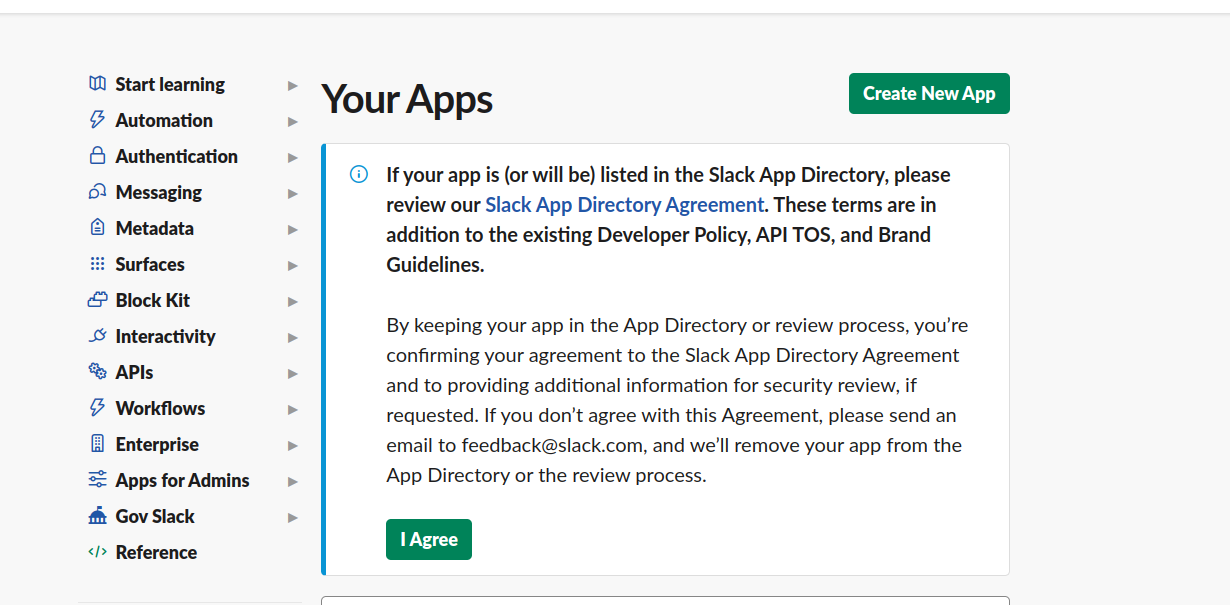
# **Slack Integration to Java Application**

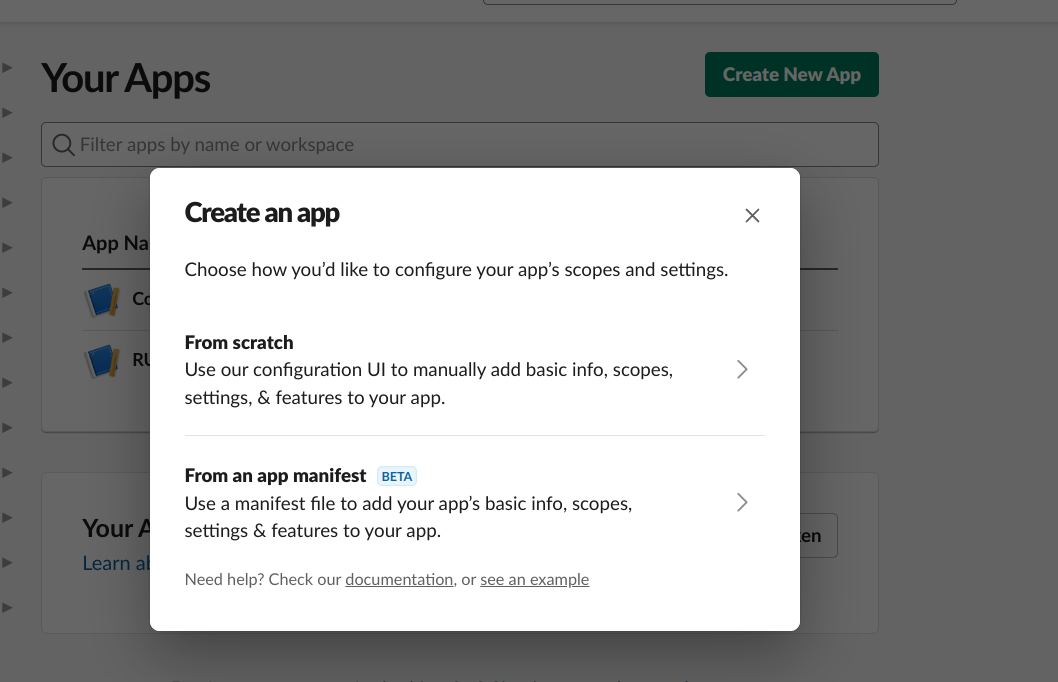
Firstly we have to configure the slack app in our slack workspace.

For that following steps will be needed:

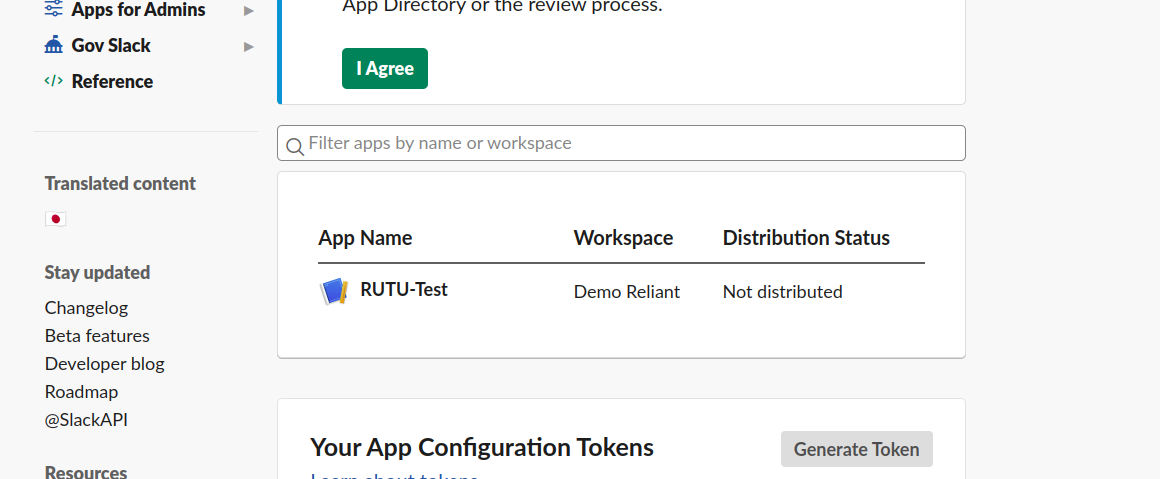
1. Create app in your slack workspace by following the below steps:
   1. Open this url: <https://api.slack.com/apps>
   2. You will get below screen:



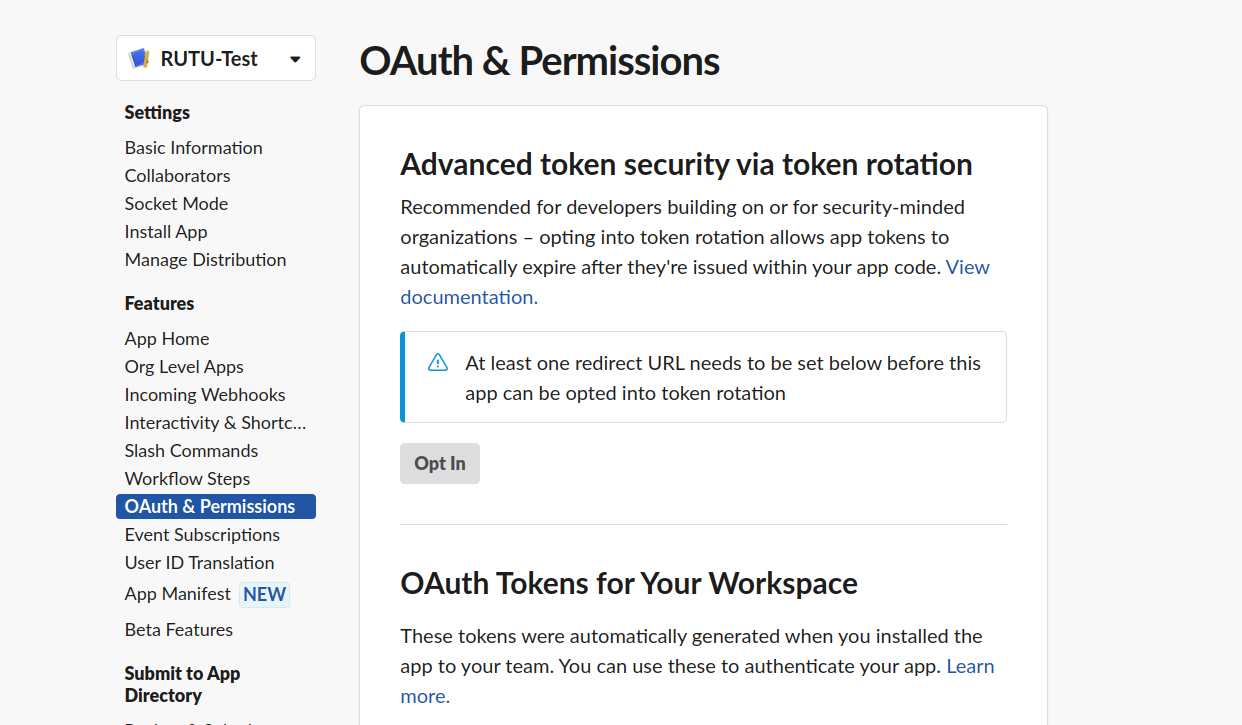
* 1. Then create new app(From Scratch):



* 1. After creating an app, Open that app from list of apps given here:



* 1. Go to the “OAuth & Permissions” tab.



* 1. Then add following scopes on this page:
     1. Bot Token Scope : channels:read, chat:write, groups:read, groups:write, im:read, im:write.
  2. After that reinstall the app from this option from same page:



Now, As slack app configuration is all done, we can move to the code part where we are gonna write the code to send messages into slack through java application.

For that we will follow below steps:

1. Create a simple spring boot project and add needed spring boot dependencies, lombok dependency and slack dependencies given in the demo project.
2. Then add **Bot User OAuth Token**(starts from xoxb-) to the environment variable.
3. Find workspace id from slack web login to workspace as given in below image and add it to the environment variable.



1. Now, configure slack in java application by **Bot User OAuth Token** in configuration file.

@Bean

public SlackClient slackClient(){

SlackClientRuntimeConfig runtimeConfig = SlackClientRuntimeConfig.*builder*()

.setTokenSupplier(() -> env.getRequiredProperty("SLACK\_BOT\_TOKEN\_MAIN"))

.build();

return SlackClientFactory.*defaultFactory*().build(runtimeConfig);

}

1. First to send a message in any channel, the channel name is needed to add an environment variable.
2. After that we will build message params as ChatPostMessageParams using channel name and message and pass that in the **postMessage** api of slack.

public ChatPostMessageParams buildBugReportMessage(final String channel, final String message){

return ChatPostMessageParams.*builder*()

.setChannelId(channel)

.setText(message)

.build();

}

public ChatPostMessageResponse sendMessage(ChatPostMessageParams params){

try{

// Send Message

return slackClient.postMessage(params).join().unwrapOrElseThrow();

}catch (IllegalStateException ex){

*log*.warn("Unable to Send Message", ex);

return null;

}

}

1. Messages will be received in the given channel.
2. Now to send a direct message to the app of a particular user, we will need the user id of that user. You can get the user id from the uri of the user profile as given in the image below.



1. Add user id in environment variables and then need to open slack conversion using user id.

public ConversationsOpenResponse openConversation(String userId){

return slackClient.openConversation(ConversationOpenParams.*builder*()

.setUsers(Collections.*singletonList*(userId)).build()

).join().unwrapOrElseThrow();

}

1. Then build message params as ChatPostMessageParams using the conversation id, which we can get from **conOpenResponse.getConversation().getId()** from the above method’s response.
2. In this build method, we can add different sections of message body, like title, message, notes, regards etc.

public ChatPostMessageParams buildMessageForReminder(String userId, String name, String message, String note) {

List<Section> sectionList = new ArrayList<>();

Section titleSection = Section.*builder*()

.setText(Text.*builder*()

.setType(TextType.*MARKDOWN*)

.setText("\*Hey " + name + ",\*")

.build())

.build();

sectionList.add(titleSection);

Section messageSection = Section.*builder*()

.setText(Text.*builder*()

.setType(TextType.*MARKDOWN*)

.setText(message)

.build())

.build();

sectionList.add(messageSection);

Section noteSection = Section.*builder*()

.setText(Text.*builder*()

.setType(TextType.*MARKDOWN*)

.setText("\*Note:\* " + note)

.build())

.build();

if (StringUtils.*hasText*(note))

sectionList.add(noteSection);

Section regardsSection = Section.*builder*()

.setText(Text.*builder*()

.setType(TextType.*MARKDOWN*)

.setText("\*Regards,\* \nDemo Reliant")

.build())

.build();

sectionList.add(regardsSection);

return ChatPostMessageParams.*builder*()

.setText("Test Direct Message")

.setChannelId(userId)

.setBlocks(sectionList).build();

}

1. Now call the **postMessage** api of slack using these params to send the message.

public ChatPostMessageResponse sendMessage(ChatPostMessageParams params){

try{

// Send Message

return slackClient.postMessage(params).join().unwrapOrElseThrow();

}catch (IllegalStateException ex){

*log*.warn("Unable to Send Message", ex);

return null;

}

}

1. And the user will get the message in app bot as given below.

