


API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

SECRET KEY	CREATED	LAST USED	
sk- . . . k6Pk	30. Jan. 2023	Never	

+ Create new secret key

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Personal 

Note: You can also specify which organization to use for each API request. See [Authentication](#) to learn more.

Organization settings

Organization name

Human-friendly label for your organization, shown in user interfaces

Personal

Organization ID

Identifier for this organization sometimes used in API requests

org-xxxxxxxxxxxxxx

Save

Changing model

In the file "bot/teamsBot.js" on line 33 you have the option to change the model used.

```
31 const response = await openai.createCompletion({  
32   // replace with the model you want to use (overview of all available models: https://platform.openai.com/docs/models)  
33   model: "gpt-3.5-turbo",  
34   prompt: txt,  
35   max_tokens: max_tokens,  
36   temperature: 0.5,  
37   user: user,  
38 });  
39
```

You can find a list of all models from OpenAI here:

<https://platform.openai.com/docs/models>

Prerequisites

5 minutes remaining

Here's a list of tools you need for building and deploying your apps.

Install	For using...
Required	
Teams Toolkit	A Microsoft Visual Studio Code extension that creates a project scaffolding for your app. Use 4.0.0 version.
Microsoft Teams ↗	Microsoft Teams to collaborate with everyone you work with through apps for chat, meetings, call - all in one place.
Node.js ↗	Back-end JavaScript runtime environment. Use the latest v16 LTS release.
Microsoft Edge ↗ (recommended) or Google Chrome ↗	A browser with developer tools.
Visual Studio Code ↗	JavaScript, TypeScript, or SharePoint Framework (SPFx) build environments. Use version 1.55 or later.
Optional	
Azure Tools for Visual Studio Code ↗ and Azure CLI	Azure tools to access stored data or to deploy a cloud-based backend for your Teams app in Azure.
React Developer Tools for Chrome ↗ OR React Developer Tools for Microsoft Edge ↗	A browser DevTools extension for the open-source React JavaScript library.
Microsoft Graph Explorer	Microsoft Graph Explorer, a browser-based tool that lets you run a query from Microsoft Graph data.
Developer Portal for Teams ↗	Web-based portal to configure, manage, and distribute your Teams app including to your

Install	For using...
	organization or the Teams store.

💡 Tip

If you work with Microsoft Graph data, you should learn about and bookmark the Microsoft Graph Explorer. This browser-based tool allows you to query Microsoft Graph outside of an app.

Prepare development environment


After you've installed the required tools, set up the development environment.

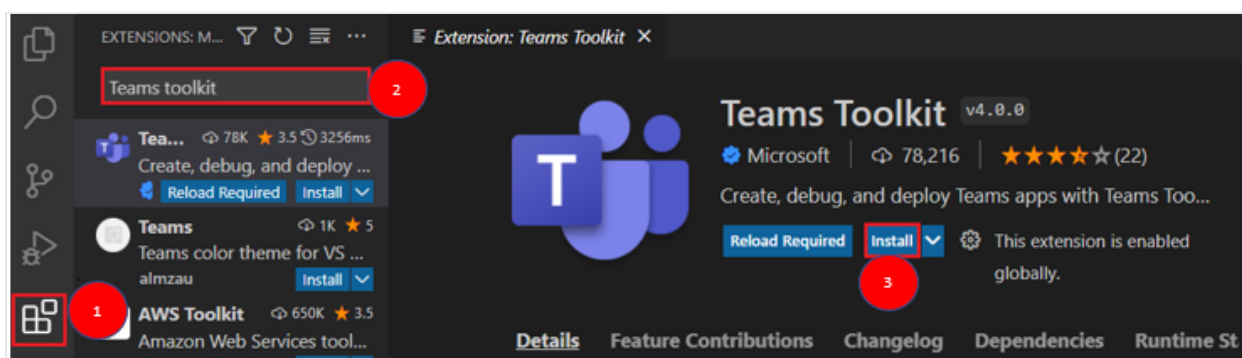
Install the Teams Toolkit

The Teams Toolkit helps simplify the development process with tools to provision and deploy cloud resources for your app, publish to the Teams store, and more.

You can use the toolkit with Visual Studio Code, or CLI (command-line interface), called `TeamsFx`.

Visual Studio Code

1. Open Visual Studio Code and select the **Extensions** view (**Ctrl+Shift+X** /  **↑-X** or **View > Extensions**).
2. In the search box, enter **Teams toolkit**.
3. Select **Install** next to the Teams Toolkit.



The Teams Toolkit icon appears in the Visual Studio Code **Activity Bar** after it's installed.



You can also find the Teams Toolkit on the [Visual Studio Code Marketplace](#).

ⓘ Note

The latest version of Teams Toolkit is 4.2.0.

Set up your Teams development tenant

A **tenant** is like a space, or a container for your organization in Teams, where you chat, share files, and run meetings. This space is also where you sideload and test your app. Let's verify if you're ready to develop with the tenant.

Check for sideloading option

After creating the app, you must load your app in Teams without distributing it. This process is known as sideloading. Sign in to your Microsoft 365 account to view this option.

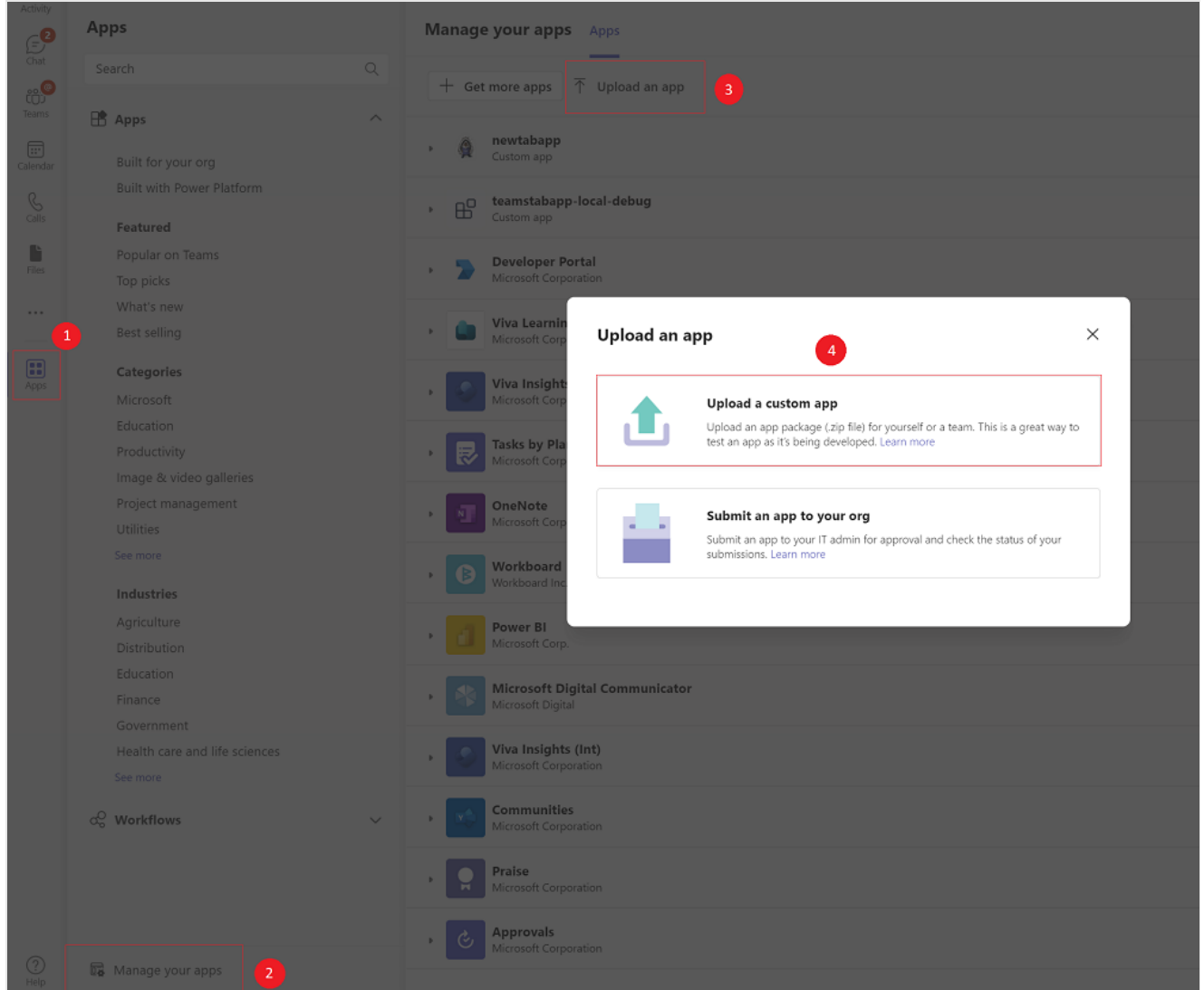
ⓘ Note

Sideloading is necessary for previewing and testing apps in Teams local environment. If it isn't enabled, you will not be able to preview and test your app in Teams locally.

Do you already have a tenant, and do you have the admin access? Let's check if you really do!

Verify if you can sideload apps in Teams:

1. In the Teams client, select the **Apps** icon.
2. Select **Manage your apps**.
3. Select **Upload an app**.
4. Look for the option to **Upload a custom app**. If you see the option, sideloading apps is enabled.



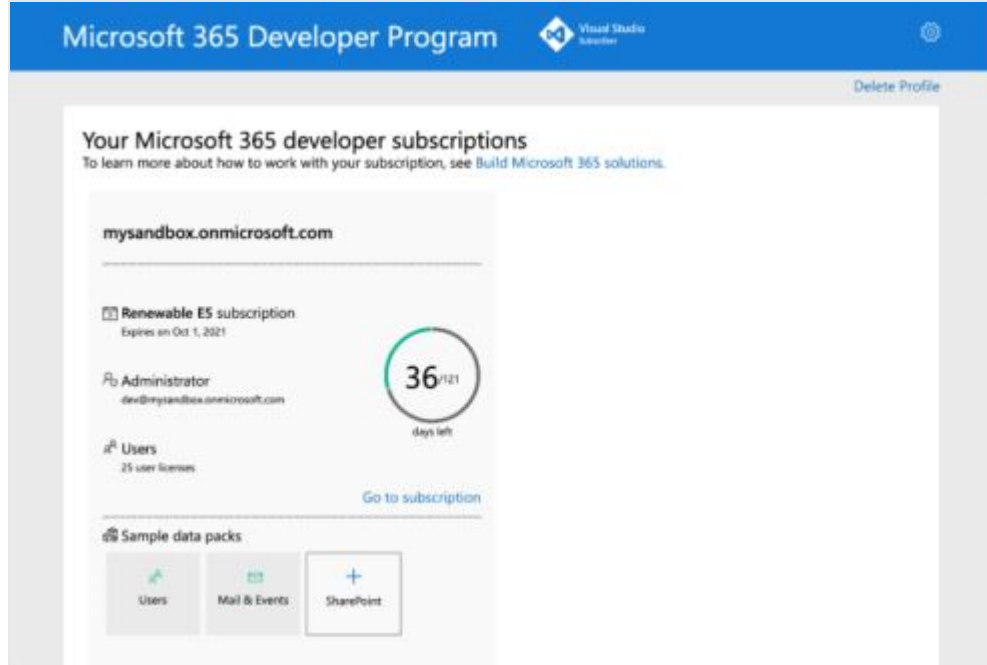
❗ Note

If you don't have the option to upload a custom app, talk to your Teams administrator.

Create a free Teams developer tenant (optional)

If you don't have a Teams developer account, you can get it free. Join the Microsoft 365 developer program!

1. Go to the [Microsoft 365 developer program](#).
2. Select **Join Now** and follow the onscreen instructions.
3. In the welcome screen, select **Set up E5 subscription**.
4. Set up your administrator account. After you finish, the following screen appears.



5. Sign in to Teams using the administrator account you just set up. Verify that you have the **Upload a custom app** option in Teams.

Get a free Azure account

If you wish to host your app or access resources in Azure, you must have an Azure subscription. [Create a free account](#) before you begin.

Now you've got all tools and set up your account. Next, let's set up your development environment and start building! Select the app you want to do first.

[← Previous](#)

Step 1 of 5

[Next →](#)

- ✓ HELLOBOT
 - ✓ .fx
 - > configs
 - > states
 - {}** subscriptionInfo.json
 - > .vscode
 - ✓ bot
 - > .deployment
 - > adaptiveCards
 - > images
 - > node_modules
 - ≡ .env.teamsfx.local
 - 🔒 .gitignore
 - ≡ .webappignore
 - JS** index.js
 - {}** package-lock.json
 - {}** package.json
 - 📘 README.md
 - JS** teamsBot.js
 - > build
 - ✓ templates
 - > appPackage
 - > azure
 - 🔒 .gitignore
 - {}** package.json

Folder name	Contents
<code>.fx/configs</code>	Configuration files that user can customize for the Teams app.
<code>.fx/configs/config.<envName>.json</code>	Configuration file for every environment.
<code>.fx/configs/azure.parameters.<envName>.json</code>	Parameters file for Azure BICEP provision for every

Folder name	Contents
	environment.
<code>.fx/configs/projectSettings.json</code>	Global project settings that apply to all environments.
<code>.fx/configs/ngrok.yml</code>	Ngrok settings
<code>bot</code>	Code for the Bot capability needed at runtime.
<code>bot/teamsBot.js</code>	Main entry point for the bot app.
<code>templates/appPackage</code>	App manifest template files, and the app icons, color.png and outline.png.
<code>templates/appPackage/manifest.template.json</code>	App manifest for running the app in local and remote environment.
<code>templates/azure</code>	BICEP template files

Tip

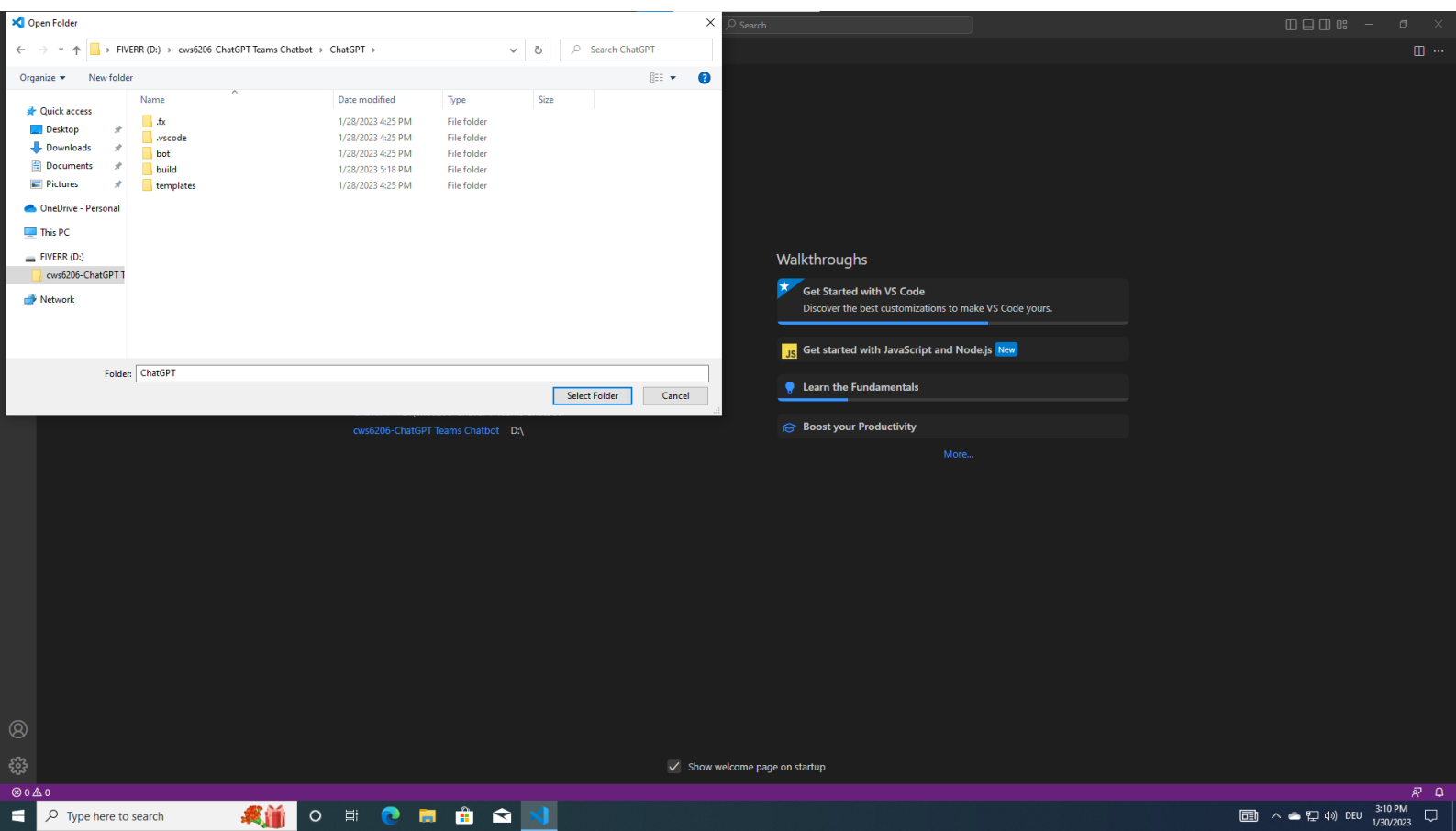
Familiarize yourself with bots outside of Teams before you integrate your first bot within Teams.

[< Previous](#)

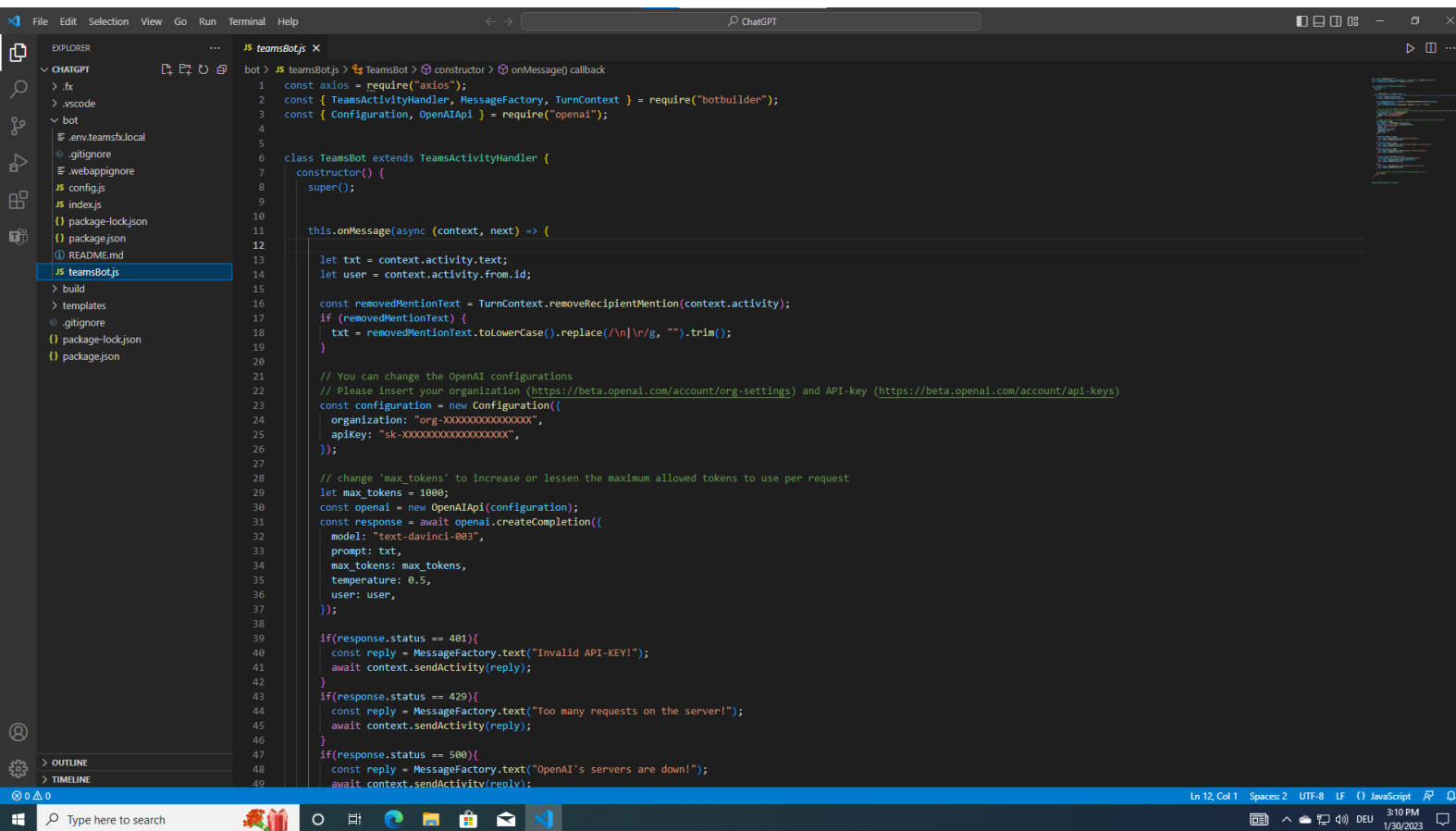
Step 2 of 5

[Next >](#)

Öffnen Sie das Projekt in VS Code



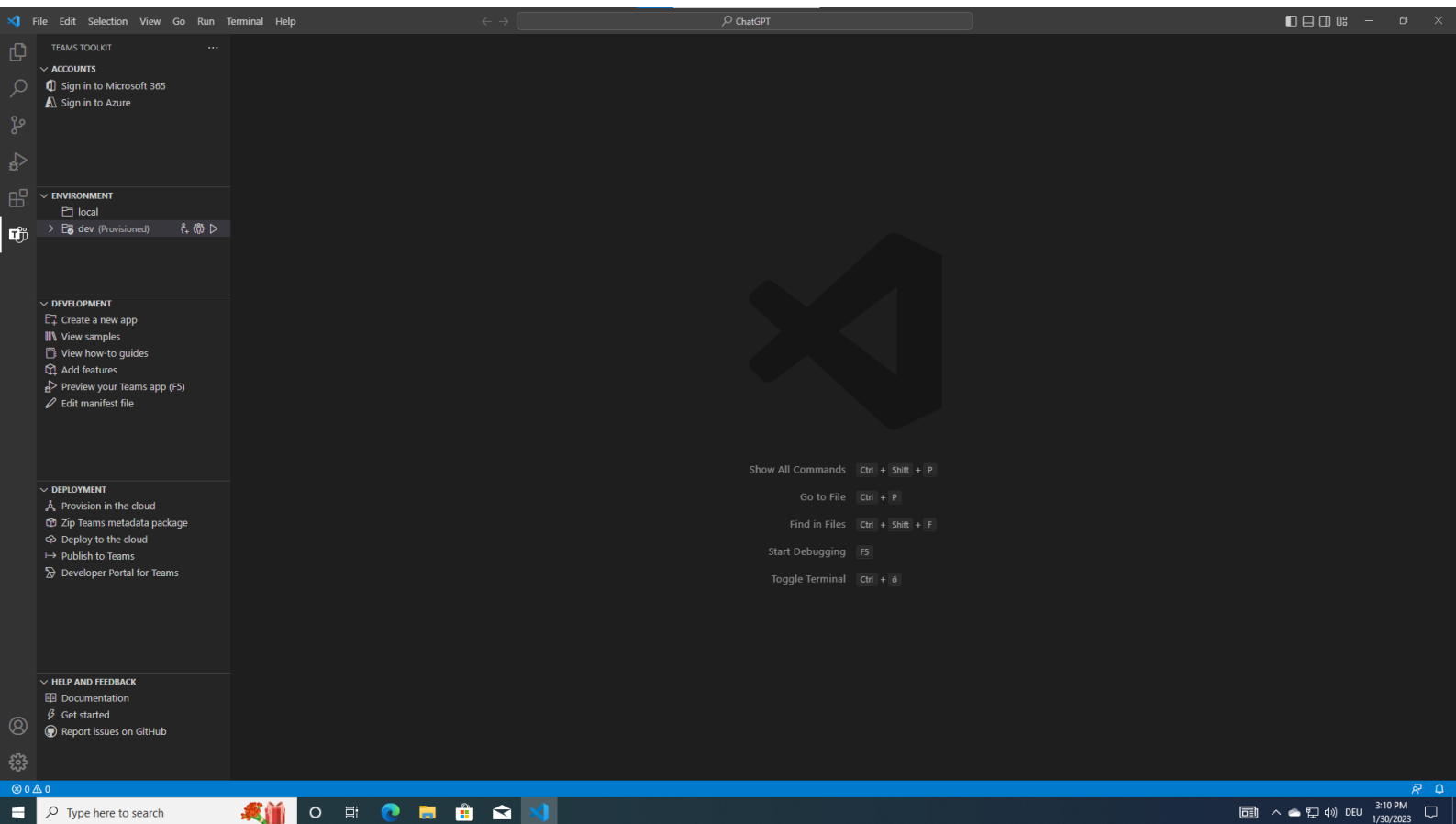
Ändern Sie in der Datei teamsBot.js den Organisation Key und ihren API-Key



```
1 const axios = require("axios");
2 const { TeamsActivityHandler, MessageFactory, TurnContext } = require("botbuilder");
3 const { Configuration, OpenAIApi } = require("openai");
4
5
6 class TeamsBot extends TeamsActivityHandler {
7   constructor() {
8     super();
9   }
10
11   this.onMessage(async (context, next) => {
12
13     let txt = context.activity.text;
14     let user = context.activity.from.id;
15
16     const removedMentionText = TurnContext.removeRecipientMention(context.activity);
17     if (removedMentionText) {
18       txt = removedMentionText.toLowerCase().replace(/\n|\\n/g, "").trim();
19     }
20
21     // You can change the OpenAI configurations
22     // Please insert your organization (https://beta.openai.com/account/org-settings) and API-key (https://beta.openai.com/account/api-keys)
23     const configuration = new Configuration({
24       organization: "org-XXXXXXXXXXXXXXXXXXXX",
25       apiKey: "sk-XXXXXXXXXXXXXXXXXXXX",
26     });
27
28     // change 'max_tokens' to increase or lessen the maximum allowed tokens to use per request
29     let max_tokens = 1000;
30     const openai = new OpenAIApi(configuration);
31     const response = await openai.createCompletion({
32       model: "text-davinci-003",
33       prompt: txt,
34       max_tokens: max_tokens,
35       temperature: 0.5,
36       user: user,
37     });
38
39     if(response.status == 401){
40       const reply = MessageFactory.text("Invalid API-KEY!");
41       await context.sendActivity(reply);
42     }
43     if(response.status == 429){
44       const reply = MessageFactory.text("Too many requests on the server!");
45       await context.sendActivity(reply);
46     }
47     if(response.status == 500){
48       const reply = MessageFactory.text("OpenAI's servers are down!");
49       await context.sendActivity(reply);
50     }
```

Hier können Sie auch die maximale Anzahl an Tokens festlegen

Wechseln Sie auf die Teams Erweiterung links im Menü und Verbinden Sie ihren Microsoft 365 Account und ihren Azure Account



Um die App in Azure zu hosten wählen Sie „Provision in the cloud" und danach „Deploy to the cloud"

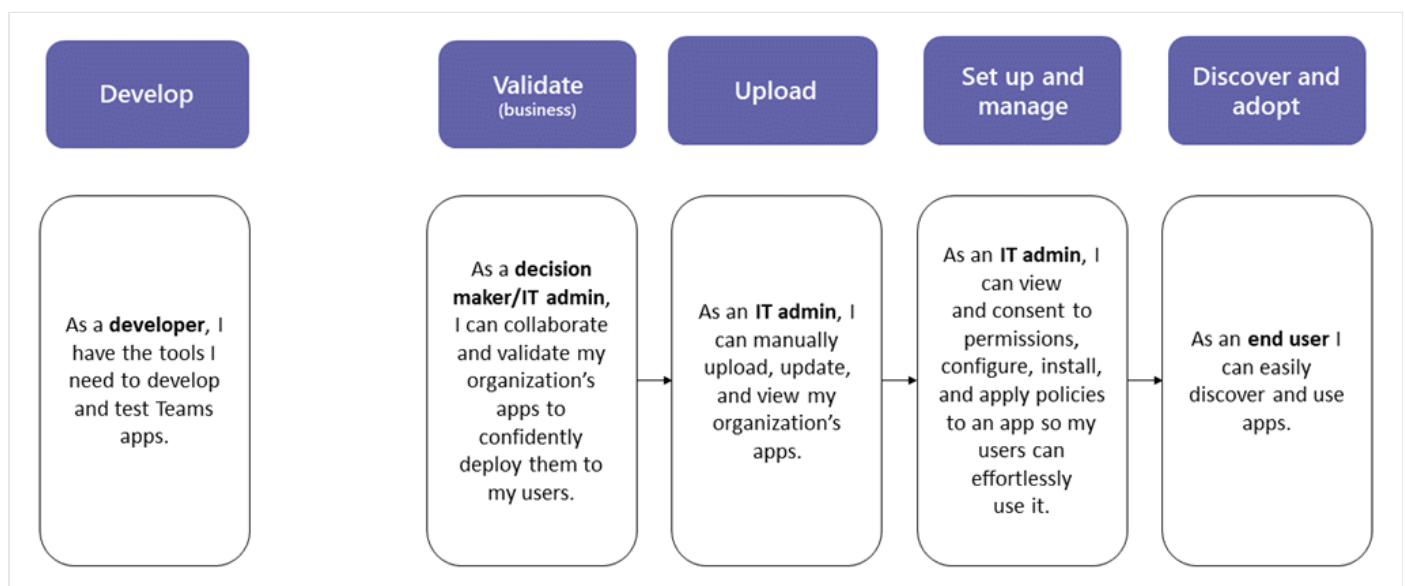
Um die App Package Datei zu erhalten wählen Sie „Zip Teams metadata package"

Publish a custom app by uploading an app package

Article • 11/10/2022 • 6 minutes to read • Applies to: Microsoft Teams

When you publish a custom Teams app, it's available to users in your organization's app store. There are two ways to publish a custom app and the way that you use depends on how you get the app. This article focuses on how to publish a custom app by uploading an app package (in .zip format) that a developer sends you. The other method, approving a custom app, is used when a developer submits an app directly to the [Manage apps](#) page through the Teams App Submission API. To learn more about that method, see [Publish a custom app submitted through the Teams App Submission API](#).

This article provides end-to-end guidance for how to take your Teams app from development to deployment to discovery. This guidance focuses on the Teams aspects of the app and is intended for admins and IT pros. For more information about developing Teams apps, see the [Teams developer documentation](#).



Create your app

The Microsoft Teams developer platform makes it easy for developers to integrate your own apps and services to improve productivity, make decisions faster, and create collaboration around existing content and workflows. Apps built on the Teams platform are bridges between the Teams client and your services and workflows, bringing them directly into the context of your collaboration platform. For more information, go to the [Teams developer documentation](#).

Validate

Receive the app package

When the app is ready for use in production, the developer produces an app package using [Developer Portal](#). The developer shares the app package in .zip format with you.

All apps in Teams store pass a mandatory [app validation](#) to comply with the app quality and security standards of the Teams apps store. In addition, Microsoft strongly encourages app developers to participate in an optional [app compliance program](#) that indicates enhanced compliance, security, and privacy controls. For more information, see [Teams app validation guidelines](#).

Allow trusted users to upload custom apps

To validate that the app is working correctly in your production tenant, you need to allow yourself and/or trusted users to upload custom apps in the production tenant. You use [app setup policies](#) to do this.

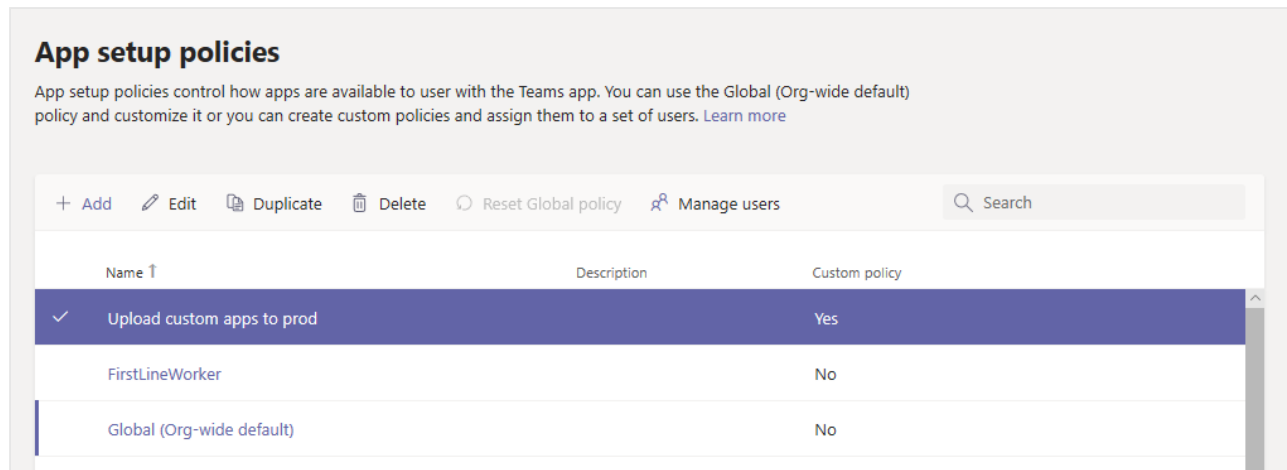
ⓘ Note

If you're uncomfortable with uploading the app to your production tenant for validation, even for yourself or trusted users, you can skip this step and follow the steps in the **Upload and Set up and manage** sections to publish the unvalidated app to your organization's app store. Then, restrict access to that app to only yourself and users you trust. These users can then get the app from your organization's app store to perform validation. After the app is validated, use the same permission policies to open access and roll the app out for production use.

To allow trusted users to upload custom apps, follow these steps:

1. Turn on the **Allow interaction with custom apps** org-wide app setting. To do this:
 - a. In the left navigation of the Microsoft Teams admin center, go to **Teams apps > Manage apps**, and then select **Org-wide app settings**.
 - b. Under **Custom apps**, turn on **Allow interaction with custom apps**, and then select **Save**.
2. Turn off the **Upload custom apps** setting in the global app setup policy. To do this:
 - a. In the left navigation of the Microsoft Teams admin center, go to **Teams apps > Setup policies**, and then select the **Global (Org-wide default)** policy.
 - b. Turn off **Upload custom apps**, and then select **Save**.
3. Create a new app setup policy that allows uploading custom apps and assign it to your set of trusted users. To do this:
 - a. In the left navigation of the Microsoft Teams admin center, go to **Teams apps > Setup policies**, and then select the **Add**. Give the new policy a name and description, turn on **Upload custom apps**, and then select **Save**.

- b. Select the new policy you created, and then select **Manage users**. Search for a user, select **Add**, and then select **Apply**. Repeat this step to assign the policy to all your trusted users.

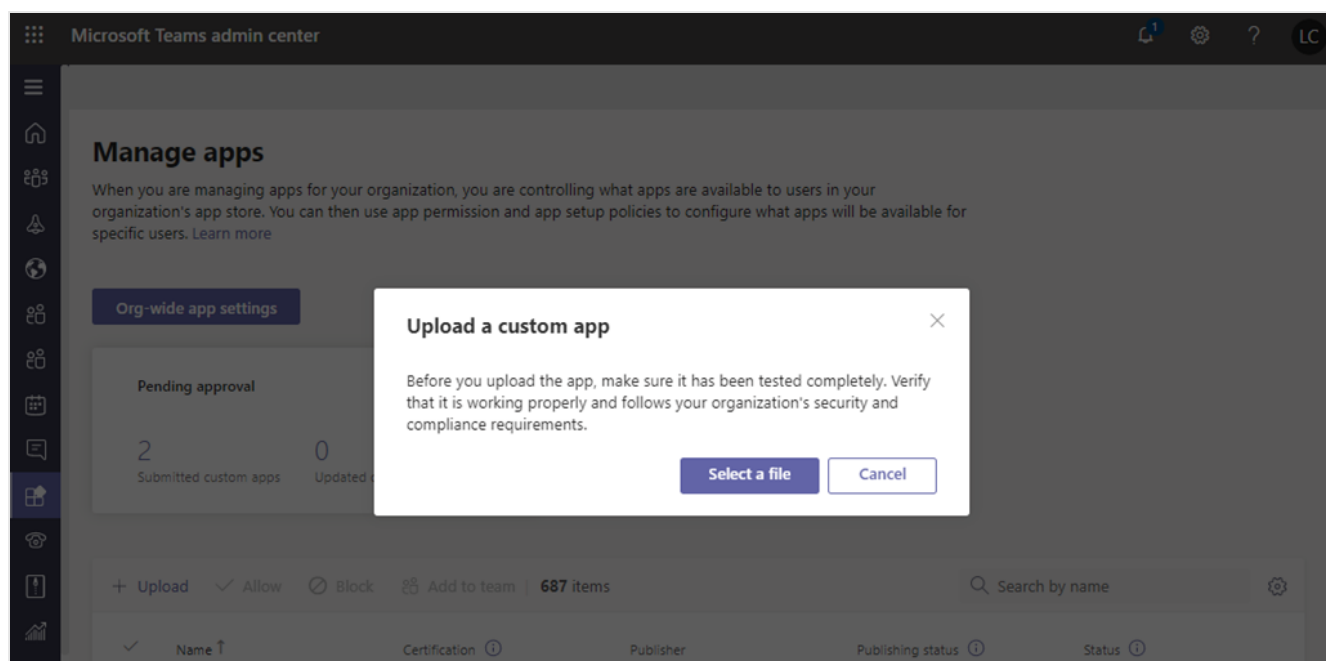


These users can now upload the app manifest to validate that the app is working correctly in the production tenant.

Upload

To make the app available to users in your organization's app store, upload the app.

1. In the left navigation of the Microsoft Teams admin center, go to **Teams apps > Manage apps** [↗](#).
2. Select **Upload**, select **Upload**, select the app package that you received from the developer, and select **Open**.



Set up and manage

Control access to the app

By default, all users in your organization can access the app in your organization's app store. To restrict and control who has permission to use the app, you can create and assign an app permission policy. To learn more, see [Manage app permission policies in Teams](#).

Pin and install the app for users to discover

By default, for users to find the app they have to go to your organization's app store and browse or search for it. To make it easy for users to get to the app, you can pin the app to the app bar in Teams. To pin the app, create an app setup policy and assign it to users. To learn more, see [Manage app setup policies in Teams](#).

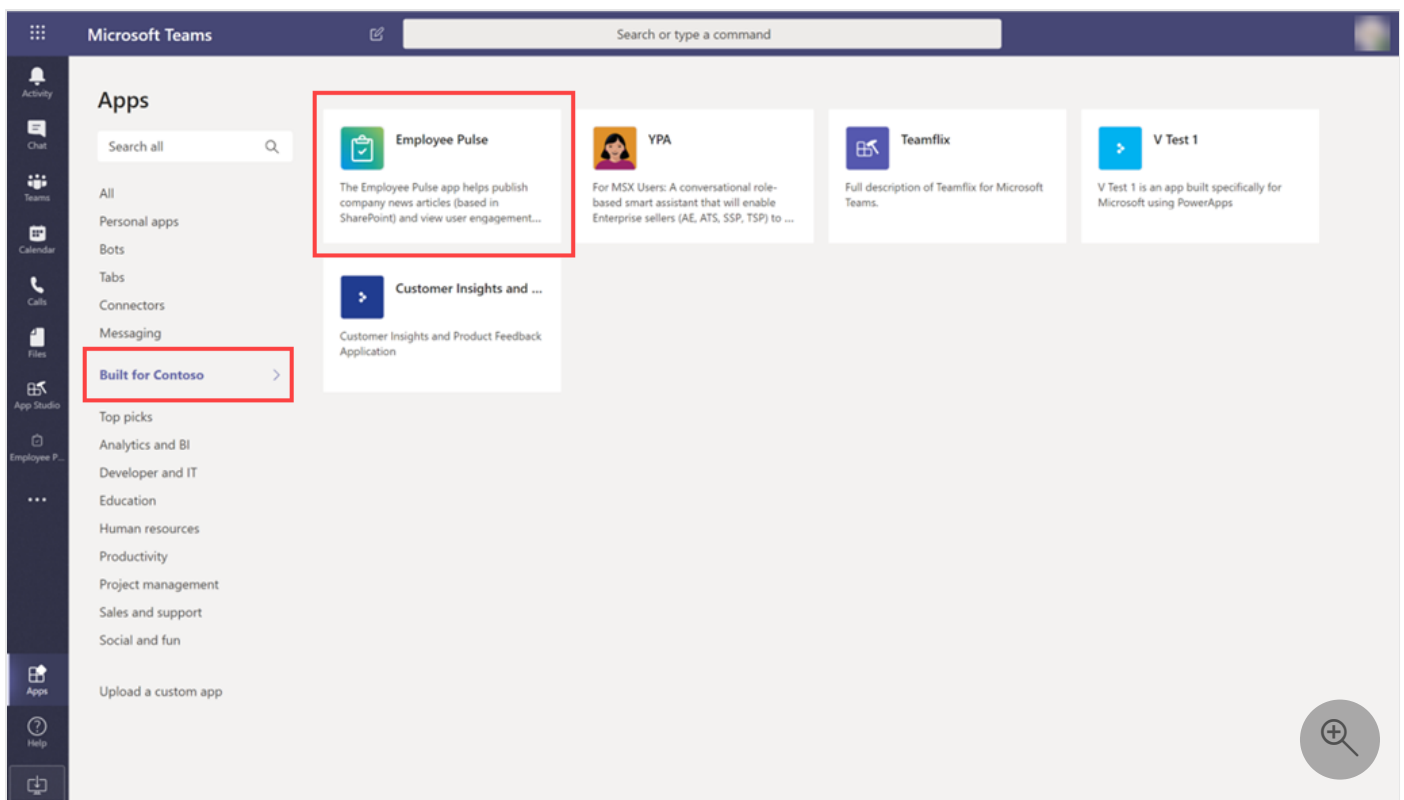
Search the audit log for Teams app events

You can search the audit log to view Teams apps activity in your organization. To learn more about auditing Teams activities, see [search the audit log for events in Teams](#).

Before you can search the audit log, you have to first turn on auditing in the [Security & Compliance Center](#). To learn more, see [Turn audit log search on or off](#). Keep in mind that audit data is only available from the point at which you turned on auditing.

Discover and adopt

End-users who have permissions to the app can find it in your organization's app store. Go to **Built for *Your Organization Name*** on the Apps page to find your organization's custom apps.



If you created and assigned an app setup policy, the app is pinned to the app bar in Teams for easy access for those users who were assigned the policy.

Update a custom app

To update an app, developers follow the steps in the [Create your app](#) and [Validate](#) sections.

You can update the app on the Manage apps page in the Microsoft Teams admin center. To update the app, in the left navigation of the Microsoft Teams admin center, go to **Teams apps** > **Manage apps**. select the app name, and then select **Update**. Updating the app replaces the existing app, and all app permission policies and app setup policies remain enforced for the updated app.

End-user update experience

In most cases, after you publish a new version of an app, it is automatically updated for end users who have added the app to their Teams client. For more information, see [end-user app update experience](#).

Remove a custom app from your organization's store

To remove an app, follow these steps:

1. Sign in to the Teams admin center.
2. Access **Teams apps** > **Manage apps** [↗](#) page.
3. Click on the name of the app to open the app details page.
4. Next to the app banner, select **Actions** > **Delete**.
5. On the dialog, select **Delete**.

Related articles

- [Publish a custom app submitted through the Teams App Submission API](#)
- [Manage your apps in the Microsoft Teams admin center](#)
- [Manage policies and settings for custom apps](#)
- [Manage app permission policies in Teams](#)
- [Manage app setup policies in Teams](#)

Connect a bot to Microsoft Teams

Article • 12/20/2022 • 2 minutes to read

APPLIES TO: SDK v4

You can configure your bot to communicate with people via Microsoft Teams. This article describes how to create a Teams app in Teams, connect your bot to your Teams app in Azure, and then test your bot in Teams.

Prerequisites

- An Azure subscription. If you don't already have one, create a [free account](#) before you begin.
- A bot published to Azure that you want to connect to Teams.
- A developer tenant in Teams with custom app uploading or sideloading enabled. For more information, see [Prepare your Microsoft 365 tenant](#).
- A valid Teams app package. For more information, see [Upload your app in Microsoft Teams](#).

Configure your bot in Azure

1. Open the [Azure portal](#).
2. Open the Azure Bot resource blade for your bot.
3. Open **Channels** and select **Microsoft Teams**:
 - a. Read and agree to the terms of service.
 - b. On the **Messaging** tab, select the cloud environment for your bot. For more information, see the [Post build](#) section of **Plan your app with Teams features**.
 - c. Select **Apply**.
4. Select **Get bot embed code**, locate the embed code for Teams, and then copy the *https* part of the code. For example, `https://teams.microsoft.com/l/chat/0/0?users=28:b8a22302e-9303-4e54-b348-343232`. You can use this code to test the bot in Teams.

Tip

- The **Calling** tab supports the Teams calling feature. For more information, see [Register calls and meetings bot for Microsoft Teams](#).
- The **Publish** tab contains information about how to publish your Teams app to the Teams Store.

Test your bot in Teams

Bots in production should be added to Teams as part of a Teams app. For more information, see [Test your app](#).

Important

Adding a bot by GUID, for anything other than testing purposes, isn't recommended. Doing so severely limits the functionality of a bot. Bots in production should be added to Teams as part of an app.

1. In your browser, open the URL you copied from your embed code, then choose the Microsoft Teams app (client or web) that you use to add the bot to Teams. You should be able to see the bot listed as a contact that you can send messages to and receive messages from in Microsoft Teams.
2. Interact with your bot in Teams.

Tip

Use one bot channel registration per environment, since your endpoint changes when you switch between local development, staging, and production environments.

Deleting the Teams channel registration will cause a new pair of keys to be generated when it's re-enabled. This invalidates all 29:xxx and a:xxx IDs that the bot may have stored for proactive messaging.

Publish your bot in Teams

For instructions on how to publish your app, see the Teams overview of how to [Distribute your Microsoft Teams app](#). It and the associated articles cover how to:

- Choose and configure install options for your bot
- Create your Teams app manifest, icon, and package
- Upload your app to Teams
- Publish your app to your org or to the Teams store

Additional information

- For more about Teams app development, see [Build apps for Microsoft Teams](#) and [Get started](#).
- For more about creating bots for Teams, see [Bots in Microsoft Teams](#).
- For more about publishing and testing a bot in Teams, see [Distribute your Microsoft Teams app](#) and [Test your app](#).
- To provide feedback and find additional resources, see [Microsoft Teams developer community channels](#).