Shu You id: shuy

Project Proposal

Project Name: MessageLite

Modules/Technology: Socket (Server and Client), PIL, Speech Recognition API

Identify the Problem

There are many message/chat apps today and each has its own features which attract different groups of people to use. In order to make the app unique, different companies are throwing features on to the app. However, many of those features might be useless to some groups of people, such as old people. It usually takes longer time for old people to learn new things, such as to learn to send an "animoji" on iPhone X (and it's quite useless to some people). For most of the time, old people use message app only to send and receive message, and they do not really need features like stickers, "animoji", timeline, and so on. For old people, different features of various chat apps only complicate the UI and make it harder for them to learn to handle the app properly. Therefore, I want to design a message app that is easy to use and only includes necessary features for those people who like to keep things simple.

How I intend to solve

There are several tasks I need to work on in building this app. The first task is to identify what features I will include in my app. I will do competitive analysis on the most popular message apps on the market to see the must have feature of a message app. At this moment, I plan to include features (other than UI design) like send and receive text between two users, send and receive pictures between two users, recognize users' speech and convert it to text, save chat history, and group chat. Then I will design the UI such as button and entry box layout based on the features I have identified before. I plan to break the tasks by the features. One feature represents one development stage. I will start from the feature that is most frequently used, which is send and receive text. I will first work on a temporary local file with event based animation framework to test the functions, after that I will modify and integrate that with socket. By doing this I can make sure I always have a working copy of the main file at a particular stage of the development.

Algorithmic Plan

One tricky part of this app is to properly place text within a conversation bubble and properly place them on the screen when the conversation length is longer than the height of the screen so that user can scroll up and down to view the entire conversation.

Handle Text:

- Set the maximum width of the text widget which is 2/3 of the width of the screen.

- Take user input as a string and pass the string to a tkinter text widget.
- Choose the width of the text widget to be the length of the text. If the length of the text exceeds the maximum width allowed for the text box, change to a new line.

Handle conversation bubble position:

- Start with a blank conversation screen. Start to place message bubble at the top of the screen.
- Once the bubble reaches the bottom of the screen, increase the offset by the height of the bubble that is to be added. Redraw all bubbles starting at the top of the screen deduct by the offset, so that the new message will always be displayed at the bottom of the screen. Previous messages will be placed above the top the screen.
- Use scroll wheel event to change offset value to enable scrolling function.
- Also, detect top and bottom boundary to make sure scroll stops at certain point.