TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP.HCM

**KHOA CÔNG NGHỆ THÔNG TIN**

BÁO CÁO ĐỒ ÁN HỌC PHẦN

**LẬP TRÌNH MẠNG MÁY TÍNH**

CHATSOCKET

Ngành: **Công Nghệ Thông Tin**

Giảng viên hướng dẫn: **ThS. Đoàn Khải Hiếu**

#### Sinh viên thực hiện:

1/ Đoàn Lê Hy Vọng

2/ Bùi Thành Đạt

3/ Lê Nguyễn Viết Dương

4/ Nguyễn Hữu Đức

MSSV: 2080600822

MSSV: 2011063152

MSSV: 2080600235

MSSV: 2080600272

#### Lớp: 20DTHD3 Lớp: 20DTHD3 Lớp: 20DTHD3

#### Lớp: 20DTHD3

##### TP. Hồ Chí Minh, ngày 1 tháng 1 năm 2024

# LỜI CAM ĐOAN

Nhóm chúng em xin cam đoan nội dung của đồ án học phần Lập Trình Mạng “ChatSocket” là sản phẩm của nhóm. Những vấn đề được trình bày trong bản báo cáo này chính là kết quả của quá trình học tập, tìm hiểu, nghiên cứu và làm việc của cả nhóm. Mọi tài liệu tham khảo có liên quan sẽ được trích dẫn đầy đủ và minh bạch ở phần TÀI LIỆU THAM KHẢO.

*TP. Hồ Chí Minh, ngày 1 tháng 1 năm 2024*

Người cam đoan

*Bùi Thành Đat*

*Đoàn Lê Hy Vọng*

*Lê Nguyễn Viết Dương*

*Nguyễn Hữu Đức*

MỤC LỤC

[**LỜI CAM ĐOAN 2**](#_Toc155132556)

[**DANH MỤC HÌNH ẢNH 4**](#_Toc155132557)

[**CHƯƠNG 1: TỔNG QUAN 5**](#_Toc155132558)

[1.1 LÝ DO CHỌN ĐỀ TÀI 5](#_Toc155132559)

[1.2 ĐỐI TƯỢNG NGHIÊN CỨU SỬ DỤNG 6](#_Toc155132560)

[1.3 MỤC TIÊU NGHIÊN CỨU 7](#_Toc155132561)

[1.4 PHẠM VI GIỚI HẠN 8](#_Toc155132562)

[**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 9**](#_Toc155132563)

[2.1 GIỚI THIỆU VỀ NGÔN NGỮ SỬ DỤNG 9](#_Toc155132564)

[2.2 CÔNG CỤ SỬ DỤNG 10](#_Toc155132565)

[**CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM 11**](#_Toc155132566)

[**CHƯƠNG 4: KẾT LUẬN VÀ KIẾN NGHỊ 15**](#_Toc155132567)

[4.1 KẾT LUẬN CHUNG 15](#_Toc155132568)

[4.1.1 Ưu Điểm 15](#_Toc155132569)

[4.1.2 Nhược Điểm 15](#_Toc155132570)

[4.2 HƯỚNG PHÁT TRIỂN 15](#_Toc155132571)

[**TÀI LIỆU THAM KHẢO 16**](#_Toc155132572)

# DANH MỤC HÌNH ẢNH

[*Hình 3. 1 Giao diện đăng nhập* 10](#_bookmark11)

[*Hình 3. 2 Giao diện đăng ký* 10](#_bookmark12)

[*Hình 3. 3 Giao diện phòng chat* 11](#_bookmark13)

[*Hình 3. 4 Giao diện gửi icon* 11](#_bookmark14)

[*Hình 3. 5 Giao diện thay đổi avatar* 12](#_bookmark15)

# CHƯƠNG 1: TỔNG QUAN

## LÝ DO CHỌN ĐỀ TÀI

Hiện nay, việc sử dụng WebChatSocket để truyền tải dữ liệu và tương tác trực tiếp với người dùng đang trở nên ngày càng phổ biến và được ứng dụng rộng rãi. Việc tìm hiểu về cách truyền tải dữ liệu qua WebChatSocket không chỉ mang lại hiểu biết thực tế mà còn mở ra nhiều cơ hội phát triển và ứng dụng trong nhiều lĩnh vực khác nhau vì đây là một hoạt động phổ biến và cần thiết trong môi trường truyền thông web hiện nay.

Nghiên cứu về WebChatSocket không chỉ giúp hiểu rõ về các giao thức liên quan như WebSocket, HTTP, mà còn tập trung vào việc xử lý bảo mật dữ liệu trong quá trình truyền tải. Bên cạnh đó, khả năng tích hợp vào nhiều ứng dụng và dự án khác nhau giúp tối ưu hóa hiệu suất và áp dụng cho nhiều ngành công nghiệp, từ thương mại điện tử đến dịch vụ tài chính và giáo dục trực tuyến.

Quan trọng hơn, việc nghiên cứu về truyền tải dữ liệu qua WebChatSocket không chỉ tập trung vào khía cạnh kỹ thuật mà còn mở ra những khía cạnh sâu rộng như quản lý lỗi, tối ưu hóa hiệu năng và đảm bảo tính tin cậy và ổn định của kết nối. Ngoài ra, có thể mở rộng đề tài này để bao gồm các chức năng như truyền tải file đa phương tiện, đồng bộ hóa dữ liệu và quản lý phiên làm việc, từ đó tạo ra những trải nghiệm tương tác trực tuyến tốt hơn và tăng cường khả năng tương tác giữa người dùng và ứng dụng web.

* 1. **ĐỐI TƯỢNG NGHIÊN CỨU SỬ DỤNG**

Đối tượng nghiên cứu của chúng ta tập trung vào việc sử dụng WebChatSocket và các người sử dụng cuối của nó. Cụ thể, đối tượng nghiên cứu bao gồm:

Người sử dụng cuối: Nghiên cứu sẽ xem xét cách người sử dụng cuối tương tác với WebChatSocket, bao gồm việc truyền tải tin nhắn và dữ liệu, xử lý sự kiện và tương tác với bot hoặc ứng dụng sử dụng WebChatSocket. Chúng tôi cũng sẽ tìm hiểu về trải nghiệm người dùng và đáp ứng các nhu cầu của họ trong việc sử dụng giao thức này.

Nhà phát triển: Nghiên cứu sẽ cân nhắc đến góc nhìn của nhà phát triển khi xây dựng và triển khai ứng dụng sử dụng WebChatSocket. Chúng ta sẽ tìm hiểu về việc tích hợp và sử dụng WebChatSocket trong các ứng dụng, xây dựng các tính năng và giao diện người dùng phù hợp, và tối ưu hóa hiệu suất và bảo mật của ứng dụng.

Quản trị viên hệ thống: Chúng ta sẽ tìm hiểu về trải nghiệm của quản trị viên hệ thống trong việc triển khai và duy trì các ứng dụng sử dụng WebChatSocket. Điều này có thể bao gồm quản lý bảo mật, giám sát hiệu suất, và xử lý sự cố liên quan đến WebChatSocket.

* + - **Sử dụng đối tượng nghiên cứu:**

Hiểu rõ hơn về cách người dùng cuối tương tác với WebChatSocket: Bằng việc nghiên cứu người dùng cuối, chúng ta có thể hiểu rõ hơn về nhu cầu, yêu cầu và thách thức mà họ đối mặt khi sử dụng WebChatSocket. Điều này giúp chúng ta tối ưu hóa giao diện người dùng, cung cấp các tính năng phù hợp và tạo ra trải nghiệm tốt hơn cho người dùng.

Đưa ra cải tiến và giải pháp mới: Dựa trên việc hiểu sâu về đối tượng nghiên cứu, chúng ta có thể đưa ra các cải tiến và giải pháp mới để cải thiện trải nghiệm người dùng, tăng cường khả năng tích hợp và mở rộng của WebChatSocket, và nâng cao hiệu suất và bảo mật của hệ thống.

Phát triển API dễ sử dụng và linh hoạt: Nghiên cứu đối tượng người sử dụng cuối và nhà phát triển giúp chúng ta xây dựng các API mạnh mẽ, dễ sử dụng và linh hoạt. Điều này tạo điều kiện thuận lợi cho việc tích hợp và phát triển ứng dụng sử dụng WebChatSocket bằng Node.js, giúp nhà phát triển tiết kiệm thời gian và công sức.

* 1. **MỤC TIÊU NGHIÊN CỨU**

Mục tiêu nghiên cứu của dự án "WebChatSocket" viết bằng Nodejs và Reactjs là phát triển và tối ưu hóa chức năng trò chuyện trực tuyến thông qua kết nối socket. Nghiên cứu tập trung vào việc triển khai một website hiệu quả và đáng tin cậy để truyền tải tin nhắn và dữ liệu giữa client và server trong môi trường mạng.

**Cụ thể, nghiên cứu sẽ tập trung vào các mục tiêu sau:**

- Xây dựng kết nối socket: Nghiên cứu sẽ triển khai kết nối socket sử dụng Node.js và thư viện Socket.io để tạo ra kết nối thời gian thực giữa client và server. Điều này cho phép truyền tải và nhận các tin nhắn giữa các thành viên trong phòng chat.

- Quản lý trạng thái và dữ liệu: Nghiên cứu sẽ xác định cách để quản lý trạng thái của các thành viên trong phòng chat, bao gồm việc xác định người dùng đang trực tuyến, gửi và nhận tin nhắn, và lưu trữ lịch sử tin nhắn. Đồng thời, nghiên cứu cũng sẽ tìm hiểu cách tối ưu hóa việc truyền tải dữ liệu và đảm bảo tính nhất quán và đáng tin cậy của dữ liệu trong môi trường mạng đa người dùng.

- Tích hợp giao diện người dùng: Nghiên cứu sẽ tập trung vào việc xây dựng giao diện người dùng sử dụng React.js để tạo ra trải nghiệm trực quan và tương tác cho người dùng. Giao diện sẽ cho phép người dùng gửi và nhận tin nhắn, tham gia vào các phòng chat, và tương tác với các thành viên khác trong phòng chat.

- Tích hợp tính năng bảo mật: Nghiên cứu sẽ tìm hiểu và triển khai các biện pháp bảo mật để đảm bảo tính riêng tư và an toàn của dữ liệu trong quá trình truyền tải và lưu trữ tin nhắn trong ứng dụng WebChatSocket.

* 1. **PHẠM VI GIỚI HẠN**
* Phạm Vi Dự Án:

Chức năng trò chuyện trực tuyến:

- Nghiên cứu tập trung vào triển khai chức năng trò chuyện trực tuyến giữa client và server thông qua kết nối socket.

- Xây dựng giao thức truyền tải tin nhắn và đảm bảo tính nhất quán và đáng tin cậy của dữ liệu trong môi trường mạng.

Kết nối socket sử dụng Node.js:

- Sử dụng Node.js và thư viện Socket.io để xây dựng kết nối socket thời gian thực giữa client và server.

- Đảm bảo tính ổn định, hiệu suất và khả năng mở rộng của kết nối socket trong môi trường mạng đa người dùng.

Giao diện người dùng sử dụng React.js:

- Tích hợp giao diện người dùng sử dụng React.js để tạo ra trải nghiệm trực quan và tương tác cho người dùng.

- Cho phép người dùng gửi và nhận tin nhắn, tham gia vào các phòng trò chuyện và tương tác với các thành viên khác trong phòng chat.

Bảo mật và xác thực:

- Tích hợp cơ chế xác thực để đảm bảo chỉ người dùng có quyền mới có thể truy cập và tham gia vào các phòng trò chuyện.

- Xây dựng lớp bảo mật để bảo vệ tính riêng tư và an toàn của dữ liệu trong quá trình truyền tải và lưu trữ tin nhắn.

Quản lý luồng và tải trọng hệ thống:

- Tối ưu hóa quản lý luồng để xử lý đồng thời nhiều kết nối socket từ nhiều client khác nhau.

- Đảm bảo khả năng mở rộng linh hoạt của hệ thống khi có nhiều yêu cầu truyền tải tin nhắn đồng thời.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

* 1. **GIỚI THIỆU VỀ NGÔN NGỮ SỬ DỤNG**
     + ***Giới thiệu về Nodejs***

Node.js: Node.js là một môi trường chạy mã JavaScript phía máy chủ, được xây dựng trên JavaScript Engine V8 của Chrome. Với Node.js, chúng ta có thể xây dựng các ứng dụng máy chủ hiệu quả, như các ứng dụng web thời gian thực, ứng dụng mạng xã hội, hay các ứng dụng IoT. Node.js hỗ trợ việc xử lý đồng thời và không đồng bộ, cho phép xử lý nhiều yêu cầu cùng lúc mà không làm chậm hiệu suất hệ thống.

* + - ***Giới thiệu về Reactjs***

React.js: React.js là một thư viện JavaScript phía người dùng (client-side) để xây dựng giao diện người dùng (UI) tương tác. React.js sử dụng cơ chế "component" để xây dựng UI theo cách modul và tái sử dụng dễ dàng. Với React.js, chúng ta có thể tạo ra các ứng dụng web động, linh hoạt và tương tác. React.js cung cấp một quy trình làm việc hiệu quả và hiệu suất cao, giúp tăng trải nghiệm người dùng và sự tương tác trên trang web.

***Ưu điểm của Node.js và React.js:***

* + - Hiệu suất cao: Cả Node.js và React.js được thiết kế để tối ưu hiệu suất. Node.js sử dụng kiến trúc không đồng bộ để xử lý đồng thời nhiều yêu cầu, trong khi React.js sử dụng cơ chế virtual DOM để chỉ cập nhật những phần thay đổi trong giao diện người dùng, làm tăng hiệu suất và tốc độ của ứng dụng.
    - Tính mở rộng và linh hoạt: Cả Node.js và React.js đều hỗ trợ mô hình phát triển mở rộng và linh hoạt. Node.js cho phép xây dựng các ứng dụng phức tạp và mở rộng dễ dàng nhờ vào hệ sinh thái các module phong phú. React.js giúp xây dựng giao diện người dùng tái sử dụng và dễ bảo trì, giúp phát triển ứng dụng một cách nhanh chóng và linh hoạt.
    - Cộng đồng và hỗ trợ: Cả Node.js và React.js đều có cộng đồng lớn và phong phú, với nhiều tài liệu, ví dụ và thư viện hỗ trợ. Chúng cũng được cập nhật và phát triển liên tục, giúp giải quyết các vấn đề và cung cấp các tính năng mới

***Nhược điểm của Node.js và React.js:***

* + - Khả năng quản lý tài nguyên: Node.js có thể tiêu tốn nhiều tài nguyên hệ thống hơn đối với các ứng dụng có lượng kết nối đồng thời lớn. React.js cũng có thể tạo ra một số overhead về tài nguyên nếu không được tối ưu hóa đúng cách.
    - Học và chuyển đổi: Node.js và React.js đòi hỏi một quá trình học tập và chuyển đổi từ các công nghệ khác. Điều này có thể tốn thời gian và công sức, đặc biệt đối với những ngườimới bắt đầu hoặc không quen thuộc với các ngôn ngữ này.

## CÔNG CỤ SỬ DỤNG

* + - ***Giới thiệu về Visual Studio Code (VS Code)***

Visual Studio Code (VS Code) là một trình biên tập mã nguồn mở và miễn phí, được phát triển bởi Microsoft. Nó hỗ trợ nhiều ngôn ngữ lập trình và cung cấp một loạt tính năng mạnh mẽ để phát triển ứng dụng.

* + - ***Ưu điểm nổi bật của Visual Studio Code (VS Code)***

Hỗ trợ nhiều ngôn ngữ lập trình: VS Code hỗ trợ nhiều ngôn ngữ lập trình phổ biến như JavaScript, TypeScript, Python, Java, C#, HTML, CSS và nhiều ngôn ngữ khác.

Cú pháp tô sáng và gợi ý mã: VS Code cung cấp tính năng cú pháp tô sáng (syntax highlighting) và gợi ý mã (code suggestions) giúp làm việc hiệu quả và giảm lỗi cú pháp.

Quản lý dự án linh hoạt: VS Code cho phép bạn làm việc với các dự án lớn hoặc nhỏ. Bạn có thể mở nhiều thư mục cùng một lúc và quản lý các tệp tin trong môi trường trực quan.

Tích hợp Git: VS Code tích hợp sẵn với Git, một hệ thống quản lý phiên bản phổ biến. Bạn có thể theo dõi lịch sử thay đổi, commit, push và pull trực tiếp từ giao diện của VS Code.

Đa nền tảng: VS Code có sẵn trên nhiều nền tảng, bao gồm Windows, macOS và Linux. Điều này cho phép bạn làm việc trên các hệ điều hành khác nhau mà không cần thay đổi công cụ.

Hỗ trợ mở rộng: VS Code có một hệ sinh thái phong phú của các tiện ích mở rộng (extensions) được phát triển bởi cộng đồng. Bạn có thể mở rộng tính năng của VS Code bằng cách cài đặt các tiện ích mở rộng từ Marketplace.

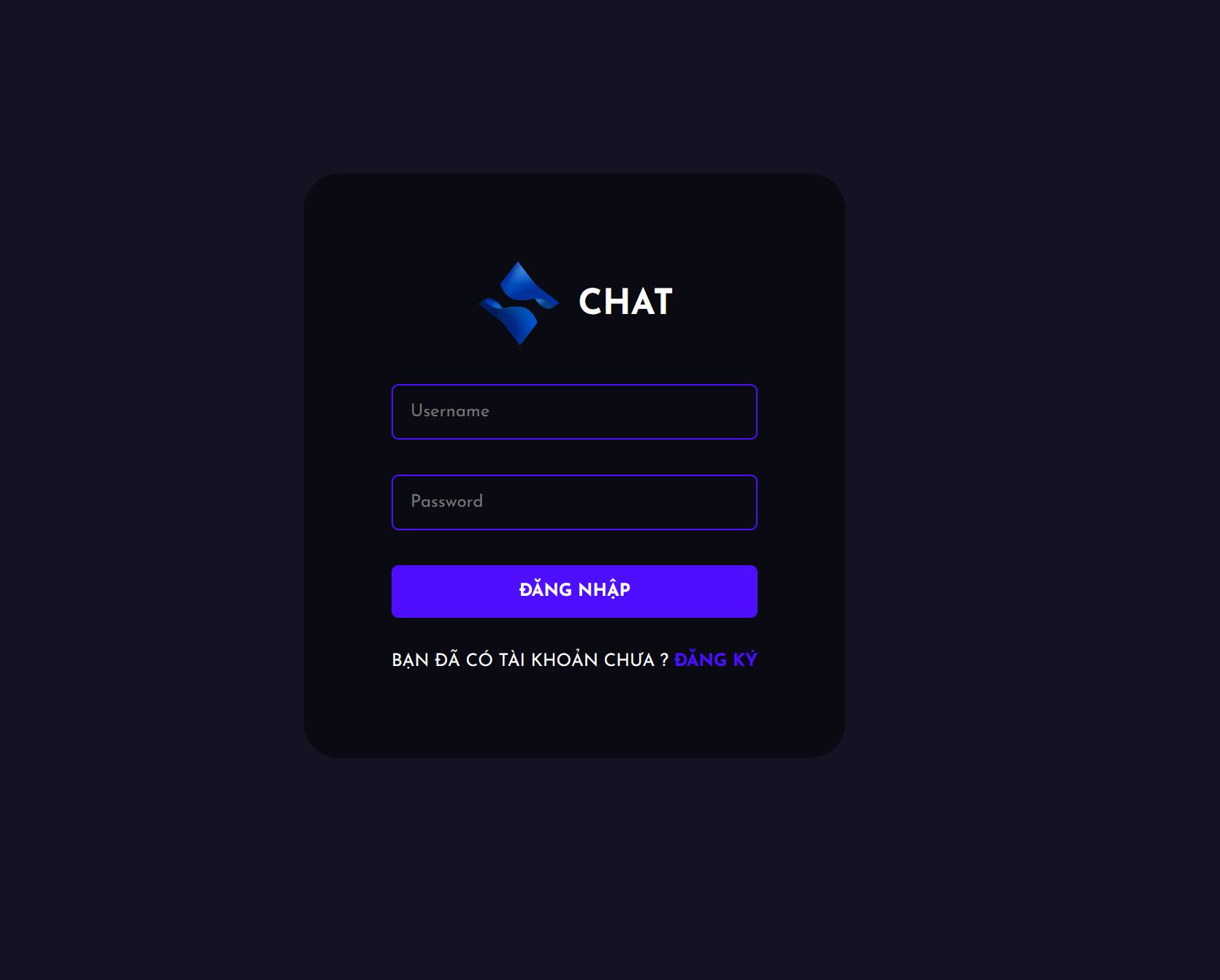
* + - ***Nhược điểm của Visual Studio Code (VS Code)***

Yêu cầu tài nguyên hệ thống: VS Code có thể yêu cầu một lượng tài nguyên hệ thống khá lớn, đặc biệt là khi bạn làm việc trên các dự án lớn hoặc sử dụng nhiều tiện ích mở rộng.

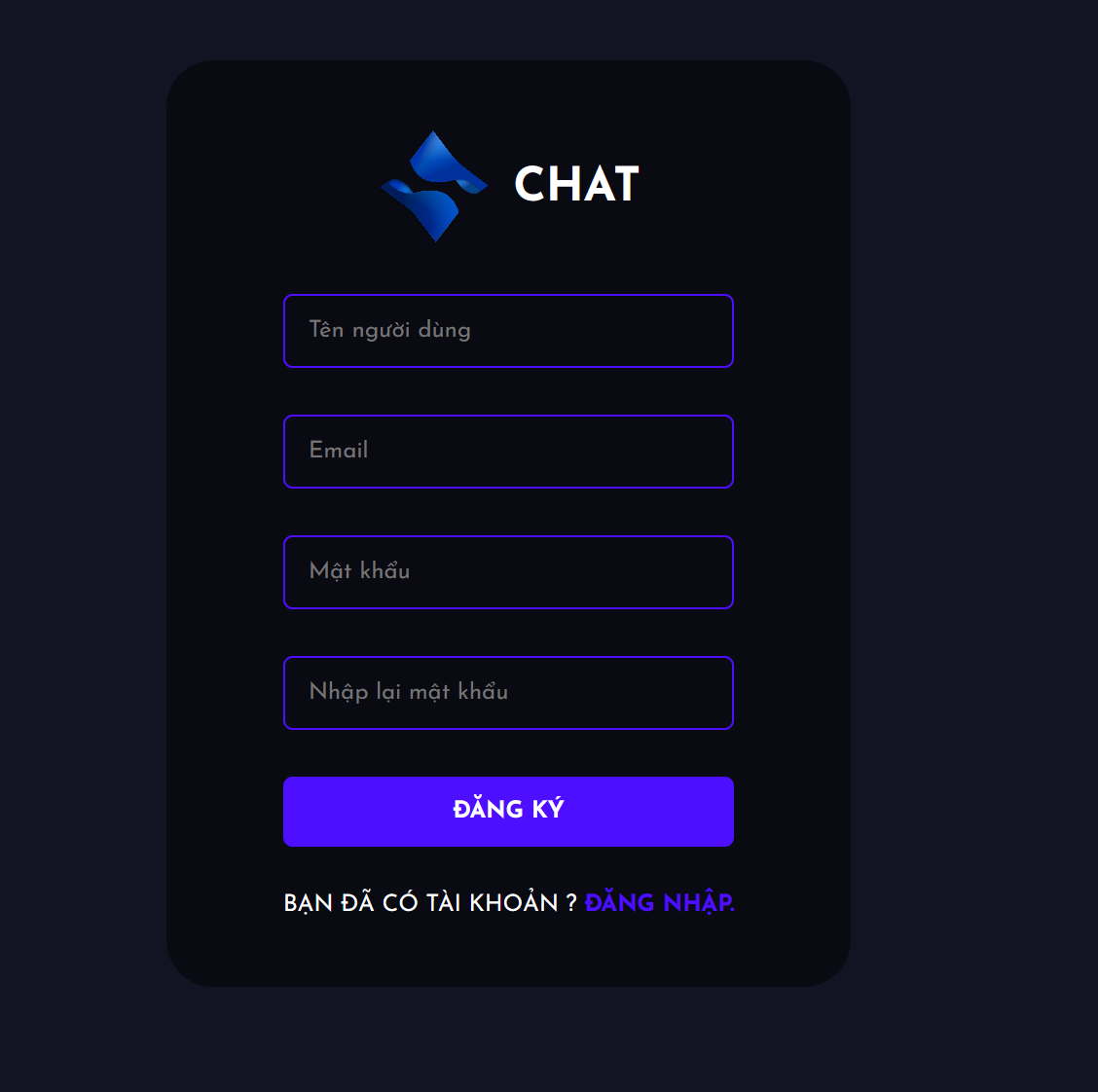
Khả năng mở rộng hạn chế: Mặc dù VS Code có hệ sinh thái mở rộng phong phú, nhưng so với một số IDE khác, nó có thể hạn chế trong việc mở rộng và tùy chỉnh.

Giao diện người dùng đơn giản: Một số người dùng có thể thấy giao diện người dùng của VS Code đơn giản hơn so với một số IDE khác..

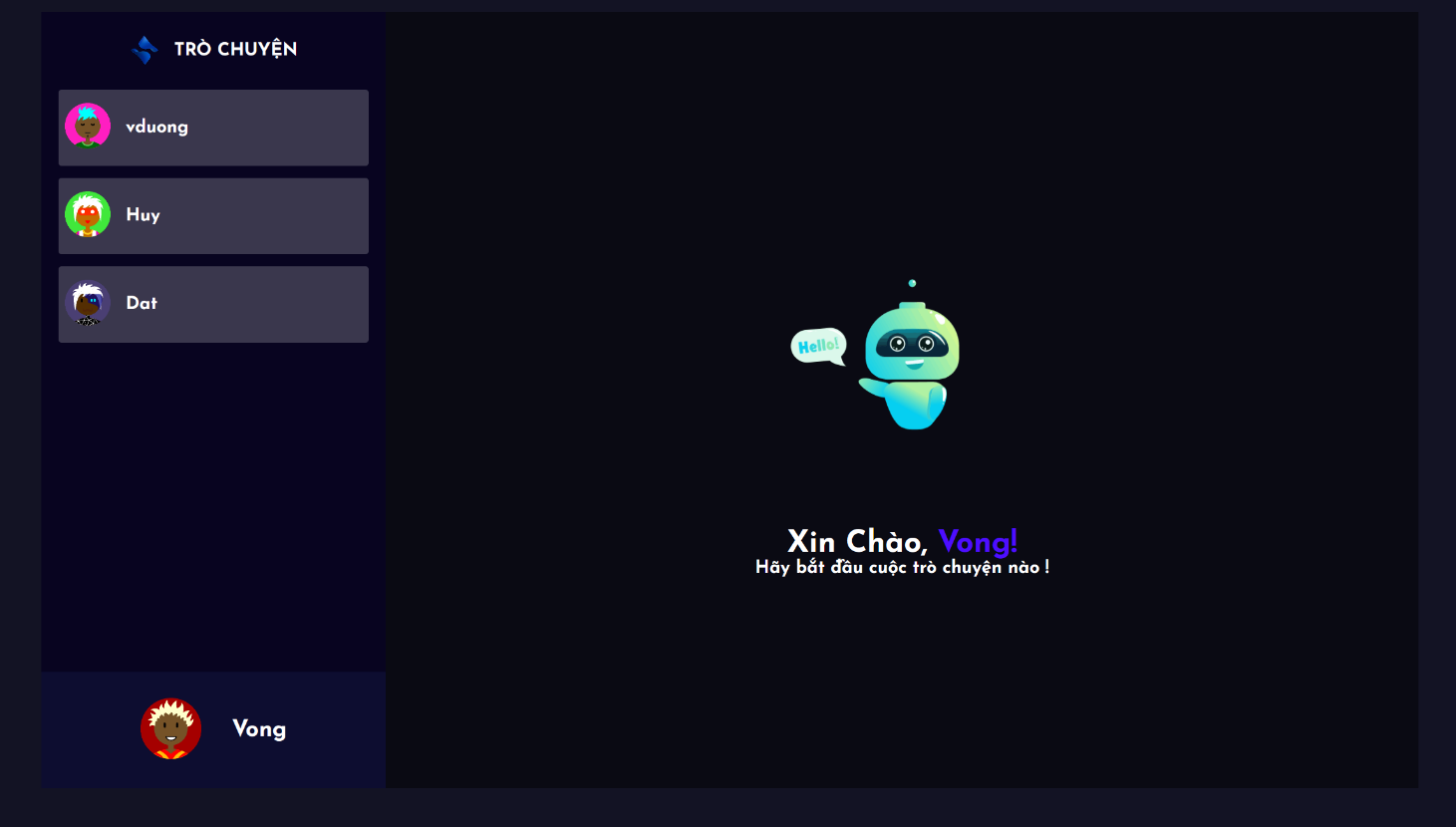
# CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM



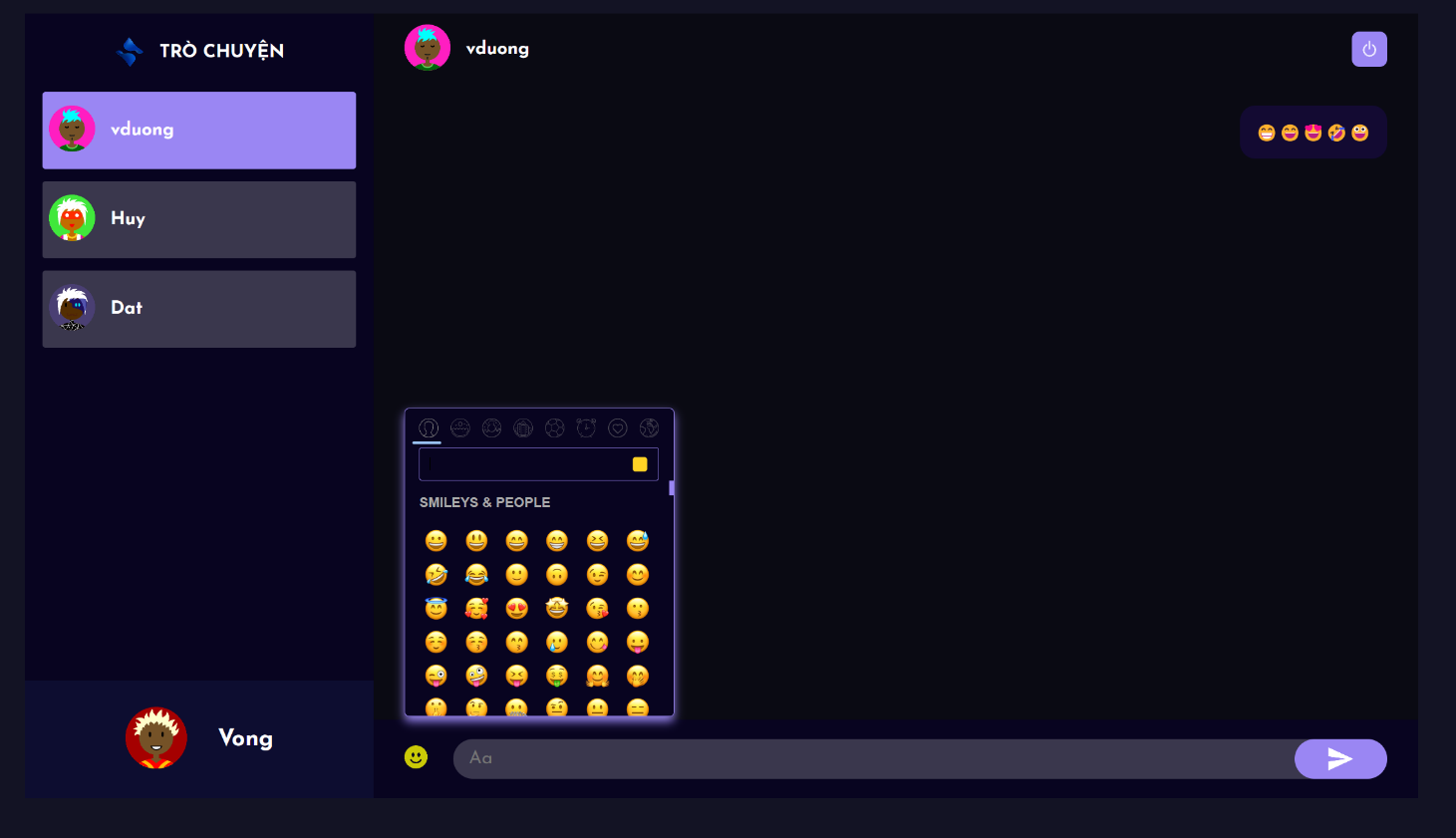
*Hình 3. 1 Giao diện đăng nhập*



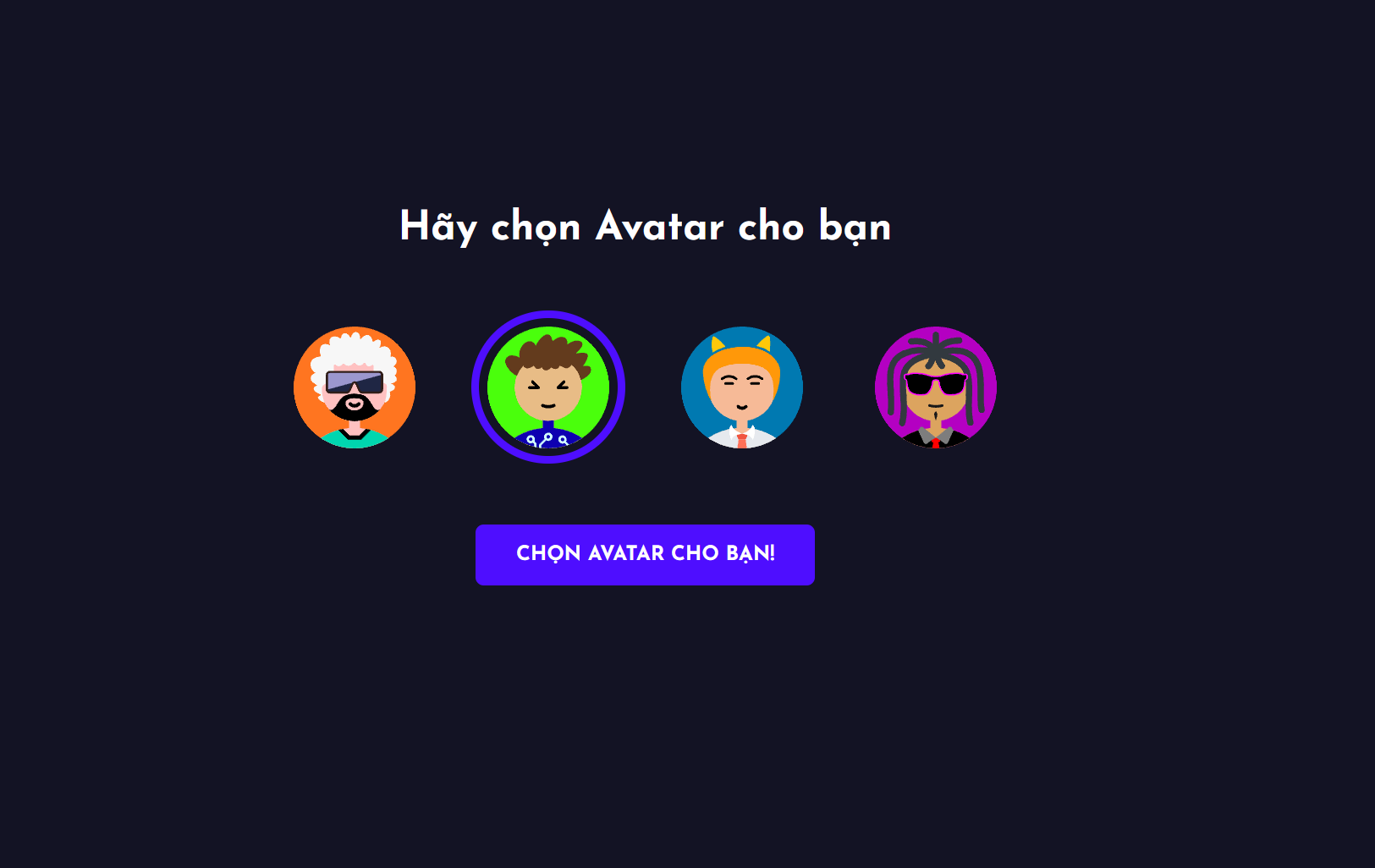
*Hình 3. 2 Giao diện đăng ký*



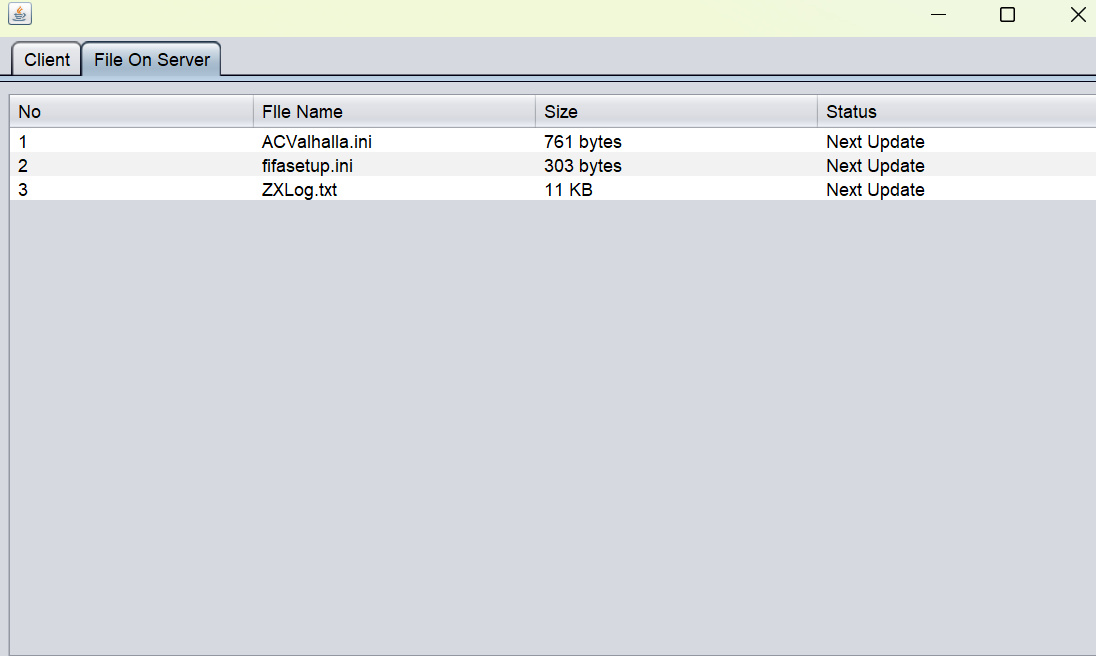
*Hình 3. 3 Giao diện phòng chat*



*Hình 3. 4 Giao diện gửi icon*



*Hình 3. 5 Giao diện thay đổi Avater*



*Hình 3. 6 Giao diện gửi file*

# CHƯƠNG 4: KẾT LUẬN VÀ KIẾN NGHỊ

* 1. **KẾT LUẬN CHUNG**

### Ưu Điểm

Hiệu suất: WebChatSocket xử lý và truyền tải dữ liệu một cách hiệu quả, đảm bảo trải nghiệm người dùng tốt.

Giao diện người dùng: Giao diện người dùng của WebChatSocket được thiết kế đơn giản và dễ sử dụng, tạo điều kiện thuận lợi cho người dùng tương tác và sử dụng dịch vụ..

### Nhược Điểm

Bảo mật: Để đảm bảo an toàn cho dữ liệu trong quá trình truyền tải, WebChatSocket có thể cần cải thiện mặt bảo mật của hệ thống. Các biện pháp bảo mật như mã hóa dữ liệu và xác thực người dùng có thể cần được triển khai để đảm bảo tính bảo mật cao.

Mở rộng: Với sự phát triển và lượng người dùng ngày càng tăng, WebChatSocket có thể cần cải thiện khả năng mở rộng để xử lý đồng thời lượng lớn người dùng. Việc tăng cường khả năng mở rộng giúp đảm bảo rằng hệ thống có thể xử lý tải trọng cao và đáp ứng nhu cầu người dùng một cách hiệu quả..

* 1. **HƯỚNG PHÁT TRIỂN**

Dựa trên những điểm trên, dưới đây là một số hướng phát triển để nâng cao chất lượng và khả năng của hệ thống Webchatsocket:

Bảo mật: Tăng cường cơ chế bảo mật là một yếu tố quan trọng để đảm bảo tính bảo mật cho dữ liệu trong quá trình truyền tải. Có thể triển khai các biện pháp bảo mật như mã hóa dữ liệu, sử dụng giao thức an toàn (như HTTPS) và xác thực người dùng để đảm bảo rằng dữ liệu không bị đánh cắp hoặc sửa đổi trái phép.

Mở rộng: Phát triển khả năng mở rộng của hệ thống là cần thiết để đáp ứng được số lượng người dùng lớn hơn và tải trọng cao hơn. Có thể sử dụng các giải pháp như tăng cường cấu trúc hệ thống, tối ưu hóa hiệu suất và sử dụng công nghệ phân tán để xử lý đồng thời các yêu cầu từ nhiều người dùng.

Tích hợp chức năng mới: Để nâng cao chức năng và trải nghiệm người dùng, có thể tích hợp thêm các chức năng mới. Ví dụ, có thể thêm tính năng chat nhóm, chia sẻ tệp tin, gửi tin nhắn thoại hoặc tích hợp các giao thức khác như WebSocket để cung cấp truyền dữ liệu realtime.

Tối ưu hóa hiệu suất: Đối với hệ thống webchatsocket, tối ưu hóa hiệu suất là một yếu tố quan trọng. Có thể tối ưu hóa mã nguồn, cải thiện cơ chế caching, giảm thiểu thời gian phản hồi và tăng cường khả năng xử lý để cung cấp trải nghiệm người dùng tốt hơn.

Giao diện người dùng: Tiếp tục nâng cấp giao diện người dùng để tăng tính thân thiện và dễ sử dụng. Cải thiện trải nghiệm người dùng, tối giản hóa giao diện và cung cấp các tính năng tương tác hấp dẫn có thể làm tăng sự hài lòng của người dùng.

# TÀI LIỆU THAM KHẢO

[1]. [https://viblo.asia/p/java-core-lesson-1-gioi-thieu-tong-quan-ve-java-core-](https://viblo.asia/p/java-core-lesson-1-gioi-thieu-tong-quan-ve-java-core-pgjLNd8d432) [pgjLNd8d432](https://viblo.asia/p/java-core-lesson-1-gioi-thieu-tong-quan-ve-java-core-pgjLNd8d432)

[2]. <https://vi.wikipedia.org/wiki/TCP/IP>