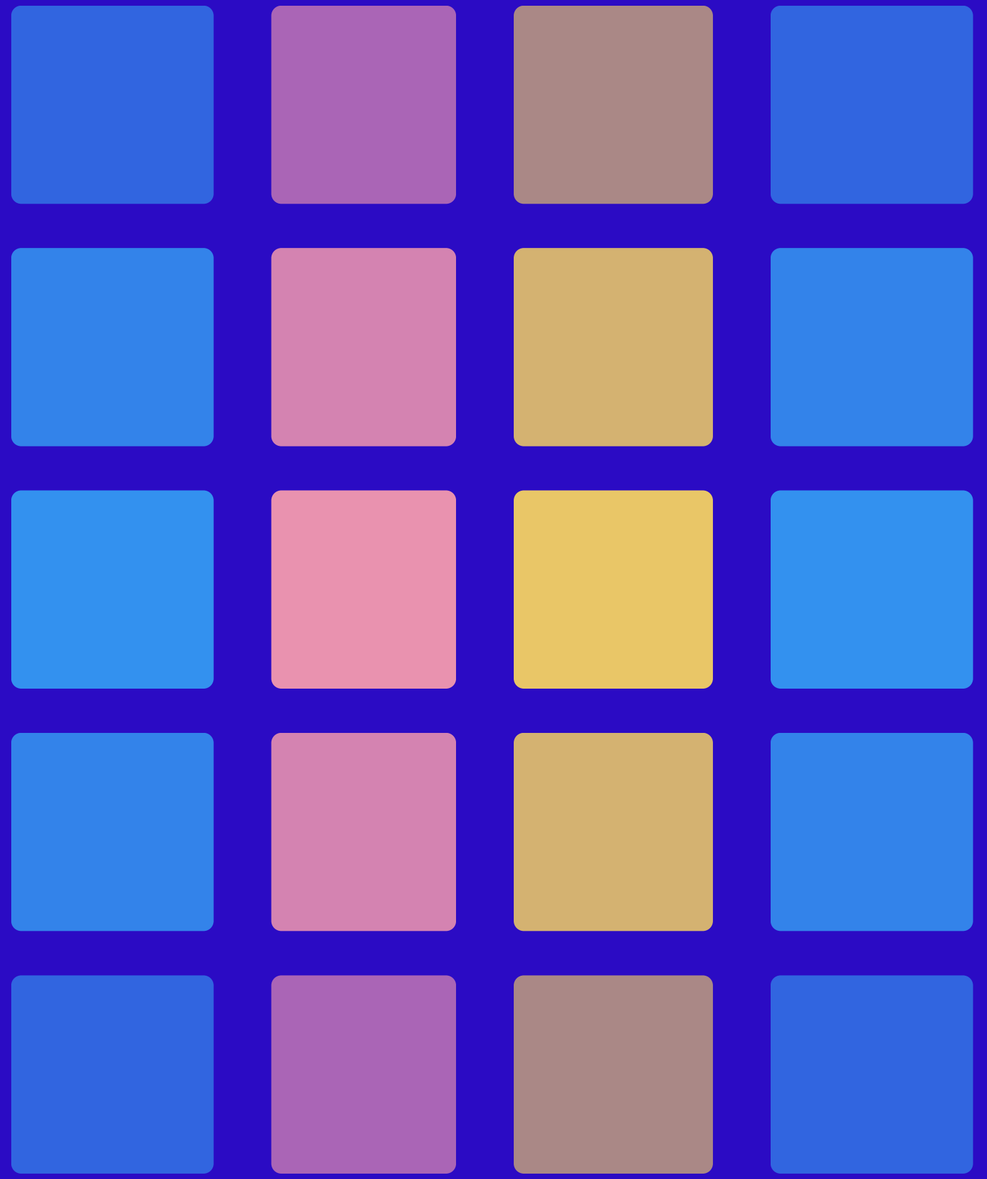
ANOUSHKA PEDNEKAR

ARYA GANGWAR

Synapse

an interactive installation



OPEN DESIGN TECHNOLOGY SEM2

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.

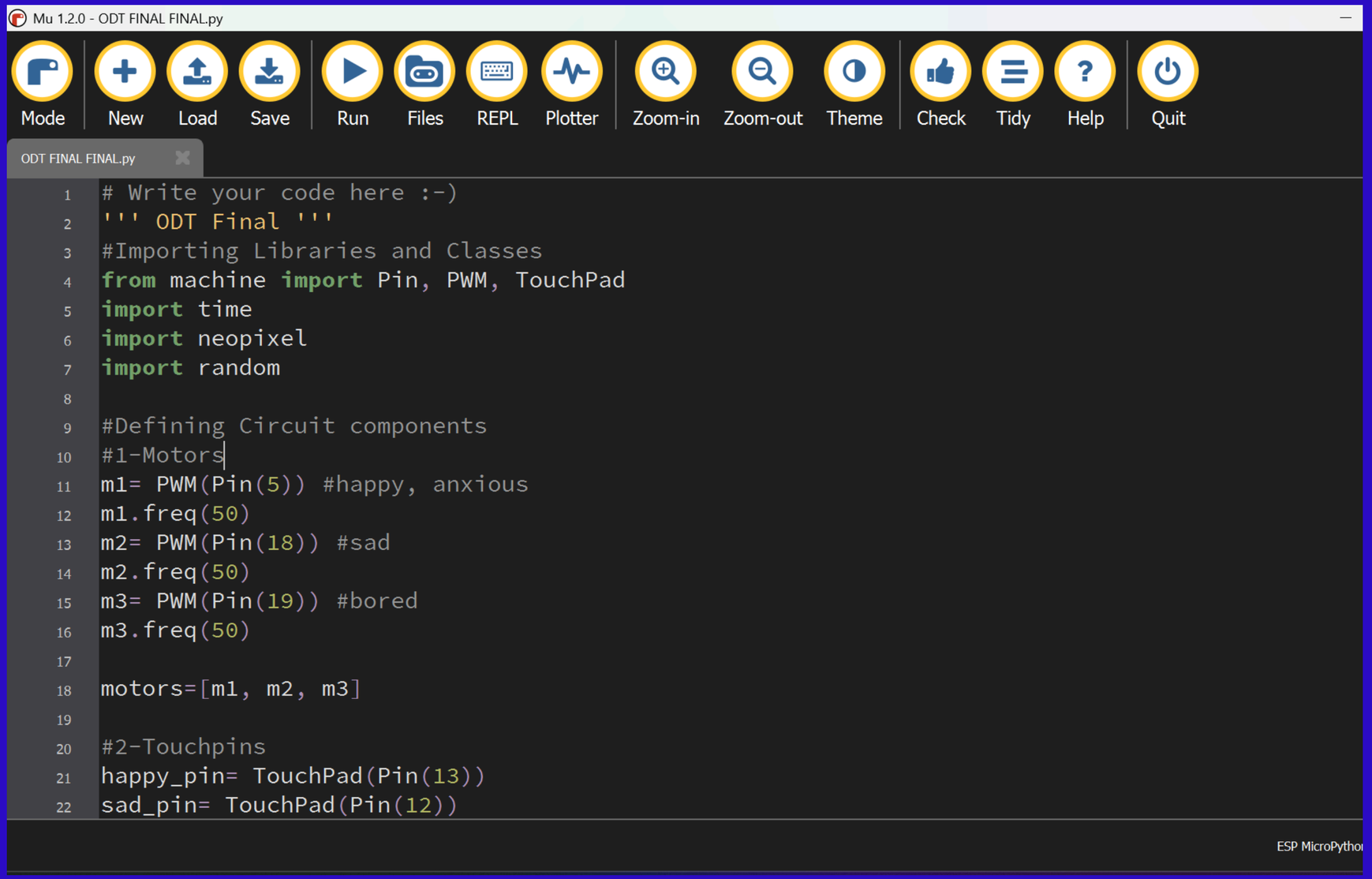
Our installation follows the principles of English sentence formation to give the user an advisory or observational response to their emotion along with the visual element of a pop-out card which can then be interpreted by the user in various ways.

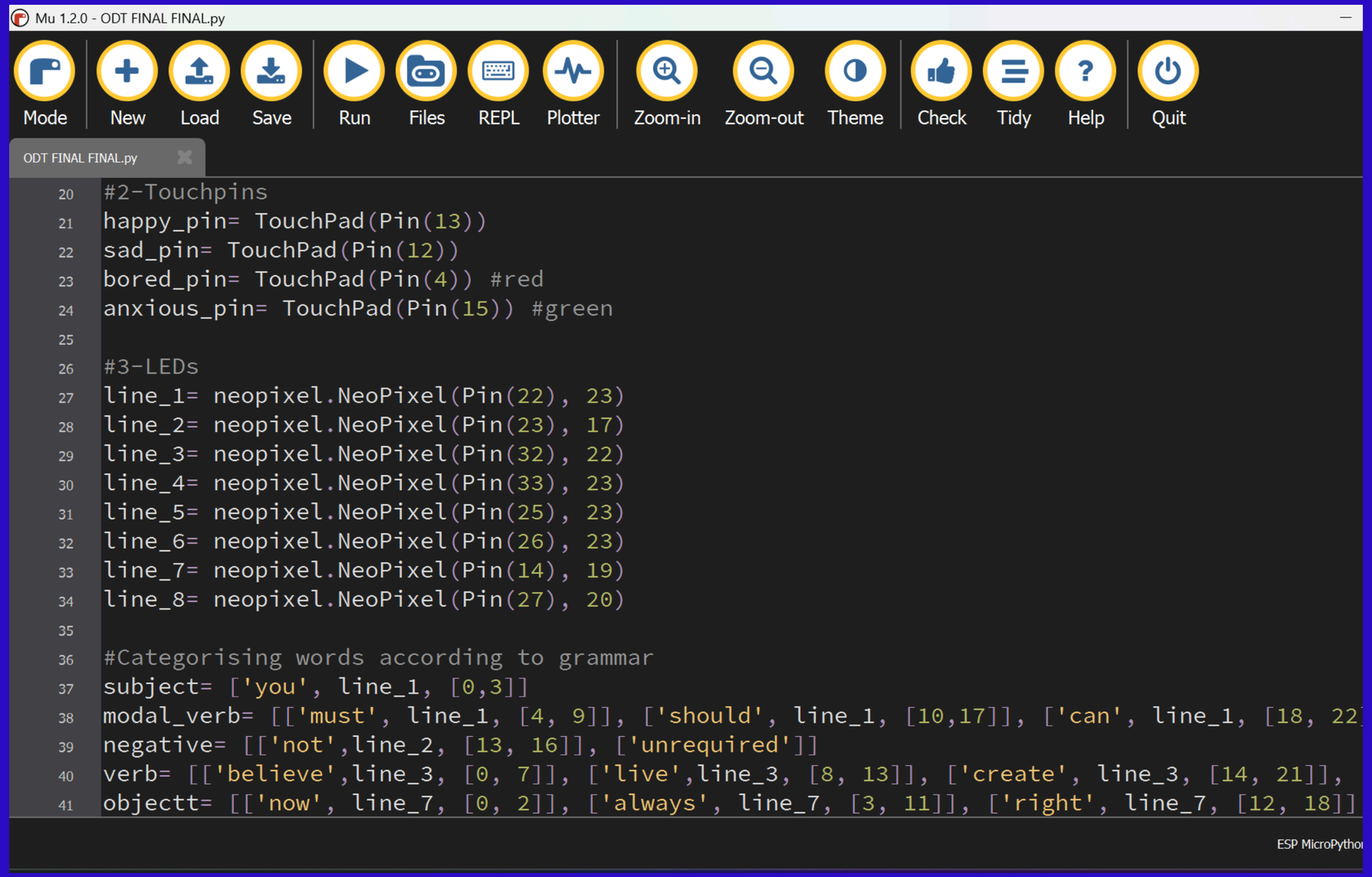
How to Interact with Synapse

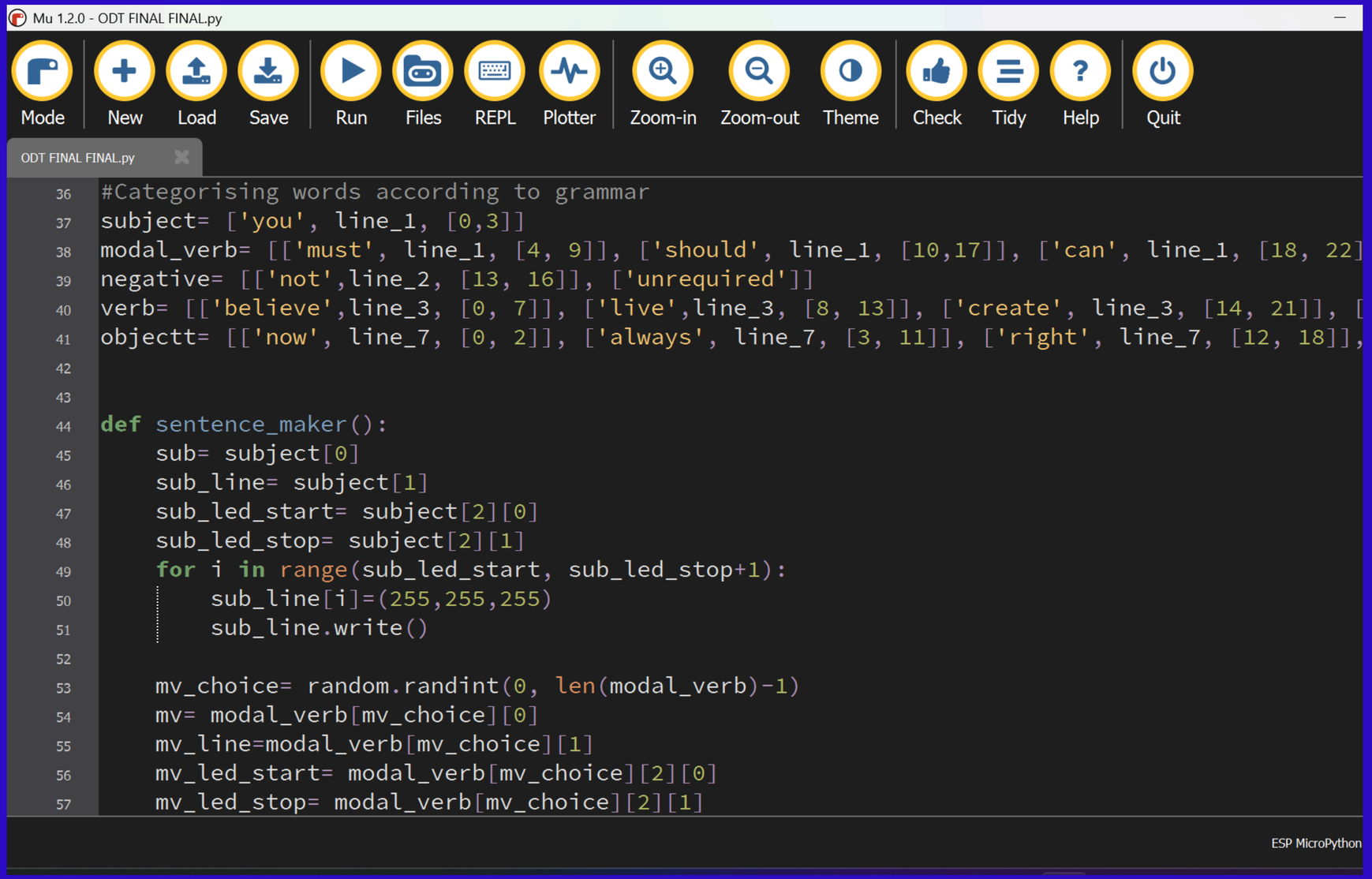
1. The user touches the Pin that corresponds to their emotion- happy, sad, anxious or bored.

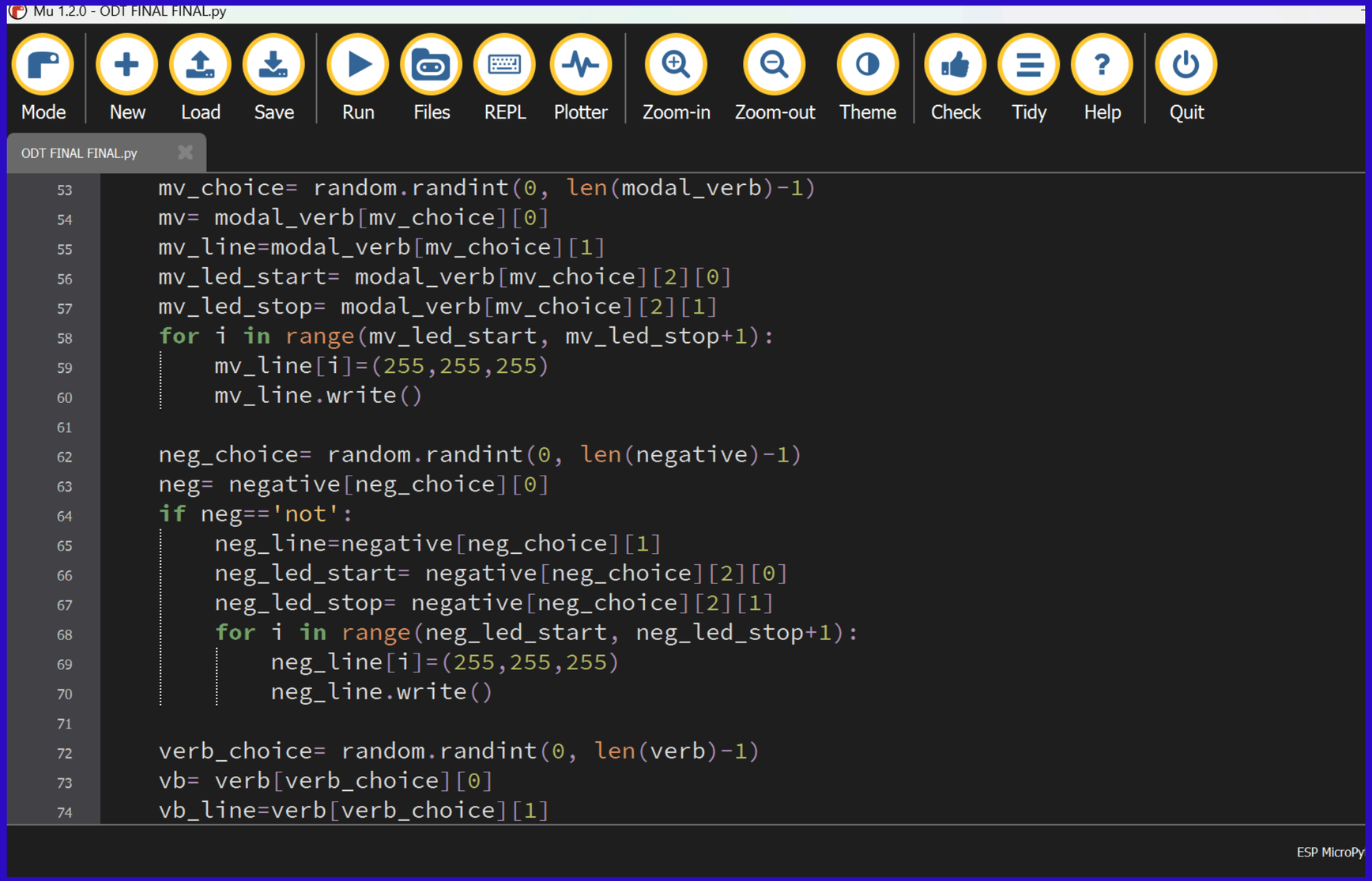
2. The installation then lights up a series of words on the board that form a sentence that the user must read and decipher for themselves. They can treat this sentence as a forecast of their future or an observation of their present.

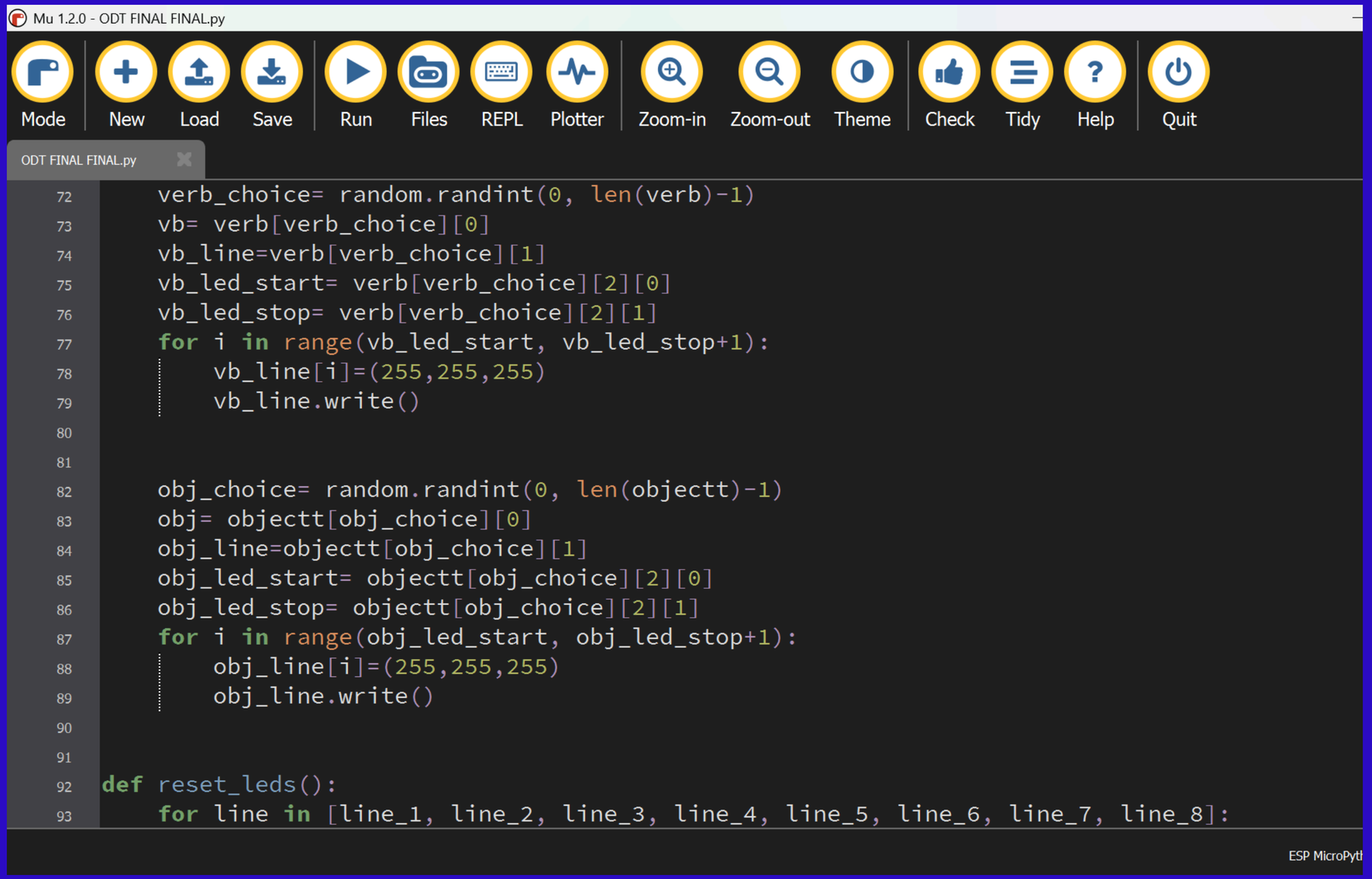
The Code


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
Write your code here :-)
''' ODT Final '''
#Importing Libraries and Classes
from machine import Pin, PWM, TouchPad
import time
import neopixel
import random
#Defining Circuit components
#1-Motors
m1= PWM(Pin(5)) #happy, anxious
m1.freq(50)
m2= PWM(Pin(18)) #sad
m2.freq(50)
m3= PWM(Pin(19)) #bored
m3.freq(50)
motors=[m1, m2, m3]
#2-Touchpins
happy_pin= TouchPad(Pin(13))
sad_pin= TouchPad(Pin(12))
ESP MicroPython

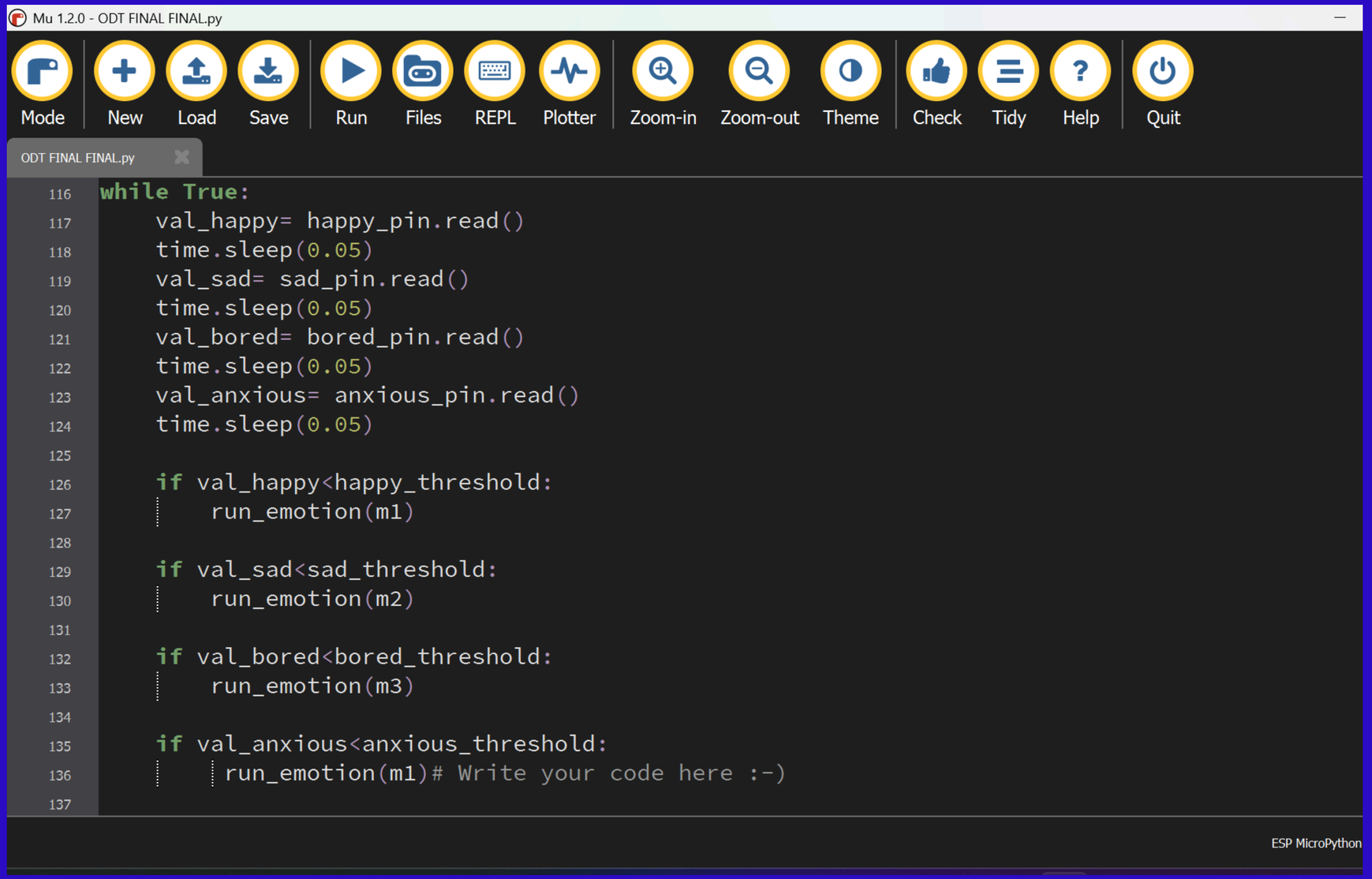

Mu 1.2.0 - ODT FINAL FINAL.py
Mode
New
Load
Save
Run
Files
REPL
Plotter
Zoom-in
Zoom-out
Theme
Check
Tidy
Help
Quit
ODT FINAL FINAL.py
#2-Touchpins
happy_pin= TouchPad(Pin(13))
sad_pin= TouchPad(Pin(12))
bored_pin= TouchPad(Pin(4)) #red
anxious_pin= TouchPad(Pin(15)) #green
#3-LEDs
line_1= neopixel.NeoPixel(Pin(22), 23)
line_2= neopixel.NeoPixel(Pin(23), 17)
line_3= neopixel.NeoPixel(Pin(32), 22)
line_4= neopixel.NeoPixel(Pin(33), 23)
line_5= neopixel.NeoPixel(Pin(25), 23)
line_6= neopixel.NeoPixel(Pin(26), 23)
line_7= neopixel.NeoPixel(Pin(14), 19)
line_8= neopixel.NeoPixel(Pin(27), 20)
#Categorising words according to grammar
subject= ['you', line_1, [0,3]]
modal_verb= [['must', line_1, [4, 9]], ['should', line_1, [10,17]], ['can', line_1, [18, 22
negative= [['not',line_2, [13, 16]], ['unrequired']]
verb= [['believe',line_3, [0, 7]], ['live',line_3, [8, 13]], ['create', line_3, [14, 21]],
objectt= [['now', line_7, [0, 2]], ['always', line_7, [3, 11]], ['right', line_7, [12, 18]]
ESP MicroPytho


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
#Categorising words according to grammar
subject= ['you', line_1, [0,3]]
modal_verb= [['must', line_1, [4, 9]], ['should', line_1, [10,17]], ['can', line_1, [18, 22]
negative= [['not',line_2, [13, 16]], ['unrequired']]
verb= [['believe',line_3, [0, 7]], ['live',line_3, [8, 13]], ['create', line_3, [14, 21]], [
objectt= [['now', line_7, [0, 2]], ['always', line_7, [3, 11]], ['right', line_7, [12, 18]],
def sentence_maker():
sub= subject[0]
sub_line= subject[1]
sub_led_start= subject[2][0]
sub_led_stop= subject[2][1]
for i in range(sub_led_start, sub_led_stop+1):
sub_line[i]=(255,255,255)
sub_line.write()
mv_choice= random.randint(0, len(modal_verb)-1)
mv= modal_verb[mv_choice][0]
mv_line=modal_verb[mv_choice][1]
mv_led_start= modal_verb[mv_choice][2][0]
mv_led_stop= modal_verb[mv_choice][2][1]
ESP MicroPython

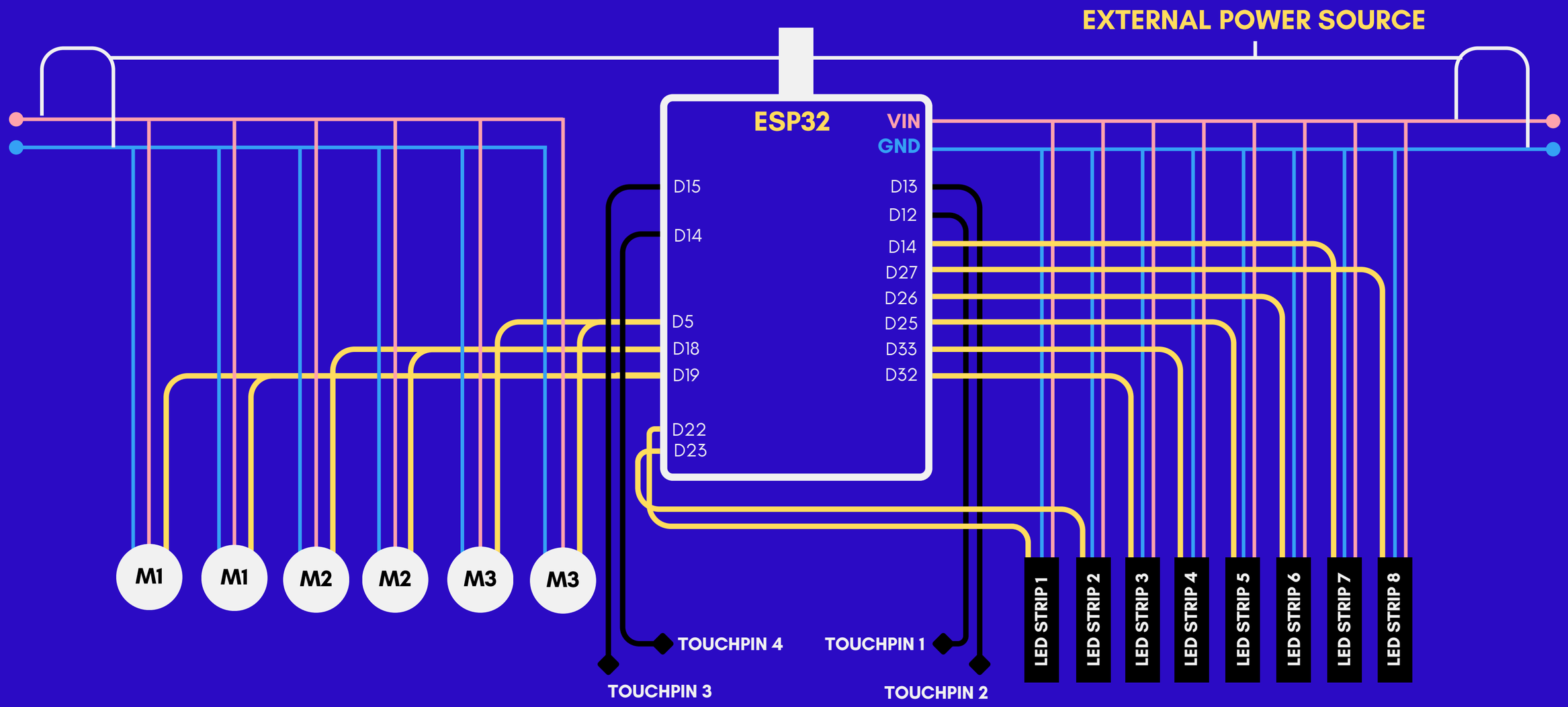

Mu 1.2.0 - ODT FINAL FINAL.py
Mode
New
Load
Save
Run
Files
REPL
Plotter
Zoom-in
Zoom-out
Theme
Check
Tidy
Help
Quit
ODT FINAL FINAL.py
53 mv_choice= random.randint(0, len(modal_verb)-1)
54 mv= modal_verb[mv_choice][0]
55 mv_line=modal_verb[mv_choice][1]
56 mv_led_start= modal_verb[mv_choice][2][0]
57 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]
ESP MicroPy


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
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]:
ESP MicroPyth


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
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
ESP MicroPy


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
while True:
val_happy= happy_pin.read()
time.sleep(0.05)
val_sad= sad_pin.read()
time.sleep(0.05)
val_bored= bored_pin.read()
time.sleep(0.05)
val_anxious= anxious_pin.read()
time.sleep(0.05)
if val_happy<happy_threshold:
run_emotion(m1)
if val_sad<sad_threshold:
run_emotion(m2)
if val_bored<bored_threshold:
run_emotion(m3)
if val_anxious<anxious_threshold:
run_emotion(m1)# Write your code here :-)
ESP MicroPython

The Circuit Diagram



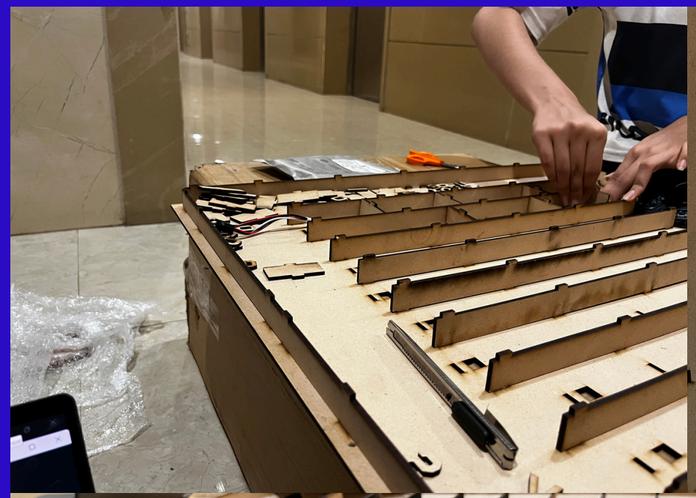
The Theory

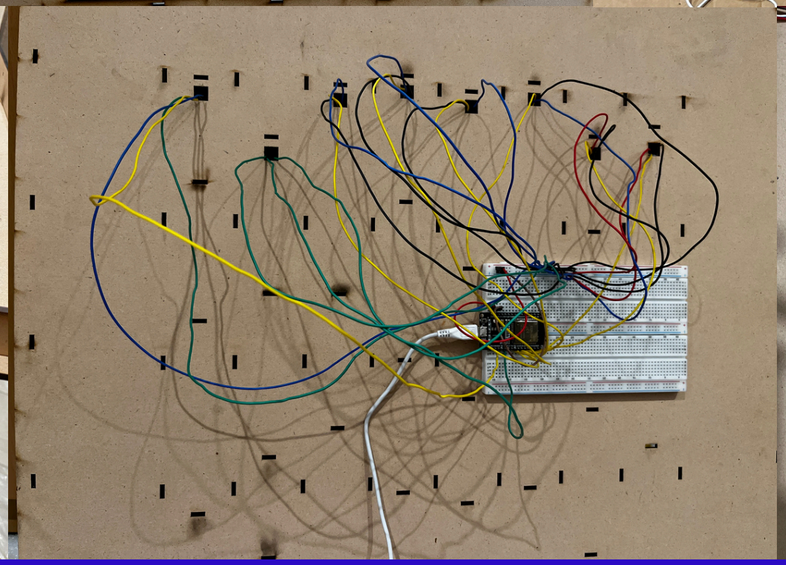
We have divided the english sentence into five parts: **the singular subject, a modal verb, an optional negative word, a verb, and an object**. Since the subject is only one, conjugation isn't required. Hence, different permutation combinations of the modal verb, verb, and object create various sentences.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

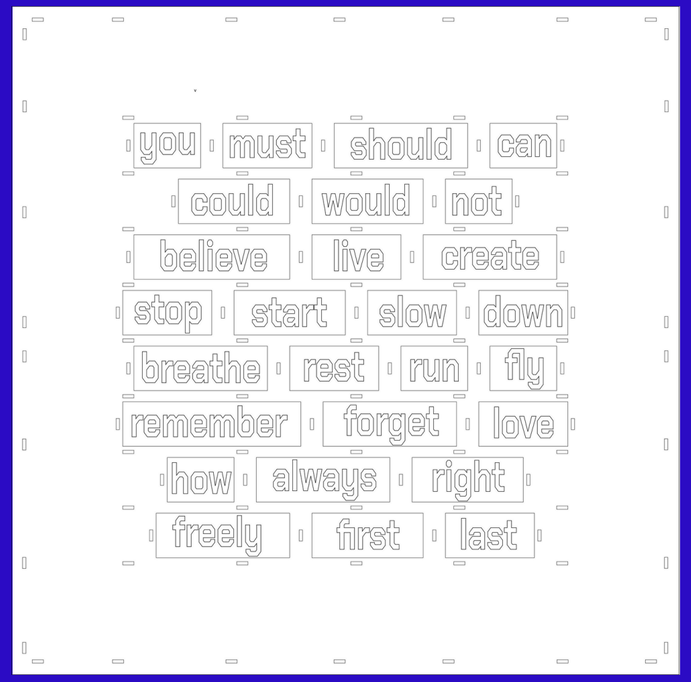
The words are laser cut on mdf and are stuck to the back of a paper screen on the main board that is backlit using an LED strip. This LED strip is programmed to light up the LEDs corresponding to the word when it is randomly assigned to do so. The pop-ups are controlled using a servo motor attached to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

The Physical Model

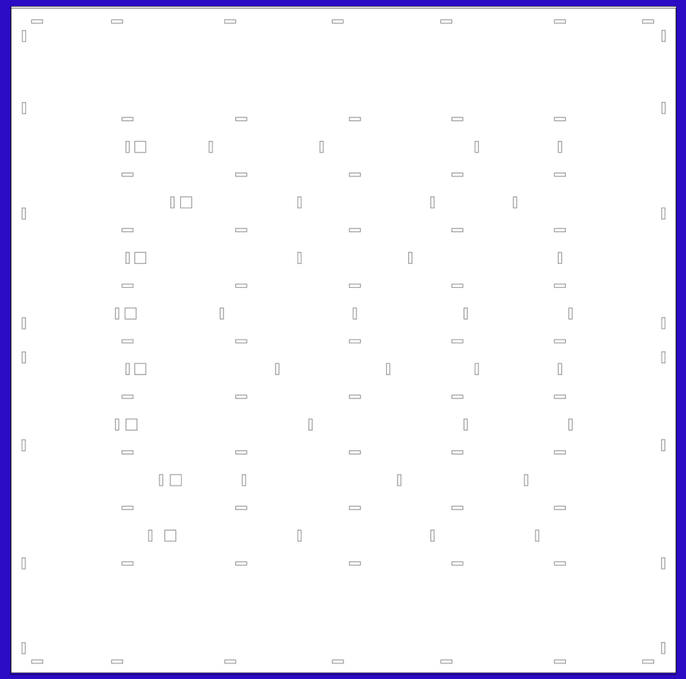




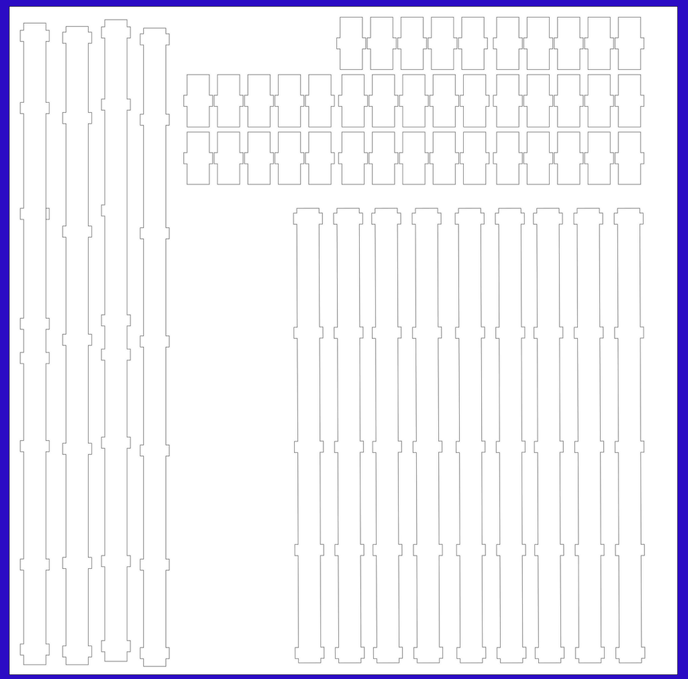
The Lasercut Board Models



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
- Documention through video
- Arya's Emotional Support

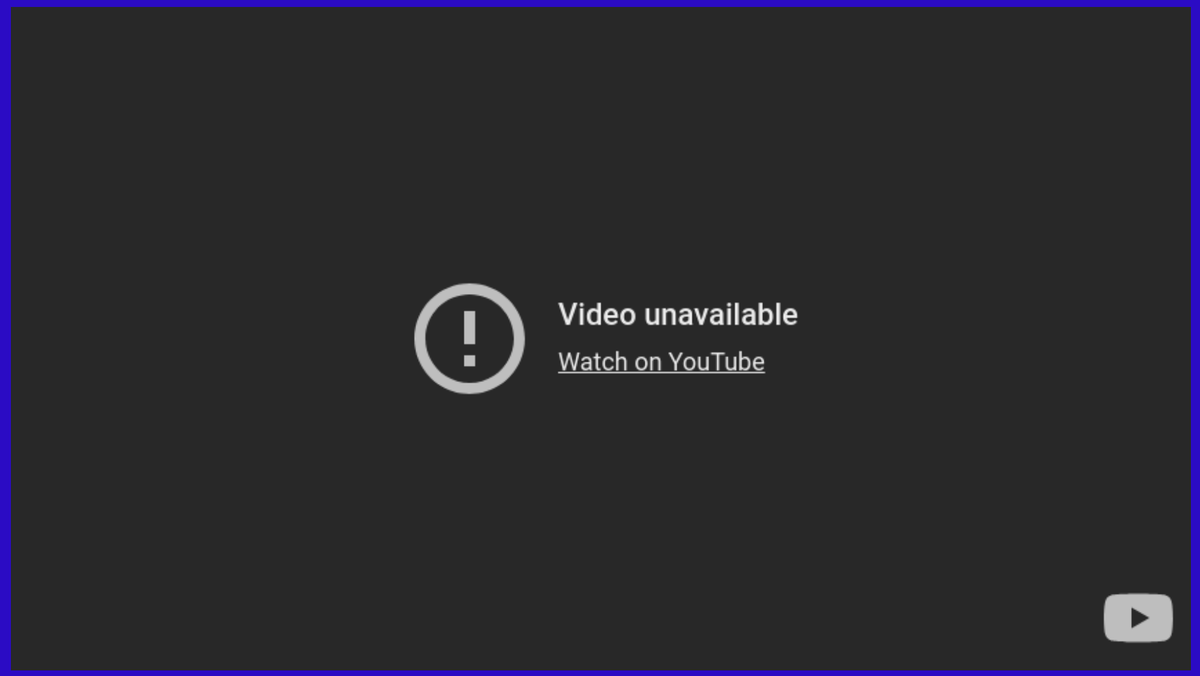
Challenges We Faced

- 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 reqiuiring 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

SYNAPSE

OPEN DESIGN TECHNOLOGY SEM2

Thankyou!

ANOUSHKA PEDNEKAR

ARYA GANGWAR