

Team 21 Plagiarism Document

Offending Teammate:

Sebastian Reel

Team Position:

Project Manager (unofficial)/GUI Developer

Abstract

Plagiarism is an inexcusable offense under any circumstances, and our team does not condone nor support such behavior. The following document will outline three major occurrences of plagiarism that have occurred within the last year by one of our teammates, Sebastian Reel. We suspect that there may be other smaller instances, but these three incidents are the most significant and directly provable. Having incidents ranging from stealing code from an online tutorial to claiming code from another teammate as his own, Sebastian has committed unforgivable and impermissible actions during the development of our software engineering project. This behavior, compounded with the fact that he has made little to no contribution to the project's development as a whole, has caused issues with development and has negatively impacted our ability to trust him as a fellow team member. We hope that you will take these findings into consideration when evaluating his performance on the senior project.

Reading Notes:

- We have provided figures under each incident for ease of reference. For clarity purposes, feel free to open the provided links to view the raw code and commits.
- Each incident format will follow the following: Header, Original Code, Sebastian's Code, Notes, and Figures.

Plagiarism Occurrences:

Incident 1: *Accrediting himself for another party's work.*

Sebastian's 2022, November 29th Commit: 423d9a9 - link	
Tutorial Code - link	Sebastian's Code - link 1 , link 2
<p>(Check Link)</p> <p>The link requires an account to register with, we recommend using: https://tempail.com/ for this.</p> <p>After the account is created, use the forgot password feature to set your password since the login credentials don't always get emailed with the first registration.</p> <p>For data_collection.py similarities, the file you will need to look at is named: DataCollection</p> <p>For abc_demo.py similarities, the file you will need to look at is named: Testing</p>	<p>[File: data_collection.py]</p> <p>Similarities:</p> <p><u>Lines 7-63</u>: are exactly the same as the tutorial code. Except for Sebastian's <u>line 19</u> which says "images/C" instead of "Data/C".</p> <p>[File: abc_demo.py]</p> <p>Similarities:</p> <p><u>Lines 7-72</u>: Are extremely similar, the only differences are as follows:</p> <ul style="list-style-type: none">- Sebastian has wrapped the majority of the program in a main statement.- Sebastian has lowercase'd the folder "Model"'s name.
Notes	
Sebastian takes full credit for the code and does not credit the original tutorial from which it was taken from. He mentions this in the code, which we can see clearly in <u>lines 1-5</u> .	

Hand Sign Detection (ASL)

☐ Video Lesson☒ Code and Files

Data Collection

```
1 import cv2
2 from cvzone.HandTrackingModule import HandDetector
3 import numpy as np
4 import math
5 import time
6
7 cap = cv2.VideoCapture(0)
8 detector = HandDetector(maxHands=1)
9
10 offset = 20
11 imgSize = 300
12
13 folder = "Data/C"
14 counter = 0
15
16 while True:
17     success, img = cap.read()
18     hands, img = detector.findHands(img)
19     if hands:
20         hand = hands[0]
21         x, y, w, h = hand['bbox']
22
23         imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255
24         imgCrop = img[y - offset:y + h + offset, x - offset:x + w + offset]
25
26         imgCropShape = imgCrop.shape
27
28         aspectRatio = h / w
29
30         if aspectRatio > 1:
31             k = imgSize / h
32             wCal = math.ceil(k * w)
33             imgResize = cv2.resize(imgCrop, (wCal, imgSize))
34             imgResizeShape = imgResize.shape
35             wGap = math.ceil((imgSize - wCal) / 2)
36             imgWhite[:, wGap:wCal + wGap] = imgResize
37
38         else:
39             k = imgSize / w
40             hCal = math.ceil(k * h)
41             imgResize = cv2.resize(imgCrop, (imgSize, hCal))
42             imgResizeShape = imgResize.shape
43             hGap = math.ceil((imgSize - hCal) / 2)
44             imgWhite[hGap:hCal + hGap, :] = imgResize
45
46         cv2.imshow("ImageCrop", imgCrop)
47         cv2.imshow("ImageWhite", imgWhite)
48
49         cv2.imshow("Image", img)
50         key = cv2.waitKey(1)
51         if key == ord("s"):
52             counter += 1
53             cv2.imwrite(f'{folder}/Image_{time.time()}.jpg', imgWhite)
54             print(counter)
```

Figure 1.1: Tutorial code from CV Zone that manages the data collection process for hand sign detection.

```

63 lines (50 sloc) | 1.84 KB
1  # Author: Sebastian Reel
2  # Project: ASL Hand Demo
3  # Description: Utilizing opencv, and a premade Google machine learning algorithm, named Teachable Machine.(which uses Tensorflow)
4  # This will demo the possibility of presenting ASL letters A, B, and C, to demonstrate the goal for
5  # detecting letters and confirming the user has shown their hand correctly.
6
7  import cv2
8  from cvzone.HandTrackingModule import HandDetector
9  import numpy as np
10 import math
11 import time
12
13 cap = cv2.VideoCapture(0)
14 detector = HandDetector(maxHands=1)
15
16 offset = 20
17 imgSize = 300
18
19 folder = "images/C/"
20 counter = 0
21
22 while True:
23     success, img = cap.read()
24     hands, img = detector.findHands(img)
25     if hands:
26         hand = hands[0]
27         x, y, w, h = hand['bbox']
28
29         imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255
30         imgCrop = img[y - offset:y + h + offset, x - offset:x + w + offset]
31
32         imgCropShape = imgCrop.shape
33
34         aspectRatio = h / w
35
36         if aspectRatio > 1:
37             k = imgSize / h
38             wCal = math.ceil(k * w)
39             imgResize = cv2.resize(imgCrop, (wCal, imgSize))
40             imgResizeShape = imgResize.shape
41             wGap = math.ceil((imgSize - wCal) / 2)
42             imgWhite[:, wGap:wCal + wGap] = imgResize
43
44         else:
45             k = imgSize / w
46             hCal = math.ceil(k * h)
47             imgResize = cv2.resize(imgCrop, (imgSize, hCal))
48             imgResizeShape = imgResize.shape
49             hGap = math.ceil((imgSize - hCal) / 2)
50             imgWhite[hGap:hCal + hGap, :] = imgResize
51
52         cv2.imshow("ImageCrop", imgCrop)
53         cv2.imshow("ImageWhite", imgWhite)
54
55
56         cv2.imshow("Image", img)
57         key = cv2.waitKey(1)
58         if key == ord("s"):
59             counter += 1
60             cv2.imwrite(f'{folder}/Image_{time.time()}.jpg', imgWhite)
61             print(counter)
62         if key == ord("q"):
63             break

```

Figure 1.2: Code that Sebastian states he is the author of and is almost identical to Figure 1.1.

Incident 2: Accrediting himself for another group member's work.

Sebastian's 2023, March 8th Commit - "theme fixes": 8de9b3c - link	
Jasons 2022, December 6th Code - link	Sebastian's 2023, March 8th Code - link
<p>[File: ASL_GUI.pyw]</p> <p><u>Lines 63-131:</u> Jason's Original Code.</p>	<p>[File: ASL_GUI.pyw]</p> <p>Similarities: <u>Lines 292-348:</u> These lines are almost one-to-one with Jason's original code. The changes made here include replacing hard-coded color values with ones stored in variables.</p>
Notes	
<p>The significance of these commits is that the function themes_button is almost identical. Sebastian's changes consist of creating global THEME color variables (THEME, THEME_OPP, FONT) which hold the associated hex color codes in a string. He then replaced all the corresponding attribute values in ASL_GUI (throughout the next few commits). Sebastian takes full credit for the theme system, where the majority of it is actually Jason's work.</p>	

```

62 # Button that recreates window with the theme page
63 def themes_button(self):
64
65     self.window_state = WindowState.THEMES
66
67     # Destroyed old window
68     self.frame_left.destroy()
69     self.frame_right.destroy()
70
71     # Creates theme left window
72     # -----
73     self.frame_left = customtkinter.CTkFrame(master=self, width=200, corner_radius=0)
74     self.frame_left.grid(row=0, column=0, sticky="nswe")
75
76     # Creates labels for the left window
77     # -----
78
79     # Creates label with the text "Users:"
80     self.label_1 = customtkinter.CTkLabel(master=self.frame_left, text="Users:")
81     self.label_1.grid(row=0, column=0, padx=10, pady=10, sticky="we")
82
83     # Default user settings button to reset all changes
84     self.button1 = customtkinter.CTkButton(master=self.frame_left, text = "Default User Settings", width = 200, height = 50, border_width = 1, corner_radius = 5, compound
85     self.button1.grid(row=1, column=0, padx=1, pady=1)
86
87     # Creates Return button
88     self.button2 = customtkinter.CTkButton(master=self.frame_left, text = "Return", width = 200, height = 60, border_width = 1, corner_radius = 5, compound = "bottom", fg
89     self.button2.grid(row=0, column=0, padx=0, pady=350, sticky="we")
90
91     # Creates the right side of the theme wind
92     # -----
93     self.frame_right = customtkinter.CTkFrame(master=self, corner_radius = 0, fg_color = "#303030")
94     self.frame_right.grid(row=0, column=1, sticky="nswe")
95
96     # Creates labels for the right window
97     # -----
98
99     # Creates label with the text "Overall Theme Settings:" to describe what the drop down menu below it does
100    self.label_2 = customtkinter.CTkLabel(master=self.frame_right, text="Overall Theme Settings:")
101    self.label_2.grid(row=0, column=0, padx=5, pady=5, sticky="w")
102
103    # Creates theme drop down menu to change general theme details all at once
104    self.optionmenu_1 = customtkinter.CTkOptionMenu(master=self.frame_right, values=["System", "Light", "Dark"], fg_color = "#292929", command=self.change_appearance_mode
105    self.optionmenu_1.grid(row=0, column=1, padx=5, pady=10, sticky="w")
106
107    # Creates label with the text "Font Size:" to describe what the slider below it does
108    self.label_3 = customtkinter.CTkLabel(master=self.frame_right, text="Font Size:")
109    self.label_3.grid(row=1, column=0, padx=5, pady=5, sticky="w")
110
111    # Creates font size progress bar to change font size of application
112    self.progressbar = customtkinter.CTkProgressBar(master=self)
113
114    self.slider_1 = customtkinter.CTkSlider(master=self.frame_right,
115                                           from_=0,
116                                           to=1,
117                                           number_of_steps=10,
118                                           command=self.progressbar.set)
119    self.slider_1.grid(row=2, column=0, columnspan=2, pady=10, padx=20, sticky="we")
120
121    # Creates label with the text "Button Size:" to describe what the slider below it does
122    self.label_3 = customtkinter.CTkLabel(master=self.frame_right, text="Button Size:")
123    self.label_3.grid(row=3, column=0, padx=5, pady=5, sticky="w")
124
125    # Creates font size progress bar to change font size of application
126    self.slider_2 = customtkinter.CTkSlider(master=self.frame_right,
127                                           from_=0,
128                                           to=1,
129                                           number_of_steps=10,
130                                           command=self.progressbar.set)
131    self.slider_2.grid(row=4, column=0, columnspan=2, pady=10, padx=20, sticky="we")

```

Figure 2.1: Jason's theme button function from last semester.

```

291 # Button that recreates window with the theme page
292 def themes_button(self):
293
294     # Destroyed old window
295     self.frame_left.destroy()
296     self.frame_right.destroy()
297
298     # Creates theme left window
299     # -----
300     self.frame_left = customtkinter.CTkFrame(master=self, width=200, corner_radius=0)
301     self.frame_left.grid(row=0, column=0, sticky="nswe")
302
303     # Creates labels for the left window
304     # -----
305
306     # Creates label with the text "Users:"
307     self.label_1 = customtkinter.CTkLabel(master=self.frame_left, text="Users:", text_color = THEME_OPP)
308     self.label_1.grid(row=0, column=0, padx=10, pady=10, sticky="nw")
309
310     # Default user settings button to reset all changes
311     self.button1 = customtkinter.CTkButton(master=self.frame_left, text = "Default User Settings", text_color = THEME_OPP, width = 200, height = 50, border_width = 1, corner_radius = 5)
312     self.button1.grid(row=1, column=0, padx=1, pady=1)
313
314     # Creates Return button
315     self.button2 = customtkinter.CTkButton(master=self.frame_left, text = "Return", text_color = THEME_OPP, width = 200, height = 60, border_width = 1, corner_radius = 5,
316     self.button2.grid(row=9, column=0, padx=0, pady=350, sticky="nw")
317
318     # Creates the right side of the theme window
319     # -----
320     self.frame_right = customtkinter.CTkFrame(master=self, corner_radius = 0)
321     self.frame_right.grid(row=0, column=1, sticky="nswe")
322
323     # Creates labels for the right window
324     # -----
325
326     # Creates label with the text "Overall Theme Settings:" to describe what the drop down menu below it does
327     self.label_2 = customtkinter.CTkLabel(master=self.frame_right, text="Overall Theme Settings:", text_color = THEME_OPP)
328     self.label_2.grid(row=0, column=0, padx=5, pady=5, sticky="nw")
329
330     # Creates theme drop down menu to change general theme details all at once
331     self.optionmenu_1 = customtkinter.CTkOptionMenu(master=self.frame_right, values=["Dark", "Light"], text_color = THEME_OPP, fg_color = THEME, command=self.change_appearance)
332     self.optionmenu_1.grid(row=0, column=1, padx=5, pady=10, sticky="nw")
333
334     # Creates label with the text "Font Size:" to describe what the slider below it does
335     self.label_3 = customtkinter.CTkLabel(master=self.frame_right, text="Font Size:", text_color = THEME_OPP)
336     self.label_3.grid(row=1, column=0, padx=5, pady=5, sticky="nw")
337
338     # Creates font size progress bar to change font size of application
339     self.progressbar = customtkinter.CTkProgressBar(master=self)
340
341     self.slider_1 = customtkinter.CTkSlider(master=self.frame_right,
342     from_=0,
343     to=1,
344     number_of_steps=10,
345     command=self.progressbar.set)
346     self.slider_1.grid(row=2, column=0, columnspan=2, pady=10, padx=20, sticky="nw")
347
348     # Creates label with the text "Button Size:" to describe what the slider below it does
349     self.label_3 = customtkinter.CTkLabel(master=self.frame_right, text="Button Size:")
350     self.label_3.grid(row=3, column=0, padx=5, pady=5, sticky="nw")
351
352     # Creates font size progress bar to change font size of application
353     self.slider_2 = customtkinter.CTkSlider(master=self.frame_right,
354     #
355     from_=0,
356     to=1,
357     number_of_steps=10,
358     command=self.progressbar.set)
359     self.slider_2.grid(row=4, column=0, columnspan=2, pady=10, padx=20, sticky="nw")

```

Figure 2.2: Sebastian's changes to the theme button and the work that he claims is his.

Incident 3: Accrediting himself for another group member's work.

Sebastian's 2023, April 24 Commit - "Theme Page Fix, Updated, Useless Functions Removed": 8147fc6 - link	
Jason's 2022, December 11th Commit 902d720 - link	Sebastian's 2023, April 24th Code - link
<p>[File: ASL_GUI.pyw]</p> <p><u>Lines 116-212 (right number column on the commit):</u> Jason's Original Code.</p>	<p>[File: ASL_GUI.pyw]</p> <p>Similarities: <u>Lines 419-544:</u> These lines are one-to-one with Jason's original commit, the only differences are <u>lines 420-428</u> in which 5 were added by other members. There are 3 lines that are currently unknown who added them.</p>
Notes	
<p>The significance of this commit is that Sebastian takes full credit for <u>lines 419-544</u> (link), the user configuration UI. The <u>lines 423-425</u> were added by Keith and <u>lines 420, 421, and 427</u> are currently unknown from who they came from. The only code that is Sebastian's are <u>lines 471-491</u> (commented out by him), and potentially the 3 unknown lines. The vast majority of these lines are Jason's and Sebastian takes full credit for them. Sebastian was in a meeting with the rest of the team and our official project manager and passed the entire code off to our team's project manager as 100% his. Please note that in the commit linked, there is no direct addition of these lines. This is because Sebastian never added, nor removed, these lines from the code. The lines of code he attempts to take credit for were already in the codebase.</p>	

Showing 1 changed file with 214 additions and 88 deletions.

```

158 + #Button mapping and functionality
159 + # PLEASE ATTACH BUTTON FUNCTIONALITY FOR ALL THESE BUTTONS
160 + self.buttonA = customtkinter.CTkButton(master=self.frame_right, text = "A", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_A)
161 + self.buttonA.grid(row=2, column=1, padx=15, pady=10, sticky="w")
162 + self.buttonB = customtkinter.CTkButton(master=self.frame_right, text = "B", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_B)
163 + self.buttonB.grid(row=2, column=2, padx=15, pady=10, sticky="w")
164 + self.buttonC = customtkinter.CTkButton(master=self.frame_right, text = "C", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_C)
165 + self.buttonC.grid(row=2, column=3, padx=15, pady=10, sticky="w")
166 + self.buttonD = customtkinter.CTkButton(master=self.frame_right, text = "D", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_D)
167 + self.buttonD.grid(row=2, column=4, padx=15, pady=10, sticky="w")
168 + self.buttonE = customtkinter.CTkButton(master=self.frame_right, text = "E", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_E)
169 + self.buttonE.grid(row=2, column=5, padx=15, pady=10, sticky="w")
170 + self.buttonF = customtkinter.CTkButton(master=self.frame_right, text = "F", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_F)
171 + self.buttonF.grid(row=2, column=6, padx=15, pady=10, sticky="w")
172 + self.buttonG = customtkinter.CTkButton(master=self.frame_right, text = "G", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_G)
173 + self.buttonG.grid(row=3, column=1, padx=15, pady=10, sticky="w")
174 + self.buttonH = customtkinter.CTkButton(master=self.frame_right, text = "H", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_H)
175 + self.buttonH.grid(row=3, column=2, padx=15, pady=10, sticky="w")
176 + self.buttonI = customtkinter.CTkButton(master=self.frame_right, text = "I", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_I)
177 + self.buttonI.grid(row=3, column=3, padx=15, pady=10, sticky="w")
178 + self.buttonJ = customtkinter.CTkButton(master=self.frame_right, text = "J", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_J)
179 + self.buttonJ.grid(row=3, column=4, padx=15, pady=10, sticky="w")
180 + self.buttonK = customtkinter.CTkButton(master=self.frame_right, text = "K", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_K)
181 + self.buttonK.grid(row=3, column=5, padx=15, pady=10, sticky="w")
182 + self.buttonL = customtkinter.CTkButton(master=self.frame_right, text = "L", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_L)
183 + self.buttonL.grid(row=3, column=6, padx=15, pady=10, sticky="w")
184 + self.buttonM = customtkinter.CTkButton(master=self.frame_right, text = "M", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_M)
185 + self.buttonM.grid(row=4, column=1, padx=15, pady=10, sticky="w")
186 + self.buttonN = customtkinter.CTkButton(master=self.frame_right, text = "N", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_N)
187 + self.buttonN.grid(row=4, column=2, padx=15, pady=10, sticky="w")
188 + self.buttonO = customtkinter.CTkButton(master=self.frame_right, text = "O", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_O)
189 + self.buttonO.grid(row=4, column=3, padx=15, pady=10, sticky="w")
190 + self.buttonP = customtkinter.CTkButton(master=self.frame_right, text = "P", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_P)
191 + self.buttonP.grid(row=4, column=4, padx=15, pady=10, sticky="w")
192 + self.buttonQ = customtkinter.CTkButton(master=self.frame_right, text = "Q", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_Q)
193 + self.buttonQ.grid(row=4, column=5, padx=15, pady=10, sticky="w")
194 + self.buttonR = customtkinter.CTkButton(master=self.frame_right, text = "R", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_R)
195 + self.buttonR.grid(row=4, column=6, padx=15, pady=10, sticky="w")
196 + self.buttonS = customtkinter.CTkButton(master=self.frame_right, text = "S", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_S)
197 + self.buttonS.grid(row=5, column=1, padx=15, pady=10, sticky="w")
198 + self.buttonT = customtkinter.CTkButton(master=self.frame_right, text = "T", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_T)
199 + self.buttonT.grid(row=5, column=2, padx=15, pady=10, sticky="w")
200 + self.buttonU = customtkinter.CTkButton(master=self.frame_right, text = "U", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_U)
201 + self.buttonU.grid(row=5, column=3, padx=15, pady=10, sticky="w")
202 + self.buttonV = customtkinter.CTkButton(master=self.frame_right, text = "V", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_V)
203 + self.buttonV.grid(row=5, column=4, padx=15, pady=10, sticky="w")
204 + self.buttonW = customtkinter.CTkButton(master=self.frame_right, text = "W", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_W)
205 + self.buttonW.grid(row=5, column=5, padx=15, pady=10, sticky="w")
206 + self.buttonX = customtkinter.CTkButton(master=self.frame_right, text = "X", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_X)
207 + self.buttonX.grid(row=5, column=6, padx=15, pady=10, sticky="w")
208 + self.buttonY = customtkinter.CTkButton(master=self.frame_right, text = "Y", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_Y)
209 + self.buttonY.grid(row=6, column=3, padx=15, pady=10, sticky="w")
210 + self.buttonZ = customtkinter.CTkButton(master=self.frame_right, text = "Z", width = 55, height = 55, border_width = 1, corner_radius = 5, compound = "bottom", fg_color = "#292929", border_color="#101010", command=self.config_Z)
211 + self.buttonZ.grid(row=6, column=4, padx=15, pady=10, sticky="w")

```

Figure 3.1: The original configuration UI code from Jason's commit last semester.

```

470 # attempting to make this dynamic, having issues
471 ...
472 for i in range(trainlist_length):
473     if self.usertrain_letters[0:6]:
474         letter_button = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[i], text_color = THEME_OPP, width = 55, height = 55, border_wi
475         #print(self.usertrain_letters[0:6])
476         letter_button.grid(row = 2, column = i, padx = 15, pady = 10)
477
478     if self.usertrain_letters[6:12]:
479         print(self.usertrain_letters[6:12])
480         letter_button.grid(row = 3, column = 1, colspan = 6, padx = 15, pady = 10)
481     if self.usertrain_letters[12:18]:
482         print(self.usertrain_letters[12:18])
483         letter_button.grid(row = 4, column = 1, colspan = 6, padx = 15, pady = 10)
484     if self.usertrain_letters[18:24]:
485         print(self.usertrain_letters[18:24])
486         letter_button.grid(row = 5, column = 1, colspan = 6, padx = 15, pady = 10)
487
488     self.usertrain_letters.append(letter_button)
489 self.del_list = self.usertrain_letters
490 ...
491
492 self.buttonA = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[0], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
493 self.buttonA.grid(row=2, column=1, padx=15, pady=10, sticky="w")
494 self.buttonB = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[1], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
495 self.buttonB.grid(row=2, column=2, padx=15, pady=10, sticky="w")
496 self.buttonC = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[2], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
497 self.buttonC.grid(row=2, column=3, padx=15, pady=10, sticky="w")
498 self.buttonD = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[3], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
499 self.buttonD.grid(row=2, column=4, padx=15, pady=10, sticky="w")
500 self.buttonE = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[4], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
501 self.buttonE.grid(row=2, column=5, padx=15, pady=10, sticky="w")
502 self.buttonF = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[5], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
503 self.buttonF.grid(row=2, column=6, padx=15, pady=10, sticky="w")
504 self.buttonG = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[6], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
505 self.buttonG.grid(row=3, column=1, padx=15, pady=10, sticky="w")
506 self.buttonH = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[7], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
507 self.buttonH.grid(row=3, column=2, padx=15, pady=10, sticky="w")
508 self.buttonI = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[8], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
509 self.buttonI.grid(row=3, column=3, padx=15, pady=10, sticky="w")
510 self.buttonJ = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[9], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
511 self.buttonJ.grid(row=3, column=4, padx=15, pady=10, sticky="w")
512 self.buttonK = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[10], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
513 self.buttonK.grid(row=3, column=5, padx=15, pady=10, sticky="w")
514 self.buttonL = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[11], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
515 self.buttonL.grid(row=3, column=6, padx=15, pady=10, sticky="w")
516 self.buttonM = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[12], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
517 self.buttonM.grid(row=4, column=1, padx=15, pady=10, sticky="w")
518 self.buttonN = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[13], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
519 self.buttonN.grid(row=4, column=2, padx=15, pady=10, sticky="w")
520 self.buttonO = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[14], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
521 self.buttonO.grid(row=4, column=3, padx=15, pady=10, sticky="w")
522 self.buttonP = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[15], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
523 self.buttonP.grid(row=4, column=4, padx=15, pady=10, sticky="w")
524 self.buttonQ = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[16], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
525 self.buttonQ.grid(row=4, column=5, padx=15, pady=10, sticky="w")
526 self.buttonR = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[17], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
527 self.buttonR.grid(row=4, column=6, padx=15, pady=10, sticky="w")
528 self.buttonS = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[18], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
529 self.buttonS.grid(row=5, column=1, padx=15, pady=10, sticky="w")
530 self.buttonT = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[19], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
531 self.buttonT.grid(row=5, column=2, padx=15, pady=10, sticky="w")
532 self.buttonU = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[20], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
533 self.buttonU.grid(row=5, column=3, padx=15, pady=10, sticky="w")
534 self.buttonV = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[21], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
535 self.buttonV.grid(row=5, column=4, padx=15, pady=10, sticky="w")
536 self.buttonW = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[22], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
537 self.buttonW.grid(row=5, column=5, padx=15, pady=10, sticky="w")
538 self.buttonX = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[23], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
539 self.buttonX.grid(row=5, column=6, padx=15, pady=10, sticky="w")
540 self.buttonY = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[24], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
541 self.buttonY.grid(row=6, column=3, padx=15, pady=10, sticky="w")
542 self.buttonZ = customtkinter.CTkButton(master=self.frame_right, text = self.usertrain_letters[25], text_color = THEME_OPP, width = 55, height = 55, border_width = 1,
543 self.buttonZ.grid(row=6, column=4, padx=15, pady=10, sticky="w")
544

```

Figure 3.2: Sebastian's changes to the configuration UI and his claimed work.

