



ابراهيم وائل يوسف 1874

Question 1:

في هذا السؤال، قمنا بتطوير تطبيق بنك ATM يعتمد على نموذج سيرفر/عميل باستخدام بروتوكول TCP في بايثون. يتيح التطبيق للعملاء الاتصال بالسيرفر، إجراء العمليات البنكية مثل التحقق من الرصيد، الإيداع، والسحب، والحصول على حالة حسابهم المحدثة عند الانتهاء. يستخدم التطبيق الثريدات المتعددة للسماح للسيرفر بمعالجة اتصالات متعددة من العملاء في نفس الوقت.

التصميم والاختيارات

1. السيرفر:

- استخدام مكتبة socket اخترنا استخدام مكتبة socket المدمجة في بايثون لإنشاء اتصال TCP بين السيرفر والعميل.
- الثريدات المتعددة: استخدمنا مكتبة threading لإنشاء ثريدات متعددة بحيث يمكن للسيرفر معالجة اتصالات متعددة في وقت واحد. كل اتصال جديد يتم تعيينه إلى ثريد جديد.
- التحقق من الحساب: يتم التحقق من صحة حساب العميل باستخدام رقم الحساب ورقم التعريف الشخصي (PIN) المقدم من العميل.
- المعاملات البنكية: يوفر السيرفر خيارات للتحقق من الرصيد، الإيداع، والسحب. يتم تحديث رصيد الحسابات على السيرفر مباشرة.

2. العميل:

- الاتصال بالسيرفر: يقوم العميل بالاتصال بالسيرفر باستخدام عنوان IP والمنفذ المحدد.
- التفاعل مع المستخدم: يتفاعل العميل مع المستخدم للحصول على رقم الحساب، رقم التعريف الشخصي، والخيارات البنكية المطلوبة.
- إرسال واستقبال البيانات: يتم إرسال البيانات إلى السيرفر واستلام الردود باستخدام بروتوكول TCP.

كود السيرفر:

```
import socket
import threading

# السيرفر على المحفظة البنكية الحسابات قائمة
accounts = [
    {'name': 'ibraheem', 'pin': '1234', 'balance': 1000.0},
    {'name': 'ali', 'pin': '5678', 'balance': 2000.0}
]

# العملاء اتصالات لمعالجة تابع
def handle_client(csocket):
    try:
        csocket.sendall(b'Welcome to the Bank ATM!\nEnter your name:
```

```

')
    name = csocket.recv(1024).decode().strip()

    # الاسم باستخدام الحساب عن البحث
    account = next((acc for acc in accounts if acc['name'] ==
name), None)
    if account is None:
        csocket.sendall(b'Invalid name.\n')
        csocket.close()
        return

    csocket.sendall(b'Enter PIN: ')
    pin = csocket.recv(1024).decode().strip()

    if account['pin'] != pin:
        csocket.sendall(b'Invalid PIN.\n')
        csocket.close()
        return

    csocket.sendall(b'Authenticated successfully.\n')

    while True:
        csocket.sendall(b'\nChoose an option:\n1. Check
Balance\n2. Deposit\n3. Withdraw\n4. Exit\n')
        option = csocket.recv(1024).decode().strip()

        if option == '1':
            balance = account['balance']
            csocket.sendall(f'Your balance is:
${balance}\n'.encode())

            elif option == '2':
                csocket.sendall(b'Enter amount to deposit: ')
                amount = float(csocket.recv(1024).decode().strip())
                account['balance'] += amount
                csocket.sendall(f'Successfully deposited ${amount}.
Your new balance is ${account["balance"]}\n'.encode())

            elif option == '3':
                csocket.sendall(b'Enter amount to withdraw: ')
                amount = float(csocket.recv(1024).decode().strip())
                if amount > account['balance']:
                    csocket.sendall(b'Insufficient funds.\n')
                else:
                    account['balance'] -= amount
                    csocket.sendall(f'Successfully withdrew
${amount}. Your new balance is ${account["balance"]}\n'.encode())

            elif option == '4':
                csocket.sendall(f'Your final balance is
${account["balance"]}\n'.encode())
                break

            else:
                csocket.sendall(b'Invalid option. Please try
again.\n')
    except Exception as e:
        print(f"Error: {e}")
    finally:
        csocket.close()

```

```

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1234))
server_socket.listen(5)
print("Server listening on port 1234")

while True:
    csocket, addr = server_socket.accept()
    print(f"Accepted connection from {addr}")
    threading.Thread(target=handle_client, args=(csocket,)).start()

```

شرح كود السيرفر:

1. استيراد المكتبات:

- Socket لإنشاء اتصالات TCP.
- Threading لإنشاء ثريدات متعددة لمعالجة الاتصالات المتعددة.

2. قائمة الحسابات:

- قائمة تحتوي على حسابات بنكية مع معلومات الحساب (الاسم، رقم التعريف الشخصي، الرصيد).

3. تابع معالجة العميل:

- يبدأ بإرسال رسالة ترحيبية للعميل وطلب الاسم.
- يتحقق من وجود الحساب باستخدام الاسم المدخل.
- إذا كان الاسم غير صحيح، يتم إغلاق الاتصال.
- يطلب رقم التعريف الشخصي (PIN) ويتحقق من صحته.
- إذا كان PIN صحيح، يتم عرض رسالة تأكيد.
- يقوم بعرض قائمة الخيارات للعميل (التحقق من الرصيد، الإيداع، السحب، أو الخروج).
- يعالج الخيارات المدخلة من العميل ويقوم بتحديث الرصيد بناءً على العملية المختارة.
- إذا اختار العميل الخروج، يتم إرسال الرصيد النهائي وإغلاق الاتصال.

4. إعداد السيرفر:

- إنشاء مقيس (socket) للسيرفر وربطه بالعنوان والمنفذ المحدد.
- يبدأ السيرفر بالاستماع للاتصالات الواردة.
- لكل اتصال جديد، يتم قبول الاتصال وبدء ثريد جديد لمعالجة العميل باستخدام التابع handle_client.

كود العميل:

```

import socket

csocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
csocket.connect(('127.0.0.1', 1234))

```

```

while True:
    response = csocket.recv(4096).decode()
    print(response, end='')

    if "Enter your name" in response or "Enter PIN" in response or
    "Choose an option" in response or "Enter amount" in response:
        message = input()
        csocket.send(message.encode())
    elif "Your final balance" in response:
        break

csocket.close()

```

النتائج والتنفيذ:

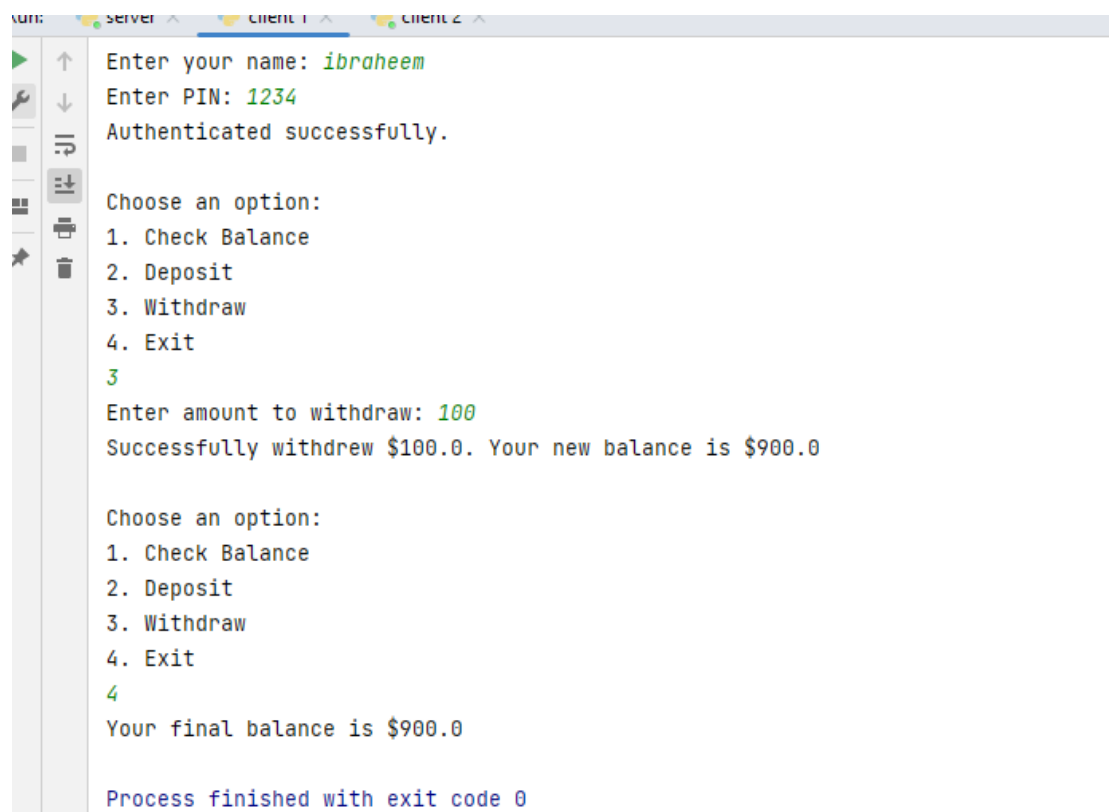
نقوم بتشغيل السيرفر ويتصل به عميلين:

```

Server listening on port 1234
Accepted connection from ('127.0.0.1', 57671)
Accepted connection from ('127.0.0.1', 57678)

```

العميل الأول قام بسحب رصيد وخروج:



```

Server: Enter your name: ibraheem
Client 1: Enter PIN: 1234
Server: Authenticated successfully.
Client 2: Choose an option:
Server: 1. Check Balance
Server: 2. Deposit
Server: 3. Withdraw
Client 2: 3
Server: Enter amount to withdraw: 100
Server: Successfully withdrew $100.0. Your new balance is $900.0
Client 2: Choose an option:
Server: 1. Check Balance
Server: 2. Deposit
Server: 3. Withdraw
Client 2: 4
Server: Your final balance is $900.0
Process finished with exit code 0

```

العميل الثاني قام بالتحقق من الرصيد وسحب صفر دولار وايداع 1500 وخروج

```
server x client 1 x client 2 x
Enter your name: ali
Enter PIN: 5678
Authenticated successfully.

Choose an option:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
1
Your balance is: $2000.0

Choose an option:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
3
Enter amount to withdraw: 0
Successfully withdrew $0.0. Your new balance is $2000.0

Choose an option:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
2
Enter amount to deposit: 1500
Successfully deposited $1500.0. Your new balance is $3500.0

Choose an option:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
4
Your final balance is $3500.0

Process finished with exit code 0
```

Question 2:

يتم انشاء سيرفر باستخدام المكتبة **flask** عن طريق اضافة المكتبة باستخدام التعبير **import** حيث سنضيف الادوات التي سنستخدمها:

1. الكلاس **Flask** لانشاء السيرفر حيث يأخذ بارمتر هو اسم المسار التنفيذي للكود
2. وأيضاً التابع **render_template** الذي يأخذ بارمتر هو صفحة ال**html** المراد إظهارها على المتصفح والتي تكون محفوظة في مجلد اسمه **templates**

التابع **route** المستخدم في **@app.route("/")** يفيد في التقاط الرابط الذي نطلبه من المتصفح وينفذ التابع المرتبط به أي التابع الذي يليه في الكود

الموقع يتكون من صفتين **html** وبالتالي لدينا تابعين يتم تنفيذهما بالاعتماد على رابط الصفحة المطلوب:

1. التابع **index** يتم تنفيذه عن ورود الرابط الأساسي أي **http://127.0.0.1:5000** ولذلك وضعنا / في التابع **route** المرافق. يقوم التابع **index** بإعادة صفحة **index.html** باستخدام التابع **render_template**
 2. التابع **about** يتم تنفيذه عن ورود الرابط **http://127.0.0.1:5000/about** ولذلك وضعنا **/about** في التابع **route** المرافق. يقوم التابع **about** بإعادة صفحة **about.html** باستخدام التابع **render_template**
- يعمل السيرفر باستخدام التابع **run** في وضع التصحيح وعلى المنفذ الافتراضي 5000.

```
from flask import Flask, render_template

app = Flask(__name__)

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

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

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

index.html

```
<!DOCTYPE html>
<html lang="ar">
<head>
  <meta charset="utf-8">
  <title>index</title>
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/5.0.2/css/bootstrap.min.css">
  <link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}">
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-light bg-light">
    <div class="container-fluid">
      <a class="navbar-brand" href="#">الثاني السؤال</a>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNav" aria-
controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link active" aria-
current="page" href="#">الرئيسية</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/about">حول</a>
          </li>
        </ul>
      </div>
    </div>
  </nav>

  <div class="container">
    <h1 class="text-center mt-5">الخاص موقعي في بكم مرحبا</h1>
    <p class="text-center mt-5">من يوسف وائل ابراهيم الطالب أنا
خصيصاً المصمم الموقع هذا في بكم ارحب خامسة سنة الاتصالات هندسة طلاب
</p>
    <p>عيسى مهند للدكتور الشبكات برمجة مقرر في الثانية للوظيفة
</p>
    <a class="btn btn-primary" href="/about">أكثر معلومات</a>
  </div>

</body>
</html>
```

مرحباً بكم في موقعي الخاص

أنا الطالب إبراهيم وأهل يوسف من طلاب هندسة الاتصالات سنة خامسة أرحب بكم في هذا الموقع المصمم خصيصاً للوظيفة الثانية في مقرر برمجة الشبكات للدكتور مهند عيسى

معلومات أكثر

about.html

صفحة حول تحوي معلومات عني كطالب هندسة اتصالات

الاسم: إبراهيم وأهل يوسف الرقم الجامعي : 1874

الرئيسية

```
<!DOCTYPE html>
<html lang="ar">
<head>
  <meta charset="utf-8">
  <title>حول</title>
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/5.0.2/css/bootstrap.min.css">
  <link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}">
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-light bg-light">
    <div class="container-fluid">
      <a class="navbar-brand" href="/">السؤال الثاني</a>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNav" aria-
controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link active" aria-
```



```

</a>الرئيسية" current="page" href="/"
</li>
<li class="nav-item">
  <a class="nav-link" href="/about">حول</a>
</li>
</ul>
</div>
</div>
</nav>

<div class="container">
  <h1 class="text-center mt-5">كطالب عني معلومات تحوي حول صفحة اتصالات هندسة
  </h1>
  <p class="text-center mt-5">الرقم يوسف وائل ابراهيم : الاسم : الجامعي 1874 :
  </p>
  <a class="btn btn-primary" href="/">الرئيسية</a>
</div>

</body>
</html>

```

تحتوي الصفحات على العديد من العناصر الأساسية التي تستخدم في بناء صفحات الويب. وهناك بعض العناصر التي يجب شرحها:

هذا الكود يمثل صفحة HTML تستخدم Bootstrap لتحسين التصميم والتخطيط العام للصفحة. وفيما يلي شرح للأجزاء الرئيسية من هذا الكود:

1. `<!DOCTYPE html>` : يحدد نوع المستند كـ HTML5.
2. `<html lang="ar">` : يحدد لغة الصفحة كعربية.
3. `<head>` : يحدد عناصر رأس الصفحة، مثل العنوان والرابطة إلى ملف CSS.
4. `<meta charset="utf-8">` : يحدد ترميز الحروف الذي يجب استخدامه في الصفحة كـ UTF-8.
5. `<title>index</title>` : يحدد عنوان الصفحة.
6. `<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/twitter-bootstrap/5.0.2/css/bootstrap.min.css">` : يربط الصفحة بملف CSS الخاص بـ Bootstrap من خلال رابط يتم تحميله من موقع cloudflare.
7. `<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">` : يربط الصفحة بملف CSS الخاص بالصفحة والذي يتم تضمينه باستخدام التتابع Flask في `url_for()`.
8. `<body>` : يحدد محتويات الجسم الرئيسية للصفحة.
9. `<nav>` : يحدد شريط التنقل العلوي (navbar) ويستخدم Bootstrap لتحسين التصميم الخاص به.
10. `<div class="container">` : يحدد عنصر يضم جميع محتويات الصفحة ويستخدم Bootstrap لتحسين التصميم الخاص به.

11. `<h1 class="text-center mt-5">` مرحبا بكم في موقعي الخاص :`</h1>` يحدد العنوان الرئيسي للصفحة ويستخدم Bootstrap لتحسين التصميم ووضع النص في وسط الصفحة.

12. `<p class="text-center mt-5">` أنا الطالب ابراهيم وائل يوسف من طلاب هندسة الاتصالات سنة خامسة ارحب بكم في هذا الموقع المصمم خصيصاً للوظيفة الثانية في مقرر برمجة الشبكات للدكتور مهند عيسى `</p>` يحدد الفقرة الرئيسية للصفحة ويستخدم Bootstrap لتحسين التصميم ووضع النص في وسط الصفحة.

13. `` ا获取更多 :`` يحدد زر ينتج الانتقال إلى صفحة أخرى، ويستخدم Bootstrap لتحسين التصميم وتحويل الزر إلى زر أنيميشن بلون أزرق غامق.

14. `</body>` يغلق عنصر الجسم الرئيسي للصفحة.

15. `</html>` يغلق عنصر html الرئيسي للصفحة.

يمكن ملاحظة أن Bootstrap يستخدم عدداً من الفئات المخصصة لتوفير تصميم مسبق للصفحات وتحسين تجربة المستخدم، مثل `navbar` و `container` و `text-center` و `btn` و `btn-primary` وغيرها. يتم تحديد هذه الفئات في عناصر HTML باستخدام الخاصية `class`، ويتم تطبيقها في ملف CSS لتغيير الألوان والأنماط والخطوط والأحجام وغيرها من الخصائص الأخرى في صفحة HTML