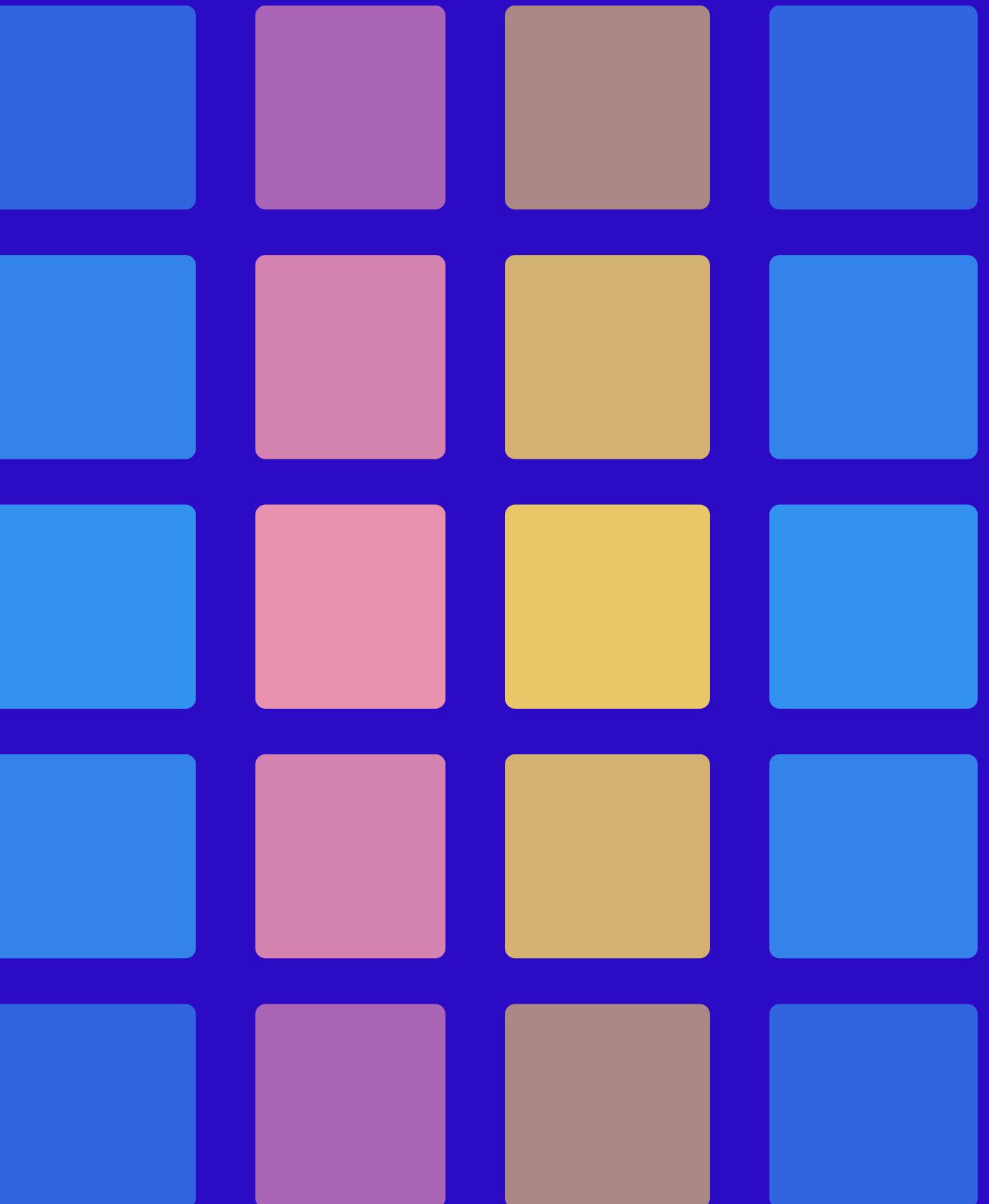


# Synapse

an interactive  
installation



# The Concept

This Installation is an attempt to explore how technology often tries to replicate human connection by responding to their emotions via algorithms.

We looked at AI chatbots that often tend to tell the user exactly what they want to hear and devices such as Magic 8 Ball which help people make decisions by deluding them into thinking that it can see the future.

Our installation follows the principles of English sentence formation to give the user an advisory or observational response to their emotion along with the visual element of a pop-out card which can then be interpreted by the user in various ways.

# How to Interact with Synapse

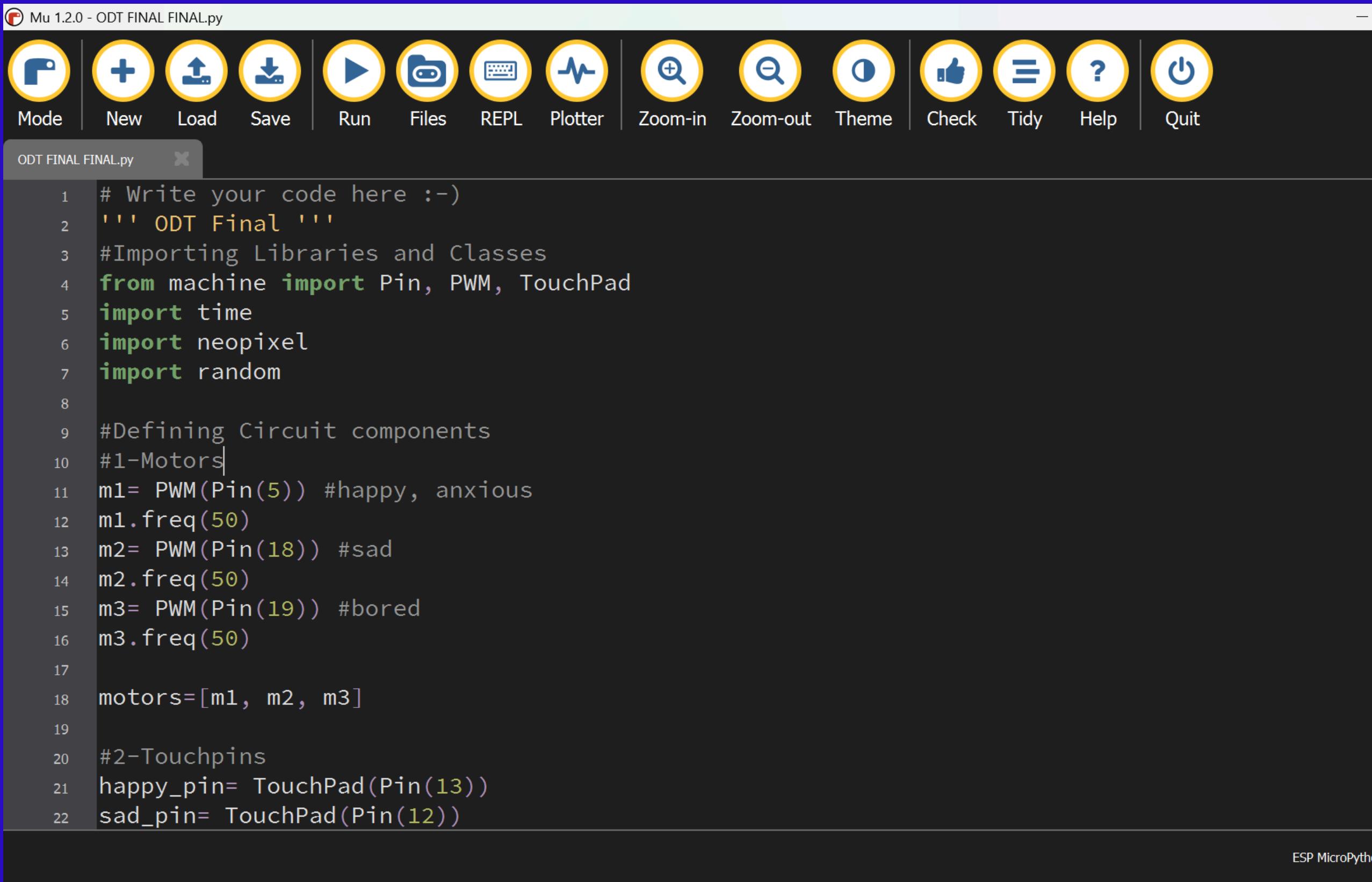
1

The user touches the Pin that corresponds to their emotion- happy, sad, anxious or bored.

2

The installation then lights up a series of words on the board that form a sentence that the user must read and decipher for themselves. They can treat this sentence as a forecast of their future or an observation of their present.

# The Code



Mu 1.2.0 - ODT FINAL FINAL.py

Mode New Load Save Run Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help Quit

ODT FINAL FINAL.py

```
1 # Write your code here :-
2 ''' ODT Final '''
3 #Importing Libraries and Classes
4 from machine import Pin, PWM, TouchPad
5 import time
6 import neopixel
7 import random
8
9 #Defining Circuit components
10 #1-Motors|
11 m1= PWM(Pin(5)) #happy, anxious
12 m1.freq(50)
13 m2= PWM(Pin(18)) #sad
14 m2.freq(50)
15 m3= PWM(Pin(19)) #bored
16 m3.freq(50)
17
18 motors=[m1, m2, m3]
19
20 #2-Touchpins
21 happy_pin= TouchPad(Pin(13))
22 sad_pin= TouchPad(Pin(12))
```

ESP MicroPython

Mu 1.2.0 - ODT FINAL FINAL.py

The image shows the Mu 1.2.0 IDE interface. The top bar features a toolbar with various icons: Mode, New, Load, Save, Run, Files, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Tidy, Help, and Quit. The main window displays a Python script named 'ODT FINAL FINAL.py'. The code is as follows:

```
20 #2-Touchpins
21 happy_pin= TouchPad(Pin(13))
22 sad_pin= TouchPad(Pin(12))
23 bored_pin= TouchPad(Pin(4)) #red
24 anxious_pin= TouchPad(Pin(15)) #green
25
26 #3-LEDs
27 line_1= neopixel.NeoPixel(Pin(22), 23)
28 line_2= neopixel.NeoPixel(Pin(23), 17)
29 line_3= neopixel.NeoPixel(Pin(32), 22)
30 line_4= neopixel.NeoPixel(Pin(33), 23)
31 line_5= neopixel.NeoPixel(Pin(25), 23)
32 line_6= neopixel.NeoPixel(Pin(26), 23)
33 line_7= neopixel.NeoPixel(Pin(14), 19)
34 line_8= neopixel.NeoPixel(Pin(27), 20)
35
36 #Categorising words according to grammar
37 subject= ['you', line_1, [0,3]]
38 modal_verb= [['must', line_1, [4, 9]], ['should', line_1, [10,17]], ['can', line_1, [18, 22]
39 negative= [['not',line_2, [13, 16]], ['unrequired']]
40 verb= [['believe',line_3, [0, 7]], ['live',line_3, [8, 13]], ['create', line_3, [14, 21]], [
41 objectt= [['now', line_7, [0, 2]], ['always', line_7, [3, 11]], ['right', line_7, [12, 18]]]
```

At the bottom right of the code editor, it says 'ESP MicroPython'.

Mu 1.2.0 - ODT FINAL FINAL.py

The image shows the Mu 1.2.0 Python IDE interface. The title bar reads "Mu 1.2.0 - ODT FINAL FINAL.py". The menu bar includes "File", "Edit", "Run", "REPL", "Plotter", "Help", and "About". The toolbar contains icons for Mode, New, Load, Save, Run, Files, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Tidy, Help, and Quit. The main window displays the following Python code:

```
36 #Categorising words according to grammar
37 subject= ['you', line_1, [0,3]]
38 modal_verb= [['must', line_1, [4, 9]], ['should', line_1, [10,17]], ['can', line_1, [18, 22]]
39 negative= [['not',line_2, [13, 16]], ['unrequired']]
40 verb= [['believe',line_3, [0, 7]], ['live',line_3, [8, 13]], ['create', line_3, [14, 21]], [
41 objectt= [['now', line_7, [0, 2]], ['always', line_7, [3, 11]], ['right', line_7, [12, 18]],
42
43
44 def sentence_maker():
45     sub= subject[0]
46     sub_line= subject[1]
47     sub_led_start= subject[2][0]
48     sub_led_stop= subject[2][1]
49     for i in range(sub_led_start, sub_led_stop+1):
50         sub_line[i]=(255,255,255)
51         sub_line.write()
52
53     mv_choice= random.randint(0, len(modal_verb)-1)
54     mv= modal_verb[mv_choice][0]
55     mv_line=modal_verb[mv_choice][1]
56     mv_led_start= modal_verb[mv_choice][2][0]
57     mv_led_stop= modal_verb[mv_choice][2][1]
```

The code defines a function `sentence_maker()` that generates a sentence by combining a subject, modal verb, and verb. It uses a list of subjects, a list of modal verbs, and a list of verbs. The code then iterates through the subject's led range and writes to a line. Finally, it selects a modal verb from the list and sets its led range.

Mu 1.2.0 - ODT FINAL FINAL.py

The image shows the Mu 1.2.0 Python IDE interface. The title bar reads "Mu 1.2.0 - ODT FINAL FINAL.py". The menu bar includes "File", "Edit", "Run", "REPL", "Plotter", "Tools", "Help", and "About". The toolbar contains icons for Mode, New, Load, Save, Run, Files, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Tidy, Help, and Quit. The main window displays the following Python code:

```
53 mv_choice= random.randint(0, len(modal_verb)-1)
54 mv= modal_verb[mv_choice][0]
55 mv_line=modal_verb[mv_choice][1]
56 mv_led_start= modal_verb[mv_choice][2][0]
57 mv_led_stop= modal_verb[mv_choice][2][1]
58 for i in range(mv_led_start, mv_led_stop+1):
59     mv_line[i]=(255,255,255)
60     mv_line.write()
61
62 neg_choice= random.randint(0, len(negative)-1)
63 neg= negative[neg_choice][0]
64 if neg=='not':
65     neg_line=negative[neg_choice][1]
66     neg_led_start= negative[neg_choice][2][0]
67     neg_led_stop= negative[neg_choice][2][1]
68     for i in range(neg_led_start, neg_led_stop+1):
69         neg_line[i]=(255,255,255)
70         neg_line.write()
71
72 verb_choice= random.randint(0, len(verb)-1)
73 vb= verb[verb_choice][0]
74 vb_line=verb[verb_choice][1]
```

The code uses the random module to select indices from lists named modal\_verb, negative, and verb. It then writes to objects named mv\_line, neg\_line, and vb\_line respectively, likely representing LED matrices or similar hardware.

Mu 1.2.0 - ODT FINAL FINAL.py

The Mu IDE interface is shown with a dark theme. The top bar features a file icon, followed by the title "Mu 1.2.0 - ODT FINAL FINAL.py". Below the title is a row of circular icons with yellow outlines and blue icons: Mode (document), New (+), Load (cloud), Save (disk), Run (play), Files (file folder), REPL (terminal), Plotter (line graph), Zoom-in (magnifying glass), Zoom-out (magnifying glass), Theme (circle with arrow), Check (thumb up), Tidy (equal sign), Help (?), and Quit (power). A dropdown menu is open over the "Mode" icon. The main area contains the Python code:

```
ODT FINAL FINAL.py
verb_choice= random.randint(0, len(verb)-1)
vb= verb[verb_choice][0]
vb_line=verb[verb_choice][1]
vb_led_start= verb[verb_choice][2][0]
vb_led_stop= verb[verb_choice][2][1]
for i in range(vb_led_start, vb_led_stop+1):
    vb_line[i]=(255,255,255)
    vb_line.write()

obj_choice= random.randint(0, len(objectt)-1)
obj= objectt[obj_choice][0]
obj_line=objectt[obj_choice][1]
obj_led_start= objectt[obj_choice][2][0]
obj_led_stop= objectt[obj_choice][2][1]
for i in range(obj_led_start, obj_led_stop+1):
    obj_line[i]=(255,255,255)
    obj_line.write()

def reset_leds():
    for line in [line_1, line_2, line_3, line_4, line_5, line_6, line_7, line_8]:
```

In the bottom right corner of the main window, the text "ESP MicroPyt" is visible.

Mu 1.2.0 - ODT FINAL FINAL.py

The image shows the Mu 1.2.0 Python IDE interface. The window title is "Mu 1.2.0 - ODT FINAL FINAL.py". The menu bar includes "Mode", "New", "Load", "Save", "Run", "Files", "REPL", "Plotter", "Zoom-in", "Zoom-out", "Theme", "Check", "Tidy", "Help", and "Quit". The main area displays the following Python code:

```
92 def reset_leds():
93     for line in [line_1, line_2, line_3, line_4, line_5, line_6, line_7, line_8]:
94         for i in range(len(line)):
95             line[i] = (0, 0, 0)
96             line.write()
97
98 def run_emotion(primary_motor):
99     sentence_maker()
100
101    for i in range(26, 80):
102        primary_motor.duty(i)
103        time.sleep(0.05)
104
105    time.sleep(5)
106    reset_leds()
107
108
109
110 happy_threshold= 100
111 sad_threshold= 100
112 anxious_threshold= 100
113 bored_threshold= 270
```

At the bottom right of the code editor, it says "ESP MicroPy".

Mu 1.2.0 - ODT FINAL FINAL.py

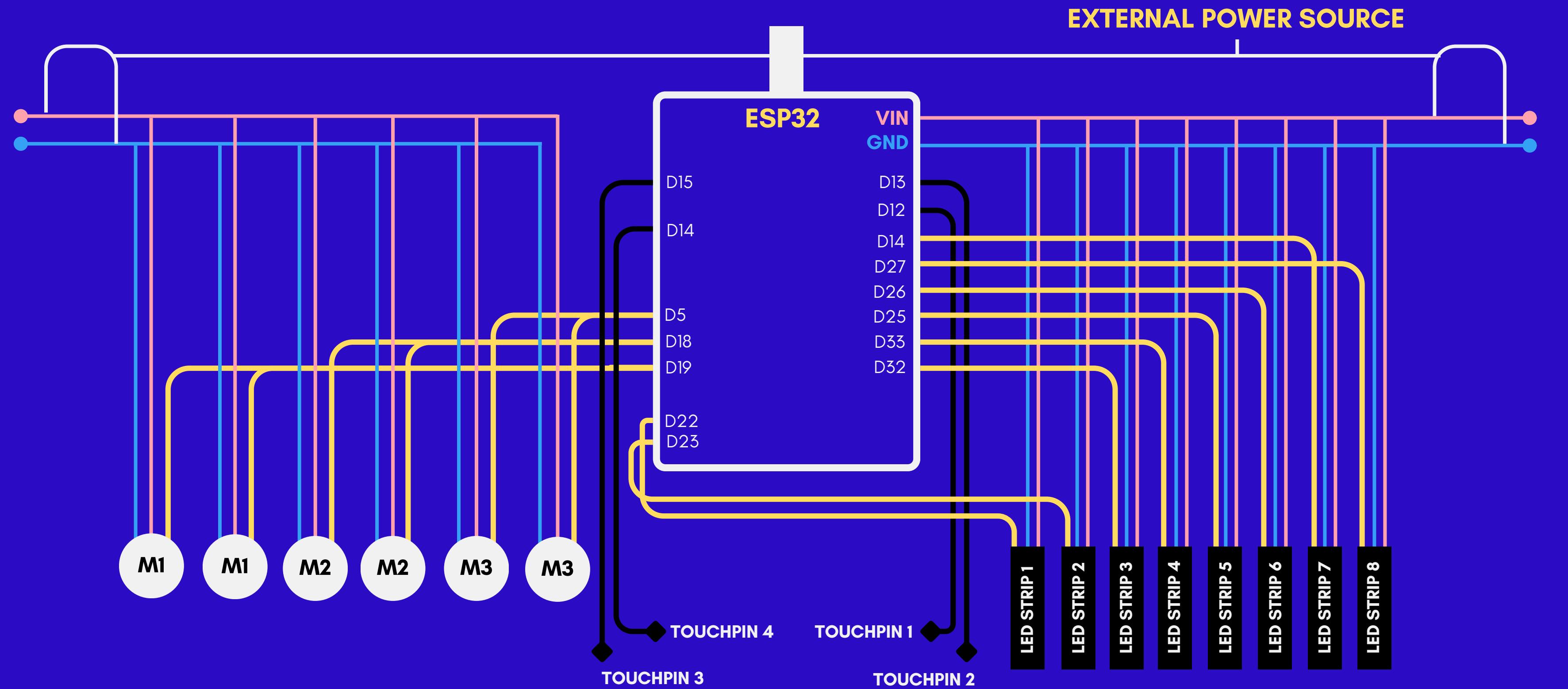
The image shows the Mu 1.2.0 Python IDE interface. The window title is "Mu 1.2.0 - ODT FINAL FINAL.py". The menu bar includes "Mode", "New", "Load", "Save", "Run", "Files", "REPL", "Plotter", "Zoom-in", "Zoom-out", "Theme", "Check", "Tidy", "Help", and "Quit". The main code editor displays the following Python script:

```
116 while True:
117     val_happy= happy_pin.read()
118     time.sleep(0.05)
119     val_sad= sad_pin.read()
120     time.sleep(0.05)
121     val_bored= bored_pin.read()
122     time.sleep(0.05)
123     val_anxious= anxious_pin.read()
124     time.sleep(0.05)

125
126     if val_happy<happy_threshold:
127         run_emotion(m1)
128
129     if val_sad<sad_threshold:
130         run_emotion(m2)
131
132     if val_bored<bored_threshold:
133         run_emotion(m3)
134
135     if val_anxious<anxious_threshold:
136         run_emotion(m1)# Write your code here :-)
```

The code uses pins to read values from sensors and triggers actions based on thresholds. The script is numbered from 116 to 137.

# The Circuit Diagram



# The Theory

We have divided the english sentence into five parts: **the singular subject, a modal verb, an optional negative word, a verb, and an object**. Since the subject is only one, conjugation isn't required. Hence, different permutation combinations of the modal verb, verb, and object create various sentences.

When the user touches a touchpin, the code randomly selects words from each category of words except the subject- keeping the negative as an option.

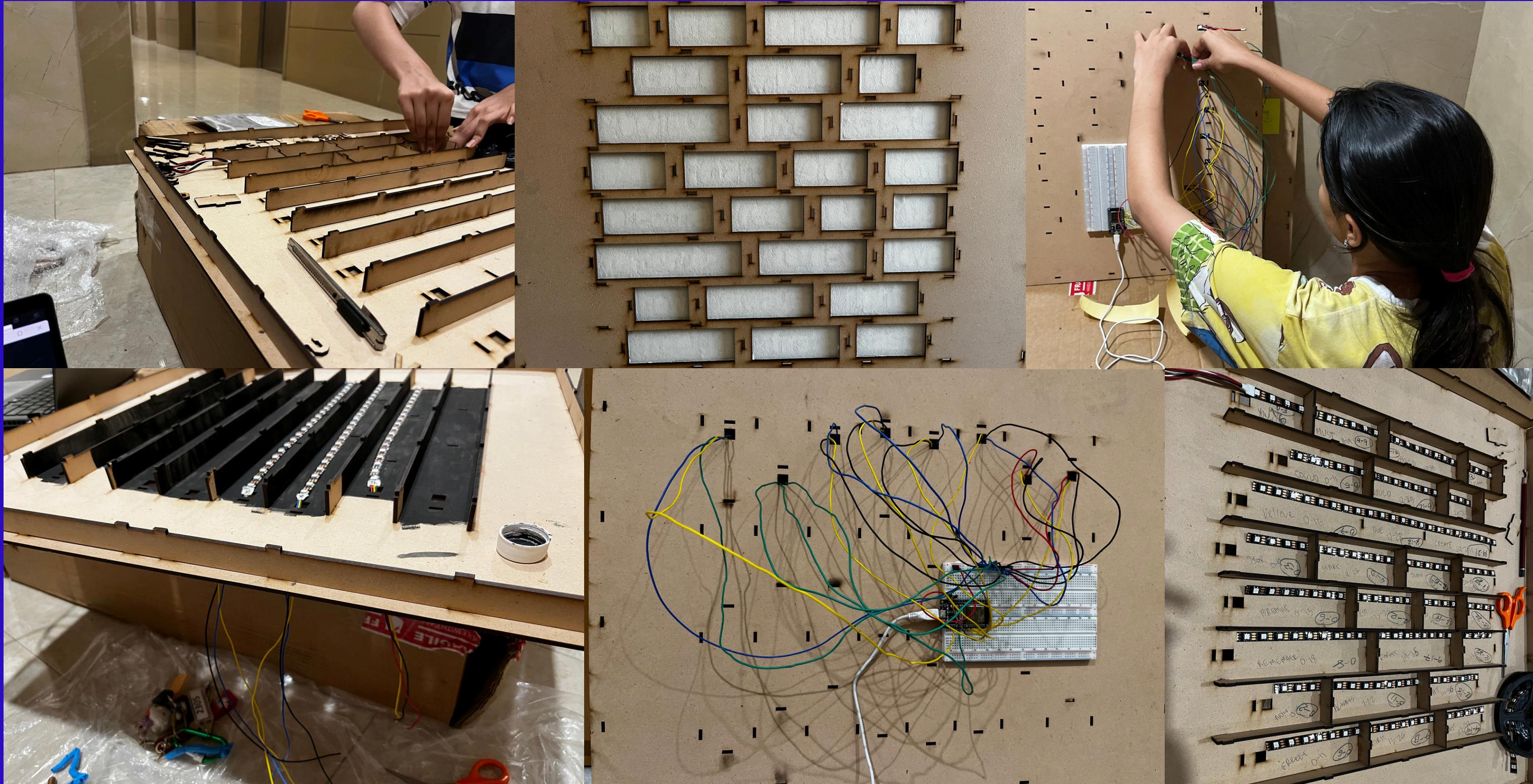
According to the sentence, a random pop-up also opens up on the board for visual stimulation.

# The Mechanism

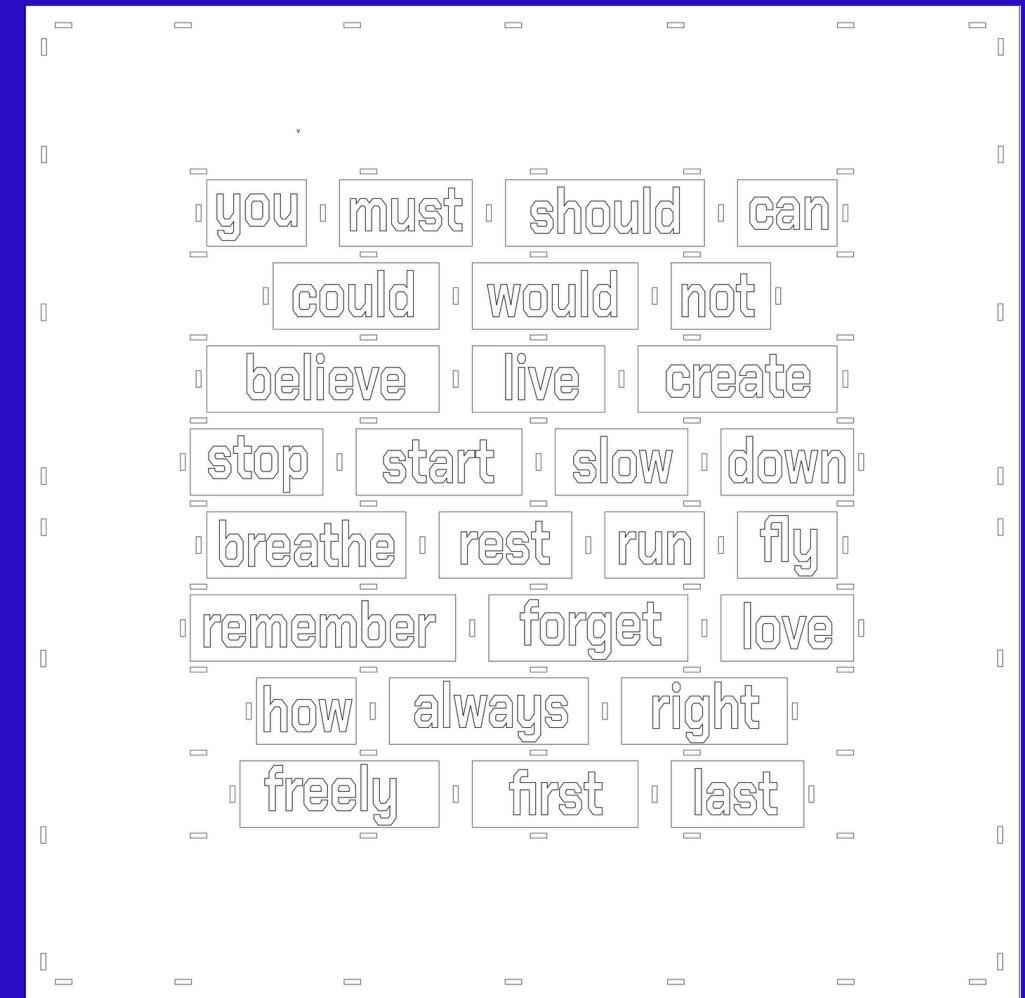


The words are laser cut on mdf and are stuck to the back of a paper screen on the main board that is backlit using an LED strip. This LED strip is programmed to light up the LEDs corresponding to the word when it is randomly assigned to do so. The pop-ups are controlled using a servo motor attached to the side of the main board

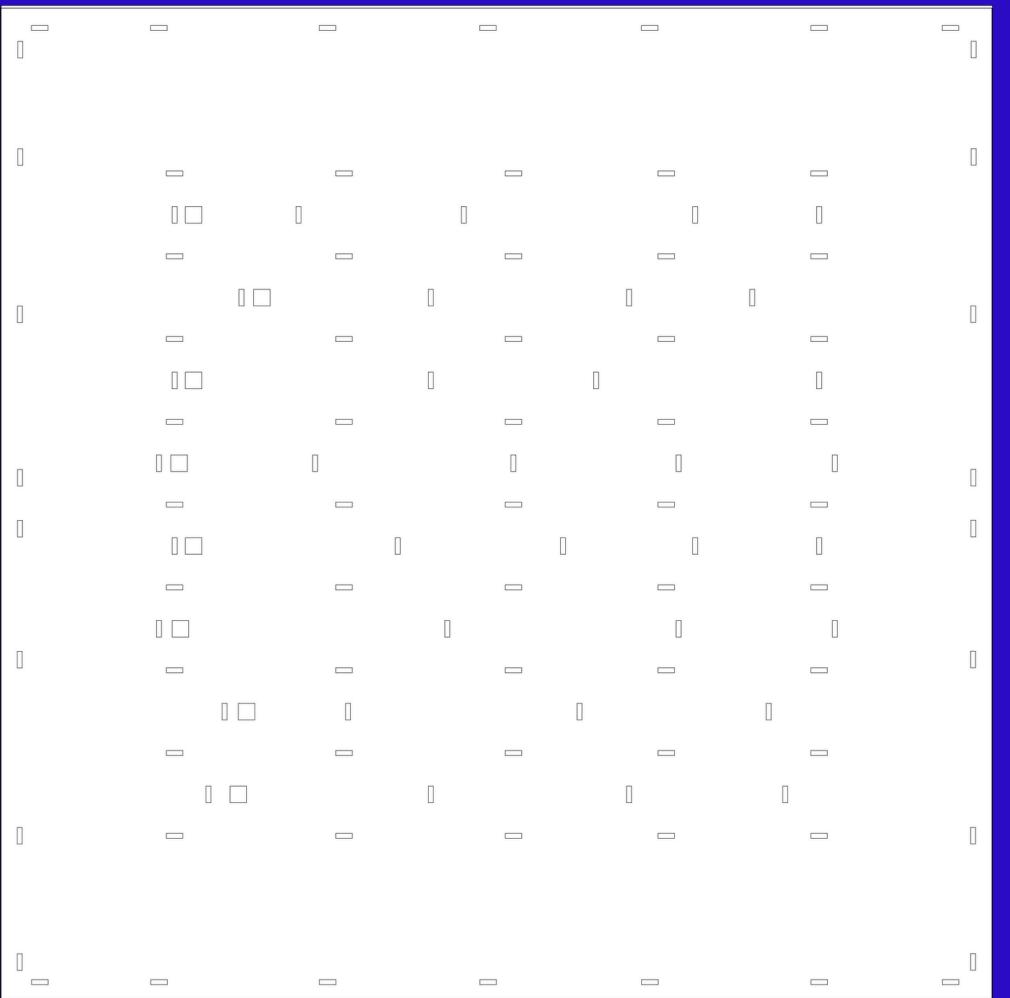
# The Physical Model



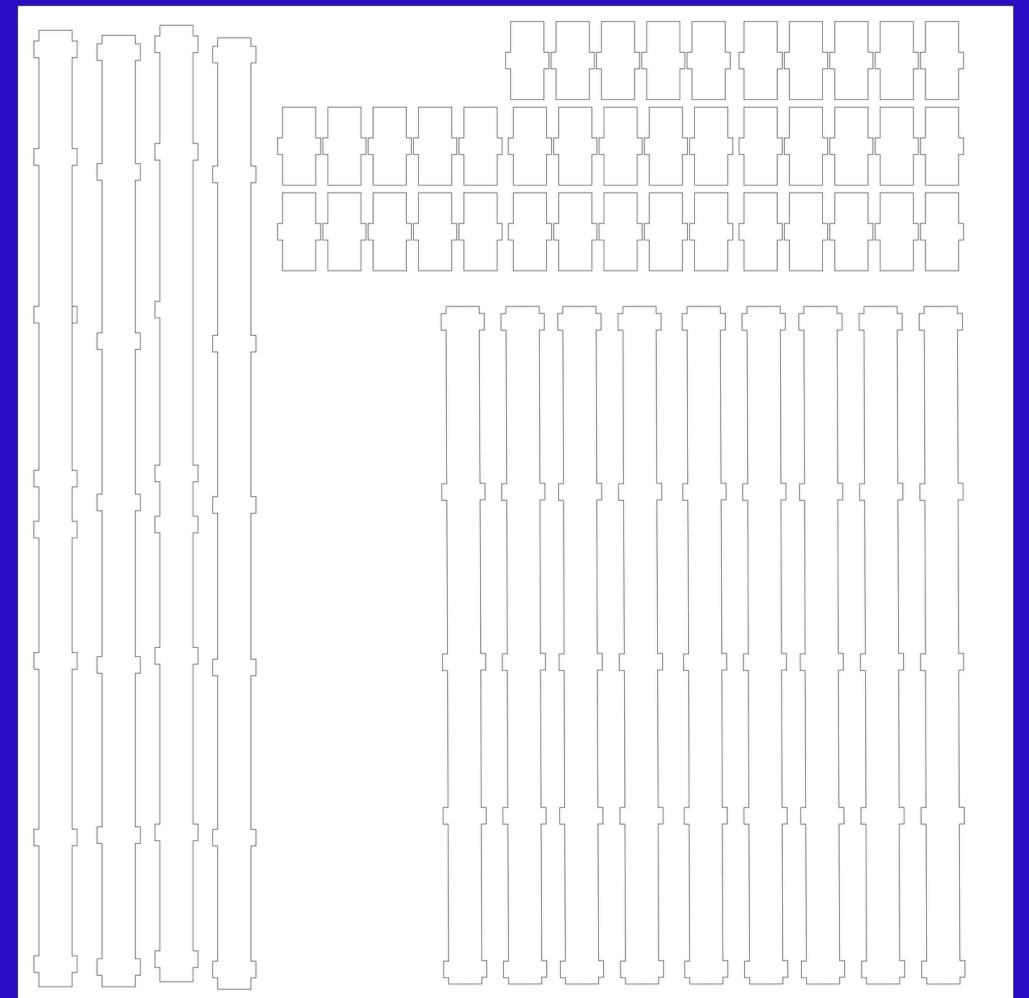
# The Lasercut Board Models



the front face with the letter  
cutouts



the back face with openings  
for the led strip wiring



the adjoining parts in the middle  
along with dividers for each  
word to prevent light spillage

# The Pop-Up Flaps



# The Words-Sentence Formation

**SUBJECT:** YOU

**MODAL VERBS:** MUST,  
SHOULD, CAN, COULD,  
WOULD

**NEGATIVE:** NOT

**VERBS:** BELIEVE, LIVE, CREATE,  
STOP, START, SLOW DOWN,  
BREATHE, REST, RUN, FLY,  
REMEMBER, FORGET, LOVE

**OBJECTS:** NOW, ALWAYS,  
RIGHT, FREELY, FIRST, LAST

**A possible sentence could be:** YOU SHOULD CREATE FREELY

# Individual Contribution

## Arya

- ideation of concept and mechanism
- flow of the code
- writing the actual code
- making most of the pop-ups
- making the mechanism of the pop-ups
- soldering and configuring the LED Strips
- making breadboard connections
- testing out individual connections and troubleshooting
- Creating the github repository
- Anoushka's Emotional Support

## Anoushka

- ideation of concept and mechanism
- sentence formation algorithm
- tweaks to the code-making lists
- making 2 pop-ups
- making the illustrator file for the physical model
- making the physical model
- soldering and configuring LED Strips
- making breadboard connections; troubleshooting
- Documentation through video
- Arya's Emotional Support

# Challenges We Faced

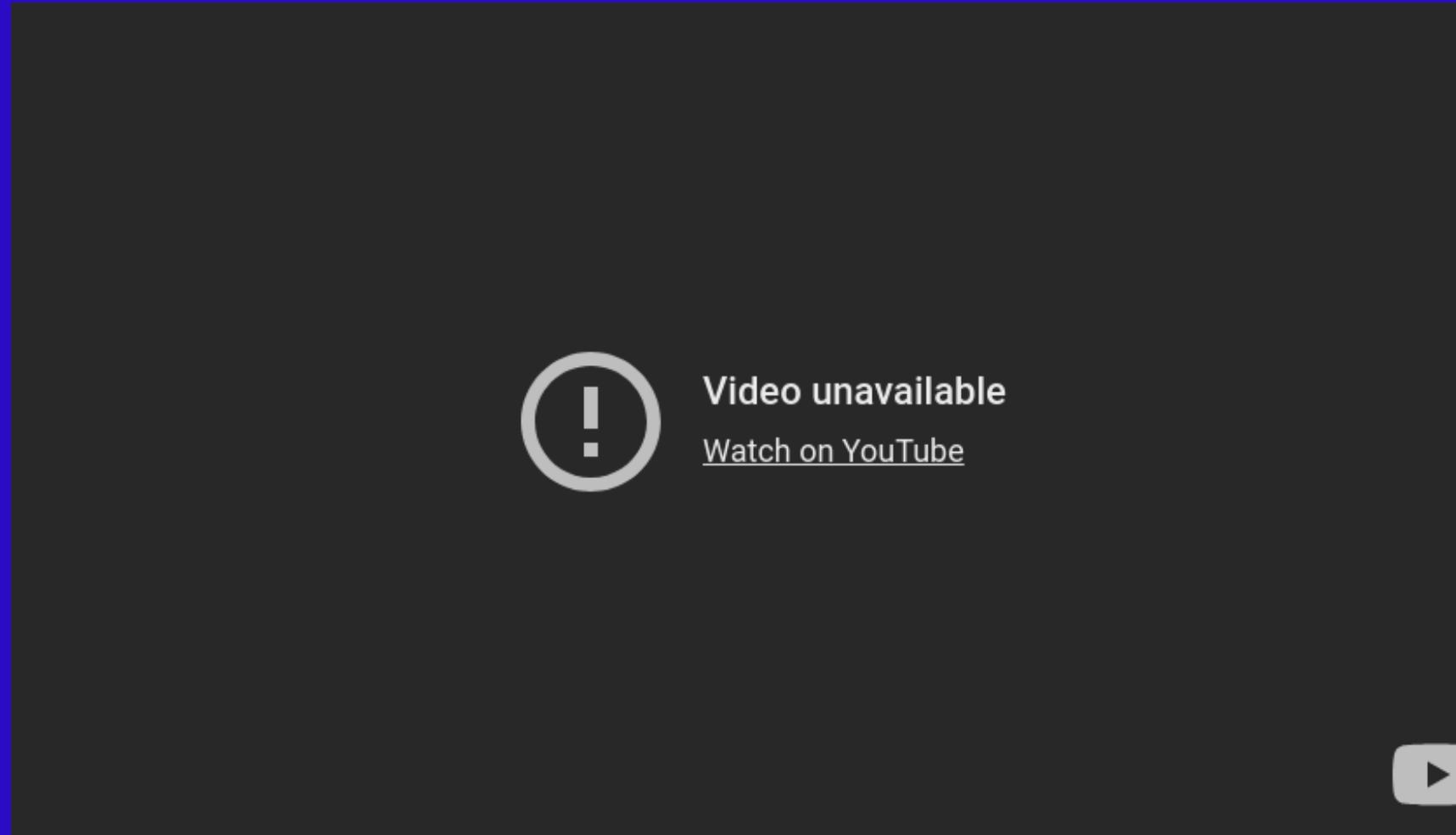
- Getting the model lasercut properly- Adobe Illustrator to CAD issues
- Retrieving our components that stopped working in our ODT Box- the ESP 32 and LED strips.
- re-configuring our code after getting the correct type of LED Strips
- Soldering the LED strips was a task since we had to de-solder it multiple times after soldering it wrong the first time
- The circuit requiring an additional external power source due to the bulk of our components- leading to a couple of melted wires.
- The size of the installation making it challenging to work with.
- Figuring out the mechanism for the flaps with servo motors
- The power module started overheating and smoking twice

# Challenges We Faced



However, at the end of every challenge is a light that shines due to our perseverance.

# Links



## [area5205/ \*\*Synapse-ODT-\*\*...](#)



From Idea to Impact- Open Design & Technology  
Final Showcase by Arya and Anoushka

1

Contributor

0

Issues

0

Stars

0

Forks



## [\*\*area5205/Synapse-ODT-2025: From Idea to Impact- Open Design & Technology Final Showcase by Arya and...\*\*](#)

From Idea to Impact- Open Design & Technology Final Showcase by Arya and Anoushka - area5205/Synapse-ODT-2025

 GitHub



Thankyou!