**Architectural and Process Overview**:

The most basic order of operations of a middleware is to:

1. Connect to the source platform
2. Retrieve data from the source platform
3. Process the data from the source platform *for the destination platform*
4. Connect to the destination platform
5. Ingest the data onto the destination platform

For visualization, please see below:



Given this, the middleware requires *at least* 4 base classes:

1. Connector

* Used to **connect** to a platform, initializes the middleware as the **client** of the platform

1. Retriever

* Used to **retrieve** data from a platform, ideally begins with the GET request and ends with the middleware having acquired raw data from said platform

1. Processor

* Used to **process** the raw data. This is where transformation occurs from the raw data, to the data that will be **properly** **ingested** by the destination

1. Ingester

* Used to **ingest** the data onto the destination platform. Ideally starts and ends with the POST request

*Note: It is possible that more classes are needed outside of these 4.*

**Scope and Limitations**:

This project is intended only for HTTP/S-based middleware.

**Tool / Framework Preference:**

Python-based frameworks such as Flask / Django. (Recommended: Flask)

**PoC**

For an eventual proof of concept, access to 2 platforms will be given, namely A mock Helix server and TheHive / ELK , where:

The mock Helix server will be the source platform and TheHive/ELK will be the destination platform. Outlined below are milestones expected to be achieved:

| Connector, Retriever, Processor, and Ingester **base classes** developed |  |
| --- | --- |
| Middleware successfully connects to and retrieves data from Helix, using classes that inherit the base Connector and Retriever classes |  |
| Middleware successfully processes the data from Helix into data for ingestion into the TheHive, using a class that inherits the base Processor class |  |
| Middleware successfully ingests data onto TheHive/ELK, using a class that inherits the base Ingester class |  |
| Recommended process for adding new integrations between two platforms are presented and documented |  |