|  |
| --- |
| Events/Notifications  Webhooks + Java + Slack |

Q

Webhooks

Slack is capable of employing Webhooks to enable a Slack Workspace to communicate with applications running at external computer installation. The same Webhook technology enables Slack to communicate with other applications within the same organization/computer installation registered to the (cloud-based) Slack Workspace.

While Slack is OAuth2-capable, this security feature is out of scope for this discussion.



**Part One** covers creating a proprietary Slack token, capable of acting as a switching mechanism for a passive receiver/destination for incoming messages.

ref <https://api.slack.com/tutorials/slack-apps-hello-world>



**Part Two** covers the Java code that acts as a publisher of messages to what Slack calls an Application - client code which forwards to a Slack (communal) Workspace.

In between the Java call site and the (Slack-fabricated) Slack Application, the proprietary Slack token is used to route messages to another endpoint, using either an IRC gateway or an XMPP gateway.

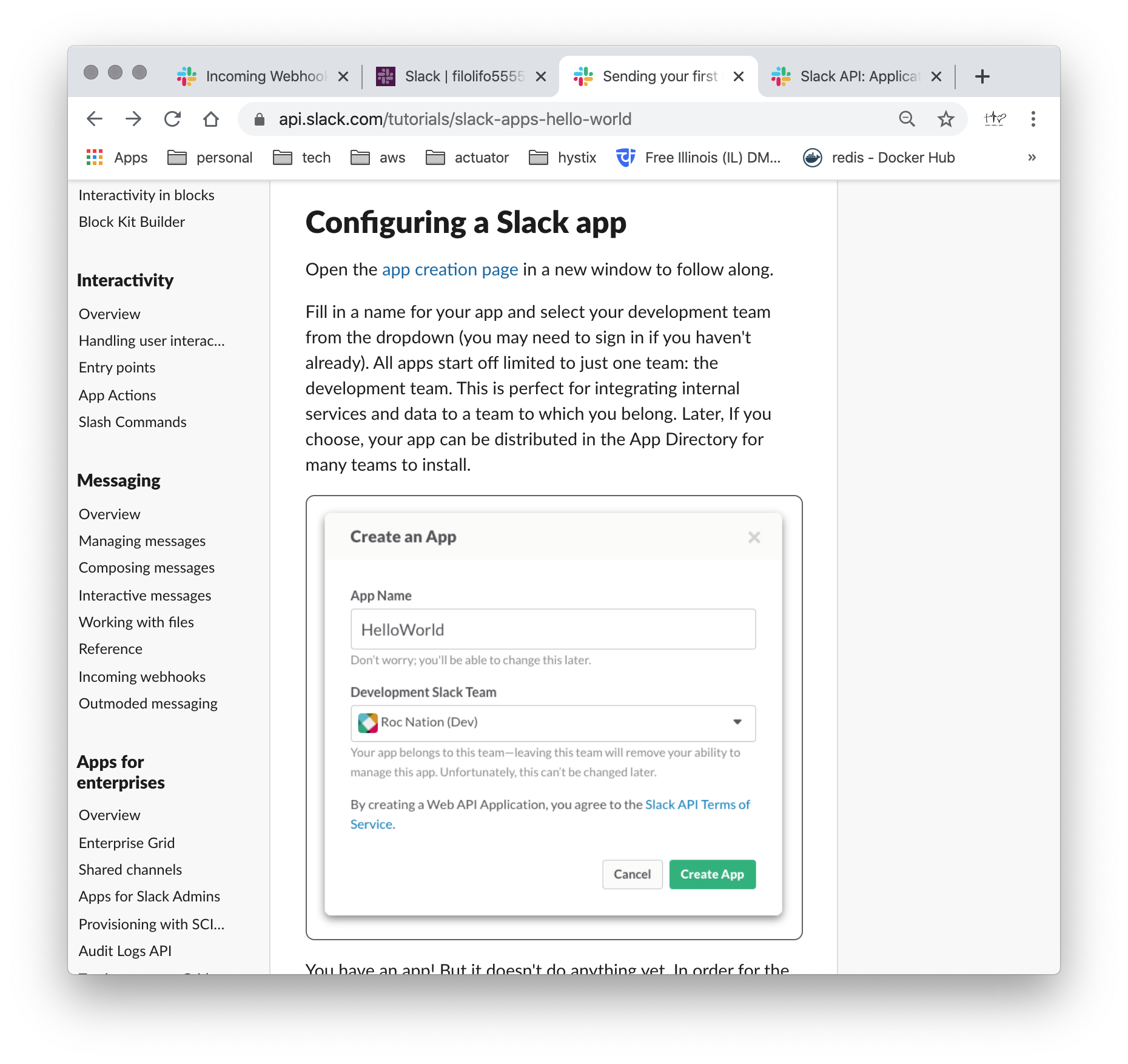
The gateway comprises the Network layer (OSI Layer 3). Firewall-tunneling may be applicable in some circumstances.

The exact nature of Slack’s underlying transmission Data Link (OSI Layer 2) protocol is not known.

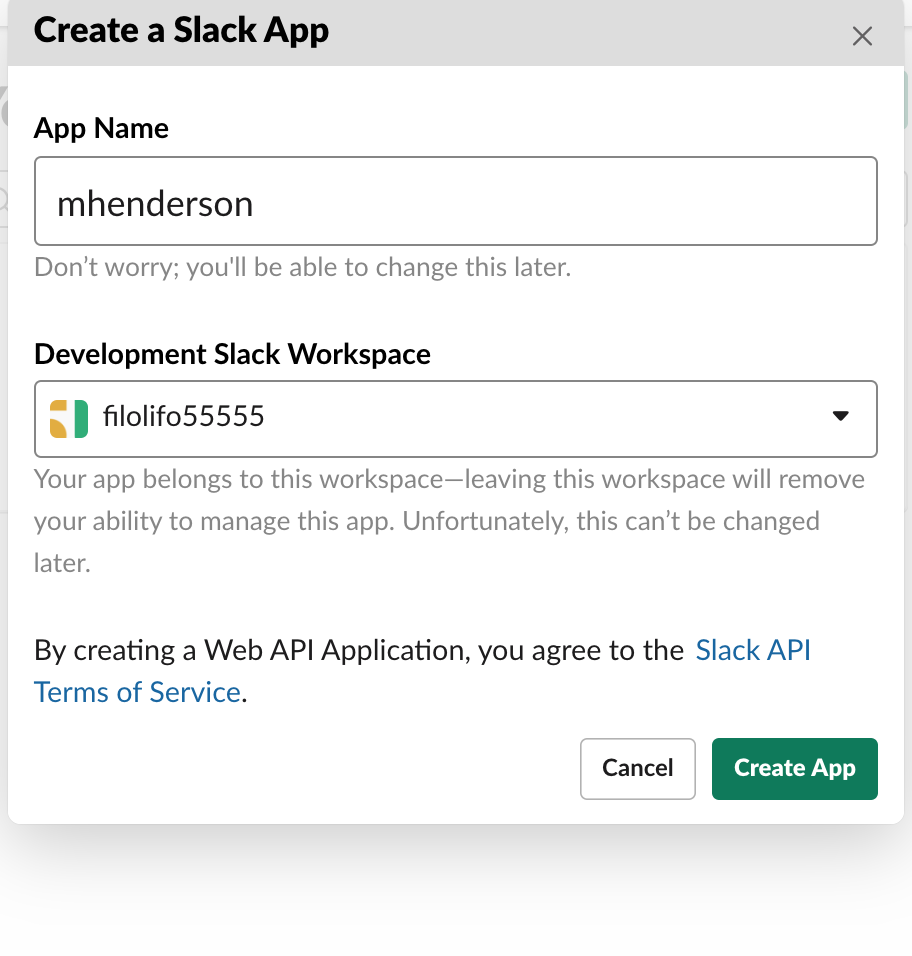
Part Onee

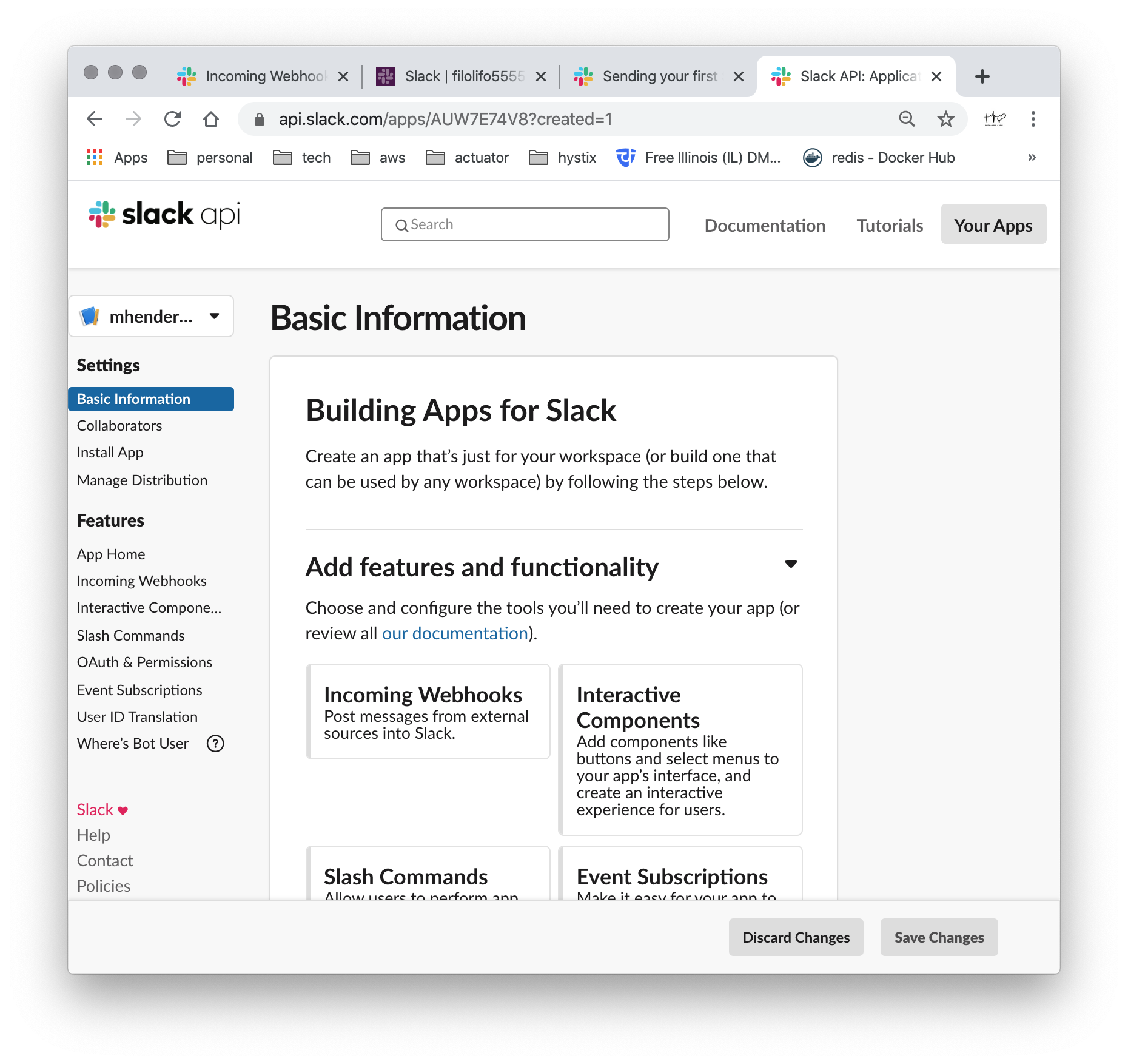
This section presents the screen shots from a succession of steps that create a Slack Webhook Token, and a Slack Application, which is associated with an extant Slack Workspace (filolifo55555).

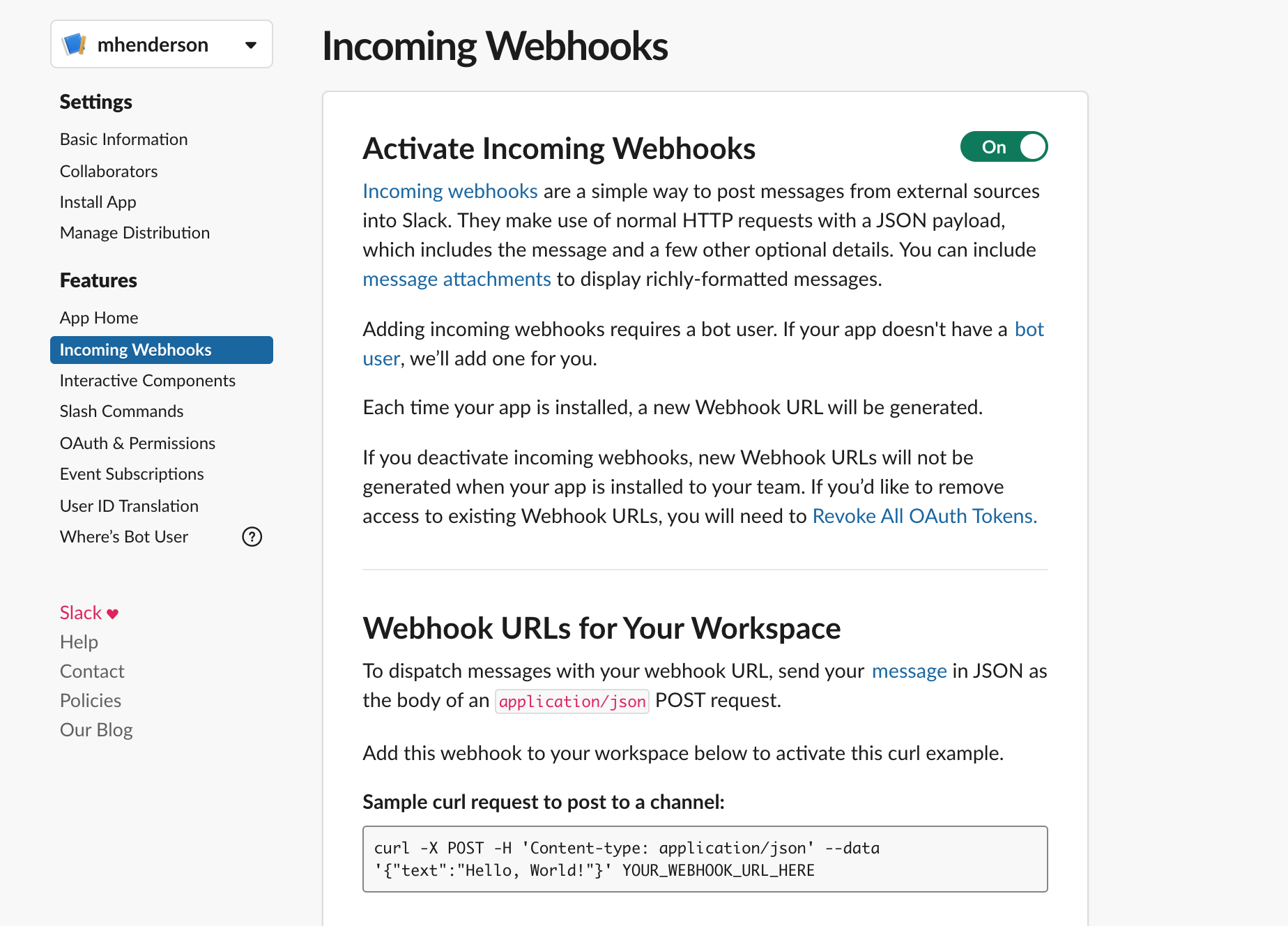
The screen shots were culled from me following along with the instructions contained in the page shown below - from the link <https://api.slack.com/tutorials/slack-apps-hello-world>:



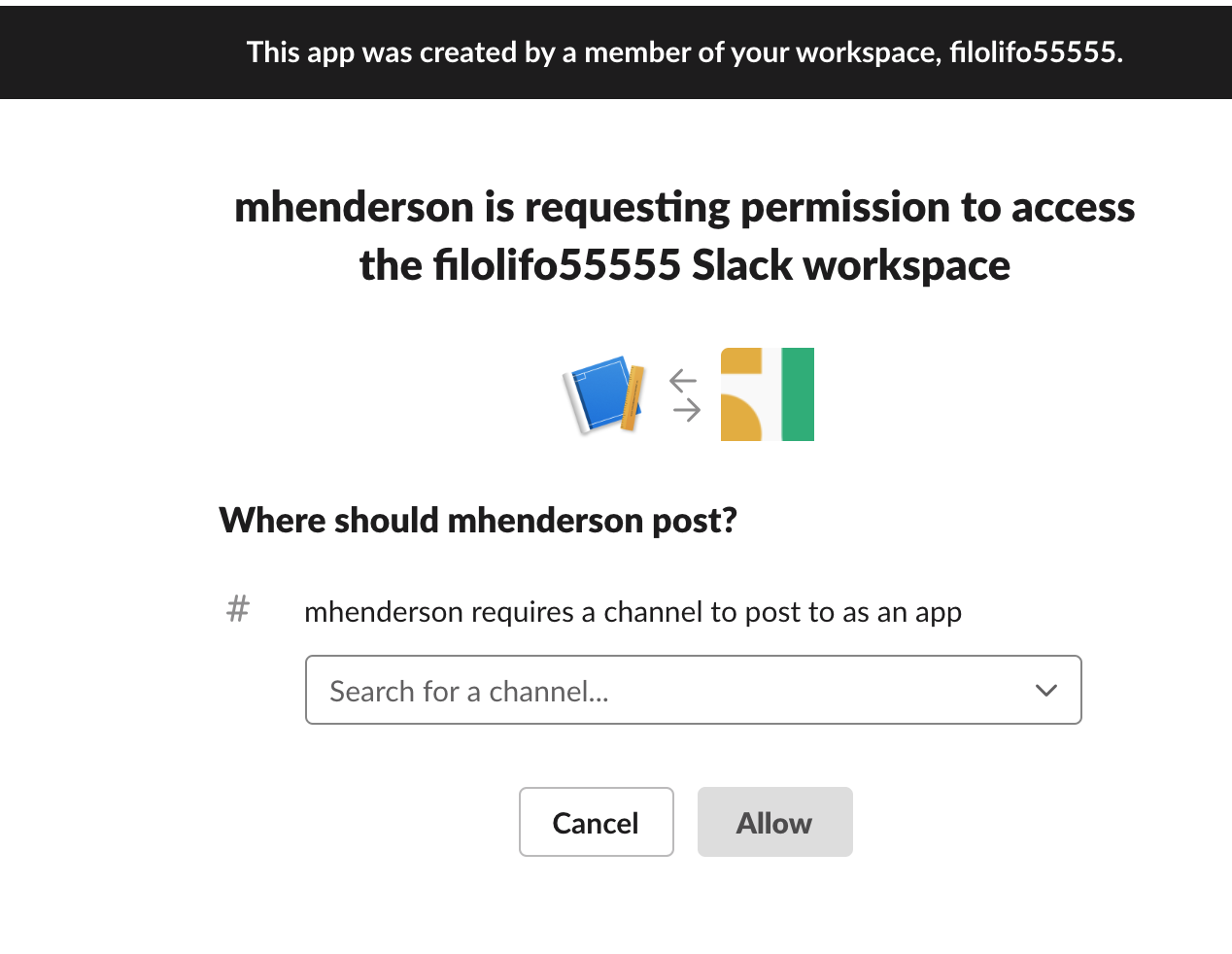
Notice the above page has text: app creation page, which is a hyperlink named link <https://api.slack.com/apps/new>. This link launches the app creation page shown below.

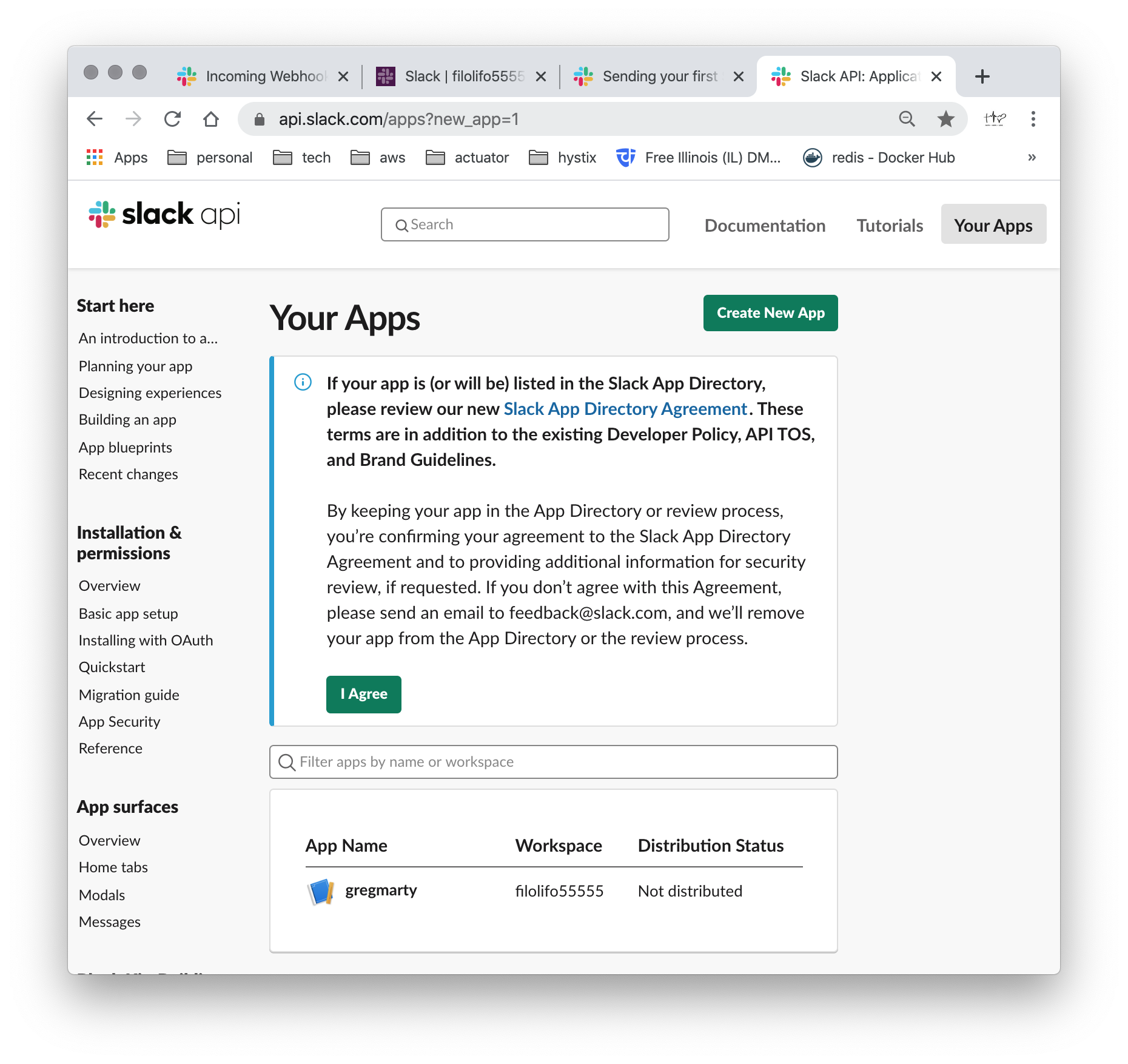






Notice the slider in the upper right-hand corner is green/on.





The app creation page, shown above already contains a Slack app named gregmarty. This app was previously created to interact with the Slack Workspace, which is named filolifo55555.

In that Slack Workspace, shown later, you’ll see messages dating back to July 1, 2018 (when the Slack Workspace and the Slack App were each created together).

Notice the green Create New App button (upper-right corner).

In the activities covered by this document, we will repurpose the filolifo55555 Slack Workspace, but instead create a new Slack App, named mhenderson. It will interact with that same Slack Workspace (filolifo55555), by means of the Slack Webhook. Click the green Create New App button (upper-right corner).

The balance of the steps outlined in this section (Part One) of this document show how an extant Slack Workspace is associated with a newly-minted Slack App, and how it can be tested using cURL that is “wired” to a Java Publisher (client code).

The second part of this document outlines the Java participant in the Slack App’s Webhook contract that fulfills the role of Publisher.

Part Twoe

Q

Slack Webhooks, along with a Slack App, allow a Stack Workspace to act as an endpoint for calling code. In our case, we want a Java client to emit messages to the Slack Workspace (via the Slack-generated Slack App/Slack Webhook.

A quick test to prove the Slack Workspace is being sent messages can be done using cURL. This serves as a baseline test for the Java client we are writing.