COAT (Certificate of analysis toolkit) Admin Guide

ProteinSimple Toronto Operation department
Author: Arvin Asgharian Rezaee

August 12, 2025

Contents

1 Introduction

COAT was first developed during summer of 2025 for automating and streamlining the process of COA creation of Maurice cartridges for the Toronto office. the purpose of this guide it to briefly touch on the software architecture of the project, alongside guides to maintain and update it.

2 Getting Started

We will first lightly touch opon compiling and deploying a project and releasing a new version.

2.1 System Requirements

At the time of creation of this document, the only supported OS for COAT is Windows 11 and Windows 10. The released executable file only supports the mentioned Systems. Though the source code can be altered to support other systems but no guarantee is currently. It is assumed that the following software is installed and working correctly on your machine.

- Git
- Winget

2.2 Setup

The source code for the project can be cloned with the following links:
https://github.com/ProteinSimple/CoA-Automation
To clone the project use the following command in your terminal

```
cd ~
mkdir temp
cd temp
git clone https://github.com/ProteinSimple/CoA-Automation.git .
```

The steps to install the program can be found in the GitHub webpage. Follow the steps to install the requirements and run the project.

3 Action Guide

The automation script that is the backend logic of COAT, is written in python and has a CLI where you can provide the application with actions you would like it to perform. This is in fact the exact method that the UI communicates with this script. The following is a list of possible actions alongside a brief and non-technical explaination. The python file containing the entry point to the program is listed below:

```
<project path>/automation/src/main.py
```

To get comprehensive list of actions run the following

```
> cd <project path>/automation/src
> python main.py --help
```

To get a description of an action alongside how to use it run it with "action-name –help" so as an example:

```
> python main.py check --help
```

Before diving into specifics of each action, there are several common argument that is shared between all of the actions. Lets first have an overview of those:

- -h, –help: Show help comments
- –run-mode: Set run mode of program (test or prod, default = prod)
- –verbose: By default the script outputs its results to the given output file. with this option it will print it to stdout instead (default = False)
- —config: Path to the configuration file containing run information (default = ./config.yaml)
- –output: Output file used to showcase the result of the program
- –user: username for saturn authentication
- –passkey: passkey for saturn authentication

3.1 Check Action

Used to check if connection to saturn is valid. Method of running is :

```
  python main.py check
```