BAT Quinterac

Aaron Safer-Rosenthal
Ben Durston
Tamir Arnesty

Daily Script Source Listing

```
import tempfile
from importlib import reload
import os
import io
import sys
import quinterac
from quinterac import FrontEnd

path = os.path.dirname(os.path.abspath(__file__))

class Daily:

   def __init__(self, terminal_input1, terminal_input2, terminal_input3, input_valid_accounts):
       helper(terminal_input1, input_valid_accounts)
       helper(terminal_input2, input_valid_accounts)
       helper(terminal_input3, input_valid_accounts)



def helper(terminal_input, input_valid_accounts):

   # cleanup quinterac
   reload(quinterac)

   # create a temporary file in the system to store output transactions
   temp_fd, temp_file = tempfile.mkstemp()
   transactionSummaryFile = temp_file

   # create a temporary file in the system to store the valid accounts:
   temp_fd2, temp_file2 = tempfile.mkstemp()
   validAccountsListFile = temp_file2

   temp_fd3, temp_file3 = tempfile.mkstemp()
   sessionFile = temp_file3
```

```
    with open(validAccountsListFile, 'w') as wf:
        wf.write('\n'.join(input_valid_accounts))

    # prepare program parameters
    sys.argv = [
        'quinterac',
        validAccountsListFile,
        transactionSummaryFile,
        sessionFile]

    # set terminal input
    sys.stdin = io.StringIO(
        '\n'.join(terminal_input))

    # run the program
    FrontEnd.FrontEnd(validAccountsListFile, transactionSummaryFile)

    os.close(temp_fd)
    os.close(temp_fd2)
    os.close(temp_fd3)

    os.remove(temp_file)
    os.remove(temp_file2)
    os.remove(temp_file3)
```

Weekly Script Source Listing

```
import os
from time import sleep
from dailyScript import Daily

class Weekly:

    def __init__(self):
        validAccounts = open('quinterac/validAccountsListFile.txt', 'r')
        self.inputValidAccounts = validAccounts.readlines()
```

```
        self.runWeek()

    def runWeek(self):
        """
        This method runs the daily script 5 times. Each time with three different transaction sets.
        """
        day = 1
        while day <= 5:
            transactionInputs1 = self.transactionSetOne(day)
            transactionInputs2 = self.transactionSetTwo(day)
            transactionInputs3 = self.transactionSetThree(day)
            a = Daily(transactionInputs1, transactionInputs2, transactionInputs3,
self.inputValidAccounts)
            os.system("python3 -m quinteracBackend")#Command runs the backend after the day is done.
            day = day + 1

    def transactionSetOne(self, day):
        """ 5 Days of one transaction set"""
        if day == 1:  # Valid accounts list is empty, so new accounts must be made on the first day.
            transactionInputs = [
                'login',
                '1',
                'new',
                '7777776',
                'Ben',
                'back',
                'logout']
        elif day == 2:
            transactionInputs = [
                'login',
                '1',
                'dep',
                '7777776',
                '7777',
                'back',
                'logout']
        elif day == 3:
            transactionInputs = [
                'login',
```

```
            '2',
            'wdr',
            '7777776',
            '500',
            'back',
            'logout']
    elif day == 4:
        transactionInputs = [
            'login',
            '2',
            'xfr',
            '7777776',
            '4444444',
            '10',
            'back',
            'logout']
    else:
        transactionInputs = [
            'login',
            '1',
            'del',
            '7777776',
            'Ben',
            'back',
            'logout']
    return transactionInputs

def transactionSetTwo(self, day):
    """ 5 Days of another transaction set"""
    if day == 1:  # Valid accounts list is empty, so new accounts must be made on the first day.
        transactionInputs = [
            'login',
            '1',
            'new',
            '4444444',
            "Aaron",
            'back',
            'logout']
    elif day == 2:
```

```
            transactionInputs = [
                'login',
                '1',
                'dep',
                '4444444',
                '69420',
                'back',
                'logout']
        elif day == 3:
            transactionInputs = [
                'login',
                '2',
                'wdr',
                '4444444',
                '500',
                'back',
                'logout']
        elif day == 4:
            transactionInputs = [
                'login',
                '2',
                'xfr',
                '1122334',
                '4444444',
                '10',
                'back',
                'logout']
        else:
            transactionInputs = [
                'login',
                '1',
                'del',
                '4444444',
                'Aaron',
                'back',
                'logout']
        return transactionInputs

    def transactionSetThree(self, day):
```

```
    """ 5 Days of the last transaction set """
    if day == 1: # Valid accounts list is empty, so new accounts must be made on the first day.
        transactionInputs = [
            'login',
            '1',
            'new',
            '1122334',
            "Tamir",
            'back',
            'logout']
    elif day == 2:
        transactionInputs = [
            'login',
            '1',
            'dep',
            '1122334',
            '200',
            'back',
            'logout']
    elif day == 3:
        transactionInputs = [
            'login',
            '2',
            'wdr',
            '1122334',
            '100',
            'back',
            'logout']
    elif day == 4:
        transactionInputs = [
            'login',
            '2',
            'xfr',
            '7777776',
            '1122334',
            '10',
            'back',
            'logout']
    else:
```

```
            transactionInputs = [
                'login',
                '1',
                'del',
                '1122334',
                'Tamir',
                'back',
                'logout']
        return transactionInputs
if __name__ == "__main__":
    a = Weekly()
```

Transaction Inputs used on Day 5

First Session:	Second Session:	Third Session:
'login',	'login',	‘login’
'1',	‘1’,	‘1’,
'del',	‘del’,	‘del’,
'7777776',	‘4444444’,	‘1122334’,
'Ben',	‘Aaron’,	‘Tamir’,
'back',	‘back’,	‘back’,
'logout'	‘logout’	‘logout’

Merged Transaction Summary File - Day 5

quinterac > mergedTransactionSummaryFile.txt

```
1   DEL 0000000 000 7777776 Ben
2   DEL 0000000 000 4444444 Aaron
3   DEL 0000000 000 1122334 Tamir
4   EOS 0000000 000 0000000 ***
5
```

Master Accounts File - Day 1

quinteracBackend > masterAccountsFile.txt

```
1    7777776 0 Ben
2    4444444 0 Aaron
3    1122334 0 Tamir
```

Master Accounts File - Day 2

quinteracBackend > masterAccountsFile.txt

```
1    7777776 7777 Ben
2    4444444 69420 Aaron
3    1122334 200 Tamir
```

Master Accounts File - Day 3

quinteracBackend > masterAccountsFile.txt

```
1    7777776 7277 Ben
2    4444444 68920 Aaron
3    1122334 100 Tamir
```

Master Accounts File - Day 4

quinteracBackend > masterAccountsFile.txt

```
1    7777776 7297 Ben
2    4444444 68900 Aaron
3    1122334 100 Tamir
```

Master Accounts File - Day 5

quinteracBackend > masterAccountsFile.txt

```
1    7777776 7297 Ben
2    4444444 68900 Aaron
3    1122334 100 Tamir
```

Integration Defect Report

Problem	**Solution**
Weekly transaction script would not work correctly, even though each day would work properly, individually. The program would think that the days were in the same session. This caused an error saying the new account couldn't be used for a transaction in the session it was created in.	Removed error checking, because we would need to overhaul our error checking to have it work with the script. We know that the script does not feed any illegal inputs to the system, so it is not necessary for it to check. To further fix this problem, we would have to refactor both our frontend and backend sessions to better meet the constraint.