**Contents of project package:**

The project package has 3 main .py programming scripts:

* **CovidReport.py** : Python file for creating the main CovidReport class and include main function to run the developed system.
* **CovRepTest.py** : Python file which includes all the unit tests performed to ensure the accuracy of designed system.
* **ConfigWriter.py** : Python file to create and initialize a Config file in the same package.
* **ReadMe.docx** : Text file containing necessary information about the program package.
* **Config.ini** : Initialized config file, that will be rewritten in case of changing ConfigWriter.py.
* **ISO\_DATE.xlsx** : Excel file containing date&iso pair to perform unit and integration testing.
* **Output.xlsx** : sample of generated output of the system for ISO\_DATE.xlsx.

**How to run the package:**

Go to the ConfigWriter.py file and change the url location to the local url address of Excel file. Note that do not include the name of Excel file as it will be asked further forward as a separate step. Hence, only place the url address of folder containing Excel file in the “ExcelFile” attribute of config\_object in the ConfigWriter.py file. After that, Run it to create the related config file.

Go to the CovidReport.py file and run it. It asks for the config file, which is set to config.ini, so you can enter config.ini, or in the case if you have changed it, enter the new name. Then it asks for the name of Excel file, requiring you to enter the name of Excel file containing date-iso pair. After that, it automatically calls class function, run, which is chained to other instance methods of CovidReport class to initialize a class object, read config file and then read the excel file into a pandas data frame object. Then it used each pair of date&iso to make API call to “https://covid-api.com/api/reports/“, reads the JSON data and extract the proper data based on that date&iso. It then append each extracted accumulated information as a row of a dataframe, named output\_table, which is used to produce output. Finally, this created data frame is saved in a local Excel file in the same folder, which in here is named Output.xlsx.

To see the performed unit tests, go to CovRepTest.py file. In here the unittest library has been utilized to perform unit testing on some corners and different cases of designed program to ensure its correctness and accuracy.

**Note:**

1. To run the main program, please make sure all the necessary packages such as openpyxl or pandas are installed. In case of any error regarding not finding a specific library, please pip instal that library.
2. In the design of the program, since in the specification it has been specifically mentioned that for each pair of date&iso a separate API call is needed, a separate Url API call and getting JSON object is designed, even though it is much more time consuming and has made the speed of the designed system slower than it could be. As a better alternative, API call could have been done only once and resulted JSON object could have been saved as a dictionary and after that the extracted information for each date&iso pair can be grabbed from this saved dictionary rather than separate url API calls.
3. Moreover, runtime for CovRepTest.py is long compared to number of test cases (120s in my personal system). It is mainly due to 2 reasons. First, the reason explained in the above paragraph and second is due to creation of a new CovidReport object and performing its functions separately for each test case, rather than constructing only one object and updating it. The reason to have this strategy is to make sure in case of a failure in one test case, others work properly. Otherwise, only one object could be constructed and its output would have been taken for test cases and in that case its run time would have decreased drastically.