

## Proje Raporu: Python ile Socket Programlama

Bu proje, Python programlama dili kullanılarak geliştirilmiş basit bir çoklu kullanıcı sohbet uygulamasını içerir. Sunucu-istemci modeline dayanan bu uygulama, bir sunucu tarafından yönetilen ve birbiriyle iletişim kuran birden fazla istemciden oluşur. İstemciler, sunucuya bağlanarak mesaj gönderebilir ve diğer kullanıcıların mesajlarını alabilirler.

Proje, iki ana dosyadan oluşmaktadır: **server.py** ve **client.py**.

- **server.py**: Bu dosya sunucu tarafını temsil eder. Sunucu, istemcilerin bağlantı kurmasını bekler, bağlantıları kabul eder ve istemciler arasındaki iletişimi yönetir.
- **client.py**: Bu dosya istemci tarafını temsil eder. İstemci, sunucuya bağlanır, sunucudan gelen mesajları alır ve sunucuya mesaj gönderir.

Proje, sunucu ve istemci dosyalarını komut isteminde çalıştırarak kullanılır. Önce sunucu başlatılır. (python server.py), ardından istemciler bağlanır (python client.py). İstemci, sunucuya bağlandığında bir kullanıcı adı girmesi istenir. Daha sonra, kullanıcılar diğer kullanıcılarla sohbet edebilirler. İki dosya arasındaki iletişim, belirli bir protokolü izler: İstemciler sunucuya ilk olarak bir isim gönderirler, ardından mesaj alışverişi başlar. Bu basit protokol, kullanıcıların kimlik doğrulamasını sağlar ve mesajların kimden geldiğini belirler.

```
server.py client.py
C: > Users > aulus > OneDrive > Masaüstü > ChatApp > server.py > ...
1 import socket
2 import threading
3 # Sunucu ayarları
4 HOST = '127.0.0.1'
5 PORT = 55555
6
7 # Bağlı istemcilerin listesi
8 clients = {}
9 lock = threading.Lock()
10
11 # İstemciye mesaj gönderme işlevi
12 def broadcast(message):
13     with lock:
14         print(message) # Mesajı sunucuda görüntüle
15         for client_name, client_socket in clients.items():
16             try:
17                 client_socket.send(bytes(message, "utf-8"))
18             except:
19                 print(f"Hata: {client_name} ile bağlantı kesildi.")
20                 del clients[client_name]
21
22 # İstemci dinleyici işlevi
23 def handle_client(client_socket, client_name):
24     welcome_message = f"{client_name} sohbet katıldı."
25     broadcast(welcome_message)
26
27     while True:
28         try:
29             message = client_socket.recv(1024).decode("utf-8")
30             if message:
31                 broadcast(f"{client_name}: {message}")
32             except:
33                 with lock:
34                     del clients[client_name]
35                 client_socket.close()
36                 broadcast(f"{client_name} sohbetten ayrıldı.")
37                 break
38
39 # Sunucu başlatma işlevi
40 def start_server():
41     server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
42     server.bind((HOST, PORT))
43     server.listen()
44
45     print("Sunucu başlatıldı. Port:", PORT)
46     while True:
47         client_socket, addr = server.accept()
48         client_name = client_socket.recv(1024).decode("utf-8")
49         clients[client_name] = client_socket
50         print(f"{client_name} bağlandı. Adres: {addr}")
51
52         # Yeni istemci iş parçası başlatma
53         client_thread = threading.Thread(target=handle_client, args=(client_socket, client_name))
54         client_thread.start()
55
56 if __name__ == "__main__":
57     start_server()
58
client.py
C: > Users > aulus > OneDrive > Masaüstü > ChatApp > server.py > handle_client
22 # İstemci dinleyici işlevi
23 def handle_client(client_socket, client_name):
24     welcome_message = f"{client_name} sohbet katıldı."
25     broadcast(welcome_message)
26
27     while True:
28         try:
29             message = client_socket.recv(1024).decode("utf-8")
30             if message:
31                 broadcast(f"{client_name}: {message}")
32             except:
33                 with lock:
34                     del clients[client_name]
35                 client_socket.close()
36                 broadcast(f"{client_name} sohbetten ayrıldı.")
37                 break
38
39 # Sunucu başlatma işlevi
40 def start_server():
41     server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
42     server.bind((HOST, PORT))
43     server.listen()
44
45     print("Sunucu başlatıldı. Port:", PORT)
46     while True:
47         client_socket, addr = server.accept()
48         client_name = client_socket.recv(1024).decode("utf-8")
49         clients[client_name] = client_socket
50         print(f"{client_name} bağlandı. Adres: {addr}")
51
52         # Yeni istemci iş parçası başlatma
53         client_thread = threading.Thread(target=handle_client, args=(client_socket, client_name))
54         client_thread.start()
55
56 if __name__ == "__main__":
57     start_server()
58
```

## Sunucu (server.py)

- **Bağlantı Yönetimi:** Sunucu, belirli bir IP adresi ve port numarası üzerinden istemcilerin bağlantı kurmasını bekler. **start\_server()** fonksiyonu, sunucuyu başlatır ve gelen bağlantıları dinler.
- **İstemci İşleme:** Her bağlantı için bir iş parçacığı başlatılır. İstemci bağlandığında, **handle\_client()** fonksiyonu tarafından işlenir. Bu işlev, istemcilerin bağlantılarını kabul eder, mesajları alır ve diğer istemcilere iletir.
- **Mesaj Yayını:** Sunucu, alınan mesajları diğer istemcilere iletmek için **broadcast\_message()** fonksiyonunu kullanır. Bu fonksiyon, mesajı sunucuda görüntüler ve bağlı olan tüm istemcilere gönderir.

### Koda Genel Bakış:

#### 1. Socket ve Thread İçer Aktarma:

```
import socket import threading
```

Soket işlemleri ve iş parçacığı kullanımı için gerekli kütüphaneleri içer aktardım.

#### 2. Sunucu Ayarları:

```
HOST = '127.0.0.1'
```

```
PORT = 55555
```

Sunucunun IP adresini ve bağlanacağı port numarasını belirledim.

#### 3. Bağlı İstemcilerin Listesi:

```
clients = {}
```

```
lock = threading.Lock()
```

Bağlı istemcilerin adlarını ve soketlerini tutacak bir sözlük oluşturur. Ayrıca, birden çok iş parçacığının aynı anda bu sözlüğe erişimini kontrol etmek için bir kilitleme mekanizması sağlar.

#### 4. İstemcilere Mesaj Gönderme Fonksiyonu:

```
def broadcast(message): ...
```

Bu fonksiyon, sunucudan tüm istemcilere mesaj göndermek için kullanılır.

#### 5. İstemci Dinleyici İşlevi:

```
def handle_client(client_socket, client_name): ...
```

Bu işlev, her bir istemci için bir iş parçacığı oluşturur. İstemciden gelen mesajları dinler ve diğer istemcilere iletir.

## 6. Sunucu Başlatma İşlevi:

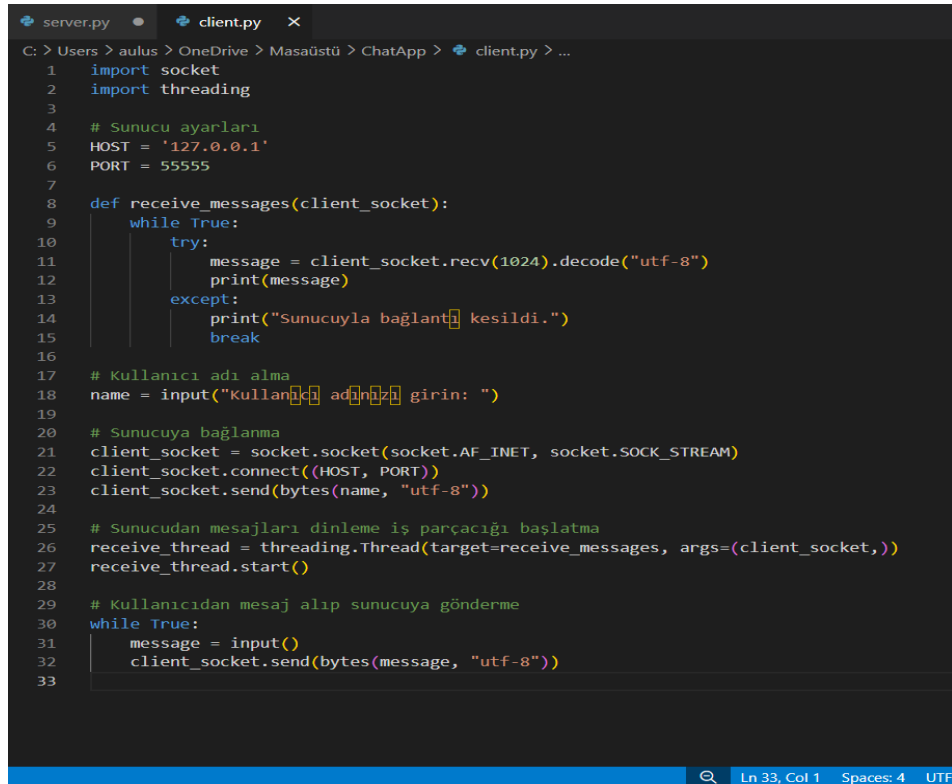
```
def start_server(): ...
```

Bu işlev, sunucuyu başlatır, gelen bağlantıları kabul eder ve her biri için bir iş parçacığı başlatır.

## 7. Ana Program:

```
if __name__ == "__main__": start_server()
```

Kodun ana çalıştırılabilir kısmı, sunucuyu başlatır.



```
server.py client.py X
C: > Users > aulus > OneDrive > Masaüstü > ChatApp > client.py > ...
1 import socket
2 import threading
3
4 # Sunucu ayarları
5 HOST = '127.0.0.1'
6 PORT = 55555
7
8 def receive_messages(client_socket):
9     while True:
10         try:
11             message = client_socket.recv(1024).decode("utf-8")
12             print(message)
13         except:
14             print("Sunucuya bağlantı kesildi.")
15             break
16
17 # Kullanıcı adı alma
18 name = input("Kullanıcı adınızı girin: ")
19
20 # Sunucuya bağlanma
21 client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
22 client_socket.connect((HOST, PORT))
23 client_socket.send(bytes(name, "utf-8"))
24
25 # Sunucudan mesajları dinleme iş parçacığı başlatma
26 receive_thread = threading.Thread(target=receive_messages, args=(client_socket,))
27 receive_thread.start()
28
29 # Kullanıcıdan mesaj alıp sunucuya gönderme
30 while True:
31     message = input()
32     client_socket.send(bytes(message, "utf-8"))
33
```

## İstemci (client.py)

client.py dosyası özellikleri:

- **Sunucuya Bağlanma:** İstemci, belirli bir sunucu IP adresi ve port numarası üzerinden sunucuya bağlanır. Kullanıcı adı alınır ve sunucuya gönderilir. [ **main():** ]
- **Mesaj Alışverişi:** İstemci, sunucudan gelen mesajları dinler ve ekrana yazdırır. [ **receive\_messages(client\_socket):** ] Aynı zamanda kullanıcıdan mesaj alır ve sunucuya gönderir.

## **Koda Genel Bakış:**

### **1. Socket ve Thread İçe Aktarma:**

```
import socket import threading
```

Gerekli kütüphaneleri içe aktardım.

### **2. Sunucu Ayarları:**

```
HOST = '127.0.0.1' PORT = 55555
```

Bağlanılacak sunucunun IP adresini ve port numarasını belirledim.

### **3. Mesajları Almak İçin İş Parçacığı:**

```
def receive_messages(client_socket): ...
```

Bu işlev, sunucudan gelen mesajları dinler ve ekrana yazar.

### **4. Kullanıcı Adı Alma:**

```
name = input("Kullanıcı adınızı girin: ")
```

Kullanıcıdan bir isim alır.

### **5. Sunucuya Bağlanma:**

```
client_socket=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
client_socket.connect((HOST, PORT))
client_socket.send(bytes(name, "utf-8"))
```

Sunucuya bağlanır ve kullanıcı adını sunucuya gönderir.

### **6. Mesajları Almak İçin İş Parçacığı Başlatma:**

```
receive_thread = threading.Thread(target=receive_messages, args=(client_socket,))
receive_thread.start()
```

Sunucudan gelen mesajları dinlemek için bir iş parçacığı başlatır.

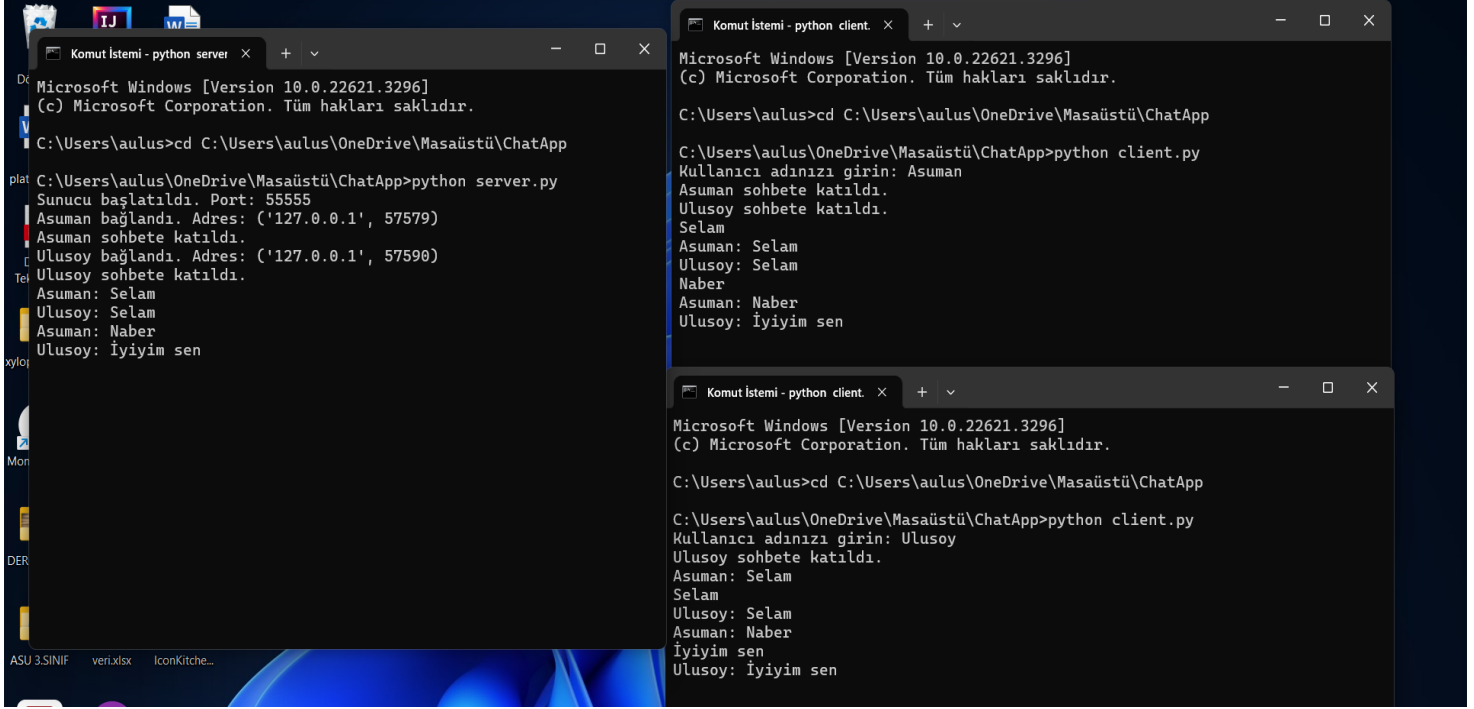
### **7. Mesaj Alıp Sunucuya Gönderme:**

```
while True: message = input() client_socket.send(bytes(message, "utf-8"))
```

Bu döngü, kullanıcıdan sürekli olarak mesaj alıp sunucuya göndermesini sağlar.

## Sonuç

Bu proje, basit bir çoklu kullanıcı sohbet uygulaması sağlar. Sunucu ve istemci arasındaki etkileşim, bir sunucu aracılığıyla gerçekleştirilir ve birden fazla kullanıcının eşzamanlı olarak sohbet etmesine izin verir. Kullanıcılar, sunucuya bağlanarak mesaj gönderebilir ve diğer kullanıcıların mesajlarını alabilirler.



```
Komut İstemi - python server
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\aulus>cd C:\Users\aulus\OneDrive\Masaüstü\ChatApp

C:\Users\aulus\OneDrive\Masaüstü\ChatApp>python server.py
Sunucu başlatıldı. Port: 55555
Asuman bağlandı. Adres: ('127.0.0.1', 57579)
Asuman sohbete katıldı.
Ulusoy bağlandı. Adres: ('127.0.0.1', 57590)
Ulusoy sohbete katıldı.
Asuman: Selam
Ulusoy: Selam
Asuman: Naber
Ulusoy: İyiyim sen

Komut İstemi - python client
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\aulus>cd C:\Users\aulus\OneDrive\Masaüstü\ChatApp

C:\Users\aulus\OneDrive\Masaüstü\ChatApp>python client.py
Kullanıcı adınızı girin: Asuman
Asuman sohbete katıldı.
Ulusoy sohbete katıldı.
Selam
Asuman: Selam
Ulusoy: Selam
Naber
Asuman: Naber
Ulusoy: İyiyim sen

Komut İstemi - python client
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\aulus>cd C:\Users\aulus\OneDrive\Masaüstü\ChatApp

C:\Users\aulus\OneDrive\Masaüstü\ChatApp>python client.py
Kullanıcı adınızı girin: Ulusoy
Ulusoy sohbete katıldı.
Asuman: Selam
Selam
Ulusoy: Selam
Asuman: Naber
İyiyim sen
Ulusoy: İyiyim sen
```