Installation/Initialization

Installation

Setting up Sdk with Package Manager(npm, yarn, bun ...) is pretty easy. You just need to run this command in your folder project, and it will be successfully installed

```
// install with npm
npm install axios socket.io-client chat-app-workhub --save

// install with yarn
yarn add axios socket.io-client chat-app-workhub

// or using bun
bun install axios socket.io-client chat-app-workhub
```

Initialization

(**Pay attention**) Before using it, you have to initialize the data retrieval function.

```
import {
  ChatUser,
  InitChatSocket,
  InitChatMagement,
  InitMessageReader
} from 'chat-app-demo-abc';

// auth/user config
const chatUser = InitChatUser(url, token); // url auth (http://123.30.145.67:14100/) . t
// if you using func: login, register .You can leave the token field blank

// chat realtime with socket
const chatSocket =  InitChatSocket('url_chat_soket', token);

// chat management
const apiManage = InitChatMagement('url_chat_manager', token); // url http://123.30.145.

// Message Management
const apiMessage = InitMessageReader(url, token); // url auth (http://123.30.145.67:1410
```

Last modified 1mo ago

Authentication

We provide multiple methods for authenticating accounts that can be synchronized with your system account:

- **SSO**
  We offer synchronization methods for Single Sign-On (SSO) solutions . If your system supports this solution, please contact us for integration.
  This is done using JSON Web Token (JWT) tokens and it can be easily integrated with React/React Native built in any framework or language
- **Sync with Jwt token**
  If your system uses JWT for account authentication, this is a simple way to synchronize accounts more easily. User information will be provided in the JWT, similar to the image below:

- **Username/Password**
  This is the most common way of using username/password for authentication, but it has limitations; you need to synchronize user data before using it.
  This is example:

```
// code
// param request
```

```
const param = {
  "userName": "string", //email account login
  "passWord": "string",  // password
  "pushToken": "string"// pushtoken is device token in firebase
}
const result = await ChatUser.login(param);

//response
{
  "access_token": "", // token access chat (you can using jwt_decode access_token to ge
  "refresh_token": ""
  "expires_in": 0, // expires time
  "token_type": "Bearer", // type Authorization
  "scope": "Auth offline_access"
}
```

Last modified 1mo ago

Chat Socket

Below is a summary of the methods for working with real-time chat:

Connect/ Disconnect

Below is example to using socket

```
import { ChatSocket } from 'chat-app-demo-abc';

const ChatScreen = () => {
  const chat = new ChatSocket('url_socket', token);

  useEffect(() => {
    chat.connect();
    chat.receiveMessage((message) => {
      //
    });

    return () => {
      chat.disconnect();
    };
    // eslint-disable-next-line react-hooks/exhaustive-deps
  }, []);
```

```
  const sendMessage = () => {
    const param = {
      message: 'text test',
      attachments: [],
      forwardFromRoomId: '',
      mentionIds: [],
      type: 0,
      quoteMessageId: '',
      forwardMessageId: '',
    };
    chat.sendMessage(param, id);
  };
};
```

Send Message

Sends/quote/forward a chat message to a specified group.

```
chat.sendMessage(
  {
    message: '', // message sent
    attachments: [
      {
        name: '', // name file
        desciption: '',
        id: '', // id attachments (send id = null or '')
        url: '', // url file
        typeAttachment: 1, // - video = 1,  // - audio = 2// - image = 3// - file = 4 /,
        sizeByte: 0,
        durationSeconds: 0,
        revoke: 1, // 1 revoke for sender, 2 revoke for all group
```

```
      },
    ],
    forwardFromRoomId: '',
    mentionIds: [], // list userId mention
    type: 0, //type message list in here:
    // - text = 0
    // - audio = 1
    // - file = 2
    // - sticker = 3
    // - gallery = 4
    // - video = 5
    quoteMessageId: '', // message quote id
    forwardMessageId: '', // message forward id
  },
  id
);
```

Receive Message

Listens for incoming chat messages and invokes a callback function when a new message is received.

```
chat.receiveMessage(message  => {
});
// response in socket message
{
  "attachments": [], // list file
  "forwardFromMessageId": "", // id message forward
  "forwardFromRoomId": "", // id room message forward
  "mentionIds": [], // list userId mention
  "message": "Test",
  "messageId": "", // id message
  "quoteFromMessageId": "", // id message for quote
```

```
  "roomId": "", // room id
  "senderId": "", //userid sender
  "senderProfile": {
    "avatar": "", // url avatar
    "displayname": "", // name
    "id": "", // userid sender
    "username": "" // username
  },
  "timeSend": "2023-09-28T09:10:46.783Z",
  "type": 0     //type message list in here:
                // - text = 0
                // - audio = 1
                // - file = 2
                // - sticker = 3
                // - gallery = 4
                // - video = 5
}
```

update Message

update message in group

```
// message object
 {
    message: '', // message sent
    attachments: [
      {
        name: '', // name file
        desciption: '',
        id: '', // id attachments (send id = null or '')
        url: '', // url file
```

```
      typeAttachment: 1, // - video = 1,  // - audio = 2// - image = 3// - file = 4 /,
      sizeByte: 0,
      durationSeconds: 0,
      revoke: 1, // 1 revoke for sender, 2 revoke for all group
    },
  ],
  forwardFromRoomId: '',
  mentionIds: [], // list userId mention
  type: 0, //type message list in here:
  // - text = 0
  // - audio = 1
  // - file = 2
  // - sticker = 3
  // - gallery = 4
  // - video = 5
  quoteMessageId: '', // message quote id
  forwardMessageId: '', // message forward id
}
```

Param	DataType	Required	Description
id_message_edit	String	Yes	id message
message	object	Yes	
group_id	String	Yes	id group

```
chat.updateMessage(id_message_edit,message, group_id)
```

Delete Message

func emit delete message

Param	DataType	Required	Description
id_message	String	Yes	id message
group_id	String	Yes	id group

```
chat.deleteMessage(id_message, group_id)
```

Receive Delete Message

Listens for incoming messages deleted and invokes a callback function when a new message is received.

```
chat.receiveDeleteMessage(message  => {
});
// response in socket message
{
  "attachments": [], // list file
  "forwardFromMessageId": "", // id message forward
  "forwardFromRoomId": "", // id room message forward
  "mentionIds": [], // list userId mention
  "message": "Test",
  "messageId": "", // id message
  "quoteFromMessageId": "", // id message for quote
  "roomId": "", // room id
```

```
  "senderId": "", //userid sender
  "senderProfile": {
    "avatar": "", // url avatar
    "displayname": "", // name
    "id": "", // userid sender
    "username": "" // username
  },
  "timeSend": "2023-09-28T09:10:46.783Z",
  "type": 0     //type message list in here:
                // - text = 0
                // - audio = 1
                // - file = 2
                // - sticker = 3
                // - gallery = 4
                // - video = 5
2
```

Receive Message Update

Listens for incoming messages update and invokes a callback function when a new message is received.

```
chat.receiveUpdateMessage(message  => {
});
// response in socket message
{
  "attachments": [], // list file
  "forwardFromMessageId": "", // id message forward
  "forwardFromRoomId": "", // id room message forward
  "mentionIds": [], // list userId mention
  "message": "Test",
  "messageId": "", // id message
```

```
  "quoteFromMessageId": "", // id message for quote
  "roomId": "", // room id
  "senderId": "", //userid sender
  "senderProfile": {
    "avatar": "", // url avatar
    "displayname": "", // name
    "id": "", // userid sender
    "username": "" // username
  },
  "timeSend": "2023-09-28T09:10:46.783Z",
  "type": 0     //type message list in here:
                // - text = 0
                // - audio = 1
                // - file = 2
                // - sticker = 3
                // - gallery = 4
                // - video = 5
}
```

Pin Message

Pin a chat message in group.

```
chat.pinMessage(
  {
    messageId: '', // message id pin
    groupId: '', // id group
    displayName:'', // display name sender
```

Receive Pin Message

Listens for incoming pin message

```
chat.receivePinMessage(message  => {
});
//response
{
  messageId: "60f405e13f061453fa210392",
  roomId: "60f405f20f061453ea472809",
  userId: "60f4166bb73a302dff8fbab1",
  displayName: "John Smith",
  messageContent: "This is a pinned Message"
}
```

Unpin Message

Unpin a chat message in group.

```
chat.unpinMessage(
  {
    messageId: '', // message id pin
    groupId: '', // id group
  }
);
```

Receive Unpin Message

Listens for incoming unpin message.

```
chat.receiveUnpinMessage(message  => {
});
//response
{
  messageId: "60f405e13f061453fa210392",
  roomId: "60f405f20f061453ea472809",
  userId: "60f4166bb73a302dff8fbab1"
}
```

React Message

React a chat message in group.

```
chat.reactMessage
(
  {
    messageId: '', // message id pin
    groupId: '',
    reactionCode:'' // name icon react (get by api listemoji)
  }
);
```

Receive React Message

Listens for incoming react message.

```
chat.receiveMessageReaction(message  => {
```

```
});
//response
{
 messageId: "60f405e13f061453fa210392",
 groupId: "60f405f20f061453ea472809",
 userId: "60fef815bc201276d8a28ed2",
 reactionCode: "haha"
}
```

Message Seen Status

Receive Message Seen Status

Listens for message seen.

```
chat.receiveMessageSeenStatus(message  => {
});
//response
{
   groupId: "60f405f20f061453ea472809",
   messageId: "60f405e13f061453fa210392",
   userId: "60fef815bc201276d8a28ed2"
}
```

Revoke Message

Revoke a chat message in group.

```
chat.revokeMessage
(
  {
    messageId: '', // message id
    groupId: '',
    revokeStatus: 2 // 2 - Revoke for sender 3 - Revoke all group
  }
);
```

Receive Revoke Message

Listens for incoming Revoke message.

```
chat.onRevokeMessage(message  => {
});
//response
{
  messageId: "60f405e13f061453fa210392",
  groupId: "60f405f20f061453ea472809",
  userId: "60f4166bb73a302dff8fbab1",
  revokeStatus: 2,
}
```

Revoke Attachment

Revoke a chat message in group.

```
chat.revokeAttachment(
  {
    messageId: '', // message id
    groupId: '',
    attachmentId:'',
    revokeStatus: 2 // 2 - Revoke for sender 3 - Revoke all group
  }
);
```

Receive Revoke Attachment

Listens for incoming revoke attachment message.

```
chat.onRevokeAttachment(message  => {
});
//response
{
  messageId: "60f405e13f061453fa210392",
  groupId: "60f405f20f061453ea472809",
  attachmentId:"60f405e13f061424da210402",
  userId: "60f4166bb73a302dff8fbab1",
  revokeStatus: 2,
}
```

API Reference

This page provides general definitions for the API and related methods

Default api response

```
{
   "data": "response api", // response all api
   "success": true, // status api true: success, false : have error
 }
```

Error definition

- 200: success
- 400: bad request
- 401: Unauthorized
- 404: Not Found
- 500, 501: Server Error

response error

```
{
 "error": {
   "code": "string", // code error
   "message": "string", // message
   "details": "string", // detail error
   "data": {
     "additionalProp1": "string", // It cannot determined
     "additionalProp2": "string",
     "additionalProp3": "string"
   },
   "validationErrors": [
     {
       "message": "string",
       "members": [
         "string"
       ]
     }
   ]
 }
}
```

Last modified 1mo ago

Users

Dive into the specifics of each API endpoint by checking out our complete documentation.

Login

Request Params

Param	DataType	Required	Description
userName	String	Yes	email/username account
passWord	String	Yes	
pushToken	String	No	pushtoken is device token in firebase

```
// example
const result = await ChatUser.login({userName:"",passWord:"" });
```

Responses

```
{
  "access_token": "", // token access chat (you can using jwt_decode access_token to ge
  "refresh_token": ""
  "expires_in": 0, // expires time
  "token_type": "Bearer", // type Authorization
  "scope": "Auth offline_access"
}
```

Register

Request Params

Param	DataType	Required	Description
userName	String	Yes	email/username account
passWord	String	Yes	
email	String	Yes	
phoneNumber	String	Yes	
gender	number	Yes	0: male ,1: female
avatar	String	No	url avatar
dateOfBirth	String	Yes	ex:2023-10-06T02:43:24.154Z
password	String	Yes	

isAdmin	Boolean	Yes	status is admin default false
image	Object	No	

```
// image object
 "image": { // optional
    "fileId": "string", // file id get from api upload(optional)
    "fileName": "string", // name file (optional)
    "extension": "string", // extension file
    "fileUrl": "string", // url file
    "contentType": "string"
  }
```

```
// example
const result = await ChatUser.register(param);
```

Responses

```
{
   "userId": "string", // user id
  "access_token": "", // token access chat (you can using jwt_decode access_token to get
  "refresh_token": ""
  "expires_in": 0, // expires time
  "token_type": "Bearer", // type Authorization
  "scope": "Auth offline_access"
}
```

Search Friends

Find friends

Request Params

Param	DataType	Required	Description
GroupId	String	No	
Filters	string	No	
phoneNumber	string	No	
Sorting	string	No	ASC, ADSC
SkipCount	number	No	
MaxResultCount	number	No	

```
// param is object
const result = await chatUser.searchFriends(param);
```

Responses

```
{
  "items": [
    {
      "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6", // userId
    "name": "string", // name
```

```
    "avatar": "string", // url avatar
    "email": "string", // email user
    "phoneNumber": "string", // phone user
    "gender": 0, // gender 0: male, 1: female
    "dateOfBirth": "2023-10-06T02:46:24.015Z",
    "isFriend": true, // status is friend
    "distance": 0 // distance(km)
    }
  ],
  "pageSize": 0,
  "totalCount": 0
}
```

list User by Contact

find list user by contact(phone)

Request Params

```
// param is array phone number
const param = ['09xxx', '08xxx'];
const result = await chatUser.listUserContactAdd(param);
```

Responses

```
[
  {
    id: '3fa85f64-5717-4562-b3fc-2c963f66afa6', // userId
```

```
    name: 'string', // name
    avatar: 'string', // url avatar
    email: 'string', // email user
    phoneNumber: 'string', // phone user
    gender: 0, // gender 0: male, 1: female
    dateOfBirth: '2023-10-06T02:46:24.015Z',
    isFriend: true, // status is friend
    distance: 0, // distance(km)
  },
]
```

List Friends

Get list friends

Request Params

Param	DataType	Required	Description
GroupId	String	No	
Filters	string	No	
phoneNumber	string	No	
Sorting	string	No	ASC, ADSC
SkipCount	number	No	
MaxResultCount	number	No	

```
// param is object
const result = await chatUser.listFriends(param);
```

Responses

```
{
  "items": [
    {
      "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6", // userId
    "name": "string", // name
    "avatar": "string", // url avatar
    "email": "string", // email user
    "phoneNumber": "string", // phone user
    "gender": 0, // gender 0: male, 1: female
    "dateOfBirth": "2023-10-06T02:46:24.015Z",
    "isFriend": true, // status is friend
    "distance": 0 // distance(km)
    }
  ],
  "pageSize": 0,
  "totalCount": 0
}
```

User Do Action

User action(add friend, unfriend, block , unblock)

Request Params

Param	DataType	Required	Description
id	String	Yes	userid user
actionType	number	Yes	0: add friend, 1: unfriend, 2: block , 3 unblock

```
// param is object
const result = await chatUser.userDoAction(param);
```

Responses

```
{
 "success":true //
}
```

Last modified 1mo ago

Chat Management

Get List Room

Retrieves a list of chat rooms in which a user has participated.

Request Params

Param	DataType	Required	Description
userId	String	No	userId (can be null default get by token)
groupName	String	No	Name group
tagFilter	String	No	
roomType	Number	No	0: group, 1: direct, 2: support
skip	Number	No	Point start record
take	Number	No	The number of records retrieved

```
// param is object
const param = { userId: '', skip: 0, take: 20 };
const result = await apiManage.ListRoomChat(param);
```

Responses

```
{
  "items": [
    {
      "id": "string", //id room chat
      "createdOn": "2023-09-28T07:35:34.233Z",
      "name": "string", // name room chat
      "typeOfGroup": 0, // type group: 0: group, 1: direct, 2: support
      "topic": "string", // topic group(canbe null)
      "description": "string", // desc group (can be null)
      "image": "string", // url image room
      "ownerId": "string", // id user owner group
      "participantCount": 0, // total participant
      "lastMessage": {
        "id": "string", // id message
        "message": "string",
        "userId": "string", // userid send send
        "userName": "string", // name sender
        "fullName": "string", // full name sender
        "nickName": "string", //
        "avatar": "string",// url avata
        "time": "2023-09-28T07:35:34.233Z",
        "attachments": [
          {
            "name": "string", // name file
```

```
      "desciption": "string", //
      "id": "string", // id attachments
      "url": "string", // url file
      "urlPreView": "string",
      "typeAttachment": 1, //  // - video = 1,  // - audio = 2// - image = 3// - 
      "messageId": "string",
      "sizeByte": 0, // size file
      "durationSeconds": 0,
      "revoke": 1 // 1 revoke for sender, 2 revoke for all group
    }
  ],
  "type": 0,
  "isEdited": true,// status edit
  "lastEditTime": "2023-09-28T07:35:34.233Z",
  "isDeleted": true // status delete
},
"participantInfo": { // Only available for one-on-one chats This is information ab
  "userId": "string", // userId
  "userName": "string", // username
  "fullName": "string", // fullname
  "nickName": "string", // nickname
  "avatar": "string", // url avata
  "onlineStatus": 0, //status online 0 offline, 1 online
},
"hasUnreadMessage": true, // status unread message
"unreadCount": 0,// total message unread
"isFavorite": true, // status favorite
"tags": [
  "string"
], // list tags group
"notifyStatus": true, // status get notification
"leaveGroupHistories": [
  {
```

```
          "participant": "string", // userid leave group
          "timeLeave": "2023-09-28T07:35:34.233Z"
        }
      ]
    }
  ],
  "totalCount": 0
}
```

Search Room Chat

Searches for chat rooms based on specified criteria.

Request Params

Param	DataType	Required	Description
roomType	Number	No	0,1,2
filter	String	Yes	String filter
skip	Number	No	Point start record
take	Number	No	The number of records retrieved

```
// param is object
const result = await apiManage.searchRoomChat(param);
```

Responses

```
{
  "items": [
    {
      "id": "string",// id room
      "name": "string", // name room
      "roomType": 0, // type room default =0 (group)
      "topic": "string",
      "image": "string", // url
      "participants": [
        "string"
      ],
      "participantCount": 0 // total participant
    }
  ],
  "totalCount": 0
}
```

Get Total Unread

```
// param is object
const  result = await  apiManage.getTotalUnread();
// response
{"DIRECT": 0, "GROUP": 0, "SUPPORT": 0} // total message unread by type group
```

Group Participants

Request Params

Param	DataType	Required	Description
skip	Number	Yes	Point start record
take	Number	Yes	The number of records retrieved

```
// param is object
const result = await apiManage.groupParticipants(groupId, param);
```

```
{
  "items": [
    {
      "userId": "string", // userid
      "userName": "string", // username account
      "surName": "string", // surname
      "name": "string", // name user
      "nickName": "string", // nickname
      "avatar": "string", // url avata
      "onlineStatus": 0, // status online
      "isAdmin": true. // status roler
    }
  ],
  "totalCount": 0
}
```

Get Info Room Chat

Request Params

Param	DataType	Required	Description
groupId	String	Yes	Id group chat

```
// param is object
const result = await apiManage.getInfoRoomChat(groupId);
```

Responses

```
{
  "id": "string", // id group chat
  "createdOn": "2023-09-28T08:28:16.072Z",
  "name": "string", // name group
  "roomType": 0, //deprecated
  "typeOfGroup": 0, // type of group default 0 , 0: group, 1: direct, 2: support
  "topic": "string",
  "description": "string", // desc room
  "image": "string", // url room
  "ownerId": "string", // id owner
  "participants": [
    "string"
  ],
  "participantCount": 0,
  "lastMessage": {
    "id": "string",
    "message": "string",
    "userId": "string",
```

```
  "userName": "string",
  "fullName": "string",
  "nickName": "string",
  "avatar": "string",
  "time": "2023-09-28T08:28:16.072Z",
  "attachments": [
    {
      "name": "string",
      "desciption": "string",
      "id": "string",
      "url": "string",
      "urlPreView": "string",
      "typeAttachment": 1,
      "messageId": "string",
      "sizeByte": 0,
      "durationSeconds": 0,
      "revoke": 1
    }
  ],
  "type": 0,
  "isEdited": true,
  "lastEditTime": "2023-09-28T08:28:16.072Z",
  "isDeleted": true
},
"participantInfo": {
  "userId": "string",
  "userName": "string",
  "fullName": "string",
  "nickName": "string",
  "avatar": "string",
  "onlineStatus": 0
},
"unreadCount": 0,
```

```
  "hasUnreadMessage": true, // status unread message. true have message unread
  "isLinkEnabled": true,
  "inviteLinkKey": "string", // url invite
  "labelAllows": [
    "string"
  ],
  "statusAllows": [
    "string"
  ], // list status allows
  "sexAllows": [
    "string"
  ], // list sex allows
  "maxAgeAllow": 0, // age max allow
  "minAgeAllow": 0, // age min allow
  "isPrivate": true, // status prive
  "localtionAllows": [
    "string"
  ], // list location allow
  "isAdminApprove": true // status who can approve join
}
```

Create Room Chat

Request Params

Param	DataType	Required	Description
name	String	Yes	Name group
roomType		No	Deprecated

typeOfGroup	Number	Yes	Type group 0,1,2, default 0
topic	String	No	
description	String	No	
image	String	No	url image group
ownerId	String	Yes	id user owner group
participants	String	Yes	List userid
tags	String	No	
sendDefaultMessage	Boolean	No	Status send default message
defaultMessageContent	String	No	Message default

```
// param is object
const result = await apiManage.createRoomChat(param);
```

Responses

```
{
  "id": "string", // id group chat
}
```

Update Info Room

Request Params

Param	DataType	Required	Description
name	String	Yes	Name group
topic	String	No	
description	String	No	
image	String	No	

```
// param is object
const result = await apiManage.updateInfoRoom(param);
// response
// success: true or 200 in status
```

Delete Chat Room

Request Params

Param	DataType	Required	Description
roomId	String	Yes	Room id

```
// param is object
const result = await apiManage.deleteChatRoom(roomId);
// response
// success: true or 200 in status
```

Join Chat Group

Request Params

Param	DataType	Required	Description
groupId	String	Yes	
userId	String	Yes	

```
const result = await apiManage.joinChatGroup(groupId, userId);
// response
// success: true or 200 in status
```

Invite Chat Group

Request Params

Param	DataType	Required	Description
groupId	String	Yes	Id Room

userId	Array string	Yes	List userId invite

```
// param is object example:
{
  "groupId": "string", // id room
  "userId": [
    "string"
  ] // list userId invite
}
const result = await apiManage.inviteChatGroup(param);
// response
// success: true or 200 in status
```

Remove Users Chat Group

Request Params

Param	DataType	Required	Description
groupId	String	Yes	Id Room
userId	String	Yes	

```
// param is object
const result = await apiManage.removeUserChatGroup(param);
// response
// success: true or 200 in status
```

Set favorite Group

Request Params

Param	DataType	Required	Description
id	String	Yes	group id
isFavorite	Boolean	Yes	

```
// param is object
const result = await apiManage.setFavorite(param);
// response
// success: true or 200 in status
```

Get favorite Group

Request Params

Param	DataType	Required	Description
skip	Number	Yes	default: 0
take	Number	Yes	default: 20

```
// param is object
const result = await apiManage.setFavorite(param);



// response
// success:
{
  "items": [
    {
      "id": "string",
      "createdOn": "2023-11-01T07:23:23.298Z",
      "name": "string",
      "roomType": 0,
      "typeOfGroup": 0,
      "topic": "string",
      "description": "string",
      "image": "string",
      "ownerId": "string",
      "participantCount": 0,
      "lastMessage": {
        "id": "string",
        "message": "string",
        "userId": "string",
        "userName": "string",
        "fullName": "string",
        "nickName": "string",
        "avatar": "string",
        "time": "2023-11-01T07:23:23.298Z",
        "attachments": [
          {
            "name": "string",
            "desciption": "string",
```

```
        "id": "string",
        "url": "string",
        "urlPreView": "string",
        "typeAttachment": 1,
        "messageId": "string",
        "sizeByte": 0,
        "durationSeconds": 0,
        "revoke": 1
      }
    ],
    "type": 0,
    "isEdited": true,
    "lastEditTime": "2023-11-01T07:23:23.298Z",
    "isDeleted": true
  },
  "participantInfo": {
    "userId": "string",
    "userName": "string",
    "fullName": "string",
    "nickName": "string",
    "avatar": "string",
    "onlineStatus": 0
  },
  "hasUnreadMessage": true,
  "unreadCount": 0,
  "isFavorite": true,
  "tags": [
    "string"
  ],
  "notifyStatus": true,
  "leaveGroupHistories": [
    {
      "participant": "string",
```

```
          "timeLeave": "2023-11-01T07:23:23.298Z"
        }
      ]
    }
  ],
  "totalCount": 0
}
```


Set Mute User

Request Params

Param	DataType	Required	Description
groupId	String	Yes	group id
isMute	Boolean	Yes	status mute , unmute
userId	String	Yes	

```
// param is object
const result = await apiManage.setFavorite(param);
// response
// success: true or 200 in status
```

Message Management

Chat History

Retrieves the chat history of a group.

Request Params

Param	DataType	Required	Description
before	String	No	Messageid before
after	String	No	Messageid after
isPined	Bolean/null	No	Boolean, true message pined. Null get all
skip	Number	Yes	Point start record default: 0
take	Number	Yes	The number of records retrieved default: 10

```
// param is object
const param = { userId: '', skip: 0, take: 20 };
const result = await apiMessage.chatHistory(roomid, param);
```

Responses

```
{
  "skip": 0,
  "take": 0,
  "messages": [
    {
      "id": "", // id message
      "messageContentRaw": "",// content message
      "message": "string",// content message
      "timeSend": "2023-09-28T09:08:34.878Z",
      "senderId": "string", // userId sender
      "roomId": "string", // room id
      "mentionsIds": [
        "string" // list userid
      ],
      "receiperId": "string", //
      "reactions": [
        {
          "reactorId": "string",
          "reactCode": "string"
        }
      ],
      "attachments": [
        {
          "name": "string",
          "desciption": "string",
```

```
        "id": "string",
        "url": "string",
        "urlPreView": "string",
        "typeAttachment": 1,
        "messageId": "string",
        "sizeByte": 0,
        "durationSeconds": 0,
        "revoke": 1
      }
    ],
    "type": 0,    /// - text = 0
                                  /// - audio = 1
                                  /// - file = 2
                                  /// - sticker = 3
                                  /// - gallery = 4
                                  /// - video = 5
    "revokeEnum": 1,
    "isDeleted": true,
    "isEdited": true,
    "lastEditTime": "2023-09-28T09:08:34.878Z",
    "countValue": 0,
    "isSave": true, // status save
    "isPined": true, // status pinner
    "isPinedById": "string",
    "lastPinedTime": "2023-09-28T09:08:34.878Z"
  }
],
"users": [
  {
    "id": "string",
    "avatar": "string",
    "roles": [
      "string"
```

```
      ],
      "username": "string",
      "nickName": "string",
      "displayName": "string",
      "defautlStatus": 0,
      "isOnline": true
    }
  ],
  "lastReadMessageId": "string",// message read lastest
  "userSendLastRead": "string",
  "lastReadMessageTimeSend": "2023-09-28T09:08:34.878Z"
}
```

Last Message User Group

Retrieves the last read message of a user in a group.

Request Params

Param	DataType	Required	Description
roomId	Yes	Yes	Id of group
userId	Yes	Yes	Id of user to check

```
// param is object
const result = await apiMessage.lastMessageUserGroup(param);
```

Responses

```
{
  "lastReadMessageId": "string", // id mess last read
  "userSendLastRead": "string",// userid last read mes
  "lastReadMessageTimeSend": "2023-09-28T09:38:52.883Z"
}
```

History Attachment

Retrieves the history of attachments in a group.

Request Params

Param	DataType	Required	Description
before	String	No	Messageid before
after	String	No	message id after
types	Array Number	Yes	VIDEO = 1,AUDIO = 2,IMAGE = 3,FILE = 4,OTHER = 5
skip	Number	Yes	Point start record
take	Number	Yes	The number of records retrieved

```
// param is object
{
```

```
  "before":"", //
  "after": '', //messageid after
  "types": [],  // -   VIDEO = 1,AUDIO = 2,IMAGE = 3,FILE = 4,OTHER = 5

  "skip": 0,  // point start record
  "take": 0// The number of records retrieved
}
```

Responses

```
{
  "skip": 0,
  "take": 0,
  "messages": [
    {
      "id": "string",
      "messageContentRaw": "string",
      "message": "string",
      "timeSend": "2023-09-28T09:39:43.708Z",
      "senderId": "string",
      "roomId": "string",
      "mentionsIds": [
        "string"
      ],
      "receiperId": "string",
      "reactions": [
        {
          "reactorId": "string",
          "reactCode": "string"
        }
      ],
```

```
        "attachments": [
          {
            "name": "string",
            "desciption": "string",
            "id": "string",
            "url": "string",
            "urlPreView": "string",
            "typeAttachment": 1,
            "messageId": "string",
            "sizeByte": 0,
            "durationSeconds": 0,
            "revoke": 1
          }
        ],
        "type": 0,
        "revokeEnum": 1,
        "isDeleted": true,
        "isEdited": true,
        "lastEditTime": "2023-09-28T09:39:43.708Z",
        "countValue": 0,
        "isSave": true,
        "isPined": true,
        }]
}
```

Search Message In Group

Retrieves the history of message in a group.

Request Params

Param	DataType	Required	Description
textSearch	String	Yes	Text search
skip	Number	Yes	Point start record
take	Number	Yes	The number of records retrieved

```
// param is object
const result = await apiMessage.searchMessegeInGroup(roomId, param);
```

Responses

```
{
  "messageId": "string", // id message
  "message": "string", // message
  "textSearch": "string",
  "messageIds": [
    "string"
  ], //list message id
  "messages": [
    {
      "id": "string",
      "messageContentRaw": "string",
      "message": "string",
      "timeSend": "2023-09-28T09:56:34.615Z",
      "senderId": "string",
      "roomId": "string",
      "mentionsIds": [
        "string"
```

```
  ],
  "receiperId": "string",
  "reactions": [
    {
      "reactorId": "string",
      "reactCode": "string"
    }
  ],
  "attachments": [
    {
      "name": "string",
      "desciption": "string",
      "id": "string",
      "url": "string",
      "urlPreView": "string",
      "typeAttachment": 1,
      "messageId": "string",
      "sizeByte": 0,
      "durationSeconds": 0,
      "revoke": 1
    }
  ],
  "type": 0,
  "revokeEnum": 1,
  "isDeleted": true,
  "isEdited": true,
  "lastEditTime": "2023-09-28T09:56:34.615Z",
  "countValue": 0,
  "isSave": true,
  "isPined": true,
  "isPinedById": "string",
  "lastPinedTime": "2023-09-28T09:56:34.615Z"
}
```

```
  ],
  "users": [
    {
      "id": "string",
      "avatar": "string",
      "roles": [
        "string"
      ],
      "username": "string",
      "nickName": "string",
      "displayName": "string",
      "defautlStatus": 0,
      "isOnline": true
    }
  ],
  "skip": 0,
  "take": 0,
  "totalCountFinded": 0
}
```


Get History Around Message

Request Params

Param	DataType	Required	Description
roomId	String	Yes	Room id search
messageId	String	Yes	Message id

take	Number	Yes	The number of records retrieved:

```
// param is object
const result = await api.getHistoryAroundMessage(param);
```

Responses

```
{
  "messageId": "string",
  "takeOffset": 0,
  "messages": [  //Similar to the description defined above
    {
      "id": "string",
      "messageContentRaw": "string",
      "message": "string",
      "timeSend": "2023-10-05T06:59:31.990Z",
      "senderId": "string",
      "roomId": "string",
      "mentionsIds": [
        "string"
      ],
      "receiperId": "string",
      "reactions": [
        {
          "reactorId": "string",
          "reactCode": "string"
        }
      ],
      "attachments": [
```

```
      {
        "name": "string",
        "desciption": "string",
        "id": "string",
        "url": "string",
        "urlPreView": "string",
        "typeAttachment": 1,
        "messageId": "string",
        "sizeByte": 0,
        "durationSeconds": 0,
        "revoke": 1
      }
    ],
    "type": 0,
    "revokeEnum": 1,
    "isDeleted": true,
    "isEdited": true,
    "lastEditTime": "2023-10-05T06:59:31.990Z",
    "countValue": 0,
    "isSave": true,
    "isPined": true,
    "isPinedById": "string",
    "lastPinedTime": "2023-10-05T06:59:31.990Z"
  }
],
"users": [
  {
    "id": "string",
    "avatar": "string",
    "roles": [
      "string"
    ],
    "username": "string",
```

```
      "nickName": "string",
      "displayName": "string",
      "defautlStatus": 0,
      "isOnline": true
    }
  ]
}
```

Get Last Message Pined

```
// param is object
const result = await api.getLastMessagePined(roomid);
```

Responses

```
{
  "messages": [ // Similar to the description defined above
    {
      "id": "string",
      "messageContentRaw": "string",
      "message": "string",
      "timeSend": "2023-10-05T06:58:45.318Z",
      "senderId": "string",
      "roomId": "string",
      "mentionsIds": [
        "string"
      ],
      "receiperId": "string",
      "reactions": [
```

```
      {
        "reactorId": "string",
        "reactCode": "string"
      }
    ],
    "attachments": [
      {
        "name": "string",
        "desciption": "string",
        "id": "string",
        "url": "string",
        "urlPreView": "string",
        "typeAttachment": 1,
        "messageId": "string",
        "sizeByte": 0,
        "durationSeconds": 0,
        "revoke": 1
      }
    ],
    "type": 0,
    "revokeEnum": 1,
    "isDeleted": true,
    "isEdited": true,
    "lastEditTime": "2023-10-05T06:58:45.318Z",
    "countValue": 0,
    "isSave": true,
    "isPined": true,
    "isPinedById": "string",
    "lastPinedTime": "2023-10-05T06:58:45.318Z"
  }
],
"users": [
  {
```

```
      "id": "string",
      "avatar": "string",
      "roles": [
        "string"
      ],
      "username": "string",
      "nickName": "string",
      "displayName": "string",
      "defautlStatus": 0,
      "isOnline": true
    }
  ],
  "total": 0
}
```

Last modified 1mo ago

CDN

Upload Emoji

Param	DataType	Required	Description
Emoji	File array	Yes	
EmojiNames	String array	Yes	

```
// example
const result = await cdn.uploadEmoji(params);
// response
// success: true or 200 in status
```

List Emoji

Param	DataType	Required	Description
filter	String	No	

skip	Number	Yes	*Default value* : 0
take	Number	Yes	*Default value* : 20

```
// example
const result = await cdn.listEmoji(params);
```

Responses

```
{
  "items": [
    {
      "name": "string", // name emoji
      "url": "string" // url
    }
  ],
  "totalCount": 0
}
```

Upload Stickers

Param	DataType	Required	Description

```
// example
const result = await cdn.uploadStickers(params);
// response
// success: true or 200 in status
```

Group	String	Yes	Group

Create Group Stickers

Param	DataType	Required	Description
groupName	String	Yes	

```
// example
const result = await cdn.createGroup(params);
// response
// success: true or 200 in status
```

Responses

```
{data:"string"}
```

Get Group Stickers

Param	DataType	Required	Description

```
// example
const result = await cdn.getGroup(params);
// response
// success: true or 200 in status
```

Responses

```
{
  "groups": [
    "string"
  ]
}
```

List Stickers

Param	DataType	Required	Description
filter	String	No	
group	String	Yes	
skip	Number	Yes	*Default value* : 0
take	Number	Yes	*Default value* : 20

```
// example
```

```
const result = await cdn.listSticker(params);
```

Responses

```
{
  "items": [
    {
      "name": "string",
      "group": "string",
      "url": "string",
      "contentType": "string",
      "size": 0
    }
  ],
  "totalCount": 0
}
```

Last modified 1mo ago