Use LangGraph with Radium

LangGraph does not connect to a model provider by itself. Configure a Radium-backed `ChatOpenAI` model and call it from your graph nodes.

Before you start

You need Python 3.10 or newer, a Radium API key, and a terminal or PowerShell window.

1. Create a project and install the packages

macOS or Linux:

```
mkdir radium-langgraph
cd radium-langgraph
python3 -m venv .venv
source .venv/bin/activate
python -m pip install langgraph langchain-openai
```

Windows PowerShell:

```
mkdir radium-langgraph
cd radium-langgraph
py -m venv .venv
.venv\Scripts\Activate.ps1
python -m pip install langgraph langchain-openai
```

2. Set your Radium credentials

macOS or Linux:

```
export RADIUM_API_KEY="your-radium-api-key"
export RADIUM_MODEL="hal-1.0"
```

Windows PowerShell:

```
$env:RADIUM_API_KEY = "your-radium-api-key"
$env:RADIUM_MODEL = "hal-1.0"
```

3. Create `main.py`

```
import os

from langchain_openai import ChatOpenAI
from langgraph.graph import END, START, MessagesState, StateGraph



api_key = os.getenv("RADIUM_API_KEY")
if not api_key:
    raise SystemExit("Set RADIUM_API_KEY before running this program.")

llm = ChatOpenAI(
    model=os.getenv("RADIUM_MODEL", "hal-1.0"),
    api_key=api_key,
    base_url="https://api.radium.cloud/v1",
    max_tokens=512,
    temperature=0,
    timeout=90,
    max_retries=0,
)



def call_radium(state: MessagesState):
    return {"messages": [llm.invoke(state["messages"])]}



builder = StateGraph(MessagesState)
builder.add_node("call_radium", call_radium)
builder.add_edge(START, "call_radium")
builder.add_edge("call_radium", END)
graph = builder.compile()

result = graph.invoke(
```

```
    {
        "messages": [
            {"role": "system", "content": "Follow the user's instruction exactly."},
            {"role": "user", "content": "Reply with exactly: Radium connected."},
        ]
    }
)

content = result["messages"][-1].content
if not content or not str(content).strip():
    raise RuntimeError("Radium returned no visible text.")
print(f"RADIUM_RESPONSE: {str(content).strip()}")
```

4. Run it

```
python main.py
```

Expected output:

```
RADIUM_RESPONSE: Radium connected.
```

Choose a model

```
export RADIUM_MODEL="tycho-1.0"  # or hal-1.0 or clarke-1.0
python main.py
```

Migrate an existing graph

Replace the model used inside your existing nodes:

```
llm = ChatOpenAI(
    model="hal-1.0",
    api_key=os.environ["RADIUM_API_KEY"],
    base_url="https://api.radium.cloud/v1",
    max_tokens=512,
)
```

Your state schema, nodes, edges, routing, and checkpoints do not need to change.

Troubleshooting

- Authentication errors: check the API key in the active terminal.
- `404`: use the exact base URL and one of the three listed model IDs.
- State errors: make sure the graph node returns a `messages` list.
- Import errors: activate `.venv` and rerun the installation command.

Validation

Version note: these instructions were verified with Python 3.12.7, `langgraph==1.2.11`, and `langchain-openai==1.6.0` on August 20, 2026. You do not need those exact package versions. Text generation and an agent tool loop passed with `hal-1.0`, `clarke-1.0`, and `tycho-1.0`.

Reference: LangGraph quick start.

Next

- API quickstart, for calling Radium directly
- Tool calling, for the `tool_use` and `tool_result` contract

Created 2026-08-23 21:36:18 UTC by Admin
Updated 2026-08-23 21:53:39 UTC by Admin