CAPSTONE PROJECT

TRAVEL PLANNER AGENT (PROBLEM STATEMENT .5)

Presented By:
GOURI NILESH DHUMAK-B.K.BIRLA COLLEGE, KALYAN (AUTONOMOUS)-B.Sc DATA SCIENCE

edunet foundation

OUTLINE

- **Problem Statement** (Should not include solution)
- **Proposed System/Solution**
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment**
- **Result (Output Image)**
- **Conclusion**
- **Future Scope**
- **References**

PROBLEM STATEMENT

Problem Statement No.5 - Travel Planner Agent

The Challenge - A Travel Planner Agent is an AI-powered assistant that helps users plan trips efficiently and intelligently. It uses real-time data to suggest destinations, build itineraries, and recommend transport and accommodation options. By understanding user preferences, budgets, and constraints, it tailors personalized travel plans. Integrated with maps, weather updates, and local guides, it ensures a smooth travel experience. The agent can also manage bookings, alert users to changes, and optimize schedules on the go. This smart assistant transforms complex travel planning into a seamless, enjoyable process.

PROPOSED SOLUTION

Proposed Solution: Travel Planner Agent

1. Data Collection:

•**User-Provided Data:** Actively collect user input via a conversational interface. This includes explicit preferences (travel style: adventure, relaxation; interests: history, food), constraints (budget, travel dates, mobility limitations), and trip goals. Historical data from a user's past trips and feedback will also be stored.

•**Real-Time External Data:** Integrate with external APIs to gather live data essential for planning. This will include:

- •**Transportation:** APIs for flights, trains, and rental cars to get live pricing and schedules.
- •**Accommodation:** APIs from hotel and rental platforms for availability, pricing, and reviews.
- •**Geospatial & Weather:** Mapping services for routing and location data, and weather APIs (like The Weather Company, an IBM business) for current forecasts.
- •**Local Information:** APIs for local guides, points of interest, restaurant reviews, and local event calendars.

2. Data Preprocessing:

•**Data Cleaning:** Implement routines to handle inconsistent data from various APIs (e.g., standardizing date formats, currencies). Manage missing values by either flagging them for the user or using logical defaults (e.g., mid-range price category if not specified).

•**Feature Engineering:** Transform raw data into meaningful features for the AI model. This involves:

- •Creating a structured user profile vector from their preferences and constraints.
- •Using Natural Language Processing (NLP) to extract keywords and sentiment from reviews of hotels and attractions.
- •Categorizing destinations and activities with relevant tags (e.g., family_friendly, nightlife, hiking).

3. Machine Learning Algorithm:

•**Core Model:** The central intelligence of the agent will be an **IBM Granite series Large Language Model (LLM)**, hosted and accessed via **IBM watsonx.ai**. The Granite model is chosen for its strong capabilities in understanding natural language, complex reasoning, and generating coherent, human-readable text.

•**Implementation Strategy (RAG):** A Retrieval-Augmented Generation (RAG) architecture will be used.

1. The user's query and profile are used to retrieve relevant, real-time data from the collected APIs (e.g., top 3 flights under $500, hotels with a pool).
2. This retrieved data, along with the user's original request, is compiled into a detailed prompt.
3. The prompt is sent to the **IBM Granite** model, which then reasons over the information to generate personalized recommendations, build itineraries, and create conversational responses.

•**Functions:** The model will be used for all core logic: suggesting destinations, justifying choices, creating day-by-day plans, and dynamically re-planning when alerts are triggered.

PROPOSED SOLUTION

4. Deployment:

•**Platform:** The entire solution will be deployed on the **IBM Cloud Lite** plan to meet the specified constraints.

•**Backend Services:** The application's backend logic, which orchestrates API calls and communicates with the watsonx.ai platform, will be deployed as serverless functions using **IBM Cloud Functions** or as an application on **IBM Code Engine**. This ensures scalability and cost-efficiency.

•**Database:** User profiles, preferences, and generated travel plans will be stored in a NoSQL database like **IBM Cloudant**, which is available on the Lite plan and provides robust, scalable data storage.

•**User Interface (UI):** A responsive web application or a chatbot will serve as the user interface. The frontend can be hosted on **IBM Cloud Object Storage** for static site hosting or deployed as part of the Code Engine application.

5. Evaluation:

•**Model Performance:** The performance of the Granite model will be assessed based on task completion and quality rather than traditional metrics like RMSE.

- **Constraint Adherence Rate:** Percentage of generated plans that successfully adhere to all user-specified constraints (budget, dates, etc.).
- **User Feedback Score:** Implement a rating system (e.g., 1-5 stars, thumbs up/down) for all major suggestions and final itineraries to gather direct user feedback on relevance and quality.
- **Conversion Rate:** Track how often recommendations for flights, hotels, or activities lead to a successful booking.

•**System Monitoring:** Use **IBM Cloud Log Analysis** to monitor application performance, API response times, and identify any operational errors.

•**Continuous Improvement:** The collected user feedback and performance data will be used to continuously refine the prompts sent to the Granite model and improve the data retrieval logic, thereby enhancing the accuracy and relevance of the agent over time.

6. Result:

The result is a fully functional, AI-powered Travel Planner Agent deployed on IBM Cloud. The agent intelligently assists users by creating personalized and optimized travel itineraries based on their unique preferences, budget, and real-time data. This transforms complex trip planning into a seamless, interactive, and enjoyable process, with the ability to manage bookings and adapt to changes on the go.

SYSTEM APPROACH

- System Requirements
- Functional Requirements:
  - **User Management:** The system must allow users to create and manage profiles containing their travel preferences, past trips, and budget information.
  - **Conversational Interface:** The agent needs to interact with users through a natural language interface (chatbot or voice) to gather trip requirements.
  - **Recommendation Engine:** It must provide personalized recommendations for destinations, flights, accommodations, and activities based on user profiles and real-time data.
  - **Itinerary Generation:** The system must be able to create logical, day-by-day travel plans that optimize for time, cost, and user preferences.
  - **Real-time Alerts:** It must monitor bookings and external conditions (like weather or flight status) to provide proactive alerts to the user.
  - **Dynamic Re-planning:** The agent must be capable of modifying the itinerary on the fly in response to disruptions or user requests.
- Non-Functional Requirements:
  - **Performance:** The system must have low latency to provide real-time, interactive responses to user queries.
  - **Availability:** It must be highly available (24/7) and reliable, especially when users are actively traveling.
  - **Scalability:** The architecture should scale automatically to handle a variable number of users without a degradation in performance.
  - **Security:** Must ensure the confidentiality and integrity of sensitive user data, including personal information and booking details.
- Technical Requirements:
  - **Cloud Platform:** The entire solution must be deployed on **IBM Cloud** using services available under the **Lite** plan.
  - **Core AI Model:** The system's reasoning and generation capabilities must be powered by an **IBM Granite** Large Language Model hosted on **watsonx.ai**.

SYSTEM APPROACH

Libraries Required to Build the Model

The backend of the Travel Planner Agent would primarily be built using Python. The key libraries required include:

- **Flask or FastAPI**: To build the web server and expose API endpoints that the user interface will interact with.
- **requests**: Essential for making HTTP calls to third-party APIs for flights, hotels, weather, maps, and local guides.
- **ibm-watson-machine-learning**: The official IBM SDK to authenticate and interact with the **IBM Granite LLM** on the watsonx.ai platform.
- **ibm-cloud-sdk-core**: A core library for interacting with other IBM Cloud services, such as IBM Cloudant or Code Engine.

ALGORITHM & DEPLOYMENT

Algorithm Selection

- The core of this system utilizes a **Large Language Model (LLM)**, specifically the **IBM Granite** model, operating within a **Retrieval-Augmented Generation (RAG)** framework.
- This approach is chosen over traditional predictive models (like ARIMA or LSTM) because the task is not to predict a numerical value but to **understand, reason, and generate** complex, personalized travel plans. An LLM excels at processing natural language requests, weighing multiple constraints (budget, user style, time), and creating coherent, human-readable text. The RAG architecture is essential as it grounds the LLM's powerful generation capabilities in factual, real-time data from external APIs, preventing outdated or incorrect suggestions.

ALGORITHM & DEPLOYMENT

Data Input

- The algorithm does not use a static dataset for training in the traditional sense. Instead, at the moment of prediction, it uses a dynamic set of inputs assembled into a prompt:
- **User-Provided Data:**
  - Natural language queries (e.g., "Find me a 4-day adventurous trip to Himachal Pradesh next month").
  - User profile information (travel style, interests, past trip feedback).
  - Specific constraints (exact dates, budget limits, accessibility needs).
- **Real-time Retrieved Data:**
  - Live flight and transport schedules, pricing, and availability.
  - Current hotel and accommodation options, including ratings and costs.
  - Real-time weather forecasts for potential destinations.
  - Information on local attractions, events, and hours of operation.

ALGORITHM & DEPLOYMENT

Training Process

- The "training" for this LLM-based system focuses on adaptation and prompt engineering, not on training a model from scratch.
- **Base Model:** We use the **IBM Granite** model, which has already been pre-trained by IBM on a vast corpus of diverse text data. This pre-training provides its foundational language understanding and reasoning abilities.
- **Prompt Engineering:** The primary development effort involves **designing and refining prompts**. This is an iterative process of creating a structured template that effectively combines the user's query with the real-time retrieved data. The goal is to engineer a prompt that consistently guides the Granite model to produce accurate, relevant, and well-structured travel plans.
- **Fine-Tuning (Optional):** For further specialization, the base Granite model can be fine-tuned on **IBM watsonx.ai** using a curated dataset of high-quality examples (e.g., sample travel requests paired with ideal itinerary outputs). This helps the model better adapt its tone and formatting specifically for travel planning.

ALGORITHM & DEPLOYMENT

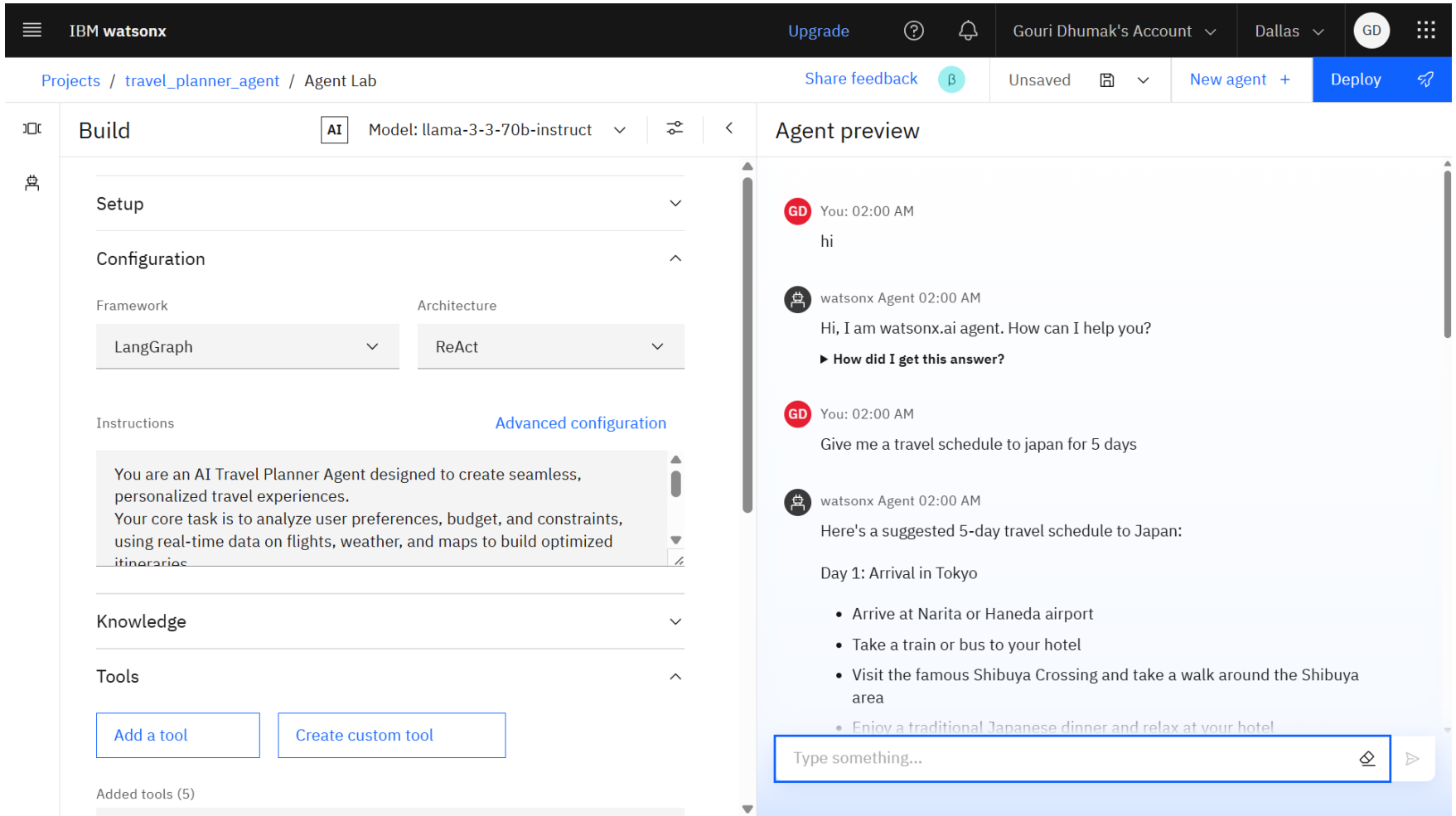
Prediction Process

- The prediction (or generation) process is executed in real-time for each user request and follows these steps:
- **Parse Query:** The system receives the user's request (e.g., "plan my weekend").
- **Retrieve Context:** The backend logic makes concurrent API calls to relevant travel, weather, and local guide services to fetch up-to-the-minute data based on the user's query and profile.
- **Construct Prompt:** The retrieved data is structured and combined with the user's original query and profile information into a single, comprehensive prompt.
- **Generate Response:** This prompt is sent to the hosted **IBM Granite LLM**. The model processes the entire context and generates a personalized response, which could be a destination suggestion, a full itinerary, or a transport recommendation.
- **Deliver Result:** The generated text is formatted and delivered back to the user through the application's interface.

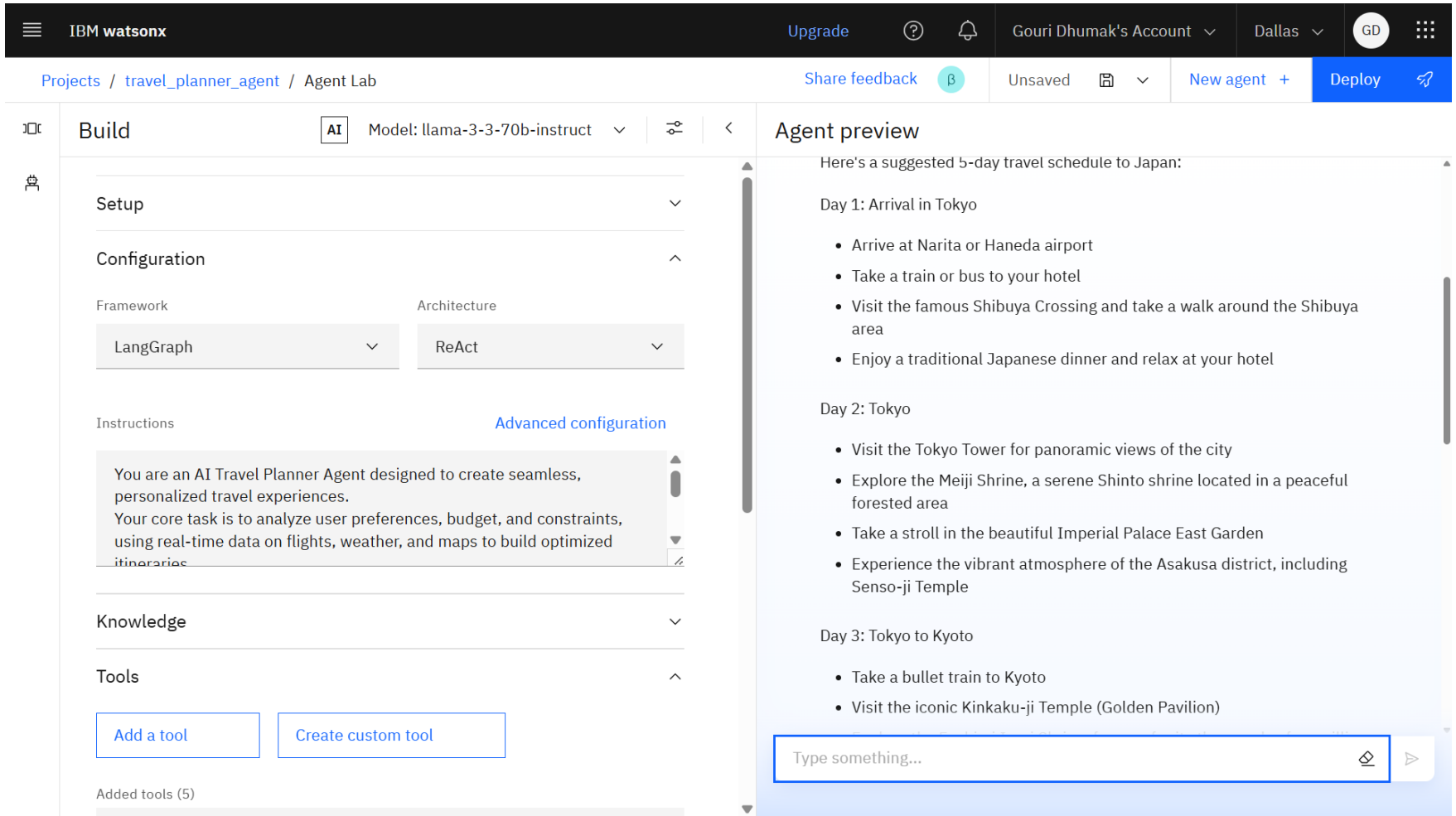
RESULT

- The successful implementation of this project yields a sophisticated, AI-powered Travel Planner Agent, fully deployed and operational on the **IBM Cloud** platform. This result represents a significant advancement over traditional travel booking websites by delivering a holistic and intelligent end-to-end travel management solution.
- **Personalized & Dynamic Itinerary Generation**
- The primary outcome is the agent's ability to produce highly personalized and dynamic travel itineraries. The agent engages in a natural language dialogue with the user to deeply understand their unique preferences and synthesizes this profile with real-time data to create a truly bespoke travel experience, not just a list of bookings.
- **Seamless End-to-End Trip Management**
- The agent's functionality extends far beyond initial planning. It acts as a central hub for the entire trip, consolidating all bookings and serving as a proactive travel companion. It monitors for disruptions, sends real-time alerts, and can dynamically re-optimize the day's schedule on the spot, handling logistical burdens so users can focus on their journey.

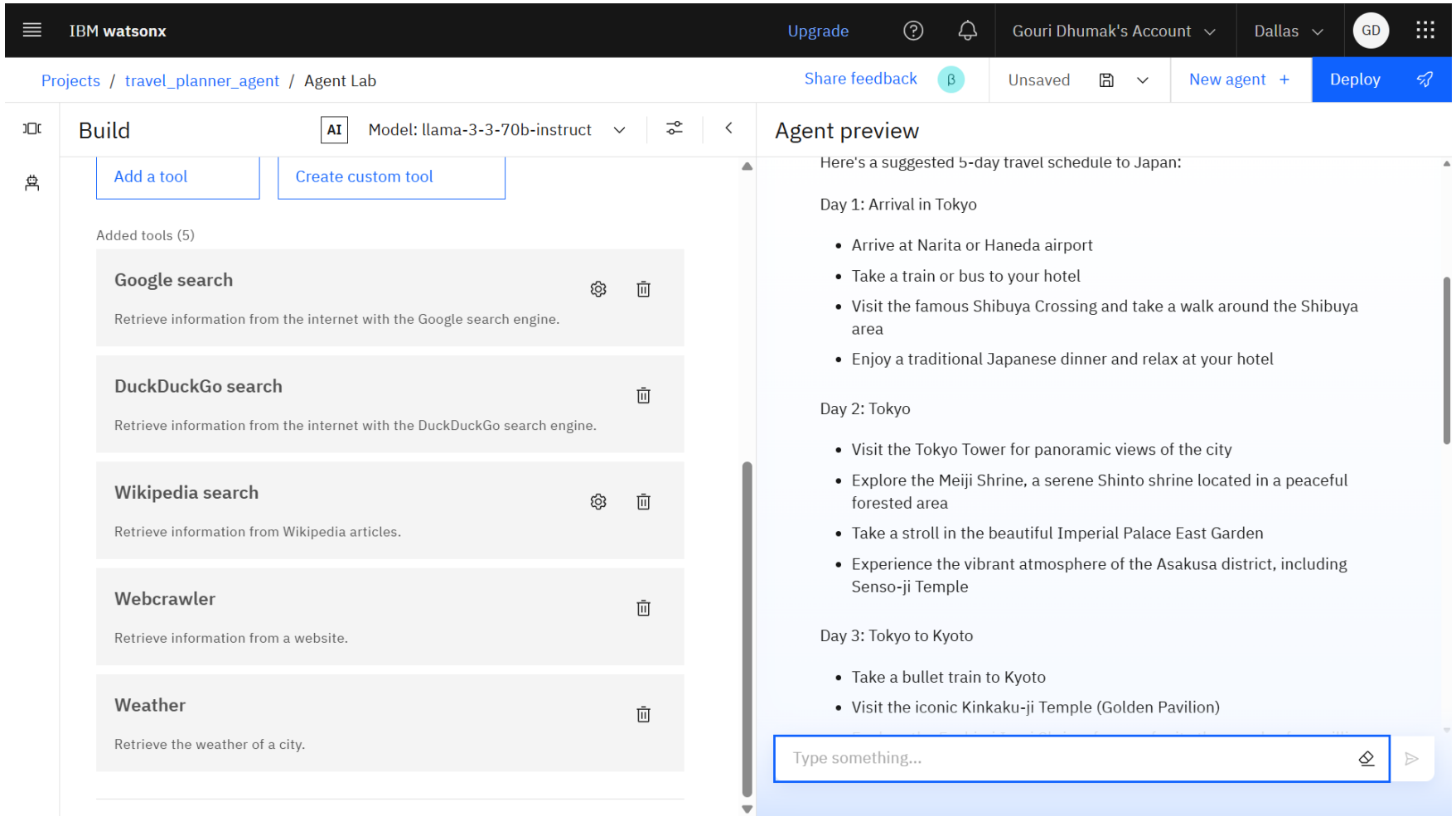
RESULT



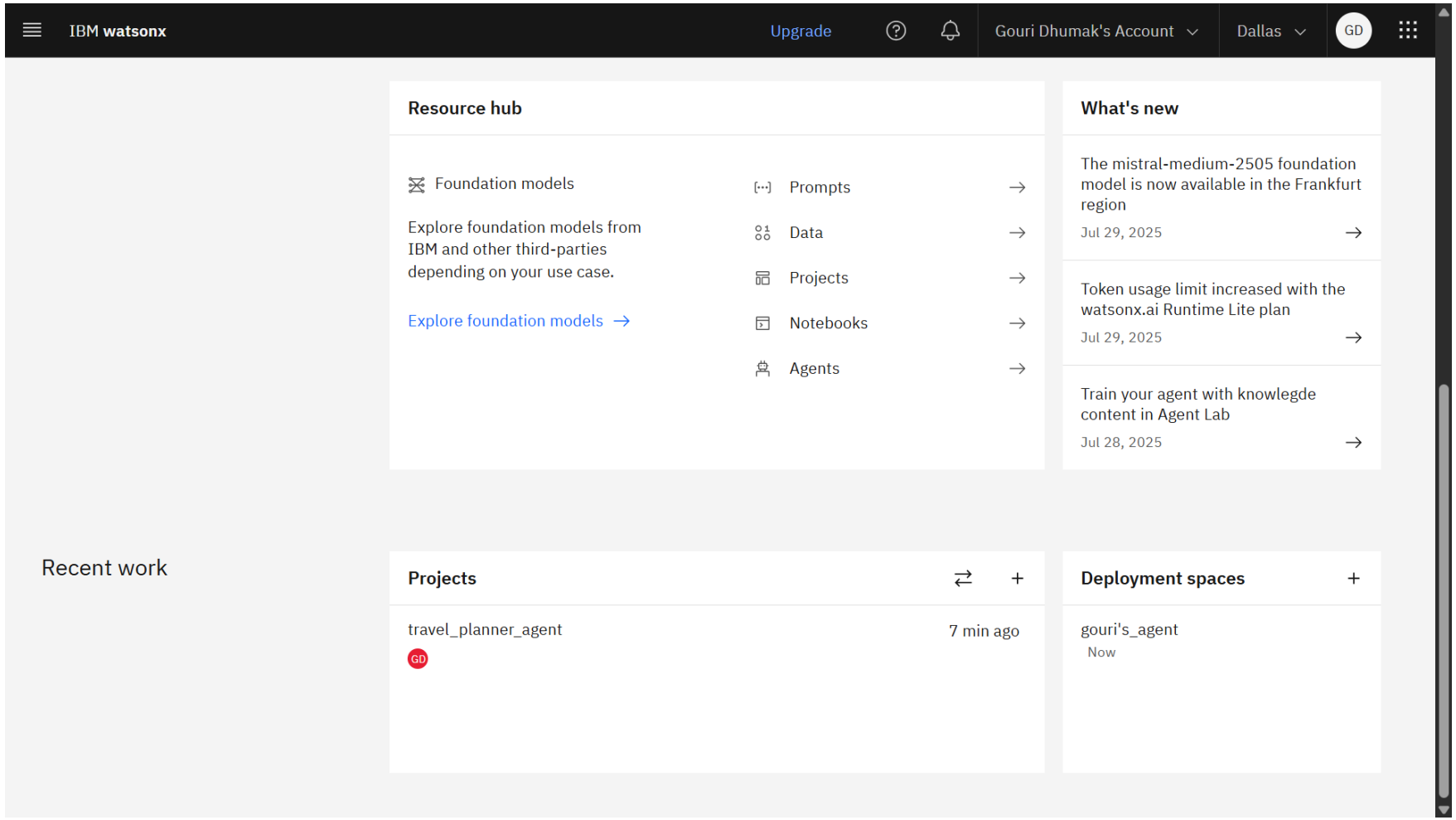
RESULT



RESULT

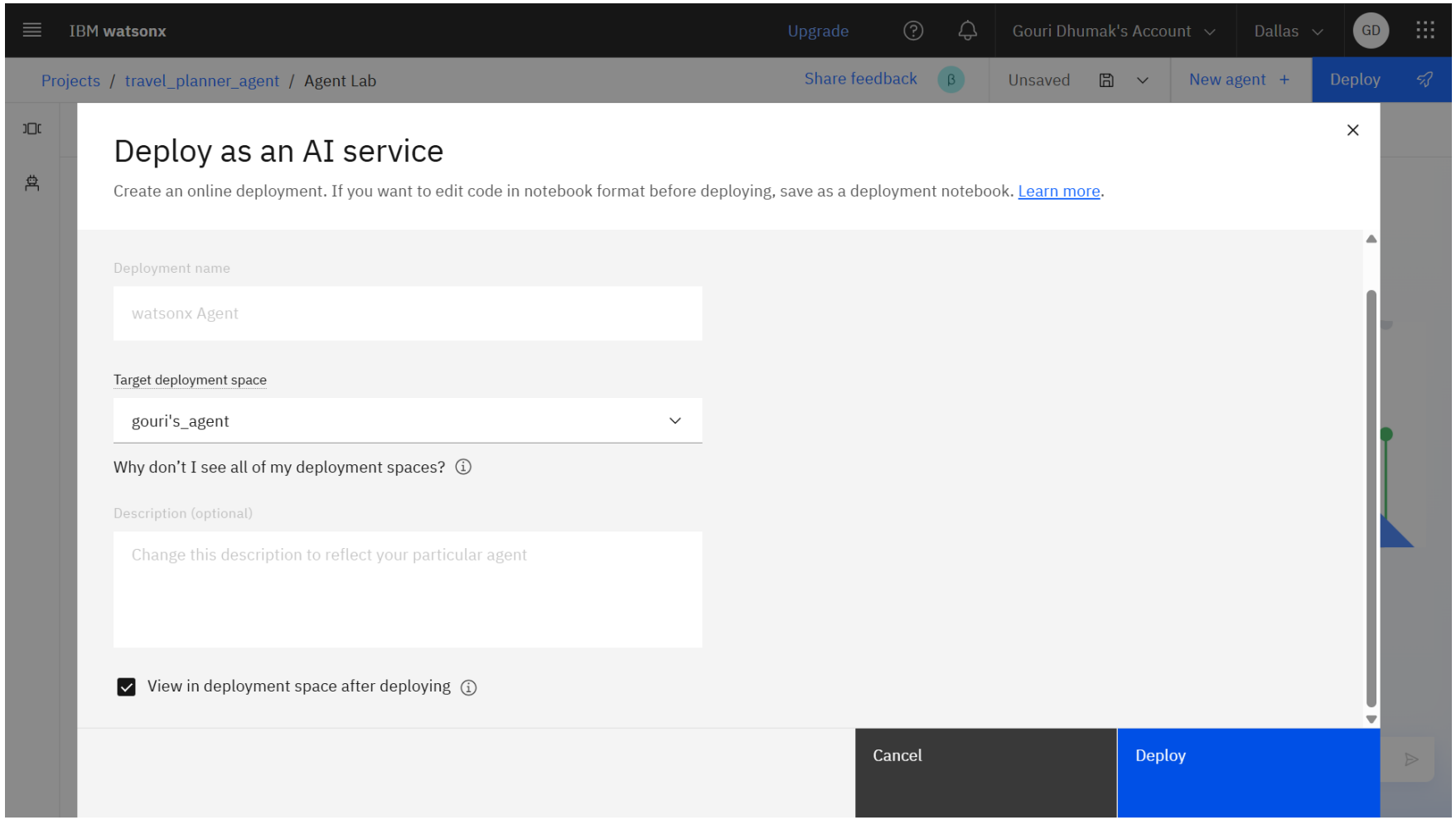


RESULT



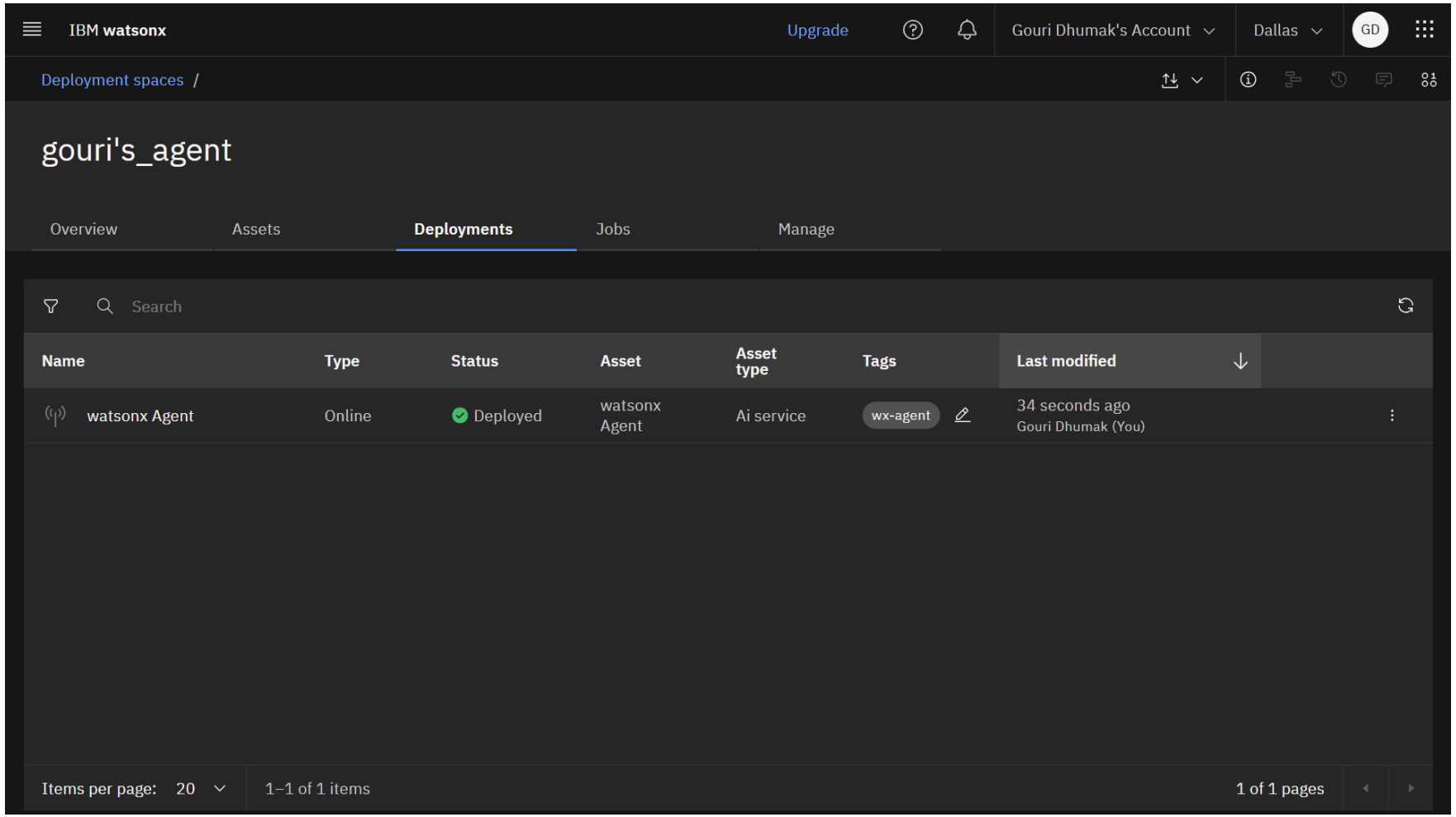
edunet foundation

RESULT

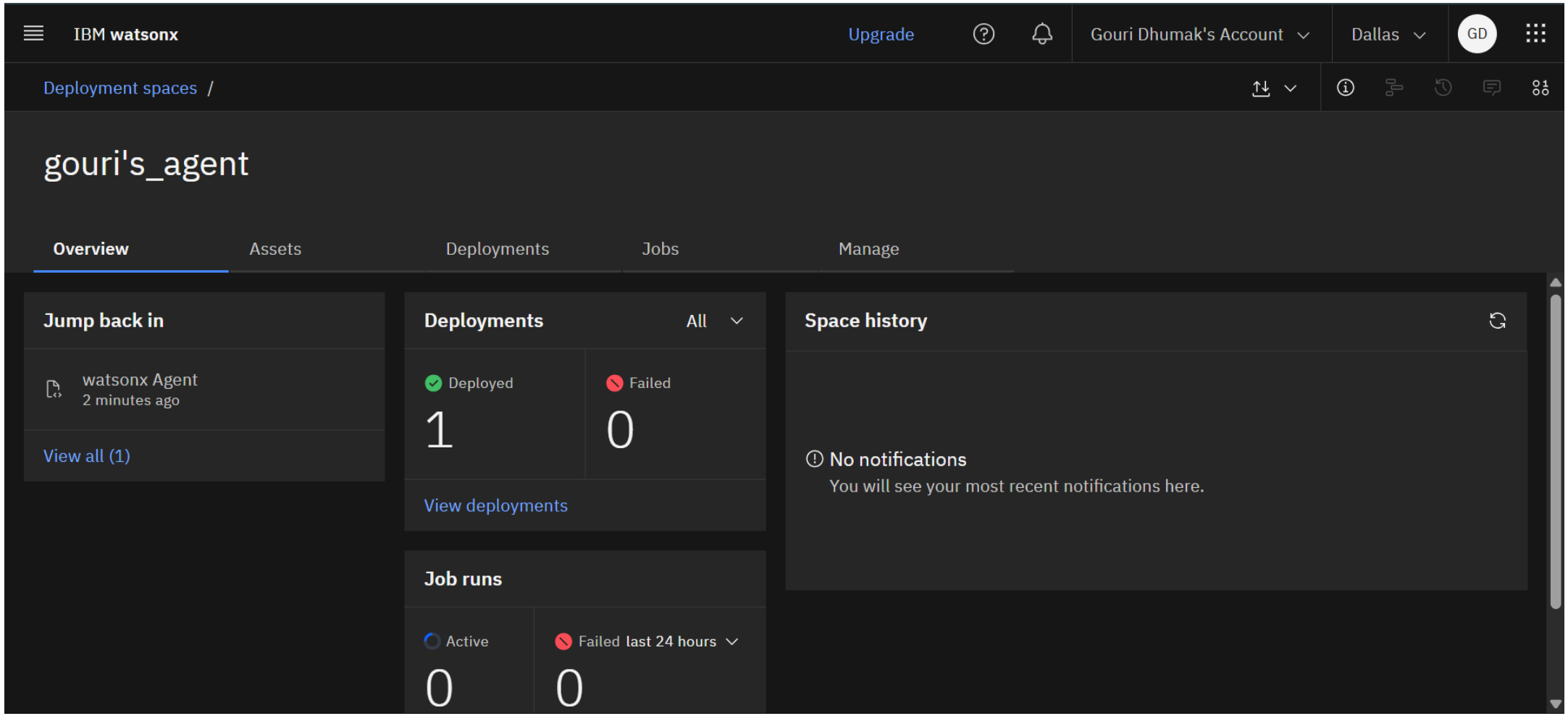


edunet foundation

RESULT



RESULT



RESULT

GITHUB REPOSITORY LINK => https://github.com/Gouridhumak/IBM_TRAVEL_PLANNER

CONCLUSION

The Travel Planner Agent project outlined in this proposal represents a strategic leap forward from conventional travel booking tools to a new paradigm of intelligent, personalized travel management. By leveraging the advanced reasoning capabilities of the **IBM Granite Large Language Model** within a robust, real-time data framework on **IBM Cloud**, this solution directly confronts the core friction points of modern travel planning: information overload, logistical complexity, and the stress of managing dynamic situations. The proposed system is designed not merely as a search engine, but as a comprehensive, end-to-end travel companion. Its ability to understand nuanced user preferences, synthesize vast amounts of live data, and generate truly personalized itineraries sets a new standard for user-centric design. Furthermore, its capacity for real-time monitoring and dynamic re-planning ensures that travelers are supported throughout their entire journey, transforming unforeseen disruptions from crises into manageable inconveniences.

In summary, the Travel Planner Agent offers a powerful value proposition: it saves users invaluable time, reduces stress, and ultimately fosters a more enjoyable and seamless travel experience. This project promises to deliver an innovative and commercially viable solution that redefines how people plan, book, and experience travel, positioning itself at the forefront of the AI-driven service industry.

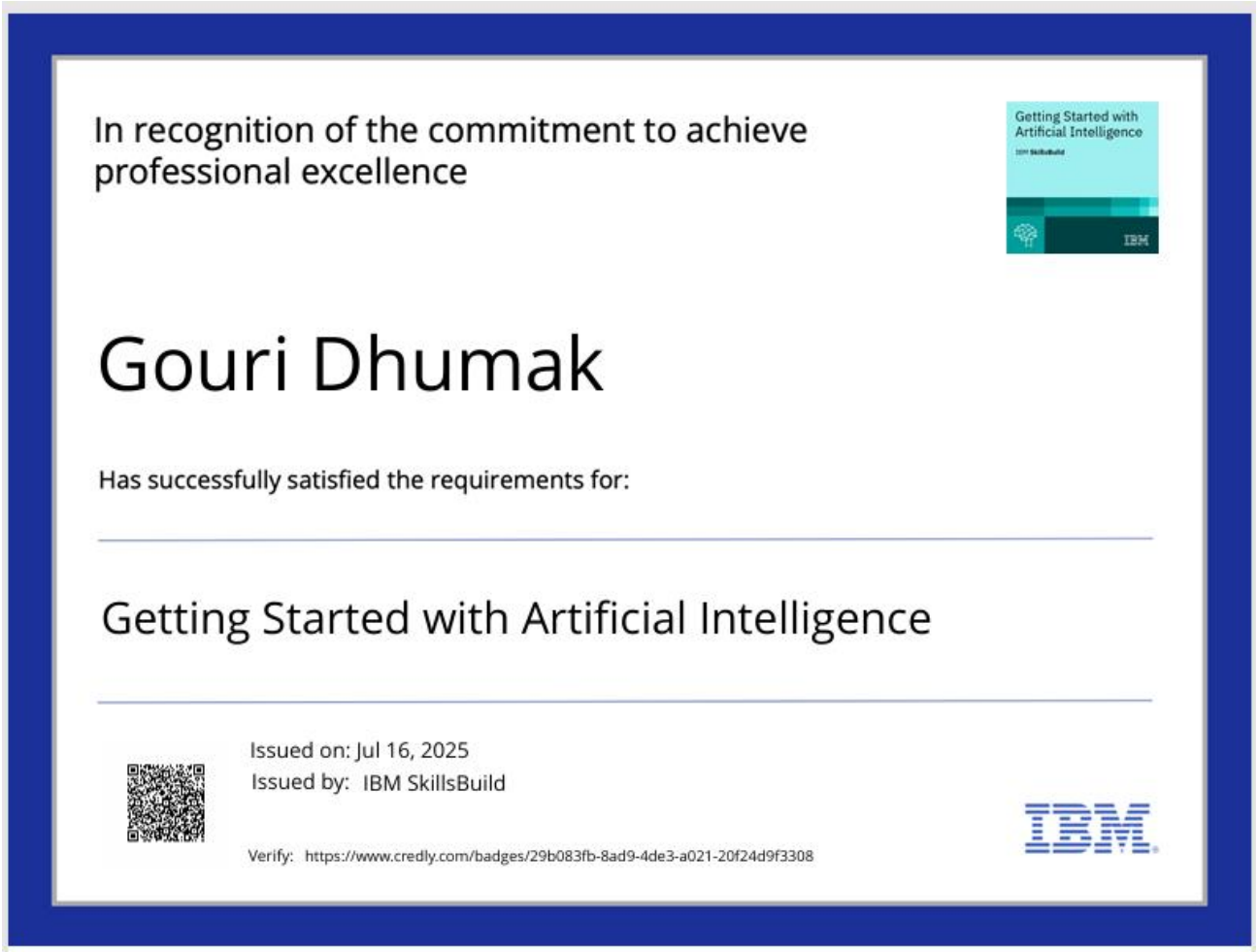
FUTURE SCOPE

- The future development of the Travel Planner Agent will focus on evolving it from a powerful planning tool into an indispensable, proactive travel partner. Key areas for expansion include:
- **Proactive and Predictive Planning:** The agent will anticipate user needs by suggesting trips based on past behavior and using predictive analytics to warn of potential flight delays or issues before they occur.
- **Seamless Digital Integration:** Deepen integration with users' digital lives by fully automating the booking process and syncing itineraries with personal calendars, emails, and expense tracking applications.
- **Collaborative and Group Travel:** Introduce a dedicated module for group planning that allows for shared itineraries, voting on activities, and seamless cost splitting among travelers.
- **Expanded Service Offerings:** Broaden the platform's reach by launching a specialized version for corporate travel and enriching leisure travel with bookable, hyper-local experiences and unique tour packages.
- **Advanced Multimodal AI:** Enhance the user interface to support sophisticated voice commands and visual search, allowing users to plan trips more naturally by speaking or using images.

REFERENCES

•**IBM Cloud Platform & Lite Account**
•IBM. (2024). *IBM Cloud Free Tier*. IBM Cloud Documentation. Retrieved from https://www.ibm.com/cloud/free
•IBM. (2024). *What is Cloud Computing?* IBM Cloud Learn Hub. Retrieved from https://www.ibm.com/cloud/learn/cloud-computing
•**watsonx.ai Platform and Granite Models**
•IBM. (2024). *IBM watsonx.ai*. IBM watsonx Product Page. Retrieved from https://www.ibm.com/watsonx/watsonx.ai
•IBM. (2024). *IBM Granite series models*. IBM watsonx.ai Documentation. Retrieved from https://www.ibm.com/docs/en/watsonx-as-a-service?topic=models-granite-series
•IBM Developer. (2023). *Introduction to foundation models*. IBM Developer Learning Path. Retrieved from https://developer.ibm.com/learningpaths/get-started-watsonx-ai/introduction-to-foundation-models/
•**Retrieval-Augmented Generation (RAG) on watsonx**
•Singh, A. (2023). *"Generative AI with RAG on IBM watsonx.ai"*. IBM Technology Blog. Retrieved from https://www.ibm.com/blogs/research/research-rag-watsonx/
•IBM. (2024). *Tutorial: Applying retrieval-augmented generation to a foundation model*. IBM watsonx.ai Documentation. Retrieved from https://www.ibm.com/docs/en/watsonx-as-a-service?topic=notebooks-tutorial-applying-retrieval-augmented-generation-foundation-model
•**Deployment and Infrastructure Services**
•IBM. (2024). *IBM Code Engine*. IBM Cloud Product Page. Retrieved from https://www.ibm.com/cloud/code-engine
•IBM. (2024). *IBM Cloud Functions*. IBM Cloud Product Page. Retrieved from https://www.ibm.com/cloud/functions
•IBM. (2024). *IBM Cloudant*. IBM Cloud Product Page. Retrieved from https://www.ibm.com/cloud/cloudant
•**SDKs and Libraries**
•IBM. (2024). *Watson Machine Learning Python SDK*. Python Package Index (PyPI). Retrieved from https://pypi.org/project/ibm-watson-machine-learning/
•IBM. (2024). *IBM Cloud SDK for Python*. GitHub Repository. Retrieved from https://github.com/IBM/python-sdk-core

IBM CERTIFICATIONS



IBM CERTIFICATIONS

In recognition of the commitment to achieve professional excellence

Gouri Dhumak

Has successfully satisfied the requirements for:

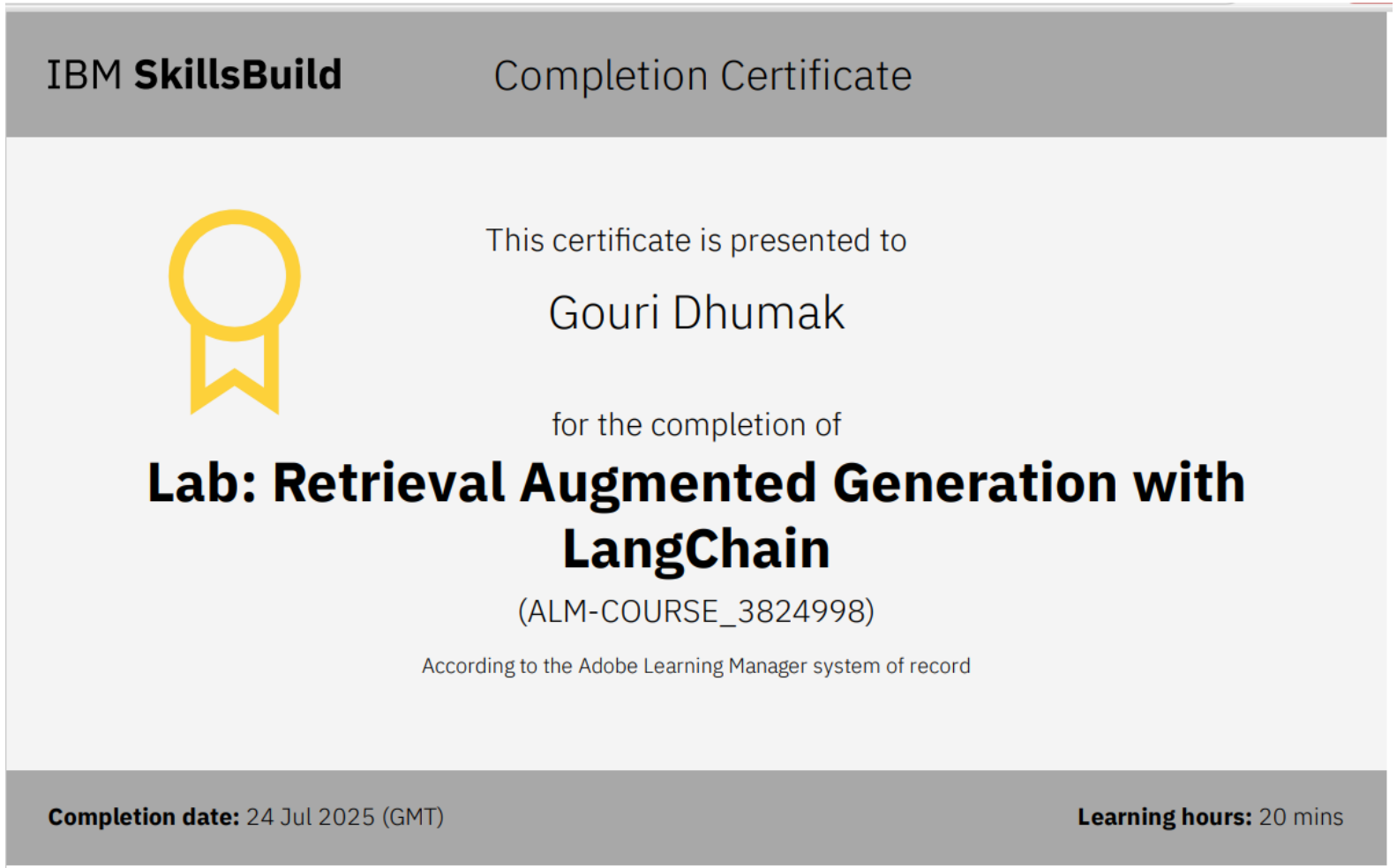
Journey to Cloud: Envisioning Your Solution

Issued on: Jul 20, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/ae21e621-05dd-4444-9491-2f3aef739230

IBM

IBM CERTIFICATIONS



IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Gouri Dhumak

for the completion of

Lab: Retrieval Augmented Generation with LangChain

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

Completion date: 24 Jul 2025 (GMT) **Learning hours:** 20 mins

THANK YOU

edunet foundation