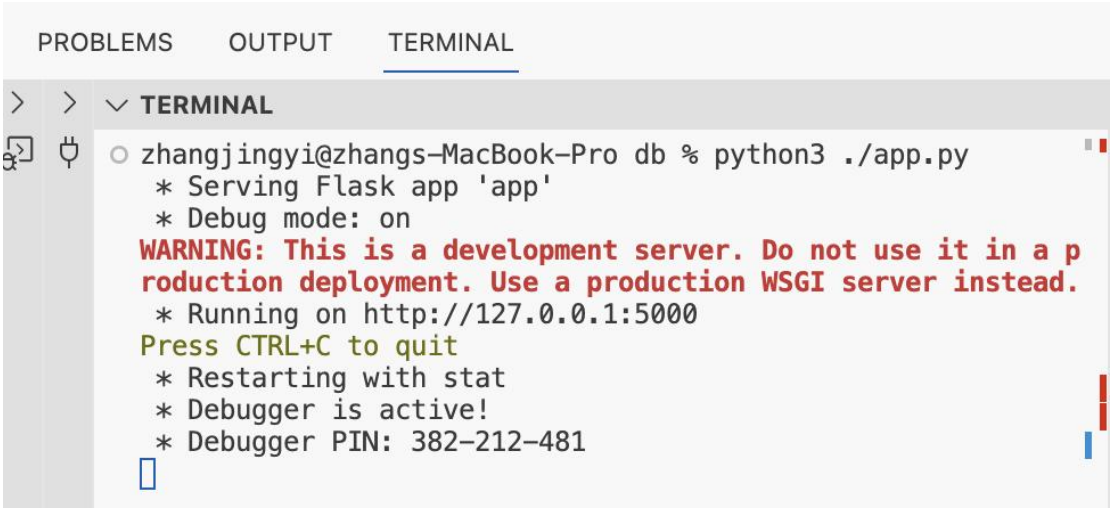
JINGYI ZHANG DEMO REPORT (jyz0328@bu.edu u26578499)

Usage Demonstration

Step 0: **download all code in main branch**

Step 1 open [**app.py**] : input **python3 ./app.py** in **terminal A** to run the app (the back end).



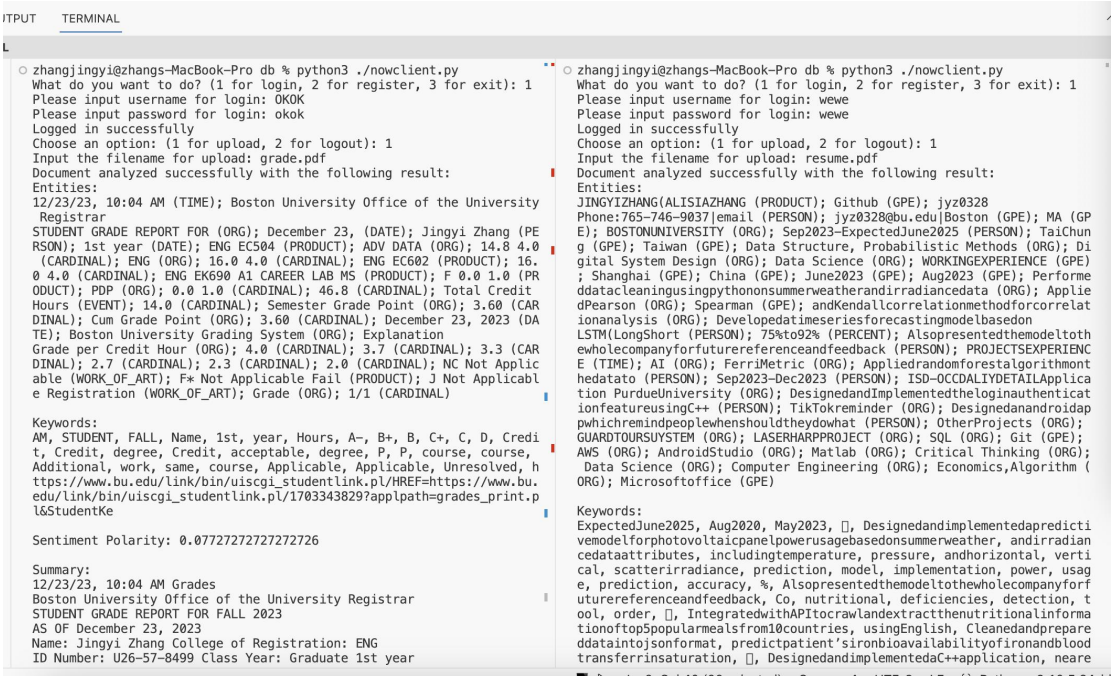
Step 2 client 1 usage : input **python3 ./nowclient.py** in **terminal B** to run the app for client 1 (the font end).

Then input 1 to login, then input username and password, then input 1 to be ready for uploading document.

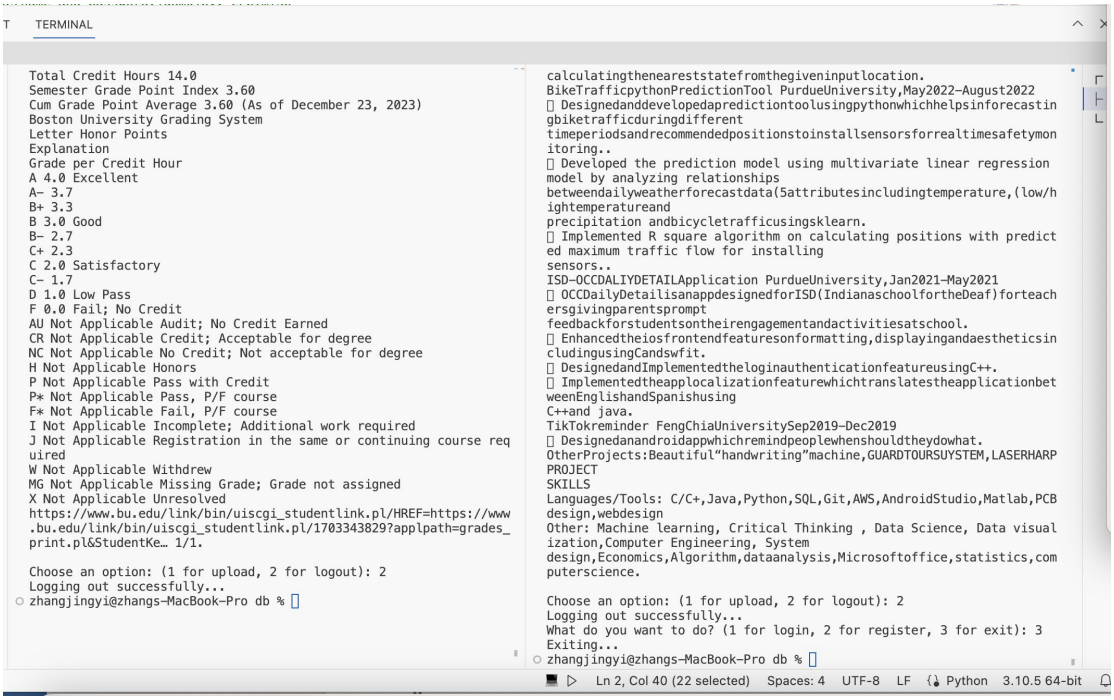
Step 3 client 2 usage : input **python3 ./nowclient.py** in **terminal C** to run the app for client 2 (the font end).

Then input 1 to login, then input username and password, then input 1 to be ready for uploading document, similar with client 1 does.

Step 4 upload document for both clients: I upload [grade.pdf] for client 1, and upload [resume.pdf] for client2, so we see both analyzing report can be formed. The analyzing report is long , you can see [result_of_grade.txt] and [result_of_example.txt] since these documents print out the analyzing result of these two documents. You can use other pdf and txt documents to test.



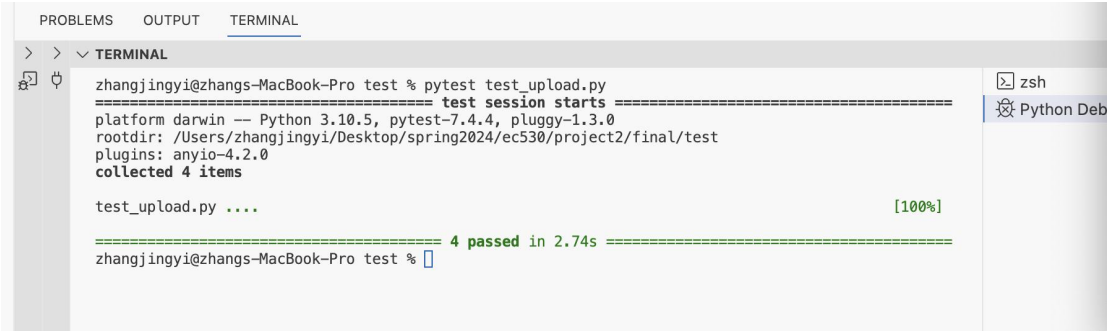
Step 5: input 2 and 3 to exit, or upload other document you want to test.



Test Part Demonstration

Step 0:download [tests] folder of this github

Step 1:open [test_upload] folder, then run [pytest test_upload.py] in terminal

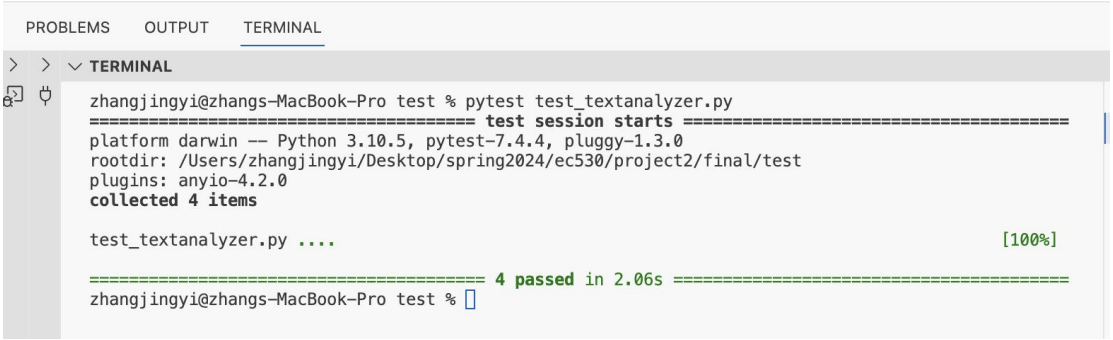


Code related to this step are test_upload.py, upload_separate.py and textanalyzer.py, in upload_separate.py we separate upload function from previous auth code parts. Also for upload_separate.py:

```
if[python3 upload_separate.py]on terminal we can return 1
if[python3 upload_separate.py none.txt]on terminal we can return 2
if[python3 upload_separate.py sample.png]on terminal we can return 3
if[python3 upload_separate.py grade.pdf]on terminal we can return 4
```

Then we build unit test based on upload_separate.py and get the demo figure.

Step 2:open [test_textanalyzer] folder, then run [pytest test_textanalyzer.py] in terminal



Code related to this step are test_textanalyzer.py and textanalyzer_separate.py, in textanalyzer_separate.py we separate textanalyzer function from previous auth and upload code parts. Also for textanalyzer_separate.py:

```
if[python3 textanalyzer_separate.py]on terminal we can return 1
```

```
if[python3 textanalyzer_separate.py none.txt]on terminal we can return
2
if[python3 textanalyzer_separate.py grade.pdf]on terminal we can return
3
if[python3 textanalyzer_separate.py test.txt]on terminal we can return
4
```

Then we build unit test based on textanalyzer_seperate.py and get the demo figure.