Workflow for updating Enrolment Report

Lambton College

Last update April, 2023.

1. Required files. Once the zip file is unzipped (regardless of directory), you should have the run\_report.py, requirements.txt and the utils folder. It might look something like this:

Graphical user interface, text, application

Description automatically generated

Within the utils folder, you should see this:

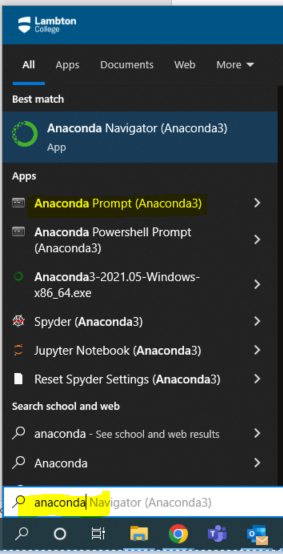
Graphical user interface, text, application, email

Description automatically generated

Logic behind the files:

* run\_report.py is the main file containing the big steps to compile the report
* utils contains all the intermediate functions to compile the report
* Think of the files within the utils folder as helper functions. This structure is preferred as is easier to update and maintain.
* requirements.txt contains the required python libraries (remember the analogy with my aunt’s desert. This file gets us ‘the whole kitchen’ so we could have a replica of my aunt’s hardware).

1. Once the files are in place, launch the anaconda prompt.

Graphical user interface

Description automatically generated with medium confidence

Logic behind the prompt:

* We live in a Windows (the operating system) world. So, we are pretty used to interact with the machine in a very comfortable and intuitive way. This might be a little different and intuitive but what’s required for this project, is pretty straight.
* The idea of using prompt comes from the Linux word (another operating system preferred by geek and nerd developers). So, for this project, you will interact with the machine through explicitly saying what you want it to do trying to mimic Linux ideas. Note: This terminal is not Linux by itself, just mimics it.
* Anaconda would create the virtual environment we need to run this project. Its main purpose is to create virtual environments. By default, it will create an empty virtual environment named **base** (the word between parenthesis in the prompt).
* Anaconda will execute what we want it to do in the backend based on what we asked in this specific virtual environment.

1. [THIS STEP YOU WOULD ONLY HAVE TO DO IT ONCE, THE FIRST TIME]

As suggested by my aunt’s analogy, you will need to set your ‘kitchen’ with the required hardware. This process would be typically only required once. First, navigate through the location on your folder explorer and get the location path. for example:

Graphical user interface, text, application, chat or text message

Description automatically generated

Once you have this string of text, copy it and write **cd \*paste path (crtl+v)\*** on the prompt.

Graphical user interface, text

Description automatically generated

Once you hit enter, it’d look like this:

Text

Description automatically generated

Write down **pip install -r requirements.txt --user**

Text

Description automatically generated

It might take a couple of minutes. You’ll see many progress bars and messages about the libraries installations. If for some reason it doesn’t work, trying deleting the **--user** argument.

Text

Description automatically generated

You might see yellow messages (warnings), they should be ok if the process ends. In case you see a red message (error), please reach out. In case it compiles well, Congratulations! Now you have set the working space for the report to run.

Logic behind the installation:

* This step is solely to install the required libraries. We’ve done that asking, via prompt, the machine to read the requirements file and install all of them in the environment.
* Pip is the default package manager included in Anaconda. So, any time a library is required, pip should be the best way of doing it.
* In the prompt, when a command is preceded by a single dash (-) is named a ‘flag’. The -r flag is used to tell the prompt we want it to read a file (the txt file)
* When a command preceded by double dash (--) is named an ‘argument’. The --user argument is to tell the prompt we are the main users and that it should be ok to install the required packages. Is important to use it in the college machines as these would automatically reject any sort of installation.

1. Once the virtual environment is set, you are ready to run the full report. To do so, for the fall intake, run the line

**python run\_report.py --intake fall** in theprompt

Text

Description automatically generated

Once you hit enter, the prompt would start executing and providing updates after each section of it. After a couple of minutes, it should be done.

Text

Description automatically generated

Logic behind the compilation process:

* As your default virtual environment, named **base**, has everything it needs, the compilation should run smooth.
* This is an idea coming directly from Linux as the prompt requires to know which language should be used to compile the code, that’s why **python** is the first word.
* After the compiler is set, **run\_report.py** contains the list of steps do be performed (as a recipe).
* The argument **–intake** expects you to input the intake for the report: fall, winter or summer. This adds flexibility to the report.
* As of April 2023, updating the resulting file into sharepoint will to have to be done by hand as Sharepoint API seems to be not working properly as code the broke.

1. If the process ran correctly, several things should have happened.

* Folders Logs and reports were created

Graphical user interface, application, Word

Description automatically generated

* Within reports, excel files, one whose name is based on the date and time of report generation and one named **DasboardInput.xlsx,** were to be created.

Text

Description automatically generated

1. Finally, you would drag the **DasboardInput.xlsx**  file and drop it into the shared [Sharepoint folder](https://mylambton.sharepoint.com/sites/EnrolmentDashboard/Shared%20Documents/Forms/AllItems.aspx). If sharepoint asks, you would replace the file. And that’s it!

Graphical user interface, application

Description automatically generated

Logic behind the compilation process:

* The final dashboard is set to update automatically. It will check changes in the sharepoint repository every two hours. When a change is found, it will update the data model, the visuals and numbers.
* Note that the whole process will create a copy of the report in your machine whose name is based on date and time of last run. This one is intended to be our tracking version, so we know how numbers progress through time (doing something with this might an idea for further enhancements for this project).