OPEN SOURCE ENGINEERING

Student ID: 2400030037

1 About Linux Distribution Used — Ubuntu

1.1 Introduction

Ubuntu is a popular and widely used Linux operating system based on Debian. It is completely free and open source. I selected Ubuntu for this project because it is very stable, user-friendly, and contains everything needed for programming, networking, and security tasks. Ubuntu is trusted by beginners as well as professionals, and it is commonly used in colleges, IT companies, and cloud servers.

1.2 Key Features of Ubuntu

- Long Term Support (LTS) provides stable updates for 5 years without issues.
- Easy software installation using the APT package manager.
- Clean and simple GNOME desktop environment that is easy to understand.
- Very large software repository and helpful global community support.
- Regular security patches protect the system from vulnerabilities.
- Supports both GUI (Graphical interface) and powerful Terminal commands.

1.3 Technical Advantages

- Ubuntu is based on the secure and fast Linux kernel, which improves performance.
- Works well on low-end laptops, high-performance computers, and servers.
- Supports a wide range of development tools such as compilers, interpreters, and IDEs.

- Excellent compatibility with encryption tools, privacy tools, Docker, and cloud platforms.
- Strong networking features, making it suitable for hosting servers and configuring services.

1.4 Usage in My Project

Ubuntu played an important role in this project. I used it for installing and using GPG encryption, hosting my self-hosted server, and setting up privacy tools from Prism-Break. I also performed Git commits and raised Pull Requests using Ubuntu while contributing to open-source projects. The package manager and terminal made every task smooth and error-free.

1.5 Conclusion

Ubuntu was a perfect choice for this project because it is stable, secure, and easy to work with. It supports development, self-hosting, and open-source tools without compatibility issues. Its simple interface and wide tool support helped me complete the project successfully and efficiently.

2 Encryption and GPG

2.1 Introduction

Encryption protects information by converting it into a secret format so that only the correct person can access it. In this project, I used **GPG (GNU Privacy Guard)**, which is a free and open-source tool for secure communication and data protection.

2.2 How GPG Works

GPG uses **asymmetric encryption** with two keys:

- **Public Key** – used to encrypt data
- **Private Key** – used to decrypt data

If someone encrypts a file or message using my public key, then only I can open it with my private key. This makes the communication safe and trustworthy.

2.3 Key Generation

A key pair is created using:

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

We enter our name, email, and passphrase during key creation. After that, GPG saves the keys securely on the system.

2.4 Digital Signatures

GPG can also add a digital signature to messages. This confirms that:

- The message is really sent by the owner of the key.
- The content has not been changed by anyone.

2.5 Advantages of GPG

- Protects files and emails with strong security.
- Works on Linux, Windows, and macOS.
- Free and open source, trusted worldwide.
- Does not need internet for encryption or decryption.

2.6 Conclusion

GPG helped make this project more secure by allowing safe communication and protected file storage. It is simple to use, powerful, and very useful for privacy-based projects.

3 Sending Encrypted Email

3.1 Introduction

An encrypted email keeps the message safe by making it readable only by the intended receiver. Even if someone else gets the email, they cannot understand it. In this project, I used **GPG encryption** to send emails securely.

3.2 Steps to Send an Encrypted Email

1. Generate your GPG key pair (public and private keys).
2. Get the receiver's **public key**.
3. Import their public key:

   ```
   gpg --import publickey.asc
   ```

4. Write the message and save it as `message.txt`.
5. Encrypt the message:

   ```
   gpg --encrypt --armor -r "Receiver Name" message.txt
   ```

6. A file named `message.txt.asc` is generated, which contains encrypted text.
7. Paste the encrypted text in an email or send the file as an attachment.

3.3 How the Receiver Reads the Email

The receiver decrypts the message using their **private key**:

```
gpg --decrypt message.txt.asc
```

GPG automatically checks the signature and confirms that the message is safe and original.

3.4 Benefits of Sending Encrypted Email

- Keeps personal and sensitive information private.
- Prevents email hacking and man-in-the-middle attacks.
- Confirms the real identity of the sender through digital signatures.

3.5 Conclusion

Sending encrypted email is a safe and professional way of communication. GPG made the process simple and secure by protecting the message and confirming the identity of the sender and receiver.

4 Privacy Tools from Prism-Break.org

4.1 Introduction

Prism-Break.org recommends free, open-source and privacy-focused software that protects users from data collection and tracking. These tools give complete control to the user instead of companies that store personal information. For this project, I selected five privacy tools that are fully compatible with Ubuntu and helped ensure private browsing, communication, email usage, operating system security, and navigation.

4.2 Privacy Tools Used

- **Thunderbird** — Thunderbird is an open-source desktop email client developed by Mozilla. It allows users to manage multiple email accounts securely and does not rely on web services that track user activity. Thunderbird supports PGP/GPG encryption, digital signatures, spam filtering, add-ons, and offline access to emails. Email data is stored locally on the system, so personal communication stays private and under user control at all times.

- **Debian** — Debian is a stable and privacy-focused Linux distribution that avoids telemetry and hidden background data collection. It follows strict open-source policies and allows users to install only the software they choose, reducing unwanted services that send data to external servers. Debian is the foundation for many other secure operating systems, including Ubuntu, Kali Linux and Tails OS. It is well-known for its long-term stability, powerful package repository and reliable community-verified security updates.

- **Firefox** — Firefox is an open-source web browser designed for safe internet browsing. It blocks online trackers, invisible ads, fingerprinting scripts and third-party cookies by default. Firefox does not build advertising profiles and does not store browsing history to target the user. It supports privacy add-ons like uBlock Origin, NoScript, Privacy Badger

and more. Firefox also includes enhanced tracking protection and encrypted DNS to stop internet providers from monitoring browsing habits.

- **OpenStreetMap** — OpenStreetMap is a collaborative open-source map project that provides global navigation data without collecting user identity or location history. Users can view maps, search locations and download offline maps without creating an account or being tracked. The data is updated by a global community of contributors to ensure accurate and constantly improving maps. Many privacy-focused GPS and routing apps use OpenStreetMap because it offers helpful navigation without giving up personal data.

- **Signal** — Signal is a secure messaging and calling application that uses end-to-end encryption for messages, voice calls, video calls and file sharing. Only the sender and receiver can access the communication because Signal does not store chat history, contact lists or metadata on its servers. The app shows no advertisements, no trackers and does not share data with third parties. Signal is widely trusted by privacy enthusiasts, researchers, journalists and security professionals for safe and private communication on both computers and mobile devices.

5 Open Source Licence Used — MIT License

5.1 Introduction

Open-source licenses define how software can be used, modified, and shared by others. They protect the original developer and also allow the community to openly contribute to the project. For this project, I used the **MIT License**, which is one of the most popular and widely used open-source licenses in the world.

5.2 About the MIT License

The MIT License is a very simple and flexible software license that allows anyone to use the software freely. People are permitted to copy it, modify it, distribute it, and even use it in commercial products. This freedom encourages teamwork and open development while still giving proper credit to the original author. The license is short, easy to understand, and does not have complicated legal restrictions.

5.3 Why I Chose the MIT License

I selected the MIT License for this project because it allows maximum freedom for users and developers. Anyone can use the software without needing permission, and they can also improve or change it based on their needs. This supports collaboration and continuous enhancement of the software. The license also protects me from legal responsibility if the software is used incorrectly, which makes it safe for sharing publicly.

5.4 Main Points of the MIT License

- Anyone can use, copy, modify, or distribute the software without restrictions.
- Proper credit to the original author must be included in all copies.
- The software is provided "as is," without any warranty or guarantee.

- The author is not responsible for any damage caused while using the software.

- The software can be used for personal, educational, research, or commercial purposes.

5.5 MIT License Header Used in My Project

Below is the license header I added to my project files:

5.6 Conclusion

The MIT License is simple, permissive, and highly developer-friendly. It gives freedom to users and contributors while still keeping the author's credit and legal protection. This is why the MIT License was the perfect choice for my open-source project.

6 Self-Hosted Server — Kutt URL Shortener

6.1 Introduction

A self-hosted server allows users to run a service on their own machine instead of depending on external cloud platforms. This gives full control over data, performance, and privacy. For this project, I self-hosted **Kutt**, an open-source URL shortener that lets users create short links with custom names, expiration dates, link statistics, and API access. Kutt can be hosted on a personal computer, VPS, or Docker container and provides a clean and modern dashboard.

6.2 About Kutt

Kutt is a simple and powerful web application that converts long URLs into short, easy-to-share links. It supports user accounts, password-protected links, link analytics, and an admin panel to manage users. Kutt provides an API that can be integrated into applications to automatically generate short URLs. It is open source, and because it is self-hosted, no data is shared with external companies — users fully control link information, analytics, and databases.

6.3 Installation on Ubuntu

I installed Kutt on Ubuntu using Docker and PostgreSQL. Below are the steps I followed:

1. Updated system packages:

   ```
   sudo apt update && sudo apt upgrade
   ```

2. Installed Docker and Docker Compose.

3. Created an `.env` file and configured database, JWT secret key, and site settings.

4. Downloaded the Kutt Docker configuration from the official repository.

5. Started the server using:

```
docker compose up -d
```

6. Accessed Kutt through the browser using the local IP address.

Localized (Translated) Document — Telugu

Kutt URL Shortener వివరణ & ఇన్‌స్టాలేషన్

Kutt ఒక తేలికైన, ఓపెన్-సోర్స్ URL షార్టనర్. ఇది డొమెయిన్‌పై చిన్న URLలు సృష్టించి, క్లిక్ ట్రాకింగ్, లింక్ సెక్యూరిటీ మరియు ప్రైవేట్/పబ్లిక్ లింక్‌లను నిర్వహించడానికి వెబ్ UI/API ని అందిస్తుంది. దీన్ని స్వయంగా హోస్ట్ చేయడం వల్ల (self-host) మీ డేటా పూర్తిగా మీ నియంత్రణలో ఉంటుంది.

ప్రధాన ఫీచర్లు

- షార్ట్ URL సృష్టించడం
- క్లిక్ అనలిటిక్స్‌ను చూడటం
- పాస్‌వర్డ్ రక్షిత లింకులు
- REST API ఇన్టిగ్రేషన్

అవసరమైనవి (Prerequisites)

- Node.js (LTS)
- PostgreSQL డేటాబేస్
- Redis (ఐచ్చికం క్యాషింగ్ మరియు రేట్ లిమిటింగ్ కోసం)
- సర్వర్ యాక్సెస్ లేదా Docker సపోర్ట్

ఇన్‌స్టాలేషన్ Docker ద్వారా

1. రిపోజిటరీ క్లోన్ చేయండి:

   ```
   git clone https://github.com/thedevs-network/kutt.git
   cd kutt
   ```

2. .env ఫైల్ సెట్ చేయండి:

   ```
   cp .env.example .env
   # .env   DATABASE_URL, REDIS_URL, SECRET_KEY
   ```

3. సర్వర్‌ను స్టార్ట్ చేయండి:

   ```
   docker compose up -d
   ```

ఇన్‌స్టాలేషన్ Node.js (Manual) ద్వారా

1. డిపెండెన్సీలు ఇన్‌స్టాల్ చేయండి:

   ```
   git clone https://github.com/thedevs-network/kutt.git
   cd kutt
   npm install
   ```

2. .env ఫైల్ సెట్ చేయండి (DATABASE_URL, SECRET_KEY, BASE_URL వంటి విలువలు).

3. మైగ్రేషన్‌ను అమలు చేసి, తరువాత సర్వర్‌ను ప్రారంభించండి:

   ```
   npm start
   ```

కాన్ఫిగరేషన్ ముఖ్యాంశాలు

- `BASE_URL` షార్ట్ డొమెయిన్ చిరునామా.
- `DATABASE_URL` PostgreSQL కనెక్షన్ స్ట్రింగ్.
- `SECRET_KEY` JWT / సెషన్ సీక్రెట్ కీ.

ప్రాజెక్ట్ వీడియో

ప్రాజెక్ట్ డెమో / ట్యుటోరియల్ వీడియో లింక్ క్రింద ఇవ్వబడి ఉంది:
Google Drive Video URL:

`https://drive.google.com/file/d/1LPELK6k4vUea6a9amzzwQiRASpVqrvnm/view?usp=sharing`

EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

Kutt - Open Source Engineering

Kutt is a powerful, free, open-source URL shortoning designed for usigned who users who need a realiable and customizaize solution to manage their links.

Features

- Open-source and self-hostable
- Free custom domain support
- Password-protected links
- Zero Cost for Basic Use
- Privacy-friendly analytics

License: MIT license

M Gnana Karthik - 2400030033 P Harsha Sai - 2400030037

7 Open Source Contributions — Pull Requests

7.1 Introduction

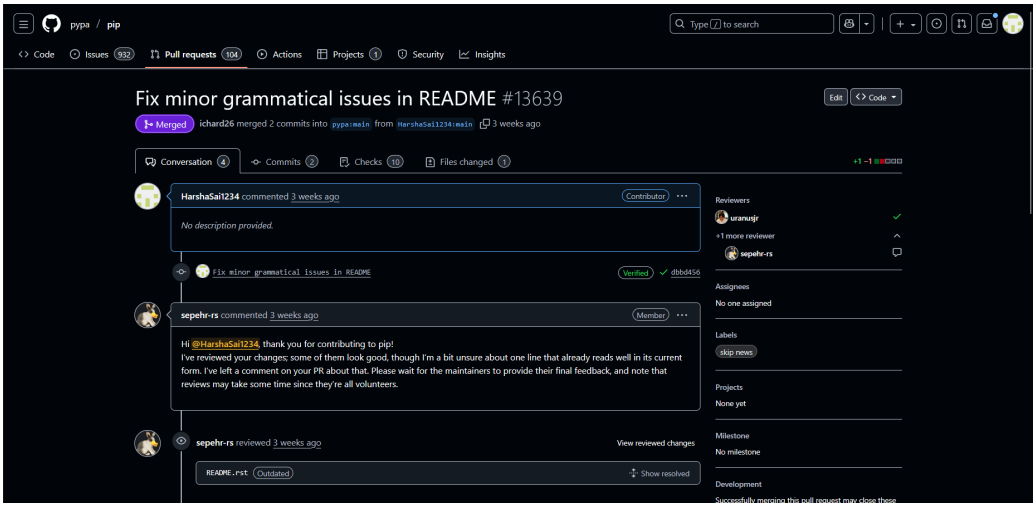
As part of this evaluation, I contributed to open-source repositories by identifying issues, fixing bugs, writing documentation, and submitting Pull Requests on GitHub. Each PR was developed on Ubuntu using Git commands and followed proper contribution guidelines of the repository.

7.2 Summary

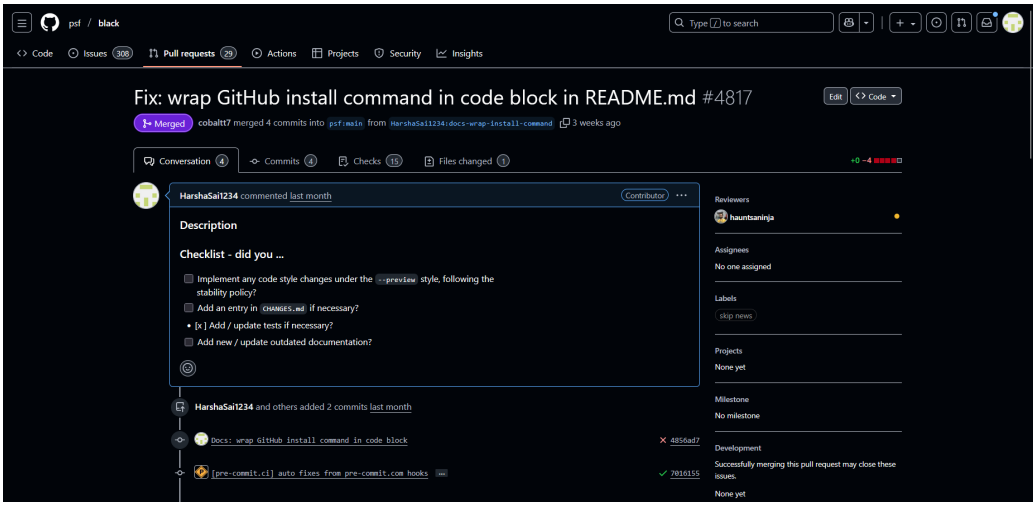
A total of **6 Pull Requests (PRs)** were successfully merged into open-source projects. Details of each PR along with screenshots are provided below.

7.3 Details of Pull Requests

PR 1 — Repository: *pip*
Issue Solved: *Fix minor grammatical issues in README*

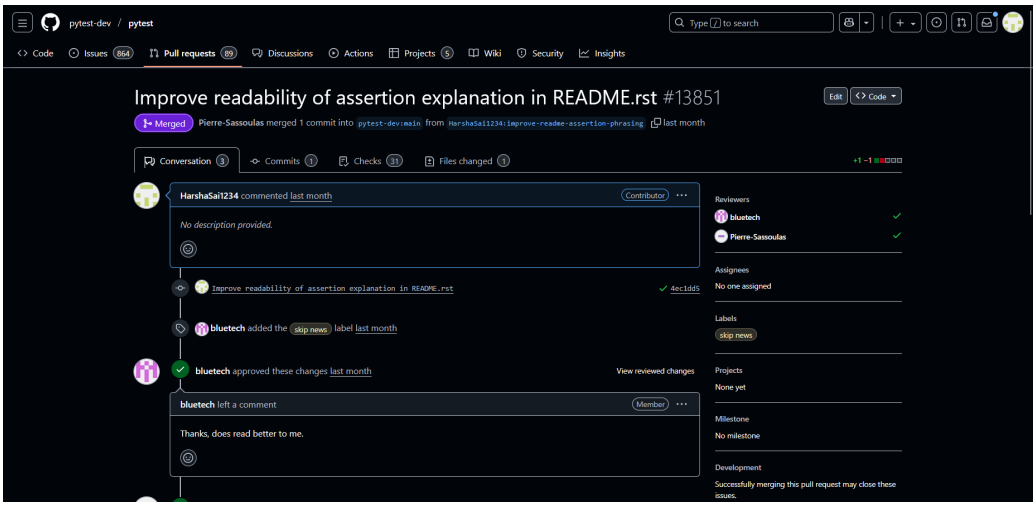


PR 2 — Repository: *black*
Issue Solved: *Fix: wrap GitHub install command in code block in README.md*



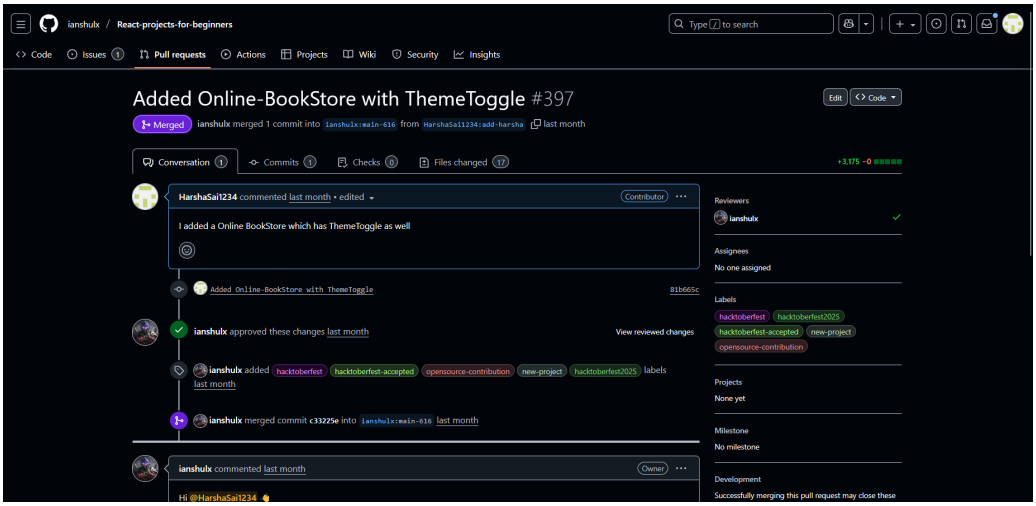
PR 3 — Repository: *pytest*

Issue Solved: *Improve readability of assertion explanation in README.rst*



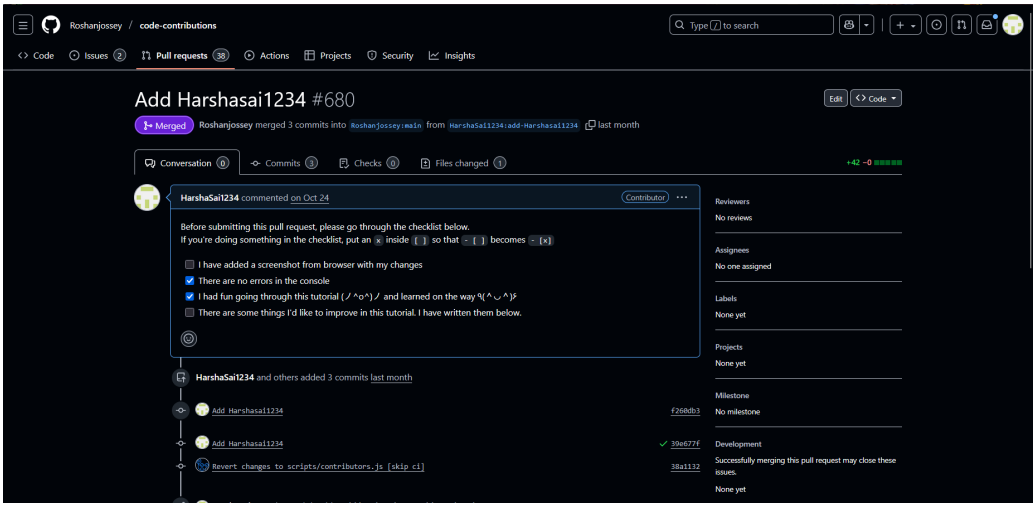
PR 4 — Repository: *React-projects-for-beginners*

Issue Solved: *Added Online-BookStore with ThemeToggle*



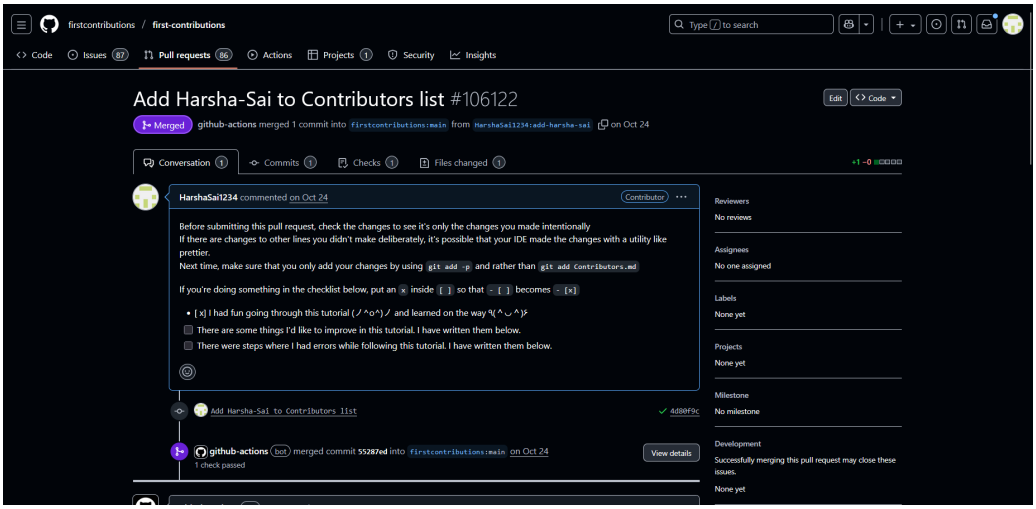
PR 5 — Repository: *code-contributions*

Issue Solved: *Add Harshasai1234-added c code for a problem from leetcode*



PR 6 — Repository: *first-contributions*

Issue Solved: *Add Harsha-Sai to Contributors list-added my name*


firstcontributions / first-contributions
Type / to search
Code
Issues 87
Pull requests 86
Actions
Projects 1
Security
Insights
Add Harsha-Sai to Contributors list #106122
Edit
Code
Merged
github-actions merged 1 commit into firstcontributions:main from HarshaSai1234:add-harsha-sai on Oct 24
Conversation 1
Commits 1
Checks 1
Files changed 1
+1 -0
HarshaSai1234 commented on Oct 24
Contributor
Before submitting this pull request, check the changes to see it's only the changes you made intentionally
If there are changes to other lines you didn't make deliberately, it's possible that your IDE made the changes with a utility like prettier.
Next time, make sure that you only add your changes by using git add -p and rather than git add Contributors.md
If you're doing something in the checklist below, put an x inside [ ] so that - [ ] becomes - [x]
• [x] I had fun going through this tutorial (ノ^o^)ノ and learned on the way ٩(^‿^)۶
There are some things I'd like to improve in this tutorial. I have written them below.
There were steps where I had errors while following this tutorial. I have written them below.
Add Harsha-Sai to Contributors list
4d80f9c
github-actions bot merged commit 55287ed into firstcontributions:main on Oct 24
1 check passed
View details
Reviewers
No reviews
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Successfully merging this pull request may close these issues.
None yet

8 LinkedIn Posts Related to the Project

8.1 Introduction

To share the progress and promote open-source learning, I posted updates on LinkedIn throughout the project. These posts helped spread awareness about self-hosting, contributing to open source, and writing technical blogs. Below are the three project-related posts published on LinkedIn.

Post 1 — Self-Hosting (Kutt URL Shortener)

Link: https://www.linkedin.com/posts/harsha-sai-polnati-3281b6302_klu-projectexpo-selfhosted-activity-7382473197750919168-aCh2?utm_source=share&utm_medium=member_desktop&rcm=ACoAAE1FkVABBX1AOtAcDb8ud36aGOIIsMJV8OQ

Post 2 — Pull Request Merged

Link: https://www.linkedin.com/posts/harsha-sai-polnati-3281b6302_hacktoberfest-opensource-opensourceengineering-activity-7392411600772812800-WtZF?utm_source=share&utm_medium=member_desktop&rcm=ACoAAE1FkVABBX1AOtAcDb8ud36aGOIIsMJV8OQ

Post 3 — Blog Publication

Link: https://www.linkedin.com/posts/harsha-sai-polnati-3281b6302_klu-hte-opensource-activity-7398360950283583488-nipu?utm_source=share&utm_medium=member_desktop&rcm=ACoAAE1FkVABBX1AOtAcDb8ud36aGOIIsMJV8OQ