Open Source Engineering

Figure 1: Linux makes Open Source possible

Student Details

Name:	LAKKAKULA YASHVANTH KUMAR
Roll Number:	2400040031
Department:	Electronics and Communication Engineering
University:	KL University
Course:	Open Source Engineering
Semester:	ODD SEMESTER

Submitted to:

Dr.AruneKumar
Department of Computer Science and Engineering
KL University

Contents

1 Linux Distribution

1.1 Distribution Used: Ubuntu 24.04 LTS

For this course named Open Source Engineering, I have used **Ubuntu 22.04 LTS** as my primary operating system.

1.2 Why Ubuntu?

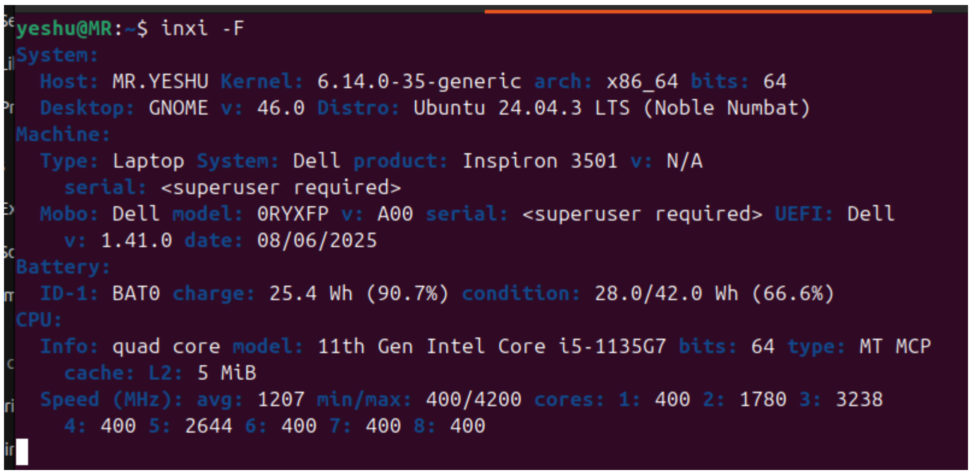
Ubuntu quickly became my go-to Linux distribution during this journey, not just because it is popular, but because it genuinely made the entire learning experience smoother and more enjoyable.

- **Beginner-Friendly Yet Powerful:** Even though I was new to Linux, Ubuntu never made me feel lost. Its clean interface and predictable behavior helped me learn faster.
- **LTS Means Peace of Mind:** The Long Term Support (LTS) version receives updates for several years, which means fewer risks, fewer crashes, and more focus on learning instead of fixing.
- **A Community That Never Sleeps:** Every error I faced already had someone discussing or solving it online. Forums, blogs, and tutorials make Ubuntu feel like a guided journey.
- **Software at Your Fingertips:** Whether I needed Git, Python, servers, or development tools, Ubuntu's APT package manager handled everything with a single command.
- **Stable Enough for Experiments:** Throughout installations, crashes, and reinstalls, Ubuntu proved stable and reliable—perfect for someone exploring open source hands-on.

{

1.3 Key Features of My Ubuntu 24.04 System

1. **Desktop Environment:** GNOME 46.0
2. **Kernel Version:** Linux 6.14.0-35-generic
3. **Package Manager:** APT 2.8.3 (amd64)
4. **Default Applications:** Brave Browser 142.1.84.141, LibreOffice 24.2.7.2, GNOME Core Utilities
5. **Snap Support:** snap 2.72+ubuntu24.04, snapd 2.72+ubuntu24.04, series 16, architecture amd64



```
yeshu@MR:~$ inxi -F
System:
  Host: MR.YESHU Kernel: 6.14.0-35-generic arch: x86_64 bits: 64
  Desktop: GNOME v: 46.0 Distro: Ubuntu 24.04.3 LTS (Noble Numbat)
Machine:
  Type: Laptop System: Dell product: Inspiron 3501 v: N/A
    serial: <superuser required>
  Mobo: Dell model: 0RYXFP v: A00 serial: <superuser required> UEFI: Dell
    v: 1.41.0 date: 08/06/2025
Battery:
  ID-1: BAT0 charge: 25.4 Wh (90.7%) condition: 28.0/42.0 Wh (66.6%)
CPU:
  Info: quad core model: 11th Gen Intel Core i5-1135G7 bits: 64 type: MT MCP
    cache: L2: 5 MiB
  Speed (MHz): avg: 1207 min/max: 400/4200 cores: 1: 400 2: 1780 3: 3238
    4: 400 5: 2644 6: 400 7: 400 8: 400
```

Figure 2: System Information

1.4 System Specifications

My system configuration:

- Operating System: Ubuntu 24.04 LTS
- Architecture: amd64
- Desktop Environment: GNOME 46.0
- Shell: Bash 5.1

1.5 Installation Process

The installation involved:

1. Downloaded Ubuntu 24.04 LTS ISO from the official website
2. Created a bootable USB using Rufus/Etcher
3. Configured dual boot with the existing OS
4. Installed essential development tools
5. Configured the system for open source development

2 Encryption and GPG

2.1 What is Encryption?

Encryption converts readable data (plaintext) into unreadable data (ciphertext) so that only authorized users can access it.

2.2 Types of Encryption

2.2.1 Symmetric Encryption

- Same key is used for both encryption and decryption.
- Examples: AES, DES.

2.2.2 Asymmetric Encryption

- Uses two keys: a public key (to encrypt) and a private key (to decrypt).
- Examples: RSA, ECC.

2.3 GNU Privacy Guard (GPG)

GPG is a free tool used for generating keys, encrypting files, and signing data using the OpenPGP standard.

2.4 Installing GPG

```
sudo apt update
sudo apt install gnupg
gpg --version
```

2.5 Generating GPG Keys

```
gpg --full-generate-key
```

Steps:

1. Choose RSA and RSA (default).
2. Select key size: 4096 bits.
3. Set validity: 1 year.
4. Enter your name and email.
5. Create a strong passphrase.

2.6 Listing Keys

```
gpg --list-keys
gpg --list-secret-keys
```

2.7 Exporting Your Public Key

```
gpg --armor --export your-email@example.com > public-key.asc
```

2.8 Encrypting a File

```
gpg --encrypt --recipient your-email@example.com document.txt
```

2.9 Decrypting a File

```
gpg --decrypt document.txt.gpg > document.txt
```

3 Sending Encrypted Email

3.1 Email Encryption Overview

Email encryption protects messages from being read by anyone except the intended recipient.

3.2 Tools Used

- **Thunderbird** – Email client with OpenPGP support.
- **GPG Keys** – Used to encrypt and sign emails.
- **ProtonMail** – Alternative encrypted email service.

3.3 Setting Up Thunderbird for GPG

3.3.1 Install Thunderbird

```
sudo apt install thunderbird
```

3.3.2 Enable OpenPGP

1. Open Thunderbird.
2. Go to **Account Settings**.
3. Select **End-to-End Encryption**.
4. Add your GPG key (or generate a new one).
5. Import the recipient's public key.

3.4 Sending an Encrypted Email

1. Compose a new email.
2. Click **Security**.
3. Choose **Require Encryption**.
4. (Optional) Add a digital signature.
5. Send the email.

3.5 Receiving an Encrypted Email

1. Message arrives encrypted.
2. Thunderbird detects encryption automatically.
3. Enter your GPG key passphrase.
4. Email is decrypted and displayed.

3.6 Best Practices

- Never share your private key.
- Use a strong passphrase.
- Backup your keys.
- Keep your key updated.
- Always verify public key fingerprints.

4 Open Source Tools for Daily Use

4.1 Why Use Open Source Tools?

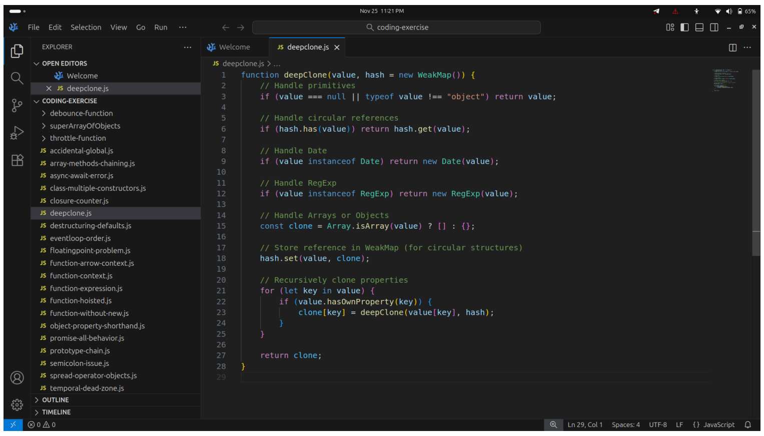
Open source tools are free, transparent, community-supported, and safe for student use. They help in learning, productivity, coding, writing, and even creative tasks without any restrictions or tracking.

4.2 Tool 1: VS Codium – Open Source Code Editor

Description: VS Codium is the telemetry-free, completely open source version of VS Code. It feels exactly the same as VS Code but without data tracking.

How Students Use It:

- Coding in C, C++, Python, Java, HTML, etc.
- Debugging programs inside the editor.
- Using extensions to speed up development.



```javascript
function deepClone(value, hash = new WeakMap()) {
    // Handle primitives
    if (value === null || typeof value !== "object") return value;

    // Handle circular references
    if (hash.has(value)) return hash.get(value);

    // Handle Date
    if (value instanceof Date) return new Date(value);

    // Handle RegExp
    if (value instanceof RegExp) return new RegExp(value);

    // Handle Arrays or Objects
    const clone = Array.isArray(value) ? [] : {};

    // Store reference in WeakMap (for circular structures)
    hash.set(value, clone);

    // Recursively clone properties
    for (let key in value) {
        if (value.hasOwnProperty(key)) {
            clone[key] = deepClone(value[key], hash);
        }
    }

    return clone;
}
```

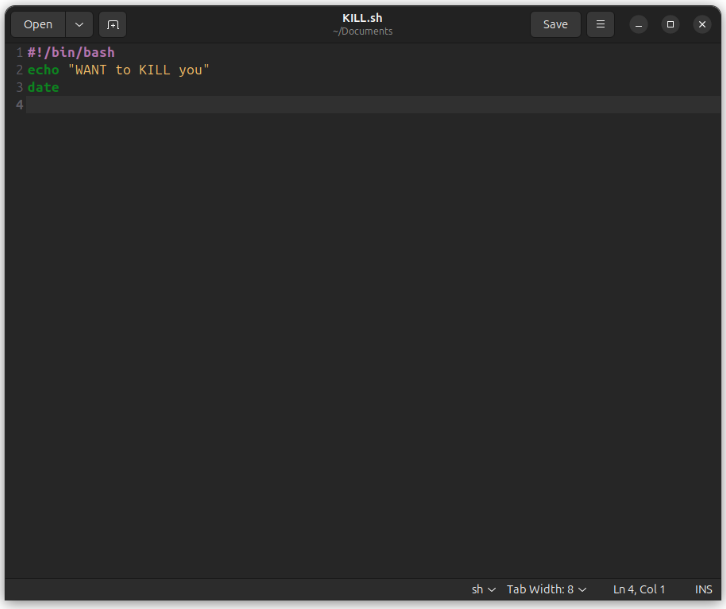
Figure 3: VS codium working interface

Key Features:

- Zero telemetry and tracking
- Works on all OS platforms
- Supports thousands of extensions

Shortcut Tip: `Ctrl + Shift + P` → Open command palette.

4.3 Tool 2: Gedit – Simple GNOME Text Editor



```bash
#!/bin/bash
echo "WANT to KILL you"
date
```

Figure 4: gedit working interface

Description: Gedit is a lightweight and beginner-friendly text editor perfect for quick editing and scripting.

How Students Use It:

- Writing small programs quickly.
- Editing configuration files.
- Making short notes or logs.

Key Features:

- Syntax highlighting for multiple languages
- Very lightweight and fast
- Simple UI for beginners

Shortcut Tip: `Ctrl + F` → Find text instantly.

4.4 Tool 3: LibreOffice – Full Office Suite

Description: LibreOffice is the open-source alternative to MS Office, including Writer, Calc, Impress, and Draw.

How Students Use It:

- Writing reports and assignments (Writer)
- Creating presentations (Impress)
- Maintaining tables and lab data (Calc)

Key Features:

- Offline, free, and open source
- Exports PDFs in one click
- Opens and edits MS Office files

Shortcut Tip: `Ctrl + Shift + P` → Quickly add superscript.

4.5 Tool 4: Thunderbird – Email Client with Encryption

Description: Thunderbird helps manage all email accounts in one simple interface and supports encryption with OpenPGP.

How Students Use It:

- Managing college and personal emails
- Sending GPG encrypted messages
- Organizing tasks and schedules

Key Features:

- Multi-account support
- Strong privacy protections
- Built-in calendar and tasks

Shortcut Tip: `Ctrl + N` → Compose a new email instantly.

4.6 Tool 5: Firefox – Privacy-Focused Web Browser

Description: Firefox is a fully open-source browser known for strong privacy, speed, and extensions.

How Students Use It:

- Research and online study
- Using extensions for productivity
- Privacy-safe browsing

Privacy Features:

- Enhanced Tracking Protection
- DNS-over-HTTPS support
- Container Tabs for separating accounts

Useful Tip: Enable **Strict Mode** in Settings → Privacy.

4.7 Tool 6: Shotcut – Open Source Video Editor

Description: Shotcut is a free video editor used for making project videos, presentations, and tutorials.

How Students Use It:

- Editing practical videos
- Creating presentations or animations
- Recording screen for submissions

Key Features:

- Multi-track video editing
- Built-in filters and effects
- No watermark, no subscription

Shortcut Tip: `S` → Split video clip at cursor.

4.8 Tool 7: GIMP – Open Source Image Editing

Description: GIMP is a powerful alternative to Photoshop, fully free and open-source.

How Students Use It:

- Editing images for projects
- Designing posters or diagrams
- Removing backgrounds or noise

Key Features:

- Layers, masks, and filters like Photoshop
- Supports plugins and custom brushes
- Works on all platforms

Shortcut Tip: `Ctrl + Shift + A` → Deselect selection.

5 Open Source License

5.1 License Used: MIT License

In my open source projects, I mostly use the **MIT License** because it is one of the simplest and most flexible licenses available for developers.

5.2 What is the MIT License?

The MIT License is a very easy-to-understand open source license. It allows anyone to:

- Use the software for personal or commercial purposes
- Modify and improve the code freely
- Share the software with anyone
- Include it in their own projects (even paid projects)
- Create their own versions based on the original work

5.3 MIT License Text (Example)

```
MIT License

Copyright (c) 2025 MR.YESHU

Permission is hereby granted, free of charge, to any person
obtaining a copy of this software and associated documentation
files (the "Software"), to deal in the Software without
restriction, including without limitation the rights to use,
copy, modify, merge, publish, distribute, sublicense, and/or
sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following
conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND...
```

5.4 Why I Prefer the MIT License

- **Very Simple:** The entire license is short, clear, and easy to understand even for beginners.
- **Highly Permissive:** Anyone can use, modify, and share your work without restrictions.
- **Good for Students:** Great for learning, collaboration, and contributing because people can build on your work easily.
- **Developer-Friendly:** Companies and developers prefer MIT because it does not force them to open-source their entire project.
- **No Legal Complexity:** No strict requirements or complicated legal terms — just use it and go.
- **Encourages Innovation:** Makes it easy for others to adopt, improve, and extend your software.

5.5 Other Common Open Source Licenses (Simple Overview)

5.5.1 GPL (GNU General Public License)

- Strong copyleft license.
- If you modify and distribute the software, your version must also be open source.
- Used by the Linux Kernel.

5.5.2 Apache License 2.0

- Permissive like MIT but includes patent protection.
- Suitable for large company-backed projects.
- Used by Apache, Kubernetes, and many cloud tools.

5.5.3 BSD License

- Extremely permissive, very similar to MIT.
- Allows almost complete freedom with minimal conditions.
- Used by FreeBSD and other OS-level projects.

6 Self-Hosted Server: Gladys Smart Home Assistant

6.1 What is Gladys?

Gladys is an open-source smart home assistant that allows users to control and automate devices in their home. It runs locally on your own system, which means:

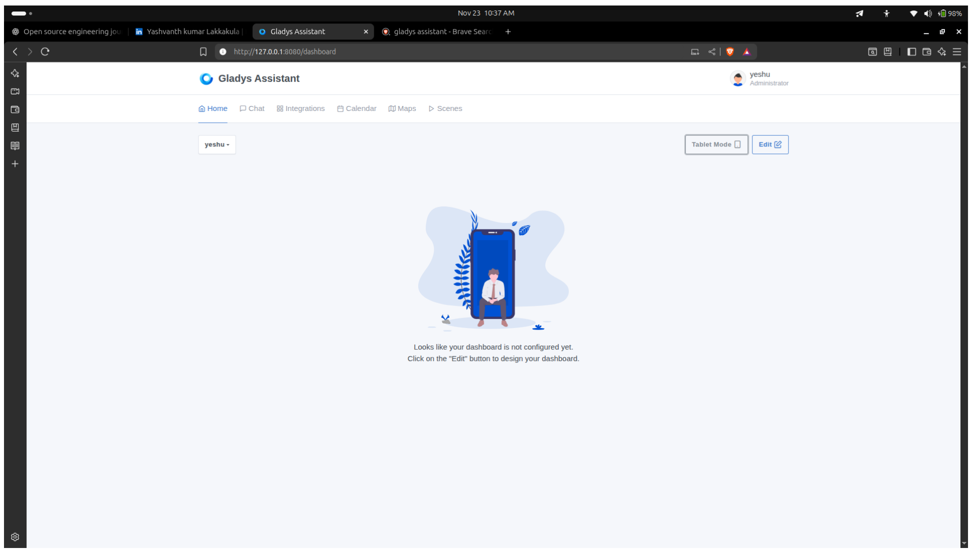
- Your data stays private
- No cloud dependency
- Full customization
- Ability to integrate any API you want

Gladys works similar to Alexa or Google Home, but instead of depending on cloud servers, it runs **locally on your machine** and is completely open source.

6.2 What Can Gladys Do?

With Gladys, you can:

- Control smart home devices (lights, fans, AC, TV)
- Monitor temperature, weather, humidity via APIs
- Receive live updates every second
- Create automation rules (e.g., turn on fan when temperature > 30°C)
- Add smart cleaners, washing machines, or any IoT device using "Link Devices"



Looks like your dashboard is not configured yet.
Click on the "Edit" button to design your dashboard.

Figure 5: Gladys Home Assistant Dashboard

6.3 Why I Self-Hosted Gladys?

1. **Data Privacy:** Everything runs locally; no data leaves my system.
2. **Customization:** I can add or modify features freely.
3. **API Integration:** I can connect weather, temperature, or custom APIs.
4. **Real-Time Updates:** Gladys responds instantly without any cloud delay.
5. **Learning Experience:** Helped me understand APIs, Node.js, and hosting.

6.4 System Requirements

Component	Minimum Requirement
Operating System	Ubuntu 24.04 LTS
Node.js Version	v16 or higher
npm Version	Latest recommended
RAM	1 GB
Storage	1 GB free
Network	Local network or Wi-Fi

Table 1: Gladys System Requirements

6.5 Installation Guide

This is the method I used to host Gladys using **Node.js and npm**.

6.5.1 Step 1: Install Node.js and npm

```
sudo apt update
sudo apt install nodejs npm -y
node -v
npm -v
```

6.5.2 Step 2: Clone Gladys Repository

```
git clone https://github.com/GladysAssistant/Gladys.git
cd Gladys
```

6.5.3 Step 3: Install Dependencies

```
npm install
```

6.5.4 Step 4: Start Gladys Server

```
npm start
```

Gladys will now run on:

```
http://localhost:1443
```

6.6 API Integration in Gladys

Gladys becomes powerful when we integrate APIs. For example:

- Weather API for live temperature updates

- Smart device API for turning devices on/off
- Sensor APIs for home automation

How API Integration Works:

1. Get a free Open Source API key (e.g., weather API).
2. Add it inside Gladys API configuration page.
3. Gladys fetches real-time data every second or as configured.
4. Devices/actions can be triggered based on API values.

Example: Temperature Automation

```
{
  "rule": "If temperature > 30 C ",
  "action": "Turn ON the fan",
  "source": "WeatherAPI"
}
```

6.7 Linking Devices

Gladys provides a built-in option called **"Link Devices"**. Using this, I connected:

- Smart Fan
- Smart Bulb
- Wi-Fi-enabled TV
- Smart AC Controller

Each device can be:

- Switched ON/OFF
- Automated based on rules
- Monitored continuously

6.8 Key Features of Gladys

- Real-time smart home control
- Multi-device integration
- Automation rules engine
- API-based sensor data
- Web dashboard for everything
- Full control through mobile or PC

6.9 Benefits of Self-Hosting Gladys

1. **Privacy:** No external servers involved.
2. **Zero Cost:** 100% free open-source platform.
3. **Full Control:** Edit, modify, or expand the system.
4. **Learning:** Helps understand Node.js, APIs, hosting, and IoT.
5. **Local Automation:** Works even without internet.

6.10 Backup and Maintenance

6.10.1 Backup Configuration

```
cp -r ~/.gladys backups/gladys-backup/
```

6.10.2 Update Gladys

```
git pull
npm install
npm restart
```

6.11 Useful Resources

- Official Website: `https://gladysassistant.com`
- GitHub Source Code: `https://github.com/GladysAssistant/Gladys`
- Documentation: `https://documentation.gladysassistant.com`

7 Open Source Contributions Overview

Open source contributions helped me understand real-world software development, documentation standards, community collaboration, and code quality. Below is a structured overview of all my contributions, divided into **Open** and **Merged** pull requests with detailed issue and solution descriptions.

7.1 Open Pull Requests

7.1.1 PR 1: Added Red-Black Tree Implementation in Python

Repository: bajajvinamr/HacktoberFest2020
PR Number: #1190
Status: Open
Issue: Repository lacked an efficient Red-Black Tree implementation.
Solution: Implemented Red-Black Tree with insertion logic, rotations, and balancing rules, following algorithmic standards.

7.1.2 PR 2: Fixed Typos and Improved Formatting in README.md

Repository: codecrafters-io/build-your-own-x
PR Number: #1621
Status: Open
Issue: README contained multiple formatting inconsistencies.
Solution: Corrected grammar, improved structure, and reformatted long paragraphs.

7.1.3 PR 3: Added "Build Your Own Alexa Assistant" Project

Repository: codecrafters-io/build-your-own-x
PR Number: #1620
Status: Open
Issue: Missing category for voice assistant projects.
Solution: Added a new project entry guiding users to build an Alexa-like assistant.

7.1.4 PR 4: Added New JavaScript Interview Question

Repository: sudheerj/javascript-interview-questions
PR Number: #335
Status: Open
Issue: Repository lacked coverage of certain JS concepts.
Solution: Added a well-structured interview question with explanation.

7.1.5 PR 5: Added deepclone.js Function

Repository: sudheerj/javascript-interview-questions
PR Number: #331
Status: Open
Issue: Deep cloning was not demonstrated in the repository.
Solution: Created a robust deep cloning function in JavaScript.

7.1.6 PR 6: Added Usage Example Section to README

Repository: TheAlgorithms/Python
PR Number: #13884
Status: Open
Issue: Many algorithms lacked usage examples.
Solution: Added usage examples to improve readability for new contributors.

7.1.7 PR 7: Added Splay Tree Implementation with Type Hints

Repository: TheAlgorithms/Python
PR Number: #13883
Status: Open
Issue: No type-hinted version of Splay Tree existed.
Solution: Added a clean Splay Tree implementation with Python type hints.

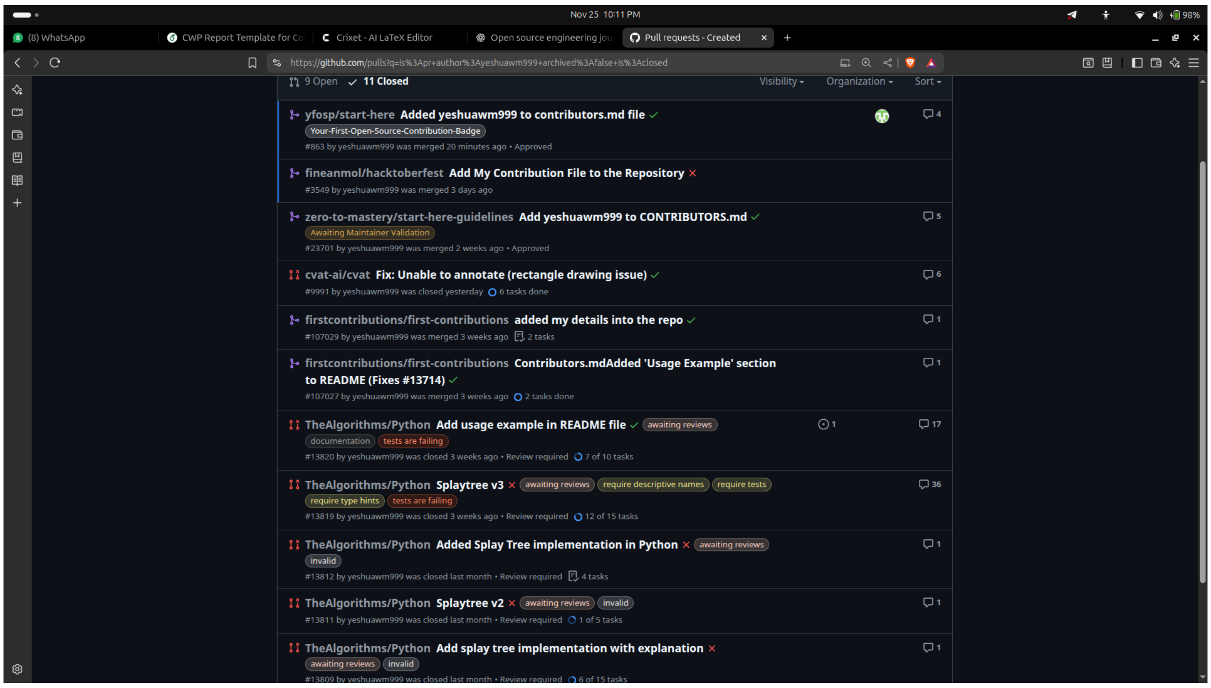
7.1.8 PR 8: Added Gladys Documentation in Telugu

Repository: KLGLUG/Y24OpenSourceEngineering
PR Number: #155
Status: Open
Issue: Regional users required onboarding documentation.
Solution: Wrote complete Telugu documentation for Gladys.

7.1.9 PR 9: Added Linux Installation Steps for Thonny

Repository: thonny/thonny
PR Number: #3716
Status: Open
Issue: Installation guide lacked Linux steps.
Solution: Added detailed installation steps for Ubuntu-based systems.

7.2 Merged Pull Requests


Nov 25 10:11 PM
98%
(8) WhatsApp
CWP Report Template for Co
Crixet - AI LaTeX Editor
Open source engineering jou
Pull requests - Created
https://github.com/pulls?q=is%3Apr+author%3Ayeshuawm999+archived%3Afalse+is%3Aclosed
9 Open
11 Closed
Visibility
Organization
Sort

yfosp/start-here Added yeshuawm999 to contributors.md file ✓ 4
Your-First-Open-Source-Contribution-Badge
#863 by yeshuawm999 was merged 20 minutes ago • Approved

fineanmol/hacktoberfest Add My Contribution File to the Repository ✗
#3549 by yeshuawm999 was merged 3 days ago

zero-to-mastery/start-here-guidelines Add yeshuawm999 to CONTRIBUTORS.md ✓ 5
Awaiting Maintainer Validation
#23701 by yeshuawm999 was merged 2 weeks ago • Approved

cvat-ai/cvat Fix: Unable to annotate (rectangle drawing issue) ✓ 6
#9991 by yeshuawm999 was closed yesterday 6 tasks done

firstcontributions/first-contributions added my details into the repo ✓ 1
#107029 by yeshuawm999 was merged 3 weeks ago 2 tasks

firstcontributions/first-contributions Contributors.mdAdded 'Usage Example' section to README (Fixes #13714) ✓ 1
#107027 by yeshuawm999 was merged 3 weeks ago 2 tasks done

TheAlgorithms/Python Add usage example in README file ✓ awaiting reviews 1 17
documentation tests are failing
#13820 by yeshuawm999 was closed 3 weeks ago • Review required 7 of 10 tasks

TheAlgorithms/Python Splaytree v3 ✗ awaiting reviews require descriptive names require tests 36
require type hints tests are failing
#13819 by yeshuawm999 was closed 3 weeks ago • Review required 12 of 15 tasks

TheAlgorithms/Python Added Splay Tree implementation in Python ✗ awaiting reviews 1
invalid
#13812 by yeshuawm999 was closed last month • Review required 4 tasks

TheAlgorithms/Python Splaytree v2 ✗ awaiting reviews invalid 1
#13811 by yeshuawm999 was closed last month • Review required 1 of 5 tasks

TheAlgorithms/Python Add splay tree implementation with explanation ✗ 1
awaiting reviews invalid
#13809 by yeshuawm999 was closed last month • Review required 6 of 15 tasks

Figure 6: Merged Pull Requests Overview

7.2.1 PR 10: Added Contributor Entry

Repository: yfosp/start-here
PR Number: #863
Status: Merged
Issue: Contributor list needed update.
Solution: Added my GitHub ID and contribution badge.

7.2.2 PR 11: Added Contribution File to Repository

Repository: fineanmol/hacktoberfest
PR Number: #3549
Status: Merged
Issue: Repository lacked a contributions record.
Solution: Added my contribution file in correct format.

7.2.3 PR 12: Added My Name to Contributors.md

Repository: zero-to-mastery/start-here-guidelines
PR Number: #23701
Status: Merged
Issue: Contributors list required updating.
Solution: Added my name with proper formatting.

7.2.4 PR 13: Added My Details in the Repo

Repository: firstcontributions/first-contributions
PR Number: #107029
Status: Merged
Issue: Contributor list update required.
Solution: Added my information to contributors.md.

7.2.5 PR 14: Added Usage Example Section to README

Repository: firstcontributions/first-contributions
PR Number: #107027
Status: Merged
Issue: README lacked usage examples.
Solution: Added a clear example section explaining contribution flow.

7.3 Contribution Summary

Metric	**Count**
Total Pull Requests	14
Open PRs	9
Merged PRs	5
Repositories Contributed	10+
Documentation Improvements	7
Code Contributions	5
Localization Work	1

Table 2: Open Source Contribution Statistics

7.4 Key Learnings from Contributions

1. **Git Workflow:** Mastered branching, committing, rebasing, and PR flow.

2. **Code Review:** Learned to handle reviewer feedback and improve quality.
3. **Documentation Writing:** Improved clarity, formatting, and readability.
4. **Collaboration:** Worked with global maintainers across multiple time zones.
5. **Algorithmic Thinking:** Implemented real data structures (RB Trees, Splay Trees).
6. **Localization Skills:** Wrote contributions in Telugu to help regional users.
7. **Professionalism:** Understood standards required for large open-source repos.

8 LinkedIn Posts

In addition to hands-on open source contributions, I have actively shared my learning journey and project updates on LinkedIn to inspire peers and document my progress. Below are the key posts related to my open-source and self-hosted server work.

8.1 Post 1: My First Open Source Contribution

Link: `https://www.linkedin.com/posts/yashvanth-kumar-lakkakula-134110352_opensource-github-developerjourney-activity-7398212740084932608-DFIX`

Summary: This was the announcement of my very first open source contribution. In this post, I shared how I took my initial step into the global developer community by raising my first PR, understanding version control workflows, and contributing to real-world projects.

Key Highlights:

- Overcame the fear of contributing to large repositories.
- Understood GitHub workflow: forking, branching, committing, pull requests.
- Gained confidence to solve documentation and code issues.
- Marked the beginning of my Open Source Engineering journey.

8.2 Post 2: My Open Source Blog – Success Journey in OSE

Link: `https://www.linkedin.com/posts/yashvanth-kumar-lakkakula-134110352_my-succes-journey-in-ose-activity-7398209427817930752-_AfA`

Summary: This blog post highlights my entire Open Source Engineering journey—from learning Linux, understanding collaborative development, to contributing to multiple global repositories. I documented the challenges, milestones, and the growth I experienced.

Key Highlights:

- Shared detailed steps I followed to get started in open source.
- Explained how I selected beginner-friendly issues.
- Reflected on skills gained: Git, documentation, debugging, community communication.
- Motivated beginners by showing that anyone can contribute with consistency.

8.3 Post 3: Self-Hosting Server Demonstration at Samyak Fest

Link: `https://www.linkedin.com/posts/yashvanth-kumar-lakkakula-134110352_my-succes-journey-in-ose-activity-7398209427817930752-_AfA`

Summary: In this post, I shared my experience presenting a fully self-hosted "Gladys — Smart Home Assistant Server" during Samyak, our university's technical fest. We demonstrated how a locally hosted smart assistant can handle home automation, real-time updates, and API-driven functionalities.

Key Highlights:

- Hosted Gladys Home Assistant locally using NPM on Ubuntu.
- Configured it with APIs such as temperature, weather, and smart device triggers.
- Explained smart home automation concepts to evaluators during the fest.
- Demonstrated API integration, automation flows, and server monitoring.
- Showcased the importance of open source tools in IoT and automation.

8.4 Impact of Sharing My Journey

- Helped fellow students understand the practical side of open source.
- Built connections with developers, maintainers, and tech communities.
- Increased confidence by presenting real projects and contributions.
- Inspired juniors to explore GitHub, open source, and self-hosting.

9 Conclusion

This report marks the completion of one of the most impactful journeys of my academic life — my exploration into the world of Open Source Engineering. Throughout this course, I did far more than just install software or write commands; I learned how powerful open-source communities are, how real-world contributions are made, and how even a student like me can build and deploy meaningful systems.

From setting up my Ubuntu development environment, understanding encryption and privacy tools, and learning to navigate the Linux terminal, to hosting my own smart home assistant (Gladys) on my system — every step taught me something new about how technology truly works behind the scenes.

My open-source contributions strengthened my confidence and helped me understand how global collaboration happens. Reviewing issues, fixing documentation, improving code, and submitting pull requests taught me that open source is not just about coding — it's about communication, patience, responsibility, and continuous learning.

Through this journey, I gained:

- Strong technical skills across Linux, Git, servers, and scripting

- Real understanding of collaborative development workflows
- Hands-on experience with version control, PR reviews, and community standards
- Deep appreciation for open-source culture and transparency
- Practical knowledge in hosting servers, using APIs, and automation
- Confidence to contribute to global projects and share my work publicly

Self-hosting the **Gladys Smart Home Assistant** was one of the most memorable achievements. It pushed me to explore APIs, networking, hosting, automation, and server configuration — transforming theory into a working real-world application. It helped me understand how modern IoT systems operate and how open-source solutions empower users with full control and flexibility.

Overall, this course has shaped me not just as a student, but as a developer ready to contribute, collaborate, and create. Open source has shown me that learning never stops, and every contribution — no matter how small — brings value to a global community. This journey has given me the confidence to continue exploring, experimenting, and building with an open mind and open tools.

In conclusion, Open Source Engineering did not just teach me technology — it taught me growth.