

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH



# MẠNG MÁY TÍNH

---

Bài tập lớn  
Nhóm: 103 — Lớp: L05

## Xây dựng ứng dụng chat với WebSocket bằng Django và ReacJS

---

GVHD: Bùi Xuân Giang  
SV thực hiện: Huỳnh Quốc Phú – 1712638  
Nguyễn Duy Nguyễn – 1712380  
Trần Mạnh Hưng – 1711646

Tp. Hồ Chí Minh, Tháng 11/2019



## Mục lục

<b>1</b>	<b>Giai đoạn một</b>	<b>2</b>
1.1	Tổng quan về các chức năng của ứng dụng . . . . .	2
1.2	Các giao thức sử dụng cho từng chức năng . . . . .	2
1.2.1	WebSocket : . . . . .	2
1.2.2	REST API : . . . . .	3
<b>2</b>	<b>Giao đoạn hai</b>	<b>4</b>
2.1	Chi tiết các chứng năng của ứng dụng . . . . .	4
2.2	Kiến trúc của ứng dụng : . . . . .	8
<b>3</b>	<b>Mã nguồn chương trình</b>	<b>8</b>

## Danh sách hình vẽ

1	Cấu trúc của một Websocket. . . . .	3
2	Hoạt động của một RESTful API. . . . .	4
3	Cơ chế Login . . . . .	4
4	Cơ chế SignUp . . . . .	5
5	Dữ liệu được lưu chia thành User/SuperUser . . . . .	5
6	Giao diện Chat chính của ứng dụng . . . . .	6
7	Cơ chế tạo cuộc trò chuyện mới. . . . .	7
8	Giao diện cho việc tạo cuộc trò chuyện mới. . . . .	7
9	Sidebar . . . . .	8

Bài báo cáo là tổng hợp của hai giai đoạn theo đặc tả của bài tập lớn.  
Phần đầu là giai đoạn một với nội dung là định các chức năng của ứng dụng và định nghĩa các giao thức sử dụng trong ứng dụng.  
Phần thứ hai gồm các miêu tả chi tiết các chức năng đã thực hiện, về kiến trúc của ứng dụng kèm theo đó là các đánh giá của nhóm và source code của ứng dụng.

## 1 Giai đoạn một

### 1.1 Tổng quan về các chức năng của ứng dụng

Dựa trên các yêu cầu của bài tập lớn, nhóm đề xuất các chức năng chính cần có của ứng dụng như sau:

- Hệ thống định danh người dùng : Đăng nhập, đăng xuất, lưu thông tin người dùng để định danh phân biệt giữa các người dùng trong hệ thống.
- Hệ thống trò chuyện : Có thể trò chuyện giữa hai người với nhau, lưu lại lịch sử trò chuyện cũng như mốc thời gian của tin nhắn.
- Hệ thống tạo cuộc trò chuyện : Có thể tạo một cuộc trò chuyện mới với các người dùng khác trong cùng hệ thống.
- Hệ thống xác định trạng thái : Xác định trạng thái hoạt động của một người dùng trong hệ thống.
- Hệ thống gửi file và ảnh : trong cuộc trò chuyện có thể gửi thêm kèm theo file hay ảnh đính kèm.

### 1.2 Các giao thức sử dụng cho từng chức năng

Đối với các chức năng đã được định nghĩa ở trên, nhóm đưa ra đề xuất sử dụng hai giao thức chính để hoàn chỉnh các tác vụ trên như sau :

#### 1.2.1 WebSocket :

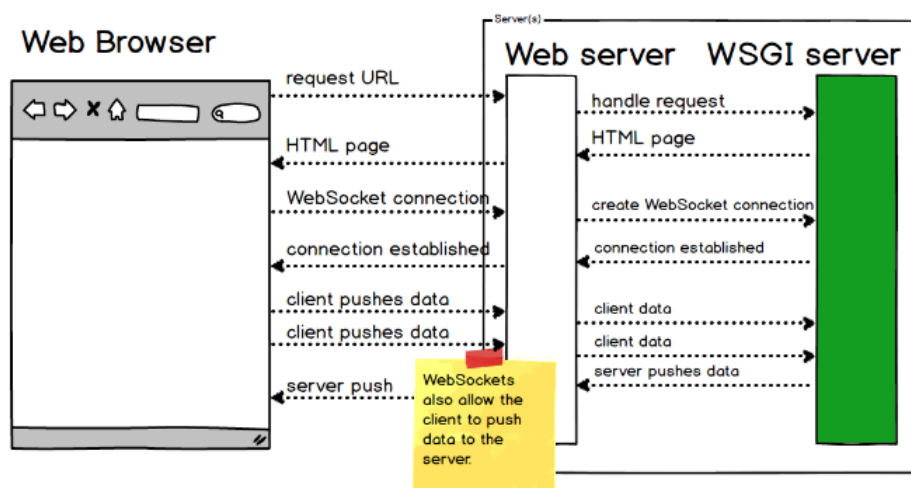
WebSocket là giao thức hỗ trợ giao tiếp hai chiều giữa client và server để tạo một kết nối trao đổi dữ liệu. Sử dụng TCP, để loại bỏ vấn đề thời hạn kết .

Giao thức WebSocket là một giao thức được sử dụng rộng rãi cho việc phát triển ứng dụng real-time do việc cung cấp giao thức giao tiếp hai chiều mạnh mẽ, có độ trễ thấp và dễ xử lý lỗi.

Các gói tin (packets) của WebSocket nhẹ hơn HTTP rất nhiều. Nó giúp giảm độ trễ của network nhiều lần.

Cấu trúc của Websoket được thể hiện như hình sau :

## WebSockets



Hình 1: Cấu trúc của một WebSocket.

WebSocket được sử dụng trong ứng dụng để hiện thực hệ thống Chat giữa người dùng với nhau và cả hệ thống gửi file và ảnh.

### 1.2.2 REST API :

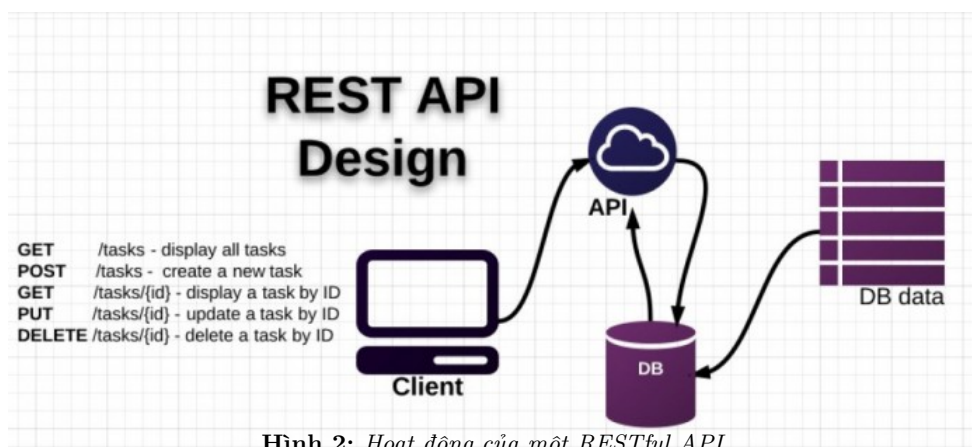
RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động...), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.

API (Application Programming Interface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML. Ở ứng dụng này, nhóm sử dụng JSON.

REST (REpresentational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE...) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API.

Hoạt động của REST API được thể hiện như hình sau :



Hình 2: Hoạt động của một RESTful API.

REST API được sử dụng trong ứng dụng để hiện thực hệ thống định danh người dùng cùng với đó là hệ thống duyệt và tạo các cuộc trò chuyện.

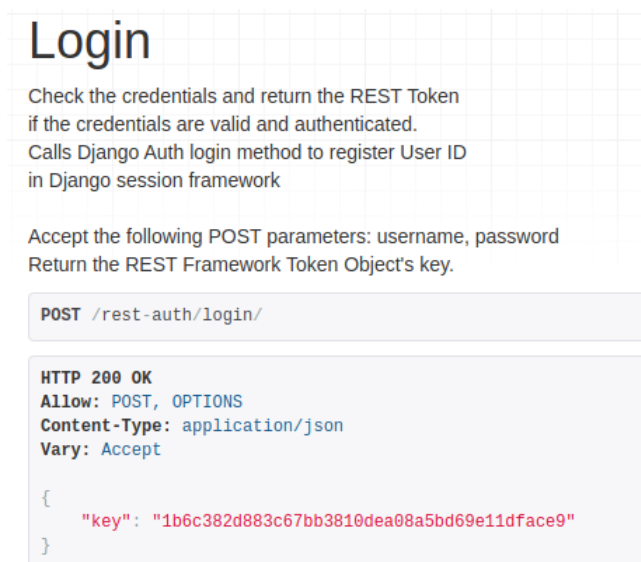
## 2 Giao đoạn hai

Trong phần này nhóm sẽ trình bày chi tiết về các chức năng, về kiến trúc của ứng dụng và tự đánh giá quá trình làm việc của từng cá nhân trong nhóm.

### 2.1 Chi tiết các chứng năng của ứng dụng

- Hệ thống định danh người dùng :

Đăng nhập, đăng ký và đăng xuất được dựa trên cơ chế REST framework trong django và được hiện thực như hình sau :



Hình 3: Cơ chế Login

## Register

POST /rest-auth/registration/

HTTP 201 Created

Allow: POST, OPTIONS

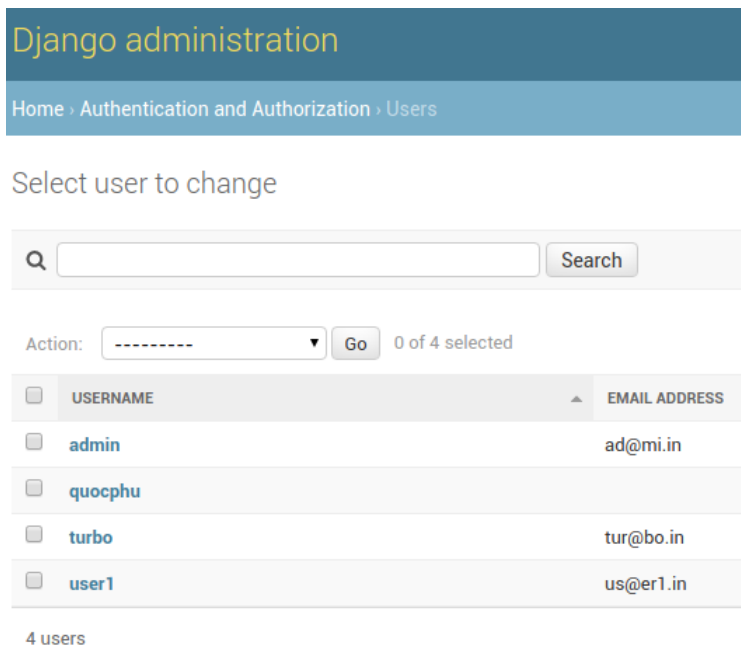
Content-Type: application/json

Vary: Accept

```
{
  "key": "fbcafe3c5e10bad428b68f550b74349912fe16de"
}
```

Hình 4: Cơ chế SignUp

Thông tin người dùng được lưu trên DataBase như sau :



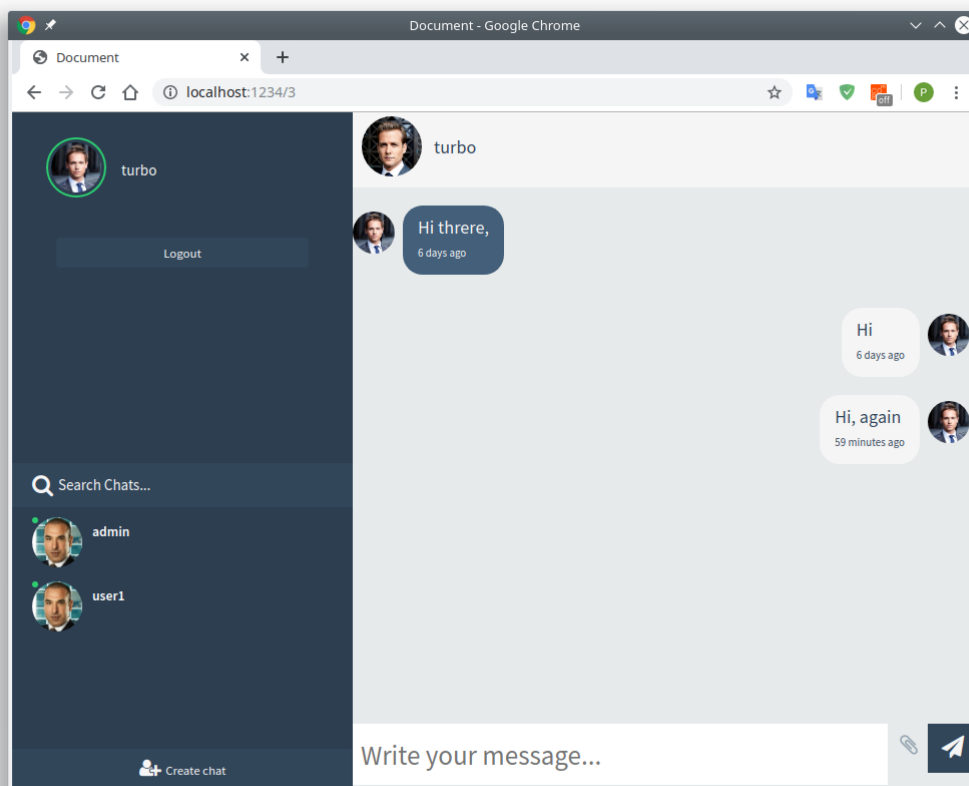
The screenshot shows the Django administration interface for managing users. The breadcrumb trail is 'Home > Authentication and Authorization > Users'. The page title is 'Django administration'. Below the title, there is a search bar and a 'Search' button. The 'Action:' dropdown is set to '-----' with a 'Go' button. The table lists 4 users: 'admin' (ad@mi.in), 'quocphu' (turbo@bo.in), 'turbo' (tur@bo.in), and 'user1' (us@er1.in). The table has columns for 'USERNAME' and 'EMAIL ADDRESS'.

USERNAME	EMAIL ADDRESS
admin	ad@mi.in
quocphu	turbo@bo.in
turbo	tur@bo.in
user1	us@er1.in

Hình 5: Dữ liệu được lưu chia thành User/SuperUser

- **Hệ thống trò chuyện :**

Có thể trò chuyện giữa hai người với nhau, lưu lại lịch sử trò chuyện cũng như mốc thời gian của tin nhắn. Được thể hiện như sau :

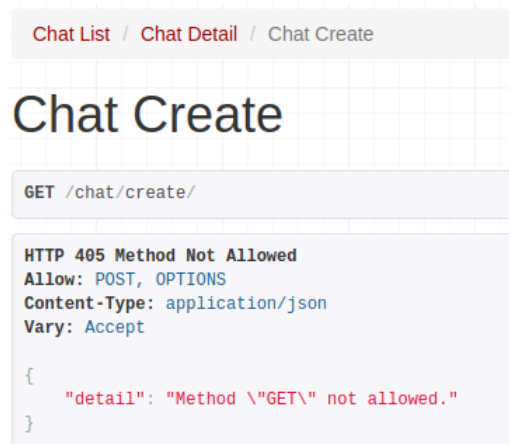


**Hình 6:** *Giao diện Chat chính của ứng dụng*

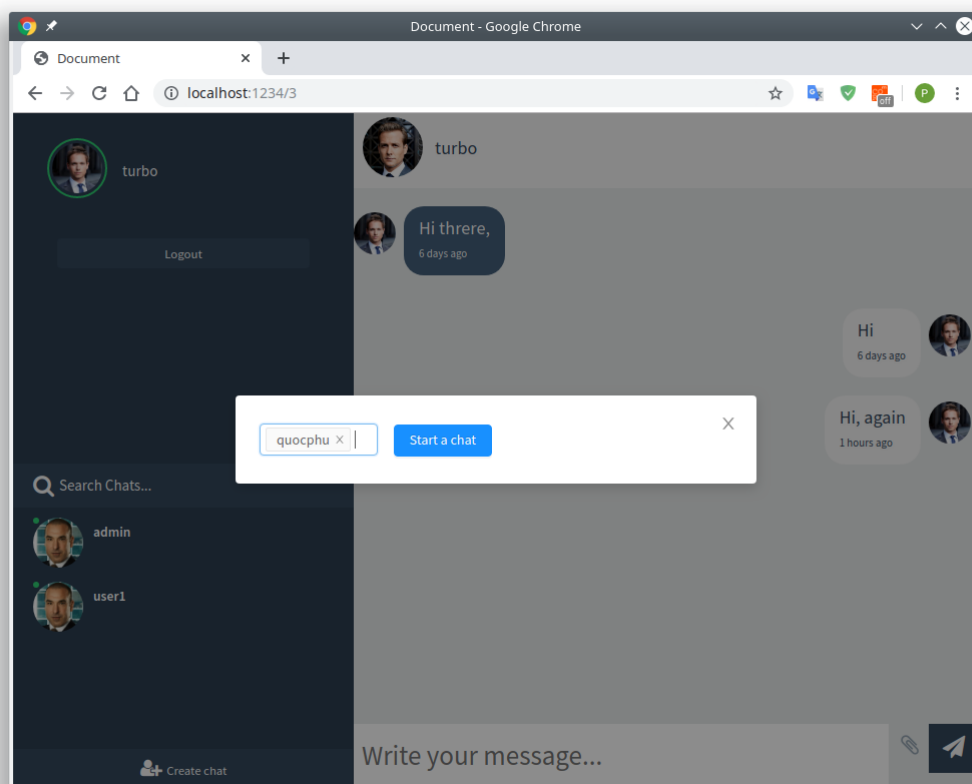
Giao diện ứng dụng chia thành 3 phần, SideBar ở bên trái, Profile ở phải trên bên phải và cuối cùng là Chat ở phía bên phải.

- **Hệ thống tạo cuộc trò chuyện :**

Có thể tạo một cuộc trò chuyện mới với các người dùng khác trong cùng hệ thống. dựa trên cơ chế REST API như hình sau :



Hình 7: Cơ chế tạo cuộc trò chuyện mới.

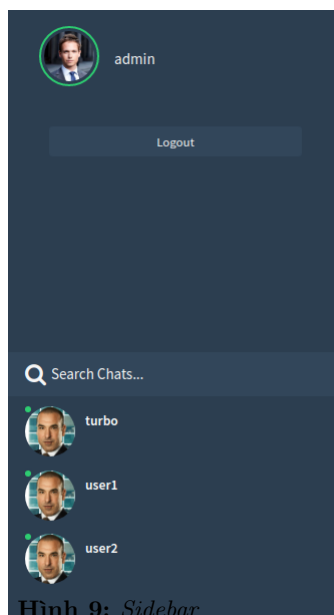


Hình 8: Giao diện cho việc tạo cuộc trò chuyện mới.

- **Hệ thống xác định trạng thái :**

Xác định trạng thái hoạt động của một người dùng trong hệ thống cũng như danh sách những người dùng mà bạn đã tạo cuộc trò chuyện cùng. như hình sau :





Hình 9: Sidebar

- **Hệ thống gửi file và ảnh :**

Được thể hiện bằng một icon gim bấm trong phần Input-message.



## 2.2 Kiến trúc của ứng dụng :

Chương trình được chia là thành hai phần tách biệt :

- **Front - End :**  
Sử dụng ReactJS cùng với Redux để hiện thực một single page application thông qua các đặc tính của React như props hay state.
- **Back - End :**  
Sử dụng Django - web framework cho Python. Bởi các lợi thế từ cấu trúc phân chia theo mô hình MVC khá rõ ràng, đến vô các số tài liệu hướng dẫn.  
Bên trong Django nhóm sử dụng Channel - là một dự án đưa Django và mở rộng khả năng của nó vượt ra ngoài HTTP - để xử lý WebSockets, giao thức trò chuyện, giao thức IoT, v.v. Nó được xây dựng trên một đặc tả Python gọi là ASGI (Asynchronous Server Gateway Interface). Giúp Django xử lý không đồng bộ với các yêu cầu gửi đến.

## 3 Mã nguồn chương trình

Cấu trúc chương trình được thể hiện như sau.

