

Simplify Your AI Journey: Unleashing the Power of AI with IBM watsonx.ai

Deepak Rangarao

Phillip Gerrard

Charley Beller

Carl Broker

Daniele Comi

Lakshmana Ekambaram

Shuvanker Ghosh

Karen Medhat

Payal Patel

Matthew Price

Shirley Shum

Mark Simmonds



Artificial Intelligence

Data and AI



IBM Redbooks

**Simplify Your AI Journey: Unleashing the Power of AI
with IBM watsonx.ai**

January 2025

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (January 2025)

This edition applies to Version 2, Release 1, Modification x of IBM watsonx.ai.

© Copyright International Business Machines Corporation 2025. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Foreword	ix
Preface	ix
Authors	x
Now you can become a published author, too!	xii
Comments welcome	xii
Stay connected to IBM Redbooks	xiii
Chapter 1. Competing with artificial intelligence	1
1.1 Competing with AI	2
1.2 Challenges in building and deploying AI models	4
1.2.1 Technical considerations for building and deploying AI models	5
1.3 Opportunities around using AI on trusted data	5
1.3.1 Enhancing decision-making with accurate insights	6
1.3.2 Driving operational efficiency	6
1.3.3 Accelerating innovation	7
1.3.4 Enhancing governance and compliance	7
1.3.5 Unlocking new revenue streams	7
1.3.6 Transforming industries with AI and trusted data	8
1.4 Improving AI model reliability	8
1.4.1 Enabling cross-enterprise collaboration	9
1.4.2 Enhancing real-time decision-making	9
1.4.3 Scaling AI-driven ecosystems	10
1.4.4 Driving sustainability and environmental, social, and governance goals	10
1.4.5 Personalizing customer experiences	10
1.5 Creating new AI-enabled products and services	11
Chapter 2. Introducing IBM watsonx.ai	13
2.1 Overview of watsonx.ai	14
2.1.1 Key capabilities	14
2.1.2 The watsonx.ai architecture	14
2.1.3 watsonx.ai empowering IBM Software offerings	14
2.1.4 Benefits of using watsonx.ai for businesses	15
2.2 Synergy between watsonx.ai and other components in the watsonx platform	15
2.2.1 Synergy between watsonx.ai and watsonx.data	15
2.2.2 Synergy between watsonx.ai and watsonx.governance	16
2.3 Business impact of these synergies	16
Chapter 3. Tools for diverse data science teams	17
3.1 Key personas for watsonx.ai	18
3.1.1 Data scientists	18
3.1.2 Machine learning engineers	18
3.1.3 AI engineers	19
3.2 Low-code, no-code, and full-code tools	20
3.2.1 No-code, low-code, and full-code tools on IBM watsonx.ai	20
Chapter 4. Building and using artificial intelligence models	27

4.1 Prerequisites and assumptions	28
4.2 How to use this chapter.	28
4.3 Building and using AI models in watsonx.ai	28
4.3.1 Overview of the watsonx.ai platform	28
4.3.2 Key features and capabilities	29
4.4 Getting started with watsonx.ai: Setting up the environment	29
4.5 Data preparation and ingestion for AI model building	34
4.5.1 Understanding the importance of data in AI	34
4.5.2 Preparing and cleaning data: data quality considerations.	35
4.5.3 Handling missing data, outliers, and bias	35
4.5.4 Ingesting data into watsonx.ai Studio	36
4.5.5 Connecting to data repositories and cloud services	36
4.6 Building AI models in watsonx.ai.	38
4.6.1 Choosing the right model for your use case	38
4.6.2 Model creation workflow	39
4.7 Deploying AI models in watsonx.ai	40
4.7.1 watsonx.ai Studio deployments	40
4.8 watsonx.ai LLM deployment	45
4.8.1 Model packaging and exporting	45
4.9 Operationalizing machine learning and LLM models	50
4.9.1 Calling ML models by using API calls	50
4.9.2 Calling Prompt Lab LLM models by using API calls	52
4.9.3 IBM watsonx Assistant	53
4.10 Additional information and where to go next.	54
4.10.1 Additional support and documentation	55
4.10.2 watsonx.ai API reference	55
4.10.3 watsonx.ai data pipeline and orchestration.	55
Chapter 5. Advanced capabilities of watsonx.ai	57
5.1 Prompt engineering.	58
5.1.1 Prompting techniques	58
5.1.2 Importance of system tokens	59
5.1.3 Model-specific peculiarities	59
5.1.4 How watsonx.ai supports prompt engineering	60
5.2 Multitask prompt tuning.	61
5.2.1 Prompt tuning parameters	62
5.2.2 Interdependencies and holistic tuning strategies	63
5.3 Fine-tuning	64
5.3.1 Challenges with fine-tuning.	64
5.3.2 How watsonx.ai addresses fine-tuning challenges	65
5.4 InstructLab	67
5.4.1 Advantages of InstructLab	70
5.4.2 How to use InstructLab	71
5.4.3 InstructLab on watsonx.ai Software-as-a-Service.	79
5.4.4 InstructLab use case examples	84
Chapter 6. Artificial intelligence agents	87
6.1 What makes an AI agent.	88
6.2 Why AI agents are needed	94
6.3 Multiple AI agents	95
6.4 AI agents on watsonx.ai	100
6.5 AI agents use case examples	107
Chapter 7. Use cases	109

7.1 Using RAG to aid a medical school admissions office	110
7.1.1 The challenge	110
7.1.2 The solution	110
7.1.3 Special considerations	111
7.2 Embedding workflow automation to streamline recommendations	111
7.2.1 The challenge	111
7.2.2 The solution	112
7.2.3 Special considerations	113
Abbreviations and acronyms	115
Related publications	117
IBM Redbooks	117
Online resources	117
Help from IBM	118

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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, US

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 jurisdictions 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 provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, 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/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

Cloudant®	IBM Instana™	Orchestrate®
Cognos®	IBM Spectrum®	Redbooks®
DataStage®	IBM Watson®	Redbooks (logo)  ®
Db2®	Informix®	SPSS®
IBM®	InfoSphere®	Turbonomic®
IBM API Connect®	Instana®	z/OS®
IBM Cloud®	Netezza®	

The following terms are trademarks of other companies:

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Red Hat, Fedora, OpenShift, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

RStudio, and the RStudio logo are registered trademarks of RStudio, Inc.

Other company, product, or service names may be trademarks or service marks of others.

Foreword

This IBM Redbooks® publication is part of a trilogy that positions and explains [IBM watsonx](#), which is IBM's strategic artificial intelligence (AI) and data platform. Each book focuses on one of the three main components of the watsonx platform:

- ▶ IBM watsonx.ai: A next-generation enterprise studio for AI builders to train, validate, tune, and deploy both traditional machine learning (ML) and new generative AI (gen AI) capabilities that are powered by foundation models (FMs).
- ▶ IBM watsonx.data: A fit-for-purpose data store that is built on an open lakehouse architecture, and is optimized for different and governed data and AI workloads.
- ▶ IBM watsonx.governance: A set of AI Governance capabilities that enables trusted AI workflows, which help organizations implement and comply with ever-changing industry and government regulations.

Organizations have long recognized the value that IBM Redbooks publications provide in guiding them with best practices, frameworks, clear explanations, and use cases as part of their solution evaluations and implementations.

This trilogy of books was possible because of close collaboration among many skilled and talented authors that were selected from IBM Technical Sales, IBM Development, IBM Expert Labs, IBM Client Success Management, and consulting services organizations to use their diverse skills, experiences, and technical knowledge across the watsonx platform.

Thanks to the authors, contributors, reviewers, and the IBM Redbooks team for their dedication, time, and effort in making this publication a valuable asset that organizations can use as part of their journey to AI.

Thanks to Mark Simmonds and Deepak Rangarao for taking the lead in shaping this request into yet another successful IBM Redbooks project.

Steve Astorino, IBM General Manager - Development, Data, AI, and Sustainability

Preface

IBM watsonx is IBM's strategic AI and data platform. This book focuses on [watsonx.ai](#), one of the three main components of the platform. IBM watsonx.ai is a next-generation enterprise studio that you can use to train, validate (test), tune, and deploy both traditional ML and new gen AI capabilities, which are powered by FMs through an open and intuitive user interface (UI). This AI studio provides a range of FMs, training and tuning tools, and a cost-effective infrastructure that facilitates the entire data and AI lifecycle, from data preparation through model development, deployment, and monitoring. The studio also includes an FM library that provides IBM® curated and trained FMs. FMs use a large, curated set of enterprise data that is backed by a robust filtering and cleansing process, and with an auditable data lineage. These models are trained on language and other modalities, such as code, time-series data, tabular data, geospatial data, and IT events data.

Here are some examples of the model categories:

- ▶ **fm.code:** Models that automatically generate code for developers through a natural-language interface to boost developer productivity and enable the automation of many IT tasks.
- ▶ **fm.NLP:** A collection of large language models (LLMs) for specific or industry-specific domains that use curated data to help mitigate bias and quickly make domains customizable by using client data.
- ▶ **fm.geospatial:** Models that are built on climate and remote sensing data to help organizations understand and plan for changes in natural disaster patterns, biodiversity, land use, and other geophysical processes that might impact their businesses

The watsonx.ai studio builds on Hugging Face open-source libraries, which offer thousands of Hugging Face open models and datasets. Users can leverage the power of IBM Granite LLMs, along with the latest Mistral, Llama, and other third-party LLMs. It is part of IBM's commitment to deliver an open ecosystem approach that enables users to leverage the best models and architecture for their unique business needs.

This IBM Redbooks publication provides a broad understanding of watsonx.ai concepts, its architecture, and the services that are available with the product. Also, several common use cases and scenarios are included that should help you better understand the capabilities of this product. Code samples of common scenarios are available at [this GitHub repository](#). For more examples, which include using Instructlab and AI agents, see [this GitHub repository](#).

This publication is for watsonx customers who seek best practices and real-world examples of how to best implement their solutions while optimizing the value of their existing and future technology, AI, data, and skills investments.

Note: Here are the other books in the trilogy:

- ▶ *Simplify Your AI Journey: Ensuring Trustworthy AI with IBM watsonx.governance*, SG24-8573
- ▶ *Simplify Your AI Journey: Unleashing the Power of AI with IBM watsonx.data*, SG24-8570

Authors

This book was produced by a team of specialists from around the world:

Deepak Rangarao is an IBM Distinguished Engineer and CTO who is responsible for Technical Sales-Cloud Paks. He leads the technical sales team that helps organizations modernize their technology landscape with IBM Cloud Paks. He has broad, cross-industry experience in the data warehousing and analytics space from building analytic applications at large organizations and performing technical pre-sales for start-ups and large enterprise software vendors. Deepak has co-authored several books on many topics, such as OLAP analytics, change data capture, data warehousing, and object storage. He is a regular speaker at technical conferences. He is a certified technical specialist in Red Hat OpenShift, Apache Spark, Microsoft SQL Server, and web development technologies.

Phillip Gerrard is a Project Leader for the International Technical Support Organization working out of Beaverton, Oregon. As part of IBM for over 15 years, he has authored and contributed to hundreds of technical documents that were published to IBM.com, and worked directly with IBM's largest customers to resolve critical situations. As a team lead and subject matter expert (SME) for the IBM Spectrum® Protect support team, he is experienced in leading and growing international teams of talented IBM employees by developing and implementing team processes, and creating and delivering education. Phillip holds a degree in computer science and business administration from Oregon State University.

Charley Beller is a Principal Data Scientist and IBM Master Inventor. He works with clients as the Worldwide Solution Engineering Lead for watsonx.ai and AI Assistants within Technology Expert Labs. Charley has been working with IBM language technologies since joining the Watson group in 2014. He is an inventor with over 100 patents, and holds a PhD in Cognitive Science.

Carl Broker is an AI Architect at IBM who specializes in enterprise gen AI solutions. With a background in both gen AI and traditional data science, Carl leads design sessions and develops proof-of-concepts for clients. Before this role, he worked as an AI Engineer and Data Scientist at IBM, focusing on predictive modeling and AI-driven solutions. Carl holds a Master of Science degree from Johns Hopkins University.

Daniele Comi is a Data Scientist, AI Engineer, and Software Engineer at IBM Italy, with over 3 years of experience in data analytics, ML, and deep learning (DL). His expertise spans the entire spectrum of AI, from architectural design to scientific research, with a focus on ML, reinforcement learning (RL), and DL. Daniele holds a master's degree in Computer Science Engineering, with a specialty in AI frameworks and models. At IBM, Daniele has been a key member of the AI and gen AI team in Italy, where he has designed and implemented complex AI and gen AI architectures for many industry applications. His technical expertise also includes Fully Homomorphic Encrypted AI, which enables secure AI solutions that help ensure data privacy.

Lakshmana Ekambaram is an IBM Senior Technical Leader with over 30 years of experience in database development, advanced analytics, and building hybrid cloud solutions. He is part of the IBM Expert Labs SWAT organization where he leads the data fabric and trusted AI journey for customers worldwide. He has developed many IBM certification courses and co-authored books about data science, AI, and data fabric.

Shuvanker Ghosh is a certified Executive Architect and Worldwide Platform Leader for Data and AI in Worldwide Solution Architecture in IBM Technology Expert Labs. With 18 years of experience at IBM, he serves as a trusted advisor to clients, offering thought leadership on IBM's Data and AI portfolio. He guides organizations in their responsible AI journey, helping them adopt best practices. His current focus is on defining solution blueprints and architectural patterns that assist clients in addressing their business challenges through responsible and trustworthy AI solutions. He possesses extensive expertise in the IBM Data and AI portfolio, including the watsonx platform and IBM Cloud Pak for Data. Shuvanker has successfully led and delivered complex programs that involve multiple teams, providing technical management, architecture, technology thought leadership, and software development methodologies and processes. His experience spans various industries, including retail, finance, insurance, healthcare, telecommunications, and government.

Karen Medhat is a Customer Success Manager Architect in the UK and the youngest IBM Certified Thought Leader Level 3 Technical Specialist. She is the Chair of the IBM Technical Consultancy Group and an IBM Academy of technology member. She holds a MSc degree with honors in Engineering in AI and Wireless Sensor Networks from the Faculty of Engineering, Cairo University, and a BSc degree with honors in Engineering from the Faculty of Engineering, Cairo University. She co-creates curriculum and exams for different IBM professional certificates. She also created and co-created courses for the IBM Skills Academy in various areas of IBM technologies. She serves on the review board of international conferences and journals in AI and wireless communication. She also is an IBM Inventor who is experienced in creating applications architecture and leading teams of different scales to deliver customers' projects successfully. She frequently mentors IT professionals to help them define their career goals, learn new technical skills, or acquire professional certifications. She has authored publications on cloud, IoT, AI, wireless networks, microservices architecture, and blockchain.

Payal Patel works in Data and AI Technical Content Development at IBM, where she creates technical learning materials for sellers, IBM Business Partners, and clients to enable them to get the most value out of IBM Data and AI products and solutions. She has worked in various roles at IBM, which include marketing analytics and as a Solutions Architect in IBM Technology Expert Labs, with a focus on Data and AI. She has worked in various technical roles across the financial services, insurance, and technology industries. She holds a Bachelor of Science degree in Information Science from UNC Chapel Hill, and a Masters in Analytics degree from North Carolina State University.

Matthew Price is a Senior watsonx Client Success Manager with 20 years of experience in IT and 10 years of experience focusing on Watson technologies. His previous experience includes writing the base code that went on to become the IBM Watson® Assistant for Citizens application, which is IBM's no-charge offering that was released during 2020 to help business and government agencies navigate the pandemic. His previous publications centered on application migration and the cloud.

Shirley Shum is a Senior Software Engineer for the IBM Fusion team. She has worked as a technical lead on IBM Storage products, such as IBM Storage Insights and Fusion. Her areas of expertise include Kafka, complex event processing, backup and restore, and AI solutions, such as watsonx.ai and InstructLab on the Red Hat OpenShift platform.

Mark Simmonds is a Program Director with IBM Data and AI. He writes extensively on AI, data science, and data fabrics, and holds multiple author recognition awards. He previously worked as an IT architect leading complex infrastructure design and corporate technical architecture projects. He is a member of the British Computer Society, holds a bachelor's degree in Computer Science, is a published author, and a prolific public speaker.

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- ▶ Send your comments in an email to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, IBM Redbooks
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on LinkedIn:

<https://www.linkedin.com/groups/2130806>

- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/subscribe>

- ▶ Stay current on recent Redbooks publications with RSS Feeds:

<https://www.redbooks.ibm.com/rss.html>



Competing with artificial intelligence

In today's fast-paced digital landscape, artificial intelligence (AI) has emerged as a game-changer that is revolutionizing the way businesses operate, innovate, and compete. As AI technologies continue to advance and become increasingly ubiquitous, organizations are faced with the daunting task of competing with AI-driven rivals while also leveraging AI to stay ahead of the competition. This chapter delves into the world of competing with AI by exploring the challenges, opportunities, and strategies that organizations can employ to remain competitive in an AI-dominated market.

The following topics are described in this chapter:

- ▶ 1.1, “Competing with AI” on page 2
- ▶ 1.2, “Challenges in building and deploying AI models” on page 4
- ▶ 1.3, “Opportunities around using AI on trusted data” on page 5
- ▶ 1.4, “Improving AI model reliability” on page 8
- ▶ 1.5, “Creating new AI-enabled products and services” on page 11

1.1 Competing with AI

To harness the competitive advantage of using AI, organizations must first understand the AI landscape and the various types of AI that exist. Organizations must also be aware of the various AI technologies that are being used to drive innovation and competitiveness, which include the following ones:

- ▶ **Machine learning (ML):** ML represents a pivotal subset of AI where algorithms are developed and trained to recognize patterns and make data-driven predictions or decisions. Unlike traditional programming, ML systems learn iteratively from data, improving performance with experience. These systems rely on vast datasets to develop statistical models that enable predictions across diverse applications, such as anomaly detection, natural language processing (NLP), and image recognition. The training process involves feeding labeled (supervised learning) or unlabeled (unsupervised learning) data to the algorithm. Over time, the system refines its parameters to minimize errors and maximize predictive accuracy. ML algorithms are central to AI's practical applications, and they drive everything from recommendation systems to fraud detection in modern business ecosystems.
- ▶ **Deep learning (DL):** DL is an advanced branch of ML that employs artificial neural networks that are modeled after the human brain's structure and functioning. Unlike traditional ML, which often depends on manual feature engineering, DL automates the extraction of complex features from raw data through multiple layers of interconnected neurons. DL excels in handling unstructured data such as images, audio, and text, making it instrumental in solving tasks like computer vision, speech recognition, and language translation. For example, convolutional neural networks (CNNs) specialize in image processing by identifying spatial hierarchies in pixels, and recurrent neural networks (RNNs) and transformers tackle sequential data with unparalleled efficiency. By leveraging high-performance computing and large datasets, DL approximates nonlinear functions to enable machines to solve intricate, high-dimensional problems.
- ▶ **Unsupervised learning:** Unsupervised learning focuses on deriving patterns and structures from unlabeled datasets. This method trains algorithms to identify inherent groupings, clusters, or associations in data without human-provided annotations. Common techniques include clustering algorithms, such as k-means and hierarchical clustering, and dimensionality reduction methods, such as Principal Component Analysis (PCA) and t-SNE. Applications of unsupervised learning are vast, ranging from customer segmentation in marketing to anomaly detection in cybersecurity. These systems are particularly valuable in exploratory data analysis, where insights emerge from raw data without prior assumptions. By uncovering hidden relationships, unsupervised learning enhances your understanding of data and informs decision-making processes.
- ▶ **Reinforcement learning (RL):** RL is a paradigm of ML where algorithms learn optimal behaviors by interacting with an environment and receiving feedback in the form of rewards or penalties. RL systems employ agents that act based on policies, and aim to maximize cumulative rewards over time. Core to RL are concepts such as the exploration-exploitation tradeoff, Markov decision processes (MDPs), and value functions. Techniques such as Q-learning and Deep Q-Networks (DQNs) extend RL's capabilities to enable applications in robotics, game playing, and autonomous vehicles. RL's emphasis on learning through trial and error aligns it closely with real-world problem-solving, where dynamic environments require adaptive strategies.

- **Foundation models:** Foundation models (FMs) represent a transformative leap in AI and ML, characterized by their scalability, versatility, and ability to generalize across tasks. These models, such as Granite, are pre-trained on vast and diverse datasets, enabling them to adapt to specific applications with minimal fine-tuning. Unlike traditional ML models that are tailored to single tasks, FMs leverage their pre-trained knowledge to excel in multiple domains. This adaptability is achieved through transfer learning, where the model's pre-trained weights are fine-tuned on domain-specific data sets. FMs empower organizations to reduce the cost and complexity of training AI systems while achieving state-of-the-art performance in tasks like natural language understanding, summarization, and multimodal reasoning.

Leveraging AI for competitive advantage

To thrive in an AI-driven landscape, organizations must strategically harness AI technologies to drive innovation, enhance operational efficiency, and improve decision-making. Key areas of focus include the following ones:

- **AI-powered automation:** Automation that is fueled by AI enables the running of repetitive and high-volume tasks with speed and precision, such as applications in Robotic Process Automation (RPA), intelligent document processing, and automated workflows. By offloading routine operations, organizations can redirect human resources to strategic and creative endeavors, which foster innovation and competitive differentiation.
- **AI-driven analytics:** AI-powered analytics transform raw data into actionable insights, which equips businesses to make data-informed decisions. Predictive analytics, which is powered by ML, enables forecasting trends and identifying potential challenges, and prescriptive analytics suggests optimal courses of action. These capabilities enable organizations to stay ahead in dynamic markets by responding proactively to opportunities and risks.
- **AI-based innovation:** AI acts as a catalyst for innovation to enable organizations to conceptualize and deliver groundbreaking products, services, and business FMs. From personalized healthcare solutions to autonomous logistics systems, AI's potential to redefine industries is immense. By embedding AI in their innovation processes, companies can create unique value propositions that resonate with customers and stakeholders alike.

By embracing these AI strategies, organizations can position themselves as leaders in the era of digital transformation. As AI continues to evolve, its synergy with trusted data will unlock unprecedented opportunities, which will reshape the competitive landscape and drive sustainable growth.

1.2 Challenges in building and deploying AI models

Building and deploying AI models is a complex and challenging endeavor that requires expertise, resources, and infrastructure. Despite the potential benefits of AI, many organizations struggle to overcome the numerous hurdles that are associated with AI model development and deployment. Here are some of the key challenges:

- ▶ **Data quality and availability:** AI models require vast amounts of high-quality, relevant, and diverse data to learn, train, and validate. However, many organizations face challenges in collecting, processing, and integrating data from disparate sources, which can lead to issues with data quality, consistency, and availability. Furthermore, data privacy and security concerns can limit access to sensitive data, which can hinder the development of accurate and reliable AI models.
- ▶ **Model complexity and interpretability:** As AI models become increasingly complex, they can be difficult to interpret and understand, which makes it challenging to identify biases, errors, or flaws in the decision-making process. The lack of transparency and explainability in AI models can lead to mistrust, regulatory issues, and reputational damage. Moreover, the complexity of AI models can make it difficult to integrate them with existing systems, processes, and infrastructure.
- ▶ **Talent acquisition and retention:** The development and deployment of AI models require specialized skills and expertise, which include data science, ML, and software engineering. However, the demand for AI talent far exceeds the supply, which can lead to challenges in acquiring and retaining top talent. Furthermore, the constant evolution of AI technologies means that professionals must continually update their skills to remain relevant, which can add to the talent acquisition and retention challenges.
- ▶ **Infrastructure and scalability:** AI models require significant computational resources, memory, and storage to process and analyze large datasets. However, many organizations lack the necessary infrastructure to support the development and deployment of AI models, which can lead to issues with scalability, performance, and reliability. Furthermore, the integration of AI models with existing systems and processes can be complex, requiring significant investment in infrastructure and architecture.
- ▶ **Bias and fairness:** AI models can perpetuate and amplify existing biases and inequalities if they are trained on biased data or designed with a particular world view. The lack of diversity and inclusion in AI development teams can exacerbate these issues, which can lead to unfair outcomes and reputational damage. Furthermore, the identification and mitigation of bias in AI models can be challenging, which requires significant expertise and resources.
- ▶ **Regulatory compliance:** The development and deployment of AI models are subject to various regulations and laws, which include data protection, intellectual property, and anti-discrimination legislation. However, the rapidly evolving nature of AI technologies can make it challenging to ensure regulatory compliance, particularly in industries with strict regulations, such as healthcare and finance.
- ▶ **Model maintenance and updates:** AI models require continuous maintenance and updates to help ensure that they remain accurate, reliable, and relevant. However, the process of updating AI models can be complex and require significant resources and expertise. Furthermore, the integration of updated AI models with existing systems and processes can be challenging, which can lead to issues with compatibility and performance.

- ▶ **Explainability and transparency:** The lack of explainability and transparency in AI models can make it challenging to understand the decision-making process, which can lead to mistrust and reputational damage. Furthermore, the identification and mitigation of errors or biases in AI models can be difficult, which can require expertise and resources.
- ▶ **Cybersecurity:** AI models can be vulnerable to cyberthreats, such as data poisoning, model hijacking, and adversarial attacks. The identification and mitigation of these threats can be challenging and require expertise and resources. Furthermore, the integration of AI models with existing security systems and processes can be complex, which can lead to issues with compatibility and performance.

1.2.1 Technical considerations for building and deploying AI models

When building and deploying AI models, there are several technical considerations that must be accounted for:

- ▶ **Data preprocessing:** Ensuring that the data that is used to train and test the model is accurate, complete, and relevant.
- ▶ **Model selection:** Choosing the most suitable algorithm and model architecture for the problem being addressed.
- ▶ **Hyper-parameter tuning:** Optimizing the model's hyper-parameters to achieve the best possible performance.
- ▶ **Model evaluation:** Evaluating the model's performance by using metrics such as accuracy, precision, and recall.
- ▶ **Model deployment:** Deploying the model in a production-ready environment, such as a cloud-based API or a containerized application.
- ▶ **Model monitoring:** Continuously monitoring the model's performance and updating it as necessary to help ensure that it remains accurate and relevant.

In addition to these technical considerations, organizations must also consider the following items:

- ▶ **Data governance:** Establishing policies and procedures for data management, which include data quality, security, and compliance.
- ▶ **Model governance:** Establishing policies and procedures for model development, deployment, and maintenance, which include model validation, testing, and updating.
- ▶ **Infrastructure governance:** Establishing policies and procedures for infrastructure management, which include infrastructure provisioning, scaling, and security.

By being conscious of these technical considerations and establishing effective governance policies and procedures, organizations can help ensure the successful development and deployment of AI models that drive business value and competitiveness.

1.3 Opportunities around using AI on trusted data

AI thrives on data. However, the effectiveness of AI systems is not solely dependent on the volume of data but also on its quality and trustworthiness. Trusted data (data that is accurate, consistent, secure, and compliant) serves as the foundation for reliable AI-driven insights. Organizations today are exploring opportunities to harness AI on trusted data to drive operational efficiency, enhance decision-making, and unlock new revenue streams. This section explores the myriad possibilities that AI unlocks when it operates on a foundation of high-quality, trusted data.

1.3.1 Enhancing decision-making with accurate insights

The fusion of AI and trusted data is reshaping decision-making processes across industries, and enabling organizations to derive precise and actionable insights. This transformation is critical in domains where decisions significantly impact outcomes, such as healthcare, finance, and supply chain management. By leveraging high-quality, trusted data, AI systems can identify patterns, predict outcomes, and provide recommendations that drive superior decisions.

For example, in healthcare, AI systems that are powered by trusted clinical and patient data enhance diagnostic precision, predict patient outcomes with remarkable accuracy, and suggest personalized treatment plans that are tailored to individual needs. In the financial sector, AI models that are trained on trusted datasets excel in detecting fraudulent activities, assessing credit risks, and automating sophisticated trading strategies that are based on dynamic market trends. These examples illustrate how trusted data amplifies the reliability and impact of AI-driven decision-making, minimizing risks and maximizing outcomes.

One of the most transformative applications of AI on trusted data is in improving decision-making. High-quality data enables AI algorithms to deliver precise and actionable insights, which are beneficial in industries like healthcare, finance, and supply chain management, where even minor errors in decision-making can have significant consequences.

- ▶ **Healthcare:** Trusted data enables AI systems to accurately predict patient outcomes, suggest personalized treatment plans, and enhance diagnostic accuracy.
- ▶ **Finance:** In financial services, AI models that are trained on trusted data can detect fraud, assess credit risks, and automate trading strategies based on market predictions.

1.3.2 Driving operational efficiency

AI's ability to automate and optimize complex processes is magnified when it is built on a foundation of trusted data. By eliminating inefficiencies and reducing the need for human intervention, organizations can achieve unprecedented levels of operational efficiency.

In the manufacturing and energy sectors, predictive maintenance systems leverage trusted sensor data to foresee equipment failures, which enable preemptive interventions that minimize downtime and reduce maintenance costs. Similarly, AI-powered customer service platforms, which are underpinned by reliable customer interaction data, provide accurate, context-aware responses that deliver personalized experiences while alleviating the workload of human agents. These advancements highlight the transformative potential of combining AI with trusted data to streamline operations across industries.

When AI is applied to trusted data, it automates complex processes, which reduce the need for human intervention, and improves efficiency:

- ▶ **Predictive maintenance:** In the manufacturing and energy sectors, AI systems use trusted sensor data to predict equipment failures, which minimize downtime and optimizes maintenance schedules.
- ▶ **Customer service automation:** AI-powered chatbots, which are fueled by reliable customer data, provide accurate responses and deliver personalized experiences, which reduce the burden of human agents.

1.3.3 Accelerating innovation

The convergence of AI and trusted data is a catalyst for innovation, which unlocks hidden patterns and opportunities that were previously inaccessible. By analyzing vast amounts of high-quality data, AI systems empower organizations to develop groundbreaking products and solutions.

For example, in product development, companies use AI to analyze customer feedback, market trends, and usage data that is extracted from trusted sources to create offerings that resonate with consumer preferences. In the realm of scientific research, AI accelerates discovery processes by interpreting extensive experimental datasets, which lead to advancements in fields such as drug development and material science. Trusted data enhances the accuracy of these insights and helps ensure the reproducibility of outcomes, which drive sustained innovation.

The combination of AI and trusted data fosters innovation by uncovering hidden patterns and opportunities that were previously inaccessible:

- ▶ **Product development:** Companies leverage AI to analyze customer feedback and market trends from trusted datasets, which helps the companies to design products that align with consumer preferences.
- ▶ **Research and development:** In scientific research, AI accelerates discovery by analyzing large volumes of trusted experimental data, which can lead to breakthroughs in areas such as drug discovery and material science.

1.3.4 Enhancing governance and compliance

In an era where regulatory landscapes are becoming increasingly stringent, trusted data plays a pivotal role in helping ensure that AI systems operate within legal and ethical frameworks. Governance and compliance initiatives are fortified by AI systems that continuously monitor operations, detect anomalies, and flag potential risks.

For example, compliance monitoring AI tools analyze operational data to help ensure adherence to industry regulations and standards, which reduce the risk of noncompliance penalties. Also, the usage of diverse and representative trusted datasets mitigates biases in AI model training, which foster fairness and ethical outcomes in critical applications such as hiring or loan approvals. Trusted data serves as a cornerstone for responsible AI development and deployment.

Trusted data helps ensure that AI systems operate within legal and ethical boundaries, which are critical factors in maintaining customer trust and avoiding regulatory penalties:

- ▶ **Compliance monitoring:** AI models can continuously analyze operational data to help ensure adherence to regulations by flagging any potential compliance risks.
- ▶ **Bias mitigation:** Trusted data, when diverse and representative, helps train AI models that are fair and unbiased, which helps ensure ethical outcomes in areas like hiring or loan approvals.

1.3.5 Unlocking new revenue streams

The monetization of trusted data through AI-driven services and products has emerged as a significant avenue for revenue generation. Organizations across sectors are capitalizing on this synergy to create innovative business models.

For example, in telecommunications and retail, companies offer AI-powered insights or analytics as services to their partners and clients, which transform data into a valuable asset. Moreover, trusted customer data enables hyper-targeted marketing campaigns, which enhance conversion rates and foster customer loyalty. By harnessing the power of trusted data, organizations can unlock untapped revenue opportunities while delivering value to stakeholders.

Organizations are monetizing their trusted data through AI-driven services and products:

- ▶ **Data monetization:** Companies in sectors such as telecommunications and retail generate new revenue by offering AI-powered insights or analytics as a service to their partners and clients.
- ▶ **Personalized marketing:** AI leverages trusted customer data to deliver hyper-targeted marketing campaigns that increase conversion rates and customer loyalty.

1.3.6 Transforming industries with AI and trusted data

The integration of AI with trusted data is revolutionizing industries in unique and profound ways. Retailers use transaction and customer data to fuel recommendation engines, which enhance sales and customer satisfaction. In agriculture, AI models analyze environmental and crop data to optimize farming practices and maximize yields. Similarly, in the energy sector, AI systems leverage consumption and grid data to predict demand, optimize distribution, and enhance energy efficiency. These transformative applications underscore the versatility and impact of trusted data-driven AI across diverse domains.

Different industries are leveraging AI and trusted data in unique ways:

- ▶ **Retail:** Trusted transaction and customer data power recommendation engines that boost sales and improve customer experiences.
- ▶ **Agriculture:** AI models analyze trusted environmental and crop data to optimize farming practices and increase yield.
- ▶ **Energy:** AI systems use trusted consumption and grid data to predict demand and optimize energy distribution.

1.4 Improving AI model reliability

The reliability and interpretability of AI models are enhanced when they are trained and validated on trusted data. High-quality data helps ensure that AI systems deliver consistent and accurate outputs, which foster stakeholder trust and facilitate broader adoption.

For example, explainable AI (XAI) models rely on trusted data to generate transparent and interpretable insights, which address concerns about the “black-box” nature of AI. Also, trusted data simplifies model auditing and debugging by enabling the identification of inconsistencies and anomalies, which leads to continuous performance improvements. By prioritizing data quality, organizations can overcome one of the primary challenges of scaling AI systems.

Trusted data enhances the reliability and interpretability of AI models, addressing one of the major challenges in deploying AI at scale:

- ▶ **Explainability and trust:** AI models that are trained on high-quality data provide more consistent and interpretable outputs, which enable stakeholders to trust and adopt AI-driven solutions.
- ▶ **Model auditing and debugging:** Trusted data helps identify inconsistencies and anomalies in AI predictions, which make it easier to debug models and improve their performance over time.

1.4.1 Enabling cross-enterprise collaboration

Trusted data serves as a bridge for collaboration across organizational silos and with external partners. This capability enhances operational transparency and fosters innovation by enabling seamless data sharing and integration.

Enterprises can leverage trusted data infrastructures to break down silos so that cross-functional teams can work collaboratively on shared objectives. Secure data exchange mechanisms further facilitate partnerships across geographies, which helps ensure compliance with data privacy regulations and fosters trust among stakeholders. Such collaborative ecosystems are critical for driving comprehensive digital transformation initiatives.

AI on trusted data enables organizations to collaborate more effectively across departments and even with external partners:

- ▶ **Data sharing across silos:** Enterprises can break down data silos and enable cross-functional collaboration to enhance operational transparency.
- ▶ **Secure data exchange:** Trusted data infrastructures help ensure that shared data between partners or across geographies remain secure and compliant.

1.4.2 Enhancing real-time decision-making

The ability to make real-time decisions is a cornerstone of modern business strategies, and trusted data is a key enabler of this capability. By processing and analyzing data streams in real time, AI systems empower organizations to act swiftly and effectively.

In industries like finance, dynamic pricing models use real-time trusted data to optimize stock-pricing strategies based on demand and inventory levels. Financial institutions employ AI systems to perform instant risk assessments to mitigate fraud and help ensure secure transactions. These applications demonstrate how trusted data enhances the agility and responsiveness of AI-driven decision-making processes.

Real-time analytics that are powered by trusted data enable businesses to make faster, more informed decisions:

- ▶ **Dynamic pricing:** In e-commerce or travel industries, AI leverages real-time trusted data to optimize pricing strategies based on demand and inventory levels.
- ▶ **Real-time risk assessment:** Financial institutions use AI to perform instant risk assessments for transactions to help mitigate fraud or credit risks.

1.4.3 Scaling AI-driven ecosystems

Large-scale AI implementations depend on the robustness and reliability of trusted data ecosystems. By building scalable infrastructures that integrate trusted data with advanced AI capabilities, organizations can unlock the full potential of AI-driven solutions.

AI as a Service (AlaaS) platforms exemplify this integration by offering modular services such as ML models, NLP, and predictive analytics that are powered by trusted data. In parallel, the integration of IoT devices with AI systems generates vast volumes of real-time data to enable actionable insights in sectors like logistics, healthcare, and smart cities. These scalable ecosystems are the foundation for sustained growth and innovation.

Trusted data serves as the backbone for large-scale AI implementations, fostering robust AI ecosystems:

- ▶ AI as a Service (AlaaS): Companies are building scalable platforms where trusted data powers modular AI services like ML models, NLP, and predictive analytics.
- ▶ Integration with IoT: IoT devices generate vast amounts of data. Trusted IoT data enables AI systems to deliver real-time analytics for industries like logistics, healthcare, and smart cities.

1.4.4 Driving sustainability and environmental, social, and governance goals

AI-powered sustainability initiatives are gaining momentum, with trusted data playing a central role in achieving environmental, social, and governance (ESG) objectives. By analyzing environmental and operational datasets, AI systems help organizations optimize resource usage, reduce carbon footprints, and enhance supply chain transparency.

For example, sustainability analytics tools use trusted data to identify inefficiencies and recommend strategies for improving energy efficiency. Similarly, AI systems provide visibility into supply chains to enable organizations to address waste and improve sustainability practices. By aligning AI initiatives with ESG goals, organizations can demonstrate their commitment to responsible and ethical operations.

AI, combined with trusted data, helps organizations meet ESG objectives:

- ▶ Sustainability analytics: AI models analyze trusted environmental and operational data to optimize resource usage and reduce carbon footprints.
- ▶ Supply chain transparency: Trusted data provides visibility into the supply chain so that AI can identify inefficiencies, reduce waste, and improve sustainability practices.

1.4.5 Personalizing customer experiences

Customer-centric industries are leveraging AI's ability to deliver highly personalized experiences, which is a capability that is rooted in trusted data. By analyzing customer behavior, preferences, and interactions, AI systems create tailored experiences that enhance satisfaction and loyalty.

For example, adaptive AI systems use real-time trusted data to modify services dynamically to help ensure relevance and engagement. Behavioral analytics enable companies to predict customer needs, which reduce churn and fosters long-term relationships. Trusted data empowers organizations to elevate customer experiences to new heights.

Customer-centric industries are capitalizing on AI's ability to deliver highly personalized experiences through trusted data:

- ▶ Adaptive AI systems: Real-time, trusted customer data enables AI systems to adapt and tailor services to create more engaging user experiences.
- ▶ Behavioral analytics: Trusted behavioral data enables companies to predict customer preferences, which reduce churn and increase satisfaction.

1.5 Creating new AI-enabled products and services

Trusted data serves as the foundation for developing innovative AI-enabled products and services that redefine industries. By harnessing historical and real-time data, organizations can anticipate needs and deliver solutions proactively.

For example, proactive support systems leverage AI to predict and address issues before they escalate, which enhances customer satisfaction and operational efficiency. Custom AI solutions, which are tailored to specific market niches, further demonstrate the transformative potential of trusted data. As organizations continue to invest in data governance and quality, the opportunities for creating AI-driven innovations will expand.

By aligning AI initiatives with business objectives and prioritizing trusted data infrastructures, organizations can unlock unparalleled levels of efficiency, innovation, and growth. As the complexity of data landscapes continues to evolve, the role of trusted data in enabling AI to achieve its full potential becomes increasingly indispensable.

Trusted data opens doors to innovations that can redefine industries:

- ▶ Proactive support systems: AI systems, which are trained on historical and real-time data, predict customer or machine needs before problems occur to offer preemptive solutions.
- ▶ Custom AI solutions: Organizations use their proprietary trusted data to build AI products that are tailored to niche market requirements.

By continuously investing in trusted data infrastructures and aligning AI initiatives with business goals, organizations can unlock new levels of efficiency, growth, and innovation. AI and trusted data offers a powerful opportunity to drive growth, innovation, and operational excellence. Organizations that prioritize data governance, ensure data quality, and build AI systems on trusted data are better positioned to harness these opportunities. As the volume and complexity of data continue to grow, the role of trusted data in enabling AI to deliver its full potential will become more critical.



Introducing IBM watsonx.ai

IBM watsonx is IBM's next-generation platform that is designed to help businesses accelerate their journey into artificial intelligence (AI)-driven insights, decision-making, and automation. The platform offers a comprehensive suite of tools and services that are tailored to simplify and streamline the development and deployment of AI solutions. It is built on three foundational pillars:

- ▶ **IBM watsonx.data:** A scalable data lakehouse that is designed for efficient and secure data management to enable hybrid cloud deployments and optimize data for AI workloads.
- ▶ **IBM watsonx.governance:** Provides robust governance to help ensure that AI models remain ethical, transparent, and compliant with regulatory standards. It also helps businesses monitor and mitigate AI-related risks.
- ▶ **IBM watsonx.ai:** A cutting-edge AI development and deployment environment. It supports the full lifecycle of AI, from model training and fine-tuning to deployment and monitoring.

The seamless integration of these components enables enterprises to leverage trusted data (watsonx.data), enforce governance and ethical standards (watsonx.governance), and develop AI-powered solutions (watsonx.ai). Together, they form a comprehensive ecosystem for deploying enterprise-grade AI at scale.

The following topics are described in this chapter:

- ▶ 2.1, "Overview of watsonx.ai" on page 14
- ▶ 2.2, "Synergy between watsonx.ai and other components in the watsonx platform" on page 15
- ▶ 2.3, "Business impact of these synergies" on page 16

2.1 Overview of watsonx.ai

watsonx.ai serves as the core engine of the watsonx platform, which is focused on the rapid development and deployment of AI models. Its architecture is designed to support various AI workloads, such as traditional machine learning (ML), deep learning (DL), and generative AI (gen AI).

2.1.1 Key capabilities

Here are the key capabilities of watsonx.ai:

- ▶ **Foundation models (FMs):** watsonx.ai provides access to pre-trained FMs, such as large language models (LLMs) and vision models, which can be fine-tuned for specific business needs.
- ▶ **Machine learning operations (MLOps):** The platform integrates tools for version control, model monitoring, and deployment to streamline AI lifecycle management.
- ▶ **Multi-cloud compatibility:** Supports hybrid and multi-cloud environments so that businesses can run AI workloads wherever they see fit.
- ▶ **Extensibility:** Developers can bring their own models or integrate open-source frameworks like PyTorch and TensorFlow.

2.1.2 The watsonx.ai architecture

The watsonx.ai architecture components include the following items:

- ▶ **Model Studio:** An interface for training, fine-tuning, and deploying models.
- ▶ **Inference Engine:** Optimized for running AI models in production environments to help ensure low latency and high scalability.
- ▶ **Integration Layer:** Enables seamless integration with watsonx.data for real-time data access and watsonx.governance for compliance.

2.1.3 watsonx.ai empowering IBM Software offerings

The watsonx.ai FMs are being infused throughout all of IBM's major software offerings. The FMs are as follows:

- ▶ **IBM watsonx Code Assistant:** Uses gen AI so that developers can automatically generate code by using a natural-language prompt.
- ▶ **IBM watsonx AIOps Insights:** Includes FMs for code and natural language processing (NLP) to provide greater visibility into performance across IT environments.
- ▶ **IBM watsonx Assistant and IBM watsonx Orchestrate®:** Boosted by an NLP FM, IBM's digital labor products enhance employee productivity and customer service experiences.
- ▶ **Environmental Intelligence Suite:** powered by the IBM geospatial FM, IBM EIS Builder Edition creates tailored solutions that address and mitigate environmental risks.

2.1.4 Benefits of using watsonx.ai for businesses

Adopting watsonx.ai offers several key advantages for enterprises looking to harness the power of AI:

- ▶ **Accelerated AI development:** watsonx.ai simplifies the development process with pre-trained FMs and built-in tools for training and deployment. Businesses can achieve faster time-to-value by reducing the complexity of building AI from scratch.
- ▶ **Enhanced productivity and efficiency:** Through automation of repetitive tasks, watsonx.ai enables teams to focus on higher-value activities. gen AI capabilities can handle complex processes, which improve the mean time to resolution for IT incidents and streamlining customer service.
- ▶ **Scalable and cost-efficient:** watsonx.ai support for hybrid cloud and open architecture provides cost flexibility. Businesses can choose the most economical deployment environment and scale AI workloads as needed.
- ▶ **Trust and governance:** With watsonx.governance tightly integrated, watsonx.ai helps ensure that AI models operate transparently and ethically. Businesses can meet regulatory compliance standards, which mitigate risks that are associated with biased or unexplainable AI decisions.
- ▶ **Business innovation:** watsonx.ai enables companies to explore new AI-driven opportunities, such as personalizing customer experiences, optimizing supply chains, and driving data-driven decision-making.

watsonx.ai is a transformative tool that empowers businesses to unlock the full potential of AI. By integrating seamlessly with watsonx.data and watsonx.governance, it offers a unified platform that combines innovation, efficiency, and compliance. Organizations adopting watsonx.ai can expect to gain a competitive edge through smarter automation, better decision-making, and faster scaling of AI solutions.

2.2 Synergy between watsonx.ai and other components in the watsonx platform

This section covers the following topics:

- ▶ Synergy between watsonx.ai and watsonx.data
- ▶ Synergy between watsonx.ai and watsonx.governance

2.2.1 Synergy between watsonx.ai and watsonx.data

watsonx.ai and watsonx.data streamline the development and deployment of AI models by ensuring that AI systems are powered by high-quality, trusted data. Here is how their synergy creates value:

- ▶ **Efficient AI model development:** watsonx.data provides a robust and scalable data lakehouse that is optimized for AI workloads. This lakehouse helps ensure that watsonx.ai has instant access to vast amounts of clean, organized, and queryable data, which accelerates training and fine-tuning of AI models.
- ▶ **Real-time data for AI:** watsonx.data facilitates real-time data streaming, which enables watsonx.ai to build and run AI models on up-to-date information. This approach enables dynamic AI use cases, such as predictive maintenance and fraud detection.

- ▶ Hybrid and multi-cloud flexibility: Both watsonx.ai and watsonx.data support deployment across hybrid and multi-cloud environments, which help ensure scalability and cost efficiency while keeping data sovereignty intact.
- ▶ Unified data governance: With watsonx.data acting as the backbone, organizations can ensure data integrity, enhance data sharing, and maintain a single source of truth for AI models that are developed in watsonx.ai.

2.2.2 Synergy between watsonx.ai and watsonx.governance

The relationship between watsonx.ai and watsonx.governance helps ensure that AI models are high-performing, compliant, ethical, and transparent. Here is how they complement each other:

- ▶ Ethical AI deployment: watsonx.governance provides guardrails for AI models that are developed and deployed through watsonx.ai. These guardrails include bias detection, explainability, and compliance tracking to help ensure that AI decisions are fair and aligned with regulatory standards.
- ▶ Lifecycle management and monitoring: watsonx.governance tracks the entire lifecycle of AI models by monitoring performance, drift, and adherence to governance policies. This approach enables watsonx.ai users to maintain the integrity of deployed models over time.
- ▶ Transparency and auditability: Models that are built on watsonx.ai benefit from the watsonx.governance robust reporting and audit capabilities, which provide stakeholders with clear insights into how AI models make decisions, which help ensure trustworthiness.
- ▶ Risk mitigation and remediation: If there are anomalies or breaches in governance policies, watsonx.governance enables quick remediation. This capability is crucial for mission-critical applications where trust in AI outputs is paramount.

2.3 Business impact of these synergies

By leveraging the combined strengths of watsonx.ai, watsonx.data, and watsonx.governance, enterprises can achieve the following goals:

- ▶ Help ensure that their AI models are trained on trusted, compliant data.
- ▶ Maintain high performance and ethical standards across AI deployments.
- ▶ Accelerate innovation while minimizing the risks that are associated with AI adoption.

This holistic approach empowers organizations to maximize ROI on their AI investments and gain a competitive edge in their industries.



Tools for diverse data science teams

Data science teams today are diverse in terms of skill sets, backgrounds, and experiences. These teams are also diverse in terms of the types of solutions that are implemented. Common roles of data science teams are data scientists, machine learning (ML) engineers, and artificial intelligence (AI) engineers. Therefore, different types of tools and solutions are needed to support data science teams.

This chapter describes a few of the key personas for IBM watsonx.ai and how the different types of tools that are available on watsonx.ai support these individuals in their day-to-day work.

The following topics are described in this chapter:

- ▶ 3.1, “Key personas for watsonx.ai” on page 18
- ▶ 3.2, “Low-code, no-code, and full-code tools” on page 20

3.1 Key personas for watsonx.ai

Many organizations are building data science teams. Depending on the level of data science maturity within the organization, the types of roles and the experience of individuals in these roles can vary. Common roles in data science teams include data analysts, data scientists, ML engineers, and AI engineers. Other roles in organizations that can benefit from IBM watsonx.ai include, but are not limited to, data science leaders, directors of enterprise architecture, and line-of-business users. This section goes through a few of these key personas by providing an overview of each role's responsibilities, common challenges that individuals in this role often face, and how IBM watsonx.ai is designed to enable individuals in these roles.

3.1.1 Data scientists

Data scientists use data to solve problems and improve decision making within an organization. Depending on the team and organization, their responsibilities include data collection, model development, data analysis, communication of findings, and providing recommendations. Data scientists often have a background in mathematics, specifically statistics and linear algebra, and programming.

Individuals in these roles often face challenges that are related to a lack of self-service access to the correct, or accurate data, for cleaning, transforming, and generating insights. They also lack integrated tools across the model lifecycle, and depending on their level of experience and skill set, they might lack experience with creating models by using certain tools or programming languages.

IBM watsonx.ai addresses this situation with various tools within the platform to provide data scientists with the flexibility that they need to build models. These tools include no-code, low-code, and full-code solutions such as AutoAI, IBM SPSS® Modeler, and Jupyter Notebooks. watsonx.ai provides many tools to select from because the correct tool to use varies based on factors such as the individuals' level of expertise and the project requirements.

Data scientists often collaborate with others in their team, such as business stakeholders, data science leaders, or fellow data scientists. watsonx.ai enables storing and sharing of assets among users within an organization through projects. In watsonx.ai, *projects* are collaborative workspaces where individuals can work with and share data and other assets to accomplish a specific goal.

3.1.2 Machine learning engineers

ML engineers work with data scientists and other IT experts, such as software developers to automate and move ML models into production. Typically, ML engineers have a background in computer science, mathematics, statistics, or software engineering. ML engineers are responsible for the data science pipeline, which can include sourcing and preparing data, building and training models, deploying models to production, and maintaining and improving existing ML systems.

Common challenges that are faced by ML engineers include difficulty in defining short and long-term goals, difficulty in scaling ML models, general lack of support for the services that are used by various teams, and incompatibility between tools.

IBM watsonx.ai helps overcome these challenges by providing various no-code, low-code, and full-code tools, which are compatible with each one. Also, with IBM Watson Machine Learning on IBM watsonx.ai, the model deployment process is simplified through capabilities such as deployment spaces. Regardless of the tool on watsonx.ai that is used to develop the model, ML engineers can deploy the model, which makes it simpler to deploy and manage ML assets.

3.1.3 AI engineers

AI engineers are responsible for developing, maintaining, and implementing AI systems. Common tasks that are performed by AI engineers include building and maintaining AI systems, and tuning AI models. AI engineers generally have a background in computer science, mathematics, software engineering, or programming.

Because the AI engineer role is new to many organizations, a common challenge many teams and individuals in this role face involves having varying levels of technical skills and experience to implement AI solutions. Also, other challenges include selecting and fine-tuning models for a specific use case, and managing the cost to implement and maintain an AI solution.

IBM watsonx.ai helps AI engineers overcome these challenges in a few ways, starting with the foundation model (FM) library that is provided by IBM. This library includes a diverse selection of AI models, such as the following ones:

- ▶ IBM trained models (the Granite and Slate model series)
- ▶ IBM selected open-source models through Hugging Face
- ▶ Third-party models such as llama and mixtral

Teams can choose among many different FMs for their use case, and choose a model that is best suited for their use case. Teams are not locked into one specific vendor or model series. Also, for more advanced AI engineers, they can upload and deploy their own FMs.

With watsonx.ai, there are different ways for AI engineers to work with models. The Prompt Lab tool in watsonx.ai enables AI engineers to write effective prompts (by using a GUI) to deploy to FMs for inferencing. For individuals who have more programming experience and technical expertise, there is also the programmatic alternative to the Prompt Lab, where users can prompt FMs by using the Python library or REST API.

You can use the tuning studio in watsonx.ai to tune a smaller FM to improve its performance. AI engineers can tune a smaller FM to achieve comparable results to larger models in the same model family, which can lead to reduced inference costs in the long term.

3.2 Low-code, no-code, and full-code tools

Because the demand for data scientists has grown over the years, finding experienced data scientists is difficult for many organizations. For organizations that are newer to data science, they find that experienced data scientists often request higher salaries or tend to seek opportunities with interesting and advanced problems to solve.

Therefore, it is important for organizations to find ML and AI platforms that support individuals with varying skill sets. There are many different types of tools for implementing data science, machine learning operations (MLOps), and generative AI (gen AI) solutions. At a high level, they can fall into one of three categories: no-code, low-code, or full-code.

- ▶ With no-code tools, users can create solutions and applications without writing code. The tool provides a GUI and includes “drag-and-drop” features so that users can build models with little to no programming knowledge.
- ▶ Low-code tools provide a visual approach to development so that users can generate solutions, such as models, with minimal hand-coding. Like no-code tools, these tools typically provide a GUI and include “drag-and-drop” features. Low-code tools enable users with minimal coding experience, such as citizen data scientists or business analysts to quickly build and implement a solution.
- ▶ With full-code tools, users can write their own code to develop solutions. These tools are typically leveraged by experienced data scientists. Full-code tools enable greater flexibility and more customization, but require deeper programming knowledge and expertise.

3.2.1 No-code, low-code, and full-code tools on IBM watsonx.ai

When to use a no-code, low-code, or full-code tool varies by project requirements, team skill set, and the time and cost that a specific solution has for an organization. IBM watsonx.ai provides data scientists, ML engineers, and AI engineers with the flexibility of choose the right tool for a use case by providing various no-code, low-code, and full-code tools as part of the overall platform.

No-code solutions on IBM watsonx.ai

This section highlights a few of the no-code tools that are available on the IBM watsonx.ai platform.

AutoAI

AutoAI is a no-code tool that data scientists can use to develop and prototype models quickly, without requiring the user to code or have programming knowledge. AutoAI helps data scientists and data science teams compare the results of multiple models efficiently, thus providing teams with the opportunity to save time and money.

Figure 3-1 on page 21 shows an AutoAI experiment on watsonx.ai.

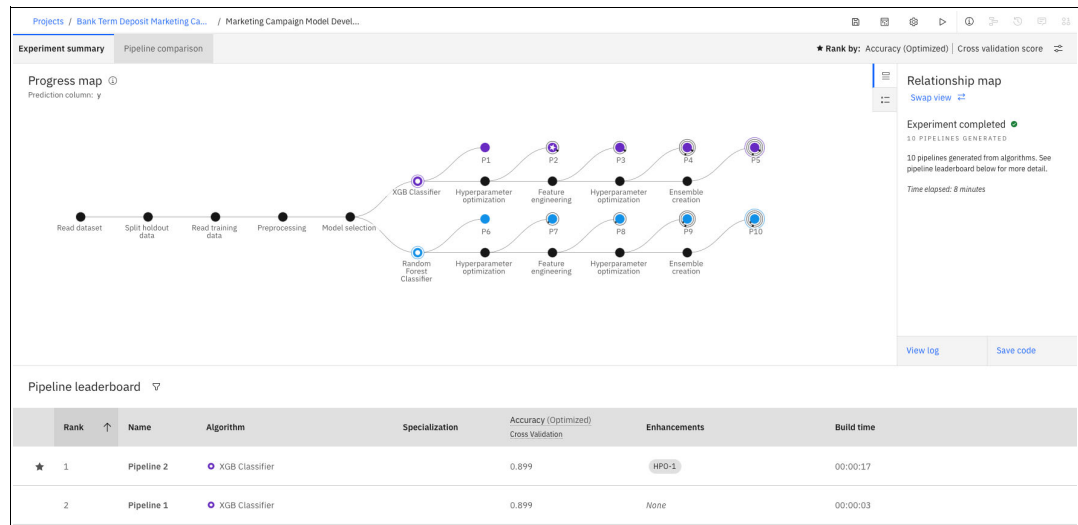


Figure 3-1 AutoAI experiment on IBM watsonx.ai

Although AutoAI is a no-code tool, the ML pipelines that are generated by an AutoAI experiment can be exported as a notebook. Therefore, more experienced data scientists can view the code “under the hood” and make modifications and updates to the underlying code.

Figure 3-2 shows an example notebook that was created from a pipeline that was generated from an AutoAI experiment.

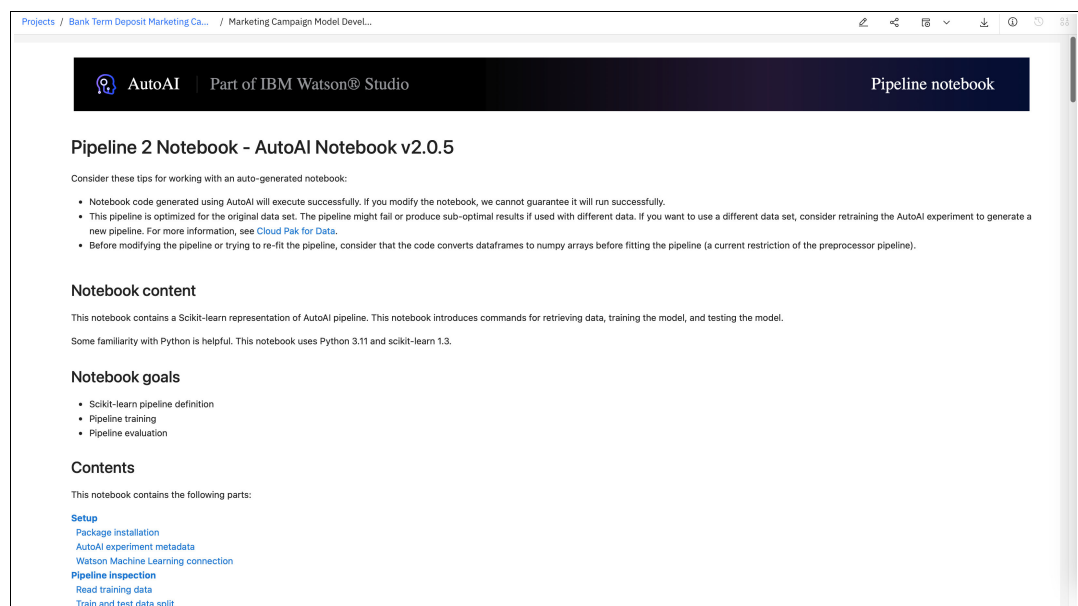


Figure 3-2 Pipeline notebook that was generated from an AutoAI experiment

AutoAI for Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is an AI framework for improving the quality of large language model (LLM)-generated responses by grounding the model in external sources of knowledge to supplement the LLM’s internal representation of information. The key benefits of RAG solutions include reducing the chance that an LLM leaks sensitive data or ‘hallucinate’ incorrect or misleading information, and reducing the need for users to continuously train the model on new data, thus resulting in lower computational and financial costs. For more information about RAG, see [What is Retrieval-Augmented Generation?](#)

Many organizations are implementing RAG solutions as part of their gen AI efforts. Although RAG has many benefits, there are challenges too, such as creating a robust and scalable pipeline, and the time to deliver and implement RAG solutions.

AutoAI for RAG is a tool that was created by IBM. Similar to AutoAI, AutoAI for RAG is intended to help AI engineers quickly build RAG solutions. With AutoAI for RAG, AI engineers can quickly build and test multiple RAG pipelines without writing code. From the pipelines that are generated, the AI engineer can assess the performance of each pipeline, select the best pipeline for their team and project, and deploy it into a production or non-production environment.

Figure 3-3 shows an AutoAI for RAG experiment.

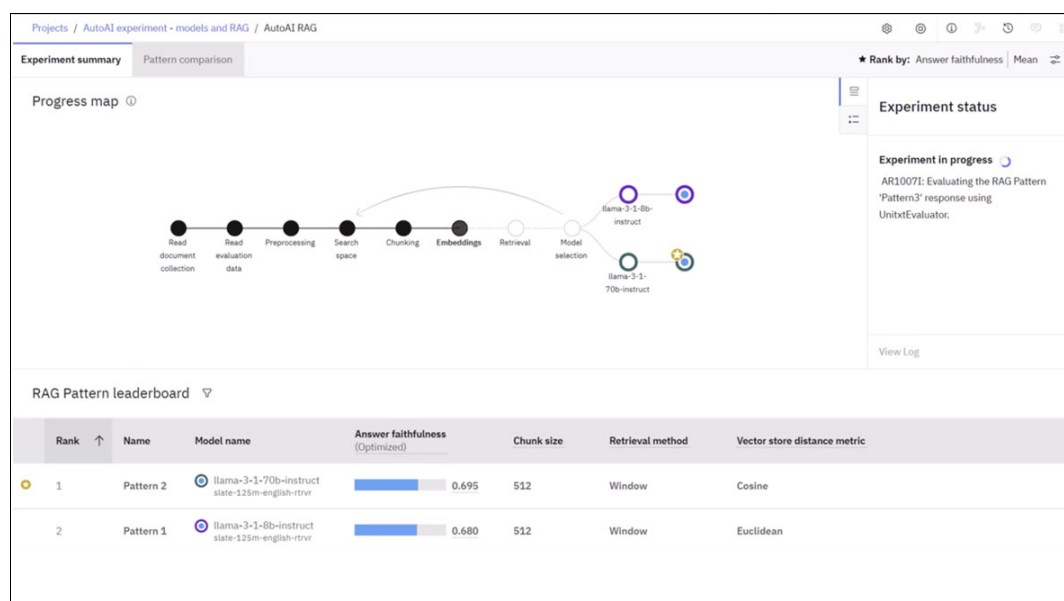


Figure 3-3 AutoAI for RAG experiment on IBM watsonx.ai

For more information about AutoAI for RAG, view the documentation and demo video at [Creating a RAG experiment \(fast path\) \(Beta\)](#).

Synthetic Data Generator

The Synthetic Data Generator is a no-code tool that you can use to generate tabular data for model training. Users have two options to generate synthetic data by using the graphical flow editor in the Synthetic Data Generator tool:

- ▶ Generate synthetic tabular data based on production data, with the goal of masking and mimicking this data.
- ▶ Generate synthetic data from a custom data schema that is defined by the user by using visual flows and modeling algorithms.

Figure 3-4 shows a view of the Synthetic Data Generator interface on watsonx.ai.

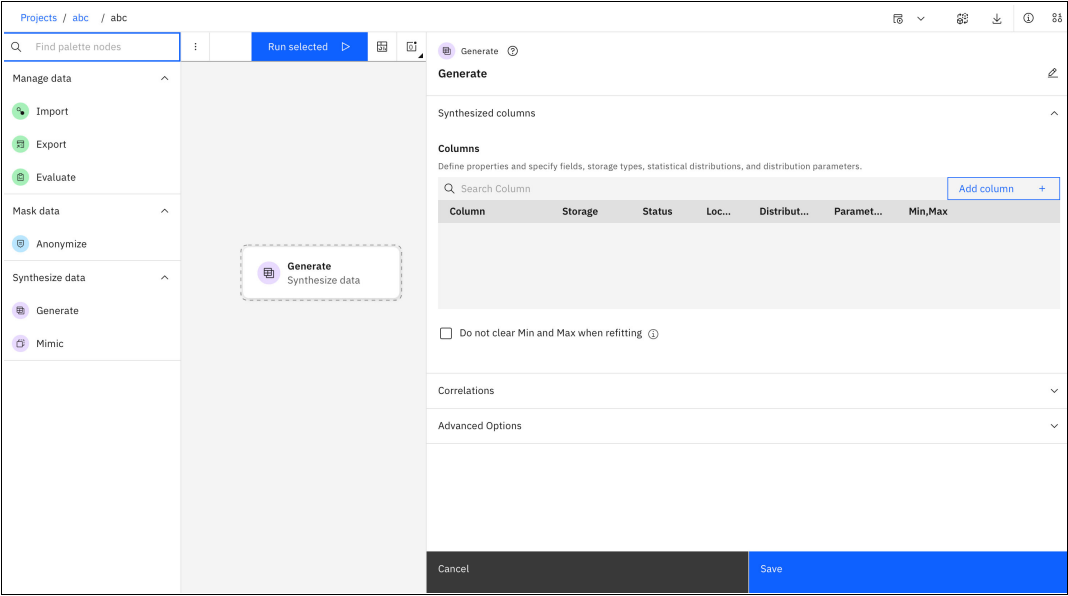


Figure 3-4 Synthetic Data Generator on IBM watsonx.ai

Data Refinery

Data Refinery is a no-code tool that you can use to prepare and visualize data without writing any code. With this tool, you can prepare the data for analysis by applying operations such as filters and aggregations, and you can generate visualizations such as pie charts and bar charts to extract insights and share findings with stakeholders.

Figure 3-5 shows an example of a Data Refinery flow on watsonx.ai.

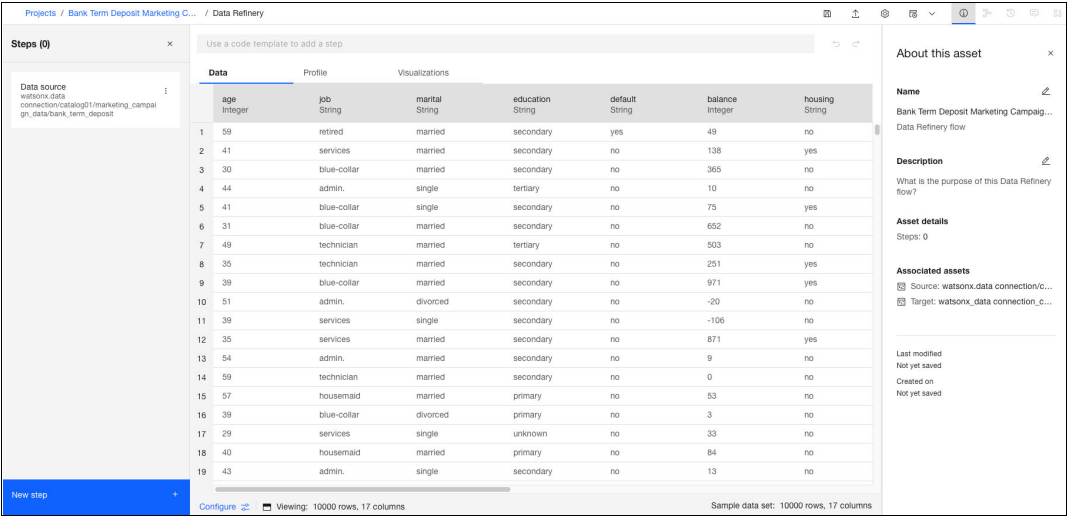


Figure 3-5 Data Refinery flow on IBM watsonx.ai

Prompt Lab

Prompt Lab is a tool in IBM watsonx.ai that you can use to experiment with prompting different FMs, and create and share effective prompts to submit to deployed FMs for inferencing. Prompt Lab is a no-code tool that provides AI engineers with three different edit modes for prompt editing: Chat, Structured, and Freeform. This flexibility enables novice and experienced AI engineers to get the most out of Prompt Lab and watsonx.ai. The Chat mode enables users to converse with a FM of their choice. The Structured mode is great for novice users because it helps them create effective prompts by providing defined fields, and also by providing sample templates to build on. The Freeform mode is great for more experienced AI engineers who know how to format a prompt; with this option, users submit prompts in plain text.

Figure 3-6 shows the Chat mode in Prompt Lab on watsonx.ai.

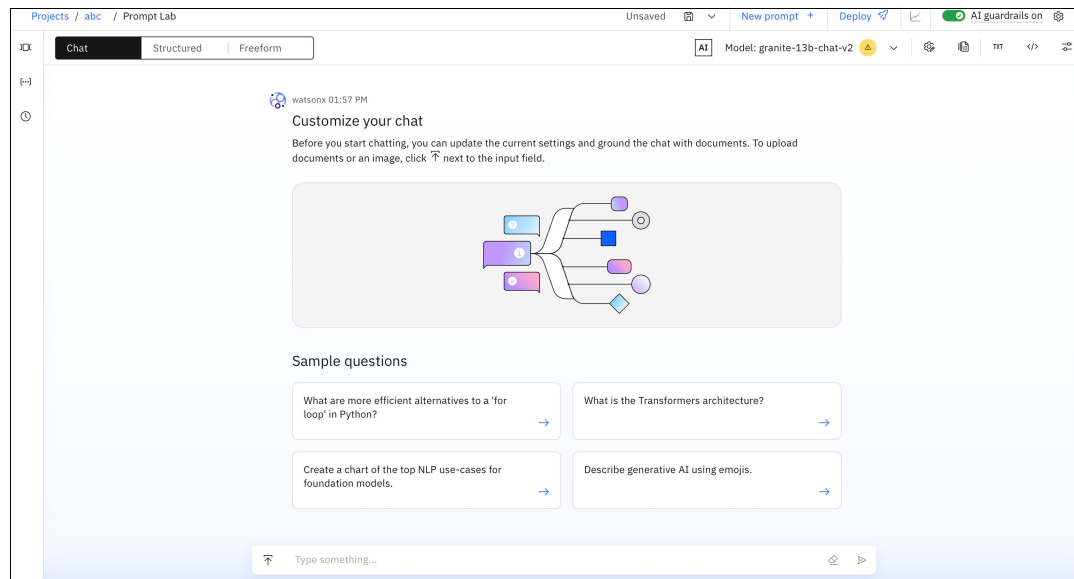


Figure 3-6 Prompt Lab Chat on IBM watsonx.ai

Low-code solutions on IBM watsonx.ai

This section highlights a few of the low-code tools that are available on the IBM watsonx.ai platform.

SPSS Modeler

SPSS Modeler is a low-code tool that you can use to transform data and build models with little to no-programming experience. You can drag-and-drop various nodes onto the canvas to create a flow to perform various tasks, such as importing data, merging data, and creating models.

Figure 3-7 on page 25 shows an example SPSS Modeler flow in watsonx.ai.

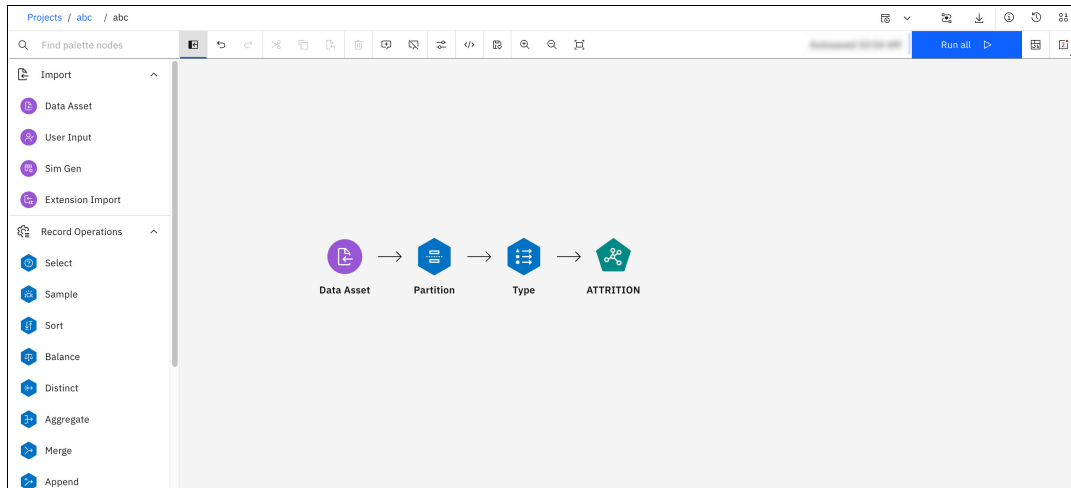


Figure 3-7 SPSS Modeler flow on IBM watsonx.ai

Full-code solutions on IBM watsonx.ai

This section highlights a few of the full-code tools that are available on the IBM watsonx.ai platform.

Jupyter Notebook editor

The Jupyter Notebook editor is a full-code tool that is available on IBM watsonx.ai. Jupyter Notebooks enable more experienced data scientists and developers to write and run code directly on the IBM watsonx.ai platform. Teams can build more customized and flexible solutions.

Figure 3-8 shows an example of a Jupyter Notebook on the watsonx.ai platform.

The image shows a Jupyter Notebook interface on the IBM watsonx.ai platform. The notebook is titled 'Marketing Campaign Analysis' and is in the 'Import Data' section. The code cell contains Python code that uses the 'itcfs' library to interact with the 'watsonx.data' connection, retrieve flight information, and load it into a pandas DataFrame. The code includes comments about row limits and display settings. Below the code, a preview of the data is shown as a table with columns: age, job, marital, education, default, balance, housing, loan, contact, day, month, duration, campaign, pdays, previous, poutcome, and y. The table displays the first 10 rows of data.

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	49	management	married	tertiary	no	4374	yes	no	cellular	31	jul	68	3	-1	0	unknown	no
1	50	blue-collar	married	secondary	no	9585	yes	no	telephone	31	jul	182	6	-1	0	unknown	no
2	40	admin.	married	secondary	no	341	no	no	cellular	31	jul	1142	5	-1	0	unknown	yes
3	36	management	married	tertiary	no	1594	yes	no	cellular	31	jul	544	5	-1	0	unknown	no
4	50	entrepreneur	married	tertiary	no	52	no	no	cellular	31	jul	22	8	-1	0	unknown	no
5	60	retired	married	secondary	no	0	no	no	telephone	31	jul	576	20	-1	0	unknown	no

Figure 3-8 Jupyter Notebook on IBM watsonx.ai

RStudio

RStudio, like the Jupyter Notebook editor, is a full-code tool that is available on IBM watsonx.ai. RStudio enables individuals with programming experience in R to visualize data, create models, and build solutions by using the R programming language on the IBM watsonx.ai platform.

Figure 3-9 shows the RStudio interface on the watsonx.ai platform.

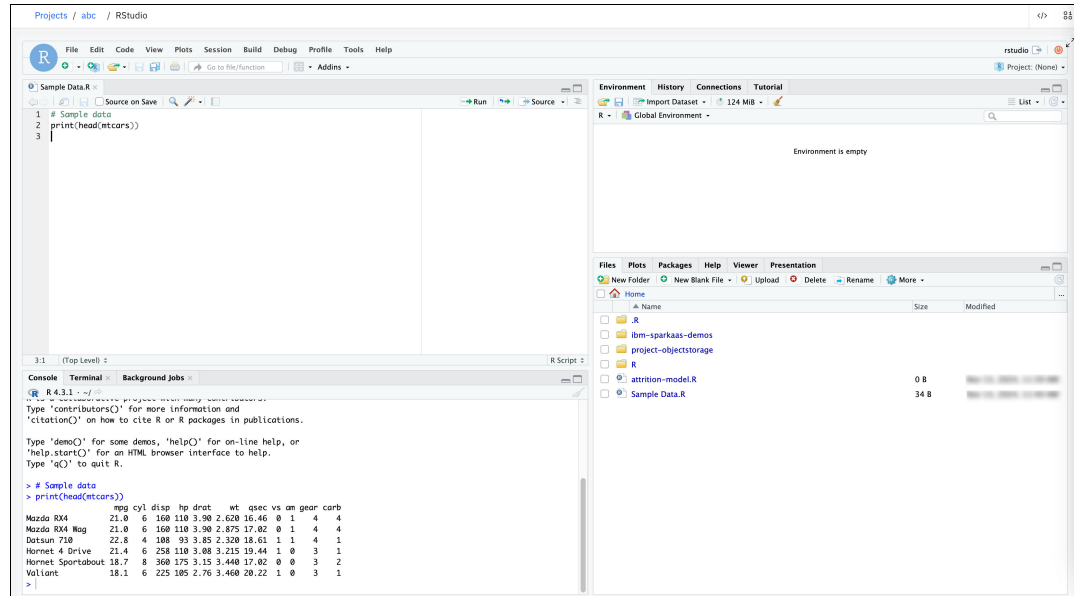


Figure 3-9 RStudio on IBM watsonx.ai

Programmatic alternative to Prompt Lab

IBM watsonx.ai has a Python library and REST API that you can use to prompt FMs. This approach is an alternative to the GUI in the Prompt Lab that is used to prompt FMs. This option is great for users who have more programming or technical experience, and for projects and teams that might require an alternative to the Prompt Lab.

For more information about using the REST API or Python library to prompt FMs, see [Coding generative AI solutions](#).



Building and using artificial intelligence models

This chapter serves as a resource for setting up, building, fine-tuning, and deploying artificial intelligence (AI) models within the IBM watsonx.ai ecosystem. By exploring key features, and best practices, it aims to empower users (beginners or experienced practitioners) to harness the power of AI for developing effective business solutions and enhancing their AI initiatives.

The following topics are described in this chapter:

- ▶ 4.1, “Prerequisites and assumptions” on page 28
- ▶ 4.2, “How to use this chapter” on page 28
- ▶ 4.3, “Building and using AI models in watsonx.ai” on page 28
- ▶ 4.4, “Getting started with watsonx.ai: Setting up the environment” on page 29
- ▶ 4.5, “Data preparation and ingestion for AI model building” on page 34
- ▶ 4.6, “Building AI models in watsonx.ai” on page 38
- ▶ 4.7, “Deploying AI models in watsonx.ai” on page 40
- ▶ 4.8, “watsonx.ai LLM deployment” on page 45
- ▶ 4.9, “Operationalizing machine learning and LLM models” on page 50
- ▶ 4.10, “Additional information and where to go next” on page 54

4.1 Prerequisites and assumptions

In this chapter, it is assumed that you have met the following prerequisites:

- ▶ An active IBM Cloud® account: You have an active IBM Cloud account. If you do not have one, see watsonx.ai.
- ▶ Familiarity with Jupyter Notebooks: You understand how to navigate and work with Jupyter Notebooks for data exploration and model development.
- ▶ Proficiency in Python: You have basic to intermediate knowledge of Python programming because many examples, scripts, and workflows in this chapter use Python code.
- ▶ Cloud computing concepts: You have a basic understanding of cloud computing principles, including concepts such as APIs, data storage, computing resources, and cloud-based environments.
- ▶ Knowledge of large language models (LLMs): A general understanding of LLMs, their capabilities, and use cases is beneficial for building and fine-tuning AI models within watsonx.ai.
- ▶ You read Chapters 1 - 3 of this book.

4.2 How to use this chapter

This chapter is structured to help users of varying expertise levels by providing a general approach to understanding, building, deploying, and optimizing AI models by using the watsonx.ai platform.

For beginners, start with the overview sections to familiarize yourself with the platform's features and capabilities. Progress through the chapter sequentially, beginning with environment setup and basic concepts before delving into more complex topics, such as model building and optimization.

For experienced users, you can go directly to specific chapters of interest, such as model optimization techniques, deployment strategies, or advanced configurations. Each section is modular and provides targeted information and best practices that you can apply immediately to your projects.

Throughout the chapter, you find practical examples, hands-on exercises, and links to more resources to help ensure a well-rounded learning experience.

4.3 Building and using AI models in watsonx.ai

This section covers the following topics:

- ▶ Overview of the watsonx.ai platform
- ▶ Key features and capabilities

4.3.1 Overview of the watsonx.ai platform

watsonx.ai is an enterprise-grade AI studio that you use to streamline the development, training, tuning, and deployment of AI models. It includes generative AI (gen AI) capabilities that are powered by foundation models (FMs).

4.3.2 Key features and capabilities

Here are the key features and capabilities of watsonx.ai:

- ▶ **Foundation models:** Access various powerful, low-cost, and fit-for-purpose models, such as the IBM Granite series and other LLMs for tasks such as content generation, summarization, and classification.
- ▶ **Data preparation:** Use tools for refining and visualizing data to help ensure high-quality inputs for model training.
- ▶ **Model development:** Build machine learning (ML) models by using open-source frameworks with options for code-based, automated, or visual data science approaches.
- ▶ **Prompt Lab:** Experiment with gen AI prompts to enable tasks like question answering, content generation, summarization, text classification, and data extraction.
- ▶ **Tuning Studio:** Fine-tune FMs to customize outputs for specific use cases to enhance model performance and accuracy.
- ▶ **InstructLab:** At the time of writing, this feature is planned to be integrated into watsonx.ai in the future.

4.4 Getting started with watsonx.ai: Setting up the environment

To set up the watsonx.ai environment, you must have an IBM Cloud account. To register for an account, see [Create an IBM Cloud account](#).

At the time of writing, here are the high-level steps to provision watsonx.ai in a Software-as-a-Service (SaaS) environment:

1. Set up your IBM Cloud account.
2. Create a Project in watsonx:
 - a. Log in to <https://dataplatform.cloud.ibm.com/login>
 - b. Expand the ‘hamburger’ navigation menu, as shown in Figure 4-1.

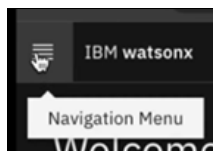


Figure 4-1 watsonx navigation menu icon

- c. Select **Projects** → **View all projects**, as shown in Figure 4-2.

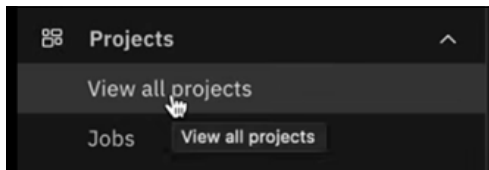


Figure 4-2 View all projects menu

- d. Click **New project +**, as shown in Figure 4-3.

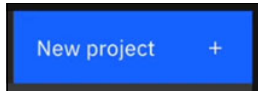


Figure 4-3 New project+ menu

- e. Enter the project name and, if applicable, upload local files. Click **Create**, as shown in Figure 4-4.



Figure 4-4 Create menu

3. Provision watsonx.ai Studio:

- a. Log in to <https://cloud.ibm.com/>.
b. To add services from the catalog, use the search box that is shown in Figure 4-5.

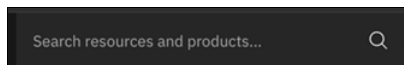


Figure 4-5 watsonx.ai utilities search box

- c. Enter “watsonx.ai Studio” and choose the studio from the catalog, as shown in Figure 4-6.

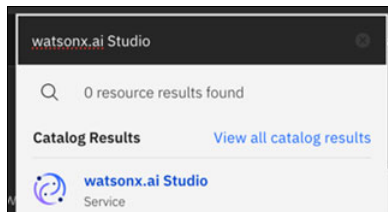


Figure 4-6 watsonx.ai utilities list

- d. Select a pricing plan (**Lite** or **Professional**), and then click **Create**.

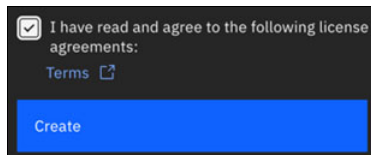


Figure 4-7 watsonx.ai pricing plan selection

4. Generate a watsonx.ai API key:

- a. Go to <https://cloud.ibm.com/iam/apikeys>.
b. Click **Create**, as shown in Figure 4-8.

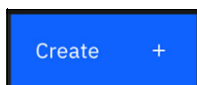


Figure 4-8 Create API key menu option

- c. Enter the relevant information, and then click Create, as shown in Figure 4-9.

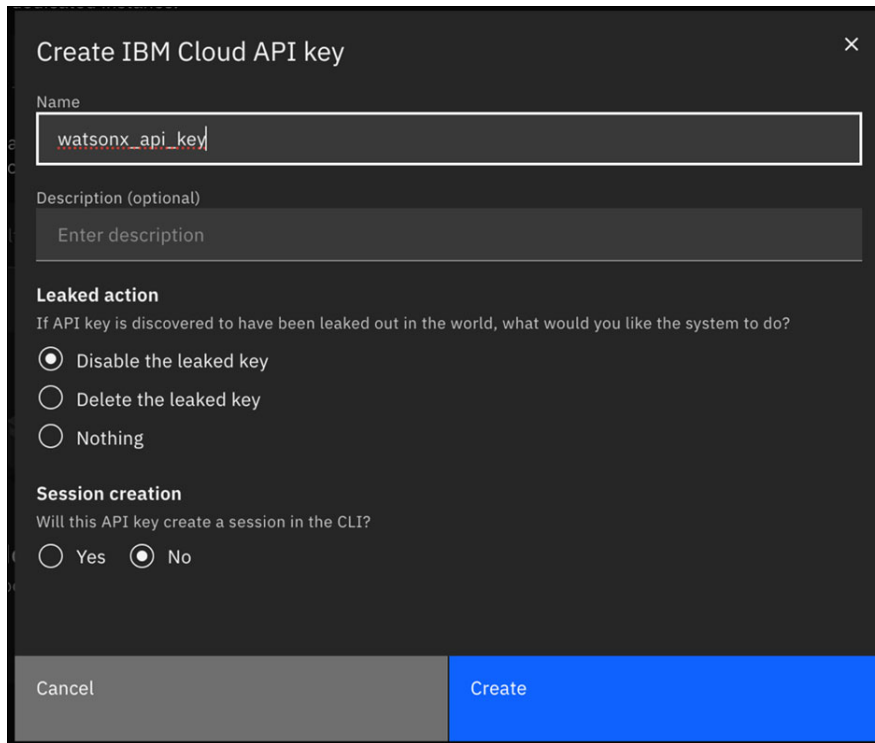
A dark-themed dialog box titled "Create IBM Cloud API key" with a close button (X) in the top right corner. It contains a "Name" field with the text "watsonx_api_key", an optional "Description" field with the placeholder "Enter description", a "Leaked action" section with three radio button options: "Disable the leaked key" (selected), "Delete the leaked key", and "Nothing", and a "Session creation" section with two radio button options: "Yes" and "No" (selected). At the bottom are "Cancel" and "Create" buttons.

Figure 4-9 Create IBM Cloud API key

- d. After the API key is successfully created, copy or download the key and save it locally, as shown in Figure 4-10

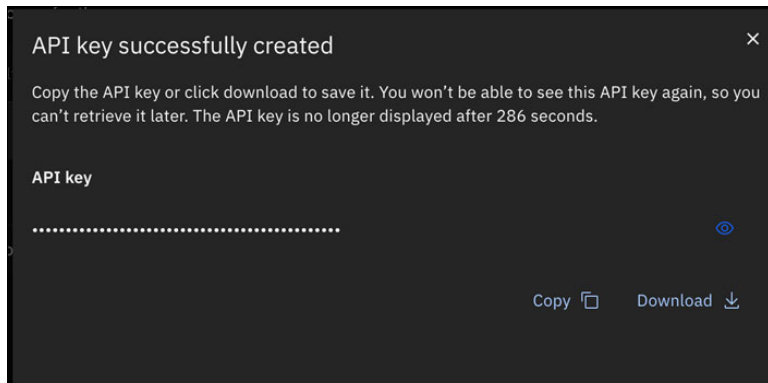
A dark-themed notification box titled "API key successfully created" with a close button (X) in the top right corner. It contains a warning message: "Copy the API key or click download to save it. You won't be able to see this API key again, so you can't retrieve it later. The API key is no longer displayed after 286 seconds." Below this is a section labeled "API key" showing a masked key "....." with an eye icon to its right. At the bottom are "Copy" and "Download" buttons with their respective icons.

Figure 4-10 API key creation

5. Open watsonx.ai Studio and associate the watsonx.ai service:
- a. In the upper left of your screen, click the four horizontal lines, as shown in Figure 4-11



Figure 4-11 watsonx.ai studio menu icon

- b. Select **Resource list**, as shown in Figure 4-12.

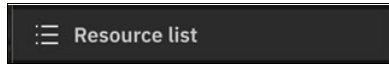


Figure 4-12 *watsonx.ai studio Resource list option*

- c. Expand **AI / Machine Learning**, as shown in Figure 4-13.

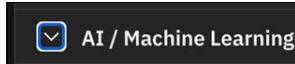


Figure 4-13 *AI / Machine Learning menu option*

- d. Click the **watsonx.ai Studio** record, as shown in Figure 4-14.

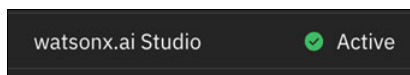


Figure 4-14 *watsonx.ai Studio record icon*

- e. Click **View full details**, as shown in Figure 4-15.

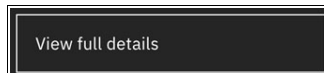


Figure 4-15 *watsonx.ai studio details*

- f. Select **Launch in** → **IBM watsonx**, as shown in Figure 4-16.

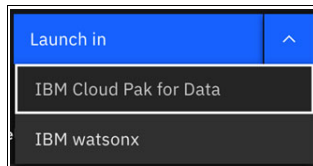


Figure 4-16 *watsonx.ai Launch in option*

Figure 4-17 shows the Welcome to watsonx window.

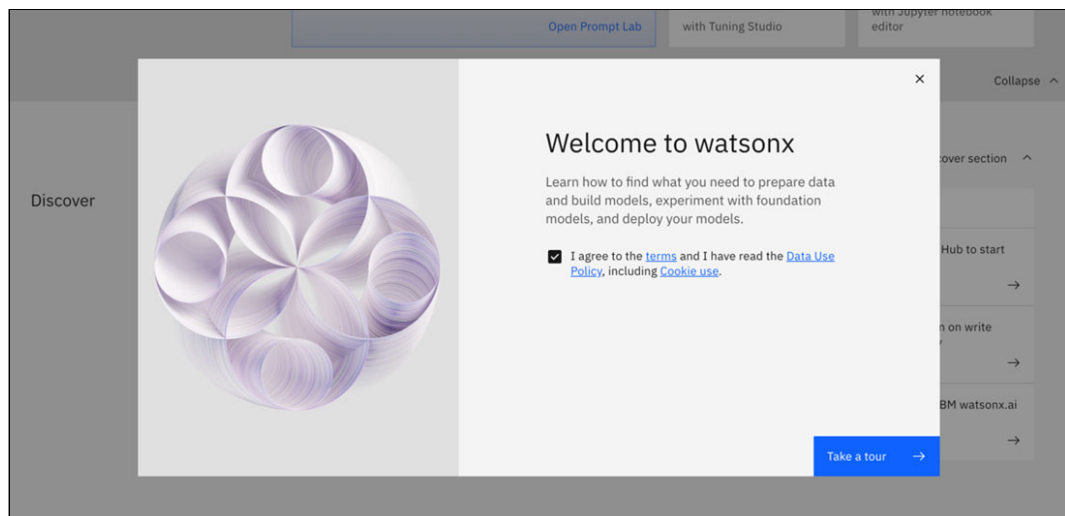


Figure 4-17 *Welcome to watsonx window*

- g. Click the + in the Projects table.
- h. Enter a project name, for example, test_project, as shown in Figure 4-18.

Define details

Name

test_project

Description (optional)

What's the purpose of this project?

Figure 4-18 Define details window

- i. Select **Storage** (if required).
- j. Click **Create**.
- k. Select **Chat and build prompts with foundations models**.
- l. Click **Associate service** to associate a Watson ML service to the project, as shown in Figure 4-19.

No Watson Machine Learning service detected

To access the Prompt Lab, you must associate a Watson Machine Learning service instance to this project.

Associate service →

Figure 4-19 Clicking Associate service

- m. Select the displayed **Machine Learning** service and click **Associate**, as shown in Figure 4-20.

Associate service

Choose an existing or add a new service to associate with your project.

Resource Groups 2 Locations

Find services New service +

Name	Type	Plan	Location	Status	Group
<input checked="" type="checkbox"/> Machine Learning-fz ⓘ	Watson Machine Learning	v2 Standard	Dallas	Not associated ⓘ	default

CancelAssociate

Figure 4-20 watsonx machine learning: Associate service

n. Go to the **Assets** tab, and then click **New asset**, as shown in Figure 4-21.

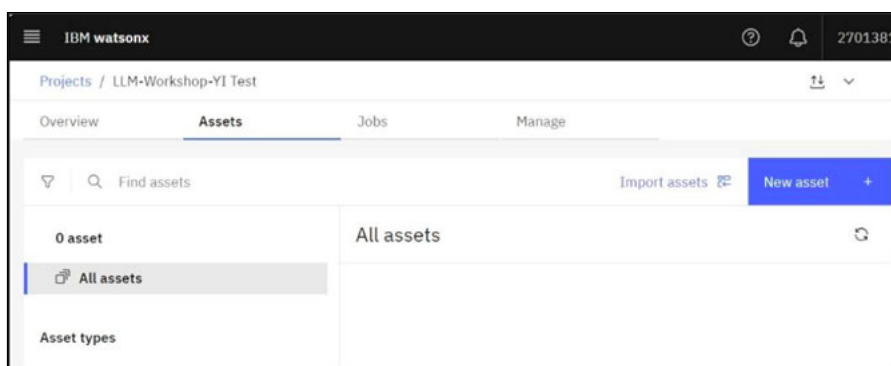


Figure 4-21 watsonx projects: All assets

o. Select **Chat and build prompts with foundation models**, as shown in Figure 4-22.



Figure 4-22 Chat and build prompts with foundation models tile

4.5 Data preparation and ingestion for AI model building

This section describes the following topics:

- ▶ Understanding the importance of data in AI
- ▶ Preparing and cleaning data: data quality considerations
- ▶ Handling missing data, outliers, and bias
- ▶ Ingesting data into watsonx.ai Studio
- ▶ Connecting to data repositories and cloud services

4.5.1 Understanding the importance of data in AI

Data is the backbone of AI. It shapes the accuracy, effectiveness, and reliability of AI models. High-quality data enables AI to learn patterns, make predictions, and deliver meaningful insights. Understanding the importance of data in AI goes beyond mere volume; it involves ensuring data accuracy, consistency, and fairness. Poor data quality, bias, or gaps can lead to flawed models, incorrect predictions, and potential ethical issues. Therefore, this section focuses on the robust data preparation, cleaning, and validation that is essential for building AI systems that are trustworthy, transparent, and impactful in real-world applications.

4.5.2 Preparing and cleaning data: data quality considerations

Data quality is a critical factor in the success of AI models because it directly influences their performance and accuracy. Here are some key considerations:

- ▶ **Accuracy:** Ensuring that data is correct, consistent, and error-free is vital for creating reliable AI models. Inaccurate data can lead to faulty predictions and flawed decision-making.
- ▶ **Completeness:** AI models rely on comprehensive datasets to learn effectively. Missing data can skew results, which cause incomplete or biased predictions.
- ▶ **Consistency:** Data must be consistent across different sources and formats to help ensure the AI model functions as expected.
- ▶ **Relevance:** Data should be pertinent to the problem the AI aims to solve. Irrelevant data might introduce noise and reduce model effectiveness.
- ▶ **Bias and fairness:** Addressing data biases and ensuring fairness are critical for building ethical and unbiased AI systems. Give careful attention to the diversity and representation in data to avoid discrimination and ensure balanced outcomes.

High data quality maximizes the accuracy, transparency, and applicability of AI systems, which ultimately enhances their value and impact.

4.5.3 Handling missing data, outliers, and bias

Handling missing data, outliers, and bias is crucial for ensuring the accuracy and fairness of AI models. Here is how each one is managed:

- ▶ **Handling missing data:**
 - **Imputation techniques:** Missing data can be filled by using statistical techniques such as mean, median, or mode imputation, or more complex methods like k-nearest neighbors (KNNs) or predictive models.
 - **Data removal:** Sometimes, records with significant missing values are removed if they are unlikely to add useful information.
 - **Domain knowledge:** Input from domain experts can help determine whether missing data should be treated differently based on context.
- ▶ **Handling outliers:**
 - **Detection:** Outliers are identified by using methods like Z-scores, Interquartile Range (IQR), or visualization techniques like boxplots.
 - **Treatment:** Once detected, outliers can be managed by removal, transformation (for example, log transformation), or capping values based on acceptable thresholds.
 - **Contextual consideration:** Not all outliers are problematic; they might represent genuine data points. Understanding their impact is crucial before deciding on a course of action.
- ▶ **Addressing bias:**
 - **Data auditing:** Systematic review of datasets to identify sources of bias, such as underrepresentation of specific groups.
 - **Data balancing:** Techniques like oversampling, undersampling, or generating synthetic data (for example, SMOTE) can help balance datasets.

- Algorithmic bias mitigation: Algorithms can be fine-tuned by using fairness constraints or reweighting schemes to minimize bias during model training.
- Regular monitoring: Bias can emerge or change over time, which requires continuous assessment and model updates to maintain fairness and inclusivity.

By handling missing data, outliers, and bias, AI models can provide more accurate, reliable, and ethical outcomes, which ultimately increase their utility and impact across diverse applications.

4.5.4 Ingesting data into watsonx.ai Studio

This section describes the following topics:

- ▶ Supported data formats and sources
- ▶ Manually uploading data to watsonx.ai Studio

Supported data formats and sources

watsonx.ai Studio supports seamless ingestion of diverse data formats and sources to facilitate efficient model development and deployment. Supported formats include structured data (CSV, JSON, and Parquet), semi-structured data (XML and Avro), and unstructured data (plain text and PDF). Data can be sourced from cloud storage platforms (such as AWS S3 and IBM Cloud Object Storage), databases (such as PostgreSQL and MySQL), APIs, and on-premises file systems.

The ingestion process is optimized for scalability and can handle large datasets while ensuring data integrity and compatibility with downstream AI workflows. Integration with watsonx.data and watsonx.governance tools enables a secure, governed data pipeline, which enhances traceability and compliance.

Manually uploading data to watsonx.ai Studio

You can upload files through the watsonx.ai Studio interface by dragging the files there, as shown in Figure 4-23.

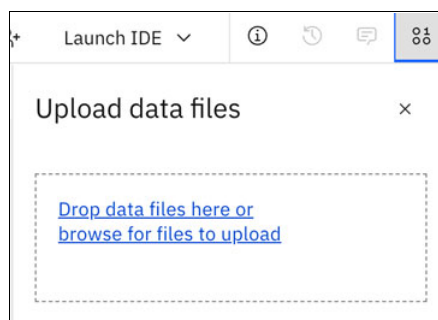


Figure 4-23 watsonx Studio data upload window

4.5.5 Connecting to data repositories and cloud services

watsonx.ai provides robust connectivity options to integrate with various data repositories and cloud services, which help ensure smooth data access for AI model development. Users can connect to cloud storage solutions such as IBM Cloud Object Storage, AWS S3, Azure Blob Storage, and Google Cloud Storage, and traditional databases like PostgreSQL, MySQL, and MongoDB.

Figure 4-24 on page 37 shows the watsonx cloud services connections window.

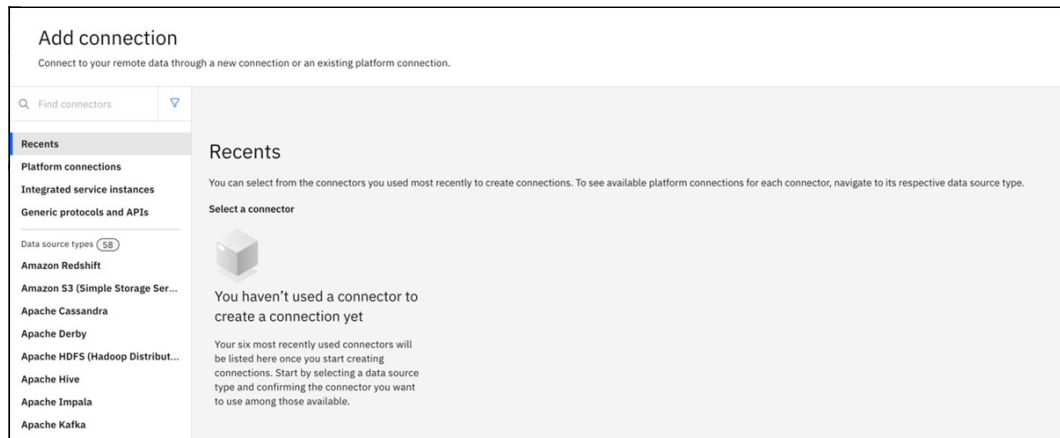


Figure 4-24 watsonx cloud services connections

At the time of publication, here are the repositories and services that are supported:

- ▶ Amazon Redshift
- ▶ Amazon S3
- ▶ Apache Cassandra
- ▶ Apache Derby
- ▶ Apache HDFS
- ▶ Apache Hive
- ▶ Apache Impala
- ▶ Apache Kafka
- ▶ Box
- ▶ DataStax Enterprise
- ▶ Denodo
- ▶ Dremio
- ▶ Dropbox
- ▶ Elasticsearch
- ▶ Google BigQuery
- ▶ Google Cloud Pub/Sub
- ▶ Google Cloud Storage
- ▶ Google Locker
- ▶ Greenplum Database
- ▶ IBM Cloud Data Engine
- ▶ IBM Cloud Object Storage
- ▶ IBM Cloudant®
- ▶ IBM Cognos® Analytics
- ▶ IBM Data Virtualization Manager for IBM z/OS®
- ▶ IBM DataStage® for Cloud Pak for Data
- ▶ IBM Db2®
- ▶ IBM Informix®
- ▶ IBM InfoSphere® DataStage
- ▶ IBM Match 360
- ▶ IBM MQ
- ▶ IBM Netezza® Performance Server
- ▶ IBM Planning Analytics
- ▶ IBM watsonx.data
- ▶ MariaDB
- ▶ Microsoft Azure Blob Storage
- ▶ Microsoft Azure Cosmos DB
- ▶ Microsoft Azure Data Lake Storage

- ▶ Microsoft Azure Databricks
- ▶ Microsoft Azure Files
- ▶ Microsoft Azure SQL Database
- ▶ Microsoft Azure Synapse Analytics
- ▶ Microsoft Power BI
- ▶ Microsoft SQL Server
- ▶ Milvus
- ▶ MongoDB
- ▶ MySQL
- ▶ Oracle Database
- ▶ PostgreSQL
- ▶ Presto
- ▶ Salesforce API
- ▶ SAP ASE
- ▶ SAP IQ
- ▶ SAP S/4HANA
- ▶ SingleStoreDB
- ▶ Snowflake
- ▶ Tableau
- ▶ Teradata database
- ▶ Vertica

watsonx.ai supports secure connection protocols, which include API-based integrations, JDBC/ODBC drivers, and file transfer mechanisms like SFTP. With built-in authentication and access controls, watsonx.ai helps ensure data security while maintaining flexibility for enterprise-scale data workflows. By streamlining access to data repositories and services, the platform empowers teams to leverage their existing data infrastructure efficiently.

4.6 Building AI models in watsonx.ai

This section describes the following topics:

- ▶ Choosing the right model for your use case
- ▶ Model creation workflow

4.6.1 Choosing the right model for your use case

watsonx.ai supports many model types to meet diverse business needs:

- ▶ *Supervised models* for predictive tasks by using labeled data.
- ▶ *Unsupervised models* for discovering patterns in unlabeled data.
- ▶ *Reinforcement learning* (RL) models for decision-making through rewards and penalties.
- ▶ *Large language models* (LLMs) for natural language understanding and generation.
Pretrained LLMs streamline AI development, which enables faster implementation and powerful insights across applications.

In addition to its native capabilities, watsonx.ai embraces flexibility with the IBM Bring Your Own Model (BYOM) feature, which enables users to integrate and fine-tune their own LLMs within the platform for customized solutions. Furthermore, watsonx.ai supports integration with Hugging Face, which enables access to a vast library of pretrained models and tools. This collaboration accelerates development by leveraging open-source innovations while maintaining watsonx.ai enterprise-grade security and scalability.

4.6.2 Model creation workflow

This section describes the workflow of model creation.

Model selection and configuration

Selecting the appropriate AI model is key to achieving your business objectives, and watsonx.ai provides the flexibility to support both LLM and non-LLM models. Whether your needs involve predictive analytics, pattern discovery, decision-making, or natural language processing (NLP), watsonx.ai helps ensure that the correct tools are at your fingertips.

For non-LLM tasks, the platform accommodates models such as regression, classification, clustering, and RL, which enable a wide range of traditional ML applications.

When working with LLMs, watsonx.ai offers various pretrained models in different sizes, from lightweight options for tasks that require efficiency and speed to larger, more complex models that are ideal for nuanced language understanding and generation. Choosing the correct size depends on your specific use case, with smaller models excelling in cost-effective, lower-latency scenarios and larger models delivering superior accuracy and depth for intricate applications.

With watsonx.ai, you can confidently match the model type and size to your project's unique requirements to help ensure scalability, efficiency, and impact.

Training the model

Training your AI model is a crucial step in tailoring it to your specific business needs. watsonx.ai provides powerful tools and workflows for training both non-LLM models and LLM models, which help ensure flexibility and precision at every stage of development.

Non-LLM models

For traditional ML tasks, watsonx.ai offers robust training capabilities that leverage tools like watsonx.ai Studio and AutoAI to streamline and enhance the development process.

watsonx.ai Studio

watson.ai Studio provides a collaborative environment for data scientists, developers, and analysts to prepare data, build, and train ML models. With features like Jupyter Notebooks, Python libraries, and model monitoring, watson.ai Studio is designed for flexibility and scalability to accommodate projects of any complexity.

For more information about watsonx.ai Studio, see [IBM Watson Studio](#).

AutoAI

If you want to accelerate the development process, AutoAI automates key stages of ML, which include feature engineering, algorithm selection, and hyperparameter optimization. It simplifies the model-building process to make it accessible to users with varying technical expertise while still delivering highly accurate results.

For more information about AutoAI, see [IBM AutoAI](#).

LLM models

watsonx.ai provides advanced features for training and fine-tuning LLMs to deliver seamless customization and performance.

Prompt Lab

Prompt Lab is an environment for creating and testing prompts that are tailored to specific tasks. With Prompt Lab, users can interact with pretrained LLMs, evaluate their outputs, and refine instructions for optimal results, all without extensive coding expertise.

Within Prompt Lab, you can explore and train various training LLM training methodologies, such as Zero-Shot, Few-Shot, Multi-Shot, and Retrieval-Augmented Generation (RAG).

For more information about Prompt Lab, see [Prompt Lab](#).

Tuning Studio

For deeper customization, Tuning Studio enables fine-tuning of LLMs on your proprietary data, which helps ensure that the model adapts to your specific domain while maintaining high performance. This feature is ideal for organizations seeking more targeted insights and applications from their AI.

For more information about Tuning Studio, see [Tuning Studio](#).

InstructLab

At the time of writing, InstructLab is not available.

InstructLab will revolutionize the training process by enabling users to craft task-specific instructions, which further enhance the precision of LLMs in generating accurate and actionable outputs. This tool simplifies the process of aligning model behavior with unique business objectives.

With these comprehensive training tools, watsonx.ai empowers users to harness the full potential of their models, whether refining traditional ML algorithms or unleashing the power of cutting-edge LLMs.

4.7 Deploying AI models in watsonx.ai

This section explores the process of deploying AI models in the watsonx.ai platform. It provides a detailed overview of two major deployment options: Studio and Prompt Lab. From deploying models as APIs for real-time inference to batch processing workflows, the watsonx.ai platform helps ensure flexibility and scalability. Readers gain insights into selecting the most appropriate deployment strategy for their use case while also learning best practices to optimize performance and reliability in production environments.

4.7.1 watsonx.ai Studio deployments

Deploying models in watsonx.ai Studio involves several key steps to help ensure that your AI assets are effectively managed and operational. Here is a concise guide to help you through the process:

1. Create a deployment space: Begin by establishing a deployment space within the studio. This space serves as a collaborative environment where you can manage and deploy your AI assets. To set up a deployment space, go to the Deployment Space section in the Studio interface and follow the prompts to create a space (Figure 4-25 on page 41).

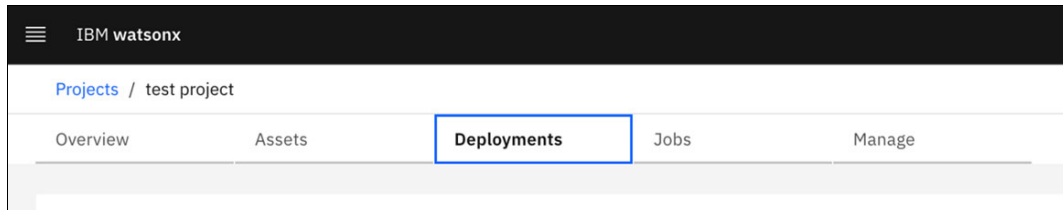


Figure 4-25 watsonx.ai studio projects: Deployments

2. Promote or import your model: When your deployment space is ready, add your trained model to it. If your model is in a project, promote it to the deployment space. Alternatively, you can import models that are trained externally by uploading them directly into the deployment space. Ensure that the model files are in a compatible format and that any necessary dependencies are addressed.
 - a. Click the three dots, and then click **Promote to space** (Figure 4-26).

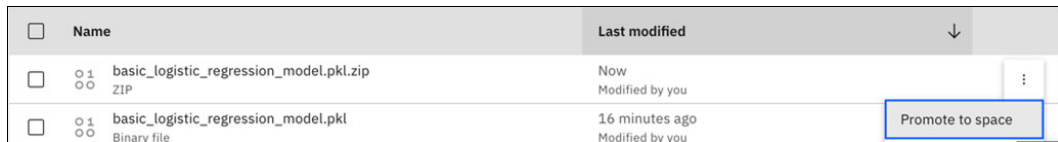


Figure 4-26 Model interface window

Note: To help ensure that your trained model is in the right format and compressed, see [Adding a model by using UI](#).

- b. Enter deployment_test in to the **Name** field, select **Production** under Deployment stage, and then click **Create** (Figure 4-27).

 The screenshot shows a 'Create a deployment space' dialog box. It has a title bar with a close button (X). Below the title, it says 'Use a space to collect assets in one place to create, run, and manage deployments'. The dialog is divided into two main sections: 'Define details' on the left and 'Select services' on the right.

 In the 'Define details' section:

- 'Name' field contains 'deployment_test'.
- 'Description (Optional)' field is empty.
- 'Deployment stage' dropdown is set to 'Production'.
- 'Deployment space tags (optional)' dropdown is set to 'Find or create tags'.
- 'Advanced Settings' section is collapsed.

 In the 'Select services' section:

- 'Select storage service' dropdown is set to 'itz-watson-apps-3lyaqj3a-cos'.
- 'Select watsonx.ai Runtime service (optional)' dropdown is set to 'Select a watsonx.ai Runtime service'.

 At the bottom right, there are two buttons: 'Cancel' and 'Create' (which is highlighted in blue).

Figure 4-27 Create a deployment space

3. Create the deployment: With your model in the deployment space, initiate the deployment process:
- a. Go to the Deployments window (Figure 4-28).

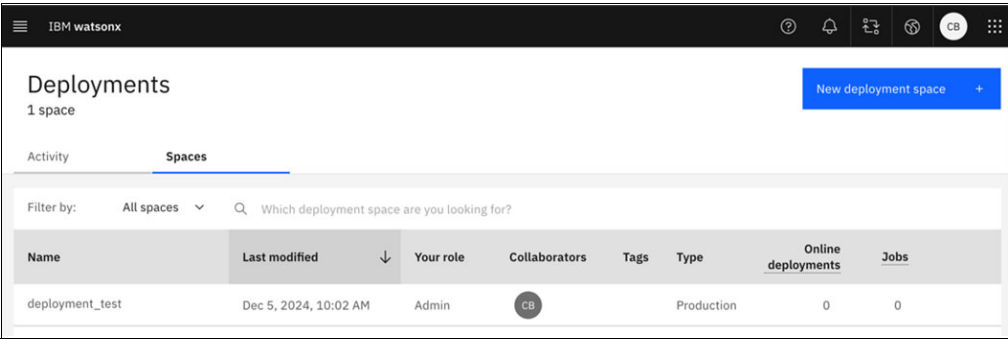


Figure 4-28 Deployments window

- a. Select deployment_test
- b. Click the generated model's service (Figure 4-29).


Name	Last modified	
 regression_model Machine learning model	5 minutes ago Service	⋮

Figure 4-29 Services

- c. Click **New deployment**, which opens the window that is shown in Figure 4-30 on page 43.

Create a deployment

Define details

Associated asset

regression_model

Deployment type

Online

Run the model on data in real-time, as data is received by a web service.

Batch

Run the model against data as a batch process.

Name

regression_model_deployment

Serving name ⓘ

regression_model_service

Description

Deployment description

Tags

Add tags to make assets easier to find.

Find or create tags

Cancel

Create

Figure 4-30 Create a deployment

- d. Under Deployment type, select either **Online** or **Batch**, and enter `regression_model_deployment` under Name and `regression_model_service` under Serving name. Click **Create**.

Online deployment is ideal for real-time processing, where the model handles input data and provides immediate predictions. Batch deployment is suitable for processing large datasets in bulk, which generate predictions for a collection of inputs at scheduled intervals or on-demand.

4. Test the deployment: After deployment, validate the model's functions by going to the newly created deployment and clicking `regression_model_deployment`, as shown in Figure 4-31.

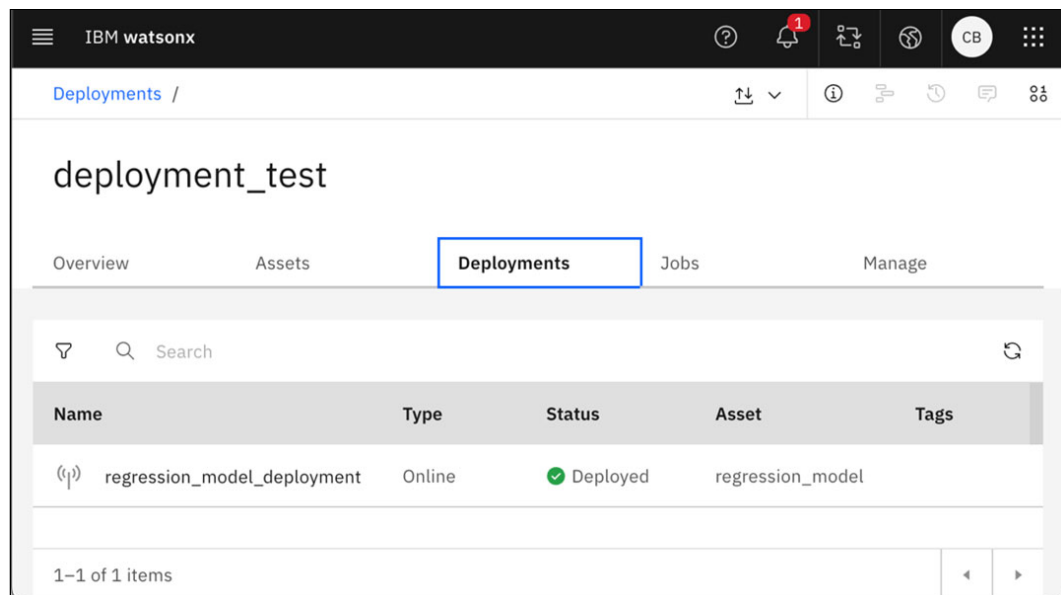


Figure 4-31 Deployment overview

watsonx.ai Studio provides multiple ways to test the deployment, such as the **Test** tab and code snippets (Figure 4-32).

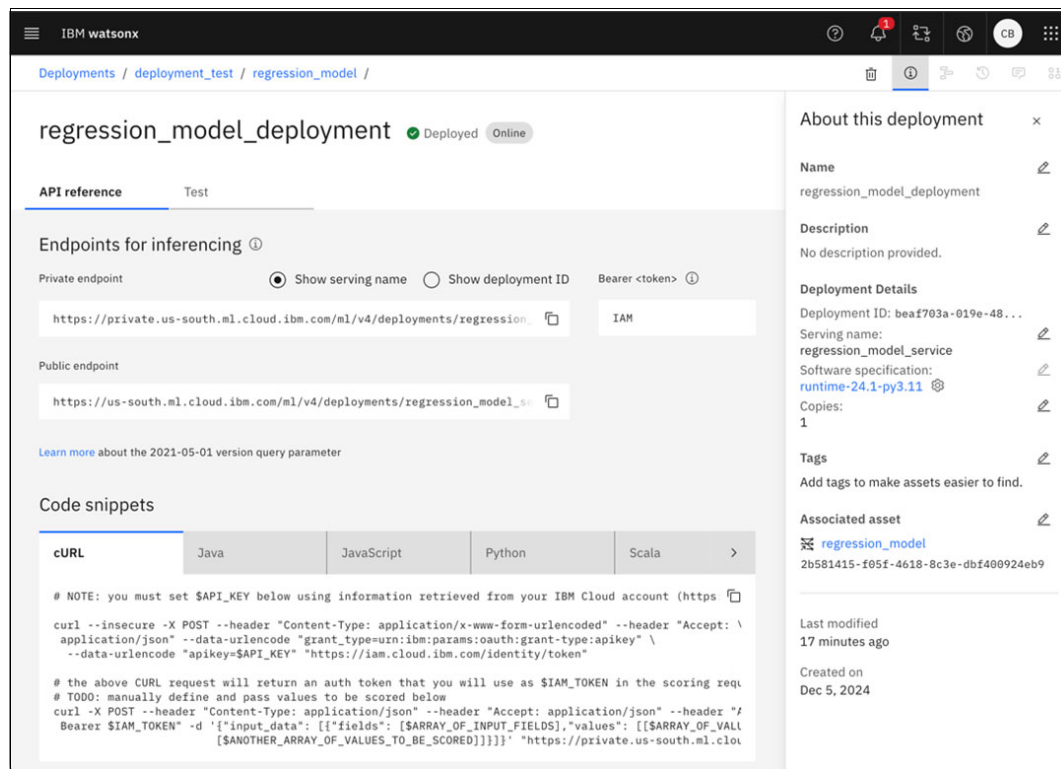


Figure 4-32 Deployment testing

By following these steps, you can effectively deploy and manage your AI models within watsonx.ai Studio, which helps ensure that the models are ready for production use and capable of delivering valuable insights.

4.8 watsonx.ai LLM deployment

Deploying models in watsonx.ai Studio involves several key steps to help ensure that your AI assets are effectively managed and operational. This section provides a concise guide to help you through the process:

4.8.1 Model packaging and exporting

To package and export a model, complete the following steps:

1. Go to the Prompt Lab window, and if you do not already have a prompt set up, populate it by using the example that is shown in Figure 4-33. In this lab, you use a text classification prompt.

The screenshot displays the 'watsonx Prompt Lab' interface. At the top, there are tabs for 'Chat', 'Structured' (which is selected), and 'Freeform'. To the right of the tabs, it says 'AI Model: granite-13b-instruct-v2' with a dropdown arrow. Further right are icons for '{#}', 'TXT', '</>', and a refresh icon.

The main content area is divided into two sections: 'Set up' and 'Try'.

Set up section:

- Instruction (optional)** ⓘ: A text box with the placeholder text 'Tell the model what to do. For example: Summarize the transcript.'
- Examples (optional)** ⓘ: A table with two columns: 'Input:' and 'Output:'.

Input:	Output:
Enter your example input here.	Enter your desired output.

Below the table is a button 'Add example +' and a trash icon.

Try section:

- Test your prompt** ⓘ: A table with two columns: 'Input:' and 'Output:'.

Input:	Output:
You are a text classification bot. Classify the text as either Positive or Negative. I love <u>watsonx</u> .	Generated output appears here.

Below the table is a button 'New test +' and a trash icon.

Figure 4-33 watsonx Prompt Lab

2. Generate a response by the prompt. Keep the decoding method as Greedy, and set the max tokens to 5 to produce Positive and Negative text only (Figure 4-34).

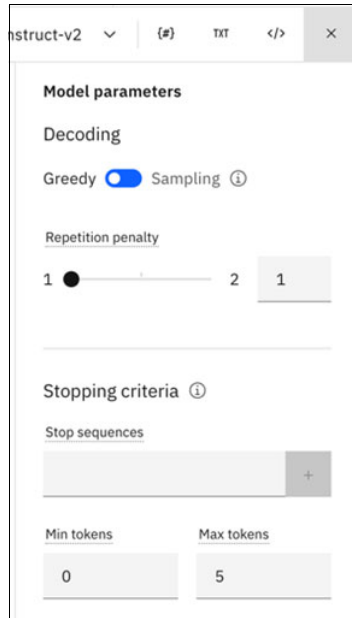


Figure 4-34 Model parameters

3. Click **Generate**, which tests the prompt. Then, click the **View code** icon (Figure 4-35).



Figure 4-35 View code icon

4. Copy the code to a notepad application (Figure 4-36).

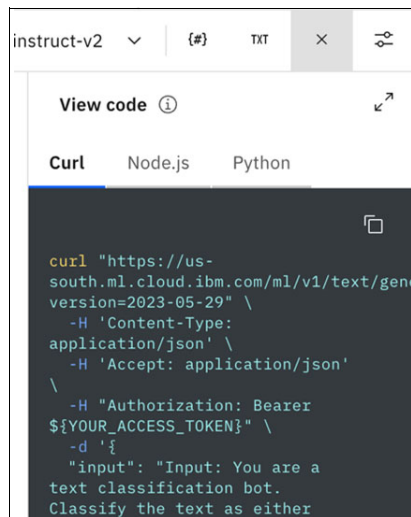


Figure 4-36 Code example

The code (Figure 4-37) is an example of a REST call that starts the model. watsonx.ai also provides a Python API for model invocation, which you review later in this lab.

The header of the REST request includes the URL where the model is hosted and a placeholder for the authentication token. At the time of writing, all users share a single model inference endpoint. In the future, IBM plans to provide dedicated model endpoints.

Security is managed by the IBM Cloud authentication token, which is described later in this section.

The body of the request contains the entire prompt.

```
curl "https://us-south.ml.cloud.ibm.com/ml/v1/text/generation?version=2023-05-29" \
-H 'Content-Type: application/json' \
-H 'Accept: application/json' \
-H "Authorization: Bearer ${YOUR_ACCESS_TOKEN}" \
-d '{
  "input": "Input: You are a text classification bot. Classify the text as either Positive or Negative. \n\nI love watsonx."
}
```

Figure 4-37 Curl command example

5. At the end of the request, you specify the model parameters and the project ID, as shown in Figure 4-38.

```
"parameters": {
  "decoding_method": "greedy",
  "max_new_tokens": 5,
  "min_new_tokens": 0,
  "stop_sequences": [],
  "repetition_penalty": 1
},
"model_id": "ibm/granite-13b-instruct-v2",
"project_id": "b98c2efd-adfe-4052-87f5-ea704260e97c",
```

Figure 4-38 Curl command details

To look up the project ID, select **Project** → **General** → **Manage** in the watsonx.ai project, as shown in Figure 4-39.

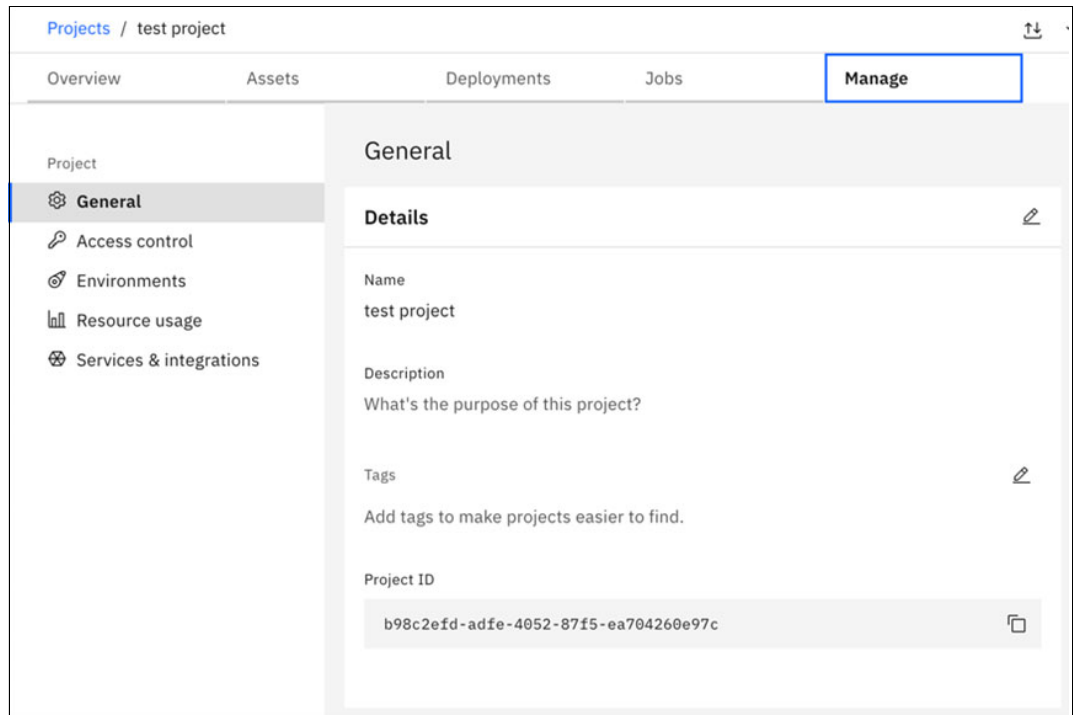


Figure 4-39 Projects Manage view

6. Save the newly configured prompt as a notebook. Select **Standard notebook** as the asset type and then select **Save as**, as shown in Figure 4-40.

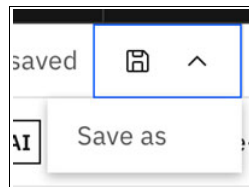


Figure 4-40 Notebook save icon

7. You will now create an authentication token. Open the navigation menu (four horizontal bars) in the upper left of the watsonx interface and select **Access (IAM)**, as shown in Figure 4-41 on page 49.

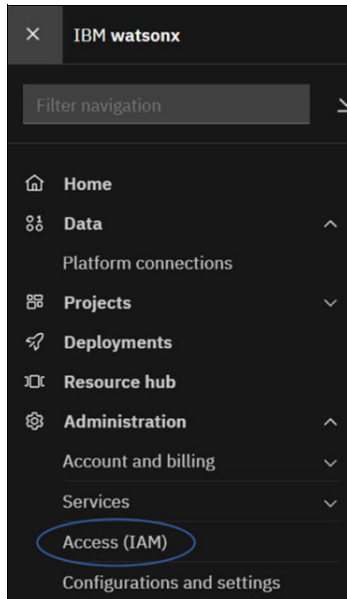


Figure 4-41 Access (IAM) menu item

8. Select **API Keys** → **Create**. Give the token a name and save it in a notepad (Figure 4-42). You use the token in a Python notebook.

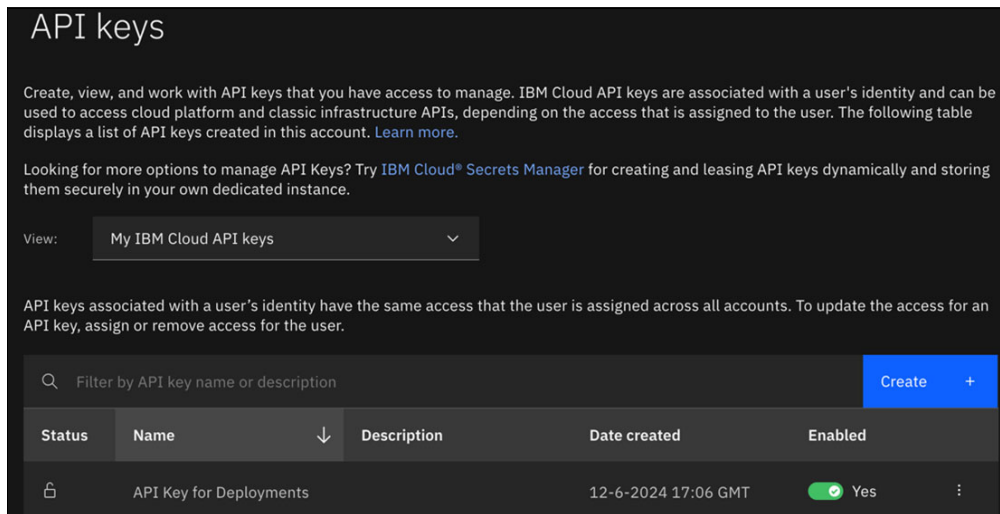


Figure 4-42 API keys

9. Go to your watsonx project and open the notebook that you saved in step 8 (Figure 4-43).

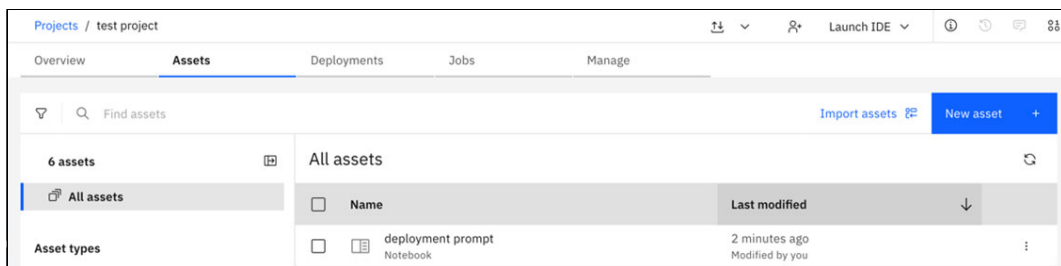


Figure 4-43 watsonx Studio projects overview

10. Review the sample notebook.

This notebook acts as a client application that starts the deployed LLM with a Python SDK. You use the notebook as a client for simplicity of testing during this lab.

Enterprise client applications can be implemented in Python, Java, .NET, and many other programming languages. LLMs that are deployed in watsonx.ai can be started either with REST calls or the Python SDK.

Run the notebook to test the LLM with your prompts.

4.9 Operationalizing machine learning and LLM models

Now that you have a machine learning (ML) model or LLM built, you now enter the operational phase. Much like how traditional application development created the need for formal DevOps tools and systems, so too have AI models created the need for ModelOps. ModelOps is the practice of enabling the deployment and management of models throughout the application development and deployment lifecycle with the goal of operationalizing models in production. As a joint endeavor with traditional DevOps, ModelOps takes the feedback and measurements that are taken in the DevOps lifecycle to iterate on the training, testing and deploying stages of the ModelOps lifecycle.

Key stages in the ModelOps lifecycle include governance, monitoring, deployment of infrastructure, and model versioning. IBM offers various tools to help facilitate these processes, each of which has its own place in the lifecycle. Examples of such tools include the following ones:

- ▶ [IBM watsonx.gov](#) helps govern and monitor model key performance indicators (KPIs).
- ▶ [IBM Instana](#)®™ helps monitor LLM performance, responsiveness, and throughput at the application level.
- ▶ [IBM Turbonomic](#)® can help dynamically scale up and down infrastructure as the workload against your models changes.
- ▶ [IBM API Connect](#)® provides a GUI wizard to create AI-aware APIs and products, plus integration with AI services to forward requests and manage responses.

To demonstrate how these models can be deployed into existing applications, we briefly walk through how to call these models, and integrate them into the DevOps and ModelOps lifecycle.

4.9.1 Calling ML models by using API calls

When you have built a model, your ML model is live and ready to perform as an inference endpoint. This endpoint is your gateway to interact with the model so that you can send data and receive predictions in return. Here, we walk through how you can use it effectively.

Securing your API key

Before making any calls to your endpoint, you need an API key for secure access. For more information about generating this key, see 4.8, “watsonx.ai LLM deployment” on page 45. Here is a quick overview:

1. Go to your deployed model in watsonx.ai Studio.
2. Go to the **Access** tab under the Deployment settings.
3. Generate your API key and store it securely. (You use it to authenticate your requests.)

Making an API call

With your API key in hand, you are ready to communicate with your model. Figure 4-44 shows an example of a simple POST request (by using a cURL command) to send input data and retrieve predictions.

The screenshot shows the IBM Watsonx interface for a deployment named 'regression_model_deployment'. The status is 'Deployed' and 'Online'. The page is divided into several sections:

- API reference**: Includes a 'Test' button and a section for 'Endpoints for inferencing'. It shows a private endpoint and a public endpoint, both with their respective URLs and IAM roles.
- Code snippets**: A tabbed interface showing code snippets for different languages. The 'cURL' tab is active, displaying a command to make a POST request to the private endpoint. The command includes headers for 'Content-Type', 'Accept', and 'Authorization' (using an API key and IAM token).
- About this deployment**: A sidebar on the right providing details about the deployment, including its name, description, deployment ID, serving name, software specification, and associated asset.

Figure 4-44 watsonx.ai regression model deployment tool

Note: The watsonx.ai user interface (UI) prepopulates different ways to call the model, such as cURL, Java, JavaScript, Python, Scala, and others.

Visualizing the process

Figure 4-45 shows a simple way to visualize what occurs during an API call.

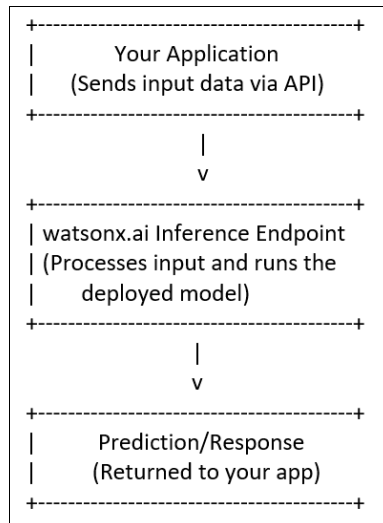


Figure 4-45 API call visualization

Your application sends input data to the endpoint; the model processes the data; and the results are sent back as predictions. Whether you are handling real-time data (through online deployment) or batch processing, this streamlined interaction helps ensure that you can make the most of your deployed model.

4.9.2 Calling Prompt Lab LLM models by using API calls

Calling an LLM model from Prompt Lab is similar to calling an LLM model from watsonx.ai's Studio; the only difference is where to find the code within the UI to do so. To accomplish this task, complete the following steps:

1. Go to Prompt Lab, and then build your prompt in either the **Chat**, **Structured**, or **Freeform** tab.
2. In the upper right, click the **View code** icon (Figure 4-46).

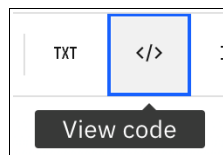


Figure 4-46 View code icon

The Prompt Lab automatically creates everything that you need to copy and paste your prompt into the application of your choosing (in either cURL, Node.js, or Python), as shown in Figure 4-47 on page 53.

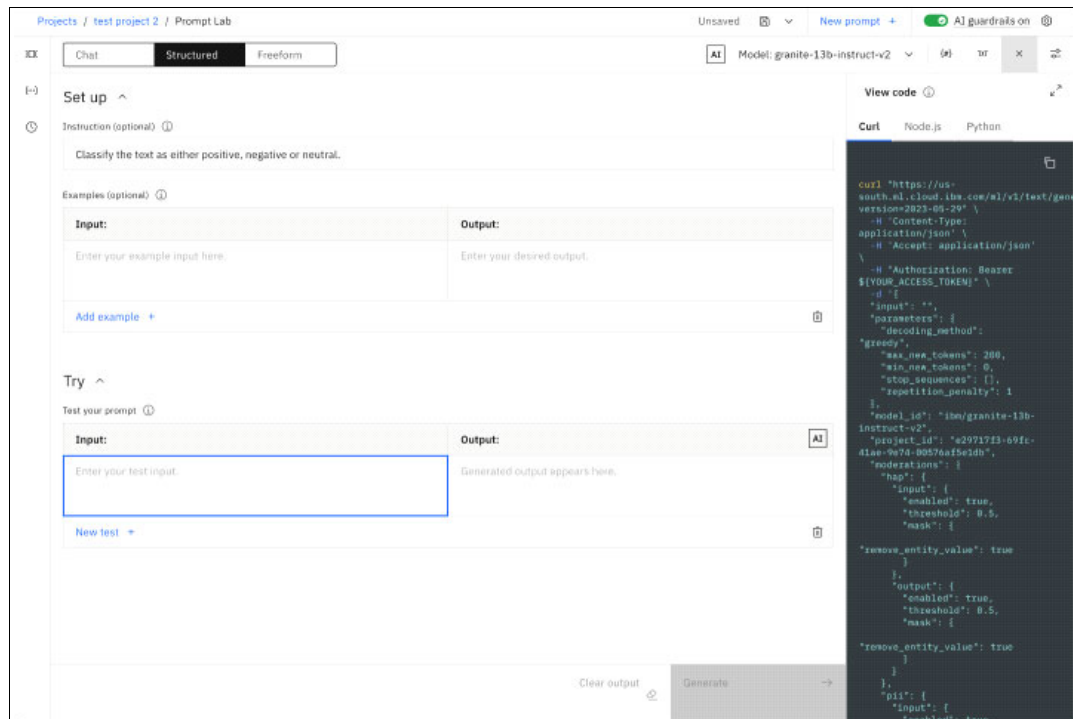


Figure 4-47 watsonx.ai Prompt Lab code window

Note: Include your bearer access token. For more information about this token within the IBM ecosystem, see [Generating a bearer token](#).

4.9.3 IBM watsonx Assistant

Operationalizing AI and ML models by using IBM technology helps ensure seamless deployment and management across enterprise environments while delivering measurable business outcomes. IBM watsonx.ai provides a robust platform for building, fine-tuning, and deploying AI models to enable data scientists to leverage pre-trained models or create custom solutions. Once models are developed, they can be containerized and deployed by using Red Hat OpenShift, which is the IBM enterprise Kubernetes platform, which helps ensure scalability, high availability, and integration with an existing infrastructure. IBM Watson Studio simplifies model lifecycle management by providing end-to-end capabilities for version control, testing, and collaboration. Real-time monitoring is enabled through IBM Instana Observability so that teams can track KPIs, detect anomalies, and maintain model health in production environments.

Figure 4-48 shows an overview of IBM watsonx Assistant.

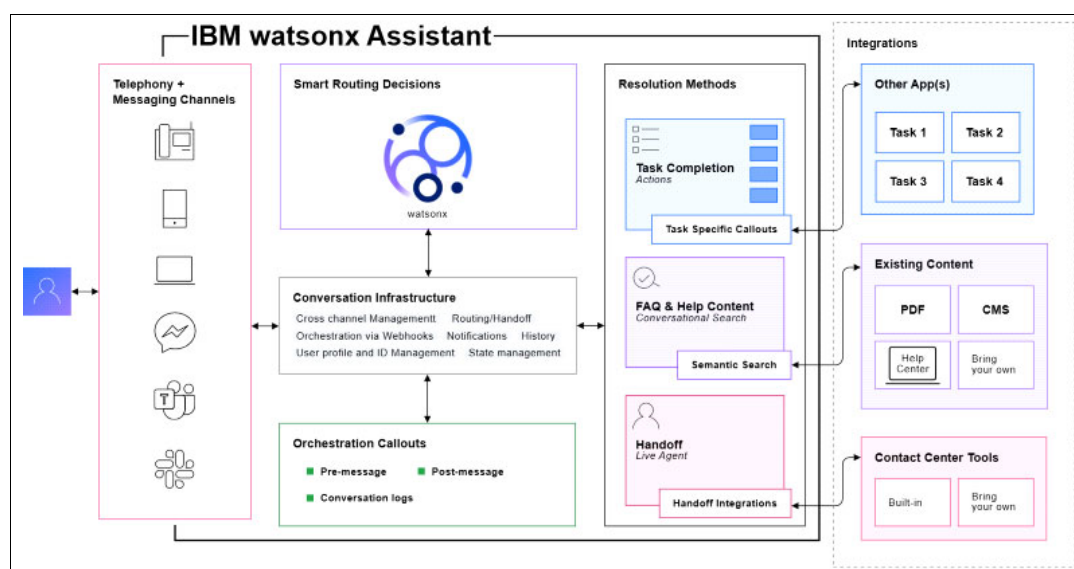


Figure 4-48 IBM watsonx Assistant overview

Ensuring the ongoing success of operational AI and ML solutions requires integrating governance, automation, and business alignment. IBM watsonx.governance enforces responsible AI principles by providing tools for bias detection, lineage tracking, and compliance management, which help organizations meet regulatory requirements and ethical standards. Automated deployment pipelines with IBM DevOps for AI streamline continuous integration and continuous delivery (CI/CD), which enables rapid updates and retraining to address data drift or evolving business needs. Feedback loops that are powered by IBM Watson Discovery facilitate continuous improvement by analyzing real-world user interactions to enhance model performance. By leveraging IBM's AI and hybrid cloud capabilities, organizations can operationalize AI and ML solutions effectively, which drive innovation while helping ensure reliability and trustworthiness.

For more information about tools that are related to operationalizing your models, see 4.10.3, "watsonx.ai data pipeline and orchestration" on page 55.

4.10 Additional information and where to go next

In this chapter, you learned the following things:

- ▶ How to set up your environment, which includes IBM Cloud accounts and project configuration.
- ▶ The key features of watsonx.ai, such as FMs, Prompt Lab, and Tuning Studio.
- ▶ The importance of data quality, cleaning, and ingestion for AI model development.
- ▶ Building and training AI models, which include traditional ML and LLMs, by using tools like AutoAI, Tuning Studio, and InstructLab.
- ▶ Deploying AI models as APIs for real-time or batch processing and integrating them with enterprise systems.

4.10.1 Additional support and documentation

watsonx.ai has extensive support and documentation to help users maximize the platform's capabilities. The [IBM watsonx Documentation Portal](#) offers a comprehensive collection of resources, which include detailed user guides, tutorials, API references, and best practices. Whether you are starting or looking to optimize your AI workflows, the portal helps ensure that you have the guidance that you need to succeed.

Highlights include the following resources:

- ▶ **Getting Started Guides:** Step-by-step instructions for onboarding and initial setup.
- ▶ **Model Development Resources:** In-depth documentation about training, fine-tuning, and deploying both LLM and non-LLM models.
- ▶ **Troubleshooting and FAQs:** Solutions to common issues and tips for resolving challenges efficiently.
- ▶ **Integration Guidance:** Instructions for incorporating watsonx.ai into existing workflows and leveraging tools, such as Hugging Face and BYOM.

This rich repository of knowledge empowers users at every skill level to confidently build, deploy, and scale AI solutions with watsonx.ai.

4.10.2 watsonx.ai API reference

For comprehensive guidance about using watsonx.ai capabilities, the [IBM watsonx API Documentation](#) offers detailed information about available APIs, including endpoints, request parameters, and response structures. This resource is essential for developers that want to integrate watsonx.ai into their applications by providing clear instructions and examples to facilitate seamless implementation. Whether you are working with FMs, performing text inference, or managing deployments, this documentation serves as a valuable reference to help ensure effective and efficient usage of watsonx.ai features.

4.10.3 watsonx.ai data pipeline and orchestration

For more information about data pipelining and orchestration, see the following resources:

- ▶ **IBM Orchestration Pipelines:**
<https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-orchestration-overview.html?context=wx>
- ▶ **IBM Seismic: Introducing AI Agent Orchestration (IBMid required):**
<https://ibm.seismic.com/Link/Content/DCbHf2RCFTPf3G9Cc2PBGggJWfGV>
- ▶ **Instana:**
<https://www.ibm.com/products/instana/generative-ai-monitoring>
- ▶ **Turbonomic:**
<https://community.ibm.com/community/user/aiops/blogs/cheuk-hung-lam/2024/05/28/turbonomic-tackles-gpus-for-genai-workloads>
<https://www.ibm.com/case-studies/ibm-big-ai-models-turbonomic>



Advanced capabilities of watsonx.ai

watsonx.ai embodies IBM's extensive expertise in artificial intelligence (AI) and foundation models (FMs), which combine advanced AI research with practical tools to make large language models (LLMs) efficient and versatile across many applications. Underlying watsonx.ai is its integration of FMs that are tuned to accelerate and optimize business operations. These models are positioned to perform at the intersection of language understanding, structured data processing, and knowledge retrieval, which enhance the ability to extract, refine, and use vast amounts of unstructured data.

The platform's capabilities extend beyond simple text processing to include complex interactions between structured and unstructured data sources, which enable the model to draw relevant information and learn domain-specific knowledge. For example, watsonx.ai supports both general-purpose and highly specialized model architectures, which facilitate the design of task-optimized LLMs that serve nuanced business needs while ensuring data privacy and regulatory compliance. Its multi-modal capabilities enable seamless handling of diverse data types (such as text, image, and audio inputs) and applications to traverse disparate data landscapes cohesively, which achieves a high level of contextual relevance and adaptability.

The watsonx.ai platform has several capabilities to support advanced use cases such as prompt engineering, multi-task prompt tuning, and fine-tuning.

The following topics are described in this chapter:

- ▶ 5.1, "Prompt engineering" on page 58
- ▶ 5.2, "Multitask prompt tuning" on page 61
- ▶ 5.3, "Fine-tuning" on page 64
- ▶ 5.4, "InstructLab" on page 67

5.1 Prompt engineering

Prompt engineering within the watsonx.ai ecosystem serves as an essential component in harnessing the full potential of language models. By precisely framing prompts, users can guide LLM responses toward relevance and coherence, which greatly enhances the utility of generated outputs. The process of prompt engineering in watsonx.ai is highly nuanced, and it involves detailed adjustments to phrasing, context, and iterative feedback mechanisms to yield wanted outputs consistently. Prompt engineering plays a pivotal role in directing LLMs to perform specific tasks with high precision, a task that requires linguistic adjustments and a deep understanding of the underlying model dynamics.

The reason why prompt engineering is important is because it is a way to make generalist models (LLMs) perform a specific task. Without quality information, well-defined instructions, and a clear set of examples, the model might misbehave and hallucinate in various ways.

Prompt engineering is about writing something in a better, clearer, and cleaner form. It also involves using specific system tokens that are for only the model that is used and, if available, a series of examples that provide better information to the model so that it can perform as intended. This approach is explored more in-depth in this section.

5.1.1 Prompting techniques

There are three main prompting techniques:

- ▶ Zero-shot prompting
- ▶ One-shot prompting
- ▶ Few-shot prompting

These techniques are not learning techniques, but prompting techniques only, which improve model performances at inference-time without modifying the original model.

These techniques leverage the model's existing capabilities without requiring fine-tuning or parameter updates, which make them lightweight and adaptable solutions for various use cases. This section provides an explanation of each prompting technique, and when it is best to use each approach.

- ▶ **Zero-shot prompting:** Relies solely on the model's pre-trained knowledge to generate responses without providing any task-specific examples in the prompt. Instead, the input typically includes clear instructions or a well-defined query that guides the model to perform the task, for example, asking a model to summarize a paragraph or translate a sentence into another language without providing prior examples. The effectiveness of this approach hinges on the clarity and precision of the prompt and the model's inherent ability to generalize across diverse tasks.
- ▶ **One-shot prompting:** In this approach, a single example of the task is embedded within the prompt, alongside the query or instructions. This example serves as a reference for the model to infer the behavior. By including a single demonstration, one-shot prompting can enhance performance for tasks that require nuanced or domain-specific understanding because it provides a concrete context for the model to interpret the instructions.

- **Few-shot prompting:** Expands on this concept by incorporating multiple examples of the task in the prompt. The additional examples provide a richer context and help the model better understand complex patterns or subtle variations in the task. Few-shot prompting is useful for tasks that require multi-step reasoning, handling of ambiguous inputs, or understanding domain-specific jargon. However, it demands careful prompt construction to balance informativeness and brevity because excessive length can lead to token limitations or diminished performance.

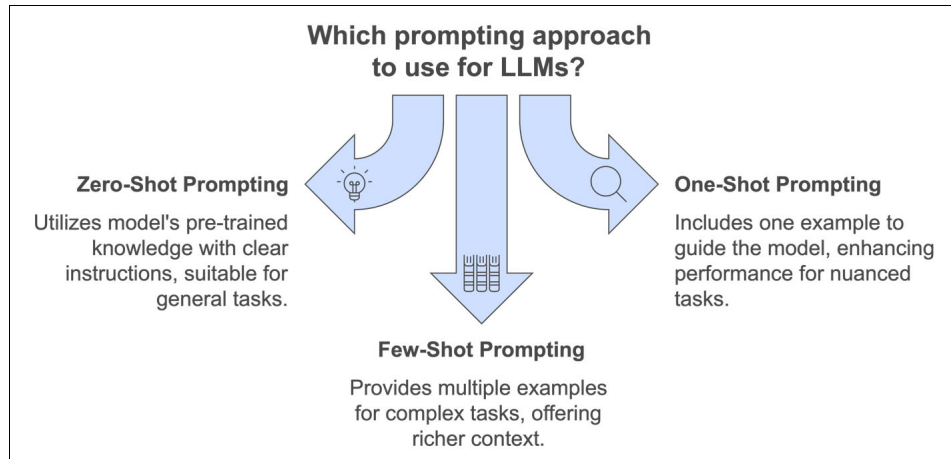


Figure 5-1 LLM prompting method types overview

5.1.2 Importance of system tokens

In addition to zero-shot, one-shot, and few-shot prompting techniques, *system tokens* (also known as system-level instructions or control tokens) play a critical role in crafting effective prompts. These tokens provide metadata or guidance to steer the behavior of the language model at a higher level, often defining the context, tone, or expected behavior of the model during inference. Importantly, the implementation and interpretation of these tokens can vary across different models, making their effective use model specific.

System tokens enable users to establish a “role” or context for the model, shaping its responses beyond what is specified in the natural language prompt.

By embedding these tokens, users can accomplish the following goals:

- **Control output behavior:** Ensure the order of prompt completions between the main prompt areas of System, Assistant, and User
- **Reduce ambiguity:** Guide the model's interpretation of the task, especially in contexts where instructions alone might be misinterpreted.
- **Enhance few-shot learning:** When combined with example-based prompting, system tokens can provide an overarching framework that amplifies the impact of the examples.

5.1.3 Model-specific peculiarities

Different language models interpret and use system tokens in unique ways due to their architecture and pre-training data. The best practices for incorporating system tokens are to understand model documentation. Because the behavior of system tokens is model-dependent, consulting the model's technical documentation is essential to understanding how tokens are implemented and what variations are supported.

5.1.4 How watsonx.ai supports prompt engineering

Regarding prompt engineering, the simplest way to interact with LLMs is extensive but peculiar. Fortunately, the watsonx.ai platform enables prompt engineering by providing a series of tools for its usage. Figure 5-2 shows a series of these tools in the Prompt Lab section of watsonx.ai.

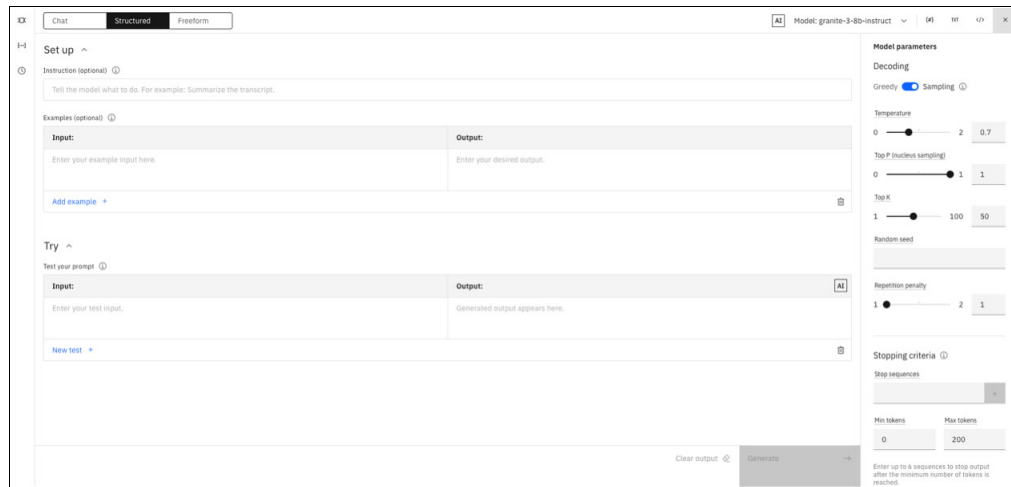


Figure 5-2 watsonx.ai Prompt Lab dashboard

Prompt Lab is the main area to access and interact with LLMs. Here, it is possible to interact with models and perform prompt engineering techniques. In Prompt Lab, users have three modes to select from to interact with LLMs:

- ▶ **Chat mode:** Provides a simplistic, multimodal chatbot-like interaction with capabilities such as memory and document understanding. It is good for model interaction and for simple system prompt definition and a zero-shot-prompting approach.
- ▶ **Structured mode:** Provides a way to set up your prompt to create a particular prompt engineering setting within the main model Instructions and Examples, where the examples are the input/output series that is provided in a few-shot-prompting setting. It automatically applies the system tokens for a specific model to best fit the one/few-shot-prompting setting.
- ▶ **Freeform mode:** Provides more advanced users with the option to be free of crafting their raw prompt by using all the possible prompt engineering capabilities that leverage raw system tokens. Although this mode leverages the power of prompt engineering and freedom, it requires specific skills in understanding system tokens, what they are for a specific model, and how to use them.

Although prompt engineering provides a faster way to the objective, it is not always the best tool to use or the most capable tool that is available. As task difficulty and complexity increases, watsonx.ai can use more advanced model enhancing techniques. We explore these techniques in the following sections of this chapter.

5.2 Multitask prompt tuning

Multitask prompt tuning within watsonx.ai builds on traditional prompt engineering by implementing adaptive mechanisms to refine the prompt's interpretative accuracy over time. This approach differs fundamentally from prompt engineering because it modifies prompt content and continuously aligns the model's interpretative layers with domain-specific expectations. In essence, prompt tuning enables models to retain learned adjustments across sessions, which support consistency and reduce the need for extensive re-engineering.

Prompt tuning leverages techniques such as embedding adjustments and parameter scaling to influence the model's internal state and guide responses within boundaries. Using the watsonx.ai dynamic configuration settings, developers can set up continuous tuning processes that adapt prompts based on evolving business contexts, which lead to a finely calibrated model that reflects current operational realities. This capability enables rapid adaptation without costly retraining, and the watsonx.ai architecture permits this tuning to take place seamlessly, which enables real-time adjustments as new data is ingested or as user preferences change.

In this setting, it is not the LLM that is modified. Instead, a dedicated, smaller LLM is trained to generate the best possible prompt adjustment for each prompt in the input. The smaller LLM leverages the system tokens that are available for each LLM on watsonx.ai and produces new, compatible virtual tokens to enhance the performances. The smaller LLM is trained by using a loss function that accounts for the resulting response from the immutable (in this setting) generative LLM model that you want to improve, and adapts its weights to create better prompts through a *tunable soft prompt*, as shown in Figure 5-3.

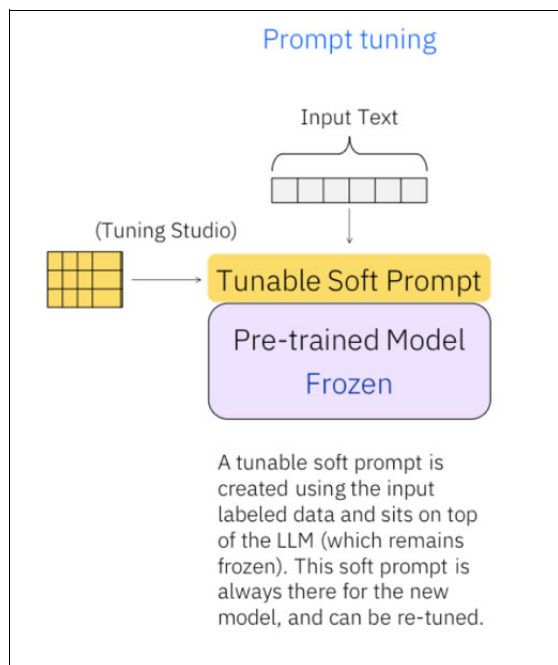


Figure 5-3 Prompt tuning overview

With prompt tuning, you create a model that automatizes the prompt engineering task, which makes it dynamically adaptive to new, incoming inputs over time. The key benefit of prompt tuning is performing tuning in ways that are better than what experts can do for certain tasks, that is, the best token leading to a successful completion of the input task. Prompt tuning can do this task because it leverages 100 - 10000 examples to learn which token is the best one to add to a starting pre-engineered prompt to minimize the loss of the generative model.

The watsonx.ai platform provides a simplistic way of using prompt tuning, as shown in Figure 5-4.

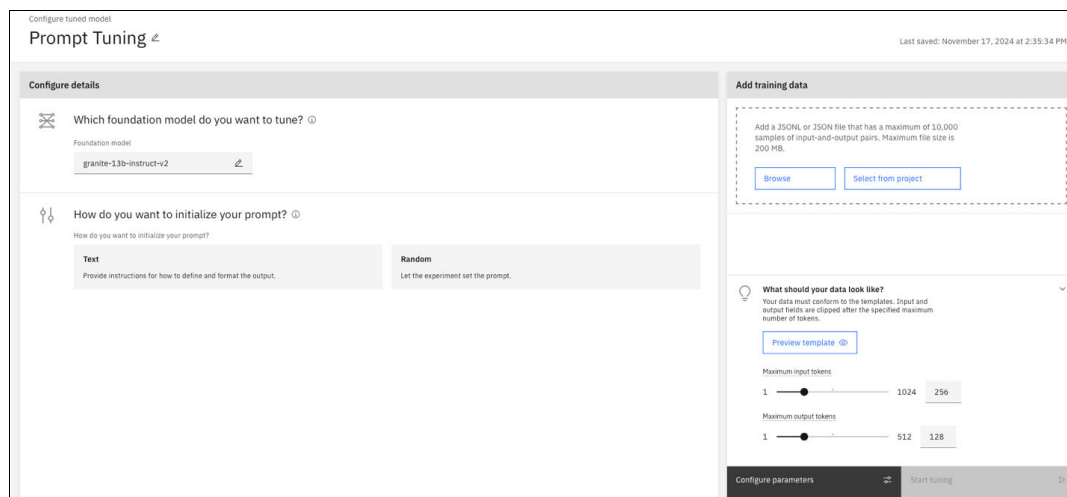


Figure 5-4 Prompt tuning in watsonx.ai Tuning Studio

5.2.1 Prompt tuning parameters

In watsonx.ai Tuning Studio, you can use multitask prompt tuning by leveraging various prompt tuning parameters. The process of optimizing hyperparameters for prompt tuning, such as batch size, the number of epochs, the learning rate, and accumulation steps, plays a critical role in achieving task-specific adaptation and helping ensure effective usage of LLMs. Each of these parameters impacts the training process in unique ways by influencing the generalization ability, stability, computational efficiency, and performance of the fine-tuned prompts.

- *Batch size* refers to the number of training samples that are processed simultaneously during each forward and backward pass through the model. It is a fundamental factor in determining the balance between computational efficiency and the quality of gradient updates. Larger batch sizes tend to stabilize gradient updates by averaging over more samples, which enable faster convergence. However, they often require significant computational resources and might overlook fine-grained variations in the dataset, which might potentially limit the prompt's ability to address nuanced tasks. Conversely, smaller batch sizes enable greater granularity in gradient computations, which is advantageous for small datasets or specific tasks. However, small batches introduce noisier gradient updates, which require more iterations to converge effectively. To optimize batch size, practitioners should aim for a balance that satisfies computational feasibility while meeting the requirements of the task. Dynamic batch sizing (adjusting the batch size during training) can further stabilize the learning process and enhance overall effectiveness.
- The *number of epochs* represents the total number of complete passes that the training algorithm makes through the dataset. This parameter directly influences how thoroughly the model explores the data to refine its prompt parameters. A higher number of epochs allows the model to capture intricate patterns, which improves task-specific adaptation. However, this approach comes with the risk of overfitting, especially with smaller datasets, which reduce the generalization of the prompts. Conversely, a lower number of epochs minimizes the risk of overfitting but might lead to underoptimized prompts that fail to leverage the model's full potential. To strike the right balance, monitor validation loss and apply early stopping criteria to help ensure that training halts before overfitting occurs. Using pre-trained checkpoints can also reduce the need for extensive epochs because these starting points encapsulate foundational knowledge that accelerates convergence.

- The *learning rate* governs the size of the updates that are made to prompt parameters during each optimization step. It influences the speed and stability of the training process. A high learning rate expedites convergence, which reduces training time but risks overshooting optimal solutions, and can lead to suboptimal performance or even divergence. Conversely, a low learning rate enables a more precise exploration of the parameter space, which increases the likelihood of finding an optimal solution at the cost of prolonged training. Effective strategies include employing learning rate schedules, such as cosine decay or step-based decay, which adjust the learning rate dynamically during training. Warm-up strategies, where the learning rate gradually increases at the start of training, can also mitigate initial instability and improve overall training robustness.
- The concept of *accumulation steps* addresses memory constraints by enabling gradient accumulation across several mini-batches before updating the model's parameters. This approach effectively simulates larger batch sizes without exceeding hardware memory limits, which make it valuable for memory-constrained environments. Accumulation steps smooth gradient updates by averaging across multiple mini-batches, which improve stability at the cost of increased training time. Optimizing this parameter involves selecting an accumulation step size that balances memory efficiency with the effective batch size. Combining this approach with batch size tuning can further optimize resource usage and enhance performance.

5.2.2 Interdependencies and holistic tuning strategies

These hyperparameters are interdependent because changes in one can influence the behavior of others. For example, increasing the number of epochs without modifying the learning rate might lead to overfitting, and combining a high batch size with too few epochs might result in undertrained prompts. To navigate these interdependencies, practitioners can employ regularization techniques such as data augmentation to counteract overfitting in high-epoch scenarios. Gradient clipping can also be used to prevent instability during training, particularly when high accumulation steps are involved.

Performance metrics, which include task-specific measures like accuracy or F1-scores, should guide the evaluation of prompt tuning effectiveness. Monitoring loss convergence and gradient stability help ensure that the chosen hyperparameters lead to tangible improvements.

Carefully calibrating batch size, the number of epochs, the learning rate, and accumulation steps enables precise optimization of prompt tuning, which unlocks the full potential of LLMs for specific tasks. By managing these parameters holistically, practitioners can achieve performance gains while balancing computational efficiency and resource constraints.

5.3 Fine-tuning

Fine-tuning within the watsonx.ai ecosystem represents the next layer of model specialization, where foundation LLMs undergo retraining on domain-specific datasets to enhance accuracy and relevance for specific applications. Fine-tuning goes beyond prompt adjustments by modifying the model's weights to encode new knowledge or adapt to complex industry-specific language structures, terminologies, and operational protocols. This process is beneficial for industries that require high precision in terminology and context, such as healthcare, finance, and legal services. Fine-tuning within the watsonx.ai ecosystem exemplifies a sophisticated approach to model specialization, which enables foundation LLMs to adapt to domain-specific needs through retraining. Unlike prompt tuning, which focuses on lightweight modifications to steer model behavior, fine-tuning directly alters the model's weights to encode new knowledge or align with complex industry-specific language structures and terminologies.

5.3.1 Challenges with fine-tuning

To fully appreciate the significance of watsonx.ai capabilities, it is essential to understand the inherent complexity of fine-tuning in general. At its core, fine-tuning involves retraining a model's internal weights on carefully curated datasets to refine its understanding of specific terminologies, language patterns, or operational protocols. This process differs from lightweight techniques like prompt tuning, which adjusts model behavior externally without altering its core structure. Fine-tuning, by contrast, modifies the model itself, embedding new knowledge directly into its architecture.

While this approach enables unparalleled precision and customization, it introduces many challenges. Here are some common challenges with fine-tuning:

- **Data management:** One of the primary difficulties with fine-tuning. Fine-tuning demands datasets that are relevant and meticulously prepared, which includes ensuring that the data is formatted correctly, free from bias, and representative of the target domain. Even determining the appropriate size of the dataset requires careful consideration because too little data risks underfitting, and too much can lead to overfitting or unnecessary computational burdens. For example, in watsonx.ai, datasets are limited to 200 MB for JSON or JSONL files, or up to 10,000 examples when sourced from connected data stores. These constraints are carefully balanced to optimize efficiency without sacrificing performance, but managing these parameters manually would be daunting for most users.
- **Precise calibration of numerous hyperparameters:** These hyperparameters include the learning rate, batch size, number of training epochs, and strategies for regularization, among others. Each of these parameters is interdependent, which means that altering one can have cascading effects on others. Achieving the optimal configuration often involves extensive trial-and-error or the usage of advanced hyperparameter optimization techniques like Bayesian search. This complexity is compounded when working with large models, which can have billions of parameters, which require significant computational resources and expertise to manage effectively.

- Computationally demanding workloads requiring a high-performance architecture: Large models can have billions of parameters, which require a significant amount of computational resources. Even with the right data and parameters in place, fine-tuning remains computationally demanding. Training these large LLMs requires access to high-performance hardware, such as multi-GPU, along with robust memory and storage capabilities. For organizations without dedicated AI infrastructure, these requirements are often prohibitive.
- Ensuring stability during the training process: Large-scale optimization algorithms are prone to issues like exploding or vanishing gradients, so achieving convergence without diverging from the optimal solution requires careful tuning and monitoring.

5.3.2 How watsonx.ai addresses fine-tuning challenges

By automating the complexities of fine-tuning, watsonx.ai transforms what was once a labor-intensive and technically demanding process into an accessible, streamlined experience. This approach lowers the barrier to entry for organizations looking to adopt AI and enables experienced practitioners to focus on higher-level strategic goals rather than getting bogged down in technical minutiae. With its combination of cutting-edge technologies, managed infrastructure, and user-centric design, watsonx.ai empowers businesses to harness the full potential of fine-tuning, which unlocks new levels of precision, efficiency, and innovation in AI-driven solutions.

watsonx.ai provides the following features:

- Hardware and resource allocation automation: The platform's automation begins with its ability to manage hardware and resource allocation seamlessly. Users do not need to worry about provisioning servers, configuring GPUs, or scaling their setups to accommodate large datasets or models. Instead, watsonx.ai handles these tasks behind the scenes, helping ensure that every fine-tuning operation runs on optimized configurations.
- Supervised Fine-Tuning Trainer (SFTTrainer): At the heart of watsonx.ai fine-tuning capabilities is the SFTTrainer, which is a powerful tool that is developed in collaboration with Hugging Face. This framework simplifies the optimization of model weights by automating key aspects of the training process, which includes the application of advanced learning rate schedules and warm-up strategies. These techniques are crucial for maintaining stability during training, particularly when dealing with complex or high-dimensional data. By leveraging SFTTrainer, watsonx.ai helps ensure that models converge rapidly and reliably without the need for extensive manual intervention.
- Low-rank adaptation (LoRA) and quantized low-rank adaptation (QLoRA): In addition to SFTTrainer, watsonx.ai incorporates cutting-edge techniques like LoRA and QLoRA. These methods represent a paradigm shift in fine-tuning efficiency. Rather than retraining all of a model's parameters, LoRA focuses on fine-tuning small modular blocks of weights while freezing most of the model. This approach reduces the computational and memory requirements of the process, which makes fine-tuning accessible even on resource-constrained hardware. QLoRA goes a step further by lowering the precision of certain parameters during training, which further optimizes performance without compromising accuracy. These innovations enable watsonx.ai to deliver results faster and with fewer resources than traditional approaches.
- Tuning Studio integration: Another key advantage of watsonx.ai is its integration with the Tuning Studio, which provides access to a library of pre-configured model templates. These templates enable users to build on pre-existing architectures that are optimized for specific tasks or domains. This approach eliminates the need to design custom models from scratch, which reduces the time and expertise that are required to initiate fine-tuning projects.

- **Custom FMs:** For organizations with unique requirements, watsonx.ai also supports the import of custom FMs if they have fewer than 20 billion parameters. This approach opens the possibility of automatically fine-tuning tons of available models on Hugging Face and on watsonx.ai. This flexibility helps ensure that the platform can accommodate a wide range of use cases and industries.
- **Monitoring and optimization:** Throughout the fine-tuning process, watsonx.ai provides robust tools for monitoring and optimization. Real-time performance tracking enables users to assess key metrics, such as validation loss and gradient stability, which helps ensure that the model improves as expected. The platform also employs advanced early-stopping mechanisms to prevent overfitting by halting training when further iterations would yield diminishing returns. These features enhance the efficiency of the process and improve the quality of the final model.

Figure 5-5 shows the watsonx.ai fine-tuning process.

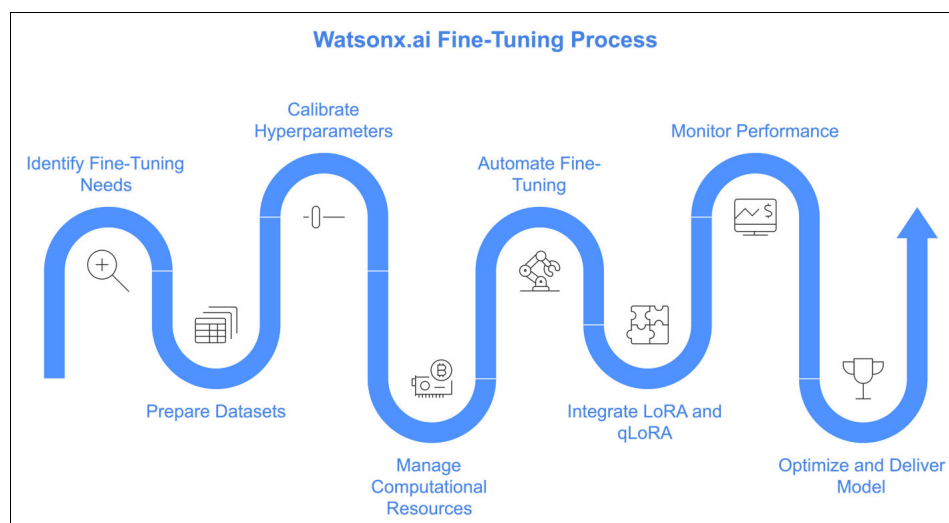


Figure 5-5 Prompt fine-tuning pipeline process overview

In conclusion, watsonx.ai revolutionizes the traditionally complex and resource-intensive process of fine-tuning LLMs by introducing a seamlessly automated, highly efficient, and scalable solution. By leveraging advanced tools like the SFTTrainer, innovative techniques such as LoRA and QLoRA, and a robust Tuning Studio, watsonx.ai enables businesses to achieve unparalleled levels of customization and precision in their AI solutions. Its ability to manage every aspect of the fine-tuning lifecycle (from data preparation and parameter optimization to resource allocation and model monitoring) removes significant technical barriers, which democratize access to AI specialization for organizations of all sizes.

This comprehensive platform empowers businesses to tailor FMs to their unique domain-specific needs, whether in healthcare, finance, legal services, or other fields that require high precision. By doing so, the platform enhances the relevance and accuracy of AI applications and accelerates time-to-value while reducing the costs that are associated with traditional fine-tuning methods. watsonx.ai stands as a testament to IBM's commitment to innovation and accessibility by providing a robust foundation for businesses to unlock the transformative potential of AI with confidence and ease.

5.4 InstructLab

InstructLab represents a groundbreaking shift in the way LLMs are fine-tuned by making the process more accessible, flexible, and efficient. At its core, InstructLab leverages a unique combination of community-driven input, synthetic data generation (SDG), and iterative training methodologies to refine LLMs in a way that dramatically lowers the barriers to entry for fine-tuning tasks (see Figure 5-6). This process makes it simpler for developers and subject matter experts (SMEs) to improve model outputs. It also accelerates the fine-tuning cycle and reduces the computational overhead that is traditionally associated with customizing models.

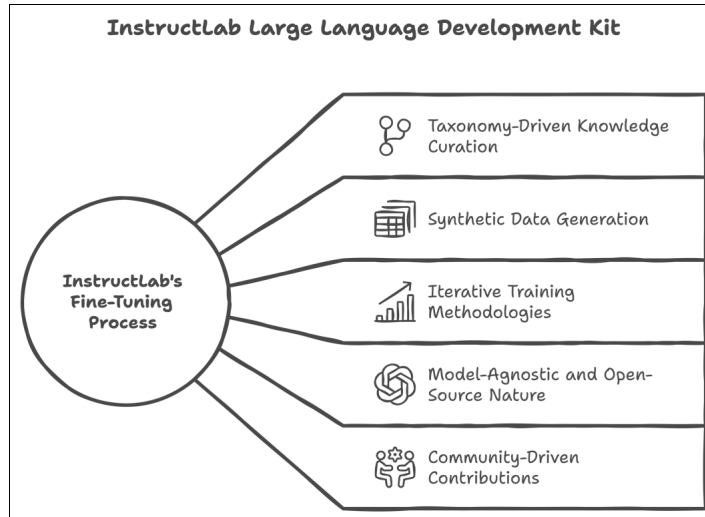


Figure 5-6 The InstructLab large language model development kit functions overview

The InstructLab approach to fine-tuning is heavily grounded in the concept of taxonomy-driven knowledge curation. *Taxonomies*, in this context, are structured frameworks of concepts and relationships that organize information into logical categories and subcategories. These taxonomies are built collaboratively by SMEs, and they serve as the foundation for the knowledge that is used to tune the model. For example, if a business wanted to fine-tune an LLM on customer support for a specific industry, an SME in that industry would work with a taxonomy that represents common questions, issues, and terminology that are relevant to the field. By formalizing domain knowledge in a taxonomy, InstructLab helps ensure that the model fine-tuning process is precise, efficient, and contextually relevant.

What makes this approach powerful is that the knowledge that is curated in these taxonomies is used to generate synthetic data. Unlike traditional fine-tuning methods that rely heavily on vast amounts of manually labeled training data, InstructLab uses SDG to produce the training examples that are needed to adjust model behavior. This task is accomplished by feeding the curated taxonomies into the system (composed of Knowledge and Skills taxonomies), which contains question-answer pairs and other types of data.

An example of Skills taxonomies is provided in Figure 5-7, where the Skill that is defined here is made to make a model learn to better interpret particular tables.

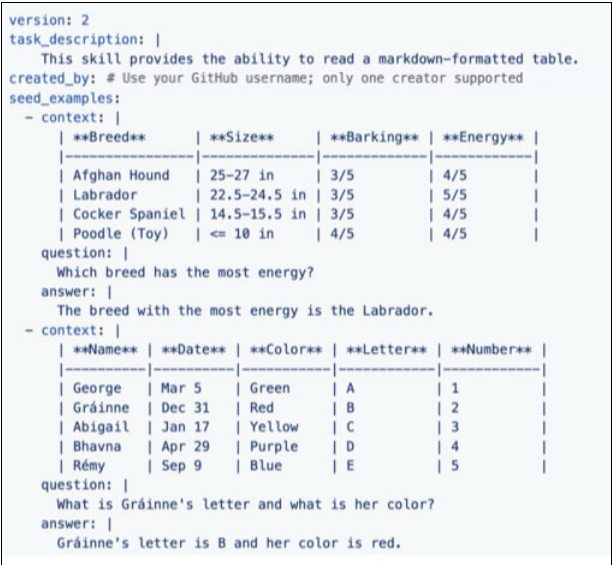


Figure 5-7 Skill Taxonomy example

This data generation process is not random: It follows predefined patterns based on the taxonomy’s structure, which helps ensure that the synthetic examples are highly relevant to the domain. This synthetic data serves as a proxy for real-world data, which enables InstructLab to adapt to niche topics or specialized knowledge areas without requiring the collection of large-scale, expensive datasets. The synthetic data can also be customized and controlled by the SME, which means that they can influence the generation process to help ensure that the model is trained on the most important or critical examples. The ability to generate synthetic training data directly from taxonomies is what makes InstructLab so efficient: It drastically reduces the need for large-scale manual data curation and opens. Furthermore, it solves the ever-existing problem of not having enough data for fine-tuning a model that is tailored for a particular business need in the generative AI (gen AI) era.

The InstructLab approach is built around iterative feedback and instruction training, which are two techniques that further streamline the fine-tuning process. Instruction training involves providing LLMs with explicit instructions (similar to how humans learn new tasks) about how to generate responses based on the synthetic data. This process is highly flexible because the SMEs can continually tweak the instructions and the knowledge base based on the evolving needs of the model and the domain it is being trained on. This process uses a 2-phase approach with a replay that serves the purpose of ensuring high diversity and quality in the synthetically generated instruction-tuning dataset while ensuring training stability. It also prevents catastrophic forgetting, which is a common situation that happens in the FM fine-tuning process when it is not well controlled.

Once the initial synthetic data is generated from the curated taxonomy, the InstructLab system trains the model by providing it with a series of questions and answers that align with the knowledge base. The process is iterative, which means that the model does not undergo a single round of training and then stop. Instead, as new feedback is gathered from the model’s performance, the data is refined, and further training is conducted. This iterative feedback loop is crucial in guiding the model toward better understanding, more accurate outputs, and more aligned responses to specific use cases.

The power of iterative feedback lies in its ability to refine the model's responses over time, which enhances its accuracy and applicability to real-world problems. SMEs can continuously assess the model's performance in relation to specific tasks or topics, and helps ensure that the model becomes progressively better at understanding the subtleties of the domain and generating more precise, contextually relevant responses.

Another key feature of InstructLab is its model-neutral and open-source nature. Unlike proprietary fine-tuning solutions that are often tightly coupled with specific models or platforms, InstructLab enables users to contribute to the fine-tuning of various LLMs regardless of their underlying architecture. InstructLab is built on the premise that fine-tuning should be an open, community-driven process. It supports a wide range of open-source LLMs from repositories like Hugging Face, which enables users to choose the model that best suits their needs, and even enabling them to experiment with models from different frameworks. The open-source nature of InstructLab also means that users have full transparency into the fine-tuning process and the ability to modify it as needed. Anyone from hobbyists to industry experts can contribute to improving the system, whether by adding new taxonomies, adjusting training data, or developing new techniques for SDG. This approach makes InstructLab a true community-driven initiative that is always evolving based on the needs and contributions of its users.

Figure 5-8 shows a community-driven InstructLab fine-tuning process example.

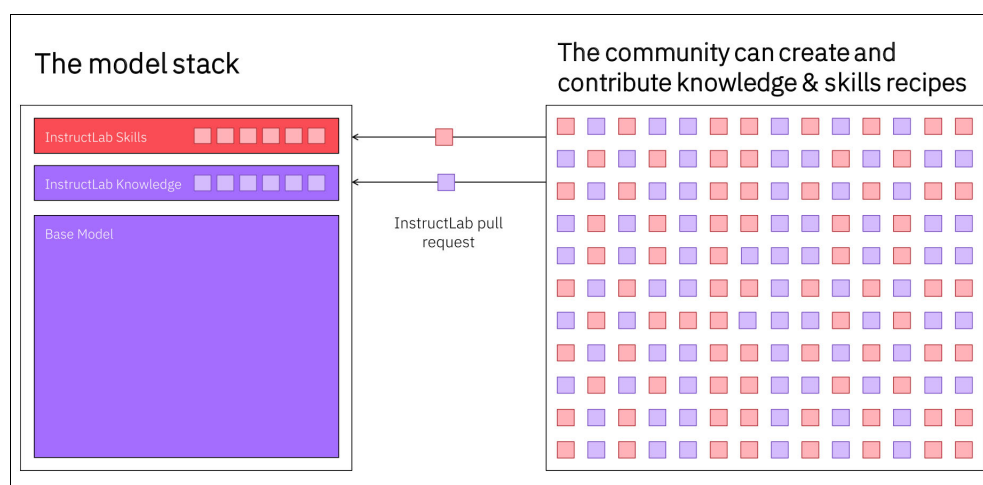


Figure 5-8 Community-driven InstructLab fine-tuning process example

Using InstructLab involves several key steps, each of which is designed to make the process as efficient and accessible as possible:

1. Users download a base model from a supported repository, such as Hugging Face, and initialize the InstructLab command-line interface (CLI).
2. Once the environment is set up, the user creates a knowledge base, which is stored in a structured directory that follows the taxonomy format. This knowledge base is populated with question-answer pairs, references to external documents (for example, PDFs or markdown files), and metadata to describe the knowledge, such as the domain and relevant attributes.
3. Now, you generate synthetic data based on the knowledge base. This synthetic data is used to train the model, with the InstructLab system automatically generating examples that adhere to the structure of the taxonomy. The SDG process is designed to produce high-quality training samples by following the patterns and relationships that are defined in the taxonomy, which reduces the time and cost of manual data creation and helps ensure that the model receives high-quality, domain-specific examples.

4. Once the synthetic data is generated, the model is trained by using the InstructLab training framework. This training can be done on a local machine or in a cloud environment, with the flexibility to use GPUs to accelerate the process.
5. After training is complete, the model undergoes a testing phase to ensure that it performs as expected. InstructLab enables users to run tests that evaluate how well the model answers questions and generates responses, which help identify areas for further improvement.
6. The final stage is deploying the trained model, which can be done through the InstructLab serving tools. Once the model is deployed, users can interact with it through the CLI or through an application interface, which enables them to assess how well it handles real-world queries and performs in live environments.

Figure 5-9 shows a high-level view of the InstructLab pipeline.

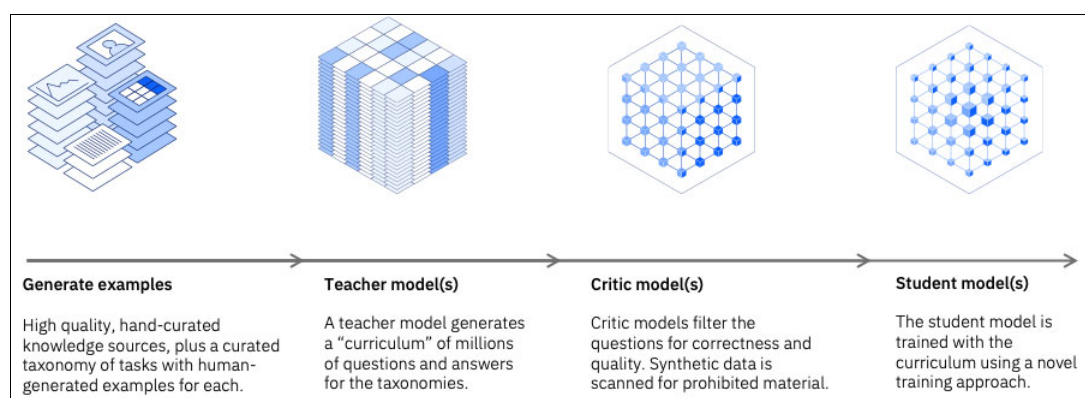


Figure 5-9 High-level overview of the InstructLab pipeline

5.4.1 Advantages of InstructLab

The InstructLab methodology provides numerous advantages over traditional fine-tuning techniques:

- ▶ **Synthetic data generation (SDG):** Facing and solving an older problem of data availability, InstructLab offers within its automatic capabilities a process to enhance, in cardinality and variability, a training dataset for LLM fine-tuning by using SDG, which is integrated into its core.
- ▶ **Cost and complexity reduction:** One of the most significant benefits of InstructLab is its ability to drastically reduce the cost and complexity of fine-tuning. By relying on SDG rather than manually labeled datasets, InstructLab eliminates one of the most time-consuming and expensive aspects of model customization.
- ▶ **Agile development:** The iterative feedback loops and the ability to work collaboratively with SMEs enable the fine-tuning process to be far more agile, with models refined and improved continuously based on real-time insights.

- **Flexible model selection and customization:** InstructLab is model-neutral, which means that users can select and fine-tune various LLMs based on their specific needs. Whether it is a general-purpose model like Granite or a more specialized compatible model that is found on Hugging Face for a particular domain, InstructLab enables users to adapt and improve the model that best fits their use case.
- **Broader audience participation:** The open-source nature and simplicity of the workflow in InstructLab make it possible for people without deep machine learning (ML) expertise to participate in model development and fine-tuning. This approach is a significant step toward democratizing AI and ensuring that more organizations, regardless of size or expertise, can harness the power of LLMs for their specific needs.

The InstructLab innovative approach to model fine-tuning (leveraging taxonomy-driven knowledge curation, SDG, and iterative training) marks a transformative shift in how LLMs can be customized and applied across a wide range of domains. By reducing the barriers to entry, lowering computational costs, and enabling highly specialized, domain-specific training, InstructLab is positioning itself as a key enabler of accessible, efficient, and scalable AI development. This open-source, community-driven initiative is paving the way for a new generation of AI practitioners to fine-tune and enhance LLMs without requiring extensive technical expertise, which expands the scope and impact of AI in real-world applications.

5.4.2 How to use InstructLab

InstructLab is a model-neutral, open-source AI project that facilitates contributions to LLMs. It is a new community-based approach to build truly open-source LLMs. InstructLab uses a synthetic-data-based alignment tuning method to train LLMs. The InstructLab tuning method is driven by manually created taxonomies. InstructLab provides a process for optimizing and tuning LLMs by collecting knowledge and skills as part of a taxonomy tree.

To start the InstructLab process, `ilab` must be installed. You can download it from its [official repository](#).

`ilab` is a CLI tool that you can use to perform the following actions:

- Download a pre-trained LLM.
- Chat with the LLM.
- Add new knowledge and skills to the pre-trained LLM by adding information to the companion taxonomy repository.

After you add knowledge and skills to the taxonomy, you can perform the following actions:

- Use `ilab` to generate new synthetic training data based on the changes in your local taxonomy repository.
- Retrain the LLM with the new training data.
- Chat with the retrained LLM to see the results.

Figure 5-10 shows the `ilab` flow of commands, which show how to start processing data for synthetic data generation and the fine-tuning process.

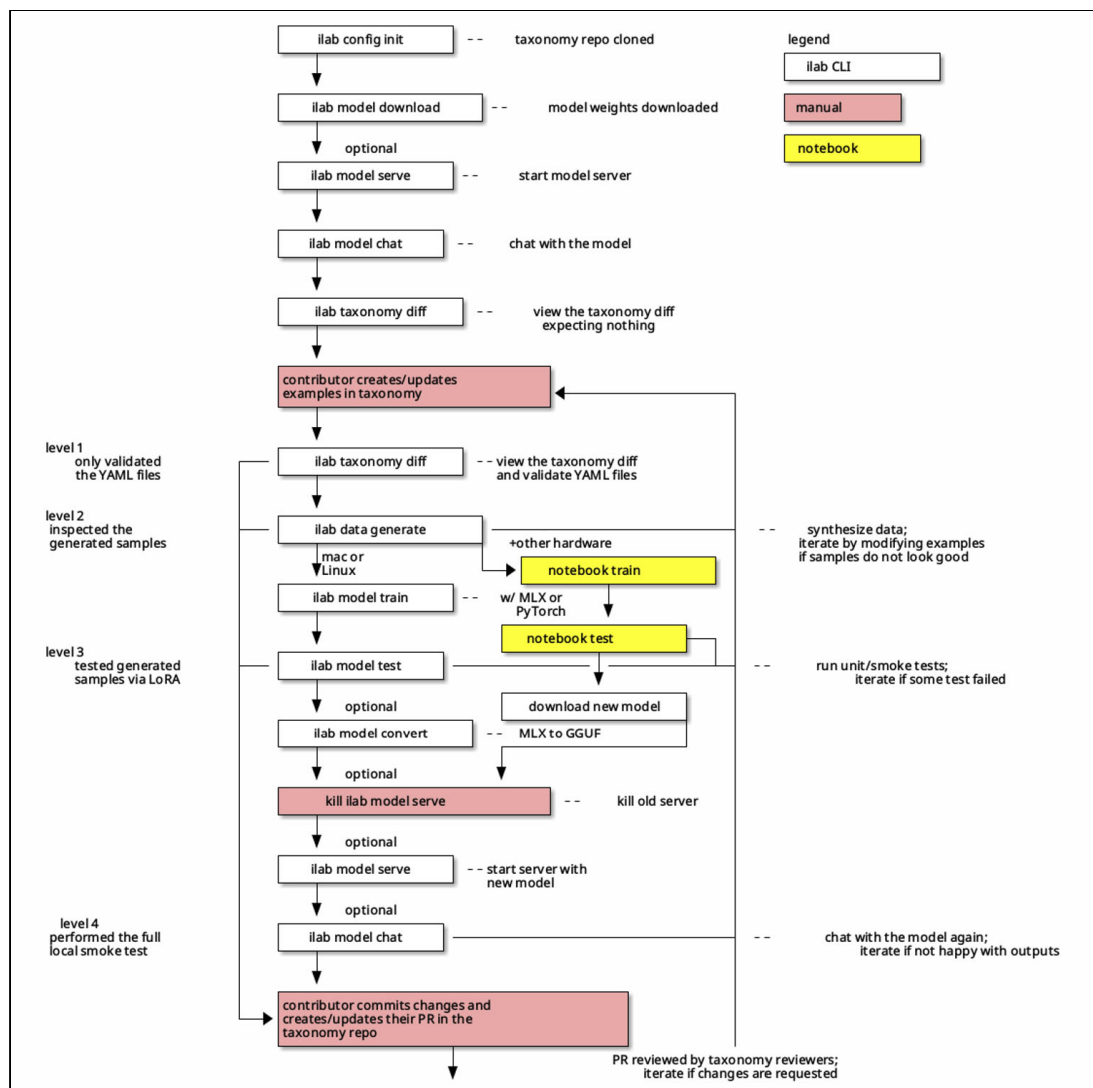


Figure 5-10 The `ilab` flow of commands

Before you begin working with `ilab`, ensure that your system meets the following requirements:

- ▶ Operating system: Linux (tested on Fedora), or macOS with Apple M1/M2/M3 chipsets.
- ▶ Disk space: Minimum of 250 GB. 500 GB is recommended for complete workflows.
- ▶ Python Version 3.10 or 3.11. At the time of writing, Python 3.12+ is unsupported due to dependency constraints.
- ▶ C++ compiler: Ensure that a modern GCC version or equivalent is installed.

If you use Python environment management tools, ensure that they build libraries that are implemented in C by including flags during Python compilation. For example, when using `pyenv`, you use the following string:

```
PYTHON_CONFIGURE_OPTS="--enable-framework" pyenv install 3.11.5
```

To install the required tools, run the following command:

```
sudo dnf install gcc gcc-c++ make git python3.11 python3.11-devel
```

ilab can be installed with various configurations, depending on your hardware and preferred accelerators. Different hardware setups often require specific steps to optimize performance and ensure compatibility with the chosen accelerators, such as Apple Metal, AMD ROCm, or NVIDIA CUDA.

If a GPU is available, you can leverage more processing power by using the following commands for initialization:

```
python3.11 -m venv --upgrade-deps venv
source venv/bin/activate
pip cache remove llama_cpp_python
CMAKE_ARGS="-DGGML_CUDA=on -DGGML_NATIVE=off" pip install 'instructlab[cuda]'
pip install vllm@git+https://github.com/openai/gpt-fast
```

After you install ilab, proceed with the first initialization by running the following command:

```
Ilab config init
```

During initialization, ilab prompts you to perform specific tasks that influence how the environment is configured. By following these prompts, you can tailor ilab to meet your needs by setting up essential components like the taxonomy repository, model paths, and training profiles. For example, selecting a training profile helps ensure compatibility with your hardware, whether it uses CPUs or GPUs, to provide the best performance and resource optimization for your setup:

1. Clone the taxonomy repository, either interactively or by specifying a path with the `--taxonomy-path` flag.
2. Specify the path to your model. By default, it uses a quantized [Granite model](#).
3. Select a training profile. For systems without dedicated GPUs, choose No Profile (CPU, Apple Metal, AMD ROCm).

After initialization, the directories that are shown in Table 5-1 are created.

Table 5-1 ilab directory overview and details

Directory	Description
<code>~/.cache/instructlab/models/</code>	Contains downloaded models.
<code>~/.local/share/instructlab/datasets/</code>	Stores the dataset outputs that are generated during workflows.
<code>~/.local/share/instructlab/taxonomy/</code>	Contains skill and knowledge data from the taxonomy repository.
<code>~/.local/share/instructlab/checkpoints/</code>	Contains model checkpoints from the training process.
<code>~/.config/instructlab/config.yaml</code>	The configuration file that is generated during initialization.

After you install the InstructLab CLI on your system, start by downloading the base model that you want to train. The foundation of using InstructLab effectively is access to its models. The `ilab` CLI simplifies this process by offering robust integration with repositories like Hugging Face or OCI. It provides authentication mechanisms, such as token-based access for Hugging Face, and features like repository specification and download acceleration. These capabilities help ensure secure and efficient downloads of pre-trained models.

To quickly get started, download compact pre-trained versions of the following models:

- ▶ `granite-7b-lab-GGUF`
- ▶ `merlinite-7b-lab-GGUF`
- ▶ `Mistral-7B-Instruct-v0.2-GGUF`

To initiate the download of a model, can run the following command:

```
ilab model download --repository <MODEL-ID>
```

When this command runs, the `ilab` CLI interacts with the designated repositories to fetch the selected models. By default, the models are stored locally in the `~/.cache/instructlab/models/` directory, which helps ensure efficient reuse because the downloaded models do not need to be fetched again for future operations unless explicitly removed or updated.

You can download a non-default LLM from Hugging Face. If a Hugging Face token is required, can add it by running `ilab model download --repository <MODEL-ID>`, but add the token after the argument `-hf-token`.

You can use OCI-compliant repositories. To do so, log in to the registry and use the following command:

```
ilab model download -rp docker://<MODEL_ID> -r1 latest
```

Once the models are downloaded, they can be served locally for inference. `ilab` supports serving both default and custom models if the system prerequisites are met. To serve models locally, ensure that the system has sufficient hardware resources, which include at least 8 GB of RAM and, for GPU-accelerated serving, an NVIDIA GPU with CUDA support. If multiple `ilab` clients attempt to connect to the same server, the first client connects successfully, and the others create temporary servers, which require more resources. To prevent conflicts, manage the connections.

To serve the model, run the following command, which provides a URL for API interaction:

```
Ilab model serve --model-path <MODEL_PATH>
```

It is possible to interact with a served model directly within `ilab` by using the following command, with optional personalization of inference parameters, such as temperature:

```
Ilab model chat --model <MODEL_PATH> [-temperature <VALUE>]
```

Now, you can start personalizing the model by adding new skills and knowledge.

To train an open source model with InstructLab, create knowledge and skills in the taxonomy directory. When you initialized the `ilab` CLI, it automatically cloned the [InstructLab taxonomy repository](#), which is the source of truth for your model training.

In the context of skill contributions, the required content is smaller in volume compared to knowledge contributions. A complete skill addition to the taxonomy tree can be represented by a few lines in a `qna.yaml` file (short for “questions and answers”) and an `attribution.txt` file to cite sources.

To make a valid skills contribution, the pull request must include a `qna.yaml` file with key-value entries that contain at least five question-and-answer pairs and an `attribution.txt` file that lists the sources that are used. The taxonomy structure serves multiple purposes: selecting the relevant subset for data generation, ensuring interpretability for contributors and maintainers, and forming part of the prompt for the LLM when generating synthetic samples.

Each `qna.yaml` file must adhere to a standard structure with specific keys:

- ▶ `version`: Must be set to 2 (required).
- ▶ `task_description`: A description of the skill (required).
- ▶ `created_by`: The GitHub username of the contributor (required).
- ▶ `seed_examples`: A collection of key-value entries with at least five examples (required for new files, although older files may contain fewer examples).
- ▶ `context`: Provides relevant information for grounded skills, which guide the model's processing (not used for ungrounded skills).
- ▶ `question`: The model's input query (required).
- ▶ `answer`: The expected response (required).

The taxonomy tree also categorizes skills as either grounded (requiring context) or ungrounded (not requiring context). For example, a grounded skill might be `grounded/linguistics/grammar`, while an ungrounded skill might be `linguistics/writing/poetry/haiku`. The `qna.yaml` file is always in the final node of the taxonomy path. Importantly, there is a limit on the content length in question-answer pairs to ensure model compatibility; contributions should not exceed approximately 2,300 words for these pairs. By adhering to these guidelines, contributors can maintain consistency and utility within the skill taxonomy framework.

To make the `qna.yaml` files faster for humans to read, it is best practice to specify `version` first, which is followed by `task_description`, then `created_by`, and finally `seed_examples`. In `seed_examples`, it is a best practice to specify `context` first (if applicable), followed by `question` and `answer`.

Example 5-1 shows an example of a `qna.yaml` file.

Example 5-1 A `qna.yaml` file

```
version: 2
task_description: <string>
created_by: <string>
seed_examples:
  - question: <string>
    answer: |
      <multi-line string>
  - context: |
      <multi-line string>
    question: <string>
    answer: |
      <multi-line string>
  ...
```

Create an `attribution.txt` file that includes the sources of your information, which can be self-authored sources.

Knowledge contributions differ from skills by focusing on answering factual, data-driven, or reference-based questions, which are often supported by documents like textbooks, technical manuals, encyclopedias, journals, or magazines. Although knowledge and skills share similarities in their taxonomy structures, knowledge nodes include additional elements to accommodate their document-based nature.

For contributors that use InstructLab 0.21.0 or later, knowledge contributions can include PDF files as valid document types, but earlier versions accept only markdown formats. Each knowledge node in the taxonomy tree contains a `qna.yaml` file that is similar in structure to the one that is used for skills, but with additional fields to support knowledge-specific attributes. Notably, all knowledge submissions must be in a Git repository, such as one hosted on GitHub, and the `qna.yaml` file must reference this repository:

- ▶ Submit the most current version of the document.
- ▶ Contributions must be text-based. Images are ignored.
- ▶ Avoid using tables in your markdown freeform contributions.

The `qna.yaml` file for knowledge contributions must follow a specific format and include the following fields:

- ▶ `version`: The version of the `qna.yaml` file format, which is set to 3.
- ▶ `created_by`: The GitHub username of the contributor.
- ▶ `domain`: The category of the knowledge.
- ▶ `seed_examples`: A collection of key-value entries.
- ▶ `context`: A chunk of information from the knowledge document. Each `qna.yaml` file must include at least five context blocks, with a maximum of 500 words per block.
- ▶ `questions_and_answers`: Holds the questions and answers based on the context. Each context block requires a minimum of three question-and-answer pairs, each with a maximum word count of 250 words.
 - `question`: A question for the model.
 - `answer`: The corresponding answer.
- ▶ `document_outline`: An overview of the document being submitted.
- ▶ `document`: The source document for the knowledge contribution.
- ▶ `repo`: The URL of the repository that contains the knowledge files.
- ▶ `commit`: The SHA of the commit in the repository for the knowledge files.
- ▶ `patterns`: A list of glob patterns specifying the files in the repository. Patterns starting with `*`, such as `*.md`, must be quoted ("`*.md`") to comply with YAML rules.

By adhering to these guidelines, knowledge contributions maintain a structured, accessible format that aligns with the taxonomy framework and supports efficient integration into the system.

When working with YAML files (for both skills and knowledge), it is crucial to adhere to specific formatting rules to help ensure correctness and avoid parser errors. Indentation and spacing play a role, and YAML requires *two spaces* for each level of indentation (*tabs must not be used* under any circumstances). Also, avoid trailing spaces at the end of lines because they can lead to issues during processing. For entries in `seed_examples`, each example begins with a `-` placed before the first field, such as `question` or `context`. Subsequent keys within the same example should not include the `-`. Pay attention to special characters like `"` and `'`, which must be escaped by using a backslash (`\`). To simplify handling these characters, YAML enables the use of the `|` character at the start of a value, which disables special character interpretation and supports multi-line strings. For example, lines that start with `|` are followed by an indented block that contains the string's content. To avoid unexpected YAML parser behavior, it is a best practice to *quote all values* by using double quotation marks (`"`). This approach prevents values such as `Yes` or `No` from being interpreted as Boolean types (`True` or `False`). For more information about managing multi-line strings and YAML nuances, see the `yaml-multiline.info` file.

Now, after creating a YAML file for skills and knowledge, as shown in Figure 5-7 on page 68, you can validate your new data. Use the `ilab taxonomy diff` command to help ensure that `ilab` is registering your new knowledge or skills and that your contributions are properly formatted. This command displays any new or modified YAML files within your taxonomy tree. You can also validate your entire taxonomy by performing a diff against an empty base by using the `--taxonomy-base=empty` argument.

After validation, it is possible to start the Synthetic Data Generation (SDG) pipeline. To generate a synthetic dataset based on newly added knowledge or skill sets in the taxonomy repository, run the `ilab data generate` command. Before proceeding, ensure the existing model to which you are adding skills or knowledge is still running. Alternatively, you can initiate the server by using the `ilab data generate` command by specifying a fully qualified model path with the `--model` flag. At the time of writing, the full CLI pipeline supports only Mixtral and Mistral Instruct Family models as the teacher model. For the simple pipeline, Merlinit 7b Lab is the only supported teacher model due to the specific model prompt templates that it uses. There is a plan to expand compatibility in the future, and on `watsonx.ai` (as described in Chapter 6, “Artificial intelligence agents” on page 87).

To start generation, run the following command:

```
ilab data generate [--pipeline full --gpu <NUM_OF_GPUS> --model <MODEL_PATH>
```

Optionally, you can start SDG by using GPUs when they are available. You can specify the teacher model that is used (the default one for the `ilab` CLI is Merlinit-7B).

After generation finished, the synthetic dataset consists of two files in the `~/local/share/instructlab/datasets` directory:

- ▶ `skills_train_msgs_*.jsonl`
- ▶ `knowledge_train_msgs_*.jsonl`

You can run the generate step against a different model through a compatible API, such as the one that is created by the `ilab model serve` or any remote or locally hosted LLM (through `ollama`, `LM Studio`, or others). Run the following command:

```
ilab data generate --endpoint-url http://localhost:8000/v1
```

Now that the curated dataset for a fine-tuning is ready, the fine-tuning process can be started.

The InstructLab model train has three pipelines: `simple`, `full`, and `accelerated`. The default is `full`.

- ▶ `simple` uses an SFTTrainer on Linux and MLX on MacOS. This type of training takes roughly an hour and produces the lowest fidelity model but should indicate whether your data is being picked up by the training process.
- ▶ `full` uses a custom training loop and data processing functions for the Granite family of models. This loop is optimized for CPU and MPS function. Use `--pipeline=full` with `--device=cpu` (Linux) or `--device=mps` (MacOS). You can also use `--device=cpu` on a MacOS machine. However, MPS is optimized for better performance on these systems.
- ▶ `accelerated` uses the `instructlab-training` library, which supports GPU-accelerated and distributed training. The full loop and data processing functions are either pulled directly from or based on of the work in this library.

To limit training time, you can adjust the `num_epoch` parameter in the `config.yaml` file. The maximum number of epochs for running the InstructLab end-to-end workflow is 10.

The following command shows how to start the automatic fine-tuning process with the previously generated dataset. Furthermore, it can specify more than the pipeline, such as the device that you want the model to be trained on (CPU, MPS, or GPU).

```
ilab model train [--pipeline <PIPE_ID> --device <DEVICE_ID> --data-path  
<DATA_PATH>]
```

This training step can potentially take from several minutes to several hours to complete, which depends on the available computing resources.

After the fine-tuning pipeline completes, it is possible to verify the quality of the new model and whether the generated dataset with the defined skills and knowledge produced good results. To thoroughly test and evaluate a newly trained model by using InstructLab, first run a series of commands that are designed to assess the performance and accuracy of the model after training. The testing process involves using the `ilab model test` command to obtain output from the model before and after the training process. With this output, you can see how well the model performs based on its previous state and after it has undergone the enhancements from the training process. The results from this test show the effectiveness of your training and provide insight into areas where further improvement might be needed.

When the model is tested, you can use the `ilab model evaluate` command to run the model through a set of predefined benchmarks to evaluate its performance across various categories. At the time of writing, there are four primary benchmarks that are supported by InstructLab:

- ▶ Multitask Language Understanding (MMLU)
- ▶ MMLUBranch
- ▶ MTBench
- ▶ MTBenchBranch.

These benchmarks assess different aspects of a model's capabilities, such as its knowledge and skills:

- ▶ MMLU evaluates the model's general knowledge on a wide range of topics.
- ▶ MMLUBranch compares the model's performance to the performance of a base model to identify improvements in knowledge.
- ▶ MTBench evaluates how well the model applies its knowledge in multi-turn conversations.
- ▶ MTBenchBranch assesses improvements or regressions in specific skill areas when compared to a base model.

For each benchmark, the evaluation generates detailed reports, which show scores and identify areas where the model performs well and areas that need further work. For example, the MMLU report provides a score for various subjects, such as abstract algebra, anatomy, and business ethics, which indicate how the model performs in each area. A typical output for MMLU looks like a series of subject categories with a score 0.0 - 1.0, with higher scores indicating better performance in the respective topics.

Running MMLUBranch involves evaluating your model's contributions compared to a base model. The evaluation outputs a score for both the base model and the newly trained model, along with a report on the improvements or regressions that are observed. For example, you might see that the model improved in one area, like "tonsils," from a score of 0.74 to 0.78, which indicates that your training enhanced the model's ability in that particular knowledge domain.

MTBench and MTBenchBranch follow a similar structure, but they focus on testing the model's skills rather than just knowledge. MTBench evaluates the model's ability to perform in multi-turn dialogs, which provide a score for each turn in the conversation, such as turn one and turn two. MTBenchBranch compares your model's skill performance to a base model, which provides a detailed breakdown of areas where your model improved or regressed, and highlighting any skills that showed no significant change. This detailed feedback helps pinpoint specific areas where more fine-tuning might be necessary to enhance the model's abilities. For each benchmark, it is important to help ensure that the model that is evaluated is in a supported format, either safetensors or GGUF. Using models directly from Hugging Face without downloading them is not supported.

Also, while using models for MMLU and MMLUBranch evaluations, GGUF models are not supported at the time of writing. When running MTBench and MTBenchBranch, it is a best practice to use the *Prometheus-8x7b-v2.0* model as the judge model, but you can use a different judge model. You can download the Prometheus model by running the `ilab model download` command for local use in these evaluations.

The entire process of running these evaluations can take from several minutes to several hours, which depend on the size of the model and the dataset that is used. Be prepared to allocate enough time for the evaluations to complete, especially when working with large datasets or multiple training epochs. The results from these evaluations provide a comprehensive view of the model's strengths and weaknesses, which offer valuable insights to guide further refinement and optimization of the model.

After the process ends and the results are good enough for the specified use case, a new fine-tuned model with new synthetic data in GGUF format is available in the `ilab` specified folder location in a GGUF format. Apart from the usage on `ilab`, it is possible to deploy it on hyperscalers such as `watsonx.ai` run time by using the Bring Your Own Model (BYOM) function, which fully enables a true, at-scale, enterprise-level fine-tuning process of FMs.

5.4.3 InstructLab on watsonx.ai Software-as-a-Service

At the time of writing, InstructLab has demonstrated its usability primarily through the `ilab` CLI. This method enables users to interact with InstructLab features and functions in a straightforward and efficient manner. However, the development team is working on integrating InstructLab with `watsonx.ai`. This upcoming integration will enhance the user experience by providing a dedicated user interface (UI). This UI will streamline the entire process, which will make it more accessible and intuitive for users who might not be comfortable with CLI operations. This development is expected to open new possibilities for a broader range of user by facilitating simpler access to powerful InstructLab tools and features.

Figure 5-11 shows the interface for tuning a generative LLM by using InstructLab. This interface is designed to provide users with an intuitive platform for refining language models to meet specific needs and requirements. It also illustrates the starting point of the overall process: the addition of skills and knowledge.

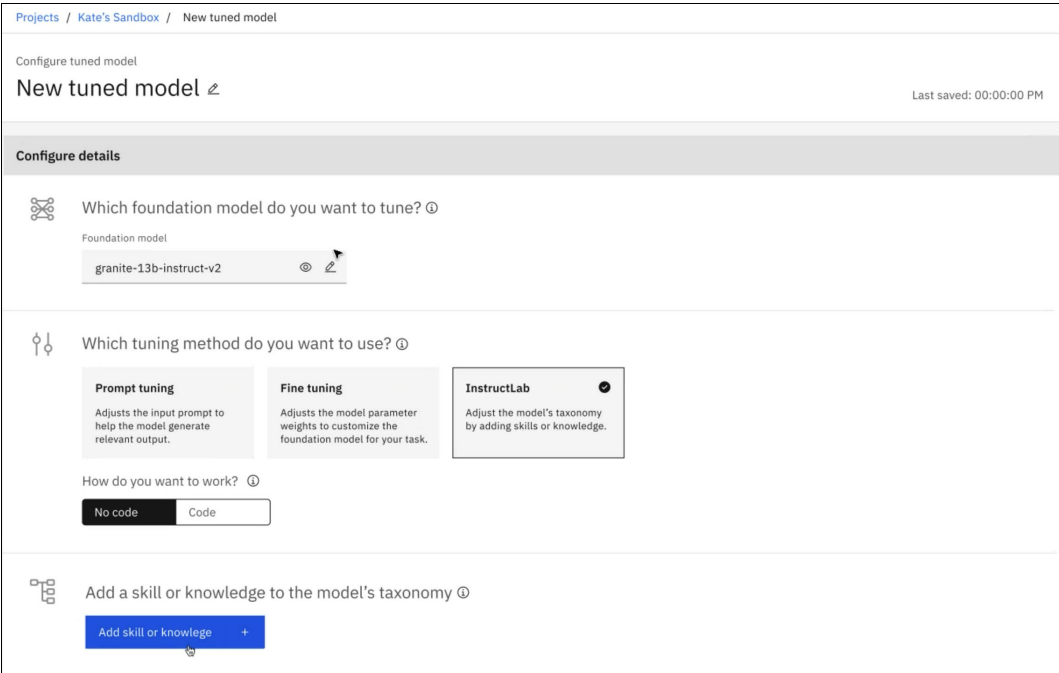


Figure 5-11 Tuning Studio interface

Users begin by selecting the FM, and then proceed to integrate various skills and knowledge areas into the model. The interface guides users through detailed steps, which include specifying configurations. Throughout this process, users can monitor their progress and make necessary adjustments to ensure that the model's performance aligns with outcomes.

Moreover, InstructLab offers advanced features such as feedback, performance metrics, and troubleshooting tools to enhance the tuning experience. This comprehensive approach helps create an accurate and efficient LLM, with continuous improvement and adaptation to evolving linguistic patterns and user needs.

Figure 5-12 on page 81 illustrates a detailed taxonomy tree of skill and knowledge files that are managed within Tuning Studio on watsonx.ai for InstructLab. The UI shows a comprehensive and organized view of the hierarchical structure of skills and knowledge areas. Each node in the taxonomy tree represents a distinct category, which facilitates navigation and access to specific training data.

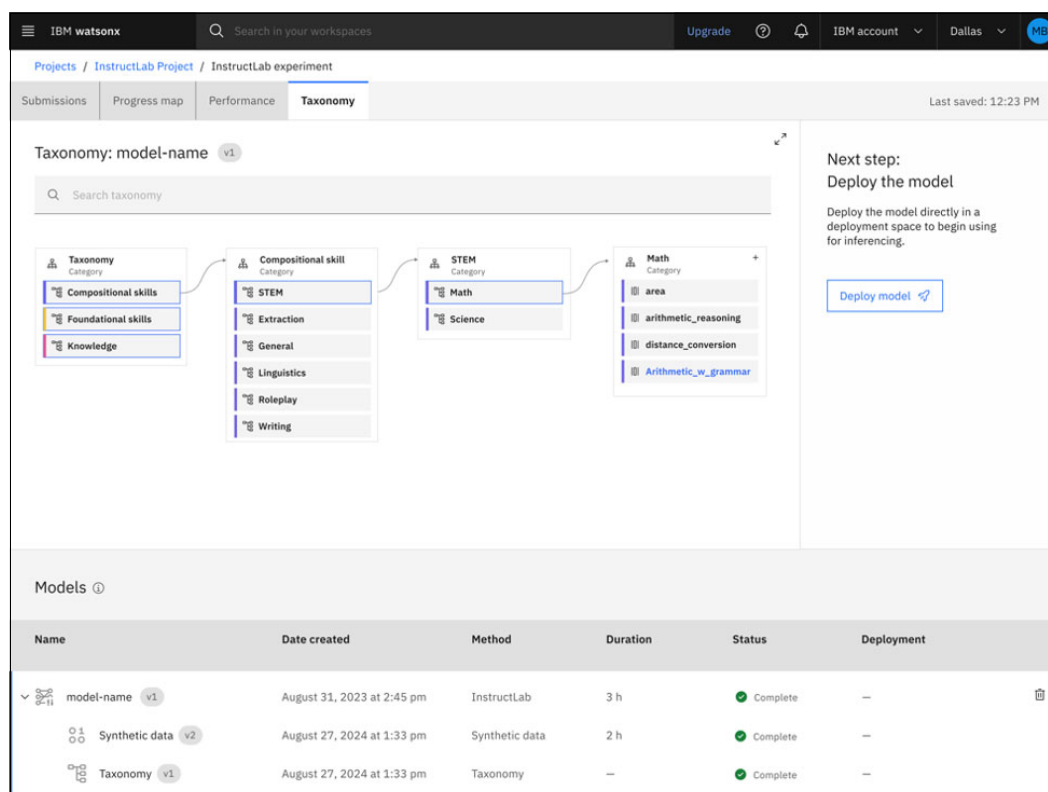


Figure 5-12 Taxonomy tree of skill and knowledge files that are managed in Tuning Studio on watsonx.ai for InstructLab

One of the key functions of this UI is its ability to visualize and maintain version history for each training run, which includes the trained models, the associated skills and knowledge taxonomy, grounding data, and the generated synthetic data. By tracking the evolution of these elements over time, users can effectively monitor the progression and improvements that are made with each iteration. This version control mechanism is essential for helping ensure the reproducibility and reliability of model training processes. Also, it will be possible to add knowledge and ingest multiple data formats, such as PDFs, docx, HTML, MD, and more.

The Tuning Studio capability to handle such a vast array of data types and their versions enables meticulous fine-tuning and enhances the overall model development lifecycle. Through this meticulous documentation and management, users can draw insights from past training runs, identify best practices, and apply learned lessons to future projects.

Figure 5-13 shows the progress map during the tuning phase of InstructLab on watsonx.ai.

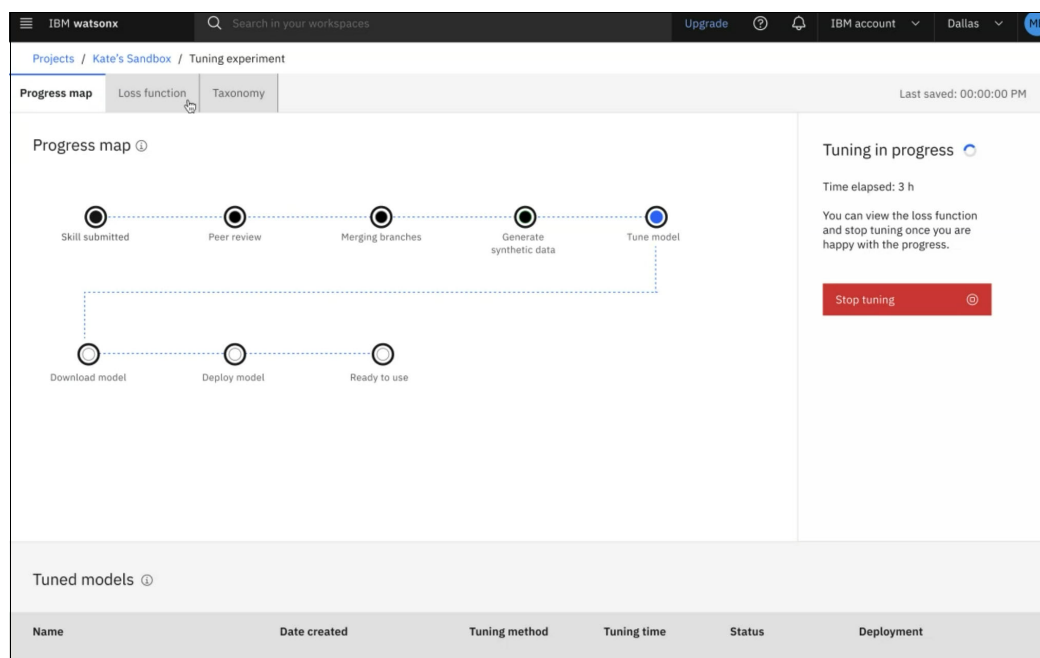


Figure 5-13 Progress map during the tuning phase of InstructLab on watsonx.ai

Before commencing the SDG and fine-tuning process, conduct a peer review of the submitted skills and knowledge files. This review involves project members and SMEs to help ensure comprehensive assessment and validation. If any modifications are identified as necessary during this review phase, new branches can be created within the version control system to incorporate these changes without disrupting the main development line.

After you incorporate the feedback and necessary adjustments, the fine-tuning process commences. This step involves leveraging pre-trained models and adjusting them to specific requirements by training on customized datasets. The goal is to enhance the model's performance in targeted areas to help ensure that it meets the specifications and accuracy levels.

On successful fine-tuning, the newly optimized model can be exported in the GGUF format, which is a versatile and widely supported format for deploying quantized FMs. Alternatively, the model can be directly deployed onto the watsonx.ai run time environment. This deployment establishes a seamless large language model operations (LLMOps) pipeline within watsonx.ai for InstructLab, which enables automated monitoring, maintenance, and iterative improvement of the model.

By integrating these processes within the watsonx.ai ecosystem, you help ensure continuous delivery and operational efficiency of AI solutions that align with best practices in modern ML workflows.

The fine-tuning process that is shown in Figure 5-14 on page 83 shows the intricate and methodical approach that is adopted by InstructLab within the watsonx.ai framework. This process is characterized by a 2-phase, fine-tuning methodology that is augmented with a replay buffer, which helps ensure enhanced performance and accuracy of the models.

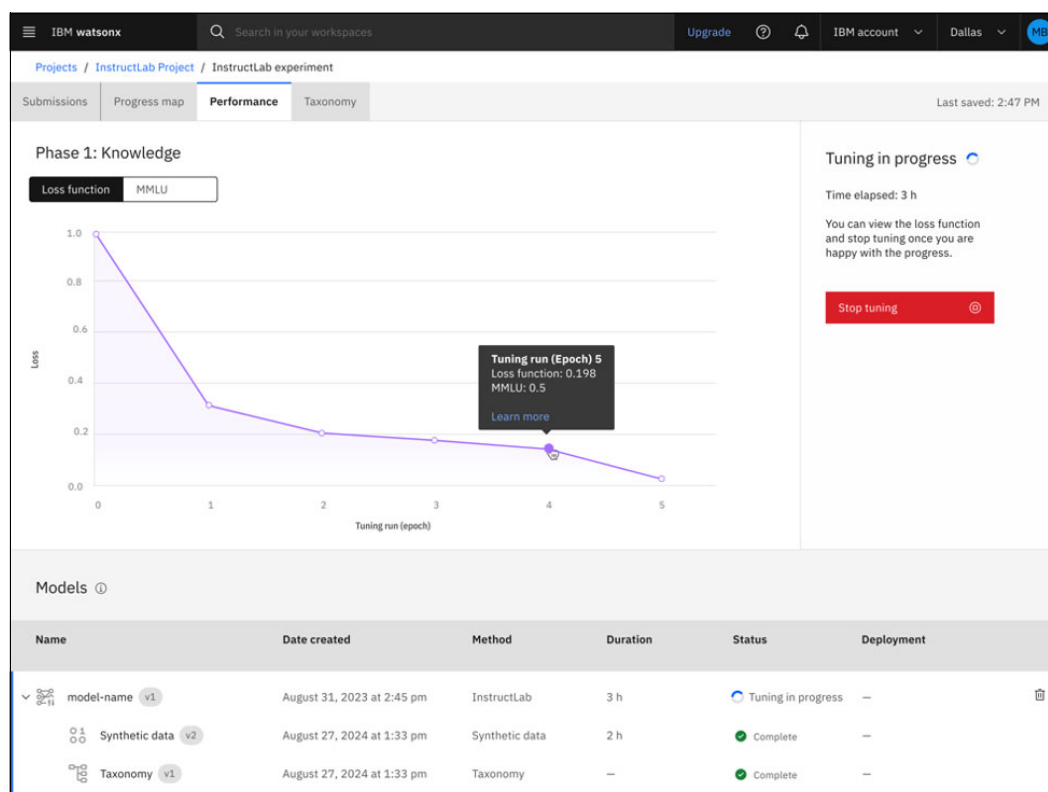


Figure 5-14 InstructLab fine-tuning process with real-time performance on the 2-phase fine-tuning process with replay buffer

In the initial phase, the pre-trained models undergo a rigorous training regimen by using targeted datasets that are pertinent to the specific knowledge areas that are identified within the taxonomy tree. This phase focuses on adapting the general-purpose models to the specialized requirements of the InstructLab projects, which hone their performance to meet the specifications. The second phase uses the skills and continues catastrophic forgetting by using a dedicated replay buffer while fine-tuning the skills in the second phase.

The replay buffer mechanism plays a crucial role in this fine-tuning process. By systematically storing and replaying past experiences (training data and model states) during the training sessions, the buffer helps ensure that the models continuously learn and adapt from the previous run. This approach mitigates catastrophic forgetting and reinforces learning from high-value data points, which improves the models' robustness and generalization capabilities. Figure 5-14 encapsulates a holistic and dynamic approach to model fine-tuning within the watsonx.ai ecosystem, which highlights the interplay of advanced methodologies and real-time performance monitoring to achieve superior gen AI solutions.

InstructLab will be available on watsonx.ai as Software-as-a-Service (SaaS), and it will be an important addition for gen AI on an enterprise level. It will enable a pool of resources to fine-tune and improve generative LLMs.

5.4.4 InstructLab use case examples

Introducing InstructLab on watsonx.ai heralds a new era of advancements in gen AI, particularly in the realm of enterprise-level applications. This innovative platform integrates seamlessly within the watsonx.ai ecosystem, which enables a dynamic and robust environment for the fine-tuning and deployment of LLMs. As an SaaS offering, InstructLab provides an unparalleled suite of tools and methodologies to optimize model performance through a meticulously structured 2-phase, fine-tuning process.

The InstructLab capabilities are impactful across many use cases, and they empower organizations to tailor AI solutions to their unique requirements. By leveraging the advanced features of InstructLab, users can achieve superior accuracy, efficiency, and scalability in their AI-driven initiatives.

Here are some of the first prominent examples of use cases where InstructLab has demonstrated significant improvements in its early life:

- ▶ **Emergency medical services use case:** A large hospital is looking to automate the processing of emergency medical records to improve the efficiency and accuracy of critical tasks. The system should be capable of completing the following tasks:
 - **Case classification:** Assigning priority levels such as green (low urgency), orange (medium urgency), or red (high urgency) flags to cases based on their severity.
 - **Peer medical review recommendations:** Identifying cases that might require further review by a medical peer to help ensure proper oversight and decision-making.
 - **Clinical compliance and guideline deviation:** Detecting discrepancies in medical reports compared to established clinical guidelines, which enhance compliance and quality assurance.
 - **Knowledge:** The system will rely on hospital compliance data, which includes clinical guidelines and historical patient records to perform its tasks.
 - **Skill requirements:**
 - Case classification to prioritize emergency scenarios.
 - Identifying deviations from clinical guidelines to help ensure regulatory compliance.
- ▶ **Call transcript processing use case:** A large North American telecommunications company requires an automated system to process and summarize incoming customer support call transcripts. The system must extract and organize information based on a predefined set of 80 questions. Examples include “Did the customer want to upgrade their plan?” or “Did the customer report bandwidth issues?”
 - **Knowledge:** The primary data source will be call transcripts, which contain varied human expressions and conversational styles.
 - **Skill requirements:** Interpreting and analyzing natural language, which includes understanding diverse writing styles and linguistic nuances to extract relevant insights from conversations.

- ▶ Personalized retail recommendations use case: A retailer aims to deploy a personalized recommendation engine that suggests in-stock products that are tailored to a user's dietary preferences, which include offering recommendations based on food allergies, nutritional goals, or ingredient restrictions.
 - Knowledge: The engine must use detailed product nutritional information and inventory data to ensure accurate and timely recommendations.
 - Skill requirements:
 - Classification of ingredients and their alignment with dietary preferences.
 - Recommending products that match user profiles while considering inventory availability.
 - Agent capabilities:
 - Understanding and analyzing inventory data in real time.
 - Interpreting customer preferences and purchase history to generate meaningful suggestions.
- ▶ Intelligent auto claims processing use case: An insurance provider requires a solution to analyze images of auto accidents and suggest insurance coverage recommendations that are based on the claimant's active policy. The system should improve the speed and accuracy of claims processing.
 - Knowledge: The system needs access to policy details, which include terms, coverage limits, and exclusions.
 - Skill requirements:
 - Classification of accident severity by analyzing damage in submitted images.
 - Matching severity with appropriate coverage based on the policy.
 - Agent capabilities:
 - Accessing and analyzing driver history to help ensure an accurate policy application.
 - Interpreting active insurance policies to provide recommendations that are aligned with coverage terms.



Artificial intelligence agents

Artificial intelligence (AI) agents will soon become pivotal in modern computing by serving as autonomous entities that can perceive their environment, reason about their goals, and perform actions to achieve wanted outcomes. These agents will be central to many AI systems, from personal assistants and autonomous vehicles to advanced simulations and decision-making tools. The versatility and capability of AI agents arise from their ability to operate independently while adapting to dynamic and often unpredictable environments. They combine advanced algorithms with AI and generative AI (gen AI) models, which enable them to make intelligent decisions, adapt to changes, and optimize outcomes.

This chapter delves into the intricacies of AI agents by exploring their fundamental characteristics, the motivation behind their development, and their applications across various domains. Through a detailed exploration, readers will understand why AI agents represent a paradigm shift in how computational systems interact with and influence their surroundings.

The following topics are described in this chapter:

- ▶ 6.1, “What makes an AI agent” on page 88
- ▶ 6.2, “Why AI agents are needed” on page 94
- ▶ 6.3, “Multiple AI agents” on page 95
- ▶ 6.4, “AI agents on watsonx.ai” on page 100
- ▶ 6.5, “AI agents use case examples” on page 107

6.1 What makes an AI agent

An *AI agent* can be formally defined as a computational entity that is equipped with sensors to perceive its environment and use its actuators to interact with it. The core of an agent lies in its ability to reason and decide, which is driven by algorithms that enable it to analyze inputs, predict outcomes, and choose actions that are aligned with specific objectives. Unlike traditional software systems, which operate on predefined instructions, AI agents exhibit autonomy and flexibility with dynamic execution flows, which enable them to handle complex scenarios with minimal human intervention. This autonomy stems from their design principles, which often draw from cognitive sciences, game theory, and control systems, enabling agents to emulate human-like decision-making processes. Furthermore, each AI agent can be designed for different levels of complexity and interaction. This adaptability makes AI agents indispensable in fields such as robotics and natural language processing (NLP), where complex interactions with dynamic environments are required.

Figure 6-1 shows a schematic view of an AI agent.

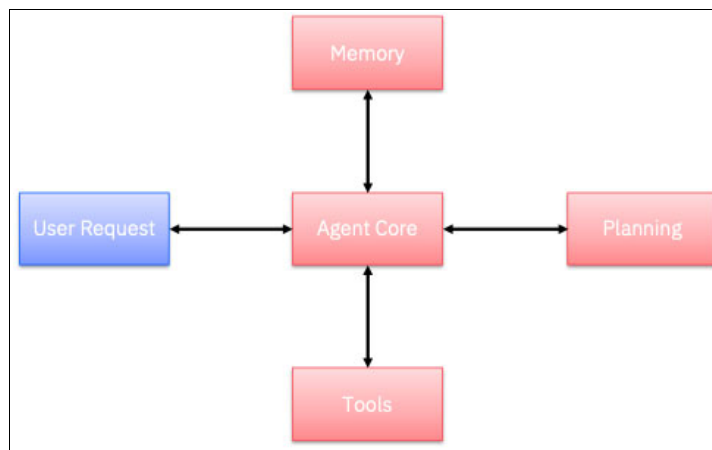


Figure 6-1 Schematic view of an AI agent

Agents, in the context of AI, can be conceptualized as sophisticated orchestrators of intelligence that can address intricate and multi-faceted problems. They achieve these goals by leveraging the reasoning capabilities of large language models (LLMs), formulating detailed plans to resolve challenges, and running these plans by using a diverse set of tools. An agent operates as a cohesive system, and its architecture can typically be broken down into four key modules:

- ▶ The agent core
- ▶ Search and memory modules
- ▶ Planning modules
- ▶ Tools

Each of these components plays a critical role in ensuring the agent's functions, adaptability, and effectiveness.

The *agent core* stands as the central coordination hub of the agent. It is often described as the “decision-making nucleus” due to its pivotal role in governing the agent's logic, behavior, and overall strategy. This module is responsible for synthesizing inputs, determining appropriate actions, and managing the interactions between other components.

To design a robust and efficient agent core, you must define several foundation aspects that serve as the blueprint for the agent's behavior:

- ▶ The general goals of the agent must be established. These goals act as the guiding principles that dictate the agent's actions and responses, which help ensure that its operations align with the overarching objectives that it is designed to achieve. Without clear goals, the agent risks becoming unfocused or inefficient.
- ▶ Another critical aspect of the agent core is the explicit definition of the tools that are available for execution, which involves creating a comprehensive "user manual" that counts and describes all tools at the agent's disposal. Each tool's capabilities, limitations, and specific use cases should be outlined to enable the agent to allocate resources effectively and run tasks with precision.
- ▶ Furthermore, the agent core must provide detailed guidance about the utilization of planning modules. These modules are instrumental in enabling the agent to adapt dynamically to varying scenarios by selecting the most suitable planning strategy based on the context. This adaptability is key to ensuring the agent's effectiveness in complex and unpredictable environments.

Memory integration is another cornerstone of the agent core. The memory system is designed to maintain and use relevant information from prior interactions or external research, which enhances the agent's ability to generate accurate and context-aware responses. This integration requires dynamic management of memory items to help ensure that only the most pertinent data is referenced during inference. Also, an optional persona definition can be incorporated into the agent core to influence the agent's tone, preferences, and behavioral nuances. By defining a persona, the agent can be tailored to exhibit specific characteristics that align with its intended use case or audience, adding a layer of uniqueness to its interactions. *Memory modules* are integral to the agent's ability to maintain contextual awareness and continuity. These modules are responsible for storing and managing information that supports the agent's operations. Memory can be categorized into two main types:

- ▶ **Short-term memory**

Short-term memory focuses on capturing the agent's immediate actions, thoughts, and observations during ongoing interactions, which include data that is retrieved from vector searches, outputs from API calls, and results from database queries. Short-term memory is essential for helping ensure that the agent can respond effectively to the immediate context of a user's query.

- ▶ **Long-term memory**

In contrast, long-term memory serves as a repository for information that is accumulated over extended periods, which include summarized logs of prior interactions, personal details and preferences of the user, and other contextual information that might influence the agent's behavior. Long-term memory enables the agent to maintain a consistent and personalized approach in its interactions, which enhance user satisfaction and engagement. For example, in a conversational agent, long-term memory enables the retention of user preferences and past conversations, which create a more seamless and intuitive experience.

When solving intricate problems, agents that are powered by LLMs are adept at navigating complexity by employing a combination of advanced methodologies that resembles planning an execution flow. One such methodology is *task and question decomposition*. This approach involves breaking down compound queries into smaller, more manageable sub-questions that can be addressed sequentially.

For example, to answer the question, “Will the temperature tomorrow be higher or lower than the historical average?” the agent must decompose it into subquestions such as identifying the location, determining the forecasted temperature for the specified location, and retrieving the historical average temperature for comparison. By addressing each sub-question individually, the agent can construct a comprehensive and accurate response.

Another essential technique is the usage of *reflection and critique frameworks*. These methodologies, which include well-established prompting strategies such as *ReAct*, *Reflexion*, *Chain of Thought*, and *Graph of Thought*, are designed to enhance the reasoning capabilities of the agent. By incorporating elements of evidence-based reasoning and iterative self-critique, these frameworks enable the agent to refine its execution plans and improve the quality of its responses. Reflection techniques enable the agent to evaluate the plausibility and coherence of its answers, which help ensure that its outputs meet high standards of reliability and relevance.

Classification and the implementation of guardrails are extra mechanisms that enhance the agent’s decision-making process. By classifying questions and queries, the agent can filter search results, identify relevant sub-agents, or even deny a response if necessary. Guardrails, which serve as a specialized form of classification, act as safeguards to help ensure that the agent’s outputs adhere to predefined ethical and operational guidelines. These mechanisms are valuable in scenarios where precision, safety, and compliance are paramount.

The *tools* that are employed by an agent are another critical aspect of its function. These tools are executable workflows that enable the agent to perform specific tasks. Analogous to specialized third-party APIs, tools provide the agent with targeted capabilities that extend its problem-solving abilities. Examples of tools include Retrieval-Augmented Generation (RAG) pipelines for generating context-aware answers, code interpreters for handling complex programming tasks, and APIs for retrieving information from the internet. Also, utility APIs such as weather services or instant messaging platforms can be integrated to address domain-specific needs. The versatility and effectiveness of an agent are enhanced by its ability to leverage these specialized tools.

Going one step forward, agents that rely solely on text inputs face inherent limitations in their ability to interact with and analyze diverse data formats. *Multi-modal agents* address this limitation by incorporating the ability to process and reason over various input types, which include images, audio files, and structured datasets. This capability expands the range of applications for agents, which enable them to tackle tasks that require visual analysis, speech processing, or combined reasoning across multiple modalities. For example, a multi-modal agent can analyze an image to extract relevant features, process an accompanying audio file for contextual information, and synthesize this data with text-based inputs to deliver a comprehensive solution.

By following the main core modules of an AI agent and through thoughtful design and continuous refinement, agents are poised to become indispensable tools in the ever-expanding landscape of AI.

An AI agent, as illustrated in Figure 6-2 on page 91, consists of a centralized planning system that is supported by a general-purpose generative LLM. This LLM serves as the core orchestrator, which can devise a structured plan of actions to address user queries. Its decision-making process is enhanced by a memory module, which maintains contextual information and is continuously updated based on past actions and outcomes, which help ensure adaptive and dynamic responses.

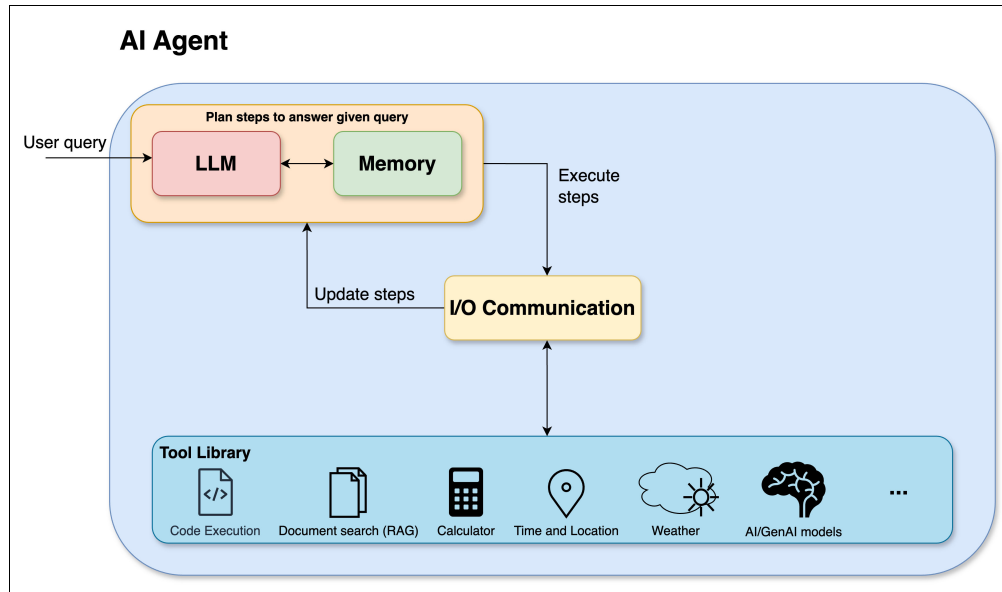


Figure 6-2 High-level view at the core of an AI agent

The planning system interacts with an I/O communication layer to run the planned actions effectively. This layer acts as a bridge between the LLM and an extensive Tool Library, which enables the agent to perform specialized tasks beyond its intrinsic capabilities. The tools include functions such as code execution, document retrieval (RAG), calculations, weather queries, time and location services, and AI and gen AI models, among others. The module of communication is a crucial part of an agent because it sets the formatting of the I/O communication in a standardized way to enable fast and transparent invocation of tools and output comprehension.

The LLM at the core of this system is typically a large foundation model (FM) that is optimized for high performance across diverse benchmarks and tasks. For example, in the context of the IBM watsonx.ai platform, this role can be fulfilled by models such as Mistral Large, Llama 3.3 70B, or Llama 3.1 405B, which are known for their robust capabilities in generative reasoning and planning.

Behind the scenes, an agent has its own called system prompt, which is the usual system prompt that can be set for any generative LLM. It describes in detail the main task for which a generative LLM should adhere to. The scheme behind the main prompt solution is shown in Figure 6-3.

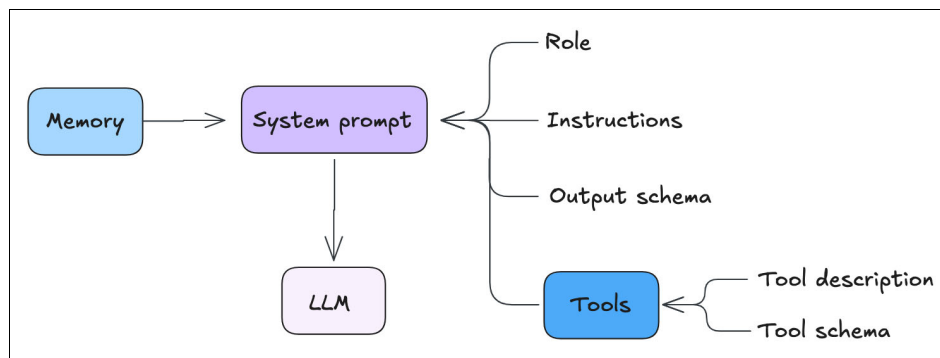


Figure 6-3 Logical prompt schema definition for agents

This system prompt is a structured set of instructions that encapsulates the core task, and defines the role of the LLM, the expected output schema, and the instructions that the model should follow.

As shown in Figure 6-3 on page 91, the system prompt integrates multiple critical elements:

- ▶ **Memory:** Provides contextual data or historical interactions that the LLM can leverage to maintain continuity and relevance in responses. This approach helps ensure that the agent adapts dynamically to ongoing tasks or user needs.
- ▶ **Role definition:** Specifies the persona or role that the agent assumes (for example, a customer support assistant, a data scientist, or a creative writer), and aligns the behavior and tone of the LLM with the intended use case.

Example 6-1 shows an example of a role prompt definition:

Example 6-1 Defining a role for the agent

#Role

As a helpful assistant, your role is to provide users with actionable insights from their data files.

- ▶ **Instructions:** Detailed guidelines about how the LLM should process inputs, interpret user queries, and generate outputs. These instructions help ensure that the model stays on task and adheres to the wanted operational framework.

Example 6-2 show an example of the instruction prompt definition.

Example 6-2 Outline instructions for the agent

Instructions

The user can see only the Final Answer. All answers must be provided there. Functions must be used to retrieve factual or historical information to answer the message. If the user suggests using a function that is not available, answer that the function is not available.

- ▶ **Output schema:** A predefined structure that dictates how the LLM should format its responses, which may include JSON formats, bullet points, or other data representations that are required by downstream tools or workflows. This prompt is important for the overall communication and interaction in the multiple steps in an agentic execution flow.

Example 6-3 shows an example of the output schema prompt definition.

Example 6-3 Defining the output schema that the agent should follow

#Output schema (also known as the agent's control logic)

You communicate only in instruction lines. The format is: "Instruction: expected output". Only use these instruction lines and must not enter empty lines or anything else between instruction lines. Skip the instruction lines Function Name, Function Input, Function Caption, and Function Output if no function calling is required.

Message: User's message. You never use this instruction line.

Thought: A single-line step-by-step plan of how to answer the user's message. You can use the available functions that are defined above. This instruction line must be immediately followed by Function Name if one of the available functions that are defined above needs to be called, or by Final Answer. Do not provide the answer here.

Function Name: Name of the function. This instruction line must be immediately followed by Function Input.

Function Input: Function parameters. An empty object is a valid parameter.

Function Output: Output of the function in JSON format. Thought: Continue your thinking process.
Final Answer: Answer the user or ask for more information or clarification. It must always be preceded by Thought.

Also, the agent can enhance its functions with tools, which extend the LLM's capabilities by integrating external APIs or specialized modules. Each tool includes the following items:

- ▶ A tool description: An explanation of its purpose and function.
- ▶ A tool schema: Specifications for how the LLM should interact with the tool, which includes input and output formats.

Example 6-4 shows the example parsed output after a certain user request regarding an explanation of a certain dataset file occurs.

Example 6-4 Agent output example

Thought: The user wants to know what the file "CSV- bill of materials.csv" is about. I can use the Python tool to read the file and provide a summary.

Function Name: Python

Function Input: {"language":"python","code":"import pandas as pd\n\ndf = pd.read_csv('CSV-billmaterials.csv')\nprint(df.head())","inputFiles":{"file_670d56abb912d7771371652e":"CSV- bill of materials.csv"}}

Function Output: The code ran successfully.

Standard output: '''

```
QUANTITY ... PART
0 3700 ... 144EC8-14101-20
1 5500 ... 096EUF-T4101D20
2 45000 ... 004ZTF-41Z01M20
3 4334 ... NaN
4 1564 ... NaN
```

[5 rows x 4 columns] '''

By combining these components, the system prompt orchestrates the interaction between the LLM and external resources, which help ensure consistency, precision, and task alignment. This modular approach enables AI agents to handle complex workflows and adapt to diverse applications.

This standard agent architecture can generate and run plans, and iteratively refine its approach by leveraging feedback and memory updates. The modular design of agents, which is supported by a versatile Tool Library, enables scalability and adaptability for a wide range of use cases, and with the usage of agents on watsonx.ai, the scaling to enterprise-grade solutions is once again possible.

6.2 Why AI agents are needed

The necessity for AI agents stems from the growing complexity of tasks and environments that surpass the capabilities of conventional software systems. Traditional systems often falter in scenarios requiring adaptive, context-aware decision-making, especially when dealing with vast datasets, uncertain outcomes, and real-time constraints. In domains like logistics, healthcare, and finance, where decision-making must balance multiple variables simultaneously, the utility of AI agents becomes evident. These agents excel in scenarios demanding real-time responses, such as autonomous vehicles navigating dynamic traffic conditions or virtual assistants managing multifaceted user requests.

Beyond efficiency, AI agents also bring about significant cost savings by automating repetitive tasks, reducing human error, and improving operational accuracy. Their ability to adapt from experience by using a memory module and to adapt to novel situations is crucial in addressing problems where predefined solutions are inadequate. For example, reinforcement learning (RL) techniques enable AI agents to optimize behavior over time, which refines their decision-making capabilities with minimal human input. By acting as adaptive problem solvers, AI agents provide a bridge between theoretical AI research and practical, real-world enterprise-grade applications.

Figure 6-4 shows the gen AI journey for AI agents.

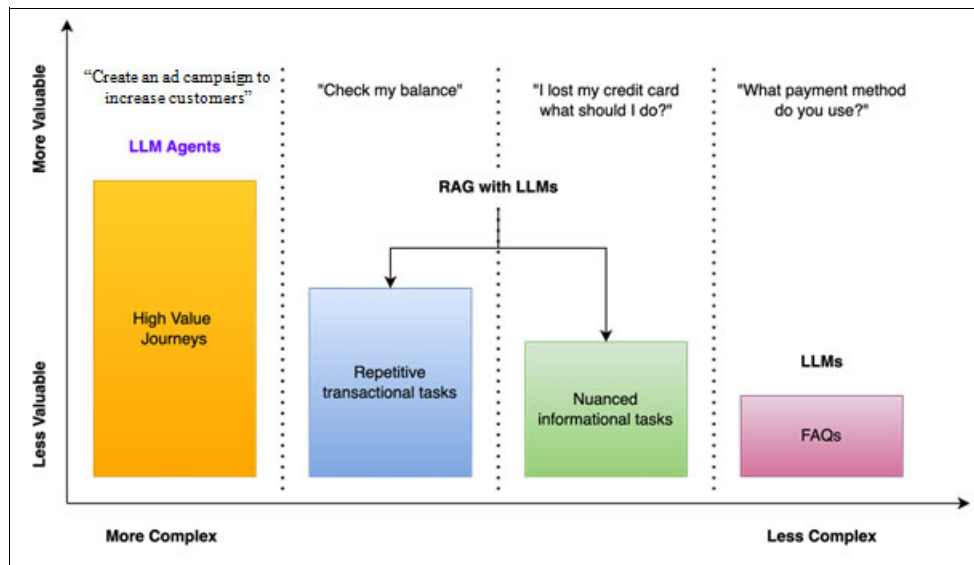


Figure 6-4 Generative AI journey for AI agents

Agents that are powered by LLMs represent the next frontier in driving productivity gains for enterprises. As businesses increasingly rely on AI to streamline operations and elevate customer experiences, agents offer a revolutionary step forward by automating complex, multi-step tasks that previously required human intervention. Unlike traditional LLMs, which excel at handling FAQs or supporting nuanced informational queries through RAG approaches, agents bring advanced capabilities to orchestrate and run high-value workflows. This ability to handle complex scenarios such as planning a marketing campaign, optimizing supply chain logistics, or conducting sophisticated data analysis, positions agents as essential tools for transforming enterprise productivity.

The true potential of agents lies in their capacity to integrate seamlessly with existing systems, tools, and data sources, which enable them to act as intelligent intermediaries that connect disparate workflows. For enterprises, this approach means automating entire business processes rather than isolated tasks. For example, an agent can retrieve customer data, analyze purchasing patterns, generate personalized recommendations, and trigger actions like sending tailored offers or updating customer relationship management systems. By eliminating manual handoffs and streamlining processes, agents enable employees to focus on strategic decision-making rather than repetitive or time-consuming tasks.

From a business perspective, agents are the key to unlocking the next wave of productivity gains. They enhance operational efficiency, reduce costs, and drive faster time-to-market for critical initiatives. Moreover, they enable businesses to scale their efforts without proportionally increasing resources, which are vital in today's competitive and resource-constrained environments. Imagine an agent that can plan, run, and monitor an entire ad campaign or a product launch task in hours, which typically require weeks of human effort. This scalability improves efficiency, and it creates opportunities for innovation because teams can redirect their focus to higher-value, creative, and strategic activities. In this context, agents are a technological enhancement and a strategic imperative for the modern enterprise. They represent a shift from reactive to proactive operations, which enable businesses to anticipate needs, respond faster to market dynamics, and drive growth in ways previously unimaginable. Enterprises that adopt agents will lead the charge in this new era of productivity, and set the benchmark for operational excellence and innovation in their industries.

6.3 Multiple AI agents

When you extend the concept of individual agents in software systems, you get multi-agent systems (MASs), which consist of multiple AI entities working collaboratively or competitively to solve complex problems within shared digital environments. These systems are transformative when augmented by gen AI capabilities because they enable agents to engage in more sophisticated tasks, such as content generation, dynamic interaction with users, or collaborative reasoning. In a software-based MAS, each agent operates autonomously while adhering to predefined protocols for communication and collaboration. This autonomy enables agents to handle tasks like distributed resource management, adaptive problem-solving, and even creative endeavors, such as generating ideas, plans, or personalized user experiences. Gen AI further amplifies their function by enabling natural language generation, image synthesis, and decision-making support, which enhances the agents' ability to interpret, reason, and produce outputs in real time.

A key challenge in software-based MASs, particularly ones that leverage gen AI, is achieving effective inter-agent communication and collaboration. Protocols that are based on message-passing, such as JSON over RESTful APIs or advanced graph-based communication models, help ensure that agents can share knowledge, negotiate responsibilities, and resolve conflicts. Generative AI enhances these interactions by enabling context-aware dialog and summarization capabilities, which make inter-agent exchanges more natural and efficient. Moreover, coordination strategies within such systems are pivotal. In leader-follower setups, generative agents with advanced reasoning capabilities may take on supervisory roles to craft high-level plans or synthesize insights for the collective. Fully distributed MASs enable agents to collaborate dynamically by relying on peer-to-peer negotiations or RL-based decision policies to achieve shared objectives. For example, in a collaborative creative task such as automated marketing content generation, different agents in the system might specialize in headline creation, visual asset generation, and audience sentiment analysis. Together, these agents leverage gen AI to deliver cohesive, high-quality outputs that surpass the capabilities of individual components.

The integration of a MAS with gen AI promises to unlock solutions for some of the most intricate software challenges, such as distributed knowledge synthesis, large-scale content personalization, and dynamic adaptation to user behaviors. As these systems continue to evolve, they are poised to redefine how software systems interact, collaborate, and solve problems, which pave the way for a new era of intelligent, cooperative, and gen AI-driven applications.

The decentralized nature of a MAS is critical in these addressed applications, which help scalability, robustness, and fault tolerance. For example, a generative MAS that is deployed in customer support can assign tasks dynamically across agents, with some agents generating empathetic responses, other agents crafting visually appealing solutions, and yet other agents analyzing sentiment data in real time. This division of labor enables the system to manage workloads efficiently, respond quickly to changes, and help ensure seamless service continuity even if individual agents face disruptions.

Being decentralized is one of the key advantages of a MAS. Unlike centralized systems where a single entity controls decision-making and system-wide operations, MASs are designed to operate without a single point of failure. If one agent fails or is compromised, the rest of the system can continue to function, which makes it highly resilient. This decentralized approach also allows MASs to scale efficiently because more agents can be added to the system without disrupting its overall performance. Furthermore, because the agents are distributed, they can operate in diverse environments, which include real-time scenarios, where traditional centralized systems might struggle. The power of a MAS lies in this collective intelligence, where the sum of the parts is greater than the individual agents' abilities.

As MASs continue to evolve, they are poised to unlock solutions for some of the most intricate challenges in fields such as distributed computing, environmental monitoring, and autonomous exploration. For example, in distributed computing, a MAS can optimize resource usage across a network of machines, which enable more efficient computation and data processing. In environmental monitoring, you can use a MAS to deploy a network of sensors that autonomously gather and process data, which provides real-time insights into environmental conditions. In autonomous exploration, a MAS can enable a fleet of robotic vehicles or drones to work together to explore unknown terrains, share information, and adapt to changes in the environment.

A notable development in the realm of MASs is the advent of multi-agent orchestration frameworks. These frameworks provide a unified platform for coordinating conversations among multiple agents, serving as a high-level abstraction for using FMs. By integrating LLMs, tools, and human inputs, various frameworks enable the seamless orchestration of agent interactions in a way that enhances the capabilities of individual agents. These frameworks typically feature highly capable, customizable, and conversational agents, which can collaborate through automated agent chats, which improve their collective ability to handle complex tasks. This integration enables MASs to function more efficiently by leveraging the power of language models, tools, and human expertise in a coordinated manner. As a result, MAS frameworks offer an exciting potential for creating more intelligent and adaptive systems that can address a wide range of challenges in both industrial and research contexts.

The ongoing evolution of MASs promises to push the boundaries of what is possible in autonomous systems by enabling new capabilities and driving innovation across various industries. As researchers continue to explore the potential of these systems, new frameworks, coordination strategies, and interaction protocols continue to emerge, further enhancing the power and flexibility of MASs in tackling the most complex and large-scale problems.

The architectural diagram in Figure 6-5 outlines a sophisticated MAS framework that is designed to integrate diverse functions such as planning, memory management, communication, and task orchestration. The system employs a modular design that facilitates scalability, adaptability, and interoperability, which caters to complex problem-solving scenarios across various domains. Let us delve into each component and their interplay within the architecture.

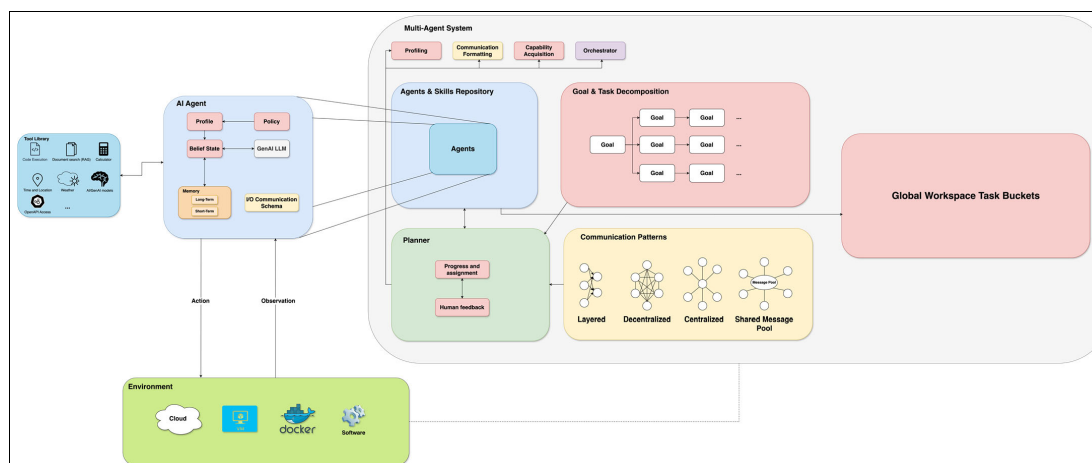


Figure 6-5 High-level view of a multi-agent system architecture

A MAS that is enhanced by gen AI represents a sophisticated and transformative paradigm in AI, where multiple autonomous agents collaborate or interact within a shared environment to achieve complex goals. These systems are built to harness the synergy of diverse agent capabilities by leveraging the powerful reasoning, creativity, and contextual understanding of LLMs to tackle problems beyond the scope of any single agent. Figure 6-5 lays out a comprehensive blueprint for such a system. Now, we will see a detailed, expert-level explanation of its various interconnected components and their interplay.

At the heart of this MAS architecture lies the concept of the agent, which operates as an autonomous unit that is equipped with distinct roles, behaviors, and tools. Each agent is imbued with a profile or persona that defines its domain expertise, operational boundaries, and response style. For example, an agent might function as a scientific researcher that is skilled in computational chemistry or as a customer service representative with expertise in natural language understanding and resolution strategies. This persona is not static because it evolves in response to feedback and environmental changes, which help ensure that the agent remains relevant and effective in dynamic settings.

Central to an agent's operation is its belief state, which is an internal model that encapsulates its understanding of the world, its knowledge of tasks at hand, and its memory of past interactions. This belief state is dynamic and constantly updated through observations, actions, and communications with other agents or external systems. The belief state also integrates inputs from memory. Memory is bifurcated into long-term and short-term storage, each serving distinct roles. Short-term memory retains ephemeral information that is crucial for immediate task execution and contextual reasoning. For example, an agent might use short-term memory to temporarily store user input or intermediate results of calculations. In contrast, long-term memory preserves knowledge that is accumulated over time, such as procedural expertise, domain-specific facts, or historical interactions. The integration of memory with the agent's belief state helps ensure that decisions are informed by both the immediate context and historical knowledge. This dual-layered memory structure enhances the system's ability to handle complex, multi-step tasks that require contextual continuity and long-term planning.

Driving the agent's reasoning and interaction is its *policy*, a framework of decision-making rules and algorithms that govern how the agent interprets inputs, prioritizes tasks, and generates outputs. Policies can be simple, predefined heuristics or sophisticated, dynamically updated models that are informed by machine learning (ML) techniques, such as RL. These policies are operationalized through the agent's LLM core, which is the generative engine that processes natural language inputs, synthesizes insights, generates context-aware responses, and even creates novel solutions to problems. The LLM also serves as the agent's interface to interpret ambiguous or unstructured information, which effectively bridges the gap between human language and computational logic.

Agents in this system are equipped with advanced communication capabilities that are defined through a structured I/O schema that enables them to interact with other agents, external tools, and the environment. Communication is about exchanging messages, negotiating shared understandings, aligning goals, and ensuring coordinated action. To accomplish these tasks, agents can rely on tools that are integrated within the system. These tools extend the agent's functional repertoire and include utilities like API access for external data retrieval, calculators for mathematical computations, code interpreters for debugging and running programs, and multimodal models for handling complex data types, such as images or video.

For domain-specific applications, tools like chemistry simulators or Robotic Process Automation (RPA) frameworks can be deployed, which enable agents to specialize in tasks that range from molecular modeling to workflow optimization. These tools are seamlessly integrated into the agent's workflow, which enables it to run specialized tasks without requiring extra human intervention.

There is no defined limit or minimum requirement regarding the number of tools that are available to a set of agents: it is something dependent on the specific use cases that are addressed. What is proposed here is just a potential set of tools that might be useful for various potential use cases in a specific set of industries. In a MAS, a set of tools might contain hundreds of tools, depending on what is the goal of the system.

At a higher level, the MAS orchestrates the activities of these agents to help ensure that they work collaboratively toward shared objectives. This orchestration is managed by several critical components.

The first is the goal and task decomposition mechanism, which takes high-level goals and breaks them into smaller, manageable tasks and subtasks. For example, if the system's goal is to generate a comprehensive market analysis, this task might be decomposed into tasks like gathering data, analyzing trends, and generating a summary report, with each task that is further divided into specific steps like querying databases or visualizing data.

To enable task execution, the system relies on a planner that assigns tasks to the most suitable agents based on their profiles, current workloads, and skillsets. The *planner* plays a pivotal role in the MAS by orchestrating the decomposition of high-level goals into granular tasks and subtasks. The planning process leverages the agent's policy and LLM to identify the optimal sequence of actions that are required to achieve the outcome. The planner also monitors progress, adapting task sequences or agent roles as needed to accommodate changing conditions or unexpected challenges. For example, if one agent encounters a bottleneck in data retrieval, the planner can reassign related tasks to another agent with overlapping capabilities.

For critical use cases, the architecture incorporates the ability to receive *human feedback* during the execution of the plan to help ensure that automation in high-stakes scenarios remains governable and aligned with human oversight. By enabling human intervention, the system can address unforeseen complexities, validate decisions, and maintain control over critical operations.

Task decomposition ensures that even complex objectives are approached methodically, with subtasks delegated to the appropriate agents or tools.

Collaboration within the MAS is further enhanced through sophisticated communication patterns, which define how agents interact and share information. The architecture supports multiple *communication patterns*:

- ▶ Layered communication: Hierarchical interactions where agents operate at different levels of abstraction by passing information up or down the chain.
- ▶ Decentralized communication: Peer-to-peer exchanges among agents to help ensure flexibility and reduce bottlenecks.
- ▶ Centralized communication: A hub-and-spoke model where a central coordinator manages all interactions.
- ▶ Shared message pool: A collaborative mechanism where agents exchange messages through a shared repository.

Hybrid patterns, such as shared message pools or global workspaces, enable agents to post intermediate results or discoveries to a common repository, which enables asynchronous collaboration and emergent problem-solving. The *blackboard pattern* or global workspace serves as a central knowledge-sharing hub within the MAS. Agents use this shared repository to post updates, intermediate calculations, or unresolved queries, which create a collaborative environment where other agents can contribute insights or take over pending tasks. For example, an agent working on a data analysis task might upload partial results to the blackboard, which another agent can use to generate visualizations or summaries.

The MAS is supported by an agents and skills repository, which is a centralized directory that catalogs the capabilities, expertise, and tools that are associated with each agent. This repository enables dynamic discovery and allocation of agents for specific tasks, which helps ensure that the system can scale and adapt to diverse challenges. It also facilitates the incorporation of new skills or agents so that the system can evolve as new tools or requirements emerge.

The architecture helps ensure that the output that is generated by the system is coherent, contextually relevant, and correctly formatted. The *I/O schema* governs the structure of inputs and outputs by standardizing interactions across agents, tools, and external systems. This schema helps ensure compatibility and consistency regardless of the task or domain. The system's communication module also plays a role in tailoring outputs to the intended audience. For example, technical results might be presented in a concise, data-rich format for domain experts, but layperson-oriented outputs would emphasize clarity and simplicity.

Despite its sophisticated design, the MAS must handle potential errors that can arise during execution:

- ▶ Failed API calls to tools: A tool might be temporarily unavailable or malfunction, which can lead to incomplete or erroneous task execution. Human feedback can help decide whether to retry the tool, use an alternative tool, or modify the task parameters.
- ▶ Infinite loops: Erroneous task decomposition or planning might result in a loop where the system repeatedly runs the same actions without progress. Human intervention can identify and correct the root cause to prevent resource wastage.
- ▶ Rogue paths or wrong tool selection or input: The system might choose an inappropriate tool or misinterpret input data, which can lead to incorrect outputs. Human feedback can help realign the system’s actions to help ensure that the task remains on track.

By integrating human feedback into the loop, particularly for infinite loops or rogue paths, the MAS can maintain a high degree of reliability and robustness by implementing a fail-safe technique on human-in-the-loop interaction after deviant agents’ behavior is detected by the overall orchestration system. This hybrid approach of automation with human oversight is especially critical in high-stakes domains where errors can have significant consequences.

The MAS operates within an environment, such as the external world or a digital context where tasks are performed and results are applied. Agents interact with the environment by observing its state, acting to modify it, and processing feedback to refine their belief states and policies. The environment is also where the whole MAS runs, and it is possible to have environments that are composed of multiple locations, such as cloud environments, virtual machines (VMs) and containers, or further proprietary software.

6.4 AI agents on watsonx.ai

IBM watsonx.ai enhances the development of an enterprise-grade agentic technology stack (Figure 6-6) by providing powerful tools, models, and middleware capabilities that are tailored for scalable, intelligent, and adaptive operations.

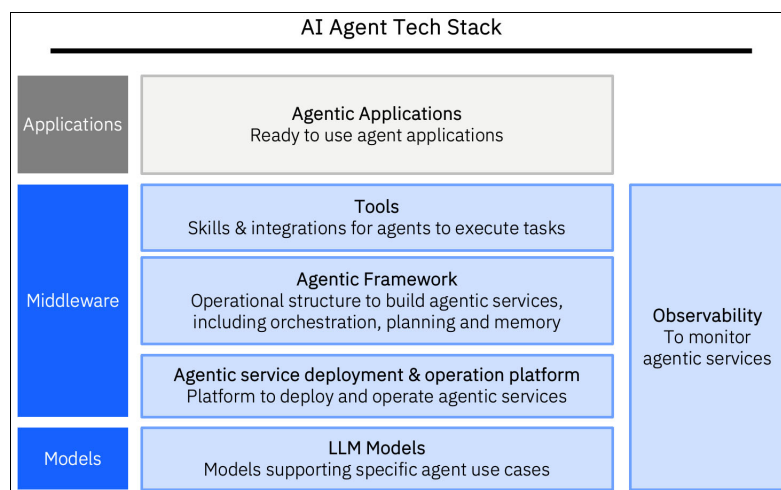


Figure 6-6 Enterprise agentic tech stack

At the foundation level, watsonx.ai delivers robust LLMs optimized for specific enterprise use cases to enable agents to interpret and act on complex queries with high accuracy and relevance. These LLMs seamlessly integrate into the Agentic Framework, which serves as the operational backbone for orchestrating tasks, planning workflows, and retaining memory for context-driven decision-making.

IBM watsonx.ai also supports the Agentic Service Deployment & Operation Platform, which offers a streamlined way to deploy and manage agentic services while ensuring reliability, scalability, and security at an enterprise level. To ensure enterprise-grade observability and monitoring, watsonx.ai incorporates observability mechanisms that provide real-time insights into agent performance, operational health, and user interactions. These mechanisms facilitate continuous optimization and help ensure that agentic services align with business goals. By bridging models, middleware, and applications, watsonx.ai creates a cohesive and modular enterprise tech stack that can address the evolving demands of modern businesses with precision, adaptability, and innovation.

watsonx.ai agents are a transformative innovation in the domain of AI. These agents are designed to provide businesses with unparalleled capabilities in automating tasks, processes, and decision-making. Through a blend of interfaces, cutting-edge technologies, and robust integration options, watsonx.ai enables the development, deployment, and optimization of intelligent agents that cater to a diverse range of enterprise needs.

At the core of watsonx.ai's agentic capabilities is the Agent Builder (Figure 6-7), which is an intuitive and powerful tool that accelerates the development lifecycle. The visual interface of the Agent Builder enables developers to construct agents with ease, which reduces the complexity that is typically associated with designing and managing such systems. Agents within watsonx.ai are defined through natural language instructions, and they can be equipped with various tools to expand their functions. These tools act as modular building blocks to enable developers to create sophisticated workflows that are tailored to specific requirements.

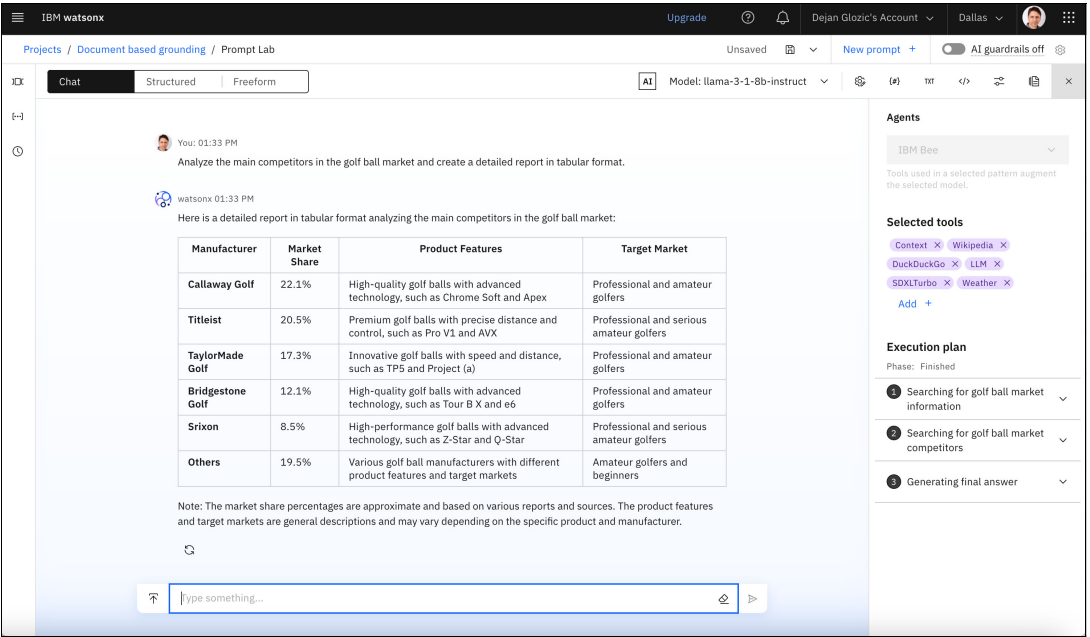


Figure 6-7 Agent Builder view on the watsonx.ai UI

One of the standout features of the Agent Builder is its seamless integration with multiple agent frameworks. In addition to IBM proprietary technologies, developers can also leverage popular open-source frameworks like LangChain and LangGraph. This flexibility helps ensure that businesses can use the best tools and methodologies that are available in the ecosystem, and adapt them to their unique operational needs. The ability to integrate open-source solutions with IBM advanced technologies provides a level of customization and extensibility that is critical for modern enterprises, as shown in Figure 6-8.

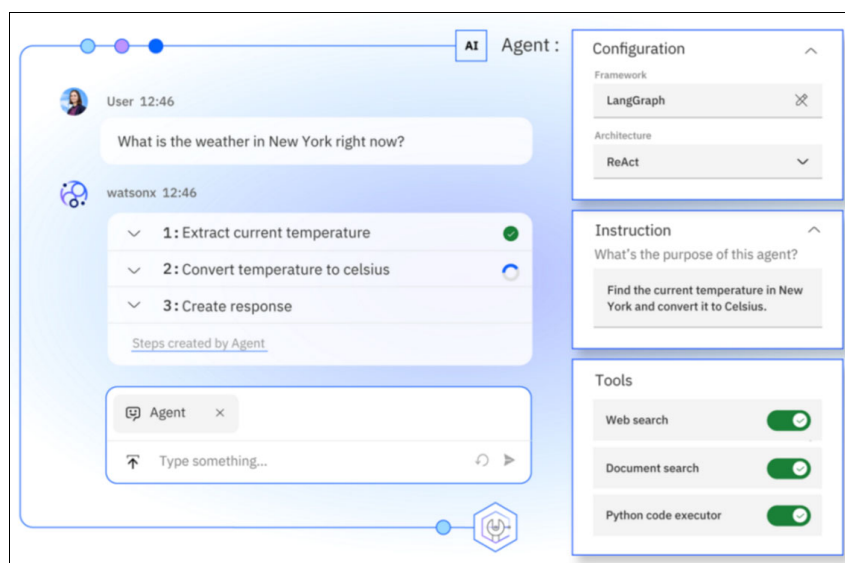


Figure 6-8 Overview of customizations on watsonx.ai Agent Builder

Testing and debugging are essential components of the agent development process, and watsonx.ai Agent Builder excels in this area. Real-time testing capabilities enable developers to identify and resolve issues as they arise, which minimizes downtime and iteration cycles. This feature is complemented by the “one-click” deployment mechanism, which simplifies the process of making agents operational. Once developed, agents can be deployed as watsonx.ai AI services, which effectively turn them into API endpoints that can be accessed by various applications and systems. This streamlined workflow reduces time-to-market and helps ensure that agents can be quickly integrated into enterprise operations.

To further enhance the functions of agents, watsonx.ai offers an extensive Tool Library of enterprise-ready tools that are designed to augment the capabilities of agents. The tools are divided into the following categories:

- ▶ Web Search
- ▶ Document Search (RAG)
- ▶ Code Execution
- ▶ Data Connectors
- ▶ Custom Tool Builder

For example, the Web Search tool empowers agents to perform real-time internet searches, which provide them with up-to-date information to enhance their decision-making and responses. The ability to access fast and relevant search results helps ensure that agents remain informed and capable of handling dynamic queries.

Another critical component of the Tool Library is the Document Search function, which uses RAG. This tool enables agents to efficiently index and retrieve documents from an organization's knowledge base, which helps ensure that they can deliver accurate and context-aware responses. By leveraging RAG, agents can access vast amounts of information and distill it into actionable insights, which makes them invaluable for knowledge-intensive tasks.

The Tool Library also includes a Code Execution feature, which enables agents to run Python code in real time. This capability opens up a wide range of possibilities, from performing complex calculations to automating repetitive tasks. By integrating this feature, agents can operate as dynamic problem solvers that can adapt to various scenarios.

Data accessibility is another cornerstone of the watsonx.ai agentic framework. With Data Connectors, agents can seamlessly interact with enterprise databases and data warehouses, which grant them access to critical organizational data. This tool helps ensure that agents operate with a comprehensive understanding of the business context, which enables more informed and effective decision-making.

The Tool Library supports the creation of custom tools so that organizations can extend the functions of their agents by integrating them with external services and unique enterprise systems. This level of customization helps ensure that agents can meet the specific demands of any organization.

Deployment is a critical phase in the lifecycle of any AI system, and watsonx.ai provides a robust, framework-neutral solution for deploying agents. The deployment process is scalable, secure, and highly available, which helps ensure that agents can meet the demands of enterprise-scale operations. Whether an organization requires a single agent for a specific task or a fleet of agents to handle complex workflows, the watsonx.ai deployment capabilities can handle the challenge. Once deployed, the performance and reliability of agents must be monitored to ensure that they operate as intended. watsonx.ai includes comprehensive monitoring tools that track key performance indicators (KPIs) and analyze logs. These tools provide valuable insights into agent behavior, which enable developers and administrators to identify areas for improvement and help ensure that agents deliver optimal results.

watsonx.ai emphasizes transparency and explainability, which are crucial for building trust in AI systems. By offering detailed explanations of agent decisions and actions, the platform helps organizations maintain compliance with regulatory requirements and ethical standards.

Looking to the future, watsonx.ai is poised to introduce the Flows Engine, which is a lightweight agentic framework and tool-building platform that will further enhance the capabilities of the platform. The Flows Engine will enable developers to rapidly build custom tools and integrate them with enterprise IT systems, which will provide unparalleled flexibility. This framework is designed to facilitate reasoning and complex decision-making, which will make agents more effective in handling intricate tasks. Also, the Flows Engine will include a chat UI widget that can be easily integrated into third-party applications, which will enable seamless AI-mediated interactions.

Complementing watsonx.ai is IBM watsonx Orchestrate, which is a platform that focuses on starting AI assistants and agents for business processes and task automation (Figure 6-9). By combining the capabilities of watsonx.ai and watsonx Orchestrate, organizations can achieve end-to-end automation to streamline operations and drive efficiency across their workflows. This synergy highlights IBM's commitment to providing comprehensive AI solutions that address the diverse needs of modern enterprises.

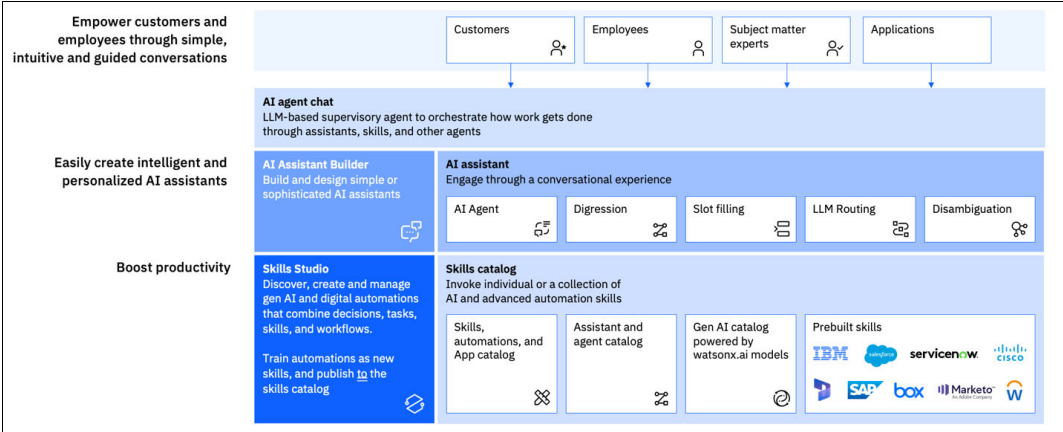


Figure 6-9 Overview of watsonx Orchestrate for Agents capabilities

The integration of watsonx Orchestrate and watsonx.ai offers an advanced, enterprise-grade framework that enhances agentic support by uniting robust automation, intelligent workflows, and conversational AI capabilities. watsonx Orchestrate functions as a supervisory agent that leverages LLMs to coordinate interactions across customers, employees, subject matter experts (SMEs), and applications. By employing Skills Studio, watsonx Orchestrate enables the discovery, creation, and management of gen AI and digital automations by combining tasks, workflows, and skills to drive seamless operational efficiency. Skills and automations can be trained and published into a comprehensive skills catalog, which includes prebuilt integrations with enterprise solutions such as SAP, Salesforce, and ServiceNow, to help ensure compatibility with diverse enterprise systems. Simultaneously, watsonx.ai powers intelligent AI assistants, enabling conversational experiences that feature advanced functions like slot filling, LLM routing, disambiguation, and digressions. This combination helps ensure personalized and context-aware interactions. Together, watsonx Orchestrate and watsonx.ai establish a scalable ecosystem to enable enterprises to simplify complex processes, reduce operational bottlenecks, and deliver intuitive, guided experiences that align AI-driven solutions with business objectives in a dynamic, flexible, and secure manner.

Because of these agentic frameworks on watsonx.ai that are combined with watsonx Orchestrate, you can use advanced assistants that can perform many types of tasks.

Figure 6-10 on page 105 provides a comprehensive depiction of Assistants with Agents on watsonx, which shows the integration of watsonx.ai and watsonx Orchestrate to enable sophisticated AI-driven solutions for enterprise workflows.

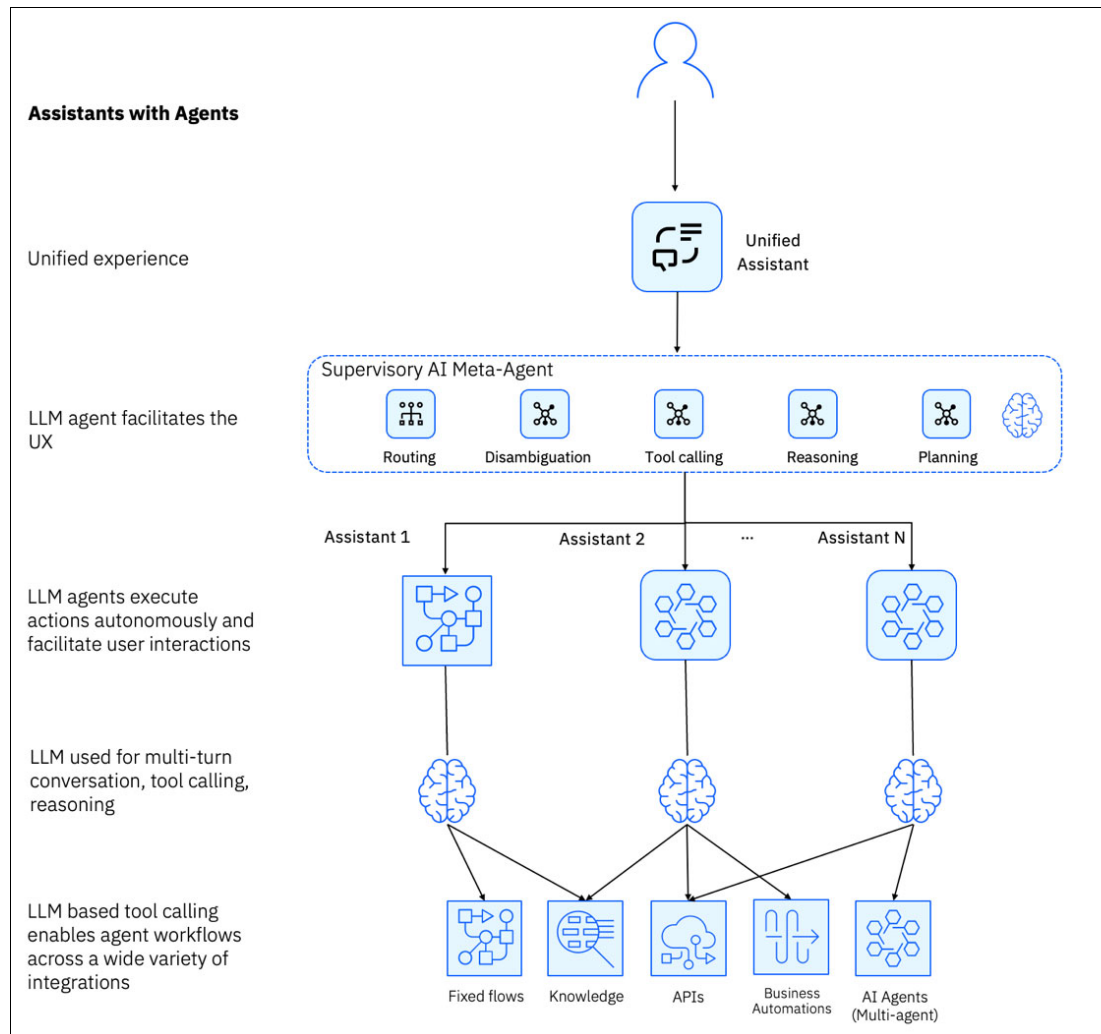


Figure 6-10 Assistant with Agents that use watsonx.ai and watsonx Orchestrate for agentic use cases

At the core of this design is the concept of a unified, intelligent assistant framework that combines modularity, scalability, and precision to deliver seamless experiences across diverse use cases and tasks. Each component in Figure 6-10 plays a critical role in orchestrating complex interactions between users, agents, tools, and workflows, which create a robust and adaptable ecosystem for AI-powered operations.

At the top of the architecture is the Unified Assistant, which serves as the single, user-facing interface. This layer is designed to provide a consistent and coherent interaction experience by simplifying user engagement and abstracting the complexities of the underlying system. The Unified Assistant is supported by the Supervisory AI Meta-Agent, which is a central coordinator that facilitates all workflows and helps ensure that user requests are processed accurately and efficiently. The meta-agent acts as the brain of the system by orchestrating the activities of multiple subordinate agents and tools to fulfill user intents. Its responsibilities include routing tasks to the appropriate agents, resolving ambiguities in user input, starting the necessary tools, reasoning through multi-step problems, and planning actions to achieve outcomes. These capabilities are made possible by the integration of LLMs that are powered by watsonx.ai, which provides the advanced natural language understanding, contextual awareness, and reasoning abilities that are needed for high-quality interactions.

Beneath the Supervisory AI Meta-Agent lies a network of specialized assistants that are labeled as Assistant 1, Assistant 2, and so on, which represents a modular and scalable approach to task execution. Each assistant is tailored to handle specific domains or functions, and they operate autonomously while contributing to the overall system. These assistants are designed to run actions independently, and they leverage the power of LLMs for NLP, multi-turn conversations, and decision-making. This autonomy enables them to reduce manual intervention so that organizations can achieve higher levels of efficiency and productivity. The assistants also facilitate user interactions by maintaining context, understanding intent, and responding dynamically to evolving requirements.

A defining feature of this architecture is its reliance on tool calling, which forms the backbone of the system's operational flexibility and extensibility. Figure 6-10 on page 105 emphasizes the integration of various categories of tools that the agents can call to complete tasks. These tools include fixed flows for handling repetitive and standardized operations; knowledge repositories for answering questions and providing insights; APIs for interacting with external systems; business automation modules for streamlining enterprise processes; and multi-agent frameworks for coordinating complex tasks that require collaboration among several AI agents. The usage of LLM-based tool calling helps ensure that the system can adapt to various workflows, which enables interoperability with existing IT infrastructures and third-party applications. watsonx.ai enhances this capability through its extensive Tool Library, which includes features such as RAG for knowledge discovery; real-time Python code execution for computational tasks; and seamless data connectors for integrating with enterprise databases and services. Custom tools can also be developed and incorporated, which enable organizations to tailor the system to their unique needs and challenges.

The bottom part of Figure 6-10 on page 105 implicitly connects to watsonx Orchestrate, which is the IBM platform for deploying and managing AI-driven workflows. By leveraging watsonx Orchestrate, this architecture gains enterprise-grade capabilities for task automation, governance, and monitoring. This integration enables organizations to deploy AI assistants quickly and securely, with the ability to scale the system as the number of tasks, agents, and integrations grows. Security and compliance are also ensured, which addresses the stringent requirements of enterprise environments. The orchestration layer further optimizes performance by streamlining the deployment and management of agents, which minimizes operational overhead while maximizing system reliability.

The overall interaction flow in Figure 6-10 on page 105 begins with the user engaging with the Unified Assistant. The user's input is processed by the Supervisory AI Meta-Agent, which applies its routing, reasoning, and tool-calling capabilities to determine the best course of action. Then, tasks are delegated to the appropriate assistants, which run them autonomously by using the integrated tools and workflows. The outputs and actions from these assistants are aggregated by the meta-agent and presented to the user, which helps ensure a cohesive and intuitive experience. This flow highlights the system's ability to handle diverse and complex tasks while maintaining a simple interface.

In conclusion, watsonx.ai agents represent a revolutionary approach to enterprise AI by offering a powerful combination of flexibility, function, and scalability. From the intuitive Agent Builder to the expansive Tool Library and forthcoming innovations like the Flows Engine, watsonx.ai empowers businesses to create intelligent agents that drive productivity and innovation. By leveraging these advanced capabilities, organizations can harness the full potential of AI to achieve their strategic goals and maintain a competitive edge in an increasingly digital world.

6.5 AI agents use case examples

AI agents are redefining the landscape of business operations and service delivery by unlocking unprecedented efficiencies and enabling more personalized, data-driven decision-making. Across industries, these intelligent systems have found applications in areas such as customer service and support, sales and marketing automation, operational efficiency, financial advisory, healthcare, and supply chain management.

The following sections provide a detailed exploration of select use cases, and highlight their impact and capabilities.

Customer service and support agents AI agents

AI agents are revolutionizing customer service by offering continuous assistance. By using tools like watsonx.ai Web Search and Document Search (powered by RAG), these agents provide timely and accurate responses to customer inquiries. By leveraging NLP and ML, they can understand context, resolve issues, and escalate complex cases to human agents when necessary. For example, chatbots that are built by using watsonx.ai Agent Builder can be deployed as API endpoints to handle FAQs, help with troubleshooting, and manage order tracking. These agents help ensure instant support while reducing operational costs, improving customer satisfaction, and fostering loyalty.

Sales and marketing automation

AI agents are indispensable tools in sales and marketing, where personalization and timely interactions drive success. Powered by watsonx.ai Data Connectors, agents can analyze customer behavior, qualify leads, and deliver tailored recommendations. By automating follow-ups and dynamically adjusting strategies by using real-time insights, these agents enhance engagement and drive conversions. For example, e-commerce platforms can use watsonx.ai agents to suggest personalized items, send promotional offers, and re-engage customers who abandon carts. Such integrations help businesses maximize revenue potential while delivering highly customized customer experiences.

Operational efficiency and process automation

Automating repetitive and routine tasks is a hallmark of AI agents, and watsonx.ai provides the tools to optimize these processes. By using the Code Execution capability, agents can run Python scripts in real time to process data, verify documents, or automate workflows. Beyond task automation, these agents can monitor workflows, identify bottlenecks, and recommend improvements. For example, administrative tasks like employee onboarding can be fully automated with watsonx.ai by processing documentation, setting up accounts, and scheduling orientation sessions, which free employees to focus on more strategic responsibilities.

Healthcare assistants and patient care

In healthcare, AI agents improve patient care and operational efficiency by acting as virtual assistants. By integrating with watsonx.ai Tool Library and leveraging custom tools, these agents can manage schedules, send medication reminders, and provide basic medical guidance. For example, telemedicine platforms can deploy agents to conduct preliminary symptom checks, monitor health metrics such as heart rate or blood pressure, and alert providers to abnormalities. These capabilities reduce the workload on healthcare staff while ensuring proactive and timely patient care, which enhances outcomes and satisfaction.

Supply chain and logistics optimization

AI agents are transforming supply chain management by combining real-time data with predictive analytics. Tools like watsonx.ai Data Connectors enable agents to forecast demand, manage inventory, and plan delivery routes effectively. Logistics companies can leverage these capabilities to analyze historical patterns, predict future needs, and maintain optimal stock levels. Also, AI-driven route optimization, when informed by traffic and weather data, helps ensure timely deliveries and reduces costs. With watsonx.ai, organizations can build scalable, secure agents that streamline supply chain operations, which enhance both efficiency and customer satisfaction.



Use cases

This chapter describes two separate use cases and shows what problems IBM watsox.ai tools can solve. It also describes a framework that outlines how companies who are trying to prepare for the future are thinking about use cases of the future to keep ahead of the curve.

The following topics are described in this chapter:

- ▶ 7.1, “Using RAG to aid a medical school admissions office” on page 110
- ▶ 7.2, “Embedding workflow automation to streamline recommendations” on page 111

7.1 Using RAG to aid a medical school admissions office

As a quick refresher, Retrieval-Augmented Generation (RAG) is a technique that combines information retrieval and language model generation to provide precise and contextually relevant responses to user queries. It works by first retrieving relevant documents or passages from a large corpus of documents by using a retrieval step, and then feeds these passages along with the original query into a large language model (LLM) to generate a response.

RAG is one of the most commonly used techniques that are used in production AI workloads. It is used in various ways, such as in question-answering systems, chatbots, and digital workers. One powerful tool that can be used in these systems is a summary of ingested documents. This section describes how a leading medical school in the US turned to the watsonx.ai platform to enable it to accomplish its goals.

7.1.1 The challenge

A leading medical school in the US decides to offer tuition-free education to its admitted students. They anticipated a surge in applications. To help manage the expected increase in applications, the institution turned to watsonx.ai. They hoped that IBM could provide a technology solution to help the admissions committee efficiently process and review the incoming applications.

7.1.2 The solution

Working with the medical school, the IBM team developed an innovative solution by using watsonx.ai to generate one-page abstracts that summarized the incoming 50 - 70-page applications. The incoming applications included essays, with each application containing 5 - 8 essays that varied from several paragraphs to several pages. The IBM granite-13b-chat-v2 model within the watsonx.ai platform was used to generate a 1 - 2 paragraph summary of each of the essays, which was included with the application abstracts.

Figure 7-1 shows the watsonx.ai workflow that accomplished this task.

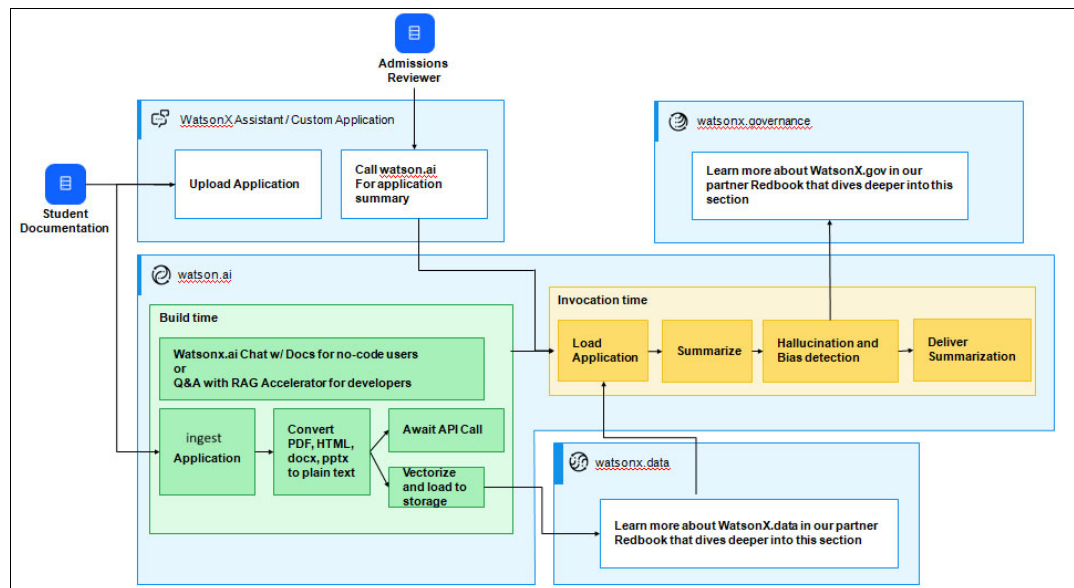


Figure 7-1 watsonx.ai workflow example

7.1.3 Special considerations

From the start of this project, both IBM and the medical school recognized the importance of developing a solution that met the institution's needs and aligned with IBM AI Ethics. The usage of AI in the admissions process raised important ethical considerations, such as helping ensure fairness, transparency, and accountability. The goal was to create a system that would augment human decision-making rather than replace it, and provide the admissions committee with the tools that they needed to make informed decisions.

The project incorporated a range of both technical and non-technical guardrails to address AI ethics considerations throughout the project lifecycle:

- ▶ **Augmenting human decision making:** The solution was designed to support human decision-makers, and not replace them. The AI-powered pipeline generated summaries and identified key information, but all decisions remained in the hands of humans.
- ▶ **Education and training:** The IBM team provided ongoing education and training on AI to both technical and business users to help ensure that everyone that was involved in the project understood the capabilities and limitations of the technology.
- ▶ **Thresholds and AI notices:** The team implemented technical guardrails by using watsonx.governance to detect and prevent potential biases or errors in the system.
- ▶ **Feedback mechanism:** The IBM team established a continuous feedback loop with the client to refine and improve the solution over time.

By considering these guardrails from the beginning, the project was able to meet the institutions needs while aligning to their governance frameworks regarding fairness, transparency, and accountability. It also limited the school's risk exposure and reduced the likelihood of having to redesign the system to incorporate new safeguards because decision points were integrated from the start.

7.2 Embedding workflow automation to streamline recommendations

This use case describes at how workflow automation can lead to directly addressing the wants and needs of a financial institution, and act as a potential opportunity pipeline for the banks serving their customers.

7.2.1 The challenge

Small local and regional banks, especially ones serving customers in more rural settings, often face different challenges than the large banks that service most of the total addressable market. Some of these challenges are self-evident, such as having comparatively limited access to capital, but other challenges are less apparent, such as an increased need to rely on low- and no-code solutions to maintain technological parity with the custom-built applications that are designed and managed by large centralized IT teams that are staffed by larger finance institutions.

For one of these banks servicing rural clients in the Midwest region of the US, they wanted to address some of the market trends they read about in [5 banking customer experience trends to consider for 2024](#), with a specific focus on providing customers with immediate service and personalized recommendations.

In these rural settings, it is often difficult for a bank’s customer base to travel to the nearest branch, and at specific times of the year, the journey takes time away from their responsibilities at farms, ranches, and processing centers, which directly impacts their annual income.

7.2.2 The solution

By leveraging a collection of tools, this regional bank was able to build a solution that incorporated three IBM tools to reduce friction with its users while leveraging largely pre-built skills and solutions. The solution centered on a watsonx Orchestrate business automation application that takes in loan applications and responds to the loan applicant with near-real-time approval or rejection notices based on thresholds that are set by the bank.

Figure 7-2 shows the watsonx Orchestrate workflow that was used in this use case.

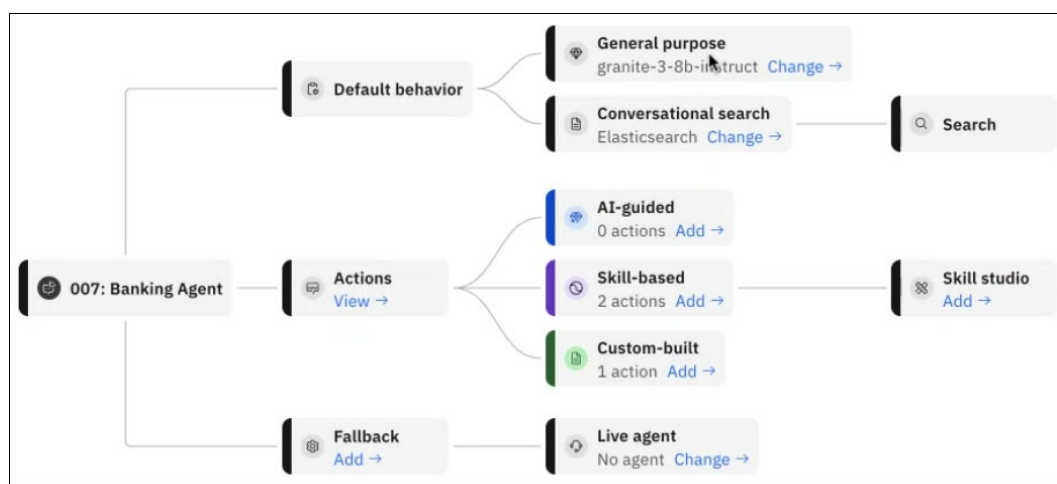


Figure 7-2 watsonx Orchestrate workflow example

Customers connected to the bank through IBM watsonx Assistant, which leveraged an IBM LLM to initiate a natural language dialog with the client and collect various loan application documents, which included custom forms that are specific to this bank. These documents were passed from watsonx Assistant to watsonx Orchestrate, which called on several of its prebuilt skills, and custom skills that were developed in the Skills Studio feature to access additional services outside of the bank (such as credit reports).

After running through a decision tree based on the inputs, the client received an approval or rejection notification. In either case, the notice was sent back to the applicant through a custom-generated response that was tailored by watsonx.ai and specific to that customer, with an explanation of the decision based on that customer’s specific application. Furthermore, the bank leveraged its own client knowledge to augment the loan decision with additional offers or suggestions to the customer based on their specific customer profile. For those customers that applied for an automotive loan and had a mortgage and business account with the bank, the bank suggested that they sign up for a wealth management account that was serviced by the bank. For those customers who applied for a small business loan who did not have a checking or savings account that was associated with the EIN on the application, the bank suggested that they open a full suite of business accounts with the loan. These examples pulled on watsonx.ai generative capabilities to create personalized recommendations based on individual customers rather than “one-size-fits-all” templates that are applied as a blanket policy to all applications. Holistically, this entire solution can be summarized as an AI agent.

7.2.3 Special considerations

With a smaller workforce to dedicate to this solution and limited previous AI experience, the bank leveraged existing tools to reduce the workload and expertise requirements on the bank's workforce. This approach was one of the key reasons IBM and the client focused on leveraging watsonx Orchestrate over designing a custom application that integrated AI tools through API calls. Instead of building a solution, the bank relied on built-in AI functions that automatically combined pre-packaged skills dynamically and in-context based on organizational knowledge and prior interactions to help workers design the workflow of their application. Users provided natural language inputs to select and sequence the required skills for a task, and watsonx Orchestrate connected them with the associated applications, tools, data, and historical details. This approach enabled the team to automate processes without needing highly specialized IT skills or expert knowledge of business processes and applications.

Abbreviations and acronyms

AI	artificial intelligence
AlaaS	AI as a Service
BYOM	Bring Your Own Model
CLI	command-line interface
CNN	convolutional neural network
DL	deep learning
DQN	Deep Q-Network
ESG	environmental, social, and governance
FM	foundation model
gen AI	Generative AI
IBM	International Business Machines Corporation
KNN	k-nearest neighbor
KPI	key performance indicator
LLM	large language model
LLMOps	large language model operations
LoRA	low-rank adaptation
MAS	multi-agent system
MDP	Markov decision processes
ML	machine learning
MLOps	machine learning operations
MMLU	Massive Multitask Language Understanding
NLP	natural language processing
PCA	Principal Component Analysis
QLoRA	quantized low-rank adaptation
RAG	Retrieval-Augmented Generation
RL	reinforcement learning
RNN	recurrent neural network
RPA	Robotic Process Automation
SaaS	Software-as-a-Service
SDG	synthetic data generation
SME	subject matter expert
UI	user interface

Related publications

The publications that are listed in this section are considered suitable for a more detailed description of the topics that are covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topics in this document. Some publications that are referenced in this list might be available in softcopy only.

- ▶ *Simplify Your AI Journey: Ensuring Trustworthy AI with IBM watsonx.governance*, SG24-8573
- ▶ *Simplify Your AI Journey: Unleashing the Power of AI with IBM watsonx.data*, SG24-8570

You can search for, view, download, or order these documents and other Redbooks, Redpapers, web docs, drafts, and additional materials, at the following website:

ibm.com/redbooks

Online resources

These websites are also relevant as further information sources:

- ▶ Code samples of common machine learning scenarios:
<https://github.com/IBM/watson-machine-learning-samples>
- ▶ Examples of using Instructlab and AI agents:
<https://github.com/IBM/watsonx-ai-platform-demos>
- ▶ IBM AI risk atlas
<https://www.ibm.com/docs/en/watsonx/w-and-w/2.1.x?topic=ai-risk-atlas>
- ▶ IBM watsonx documentation (Includes links to all watsonx products)
<https://www.ibm.com/docs/en/watsonx>
- ▶ IBM watsonx.governance product
<https://www.ibm.com/products/watsonx-governance>
- ▶ IBM watsonx product portfolio
<https://www.ibm.com/watsonx>

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



SG24-8574-00

ISBN 0738461989

Printed in U.S.A.

Get connected

