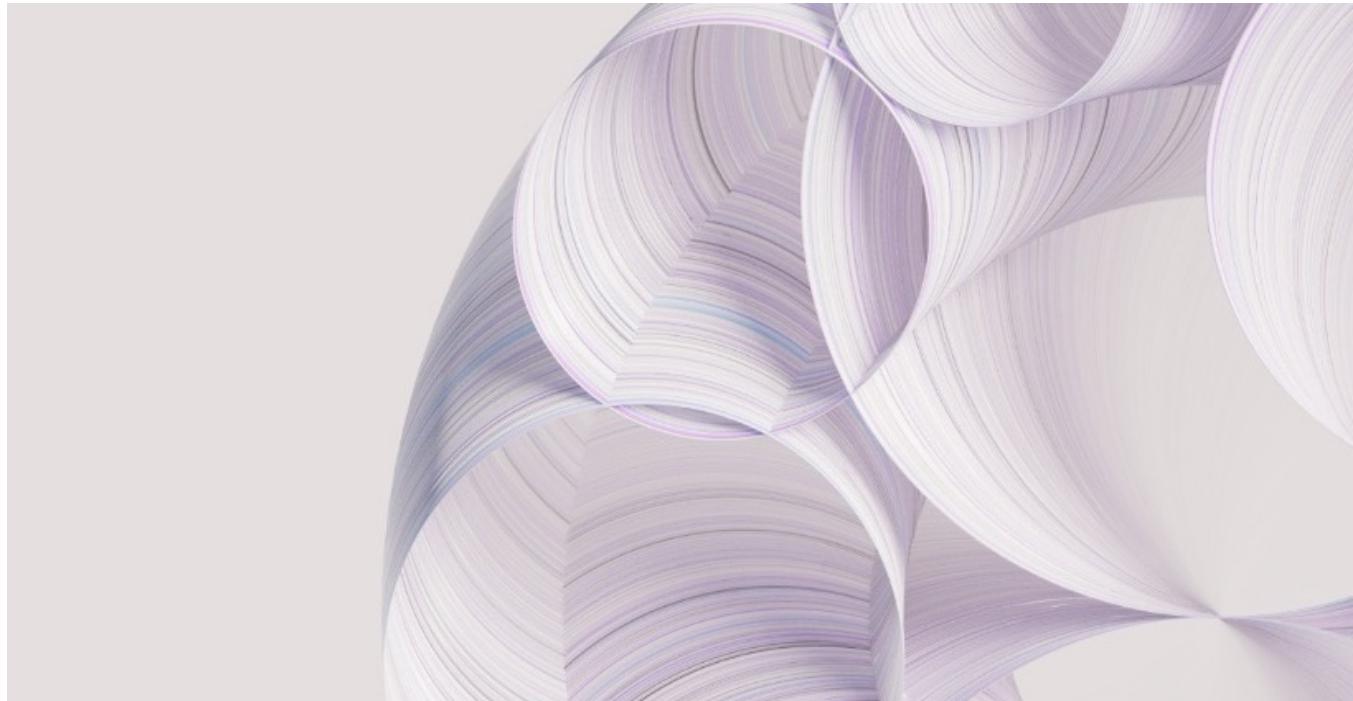


Level 3 Generative AI Models

Hands-on lab guide

Watsonx.ai



Felix Lee
felix@ca.ibm.com
Principal, Learning Content Development, Data and AI

Document edition: June 2025

LEGAL NOTICES

This information was developed for products and services offered in the USA. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation
North Castle Drive, MD-NC119 Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND,
EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT,
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in
certain transactions; therefore, this statement may not apply to you. This information could include technical inaccuracies or typographical
errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM
may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.
Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an
endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites
is at your own risk. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any
obligation to you. Information concerning non-IBM products was obtained from the suppliers of those products, their published
announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance,
compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to
the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the
examples include the names of individuals, companies, brands, and products. All names and references for organizations and other business
institutions used in this deliverable's scenarios are fictional. Any match with real organizations or institutions is coincidental. All names and
associated information for people in this deliverable's scenarios are fictional. Any match with a real person is coincidental.

TRADEMARKS

IBM, the IBM logo, and [ibm.com](#) are trademarks or registered trademarks of International Business Machines Corp., registered in many
jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks
is available on the web at "Copyright and trademark information" at <https://www.ibm.com/legal/copyright-trademark>.

*<Add any 3rd-party copyright information here for software or material of any 3rd party vendor that we are allowed to use in this lab or demo.
REMOVE this section if not required>*

© Copyright International Business Machines Corporation 2025. This document may not be reproduced in whole or in part without the prior
written permission of IBM.

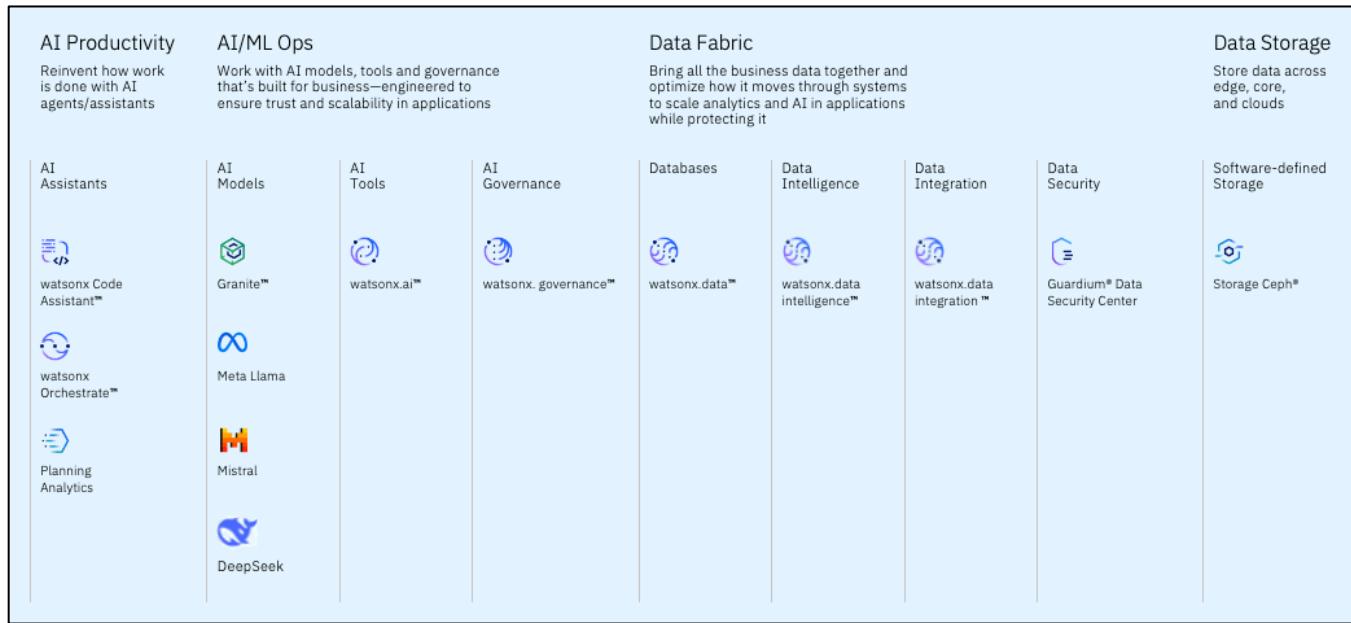
Table of Contents

Introduction	4
About this Lab	6
Disclaimer	6
Prerequisites.....	6
Logging into watsonx.ai.....	7
Watsonx.ai Foundation Models	14
Learning goals.....	14
Watsonx.ai Foundation Model Library	15
Section summary.....	28
Evaluating foundation model.....	29
Learning Goals	29
Evaluating model response and cost	29
Sub-section summary	39
Evaluating model response to tables in PDF.....	39
Subsection summary	46
Evaluating multi-model models	47
Subsection summary	59
Reasoning with Granite models.....	60
Learning Goals	60
Creating an API key	60
Adding a Jupyter Notebook to your project.....	62
Section summary.....	70
Custom Foundation Model (CFM)	71
Deploy on-demand	72
Summary.....	73
Appendix A: Handling Model Deprecation	74

Introduction

IBM Watsonx is a core component of the IBM Data Platform - an enterprise-ready AI and data platform designed for clients to access trusted and secure data to drive AI productivity.

Within the data platform is a complete set of AI features which includes AI assistants, AI models, AI tools, and AI governance. IBM Watsonx.ai is a complete AI development platform supporting traditional machine learning applications and generative AI applications.



This lab focuses on generative AI models.

The core of generative AI still evolves around foundation models. The performance of a simple prompt session, a Retrieval Augmented Generation (RAG) application, an AI assistant, or an AI agent depends heavily on the foundation model's capabilities. There are many factors in the selection of an FM:

- Functional capability: can the FM support the business use cases?
- If tuning is required (very likely, rarely will an FM work out-of-box), what resources are necessary to tune the model?
- Cost of using, tuning, and maintaining the model – the best model may also be the most expensive. Can a smaller model be trained to provide similar performance but at a much lower cost?
- Business requirements
 - Open source
 - Region and sovereignty, regulation requirements
 - Alignment with current business practices, policies, and strategy

Clients need to understand how different FMs behave in various use cases. It is useful to compare output, speed of output, cost of output, and other behaviors.

In general, a larger model can address a wider variety of tasks. However, larger models may not perform well (can take longer to provide a completion), can be too creative (in hallucinating), and more costly. Different models can have different strengths: in the richness of output style, in the ability to format outputs, in supporting coding tasks, translation tasks, or other generative AI use cases.

IBM's firm conviction is that no one model will rule them all, and this drives IBM's strategy in providing choices for clients: these include models in different sizes and from different vendors. In all cases, IBM takes a vigorous approach to selecting the best-of-breed models for clients.

Clients need to know the model options available and which ones can satisfy their functional and business needs.

About this Lab

This lab introduces the general features of watsonx.ai for working with models, along with the specific capabilities of various supported models. The sections of this lab include:

- General watsonx.ai model operations
- Working with and evaluating models from:
 - IBM (Granite reasoning and multimodal models)
 - Meta (Llama 3.3 and 4.0)
 - Mistral AI

Disclaimer

Watsonx.ai is being developed and released in an agile manner. Models are constantly being added, modified, and deprecated. The comparison tool is a self-service application and is being updated at a lower frequency. There will be times when new models are introduced but they are not reflected in the tool. You can expect to encounter some of the following:

- New models become available, and existing models are being deprecated (and eventually removed) from watsonx.ai. If a model used in this lab is getting deprecated or is no longer available when you perform the lab exercises, simply choose a similar model (from the same family, such as using another llama model to replace a deprecated llama model), or another model of similar size (or as close as possible). Be aware that your completion will look different if you do so.

Users should refer to the watsonx.ai documentation to find the [latest list](#). Note that some models may be available in specific Geo locations initially (for example, the elyza-japanese-llama-2-7b-instruct is initially only available in the Tokyo data center).

- The User Interface can look different.
- Deprecated (or removed) models that may cause error or failure in the tool (as they can no longer be called from watsonx.ai).
- Different comparison modes (such as image-based comparison) may not be available immediately for comparison when new multi-modal models become available.

Please raise questions on the #data-ai-demo-feedback Slack channel (IBMer only). IBM partners can request help at the Partner Plus Support (<https://www.ibm.com/partnerplus/support>) website.

Prerequisites

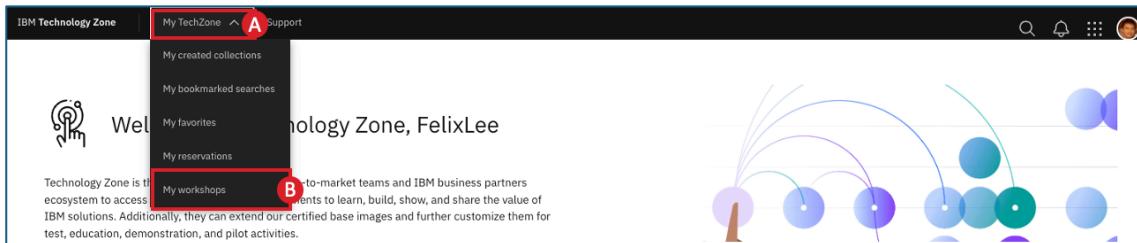
Important: You must use a TechZone account to perform this lab, or you will run into missing services or other issues. When you use TechZone, you will not (**and should not**) need to subscribe to any additional services (such as the free Lite Watson Machine Learning service).

Follow the instructions documented in the [Hands-on lab setup guide - watsonx.ai and watsonx.governance with Student Id TechZone](#) document to prepare your watsonx.ai environment for this lab.

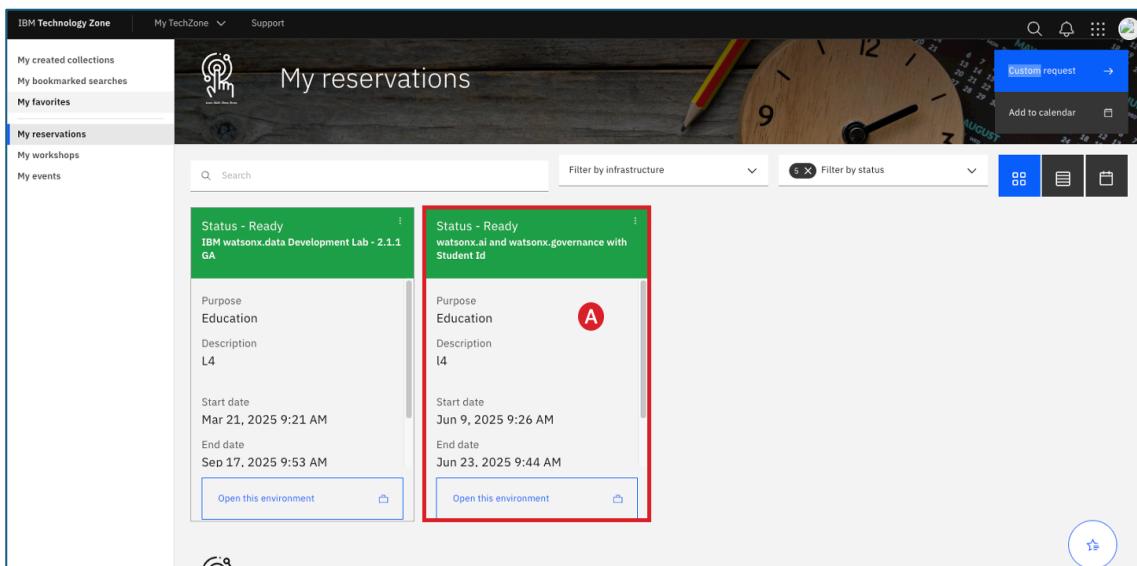
Logging into watsonx.ai

You must use the login URL, username, and password from your TechZone reservation to access your watsonx.ai environment. You can log in as follows.

1. Go to <https://techzone.ibm.com> and log in with your IBM id.
2. Click My TechZone (A), then My reservations (B).



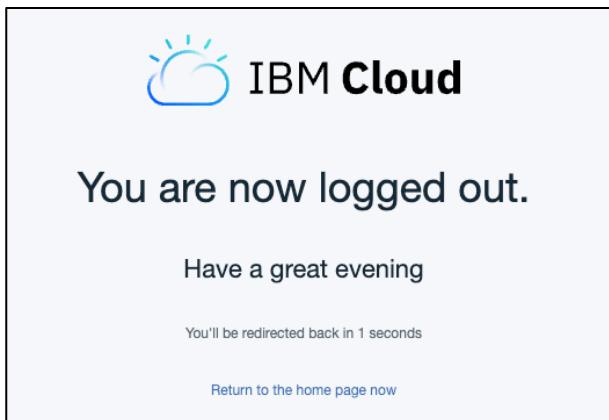
3. Find and click on the **watsonx.ai** and **watsonx.governance with Student Id** tile to open up your reservation detail page.



4. Click on the log out of IBM Cloud link (A).

The screenshot shows the 'Reservation Details' page. At the top, it says 'Reservation Details'. Below that is a section for 'watsonx Troubleshooting Guide' with a link to <https://www.ibm.com/support/pages/node/7182220>. A red box highlights a note: 'First make sure you are logged out of IBM Cloud.' Below that is another red box containing the URL <https://cloud.ibm.com/logout>. There is also a red circle with the letter 'A' on the right side of the page.

5. Another window tab opens and you see this message:



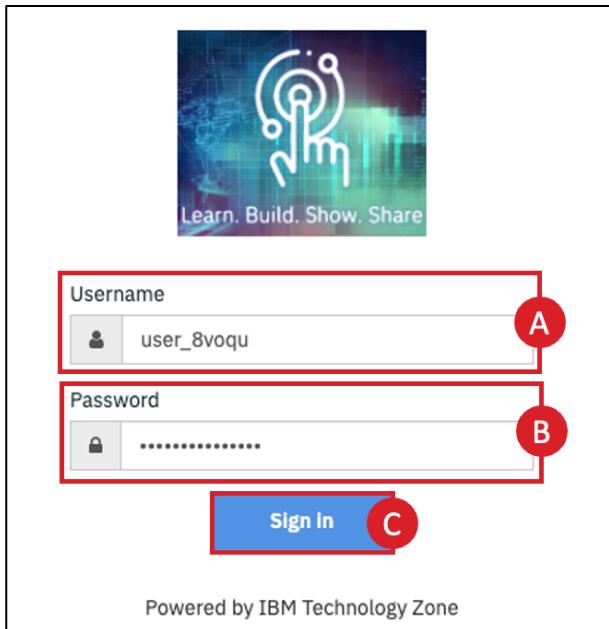
6. Close this window. On the reservation detail page, click on the provided **IBM Cloud Login** link (A):

A screenshot of a web page titled "Reservation Details". The page contains several links and text snippets. One snippet is a troubleshooting guide: "watsonx Troubleshooting Guide" followed by a blue link "https://www.ibm.com/support/pages/node/7182220". Another snippet says "First make sure you are logged out of IBM Cloud." followed by a blue link "https://cloud.ibm.com/logout". A third snippet shows an "IBM Cloud Login" button with a blue link below it: "https://cloud.ibm.com/authorize/itzwatsonx033". This final link is highlighted with a red rectangular box and has a red circle with the letter "A" in it.

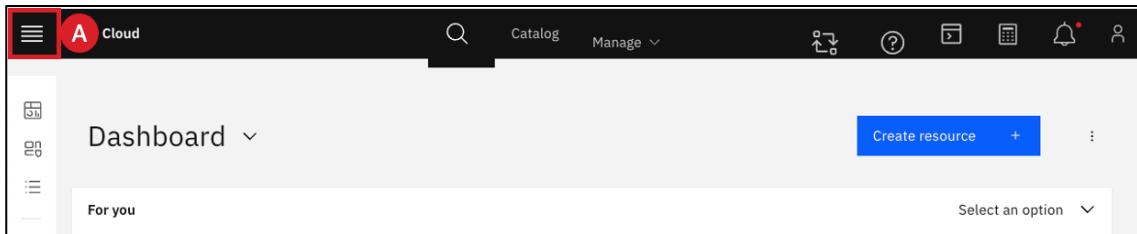
The following login page opens on a separate window.

A screenshot of a login page. At the top is a decorative graphic featuring a stylized antenna or signal tower against a blue and green background. Below the graphic is the slogan "Learn. Build. Show. Share". The main form area starts with a "Username" field, which includes a small user icon and a text input box. Next is a "Password" field, which includes a small lock icon and a text input box. Below these fields is a large blue "Sign in" button. At the bottom of the form is the text "Powered by IBM Technology Zone".

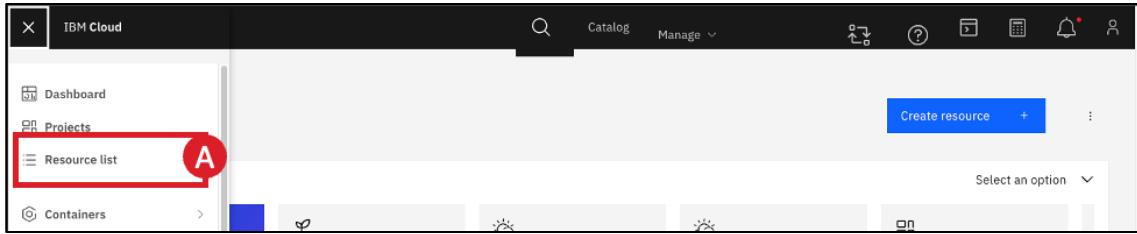
7. Use the **Username** (A) and **Password** (B) from your TechZone reservation detail. Then click **Sign in** (C).



8. The IBM Cloud Dashboard appears. Click the main menu(A), represented by four horizontal bars in the upper left corner.



9. A pull-down menu appears. Scroll down and click **Resource list** (A).



10. The **Resource list** page opens. Scroll down and click to expand the **AI/Machine Learning** (A) section. Then click on the **itz*** entry (B) associated with **watsonx.ai Studio** (itzws-110000f3jk-8voqu in this example).

Name	Group	Location	Product	Status	Tag
Watson Knowledge Catalog-itz		Dallas (us-south)	watsonx.data intelligence	Active	-
Watson OpenScale-itz		Dallas (us-south)	watsonx.governance	Active	-
Watson OpenScale-itz-fra		Frankfurt (eu-de)	watsonx.governance	Active	1
itzml-110000f3jk-8voqu	watsonxai-6825085cc9b90c82	Dallas (us-south)	watsonx.ai Runtime	Active	-
itzws-110000f3jk-8voqu	watsonxai-6825085cc9b90c82	Dallas (us-south)	watsonx.ai Studio	Active	-
watsonx.governance-x2		Sydney (au-syd)	watsonx.governance	Active	-

11. The **watsonx.ai Studio in Cloud Pak for Data and watsonx** page opens. Click **Launch in** (A) and then click **IBM watsonx** (B).

itzws-110000f3jk-8voqu ● Add tags 🔗

Details Actions ▼

Manage

Plan

watsonx.ai Studio
in
Cloud Pak for
Data and watsonx

Build and deploy machine learning models on either platform. Work with foundation models on watsonx as a Service.

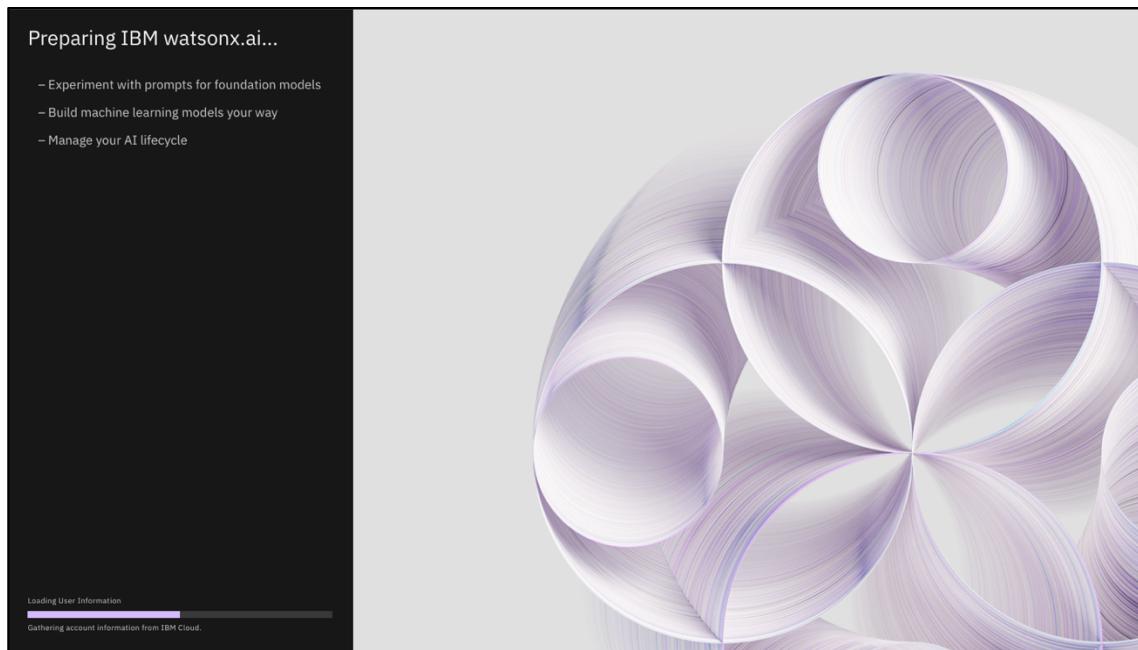
IBM watsonx.ai Studio is part of IBM Cloud Pak for Data and watsonx, and serves as the AI capability of the data fabric architecture.

Launch in A

IBM Cloud Pak for Data

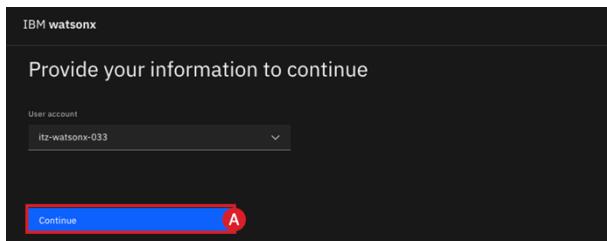
IBM watsonx B

12. The **Preparing IBM watsonx.ai** page opens. There is no action required – simply let the process finish.

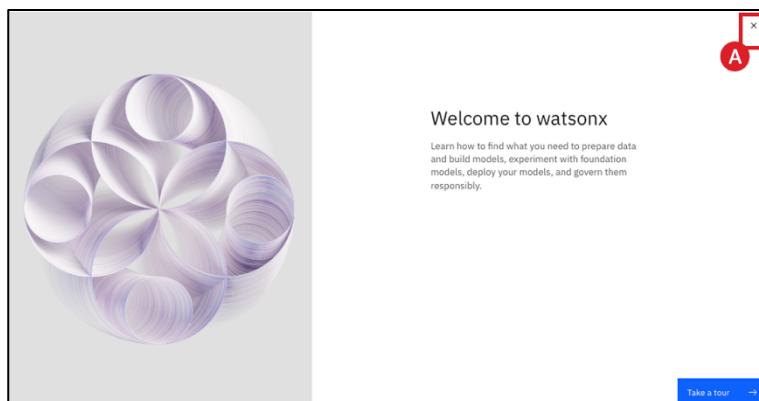


On rare occasions, you may get an error that the process cannot complete. Simply click **Retry**.

13. You may see the following. Click **Continue** (A) to proceed.



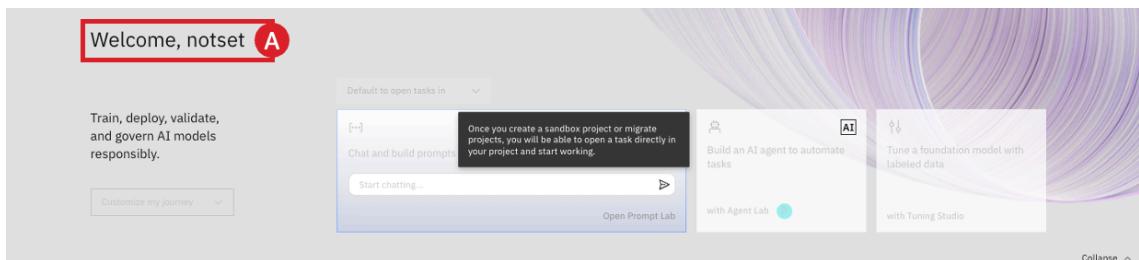
14. The **Welcome to watsonx** page opens. Click **x** (A) to close it.



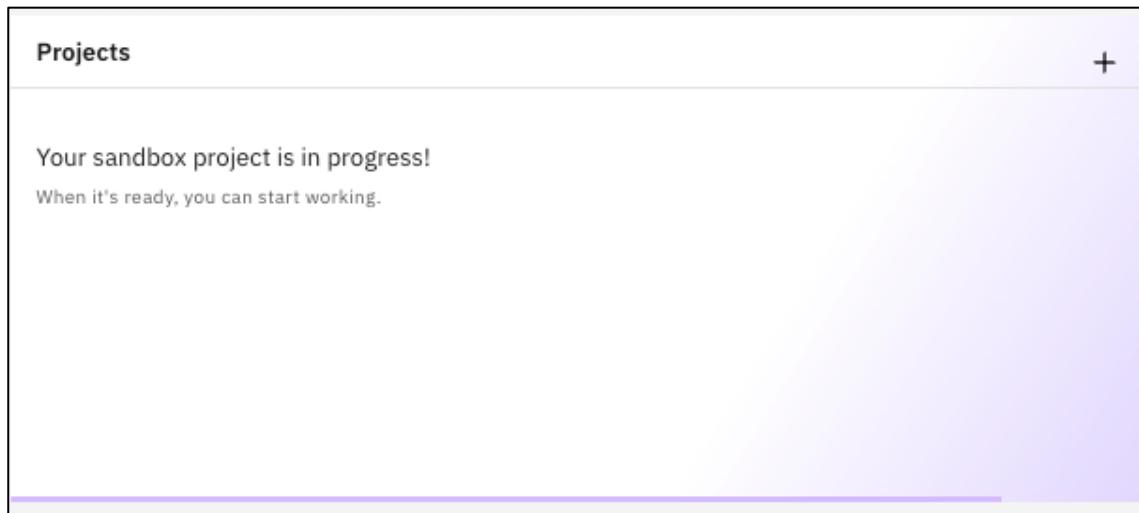
15. Click **x** (A) on the Dive deeper window to close it.



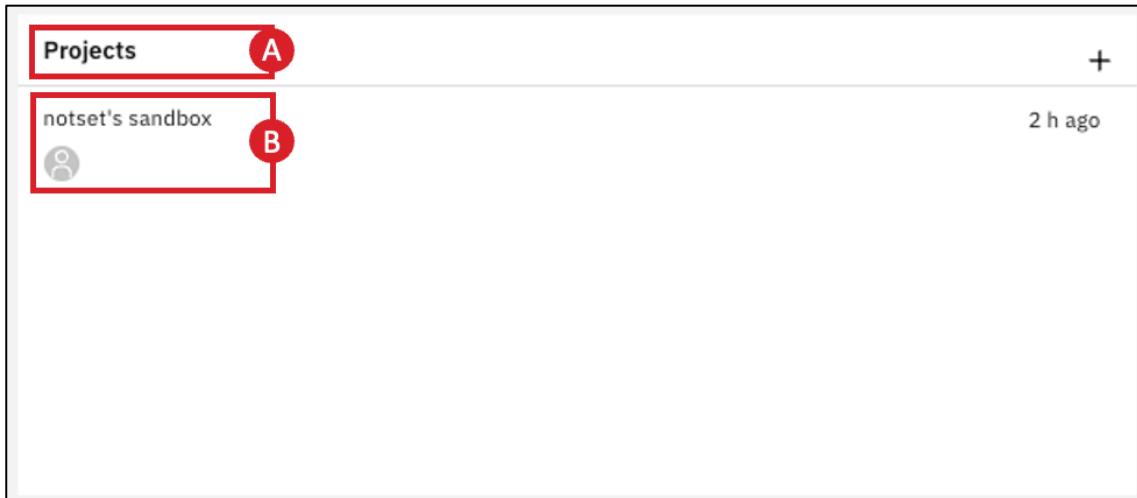
16. IBM watsonx.ai opens to the **Welcome, notset** (A) page. The name **notset** is the default name used in TechZone reservations.



17. If this is the first time you are logging in, watsonx.ai creates a sandbox project for you (skip the rest of the steps in this section if the project already exists). Scroll down and you will see that a sandbox project is being created automatically. You will see this:



18. The process should not take long. However, sometimes the User Interface (UI) may seem to be frozen. If you do not see completion or an error after a minute, reload your browser window (you may have to close the **Welcome to watsonx** and the **Dive deeper** windows again).
19. Scroll down to the **Projects** (A) area and you should see a **notset's sandbox** (B) project.



You are now ready to perform the exercises in this lab.

20. You can use the process in this section to log back into the watsonx.ai console, though creating a project is no longer required. Always use the link, username, and password provided by TechZone to access watsonx.ai.

Watsonx.ai Foundation Models

In this section, you will work with model cards and watsonx.ai filtering to identify the appropriate set of models for specific use cases.

Learning goals

In the early stages of generative AI (gen AI), the primary focus was on prompt engineering. When foundation models (FMs) gained widespread attention through platforms like ChatGPT, the initial excitement centered on their ability to generate diverse outputs—ranging from answering questions and classifying content to composing emails, stories, and essays. Users often interact directly with the models.

Gen AI rapidly evolved to support enterprise use cases, with Retrieval-Augmented Generation (RAG) emerging as a common pattern. RAG enables enterprises to leverage internal data to train and tune models, generating content grounded in validated business data. Identifying the correct grounding data is critical. The FM remains essential, as both the prompt and the retrieved grounding data are passed into the FM to generate a response. Therefore, model selection is a key consideration.

AI assistants represent another important development in gen AI. IBM offers solutions such as watsonx Assistant for conversational AI and watsonx Code Assistant to enhance developer productivity. In these applications, users interact with the assistant rather than the underlying FM. However, assistants like the Code Assistant rely on domain-specific FMs to function effectively. In general, AI assistants are designed for task-specific support and require carefully selected models to ensure optimal performance.

Agentic AI represents a new frontier in gen AI. Unlike traditional prompt-based interactions or conversational assistants, AI agents are designed to act autonomously on behalf of users. These agents operate using a Think-Act-Observe (TAO) loop and can collaborate with other agents or invoke tools to complete complex tasks. Agentic AI applications showcase advanced reasoning, planning, and execution capabilities. Despite this autonomy, FMs remain the core engine behind these systems, powering logic, decision-making, and tool integration.

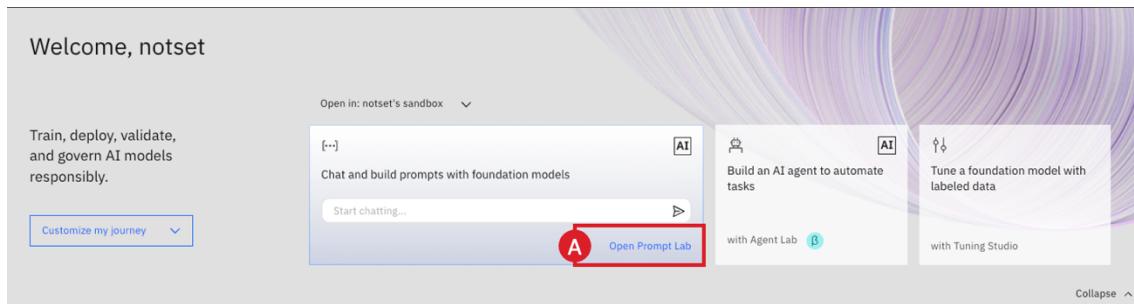
As gen AI applications mature, the FM becomes less visible to end users. However, it continues to serve as the foundation of these systems. Model selection and capability remain critical to clients, as the performance and cost of FMs directly influence the success of AI initiatives and investments.

In this section, you will:

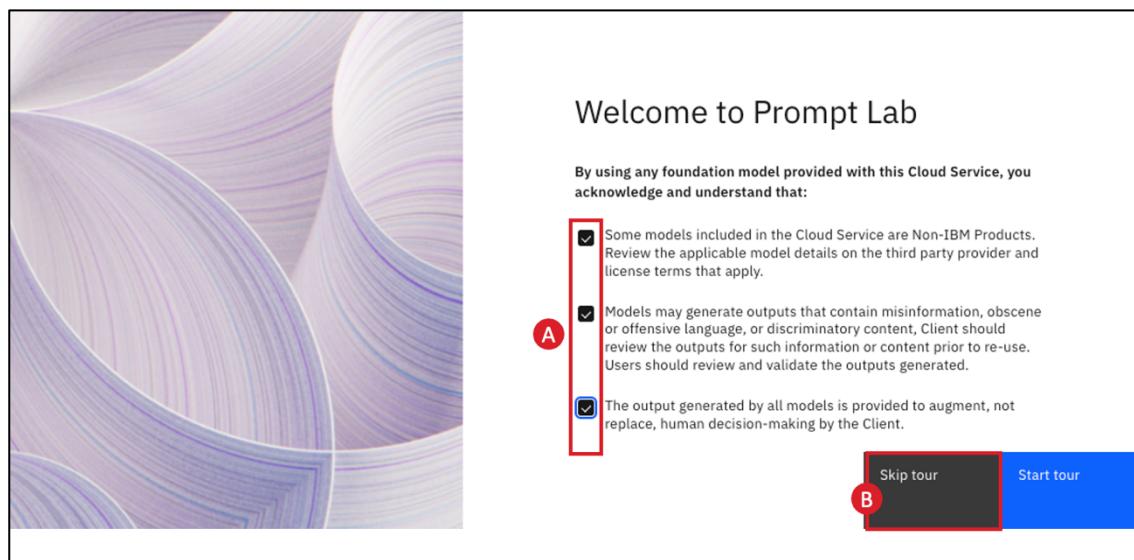
- Investigate the watsonx.ai foundation model library
- Learn how to extract key information from model cards
- Identify the most suitable foundation model (FM) candidates for specific use cases

Watsonx.ai Foundation Model Library

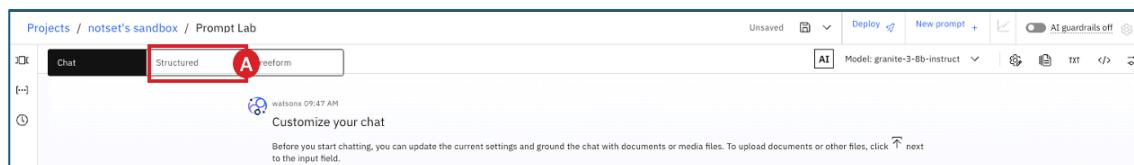
1. Log into the watsonx.ai console. Click **Open Prompt Lab** (A).



2. This opens the Prompt Lab. If you are doing this for the first time, you will see the **Welcome to Prompt Lab** window. Check all the boxes (A) and click **Skip tour** (B).



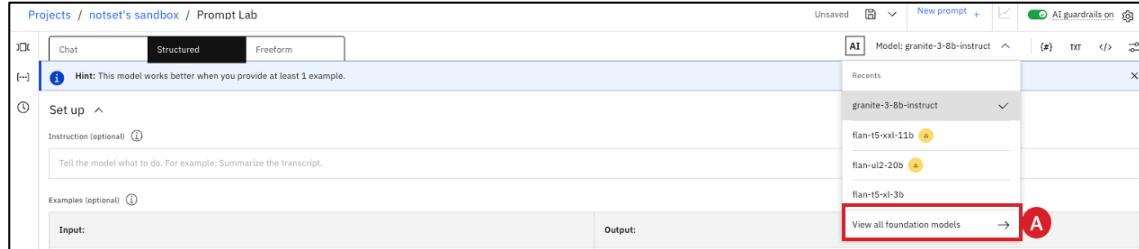
3. The **Prompt Lab** by default opens to the **Chat** tab. Select the **Structured** tab (A).



4. Click on the drop-down menu (A) beside the currently selected model.



5. On the pull-down menu, you may see other models (if this is not your first time using watsonx.ai), click on **View all foundation models** (A).



6. The **Select a foundation model** page opens. This shows the content of the **watsonx.ai foundation library**. Note the following:

- This list is dynamic, and IBM continuously updates the foundation model library. Your list may differ.
- Models marked with a yellow warning icon (A), such as **granite-20b-code-instruct**, are being deprecated. For watsonx.ai SaaS, clients should consult the [Foundation Models Lifecycle](#) for IBM's recommended replacements. See Appendix A for more on model deprecation.
- Models marked with a blue beta icon (B), such as **llama-4-scout-17b-16e-instruct**, are in beta. Clients are encouraged to experiment with these.

The screenshot shows the "Select a foundation model" page. It lists various foundation models from different providers and types. Several models are highlighted with red boxes and icons:

- granite-20b-code-instruct** (A): An IBM-trained, dense decoder-only model, marked with a yellow warning icon.
- llama-4-scout-17b-16e-instruct** (B): A Meta-trained, provided model, marked with a blue beta icon.
- llama-3-2-1b-vision-instruct**: A Meta-trained, provided model.
- llama-4-maverick-17b-128e-instruct-fp8**: A Meta-trained, provided model.
- llama-4-scout-17b-16e-instruct** (B): Another Meta-trained, provided model, marked with a blue beta icon.

 Other models listed include granite-3-8b-instruct, granite-3-13b-instruct-v2, granite-3-3b-instruct, granite-8b-code-instruct, granite-3-2-8b-instruct, granite-3-2b-instruct, granite-3-3-8b-instruct, granite-3-2-11b-vision-instruct, llama-3-1-70b-instruct, llama-3-2-1b-instruct, llama-3-2-3b-instruct, ptxrl-12b, and mistral-large.

7. You will see open-source models, IBM models, and any Custom Foundation Models (CFMs) that you may have added. The Granite models are developed by IBM. Click on the filter icon (A) to display a slide-out that shows possible filters you can use to narrow down the list of models (B).

You can filter by different categories:

- **By Task:** Options include **Chat**, **Agents**, **Question answering**, and so on. You can choose multiple tasks. If you were in the Chat tab when you chose to view the models, the Chat option is pre-selected, and you will not be able to select any other options.
- **By Provider:** The options are **Google**, **IBM**, **Meta**, and **Mistral AI**.
- **By Modalities:** Either **Language** or **Image**.
- **By Modal type:** This allows you to focus on **Provided model** versus **Tech Preview** model.

Note that the categories and the options for each category do change so don't be surprised if you have something different.

Model	Description	Provider	Type	Action
granite-3-8b-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
flan-t5-xl-3b	A pretrained T5 - an encoder-decoder model pre-trained on a mixture of supervised / unsupervised tasks converted into a text-to-text format.	Google	Provided model	Want to bring your own model?
flan-t5-xxl-11b	Flan-T5 is an 11 billion parameter model based on the Flan-T5 family.	Google	Provided model	Want to bring your own model?
flan-ul2-20b	Flan-U2 is an encoder-decoder model based on the T5 architecture and instruction-tuned using the Fine-tuned Language Net.	Google	Provided model	Want to bring your own model?
granite-13b-instruct-v2	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-20b-code-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-3-2-8b-instruct	Granite 3.2 B Instruct is a text-only model capable of reasoning which can be enabled or disabled by the user to use the right capability for the right use case.	IBM	Provided model	Want to bring your own model?
granite-3-2b-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-3-3-8b-instruct	Granite-3.3-B-Instruct is an IBM-trained, dense decoder-only model, which is particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-34b-code-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-3b-code-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?
granite-8b-code-instruct	The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	IBM	Provided model	Want to bring your own model?

8. Check **Agents** (A). The list shows (as of May 2025) five models (B):

- llama-3-2-11b-vision-instruct
- llama-3-2-90b-vision-instruct
- llama-3-3-70b-instruct
- mistral-large

Select a foundation model

To choose a model, review characteristics such as tasks that models perform. Compare model benchmarks with scores in the range 0–100. Higher scores are better.

All models Model benchmarks

Filter

Task

- Chat (22)
- Agents (4)
- Question answering (29)
- Summarization (28)
- Retrieval-Augmented Generation(RAG) (22)
- Classification (28)
- Generation (28)
- Code generation and conversion (22)
- Extraction (28)
- Translation (16)
- Function calling (14)
- Code generation (8)
- Code explanation (8)
- Code fixing (8)

Provider

- Google (3)
- Meta (4)

Agents

Want to bring your own model?

llama-3-2-11b-vision-instruct Llama-3-2-11b-vision-instruct is an auto-regressive language model that uses an optimized transformer architecture. Provider: Meta Type: Provided model	llama-3-2-90b-vision-instruct Llama-3-2-90b-vision-instruct is an auto-regressive language model that uses an optimized transformer architecture. Provider: Meta Type: Provided model
llama-3-3-70b-instruct This version of Llama-3-3-70b-instruct is also the FP8 quantized version of the original FP16 weights. Provider: Meta Type: Provided model	mistral-large Mistral Large, the most advanced Large Language Model (LLM) developed by Mistral AI, is an exceptionally powerful model. Thanks to its state-of-the-art reasoning capabilities it can be applied to any language-based task, including the most sophisticated ones. Provider: Mistral AI Type: Provided model

9. Click on the **llama-3-3-70b-instruct** tile.

10. The **llama-3-3-70b-instruct** model card opens. The model card provides useful information about the model. You can scroll down to read the details. Look at the top part of the model card and it shows the use case supported by this model (A).

llama-3-3-70b-instruct

Provider: Meta | Version: 3.3 | Type: Provided model

Question answering Summarization Retrieval-Augmented Generation(...) Classification Generation Code generation and conversion Extraction Translation Function calling A

Note that among common generative AI use cases, this Llama model supports **Retrieval Augmented Generation**, **Code generation and conversion**, and **Function calling** (especially important for agent development).

11. The **Model Architecture** provides usage information. The Llama 3.3 model takes **multilingual text** (A) as input and can produce **multilingual text or code** (B) as output. This means you can provide natural language input and ask this model to create code snippets.

Model Architecture: Llama 3.3 is an auto-regressive language model that uses an optimized transformer architecture. The tuned versions use supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align with human preferences for helpfulness and safety.								
	Training Data	Params	Input modalities	Output modalities	Context length	GQA	Token count	Knowledge cutoff
Llama 3.3	A new mix of publicly available online data.	70b	A Multilingual Text	B Multilingual Text and code	128k	Yes	15T+	Dec-23

Copilot usage **starts here**.

12. Scroll down for additional model details. For example, the **Multilinguality** section notes that Llama 3.3 supports eight languages: English, French, German, Hindi, Italian, Portuguese, Spanish, and Thai. Fine-tuning is required for use with unsupported languages

Clients should review the model card to understand its intended use, limitations, and other relevant information provided by the model provider. Note that the level of detail varies by provider.

13. Click **Back** (A) to return to the model list.

llama-3-3-70b-instruct

Provider: Meta | Version: 3.3 | Type: Provided model

Question answering Summarization Retrieval-Augmented Generation Classification Generation Code generation and conversion Extraction Translation Function calling

Note: This model is a Non-IBM Product governed by a third-party license that may impose use restrictions and other obligations. By using this model you agree to these terms. [Read terms](#)

New capabilities

Note that this release introduces new capabilities, including a longer context window, multilingual inputs and outputs and possible integrations by developers with third party tools. Building with these new capabilities requires specific considerations in addition to the best practices that generally apply across all Generative AI use cases.

Tool-use: Just like in standard software development, developers are responsible for the integration of the LLM with the tools and services of their choice. They should define a clear policy for their use case and assess the integrity of the third party services they use to be aware of the safety and security limitations when using this capability. Refer to the Responsible Use Guide for best practices on the safe deployment of the third party safeguards.

Multilinguality: Llama 3.3 supports 7 languages in addition to English: French, German, Hindi, Italian, Portuguese, Spanish, and Thai. Llama may be able to output text in other languages than those that meet performance thresholds for safety and helpfulness. We strongly discourage developers from using this model to converse in non-supported languages without implementing finetuning and system controls in alignment with their policies and the best practices shared in the Responsible Use Guide.

Evaluations

We evaluated Llama models for common use cases as well as specific capabilities. Common use cases evaluations measure safety risks of systems for most commonly built applications including chat bot, coding assistant, tool calls. We built dedicated, adversarial evaluation datasets and evaluated systems composed of Llama models and Llama Guard 3 to filter input prompt and output response. It is important to evaluate applications in context, and we recommend building dedicated evaluation dataset for your use case. Prompt Guard and Code Shield are also available if relevant to the application. Capability evaluations measure vulnerabilities of Llama models inherent to specific capabilities, for which we crafted dedicated benchmarks including long context, multilingual, tools calls, coding or memorization.

Back A Select model

14. On the Select a Foundation Model page, click the **granite-vision-3-2-2b** tile (A).

Select a foundation model

To choose a model, review characteristics such as tasks that models perform. Compare model benchmarks with scores in the range 0–100. Higher scores are better.

All models Model benchmarks

Search for a model or task

IBM	Provided model	IBM	Provided model	IBM	Provided model	IBM	Provided model
	granite-3-2b-instruct		granite-3-3-8b-instruct		granite-34b-code-instruct		granite-3b-code-instruct
Provider: IBM	Type: Provided model	Provider: IBM	Type: Provided model	Provider: IBM	Type: Provided model	Provider: IBM	Type: Provided model
The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.		The Granite-3.3-8b-Instruct is an IBM-trained, dense decoder-only models, which is particularly well-suited for generative tasks.		The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.		The Granite model series is a family of IBM-trained, dense decoder-only models, which are particularly well-suited for generative tasks.	
llama-2-13b-chat		llama-3-1-70b-instruct		llama-3-1-8b-instruct		llama-3-2-11b-vision-instruct	
Provider: Meta	Type: Provided model	Provider: Meta	Type: Provided model	Provider: Meta	Type: Provided model	Provider: Meta	Type: Provided model
llama-2-13b-chat is an auto-regressive language model that uses an optimized transformer architecture.		llama-3-1-70b-instruct is an auto-regressive language model that uses an optimized transformer architecture.		llama-3-1-8b-instruct is an auto-regressive language model that uses an optimized transformer architecture.		llama-3-2-11b-vision-instruct is an auto-regressive language model that uses an optimized transformer architecture.	

A

15. The **granite-vision-3-2-2b** model card opens. Note the following information at the top:

- **Provider and Version** (A): This Granite model is provided by IBM and is version 1:0.
- **Use case** (B): This Granite model supports Question answering.
- **Model summary** (C): This Granite model is “designed for visual document understanding, enabling automated context extraction from tables, charts, infographics, plots, diagrams, and more”. This model is best suited for answering questions about images, such as graphs, charts, tables, and so similar visual content.
- **Evaluation** (D): This Granite model card also provides information on benchmark evaluations of the Granite vision model compared to other vision models.

The screenshot shows a model card for 'granite-vision-3-2-2b'. At the top, there's a header with the model name and provider information (A). Below the header is a button labeled 'Question answering' (B). A note at the bottom of the card states 'Note: By using the model you agree to these terms.' with a 'Read terms' link. The main content area is titled 'Granite-Vision-3.2-2B'. It contains a 'Model Summary' section (C) which reads: 'Model Summary: Granite-vision-3.2-2b is a compact and efficient vision-language model, specifically designed for visual document understanding, enabling automated content extraction from tables, charts, infographics, plots, diagrams, and more. The model was trained on a meticulously curated instruction-following dataset, comprising diverse public datasets and synthetic datasets tailored to support a wide range of document understanding and general image tasks. It was trained by fine-tuning a Granite large language model with both image and text modalities.' Below the summary is an 'Evaluations' section (D) which includes a table comparing the model to others on benchmarks like DocVQA. The table has columns for Molmo-E, InternVL2, Phi3v, Phi3.5v, and Granite Vision. The 'Document benchmarks' row shows values of 0.66, 0.87, 0.87, 0.88, and 0.89 respectively.

	Molmo-E	InternVL2	Phi3v	Phi3.5v	Granite Vision
Document benchmarks					
DocVQA	0.66	0.87	0.87	0.88	0.89

16. Scrolling further down, the Granite model card includes guidance on using the Granite vision model with **Transformers** (A) and **vLLM** (B).

The screenshot shows the 'Usage with transformers' section (A) highlighted with a red box. It contains code for pip installation and usage examples. Below it, the 'Usage with vLLM' section (B) is also highlighted with a red box, showing a note about loading the model with vLLM.

```

granite-vision-3-2-2b
Provider: IBM | Version: 1.0 | Type: Provided model
Question answering
Note: By using the model you agree to these terms.
Read terms ×

Usage with transformers
First, make sure to build the latest versions of transformers: A

pip install transformers>=4.2.9

Then run the code:

from transformers import AutoProcessor, AutoModelForVision2Seq
from huggingface_hub import hf_hub_download
import torch

device = "cuda" if torch.cuda.is_available() else "cpu"

model_path = "lmsys/granite-vision-3.2-2b"
processor = AutoProcessor.from_pretrained(model_path)
model = AutoModelForVision2Seq.from_pretrained(model_path).to(device)

# prepare image and text prompt, using the appropriate prompt template
img_path = hf_hub_download(repo_id=model_path, filename='example.png')

conversation = [
    {
        "role": "user",
        "content": [
            {"type": "image", "url": img_path},
            {"type": "text", "text": "What is the highest scoring model on ChatQA and what is its score?"}
        ]
    }
]
inputs = processor.apply_chat_template(
    conversation,
    add_special_tokens=True,
    tokenize=True,
    return_tensors="pt",
    return_token_type_ids=True
).to(device)

# autogradically completes prompt
output = model.generate(inputs, max_new_tokens=100)
print(processor.decode(output[0], skip_special_tokens=True))

Usage with vLLM
The model can also be loaded with vLLM . First make sure to install the following libraries:

```

17. Scrolling further down, the model card provides links to **Notebooks** that demonstrate how to **fine-tune** the Granite model (A) or use it to support **Multi-Modal RAG** (B). Examples are an effective way to demonstrate how models can be used. Models often require tuning to optimize performance for various use cases. With the Granite Vision model, you can work with both text and image inputs. The sample notebooks provide insights into how to perform these tasks. Additional information is available on the model's architecture, training data (important for model transparency), and other relevant details.

The screenshot highlights the 'Fine-tuning' section (A) and the 'Use Granite Vision for MM RAG' section (B), both enclosed in red boxes. Below these, there is a 'Model Architecture' section, a 'Training Data' section, and a note about the training data sources.

Fine-tuning A
For an example of fine-tuning Granite Vision for new tasks refer to [this notebook](#).

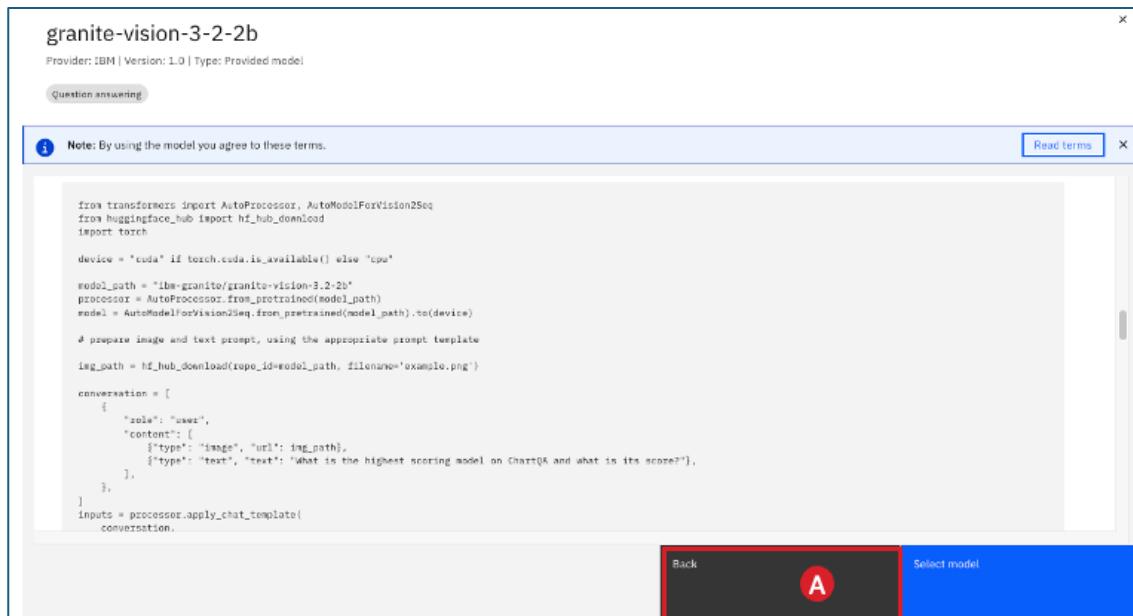
Use Granite Vision for MM RAG B
For an example of MM RAG using granite vision refer to [this notebook](#).

Model Architecture:
The architecture of granite-vision-3.2-2b consists of the following components:
(1) Vision encoder: SigLIP (https://huggingface.co/docs/transformers/en/model_doc/siglip).
(2) Vision-language connector: two-layer MLP with gelu activation function.
(3) Large language model: granite-3.1-2b-instruct with 128k context length (<https://huggingface.co/ibm-granite/granite-3.1-2b-instruct>).

We built upon LLaVA (<https://llava-vl.github.io>) to train our model. We use multi-layer encoder features and a denser grid resolution in AnyRes to enhance the model's ability to understand nuanced visual content, which is essential for accurately interpreting document images.

Training Data:
Overall, our training data is largely comprised of two key sources: (1) publicly available datasets (2) internally created synthetic data targeting specific capabilities including document understanding tasks. A detailed attribution of datasets can be found in the [technical report](#).

18. Click **Back** (A) to return to the model list.



19. Click the **funnel** icon (A) to open the slide-out of the model **Filter** panel (B).

Selecting the optimal combination of filter options enables you to narrow down the model list and identify the best candidate for your specific use case. You can filter by:

- **Task** – with these sub-categories
 - Typical generative AI use cases such as Question answering, Summarization, Classification, Generation, Extraction, and Chat
 - Code-related use cases such as Code generation and conversion, Code explanation, and Code fixing
 - Agents
 - Retrieval-Augmented Generation (RAG)
 - Function calling
- **Provider**: Google, IBM, Meta, or Mistral AI
- **Modalities**: Language or Image
- **Model type**: All, Provided model, or Tech Preview

Note that if you select multiple options from one category, the operation performs an OR action, not an AND. For example, if you select both Classification and Generation, the resulting list includes models that support either use case, but not necessarily both.

20. For example, a European client wants a local model that supports **Agents** (A) and **Function calling** (B). You know that **Mistral AI** (C) is a European-based company, and its models are widely accepted in the European Union (EU). If you choose the following options from the filter:

The screenshot shows the WatsonX AI Model Filter interface. At the top, there are two tabs: "All models" (selected) and "Model benchmarks". Below the tabs is a search bar with a magnifying glass icon and a placeholder "Search for a model or task". Underneath the search bar is a "Filter" section with a close button "X". The "Task" filter is expanded, showing various AI tasks with checkboxes. Two checkboxes are checked and highlighted with red boxes and letters A and B: "Agents" (A) and "Function calling" (B). Other tasks listed include Chat, Question answering, Summarization, Retrieval-Augmented Generation(RAG), Classification, Generation, Code generation and conversion, Extraction, and Translation. The "Provider" filter is also expanded, showing Google, IBM, Meta, and Mistral AI. The "Mistral AI" provider is checked and highlighted with a red box and letter C. The "Modality" filter is partially visible at the bottom.

The filter returns two models: **mistral-large** and **mistral-small-3-1-24b-instruct-2503** (A). Because this is an OR operation, you cannot assume both models can support Agents and Function calling.

Select a foundation model

To choose a model, review characteristics such as tasks that models perform. Compare model benchmarks with scores in the range 0–100. Higher scores are better.

All models Model benchmarks

Filter

Task

- Chat (29)
- Agents (5)
- Question answering (31)
- Summarization (31)
- Retrieval-Augmented Generation(RAG) (25)
- Classification (31)
- Generation (31)
- Code generation and conversion (25)
- Extraction (31)
- Translation (19)
- Function calling (16)
- Code generation (9)
- Code explanation (9)
- Code fixing (9)

Provider

- Google (3)
- IBM (12)
- Meta (12)
- Mistral AI (5)

Agents ✖ Function calling ✖ Mistral AI ✖ Reset filters

mistral-large A

Mistral Large, the most advanced Large Language Model (LLM) developed by Mistral AI, is an exceptionally powerful model. Thanks to its state-of-the-art reasoning capability, it can be applied to any language-based task, including the

Mistral AI B

This model is an instruction-finetuned version of: Mistral-Small-3.1-24B-Base-2503.

Provider: Mistral AI Type: Provided model

Want to bring your own model?

21. You can verify this by deselecting **Function calling** (A). Now only **mistral-large** (B) is returned.

Select a foundation model

To choose a model, review characteristics such as tasks that models perform. Compare model benchmarks with scores in the range 0–100. Higher scores are better.

All models Model benchmarks

Filter

Task

- Chat (29)
- Agents (5)
- Question answering (31)
- Summarization (31)
- Retrieval-Augmented Generation(RAG) (25)
- Classification (31)
- Generation (31)
- Code generation and conversion (25)
- Extraction (31)
- Translation (19)
- Function calling (16) A
- Code generation (9)
- Code explanation (9)
- Code fixing (9)

Provider

- Google (3)
- IBM (12)
- Meta (12)
- Mistral AI (5)

Agents ✖ Mistral AI ✖ Reset filters

mistral-large B

Mistral Large, the most advanced Large Language Model (LLM) developed by Mistral AI, is an exceptionally powerful model. Thanks to its state-of-the-art reasoning capability, it can be applied to any language-based task, including the

Mistral AI B

This model is an instruction-finetuned version of: Mistral-Small-3.1-24B-Base-2503.

Provider: Mistral AI Type: Provided model

Want to bring your own model?

This indicates that the **mistral-small-3-1-24b-instruct-2503** model is not for an **Agents** use case. It appears in Step 19 because it supports **Function calling**.

22. If you now **deselect Agents** (A) and **select Function calling** (B) again, you get both Mistral models again. This means that the **mistral-large** model - unlike the **mistral-small-3-1-24b-instruct-2503** model) - supports both **Agents** and **Function calling**.

The screenshot shows the "Select a foundation model" interface. The "Task" filter sidebar has "Agents" (A) selected and "Function calling" (B) deselected. The main search results show two models: "mistral-large" and "mistral-small-3-1-24b-instruct-2503". Both models are listed as "Provided model" by "Mistral AI".

23. When you select options across different categories, watsonx.ai performs an **AND** operation. For example, if you select **Function calling** from the **Task** (A) category and **IBM** from the **Provider** (B) category, the filter returns models that are provided by IBM **AND** support Function calling. In this case, only IBM Granite models are listed, and no Mistral models appear (C).

The screenshot shows the "Select a foundation model" interface. The "Task" filter sidebar has "Function calling" (A) selected and "Agents" deselected. The "Provider" filter sidebar has "IBM" (B) selected and "Google" deselected. The search results are highlighted with a red box and labeled (C), showing four IBM Granite models: "granite-3-8b-instruct", "granite-3-2-8b-instruct", "granite-3-2b-instruct", and "granite-3-3-8b-instruct". All four models are listed as "Provided model" by "IBM".

Section summary

- In this section, you explored how to navigate to the watsonx.ai Foundation Model Library. Keep in mind that the content of this library is constantly evolving. You should always check to see which new models are being added and which ones are being deprecated.
- Selecting a model in the library opens its model card (sometimes referred to as the fact sheet). You should examine this card to find vital information about the model, such as intended usage, limitations, supported use cases, evaluation and benchmark data, and more. Note that different vendors provide varying levels of transparency and detail.
- Use the filter to narrow down model candidates based on a client's use cases. Note that selecting multiple options within a single category results in an OR operation (the resulting list includes models that satisfy any of the selected options). Selecting options across different categories results in an AND operation.
- Clients can use filtering to identify the best set of candidates for their use cases. It is important to read the model cards to further understand each model's strengths and limitations.

Evaluating foundation model

The last section walks through how you can identify potential foundation models for your use case. In this section, you look at various ways you can test and evaluate different models. Evaluating and choosing the right foundation model is vital to the success of gen AI applications.

Learning goals

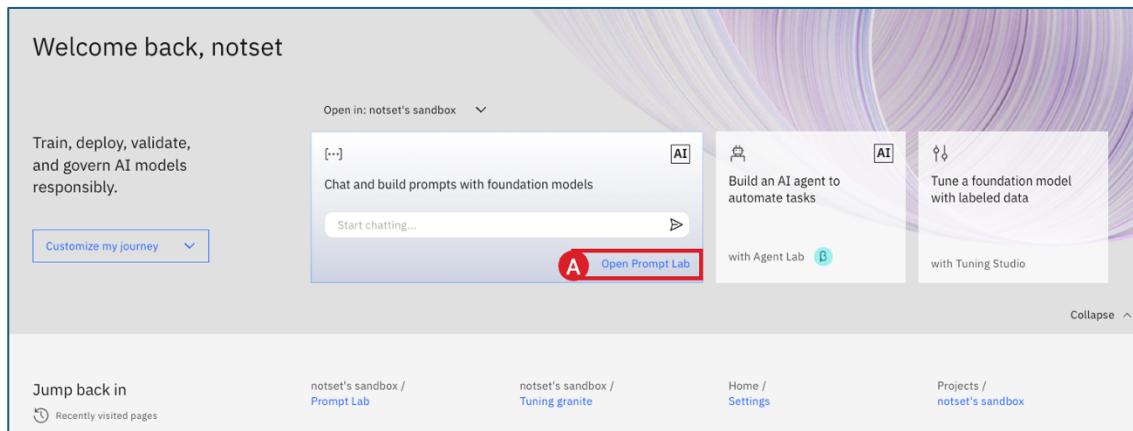
In this section, you look at different ways to evaluate models:

- Test different models with the same prompt and see how they respond. Compare the inference cost of different models. All things being equal, the cost of inference is often an important factor in model selection.
- Test and observe how models interpret data from a table.
- Test and observe how models interpret data from a picture.

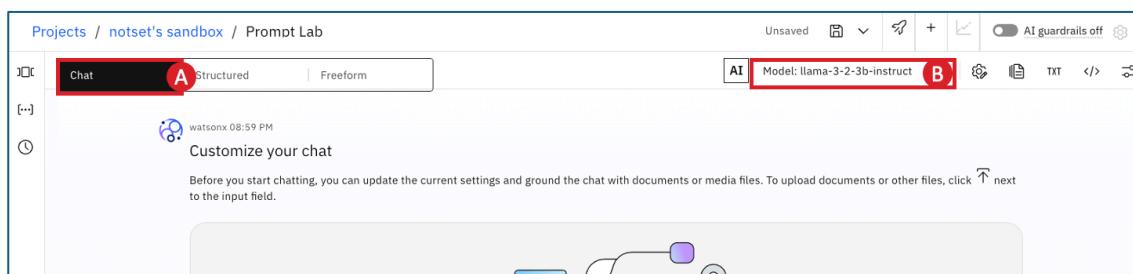
Evaluating model response and cost

You will get multiple results (the number depends on the filter setting) when you use filters to narrow down the set of potential model candidates. It is necessary to test with sample prompts and get an idea of the comparative performance cost of different models.

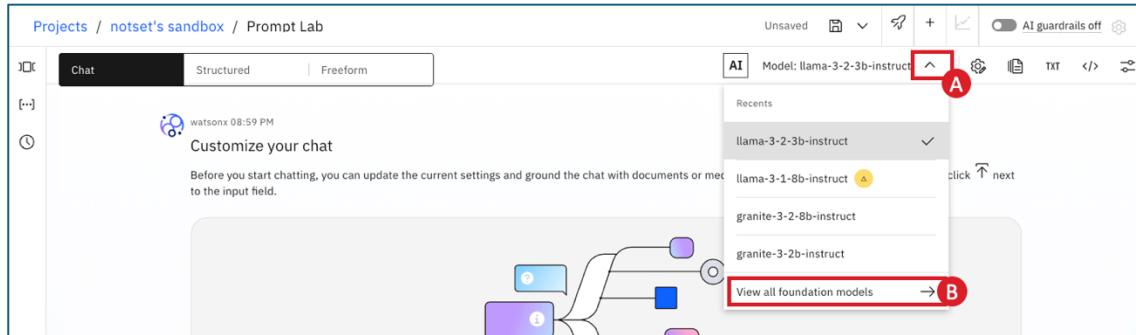
1. Log into watsonx.ai and click **Open Prompt Lab** (A).



2. The Prompt Lab opens up to the **Chat** (A) tab. The default model, **llama-3-2-3b-instruct** (B) in this example, depends on the last activity you performed and can be something different.



3. If you have a different model shown as the default, click the **downward arrow** (A) next to the model (it will turn into an upward arrow) to open the pull-down menu. Click **View all foundation models** (B).



4. On the **Select a foundation model** page, find and click on the **llama-3-2-3b-instruct** model tile (A).

The screenshot shows the "Select a foundation model" page. At the top, it says "Select a foundation model" and "To choose a model, review characteristics such as tasks that models perform. Compare model benchmarks with scores in the range 0–100. Higher scores are better." Below this, there are two tabs: "All models" (which is selected) and "Model benchmarks". There is also a search bar with the placeholder "Search for a model or task".

Model	Description	Provider	Type
llama-3-1-70b-instruct	Llama-3-1-70b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model
llama-3-1-8b-instruct	Llama-3-1-8b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model
llama-3-2-11b-vision-instruct	Llama-3-2-11b-vision-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model
llama-3-2-1b-instruct	Llama-3-2-1b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model
llama-3-2-3b-instruct (highlighted)	Llama-3-2-3b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model
llama-3-2-90b-vision-instruct	Llama-3-2-90b-vision-instruct is an auto-regressive language model that uses an optimized transformer architecture.	Provider: Meta	Type: Provided model

A red box highlights the "llama-3-2-3b-instruct" row, and a red circle with the letter "A" is placed over the model name in that row.

5. The **llama-3-2-3b-instruct** model card opens. You should browse through the card. Click **Select model** (A) when you finish reading.

llama-3-2-3b-instruct

Provider: Meta | Version: 3.2 | Type: Provided model

Question answering Summarization Retrieval-Augmented Generation... Classification Generation Code generation and conversion Extraction Translation Function calling Code generation
Code explanation Code fixing

Note:
This model is a Non-IBM Product governed by a third-party license that may impose use restrictions and other obligations. By using this model you agree to these terms. [Read terms](#)

Model Information

The Meta Llama 3.2 collection of multilingual large language models (LLMs) is a collection of pretrained and instruction-tuned generative models in 1B and 3B sizes (text in/text out). The Llama 3.2 instruction-tuned text only models are optimized for multilingual dialogue use cases, including agentic retrieval and summarization tasks. They outperform many of the available open source and closed chat models on common industry benchmarks.

Built with Meta Llama 3.2. Meta Llama 3.2 is licensed under the META LLAMA 3.2 Community License, Copyright © Meta Platforms, Inc. All Rights Reserved.

Back **Select model** A

6. Enter the following prompt (A) and press **Enter**.

Find the roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$.

Projects / notset's sandbox / Prompt Lab

Chat Structured Freeform AI Model: llama-3-2-3b-instruct AI guardrails off

Sample questions

What are more efficient alternatives to a 'for loop' in Python? →

What is the Transformers architecture? →

Create a chart of the top NLP use-cases for foundation models. →

Describe generative AI using emojis. →

A

Find the roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$

7. You get something like the following:

The screenshot shows a AI-powered chat interface. The user (You) asks for the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$. The AI (watsonx) responds by suggesting factoring or numerical methods, noting that the Rational Root Theorem can be challenging. It then uses synthetic division to find that $X = -1$ is a root. Factoring the equation gives $(X+1)(X^2+X-6)$, which further factors into $(X+1)(X+3)(X-2)$. The AI concludes that the roots are $X = -1, -3, 2$.

You: 01:25 PM
Find the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$.

watsonx 01:25 PM
To find the roots of the given cubic equation $X^3 + 2X^2 - 5X - 6 = 0$, we can try factoring it or use numerical methods. However, factoring a cubic equation can be challenging.

Unfortunately, this equation does not factor easily, so we'll need to use numerical methods or the Rational Root Theorem to find potential roots.

The Rational Root Theorem states that any rational root, expressed in its lowest terms as p/q , must have p as a factor of the constant term (-6) and q as a factor of the leading coefficient (1).

The factors of -6 are $\pm 1, \pm 2, \pm 3, \pm 6$. Since the leading coefficient is 1, the potential rational roots are $\pm 1, \pm 2, \pm 3, \pm 6$.

We can use synthetic division or plug in these values to find a root. Let's try plugging in some of these values:

- Trying $X = 1: 1^3 + 2(1)^2 - 5(1) - 6 = 1 + 2 - 5 - 6 = -8$, which is not 0.
- Trying $X = -1: (-1)^3 + 2(-1)^2 - 5(-1) - 6 = -1 + 2 + 5 - 6 = 0$.

We found that $X = -1$ is a root of the equation. Now we can perform polynomial division or synthetic division to factor the equation:

$$X^3 + 2X^2 - 5X - 6 = (X + 1)(X^2 + X - 6)$$

We can further factor the quadratic equation:

$$X^2 + X - 6 = (X + 3)(X - 2)$$

So, the factored form of the original equation is:

$$X^3 + 2X^2 - 5X - 6 = (X + 1)(X + 3)(X - 2) = 0$$

This gives us the roots $X = -1, X = -3$, and $X = 2$.

Please note that there might be other roots, but these are the ones we found using the Rational Root Theorem and synthetic division.

The model correctly identifies the three roots: $X = -1, X = -3$, and $X = 2$. The model used the Rational Root Theorem to identify candidates and then tested them to find the first root $X = -1$. Then it used factoring to find the other two roots.

8. Repeat steps 2 – 4 but select the **llama-3-3-70b-instruct** model this time. When you click “Select model,” you are presented with the following question. Click **Clear (A)**.

The screenshot shows the "llama-3-3-70b-instruct" model selection dialog. It displays a note about the model being a Non-IBM Product governed by a specific license. A "Start a new chat?" dialog is overlaid, asking if the user wants to start a new chat with this model. The "Clear" button is highlighted with a red box and labeled with a red letter "A".

llama-3-3-70b-instruct

Provider: Meta | Version: 3.3 | Type: Provided model

Question answering Summarization Retrieval-Augmented

Note:
This model is a Non-IBM Product governed by a specific license. By using this model you agree to these terms.

Start a new chat?

If you select a different model, you start a new chat and save this chat in the session history

Don't show this again

Cancel Clear A

NOTE: This version of Llama-3.3-70b-...

Model Information

The Meta Llama 3.3 multilingual large language model (LLM) is a pretrained and instruction tuned generative model in 70B (text

Back Select model

9. Enter the following prompt (A) and press **Enter**.

Find the roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$.

The screenshot shows the WatsonX AI interface. In the top navigation bar, it says "Projects / notset's sandbox / Prompt Lab". Below the navigation, there are tabs for "Chat", "Structured", and "Freeform". On the right, it says "AI Model: llama-3-3-70b-instruct" and "AI guardrails on". The main area is titled "Sample questions" and contains four examples with arrows pointing to the right. Below these examples is a text input field containing the prompt "Find the roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$ ". A red circle with the letter "A" is overlaid on the right side of this input field.

Observe the output. The model appears to pause briefly—about a second—before completing its response. The model provides the three correct root values: $X = -1$, $X = -3$, and $X = 2$ (A).

The screenshot shows the AI's response to the prompt. It starts with the user's message "Find the roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$ ". The AI responds with "watsonx 11:22 AM Finding Roots of a Cubic Equation". It explains that to find the roots of the given cubic equation $X^3 + 2X^2 - 5x - 6 = 0$, we can attempt to factor it or use the Rational Root Theorem as a starting point. It then provides a step-by-step guide: "Step 1: Factor the Equation if Possible", "Step 2: Apply the Rational Root Theorem", "Step 3: Test Potential Rational Roots", and "Step 4: Factor the Polynomial Using the Found Root". It includes a code snippet for testing potential rational roots:

```

def test_root(root):
    return root**3 + 2*root**2 - 5*root - 6 == 0

# Test potential roots
for root in [1, -1, 2, -2, 3, -3, 6, -6]:
    if test_root(root):
        print(f"Found a root: {root}")

```

Upon testing, we find that $X = -1$ is a root because $(-1)^3 + 2(-1)^2 - 5(-1) - 6 = -1 + 2 + 5 - 6 = 0$. The AI then factors the polynomial: $X^3 + 2X^2 - 5x - 6 = (X + 1)(X^2 + X - 6)$. It solves the quadratic equation $X^2 + X - 6 = 0$, factoring it into $(X + 3)(X - 2) = 0$. Thus, the roots of the quadratic equation are $X = -3$ and $X = 2$. The final conclusion is: "The roots of the equation $X^3 + 2X^2 - 5x - 6 = 0$ are $X = -1$, $X = -3$, and $X = 2$ ". A red circle with the letter "A" is overlaid on the right side of the conclusion text.

Note the following differences:

- The answer is laid out in steps. For this problem, this may not make a big difference, for more complex problems the steps help to lay out the answers in a logical and clear manner.
 - Both the **llama-3-2-3b-instruct** and the **llama-3-3-70b-instruct** models began with the Rational Root Theorem to identify the first root $X = -1$. However, the larger **llama-3-3-70b-instruct** model used the quadratic equation to find the final two roots.
10. Repeat steps 10 – 11, this time using the **granite-3-2-8b-instruct** model. This model quickly solves the problem and correctly identifies the three roots in one step: $x = -1$, $x = 2$, and $x = -3$ (A). The model performs a validation step (B) to confirm that its solution is correct.

NN You: 12:58 PM
Find the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$.

watsonx 12:58 PM
Here is my thought process:
To find the roots of the cubic equation $X^3 + 2X^2 - 5X - 6 = 0$, I will first attempt to factorize the equation if possible. If not, I will use the cubic formula, though it can be complex.
Here is my response:
To find the roots of the equation ($X^3 + 2X^2 - 5X - 6 = 0$), we can attempt to factorize it first.
Let's try to factor by grouping or using the Rational Root Theorem to find possible rational roots.
1. **Rational Root Theorem:** This theorem suggests that any rational root (p/q) (where (p) and (q) are integers with no common factors other than 1) of the polynomial ($a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0$) must have (p) as a divisor of the constant term (a_0) and (q) as a divisor of the leading coefficient (a_n).
For our polynomial ($X^3 + 2X^2 - 5X - 6$):

- Possible numerators (divisors of -6): ($\pm 1, \pm 2, \pm 3, \pm 6$)
- Possible denominators (divisors of 1): (± 1)

2. **Testing possible roots:**

- ($f(1) = 1^3 + 2 \cdot 1^2 - 5 \cdot 1 - 6 = 1 + 2 - 5 - 6 = -8$) (not a root)
- ($f(-1) = (-1)^3 + 2 \cdot (-1)^2 - 5 \cdot (-1) - 6 = -1 + 2 + 5 - 6 = 0$) (root)
- ($f(2) = 2^3 + 2 \cdot 2^2 - 5 \cdot 2 - 6 = 8 + 8 - 10 - 6 = 0$) (root)
- ($f(-2) = (-2)^3 + 2 \cdot (-2)^2 - 5 \cdot (-2) - 6 = -8 + 8 + 10 - 6 = 4$) (not a root)
- ($f(3) = 3^3 + 2 \cdot 3^2 - 5 \cdot 3 - 6 = 27 + 18 - 15 - 6 = 24$) (not a root)
- ($f(-3) = (-3)^3 + 2 \cdot (-3)^2 - 5 \cdot (-3) - 6 = -27 + 18 + 15 - 6 = 0$) (root)

A
We found three roots: ($x = -1, 2, -3$).

To confirm these are all roots, we can perform polynomial division or synthetic division using each root. For instance, dividing by ($(x + 1)$):

[$\frac{x^3 + 2x^2 - 5x - 6}{x + 1} = x^2 + x - 6$]

B
Now, factorizing ($x^2 + x - 6$):
[$x^2 + x - 6 = (x - 2)(x + 3)$]
Thus, the complete factorization of the original polynomial is:
[$x^3 + 2x^2 - 5x - 6 = (x + 1)(x - 2)(x + 3)$]
Therefore, the roots of the equation ($X^3 + 2X^2 - 5X - 6 = 0$) are ($x = -1, 2, -3$).

The Granite model seems to be better equipped for this type of query; it completed the task in one step, it did not need to pause, and it performed a validation.

11. Copy and paste the following prompt into your clipboard:

```
Find the roots of the equation X^3 + 2X^2 - 5x - 6 = 0.
```

12. Click the **Structured** (A) tab.



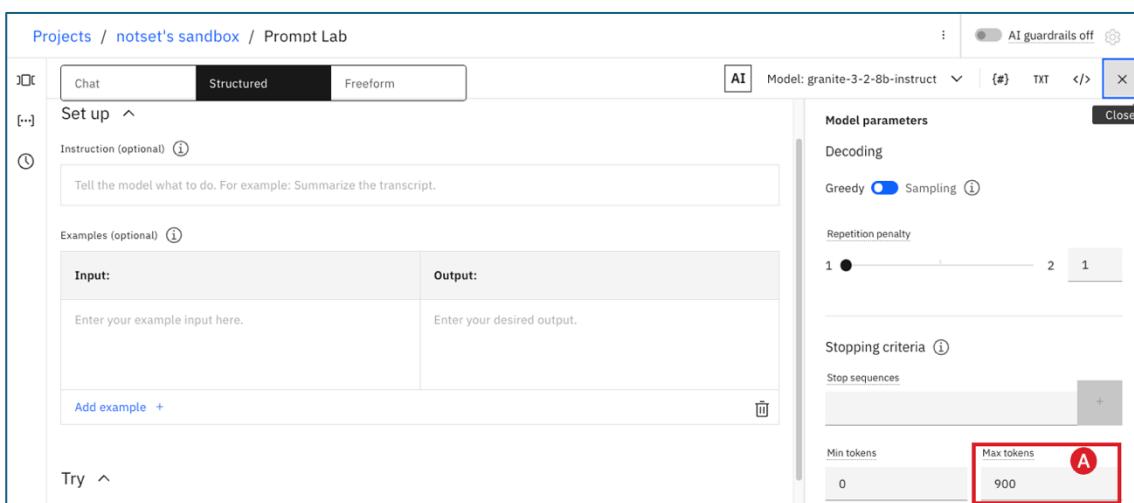
13. The **Structured** tab opens. The default model should remain the **granite-3-2-8b-instruct** (A) model. If it is not, select it as you did in step 12.



14. Click on the **Model parameters** icon.



15. The **Model parameters** slide-out appears. Change the value of **Max tokens** to 900 (A).



16. Click **x** (A) to close the slide-out.

17. Paste the prompt from your clipboard to the **Input** field in the **Try** section (A) and click **Generate** (B).

18. The Granite model puts out the same correct answer as in step 12. This time, observe the information provided at the lower left corner (A).

The Granite model took **10.8 seconds** to complete the inference. It consumed 31 input tokens and generated 287 output tokens for a **total of 318 tokens**.

19. Click **Clear output** (A).

The screenshot shows the WatsonX AI Model interface. The top navigation bar includes 'Projects / notset's sandbox / Prompt Lab'. The main area has tabs for 'Chat', 'Structured' (which is selected), and 'Freeform'. The AI status bar shows 'Model: granite-3-2-8b-instruct'. Below the tabs, there is a text input field containing a mathematical equation: $(-3) * 1 * 2 = -6$ and $-3 + 1 - 2 = 2$. The AI response follows: 'So, we can rewrite the equation as: $(X - 3)(X + 1)(X - 2) = 0$ '. It continues, 'Now, setting each factor equal to zero gives us the roots: $X - 3 = 0 \Rightarrow X = 3$ ', ' $X + 1 = 0 \Rightarrow X = -1$ ', and ' $X - 2 = 0 \Rightarrow X = 2$ '. The final statement is 'Thus, the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$ are $X = 3$, $X = -1$, and $X = 2$ '. At the bottom, there is a 'New test +' button, a 'Stop reason: End of sequence token encountered' message, and a summary: 'Tokens: 31 input + 287 generated = 318 out of 131072 Time: 10.8 seconds'. A red box labeled 'A' highlights the 'Clear output' button.

20. Change to use the **llama-3-2-3b-instruct** model (A). Click **Generate** (B).

The screenshot shows the WatsonX AI Model interface. The top navigation bar includes 'Projects / notset's sandbox / Prompt Lab'. The main area has tabs for 'Chat', 'Structured' (which is selected), and 'Freeform'. The AI status bar shows 'Model: llama-3-2-3b-instruct'. Below the tabs, there is a text input field containing the prompt: 'Try ^' and 'Test your prompt ⓘ'. The AI response area shows 'Input: Find the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$ ' and 'Output: Generated output appears here.' A red box labeled 'A' highlights the 'Model: llama-3-2-3b-instruct' dropdown. At the bottom, there is a 'New test +' button, a 'Stop reason: End of sequence token encountered' message, and a summary: 'Tokens: 31 input + 448 generated = 479 out of 131072 Time: 6.3 seconds'. A red box labeled 'B' highlights the 'Generate' button.

21. You get the following interesting output:

The screenshot shows an AI interface with the following text output:

```

Output: AI
The roots of the equation are X = -1, X = 1, X = -2. A
## Step 1: Define the equation
The given equation is X^3 + 2X^2 - 5X - 6 = 0.

## Step 2: Find the roots of the equation
To find the roots, we can use the Rational Root Theorem, which states that any rational root, expressed in its lowest terms as p/q, must have p as a factor of the constant term and q as a factor of the leading coefficient.

## Step 3: Apply the Rational Root Theorem
The constant term is -6, and its factors are ±1, ±2, ±3, ±6. The leading coefficient is 1, and its factors are ±1.

## Step 4: Test possible roots
We can start by testing the possible roots, which are the factors of the constant term divided by the factors of the leading coefficient. The possible roots are ±1, ±2, ±3, ±6.

## Step 5: Test the possible roots
By substituting the possible roots into the equation, we find that X = -1 is a root. B
## Step 6: Factor the equation
Since X = -1 is a root, we can factor the equation as (X + 1)(X^2 + X - 6) = 0. C
## Step 7: Solve the quadratic equation
We can solve the quadratic equation X^2 + X - 6 = 0 by factoring or using the quadratic formula.

## Step 8: Factor the quadratic equation
The quadratic equation X^2 + X - 6 = 0 can be factored as (X + 3)(X - 2) = 0.

## Step 9: Find the roots of the quadratic equation
The roots of the quadratic equation are X = -3 and X = 2. D
## Step 10: List all the roots
The roots of the equation are X = -1, X = 1, X = -2. E
The final answer is: $\boxed{-1, 1, -2}$
```

- The **llama-3-2-3b-instruct** model initially identified the roots to be -1, 1, and -2 (A). This is incorrect. While -1 is a correct root, neither 1 nor -2 is a root of the equation.
- The model proceeded to explain, and in Step 5 it correctly evaluated that $X = -1$ is a root (B).
- The model correctly factored out $(X+1)$ from the original equation to arrive at the quadratic equation $X^2 + X - 6 = 0$ (C).
- The model applied the quadratic formula and correctly concluded that $X = -3$ and $X = 2$ as the two other roots (D).
- However, it incorrectly concluded that the roots are $X = 1$, $X = 1$, and $X = -2$ (E). There is no clear explanation for why this occurred.

22. The tokens and time consumed data is available on the lower left.

The screenshot shows the WatsonX AI interface with the following details:

Projects / notset's sandbox / Prompt Lab

Chat Structured Freeform

Model: llama-3-2-3b-instruct

New test +

Stop reason: End of sequence token encountered

Tokens: 31 input + 448 generated = 479 out of 1310?2

Time: 11.3 seconds

Clear output Generate

The **llama-3-2-3b-instruct** model consumed more output tokens (448). It took 11.3 seconds. The following table compares the data between the Granite model and the Llama model. The cost calculation is based on information lifted from [Supported foundation models in watsonx.ai](#).

Important note: As you are sharing a watsonx.ai instance with other users, your performance, reflected in the time taken to generate results, may vary. However, the relative time should remain similar.

	granite-3-2-8b-instruct	llama-3-2-3b-instruct
Time to generate results	10.8 seconds	11.3 seconds
Input tokens	31	31
Output tokens	287	448
Input/Output tokens cost	0.0002 USD/1,000 tokens	0.00015 USD/1,000 tokens
Inference cost	0.0000636 USD	0.00007185 USD

While the cost difference appears to be totally insignificant, keep in mind that an enterprise will have tens of thousands of queries daily (if not more). Another way to look at the cost is that the **llama-3-2-3b-instruct** model is 13% more expensive than the **granite-3-2-8b-instruct** model in these types of queries. In addition, you have to read the Llama model output carefully to get the right answer.

Sub-section summary

- Models supporting the same use case (math problem in this example) behave differently. Some models provide longer and more detailed answers, while others may require multiple prompts (for example, the **llama-3-2-3b-instruct** model in a chat). Some models take longer to “think through” a problem.
- Some models may behave differently depending on how they are called. For example, the **llama-3-2-3b-instruct** model requires two steps in a chat, but it does provide the correct answer. However, when called in a structured prompt the model showed that it can do the calculation and still provide a wrong conclusion.
- Clients need to look at cost and performance. This exercise provides a simple example, but clients expecting tens of thousands of queries need to be aware of and evaluate the model behavior. Does the client need longer and more detailed output? And does the performance justify the cost difference? These must be considered at scale.
- Bonus: what if you try using the **llama-3-3-70b-instruct** model in the structured example? Does the larger model give you better results?
- Models from the same family put out different responses to the same query. Larger models do not guarantee better (or more accurate) answers. Testing is necessary.

Evaluating model response to tables in PDF

Models are often required to generate output based on non-textual input such as pictures, charts, graphs, and more. In some use cases, the Large Language Model (LLM) can handle them well. This section examines how to evaluate the responses of different LLMs to a table in a PDF file.

- Download the Prescription side effects PDF file from <https://ibm.biz/BdngKS> and save it locally on your computer. The content of the file is as follows:

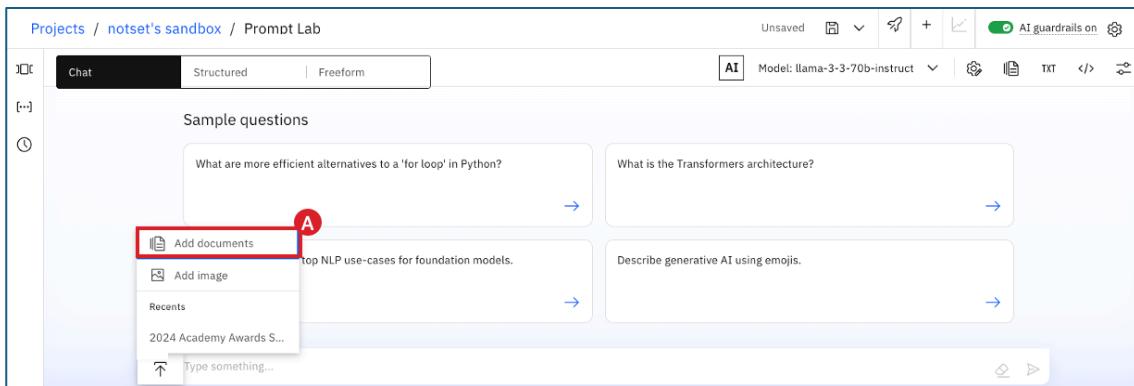
The following table shows the side effects of various prescriptions.

	Nausea and vomiting,	Non-healing wounds	Pain in the Chest	Difficulty talking	Swelling, redness of the eye	Mood swing and depression	Blurring vision
Prescription A	Yes	Yes	No	No	Yes	No	No
Prescription B	No	Yes	Yes	No	No	Yes	Yes
Prescription C	Yes	No	Yes	Yes	Yes	No	No
Prescription D	Yes	Yes	No	Yes	Yes	Yes	No
Prescription E	Yes	No	Yes	No	No	No	Yes
Prescription F	Yes	No	Yes	No	Yes	No	No

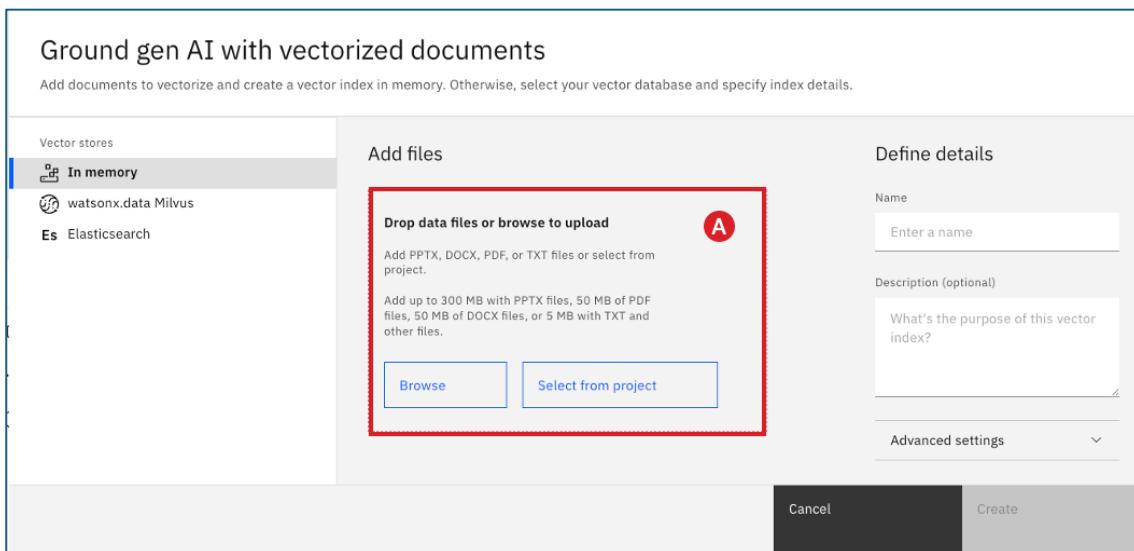
- Log into watsonx.ai and click Open Prompt Lab (A).

- Click the Chat tab if it does not open by default.
- Select the **llama-3-3-70b-instruct** model if it is not selected by default (A), and click on the Upload files button (B).

5. Click Add documents (A).



6. The **Ground gen AI with vectorized documents** page opens. Drag and drop the downloaded Prescription side effect PDF file to the **Add files** area (A).



7. The file takes a few seconds to load (A). Click **Create** (B).

Ground gen AI with vectorized documents

Add documents to vectorize and create a vector index in memory. Otherwise, select your vector database and specify index details.

Vector stores

- In memory
- watsonx.data Milvus
- Elasticsearch

Add files

Drop data files or browse to upload

Add PPTX, DOCX, PDF, or TXT files or select from project.

Add up to 300 MB with PPTX files, 50 MB of PDF files, 50 MB of DOCX files, or 5 MB with TXT and other files.

Browse Select from project

Prescription side effects.pdf 32.8KB

Define details

Name: Prescription side effects

Description (optional): What's the purpose of this vector index?

Advanced settings

Cancel Create

8. On the Chat tab, the Prescription side effects file is shown (A). Enter the following prompt (B) and press **Enter**.

What side effects does Prescription C cause?

Projects / notset's sandbox / Prompt Lab

Chat Structured Freeform

Sample questions

Summarize the document. → Summarize the document as a list of 5 questions. →

Prescription side effects A

What side effects does Prescription C cause? B

9. You get the following completion:

You: 04:19 PM
What side effects does Prescription C cause?

watsonx 04:19 PM
Prescription C Side Effects

The following side effects are associated with Prescription C:

- Nausea and vomiting
- Pain in the chest
- Difficulty talking
- Swelling, redness of the eye
- Mood swing and depression are not associated with Prescription C, however,
- Blurring vision is also not associated with Prescription C.

The Llama 3 model correctly identified Prescription C's side effects. However, it also consumed tokens to say that **Mood swing and depression** and **Blurring vision** are not Prescription C's side effects. While this is correct, it added no useful information. In fact, this causes some confusion as the model completely omitted **Non-healing wounds**. Clients may question whether the model simply missed that side effect, which casts doubt on the rest of the response.

10. You will test with a different model. To switch to a different model, you must first clear the chat. Click the **Clear chat** icon (A).

Projects / notset's sandbox / Prompt Lab

Chat Structured | Freeform

NN You: 04:19 PM
What side effects does Prescription C cause?

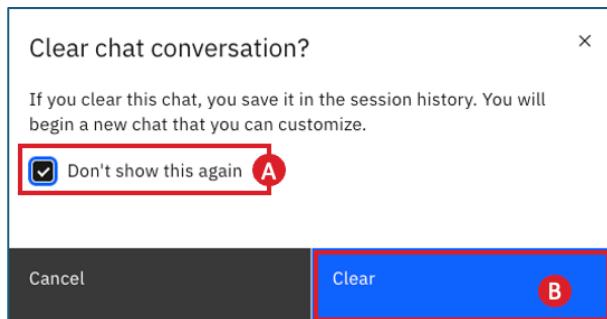
watsonx 04:19 PM
Prescription C Side Effects

The following side effects are associated with Prescription C:

- Nausea and vomiting
- Pain in the chest
- Difficulty talking
- Swelling, redness of the eye
- Mood swing and depression are not associated with Prescription C, however,
- Blurring vision is also not associated with Prescription C.

A

11. The **Clear chat conversation?** dialog box appears. Select **Don't show this again** (A) and click **Clear** (B).



12. Change to use the **mistral-large** model (A). Enter the following prompt (B) and press **Enter**.

13. The **mistral-large** model returns the following:

The **mistral-large** model correctly identifies all the correct side effects. Unlike the **llama-3-3-70b-instruct** model, the **mistral-large** model does not partially identify non-side effects. The **mistral-large** model's answer is clearer and leaves no uncertainties.

14. Change the model one more time to the **llama-4-maverick-17b-128e-instruct-fp8** (A) model and issue the same prompt as follows (B) and press **Enter**.

The screenshot shows the WatsonX AI Prompt Lab interface. At the top, there's a navigation bar with 'Projects / notset's sandbox / Prompt Lab'. Below it is a toolbar with 'Chat' (selected), 'Structured', 'Freeform', and other icons. A dropdown menu 'Model: llama-4-maverick-17b-128e-instruct-fp8' is open, with a red box labeled 'A' highlighting the model name. To the right of the model dropdown is a gear icon. Below the toolbar is a complex flowchart with various nodes (circles, squares, diamonds) connected by lines. In the center, there's a large white area labeled 'Sample questions' containing two examples: 'Summarize the document.' and 'Summarize the document as a list of 5 questions.', each with a blue arrow pointing to the right. At the bottom, there's a search bar with a magnifying glass icon and the text 'Prescription side effects' followed by a red box labeled 'B' containing the prompt 'What side effects does Prescription C cause?'. To the right of the search bar are a close button 'x', a dropdown arrow, and a right-pointing arrow.

15. You notice two things:

- It takes slightly longer for the model to return its completion.
- The model provides the correct answer (A) and some logical thinking (B) behind the answer.

watsonx 07:31 PM

Side Effects of Prescription C

Prescription C causes the following side effects:

- Nausea and vomiting: Yes
- Non-healing wounds: No
- Pain in the Chest: Yes
- Difficulty talking: Yes
- Swelling, redness of the eye: Yes
- Mood swing and depression: No
- Blurring vision: No

The side effects can be summarized in a table as follows:

Side Effects	Prescription C
Nausea and vomiting	Yes
Non-healing wounds	No
Pain in the Chest	Yes
Difficulty talking	Yes
Swelling, redness of the eye	Yes
Mood swing and depression	No
Blurring vision	No

In list format, the side effects are:

- Nausea and vomiting
- Pain in the Chest
- Difficulty talking
- Swelling, redness of the eye

Subsection summary

In this part of the lab, you tested three different models for their ability to interpret a table in a PDF file. While the Llama 3 and Mistral models provided similarly sized outputs, the **llama-4-maverick-17b-128e-instruct-fp8** model produced a significantly longer output. In the table below, the number of words is used as a proxy for token count when estimating costs (your numbers may be different). Note that this is not an accurate measure, as words do not map one-to-one with tokens. However, the intent is to consider relative cost, and the ratio should be approximately correct.

	llama-3-3-70b-instruct	mistral-large	llama-4-maverick-17b-128e-instruct-fp8
Answer	Correct, can be misleading	Correct and succinct	Correct with explanation
Number of words (tokens)	Input 7, output 43	Input 7, output 35	Input 7, output 104
Inference cost (per 1,000 tokens)	0.00071 for input and output	0.003 for input, 0.01 for output	0.00035 for input, 0.0014 for output
Inference cost	0.0355	0.371	0.14805

- While all three models can find the right answer, the quality and cost vary significantly.
- The **llama-3-3-70b-instruct** model is the cheapest and it correctly answered the question. However, its completion can be misleading. Clients may be unsatisfied with ambiguous answers that may require human intervention to clarify. The cost is attractive.
- The **mistral-large** provided the best answer – it is the shortest, to the point, and has no ambiguity. However, it is also by far the most expensive. It is ten times more expensive than the **llama-3-3-70b-instruct** model and 2.5 times that of the **llama-4-maverick-17b-128e-instruct-fp8** model.
- The **llama-4-maverick-17b-128e-instruct-fp8** model falls in between. This is a Mixture of Experts (MoE) model with 128 experts but only utilizes 17b parameters at inference time. This model produced the correct result but was excessively verbose, generating 2.5 to 3 times the necessary output, which significantly impacted the cost. It is efficient as it utilizes a lot fewer parameters than the Llama and Mistral models. Clients looking for performance may consider this model.
- This section demonstrates how various models process tables in PDF files. Although all models produced accurate results, their outputs and associated costs differed significantly. While **llama-3-3-70b-instruct** offered the lowest cost, it may necessitate further model tuning or answer refinement, potentially offsetting the cost savings. Conversely, the most effective model, **mistral-large**, is costly. This scenario exemplifies the common trade-off between cost and performance. Clients should evaluate and select models based on their specific business needs and requirements.
- Looking at the cost, clients need to consider multiple factors:
 - **Accuracy** – Does the model provide the correct or desirable answer?
 - **Performance** – How long does it take the model to respond?
 - **Verbosity** – How many tokens or words does the model use to provide a correct answer?
 - **Token cost** – How much does the model charge per input or output token?

Clients will need to balance these factors to identify the optimal model based on their business requirements.

Evaluating multi-modal models

Increasingly, clients require multi-modal models that can handle both text and images. The last section showed that some LLMs can handle tables in PDFs. However, there are other images such as pictures, charts, graphs, histograms, and more.

IBM watsonx.ai offers multi-modal models for various use cases. These models are trained to understand and interpret images. As of June 2025, the multi-modal models available for watsonx.ai Software as a Service (SaaS) are vision models, which are used to generate text based on an input image:

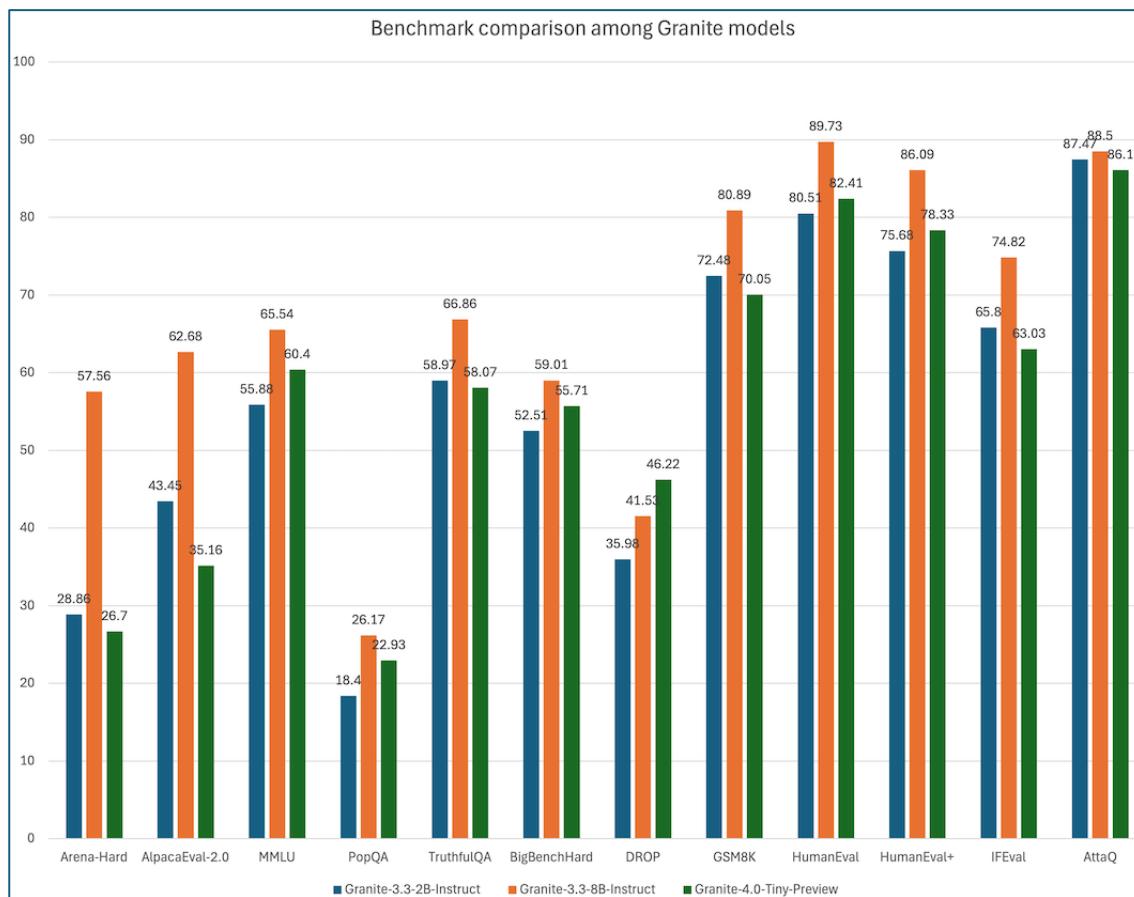
- granite-vision-3-2-2b
- llama-3-2-11b-vision-instruct
- llama-3.2-90b-vision-instruct
- llama-4-maverick-17b-128e-instruct-fp8
- mistral-medium-2505
- mistral-small-3-1-24b-instruct-2503
- pixtral-12b

There is also a llama-guard-3-11b-vision model that can be used to screen harmful input or output. Another model llama-4-scout-17b-16e-instruct is in beta.

Other multi-modal models for audio are in the roadmap.

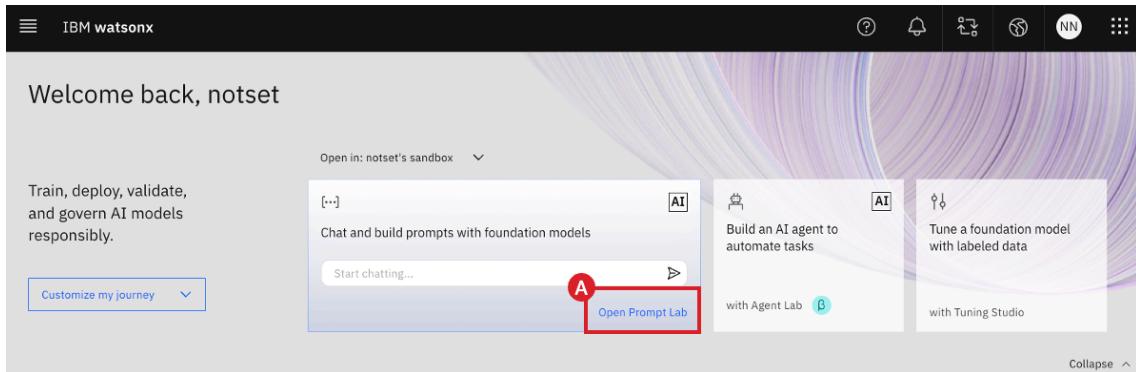
In this section, you will test several of these models using a chart.

16. Download the Granite_benchmark PNG file from <https://ibm.biz/Bdnhvq> and save it locally on your computer. The content of the file is as follows:



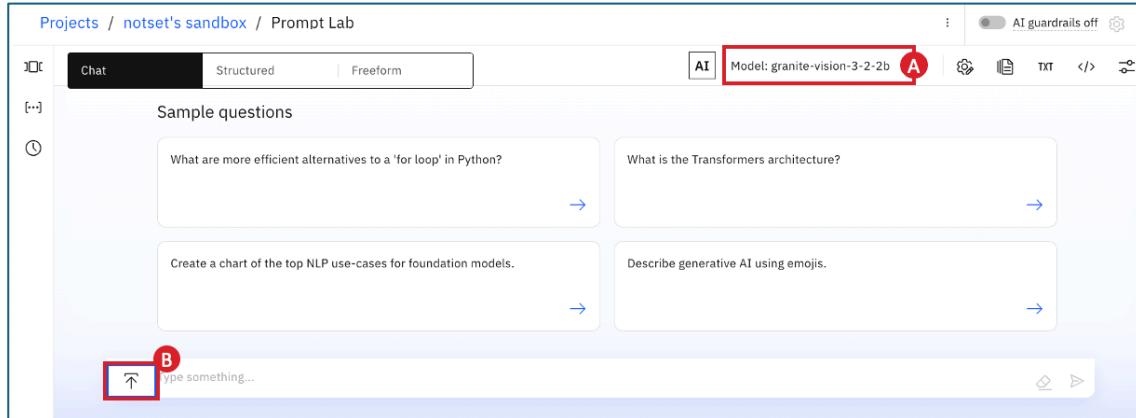
This information is taken from Hugging Face. You can find it [here](#).

17. Log into watsonx.ai and click Open Prompt Lab (A).

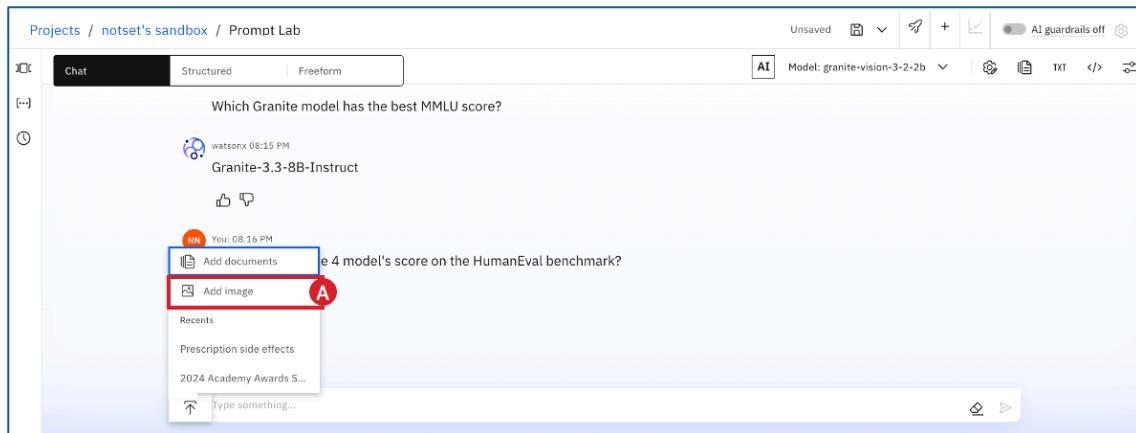


18. Click the Chat tab if it does not open by default.

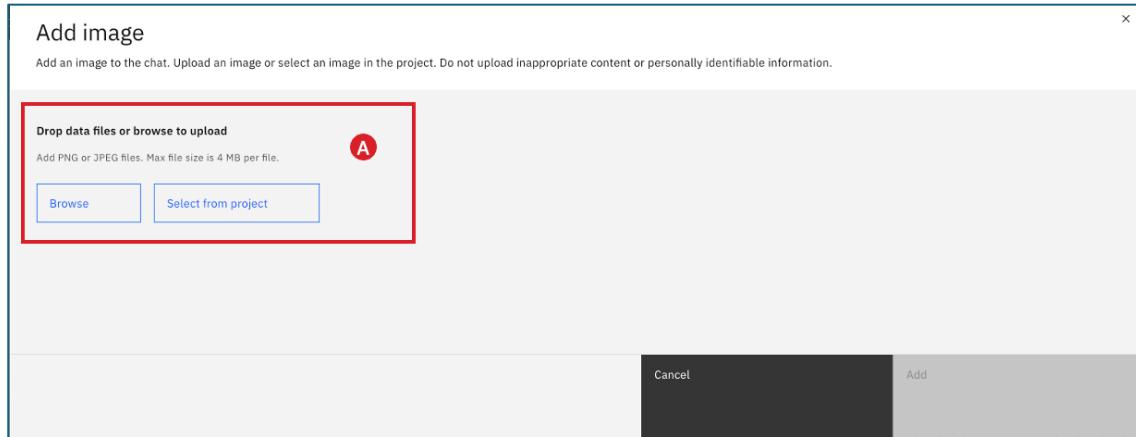
19. Select the **granite-vision-3-2-2b** model if it is not selected by default (A), and click on the Upload files button (B).



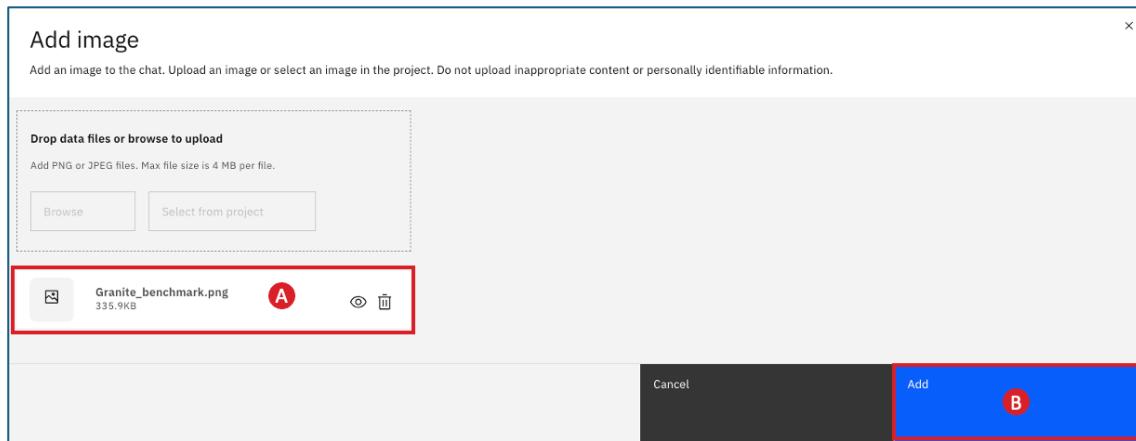
20. Click Add Image (A).



21. The Add image page opens. Drag and drop the downloaded Granite_benchmark png file to the **Add files** area (A).

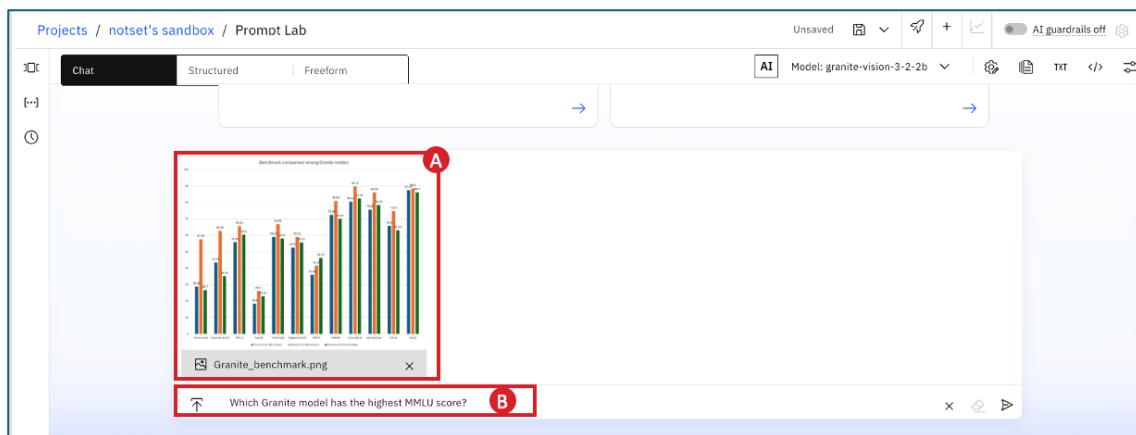


22. The file takes a few seconds to load (A). Click **Add** (B).



23. On the Chat tab, the Granite_benchmark graph is shown (A). Enter the following prompt (B) and press **Enter**.

Which Granite model has the highest MMLU score?



24. The Granite model correctly names the **Granite-3.3-8B-Instruct** model (A). Enter this follow-on prompt (B) and press **Enter**.

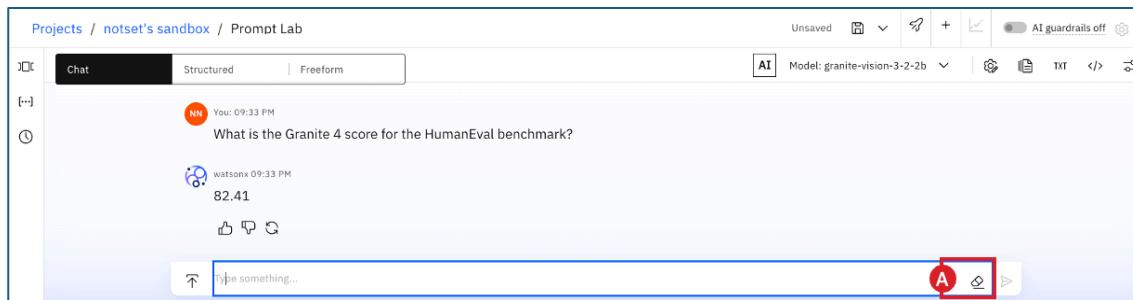
What is the Granite 4 score for the HumanEval benchmark?

The screenshot shows the WatsonX AI Prompt Lab interface. At the top, there's a bar chart titled "MMLU" comparing models like Anthropic, AlpacaVLLM, and others. The chart includes a legend for "Granite-3.3-8B-Instruct" (blue), "Granite-3.3-8B-Instruct" (orange), and "Granite-4.0-Tiny-Preview" (green). Below the chart, a question is asked: "Which Granite model has the highest MMLU score?". The AI response "Granite-3.3-8B-Instruct" is highlighted with a red box and labeled "A". In the input field below, the question "What is the Granite 4 score for the HumanEval benchmark?" is typed, with the "B" label placed over it.

25. The Granite model accurately responds with **82.41** (A). On the chart, it is labeled as **Granite-4.0-Tiny-Preview** (B). Notably, the **granite-vision-3-2-2b** model correctly linked the input "Granite 4" to this entry despite the name discrepancy.

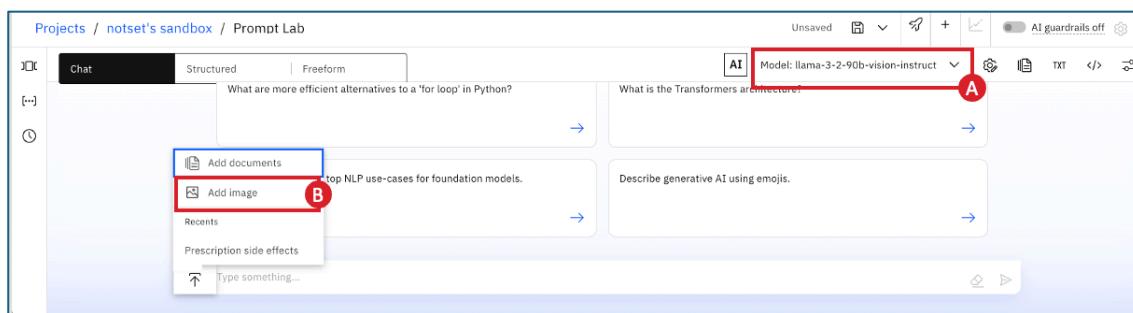
The screenshot shows the WatsonX AI Prompt Lab interface. At the top, there's a bar chart titled "MMLU" comparing models like Anthropic, AlpacaVLLM, and others. The chart includes a legend for "Granite-3.3-8B-Instruct" (blue), "Granite-3.3-8B-Instruct" (orange), and "Granite-4.0-Tiny-Preview" (green). Below the chart, a question is asked: "Which Granite model has the highest MMLU score?". The AI response "Granite-3.3-8B-Instruct" is highlighted with a red box and labeled "A". In the input field below, the question "What is the Granite 4 score for the HumanEval benchmark?" is typed, with the "B" label placed over it. The AI response "82.41" is highlighted with a red box and labeled "A".

26. Click the **Clear chat** icon (A).

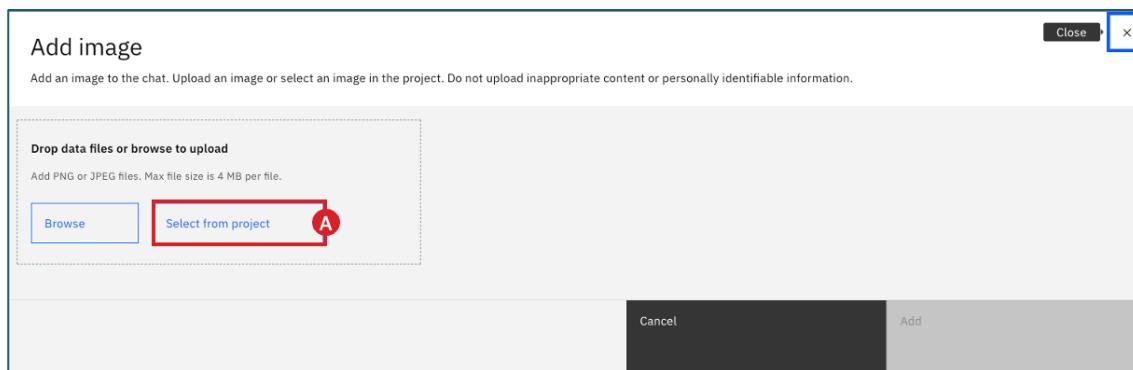


Unlike using a document, when you clear a chat with an image, the image is no longer loaded. If you want to test another model you need to add the image again.

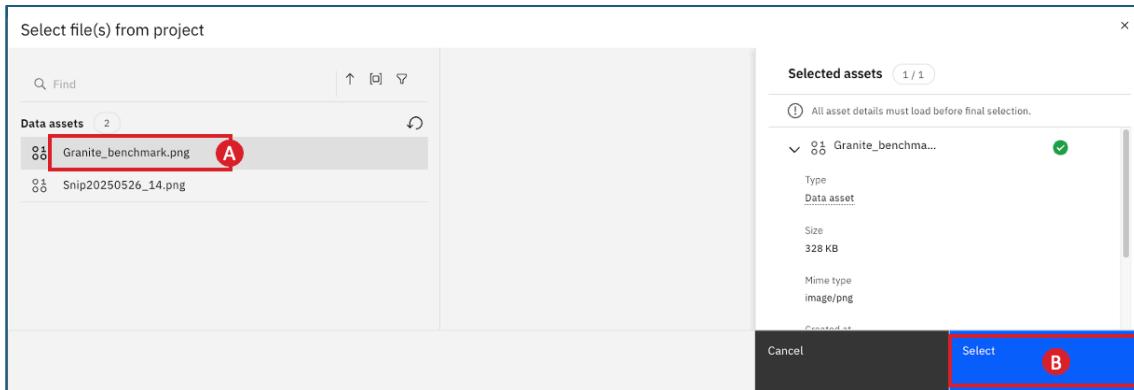
27. Change to use the **llama-3-2-90b-vision-instruct** model (A). Click **Upload files** and then Click **Add Image** (B).



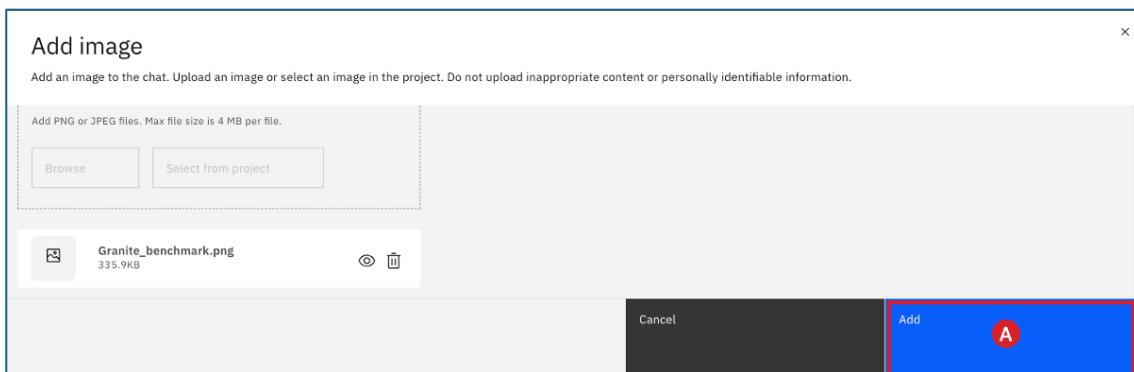
28. Upon initial addition, the image was also included in the sandbox project. On the **Add image** page, click **Select from project** (A).



29. The Select file(s) from project page opens. Click **Granite_benchmark.png** (A), then click **Select** (B).



30. Click **Add** (A).



31. Enter the following prompt (A) and press **Enter**.

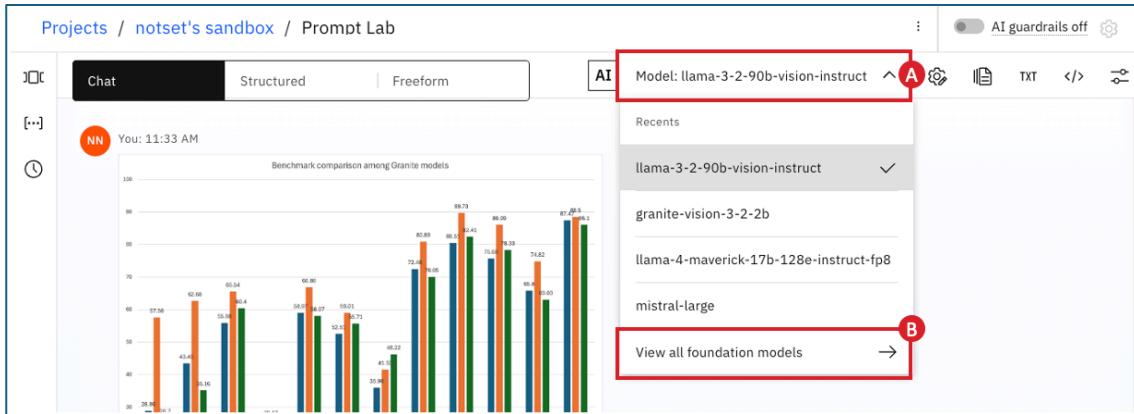
Which Granite model has the highest MMLU score?

32. The model returned the following:



This is just wrong. The **llama-3-2-90b-vision-instruct** model did not interpret the picture properly. This is somewhat surprising.

33. Click **llama-3-2-90b-vision-instruct** (A) to open up the model menu drop-down. Click **View all foundation models** (B).



34. The **Select a foundation model** page opens. Locate and click on the **llama-3-2-90b-vision-instruct** tile (A).

All models		Model benchmarks	
<input type="button" value="Search for a model or task"/> <input type="text"/>			
llama-3-1-70b-instruct	llama-3-1-8b-instruct	llama-3-2-11b-vision-instruct	
Llama-3-1-70b-instruct is an auto-regressive language model that uses an optimized transformer architecture.		Llama-3-1-8b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	
Provider: Meta	Type: Provided model	Provider: Meta	Type: Provided model
llama-3-2-1b-instruct	llama-3-2-3b-instruct	llama-3-2-90b-vision-instruct	
Llama-3-2-1b-instruct is an auto-regressive language model that uses an optimized transformer architecture.		Llama-3-2-3b-instruct is an auto-regressive language model that uses an optimized transformer architecture.	
Provider: Meta	Type: Provided model	Provider: Meta	Type: Provided model

35. The **Model Information** section of the **llama-3-2-90b-vision-instruct** model card has the following:

The Llama 3.2-Vision collection of multimodal large language models (LLMs) is a collection of pretrained and instruction-tuned image reasoning generative models in 11B and 90B sizes (text + images in / text out). The Llama 3.2-Vision instruction-tuned models are **optimized for visual recognition, image reasoning, captioning, and answering general questions about an image**. The models outperform many of the available open source and closed multimodal models on common industry benchmarks.

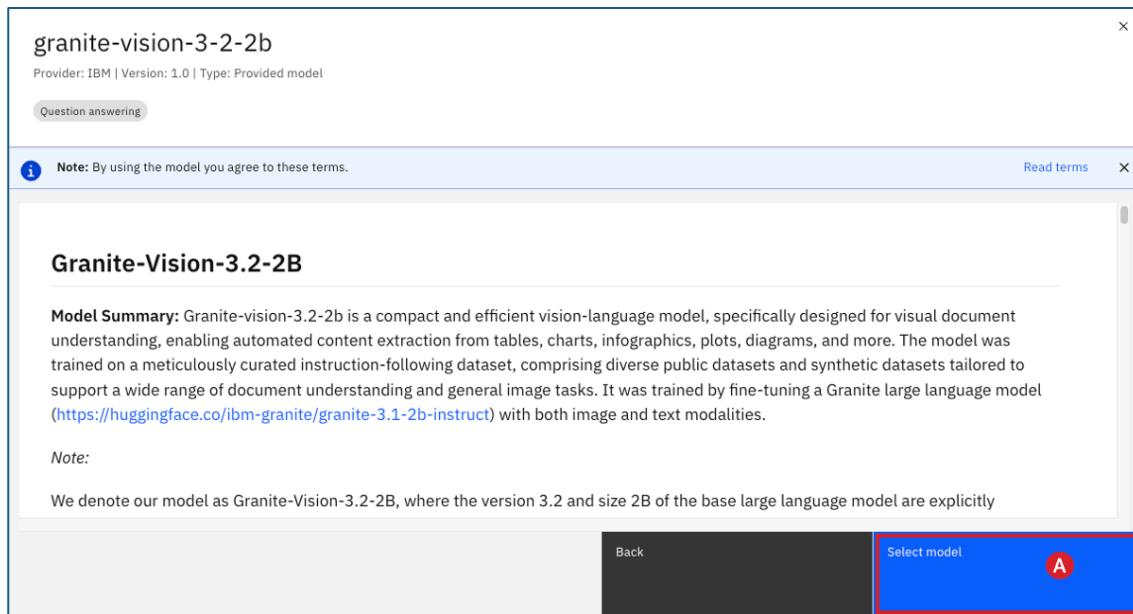
Repeat steps 18-19 to view the model card for **granite-vision-3-2-2b**. The model card says this:

Granite-vision-3.2-2b is a compact and efficient vision-language model, specifically designed for visual document understanding, enabling **automated content extraction from tables, charts, infographics, plots, diagrams, and more**. The model was trained on a meticulously curated instruction-following dataset, comprising diverse public datasets and synthetic datasets tailored to support a wide range of document understanding and general image tasks. It was trained by fine-tuning a Granite large language model (<https://huggingface.co/ibm-granite/granite-3.1-2b-instruct>) with both image and text modalities.

This comparison provides clues to the behavior:

- The smaller Granite vision model is optimized to work with charts, plots, diagrams, and so on. It works well with a histogram.
- The much bigger Llama vision model is optimized for visual recognition, image reasoning, and so on. It does not explicitly mention graphs, charts, and other graphical images. In the next test you will try an image and see its strength.

36. With the **granite-vision-3-2-2b** model card opened, click **Select model** (A).



37. Download the Granite Accident jpeg file from <https://ibm.biz/BdnVv4> and save it locally on your computer.

38. Repeat Steps 20-22 to add this image.

39. Enter the following prompt (A) and press **Enter**.

A screenshot of the Prompt Lab application interface. At the top, it shows 'Projects / notset's sandbox / Prompt Lab'. The toolbar includes 'Chat' (selected), 'Structured', 'Freeform', 'AI' (button), 'Model: granite-vision-3-2-2b', and various document and AI-related icons. In the center, there's a 'Sample questions' section with two boxes: 'Summarize the document.' and 'Summarize the document as a list of 5 questions.', each with a right-pointing arrow. Below this is a 'Prescription side effects' section with a small image of a car accident. A file browser window shows an image named 'Accident.jpeg'. At the bottom, a text input field contains the prompt 'Please assess the damage.' with a red circle containing the letter 'A'.

40. The model returns the following answer:

The screenshot shows the WatsonX AI Prompt Lab interface. At the top, it says "Projects / notset's sandbox / Prompt Lab". Below that is a toolbar with "Chat" (selected), "Structured", "Freeform", and an "AI" button. The "Model" dropdown is set to "granite-vision-3-2-2b". On the left, there's a sidebar with a "Chat" section and a "Watson AI" icon. The main area shows a dark-colored car with significant front-end damage, including a crumpled bumper and detached hood, parked on a wet road. Below the image is the prompt "Please assess the damage.". A message from "watsonx 02:18 PM" follows: "The damage to the car is extensive. The front bumper is severely crumpled, and the hood is detached, exposing the engine and other internal components. The windshield is shattered, and the side mirrors are bent. The car's bodywork is dented and scratched, and the tires are flat. The damage suggests a high-impact collision, likely involving a significant force to the front of the vehicle." There are also "Unsaved" and "AI guardrails off" buttons at the top right.

Reading the description, this is what the model gets right:

- The front bumper is severely crumpled, and the hood is detached, exposing the engine and other internal components.
- The car's bodywork is dented and scratched, and the tires are flat.

However, the windshield is not shattered, and the side mirrors do not look bent.

41. Change to use the **llama-3-2-90b-vision-instruct** model (A). Use **Add image** to add the **Accident.jpeg** file (B). Enter the following prompt (C) and press **Enter**.

The screenshot shows the WatsonX AI Prompt Lab interface. At the top, it says "Projects / notset's sandbox / Prompt Lab". Below that is a toolbar with "Chat" (selected), "Structured", "Freeform", and an "AI" button. The "Model" dropdown is set to "llama-3-2-90b-vision-instruct" (labeled A). The main area has several input fields with arrows pointing right:

- "What are more efficient alternatives to a 'for loop' in Python?"
- "What is the Transformers architecture?"
- "Create a chart of the top NLP use-cases for foundation models."
- "Describe generative AI using emojis."

Below these is a large image of a dark-colored car with significant front-end damage, including a crumpled bumper and detached hood, parked on a wet road. This image is highlighted with a red border and labeled B. At the bottom, there's a text input field containing "Please assess the damage." (labeled C). There are "Unsaved" and "AI guardrails off" buttons at the top right.

42. The model returns the following:

watsonx 02:45 PM

Based on the image, here is a detailed assessment of the damage to the vehicle:

Front End Damage:

- The front bumper is severely damaged and appears to be torn off, with debris scattered on the ground.
- The hood is crumpled and dented, with visible creases and folds.
- The front grille is broken and hanging off, with some parts missing.
- The headlights are damaged, with one of them cracked and the other one broken.

Body Damage:

- The front quarter panel is dented and scratched, with some paint chipped off.
- The driver's side door has a large dent and scratch, with some paint transfer from another vehicle.
- The rear quarter panel has a small dent and scratch, but it appears to be minor.

Glass Damage:

- The windshield is cracked and broken, with some shards of glass on the ground.
- The side windows are intact, but there are some scratches and scuffs on the glass.

Other Damage:

- The front tire is flat and damaged, with some debris stuck in the wheel well.
- The front suspension appears to be damaged, with some components bent or broken.
- There is some damage to the undercarriage, with some parts hanging off or broken.

Overall, the damage to the vehicle appears to be significant, with major damage to the front end, body, and glass. The vehicle may require extensive repairs or even be considered a total loss.

This model provides a lot more details. Some items (such as the scratch on the side door and damage to the rear panel) can only be seen visually when you enlarge the picture.

However, there are some minor errors. Based on the position of the steering wheel, the damage is on the passenger side, not the driver's side. The windshield appears to have a small crack but does not show any additional damage. The side windows look untouched. The model might have mistaken rain splatters as scratches.

Subsection summary

Multi-modal models are designed to process inputs beyond plain text. In this section, you explored vision-based multi-modal models, which are trained to interpret images. There is a wide variety of image types, and not every model performs equally well across all of them.

- Tabular data embedded in PDF files are common image-based inputs. In such cases, it may be possible to use large language models (LLMs). A key advantage of using LLMs is the ability to leverage existing models already in use, which helps reduce maintenance and development efforts.
- Some models, such as the Granite vision model, are specifically trained to interpret structured visual content like graphs, plots, infographics, and histograms. These models often outperform larger, general-purpose models that are primarily trained for broad image interpretation tasks.
- Other image models, such as the Llama Vision model, perform exceptionally well in image interpretation and can identify details that may be overlooked by the human eye. Organizations, such as insurance companies, can leverage these models to efficiently assess various cases. Depending on the use case, smaller models (such as the Granite Vision model) may be ideal for rapid, on-the-spot assessments, while larger models (such as the Llama Vision model) can provide more detailed analyses suitable for comprehensive reporting.
- Keep in mind that models can produce hallucinated outputs. It is essential to implement validation mechanisms to verify the accuracy and reliability of model-generated content.
- In all cases, it is essential to test models using business-specific images to understand how they respond to real-world scenarios.

Reasoning with Granite models

Large language models (LLMs) demonstrate strong thinking (or reasoning) abilities, particularly in deductive, inductive, and multi-step reasoning. With deductive reasoning, LLMs apply general rules to specific situations to draw logical conclusions. For example, inferring that a whale is warm-blooded because all mammals are. Through inductive reasoning, LLMs can identify patterns from specific examples and make generalizations, such as predicting future events based on observed trends.

LLMs also excel at multi-step reasoning, where LLMs break down complex problems into a sequence of logical steps. This allows LLMs to solve math problems, analyze scenarios, or plan tasks by reasoning through intermediate stages.

IBM Granite models have reasoning capabilities similar to other non-IBM models. However, IBM Granite models are unique in their ability to enable or disable thinking.

Why is this important? While thinking and reasoning can help solve complex problems, large language models will "think out loud," which consumes a large number of tokens. This translates to increased costs.

While the thinking capability is beneficial for complex queries, it may introduce unnecessary overhead and costs if applied to all queries. With Granite's ability to toggle thinking on or off, clients can leverage the advanced problem-solving capabilities of a thinking model when needed and avoid unnecessary costs when it is not.

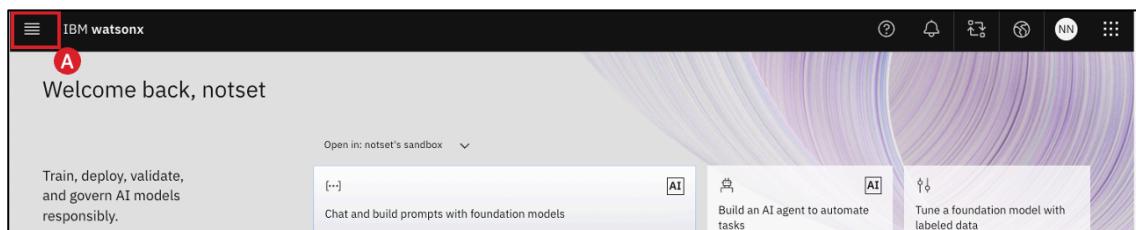
Learning Goals

In this section, you will:

- Work with a Jupyter Notebook to observe how the granite-3-3-8b-instruct model enables or disables thinking.
- Look at the relative cost of disabling versus enabling thinking.
- Test and observe how the granite-3-3-8b-instruct model responds to a logical problem with thinking disabled versus enabled.

Creating an API key

1. You need an API key to work with the notebook. Log into watsonx.ai and click the **main menu** (A).



2. Scroll down to find and click on Access (IAM) (A).

The screenshot shows the IBM WatsonX dashboard. On the left, a navigation sidebar includes Home, Data, Projects, AI governance, AI use cases, External models, Inventories, Deployment spaces, Resource hub, Administration, Account and billing, Services, Access (IAM) (highlighted with a red box and circled A), Configurations and settings, and Environment runtimes. The main area displays several AI-related services: Chat and build prompts with foundation models (with a Start chatting... button and Open Prompt Lab link), Build an AI agent to automate tasks (with a Build Agent Lab link), Tune a foundation model with labeled data (with a Tune with Tuning Studio link), notset's sandbox / Granite thinking, Projects / notset's sandbox, notset's sandbox / Prompt Lab, and notset's sandbox / test 33. At the bottom, there are sections for Developer access and Developer hub.

3. The IBM Cloud Identity and Access Management page opens. Click API keys (A).

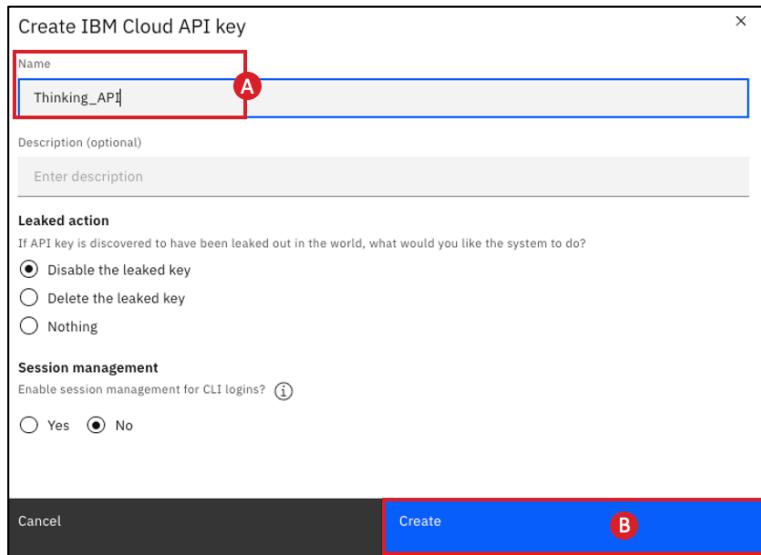
The screenshot shows the IBM Cloud IAM Overview page. The left sidebar lists Overview, Dashboard, Manage identities (Users, Trusted profiles, Service IDs, API keys highlighted with a red box and circled A), Manage access (Access groups, Authorizations, Roles), and Identity providers. The main content area features a heading "IBM Cloud Identity and Access Management" with a subtext "Securely authenticate users for platform services and control access to resources." It includes buttons for "Invite users" and "Create access group" and a "Help" link. A "Last updated: 2/6/2024" timestamp and a "What's new?" section are also present. To the right is a diagram illustrating identity management concepts like fingerprint scanning and cloud connectivity.

4. The API keys page opens. Click Create (A).

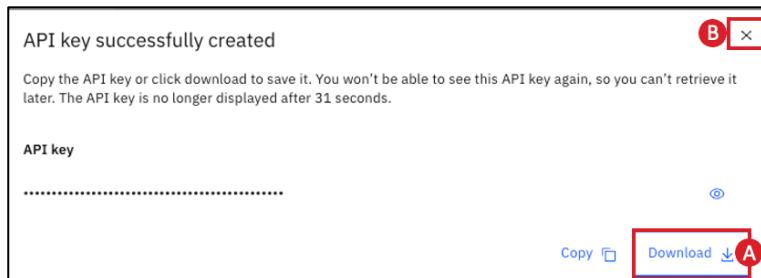
The screenshot shows the API keys page within the IBM Cloud IAM interface. The left sidebar is identical to the previous screenshot. The main content area has a heading "API keys" with a subtext about managing API keys. It includes a "View My IBM Cloud API keys" dropdown, a search bar "Filter by API key name or description", and a "Create" button (highlighted with a red box and circled A). Below is a table with columns: Status, Name, Description, Date created, and Enabled. One row is visible in the table.

Status	Name	Description	Date created	Enabled
	G0f-spIkey-XvcbA5ygFB-a627240a-bbe3-4261-80e7-f33b734e1466-2025-05-22T22:57:02Z	API key created/managed by task credentials. It is managed for your use with Watson Studio operations. Please do not delete here.	5-22-2025 22:57 GMT	<input checked="" type="checkbox"/> Yes

5. The **Create IBM Cloud API key** dialog opens. Enter **Thinking_API** (A) as **Name**. Leave all other fields at their default values and click **Create** (B).



6. The **API key successfully created** page opens. Click **Download** (A) to download the **apikey.json** file with the API key information. Click **x** (B) to close the window.



7. The content of your **apikey.json** file looks like this (with the actual key redacted in this sample output):

```
{
  "name": "Thinking_API",
  "description": "",
  "createdAt": "2025-06-04T03:02+0000",
  "apikey": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
```

You need this apikey to use the Jupyter Notebook.

Adding a Jupyter Notebook to your project

1. Download the Granite Thinking ipynb file from <https://ibm.biz/BdnJUx> and save it locally on your machine.

2. If you have not closed your **API keys** page, your newly created API key is shown on the list (A). Click on the **main menu** (B). Skip to step 4.

The screenshot shows the IBM Cloud interface with the main menu (B) highlighted. The left sidebar is open, showing various IAM management options. The central area displays the 'API keys' section with a table listing a single API key. The key is named 'Thinking_API' (A), which is highlighted with a red box. The table columns include Status, Name, Description, Date created, and Enabled. The status is 'Active', name is 'Thinking_API', description is 'API key created/managed by task credentials. It is managed for your use with Watson Studio operations. Please do not delete here.', date created is '5-22-2025 22:57 GMT', and enabled status is 'Yes'.

3. If you had logged out of watsonx.ai, log back in using the credentials from your TechZone reservation. Click on the **main menu** (A).

The screenshot shows the IBM watsonx dashboard. The main menu (A) is highlighted. Below it, there's a welcome message 'Welcome, notset'. On the right, there are three AI-related buttons: 'Chat and build prompts with foundation models' (AI icon), 'Build an AI agent to automate tasks' (AI icon), and 'Tune a foundation model with labeled data' (NN icon).

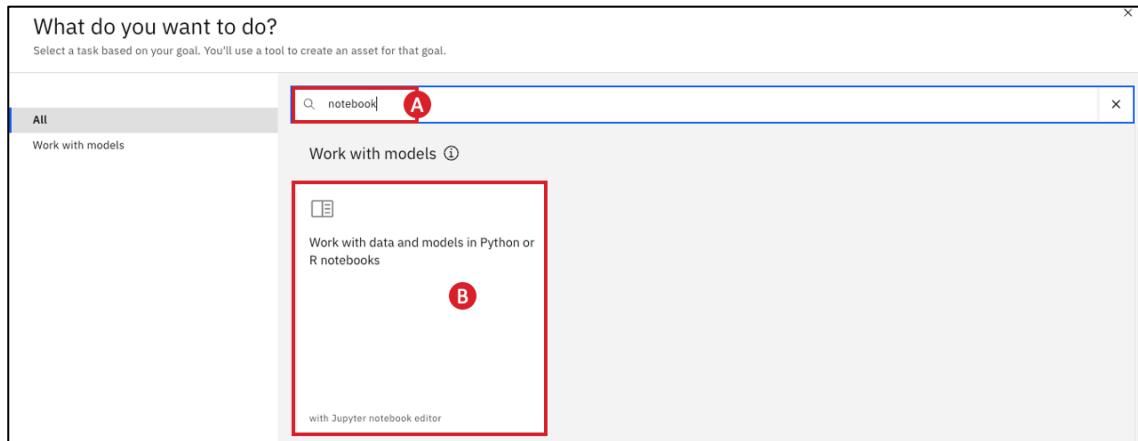
4. Scroll down to the **Projects** area (A) and click **notset's sandbox** project (B).

The screenshot shows the 'Recent work' section of the IBM watsonx dashboard. The 'Projects' area (A) is highlighted. A project named 'notset's sandbox' (B) is selected and highlighted with a red box. The 'Deployment spaces' area is also visible on the right.

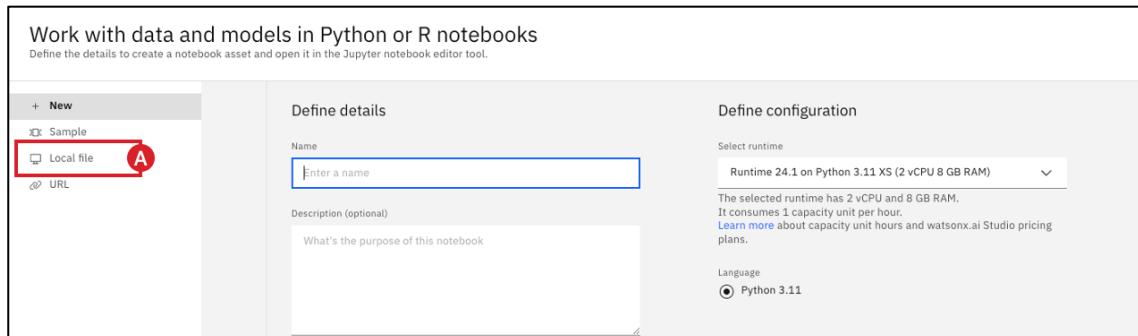
5. Click the **Assets** tab (A) if not opened by default. Click **New asset** (B).

The screenshot shows the 'notset's sandbox' assets page. The 'Assets' tab (A) is highlighted. In the top right corner, there is a 'New asset' button (B) with a red box around it. The main area displays a table of assets, with one entry visible: 'reason_notebook_1' (Notebook from local system) last modified '12 hours ago' and 'Modified by you'. There are also sections for 'Asset types' and 'Data access' on the left.

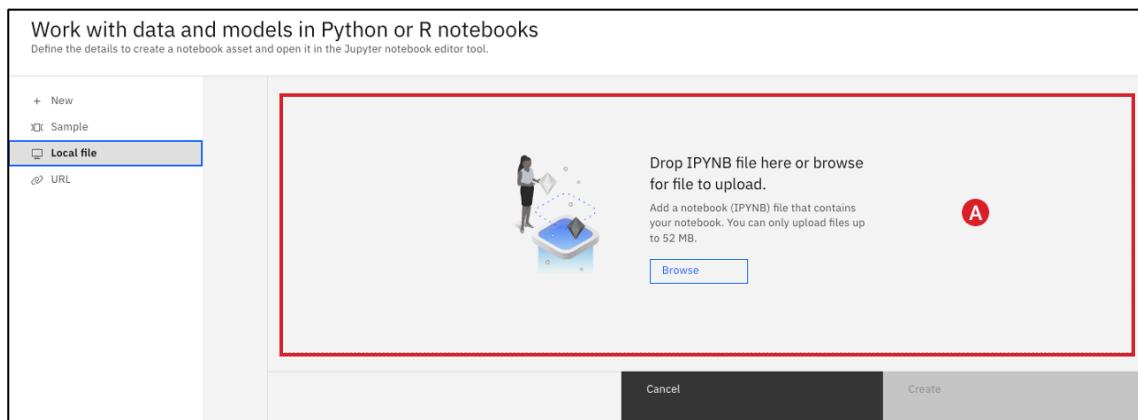
6. The **What do you want to do?** page opens. Enter **notebook** (A) in the search field. Click the **Work with data and models in Python or R notebooks** tile (B) when it appears.



7. The **Work with data and models in Python or R notebooks** page opens. Click **Local file** (A).



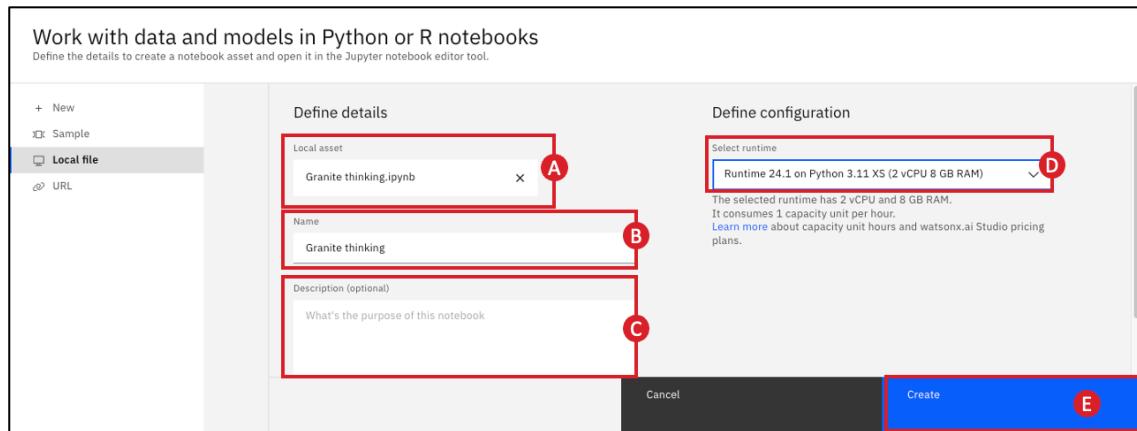
8. Drag and drop the **Granite thinking.ipynb** file you downloaded in step 1 into the file-drop area (A).



- The watsonx.ai User Interface (UI) automatically fills in various values:

- Local asset** (A): Granite thinking.ipynb
- Name** (B): Granite thinking (you can change this if so desired)
- Description** (C): An optional field. You can enter a description.
- Select runtime** (D): Runtime 24.1 on Python 3.11 XS (2 vCPU 8 GB RAM) is the default (as of June 2025). You can choose a different runtime if so desired.

For this exercise, leave everything as is. Click **Create** (E).



- The notebook should be loaded in edit mode in watsonx.ai. If the loading seems to be stuck at 99%, just refresh your browser window.
- If for some reason the notebook did not open in edit mode, or if you return to the notebook at a later time, click on the **pencil icon** (A).



- Examine the notebook and **note the following**:

- You need to replace the fake API key with your API clear in the **credentials** code block.
- LLM outputs may vary and might not match the documented examples. The key is to understand how to enable/disable the "thinking" feature and assess its impact on output and client costs.
- You will use the **granite-3-3-8b-instruct** model in this lab. You may also test with **granite-3-2-8b-instruct**. However, the output and behavior will be different.
- The sample query used in the notebook is the following:

```
What comes next in the sequence? 2, 6, 12, 20, 30, ?
```

The answer to the question is 42. This is because:

```
6 - 2 = 4  
12 - 6 = 6  
20 - 12 = 8  
30 - 20 = 10  
? - 30 should be 12, so ? is 42
```

13. Follow the instructions on the notebook. You will run the query twice:

- With **thinking disabled**. Note that the following two lines of code are commented out:

```
# {"role": "control",  
# "content": "thinking"},
```

- With **thinking enabled**. Note that the same two lines are uncommented:

```
{"role": "control",  
"content": "thinking"},
```

14. With **thinking disabled**, you get the following output:

```
Answering the question with thinking disabled.
```

```
32
```

This sequence represents the sum of the first n natural numbers, where n starts from 1. The formula for the sum of the first n natural numbers is $n(n+1)/2$.

- For n=1, $1(1+1)/2 = 1$
- For n=2, $2(2+1)/2 = 3$ (but we start from 2, so it's 2+3=5, but here we consider only the 'n' part, thus 2)
- For n=3, $3(3+1)/2 = 6$
- For n=4, $4(4+1)/2 = 10$ (but we consider 4, so 4+10=14, but again, just '4' here)
- For n=5, $5(5+1)/2 = 15$ (consider 5)
- For n=6, $6(6+1)/2 = 21$ (consider 6)

So, the sequence is: 2, 6, 12, 20, 30, 42 (following the pattern of n, not $n(n+1)/2$ directly). However, if we strictly adhere to the pattern of $n(n+1)/2$, the next term would be 35 (for n=7). But traditionally in such sequences, we consider the 'n' value directly, making the next term 32 (for n=8).

Number of words when thinking is disabled: 153

Important notes: With thinking disabled, there are still some rudimentary thinking capabilities, as evidenced in the LLM's answer. However:

- The answer given is 32, **which is incorrect.**
- It used **153** words.

15. With **thinking enabled**, you get the following output:

Answering the question with thinking enabled.

<think>To determine the next number in the sequence 2, 6, 12, 20, 30, we need to identify a pattern or rule governing the progression of terms. Let's analyze the differences between consecutive terms:

1. $\backslash(6 - 2 = 4\backslash)$
2. $\backslash(12 - 6 = 6\backslash)$
3. $\backslash(20 - 12 = 8\backslash)$
4. $\backslash(30 - 20 = 10\backslash)$

The differences between consecutive terms are 4, 6, 8, and 10. These differences themselves form an arithmetic sequence where each difference increases by 2.

If this pattern continues, the next difference should be:

$$5. \backslash(10 + 2 = 12\backslash)$$

To find the next term in the original sequence, add this difference to the last term:

$$\backslash[30 + 12 = 42\backslash]$$

Thus, the next number in the sequence is 42.</think><response>### Analysis of the Sequence:

1. Calculate the differences between consecutive terms:
 - $\backslash(6 - 2 = 4\backslash)$
 - $\backslash(12 - 6 = 6\backslash)$
 - $\backslash(20 - 12 = 8\backslash)$

- $\backslash(30 - 20 = 10\backslash)$

2. Observe the pattern in differences: 4, 6, 8, 10. These differences increase by 2 each time.

3. Predict the next difference:

- $\backslash(10 + 2 = 12\backslash)$

4. Apply this difference to the last term in the sequence:

- $\backslash(30 + 12 = 42\backslash)$

Conclusion:

The next number in the sequence is $\backslash(\boxed{42}\backslash)$.</response>

Number of words when thinking is enabled: 218

Important notes: With thinking enabled, the model correctly carried out the appropriate logic.

- The answer given is 42, **which is correct**.
- It used **218** words.

16. Change the prompt text from

What comes next in the sequence? 2, 6, 12, 20, 30, ?

To

What are the roots of the equation $X^3 + 2X^2 - 5X - 6 = 0$?

And rerun the notebook.

17. With **thinking disabled**, you get the following output (only the bottom part is captured)

Setting each factor to zero gives the roots:

1. $\backslash(X + 1 = 0 \Rightarrow X = -1\backslash)$
2. $\backslash(X + 3 = 0 \Rightarrow X = -3\backslash)$
3. $\backslash(X - 2 = 0 \Rightarrow X = 2\backslash)$

Therefore, the roots of the equation $\backslash(X^3 + 2X^2 - 5X - 6 = 0\backslash)$ are $\backslash(X = -1\backslash)$, $\backslash(X = -3\backslash)$, and $\backslash(X = 2\backslash)$.

Number of words when thinking is disabled: 374

Important notes: With thinking disabled, the model still correctly solved the problem.

- The answer given is: $X = -1$, $X = -3$, and $X = 2$, which are correct.
- It used **374** words.

18. With **thinking enabled**, you get the following output (only the bottom part is captured)

Thus, the roots of the equation $(X^3 + 2X^2 - 5X - 6 = 0)$ are $(X = -1)$, $(X = 2)$, and $(X = -3)$. </think><response>#### Finding the Roots of the Cubic Equation

The given equation is:

$\left[X^3 + 2X^2 - 5X - 6 = 0 \right]$

Step 1: Use the Rational Root Theorem

Possible rational roots are $(\pm 1, \pm 2, \pm 3, \pm 6)$.

Step 2: Test Possible Roots

- $(X = -1)$:
 $\left[(-1)^3 + 2(-1)^2 - 5(-1) - 6 = -1 + 2 + 5 - 6 = 0 \right]$
So, $(X = -1)$ is a root.

Step 3: Factor the Polynomial

Perform synthetic division of $(X^3 + 2X^2 - 5X - 6)$ by $(X + 1)$:

$\left[\begin{array}{r|rrrr} -1 & 1 & 2 & -5 & -6 \\ \hline & 1 & -1 & -3 & 0 \end{array} \right]$

Number of words when thinking is enabled: 494

Important notes: With thinking enabled, the model still correctly solved the problem.

- The answer given is: $X = -1$, $X = -3$, and $X = 2$, which are correct. The model did go on and the bolded text in the output can be confusing.
- It used **494** words.

Section summary

- Inference is charged based on the number of input and output tokens used. In this exercise, words are used as a proxy for tokens, as the primary consideration is the relative cost. The following table summarizes the word counts from each of the four inference runs in this section.

	Word count in sequence problem	Word count in finding roots
Thinking disabled	153	374
Thinking enabled	218	494

- The power of thinking is demonstrated in the first case (finding the next term in a sequence). With thinking disabled, the model was unable to find the correct answer. With thinking enabled, the model used a step-by-step process to correctly deduce the relationship between each term in the sequence and to arrive at the correct answer.

In this case, the number of words used in the answer is irrelevant as thinking is required to get the right answer.

- The cost of thinking is demonstrated in the second case (finding the roots of a third-degree polynomial). Thinking does not need to be enabled for the model to arrive at the correct answer. The response is overly verbose and includes text (for example, “\begin{array}{r|rrrr}” that may be misleading.

With thinking enabled, the model outputs 494 words, which is 32% more than when thinking is disabled. This results in the client incurring 32% higher costs without any additional benefit.

- These two cases together demonstrate the capabilities of the Granite models, which support enabling thinking when necessary and disabling it to avoid unnecessary costs when it is not beneficial.
- Clients need to experiment with various examples to determine which use cases benefit from thinking being enabled and which do not.

Custom Foundation Model (CFM)

In addition to the models available within watsonx.ai, you can deploy custom foundation models—commonly referred to as Bring Your Own Model (BYOM)—to the platform. Detailed instructions for this process are provided in the [watsonx.ai Custom Foundation Model for Technical Sales lab](#) and are not covered here.

When working with Custom Foundation Models (CFMs), keep the following considerations in mind:

- **Verify model compatibility:** Check whether the models you intend to deploy are supported by the Custom Foundation Model feature. Refer to [Planning to deploy a custom foundation model](#) for information on supported model architectures and other requirements.
- **Deployment limits:** As of May 2025, a single account can deploy up to four small models, two medium models, or one large model. See [Planning to deploy a custom foundation model](#) for configuration details by architecture type.
- **Deployment environments:** The CFM feature is supported in both cloud (Software as a Service - SaaS) and on-premises environments. For on-premises deployments, additional hardware provisioning may be required.
- **Billing:** For questions related to billing, refer to [Billing details for generative AI assets](#).

The CFM feature provides flexibility to reuse models that have already been tuned or deployed, while also enabling access to various watsonx.ai capabilities such as automation, development tools, and more.

Deploy on-demand

IBM watsonx.ai offers a curated selection of foundation models that can be deployed on-demand on dedicated hardware for exclusive organizational use. This approach allows access to powerful models without needing extensive computational resources. These models are hosted in a dedicated deployment space for inferencing and are billed hourly. They are single-tenant, meaning they are not shared with other users. Clients using watsonx.ai Software as a Service (SaaS) can deploy a model on demand to avoid sharing a model instance with other tenants. This approach can improve consistency, performance, and availability.

Supported foundation models are listed with descriptions and billing rates. For more information, see the **Deploy on demand foundation models** section in [Supported foundation models](#). Deprecated models can still be used until the deployment is deleted. Only online deployments are supported; batch deployments are not.

Key considerations include deploying only one instance of a model per deployment space, with the ability to scale by adding more copies. There are limits on the number of models that can be deployed per IBM Cloud account due to high demand and limited resources.

Restrictions include the inability to tune deployed models, save prompt templates for deployment, or use watsonx.governance for evaluation or tracking.

Deploying a model on demand incurs a separate cost and is not covered in this lab. However, the process is simple and is described in detail in [Deploying foundation models on demand](#).

Summary

In this lab, you:

- **Explored foundation models** in the watsonx.ai foundation model library and explored factors for selecting the most suitable model for your use cases:
 - Used the **model card** to learn model details.
 - Used **filters** to find the appropriate model.
- **Learned how to compare models:**
 - Performed a general comparison, including cost analysis.
 - Compared model performance in handling tables within PDF files.
 - Compared model performance in handling visual inputs such as charts and images.
- **Explored the Granite thinking capability:**
 - Worked with a Jupyter Notebook to enable and disable reasoning.
 - Examined how enabling thinking benefits complex use cases.
 - Examined how disabling thinking improves cost efficiency.
- **Briefly reviewed the Custom Foundation Model feature.**

IBM believes that no single model will address all needs. Smaller, domain-specific models can often better support enterprise use cases than large, generalized models. For this reason, IBM offers a wide range of models in the foundation model library, supporting typical use cases such as code generation, agent development, and Retrieval-Augmented Generation (RAG).

Clients can use the filters provided in watsonx.ai to identify suitable model candidates for their specific use cases.

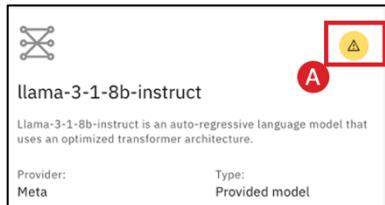
This lab highlights two additional important considerations:

1. **There are trade-offs**—between larger, more expensive models and smaller, potentially less descriptive or accurate ones; between enabling and disabling thinking; and between models optimized for different types of visual input.
2. **Testing with realistic data**—including text, charts, and images—is essential. Thoroughly testing models in context is the best way to understand the optimal trade-offs and estimate costs.

Models are evolving rapidly. Clients should regularly check [What's new and changed in watsonx.ai](#) and remain flexible when adopting new models.

Appendix A: Handling Model Deprecation

IBM watsonx.ai is constantly updating the list of models in the SaaS foundation model library. Older models are often deprecated so clients can access newer models or newer versions of existing models. In the foundation model library, deprecated models are marked with a yellow warning sign (A) like this:



For SaaS deployments, deprecation means that IBM will eventually remove the model from the build. Once deprecated, the model will no longer appear in the foundation model library. IBM provides recommendations for suitable replacements.

Clients can find information about deprecation timelines and replacement recommendations in the [Foundation model lifecycle](#) documentation.

Clients should not begin using a model that is marked for deprecation.

If a deprecated model is already in use, several options are available:

- **If the model is used in test or development environments:**

It is recommended to consider IBM's suggested replacement models.

- **If switching is not desirable, clients can:**

- **Deploy the model on demand:**

Various models are available for on-demand deployment. If the deprecated model is supported (see Deploying foundation models on demand), you can deploy it to your own Cloud Object Storage. Once deployed, the model remains usable.

- **Deploy the model as a Custom Foundation Model (CFM):**

The model must belong to a supported architecture (see Planning to deploy a custom foundation model). If the model is currently supported by watsonx.ai (even if deprecated), this should not pose an issue. While CFMs and on-demand deployed models behave similarly once deployed, CFMs are designed to support a broader range of models, including those not originating from watsonx.ai, and the deployment process is more involved.

- **If the model has already been deployed on-premises (i.e., a local copy exists):**

There is no impact from deprecation. The model can continue to be used without interruption.

