Implementación Aplicación para transferencia de ficheros

Fichero *“cli_fich.py”*:

```
#!/usr/bin/env python3

import socket, sys, os
import szasar

SERVER = 'localhost'
PORT = 6012
ER_MSG = (
      "Correcto.",
      "Comando desconocido o inesperado.",
      "Usuario desconocido.",
      "Clave de paso o password incorrecto.",
      "Error al crear la lista de ficheros.",
      "El fichero no existe.",
      "Error al bajar el fichero.",
      "Un usuario anonimo no tiene permisos para esta operacion.",
      "El fichero es demasiado grande.",
      "Error al preparar el fichero para subirlo.",
      "Error al subir el fichero.",
      "Error al borrar el fichero." )

class Menu:
      List, Download, Upload, Delete, Exit = range( 1, 6 )
      Options = ( "Lista de ficheros", "Bajar fichero", "Subir fichero", "Borrar
fichero", "Salir" )

      def menu():
            print( "+{}+".format( '-' * 30 ) )
            for i,option in enumerate( Menu.Options, 1 ):
                  print( "| {}.- {:<25}|".format( i, option ) )
            print( "+{}+".format( '-' * 30 ) )

            while True:
                  try:
                        selected = int( input( "Selecciona una opción: " ) )
                  except:
                        print( "Opción no válida." )
                        continue
                  if 0 < selected <= len( Menu.Options ):
                        return selected
                  else:
                        print( "Opción no válida." )

def iserror( message ):
      if( message.startswith( "ER" ) ):
            code = int( message[2:] )
            print( ER_MSG[code] )
            return True
      else:
            return False

def int2bytes( n ):
      if n < 1 << 10:
            return str(n) + " B  "
      elif n < 1 << 20:
```

```
            return str(round( n / (1 << 10) ) ) + " KiB"
      elif n < 1 << 30:
            return str(round( n / (1 << 20) ) ) + " MiB"
      else:
            return str(round( n / (1 << 30) ) ) + " GiB"



if __name__ == "__main__":
      if len( sys.argv ) > 3:
            print( "Uso: {} [<servidor> [<puerto>]]".format( sys.argv[0] ) )
            exit( 2 )

      if len( sys.argv ) >= 2:
            SERVER = sys.argv[1]
      if len( sys.argv ) == 3:
            PORT = int( sys.argv[2])

      s = socket.socket( socket.AF_INET, socket.SOCK_STREAM )
      s.connect( (SERVER, PORT) )

      while True:
            user = input( "Introduce el nombre de usuario: " )
            message = "{}{}\r\n".format( szasar.Command.User, user )
            s.sendall( message.encode( "ascii" ) )
            message = szasar.recvline( s ).decode( "ascii" )
            if iserror( message ):
                  continue

            password = input( "Introduce la contraseña: " )
            message = "{}{}\r\n".format( szasar.Command.Password, password )
            s.sendall( message.encode( "ascii" ) )
            message = szasar.recvline( s ).decode( "ascii" )
            if not iserror( message ):
                  break


      while True:
            option = Menu.menu()

            if option == Menu.List:
                  message = "{}\r\n".format( szasar.Command.List )
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode( "ascii" )
                  if iserror( message ):
                        continue
                  filecount = 0
                  print( "Listado de ficheros disponibles" )
                  print( "-------------------------------" )
                  while True:
                        line = szasar.recvline( s ).decode("ascii")
                        if line:
                              filecount += 1
                              fileinfo = line.split( '?' )
                              print( "{:<20} {:>8}".format( fileinfo[0],
int2bytes( int(fileinfo[1]) ) ) )
                        else:
                              break
                  print( "-------------------------------" )
                  if filecount == 0:
                        print( "No hay ficheros disponibles." )
```

```
                  else:
                        plural = "s" if filecount > 1 else ""
                        print( "{0} fichero{1}
disponible{1}.".format( filecount, plural ) )

            elif option == Menu.Download:
                  filename = input( "Indica el fichero que quieres bajar: " )
                  message = "{}{}\r\n".format( szasar.Command.Download, filename
)
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode ("ascii" )
                  if iserror( message ):
                        continue
                  filesize = int( message[2:] )
                  message = "{}\r\n".format( szasar.Command.Download2 )
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode( "ascii" )
                  if iserror( message ):
                        continue
                  filedata = szasar.recvall( s, filesize )
                  try:
                        with open( filename, "wb" ) as f:
                              f.write( filedata )
                  except:
                        print( "No se ha podido guardar el fichero en disco." )
                  else:
                        print( "El fichero {} se ha descargado
correctamente.".format( filename ) )

            elif option == Menu.Upload:
                  filename = input( "Indica el fichero que quieres subir: " )
                  try:
                        filesize = os.path.getsize( filename )
                        with open( filename, "rb" ) as f:
                              filedata = f.read()
                  except:
                        print( "No se ha podido acceder al fichero
{}.".format( filename ) )
                        continue

                  message = "{}{}?{}\r\n".format( szasar.Command.Upload,
filename, filesize )
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode( "ascii" )
                  if iserror( message ):
                        continue

                  message = "{}\r\n".format( szasar.Command.Upload2 )
                  s.sendall( message.encode( "ascii" ) )
                  s.sendall( filedata )
                  message = szasar.recvline( s ).decode( "ascii" )
                  if not iserror( message ):
                        print( "El fichero {} se ha enviado
correctamente.".format( filename ) )

            elif option == Menu.Delete:
                  filename = input( "Indica el fichero que quieres borrar: " )
                  message = "{}{}\r\n".format( szasar.Command.Delete, filename )
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode( "ascii" )
                  if not iserror( message ):
```

```
                        print( "El fichero {} se ha borrado
correctamente.".format( filename ) )

            elif option == Menu.Exit:
                  message = "{}\r\n".format( szasar.Command.Exit )
                  s.sendall( message.encode( "ascii" ) )
                  message = szasar.recvline( s ).decode( "ascii" )
                  break
      s.close()
```

Fichero *“serv_fich.py”*:

```
#!/usr/bin/env python3

import socket, sys, os, signal
import szasar

PORT = 6012
FILES_PATH = "files"
MAX_FILE_SIZE = 10 * 1 << 20 # 10 MiB
SPACE_MARGIN = 50 * 1 << 20  # 50 MiB
USERS = ("anonimous", "sar", "sza")
PASSWORDS = ("", "sar", "sza")

class State:
      Identification, Authentication, Main, Downloading, Uploading = range(5)

def sendOK( s, params="" ):
      s.sendall( ("OK{}\r\n".format( params )).encode( "ascii" ) )

def sendER( s, code=1 ):
      s.sendall( ("ER{}\r\n".format( code )).encode( "ascii" ) )

def session( s ):
      state = State.Identification

      while True:
            message = szasar.recvline( dialog ).decode( "ascii" )
            if not message:
                  return

            if message.startswith( szasar.Command.User ):
                  if( state != State.Identification ):
                        sendER( s )
                        continue
                  try:
                        user = USERS.index( message[4:] )
                  except:
                        sendER( s, 2 )
                  else:
                        sendOK( s )
                        state = State.Authentication

            elif message.startswith( szasar.Command.Password ):
                  if state != State.Authentication:
                        sendER( s )
                        continue
                  if( user == 0 or PASSWORDS[user] == message[4:] ):
                        sendOK( s )
                        state = State.Main
```

```
                  else:
                        sendER( s, 3 )
                        state = State.Identification

            elif message.startswith( szasar.Command.List ):
                  if state != State.Main:
                        sendER( s )
                        continue
                  try:
                        message = "OK\r\n"
                        for filename in os.listdir( FILES_PATH ):
                              filesize =
os.path.getsize( os.path.join( FILES_PATH, filename ) )
                              message += "{}?{}\r\n".format( filename,
filesize )
                        message += "\r\n"
                  except:
                        sendER( s, 4 )
                  else:
                        s.sendall( message.encode( "ascii" ) )

            elif message.startswith( szasar.Command.Download ):
                  if state != State.Main:
                        sendER( s )
                        continue
                  filename = os.path.join( FILES_PATH, message[4:] )
                  try:
                        filesize = os.path.getsize( filename )
                  except:
                        sendER( s, 5 )
                        continue
                  else:
                        sendOK( s, filesize )
                        state = State.Downloading

            elif message.startswith( szasar.Command.Download2 ):
                  if state != State.Downloading:
                        sendER( s )
                        continue
                  state = State.Main
                  try:
                        with open( filename, "rb" ) as f:
                              filedata = f.read()
                  except:
                        sendER( s, 6 )
                  else:
                        sendOK( s )
                        s.sendall( filedata )

            elif message.startswith( szasar.Command.Upload ):
                  if state != State.Main:
                        sendER( s )
                        continue
                  if user == 0:
                        sendER( s, 7 )
                        continue
                  filename, filesize = message[4:].split('?')
                  filesize = int(filesize)
                  if filesize > MAX_FILE_SIZE:
                        sendER( s, 8 )
                        continue
```

```
                  svfs = os.statvfs( FILES_PATH )
                  if filesize + SPACE_MARGIN > svfs.f_bsize * svfs.f_bavail:
                        sendER( s, 9 )
                        continue
                  sendOK( s )
                  state = State.Uploading

            elif message.startswith( szasar.Command.Upload2 ):
                  if state != State.Uploading:
                        sendER( s )
                        continue
                  state = State.Main
                  try:
                        with open( os.path.join( FILES_PATH, filename), "wb" )
as f:
                              filedata = szasar.recvall( s, filesize )
                              f.write( filedata )
                  except:
                        sendER( s, 10 )
                  else:
                        sendOK( s )

            elif message.startswith( szasar.Command.Delete ):
                  if state != State.Main:
                        sendER( s )
                        continue
                  if user == 0:
                        sendER( s, 7 )
                        continue
                  try:
                        os.remove( os.path.join( FILES_PATH, message[4:] ) )
                  except:
                        sendER( s, 11 )
                  else:
                        sendOK( s )

            elif message.startswith( szasar.Command.Exit ):
                  sendOK( s )
                  return

            else:
                  sendER( s )



if __name__ == "__main__":
      s = socket.socket( socket.AF_INET, socket.SOCK_STREAM )

      s.bind( ('', PORT) )
      s.listen( 5 )

      signal.signal(signal.SIGCHLD, signal.SIG_IGN)

      while True:
            dialog, address = s.accept()
            print( "Conexión aceptada del socket {0[0]}:{0[1]}.".format( address
) )
            if( os.fork() ):
                  dialog.close()
            else:
                  s.close()
```

```
                session( dialog )
                dialog.close()
                exit( 0 )
```

Fichero *“szasar.py”*:

```
class Command:
      User, Password, List, Download, Download2, Upload, Upload2, Delete, Exit =
("USER", "PASS", "LIST", "DOWN", "DOW2", "UPLO", "UPL2", "DELE", "EXIT")

def recvline( s, removeEOL = True ):
      line = b''
      CRreceived = False
      while True:
            c = s.recv( 1 )
            if c == b'':
                  raise EOFError( "Connection closed by the peer before
receiving an EOL." )
            line += c
            if c == b'\r':
                  CRreceived = True
            elif c == b'\n' and CRreceived:
                  if removeEOL:
                        return line[:-2]
                  else:
                        return line
            else:
                  CRreceived = False

def recvall( s, size ):
      message = b''
      while( len( message ) < size ):
            chunk = s.recv( size - len( message ) )
            if chunk == b'':
                  raise EOFError( "Connection closed by the peer before
receiving the requested {} bytes.".format( size ) )
            message += chunk
      return message
```