**Techniques used:**

* Python language
* Multithreading
* Socket library
* Time library
* File handling
* Tkinter GUI

**Implementation:**

* First of all we created a server.py and a client.py file (same client.py file for all clients).
* Multithreading is used to handle multiple clients at a time.
* Multithreads providing live chat.
* Dictionaries are used to save every client’s info.
* Lists (inside dictionaries) are used to blacklist clients.
* Block/unblock/sleep/name are implemented through comparison of incoming messages with predefined variables.
* Digital material is sent through file handling techniques.
* A list of supported file formats is initialized.
* Maximum file size is set to 4MB (megabytes)
* Tkinter is used to provide user interface.

**Analysis:**

* Live chat without multithreading lead us to wrong functionality.
* Small buffer size took too long to read and write data of digital material.
* Keys mapping to Lists inside dictionaries lead us to logical errors.
* Distinguishing between text message and digital material was a bit tricky and made our code a bit inefficient.
* Multithreading without main thread was logically a wrong approach.
* Reading and writing files using loops caused reading and writing of data abnormally
* Identifying file format without list of supported formats was difficult.