2024

**إشراف الدكتور المهندس: مهند عيسى**

**السنة الخامسة اتصالات**

Second Network Programming Homework

إعداد الطلاب: سيزار أحمد شعبان 2897 جعفر سليمان علي 2621

**Question 1:** Bank ATM Application with TCP Server/Client and Multi-threading

برنامج السيرفر أولاً:

import socket

import threading

ACCOUNTS = {

    "Sezar1": 5000,

    "Jafar2": 3000}

def handle\_client(conn, addr):

    print(f"New connection from {addr}")

    try:

        account\_id = conn.recv(1024).decode()

        if account\_id not in ACCOUNTS:

            conn.sendall("Invalid account ID".encode())

            return

        balance = ACCOUNTS[account\_id]

        conn.sendall(f"Welcome to the bank ATM! Your current balance is: {balance}".encode())

        while True:

            choice = conn.recv(1024).decode()

            if choice == "1":

                conn.sendall(f"Your balance is: {balance}".encode())

            elif choice == "2":

                amount = int(conn.recv(1024).decode())

                balance += amount

                ACCOUNTS[account\_id] = balance

                conn.sendall(f"Deposit successful. Your new balance is: {balance}".encode())

            elif choice == "3":

                amount = int(conn.recv(1024).decode())

                if amount > balance:

                    conn.sendall("Insufficient funds".encode())

                else:

                    balance -= amount

                    ACCOUNTS[account\_id] = balance

                    conn.sendall(f"Withdrawal successful. Your new balance is: {balance}".encode())

            elif choice == "4":

                conn.sendall(f"Thank you for using the bank ATM. Your final balance is: {balance}".encode())

                break

            else:

                conn.sendall("Invalid choice".encode())

    except Exception as e:

        print(f"Error: {e}")

    finally:

        conn.close()

def main():

    server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    server\_socket.bind(("0.0.0.0", 8000))

    server\_socket.listen(5)

    print("Server is listening on localhost:8000")

    while True:

        conn, addr = server\_socket.accept()

        client\_thread = threading.Thread(target=handle\_client, args=(conn, addr))

        client\_thread.start()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

آلية بناء وإنشائنا للكود:

1. قمنا باستيراد المكتبتين socket وthreading لبناء المقابس وحتى نجعل السيرفر يعمل بنفس الوقت مع عدة مستخدمين وتخديمهم جميعاً.
2. تم بناء وإنشاء الحسابات بتعريف القاموس ACCOUNTS الذي يحوي اسم صاحب الحساب والمبلغ المالي الذي يملكه حالياً.
3. السيرفر يعمل على العنوان: 0.0.0.0 وتم ربطه مع رقم المنفذ 8000، طبعاً السيرفر يتعامل مع أي عنوان IP ضمن التطبيق.
4. عن طريق اشتقاق الغرض client\_thread من الصنف Thread نستطيع التعامل مع العملاء بنفس الوقت، وذلك بإمرار التابع الذي يتعامل مع اتصالات العملاء وبارامتراته.
5. التابع handle\_client(conn, addr) يأخذ بارمترين هما: عنوان العميل الذي يتصل مع السيرفر ومقبس العميل.
6. يطبع التابع رسالة إضافة الاتصال مع العميل الجديد ويطبع بجانبها عنوانه.
7. نقوم باستقبال اسم صاحب الحساب عبر المتحول account\_id والتحقق منه إن كان موجود أم لا.
8. لأجل كل عميل هناك العديد من الخيارات للعمليات مثل: عرض إجمالي النقود في الحساب، أو إضافة مبلغ مالي على الحساب، أو سحب مبلغ.

كود العميل الأول:

import socket

def main():

    client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    client\_socket.connect(("127.0.0.1", 8000))

    print("Enter your account ID:")

    account\_id = input()

    client\_socket.sendall(account\_id.encode())

    response = client\_socket.recv(1024).decode()

    print(response)

    while True:

        print("\nChoose an option:")

        print("1. Check balance")

        print("2. Deposit money")

        print("3. Withdraw money")

        print("4. Exit")

        choice = input()

        client\_socket.sendall(choice.encode())

        if choice == "1":

            balance = client\_socket.recv(1024).decode()

            print(balance)

        elif choice == "2":

            print("Enter the amount to deposit:")

            amount = int(input())

            client\_socket.sendall(str(amount).encode())

            response = client\_socket.recv(1024).decode()

            print(response)

        elif choice == "3":

            print("Enter the amount to withdraw:")

            amount = int(input())

            client\_socket.sendall(str(amount).encode())

            response = client\_socket.recv(1024).decode()

            print(response)

        elif choice == "4":

            response = client\_socket.recv(1024).decode()

            print(response)

            break

        else:

            response = client\_socket.recv(1024).decode()

            print(response)

    client\_socket.close()

if \_\_name\_\_ == "\_\_main\_\_":

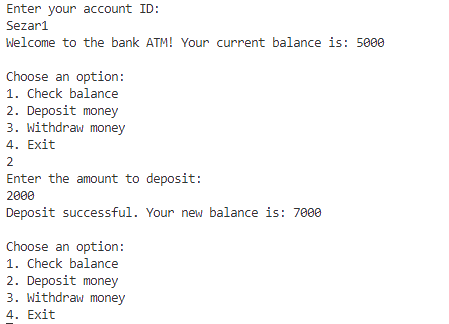
    main()

الزبون (العميل) يقوم بإجراءات محددة بعد إدخال اسمه، وهي توافق أرقام الخيارات 1،2،3،4.

ويقوم بإرسال هذه القيم للسيرفر لمعالجتها.

التنفيذ:





**Question 2: Chat Text Analyzer:**

import re

def analyze\_text(file\_path):

    with open(file\_path, 'r') as file:

        text = file.read()

    words = text.split()

    word\_count = len(words)

    char\_count = len(text)

    sentences = re.split(r'[.!?]+', text)

    sentence\_count = len(sentences)

    return word\_count, char\_count, sentence\_count

file\_path = 'text\_file.txt'

word\_count, char\_count, sentence\_count = analyze\_text(file\_path)

print(f"Word count: {word\_count}")

print(f"Character count: {char\_count}")

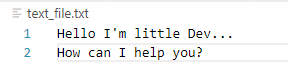
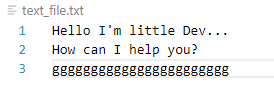
print(f"Sentence count: {sentence\_count}")

هذا البرنامج يقوم بتحليل العبارات والجمل الموجودة في ملف TXT، ويقوم بطباعة عدد الكلمات، وعدد المحارف...

1. import re: يتم استيراد المكتبة re التي توفر وظائف لمطابقة ومعالجة التعابير العادية.
2. def analyze\_text(file\_path): تعريف الدالة analyze\_text التي تقوم بتحليل النص الموجود في ملف نصي معين بواسطة.
3. with open(file\_path, 'r') as file:
4. text = file.read(): يتم فتح الملف الموجود في file\_path في وضع القراءة ('r') وقراءة نصه وحفظه في المتغير text.
5. words = text.split()

يتم تقسيم النص إلى كلمات وحفظها في قائمة من خلال استخدام الفراغ كفاصل.

1. word\_count = len(words): يتم حساب عدد الكلمات الكلي في النص من خلال عدد العناصر في القائمة words.
2. char\_count = len(text): تم حساب عدد الأحرف في النص الكلي من خلال عدد الأحرف في المتغير text.
3. sentences = re.split(r'[.!?]+', text): تقسيم النص إلى جمل من خلال استخدام re.split وتحديد النمط [.!?]+ الذي يعني التقسيم بناءً على العلامات الترقيمية.
4. sentence\_count = len(sentences): يتم حساب عدد الجمل في النص من خلال عدد العناصر في القائمة sentences.
5. return word\_count, char\_count, sentence\_count: الدالة ترجع عدد الكلمات، عدد الأحرف، وعدد الجمل.
6. file\_path = 'text\_file.txt': تعيين المسار للملف النصي المراد تحليله.
7. word\_count, char\_count, sentence\_count = analyze\_text(file\_path): استدعاء الدالة analyze\_text وتخزين قيمة الكلمات، الأحرف، والجمل في المتغيرات المقابلة.
8. print(f"Word count: {word\_count}"), print(f"Character count: {char\_count}"), print(f"Sentence count: {sentence\_count}"): يتم طباعة عدد الكلمات، عدد الأحرف، وعدد الجمل بواسطة تنسيق السلسلة f-string في كل تمريرة.



Word count: 10

Character count: 67

Sentence count: 3