一、平台用户管理-用户列表

前面我们完成了用户登录、用户认证与就诊人，现在我们需要把这些信息在我们的平台管理系统中进行统一管理

1、用户列表接口

（1）添加UserInfoService接口

```
//用户列表（条件查询带分页）
IPage<UserInfo> selectPage(Page<UserInfo> pageParam, UserInfoQueryVo userInfoQueryVo)

```

（2）添加UserInfoServiceImpl实现

```
@Override
public IPage<UserInfo> selectPage(Page<UserInfo> pageParam, UserInfoQueryVo userInfoQ
    //UserInfoQueryVo获取条件值
    String name = userInfoQueryVo.getKeyword(); //用户名称
    Integer status = userInfoQueryVo.getStatus();//用户状态
    Integer authStatus = userInfoQueryVo.getAuthStatus(); //认证状态
    String createTimeBegin = userInfoQueryVo.getCreateTimeBegin(); //开始时间
    String createTimeEnd = userInfoQueryVo.getCreateTimeEnd(); //结束时间
    //对条件值进行非空判断
    QueryWrapper<UserInfo> wrapper = new QueryWrapper<>();
    if(!StringUtils.isEmpty(name)) {
        wrapper.like("name",name);
    }
    if(!StringUtils.isEmpty(status)) {
        wrapper.eq("status",status);
    }
    if(!StringUtils.isEmpty(authStatus)) {
        wrapper.eq("auth_status",authStatus);
    }
    if(!StringUtils.isEmpty(createTimeBegin)) {
        wrapper.ge("create_time",createTimeBegin);
    }
```

```
    if(!StringUtils.isEmpty(createTimeEnd)) {
        wrapper.le("create_time",createTimeEnd);
    }
    //调用mapper的方法
    //调用mapper的方法
    IPage<UserInfo> pages = baseMapper.selectPage(pageParam, wrapper);
    //编号变成对应值封装
    pages.getRecords().stream().forEach(item -> {
        this.packageUserInfo(item);
    });
    return pages;
}

//编号变成对应值封装
private UserInfo packageUserInfo(UserInfo userInfo) {
    //处理认证状态编码
    userInfo.getParam().put("authStatusString",AuthStatusEnum.getStatusNameByStatus(u
    //处理用户状态 0  1
    String statusString = userInfo.getStatus().intValue()==0 ?"锁定" : "正常";
    userInfo.getParam().put("statusString",statusString);
    return userInfo;
}
```

（3）添加UserController方法

```
@RestController
@RequestMapping("/admin/user")
public class UserController {

    @Autowired
    private UserInfoService userInfoService;

    //用户列表（条件查询带分页）
    @GetMapping("{page}/{limit}")
    public R list(@PathVariable Long page,
                  @PathVariable Long limit,
                  UserInfoQueryVo userInfoQueryVo) {
        Page<UserInfo> pageParam = new Page<>(page,limit);
        IPage<UserInfo> pageModel =
                userInfoService.selectPage(pageParam,userInfoQueryVo);
```

```
        return R.ok().data("pageModel",pageModel);
    }
}
```

2、用户列表前端整合

（1）添加用户管理路由

在 src/router/index.js 文件添加路由

```
{
    path: '/user',
        component: Layout,
            redirect: '/user/userInfo/list',
                name: 'userInfo',
                    meta: { title: '用户管理', icon: 'table' },
                        alwaysShow: true,
                            children: [
                                {
                                    path: 'userInfo/list',
                                    name: '用户列表',
                                    component: () =>import('@/views/user/userInfo/lis
                                    meta: { title: '用户列表', icon: 'table' }
                                }
                            ]
},
```

（2）封装api请求

创建/api/yygh/userinfo.js

```
import request from '@/utils/request'

const api_name = '/admin/user'

export default {

```

```
    getPageList(page, limit, searchObj) {
        return request({
            url: `${api_name}/${page}/${limit}`,
            method: 'get',
            params: searchObj
        })
    }
}
```

（3）添加列表页面

创建views/user/userInfo/list.vue

```
<template>
    <div class="app-container">
        <!--查询表单-->
        <el-form  :inline="true" class="demo-form-inline">
            <el-form-item>
                <el-input v-model="searchObj.keyword" placeholder="姓名/手机"/>
            </el-form-item>

            <el-form-item  label="创建时间">
                <el-date-picker
                v-model="searchObj.createTimeBegin"
                type="datetime"
                placeholder="选择开始时间"
                value-format="yyyy-MM-dd HH:mm:ss"
                default-time="00:00:00"
                />
            </el-form-item>
            至
            <el-form-item>
                <el-date-picker
                v-model="searchObj.createTimeEnd"
                type="datetime"
                placeholder="选择截止时间"
                value-format="yyyy-MM-dd HH:mm:ss"
                default-time="00:00:00"
                />
            </el-form-item>

```

```
            <el-button type="primary" icon="el-icon-search" @click="fetchData()">查询
            <el-button type="default" @click="resetData()">清空</el-button>
        </el-form>

        <!-- 列表 -->
        <el-table
        v-loading="listLoading"
        :data="list"
        stripe
            style="width: 100%">

            <el-table-column
            label="序号"
            width="70"
            align="center">
                <template slot-scope="scope">
                        {{ (page - 1) * limit + scope.$index + 1 }}
                </template>
            </el-table-column>

            <el-table-column prop="phone" label="手机号"/>
            <el-table-column prop="nickName" label="昵称"/>
            <el-table-column prop="name" label="姓名"/>
            <el-table-column label="状态" prop="param.statusString"/>
            <el-table-column label="认证状态" prop="param.authStatusString"/>
            <el-table-column prop="createTime" label="创建时间"/>

            <el-table-column label="操作" width="200" align="center">
            </el-table-column>
        </el-table>

        <!-- 分页组件 -->
        <el-pagination
        :current-page="page"
        :total="total"
        :page-size="limit"
        :page-sizes="[5, 10, 20, 30, 40, 50, 100]"
        style="padding: 30px 0; text-align: center;"
        layout="sizes, prev, pager, next, jumper, ->, total, slot"
        @current-change="fetchData"
        @size-change="changeSize"
        />
    </div>
```

```
</template>
<script>
import userInfoApi from '@/api/yygh/userinfo'
export default {
    // 定义数据
    data() {
        return {
            listLoading: true, // 数据是否正在加载
            list: null, // banner列表
            total: 0, // 数据库中的总记录数
            page: 1, // 默认页码
            limit: 10, // 每页记录数
            searchObj: {} // 查询表单对象
        }
    },
    // 当页面加载时获取数据
    created() {
        this.fetchData()
    },
    methods: {
    // 调用api层获取数据库中的数据
    fetchData(page = 1) {
        console.log('翻页。。。' + page)
        // 异步获取远程数据（ajax）
        this.page = page
            userInfoApi.getPageList(this.page, this.limit, this.searchObj).then(
                response => {
                this.list = response.data.pageModel.records
                this.total = response.data.pageModel.total
                // 数据加载并绑定成功
                this.listLoading = false
            }
        )
    },
    // 当页码发生改变的时候
    changeSize(size) {
        console.log(size)
        this.limit = size
        this.fetchData(1)
    },
    // 重置查询表单
    resetData() {
        console.log('重置查询表单')
```

```
        this.searchObj = {}
        this.fetchData()
    }
  }
}
</script>
```

二、平台用户管理-用户锁定

1、用户锁定接口

（1）添加UserInfoService接口和实现

```
/**
 * 用户锁定
 * @param userId
* @param status 0：锁定 1：正常
 */
void lock(Long userId, Integer status);

//实现方法
@Override
public void lock(Long userId, Integer status) {
    if(status.intValue() == 0 || status.intValue() == 1) {
        UserInfo userInfo = this.getById(userId);
        userInfo.setStatus(status);
        this.updateById(userInfo);
    }
}
```

（2）添加UserController方法

```
@ApiOperation(value = "锁定")
@GetMapping("lock/{userId}/{status}")
public R lock(
```

```
    @PathVariable("userId") Long userId,
    @PathVariable("status") Integer status){
    userInfoService.lock(userId, status);
    return R.ok();
}
```

2、用户锁定前端整合

（1）封装api方法

在api/yygh/userinfo.js添加方法

```
lock(id, status) {
    return request({
        url: `${api_name}/lock/${id}/${status}`,
        method: 'get'
    })
}
```

（2）在用户列表页面添加组件

```
<el-table-column label="操作" width="200" align="center">
    <template slot-scope="scope">
        <el-button v-if="scope.row.status == 1" type="primary" size="mini" @click="lo
        <el-button v-if="scope.row.status == 0" type="danger" size="mini" @click="loc
    </template>
</el-table-column>
```

（3）在页面列表页面添加方法

```
// 锁定
lock(id, status) {
    this.$confirm('确定该操作吗?', '提示', {
        confirmButtonText: '确定',
        cancelButtonText: '取消',
```

```
        type: 'warning'
        }).then(() => { // promise
            // 点击确定，远程调用ajax
            return userInfoApi.lock(id, status)
        }).then((response) => {
            this.fetchData(this.page)
            if (response.code) {
                this.$message({
                type: 'success',
                message: '操作成功!'
            })
        }
    })
}
```

三、平台用户管理-用户详情

详情展示用户信息、用户就诊人信息

1、用户详情接口

（1）添加UserInfoService接口

```
/**
 * 详情
 * @param userId
* @return
*/
Map<String, Object> show(Long userId);

```

（2）添加UserInfoServiceImpl实现

```
@Autowired
private PatientService patientService;

@Override
```

```
public Map<String, Object> show(Long userId) {
    Map<String,Object> map = new HashMap<>();
    //根据userid查询用户信息
    UserInfo userInfo = this.packageUserInfo(baseMapper.selectById(userId));
    map.put("userInfo",userInfo);
    //根据userid查询就诊人信息
    List<Patient> patientList = patientService.findAllUserId(userId);
    map.put("patientList",patientList);
    return map;
}
```

（3）添加UserController方法

```
//用户详情
@GetMapping("show/{userId}")
public R show(@PathVariable Long userId) {
    Map<String,Object> map = userInfoService.show(userId);
    return R.ok().data(map);
}

```

2、用户详情前端

（1）添加路由

```
{
    path: 'userInfo/show/:id',
        name: '用户查看',
            component: () =>import('@/views/user/userInfo/show'),
                meta: { title: '用户查看' },
                    hidden: true
}
```

（2）封装api方法

在api/yygh/userinfo.js添加

```
//用户详情
show(id) {
    return request({
      url: `${api_name}/show/${id}`,
      method: 'get'
    })
}
```

（3）修改列表组件，添加查看按钮

```
 <router-link :to="'/user/userInfo/show/'+scope.row.id">
     <el-button type="primary" size="mini">查看</el-button>
</router-link>
```

（4）添加详情页面

添加/views/user/userInfo/show.vue组件

```
<template>
    <div class="app-container">
        <h4>用户信息</h4>
        <table class="table table-striped table-condenseda table-bordered" width="100
        <tbody>
            <tr>
                <th width="15%">手机号</th>
                <td width="35%"><b>{{ userInfo.phone }}</b></td>
                <th width="15%">用户姓名</th>
                <td width="35%">{{ userInfo.name }}</td>
            </tr>
        <tr>
            <th>状态</th>
            <td>{{ userInfo.status == 0 ? '锁定' : '正常' }}</td>
            <th>注册时间</th>
            <td>{{ userInfo.createTime }}</td>
        </tr>
        </tbody>
        </table>
        <h4>认证信息</h4>
```

```
        <table class="table table-striped table-condenseda table-bordered" width="10(
            <tbody>
            <tr>
                <th width="15%">姓名</th>
                <td width="35%"><b>{{ userInfo.name }}</b></td>
                <th width="15%">证件类型</th>
                <td width="35%">{{ userInfo.certificatesType }}</td>
            </tr>
            <tr>
                <th>证件号</th>
                <td>{{ userInfo.certificatesNo }}</td>
                <th>证件图片</th>
                <td><img :src="userInfo.certificatesUrl" width="80px"></td>
            </tr>
            </tbody>
        </table>
    <h4>就诊人信息</h4>
    <el-table
        v-loading="listLoading"
        :data="patientList"
        stripe
            style="width: 100%">
        <el-table-column
        label="序号"
        width="70"
        align="center">
        <template slot-scope="scope">
                {{ scope.$index + 1 }}
        </template>
        </el-table-column>

        <el-table-column prop="name" label="姓名"/>
        <el-table-column prop="param.certificatesTypeString" label="证件类型"/>
        <el-table-column prop="certificatesNo" label="证件编号"/>
        <el-table-column label="性别">
        <template slot-scope="scope">
                {{ scope.row.sex == 1 ? '男' : '女' }}
        </template>
        </el-table-column>
        <el-table-column prop="birthdate" label="出生年月"/>
        <el-table-column prop="phone" label="手机"/>
        <el-table-column label="是否结婚">
        <template slot-scope="scope">
```

```
                {{ scope.row.isMarry == 1 ? '时' : '否' }}
        </template>
    </el-table-column>
    <el-table-column prop="param.fullAddress" label="地址"/>
    <el-table-column prop="createTime" label="注册时间"/>
    </el-table>
    <br>
    <el-row>
    <el-button  @click="back">返回</el-button>
    </el-row>
    </div>
</template>
<script>
import userInfoApi from '@/api/yygh/userinfo'
export default {
    // 定义数据
    data() {
        return {
            id: this.$route.params.id,
            userInfo: {}, // 会员信息
            patientList: [] // 就诊人列表
        }
    },
    // 当页面加载时获取数据
    created() {
        this.fetchDataById()
    },
    methods: {
        // 根据id查询会员记录
        fetchDataById() {
            userInfoApi.show(this.id).then(response => {
                this.userInfo = response.data.userInfo
                this.patientList = response.data.patientList
            })
        },
        back() {
            window.history.back(-1)
        }
    }
}
</script>
```