

Developing for IBM i

with AI tools

Consuming AI

AI for RPGLE

AI for Db2 for IBM i

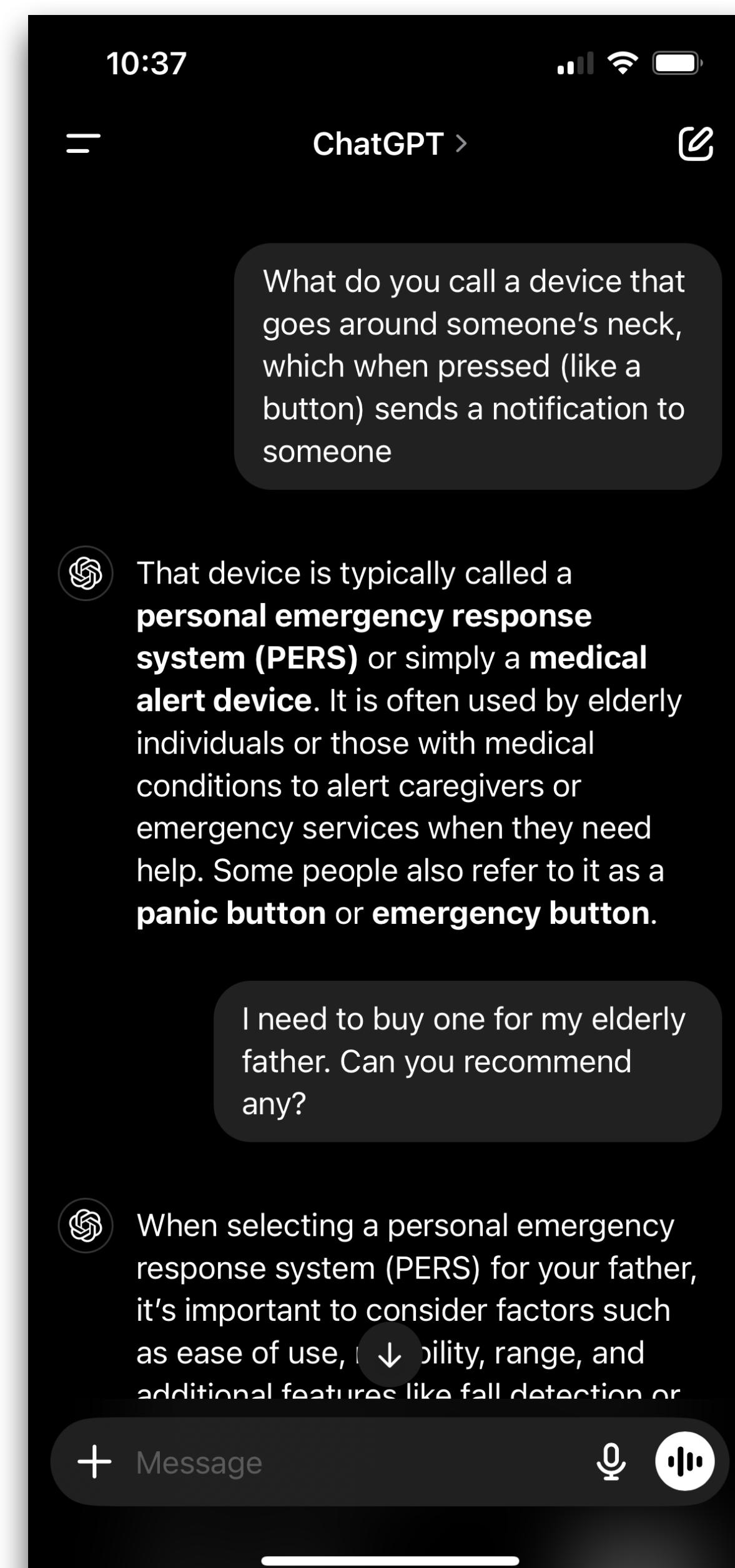
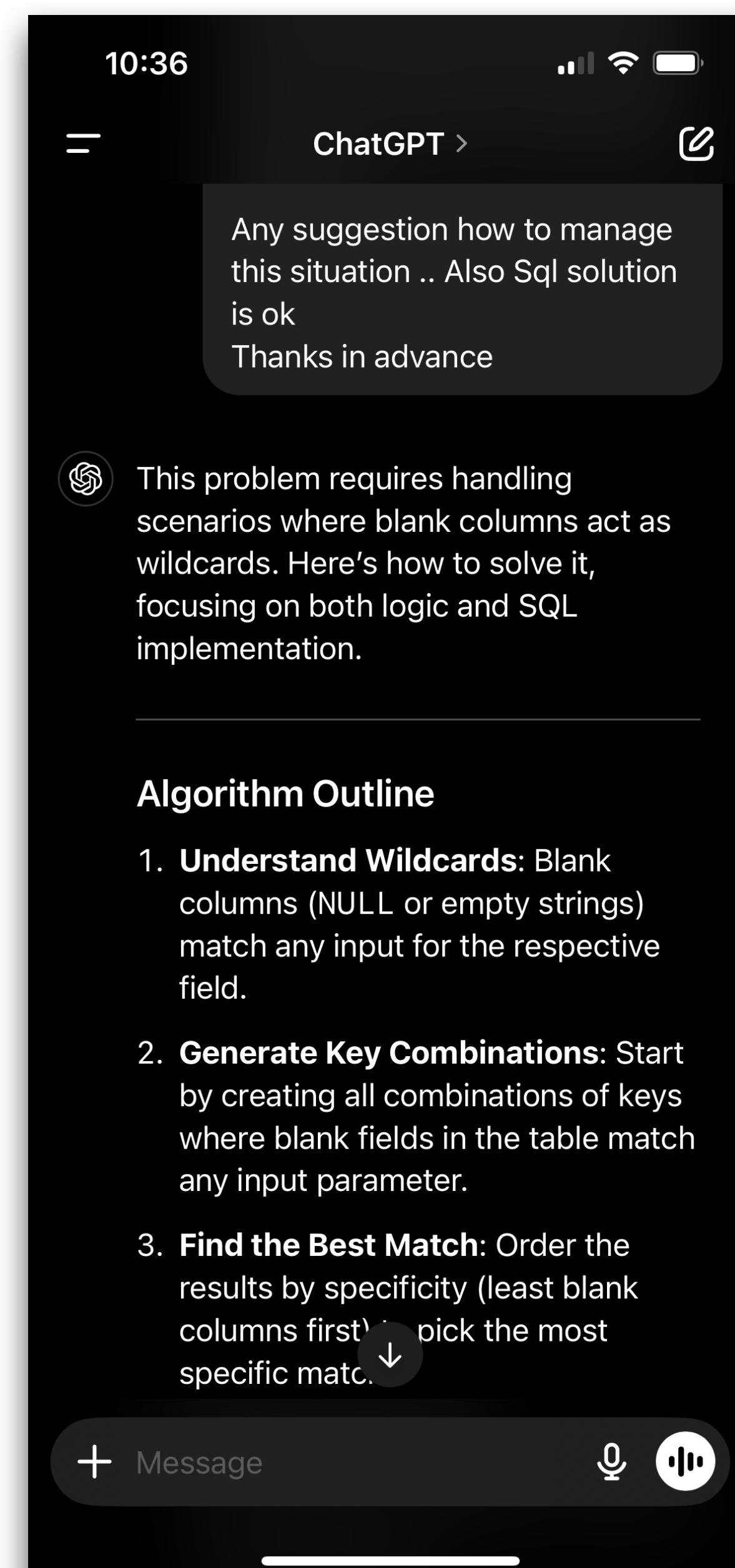
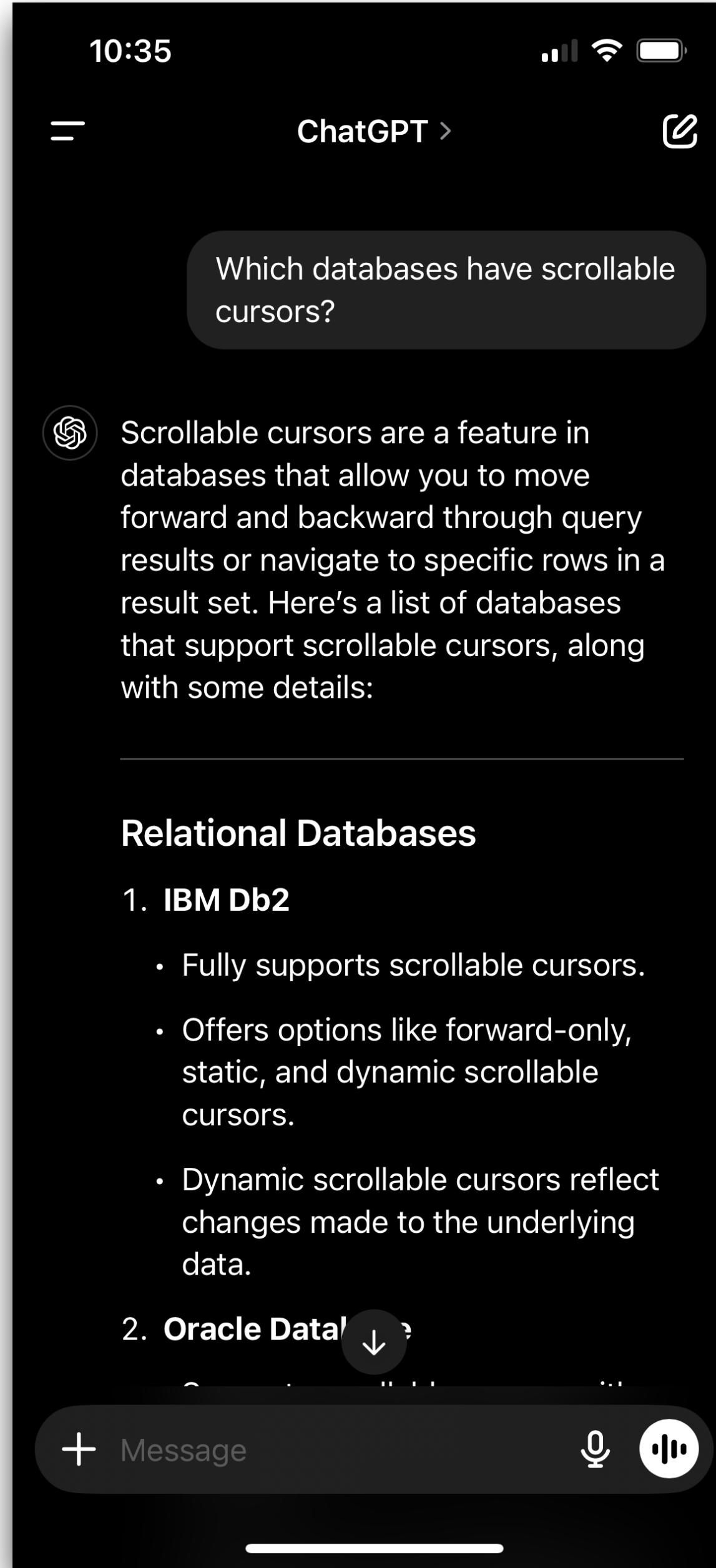
**Prompt
Engineering and
Enrichment**

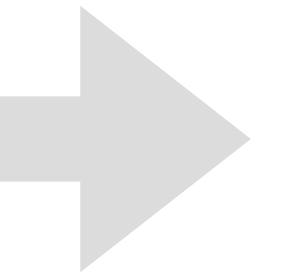
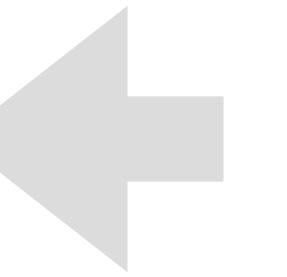
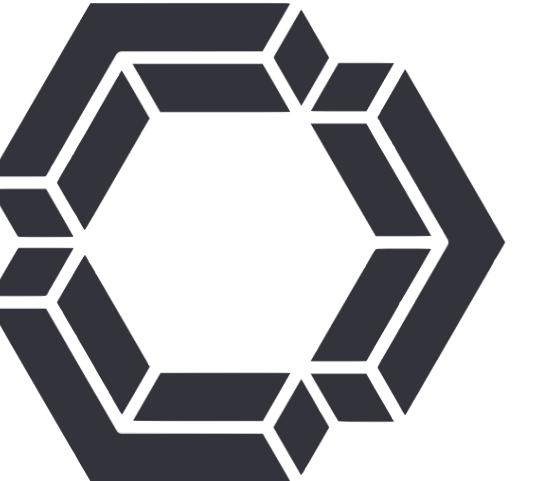
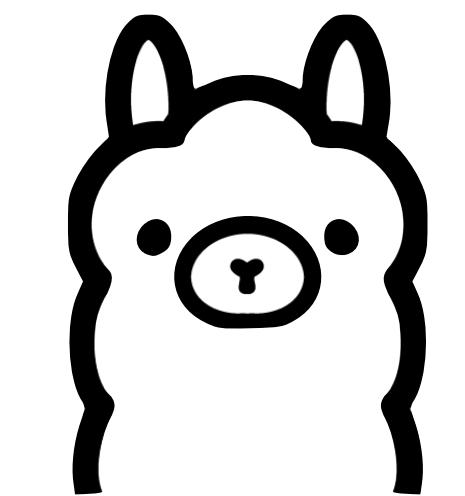
Consumer

ChatGPT
Watson Code Assistant
Copilot
Continue
Ollama
Hugging Face
Cline

Development

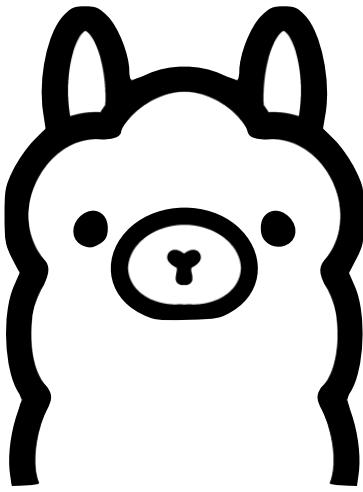
Ollama
Hugging Face
Copilot
Continue
Watson Code Assistant













ollama.com

granite-code

A family of open foundation models by IBM for Code Intelligence

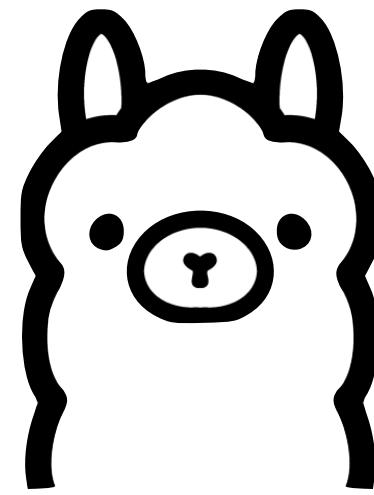
[3b](#) [8b](#) [20b](#) [34b](#)

 175.9K Pulls  Updated 4 months ago

3b	▼	162 Tags	ollama run granite-code	
Updated 4 months ago			becc94fe1876 · 2.0GB	
model	arch llama · parameters 3.48B · quantization Q4_0		2.0GB	
params	{ "stop": ["System:", "Question:", "Answer:"] }		43B	
template	{{ if .Suffix }}<fim_prefix> {{ .Prompt }}<fim_suffix> {{ ... }}		679B	
license	Apache License Version 2.0, January 2004		11kB	

```
[barry@Liams-MBP ~ % ollama serve
2025/01/23 11:18:51 routes.go:1195: INFO server config env="map[HTTPS_PROXY: HTTP_PROXY: NO_PROXY: OLLAMA_DEBUG:false OLLAMA_FLASH_ATTENTION:false OLLAMA_GPU_OVERHEAD:0 OLLAMA_HOST:http://127.0.0.1:11434 OLLAMA_KEEP_ALIVE:5m0s OLLAMA_KV_CACHE_TYPE: OLLAMA_LLM_LIBRARY: OLLAMA_LOAD_TIMEOUT:5m0s OLLAMA_MAX_LOADED_MODELS:0 OLLAMA_MAX_QUEUE:512 OLLAMA_MODELS:/Users/barry/.ollama/models OLLAMA_MULTIUSER_CACHE:false OLLAMA_NOHISTORY:false OLLAMA_NOPRUNE:false OLLAMA_NUM_PARALLEL:0 OLLAMA_ORIGINS:[http://localhost https://localhost http://localhost:* https://localhost:* http://127.0.0.1 https://127.0.0.1 http://127.0.0.1:* https://127.0.0.1:* http://0.0.0.0 https://0.0.0.0 http://0.0.0.0:* https://0.0.0.0:* app:///* file:///* tauri:///* vscode-webview:///*] OLLAMA_SCHED_SPREAD:false OLLAMA_TMPDIR: http_proxy: https_proxy: no_proxy:]"
time=2025-01-23T11:18:51.384-05:00 level=INFO source=images.go:753 msg="total blobs: 5"
time=2025-01-23T11:18:51.384-05:00 level=INFO source=images.go:760 msg="total unused blobs removed: 0"
time=2025-01-23T11:18:51.385-05:00 level=INFO source=routes.go:1246 msg="Listening on 127.0.0.1:11434 (version 0.5.1)"
time=2025-01-23T11:18:51.392-05:00 level=INFO source=common.go:135 msg="extracting embedded files" dir=/var/folders/_3/4hxcf12x4mq5gl3y96cxn3fh0000gn/T/ollama1634341714/runners
time=2025-01-23T11:18:51.461-05:00 level=INFO source=common.go:49 msg="Dynamic LLM libraries" runners=[metal]
time=2025-01-23T11:18:51.490-05:00 level=INFO source=types.go:123 msg="inference compute" id=0 library=metal variant="" compute="" driver=0.0 name="" total="21.3 GiB" available="21.3 GiB"
[GIN] 2025/01/23 - 11:18:57 | 200 | 1.161458ms | 127.0.0.1 | HEAD | /
[GIN] 2025/01/23 - 11:18:57 | 200 | 1.048208ms | 127.0.0.1 | GET | /api/tags"
```

```
Last login: Thu Jan 23 11:18:36 on ttys000
[barry@Liams-MBP ~ % ollama list
NAME                  ID          SIZE      MODIFIED
granite-code:8b      36c3c3b9683b    4.6 GB    6 weeks ago
barry@Liams-MBP ~ %
```



```
barry@Liams-MBP ~ % ollama run llama3.1:8b
>>> Hello
How can I help you today?

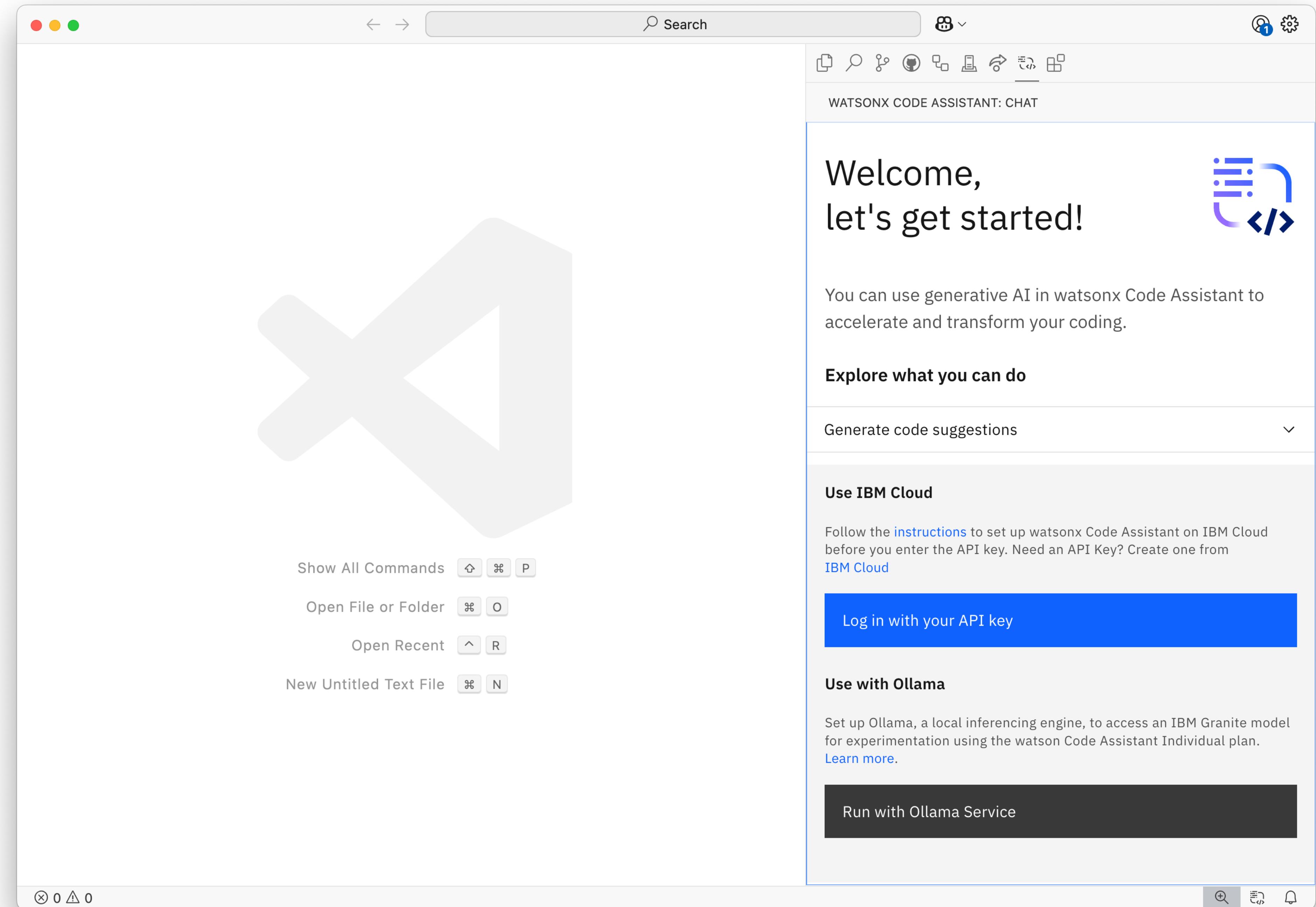
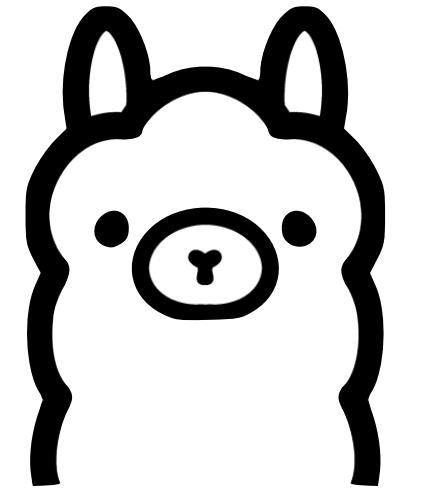
>>> What languages do you support?
I'm a multilingual AI and can understand and respond in many languages.
Here are some of the languages I support:

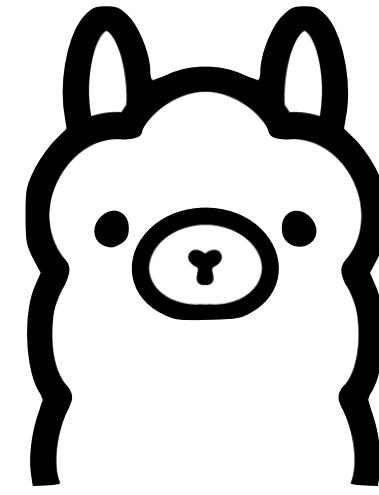
**Main languages:**

1. English (default)
2. Spanish
3. French
4. German
5. Italian
6. Portuguese
7. Dutch
8. Russian
9. Chinese (Simplified and Traditional)
10. Japanese

**Other languages:**

I can also understand and respond in many other languages, including but
```





\$ ollama serve

The screenshot shows a code editor window with a terminal tab open. The terminal output displays the command '\$ ollama serve' followed by a list of token definitions and a success message from GIN.

```
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE ...
```

```
llm_load_print_meta: model_size = 4.27 GiB (4.56 BPW)
llm_load_print_meta: general.name = Granite 8b Code Instruct 128k
llm_load_print_meta: BOS token = 0 '<|endoftext|>'
llm_load_print_meta: EOS token = 0 '<|endoftext|>'
llm_load_print_meta: UNK token = 0 '<|endoftext|>'
llm_load_print_meta: PAD token = 0 '<|endoftext|>'
llm_load_print_meta: LF token = 145 'Ä'
llm_load_print_meta: EOT token = 0 '<|endoftext|>'
llm_load_print_meta: EOG token = 0 '<|endoftext|>'
llm_load_print_meta: max token length = 512
llama_model_load: vocab only - skipping tensors
[GIN] 2025/01/23 - 11:28:58 | 200 | 11.28697725s | 127.0.0.1 | POST "/api/chat"
```

The right side of the interface features a WatsonX Code Assistant chat session titled 'Chat session 3'. The AI is identified as 'watsonx' and responds with its capabilities, mentioning support for multiple programming languages like Python, Java, JavaScript, C++, and more. A user message 'What languages do you support?' is shown, and the AI provides a detailed response.

WatsonX CODE ASSISTANT: CHAT

Chat session 3

watsonx

Hi, how can I help you today?

You

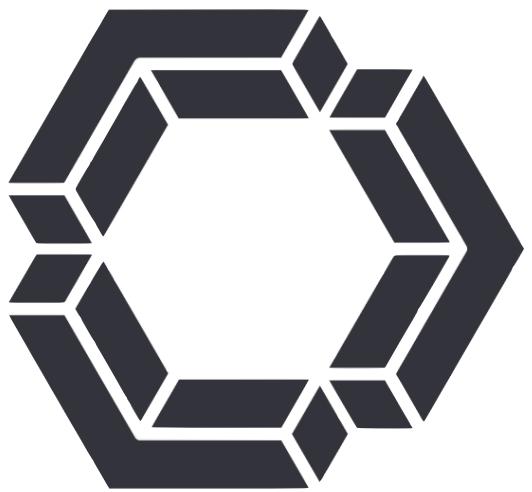
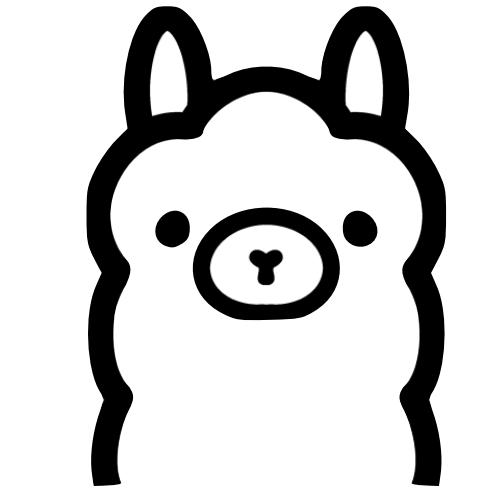
WatsonX Code Assistant

What languages do you support? :

watsonx

I am a code assistant and I support multiple programming languages. Some of the popular programming languages that I support include Python, Java, JavaScript, C++, and many more. However, please note that my proficiency and accuracy in coding and programming depend on the quality and completeness of the information provided by the user. Therefore, it is always a good idea to double-check my responses and provide as much relevant information as possible when asking questions.

Enter / for commands



Search CONTINUE

Ask anything, '@' to add context
Granite Chat @ [@codebase](#) Enter

Last Session

Quickstart Best experience Local with Ollama

Install Ollama

<https://ollama.ai/download>

Download Chat model

ollama pull llama3.1:8b ✓

Download Autocomplete model

ollama pull qwen2.5-coder:1.5b ✓

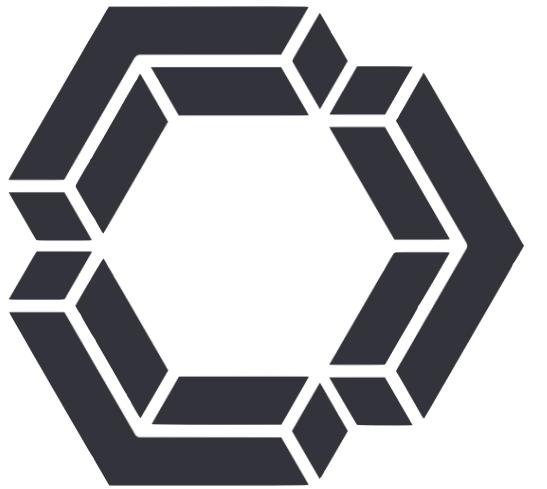
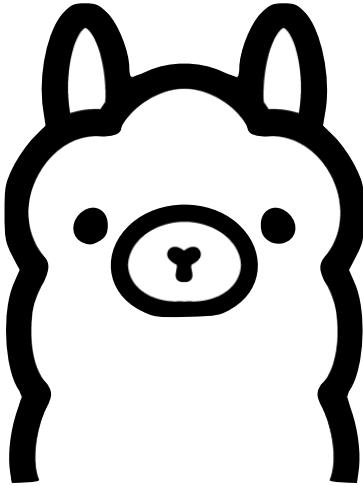
Connect

This will update your [config file](#)

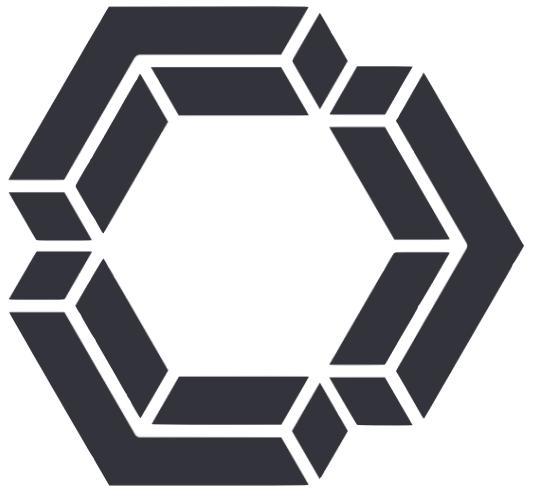
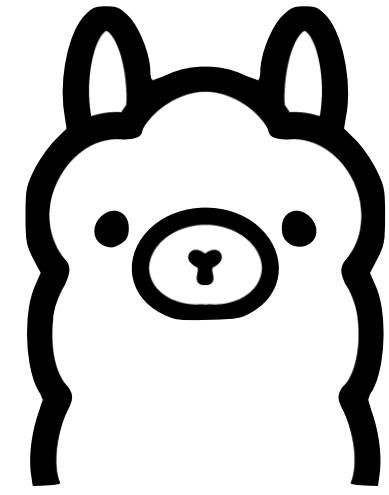
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE RUN AND DEBUG PORTS + ⌂ ollama ⌂ ollama

pulling manifest
pulling 667b0c1932bc... 100% | 4.9 GB
pulling 948af2743fc7... 100% | 1.5 KB
pulling 0ba8f0e314b4... 100% | 12 KB
pulling 56bb8bd477a5... 100% | 96 B
pulling 455f34728c9b... 100% | 487 B
verifying sha256 digest
writing manifest
success
barry@Liams-MBP ~ % ollama pull qwen2.5-coder:1.5b
pulling manifest
pulling 29d8c98fa6b0... 100% | 986 MB
pulling 66b9ea09bd5b... 100% | 68 B

0 0 Continue



A screenshot of the Visual Studio Code interface. A large, semi-transparent 'X' watermark is overlaid on the left side of the editor area. The top navigation bar includes icons for file operations, search, and user profile. The main menu bar has options like File, Edit, View, Insert, Select, Search, Run, Debug, Terminal, and Help. The terminal at the bottom shows a command-line session for pulling a Qwen 2.5-coder image. The status bar at the bottom right shows connection details and a battery icon.



The screenshot shows the Visual Studio Code interface with the 'Extension Development Host' tab selected. The search bar at the top contains the query '[Extension Development Host] sql'. The left sidebar displays the 'Continue - Codestral, Claude, and more' extension details, including its logo, name, developer information, ratings, and download count (370,224). Below this are tabs for 'DETAILS', 'FEATURES', and 'CHANGELOG'. A large preview image of the extension's user interface is shown, featuring the word 'Continue' and several categories like 'Programming Languages', 'Snippets', 'Machine Learning', 'Education', 'AI', and 'Chat'. The 'TERMINAL' tab is active, showing a terminal window with the command 'cd ~/Documents/IBM/sandbox/oss/ai/sql'. A context menu is open over the terminal, with the 'CONTINUE' option highlighted. A modal dialog titled 'Add Chat model' is displayed, prompting for provider (set to 'IBM watsonx'), model ('watsonx - Granite Code 20b'), watsonx URL (empty), project ID (empty), API key (empty), and API version ('2023-05-29'). A note at the bottom of the dialog states 'This will update your config file'.



CHAT + ⏪ ...

Ask Copilot

Sign in with GitHub to use GitHub Copilot, your AI pair programmer.

[Sign in to GitHub.com](#)

[Sign in with a GHE.com account](#)

Or explore the [Copilot walkthrough](#) to learn more!

Search

Show All Commands ⌘ P

Open File or Folder ⌘ O

Open Recent ⌘ R

New Untitled Text File ⌘ N

Open Chat ⌘ I

✖ 0 ⌛ 0



A promotional banner for GitHub Copilot in VS Code. It features a dark background with a purple-to-blue gradient. On the right side, there is a stylized, glowing version of the GitHub Copilot logo (a robot head) with a purple and blue glow. To the left of the logo, the text "GitHub Copilot in VS Code" is written in white, and below it, the word "FREE" is written in a large, bold, white font.



← → ⌂ Search ⌂ v 🔎 🔍

CHAT + ⏪ ⏴ ⏵

 worksofliam

What languages do you support?

 GitHub Copilot

I support a wide range of programming languages, including but not limited to:

- Python
- JavaScript
- TypeScript
- Java
- C#
- C++
- Ruby
- Go
- Swift
- Kotlin
- PHP
- HTML
- CSS
- SQL
- R
- MATLAB
- Bash/Shell scripting
- JSON
- YAML
- Markdown

What language should I use?

Ask Copilot @ ⌂ ⌂ GPT 4o ⏪ ⏴ ⏵

Show All Commands ⌂ ⌂ P
Open File or Folder ⌂ O
Open Recent ⌂ R
New Untitled Text File ⌂ N
Open Chat ⌂ ⌂ I

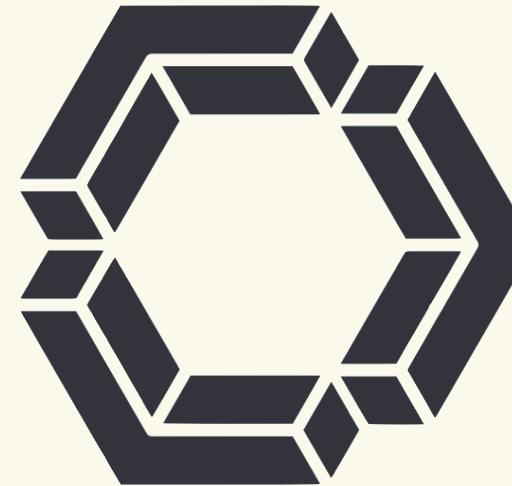
✖

⊗ 0 ⌂ 0 🔎 🔍

A word on context building



Base models provided understand RPGLE
Prompt engineering powered by the Copilot extension
Prompt enrichment for RPGLE not available



Some models know RPGLE well
Prompt engineering powered by the Continue extension
Prompt enrichment for RPGLE not available



Granite knows RPGLE
Prompt engineering for RPGLE not available
Prompt enrichment for RPGLE not available

Coming soon!

What is prompt engineering?

chatgpt.com

Actions Overview and Usage

What are Actions? | Code for IBM i Docs

ChatGPT

An action is used to perform a task on a member, streamfile or other type of object. A comprehensive default set of actions is loaded directly from the extension. You can also easily change or add to actions—see View/Change/Add Actions, below.

Actions are defined commands used to perform tasks on members, streamfiles and other types of objects. For example, to compile a source file. Actions can be executed from two different places:

- As you're writing code. For example, to compile a program or module.
- When you right click on a member, streamfile or an object from the OBJECT BROWSER.

A comprehensive set of Actions is provided, but you can add more or change the ones provided.

Can you summarize an action in one sentence?

An action is a defined command that performs a specific task, such as compiling code, on members, streamfiles, or other objects, and can be executed from the editor or object browser.

Message ChatGPT

ChatGPT can make mistakes. Check important info.

What is prompt engineering?

Additional context provided by extensions

The variable `taxPercentage` is used in the `calculateTotal` and `getTaxPercByCountry` functions. The value is gathered from a database statement that returns a row based on the country.

What is the purpose of the variable `taxPercentage`?

User input

```
function getTaxPercByCountry(country: string): number {
  const [row] = query(`SELECT tax_code FROM countries WHERE country = ${country}`);

  return row.tax_code;
}

function calculateTotal(input, country) {
  const taxPercentage = getTaxPercByCountry(country);

  return input * (1 + taxPercentage / 100);
}

// Example usage
const total = calculateTotal(100, 'UK');
```

What is prompt enrichment?

The references table 'countries' has 6 columns, but the important columns are tax_percent and country. Country is a character(2) column, where tax_percent is a decimal(3, 2) and is likely used to store the tax percentage for each country.

The variable taxPercentage is used in the calculateTotal and getTaxPercByCountry functions. The value is gathered from a database statement that returns a row based on the country.

What is the purpose of the variable taxPercentage?

Specific enrichment specific to this source

Additional context provided by extensions

User input

```
dcl-proc getTaxPercByCountry;
dcl-pi *n packed(3:2);
country char(2) const;
end-pi;

dcl-s taxPerc packed(3:2);

exec sql
...
select tax_percent
into :taxPerc
from counties
where country = :country;

return taxPerc;
end-proc;

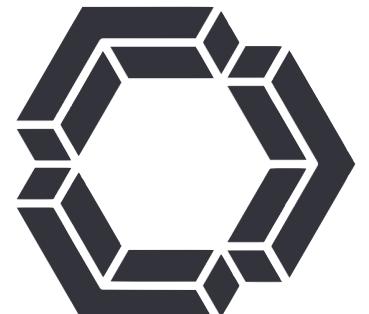
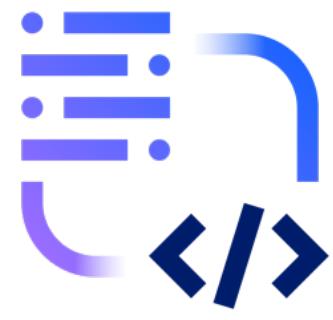
dcl-proc calculateTotal;
dcl-pi *n;
amount packed(7:2) const;
country char(2) const;
end-pi;

dcl-s taxPercentage packed(3:2);

taxPercentage = getTaxPercByCountry(country);

return amount + (amount * taxPercentage / 100);
end-proc;
```

Usage



Additional context specific to IBM i
added by RPGLE extension

User input

Specific enrichment specific to this source

The references table 'countries' has 6 columns, but the important columns are tax_percent and country. Country is a character(2) column, where tax_percent is a decimal(3, 2) and is likely used to store the tax percentage for each country.

The variable taxPercentage is used in the calculateTotal and getTaxPercByCountry functions. The value is gathered from a database statement that returns a row based on the country.

What is the purpose of the variable taxPercentage?

User input

Additional context provided by extensions

Prompting case study

GitHub-Copilot-Cobol-Extension

GitHub Copilot Cobol Extension

Description

This Visual Studio Code extension provides tools and utilities to assist with COBOL programming. It includes features for generating documentation, explaining code, and analyzing COBOL programs.

Features

- **Explain COBOL Program:** Generate detailed explanations for COBOL programs, including program info, data division, and procedure division.
- **Generate Documentation:** Automatically generate markdown documentation for COBOL programs.
- **Analyze COBOL Code:** Chunk COBOL code into divisions, sections, and paragraphs for detailed analysis.

Commands

The extension provides the following commands:

- `explain` : Explain the entire COBOL program.
- `info` : Extract and display information about the COBOL program.
- `data` : Explain the data division of the COBOL program.
- `procedure` : Explain the procedure division of the COBOL program.
- `gendoc` : Generate a markdown documentation file for the COBOL program.

```
stream.progress("Checking file content...");
if (!isCobolProgram(content)) {
    const code = await explainCode(content, stream, request, token);
    stream.markdown(code);
    return;
}
```

```
export async function explainCode(
  code: string,
  stream: vscode.ChatResponseStream,
  request: vscode.ChatRequest,
  token: vscode.CancellationToken
) {
  stream.progress(`Analyzing code...`);

  const prompt = explainCodePrompt.replace(`{{code}}`, code);

  const msgs = [
    vscode.LanguageModelChatMessage.Assistant(systemPrompt),
    vscode.LanguageModelChatMessage.User(prompt),
  ];
  let contentResult = "";
  const result = await request.model.sendRequest(msgs, {}, token);
  for await (const fragment of result.text) {
    stream.markdown(fragment);
    contentResult += fragment;
  }
  return contentResult;
}
```

```
export const systemPrompt =
  "You are a useful Assistant that helps to generate docs for COBOL code, do not add code samples.";

export const explainCodePrompt = `
  Write an explanation for the COBOL program as paragraphs of text
  Do not insert source code inside explanation
  Do not generate or add examples
  Do not add consideration or comments

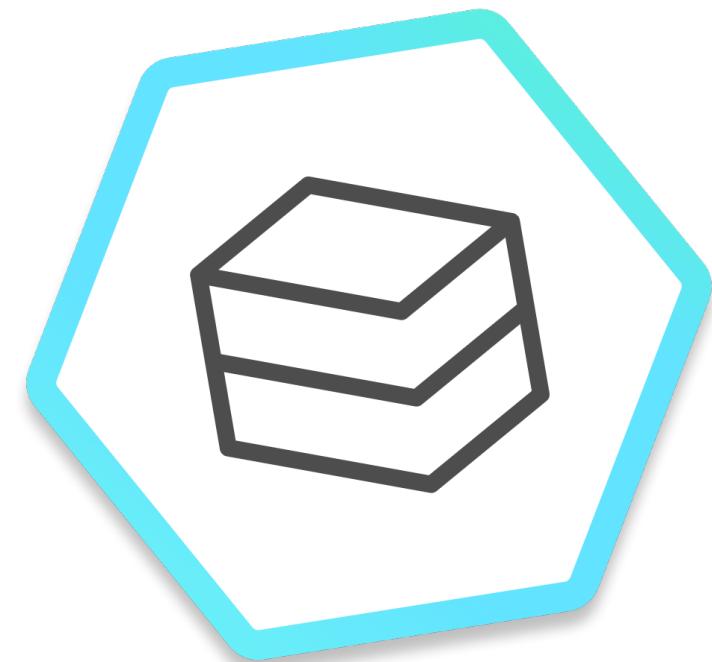
  Below is the code, do not add it to output:

  ### CODE
  \n  {{code}}
  ### END CODE
  `;
```

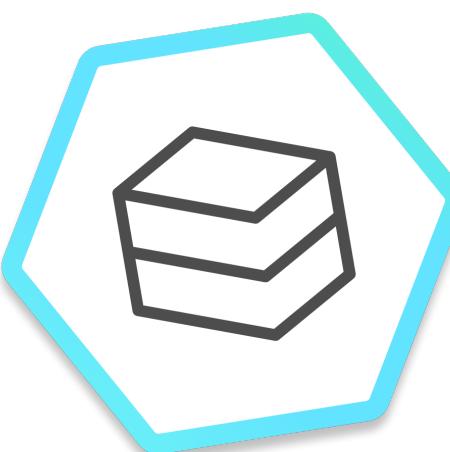
```

60
61 - 2: ...
62
63 ##### Diagram
64 mermaid_flow_diagram
65
66 ##### External Calls
67 - call_name
68 ##### Internal Calls
69 - perform_name
70
71
72
73 Below is the code, do not add it to output:
74
75 ### CODE
76 \n {{code}}
77 ### END CODE
78 `;
79 export const dataSectionPrompt = `list all variables, labels and constants at all nested levels as a bullet list, for each variable , label and constant at every level and nested level add a paragraph with description and data type
80 respect the hierarchy levels of the variables, labels and constants in the list and put level number in brackets after the constant and variable name like VARIABLE_NAME (level)
81 do not add any title like 'Variables, Labels, and Constants...' or similar
82 each bullet point should be like this:
83 - **VARIABLE_NAME (LEVEL_NUMBER)**
84 | - Description: Variable description.
85 | - Data Type: Variable data type.
86 If you dont find any variables, labels or constants just respond with 'none'
87 Below is the code, do not add it to output:
88
89
90 ### CODE
91 \n {{code}}
92 ### END CODE
93 `;
94
95 export const explainCodePrompt = `Write an explanation for the COBOL program as paragraphs of text
96 Do not insert source code inside explanation
97 Do not generate or add examples
98 Do not add consideration or comments
99
100 Below is the code, do not add it to output:
101
102
103 ### CODE
104 \n {{code}}
105 ### END CODE
106 `;

```



**Db2 for IBM i,
AI for development**



/help What can you do?

@db2i Chat with the Db2 for i AI assistant

@

GPT 4o



CONTINUE

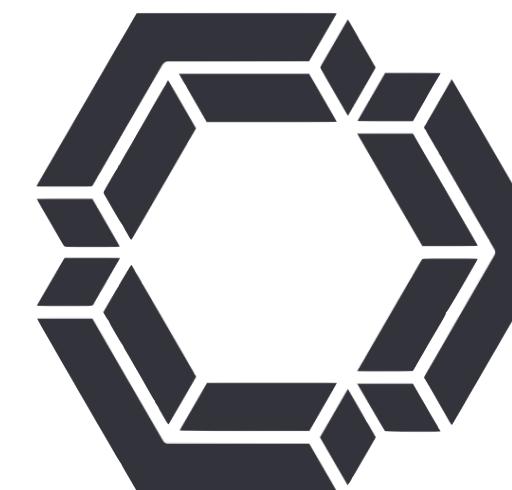
@

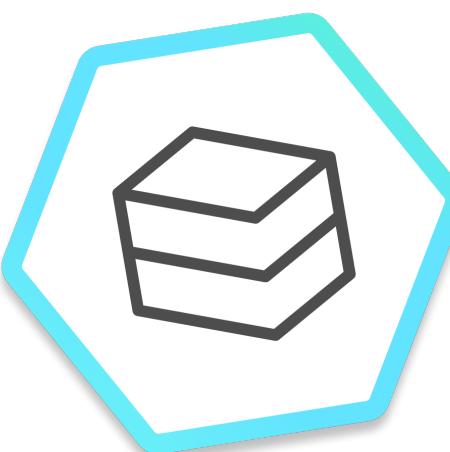
Files

Codebase

Db2i Db2 for i Context Provider

Add more context providers





User input _____

CONTINUE

+

@Db2i tell me about the columns in sample.employee

Autodetect (1) - granite3-dense... @

⌘ ⇣ @codebase

↔ Enter

▽ 2 context items

@ SYSTEM PROMPT system prompt context

@ New job (1)-SAMPLE-employee Schema and table information for employee

Additional context info _____

Response _____

The `sample.employee` table has the following columns:

1. `EMPNO` : A 6-character, not null column with data type CHAR.
2. `FIRSTNAME` : A 12-character, not null column with data type VARCHAR.
3. `MIDINIT` : A 1-character, not null column with data type CHAR.
4. `LASTNAME` : A 15-character, not null column with data type VARCHAR.
5. `WORKDEPT` : A 3-character, nullable column with data type CHAR and a constraint named 'RED'.
6. `PHONE NO` : A 4-character, nullable column with data type CHAR.
7. `HIREDATE` : A DATE column with no null values.
8. `JOB` : An 8-character, nullable column with data type CHAR.
9. `EDLEVEL` : A small integer (SMALLINT) column with a scale of 0 and a precision of 4, not null.
10. `SEX` : A 1-character, nullable column with data type CHAR.
11. `BIRTHDATE` : A DATE column with no null values.
12. `SALARY` : A decimal column with a scale of 2 and a precision of 9, nullable.
13. `BONUS` : A decimal column with a scale of 2 and a precision of 9, nullable.
14. `COMM` : A decimal column with a scale of 2 and a precision of 9, nullable.

□ ⌘ ⌘ ⌘

Ask a follow-up

Autodetect (1) - granite3-dense... @

⌘ ⇣ @codebase

↔ Enter

This screenshot shows a Microsoft Visual Studio Code (VS Code) interface with an integrated AI extension. The top bar displays the title '[Extension Development Host] vscode'. The main area contains a code editor with the following SQL query:

```
SELECT E.EMPNO, E.FIRSTNME, E.LASTNAME, Untitled-1
1  SELECT E.EMPNO, E.FIRSTNME, E.LASTNAME, D.DEPTNAME
2  FROM EMPLOYEE E JOIN DEPARTMENT D ON E.WORKDEPT = D.DEPTNO;
3
```

The code editor includes standard VS Code navigation and search tools. Below the code editor is a toolbar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, IBM I (which is selected), GITLENS, and COMMENTS.

A large arrow points from the 'RESULTS' tab in the bottom-left corner towards the code editor, indicating the results of the executed query.

The results are displayed in a table:

EMPNO	FIRSTNME	LASTNAME	DEPTNAME
000010	CHRISTINE	HAAS	SPIFFY COMPUTER SERVICE DIV.
000110	VINCENZO	LUCCHESI	SPIFFY COMPUTER SERVICE DIV.
000120	SEAN	O'CONNELL	SPIFFY COMPUTER SERVICE DIV.
200010	DIAN	HEMMINGER	SPIFFY COMPUTER SERVICE DIV.
200120	GREG	ORLANDO	SPIFFY COMPUTER SERVICE DIV.
000020	MICHAEL	THOMPSON	PLANNING
000030	SALLY	KWAN	INFORMATION CENTER
000130	DELORES	QUINTANA	INFORMATION CENTER
000140	HEATHER	NICHOLLS	INFORMATION CENTER
200140	KIM	NATZ	INFORMATION CENTER
000060	IRVING	STERN	MANUFACTURING SYSTEMS
000150	BRUCE	ADAMSON	MANUFACTURING SYSTEMS
000160	ELIZABETH	PIANKA	MANUFACTURING SYSTEMS
000170	MASATOSHI	YOSHIMURA	MANUFACTURING SYSTEMS
000180	MARILYN	SCOUTTEN	MANUFACTURING SYSTEMS
000190	JAMES	WALKER	MANUFACTURING SYSTEMS
000200	DAVID	BROWN	MANUFACTURING SYSTEMS
000210	WILLIAM	JONES	MANUFACTURING SYSTEMS
000220	JENNIFER	LUTZ	MANUFACTURING SYSTEMS

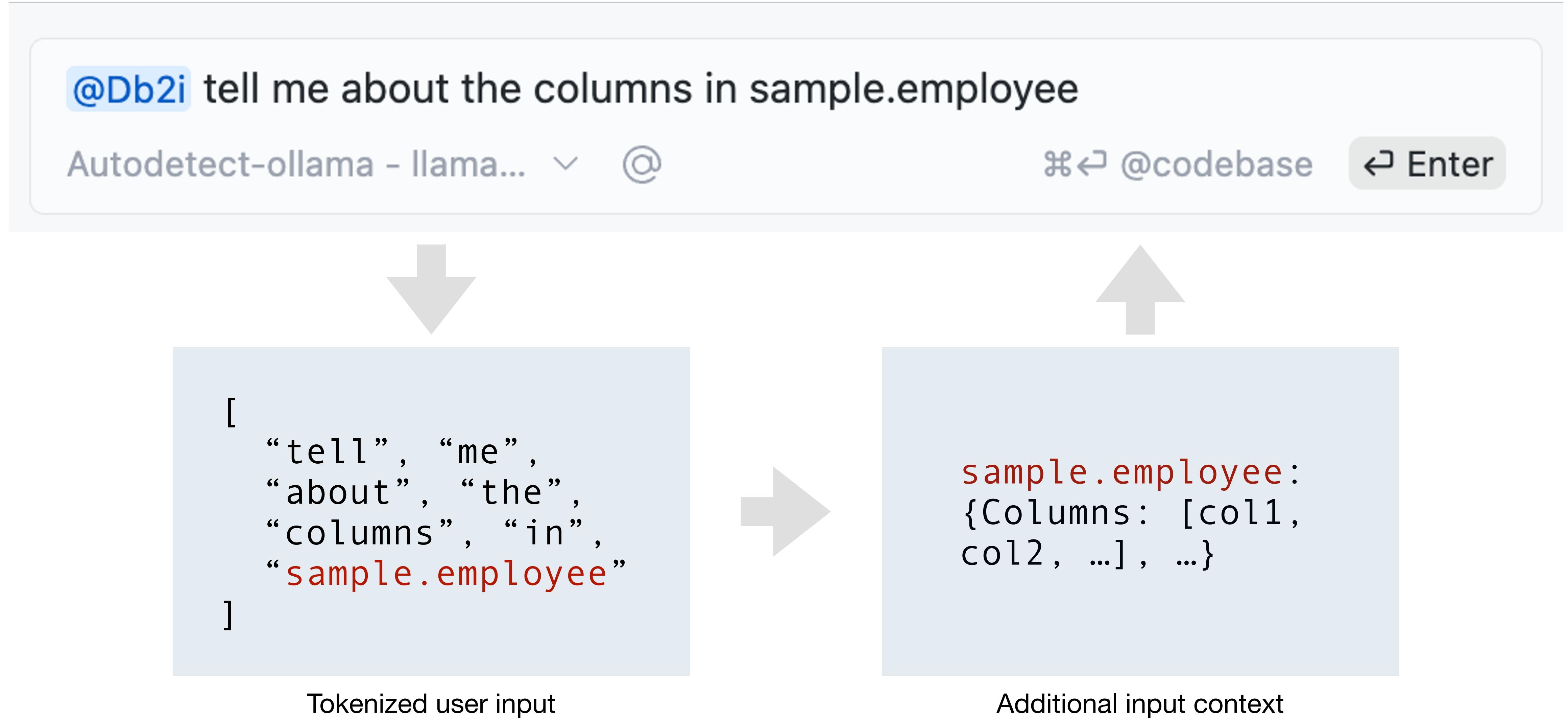
To the right of the results, there is an AI chat interface. The user asks: '@Db2i how can I find the department name for each employee?'. The AI responds by providing the SQL query and explaining its purpose:

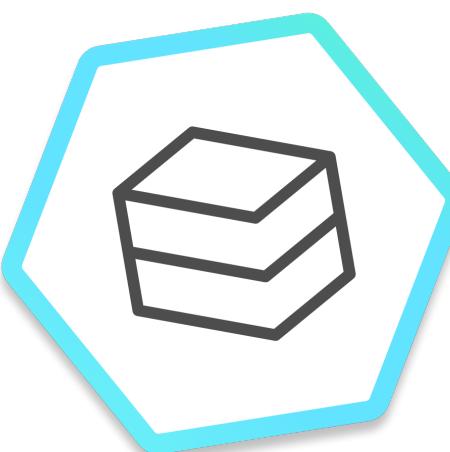
To find the department name for each employee in Db2 for i, you can use a SQL query that joins the EMPLOYEE and DEPARTMENT tables based on the WORKDEPT column. Here's an example query:

```
SELECT E.EMPNO, E.FIRSTNME, E.LASTNAME, D.DEPTNAME
FROM EMPLOYEE E JOIN DEPARTMENT D ON E.WORKDEPT = D.DEPTNO;
```

The AI also provides a section for asking follow-up questions and a link to start a new session.

Prompt enrichment in play





← → 🔍 [Extension Development Host] sql

CHAT + ⏷ ⏵ ...

ajshedivy
@db2i write me an SQL statement that calculates the average employee salary in each department

Untitled-1:1-36

Db2 for i
Sure, here is an SQL statement that calculates the average employee salary in each department:

```
SELECT
    e.WORKDEPT AS Department,
    AVG(e.SALARY) AS AverageSalary
FROM
    SAMPLE.EMPLOYEE e
GROUP BY
    e.WORKDEPT;
```

This query groups the employees by their department and calculates the average salary for each department.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL IBM I ...

RESULTS

DEPARTMENT	AVERAGESALARY
E11	21020
A00	40850
D21	25668.571428571428
C01	29722.5
E01	40175
D11	25147.272727272728

[Extension Development Host] sql

) FROM EMPLOYEE GROUP BY DEPT

CONTINUE

Untitled-2 (1-131)

```
{  
    "JOB_NAME": "683585/QUSER/QZDASOINIT",  
    "USER_NAME": "ASHEDIVY",  
    "REASON_CODE": null,  
    "LOGGED_TIME": "2024-10-21 18:46:15.831000",  
    "LOGGED_SQLSTATE": "42703",  
    "LOGGED_SQLCODE": -206,  
    "MATCHES": 1,  
    "STMTTEXT": "SELECT DEPT, COUNT(*) FROM EMPLOYEE GROUP BY  
DEPT",  
    "MESSAGE_TEXT": "Column or global variable &1 not found.",  
    "MESSAGE_SECOND_LEVEL_TEXT": "Cause . . . . . : &1 was not  
found as a column of table &2 in &3 and was not found as a  
global variable in &2. If the table is *N, &1 is not a column  
of any table or view that can be referenced, or &1 is a special  
register that cannot be set in an atomic compound statement. ✓  
Recovery . . . . : Do one of the following and try the request
```

@Db2i break down this error for me

Autodetect-ollama - llama... @

⌘ ⇧ @codebase ⌘ ⇧ Enter

CONSOLE TERMINAL IBM I COMMENTS

variable DEPT not found.,

SQL ERROR LOGGING FACILITY (SELF)

- > ⚠ 01545 (12) Correlation without qualification occurred for column &1 to table &2. 7
- > ✗ 42703 (-206) Column or global variable &1 not found. 1
 - Statement SELECT DEPT, COUNT(*) FROM EMPLOYEE GROUP BY DEPT
 - Job 683585/QUSER/QZDASOINIT
 - Client Name vscode-db2i 1.6.0
 - Client Program VSCode connector | Version 2.1.4
 - Object QSYS/QZDASRV (*SRVPGM, QZDASQL)
- > ✗ 42601 (-104) Token &1 was not valid. Valid tokens: &2. 2

Let's take a closer look at the error message and stack trace.

Error Message:

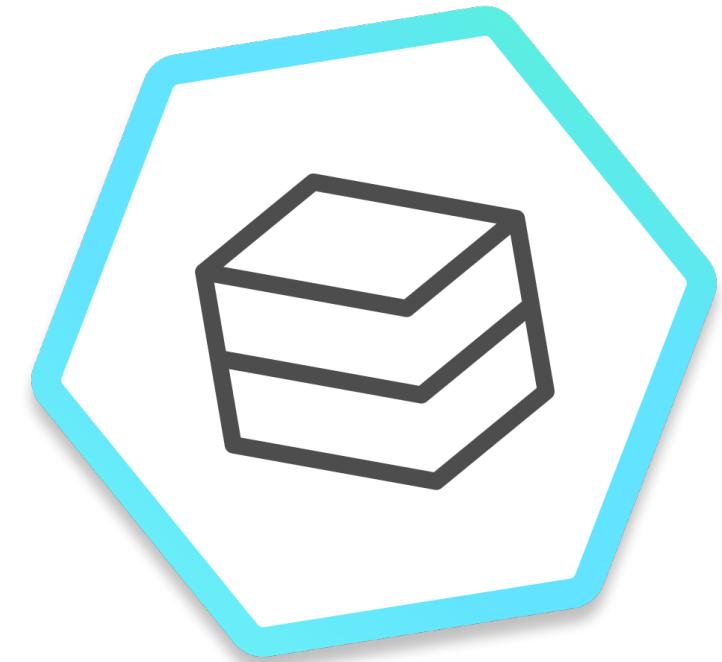
Column or global variable &1 not found.

This is the actual error message, indicating that there's a problem with a column reference (&1) in the SQL statement.

SQL Statement:

SELECT DEPT, COUNT(*) FROM EMPLOYEE GROUP BY DEPT

The SQL statement being executed is a simple **SELECT** statement that groups results by department (DEPT) and counts the number of employees in each department. The error occurs because the column reference 'DEPT' is not properly qualified with a table name, causing the database to search for it as a global variable or a column in another table, which is not found.



codefori.github.io/docs/extensions/db2i/

**Db2 for IBM i,
AI for applications**

 README

 GPL-3.0 license



WatsonX-SDK-Db2-IBMi

WatsonX client SDK for Db2 on IBM i

Usage

Login

```
call watsonx.logoutjob();
call watsonx.setapikeyforjob('x');
call watsonx.setprojectidforjob('y');

-- Should return Y
values watsonx.ShouldGetNewToken();

-- Should return Y
values watsonx.authenticate();

-- Should return N
values watsonx.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 ^ X

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-12-v2	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* oss74dev Deploy New job (1) + Ln 1, Col 26 Spaces: 2 LF {} MS SQL

The screenshot shows a code editor interface with the following details:

- File:** a.sql
- Content:**

```
1 call watsonx.setapikeyforjob('EIdSiJwJyqC4FGFUBC7zpMrpo7Wggs8kH4a5FrSh-Qt6');
2 call watsonx.setprojectidforjob('f2872353-8bf4-4c8f-969d-04dcee0198ef');
3
4 -- Should return Y
5 values watsonx.authenticate();
```

- Execution Results:** The results are displayed in the "RESULTS" section of the interface.

 - Line 1:** 00001
 - Line 2:** Y (highlighted with a red arrow)
 - Message:** Loaded 1. End of data.
 - User ID:** 606512/QUSER/QZDASOINIT

- Bottom Status Bar:** Shows file status (main*), deployment status (oss74dev), job count (0), and other system information (Ln 5, Col 30 (29 selected), Spaces: 2, LF, MS SQL).

The screenshot shows a code editor interface with the following details:

- Title Bar:** WatsonX-SDK-Db2-IBMi
- File Tab:** a.sql (M)
- Code Editor:** Contains two lines of code:

```
1 values watsonx.generate('Hello world');
2 values watsonx.generate('Hello world', model_id => 'ibm/granite-3-8b-instruct');
```
- Toolbar:** Includes icons for file operations (New, Open, Save, etc.) and a search bar.
- Bottom Navigation:** PROBLEMS, IBM I (selected), TERMINAL, OUTPUT, DEBUG CONSOLE, RUN AND DEBUG, PORTS, COMMENTS.
- RESULTS Panel:** Shows the output of the code execution:
 - Message: "00001" (highlighted in grey).
 - Text: "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 mistakes."
 - Text: "Loaded 1. End of data."
 - User ID: 606512/QUSER/QZDASOINIT
- Bottom Status Bar:** Includes icons for main*, oss74dev, Deploy, New job (1), Ln 1, Col 39 (38 selected), Spaces: 2, LF, {}, MS SQL, and a refresh icon.

WatsonX-SDK-Db2-IBMi

a.sql M WATSONX.SQLRPGLE ...

LIAMA > QRPGLESRC > 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 TERMINAL OUTPUT DEBUG CONSOLE RUN AND DEBUG PORTS COMMENTS Call program + ×

Library list: SAMPLE QTEMP QSYSINC LIAMA ILEASIC
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* oss74dev Deploy New job (1) Ln 16, Col 43 Spaces: 2 LF RPGLE



Watsonx Code Assistant

for IBM i

Statement of direction: code assistant for RPG

Published: 21 October 2024 AD24-2179 [Statement of direction](#)

Statement of direction

IBM intends to deliver a code assistant for RPG - a generative AI tool which helps developers of IBM i software understand existing RPG code, create new RPG function using natural language description, and automatically generate test cases for RPG code.

per IBM

**free Untitled-1

```
49 Dow (Not Exit);
50   Select;
51
52   Endsl;
53
54   Enddo;
55
56 *INLR = *ON;
57
58 Return;
59
60 //-----
61
62 Explain | Summarise | Document
63 Dcl-Proc ClearSubfile;
64   . SflDspCtl = *Off;
65   . SflDsp = *Off;
66
67   . Write SFLCTL;
68
69   . SflDspCtl = *On;
70
71   . rrn = 0;
72
73 End-Proc;
74
75 Explain | Summarise | Document
76 Dcl-Proc LoadSubfile;
77   Dcl-S lCount Int(5);
```

WATSONX CODE ASSISTANT: CHAT

Chat session 10

watsonx

Hi, how can I help you today?

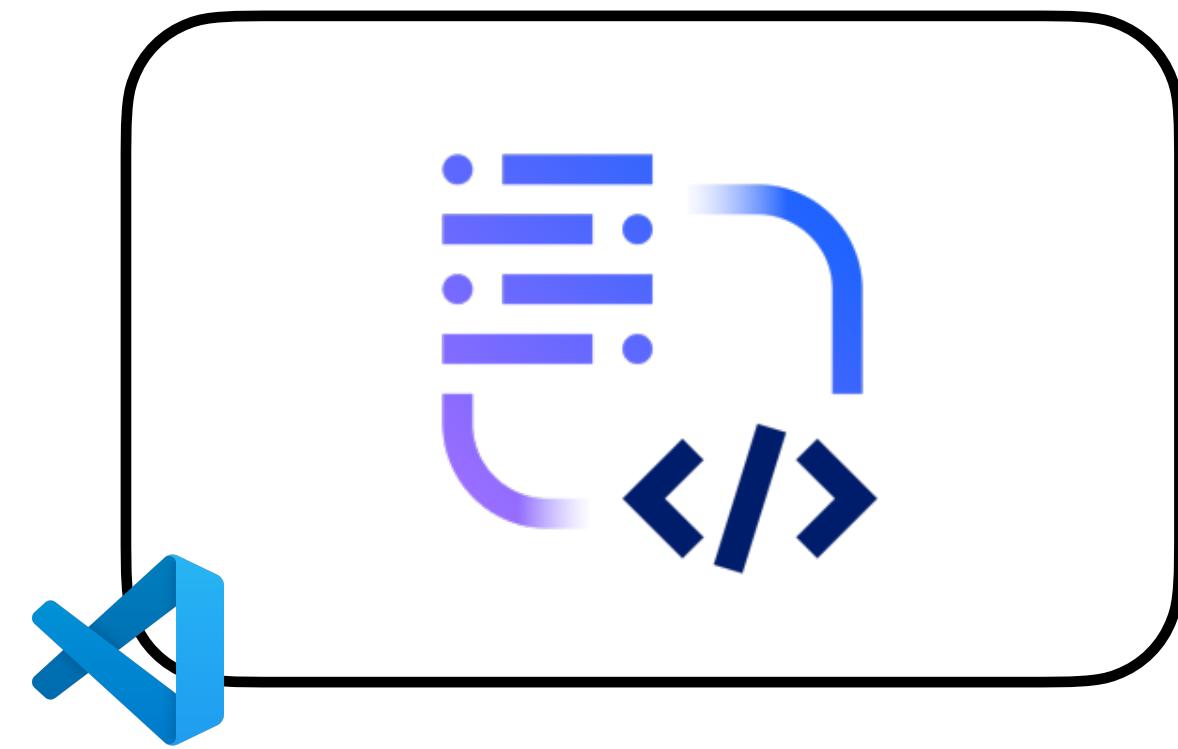
You

Explain the selected code... :

watsonx

The selected code is a procedure called ClearSubfile. It is used to clear the contents of a subfile. The code does not have any parameters or variables. It simply uses the CLRF instruction to clear the subfile.

Enter / for commands ➤



Building our own
RPGLE model

ILE RPGLE only

Fixed-format to
free-format long term

OPM to be supported

Initially VS Code only

RPGLE to another
language not supported

COBOL to be considered
down the line

Explain and document
RPGLE code first

Explain and fix compile
errors coming later

Generate new code
coming later

Would you like to get involved?

Join us in creating the new RPG code assistant for
IBM i.

Engage with our team across various research activities,
including but not limited to discussions and workshops to
help define the offering, alpha & beta testing of the offering
pre-GA, and submitting source code to help train and/or
test the RPG code assistant model.

Your engagement helps [shape the future of IBM Power](#).

1. Advisor
2. Code Contributor

aforibmi@ibm.com



Alpha ready. Model evaluation by contributors

End of Q1, 2025

Explain functionality show publicly

~Q2~, 2025

Model will be made available to group of people

~Q2~, 2025

GA for the first version of WCA4i

2H~, 2025



**RPGLE,
AI for development**