**AC21011.2 – Snapchat**

**Group:** James Keefe, Jamison Carlen, Alexander Palatnik, Jared Basile

Interactions:

1. Drawing tool - Press ‘1’, double click color, up/down arrow key for change in thickness, press mouse to draw
2. Stickers/Shapes – Press ‘2’, click and drag to location, choose different shapes with ‘q’, ‘w’, ‘e’, ‘r’, ‘t’.
3. Date – Press ‘3’
4. Timestamp – Press ‘4’
5. Image filters – Press 5 to enter filter mode, press ‘w’ for black and white, press ‘q’ for grayscale, press ‘e’ to clear filters
6. Option to add text to the image – Press 6 and type to add text to the image
7. RGB image tints – Press 7 to enter tint mode, press ‘q’ for red filter, press ‘w’ for green filter, press ‘e’ for blue filter, press ‘r’ to clear tint
8. Add sounds to the image – Press ‘q’ to start sound, press ‘w’ to stop sound and rewind
9. Save current picture – press ‘9’
10. Reset to original image – Press ‘0’

Almost all of the interactions we chose to include in this assignment were inspired directly from the Snapchat app. When we were brainstorming what to include in our sketch, we decided to take a deeper look at the functions included in the actual Snapchat app, particularly the ones we use on a daily basis. Then we began mirroring them in our own code. A user is able to put different filters on their image and add the current time/date stamps very much like those available to us in Snapchat. There is a side bar with color selections to draw on the image, also very similar to the one in the app. We have shape stickers that can be placed on the image and a specific key to increase or decrease the size. These are comparable to the emoji’s and shapes that are offered in Snapchat. We also added an option to add text to the image. This is one of the most used functions in Snapchat so we thought it was important to include it in our version as well. In further discussion within the group, we decided to add the ability to add sound bites to an image. Lastly, we included a reset button to return the image to its original form as well as a save option because they are basic necessities for any photo-editing program.

The most challenging part of the coding process was combining all the interactions together. Each group member took a specific interaction to work on. However, upon combining all of it together, we needed to make sure each interaction had its own unique key or mouse activity while not being too complex. The drawing tool was the easiest to come to a conclusion on. It was logical for the mouse to be used to draw on the image by clicking on a color and then pressing down on the mouse to use it. For all the other interactions, we came to a consensus to set specific keys on the keyboard for them.

In addition to combining all the interactions into one program, we also were challenged with keeping the code organized and accurate. We used comments to break up each section of code that pertained to a specific interaction. For example, all the code related to filters is defined by commented line breaks. We also went back through all the combined code to change variables to consistent names throughout.

A more specific coding challenge came with the implementation of the Android sticker. Like all the other shapes, we wanted the Android to be able to be resized. However, the size could not simply be adjusted using the size/resize functions because all parts of the Android would resize separately and overlap each other. After some searching on Google, the solution was found to be the scale function. Once this was discovered, we had to also figure out how to use the same key value for both resizing the shapes/stickers and also scaling the unique Android sticker. The solution was setting one key value to call the resizing function if a shape was also called, but call the scaling function if an Android was called.

The last major coding challenge we came across during this assignment was with the image filters and rgb tints. The first problem was that all the filters and tints were being called at once and we could not switch from one to another. To solve this, we created a special key value to enter filter mode and then special key values to call the grayscale filter or the black and white filter. The same was done for the rgb tints. There is a special key to enter the tint mode and another key for each of the rgb values. Once we solved this problem, we needed to figure out how to reset the image without the selected tints. After a little research, we found the noTint function. Therefore, we added another special key to call that function and set the image back to its original state.