# Code FE lấy từ <https://github.com/antitank19/MeetingAppFull-Youtube> branch ChangePresenceHub

# Meeting hub

## FE Connect

src\app\\_services\chat-hub.service.ts

createHubConnection(user: User, meetingId: string) {

        alert(user.token);

        logForTrack('createHubConnection(user: User, roomId: string)');

        this.chatHubConnection = new HubConnectionBuilder()

            .withUrl(this.hubUrl + 'meetinghub?meetingId=' + meetingId, {

                accessTokenFactory: () => user.token

            }).withAutomaticReconnect().build()

        this.chatHubConnection.start().catch(err => console.log(err));

## FE catch tín hiệu từ hub

#### "UserOnlineInMeeting": Thông báo có người mới vô meet

##### BE gửi

SendAsync(“UserOnlineInMeeting”, MemberSignalrDto)

##### FE bắt

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('UserOnlineInMeeting', (user: Member) => {

        logForTrack(`hubConnection.on('UserOnlineInMeeting', (user: Member) =>`);

        //code xử lí fe, chắc là đẩy username vô cái meet

            this.oneOnlineUserSource.next(user);

            this.toastr.success(user.displayName + ' has join room!')

        })

src\app\components\home\home.component.ts

this.subscriptions.add(

            this.chatHub.oneOnlineUser$.subscribe(member => {

                if (this.currentUser.userName !== member.userName) {

                    // Let some time for new peers to be able to answer

                    setTimeout(() => {

                        const call = this.myRTCPeer.call(member.userName, this.stream, {

                            metadata: { userId: this.currentMember },

                        });

                        call.on('stream', (otherUserVideoStream: MediaStream) => {

                            this.addOtherUserVideo(member, otherUserVideoStream);

                        });

                        call.on('close', () => {

                            this.videos = this.videos.filter((video) => video.user.userName !== member.userName);

                            //xoa user nao offline tren man hinh hien thi cua current user

                            this.tempvideos = this.tempvideos.filter(video => video.user.userName !== member.userName);

                        });

                    }, 1000);

                }

            })

        );

#### “OnConnectMeetHubSuccessfullyMsg” và “OnTestReceiveInvoke” dùng để test coi có connect thành công chưa. Nếu connect thành công, be sẽ gửi 1 đoạn msg thông báo là đã connect meetingHub thành công. Ông có thể log hoặc alert đoạn msg này để dễ debug

##### BE send

SendAsync(“OnConnectMeetHubSuccessfully”, msg: string);

SendAsync(“OnTestReceiveInvoke”, msg: string);

##### FE bắt

src\app\\_services\chat-hub.service.ts

//For tesOnly

        this.chatHubConnection.on('OnConnectMeetHubSuccessfully',  (msg : String) =>{

            console.log(msg);

            alert(msg);

            this.chatHubConnection.invoke("TestReceiveInvoke", "Go fuck your self")

        })

        this.chatHubConnection.on('OnTestReceiveInvoke',  (msg : String) =>{

            console.log(msg);

            alert(msg);

        })

#### “UserOfflineInMeeting” Thông báo có người rời meeting

##### BE send

SendAsync(“UserOfflineInGroupMsg”, offlineUser: MemberSignalrDto)

##### FE bắt

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('UserOfflineInMeeting', (user: Member) => {

        logForTrack(`hubConnection.on('UserOfflineInMeeting', (user: Member) =>`);

        // Sử lí code

            this.oneOfflineUserSource.next(user);

            this.toastr.warning(user.displayName + ' has left room!')

        })

src\app\components\home\home.component.ts

this.subscriptions.add(this.chatHub.oneOfflineUser$.subscribe(member => {

            this.videos = this.videos.filter(video => video.user.userName !== member.userName);

            //xoa user nao offline tren man hinh hien thi current user

            this.tempvideos = this.tempvideos.filter(video => video.user.userName !== member.userName);

        }));

#### “OnShareScreen” Thông báo có user nào đang show screen ko. Cho FE biết để chuyển màn hình chính qua lại chế độ show các cam và chế độ share screen

##### BE send

SendAsync(“OnShareScreen”, isShareScreen: bool)

##### FE bắt

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('OnShareScreen', (isShareScreen) => {

        logForTrack(`hubConnection.on('OnShareScreen', (isShareScreen) =>`);

        this.muteCamMicService.ShareScreen = isShareScreen

        })

src\app\\_services\mute-cam-mic.service.ts

private shareScreenSource = new Subject<boolean>();

shareScreen$ = this.shareScreenSource.asObservable();

// Một vài dòng code sau

set ShareScreen(value: boolean) {

    this.shareScreenSource.next(value);

  }

src\app\components\home\home.component.ts

// bật chế độ share 1 màn hình lên, nhận từ chatHub

        this.subscriptions.add(

            this.shareScreenService.shareScreen$.subscribe(hasUserSharing => {

                //Nếu có người đang share screen thì làm 1 số cái như tắt nút shareScreen,

                // lưu local storage

                //Đổi status Screen để làm đó (tao BE đọc ko hiểu)

                if (hasUserSharing) {//true = share screen

                    this.statusScreen = eMeet.SHARESCREEN

                    this.enableShareScreen = false;// enable or disable button sharescreen

                    localStorage.setItem('share-screen', JSON.stringify(this.enableShareScreen));

                } else {// false = stop share

                    this.statusScreen = eMeet.NONE

                    this.enableShareScreen = true;

                    localStorage.setItem('share-screen', JSON.stringify(this.enableShareScreen));

                }

            })

        )

#### “OnShareScreenLastUser” Thông báo tới người đang share screen là có người mới, shareScreenPeer share luôn cho người mới tới

##### BE send

SendAsync(“OnShareScreenLastUser”, new { usernameTo: string, isShare: bool })

##### FE bắt

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('OnShareScreenLastUser', ({ usernameTo, isShare }) => {

        // Xử lí code fe

        this.muteCamMicService.LastShareScreen = { username: usernameTo, isShare }

        })

src\app\\_services\mute-cam-mic.service.ts

private lastShareScreenSource = new Subject<LastUserJoinRoom>();

lastShareScreen$ = this.lastShareScreenSource.asObservable();

//Some code later

set LastShareScreen(value: LastUserJoinRoom) {

this.lastShareScreenSource.next(value);

}

src\app\components\home\home.component.ts

 // bắt đầu share stream tới user vao sau cùng từ user xuất phát stream

        this.subscriptions.add(this.shareScreenService.lastShareScreen$.subscribe(val => {

            if (val.isShare) {//true = share screen

                this.chatHub.shareScreenToUser(Number.parseInt(this.roomId), val.username, true)

                setTimeout(() => {

                    const call = this.shareScreenPeer.call('share\_' + val.username, this.shareScreenStream);

                }, 1000)

            }

        }))

#### “OnUserIsSharing” Thông báo người nào đang share screen

##### BE send

##### SendAsync(“OnUserIsSharing”, screenSharerUsername: string);

##### Fe bắt

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('OnUserIsSharing', currentUsername => {

       //Sử lý FE

        this.muteCamMicService.UserIsSharing = currentUsername

        })

src\app\\_services\mute-cam-mic.service.ts

private userIsSharingSource = new Subject<string>();

  userIsSharing$ = this.userIsSharingSource.asObservable();

//Some code

set UserIsSharing(value: string){

    this.userIsSharingSource.next(value);

  }

src\app\components\home\home.component.ts

 this.subscriptions.add(this.shareScreenService.userIsSharing$.subscribe(val => {

            this.userIsSharing = val

        }))

#### “OnMuteCamera” Thông báo tình trạng muteCam của username. Chỉ dùng để thay đổi icon Cam trên màn hình của người khác. Việc truyền Cam hay không là do peer trên FE quyết định

##### BE send

##### SendAsync(“OnMuteCamera”, new { username: String, mute: bool })

##### FE catch

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('OnMuteCamera', ({ username, mute }) => {

       //Sử lí FE

        this.muteCamMicService.Camera = { username, mute }

        })

src\app\\_services\mute-cam-mic.service.ts

private muteCamera: MuteObject;

  private muteCameraSource = new Subject<MuteObject>();

  muteCamera$ = this.muteCameraSource.asObservable();

//Some code

set Camera(value: MuteObject) {

    this.muteCamera = value;

    this.muteCameraSource.next(value);

  }

  get Camera(): MuteObject {

    return this.muteCamera;

  }

src\app\components\video-box-user\video-box-user.component.ts

this.subscriptions.add(this.muteService.muteCamera$.subscribe(data=>{

      if(this.userVideo.user.userName === data.username){

        this.enableCamera = data.mute

      }

    }))

#### “OnMuteMicro” Thông báo tình trạng muteMic của username. Chỉ dùng để thay đổi icon mic trên màn hình của người khác. Việc truyền mic hay không là do peer trên FE quyết định

##### BE send

##### SendAsync(“OnMuteMicro”, new { username: String, mute: bool })

##### FE catch

src\app\\_services\chat-hub.service.ts

this.chatHubConnection.on('OnMuteMicro', ({ username, mute }) => {

        //Sử lí FE

        this.muteCamMicService.Microphone = { username, mute }

        })

src\app\\_services\mute-cam-mic.service.ts

private muteMicro: MuteObject;

  private muteMicroSource = new Subject<MuteObject>();

  muteMicro$ = this.muteMicroSource.asObservable();

//some code later

set Microphone(value: MuteObject) {

    this.muteMicro = value;

    this.muteMicroSource.next(value);

  }

  get Microphone(): MuteObject {

    return this.muteMicro;

  }

src\app\components\video-box-user\video-box-user.component.ts

this.subscriptions.add(this.muteService.muteMicro$.subscribe(data=>{

      if(this.userVideo.user.userName === data.username){

        this.enableMicro = data.mute

      }

    }))

#### “NewMessage” Thông báo có Chat Message mới

##### BE send

SendAsync("NewMessage", MessageSignalrGetDto)

##### FE bắt

src\app\\_services\chat-hub.service.ts

private messagesThreadSource = new BehaviorSubject<Message[]>([]);

    messagesThread$ = this.messagesThreadSource.asObservable();

//Some code later

this.chatHubConnection.on('NewMessage', message => {

        logForTrack(`hubConnection.on('NewMessage', message =>`);

        if (this.messageCountStreamService.activeTabChat) {

                this.messageCountStreamService.MessageCount = 0;

            } else {

                this.messageCountStreamService.MessageCount += 1

            }

            this.messagesThread$.pipe(take(1)).subscribe(messages => {

                this.messagesThreadSource.next([...messages, message])

            })

        })

src\app\components\home\home.component.ts

this.subscriptions.add(

            this.chatHub.messagesThread$.subscribe(messages => {

                this.messageInGroup = messages;

            })

        );

## FE invoke ChatHub

# SignalR DTO

## MemberSignalrDto

public class MemberSignalrDto

{

public int Id { get; set; }

public string UserName { get; set; }

public string DisplayName => UserName;

public string FullName { get; set; }

}

## MessageSignalrGetDto

public class MessageSignalrGetDto

{

public string SenderDisplayName { get; set; }

public string SenderUsername { get; set; }

public string Content { get; set; }

public DateTime MessageSent { get; set; }

}