

= An introduction to Langchain
:order: 1
:type: lesson

This lesson will teach you about Langchain, an open-source framework for building AI applications.

== What is Langchain?

link:<https://langchain.com>[LangChain^] is designed to accelerate the development of LLM applications.

Langchain enables software developers to build LLM applications quickly, integrating with external components and data sources.

Developers can build Langchain applications with
link:<https://python.langchain.com/>[Python^] or
link:<https://js.langchain.com/>[JavaScript/TypeScript^].

Langchain supports multiple LLMs and allows you to swap one for another with a single parameter change.

Meaning you can quickly test multiple LLMs for suitability and utilize different models for different use cases.

Langchain provides out-of-the-box integrations with APIs and databases including
link:<https://python.langchain.com/docs/integrations/providers/neo4j>[Neo4j^].

Langchain's flexibility allows you to test different LLM providers and models with minimal code changes.

== How does it work?

LangChain applications bridge users and LLMs, communicating back and forth with the LLM through **Chains**.

The key components of a Langchain application are:

* **Model Interaction (Model I/O)**: Components that manage the interaction with the language model, overseeing tasks like feeding inputs and extracting outputs.

* **Data Connection and Retrieval**: Retrieval components can access, transform, and store data, allowing for efficient queries and retrieval.

* **Chains**: Chains are reusable components that determine the best way to fulfill an instruction based on a `_prompt_`.

* **Agents**: Agents orchestrate commands directed at LLMs and other tools, enabling them to perform specific tasks or solve designated problems.

* **Memory:** Allow applications to retain context, for example, remembering the previous messages in a conversation.

[%collapsible]

.Click to reveal an example Langchain application

====

This example program uses Langchain to build a chatbot that answers questions about Neo4j Cypher queries.

The program interacts with an OpenAI LLM, uses a prompt template to instruct the LLM on `_how to act_`, and uses a memory component to retain context and store the history in Neo4j.

After completing this module, you will understand what this program does and how it works.

[source, python]

```
include::code/example_application.py[ ] [ ]
```

====

In the next lesson, you will set up your development environment and use Langchain to query an LLM.

== Check Your Understanding

```
include::questions/1-languages.adoc[leveloffset=+1]
```

[.summary]

== Lesson Summary

In this lesson, you learned about Langchain, an open-source framework for building AI applications.

In the next lesson, you will learn how to set up your development environment and use Langchain to query an LLM.