Syrian Arab Republic

Lattakia - Tishreen University

Department of Communication and electrical engineering

5th , Network Programming : Homework No2

الجمهورية العربية السورية

اللاذقية -جامعـــة تشريـــن

كلية الهندسة الكهربائية والميكانيكية

قسم هندسة الاتصالات والالكترونيات

السنة الخامسة: وظيفة2 برمجة شبكات

لما مالك حيدر 2795
دانيال جعفر رحمون 2836
لجين أحمد مهنا 2894

السؤال الأول

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

كود السيرفر

```
import socket
import threading

# Define the server parameters
HOST = '127.0.0.1'
PORT = 65432

# Define the account details
accounts = {
    'lama': {'password': '2795', 'balance': 1000.0},
    'danial': {'password': '2836', 'balance': 500.0},
    'lujain': {'password': '2894', 'balance': 2000.0}
}

# Function to handle a client connection
def handle_client(conn, addr):
    print(f'New connection from {addr}')

    # Authenticate the client
    while True:
        username = conn.recv(1024).decode()
        password = conn.recv(1024).decode()
        if username in accounts and accounts[username]['password'] ==
password:
            conn.sendall(b'Authentication successful')
            break
        else:
            conn.sendall(b'Invalid username or password. Please try again.')

    # Handle banking operations
    while True:
        operation = conn.recv(1024).decode().strip()
        if operation == 'balance':
            balance = accounts[username]['balance']
            conn.sendall(f'Your current balance is: {balance:.2f}'.encode())
        elif operation == 'deposit':
```

```
            conn.sendall(b'Enter the amount to deposit:')
            amount = float(conn.recv(1024).decode().strip())
            accounts[username]['balance'] += amount
            conn.sendall(f'Deposit successful. Your new balance is:
{accounts[username]["balance"]:.2f}'.encode())
        elif operation == 'withdraw':
            conn.sendall(b'Enter the amount to withdraw:')
            amount = float(conn.recv(1024).decode().strip())
            if amount <= accounts[username]['balance']:
                accounts[username]['balance'] -= amount
                conn.sendall(f'Withdrawal successful. Your new balance is:
{accounts[username]["balance"]:.2f}'.encode())
            else:
                conn.sendall(b'Insufficient funds.')
        elif operation == 'quit':
            conn.sendall(f'Your final balance is:
{accounts[username]["balance"]:.2f}'.encode())
            break
        else:
            conn.sendall(b'Invalid operation. Please try again.')

    print(f'Closing connection with {addr}')
    conn.close()

# Start the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen()
    print(f'Server listening on {HOST}:{PORT}')

    while True:
        conn, addr = s.accept()
        threading.Thread(target=handle_client, args=(conn, addr)).start()
```

هذا الكود يمثل خادم (server) لنظام بنكي بسيط. الخادم يتواصل مع العملاء عبر اتصال شبكي باستخدام بروتوكول TCP. يتم تعريف المتغيرات الأساسية للخادم مثل عنوان IP (HOST) ورقم المنفذ (PORT) التي سيستخدمها الخادم. كما تم تعريف قاعدة بيانات حسابات المستخدمين والمعلومات المتعلقة بها مثل كلمات المرور والأرصدة المالية.

الوظيفة الرئيسية للخادم هي "handle_client"، والتي تتعامل مع اتصال العميل. بعد إنشاء الاتصال، يقوم العميل بمصادقة نفسه من خلال إرسال اسم المستخدم وكلمة المرور. بمجرد التحقق من المصادقة، يتم السماح للعميل بإجراء عمليات مصرفية مثل الاستعلام عن الرصيد والإيداع والسحب. يقوم الخادم بتحديث أرصدة المستخدمين حسب العمليات التي يقوم بها العميل. عندما يختار العميل إنهاء الاتصال، يتم إرسال الرصيد النهائي للعميل وإغلاق الاتصال.

كود الكلاينت

```
import socket

# Define the server parameters
HOST = '127.0.0.1'
PORT = 65432

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))

    # Authenticate the user
    username = input('Enter your username: ')
    s.sendall(username.encode())
    password = input('Enter your password: ')
    s.sendall(password.encode())
    msg = s.recv(1024).decode()
    if msg == 'Authentication successful':
        print('Authentication successful')
    else:
        print('Authentication failed. Exiting...')
        exit()

    # Perform banking operations
    while True:
        operation = input('Enter "balance", "deposit", "withdraw", or "quit":
')
        s.sendall(operation.encode())
        if operation == 'balance':
            msg = s.recv(1024).decode()
            print(msg)
        if operation == 'deposit':
            msg = s.rlujainecv(1024).decode()
            print(msg)
            a=input()
            s.sendall(a.encode())
            msg = s.recv(1024).decode()
            print(msg)
        if operation == 'withdraw':
            msg = s.recv(1024).decode()
            print(msg)
            a = input()
            s.sendall(a.encode())
            msg = s.recv(1024).decode()
            print(msg)
        if operation == 'quit':
            break
```

العميل يقوم بالاتصال بالخادم باستخدام عنوان (HOST)ورقم المنفذ (PORT) المعرفين مسبقًا. بعد الاتصال، يتم طلب اسم المستخدم وكلمة المرور من العميل، والتحقق من صحتهما من قبل الخادم. إذا كانت

المصادقة ناجحة، يتم السماح للعميل بالقيام بعمليات مصرفية مثل الاستعلام عن الرصيد والإيداع والسحب. يقوم العميل بإرسال الطلبات إلى الخادم وعرض الردود المستلمة. عندما يختار العميل إنهاء الاتصال، يتم إرسال الرصيد النهائي للعميل من قبل الخادم قبل إغلاق الاتصال.

الخرج:

يوضح الشكل التالي عملية تشغيل السيرفر واتصال العملاء بنفس الوقت:

السيرفر:

```
Server listening on 127.0.0.1:65432
New connection from ('127.0.0.1', 50413)
Closing connection with ('127.0.0.1', 50413)
New connection from ('127.0.0.1', 50592)
New connection from ('127.0.0.1', 50636)
New connection from ('127.0.0.1', 50725)
Closing connection with ('127.0.0.1', 50592)
```

العملاء:

```
Enter your username: danial
Enter your password: 2836
Authentication successful
Enter "balance", "deposit", "withdraw", or "quit": balance
Your current balance is: 500.00
Enter "balance", "deposit", "withdraw", or "quit": deposit
Enter the amount to deposit:
555
Deposit successful. Your new balance is: 1055.00
Enter "balance", "deposit", "withdraw", or "quit": withdraw
Enter the amount to withdraw:
333
Withdrawal successful. Your new balance is: 722.00
Enter "balance", "deposit", "withdraw", or "quit": quit

Process finished with exit code 0
```

```
Enter your username: lujain
Enter your password: 2894
Authentication successful
Enter "balance", "deposit", "withdraw", or "quit":
```

```
Enter your username: lama
Enter your password: 2795
Authentication successful
Enter "balance", "deposit", "withdraw", or "quit":
```

السؤال الثاني

Question 2: Simple Website Project with Python Flask Framework

كود السيرفر

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/page1.html')
def page1():
    return render_template('page1.html')

@app.route('/page2.html')
def page2():
    return render_template('page2.html')

@app.route('/page3.html')
def page3():
    return render_template('page3.html')

if __name__ == '__main__':
    app.run(debug=True,port=1234)
```

أولاً يتم استيراد Flask و render_template من مكتبة Flask ثم يتم إنشاء تطبيق Flask اسمه app

ثم يتم تعريف مسارات الصفحات باستخدام المنسق @app.route() ، حيث يتم تعريف الصفحة الرئيسية / وصفحات page1.html و page2.html وpage3.html

تحت كل عنوان صفحة، يتم استدعاء تابع يحوي التابع render_template() الذي يقوم بإرجاع صفحة HTMLالمحددة يتم وضع ملفات الصفحات الخاصة بكل مسار في مجلدtemplates

يتم تشغيل التطبيق عن طريق القيام بالتحقق من أن المتغير name هو main ومن ثم يتم تشغيل التطبيق على السيرفر الذي يعمل على الرقم المحدد للمنفذ1234 (port) مع تمكين وضع التصحيح الخطأ (debug mode).

نفتح المتصفح ونطلب العنوان http://127.0.0.1:1234/ فيتم تحميل الصفحة index.html بالشكل:

الصفحة :page1

الصفحة :page2

الصفحة :page3

ملف html الخاص بالصفحة :index

```
<!DOCTYPE html>
<html lang="ar" dir="rtl">
<head>
    <meta charset="UTF-8">
    <title>الـرئـيـسية</title>
    <style>
       body {
          background-color: #00a3f2;
          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
```

```
            font-size: 18px;
            text-align: center;
            color: #444;
            margin: 0;
            padding: 0;
        }
        h1 {
            color: #1eff00;
            margin-top: 50px;
            font-size: 36px;
            font-weight: bold;
            text-transform: uppercase;
        }
        p {
            margin-top: 50px;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
        }
        ul {
            list-style-type: none;
            padding: 0;
            margin-top: 50px;
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            align-items: center;
        }
        li a {
            display: block;
            background-color: #100010;
            color: #fff;
            padding: 20px 40px;
            margin: 10px;
            border-radius: 50px;
            text-decoration: none;
            transition: background-color .3s;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
            letter-spacing: 2px;
        }

    </style>
</head>
<body>
    <h1>حيدر مالك لما</h1>
    <p>الرقم: 2795</p>
    <h1>رحمون جعفر دانيال</h1>
<p>الرقم: 2836</p>
    <h1>مهنا أحمد لجين</h1>
<p>الرقم: 2894</p>
    <ul>
        <li><a href="page1.html">أكثر معلومات</a></li>
        <li><a href="page2.html">السؤال الأول</a></li>
        <li><a href="page3.html">السؤال الثاني</a></li>
```

```
    </ul>
</body>
</html>
```

- <!DOCTYPE html> تعريف نوع المستند كـ HTML5.

- <html lang="ar" dir="rtl">تفاصيل اللغة والاتجاه الذي سيتم عرض الصفحة به.

- <head> يحتوي على المعلومات الضرورية للصفحة والتي لا تظهر في الصفحة مثل العنوان وترميز المحارف وأنماط css.

- <meta charset="UTF-8"> تعيين ترميز المحارف لـ UTF-8.

- `<title>الرئيسية</title>` عنوان صفحة الويب.

- <style> يستخدم لتحديد أنماط صفحة الويب.

- body تعيين الخصائص الأساسية للجسم الذي يحتوي على المحتوى.

- h1 تعيين عنصر العنوان.

- p تعيين عنصر الفقرة.

- ul تعيين قائمة غير مرتبة.

- li a تعيين عنصر رابط داخل عنصر القائمة.

- href تحديد الرابط الذي سيتم الانتقال إليه عند النقر على الرابط.

يتم عرض اسمائنا وارقامنا الجامعية في الصفحة

ملف html الخاص بالصفحة :page 1

```
<!DOCTYPE html>
<html lang="ar" dir="rtl">
<head>
    <meta charset="UTF-8">
    <title>أكثر معلومات</title>
    <style>
       body {
          background-color: #00a3f2;
          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
          font-size: 18px;
          text-align: center;
```

```
            color: #444;
            margin: 0;
            padding: 0;
        }
        h1 {
            color: #1eff00;
            margin-top: 50px;
            font-size: 36px;
            font-weight: bold;
            text-transform: uppercase;
        }
        p {
            margin-top: 50px;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
        }
        ul {
            list-style-type: none;
            padding: 0;
            margin-top: 50px;
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            align-items: center;
        }
        li a {
            display: block;
            background-color: #100010;
            color: #fff;
            padding: 20px 40px;
            margin: 10px;
            border-radius: 50px;
            text-decoration: none;
            transition: background-color .3s;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
            letter-spacing: 2px;
        }

    </style>
</head>
<body>
    <h1>المعلومات من مزيد</h1>
    <p>الصفحة عنوان: "أكثر معلومات"</p>
    <p>بسيط مثال</p>
    <ul>
        <li><a href="/">الرئيسية إلى العودة</a></li>
        <li><a href="page2.html">الأول السؤال</a></li>
        <li><a href="page3.html">الثاني السؤال</a></li>
    </ul>
</body>
</html>
```

ملف html الخاص بالصفحة page 2:

```
<!DOCTYPE html>
<html lang="ar" dir="rtl">
<head>
    <meta charset="UTF-8">
    <title>السؤال الأول</title>
    <style>
        body {
            background-color: #00a3f2;
            font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
            font-size: 18px;
            text-align: center;
            color: #444;
            margin: 0;
            padding: 0;
        }
        h1 {
            color: #1eff00;
            margin-top: 50px;
            font-size: 36px;
            font-weight: bold;
            text-transform: uppercase;
        }
        p {
            margin-top: 50px;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
        }
        ul {
            list-style-type: none;
            padding: 0;
            margin-top: 50px;
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            align-items: center;
        }
        li a {
            display: block;
            background-color: #100010;
            color: #fff;
            padding: 20px 40px;
            margin: 10px;
            border-radius: 50px;
            text-decoration: none;
            transition: background-color .3s;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
            letter-spacing: 2px;
        }

    </style>
</head>
<body>
```

```
    <h1>السؤال الأول</h1>
    <p>Question 1: Bank ATM Application with TCP Server/Client and Multi-
threading</p>
    <h2>Project Description:</h2>
 <p> 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.
</p>
    <ul>
       <li><a href="/">العودة إلى الرئيسية</a></li>
       <li><a href="page1.html">أكثر معلومات</a></li>
       <li><a href="page3.html">السؤال الثاني</a></li>
    </ul>
</body>
</html>
```

ملف html الخاص بالصفحة page 3:

```
<!DOCTYPE html>
<html lang="ar" dir="rtl">
<head>
    <meta charset="UTF-8">
    <title>السؤال الثاني</title>
    <style>
       body {
          background-color: #00a3f2;
          font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
          font-size: 18px;
          text-align: center;
          color: #444;
          margin: 0;
          padding: 0;
       }
       h1 {
          color: #1eff00;
          margin-top: 50px;
          font-size: 36px;
          font-weight: bold;
          text-transform: uppercase;
       }
       p {
          margin-top: 50px;
          font-size: 24px;
          font-weight: bold;
          text-transform: uppercase;
       }
       ul {
          list-style-type: none;
          padding: 0;
          margin-top: 50px;
```

```
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            align-items: center;
        }
        li a {
            display: block;
            background-color: #100010;
            color: #fff;
            padding: 20px 40px;
            margin: 10px;
            border-radius: 50px;
            text-decoration: none;
            transition: background-color .3s;
            font-size: 24px;
            font-weight: bold;
            text-transform: uppercase;
            letter-spacing: 2px;
        }

    </style>
</head>
<body>
    <h1>السؤال الثاني</h1>
    <p>Question 2: Simple Website Project with Python Flask Framework</p>
    <ul>
        <li><a href="/">الرئيسية إلى العودة</a></li>
        <li><a href="page1.html">أكثر معلومات</a></li>
        <li><a href="page2.html">السؤال الأول</a></li>
    </ul>
</body>
</html>
```

استخدمنا نفس التنسيق في جميع الصفحات.