**MINISTERUL EDUCAŢIEI, CULTURII și CERCETĂRII**

**al REPUBLICII MOLDOVA**

**UNIVERSITATEA TEHNICĂ a MOLDOVEI**

**FACULTATEA CALCULATOARE, INFORMATICǍ**

**şi MICROELECTRONICǍ**

**Departamentul Informatică și Ingineria Sistemelor**

**RAPORT**

**la practica în producție/tehnologică**

**Tema: Elaborarea unei aplicațiii**

**A efectuat:**

**st. gr. C-171 D. Melniciuc**

**Chişinău 2020**

**Cuprins**

[Capitolul I 2](#_Toc44510261)

[Descrierea proiectului 2](#_Toc44510262)

[Tema: 2](#_Toc44510263)

[Informatie generala: 3](#_Toc44510264)

[Capitolul II 4](#_Toc44510265)

[Descrierea instrumentelor și tehnologiilor utilizate. 4](#_Toc44510266)

[Instrumentul principal: 4](#_Toc44510267)

[Instrumente aditionale: 5](#_Toc44510268)

[Capitolul III 5](#_Toc44510269)

[Elaborarea aplicației 5](#_Toc44510270)

[Schema programului: 5](#_Toc44510271)

[Fragmente semnificative din cod 7](#_Toc44510272)

[Importurile: 7](#_Toc44510273)

[Alegerea webdriverului 7](#_Toc44510274)

[Citirea informatiei de la consola 8](#_Toc44510275)

[Inserarea informatiei pe site 8](#_Toc44510276)

[Incarcarea a mai multe pagini cu informatie 9](#_Toc44510277)

[Preluarea link-urilor 9](#_Toc44510278)

[Accesarea unei pagini 10](#_Toc44510279)

[Salvarea informatiei 10](#_Toc44510280)

[Alegerea pentru a curs valutar 11](#_Toc44510281)

[Introducerea sumei in cimp 12](#_Toc44510282)

[Preluarea datelor la curs valutar 12](#_Toc44510283)

[Inchiderea browserului si programului 13](#_Toc44510284)

[Concluzie: 13](#_Toc44510285)

# Capitolul I

## Descrierea proiectului

### Tema:

Tema proiectului este "Sistem pentru cautarea informatiei in reteaua Internet".

Am ales tema data deoarece sunt interesat in automarea proceselor in reateaua Internet, iar tema data ofera posibilitatea de a invata automarea si de a crea deprinder in lucru automatizat

### Informatie generala:

Aplicatia creata este realativ simpla dar are potential de a fi dezvoltata in ceva de scala mai mare. Aplicatia reprezinta un webcrawl-er, internet bot citeodata numit spider sau prescurtat crawler, ce acceseaza internetul in cautarea informatiei introdusa de utilizator. Search engine-urile si unele alte site-uri folosesc software crawling pentru a-și actualiza conținutul web sau indicii de conținut web al altor site-uri. Crawler-urile web copiază paginile pentru procesare de către un Search engine care indexează paginile descărcate astfel încât utilizatorii să poată căuta mai eficient.

Numărul de pagini de Internet este extrem de mare, chiar și cele mai mari crawler-uri nu pot face un index complet. Din acest motiv, search engine-urile aveau probleme cu preluarea rezultatatelor de căutare relevante în primii ani ai World Wide Web, înainte de 2000. Astăzi, rezultatele relevante sunt date aproape instantaneu ce demonstreaza potentialul crawler-lor.

Web crawler-urile pot valida hyperlink-uri și cod HTML. Ele pot fi, de asemenea, utilizate pentru razuirea web (a se vedea, de asemenea, programarea bazată pe date).

Un crawler începe cu o listă de adrese URL de vizitat, numite “seeds”. Pe măsură ce crawlerul vizitează aceste adrese URL, identifică toate hiperlinkurile din pagini și le adaugă pe lista URL-urilor de vizitat, denumite “crawl frontier”. Adresele URL de la frontieră sunt vizitate recursiv conform unui set de politici. Dacă crawlerul efectuează arhivarea site-urilor web, acesta copiază și salvează informațiile. Arhivele sunt de obicei stocate astfel încât să poată fi vizualizate, citite și navigate așa cum au fost pe web live, dar sunt păstrate ca „ snapshots”.

Arhiva este cunoscută sub numele de repositoriu și este proiectată pentru a stoca și gestiona colecția de pagini web. Repositoriul stochează doar pagini HTML și aceste pagini sunt stocate ca fișiere distincte. Un repositoriu este similar cu orice alt sistem care stochează date, precum o bază de date modernă. Singura diferență este că un depozit nu are nevoie de toate funcționalitățile oferite de un sistem de baze de date. Depozitul stochează cea mai recentă versiune a paginii web preluată de crawler.

Volumul mare presupune că crawlerul poate descărca doar un număr limitat de pagini Web într-un anumit timp, deci trebuie să acorde prioritate descărcărilor sale. Rata mare de modificare poate presupune că paginile au fost deja actualizate sau chiar șterse.

Numărul de URL-uri posibile de preluat de crawler sunt generate de software-ul din partea serverului a făcut dificil pentru crawler-urile web să evite preluarea de conținut duplicat. Există o multime de combinații de parametri HTTP GET (pe bază de URL), dintre care doar o mică selecție va returna conținut unic. De exemplu, o galerie foto simplă online poate oferi utilizatorilor trei opțiuni, după cum se specifică prin parametrii HTTP GET din URL. Dacă există patru moduri de a sorta imagini, trei opțiuni de dimensiune a imaginii, două formate de fișiere și o opțiune de dezactivare a conținutului furnizat de utilizator, atunci același set de conținut poate fi accesat cu 48 de adrese URL diferite, toate acestea fiind conectate la un site-ul. Această combinație matematică creează o problemă pentru crawler-uri, deoarece acestea trebuie să sorteze prin combinații interminabile de modificări scripturi relativ minore pentru a prelua conținut unic.

# Capitolul II

## Descrierea instrumentelor și tehnologiilor utilizate.

### Instrumentul principal:

Proiectul poate fi efectuat in diferite limbaje de programare.

Cele mai des folosite limbaje de programare in crearea unui web crawler sunt:

-Python;

-Node JS;

-Ruby;

-Php;

-C si C++;

Eu am ales limbajul Python deoarece este un limbaj in care am mai lucrat si sunt familiar cu acesta. Versia folosita este Python 3.8.3

### Instrumente aditionale:

Pentru efectuarea proiectului este necesara folosirea citorva instrumente aditionale, aceastea fiind

-Selenium

-webDriver

Selenium este un framework pentru testarea aplicatiilor web. Fara acesta nu codul python nu va functiona pentru web. Selenium este instalat ca un package in python. Mai exista exista un package pentru python numit Beautiful Soup dar alegerea a ramas la Selenium deoarece este putin mai usor de folosit.

Cu framework pentru testarea instalat trebuie sa conectam scriptul din python cu un browser la alegere. Aceasta se face cu ajutorul unui webdriver pentru browserul care va fi folosit. In cazul meu am folosit browserul Chrome.

# Capitolul III

## Elaborarea aplicației

### Schema programului:

Incarcam

informatia

Mai multa info

Operatie

Accesam info

Salvam

informatia

Inchidem browser

preluam

informatia

Incarcam

informatia

Mai multa info

Operatie

Accesam info

Salvam

informatia

Inchidem browser

preluam

informatia

Alegem una

din 3 optiuni

Introducem informatia

spre cautare

Optiunea

informatia

Optiunea

Alegem valuta

Accesam info

Inchidem browser

preluam

informatia

Alegem suma

### Fragmente semnificative din cod

#### Importurile:

import os

import time

import sys

import webbrowser

from selenium import webdriver

#### Alegerea webdriverului

browser = webdriver.Chrome()

#### Citirea informatiei de la consola

op1 = 0

while op1 == 0:

brwIn = str(input("\nEnter option: "))

print("server = ", brwIn)

if brwIn.isnumeric():

brw = int(brwIn)

if brw > 3 or brw < 1 :

print("[-] no such option")

else:

op1 = 1

else:

print("[-] error with user input")

opSearch = 0

while opSearch == 0:

keyWord = input("\nEnter keyword: ")

if not keyWord:

print("[-] nothing to search?")

else:

search\_term = keyWord

print("searching for " + search\_term)

opSearch = 1

opSearch = 1

#### Inserarea informatiei pe site

browser.get("https://duckduckgo.com")

seach\_box = browser.find\_element\_by\_id("search\_form\_input\_homepage")

seach\_box.send\_keys(search\_term)

seach\_box.submit()

#### Incarcarea a mai multe pagini cu informatie

more = browser.find\_element\_by\_class\_name("result--more\_\_btn")

more.click()

#### Preluarea link-urilor

browser.get("https://duckduckgo.com")

seach\_box = browser.find\_element\_by\_id("search\_form\_input\_homepage")

seach\_box.send\_keys(search\_term)

seach\_box.submit()

time.sleep(1)

linksDuck = browser.find\_elements\_by\_class\_name("result\_\_a")

resultDuck = []

OutputDuck = []

x = 1

for link in linksDuck:

href = link.get\_attribute("href")

resultDuck.append(href)

x = x + 1

no\_deleteDuck = "duckduckgo"

show = [zx for zx in resultDuck if not no\_deleteDuck in zx]

for x in range(len(show)):

print (x+1, ")", show[x])

#### Accesarea unei pagini

for x in range(len(Output)):

print (x+1, ")", Output[x])

print("\nChoose site ")

op2 = 0

while op2 == 0:

ch2In = str(input("Enter nr of site: "))

if ch2In.isnumeric():

ch2 = int(ch2In)

if type(ch2) is not int or ch2 > len(Output) or ch2 < 1:

print("[-] no such option\n")

else:

op2 = 1

else:

print("[-] error with user input\n")

# print(type(ch2), ch2)

print('site to access: ', Output[ch2-1])

browser.get(Output[ch2-1])

input("Press Enter to exit...")

browser.close()

sys.exit(0)

#### Salvarea informatiei

outF = open("out.txt", "w")

for line in Output:

# write line to output file

outF.write(line)

outF.write("\n")

outF.close()

print ("Results are saved sucessfuly in out.txt")

input("Press Enter to exit...")

browser.close()

sys.exit(0)

#### Alegerea pentru a curs valutar

browser.get("https://www.curs.md/ro")

currency = ["mdl","usd","eur","rub","ron","uah"]

print ("\navailable currency\nmdl","usd","eur","rub","ron","uah")

op1 = 0

x = 0

while op1 == 0:

optIn = str(input("\nEnter option: "))

if optIn in currency:

x = x + 1

op1 = 1

else:

print(optIn, "not available")

index = currency.index(optIn) + 1

index = str(index)

path = "//\*[@id=\"suggest-block\"]/div/div[1]/div[1]/div[2]/button[" + index + "]"

val = browser.find\_element\_by\_xpath(path)

val.click()

op1 = 0

while op1 == 0:

brwIn = str(input("\nEnter sum: "))

if brwIn.isnumeric():

brw = int(brwIn)

op1 = 1

else:

print("[-] error with user input")

#### Introducerea sumei in cimp

send\_sum = browser.find\_element\_by\_xpath("//\*[@id=\"suggest-block\"]/div/div[1]/div[1]/div[1]/input")

send\_sum.click()

send\_sum.send\_keys(brw)

#### Preluarea datelor la curs valutar

for x in range(len(currency)):

if x == currency.index(optIn):

asd = 0

else:

temp = ""

temp = str(x+1)

path = "//\*[@id=\"suggest-block\"]/div/div[1]/div[3]/div[3]/button[" + temp + "]"

changeCurency = browser.find\_element\_by\_xpath(path)

changeCurency.click()

zxc = browser.find\_element\_by\_xpath("//\*[@id=\"suggest-block\"]/div/div[1]/div[3]/div[2]/input").get\_attribute('value')

print (brwIn ,optIn , "to", currency[x], "=", zxc)

temp2 = zxc.replace('.', "")

temp2 = temp2.replace(',', '.')

temp2 = float(temp2)

temp3 = float(brwIn)

avg = temp3 / temp2

st = ""

st = ("{:.2f}".format(avg))

string = string + st + " " + optIn + " for 1 " + currency[x] + "\n"

#### Inchiderea browserului si programului

browser.close()

sys.exit(0)

# Concluzie:

In urma efectuarii paracticii tehnologice am obtinut abilitati in domeniu atumatizarii. Scriptul python creat are posibilitatea de a extrage informatie din reteaua Internet cit si de a plasa diferite variabile pe saituri diferite. Deasemenea am intarit cunostintele anterioare din limbajul de programare python lucrind cu liste, tupluri si folosind functiile native acestui prgram.