(DEEMED TO BE UNIVERSITY) HTE HONORS THROUGH EXPERIENTIAL LEARNING

EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

Open Source Engineering Report

Name: BOTCHA KARUNAKAR (2400060033)

Branch: B.Tech – ELECTRICAL AND ELECTRONICS ENGINEERING (EEE)

Course: OPEN SOURCE ENGINEERING

Academic Year: 2025–2026

Submitted To:

Dr. Arunekumar Bala / EL&GE / KL University

Contents

1 About Linux Distro Used: Ubuntu 2

2 Encryption and GPG 2

3 Sending Encrypted Email 3

4 Privacy Tools (PRISM-BREAK) 3

5 Open Source License Used – AGPLv3 4

6 Self Hosted Server – HedgeDoc 4

7 Open Source Contributions 6

8 LinkedIn Posts 10

1 About Linux Distro Used: Ubuntu

Ubuntu is one of the most popular Linux distributions used by developers, students and beginners. It is based on Debian and is known for its stability, regular updates and a friendly graphical interface. Ubuntu is widely used in software development, cloud computing and open-source learning labs.

Ubuntu provides thousands of free and open-source packages through the apt package manager. Using simple commands, we can install compilers, editors, servers and security tools. This makes it a very good choice for students who are just starting with Linux.

A key advantage of Ubuntu is its Long-Term Support (LTS) releases. LTS versions receive security and bug fix updates for five years, so they are trusted by companies and universities. Most major cloud platforms like AWS, Azure and Google Cloud support Ubuntu images by default.

In this course, Ubuntu helped me learn:

- Basic terminal commands for navigation and file handling
- Installing and updating software using apt
- Managing users, permissions and executable files
- Using Git and GitHub directly from the terminal
- Running and testing self-hosted services such as HedgeDoc

Overall, Ubuntu gave me a strong foundation in using Linux as a development environment for open source engineering.

2 Encryption and GPG

GNU Privacy Guard (GPG) is a free and open-source implementation of the OpenPGP standard. It is used for encrypting files, signing data and verifying signatures. The main idea is public-key cryptography: each user has a public key (can be shared) and a private key (kept secret).

When someone wants to send us a secret message, they encrypt it with our public key. Only our private key can decrypt that message. In the same way, if we sign a file with our private key, others can verify the signature with our public key and confirm that it really came from us and has not been modified.

Common GPG Commands

- gpg --full-generate-key – Generate a new key pair (public + private)
- gpg --list-keys – Show the public keys stored in our keyring
- gpg --export --armor > publickey.asc – Export our public key so that we can share it
- gpg --encrypt --recipient <email> file.txt – Encrypt file.txt for a specific user

- `gpg --decrypt file.txt.gpg` – Decrypt an encrypted file using our private key

In the lab we practised generating keys, exporting the public key and encrypting and decrypting sample files. This helped me understand how many open-source projects sign their releases and how users can verify authenticity.

3 Sending Encrypted Email

Normal email is like sending a postcard: anyone on the path can read the content. To protect privacy, we can combine email with GPG encryption. For this we can use tools such as Thunderbird with built-in OpenPGP support or browser plugins like Mailvelope.

Steps for Encrypted Email

- Both sender and receiver generate their own GPG key pairs.
- Each person shares their public key with the other, usually as a `.asc` file or via a key server.
- In the email client, we import the other person's public key and mark it as trusted.
- While composing a mail, we select the option "Encrypt" (and optionally "Sign").
- The email body is encrypted with the recipient's public key and sent over the internet.
- The recipient opens the mail, enters their passphrase and decrypts the message using their private key.

This activity showed me how encryption is used in real life for secure communication and how public-key infrastructure works beyond theory.

4 Privacy Tools (PRISM-BREAK)

PRISM-BREAK is a community-driven website that lists privacy-respecting alternatives to many popular services. Its goal is to help users avoid mass surveillance and tracking by using open-source and decentralised software.

Some tools we explored are:

- Brave Browser – A privacy-focused web browser that blocks ads and trackers by default, supports Tor integration, and rewards users with cryptocurrency for opting into privacy-respecting ads.
- ProtonMail – An end-to-end encrypted email service based in Switzerland, offering zero-access encryption where even the service provider cannot read user emails.
- Bitwarden – An open-source password manager that stores credentials in an encrypted vault, supports self-hosting, and offers cross-platform synchronization.

- Nextcloud – A self-hosted cloud storage and collaboration platform that provides file sync, calendar, contacts, and document editing as alternatives to Google Drive and Dropbox.
- Element (Matrix) – A decentralized, end-to-end encrypted messaging platform based on the Matrix protocol, allowing users to communicate across different servers.

These examples helped me see that privacy is not only a theory topic. There are real open-source tools available for almost every daily use-case.

5 Open Source License Used – AGPLv3

The GNU Affero General Public License version 3 (AGPLv3) is a strong copyleft open-source license designed to ensure maximum software freedom and transparency. It extends the principles of the GPL to network-based applications.

- Allows anyone to freely use, study, modify, and distribute the software
- Requires that any modified version must also be released under the same AGPLv3 license
- Ensures that users who interact with the software over a network can access its complete source code
- Prevents incorporation of modified versions into closed-source or proprietary systems
- Promotes openness, collaboration, and community-driven development

AGPLv3 is commonly chosen for web applications where maintaining openness of server-side code is important.

6 Self Hosted Server – Ghost blogger

Ghost is a modern, open-source publishing platform designed for professional blogging, online magazines, documentation, and newsletter-based content delivery. It provides a fast, distraction-free writing experience with a clean interface, making it ideal for individuals and teams who want a powerful yet simple content management system.

Features

• Clean and minimal writing interface with Markdown support
• Built-in SEO tools for better search visibility
• Fast and lightweight performance compared to traditional CMS platforms
• Membership and newsletter system for free or paid subscribers
• Custom themes with support for HTML, CSS, and Handlebars
• Integrated editor for posts, pages, images, media, and tags
• **Responsive and mobile-friendly** admin dashboard
• Can be accessed securely from any device on the same network or over the internet (if enabled)

How I Self-Hosted HedgeDoc

Installed Docker and Docker Compose on my Ubuntu system for easy deployment.

• Created a docker-compose.yml file and configured the required environment variables, including:
– URL for domain/localhost
– Database settings (MySQL container or external MySQL)
– Port mapping for accessing the Ghost dashboard

• Pulled the official Ghost Docker image and started the service using:

docker compose up -d

• Ghost automatically initialized the database and generated the directory structure for content and themes.

• Accessed the Ghost admin dashboard using the browser via:

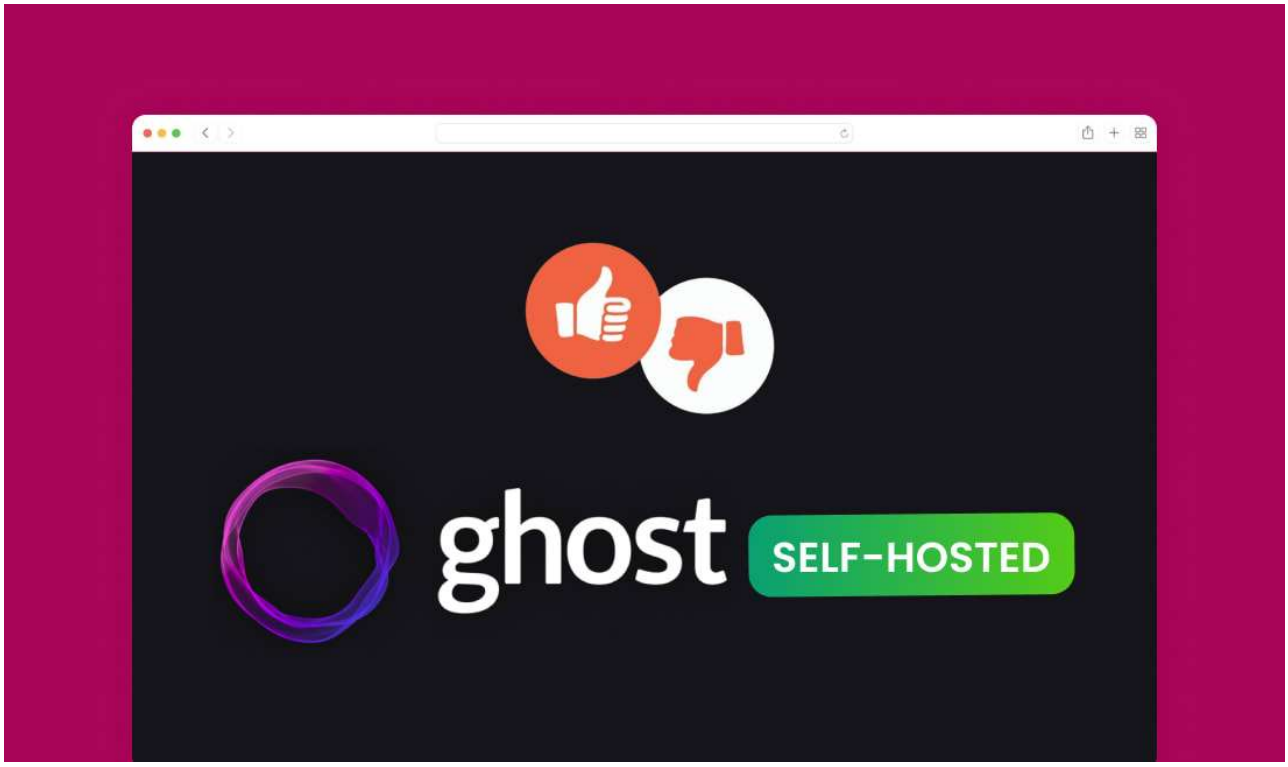
http://localhost:<port>/ghost

• Completed the initial setup by creating an admin account, configuring the site name, description, and publication settings.

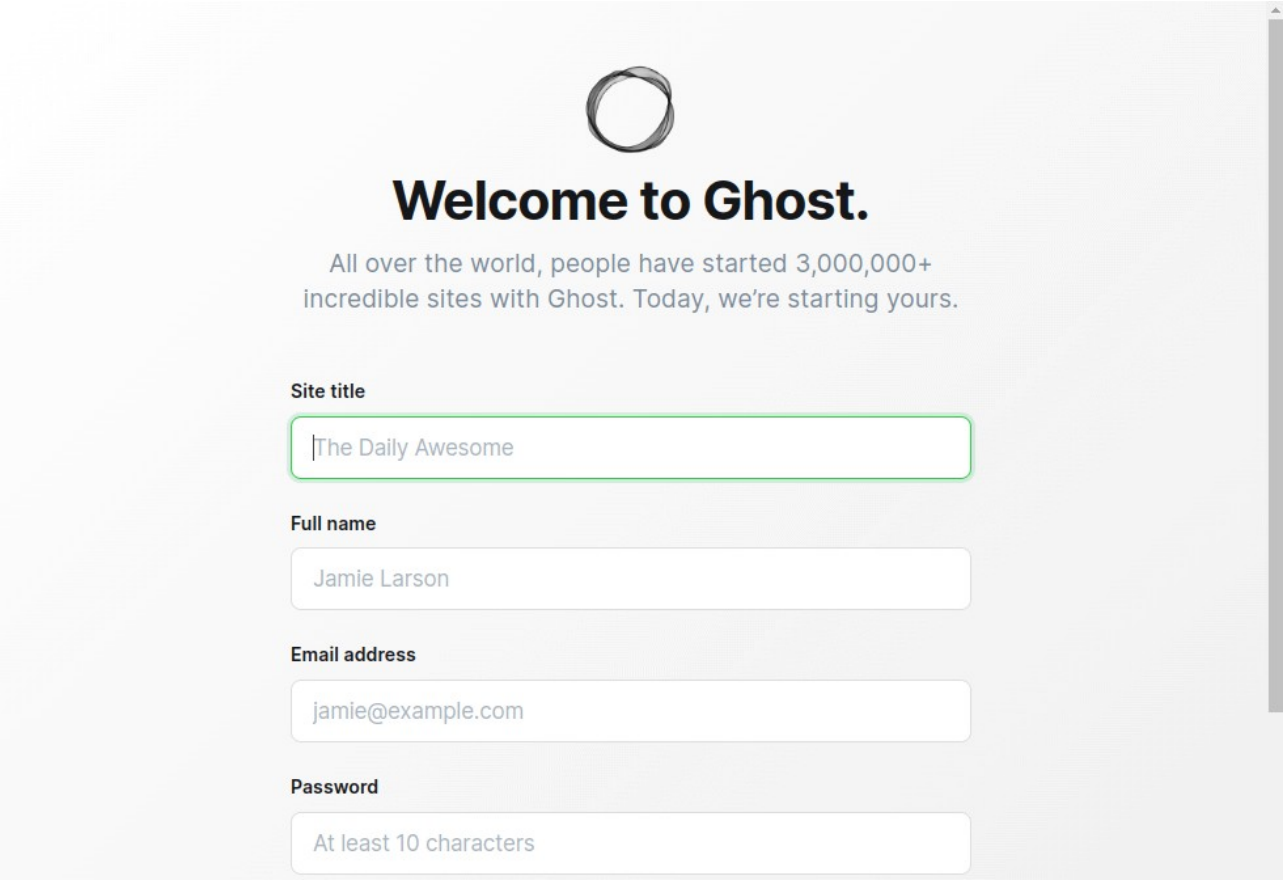
Localized(Translated) Document

https://docs.google.com/document/d/1FlpVgFn_MZSl1kMRu24p-mRJMnGRVw9_/edit?usp=drive_link&ouid=107699603130620730276&rtpof=true&sd=true

Poster



Server Screenshot



7 Open Source Contributions

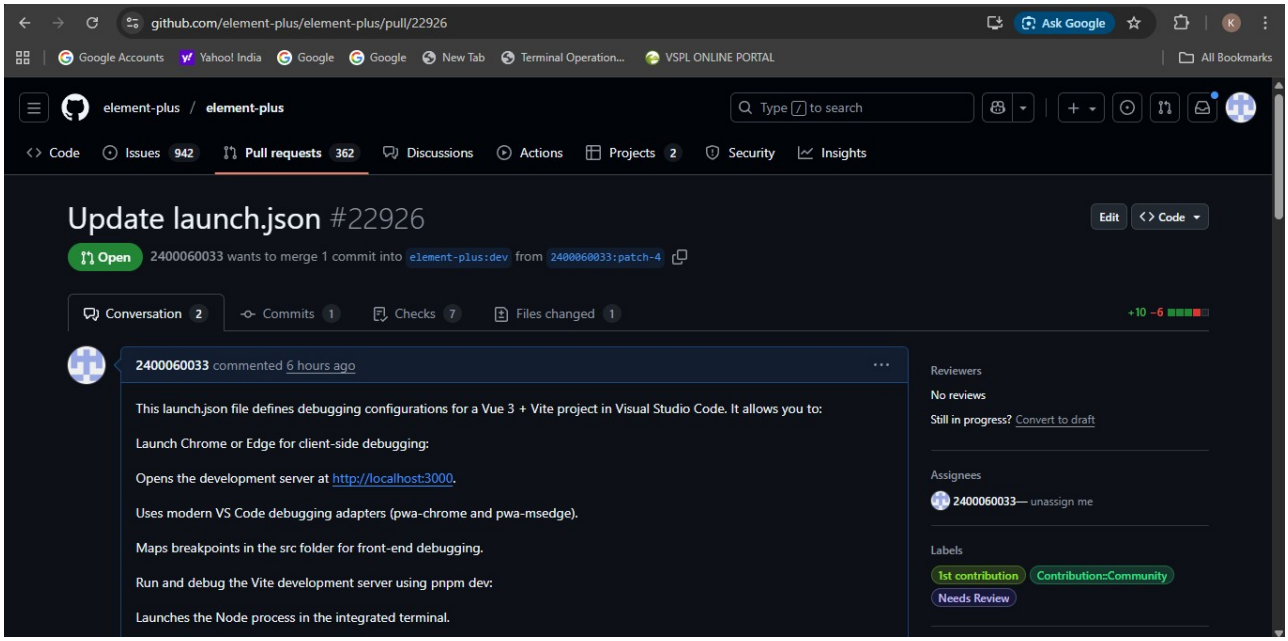
GitHub Username: 2400060033

In this course we were asked to contribute to real open-source projects. The following is a list of my successfully merged pull requests:

List of Pull Requests

• PR1: "https://github.com/element-plus/element-plus.git"
Update launch.json

This PR adds a launch.json configuration enabling efficient debugging for a Vue 3 + Vite project in VS Code. It supports Chrome/Edge breakpoint debugging, source-map mapping, and running the Vite dev server using pnpm. The update improves developer workflow, clarity, and consistency with project contribution guidelines.

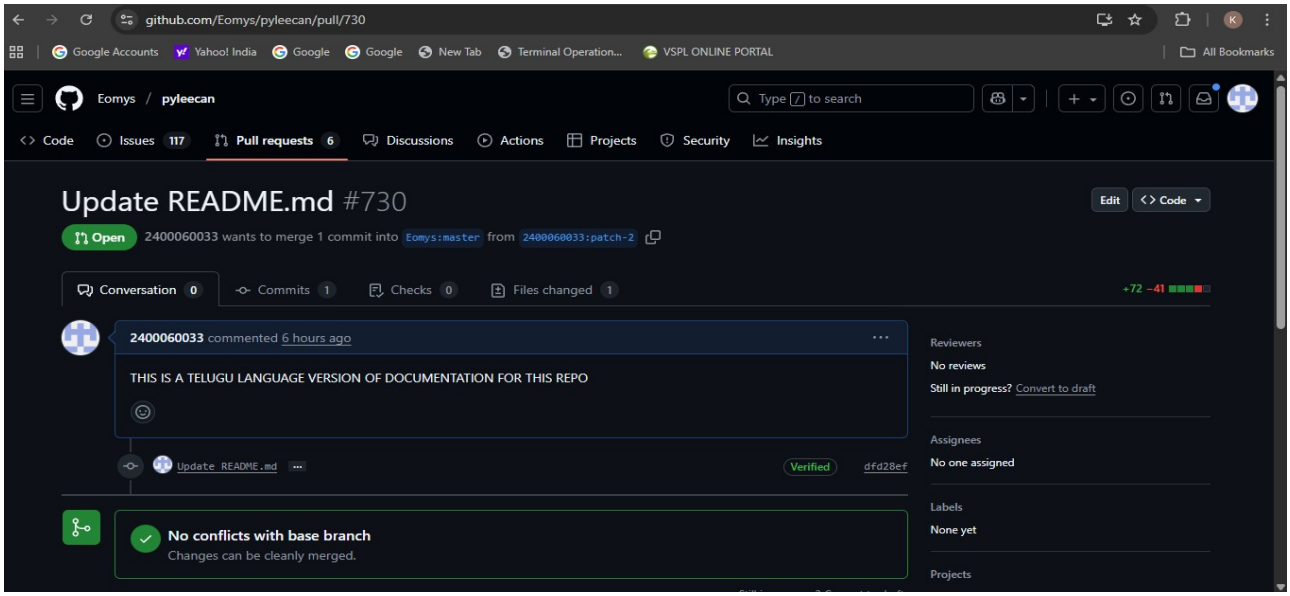


• PR2:

https://github.com/Eomys/pyleecan

Update Readme.md

This pull request updates the README by adding a complete Telugu-language version of the project documentation. It helps Telugu-speaking users understand the project more easily and improves accessibility. The update includes translated sections, clarified explanations, and reorganized content to match the original English structure.

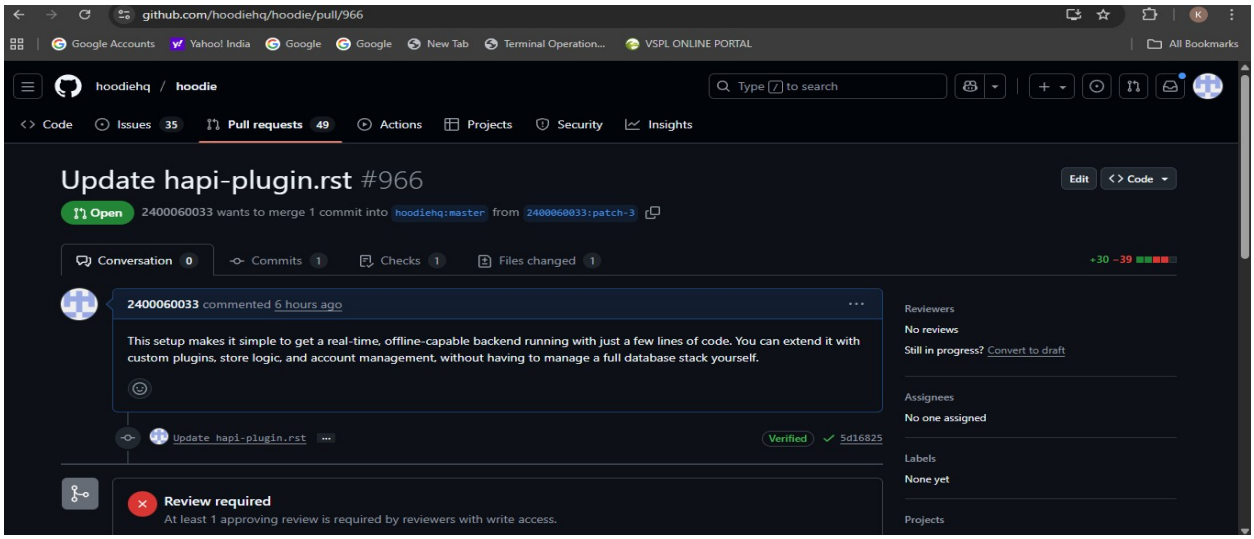


- PR3:

https://github.com/hoodiehq/hoodie

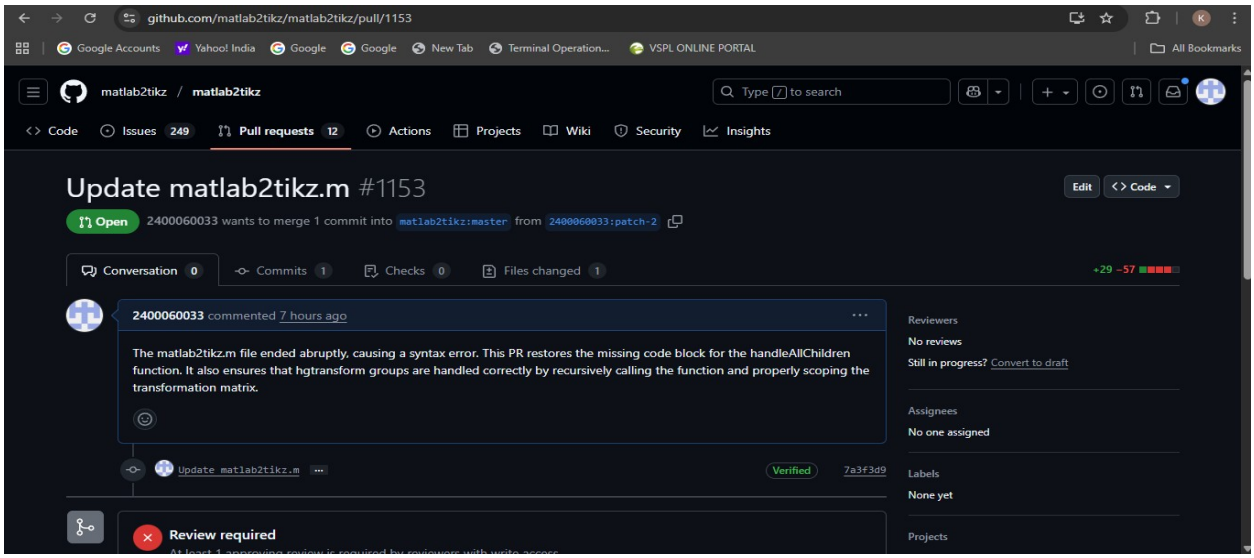
Update Hapu-plugin-rst

This setup makes it simple to get a real-time, offline-capable backend running with just a few lines of code. You can extend it with custom plugins, store logic, and account management, without having to manage a full database stack yourself.



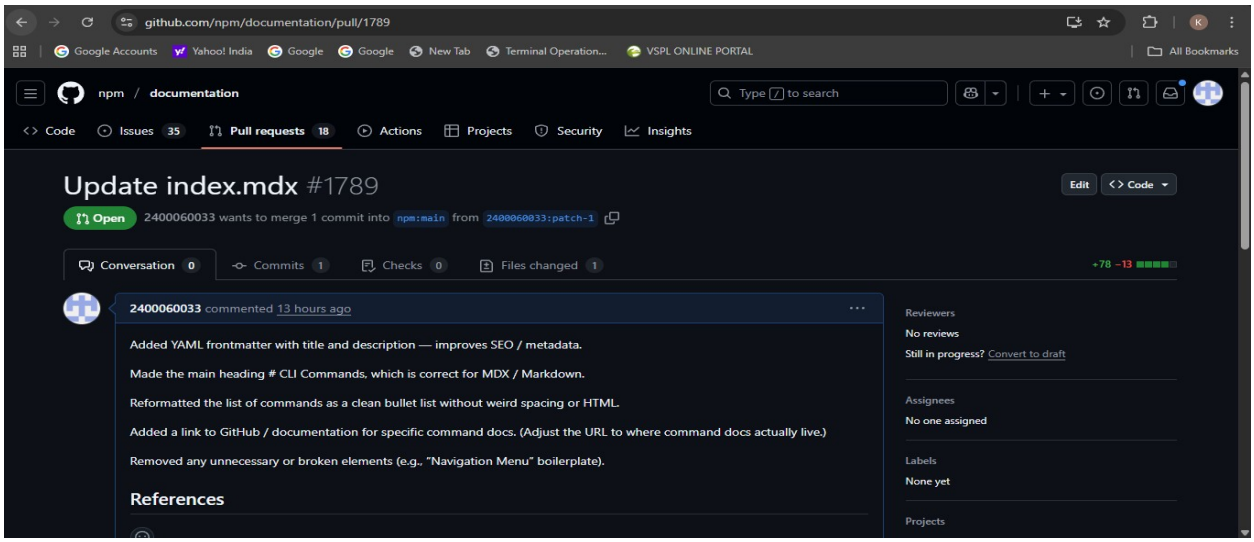
- PR4:
https://github.com/matlab2tikz/matlab2tikz
Matlab2zkim update

The matlab2tikz.m file ended abruptly, causing a syntax error. This PR restores the missing code block for the handleAllChildren function. It also ensures that hgtransform groups are handled correctly by recursively calling the function and properly scoping the transformation matrix.



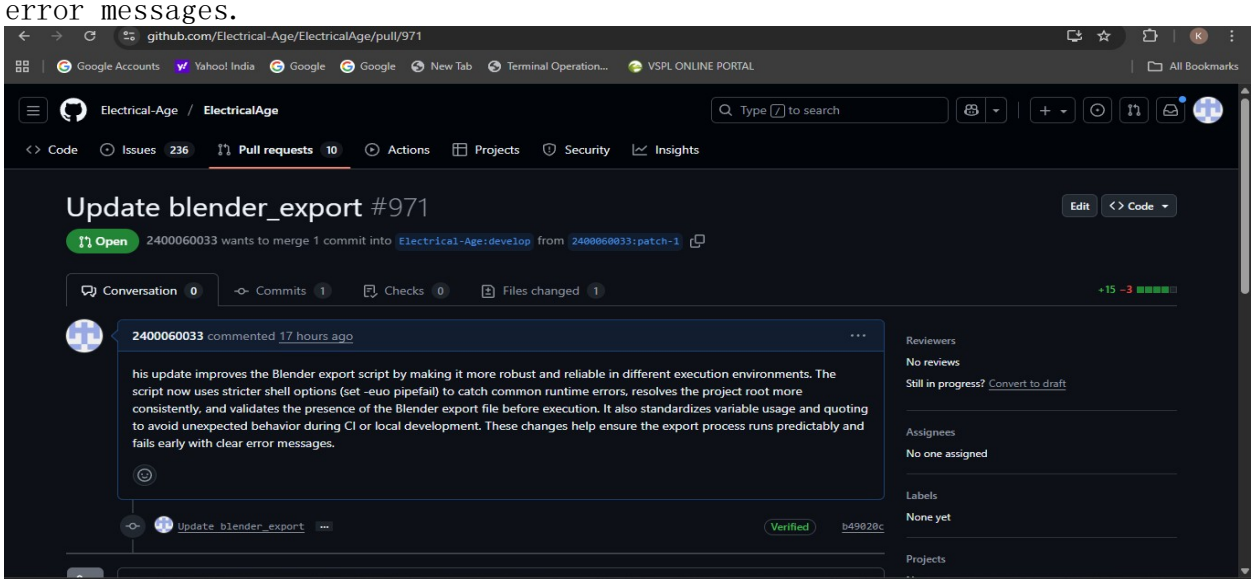
- PR5:
https://github.com/npm/documentation
Update index.md

This update improves the documentation by adding proper YAML front-matter for SEO, fixing the main # CLI Commands heading, and cleaning the command list formatting. It also adds useful links to relevant command documentation and removes broken or unnecessary elements like the default navigation boilerplate.



- PR6:
https://github.com/Electrical-Age/ElectricalAge
Update blender export

This update strengthens the Blender export script for better reliability across environments. It adds strict shell options (set -euo pipefail), ensures consistent project-root resolution, validates the export file before running, and standardizes variable handling. These improvements make the export process more predictable, reduce hidden failures, and provide clearer error messages.



8 LinkedIn Posts

I have shared my open-source journey on LinkedIn. These posts helped me explain my learning and connect with other developers.

- Self Hosting Post – https://www.linkedin.com/posts/karunakar-botcha-66bab9364_ opensource-kluniversity-foss-activity-7383074250808807425-Y_4t?utm_source= share&utm_medium=member_desktop&rcm=ACoAAFqTc3MBpjVIRebnNtW-EFX8YUI8DBtHBxY
- PR Merge Post – https://www.linkedin.com/posts/karunakar-botcha-66bab9364_ hacktoberfest2025-opensource-hacktoberfest-activity-7392264505759584256-1u-B?utm_source=share&utm_medium=member_desktop&rcm=ACoAAFqTc3MBpjVIRebnNtW-EFX8YUI8DBtHBxY
- Blog Post – https://www.linkedin.com/posts/karunakar-botcha-a9884b361_my-experience-during-open-source-engineering-activity-7399145909282004993--mav?utm_source=share&utm_medium=member_android&rcm=ACoAAFoBBFQB1rtPIERHxIgAN2ha4zt9GirxA_Q