HOWARD UNIVERSITY

X

NOVAORBIT CORP

PRECISION IN MOTION

Autonomous Environmental Monitoring Drones (AEMDs) for Lunar Exploration

Masia Wisdom & Richard Coaxum
Advisor: Marley Belot

Background

- Lunar surface challenges: Extreme temperatures, abrasive dust, and GPS-denied navigation.
- Need for reliable environmental monitoring, communication, and power for astronaut missions.

Problem Statement

"Lunar surface technologies face significant challenges due to harsh, ever-changing conditions, necessitating reliable environmental monitoring, communication, and long-lasting power for autonomous drones assisting astronauts."

DESIGN CONSTRAINTS

AEMD Lunar Drone – Key Constraints

Extreme Temperatures: Operates from -173°C to +127°C

Radiation Exposure: Needs radiation-hardened electronics

Lunar Dust: Must resist abrasive, electrostatic regolith

Low Gravity: Affects mobility, stability, and sensor tuning

Power Limits: minimal battery use

Comms Delay: 1.3s latency; limited bandwidth

Edge-only processing

Mass & Volume Limits:

SOLUTION DESIGN

Key Components:

- **Transmitter:**

ESP32-S3 with onboard sensors:

- **BMP388** – Barometric pressure and altitude
- **INA219** – Voltage, current, and power measurement
- Communicates wirelessly via **ESP-NOW**

- **Receiver:**

Seeed Studio XIAO ESP32-S3

- **Receives data via ESP-NOW**
- **Forwards data to Raspberry Pi via UART**

- **Graphical User Interface (GUI):**

Built with Python Tkinter

- **Displays real-time plots, system alerts, and telemetry data**
- **Supports data export and monitoring feedback**

SPRINT 1: HARDWARE PROTOTYPING

Challenges:

- Power constraints
- Limited processing on ESP32-S3
- Wiring complexity

Solutions:

✓ Adaptive power management

✓ Improved hardware persistence and layout optimization

SPRINT 2: SOFTWARE INTEGRATION

Challenges: SSH Verification, Board Programming, GUI Glitches

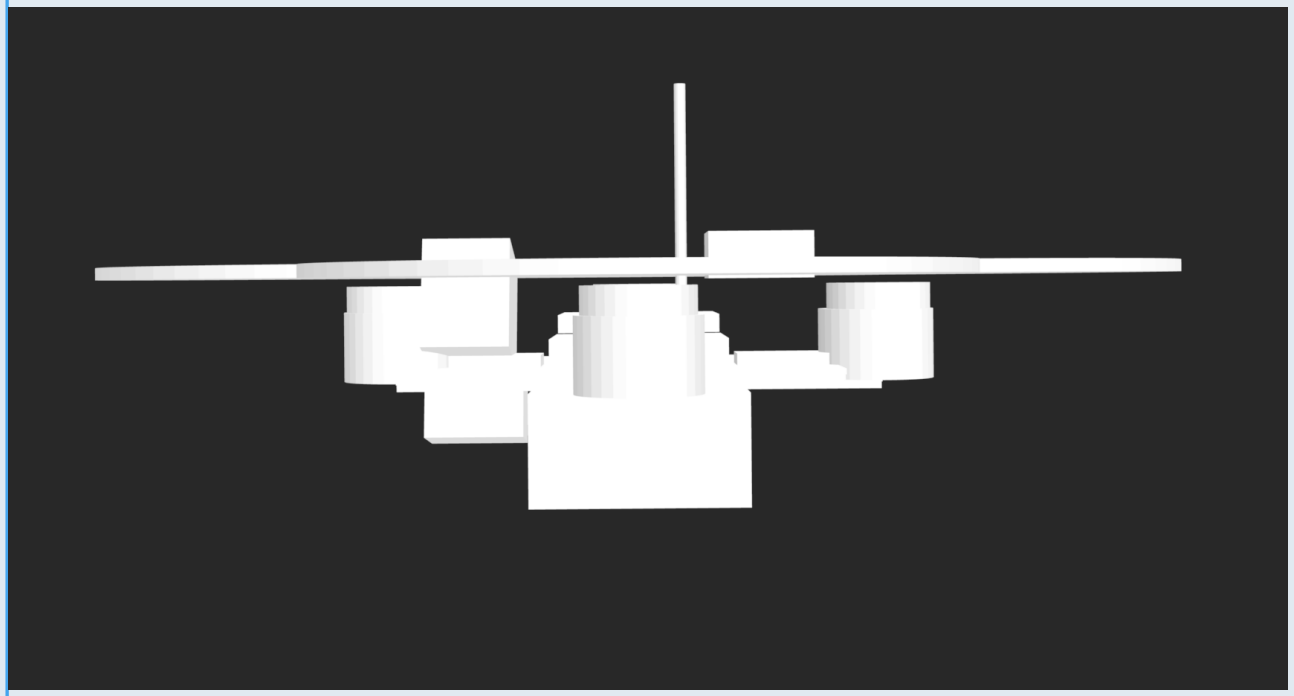
- Troubleshooting, Coded GUI and wrote Arduino Sketches in C

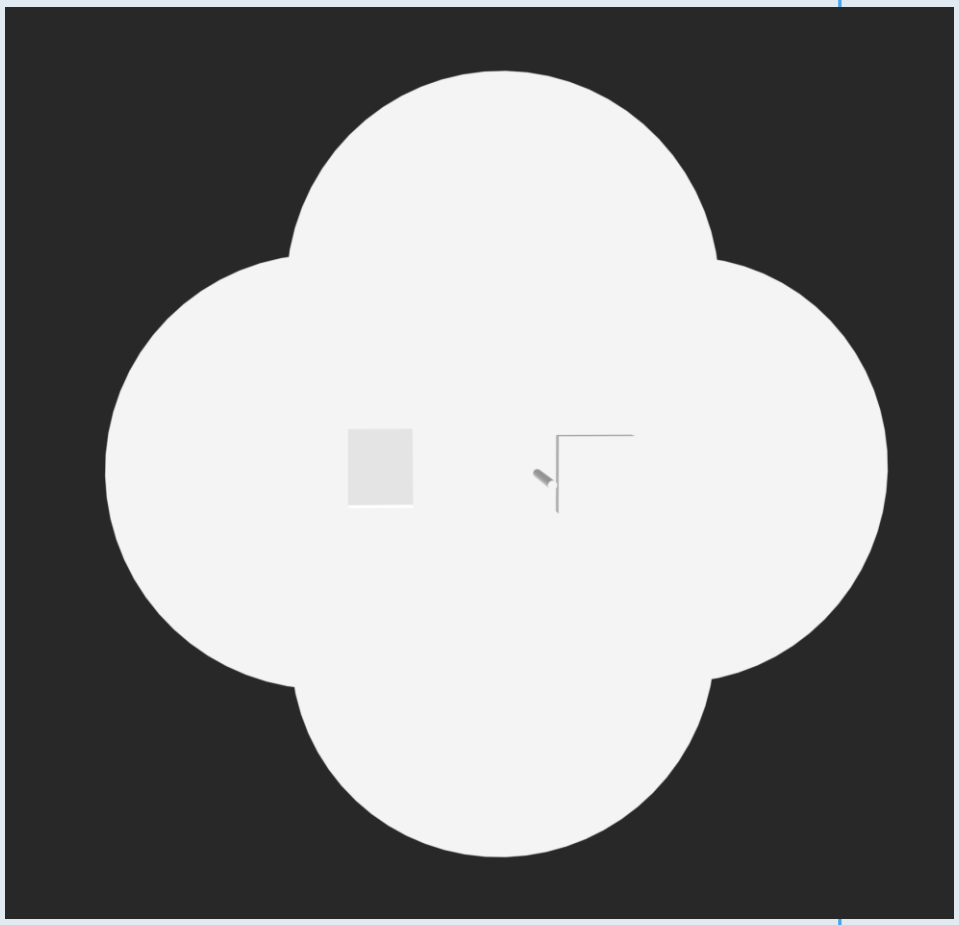
SPRINT 3: PIPELINE

Challenges: Failing Sensors, Data transfer

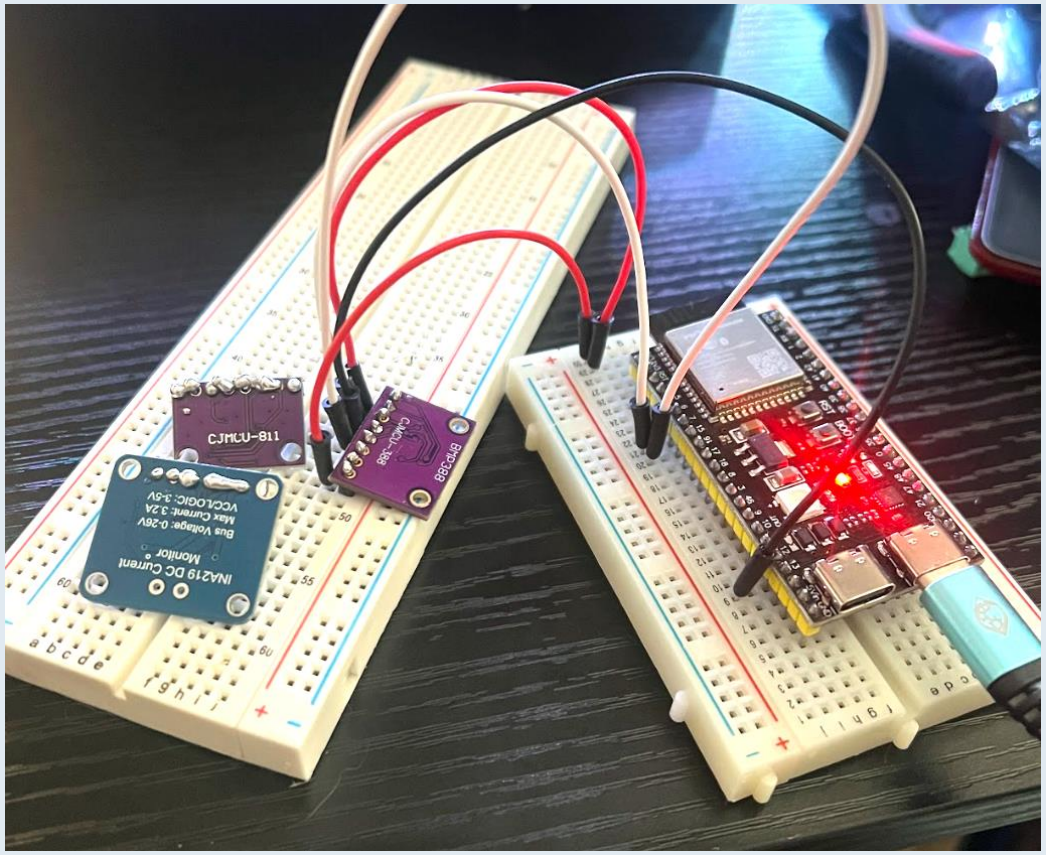
- Refactoring

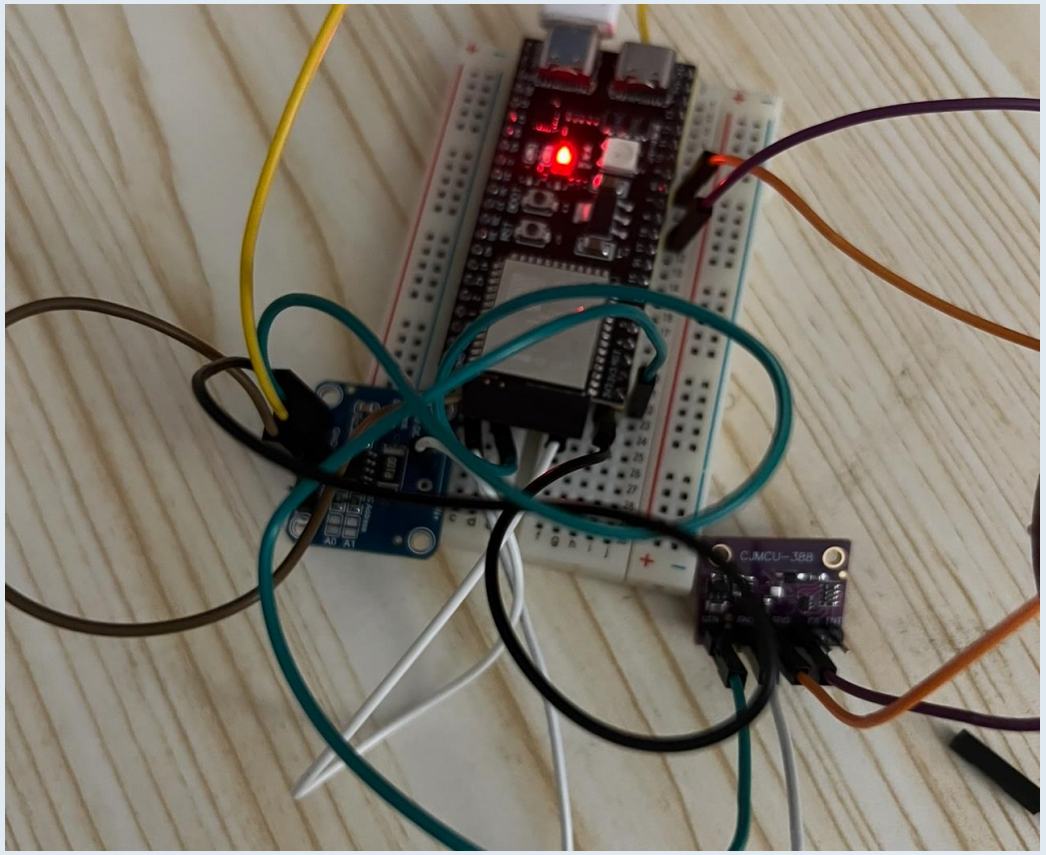
Top-down/side views of drone render



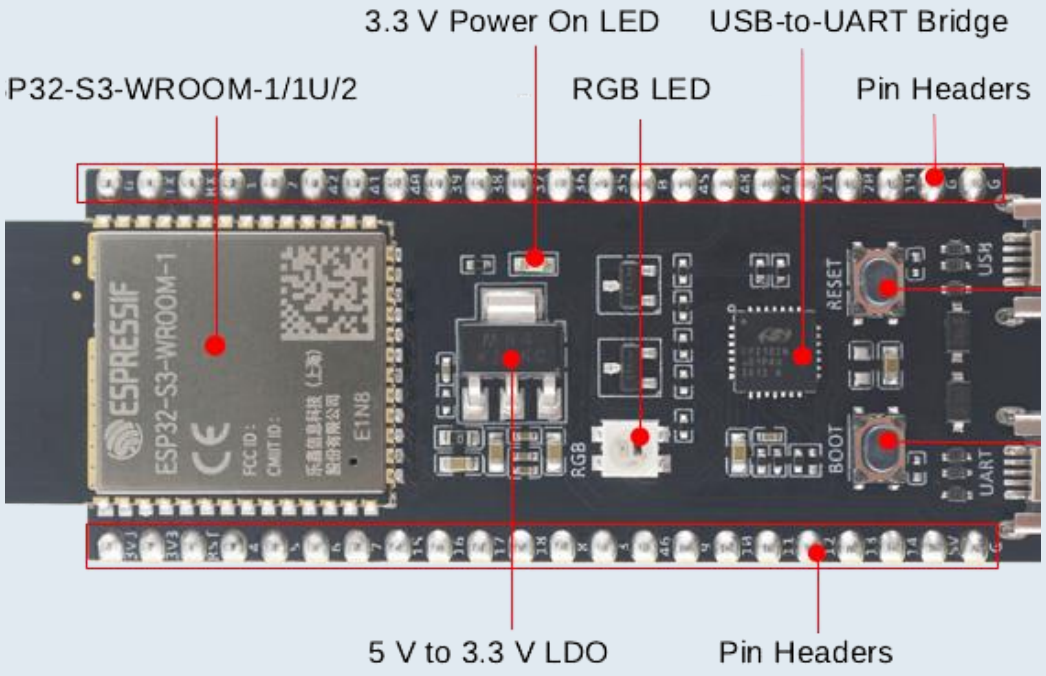


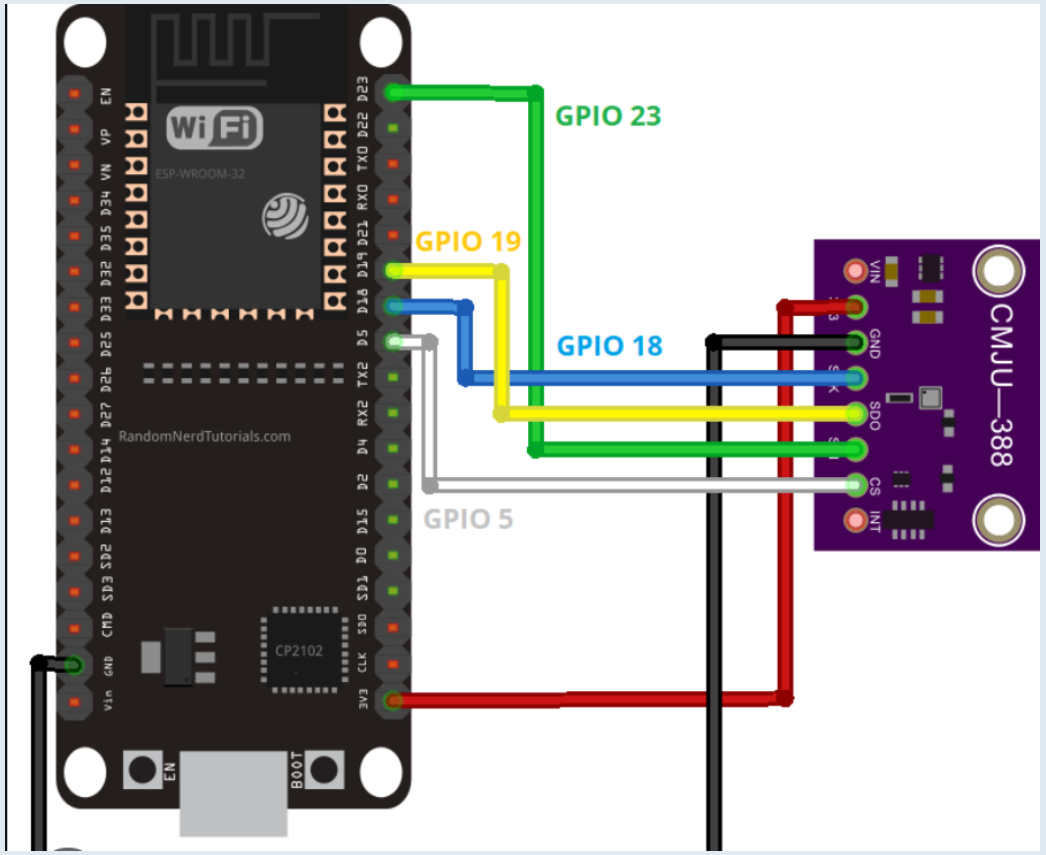
SOLDERED COMPONENTS & CIRCUITS



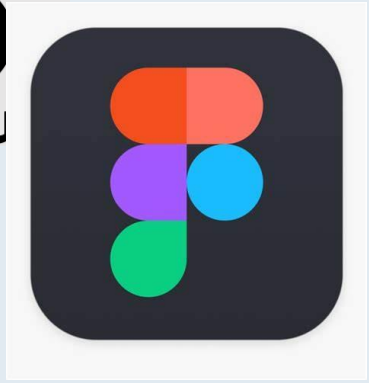


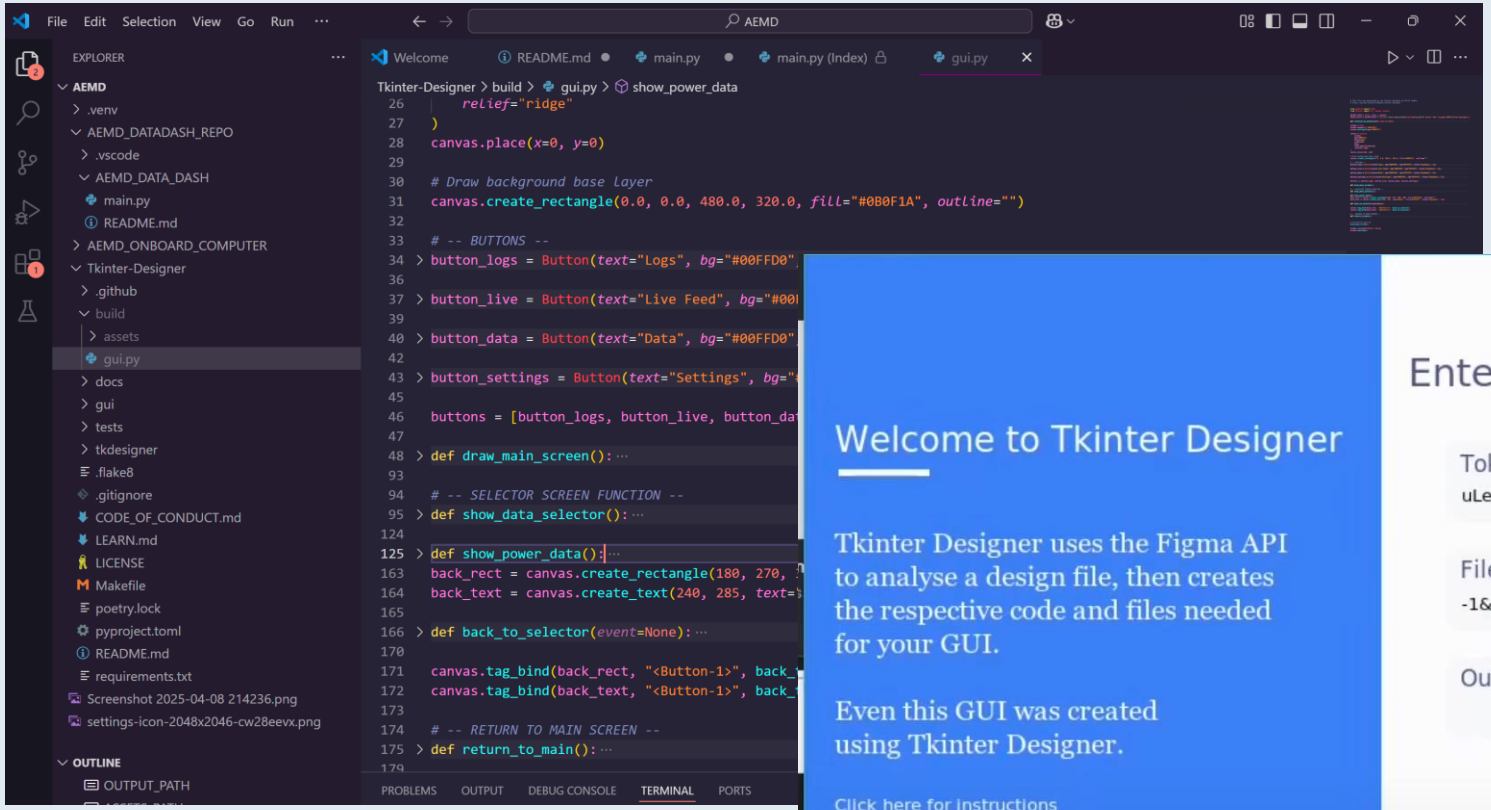
COMPONENT SCHEMATIC

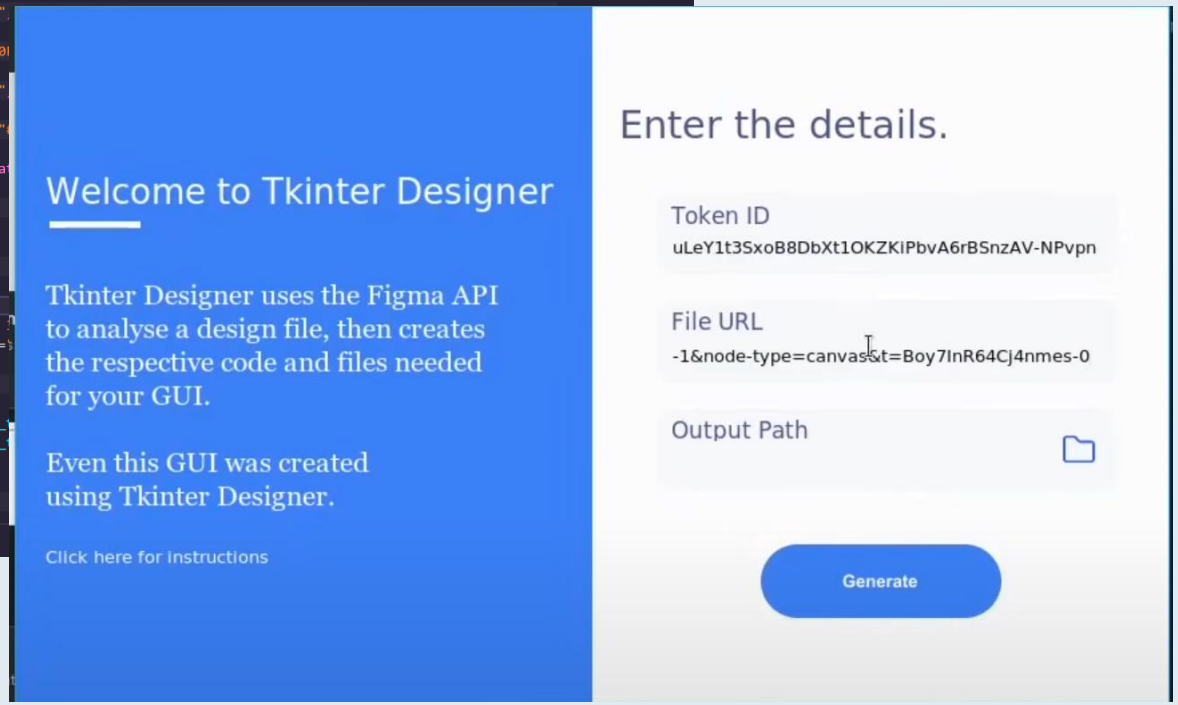




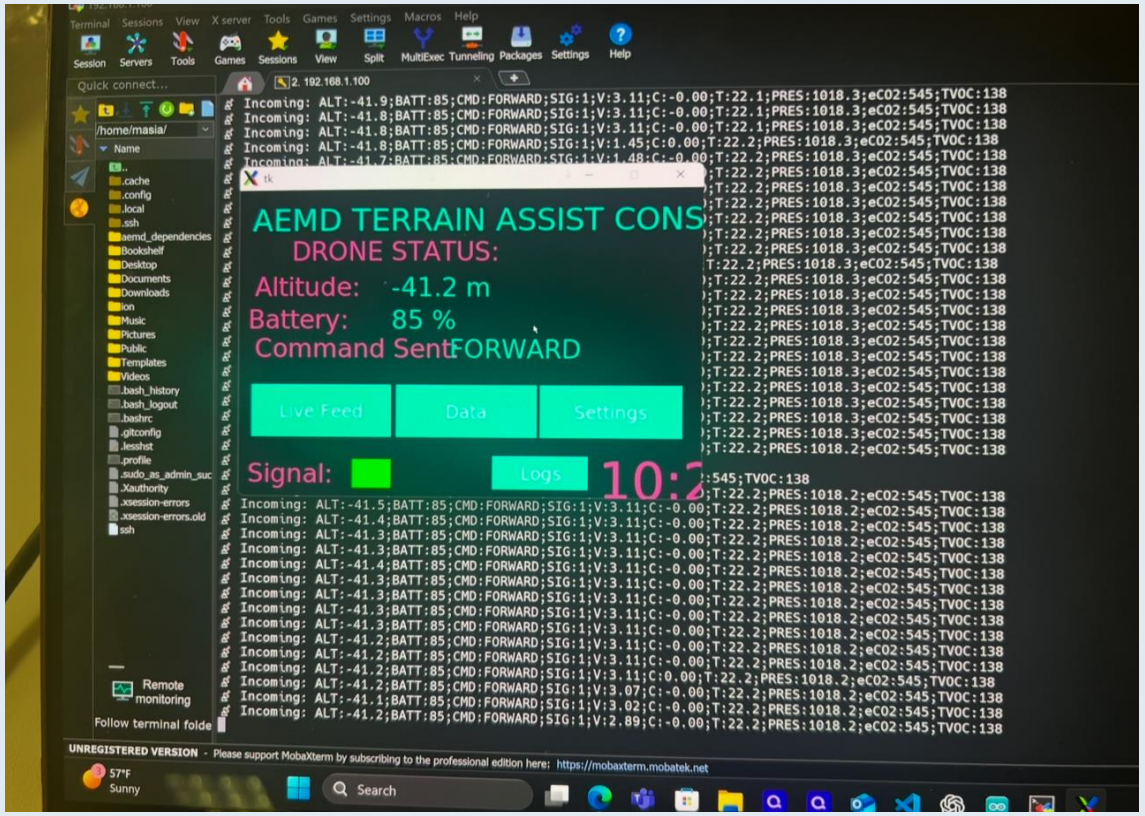
GUI Code

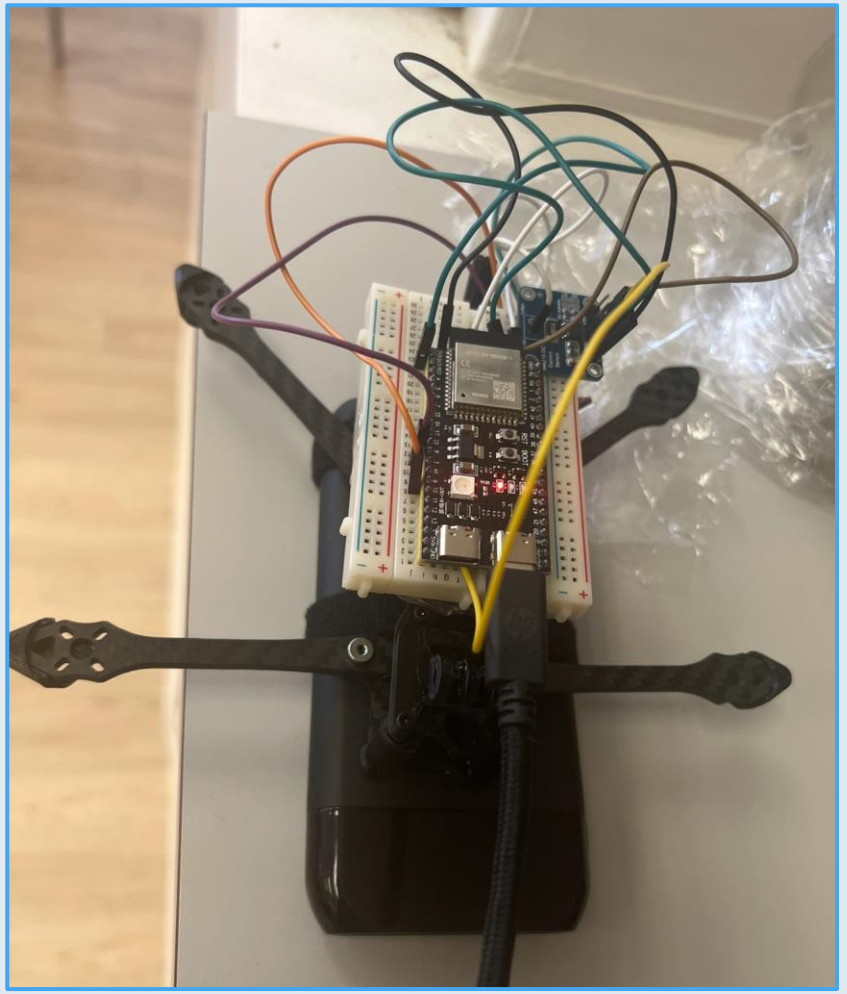



File Edit Selection View Go Run
AEMD
EXPLORER
AEMD
.venv
AEMD_DATADASH_REPO
.vscode
AEMD_DATA_DASH
main.py
README.md
AEMD_ONBOARD_COMPUTER
Tkinter-Designer
.github
build
assets
gui.py
docs
gui
tests
tkdesigner
.flake8
.gitignore
CODE_OF_CONDUCT.md
LEARN.md
LICENSE
Makefile
poetry.lock
pyproject.toml
README.md
requirements.txt
Screenshot 2025-04-08 214236.png
settings-icon-2048x2046-cw28eevx.png
OUTLINE
OUTPUT_PATH
Welcome
README.md
main.py
main.py (Index)
gui.py
Tkinter-Designer > build > gui.py > show_power_data
relief="ridge"
)
canvas.place(x=0, y=0)
# Draw background base layer
canvas.create_rectangle(0.0, 0.0, 480.0, 320.0, fill="#0B0F1A", outline="")
# -- BUTTONS --
button_logs = Button(text="Logs", bg="#00FFD0"
button_live = Button(text="Live Feed", bg="#00
button_data = Button(text="Data", bg="#00FFD0"
button_settings = Button(text="Settings", bg="
buttons = [button_logs, button_live, button_da
def draw_main_screen():
# -- SELECTOR SCREEN FUNCTION --
def show_data_selector():
def show_power_data():
back_rect = canvas.create_rectangle(180, 270,
back_text = canvas.create_text(240, 285, text=
def back_to_selector(event=None):
canvas.tag_bind(back_rect, "<Button-1>", back_
canvas.tag_bind(back_text, "<Button-1>", back_
# -- RETURN TO MAIN SCREEN --
def return_to_main():
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS


Welcome to Tkinter Designer
Tkinter Designer uses the Figma API to analyse a design file, then creates the respective code and files needed for your GUI.
Even this GUI was created using Tkinter Designer.
Click here for instructions
Enter the details.
Token ID
uLeY1t3SxoB8DbXt1OKZKiPbvA6rBSnzAV-NPvpn
File URL
-1&node-type=canvas&t=Boy7InR64Cj4nmes-0
Output Path
Generate

FINAL BUILD


AEMD TERRAIN ASSIST CONS
DRONE STATUS:
Altitude: -41.2 m
Battery: 85 %
Command Sent: FORWARD
Live Feed
Data
Settings
Signal:
Logs
Remote monitoring
Follow terminal folde
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
57°F
Sunny
Search



CONCLUSIONS

- Achieved wireless, low-latency sensor monitoring.
- Framed our project using NASA concepts
  - Autonomous sensor nodes
  - Low power edge processing
- Expandable for IoT applications (e.g., smart agriculture).