

How do I create a type?

# OpenText<sup>™</sup> Thrust Studio Aviator

Basic Test Scenario

**This basic test scenario has been created for software version OpenText™ Thrust Studio Aviator 24.2.2 CE.**

It is also valid for subsequent software releases unless OpenText has made newer documentation available with the product, on an OpenText website, or by any other means.

Note that if you are using this test scenario with a later version of the OpenText™ Thrust Studio Aviator, the screenshots and usage might not always correspond.

**Open Text Corporation**

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111

Toll Free Canada/USA: 1-800-499-6544 | International: +800-4996-5440

Fax: +1-519-888-0677

Support: <https://support.opentext.com>

For more information, visit <http://www.opentext.com>

**Copyright © 2024 Open Text. All Rights Reserved.**

Trademarks owned by Open Text.

One or more patents may cover this product. For more information, please visit <https://www.opentext.com/patents>

**Disclaimer**

No Warranties and Limitation of Liability

Every effort has been made to ensure the accuracy of the features and techniques presented in this publication. However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

Last updated: 05/22/2024

# Contents

<b>Introduction .....</b>	<b>4</b>
<b>1 Setting up your environment .....</b>	<b>5</b>
1.1 Prerequisites .....	5
1.2 Download and install VS Code .....	6
1.3 Add the OpenText Cloud Developer Tools extension pack to VS Code .....	9
1.4 Install the OpenText™ Thrust Studio Aviator extension from VSIX .....	11
<b>2 Set up the organization profiles .....</b>	<b>13</b>
2.1 Add the organization profile for your developer organization .....	13
2.2 Add a tenant to the developer organization profile .....	18
2.3 OPTIONAL: Add the organization profile for the OpenText™ Thrust Studio Aviator .....	20
<b>3 Getting started with Thrust Studio Aviator .....</b>	<b>22</b>
3.1 Exploring the chat panel and selecting the Aviator organization .....	22
3.2 A developer conversation with Thrust Studio Aviator .....	27
3.3 Building an application with the assistance of Thrust Studio Aviator .....	46
<b>About OpenText .....</b>	<b>50</b>

# Introduction

This basic test scenario document is intended as an introduction to using the OpenText™ Thrust Studio Aviator digital assistant.

It will guide you through:

- Setting up the developer environment
- Configuring the organization (connection) profiles:
  - Organization profile to connect to your development organization
  - Organization profile to connect with the OpenText™ Thrust Studio Aviator
- Understanding how to ask questions and which responses to expect from the Generative AI enabled OpenText™ Thrust Studio Aviator
- Building an OpenText™ Thrust Services consuming application with the help of the OpenText™ Thrust Studio Aviator

# 1 Setting up your environment

The following software environment is used:

Software	Version
Operating System	Windows 11 Enterprise 22H2, 64-bit
VS Code	1.88.1
OpenText™ Cloud Developer Tools for VS Code	24.2.2 CE



## Note

The procedure steps, images, and the other references used in this testing guide are based on the above environment. If you are using any other environment or software versions, refer to the software documentation for that version.

## 1.1 Prerequisites

Make sure you can sign in to the [developer.opentext.com](https://developer.opentext.com) website and have an active trial or paid developer plan in the **United States** region.

To sign up for a trial, navigate to [developer.opentext.com/plans](https://developer.opentext.com/plans) to sign up for one.

**IMPORTANT:** Please, be certain to select the **United States** option when subscribing to the trial plan (see below screenshot).

Want to try OpenText Developer for free?

Our 90 day, **free** trial gives you first-hand experience developing applications that leverage the power of OpenText APIs.

United States ▼  
Trial will be created in United States.

My Developer Organization

-- Please select your location -- ▼

Purpose

0 / 150

For a detailed description on how to create a developer account and sign up for a developer trial, refer to: [developer.opentext.com/services/trial](https://developer.opentext.com/services/trial).

## 1.2 Download and install VS Code

1. Go to <https://code.visualstudio.com/download> and download the Microsoft VS Code distribution that matches your system.



### Note

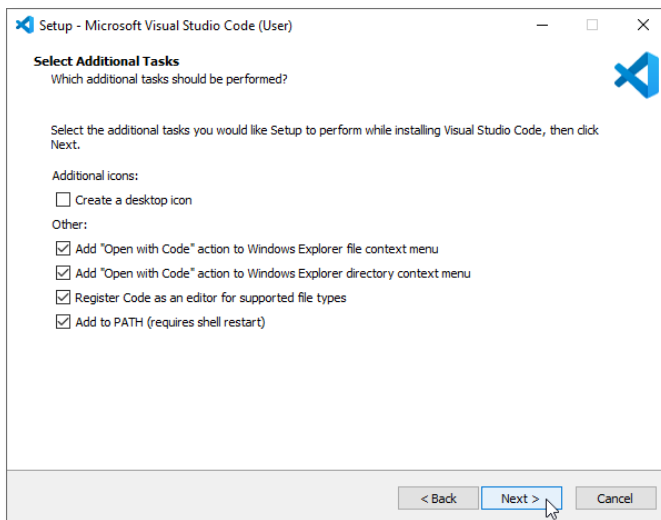
To install VS Code on a 64-bit Windows 11 system, choose to download the x64 User Installer for Windows and install the latest version of VS Code.



### Important

OpenText Cloud Developer Tools for VS Code is tested with Windows and Mac Operating Systems. You can use Linux Operating System but if you run into problems, OpenText might not be able to provide a solution.

2. Save and run the installer.
3. Select **I accept the agreement** and click **Next** to continue.
4. Select the installation destination location and click **Next**. You can use the suggested default location.
5. Keep the default setting for the selecting of the start menu folder and click **Next**.
6. Select all the additional tasks under **Other** and optionally select **Create a desktop icon**. Click **Next** to continue.

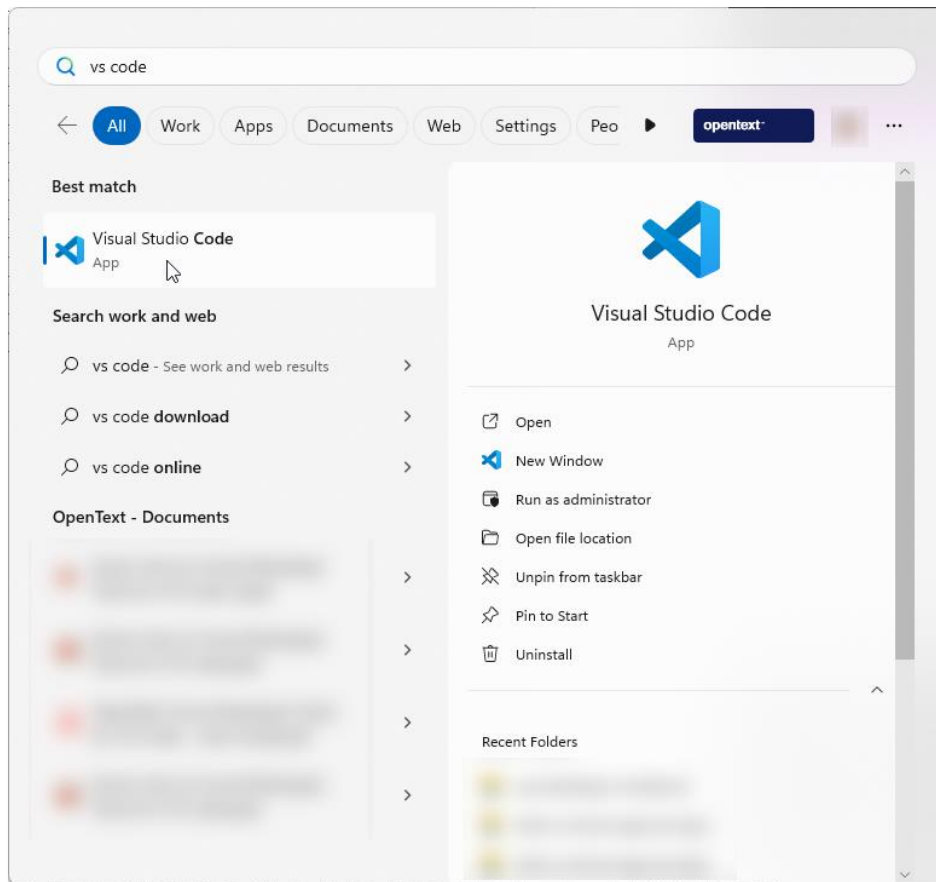


7. Verify your choices and click **Install** to start the VS Code installation.
8. After the installation is complete, you can click **Finish** to close the VS Code Setup Wizard.
9. After VS Code is installed, in the Windows **Start** menu, type `vs code` in the search box and select the **Visual Studio Code** application.



### Tip

You can Pin to taskbar for easy access from the task bar.



10. The **Get Started with VS Code** wizard is displayed when VS Code is launched for the first time.
11. Select a theme and click **Mark Done** to confirm your choice.

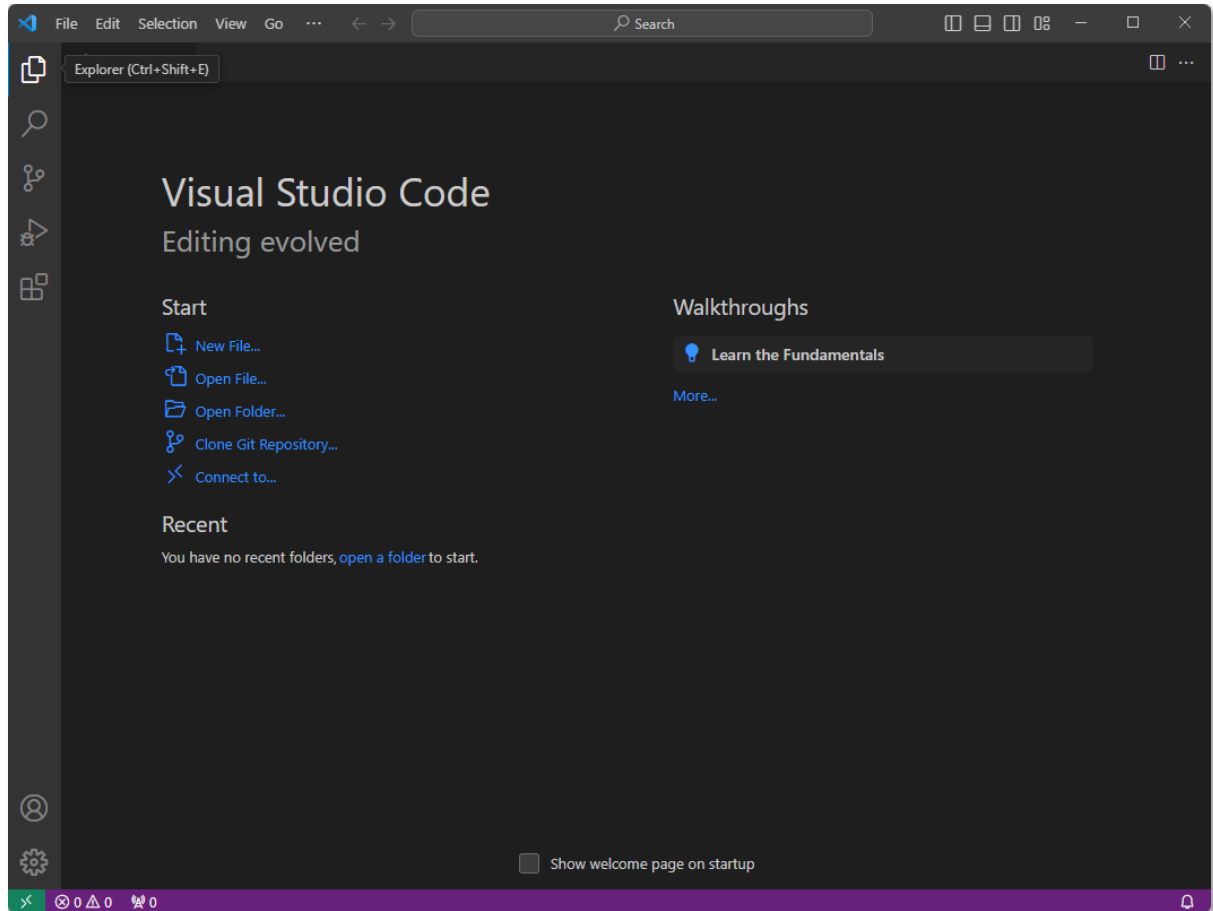


**Note**

For this testing guide the **Dark Modern** theme is selected. Take this into account when comparing your VS Code appearance with the VS Code screen shots throughout this document.

12. The standard VS Code welcome page is now displayed.

13. Make sure **Show welcome page on startup** is not selected so that the page does not appear when VS Code is launched every time.



14. Close the welcome page.

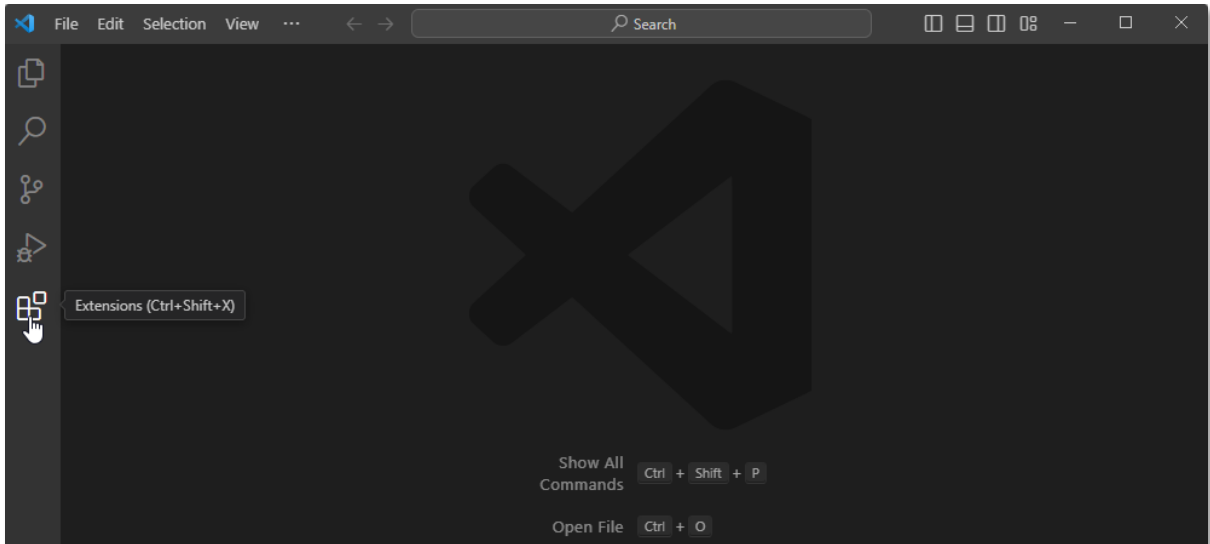
Next step:

Add the OpenText Cloud Developer Tools VS Code extension pack to your VS Code IDE.



## 1.3 Add the OpenText Cloud Developer Tools extension pack to VS Code

1. On the VS Code Activity Bar, click **Extensions**.

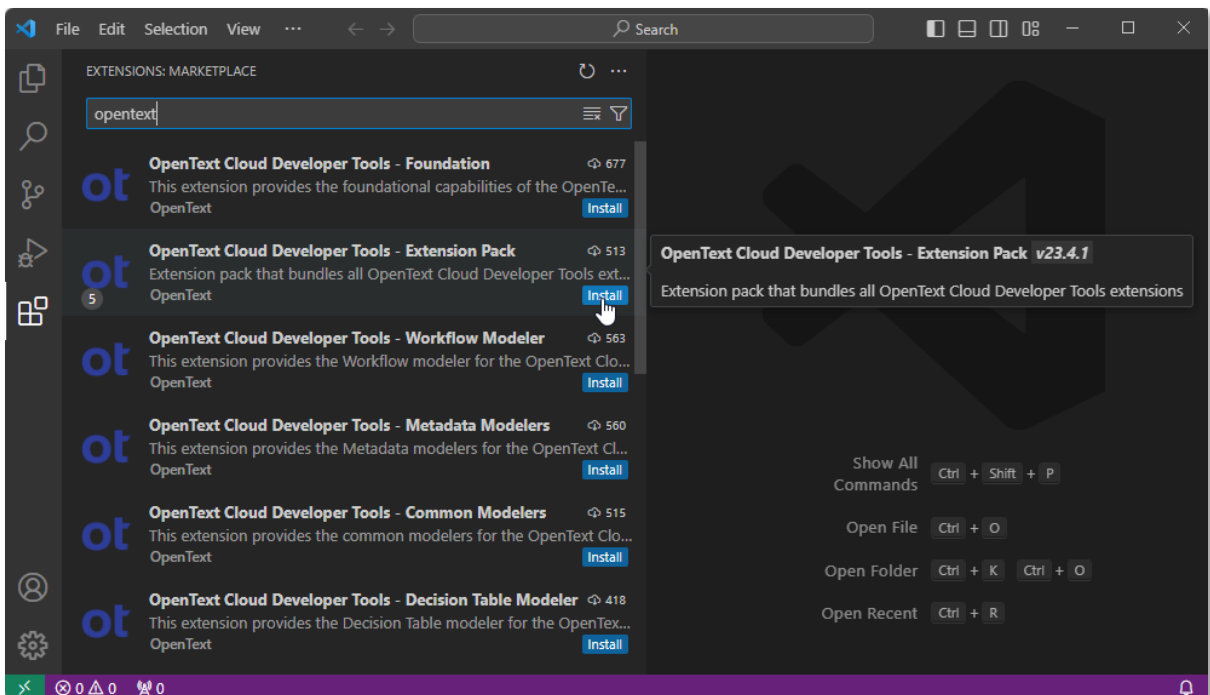


2. On the **Search** Extensions in Marketplace search bar, type `opentext` and choose to install the **OpenText Cloud Developer Tools - Extension Pack**.

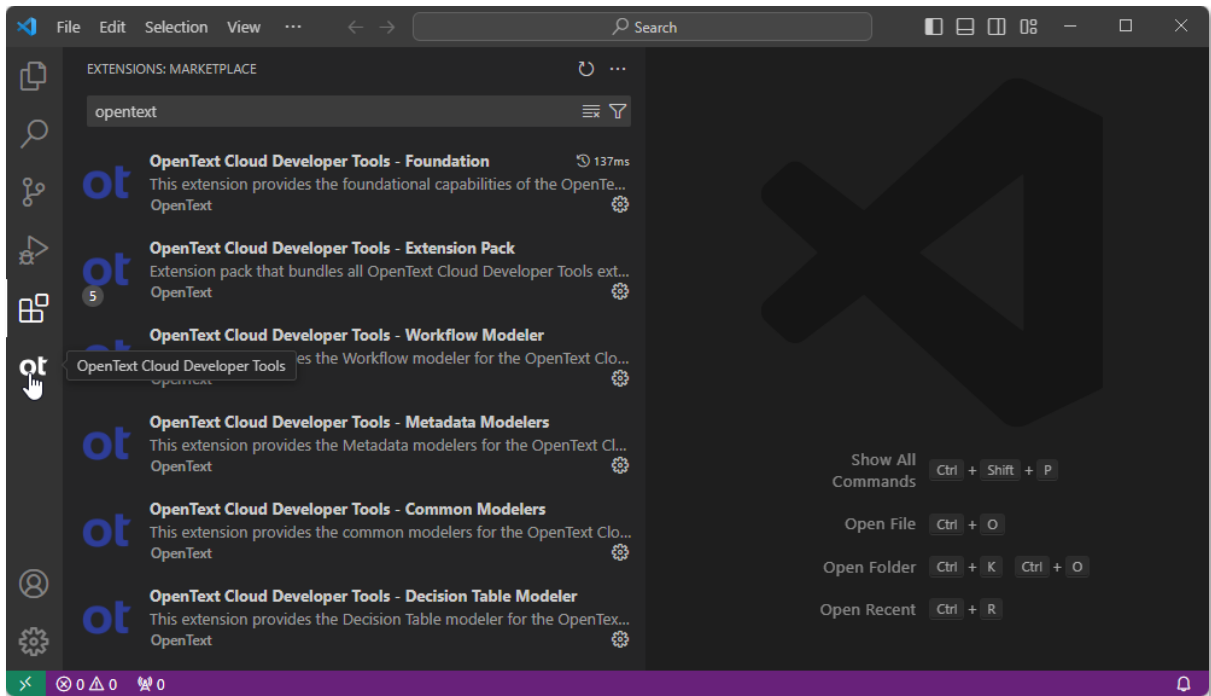


### Note

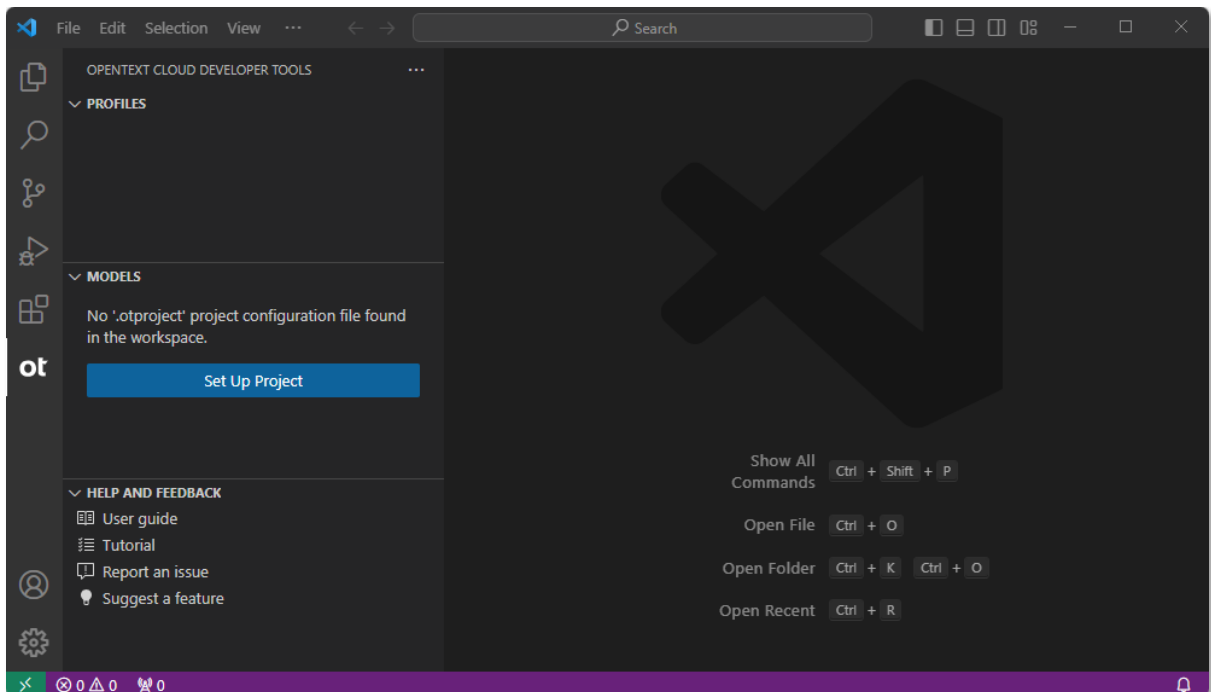
If prompted, click Reload to ensure that the installed VS Code extension pack is enabled.



- On the **Activity Bar**, click the **OpenText Cloud Developer Tools** button .



- This opens the **OpenText Cloud Developer Tools** view.

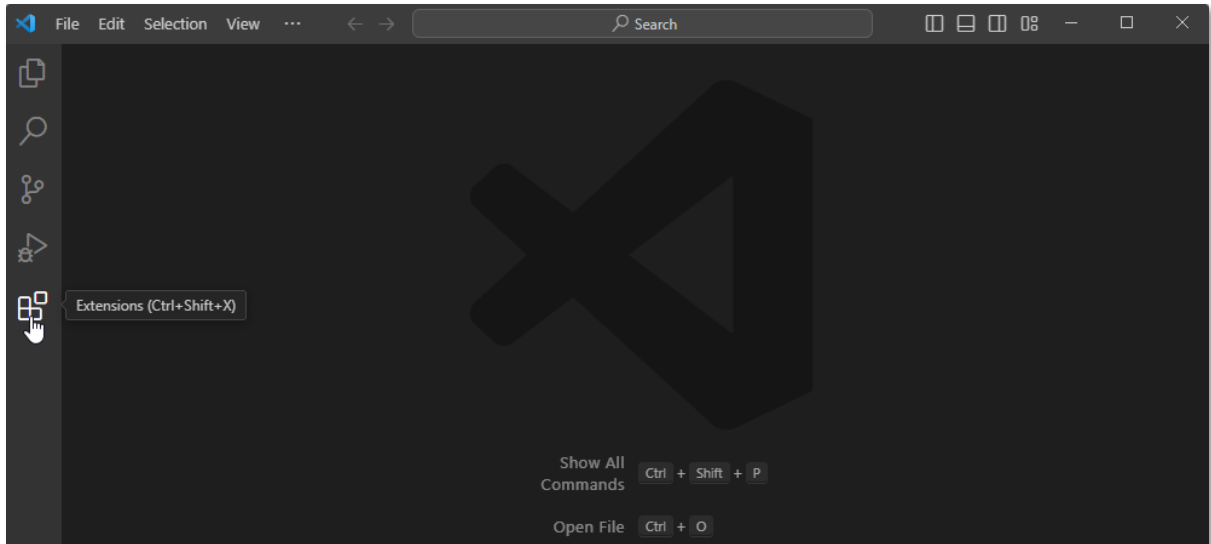



Next step:

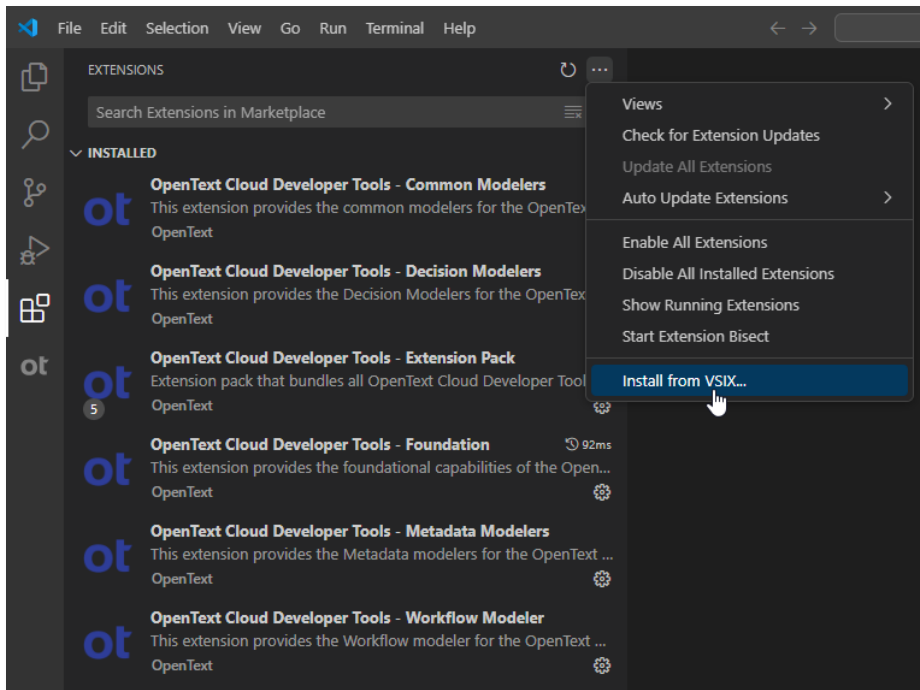
Install the OpenText™ Thrust Studio Aviator extension from VSIX.

## 1.4 Install the OpenText™ Thrust Studio Aviator extension from VSIX

1. Download the **ocp-vscode-aviator.vsix** file from this [link](#).
2. In VS Code, from the Activity Bar, click **Extensions**.

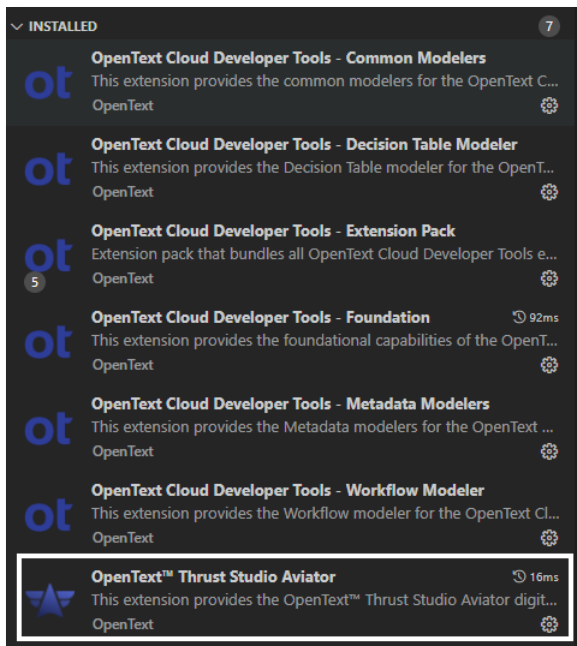


3. Click the **View and More Actions...** button  and select **Install from VSIX...**



4. Select the previously downloaded **ocp-vscode-aviator.vsix** and click **Install**.

5. The **OpenText™ Thrust Studio Aviator** extension is now installed.



Next chapter:


Set up the organization profiles.

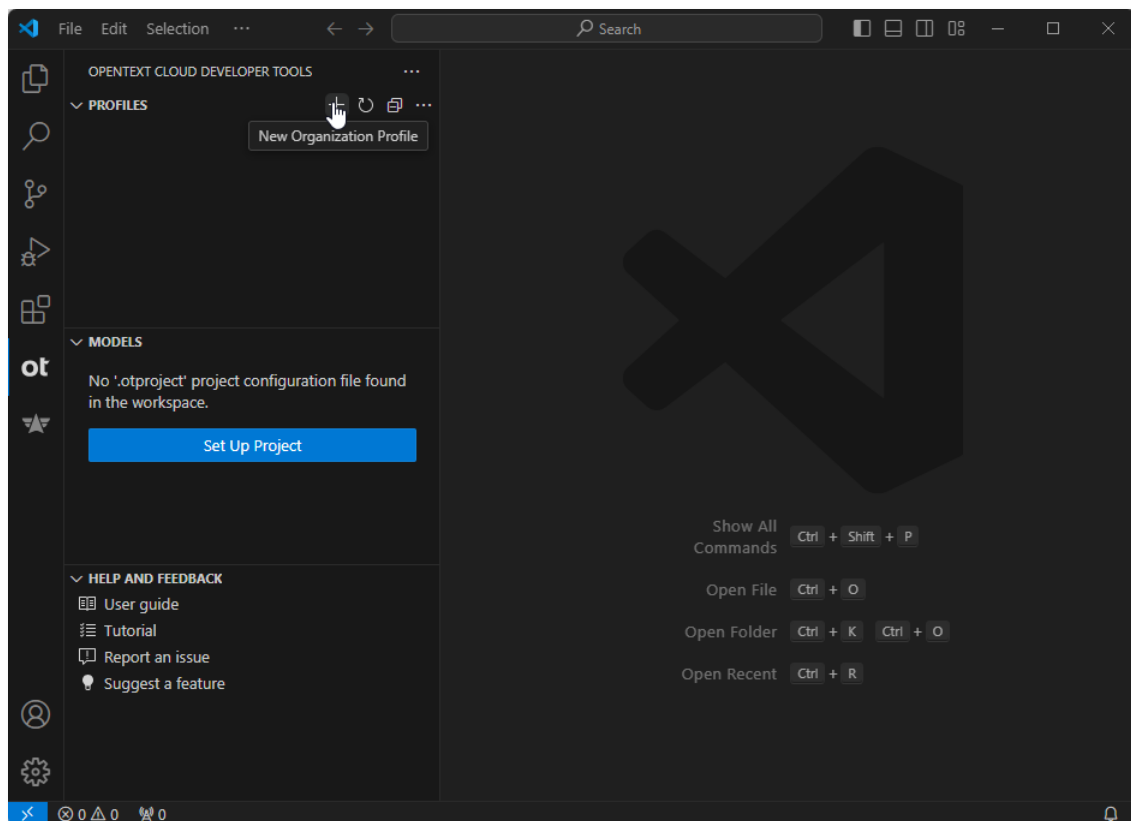
## 2 Set up the organization profiles

### 2.1 Add the organization profile for your developer organization

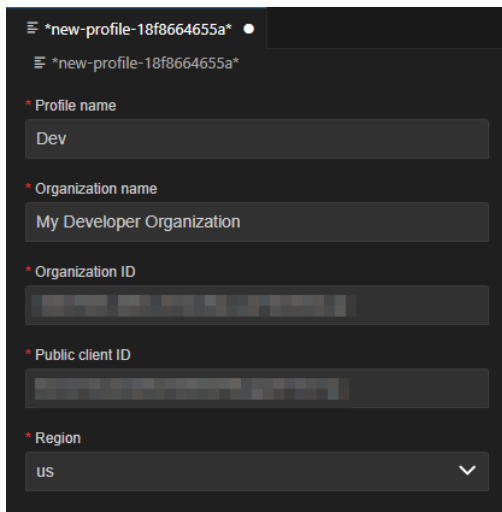
**IMPORTANT:** If you want to use a different developer organization than the one in the United States (US) region as per the [Prerequisites](#) (e.g., you have a developer organization on the NA Build & Test region or in Europe), you can choose to use that organization in this step, and then do the [OPTIONAL: Add the organization profile for the OpenText™ Thrust Studio Aviator](#) configuration step to add the US-based organization for the Thrust Studio Aviator digital assistant.

In case you only use a single (US) developer organization, there is no need to do the optional configuration step.

1. Open VS Code and on the Activity Bar, select the **OpenText Cloud Developer Tools** view.
2. In the **PROFILES** section, click the **New Organization Profile** button  to create a new organization profile.



- In the organization profile screen, fill the property fields for the developer organization:



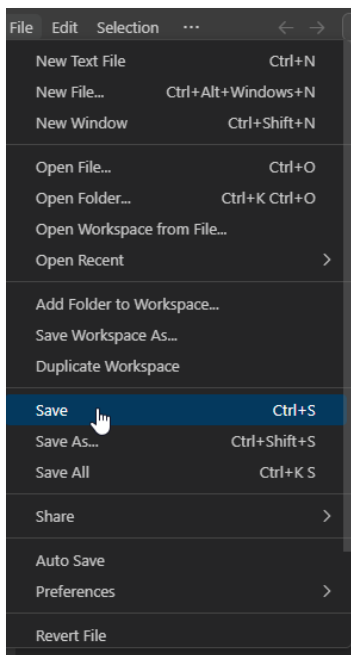
The following table describes the organization profile property fields:

Property	Description
Profile name	The profile name is used as a display name in the PROFILES section.
Organization name	The name of the organization. It is recommended to use the value from the organization overview page in Admin Center.  For more information on accessing the organization overview page in Admin Center, see <a href="#">Admin Center User Interface</a> .
Organization ID	The unique identifier for the organization.  For more information about how to view your organization ID, see <a href="#">View your organization ID using Admin Center</a> .
Public client ID	The unique identifier of the public OAuth service client for the organization.  For more information about how to view your public client ID, see <a href="#">View and manage OAuth service clients</a> .
Region	The region where the organization is made available. By default, the value is na-1-dev but except if you are using a different developer organization you should select <b>us</b> .  If you want to confirm the region of your developer organization, you can do this by opening Admin Center for the organization and using the URL (region is part of the domain name).

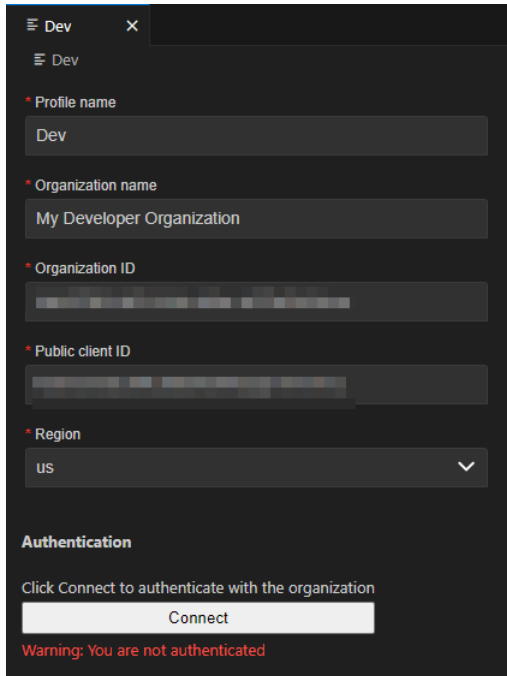
4. Select **File > Save** to save the organization profile.

**Note**

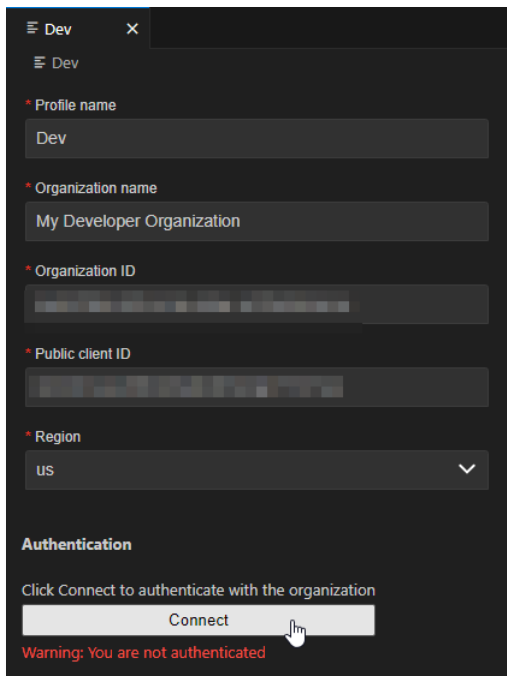
OpenText Cloud Developer Tools configuration artifacts such as setting up organization connection, project set up, and different model configurations use the standard VS Code file saving functionality. To save your changes to any configuration artifact, press **Ctrl+S** (for Windows systems) or select **File > Save**.



5. After saving, the organization profile is listed as the default profile in the **PROFILES** section and the tab above the organization profile is renamed to the profile name. On the organization profile screen an **Authentication** section with the **Connect** button appears.



6. Click **Connect** to test the newly configured connection to your developer organization.



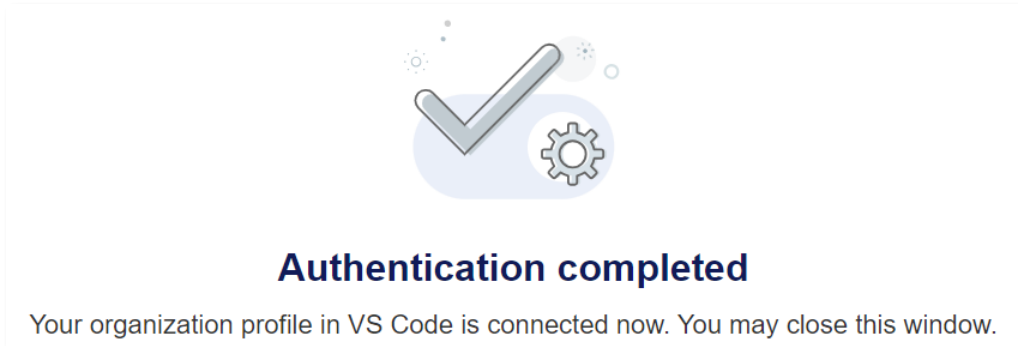


7. On the sign in page, authenticate with your developer.opentext.com username and password.

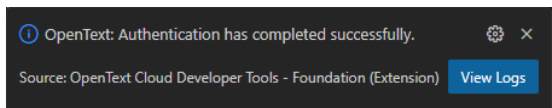


**Note**

If you are already signed in, the browser will immediately show the **Authentication completed** page.



8. After successful authentication, a confirmation message is displayed in VS Code.



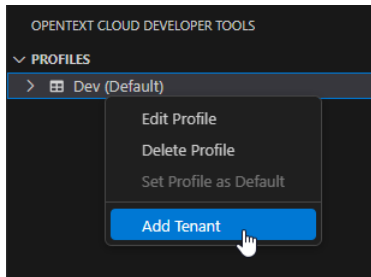
9. Close the organization profile screen.

Next step:

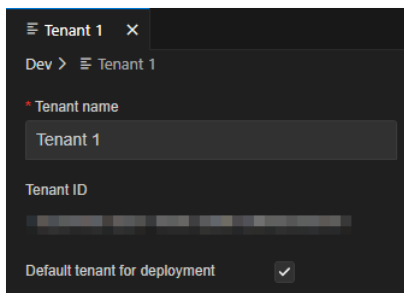
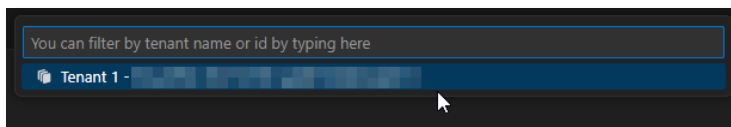
Add a tenant to the developer organization profile.

## 2.2 Add a tenant to the developer organization profile

1. In the **PROFILES** section, right-click the profile and select **Add Tenant**.



2. From the drop-down in the top center of VS Code, select the developer tenant you want to use (typically "Tenant 1" if you are on a trial plan).



The following table describes the tenant property fields:

Property	Description
Tenant name	The name of the tenant. It is recommended to use the value from the tenant page in Admin Center. For more information on accessing the tenant page in Admin Center, see <a href="#">Retrieve your tenant ID or reset tenant password using Admin Center</a> .
Tenant ID	The unique identifier for the tenant. For more information about how to view your tenant ID, see <a href="#">Retrieve your tenant ID or reset tenant password using Admin Center</a> .
Default tenant for deployment	Whether to use this tenant by default when deploying to the corresponding organization.



### Note

The first tenant added to the organization profile is made the default tenant for deployment.

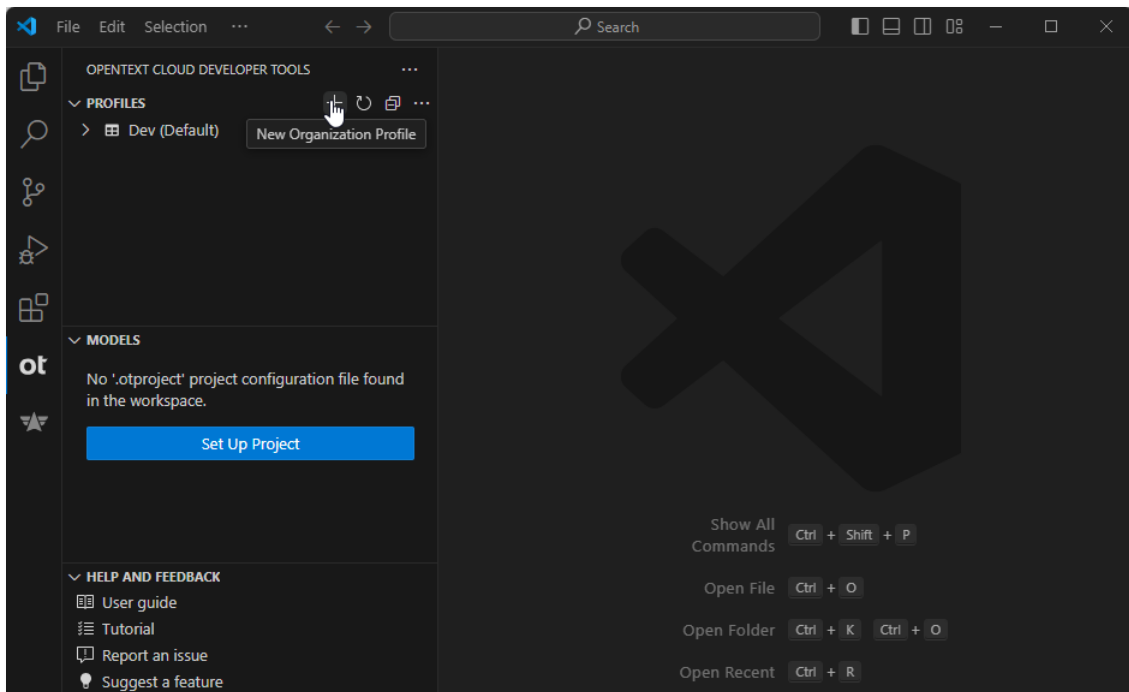
Next step:

OPTIONAL: Add the organization profile for the OpenText™ Thrust Studio Aviator.

## 2.3 OPTIONAL: Add the organization profile for the OpenText™ Thrust Studio Aviator

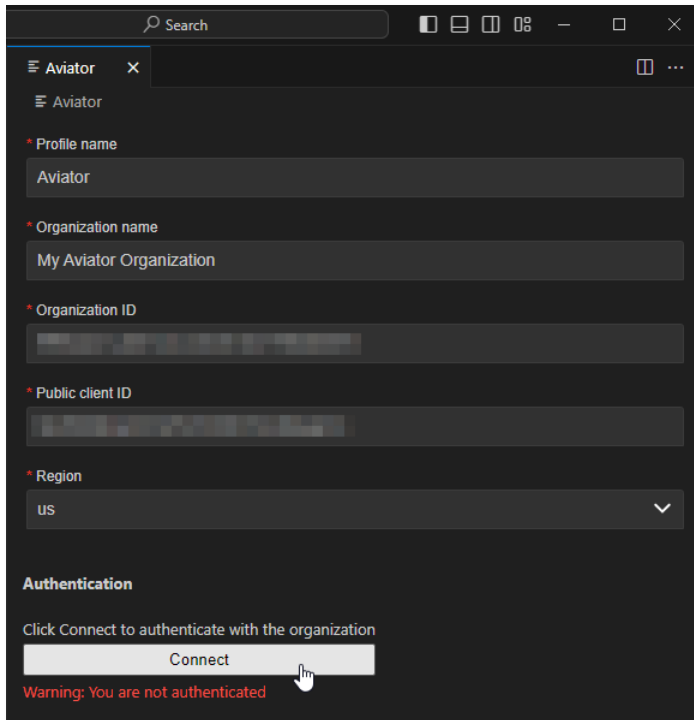
This optional step is only required if you want to work with a different developer organization than the **United States** based organization you subscribed to (see [Prerequisites](#)), and you have set this non-US organization as the developer organization when performing the [Add the organization profile for your developer organization](#) step.

1. From the **OpenText Cloud Developer Tools** view, add another organization profile.

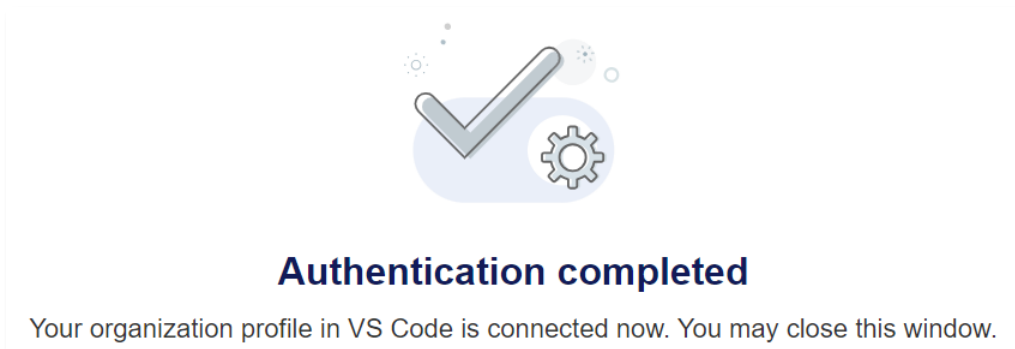


2. In the organization profile screen, fill the property fields for the US-based organization (in the same way as described in [Add the organization profile for your developer organization](#)) giving it a name indicating its use for the OpenText™ Thrust Studio Aviator (e.g., naming the profile “Aviator”):

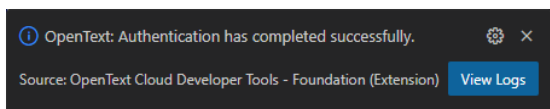
3. Select **File > Save** to save the organization profile and click **Connect** to test the newly configured connection to your Aviator organization.



4. On the sign in page, authenticate with your developer.opentext.com username and password.



5. After successful authentication, a confirmation message is displayed in VS Code.



6. Close the organization profile screen.

Next chapter:

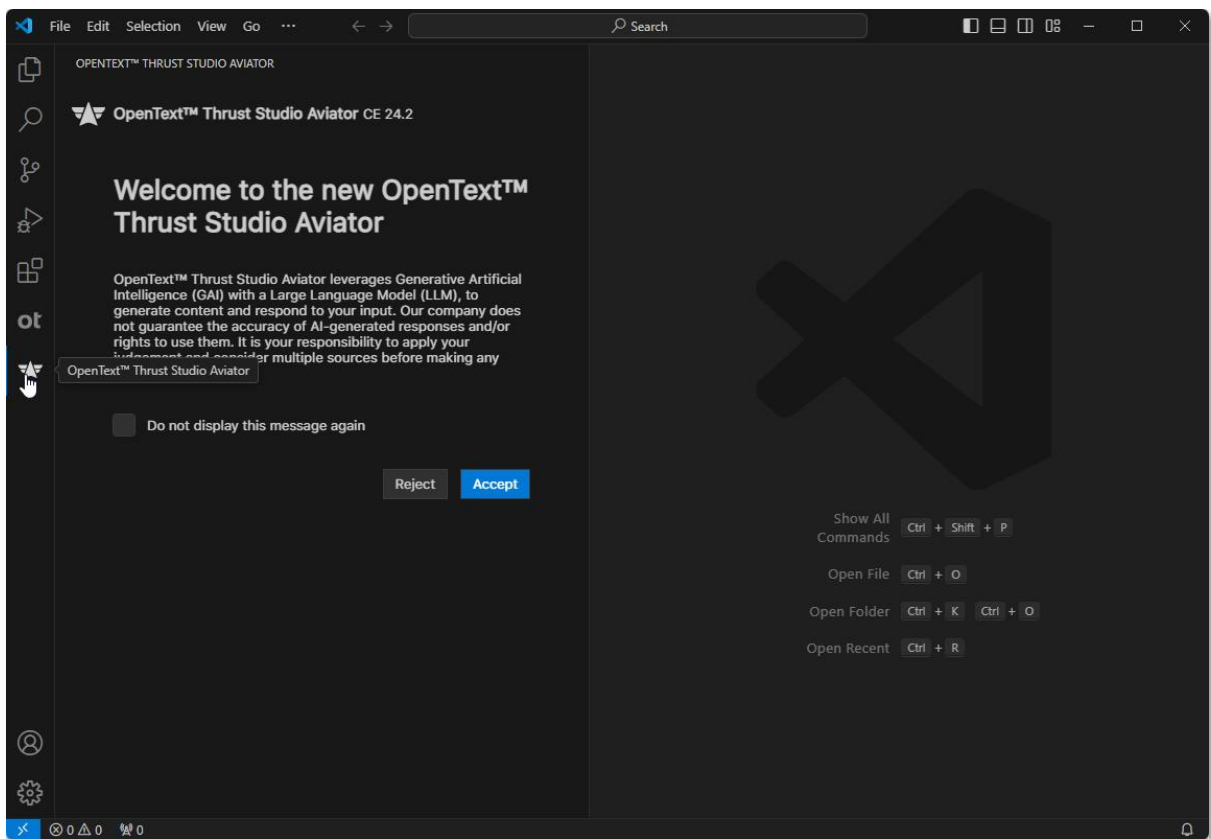
Getting started with Thrust Studio Aviator.

## 3 Getting started with Thrust Studio Aviator

This chapter will introduce you to using the OpenText™ Thrust Studio Aviator, by exploring the chat panel and asking a set of questions you would typically ask in context of working with the OpenText™ Thrust Services.

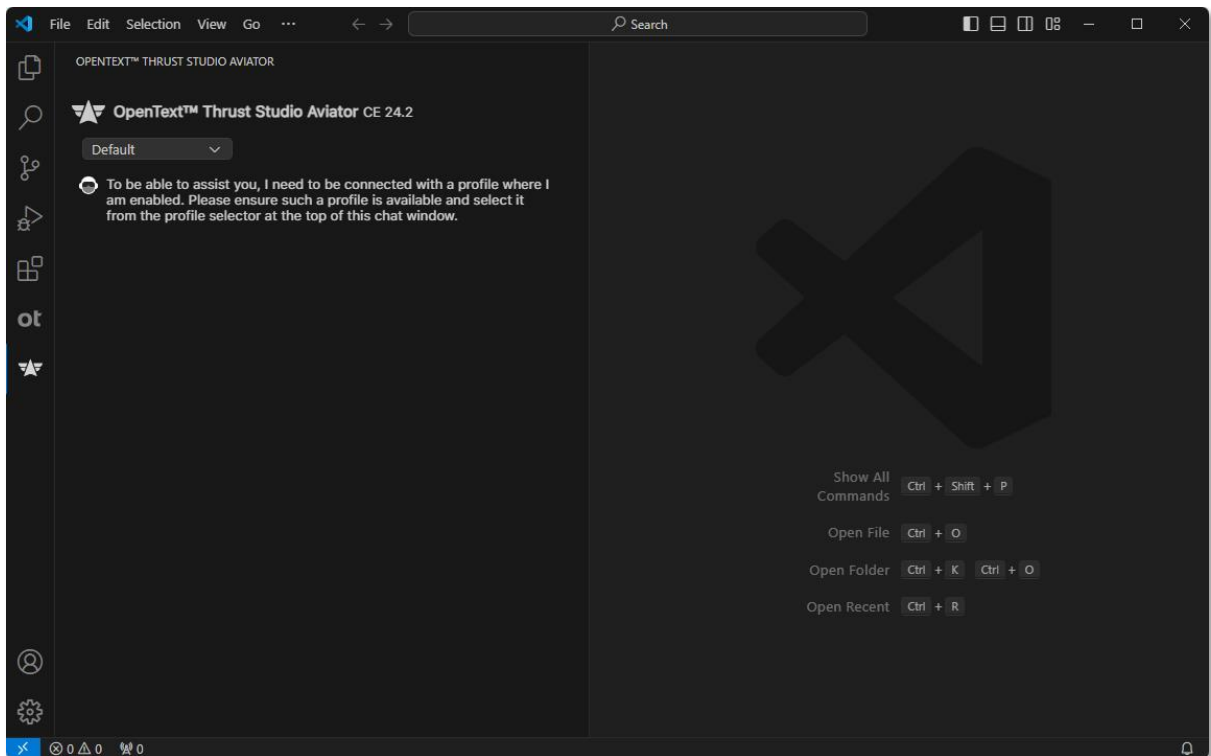
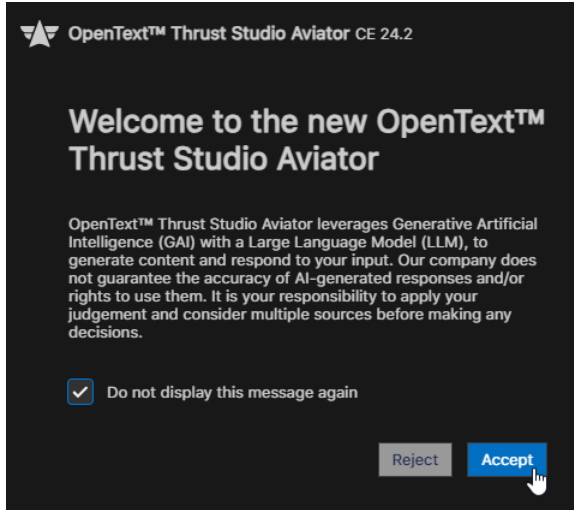
### 3.1 Exploring the chat panel and selecting the Aviator organization

1. Open VS Code and on the Activity Bar, select the **OpenText™ Thrust Studio Aviator** view.



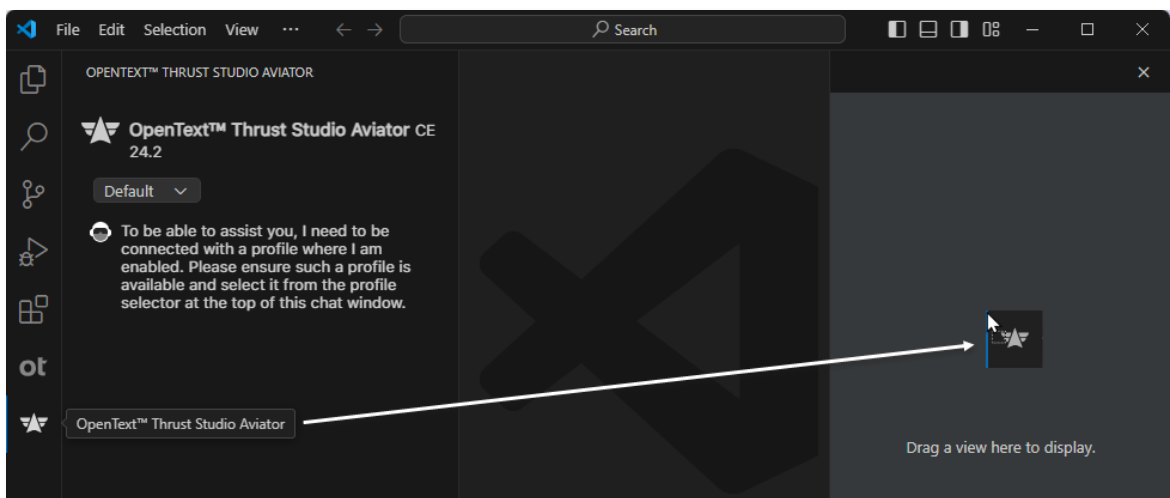
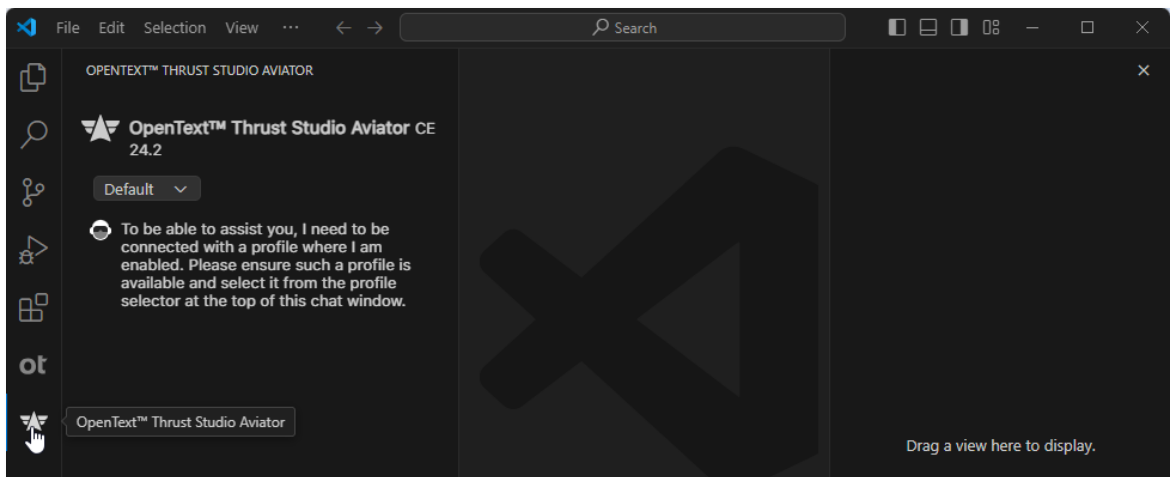
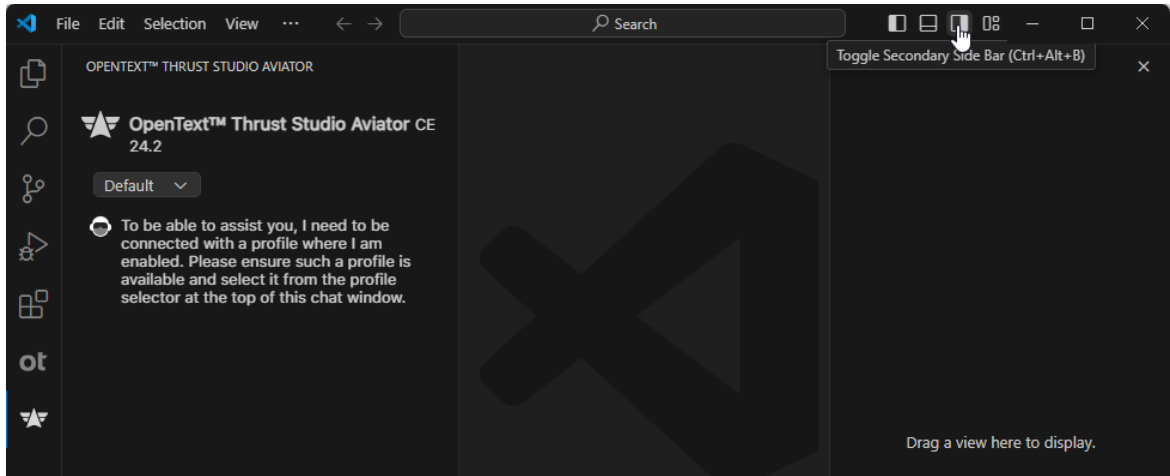
2. Read the disclaimer and click **Accept** if you accept it.

You can additionally choose to select **Do not display this message again** so that the disclaimer no longer displays when you open VS Code and use the OpenText™ Thrust Studio Aviator.

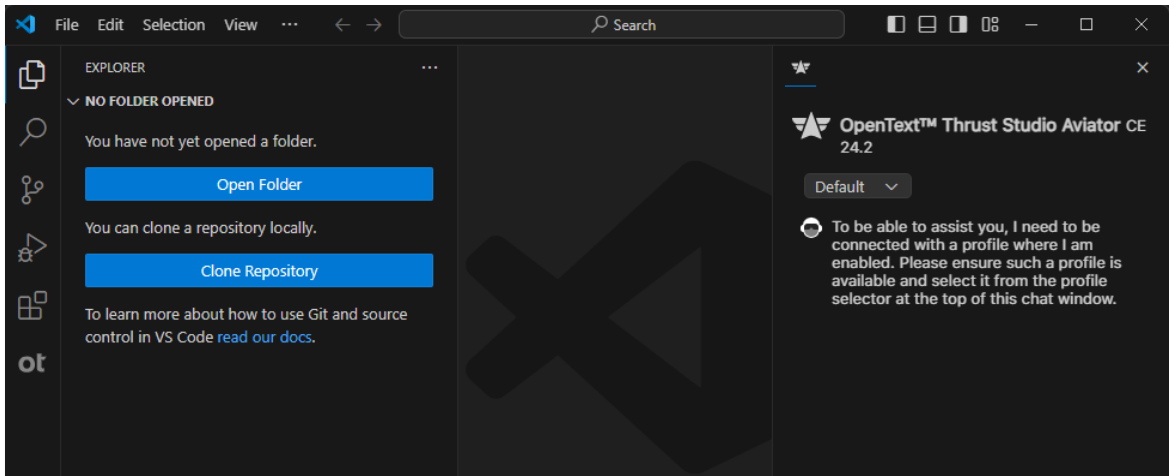


Note that the above screen shot is an example with a separate developer organization that is not US-based. In case your default (developer) organization is the US-based one (i.e., the one where OpenText™ Thrust Studio Aviator is enabled), you will see a different message stating that the Aviator is ready to take your questions.

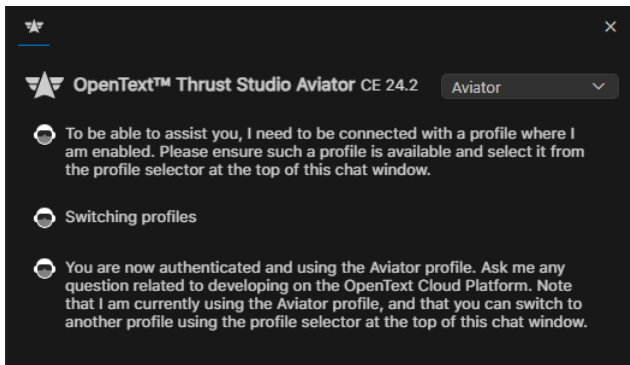
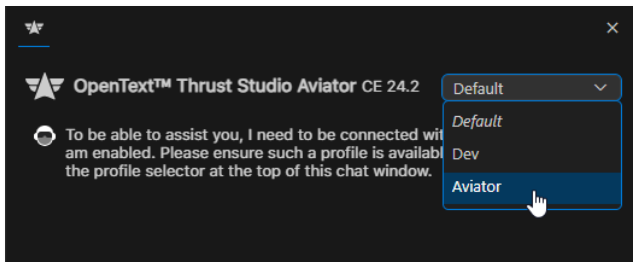
3. First, for convenience, select to expand the right pane and drag and drop the Thrust Studio Aviator view to that pane (so that you can use the left pane for other activities).



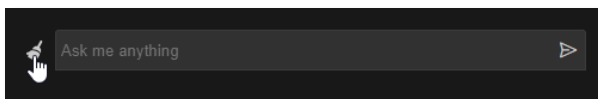


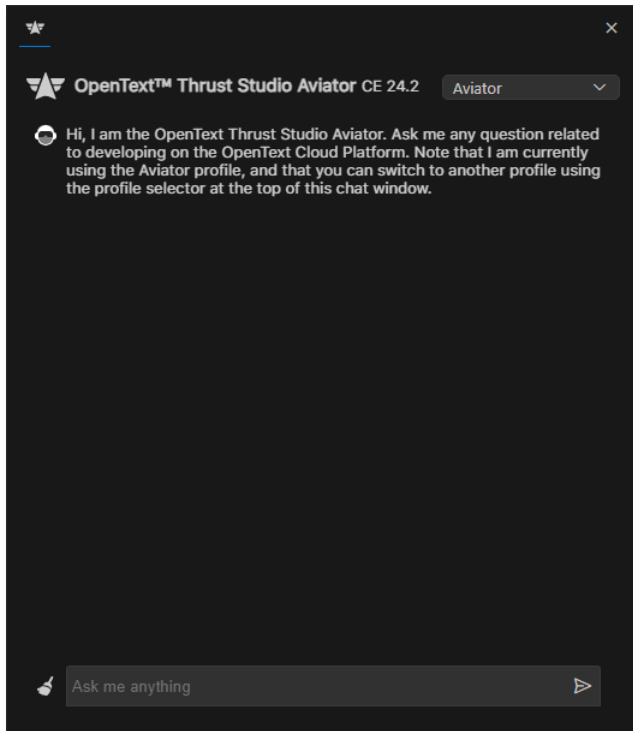


4. In case the default organization is not the US-based one, from the Thrust Studio Aviator organization profile dropdown, select the **Aviator** organization.



5. As a last step, you can clear/reset the chat panel with the brush (🗑️) button next to the chat input box to have a nice clean chat window.





6. You are now ready to start asking questions to the OpenText™ Thrust Studio Aviator.

Next step:

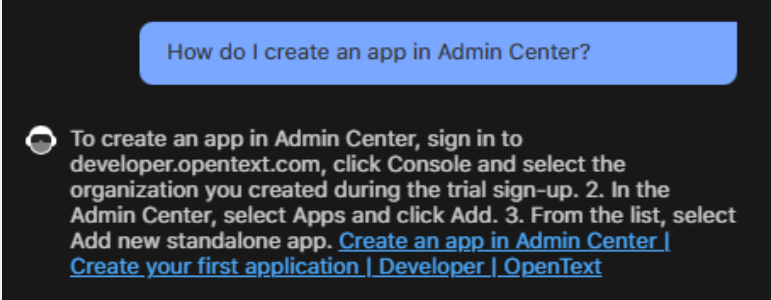
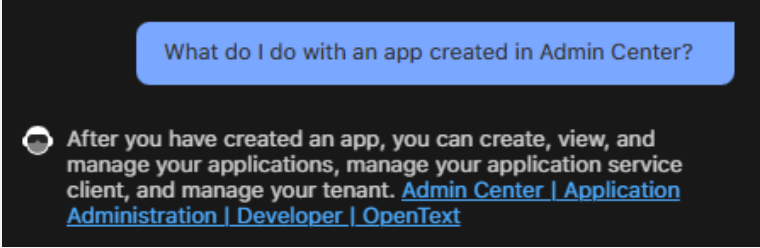
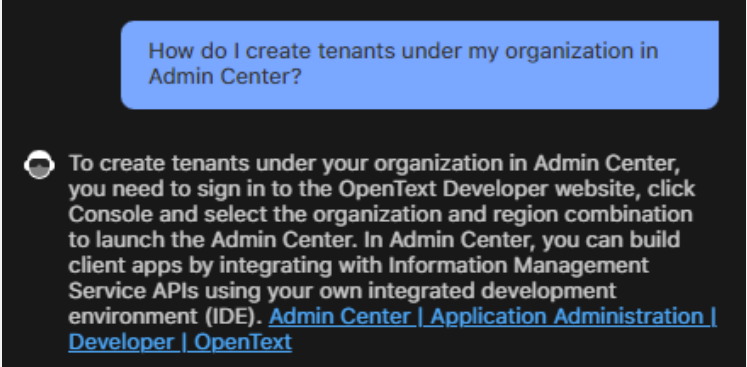
A developer conversation with Thrust Studio Aviator.

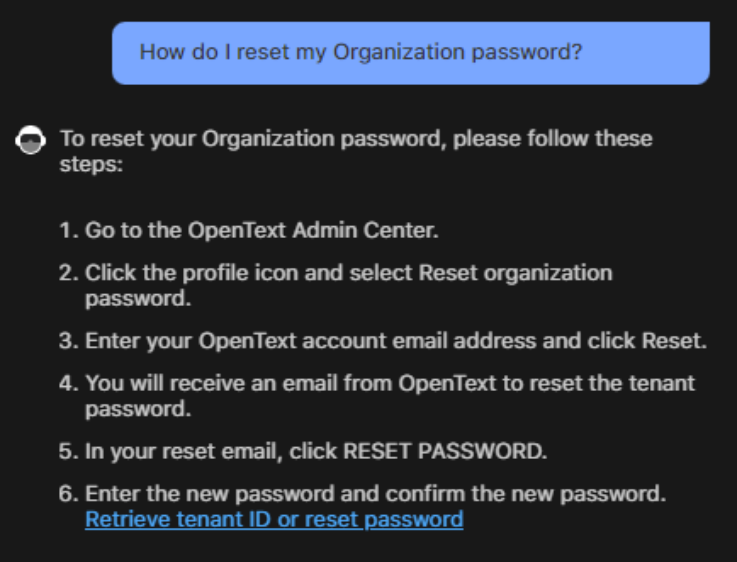
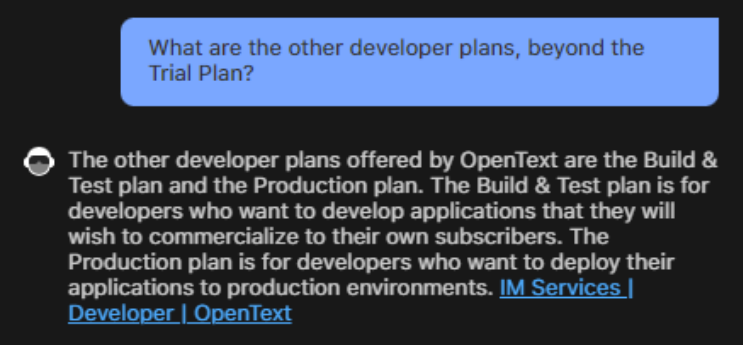
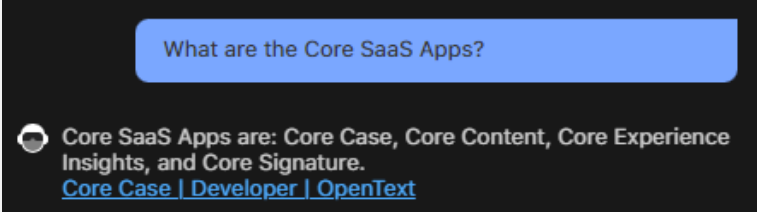
## 3.2 A developer conversation with Thrust Studio Aviator

Please ask the following questions to the Thrust Studio Aviator to see how it responds. We have also provided the responses (as “Our test answer”) we have observed on our test run for your convenience.

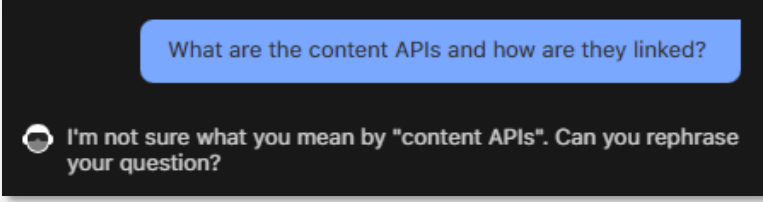
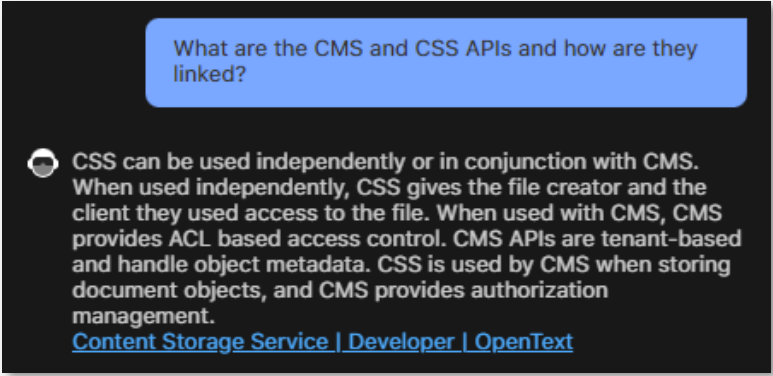
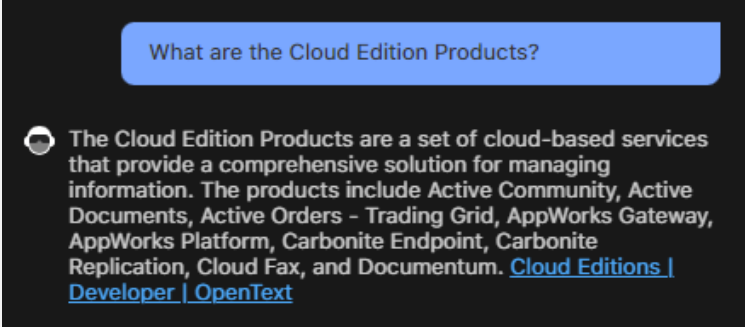
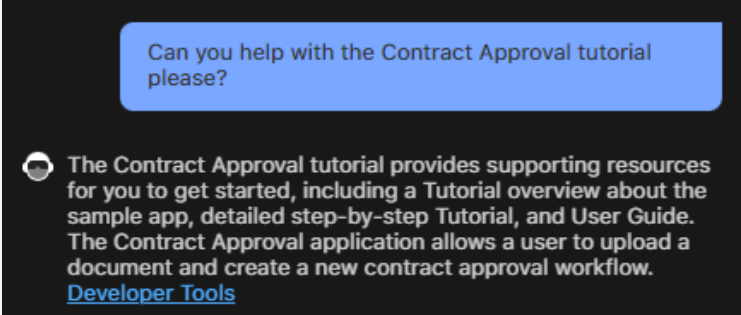
Question	What is OpenText Developer Tools?
Our test answer	 <p>The screenshot shows a chat window with a dark background. At the top, a blue bubble contains the question: "What is OpenText Developer Tools?". Below it, a white speech bubble with a circular icon contains the answer: "OpenText Developer Tools is a website that provides developers with access to Swagger documentation, Postman collections, tutorials, and samples. It also provides access to OpenText Information Management Services APIs based on the entitlement. The website also provides support tools in the form of Visual Studio (VS) Code extension packs. <a href="#">Getting started   Developer   OpenText</a>".</p>
Question	What are the OpenText Cloud Developer Tools for VS Code?
Our test answer	 <p>The screenshot shows a chat window with a dark background. At the top, a blue bubble contains the question: "What are the OpenText Cloud Developer Tools for VS Code?". Below it, a white speech bubble with a circular icon contains the answer: "The OpenText Cloud Developer Tools for VS Code are a set of low-code modelers that work on top of the OpenText Cloud Platform APIs. They also include an Application Lifecycle Management (ALM) service providing deployment capabilities and a command line utility to help you automate the deployment actions in your CI/CD pipeline. <a href="#">Developer Tools</a>".</p>
Question	Can you help me with a list of APIs provided by OpenText that I can use?
Our test answer	 <p>The screenshot shows a chat window with a dark background. At the top, a blue bubble contains the question: "Can you help me with a list of APIs provided by OpenText that I can use?". Below it, a white speech bubble with a circular icon contains the answer: "The following APIs are provided by OpenText: Active Documents Visibility REST Services, Business Workspaces 21.4 REST API, Business Workspaces 22.2 REST API, Business Workspaces 22.3 REST API, Capture Service, Content Metadata Service, Content Storage Service, Decision, and Documentum. Is there anything else I can help you with? <a href="#">APIs   Developer   OpenText</a>".</p>

Question	How do I start working with OpenText APIs?
Our test answer	
Question	How do I Sign up for a trial account to set up an Organization?
Our test answer	
Question	Can you help me with a list of APIs under IM Services?
Our test answer	
Question	Provide me a list of all IM Services.
Our test answer	

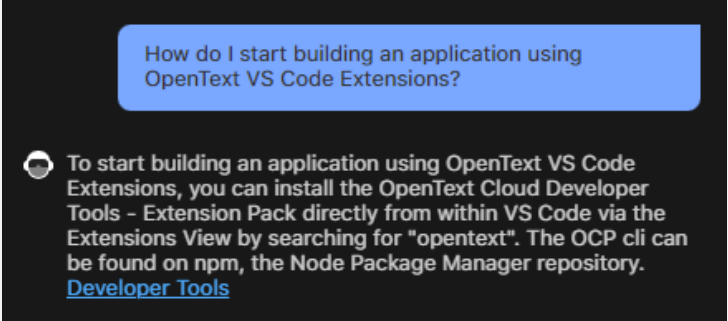
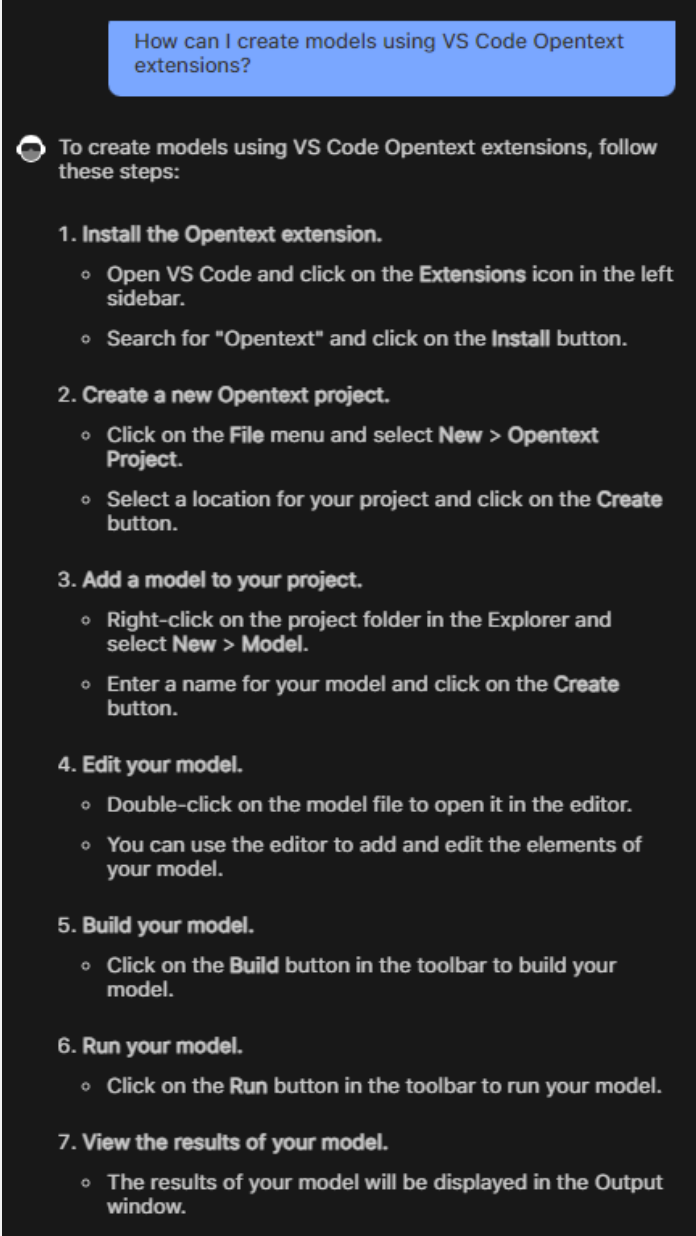
Question	How do I create an app in Admin Center?
Our test answer	 <p>How do I create an app in Admin Center?</p> <ul style="list-style-type: none"> <li>To create an app in Admin Center, sign in to <a href="#">developer.opentext.com</a>, click Console and select the organization you created during the trial sign-up. 2. In the Admin Center, select Apps and click Add. 3. From the list, select Add new standalone app. <a href="#">Create an app in Admin Center   Create your first application   Developer   OpenText</a></li> </ul>
Question	What do I do with an app created in Admin Center?
Our test answer	 <p>What do I do with an app created in Admin Center?</p> <ul style="list-style-type: none"> <li>After you have created an app, you can create, view, and manage your applications, manage your application service client, and manage your tenant. <a href="#">Admin Center   Application Administration   Developer   OpenText</a></li> </ul>
Question	How do I create tenants under my organization in Admin Center?
Our test answer	 <p>How do I create tenants under my organization in Admin Center?</p> <ul style="list-style-type: none"> <li>To create tenants under your organization in Admin Center, you need to sign in to the OpenText Developer website, click Console and select the organization and region combination to launch the Admin Center. In Admin Center, you can build client apps by integrating with Information Management Service APIs using your own integrated development environment (IDE). <a href="#">Admin Center   Application Administration   Developer   OpenText</a></li> </ul>

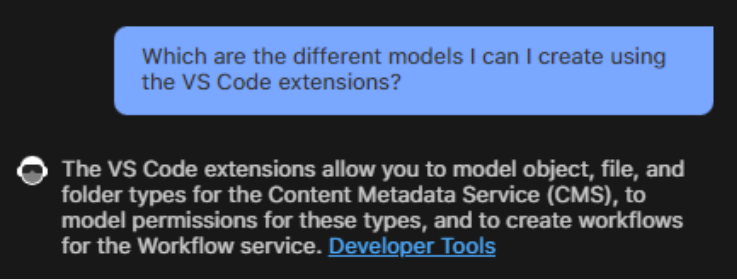
Question	How do I reset my Organization password?
Our test answer	 <p>How do I reset my Organization password?</p> <p>To reset your Organization password, please follow these steps:</p> <ol style="list-style-type: none"> <li>1. Go to the OpenText Admin Center.</li> <li>2. Click the profile icon and select Reset organization password.</li> <li>3. Enter your OpenText account email address and click Reset.</li> <li>4. You will receive an email from OpenText to reset the tenant password.</li> <li>5. In your reset email, click RESET PASSWORD.</li> <li>6. Enter the new password and confirm the new password. <a href="#">Retrieve tenant ID or reset password</a></li> </ol>
Question	What are the other developer plans, beyond the Trial Plan?
Our test answer	 <p>What are the other developer plans, beyond the Trial Plan?</p> <p>The other developer plans offered by OpenText are the Build &amp; Test plan and the Production plan. The Build &amp; Test plan is for developers who want to develop applications that they will wish to commercialize to their own subscribers. The Production plan is for developers who want to deploy their applications to production environments. <a href="#">IM Services   Developer   OpenText</a></p>
Question	What are the Core SaaS Apps?
Our test answer	 <p>What are the Core SaaS Apps?</p> <p>Core SaaS Apps are: Core Case, Core Content, Core Experience Insights, and Core Signature. <a href="#">Core Case   Developer   OpenText</a></p>

Question	What are Core SaaS Apps?
Our test answer	<p>What are Core SaaS Apps?</p> <ul style="list-style-type: none"> <li>Core SaaS Apps are a family of products that offer robust content management capabilities and integration to lead business applications including SAP S/4HANA Public Cloud. These apps allow customers to extend and integrate their organization's tenant, expand the solution with no-code templates, and have the flexibility to build new extensions and automation. Core SaaS Apps include Core Case, Core Content, Core Experience Insights, and Core Signature. <a href="#">SaaS Apps</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></li> </ul>
Question	What is Core Signature useful for?
Our test answer	<p>What is Core Signature useful for?</p> <ul style="list-style-type: none"> <li>Core Signature is a developer API that allows you to create signing flows that will allow your app users to send and sign documents. <a href="#">Core SaaS</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></li> </ul>
Question	What is Content Metadata Service?
Our test answer	<p>What is Content Metadata Service?</p> <ul style="list-style-type: none"> <li>Content Metadata Service provides all the capabilities and integrations expected from a content management platform. Integrations with the Retention Service for records provide an even deeper functionality set to further your goals in creating a content-oriented application. <a href="#">Content Metadata Service</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></li> </ul>
Question	What is Content Storage Service?
Our test answer	<p>What is Content Storage Service?</p> <ul style="list-style-type: none"> <li>Content Storage Service (CSS) is an object store responsible for all functionality regarding object data and physical content. It provides storage and indexing for content and documents, including encryption/decryption of the content, anti-virus and authorization checks on upload and download. CSS can be used standalone or in conjunction with the Content Metadata Service (CMS). <a href="#">Content Storage Service</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></li> </ul>

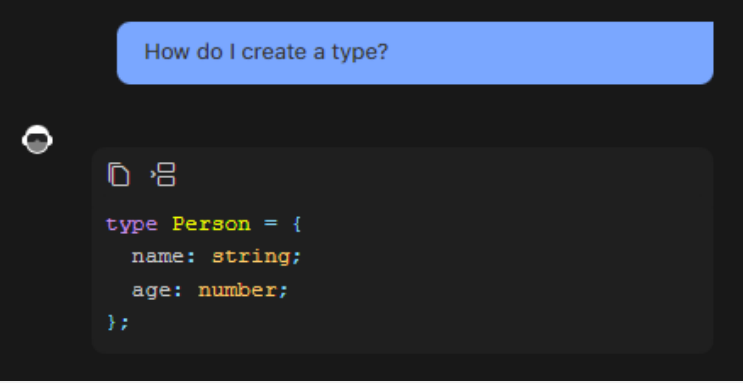
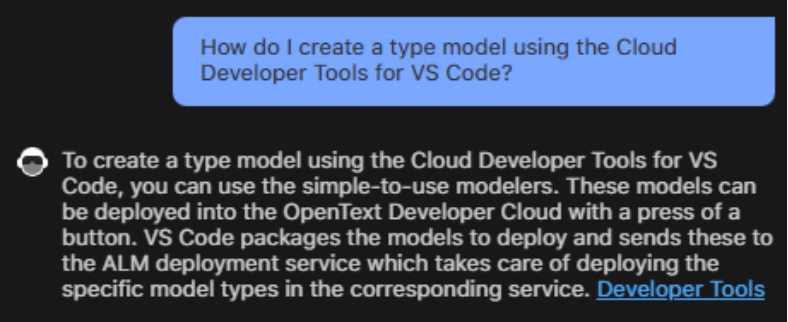
Question	What are the content APIs and how are they linked?
Our test answer	 <p>What are the content APIs and how are they linked?</p> <p>I'm not sure what you mean by "content APIs". Can you rephrase your question?</p>
Question	What are the CMS and CSS APIs and how are they linked?
Our test answer	 <p>What are the CMS and CSS APIs and how are they linked?</p> <p>CSS can be used independently or in conjunction with CMS. When used independently, CSS gives the file creator and the client they used access to the file. When used with CMS, CMS provides ACL based access control. CMS APIs are tenant-based and handle object metadata. CSS is used by CMS when storing document objects, and CMS provides authorization management.</p> <p><a href="#">Content Storage Service   Developer   OpenText</a></p>
Question	What are the Cloud Edition Products?
Our test answer	 <p>What are the Cloud Edition Products?</p> <p>The Cloud Edition Products are a set of cloud-based services that provide a comprehensive solution for managing information. The products include Active Community, Active Documents, Active Orders - Trading Grid, AppWorks Gateway, AppWorks Platform, Carbonite Endpoint, Carbonite Replication, Cloud Fax, and Documentum. <a href="#">Cloud Editions   Developer   OpenText</a></p>
Question	Can you help with the Contract Approval tutorial please?
Our test answer	 <p>Can you help with the Contract Approval tutorial please?</p> <p>The Contract Approval tutorial provides supporting resources for you to get started, including a Tutorial overview about the sample app, detailed step-by-step Tutorial, and User Guide. The Contract Approval application allows a user to upload a document and create a new contract approval workflow.</p> <p><a href="#">Developer Tools</a></p>



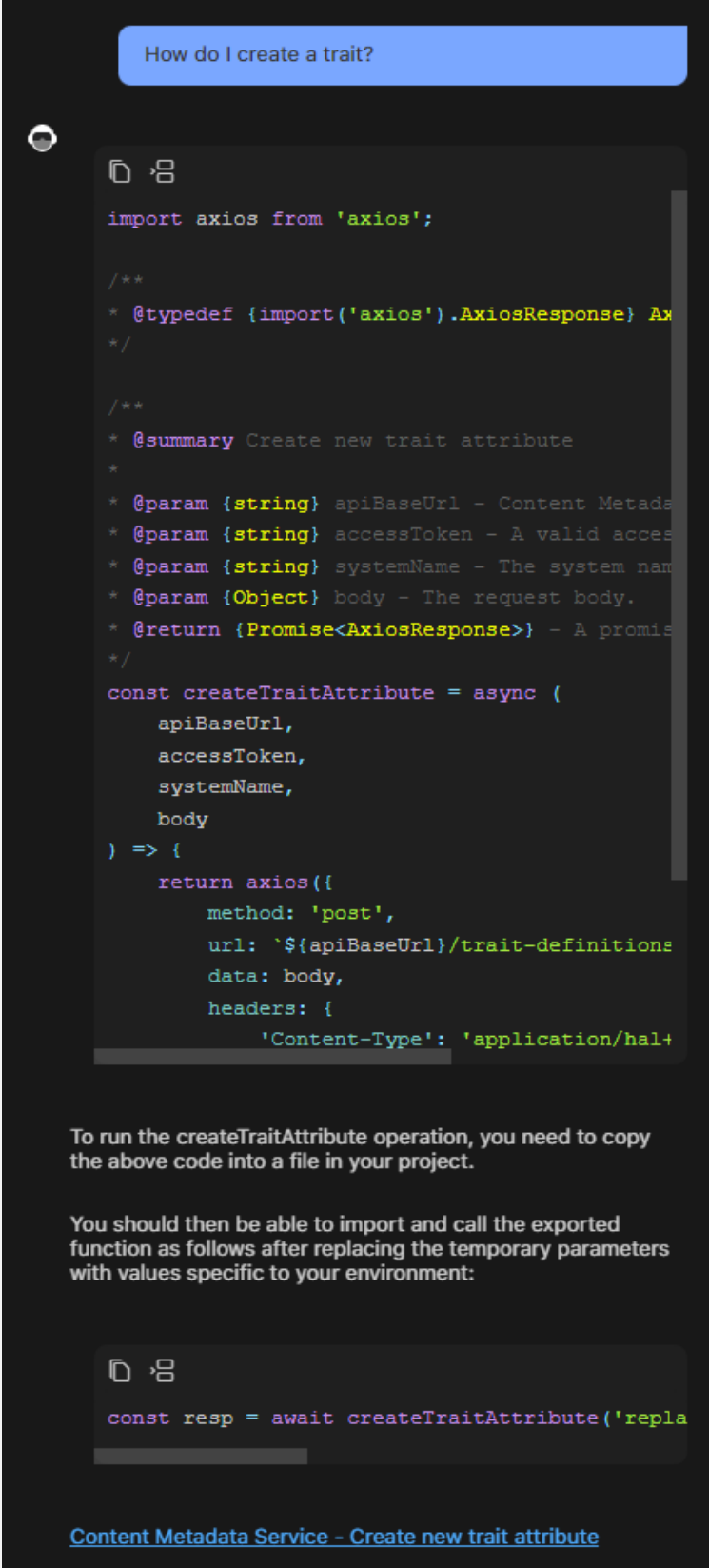
Question	How do I start building an application using OpenText VS Code Extensions?
Our test answer	 <p>How do I start building an application using OpenText VS Code Extensions?</p> <p>To start building an application using OpenText VS Code Extensions, you can install the OpenText Cloud Developer Tools - Extension Pack directly from within VS Code via the Extensions View by searching for "opentext". The OCP cli can be found on npm, the Node Package Manager repository. <a href="#">Developer Tools</a></p>
Question	How can I create models using VS Code Opentext extensions?
Our test answer	 <p>How can I create models using VS Code Opentext extensions?</p> <p>To create models using VS Code Opentext extensions, follow these steps:</p> <ol style="list-style-type: none"> <li><b>1. Install the Opentext extension.</b> <ul style="list-style-type: none"> <li>Open VS Code and click on the <b>Extensions</b> icon in the left sidebar.</li> <li>Search for "Opentext" and click on the <b>Install</b> button.</li> </ul> </li> <li><b>2. Create a new Opentext project.</b> <ul style="list-style-type: none"> <li>Click on the <b>File</b> menu and select <b>New &gt; Opentext Project</b>.</li> <li>Select a location for your project and click on the <b>Create</b> button.</li> </ul> </li> <li><b>3. Add a model to your project.</b> <ul style="list-style-type: none"> <li>Right-click on the project folder in the Explorer and select <b>New &gt; Model</b>.</li> <li>Enter a name for your model and click on the <b>Create</b> button.</li> </ul> </li> <li><b>4. Edit your model.</b> <ul style="list-style-type: none"> <li>Double-click on the model file to open it in the editor.</li> <li>You can use the editor to add and edit the elements of your model.</li> </ul> </li> <li><b>5. Build your model.</b> <ul style="list-style-type: none"> <li>Click on the <b>Build</b> button in the toolbar to build your model.</li> </ul> </li> <li><b>6. Run your model.</b> <ul style="list-style-type: none"> <li>Click on the <b>Run</b> button in the toolbar to run your model.</li> </ul> </li> <li><b>7. View the results of your model.</b> <ul style="list-style-type: none"> <li>The results of your model will be displayed in the Output window.</li> </ul> </li> </ol>

Question	Which are the different models I can I create using the VS Code extensions?
Our test answer	 <p>The screenshot shows a chat window with a blue question bubble and a white answer bubble. The question bubble contains the text: "Which are the different models I can I create using the VS Code extensions?". The answer bubble contains the text: "The VS Code extensions allow you to model object, file, and folder types for the Content Metadata Service (CMS), to model permissions for these types, and to create workflows for the Workflow service. <a href="#">Developer Tools</a>".</p>

<p>Question</p>	<p>How do I create a namespace?</p>
<p>Our test answer</p>	<div data-bbox="549 371 1295 2018"> <p>How do I create a namespace?</p> <pre> /**  * @summary Create a new namespace  *  * @param {string} apiBaseUrl - Content Metadata Service API Base URL  * @param {string} accessToken - A valid access token  * @param {Object} body - The request body.  * @return {Promise&lt;AxiosResponse&gt;} - A promise that resolves to the response  */ const createNamespace = async (   apiBaseUrl,   accessToken,   body ) =&gt; {   return axios({     method: 'post',     url: `\${apiBaseUrl}/namespaces`,     data: body,     headers: {       'Content-Type': 'application/hal+json',       Authorization: `Bearer \${accessToken}`     }   }); };  export default createNamespace; </pre> <p>To run the createNamespace operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <pre>const resp = await createNamespace('replace-with-api-base-url', 'replace-with-access-token', {   // request body });</pre> <p><a href="#">Content Metadata Service - Create a new namespace</a></p> </div>

Question	How do I create a type?
Our test answer	 <p>The screenshot shows a chat window with a blue question bubble: "How do I create a type?". Below it is an AI response icon and a code block containing the following TypeScript code:</p> <pre>type Person = {   name: string;   age: number; };</pre>
Question	How do I create a type model using the Cloud Developer Tools for VS Code?
Our test answer	 <p>The screenshot shows a chat window with a blue question bubble: "How do I create a type model using the Cloud Developer Tools for VS Code?". Below it is an AI response icon and a text block:</p> <p>To create a type model using the Cloud Developer Tools for VS Code, you can use the simple-to-use modelers. These models can be deployed into the OpenText Developer Cloud with a press of a button. VS Code packages the models to deploy and sends these to the ALM deployment service which takes care of deploying the specific model types in the corresponding service. <a href="#">Developer Tools</a></p>

<p>Question</p>	<p>How do I create a type definition?</p>
<p>Our test answer</p>	<div data-bbox="549 371 1294 2031"> <p>How do I create a type definition?</p> <pre> import axios from 'axios';  /**  * @typedef {import('axios').AxiosResponse} AxiosResponse  */  /**  * @summary Create a new custom type definition  * @description Create a new custom type definition  *  * @param {string} apiBaseUrl - Content Metadata Service API Base URL  * @param {string} accessToken - A valid access token  * @param {string} category - The category type  * @param {URLSearchParams} params - The request parameters  * @param {Object} body - The request body.  * @return {Promise&lt;AxiosResponse&gt;} - A promise that resolves to an AxiosResponse  */ const createTypeDefinition = async (   apiBaseUrl,   accessToken,   category,   params,   body ) =&gt; {   return axios({     method: 'post',     url: `\${apiBaseUrl}/type-definitions/`   }); }; </pre> <p>To run the createTypeDefinition operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <pre> const resp = await createTypeDefinition('repla </pre> <p><a href="#">Content Metadata Service - Create a new custom type definition for the given category.</a></p> </div>

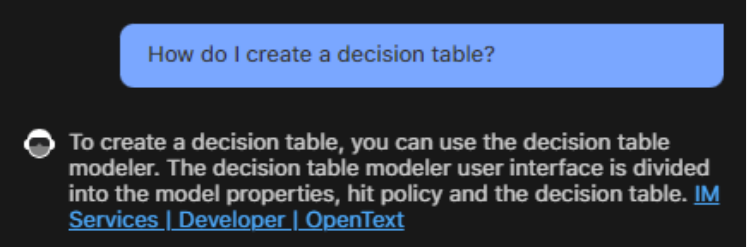
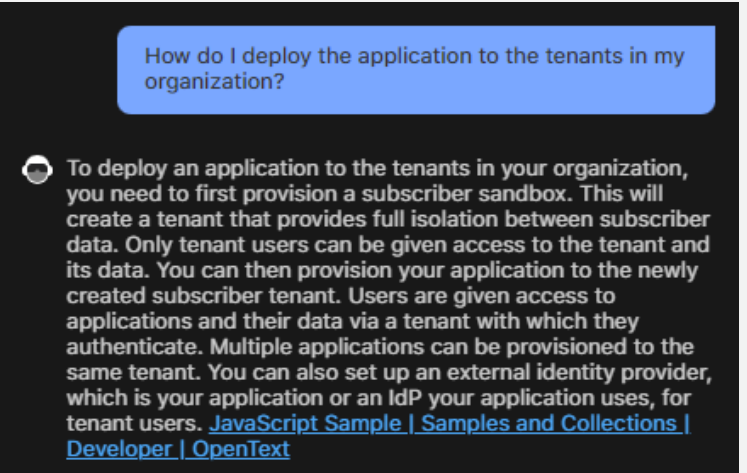
Question	How do I create a trait?
Our test answer	 <p>The screenshot shows a code editor with a dark theme. At the top, a blue search bar contains the text "How do I create a trait?". Below it, a code editor displays the following TypeScript code:</p> <pre>import axios from 'axios';  /**  * @typedef {import('axios').AxiosResponse} AxiosResponse  */  /**  * @summary Create new trait attribute  *  * @param {string} apiBaseUrl - Content Metadata Service API Base URL  * @param {string} accessToken - A valid access token  * @param {string} systemName - The system name  * @param {Object} body - The request body.  * @return {Promise&lt;AxiosResponse&gt;} - A promise that resolves to the response  */ const createTraitAttribute = async (   apiBaseUrl,   accessToken,   systemName,   body ) =&gt; {   return axios({     method: 'post',     url: `\${apiBaseUrl}/trait-definitions`,     data: body,     headers: {       'Content-Type': 'application/hal+json'     }   }); };</pre> <p>Below the code, there is explanatory text:</p> <p>To run the createTraitAttribute operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <pre>const resp = await createTraitAttribute('repla</pre> <p>At the bottom of the screenshot, there is a blue link: <a href="#">Content Metadata Service - Create new trait attribute</a></p>

Question	Can you show me how to do this using fetch?
Our test answer	 <p>The screenshot shows a chat window with a blue input bubble containing the question "Can you show me how to do this using fetch?". Below it is a dark-themed code editor with a light blue header bar containing the same question. The code is as follows:</p> <pre>// Fetch the data from the API fetch('https://api.example.com/data')   .then(response =&gt; response.json())   .then(data =&gt; {     // Process the data here     console.log(data);   })   .catch(error =&gt; {     // Handle any errors here     console.error(error);   });</pre>
Question	I'm using Common JS instead of ECMA.
Our test answer	 <p>The screenshot shows a chat window with a blue input bubble containing the question "I'm using Common JS instead of ECMA.". Below it is a dark-themed chat bubble with a light blue header bar containing the same question. The text in the bubble is:</p> <p>I'm sorry I could not find any information about using Common JS instead of ECMA. However, I found that OpenText AppWorks uses common web technologies such as HTML, CSS and JavaScript and may be tested in any environment where these technologies may be rendered (eg a browser). There is no AppWorks specific IDE, in fact any text editing software may be used to generate the source files, from Sublime Text to Eclipse. <a href="#">First steps for creating mobile apps with AppWorks   Developer   OpenText</a></p>
Question	How do I create a workflow model?
Our test answer	 <p>The screenshot shows a chat window with a blue input bubble containing the question "How do I create a workflow model?". Below it is a dark-themed chat bubble with a light blue header bar containing the same question. The text in the bubble is:</p> <p>The workflow modeler is the primary means to create BPMN diagrams and its user interface is roughly divided into four areas: the menu bar, palette, canvas, and attribute bar. <a href="#">Workflow Service   Developer   OpenText</a></p>

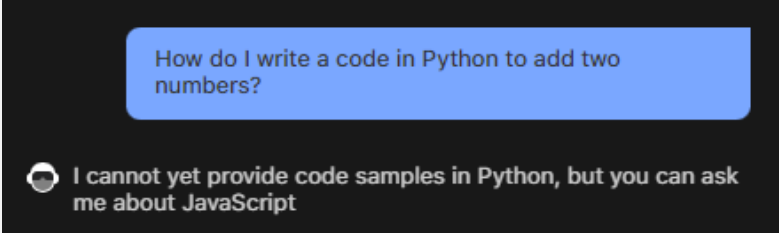
<p>Question</p>	<p>How do I create a workflow model using code?</p>
<p>Our test answer</p>	<div data-bbox="549 376 1342 1944" style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <div style="background-color: #4a90e2; color: white; padding: 5px; border-radius: 10px; text-align: center; margin-bottom: 10px;">             How do I create a workflow model using code?         </div> <div style="background-color: #2d3748; color: #c6c8ca; padding: 10px; border-radius: 10px; margin-bottom: 10px;"> <pre> import axios from 'axios';  /**  * @typedef {import('axios').AxiosResponse} AxiosR  */  /**  * @summary Create a non-executable process model  * @description Save the contents of BPMN based pr  *  * Sample request body :  * {  *   "name": "SampleModel",  *   "key": "SampleKey",  *   "category": "SampleCategory",  *   "modelType": "xml/json",  *   "modelContent": "encoded BPMN xml content for  * }  *  * @param {string} apiBaseUrl - Workflow REST API !  * @param {string} accessToken - A valid access to  * @param {Object} body - The request body.  * @return {Promise&lt;AxiosResponse&gt;} - A promise fo  */ const createWorkflowModel = async (             </pre> </div> <p>To run the createWorkflowModel operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <div style="background-color: #2d3748; color: #c6c8ca; padding: 10px; border-radius: 10px; margin-bottom: 10px;"> <pre> const resp = await createWorkflowModel('replace-wit             </pre> </div> <p style="text-align: center;"><a href="#">Workflow REST API - Create a non-executable process model</a></p> </div>

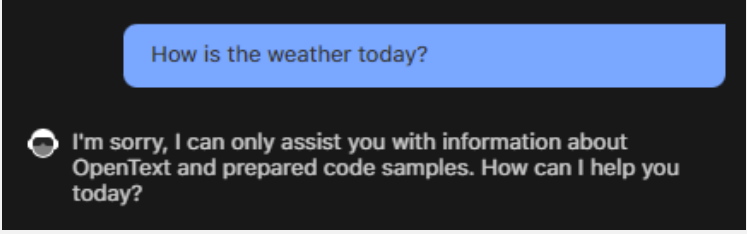
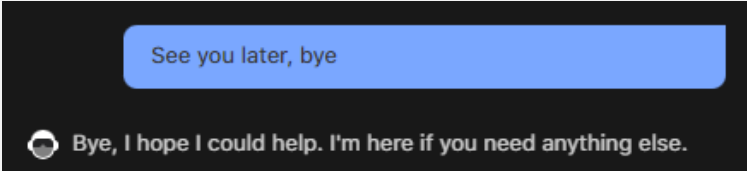


<p>Question</p>	<p>How do I create a workflow model using code and calling the REST API via fetch?</p>
<p>Our test answer</p>	<div data-bbox="719 416 1321 506" style="background-color: #e0e0ff; border-radius: 10px; padding: 5px; margin-bottom: 10px;"> <p>How do I create a workflow model using code and calling the REST API via fetch?</p> </div> <div data-bbox="635 577 1321 1525" style="background-color: #2e3436; color: #eeeeec; padding: 10px; border-radius: 10px;"> <pre> /**  * @summary Get non-executable process models  * @description Get the non-executable process mod  *  * @param {string} apiBaseUrl - Workflow REST API  * @param {string} accessToken - A valid access to  * @param {URLSearchParams} params - The request p  * @return {Promise&lt;Response&gt;} - A promise for the  */ const listModels = async (   apiBaseUrl,   accessToken,   params ) =&gt; {   const response = await fetch(`\${apiBaseUrl}/d     method: 'get',     headers: {       Authorization: `Bearer \${accessToken}     }   });   if (!response.ok) {     throw new Error(`Status: \${response.statu   }   return response.json(); };  export default listModels; </pre> </div> <p>To run the listModels operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <div data-bbox="635 1827 1321 1944" style="background-color: #2e3436; color: #eeeeec; padding: 10px; border-radius: 10px; margin-top: 10px;"> <pre> const resp = await listModels('replace-with-api-ba: </pre> </div> <p style="text-align: center; margin-top: 10px;"><a href="#">Workflow REST API - Get non-executable process models</a></p>

Question	How do I create a decision table?
Our test answer	 <p>How do I create a decision table?</p> <p>To create a decision table, you can use the decision table modeler. The decision table modeler user interface is divided into the model properties, hit policy and the decision table. <a href="#">IM Services</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></p>
Question	How do I deploy the application to the tenants in my organization?
Our test answer	 <p>How do I deploy the application to the tenants in my organization?</p> <p>To deploy an application to the tenants in your organization, you need to first provision a subscriber sandbox. This will create a tenant that provides full isolation between subscriber data. Only tenant users can be given access to the tenant and its data. You can then provision your application to the newly created subscriber tenant. Users are given access to applications and their data via a tenant with which they authenticate. Multiple applications can be provisioned to the same tenant. You can also set up an external identity provider, which is your application or an IdP your application uses, for tenant users. <a href="#">JavaScript Sample</a>   <a href="#">Samples and Collections</a>   <a href="#">Developer</a>   <a href="#">OpenText</a></p>

<p>Question</p>	<p>How do I import a JavaScript component into my code?</p>
<p>Our test answer</p>	<div data-bbox="549 376 1295 2033"> <p>How do I import a JavaScript component into my code?</p> <pre> import axios from 'axios';  /**  * @typedef {import('axios').AxiosResponse} AxiosResponse  */  /**  * @summary Get user  * @description Retrieve existing user by ID  *  * @param {string} apiBaseUrl - Administration  * @param {string} accessToken - A valid access token  * @param {string} organizationId - The organization ID  * @param {string} tenantId - The tenantId path parameter  * @param {string} userId - The userId path parameter  * @return {Promise&lt;AxiosResponse&gt;} - A promise  */ const getUser = async (   apiBaseUrl,   accessToken,   organizationId,   tenantId,   userId ) =&gt; {   return axios({     method: 'get',     url: `\${apiBaseUrl}/organizations/\${c </pre> <p>To run the getUser operation, you need to copy the above code into a file in your project.</p> <p>You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <pre> const resp = await getUser('replace-with-api-b </pre> <p><a href="#">Administration API - Get user</a></p> </div>

Question	How do I write a code in Python to add two numbers?
Our test answer	 <p>The screenshot shows a chat window with a blue input box containing the question: "How do I write a code in Python to add two numbers?". Below the input box, a response from an AI assistant is visible, starting with a speech bubble icon and the text: "I cannot yet provide code samples in Python, but you can ask me about JavaScript".</p>
Question	How do I write code to add two numbers in JavaScript?
Our test answer	 <p>The screenshot shows a chat window with a blue input box containing the question: "How do I write code to add two numbers in JavaScript?". Below the input box, a response from an AI assistant is visible, starting with a speech bubble icon and a code block. The code block contains the following JavaScript code:</p> <pre>// Function to add two numbers function add(num1, num2) {   // Return the sum of the two numbers   return num1 + num2; }  // Example usage const result = add(5, 10);  // Print the result console.log(result); // Output: 15</pre>
Question	How do I write code to add two numbers?
Our test answer	 <p>The screenshot shows a chat window with a blue input box containing the question: "How do I write code to add two numbers?". Below the input box, a response from an AI assistant is visible, starting with a speech bubble icon and a code block. The code block contains the following Python code:</p> <pre>x = 5 y = 10 z = x + y print(z) # Output: 15</pre>

Question	How is the weather today?
Our test answer	 <p>The screenshot shows a chat interface with a dark background. A blue bubble on the left contains the text "How is the weather today?". A white bubble on the right contains the text "I'm sorry, I can only assist you with information about OpenText and prepared code samples. How can I help you today?".</p>
Question	See you later, bye
Our test answer	 <p>The screenshot shows a chat interface with a dark background. A blue bubble on the left contains the text "See you later, bye". A white bubble on the right contains the text "Bye, I hope I could help. I'm here if you need anything else."</p>

Next step:

Building an application with the assistance of Thrust Studio Aviator

### 3.3 Building an application with the assistance of Thrust Studio Aviator

Please ask the following questions to the Thrust Studio Aviator. However, differently from the previous section, you will use the Thrust Studio Aviator responses to really build an application that consumes the OpenText™ Thrust Services.

Question	How do I build a simple application that can authenticate and use the OpenText Cloud Platform services?
----------	---

Our test answer

How do I build a simple application that can authenticate and use the OpenText Cloud Platform services?

## Setting up a Node.js project to call OCP APIs

The base service URLs for the different APIs can be obtained via the API documentation on the OpenText Developer site. For instance, to get the URL for the Content Metadata Service, see the *Servers* dropdown [here](#).

### Prerequisites

- Install the [latest version of Node JS](#).
- You must have a trial set up with an organization and tenant. You will also need to create a test app and have the confidential client ID and secret values ready to use for authentication. For background information on this see the [documentation](#).

### Setup

- Create a new project directory and open it with your IDE. [Visual Studio Code](#) can be used for this if you don't have a preferred IDE.
- Open the command line within your IDE in the root directory. Create a new package.json using the default configuration:

```
npm init -y
```

- Add the [Dotenv](#) dependency to read in your configuration

```
npm i dotenv
```

- Create a file in the root of the project called `.env`. Copy the following values into it and replace the temporary config with config for your OCP app:

```
BASE_SERVICE_URL=replace-with-baseServiceUrl
TENANT_ID=replace-with-tenantId
CONF_CLIENT_ID=replace-with-confClientId
```

- Create a file in the root of the project called `.env.secret`. Copy the following value into it and replace the temporary config with config for your OCP app:

```
CLIENT_SECRET=replace-with-clientSecret
```

This is for local development purposes only. The `.env.secret` file should never be committed to source control. The client secret should always be handled using a secure secret management procedure.

- Add the [Axios](#) dependency to make HTTP calls:

```
npm i axios
```

- Set the project type to ES modules:

```
npm pkg set type="module"
```

- Create a file in the root of the project called *fetchAccessToken.js*. Copy the following code into it:

```

import axios from 'axios';

/**
 * @typedef {import('axios').AxiosResponse} AxiosResponse
 */

/**
 * @summary Authenticate and fetch an access token using the
 * @description This operation will use a confidential client
 *
 * @param {string} baseServiceUrl - The base URL for the OCP
 * @param {string} tenantId - The ID for the tenant your app
 * @param {string} confClientId - The ID for the confidential
 * @param {string} clientSecret - The secret for the confidential
 * @return {Promise<string>} - A promise for the access token
 */
const fetchAccessToken = async (
  baseServiceUrl,
  tenantId,
  confClientId,
  clientSecret
) => {
  const wellKnowConfig = await axios.get(`${baseServiceUrl}/well-know`);
  const authResp = await axios({
    method: 'post',
    url: wellKnowConfig.data.token_endpoint,
    headers: {
      'Content-Type': 'application/x-www-form-urlencoded',
      'Authorization': `Basic ${Buffer.from(`${confClientId}:${clientSecret}`).toString('base64')}`
    }
  });
  return authResp.data.access_token;
};

```

- Create a file in the root of the project called *index.js*. Copy the following code into it:

```

import dotenv from "dotenv";
import fetchAccessToken from './fetchAccessToken.js';

dotenv.config();
dotenv.config({ path: '.env.secret' });

const env = process.env;

const startApp = async () => {
  const accessToken = await fetchAccessToken(
    env.BASE_SERVICE_URL,
    env.TENANT_ID,
    env.CONF_CLIENT_ID,
    env.CLIENT_SECRET
  );
  // Make a call to an OCP service and pass in the access token
};

startApp();

```

- Add a start script:

```

npm pkg set scripts.start="node index.js"

```

- Run the app:

```

npm start

```



<p>Question</p>	<p>How can I now add code to get all the type definitions for my application?</p>
<p>Our test answer</p>	<div data-bbox="730 398 1316 488" style="background-color: #ADD8E6; padding: 5px; border-radius: 10px; margin-bottom: 10px;"> <p>How can I now add code to get all the type definitions for my application?</p> </div> <div data-bbox="635 548 1316 1478" style="background-color: #2E3436; color: #EEEEEC; padding: 10px; border-radius: 10px; margin-bottom: 10px;"> <pre> import axios from 'axios';  /**  * @typedef {import('axios').AxiosResponse} AxiosRe  */  /**  * @summary List all type definitions  *  * @param {string} apiBaseUrl - Content Metadata Se  * @param {string} accessToken - A valid access tok  * @param {URLSearchParams} params - The request pa  * @return {Promise&lt;AxiosResponse&gt;} - A promise for  */ const listTypeDefinitions = async (   apiBaseUrl,   accessToken,   params ) =&gt; {   return axios({     method: 'get',     url: `\${apiBaseUrl}/type-definitions`,     params,     headers: {       Authorization: `Bearer \${accessToken}`     }   }); </pre> </div> <p data-bbox="608 1541 1305 1592">To run the listTypeDefinitions operation, you need to copy the above code into a file in your project.</p> <p data-bbox="608 1630 1313 1704">You should then be able to import and call the exported function as follows after replacing the temporary parameters with values specific to your environment:</p> <div data-bbox="635 1765 1316 1892" style="background-color: #2E3436; color: #EEEEEC; padding: 10px; border-radius: 10px; margin-bottom: 10px;"> <pre> const resp = await listTypeDefinitions('replace-with </pre> </div> <p data-bbox="608 1944 1126 1973"><a href="#">Content Metadata Service - List all type definitions</a></p>

# About OpenText

OpenText enables the digital world, creating a better way for organizations to work with information, on-premises or in the cloud. For more information about OpenText (NASDAQ/TSX: OTEX), visit [opentext.com](https://opentext.com).

**Connect with us:**

[OpenText CEO Mark Barrenechea's blog](#)

[Twitter](#) | [LinkedIn](#)