## Tests for Step3.py

## Written by B00319125

## 2017-04

## Step 3

### Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Functional Requirement | Description of Functionality | Description of Test | Test Case # |
| 1 | Provide a means of persistently storing the list of currently used request ID numbers in a file called Requests.txt. (Requests.pickle in this case) | Save the request list in a file called Requests.pickle | 1 |
| 2 | Use this file to verify if each newly generated ID is unique | Read the file Request.pickle to retrieve the request list and verify all the news request id | 2 |
| 3 | Keep this file up to date by adding new request id’s | For each valid new request id store in the request list and update the file Request.pickle, | 3 |
| 4 | Persistently store each request created by the user in a separate file    Provide Makefile and Readfile functions to facilitate this | For each new request, create a new file called <request-id-number>.pickle | 4 |
| 5 | Demonstrate that the all files are correctly stored to disk and that the index file (Requests.pickle) is correctly updated when new requests are created | Print the values in the request.pickle file to demonstrate the file is successfully updated | 5 |
| 6 | Whenever a new request is created and written to file, verify the file operation by reading the file and displaying the data using one of the display methods of the Request class | Print the values of each new request handling in with the correspondent pickle file | 6 |

### Test Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Description of Functionality | Test Data | Expected Result |
| 1 | Save the request list in a file called Requests.pickle | New Random Request ID | Save the data in the file request.pickle |
| 2 | Read the file Request.pickle to retrieve the request list and verify all the news request id | The request.pickle file created in the previous test | The list of stored request id handle by the file request.pickle |
| 3 | For each valid new request id store in the request list and update the file Request.pickle | Create a new request with any dummy data | Check if each new request are added to the request list inside in the Request.pickle file |
| 4 | For each new request, create a new pickle file called <request-id-number>.pickle | Create a new request with any dummy data | The files for each new request are successfully stored in the local directory |
| 5 | Print the values in the Request.pickle file to demonstrate the file is successfully updated | Have previously created a request | Every time that a new request id created, the program print the content of the Request.pickle file to demonstrate if it updated |
| 6 | Print the values of each new request handling in with the correspondent pickle file | Have previously created a request | Every time that a new request id created, the program print the content of the <request-id-number>.pickle file to demonstrate if it updated |
|  |  |  |  |

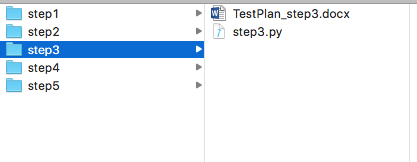
### Test Results & Actions

|  |  |  |
| --- | --- | --- |
| Test Case | Actual Results | Action Required |
| 1 | Success | None |
| 2 | Success | None |
| 3 | Success | None |
| 4 | Success | None |
| 5 | Success | None |
| 6 | Success | None |

### Evidence

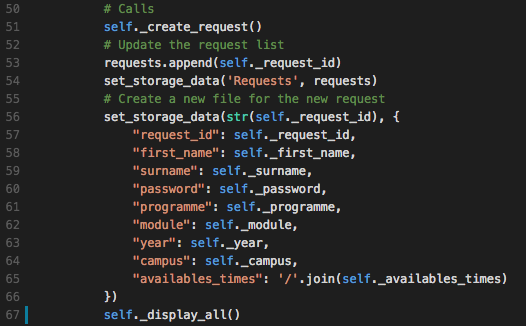
Test 1, Test 4

First this is a screenshot of the working directory after the tests



Note that there is no extra file yet

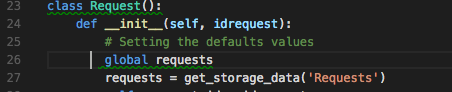
The code bellow shows the new calls stack order for this step



Create the request -> update the list of requests with the new request id -> Save the updated list of request in the Request.pickle file -> Create a new file for the current request -> display the data of the current request.

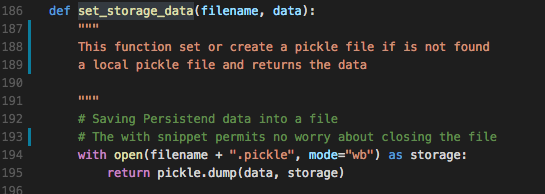
Test 2

Now the request list is set with the file Requests.pickle (If not exists the requests list is empy)

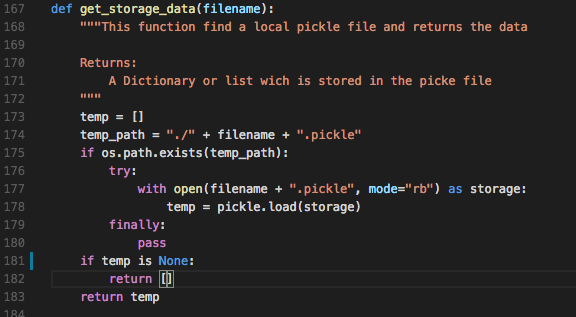


Test 3,4

The code bellow shows the set\_storage\_data (Makefile method equivalent):

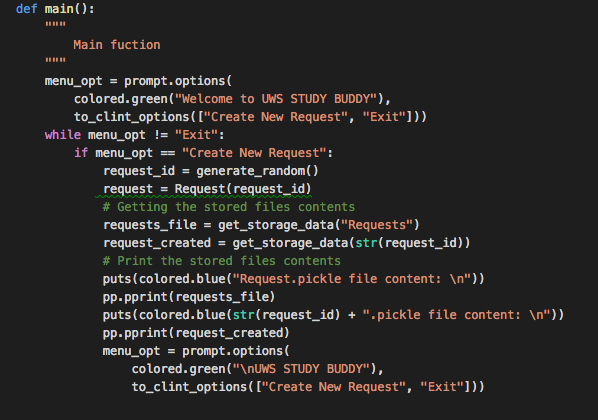


The code bellow shows the set\_storage\_data (Readfile method equivalent):



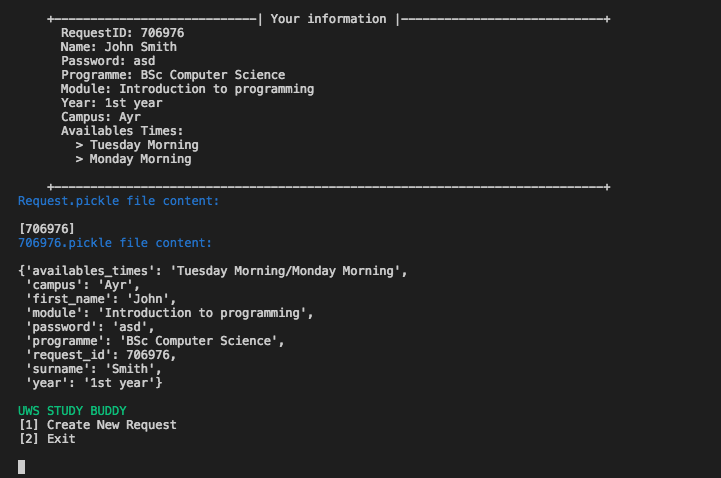
Note that method returns an empty list if the file not exists

The main function is modified for this test to print the contents of the files previously created:

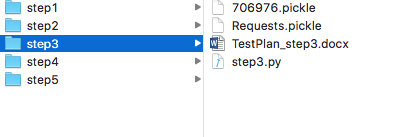


Test 3,5,6

When the requests files are successfully created the program print the content of the files and the content of the current request to compare them:

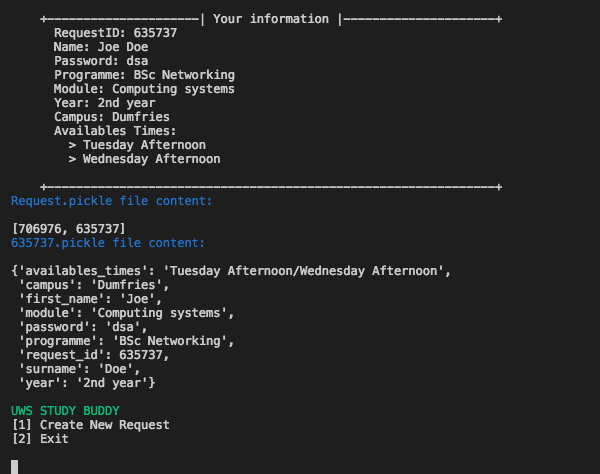


And the working directory looks like this:



With the new files created

One more test:



Note that the Request.pickle is update with the new request and the directory looks like:

