PES UNIVERSITY

COMPUTER NETWORKING PROJECT

SEARCH ENGINE

PROFESSOR: DR. ANNAPURNA

\

SECTION : F

TEAM-MEMBERS:

NISHAN HOLLA	-	PES2UG21CS340
NEERAJ PATIL	-	PES2UG21CS328
NISHANK KOUL	-	PES2UG21CS342

ABSTRACT

The server is built using Python's socket programming library, which enables communication between the client and server over a network. The server stores the documents and uses a pre-built indexing algorithm to efficiently search through the documents. To use the search engine, the user enters the search query into the client interface. The client sends the query to the server, which returns document that match the query. The client then displays the list of matching documents, allowing the user to view the contents of the desired document.
Overall, this search engine provides a simple and efficient way for users to search for information within a collection of documents. The use of socket programming in Python makes it easy to deploy and use the system over a network.

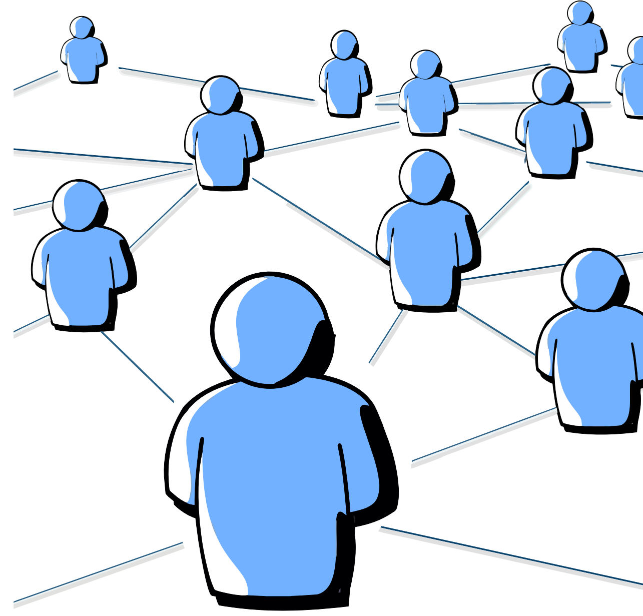
Client's Code

```
import socket
import threading

host = socket.gethostbyname(socket.gethostname())

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect((host, 55555))

def receive():
while True:
try:
message = client.recv(1024).decode('ascii')
print(message)
except:
print("An error occured!")
client.close()
break
```

```
def write():
 while True:
query = '{}'.format(input(''))
client.send(query.encode('ascii'))

# Starting Threads For Listening And Writing
receive_thread = threading.Thread(target=receive)
receive_thread.start()

write_thread = threading.Thread(target=write)
write_thread.start()
```

SERVER'S CODE

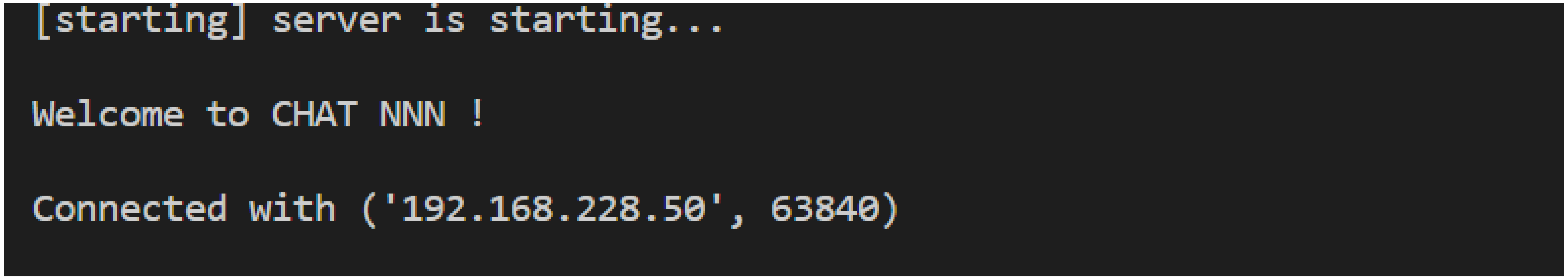
```
import socket
import threading

print('[starting] server is starting... \n \nWelcome to CHAT NNN
!\n ')

host = socket.gethostbyname(socket.gethostname())
port = 55555

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((host, port))
server.listen()

client, address = server.accept()
print("Connected with {}".format(str(address)))

client.send('Connected to server!'.encode('ascii'))
```



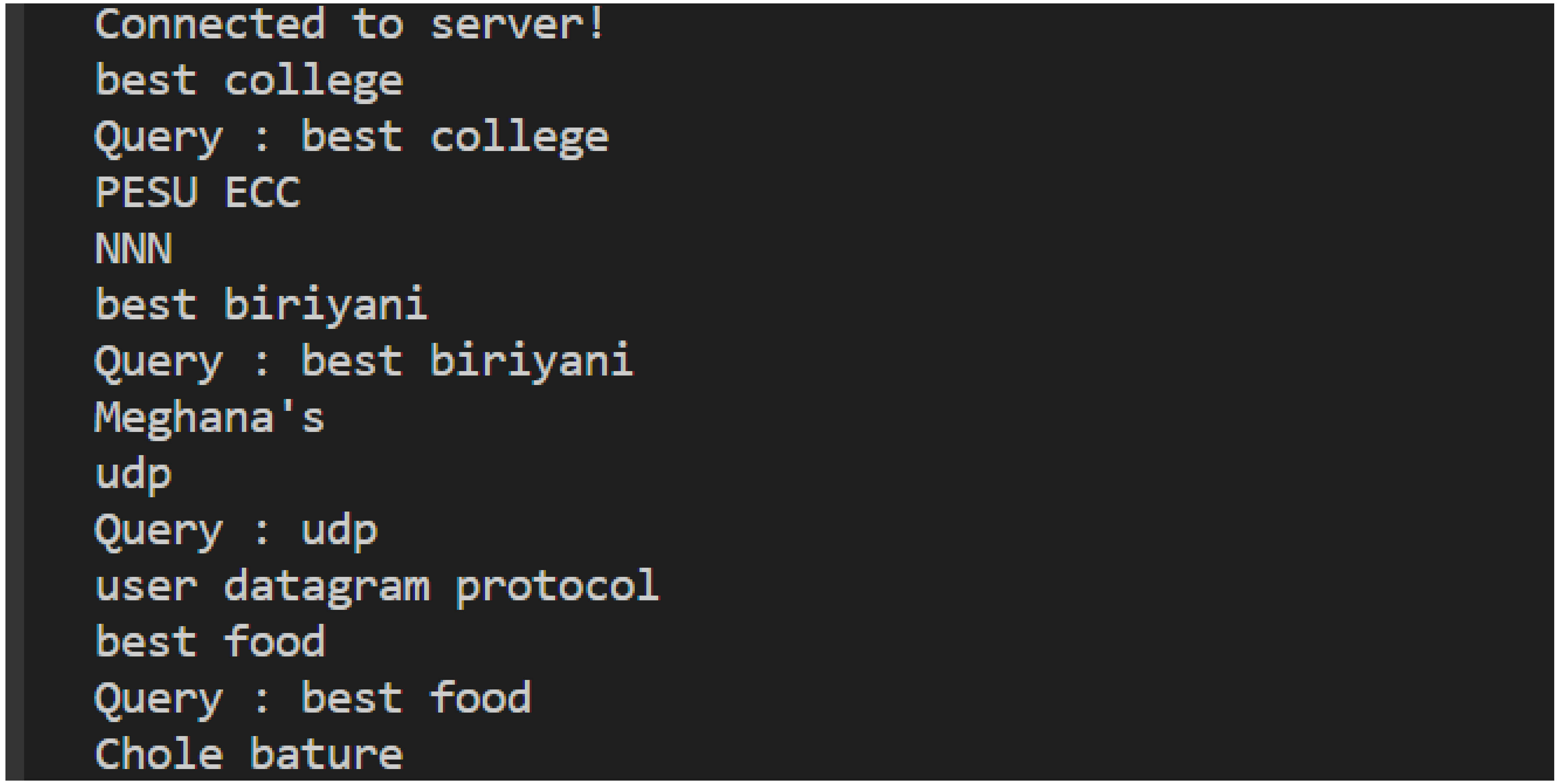
```
quest = ["best college","best
subject","fruit","food","vegetable","pizza","sandwich","hello","creator","transport layer
protocols","application layer protocol","network layer protocol","team name","best
shampoo","best teacher","best milk","best ice cream","best dosa","best
biriyani","tcp","udp","burger","best political party","best leader"]
answ = ["PESU ECC ","Computer Networks","Apple","Chole
bature","carrot","dominos","cool joint","nice to meet you","Nishan Neeraj Nishank","TCP
UDP" ,"HTTP FTP SMTP " , "IP","NNN ","SUNSILK","DR ANNAPURNA","NANDINI","Corner
House","MTR","Meghana's","transport layer protocol","user datagram
protocol","mcdonald's","BJP","Narendra Modi"]

def broadcast(message):
ans = '{}'.format(message)
client.send(ans.encode('ascii'))

def handle(client):
while True:
message = client.recv(1024).decode('ascii')
ref = "Query : "+message
client.send(ref.encode('ascii'))
i=0
```

```
while i<len(quest):
if message == quest[i]:
broadcast(answ[i])
break
i=i+1

if i==len(quest):
broadcast("Sorry I dont have the answer to "+message)

thread = threading.Thread(target=handle, args=(client,))
thread.start()
```

OUTPUT:



```
[starting] server is starting...

Welcome to CHAT NNN !

Connected with ('192.168.228.50', 63840)
```



```
Connected to server!
best college
Query : best college
PESU ECC
NNN
best biriyani
Query : best biriyani
Meghana's
udp
Query : udp
user datagram protocol
best food
Query : best food
Chole bature
```

CONCLUSION

Hence, in this way we can create a search engine application using socket programing in python.

THANK YOU