Crawler Tool - HY Judge

Introduction

This tool is designed especially for HY Judge users. It help both the administrator to acess to the Judge system with a simple terminal windows. In addition to the functions on the website, this tool can help you download your source code along with all message on the website.Also, if you have sovled certain problem, this tool allows users to read other classmates' masterpieces. After all, this is just a prototype which can be universal for any Judge Girl system by simply changing the domain of the Judge Girl system.

Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes. See deployment for notes on how to deploy the project on a live system.

Prerequisite

Things you need to install beforehand and how to install them

```
Python3
pip install bs4, json

(default libaray) requests, os, collections
```

Executing

1. After the installation, you must keep all files(including `main.py` , `tools.py` and `usr.py`).
2. To start the game, execute `main.py` or enter following command in your terminal `python main.py`
3. all downloaded file will be in the same directory named after your uid

Instuctions

1. Login for more features: You will be asked to enter your username and password for HY Judge.After that you grante acess to more features functions.

2. Problems: Without login, you still full acess to the Problems

3. Check my latest submission status: You will be asked to enter your username and how many submissions you want to check

4. submissions(live): By choosing this one you can check submissions including your classmates'.

Further details of each functions:

`1. Login for more features:`

`(1)Download all of my code`
`(2)Download all of my ac code`
`(3)Check certain code with sid` : you will get this number when you check submissions
`logout: logout`

`2. Problems:`

`(1) latest problems` : It prints out problems which has less than 40 user acepted or latest 5 problems
`(2) all problems` : This prints out all the problems on the Judge

`3. Check my latest submission status:` check your submissions status with your username for any numbers you like

`4. submissions(live):` see 25 latest submissions on the Judg

Function Explaination

*** the code below omits some detail operations to be more readable**

`main.py`

```
from bs4 import BeautifulSoup
import requests, json, os
import tools, usr
def api(level):
    valid1 = ['1','2','3','4','q']
    valid2 = ['1','2','3','logout','q']
    clear()
    while level == 1:
        selection = input('Please input something to move on\n')
        if selection in valid1:
            return selection
        else:
            print('Invalid input')
    while level ==2:
        selection = input('Please input something to move on\n')
        if selection in valid2:
            return selection
        else:
            print('Invalid input')

def clear():
    os.system('cls' if os.name == 'nt' else 'clear')

restext = ['', 'CE', 'OLE', 'MLE', 'RE', 'TLE', 'WA', 'AC', 'Uploading...
session = None

while True:
    selecton = api(1)
    if selecton == '1':
        if session == None:
            session, lgn = tools.login()
        uid = str(tools.getuid(lgn=lgn))
        me = usr.user(uid)
        selection = api(2)
        if selection == '1':
            me.fetch_all(session=session,r=False, status ='all')
        if selection == '2':
            me.fetch_all(session=session,r=False,status ='AC')
        if selection == '3':
            # for checking the certain submissions with sid
        if selection == 'logout':
            session = None
    if selecton == '2':
        tools.latestproblem()
    if selecton == '3':
       # for checking the latest n submissions with username
    if selecton == '4':
        response = requests.get("http://140.112.17.207/api/submission")
        subs = json.loads(response.text)
        for sub in subs:
            print('{0}: {1} {2} {3} {4}'.format(sub['ttl'],restext[int(sul
        print('---------------------------')
        input('Press any key to continue')
    if selecton == 'q':
        break
```

The main while loop is for users to choose which function to use

Variables	Type	Usage
sessions	request.seeions	For login session
selections	str	To switch between different functions
uid	str	A number which Judge distribute when your account registred and is used for most case
me	user	to store information about current user and can call several functions
sid	str	A number which assigned by the Judge for us to get acess to certain submission
filename	str	For reading files or writing in data
n	int	Refer to the number of submissions you want to print out
response	requset	storing a the response for our request
subs	dict	A dictionary for live sunbmissions

```
def api(level):
    valid1 = ['1','2','3','4','q']
    valid2 = ['1','2','3','logout','q']
    clear()
    while level == 1:
        selection = input('Please input something to move on\n')
        if selection in valid1:
            return selection
        else:
            print('Invalid input')
    while level ==2:
        selection = input('Please input something to move on\n')
        if selection in valid2:
            return selection
        else:
            print('Invalid input')
```

Function: It print out a menu determined by the argument level and repeat until the user input a valid input

Return: The user's selection

```
def clear():
    os.system('cls' if os.name == 'nt' else 'clear')
```

Function: It use th built-in command to clean the screen

Return: None

usr.py

```
class submission:
    def __init__(self,raw):
        self.sid = raw['sid']
        self.uid = raw['uid']
        self.pid = raw['pid']
        self.res = raw['res']
        self.scr = raw['res']
        self.scr = raw['scr']
        self.ttl = raw['ttl']
```

Class: It creat an object which can store some attributes of a submission
Return: None

```
class user:
    def __init__(self,uid):
        try:
            os.mkdir(str(uid))
        except:
            pass
            #print("Existed")
        self.uid = uid
        self.subcount = 0
        self.sublist = []
        self.para  = {'limit': '25', 'uid': str(self.uid),'page':'1'}
        self.restext = ['', 'CE', 'OLE', 'MLE', 'RE', 'TLE', 'WA', 'AC',
        print("Create user successfully")
```

Class: A user class store status of certain user and initailize some parameter
Return: None

```
    def update_submissions(self):
        filename = "./"+str(self.uid)+"/"+"submissions.json"
        with open(filename,'w') as f:
            f.write('[')
            for pagenum in range(100):
                self.para['page'] = pagenum
                response = requests.get("http://140.112.17.207/api/submis
                if len(response.text) == 2:
                    break
                else:
                    a = response.text.strip('[')
                    a = a.strip(']')
                    if pagenum != 0:
                        a = ',' +a
                f.write(a)
                data = json.loads(response.text)
                for raw in data:
                    self.sublist.append(submission(raw))
            f.write(']')
        self.subcount = len(self.sublist)
```

Funcion: It update all submissions of the user and store it in a .json file
Return: None

```
    def fetch_all(self,*,session,r=False,status='all'):
        self.update_submissions()
        for i in range(len(self.sublist)):
            sub = self.sublist[i]
            print(i,": "+tools.download_source(sid = sub.sid,r=r,session=
```

Funcion: It calls the download_source function in tools to download all submission of the user.And print out if the crwal succeed or not.
Return: None

```
def latest_submissions(self,n):
        self.update_submissions()
        filename = "./"+str(self.uid)+"/"+"submissions.json"
        with open(filename) as f:
            sublist = json.loads(f.read())
            for i in range(n):
                sub = sublist[i]
                ttl = sub['ttl']
                res = self.restext[int(sub['res'])]
                sid = sub['sid']
                scr = sub['scr']
                print('{0} : {1} {2} {3}'.format(sid,ttl,res,scr))
```

Funcion: It print out the latest n submission of the user
Return: None

`tools.py`

```
def login():
    data = {'lgn':"",'pwd':""}
    s = requests.session()
    while True:
        data['lgn'] = input("username: ")
        data['pwd'] = input("password: ")
        if resp.text.find("帳號或密碼錯誤") == -1:
            print("Login Succesfully!")
            break
        else:
            print("帳號或密碼錯誤")
    return s, data['lgn']

```

Function: Repeats until the user sucessfully login with their username and password

Return: request.session, username of the user

```
def download_source(*,session,sid,r,status='all',solo=False):
    s = session
    data = json.loads(requests.get('http://140.112.17.207/api/result?sid=
    res, ttl, scr, uid = int(data['res']), str(data['ttl']), str(data['scr
    restext = ['', 'CE', 'OLE', 'MLE', 'RE', 'TLE', 'WA', 'AC', 'Uploading
    filename = "./"+uid+"/"+ name +"/"+str(sid)+".txt"
    if str(sid)+".txt" in os.listdir('./'+uid+"/"+name) and not r:
        return str(sid)+" Existed!"
    resp = s.get('http://140.112.17.207/source/highlight/'+str(sid))
    source = s.get('http://140.112.17.207/source/'+str(sid))
    soup = BeautifulSoup(resp.text,'html.parser')
    if soup.find('h3').string != 'Result':
        return str(sid)+"Error message!"
    if solo:
        filename = "./temp/"+ str(sid)+ ".txt"
        with open(filename,'w') as f:

            f.write(source.text)
    filename = "./"+uid+"/"+ name +"/"+str(sid)+".txt"
    with open(filename,'w') as f:
        f.write(source.text)
    return str(sid)+": Sucessful!"

```

Function: It downloads source code along with all messages on the website and store it in a txt file. All the files will be classified by their problem titles.

Return: Message whether this crawl succeed or not

```
def update_usr_dic():
    response = requests.get("http://140.112.17.207/ranklist?page=2")
    soup = BeautifulSoup(response.text,'html.parser')
    pagelink = soup.find_all('a',{"class":"page-number"})
    usr = dict()
    for link in pagelink:
        response = requests.get("http://140.112.17.207"+ link.get('href'))
        soup = BeautifulSoup(response.text,'html.parser')
        a = soup.find_all('a',{"class":"nav_a"})
        for i in a:
            usernumber = i.get('href').split("/")[1].lower()
            username = i.string
            usr[username] = usernumber
    with open('user.json','w') as f:
        f.write(json.dumps(usr))

```

Function: This function update the dictionary of username and its uid accordingly. All data also store in a json file

Return: None

```
def update_problemset():
    response = requests.get("http://140.112.17.207/problems/domain/0#1")
    soup = BeautifulSoup(response.text,'html.parser')
    problemlinks = soup.find_all('a',{"class":"pure-menu-link","style":"te
    problemset = dict()
    problempage = []
    payloads = {'did':'0','uid':'101','lid':'1'}
    problemset =  collections.OrderedDict(sorted(problemset.items()))
    with open('problemset.json','w') as j:
        j.write(json.dumps(problemset))
    with open('problemset.txt','w') as t:
        for key in problemset:
            t.write(str(key)+": "+str(problemset[key][0])+'\n'*2)
```

Function: It automatically craw down all problem from the website and store information in a json file and in a txt file

Return: None

```
def getuid(*,lgn='',uid=''):
    update_usr_dic()
    with open('user.json','r') as f:
        usrdict = json.loads(f.read())
    return usrdict[username]

```

Function: It can either convert uid to username or username to uid. If no input is given it ask the user to input their username
Return: uid or username according to the input argument

```
def latestproblem():
    update_problemset()
    with open('problemset.json') as f:
        probdict = json.loads(f.read())
    count = 0
    while True:
        print("1. latest problems")
        print('2. all problems')
        print('b: back')
        selection = input('Please choose a mode\n')
        if selection == 'q':
            return
    if mode == '1':
        print(key,probdict[key][0])
        if count == 0:
           print(key,probdict[key][0])
    if mode == '2':
        for key in probdict:
            print(key,probdict[key])
    print('-------------')
    input("Press enter to move on")
```

Function: print out latest submissions according to the mode user input. If they choose 1, latest problems,which are problems of less than 40 people or 5 latest problems. If they choose 2, all problem will be printed out.

Return: None

Running the tests

- a good way to use this tool is to check the sid of a certain submission which interest you
- then login to check the submission
- note that you can only acess to submisions when you have solved the problem
- another useful function is to download all the AC submissions

Built With

- Anaconda (https://www.anaconda.com/) - The environment used
- bs4 (https://www.crummy.com/software/BeautifulSoup/bs4/doc/) - module used to sort out response of each request
- json (https://docs.python.org/3/library/json.html) - module used to decode and encode json file