**TECHNICAL DETAILS FOR GUI**

We wanted to create an executable file which can be run from desktop itself. For which we used TKinter for the User Interface and Pyinstaller for creating a Windows executable file.

**Important widgets used in TKinter:**

* **TK():** For creating a window.
* **Label ():** For creating texts on window.



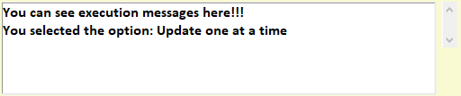
* **Entry():** For creating text fields to get input text from user.



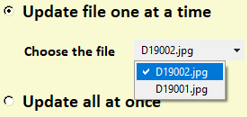
* **Scrollbar:** For creating a scrollbar alongside message box. Here, we used tk.ttk () for better styling.



* **Listbox:** For creating a message box which will display appropriate messages at the time of execution. It is linked to the Scrollbar widget.



* **Radiobuttons:** Creating Two Radiobuttons, one for Updating one at a time and another for Updating all at once.



* **OptionMenu:** This is used to list all the files in the specified path. Used tk.ttk() for better styling of the list.



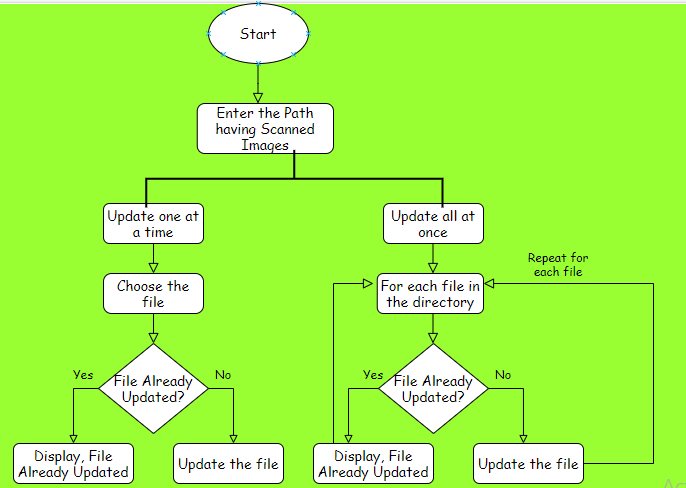
* **Button:** It is used to create the *UPDATE* button.



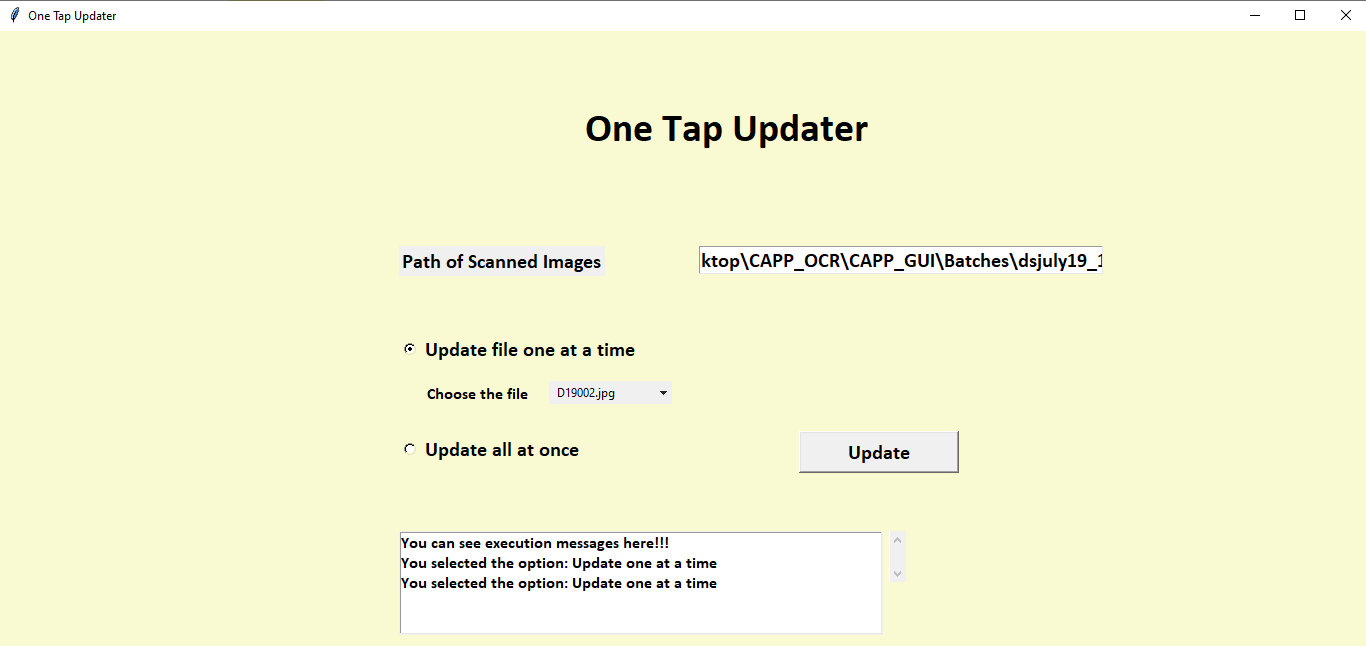
In the GUI we have used ***place ()*** function for setting the geometry of the widgets.

Font used in the GUI is ***Calibri*** with background colour as ***light goldenrod yellow***.

**Technical Flow Diagram:**



**THE GUI**



**Different Modules used in the GUI**

|  |  |
| --- | --- |
| **Module Name** | **Purpose** |
| getdata () | It will give the final predicted labels. |
| inner\_box () | It will extract the boxes from the input image. |
| preprocess () | It will preprocess the extracted boxes. |
| validrects () | It will validate if the extracted boxes are the desired boxes. |
| inference () | It is used to predict the extracted characters from the input image. |
| sel () | Command used for Radiobutton *update one at a time*. |
| sel2() | Command used for Radiobutton *update all at once*. |
| update() | Command used for *Update* Button. |

Next step is converting the python script into .exe file.

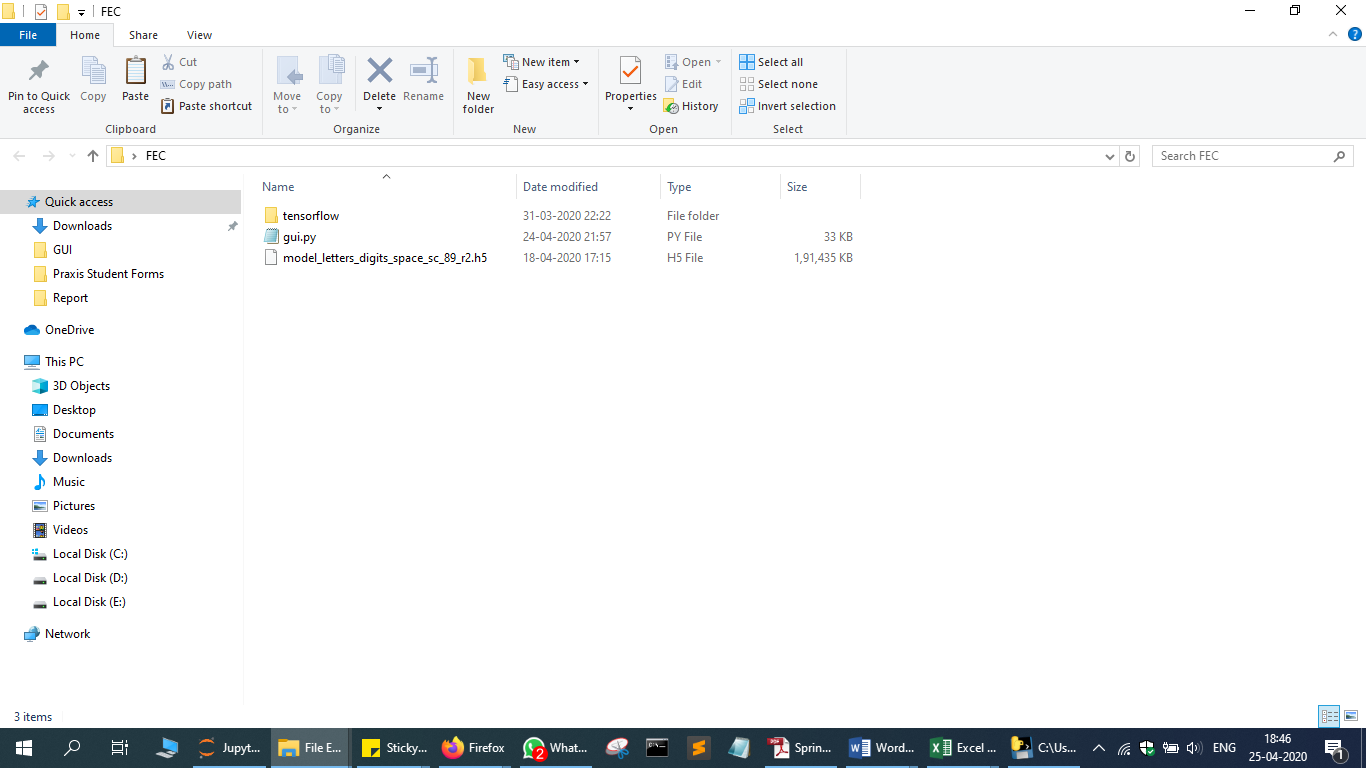
These are the following steps for creating a .exe file from the python script:

1. Install **pyinstaller** using below command

pip install pyinstaller

1. Change all images or models load from system as relative path.
2. Copy all python files and dependences in one folder like model, any additional library import.

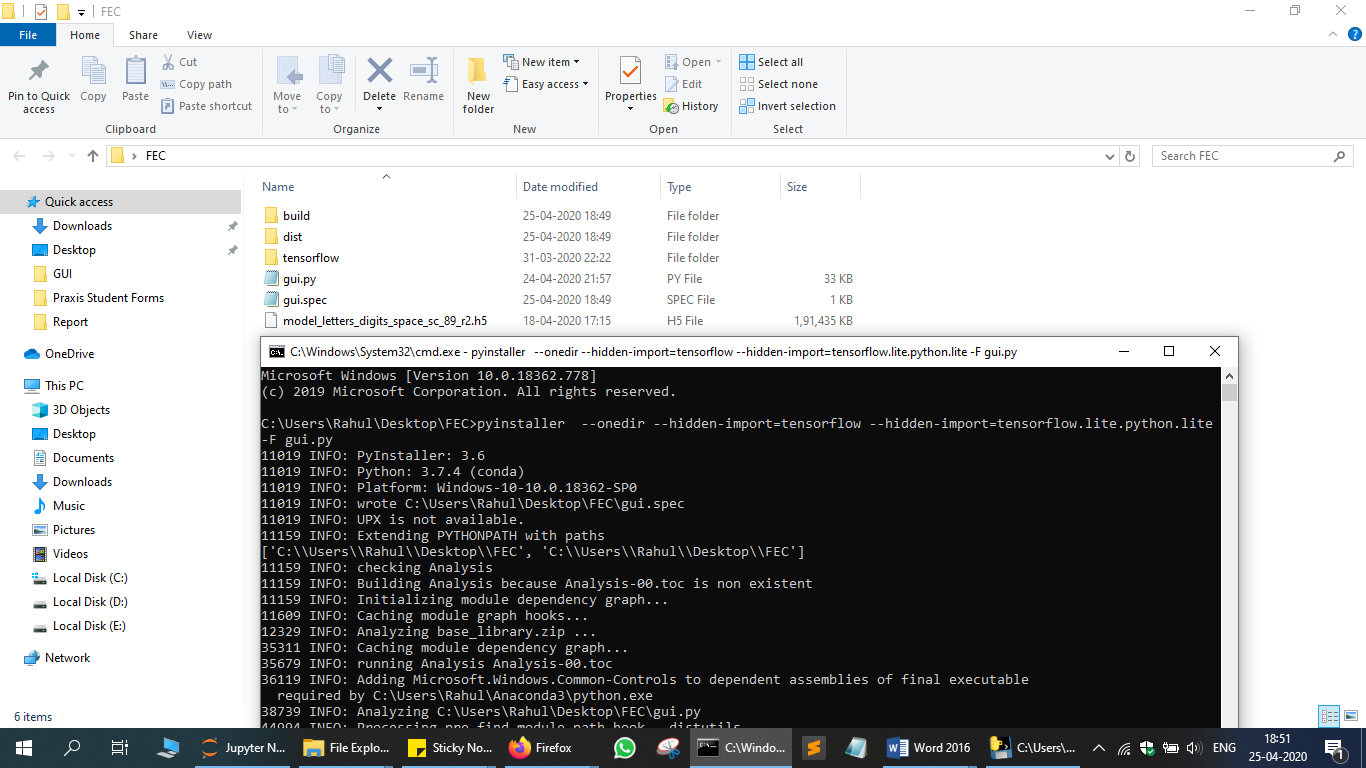
In our case copied tensorflow-core from Anaconda site packages for loading model as shown below.



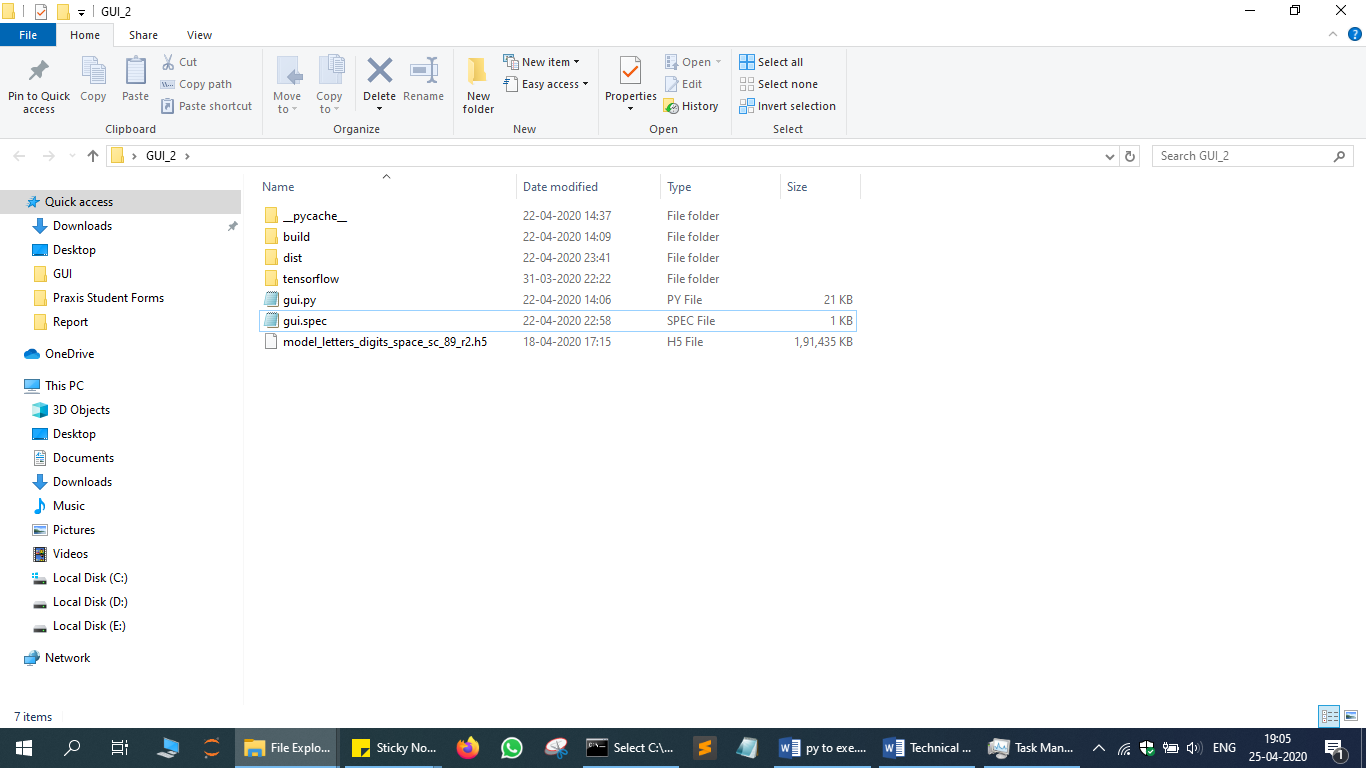
1. Open cmd and change directory to folder created above.
2. Run below command

***pyinstaller --onedir --hidden-import=tensorflow –hidden-import=tensorflow.lite.python.lite -F gui.py***

* + - –onedir will create separate folder containing exe file in ‘dist’ folder.
    - --hidden import will load packages mentioned specifically. For model load we are loading tensorflow.
    - Gui.py is our script name

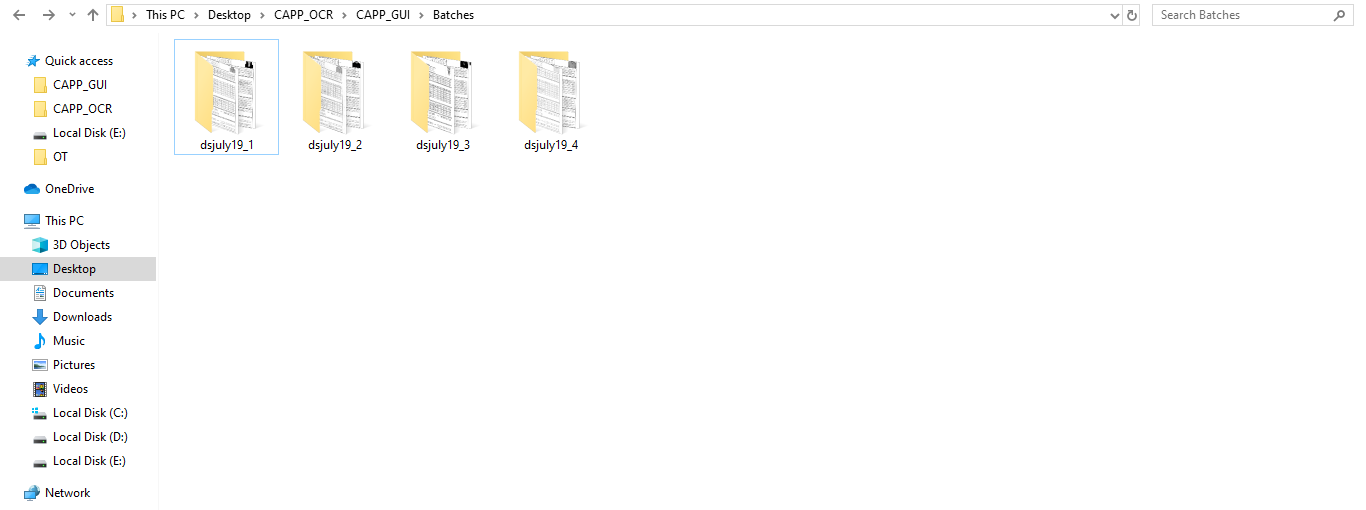


1. As shown below exe file is ready to use.

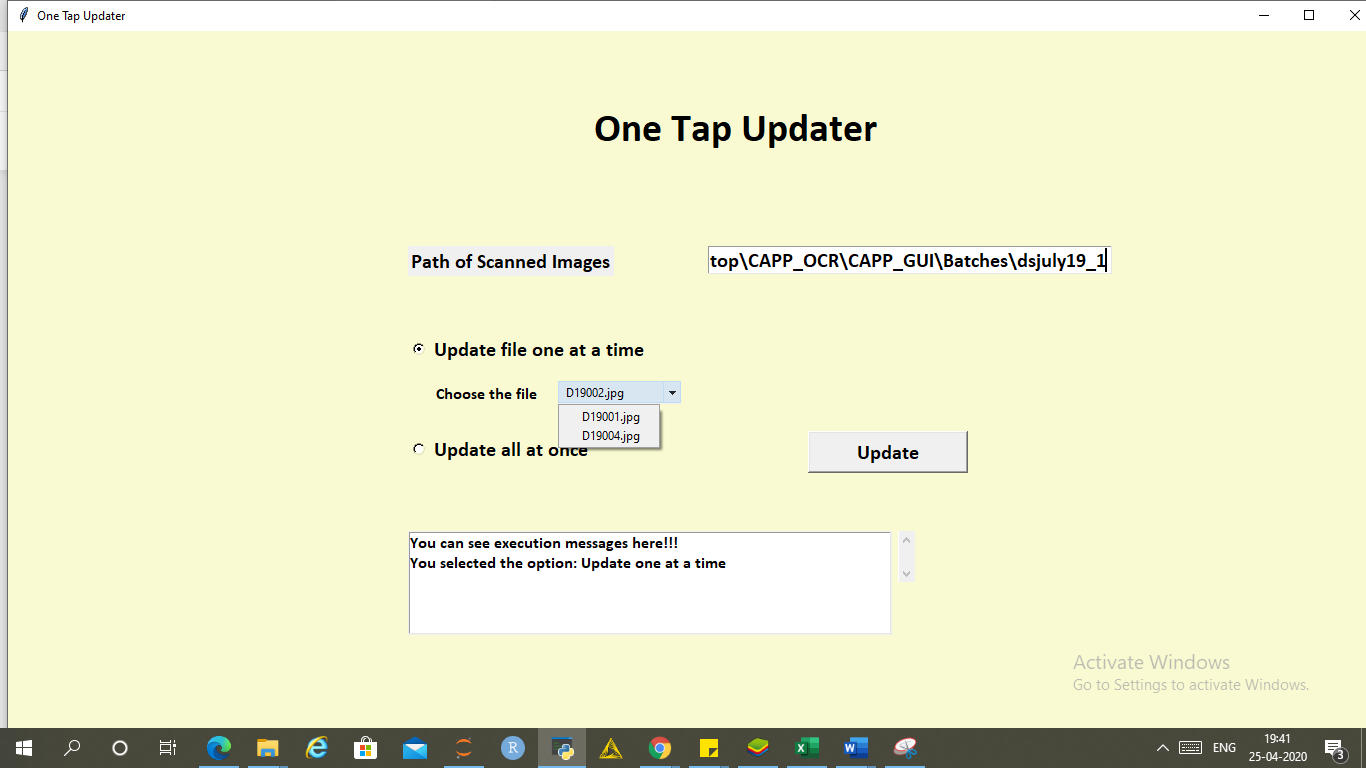


**Screenshots of the GUI for test cases.**

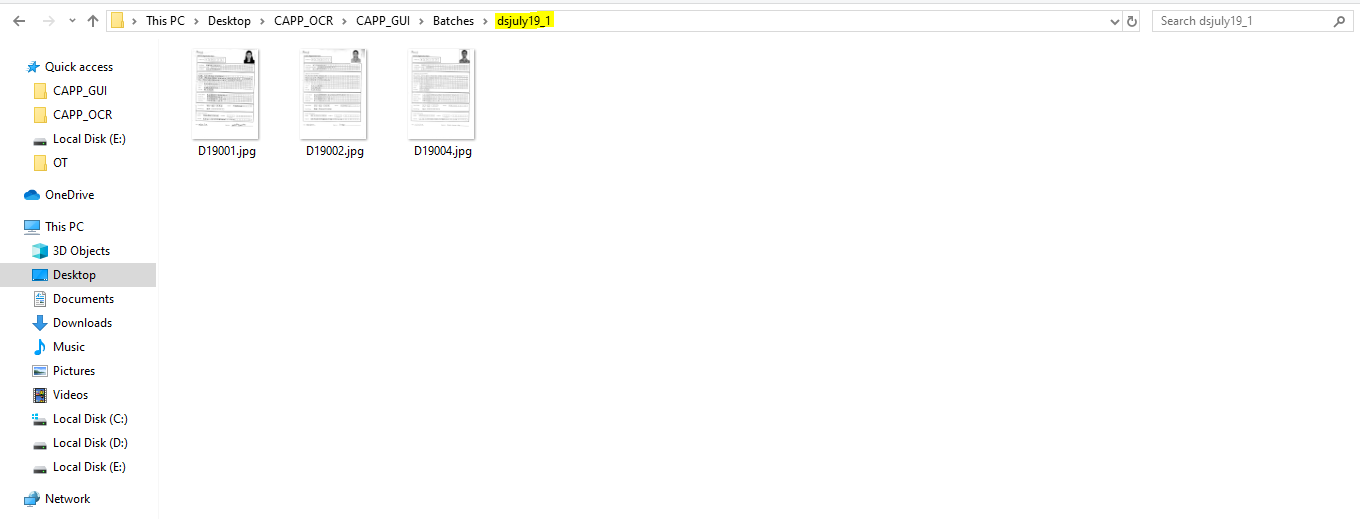
Folder having scanned images



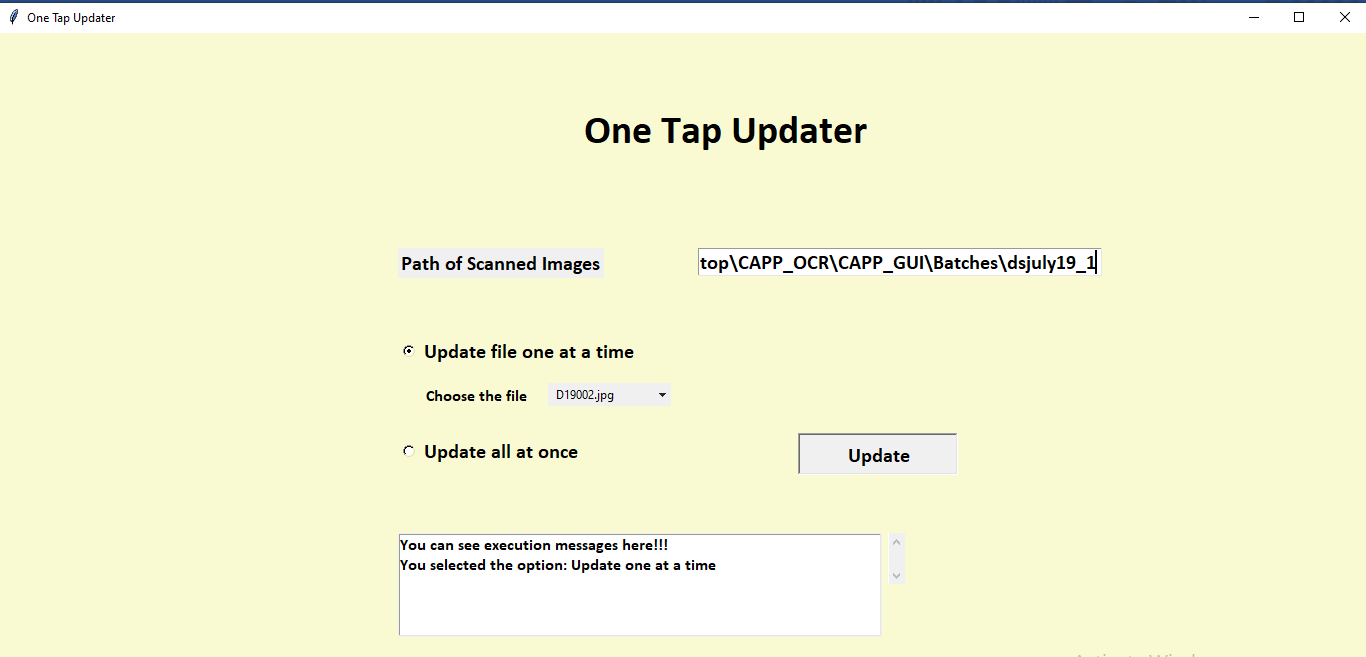
1. Updating one file at a time.



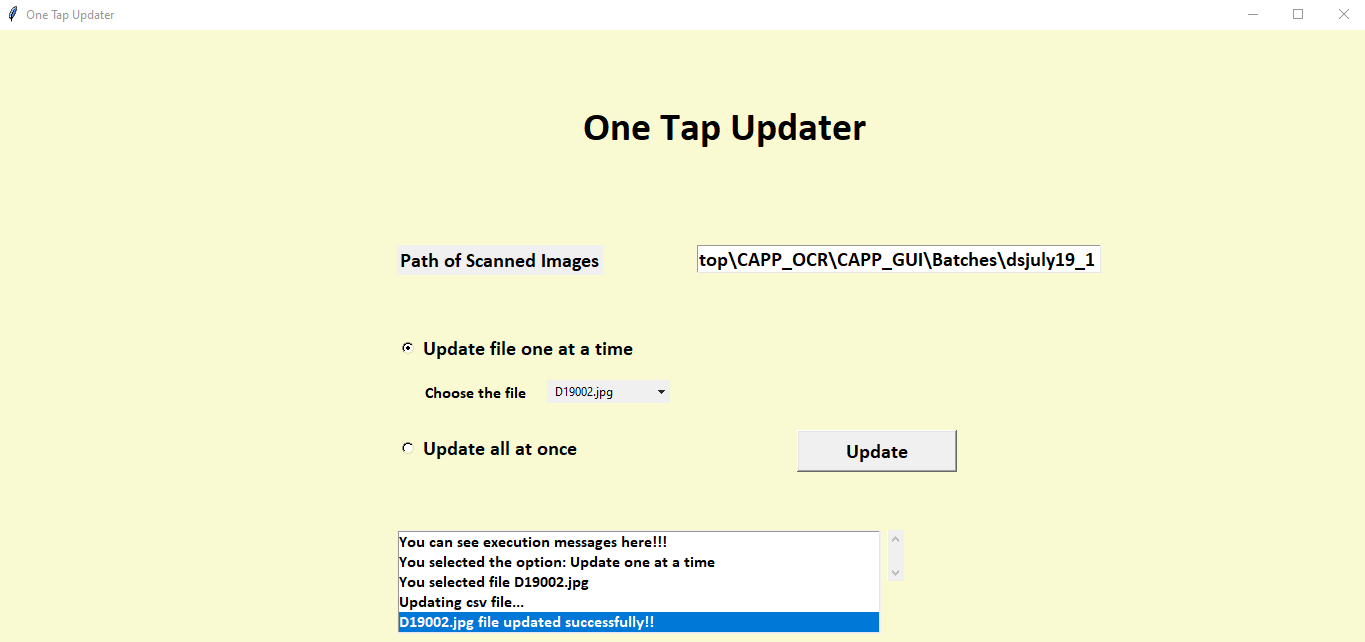
* Giving the path name of the folder having scanned images.
* We have selected the button ***update one at a time*** which can be seen in the message box below.
* The drop-down arrow gives the list of files in the directory and selected D19002.



Clicking on update button

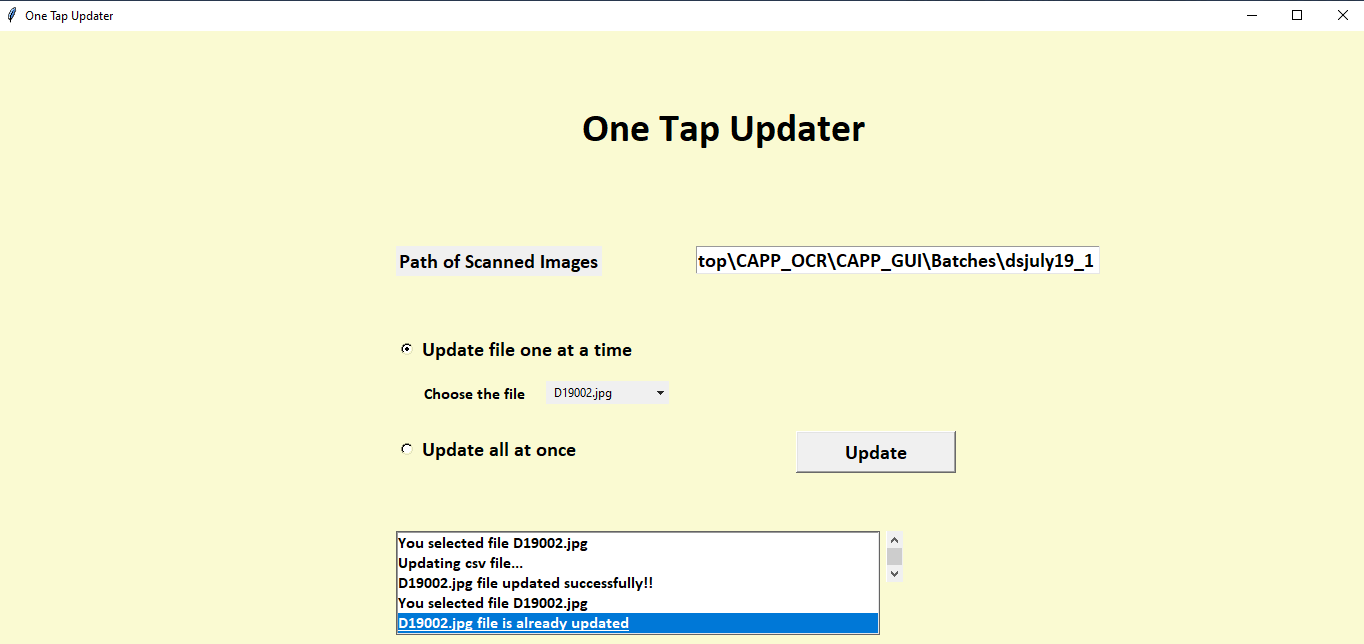


File update successfully



Running this will generate two files in the working directory one details.csv containing the predicted details and other track.txt containing name of the image already updated for reference.

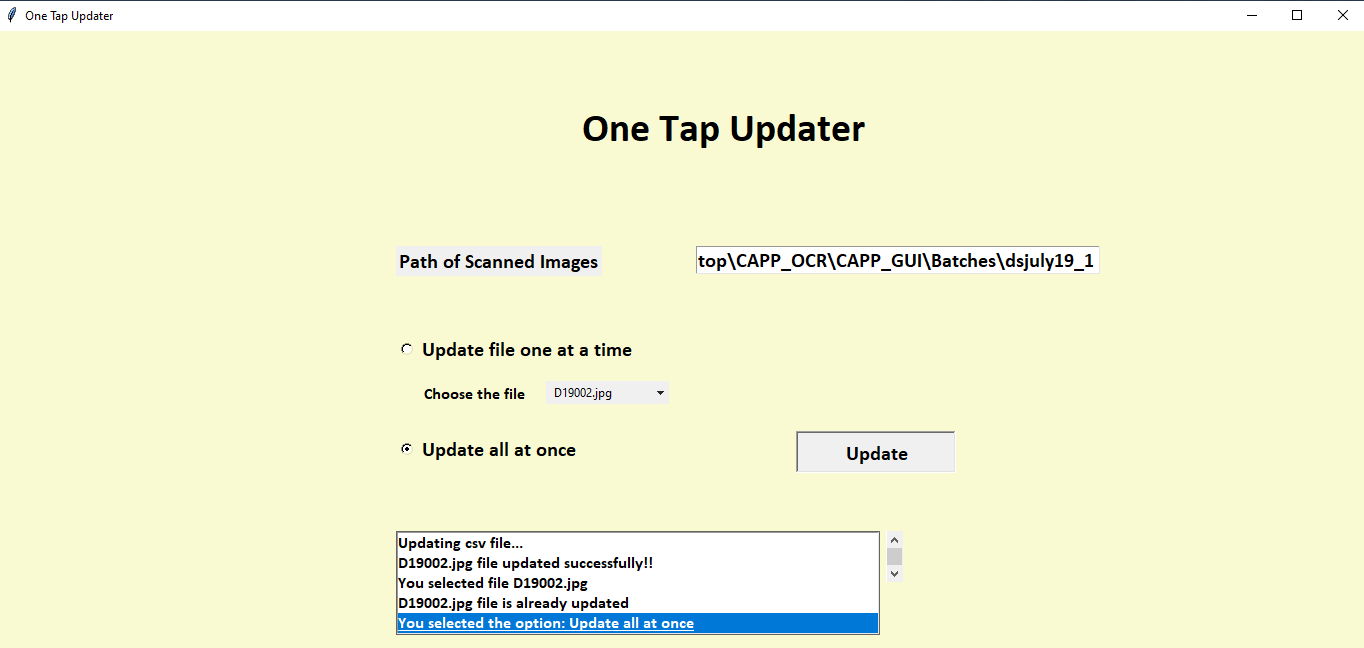
Now we will try to update that file again. let’s see:

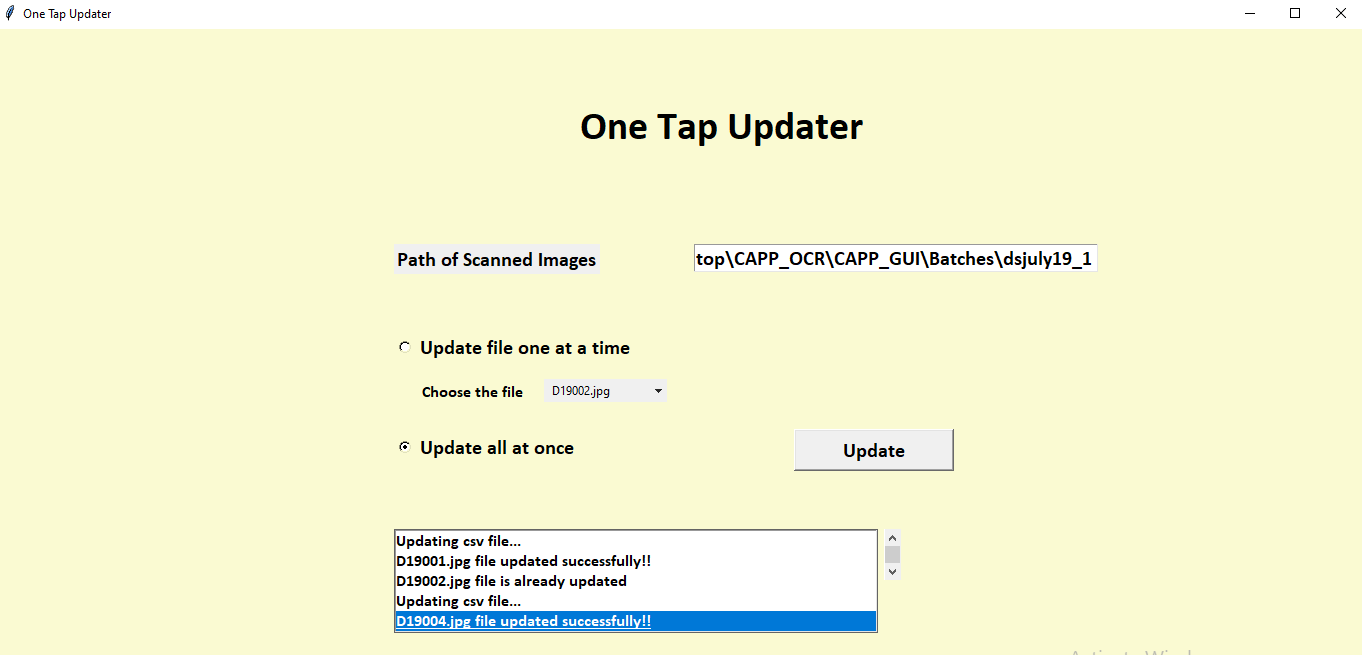


We get a message that file has already been updated.

1. Update all at once.

Clicking on Update.





Rest of all the files (two files) gets updated. we can see the messages in the message box.