湖南科技大学计算机科学与工程学院

综合实训 课程设计报告

|  |  |
| --- | --- |
| **专业班级：** | 计算机科学与技术六班 |
| **姓 名：** | 周俊哲 |
| **学 号：** | 2205010711 |
| **指导教师：** | 尹路修 |
| **时 间：** |  |
| **地 点：** |  |

|  |
| --- |
| 指导教师评语：    **成绩： 等级：**  **签名：**  **年 月 日** |

实验题目

**题目一 仿QQWeb即时聊天系统**

### 一．功能要求

* 实现Web的点对点即时的文本消息聊天功能。
* 实现Web的表情的发送、接收和显示功能。
* 实现Web的图片的发送、接收和显示功能。
* 实现本地消息的存储，在离线的时候也能加载和查看历史消息；
* 要求使用WebSocket；

### 二、实验目的

* 通过仿QQWeb即时聊天系统的开发，学生不仅可以掌握WebSocket及相关前端、后端技术，还能提升综合开发能力，了解完整的Web应用开发流程，同时学习解决实际开发中的各种技术挑战。这个项目能够帮助学生在理论知识和实践技能之间建立起有效的桥梁，为未来从事Web开发和实时通讯相关工作打下坚实的基础。

## 三、总体设计（含背景知识或基本原理与算法、或模块介绍、设 计步骤等）

### **1. 背景知识与基本原理**

WebSocket

WebSocket是一种在单个TCP连接上进行全双工通信的协议，旨在取代传统的HTTP请求-响应模型，提供更高效的实时通信方式。WebSocket协议由浏览器和服务器共同支持，在通信开始时通过HTTP协议握手，一旦连接建立，数据可以在客户端和服务器之间实时传输，无需重复握手过程。

**前后端分离**

前后端分离是现代Web开发的一种常见架构，将用户界面与业务逻辑、数据处理分离开来，提高代码的可维护性和开发效率。前端通常使用HTML、CSS和JavaScript构建，后端使用各种编程语言和框架（如Java、Node.js等）处理业务逻辑和数据库交互。

**2. 系统模块介绍**

整个即时聊天系统可以划分为以下几个主要模块：

* **用户管理模块**：
  + 负责用户注册、登录、身份验证。
  + 管理用户信息，如用户名、头像等。
* **消息传输模块**：
  + 通过WebSocket实现客户端和服务器之间的消息传输。
  + 支持文本消息、表情消息和图片消息的发送与接收。
* **消息存储模块**：
  + 负责将消息持久化存储到数据库中。
  + 提供消息的历史记录查询功能，支持离线消息查看。
* **前端界面模块**：
  + 提供用户友好的聊天界面，包含消息输入框、消息显示区域、表情选择器和图片上传功能。
  + 实现消息的实时更新和显示。

**3. 设计步骤**

**步骤一：需求分析**

* 确定功能需求：即时文本聊天、表情和图片发送接收、历史消息存储和离线查看等。
* 确定非功能需求：实时性、安全性、可扩展性和用户体验。

**步骤二：技术选型**

* 前端：HTML、CSS、JavaScript
* 后端：Java（Spring Boot）
* 数据库：MySQL
* 实时通信：WebSocket。

**步骤三：系统架构设计**

* **前端架构**：使用BootStrap
* **后端架构**：使用MVC架构，控制器处理WebSocket连接和消息传递，服务层处理业务逻辑，持久层与数据库交互。

**步骤四：数据库设计**

* 设计用户表、消息表，确定字段和索引。
* 用户表包含用户ID、用户名、密码、头像等信息。
* 消息表包含消息ID、发送者ID、接收者ID、消息内容、消息类型（文本、表情、图片）、时间戳等。

**步骤五：前端开发**

* 搭建基础框架，设计聊天界面。
* 实现WebSocket连接，处理消息发送和接收。
* 实现表情选择和图片上传功能。

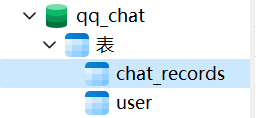
**步骤六：后端开发**

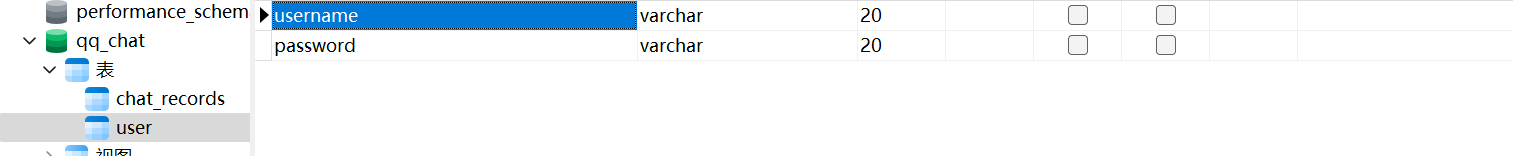
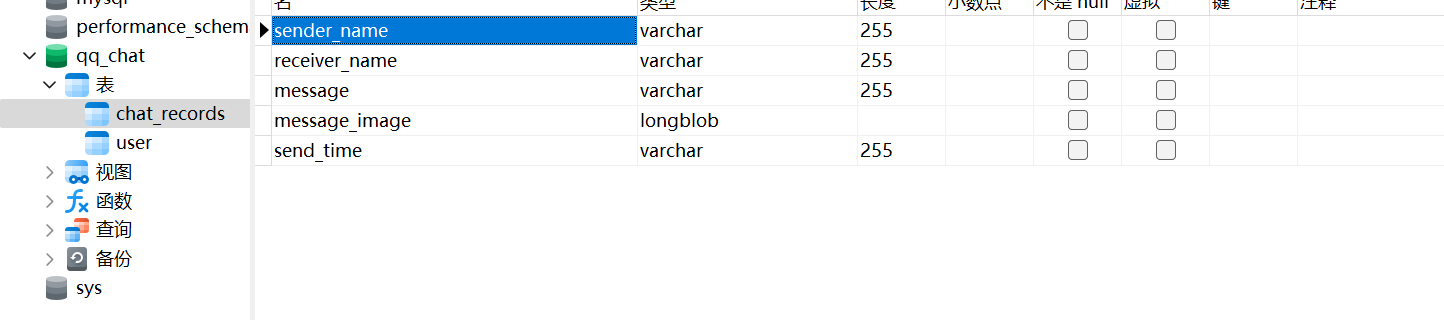
* 搭建WebSocket服务器，处理连接和消息传递。
* 实现用户管理功能，包含注册、登录和身份验证。
* 实现消息存储和历史消息查询功能。

# 四、详细设计（含主要的数据结构、程序流程图、关键代码等）

### 1.数据库设计

表设计结构如下:



表数据结构如下: 

消息表包含消息ID、发送者ID、接收者ID、消息内容、消息类型（文本、表情、图片）、时间戳等。

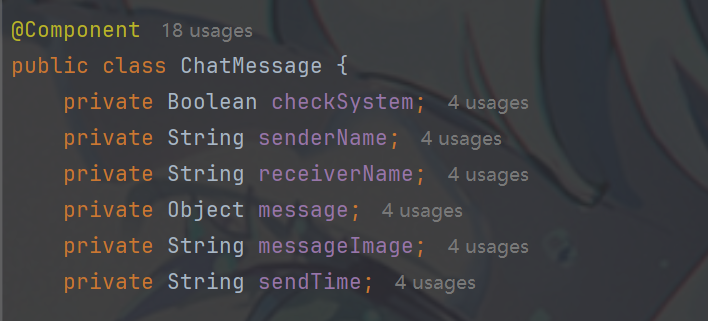
用户表包含用户ID、用户名、密码等信息。

配置信息如下:

spring:  
 *# TODO 配置数据源相关信息* datasource:  
 type: com.alibaba.druid.pool.DruidDataSource  
 driver-class-name: com.mysql.cj.jdbc.Driver  
 url: jdbc:mysql://localhost:3306/qq\_chat  
 username: root  
 password: 123456  
  
server:  
 port: 8080  
 address: localhost  
logging:  
 level:  
 root: info

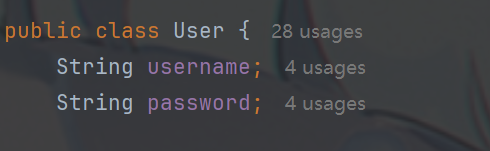
### 2.后端设计

实体类设计:

1.消息实体类:

2.消息记录实体类:

3.用户实体类:



控制器设计:

①消息控制器:

用于消息的获取与接收

* @GetMapping("/getChatMessages")  
  public Result getChatMessages(@RequestParam("username") String username, @RequestParam("toname") String toname){  
   List<ChatRecord> chatRecords=chatService.selectChat(username,toname);  
   List<ChatMessage> chatMessages=new ArrayList<>();  
   for (ChatRecord chatRecord : chatRecords) {  
   ChatMessage chatMessage = new ChatMessage();  
   chatMessage.setCheckSystem(false);  
   chatMessage.setSenderName(chatRecord.getSenderName());  
   chatMessage.setReceiverName(chatRecord.getReceiverName());  
   chatMessage.setMessage(chatRecord.getMessage());  
   chatMessage.setMessageImage(chatRecord.getMessageImage() != null ? Base64.*getEncoder*().encodeToString(chatRecord.getMessageImage()) : null);  
   chatMessage.setSendTime(chatRecord.getSendTime());  
   chatMessages.add(chatMessage);  
   }  
   result.setData(chatMessages);  
   result.setMsg("success");  
   return result;

②.页面控制器

用于实现跳转页面

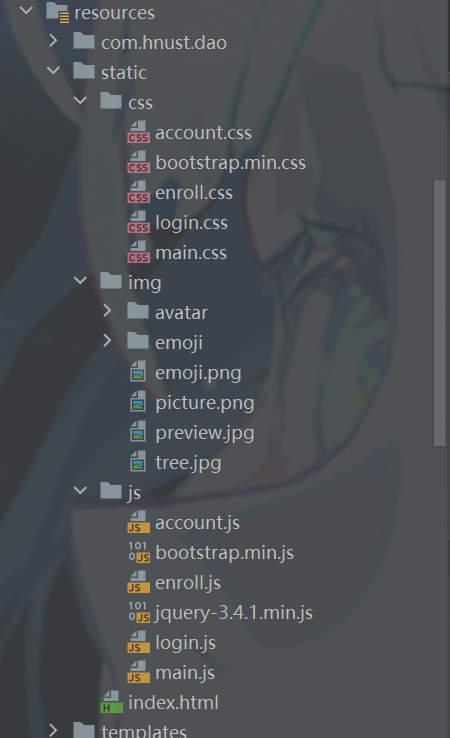
* @GetMapping("/chat/main\_page")  
  public String main()  
  {  
   *logger*.info("i'm coming------main");  
   return "main";  
  }

③.用户控制器

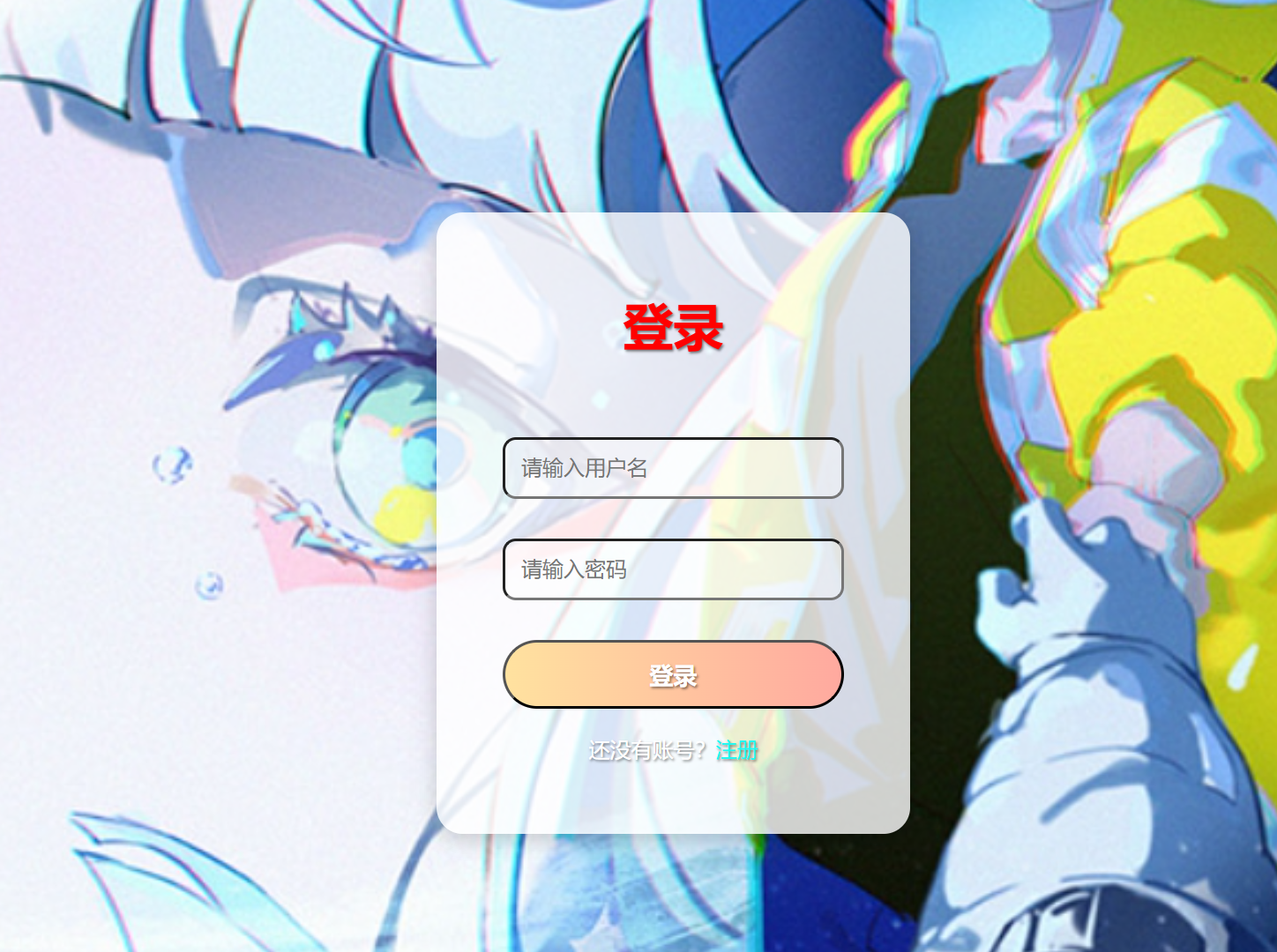
用于控制用户注册,删除,登录,获取用户信息,用户状态,用户密码,修改用户密码等

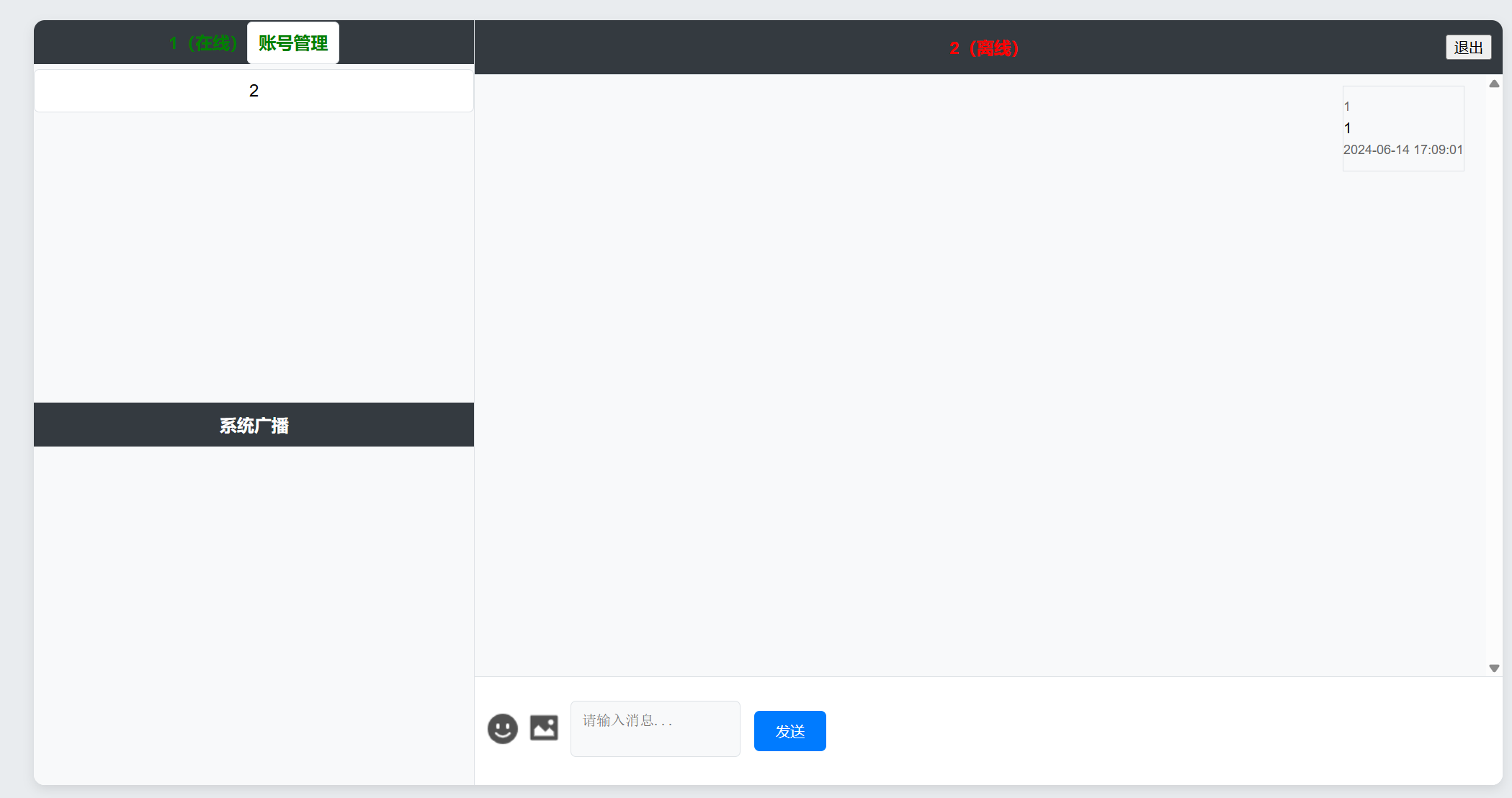
//登录逻辑处理  
@PostMapping("/login\_solve")  
public Result login\_solve(@RequestBody User user, HttpServletRequest request) {  
 User cur=userService.selectByUsername(user.getUsername());  
 // 判断账号和密码是否正确  
 if (cur!=null&&user.getPassword().equals(cur.getPassword())) {  
 if(ChatEndpoint.*checkSessionMap*(cur.getUsername()))  
 {  
 *logger*.info(user+"：登录失败--------用户名已经登录");  
 result.setMsg("failure");  
 result.setData(null);  
 }else{  
 HttpSession session = request.getSession();  
 ChatEndpoint.*closeBefore*(session);  
 session.setAttribute("user", cur.getUsername());  
 *logger*.info(cur+"：登录成功--------");  
 result.setMsg("success");  
 result.setData(cur);  
 }  
 } else {  
 *logger*.info(user+"：登录失败--------用户名或者密码错误");  
 result.setMsg("wrong");  
 result.setData(null);  
 }  
 return result;  
}

### 3.前端设计



登录页面:



输入正确的用户名和密码即可进入聊天主页面:  


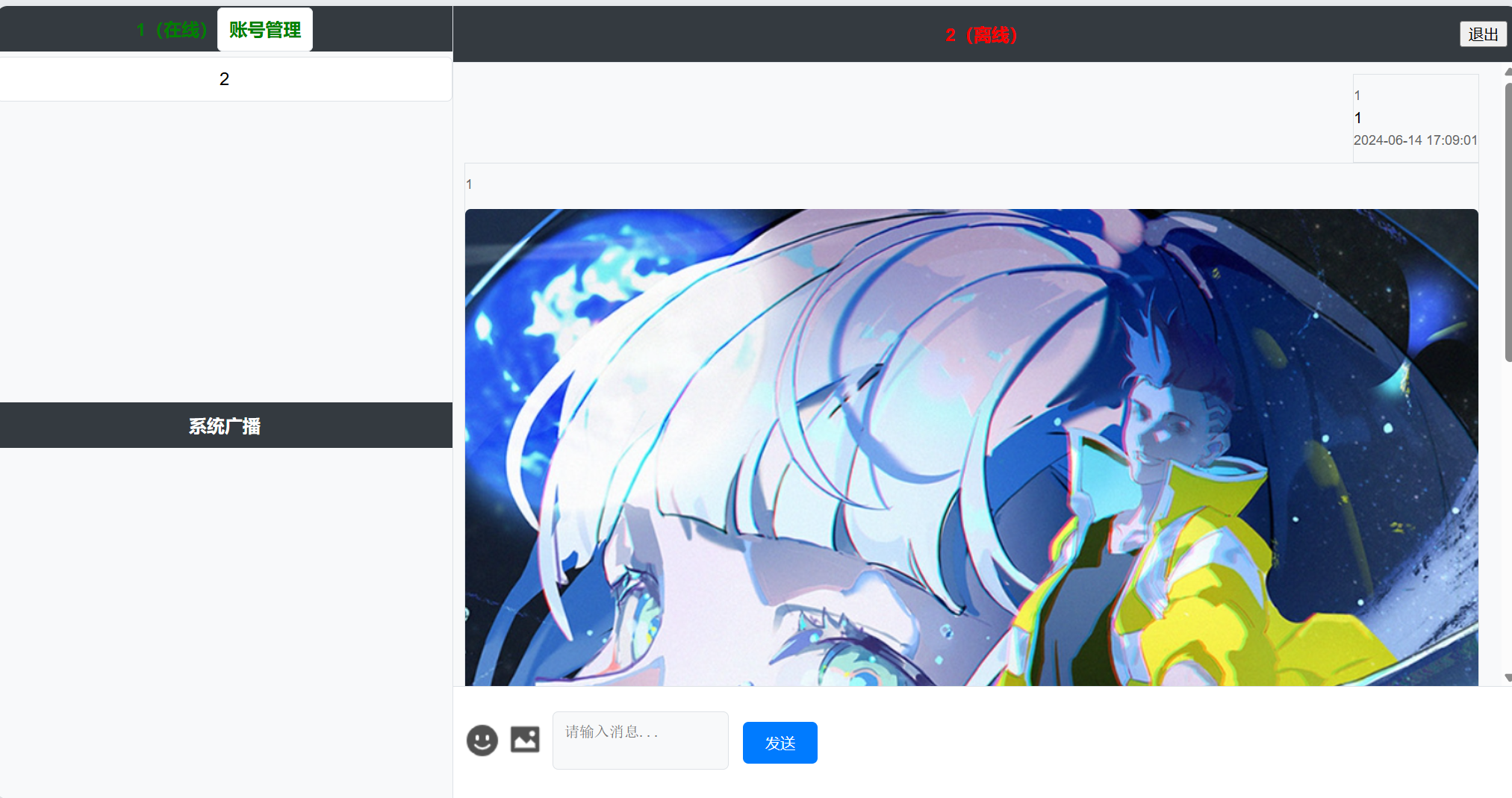
如果没有注册用户名和密码,可以点击注册按钮进入注册页面:  

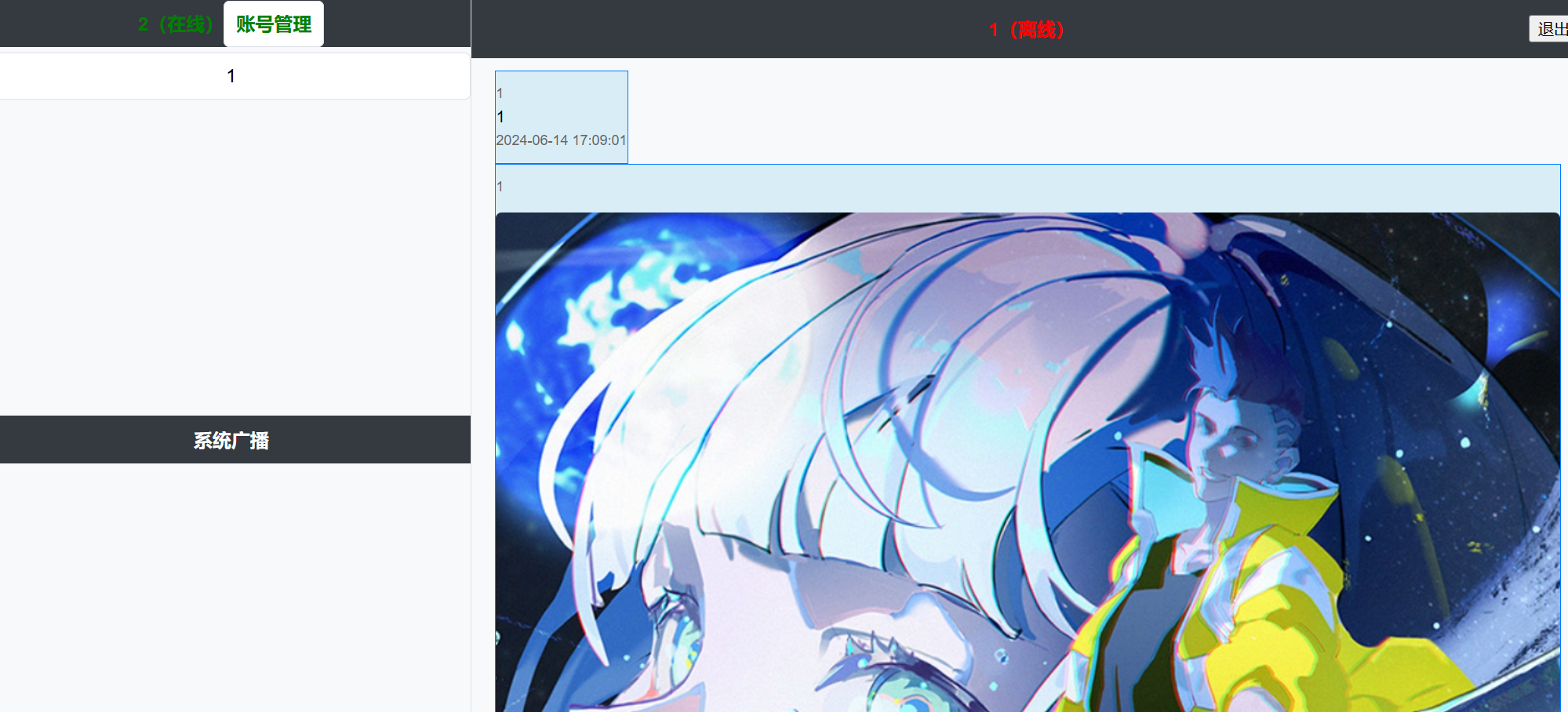

### 4.Websocket配置

* //该类为WebSocket的配置类，用于配置WebSocket的相关属性  
  @Configuration  
  public class WebSocketConfig {  
    
   @Bean  
   //注入ServerEndpointExporter bean对象，自动注册使用了@ServerEndpoint注解的bean  
   public ServerEndpointExporter serverEndpointExporter() {  
   return new ServerEndpointExporter();  
   }  
  }

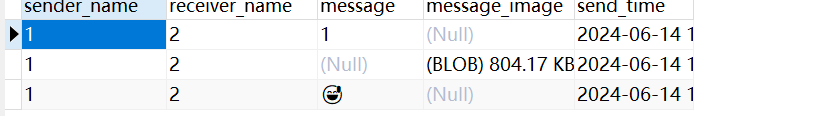
### 5.效果展示

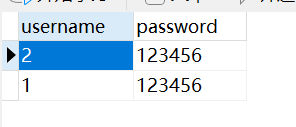
首先进入登录页面: 

输入账号密码并验证后进入聊天室,聊天室中可以发送文本,图片和表情等内容,且本地存储,之后进入页面仍能看到之前的内容消息能够被对方接收到,且对方也能发送数据被本人接收到



消息数据和用户数据全部保存在本地的数据库中,实现的数据的漫游处理





## 五、实验结果与分析

### 1. 消息传输的可靠性

**问题**: 确保消息在网络波动或服务器故障时能够可靠传输。

**解决方法**:

* 在客户端和服务器之间实现消息确认机制，确保消息被成功接收。
* 对未确认的消息进行重传，直到收到确认信息。

### 2. 多媒体内容处理

**问题**: 处理图片和表情等多媒体内容的传输和显示。

**解决方法**:

* 使用Base64编码将图片转换为字符串进行传输，客户端再进行解码和展示。

### 3. 本地消息存储与加载

**问题**: 离线时如何加载和查看历史消息。

**解决方法**:

* 离线时从本地存储加载历史消息，在线时从服务器获取最新消息。

### 4. 扩展性与可维护性

**问题**: 确保系统易于扩展和维护。

**解决方法**:

* 使用模块化设计，将系统分成独立的模块，如消息管理模块、用户管理模块等，方便维护和扩展。

## 六、小结与心得体会

在完成这个仿QQWeb即时聊天系统的实验过程中，我获得了一些宝贵的经验和心得体会：

1. **技术学习与实践**: 这个实验让我深入了解了WebSocket技术，并通过实践掌握了其在即时通讯中的应用。通过自己动手实现一个完整的聊天系统，我对WebSocket的原理、使用方法等有了更深入的理解。
2. **前后端交互**: 实现这个系统需要涉及到前端和后端的交互，这让我更加熟悉了前后端的分工与协作方式。前端负责界面展示和用户交互，后端负责数据处理和逻辑控制，两者紧密配合才能实现系统的功能。
3. **用户体验设计**: 设计一个用户友好的界面对于提升系统的使用体验至关重要。我学会了如何设计简洁清晰的界面，让用户能够方便地使用系统，提高了系统的可用性和用户满意度。
4. **错误调试与解决**: 在实现过程中遇到了各种各样的问题和错误，但通过不断地调试和排查，我学会了快速定位问题的方法，并找到了解决方案。这个过程提高了我的问题解决能力和技术水平。
5. **团队合作与沟通**: 如果是在团队中完成这个实验，我学会了团队合作和沟通的重要性。及时的沟通和有效的协作能够提高团队的效率和工作质量，使得项目顺利完成。

通过这个实验，我不仅掌握了新的技术和知识，还培养了解决问题的能力和团队合作精神。这些经验将对我的今后学习和工作都有很大的帮助。