aws

User Guide

Amazon Bedrock Studio

Amazon Bedrock Studio: User Guide

Table of Contents

What is Amazon Bedrock Studio?

Amazon Bedrock Studio is a web app that lets you easily protoype apps that use Amazon Bedrock models and features, without having to set up and use a developer environment. For example, you can use Amazon Bedrock to try a prompt with an Anthropic Claude model without having to write any code. Later, you can use Bedrock Studio to create a prototype app that uses an Amazon Bedrock model and features, such as a Knowledge Base or a guardrail, again without having to write any code.

Apps that you create with Bedrock Studio can integrate the following Amazon Bedrock features (known as *components* in Bedrock Studio).

- **Knowledge Bases** — Enrich apps by including context that is received from querying a Knowledge Base.
- **Guardrails** — Lets you implement safeguards for your Bedrock Studio app based on your use cases and responsible AI policies.
- **Function calling** — Lets a model call a function to access a specific capability or module when handling a prompt.

To use Amazon Bedrock Studio, you must be a member of a workspace. Your organization will provide you with login details. If you don't have login details, contact your administrator.

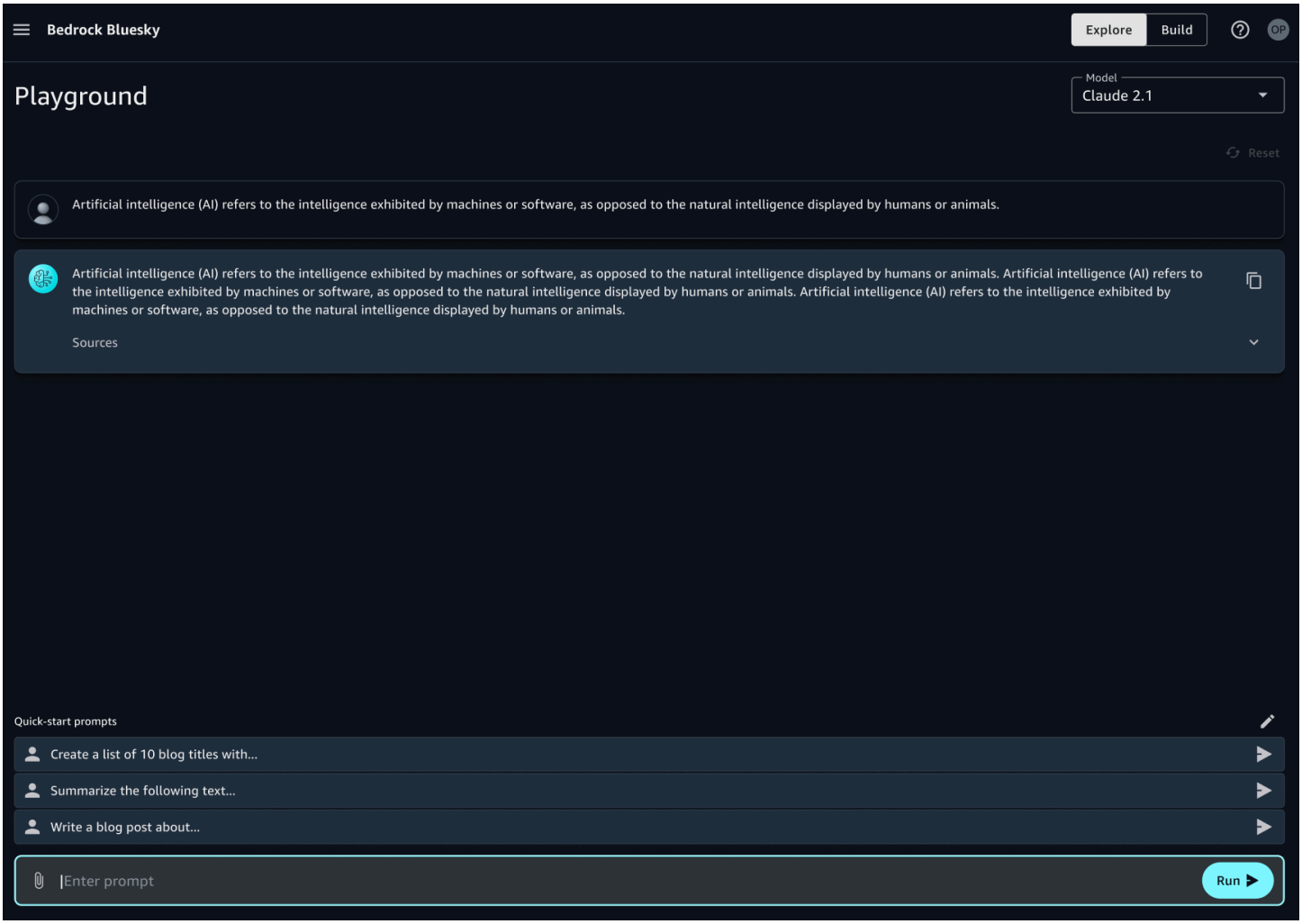
> **Note**
>
> If you are administrator and need information about administering an Amazon Bedrock Studio workspace, see Amazon Bedrock Studio in the Amazon Bedrock user guide.

Workspaces

A *workspace* in Amazon Bedrock Studio is where you experiment with Amazon Bedrock models and where you can build Amazon Bedrock enabled apps. As an Amazon Bedrock Studio user, you can use a workspace in two types of user modes, *Explore* or *Build*.

Explore mode

Explore mode provides a *playground* that lets you easily try a model by sending prompts to the model and viewing the responses. For more information, see Getting started with Amazon Bedrock Studio



Build mode

Build mode is where you can create apps that use Amazon Bedrock models. Within Build mode, you use projects to organize the apps you create and the components an app uses. If you work on a team, you can collaborate by sharing a project with other team members. In your app you can include Amazon Bedrock features, such as Knowledge Bases and guardrails. For more information, see Building an app with Amazon Bedrock Studio.

Are you a first-time Amazon Bedrock Studio user?

If you're a first-time user of Bedrock Studio, we recommend that you read the following sections in order:

1. **Getting started with Amazon Bedrock Studio** – In this section, you access your Amazon Bedrock Studio workspace for the first time and use Explore mode to experiment with an Amazon Bedrock model.
2. **Building an app with Amazon Bedrock Studio** – In this section, you use Build mode to create a project in your workspace and create a simple Amazon Bedrock Studio app.
3. **Sharing an Amazon Bedrock Studio project** – In this section, you learn how to collaborate on a project by sharing a project with your team members.

Getting started with Amazon Bedrock Studio

Amazon Bedrock Studio is a web app that lets you easily protoype apps that use Amazon Bedrock models and features.

In these getting started instructions, we show you how to access your workspace and use the Explore mode to send a prompt to one of the Amazon Bedrock models. The Explore mode provides a *playground* that lets you easily try a model by sending prompts to the model and viewing the responses. After experimenting with the playground, you can then create an app to experiment with other Amazon Bedrock features such guardrails and system prompts. To create an app with Amazon Bedrock Studio, see Building an app with Amazon Bedrock Studio.

Topics

Prompts

A prompt is input you send to a model in order for it to generate a response, in a process known as inference. For example, you could send the following prompt to a model.

```
What is Avebury stone circle?
```

When you run the prompt in the playground, the playground shows a response from the model that is similar to the following.

```
Avebury stone circle is a Neolithic monument located in Wiltshire, England.
It consists of a massive circular bank and ditch, with a large outer circle of standing
 stones
that originally numbered around 100.
```

The Explore playground also provides quick start prompts that illustrate the kinds of prompt that you can send to a model.

The Explore playgrounds lets you try *multimodal* prompts, which are prompts that include images and text. This allows you to pass an image to a model and ask questions such as *What's in this image?*. Not all models support multimodal prompts.

The actual response that you get for a prompt depends on the model you use.

In apps you create, you can influence the response in various ways, such as by using inference parameters, which allow to filter out lower probability responses or limit the response in other ways. To experiment with inference parameters, or use an Amazon Bedrock component, such as as a guardrail, you need to create a Bedrock Studio app. For more information, see Building an app with Amazon Bedrock Studio.

Using Explore mode to send prompts to a model

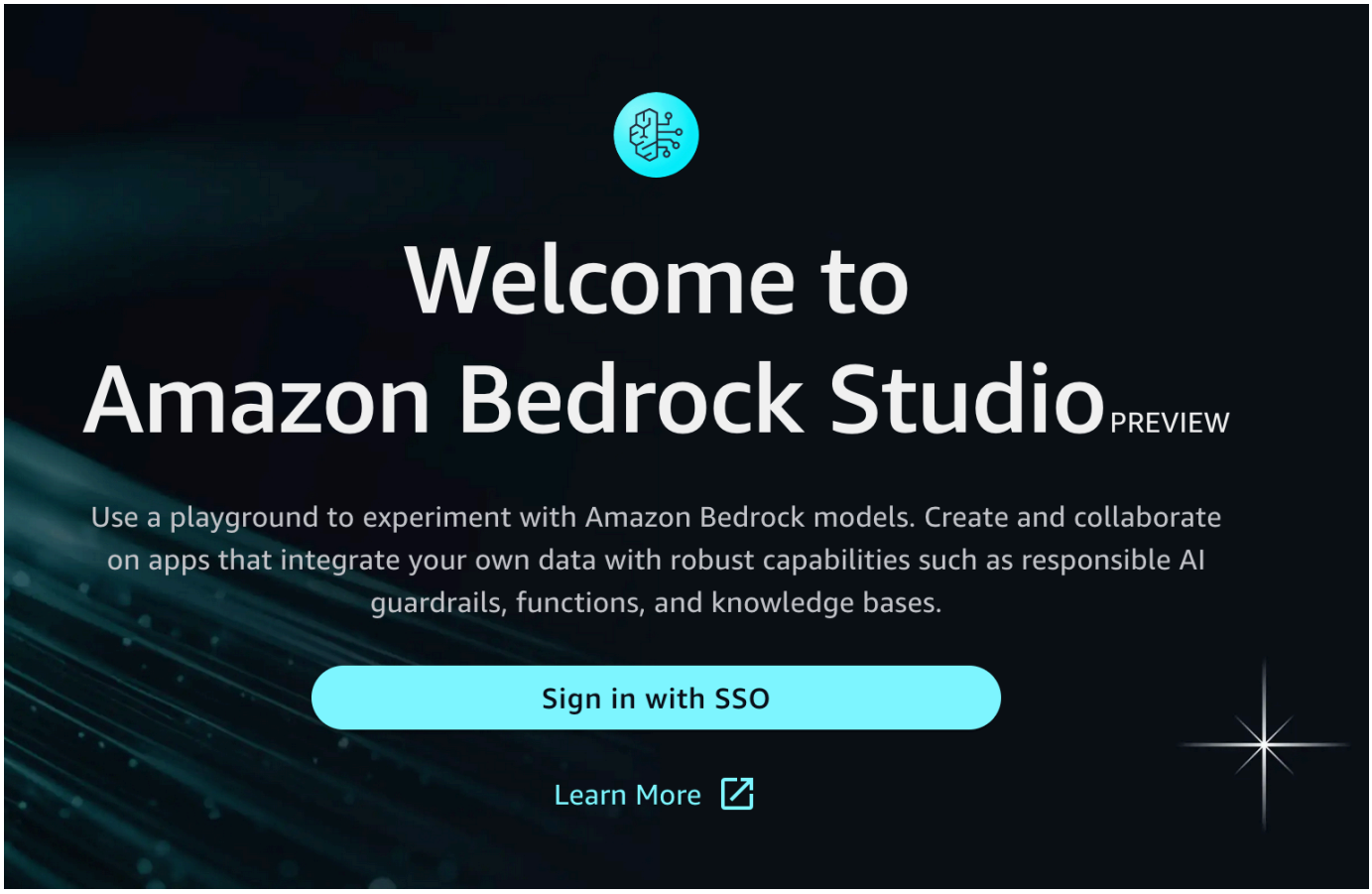
These instructions show you how to use the Explore playground to send a prompt to an Amazon Bedrock model.

To get started with Amazon Bedrock Studio, you need an email invitation to an Amazon Bedrock Studio workspace. If you haven't received the invitation email from your organization, contact your organization's adminstrator.

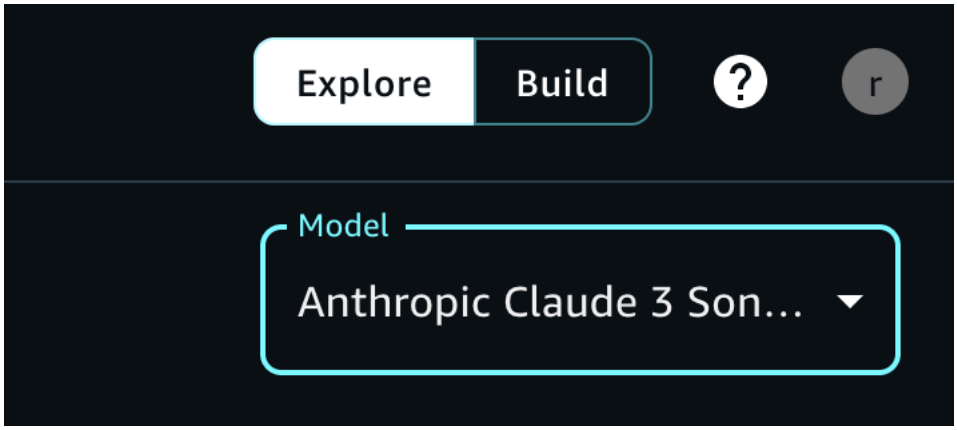
Your organization's administrator determines the Amazon Bedrock Studio features that you have access to. If you need access to a model or feature that you don't currently have access to, contact your organization's adminstrator.

To send a prompt to a model

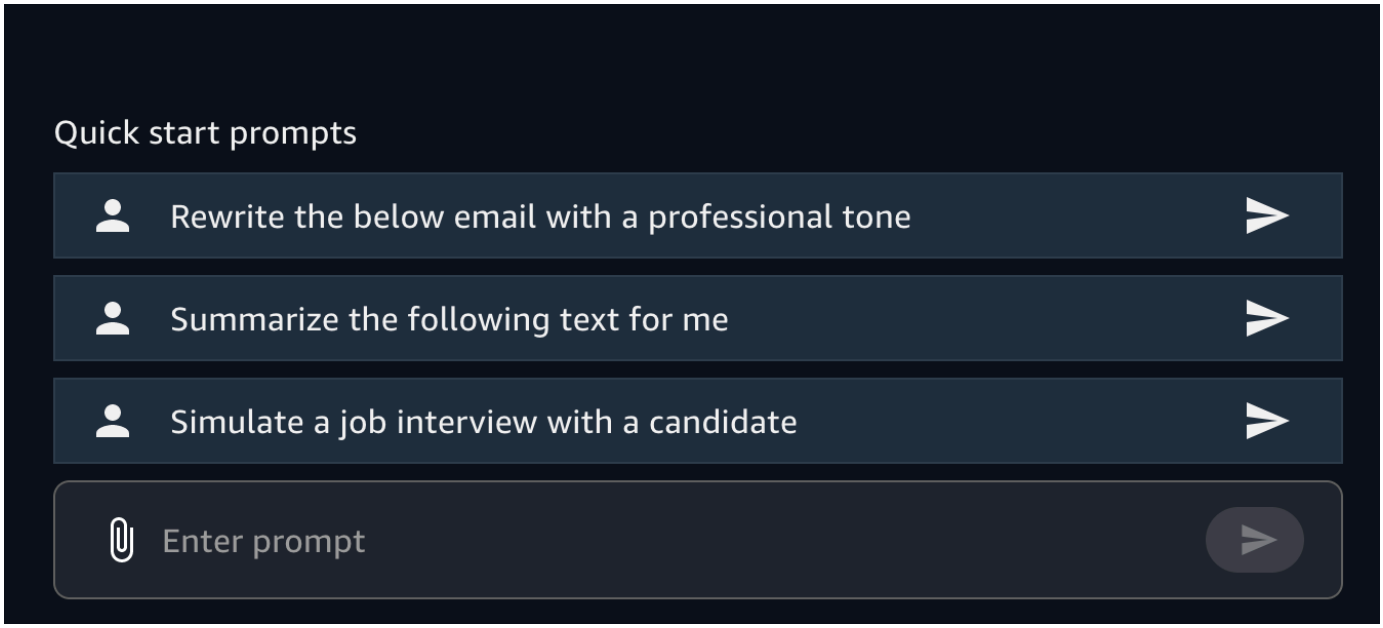
1. Open the invitation email sent to you by your organization.
2. In the email, choose the link to the workspace that you are invited to.
3. Choose **Sign in with SSO** to open your Amazon Bedrock Studio workspace.



4. Enter your the user name that you use to access your organization's resources and choose **Next**.
5. Enter your password and choose **Sign in**.
6. Make sure that you are in the Explore mode, by choosing **Explore** at the top right of the page.



7. In **Model** select a model to use, or use the default model.
8. In the **Quick start prompts** section at the bottom of the page, choose one of prompts.



The playground sends the prompt to the model and shows the text that the model generated in response. Continue the conversation until you are ready for the next step.

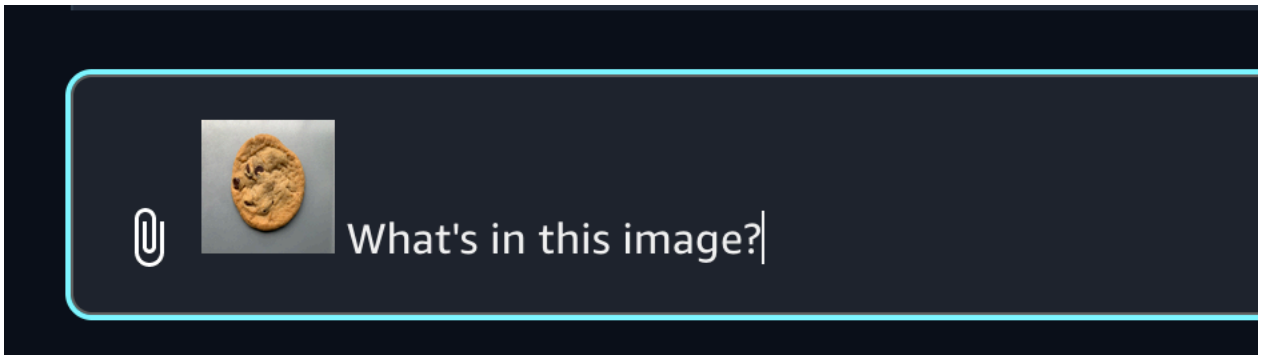
9. Choose **Reset** to clear the playground.
10. Try one of your own prompts, such as *What is Avebury stone circle?*, by entering the prompt in the **Enter prompt**. Press enter to send the prompt to the model. Continue the conversation by enter the prompt *Is there a museum there?*. The response shows how the model uses the previous prompt as context for generating its next response.
11. Some models are multimodal in that that they can process text and images. To try analyzing an image, do the following.

    a. In **Model** choose **Anthropic Claude 3 Sonnet**.

    b. Choose the paper clip button to open the file upload dialog box.

    c. Choose an image from your local computer.

    d. In the prompt edit box, next to the image that you uploaded, enter *What's in this image?*



e. Press enter to send the prompt. The playground will show the image and underneath the image show a description of the image.

12. (Optional) Try using another model and different prompts. Different models have different recommendations for creating, or engineering, prompts. For more information, see Prompt engineering guides.

Now that you are familiar with the explorer playground, try creating a Bedrock Studio app next. For more information, see Building an app with Amazon Bedrock Studio.

Prompt engineering guides

For general guideslines about creating prompts, see General guidelines for Amazon Bedrock LLM users.

For model specific information, see the following prompt engineering guides.

- **Anthropic Claude model prompt guide:** https://docs.anthropic.com/claude/docs
- **Anthropic Claude prompt engineering resources:** https://docs.anthropic.com/claude/docs/guide-to-anthropics-prompt-engineering-resources
- **Cohere prompt guide:** https://txt.cohere.com/how-to-train-your-pet-llm-prompt-engineering
- **AI21 Labs Jurassic model prompt guide:** https://docs.ai21.com/docs/prompt-engineering
- **Meta Llama 2 prompt guide:** https://ai.meta.com/llama/get-started/#prompting
- **Stability documentation:** https://platform.stability.ai/docs/getting-started
- **Mistral AI prompt guide:** https://docs.mistral.ai/guides/prompting_capabilities/

Building an app with Amazon Bedrock Studio

In Amazon Bedrock Studio you use the Build mode to create prototype apps that uses Amazon Bedrock models and features. You can also use the Build mode to try experiments not supported in the Explore mode playground, such as setting inference parmeters.

In the Build mode, you use a *project* to manage the apps that you create. An app can optionally use Amazon Bedrock components, such as function calling, or guardrails. You can also collaborate on a project by sharing the project with other team members. For more information, see Sharing an Amazon Bedrock Studio project.

In this section you create a simple app that creates song playlists for a radio station. Later you add the following features.

- A guardrail to prevent songs with inappropriate song titles.
- A data source that lets the app create playlists using your unique song information.
- A function that gets today's top 10 songs.

Topics

Creating an Amazon Bedrock Studio app

In this section, you learn how create a simple Amazon Bedrock Studio app that creates playlists for a radio station.

In the app, you use a system prompt to specify that the model should behave as an app that creates playlists for a radio station that plays rock and pop music. A system prompt is a type of prompt that provides instructions or context to the model about the task it should perform, or the persona it should adopt during the conversation.

The user can then use the app to create playlists based on different themes, such as songs that are related by shared artists.

The instructions show how you can change the genre of music that the app creates playlists for by changing the system prompt.

In the app, you can experiment with the randomness and diversity of the response that the model returns by changing the inference parameters. For example, you can influence the model to only add more popular songs to the playlists it generates.

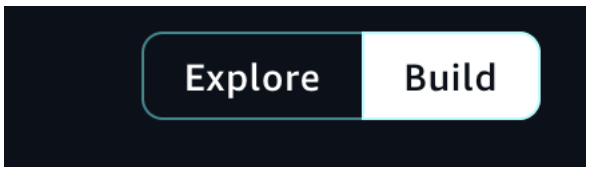
To create the app, you first need to create a *project*. A project can contain multiple apps and is also where you can add the Amazon Bedrock components that you want your apps to use. Later you will add guardail, data source, and function calling components to your app. You can share a project with other users and groups of users. For more information, see Sharing an Amazon Bedrock Studio project.

If you want to quickly experiment with sending prompts to a model, consider user the Explore view. For more information, see Getting started with Amazon Bedrock Studio.

For information about managing your projects and apps, see Managing your Amazon Bedrock Studio workspace.

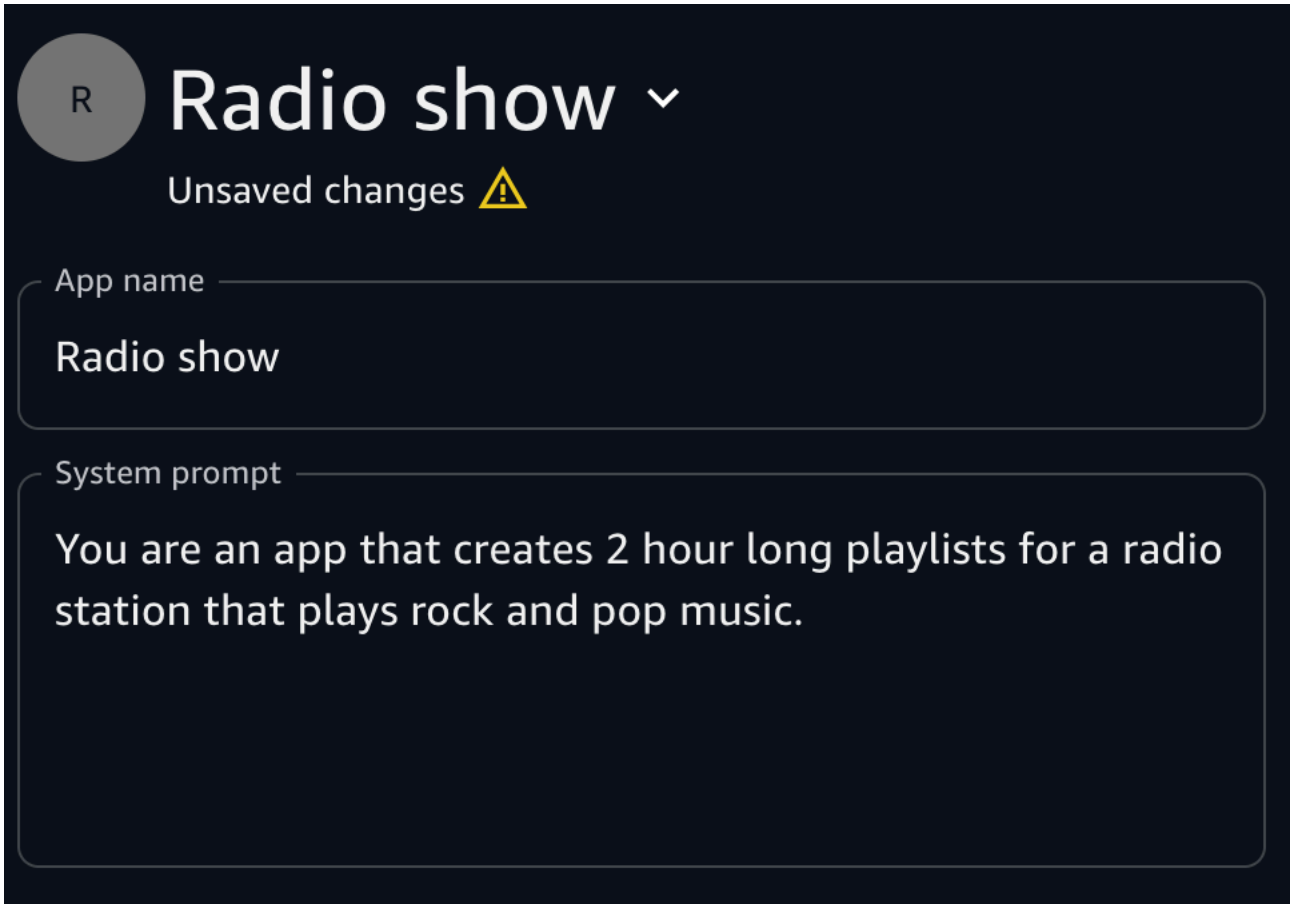
To create a Bedrock Studio app

1. Open your Amazon Bedrock Studio workspace.
2. Make sure that you are in Build mode, by choosing **Build** at the top right of the page.

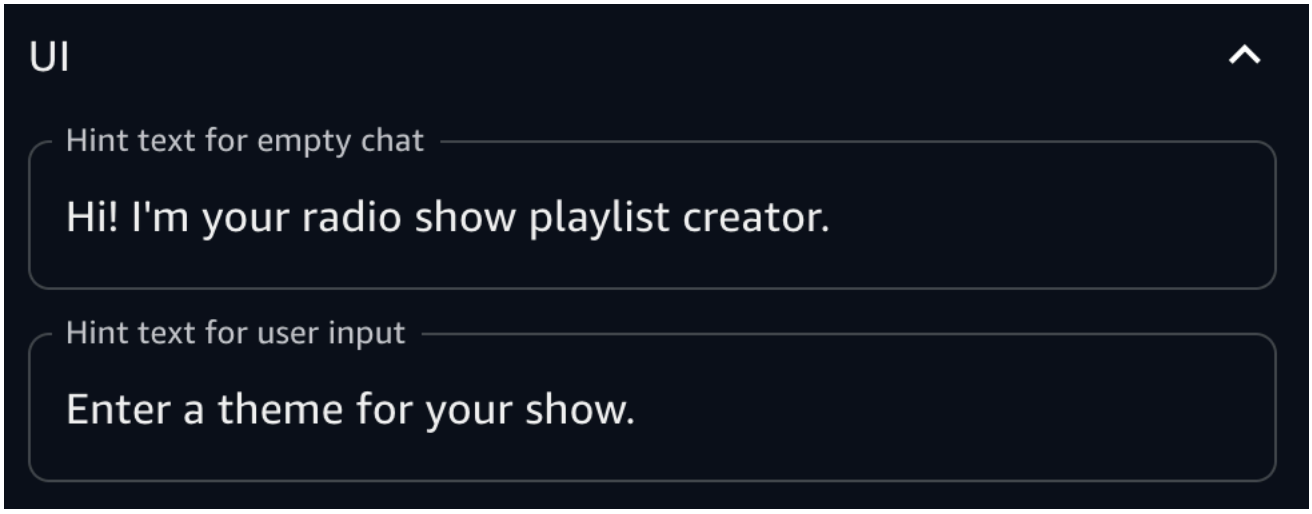
   

   If this is the first time that you've opened the workspace, Amazon Bedrock Studio creates a default project for you.
3. Open the default project or create a new project by do the following.
   a. On the **All projects** page, choose **Create new project**.
   b. In the **Create new project** section, enter a name for your project in **Project name**.
   c. Enter a description for your project in **Project description**.
   d. Choose **Create**. It might take a few minutes to create the project.
   e. In the tile for your app, choose **Go to project** to open your project.
4. On the project page, in the **Apps** section, choose **Create app** to create an app in the project.

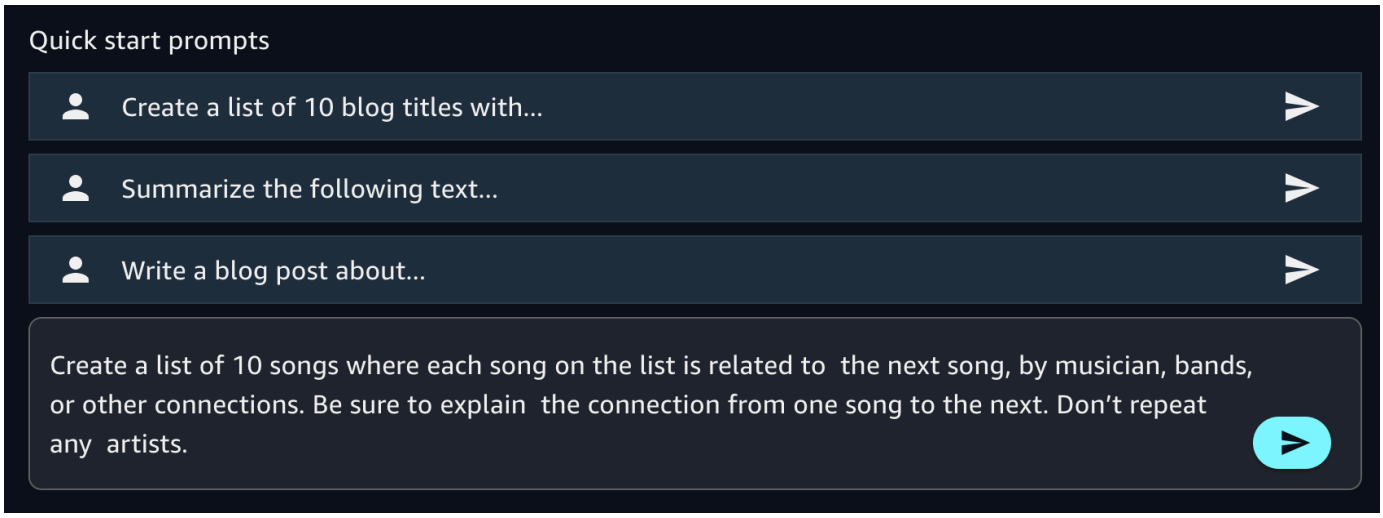
5. In the **Configs** pane, choose the model that you want your app to use in **Model**.
6. Enter *Radio show* as the name for your app in **App name**.
7. In **System prompt**, enter *You are an app that creates 2 hour long playlists for a radio station that plays rock and pop music.*



8. Enter hint text for the app by entering the following in the **UI** section.
   a. In **Hint text for empty chat** enter *Hi! I'm your radio show playlist creator.*
   b. In **Hint text for user input** enter *Enter a theme for your show.*



9. Choose **Save** to save your app.
10. In the prompt edit box, enter *Create a list of 10 songs where each song on the list is related to the next song, by musician, bands, or other connections. Be sure to explain the connection from one song to the next. Don't repeat any artists.*
11. Choose the run button to send the prompt to the model.



    The app shows the prompt and the response from the model in the **Preview** pane.
12. Experiment with influencing the model response by changing the inference parameters in the **Parameters** section. For example, increase the randomness of the songs in the playlist by increasing the **Temperature** inference parameter.

    The inference parameters you can change are *Temperature*, *Top P*, and *Top K*. Not all models support each of these inference parameters. For more information, see Inference parameters.

13. Choose **Reset** to clear the conversation.
14. Change the system prompt to create a playlist of songs for a different genre of music and show length. For example, enter *You are an app that creates 3 hour long playlists for a radio station that plays ambient music.*
15. Choose **Save** to save your app.
16. (Optional) Share your project with other by following the instructions at Sharing an Amazon Bedrock Studio project.
17. Next step: Add a guardrail to your app by following the instructions at Adding a guardrail to your app.

Inference parameters

Inference parameters are values that you can adjust to limit or influence the model response. The following categories of parameters are commonly found across different models.

Randomness and diversity

For any given sequence, a model determines a probability distribution of options for the next token in the sequence. To generate each token in an output, the model samples from this distribution. Randomness and diversity refer to the amount of variation in a model's response. You can control these factors by limiting or adjusting the distribution. Foundation models typically support the following parameters to control randomness and diversity in the response.

- **Temperature**– Affects the shape of the probability distribution for the predicted output and influences the likelihood of the model selecting lower-probability outputs.
  - Choose a lower value to influence the model to select higher-probability outputs.
  - Choose a higher value to influence the model to select lower-probability outputs.

  In technical terms, the temperature modulates the probability mass function for the next token. A lower temperature steepens the function and leads to more deterministic responses, and a higher temperature flattens the function and leads to more random responses.
- **Top K** – The number of most-likely candidates that the model considers for the next token.
  - Choose a lower value to decrease the size of the pool and limit the options to more likely outputs.
  - Choose a higher value to increase the size of the pool and allow the model to consider less likely outputs.

  For example, if you choose a value of 50 for Top K, the model selects from 50 of the most probable tokens that could be next in the sequence.
- **Top P** – The percentage of most-likely candidates that the model considers for the next token.
  - Choose a lower value to decrease the size of the pool and limit the options to more likely outputs.
  - Choose a higher value to increase the size of the pool and allow the model to consider less likely outputs.

  In technical terms, the model computes the cumulative probability distribution for the set of responses and considers only the top P% of the distribution.

  For example, if you choose a value of 0.8 for Top P, the model selects from the top 80% of the probability distribution of tokens that could be next in the sequence.

The following table summarizes the effects of these parameters.

Parameter	Effect of lower value	Effect of higher value
Temperature	Increase likelihood of higher-probability tokens Decrease likelihood of lower-probability tokens	Increase likelihood of lower-probability tokens Decrease likelihood of higher-probability tokens
Top K	Remove lower-probability tokens	Allow lower-probability tokens
Top P	Remove lower-probability tokens	Allow lower-probability tokens

As an example to understand these parameters, consider the example prompt **`I hear the hoof beats of "`**. Let's say that the model determines the following three words to be candidates for the next token. The model also assigns a probability for each word.

```
{
    "horses": 0.7,
    "zebras": 0.2,
    "unicorns": 0.1
}
```

- If you set a high **temperature**, the probability distribution is flattened and the probabilities become less different, which would increase the probability of choosing "unicorns" and decrease the probability of choosing "horses".
- If you set **Top K** as 2, the model only considers the top 2 most likely candidates: "horses" and "zebras."
- If you set **Top P** as 0.7, the model only considers "horses" because it is the only candidate that lies in the top 70% of the probability distribution. If you set **Top P** as 0.9, the model considers "horses" and "zebras" as they are in the top 90% of probability distribution.

Adding a guardrail to your app

Guardrails for Amazon Bedrock lets you implement safeguards for your Amazon Bedrock Studio app based on your use cases and responsible AI policies. You can create multiple guardrails tailored to different use cases and apply them across multiple foundation models, providing a consistent user experience and standardizing safety controls across generative AI apps. You can configure denied topics to disallow undesirable topics and content filters to block harmful content in the prompts you send to a model and to the responses you get from a model. You can use guardrails with text-only foundation models.

Topics

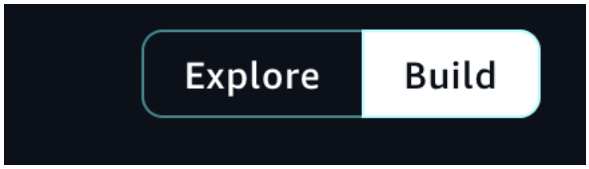
Adding a guardrail

This procedure shows you how to use a guardrail to safeguard the app you created in Creating an Amazon Bedrock Studio app. The guardrail prevents inappropriate language in song titles and filters out unwanted music genres.

If you just completed Creating an Amazon Bedrock Studio app, Bedrock Studio should be open on the app page and you don't need to do steps 1-3.

To add a guardrail to an Amazon Bedrock Studio app

1. Open your Amazon Bedrock Studio workspace.
2. Make sure that you are in Build mode, by choosing **Build** at the top right of the page.



3. Open the app that you created in Creating an Amazon Bedrock Studio app, or create a new app.
4. In the app pane, choose **Guardrails** and then **Create new guardrail**.
5. Enter *Clean_song_titles* in **Guardrail name**.
6. Enter *Detects songs that have inappropriate song titles* in **Guardrail description**.

7. In **Content filters** make sure **Enable content filters** is checked. For more information, see Content filters.
8. In **Filter for prompts** make sure the filter for each category is set to **High**.
9. Make sure **Apply the same filters for responses** is checked.
10. In **Blocked messsaging** do the following.
    a. Enter *Sorry, your prompt contained inappropriate text.* in **Blocked messaging for prompts**.
    b. Enter *Sorry, but I can't respond with information that contains inappropriate text.* in **Blocked messaging for responses**.
11. Choose **Create** to create the guardrail.
12. In the app pane, make sure that the selected guardail in **Guardrails** is the guardrail you just created.
13. Test the guardrail by entering *Create a list of 10 songs where each song has a swear word in the title.* In the prompt edit box.
14. Choose the run button to send the prompt to the model. The model should respond with the message **Sorry, but I can't respond with information that contains inappropriate text.**
15. Use a denied topic filter to prevent requests for music from a specific music genere. For information about denied topics, see Denied topics.

    To add the filter, do the following.
    a. In the **Guardrails** section of the app pane, select the guardrail and choose **Preview**.
    b. Choose **Edit** to edit the guardrail.
    c. In **Denied topics** choose **Add topic**.
    d. Enter *heavy metal* in **Name**.
    e. Enter *Avoid mentioning songs that are from the heavy metal genre of music.* in **Definition for topic**.
    f. Enter *Create a playlist heavy metal songs* in **Sample phrases - optional**.
    g. Choose **Add phrase**.
    h. Choose **Save**.
    i. Choose **Update** to update the guardrail.
    j. Test the guardrail by entering *Create a list of heavy metal songs.* in the prompt edit box.

k. Choose the run button to send the prompt to the model. The model should respond with the message **Sorry, but I can't respond with information that contains inappropriate text**.

16. Next step: Add a knowldege base to your app by following the instructions at Adding a data source to your app.

Guardrail policies

A guardrail consists of the following policies to avoid content that falls into undesirable or harmful categories.

- Content filters – Adjust filter strengths to filter input prompts or model responses containing harmful content.
- Denied topics – You can define a set of topics that are undesirable in the context of your app. These topics will be blocked if detected in user queries or model responses.

Topics

- Content filters
- Denied topics

Content filters

Guardrails in Bedrock Studio support the following content filters to detect and filter harmful user inputs and FM-generated outputs.

- **Hate** – Describes language or a statement that discriminates, criticizes, insults, denounces, or dehumanizes a person or group on the basis of an identity (such as race, ethnicity, gender, religion, sexual orientation, ability, and national origin).
- **Insults** – Describes language or a statement that includes demeaning, humiliating, mocking, insulting, or belittling language. This type of language is also labeled as bullying.
- **Sexual** – Describes language or a statement that indicates sexual interest, activity, or arousal using direct or indirect references to body parts, physical traits, or sex.
- **Violence** – Describes language or a statement that includes glorification of or threats to inflict physical pain, hurt, or injury toward a person, group or thing.

Content filtering depends on the confidence classification of user inputs and FM responses across each of the four harmful categories. All input and output statements are classified into one of four confidence levels (NONE, LOW, MEDIUM, HIGH) for each harmful category. For example, if a statement is classified as *Hate* with HIGH confidence, the likelihood of the statement representing hateful content is high. A single statement can be classified across multiple categories with varying confidence levels. For example, a single statement can be classified as *Hate* with HIGH confidence, *Insults* with LOW confidence, *Sexual* with NONE confidence, and *Violence* with MEDIUM confidence.

For each of the harmful categories, you can configure the strength of the filters. The filter strength determines the degree of filtering harmful content. As you increase the filter strength, the likelihood of filtering harmful content increases and the probability of seeing harmful content in your app reduces. The following table shows the degree of content that each filter strength blocks and allows.

Filter strength	Blocked content confidence	Allowed content confidence
None	No filtering	None, Low, Medium, High
Low	High	None, Low, Medium
Medium	High, Medium	None, Low
High	High, Medium, Low	None

Denied topics

Guardrails can be configured with a set of denied topics that are undesirable in the context of your generative AI app. For example, a bank may want their online assistant to avoid any conversation related to investment advice or engage in conversations related to fraudulent activities such as money laundering.

You can define up to five denied topics. Input prompts and model completions will be evaluated against each of these topics. If one of the topics is detected, the blocked message configured as part of the guardrail will be returned to the user.

Denied topics can be defined by providing a natural language definition of the topic along with a few optional example phrases of the topic. The definition and example phrases are used to detect if an input prompt or a model completion belongs to the topic.

Denied topics are defined with the following parameters.

- Name – The name of the topic. The name should be a noun phrase. Don't describe the topic in the name. For example:
  - **`Investment Advice`**
- Definition – Up to 200 characters summarizing the topic content. The description should describe the content of the topic and its subtopics.

  > **Note**
  >
  > For best results, adhere to the following principles:
  >
  > - Don't include examples or instructions in the description.
  > - Don't use negative language (such as "don't talk about investment" or "no content about investment").

  The following is an example topic description that you can provide:

  - **`Investment advice refers to inquires, guidance or recommendations regarding the management or allocation of funds or assets with the goal of generating returns or achieving specific financial objectives.`**
- Sample phrases – A list of up to five sample phrases that refer to the topic. Each phrase can be up to 1,000 characters. An sample is a prompt or continuation that shows what kind of content should be filtered out. For example:
  - **`Is investing in the stocks better than bonds?`**
  - **`Should I invest in gold?`**

Adding a data source to your app

You can use your own data into your application by adding documents or a Knowledge Base to your app. Doing this allows your app to access to information that is only available to you. In Bedrock Studio you can bring your own data, either as document, or by creating a Knowledge Base.

A document is a file that contains information that you want the model to use when generating a response. By using a document as a data source, your app users can chat with a document. For example, they can use a document to answers questions, make an analysis, create a summary,

itemize fields in a numbered list, or rewrite content. Amazon Bedrock Studio doesn't store your document or its data after use.

The document file or the Knowledge Base source file must be in PDF, MD, TXT, DOC, DOCX, HTML, CSV, XLS or XLSX format. The maximum file size is 10 MB.

You can also use a Knowledge Base to store your data. A Knowledge Base provides you the capability of amassing data sources into a repository of information. With Knowledge Bases, you can easily build an app that takes advantage of *retrieval augmented generation (RAG)*, a technique in which the retrieval of information from data sources augments the generation of model responses. You can only access Knowledge Bases that you create within Amazon Bedrock Studio. You can't access Knowledge Bases that you create in Amazon Bedrock.

In this topic, you update the app you created in Creating an Amazon Bedrock Studio app to use a CSV document as a document data source. The CSV file includes information about bands that don't have public meta data such song length, or music genre. The user can use the app to create a playlist for a radio show based on criteria such as song length or music genre.

If you just completed Creating an Amazon Bedrock Studio app, Bedrock Studio should be open on the app page and you don't need to do steps 1-3.

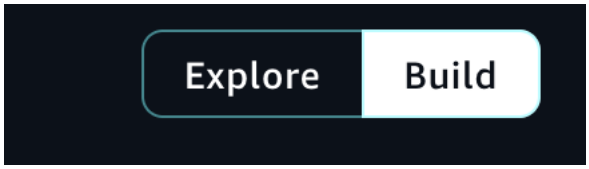
To add your own data to an Amazon Bedrock Studio app

1. Create a CSV file name *songs.csv* and fill with the following ficticious CSV data.

```
song,artist,genre,length-seconds
"Celestial Odyssey","Starry Renegades","Cosmic Rock",240
"Neon Rapture","Synthwave Siren","Synthwave Pop",300
"Wordsmith Warriors","Lyrical Legions","Lyrical Flow",180
"Nebula Shredders","Galactic Axemen","Cosmic Rock",270
"Electro Euphoria","Neon Nomads","Synthwave Pop",210
"Rhythm Renegades","Percussive Pioneers","Lyrical Flow",240
"Stardust Rift","Cosmic Crusaders","Cosmic Rock",180
"Synthwave Serenade","Electro Enchanters","Synthwave Pop",300
"Lyrical Legends","Rhyme Royale","Lyrical Flow",240
"Supernova Shredders","Amplified Ascension","Cosmic Rock",300
"Celestial Chords","Ethereal Echoes","Cosmic Rock",240
"Neon Nirvana","Synthwave Sirens","Synthwave Pop",270
"Verbal Virtuoso","Lyrical Maestros","Lyrical Flow",210
"Cosmic Collision","Stellar Insurgents","Cosmic Rock",180
"Pop Paradox","Melodic Mavericks","Synthwave Pop",240
```

```
"Flow Fusion","Verbal Virtuosos","Lyrical Flow",300
"Shredding Shadows","Crimson Crusaders","Cosmic Rock",270
"Synth Serenade","Electro Enchanters","Synthwave Pop",180
"Wordsmith Warlords","Lyrical Legionnaires","Lyrical Flow",240
"Sonic Supernova","Amplified Ascension","Cosmic Rock",210
"Celestial Symphony","Ethereal Ensemble","Cosmic Rock",300
"Electro Euphoria","Neon Nomads","Synthwave Pop",180
"Lyrical Legends","Rhyme Royale","Lyrical Flow",270
"Crimson Crescendo","Scarlet Serenaders","Cosmic Rock",240
"Euphoric Tides","Melodic Mystics","Synthwave Pop",210
"Rhythm Renegades","Percussive Pioneers","Lyrical Flow",180
"Cosmic Collision","Stellar Insurgents","Cosmic Rock",300
"Stardust Serenade","Celestial Crooners","Synthwave Pop",240
"Wordsmith Warriors","Lyrical Legions","Lyrical Flow",270
"Sonic Supernova III","Amplified Ascension","Cosmic Rock",180
```

2. Open your Amazon Bedrock Studio workspace.
3. Make sure that you are in Build mode, by choosing **Build** at the top right of the page.



4. Open the app that you created in Creating an Amazon Bedrock Studio app.
5. Disable any guardrails that you have enabled by choosing **None selected** in **Guardrails**.
6. In **Data**, choose **Use single file only**.
7. In **Add data sources**, choose **Click to upload** and upload the CSV file you created in step 1. Alternatively, add your source documents by dragging and dropping the CSV from your computer.

8. Test the data source by entering *Create a playlist of songs in the Lyrical Flow genre* in the prompt edit box.
9. Choose the run button to send the prompt to the model. The model should respond with a playlist of songs from the Lyrical Flow genre that the CSV file contains.

Adding a function call to an Amazon Bedrock Studio app

Amazon Bedrock Studio functions let a model include information that it has no previous knowledge of in its response. For example, you can use a function to include dynamic information in a model's response such as a weather forecast, sports results, or traffic conditions.

To use a function in Amazon Bedrock Studio you add a function component to your app. As part of the function, you define an OpenAPI schema for the API that you want the model to call. You also specify how to authorize the call to the API. When a model receives a prompt, it uses the schema and the prompt to determine if an API should be called and the parameters that the API should receive. If the API is called, the response from the model includes the output from the API.

APIs that you call in a function must have a response size that is less than 20K.

If you add a function to an existing app, you need to also update the app's system prompt. The system prompt needs to be at least 40 characters long and should mention the new skills that the new function introduces.

Amazon Bedrock Studio has the following requirements for the schema that you use to create a function.

- The function schema must be OpenAPI version 3.0.0.
- The function can have no authorization or authorization type apiKey.
- You can have 0 or 1 server Url.
- Parameters (**parameter.in**) must be pass passed through query or path. You can't use cookies or headers to pass parameters.
- Parameters (**parameter schema type**) must be primitive types, arrays, or objects (one-level JSON). You can't pass complext nested objects.
- Parameter content (**parameter.content**) is mutually exclusive with the schema. Schema is more commonly used. Use content only for more complex types, or for complex serialization scenarios that are not covered by style and explode.
- Parameter **style** and **explode** values. `form` and `true` for query, `simple` and `false` for paths). For more information, see Parameter Serialization.
- Request body content must be passed as `application/json`.
- The schema can have up to 5 APIs and an app can use up to 5 APIs across all functions. For the model to correctly choose function, it is important to provide detailed descriptions of the API, including parameters, properties, and responses.

In this procedure, you add a function to the app that you created in Creating an Amazon Bedrock Studio app. so that users can get a list of the top 10 songs played on the radio station that day.

To add a function to an Amazon Bedrock Studio app

1. Create a HTTPS server that implements a `TopSongs` function. Make sure the function adheres to the following schema.

```
openapi: 3.0.0
info:
  title: Top Songs API
  description: API to retrieve the top 10 songs played today
  version: 1.0.0
```

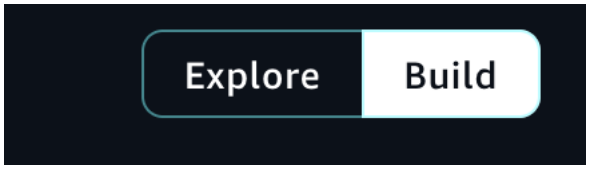
```
paths:
  /top-songs:
    get:
      summary: Get the top 10 songs played today
      description: >
        This endpoint returns an array of the top 10 songs played today,
        ordered by popularity. The first element in the array (index 0)
        represents the most popular song, and the last element (index 9)
        represents the 10th most popular song.
      responses:
        '200':
          description: Successful response
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/TopSongs'

components:
  schemas:
    TopSongs:
      type: array
      items:
        $ref: '#/components/schemas/Song'
      description: >
        An array containing the top 10 songs played today. The first element
        (index 0) is the most popular song, and the last element (index 9)
        is the 10th most popular song.
      example:
        - title: 'Song Title 1'
          artist: 'Artist Name 1'
          album: 'Album Name 1'
        - title: 'Song Title 2'
          artist: 'Artist Name 2'
          album: 'Album Name 2'
        # ... up to 10 songs

    Song:
      type: object
      properties:
        title:
          type: string
          description: The title of the song
        artist:
```

```
        type: string
        description: The name of the artist or band
      album:
        type: string
        description: The name of the album the song is from
    required:
      - title
      - artist
      - album
```

2. Open your Amazon Bedrock Studio workspace.
3. Make sure that you are in Build mode, by choosing **Build** at the top right of the page.



4. Open the app that you created in Creating an Amazon Bedrock Studio app.
5. In **Models**, choose a model that supports functions, such as *Anthropic Claude 2.1*.
6. In *Functions*, choose **Create new function**.
7. In the **Create function** pane, do the following.
   a. Enter *TopTenSongsToday* in **Function name**.
   b. Enter *Today's top 10 songs.* in **Function description**
   c. Enter the OpenAPI schema from step one in **Function schema**
   d. In **Authentication method** choose the authentication method for your HTTP server. You can choose between **No authentication** or **Authenticate using API Keys**. You can have a maximum of two keys. Enter your first key in the **API Key 1**. For more information, see API keys.
   e. In **API servers**, enter the URL for your server in **Server URL**. This value is autopopulated if the server URL is in the schema.
   f. Choose **Create** to create your function.
8. Test the function by doing the following.
   a. Enter *What are today's top 10 songs?* in the prompt edit box.
   b. Choose the run button to send the prompt to the model. The model should respond with the list of today's top 10 songs.

Managing your Amazon Bedrock Studio workspace

Here you learn how to manage the resources (projects, components, and models) in your workspace.

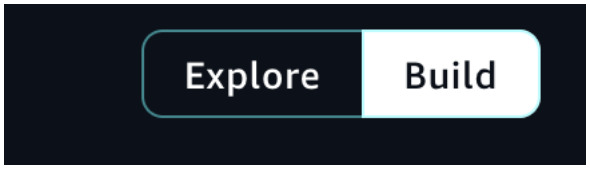
Topics

Creating an Amazon Bedrock Studio project

A Amazon Bedrock Studio project is where you organize the apps and components that you create. After you create a project you can start creating apps and components. For more information, Creating an Amazon Bedrock Studio app. You can also share your projects with other workspace members.

To create a project

1. Open your Amazon Bedrock Studio workspace.
2. Make sure that you are in Build mode, by choosing **Build** at the top right of the page.



3. Create a project for your app, by doing one of the following.
    a. If you are on on the **All projects** page, choose **Create new project**.
    b. If you are on an existing project page and want to ue a different project, Choose the **Project** dropdown list box and choose the project that you want to use, or create a new project.

Sharing an Amazon Bedrock Studio project

You can share a project with up to 20 workspace members. A workspace member can be a user or a group of users. A project can have up to 100000 users across up to 20 groups. To manage the members in your workspace, contact your administrator. For more information, see Add or remove workspace members.

Note that members that you share a project with have permissions to delete the project.

To share a project

1. Open your Amazon Bedrock Studio workspace.
2. Choose **Build**.
3. At the top of the page, choose the **Project** dropdown list box and choose **View all projects**.
4. On the card for the project that you want to share, choose the menu button and the choose **Share project**.
5. In **Member type** choose **Individual user** or **Group**, depending on the type of member that you want share the project with.
6. Search for the users or groups that you want to share the project by entering the user name or group in the **Search by alias to invite members** edit box.
7. In the drop down list box, the select the matching user name or group that want to share with. Bedrock Studio automatically adds the workspace member to the project.
8. Choose **Copy link** to copy the URL link that project members need to access the project.
9. Use email, or another secure communication method, to send the URL link to the members that you added to the project.

Stop sharing an Amazon Bedrock Studio project

You can stop sharing a project with members of your workspace.

To stop sharing a project

1. Open your Amazon Bedrock Studio workspace.
2. Choose **Build**.
3. At the top of the page, choose the **Project** dropdown list box and choose **View all projects**.

4. On the card for the project that you want to share, choose the menu button and the choose **Share project**.
5. In **All members with access**, choose the trash icon next to the user or group that you want to stop sharing with. Bedrock Studio studio automatically stops sharing the project to the member.

Deleting an Bedrock Studio project

You can delete projects that you have created and projects that others have shared to you. Before you can delete a project, you must first delete any associated apps and components.

To delete a Amazon Bedrock Studio project

1. Open your Amazon Bedrock Studio workspace.
2. Choose **Build**.
3. At the top of the page, choose the **Project** dropdown list box and choose the project that you want to delete.
4. Delete the components by doing the following.
    a. Select the first component in the **Components** list.
    b. Choose the meunu option and the select choose **Delete**.
    c. In the delete component dialog box, choose **Delete** to delete the component.
    d. Repeat these steps until you delete all components in the project.
5. Delete the apps by doing the following.
    a. Select the first app in the **Aps** list.
    b. Choose the trash button.
    c. In the delete app dialog box, choose **Delete** to delete the app.
    d. Repeat these steps until you delete all apps in the project.
6. Choose **Delete** to delete the project. If the project still has associated apps or components, Bedrock Studio shows a dialog box. Chose **Got it** to go back to the project page and delete the remaining apps and components.
7. Contact the project teams member to let them know the project has been deleted. Bedrock Studio doesn't automatically notify team members that the project has been deleted.

Adding apps or components to an Amazon Bedrock Studio project

You use a project to manage your apps and the components that your apps use. After you add an app, and optional components to a project, you can configue and experiment with the app. For more information, see the section called “Creating an app”.

To add an app or a component to a project

1. Open your Amazon Bedrock Studio workspace.
2. Choose **Build**.
3. At the top of the page, choose the **Project** dropdown list box and choose the project that you want to add an app or component to.
4. Add an app by choosing **Create app** in **Apps**.
5. Add a component by choosing **Create component** in **Components**.

Deleting apps and components from an Amazon Bedrock Studio project

A project can contain one or more apps or components. You can delete apps and components that you no longer need.

> **Note**
>
> If want to delete a project, you must first delete all apps and components in the project (Deleting an Bedrock Studio project).

To add an app or a component to a project

1. Open your Amazon Bedrock Studio workspace.
2. Choose **Build**.
3. At the top of the page, choose the **Project** dropdown list box and choose the project that you want to add an app or component to.
4. Delete a component by doing the following.

a. Select the component in the **Components** list.

b. Choose the meunu option and the select choose **Delete**.

c. In the delete component dialog box, choose **Delete** to delete the component.

5. Delete a app by doing the following.

a. Select the app in the **Aps** list.

b. Choose the trash icon.

c. In the delete app dialog box, choose **Delete** to delete the app.

Document history for the Amazon Bedrock Studio User Guide

- **Latest documentation update:** May 7th 2024

The following table describes important changes in each release of Amazon Bedrock Studio. For notification about updates to this documentation, you can subscribe to an RSS feed.

Change	Description	Date
Preview release	Preview release of Amazon Bedrock Studio.	May 7, 2024