LED Project

# GoMoji

### Elizabeth Minty

Problem

|  |
| --- |
|  |

Entertainment is a key aspect in the modern human life. Everyday, we are seeing people of all ages seek new and exciting, yet affordable and accessible forms of fun. We are seeing the likes of social media and handheld, portable devices still being used as main forms of entertainment, as well as a consistent demand for aestheticism in tech and trends from the 90’s/early 2000’s. How can we create something simple, yet just as entertaining with endless possibilities for the future?

Purpose

|  |
| --- |
|  |

The ‘GoMoji’ is a handheld product that displays a cute virtual pet (called GoMoji) for the user to take care of. The GoMoji can come in many forms and will need to be regularly fed, pet, and given sleep to keep it satisfied. The GoMoji’s different states of being is displayed to the user using LEDs and an LCD screen with buttons to give the GoMoji what it needs.

The GoMoji serves to be the user’s unique pet as well as a keychain accessory that can be taken and worn anywhere.

Use Case

|  |
| --- |
|  |

Looking for an aesthetic piece of tech to add to your collection or for wear? The GoMoji acts as a keychain as well as a badge to add to any outfit. The device itself offers a unique appearance for users to decorate and change to their preference, fitting with any aesthetic or outfit any day.

The GoMoji is a simple, yet rewarding device that gives users small doses of satisfaction throughout the day as they complete tasks to ensure the comfort of their virtual pet.

As it is small and portable, users can use it anywhere even in class or meetings! This offers the chance to share with friends and converse about their unique pets.

Implementation

|  |
| --- |
|  |

The solution that the GoMoji will offer after further iterations is as follows:

* + Evolutions and ability to dress the pet
  + A cute pet that rewards users when they take care of it
  + More states/needs like sickness, excitement, contentment, focus, etc
  + More options to interact with the GoMoji
  + Sound and music

Currently, the product offers the following solution:

* + A cute pet that changes design depending on state
  + Basic states/needs such as contentment, hunger, affection, and fatigue
  + 3 colours to represent these states (red = tired, yellow = hungry, green = affection needed)

Scope & Insights

|  |
| --- |
|  |

The scope for the project was simple: display 3 states of being at random intervals using 3 LEDs, an LCD screen for the character, and 3 buttons to revert those states to normal. This remained relatively the same for the duration of the project’s implementation. Ideally, a box to contain the Arduino and breadboards was to be made if there had been time.

There were some limitations as follows:

* + Tight schedule due to academic and personal commitments
  + This being a solo project
  + Materials

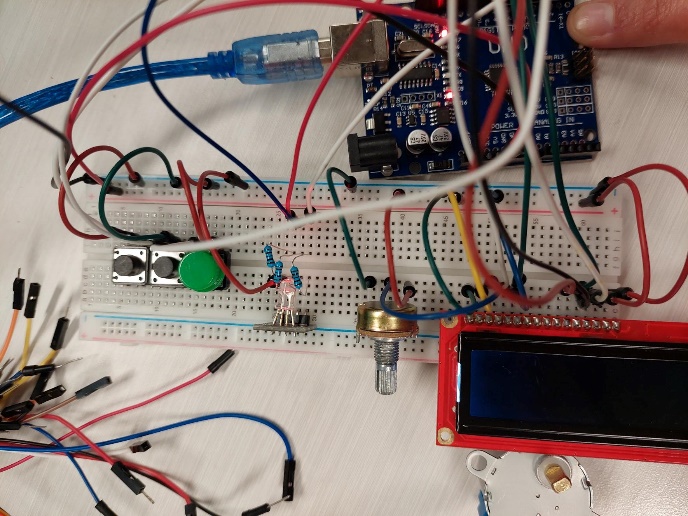
A lot of time went into research on hooking up each part, trying out different parts, methods for the LCD library, code examples, and how to display unique characters. I learned a lot about the pin mapping of different parts and further developed my interest in hardware.

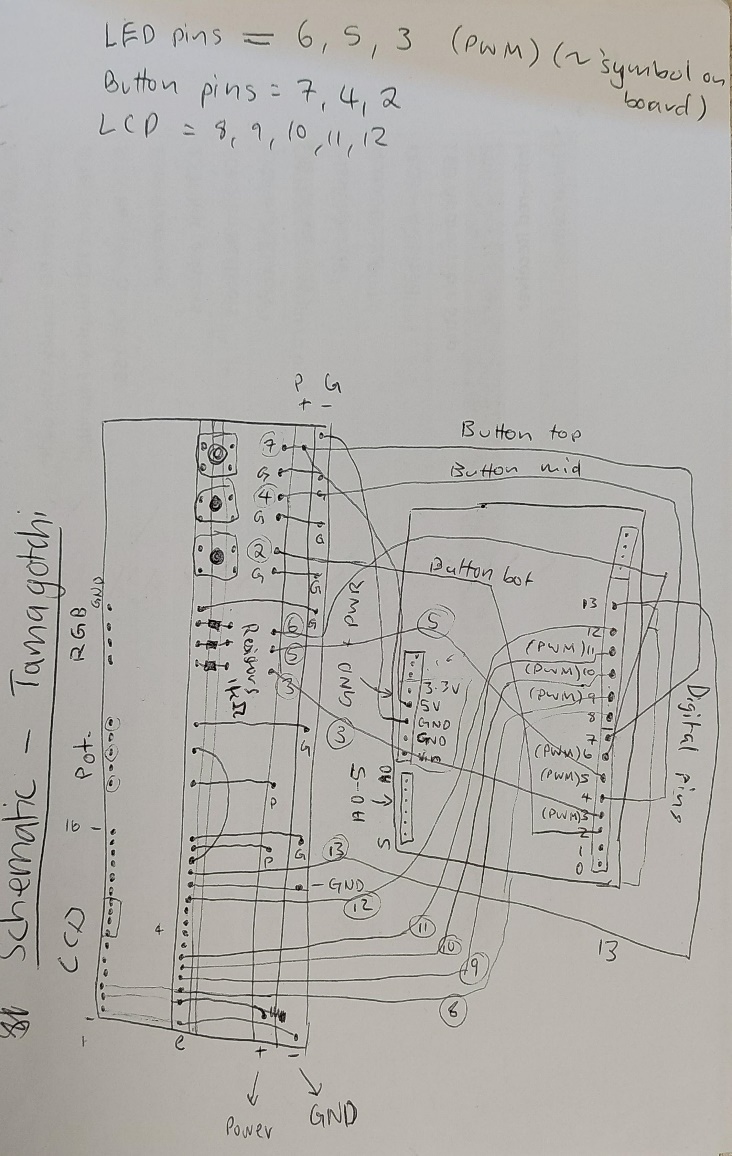
A major assumption that disrupted the flow of production was that the SparkFun guide and LCD screen would work. A big issue at the beginning of the project was that I used a SparkFun LCD screen along with the associated hookup guide to use for my GoMoji display. I tried a second SparkFun LCD and that also did not work. I then had to move on to an LCD screen with a different manufacturer which then worked with my hookup.

Design

|  |
| --- |
|  |

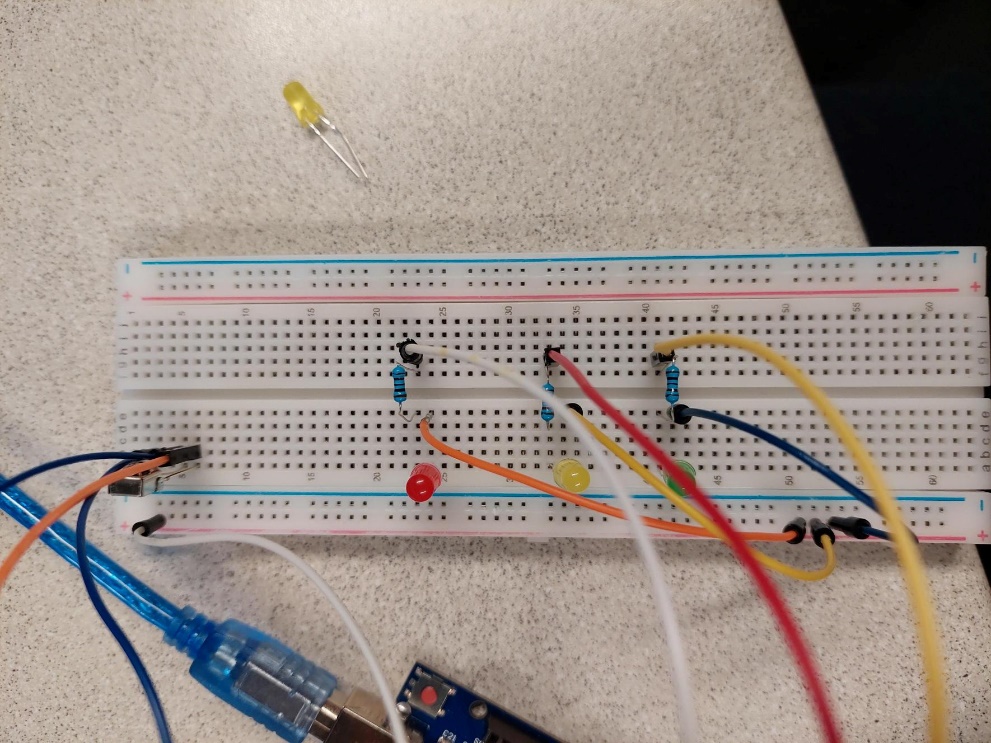
Due to the issues with the SparkFun LCD (SFLCD), the schematic changed pretty frequently. I started off with the SFLCD, a 10k potentiometer (pot), 1 breadboard, an RGB led, 3 buttons, and 3 resistors.

 *Fig 1.a. Initial Schematic with Pot, SFLCD, and RGB LED*

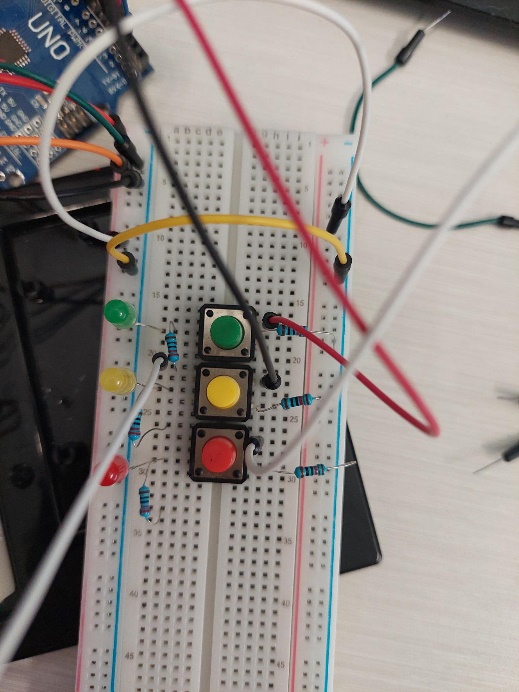
*Fig 1.b. Sketched schematic for initial design with SFLCD, pot, and RGB LED*

This was the first design I had in mind which, according to my sources, should have worked. I had to replace the SFLCD with a different manufacturer and decided to use individual LEDs rather than an RGB LED. The reason for this was that I only needed 3 colours for 3 states at the time and didn’t have a second potentiometer.

I asked Vaughn for help over the weekend. So, as I waited, I started planning for the LEDs and buttons.

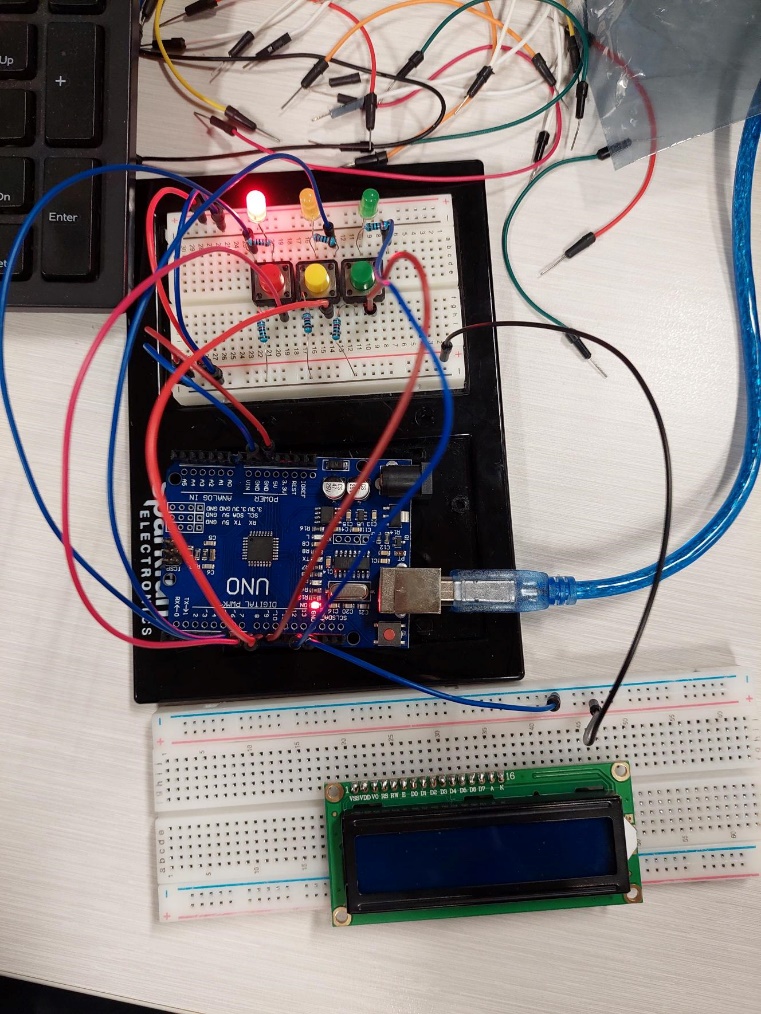
*Fig 2.a. Initial setup of 3 LEDs* 

I started with the LEDs. The scope required 3 LEDs (minimum) that would randomly light up to display a need/state of the pet. This was to happen at random and would only have 1 LED on at any time. The LED would stay on until turned off via the corresponding button press. This is the set up for the LEDs on their own in the final design as well. I initially coded the LEDs to turn on at random individually for this draft design. This helped with the final code by providing some setup as well as solidifying my knowledge on coding with LEDs.

*Fig 2.b. Initial setup of 3 buttons* 

Following the draft design of the LEDs, I implemented the same practice for the button design. I set each button up to ensure I had a good idea of how they need to be wired.

*Fig 3. Final design for buttons that turn off LEDs*



The last step I took before coding the LCD screen, I put the LED and button designs together to ensure they worked seamlessly. Then, I created code that turned on a randomly selected LED at a random interval that would not turn off until the corresponding-coloured button was pressed and released.

This step took a significant amount of time due to my assumption that I’d wired the resistors correctly. The LEDs would keep flickering, would change colour without a button press, and the read value of each button would change without being pressed. Turns out, I had connected the resistor for the LEDs to the buttons as well, so the buttons had 2 resistors. Fortunately, while I thought it was a programming

// components used and hookup

// why didn’t use other compoennts

// design flaws or why did/didn’t use some concepts (eg. Interrupts)

Future Improvements

|  |
| --- |
|  |

Finally, the solution isn’t perfect. There needs to be a few more iterations before it can complete. There is flickering when the states change on the LCD screen, there are limited states, and the code could use more modularity.