



Open Source Engineering Report

Student Name: Bangaru Krishna Sri Chaitanya (2400030120)

Branch: B.Tech – Computer Science Engineering (CSE)

Course: Open Source Engineering

Academic Year: 2025–2026

Submitted To:

(Dr. Sripath Roy Koganti / EL&GE / KL University)

In Partial Fulfilment of the Requirements of

Open Source Engineering Laboratory

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 – Paperless-ngx	4
7	Open Source Contributions	6
8	LinkedIn Posts	8

1 About Linux Distro Used: Ubuntu

Ubuntu is a popular, beginner-friendly Linux distribution based on Debian. It provides a stable, secure, and user-friendly environment for learning Linux. Widely used in servers, cloud computing, and development environments, making it industry-relevant.

Includes a powerful package manager (APT) for easy installation and updating of software. Using simple commands, we can install compilers, editors, servers and security tools. Offers strong community support, documentation, and tutorials.

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:

- How to navigate and use the Linux command line effectively.
- Installing and updating software using `apt`
- Understanding file system hierarchy and directory management.
- Using Git and GitHub directly from the terminal
- Running and testing self-hosted services such as Scribble
- How to work with text editors like nano, vim, or gedit.

Ubuntu, a widely used and beginner-friendly Linux distribution, provided a stable and reliable environment throughout the course.

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

Sending encrypted email ensures that only the intended recipient can read the message. Using GPG, we generate key pairs and exchange public keys with the recipient. The email message is written in a file and encrypted using the recipient's public key.

Steps for Encrypted Email

your own public and private key pair. The public key is shared with others, while the private key remains confidential.

- 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 recipient downloads the encrypted file and uses their private key to decrypt and read the original message safely.
- 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 an online project that provides a collection of privacy-respecting software, tools, and services. The main goal of PRISM-BREAK is to help people protect their digital privacy by recommending open-source, secure, and decentralized alternatives to popular services that may track users.

Some tools we explored are:

- **Tor Browser** – A browser that hides your identity by routing traffic through Tor.
- **Firefox** – Open-source browser with strong privacy customization.
- **Signal** – Encrypted messaging app with strong privacy features.
- **WireGuard** – Fast and modern VPN protocol.

- **LineageOS** – A free Android-based operating system that removes bloatware and gives more control over permissions.

These examples helped me see that privacy is not only a theory topic. Protecting personal privacy while using the internet. 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

6 Self Hosted Server – Paperless-ngx

Paperless-ngx is an open-source, self-hosted document management system. It helps users digitise paperwork by scanning, uploading and organising documents in one central place.

Features

- Web-based interface for uploading and viewing documents
- OCR support for making scanned documents searchable
- Tagging, filtering and full-text search
- Runs completely on our own machine or server
- Runs completely on our own machine or server, giving better privacy


How I Self-Hosted Paperless-ngx

- Installed the required dependencies (Docker and Docker Compose) on Ubuntu.
- Created a docker-compose.yml file using the configuration from the Paperless-ngx documentation.
- Pulled the Docker images and started the containers using `docker compose up -d`.
- Accessed the Paperless-ngx dashboard in the browser at `http://localhost:8000`.
- Uploaded sample PDF documents and tested searching and tagging features.


paperless-ngx

Translated Documnet

poster



OPEN SOURCE ENGINEERING



Paperless-ngx is an open-source document management system that helps you store, organize, and search your digital documents efficiently. It allows you to go fully paperless by scanning, tagging, and securely managing all your files in one place.

LICENCE :GNU General Public License version 3 (GPL-3.0).

- **OCR (Optical Character Recognition) on scanned documents to extract selectable/searchable text.**
- **Full-text search with autocomplete, relevance sorting, and ability to highlight matching content.**
- **Tags, correspondents, document types, and custom metadata fields for organizing documents.**

Atmakuru Sathvik-2400040068
Bangaru Krishna Sri Chaitanya-2400030120

7 Open Source Contributions

GitHub Username: **krishna3324**

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

List of Pull Requests

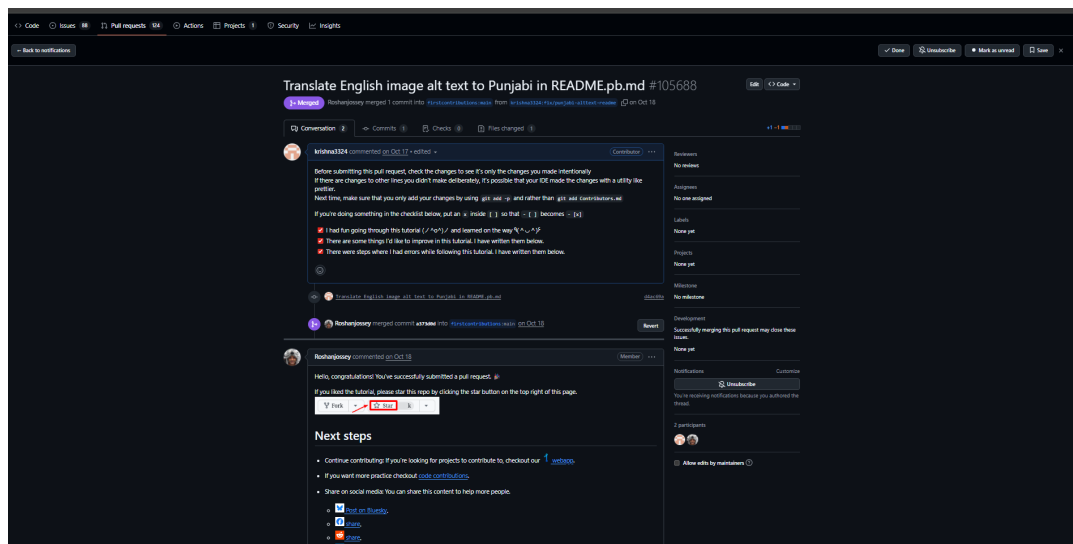
- **PR 1: First contribution** – “Add my name to Contributors list” (First Contribution PR)

This was my first ever pull request on GitHub. I forked the repository, cloned it to my local system and added my name to the `Contributors.md` file. Then I committed the change and pushed it back to my fork, and finally created a pull request to the main repository.

Through this contribution I learnt the full basic workflow of Git and GitHub:

- how to fork a project,
- how to create a new branch,
- how to commit and push changes,
- and how to wait for maintainers to review and merge the PR.

The PR was **merged**, which gave me confidence to start working on other open source issues.



- **PR 2: KLGLUG / Y24OpenSourceEngineering** – “Added Paperless-ngx self-hosted project documentation in Telugu” (PR #85)

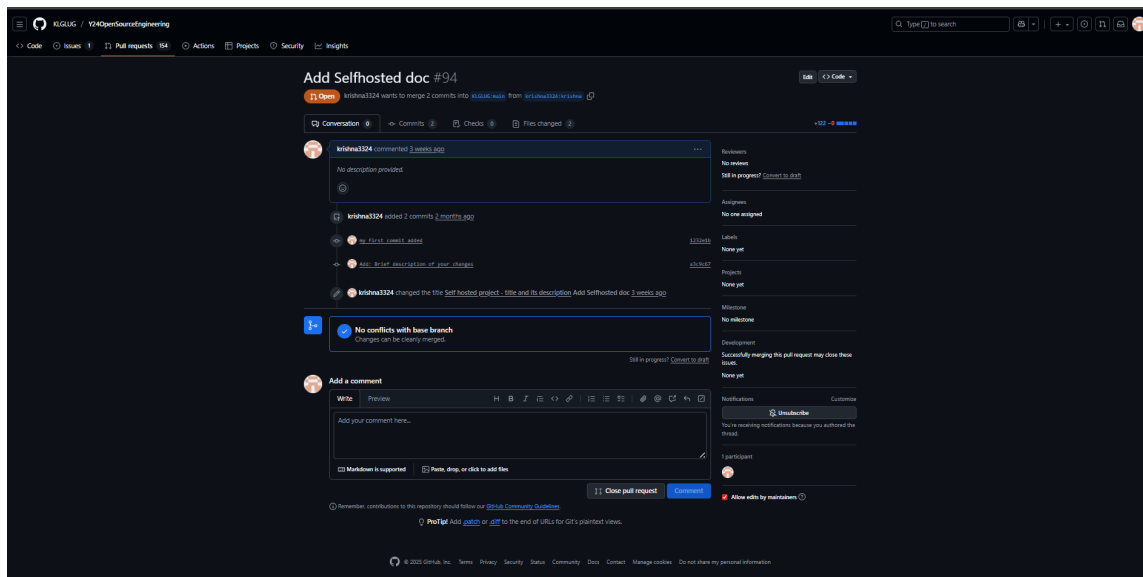
In this pull request I wrote detailed documentation in Telugu explaining how to self-host Paperless-ngx. My aim was to help Telugu-speaking students easily understand the steps and try self-hosting on their own systems.

This contribution taught me:

- how to write clear step-by-step documentation,

- how to follow repository folder structure and naming rules,
- and how localisation makes open source projects more inclusive.

At the time of writing this report, the PR is **open and under review**.

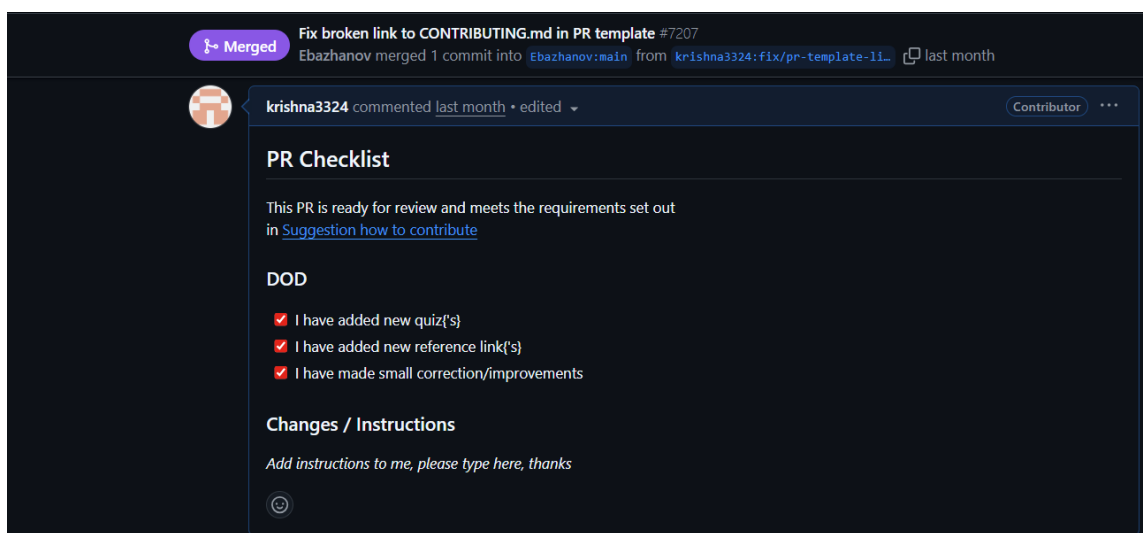


PR 3: linked in skill assessments – "Add my name to Contributors list"

In this contribution, I submitted a Pull Request (PR) to fix issues and improve documentation in the project. The PR contained updates such as adding new quiz entries, adding new reference links, and making minor corrections to the existing template.

This contribution taught me:

- How to fix documentation issues like broken links and formatting.
- How to write clear commit messages and describe changes professionally.

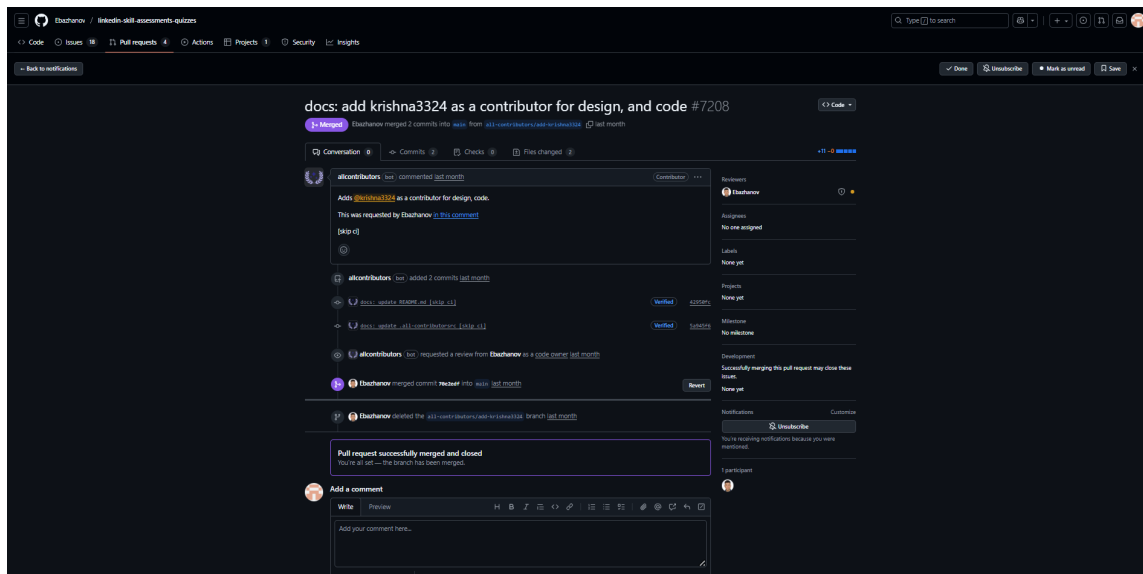


PR 4: linked in skill assessments -"Add my name to Contributors list"

In this contribution, I was officially added as a contributor to the project for my work in design and code. The project maintainers recognized my earlier contributions and updated the repository's contributor list to include my name and username.

This contribution taught me:

- The importance of clear contribution categories like design, code, and documentation.
- How maintainers review, approve, and merge contributor-related PRs.

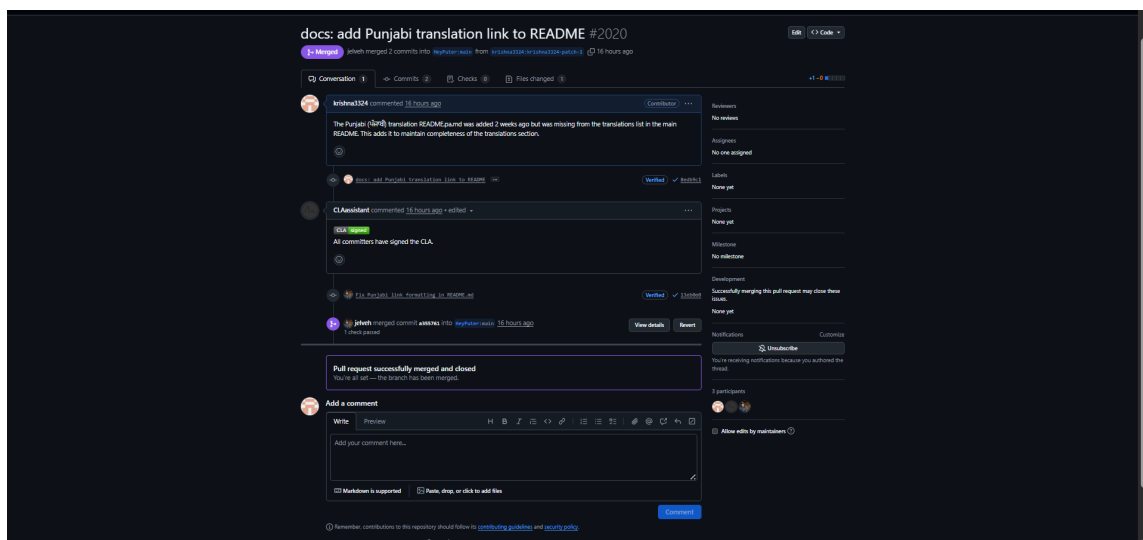


PR 5:puter -” Add my name to Contributors list”

In this contribution, I added the missing Punjabi translation link to the project’s main README file. The Punjabi translation file already existed in the repository, but it was not listed in the translations section.

This contribution taught me:

- The importance of keeping documentation complete and consistent.
- How to identify missing links or gaps in project README files.



8 LinkedIn Posts

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

- Post about my first merged PR in first-contributions – https://www.linkedin.com/posts/krishna-bangaru-88a616366_hacktoberfest2025-opensource-hacktoberfest-a-utm_source=share&utm_medium=member_desktop&rcm=ACoAAFrcJmoBs6fJitGGPnFbVthW-Aud-
- Open Source Blog – https://www.linkedin.com/posts/krishna-bangaru-88a616366_my-journey-in-open-source-engineering-activity-7398426133295218688-rVe5?utm_source=share&utm_medium=member_desktop&rcm=ACoAAFrcJmoBs6fJitGGPnFbVthW-Aud-
- Post about self-hosting Paperless-ngx – https://www.linkedin.com/posts/krishna-bangaru-8we-successfully-deployed-paperless-ngx-a-activity-7390447052939436033-eUQ1?utm_source=share&utm_medium=member_desktop&rcm=ACoAAFrcJmoBs6fJitGGPnFbVthW-Aud-