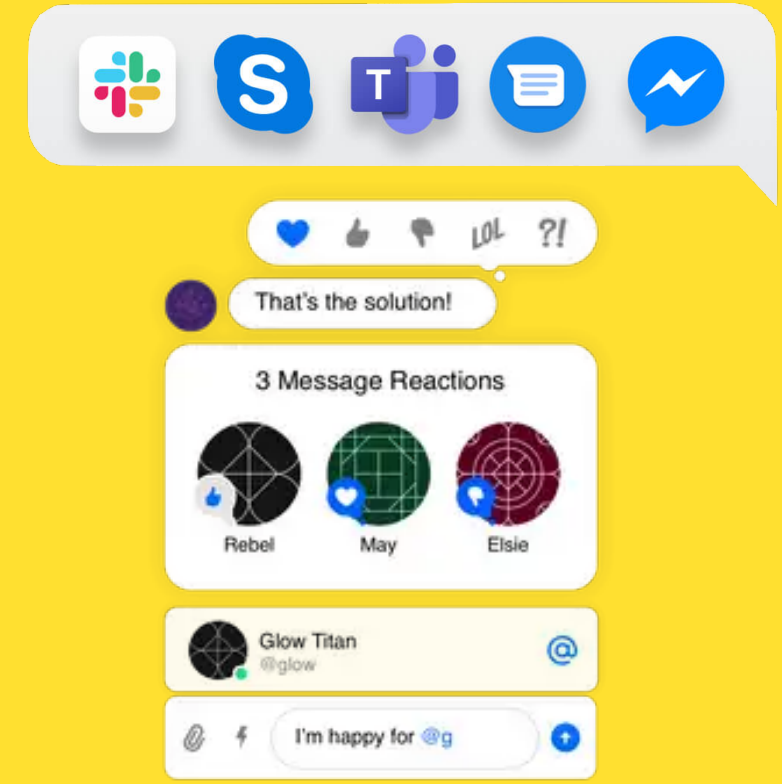
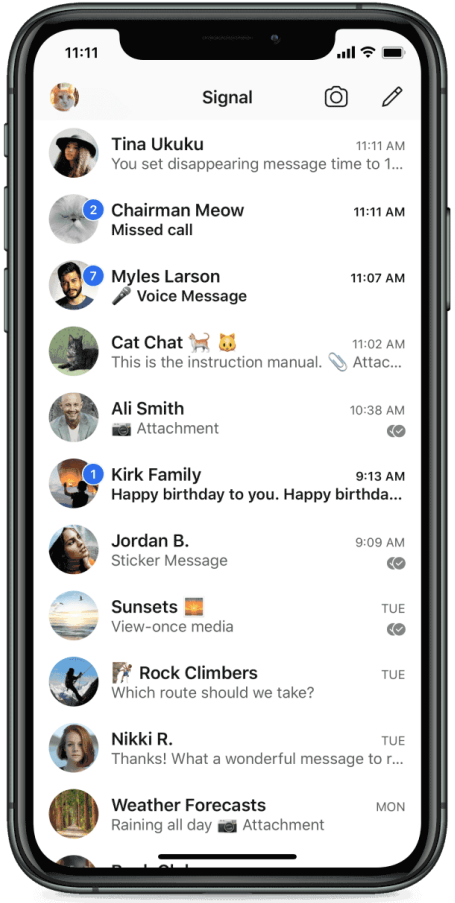


Chat Application

React Native tutorial for beginners

We are going to build



We are going to build

- ✨1-on-1 Real-time messaging
- ✨Group messaging
- ✨Emojis, Reactions, GIFs
- ✨Image, Video and Attachment messages
- ✨URL previews
- ✨and more

Prerequisites

1. Asset Bundle (dummy data, images, icons, PDF presentation, unlimited karma):
https://assets.notjust.dev/chat

2. Expo environment setup
http://bit.ly/expo-vadim

3. Stream account (30 days free trial)
https://gstrm.io/notjustdev

Let’s get started

1. Initialise the expo project:

```
$ expo init Messenger -t tabs
```

2. Open project

```
$ cd Messenger
```

```
$ code .
```

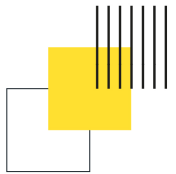
3. Run the development server

```
$ npm start
```

4. Run the app on device or emulator

```
$ npm run android
```

```
$ npm run iOS
```



Install Stream Chat SDK

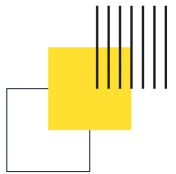
1. Stream Chat SDK Docs:

   https://getstream.io/chat/docs/sdk/reactnative/

2. Open project

```
$ expo install stream-chat-expo

$ expo install @react-native-community/netinfo
expo-blur expo-document-picker expo-file-system
expo-haptics expo-image-manipulator expo-image-
picker expo-media-library expo-sharing react-
native-gesture-handler react-native-reanimated
react-native-safe-area-context react-native-svg
```



Create a client

```
const client = StreamChat.getInstance(API_KEY);
```

Connect the user

```
await client.connectUser(
  {
    id: 'vadim',
    name: 'Jim Lahey',
    image: 'https://i.imgur.com/fR9Jz14.png',
  },
  client.devToken('vadim'),
);
```

Disable Auth checks

On the **Dashboard**:

- Open *Select App.*
- Select the App you want to enable developer tokens on.
- Open Chat dropdown and select overview
- Scroll to *Chat Events > Authentication*
- Toggle *Disable Auth Checks*
- Save these settings.

https://getstream.io/chat/docs/javascript/tokens_and_authentication/?language=javascript#developer-tokens

Create a channel

```
const channel = client.channel(
  'messaging',
  'notjustdev',
  { name: ‘notJust Development', }
);

await channel.create();
```

Overlay Provider

- Highest level Stream Chat component
- used for message reaction overlays, fullscreen image viewer, attachment picker

Full docs: https://getstream.io/chat/docs/sdk/reactnative/core-components/overlay-provider/

Chat

- provides the context for the chat
- can wrap the whole app, or only a specific screen
- required prop: client

Full docs: https://getstream.io/chat/docs/sdk/reactnative/core-components/chat/

Chat List

- Displays a list of channels

Full docs: https://getstream.io/chat/docs/sdk/reactnative/core-components/channel-list/

Channel

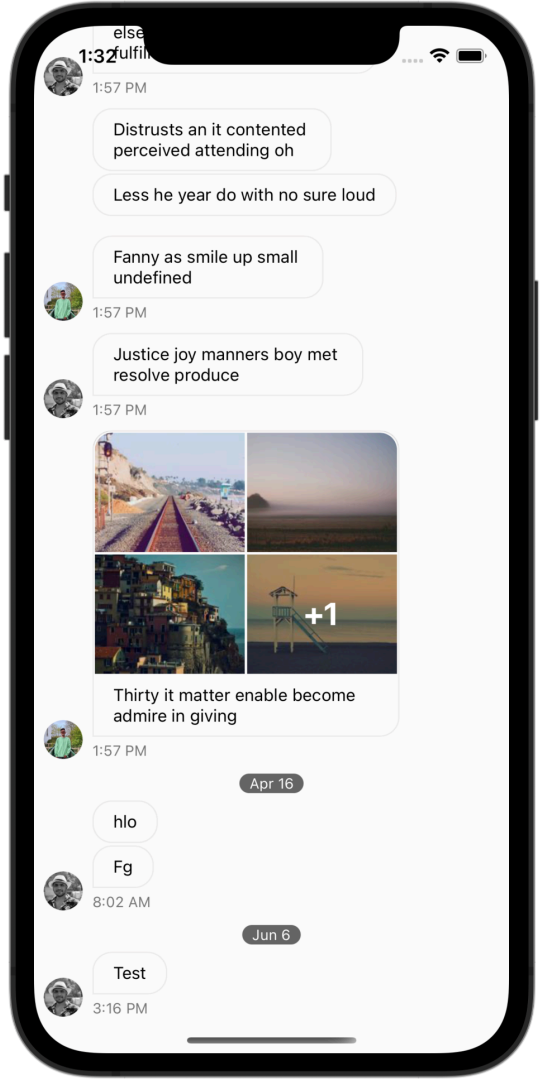
- Provides the context for a specific channel
- required prop: channel

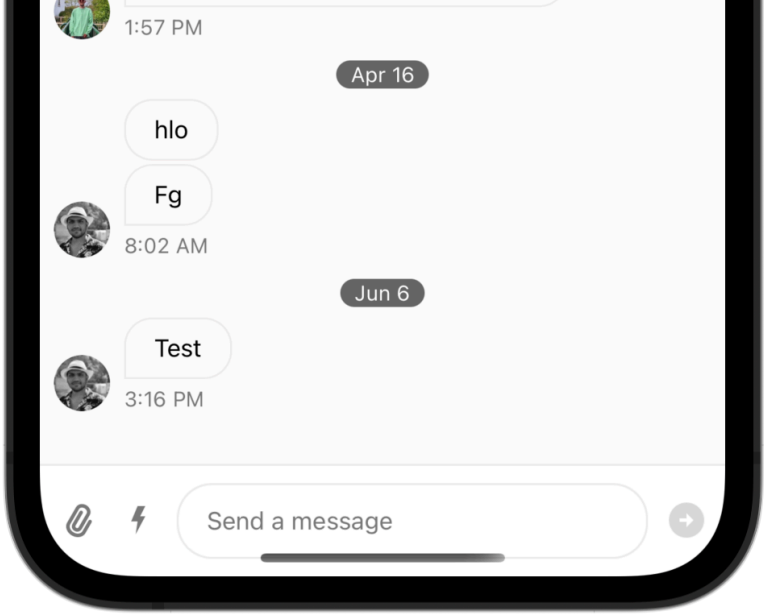
Full docs: https://getstream.io/chat/docs/sdk/reactnative/core-components/channel/

Message List

- Displays a list of messages of one channel

Full docs: https://getstream.io/chat/docs/sdk/reactnative/ui-components/message-list/





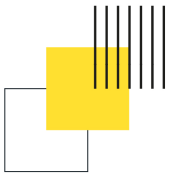
Message Input

- The input for new messages

Full docs: https://getstream.io/chat/docs/sdk/reactnative/ui-components/message-input/

Summary

- You need a client that is an instance of StreamChat to provide to the UI components so they function correctly.
- You need to connect a user via the connectUser call to interact with the backend properly. To do this you need to create tokens as per the Tokens & Authentication docs.
- You need to surround your app with the OverlayProvider for the UI to function properly, and your app or chat screen with the Chat component to provide data to the UI components.
- ChannelList provides access to the list of available channels in the UI, and using props, can be queried, filtered, and sorted as desired. ChannelList also provides convenient functionality for implementing navigation to a selected Channel.
- The Channel component is the functional wrapper of the MessageList, MessageInput, and Thread components. It is the access point for the majority of the customization of the SDK, and controls much of the functionality.



Basic Authentication

- Add an Authentication screen that will simulate the authentication of the user
- connectUser to the Stream Chat SDK when logged in
- Navigate home, only if user is logged in

List of users

- Get the list of users

```
await client.queryUsers({});
```

- Render the users in a FlatList

Create a chatroom with a user

- When clicking on a user, create a chatroom with him

Navigation

Navigate to the chatroom when clicking on it in the list