Code for the server side:

import socket

from tqdm import tqdm

IP = socket.gethostbyname(socket.gethostname())

PORT = 8080

ADDR = (IP, PORT)

SIZE = 1024

FORMAT = 'utf-8'

def main():

  server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

  server.bind(ADDR)

  server.listen(5)

  print(f'Server is listening at {IP}:{PORT}')

  client, addr = server.accept()

  print(f'Connection from {addr[0]:{addr[1]}}')

  data = client.recv(SIZE)

  print(f'Received data: {data.decode(FORMAT)}')

  item = data.decode(FORMAT).split("\_")

  FILENAME = item[0]

  FILESIZE = int(item[1])

  print(f'File name: {FILENAME}')

  print(f'File size: {FILESIZE}')

  client.send(data)

  bar = tqdm(range(FILESIZE), f'Recieving {FILENAME}', unit='B', unit\_scale=True, unit\_divisor=1024)

*with* open(f"recieved\_{FILENAME}", 'w') *as* f:

*while* True:

      data = client.recv(SIZE)

*if* not data:

*break*

      f.write(data.decode(FORMAT))

      client.send("Data received".encode(FORMAT))

      bar.update(len(data))

  client.close()

  server.close()

*if* \_\_name\_\_ == '\_\_main\_\_':

  main()

code for the client side:  
import socket

from tqdm import tqdm

import os

IP = socket.gethostbyname(socket.gethostname)

PORT = 8080

ADDR = (IP, PORT)

SIZE = 1024

FORMAT = 'utf-8'

FILENAME = input(str('Enter file name: '))

FILESIZE = os.path.getsize(FILENAME)

def main():

  client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

  client.connect(ADDR)

  print(f'Connected to {IP}:{PORT}')

  data = f"{FILENAME}\_{FILESIZE}"

  client.send(data.encode(FORMAT))

  msg = client.recv(SIZE).decode(FORMAT)

  print(f"server: {msg}")

  bar = tqdm(range(FILESIZE), f'Sending {FILENAME}', unit='B', unit\_scale=True, unit\_divisor=1024)

*with* open(FILENAME, 'r') *as* f:

*while* True:

      data = f.read(SIZE)

*if* not data:

*break*

      client.send(data.encode(FORMAT))

      msg = client.recv(SIZE).decode(FORMAT)

      bar.update(len(data))

  client.close()

*if* \_\_name\_\_ == '\_\_main\_\_':

  main()

directory before running the codes:

Text

Description automatically generated

Properties of the text.txt file:

Text

Description automatically generated

After running server.py:

Background pattern

Description automatically generated

After running client.py

A picture containing background pattern

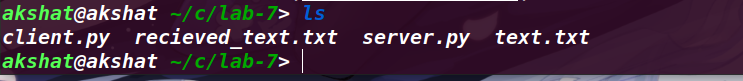
Description automatically generated

After entering the file name:  
Graphical user interface

Description automatically generated with low confidence

On the server side:  
A screenshot of a computer

Description automatically generated with medium confidence

The directories in the folder after transfer:  


The properties of the received file:  
A screenshot of a computer

Description automatically generated with medium confidence