



Red Hat Ansible Lightspeed with IBM watsonx Code Assistant 2.x latest

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant User Guide

Learn how to use Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant 2.x latest Red Hat Ansible Lightspeed with IBM watsonx Code Assistant User Guide

Learn how to use Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide shows you how to use Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	5
CHAPTER 1. INTRODUCTION TO RED HAT ANSIBLE LIGHTSPEED	6
1.1. ABOUT RED HAT ANSIBLE LIGHTSPEED	6
1.1.1. Accessing Red Hat Ansible Lightspeed with IBM watsonx Code Assistant	6
1.1.2. Benefits of using Red Hat Ansible Lightspeed	6
1.2. KEY FEATURES OF RED HAT ANSIBLE LIGHTSPEED	7
1.3. USING RED HAT ANSIBLE LIGHTSPEED WITH IBM WATSONX CODE ASSISTANT	7
1.4. DATA GATHERED TO TRAIN THE IBM WATSONX CODE ASSISTANT MODELS	8
1.4.1. Models	8
1.4.2. Data sources	8
1.4.3. Data telemetry	8
1.4.4. Telemetry data collection notice for the Admin dashboard	8
1.4.4.1. What information does Red Hat collect?	8
1.4.4.2. Personal Data	9
1.4.4.2.1. Retention	9
1.4.4.2.2. Data Security	9
1.4.4.2.3. Data Sharing	9
1.4.4.2.4. Third Party Service Providers	9
1.4.4.2.5. User Control/ Enabling and Disabling Admin Dashboard Telemetry Collection	9
CHAPTER 2. LOGGING INTO THE ANSIBLE LIGHTSPEED ADMINISTRATOR PORTAL	10
2.1. LOGGING IN TO THE ANSIBLE LIGHTSPEED ADMINISTRATOR PORTAL	10
CHAPTER 3. SETTING UP RED HAT ANSIBLE LIGHTSPEED	11
3.1. CONFIGURATION REQUIREMENTS	11
3.2. SETTING UP RED HAT ANSIBLE LIGHTSPEED CLOUD SERVICE	11
3.2.1. Logging in to the Ansible Lightspeed administrator portal	11
3.2.2. Configuring Red Hat Ansible Lightspeed cloud service	12
3.3. SETTING UP RED HAT ANSIBLE LIGHTSPEED ON-PREMISE DEPLOYMENT	13
3.3.1. Overview	13
3.3.1.1. System requirements	13
3.3.1.2. Prerequisites	14
3.3.1.3. Process for configuring a Red Hat Ansible Lightspeed on-premise deployment	14
3.3.2. Installing the Red Hat Ansible Automation Platform operator	14
3.3.3. Creating an OAuth application	15
3.3.4. Creating connection secrets	16
3.3.5. Creating and deploying a Red Hat Ansible Lightspeed instance	19
3.3.6. Updating the Redirect URIs	20
3.3.7. Configuring Ansible VS Code extension for Red Hat Ansible Lightspeed on-premise deployment	21
3.3.8. Updating connection secrets on an existing Red Hat Ansible Automation Platform operator	22
3.4. CONFIGURING CUSTOM MODELS	23
3.4.1. Process for configuring custom models	23
3.4.2. Creating a training data set by using content parser tool	23
3.4.2.1. Methods of creating training data sets	24
3.4.2.2. Supported data sources	24
3.4.2.3. Process of creating a training data set	24
3.4.2.4. Installing content parser tool	24
3.4.2.5. Generating a custom model training data set	26
3.4.2.5.1. Methods of generating a training data set	26
3.4.2.6. Viewing the generated training data set	28
3.4.2.6.1. Structure of custom model training data set	28

3.4.2.6.1.1. Using report.txt file to resolve ansible-lint rule violations	28
3.4.2.7. Resolving ansible-lint rule violations	29
3.4.2.7.1. How does the content parser tool handle rule violations?	29
3.4.2.8. Merging multiple training data sets into a single file	30
3.4.3. Create and deploy a custom model in IBM watsonx Code Assistant	31
3.4.4. Configuring Red Hat Ansible Lightspeed to use custom models	31
3.4.4.1. Configuring the custom model for all Ansible users in your organization	31
3.4.4.2. Configuring the custom model for select Ansible users in your organization	31
CHAPTER 4. LOGGING IN AND OUT OF THE ANSIBLE LIGHTSPEED PORTAL	33
4.1. LOGGING IN TO THE ANSIBLE LIGHTSPEED PORTAL	33
4.2. LOGGING OUT OF THE ANSIBLE LIGHTSPEED PORTAL	33
CHAPTER 5. INSTALLING AND CONFIGURING THE ANSIBLE VS CODE EXTENSION	34
5.1. INSTALLING THE ANSIBLE VS CODE EXTENSION	34
5.2. CONFIGURING THE ANSIBLE VS CODE EXTENSION	35
5.3. LOGGING IN TO ANSIBLE LIGHTSPEED THROUGH THE ANSIBLE VS CODE EXTENSION	36
CHAPTER 6. REQUESTING TASK RECOMMENDATIONS	38
6.1. OVERVIEW	38
6.1.1. Best practices to improve the recommended guidance	38
6.2. REQUESTING CODE RECOMMENDATIONS FOR A SINGLE TASK	39
6.3. REQUESTING CODE RECOMMENDATIONS FOR MULTIPLE TASKS	43
6.4. VIEWING ANSIBLE LIGHTSPEED TRAINING MATCHES	48
6.5. PROVIDING FEEDBACK ON THE ANSIBLE LIGHTSPEED SERVICE	49
CHAPTER 7. GENERATING PLAYBOOKS AND VIEWING PLAYBOOK EXPLANATIONS	52
7.1. BEST PRACTICES TO GENERATE PLAYBOOKS	52
7.2. GENERATING ANSIBLE PLAYBOOKS	52
7.3. VIEWING PLAYBOOK EXPLANATIONS	54
CHAPTER 8. INSTALLING AND CONFIGURING THE ANSIBLE CODE BOT	56
8.1. INSTALLING THE ANSIBLE CODE BOT	56
8.1.1. Uninstalling the Ansible code bot	57
8.2. MANAGING REPOSITORY SCANS	58
8.2.1. Manually scanning your Git repositories	58
8.2.1.1. Manually scanning the repository from the Ansible code bot dashboard	58
8.2.1.2. Manually scanning the repository from GitHub	59
8.2.2. Configuring the Ansible code bot to scan your repository at regular intervals	59
8.2.3. Viewing your repository's scan history	60
8.2.4. Adding or removing repositories from the Ansible code bot	60
8.3. HOW ANSIBLE CODE BOT HANDLES DUPLICATE PULL REQUESTS	61
CHAPTER 9. VIEWING AND MANAGING ADMIN DASHBOARD TELEMETRY	62
9.1. PREREQUISITES	62
9.2. WHAT TELEMETRY DATA IS COLLECTED?	62
9.3. VIEWING THE ADMIN DASHBOARD TELEMETRY	62
9.4. DISABLING THE ADMIN DASHBOARD TELEMETRY	63
CHAPTER 10. TROUBLESHOOTING	65
10.1. TROUBLESHOOTING RED HAT ANSIBLE LIGHTSPEED CONFIGURATION ERRORS	65
10.1.1. Cannot access the Ansible Lightspeed administrator portal	65
10.1.2. Cannot save the API key	65
10.1.3. Cannot configure the model ID due to authentication failure	65
10.1.4. Cannot configure the model ID due to inference failure	65

10.2. TROUBLESHOOTING ANSIBLE VISUAL STUDIO CODE EXTENSION ERRORS	66
10.2.1. Cannot view the generated code recommendations using the Ansible VS Code extension	66
10.2.2. Cannot request code recommendations by using the Ansible VS Code extension	66
10.3. TROUBLESHOOTING ANSIBLE CODE BOT ERRORS	67
10.3.1. Cannot access Ansible code bot	67
10.3.2. Cannot scan your Git repository using Ansible code bot	67
10.3.3. Cannot create pull requests	68

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

CHAPTER 1. INTRODUCTION TO RED HAT ANSIBLE LIGHTSPEED

Learn about Red Hat Ansible Lightspeed with IBM watsonx Code Assistant, its benefits, key features, process, and data gathered to train the IBM watsonx Code Assistant models.

1.1. ABOUT RED HAT ANSIBLE LIGHTSPEED

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant is a generative AI service that helps automation teams create, adopt, and maintain Ansible content more efficiently. It uses natural language prompts to generate code recommendations for automation tasks based on Ansible best practices.

Red Hat Ansible Lightspeed is the cloud service that enables integration of generative AI into Ansible Automation Platform. This document specifically describes the integration of Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

Red Hat Ansible Lightspeed uses IBM watsonx Code Assistant models trained on subject matter expertise across the Ansible ecosystem, which includes Galaxy, GitHub, and Ansible certified and validated content. For ease of use, Red Hat Ansible Lightspeed is integrated with your existing Ansible developer workflows. For example, you can use your existing Git repositories (both public and private) to train your IBM watsonx Code Assistant models. You can also access Lightspeed content suggestions in VS Code through the Ansible VS code extension.

1.1.1. Accessing Red Hat Ansible Lightspeed with IBM watsonx Code Assistant

To use Red Hat Ansible Lightspeed with IBM watsonx Code Assistant, your organization must have:

- A trial or paid subscription to Ansible Automation Platform
- A trial or paid subscription to IBM watsonx Code Assistant

1.1.2. Benefits of using Red Hat Ansible Lightspeed

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant offers the following benefits:

- **Reduces the onboarding learning period for Ansible developers**
With just a basic understanding of YAML syntax, Ansible developers can use natural language prompts in English language to describe the automation goal. Red Hat Ansible Lightspeed then offers Ansible code recommendations to help achieve the automation goal more efficiently. This combination of content and best practice suggestions reduces the learning curve and offers a smoother onboarding experience for new Ansible users.

For example, to get a multitask code recommendation, you can enter the prompt **Install postgresql-server & run postgresql-setup command**. The Ansible Lightspeed service reads the text, interacts with IBM watsonx Code Assistant, and generates code recommendations to automate a multitask that installs a PostgreSQL server and sets up a PostgreSQL database. You can then view and accept the code recommendations to create tasks in an Ansible YAML file.

- **Increases productivity with quality content creation**
Red Hat Ansible Lightspeed offers automation code recommendations that adhere to Ansible best practices, and IBM watsonx Code Assistant provides model fine-tuning features to improve the accuracy of suggested content based on your organization's existing Ansible content.

Therefore, the AI-generated code recommendations are more accurate, more reliable, and integrated with your existing automation development workflows.

- **Extends trust with AI-generated code recommendations**

The AI-generated code recommendations enable you to extend trust, with an automation code base that adheres to accepted Ansible best practices and significant data safeguards.

1.2. KEY FEATURES OF RED HAT ANSIBLE LIGHTSPEED

Red Hat Ansible Lightspeed offers the following key features:

- **Ansible-specific IBM watsonx Code Assistant models**

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant uses Ansible-specific IBM watsonx Granite models unique to your organization, which are provided, managed, and maintained by IBM.

- **Single tasks and multitask generation**

Using natural language prompts, you can generate single task or multiple task recommendations for Ansible task files and playbooks. To request multitask code recommendations, you can enter a sequence of natural language task prompts in a YAML file comment separated by ampersand (&) symbols.

Currently, Red Hat Ansible Lightspeed supports user prompts in English language only. However, there could be instances where the training data that was used to train the IBM watsonx Code Assistant models included non-English language. In such scenarios, the model can generate code recommendations for prompts made in the same non-English language, but the generated code recommendations might or might not be accurate.

- **Content source matching**

For each generated code recommendation, Red Hat Ansible Lightspeed lists content source matches, including details such as potential source, content author, and relevant licenses. You can use this data to gain insight into potential training data sources used to generate the code recommendations.

- **Post-processing capabilities**

Red Hat Ansible Lightspeed offers post-processing capabilities that augment IBM watsonx Code Assistant and improve the quality and accuracy of code recommendations.

- **Content maintenance and modernization**

The Ansible code bot scans existing content collections, roles, and playbooks through Git repositories, and proactively creates pull requests whenever best practices or quality improvement recommendations are available. The bot automatically submits pull requests to the repository, which proactively alerts the repository owner to a recommended change to their content.

1.3. USING RED HAT ANSIBLE LIGHTSPEED WITH IBM WATSONX CODE ASSISTANT

Prerequisites

To use Red Hat Ansible Lightspeed, ensure that you have the following components:

- Trial or paid subscription to Ansible Automation Platform

- Trial or paid subscription to IBM watsonx Code Assistant
- VS Code version 1.70.1 or later
- The Ansible extension for VS Code version 2.8 or later

1.4. DATA GATHERED TO TRAIN THE IBM WATSONX CODE ASSISTANT MODELS

1.4.1. Models

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant uses Ansible-specific IBM watsonx Granite models unique to your organization. These models are provided, managed, and maintained by IBM.

1.4.2. Data sources

IBM watsonx Code Assistant models are trained on Ansible content from Ansible Galaxy, data from public Git repositories, and Red Hat Ansible subject matter expert examples.

If you publish content to Ansible Galaxy and want to restrict your Ansible Galaxy content from being used to train the models, you can opt out of sharing your Ansible Galaxy data in the Ansible Galaxy namespace configuration.

1.4.3. Data telemetry

Red Hat Ansible Lightspeed collects the following telemetry data by default:

- **Operational telemetry data**
This is the data that is required to operate and troubleshoot the Ansible Lightspeed service. For more information, refer the Enterprise Agreement. You cannot disable the collection of operational telemetry data.
- **Admin dashboard telemetry data**
This is the data that provides insight into how your organization users are using the Ansible Lightspeed service, and the metrics are displayed on the Admin dashboard. You can also disable the Admin dashboard telemetry if you no longer want to collect and monitor the telemetry data. For more information about Admin dashboard telemetry, see [Viewing and managing Admin dashboard telemetry](#).

1.4.4. Telemetry data collection notice for the Admin dashboard

In connection with your use of this Red Hat offering, Red Hat may collect telemetry data about your use of the software. This data allows Red Hat to monitor the software and to improve Red Hat offerings and support, including identifying, troubleshooting, and responding to issues that impact users. The data may also be used to enable you to track your entitlements to Red Hat subscriptions and take advantage of future Red Hat purchasing programs. It may also allow Red Hat to assist you in implementing upgrades to minimize service impact. The data may be shared internally within Red Hat to improve the user experience. If you are evaluating Red Hat software, the data will help Red Hat determine if you need assistance.

1.4.4.1. What information does Red Hat collect?

Tools within the software monitor various metrics and this information is transmitted to Red Hat. The following metrics are monitored:

- Organization you are logged into (Organization ID, account number)
- Large language model (or models) that you are connected to
- Prompts and content suggestions, including accept or reject of the content suggestions
- User sentiment feedback

1.4.4.2. Personal Data

Red Hat does not intend to collect personal information. If Red Hat discovers that personal information has been inadvertently received, Red Hat will delete such information. To the extent that any telemetry data constitutes personal data, refer the [Red Hat Privacy Statement](#) for more information about Red Hat's privacy practices.

1.4.4.2.1. Retention

Red Hat retains and stores telemetry data only for as long as it's needed for the purposes described above or as otherwise required or permitted by law.

1.4.4.2.2. Data Security

Red Hat employs technical and organizational measures designed to protect the telemetry data. Data stored in the Red Hat cloud is being protected, where possible, through encryption. Data is also segmented, and therefore is not accessible across organizations.

1.4.4.2.3. Data Sharing

Red Hat may share telemetry data with its business partners in an aggregated form that does not identify customers to help the partners better understand their markets and their customer's use of Red Hat offerings or ensure the successful integration of products jointly supported by those partners.

1.4.4.2.4. Third Party Service Providers

Red Hat may engage certain service providers to assist in the collection and storage of the telemetry data.

1.4.4.2.5. User Control/ Enabling and Disabling Admin Dashboard Telemetry Collection

You cannot disable collection of operational telemetry data. Operational telemetry data includes only data that is necessary to operate and troubleshoot the service. However, you can disable the collection of Admin Dashboard telemetry data. For more information, see [Disabling the Admin dashboard telemetry](#).

CHAPTER 2. LOGGING INTO THE ANSIBLE LIGHTSPEED ADMINISTRATOR PORTAL

As a Red Hat account organization administrator, you can use the Ansible Lightspeed administrator portal to configure settings to connect Red Hat Ansible Lightspeed to IBM watsonx Code Assistant.

2.1. LOGGING IN TO THE ANSIBLE LIGHTSPEED ADMINISTRATOR PORTAL

Use the Ansible Lightspeed administrator portal to connect Red Hat Ansible Lightspeed to IBM watsonx Code Assistant.

Prerequisites

- You have organization administrator privileges to a Red Hat Customer Portal organization with a valid Red Hat Ansible Automation Platform subscription.

Procedure

1. Log in to the [Ansible Lightspeed portal](#) as an organization administrator.
2. Click **Log in** → **Log in with Red Hat**
3. Enter your Red Hat account username and password. The Ansible Lightspeed Service uses Red Hat Single Sign-On (RH-SSO) for authentication.
As part of the authentication process, the Ansible Lightspeed Service checks whether your organization has an active Ansible Automation Platform subscription. On successful authentication, the login screen is displayed along with your username and your assigned user role.
4. From the login screen, click **Admin Portal**.
You are redirected to the Red Hat Ansible Lightspeed with IBM watsonx Code Assistant administrator portal where you can connect Red Hat Ansible Lightspeed to your IBM watsonx Code Assistant instance.

CHAPTER 3. SETTING UP RED HAT ANSIBLE LIGHTSPEED

As a Red Hat customer portal administrator, you must configure Red Hat Ansible Lightspeed to connect to your IBM watsonx Code Assistant instance. This chapter provides information about configuring both the Red Hat Ansible Lightspeed cloud service and on-premise deployment.

3.1. CONFIGURATION REQUIREMENTS

To use Red Hat Ansible Lightspeed with IBM watsonx Code Assistant, your organization must have the following subscriptions:

- A trial or paid subscription to Red Hat Ansible Automation Platform
- A trial or paid subscription to IBM watsonx Code Assistant

You need the following IBM watsonx Code Assistant information:

- **API key**

A unique API key authenticates all requests made from Red Hat Ansible Lightspeed to IBM watsonx Code Assistant. Each Red Hat organization with a valid Ansible Automation Platform subscription must have a configured API key. When an authenticated RH-SSO user creates a task request in Red Hat Ansible Lightspeed, the API key associated with the user's Red Hat organization is used to authenticate the request to IBM watsonx Code Assistant.

- **Model ID**

A unique model ID identifies an IBM watsonx Code Assistant model in your IBM Cloud account. The model ID that you configure in the Ansible Lightspeed administrator portal is used as the default model, and can be accessed by all Ansible Lightspeed users within your organization.



IMPORTANT

You must configure both the API key and the model ID when you are initially configuring Red Hat Ansible Lightspeed.

3.2. SETTING UP RED HAT ANSIBLE LIGHTSPEED CLOUD SERVICE

As a Red Hat customer portal administrator, you must configure Red Hat Ansible Lightspeed cloud service to connect to your IBM watsonx Code Assistant instance.

3.2.1. Logging in to the Ansible Lightspeed administrator portal

Use the Ansible Lightspeed administrator portal to connect Red Hat Ansible Lightspeed to IBM watsonx Code Assistant.

Prerequisites

- You have organization administrator privileges to a Red Hat Customer Portal organization with a valid Red Hat Ansible Automation Platform subscription.

Procedure

1. Log in to the [Ansible Lightspeed portal](#) as an organization administrator.
2. Click **Log in** → **Log in with Red Hat**

3. Enter your Red Hat account username and password. The Ansible Lightspeed Service uses Red Hat Single Sign-On (RH-SSO) for authentication.
As part of the authentication process, the Ansible Lightspeed Service checks whether your organization has an active Ansible Automation Platform subscription. On successful authentication, the login screen is displayed along with your username and your assigned user role.
4. From the login screen, click **Admin Portal**.
You are redirected to the Red Hat Ansible Lightspeed with IBM watsonx Code Assistant administrator portal where you can connect Red Hat Ansible Lightspeed to your IBM watsonx Code Assistant instance.

3.2.2. Configuring Red Hat Ansible Lightspeed cloud service

Use this procedure to configure the Red Hat Ansible Lightspeed cloud service.

Prerequisites

- You have obtained an API key and a model ID from IBM watsonx Code Assistant that you want to use in Red Hat Ansible Lightspeed.
For information about how to obtain an API key and model ID from IBM watsonx Code Assistant, see the [IBM watsonx Code Assistant documentation](#).

Procedure

1. Log in to the [Ansible Lightspeed portal](#) as an organization administrator.
2. From the login screen, click **Admin Portal**.
3. Specify the API key of your IBM watsonx Code Assistant instance:
 - a. Under **IBM Cloud API Key**, click **Add API key**. A screen to enter the **API Key** is displayed.
 - b. Enter the API Key.
 - c. Optional: Click **Test** to validate the API key.
 - d. Click **Save**.
4. Specify the model ID of the model that you want to use:
 - a. Click **Model Settings**.
 - b. Under **Model ID**, click **Add Model ID**. A screen to enter the **Model Id** is displayed.
 - c. Enter the **Model ID** that you obtained in the previous procedure as the default model for your organization.
 - d. Optional: Click **Test model ID** to validate the model ID.
 - e. Click **Save**.
When the API key and model ID is successfully validated, Red Hat Ansible Lightspeed is connected to your IBM watsonx Code Assistant instance.

Additional resources

- [Troubleshooting Red Hat Ansible Lightspeed configuration errors](#)

3.3. SETTING UP RED HAT ANSIBLE LIGHTSPEED ON-PREMISE DEPLOYMENT

As an Red Hat Ansible Automation Platform administrator, you can set up a Red Hat Ansible Lightspeed on-premise deployment and connect it to an IBM watsonx Code Assistant instance. After the on-premise deployment is successful, you can start using the Ansible Lightspeed service with the Ansible Visual Studio (VS) Code extension.

The following capabilities are not yet supported on Red Hat Ansible Lightspeed on-premise deployment:

- Generating playbooks and viewing playbook explanations
- Viewing telemetry data on the Admin dashboard



NOTE

Red Hat Ansible Lightspeed on-premise deployments are supported on Red Hat Ansible Automation Platform version 2.4.

3.3.1. Overview

This section provides information about the system requirements, prerequisites, and the process for setting up a Red Hat Ansible Lightspeed on-premise deployment.

3.3.1.1. System requirements

Your system must meet the following minimum system requirements to install and run the Red Hat Ansible Lightspeed on-premise deployment.

Requirement	Minimum requirement
RAM	5 GB
CPU	1
Local disk	40 GB

To see the rest of the Red Hat Ansible Automation Platform system requirements, see *Chapter 4. System requirements* in the [Red Hat Ansible Automation Platform Planning Guide](#).



NOTE

You must also have a trial or paid subscription to IBM watsonx Code Assistant, and have a model in the IBM watsonx Code Assistant deployment space. For information about creating a IBM watsonx Code Assistant model, see the [IBM watsonx Code Assistant documentation](#). For information about installing IBM watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data, see the [watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data documentation](#).

3.3.1.2. Prerequisites

- You have administrator privileges for Red Hat Ansible Automation Platform.
- Your organization has a trial or paid subscription to both Red Hat Ansible Automation Platform and IBM watsonx Code Assistant.
- Your system meets the minimum system requirements to set up Red Hat Ansible Lightspeed on-premise deployment.
- You have obtained an API key and a model ID from IBM watsonx Code Assistant. For information about obtaining an API key and model ID from IBM watsonx Code Assistant, see the [IBM watsonx Code Assistant documentation](#). For information about installing IBM watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data, see the [watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data documentation](#).
- You have an existing external PostgreSQL database configured for the Red Hat Ansible Automation Platform, or have a database created for you when configuring the Red Hat Ansible Lightspeed on-premise deployment.

3.3.1.3. Process for configuring a Red Hat Ansible Lightspeed on-premise deployment

Perform the following tasks to install and configure a Red Hat Ansible Lightspeed on-premise deployment:

1. [Install the Red Hat Ansible Automation Platform operator](#)
2. [Create an OAuth application](#)
3. [Create connection secrets for Red Hat Ansible Automation Platform and IBM watsonx Code Assistant both](#)
4. [Create and deploy an Red Hat Ansible Lightspeed instance](#)
5. [Update the Redirect URI to connect to your Red Hat Ansible Lightspeed on-premise deployment](#)
6. [Install and configure Ansible Visual Studio Code extension to connect to Red Hat Ansible Lightspeed on-premise deployment](#)
7. Optional: If you want to connect to a different IBM watsonx Code Assistant, [update the connection secrets on an existing Red Hat Ansible Automation Platform operator](#)

3.3.2. Installing the Red Hat Ansible Automation Platform operator

Use this procedure to install the Ansible Automation Platform operator on the Red Hat OpenShift Container Platform.

Prerequisites

- You have installed and configured automation controller.

Procedure

1. Log in to the Red Hat OpenShift Container Platform as an administrator.

2. Create a namespace:
 - a. Go to **Administration** → **Namespaces**.
 - b. Click **Create Namespace**.
 - c. Enter a unique name for the namespace.
 - d. Click **Create**.
3. Install the operator:
 - a. Go to **Operators** → **OperatorHub**.
 - b. Select the namespace where you want to install the Red Hat Ansible Automation Platform operator.
 - c. Search for the Ansible Automation Platform operator.
 - d. From the search results, select the Ansible Automation Platform (provided by Red Hat) tile.
 - e. Select an **Update Channel**. You can select either **stable-2.x** or **stable-2.x-cluster-scoped** as the channel.
 - f. Select the destination namespace if you selected “stable-2.x” as the update channel.
 - g. Select **Install**. It takes a few minutes for the operator to be installed.
4. Click **View Operator** to see the details of your newly installed Red Hat Ansible Automation Platform operator.

3.3.3. Creating an OAuth application

Use this procedure to create an OAuth application for your Red Hat Ansible Lightspeed on-premise deployment.

Prerequisites

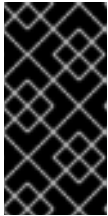
- You have an operational Ansible automation controller instance.

Procedure

1. Log in to the automation controller as an administrator.
2. Under **Administration**, click **Applications** > **Add**
3. Enter the following information:
 - a. **Name**: Specify a unique name for your application.
 - b. **Organization**: Select a preferred organization.
 - c. **Authorization grant type**: Select **Authorization code**.
 - d. **Redirect URIs**: Enter a temporary URL for now, for example, <https://temp/>.
The exact Red Hat Ansible Lightspeed application URL is generated after the on-premise deployment is completed. After the deployment is completed, you must change the

Redirect URI to point to the generated Red Hat Ansible Lightspeed application URL. For more information, see [Updating the Redirect URIs](#).

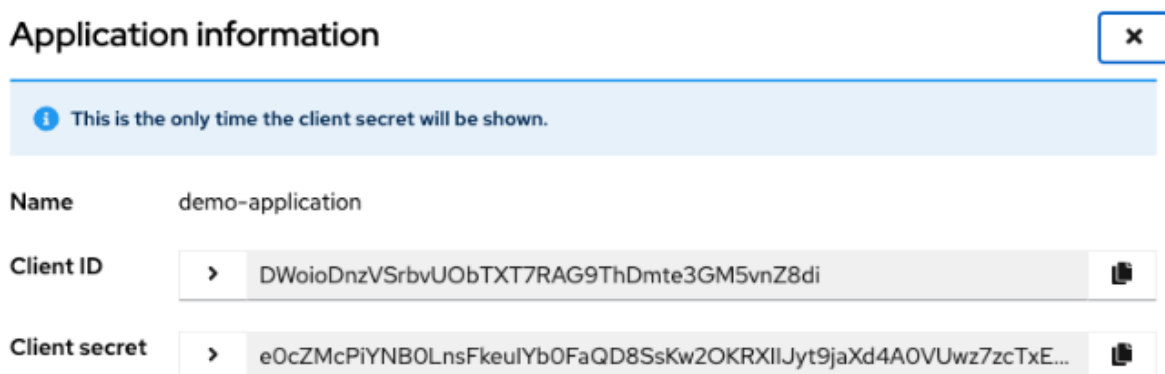
- e. From the **Client type** list, select **Confidential**.
4. Click **Save**.
A pop-up window is displayed along with the generated application client ID and client secret.
5. Copy and save both the generated client ID and client secret for future use.



IMPORTANT

This is the only time the pop-up window is displayed. Therefore, ensure that you copy both the client ID and client secret, as you need these tokens to create connections secrets for Red Hat Ansible Automation Platform and IBM watsonx Code Assistant both.

The following image is an example of the generated client ID and client secret:



3.3.4. Creating connection secrets

Use this procedure to create secrets to connect to Red Hat Ansible Automation Platform and IBM watsonx Code Assistant.

Prerequisites

- You have installed the Ansible Automation Platform operator on the Red Hat OpenShift Container Platform.
- You have created an OAuth application in the automation controller.
- You have obtained an API key and a model ID from IBM watsonx Code Assistant.
For information about obtaining an API key and model ID from IBM watsonx Code Assistant, see the [IBM watsonx Code Assistant documentation](#) . For information about installing IBM watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data, see the [watsonx Code Assistant for Red Hat Ansible Lightspeed on Cloud Pak for Data documentation](#).

Procedure

1. Go to the Red Hat OpenShift Container Platform.

2. Select **Select Workloads → Secrets**.
3. Click **Create → Key/value secret**
4. From the **Projects** list, select the namespace that you created when you installed the Red Hat Ansible Automation Platform operator.
5. Create an **authorization secret** to connect to the Red Hat Ansible Automation Platform:
 - a. Click **Create → Key/value secret**
 - b. In **Secret name**, enter a unique name for the secret. For example, **auth-aiconnect**.
 - c. Add the following keys and their associated values individually:

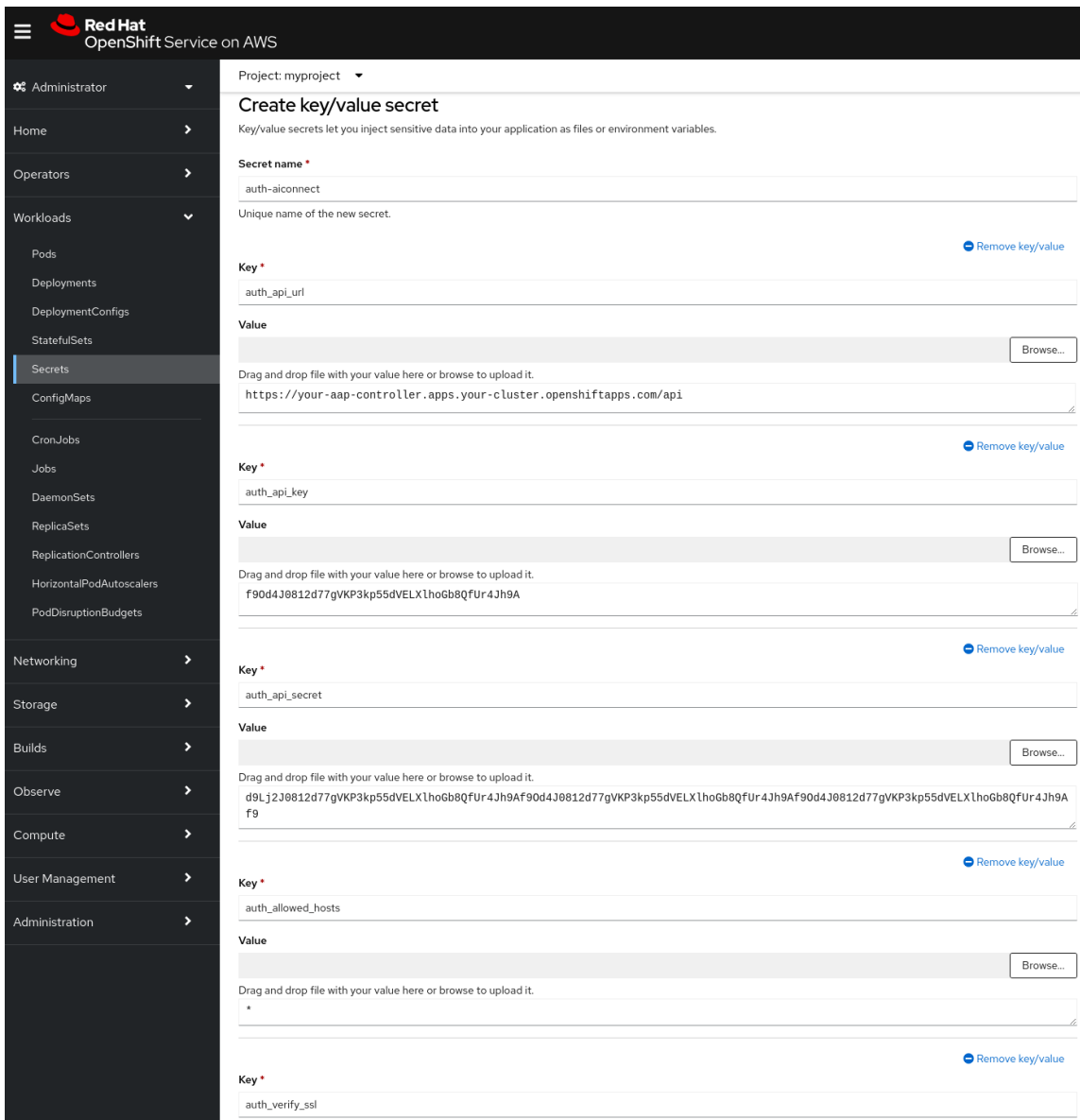
Key	Value
auth_api_url	Enter the API URL of the automation controller in the following format: <a href="https://<CONTROLLER_SERVER_NAME>/api">https://<CONTROLLER_SERVER_NAME>/api .
auth_api_key	Enter the client ID of the OAuth application that you recorded earlier.
auth_api_secret	Enter the client secret of the OAuth application that you recorded earlier.
auth_allowed_hosts	Enter the list of strings representing the host/domain names used by the underlying Django framework to restrict which hosts can access the service. This is a security measure to prevent HTTP Host header attacks. For more information, see Allowed hosts in django documentation .
auth_verify_ssl	Enter the value as true .



IMPORTANT

Ensure that you do not accidentally add any whitespace characters (extra line, space, and so on) to the value fields. If there are any extra or erroneous characters in the secret, the connection to Red Hat Ansible Automation Platform fails.

- d. Click **Create**.
The following image is an example of an authorization secret:



6. Similarly, create a **model secret** to connect to an IBM watsonx Code Assistant model:

- a. Click **Create** → **Key/value secret**
- b. In **Secret name**, enter a unique name for the secret. For example, **model-aiconnect**.
- c. Add the following keys and their associated values individually:

Key	Value
username	Enter the username you use to connect to an IBM Cloud Pak for Data deployment.
model_type	Enter wca-onprem to connect to an IBM Cloud Pak for Data deployment.
model_url	Enter the URL to IBM watsonx Code Assistant.
model_api_key	Enter the API key of your IBM watsonx Code Assistant model in your IBM Cloud Pak for Data deployment.

Key	Value
model_id	Enter the model ID of your IBM watsonx Code Assistant model in your IBM Cloud Pak for Data deployment.



IMPORTANT

Ensure that you do not accidentally add any whitespace characters (extra line, space, and so on) to the value fields. If there are any extra or erroneous characters in the secret, the connection to IBM watsonx Code Assistant fails.

- d. Click **Create**.

3.3.5. Creating and deploying a Red Hat Ansible Lightspeed instance

Use this procedure to create and deploy a Red Hat Ansible Lightspeed instance to your namespace.

Prerequisites

- You have created connection secrets for both Red Hat Ansible Automation Platform and IBM watsonx Code Assistant.

Procedure

1. Go to Red Hat OpenShift Container Platform.
2. Select **Operators** → **Installed Operators**.
3. From the **Projects** list, select the namespace where you want to install the Red Hat Ansible Lightspeed instance and where you created the connection secrets.
4. Locate and select the **Ansible Automation Platform (provided by Red Hat)** operator that you installed earlier.
5. Select **All instances** → **Create new**.
6. From the **Create new** list, select the **Ansible Lightspeed** modal.
7. Ensure that you have selected **Form view** as the configuration mode.
8. Provide the following information:
 - a. **Name:** Enter a unique name for your Red Hat Ansible Lightspeed instance.
 - b. **Secret where the authentication information can be found** Select the authentication secret that you created to connect to the Red Hat Ansible Automation Platform.
 - c. **Secret where the model configuration can be found** Select the model secret that you created to connect to IBM watsonx Code Assistant.
9. Click **Create**.

The Red Hat Ansible Lightspeed instance takes a few minutes to deploy to your namespace. After the installation status is displayed as **Successful**, the Ansible Lightspeed portal URL is available under **Networking → Routes** in Red Hat OpenShift Container Platform.

3.3.6. Updating the Redirect URIs

When Ansible users log in or log out of Ansible Lightspeed service, the Red Hat Ansible Automation Platform redirects the user's browser to a specified URL. You must configure the redirect URLs so that users can log in and log out of the application successfully.

Prerequisites

- You have created and deployed a Red Hat Ansible Lightspeed instance to your namespace.

Procedure

1. Get the URL of the Ansible Lightspeed instance:
 - a. Go to Red Hat OpenShift Container Platform.
 - b. Select **Networking → Routes**
 - c. Locate the Red Hat Ansible Lightspeed instance that you created.
 - d. Copy the Location URL of the Red Hat Ansible Lightspeed instance.
2. Update the **login** redirect URI:
 - a. In the automation controller, go to **Administration → Applications**
 - b. Select the Lightspeed Oauth application that you created.
 - c. In the **Redirect URIs** field of the Oauth application, enter the URL in the following format: https://<lightspeed_route>/complete/aap/

An example of the URL is <https://lightspeed-on-prem-web-service.com/complete/aap/>.



IMPORTANT

The Redirect URL must include the following information:

- The prefix **https://**
- The **<lightspeed_route>** URL, which is the URL of the Red Hat Ansible Lightspeed instance that you copied earlier
- The suffix **/complete/aap/** that includes a backslash sign (`/`) at the end

- d. Click **Save**.

3. Update the **logout** redirect URI:
 - a. Log in to the cluster on which the Red Hat Ansible Automation Platform instance is running.
 - b. Identify the **AutomationController** custom resource.

- c. Select **[YAML view]**.
- d. Add the following entry to the YAML file:

```
``yaml
spec:
  ...
  extra_settings:
    - setting: LOGOUT_ALLOWED_HOSTS
      value: "<lightspeed_route-HostName>"
  ...
```

IMPORTANT

Ensure the following while specifying the **value:** parameter:

- Specify the hostname without the network protocol, such as **https://**. For example, the correct hostname would be **my-aiconnect-instance.somewhere.com**, and not <https://my-aiconnect-instance.somewhere.com>.
- Use the single and double quotes exactly as specified in the codeblock. If you change the single quotes to double quotes and vice versa, you will encounter errors when logging out.
- Use a comma to specify multiple instances of Red Hat Ansible Lightspeed deployment. If you are running multiple instances of Red Hat Ansible Lightspeed application with a single Red Hat Ansible Automation Platform deployment, add a comma (,) and then add the other hostname values. For example, you can add multiple hostnames, such as **"my-lightspeed-instance1.somewhere.com','my-lightspeed-instance2.somewhere.com"**

4. Apply the revised YAML. This task restarts the automation controller pods.

3.3.7. Configuring Ansible VS Code extension for Red Hat Ansible Lightspeed on-premise deployment


To access the on-premise deployment of Red Hat Ansible Lightspeed, all Ansible users within your organization must install the Ansible Visual Studio (VS) Code extension in their VS Code editor, and configure the extension to connect to the on-premise deployment.

Prerequisites

- You have installed VS Code version 1.70.1 or later.

Procedure

1. Open the VS Code application.
2. From the **Activity** bar, click the **Extensions** icon.
3. From the **Installed Extensions** list, select **Ansible**.

4. From the **Ansible** extension page, click the **Settings** icon () and select **Extension Settings**.
5. Select **Ansible Lightspeed** settings and specify the following information:
 - In the **URL for Ansible Lightspeed** field, enter the **Route URL** of the Red Hat Ansible Lightspeed on-premise deployment. Ansible users must have Ansible Automation Platform controller credentials.
 - Optional: If you want to use a custom model instead of the default model, in the **Model ID Override** field, enter the custom model ID. Your settings are automatically saved in VS Code.
After configuring Ansible VS Code extension to connect to Red Hat Ansible Lightspeed on-premise deployment, you must [log in to Ansible Lightspeed through the Ansible VS Code extension](#).

3.3.8. Updating connection secrets on an existing Red Hat Ansible Automation Platform operator

After you have set up the Red Hat Ansible Lightspeed on-premise deployment successfully, you can modify the deployment if you want to connect to another IBM watsonx Code Assistant model. For example, you connected to the default IBM watsonx Code Assistant model but now want to connect to a custom model instead. To connect to another IBM watsonx Code Assistant model, you must create new connection secrets, and then update the connection secrets and parameters on an existing Red Hat Ansible Automation Platform operator.

Prerequisites

- You have set up a Red Hat Ansible Lightspeed on-premise deployment.
- You have obtained an API key and a model ID of the IBM watsonx Code Assistant model you want to connect to.
- You have created new authorization and model connection secrets for the IBM watsonx Code Assistant model that you want to connect to. For information about creating authorization and model connection secrets, see [Creating connection secrets](#).

Procedure

1. Go to the Red Hat OpenShift Container Platform.
2. Select **Operators → Installed Operators**.
3. From the **Projects** list, select the namespace where you installed the Red Hat Ansible Lightspeed instance.
4. Locate and select the **Ansible Automation Platform (provided by Red Hat)** operator that you installed earlier.
5. Select the **Ansible Lightspeed** tab.
6. Find and select the instance you want to update.
7. Click **Actions > Edit AnsibleLightspeed** The editor will switch to a text or YAML view.

8. Scroll the text to find the **spec:** section.

```
uid: 319dbc7e-10df-4215-a38f-58c40a9a5e64
spec:
  service_type: ClusterIP
  ingress_type: Route
  no_log: true
  image_pull_policy: IfNotPresent
  admin_email: test@example.com
  model_config_secret_name: model-config
  auth_config_secret_name: auth-config
  set_self_labels: true
  api:
    replicas: 1
  database:
    postgres_data_volume_init: false
  admin_user: admin
```

9. Find the entry for the secret you have changed and saved under a new name.

10. Change the name of the secrets to the new secrets.

11. Click **Save**.

The new AnsibleLightspeed Pods are created. After the new pods are running successfully, the old AnsibleLightspeed Pods are terminated.

3.4. CONFIGURING CUSTOM MODELS

As an organization administrator, you can create and use fine-tuned, custom models that are trained on your organization's existing Ansible content. With this capability, you can tune the models to your organization's automation patterns and improve the code recommendation experience.

You can configure multiple custom models for your organization. For example, you can create a custom model for your corporate IT automation team and a different one for your engineering team's infrastructure. You can also configure a custom model to make it available for all Ansible users or select Ansible users in your organization.

3.4.1. Process for configuring custom models

To configure a custom model, perform the following tasks:

1. [Create a training data set by using content parser tool](#)
2. [Create and deploy a custom model by using IBM watsonx Code Assistant](#)
3. [Configure Red Hat Ansible Lightspeed to use the custom model](#)

3.4.2. Creating a training data set by using content parser tool

Use the content parser tool, a command-line interface (CLI) tool, to scan your existing Ansible files and generate a custom model training data set. The training data set includes a list of Ansible files and their

paths relative to the project root. You can then upload this data set to IBM watsonx Code Assistant, and use it to create a custom model that is trained on your organization's existing Ansible content.

3.4.2.1. Methods of creating training data sets

You can generate a training data set by using one of the following methods:

- **With ansible-lint preprocessing**

By default, the content parser tool generates training data sets by using ansible-lint preprocessing. The content parser tool uses ansible-lint rules to scan your Ansible files and ensure that the content adheres to Ansible best practices. If rule violations are found, the content parser tool excludes these files from the generated output. In such scenarios, you must resolve the rule violations, and run the content parser tool once again so that the generated output includes all your Ansible files.

- **Without ansible-lint preprocessing**

You can generate a training data set without ansible-lint preprocessing. In this method, the content parser tool does not scan your Ansible files for ansible-lint rule violations; therefore, the training data set includes all files. Although the training data set includes all files, it might not adhere to Ansible best practices and could affect the quality of your code recommendation experience.

3.4.2.2. Supported data sources

The content parser tool scans the following directories and file formats:

- Local directories
- Archived files, such as **.zip**, **.tar**, **.tar.gz**, **.tar.bz2**, and **.tar.xz** files
- Git repository URLs (includes both private and public repositories)

3.4.2.3. Process of creating a training data set

To create a custom model training data set, perform the following tasks:

1. [Install content parser tool on your computer](#)
2. [Generate a custom model training data set](#)
3. [View the generated training data set](#)
4. (Optional: If you generated a training data set with ansible-lint preprocessing and detected ansible-lint rule violations) [Resolve ansible-lint rule violations](#)
5. (Optional: If you generated multiple training data sets) [Merge multiple training data sets into a single JSONL file](#)

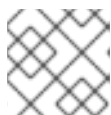
3.4.2.4. Installing content parser tool

Install the content parser tool, a command-line interface (CLI) tool, on your computer.

Prerequisites

Ensure that your computer has one of the following supported OS:

- Python version 3.10 or later.
- UNIX OS, such as Linux or Mac OS.

**NOTE**

Installation of the content parser tool on Microsoft Windows OS is not supported.

Procedure

1. Create a working directory and set up **venv** Python virtual environment:

```
$ python -m venv ./venv
```

```
$ source ./venv/bin/activate
```

2. Install the latest version of content parser tool from the **pip** repository:

```
$ pip install --upgrade pip
```

```
$ pip install --upgrade ansible-content-parser
```

3. Perform one of the following tasks:

- To generate a training data set without ansible-lint preprocessing, go to section [Generating a custom model training data set](#).
- To generate a training data set with ansible-lint preprocessing, ensure that you have the latest version of ansible-lint installed on your computer:

- i. View the ansible-lint versions that are installed on your computer.

```
$ ansible-content-parser --version
```

```
$ ansible-lint --version
```

A list of application versions and their dependencies are displayed.

- ii. In the output, verify that the version of ansible-lint that was installed with the content parser tool is the same as that of the previously-installed ansible-lint. A mismatch in the installed ansible-lint versions causes inconsistent results from the content parser tool and ansible-lint.

For example, in the following output, the content parser tool installation includes ansible-lint version 6.20.0 which is a mismatch from previously-installed ansible-lint version 6.13.1:

```
$ ansible-content-parser --version
ansible-content-parser 0.0.1 using ansible-lint:6.20.0 ansible-core:2.15.4
$ ansible-lint --version
ansible-lint 6.13.1 using ansible 2.15.4
A new release of ansible-lint is available: 6.13.1 → 6.20.0
```

- iii. If there is a mismatch in the ansible-lint versions, deactivate and reactivate **venv** Python virtual environment:

```
$ deactivate
```

```
$ source ./venv/bin/activate
```

- iv. Verify that the version of `ansible-lint` that is installed with the content parser tool is the same as that of the previously-installed `ansible-lint`:

```
$ ansible-content-parser --version
```

```
$ ansible-lint --version
```

For example, the following output shows that both `ansible-lint` installations on your computer are of version 6.20.0:

```
$ ansible-content-parser --version
ansible-content-parser 0.0.1 using ansible-lint:6.20.0 ansible-core:2.15.4
$ ansible-lint --version
ansible-lint 6.20.0 using ansible-core:2.15.4
ansible-compat:4.1.10 ruamel-yaml:0.17.32 ruamel-yaml-clib:0.2.7
```

3.4.2.5. Generating a custom model training data set

After installing content parser tool, run it to scan your custom Ansible files and generate a custom model training data set. The training data set includes a list of Ansible files and their paths relative to the project root. You can then upload the training data set to IBM watsonx Code Assistant and create a custom model for your organization. If you used `ansible-lint` preprocessing and encountered rule violations, you must [resolve the rule violations](#) before uploading the training data set to IBM watsonx Code Assistant.

3.4.2.5.1. Methods of generating a training data set

You can generate a training data set by using one of the following methods:

- **With `ansible-lint` preprocessing**
By default, the content parser tool generates training data sets by using `ansible-lint` preprocessing. The content parser tool uses `ansible-lint` rules to scan your Ansible files and ensure that the content adheres to Ansible best practices. If rule violations are found, the content parser tool excludes these files from the generated output. In such scenarios, you must resolve the rule violations, and run the content parser tool once again so that the generated output includes all your Ansible files.
- **Without `ansible-lint` preprocessing**
You can generate a training data set without `ansible-lint` preprocessing. In this method, the content parser tool does not scan your Ansible files for `ansible-lint` rule violations; therefore, the training data set includes all files. Although the training data set includes all files, it might not adhere to Ansible best practices and could affect the quality of your code recommendation experience.

Prerequisites

- You must have installed content parser tool on your computer.
- You must have verified that the version of `ansible-lint` that is installed with the content parser tool is the same as that of the previously-installed `ansible-lint`.

Procedure

1. Run the content parser tool to generate a training data set:
 - With `ansible-lint` preprocessing: **`$ ansible-content-parser source output`**

- Without `ansible-lint` preprocessing: **`$ ansible-content-parser source output -S`**
The following table lists the required parameters.

Parameter	Description
<code>source</code>	Specifies the source of the training data set.
<code>output</code>	Specifies the output of the training data set.
<code>-S</code> or <code>--skip-ansible-lint</code>	Specifies to skip <code>ansible-lint</code> preprocessing while generating the training data set.

For example: If the source is a Github URL <https://github.com/ansible/ansible-tower-samples.git>, and the output directory is `/tmp/out`, the command prompt is as follows:

`$ ansible-content-parser https://github.com/ansible/ansible-tower-samples.git /tmp/out`

- Optional: To generate a training data set with additional information, specify the following parameters while running the content parser tool.

Parameter	Description
<code>--source-license</code>	Specifies to include the licensing information of the source directory in the training data set.
<code>--source-description</code>	Specifies to include the descriptions of the source directory in the training data set.
<code>--repo-name</code>	Specifies to include the repository name in the training data set. If you do not specify the repository name, the content parser tool automatically generates it from the source name.
<code>--repo-url</code>	Specifies to include the repository URL in the training data set. If you do not specify the repository URL, the content parser tool automatically generates it from the source URL.
<code>-v</code> or <code>--verbose</code>	Displays the console logging information.

Example of a command prompt for Github repository `ansible-tower-samples`

```
$ ansible-content-parser --profile min \  
--source-license undefined \  
--source-description Samples \  
--repo-name ansible-tower-samples \  
--repo-url 'https://github.com/ansible/ansible-tower-samples' \  
git@github.com:ansible/ansible-tower-samples.git /var/tmp/out_dir
```

Example of a generated training data set for Github repository `ansible-tower-samples`

The training data set is formatted with Jeff Goldblum (`jpg`), a command-line JSON processing tool.

```
$ cat out_dir/ftdata.json| jq
{
  "data_source_description": "Samples",
  "input": "---\n- name: Hello World Sample\n hosts: all\n tasks:\n - name: Hello Message",
  "license": "undefined",
  "module": "debug",
  "output": " debug:\n msg: Hello World!",
  "path": "hello_world.yml",
  "repo_name": "ansible-tower-samples",
  "repo_url": "https://github.com/ansible/ansible-tower-samples"
}
```

3.4.2.6. Viewing the generated training data set

After content parser tool scans your Ansible files, it generates the training data set in an output subdirectory within your local directory. The training data set includes a **ftdata.jsonl** file, which is the main output of the content parser tool. The file is available in JSON Lines files format, where each line entry represents a JSON object. You must upload this JSONL file to IBM watsonx Code Assistant to create a custom model.

3.4.2.6.1. Structure of custom model training data set

The following is the file structure of an output subdirectory:

```
output/
|-- ftdata.jsonl # Training dataset 1
|-- report.txt # A human-readable report 2
|
|-- repository/ 3
|   |-- (files copied from the source repository)
|
|-- metadata/ 4
|   |-- (metadata files generated during the execution)
```

Where:

- 1 **ftdata.jsonl**: A training data set file, which is the main output of the content parser tool. The file is available in JSON Lines files format, where each line entry represents a JSON object. You must upload this JSONL file in IBM watsonx Code Assistant to create a custom model.
- 2 **report.txt**: A human-readable text file that provides a summary of all content parser tool executions.
- 3 **repository**: A directory that contains files from the source repository. Sometimes, ansible-lint updates the directory according to the configured rules, so the file contents of the output directory might differ from the source repository.
- 4 **metadata**: A directory that contains multiple metadata files that are generated during each content parser tool execution.

3.4.2.6.1.1. Using report.txt file to resolve ansible-lint rule violations

The **report.txt** file, that can be used to resolve ansible-lint rule violations, contains the following information:

- File counts per type: A list of files according to their file types, such as playbooks, tasks, handlers, and jinja2.
- List of Ansible files that were identified: A list of files identified by ansible-lint with a file name, a file type, and whether the file was excluded from further processing, or automatically fixed by ansible-lint.
- List of Ansible modules found in tasks: A list of modules identified by ansible-lint with a module name, a module type, and whether the file was excluded from further processing, or automatically fixed by ansible-lint.
- Issues found by ansible-lint: A list of issues along with a brief summary of ansible-lint execution results. If ansible-lint encounters files with syntax-check errors in the first execution, then ansible-lint initiates a second execution and excludes the files with errors from the scan. You can use this information to resolve ansible-lint rule violations.

3.4.2.7. Resolving ansible-lint rule violations

By default, the content parser tool uses ansible-lint rules to scan your Ansible files and ensure that the content adheres to Ansible best practices. If rule violations are found, the content parser tool excludes these files from the generated output. In such scenarios, it is recommended that you fix the files with rule violations before uploading the training data set to IBM watsonx Code Assistant.

By default, ansible-lint applies the rules that are configured in **ansible-lint/src/ansiblelint/rules** while scanning your Ansible files. For more information about ansible-lint rules, see the [Ansible Lint documentation](#).

3.4.2.7.1. How does the content parser tool handle rule violations?

- **Using autofixes**

The content parser tool runs ansible-lint with the **--fix=all** option to perform autofixes, which can fix or simplify fixing issues identified by that rule.

If ansible-lint identifies rule violations that have an associated autofix, it automatically fixes or simplifies the issues that violate the rules. If ansible-lint identifies rule violations that do not have an associated autofix, it reports these instances as rule violations which you must fix manually. For more information about autofixes, see [Autofix](#) in Ansible Lint Documentation.

- **Using syntax-checks**

Ansible-lint also performs syntax checks while scanning your Ansible files. If any syntax-check errors are found, ansible-lint stops processing the files. For more information about syntax-check errors, see [syntax-check](#) in Ansible Lint Documentation.

The content parser tool handles syntax-check rule violations in the following manner:

- If **syntax-check** errors are found in the first execution of ansible-lint, content parser tool generates a list of files that contain the rule violations.
- If one or more **syntax-check** errors are found in the first execution of ansible-lint, the content parser tool runs ansible-lint again but excludes the files with syntax-check errors. After the scan is completed, the content parser tool generates a list of files that contain rule

violations. The list includes all files that caused syntax-check errors as well as other rule violations. The content parser tool excludes files with rule violations in all future scans, and the final training data set does not include data from the excluded files.

Procedure

Use one of the following methods to resolve ansible-lint rule violations:

- Run the content parser tool with the **--no-exclude** option. If any rule violations, including syntax-check errors, are found, the execution is aborted with an error and no training data set is created.
- Limit the set of rules that ansible-lint uses to scan your data with the **--profile** option. It is recommended that you fix the files with rule violations. However, if you do not want to modify the source files, you can limit the set of rules that ansible-lint uses to scan your data. To limit the set of rules that ansible-lint uses to scan your data, specify the **--profile** option with a predefined profile (for example, minimum, basic, moderate, safety, shared, or production profiles) or by using ansible-lint configuration files. For more information, see the [Ansible Lint documentation](#).
- Run the content parser tool by skipping ansible-lint preprocessing. You can run the content parser without ansible-lint preprocessing. The content parser tool generates a training data set without scanning for ansible-lint rule violations.

To run content parser tool without ansible-lint preprocessing, execute the following command:

```
$ ansible-content-parser source output -S
```

Where:

- **source**: Specifies the source of the training data set.
- **output**: Specifies the output of the training data set.
- **-S** or **--skip-ansible-lint**: Specifies to skip ansible-lint preprocessing while generating the training data set.

3.4.2.8. Merging multiple training data sets into a single file

For every execution, the content parser tool creates a training data set JSONL file named **ftdata.jsonl** that you upload to IBM watsonx Code Assistant for creating a custom model. If the content parser tool runs multiple times, multiple JSONL files are created. IBM watsonx Code Assistant supports a single JSONL file upload only; therefore, if you have multiple JSONL files, you must merge them into a single, concatenated file. You can also merge the multiple JSONL files that are generated in multiple subdirectories within a parent directory into a single file.

Procedure

1. Using the command prompt, go to the parent directory.
2. Run the following command to create a single, concatenated file:
find . -name ftdata.json | xargs cat > concatenated.json
3. Optional: Rename the concatenated file for easy identification.

You can now upload the merged JSONL file to IBM watsonx Code Assistant and create a custom model.

3.4.3. Create and deploy a custom model in IBM watsonx Code Assistant

After the content parser tool generates a custom model training data set, upload the JSONL file `ftdata.jsonl` to IBM watsonx Code Assistant and create a custom model for your organization.



IMPORTANT

IBM watsonx Code Assistant might take a few hours to create a custom model, depending on the size of your training data set. You must continue monitoring the IBM Tuning Studio for the status of custom model creation.

For information about how to create and deploy a custom model in IBM watsonx Code Assistant, see the [IBM watsonx Code Assistant documentation](#).

3.4.4. Configuring Red Hat Ansible Lightspeed to use custom models

After you create and deploy a custom model in IBM watsonx Code Assistant, you must configure Red Hat Ansible Lightspeed so that you can use the custom model for your organization.

You can specify one of the following configurations for using the custom model:

- **Enable access for all users in your organization**
You can configure the custom model as the default model for your organization. All users in your organization can use the custom model.
- **Enable access for select Ansible users in your organization**
Using the **model-override** setting, you can configure the custom model and make it available for select Ansible users only. For example, if you are using Red Hat Ansible Lightspeed both as an organization administrator and an end user, you can test the custom model for select Ansible users before making it available for all users in your organization.

3.4.4.1. Configuring the custom model for all Ansible users in your organization

You can configure the custom model as the default model for your organization, so that all users in your organization can use the custom model.

Procedure

1. Log in to the [Ansible Lightspeed with IBM watsonx Code Assistant Hybrid Cloud Console](#) as an organization administrator.
2. Specify the model ID of the custom model:
 - a. Click **Model Settings**.
 - b. Under **Model ID**, click **Add Model ID**. A screen to enter the **Model ID** is displayed.
 - c. Enter the **Model ID** of the custom model.
 - d. Optional: Click **Test model ID** to validate the model ID.
 - e. Click **Save**.

3.4.4.2. Configuring the custom model for select Ansible users in your organization

Using the **model-override** setting in Ansible Visual Studio (VS) Code, you can configure the custom model and make it available for select Ansible users only. For example, If you are using Red Hat Ansible Lightspeed as both an organization administrator and an end user, you can test the custom model for select Ansible users before making it available for all users in your organization.

Procedure

1. Log in to the VS Code application using your Red Hat account.

2. From the Activity bar, click the **Extensions** icon  .

3. From the Installed Extensions list, select **Ansible**.

4. From the **Ansible** extension page, click the **Settings** icon and select **Extension Settings**.

5. From the list of settings, select **Ansible Lightspeed**.

6. In the **Model ID Override** field, enter the model ID of the custom model.
Your settings are automatically saved in VS Code, and you can now use the custom model.

CHAPTER 4. LOGGING IN AND OUT OF THE ANSIBLE LIGHTSPEED PORTAL

You can access Ansible Lightspeed through the [Ansible Lightspeed portal](#). After you enter your Red Hat Single Sign-On (RH-SSO) account credentials, your account is authenticated and you are granted access. Your assigned user role is displayed on the login screen of the Ansible Lightspeed portal.

Table 4.1. User login scenarios

Scenario	Result
You are a RH-SSO user. NOTE: This is the typical scenario for accessing Red Hat Ansible Lightspeed as an Ansible user.	You are routed to the Red Hat Ansible Lightspeed paid commercial offering.
You are a RH-SSO user, but your organization administrator has not configured Red Hat Ansible Lightspeed to connect with IBM watsonx Code Assistant.	You are routed to the Red Hat Ansible Lightspeed paid commercial offering with a message that your organization administrator has not configured a model for your organization.

4.1. LOGGING IN TO THE ANSIBLE LIGHTSPEED PORTAL

Procedure


1. Go to the [Ansible Lightspeed portal login page](#).
2. Click **Log in** → **Log in with Red Hat**
3. Enter your Red Hat account username and password.

On successful authentication, the login screen is displayed along with your username and your assigned user role.

4.2. LOGGING OUT OF THE ANSIBLE LIGHTSPEED PORTAL

To log out of the Ansible Lightspeed Service, you must log out of both the Ansible Lightspeed VS Code extension and the Ansible Lightspeed portal.

Procedure

- Log out of the Ansible Lightspeed VS Code extension:
 - Click the **Person** icon . You will see a list of accounts that VS Code is logged into.
 - Select **Ansible Lightspeed** → **Sign Out**.
- Log out of the Ansible Lightspeed portal:
 - Navigate to the [Ansible Lightspeed portal login page](#).
 - Click **Log out**.

CHAPTER 5. INSTALLING AND CONFIGURING THE ANSIBLE VS CODE EXTENSION

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant is integrated with the Ansible Visual Studio (VS) Code extension in VS Code. The Ansible VS Code extension, with Red Hat Ansible Lightspeed features enabled, automatically collects recommendations, usage telemetry, and Ansible YAML file state through automated events.

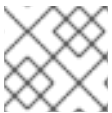
To access Red Hat Ansible Lightspeed, all Ansible users must install and configure the Ansible VS Code extension in their VS Code. The Ansible VS Code extension uses the Ansible-specific IBM watsonx Granite model configured in the Red Hat Ansible Lightspeed administrator portal as the default mode for all users in your organization.

You can also use a custom, fine-tuned model if your organization administrator has created a custom model and has shared the model ID with you separately. Use the **model-override** setting in the Ansible VS Code extension to override the default model, and use the custom model instead. Using a custom model enables you to improve the code recommendation experience and tune the model to your organizational automation patterns. For example, if you are using Red Hat Ansible Lightspeed both as an organization administrator and a user, you can test the custom model for select Ansible users before making it available for all users in your organization. For more information, see [Configuring custom models](#).

5.1. INSTALLING THE ANSIBLE VS CODE EXTENSION

Prerequisites

- VS Code version 1.70.1 or later.



NOTE

You can also install VScode derivatives, such as VScode Insider or VS Codium.

Procedure

1. Open the VS Code application.
2. From the navigation menu, click the **Extensions** icon.
3. In the **Search** field, enter **Ansible**.
4. Select **Ansible** to choose the Ansible language support extension published by Red Hat.
5. Click **Install**.
6. After installation is complete, verify your VSCode installation:
 - a. Create a new YAML file using the **.yml** or **.yaml** file extension.
 - b. From the **Status** toolbar, click the language indicator and select **Ansible** to associate the Ansible language type with the new YAML file.
 - c. Start writing a test playbook. Contextual aids are displayed as you start creating your content.

5.2. CONFIGURING THE ANSIBLE VS CODE EXTENSION

You can configure the Ansible VS Code extension to enable Red Hat Ansible Lightspeed and specify its portal URL and IBM watsonx Code Assistant model ID.

Prerequisites

- Your organization administrator has configured an IBM watsonx Code Assistant model for your organization.

Procedure

1. Open the VS Code application.

2. From the Activity bar, click the **Extensions** icon  .

3. From the Installed Extensions list, select **Ansible**.

4. From the **Ansible** extension page, click the **Settings** icon and select **Extension Settings**.

5. Select **Ansible Lightspeed** settings, and specify the following information:

- a. Select the **Enable Ansible Lightspeed** checkbox.

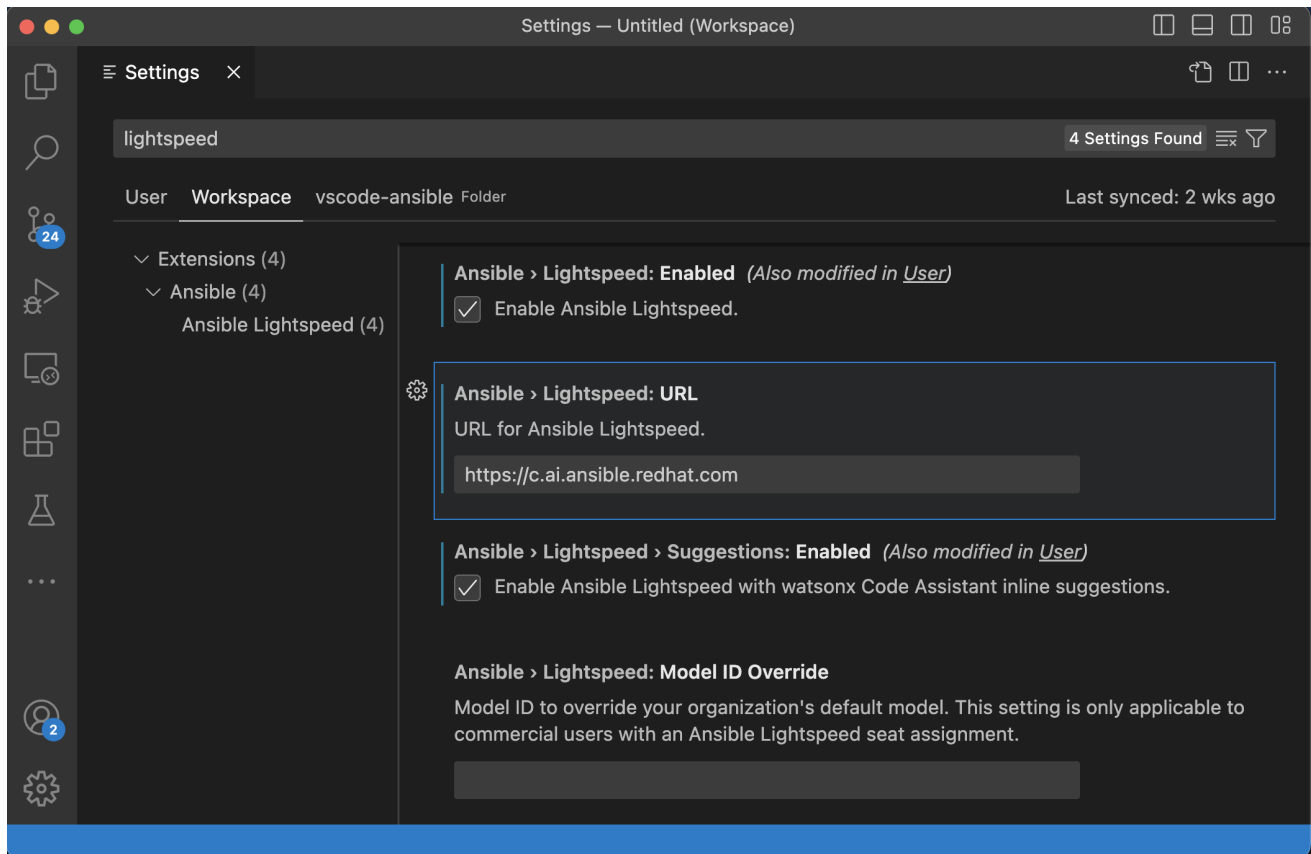
- b. In the **URL for Ansible Lightspeed** field, verify that you have the following URL:
<https://c.ai.ansible.redhat.com/>.

- c. Select the **Enable Ansible Lightspeed with watsonx Code Assistant inline suggestions** checkbox.

6. Optional: If you want to use the custom model instead of the default model, in the **Model ID Override** field, enter the custom model ID. The **model-override** setting enables you to override the default model and use the custom model, after your organization administrator has created a custom model and has shared the model ID with you separately.
Your settings are automatically saved in VS Code.

The following illustration displays the configured settings for the Ansible VS Code extension:

Figure 5.1. Configured settings for the Ansible VS Code extension



Additional resources

- [Troubleshooting Ansible Visual Studio Code extension errors](#)

5.3. LOGGING IN TO ANSIBLE LIGHTSPEED THROUGH THE ANSIBLE VS CODE EXTENSION

After installing and configuring the VS Code extension, you can log in to the Ansible Lightspeed service.

Procedure

1. Open the VS Code application.
2. Use one of the following ways to connect to the Ansible Lightspeed service.
 - Using the **Connect** button:
 - a. From the navigation menu, click the **Ansible** icon.
 - b. Under Ansible Lightspeed Login, click **Connect**.
 - Using the **Accounts** button:
 - a. From the navigation menu, click **Accounts icon > Sign in with Ansible Lightspeed**

**NOTE**

This option is displayed when the VS Code extension is in an active state. The extension is activated after you open the Ansible side panel or after you open an Ansible file in the VS Code editor. If you do not see this option, use the **Connect** button to link to the Ansible Lightspeed service.

3. When prompted, click **Allow** to sign in.
4. In the **Authorize Ansible Lightspeed for VS Code** window, click **Authorize**.
5. In the **Do you want Code to open the external website?** window, click **Open**. The [Ansible Lightspeed portal login page](#) is displayed.
6. Click **Log in → Log in with Red Hat**
7. Enter your Red Hat account username and password.
On successful authentication, the login screen is displayed along with your username and your assigned user role. The VS code extension is now connected with Ansible Lightspeed service.

CHAPTER 6. REQUESTING TASK RECOMMENDATIONS

Red Hat Ansible Lightspeed is integrated into Visual Studio (VS) Code through the Ansible VS Code extension. You can request code recommendations for your task intent by using Ansible VS Code extension.

6.1. OVERVIEW

You can perform the following tasks from the Ansible VS Code extension:

- **Create single task or multitask requests by using natural language prompts**
 - Create a single task prompt
Write a description of your task in the **- name:** key of a new task line in your Ansible file. For example, to automate a task of installing PostgreSQL server, you can enter the prompt **- name: Install postgresql-server**.
 - Create a multitask prompt
Place your cursor on a new line in your Ansible YAML file at the correct indentation, and start your prompt with a Pound key (#).

Write the descriptions of your tasks, separating each prompt by using Ampersand symbols (&). For example, to automate a multitask of installing PostgreSQL server and running the initial PostgreSQL setup command, you can enter the prompt **# Install postgresql-server & run postgresql-setup command**.

The Ansible Lightspeed service reads the text, interacts with the IBM watsonx Code Assistant model, and generates Ansible task recommendations based on your natural language prompt.



NOTE

Currently, Red Hat Ansible Lightspeed supports user prompts in English language only. However, there could be instances where the training data that was used to train the IBM watsonx Code Assistant models included non-English language. In such scenarios, the model can generate code recommendations for prompts made in the same non-English language, but the generated code recommendations might or might not be accurate.

- **View the content source matching results**
For each generated code recommendation, Red Hat Ansible Lightspeed lists content source matches, including details such as potential source, content author, and relevant licenses. You can use this data to gain insight into potential training data sources used to generate the code recommendations.
- **Provide feedback on the Ansible Lightspeed service**
The Ansible Lightspeed service learns your organizational patterns and improves the code recommendation experience over time. You can provide feedback on whether the generated code recommendations were suitable for your task intent. This feedback enables Red Hat Ansible Lightspeed with IBM watsonx Code Assistant to improve on the quality of its suggestions.

6.1.1. Best practices to improve the recommended guidance

Follow these guidelines to improve the likelihood of a quality code recommendation.

- Ensure that your YAML file is properly formatted. See the [Ansible YAML syntax guidelines](#) for details.
- Avoid context switching within a single playbook file.
The Ansible Lightspeed service attempts to correlate earlier tasks to the active recommendation, and the entire contents of the file before the cursor position are used as context by the model. If the earlier task is not relevant to your prompt, VS code provides inline suggestions instead of code recommendations.
- Reword your natural language prompts to get code recommendations that match your task intent.
If you get a recommendation that does not align with the intent of your task name, then rewording your prompt to provide more information about what is desired can lead to improved results.
- Use descriptive prompts and provide additional content to improve the code recommendations. Red Hat Ansible Lightspeed reads the full Ansible YAML file when generating a code recommendation. Using descriptive prompts and having additional YAML file content related to the desired task improves the code recommendation. For example, you can add the previous Ansible tasks and appropriate playbook and variable names to improve the code recommendations.

6.2. REQUESTING CODE RECOMMENDATIONS FOR A SINGLE TASK

You can request code recommendations for a single task by entering natural language prompts in Ansible VS Code extension. For example, to automate a task of installing a PostgreSQL server, you can enter the prompt - **name: Install postgresql-server**. The Ansible Lightspeed service reads the text, interacts with the IBM watsonx Code Assistant model, and generates the code recommendations.

Prerequisites

- You are part of an organization that has a trial or paid subscription to both Ansible Automation Platform and IBM watsonx Code Assistant.
- You have [installed and configured the Ansible VS Code extension](#) .

Procedure

1. Log in to VS Code with your Red Hat account.
2. Create a new YAML file or use an existing YAML file:
 - Create a YAML file:
 - i. Select **File** → **New Text File**.
 - ii. From the lower right of the screen, click **Plain Text**, and in the language mode, select **Ansible**.
 - iii. Save the file as a YAML file format extension (**.yml** or **.yaml**).
 - Use an existing YAML file:

- i. On the bottom right of the screen, click the existing language mode, and in the language mode settings, select **Ansible**.



NOTE

If you do not see the language mode section in your VS Code editor, from the Command Palette, select **Configure Language Mode** → **Ansible**.

3. Verify that you see an entry for **Lightspeed** on the status bar at the lower right of VS Code. If **Ansible** is already selected as the desired language but the **Lightspeed** entry is not displayed, re-select **Ansible** as the language mode. The following illustration shows **Lightspeed** and **Ansible** entries on the VS Code status bar.

Figure 6.1. Ansible and Lightspeed set as selected language mode


```

1 ---
2 - name: Configure Database servers
3   hosts: databases
4   become: true
5
6   tasks:
7     - name: Install postgresql-server
8       ansible.builtin.package:
          name: postgresql-server
          state: present
  
```

quest #11 Ln 8, Col 5 Spaces: 2 UTF-8 LF Ansible (↓ [EE] 2.15.3 Lightspeed Python 3.11.4

4. Optional: If you see an error message about missing Ansible lint, you can install the missing module or disable it. Perform any one of the following tasks:
 - Install Ansible lint: For installation information, see the [Installing](#) section of the Ansible Lint documentation.
 - Disable Ansible lint:



- i. From the Activity bar, click the **Extensions** icon  .
 - ii. From the **Installed** extensions list, select **Ansible**.
 - iii. From the **Ansible** extension page, click the **Settings** icon and select **Extension Settings**.
 - iv. Clear the **Ansible > Validation > Lint: Enabled** checkbox.
5. Create a playbook or use an existing playbook.
For more information, see [Creating playbooks](#) in the Ansible Automation Platform Creator Guide.
 6. In the playbook, provide the following information to request code recommendations for a single task:
 - i. Add a new Ansible task by starting a new line with **- name:** at the correct indentation.
 - ii. Add a detailed natural language prompt in the task description after **- name:** on the same line. For example, you can specify the following single task prompt: **- name: Install postgresql-server**
 - iii. Press **Enter** directly after the task description. Keep the cursor at the same location in your file, and wait for the code recommendation results to populate.
The Ansible Lightspeed service is engaged, and it starts generating code recommendations for a single task.



IMPORTANT

Ansible Lightspeed service takes around 5 seconds per task to populate the code recommendations. If you are using a multitask prompt, the Ansible Lightspeed service takes a bit longer (number of tasks times 5 seconds) to populate the results. Do not move your cursor or press any key while the code recommendation is being generated. If you change the cursor location or press any key, Ansible VS Code extension cancels the request and the Ansible Lightspeed service does not process your request.

When the Ansible Lightspeed service is engaged, a **Lightspeed** processing status indicator is displayed in the lower right of the screen to denote that your code recommendation is being generated.



7. View your code recommendations and ensure that the recommendations match your task intent. The following illustration shows the code recommendations generated by the Ansible Lightspeed service for the single task **Install postgresql-server**:

```

- name: Configure Database servers
  hosts: databases
  become: true

  tasks:
    - name: Install postgresql-server
      ansible.builtin.package:
        name: postgresql-server
        state: present

```

8. Accept or reject the code recommendations:

- To accept a code recommendation, press **Tab**.
- To reject a code recommendation, press **Esc**.



NOTE

If you reject a recommendation, you can modify the prompt and review the generated code recommendations once again to match your task intent.

9. On the **ANSIBLE: LIGHTSPEED TRAINING MATCHES** tab, view the content source matching results.

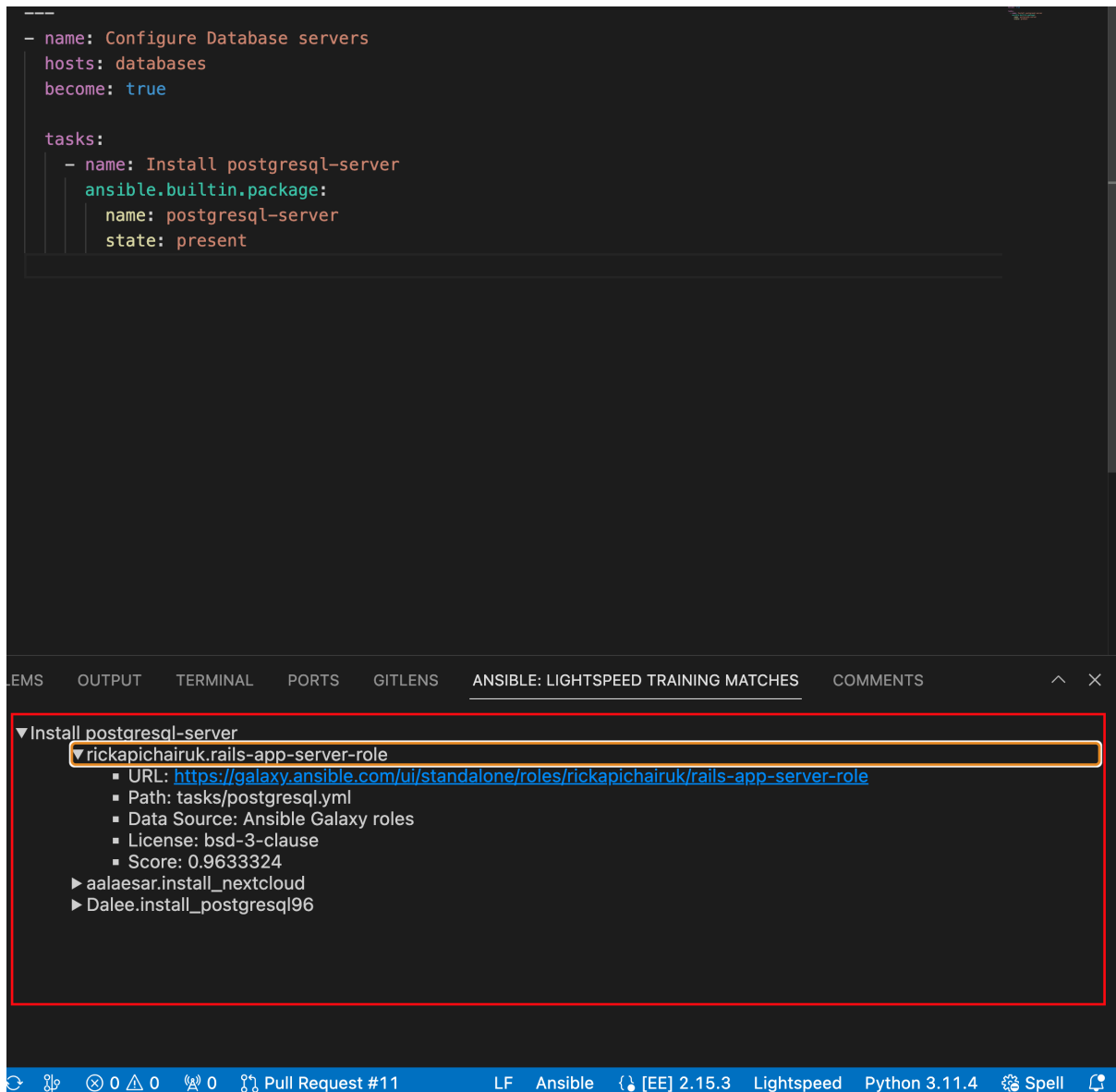
The following illustration shows the training matches found in existing Ansible Galaxy content for the task prompt **Install postgresql-server**:

```

- name: Configure Database servers
  hosts: databases
  become: true

  tasks:
    - name: Install postgresql-server
      ansible.builtin.package:
        name: postgresql-server
        state: present

```



The screenshot shows the VS Code interface with the 'ANSIBLE: LIGHTSPEED TRAINING MATCHES' panel open. A red box highlights a recommendation for the task 'Install postgresql-server'. The recommendation details are as follows:

- Task: Install postgresql-server
- Role: rickapichairuk.rails-app-server-role
- URL: <https://galaxy.ansible.com/ui/standalone/roles/rickapichairuk/rails-app-server-role>
- Path: tasks/postgresql.yml
- Data Source: Ansible Galaxy roles
- License: bsd-3-clause
- Score: 0.9633324
- Other roles: aalaesar.install_nextcloud, Dalee.install_postgresql96

10. Click **Save** to save the code recommendation changes in your Ansible YAML file.

Additional resources

- [Troubleshooting Ansible Visual Studio Code extension errors](#)
- [Troubleshooting Ansible code bot errors](#)

6.3. REQUESTING CODE RECOMMENDATIONS FOR MULTIPLE TASKS

You can request multitask code recommendations by entering a sequence of natural language task prompts in Ansible VS Code extension. In a YAML file, start a comment by using a Pound key (#), and separate each prompt by using Ampersand (&) symbols.

For example, to automate a multitask of installing PostgreSQL server and running the initial PostgreSQL setup command, you can enter the prompt **# Install postgresql-server & run postgresql-setup command**. The Ansible Lightspeed service reads the text, interacts with the IBM watsonx Code Assistant models, and generates the code recommendations.

Prerequisites

- You are part of an organization that has a trial or paid subscription to both Ansible Automation Platform and IBM watsonx Code Assistant.
- You have [installed and configured the Ansible VS Code extension](#) .

Procedure

1. Log in to VS Code with your Red Hat account.
2. Create a new YAML file or use an existing YAML file.
 - Create a YAML file:
 - i. Select **File** → **New Text File**.
 - ii. From the lower right of the screen, click **Plain Text**, and in the language mode, select **Ansible**.
 - iii. Save the file as a YAML file format extension (**.yml** or **.yaml**).
 - Use an existing YAML file:
 - i. On the bottom right of the screen, click the existing language mode, and in the language mode settings, select **Ansible**.



NOTE

If you do not see the language mode section in your VS Code editor, from the Command Palette, select **Configure Language Mode** → **Ansible**.

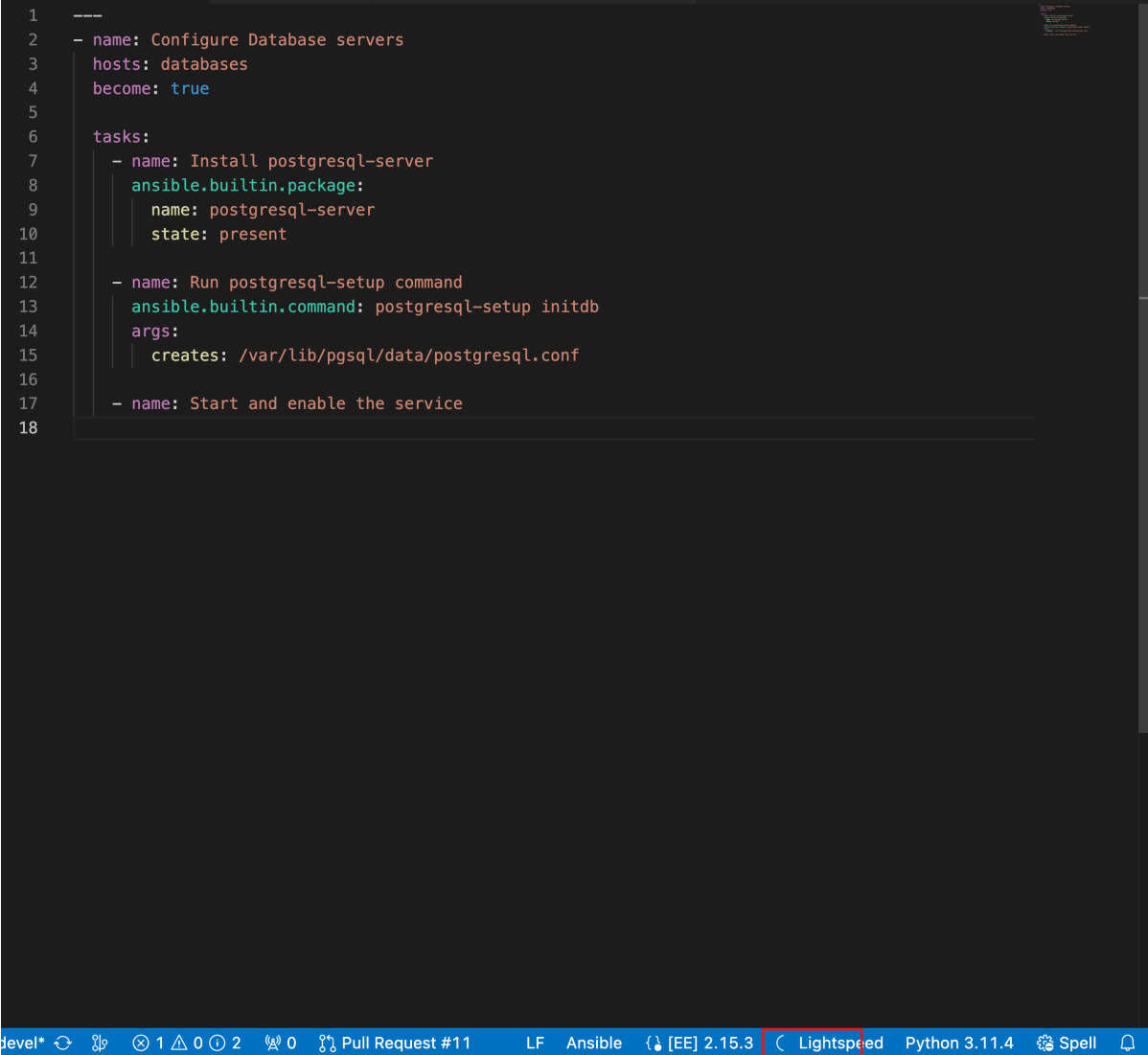
3. Verify that you see an entry for **Lightspeed** on the status bar at the lower right of VS Code. If **Ansible** is already selected as the desired language but the **Lightspeed** entry is not displayed, re-select **Ansible** as the language mode. The following illustration shows **Lightspeed** entry on the VS Code status bar.


Figure 6.2. Ansible and Lightspeed set as selected language mode

```

1  ---
2  - name: Configure Database servers
3    hosts: databases
4    become: true
5
6    tasks:
7      - name: Install postgresql-server
8        ansible.builtin.package:
9          name: postgresql-server
10         state: present
11
12      - name: Run postgresql-setup command
13        ansible.builtin.command: postgresql-setup initdb
14        args:
15          creates: /var/lib/pgsql/data/postgresql.conf
16
17      - name: Start and enable the service
18

```



4. Optional: If you see an error message about missing Ansible lint, you can install the missing module or disable it. Perform any one of the following tasks:
 - Install Ansible lint: For installation information, see the [Installing](#) section of the Ansible Lint documentation.
 - Disable Ansible lint:
 - i. From the Activity bar, click the **Extensions** icon .
 - ii. From the **Installed** extensions list, select **Ansible**.
 - iii. From the **Ansible** extension page, click the **Settings** icon and select **Extension Settings**.
 - iv. Clear the **Ansible > Validation > Lint: Enabled** checkbox.
5. Create a playbook or use an existing playbook.
For more information, see [Creating playbooks](#) in the Ansible Automation Platform Creator Guide.
6. In the playbook, provide the following information to request multitask code recommendations:

- i. Start a new YAML file comment by entering a Pound key (#) at the correct indentation.
- ii. Add a detailed natural language prompt in a sequence, separating each task by using an Ampersand (&) symbol. For example, to automate the multitask of installing PostgreSQL server and running the PostgreSQL setup command, enter the following natural language prompt **# Install postgresql-server & run postgresql-setup command**.
- iii. Press **Enter** directly after the task description. Keep the cursor at the same location in your file, and wait for the code recommendation results to populate.
The Ansible Lightspeed service is engaged, and it starts generating code recommendations for multiple tasks.



IMPORTANT

Ansible Lightspeed service takes around 5 seconds per task to populate the code recommendations. If you are using a multitask prompt, the Ansible Lightspeed service takes a bit longer (number of tasks times 5 seconds) to populate the results. Do not move your cursor or press any key while the code recommendation is being generated. If you change the cursor location or press any key, Ansible VS Code extension cancels the request and the Ansible Lightspeed service does not process your request.

When the Ansible Lightspeed service is engaged, a **Lightspeed** processing status indicator is displayed in the lower right of the screen to denote that your code recommendation is being generated.



7. Optional: If multitask code recommendations are not being generated, log out of VS Code and log in again using your Red Hat account.
8. View your code recommendations and ensure that the recommendations match your task intent. The following illustration shows the code recommendations generated by the Ansible Lightspeed service for the multitask prompt **Install postgresql-server & run postgresql-setup command** :

```

---
- name: Configure Database servers
  hosts: databases
  become: true

  tasks:
    # Install postgresql-server & run postgresql-setup command
    - name: Install postgresql-server
      ansible.builtin.package:
        name: postgresql-server
        state: present

    - name: Run postgresql-setup command
      ansible.builtin.command: postgresql-setup initdb
      args:
        creates: /var/lib/pgsql/data/postgresql.conf

```

9. Accept or reject the code recommendations:

- To accept a code recommendation, press **Tab**.
- To reject a code recommendation, press **Esc**.



NOTE

If you reject a recommendation, you can modify the prompt and review the generated code recommendations once again to match your task intent.

10. On the **ANSIBLE: LIGHTSPEED TRAINING MATCHES** tab, view the content source matching results.

The following illustration shows the training matches found in existing Ansible Galaxy content for the task prompt multitask prompt **Install postgresql-server & run postgresql-setup command**:

The screenshot shows a code editor with an Ansible playbook snippet. A red box highlights the following tasks:

```

- name: Configure Database servers
  hosts: databases
  become: true

  tasks:
    # Install postgresql-server & run postgresql-setup command
    - name: Install postgresql-server
      ansible.builtin.package:
        name: postgresql-server
        state: present

    - name: Run postgresql-setup command
      ansible.builtin.command: postgresql-setup initdb
      args:
        creates: /var/lib/pgsql/data/postgresql.conf
  
```

Below the code editor, a panel titled "ANSIBLE: LIGHTSPEED TRAINING MATCHES" is visible. A red box highlights the following matches:

- ▼ Install postgresql-server
 - ▶ rickapichairuk.rails-app-server-role
 - ▶ aalaesar.install_nextcloud
 - ▶ Dalee.install_postgresql96
- ▼ Run postgresql-setup command
 - ▶ elan.opencast_postgresql
 - ▶ gurvanjossec.awx_install
 - ▶ gurvanjossec.awx_install

The bottom status bar shows: Pull Request #11, LF, Ansible, [EE] 2.15.3, Lightspeed, Python 3.11.4, 2 Spell.

11. Click **Save** to save the code recommendation changes in your Ansible YAML file.

Additional resources

- [Troubleshooting Ansible Visual Studio Code extension errors](#)
- [Troubleshooting Ansible code bot errors](#)

6.4. VIEWING ANSIBLE LIGHTSPEED TRAINING MATCHES

The Red Hat Ansible Lightspeed with IBM watsonx Code Assistant machine learning model is trained on the following content: * Existing public or private Git repositories * Content from Ansible Galaxy

Owing to IBM watsonx Code Assistant's generative AI technology, as well as the types of Ansible content that were used to train the model, it is not possible to identify the specific set of training data that contributed to the generated code recommendations. However, Ansible Lightspeed provides a capability that helps you to understand the possible origins of generated code recommendations.

For each generated code recommendation, Red Hat Ansible Lightspeed lists the content source matches, including details such as potential source, content author, and relevant licenses. You can use this data to gain insight into potential training data sources used to generate the code recommendations.

After you enter a natural language prompt in VS Code and see the generated code recommendations, you can view the content source matches on the **ANSIBLE: LIGHTSPEED TRAINING MATCHES** tab.

For example, the following illustration shows the training matches for the multitask recommendation **Install postgresql-server & run postgresql-setup command**

Figure 6.3. Training matches for a multitask recommendation

The screenshot shows a VS Code editor with an Ansible playbook and a panel displaying training matches. The code in the editor is as follows:

```

- name: Configure Database servers
  hosts: databases
  become: true

  tasks:
    # Install postgresql-server & run postgresql-setup command
    - name: Install postgresql-server
      ansible.builtin.package:
        name: postgresql-server
        state: present

    - name: Run postgresql-setup command
      ansible.builtin.command: postgresql-setup initdb
      args:
        creates: /var/lib/pgsql/data/postgresql.conf
  
```

The panel below the editor, titled "ANSIBLE: LIGHTSPEED TRAINING MATCHES", shows the following matches:

- ▼ Install postgresql-server
 - ▶ rickapichairuk.rails-app-server-role
 - ▶ aalaesar.install_nextcloud
 - ▶ Dalee.install_postgresql96
- ▼ Run postgresql-setup command
 - ▶ elan.opencast_postgresql
 - ▶ gurvanjossec.awx_install
 - ▶ gurvanjossec.awx_install

The bottom status bar of VS Code shows: Pull Request #11, LF, Ansible, [EE] 2.15.3, Lightspeed, Python 3.11.4, 2 Spell.

This capability enables you to find out the open source license terms that are associated with related training data. However, it is unlikely that either the training data used in fine-tuning the code or the output recommendations themselves are protected by copyright, or that the output reproduces training data that is controlled by copyright licensing terms.



NOTE

Red Hat does not claim any copyright or other intellectual property rights in the suggestions generated by Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

6.5. PROVIDING FEEDBACK ON THE ANSIBLE LIGHTSPEED SERVICE

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant is designed to be improved through feedback on the quality of its suggestions. The technical details of user experiences with Red Hat Ansible Lightspeed are useful in informing further improvements.

You can submit feedback through the following channels:

- From the Ansible VS Code extension: Use this method to provide feedback about the quality of the suggested code recommendations.



IMPORTANT

Red Hat Support cannot assist with the suggestion quality reports. Content quality issues are routed to IBM for resolution.

- From the [Red Hat customer portal](#): Use this method to log bug reports and service disruption incidents, and feature requests.



NOTE

On the login screen of the [Ansible Lightspeed Portal](#), there is a **Chat** link that redirects you to a Matrix channel. Use the Matrix channel to ask questions pertaining to your Ansible Lightspeed experience and request help to troubleshoot your issues. However, the Matrix channel is not an official Support channel, and issues raised in the Matrix chat would not be tracked through Red Hat Service Level Agreement (SLA). To raise a bug or a feature request, contact [Red Hat Support](#) and open a support ticket.

Prerequisites

- You are part of an organization that has a trial or paid subscription to both Ansible Automation Platform and IBM watsonx Code Assistant.

Procedure

1. Open Visual Studio Code.
2. Click the **Lightspeed** entry in your status bar to see options.
3. In the **Tell us why** field, provide your feedback. Here, provide feedback about what results you were expecting to receive, compared to what results were generated and the training match.
4. Select the issue type: **Bug report**, **Feature request**, or **Suggestion feedback**.



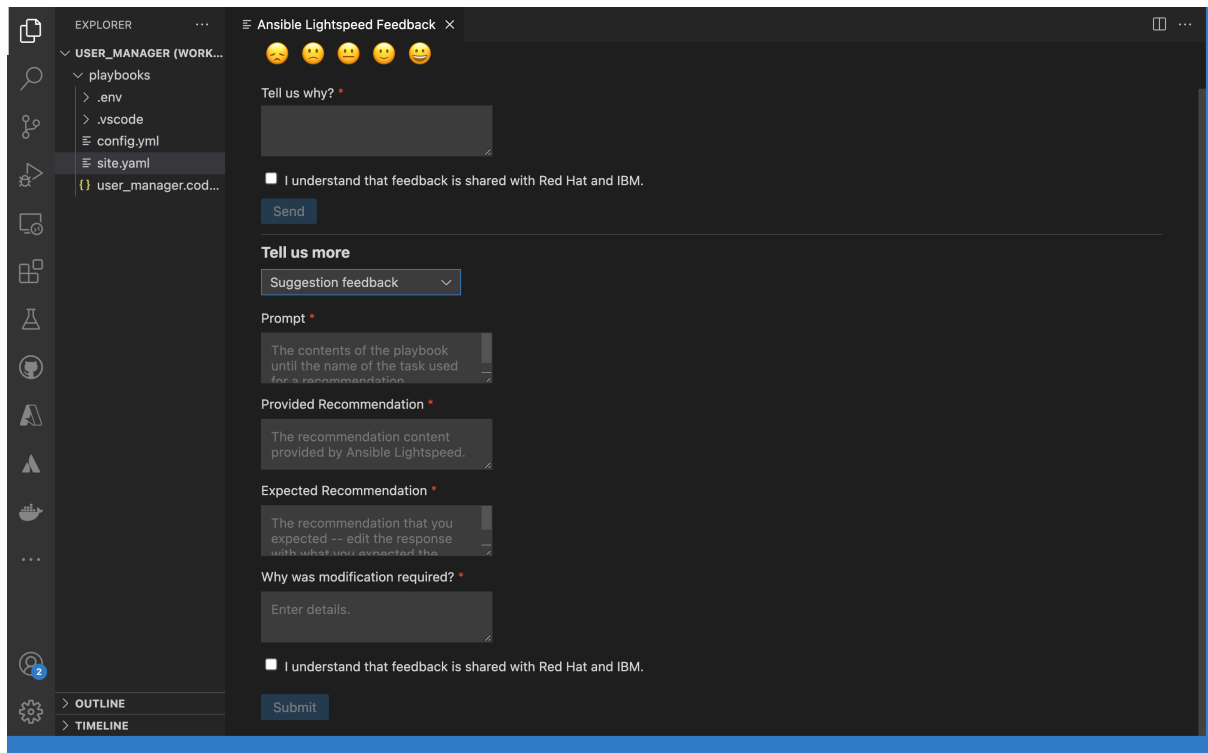
NOTE

To raise a bug or feature request, contact [Red Hat Support](#) and open a support ticket. Bug features and feature requests made through Ansible Lightspeed feedback are not tracked through the Red Hat Service Level Agreement (SLA).

5. Select the **I understand that feedback is shared with Red Hat and IBM** checkbox.
6. Click **Send**.

The following image shows an example of providing suggestion feedback:

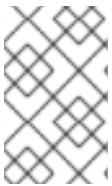
Figure 6.4. Providing feedback on Ansible Lightspeed



CHAPTER 7. GENERATING PLAYBOOKS AND VIEWING PLAYBOOK EXPLANATIONS

Using the Ansible VS Code extension, you can create Ansible playbooks using a natural language interface in English. Red Hat Ansible Lightspeed with IBM watsonx Code Assistant reads the natural language prompts and generates an entire playbook recommendation based on your intent. You can also view the explanations for new or existing playbooks. The playbook explanations describe what the playbook does and contextualize its impact.

These capabilities enable Ansible developers to use natural language prompts to create new Ansible playbooks quickly and more efficiently and also get an explanation for existing Ansible playbook, thereby reducing the overall onboarding learning period. For information about Ansible playbooks, see the [Getting started with Ansible Playbooks](#) guide.



NOTE

You can generate playbooks and view playbook explanations when connecting to the Red Hat Ansible Lightspeed cloud service. These features are not yet available with Red Hat Ansible Lightspeed on-premise deployments.

7.1. BEST PRACTICES TO GENERATE PLAYBOOKS

Follow these guidelines for the highest quality of a playbook recommendation.

- Ensure that the goal statements directly specify what the playbook must do. Your statement should start with the goal of the playbook, for example, **Apply security patches to RHEL9**. Avoid starting statements with **Create a playbook that**, **Please prepare a playbook that**, or **I need help with**.
- Ensure that the goal statement does not contain new lines.
- Ensure that the goal statement is not more than one sentence. You might have to repeat the details in the goal statement to produce the best results. It is recommended that you use the generated outline as feedback about whether your goal statement might benefit from more or less details, and then modify the goal statement as necessary.
- Ensure the following when you edit the outline:
 - Do not restate the goal of the playbook.
 - Verify that the steps considered capture the key steps in the playbook. The steps need not reflect each and every task that is expected in the playbook.
 - Keep the step description in one sentence without adding new lines to the outline.

7.2. GENERATING ANSIBLE PLAYBOOKS

You can use the natural language interface in the Ansible VS Code extension to generate an entire Ansible playbook.

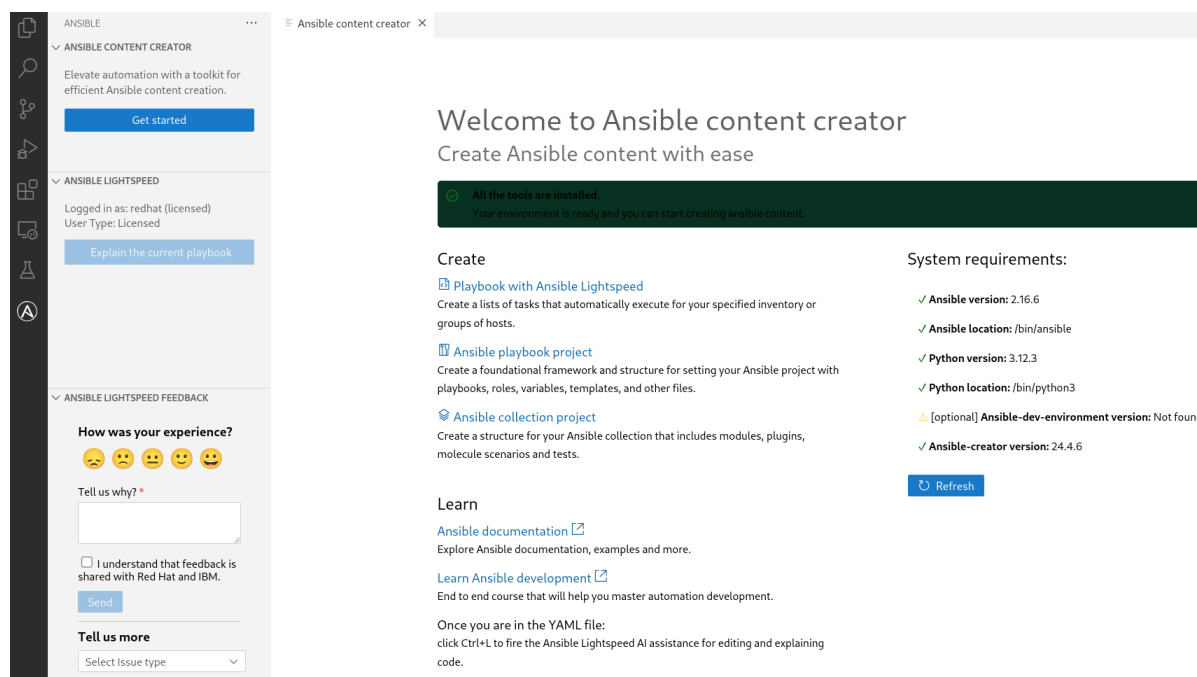
Prerequisites

- You are part of an organization that has a trial or paid subscription to Red Hat Ansible Automation Platform and IBM watsonx Code Assistant both.
- You have installed and configured the Ansible VS Code extension.

Procedure

1. Log in to VS Code with your Red Hat account.
2. From the **Activity** bar, click the **Ansible** icon.
3. Under **Ansible Creator**, click **Get started**. The **Ansible Content Creator** page is displayed. The following illustration displays the **Ansible Content Creator** page:

Figure 7.1. Settings to create Ansible playbooks



4. Select the **Playbook with Ansible Lightspeed** tile. The **Create a playbook** page is displayed.
5. In the **What do you want the playbook to accomplish?** field, enter the prompts to create a playbook and click **Analyze**.
After a few seconds, the recommended steps for your playbook intent are displayed in the **Review the suggested steps for your playbook and modify as needed** field.
6. Perform one of the following tasks:
 - If the steps match your intent: Click **Generate Playbook**.
 - If any modifications are required: Click the editor and update the tasks or steps to suit your intent.
 - If the task suggestions do not match your intent: Click **Back** to change the original prompt and start over.
 - If you want to restore the original task suggestions: Click **Reset** and proceed to the next step.
7. After you verify the steps, click **Generate playbook**.

It takes a few seconds for the playbook to generate, and **The following playbook was generated for you** field displays the newly generated playbook.

8. Click **Open editor**. The generated playbook opens as an untitled YAML file in the VS Code editor.
9. Save the untitled YAML file.

7.3. VIEWING PLAYBOOK EXPLANATIONS

You can request explanations for a newly created playbook as well as an existing Ansible playbook.

Prerequisites

- You are part of an organization that has a trial or paid subscription to Red Hat Ansible Automation Platform and IBM watsonx Code Assistant both.
- You have installed and configured the Ansible VS Code extension.
- You have opened the playbook whose explanation you want to view.

Procedure

1. Log in to VS Code with your Red Hat account.
2. Open an Ansible playbook YAML file in VS Code.
3. Use one of the following methods to view the playbook explanation:
 - **From an active playbook YAML file**
 - a. Place your cursor anywhere within the playbook file.
 - b. Right-click > Select **Explain the playbook with Ansible Lightspeed**
 - **From the Ansible panel**
 - a. From the navigation menu, click the **Ansible** icon.
 - b. Select **Explain the current playbook**
The playbook explanation is displayed on the right panel of the VS Code screen.

The following illustration shows an example of a playbook explanation:

Figure 7.2. Example of a playbook explanation

The screenshot shows the Ansible Creator interface with a central editor displaying a playbook and a right-hand pane showing its explanation.

Playbook Content:

```

2  - name: Update web servers
3  hosts: webservers
4  remote_user: root
5
6
7  - name: Ensure apache is at the latest version
8  ansible.builtin.yum:
9    name: httpd
10   state: latest
11
12 - name: Write the apache config file
13 ansible.builtin.template:
14   src: /srv/httpd.j2
15   dest: /etc/httpd.conf
16
17 - name: Update db servers
18 hosts: databases
19 remote_user: root
20
21
22 - name: Ensure postgresql is at the latest version
23 ansible.builtin.yum:
24   name: postgresql
25   state: latest
26
27 - name: Ensure that postgresql is started
28 ansible.builtin.service:
29   name: postgresql
30   state: started

```

Explanation Pane Content:

Summary:
This playbook updates web servers and database servers by installing the latest versions of Apache and PostgreSQL, respectively.

Pre-requisites:
The playbook assumes that Ansible is installed on the Ansible learner's system and that the learner has a basic understanding of Ansible concepts such as inventory, playbooks, and tasks.

Task Explanations:

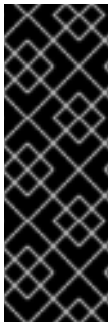
1. The first task in the playbook ensures that the latest version of Apache is installed on all web servers by using the yum module. The yum module ensures that the specified package is installed or updated on the system.
2. The second task in the playbook writes the Apache configuration file using the template module. The template module allows you to generate a configuration file from a template file, which can contain variables and other dynamic elements.
3. The third task in the playbook ensures that the latest version of PostgreSQL is installed on all database servers using the yum module.
4. The fourth task in the playbook starts the PostgreSQL service on all database servers using the service module. The service module allows you to manage services on a system, such as starting, stopping, restarting, and checking the status of a service.

CHAPTER 8. INSTALLING AND CONFIGURING THE ANSIBLE CODE BOT

The Ansible code bot scans existing content collections, roles, and playbooks hosted in GitHub repositories, and proactively creates pull requests whenever best practices or quality improvement recommendations are available.

Ansible code bot scans your code repositories to recommend code quality improvements. It promotes Ansible best practices while avoiding common errors that can lead to bugs or make code harder to maintain. The bot automatically submits pull requests to the repository, which proactively alerts the repository owner to a recommended change to their content. You can configure Ansible code bot to scan your existing Git repositories (both public and private). Your organization must have an active subscription to Red Hat Ansible Automation Platform to use the Ansible code bot.

After you install the Ansible code bot, you can access the Ansible code bot dashboard that displays all your repositories that have the bot installed along with their scan status. From the dashboard, you can start a manual scan, view the scan history, and view the repository. From GitHub, you can configure a schedule to scan your repository at regular intervals, and add or remove a repository from being scanned. For more information, see [Managing repository scans](#).



IMPORTANT

Ansible code bot is supported on the following GitHub versions:

- GitHub.com
 - GitHub Enterprise Cloud
- Ansible code bot is not supported on GitHub Enterprise Server. For more information, see [GitHub's plans](#) in the GitHub documentation.

The following examples are code recommendations that the Ansible code bot can suggest:

- Available alternatives for deprecated legacy syntax or implementation patterns
- Module version changes and updates, such as:
 - Adding any new required parameters
 - Flagging deprecated parameters
 - Removing unused parameters
- Applying YAML best practices
- Adding comment blocks
- Fixing casing issues in name fields

8.1. INSTALLING THE ANSIBLE CODE BOT

Install the Ansible code bot to get code recommendations for your repositories, and then log in to the Ansible code bot dashboard to monitor and manage your repository scans.

Procedure

1. Log in to GitHub by using the account associated with your organization.
2. Go to the [Ansible code bot](#) GitHub app.
3. Select the Ansible repositories that you want the app to access:
 - **All repositories:** Provides access to read the metadata of all repositories.
 - **Only select repositories:** Provides access to read the metadata of only the repositories that you select.
4. Optional: If you selected **Only select repositories** in the previous step, select the repositories that you want the Ansible code bot to access from the **Select repositories** list.
5. Click **Install & Authorize**. A message is displayed that specifies the following permissions are granted automatically to the bot during installation:
 - Read access to metadata
 - Read and write access to code and pull requests
6. When prompted, log in to your Red Hat Single Sign-On account as an organization administrator.
7. Log in to the Ansible code bot dashboard:
 - a. On the **Authorize Ansible code bot** page, verify your account and repository permissions.
 - b. Click **Authorize Ansible**.
From the **Authorize Ansible code bot** page, the following actions occur:
 - Ansible code bot verifies that you are a part of an organization that has an active subscription to Red Hat Ansible Automation Platform.
 - GitHub requests read permissions to access the repositories associated with your account.

On successful authorization, you are logged in to Ansible code bot dashboard that displays all your repositories that have the Ansible code bot installed along with their scan status. If you did not set up a scan schedule earlier, the dashboard displays the repositories without any associated scan history. You can scan your Git repository by starting a manual scan, or configure a schedule to scan your repository at regular intervals. You can also add a repository for scanning or remove an existing repository from being scanned. For more information, see [Managing repository scans](#).

Additional resources

- [Troubleshooting Ansible code bot errors](#)

8.1.1. Uninstalling the Ansible code bot

If you no longer want to use the Ansible code bot, you can uninstall it from GitHub. Once uninstalled, you can still access the Ansible code bot dashboard but you cannot see the repositories on the dashboard or scan your repositories.

Procedure

1. Log in to GitHub by using the account associated with your organization.

2. In GitHub, click your profile photo > **Settings**.
3. Under Integrations, click **Applications** > **Installed GitHub Apps**
4. Click **Configure** beside the Ansible code bot app.
5. Under the **Danger zone** area, click **Uninstall**.
The Ansible code bot app is uninstalled from your GitHub account.

8.2. MANAGING REPOSITORY SCANS

The Ansible code bot dashboard displays a list of your repositories where the code bot is installed, and indicates if the scan schedule is not set, or is set to manual or scheduled scan.

You can scan your Git repository by starting a manual scan, or configure a schedule to scan your repository at regular intervals. After the scan is completed, you can view the scan history (start time, status, type of scan, link to the pull request if it was created, and the log message if the scan failed). You can also add new repositories for scanning or remove existing repositories from being scanned.

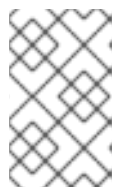
8.2.1. Manually scanning your Git repositories

You can manually scan your Git repositories if you did not set up a scanning schedule for your Ansible code bot or if you do not want to wait for the next scheduled scan. If you manually scan your repository, and no pull request was created, it is likely so because a duplicate pull request already exists. You can scan your repository from both the Ansible code bot dashboard and GitHub.

8.2.1.1. Manually scanning the repository from the Ansible code bot dashboard



Procedure

1. Log in to the [Ansible code bot dashboard](#).
The **Repositories** list displays a list of repositories that you selected for scanning.



NOTE

If you do not see your repository in the **Repositories** list, you can add it for scanning. For more information, see [Adding or removing repositories from the Ansible code bot](#).

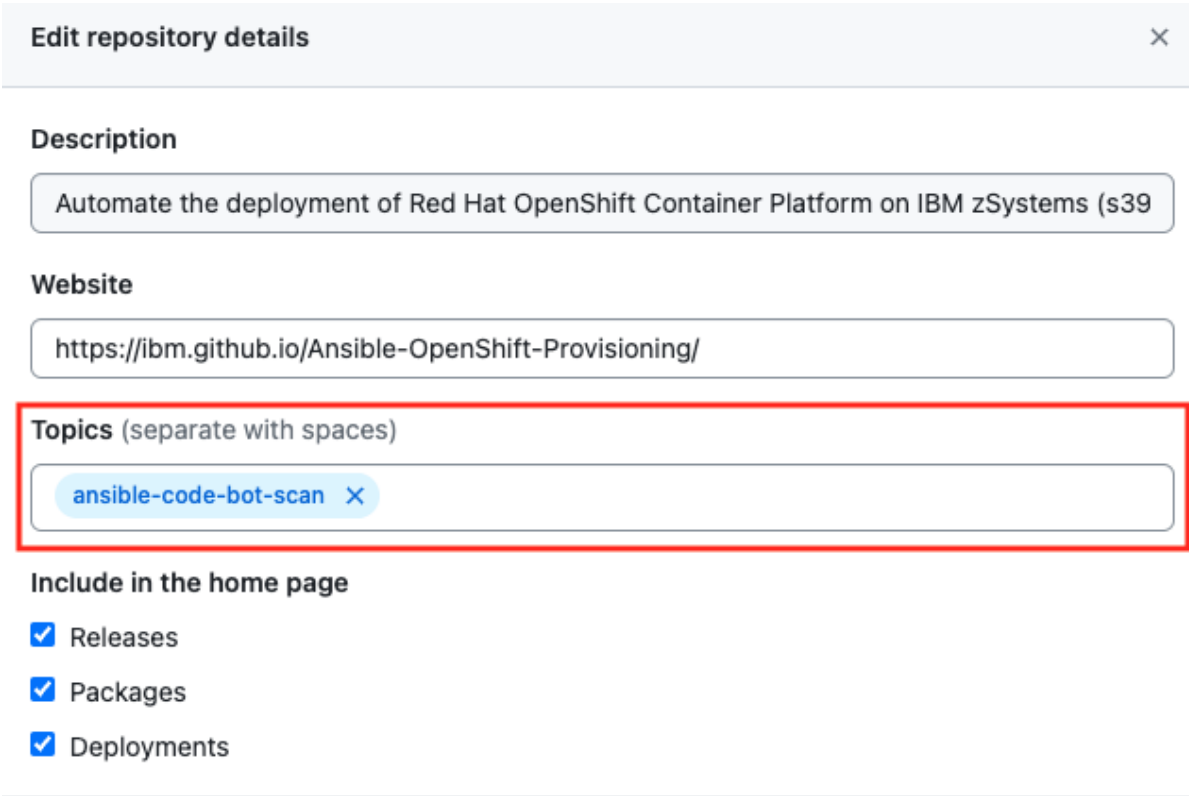
2. To start a manual scan of your repository, click the **Ellipsis** icon () beside the repository that you want to scan and select **Scan now**.
3. Click **Refresh** to view the status of the scan job.
4. To view more details about your repository scans, click the **Ellipsis** icon () beside the repository and select **View scan history**.
The repository's scan history is displayed along with the scan start time, scan status, type of scan (scheduled or manual), link to the pull request if it was created, and the log message if the scan failed.

- To view your repository on GitHub, click the **Ellipsis** icon () beside the repository and select **View repository**.

8.2.1.2. Manually scanning the repository from GitHub

Procedure

- In GitHub, go to the main page of the repository that you want to scan.
- To modify the repository settings, click the **Settings** icon beside the **About** area.
- In the **Topics** field, enter the keyword topic **ansible-code-bot-scan** to the repository. The following illustration shows the keyword topic for starting a manual scan:



Edit repository details ×

Description

Automate the deployment of Red Hat OpenShift Container Platform on IBM zSystems (s39)

Website

https://ibm.github.io/Ansible-OpenShift-Provisioning/

Topics (separate with spaces)

ansible-code-bot-scan ×

Include in the home page

- Releases
- Packages
- Deployments

Cancel Save changes

- Click **Save changes**.
Based on the repository webhook event, Ansible code bot starts a manual scan of your repository. If the avoid duplicate pull requests condition is not met, then the manual scan results in a new pull request with all the necessary Ansible code bot recommendations.

Additional resources

- [Troubleshooting Ansible code bot errors](#)

8.2.2. Configuring the Ansible code bot to scan your repository at regular intervals

You can schedule the Ansible code bot to scan your repositories at daily, weekly, or monthly intervals. To specify a scan schedule for your repository, create a configuration file **ansible-code-bot.yml** within your repository and specify your scan schedule in the **.yml** file.

You can specify one of following interval cadence to scan your Git repositories:

- Daily: Runs every day from Monday to Sunday.
- Weekly: Runs once a week. By default, this is on Monday.
- Monthly: Runs on the first day of the month, once each month.

For each interval cadence, Ansible code bot starts scanning your Git repositories at 9 AM UTC.

Procedure

1. In GitHub, navigate to the repository that you want to scan.
2. Create a **.yml** configuration file named **ansible-code-bot.yml** in your repository **.github** folder. For example, **.github/ansible-code-bot.yml**.
3. In the configuration file, specify the **interval** parameter. You can specify the **interval** parameter as **daily**, **weekly**, or **monthly**. For example:

```
schedule:
  interval: "<daily | weekly | monthly>"
```



4. Commit your changes.

The Ansible code bot starts scanning your repository per the schedule you configured at 9 AM UTC time.

8.2.3. Viewing your repository's scan history

Use the Ansible code bot dashboard to see a list of your repositories and their scan history.

Procedure

1. Log in to the [Ansible code bot dashboard](#) .
The Ansible code bot dashboard displays a list of your repositories where the code bot is installed, and indicates if the scan schedule is not set, or is set to manual or scheduled scan.
2. To view the history of your repository's scans, click the **Ellipsis** icon () beside the repository and select **View scan history**.
The repository's scan history is displayed along with the scan start time, scan status, type of scan (scheduled or manual), link to the pull request if it was created, and the log message if the scan failed.
3. To view your repository on GitHub, click the **Ellipsis** icon () beside the repository and select **View repository**.

8.2.4. Adding or removing repositories from the Ansible code bot

You can enable the Ansible code bot for a repository, or remove repositories that you no longer want to manage.

Procedure

1. Log in to the [Ansible code bot dashboard](#).
2. Click **Manage Code Bot on GitHub**.
3. In GitHub, click your profile photo > **Settings**.
4. Under **Integrations**, click **Applications**.
5. From the **Repository access** area, perform one of the following tasks:
 - **Add a new repository.** From the **Select repositories** list, select the repository that you want to add. The newly-added repository is displayed on the Ansible code bot dashboard.
 - **Remove an existing repository.** From the **Select repositories** list, click the **Cross** icon beside the repository that you want to delete. The deleted repository details are no longer visible on the Ansible code bot dashboard.
6. Click **Save**.

8.3. HOW ANSIBLE CODE BOT HANDLES DUPLICATE PULL REQUESTS

- If Ansible code bot has created a pull request on the latest commit default branch, it does not scan the repository. The bot skips scanning the repository because the pull request was committed on the latest default branch, and no new commit was made after that pull request.
- If there is an existing pull request that is not on the latest commit default branch, the Ansible code bot does a pull request difference to compare the changes in both branches. The following scenarios are possible:
 - **There is no difference in the existing and new scan results** Ansible code bot does not push the scan results as a new pull request.
 - **There are differences found in the existing and the new scan results** the Ansible code bot creates a new pull request. The newly-created pull request does not close the existing pull request, against which the pull request difference was noted. This behavior makes it easier for the repository administrator to review only the latest pull request created by the Ansible code bot, and the administrator can avoid reviewing the older pull requests created by the bot. If required, the administrator can close the older pull requests.

CHAPTER 9. VIEWING AND MANAGING ADMIN DASHBOARD TELEMETRY

Red Hat Ansible Lightspeed collects the following telemetry data by default:

- **Operational telemetry data**
This is the data that is required to operate and troubleshoot the Ansible Lightspeed service. For more information, refer the Enterprise Agreement. You cannot disable the collection of operational telemetry data.
- **Admin dashboard telemetry data**
This is the data that provides insight into how your organization users are using the Ansible Lightspeed service, and the metrics are displayed on the Admin dashboard. You can also disable the Admin dashboard telemetry if you no longer want to collect and monitor the telemetry data.



NOTE

Viewing telemetry data on the Admin dashboard is not yet supported on Red Hat Ansible Lightspeed on-premise deployments.

9.1. PREREQUISITES

To view and manage the Admin dashboard telemetry data, ensure that you have the following:

- You have organization administrator privileges to a Red Hat Customer Portal organization with a valid Red Hat Ansible Automation Platform subscription.
- You have installed the Ansible VS Code extension v2.13.148 that is required to collect Admin dashboard telemetry.



IMPORTANT

Red Hat Ansible Lightspeed does not collect users' personal information, such as usernames or passwords. If any personal information is inadvertently received, the data is deleted. For more information about Red Hat Ansible Lightspeed's privacy practices, see the [Telemetry Data Collection Notice for the Admin dashboard](#).

9.2. WHAT TELEMETRY DATA IS COLLECTED?

Following is the list of telemetry data that Red Hat Ansible Lightspeed collects:

- Details of the organization that you are logged into, such as organization ID and account number
- Large language models that you are connected to
- Inline suggestions that were accepted, rejected, or ignored by your organization users
- User sentiment feedback
- Top 10 modules returned in code recommendations

9.3. VIEWING THE ADMIN DASHBOARD TELEMETRY

The Admin dashboard displays the analytics telemetry data that you can use to gain insight into how your organization users are using the Ansible Lightspeed service.

The Admin dashboard displays the following charts:

- **Inline suggestions accepted, rejected, or ignored by users**
This graph tracks the number of inline suggestions that were accepted, rejected, or ignored by users in your organization. Use this graph to gain insight into how your organization users are using the Ansible Lightspeed service.
- **User sentiment**
This graph measures the users' feedback (feelings, opinions). Use this graph to gain insight into the overall user experience with Red Hat Ansible Lightspeed.
- **Top 10 modules returned in code recommendations**
This graph displays the top 10 modules returned in code recommendations. Use this metric to determine which modules are being suggested the most to your organization's automation developers.

Procedure

1. Log in to the [Ansible Lightspeed with IBM watsonx Code Assistant Hybrid Cloud Console](#) as an organization administrator.
2. From the navigation panel, select **Ansible Lightspeed > Admin Dashboard**
The Admin dashboard displays a graphical representation of analytics telemetry data for the last 30 days by default.
3. Use the following filters to refine your telemetry data:
 - To view the telemetry data for a specific time period or for a custom date range, select the date range from the **Quick Date Range** list.
 - To view the telemetry data for a specific IBM watsonx Code Assistant model only, select the model ID from the **Model Name** list. By default, the Admin dashboard displays telemetry data for all models.

9.4. DISABLING THE ADMIN DASHBOARD TELEMETRY

Red Hat Ansible Lightspeed collects the Admin dashboard telemetry data by default. The data provides insight into how your organization users are using the Ansible Lightspeed service. If you no longer want to collect analytics telemetry data for your organization, you can disable the Admin dashboard telemetry.

After you disable the Admin dashboard telemetry, the Ansible Lightspeed service no longer collects the analytics telemetry data for your organization. The earlier telemetry data is still available on the Admin dashboard, but no latest data is displayed. If you re-enable the Admin dashboard telemetry, the Ansible Lightspeed service starts collecting data for your organization, and the metrics are displayed on the Admin dashboard after 24 hours.

Prerequisites

- You have organization administrator privileges to a Red Hat Customer Portal organization with a valid Red Hat Ansible Automation Platform subscription.

Procedure

1. Log in to the [Ansible Lightspeed portal](#) as an organization administrator.
2. From the login screen, click **Admin Portal**.
3. Under Admin Portal, click **Telemetry**.
4. To disable the Admin dashboard telemetry, select **Operational telemetry data only**.



NOTE

To re-enable the Admin dashboard telemetry, select **Admin dashboard telemetry data**.

5. Click **Save**.

CHAPTER 10. TROUBLESHOOTING

This section contains information to help you diagnose and resolve issues with using Red Hat Ansible Lightspeed with IBM watsonx Code Assistant.

10.1. TROUBLESHOOTING RED HAT ANSIBLE LIGHTSPEED CONFIGURATION ERRORS

10.1.1. Cannot access the Ansible Lightspeed administrator portal

The Ansible Lightspeed administrator portal can be accessed by the Red Hat organization administrator only.

If you are the Red Hat organization administrator, before you access the Ansible Lightspeed administrator portal, ensure that:

- You have a valid Ansible Automation Platform subscription.

10.1.2. Cannot save the API key

When you enter the IBM watsonx Code Assistant API key, authentication fails and shows the following error message:

IBM Cloud API key is invalid

Red Hat Ansible Lightspeed verifies the API key by generating an associated access token. To resolve the error, ensure that you have not accidentally included any extra spaces when obtaining the API key from IBM watsonx Code Assistant. If you still cannot upload the API key, contact [IBM Support](#).

10.1.3. Cannot configure the model ID due to authentication failure

When you enter the model ID in the Red Hat Ansible Lightspeed administrator portal, the authentication fails.

To resolve the error, ensure that:

- You have configured a valid API key before you upload the model ID.
- You have not accidentally included any extra spaces when entering the model ID.

10.1.4. Cannot configure the model ID due to inference failure

While validating the model ID, Red Hat Ansible Lightspeed performs a test inference. If Red Hat Ansible Lightspeed detects an error, the validation fails and an **Inference failed** message is displayed.

To resolve the error, ensure that:

- You have a valid API key and model ID.
- You have not accidentally included any extra spaces when obtaining the API key and model ID from IBM watsonx Code Assistant.

10.2. TROUBLESHOOTING ANSIBLE VISUAL STUDIO CODE EXTENSION ERRORS

10.2.1. Cannot view the generated code recommendations using the Ansible VS Code extension

The following scenarios are possible:

- You receive a **403 error** message.
To resolve this error, ensure that:
 - Your organization administrator has configured Red Hat Ansible Lightspeed for your organization.
 - You are part of an organization that has a trial or paid subscription to both Ansible Automation Platform and IBM watsonx Code Assistant.
- You have not configured the required Ansible VS code extension settings.
 - To resolve this error, ensure that you have enabled the **Lightspeed:Enabled** and **Lightspeed → Suggestions:Enabled** settings. For more information, see [Configure the Ansible VS Code extension](#).
- You receive a **Failure on completion requests** error when you make inference requests in VS Code.
If you are part of an organization that has a trial or paid subscription to both Ansible Automation Platform and IBM watsonx Code Assistant, but your organization administrator has not configured an IBM watsonx Code Assistant model for your organization, you will encounter a **Failure on completion requests** error when you make inference requests in VS Code.
- Your VS Code Workspace settings override user settings.
If your Workspace settings are configured, they can override our user settings even if you have configured the Ansible VS Code extension correctly. The Workspace settings can disable your VS Code extension settings, and therefore you cannot access the Ansible Lightspeed service.

To resolve this error, ensure that there are no Workspace settings configured in VS Code. For more information, see [Workspace settings](#) in the VS Code documentation.
- You entered a multitask prompt, but code recommendations were not generated.
To resolve this error, log out of VS Code and log in again using your Red Hat account.
- You clicked a different location or switched to a different window; therefore, the populated code recommendations disappeared.
The Red Hat Ansible Lightspeed service could take multiple seconds per task to populate the code recommendations. If you are using a multitask prompt, the Red Hat Ansible Lightspeed service takes a bit longer to populate the results. Do not move your cursor or press any key while the code recommendation is being generated. If you change the cursor location or press any key, the Ansible VS Code extension cancels the request and the Red Hat Ansible Lightspeed service does not process your request. In this scenario, you must get the cursor back to its original position and repopulate the results.

10.2.2. Cannot request code recommendations by using the Ansible VS Code extension

The following error message is displayed:

Your trial to the generative AI model has expired. Refer to your IBM Cloud Account to re-enable access to the IBM watsonx Code Assistant.

To resolve this error, refer to your IBM Cloud account and select an upgrade option.

10.3. TROUBLESHOOTING ANSIBLE CODE BOT ERRORS

10.3.1. Cannot access Ansible code bot

After you install Ansible code bot and attempt to log in, you receive the following error message:

Your organization does not have a valid Red Hat Ansible Lightspeed subscription

After you install Ansible code bot, you are redirected to a page that shows an active subscription status, as shown in the following image:

Figure 10.1. Ansible code bot login screen with an active subscription



If the login screen displays an inactive subscription status, Ansible code bot does not scan your Git repositories. The error occurs because your organization does not have a valid Ansible Automation Platform subscription. To resolve this error, ensure that you are part of an organization that has a valid Red Hat Ansible Automation Platform subscription.

10.3.2. Cannot scan your Git repository using Ansible code bot

If the Ansible code bot is not configured correctly, it does not scan your Git repositories or does not create pull requests.

To resolve Ansible code bot errors, ensure that:

- You have selected all the Git repositories that you want to scan.
- You have a **.yml** configuration file named **ansible-code-bot.yml** in your repository **.github** folder. For example, **.github/ansible-code-bot.yml**.

Run a manual scan on your git repositories by adding the **ansible-code-bot-scan** topic to your repository. For more information, see [Manually scan your Git repositories](#).

If the Ansible code bot still cannot scan your Git repository, the following scenarios are possible:

- The Ansible code bot did not identify any ansible-lint violations in the Git repository.
- The Ansible code bot does not have permission to scan the Git repository.
- Your organization does not have a valid Red Hat Ansible Automation Platform subscription.

10.3.3. Cannot create pull requests

You might encounter an error where the Ansible code bot cannot create pull requests after scanning your Git repositories.

To resolve this error, ensure that:

- You have verified that there are no duplicate pull requests. For more information, see [How Ansible code bot handles duplicate pull requests](#).
- You have deleted the branches after closing the pull requests created by the Ansible code bot. For more information, see [Deleting a branch used for a pull request](#).