

Posit Assistant

[Posit Assistant](#) is a set of artificial intelligence (AI) features available as a downloadable extension for RStudio. To provide feedback or report bugs, please open a [GitHub Issue on the RStudio repository](#).

Prerequisites

- To use Posit Assistant in RStudio, you must have a compatible version of RStudio installed. Posit Assistant is available for RStudio Desktop or Server 2026.04.0 and later. Posit Workbench disables Posit Assistant by default, but an administrator can enable it.
- To use Posit Assistant, you must have access to the internet in order to send requests to the Posit AI APIs and receive suggestions.

Posit Assistant in RStudio Server and Workbench

Support for RStudio Server and Posit Workbench is currently considered experimental and will be fully supported in a future release.

Setup

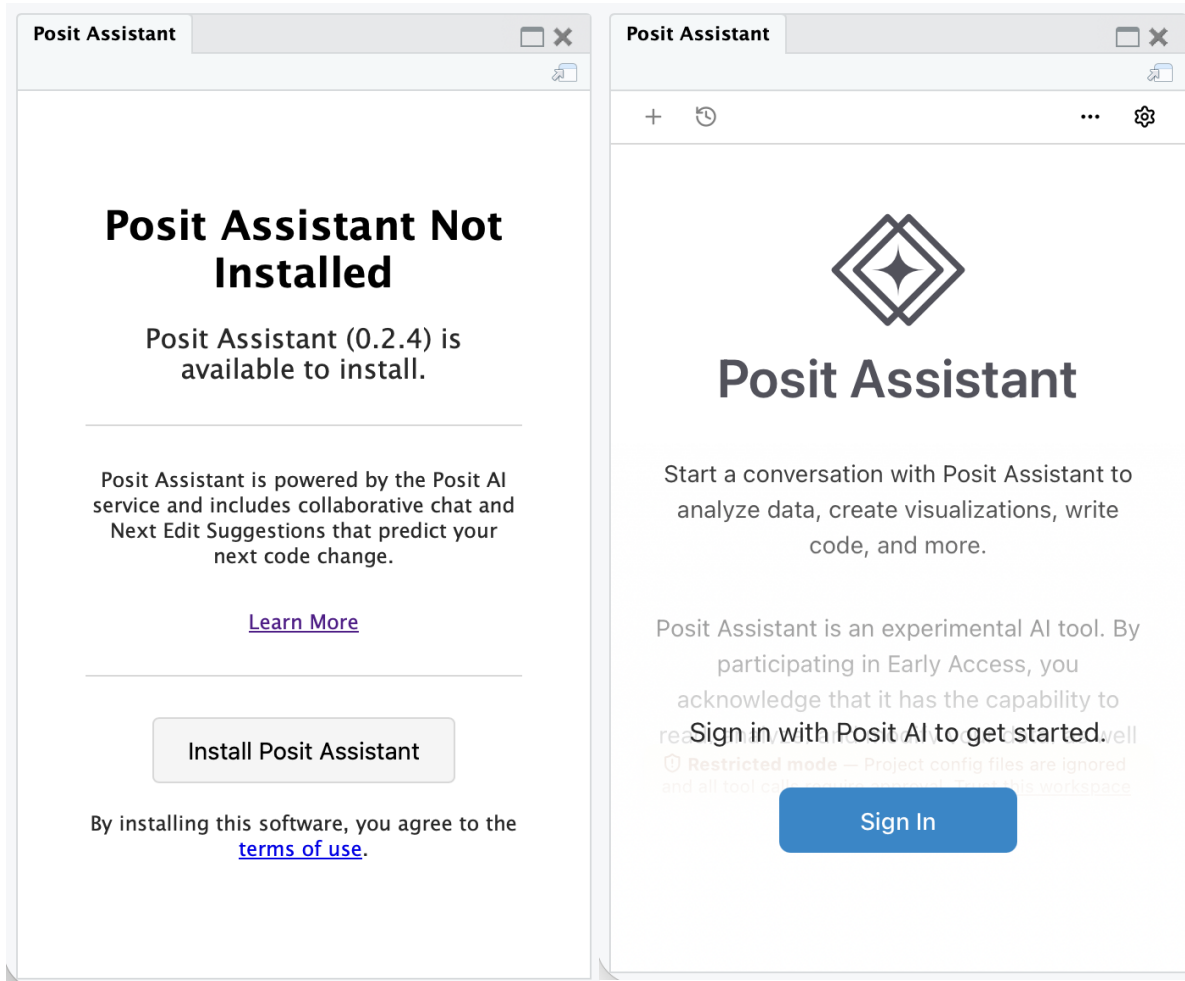
To configure Posit Assistant in RStudio click on the “Posit Assistant” button on the toolbar to display the “Posit Assistant” pane.



Click **Install Posit Assistant** to download and install the add-in package, then click **Sign In** to login and signup for Posit-AI.

Posit AI Account Assistance

For more information on signing up and managing your Posit AI account visit <https://docs.posit.co/posit-ai>.



Once signed up and authenticated, the Posit Assistant chat feature is ready to use.

Using Posit Assistant

For information on using Posit Assistant, visit the [Posit Assistant documentation](#).

Removing Posit Assistant

To uninstall the downloaded Posit Assistant components, select **Help > Diagnostics > Uninstall Posit Assistant**. RStudio restarts after the uninstall completes.

RStudio preserves your chat history and account settings, which remain available if you reinstall Posit Assistant.

Hiding Posit Assistant features

To disable and hide the Posit Assistant features entirely, set the `RSTUDIO_DISABLE_POSIT_ASSISTANT` environment variable to any value (e.g., `RSTUDIO_DISABLE_POSIT_ASSISTANT=1`) before starting RStudio. When the variable is set, RStudio does not display the Posit Assistant toolbar button or pane, and you cannot install or use Posit Assistant in that session.

On RStudio Server, administrators can also disable Posit Assistant for all users by adding the following line to `/etc/rstudio/rsession.conf`:

```
posit-assistant-enabled=0
```

See the [Environment Variables](#) reference for other environment variables that affect RStudio.

Configuring LLM providers

Posit Assistant 0.7.7 and later can connect to a number of LLM providers in addition to Posit AI, including Anthropic (Claude), OpenAI, GitHub Copilot, Amazon Bedrock, Google Gemini, Google Vertex AI, OpenRouter, Snowflake Cortex, Microsoft Foundry, DeepSeek, and local models through Ollama or LM Studio.

To connect to a provider, click the model selector at the bottom of the Posit Assistant pane and choose **Configure LLM providers**. Select a provider and complete its setup, which varies by provider: entering an API key, signing in through the provider's own flow (Posit AI, GitHub Copilot), using AWS credentials (Amazon Bedrock), or pointing to a local endpoint (Ollama, LM Studio).

Most provider setup can be completed in that dialog. Settings the dialog does not expose – custom base URLs, extra HTTP headers, model lists, and fully custom providers – are configured by editing the provider configuration file described below.

The provider configuration file

Provider settings are stored in a JSON file shared by Posit products (RStudio and Positron):

- macOS / Linux: `~/.posit/ai/providers.json`
- Windows: `%USERPROFILE%\\.posit\ai\providers.json`

Posit Assistant watches this file and applies changes automatically; you do not need to restart RStudio. A JSON Schema (`providers.schema.json`) is written alongside the file, so editors that support the `$schema` property can provide validation and autocomplete while you edit.

! Never store secrets in `providers.json`

API keys and other secrets do not belong in `providers.json`. Enter credentials through the **Configure LLM providers** dialog (they are saved to a separate credential store) or supply them via [environment variables](#).

Each built-in provider has a key in the `providers` object: `positai`, `anthropic`, `copilot`, `openai`, `bedrock`, `gemini`, `openrouter`, `google-vertex`, `ollama`, `lmstudio`, `openai-compatible`, `snowflake-cortex`, `ms-foundry`, and `deepseek`. The most common per-provider fields are:

Field	Description
<code>enabled</code>	Whether the provider appears in the UI.
<code>baseUrl</code>	Custom base URL for the provider's API. Include the full path prefix, including any version segment (e.g. <code>https://api.anthropic.com/v1</code>) – a bare host will fail.
<code>endpoint</code>	Endpoint URL for local model servers (Ollama, LM Studio), e.g. <code>http://localhost:11434</code> .
<code>customHeaders</code>	Extra HTTP headers attached to every request for this provider.
<code>models</code>	Controls which models are listed. See Controlling the model list .

Enabling and disabling providers

All providers are enabled by default. To show only the providers you use, disable everything by default and enable providers individually:

```
{
  "providers": {
    "default": { "enabled": false },
    "positai": { "enabled": true },
    "anthropic": { "enabled": true }
```

```
}  
}
```

Custom base URLs and headers

To route a provider's traffic through an enterprise proxy or API gateway, set a custom base URL and any headers the gateway requires:

```
{  
  "providers": {  
    "anthropic": {  
      "baseUrl": "https://gateway.example.com/v1",  
      "customHeaders": {  
        "x-gateway-tenant": "team-42"  
      }  
    }  
  }  
}
```

Do not put authentication headers managed by the provider SDK (`Authorization`, `x-api-key`, `anthropic-version`) in `customHeaders` – they would overwrite the credentials Posit Assistant sends and break the connection.

Custom providers

Providers that are not in the built-in list – for example a self-hosted model server or an enterprise LLM gateway – are defined under `providers.custom`. Each entry is keyed by a name of your choosing (which must not collide with a built-in provider key) and requires a `type` field that selects the API the provider speaks: `openai-compatible`, `aws`, `snowflake`, `google-vertex`, `ollama`, `lmstudio`, `deepseek`, `openrouter`, or `ms-foundry`.

For most servers and gateways, use `openai-compatible`, which works with any endpoint implementing the OpenAI chat completions API (including gateways that proxy Anthropic, OpenAI, or Gemini models):

```
{  
  "providers": {  
    "custom": {  
      "my-gateway": {  
        "type": "openai-compatible",  
        "baseUrl": "https://llm.example.com/v1"  
      }  
    }  
  }  
}
```

```

    }
  }
}

```

If the server exposes a `/models` endpoint, its models are discovered automatically. Otherwise, turn discovery off and declare the models yourself:

```

{
  "providers": {
    "custom": {
      "my-gateway": {
        "type": "openai-compatible",
        "baseUrl": "https://llm.example.com/v1",
        "models": {
          "discovery": "off",
          "custom": [
            {
              "id": "llama-3.3-70b",
              "name": "Llama 3.3 70B",
              "maxContextLength": 128000,
              "supportsTools": true,
              "supportsImages": false,
              "supportsToolResultImages": false,
              "supportsWebSearch": false
            }
          ]
        }
      }
    }
  }
}

```

After saving the file, the custom provider appears in the **Configure LLM providers** dialog under the name you gave it, where you can enter its API key if it requires one.

Controlling the model list

Any provider block (built-in or custom) can include a `models` section to control which models appear in the model selector:

Field	Description
discovery	"auto" (default) queries the provider's model listing endpoint; "off" disables discovery.
allow	When non-empty, an exclusive allowlist of model IDs.
deny	Model IDs to remove from the list; takes precedence over allow .
overrides	Partial metadata patches keyed by model ID (e.g. correct a context length).
custom	Complete definitions for models that discovery does not return.

For example, to limit OpenAI to two models:

```
{
  "providers": {
    "openai": {
      "models": {
        "allow": ["gpt-5", "gpt-5-mini"]
      }
    }
  }
}
```

Providing API keys with environment variables

As an alternative to the **Configure LLM providers** dialog, built-in providers can read API keys from environment variables:

Provider	Environment variable(s)
Anthropic	ANTHROPIC_API_KEY
OpenAI	OPENAI_API_KEY
Google Gemini	GEMINI_API_KEY
OpenRouter	OPENROUTER_API_KEY
OpenAI-compatible	OPENAI_COMPATIBLE_API_KEY
Microsoft Foundry	MS_FOUNDRY_API_KEY
Snowflake Cortex	SNOWFLAKE_TOKEN
DeepSeek	DEEPSEEK_API_KEY
Amazon Bedrock	AWS_REGION, AWS_PROFILE, AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, AWS_SESSION_TOKEN

The Posit Assistant backend inherits the R session's environment, so these can be set in `~/.Renviron`; restart RStudio for changes to take effect. Custom providers do not support environment variables – enter their API keys in the **Configure LLM providers** dialog.

Terms and Conditions

- [Posit Assistant](#)