॥वसुधैव कुटुम्बकम्॥

SYMBIOSIS INTERNATIONAL (DEEMED UNIVERSITY)

(Established under section 3 of the UGC Act 1956)

Re-accredited by NAAC with "A" Grade

Department of Electronics & Telecommunication Engineering

Academic Year 2025-26

Batch 2023-27, Semester - V

Subject: Open Source Technologies (OST) **Credits: 3**

Semester: V **Year: AY 25-26**

Title of the Project: Morse code decoder

Group Information:

Sr no.	PRN	Name
1	23070123019	Amey Kumar
2	23070123025	Anushka Shinde
3	23070123070	Jiya Palod
4	24070123509	Maitreyee Gohad

1. Working of Project:

- **Project Title**: Morse Code Decoder.
- **Description**:
  This is an Arduino-based project where the user inputs Morse code (dots "·" and dashes "–") via two push buttons. The system decodes the Morse sequence into alphanumeric characters and displays them on a 16×2 LCD screen.
- **Tools and Technologies used:**
  1) **Hardware :** Arduino Uno, push buttons, LEDs, resistors, jumper wires, LCD, breadboard.
  2) **Software :** Arduino IDE(C++, liquidCrystal library.
  3) **Documentation and repo :** Github

- **Screenshots of Execution**:

Fig.(a) TinkerCAD Simulation

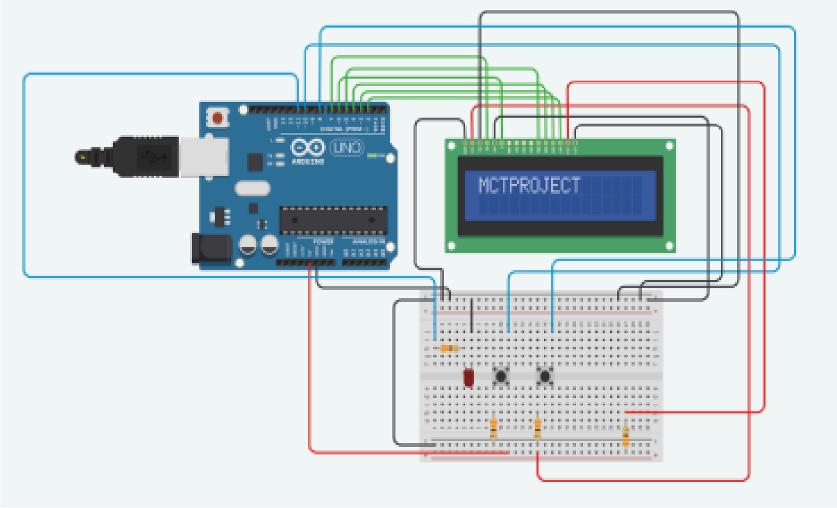
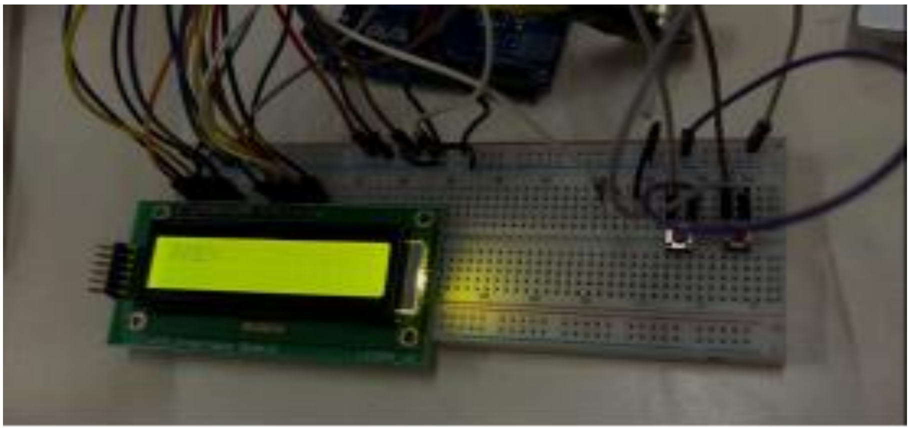
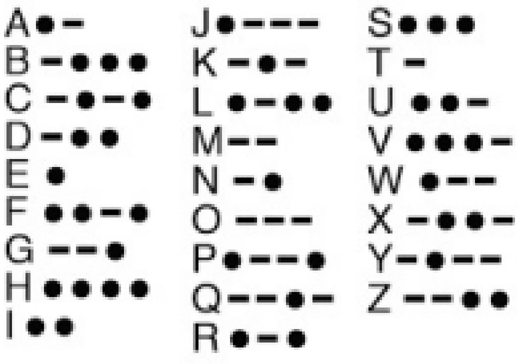


Fig.(b)Breadboard **Fig(c)Morse Code reference**





A	•–	J	•–––	S	•••
B	–•••	K	–•–	T	–
C	–•–•	L	•–••	U	••–
D	–••	M	––	V	•••–
E	•	N	–•	W	•––
F	••–•	O	–––	X	–••–
G	––•	P	•––•	Y	–•––
H	••••	Q	––•–	Z	––••
I	••	R	•–•		

- **How to Run the Project**:
    1) Connect the circuit using the hardware components on breadboard as per the schematic diagram.
    2) Open `MorseDecoder.ino` in **Arduino IDE** (file in repo) .
       Github - https://github.com/jiyapalod/Morse-Code-Decoder
    3) Install LiquidCrystal library.
    4) Select **Arduino Uno** board and correct **COM port**.
    5) Upload the code.
    6) Use dots and dashes using push buttons to enter Morse code.
    7) The decode output is displayed on the LCD.

- **Expected vs Actual Output**:

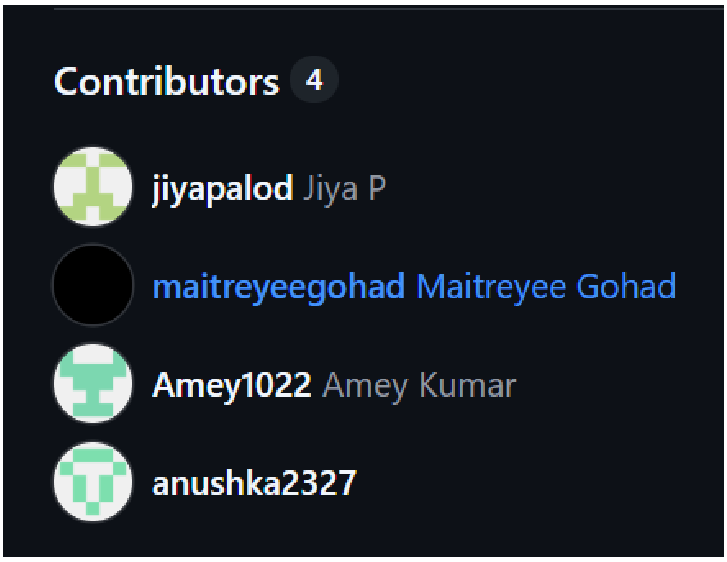
1) Expected Output: When you input a valid Morse code for "A" (·–), the LCD should display "A".
2) Actual Output: The project works correctly most of the time and decodes Morse inputs accurately. However, it occasionally struggles to distinguish between **letters and words**, especially when the pause between inputs is inconsistent.

2. Innovation / Novelty / Contribution:

- **Unique Feature or Enhancement**:

  → This project converted combinations of dots and dashes into readable and meaningful letters and words.

    1. More time gap between inputs created a differentiation between letters and words so that the combinations were made into meaningful sentences.
    2. Two-button integration to make the differentiation between dots and dashes easier.
    3. Used an Arduino microcontroller for all the processing, making it inexpensive and affordable.



Contributors 4

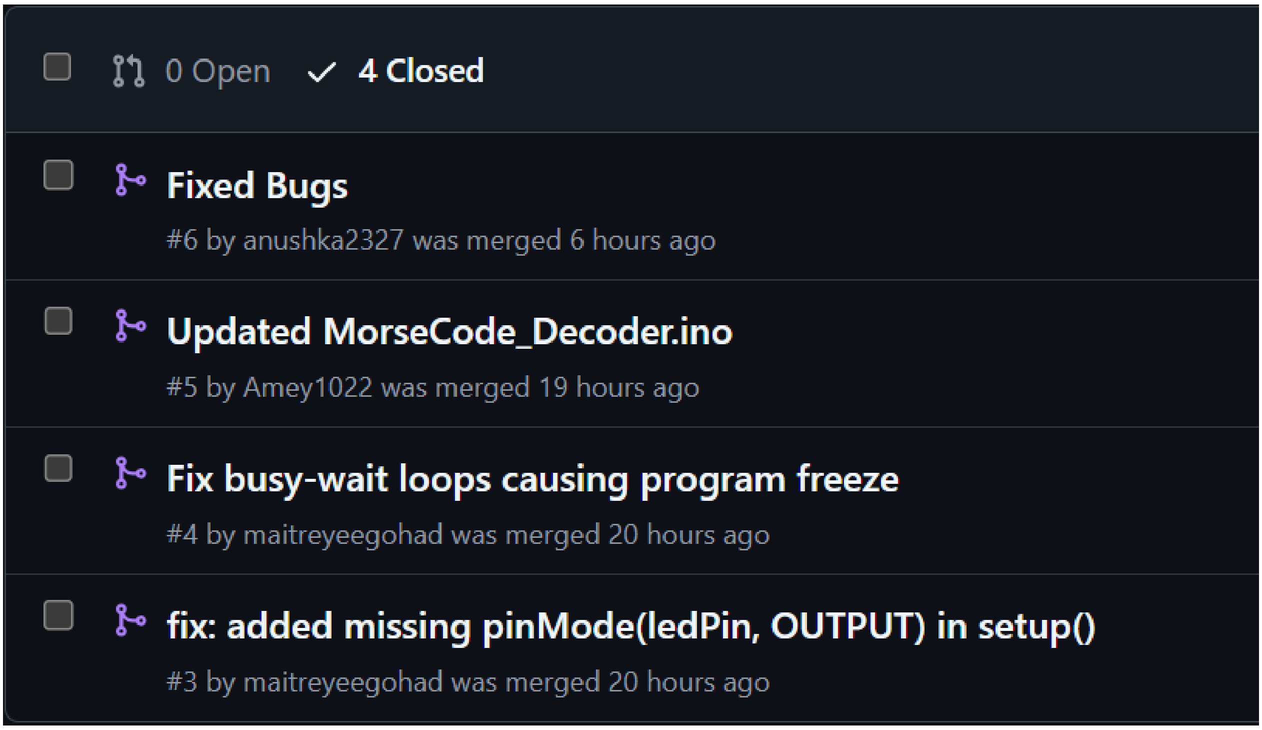
jiyapalod Jiya P

maitreyeegohad Maitreyee Gohad

Amey1022 Amey Kumar

anushka2327

PULL REQUESTS:



□ 0 Open ✓ 4 Closed

□ Fixed Bugs
#6 by anushka2327 was merged 6 hours ago

□ Updated MorseCode_Decoder.ino
#5 by Amey1022 was merged 19 hours ago

□ Fix busy-wait loops causing program freeze
#4 by maitreyeegohad was merged 20 hours ago

□ fix: added missing pinMode(ledPin, OUTPUT) in setup()
#3 by maitreyeegohad was merged 20 hours ago

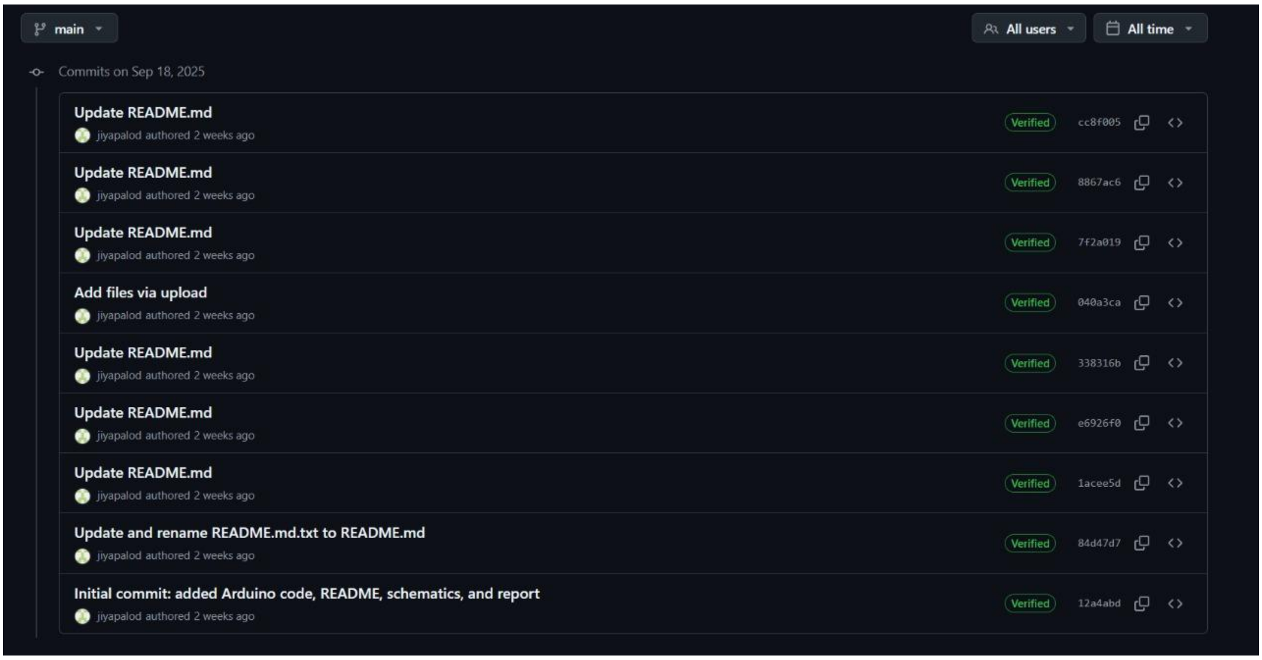
Student Name	Role/Contribution	GitHub Profile	Key Commits / Pull Requests
Jiya Palod	Backend Design, ReadMe	https://github.com/jiyapalod	https://github.com/jiyapalod/Morse-Code-Decoder/commits/main/
Maitreyee Gohad	Automation	https://github.com/maitreyeegohad	https://github.com/jiyapalod/Morse-Code-Decoder/pull/4
Anushka Shinde	Testing	https://github.com/anushka2327	https://github.com/jiyapalod/Morse-Code-Decoder/pull/6
Amey Kumar	Bug Fixes	https://github.com/Amey1022	https://github.com/jiyapalod/Morse-Code-Decoder/pull/5

- **Challenges Faced and Solved**:

1. LCD Integration: The major challenge we faced was the display not working with the arduino when connected. Fixed the issue by changing the LCD panel, making the connection more secure and setting the regulator on the desired level.
2. Button Recognition: Arduino did not recognize button long press and short press. Fixed the issue by using two buttons- 1 for dot ,and 2 for dash. Also increased the detection duration.

3. Timely Submission:

- **Evidence of Progress**: Screenshots of commit history or logs.



Commits on Sep 18, 2025

Update README.md
jiyapalod authored 2 weeks ago
Verified cc8f005

Update README.md
jiyapalod authored 2 weeks ago
Verified 8867ac6

Update README.md
jiyapalod authored 2 weeks ago
Verified 7f2a019

Add files via upload
jiyapalod authored 2 weeks ago
Verified 040a3ca

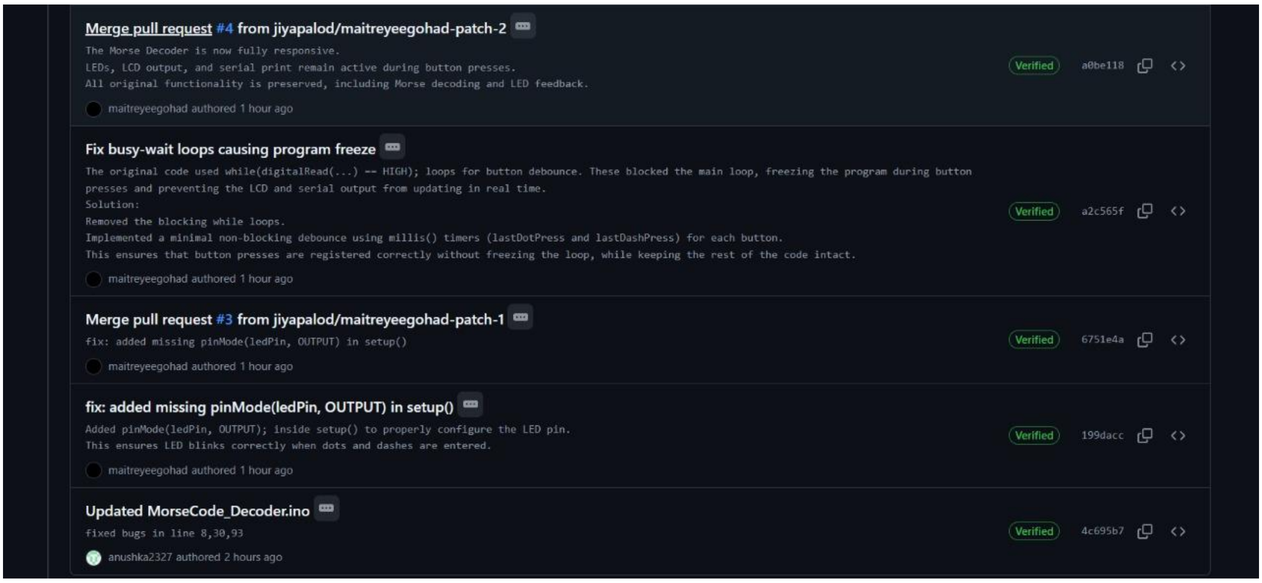
Update README.md
jiyapalod authored 2 weeks ago
Verified 338316b

Update README.md
jiyapalod authored 2 weeks ago
Verified e6926f0

Update README.md
jiyapalod authored 2 weeks ago
Verified 1acee5d

Update and rename README.md.txt to README.md
jiyapalod authored 2 weeks ago
Verified 84d47d7

Initial commit: added Arduino code, README, schematics, and report
jiyapalod authored 2 weeks ago
Verified 12a4abd



Merge pull request #4 from jiyapalod/maitreyeegohad-patch-2

```
The Morse Decoder is now fully responsive.
LEDs, LCD output, and serial print remain active during button presses.
All original functionality is preserved, including Morse decoding and LED feedback.
```

maitreyeegohad authored 1 hour ago — Verified a0be118

Fix busy-wait loops causing program freeze

```
The original code used while(digitalRead(...) -- HIGH); loops for button debounce. These blocked the main loop, freezing the program during button
presses and preventing the LCD and serial output from updating in real time.
Solution:
Removed the blocking while loops.
Implemented a minimal non-blocking debounce using millis() timers (lastDotPress and lastDashPress) for each button.
This ensures that button presses are registered correctly without freezing the loop, while keeping the rest of the code intact.
```

maitreyeegohad authored 1 hour ago — Verified a2c565f

Merge pull request #3 from jiyapalod/maitreyeegohad-patch-1

```
fix: added missing pinMode(ledPin, OUTPUT) in setup()
```

maitreyeegohad authored 1 hour ago — Verified 6751e4a

fix: added missing pinMode(ledPin, OUTPUT) in setup()

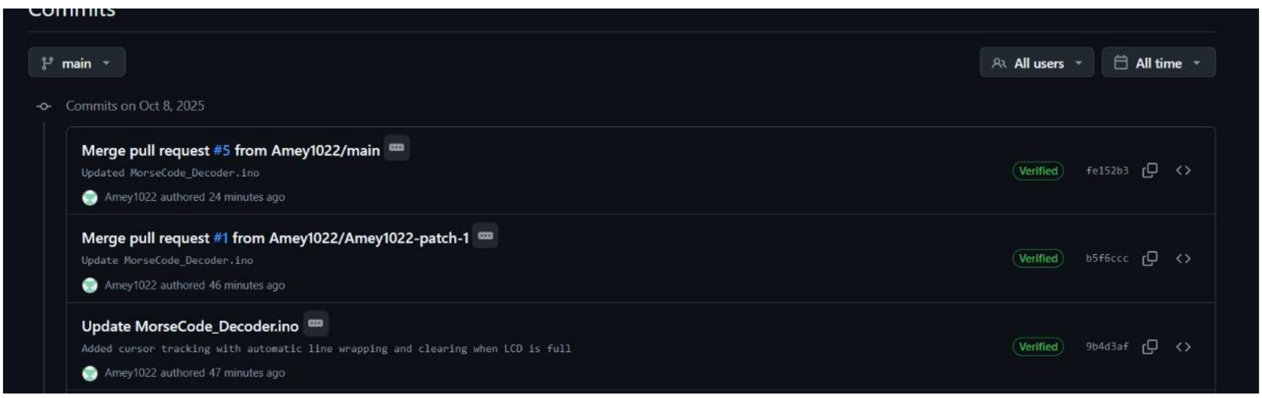
```
Added pinMode(ledPin, OUTPUT); inside setup() to properly configure the LED pin.
This ensures LED blinks correctly when dots and dashes are entered.
```

maitreyeegohad authored 1 hour ago — Verified 199dacc

Updated MorseCode_Decoder.ino

```
fixed bugs in line 8,30,93
```

anushka2327 authored 2 hours ago — Verified 4c695b7



Commits on Oct 8, 2025

Merge pull request #5 from Amey1022/main

```
Updated MorseCode_Decoder.ino
```

Amey1022 authored 24 minutes ago — Verified fe152b3

Merge pull request #1 from Amey1022/Amey1022-patch-1

```
Update MorseCode_Decoder.ino
```

Amey1022 authored 46 minutes ago — Verified b5f6ccc

Update MorseCode_Decoder.ino

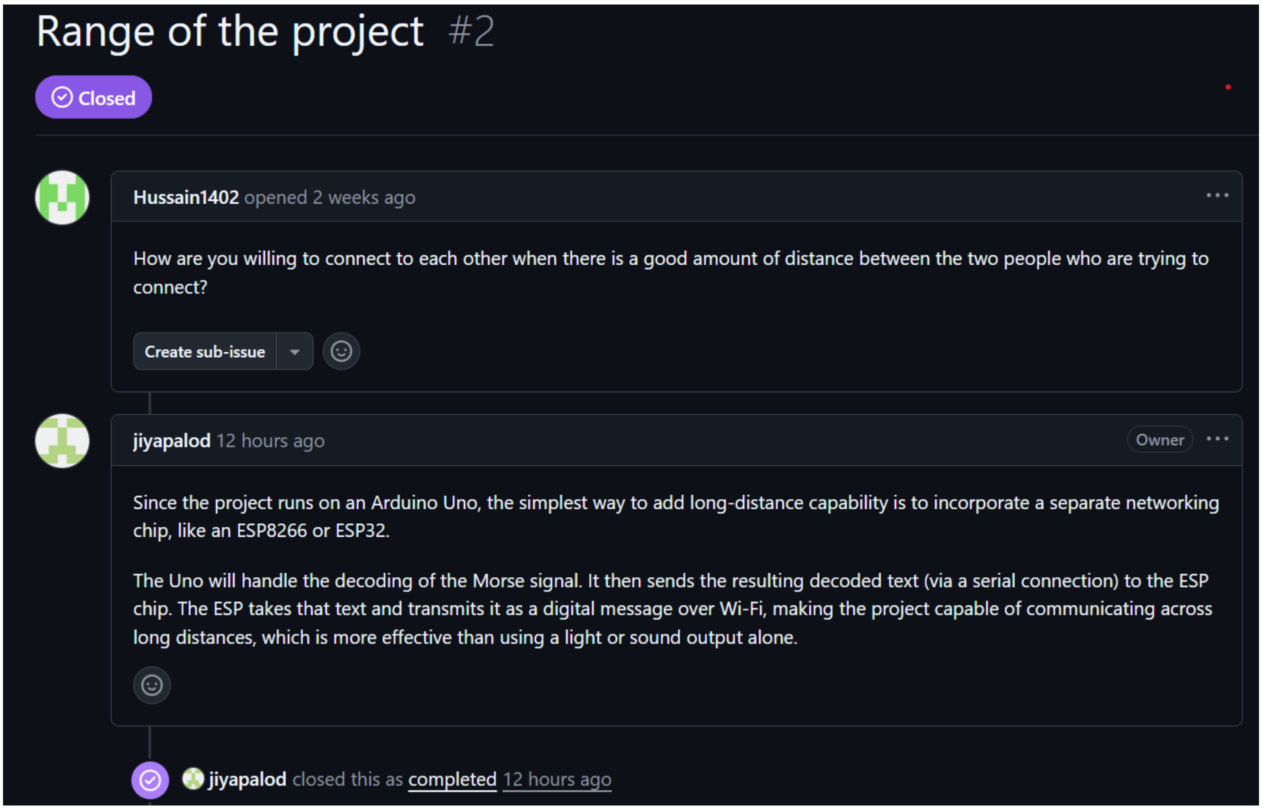
```
Added cursor tracking with automatic line wrapping and clearing when LCD is full
```

Amey1022 authored 47 minutes ago — Verified 9b4d3af

4. Pushing the Project to GitHub:

- **GitHub Repository Link**: Morse-Code Decoder

- **Issues:**



Range of the project #2

Closed

Hussain1402 opened 2 weeks ago

How are you willing to connect to each other when there is a good amount of distance between the two people who are trying to connect?

jiyapalod 12 hours ago

Since the project runs on an Arduino Uno, the simplest way to add long-distance capability is to incorporate a separate networking chip, like an ESP8266 or ESP32.

The Uno will handle the decoding of the Morse signal. It then sends the resulting decoded text (via a serial connection) to the ESP chip. The ESP takes that text and transmits it as a digital message over Wi-Fi, making the project capable of communicating across long distances, which is more effective than using a light or sound output alone.

jiyapalod closed this as completed 12 hours ago

- **README File**: It contains all the necessary information related to the project repository which explains the objective and working of the project. It helps users understand the reason and operation of the given project.

- **Git commands used:**
  1. git init: Initialize a new Git repository
  2. git add: Add all project files to staging area
  3. git commit -m "initial commit" : Commit the files with a message
  4. git branch -M main: Rename the default branch to 'main'
  5. git remote add origin https://github.com/jiyapalod/morse-code-decoder.git :- Connect the local repo to GitHub
  6. git push -u origin main: Push the files to GitHub

5. Conclusion:

We successfully built a morse code decoder and pushed the project to the github repository. This repository contains all the documents and codes related to the project. We also modified the code to make it more efficient and add new enhancements while solving issues raised by other users.