```python
from socket import *
import sys

if len(sys.argv) <= 1:
    print(
        'Usage : "python ProxyServer.py server_ip"\n[server_ip: IP Address Of Proxy Server')
    sys.exit(2)

# Create a server socket, bind it to a port, and start listening
tcpSerSock = socket(AF_INET, SOCK_STREAM)
# You can choose a different port if needed
tcpSerSock.bind(('10.9.190.34', 8888))
tcpSerSock.listen(5)

# Create a dictionary to keep track of requested files
requested_files = {}

while True:
    # Start receiving data from the client
    print('Ready to serve...')
    tcpCliSock, addr = tcpSerSock.accept()
    print('Received a connection from:', addr)
    message = tcpCliSock.recv(1024).decode()  # Receive data from the client
    print(message)

    # Extract the filename from the given message
    filename = message.split()[1].partition("/")[2]
    print(filename)
    filetouse = "/" + filename

    if filetouse not in requested_files:
        # The file has not been requested before, fetch it from the internet
        try:
            # Create a socket on the proxy server to connect to the web server
            c = socket(AF_INET, SOCK_STREAM)
            hostn = filename.replace("www.", "", 1)
            print(hostn)
```

```
        # Connect to the socket to port 80
        c.connect((hostn, 80))
        # Create a temporary file on this socket and ask port 80
        fileobj = c.makefile('r', 0)
        fileobj.write(f"GET /{filename} HTTP/1.0\n\n")

        # Read the response into a buffer
        buffer = fileobj.readlines()

        # Create a new file in the cache for the requested file
        tmpFile = open(filetouse, "wb")
        for data in buffer:
            tmpFile.write(data.encode())
            tcpCliSock.send(data.encode())

        # Mark the file as requested
        requested_files[filetouse] = True

    except Exception as e:
        print("Error: ", e)

# Proxy server finds a cache hit and generates a response message
try:
    f = open(filetouse[1:], "r")
    outputdata = f.readlines()

    tcpCliSock.send("HTTP/1.0 200 OK\r\n")
    tcpCliSock.send("Content-Type: text/html\r\n")
    for i in range(len(outputdata)):
        tcpCliSock.send(outputdata[i].encode())
    print('Read from cache')

except IOError:
    pass

# Close the client and the server sockets
```

tcpCliSock.close()