```python
"""
*FTPChat - Encrypted FTP-based Messaging Protocol
*Version 1.3 — September 2025
*Author: Ahmed Omar Saad
*Contact: ahmedomardev@outlook.com
*License: Custom MIT — Commercial use requires written permission
*All rights to the name “FTPChat” and its protocol specification are retained by the author.
*For more information, Check the `LICENSE` file."""

from datetime import datetime
from ftplib import FTP
from os import name, system


# !The start of the characters list and key for mac encryption layers


CHARACTERS = (
    "a",
    "b",
    "c",
    "d",
    "e",
    "f",
    "g",
    "h",
    "i",
    "j",
    "k",
    "l",
    "m",
    "n",
    "o",
    "p",
    "q",
    "r",
    "s",
    "t",
    "u",
    "v",
    "w",
    "x",
    "y",
    "z",
    "A",
    "B",
    "C",
    "D",
    "E",
    "F",
    "G",
    "H",
    "I",
    "J",
    "K",
    "L",
    "M",
    "N",
    "O",
    "P",
    "Q",
    "R",
    "S",
    "T",
    "U",
    "V",
    "W",
    "X",
    "Y",
    "Z",
    "!",
    '"',
    "#",
    "$",
    "%",
    "&",
    "'",
    "(",
    ")",
    "*",
    "+",
    ",",
    "-",
    ".",
    "/",
    ":",
    ";",
    "<",
    "=",
    ">",
    "?",
    "@",
    "[",
    "\\",
    "]",
    "^",
    "_",
    "`",
    "{",
    "|",
    "}",
    "~",
    "0",
    "1",
    "2",
    "3",
    "4",
    "5",
    "6",
    "7",
    "8",
    "9",
)
KEY_FOR_MAC_1 = (
    "!",
    ")",
    ".",
    "3",
    "(",
    "J",
    "b",
    "C",
    "A",
    ";",
    "n",
    "_",
    "$",
    "0",
    ",",
    "Z",
    "B",
    "c",
    "|",
        "g",
        "7",
        "^",
        '"',
        "I",
        "v",
        "P",
        "Y",
        "M",
        "i",
        "O",
        "k",
        "L",
        "%",
        "y",
        "D",
        "#",
        "@",
        "<",
        "9",
        "r",
        "K",
        "2",
        "-",
        "t",
        "8",
        "H",
        "=",
        "+",
        "`",
        "o",
        "T",
        "u",
        "s",
        "p",
        "U",
        "S",
        "Q",
        "~",
        "'",
        "G",
        "*",
        "&",
        ">",
        "X",
        "a",
        "N",
        "4",
        "d",
        "m",
        "l",
        "V",
        "w",
        "z",
        ":",
        "q",
        "?",
        "5",
        "x",
        "{",
        "/",
        "F",
        "j",
        "]",
        "6",
        "W",
        "f",
    "R",
    "1",
    "}",
    "e",
    "[",
    "\\",
    "h",
    "E",
)
KEY_FOR_MAC_2 = (
    "$",
    "i",
    "o",
    "c",
    "&",
    "L",
    "/",
    "~",
    "O",
    "@",
    "(",
    "[",
    "Q",
    "v",
    "a",
    "Y",
    "7",
    "3",
    "*",
    "X",
    "x",
    "8",
    "{",
    "K",
    "E",
    "4",
    "M",
    "q",
    "|",
    "J",
    "0",
    "<",
    "Z",
    "e",
    "_",
    "V",
    "H",
    "m",
    "G",
    "A",
    "d",
    "y",
    "T",
    ".",
    "S",
    "j",
    "I",
    "#",
    "]",
    "p",
    "\\",
    "F",
    "b",
    '"',
    "W",
    "k",
    ";",
    "5",
    "N",
    "s",
    "2",
    "l",
    "1",
    ":",
    "r",
    "-",
    "`",
    "h",
    ">",
    "t",
    "}",
    "^",
    "+",
    "U",
    "B",
    "!",
    "6",
    ")",
    ",",
    "P",
    "g",
    "=",
    "C",
    "D",
    "?",
    "R",
    "9",
    "z",
    "'",
    "n",
    "w",
    "f",
    "%",
    "u",
)
KEY_FOR_MAC_3 = (
    "h",
    "R",
    "-",
    "I",
    "4",
    "$",
    "Z",
    "c",
    "/",
    "Q",
    "O",
    "a",
    "E",
    "V",
    "_",
    "q",
    "A",
    "!",
    ",",
    "X",
    "N",
    "H",
    "`",
    ".",
    "K",
    "e",
    "*",
    "v",
        "\\",
        "T",
        "k",
        "M",
        "Y",
        "^",
        "6",
        "l",
        "j",
        "G",
        "'",
        "m",
        "x",
        "7",
        "3",
        "L",
        ";",
        "C",
        "1",
        "p",
        "2",
        "S",
        "]",
        "r",
        "%",
        "g",
        "@",
        "0",
        ")",
        "}",
        "B",
        "W",
        "|",
        "t",
        "=",
        "?",
        ">",
        "{",
        "b",
        "d",
        '"',
        "u",
        "f",
        "+",
        "P",
        ":",
        "z",
        "i",
        "F",
        "~",
        "[",
        "(",
        "&",
        "o",
        "n",
        "D",
        "#",
        "5",
        "8",
        "J",
        "w",
        "s",
        "U",
        "9",
        "<",
        "y",
    )
```

```
KEY_FOR_MAC_4 = (
    "_",
    "(",
    "H",
    "7",
    "~",
    "t",
    "F",
    "D",
    "j",
    "5",
    "C",
    "#",
    "o",
    "^",
    "X",
    "z",
    "9",
    "-",
    "k",
    '"',
    "N",
    "E",
    "+",
    "q",
    "P",
    "/",
    ";",
    "S",
    "W",
    "h",
    "s",
    "f",
    ".",
    "A",
    "4",
    "'",
    "\\",
    "Q",
    ",",
    "m",
    "e",
    "$",
    "6",
    "p",
    "i",
    "y",
    "|",
    "U",
    "a",
    "3",
    "L",
    "g",
    "c",
    "]",
    "0",
    ")",
    "v",
    "x",
    "8",
    "T",
    "l",
    "u",
    "b",
    "[",
    "V",
    ":",
    "M",
    "r",
    "B",
    "G",
    "1",
    "K",
    "d",
    ">",
    "O",
    "`",
    "*",
    "I",
    "?",
    "2",
    "}",
    "n",
    "@",
    "<",
    "&",
    "Z",
    "{",
    "R",
    "%",
    "!",
    "Y",
    "w",
    "=",
    "J",
)
KEY_FOR_MAC_5 = (
    "W",
    "f",
    "v",
    "l",
    '"',
    "h",
    "@",
    "C",
    ";",
    "!",
    "a",
    "8",
    "5",
    "S",
    "G",
    "X",
    "-",
    "{",
    "^",
    "~",
    "|",
    "m",
    "=",
    "&",
    "H",
    "O",
    "M",
    "Q",
    "0",
    "d",
    "I",
    "g",
    "6",
    "J",
    "9",
    "E",
    "z",
    "K",
    "L",
    "?",
    "/",
    ">",
    "o",
    "]",
    "n",
    "Y",
    "F",
    "t",
    "i",
    "D",
    "w",
    "#",
    "(",
    "`",
    "y",
    "%",
    "r",
    "[",
    "p",
    "*",
    "b",
    "j",
    "<",
    "}",
    "'",
    "\\",
    ",",
    "B",
    "R",
    "q",
    "U",
    "x",
    ")",
    "$",
    "_",
    "1",
    "+",
    "T",
    "N",
    "s",
    "7",
    "c",
    "k",
    ":",
    "4",
    "3",
    "e",
    "Z",
    "V",
    "P",
    "2",
    ".",
    "u",
    "A",
)
KEY_FOR_MAC_6 = (
    "}",
    "X",
    "j",
    "%",
    "I",
    "T",
    "L",
    "s",
      "^",
      "m",
      "d",
      "K",
      ")",
      "-",
      "B",
      "D",
      "o",
      "E",
      "@",
      "Z",
      "`",
      "2",
      "f",
      "a",
      "6",
      "q",
      "e",
      "n",
      "*",
      "R",
      "[",
      ";",
      "V",
      '"',
      "z",
      "y",
      "+",
      "S",
      "W",
      "7",
      "x",
      "&",
      "<",
      "9",
      "U",
      "N",
      "]",
      "#",
      "=",
      "b",
      "Q",
      "Y",
      "(",
      "r",
      "\\",
      "M",
      "A",
      "v",
      "H",
      "{",
      "w",
      "p",
      "3",
      ",",
      "k",
      "'",
      "t",
      ":",
      "u",
      "J",
      "8",
      "F",
      ">",
      "i",
      "G",
    "P",
    "0",
    "/",
    "~",
    "|",
    "h",
    "O",
    "g",
    "_",
    "c",
    "l",
    "C",
    "4",
    "?",
    "1",
    ".",
    "!",
    "$",
    "5",
)
KEY_FOR_MAC_7 = (
    "U",
    "F",
    ".",
    "T",
    "n",
    "}",
    ")",
    "E",
    "I",
    "S",
    "{",
    "\\",
    ">",
    "B",
    "a",
    "j",
    "Z",
    "/",
    "<",
    "P",
    "e",
    "O",
    "|",
    ",",
    "i",
    "0",
    "l",
    "6",
    "@",
    ";",
    "X",
    "M",
    "J",
    "#",
    "2",
    "7",
    "+",
    "s",
    "]",
    "h",
    "x",
    "&",
    "y",
    "-",
    "`",
    "G",
    "3",
    ":",
    "(",
    "L",
    "=",
    "f",
    "*",
    "H",
    "Q",
    "R",
    "8",
    "4",
    "z",
    "[",
    "^",
    "5",
    "g",
    "D",
    "Y",
    "!",
    "W",
    "$",
    "q",
    "9",
    "K",
    "k",
    '"',
    "d",
    "p",
    "A",
    "u",
    "w",
    "C",
    "_",
    "o",
    "b",
    "'",
    "r",
    "t",
    "N",
    "v",
    "?",
    "m",
    "1",
    "c",
    "%",
    "V",
    "~",
)
KEY_FOR_MAC_8 = (
    "u",
    "5",
    "_",
    "Y",
    "l",
    "p",
    "D",
    "L",
    "b",
    "k",
    "@",
    "9",
    "O",
    "0",
    "*",
    "}",
    "f",
        "w",
        "C",
        "d",
        '"',
        "g",
        "t",
        "K",
        "&",
        ";",
        "s",
        "v",
        "o",
        "=",
        "7",
        "I",
        "]",
        ".",
        "R",
        "c",
        "?",
        "a",
        "^",
        "V",
        "x",
        "%",
        "n",
        "\\",
        "z",
        "B",
        "'",
        "Q",
        "U",
        "+",
        "m",
        "1",
        ",",
        "F",
        "M",
        "$",
        "T",
        "3",
        "N",
        ")",
        "r",
        "y",
        ">",
        "8",
        "P",
        "q",
        "2",
        "<",
        "|",
        "`",
        "4",
        "~",
        "/",
        "A",
        ":",
        "J",
        "{",
        "(",
        "H",
        "-",
        "h",
        "i",
        "j",
        "G",
    "S",
    "X",
    "[",
    "6",
    "Z",
    "E",
    "W",
    "#",
    "e",
    "!",
)
KEY_FOR_MAC_9 = (
    ",",
    "T",
    "g",
    "B",
    "5",
    "n",
    "f",
    "s",
    "G",
    "W",
    "V",
    "`",
    "M",
    "[",
    "!",
    "e",
    "L",
    "-",
    "_",
    "Q",
    "9",
    "^",
    ")",
    "P",
    "0",
    "j",
    "2",
    "v",
    "?",
    "E",
    "Y",
    "Z",
    "1",
    "|",
    ".",
    "#",
    "w",
    "D",
    "J",
    "l",
    "O",
    "@",
    "t",
    "{",
    "x",
    "+",
    "4",
    "S",
    ";",
    "u",
    "F",
    "~",
    "q",
    "}",
    "<",
```

```python
    "N",
    "o",
    "(",
    "K",
    "/",
    "z",
    "a",
    "\\",
    "p",
    "3",
    '"',
    "i",
    "8",
    "7",
    "%",
    "6",
    "U",
    ">",
    "C",
    "X",
    "$",
    "R",
    "&",
    "d",
    "c",
    "I",
    "y",
    "H",
    "=",
    "'",
    "h",
    "r",
    "m",
    "*",
    "b",
    ":",
    "A",
    "]",
    "k",
)
KEY_FOR_MAC_10 = (
    "0",
    "R",
    "h",
    "C",
    "~",
    "o",
    "i",
    "g",
    ",",
    "k",
    "8",
    "H",
    "Q",
    "p",
    "9",
    "v",
    "$",
    "u",
    "5",
    "-",
    "a",
    "{",
    '"',
    "s",
    "A",
    ";",
        "[",
        "Y",
        "^",
        "r",
        "M",
        "b",
        "G",
        "_",
        "m",
        "e",
        "x",
        "I",
        "T",
        "n",
        ">",
        "<",
        "P",
        "J",
        "U",
        "2",
        "3",
        "4",
        "#",
        "W",
        ":",
        "w",
        "D",
        "|",
        "(",
        "z",
        "q",
        "K",
        "V",
        "B",
        "L",
        "j",
        "d",
        "&",
        "!",
        "1",
        "+",
        "\\",
        "X",
        "/",
        "?",
        "@",
        "*",
        "t",
        "c",
        "}",
        "Z",
        "%",
        ".",
        ")",
        "E",
        "`",
        "f",
        "N",
        "y",
        "6",
        "7",
        "O",
        "=",
        "]",
        "F",
        "l",
        "S",
    "'",
)
KEY_FOR_MAC_11 = (
    "S",
    "}",
    "!",
    "N",
    "\\",
    "C",
    "M",
    "[",
    "z",
    "?",
    "%",
    "7",
    "q",
    "Y",
    ".",
    "$",
    "l",
    "D",
    "G",
    "f",
    "+",
    "/",
    "6",
    "#",
    "5",
    "w",
    "P",
    "O",
    "U",
    "*",
    "X",
    "d",
    "=",
    "3",
    ":",
    "i",
    "y",
    "h",
    "v",
    "(",
    "R",
    "W",
    "x",
    '"',
    "1",
    "c",
    "A",
    "<",
    "J",
    "_",
    "L",
    "|",
    "T",
    "^",
    ">",
    "H",
    "2",
    "`",
    "I",
    ";",
    "0",
    "k",
    "-",
    "u",
```

```python
    "o",
    "&",
    "Z",
    "j",
    "p",
    ")",
    ",",
    "r",
    "9",
    "'",
    "Q",
    "4",
    "]",
    "m",
    "t",
    "B",
    "a",
    "F",
    "V",
    "b",
    "8",
    "E",
    "n",
    "g",
    "@",
    "e",
    "s",
    "~",
    "{",
    "K",
)


# !The end of the characters list and key for mac encryption layers


# !---The start for the all encryption and decryption functions---


def reverser(text):
    """This function reverses the text"""
    return text[::-1]


# ---The end of reverser function.---


def mac1_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (1)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_1[index])
        else:
            result.append(char)
    return "".join(result)


def mac1_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (1)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_1:
            index = KEY_FOR_MAC_1.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (1)---


def mac2_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (2)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_2[index])
        else:
            result.append(char)
    return "".join(result)


def mac2_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (2)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_2:
            index = KEY_FOR_MAC_2.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (2)---


def mac3_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (3)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_3[index])
        else:
            result.append(char)
    return "".join(result)


def mac3_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (3)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_3:
            index = KEY_FOR_MAC_3.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (3)---


def mac4_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (4)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_4[index])
        else:
            result.append(char)
    return "".join(result)


def mac4_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (4)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_4:
            index = KEY_FOR_MAC_4.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (4)---


def mac5_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (5)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_5[index])
        else:
            result.append(char)
    return "".join(result)


def mac5_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (5)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_5:
            index = KEY_FOR_MAC_5.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (5)---


def mac6_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (6)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_6[index])
        else:
            result.append(char)
    return "".join(result)


def mac6_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (6)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_6:
            index = KEY_FOR_MAC_6.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (6)---


def mac7_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (7)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_7[index])
        else:
            result.append(char)
    return "".join(result)


def mac7_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (7)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_7:
            index = KEY_FOR_MAC_7.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (7)---


def mac8_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (8)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_8[index])
        else:
            result.append(char)
    return "".join(result)


def mac8_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (8)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_8:
            index = KEY_FOR_MAC_8.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (8)---


def mac9_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (9)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_9[index])
        else:
            result.append(char)
    return "".join(result)


def mac9_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (9)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_9:
            index = KEY_FOR_MAC_9.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (9)---


def mac10_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (10)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_10[index])
        else:
            result.append(char)
    return "".join(result)


def mac10_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (10)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_10:
            index = KEY_FOR_MAC_10.index(char)
            result.append(CHARACTERS[index])
        else:
            result.append(char)
    return "".join(result)


# ---The end for the encryption and decryption functions - Number (10)---


def mac11_encode(text):
    """Encryption layer for Mono-Alphabetic cipher - Number (11)."""
    result = []
    for char in text:
        if char in CHARACTERS:
            index = CHARACTERS.index(char)
            result.append(KEY_FOR_MAC_11[index])
        else:
            result.append(char)
    return "".join(result)


def mac11_decode(text):
    """Decryption layer for Mono-Alphabetic cipher - Number (11)."""
    result = []
    for char in text:
        if char in KEY_FOR_MAC_11:
            index = KEY_FOR_MAC_11.index(char)
```

```python
                result.append(CHARACTERS[index])
            else:
                result.append(char)
        return "".join(result)


    # ---The end for the encryption and decryption functions - Number (11)---


    # !---The end for the all encryption and decryption functions---


    def all_mac_encryption(text):
        """12 Layers combined MAC encryption"""
        layer1 = mac5_encode(text)
        layer2 = reverser(layer1)
        layer3 = mac4_encode(layer2)
        layer4 = mac2_encode(layer3)
        layer5 = mac1_encode(layer4)
        layer6 = mac3_encode(layer5)
        layer7 = mac8_encode(layer6)
        layer8 = mac7_encode(layer7)
        layer9 = mac9_encode(layer8)
        layer10 = mac6_encode(layer9)
        layer11 = mac11_encode(layer10)
        return str(mac10_encode(layer11))


    def all_mac_decryption(text):
        """12 Layers combined MAC decryption"""
        layer1 = mac10_decode(text)
        layer2 = mac11_decode(layer1)
        layer3 = mac6_decode(layer2)
        layer4 = mac9_decode(layer3)
        layer5 = mac7_decode(layer4)
        layer6 = mac8_decode(layer5)
        layer7 = mac3_decode(layer6)
        layer8 = mac1_decode(layer7)
        layer9 = mac2_decode(layer8)
        layer10 = mac4_decode(layer9)
        layer11 = reverser(layer10)
        return str(mac5_decode(layer11))


    def encrypt(text):
        """Doubled 12 Layer MAC encryption"""
        return str(all_mac_encryption(all_mac_encryption(text)))


    def decrypt(text):
        """Doubled 12 Layer MAC decryption"""
        return str(all_mac_decryption(all_mac_decryption(text)))


    # !---The sending and reading messages functions---


    def send_message(
        username, message, ftp_host, ftp_user, ftp_pass, chat_input, chat_name
    ):
        """Send a message to the chat file on the FTP server."""
        try:
            if message == "REFRESH":
                read_messages(ftp_host, ftp_user, ftp_pass, chat_input, chat_name)
            else:
                msg = encrypt(f"{datetime.now()}:{username}: {message}\n")
                ftp = FTP(ftp_host)
                ftp.login(ftp_user, ftp_pass)
                try:
                    with open(chat_input, "wb") as file:
                        ftp.retrbinary(f"RETR {chat_name}", file.write)
                except:
                    pass

                with open(chat_input, "a", encoding="utf-8") as file:
                    file.write(f"{msg}\n")

                with open(chat_input, "rb") as file:
                    ftp.storbinary(f"STOR {chat_name}", file)

                ftp.quit()
        except Exception as error:
            print(f"Error sending message: {error}")
            input()


    def read_messages(ftp_host, ftp_user, ftp_pass, chat_input, chat_name):
        """Read messages from the chat file on the FTP server."""
        try:
            with FTP(ftp_host) as ftp:
                ftp.login(ftp_user, ftp_pass)
                with open(chat_input, "wb") as file:
                    ftp.retrbinary(f"RETR {chat_name}", file.write)
            with open(chat_input, "r", encoding="utf-8") as file:
                lines = file.readlines()
                if not lines:
                    print("No messages yet.")
                    return
                print("------Messages------")
                for line in lines:
                    decrypted_line = decrypt(line.strip())
                    print(decrypted_line)
        except Exception as error:
            print(f"Error reading messages. ({error})")
            input()


    def multiline_input(prompt):
        """Get multiline input from the user until 'END' is entered."""
        print(prompt)
        lines = []
        while True:
            line = input()
            if line.upper() == "END":
                break
            if line.upper() == "REF":
                line = "REFRESH"
                break
            if line.upper() == "CLS":
                system("cls" if name == "nt" else "clear")
                break
            lines.append(line)
        return "".join(lines)


    print(
        r"""
        ________________  ________          ____ \-\
       / ____/_  __/ __ \/ ____/ /_  ____ _/ /    \-\
      / /_    / / / /_/ / /   / __ \/ __ `/ __/    \-\
     / __/   / / / ____/ /___/ / / / /_/ / /_      /-/
    /_/     /_/ /_/    \____/_/ /_/\__,_/\__/     /-/
                                                 /-/
    """
    )

    all_filled = False

    while True:
        ftp_host = input("Please, type the FTP host:\n")
        if not ftp_host:
            print("No Host entered, Exiting the app")
            input()
            break

        ftp_user = input("Please, type the FTP username:\n")
        if not ftp_user:
            print("No FTP User entered, Exiting the app")
            input()
            break

        ftp_pass = input("Please, type the FTP password:\n")
        if not ftp_pass:
            print("No Password entered, Exiting the app")
            input()
            break

        username = input("Please, type your username:\n")
        if not username:
            username = "Anonymous"
            print("Username is empty, using 'Anonymous' as username.")
            continue

        chat_input = input("Please, type the chat file name:\n") + ".txt"
        if not chat_input:
            print("No chat file name input, exiting the program")
            input()
            break

        chat_name = f"/usb1_1/{chat_input}"
        all_filled = True
        break


    if all_filled:
        print(
            """
    Notes before start messaging:
    1. Type 'END' to finish your message.
    2. Type 'REF' to refresh messages.
    3. Type 'CLS' to clear the terminal.
    4. Don't turn off your FTP Server while using this chat.
    5. Messages are encrypted for security.
    6. To secure your chat, use a unique chat file name, and share the file name with the participants.
    """
        )

        while True:

            read_messages(ftp_host, ftp_user, ftp_pass, chat_input, chat_name)
            message = multiline_input("Type your message:\n")
            send_message(
                username, message, ftp_host, ftp_user, ftp_pass, chat_input, chat_name
            )

```