NawaabChat

Release 2022

Yash Ruhatiya, Aryan Mathe, Yashwanth Reddy Challa

Nov 26, 2022

CONTENTS:

CHAPTER

ONE

FASTCHAT

1.1 DM module

1.2 client module

client.**HEADER_LENGTH = 10**

In order to communicate large messages over a socket, they are broken into multiple smaller messages. The first HEADER_LENGTH characters of the initial message inform the listener how many bytes of data to receive, so that they may stop listening once these many bytes have been received.

client.**addNewDM**(*MY_USERNAME*, *username*, *proxy*)

Adding a new DM to username as requested by MY_USERNAME

Parameters

- **[MY_USERNAME]** (*str*) – username of the client who requested DM
- **[username]** (*str*) – username of the other client

Returns
True for success and False for failure

Return type
bool

client.**checkSocketReady**(*socket*)

Parameters
[socket] (*socket*) – socket in question

Returns
return the socket if it is ready to be read, otherwise return false

Return type
bool

client.**connectMydb**(*dbName*)

Parameters
[dbName] (*str*) – username of the client whose database we need to connect to

Returns
cursor pointing to that user's local database

Return type
_Cursor

client.**createGroup**(*grpName*, *ADMIN*, *proxy*)

Create a new group by updating the database

Parameters

- **[grpName]** (*str*) – name of the new group
- **[ADMIN]** (*str*) – username of the creator
- **[proxy]** (*ServerProxy*) – the proxy server, used for a remote call to createGroupAtServer

client.**decryptMessage**(*message*, *cur*, *MY_USERNAME*)

Parameters

- **[message]** (*str*) – encrypted message
- **[cur]** (*_Cursor*) – cursor pointing to the user's local database
- **[MY_USERNAME]** (*str*) – username of the client in question

Returns

the decrypted message

Return type

str

client.**getAllUsers**(*MY_USERNAME*)

Parameters

[MY_USERNAME] (*str*) – username of the client whose connections we need to check

Returns

lists DM, group of all the users and groups in MY_USERNAME's connections

Return type

list, list

client.**getOwnPrivateKey**(*sender*)

Get the sender's private key from local database

Parameters

[sender] (*str*) – username of the sender

Returns

sender's private key

Return type

rsa.key.PrivateKey

client.**getOwnPublicKey**(*sender*)

Get the sender's public key from local database

Parameters

[sender] (*str*) – username of the sender

Returns

sender's public key

Return type

rsa.key.PublicKey

client.**getPrivateKey**(*group*, *sender*)

Parameters

- **[group]** (*str*) – group of which the sender is a participant
- **[sender]** (*str*) – username of the sender of the message

Returns
parameters the private key of the group

Return type
tuple

client.**getPublicKey**(*reciever*, *sender*)

Parameters

- **[reciever]** (*str*) – username of the receiver of the message
- **[sender]** (*str*) – username of the sender of the message

Returns
parameters n and e of the public key of the receiver

Return type
list

client.**goOnline**(*username*, *IP*, *PORT*)

Parameters

- **[username]** (*str*) – group of which the sender is a participant
- **[IP]** (*str*) – IP address of the server
- **[PORT]** (*int*) – PORT of the server

Returns
parameters the private key of the group

Return type
tuple

client.**handlePendingMessages**(*client_pending_socket*, *proxy*)

Handles the sending of pending messages. Called every time the client logs in.

Parameters

- **[client_pending_socket]** (*socket*) – socket belonging to the client having pending messages
- **[proxy]** (*ServerProxy*) – proxy server for rpc

client.**isAdminOfGroup**(*grpName*, *MY_USERNAME*)

Parameters

- **[grpName]** (*str*) – name of the group
- **[MY_USERNAME]** (*str*) – admin username

Returns
whether MY_USERNAME is an admin of grpName

Return type
bool

client.**isInConnections**(*MY_USERNAME*, *username*)

Parameters

- **[MY_USERNAME]** (*str*) – username of the client whose connections we need to check
- **[username]** (*str*) – username of the other client

Returns
whether username is in MY_USERNAME's connections

Return type
bool

client.**receive_message**(*data*, *proxy*)

Handle the reception of normal messages as well as the SEND_IMAGE and ADD_PARTICIPANT keywords along with updating the user-side database

Parameters

- **[data]** (*dict*) – dictionary containing all details of the message received
- **[proxy]** (*ServerProxy*) – proxy server for remote calls

client.**replace_quote**(*msg*, *fernet*)

Duplicate all occurences of both double and single quotes

Parameters

- **[msg]** (*str*) – message string
- **[fernet]** (*str*) – fernet string

Returns
return the string with duplicated quotes

Return type
str,str

client.**sendAck**(*client_socket*, *messageId*, *isImage*)

Send an acknowledgement to the server on receipt of a message over socket

Parameters

- **[client_socket]** (*socket*) – the socket that is sending the ack
- **[messageId]** (*int*) – unique id used to identify the message
- **[isImage]** (*bool*) – whether the message is an image or not

client.**unpack_message**(*client_socket*)

Receive as many bytes as specified by the header in units of 16 bytes

Parameters
[client_socket] (*socket*) – the socket that is receiving data

Returns
Dictionary of the received json data. False if any exception occured

Return type
dict/bool

1.3 interface module

1.4 loadBalancer module

1.5 performance module

performance.**latency**()

Latency in seconds by average of differences between send time and receive time

Returns
the latency in seconds

Return type
float

performance.**throughput**(*t*)

Throughput by average messages sent/received in total time

Parameters
[t] (*float*) – interval-width

Returns
the throughputs in messages/second

Return type
float,float

1.6 server module

1.7 serverDatabase module

1.8 signIn module

signIn.**handleSignIn**(*proxy*, *IP*, *PORT*)

Handles sign-in requests

Parameters

- **[proxy]** (*ServerProxy*) – proxy server for calls to checkUserName and isValidPassword
- **[IP]** (*str*) – server IP
- **[PORT]** (*int*) – server PORT

Returns
username and the created socket

Return type
str, socket

1.9 signUp module

signUp.**handleSignUp**(*proxy*, *IP*, *PORT*)

Handles sign-up requests

Parameters

- **[proxy]** (*ServerProxy*) – proxy server for remote calls
- **[IP]** (*str*) – server IP
- **[PORT]** (*int*) – server PORT

Returns

username and the created socket

Return type

str, socket

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX