

轻量化第三次小作业

21301114 俞贤皓

1. 简介

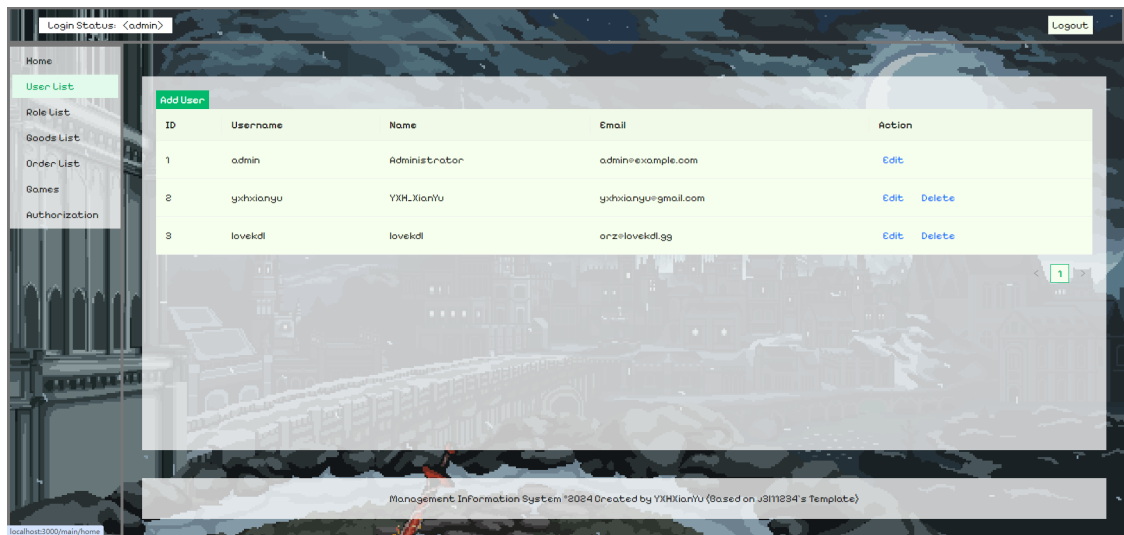
- 作业内容：信息管理系统
- 本次作业内容开源在 [我的Github](#)
- 我完成了以下功能
 - 纯React静态页面 60分
 - 后台用户管理 10分
 - 数据持久化 10分
 - 后端联动 20分
 - **总分 100分** (60+10+10+20)
- 注意：我参考了JS老师提供的第三次作业模板，但我对模板学习后，进行了 **大幅度** 的修改
 - 修改内容包括但不限于：
 - **代码风格**：我觉得React把所有内容塞在return里的风格非常不优雅，所以我限制 **一个组件必须在一行内** 编写完成。多余的部分应该在函数体内进行定义。

■ 例如：

```
53  return (
1   <MyForm onFinish={onFinish}>
2     <Form.Item style={...styleMargin} label="Username" name="username" rules={usernameRules} >
3       <Input />
4     </Form.Item>
5     <Form.Item style={...styleMargin} label="Password" name="password" rules={passwordRules} >
6       <Input.Password />
7     </Form.Item>
8     <Form.Item style={...styleMargin, ...styleCentered} name="remember" valuePropName="checked" >
9       <Checkbox>Remember Me</Checkbox>
10    </Form.Item>
11    <Form.Item style={...styleMargin, ...styleCentered}>
12      <Space>
13        <Button type="primary" htmlType="submit"> Login </Button>
14        <Button onClick={() => { navigate("/register") }}> Register </Button>
15      </Space>
16    </Form.Item>
17    <Alert style={...styleMargin, marginTop: '40px'} message="Tips: default user is admin/123456Aa" type="info" />
18  </MyForm>
19 )
```

限制所有组件必须在一行内，
这样结构看起来会非常清晰

- **后端与异步**：因为我添加了后端，所以前端需要通过axios发送请求。而发送请求就带来了异步问题，我将数据交换 **改为了异步**，并且添加了对应的States，**使得数据能够正确刷新**
 - **封装**：模板中存在一些代码，在多个地方出现，需要抽象和封装。所以我对这些代码进行了抽象和封装，例如：**MyForm** (Login和Register的表格)、**Rules** (Input输入的规则) 等。
 - **UI与风格**：改变了UI风格为 **我最喜欢的像素风格**：像素字体 + 像素背景 + 扁平化UI
- 预览
 -



2. 代码说明

- 前端位于 `./frontend`
 - 构建: `npm install`
 - 运行: `npm start`
 - 根据JS老师模板修改的前端, 基于 React + Ant Design
 - 预编译版本位于 `./frontend/build`
 - 在 `./frontend` 目录下执行

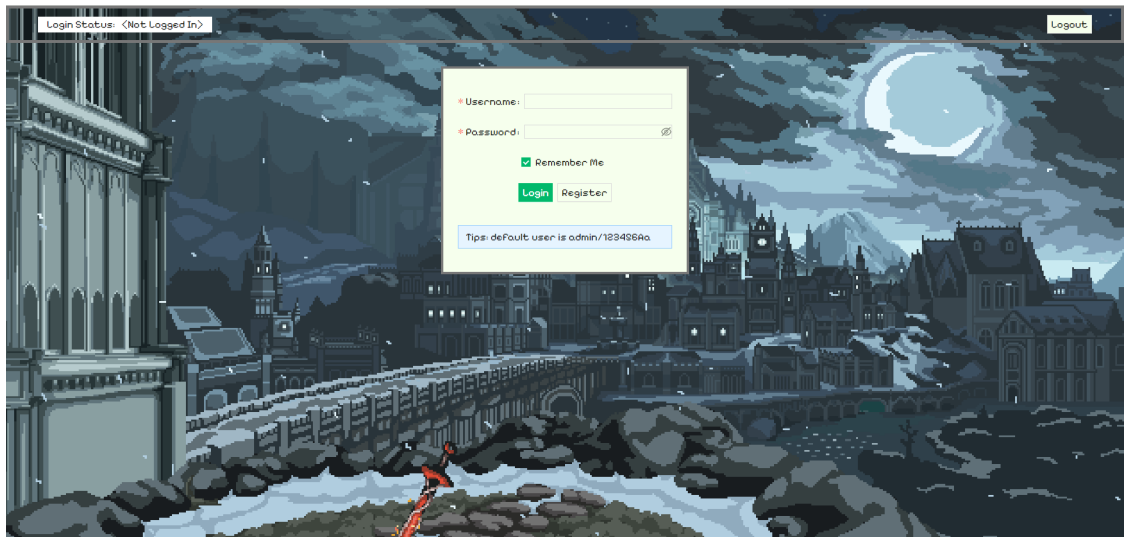
```
npm install -g serve  
serve -s ./build
```

- 即可启动前端
- 后端位于 `./backend`
 - 构建: `npm install`
 - 运行: `npm start` 或 `node ./index.js`
 - 使用express.js实现的一个 **极端轻量化** 的 **Redis风格** 后端
 - 后端无预编译版本 (因为本身就只有单文件)

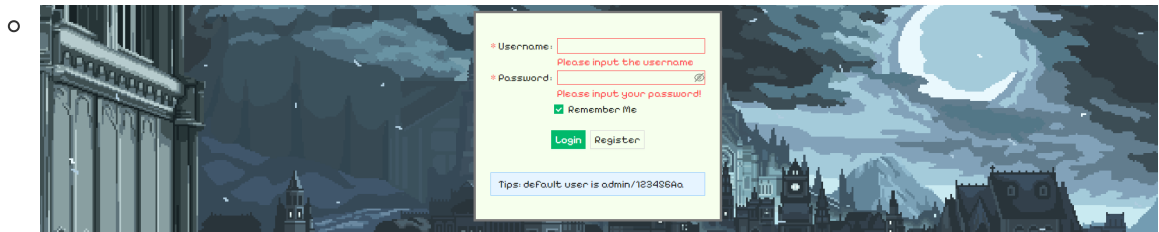
3. 功能说明

3.1 纯React静态页面 (60分)

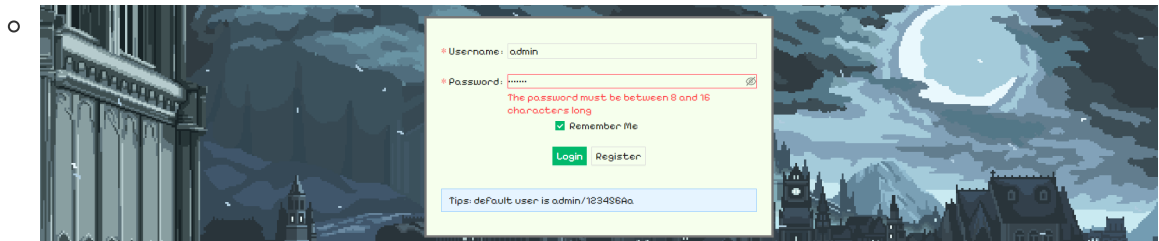
- 登录页面
 -



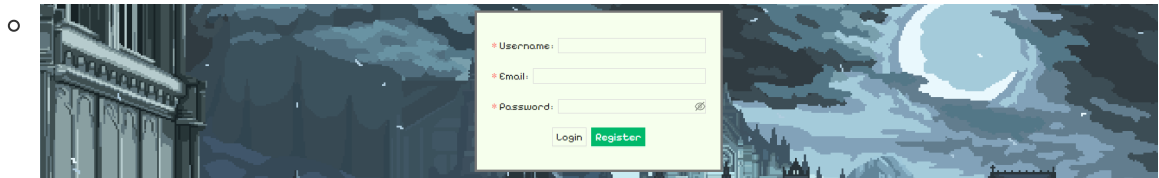
- 登陆页面（缺少输入）



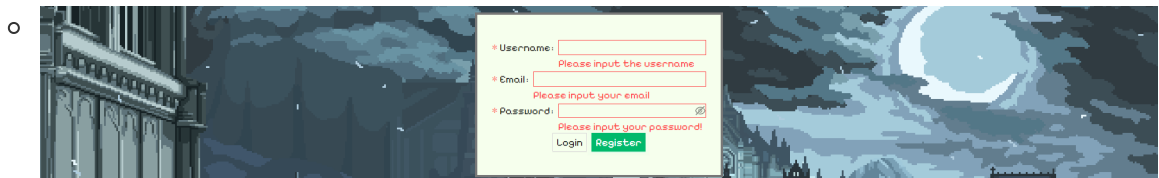
- 登陆页面（密码不够长）



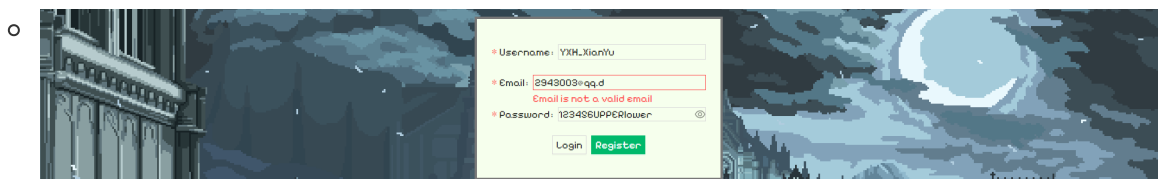
- 注册页面



- 注册页面（缺少输入）



- 注册页面（邮箱格式不正确）



- 注册页面（密码缺少大写字母或小写字母）

○

Username: YXH_XianYu

Email: 2943003@qq.com

Password: 123456lower

The password must contain at least one uppercase letter, one lowercase letter, and one number

Login Register

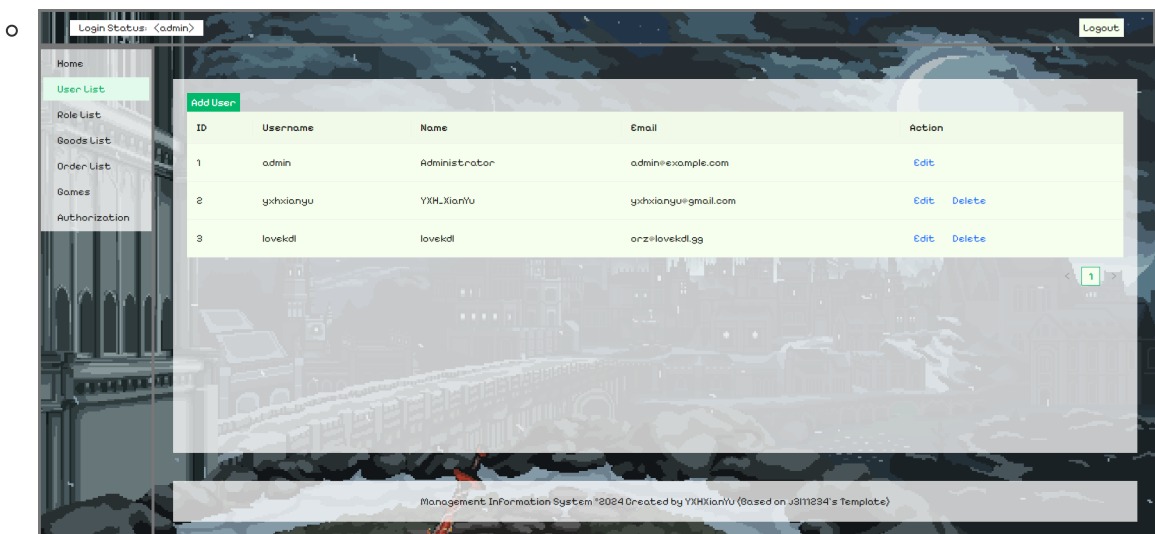
- 主页面



- 其他页面

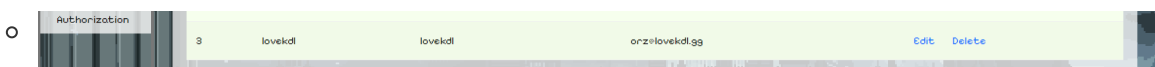


- 用户管理界面

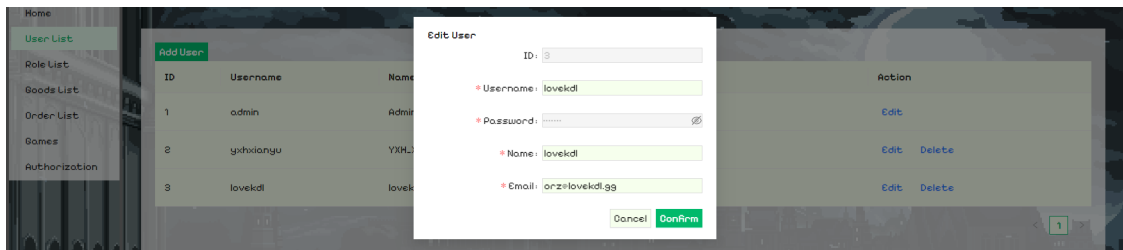


3.2 后台用户管理（10分）

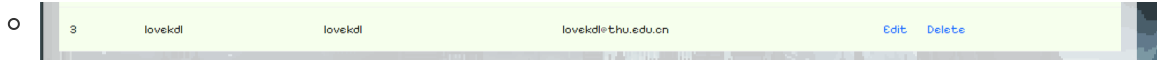
- 修改用户（修改lovekdl用户的邮箱，修改前）



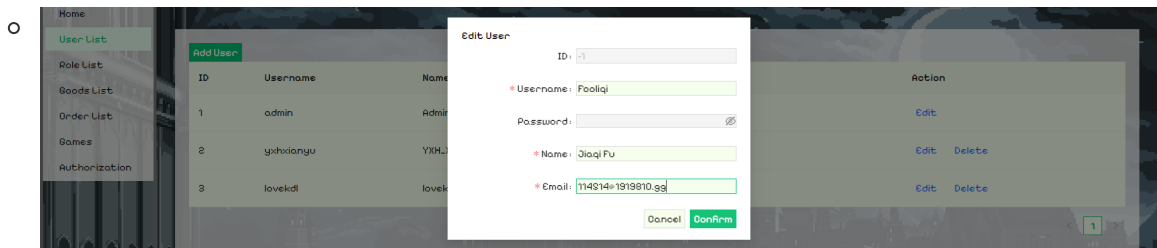
○



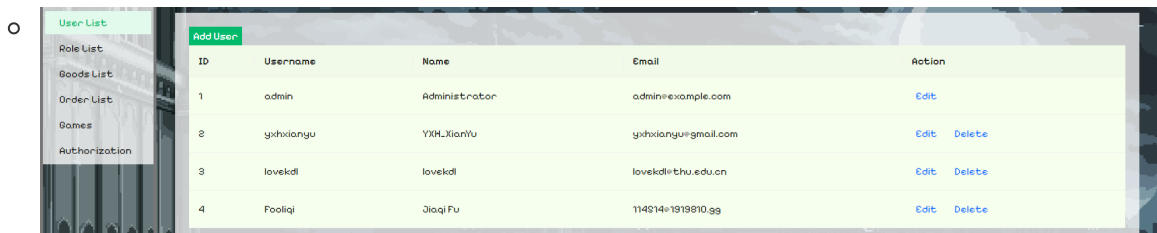
- 修改用户（修改lovekdl用户的邮箱，修改后）



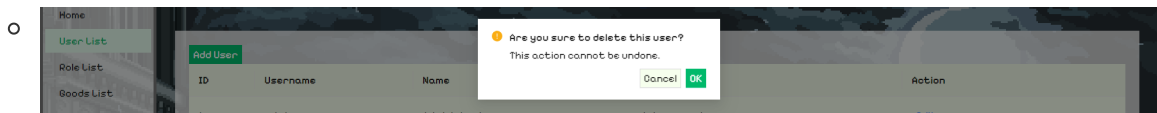
- 添加用户（添加FJQ同学）



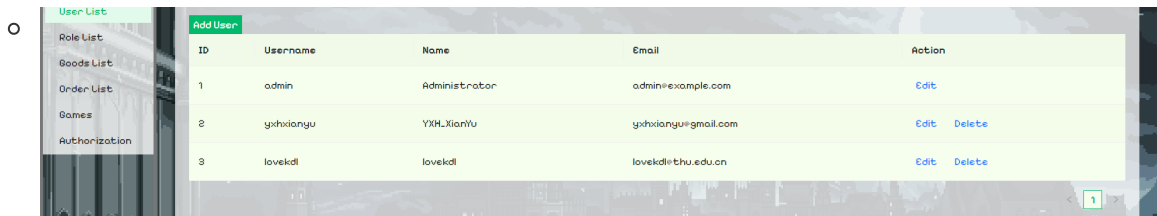
- 添加用户（结果）



- 删除用户（删除FJQ同学）



- 删除用户（结果）



- 重启浏览器，发现数据仍保留



6.3 数据持久化（10分）

- 经检查，数据已经同步到后端数据库中

- 删除前

■

```
16      {
17        "id": 3,
18        "username": "lovekd1",
19        "password": "3008453",
20        "name": "lovekd1",
21        "email": "lovekd1@thu.edu.cn"
22      },
23      {
24        "id": 4,
25        "username": "Fooliq1",
26        "name": "Jiaqi Fu",
27        "email": "114514@1919810.gg"
28      }
29    ]
30  }
}
```

- 删除后

```
3      "name": "Administrator",
4      "email": "admin@example.com"
5    },
6    {
7      "id": 1,
8      "username": "yhxixianyu",
9      "password": "3008453",
10     "name": "YXH_XianYu",
11     "email": "yhxixianyu@gmail.com"
12   },
13   {
14     "id": 3,
15     "username": "lovekd1",
16     "password": "3008453",
17     "name": "lovekd1",
18     "email": "lovekd1@thu.edu.cn"
19   }
20 ]
}
```

6.4 后端联动 (20分)

- 后端日志

```
2024/5/27 14:28:09 get { key: 'userList' }
true
2024/5/27 14:28:18 set userList
2024/5/27 14:28:21 set userList
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:24 get { key: 'userList' }
true
2024/5/27 14:28:27 get { key: 'userList' }
true
2024/5/27 14:28:57 set userList
true
```

- 后端核心代码

```
11 app.post('/set', (req, res) => {
12   console.log(getTime(), chalk.red('set'), req.body.key)
13   const { key, value } = req.body
14   if (!key || value === undefined) {
15     return res.status(400).json({ error: 'Key and value are required' })
16   }
17   const data = readData()
18   data[key] = value
19   writeData(data)
20   res.json({ success: true })
21 }
22
44 You, 12 hours ago • backend
1 app.get('/get/:key', (req, res) => {
2   console.log(getTime(), chalk.red('get'), req.params)
3   const { key } = req.params
4   const data = readData()
5   console.log(key in data)
6   if (!(key in data)) {
7     return res.status(404).json({ error: 'Key not found' })
8   }
9   // console.log(chalk.red('get result:'), data[key])
10  res.json({ key, value: data[key] })
11 }
```

- 前端交互部分代码:

-

```

14   async _getData() {
13       await axios.get('http://localhost:5000/get/userList')
12           .then(response => {
11               if (response.status === 200) {
10                   this.userList = response.data.value
9                   } else {
8                       this.userList = defaultUserList
7                       this._setData()
6                   }
5               }
4           })
3       .catch(error => {
2           console.log('Error fetching data: ', error)
1           this.userList = defaultUserList
133          this._setData()
        })
    }

1
2
3   async _setData() {
4       const data = {
5           'key': 'userList',
6           'value': this.userList,
7       }
8       await axios.post('http://localhost:5000/set', data)
9       .catch(error => {
10           console.log('Error saving data: ', error)
11       })
12   }

```

- 前端异步代码：检查用户是否登录，路由跳转

- ```

15
14 useEffect(() => {
13 const func = async () => {
12 const user = await userService.getCurrentUser()
11 if (!user) {
10 message.open({
9 type: "warning",
8 content: "Please login first",

```

- 前端异步代码：Header的登陆状态

- ```

10
9       const [ username, setUsername ] = useState('Not Logged In')
8       useEffect(() => {
7           const func = async () => {
6               const user = await userService.getCurrentUser()
5               if (user) setUsername(user.username)
4               else setUsername('Not Logged In')
3           }
2           func()
1       })
32

```