Vue前端小作业说明文档

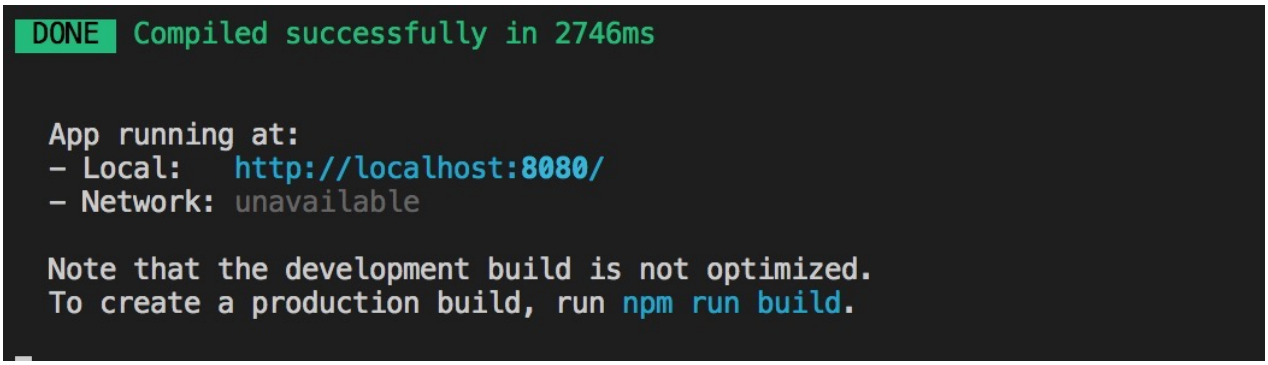
环境准备

- 安装npm
- 在项目根目录执行以下命令安装依赖：
  `npm install`

项目启动

```
npm run serve
```

通过以上命令启动dev server，在这种情况下，每次代码保存后都会自动编译，编译成功后期望有如下输出：



如图所示，此时我们启动了一个dev server在本机的8080端口提供web服务。在浏览器地址栏中输入

```
http://localhost:8080
```

可以看到如下页面

图中是一个未完成的留言板应用，本次作业的最终目标就是补全代码完成它。

本次作业分为四个阶段，每个阶段实现留言板的部分功能，并在一个单独的页面展示，除了最终阶段是访问上述地址外，其余各个阶段对应的页面地址分别为：

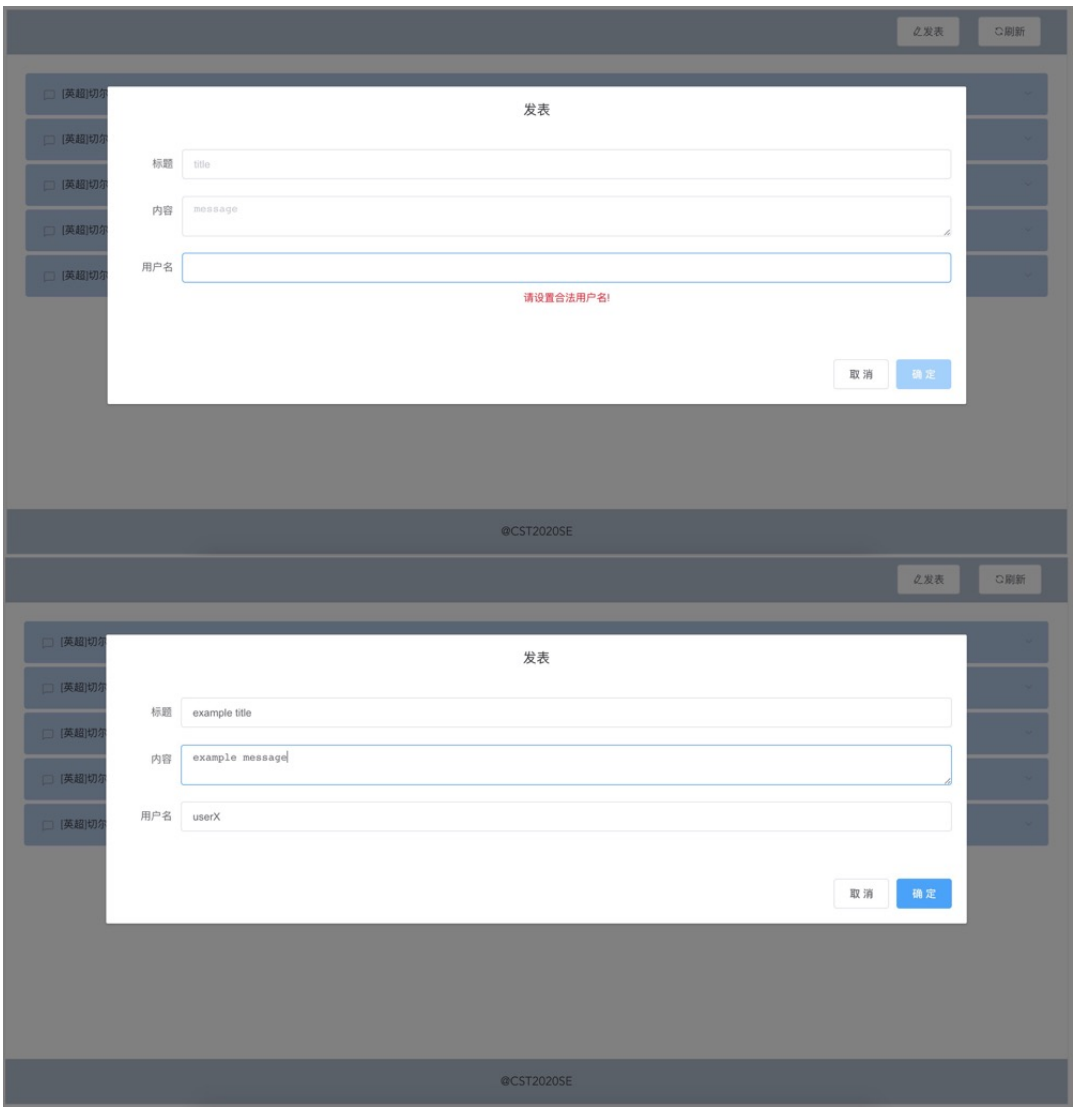
```
http://localhost:8080/stage1
http://localhost:8080/stage2
http://localhost:8080/stage3
```

作业需求与示例

在本作业中，各位同学需要通过补全助教提供的代码实现一个简单的留言板应用的前端部分，需求如下：

- 实现一个包含导航栏和信息板的留言板，如下所示：

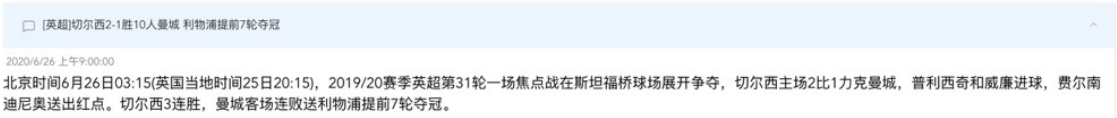
- 点击导航栏的发表按钮触发弹出对话框，发表留言，点击刷新按钮更新留言
    - 发表留言：
        - 点击留言，用户名不合法时确定按钮不可用，并显示“请设置合法用户名”，用户名合法的情况下才允许发帖



        - 点击确定后将留言发给后端，根据后端返回内容，弹出对话框提示是否发表成功

        - 发表成功后留言列表更新

    - 请通过修改cookie实现页面中用户名的持续存储，使得发表留言后刷新页面，用户名仍然保存
    - 本次前端项目不做用户登录校验，本质上是一个匿名的留言板
  - 刷新页面：
    - 点击按钮后从后端获取最新的留言数据，更新页面内容
- 信息板：
  - 通过获取的消息列表逐条显示即可，要求显示标题、内容、用户名和时间



在这次小作业中，我们把留言板拆分成了若干vue组件，并基于它们设计了四个阶段的任务，通过按顺序完成这四个阶段的任务，我们期望各位同学能有以下收获：

- 学会使用vue框架html模板，循环渲染等特性
- 掌握vue框架组件的实现和使用
- 修改和删除cookie实现前端持久化存储
- 学会使用mock、devServer等方式调试

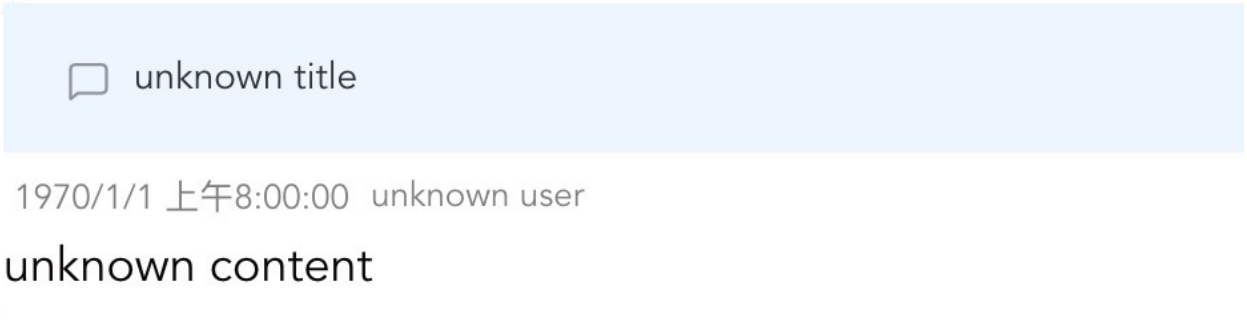
[阶段一(10%)]单条留言的展示

目标

- 完成MessageBlock组件从而实现展示一条留言的时间、用户、标题和内容

任务要求

在src/components/MessageBlock.vue中按照注释的要求完成代码填空，使用js代码中的数据和方法显示标题和时间，使得stage1页面能呈现下图效果



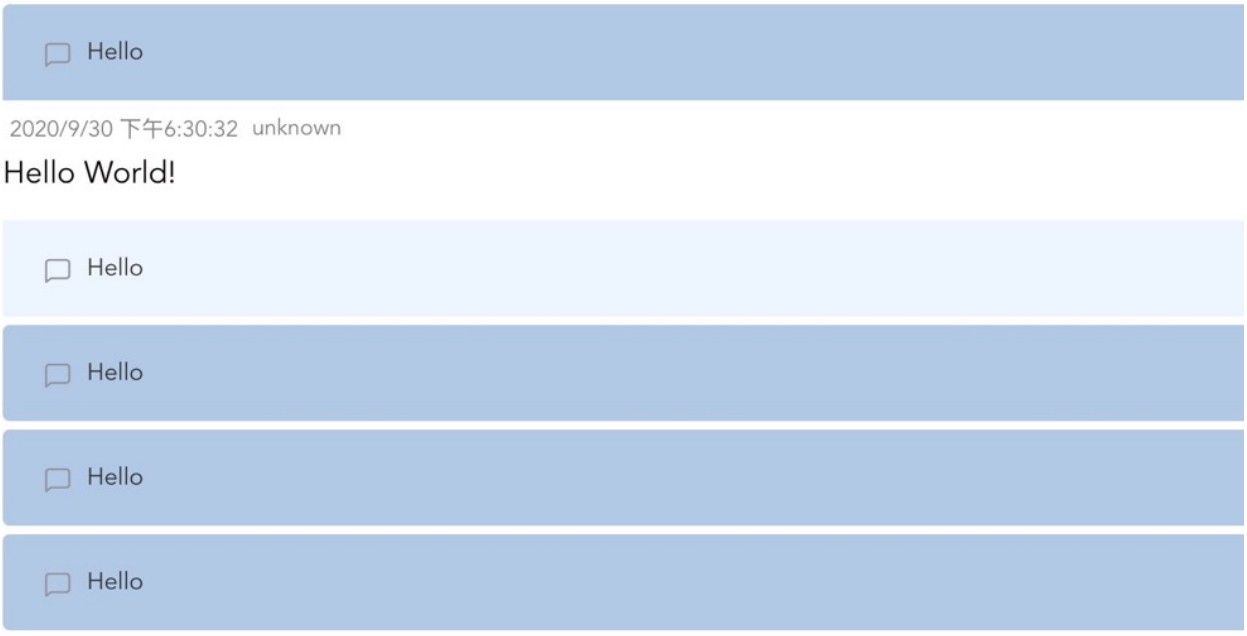
[阶段二(20%)]消息列表

目标

- 完成MessageList组件，展示多条留言

任务要求

在src/components/MessageList.vue中按照注释的要求完成代码填空，使用阶段1的MessageBlock组件和js代码中的数据显示若干条留言，使得stage2页面呈现下图效果



[阶段三(30%)]输入表单

目标

- 完成PostDialog组件，实现接收发表留言的输入
- 完成stage3页面，实现在前端将输入的留言更新到消息列表中

任务要求

- 在src/components/PostDialog.vue中按照注释要求完成代码填空，实现：
  - 标题，内容，用户名的输入
  - 点击确定后产生自定义事件，从而把输入的留言传递给父组件，并隐藏对话框
  - 点击取消后隐藏输入对话框
- 在src/views/Stage3.vue中按照注释要求完成代码填空，实现：
  - 监听postdialog组件产生的事件，接收数据，并更新messageList内容
    最终效果如下图，输入对话框可以输入留言标题，内容以及用户名，点击确定后对话框消失，页面出现一条新发表的留言

[最终阶段(40%)]留言板

目标

- 完成MessageBoard组件，实现完整的留言板

任务要求

- 在src/components/MessageBoard.vue中按照注释的要求完成代码填空，实现 作业需求与实例 描述的留言板
- 在src/utils/communication.js中实现用来向后端发送请求的函数（建议使用src/utils/API.js中定义好的API常量）

dev server模式下配置反向代理

- 在这个阶段中，你需要用到反向代理来把发给Django后端的请求通过dev server转发，配置方式如

下：

1. 在vue.config.js文件中修改图中红框里的地址为你的后端地址

2. 在src/utils/communication.js文件中注释掉mock的import代码从而停用mock server模式

接口说明

本作业中使用的接口即为后端小作业中实现的接口，其中在使用GET请求获取留言时不需要使用到limit和offset参数，按照该接口的默认行为调用即可（评测时不会出现超过100条留言的情况）

- 获取消息列表：
  - timestamp为13位时间戳

接口	方法	类型	示例
/api/message	GET	json	[{"title":"hello","content":"world","user":"Alice", "timestamp":1593133200000},{"title":"hello2","content":"world2","user":"Bob", "timestamp":1593133200000}]

- 发表留言
  - user字段请通过cookie传递

接口	方法	类型	示例	其他
/api/message	POST	json	{"title":"hello","content":"world"}	cookie中传递user字段

作业提交说明

- 完成作业的时间为两周，同学们需要在10月14日24：00之前提交至网络学堂
- 提交的内容包括所有源代码和配置文件，目录结构如下
  ├── babel.config.js
  ├── jest.config.js
  ├── vue.config.js
  ├── package-lock.json
  ├── package.json
  ├── public
  ├── src

└—— tests

- 不用提交node_modules目录的内容

评分标准

- 检测在dev server模式下进行
- 检测方式黑盒与白盒结合，部分功能黑盒测试，测试通过即得到对应分数，未通过则会补充白盒检查
- 所有阶段的任务均正确完成记满分，部分任务未完成酌情扣分，各阶段占比分别为10%,20%,30%,40%