

Running Large Language Models (LLMs) on IBM i

Adam Shedivy, IBM

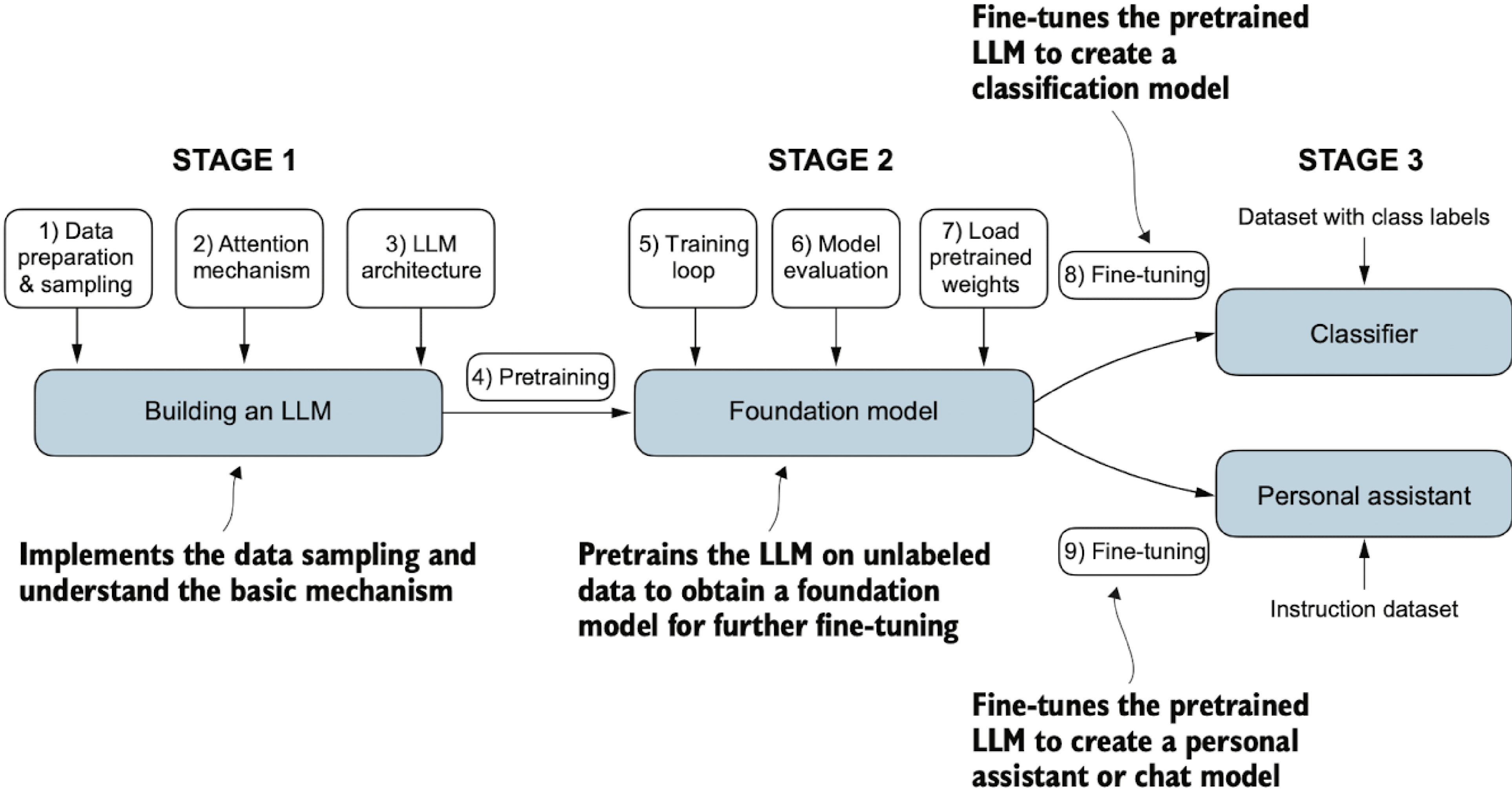
Software Developer, AI solutions for IBM i

adam.shedivy@ibm.com

Goals for today

- Understand how LLMs work
- Run an LLM on IBM i
- Integrate with Db2 for i workloads

Overview



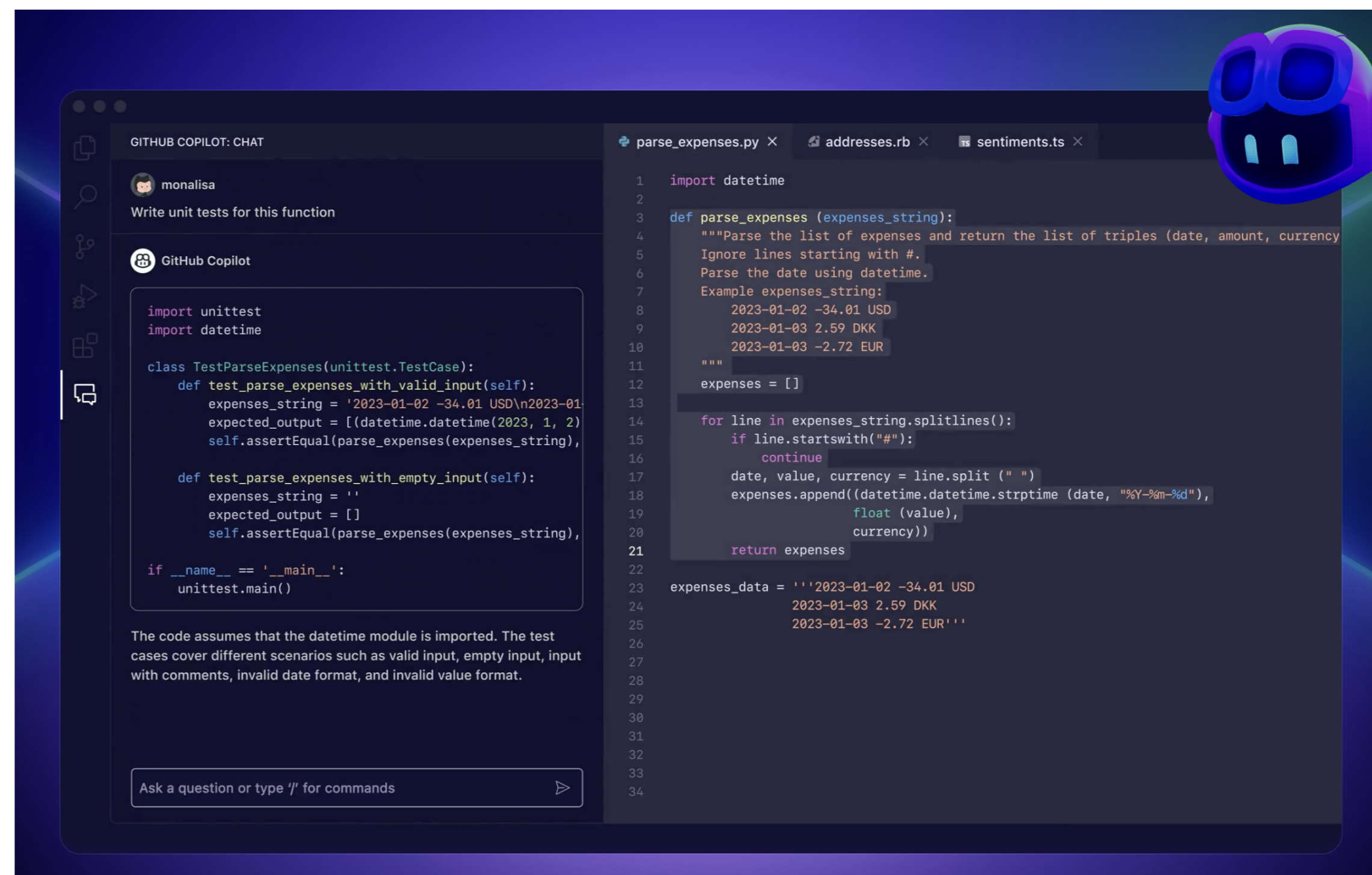
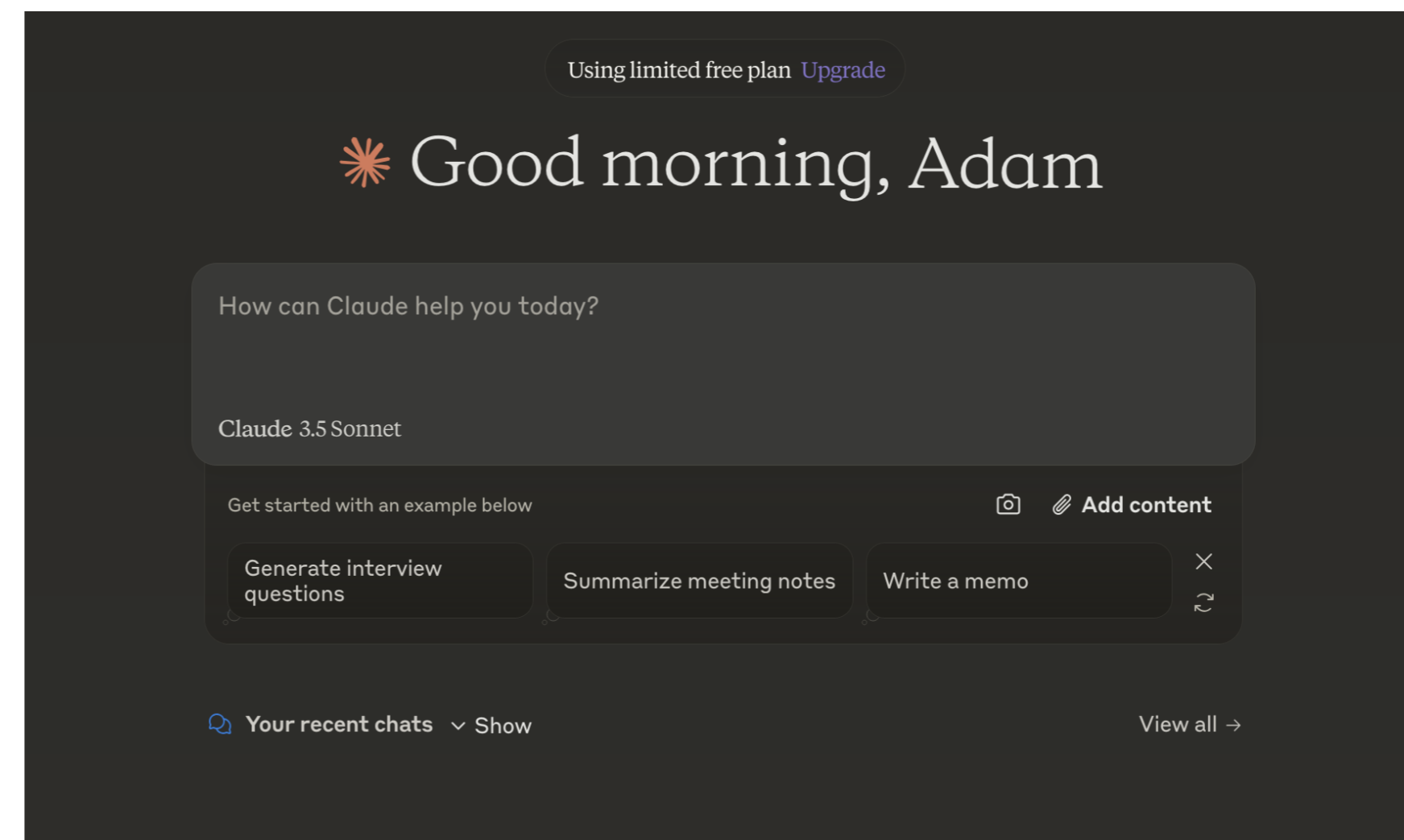
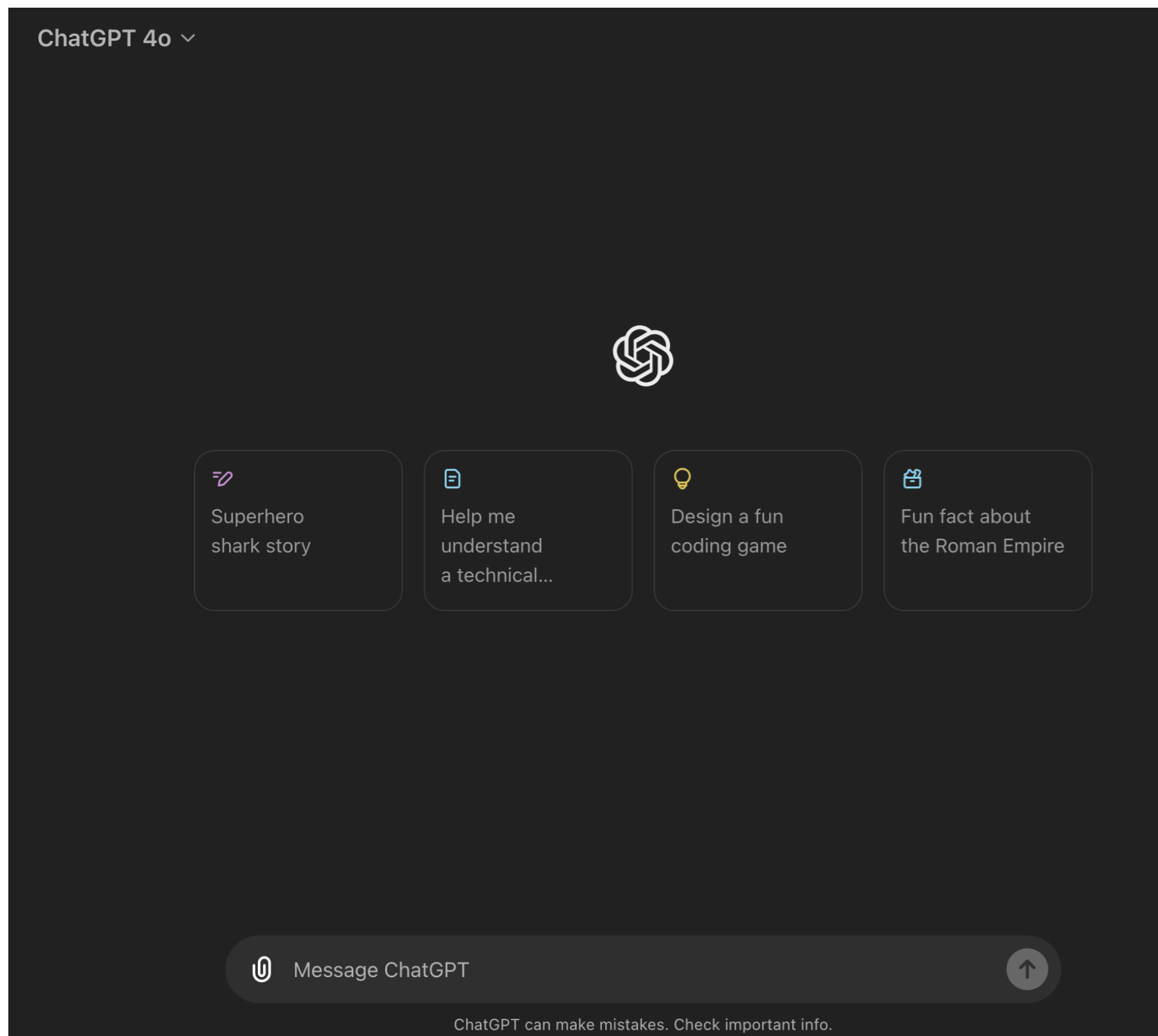
Code

- Full Demo: <https://github.com/ajshedivy/LLMs-from-scratch/tree/adam/demo>

The screenshot displays a JupyterLab environment. On the left, a file browser shows a directory structure with files like 'DEMO.md', 'gpt_visual_aten...', 'gpt_visual_train.py', 'gpt_visual.py', 'loss-plot.pdf', 'README.md', 'the-verdict.txt', and 'utils.py'. The main area shows a notebook titled 'Build an LLM from Scratch - Complete Tutorial' by Adam Shedivy, based on Sebastian Raschka's 'Build a Large Language Model (From Scratch)'. The notebook content includes 'Learning Objectives' and 'What You'll Build & Learn' sections. The 'What You'll Build & Learn' section is divided into three numbered parts: 1. Text Data Processing, 2. Attention Mechanisms, and 3. GPT Architecture Implementation. The 'Text Data Processing' section lists: Tokenization (converting raw text into model-processable tokens), Embeddings (transforming discrete tokens into continuous vector representations), Byte-Pair Encoding (BPE) (handling out-of-vocabulary words efficiently), and Data Loading (creating training datasets with sliding window techniques). The 'Attention Mechanisms' section lists: Contextual Understanding (how models focus on relevant information), Query-Key-Value Systems (the mathematical foundation of attention), Multi-Head Attention (parallel processing of different relationship types), and Visualization Tools (interactive exploration of attention patterns). The 'GPT Architecture Implementation' section is partially visible at the bottom.

Local (PC) AI Inference

What tools exist?



How to use LLMs locally?

Lots of ways to run a LLM locally

– Llama.cpp, Ollama, transformers (Hugging Face), LMStudio, LangChain

The logo for LLaMA C++ features the text "LLaMA" in white and "C++" in orange on a black background.

LM Studio



Running LLMs on Power

- Run LLMs locally using Ollama



Performance Comparison 🔥 😎

Prompt: Why is the sky blue?

P10



```
1 total duration:      2.079801045s
2 load duration:       8.568136ms
3 prompt eval count:  13 token(s)
4 prompt eval duration: 137.423ms
5 prompt eval rate:   94.60 tokens/s
6 eval count:         78 token(s)
7 eval duration:      1.883978s
8 eval rate:          41.40 tokens/s
```

P9



```
1 total duration:     27.506958826s
2 load duration:      18.1912ms
3 prompt eval count:  13 token(s)
4 prompt eval duration: 3.922594s
5 prompt eval rate:   3.31 tokens/s
6 eval count:         79 token(s)
7 eval duration:     23.52334s
8 eval rate:          3.36 tokens/s
```

Running LLMs on IBM i

Running LLMs on IBM i

```
-bash-5.2$ ./llama.cpp/build_cpu/bin/llama-run ~/models/Llama-3.2-1B-Instruct-f32-be-Q4_0.gguf -t 16  
>
```



- Llama.cpp is supported on IBM i
- Llama.cpp is a light-weight pure C/C++ library for LLM inference
- Offers built in CLI (shown in GIF), Server, chat UI

Blog Post: <https://ibm.biz/llamacpp-ibmi>

Advantages

- Data Privacy & Security
- Performance and Cost
- Flexibility

Drawbacks

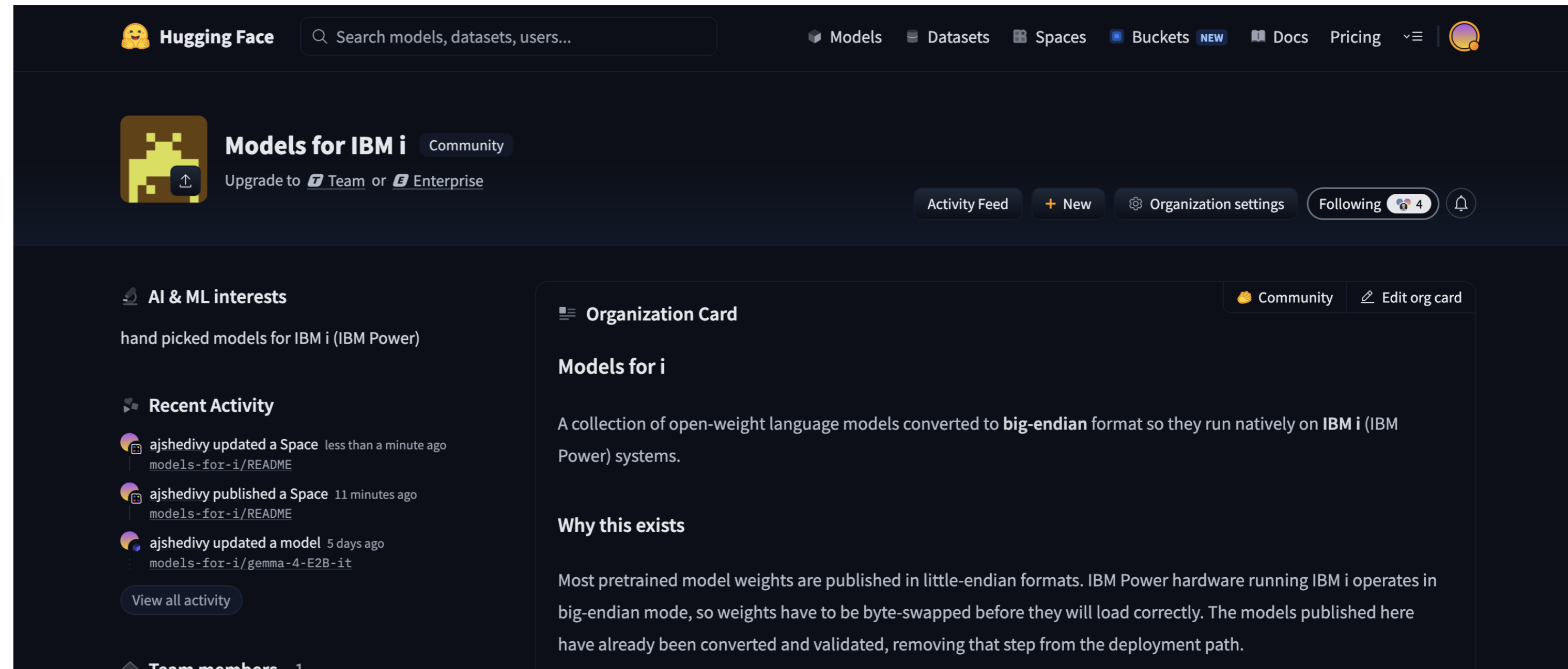
- Complex setup and maintenance
- Limited Scalability
- Model Updates and latest features

Bigger isn't
always better.



How this works

1. Build/Install Llama.cpp (CPU) on IBM i
2. Transfer model to IBM i
3. Run Llama.cpp

The logo for LLaMA C++ is displayed in white and orange text on a dark background. The word "LLaMA" is in white, and "C++" is in orange.A screenshot of the Hugging Face website showing the 'Models for IBM i' organization page. The page features a dark theme with a navigation bar at the top containing 'Hugging Face', a search bar, and links for 'Models', 'Datasets', 'Spaces', 'Buckets', 'Docs', and 'Pricing'. The organization's profile includes a yellow pixelated avatar, the name 'Models for IBM i', and options to 'Upgrade to Team or Enterprise'. Below the profile, there are sections for 'AI & ML interests' (hand picked models for IBM i), 'Recent Activity' (listing updates to a Space and a model by user 'ajshdivy'), and an 'Organization Card' with a description: 'A collection of open-weight language models converted to big-endian format so they run natively on IBM i (IBM Power) systems.' and a 'Why this exists' section explaining the need for big-endian conversion on IBM Power hardware.

Build/Install Llama.cpp

```
yum install llama-cpp
```

Or, if you're a weirdo....

```
cmake -B build \  
-DCMAKE_AR="/QOpenSys/pkg/bin/gcc-ar-12" \  
-DCMAKE_RANLIB="/QOpenSys/usr/bin/ranlib" \  
-DCMAKE_C_COMPILER="/QOpenSys/pkg/bin/gcc-12" \  
-DCMAKE_CXX_COMPILER="/QOpenSys/pkg/bin/g++-12" \  
-DCMAKE_CXX_FLAGS="-pthread" \  
-DCMAKE_EXE_LINKER_FLAGS="-pthread -lutil" \  
-DCMAKE_CXX_ARCHIVE_CREATE="<CMAKE_AR> -X64 qc <TARGET> <LINK_FLAGS> <OBJECTS>" \  
-DCMAKE_CXX_ARCHIVE_APPEND="<CMAKE_AR> -X64 q <TARGET> <LINK_FLAGS> <OBJECTS>" \  
-DCMAKE_CXX_ARCHIVE_FINISH="<CMAKE_RANLIB> <TARGET>" \  
-DCMAKE_BUILD_TYPE=Release \  
-DGGML_LLAMAFILE=OFF
```

Download a model

The screenshot shows the Hugging Face website interface. At the top, there is a search bar with the text "Search models, datasets, users...". Navigation links include "Models", "Datasets", "Spaces", "Buckets NEW", "Docs", and "Pricing". The main content area features a profile for "Models for IBM i" with a "Community" tag and a button to "Upgrade to Team or Enterprise". Below this, there is an "Organization Card" for "Models for i" with the following text: "A collection of open-weight language models converted to **big-endian** format so they run natively on **IBM i** (IBM Power) systems." and "Why this exists: Most pretrained model weights are published in little-endian formats. IBM Power hardware running IBM i operates in big-endian mode, so weights have to be byte-swapped before they will load correctly. The models published here have already been converted and validated, removing that step from the deployment path." To the left of the organization card, there is a "Recent Activity" section listing updates to a Space and a model by user "ajshedivy".

<https://huggingface.co/models-for-i>

Llama-server

```
adamshedivy — -zsh — 89x16
(base) ~
$ curl -s -X POST http://172.20.25.3:8080/completions \
  -H "Content-Type: application/json" \
  -d '{
    "prompt": "In a quiet village, an unusual event occurred: ",
    "stream": false,
    "n_predict": 60,
    "temperature": 0.8
  }' \
| jq .
```

Terminal window showing two panes:

- ssh pane:**

```
-bash-5.2$ ./llama.cpp/build_cpu/bin/llama-server -m models/Llama-3.2-1B-Instruct-f32-be-Q4_0.gguf -t 16 --host 0.0.0.0 -v
```
- zsh pane:**

```
(base) ~
$ curl -s -X POST http://172.20.25.3:8080/completions \
  -H "Content-Type: application/json" \
  -d '{
    "prompt": "Explain the difference between a stack and a queue in two sentences.",
    "stream": false,
    "temperature": 0.3
  }' \
| jq -cr '.content'
```

ssh [Warning Icon] ×

```

kens = 79, truncated = 0
srv update_slots: decoding batch, n_tokens = 1
set_embeddings: value = 0
clear_adapter_lora: call
slot process_token: id 0 | task 78 | stopped by limit, n_decoded = 50, n_predict = 50
slot process_token: id 0 | task 78 | n_decoded = 50, n_remaining = 0, next token: 933 ']'

slot release: id 0 | task 78 | stop processing: n_past = 79, truncated = 0
slot print_timing: id 0 | task 78 |
prompt eval time = 1009.90 ms / 29 tokens ( 34.82 ms per token, 28.72 tokens per second)
eval time = 7275.74 ms / 50 tokens ( 145.51 ms per token, 6.87 tokens per second)
total time = 8285.64 ms / 79 tokens
srv send: sending result for task id = 78
srv send: task id = 78 pushed to result queue
srv update_slots: run slots completed
que start_loop: waiting for new tasks
que start_loop: processing new tasks
que start_loop: processing task, id = 128
que start_loop: update slots
srv update_slots: all slots are idle
que start_loop: waiting for new tasks
srv remove_waiter: remove task 78 from waiting list. current waiting = 1 (before remove)
srv log_server_r: request: POST /completions 172.29.96.122 200
srv log_server_r: request: {
  "prompt": "Write a Python function called fib_iter that returns the nth Fibonacci number using an iterative approach. Do not explain the function, just return the python code",
  "stream": false,
  "n_predict": 50,
  "temperature": 0
}
srv log_server_r: response: {"index":0,"content":".\n\n```\npython\ndef fib_iter(n):\n    fib = [0, 1]\n    for i in range(2, n):\n        fib.append(fib[i-1] + fib[i-2])\n    return fib[n-1]\n", "tokens": [], "id_slot": 0, "stop": true, "model": "gpt-3.5-turbo", "tokens_predicted": 50, "tokens_evaluated": 30, "generation_settings": {"n_predict": 50, "seed": 4294967295, "temperature": 0.0, "dynatemp_range": 0.0, "dynatemp_exponent": 1.0, "top_k": 40, "top_p": 0.949999988079071, "min_p": 0.05000000074505806, "xtc_probability": 0.0, "xtc_threshold": 0.10000000149011612, "typical_p": 1.0, "repeat_last_n": 64, "repeat_penalty": 1.0, "presence_penalty": 0.0, "frequency_penalty": 0.0, "dry_multiplier": 0.0, "dry_base": 1.75, "dry_allowed_length": 2, "dry_penalty_last_n": 4096, "dry_sequence_breakers": ["\n", ":", "\n", "*"], "mirostat": 0, "mirostat_tau": 5.0, "mirostat_eta": 0.10000000149011612, "stop": [], "max_tokens": 50, "n_keep": 0, "n_discard": 0, "ignore_eos": false, "stream": false, "logit_bias": [], "n_probs": 0, "min_keep": 0, "grammar": "", "grammar_lazy": false, "grammar_triggers": [], "preserved_tokens": [], "chat_format": "Content-only", "samplers": ["penalties", "dry", "top_k", "typ_p", "top_p", "min_p", "xtc", "temperature"], "speculative.n_max": 16, "speculative.n_min": 0, "speculative.p_min": 0.75, "timings_per_token": false, "post_sampling_probs": false, "lora": []}, "prompt": "<|begin_of_text|>Write a Python function called fib_iter that returns the nth Fibonacci number using an iterative approach. Do not explain the function, just return the python code", "has_new_line": true, "truncated": false, "stop_type": "limit", "stopping_word": "", "tokens_cached": 79, "timings": {"prompt_n": 29, "prompt_ms": 1009.903, "prompt_per_token_ms": 34.824241379310344, "prompt_per_second": 28.71562912477733, "predicted_n": 50, "predicted_ms": 7275.741, "predicted_per_token_ms": 145.51482, "predicted_per_second": 6.872152266002872}}

```

zsh [Warning Icon] ×

```

(base) ~
$ curl -s -X POST http://172.20.25.3:8080/completions \
-H "Content-Type: application/json" \
-d '{
  "prompt": "In a quiet village, an unusual event occurred: ",
  "stream": false,
  "n_predict": 60,
  "temperature": 0.8
}' \
| jq -cr '.content'

```

Conversations

llama.cpp



+ New conversation

Send a message to start

Overview

- You can run LLMs on IBM i using **Llama.cpp**
- Leverage Power hardware for running LLMs on CPU
- Build From source: <https://ibm.biz/llamacpp-ibmi>

Embedded AI Transactions

Embedding AI into a Db2 Transaction

Bringing AI into the transaction

Goal

- Embed AI inside a database transaction

Architecture

- Integrate LLM Provider REST APIs

Application

- Data summarization
- Data classification
- Data analysis

Today

- Watsonx, Ollama, OpenAI-Compatible support
- Manual Installation, basic usage

Soon...

- Other LLM providers
- Improved usage and support
- Better docs
- Integrations with other AI stacks

AI SDK for Db2 for i

<https://github.com/IBM/AI-SDK-Db2-IBMi>

AI SDK: Watsonx

Set API key and project ID
(configured in watsonx)

Authenticate with watsonx

```
1 call dbsdk_v1.wx_logoutjob();
2 call dbsdk_v1.wx_setapikeyforjob('x');
3 call dbsdk_v1.wx_setprojectidforjob('y');
4
5 -- Should return Y
6 values dbsdk_v1.wx_ShouldGetNewToken();
7
8 -- Should return Y
9 values dbsdk_v1.wx_authenticate();
10
11 -- Should retur
12 values dbsdk_v1.wx_ShouldGetNewToken();
```

WatsonX-SDK-Db2-IBMi

call watsonx.getmodels(); Untitled-1

```
1 call watsonx.getmodels();
```

PROBLEMS IBM I TERMINAL OUTPUT DEBUG CONSOLE RUN AND DEBUG PORTS COMMENTS

RESULTS

MODEL_ID	LABEL	PROVIDER	SHORT_DESCRIPTION
codellama/codellama-34b-instruct-hf	codellama-34b-instruct-hf	Code Llama	Code Llama is an AI model built on t...
cross-encoder/ms-marco-minilm-l-1...	ms-marco-minilm-l-12-v2	cross-encoder	Used for Information Retrieval: Enc...
google/flan-t5-xl	flan-t5-xl-3b	Google	A pretrained T5 - an encoder-decod...
google/flan-t5-xxl	flan-t5-xxl-11b	Google	flan-t5-xxl is an 11 billion parameter ...
google/flan-ul2	flan-ul2-20b	Google	flan-ul2 is an encoder decoder mod...
ibm/granite-13b-chat-v2	granite-13b-chat-v2	IBM	The Granite model series is a family...
ibm/granite-13b-instruct-v2	granite-13b-instruct-v2	IBM	The Granite model series is a family...
ibm/granite-20b-code-instruct	granite-20b-code-instruct	IBM	The Granite model series is a family...
ibm/granite-20b-multilingual	granite-20b-multilingual	IBM	The Granite model series is a family...
ibm/granite-3-2b-instruct	granite-3-2b-instruct	IBM	The Granite model series is a family...
ibm/granite-3-8b-instruct	granite-3-8b-instruct	IBM	The Granite model series is a family...
ibm/granite-34b-code-instruct	granite-34b-code-instruct	IBM	The Granite model series is a family...
ibm/granite-3b-code-instruct	granite-3b-code-instruct	IBM	The Granite model series is a family...

Loaded 42. End of data.

606512/QUSER/QZDASOINIT

main* 0 0 oss74dev Deploy New job (1) Ln 1, Col 26 Spaces: 2 LF {} MS SQL

WatsonX-SDK-Db2-IBMi

a.sql M values watsonx.generate('Hello world'); Untitled-1

```
1 values watsonx.generate('Hello world');
2 values watsonx.generate('Hello world', model_id => 'ibm/granite-3-8b-instruct');
```

PROBLEMS IBM I TERMINAL OUTPUT DEBUG CONSOLE RUN AND DEBUG PORTS COMMENTS

RESULTS

00001

! 🌐👋 I'm a new AI, and I'm here to help you with any questions or tasks you may have. I'm still learning, so please bear with me if I make any

Loaded 1. End of data. 606512/QUSER/QZDASOINIT

main* 0 0 oss74dev Deploy New job (1) Ln 1, Col 39 (38 selected) Spaces: 2 LF {} MS SQL

WatsonX-SDK-Db2-IBMi

a.sql M WATSONX.SQLRPGLE x

LIAMA > QRPGLSRC > WATSONX.SQLRPGLE > ...

```
8
9  dcl-pr printf int(10) extproc('printf');
10 |  format pointer value options(*string);
11  end-pr;
12
13  inputPrompt = 'Hello world';
14
15  exec sql call watsonx.setapikeyforjob('');
16  exec sql call watsonx.setprojectidforjob('');
17  exec sql select watsonx.authenticate() into :authed from sysibm.sysdummy1;
18
19  exec sql set :output = watsonx.generate(:inputPrompt);
20
21  printf(output);
22
```

PROBLEMS 1 IBM I TERMINAL OUTPUT DEBUG CONSOLE RUN AND DEBUG PORTS COMMENTS

Library list: SAMPLE QTEMP QSYSINC LIAMA ILEASTIC
Working directory: /home/LIAMA
Commands:
?CALL LIAMA/WATSONX

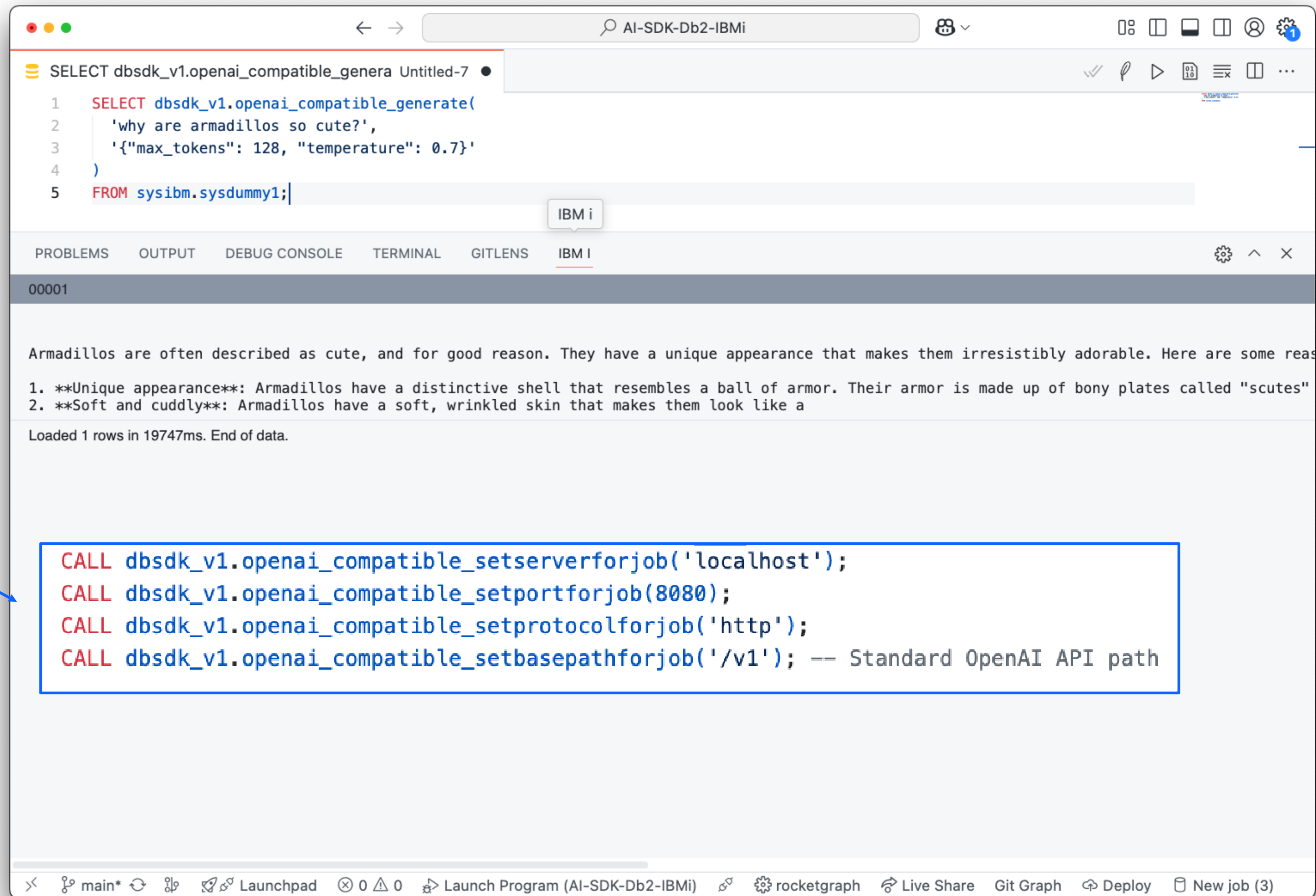
!

I'm a new AI, and I'm here to help you with any questions or tasks you may have. I'm still learning, so please bear with me if I make any *EVENTF not found in command string. Not fetching errors for LIAMA/WATSONX.

main* 0 0 1 oss74dev Deploy New job (1) Ln 16, Col 43 Spaces: 2 LF RPGLE

AI SDK: OpenAI- Compatibility

- Plug in any OpenAI-compatible Endpoint
- Llama.cpp, Ollama, or any custom LLM deployment



The screenshot shows a database IDE window titled "AI-SDK-Db2-IBMi". The editor contains a SQL query:

```
SELECT dbsdk_v1.openai_compatible_generate(
1  SELECT dbsdk_v1.openai_compatible_generate(
2  'why are armadillos so cute?',
3  '{"max_tokens": 128, "temperature": 0.7}'
4  )
5  FROM sysibm.sysdummy1;
```

The output pane shows the result of the query:

```
00001

Armadillos are often described as cute, and for good reason. They have a unique appearance that makes them irresistibly adorable. Here are some reasons why:
1. Unique appearance: Armadillos have a distinctive shell that resembles a ball of armor. Their armor is made up of bony plates called "scutes"
2. Soft and cuddly: Armadillos have a soft, wrinkled skin that makes them look like a

Loaded 1 rows in 19747ms. End of data.
```

A blue box highlights the following SQL commands:

```
CALL dbsdk_v1.openai_compatible_setserverforjob('localhost');
CALL dbsdk_v1.openai_compatible_setportforjob(8080);
CALL dbsdk_v1.openai_compatible_setprotocolforjob('http');
CALL dbsdk_v1.openai_compatible_setbasepathforjob('/v1'); -- Standard OpenAI API path
```

Simple Demo

```
1  CALL dbsdk_v1.openai_compatible_setserverforjob('127.0.0.1');  
2  CALL dbsdk_v1.openai_compatible_setportforjob(8080);  
3  CALL dbsdk_v1.openai_compatible_setprotocolforjob('http');  
4  
5  SELECT dbsdk_v1.openai_compatible_generate('What is Db2 for i?')  
6  FROM sysibm.sysdummy1;
```

Result set from response

▼ RESULTS 📄 ↻ ☰ ⚙️

00001

Db2 is a powerful relational database management system (RDBMS) developed by IBM. It is widely used in enterprise environments for storing, managing, and managing large amounts of data.

****Key aspects of Db2:****

1. ****Relational Database:**** Db2 stores data in a structured, relational format, meaning data is organized into tables with predefined schemas, allowing for efficient querying and data integrity.
2. ****High Performance:**** It is known for its robust performance, handling complex queries and high transaction volumes efficiently, making it suitable for demanding business applications.
3. ****Data Integrity:**** Db2 provides strong mechanisms to ensure data accuracy and consistency through constraints, triggers, and ACID properties (Atomicity, Consistency, Isolation, Durability).
4. ****Scalability:**** It can scale to handle massive datasets and high user loads, supporting large-scale enterprise needs.
5. ****SQL Compliance:**** It adheres to the standard SQL language, ensuring compatibility with other database systems and simplifying development efforts.
6. ****Enterprise Focus:**** It is designed to meet the rigorous demands of large organizations in terms of reliability, security, and compliance.
7. ****Advanced Features:**** It offers advanced features such as sophisticated indexing, stored procedures, and advanced analytical functions.

llama-ibmi

```
CALL dbsdk_v1.openai_compatible_setserver Untitled-1
1 CALL dbsdk_v1.openai_compatible_setserverforjob('127.0.0.1');
2 CALL dbsdk_v1.openai_compatible_setportforjob(8080);
3 CALL dbsdk_v1.openai_compatible_setprotocolforjob('http');
4
5 SELECT dbsdk_v1.openai_compatible_generate('What is Db2 for i?')
6 FROM sysibm.sysdummy1;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL IBM I AZURE

RESULTS

00001

Db2 is a powerful relational database management system (RDBMS) developed by IBM. It is widely used in enterprise environments for storing, managing, and managing large amounts of data.

****Key aspects of Db2:****

- **Relational Database:**** Db2 stores data in a structured, relational format, meaning data is organized into tables with predefined schemas, allowing for efficient querying and data integrity.
- **High Performance:**** It is known for its robust performance, handling complex queries and high transaction volumes efficiently, making it suitable for demanding business applications.
- **Data Integrity:**** Db2 provides strong mechanisms to ensure data accuracy and consistency through constraints, triggers, and ACID properties (Atomicity, Consistency, Isolation, Durability).
- **Scalability:**** It can scale to handle massive datasets and high user loads, supporting large-scale enterprise needs.
- **SQL Compliance:**** It adheres to the standard SQL language, ensuring compatibility with other database systems and simplifying development efforts.
- **Enterprise Focus:**** It is designed to meet the rigorous demands of large organizations in terms of reliability, security, and compliance.
- **Advanced Features:**** It offers advanced features such as sophisticated indexing, stored procedures, and advanced analytical functions.

Loaded 1 rows in 80296ms. End of data. 074245/QUSER/QZDASOINIT

@ 2026 IBM

0 0 common1 Live Share Deploy New job (1) Select Postgres Server Spaces: 4 UTF-8 LF {} MS SQL Go Live Reload Continue (NE) Prettier

Use Cases

1. Summarize customer feedback
2. Analyze data trends
3. Generate and Store custom content



```
1  -- Create a temporary table with customer feedback
2  drop table coolstuff.feedback;
3
4  CREATE or REPLACE TABLE coolstuff.feedback (
5      id INT PRIMARY KEY,
6      comment VARCHAR(1000)
7  );
8
9  select * from coolstuff.feedback;
10
11 INSERT INTO coolstuff.feedback VALUES
12     (1, 'I love your product, it works perfectly!'),
13     (2, 'The software is good but the documentation could be better'),
14     (3, 'This is terrible, nothing works as expected');
15
16 -- Classify each feedback
17 SELECT
18     f.id,
19     f.comment,
20     dbsdk_v1.openai_compatible_generate(
21         'Classify the following customer feedback into one of these categories: Positive, Neutral, Negative. Return just a JSON string with a single "category" field
22
23         Feedback: ' || f.comment,
24         '{
25             "temperature": 0.1,
26             "max_tokens": 30
27         }'
28     ) AS classification
29 FROM coolstuff.feedback f;
```

▼ RESULTS

ID	COMMENT	CLASSIFICATION
1	I love your product, it works perfectly!	<pre>```json { "category": "Positive" } ```</pre>
2	The software is good but the documentation could be better	<pre>. ```json { "category": "Neutral" } ```</pre>
3	This is terrible, nothing works as expected	<pre>. ```json { "category": "Negative" } ```</pre>

```
WITH Untitled-7
1 WITH
2   customer_feedback AS (
3     SELECT 1 AS id, 'The product was delivered late but customer service was very helpful' AS feedback FROM sysibm.sysdummy1
4     UNION ALL SELECT 2, 'Excellent quality and fast shipping, will buy again!' FROM sysibm.sysdummy1
5     UNION ALL SELECT 3, 'Difficult to set up but works well once configured' FROM sysibm.sysdummy1
6     UNION ALL SELECT 4, 'Very disappointed with the quality, returning item' FROM sysibm.sysdummy1
7     UNION ALL SELECT 5, 'The product is great, but the packaging is not as good as I expected' FROM sysibm.sysdummy1
8   ),
9   formatted_data AS (
10    SELECT LISTAGG('ID: ' || id || ', Feedback: "' || feedback || "', '
11  ') AS all_feedback
12    FROM customer_feedback
13  )
14  SELECT dbsdk_v1.openai_compatible_generate(
15    'Analyze the following customer feedback. For each entry, determine the sentiment (positive/negative/mixed),
16    identify key themes, and suggest appropriate follow-up actions:
17    ' || all_feedback,
18    '{
19      "temperature": 0.5,
20      "max_tokens": 750
21    }'
22  ) FROM formatted_data;
```

00001

Analysis

ID: 1

- * **Sentiment:** Mixed
- * **Key Themes:** Delivery timeliness, Customer service responsiveness.
- * **Follow-up Actions:** Investigate the delay in delivery logistics. Acknowledge and commend the customer service team for their helpfulness.

ID: 2

- * **Sentiment:** Positive
- * **Key Themes:** Product quality, Shipping speed, Customer loyalty.
- * **Follow-up Actions:** Continue current processes for quality control and shipping efficiency. Encourage repeat purchases through targeted marketing.

ID: 3

- * **Sentiment:** Mixed
- * **Key Themes:** Ease of setup, Functionality after setup.
- * **Follow-up Actions:** Improve the initial setup instructions/tutorials. Gather feedback on the setup process to identify friction points.

ID: 4

- * **Sentiment:** Negative
- * **Key Themes:** Product quality, Return process.
- * **Follow-up Actions:** Conduct an immediate quality audit on the specific product batch. Review the return policy process for clarity and customer experience.

ID: 5

- * **Sentiment:** Mixed
- * **Key Themes:** Product satisfaction, Packaging quality.
- * **Actions:** Review packaging materials and design to enhance protection during transit.

Final Review

The analysis is thorough and the sentiment determination and suggested actions are logical and actionable. The structure is clear and directly addresses the prompt for each ID. No significant err

Use case: Analyze data trends
(data summarization)

```
WITH Untitled-7
1 WITH
2   sales_data AS (
3     SELECT 'North' as region, 150000 as q1_sales, 180000 as q2_sales, 120000 as q3_sales, 200000 as q4_sales FROM sysibm.sysdummy1
4     UNION ALL SELECT 'South', 120000, 110000, 140000, 130000 FROM sysibm.sysdummy1
5     UNION ALL SELECT 'East', 160000, 170000, 180000, 190000 FROM sysibm.sysdummy1
6     UNION ALL SELECT 'West', 130000, 140000, 150000, 160000 FROM sysibm.sysdummy1
7   ),
8   formatted_data AS (
9     SELECT LISTAGG('Region: ' || region || ', Q1: ' || q1_sales || ', Q2: ' || q2_sales ||
10    | | ', Q3: ' || q3_sales || ', Q4: ' || q4_sales, '
11   ') AS sales_text
12   FROM sales_data
13  )
14  -- Generate a summary
15  SELECT dbsdk_v1.openai_compatible_generate(
16    'Analyze the following sales data and provide insights:
17    ' || sales_text,
18    '{
19      "temperature": 0.7,
20      "max_tokens": 500
21    }'
22  ) FROM formatted_data;
```

00001

Sales Data Analysis

1. Overall Performance Overview

To get a holistic view, let's calculate the total sales for each region across all four quarters.

- * **North Total:** \$150,000 + 180,000 + 120,000 + 200,000 = \$650,000
- * **South Total:** \$120,000 + 110,000 + 140,000 + 130,000 = \$500,000
- * **East Total:** \$160,000 + 170,000 + 180,000 + 190,000 = \$700,000
- * **West Total:** \$130,000 + 140,000 + 150,000 + 160,000 = \$580,000

Insight: The **East** region generated the highest total sales (\$700,000), followed closely by the **North** (\$650,000). The **South** region had the lowest total sales (\$500,000).

2. Quarterly Trend Analysis (Seasonality)

Let's examine the trend within each region to identify seasonal patterns.

- * **North:** \$150k → 180k → 120k → 200k (Fluctuating, with a dip in Q3)
- * **South:** \$120k → 110k → 140k → 130k (Shows a dip in Q2)
- * **East:** \$160k → 170k → 180k → 190k (Consistent upward trend)
- * **West:** \$130k → 140k → 150k → 160k (Consistent upward trend)

Use Case: Generate and Store custom content

```
AI-SDK-Db2-IBMi
-- Create a table to store the generated content
1  -- Create a table to store the generated content
2  CREATE TABLE generated_content (
3      id INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
4      prompt VARCHAR(1000),
5      content CLOB(2G),
6      generation_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
7  );
8
9  -- Generate and store content
10 INSERT INTO generated_content (prompt, content)
11 SELECT 'Write a short poem about databases',
12        dbsdk_v1.openai_compatible_generate(
13            'Write a short poem about databases',
14            '{
15                "temperature": 0.7,
16                "max_tokens": 200,
17                "top_p": 0.95
18            }'
19        )
20 FROM sysibm.sysdummy1;
21
22 -- Retrieve the stored content
23 SELECT * FROM generated_content;
```

▼ RESULTS

ID	PROMPT	CONTENT	GENERATION_TIME
1	Write a short poem about databases	<p>.</p> <p>In structured rows, Data neatly stored, Indexed for quick retrieval, A digital treasure.</p> <p>Relational links bind, Tables in perfect alignment, SQL commands command, Knowledge in alignment.</p> <p>A digital vault, secure and deep, Where information safely sleeps.</p>	2026-04-28 17:01:04.250550

```

exfmt test;
dow not (*in03 or *in12);
  if *in05;
    clear usertext;
    clear response;
  else;
    exec sql select
      LEFT( dbsdk_v1.wallaroo_generate(:usertext :text_null), 65)
      into :response :response_null
      from sysibm.sysdummy1;
    *in99 = *off;
    if sqlstate <> '00000';
      exec sql get diagnostics condition 1
        :errmsg = MESSAGE_TEXT;
      response = errmsg;
      *in99 = *on;
    elseif response_null <> 0;
      response = 'Null response';
    endif;
  endif;
exfmt test;
enddo;

```



```
Enter your input  
What is the capital of Vermont?
```

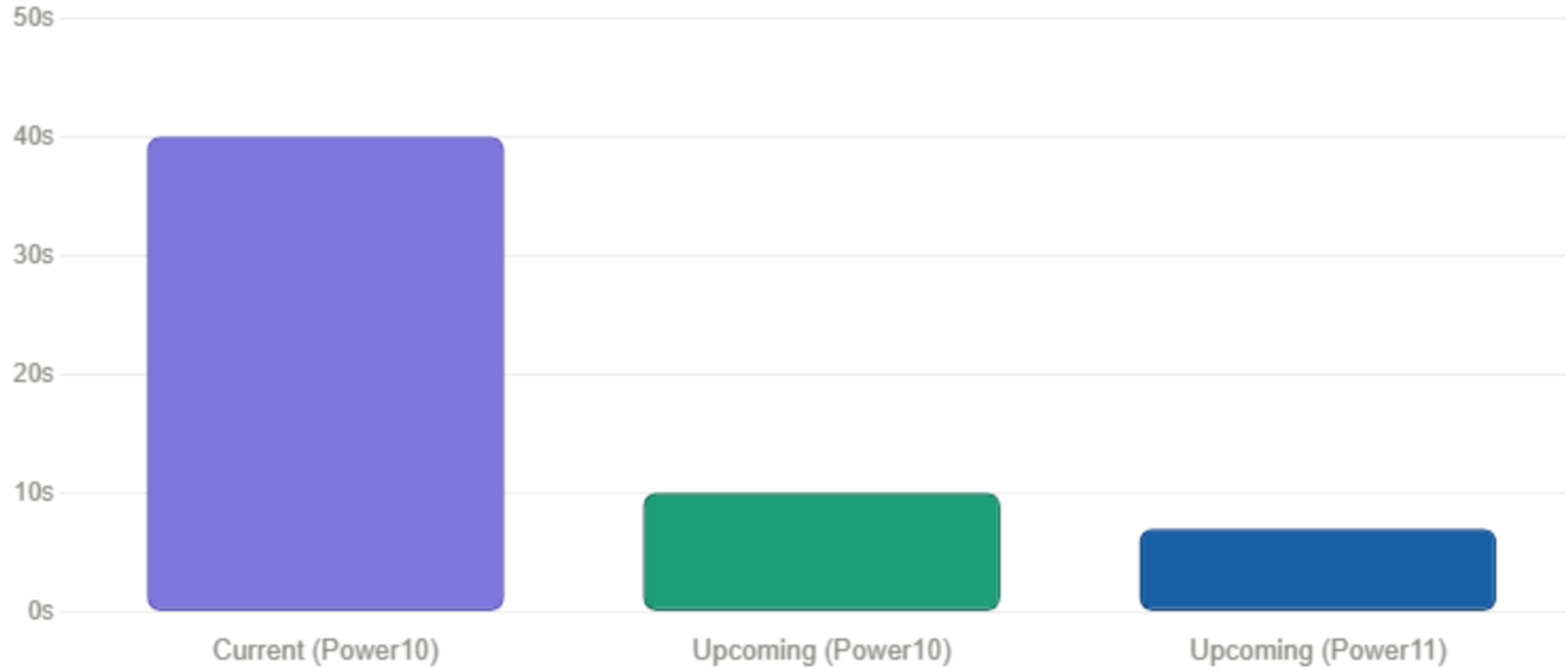
```
Response:
```

```
0:16 The capital of Vermont is Montpelier.
```

Virtual Assistant Use Case

(benchmarked with Wallaroo AI Starter Kit)

■ Current (Power10) ■ Upcoming (Power10) ■ Upcoming (Power11)



Other features...

A Db2 for i SDK for connecting to various services

 Ctrl K

- Home
- IBM watsonx >
- Ollama >
- OpenAI-compliant endpoints >
- Kafka >
 - Requisites
 - Usage
 - Examples**
- PASE >
- Slack >
- SMS (Twilio) >
- Other Useful links (external) >

Examples

Publish a message

Then at some future time (even a different session):

```
values(watsonx.kafka_publish( TOPIC => 'fun_facts',  
                             KEY     => 'baritone43',  
                             MSGDATA => 'The baritone is a very interesting musical instru
```

Publish a message with a per-job topic

Then at some future time (even a different session):

```
call watsonx.kafka_settopicforjob('fun_facts');  
values(watsonx.kafka_publish( KEY     => 'baritone43',  
                             MSGDATA => 'The baritone is a very interesting musical instru
```

Other features...

```
23 select * from table(jesseg.callpase('echo "this\n\nthat'))
```

OUTPUT

this

that

Overview: AI SDK for Db2 for i

Easy Db2 calls for:

- watsonx.ai
- Ollama
- OpenAI and OpenAI-compliant solutions
 - LLaMa.cpp
 - Rocket CE
 - vLLM
 - LM Studio
- Kafka
- Slack
- Twilio (SMS)
- PASE commands

Project Link: <https://ibm.github.io/AI-SDK-Db2-IBMi/>



LM Studio

Links!

- Llama.cpp Blog: <https://ibm.biz/llamacpp-ibmi>
- AI SDK for Db2 for i: <https://ibm.github.io/AI-SDK-Db2-IBMi/>
- Models for i: <https://huggingface.co/models-for-i>

IBM