NVIDIA Agent Intelligence Toolkit Overview

NVIDIA Agent Intelligence (AIQ) toolkit is a flexible, lightweight, and unifying library that allows you to easily connect existing enterprise agents to data sources and tools across any framework.

Note

Agent Intelligence toolkit was previously known as AgentIQ, however the API has not changed and is fully compatible with previous releases. Users should update their dependencies to depend on `aiqtoolkit` instead of `agentiq`. The transitional package named `agentiq` is available for backwards compatibility, but will be removed in the future.

Key Features

- **Framework Agnostic:** AIQ toolkit works side-by-side and around existing agentic frameworks, such as LangChain, LlamaIndex, CrewAI, and Microsoft Semantic Kernel, as well as customer enterprise frameworks and simple Python agents. This allows you to use your current technology stack without replatforming. AIQ toolkit complements any existing agentic framework or memory tool you're using and isn't tied to any specific agentic framework, long-term memory, or data source.
- **Reusability:** Every agent, tool, and agentic workflow in this library exists as a function call that works together in complex software applications. The composability between these agents, tools, and workflows allows you to build once and reuse in different scenarios.
- **Rapid Development:** Start with a pre-built agent, tool, or workflow, and customize it to your needs. This allows you and your development teams to move quickly if you're already developing with agents.
- **Profiling:** Use the profiler to profile entire workflows down to the tool and agent level, track input/output tokens and timings, and identify bottlenecks.
- **Observability:** Monitor and debug your workflows with any OpenTelemetry-compatible observability tool, with examples using Phoenix and W&B Weave.
- **Evaluation System:** Validate and maintain accuracy of agentic workflows with built-in evaluation tools.
- **User Interface:** Use the AIQ toolkit UI chat interface to interact with your agents, visualize output, and debug workflows.
- **Full MCP Support:** Compatible with Model Context Protocol (MCP). You can use AIQ toolkit as an MCP client to connect to and use tools served by remote MCP servers. You can also use AIQ toolkit as an MCP server to publish tools via MCP.

Installing NVIDIA Agent Intelligence Toolkit

This guide will help you set up your NVIDIA Agent Intelligence (AIQ) toolkit development environment, run existing workflows, and create your own custom workflows using the `aiq` command-line interface.

Supported LLM APIs:

- NIM (such as Llama-3.1-70b-instruct and Llama-3.3-70b-instruct)
- OpenAI

Framework Integrations

To keep the library lightweight, many of the first party plugins supported by AIQ toolkit are located in separate distribution packages. For example, the `aiqtoolkit-langchain` distribution contains all the LangChain specific plugins and the `aiqtoolkit-mem0ai` distribution contains the Mem0 specific plugins.

To install these first-party plugin libraries, you can use the full distribution name (for example, `aiqtoolkit-langchain`) or use the `aiqtoolkit[langchain]` extra distribution. A full list of the supported extras is listed below:

- `aiqtoolkit[agno]` or `aiqtoolkit-agno` - Agno specific plugins
- `aiqtoolkit[crewai]` or `aiqtoolkit-crewai` - CrewAI specific plugins
- `aiqtoolkit[langchain]` or `aiqtoolkit-langchain` - LangChain specific plugins
- `aiqtoolkit[llama-index]` or `aiqtoolkit-llama-index` - LlamaIndex specific plugins
- `aiqtoolkit[mem0ai]` or `aiqtoolkit-mem0ai` - Mem0 specific plugins
- `aiqtoolkit[semantic-kernel]` or `aiqtoolkit-semantic-kernel` - Microsoft Semantic Kernel specific plugins
- `aiqtoolkit[test]` or `aiqtoolkit-test` - AIQ toolkit Test specific plugins
- `aiqtoolkit[weave]` or `aiqtoolkit-weave` - Weights & Biases Weave specific plugins
- `aiqtoolkit[zep-cloud]` or `aiqtoolkit-zep-cloud` - Zep specific plugins

Prerequisites

NVIDIA Agent Intelligence (AIQ) toolkit is a Python library that doesn’t require a GPU to run the workflow by default. You can deploy the core workflows using one of the following:

- Ubuntu or other Linux distributions, including WSL, in a Python virtual environment.

Before you begin using AIQ toolkit, ensure that you meet the following software prerequisites.

- Install Git
- Install Git Large File Storage (LFS)
- Install uv

Install From Source

1. Clone the AIQ toolkit repository to your local machine.

```
2. git clone git@github.com:NVIDIA/AIQToolkit.git aiqtoolkit
3. cd aiqtoolkit
```

4. Initialize, fetch, and update submodules in the Git repository.

```
5. git submodule update --init --recursive
```

6. Fetch the data sets by downloading the LFS files.

```
7. git lfs install
8. git lfs fetch
9. git lfs pull
```

10. Create a Python environment.

```
11.  uv venv --seed .venv
12.  source .venv/bin/activate
```

13. Install the AIQ toolkit library. To install the AIQ toolkit library along with all of the optional dependencies. Including developer tools (`--all-groups`) and all of the dependencies needed for profiling and plugins (`--all-extras`) in the source repository, run the following:

```
14.  uv sync --all-groups --all-extras
```

Alternatively to install just the core AIQ toolkit without any plugins, run the following:

```
uv sync
```

At this point individual plugins, which are located under the `packages` directory, can be installed with the following command `uv pip install -e '.[<plugin_name>]'`. For example, to install the `langchain` plugin, run the following:

```
uv pip install -e '.[langchain]'
```

Note

Many of the example workflows require plugins, and following the documented steps in one of these examples will in turn install the necessary plugins. For example following the steps in the `examples/simple/README.md` guide will install the `aiqtoolkit-langchain` plugin if you haven't already done so.

In addition to plugins, there are optional dependencies needed for profiling. To install these dependencies, run the following:

```
uv pip install -e .[profiling]
```

15. Verify that you’ve installed the AIQ toolkit library.

```
16. aiq --help
17. aiq --version
```

If the installation succeeded, the `aiq` command will log the help message and its current version.

Obtaining API Keys

Depending on which workflows you are running, you may need to obtain API keys from the respective services. Most AIQ toolkit workflows require an NVIDIA API key defined with the `NVIDIA_API_KEY` environment variable. An API key can be obtained by visiting **`build.nvidia.com`** and creating an account.

Running Example Workflows

Before running any of the AIQ toolkit examples, set your NVIDIA API key as an environment variable to access NVIDIA AI services.

```
export NVIDIA_API_KEY=<YOUR_API_KEY>
```

Note

Replace `<YOUR_API_KEY>` with your actual NVIDIA API key.

Running the Simple Workflow

1. Install the `aiq_simple` Workflow

```
2. uv pip install -e examples/simple
```

3. Run the `aiq_simple` Workflow

```
4. aiq run --config_file=examples/simple/configs/config.yml --input "What is LangSmith"
```

5. **Run and evaluate the `aiq_simple` Workflow**

   The `eval_config.yml` YAML is a super-set of the `config.yml` containing additional fields for evaluation. To evaluate the `aiq_simple` workflow, run the following command:

```
aiq eval --config_file=examples/simple/configs/eval_config.yml
```

AIQ Toolkit Packages

Once an AIQ toolkit workflow is ready for deployment to production, the deployed workflow will need to declare a dependency on the `aiqtoolkit` package, along with the needed plugins. When declaring a dependency on AIQ toolkit it is recommended to use the first two digits of the version number. For example if the version is `1.0.0` then the dependency would be `1.0`.

For more information on the available plugins, refer to Framework Integrations.

Example dependency for AIQ toolkit using the `langchain` plugin for projects using a `pyproject.toml` file:

```
dependencies = [
"aiqtoolkit[langchain]~=1.0",
# Add any additional dependencies your workflow needs
]
```

Alternately for projects using a `requirements.txt` file:

```
aiqtoolkit[langchain]==1.0.*
```

Next Steps

- AIQ toolkit contains several examples which demonstrate how AIQ toolkit can be used to build custom workflows and tools. These examples are located in the `examples` directory of the AIQ toolkit repository.
- Refer to the AIQ toolkit tutorials for more detailed information on how to use AIQ toolkit.