

ASSIGNMENT 1:
FINANCIAL PLANNING AGENT

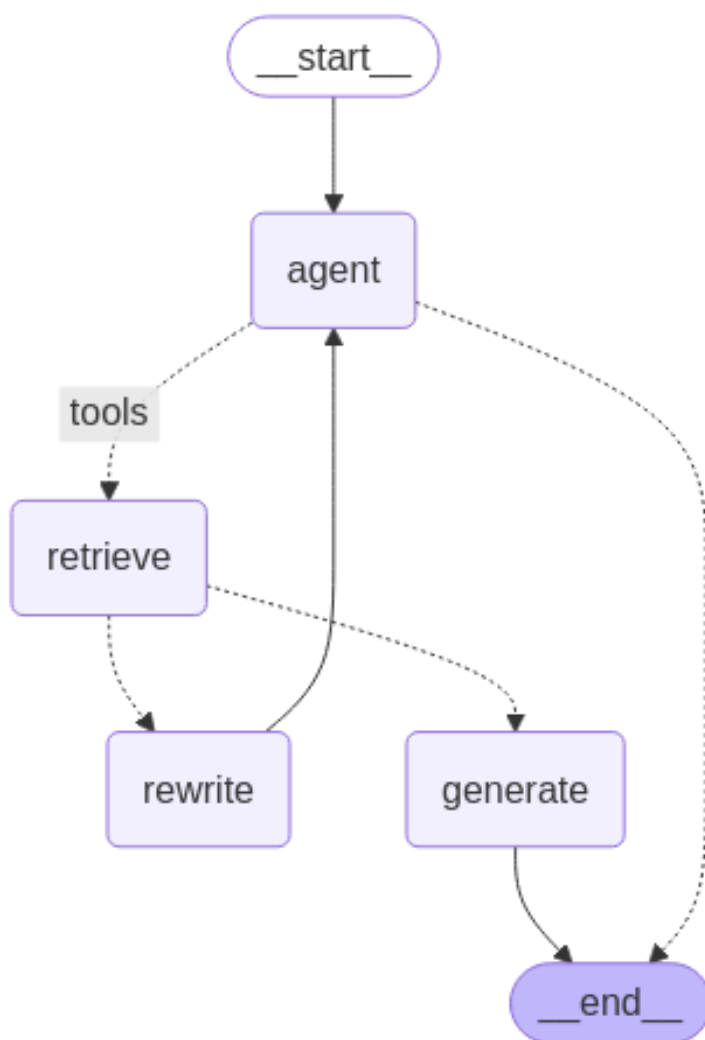
Kevin Geidel
MSDS 442: AI Agent Design & Development
Northwestern University
May 4, 2025

Requirement 1: Graph the Agent with LangChain/LangGraph

The construction of the agent and accompanying graph begins in cell 2 (see appendix). The actual assembly of the graph occurs in cell 7. However, some of the components, such as the retriever **ToolNode** and **AgentState** class, are built above. Following the logic in cell 7 we first instantiate an empty graph:

```
workflow = StateGraph(AgentState)
```

The workflow object has `add_node` and `add_edge` methods that allow us to assemble the components created in cells 2-6. The output is displayed graphically in cell 8 (reproduced below.)



Requirement 2: Load the Fidelity web pages

Loading the web pages occurs in in cell 2. A list of URLs are defined and based to the **WebBaseLoader**. The **RecursiveCharacterTextSplitter** chunks the documents into workable pieces. The result is a list of **LangChain** documents that each have a portion of each site. The output is sampled below.

```
10 # Use WebBaseLoader from langchain to load the fidelity.com webpage into a document.
11 docs = [WebBaseLoader(url).load() for url in uris]
12 docs_list = [item for sublist in docs for item in sublist]
13
14 text_splitter = RecursiveCharacterTextSplitter.from_tiktoken_encoder(
15     chunk_size=100, chunk_overlap=50
16 )
17 doc_splits = text_splitter.split_documents(docs_list)
18 print pprint(doc_splits)
```

Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Why Fidelity Funds...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Customer Service...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Portfolio Log In Required...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Cash Management...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Security Settings...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Fidelity Rewards...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Financial Basics...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Charitable Giving...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Stocks, ETFs, Crypto...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='EFTs...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Customer Service...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Home...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Message...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Facebook...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='What We Offer...')
Document(metadata={'source': 'https://www.fidelity.com/mutual-funds/why-fidelity-funds', 'title': 'Why Fidelity Funds', 'language': 'en-US'}, page_content='Active Management...')

Requirement 3: Store documents in ChromaDB

The documents are turned into vector embeddings and store in the ChromaDB vector store. This happens in cell 3. The embedding model used is **OpenAIEmbeddings**. The result is a collection of vectors with id's for use in retrieval.

```
1 # Requirement 3
2
3 # Store the document in the Chroma vector store
4 vector_store = Chroma.from_documents(
5     documents=doc_splits,
6     collection_name="rag-chroma",
7     embedding=OpenAIEmbeddings(),
8 )
```

[38] ✓ 3.8s

```
1 vector_store.get()
```

[43] ✓ 0.1s

```
{'ids': ['189b8d77-719b-4b88-a545-9c0a1c77ba98',
'ae554685-5fb9-45d0-9ac1-3a041eeb0d1f',
'd7ea06ea-40ea-4611-8b8e-27007610a1a6',
'70cf0776-c3c8-4796-8bf7-b84d15b4dcad',
'66d0e912-78a2-4813-9c34-756577481c94',
'25ef01e3-8286-498e-ba05-253435dab0d0',
'7c9bfbdef-2c7f-4bfa-9ed6-f06b52892fa7',
'e1d11b57-af7f-48de-81da-65adc2db8de5',
'2048ad8c-bec5-409c-9c4c-a5cadf81835b',
'7789086a-7d74-4a65-a3b4-e07b61287902',
'1da784e3-11aa-4c51-8569-cb4289db8c13',
'c6f4b77b-b4f4-4885-b286-b443f6-35-885f']}
```

Requirement 4: Retriever tool for relevance analysis

In cell 4 the retriever tool that will be used by the AI Agent to verify that the document is relevant to the question asked is created. The vector store has an `as_retriever` method. The tool is created from the retriever and labeled so the agent will know when to use it. The output is shown as the output of cell 4.

Requirement 5: Create a chain to determine relevance of question

Using the agent assembled in cell 7 we prepare to use the chain to evaluate if a question is relevant to the bot's function and knowledge base. The prompt is constructed in cell 6 in the `grade_documents` action. The system message is built from the prompt, which informs the LLM it is a financial advisor and spells out how the session will proceed. The output for cell 6 shows the rag prompt ready to be deployed.

Requirement 6: Prompt the graph

In cell 9 I construct a function, `query_agent`, that passes the user's message to the graph in a `HumanMessage` along with the system prompt. This allows use to query the agent in a DRY fashion. The prompt for requirement 6, *'What are the steps that I should take to determine how much I need to save for retirement planning'*, yielded the following response:

```
{ 'messages': [ "To maintain your preretirement lifestyle in retirement, it's "
                'generally recommended to save 10 times your preretirement '
                'income by age 67. This goal is based on the assumption of '
                'saving 15% of your income annually starting at age 25. '
                'Age-based milestones suggest aiming for 1x your income by age '
                '30, 3x by age 40, and 6x by age 50 to stay on track.'
                ]
}
```

(To see the full trace, including the steps followed by the agent, see the output of cell 10.)

Requirement 7: Inspect and evaluate response

I am not a financial planner but this advice seems sound. The agent recommends a decision tree that can be used to determine a quantitative answer for the question "How much do I need to retire?" In addition to providing the overall method for finding the answer to the question it expands by suggesting milestones that align with the answer.

Requirement 8: A similar prompt

Another prompt with similar semantic meaning is '*What strategies should someone take when behind on their retirement milestones?*'. When queried, the agent gave the following response:

```
AIMessage(content="I can help with questions related to Fidelity
mutual funds and retirement planning. If you're looking for strategies to catch up
on retirement milestones, here are some general tips:\n\n
1. **Assess Your Current Situation**: Take a close look at your current savings,
investments, and retirement accounts. Understand how much you have saved and how
much you need to retire comfortably.\n\n
2. **Increase Contributions**: If possible,
increase your contributions to retirement accounts. Consider maxing out contributions
to employer-sponsored plans like a 401(k) or an IRA.\n\n
3. **Catch-Up Contributions**: If you are age 50 or older, take advantage of catch-up
contributions that allow you to contribute more to your retirement accounts.\n\n
4. **Create a Budget**: Review your expenses and create a budget that allows you to
allocate more funds toward retirement savings. Cut unnecessary expenses where possible.
5. **Consider Additional Income Sources**: Look for ways to increase your income, such
as taking on a part-time job, freelancing, or selling unused items.\n\n
6. **Invest Wisely**: Ensure that your investments are aligned with your retirement goals.
Consider a diversified portfolio that balances risk and growth potential.\n\n
7. **Delay Retirement**: If feasible, consider delaying your retirement age. Working
longer can increase your savings and reduce the number of years you need to draw
from your retirement funds.\n\n
8. **Consult a Financial Advisor**: A financial advisor
can provide personalized advice based on your specific situation and help you
create a plan to get back on track.\n\n
If you have specific questions about Fidelity
mutual funds or retirement planning, feel free to ask!",
```

```

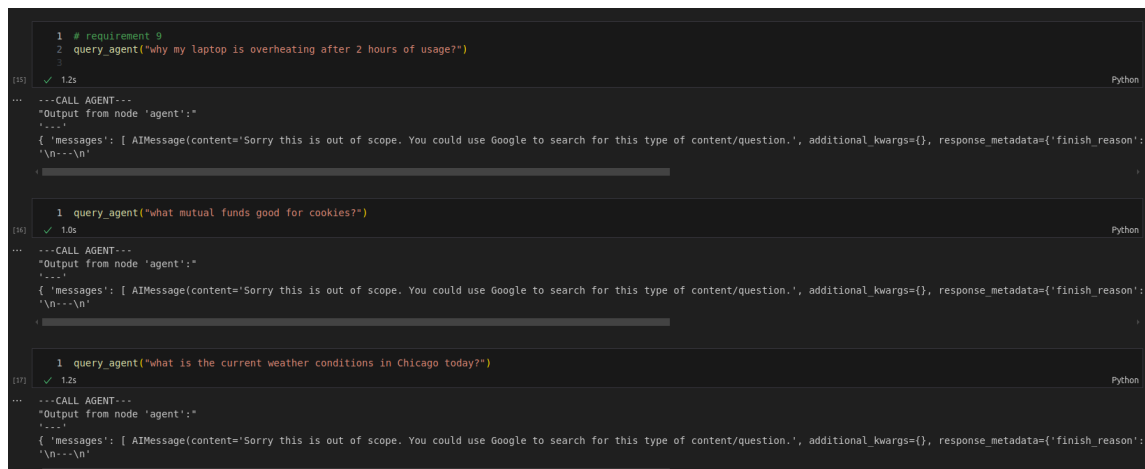
additional_kwargs={}, response_metadata={
    'finish_reason': 'stop', 'model_name': 'gpt-4o-mini-2024-07-18',
    'system_fingerprint': 'fp_129a36352a'
}, id='run-47220c96-088b-4719-8764-1b241a224edf-0')

```

This response provides some specific steps for catching up with retirement milestones!

Requirement 9: Irrelevant prompt

When passed a prompt that is not relevant to the knowledge base documents the agent responded accordingly.



```

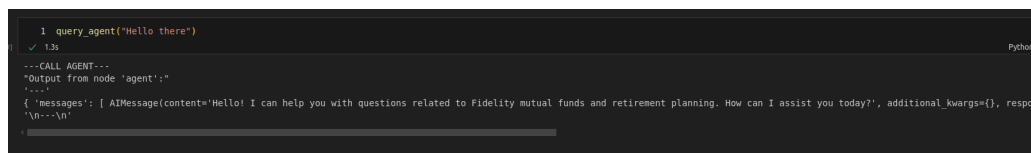
1 # requirement 9
2 query_agent("why my laptop is overheating after 2 hours of usage?")
3
[15] ✓ 1.2s Python
---CALL AGENT---
"Output from node 'agent':"
....
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use Google to search for this type of content/question.', additional_kwargs={}, response_metadata={'finish_reason':
'\n---\n'

1 query_agent("what mutual funds good for cookies?")
[16] ✓ 1.0s Python
---CALL AGENT---
"Output from node 'agent':"
....
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use Google to search for this type of content/question.', additional_kwargs={}, response_metadata={'finish_reason':
'\n---\n'

1 query_agent("what is the current weather conditions in Chicago today?")
[17] ✓ 1.2s Python
---CALL AGENT---
"Output from node 'agent':"
....
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use Google to search for this type of content/question.', additional_kwargs={}, response_metadata={'finish_reason':
'\n---\n'

```

The agent also responded appropriately to the ‘greeting-like’ prompt:



```

1 query_agent("Hello there")
2
[18] ✓ 1.3s Python
---CALL AGENT---
"Output from node 'agent':"
....
{ 'messages': [ AIMessage(content='Hello! I can help you with questions related to Fidelity mutual funds and retirement planning. How can I assist you today?', additional_kwargs={}, respon
'\n---\n'

```

Cells 10-20 show the queries made to the agent graph. Relevant prompts are answered with quality responses. Irrelevant prompts are flagged out of scope. The question about Fidelity specific funds are answered and the prompt about Schwab funds are correctly identified as out of scope.

Requirement 10: Load a PDF into context

In cell 21 the *FFFGX.pdf* document is loaded using the **PyPDFLoader**. In cell 22 we view the meta data and content. In cell 23 the document is added to the ChromaDB vector store. Finally, in cells 24 and 25 we display a graphic representation of what was processed.

Requirement 11: Test the context with queries

In cell 26 I construct a function, **query_llm**, that can be used to query an OpenAI chat client in a DRY manner. Cells 27-30 show the prompts and responses. The LLM, using the PDF context, was able to answer questions from the text and tables of the document. Very impressive libraries on display here!

```

1 query_llm("what is the name of this fund?")
1.5s
Python
... The name of the fund is the **Fidelity Freedom® 2045 Fund (FFFGX)**.
```

```

1 query_llm("who is the fund manager?")
2.6s
Python
... The fund managers for the Fidelity Freedom® 2045 Fund (FFFGX) are Andrew J. Dieford (Co-Manager since June 21, 2011) and Brett F. Summers (Co-Manager).
```

```

1 query_llm("what is the calendar year return for 2022 for this fund and S&P 500?")
2.1s
Python
... The calendar year return for 2022 is as follows:
- **Fidelity Freedom® 2045 Fund (FFFGX)**: -18.20%
- **S&P 500®**: -18.11%
```

```

1 query_llm("what is the Portfolio Net Assets")
2.1s
Python
... The Portfolio Net Assets of the Fidelity Freedom® 2045 Fund (FFFGX) as of 11/30/2024 are $21,463,148, and it has Share Class Net Assets of $4,597,927 as of the same date.
```

```

1 query_llm("what is the Morningstar rating for this fund? How many funds were used to rate this fund?")
3.3s
Python
... The Morningstar rating for the Fidelity Freedom® 2045 Fund (FFFGX) is 4 stars, and it is ranked within the "Target-Date 2045" category.
The number of funds used to rate this fund is 199.
```

```
[1]: #####
# MSDS 442: AI Agent Design and Development
# Spring '25
# Dr. Bader
#
# Assignment 1 - Financial Planning Agent
#
# Kevin Geidel
#####

# OBJECTIVE:
# The following will construct an AI agent using the LangChain & LangGraph
→ frameworks
# The agent will act as a Fidelity financial advisor to answer user questions
→ about retirement planning.

# Load environment variables
from dotenv import load_dotenv
load_dotenv()

# Python native imports
import os, sys, pprint

# Third party library import
from langchain import hub
from langchain_core.prompts import PromptTemplate
from langchain_community.document_loaders import PyPDFLoader
from langchain_openai import ChatOpenAI, OpenAIEmbeddings
from langchain_core.output_parsers import StrOutputParser
from langgraph.prebuilt import tools_condition
from langgraph.graph import END, StateGraph, START
from langchain_core.vectorstores import InMemoryVectorStore
from langgraph.prebuilt import ToolNode
from langchain.document_loaders import WebBaseLoader
os.environ['USER_AGENT'] = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
→ AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'
__import__('pysqlite3')
sys.modules['sqlite3'] = sys.modules.pop('pysqlite3')
from langchain_chroma import Chroma
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain.tools.retriever import create_retriever_tool
from typing import Annotated, Sequence, Literal
from typing_extensions import TypedDict
from langchain_core.messages import BaseMessage, HumanMessage
from langgraph.graph.message import add_messages
from pydantic import BaseModel, Field
```



```
# Assign experiment-wide variables
model_name = 'gpt-4o-mini'
```

[2]: *# Requirments 1 and 2*

```
urls = [
    "https://www.fidelity.com/mutual-funds/why-fidelity-funds",
    "https://www.fidelity.com/mutual-funds/fidelity-funds/overview",
    "https://www.fidelity.com/viewpoints/retirement/
    ↪how-much-do-i-need-to-retire",
]

# Use WebBaseLoader from LangChain to load the Fidelity.com webpage into a
↪document.
docs = [WebBaseLoader(url).load() for url in urls]
docs_list = [item for sublist in docs for item in sublist]

text_splitter = RecursiveCharacterTextSplitter.from_tiktoken_encoder(
    chunk_size=100, chunk_overlap=50
)
doc_splits = text_splitter.split_documents(docs_list)
```

[3]: *# Requirement 3*

```
# Store the document in the Chroma vector store
vector_store = Chroma.from_documents(
    documents=doc_splits,
    collection_name="rag-chroma",
    embedding=OpenAIEmbeddings(),
)
```

```
[4]: # Requirement 4

# Create a retriever tool that will be used by the AI Agent to verify that the
→document is relevant to the prompt
retriever = vector_store.as_retriever()

retriever_tool = create_retriever_tool(
    retriever,
    "retrieve_fidelity_docs",
    """Search Fidelity Knowledge base in the vector store and return information
→for:
    - Calculating how much money a person needs to retire
    - Mutual Fund names
    - Types of funds""",
)

tools = [retriever_tool]
tools
```

```
[4]: [Tool(name='retrieve_fidelity_docs', description='Search Fidelity Knowledge base
in the vector store and return information for:\n          - Calculating how much
money a person needs to retire\n          - Mutual Fund names\n          - Types of
funds', args_schema=<class 'langchain_core.tools.retriever.RetrieverInput'>,
func=functools.partial(<function _get_relevant_documents at 0x7fa09df57c40>,
retriever=VectorStoreRetriever(tags=['Chroma', 'OpenAIEmbeddings'],
vectorstore=<langchain_chroma.vectorstores.Chroma object at 0x7fa09dd50790>,
search_kwargs={})),
document_prompt=PromptTemplate(input_variables=['page_content'], input_types={},
partial_variables={}, template='{page_content}'), document_separator='\n\n',
response_format='content'), coroutine=functools.partial(<function
_aget_relevant_documents at 0x7fa09df57d80>,
retriever=VectorStoreRetriever(tags=['Chroma', 'OpenAIEmbeddings'],
vectorstore=<langchain_chroma.vectorstores.Chroma object at 0x7fa09dd50790>,
search_kwargs={})),
document_prompt=PromptTemplate(input_variables=['page_content'], input_types={},
partial_variables={}, template='{page_content}'), document_separator='\n\n',
response_format='content'))]
```

```
[5]: # The following classes are used to maintain agent state
# and a agent graph that maps the nodes and edges

class AgentState(TypedDict):
    # The add_messages function defines how an update should be processed
    # Default is to replace. add_messages says "append"
    messages: Annotated[Sequence[BaseMessage], add_messages]
```

```
[6]: ### Edges

def grade_documents(state) -> Literal["generate", "rewrite"]:
    """
    Determines whether the retrieved documents are relevant to the question.

    Args:
    state (messages): The current state

    Returns:
    str: A decision for whether the documents are relevant or not
    """

    print("---CHECK RELEVANCE---")

    # Data model
    class grade(BaseModel):
        """Binary score for relevance check."""

        binary_score: str = Field(description="Relevance score 'yes' or 'no'")

    # LLM
    # model = ChatOpenAI(temperature=0, model="gpt-4-0125-preview",
    ↪streaming=True)
    model = ChatOpenAI(temperature=0, model="gpt-4o-mini", streaming=True)

    # LLM with tool and validation
    llm_with_tool = model.with_structured_output(grade)

    # Prompt
    prompt = PromptTemplate(
        template="""You are a financial advisor answering user question about
    ↪retirement planning. \n
        Here is the retrieved document: \n\n {context} \n\n
        Here is the user question: {question} \n
        If the document contains keyword(s) or semantic meaning related to the
    ↪user question, grade it as relevant. \n
        Give a binary score 'yes' or 'no' score to indicate whether the document
    ↪is relevant to the question.""",
        input_variables=["context", "question"],
    )

    # Chain
    chain = prompt | llm_with_tool

    messages = state["messages"]
    last_message = messages[-1]
```

```

question = messages[0].content
docs = last_message.content

scored_result = chain.invoke({"question": question, "context": docs})

score = scored_result.binary_score

if score == "yes":
    print("---DECISION: DOCS RELEVANT---")
    return "generate"

else:
    print("---DECISION: DOCS NOT RELEVANT---")
    print(score)
    return "rewrite"

### Nodes

def agent(state):
    """
    Invokes the agent model to generate a response based on the current state.
    ↳ Given
    the question, it will decide to retrieve using the retriever tool, or simply
    ↳ end.

    Args:
        state (messages): The current state

    Returns:
        dict: The updated state with the agent response appended to messages
    """
    print("---CALL AGENT---")
    messages = state["messages"]
    model = ChatOpenAI(temperature=0, streaming=True, model="gpt-4o-mini")

    model = model.bind_tools(tools)
    response = model.invoke(messages)
    return {"messages": [response]}

def rewrite(state):
    """
    Transform the query to produce a better question.

    Args:
        state (messages): The current state

```

```

Returns:
    dict: The updated state with re-phrased question
    """

print("---TRANSFORM QUERY---")
messages = state["messages"]
question = messages[0].content

msg = [
    HumanMessage(
        content=f""" \n
        Look at the input and try to reason about the underlying semantic intent /_
↪meaning. \n
        Here is the initial question:
        \n ----- \n
        {question}
        \n ----- \n
        Formulate an improved question: """,
    )
]

# Grader
model = ChatOpenAI(temperature=0, model="gpt-4o-mini", streaming=True)
response = model.invoke(msg)
return {"messages": [response]}

def generate(state):
    """
    Generate answer

    Args:
        state (messages): The current state

    Returns:
        dict: The updated state with re-phrased question
        """
    print("---GENERATE---")
    messages = state["messages"]
    question = messages[0].content
    last_message = messages[-1]

    docs = last_message.content

    # Prompt
    prompt = hub.pull("rlm/rag-prompt")

```

```

# LLM
llm = ChatOpenAI(model_name="gpt-4o-mini", temperature=0, streaming=True)

# Post-processing
def format_docs(docs):
    return "\n\n".join(doc.page_content for doc in docs)

# Chain
rag_chain = prompt | llm | StrOutputParser()

# Run
response = rag_chain.invoke({"context": docs, "question": question})
return {"messages": [response]}

print("*" * 20 + "Prompt[rlm/rag-prompt]" + "*" * 20)
prompt = hub.pull("rlm/rag-prompt").pretty_print()

```

```

*****Prompt[rlm/rag-prompt]*****
===== Human Message
=====

```

You are an assistant for question-answering tasks. Use the following pieces of retrieved context to answer the question. If you don't know the answer, just say that you don't know. Use three sentences maximum and keep the answer concise.

Question: {question}

Context: {context}

Answer:

[7]: *# Requirment 5*

```

# Define a new graph
workflow = StateGraph(AgentState)

# Define the nodes we will cycle between
workflow.add_node("agent", agent) # agent
retrieve = ToolNode([retriever_tool])
workflow.add_node("retrieve", retrieve) # retrieval
workflow.add_node("rewrite", rewrite) # Re-writing the question
workflow.add_node(
    "generate", generate
) # Generating a response after we know the documents are relevant
# Call agent node to decide to retrieve or not
workflow.add_edge(START, "agent")

# Decide whether to retrieve

```

```

workflow.add_conditional_edges(
    "agent",
    # Assess agent decision
    tools_condition,
    {
        # Translate the condition outputs to nodes in our graph
        "tools": "retrieve",
        END: END,
    },
)

# Edges taken after the `action` node is called.
workflow.add_conditional_edges(
    "retrieve",
    # Assess agent decision
    grade_documents,
)
workflow.add_edge("generate", END)
workflow.add_edge("rewrite", "agent")

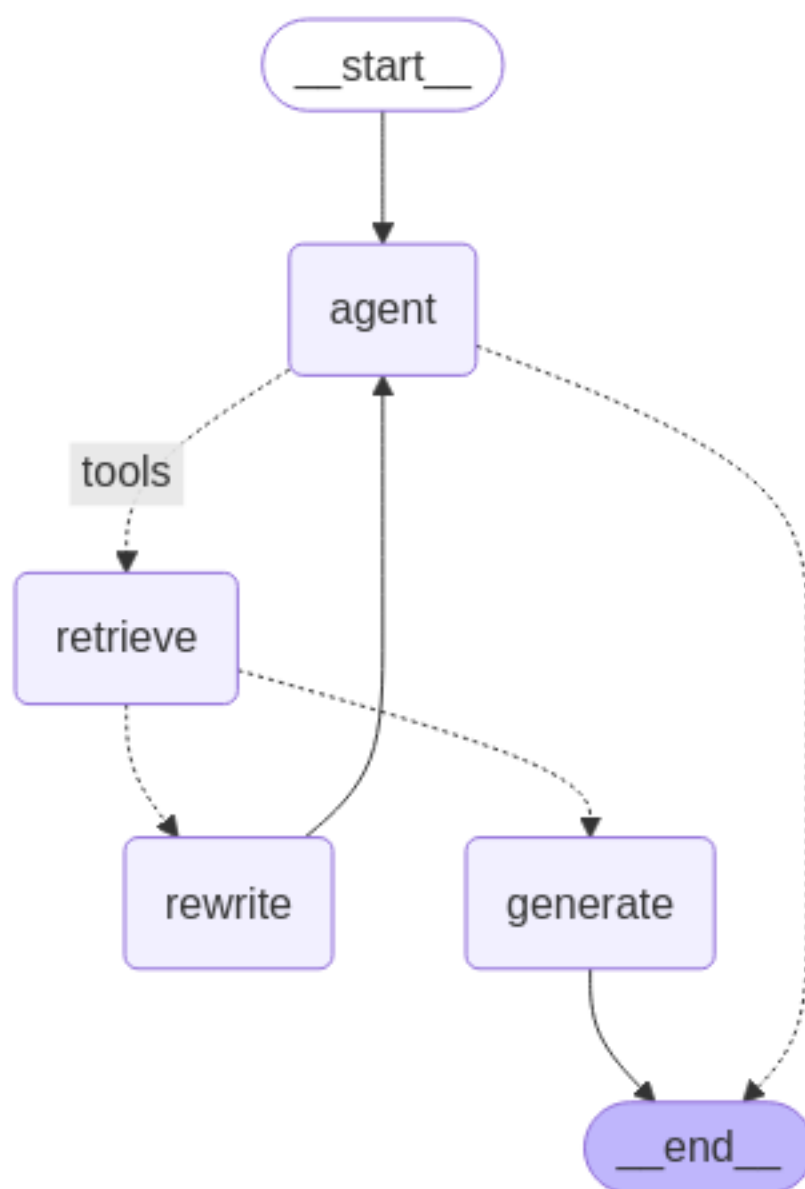
# Compile
graph = workflow.compile()

```

```

[8]: # View the graph
from IPython.display import Image, display
display(Image(graph.get_graph(xray=True).draw_mermaid_png()))

```




```
[9]: # Assemble the function that will send a user message to the agent

def query_agent(user_message):
    inputs = {
        "messages": [
            ("system", """
                Instructions:
                - Check on the user question, query, or prompt content,
                and if it is related to fidelity mutual fund, names of available mutual
                funds, or retirement
                use the tools to retrieve data from your knowledge
                base in the vector store.
                - Provide detailed answer for questions related to
                retirement or mutual funds based on the knowledge base in the vector store.
                - For greeting type messages, respond with message that
                you can help with questions related to Fidelity mutual funds and retirement
                planning,
                otherwise reply with the message: Sorry this is out
                of scope. You could use Google to search for this type of content/question"""),
            ("user", user_message),
        ]
    }
    for output in graph.stream(inputs):
        for key, value in output.items():
            pprint.pprint(f"Output from node '{key}':")
            pprint.pprint("---")
            pprint.pprint(value, indent=2, width=80, depth=None)
            pprint.pprint("\n---\n")
```

```
[10]: # requirement #6
query_agent("What are the steps that I should take to determine how much I need
to save for retirement planning")
```

---CALL AGENT---

```
"Output from node 'agent':"
'---'
```

```
{ 'messages': [ AIMessage(content='', additional_kwargs={'tool_calls':
[{'index': 0, 'id': 'call_4s1cpKB0UkX6KXvyRH9If666', 'function': {'arguments':
'{"query":"how much money a person needs to retire"}', 'name':
'retrieve_fidelity_docs'}, 'type': 'function'}]]},
response_metadata={'finish_reason': 'tool_calls', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-c9ad04d5-79e7-47f0-a31e-ebc47ba602e8-0', tool_calls=[{'name':
'retrieve_fidelity_docs', 'args': {'query': 'how much money a person needs to
retire'}, 'id': 'call_4s1cpKB0UkX6KXvyRH9If666', 'type': 'tool_call'}])}]}
```

```

'\n---\n'
---CHECK RELEVANCE---
---DECISION: DOCS RELEVANT---
"Output from node 'retrieve':"
'----'
{ 'messages': [ ToolMessage(content="Consider some hypothetical examples (see
graphic). Max plans to delay retirement until age 70, so he will need to have
saved 8x his final income to sustain his preretirement lifestyle. Amy wants to
retire at age 67, so she will need to have saved 10x her preretirement income.
John plans to retire at age 65, so he would need to have saved at least 12x his
preretirement income.\n\nOur savings factors are based on the assumption that a
person saves 15% of their income annually beginning at age 25 (which includes
any employer match), invests more than 50% on average of their savings in stocks
over their lifetime, retires at age 67, and plans to maintain their
preretirement lifestyle in retirement (see footnote 1 for more
details).\n\nBased on those assumptions, we estimate that saving 10x (times)
your preretirement income by age 67, together with other steps, should help
ensure that you have enough income to maintain your current lifestyle in
retirement. That 10x goal may seem ambitious. But you have many years to get
there. To help you stay on track, we suggest these age-based milestones: Aim to
save at least 1x your income by age 30, 3x by 40, 6x by 50,\n\nLet's look at
some hypothetical investors who are planning to retire at 67. Joe is planning to
downsize and live frugally in retirement, so he expects his expenses to be
lower. His savings factor might be closer to 8x than 10x. Elizabeth is planning
to retire at age 67 and her goal is to maintain her lifestyle in retirement, so
her savings factor is 10x. Sean sees retirement as an opportunity to travel
extensively, so it may make sense for him to save more and",
name='retrieve_fidelity_docs', id='83bca74d-095b-45c3-a739-108a57536c41',
tool_call_id='call_4s1cpKB0UkX6KXvyRH9If666')]}}
'\n---\n'
---GENERATE---
"Output from node 'generate':"
'----'
{ 'messages': [ "To maintain your preretirement lifestyle in retirement, it's "
'generally recommended to save 10 times your preretirement '
'income by age 67. This goal is based on the assumption of '
'saving 15% of your income annually starting at age 25. '
'Age-based milestones suggest aiming for 1x your income by age '
'30, 3x by age 40, and 6x by age 50 to stay on track.']]
'\n---\n'

```

```
[11]: # requirement #7

# I am not a financial planner but this advice seems sound. The agent recommends
↳a
# decision tree that can be used to determine a quantitative answer for the
↳question
# "How much do I need to retire?"
```

```
[13]: # requirement #8

query_agent("What strategies should someone take when behind on their retirement
↳milestones?")
```

---CALL AGENT---

"Output from node 'agent':"

'---'

```
{ 'messages': [ AIMessage(content="I can help with questions related to Fidelity
mutual funds and retirement planning. If you're looking for strategies to catch
up on retirement milestones, here are some general tips:\n\n1. **Assess Your
Current Situation**: Take a close look at your current savings, investments, and
retirement accounts. Understand how much you have saved and how much you need to
retire comfortably.\n\n2. **Increase Contributions**: If possible, increase your
contributions to retirement accounts. Consider maxing out contributions to
employer-sponsored plans like a 401(k) or an IRA.\n\n3. **Catch-Up
Contributions**: If you are age 50 or older, take advantage of catch-up
contributions that allow you to contribute more to your retirement
accounts.\n\n4. **Create a Budget**: Review your expenses and create a budget
that allows you to allocate more funds toward retirement savings. Cut
unnecessary expenses where possible.\n\n5. **Consider Additional Income
Sources**: Look for ways to increase your income, such as taking on a part-time
job, freelancing, or selling unused items.\n\n6. **Invest Wisely**: Ensure that
your investments are aligned with your retirement goals. Consider a diversified
portfolio that balances risk and growth potential.\n\n7. **Delay Retirement**:
If feasible, consider delaying your retirement age. Working longer can increase
your savings and reduce the number of years you need to draw from your
retirement funds.\n\n8. **Consult a Financial Advisor**: A financial advisor can
provide personalized advice based on your specific situation and help you create
a plan to get back on track.\n\nIf you have specific questions about Fidelity
mutual funds or retirement planning, feel free to ask!", additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-47220c96-088b-4719-8764-1b241a224edf-0')]}
```

'\n---\n'

```
[14]: # This response provides some specific steps for catching up with retirement_
      ↳ milestones!
```

```
[15]: # requirement 9
      query_agent("why my laptop is overheating after 2 hours of usage?")
```

```
---CALL AGENT---
```

```
"Output from node 'agent':"
'___'
```

```
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use
Google to search for this type of content/question.', additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-0772a5c2-722b-4346-a1ac-b506c71fa2fa-0')] }
'\n---\n'
```

```
[16]: query_agent("what mutual funds good for cookies?")
```

```
---CALL AGENT---
```

```
"Output from node 'agent':"
'___'
```

```
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use
Google to search for this type of content/question.', additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-c6b68c68-c899-43dc-b195-cf249b25a755-0')] }
'\n---\n'
```

```
[17]: query_agent("what is the current weather conditions in Chicago today?")
```

```
---CALL AGENT---
```

```
"Output from node 'agent':"
'___'
```

```
{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use
Google to search for this type of content/question.', additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-31a9786c-bedd-4d18-bc98-32801b667454-0')] }
'\n---\n'
```

```
[18]: query_agent("Hello there")
```

```
---CALL AGENT---
```

```
"Output from node 'agent':"
'___'
```

```
{ 'messages': [ AIMessage(content='Hello! I can help you with questions related
to Fidelity mutual funds and retirement planning. How can I assist you today?',
additional_kwargs={}, response_metadata={'finish_reason': 'stop', 'model_name':
```

```
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'}, id='run-be384568-6100-46e4-850d-5b652abc8af3-0')]]}
'\n---\n'
```

```
[19]: query_agent("what are the names of mutual funds from Fidelity?")
```

```
---CALL AGENT---
"Output from node 'agent':"
'---'
{ 'messages': [ AIMessage(content='', additional_kwargs={'tool_calls':
[{'index': 0, 'id': 'call_Zv74q8rph0R88eunsSXHWxYH', 'function': {'arguments':
'{"query":"names of mutual funds from Fidelity"}', 'name':
'retrieve_fidelity_docs'}, 'type': 'function'}]]},
response_metadata={'finish_reason': 'tool_calls', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-2e92958f-7fd3-4eb1-b4d9-58aff9806275-0', tool_calls=[{'name':
'retrieve_fidelity_docs', 'args': {'query': 'names of mutual funds from
Fidelity'}, 'id': 'call_Zv74q8rph0R88eunsSXHWxYH', 'type': 'tool_call'}])]]}
'\n---\n'
---CHECK RELEVANCE---
---DECISION: DOCS NOT RELEVANT---
no
"Output from node 'retrieve':"
'---'
{ 'messages': [ ToolMessage(content='Fidelity® Magellan® Fund
(FMAGX)\n\nFidelity® New Millennium Fund (FMILX)\n\n\n\n\n\n\n\nLarge
Growth\n\nFidelity® Blue Chip Growth Fund (FBGRX)\n\nFidelity® Focused Stock
Fund (FTQGX)\n\nFidelity® Fund (FFIDX)\n\nFidelity® Growth Discovery Fund
(FDSVX)\n\nFidelity® Growth Company Fund (FDGRX)\n\nFidelity® Growth & Income
Portfolio Fund (FGRIX)\n\nFidelity® Dividend Growth Fund
(FDGFY)\n\n\n\n\n\n\n\nLarge Blend\n\nFidelity® Hedged Equity Fund
(FEQHX)\n\nFidelity®\xa0Sustainable U.S. Equity Fund (FSEBX)\n\nFidelity® US Low
Volatility Equity Fund (FULVX)\n\nFidelity®\xa0Sustainable U.S. Equity Fund
(FSEBX)\n\nFidelity® US Low Volatility Equity Fund (FULVX)\n\nFidelity®
Disciplined Equity Fund (FDEQX)\n\nFidelity® Large-Cap Stock Fund
(FLCSX)\n\nFidelity® Mega-Cap Stock Fund (FGRTX)\n\nSmall/Mid Blend\n\nFidelity®
Michigan Municipal Income Fund (FMHTX)\n\nFidelity® Minnesota Municipal Income
Fund (FIMIX)\n\nFidelity® New Jersey Municipal Income Fund (FNJHX)\n\nFidelity®
New York Municipal Income Fund (FTFMX)\n\nFidelity® Ohio Municipal Income Fund
(FOHFX)\n\nFidelity® Pennsylvania Municipal Income Fund
(FPXTX)\n\n\n\n\n\n\nResearch all Fidelity fixed income funds',
name='retrieve_fidelity_docs', id='3d38377c-18bd-4f27-b14d-c84f8ab98b0a',
tool_call_id='call_Zv74q8rph0R88eunsSXHWxYH')]]}
'\n---\n'
---TRANSFORM QUERY---
"Output from node 'rewrite':"
'---'
{ 'messages': [ AIMessage(content='What are some Fidelity mutual funds that I
```

```

can consider for my retirement planning?', additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_dbaca60df0'},
id='run-91aa9e44-e731-40bb-b813-8df3b362639d-0')]]}
'\n---\n'
---CALL AGENT---
"Output from node 'agent':"
'---'

{ 'messages': [ AIMessage(content="Here are some of the names of mutual funds
offered by Fidelity that you might consider:\n\n### Large Growth Funds\n-
**Fidelity® Magellan® Fund (FMAGX)**\n- **Fidelity® New Millennium Fund
(FMILX)**\n- **Fidelity® Blue Chip Growth Fund (FBGRX)**\n- **Fidelity® Focused
Stock Fund (FTQGX)**\n- **Fidelity® Fund (FFIDX)**\n- **Fidelity® Growth
Discovery Fund (FDSVX)**\n- **Fidelity® Growth Company Fund (FDGRX)**\n-
**Fidelity® Growth & Income Portfolio Fund (FGRIX)**\n- **Fidelity® Dividend
Growth Fund (FDGFX)**\n\n### Large Blend Funds\n- **Fidelity® Hedged Equity Fund
(FEQHX)**\n- **Fidelity® Sustainable U.S. Equity Fund (FSEBX)**\n- **Fidelity®
US Low Volatility Equity Fund (FULVX)**\n- **Fidelity® Disciplined Equity Fund
(FDEQX)**\n- **Fidelity® Large-Cap Stock Fund (FLCSX)**\n- **Fidelity® Mega-Cap
Stock Fund (FGRTX)**\n\n### Small/Mid Blend Funds\n- **Fidelity® Michigan
Municipal Income Fund (FMHTX)**\n- **Fidelity® Minnesota Municipal Income Fund
(FIMIX)**\n- **Fidelity® New Jersey Municipal Income Fund (FNJHX)**\n-
**Fidelity® New York Municipal Income Fund (FTFMX)**\n- **Fidelity® Ohio
Municipal Income Fund (FOHFX)**\n- **Fidelity® Pennsylvania Municipal Income
Fund (FPXTX)**\n\nThese funds cover a range of investment strategies and risk
profiles, so it's important to consider your individual retirement goals and
risk tolerance when selecting funds. If you need more specific information or
guidance on retirement planning, feel free to ask!", additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-1982b0e5-38a4-44ff-bde1-086dc59d8b59-0')]]}
'\n---\n'

```

```
[20]: query_agent("what are the names of mutual funds from Schwab?")
```

```

---CALL AGENT---
"Output from node 'agent':"
'---'

{ 'messages': [ AIMessage(content='Sorry this is out of scope. You could use
Google to search for this type of content/question.', additional_kwargs={},
response_metadata={'finish_reason': 'stop', 'model_name':
'gpt-4o-mini-2024-07-18', 'system_fingerprint': 'fp_129a36352a'},
id='run-90f85e32-4ff8-44af-b692-6fdc3c75a44f-0')]]}
'\n---\n'

```

```
[21]: # requirement 10

file_path = os.path.join('src', 'FFFGX.pdf')

# Load the PDF
loader = PyPDFLoader(file_path)

# Store pages
pages = []
async for page in loader.alazy_load():
    pages.append(page)

print(f"Loaded {len(pages)} pages successfully!")
```

Loaded 6 pages successfully!

```
[22]: print(f"{pages[1].metadata}\n")
print(pages[3].page_content)
```

```
{'producer': '', 'creator': 'Quadiant~Inspire~16.0.716.7', 'creationdate':
'2024-12-21T04:30:27+00:00', 'title': 'MFL_Doc', 'source': 'src/FFFGX.pdf',
'total_pages': 6, 'page': 1, 'page_label': '2'}
```

Allocation

Glossary Of Terms

Beta: A measure of a portfolio's sensitivity to market movements (as represented by a benchmark index). The benchmark index has a beta of 1.0. A beta of more (less) than 1.0 indicates that a fund's historical returns have fluctuated more (less) than the benchmark index. Beta is a more reliable measure of volatility when used in combination with a high R² which indicates a high correlation between the movements in a fund's returns and movements in a benchmark index.

Distribution and/or service fee(12b-1) Fees: The 12b-1 fee represents the maximum annual charge deducted from fund assets to pay for distribution and marketing costs. Total 12b-1 fees, excluding loads, are capped at 1.00% of average net assets annually. Of this, the distribution and marketing portion of the fee may account for up to 0.75%. The other portion of the overall 12b-1 fee, the service fee, may account for up to 0.25%.

Expense Ratio (Gross): Expense ratio is a measure of what it costs to operate an investment, expressed as a percentage of its assets, as a dollar amount, or in basis points. These are costs the investor pays through a reduction in the investment's rate of return. For a mutual fund, the gross expense ratio is the total annual fund or class operating expenses directly paid by the fund from the fund's most recent prospectus (before waivers or reimbursements). This ratio also includes Acquired Fund Fees and Expenses, which are expenses indirectly incurred by a fund through its ownership of shares in other investment

companies. If the investment option is not a mutual fund, the expense ratio may

be calculated using methodologies that differ from those used for mutual funds.

Expense Ratio (Net): Expense ratio is a measure of what it costs to operate an investment, expressed as a percentage of its assets, as a dollar amount, or in basis points. These are costs the investor pays through a reduction in the investment's rate of return. For a mutual fund, the net expense ratio is the total

annual fund or class operating expenses directly paid by the fund from the fund's most recent prospectus, after any fee waiver and/or expense reimbursements that will reduce any fund operating expenses. This ratio also includes Acquired Fund Fees and Expenses, which are expenses indirectly incurred by a fund through its ownership of shares in other investment companies. This number does not include any fee waiver arrangement or expense reimbursement that may be terminated without agreement of the fund's board of trustees during the one-year period. If the investment option is not a mutual fund, the expense ratio may be calculated using methodologies that differ from those used for mutual funds.

Fidelity Freedom 2045 Composite Index: Fidelity Freedom 2045 Composite Index is a customized blend of the following unmanaged indexes: Bloomberg Global Aggregate Treasury ex USD, ex Emerging Markets, RIC Capped, Float Adjusted Index (Hedged USD), Bloomberg U.S. 3-6 Month Treasury Bill Index, Bloomberg U.S. Long Treasury Bond Index, Bloomberg U.S. Aggregate Bond Index, Bloomberg U.S. Treasury Inflation-Protected Securities (TIPS) 0-5 Years Index, Bloomberg U.S. Treasury Inflation-Protected Securities (TIPS) 5+ Years Index, Dow Jones U.S. Total Stock Market Index, and MSCI All Country World ex U.S. Index (Net MA). The index weightings are adjusted monthly to reflect each fund's changing asset allocations. The compositions differed in periods prior to June 1, 2022.

Net Asset Value (NAV): The dollar value of one mutual fund's share, excluding any sales charges or redemption fees. The NAV is calculated by subtracting liabilities from the value of a fund's total assets and dividing it by the number of fund's shares outstanding.

Portfolio Net Assets (\$M): The difference between a portfolio's total assets and liabilities, including all share classes of the fund.

R2: A measurement of how closely the portfolio's performance correlates with the performance of the fund's primary benchmark index or equivalent. R 2 is a proportion which ranges between 0.00 and 1.00. An R2 of 1.00 indicates perfect correlation to the benchmark index, that is, all of the portfolio's fluctuations are explained by performance fluctuations of the index, while an R2 of 0.00 indicates no correlation. Therefore, the lower the R2, the more the fund's performance is affected by factors other than the market as measured by that benchmark index. An R 2 value of less than 0.5 indicates that the Annualized Alpha and Beta are not reliable performance statistics.

S&P 500 Index: S&P 500 Index is a market capitalization-weighted index of 500 common stocks chosen for market size, liquidity, and industry group representation to represent U.S. equity performance.

Share Class Net Assets (\$M): The difference between the total assets and liabilities of a single share class of a fund.

Sharpe Ratio: The Sharpe ratio is a measure of historical risk-adjusted performance. It is calculated by dividing the fund's excess returns (the fund's average annual return for the period minus the 3-month "risk free" return rate) and dividing it by the standard deviation of the fund's returns. The higher the ratio, the better the fund's return per unit of risk. The three month "risk free" rate used is the 90-day Treasury Bill rate.

Standard Deviation: Statistical measure of how much a return varies over an extended period of time. The more variable the returns, the larger the standard deviation. Investors may examine historical standard deviation in conjunction with historical returns to decide whether an investment's volatility would have been acceptable given the returns it would have produced. A higher standard deviation indicates a wider dispersion of past returns and thus greater historical volatility. Standard deviation does not indicate how an investment actually performed, but it does indicate the volatility of its returns over

time. Standard deviation is annualized. The returns used for this calculation are not load-adjusted.

Target-Date 2045: Target-date portfolios provide diversified exposure to stocks, bonds, and cash for those investors who have a specific date in mind (in this case, the years 2041-2045) for retirement. These portfolios aim to provide investors with an optimal level of return and risk, based solely on the target date. Management adjusts the allocation among asset classes to more-conservative mixes as the target date approaches, following a preset glide path. A target-date portfolio is part of a series of funds offering multiple retirement dates to investors.

Turnover Rate: The lesser of amounts of purchases or sales of long-term portfolio securities divided by the monthly average value of long-term securities

owned by the fund.

Important Information

0.

Before investing, consider the investment objectives, risks, charges and expenses of the fund or annuity and its investment options. Contact Fidelity for a free prospectus and, if available, summary prospectus containing this information. Read it carefully.

Page 4 of 6

```
[23]: vector_store = InMemoryVectorStore.from_documents(pages, OpenAIEmbeddings())
docs = vector_store.similarity_search("What is the fund manager?", k=2)
for doc in docs:
    print(f'Page {doc.metadata["page"]}: {doc.page_content[:1000]}\n')
```

Page 2: Allocation

Fund Overview (continued)

Additional Disclosures

This description is only intended to provide a brief overview of the mutual fund. Read the fund's prospectus for more detailed information about the fund.

Page 3 of 6

Page 3: Allocation

Glossary Of Terms

Beta: A measure of a portfolio's sensitivity to market movements (as represented by a benchmark index). The benchmark index has a beta of 1.0. A beta of more (less) than 1.0 indicates that a fund's historical returns have fluctuated more (less) than the benchmark index. Beta is a more reliable measure of volatility when used in combination with a high R2 which indicates a high correlation between the movements in a fund's returns and movements in a benchmark index.

Distribution and/or service fee(12b-1) Fees: The 12b-1 fee represents the maximum annual charge deducted from fund assets to pay for distribution and marketing costs. Total 12b-1 fees, excluding loads, are capped at 1.00% of average net assets annually. Of this, the distribution and marketing portion of the fee may account for up to 0.75%. The other portion of the overall 12b-1 fee, the service fee, may account for up to 0.25%.

Expense Ratio (Gross): Expense ratio is a measure of what it cos

```
[24]: import base64
import io

import fitz
from PIL import Image

def pdf_page_to_base64(pdf_path: str, page_number: int):
    pdf_document = fitz.open(pdf_path)
    page = pdf_document.load_page(page_number - 1) # input is one-indexed
    pix = page.get_pixmap()
    img = Image.frombytes("RGB", [pix.width, pix.height], pix.samples)

    buffer = io.BytesIO()
    img.save(buffer, format="PNG")

    return base64.b64encode(buffer.getvalue()).decode("utf-8")

[25]: from IPython.display import Image as IImage
base64_image = pdf_page_to_base64(file_path,1)
display(IImage(data=base64.b64decode(base64_image)))
```



Allocation

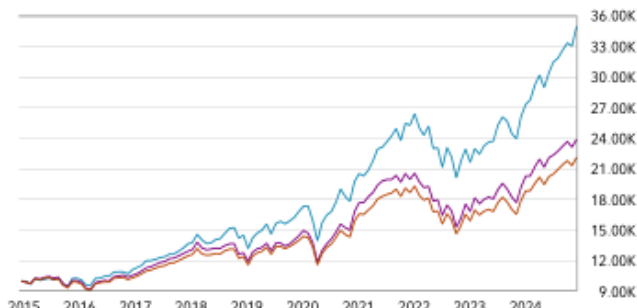
Fidelity Freedom® 2045 Fund (FFFGX)

NTF No Transaction Fee*

Hypothetical Growth of \$10,000^{6,7}

AS OF 11/30/2024 ; Target-Date 2045

● FFFGX : \$23,898 ● S&P 500 Index : \$35,002 ● Target-Date 2045 : \$22,139



Performance^{5,6,8,10}

AS OF 11/30/2024

Monthly	YTD (Monthly)	Average Annual Total Returns				
		1 Yr	3 Yrs	5 Yrs	10 Yrs	Life
Fidelity Freedom® 2045 Fund	17.79%	24.39%	6.15%	10.59%	9.10%	7.37%
S&P 500	28.07%	33.89%	11.44%	15.77%	13.35%	10.86%
FID FF 2045 Comp Idx	17.54%	23.78%	6.13%	10.01%	9.08%	8.02%
Target-Date 2045	17.45%	23.56%	5.99%	9.76%	8.42%	--
Rank in Morningstar Category		27%	40%	12%	13%	--
# of Funds in Morningstar Category		199	188	164	109	--
Quarter-End (AS OF 09/30/2024)						
Fidelity Freedom® 2045 Fund		29.83%	6.34%	11.66%	9.35%	7.39%

Calendar Year Returns^{5,6,8,10}

AS OF 11/30/2024

	2020	2021	2022	2023	2024
Fidelity Freedom® 2045 Fund	18.15%	16.45%	-18.26%	20.56%	17.79%
S&P 500	18.40%	28.71%	-18.11%	26.29%	28.07%
FID FF 2045 Comp Idx	16.54%	16.18%	-18.17%	20.11%	17.54%
Target-Date 2045	15.10%	16.63%	-17.75%	18.95%	17.45%

Morningstar® Snapshot^{*11}

AS OF 11/30/2024

Morningstar Category	Target-Date 2045
Risk of this Category	Lower Higher
Overall Rating	★★★★★ Out of 188 funds
Returns	Low Avg High
Expenses	Low Avg High

*Data provided by Morningstar

Details

Morningstar Category	Target-Date 2045
Fund Inception	06/01/2006
NAV 12/19/2024	\$13.73
Exp Ratio (Gross) 05/30/2024	0.75%
Exp Ratio (Net) 05/30/2024	0.75%
Minimum to Invest	\$0.00
Turnover Rate 09/30/2024	13.00%
Portfolio Net Assets (\$M) 11/30/2024	\$21,466.13
Share Class Net Assets (\$M) 11/30/2024	\$4,597.27

Equity StyleMap^{*3}

AS OF 10/31/2024

■ Historical ● Current



Large Blend

*91.96% Fund Assets Covered

Fund Manager(s)

Co-Manager : Andrew J. Dierdorf (since 06/21/2011)
Co-Manager : Brett F. Sumsion (since 01/21/2014)

```
[26]: # Requirement 11

# Grab a chat client to test the PDF context
llm = ChatOpenAI(model=model_name)

# lets set up a function to query this LLM in a DRY fashion
def query_llm(query):
    message = HumanMessage(
        content=[
            {"type": "text", "text": query},
            {
                "type": "image_url",
                "image_url": {"url": f"data:image/jpeg;base64,{base64_image}"},
            },
        ],
    )
    response = llm.invoke([message])
    print(response.content)
```

```
[27]: query_llm("What is the name of this fund?")
```

The name of the fund is the **Fidelity Freedom® 2045 Fund (FFFGX)**.

```
[28]: query_llm("Who is the fund manager?")
```

The fund managers for the Fidelity Freedom® 2045 Fund (FFFGX) are Andrew J. Dieford (Co-Manager since June 21, 2011) and Brett F. Sumners (Co-Manager).

```
[29]: query_llm("What is the calendar year return for 2022 for this fund and S&P 500")
```

The calendar year return for 2022 is as follows:

- **Fidelity Freedom® 2045 Fund (FFFGX)**: -18.26%
- **S&P 500**: -18.11%

```
[30]: query_llm("What is the Portfolio Net Assets")
```

The Portfolio Net Assets of the Fidelity Freedom® 2045 Fund (FFFGX) as of 11/30/2024 are \$21,463,148, and it has Share Class Net Assets of \$4,597,927 as of the same date.