

App UI Guidelines

Building great user experiences
on Slack





Contents

Building Apps for Slack	4
Understand Your Audience	5
Timezones	5
Team size	5
Enterprise grid	6
User roles and access	6
Desktop vs. mobile	7
User and team preferences	7
Familiarity with apps	11
Voice and Tone: Communicate with Clarity	12
Clear, concise and human	12
How much personality is too much?	13
Let's talk about the actual words: be brief	14
Be empathetic.....	17
Spend time rewriting	18
Outline Your Use Cases	20
You've got options	20
Message structure.....	21
Building blocks of a message	22
Adding interactivity to messages	25
Breaking a message down	27
Collecting information from users.....	28

Contents

Storyboard Your Interactions	30
Interactions have a life-cycle	30
Pick entry points for interactions.....	31
Keep messages glance-able	32
Keep text segments bite-sized and conversational	33
Choose sensible default options	35
Cleaning up after your app	35
 Being a Good Citizen Inside of Slack	 38
Considerate notifications.....	38
Sending direct messages to users	41
Responding in channel	42
 Create a Good Onboarding Experience	 44
Say hello.....	45
Offer help	46
Make your app visible	50
 Appendix	 52
App templates.....	52
Onboarding experiences.....	56
App stencil	63

Building Apps for Slack

We all want to make our users' working lives more pleasant and productive. And we understand that you want as many people in a workspace to use your app—we do too! This guide is meant to steer you in the right direction when designing a Slack app.

The best experience will look a little different for every app, but it will always be one built with the end user in mind.

Understand Your Audience

Slack users are people of all ages, races, genders, and ability levels. They may have poor internet connections, use Slack only on mobile, or they might never have used a Slack app before. We want them all to have a great experience on Slack, so please be sympathetic of your audience when designing your app. Here are a few factors to keep in mind:

Timezones

Slack is used in over 150 countries around the world. Even in a single Slack workspace, there may be users spread out across multiple timezones. If your app posts messages at a particular time of day, understand that people may see it at a different time than everyone else on their team.



Design Tip: If your app sends notifications, allow users to configure when and how frequently the message is posted, preferably on a per-channel basis.

Team size

Not all Slack workspaces are the same size. There are five-person non-profits using Slack, and there are 50,000-person enterprise companies using Slack. Some teams keep their conversations in a dozen channels, while others create five channels for every new project or sub-team across their organization.



Thought Starter: If your app uses lists of users and channels, consider how the experience may be on larger teams. You may need to find ways to abbreviate, truncate, or paginate lists you display in your UI.

Enterprise grid

Enterprise Grid is a tier of Slack that allows large organizations to communicate across several workspaces. It also introduces new features, like Shared Channels, which enable cross-organization communication. In order to support Enterprise Grid, you'll want to handle a few new considerations:

- **Handle unknown users gracefully:** In shared channels, your app will encounter users it isn't aware of from other organizations. Their user IDs will not be included in calls to methods like `users.list`.
- **Duplicate events:** If you're using the RTM API for development (we recommend using the Events API instead), it's possible you will receive duplicate message events when your app is in a shared channel.
- **Changes to user objects:** A few things change about user objects in an Enterprise Grid world. The most important is that a user ID will likely start with a "W" instead of the "U" your app is used to. There are also new fields on the user object, like `enterprise_id`, `enterprise_name`, `is_admin`, and `is_owner`. These may or may not matter to your app during development, but they're there if you need them.



Development Tip: The Events API de-duplicates events for you by default and supports newer platform features like dialogs and actions. The RTM API should only be used in unique situations when Events API cannot be used (like IT firewall limitations).







You can read more on how to design for Enterprise Grid at api.slack.com/enterprise-grid

User roles and access


There are [many different user types in Slack](#). Aside from standard users, you should be aware of guest accounts, which are restricted to fewer channels and permissions. You may also come across actions performed by other bot users. Depending on your app's purpose, it may be appropriate to treat these users the same (or different) than regular users.

You can read on user permissions on Slack at [get.slack.help](#).

 5 Members 

-  Team Owner
-  Multi-channel Guest
-  Standard Member
-  Team Admin
- ▲  Single-channel Guest
-  External Guest



 **Development Tip:** Your app can call the `users.info` method with any user ID to determine their user type. If your app posts sensitive data that guest accounts shouldn't have access to, you may want to call `users.info` with the ID of any user that interacts with your app (before posting any information).

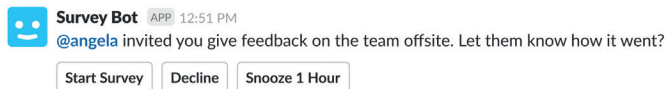
In addition to guest users, some standard users may have permissions that others do not. For example, team administrators may only permit certain people to install and manage 3rd party apps, so prompting a guest user to authenticate their account may lead to a dead end. If you can, test your app using a variety of user types.

Desktop vs. mobile

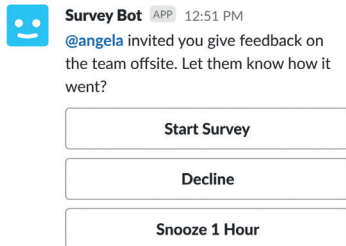
While some people only use Slack on their computer, an increasing number are using mobile alongside desktop.


Every app feature Slack releases is tested across the different Slack clients, though there will be changes to the way an interface appears or how a user interacts with your app. Therefore, it's important to make sure that you test out your app's interactivity and messages on as many screens as you can.

Desktop



Mobile



 **Thought Starter:** One reason people use Slack on their phones is to consume information quickly. If your app posts messages like notifications or status updates into channel, think of the mobile experience and make those messages immediately glanceable.

User and team preferences

No two teams or individuals use Slack in the same way. Our Help Center (get.slack.help) lists all team and user-level settings, but here are some specific ones worth considering when developing an app:

Team-level setting	Why this matters
Message and file retention	Messages and files may disappear after a set amount of time, starting at a day. If your app is dependent on messages or files staying around, you'll need to design around this setting.
Workspace language	Workspaces can set their default language and Slack will be localized to that language. If you're developing an app that will be used in several different languages, you can use this to localize your app interactions.



Development Tip: You can add an optional `include_locale` for calls that retrieve information about users and channels (like `users.info`, `conversations.info`, `channels.info`, `im.info`).

User-level setting	Why this matters
Automatically collapse all image and file previews	If you're sending images and files to a channel, keep in mind that some users may not automatically see them. These users will still have the optionality to expand images and files if they choose.
Disable animated images and emoji	If your app is sending GIFs, be aware that some users may not see them if they've disabled animated images.
Availability and Do Not Disturb	Your app should respect user availability and Do Not Disturb hours, especially if you're sending a message you want the user to interact with (like a DM).
Change your timezone	A user may move across the world or may just be traveling for work. If you're using a user's timezone for anything, make sure to fetch the most up-to-date one instead of relying on older information.



Development Tip: To get a user's timezone, your app can call the `users.info` endpoint and look under at the `tz` and `tz_offset` fields. The `users.getPresence` and `dnd.info` endpoints allow your app to view a Slack user's availability and Do Not Disturb hours, respectively.

You can see all of the available team and user settings in our Help Center at get.slack.help

Familiarity with apps

As a developer, it's your responsibility to make a great first impression to those interacting with your app, no matter their familiarity level.

Imagine that someone who doesn't know how to use your app installs it to their Slack workspace. Or think about the perspective of others in the same workspace as an installing user, who probably don't know that your app was installed. Try to picture the experience of a user who has never encountered any Slack app before.

Read the Onboarding experiences section for guidance on developing this interaction flow.

Try it out

- Where are your users primary located geographically?

- What segment are your users more likely a part of? Consumer, Small-to-Medium businesses (SMB), or Enterprise businesses?

- Look back at the different kinds of users in the User roles and access section and think about the different kinds of users that will use your app in Slack. Should your app behave differently for a standard member and a team owner? How about a standard member and single-channel guest?

Voice and Tone: Communicate with Clarity

Your Slack app is a representation of your brand, and the way your app communicates will become part of your brand voice, especially if you're building a conversational interface. Now, you might just be two people in a room who have put together an amazingly useful app and want to get it into the world (and into Slack), and you're thinking, "But we don't have a brand voice."

Well, you do now. Even if the only place it's used right now is this one app. So you need to be thoughtful about defining what you brand voice is. Unless you're bringing in a seasoned professional writer to give the thing a real personality, the best thing to do is think of this voice as an extension of your own voice.

Clear, concise and human

If you stick to being clear, concise, and human, you can't go far wrong. Whatever you're trying to say, think about how you would explain it first. Don't leap into "what words would my app choose." You'll probably end up picking bizarre ones.

You might find it helpful to jot down some attributes or characteristics of your brand voice (e.g. friendly, authoritative), once you've thought about it. MailChimp's Voice and Tone guide (styleguide.mailchimp.com/voice-and-tone) is a widely-admired example.

How much personality is too much?

Your app's primary purpose is not to sound clever or to entertain your users, it's to help them accomplish a task— even if that task is inherently entertaining, like finding just the right cat GIF.

No matter how useful the service that your app is providing, it'll be for nothing if you annoy people so much they'd rather go through whatever laborious process they were using before than have to deal with your overly chipper app one more time.

You want to be able to differentiate yourself from the crowd, for sure, but try adhering to these guidelines:

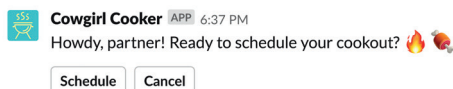
- Don't construct a personality that means you have to add sentence upon sentence in order to get to a joke "in keeping" with your app's sense of humor. Just get to the point.
- Try to avoid torturous puns or wordplay if they distract from the meaning.
- Informality is good, but getting over-friendly is charming to a very small number of people. The rest will find it grating, or, culturally insensitive (particularly in a workplace).
- If you decide to give gender to your app (and it's very easy not to), then be appropriate with the kind of things they should and shouldn't say. Don't be tempted to use it as an excuse to get lazy about behaviors or phrases stereotypical to any gender (or to particular age groups, or anything else). You'll end up driving people away.
- Using contractions and conversational cadence is a good way to lightly infuse your app with human personality - "You'll be able to" rather than "You will" and that sort of thing.

A little goes a long way. We cannot say this enough.

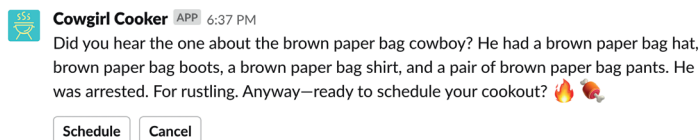
Let's talk about the actual words: Be brief

Don't add a joke or aside just to add one—almost every word your app says should facilitate an interaction (courteous parts of speech, such as greetings, are also useful).

Like this



Not like this



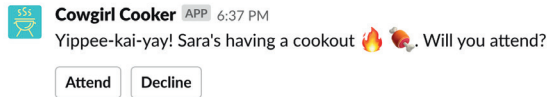
The first example still has plenty of distinctive personality, but gets straight to the point, and doesn't risk users tuning out/not wanting to wade through unimportant content.

Be clear

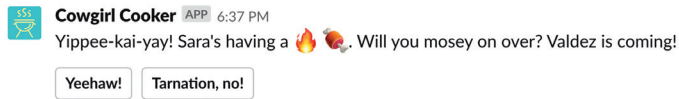
Words and copy used in your interactions should be easily understood even by someone who doesn't speak the same language fluently. That means:

- Don't use jargon and slang in the important parts of message text
- Avoid culturally specific references, like jokes from movies
- Stick to common, simple words
- Buttons labels should be clear and specific
- Make buttons active-voice and reflect the user's outcome (Save, Book Flight, Place Order)
- Avoid vague, non-actionable text like Click here or Settings
- Don't replace words with emoji

Like this



Not like this



The second example includes a reference to an obscure film that's likely to confuse many more users than it delights. The emoji combination is also potentially confusing, and may stall some users as they try to decipher it ("Fire... meat? Firemeat?").

The message button copy is similarly difficult to understand, and, in the worst case, could prevent users from selecting a response at all. Try to use standard combinations on buttons (Attend/Decline, Confirm/Cancel) to help your users have a smooth, simple experience with your app.

Read over your copy and ask yourself, "Is there anywhere a user may pause in confusion?" If so, rewrite.

Be empathetic

There is a diversity of people using Slack, and we previously described how important it is to understand that variation in terms of their ability to use your app properly. But that diversity is also important when thinking about the tone you use to communicate with them.

Ensure that your voice and tone has empathy towards every single person who uses your app. Some basic steps to take include:

- Use gender-neutral pronouns
- Use a variety of emoji skin tones
- Don't use sexist, racist, or ableist language
- Make an effort to trial your voice and tone with people from a diversity of backgrounds
- Don't assume any level of technical fluency from your users, keep instructions clear and simple

Writing for a broad audience takes a bit of practice if you're new to it, and it is usually easier if you have a diverse team of people working on your app from the start.

Spend time rewriting

We spend a good amount of time at Slack writing, rewriting, agonizing over, and then rewriting just once more to get every sentence as good as we can make it. If you've worked with a professional writer before, you know that no one, not even the people who make a living at this, gets it right the first time.

Here are some techniques we find helpful in revising:

- Read your work out loud to yourself. Ah, you found a typo, didn't you?
- Find someone to read your script out loud with, to prototype the two-part interaction (Thanks to Erika Hall for this advice - abookapart.com/products/conversational-design). Where does it sound unnatural? Forced?
- Rewrite as though you're writing to a friend. Does that sound more natural or human?
- Record yourself verbally explaining the concepts you're trying to convey. This may help you find a more casual way of expressing your app's message.

After a few rounds of revision, your app should be ready to help your users accomplish tasks in Slack, while avoiding confusion, frustration, and consternation.

Congratulations!

Try it out

- Brainstorm and list five adjectives that describe your app's personality and tone.

- Write a message to send to the installer of your app. Make sure to include the purpose of your app and how to use it. Keep in mind the audience you defined last chapter.

Outline Your Use Cases

Now is the time to figure out how your app will help make other people's working lives simpler, more pleasant, and more productive.

Maybe your app can save people time by posting notifications from your service into Slack, reducing the number of times they need to switch between apps. Maybe your app provides a quicker way for people to get common tasks done — like approving expense reports or assigning tasks among a team.

You've got options

In order to decide which workflows make sense to hook into Slack, it's helpful to know what you can do with Slack's APIs. You can:

- Send rich, interactive messages with the Web API.
- Respond to activities in Slack using the Events API.
- Prompt users to take action on a message with message actions.
- Empower users to invoke workflows at will using slash commands.
- Make messages interactive with buttons, select menus, and date-pickers.
- Collect form-like data using dialogs.

Message structure

Messages are a central interaction point between users and your app. Your app's messages are vehicles for all kinds of content and experiences.

Think about what data you can include in a message that will help people be productive after they receive it. Maybe the text of a notification is enough; maybe you want to link out to your service for more information, or let people take actions from within Slack. Match your message layout to the work you want people to get done.

App Name

APP 10:23 AM

This section block just contains plain text. Perfect for sending all kinds of words!

A Link

Sections can also contain markdown

bold *italic* `code` ~~strikethrough~~

Link to a @user or #channel, too

Context blocks hold images and text

See that little grey line above? That's a divider block.

Section blocks

can have action elements or images as accessories.

Button

Action elements

include buttons, select menus, datepickers, and overflow menus

Select ▼

Action

Blocks

Also Hold ▼

Elements

...

Section accessories to the right



Design Tip: Use the Block Kit Builder (api.slack.com/tools/block-kit-builder) to quickly prototype your app design or modify an existing design template.

Building blocks of a message


While plain text messages may be suitable at times, using blocks will open your messages up to new possibilities — displaying images, providing contextual information, and adding interactive elements like buttons and date-pickers.

Blocks let you add rich layouts that structure data in an easily readable and understandable way. While you may not use every single block, it's helpful to know the tools your messages have at their disposal:

Section

BlockKit APP 12:51 PM

Section blocks accepts a plaintext or markdown element in addition to an optional accessory that can be an image or action element. 😊 Try to keep formatting simple, and remember that messages will look slightly different on mobile devices than they do on a computer screen



As one of the most flexible blocks, section blocks can be used for text, in combination with fields, or side-by-side with block elements like images, buttons, or date-pickers.

Image

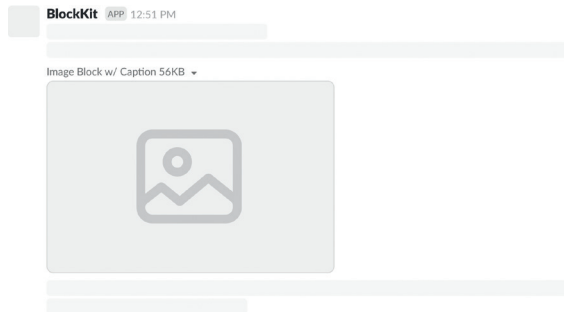


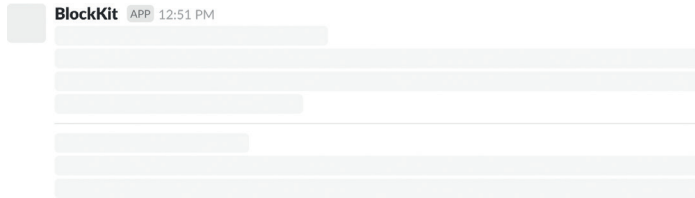
Image blocks are used to display full-sized images in a message (including a title).

Context



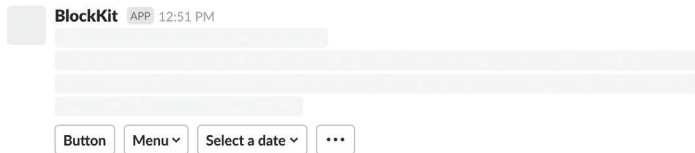
Context blocks display smaller images and text to provide contextual information within a message.

Divider



Divider blocks are used to separate distinct parts of a message with a small grey dividing line.

Actions



The actions block can hold up to five interactive elements. These elements include buttons, select menus, date-pickers, and overflow menus.



Development Tip: Blocks can be used anywhere you send a message to Slack, whether it's using the Web API or webhooks. Dive deeper into block specifics on api.slack.com/messaging.

Adding interactivity to messages

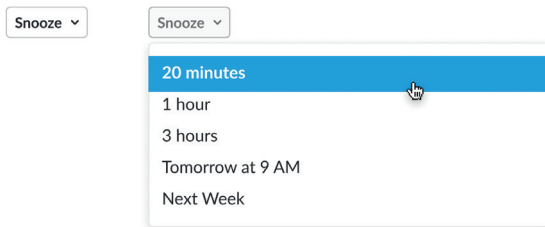
Interactive components can be used in actions blocks and as accessories in section blocks. They allow people to take immediate action on a message in Slack.

Button



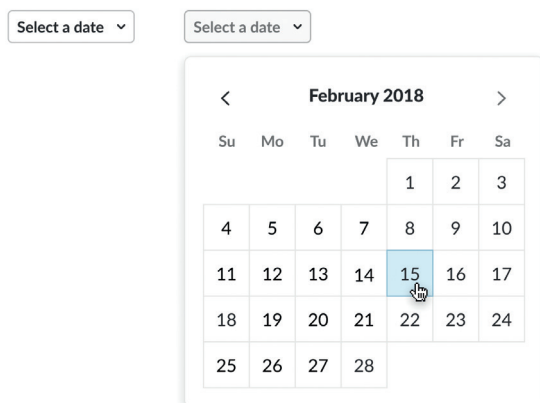
Buttons provide an interface for simple interactions.

Select Menu



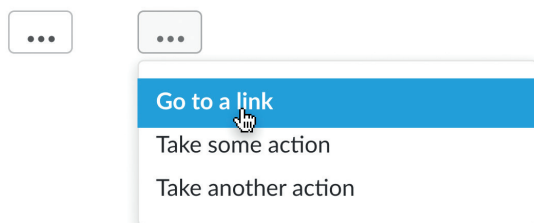
Select menus offer a list of options for users to pick from, with a type-ahead text field to narrow down the options. The options can be predetermined, auto-filled with Slack users or conversations, or pulled in from an external data source.

Date-picker



Date-pickers allow users to pick a date from a calendar UI.

Overflow menu

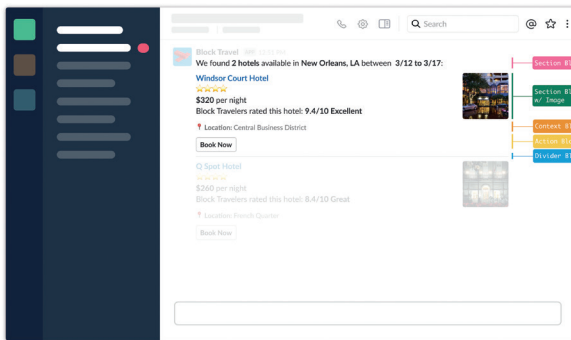


Overflow menus offer a predetermined set of actions displayed in a compact list.

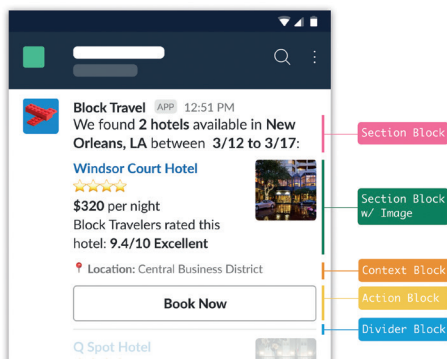
Breaking a message down

To give you a more concrete grasp of how blocks work together to compose messages, let's use an example. This is a notification from a travel app that shows search results for hotels available in New Orleans between March 12 and March 17. The different blocks used to compose the message are labeled to the right.

Desktop



Mobile



Collecting information from users

Dialogs allow apps to quickly collect form-styled input from users rather than trying to prompt for it through a conversational interface. They must be triggered from a user invocation, whether it be interactive components like buttons and select menus or other invocations like message actions and slash commands.

Dialogs allow you to use text elements, text-areas, and select menus:

- **Text elements** are single-line plaintext fields.
 - **Text-areas** are multi-line plaintext fields. These are helpful when you are expecting a relatively long answer from your users.
 - **Select menus** allow users to select items from a list. Items in the select menu can be static, generated from users or conversations in a workspace, or fetched from an external source.
- ✨ **Thought Starter:** When quickly prompting users for action, it's best to use **interactive components** in your app's messages. But if your app needs more input, like writing ticket details or filling out daily stand-ups, **dialogs** make more sense for users and keeps channels clutter free.

Here's a breakdown of a dialog to show you how each element can be used:

The dialog box has a title bar with a close button (X) and a title "Dialog Title". It contains three sections:

- Text input:** A single-line text field with the placeholder text "Text inputs are best for shorter input".
- Select menu:** A dropdown menu with the placeholder text "Select menus lets users pick from a static or dynamic dropdown list" and a downward arrow.
- Textarea:** A multi-line text area with the placeholder text "Textareas are perfect for input from users that you expect to be longer than regular old text inputs".

At the bottom, there is a link "Learn more about Help Desk" with a question mark icon, and two buttons: "Cancel" and "Submit".

Try it out

- Pretend you're designing a message that lets users take a poll and displays the poll results. Draw the message below using the app stencil included with this book. *(Hint: there's an example of a polling app in the **App templates** section of the Appendix)*

Storyboard Your Interactions

Storyboarding is a great way to help you figure out the best flow for interactions in your app. Taking it one step at a time, let's walk through designing interactions that will help your users get work done efficiently inside of Slack.

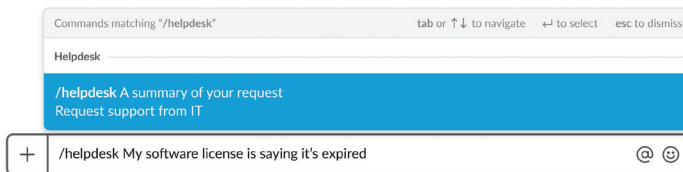
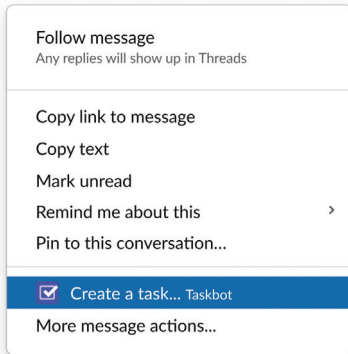
Interactions have a life-cycle


Users will be interacting with your app asynchronously. They may begin a task, then get interrupted - and your app should plan for that. Interruptions or periods of inactivity may or may not signal a loss of intent on the part of the user.

While you're storyboarding, decide whether anything in your workflow is time-sensitive. Should someone be able to jump in on the task again at any time? Should some content expire or need to be re-done? Make a conscious choice about the lifespan of your message.

Pick entry points for interactions

Put yourself in a user's shoes: for the task you're trying to help them complete. Will it be easiest for them to initiate it by using an action in the message overflow menu, by calling a slash command, or by interacting with your messages?

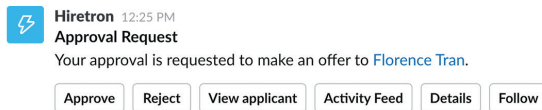


 **Design Tip:** Slash commands are intuitive for a technical audience, but less usable for general audiences. If you're designing for more than one kind of user, giving your app more than one entry point can improve the user experience across the board.

There is no one correct answer. There may be a different answer for each user story (e.g., admins should be able to approve requests in Slack; users should be able to get help from us when they need it). It is always worth the time to explore different possible ways of guiding a person through getting this work done, and *picking whatever is simplest for your end user*.

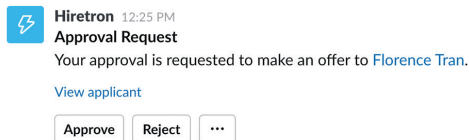
Keep messages glance-able

Blocks allow you to compose visually rich messages with interactive elements such as buttons, menus, and date pickers. With the ability to use so many different blocks in different ways, it's easy to overload a message. For example, take a look at this message:



While all of those actions are valuable, it's difficult to intuit the most important action to take on the message.

Interactions should focus on what members are trying to do right then and there. Focus on the simple, common tasks that make sense to complete inside of Slack conversations. In this case, we could use the overflow menu to de-emphasize the less important actions.



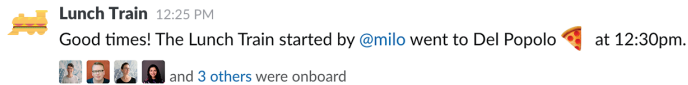
Keep text segments bite-sized and conversational

People don't read big blocks of text. They do read short ones.

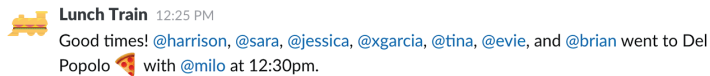
Pictures are worth a thousand words


Sometimes faces can be better than names, or maps better than addresses. Sprinkle image elements into blocks to remove some of that text.

Like this



Not like this

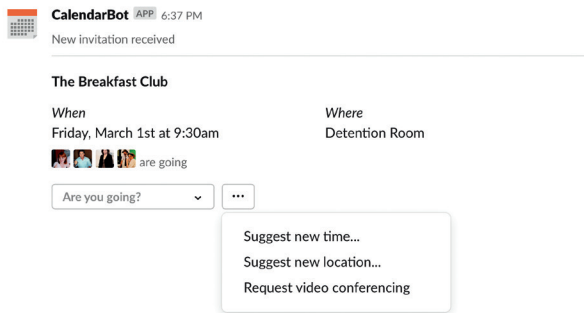


 **Design Tip:** The “Like this” image above is using a context block, which is great for storing information that may be helpful in a message, but isn’t primary content. Context blocks can store text, images, and emoji.

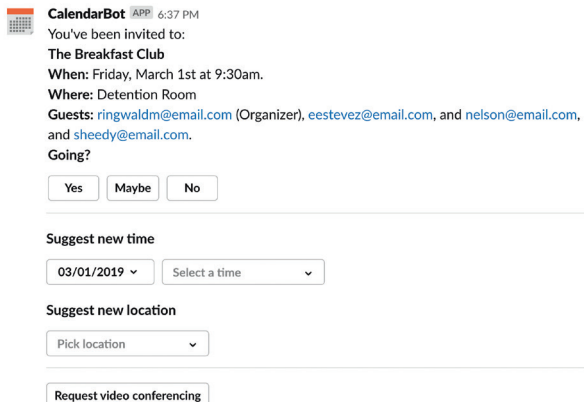
Use interactivity to reduce complexity

The workflow within apps might be complex, with many options along the way, but the messages they publish shouldn't be. Use interactive elements in messages to break workflows into steps, and only show what's needed for the current step. Let interaction choices reveal further information or options when they're necessary, not before.

Like this

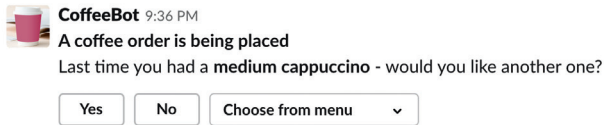


Not like this



Choose sensible default options

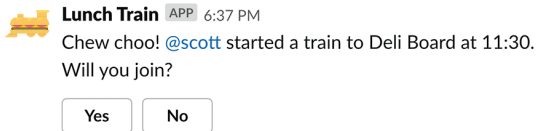
Save people work wherever you can by minimizing the choices they have to make. When you give menus and buttons good default values, you decrease the number of choices they have to make from many to just one - yes or no.



Say your app helps people buy coffee. Instead of presenting a full menu of choices every time someone orders, you could make the user's last order the default option. In the best case scenario, this reduces the coffee order to a single click.

Cleaning up after your app

Visually rich messages are great in the moment you receive them: they're easy to read, nice to look at, and simple to interact with. They also take up a lot of space on a person's screen.



While you're, think about what a person will need to remember about their interaction with your app when they come back to it later, at the end of the message's life— or after an exchange of several messages. Do those buttons and menus need to stick around, or can you condense the message down to a simple text record of what happened?

Be considerate and update your message after the interactive flow or the conversation expires.



Lunch Train APP 6:37 PM

The train started by @scott has left the station!
You chose not to join this time.



Development Tip: The `chat.update` method can be used to update an existing message. You'll need to pass the original message `ts` (it's in the response payload when you send a message) so make sure to keep that for later.

Try it out

Let's think about your app specifically, about your app's main use-case. If you're designing a helpdesk app, the main use-case is probably submitting helpdesk tickets. If you're designing an app that sends sales leads into Slack, the main use-case is probably sending actionable notifications. No matter what it is, let's design it (we'll help you through it). Use the app stencil included with this book.

Step 1: Pick an entry point

How are users going to initiate with your app? Will they use a slash command? A message action? Will your app post a notification into a channel? Draw the entry point below.

Step 2: Respond to the user

How action should a user take after they initiate your app? Should they interact with actions in your app's message? Should the user fill out more information in a dialog? Draw the user's interaction below.

Step 3: Complete the flow

What happens after a user takes the action? How should your app respond? Should your app send a message to channel? Should it update an existing message? Draw your app's response after the user takes the action in Step 2.

Being a Good Citizen Inside of Slack

Now that you've storyboarded your interactions, given thought to your messages' format and content, and considered the voice and tone of your app; you're in great shape. These last few things can take your app from being good to delightful by making it more considerate.

Considerate notifications

Notifications can be incredibly useful. Many Slack apps exist primarily to pipe notifications into relevant channels. Here are some Slack-specific considerations that can help ensure the notifications your users want don't turn into noise.



Design Tip: When your app is first installed, allow the installer to easily set notification preferences. Consider settings like the rate of notifications, the channel they should be posted to, and if applicable, the type(s) of notifications your app should post.

Frequency

Consider how frequently your events are generating notifications for a user. When it makes sense, offer the user an option to get a digest of these notifications rather than being alerted for individual events. This is especially important when the notifications are coming from an automatic source like service logs. In extreme cases, sending too many notifications can get your token revoked due to rate limits.



Thought Starter: When your app sends a lot of notifications, users will notice (and might grow annoyed). This can lead them to remove your app from a channel or even to uninstall the app from the entire workspace. One way to mitigate this is by surfacing notification preferences often. Make it simple and intuitive for people to use your app how and when they want.

**Code Bot** APP | 4:00 PMA build error was triggered for [build #112](#)

```
ERROR: InvocationError for command '/build/desktop-app/setup.py' (exited with code 1)
```

Branch

desktop-app/master

Author[@ryan](#)

Build failed at April 9 at 2:32PM

[Stop sending notifications like this](#)

Match notification types and channels

You may already be segmenting or categorizing the types of notifications that users can receive. If your service already does this, you can make the Slack notification experience even better by giving people the option to pipe different notification types into different channels.

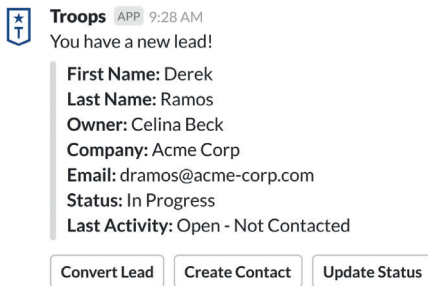
For example, an HR tool might want to pipe notifications about a candidate's background check only into #hr-team, but after an offer has been accepted and signed, maybe a #new-hires channel wants to be notified so they can celebrate.




Design Tip: Don't post to a workspace's #general channel by default. You'll likely be unnecessarily disrupting many users and there is probably a more relevant channel for your app to post in.

Make notifications actionable

Some notifications are best as simple text that alerts a channel that something happened in a 3rd party system. But what if that user wants to dig into your service after receiving your notification? Or what if you send a system status that requires immediate action?



You can save people work by adding action elements directly into your app's Slack messages. Buttons, select menus, and dialogs let people react in the moment and get work done faster. If your app is considerate with alerts and saves people time, they'll likely find your app an essential part of their workday.

 **Thought Starter:** Your actionable notifications don't need to be complex to be helpful. Just think about what action the receiver of the notification most likely wants to take. For example, if your app sends expense approval notifications, the receiver probably wants to approve or deny the expense — interactive buttons are perfect for this.

Sending direct messages to users

First, consider when and whether you should send direct messages (DMs) to specific users. Remember that DMs generate push notifications for mobile users and badges on desktop and mobile. These are useful but also disruptive and can be unexpected, so be cautious with your use of DMs. You should only DM a user when:

- The user DMs you first.
- The user is the only one you're providing a service to, rather than the team.
- Your app is sharing confidential interactions or information.

It's worth a note that users probably assume that information shared back and forth with your app in DM is private. Be aware of any sensitive information that's being shared, and don't surprise users by announcing results of DM conversations in channel without letting them know first.

Responding in channel

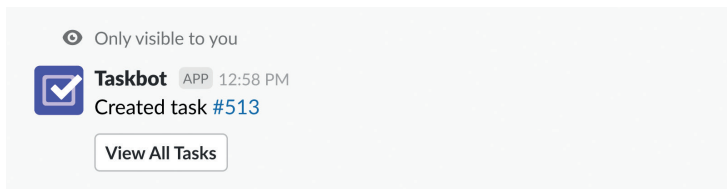
Whatever you post in channel is going to be long-lived and add to the group conversation that people are reading, both in the moment and later when they may look back at what was shared.

Public and ephemeral responses

By default, the response messages sent to actions and commands will only be visible to the user that issued the command (we call these “ephemeral” messages). However, if you would like the response to be visible to all channel members where the user typed the command you can add a `response_type` of `in_channel` to the JSON response:

```
{
  response_type: "in_channel",
  text: "It's 70 degrees right now."
}
```

Remember that a chatty app is not necessarily a good thing, so only pick responses that are important to the entire team for this type of response. If the response only needs to be displayed to the user, it's a better idea to use `response_type: ephemeral` than it is to generate a DM.



Don't use broad mentions

There are very few cases where your app should send broad mentions like `@channel`, `@here`, or `@everyone`. One exception might be if your app sends a notification for immediate action when a critical system or service goes down. Even then, you should get explicit permission from the installer before your app uses one of these mentions.

Try it out

- Pretend you're designing an app that sends notifications to a `#help-it` channel every time a help desk ticket is submitted in your company. How can you make this message actionable to make it easier for the IT team? Design an actionable notification below using the app stencil included with this book.

Create a good onboarding experience

Your app only has one chance to make a first impression, so it's worth the time to make your onboarding experience a great one.

When people first interact with your app inside of Slack, they may have varying context about your app and what it does:

- They may have used it before
- They may be familiar with your app on another platform, but using it in Slack for the first time
- They may have seen others on their team using the app, but not have used it themselves
- They may know nothing about your app at all

Your onboarding should equip people to get something done as quickly as possible, no matter what context they have. Determine the key tasks you need a user to accomplish in the first 30 seconds of interacting with your app, and design your onboarding to help them get it done.

Say hello

It's a good idea to have an app announce its presence and teach people how to interact with it. There is, however, a fine line between being informative and being spammy. Here's how to do it right:

Have an informative, concise welcome message

When someone installs your app, it's a good idea to DM that person with a welcome message. A welcome message should be concise, clearly state what your app does, and include instructions about how to use it.



donut APP 3:19 PM

Hi @sam! 🍩

I'm here to introduce people from a specific channel who don't know each other well and encourage them to meet for ☕, 🍷, or 🍷 via a group DM.

🔥 To get started `/invite` me to a public channel

💡 I recommend creating one like `#coffee-buddies` or `#lunch-pals`

(Donut works best when everyone can opt in by joining the channel).

Say `/faq` if you have any questions.



Thought Starter: What is the call to action for the installer? Should they add your app to a specific channel? Should they (or their admin) link a 3rd party account? Whatever it is, make the call to action clear and actionable from your welcome message. If it's not, it's likely that the installing user won't configure your app properly and they won't get the full value from it.

It may also be useful to listen to the `app _ home _ opened` Events API event. This sends your app a payload when a user opens your app's DM space. If it's the first time the user opens this space, it's an indication they want to know what your app is and how to use it so it's a good idea to send them a welcome message.

However, you should only send the welcome message the first time a user opens the space, otherwise your app can quickly become noisy and unhelpful.

Introducing yourself to the team

Do not DM the entire team as unexpected messages from an app can prompt uninstallation. Only the installing user should receive a direct message from your app.

If your app has a bot user, it should say hello when it's added to a new channel. In addition to explaining the app's purpose, the bot user should give a quick explanation of how to use your app and what configurations have been set (if any). For example, if your app is going to post a daily update at 10am, that's helpful to know.



Design Tip: If your app has help documentation hosted on your website or if people can DM your app for help, now's a great time to let them know.



Taskbot

APP 12:51 PM

Hello, I'm Taskbot. I'm here to help your team keep track of things you want to get done.

You can create tasks by typing `/task some details about your task` from any channel or clicking "Create a task" in any message's context menu. You can also DM me here to create tasks or share them with teammates or entire channels. Type the name of a task here to get started. 📌

Make deeper dives opt-in

Your welcome message should contain enough information to help someone complete a task for the first time. That said, you may want to help people use your app beyond the first quick-win scenario. Let people choose to have an extended walkthrough for more complex tasks, especially if your app allows users to do more than one thing.



Thought Starter: What is the call to action for the installer? Should they add your app to a specific channel? Should they (or their admin) link a 3rd party account? Whatever it is, make the call to a


Offer help

Onboarding is really about proactively helping people use your app. Though even after a well-designed onboarding experience, users may still have questions about your app or may come back later and forget the onboarding you led them through.

Provide a help action

When slash commands are a central part of your app's experience, it's common practice to provide a help action, e.g. `/myapp help`, that will offer the user assistance. This could be as simple as giving more information about your app and listing your app's commands.

If your app is more conversational, then reply in your app's DM when a user asks for help or your app doesn't understand something a user says to you.


 **Doodle Bot** APP 11:31 AM

Hi @rose, I'm here to help!

- **Start** a new poll by typing `/doodle`, then click Enter!
- **Access & manage** all your polls by typing `/doodle list`
- **Sign up** to your free Doodle account to unlock more options: `/doodle signup`

Leave us feedback or reach the Doodle support team: dbot-support@doodle.com

👉 Here's a quick video to see it all in action: <https://vimeo.com/316571924>

 Vimeo | Doodle

Doodle Bot for Slack

Suggest times for your meeting, invite participants and find the best time for everyone — without leaving Slack! Access and manage your Doodle polls directly where you work. ▼





Design Tip: If you're designing an app with more than one workflow, it can be helpful to offer a select menu in your help message that gives users the option to kick off any of your app's workflows to see what they are and how they work.

Respond when your app is @mentioned

When your app is @mentioned, a user probably wants to use your app or doesn't know how to use your app but wants to learn. This is another good moment to surface help to the user and allow them to start using your app as quickly as possible.



Development Tip: Listening to app_mention event in the Events API will send you a payload every time someone directly @mentions your bot user.

Help and feedback sometimes look alike

People don't just want to reach you when they're critically stuck. Sometimes, they'll want to give feedback on simple things like the ease of going through a particular workflow, or whether a bot successfully understood their intent. If you have a place to route feedback requests, provide a way for people to get there inside Slack. Only offer a feedback channel if you plan to collect and review feedback.

Make your app visible

There are a few things you can do before your app is even installed to help potential users discover and install your app.

App suggestions

If they post a link from your website into channel, there's a good chance they might want to install your app. As part of the app directory, your app can suggest users install your Slack app when links associated with your domain are posted in a channel. This helps users less familiar with the App Directory or Slack apps in general find your app in the first place.



Jaime 10:23 AM

Here's the link to the task board for the project –
<https://taskbot-app.com/board/01293812>



Only visible to you



Slackbot 10:23 AM

That looks like a Taskbot link. Would you like to install the Taskbot app from the [Slack App directory](#)?

Taskbot

Create and manage tasks in Slack.

[Learn more](#)



Development Tip: You can find HTML code to embed on your website on your app management page. Or learn more about app suggestions at <https://api.slack.com/docs/slack-apps-suggestions>.

Install directly from the App Directory

Direct install creates less friction for people thinking about installing your app. Instead of redirecting them from the App Directory to your site to install the app, you can provide a **Direct Install URL**, which simply redirects the current user to Slack's OAuth authorize step directly.



Development Tip: The steps to develop a direct install authorization flow can be found on the API site at https://api.slack.com/slack-apps#direct_install.

Diving deeper into onboarding experiences

If you want to explore more examples of onboarding experiences inside of Slack, checkout the Onboarding experiences section in the Appendix.

Try it out

- Review the **Offer help** section. Now, using your own app as an example, design the message sent to your app's users when they ask for help. Draw it below using the app stencil included with this book.

Appendix

App templates


Templates are designed around common workflows that apps implement on the Slack platform. Whether you're designing simple notifications or something more complex, app templates give you a designer-approved template to base your app on.



Design Tip: All of the app templates are on the Block Kit Builder, where you customize them to your app's needs. Design away at <https://api.slack.com/tools/block-kit-builder>.

Approval templates


Example messages for receiving and responding to requests.

 **Approvals** APP 9:00 PM

You have a new request:

[Fred Enrriquez — Time off request](#)

Type: Paid time off
When: Aug 10–Aug 11
Hours: 16.0 (2 days)
Remaining balance: 32.0 hours (4 days)
Comments: "Family in town, going camping!"

 **Approvals** APP 9:00 PM

You have a new request:


[Fred Enrriquez - New device request](#)

Type:	When:
Computer (laptop)	Submitted Aug 10
Last Update:	Reason:
Mar 10, 2015 (3 years, 5 months)	"All vowel keys aren't working."
Specs:	
"Cheetah Pro 15" - Fast, really fast"	




Notification template


Example message for getting updates on new info and taking relevant action.

**Agenda** APP 10:22 AM

Looks like you have a scheduling conflict with this event:

Iris / Zelda 1-1
Tuesday, January 21 - 4:00–4:30pm
Building 2 - Havarti Cheese (3)
2 guests



 Conflicts with Team Huddle - 4:15–4:30pm

Propose a new time:

Today - 4:30–5pm
Everyone is available: @iris, @zelda

Tomorrow - 4–4:30pm
Everyone is available: @iris, @zelda

Tomorrow - 6–6:30pm
Some people aren't available: @iris, @zelda

[Show more times](#)


Choose

Choose



Choose



Polling template


Example message for respond to a poll and seeing current results.

**Polls** APP 10:22 AM

Where should we order lunch from? Poll by @mm

**Ace Wasabi Rock-n-Roll Sushi Bar**
 3 votes

**Super Hungryman Hamburgers**
 2 votes

**Kagawa-Ya Udon Noodle Shop**
No votes

Vote


Vote

Vote

Add a Suggestion


Search esult templates

Example messages for seeing top search results and browsing additional info.



TripAgent
APP 10:22 AM

We found 205 hotels in New Orleans, LA, from 12/14 to 12/17: ...


Windsor Court Hotel
 ★★★★★
 \$340 per night
 Rated: 9.4 - Excellent
 Location: Central Business District




The Ritz-Carlton New Orleans
 ★★★★★
 \$340 per night
 Rated: 9.1 - Excellent
 Location: French Quarter




Omni Royal Orleans Hotel
 ★★★★★
 \$419 per night
 Rated: 8.8 - Excellent
 Location: French Quarter



Next 2 Results


FileCard Agent
APP 10:22 AM

 Search results for Cata

Use Case Catalogue
 Use Case Catalogue for the following departments/roles...

Manage ▾

Customer Support - Workflow Diagram Catalogue
 This resource was put together by members of...

Manage ▾

Self-Serve Learning Options Catalogue
 See the learning and development options we...

Manage ▾

Use Case Catalogue - CF Presentation - [June 12, 2018]
 This is presentation will continue to be updated as...

Manage ▾

Comprehensive Benefits Catalogue - 2019
 Information about all the benfits we offer is...

Manage ▾

Next 5 Results

Onboarding Experiences

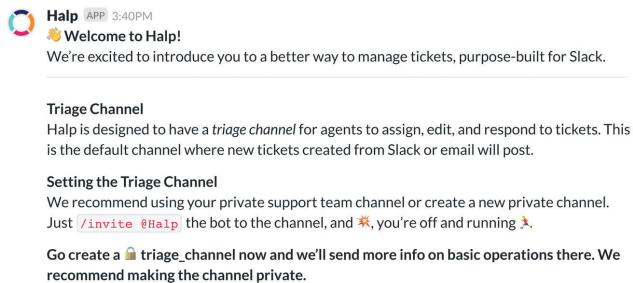
This section details onboarding experiences of Slack apps out in the wild. We've included screenshots and descriptions to highlight parts of the experiences you may be able to use for your own app's onboarding design.

Halp

Halp lets you assign, prioritize, and answer internal ticketing requests from Slack with ease.

Welcome Message

Message sent to user when the app is installed



Tell users what your app does

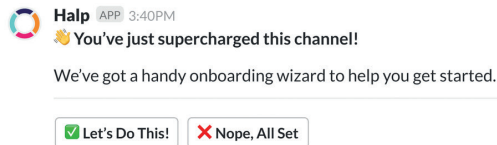
- This welcome message clearly tells the installing user what the app is for and explains app-specific terminology (like “Triage Channel”)

Let users know how to use your app


- This welcome message tells users the first step in using your app.
- It also does a good job of giving the user usage recommendations, like making the channel private since support tickets will be sent there

App Added to Channel

Sent when the app is invited to a channel



Sent when a user opts-in to the onboarding walkthrough

 **Halp** APP 3:40PM

There are two simple ways to create tickets from Slack.

1) Using a Slash Command
Users can initiate tickets with the `/halp` command anywhere in Slack. You can also set this up to launch a form inside Slack to better categorize requests.

2) Using an Emoji
If a user DMs you or posts into a public channel such as #it-help, you can just add the 🛠️ emoji to their message to create a ticket. (ps this is customizable 🗨️)

Let's Try it Out!
Looks like Jim is having some issues (classic Jim). Go ahead and react to his message with a 🛠️. We'll simulate the end-user experience in the thread of his original message and agent experience posted below it. Try responding in both to see how they sync.

[👉 START DEMO 👈](#)

Jump to Section ▾

NEXT ➡

Make onboarding optional

Halp lets you assign, prioritize, and answer internal ticketing requests from Slack with ease.

Welcome Message

- The first message makes it clear that onboarding is optional. If a user has used the app before and doesn't need a refresher, they can skip the walkthrough.
- The second message allows you to skip around to different sections of the onboarding walkthrough. This makes it easy to find the part(s) most relevant to how you want to use the app in Slack.

Let users know how to use your app


- This clearly states the different ways you can use the app in Slack. It also gives users more than one option — while some users may prefer using slash commands, others may not (and this app works for both types of users).

Woobot

Woobot connects your teams in Slack with their important Salesforce data so they can get things done.

Make onboarding optional

Message sent to channel when user installs app

**Woobot** APP 11:57 AM

Hey there @carole! I'm Woobot.
I'm here to make doing work in Slack easier.

Can Work Be Fun Again? Yes it can.

With [Woobot.io](#), anyone can power their teams in Slack with their important Salesforce data.

We've brought together custom workflows, powerful integrations, and a full set of commands you can use to map to your Salesforce workflows to Slack so your team can perform easy and targeted updates where you already work.

If you search it in Salesforce, Woobot can surface it, and you control what is editable.

Performing magic with me starts by setting things up properly — so [these docs are here](#) to help you learn about all the fun stuff you can do with Woobot.

Here are some commands you should know

Put me to work anytime by typing `/woobot`

If you need help at any time, just type `/woobot-help`

To request support or provide feedback reach us at `/woobot-feedback`

We also have a ton of resources to help you on our [Woobot Docs Page](#).

`/woobot`

`/woobot-help`

`/woobot-feedback`

Tell users what your app does

- This welcome message clearly states the purpose of what the app is and why it was added to the team.

Let users know how to use your app

- This welcome message shows users how to start using the app by providing the slash commands users have access to in Slack.
- The message also provides buttons inline with the command descriptions to give users a simple way to try the commands out right from the welcome message.

Give users support

- This welcome message has two clear ways to get support — by using the /woobot-feedback command or going to their docs page to learn more.

App Configuration

Message sent to installer when app is installed



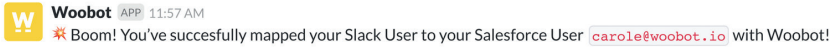
Woobot APP 11:57 AM

Ok! We're almost done setting things up!

First, you'll need to connect to Salesforce by signing into the [Woobot Brain](#). Then follow [these instructions](#) to connect.

Woobot Brain

After user configures 3rd party service



Linking 3rd party services

- If you need users to link a 3rd party service, you can DM the installing user and guide them to your service.

Always close the loop

- The second message is sent when the user is finished authenticating with the 3rd party service. This is important so the user can verify they successfully linked their account when they return to Slack.
- After the loop is closed, the app sends a welcome message similar to the one in channel that shows the user how to start using the app. This is important so that the user knows how to start getting value out of your app.

App stencil

A stencil is included with this book that includes all of the major interface elements you can use with your app — everything from buttons to dropdowns to emoji reactions. You can use the stencil to sketch out and storyboard the basic flow of an app before committing any design or development resources to get an idea of how your app will feel and discover pieces that might be missing.

At first glance, it might look like a template for just one kind of app, but the idea is you can mix and match the pieces to suit your interface. If your app just needs a button to add some action to a notification, feel free to keep it simple. If you're building a more robust back-and-forth workflow with inputs, inline images, and a dialog, use them all. The stencil also includes width guides for various elements, like the dialog modal or a divider, so it's easier to sketch out those flows of your app too.

And if you don't have access to the stencil or you want to see the code that composes your prototyped message, you can use the Block Kit Builder online at api.slack.com/tools/block-kit-builder.

Notes

A blank canvas to storyboard your app, jot down notes, or just doodle

Notes

Notes

the 1990s, the incidence of *S. flexneri* has increased in the United Kingdom [10]. In the United States, *S. flexneri* has been reported to be the most common serotype of *Shigella* isolated from children with shigellosis [11]. In the United Kingdom, *S. flexneri* is the most common serotype isolated from children with shigellosis [12].

There is a paucity of data on the epidemiology of *S. flexneri* in the United Kingdom. In the 1980s, *S. flexneri* was the most common serotype isolated from children with shigellosis in the United Kingdom [12]. In the 1990s, the incidence of *S. flexneri* has increased in the United Kingdom [10].

In the United States, *S. flexneri* has been reported to be the most common serotype of *Shigella* isolated from children with shigellosis [11]. In the United Kingdom, *S. flexneri* is the most common serotype isolated from children with shigellosis [12].

There is a paucity of data on the epidemiology of *S. flexneri* in the United Kingdom. In the 1980s, *S. flexneri* was the most common serotype isolated from children with shigellosis in the United Kingdom [12]. In the 1990s, the incidence of *S. flexneri* has increased in the United Kingdom [10].

In the United States, *S. flexneri* has been reported to be the most common serotype of *Shigella* isolated from children with shigellosis [11]. In the United Kingdom, *S. flexneri* is the most common serotype isolated from children with shigellosis [12].

There is a paucity of data on the epidemiology of *S. flexneri* in the United Kingdom. In the 1980s, *S. flexneri* was the most common serotype isolated from children with shigellosis in the United Kingdom [12]. In the 1990s, the incidence of *S. flexneri* has increased in the United Kingdom [10].

In the United States, *S. flexneri* has been reported to be the most common serotype of *Shigella* isolated from children with shigellosis [11]. In the United Kingdom, *S. flexneri* is the most common serotype isolated from children with shigellosis [12].

There is a paucity of data on the epidemiology of *S. flexneri* in the United Kingdom. In the 1980s, *S. flexneri* was the most common serotype isolated from children with shigellosis in the United Kingdom [12]. In the 1990s, the incidence of *S. flexneri* has increased in the United Kingdom [10].

In the United States, *S. flexneri* has been reported to be the most common serotype of *Shigella* isolated from children with shigellosis [11]. In the United Kingdom, *S. flexneri* is the most common serotype isolated from children with shigellosis [12].

