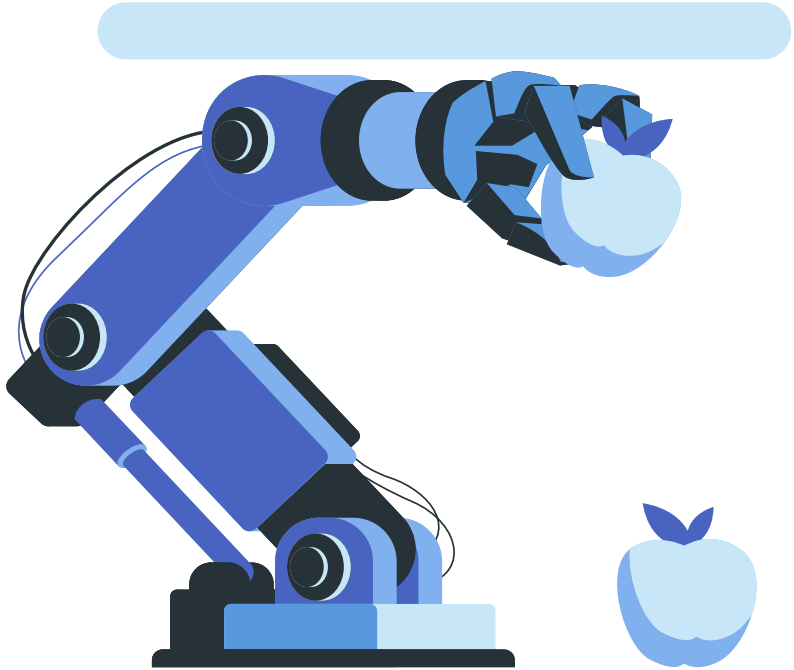
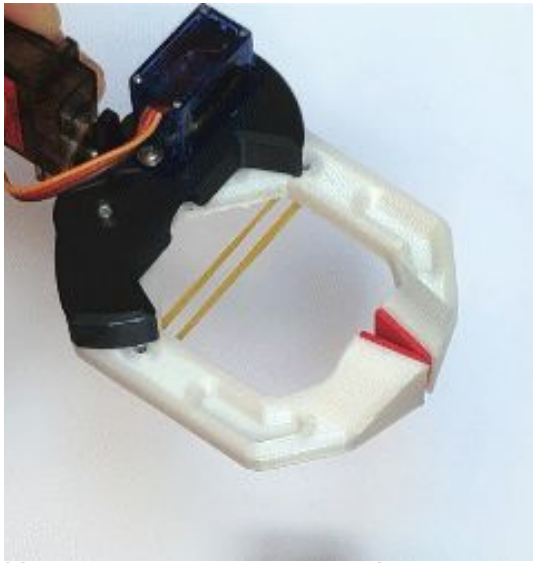
Project 1: Gripper Presentation

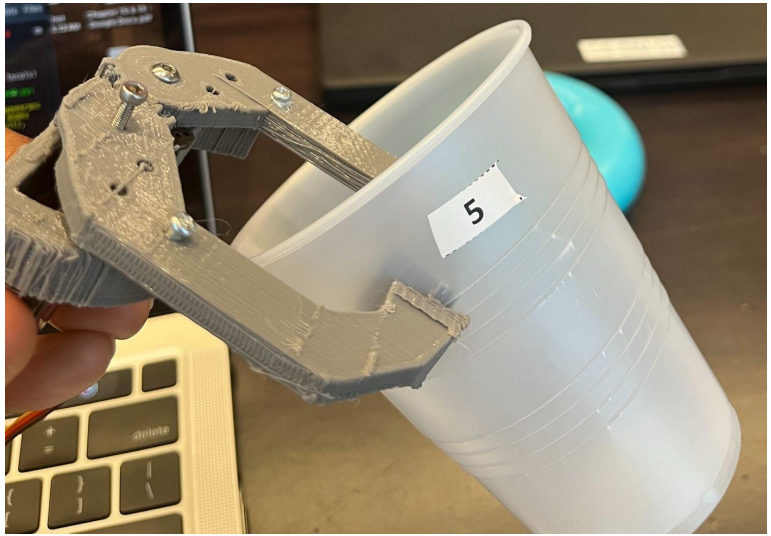
By: Kelly Pleitez, Winnie Delinois, and Edwin Ramirez



Why We Chose It

- We chose the gripper since we believed it would be a simple design that met our specific goals.
- These goals included:
  - Grasping various objects of different lengths and sizes
  - A gripper that was not time consuming to understand and quick to assemble
  - A design that can be easily troubleshooted for errors and can be printed in a short amount of time





https://www.thingiverse.com/thing:3648782

Strengthens / Weakness

Strengths:

- ➔ Robot gripper design only required two parts for printing, making it easy to access and assemble
- ➔ Gripper has been designed to catch and grasp objects that are of small to medium size
- ➔ On Thingiverse website, the gripper comes with written assembly guide along with assembly videos, and example objects that the gripper can grasp
- ➔ Designed with opening for servo

Weaknesses:

- ➔ Gripper parts did not print as nicely as preferred, which made assembly difficult
- ➔ Assembly required modification and access to outside tools for better functionality such as different screw sizes and purchase of **Eva Foam** for grip
- ➔ Tightening the screws can affect how freely the clamp can move which required adjustments
- ➔ The servo becomes hot in a short amount of time, which can result in finger burns when handling the gripper for long periods

Developing the GUI Design

Python: Gripper Control System Code

```python
#defines the slider that controls the gripper when it opens
def open_slider_event(value):
    global servo_value

    servo_value = int(value)
    txt = "Open Position: {}".format(servo_value)#displays the value of the angle the slider has been moved to
    label_width.configure(text=txt)
    write_data() #this functions calls the gripper to write data

#controls the slider that controls the gripper when it closes
def close_slider_event(value):
    global servo_value

    servo_value = int(value)
    txt = "Close Position: {}".format(servo_value)
    label_width2.configure(text=txt)
    write_data() #function that calls the gripper to write data

#the button that defines the gripper when it is opening
def open_gripper():
    open_slider.set(75)
    open_slider_event(75)#setting it to 130 right now to test something out. Originally it will be 180

#defines the button that the user clicks to close the gripper
def close_gripper():
    close_slider.set(0)
```

```python
#creates the label for Open Position and displays what the angle of the servo has been moved to
label_width= customtkinter.CTkLabel(master = app, text = "Open Position: 0", font = custom_font)
label_width.place(relx = 0.1, rely = 0.2, anchor = tkinter.W)

#the Open slider-> the range that the slider moves from  0 - 180 and the deign of the slider
open_slider = customtkinter.CTkSlider(master = app, from_=75, to =100, command = open_slider_event, height = 35, width = 250, fg_color = "#928
open_slider.set(0)
open_slider.place(relx = 0.5, rely = 0.3, anchor = tkinter.CENTER) #where the slider is being positioned


#creates the label for Close Slider and sets the position of where it is being placed which is below the Open Position label
label_width2 = customtkinter.CTkLabel(master = app, text = "Close Position: 0", font = custom_font)
label_width2.place(relx = 0.1, rely = 0.4, anchor = tkinter.W)

#Close Slider-> the user is able to
close_slider = customtkinter.CTkSlider(master = app, from_=75, to = 180, command = close_slider_event, height = 35, width = 250, fg_color="#93
close_slider.set(0)
close_slider.place(relx = 0.5, rely = 0.5, anchor = tkinter.CENTER)

#Open Button Widget
open_button = customtkinter.CTkButton(master = app, text = "Open Gripper", command = open_gripper, font = custom_font)
open_button.place(relx = 0.3, rely = 0.7, anchor = tkinter.CENTER)

#Close Button
close_button = customtkinter.CTkButton(master = app, text = "Close Gripper", command = close_gripper, font = custom_font)
close_button.place(relx = 0.7, rely = 0.7, anchor = tkinter.CENTER)

app.mainloop()
```

Arduino Code:

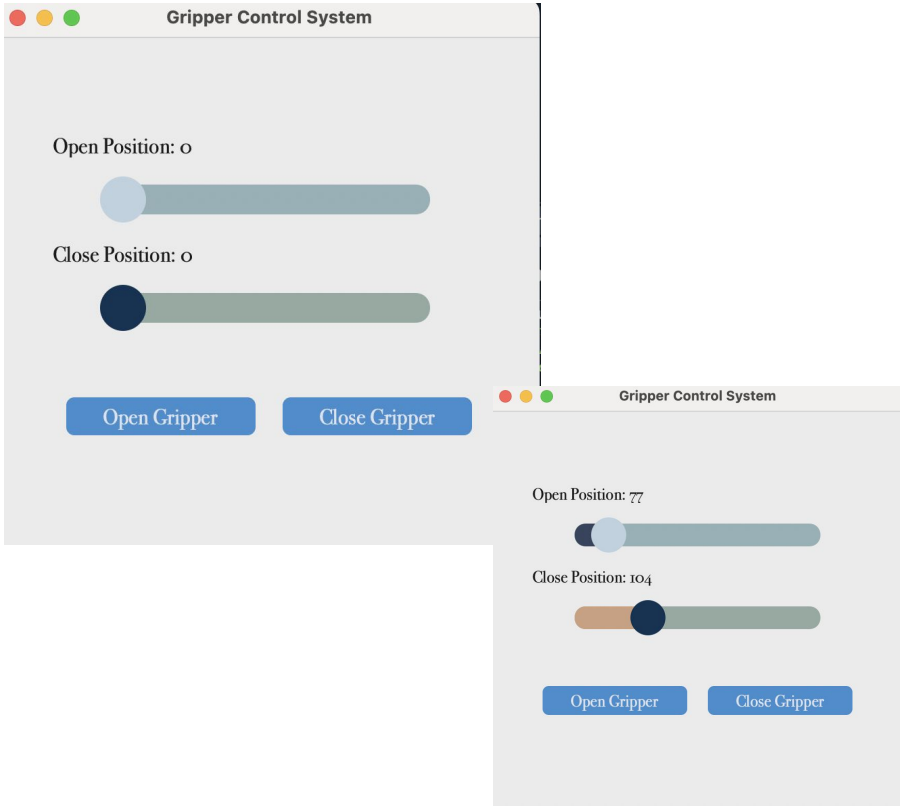
```arduino
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.setTimeout(1);

  servo.attach (A2); // Declare Servo Object (A2 = "Resume" pin)

  bufferIndex = 0;
  servo.write(75); //setting the position to 75
  delay(2000);
  servo.write(180);//setting the position to 180
  delay(2000);

} // Setup End
```

Gripper Control System GUI



Reason behind design:

- ➔ Wanted a user friendly interface that would allow us to control the gripper
- ➔ We chose to use sliders as it was easier to adjust the position of the gripper more precisely

Strengthens:

- ➔ Easy to understand and to control the open, close interface. Or setting a position for the gripper.
- ➔ When the Open or Close sliders are adjusted the position of the servo's angle updates.

Experiences:

- ➔ We ran tests and made adjustments to the code while working with this control system to improve useability and underlying bugs in the code
- ➔ Understanding what is happening when the buttons are clicked or when the sliders are being adjusted.

Limitations of the Design

Limitations and Possible Solutions:

- Left_and_Right_nails file for gripper clamps are designed molded together, which made it difficult to print
  - Solution: Separating the left and right clamps so that it prints better in design phase before printing for easier assembly process. Gears won't be stuck together, which makes for better functionality
- Gripper comes with slot for servo but at times can be difficult to hold due to overheating
  - Solution: Create a separate part/stand that can attach to clamp or servo so there is no direct skin contact
- Gripper is only able to open and close
  - Solution: Possibly design code that allows gripper to twist in different directions without having to assist the gripper manually
- Gripper sometimes does not close in the exact same way due to overheating
  - Solution: Using a rubber band that helps close the gripper freely by using it's resistance. Helps to not force the clamp close and possibly damaging it and overworking the servo, giving it time to rest

Conclusion

In summary, we chose the gripper because it met the project's objectives, which included easy construction, simple troubleshooting, and diversity in gripping things. We faced difficulties like erroneous printing, assembly changes, and servo overheating, even though the gripper design had advantages including ease of printing and functionality for small to medium-sized objects.

Our choice to employ sliders for control provided a user-friendly interface that allowed for accurate adjustments and ongoing development via iterative testing and code modifications. We identified potential solutions, such as redesigning parts for better printing and investigating alternative closing methods, despite its limitations, which included the inability to perform twisting motions and occasionally inconsistent gripping due to overheating.

In the future, resolving these issues will improve the gripper's functionality and performance, guaranteeing its efficacy in a range of applications that fall under the purview of our project.