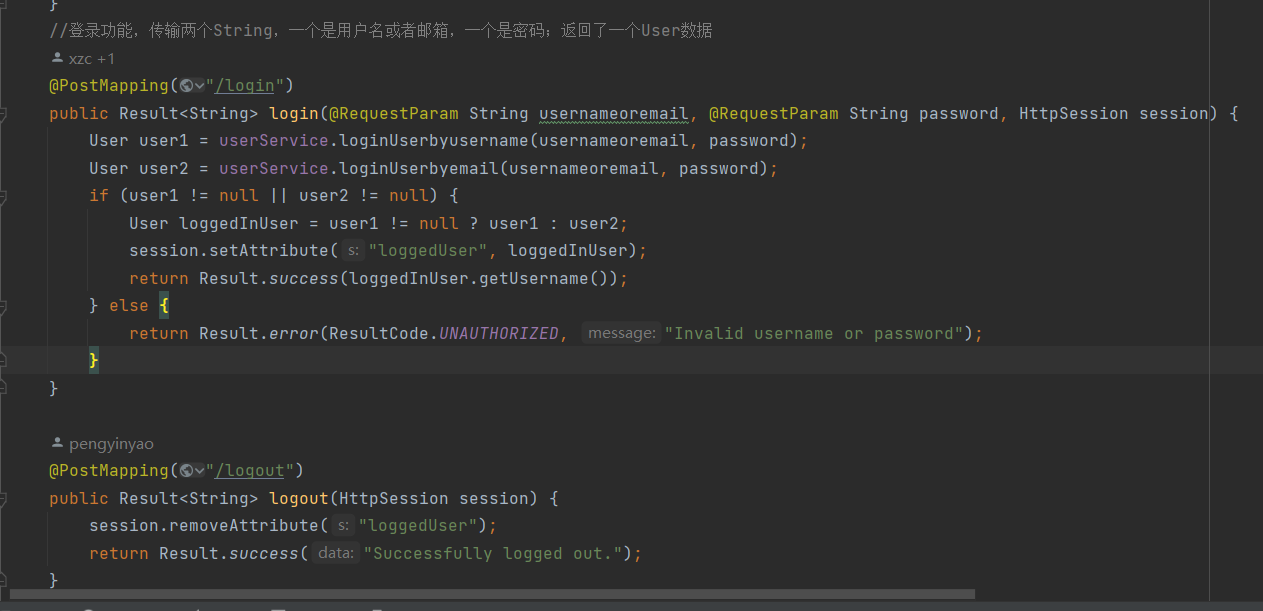
**彭尹垚**

Java日志

今天将活动的广场的搜索功能给完善，并且在登陆和登出请求时加入httsession，暂存用户信息，并且做了登陆检测功能，能识别在未登录状态下时跳转到登陆界面，对数据库进行保护。并利用sesion能获取到当期登录用户的信息。与前端沟通对接和cookie问题







**谢泽川**

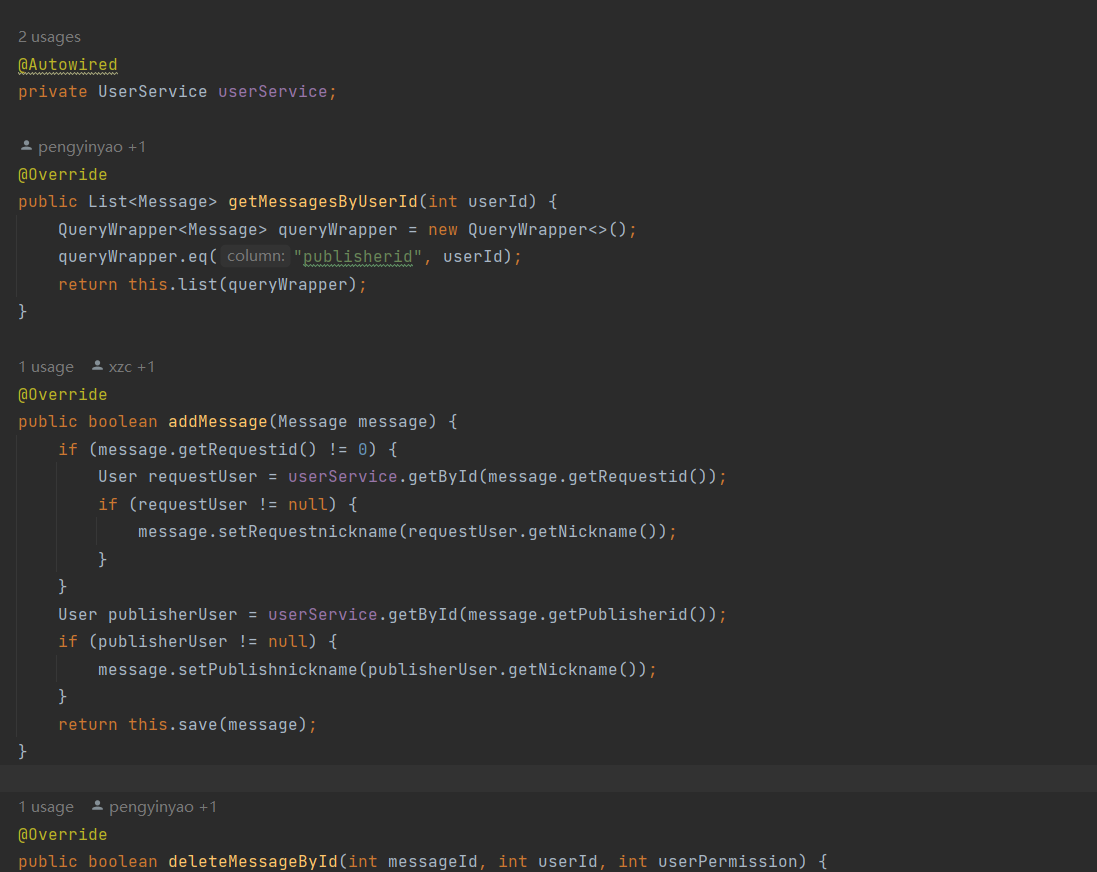
##### Java学习日志 - JAVASpringboot对校园活动的编写实现

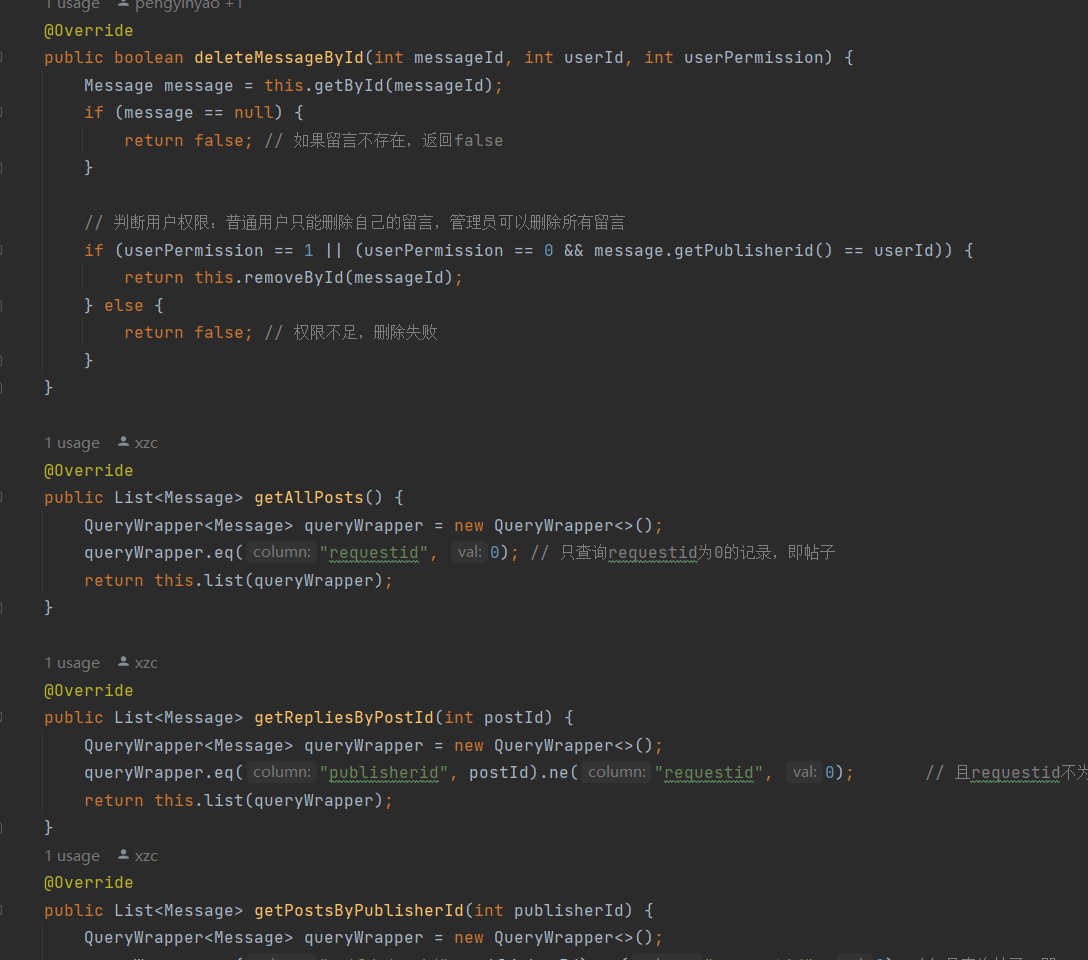
##### 日期：2024年8月26日

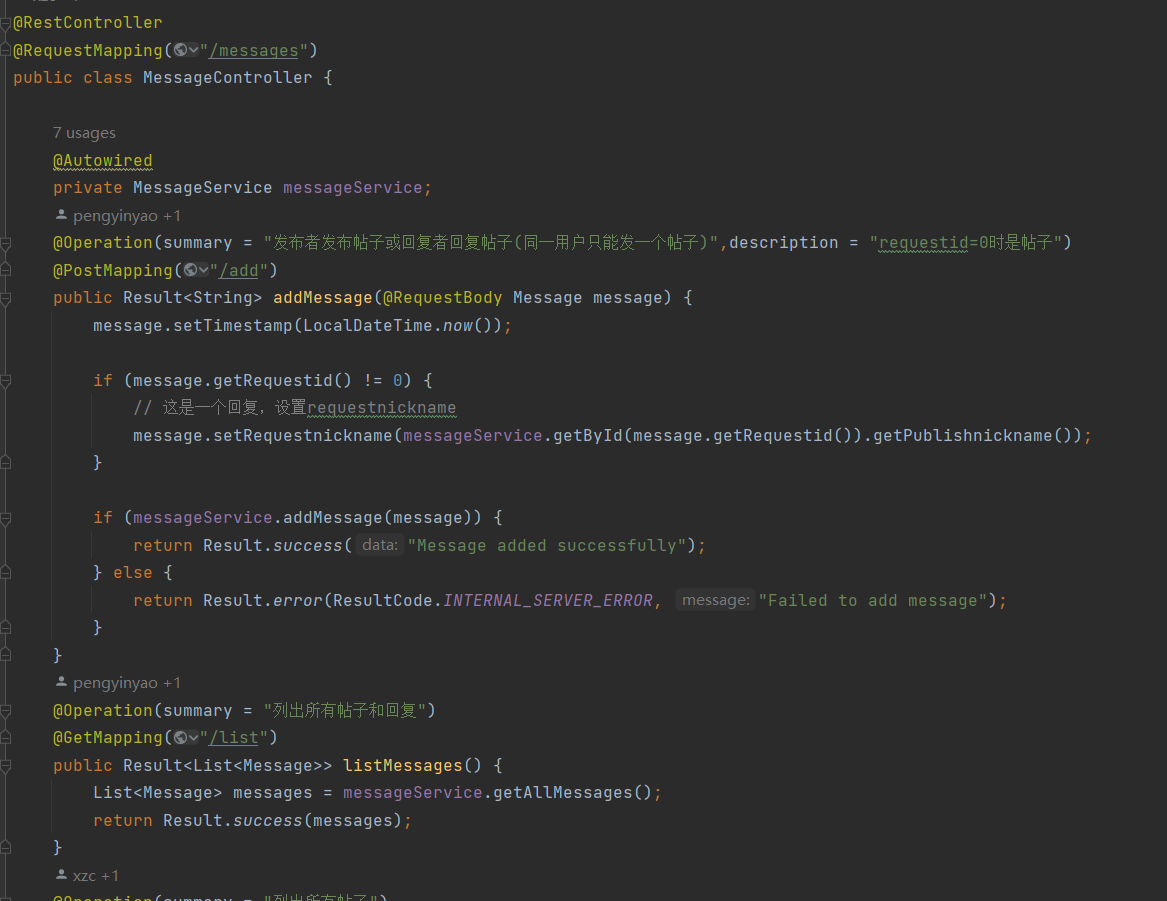
##### 实现内容：

留言板的回复与发布功能：









该用户参与的活动（未结束的）的所有信息：





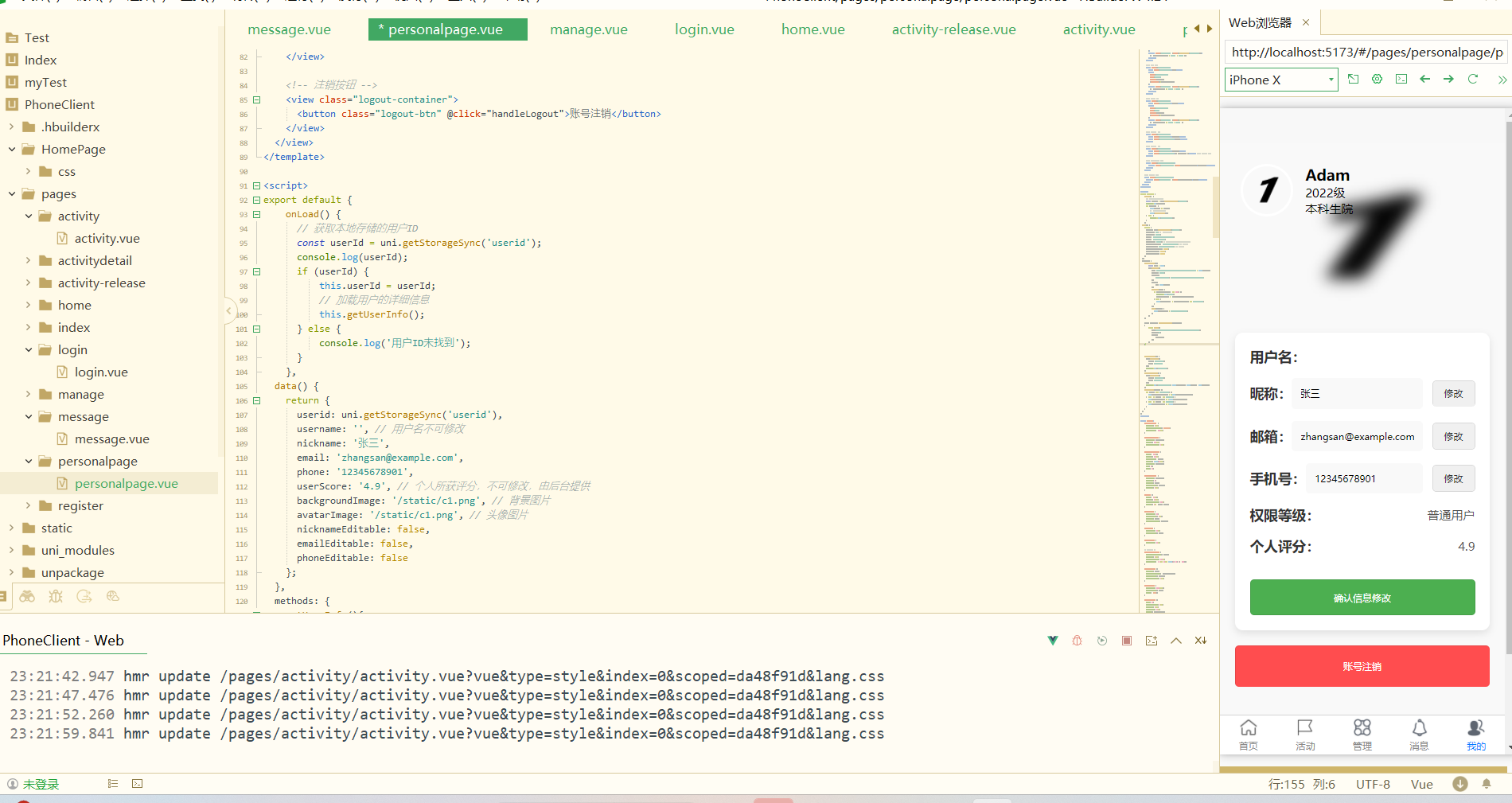
该用户参与的活动（已结束）的所有信息：





**张世纪**

进行了前后端连接测试



完善了前端页面

**叶翔昊**

8·26实训日志

总体情况：作为前端完成了登录注册和后端的对接，但在个人中心和登录状态的对接上出现了一点问题，正在解决。

代码：

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title></title>

<style type="text/css">

\*{

margin: 0;

padding: 0;

}

body{

height: 100vh;

display: flex;

justify-content: center;

align-items: center;

}

.container{

position: absolute;

background-color: #90cbe3;

width:300px;

height: 500px;

border: 1px soid beige;

border-radius: 5%;

box-shadow: 10px 10px 15px 5px rgba(70, 70, 70, 0.6);

}

.content{

display: flex;

flex-direction: column;

padding: 100px 0 0px 0px;

}

.content.input{

margin: 0 auto 50px auto;

width: 300px;

height: 50px;

border: 1 0px solid rgba(187, 110, 225, 5);

}

.content.input:nth-child(2){

margin: 80px auto 50px auto;

width: 300px;

height: 50px;

border: 1px solid rgba(187, 110, 225, 5);

}

.content button{

margin: 40px auto 10px auto;

width: 300px;

height: 35px;

}

.content.span{

margin:40px auto 40px 110px;

width:300px;

height: 35px;

}

</style>

</head>

<body>

<div class="container">

<div class="content">

<input type="text" placeholder="用户名" />

<input type="password" placeholder="密码" />

<button type="button">登录</button>

<span>没有账号？<a href="#">去注册</a></span>

</div>

</div>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>用户登录</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

<style>

html, body {

height: 100%;

margin: 0;

padding: 0;

font-family: Arial, sans-serif;

overflow: hidden;

}

body {

background: url('123/original.jpg') no-repeat center center fixed;

background-size: cover;

position: relative;

min-height: 100vh;

}

body::before {

content: "";

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

z-index: -1;

}

.header {

text-align: center;

color: white;

margin-top: 50px;

font-size: 2em;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

}

.container {

max-width: 400px;

margin: 100px auto;

padding: 20px;

background-color: rgba(255, 255, 255, 0.9);

border-radius: 15px; /\* 圆滑边缘 \*/

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); /\* 阴影效果 \*/

position: relative;

width: 100%;

height: auto;

}

h2 {

text-align: center;

}

label {

display: block;

margin-top: 10px;

}

.password-container {

position: relative;

}

input[type="password"], input[type="text"] {

width: 100%;

padding: 8px;

margin-top: 5px;

box-sizing: border-box;

border-radius: 5px; /\* 输入框圆滑边缘 \*/

border: 1px solid #ccc; /\* 输入框边框 \*/

}

.toggle-password {

position: absolute;

right: 10px;

top: 50%;

transform: translateY(-50%);

cursor: pointer;

}

button {

width: 100%;

padding: 15px;

background-color: #28a745;

color: white;

border: none;

border-radius: 5px;

margin-top: 20px;

cursor: pointer;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

font-size: 16px;

font-weight: bold;

text-transform: uppercase;

}

button:hover {

background-color: #218838;

}

.error {

color: red;

margin-top: 10px;

text-align: center;

}

.switch-page {

text-align: center;

margin-top: 5px;

}

.switch-page a {

color: #007bff;

text-decoration: none;

}

.switch-page a:hover {

text-decoration: underline;

}

.quote { /\* 样式定义 \*/

text-align: center;

margin-top: 20px;

font-style: italic;

color: white;

text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.5);

}

</style>

</head>

<body>

<div class="header">学生活动发布系统</div>

<div class="container">

<h2>登录</h2>

<form id="loginForm">

<label for="loginUsername">账号：</label>

<input type="text" id="loginUsername" placeholder="用户名/邮箱" required>

<label for="loginPassword">密码：</label>

<div class="password-container">

<input type="password" id="loginPassword" placeholder="请输入密码" required>

<i class="fas fa-eye-slash toggle-password" onclick="togglePassword('loginPassword')"></i>

</div>

<button type="submit">登录</button>

<p class="error" id="loginError"></p>

</form>

<div class="switch-page">

<p>没有账户？<a href="register.html">点此注册</a></p>

</div>

</div>

<div class="quote" id="quote"></div> <!-- 显示名言的容器 -->

<script>

// 名言数组

const quotes = [

"「生活不只是眼前的苟且，还有诗和远方。」",

"「知之愈明，则行之愈笃;行之愈笃，则知之益明。」",

"「你必须参加生活中的每一场战斗，才能真正享受生活。」",

"「天空没有翅膀的痕迹，但鸟儿已经飞过。」",

"「行动是治愈恐惧的良药，而犹豫、拖延将不断滋养恐惧。」",

"「世界是一本大书，若不到处走走的话，看到的总是同一页的内容。」",

"「如果你不出去走走，你就会以为这就是世界。」",

];

// 随机选择名言并显示

function displayRandomQuote() {

const randomIndex = Math.floor(Math.random() \* quotes.length);

const quoteElement = document.getElementById('quote');

quoteElement.textContent = quotes[randomIndex];

}

// 页面加载时显示随机名言

window.onload = displayRandomQuote;

function togglePassword(inputId) {

const input = document.getElementById(inputId);

const icon = input.nextElementSibling;

if (input.type === "password") {

input.type = "text";

icon.classList.remove("fa-eye-slash");

icon.classList.add("fa-eye");

} else {

input.type = "password";

icon.classList.remove("fa-eye");

icon.classList.add("fa-eye-slash");

}

}

document.getElementById('loginForm').addEventListener('submit', async function (event) {

event.preventDefault();

const username = document.getElementById('loginUsername').value;

const password = document.getElementById('loginPassword').value;

try {

const response = await fetch('http://192.168.1.43:8080/auth/login', {

method: 'POST',

headers: {

'Content-Type': 'application/x-www-form-urlencoded'

},

body: new URLSearchParams({

'usernameoremail': username,

'password': password

})

});

const result = await response.json();

console.log('Response Status:', response.status);

console.log('Response Data:', result);

if (response.ok && result.code === 200) {

// 保存用户信息到 localStorage

localStorage.setItem('user', JSON.stringify(result.data));

////////

const headid= result.data.id; // 这个值应该是从服务器获取的用户 ID

// 保存用户 ID 到 localStorage

localStorage.setItem('headid', headid);

console.log(headid);

///////

// 登录成功后跳转到主页

window.location.href = '../homepage/homepage.html';

} else {

document.getElementById('loginError').textContent = '账号或密码错误';

}

} catch (error) {

console.error('Error:', error); // 在控制台中记录错误

document.getElementById('loginError').textContent = '网络错误，请稍后重试';

}

});

</script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>用户注册</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

<style>

html, body {

height: 100%;

margin: 0;

padding: 0;

font-family: Arial, sans-serif;

overflow: hidden;

}

body {

background: url('123/original.jpg') no-repeat center center fixed;

background-size: cover;

position: relative;

min-height: 100vh;

}

body::before {

content: "";

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

z-index: -1;

}

.header {

font-family: "微软雅黑", sans-serif;

text-align: center;

color: white;

margin-top: 50px;

font-size: 2em;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

}

.container {

max-width: 400px;

margin: 50px auto;

padding: 20px;

background-color: rgba(255, 255, 255, 0.9);

border-radius: 15px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

position: relative;

width: 100%;

height: auto;

}

h2 {

text-align: center;

}

label {

display: block;

margin-top: 10px;

}

.password-container {

position: relative;

}

input[type="password"], input[type="text"], input[type="email"] {

width: 100%;

padding: 8px;

margin-top: 5px;

box-sizing: border-box;

border-radius: 5px;

border: 1px solid #ccc;

}

.toggle-password {

position: absolute;

right: 10px;

top: 50%;

transform: translateY(-50%);

cursor: pointer;

}

.check-mark {

position: absolute;

right: 35px;

top: 50%;

transform: translateY(-50%);

color: green;

display: none;

}

.strength-bar-container {

display: none;

position: relative;

top: -20px; /\* 调整这个值来控制与输入框的距离 \*/

margin-bottom: -20px; /\* 确保与下一个元素有合适的间距 \*/

width: 50%;

margin-left: 55px;

}

.strength-bar {

display: flex;

height: 5px; /\* 调整整体高度 \*/

}

.strength-bar div {

flex: 1;

background-color: #ccc;

margin: 0 2px; /\* 为各个进度条添加间距 \*/

}

.strength-text {

font-size: 0.8em;

margin-top: 5px;

}

.error {

color: red;

margin-top: 10px;

text-align: center;

}

.switch-page {

text-align: center;

margin-top: 20px;

}

.switch-page a {

color: #007bff;

text-decoration: none;

}

.switch-page a:hover {

text-decoration: underline;

}

button {

width: 100%;

padding: 15px;

background-color: #28a745;

color: white;

border: none;

border-radius: 5px;

margin-top: 20px;

cursor: pointer;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

font-size: 16px;

font-weight: bold;

text-transform: uppercase;

}

button:hover {

background-color: #218838;

}

</style>

</head>

<body>

<div class="header">学生活动发布系统</div>

<div class="container">

<h2>注册</h2>

<form id="registerForm">

<label for="registerUsername">用户名：</label>

<input type="text" id="registerUsername" name="username" placeholder="请输入用户名" required>

<label for="registerEmail">邮箱：</label>

<input type="email" id="registerEmail" name="email" placeholder="请输入邮箱" required>

<label for="registerPassword">密码：</label>

<div class="strength-bar-container" id="strengthBarContainer">

<div class="strength-bar">

<div id="bar1"></div>

<div id="bar2"></div>

<div id="bar3"></div>

<div id="bar4"></div>

<div id="bar5"></div>

</div>

<div class="strength-text" id="strengthText">密码强度：非常弱</div>

</div>

<div class="password-container">

<input type="password" id="registerPassword" name="password" placeholder="请输入密码" required>

<i class="fas fa-eye-slash toggle-password" onclick="togglePassword('registerPassword')"></i>

<i class="fas fa-check check-mark" id="passwordMatchCheck"></i>

</div>

<label for="confirmPassword">确认密码：</label>

<div class="password-container">

<input type="password" id="confirmPassword" name="confirmPassword" placeholder="请再次输入密码" required>

<i class="fas fa-eye-slash toggle-password" onclick="togglePassword('confirmPassword')"></i>

<i class="fas fa-check check-mark" id="confirmMatchCheck"></i>

</div>

<button type="submit">注册</button>

<p class="error" id="registerError"></p>

</form>

<div class="switch-page">

<p>已有账户？<a href="login.html">点此登录</a></p>

</div>

</div>

<script>

function togglePassword(inputId) {

const input = document.getElementById(inputId);

const icon = input.nextElementSibling;

if (input.type === "password") {

input.type = "text";

icon.classList.remove("fa-eye-slash");

icon.classList.add("fa-eye");

} else {

input.type = "password";

icon.classList.remove("fa-eye");

icon.classList.add("fa-eye-slash");

}

}

function checkPasswordStrength(password) {

const strengthBarContainer = document.getElementById('strengthBarContainer');

const strengthBar = [

document.getElementById('bar1'),

document.getElementById('bar2'),

document.getElementById('bar3'),

document.getElementById('bar4'),

document.getElementById('bar5')

];

const strengthText = document.getElementById('strengthText');

if (password) {

strengthBarContainer.style.display = 'block';

} else {

strengthBarContainer.style.display = 'none';

return;

}

let score = 0;

// 密码长度

if (password.length <= 4) {

score += 5;

} else if (password.length <= 7) {

score += 10;

} else {

score += 25;

}

// 字母

const hasLower = /[a-z]/.test(password);

const hasUpper = /[A-Z]/.test(password);

if (hasLower || hasUpper) {

score += hasLower && hasUpper ? 20 : 10;

}

// 数字

const numMatches = (password.match(/\d/g) || []).length;

if (numMatches > 0) {

score += numMatches === 1 ? 10 : 20;

}

// 符号

const symbolMatches = (password.match(/[^a-zA-Z0-9]/g) || []).length;

if (symbolMatches > 0) {

score += symbolMatches === 1 ? 10 : 25;

}

// 奖励

if (hasLower || hasUpper) {

if (numMatches > 0) {

if (symbolMatches > 0) {

score += hasLower && hasUpper ? 5 : 3;

} else {

score += 2;

}

}

}

// 更新进度条颜色

strengthBar.forEach(bar => bar.style.backgroundColor = '#ccc');

if (score >= 90) {

strengthText.textContent = '密码强度：非常安全';

strengthBar.forEach(bar => bar.style.backgroundColor = 'purple');

} else if (score >= 80) {

strengthText.textContent = '密码强度：安全';

strengthBar.slice(0, 4).forEach(bar => bar.style.backgroundColor = 'purple');

} else if (score >= 70) {

strengthText.textContent = '密码强度：非常强';

strengthBar.slice(0, 3).forEach(bar => bar.style.backgroundColor = 'red');

} else if (score >= 60) {

strengthText.textContent = '密码强度：强';

strengthBar.slice(0, 3).forEach(bar => bar.style.backgroundColor = 'orange');

} else if (score >= 50) {

strengthText.textContent = '密码强度：一般';

strengthBar.slice(0, 2).forEach(bar => bar.style.backgroundColor = 'green');

} else if (score >= 25) {

strengthText.textContent = '密码强度：弱';

strengthBar.slice(0, 1).forEach(bar => bar.style.backgroundColor = 'green');

} else {

strengthText.textContent = '密码强度：非常弱';

}

}

function checkPasswordsMatch() {

const password = document.getElementById('registerPassword').value;

const confirmPassword = document.getElementById('confirmPassword').value;

const passwordMatchCheck = document.getElementById('passwordMatchCheck');

const confirmMatchCheck = document.getElementById('confirmMatchCheck');

if (password && confirmPassword && password === confirmPassword) {

passwordMatchCheck.style.display = 'inline';

confirmMatchCheck.style.display = 'inline';

} else {

passwordMatchCheck.style.display = 'none';

confirmMatchCheck.style.display = 'none';

}

}

document.getElementById('registerPassword').addEventListener('input', function () {

const password = this.value;

checkPasswordStrength(password);

checkPasswordsMatch();

});

document.getElementById('confirmPassword').addEventListener('input', checkPasswordsMatch);

document.getElementById('registerForm').addEventListener('submit', async function(event) {

event.preventDefault(); // 防止表单默认提交

const username = document.getElementById('registerUsername').value;

const email = document.getElementById('registerEmail').value;

const password = document.getElementById('registerPassword').value;

const confirmPassword = document.getElementById('confirmPassword').value;

if (password !== confirmPassword) {

document.getElementById('registerError').textContent = '密码和确认密码不一致';

return;

}

const userData = {

username: username,

password: password,

email: email

};

try {

const response = await fetch('http://192.168.1.43:8080/auth/register', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({

id: 0,

username: userData.username,

password: userData.password,

email: userData.email,

nickname: "",

community: "",

sex: "",

pnumber: 0,

university: "",

hobby: "",

permission: 0,

reputation: 0,

profileimageurl: ""

})

});

const result = await response.json();

if (response.ok&&result.code === 200) {

alert('注册成功！');

window.location.href = 'login.html'; // 跳转到登录页面

} else if (response.status === 400) {

document.getElementById('registerError').textContent = '用户名已存在';

} else {

document.getElementById('registerError').textContent = '用户名已存在';

}

} catch (error) {

console.error('Error:', error);

document.getElementById('registerError').textContent = '注册失败，请稍后再试。';

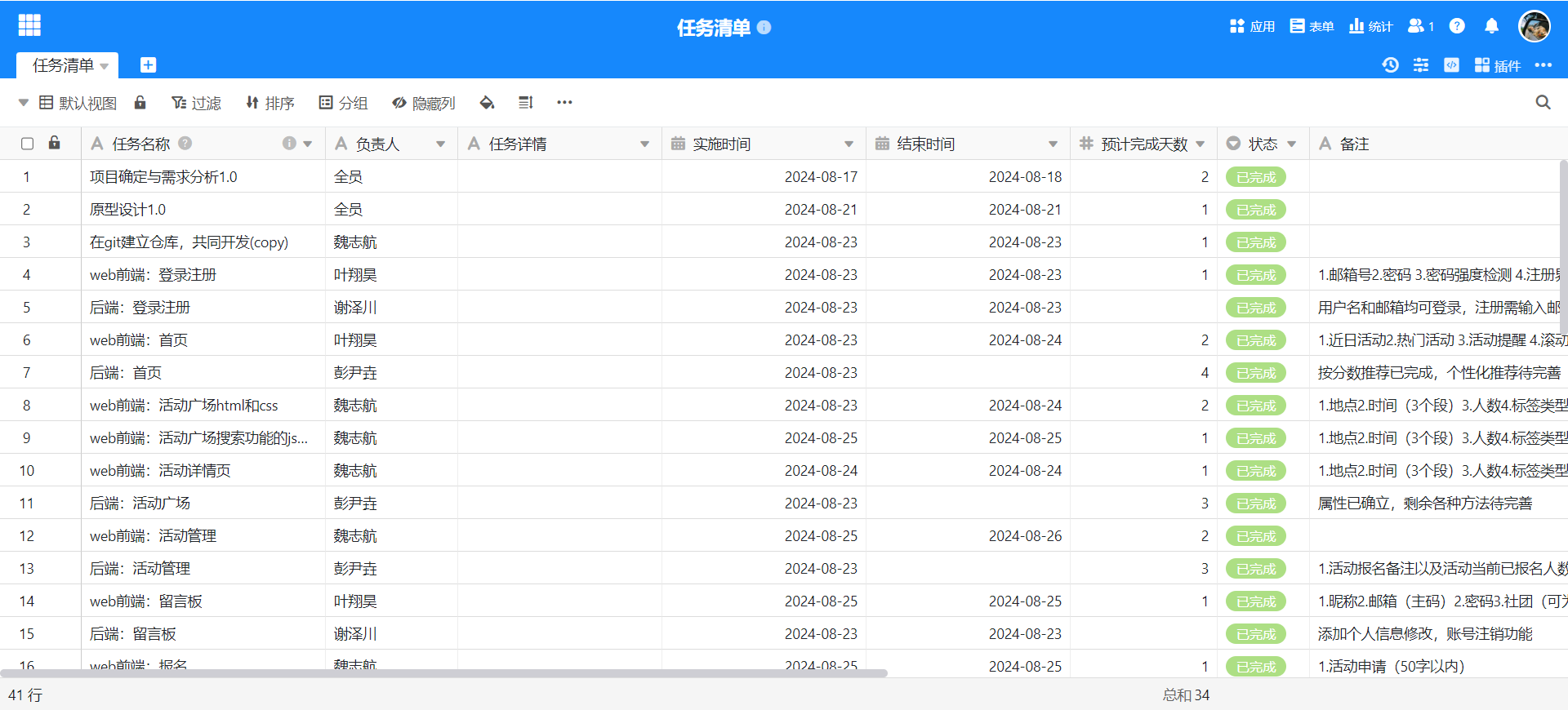
}

});

</script>

</body>

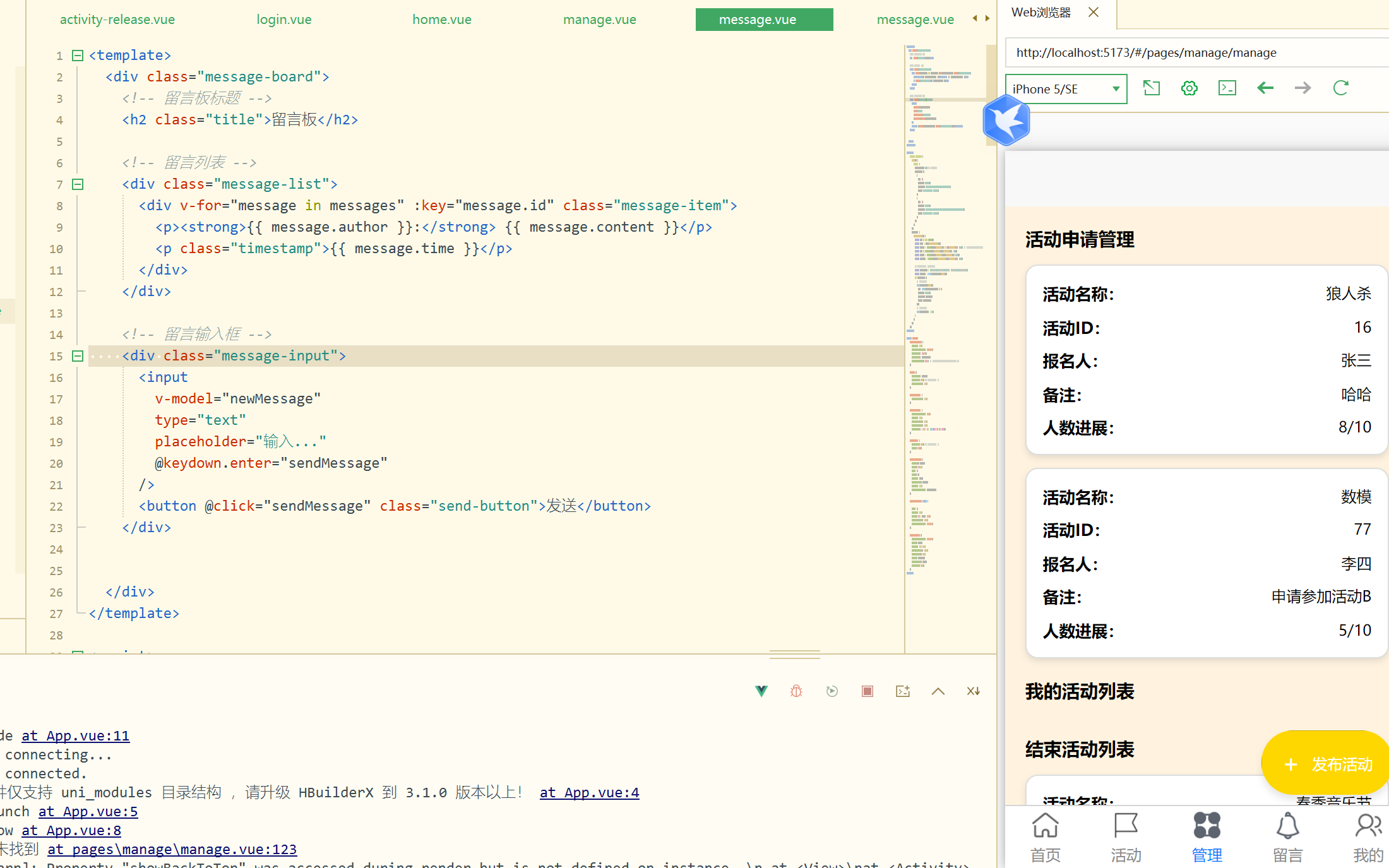
</html>



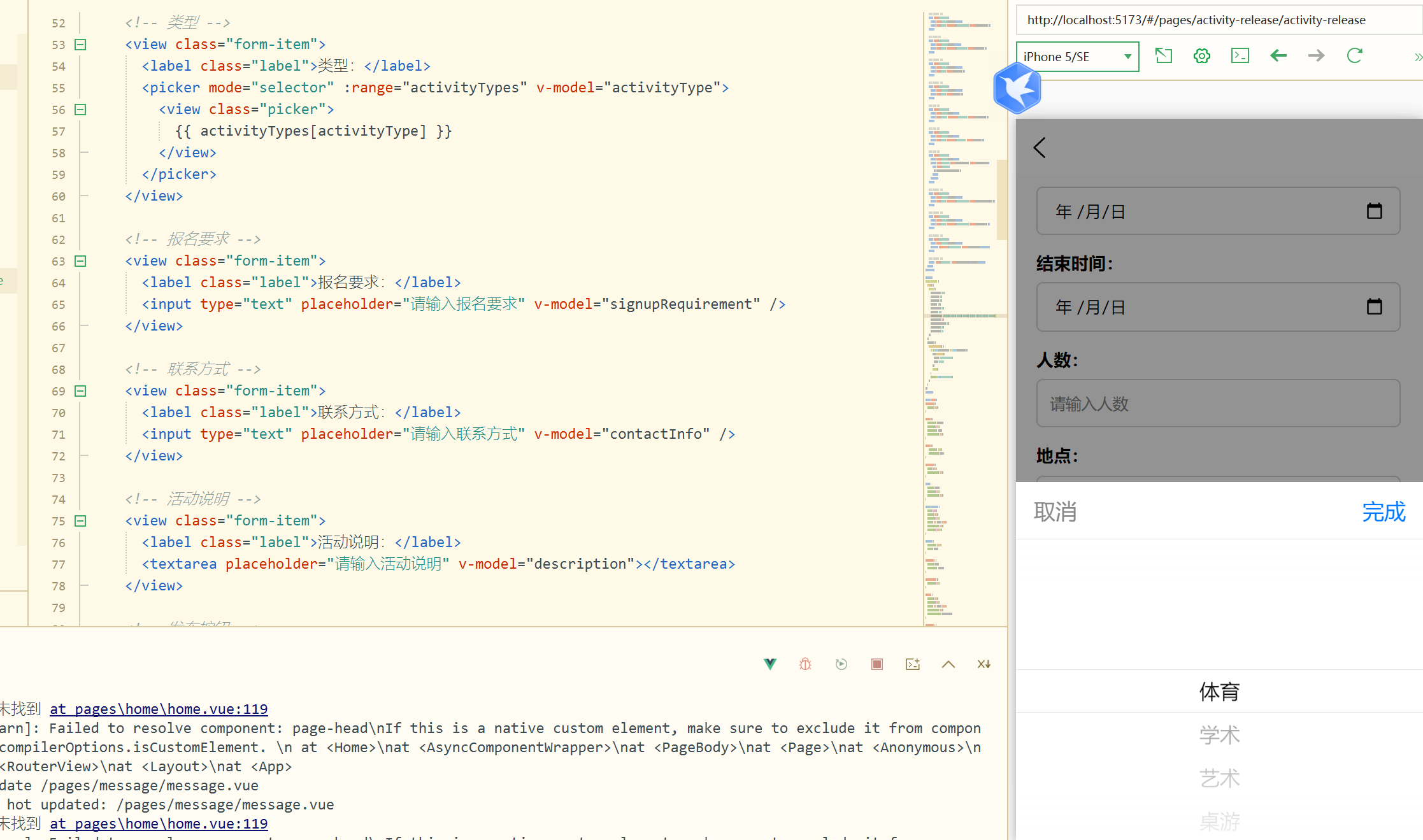
**兰天阳**

2024.8.24

编写了活动管理界面，实现了活动列表中的活动添加和删除



实现了活动发布界面，通过传输图片、名称等数据创建新活动



实现了登录、注册界面的前后端连接，同时在登录时获得userid，再通过uni.setStorageSync('userid', res.data.data.id)存储到全局，供后面界面调用，在传递活动id等全局参数时也可以用到此方法



实现了个人信息、近期活动和热门活动的前后端对接