Test Svc

The project utilizes the following libraries for logging, command-line argument parsing, XML and JSON manipulation, and network operations:

- spdlog: Used for logging. spdlog provides fast and convenient logging capabilities. It is a header-only library, making it easy to integrate into the project.
- CLI11: Used for parsing command-line arguments. CLI11 is a header-only library that simplifies the process of defining and parsing command-line options.
- RapidXML: Used for handling XML data. RapidXML is a header-only library known for its speed and ease of use in parsing XML documents.
- nlohmann/json: Used for working with JSON data. nlohmann/json is a header-only library that provides a simple interface for parsing and generating JSON data.
- libcurl: Used for making HTTP requests. libcurl is a robust library for transferring data with URLs. While it is not header-only, a static build of libcurl has been used in the project for its ease of integration and use. These libraries were chosen for their speed, convenience, and user-friendly licenses, which allow for easy integration into the project.

Brief overview

The ProgramConfiguration class manages the service's configuration for both SCM (Service Control Manager) and console operations. It can be configured either through the console or by using configuration file(s). The class provides methods for setting and getting configuration values, as well as a function to validate configuration files in memory.

The SvcMain function is the main entry point for the service, called by the SCM through the main function. It handles the service's startup and initializes its own logger. This function then starts the SvcMainThread.

The SvcMainThread function manages the service's execution process. It includes the main loop, which checks if the service should stop, and it also starts the WorkThread function on a separate thread. It controls the WorkThread by determining if it should stop and signals it if necessary.

The WorkThread function is started on a separate thread and performs all the intended work of the service. It contains a main loop synchronized with a fetch rate. In each iteration, it checks for new configuration from files, makes URL

requests to the NBU API, retrieves the data, and calls the SaveTable function to perform the intended work.