**HỌC VIỆN KỸ THUẬT QUÂN SỰ**

**HỌ VÀ TÊN: HỨA HOÀNG LINH**

**KHÓA: 13**

**HỆ ĐÀO TẠO: KỸ SƯ DÂN SỰ**

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**CHUYÊN NGÀNH: CÔNG NGHỆ PHẦN MỀM**

**ĐỀ TÀI: HỆ THỐNG TRÒ CHUYỆN TRỰC TUYẾN**

**ỨNG DỤNG CÔNG NGHỆ WEBRTC**

**HÀ NỘI, NĂM 2018**

|  |  |
| --- | --- |
| **HỌC VIỆN KỸ THUẬT QUÂN SỰ**  **KHOA: CÔNG NGHỆ THÔNG TIN**  **BỘ MÔN: CNPM** | **CỘNG HÒA XÃ HỘI CỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc** |

# **NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

Họ và tên: Hứa Hoàng LinhLớp: CNPM13 Khóa: 13

Ngành: Công nghệ thông tin Chuyên ngành: Công nghệ phần mềm

# **DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Từ viết tắt | Ý nghĩa |
| API | Application Progamming Interface |
| SQL | Structured Query Language |
| DB | Database |
| WebRTC | Web Real-time Communication |
| CSDL | Cơ sở dữ liệu |
| UC | Use case |
| FR | Functional Requirement |
| ID | Identification |
| DESC | Description |
| RAT | Rational |
| DEP | Dependency |
| WS | Web Socket |
| HTTP | Hyper Text Transfer Protocol |
| HTTPS | Hyper Text Transfer Protocol Secure |

# **DANH MỤC CÁC HÌNH ẢNH**

[Hình 2‑1 Các thành phần của Java 13](#_Toc7983605)

[Hình 2‑2 Luồng xử lý của Redux 20](#_Toc7983606)

[Hình 2‑3 Cách thức hoạt động của STUN 21](#_Toc7983607)

[Hình 3‑1 Mô hình Use Case 28](#_Toc7983608)

[Hình 4‑1 Mô hình kiến trúc 36](#_Toc7983609)

[Hình 4‑2 Package Diagram 37](#_Toc7983610)

[Hình 4‑3 Package core 38](#_Toc7983611)

[Hình 4‑4 Package apiserver 38](#_Toc7983612)

[Hình 4‑5 Package apiserver.handler 39](#_Toc7983613)

[Hình 4‑6 Package apiserver.clientrequest 39](#_Toc7983614)

[Hình 4‑7 Package apiserver.response 40](#_Toc7983615)

[Hình 4‑8 Package chatserver 40](#_Toc7983616)

[Hình 4‑9 Package chatserver.connection 41](#_Toc7983617)

[Hình 4‑10 Package chatserver.messageio 41](#_Toc7983618)

[Hình 4‑11 Package entities 42](#_Toc7983619)

[Hình 4‑12 Package session 42](#_Toc7983620)

[Hình 4‑13 Mô hình tuần tự khởi tạo server 43](#_Toc7983621)

[Hình 4‑14 Mô hình tuần tự tiếp nhận request từ client 44](#_Toc7983622)

[Hình 4‑15 Mô hình tuần tự tiếp nhận kết nối qua Web Socket 44](#_Toc7983623)

[Hình 4‑16 Mô hình tuần tự tiếp nhận tin nhắn qua socket 45](#_Toc7983624)

[Hình 4‑17 Mô hình tuần tự tiếp nhận sự kiện đóng kết nối 45](#_Toc7983625)

[Hình 4‑18 Mô hình tuần tự xử lý kết nối lỗi 46](#_Toc7983626)

[Hình 4‑19 Mô hình tuần tự nhận và gửi tin nhắn 46](#_Toc7983627)

[Hình 4‑20 Mô hình tuần tự của API Login 47](#_Toc7983628)

[Hình 4‑21 Mô hình tuần tự của API Register 48](#_Toc7983629)

[Hình 4‑22 Mô hình tuần tự của API GetUserInfo 49](#_Toc7983630)

[Hình 4‑23 Mô hình tuần tự của API ChangePassword 49](#_Toc7983631)

[Hình 4‑24 Mô hình tuần tự của API UpdateUserInfo 50](#_Toc7983632)

[Hình 5‑1 Mô hình sau khi triển khai hệ thống 53](#_Toc7983633)

[Hình 5‑2 Triền khai Server API 56](#_Toc7983634)

[Hình 5‑3 Triển khai Web Client 57](#_Toc7983635)

[Hình 5‑4 Giao diện trang đăng nhập 66](#_Toc7983636)

[Hình 5‑5 Giao diện trang đăng ký 66](#_Toc7983637)

[Hình 5‑6 Giao diện trang chủ 67](#_Toc7983638)

[Hình 5‑7 Giao diện thay đổi thông tin cá nhân 67](#_Toc7983639)

[Hình 5‑8 Giao diện thay đổi mật khẩu 68](#_Toc7983640)

[Hình 5‑9 Giao diện danh sách cuộc trò chuyện 68](#_Toc7983641)

[Hình 5‑10 Giao diện tìm kiếm bạn bè 69](#_Toc7983642)

[Hình 5‑11 Giao diện danh sách bạn bè 70](#_Toc7983643)

[Hình 5‑12 Giao diện tạo cuộc trò chuyện nhóm 70](#_Toc7983644)

[Hình 5‑13 Giao diện lịch sử trò chuyện 71](#_Toc7983645)

[Hình 5‑14 Giao diện thông tin cá nhân của bạn bè 71](#_Toc7983646)

[Hình 5‑15 Giao diện thông tin về cuộc trò chuyện nhóm 72](#_Toc7983647)

[Hình 5‑16 Giao diện cuộc gọi đến 72](#_Toc7983648)

[Hình 5‑17 Giao diện cuộc gọi video 73](#_Toc7983649)

# **DANH MỤC CÁC BẢNG BIỂU**

[Bảng 2‑1 So sánh giữa Jetty và Apache 13](#_Toc7961583)

[Bảng 2‑2 So sánh MongoDB và MySQL 16](#_Toc7961584)

[Bảng 3‑1 Đăng ký 28](#_Toc7961585)

[Bảng 3‑2 Đăng nhập 28](#_Toc7961586)

[Bảng 3‑3 Đăng xuất 28](#_Toc7961587)

[Bảng 3‑4 Lấy lại tài khoản 29](#_Toc7961588)

[Bảng 3‑5 Hiển thị thông tin người dùng 29](#_Toc7961589)

[Bảng 3‑6 Cập nhật thông tin người dùng 30](#_Toc7961590)

[Bảng 3‑7 Thay đổi mật khẩu 31](#_Toc7961591)

[Bảng 3‑8 Tìm kiếm bạn bè 31](#_Toc7961592)

[Bảng 3‑9 Thêm liên lạc 32](#_Toc7961593)

[Bảng 3‑10 Hiển thị danh sách liên lạc 32](#_Toc7961594)

[Bảng 3‑11 Xóa liên lạc 32](#_Toc7961595)

[Bảng 3‑12 Hiển thị tin nhắn gần nhất 32](#_Toc7961596)

[Bảng 3‑13 Hiển thị lịch sử tin nhắn 33](#_Toc7961597)

[Bảng 3‑14 Gửi tin nhắn văn bản 33](#_Toc7961598)

[Bảng 3‑15 Gửi tệp đính kèm 33](#_Toc7961599)

[Bảng 3‑16 Tạo cuộc trò chuyện nhóm 34](#_Toc7961600)

[Bảng 3‑17 Xóa cuộc trò chuyện 34](#_Toc7961601)

[Bảng 3‑18 Voice/Video call 34](#_Toc7961602)

[Bảng 4‑1 Collection user 50](#_Toc7961603)

[Bảng 4‑2 Collection contact 50](#_Toc7961604)

[Bảng 4‑3 Collection conversation 50](#_Toc7961605)

[Bảng 4‑4 Collection unread\_conversation 51](#_Toc7961606)

[Bảng 4‑5 Collection của database chatlog 51](#_Toc7961607)

# **MỤC LỤC**

[**NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP** 2](#_Toc7985978)

[**DANH MỤC TỪ VIẾT TẮT** 3](#_Toc7985979)

[**DANH MỤC CÁC HÌNH ẢNH** 4](#_Toc7985980)

[**DANH MỤC CÁC BẢNG BIỂU** 6](#_Toc7985981)

[**MỤC LỤC** 7](#_Toc7985982)

[**LỜI MỞ ĐẦU** 10](#_Toc7985983)

[**1.** **CHƯƠNG I: KHẢO SÁT ĐỀ TÀI** 11](#_Toc7985984)

[**I.** **Tổng quan về đề tài** 11](#_Toc7985985)

[**II.** **Mục đích của đề tài** 11](#_Toc7985986)

[**III.** **Các ứng dụng trong thực tế** 11](#_Toc7985987)

[**1.** **Skype** 11](#_Toc7985988)

[**2.** **Facebook Messenger** 13](#_Toc7985989)

[**IV.** **Kết luận chương** 13](#_Toc7985990)

[**2.** **CHƯƠNG II: CƠ SỞ LÝ THUYẾT** 14](#_Toc7985991)

[**I.** **Ngôn ngữ Java** 14](#_Toc7985992)

[**1.** **Java là gì?** 14](#_Toc7985993)

[**2.** **Ngôn ngữ Java** 14](#_Toc7985994)

[**3.** **Đặc điểm cơ bản của Java** 15](#_Toc7985995)

[**4.** **Các nền tảng cơ bản của Java** 15](#_Toc7985996)

[**5.** **Tại sao chọn Java để phát triền Server-side** 16](#_Toc7985997)

[**II.** **Embedded Jetty** 16](#_Toc7985998)

[**1.** **Embedded Jetty là gì?** 16](#_Toc7985999)

[**2.** **So sánh Jetty với Tomcat** 17](#_Toc7986000)

[**3.** **Ưu điểm của Jetty** 17](#_Toc7986001)

[**III.** **MongoDB** 17](#_Toc7986002)

[**1.** **Giới thiệu về MongoDB** 17](#_Toc7986003)

[**2.** **Ưu điểm và nhược điểm của MongoDB** 18](#_Toc7986004)

[**2.1.** **Ưu điểm** 18](#_Toc7986005)

[**2.2.** **Nhược điểm** 18](#_Toc7986006)

[**3.** **So sánh MongoDB với MySQL** 19](#_Toc7986007)

[**4.** **Tại sao chọn MongoDB?** 19](#_Toc7986008)

[**IV.** **ReactJS** 20](#_Toc7986009)

[**1.** **Giới thiệu về ReactJS** 20](#_Toc7986010)

[**2.** **Ưu điểm và nhược điểm của ReactJS** 20](#_Toc7986011)

[**2.1.** **Ưu điểm** 20](#_Toc7986012)

[**2.2.** **Nhược điểm** 21](#_Toc7986013)

[**3.** **Redux** 21](#_Toc7986014)

[**3.1.** **Giới thiệu về Redux** 21](#_Toc7986015)

[**3.2.** **Nguyên lý vận hành của Redux** 22](#_Toc7986016)

[**4.** **Tại sao lựa chọn ReactJS và Redux để phát triển Client-side** 23](#_Toc7986017)

[**V.** **WebRTC** 23](#_Toc7986018)

[**1.** **WebRTC là gì?** 23](#_Toc7986019)

[**2.** **STUN, TURN, ICE** 24](#_Toc7986020)

[**2.1.** **STUN** 24](#_Toc7986021)

[**2.2.** **TURN** 24](#_Toc7986022)

[**2.3.** **ICE** 25](#_Toc7986023)

[**3.** **Tại sao lại lựa chọn WebRTC** 25](#_Toc7986024)

[**VI.** **Nginx** 25](#_Toc7986025)

[**1.** **Giới thiệu về Nginx** 25](#_Toc7986026)

[**2.** **Tính năng của Nginx** 26](#_Toc7986027)

[**3.** **Tại sao lựa chọn Nginx?** 26](#_Toc7986028)

[**VII.** **Kết luận chương** 26](#_Toc7986029)

[**3.** **CHƯƠNG III: ĐẶC TẢ YÊU CẦU PHẦN MỀM** 27](#_Toc7986030)

[**I.** **Phân tích chức năng, nghiệp vụ** 27](#_Toc7986031)

[**1.** **Các chức năng chính** 27](#_Toc7986032)

[**2.** **Mô tả chi tiết yêu cầu chức năng** 27](#_Toc7986033)

[**II.** **Phân tích các Usecase** 31](#_Toc7986034)

[**1.** **Mô hình Use Case tổng thể** 31](#_Toc7986035)

[**2.** **Đặc tả chi tiết các Use Case** 31](#_Toc7986036)

[**4.** **CHƯƠNG VI: THIẾT KẾ HỆ THỐNG** 41](#_Toc7986037)

[**I.** **Mô hình kiến trúc** 41](#_Toc7986038)

[**1.** **Deployment Diagram** 41](#_Toc7986039)

[**2.** **Package Diagram** 41](#_Toc7986040)

[**3.** **Class Diagram** 42](#_Toc7986041)

[**II.** **Mô hình tuần tự** 47](#_Toc7986042)

[**1.** **Mô hình tuần tự của Server** 48](#_Toc7986043)

[**2.** **Mô hình tuần tự của API** 52](#_Toc7986044)

[**1.** **Thiết kế CSDL** 55](#_Toc7986045)

[**1.** **Database user** 55](#_Toc7986046)

[**2.** **Database chatlog** 57](#_Toc7986047)

[**5.** **CHƯƠNG V: XÂY DỰNG HỆ THỐNG** 58](#_Toc7986048)

[**I.** **Cài đặt môi trường** 59](#_Toc7986049)

[**1.** **Cài đặt các thư viện cơ bản và cần thiết:** 59](#_Toc7986050)

[**2.** **Cài đặt Java** 60](#_Toc7986051)

[**3.** **Cài đặt Nodejs và NPM** 61](#_Toc7986052)

[**4.** **Cài đặt MongoDB** 61](#_Toc7986053)

[**5.** **Cài đặt Nginx** 61](#_Toc7986054)

[**II.** **Triển khai hệ thống** 62](#_Toc7986055)

[**1.** **Triển khai Server API** 62](#_Toc7986056)

[**2.** **Triển khai Web Client** 63](#_Toc7986057)

[**3.** **Cài đặt WebRTC Server** 64](#_Toc7986058)

[**4.** **Cấu hình Nginx** 65](#_Toc7986059)

[**III.** **Giới thiệu về hệ thống** 72](#_Toc7986060)

[**TÀI LIỆU THAM KHẢO** 80](#_Toc7986061)

# **LỜI MỞ ĐẦU**

# **CHƯƠNG I: KHẢO SÁT ĐỀ TÀI**

1. **Tổng quan về đề tài**

Giao tiếp là một trong những hoạt động thiết yếu trong cuộc sống của con người. Sự giao tiếp đã phát triền từ hàng thế kỷ trước, khi con người vẫn còn dùng thư và người đưa thư để gửi đi những thông tin quan trọng. Ngày nay việc giao tiếp đã đơn giản hơn khi tin nhắn điện tử ra đời.

Xã hội ngày càng phát triền thì nhu cầu giao tiếp giữa còn người và con người cần thiết hơn bao giờ hết. Giao tiếp là một trong những hoạt động thiết yếu nhất. Từ chuyện công việc tới chuyện cá nhân hay những cuộc trò chuyện tán gẫu bình thường. Tuy nhiên, sử dụng tin nhắn SMS gặp phải một số hạn chế như tốn chi phí cho từng tin nhắn. Việc phủ sóng mạng 4G, sự phổ biến của wifi và cả sự phát triển của công nghệ làm cho những ứng dụng nhắn tin trực tuyến trở nên hữu ích hơn bao giờ hết.

Với ứng dụng nhắn tin trực tuyến, người dùng không phải tốn chi phí cho từng tin nhắn. Chưa kể những tính năng mới, tiện ích được phát triển để phù hợp với nhu cầu của người dùng.

1. **Mục đích của đề tài**

Hệ thống được xây dựng với mục đích giúp người dùng có thể giao tiếp được với nhau thông qua Internet. Các phương thức giao tiếp bao gồm: tin nhắn văn bản, tin nhắn tệp, voice/video call. Không những vậy, hệ thống cũng được thiết kế với giao diện đơn giản mà vẫn đầy đủ tính năng, thân thiện ngay cả với người sử dụng lần đầu tiên.

Hệ thống cũng được xây dựng với công nghệ và những kiến trúc tối ưu nhất để có thể vận hành được với một số lượng người dùng khổng lồ và dễ dàng mở rộng cũng như bảo trì.

Ngoài ra, cũng là kết quả của quá trình tự học và tự nghiên cứu. Tạo nền tảng tốt hơn cho việc tự nghiên cứu cũng như tìm hiều các công nghệ khác sau này.

1. **Các ứng dụng trong thực tế**
2. **Skype**
   1. **Giới thiệu**

Skype là dịch vụ gọi điện miễn phí trên nền IP (Voice Over IP) – được phát hành lần đầu tiên vào năm 2003 bởi sự hợp tác của các thành viên từ nhiều quốc gia (Đan Mạch, Thụy Điển…). Skype đã có những bước phát triển mạnh mẽ, sau 7 năm phát hành, đến năm 2010, đã có gần 7 trăm triệu người sử dụng. Năm 2011, Microsoft mua lại Skype với giá 8.5 tỉ USD và hiện Skype vẫn đang thuộc sở hữu của Microsoft.

Hiện nay, Skype đang dần được tích hợp với các dịch vụ khác của Microsoft (bạn có thể đăng nhập Skype bằng tài khoàn Outlook). Chức năng cơ bản của Skype là chat (instant messaging), free call, chia sẻ màn hình… Bạn cũng có thể sử dụng Skype trên các smartphone bằng cách tải ứng dụng Skype từ kho ứng dụng (ví dụ, Goolge Play nếu bạn dùng smartphone Android)

* 1. **Ưu điểm/Nhược điểm**

Ưu điểm:

* Đơn giản, dễ sử dụng
* Hỗ trợ nhiều nền tảng. Hiện nay có rất nhiều ứng dụng gọi điện, nhắn tin miễn phí như Viber, Facebook, Yahoo Messenger,… nhưng dưới góc độ người dùng trong các doanh nghiệp thì Skype vẫn được sử dụng phổ biến hơn cả. Skype có lúc là công cụ chính thống để trao đổi công việc trong các doanh nghiệp, trong khi các ứng dụng gọi điện nhắn tin miễn phí trên mobile chỉ mang tính chất giải trí, cá nhân.
* Nhiều tính năng hay, thiết thực: Quản lý nhóm rất tốt, quản lý lịch sử chat (đặc biệt là tính năng cho phép sửa/xóa nội dung đã gửi),gửi nhận file rất tốt (nhanh – do tính năng tự động nhận diện mạng nội bộ, nếu bạn gửi file qua Skype từ các máy trong cùng mạng LAN thì tốc độ không kém gì khi bạn copy file giữa các máy bằng cách thông thường trong Windows Explorer)

Nhược điểm:

* không hỗ trợ tốt việc gửi tin nhắn offline. Ví dụ, A gửi cho B một tin nhắn trong khi B đang offline, rồi sau đó A cũng offline => thì khi B online trở lại sẽ không nhận được ngay tin nhắn của A mà phải chờ đến khi A và B cùng online.
  1. **Một số tính năng nổi bật**
* Kết nối mọi người
* Gửi tin nhắn
* Voice/Video call
* Gửi tệp
* Trò chuyện nhóm
* Livestream trong nhóm

1. **Facebook Messenger**
   1. **Giới thiệu**

Facebook Messenger là một dịch vụ và ứng dụng phần mềm tin nhắn tức thời chia sẻ giao tiếp bằng ký tự và giọng nói. Được tích hợp trên ứng dụng Chat (trò chuyện) của [Facebook](https://vi.wikipedia.org/wiki/Facebook) và được xây dựng trên giao thức [MQTT](https://vi.wikipedia.org/w/index.php?title=MQTT&action=edit&redlink=1). Messenger cho phép người dùng Facebook trò chuyện với bạn bè trên cả di động và trang web chính.

Theo báo các của Facebook vào tháng 3 năm 2015 Facebook Messenger đạt 600 triệu người sử dụng. [David A. Marcus](https://vi.wikipedia.org/w/index.php?title=David_A._Marcus&action=edit&redlink=1) đứng đầu Facebook Messenger và tham gia Facebook dựa trên lời mời của [Mark Zuckerberg](https://vi.wikipedia.org/wiki/Mark_Zuckerberg), CEO của Facebook. Đến năm 2016 Facebook Messenger đã lên tới 1 tỉ người

* 1. **Ưu điểm/Nhược điểm**

Ưu điểm

* Trao đổi thông tin với người thân dễ dàng
* Chia sẻ tệp, thông báo địa điểm

Nhược điểm

* Thường xuyên nhận được tin nhắn rác
  1. **Một số tính năng nổi bật**
* Trò truyện với bạn bè hoặc người lạ
* Gửi tệp
* Voice/Video call
* Trò chuyện nhóm
* Trò chuyện bí mật
* Chơi game

1. **Kết luận chương**

Trong chương 1 đã nêu ra được các vấn đề chính sau:

* Đưa ra cái nhìn tổng quan và mục đích của đề tài
* Khảo sát các ứng dụng nổi tiếng trên thế giới tại thời điểm hiện tại

# **CHƯƠNG II: CƠ SỞ LÝ THUYẾT**

1. **Ngôn ngữ Java**
2. **Java là gì?**

Java là một ngôn ngữ lập lập trình, được phát triển bởi Sun Microsystem vào năm 1995, là ngôn ngữ kế thừa trực tiếp từ C/C++ và là một ngôn ngữ lập trình hướng đối tượng.

Vì sao ngôn ngữ này lại được đặt tên là Java? Java là tên một hòn đảo ở Indonesia - hòn đảo nổi tiếng với loại coffee Peet và cũng là loại nước uống phổ biến của các kỹ sư Sun. Ban đầu Ngôn ngữ này được đặt tên là "Oak" (có nghĩa là "Cây sồi" - 1991), nhưng các luật sư của Sun xác định rằng tên đó đã được đăng ký nhãn hiệu nên các nhà phát triển đã phải thay thế bằng một tên mới - và cũng vì lý do trên mà cái tên Java đã ra đời và trở thành tên gọi chính thức của Ngôn ngữ này - Ngôn ngữ Lập trình Java.

1. **Ngôn ngữ Java**

Java là một ngôn ngữ lập trình hướng đối tượng (OOP) và dựa trên các lớp (class). Khác với phần lớn ngôn ngữ lập trình thông thường, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành bytecode, bytecode sau đó sẽ được môi trường thực thi (runtime environment) chạy.

Trước đây, Java chạy chậm hơn những ngôn ngữ dịch thẳng ra mã máy như C và C++, nhưng sau này nhờ công nghệ "biên dịch tại chỗ" - Just in time compilation, khoảng cách này đã được thu hẹp, và trong một số trường hợp đặc biệt Java có thể chạy nhanh hơn. Java chạy nhanh hơn những ngôn ngữ thông dịch như Python, Perl, PHP gấp nhiều lần. Java chạy tương đương so với C#, một ngôn ngữ khá tương đồng về mặt cú pháp và quá trình dịch/chạy

Cú pháp Java được vay mượn nhiều từ C & C++ nhưng có cú pháp hướng đối tượng đơn giản hơn và ít tính năng xử lý cấp thấp hơn. Do đó việc viết một chương trình bằng Java dễ hơn, đơn giản hơn, đỡ tốn công sửa lỗi hơn.

Trong Java, hiện tượng rò rỉ bộ nhớ hầu như không xảy ra do bộ nhớ được quản lý bởi Java Virtual Machine (JVM) bằng cách tự động "dọn dẹp rác". Người lập trình không phải quan tâm đến việc cấp phát và xóa bộ nhớ như C, C++. Tuy nhiên khi sử dụng những tài nguyên mạng, file IO, database (nằm ngoài kiểm soát của JVM) mà người lập trình không đóng (close) các streams thì rò rỉ dữ liệu vẫn có thể xảy ra.

1. **Đặc điểm cơ bản của Java**

Tiêu chí hàng đầu của Ngôn ngữ Lập trình Java là "Write Once, Run Anywhere" (Viết một lần, chạy mọi nơi), nghĩa là Java cho phép chúng ta viết code một lần và thực thi được trên các hệ điều hành khác nhau. Ví dụ, chúng ta viết code trên Hệ điều hành Windows và nó có thể thực thi được trên các Hệ điều hành Linux và Mac OS...

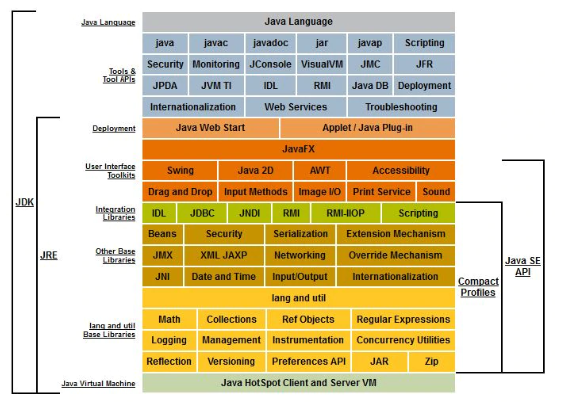
Với đặc điểm nổi bật đó, Java có những đặc điểm cơ bản như sau:

* Đơn giản và quen thuộc: Vì Java kế thừa trực tiếp từ C/C++ nên nó có những đặc điểm của ngôn ngữ này, Java đơn giản vì mặc dù dựa trên cơ sở C++ nhưng Sun đã cẩn thận lược bỏ các tính năng khó nhất của của C++ để làm cho ngôn ngữ này dễ sử dụng hơn.
* Hướng đối tượng và quen thuộc.
* Mạnh mẽ (thể hiện ở cơ chế tự động thu gom rác - Garbage Collection) và an toàn.
* Kiến trúc trung lập, độc lập nền tảng và có tính khả chuyển (Portability).
* Hiệu suất cao.
* Máy ảo (biên dịch và thông dịch).
* Phân tán.
* Đa nhiệm: Ngôn ngữ Java cho phép xâ dựng trình ứng dụng, trong đó nhiều quá trình có thể xảy ra đồng thời. Tính đa nhiệm cho phép các nhà lập trình có thể biên soạn phần mềm đáp ứng tốt hơn, tương tác tốt hơn và thực hiện theo thời gian thực.

1. **Các nền tảng cơ bản của Java**

Java gồm có 3 thành phần chính:

* Java Virtual Machine (Java VM): Máy ảo Java.
* Java Application Programming Interface (Java API).
* Java Development Kit (JDK) gồm trình biên dịch, thông dịch, trợ giúp, soạn tài liệu... và các thư viện chuẩn.



Hình 2‑1 Các thành phần của Java

1. **Tại sao chọn Java để phát triền Server-side**

* Miễn phí
* Cộng đồng phát triển mạnh mẽ có thể hỗ trợ mọi vấn đề
* Java quản lý bộ nhớ tốt, giúp lập trình viên có thể tập trung vào vấn đề phát triền, không phải suy nghĩ tới vấn đề dọn rác
* Đa nền tảng: Java có thể triển khai trên mọi hệ điều hành
* Native language: Java là ngôn ngữ native nên có thể tương tác với phần cứng nhiều hơn => rất có ích đối với các hệ thống lớn

1. **Embedded Jetty**
2. **Embedded Jetty là gì?**

Eclipse Jetty vừa là Java HTTP Server và Java Servet Container được phát triển bởi Eclipse Foudation. Jetty được phát triển cho mọi người sử dụng miễn phí và là phần mềm mã nguồn mở. Jetty được biết đến là một trong những công cụ mạnh mẽ trong việc phát triển phần mềm theo kiến trúc Mirco Service

Embedding Jetty có một slogan là “Don’t deploy your application in Jetty, deploy Jetty in your application!” nghĩa là thay vì đóng gói ứng dụng và deploy nó trên Web Server thì chúng ta có thể nhúng trực tiếp Web Server vào ứng dụng của mình

Embedded Jetty là một thư viện giúp cho lập trình viên có thể nhúng Jetty Web Server vào chương trình của mình thay vì triển khai ứng dụng trên 1 Web Server.

1. **So sánh Jetty với Tomcat**

Ở đây chúng ta sẽ so sánh Jetty với Tomcat. Tại sao lại là Tomcat mà không phải một Web Server nào khác? Đơn giản là bởi vì Tomcat là Web Server có cộng đồng lập trình viên sử dụng đông đảo nhất và cũng là một trong những công nghệ nổi tiếng nhất

|  |  |
| --- | --- |
| **Tomcat** | **Jetty** |
| Tomcat thuộc sở hữu của Apache | Jetty thuộc sở hữu của Eclipse |
| Tomcat được sử dụng rộng rãi và có một cộng đồng lớn có thể hỗ trợ mọi vấn đề liên quan tới Tomcat | Jetty ít được biết tới hơn, tuy nhiên cũng được sử dụng khá rộng rãi bởi khả năng nhúng của nó |
| Tomcat là Java Servlet cho các server khác, bao gồm JBoss | Jetty là thư viện viết bằng Java và có thể sử dụng trực tiếp trong mã nguồn |
| Tomcat khá là nặng và khó tìm hiểu | Jetty dễ dàng nhúng và nhẹ hơn rất nhiều |

Bảng 2‑1 So sánh giữa Jetty và Apache

1. **Ưu điểm của Jetty**

Tuy có nhiều điểm không bằng Tomcat, nhưng Jetty cũng có những ưu điểm vượt trội hơn so với các Web Server khác:

* Sử dụng ít bộ nhớ hơn và nhẹ hơn do đó cung cấp tốc độ và khả năng mở rộng
* Có thể được nhúng dễ dàng trong ứng dụng Java Web cũng như phục vụ như máy chủ không đồng bộ
* Mã nguồn mở và có sự hỗ trợ tốt từ cộng đồng
* Có vùng phủ bộ nhớ (memory footprint) nhỏ nên có thể khởi chạy và khởi động lại nhanh chóng
* Tùy biến cao
* Quy mô nhỏ cùng với hiệu quả cao, chi phí bảo trì và tổng chi phí sở hữu thấp

1. **MongoDB**
2. **Giới thiệu về MongoDB**

MongoDB là một trong những cơ sở dữ liệu mã nguồn mở NoSQL phổ biến nhất được biết bằng C++. Tính đến tháng 2/2015, MongoDB được xếp thứ 4 trong số các hệ thống cơ sở dữ liệu phổ biến nhất. Nó được phát triển bởi công ty 10gen sau này được biết đến với tên MongoDB Inc..

MongoDB hỗ trợ đa nền tảng, nó có thể chạy trên Windows, Linux và Mac...Nó hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như C#, Java, PHP, Javascript...và các môi trường phát triển khác nhau.

MongoDB là cơ sở dữ liệu hướng tài liệu, nó lưu trữ dữ liệu trong các bản ghi dạng BSON với cấu trúc động rất linh hoạt. Nghĩa là bạn có thể lưu các bản ghi mà không cần lo lắng về cấu trúc dữ liệu như là số trường, kiểu của trường lưu trữ. Cấu trúc bản ghi của MongoDB tương tự như các đối tượng JSON.

1. **Ưu điểm và nhược điểm của MongoDB**
   1. **Ưu điểm**

* Mã nguồn mở: MongoDB là phần mềm mã nguồn mở miễn phí và có cộng đồng phát triển rất lớn
* Hiệu năng cao: Tốc độ truy vấn của MongoDB nhanh hơn hẳn so với Hệ quản trị CSDL quan hệ (RDBMS). Thử nghiệm cho thấy tốc độ thêm bản ghi mới nhanh hơn SQL khoảng 100 lần
* Dữ liệu linh hoạt: MongoDB là dạng cơ sở dữ liệu hướng tài liệu (document database), lưu dữ liệu dướng dạng BSON, không bó buộc về số lượng trường và kiểu dữ liệu.
* Rich Query Language: MongoDB có sẵn các phương thức CRUD khiến cho việc sử dụng trở nên dễ dàng hơn
* Tính sẵn có: Giống như các RDBMS khác, MongoDB cũng hỗ trợ Replica Set nhằm việc sao lưu và khôi phục dữ liệu
* Khả năng mở rộng: Trong MongoDB có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một node với vào cluster
  1. **Nhược điểm**

Tuy có rất nhiều ưu điểm đáng chú ý, nhưng MongoDB vẫn có một số những nhược điểm sau:

* MongoDB không có các tính chất ràng buộc như trong RDBMS –> dễ bị làm sai dữ liệu
* Không hỗ trợ join giống như RDBMS nên khi viết function join trong code ta phải làm bằng tay khiến cho tốc độ truy vấn bị giảm.
* Sử dụng nhiều bộ nhớ: do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên sẽ bị dữ thừa dữ liệu (trong RDBMS thì ta chỉ cần lưu 1 bản ghi rồi các bản ghi khác tham chiếu tới còn trong MongoDB thì không)

1. **So sánh MongoDB với MySQL**

Ở đây lựa chọn MySQL để so sánh trực tiếp vì MySQL là một trong những Hệ quản trị CSDL quan hệ rất mạnh và có cộng đồng cũng rất phát triển

|  |  |  |
| --- | --- | --- |
|  | **MySQL** | **MongoDB** |
| Viết bởi ngôn ngữ | C, C++ | C, C++, JS |
| Kiểu | RDBMS | Document Oriented |
| Đặc điểm chính | Table, Row, Column | Collection, Document, Field |
| Giấy phép | GPL v2 / Giấy phép thương mại có sẵn OD | OD GNU AGPL v3.0 / Giấy phép thương mại có sẵn OD |
| Lược đồ | Strict | Dynamic |
| Mở rộng | Theo chiều dọc | Theo chiều ngang |
| Các tính năng chính | - Tìm kiếm và đán chỉ mục  - Hỗ trợ replication  - Trigger  - Truy vấn lồng  - Truy vấn bộ nhớ đệm  - Hỗ trợ SSL  - Hỗ trợ Unicode  - Công cụ lưu trữ khác nhau với các đặc tính hiệu suất khác nhau | - Auto-sharding  - Hỗ trợ replication  - Hỗ trợ dữ liệu nhúng  - Chỉ mục phụ toàn diện  - Ngôn ngữ truy vấn phong phú  - Hỗ trợ công cụ lưu trữ khác nhau |
| Sử dụng tốt nhất cho | - Cấu trúc dữ liệu phù hợp với bảng và hàng  - Cập nhật thường xuyên và sửa đổi khối lượng lớn bản ghi  - Bộ dữ liệu tương đối nhỏ | - Chịu tải read/write tốt  - Lược đồ không ổn định  - phù hợp với hệ thống dữ liệu lớn  - Tính sẵn sàng cao  - Không có CSDL của quản trị viên |

Bảng 2‑2 So sánh MongoDB và MySQL

1. **Tại sao chọn MongoDB?**

* Vì đây là hệ thống trò chuyện trực tuyến nên yếu tố realtime là quan trọng nhất. Do đó với đặc điểm truy vấn cực nhanh của MongoDB là một trong những yếu tố quan trọng nhất
* Hệ thống phải lưu lại rất nhiều dữ liệu người dùng, lịch sử trò chuyện… nên với số lượng người dùng lớn đây sẽ trở thành hệ thống bigdata. MongoDB làm việc rất hiệu quả với bigdata hiệu quả hơn RDBMS rất nhiều
* Tần suất read/write của hệ thống là rất lớn

1. **ReactJS**
2. **Giới thiệu về ReactJS**

React là một thư viện UI phát triển tại Facebook để hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được.

ReactJS đang nổi lên trong những năm gần đây với xu hướng Single Page Application. Trong khi những framework khác cố gắng hướng đến một mô hình MVC hoàn thiện thì React nổi bật với sự đơn giản và dễ dàng phối hợp với những thư viện Javascript khác

React cho phép những nhà phát triển web tạo ra giao diện người dung nhanh chóng. Phần Views của Reactjs thường được hiển thị bằng việc chủ yếu dung các component mà chứa các component cụ thể hoặc các thẻ HTML. Một trong những đặc trưng duy nhất của Reactjs là việc render dữ liệu không những có thể thực hiện ở tầng server mà còn ở tầng client.

Nó cũng sử dụng khái niệm là Virtual DOM (DOM ảo). Virtual DOM tạo ra bản cache cấu trúc dữ liệu của ứng dụng trên bộ nhớ. Sau đó, ở mỗi vòng lặp, nó liệt kê những thay đổi và sau đó là cập nhật lại sự thay đổi trên DOM của trình duyệt một cách hiệu quả. Điều này cho phép ta viết các đoạn code như thể toàn bộ trang được render lại dù thực tế là Reactjs chỉ render những component hay subcomponent nào thực sự thay đổi.

React được sử dụng tại Facebook trong production, và trang web nổi tiếng www.instagram.com được viết hoàn toàn trên React.

1. **Ưu điểm và nhược điểm của ReactJS**
   1. **Ưu điểm**

* Reactjs cực kì hiệu quả: Reactjs tạo ra cho chính nó DOM ảo – nơi mà các component thực sự tồn tại trên đó. Điều này sẽ giúp cải thiện hiệu suất rất nhiều. Reactjs cũng tính toán những thay đổi nào cần cập nhật len DOM và chỉ thực hiện chúng. Điều này giúp Reactjs tránh những thao tác thừa trên DOM
* Reactjs giúp việc viết các đoạn code JS dễ dàng hơn: Nó dung cú pháp đặc biệt là JSX (Javascript mở rộng) cho phép ta trộn giữa code HTML và Javascript. Ta có thể them vào các đoạn HTML vào trong hàm render mà không cần phải nối chuỗi. Đây là đặc tính thú vị của Reactjs. Nó sẽ chuyển đổi các đoạn HTML thành các hàm khởi tạo đối tượng HTML bằng bộ biến đổi JSX.
* Reactjs có nhiều công cụ phát triển: Khi bạn bắt đầu Reactjs, đừng quên cài đặt ứng dụng mở rộng của Chrome dành cho Reactjs. Nó giúp bạn debug code dễ dàng hơn. Sau khi bạn cài đặt ứng dụng này, bạn sẽ có cái nhìn trực tiếp vào virtual DOM như thể bạn đang xem cây DOM thông thường.
* Render tầng server: Một trong những vấn đề với các ứng dụng đơn trang là tối ưu SEO và thời gian tải trang. Nếu tất cả việc xây dựng và hiển thị trang đều thực hiện ở client, thì người dung sẽ phải chờ cho trang được khởi tạo và hiển thị lên. Điều này thực tế là chậm. Hoặc nếu giả sử người dung vô hiệu hóa Javascript thì sao? Reactjs là một thư viện component, nó có thể vừa render ở ngoài trình duyệt sử dụng DOM và cũng có thể render bằng các chuỗi HTML mà server trả về
* Làm việc với vấn đề test giao diện: Nó cực kì dễ để viết các test case giao diện vì virtual DOM được cài đặt hoàn toàn bằng JS.
* Hiệu năng cao đối với các ứng dụng có dữ liệu thay đổi liên tục, dễ dàng cho bảo trì và sửa lỗi.
  1. **Nhược điểm**
* ReactJS chỉ phục vụ cho tầng View. React chỉ là View Library nó không phải là một MVC framework như những framework khác. Đây chỉ là thư viện của Facebook giúp render ra phần view. Vì thế React sẽ không có phần Model và Controller, mà phải kết hợp với các thư viện khác. React cũng sẽ không có 2-way binding hay là Ajax
* Tích hợp Reactjs vào các framework MVC truyền thống yêu cầu cần phải cấu hình lại.
* ReactJS khá nặng nếu so với các framework khác React có kích thước tương tương với Angular (Khoảng 35kb so với 39kb của Angular). Trong khi đó Angular là một framework hoàn chỉnh
* Khó tiếp cận cho người mới học Web

1. **Redux**
   1. **Giới thiệu về Redux**

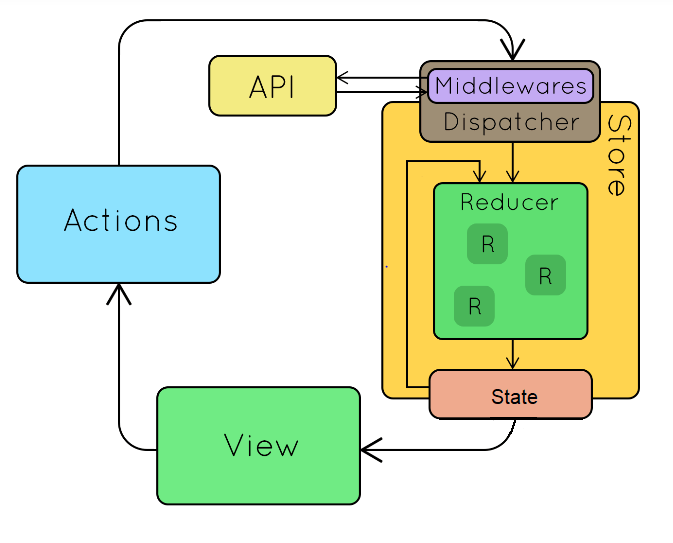
Do yêu cầu cho các ứng dụng single-page sử dụng Javascript ngày càng trở lên phức tạp thì code của chúng ta phải quản lý nhiều state hơn. State có thể bao gồm là data trả về từ phía Server và được cached lại hay như dữ liệu được tạo ra và thao tác ở phía client mà chưa được đẩy lên phía server. UI state cũng trở lên phức tạp vì chúng ta cần quản lý việc active Routes, selected tabs, spinners, điều khiển phân trang …vv.

Việc quản lý từng thay đổi của state là rất khó: Vì nếu như khi ta thay đổi 1 Model -> Model khác cũng thay đổi theo và nếu 1 View thay đổi -> Model thay đổi -> Model khác nữa cũng thay đổi theo. Về phía Front-end development, thì được kỳ vọng xử lý các vấn đề như “optimistic updates” (Hiểu như thể một thay đổi trên giao diện được thực hiện thành công trước khi được xác nhận từ phía server), render phía server, featching data trước khi thực hiện chuyển trang …vv. Nếu như ko có một luồng xử lý khoa học và rõ ràng thì việc quản lý dữ liệu với các dự án lớn là vô cùng phức tạp.

Vấn đề phức tạp như đề cập ở trên sẽ còn càng khó xử lý hơn nữa vì chúng ta đang trộn lẫn 2 khái niệm mà nó ko tự nhiên với tư duy thông thường của con người đó là: Biến đổi dữ liệu (Mutation) và tính bất đồng bộ (Asynchronicity), mặc dù chúng sẽ là tuyệt vời nếu dùng tách rời nhau, nhưng hoạt động cùng nhau sẽ gây ra sự hỗn độn.

ReactJS đã cố gắng giải quyết điều này ở trong tầng View bằng việc loại bỏ Bất đồng bộ và Không thao tác trực tiếp trên DOM, tuy nhiên việc quản lý state là tùy thuộc vào cách của bạn do vậy đây chính là lí do Redux xuất hiện để giúp bạn quản lý state một cách khoa học và hiệu quả hơn.

* 1. **Nguyên lý vận hành của Redux**



Hình ‑ Luồng xử lý của Redux

1. **Tại sao lựa chọn ReactJS và Redux để phát triển Client-side**

* Dễ dàng tích hợp
* ReactJS cung cấp các module thích hợp để quản lý định tuyến, trạng thái của ứng dụng, testing và một loạt các thư viện có thể sử dụng ngay lập tức để giải quyết hầu hết các vấn đề phổ biến nhất
* Sở hữu Virtual DOM để quản lý thay đổi của DOM, nâng cao hiệu suất và tốc độ của ứng dụng
* Với sự hỗ trợ của Redux sẽ tránh được sữ phức tạp của liên kết dữ liệu 2 chiều. Các thành phần sẽ được phân cấp cha-con và giao tiếp với nhau thông qua action-store-reducer

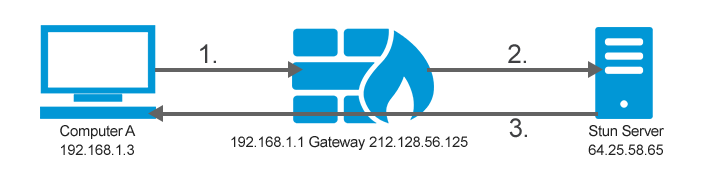
1. **WebRTC**
2. **WebRTC là gì?**

WebRTC (Web Real-Time Communication) là một dự án nguồn mở, miễn phí, cung cấp các trình duyệt web và ứng dụng di động với giao tiếp thời gian thực (RTC) thông qua các API. Nó cho phép giao tiếp âm thanh và video hoạt động bên trong các trang web bằng cách cho phép giao tiếp ngang hàng trực tiếp, loại bỏ nhu cầu cài đặt plugin hoặc tải xuống ứng dụng gốc. WebRTC cược hỗ trợ bởi Google, Microsoft, Mozilla và Opera, WebRTC đang được chuẩn hóa thông qua World Wide Web Consortium (W3C) và Lực lượng đặc nhiệm kỹ thuật Internet (IETF).

WebRTC gồm 3 thành phần chính là STUN, TURN, ICE

1. **STUN, TURN, ICE**
   1. **STUN**

STUN (Simple Traversal of User Datagram Protocol [UDP] Through Network Address Translators [NATs])cho phép các máy khách NAT (tức các máy tính nằm sau tường lửa) thiết lập cuộc gọi với máy cung cấp VOIP nằm ngoài mạng nội bộ.



Hình ‑ Cách thức hoạt động của STUN

STUN cho phép máy khách tìm ra địa chỉ công cộng của mình, loại NAT mà chúng đang đứng sau và cổng phía internet được NAT gắn liền với cổng nội bộ nào đó. Thông tin này được sử dụng để thiết lập giao tiếp UDP giữa máy khách và máy cung cấp VOIP và qua đó thiết lập cuộc gọi. Giao thức STUN được khai báo trong RFC 3489.

STUN được giao tiếp qua cổng UDP 3478, tuy nhiên máy chủ cũng sẽ gợi ý cho các máy khách thử kết nối với IP và số cổng khác (máy chủ STUN có hai địa chỉ IP). Chuẩn RFC quy định rằng cổng này và địa chỉ IP là tùy ý.

* 1. **TURN**

TURN: Traversal Using NAT Relay

Cũng giống như STUN tuy nhiên TURN hỗi trợ cả giao thức TCP làm giao thức truyền tải. TURN bổ xung cho hạn chế của STUN là hỗ trợ Symmetric NAT. Dữ liệu thay vì được gửi trực tiếp tới các peer thì các peer sẽ gửi dữ liệu tới các TURN server và TURN server sẽ đóng vai trò trung gian vận chuyển gói tin. Điều này nâng cao giúp chất lượng dịch vụ của ứng dụng mà còn đảm bảo an toàn thông tin khi truyền dẫn.

Tuy nhiên TURN cần một lưu lượng băng thông lớn nhất là với các chế độ từ full-HD trở lên

* 1. **ICE**

ICE (Interactive Communication Establishment) là giao thức được dung để thiết lập phiên media dựa trên UDP đi qua NAT một cách nhanh nhất

ICE sẽ tìm được kết nối tốt nhất giữa các peer bằng cách thử các khả năng kết nối một cách song song

ICE sẽ cố gắng tạo ra kết nối bằng cách sử dụng STUN, tuy nhiên nếu không thành công thì sẽ chuyển qua sử dụng TURN

1. **Tại sao lại lựa chọn WebRTC**

* WebRTC là dự án mã nguồn mở nên sẽ không phát sinh chi phí cho việc phát triển
* WebRTC dễ cài đặt và có thể cài đặt trực tiếp trên Server nên đảm bảo được quyền riêng tư của người dùng
* WebRTC hỗ trợ tất cả các trình duyệt
* WebRTC có thể hỗ trợ cuộc gọi video 1-1 hoặc cuộc gọi nhóm thậm chí là live stream

1. **Nginx**
2. **Giới thiệu về Nginx**

Nginx là một máy chủ proxy ngược mã nguồn mở (open source reverse proxy server) sử dụng phổ biến giao thức HTTP, HTTPS, SMTP, POP3 và IMAP , cũng như dùng làm cân bằng tải (load balancer), HTTP cache và máy chủ web (web server). Dự án Nginx tập trung vào việc phục vụ số lượng kết nối đồng thời lớn (high concurrency), hiệu suất cao và sử dụng bộ nhớ thấp. Nginx được biết đến bởi sự ổn định cao, nhiều tính năng, cấu hình đơn giản và tiết kiệm tài nguyên

NGINX đã được tạo ra để đáp ứng với những thách thức C10K xử lý ít nhất 10.000 khách hàng kết nối đồng thời trên một máy chủ duy nhất. NGINX sử dụng một kiến trúc event-driven không đồng bộ để xử lý những số lượng kết nối khổng lồ này. Kiến trúc này làm cho việc xử lý cao và dao động tải nhiều hơn dự đoán về cách sử dụng bộ nhớ RAM, sử dụng CPU, và độ trễ.

1. **Tính năng của Nginx**

* Có khả năng xử lý hơn 10.000 kết nối cùng lúc với bộ nhớ thấp
* Phục vụ tập tin tĩnh (static files) và lập chỉ mục tập tin
* Tăng tốc proxy ngược bằng bộ nhớ đệm (cache); cân bằng tải đơn giản và khả năng chịu lỗi
* Hỗ trợ tăng tốc với bộ nhớ đệm của FastCGI, uwsgi, SCGI, và các máy chủ memcached
* Kiến trúc modular; tăng tốc độ nạp trang bằng nén gzip tự động
* Hỗ trợ mã hoá SSL và TLS
* Cấu hình linh hoạt; lưu lại nhật ký truy vấn
* Chuyển hướng lỗi 3XX-5XX
* Rewrite URL (URL rewriting) dùng regular expressions
* Hạn chế tỷ lệ đáp ứng truy vấn
* Giới hạn số kết nối đồng thời hoặc truy vấn từ 1 địa chỉ
* Khả năng nhúng mã PERL
* Hỗ trợ và tương thích với IPv6
* Hỗ trợ WebSockets
* Hỗ trợ truyền tải file FLV và MP4

1. **Tại sao lựa chọn Nginx?**

* Lý do sử dụng Nginx đơn giản là bời vì nó phù hợp với hệ thống đang xây dựng. Khi hệ thống hoạt động, số lượng người dùng truy cập và sử dụng sẽ là rất lớn và khi đó sẽ cần sử dụng đến khả năng xử lý nhiều kết nối cùng lúc của Nginx
* Ngoài ra khi hệ thống trở nên lớn hơn sẽ cần Nginx là load-balancer để phân luồng
* Nginx sử dụng để làm proxy cho giao thức HTTPS: Do hệ thống dử dụng micro và camera của người dùng cho nên bắt buộc phải sử dụng giao thức HTTPS. Tuy nhiên việc setup HTTPS cho từng thành phần là rất tốn thời gian cho nên sẽ sử dụng Nginx để tiếp nhận các request HTTPS và chuyển hướng tới đích bằng giao thức HTTP

1. **Kết luận chương**

Trong chương này đã giải quyết được các vấn đề sau:

* Giới thiệu các công nghệ được sử dụng trong hệ thống
* Ưu điểm và nhược điểm của hệ thống
* Lý do tại sao lại sử dụng các công nghệ đó để xây dựng hệ thống

# **CHƯƠNG III: ĐẶC TẢ YÊU CẦU PHẦN MỀM**

1. **Phân tích chức năng, nghiệp vụ**
2. **Các chức năng chính**

* Đăng ký
* Đăng nhập
* Đăng xuất
* Lấy lại tài khoản
* Hiển thị thông tin người dùng
* Cập nhật thông tin
* Thay đổi mật khẩu
* Tìm kiếm bạn bè
* Thêm liên lạc mới
* Xóa liên lạc
* Hiển thị danh sách liên lạc
* Hiển thị danh sách tin nhắn gần nhất
* Hiển thị lịch sử tin nhắn
* Gửi tin nhắn văn bản
* Gửi tệp đính kèm
* Tạo cuộc trò chuyện nhóm
* Thêm thành viên vào cuộc trò chuyện nhóm
* Xóa cuộc trò chuyện
* Voice call
* Video call

1. **Mô tả chi tiết yêu cầu chức năng**

* **ID: FR01**

TITLE: Đăng ký

DESC: Người dùng đăng ký tài khoản mới

RAT: Chức năng cơ bản bắt buộc phải có

* **ID: FR02**

TITLE: Đăng nhập

DESC: Người dùng sử dụng tài khoản đã đăng ký trước đó để đăng nhập vào hệ thống

RAT: Chức năng cơ bản bắt buộc phải có

DEP: FR01

* **ID: FR03**

TITLE: Đăng xuất

DESC: Người dùng đăng xuất ra khỏi tài khoản hiện tại để đăng ký hoặc đăng nhập với tài khoản khác

RAT: Chức năng cơ bản bắt buộc phải có

DEP: FR01, FR02

* **ID: FR04**

TITLE: Lấy lại tài khoản

DESC: Người dùng sử dụng chức năng này để có thể lấy lại tài khoản trong trường hợp quên tài khoản hoặc tài khoản bị đánh cắp. Người dùng sẽ phải cung cấp một số thông tin để có thể lấy lại tài khoản của mình

RAT: Sử dụng để lấy lại tài khoản của người dùng

* **ID: FR05**

TITLE: Hiển thị thông tin người dùng

DESC: Người dùng sử dụng tính năng này để hiển thị thông tin của bản thân hoặc của bạn bè. Các thông tin cơ bản sẽ được hiển thị bao gồm: tên, giới tính, ngày sinh, số điện thoại, ảnh đại diện

RAT: Sử dụng để hiển thị thông tin của chính mình hoặc người khác

DEP: FR01, FR02

* **ID: FR06**

TITLE: Cập nhật thông tin

DESC: Người dùng có nhu cầu thay đổi các thông tin cá nhân sẽ sử dụng tính năng này để thay đổi. Người dùng sẽ được phép thay đổi các thông tin cá nhân cơ bản như: tên, giới tính, ngày sinh, số điện thoại, ảnh đại diện

RAT: Sử dụng để thay đổi thông tin người dùng

DEP: FR01, FR02

* **ID: FR07**

TITLE: Thay đổi mật khẩu

DESC: Người dùng có thể thay đổi mật khẩu của mình trong trường hợp nghi ngờ người khác biết mật khẩu

RAT: Sử dụng để thay đổi mật khẩu

DEP: FR01, FR02

* **ID: FR08**

TITLE: Tìm kiếm bạn bè

DESC: Người dùng sẽ sử dụng tính năng này để tìm kiếm bạn bè mới và có thể xem một số thông tin cơ bản của họ

RAT: Sử dụng để người dùng tìm kiếm bạn bè

DEP: FR01, FR02

* **ID: FR09**

TITLE: Thêm liên lạc mới

DESC: Người dùng sẽ thêm một người dùng khác vào danh sách liên lạc của mình

RAT: Sử dụng để người dùng thêm bạn bè

DEP: FR01, FR02, FR08

* **ID: FR10**

TITLE: Xóa liên lạc

DESC: Người dùng sẽ xóa người dùng khác khỏi danh sách liên lạc của mình

RAT: Sử dụng để người dùng xóa liên lạc

DEP: FR01, FR02, FR11

* **ID: FR11**

TITLE: Hiển thị danh sách liên lạc

DESC: Hiển thị danh sách liên lạc của người dùng

RAT: Người dùng phải biết danh sách liên lạc của mình gồm những ai

DEP: FR01, FR02, FR10, FR09

* **ID: FR12**

TITLE: Hiển thị danh sách tin nhắn gần nhất

DESC: Người dùng sẽ có thể xem danh sách tin nhắn gần nhất của mình, các tin nhắn sẽ được sắp xếp theo thứ tự thời gian

RAT: Người dùng có thể xem danh sách tin nhắn gần nhất

DEP: FR01, FR02

* **ID: FR13**

TITLE: Hiển thị lịch sử tin nhắn

DESC: Khi người dùng bấm vào chi tiết trong danh sách tin nhắn gần nhất, người dùng sẽ xem được lịch sử cuộc trò chuyện đó

RAT: Sử dụng để xem tin nhắn của cuộc trò chuyện

DEP: FR01, FR02, FR12

* **ID: FR14**

TITLE: Gửi tin nhắn văn bản

DESC: Tính năng quan trọng nhất của phần mềm giúp người dùng có thể trò chuyện với những người khác

RAT: Sử dụng để gửi tin nhắn cho những người khác

DEP: FR01, FR02

* **ID: FR15**

TITLE: Gửi tin tệp đính kèm

DESC: Tính năng quan trọng nhất của phần mềm giúp người dùng có thể gửi các tệp tới những người khác

RAT: Sử dụng để gửi tệp cho những người khác

DEP: FR01, FR02

* **ID: FR16**

TITLE: Tạo cuộc trò chuyện nhóm

DESC: Trong trường hợp người dùng muốn tạo cuộc trò chuyện có nhiều hơn 2 người sẽ sử dụng tính năng này để thêm những người dùng khác vào cuộc trò chuyện

RAT: Sử dụng để tạo cuộc trò chuyện nhóm

DEP: FR01, FR02

* **ID: FR17**

TITLE: Thêm thành viên vào cuộc trò chuyện nhóm

DESC: Trong trường hợp người dùng muốn thêm một người khác vào cuộc trò chuyện nhóm đã tạo trước đó

RAT: Sử dụng để thêm người và cuộc trò chuyện nhóm

DEP: FR01, FR02, FR16

* **ID: FR18**

TITLE: Xóa cuộc trò chuyện

DESC: Người dùng sử dụng tính năng để xóa toàn bộ lịch sử của cuộc trò chuyện

RAT: Sử dụng để xóa lịch sử trò chuyện

DEP: FR01, FR02

* **ID: FR19**

TITLE: Voice/video call

DESC: Người dùng sử dụng tính năng gọi điện bằng âm thanh hoặc hình ảnh cho người khác

RAT: Sử dụng để tạo cuộc gọi với người khác

DEP: FR01, FR02

1. **Phân tích các Usecase**
2. **Mô hình Use Case tổng thể**

****

Hình ‑ Mô hình Use Case

1. **Đặc tả chi tiết các Use Case**

|  |  |
| --- | --- |
| **Use Case** | **Đăng ký** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web |
| **Điều kiện sau** | * Người dùng chuyển sang màn hình chính |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào tab “Đăng ký” 2. Người dùng nhập các thông tin: e-mail, tên người dùng, ngày sinh, giới tính, số điện thoại, mật khẩu, mật khẩu xác nhận 3. Người dùng bấm vào nút “Đăng ký” 4. Hệ thống đăng ký thành công và chuyển người dùng tới trang chính |
| **Luồng sự kiện thay thế** | Tại bước 4, ở các trường hợp:   * E-mail đã đăng ký * E-mail không hợp lệ * Mật khẩu không phù hợp * Mật khẩu xác nhận không đúng * Ngày sinh không hợp lệ   Hệ thống sẽ trả thông báo lỗi ở màn hình đăng ký và người dung sẽ phải nhập thông tin phù hợp |

Bảng ‑ Đăng ký

|  |  |
| --- | --- |
| **Use Case** | **Đăng nhập** |
| **Điều kiên trước** | * Người dùng truy cập vào trang web |
| **Điều kiện sau** | * Người dùng chuyển sang màn hình chính |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào tab “Đăng nhập” 2. Người dùng nhập các thông tin: e-mail, mật khẩu 3. Người dùng bấm vào nút “Đăng nhập” 4. Hệ thống kiểm tra thông tin người dùng và chuyển người dùng tới trang chính |
| **Luồng sự kiện thay thế** | Tại bước 4, ở các trường hợp:   * E-mail không hợp lệ * Mật khẩu không đúng   Hệ thống sẽ trả thông báo lỗi ở màn hình đăng nhập và người dùng sẽ phải nhập thông tin phù hợp |

Bảng ‑ Đăng nhập

|  |  |
| --- | --- |
| **Use Case** | **Đăng xuất** |
| **Điều kiên trước** | * Người dùng đã đăng nhập vào trang chính |
| **Điều kiện sau** | * Người dùng được chuyển về màn hình đăng ký/đăng nhập |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào biểu tượng bánh răng bên cạnh tên của mình 2. Menu hiển thị các chức năng trong đó có chức năng đăng xuất 3. Người dùng bấm vào nút “Đăng xuất” 4. Hệ thống xóa dữ liệu đăng nhập và chuyển người dùng tới trang đăng ký/đăng nhập |

Bảng ‑ Đăng xuất

|  |  |
| --- | --- |
| **Use Case** | **Lấy lại tài khoản** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng ký tài khoản trong hệ thống nhưng quên mất mật khẩu |
| **Điều kiện sau** | * Người dùng nhận thay đổi mật khẩu mới thành công và được chuyển tới trang chính |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào chức năng “Quên mật khẩu” 2. Người dùng nhập thông tin e-mail 3. Người dùng nhận được 1 email có chứa mã xác nhận 4. Người dùng nhập mã xác nhận cùng với mật khẩu mới và mật khẩu xác nhận 5. Hệ thống thay đổi mật khẩu mới cho người dùng và chuyển người dùng tới trang chính |
| **Luồng sự kiện thay thế** | Tại bước 2, ở các trường hợp:   * E-mail không hợp lệ * E-mail không tồn tại trong hệ thống   Hệ thống đưa ra thông báo “E-mail không hợp lệ, vui lòng kiểm tra lại”  Tại bước 4, ở các trường hợp:   * Mã xác nhận không đúng * Mật khẩu và mật khẩu xác nhận không trùng khớp   Người dung cũng sẽ được nhận thông báo lỗi tương ứng trên màn hình |

Bảng ‑ Lấy lại tài khoản

|  |  |
| --- | --- |
| **Use Case** | **Hiển thị thông tin người dùng** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Hiển thị thông tin chi tiết của người dùng |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào tên người dùng hoặc ảnh đại diện của bất kỳ người dùng nào (bao gồm chính mình) 2. Popup mới hiện lên trên màn hình và hiển thị thông tin về người dùng bao gồm: E-mail, tên người dùng, ngày sinh, giới tính, số điện thoại |

Bảng ‑ Hiển thị thông tin người dùng

|  |  |
| --- | --- |
| **Use Case** | **Cập nhật thông tin** |
| **Điều kiên trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Thông tin mới của người dùng được cập nhật thành công |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào hình bánh răng bên cạnh tên của mình 2. Menu mở rộng và hiển thị chức năng “Thay đổi thông tin cá nhân”, người dùng bấm vào chức năng này 3. Popup mới hiện lên trên màn hình và hiển thị thông tin về người dùng bao gồm: tên người dùng, ngày sinh, giới tính, số điện thoại. Ngoài ra người dùng sẽ phải nhập thêm mật khẩu hiện tại để xác nhận 4. Người dùng thanh đổi thông tin cá nhân của mình 5. Người dùng bấm vào nút cập nhật 6. Hệ thống thông báo người dùng cập nhật thành công và ẩn popup |
| **Luồng sự kiện thay thế** | Tại bước 5, ở các trường hợp:   * Ngày sinh không hợp lệ * Tên người dùng không phù hợp * Mật khẩu hiện tại không đúng   Người dùng cũng sẽ được nhận thông báo lỗi tương ứng trên màn hình |

Bảng ‑ Cập nhật thông tin người dùng

|  |  |
| --- | --- |
| **Use Case** | **Thay đổi mật khẩu** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Thông tin mới của người dùng được cập nhật thành công |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào hình bánh răng bên cạnh tên của mình 2. Menu mở rộng và hiển thị chức năng “Thay đổi thông mật khẩu”, người dùng bấm vào chức năng này 3. Popup mới hiện lên trên màn hình và hiển thị 3 ô: mật khẩu mới, mật khẩu xác nhận, mật khẩu cũ. 4. Người dùng điền đủ thông tin và bấn nút “Cập nhật” 5. Hệ thống thông báo người dùng thay đổi mật khẩu thành công và ẩn popup |
| **Luồng sự kiện thay thế** | Tại bước 4, ở các trường hợp:   * Mật khẩu xác nhận không đúng * Mật khẩu không phù hợp * Mật khẩu cũ không chính xác   Người dùng cũng sẽ được nhận thông báo lỗi tương ứng trên màn hình |

Bảng ‑ Thay đổi mật khẩu

|  |  |
| --- | --- |
| **Use Case** | **Tìm kiếm bạn bè** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Hiển thị danh sách những người có tên hoặc email gần giống với chuỗi mà người dùng tìm kiếm |
| **Luồng sử kiện chính** | 1. Nhập ký tự bất kỳ vào thanh tìm kiếm 2. Hệ thống sẽ tự động hiển thị ra danh sách những người có tên và email gần giống với chuỗi được tìm kiếm |

Bảng ‑ Tìm kiếm bạn bè

|  |  |
| --- | --- |
| **Use Case** | **Thêm liên lạc mới** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống * Người dùng sử dụng chức năng tìm kiếm bạn bè và được hệ thống trả kết quả |
| **Điều kiện sau** | * Những người được thêm và sẽ được hiển thị ở danh sách liên lạc |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào biểu tượng “Thêm” bên cạnh tên của những người trong danh sách tìm kiếm 2. Sau khi bấm, nút thêm sẽ bị khóa lại và tên liên lạc mới được hiển thị ở danh sách liên lạc |

Bảng ‑ Thêm liên lạc

|  |  |
| --- | --- |
| **Use Case** | **Hiển thị danh sách liên lạc** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Hiển thị danh sách những người dùng trong danh sách liên lạc |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào chức năng “Liên lạc” 2. Hiển thị danh sách liên lạc của người dùng |

Bảng ‑ Hiển thị danh sách liên lạc

|  |  |
| --- | --- |
| **Use Case** | **Xóa liên lạc** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống * Người dùng bấm vào chức năng “Liên lạc” |
| **Điều kiện sau** | * Liên lạc bị xóa không xuất hiện trong danh sách liên lạc |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào nút xóa trong danh sách liên lạc tương ứng với người cần xóa 2. Danh sách không hiển thị người vừa xóa |

Bảng ‑ Xóa liên lạc

|  |  |
| --- | --- |
| **Use Case** | **Hiển thị danh sách tin nhắn gần nhất** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Hiên thị danh sách tin nhắn gần nhất |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào chức năng “Tin nhắn” 2. Hiển thị danh sách liên lạc của người dùng 3. Các thông tin được hiện bao gồm: tên, thời gian, nội dung tin nhắn, số tin nhắn chưa đọc |

Bảng ‑ Hiển thị tin nhắn gần nhất

|  |  |
| --- | --- |
| **Use Case** | **Hiển thị lịch sử tin nhắn** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Hiên thị lịch sử tin nhắn của người dùng với liên lạc đó |
| **Luồng sử kiện chính** | 1. Người dùng bấm 2 lần liên tiếp vào các phần tử được hiển thị trong danh sách tin nhắn gần nhất hoặc trong danh sách liên lạc 2. Hiển thị lịch sử tin nhắn của người dùng |

Bảng ‑ Hiển thị lịch sử tin nhắn

|  |  |
| --- | --- |
| **Use Case** | **Gửi tin nhắn văn bản** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống * Người dùng sử dụng chức năng lịch sử tin nhắn |
| **Điều kiện sau** | * Gửi tin nhắn thành công tới người nhận |
| **Luồng sử kiện chính** | 1. Ở màn hình lịch sử tin nhắn, người dùng nhập chuỗi bất kỳ và bấm “Gửi” hoặc phím Enter 2. Tin nhắn mới sẽ được hiển thị trên màn hình lịch sử tin nhắn 3. Phía người nhận nhận được tin nhắn |

Bảng ‑ Gửi tin nhắn văn bản

|  |  |
| --- | --- |
| **Use Case** | **Gửi tệp đính kèm** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống * Người dùng sử dụng chức năng lịch sử tin nhắn |
| **Điều kiện sau** | * Gửi tin nhắn thành công tới người nhận |
| **Luồng sử kiện chính** | 1. Ở màn hình lịch sử tin nhắn, bấm vào icon gửi tệp đính kèm 2. Người dùng chọn tệp cần gửi và gửi cho người nhận 3. Phía người nhận nhận được tệp |

Bảng ‑ Gửi tệp đính kèm

|  |  |
| --- | --- |
| **Use Case** | **Tạo cuộc trò chuyện nhóm** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Tạo một nhóm chat mới bao gồm các liên lạc đã được thêm |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào nút “Tạo nhóm mới” 2. Hiển thi popupcho phép tìm kiếm và chọn liên lạc trong danh sách liên lạc để thêm vào nhóm mới 3. Người dùng bấm nút tạo nhóm |

Bảng ‑ Tạo cuộc trò chuyện nhóm

|  |  |
| --- | --- |
| **Use Case** | **Xóa cuộc trò chuyện** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * Cuộc trò chuyện biến mất khỏi danh sách tin nhắn gần nhất và không còn lịch sử chat giữa 2 người |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào nút “Xóa cuộc trò chuyện” trên menu mở rộng của lịch sử tin nhắn |

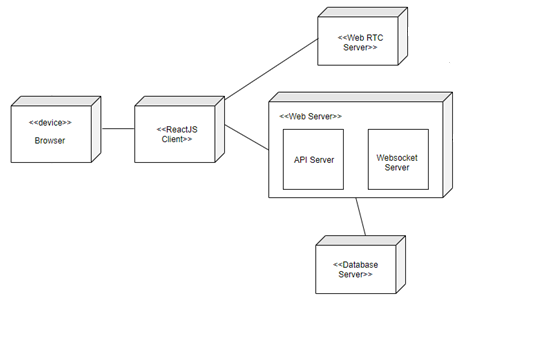
Bảng ‑ Xóa cuộc trò chuyện

|  |  |
| --- | --- |
| **Use Case** | **Voice/Video call** |
| **Điều kiện trước** | * Người dùng truy cập vào trang web * Người dùng đã đăng nhập vào hệ thống |
| **Điều kiện sau** | * 2 người có thể nhìn, nghe thấy nhau và trò chuyện bình thường |
| **Luồng sử kiện chính** | 1. Người dùng bấm vào nút “Gọi” trên menu mở rộng của lịch sử tin nhắn 2. Phía người nhận sẽ hiển thị màn hình chờ khi người nhận chưa bắt máy 3. Khi người nhận bắt máy sẽ hiển thị một frame chứa hình ảnh của người nhận |
| **Luồng sự kiện thay thế** | Ở bước 3, Nếu người nhận không bắt máy sẽ không hiện bất kỳ frame nào và thông báo cho người dùng |

Bảng ‑ Voice/Video call

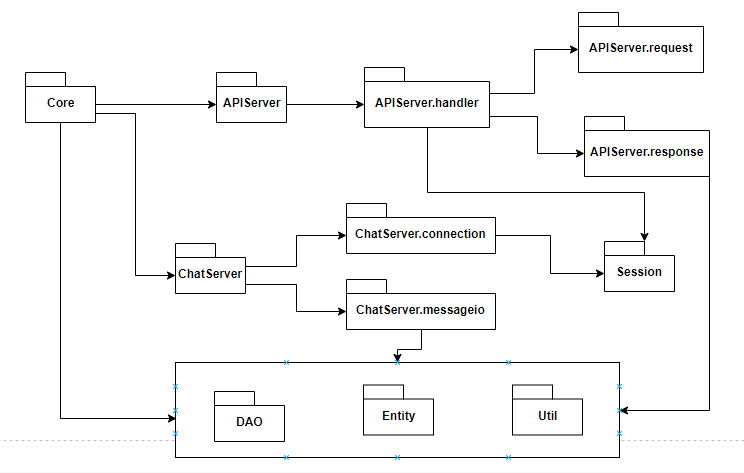
# **CHƯƠNG VI: THIẾT KẾ HỆ THỐNG**

1. **Mô hình kiến trúc**
2. **Deployment Diagram**



Hình ‑ Mô hình kiến trúc

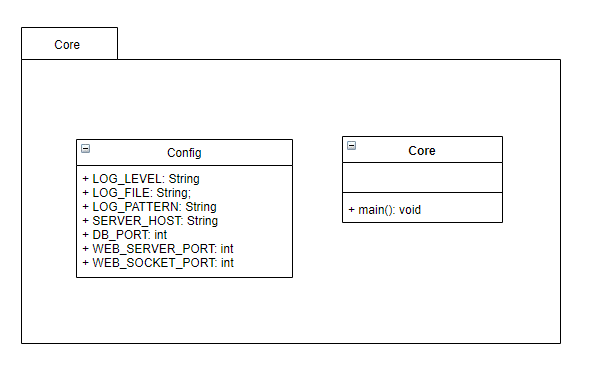
1. **Package Diagram**



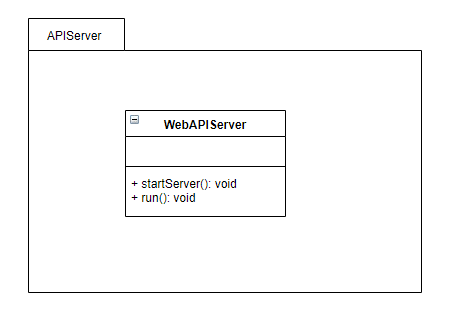
Hình ‑ Package Diagram

* Core: Chứa các configuration của server và chứa hàm main để khởi động toàn server
* DAO: Thực hiện việc kết nối tới DB
* Entity: Chứa các lớp thực thể
* Util: Chứa các hàm dùng chung và các constant
* APIServer: Thực hiện khởi tạo API Server
* APIServer.handler: Xử lý các request được gửi tới từ client
* ApiServer.request: Chứa thông tin request của Client
* ApiServer.response: Chứa các class xử lý api chuyên biệt
* ChatServer: Thực hiện khởi tạo WebSocket Server
* ChatServer.connection: Chứa các websocket connection đã được authen
* ChatServer.messageio: Thực hiện việc nhận và gửi dữ liệu qua socket
* Session: Chứa các phiên làm việc của người dùng, sử dụng để xác thực khi có yêu cầu xử lý hoặc yêu cầu kết nối từ của phía client

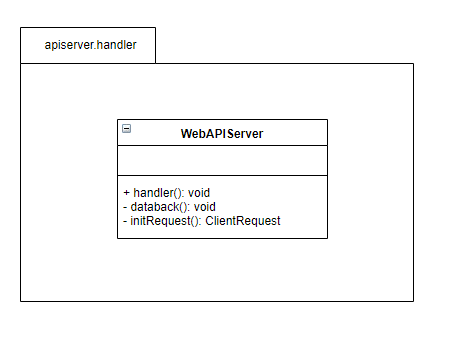
1. **Class Diagram**



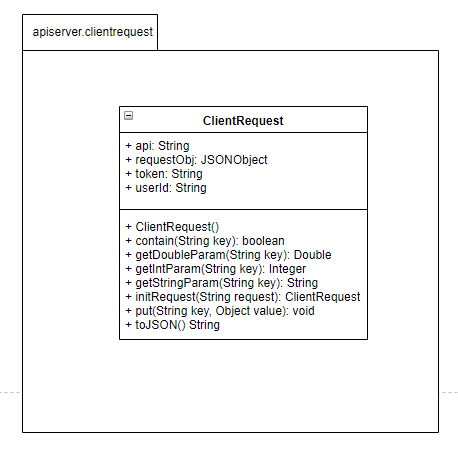
Hình ‑ Package core



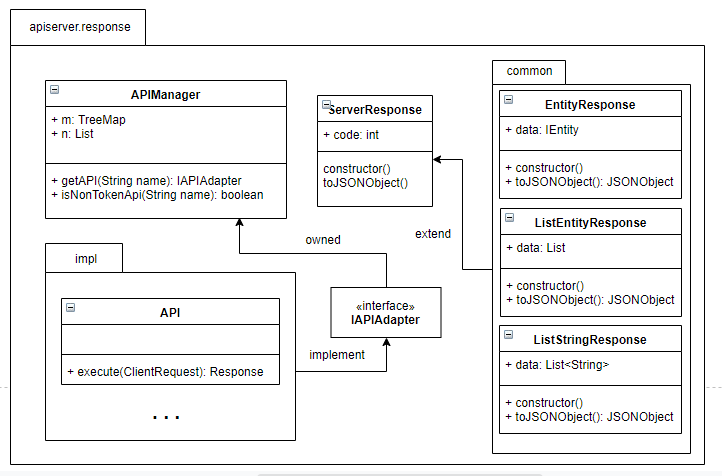
Hình ‑ Package apiserver



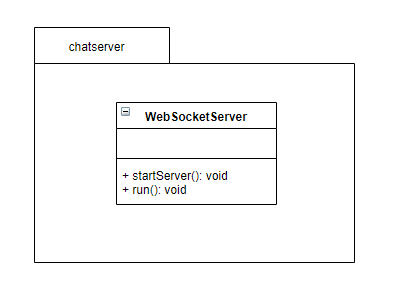
Hình ‑ Package apiserver.handler



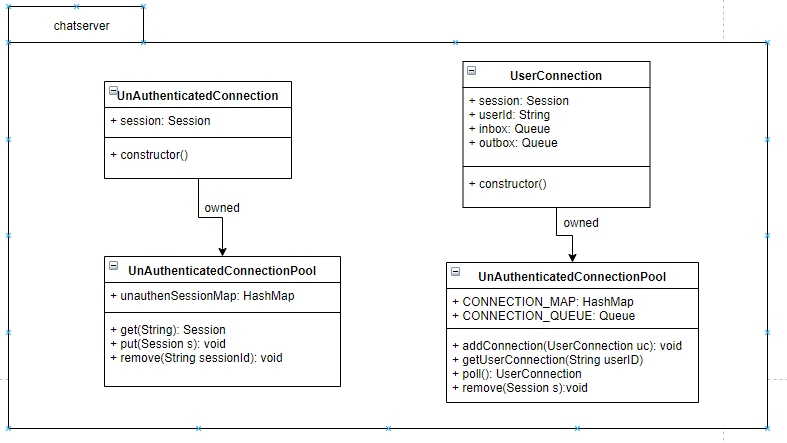
Hình ‑ Package apiserver.clientrequest



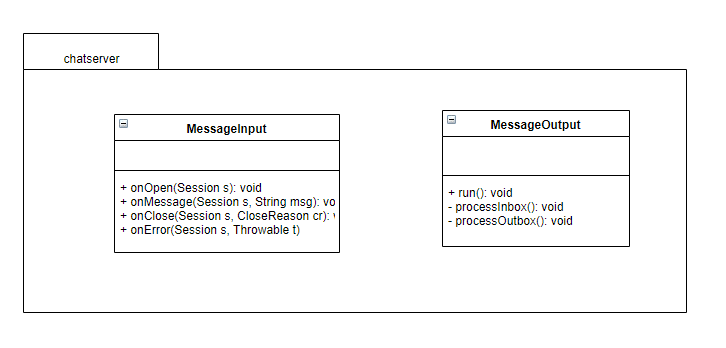
Hình ‑ Package apiserver.response



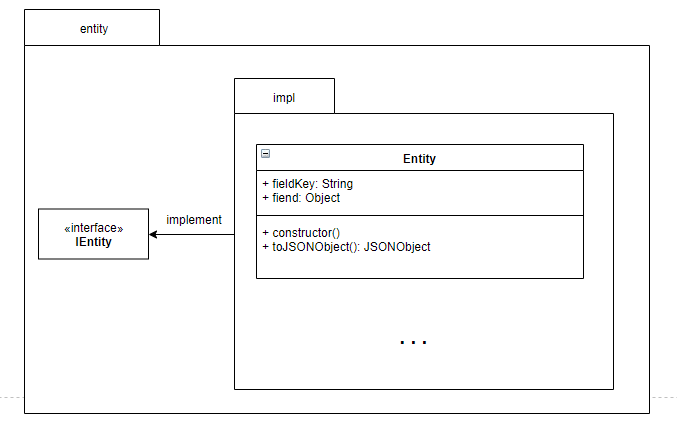
Hình ‑ Package chatserver



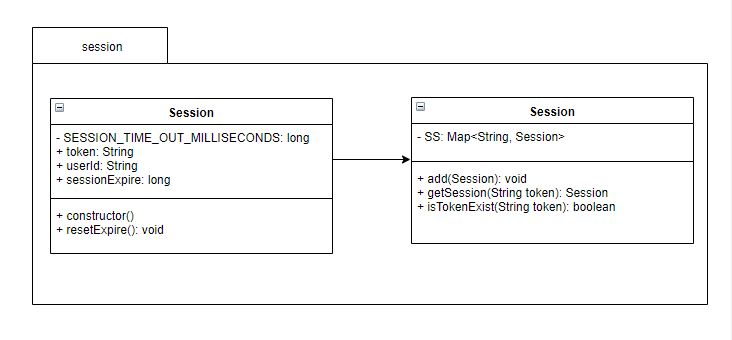
Hình ‑ Package chatserver.connection



Hình ‑ Package chatserver.messageio

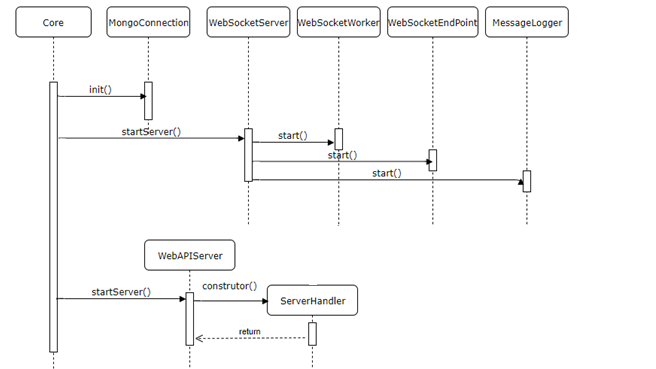


Hình ‑ Package entities

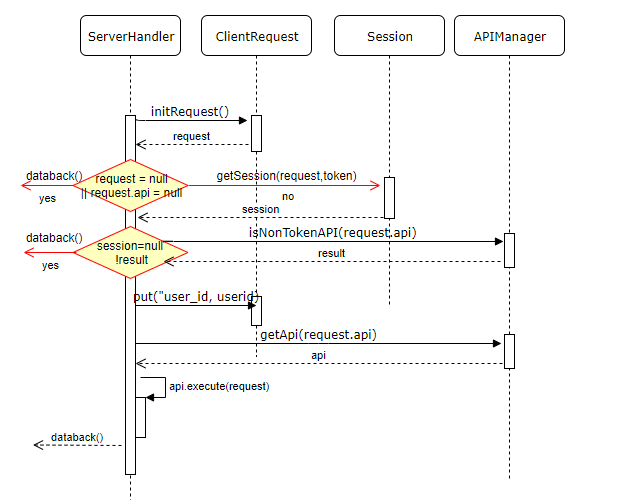


Hình ‑ Package session

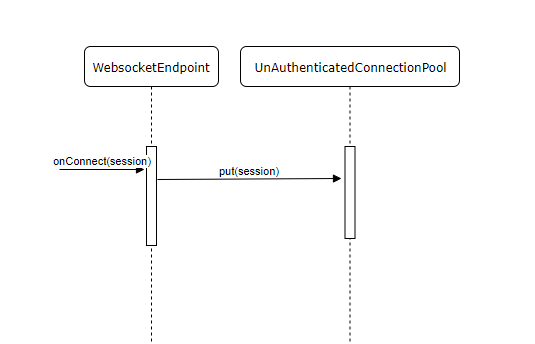
1. **Mô hình tuần tự**
2. **Mô hình tuần tự của Server**



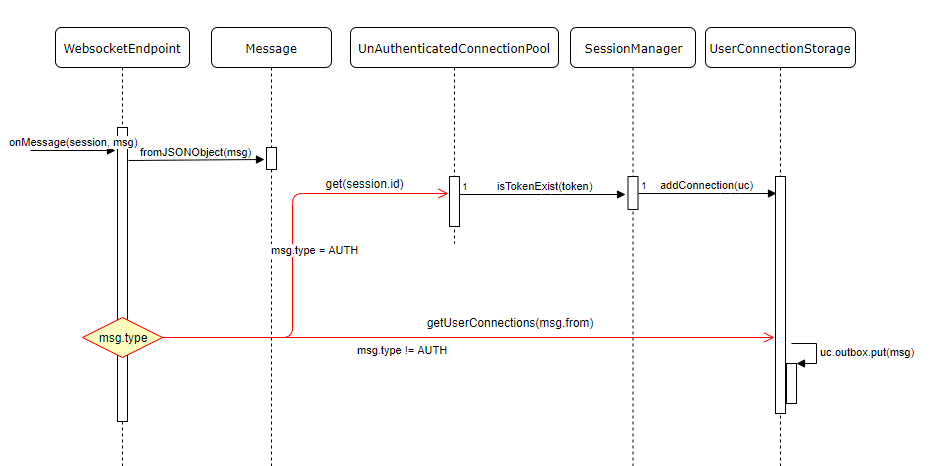
Hình ‑ Mô hình tuần tự khởi tạo server



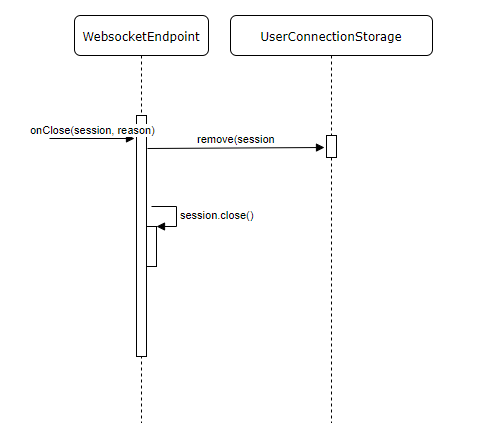
Hình ‑ Mô hình tuần tự tiếp nhận request từ client



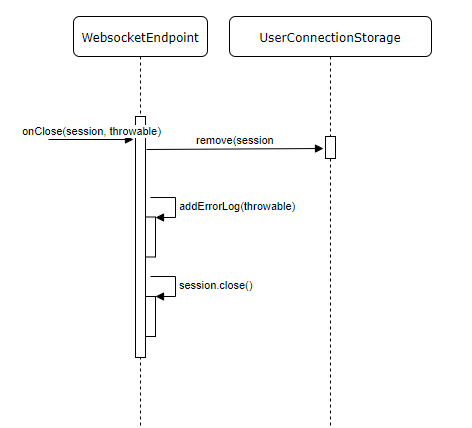
Hình ‑ Mô hình tuần tự tiếp nhận kết nối qua Web Socket



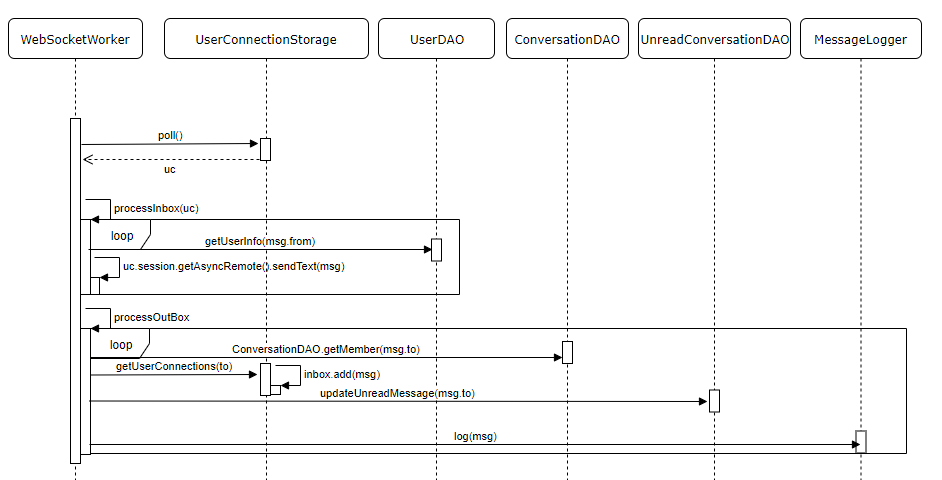
Hình ‑ Mô hình tuần tự tiếp nhận tin nhắn qua socket



Hình ‑ Mô hình tuần tự tiếp nhận sự kiện đóng kết nối

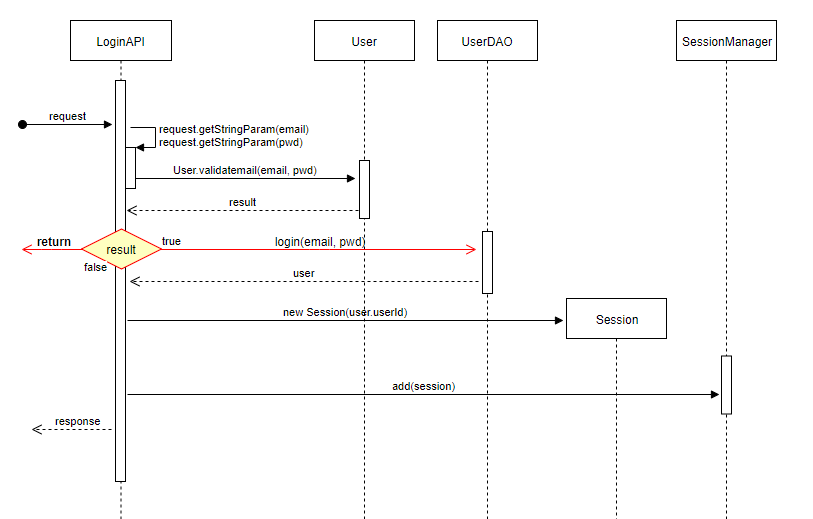


Hình ‑ Mô hình tuần tự xử lý kết nối lỗi

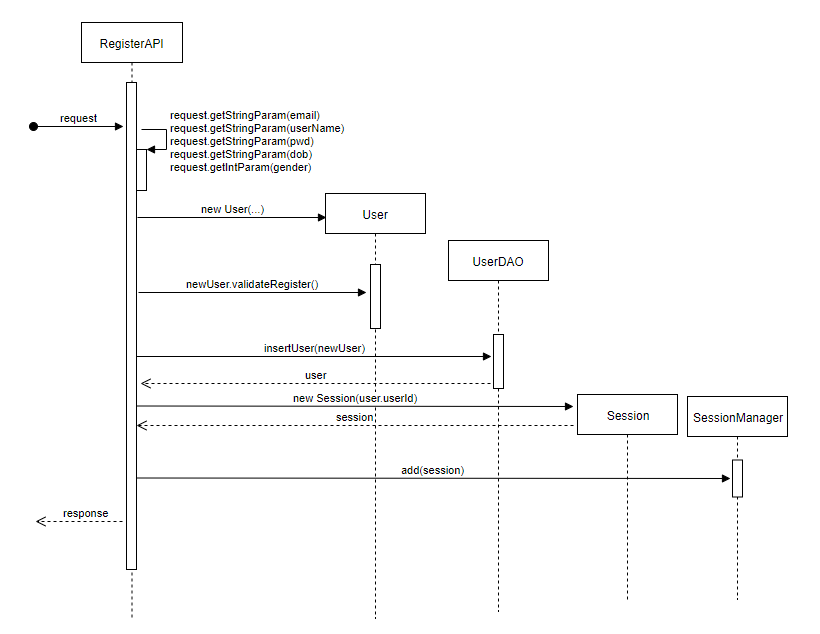


Hình ‑ Mô hình tuần tự nhận và gửi tin nhắn

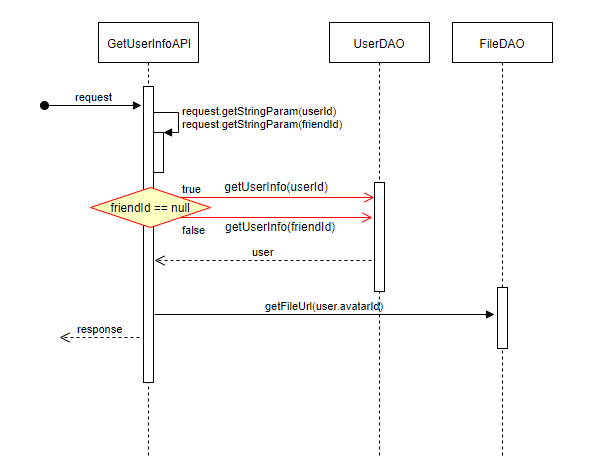
1. **Mô hình tuần tự của API**



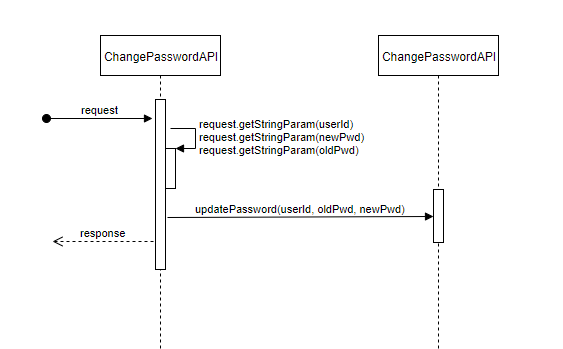
Hình ‑ Mô hình tuần tự của API Login



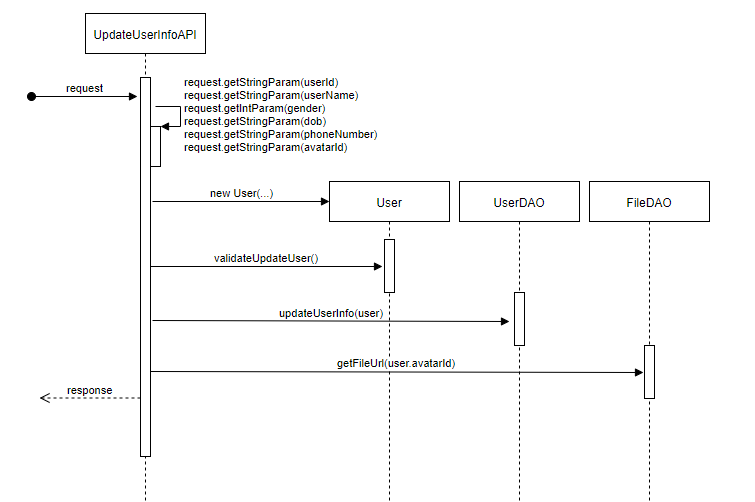
Hình ‑ Mô hình tuần tự của API Register



Hình ‑ Mô hình tuần tự của API GetUserInfo



Hình ‑ Mô hình tuần tự của API ChangePassword



Hình ‑ Mô hình tuần tự của API UpdateUserInfo

1. **Thiết kế CSDL**
2. **Database user**

* Collection user: lưu thông tin người dùng

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| user\_id | String | Mã người dùng |
| email | String | Email của người dùng |
| user\_name | String | Tên người dùng, sử dụng để hiển thị |
| sort\_name | String | Tên người dùng, sử dụng để tìm kiếm |
| gender | Int32 | Giới tính  Nam: 0  Nữ: 1 |
| dob | String | Ngày sinh, theo format  yyyy-MM-dd |
| ava\_id | String | Mã ảnh đại diện |
| original\_pwd | String | Mật khẩu |
| pwd | String | Mât khẩu dạng mã hóa, sử dụng để kiểm tra đăng nhập |
| phone\_number | String | Số điện thoại |

Bảng ‑ Collection user

* Collection contact: Lưu thông tin danh sách liên lạc

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| user\_id | String | Mã người dùng |
| friend\_id | String | Mã của liên lạc |
| time | String | Ngày thêm liên lạc, theo format yyyyMMddHHmmss |

Bảng ‑ Collection contact

* Collection converstion: Lưu thông tin cuộc trò chuyện

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| \_id | ObjectId | Mã người dùng |
| lst\_user | List | Danh sách những người trong cuộc trò chuyện |
| user\_id | String | Mã người dùng trong cuộc trò chuyện |
| conversation\_type | Int32 | Phân loại cuộc trò chuyện  0: tin nhắn riêng tư  1: nhóm |
| time | String | Ngày thêm cuộc trò chuyện, theo format yyyyMMddHHmmss |
| last\_message\_from | String | Người gửi tin nhắn cuối cùng |
| last\_message\_time | String | Thời gian gửi tin nhắn cuối cùng, theo format yyyyMMddHHmmss |
| last\_message\_type | String | Loại tin nhắn cuối cùng được gửi |
| last\_message\_value | String | Nội dung tin nhắn cuối cùng |

Bảng ‑ Collection conversation

* Collection unread\_conversation: Lưu số lượng tin nhắn chưa đọc theo từng user

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| conversation\_id | String | Mã cuộc trò chuyện |
| user\_id | String | Mã người dùng |
| unread\_number | Int32 | Số tin nhắn chưa đọc |

Bảng ‑ Collection unread\_conversation

1. **Database chatlog**

Collection trong database này sẽ lưu theo mã cuộc trò chuyện. Tức là collection name = conversation\_id

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| msg\_id | String | Mã tin nhắn |
| from | String | Mã người dùng gửi tin nhắn |
| type | String | Loại tin nhắn |
| value | String | Nội dung tin nhắn |
| time | String | Thời gian gửi tin nhắn, theo format yyyyMMddHHmmss |
| flag | Int32 | Trạng thái tin nhắn |

Bảng ‑ Collection của database chatlog

# **CHƯƠNG V: XÂY DỰNG HỆ THỐNG**

1. **Cài đặt hệ thống**

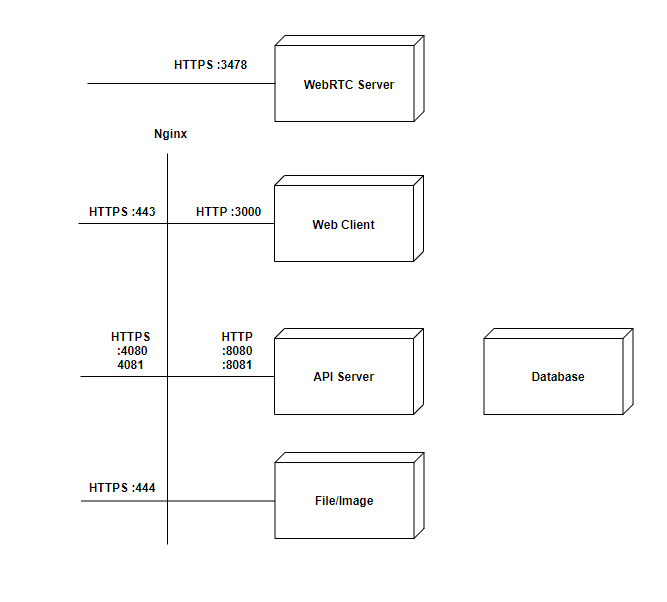
Trong phần này sẽ hướng dẫn cụ thể cách cài đặt hệ thống trên hệ điều hành CentOS 6. Khi đã nắm rõ cách hoạt động của hệ thống thì cách cài đặt trên những hệ điều hành khác cũng là tương tự.

Yêu cầu tối thiểu về phần cứng:

* Hệ điều hành Centos 6
* CPU 4 Core 8 Thread
* Ram 8G
* Có kết nối Internet

Đây là yêu cầu tối thiểu về phần cứng, sau này khi số lượng người dùngtăng buộc phải nâng cấp cấu hình của CPU và RAM.

Mô hình của hệ thống sau khi cài đặt hoàn chỉnh



Hình ‑ Mô hình sau khi triển khai hệ thống

1. **Cài đặt môi trường**
2. **Cài đặt các thư viện cơ bản và cần thiết:**

yum –y update

yum groupinstall 'Development Tools'

yum –y install git autoconf automake libtool ncurses-devel libjpeg-devel sqlite-devel libcurl-devel pcre-devel speex-devel libedit-devel

yum -y install expat-devel openssl-devel libtiff-devel libX11-devel unixODBC-devel openssl-devel python-devel zlib-devel libzrtpcpp-devel alsa-lib-devel libogg-devel libvorbis-devel perl-libs gdbm-devel uuid-devel libuuid libuuid-devel perl perl-ExtUtils-Embed php php-devel ldns-devel

1. **Cài đặt Java**

yum –y install wget

cd /opt/

wget --no-cookies --no-check-certificate --header "Cookie: gpw\_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" "https://download.oracle.com/otn-pub/java/jdk/8u201-b09/42970487e3af4f5aa5bca3f542482c60/jdk-8u201-linux-x64.tar.gz"

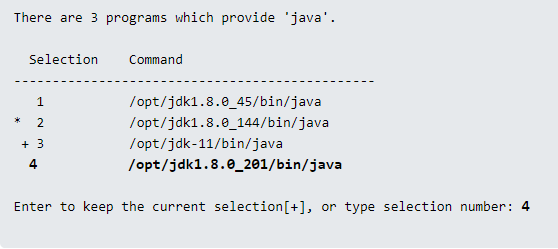
tar xzf jdk-8u201-linux-x64.tar.gz

cd jdk1.8.0\_201/

alternatives --install /usr/bin/java java /opt/jdk1.8.0\_201/bin/java 2

alternatives --config java

Màn hình sẽ hiển thị như hình dưới, ta đang cài java8 nên chọn 4



alternatives --install /usr/bin/jar jar /opt/jdk1.8.0\_201/bin/jar 2

alternatives --install /usr/bin/javac javac /opt/jdk1.8.0\_201/bin/javac 2

alternatives --set jar /opt/jdk1.8.0\_201/bin/jar

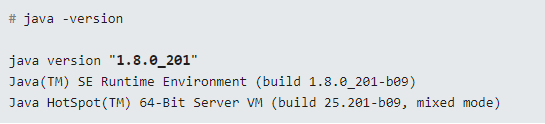
alternatives --set javac /opt/jdk1.8.0\_201/bin/javac

export JAVA\_HOME=/opt/jdk1.8.0\_201

export JRE\_HOME=/opt/jdk1.8.0\_201/jre

export PATH=$PATH:/opt/jdk1.8.0\_201/bin:/opt/jdk1.8.0\_201/jre/bin

Kiểm tra xem java cài đặt thành công chưa



Nếu màn hình hiển thị như trên thì là thành công

1. **Cài đặt Nodejs và NPM**

yum install -y gcc-c++ make

curl -sL https://rpm.nodesource.com/setup\_12.x | sudo -E bash –

yum -y install nodejs

npm install serve

1. **Cài đặt MongoDB**

Sử dụng câu lệnh để tạo file mongodb-org-4.0.repo

vi /etc/yum.repos.d/mongodb-org-4.0.repo

Thêm vào nội dung file như sau:

[mongodb-org-4.0]

name=MongoDB Repository

baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-org/4.0/x86\_64/

gpgcheck=1

enabled=1

gpgkey=https://www.mongodb.org/static/pgp/server-4.0.asc

Lưu file lại sau đó chuyển sang bước tiếp theo

yum install -y mongodb-org

service mongod start

chkconfig mongod on

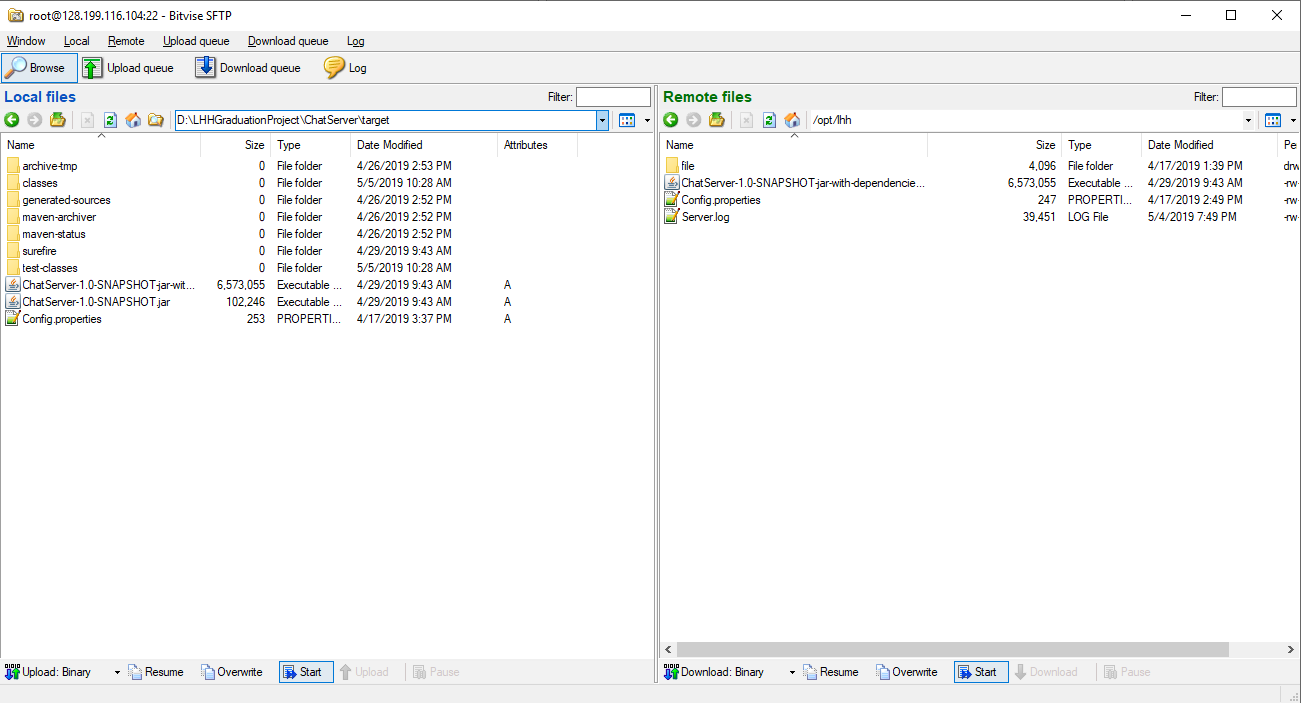
1. **Cài đặt Nginx**

yum –y install nginx

1. **Triển khai hệ thống**
2. **Triển khai Server API**

Build file jar, sau đó đưa file **ChatServer-1.0-SNAPSHOT-jar-with-dependencies.jar** và file **Config.properties** lên hệ thống. Ở đây ta sẽ đưa vào thư mục **/opt/lhh**

**Lưu ý: chỉ cần đưa file jar (đã có lib) và file properties**



Hình ‑ Triền khai Server API

Tạo file configuration để có thể khởi động server như 1 service của hệ điều hành:

vi /etc/init/ lhh-server.conf

Thêm vào file nội dung sau:

description "lhh-server"

author "hualinh1012@gmail.com"

start on runlevel [2345]

stop on runlevel [016]

chdir /opt/lhh/

exec /usr/bin/java -jar -Dfile.encoding=UTF-8 ChatServer-1.0-SNAPSHOT-jar-with-dependencies.jar

respawn

Cách thức khởi động server:

initctl start/stop/restart lhh-server

1. **Triển khai Web Client**

Upload thư mục **public, src, package-lock.json, package.json** vào thư mục **/var/www/html/chat-client**



Hình ‑ Triển khai Web Client

Download các thư viện cần thiết

npm –g install

Build project:

npm run build

Tạo file configuration để có thể khởi động như 1 service của hệ điều hành:

vi /etc/init/ lhh-web.conf

Thêm vào file nội dung sau:

description "lhh-web"

author "linhh"

start on runlevel [2345]

stop on runlevel [016]

chdir /var/www/html/chat-client

exec /usr/bin/serve -s build -p 3000 2>&1 | logger -p local1.info -t lhh-web

respawn

Cách thức khởi động server:

initctl start/stop/restart lhh-web

Để thay đổi cấu hình của Web Client bao gồm Server API IP, WebRTC Server IP thì vào file **src/constant/index.js** để thay đổi

1. **Cài đặt WebRTC Server**

Cài đặt các thư viện cần thiết:

yum install -y make gcc cc gcc-c++ wget  
yum install -y openssl-devel libevent libevent-devel mysql-devel mysqlserver

Tải và cài đặt module LibEvent

wget https://github.com/downloads/libevent/libevent/libevent-2.0.21-stable.tar.gz

tar xvfz libevent-2.0.21-stable.tar.gz

cd libevent-2.0.21-stable && ./co  
make && make install && cd..

Tải và cài đặt module TURN

wget http://turnserver.open-sys.org/downloads/v4.5.0.6/turnserver-  
4.5.0.6.tar.gz

tar -zxvf turnserver-4.5.0.6.tar.gz

cd turnserver-4.5.0.6 && ./configure  
make && make install

Cài đặt server:

vi /usr/local/etc/turnserver.conf

Thêm các dòng sau:

user=iceuser:123456 *//Tùy ý đặt*

realm= *//ip hoặc domain đang sử dụng*

listening-ip= *//ip hoặc domain đang sử dụng*

relay-ip= *//ip hoặc domain đang sử dụng*

lt-cred-mech

no-stdout-log

syslog

Tạo file configuration để có thể khởi động như 1 service của hệ điều hành:

vi /etc/init/ lhh-turn.conf

Thêm vào file nội dung sau:

description "lhh-turn"

author "linhh"

start on runlevel [2345]

stop on runlevel [016]

exec turnserver

respawn

Cách thử khởi động server

initctl start/stop/restart lhh-turn

Sau khi cài đặt và khởi động server, truy cập vào địa chỉ realm với port là 3478 hoặc 3479, nếu xuất hiện dòng chữ “Turn server” thì tức là đã thành công

VD: http://128.199.116.104:3479/

1. **Cấu hình Nginx**

Do Nginx cần sử dụng đến HTTPS nên trước tiên ta sẽ cài đặt các chứng chỉ TSL/SSL cho hệ thống. Ở đây ta sẽ tạo chứng chỉ miễn phí bằng cách sử dụng Lets Encrypt. Ngoài ra có thể dùng self-sign certificate, tuy nhiên trình duyệt sẽ báo không an toàn và chặn người dùng nên ta sẽ không sử dụng cách này.

yum -y install git python-tools python-pip

cd /opt

git clone https://github.com/letsencrypt/letsencrypt

cd letsencrypt

./letsencrypt-auto certonly -a standalone -d yourdomain.com -d www.yourdomain.com

Sau khi chạy xong câu lệnh, chứng chỉ sẽ xuất hiện tại thư mục **/etc/letsencrypt/live/yourdomain.com**

Nếu không có domain thì có thể sử dụng IP như bình thường

Tạo shortcut để Nginx có thể sử dụng được chứng chỉ

ln -s /etc/letsencrypt/live/yourdomain.com /etc/nginx/ssl

Cấu hình cho Nginx

vi /etc/nginx/nginx.conf

thêm vào trong phần http nội dung sau

upstream WEB\_CLIENT\_3000 {

least\_conn;

server localhost:3000;

}

upstream API\_8080 {

least\_conn;

server localhost:8080;

}

upstream WS\_8081 {

least\_conn;

server localhost:8081;

}

upstream MEDIA\_8090 {

least\_conn;

server localhost:8090;

}

Trong thư mục **/etc/nginx/conf.d** tạo các file sau:

**4080\_HTTPS.conf:** Chuyển hướng cổng 4080 sang cổng 8080 của Web Server

**#**4080(HTTPS)

server {

listen 4080;

server\_name \_;

ssl on;

ssl\_certificate ssl/yourdomain.com/fullchain.pem;

ssl\_certificate\_key ssl/yourdomain.com/privkey.pem;

ssl\_trusted\_certificate ssl/yourdomain.com/chain.pem;

proxy\_set\_header X-Forwarded-Port $server\_port;

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Forwarded-Host $http\_host;

proxy\_set\_header X-Forwarded-Server $host;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

add\_header 'Content-Type' 'application/json; charset=utf-8';

# HTTP

location / {

proxy\_pass http://API\_8080/;

proxy\_redirect default;

set $do\_not\_cache 0;

if ($request\_method != GET) {

set $do\_not\_cache 1;

}

if ($uri ~\* "\.(jpg|png|gif|jpeg|css|js|swf|pdf|html|htm|ico)$") {

expires 5m;

}

if ($uri !~\* "\.(jpg|png|gif|jpeg|css|js|swf|pdf|html|htm|ico)$") {

set $do\_not\_cache 1;

}

proxy\_no\_cache $do\_not\_cache;

proxy\_cache\_bypass $do\_not\_cache;

proxy\_cache\_key $scheme$host$uri$is\_args$args;

proxy\_cache\_valid 200 10m;

proxy\_cache\_valid any 1m;

proxy\_buffer\_size 8k;

proxy\_buffers 50 8k;

proxy\_busy\_buffers\_size 128k;

proxy\_temp\_file\_write\_size 128k;

proxy\_cache\_lock on;

proxy\_cache\_lock\_timeout 5s;

proxy\_cache\_use\_stale error timeout invalid\_header updating;

}

}

**4081\_WSS.conf:** Chuyển hướng tử cổng 4081 sang cổng 8081 của Web Socket

#4081(WSS)

server {

listen 4081;

server\_name \_;

ssl on;

ssl\_certificate ssl/yourdomain.com/fullchain.pem;

ssl\_certificate\_key ssl/yourdomain.com/privkey.pem;

ssl\_trusted\_certificate ssl/yourdomain.com/chain.pem;

proxy\_set\_header X-Forwarded-Port $server\_port;

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Forwarded-Host $http\_host;

proxy\_set\_header X-Forwarded-Server $host;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_read\_timeout 36000s;

# WS

location / {

proxy\_pass http://WS\_8081/;

proxy\_set\_header Host $host;

# WebSocket support

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

}

}

**443\_WEB.conf:** Chuyển hướng tử cổng 443 sang cổng 3000 của Web Client

#

# HTTPS server configuration

#

server {

listen 443 ssl http2 default\_server;

listen [::]:443 ssl;

server\_name \_;

root /usr/share/nginx/html;

ssl on;

ssl\_certificate ssl/linhh.dev/fullchain.pem;

ssl\_certificate\_key ssl/linhh.dev/privkey.pem;

ssl\_trusted\_certificate ssl/linhh.dev/chain.pem;

ssl\_session\_cache shared:SSL:10m;

ssl\_session\_timeout 10m;

ssl\_protocols TLSv1.1 TLSv1.2;

ssl\_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK';

ssl\_prefer\_server\_ciphers on;

add\_header Strict-Transport-Security max-age=15768000;

ssl\_stapling on;

ssl\_stapling\_verify on;

resolver 8.8.8.8 8.8.4.4 valid=86400;

resolver\_timeout 10;

# Load configuration files for the default server block.

# include /etc/nginx/default.d/\*.conf;

location / {

proxy\_pass http://WEB\_CLIENT\_3000/;

proxy\_redirect default;

set $do\_not\_cache 0;

if ($request\_method != GET) {

set $do\_not\_cache 1;

}

if ($uri ~\* "\.(jpg|png|gif|jpeg|css|js|swf|pdf|html|htm|ico)$") {

expires 5m;

}

if ($uri !~\* "\.(jpg|png|gif|jpeg|css|js|swf|pdf|html|htm|ico)$") {

set $do\_not\_cache 1;

}

proxy\_no\_cache $do\_not\_cache;

proxy\_cache\_bypass $do\_not\_cache;

proxy\_cache\_key $scheme$host$uri$is\_args$args;

proxy\_cache\_valid 200 10m;

proxy\_cache\_valid any 1m;

proxy\_buffer\_size 8k;

proxy\_buffers 50 8k;

proxy\_busy\_buffers\_size 128k;

proxy\_temp\_file\_write\_size 128k;

proxy\_cache\_lock on;

proxy\_cache\_lock\_timeout 5s;

proxy\_cache\_use\_stale error timeout invalid\_header updating;

}

}

**444\_FILE.conf:** Mở cổng để người dùng có thể truy cập vào file trên hệ thống

#

# HTTPS server configuration

#

server {

listen 444 ssl http2 default\_server;

listen [::]:444 ssl;

server\_name \_;

root /opt/lhh/file;

ssl on;

ssl\_certificate ssl/linhh.dev/fullchain.pem;

ssl\_certificate\_key ssl/linhh.dev/privkey.pem;

ssl\_trusted\_certificate ssl/linhh.dev/chain.pem;

ssl\_session\_cache shared:SSL:10m;

ssl\_session\_timeout 10m;

ssl\_protocols TLSv1.1 TLSv1.2;

ssl\_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK';

ssl\_prefer\_server\_ciphers on;

add\_header Strict-Transport-Security max-age=15768000;

ssl\_stapling on;

ssl\_stapling\_verify on;

resolver 8.8.8.8 8.8.4.4 valid=86400;

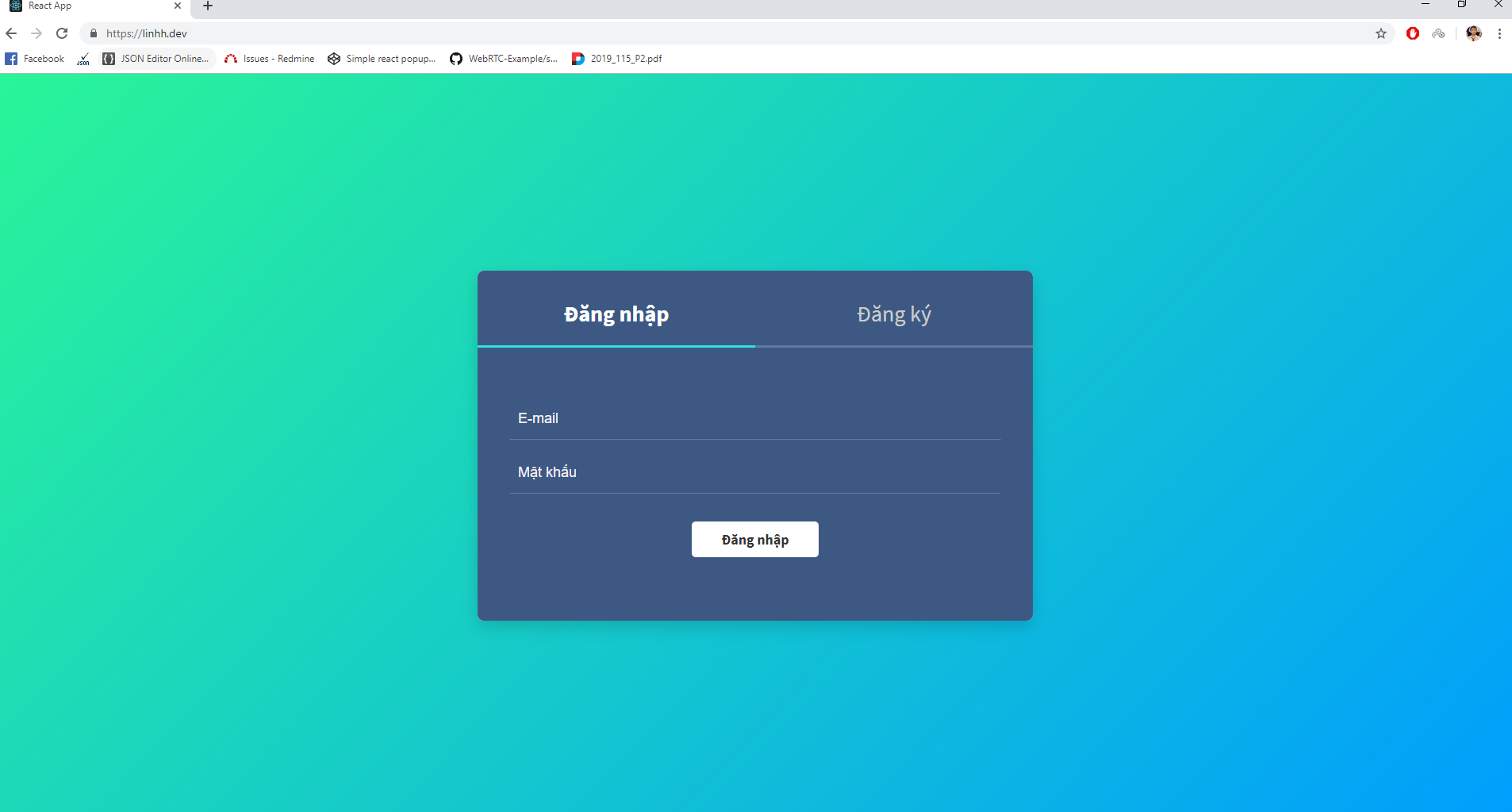
resolver\_timeout 10;

location / {

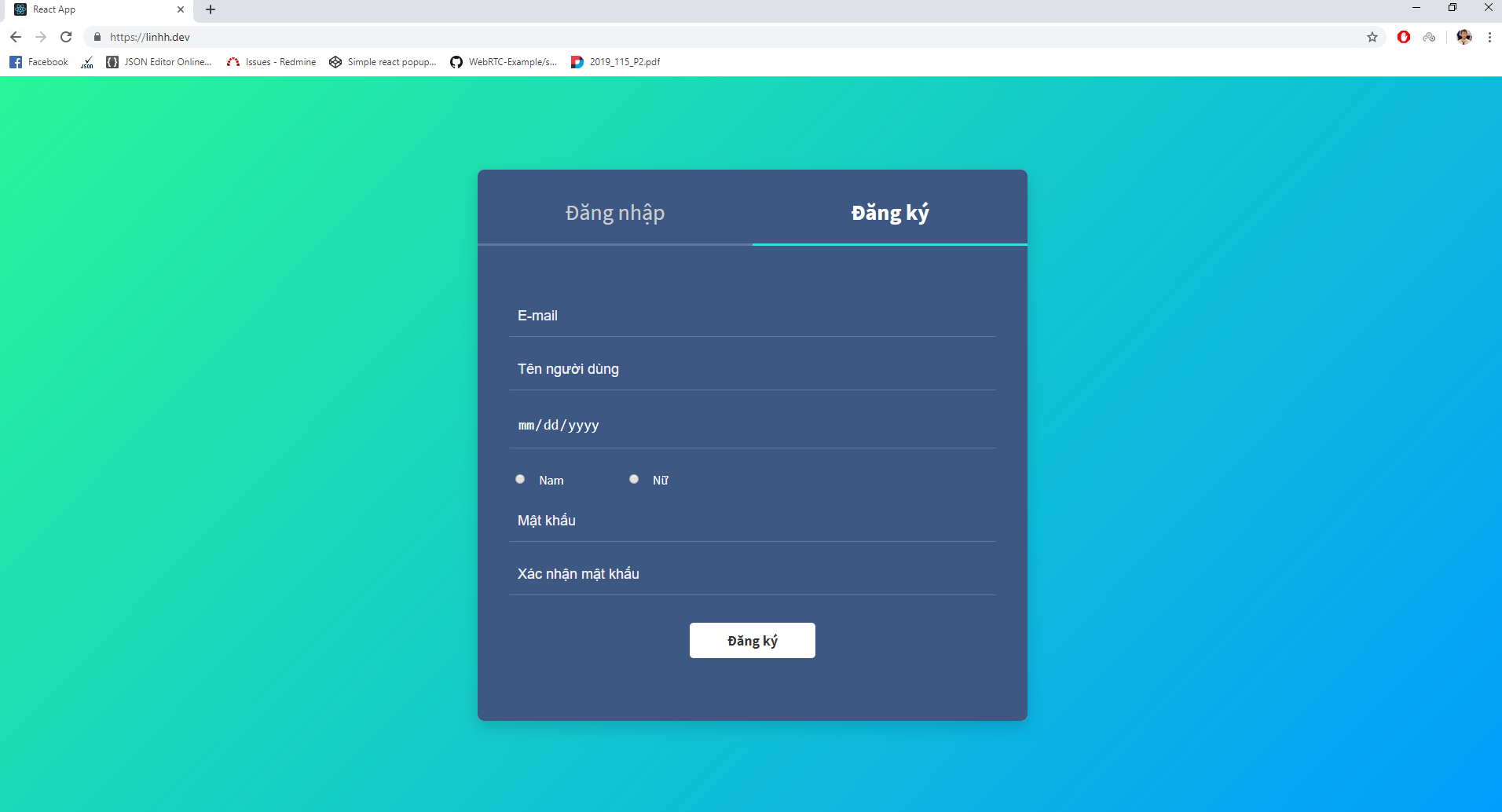
}

}

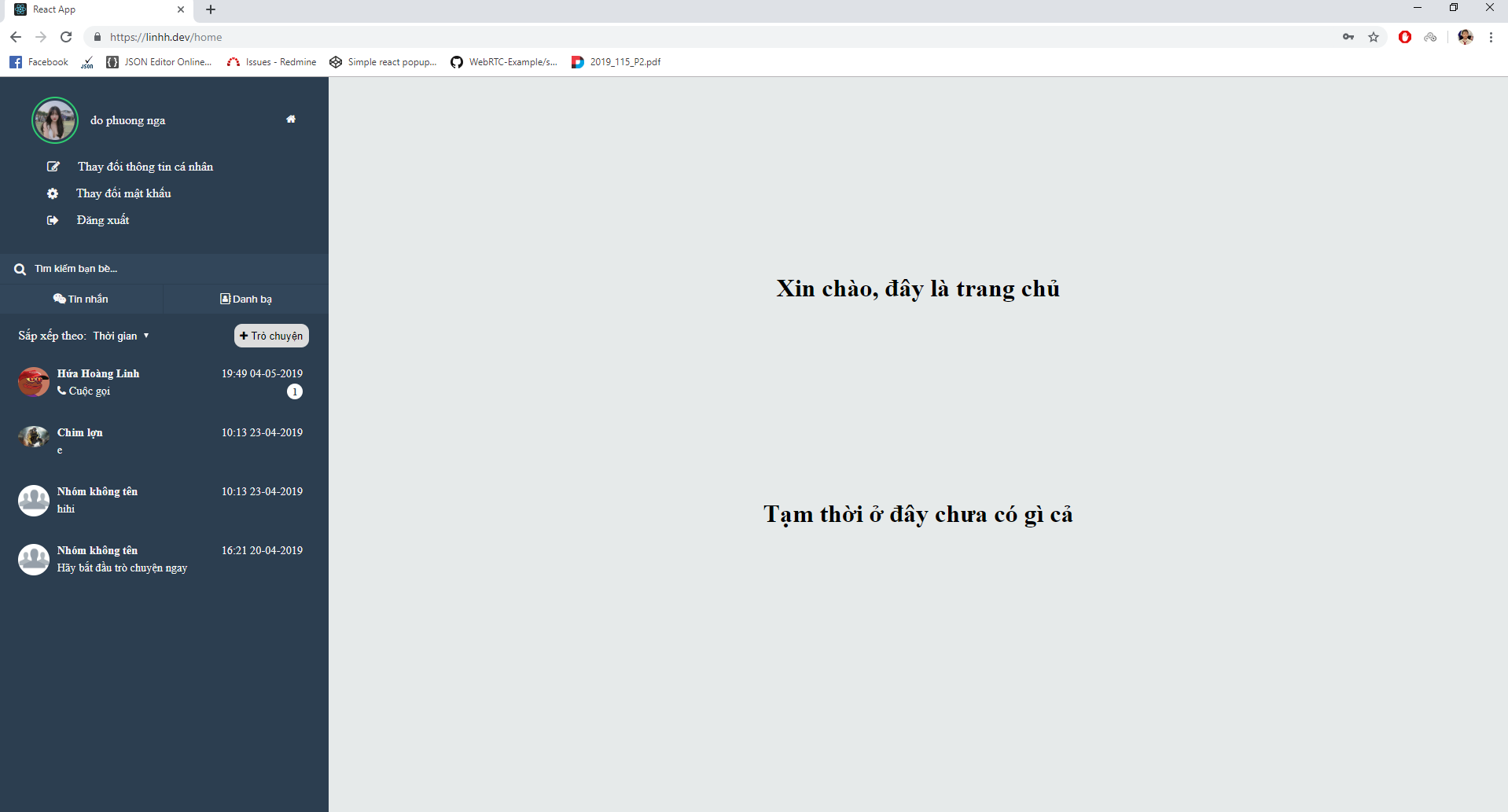
1. **Giới thiệu về hệ thống**



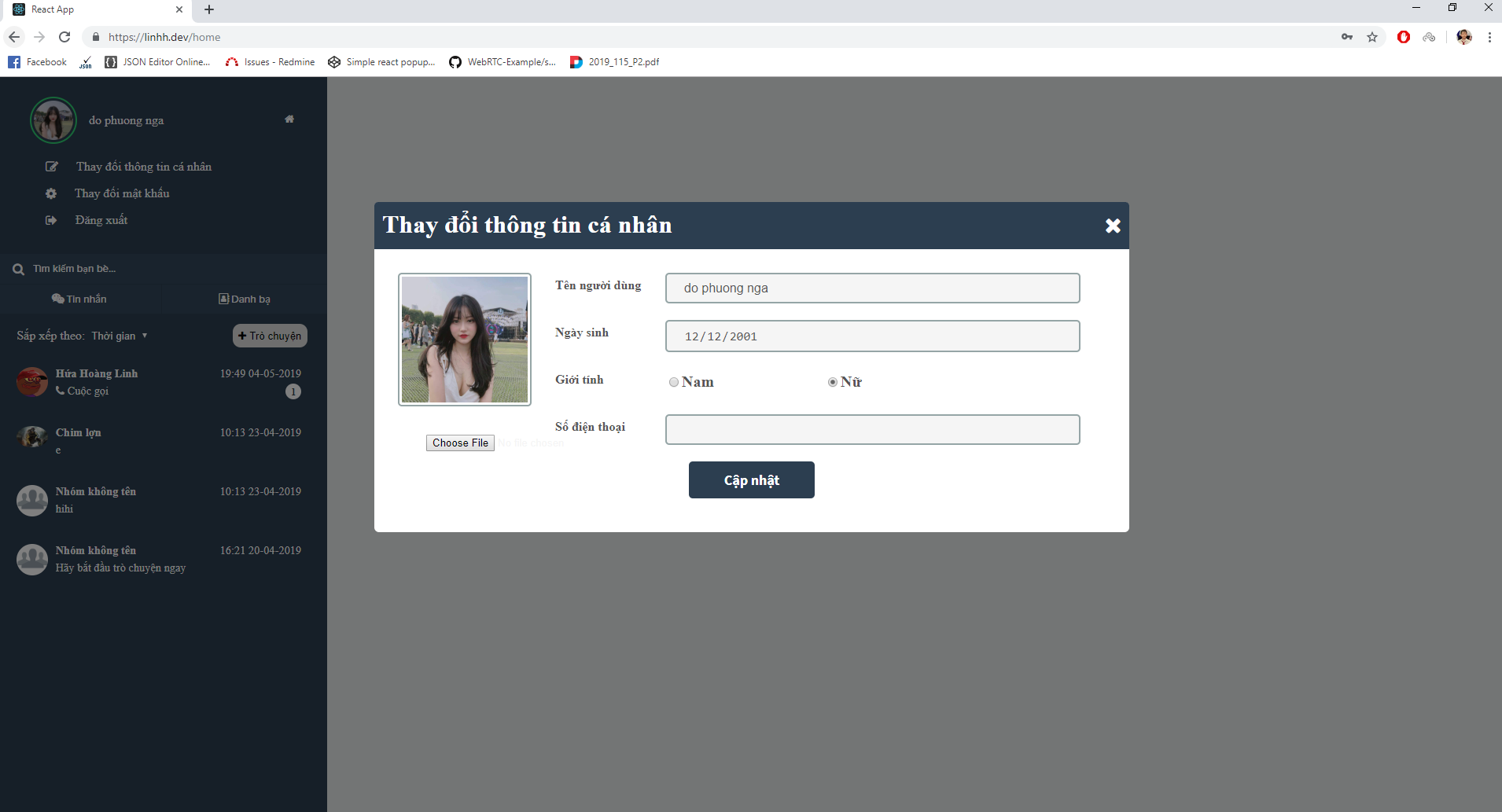
Hình ‑ Giao diện trang đăng nhập



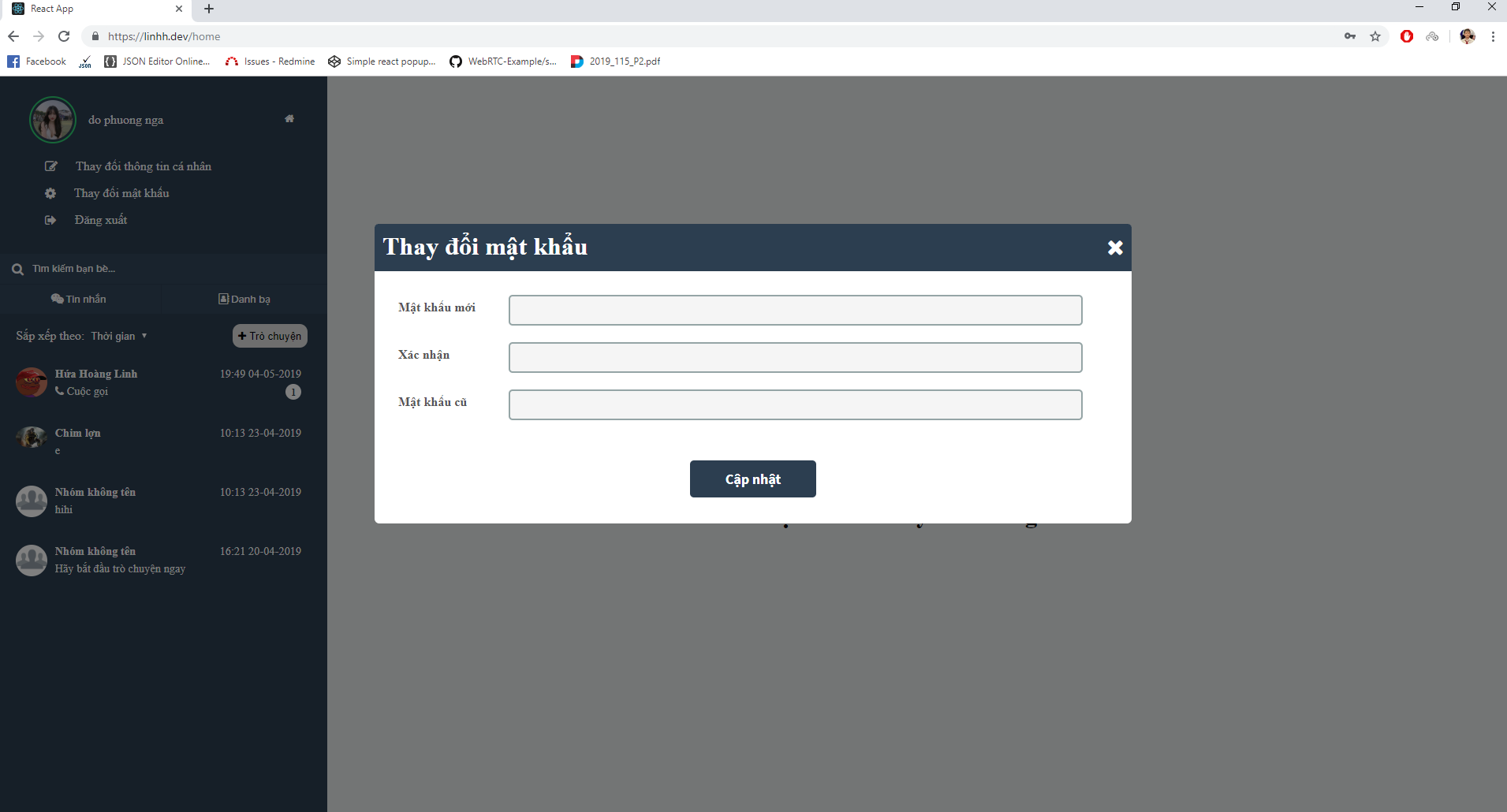
Hình ‑ Giao diện trang đăng ký



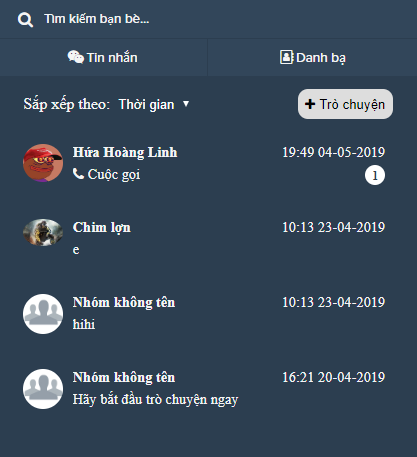
Hình ‑ Giao diện trang chủ



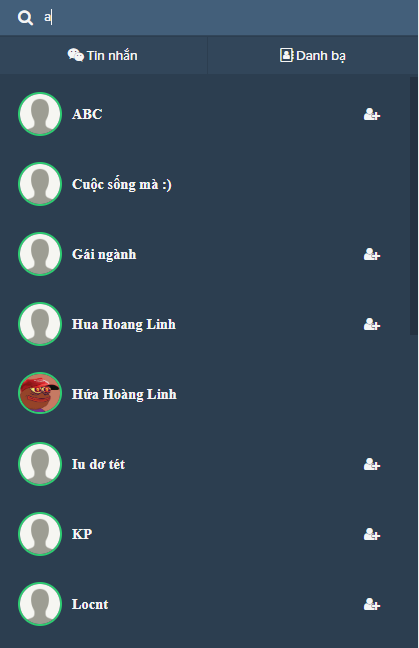
Hình ‑ Giao diện thay đổi thông tin cá nhân



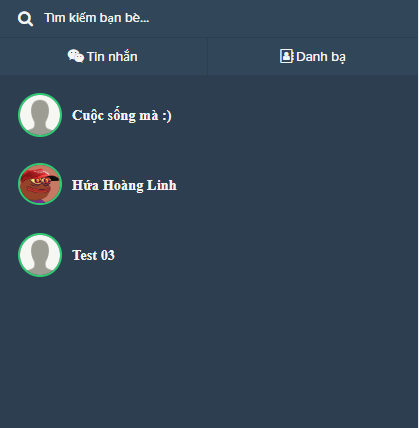
Hình ‑ Giao diện thay đổi mật khẩu



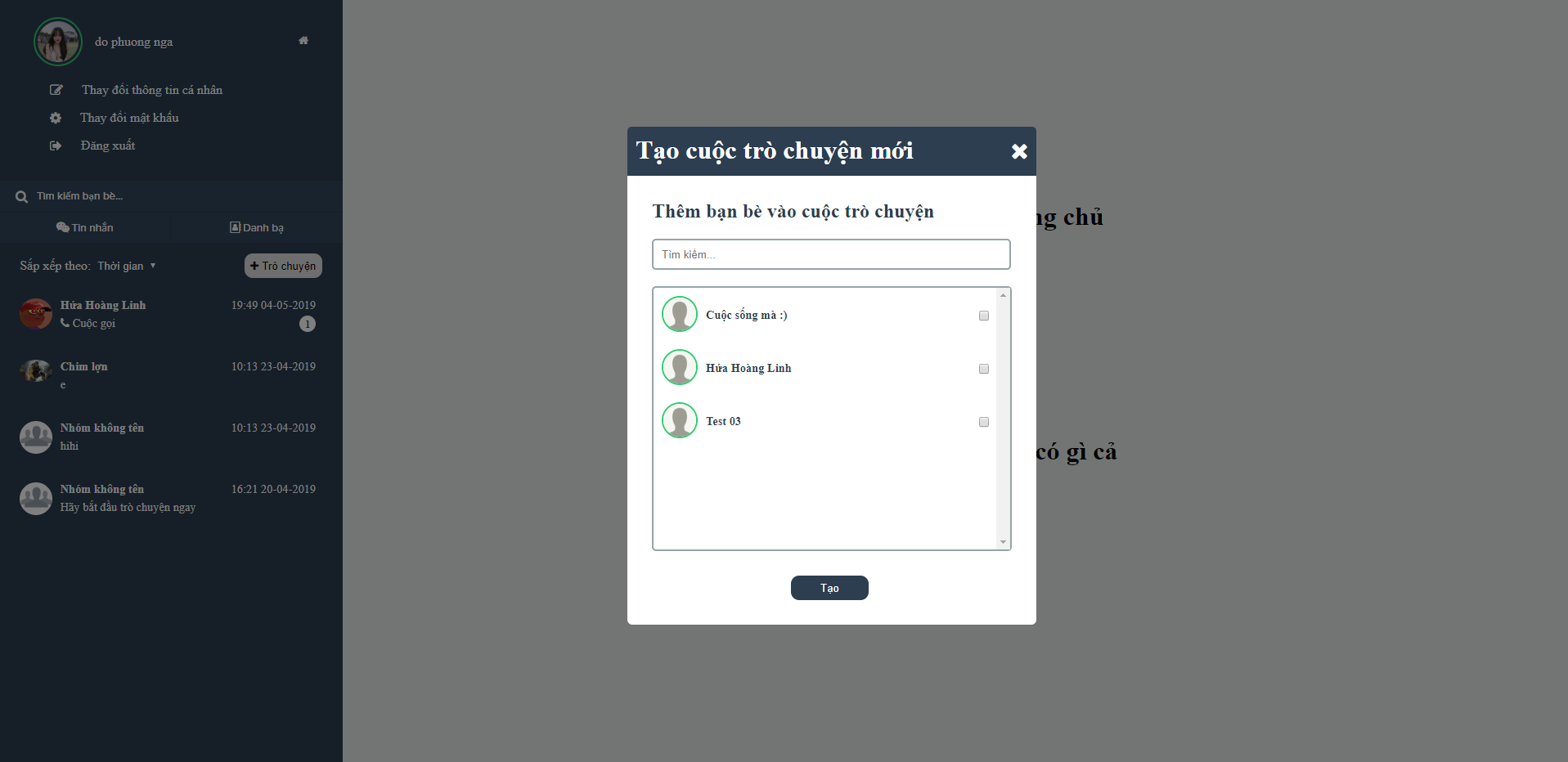
Hình ‑ Giao diện danh sách cuộc trò chuyện



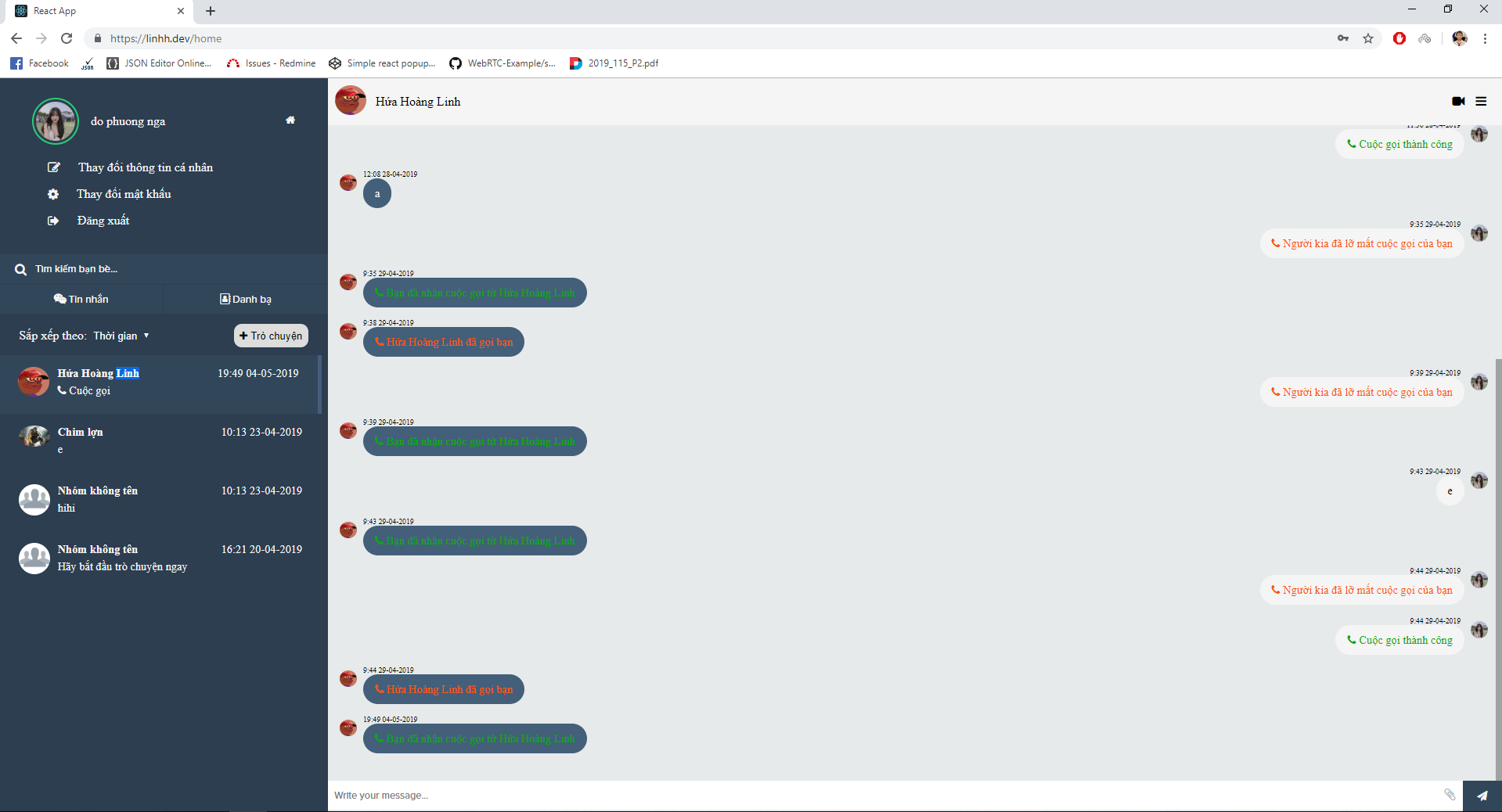
Hình ‑ Giao diện tìm kiếm bạn bè



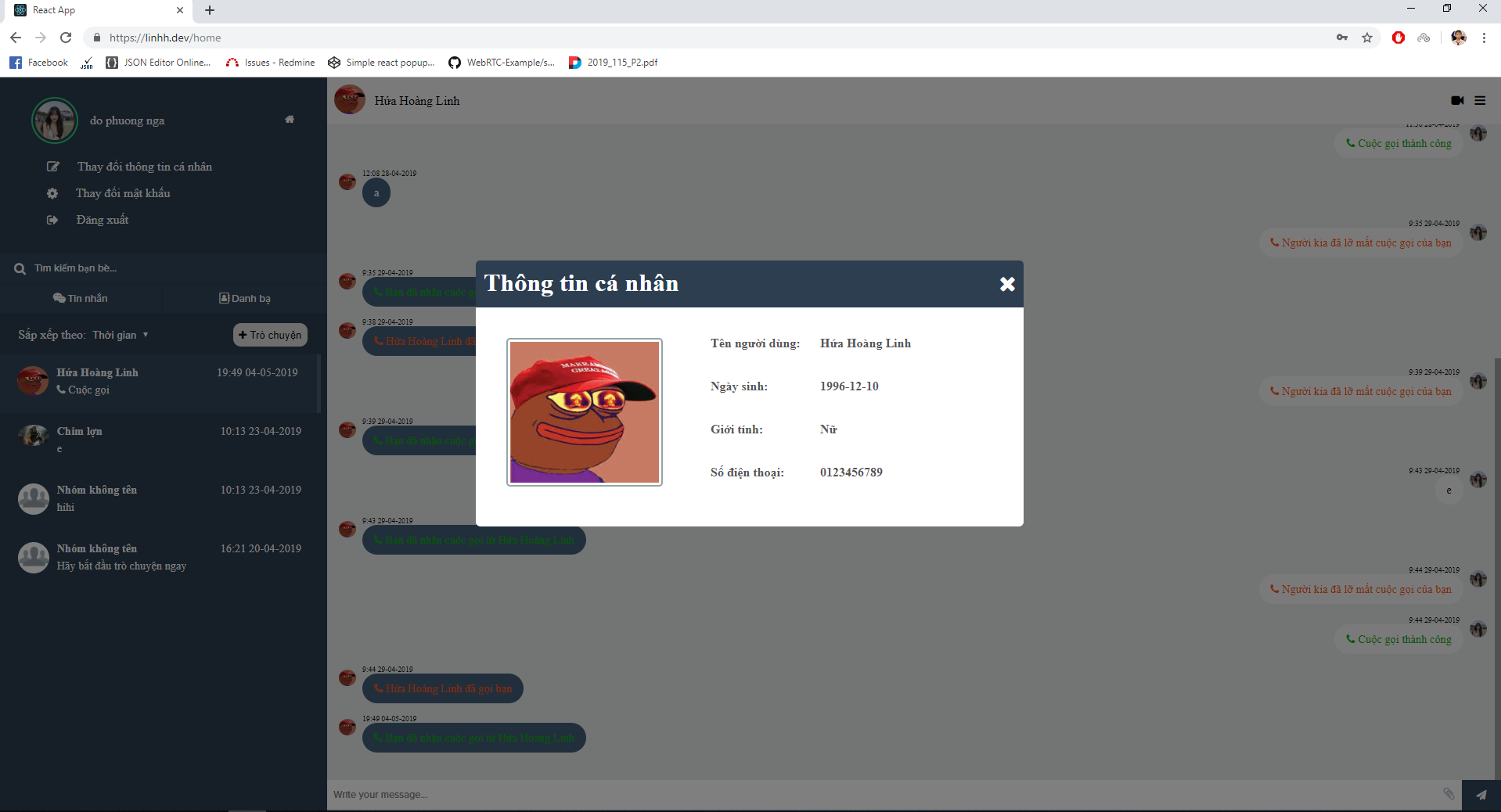
Hình ‑ Giao diện danh sách bạn bè



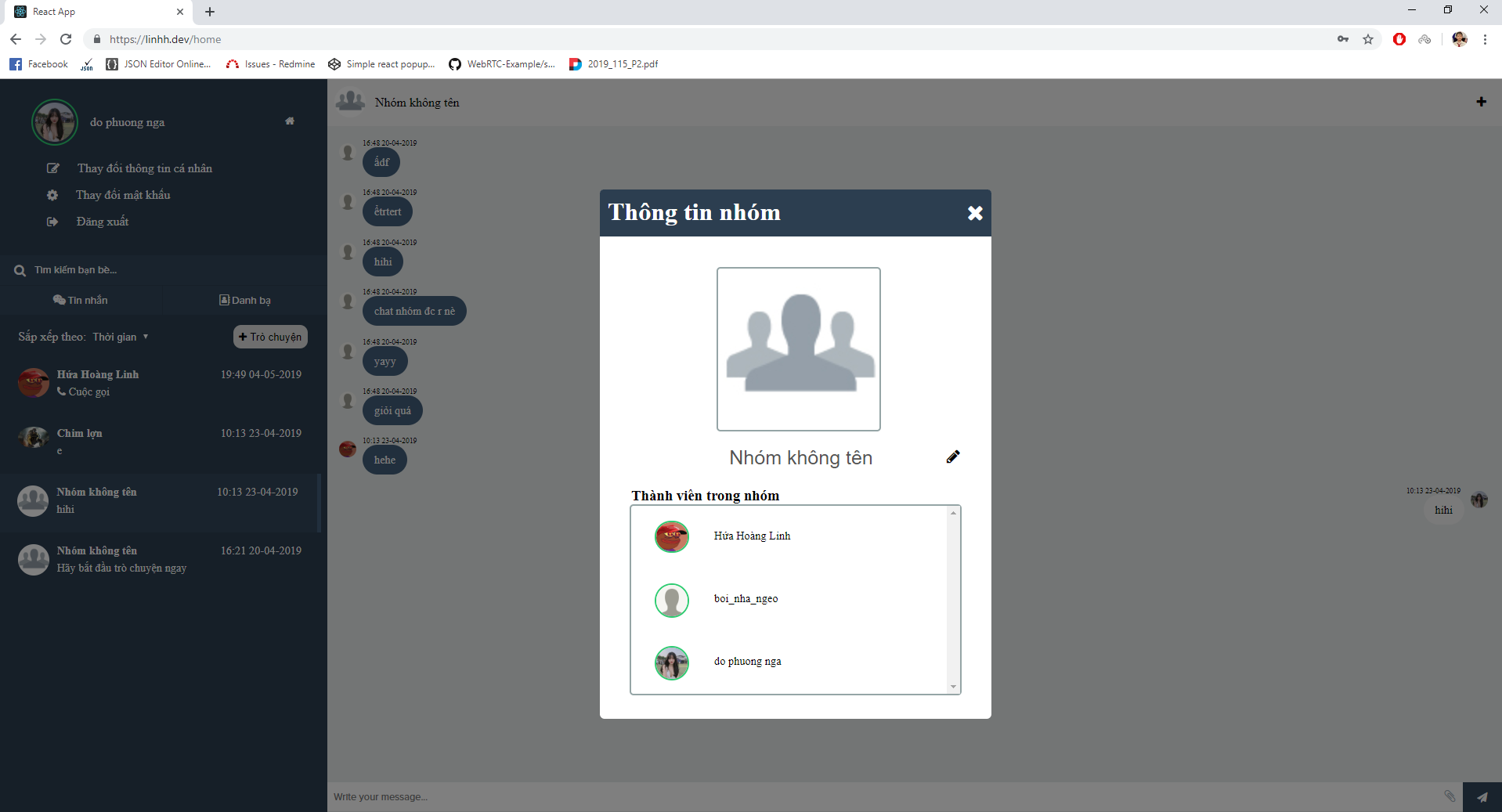
Hình ‑ Giao diện tạo cuộc trò chuyện nhóm



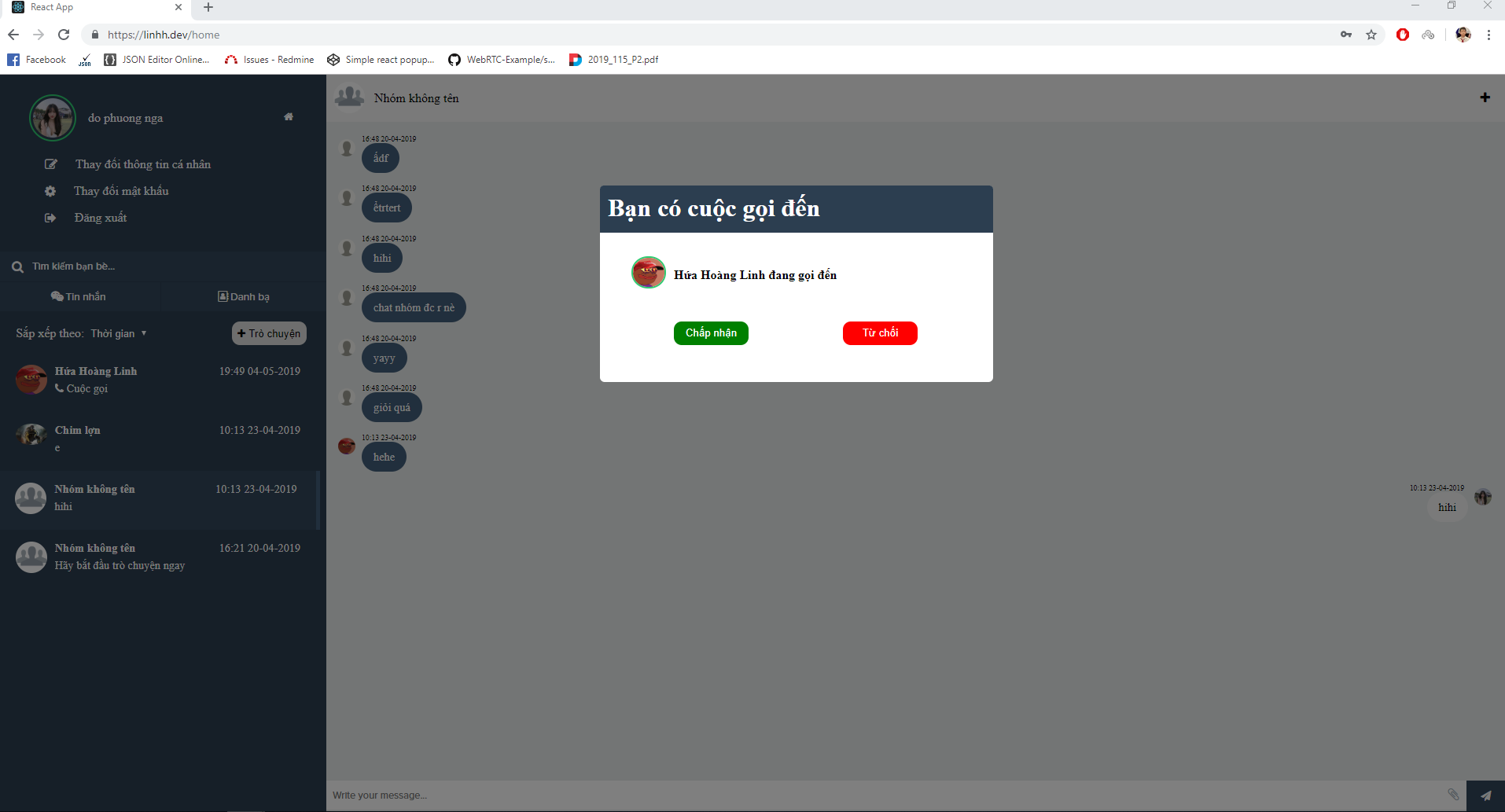
Hình ‑ Giao diện lịch sử trò chuyện



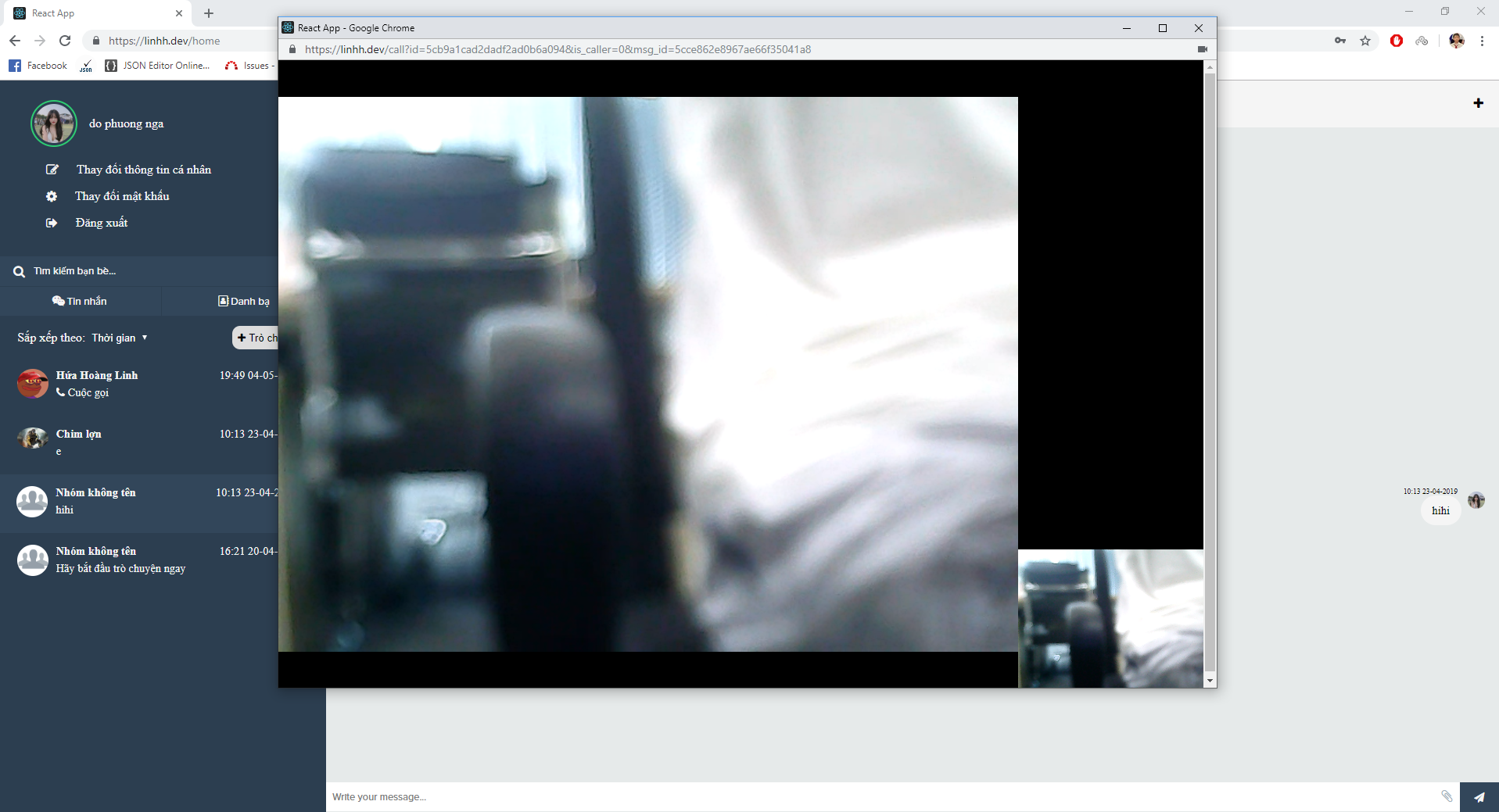
Hình ‑ Giao diện thông tin cá nhân của bạn bè



Hình ‑ Giao diện thông tin về cuộc trò chuyện nhóm



Hình ‑ Giao diện cuộc gọi đến



Hình ‑ Giao diện cuộc gọi video

# **TÀI LIỆU THAM KHẢO**

**[1] Ibrahim Tasyurt - Jetty Server Cookbook**

**[2] Tài liệu hướng dẫn chi tiết về MongoDB**

https://docs.mongodb.com/manual/

**[3] Hướng dẫn cài đặt MongoDB trên CentOS**

https://docs.mongodb.com/manual/tutorial/install-mongodb-on-red-hat/

**[4] Robin Wieruch (2018) - The Road to Learn React**

**[5] Carlos Santana Roldán - React Design Patterns and Best Practices (2nd Edition)**

**[6] Tài liệu và hướng dẫn cơ bản về WebRTC**

https://webrtc.org/start/

**[7] Rob Manson – Getting Started with WebRTC**

**[8] Tài liệu về Nginx**

http://nginx.org/en/docs/