### 编写node路由，连接后台服务

1. 连接代理

const TAF = require('@taf/taf-rpc').Communicator.New();

const Test = require('./UserProxy').Test;

let servant = 'Test.UserZjServer.UserZjObj';

if (!process.env.TAF\_CONFIG) {

    servant += '@tcp -h 127.0.0.1 -p 14001 -t 60000';

}

const proxy = TAF.stringToProxy(Test.UserProxy, servant);

*exports*.Test = Test;

*exports*.proxy = proxy;

1. 编写路由
   * + 1. 登录

router.post('/loginForAd', (*req*, *res*) => {

    const { username, password } = req.body;

    const stReq = **new** *Test*.*LoginReq*();

    stReq.readFromObject({

        username,

        password,

    });

    proxy.loginForAd(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

* + - 1. 添加用户

// 增加用户接口

router.post('/saveUser', (*req*, *res*) => {

    const { username, password, tel, gender, age } = req.body;

    const stReq = **new** *Test*.*SaveUserInfoReq*();

    stReq.readFromObject({

        userinfo: {

            username,

            password,

            gender,

            tel,

            age

        }

    });

    proxy.saveUser(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

* + - 1. 获取用户列表

// 获取用户列表

router.get('/getUserListForAd', (*req*, *res*) => {

    const { pageNum, pageSize } = req.query;

    const stReq = **new** *Test*.*PageInfoReq*();

    stReq.readFromObject({

        pageNum,

        pageSize

    });

    proxy.getUserListForAd(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

* + - 1. 删除用户

// 删除用户

router.get('/delUser', (*req*, *res*) => {

    const { uid } = req.query;

    const stReq = **new** *Test*.*QueryIdReq*();

    stReq.readFromObject({

        uid

    });

    proxy.delUser(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

* + - 1. 修改用户

// 修改用户

router.post('/updateUser', (*req*, *res*) => {

    const { uid, username, password, tel, gender, age } = req.body;

    const stReq = **new** *Test*.*SaveUserInfoReq*();

    stReq.readFromObject({

        userinfo: {

            uid,

            username,

            password,

            gender,

            tel,

            age

        }

    });

    proxy.saveUser(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

* + - 1. 用户详细信息

// 用户详细

router.get('/getUserDetailForAd', (*req*, *res*) => {

    const { uid } = req.query;

    const stReq = **new** *Test*.*QueryIdReq*();

    stReq.readFromObject({

        uid

    });

    proxy.getUserDetailForAd(stReq).then(*ret* => {

        jsonWrite(res, ret.response.arguments.stRsp.toObject());

    }, *ret* => {

*console*.log('error');

    });

});

### 设计界面，编写页面逻辑请求数据

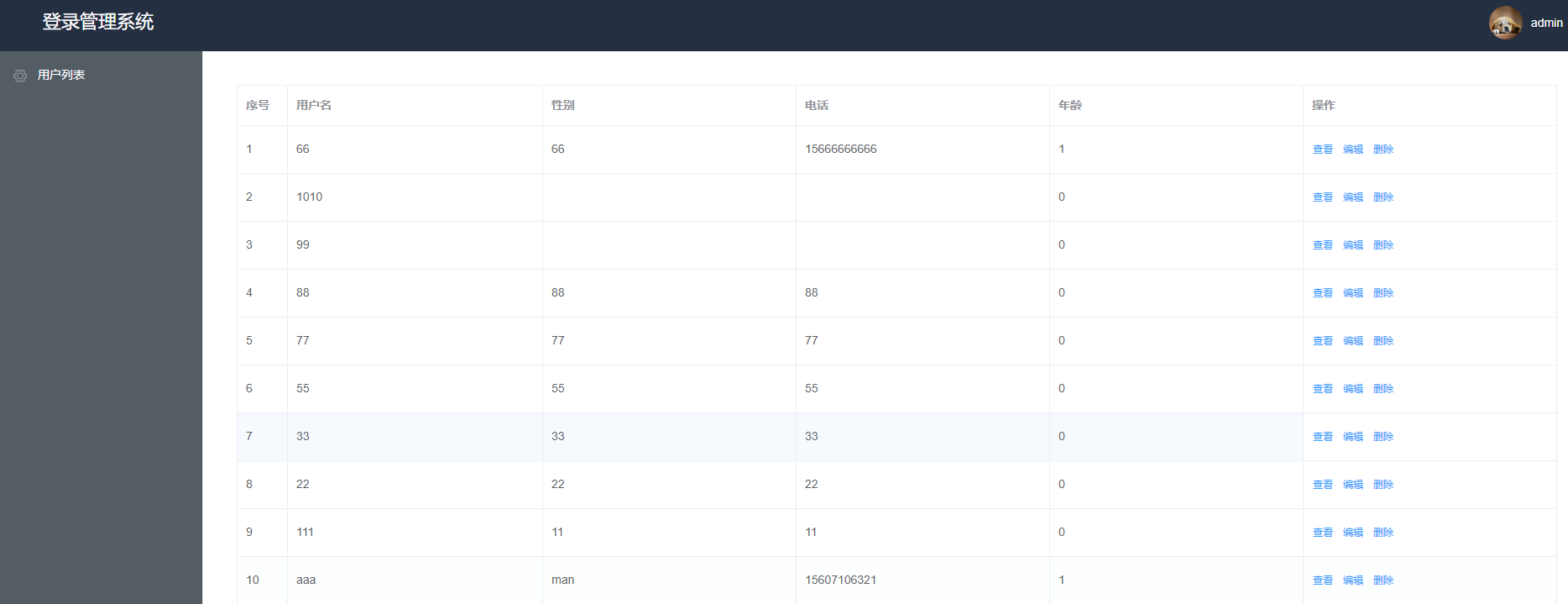
1. 登录



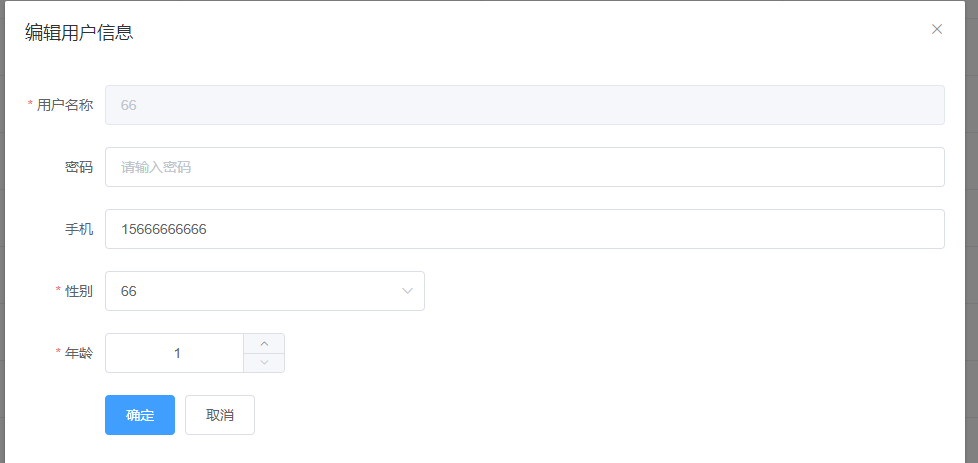
1. 注册



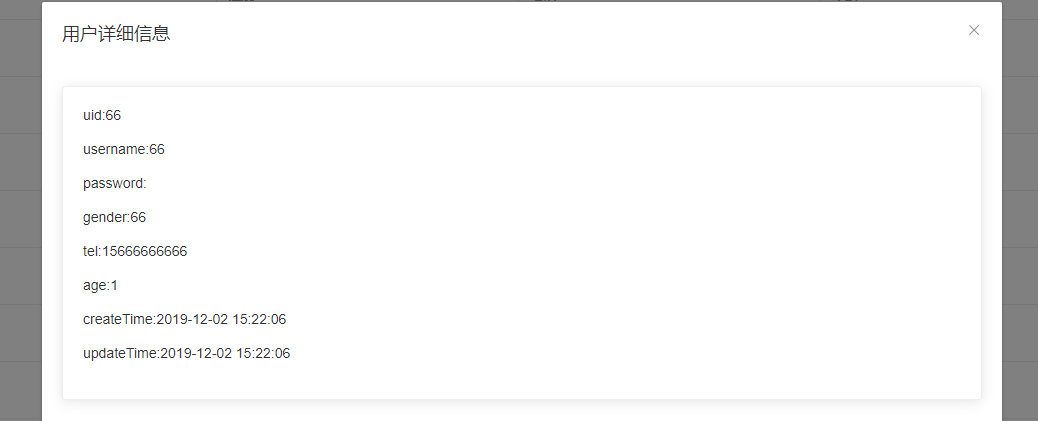
1. 用户列表



1. 编辑



1. 详情



### 明日计划

熟悉dcache api,完成dcache操作