



جامعة تشرين  
كلية الهندسة الميكانيكية والكهربائية  
قسم الاتصالات والإلكترونيات  
مقرر الاتصالات الفضائية  
السنة الخامسة

إعداد الطلاب :

هناء كثم بلول 2945	هناء زهير مرجان 2804
--------------------	----------------------

إشراف الدكتور:

مهند عيسى

برمجة شبكات

## Homework2

العام الدراسي :

2023 -2024

## Contents

<b>Question 1:</b> .....	3
<b>Question 2:</b> .....	9

## Question 1:

### 1. الخادم:

- إنشاء خادم TCP يستمع لاتصالات العميل الواردة.
- تنفيذ تعدد العمليات للتعامل مع اتصالات العملاء المتعددة في وقت واحد.
- الاحتفاظ بقاموس أو قاعدة بيانات لتخزين الحسابات المصرفية المحددة مسبقاً وأرصدها.
- التحقق من العملاء وإجراء العمليات المصرفية المطلوبة.
- تحديث أرصدة الحساب وإرسال الرصيد النهائي للعميل.

### 2. العميل:

- إنشاء عميل TCP يتصل بالخادم.
- التحقق من العميل بتفاصيل حسابه.
- السماح للعميل بالقيام بالعمليات المصرفية (التحقق من الرصيد، الإيداع، السحب).
- استلام رصيد الحساب الختامي من الخادم وعرضه للمستخدم.

## خطوات التنفيذ:

### 1. جانب الخادم:

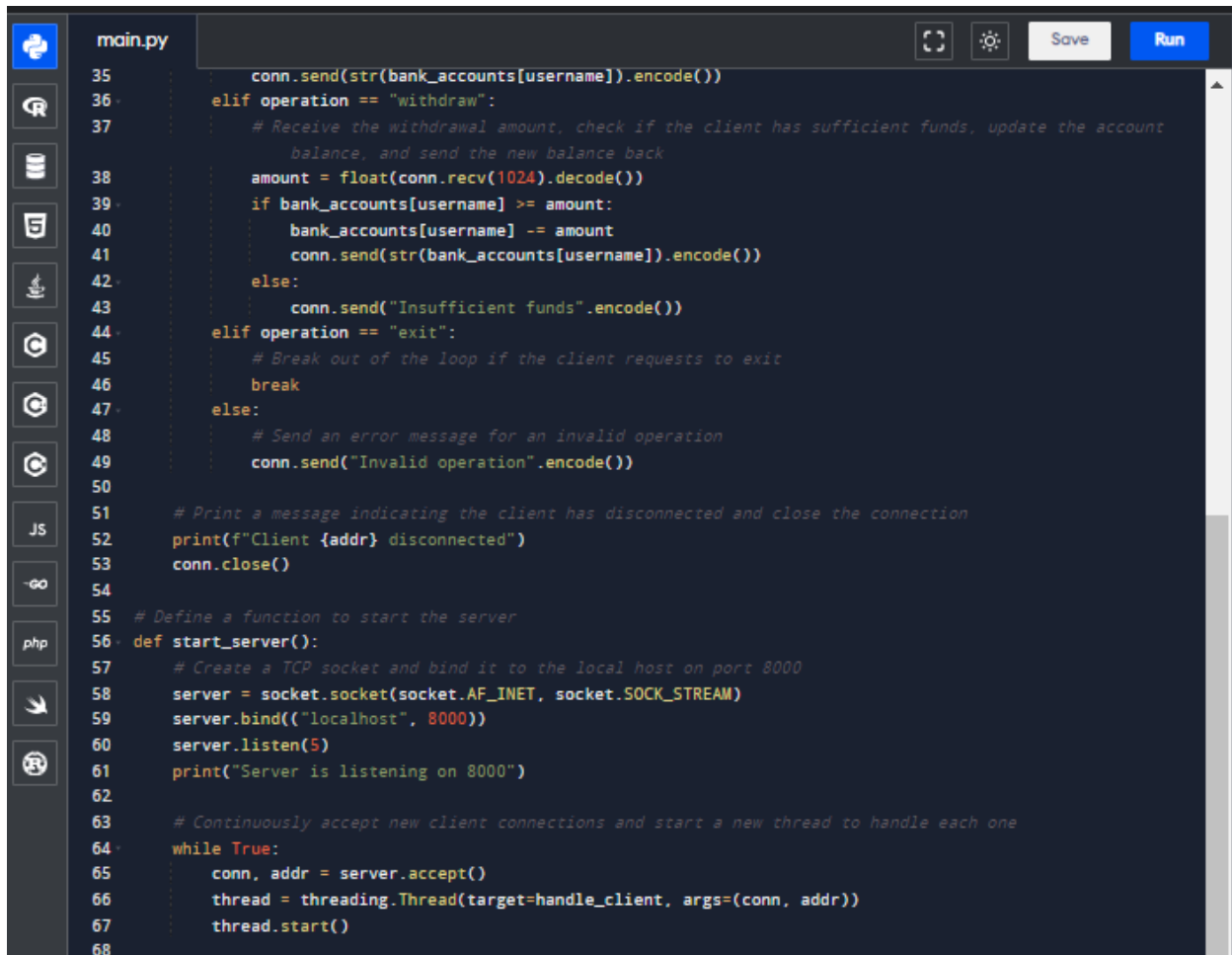
- استيراد الوحدات الضرورية (على سبيل المثال، `socket`، `threading`).
- إنشاء مقبس خادم TCP وربطه بعنوان ومنفذ محدد.
- تنفيذ وظيفة للتعامل مع اتصالات العملاء والعمليات المصرفية.
- استخدام قاموساً أو قاعدة بيانات لتخزين الحسابات المصرفية المحددة مسبقاً وأرصدها.
- التحقق من العملاء وإجراء العمليات المصرفية المطلوبة.
- تحديث أرصدة الحساب وإرسال الرصيد النهائي للعميل.
- استخدام مؤشرات الترابط المتعددة للتعامل مع اتصالات العملاء المتعددة في وقت واحد.

## 2. جانب العميل:

- استيراد الوحدات الضرورية (على سبيل المثال، "socket").
- إنشاء socket عميل TCP والاتصال بالخادم.
- التحقق من العميل بتفاصيل حسابه.
- السماح للعميل بالقيام بالعمليات المصرفية (التحقق من الرصيد، الإيداع، السحب).
- استلام رصيد الحساب الختامي من الخادم وعرضه للمستخدم.

### CODE:

```
main.py
1 # Import the necessary modules
2 #Server-side
3 import socket
4 import threading
5
6 # Create a dictionary to store bank accounts and their balances
7 bank_accounts = {
8     "user1": 2000.0,
9     "user2": 2500.0,
10    "user3": 500.0
11 }
12
13 # Define a function to handle client connections
14 def handle_client(conn, addr):
15     # Print a message indicating a new connection
16     print(f"New connection from {addr}")
17
18     # Authenticate the client
19     username = conn.recv(1024).decode()
20     password = conn.recv(1024).decode()
21     # Implement authentication logic here
22
23     # Perform banking operations
24     while True:
25         # Receive the client's requested operation
26         operation = conn.recv(1024).decode()
27         if operation == "balance":
28             # Retrieve the client's account balance and send it back
29             balance = bank_accounts[username]
30             conn.send(str(balance).encode())
31         elif operation == "deposit":
32             # Receive the deposit amount, update the account balance, and send the new balance back
33             amount = float(conn.recv(1024).decode())
34             bank_accounts[username] += amount
35             conn.send(str(bank_accounts[username]).encode())
36         elif operation == "withdraw":
37             # Receive the withdrawal amount, update the account balance, and send the new balance back
38             amount = float(conn.recv(1024).decode())
39             bank_accounts[username] -= amount
40             conn.send(str(bank_accounts[username]).encode())
41         else:
42             conn.send("Invalid operation".encode())
43
44     # Close the connection
45     conn.close()
```



```
35     conn.send(str(bank_accounts[username]).encode())
36 elif operation == "withdraw":
37     # Receive the withdrawal amount, check if the client has sufficient funds, update the account
    balance, and send the new balance back
38     amount = float(conn.recv(1024).decode())
39     if bank_accounts[username] >= amount:
40         bank_accounts[username] -= amount
41         conn.send(str(bank_accounts[username]).encode())
42     else:
43         conn.send("Insufficient funds".encode())
44 elif operation == "exit":
45     # Break out of the loop if the client requests to exit
46     break
47 else:
48     # Send an error message for an invalid operation
49     conn.send("Invalid operation".encode())
50
51 # Print a message indicating the client has disconnected and close the connection
52 print(f"Client {addr} disconnected")
53 conn.close()
54
55 # Define a function to start the server
56 def start_server():
57     # Create a TCP socket and bind it to the local host on port 8000
58     server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
59     server.bind(("localhost", 8000))
60     server.listen(5)
61     print("Server is listening on 8000")
62
63     # Continuously accept new client connections and start a new thread to handle each one
64     while True:
65         conn, addr = server.accept()
66         thread = threading.Thread(target=handle_client, args=(conn, addr))
67         thread.start()
68
```



```
main.py  [ ] [ ] Save Run

1  # Import the necessary module
2  #Client-side
3  import socket
4
5  # Define a function to connect to the server
6  def connect_to_server():
7      # Create a TCP socket and connect it to the server running on localhost at port 8000
8      client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9      client.connect(("localhost", 8000))
10     return client
11
12  # Define a function to authenticate the client
13  def authenticate(client):
14      # Prompt the user for their username and password, and send them to the server
15      username = input("Enter your username: ")
16      password = input("Enter your password: ")
17      client.send(username.encode())
18      client.send(password.encode())
19
20  # Define a function to perform banking operations
21  def perform_banking_operations(client):
22      # Continuously prompt the user to choose an operation until they choose to exit
23      while True:
24          print("Choose an operation:")
25          print("1. Check balance")
26          print("2. Deposit")
27          print("3. Withdraw")
28          print("4. Exit")
29          choice = input("Enter your choice (1-4): ")
30
31          # Perform the selected operation and display the result
32          if choice == "1":
```

```
main.py

31 # Perform the selected operation and display the result
32 if choice == "1":
33     client.send("balance".encode())
34     balance = float(client.recv(1024).decode())
35     print(f"Your balance is: {balance}")
36 elif choice == "2":
37     amount = float(input("Enter the deposit amount: "))
38     client.send("deposit".encode())
39     client.send(str(amount).encode())
40     new_balance = float(client.recv(1024).decode())
41     print(f"Your new balance is: {new_balance}")
42 elif choice == "3":
43     amount = float(input("Enter the withdrawal amount: "))
44     client.send("withdraw".encode())
45     client.send(str(amount).encode())
46     response = client.recv(1024).decode()
47     if response == "Insufficient funds":
48         print("Insufficient funds")
49     else:
50         new_balance = float(response)
51         print(f"Your new balance is: {new_balance}")
52 elif choice == "4":
53     # Send the 'exit' command to the server and display the final balance
54     client.send("exit".encode())
55     final_balance = float(client.recv(1024).decode())
56     print(f"Final balance: {final_balance}")
57     break
58 else:
59     print("Invalid choice. Please try again.")
60
61 # Execute the main program when the script is run
```

```
57     break
58 else:
59     print("Invalid choice. Please try again.")
60
61 # Execute the main program when the script is run
62 if __name__ == "__main__":
63     # Connect to the server, authenticate the client, and perform banking operations
64     client = connect_to_server()
65     authenticate(client)
66     perform_banking_operations(client)
```



## Question 2:

```
main.py
1 from flask import Flask, render_template
2
3 app = Flask(__name__)
4
5 @app.route('/')
6 def index():
7     return render_template('index.html')
8
9 @app.route('/about')
10 def about():
11     return render_template('about.html')
12
13 @app.route('/contact')
14 def contact():
15     return render_template('contact.html')
16
17 if __name__ == '__main__':
18     app.run(debug=True)
```

عند تشغيل الكود والدخول الى المتصفح والدخول الى localhost

اتصل بنا من نحن الصفحة الرئيسية

مرحبًا بكم في موقعنا الإلكتروني

هذا هو محتوى الصفحة الرئيسية

كود html الذي يظهر الصفحة الرئيسية للموقع:

```
HTML 13 unsaved changes X
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>موقعي الإلكتروني</title>
7 <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10 <header>
11 <nav>
12 <ul>
13 <li><a href="#">الصفحة الرئيسية</a></li>
14 <li><a href="#">من نحن</a></li>
15 <li><a href="#">اتصل بنا</a></li>
16 </ul>
17 </nav>
18 </header>
19
20 <main>
21 <section>
22 <h1>مرحبًا بكم في موقعنا الإلكتروني</h1>
23 <p>هذا هو محتوى الصفحة الرئيسية</p>
```



HTML

14 unsaved changes X



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>موقعي الإلكتروني</title>
7     <link rel="stylesheet" href="{{ url_for('static', filename='css/styles.css') }}">
8 </head>
9 <body>
10 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
11     <a class="navbar-brand" href="{{ url_for('index') }}">
12         
14         موقعي الإلكتروني
15     </a>
16     <button class="navbar-toggler" type="button" data-toggle="collapse" data-
17     target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
18     navigation">
19         <span class="navbar-toggler-icon"></span>
20     </button>
21     <div class="collapse navbar-collapse" id="navbarNav">
22         <ul class="navbar-nav">
23             <li class="nav-item">
```

HTML14 unsaved changes

```
navigation">
16     <span class="navbar-toggler-icon"></span>
17 </button>
18 * <div class="collapse navbar-collapse" id="navbarNav">
19 *     <ul class="navbar-nav">
20 *         <li class="nav-item">
21 *             <a class="nav-link" href="{{ url_for('index') }}">الصفحة الرئيسية</a>
22             </li>
23 *         <li class="nav-item">
24 *             <a class="nav-link" href="{{ url_for('about') }}">من نحن</a>
25             </li>
26 *         <li class="nav-item">
27 *             <a class="nav-link" href="{{ url_for('contact') }}">اتصل بنا</a>
28             </li>
29         </ul>
30     </div>
31 </nav>
32
33 * <div class="container my-5">
34     {% block content %}
35     {% endblock %}
36 </div>
37
```