Python RAG Tutorial (with Local LLMs): AI For Your PDFs

pixegami
75.2K subscribers
Subscribe

15K | Share | Ask | Download | ...

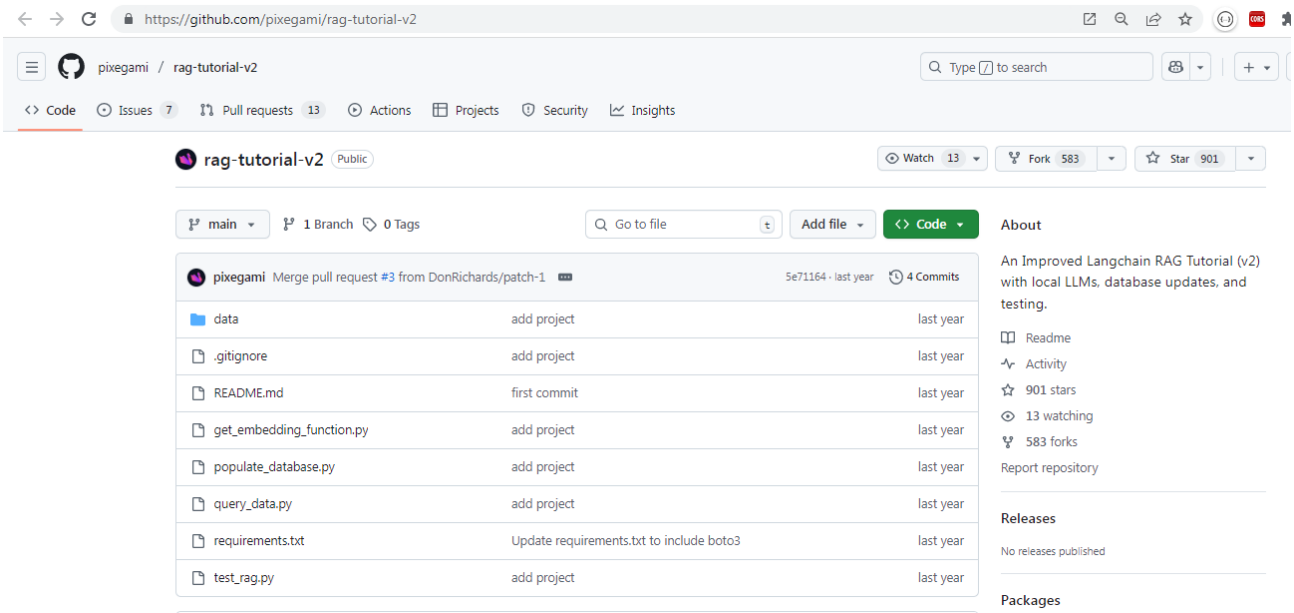
566,164 views Apr 17, 2024
Learn how to build a RAG (Retrieval Augmented Generation) app in Python that can let you query/chat with your PDFs using generative AI.

This project contains some more advanced topics, like how to run RAG apps locally (with Ollama), how to update a vector DB with new items, how to use RAG with PDFs (or any other files), and how to test the quality of AI generated responses.

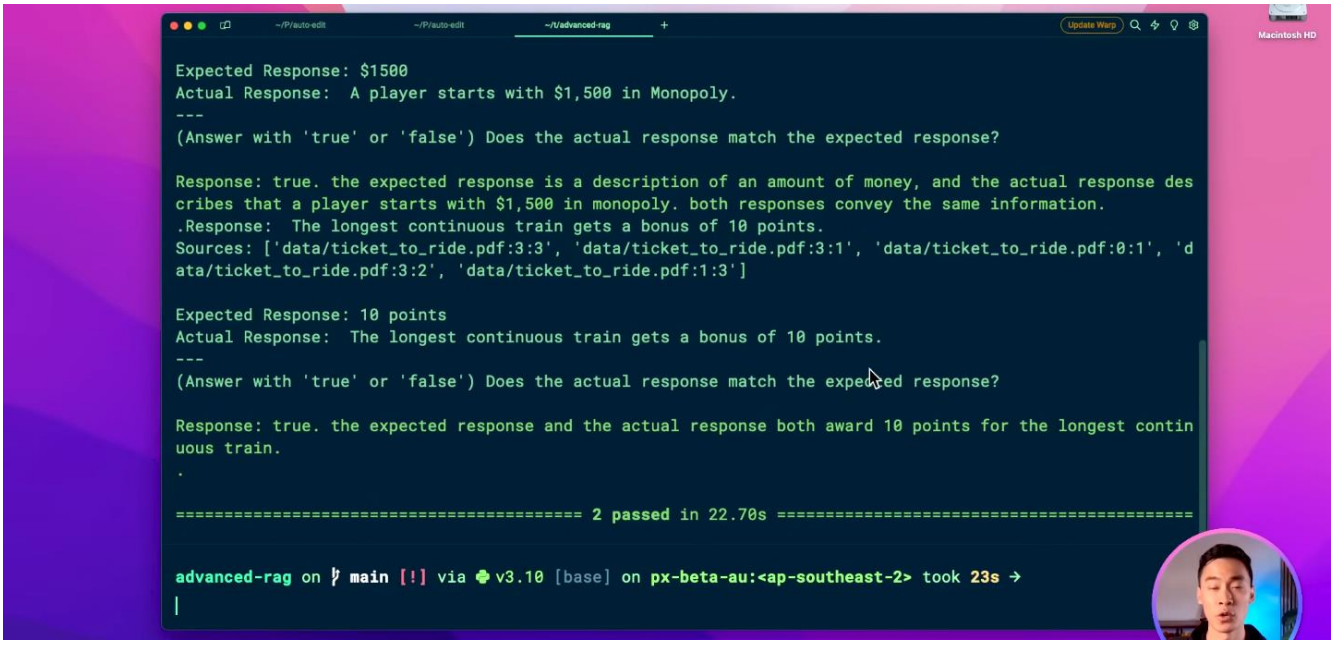
☞ Links
⇜ GitHub: https://github.com/pixegami/rag-tutor...
⇜ Basic RAG Tutorial: • RAG + Langchain Python Project: Easy AI/Ch...
⇜ PyTest Video: • How To Write Unit Tests in Python • Pytest...

☞ Resources
⇜ Document loaders: https://python.langchain.com/docs/mod...
⇜ PDF Loader: https://python.langchain.com/docs/mod...
⇜ Ollama: https://ollama.com

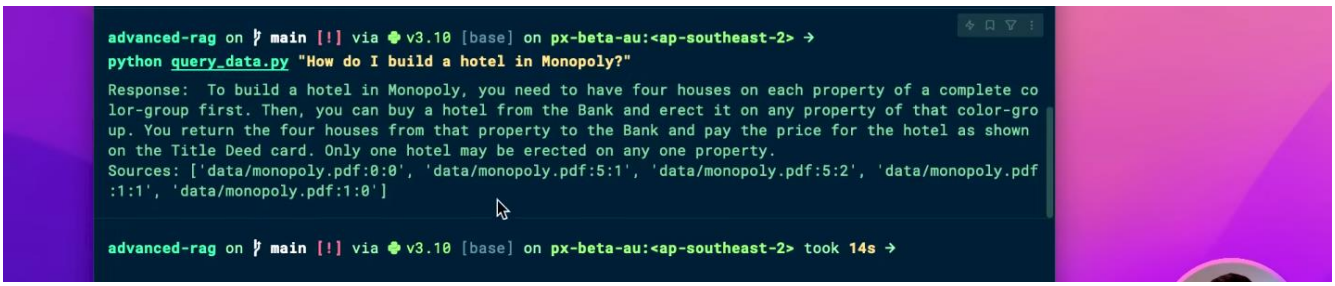
🕮 Chapters
00:00 Introduction
01:06 RAG Recap
03:22 Loading PDF Data
05:08 Generate Embeddings
07:16 How To Store and Update Data
10:46 Updating Database
11:45 Running RAG Locally
15:12 Unit Testing AI Output
20:29 Wrapping Up

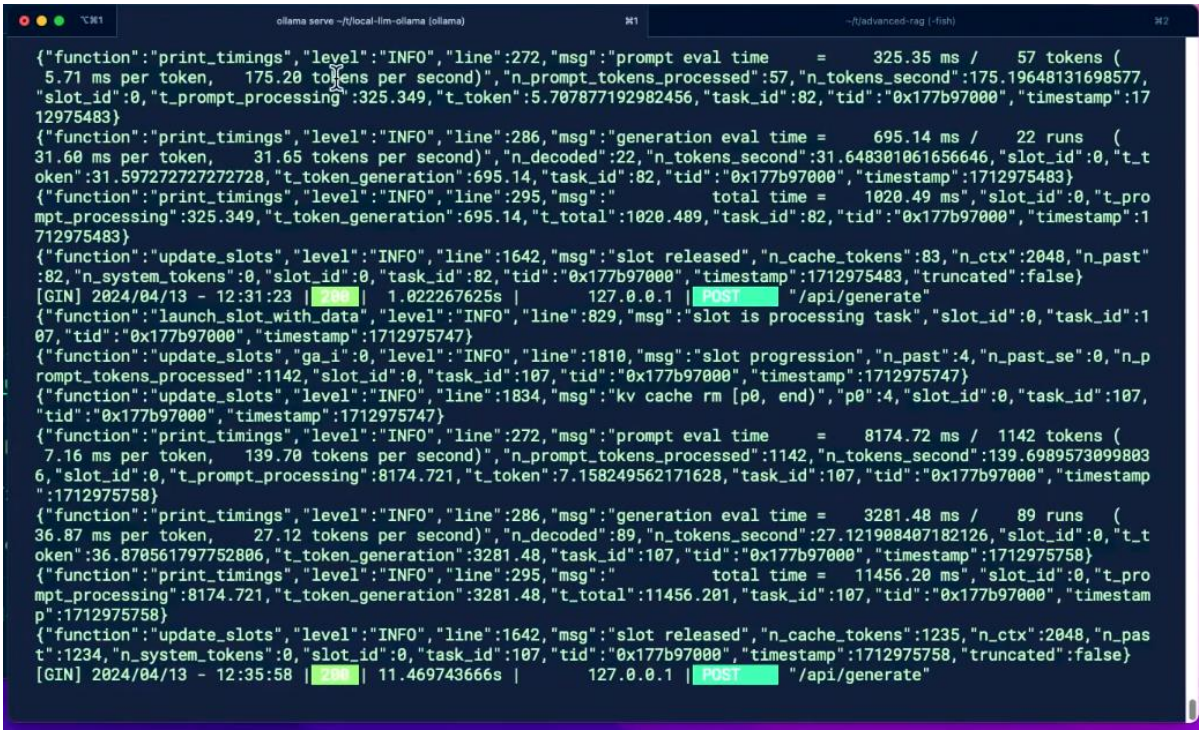


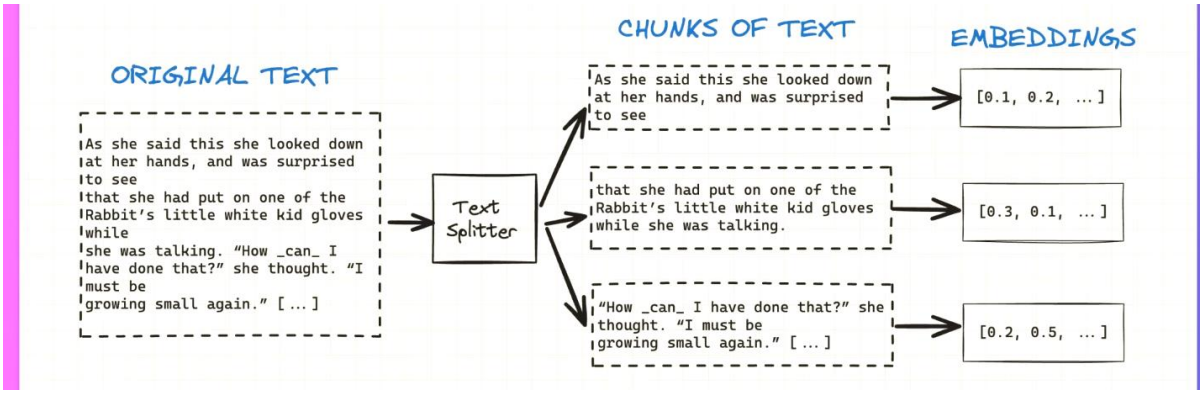
```
~$ python query_data.py "How do I build a hotel in Monopoly?"

Response:  To build a hotel, you must first have all
houses built on every property of a completed color group.
Then, you can buy a hotel from the Bank and erect it on any property
of that color group.

Sources: ['data/monopoly.pdf:0:0', 'data/monopoly.pdf:5:1']
```

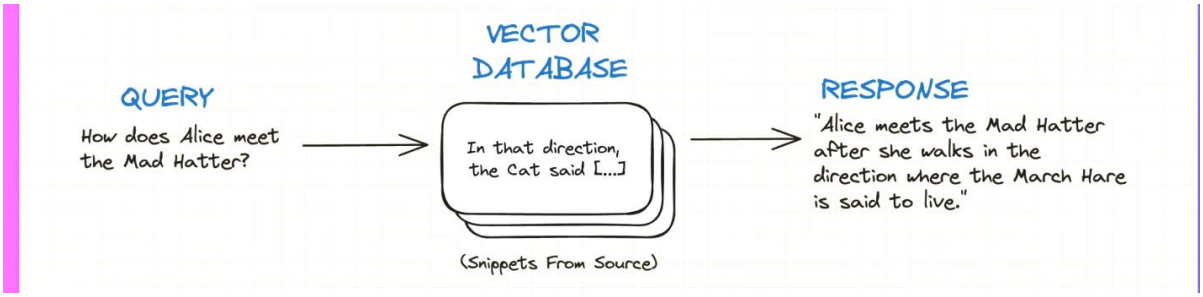


RETRIEVAL AUGMENTED GENERATION





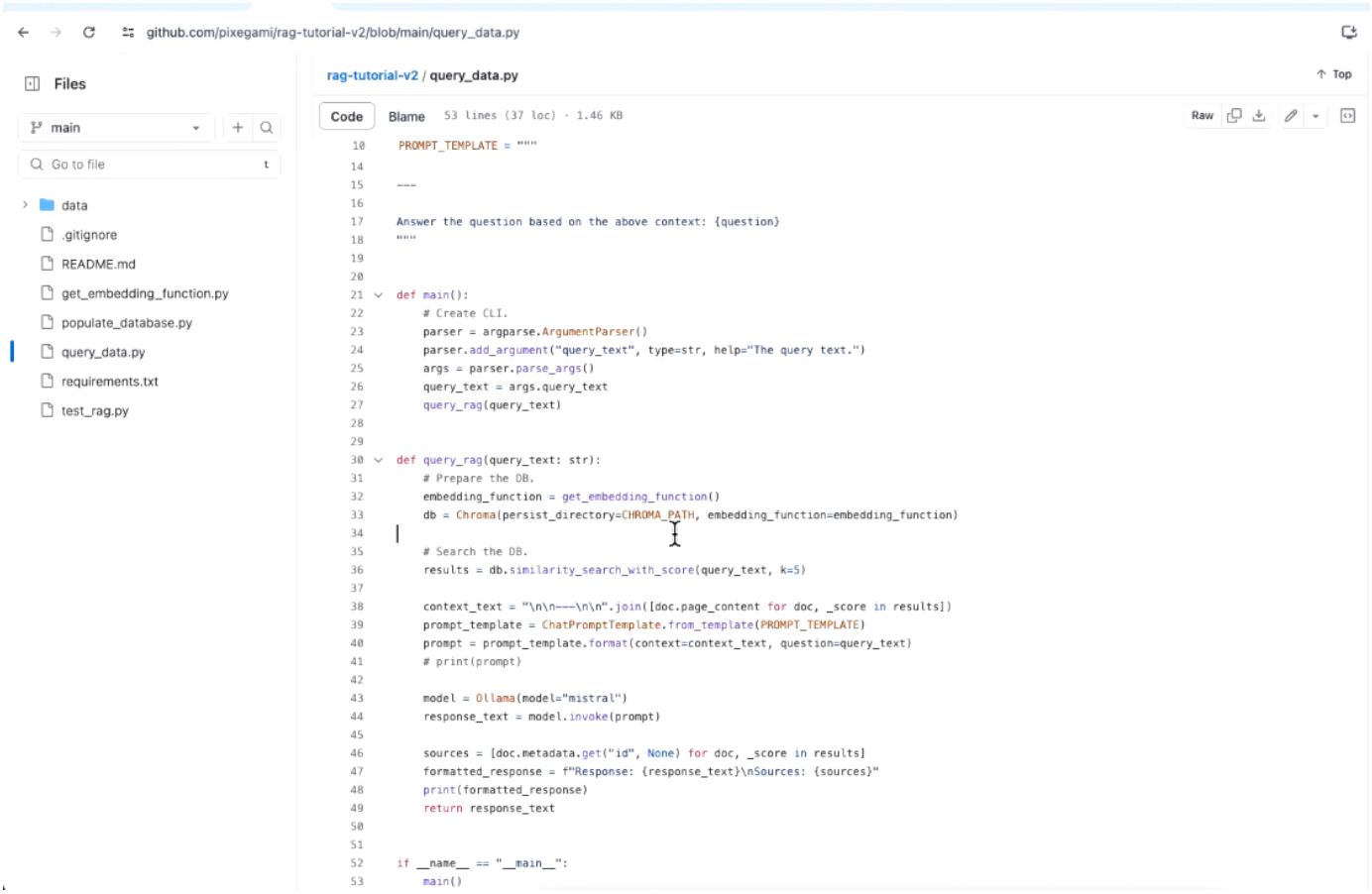




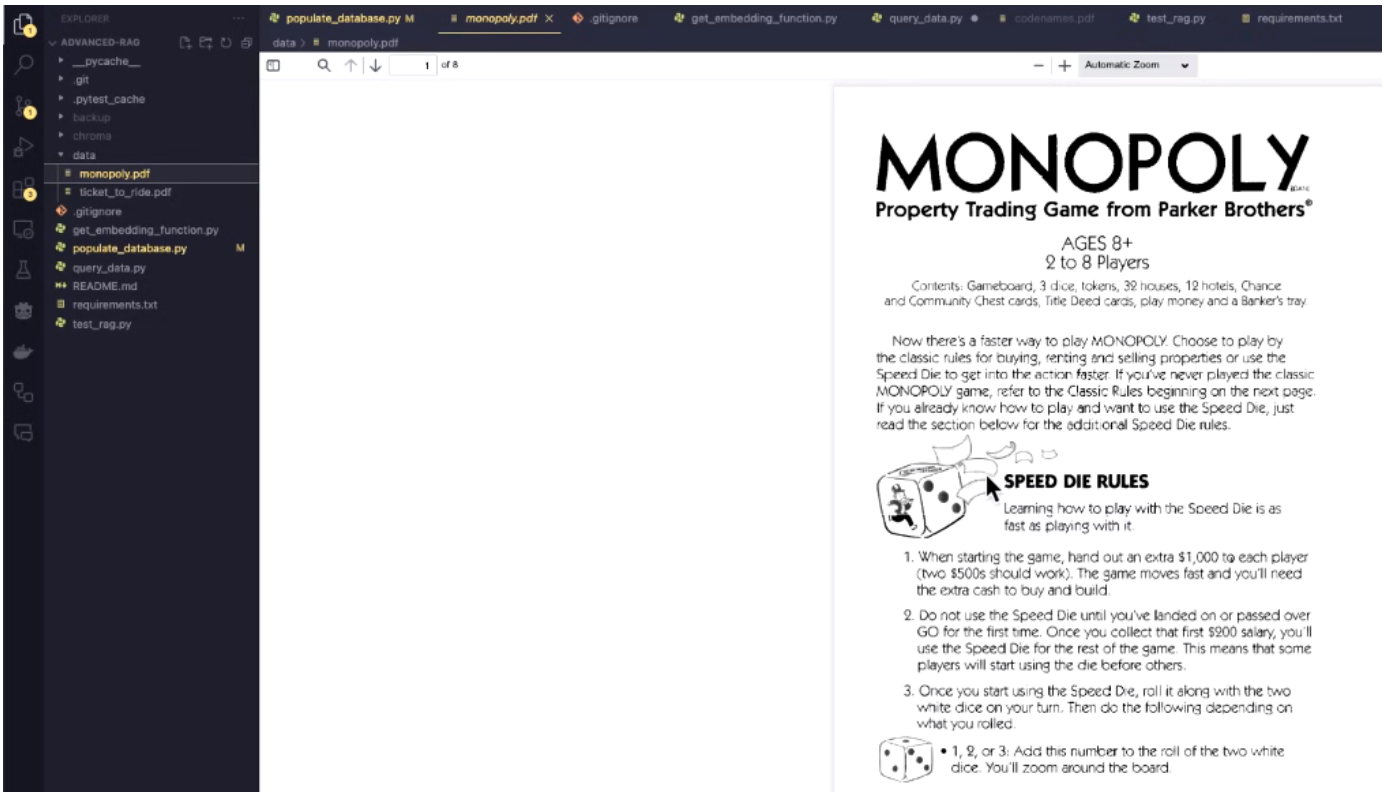
👉 Running RAG Locally

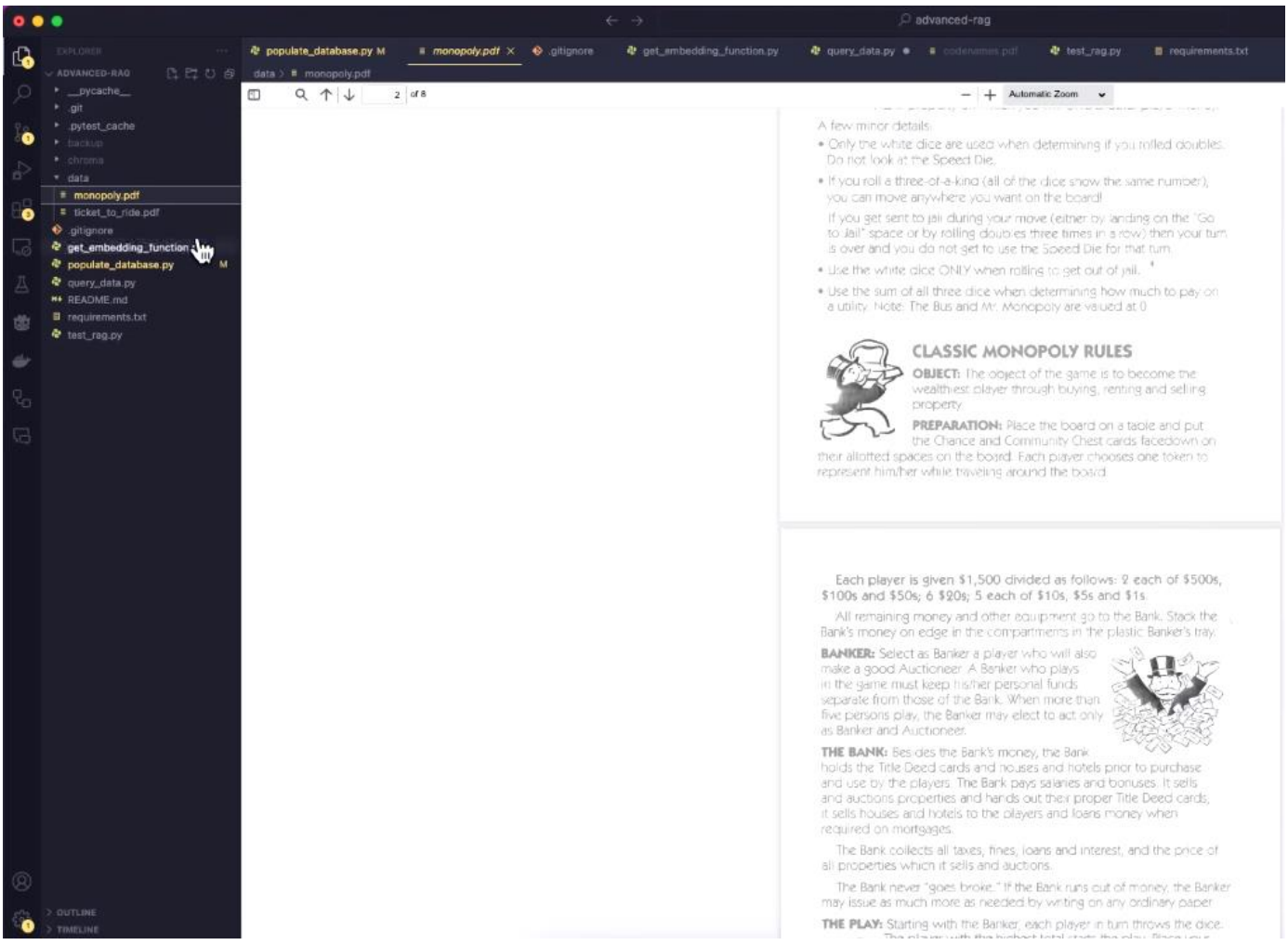
👉 Updating Your Database

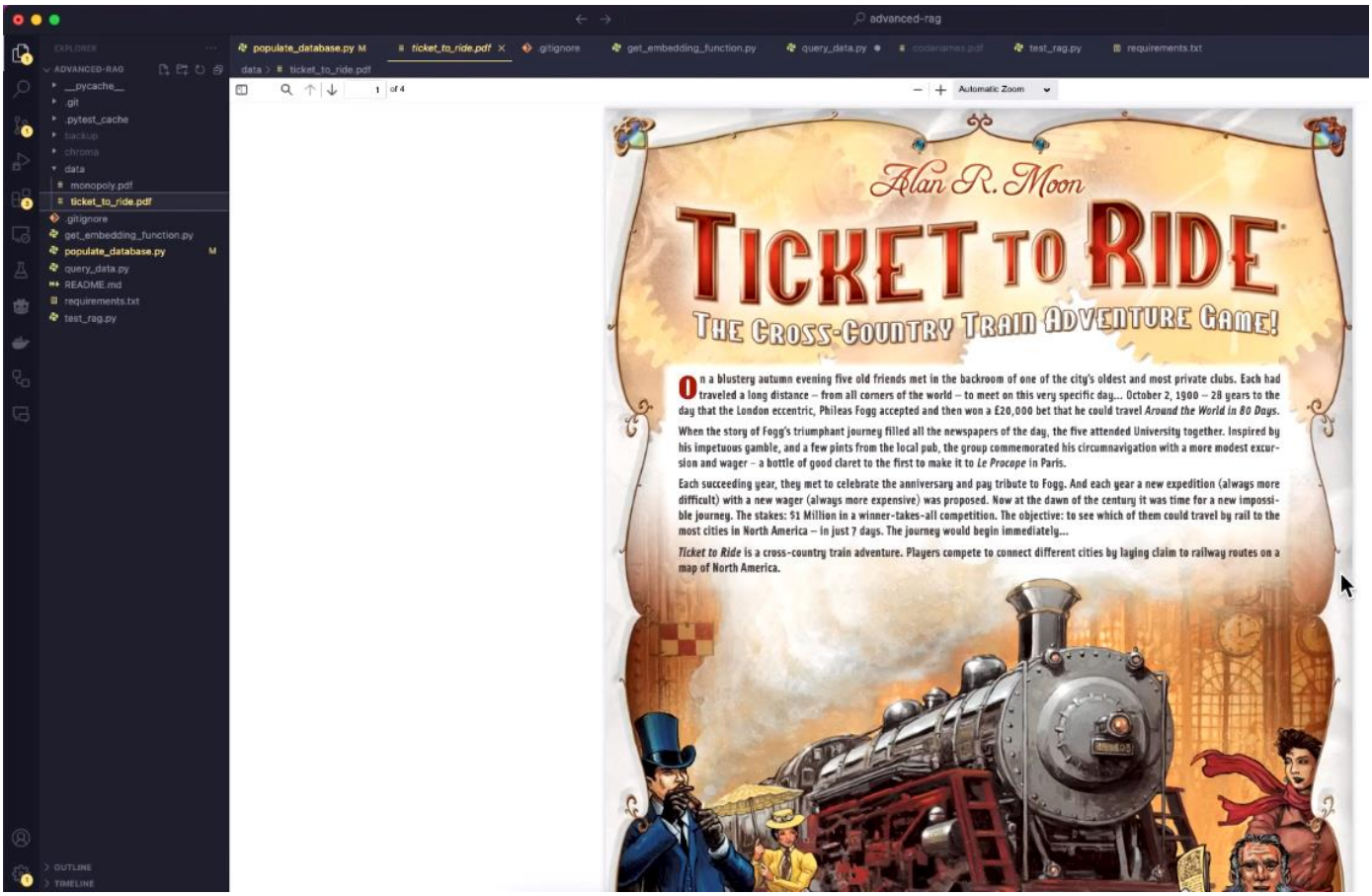
👉 Testing RAG Output

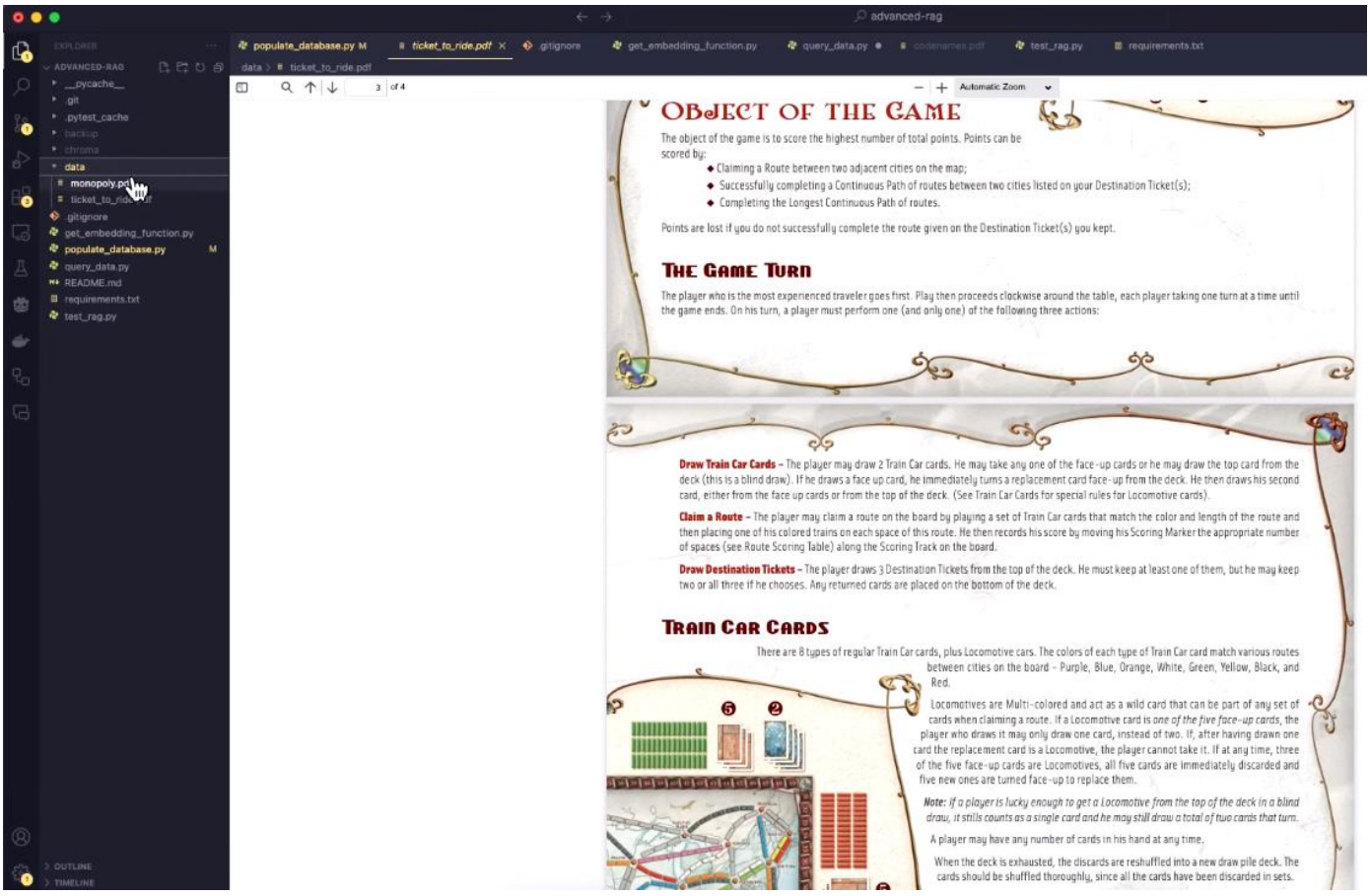


LOADING THE DATA


EXPLORER
ADVANCED-RAG
__pycache__
.git
.pytest_cache
backup
chroma
data
monopoly.pdf
ticket_to_ride.pdf
.gitignore
get_embedding_function.py
populate_database.py M
query_data.py
README.md
requirements.txt
test_rag.py
populate_database.py M
monopoly.pdf
.gitignore
get_embedding_function.py
query_data.py
codenames.pdf
test_rag.py
requirements.txt
data > monopoly.pdf
1 of 8
Automatic Zoom
MONOPOLY
Property Trading Game from Parker Brothers®
AGES 8+
2 to 8 Players
Contents: Gameboard, 3 dice, tokens, 32 houses, 12 hotels, Chance and Community Chest cards, Title Deed cards, play money and a Banker's tray
Now there's a faster way to play MONOPOLY. Choose to play by the classic rules for buying, renting and selling properties or use the Speed Die to get into the action faster. If you've never played the classic MONOPOLY game, refer to the Classic Rules beginning on the next page. If you already know how to play and want to use the Speed Die, just read the section below for the additional Speed Die rules.
SPEED DIE RULES
Learning how to play with the Speed Die is as fast as playing with it.
1. When starting the game, hand out an extra $1,000 to each player (two $500s should work). The game moves fast and you'll need the extra cash to buy and build.
2. Do not use the Speed Die until you've landed on or passed over GO for the first time. Once you collect that first $200 salary, you'll use the Speed Die for the rest of the game. This means that some players will start using the die before others.
3. Once you start using the Speed Die, roll it along with the two white dice on your turn. Then do the following depending on what you rolled.
• 1, 2, or 3: Add this number to the roll of the two white dice. You'll zoom around the board.

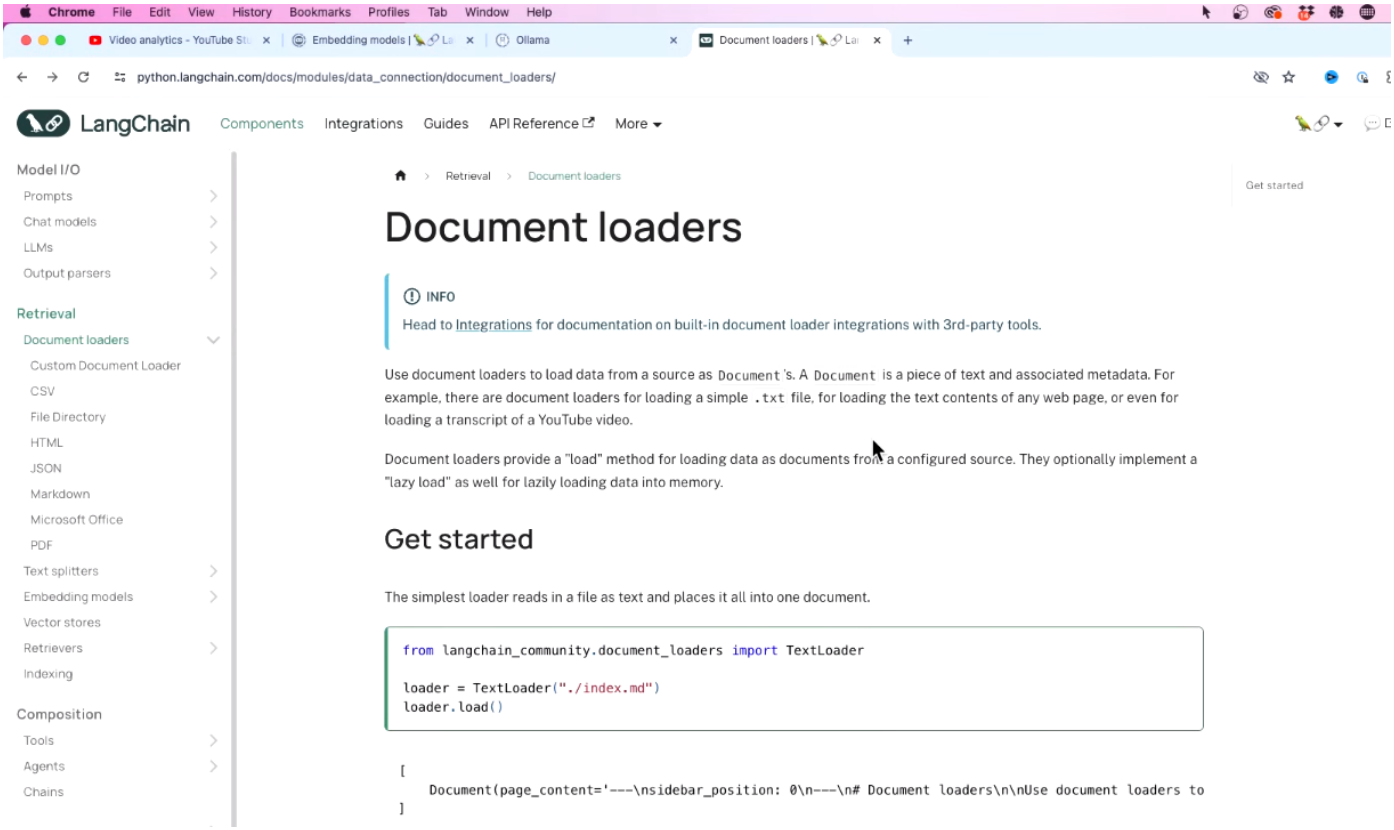

advanced-rag
EXPLORER
ADVANCED-RAG
__pycache__
.git
.pytest_cache
backup
chroma
data
monopoly.pdf
ticket_to_ride.pdf
.gitignore
get_embedding_function.py
populate_database.py M
query_data.py
README.md
requirements.txt
test_rag.py
OUTLINE
TIMELINE
populate_database.py M
monopoly.pdf
.gitignore
get_embedding_function.py
query_data.py
codenames.pdf
test_rag.py
requirements.txt
data > monopoly.pdf
2 of 8
Automatic Zoom
A few minor details:
• Only the white dice are used when determining if you rolled doubles. Do not look at the Speed Die.
• If you roll a three-of-a-kind (all of the dice show the same number), you can move anywhere you want on the board!
If you get sent to jail during your move (either by landing on the "Go to Jail" space or by rolling doubles three times in a row) then your turn is over and you do not get to use the Speed Die for that turn.
• Use the white dice ONLY when rolling to get out of jail.
• Use the sum of all three dice when determining how much to pay on a utility. Note: The Bus and Mr. Monopoly are valued at 0.
CLASSIC MONOPOLY RULES
OBJECT: The object of the game is to become the wealthiest player through buying, renting and selling property.
PREPARATION: Place the board on a table and put the Chance and Community Chest cards facedown on their allotted spaces on the board. Each player chooses one token to represent him/her while traveling around the board.
Each player is given $1,500 divided as follows: 2 each of $500s, $100s and $50s; 6 $20s; 5 each of $10s, $5s and $1s.
All remaining money and other equipment go to the Bank. Stack the Bank's money on edge in the compartments in the plastic Banker's tray.
BANKER: Select as Banker a player who will also make a good Auctioneer. A Banker who plays in the game must keep his/her personal funds separate from those of the Bank. When more than five persons play, the Banker may elect to act only as Banker and Auctioneer.
THE BANK: Besides the Bank's money, the Bank holds the Title Deed cards and houses and hotels prior to purchase and use by the players. The Bank pays salaries and bonuses. It sells and auctions properties and hands out their proper Title Deed cards; it sells houses and hotels to the players and loans money when required on mortgages.
The Bank collects all taxes, fines, loans and interest, and the price of all properties which it sells and auctions.
The Bank never "goes broke." If the Bank runs out of money, the Banker may issue as much more as needed by writing on any ordinary paper.
THE PLAY: Starting with the Banker, each player in turn throws the dice.


advanced-rag
populate_database.py M
ticket_to_ride.pdf
gitignore
get_embedding_function.py
query_data.py
codenames.pdf
test_rag.py
requirements.txt
ADVANCED-RAG
__pycache__
.git
.pytest_cache
backup
chroma
data
monopoly.pdf
ticket_to_ride.pdf
.gitignore
get_embedding_function.py
populate_database.py
query_data.py
README.md
requirements.txt
test_rag.py
Alan R. Moon
TICKET TO RIDE
THE CROSS-COUNTRY TRAIN ADVENTURE GAME!
On a blustery autumn evening five old friends met in the backroom of one of the city's oldest and most private clubs. Each had traveled a long distance – from all corners of the world – to meet on this very specific day... October 2, 1900 – 28 years to the day that the London eccentric, Phileas Fogg accepted and then won a £20,000 bet that he could travel Around the World in 80 Days.
When the story of Fogg's triumphant journey filled all the newspapers of the day, the five attended University together. Inspired by his impetuous gamble, and a few pints from the local pub, the group commemorated his circumnavigation with a more modest excursion and wager – a bottle of good claret to the first to make it to Le Procope in Paris.
Each succeeding year, they met to celebrate the anniversary and pay tribute to Fogg. And each year a new expedition (always more difficult) with a new wager (always more expensive) was proposed. Now at the dawn of the century it was time for a new impossible journey. The stakes: $1 Million in a winner-takes-all competition. The objective: to see which of them could travel by rail to the most cities in North America – in just 7 days. The journey would begin immediately...
Ticket to Ride is a cross-country train adventure. Players compete to connect different cities by laying claim to railway routes on a map of North America.

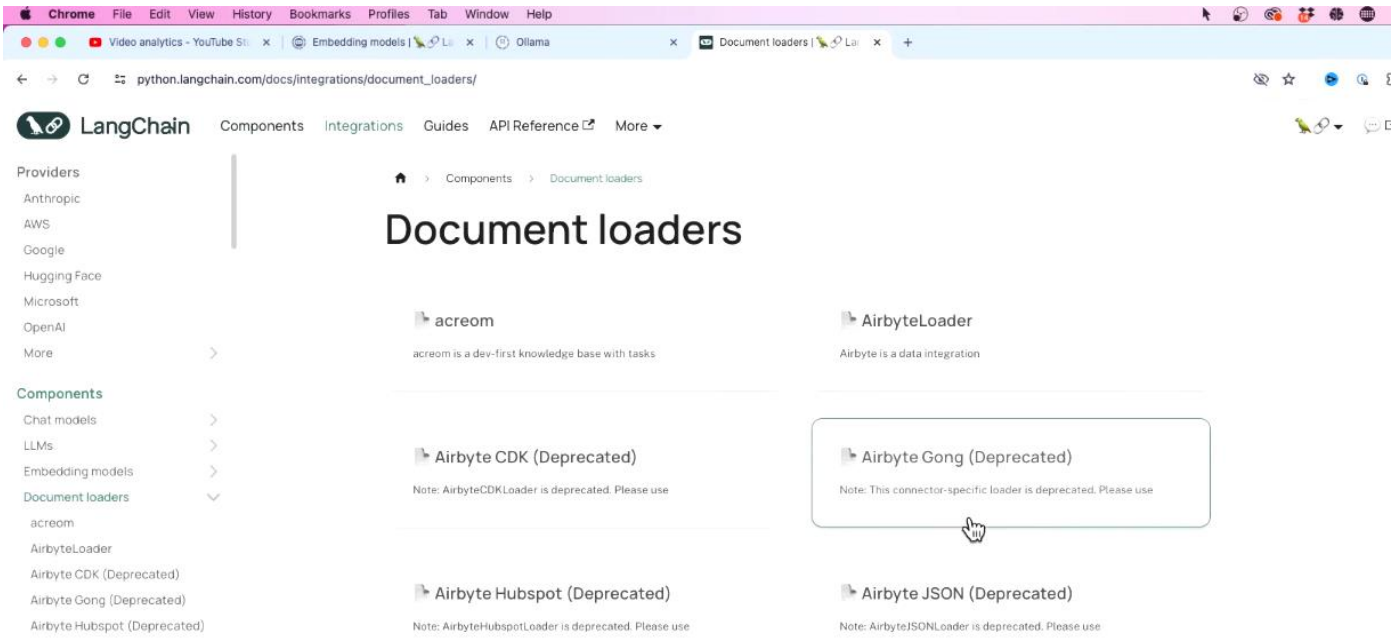

advanced-rag
OBJECT OF THE GAME
The object of the game is to score the highest number of total points. Points can be scored by:
Claiming a Route between two adjacent cities on the map;
Successfully completing a Continuous Path of routes between two cities listed on your Destination Ticket(s);
Completing the Longest Continuous Path of routes.
Points are lost if you do not successfully complete the route given on the Destination Ticket(s) you kept.
THE GAME TURN
The player who is the most experienced traveler goes first. Play then proceeds clockwise around the table, each player taking one turn at a time until the game ends. On his turn, a player must perform one (and only one) of the following three actions:
Draw Train Car Cards – The player may draw 2 Train Car cards. He may take any one of the face-up cards or he may draw the top card from the deck (this is a blind draw). If he draws a face up card, he immediately turns a replacement card face-up from the deck. He then draws his second card, either from the face up cards or from the top of the deck. (See Train Car Cards for special rules for Locomotive cards).
Claim a Route – The player may claim a route on the board by playing a set of Train Car cards that match the color and length of the route and then placing one of his colored trains on each space of this route. He then records his score by moving his Scoring Marker the appropriate number of spaces (see Route Scoring Table) along the Scoring Track on the board.
Draw Destination Tickets – The player draws 3 Destination Tickets from the top of the deck. He must keep at least one of them, but he may keep two or all three if he chooses. Any returned cards are placed on the bottom of the deck.
TRAIN CAR CARDS
There are 8 types of regular Train Car cards, plus Locomotive cars. The colors of each type of Train Car card match various routes between cities on the board – Purple, Blue, Orange, White, Green, Yellow, Black, and Red.
Locomotives are Multi-colored and act as a wild card that can be part of any set of cards when claiming a route. If a Locomotive card is one of the five face-up cards, the player who draws it may only draw one card, instead of two. If, after having drawn one card the replacement card is a Locomotive, the player cannot take it. If at any time, three of the five face-up cards are Locomotives, all five cards are immediately discarded and five new ones are turned face-up to replace them.
Note: If a player is lucky enough to get a Locomotive from the top of the deck in a blind draw, it stills counts as a single card and he may still draw a total of two cards that turn.
A player may have any number of cards in his hand at any time.
When the deck is exhausted, the discards are reshuffled into a new draw pile deck. The cards should be shuffled thoroughly, since all the cards have been discarded in sets.

```python
from langchain.document_loaders.pdf import PyPDFDirectoryLoader

def load_documents():
    document_loader = PyPDFDirectoryLoader(DATA_PATH)
    return document_loader.load()
```

This is the code to load the documents from the data folder in our code



You can also load other types of documents by using the LangChain loaders here or integrations below



```
documents = load_documents()
print(documents[0])
```

```
page_content='If you roll a three-of-a-kind [...]'
metadata={'source': 'data/monopoly.pdf', 'page': 0}
```

SPLIT THE DOCUMENTS

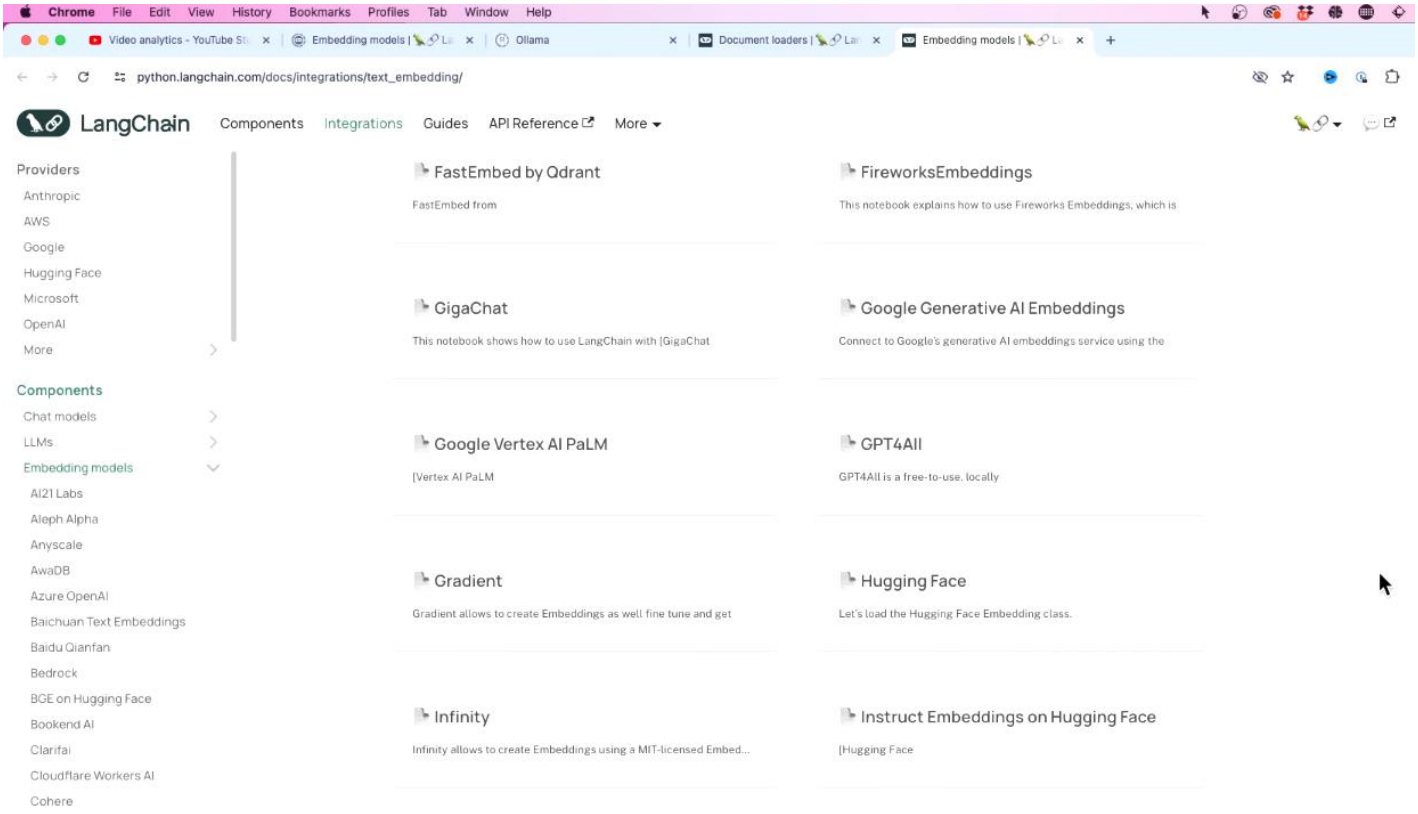
```
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain.schema.document import Document

def split_documents(documents: list[Document]):
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=800,
        chunk_overlap=80,
        length_function=len,
        is_separator_regex=False,
    )
    return text_splitter.split_documents(documents)
```

```
documents = load_documents()
chunks = split_documents(documents)
print(chunks[0])
```

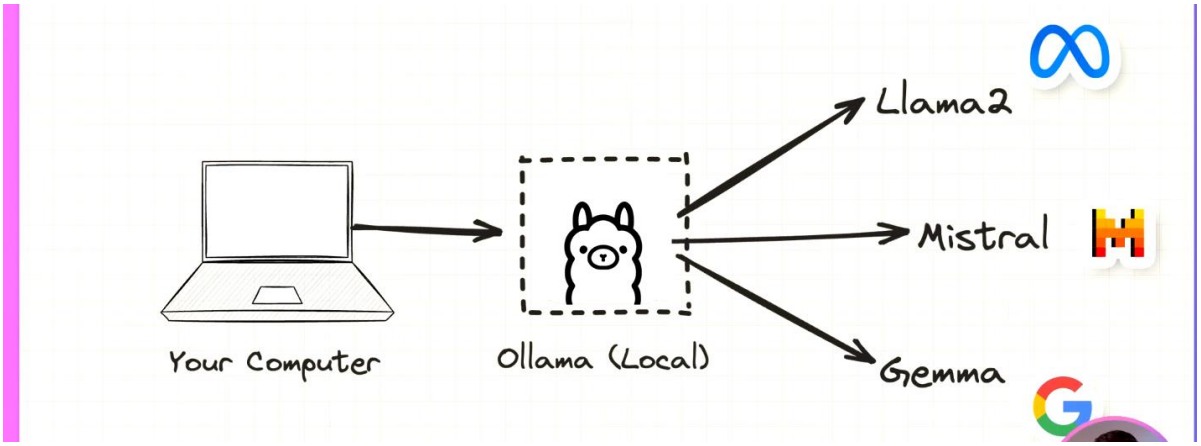
EMBEDDING FUNCTION

```
from langchain_community.embeddings.bedrock import BedrockEmbeddings


def get_embedding_function():
    embeddings = BedrockEmbeddings(
        credentials_profile_name="default", region_name="us-east-1"
    }
    return embeddings
```

Use an embedding function because we will need it in 2 separate places, when you create the database itself and when we want to query the database. LangChain comes with many different embedding functions like the **BedrockEmbeddings()** above

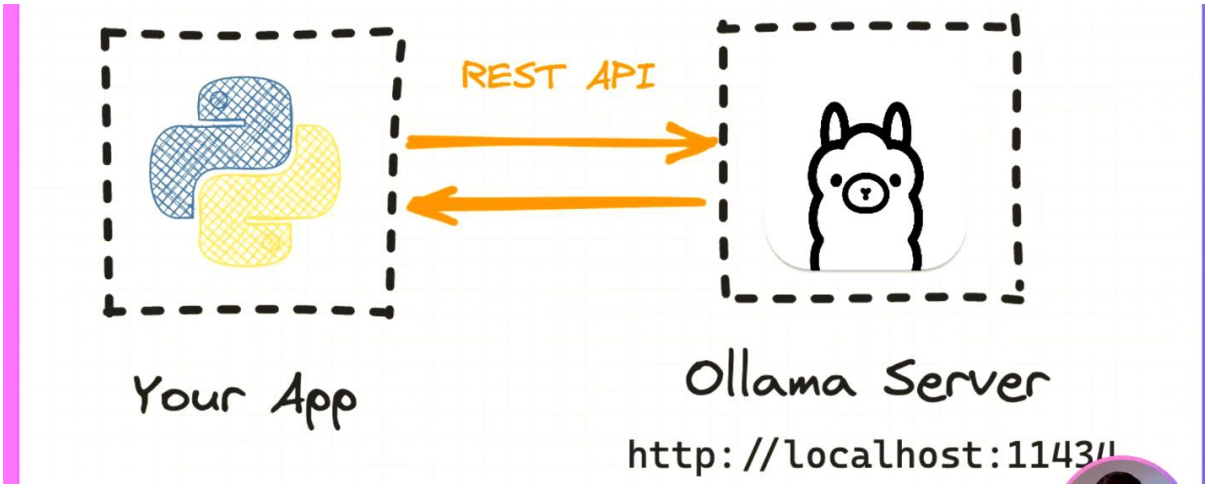



```python
from langchain_community.embeddings.ollama import OllamaEmbeddings

def get_embedding_function():
    embeddings = OllamaEmbeddings(model="nomic-embed-text")
    return embeddings
```




```
ollama pull llama2
ollama pull mistral
```

```
ollama serve
```



https://ollama.com/search?c=embedding&q=nomic

Cloud | Embedding | Vision | Tools | Thinking | Popular

nomic-embed-text
A high-performing open embedding model with a large token context window.
embedding
43.9M Pulls · 3 Tags · Updated 1 year ago

qwen3-embedding
Building upon the foundational models of the Qwen3 series, Qwen3 Embedding provides a comprehensive range of text embeddings models in various sizes
embedding · 0.6b · 4b · 8b
91.4K Pulls · 12 Tags · Updated 1 month ago

bge-large
Embedding model from BAAI mapping texts to vectors.
embedding · 335m
136.8K Pulls · 3 Tags · Updated 1 year ago

granite-embedding
The IBM Granite Embedding 30M and 278M models models are text-only dense biencoder embedding models, with 30M available in English only and 278M serving multilingual use cases.
embedding · 30m · 278m
107.5K Pulls · 6 Tags · Updated 10 months ago

jina/jina-embeddings-v2-base-de
Text embedding model (base) for English and German input of size up to 8192 tokens
embedding
172.7K Pulls · 1 Tag · Updated 1 year ago

CREATING THE DATABASE

```python
from get_embedding_function import get_embedding_function
from langchain.vectorstores.chroma import Chroma

def add_to_chroma(chunks: list[Document]):
    db = Chroma(
        persist_directory=CHROMA_PATH, embedding_function=get_embedding_function()
    )
    db.add_documents(new_chunks, ids=new_chunk_ids)
    db.persist()
```

Once we have the documents splitted into chunks, we can use the embedding function to build a vector database with it

```
page_content='If you roll a three-of-a-kind [...]'
metadata={'source': 'data/monopoly.pdf', 'page': 0}
```

To update the ChromaDB database, we need to tag every item in it with an id value

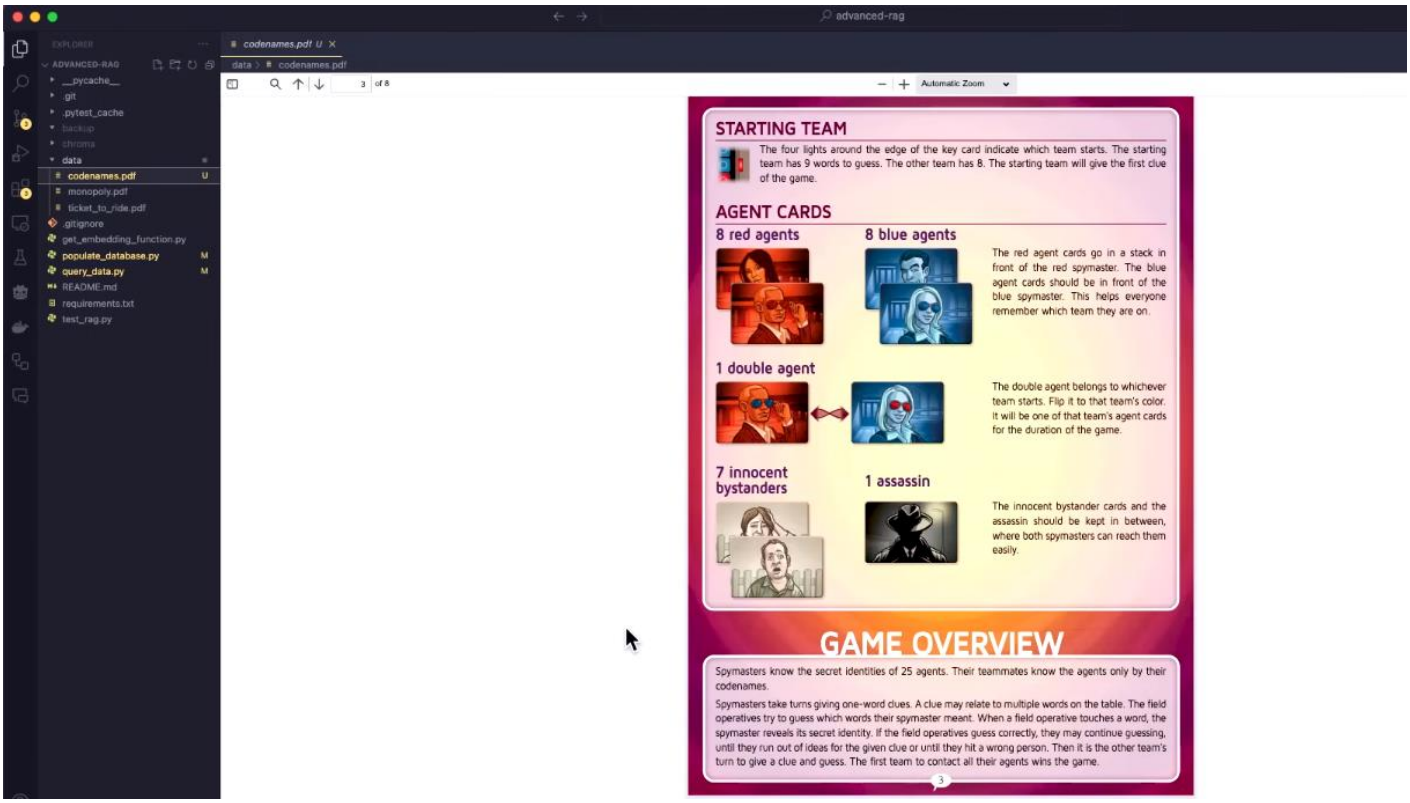
```
data/monopoly.pdf:6:2
```

```python
for chunk in chunks:
    source = chunk.metadata.get("source")
    page = chunk.metadata.get("page")
    current_page_id = f"{source}:{page}"
```

```python
last_page_id = None
current_chunk_index = 0

for chunk in chunks:
    [...]
    # If the page ID is the same as the last one, increment the index.
    if current_page_id == last_page_id:
        current_chunk_index += 1
    else:
        current_chunk_index = 0
```

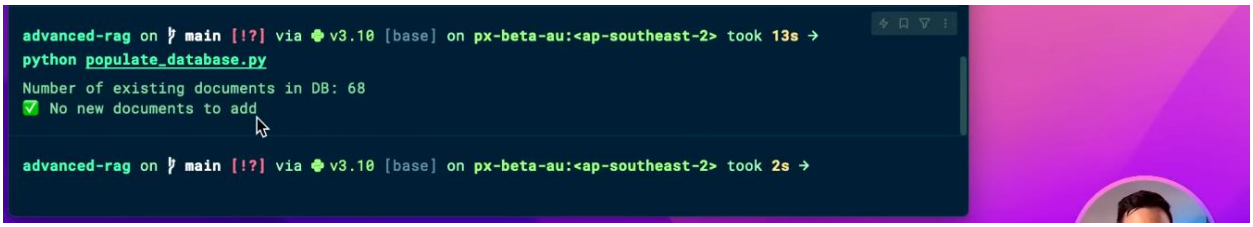
```
data/monopoly.pdf:0:0
data/monopoly.pdf:0:1
data/monopoly.pdf:0:2
data/monopoly.pdf:1:0
data/monopoly.pdf:1:1
```

```python
# Add it to the chunk meta-data.
chunk.metadata["id"] = chunk_id
```



We want the document chunking to only work for new documents added to the data folder, that is why we are using the **document:page:chunk** naming strategy above

We can see that it detected 41 already existing vector chunks inside the database, and we have 27 new document chunks to add from the newly added pdf file dropped in the **data** folder



When we run the same embedding function again against the **data** folder, we can see that it recognized that all the documents chunk already exist in the database and do not need to be added again

But if we modify a pdf file content, we need a way to detect that and replace all the document chunks in the database

```python
from langchain.vectorstores.chroma import Chroma

db = Chroma(
    persist_directory=CHROMA_PATH,
    embedding_function=get_embedding_function()
)
```

Now that all chunks have a unique id, we can add them to the vector database

```python
# Add or Update the documents.
existing_items = db.get(include=[])  # IDs are always included by default
existing_ids = set(existing_items["ids"])
print(f"Number of existing documents in DB: {len(existing_ids)}")
```

```python
# Only add documents that don't exist in the DB.
new_chunks = []
for chunk in chunks_with_ids:
    if chunk.metadata["id"] not in existing_ids:
        new_chunks.append(chunk)
```

```python
new_chunk_ids = [chunk.metadata["id"] for chunk in new_chunks]
db.add_documents(new_chunks, ids=new_chunk_ids)
db.persist()
```

RUNNING RAG LOCALLY

```python
from langchain_community.embeddings.ollama import OllamaEmbeddings

def get_embedding_function():
    embeddings = OllamaEmbeddings(model="nomic-embed-text")
    return embeddings
```

```python
from langchain_community.embeddings.bedrock import BedrockEmbeddings


def get_embedding_function():
    embeddings = BedrockEmbeddings(
        credentials_profile_name="default", region_name="us-east-1"
    }
    return embeddings
```

Local LLMs and local embedding models are good if we can validate them

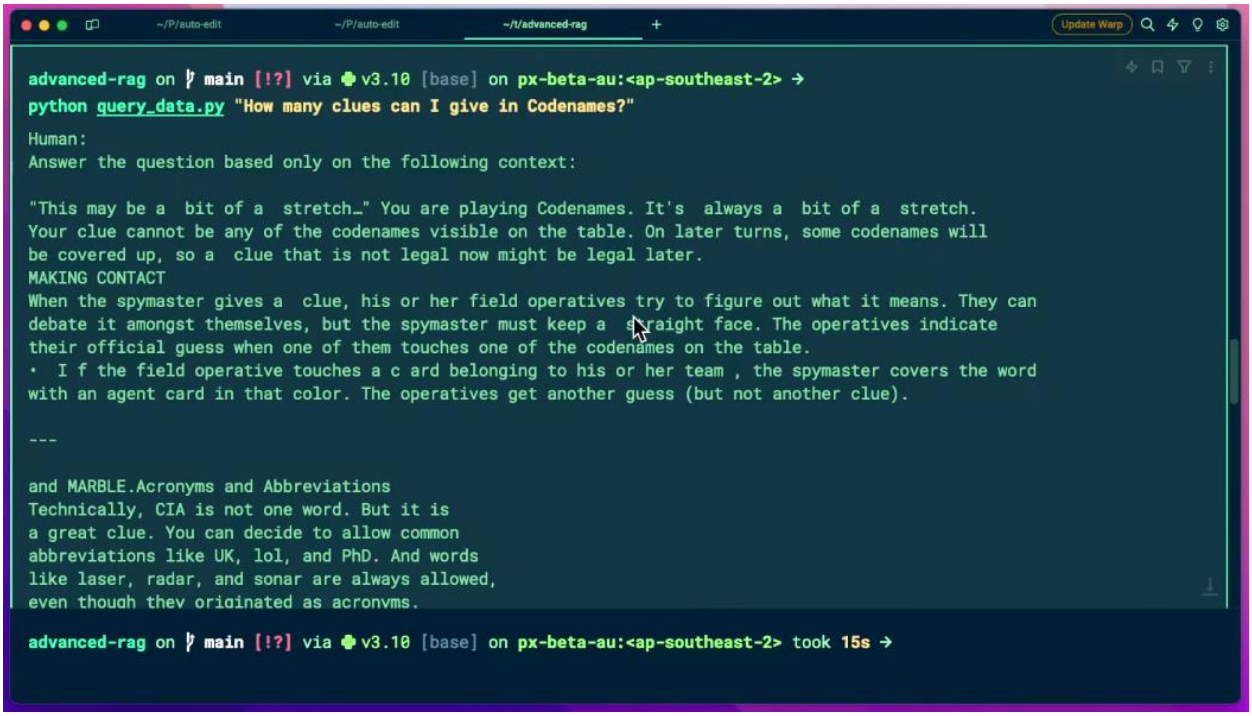
```python
def query_rag(query_text: str):
    ...
```

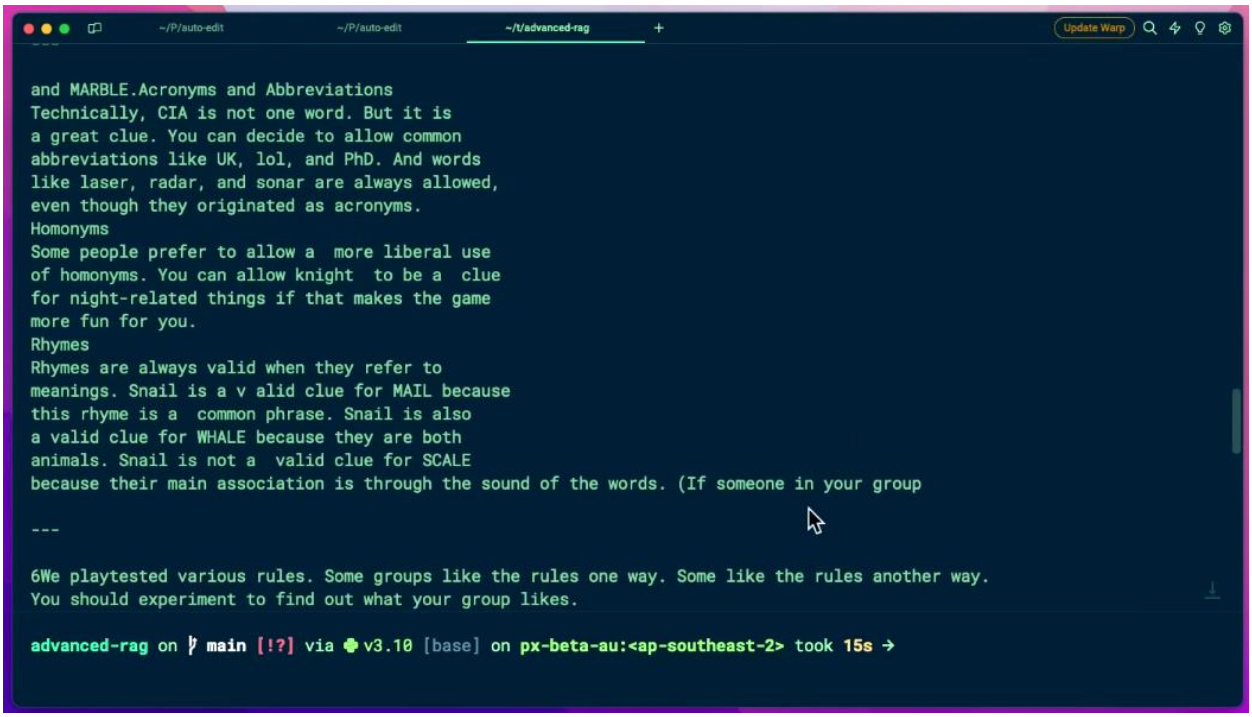
```python
embedding_function = get_embedding_function()
db = Chroma(
    persist_directory=CHROMA_PATH,
    embedding_function=embedding_function
)
```

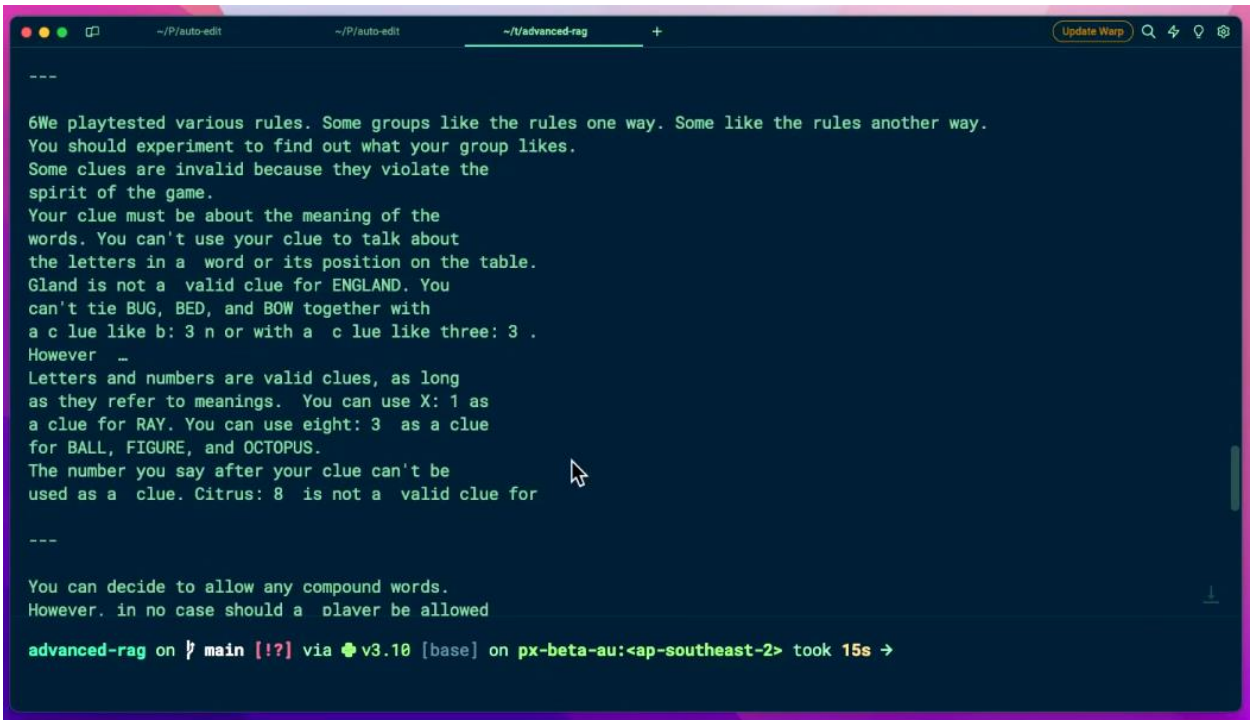
```python
PROMPT_TEMPLATE = """
Answer the question based only on the following context:
{context}

---
Answer the question based on the above context: {question}
"""
```

```python
results = db.similarity_search_with_score(query_text, k=5)
```

```python
context_text = "\n\n---\n\n".join([doc.page_content for doc, _score in results])
prompt_template = ChatPromptTemplate.from_template(PROMPT_TEMPLATE)
prompt = prompt_template.format(context=context_text, question=query_text)
```

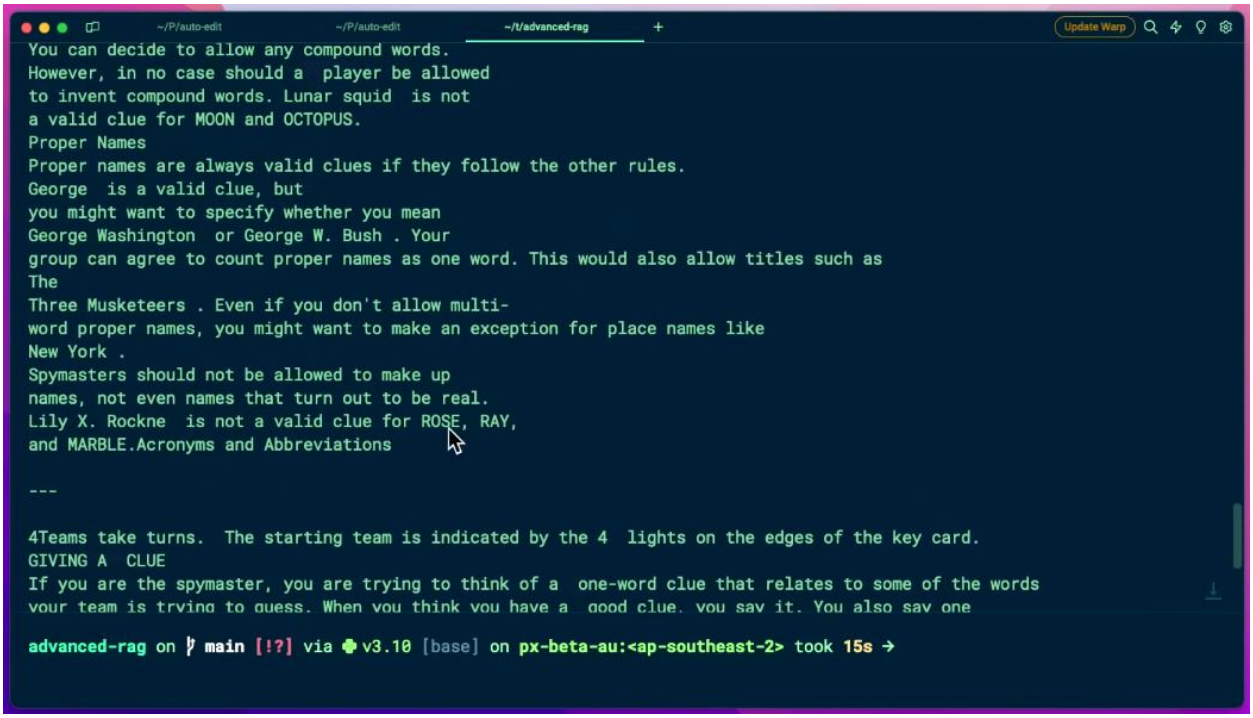

~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> →
python query_data.py "How many clues can I give in Codenames?"
Human:
Answer the question based only on the following context:
"This may be a bit of a stretch…" You are playing Codenames. It's always a bit of a stretch.
Your clue cannot be any of the codenames visible on the table. On later turns, some codenames will
be covered up, so a clue that is not legal now might be legal later.
MAKING CONTACT
When the spymaster gives a clue, his or her field operatives try to figure out what it means. They can
debate it amongst themselves, but the spymaster must keep a straight face. The operatives indicate
their official guess when one of them touches one of the codenames on the table.
• If the field operative touches a card belonging to his or her team, the spymaster covers the word
with an agent card in that color. The operatives get another guess (but not another clue).
---
and MARBLE.Acronyms and Abbreviations
Technically, CIA is not one word. But it is
a great clue. You can decide to allow common
abbreviations like UK, lol, and PhD. And words
like laser, radar, and sonar are always allowed,
even though they originated as acronyms.
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 15s →


~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
and MARBLE.Acronyms and Abbreviations
Technically, CIA is not one word. But it is
a great clue. You can decide to allow common
abbreviations like UK, lol, and PhD. And words
like laser, radar, and sonar are always allowed,
even though they originated as acronyms.
Homonyms
Some people prefer to allow a more liberal use
of homonyms. You can allow knight to be a clue
for night-related things if that makes the game
more fun for you.
Rhymes
Rhymes are always valid when they refer to
meanings. Snail is a valid clue for MAIL because
this rhyme is a common phrase. Snail is also
a valid clue for WHALE because they are both
animals. Snail is not a valid clue for SCALE
because their main association is through the sound of the words. (If someone in your group
---
6We playtested various rules. Some groups like the rules one way. Some like the rules another way.
You should experiment to find out what your group likes.
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 15s →

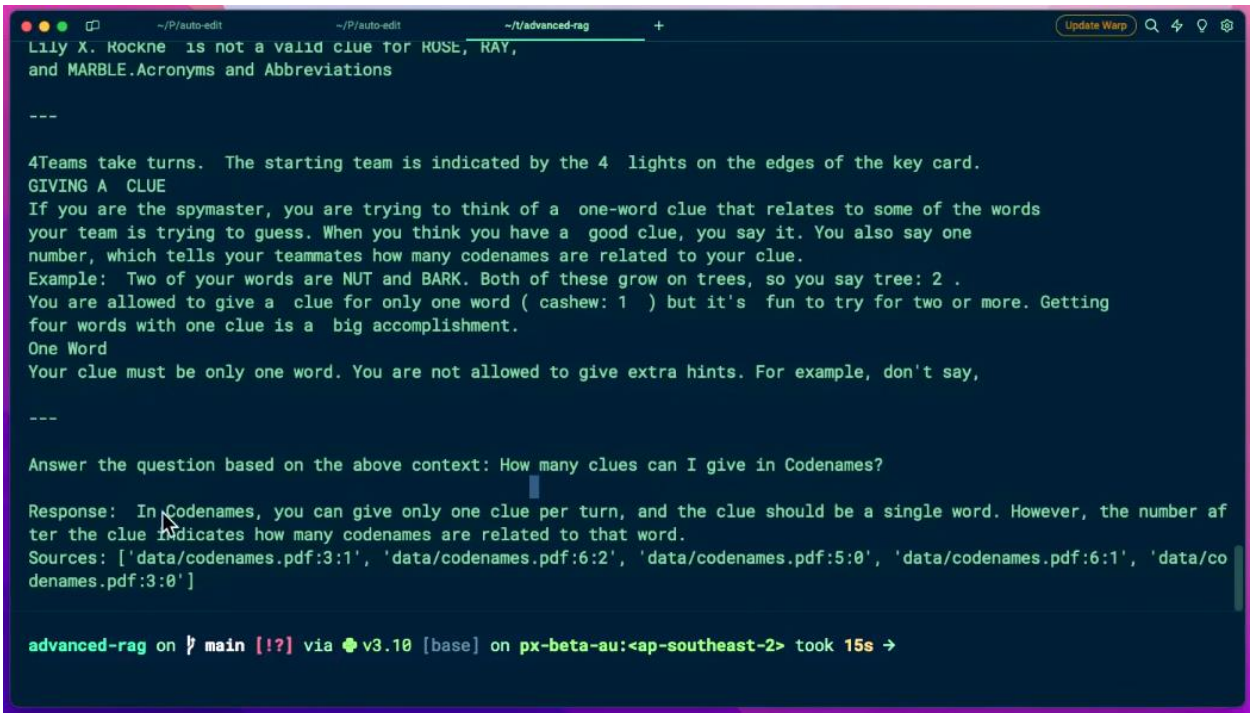

~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp

6We playtested various rules. Some groups like the rules one way. Some like the rules another way.
You should experiment to find out what your group likes.
Some clues are invalid because they violate the
spirit of the game.
Your clue must be about the meaning of the
words. You can't use your clue to talk about
the letters in a word or its position on the table.
Gland is not a valid clue for ENGLAND. You
can't tie BUG, BED, and BOW together with
a c lue like b: 3 n or with a c lue like three: 3 .
However …
Letters and numbers are valid clues, as long
as they refer to meanings. You can use X: 1 as
a clue for RAY. You can use eight: 3 as a clue
for BALL, FIGURE, and OCTOPUS.
The number you say after your clue can't be
used as a clue. Citrus: 8 is not a valid clue for

You can decide to allow any compound words.
However, in no case should a player be allowed
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 15s →


~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
You can decide to allow any compound words.
However, in no case should a player be allowed
to invent compound words. Lunar squid is not
a valid clue for MOON and OCTOPUS.
Proper Names
Proper names are always valid clues if they follow the other rules.
George is a valid clue, but
you might want to specify whether you mean
George Washington or George W. Bush . Your
group can agree to count proper names as one word. This would also allow titles such as
The
Three Musketeers . Even if you don't allow multi-
word proper names, you might want to make an exception for place names like
New York .
Spymasters should not be allowed to make up
names, not even names that turn out to be real.
Lily X. Rockne is not a valid clue for ROSE, RAY,
and MARBLE.Acronyms and Abbreviations

4Teams take turns. The starting team is indicated by the 4 lights on the edges of the key card.
GIVING A CLUE
If you are the spymaster, you are trying to think of a one-word clue that relates to some of the words
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 15s →

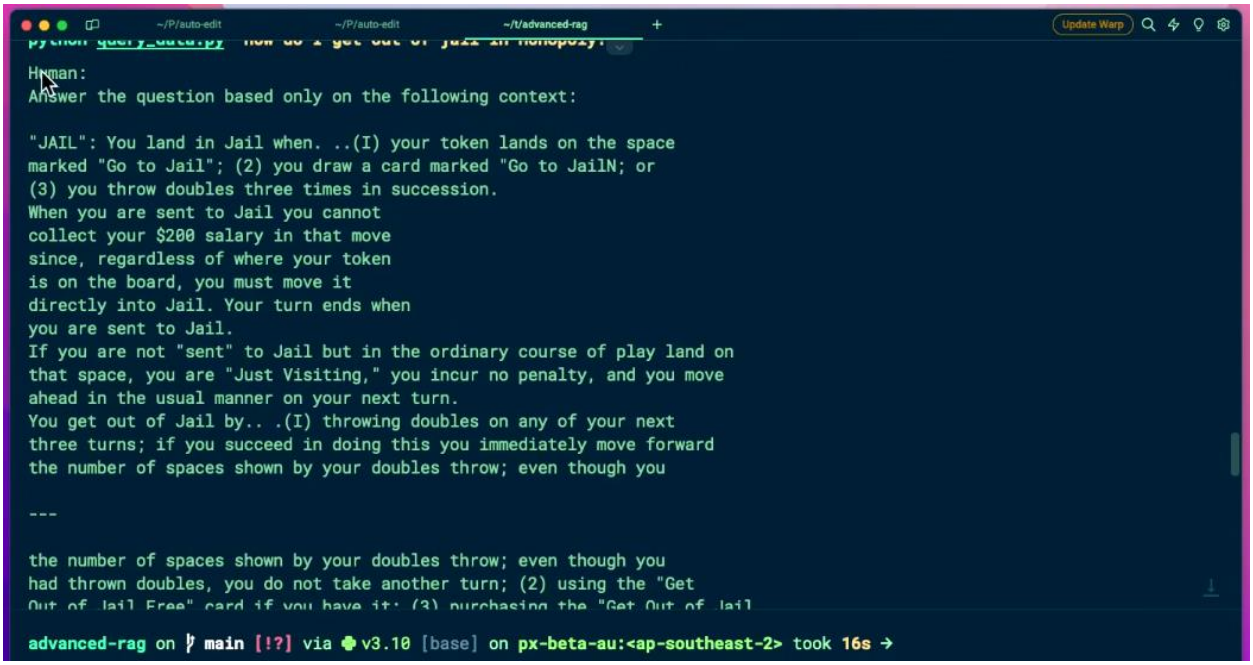

~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
Lily X. Rockne is not a valid clue for ROSE, RAY,
and MARBLE.Acronyms and Abbreviations

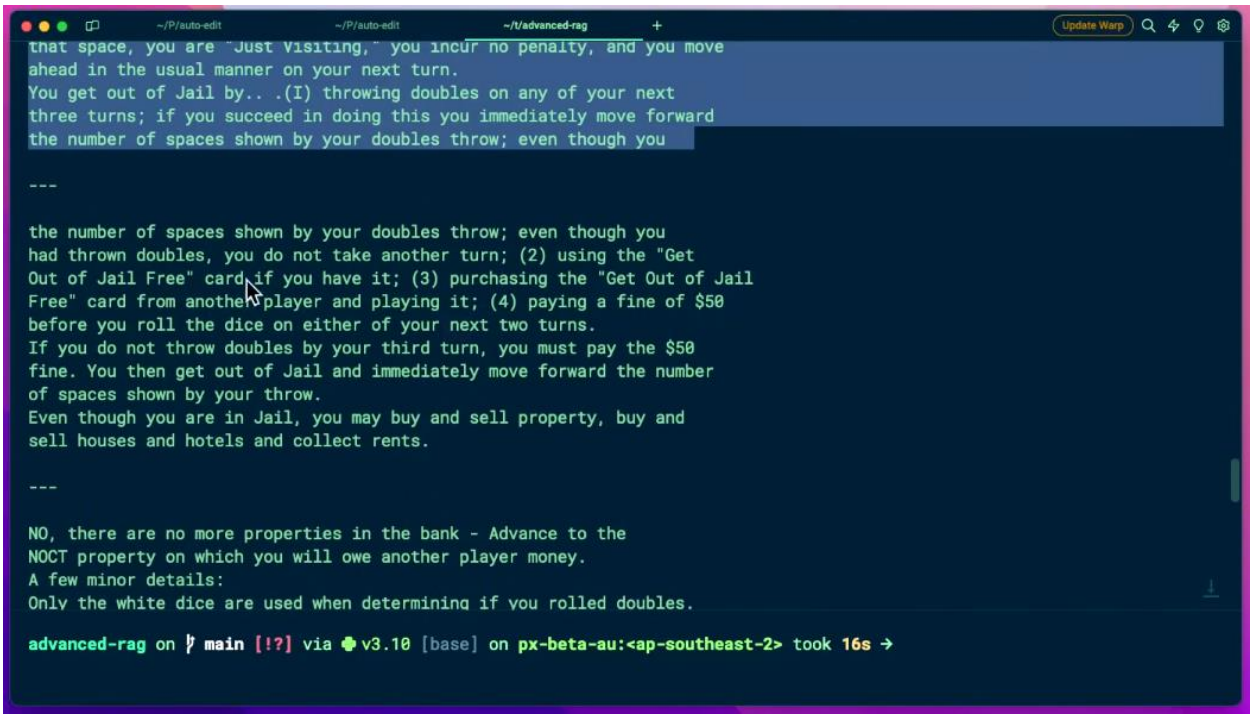
4Teams take turns. The starting team is indicated by the 4 lights on the edges of the key card.
GIVING A CLUE
If you are the spymaster, you are trying to think of a one-word clue that relates to some of the words
your team is trying to guess. When you think you have a good clue, you say it. You also say one
number, which tells your teammates how many codenames are related to your clue.
Example: Two of your words are NUT and BARK. Both of these grow on trees, so you say tree: 2 .
You are allowed to give a clue for only one word (cashew: 1) but it's fun to try for two or more. Getting
four words with one clue is a big accomplishment.
One Word
Your clue must be only one word. You are not allowed to give extra hints. For example, don't say,
---
Answer the question based on the above context: How many clues can I give in Codenames?
Response: In Codenames, you can give only one clue per turn, and the clue should be a single word. However, the number af
ter the clue indicates how many codenames are related to that word.
Sources: ['data/codenames.pdf:3:1', 'data/codenames.pdf:6:2', 'data/codenames.pdf:5:0', 'data/codenames.pdf:6:1', 'data/co
denames.pdf:3:0']
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 15s


from langchain_community.llms.ollama import Ollama
model = Ollama(model="mistral")
response_text = model.invoke(prompt)
print(response_text)

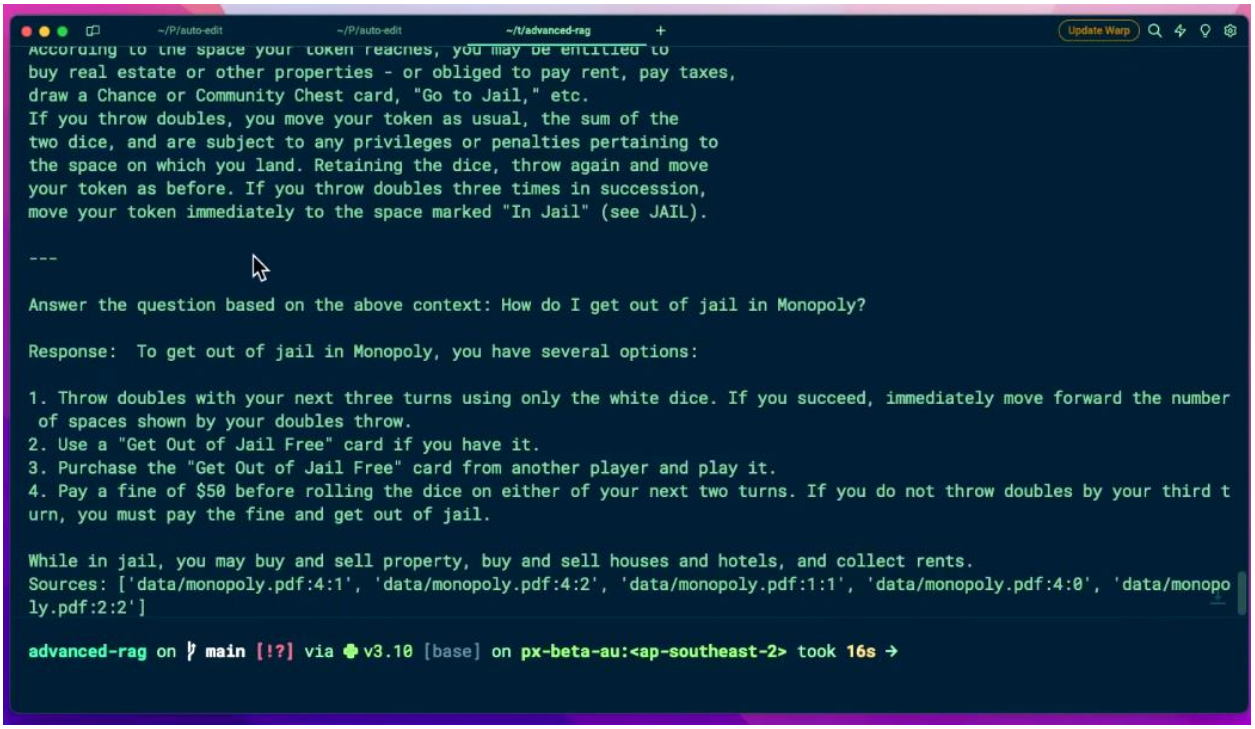

sources = [doc.metadata.get("id", None) for doc, _score in results]


advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> →
python query_data.py "How do I get out of jail in Monopoly?"


~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
Human:
Answer the question based only on the following context:
"JAIL": You land in Jail when. ..(I) your token lands on the space
marked "Go to Jail"; (2) you draw a card marked "Go to JailN; or
(3) you throw doubles three times in succession.
When you are sent to Jail you cannot
collect your $200 salary in that move
since, regardless of where your token
is on the board, you must move it
directly into Jail. Your turn ends when
you are sent to Jail.
If you are not "sent" to Jail but in the ordinary course of play land on
that space, you are "Just Visiting," you incur no penalty, and you move
ahead in the usual manner on your next turn.
You get out of Jail by.. .(I) throwing doubles on any of your next
three turns; if you succeed in doing this you immediately move forward
the number of spaces shown by your doubles throw; even though you
---
the number of spaces shown by your doubles throw; even though you
had thrown doubles, you do not take another turn; (2) using the "Get
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 16s

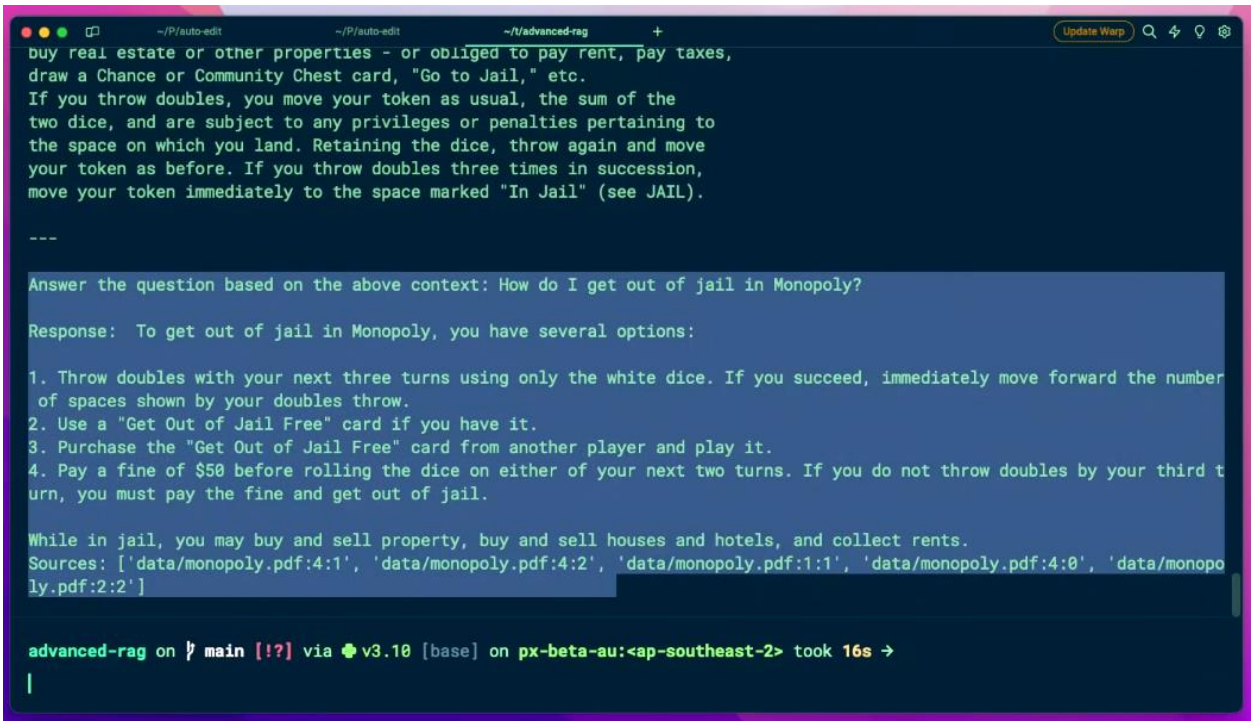

~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
that space, you are "Just Visiting," you incur no penalty, and you move
ahead in the usual manner on your next turn.
You get out of Jail by.. .(I) throwing doubles on any of your next
three turns; if you succeed in doing this you immediately move forward
the number of spaces shown by your doubles throw; even though you
---
the number of spaces shown by your doubles throw; even though you
had thrown doubles, you do not take another turn; (2) using the "Get
Out of Jail Free" card if you have it; (3) purchasing the "Get Out of Jail
Free" card from another player and playing it; (4) paying a fine of $50
before you roll the dice on either of your next two turns.
If you do not throw doubles by your third turn, you must pay the $50
fine. You then get out of Jail and immediately move forward the number
of spaces shown by your throw.
Even though you are in Jail, you may buy and sell property, buy and
sell houses and hotels and collect rents.

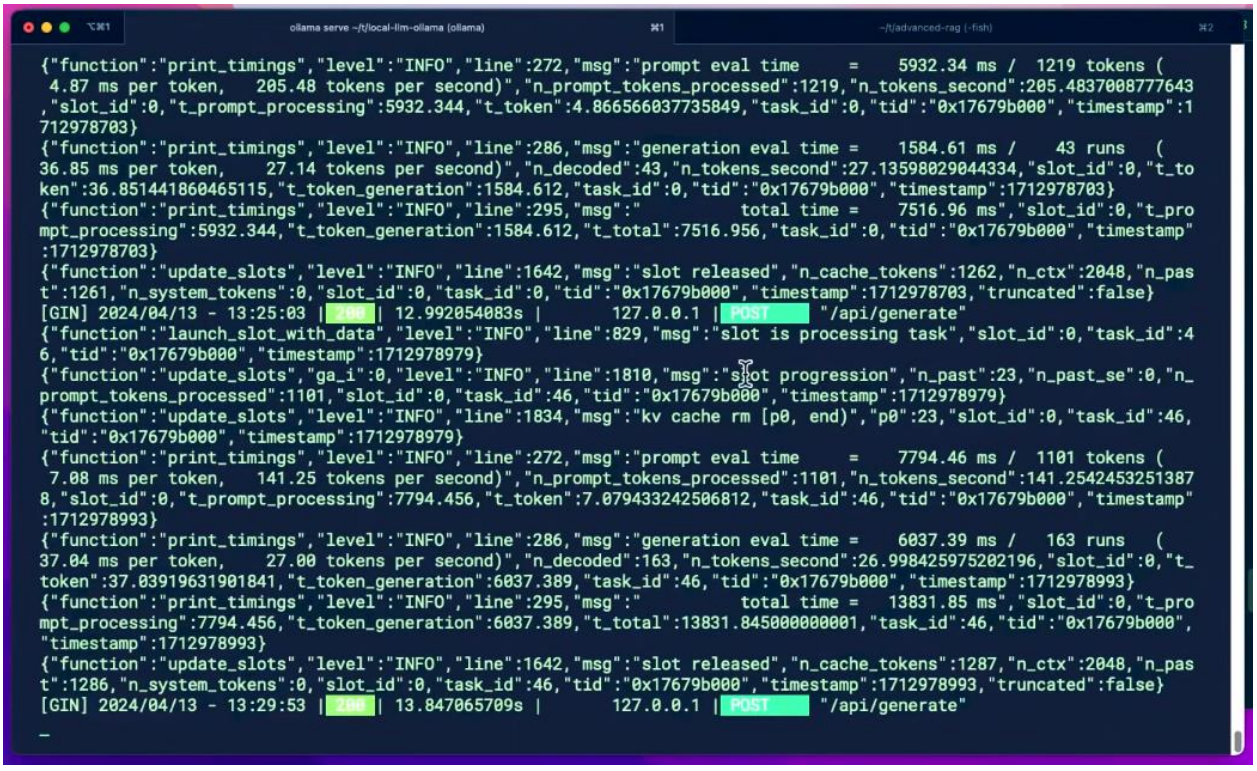
NO, there are no more properties in the bank - Advance to the
NOCT property on which you will owe another player money.
A few minor details:
Only the white dice are used when determining if you rolled doubles.
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 16s


~/P/auto-edit
~/P/auto-edit
~/t/advanced-rag
Update Warp
buy real estate or other properties - or obliged to pay rent, pay taxes,
draw a Chance or Community Chest card, "Go to Jail," etc.
If you throw doubles, you move your token as usual, the sum of the
two dice, and are subject to any privileges or penalties pertaining to
the space on which you land. Retaining the dice, throw again and move
your token as before. If you throw doubles three times in succession,
move your token immediately to the space marked "In Jail" (see JAIL).

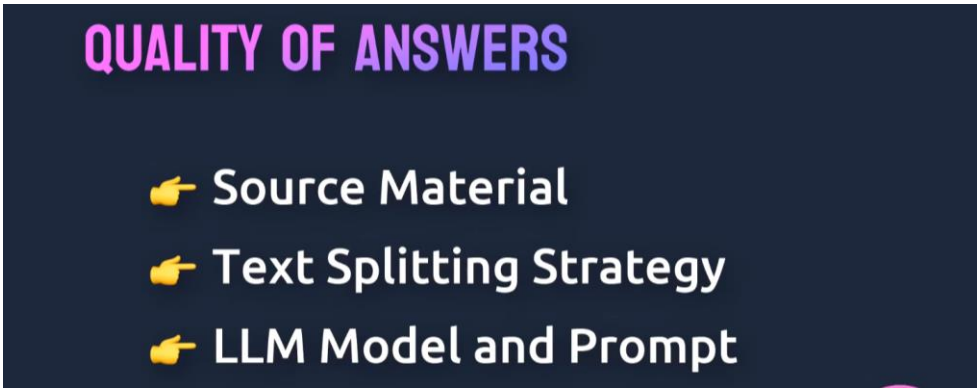
Answer the question based on the above context: How do I get out of jail in Monopoly?
Response: To get out of jail in Monopoly, you have several options:
1. Throw doubles with your next three turns using only the white dice. If you succeed, immediately move forward the number
of spaces shown by your doubles throw.
2. Use a "Get Out of Jail Free" card if you have it.
3. Purchase the "Get Out of Jail Free" card from another player and play it.
4. Pay a fine of $50 before rolling the dice on either of your next two turns. If you do not throw doubles by your third t
urn, you must pay the fine and get out of jail.
While in jail, you may buy and sell property, buy and sell houses and hotels, and collect rents.
Sources: ['data/monopoly.pdf:4:1', 'data/monopoly.pdf:4:2', 'data/monopoly.pdf:1:1', 'data/monopoly.pdf:4:0', 'data/monopo
ly.pdf:2:2']
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 16s

It worked!





TESTING RAG OUTPUT




```
Question: How much total money does a player start with in Monopoly?
Expected Answer: $1500
```

```
acutal_answer = query_rag(question)
assert actual_answer == expected_answer
```

```
Error:
Expected "$1500" but got "1,500 Dollars".
```

The problem here is that we can't do a strict output string comparison because there are many possible LLM responses

```
EVAL_PROMPT = """
Expected Response: {expected_response}
Actual Response: {actual_response}
---
(Answer with 'true' or 'false') Does the actual response match the expected response?
"""
```

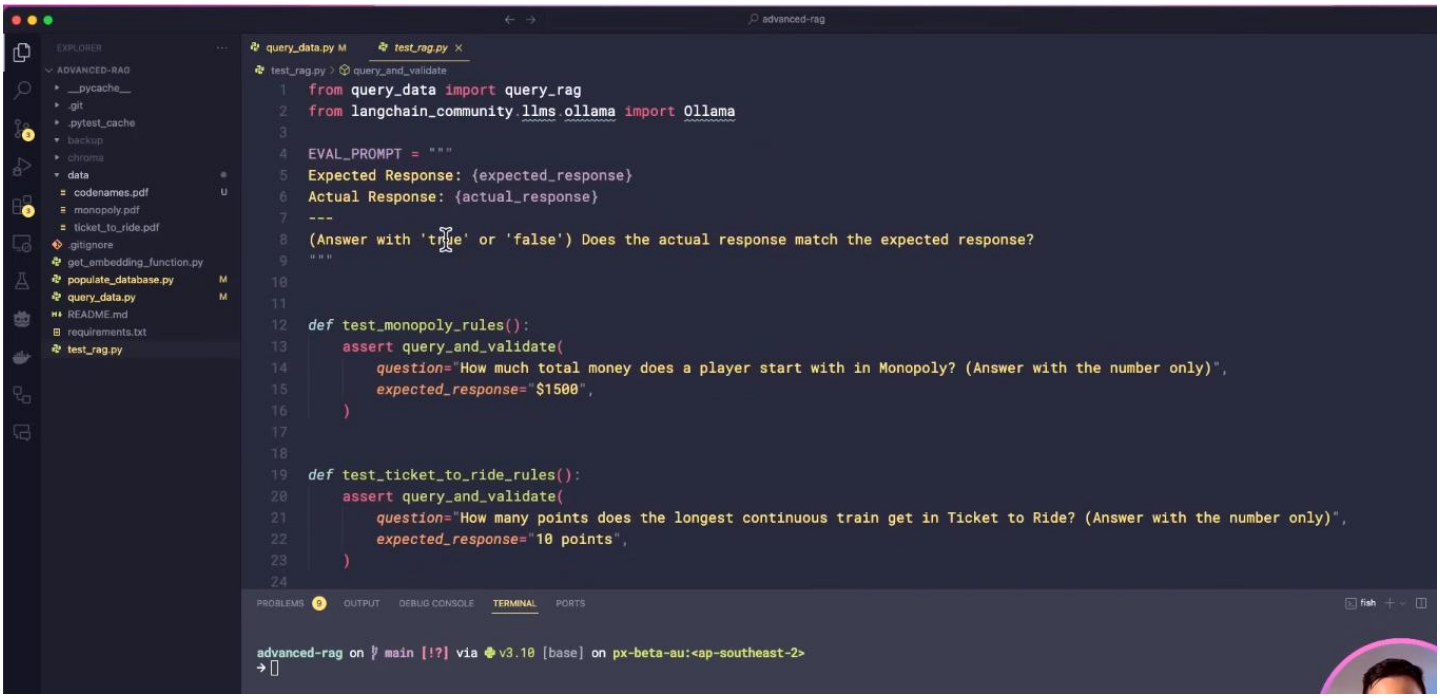
```
response_text = query_rag(question)
prompt = EVAL_PROMPT.format(
    expected_response=expected_response,
    actual_response=response_text
)

model = Ollama(model="mistral")
evaluation_results_str = model.invoke(prompt)
```

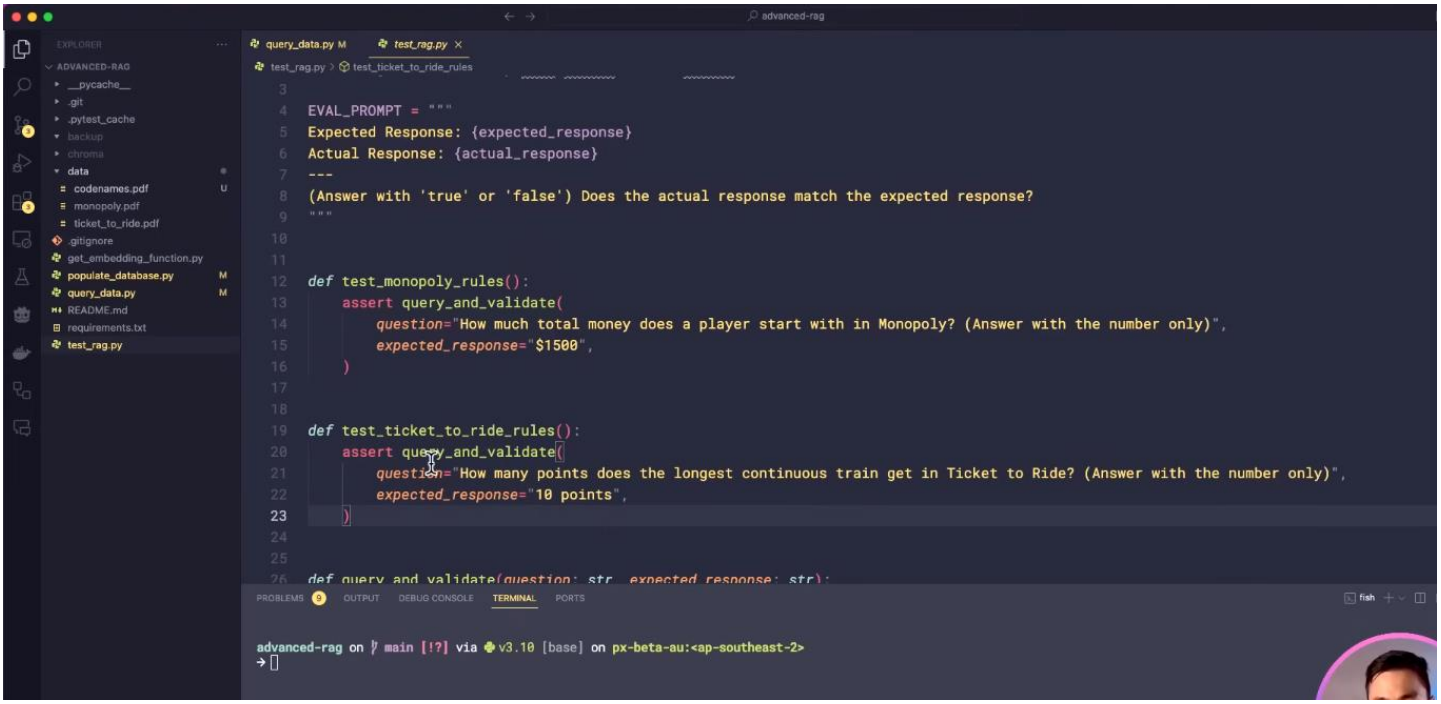
```python
final_result = evaluation_results_str.strip().lower()
if "true" in final_result:
    return True
elif "false" in final_result:
    return False
else:
    raise ValueError("Cannot determine if true or false")
```

```python
def query_and_validate(question: str, expected_response: str):
    ...
```

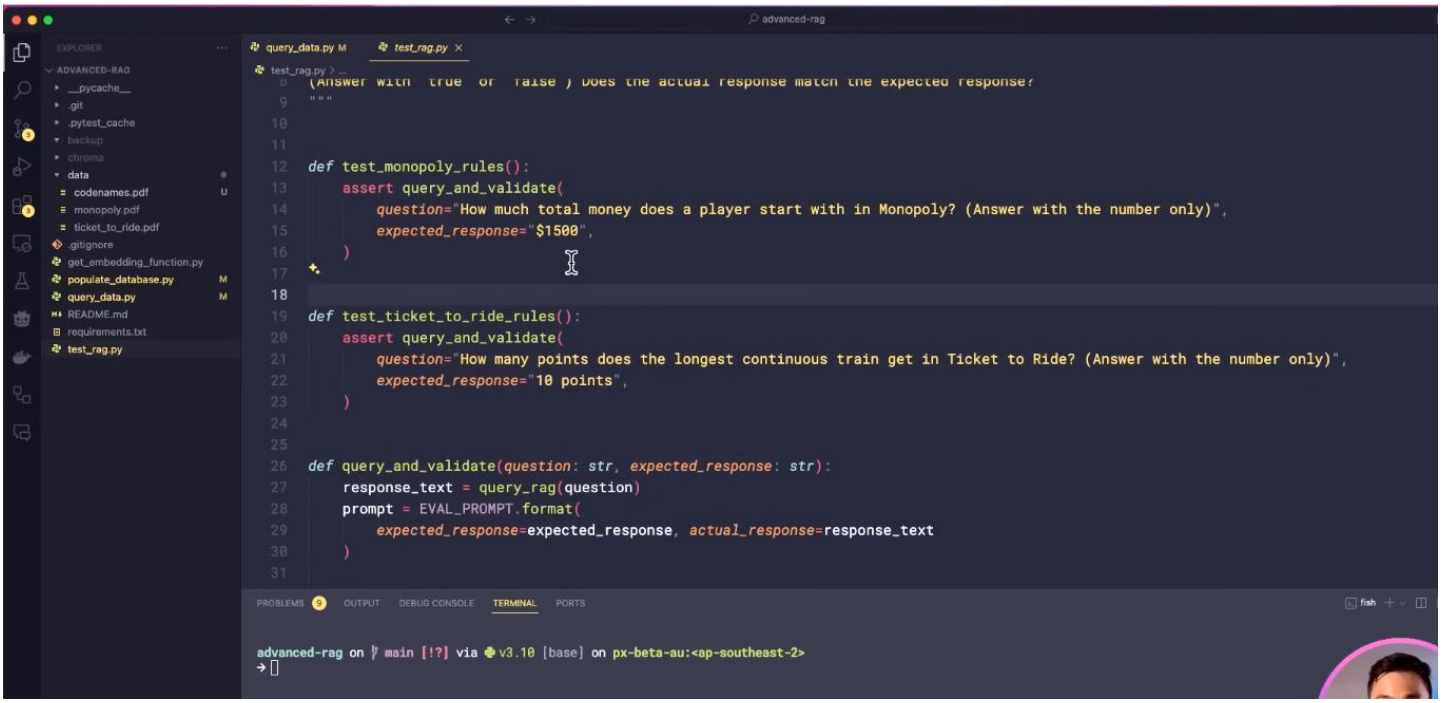
We can wrap this into a helper function that will return true or false as above. Then we can write unit tests as below

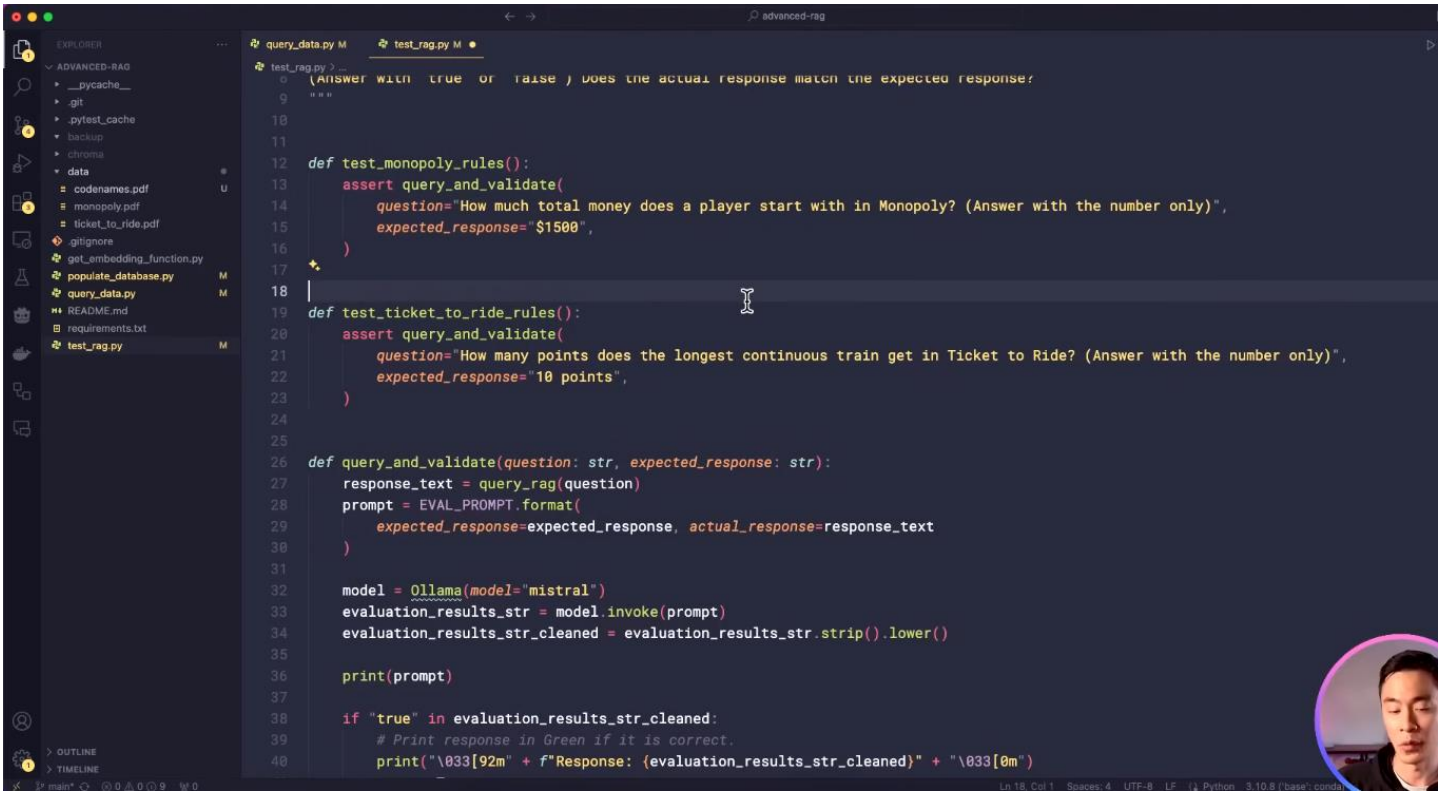
```python
def test_monopoly_rules():
    assert query_and_validate(
        question="How much total money does a player start with in Monopoly?",
        expected_response="$1500",
    )
```

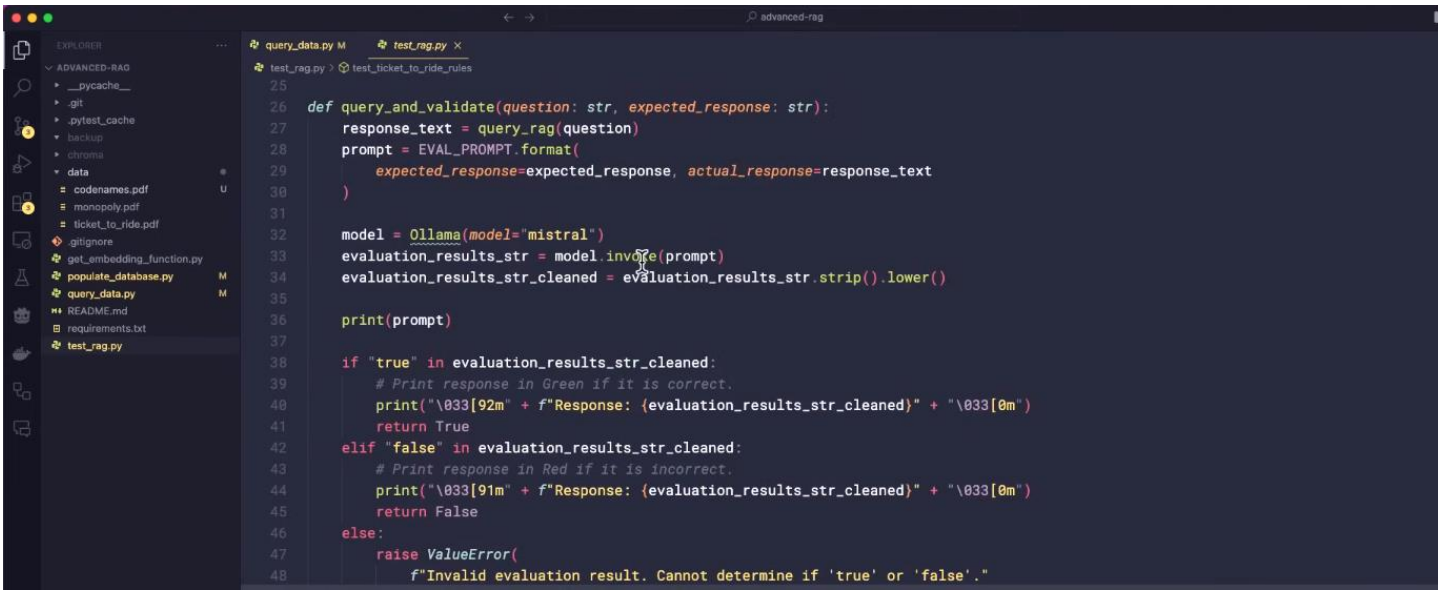



advanced-rag
query_data.py M
test_rag.py
test_rag.py > test_ticket_to_ride_rules
EXPLORER
ADVANCED-RAG
__pycache__
.git
.pytest_cache
backup
chroma
data
codenames.pdf
monopoly.pdf
ticket_to_ride.pdf
.gitignore
get_embedding_function.py
populate_database.py
query_data.py
README.md
requirements.txt
test_rag.py
EVAL_PROMPT = """
Expected Response: {expected_response}
Actual Response: {actual_response}

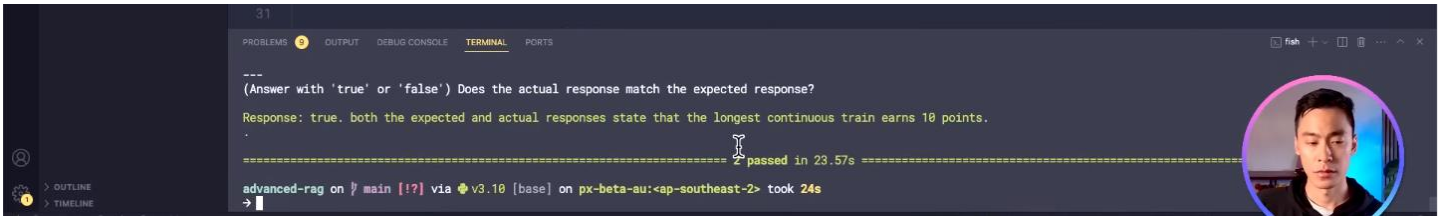
(Answer with 'true' or 'false') Does the actual response match the expected response?
"""
def test_monopoly_rules():
assert query_and_validate(
question="How much total money does a player start with in Monopoly? (Answer with the number only)",
expected_response="$1500",
)
def test_ticket_to_ride_rules():
assert query_and_validate(
question="How many points does the longest continuous train get in Ticket to Ride? (Answer with the number only)",
expected_response="10 points",
)
def query_and_validate(question: str, expected_response: str):
PROBLEMS 9
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
fish
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2>


advanced-rag
query_data.py M
test_rag.py
test_rag.py
(Answer with 'true' or 'false') Does the actual response match the expected response?
"""
def test_monopoly_rules():
assert query_and_validate(
question="How much total money does a player start with in Monopoly? (Answer with the number only)",
expected_response="$1500",
)
def test_ticket_to_ride_rules():
assert query_and_validate(
question="How many points does the longest continuous train get in Ticket to Ride? (Answer with the number only)",
expected_response="10 points",
)
def query_and_validate(question: str, expected_response: str):
response_text = query_rag(question)
prompt = EVAL_PROMPT.format(
expected_response=expected_response, actual_response=response_text
)
PROBLEMS 9
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
fish
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2>

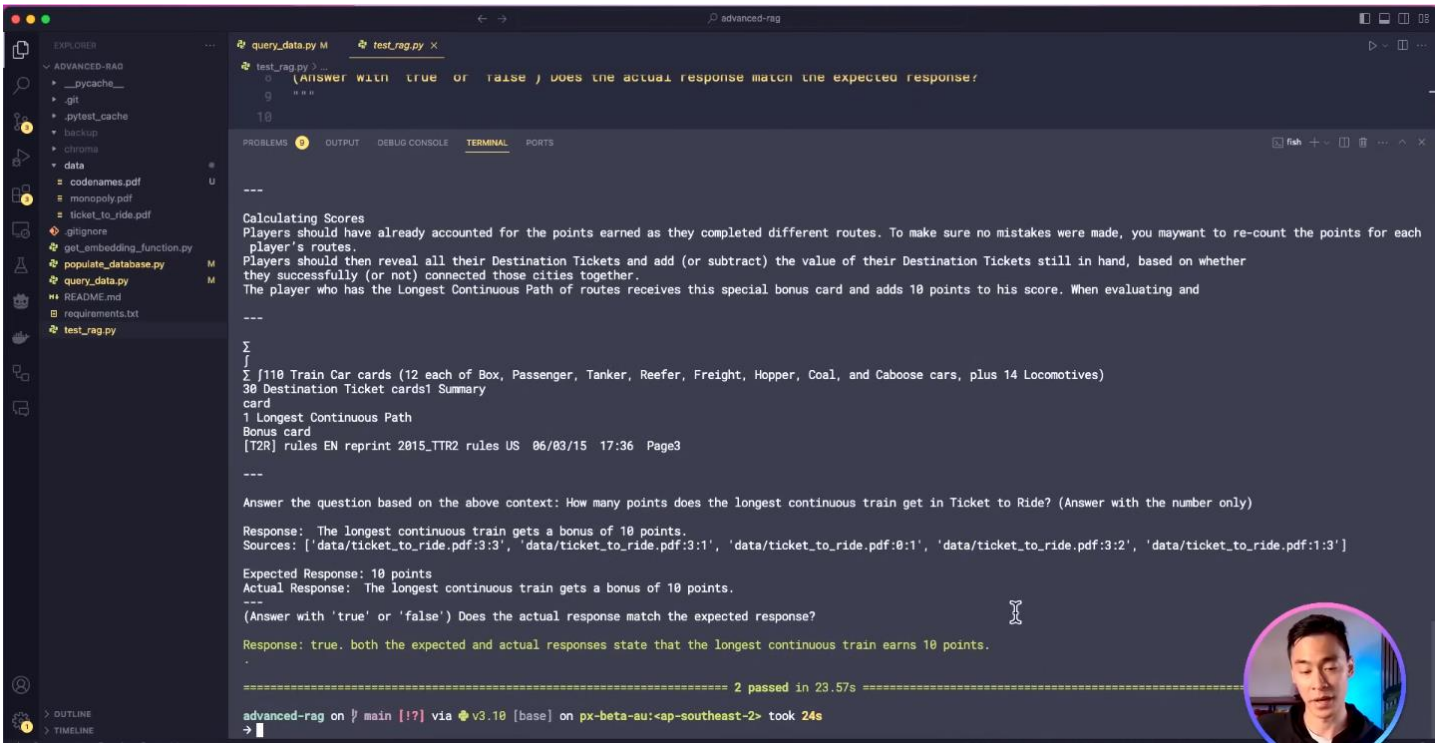

query_data.py M
test_rag.py M
(Answer with "true" or "false") Does the actual response match the expected response?
"""
def test_monopoly_rules():
assert query_and_validate(
question="How much total money does a player start with in Monopoly? (Answer with the number only)",
expected_response="$1500",
)
def test_ticket_to_ride_rules():
assert query_and_validate(
question="How many points does the longest continuous train get in Ticket to Ride? (Answer with the number only)",
expected_response="10 points",
)
def query_and_validate(question: str, expected_response: str):
response_text = query_rag(question)
prompt = EVAL_PROMPT.format(
expected_response=expected_response, actual_response=response_text
)
model = Ollama(model="mistral")
evaluation_results_str = model.invoke(prompt)
evaluation_results_str_cleaned = evaluation_results_str.strip().lower()
print(prompt)
if "true" in evaluation_results_str_cleaned:
# Print response in Green if it is correct.
print("\033[92m" + f"Response: {evaluation_results_str_cleaned}" + "\033[0m")


query_data.py M
test_rag.py
def query_and_validate(question: str, expected_response: str):
response_text = query_rag(question)
prompt = EVAL_PROMPT.format(
expected_response=expected_response, actual_response=response_text
)
model = Ollama(model="mistral")
evaluation_results_str = model.invoke(prompt)
evaluation_results_str_cleaned = evaluation_results_str.strip().lower()
print(prompt)
if "true" in evaluation_results_str_cleaned:
# Print response in Green if it is correct.
print("\033[92m" + f"Response: {evaluation_results_str_cleaned}" + "\033[0m")
return True
elif "false" in evaluation_results_str_cleaned:
# Print response in Red if it is incorrect.
print("\033[91m" + f"Response: {evaluation_results_str_cleaned}" + "\033[0m")
return False
else:
raise ValueError(
f"Invalid evaluation result. Cannot determine if 'true' or 'false'."


expected_response=expected_response, actual_response=response_text
PROBLEMS 9
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2>
pytest -s


PROBLEMS 9
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

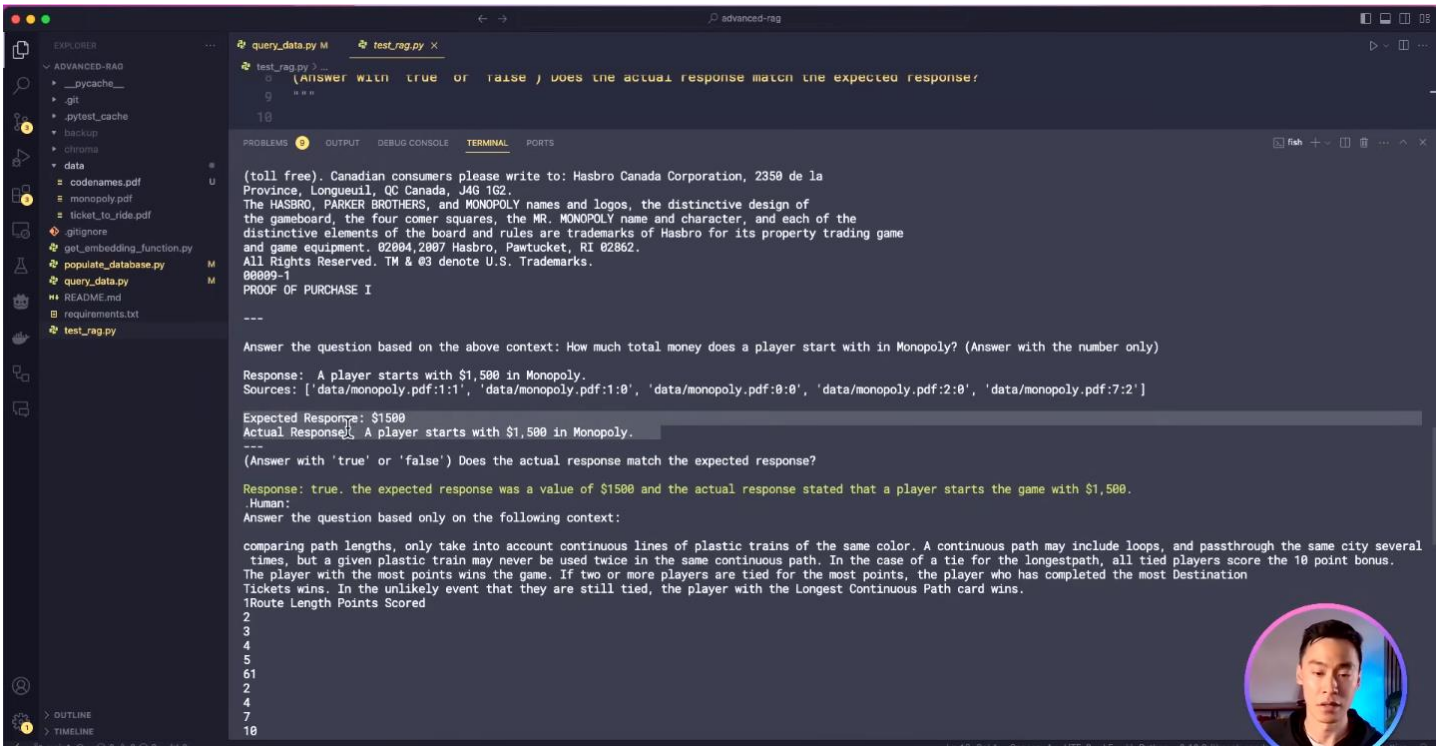
(Answer with 'true' or 'false') Does the actual response match the expected response?
Response: true. both the expected and actual responses state that the longest continuous train earns 10 points.
.
2 passed in 23.57s
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 24s


advanced-rag
query_data.py M
test_rag.py
(Answer with 'true' or 'false') Does the actual response match the expected response?
"""
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Calculating Scores
Players should have already accounted for the points earned as they completed different routes. To make sure no mistakes were made, you maywant to re-count the points for each player's routes.
Players should then reveal all their Destination Tickets and add (or subtract) the value of their Destination Tickets still in hand, based on whether they successfully (or not) connected those cities together.
The player who has the Longest Continuous Path of routes receives this special bonus card and adds 10 points to his score. When evaluating and
---
∑
∫
∑ ∫110 Train Car cards (12 each of Box, Passenger, Tanker, Reefer, Freight, Hopper, Coal, and Caboose cars, plus 14 Locomotives)
30 Destination Ticket cards1 Summary
card
1 Longest Continuous Path
Bonus card
[T2R] rules EN reprint 2015_TTR2 rules US 06/03/15 17:36 Page3

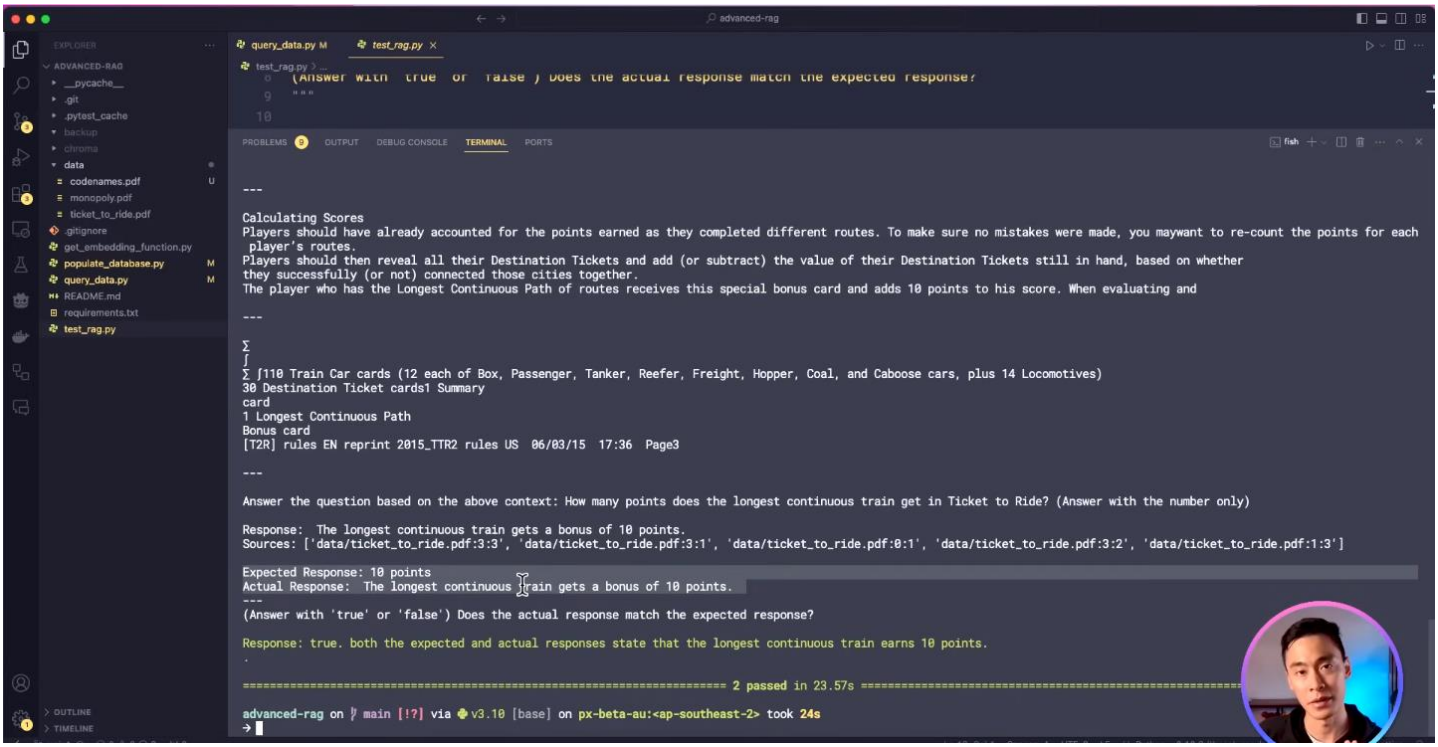
Answer the question based on the above context: How many points does the longest continuous train get in Ticket to Ride? (Answer with the number only)
Response: The longest continuous train gets a bonus of 10 points.
Sources: ['data/ticket_to_ride.pdf:3:3', 'data/ticket_to_ride.pdf:3:1', 'data/ticket_to_ride.pdf:0:1', 'data/ticket_to_ride.pdf:3:2', 'data/ticket_to_ride.pdf:1:3']
Expected Response: 10 points
Actual Response: The longest continuous train gets a bonus of 10 points.

(Answer with 'true' or 'false') Does the actual response match the expected response?
Response: true. both the expected and actual responses state that the longest continuous train earns 10 points.
.
========== 2 passed in 23.57s ==========
advanced-rag on main [!?] via v3.10 [base] on px-beta-au:<ap-southeast-2> took 24s

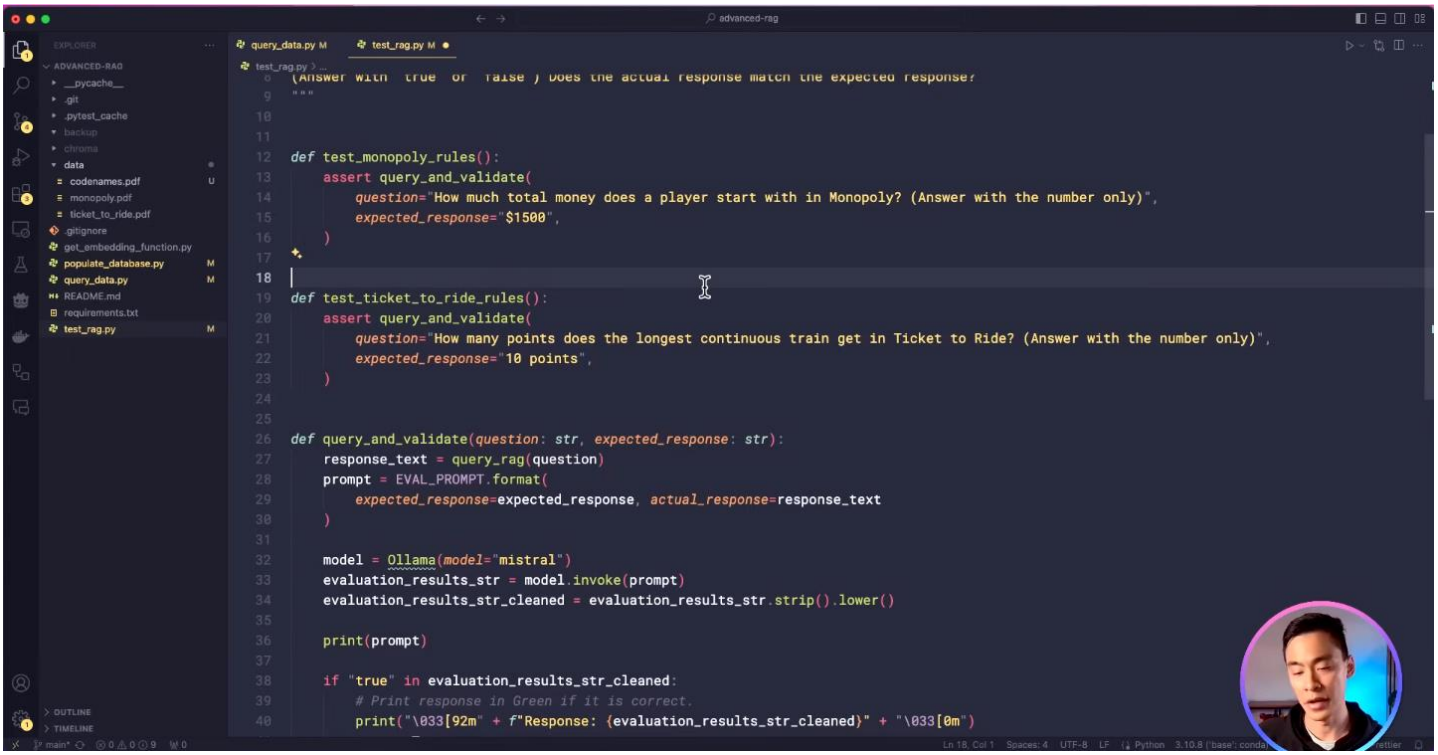

advanced-rag
query_data.py M
test_rag.py
(Answer with 'true' or 'false') Does the actual response match the expected response?
"""
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS
(toll free). Canadian consumers please write to: Hasbro Canada Corporation, 2350 de la Province, Longueuil, QC Canada, J4G 1G2.
The HASBRO, PARKER BROTHERS, and MONOPOLY names and logos, the distinctive design of the gameboard, the four comer squares, the MR. MONOPOLY name and character, and each of the distinctive elements of the board and rules are trademarks of Hasbro for its property trading game and game equipment. 02004,2007 Hasbro, Pawtucket, RI 02862.
All Rights Reserved. TM & @3 denote U.S. Trademarks.
00009-1
PROOF OF PURCHASE I

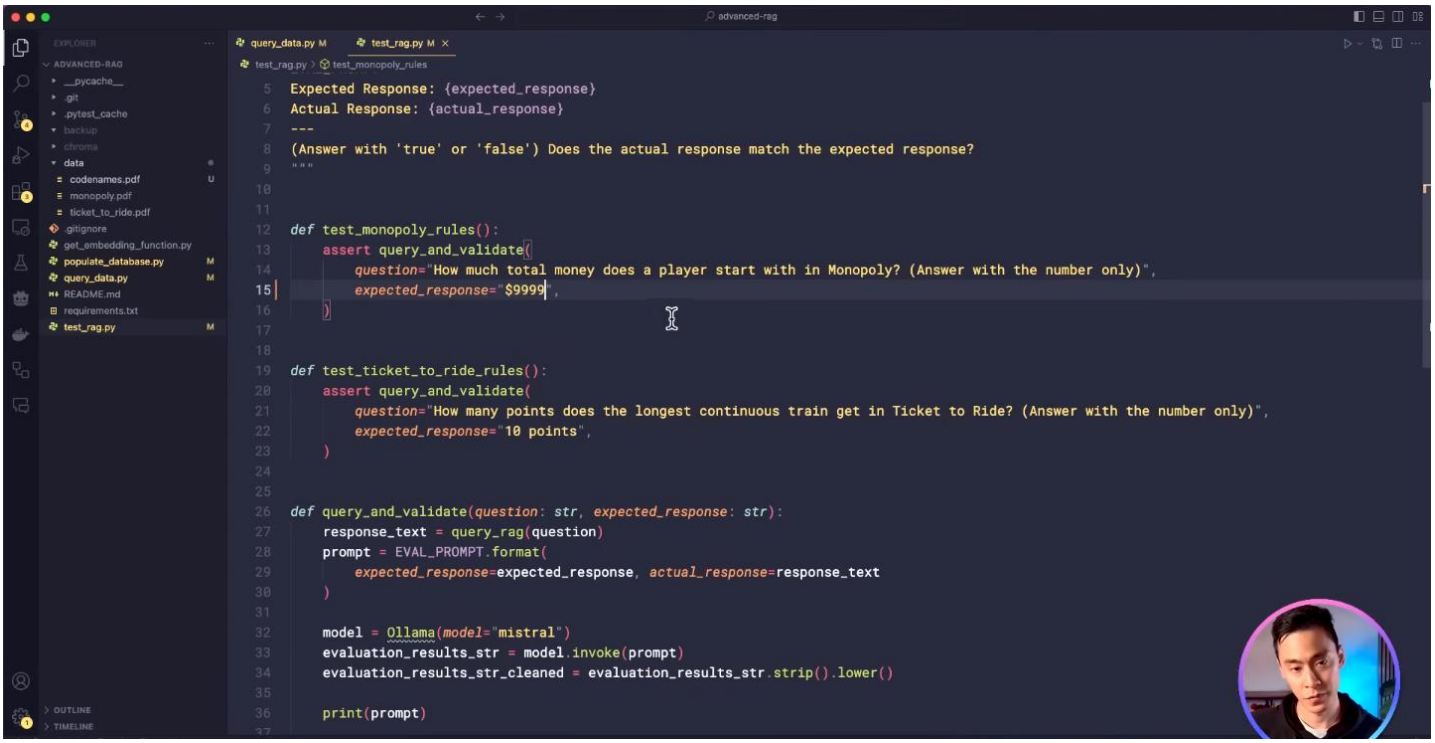
Answer the question based on the above context: How much total money does a player start with in Monopoly? (Answer with the number only)
Response: A player starts with $1,500 in Monopoly.
Sources: ['data/monopoly.pdf:1:1', 'data/monopoly.pdf:1:0', 'data/monopoly.pdf:0:0', 'data/monopoly.pdf:2:0', 'data/monopoly.pdf:7:2']
Expected Response: $1500
Actual Response: A player starts with $1,500 in Monopoly.

(Answer with 'true' or 'false') Does the actual response match the expected response?
Response: true. the expected response was a value of $1500 and the actual response stated that a player starts the game with $1,500.
.Human:
Answer the question based only on the following context:
comparing path lengths, only take into account continuous lines of plastic trains of the same color. A continuous path may include loops, and passthrough the same city several times, but a given plastic train may never be used twice in the same continuous path. In the case of a tie for the longestpath, all tied players score the 10 point bonus.
The player with the most points wins the game. If two or more players are tied for the most points, the player who has completed the most Destination Tickets wins. In the unlikely event that they are still tied, the player with the Longest Continuous Path card wins.
1Route Length Points Scored
2
3
4
5
61
2
4
7
10

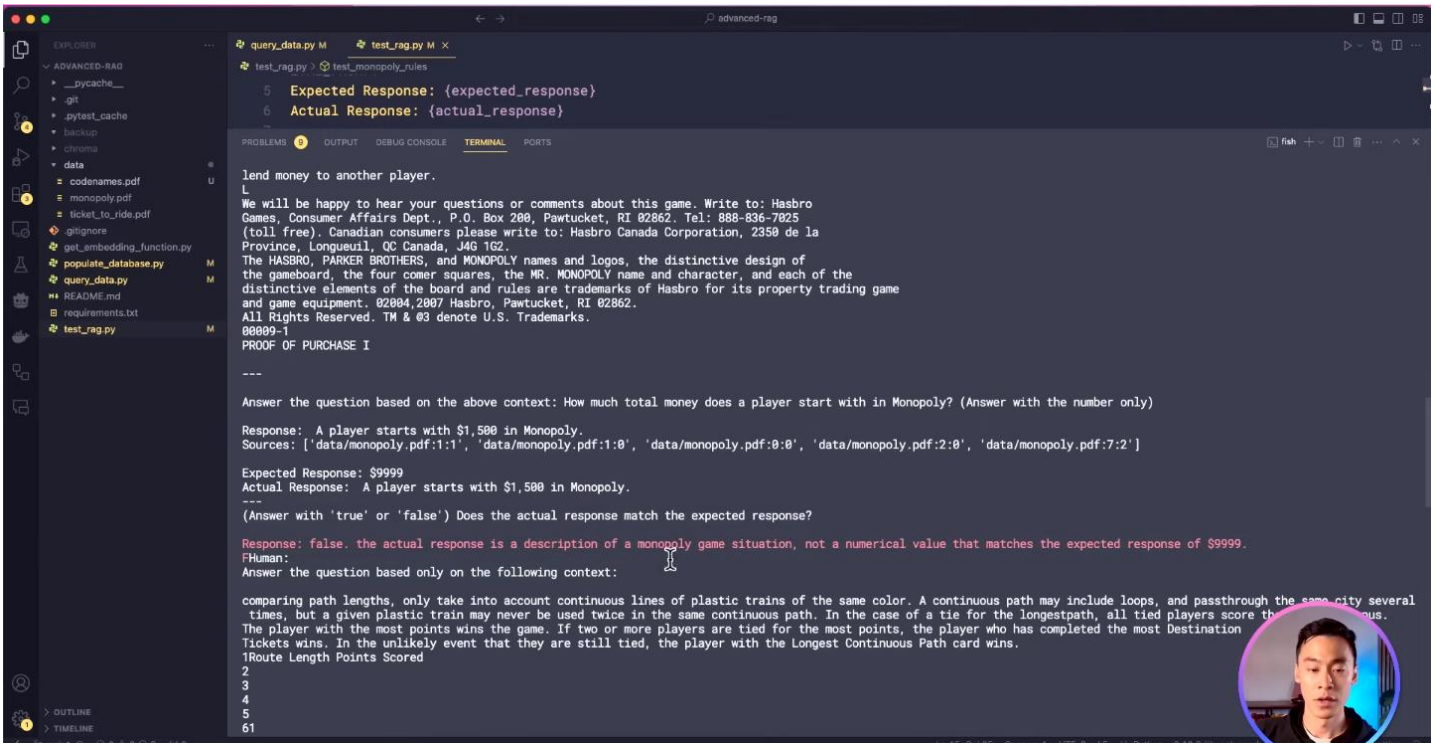


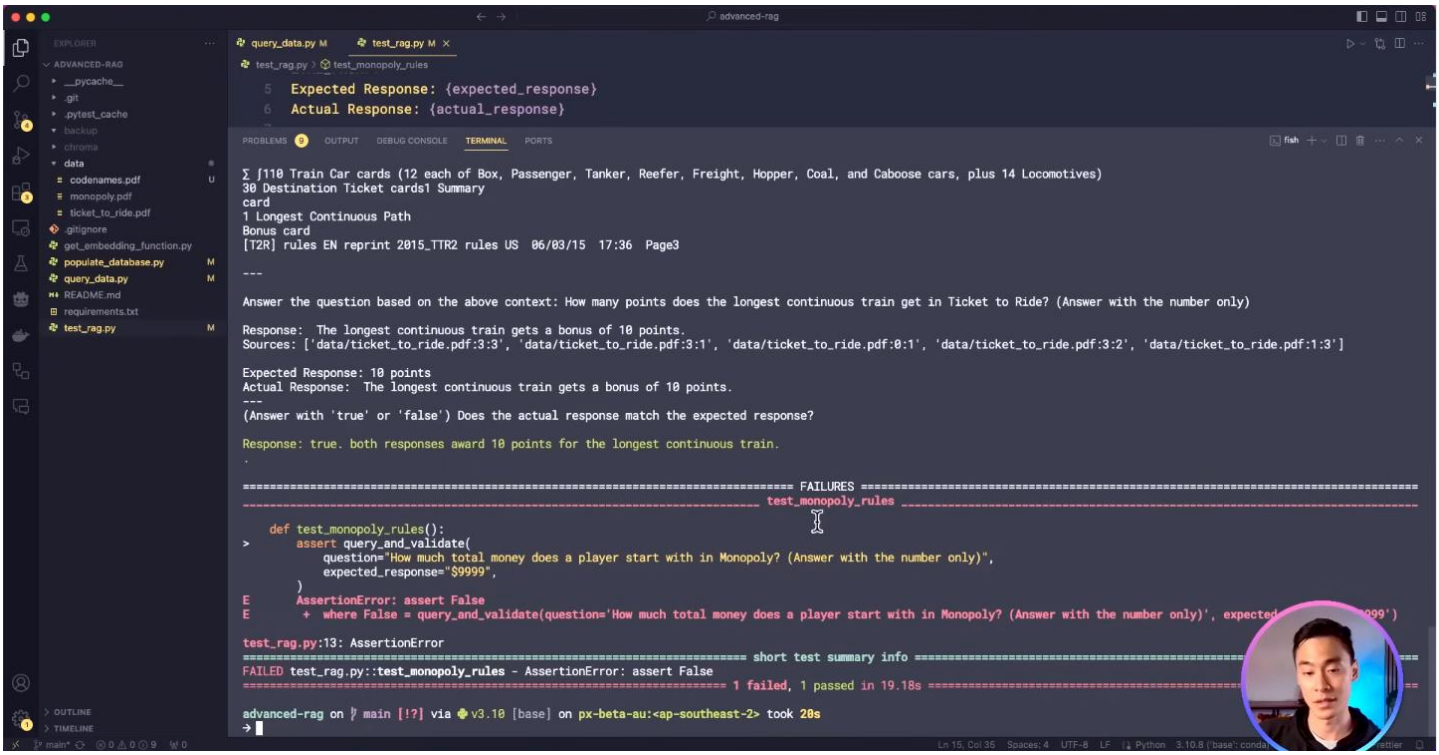
This works! But it is also good to do a negative test case to check that the LLM responses are indeed correct



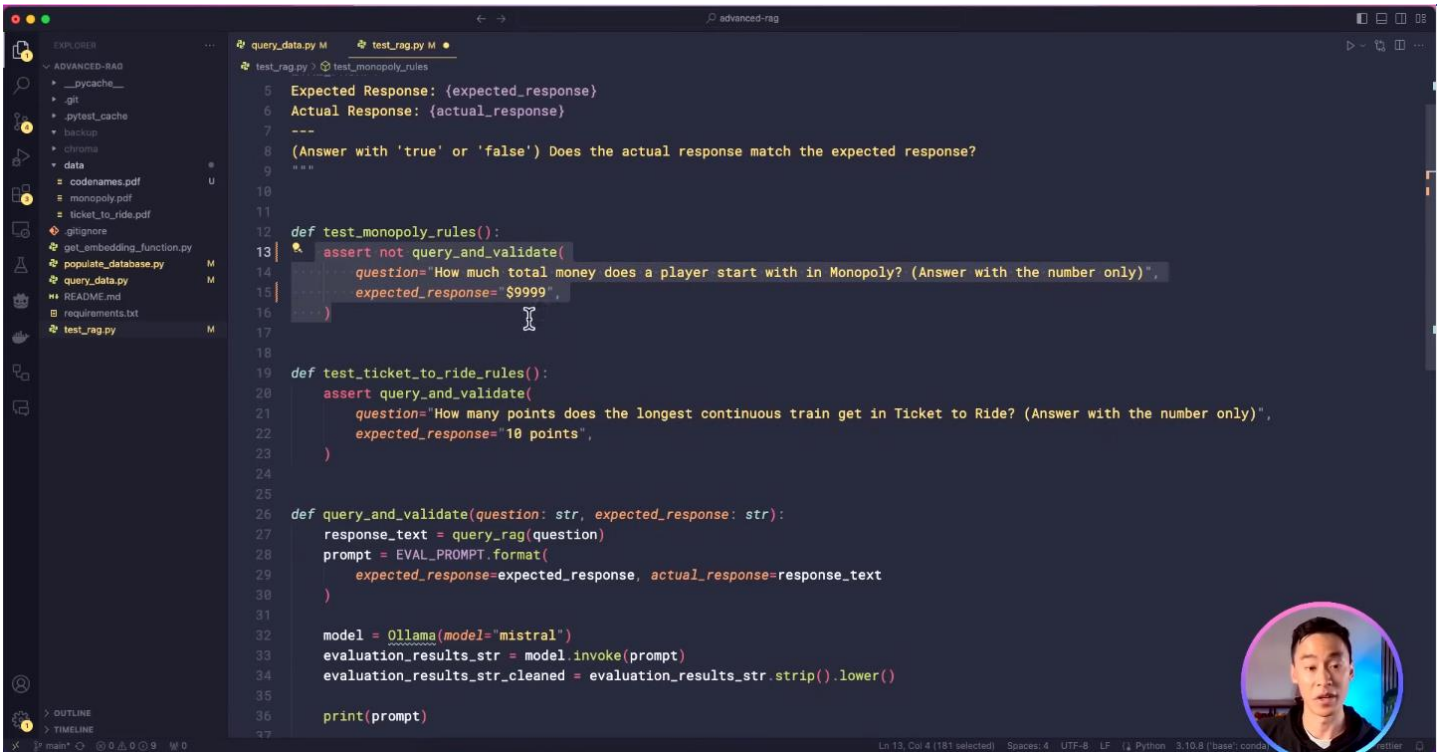


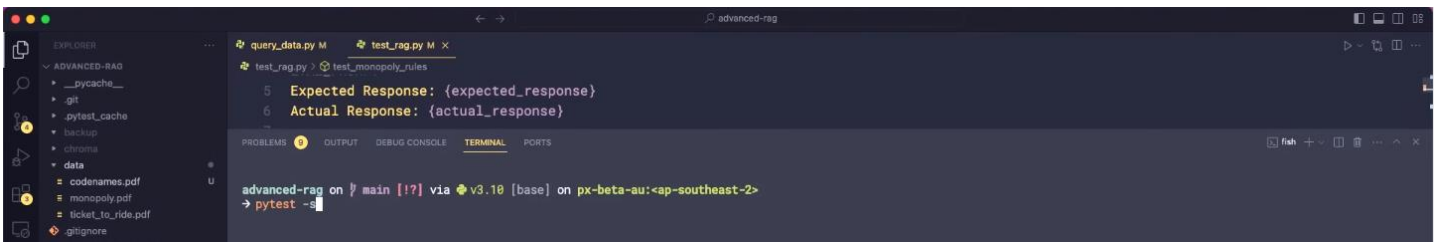
We change the 1500 to 999 so that the test should now fail.

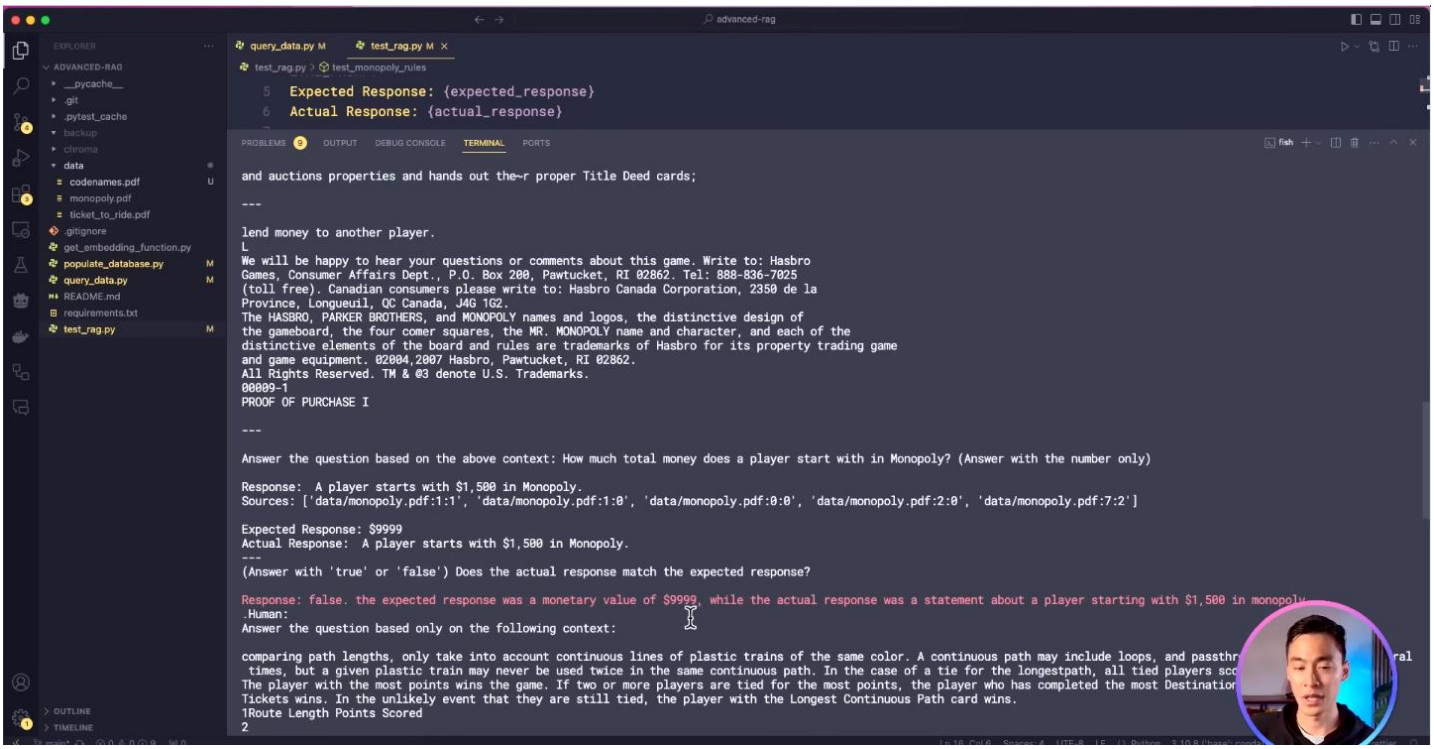


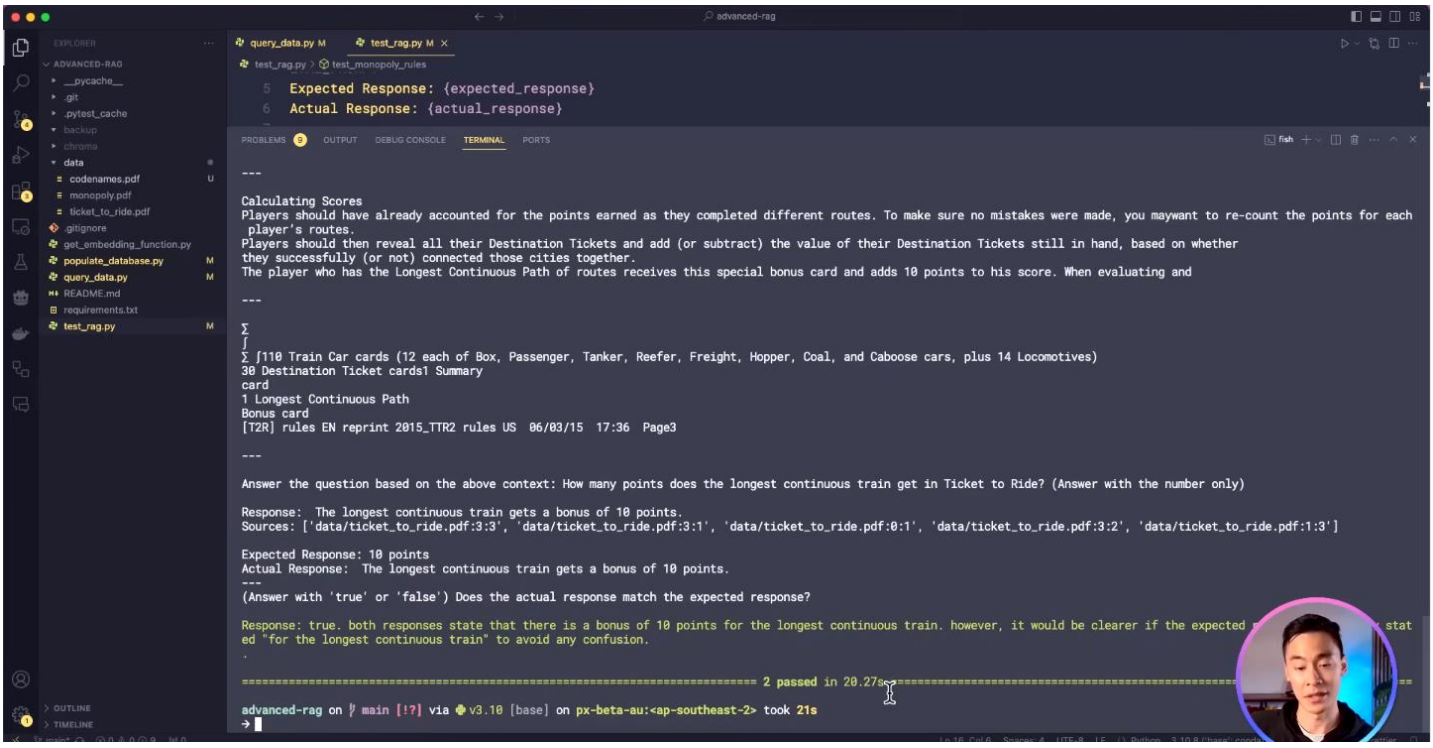


We can include this negative test in our test suite output by inverting the test assertion as below

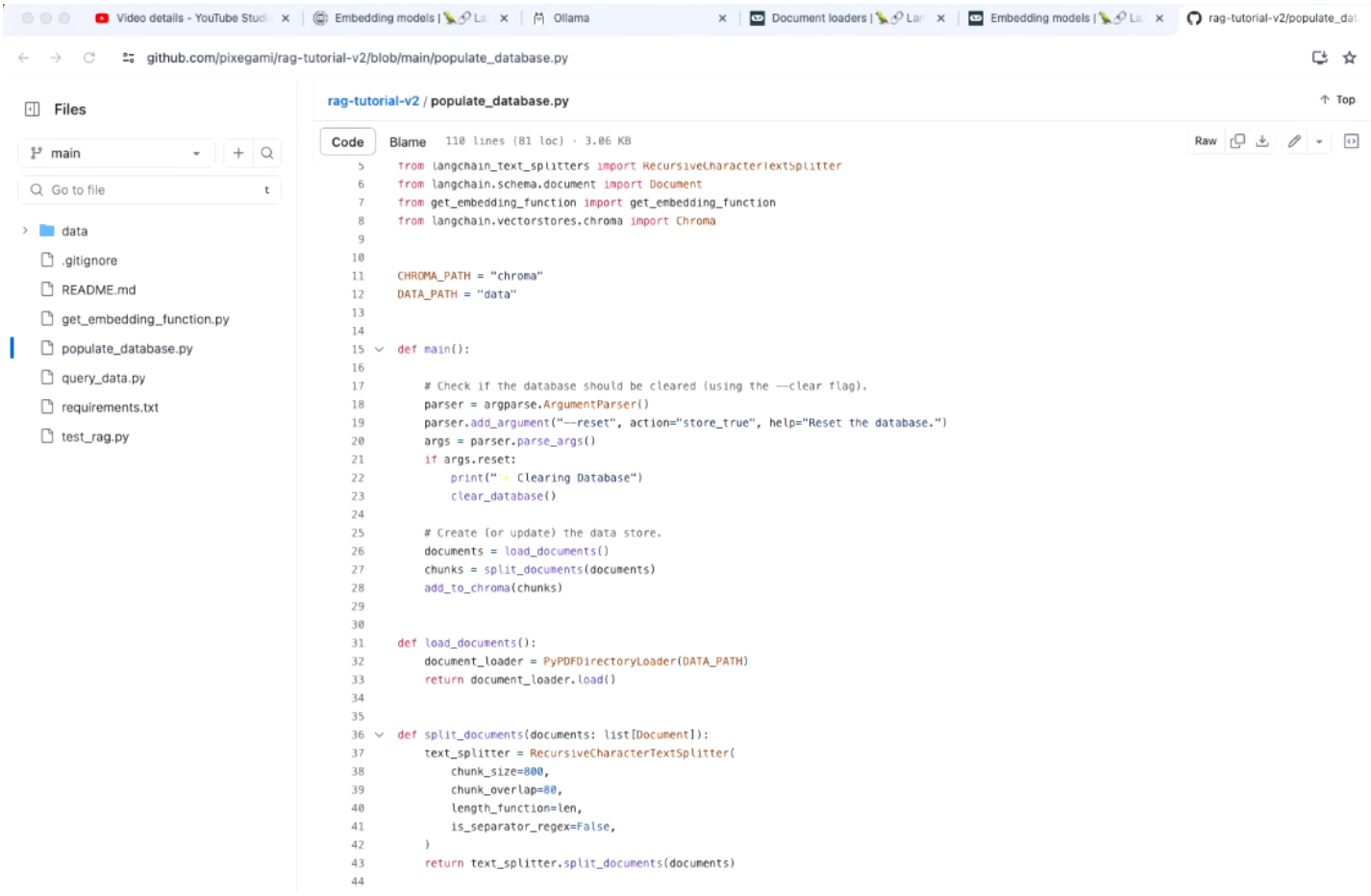








Our entire test suite is now passing as expected and we have both positive and negative test cases documented. We can set a test pass threshold of 90% to have our LLM still good enough for use.


Video details - YouTube Studi
Embedding models | 🦜🔗 La
Ollama
Document loaders | 🦜🔗 Lan
Embedding models | 🦜🔗 La
rag-tutorial-v2/populate_dat
github.com/pixegami/rag-tutorial-v2/blob/main/populate_database.py
Files
main
Go to file
t
data
.gitignore
README.md
get_embedding_function.py
populate_database.py
query_data.py
requirements.txt
test_rag.py
rag-tutorial-v2 / populate_database.py
Top
Code
Blame
110 lines (81 loc) · 3.06 KB
Raw
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain.schema.document import Document
from get_embedding_function import get_embedding_function
from langchain.vectorstores.chroma import Chroma
CHROMA_PATH = "chroma"
DATA_PATH = "data"
def main():
# Check if the database should be cleared (using the --clear flag).
parser = argparse.ArgumentParser()
parser.add_argument("--reset", action="store_true", help="Reset the database.")
args = parser.parse_args()
if args.reset:
print("✨ Clearing Database")
clear_database()
# Create (or update) the data store.
documents = load_documents()
chunks = split_documents(documents)
add_to_chroma(chunks)
def load_documents():
document_loader = PyPDFDirectoryLoader(DATA_PATH)
return document_loader.load()
def split_documents(documents: list[Document]):
text_splitter = RecursiveCharacterTextSplitter(
chunk_size=800,
chunk_overlap=80,
length_function=len,
is_separator_regex=False,
)
return text_splitter.split_documents(documents)