FastChat Documentation

A.K.A

Nov 25, 2022

CONTENTS:

1 server **1**

1.1 server module 1

2 Indices and tables **5**

Python Module Index **7**

Index **9**

CHAPTER

ONE

SERVER

1.1 server module

server.**Send_group_key**(*message*, *sender_ID*)

Share the group key with a single participant of the group

Parameters

- **message** (*dictionary*) – The message contains the ID of the client whom the message has to be sent.
- **sender_ID** (*int*) – The ID of the client that requested the message

server.**accept_connections**()

Function accepts any connection and processes it based upon the type of the message recieved.

server.**add**(*message*, *sender_ID*)

Adding a member to the group.

Function checks whether the request has been made by the admin of the group. Send a message to the other members and another message to the person added.

Parameters

- **message** (*dictionary*) – Message to add a specified ID to the group.
- **sender_ID** (*int*) – The ID of the client that requested the message.

server.**add_friend**(*message*, *sender_ID*)

Add a client as a friend from the users of the application.

Parameters

- **message** (*dictionary*) – The message contains the ID of the client whom we want to add as friend.
- **sender_ID** (*int*) – The ID of the client that requested the message

server.**client_reg**(*credentials*)

Registers the clients into the application. Creates a new ID and sends that ID to the client.

Parameters **credentials** (*dictionary*) – The credentials of the client

server.**client_verify**(*credentials*)

Verifies the clients by authorising it from the servers by matching its information from the database. Closes the socket if the credentials are incorrect. Returns an acknowledgement to the client whether it was verified.

Parameters **`credentials`** (*`dictionary`*) – The credentials of the client

`server.`**`connect_server`**(*server*)

Connect to a server with the given information. Sends the server authentication message to the another server. Adds the other server to the list of others_servers

Parameters **`server`** (*`tuple`*) – The information of the server with which this server has to connect.

`server.`**`connect_servers`**()

Connect with the servers that are online by fetching their information from the database. Starts a separate thread for each server.

`server.`**`create_group`**(*message*, *sender_ID*)

Handle the create_group query from a client. Assigns a Key to the group requested by the client using the Groups table in database. Sends a confirmation message back to the client including the group ID assigned. Adds the group created into the original database.

Parameters

- **`message`** (*`dictionary`*) – Contains the Name of the group to be created.
- **`sender_ID`** (*`int`*) – Client who has sent the message

`server.`**`distribute_grp`**(*message*, *group_members*)

Function to distribute the messages of a group among the servers.

Messages are distributed among the servers based upon inverse ratio of their loads. For a server i with load L_i the number of messages it has to deliver is proportional to 1/L_i. It calculates the number of messages to be sent by individual servers and distributes them using “sent_msg” function.

Parameters

- **`message`** (*`json`*) – Message to be sent to the recipients
- **`group_members`** (*`list`*) – List of people whom the message has to be sent

Returns The function makes call to the send_msg function for different servers and calls send_members on messages it sends by itself

Return type void

`server.`**`friend_key`**(*message*, *sender_ID*)

Send encrypted symmetric key to the client_ID specified by the sender

Parameters

- **`message`** (*`dictionary`*) – The message contains the ID of the client to whom the encrypted key has to be sent.
- **`sender_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`group_image`**(*message*, *sender_ID*)

Sends a group image to all the participants of the group by calling the distribute_grp.

Parameters

- **`message`** (*`dictionary`*) – The message that has to be delivered in the group.
- **`sender_ID`** (*`int`*) – The sender's ID

`server.`**`group_message`**(*message*, *sender_ID*)

Sends a group message by calling the distribute_group function on the participants of the group.

Parameters

- **`message`** (*`dictionary`*) – The message that has to be delivered in the group.
- **`sender_ID`** (*`int`*) – The sender's ID

`server.`**`handle_client`**(*client*, *client_ID*)

Handles the Queries requested by the clients based on the message type. Recieves the messages on the socket and calls the involved function. Removes the connection with the client in case of any exception.

Parameters **`client_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`handle_server`**(*server*)

Function that handles any request from the other servers. If the request is to send messages to members of a group it calls the send_members function.

Parameters **`server`** (*`socket`*) – The recieving socket.

`server.`**`kick`**(*message*, *sender_ID*)

Kicks a client from the group specified in the message.

Checks whether the Admin of the group has made the request. It sends a kick message to the specified person and acknowledge to the other participants.

Parameters

- **`message`** (*`dictionary`*) – The message contains the ID to be kicked and the group number.
- **`sender_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`q`**(*sender_ID*)

Function ends the socket connection with the client that requested for exit.

Parameters **`sender_ID`** (*`int`*) – The ID of the client that requested to quit

`server.`**`receive_client`**(*client*, *ID*)

Recieves the client to be handled from the accept connection function. Adds the client to the dictionary of clients

Parameters

- **`client`** (*`socket`*) – The socket of the client that has to be recieved.
- **`ID`** (*`int`*) – The ID of the client that has to be handled

`server.`**`receive_server`**(*sock*, *ID*)

Recieves the server to be handled from the accept connection function. Adds the server to the dictionary of other_servers.

Parameters

- **`client`** (*`socket`*) – The socket of the client that has to be recieved.
- **`ID`** (*`int`*) – The ID of the client that has to be handled

`server.`**`send_client`**(*status*, *ID*, *msg*)

Sends the message to the client specified by the ID. If the client is online it delivers the message else stores it in the pending messages of client.

Parameters

- **`msg`** (*`dictionary`*) – The message to be sent to the recipient.
- **`ID`** (*`int`*) – Client ID whom the message has to be delivered.
- **`status`** (*`bool`*) – The status of client (online or offline)

Returns

(status = True) The function calls the send function on the socket of respective client.

(status = False)

`server.`**`send_members`**(*message*, *members*)

Sends the message to all the clients specified in the members list. Calls send_client function on each of the client present in the list.

Parameters

- **`message`** (*`dictionary`*) – The message to be sent to all the recipients present in the list.
- **`members`** (*`list`*) – The list of persons whom the message has to be sent.

`server.`**`send_pending`**(*sender_ID*)

Sends the pending messages of a user whenever the user queries for it. Updates the database and removes the pending messages of the client.

Parameters **`sender_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`single_image`**(*message*, *sender_ID*)

Send a personal image to a friend.

Parameters

- **`message`** (*`dictionary`*) – The message contains the ID of the client whom the message has to be sent.
- **`sender_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`single_message`**(*message*, *sender_ID*)

Send a personal message to a friend.

Parameters

- **`message`** (*`dictionary`*) – The message contains the ID of the client whom the message has to be sent.
- **`sender_ID`** (*`int`*) – The ID of the client that requested the message

`server.`**`update_load`**()

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

s

server, 1

INDEX

A

accept_connections() (*in module server*), 1
add() (*in module server*), 1
add_friend() (*in module server*), 1

C

client_reg() (*in module server*), 1
client_verify() (*in module server*), 1
connect_server() (*in module server*), 2
connect_servers() (*in module server*), 2
create_group() (*in module server*), 2

D

distribute_grp() (*in module server*), 2

F

friend_key() (*in module server*), 2

G

group_image() (*in module server*), 2
group_message() (*in module server*), 3

H

handle_client() (*in module server*), 3
handle_server() (*in module server*), 3

K

kick() (*in module server*), 3

Q

q() (*in module server*), 3

R

receive_client() (*in module server*), 3
receive_server() (*in module server*), 3

S

send_client() (*in module server*), 4
Send_group_key() (*in module server*), 1
send_members() (*in module server*), 4
send_pending() (*in module server*), 4
server (*module*), 1
single_image() (*in module server*), 4
single_message() (*in module server*), 4

U

update_load() (*in module server*), 4