

Name: بانه بلال ابراهيم , Number: 2826 , Submitted To GitHub:_____

Second Network Programming Homework

Question 1:

Bank ATM Application with TCP Server/Client and Multi-threading Project Description: Build a TCP server and client Bank ATM application using Python. The server should handle multiple client connections simultaneously using multi-threading. The application should allow clients to connect, perform banking operations (such as check balance, deposit, and withdraw), and receive their updated account status upon completion.

Requirements:

- A. The server should be able to handle multiple client connections concurrently.
- B. The server should maintain a set of pre-defined bank accounts with balances.
- C. Each client should connect to the server and authenticate with their account details.
- D. Clients should be able to perform banking operations: check balance, deposit money, and withdraw money.
- E. The server should keep track of the account balances for each client.
- F. At the end of the session, the server should send the final account balance to each client.

Guidelines: • Use Python's socket module without third-party packages. • Implement multi-threading to handle multiple client connections concurrently. • Store the account details and balances on the server side.

Notes: • Write a brief report describing the design choices you made and any challenges faced during implementation. • You can choose to create a TCP Server/Client Bank ATM application or any other appropriate application that fulfills all requirements.

الحل:

هذا الكود يُعالج اتصالات العملاء مع الخادم ويُنفذ الأوامر المصرفية المطلوبة

```
Q2-H2.py net_homework2.py Q1-H2.py x
1 import socket
2 import threading
3
4 # قائمة الحسابات المصرفية مع الأرصدة المبدئية
5 bank_accounts = {
6     '2826': 1000,
7     '5678': 2000,
8     # يمكنك إضافة المزيد من الحسابات هنا ...
9 }
10
11
12 1 usage
13 def handle_client(client_socket, client_address):
14     """
15     تُعالج اتصال العميل وتنفذ الأوامر المطلوبة.
16     :param client_socket: مآخذ العميل
17     :param client_address: عنوان العميل
18     """
19     try:
20         # استقبال تفاصيل الحساب من العميل
21         account_number = client_socket.recv(1024).decode()
22         password = client_socket.recv(1024).decode()
23
24         # التحقق من صحة تفاصيل الحساب
25         if account_number in bank_accounts and password == 'password':
26             client_socket.send(b'Authentication successful!')
27             while True:
28                 # ("deposit" أو "check_balance" مثل)
```

```
Q2-H2.py net_homework2.py Q1-H2.py x
12 def handle_client(client_socket, client_address):
13
14     while True:
15         # ("deposit" أو "check_balance" مثل) استقبال الأمر من العميل
16         command = client_socket.recv(1024).decode()
17         if command == 'check_balance':
18             balance = bank_accounts[account_number]
19             client_socket.send(f'Your balance: {balance}'.encode())
20         elif command == 'deposit':
21             amount = float(client_socket.recv(1024).decode())
22             bank_accounts[account_number] += amount
23             client_socket.send(b'Deposit successful!')
24             # يمكنك إضافة المزيد من الأوامر هنا ...
25         else:
26             client_socket.send(b'Authentication failed!')
27     except Exception as e:
28         print(f"Error handling client {client_address}: {e}")
29     finally:
30         client_socket.close()
31
32
33
34
35
36
37
38
39
40
41
42
43
44
```

```

44
45 1 usage
46 def main():
47     server_ip = '127.0.0.1'
48     server_port = 12345
49
50     server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
51     server_socket.bind((server_ip, server_port))
52     server_socket.listen(5)
53     print(f"Listening on {server_ip}:{server_port}...")
54
55     while True:
56         client_socket, client_address = server_socket.accept()
57         print(f"Accepted connection from {client_address}")
58         client_thread = threading.Thread(target=handle_client, args=(client_socket, client_address))
59         client_thread.start()
60
61 > if __name__ == "__main__":
62     main()
63

```

خطوات عمل الكود:

1. تعريف قائمة الحسابات المصرفية:

- تم تعريف قائمة bank_accounts التي تحتوي على أرقام حسابات مصرفية مع الأرصدة المبدئية لكل حساب.

2. دالة: `handle_client`

- هذه الدالة تُعالج اتصال العميل وتنفذ الأوامر المطلوبة.
- تستقبل مأخذ العميل (client_socket) وعنوان العميل (client_address).
- تبدأ بقراءة تفاصيل الحساب من العميل (رقم الحساب وكلمة المرور).
- تقوم بالتحقق من صحة تفاصيل الحساب:
- إذا كان رقم الحساب موجودًا في قائمة bank_accounts وكانت كلمة المرور هي "password"، يتم إرسال رسالة "Authentication successful!" إلى العميل.
- إلا إذا فشل التحقق، يتم إرسال رسالة "Authentication failed!"
- بعد التحقق، يستمر في استقبال الأوامر من العميل (مثل "check_balance" أو "deposit") وينفذها.

- إذا كانت الأمر "check_balance" ، يُظهر رصيد الحساب للعميل.
- إذا كانت الأمر "deposit" ، يتم إيداع مبلغ معين في الحساب.
- في حالة حدوث أي استثناء، يتم طباعة رسالة خطأ.

3. الدالة الرئيسية: `main`

- تعيد تعريف معلومات الخادم (عنوان IP ومنفذ الاستماع).
- تقوم بإنشاء مأخذ الخادم وتبدأ في الاستماع للاتصالات الواردة.
- عند قبول اتصال من عميل، يتم إنشاء موضوع جديد لمعالجة العميل باستخدام دالة `handle_client`.

ويكون الخرج عند التنفيذ:

```

Run Q1-H2 x
C:\Users\BANA.IB\anaconda3\envs\pose_estimation\python.exe C:\Users\BANA.IB\PycharmProjects\pythonProject\ggg\Q1-H2.py
Listening on 127.0.0.1:12345...

```

Question 2:

Simple Website Project with Python Flask Framework (you have choice to use Django or any Other Deferent Useful Python Project “from provide Project Links”)

Create a simple website with multiple pages using Flask, HTML, CSS, and Bootstrap. The website should demonstrate your understanding of web design principles .

Requirements :

- G. Set up a local web server using XAMPP, IIS, or Python's built-in server (using Flask) .
- H. Apply CSS and Bootstrap to style the website and make it visually appealing .
- I. Ensure that the website is responsive and displays correctly on different screen sizes
- J. Implement basic server-side functionality using Flask to handle website features

بالاستفادة من الرابط التالي "المرفق بالسؤال"

Python Projects – Beginner to Advanced

<https://www.geeksforgeeks.org/python-projects-beginner-to-advanced/>

Program to extract frames using OpenCV:

الكود التالي يقوم باستخراج الإطارات من مقطع فيديو باستخدام مكتبة OpenCV في لغة Python :

```
Q2-H2.py x net_homework2.py Q1-H2.py
1 import cv2
2
3 1 usage
4 def FrameCapture(path):
5     vidObj = cv2.VideoCapture(path)
6     count = 0
7
8     while True:
9         success, image = vidObj.read()
10        if not success:
11            break
12
13        cv2.imwrite("frame%d.jpg" % count, image)
14        count += 1
15
16        print('تم قراءة إطار جديد:', success)
17
18        vidObj.release()
19
20 if __name__ == '__main__':
21     video_path = "C:\\Users\\BANA.IB\\Desktop\\Bana Documents\\Short-Video-Maker_Lead.mp4"
22     FrameCapture(video_path)
23
24
25
26
```

خطوات الكود:

1. قراءة الفيديو:

- يتم استخدام الدالة cv2.VideoCapture() لفتح ملف الفيديو. قمت تمرير مسار الملف كوسيط لهذه الدالة.

- في الكود الخاص بي تم استخدام المسار "C:\\Users\\BANA.IB\\Desktop\\Bana Documents\\your_video.mp4" كمثال. (ملاحظة: يجب أن تستبدل "your_video.mp4" بالمسار الفعلي لملف الفيديو الذي ترغب في استخدامه.)

2. استخراج الإطارات:

- يتم استخدام حلقة while لقراءة الإطارات من الفيديو.
- الدالة `captured.read()` تُقرأ إطارًا من الفيديو في كل تكرار للحلقة. إذا تمت قراءة الإطار بنجاح، يكون `ret` قيمة `True`.
- يتم حفظ الإطارات كملفات صور باستخدام `cv2.imwrite()`.

وتكون نتيجة تنفيذ الكود:

[illegible]