# Final Project

## Final Project Guideline:

[Google Doc Link](https://docs.google.com/document/d/1SU8x_TNHdjzocSAzvC8QoXhl9OhHDEsWQrIHwncAQSI/edit?usp=sharing)

## GUI Elements To-Do

* [o] Upgrade the Design the UI/UX
  + ☐ Finish login features
  + ☐ Create a user dashboard
  + ☐ Add buttons to replace typed in user functions
  + ☒ Make it so that the chat box is deactivated unless there is being a message displayed
  + ☒ Make sure that while chatting, when the user types exit in the chat room, they leave immediately
  + ☒ Add the user’s name at the top of the window
* ☐ Add login features
  + ☐ Add a password login box
  + ☐ Password protected message history
  + ☐ Index system that stores messages with prior friends
* ☐ Disable the chat box when it is not in use

## Ngrok Elements To-Do

* ☐ Issue:
  + ☐ Fix the creation of the .idx documents. That’s causing errors when connecting. I should make them locally
  + ☐ There is an issue with repeat names I’m not so sure of the root of. It’s very confusing
  + ☐ I will have to look at how idx files are created. All data should be saved locally
  + ☐ There is an issue with repeating names in the chat system
    - I’m trying to delete the .idx files but I’m not sure if that’s the issue. This is very strange ## Completed Tasks
* ☒ Make it so that people who connect to my website are able to chat with me
  + ☒ Go onto another git branch for this
  + ☒ This will require socket programming skills
* ☒ Fixed the state issue
* ☒ Make sure that your messages show up when I text
* ☒ Fix the problem of the client crashing when disconnecting and then reconnecting
* ☒ Add the basic GUI interface

## Schedule:

* Team Formation (please find your team by April 30)
  + each team can have two or three members.
  + But for teams of three, you need to complete the compulsory topic and two selective topics.)
  + Submitting your code and video record (by May 12)

## Notes:

* You are welcome to meet with instructors and LAs for the project. Feel free to book a meeting.
* Q&A: (A few questions and solutions found in the previous semesters) [Docs Link](https://docs.google.com/document/d/1VGs13szC3GiDqtlSiaJRoMwcjUz3fRV18LoXR2kd5sE/edit?usp=sharing) ## Gallery: (works of students in previous semesters)
* Teris and secure message: [Drive Link](https://drive.google.com/file/d/16kr_c9RbebGZiefRRDeyR3boGF0b6DNk/view?usp=sharing)
* Online game - Pong: [Drive Link](https://drive.google.com/file/d/14nb__fM5pX4BG5gZwVJYo0amWJdxDBIM/view?usp=sharing)
* Secure message and GUI ( a project of a team of three): [Video 1](https://drive.google.com/file/d/1GXiGjgzgAslYVLCWDHSJI7crEgkO5-uV/view?usp=sharing) [Video 2](https://drive.google.com/file/d/19VRoLFmyIIF-P8wmsN5_Z2thAE6VLIpO/view?usp=sharing)

## Graphic User Interface (GUI)

### Current State

Currently, the chat system is run from the terminal command line. All input and output are only found from the plain terminal window. You might want to add a user interface to enhance the chatting experience.

### The Goal

To construct an interface using Python packages like Tkinter, so that there is no need to type or read from the terminal window when a client connects to the chat server. All information being exchanged will be displayed on the GUI.

### The Bottom Line

is that you need to make a GUI that can take and display messages for the user. Other interesting features to try out are logging in with passwords, file transferring, voice/video chatting, etc.

### Extras

Besides, you should try to integrate your work on the selective topic with the GUI. For example, if your selective topic is Secure Messaging, you can use the encryption algorithms to transmit the messages behind the GUI. Or, if your selective topic is Online Gaming, you may add a button in the GUI that can start the game you designed.

## Secure Messaging

### Introduction

Turing and his friends built one of the first computers to crack the Enigma code. Every technology, by itself, stands as a double-edged sword. As a user, you have the right to protect your transactions against eavesdropping. But the question is: how?

### Assignment

The intuitive idea is, of course, to scramble your message, a process that we call encryption. You want your friend to read the clear message, so you will need decryption as well.` This takes the form of an agreement between you and your friend before you send your scrambled message.

### Information

On the Internet, everything said is in the open. So there is no hideaway in which you and your friend can meet. So how is it possible to establish such an agreement, and then communicate?

The field of computer security (and a related field, data privacy) is a vast and interesting research area. In this project, however, we only want to get a taste of it.

### The goal:

* Get two chat clients to communicate with each other securely, meaning that the server who is passing the messages in between has no idea what they are talking about.
* You will need to implement a shared key protocol as well as a simple encryption/decryption algorithm for the client-side, and make this work!
* Reference: MacCormick, 9 algorithms, Chapter 4.

## Grading Policy

The work will be evaluated on the following criteria: - Compulsory topic: 50% - [ ] Baseline (40%): the GUI can take and display messages for users - [X] Innovative features (10%): the GUI contains one (or more) ingenious functions, e.g., login password, emoji, etc. - [X] Add user authentication - Selective topic: 20% - [ ] Completeness (10%): the code works properly, and the goals of the topic are achieved. - [ ] Integrated into the GUI (10%): the functions work with the GUI - Video Presentation: 30% - [ ] Powerpoint - [ ] Introduction(6%): Tell the audience what your project is; what is your motivation? - [ ] Discussion (6%): Tell the audience what you did in the development, e.g., what packages you use, and/or, the significant modification in the original chat system - [ ] Analysis(6%): Your reflections on the development, e.g., is the organization of your code reasonable? What classes do you define? Or, the possible improvements - [ ] Impression(6%): This is about the impression of your presentation and your workload, being ranked with regard to others - [ ] Demo of the app(6%): A demo of your work, showing what your app looks like and how it works