非凡聊天室功能详细设计

详细设计说明书

1 引言

1.1 编写目的

本聊天室详细说明书的主要目的是为了清楚地介绍聊天室的功能、操作方法以及使用规则，从而帮助用户能够安全、有效地使用这一工具。以下是编写聊天室详细说明书时预期的读者群体：

(1)开发人员：他们需要了解系统的详细功能需求、技术架构、接口定义以及实现细节，以便正确且高效地构建系统。

(2)聊天室用户：他们需要了解如何注册、登录、使用聊天室的各项功能（如创建聊天室、加入聊天室、发送消息、设置个人偏好等）。

(3)投资者和高层管理人员：他们关心项目的商业价值、投资回报率以及市场前景。因此，说明书需要清晰地阐述聊天室系统的进度安排，预期的商业效果。

1.2 背景

a. 非凡聊天室

b.本项目的任务提出者：卓伊杰

开发者：卓伊杰，邵振峰，冯博，卢亚晗，袁硕，代小雨

用户：所有人员

运行该程序系统的计算中心：Windows 11 家庭中文版

1.3 定义

GO 是一个开源的编程语言，它能让构造简单、可靠且高效的软件变得容易。Go 是从 2007 年末由 Robert Griesemer, Rob Pike, Ken Thompson 主持开发，后来还加入了 Ian Lance Taylor, Russ Cox 等人，并最终于 2009 年 11 月开源，在 2012 年早些时候发布了 Go 1 稳定版本。现在 Go 的开发已经是完全开放的，并且拥有一个活跃的社区。

Vue 关于 Vue 简介，百度百科给出的解释是：Vue.js 是一套构建用户界面的渐进式框架。与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，并且非常容易学习，非常容易与其它库或已有项目整合。另一方面，Vue 完全有能力驱动采用单文件组件和 Vue 生态系统支持的库开发的复杂单页应用。

Vue.js 是一个提供了 MVVM 风格双向数据绑定的 Javascript 库（无依赖别的 js 库，直接引入一个 js 文件就可以使用，跟 jquery 差不多），专注于 View 层。它的核心是 MVVM 中的 VM，也就是 ViewModel。 ViewModel 负责连接 View 和 Model，保证视图和数据的一致性，这种轻量级的架构让前端开发更加高效、便捷。

1.4 参考资料

列出有关的参考资料：

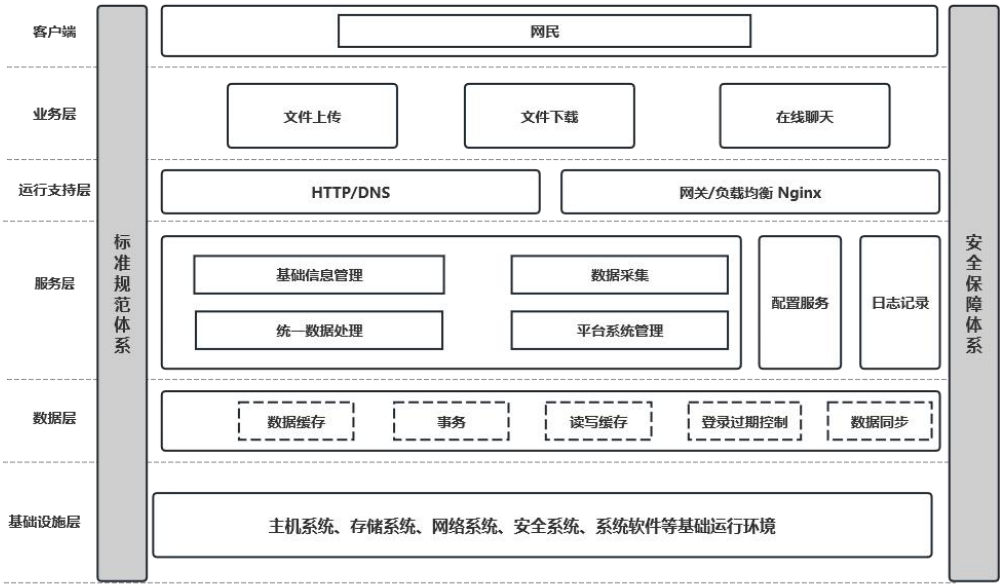
[1]《go 语言之路》，电子工业出版社出版，李文周

[2]Vue 官网：https://cn.vuejs.org/guide/quick-start.html

[3]go 语言教程：https://www.runoob.com/go/go-tutorial.html

[4]聊天室设计:https://golang2.eddycjy.com/posts/ch4/03-requirement-and-design/

2 程序系统的结构



3 功能设计说明

3.1 文件上传功能

3.1.1 功能描述

为用户提供在聊天室中上传文件（只允许 txt 和 pdf 格式）的能力，使其他参与者可以下载并查看这些文件。

3.1.2 界面设计

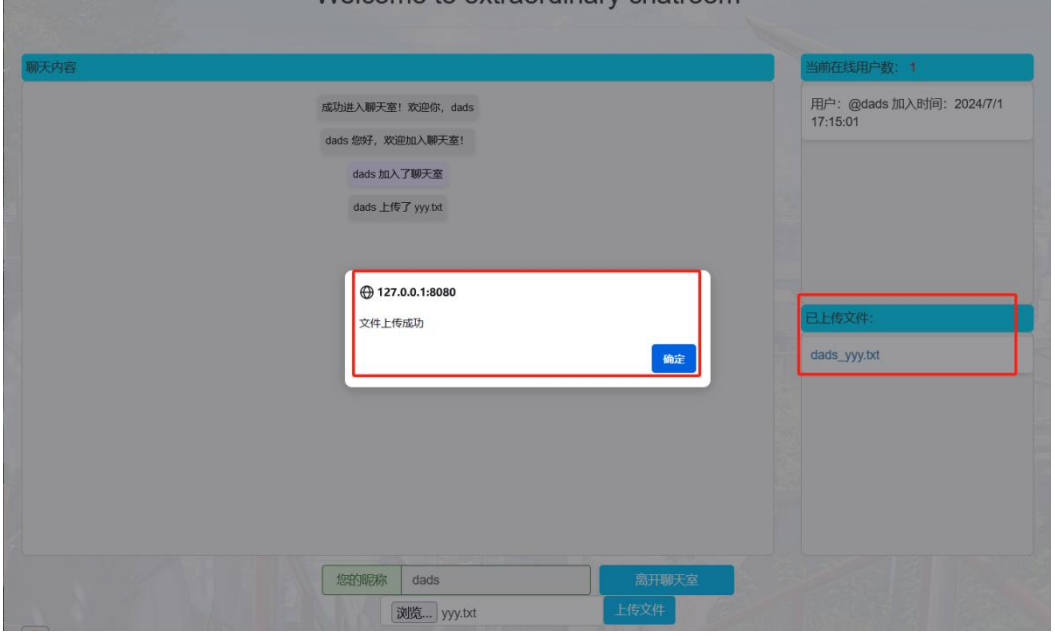
```
func handleUpload(w http.ResponseWriter, request *http.Request) {
    var token string
    //文件上传只允许 POST 方法
    if request.Method != http.MethodPost {
        w.WriteHeader(http.StatusMethodNotAllowed)
        _, _ = w.Write([]byte("Method not allowed"))
```

```
    return
}

//从表单中读取文件
file, fileHeader, err := request.FormFile("file")
authorizationHeader := request.Header.Get("Authorization")
if authorizationHeader != "" {
    // Split the header value by space
    parts := strings.Split(authorizationHeader, " ")
    if len(parts) == 2 && parts[0] == "Bearer" {
        token = parts[1]
        fmt.Println("token =" + token)
    } else {
        fmt.Println("Authorization header format must be: Bearer {token}")
    }
} else {
    fmt.Println("No Authorization header provided")
}
fmt.Println("token =" + token)
nickname, err := logic.ParseTokenAndValidate(token)
fmt.Println(nickname)
if err != nil {
    _, _ = io.WriteString(w, "Read file error")
    return
}
//defer 结束时关闭文件
defer file.Close()
newFilename := nickname + "_" + fileHeader.Filename
log.Println("filename: " + newFilename)

//创建文件
newFile, err := os.Create("../upload/" + newFilename)
if err != nil {
    _, _ = io.WriteString(w, "Create file error")
    return
}
//defer 结束时关闭文件
defer newFile.Close()

//将文件写到本地
_, err = io.Copy(newFile, file)
if err != nil {
    _, _ = io.WriteString(w, "Write file error")
    return
```

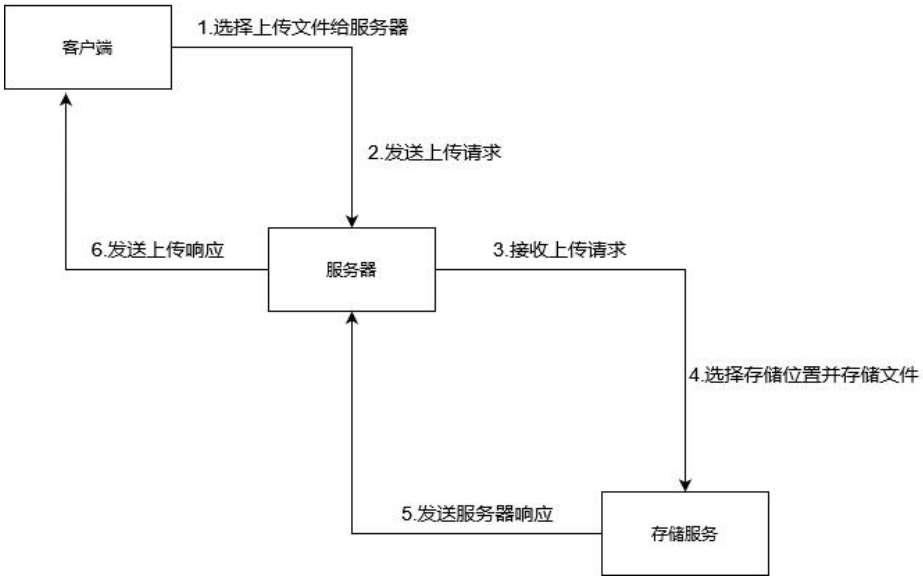



```
}
// msg := logic.NewFileUploadMessage(username, fileHeader.Filename)
// logic.Broadcaster.Broadcast(msg)
msg := logic.NewFileUploadMessage(nickname, fileHeader.Filename)
logic.Broadcaster.Broadcast(msg)

_, _ = io.WriteString(w, "Upload success")

}
```

3.1.3 流程逻辑



3.1.4 接口设计

数据名称	数据类型
fileslength	int
filesname	vchar
nickname	vchar
XMLHttpRequest	text

3.1.5 外部依赖

无

3.1.6 其它说明

限制上传文件类型：txt，pdf

3.2 文件下载功能

3.1.1 功能描述

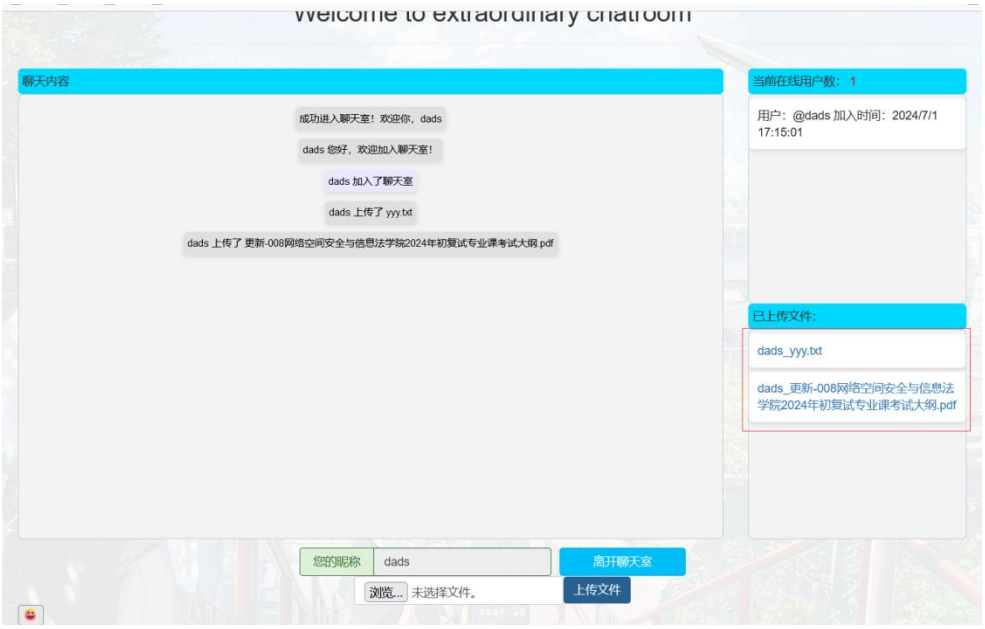
用户可以点击聊天室上传的文件进行下载，主要步骤：用户在聊天室右侧选择要下载的文件，并发送下载请求给服务器。服务器接收到下载请求后，确认用户权限和文件的有效性。服务器确定文件的位置并准备下载。服务器将文件内容通过网络传输到客户端。客户端接收文件内容，并根据下载进度显示下载状态。下载完成后，客户端保存文件到本地存储。

3.1.2 界面设计

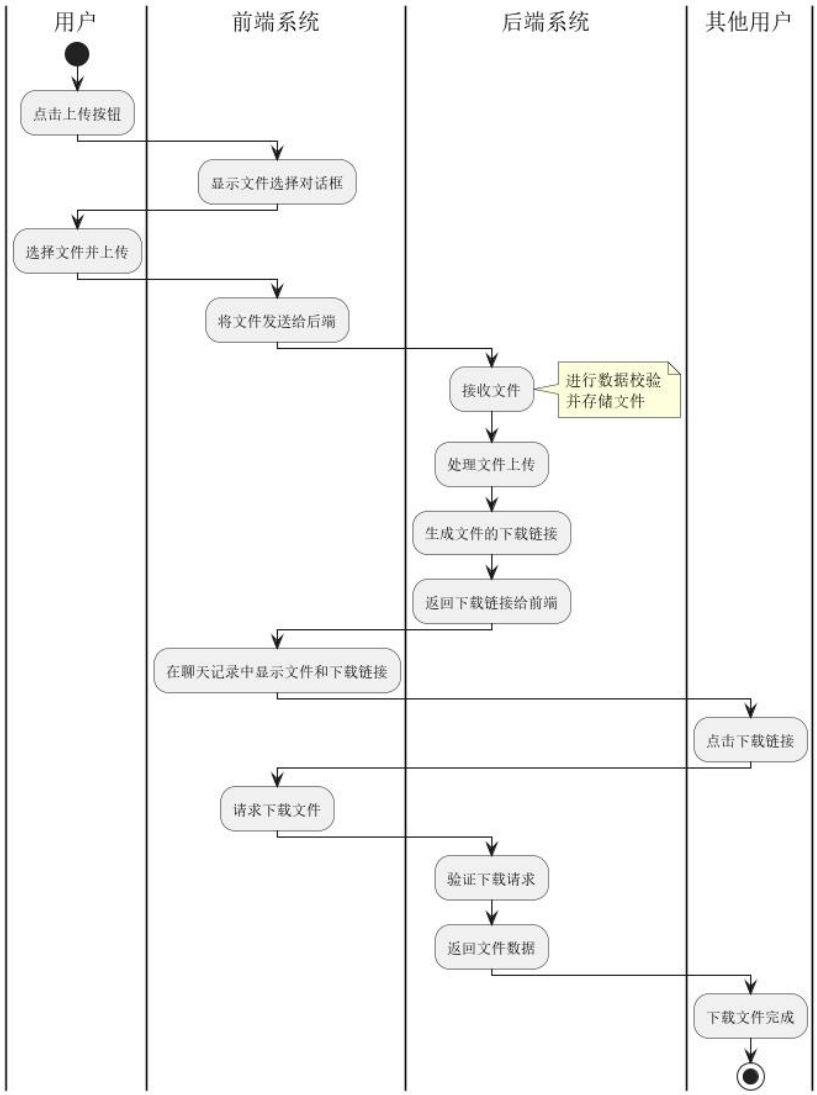
```
func handleDownload(w http.ResponseWriter, request *http.Request) {
    //文件下载只允许 GET 方法
    if request.Method != http.MethodGet {
        w.WriteHeader(http.StatusMethodNotAllowed)
        _, _ = w.Write([]byte("Method not allowed"))
        return
    }
    //文件名
    filename := request.FormValue("filename")
    if filename == "" {
        w.WriteHeader(http.StatusBadRequest)
        _, _ = io.WriteString(w, "Bad request")
        return
    }
    // 检查 filename 是否包含"../"，防止路径遍历攻击
    if strings.Contains(filename, "../") {
        w.WriteHeader(http.StatusForbidden)
        _, _ = io.WriteString(w, "Illegal access")
        return
    }
    log.Println("filename: " + filename)
    //打开文件
    file, err := os.Open("../upload/" + filename)
    if err != nil {
        w.WriteHeader(http.StatusBadRequest)
        _, _ = io.WriteString(w, "Bad request")
        return
    }
    //结束后关闭文件
    defer file.Close()

    //设置响应的 header 头
    w.Header().Add("Content-type", "application/octet-stream")
    w.Header().Add("content-disposition", "attachment; filename=\""+filename+"\"")
    //将文件写至 responseBody
    _, err = io.Copy(w, file)
```

```
    if err != nil {
        w.WriteHeader(http.StatusBadRequest)
        _, _ = io.WriteString(w, "Bad request")
        return
    }
}
```



3.1.3 流程逻辑



3.1.4 接口设计

数据名称	数据类型
fileslength	int
filesname	vchar
nickname	vchar
XMLHttpRequest	text

3.1.5 外部依赖

无

3.1.6 其它说明

限制下载文件类型：txt，pdf

3.3 聊天功能

3.1.1 功能描述

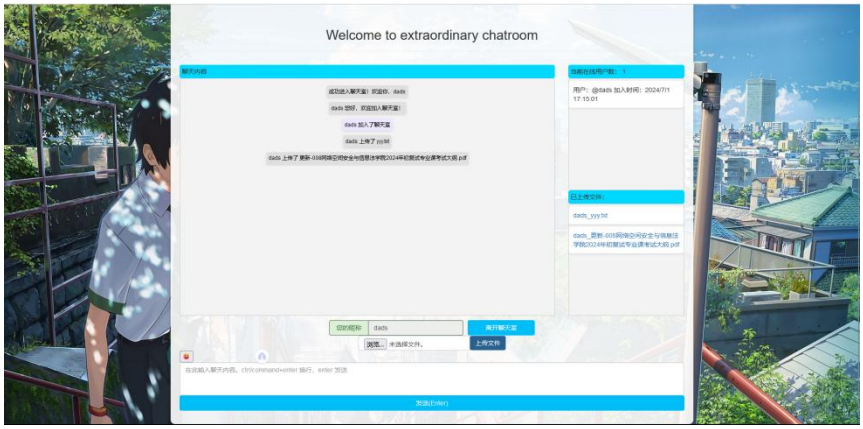
本聊天室能最多同时使 75 个人在线聊天，发送表情，显示在线用户数，实时显示时间和消息，进行匿名聊天。

3.1.2 界面设计

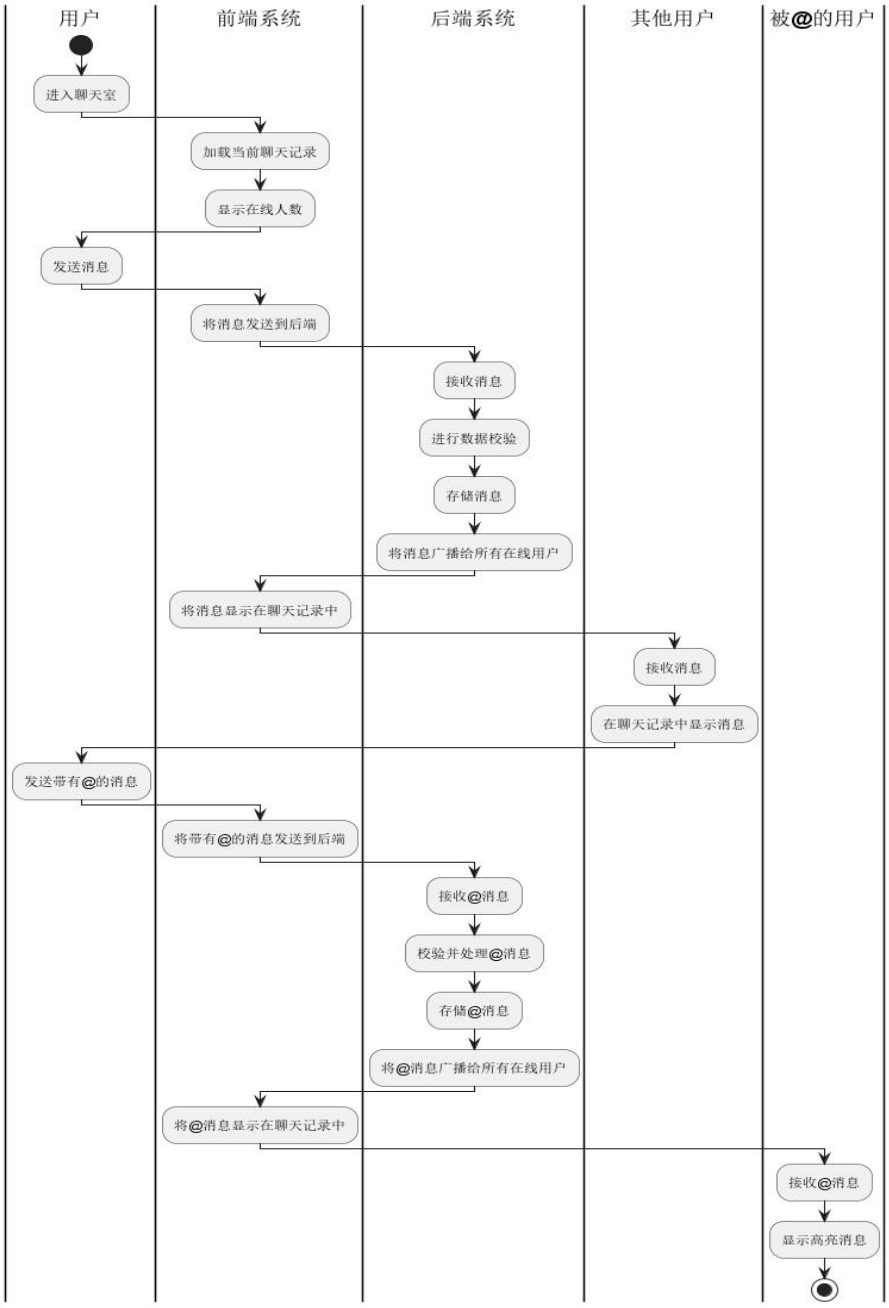
```
<body>
    <div class="container" id="app">
        <div class="row">
            <div class="col-md-12">
                <div class="page-header text-center">
                    <h2>Welcome to extraordinary chatroom</h2>
                </div>
            </div>
        </div>
        <div class="row d-flex align-items-stretch">
            <div class="col-md-9 equal-height">
                <div class="highlight">聊天内容</div>
                <div class="msg-list" id="msg-list">
                    <div class="message" v-for="msg in msglist" :class="{ system:
msg.type > 0, myself: msg.user.nickname == curUser.nickname }">
                        <div class="meta" v-if="msg.type == 0">
                            <span class="author">${ msg.user.nickname }</span> at
${ formatDate(msg.msg_time) } ${ calc(msg) }
                        </div>
                        <div>
                            <span class="content" style="white-space:
pre-wrap;">${ msg.content }</span>
                        </div>
                    </div>
                </div>
            </div>
            <div class="col-md-3 equal-height">
                <div class="row equal-height">
                    <div class="col-md-12">
                        <div class="highlight">当前在线用户数：
```

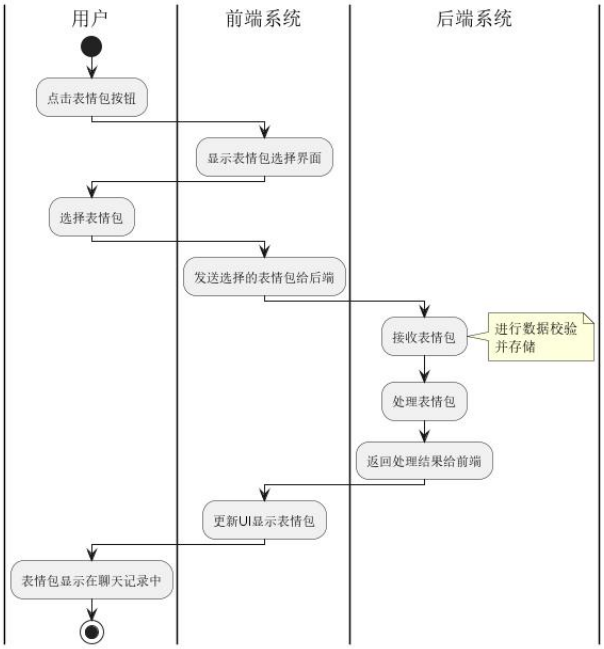
```
                        <font color="red">${ onlineUserNum }</font>
                    </div>
                    <div class="user-list">
                        <div class="user" v-for="user in users">
                            用户：@${ user.nickname } 加入时间：
${ formatDate(user.enter_at) }
                        </div>
                    </div>
                </div>
                <div class="col-md-12">
                    <div class="highlight">已上传文件：</div>
                    <div class="file-list">
                        <div class="file" v-for="filename in uploadedFiles">
                            <a :href="'/download?filename=' +
filename" :download="filename">${ filename }</a>
                        </div>
                    </div>
                </div>

            </div>
        </div>
    </div>
```



3.1.3 流程逻辑


用户
前端系统
后端系统
其他用户
被@的用户
进入聊天室
加载当前聊天记录
显示在线人数
发送消息
将消息发送到后端
接收消息
进行数据校验
存储消息
将消息广播给所有在线用户
将消息显示在聊天记录中
接收消息
在聊天记录中显示消息
发送带有@的消息
将带有@的消息发送到后端
接收@消息
校验并处理@消息
存储@消息
将@消息广播给所有在线用户
将@消息显示在聊天记录中
接收@消息
显示高亮消息



3.1.4 接口设计

数据名称	数据类型
UID	int
Nickname	string
EnterAt	time.Time
MessageChannel	chan *Message
Token	String
isNew	bool
conn	*websocket.Conn

3.1.5 外部依赖

```
<script type="module"
src="https://cdn.jsdelivr.net/npm/emoji-picker-element@^1/index.js"></script>
    <script>
        let emojiButton = document.getElementById('emoji-button');
        let emojiPicker = document.getElementById('emoji-picker');
        let chatContent = document.getElementById('chat-content');

        // 当点击按钮时，显示或隐藏 emoji 选择器
        emojiButton.addEventListener('click', function() {
            if (emojiPicker.style.display === 'none') {
                emojiPicker.style.display = 'block';
            } else {
                emojiPicker.style.display = 'none';
            }
        });

        // 当选择一个 emoji 时，将其添加到输入框中，并隐藏 emoji 选择器
        emojiPicker.addEventListener('emoji-click', function(event) {
            chatContent.value += event.detail.unicode;
            emojiPicker.style.display = 'none';
        });
</script>
```

3.1.6 其它说明

最大聊天人数：75

4 进度安排

已完成