EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

OPEN SOURCE ENGINEERING

Student ID: 2400031833

Semester: Odd

Academic Year: 2025-2026

Course Code: 24CS02EF

Under the guidance of

Dr. Sripath Roy Koganti

1 Understanding the Core Ubuntu Linux Distribution

1.0.1 1. Overview and Philosophy

Ubuntu is a powerful, free, and open-source operating system built upon the stable foundation of Debian Linux. It stands as the world's most popular Linux distribution for desktop use, successfully blending cutting-edge features with unparalleled user-friendliness. Developed and maintained by Canonical Ltd., Ubuntu's guiding principle is ”Linux for human beings.” This philosophy drives its commitment to accessibility, stability, and providing an intuitive computing experience for everyone, from novice users to seasoned developers.

1.0.2 2. The Desktop Experience (GNOME)

The standard Ubuntu desktop utilizes the **GNOME** desktop environment, which presents a modern, clean, and highly efficient graphical interface. The key design elements include a permanent dock (launcher) on the left side for quick access to essential applications, and the **Activities Overview**. This view, easily accessed by pressing the Super (Windows) key, provides a centralized hub for managing all open windows, workspaces, and system-wide searching. This streamlined workflow makes Ubuntu feel contemporary and ensures high productivity. Furthermore, Ubuntu is recognized for its strong, out-of-the-box hardware detection and compatibility, simplifying the setup process for most users.

1.0.3 3. Software Management and Packaging

Ubuntu employs a robust dual-system for software management. The traditional and reliable **Advanced Packaging Tool (APT)** manages **DEB** packages, handling core system utilities and standard applications sourced from official repositories. Complementing this is the use of **Snaps**, a modern, containerized package format pioneered by Canonical. Snaps bundle an application with all its required dependencies, guaranteeing consistent performance across different Ubuntu versions. Crucially, Snaps run in a **sandboxed** environment, isolating them from the rest of the operating system to significantly enhance overall application security. This flexibility ensures users have access to a vast, up-to-date, and secure software library.

2 Encryption and GPG

2.1 Types of Encryption in Ubuntu

Ubuntu offers two primary approaches to encryption: **Full Disk Encryption (FDE)** and **File/Directory Encryption**.

2.1.1 1. Full Disk Encryption (FDE)

- **What it is:** FDE encrypts the entire hard drive or a large partition, including the operating system files, swap space, and user directories.
- **How it works:** Ubuntu uses **LUKS** (Linux Unified Key Setup) for FDE. When the system boots, you are prompted for a **passphrase**. If correct, LUKS decrypts the entire drive, and the decryption process runs transparently in the background while the system is in use.
- **Purpose:** The primary defense against data loss due to **theft** or **physical access** to the computer when it is turned off. If someone steals the hard drive, the data is useless without the LUKS passphrase.
- **Implementation:** FDE is typically enabled during the Ubuntu installation process by selecting the "Encrypt the new Ubuntu installation" option. It's much more difficult to enable after installation.

2.1.2 2. File and Directory Encryption

- **What it is:** This method encrypts specific files, directories, or messages, offering granular control over which data is protected.
- **Tools:**
  - **GPG (GNU Privacy Guard):** The standard, used for encrypting individual files and especially for secure communication using **public-key cryptography**.
  - **eCryptfs (older):** Previously used for encrypting the user's Home directory, but has been largely phased out for FDE.

2.2 GPG (GNU Privacy Guard) Explained

GPG is the GNU implementation of the **OpenPGP** standard (originally Pretty Good Privacy - PGP). It is essential for protecting individual files and ensuring secure, authenticated communication.

2.2.1 1. Core GPG Concepts

GPG relies on **asymmetric cryptography**, which uses a pair of mathematically linked keys:

- **Public Key:** This key is shared with everyone. It can be used to **encrypt** a message that only you can read, or to **verify** a signature you created.
- **Private (Secret) Key:** This key is kept **secret** and is protected by a strong passphrase. It is used to **decrypt** messages sent to you, or to **digitally sign** files to prove they came from you.

2.2.2 2. Basic GPG Command-Line Usage

GPG is usually pre-installed on Ubuntu and is primarily used through the command line (Terminal).

A. Generating a Key Pair The first step is to create your public and private key pair:
Bash

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

You will be prompted to select the key type (RSA and RSA is common), keysize (4096 is recommended), expiration date, and your Real Name, Email, and a strong **passphrase** to protect your private key.

B. Encrypting a File for Yourself (Symmetric Encryption) To quickly encrypt a file using a single passphrase (like a standard password), use symmetric encryption:
Bash

```
gpg -c myfile.txt
```

This command will prompt you for a passphrase and create an encrypted file named `myfile.txt.gpg`.

C. Encrypting a File for Someone Else (Asymmetric Encryption) To securely send a file, you must use the recipient's **Public Key** (which you must have previously imported into your keyring with `gpg --import`):
Bash

```
gpg --encrypt --recipient "recipient@example.com" mysecretfile.doc
```

This creates `mysecretfile.doc.gpg`. Only the recipient, who holds the corresponding Private Key, can decrypt it.

D. Decrypting a File To decrypt a file that was encrypted for you:

Bash

```
gpg --decrypt mysecretfile.doc.gpg
```

You will be prompted for the passphrase that protects your Private Key. You can use the `--output` option to specify the decrypted

3 Sending Encrypted Email

3.1 Prerequisite: Setting Up GPG

Before you can send and receive encrypted mail, both you and your recipient must have GPG keys set up and exchanged:

1. **Generate Keys:** Both parties must have generated a public/private key pair using GPG (as discussed previously, using `gpg --full-generate-key`).
2. **Exchange Public Keys:** You need the recipient's **Public Key**, and they need your Public Key. You can exchange these by:
   - **Exporting** the key: `gpg --armor --export 'Recipient Name' > recipient_key.asc` and sending the `.asc` file.
   - **Uploading** the key to a public key server.
3. **Import Key:** You must import the recipient's key into your GPG keyring: `gpg --import recipient_key.asc`.

3.2 Sending the Encrypted Email

The most common and user-friendly way to send GPG-encrypted emails on Ubuntu is by using **Mozilla Thunderbird** with the **Enigmail** add-on (or its built-in equivalent in modern versions of Thunderbird).

3.2.1 1. Compose the Message

- **Open Thunderbird** and start composing a new email.
- Write your message as usual.

3.2.2 2. Encryption and Signing

You will use the GPG function built into the mail client to perform two critical steps:

1. **Encryption:** You must encrypt the email using the **recipient's Public Key**. Only their corresponding **Private Key** can decrypt it. If you have multiple recipients, you must encrypt the message using the Public Key of *every single recipient*.
2. **Digital Signature:** You **sign** the email using **your Private Key**. This allows the recipient to verify that the email truly came from you and has not been tampered with in transit.

In Thunderbird, this is typically done by clicking a dedicated **OpenPGP or Security** menu or button within the compose window and ensuring both the **"Encrypt"** and **"Sign"** options are checked.

3.2.3 3. Verification and Sending

- The client will check that you have the required **Public Key** for the recipient(s). If a key is missing, it will warn you.
- When you click **Send**, Thunderbird uses GPG to encrypt the message body and attach your digital signature before transmitting the scrambled data.

3.2.4 4. Recipient's Experience (Decryption)

1. The recipient receives the scrambled email.
2. Their email client automatically uses their **Private Key** (protected by their passphrase) to decrypt the message contents, revealing the original text.
3. Their client simultaneously uses your **Public Key** to verify the digital signature, confirming the email's authenticity.

4 Privacy Tools From Prism Break

4.0.1 1. Tor Browser (Web Browsers / Anonymizing Networks)

- **What it is:** A web browser built on Firefox that routes your internet traffic through the Tor network, a volunteer-operated network of relays.
- **Privacy Focus:** Provides **strong anonymity** by obscuring your IP address and location from the websites you visit. It also includes anti-fingerprinting measures.

- **PRISM Break Note:** PRISM Break strongly recommends using Tor Browser for all web surfing when maximum anonymity is required.

4.0.2 2. Debian (Operating Systems)

- **What it is:** A popular and highly ethical GNU/Linux distribution known for its strict adherence to Free Software principles and ethical manifesto.
- **Privacy Focus:** Unlike proprietary operating systems like Windows and macOS (which PRISM Break generally avoids), Debian is fully open-source, allowing for audits. It has a long tradition of software freedom and transparency.
- **PRISM Break Note:** It's recommended as a top GNU/Linux choice for users transitioning from proprietary systems, highlighting its commitment to free software and its stable nature.

4.0.3 3. Thunderbird (Email Clients)

- **What it is:** A free, open-source, and cross-platform email client developed by Mozilla.
- **Privacy Focus:** Thunderbird is the top choice for desktop email due to its open-source nature and its long-standing **native support for OpenPGP** (GPG) encryption and digital signatures. This allows users to easily encrypt and authenticate their emails end-to-end.
- **PRISM Break Note:** It is highly recommended for securely managing email with built-in PGP features.

4.0.4 4. KeePassXC (Password Managers)

- **What it is:** A free, open-source, and cross-platform password manager.
- **Privacy Focus:** It stores all your passwords in a single, highly encrypted database file that is stored **locally** on your device, giving you total control over your sensitive data. It does not rely on a cloud service.
- **PRISM Break Note:** It is preferred for its strong encryption, open-source license, and local-only storage, minimizing exposure to third-party services.

4.0.5 5. Firefox (Web Browsers)

- **What it is:** A fast, flexible, and secure web browser developed by the non-profit Mozilla Foundation.

- **Privacy Focus:** Firefox is open-source and provides extensive privacy controls, including enhanced tracking protection (ETP), container technology, and a robust add-on ecosystem for further hardening security (like uBlock Origin).
- **PRISM Break Note:** While Tor Browser is for anonymity, Firefox is the recommended alternative for general web use when a site doesn't work well with Tor, provided the user configures its settings and replaces the default search engine with a privacy-focused one.

5 Open Source License

5.1 License I Used

For my open-source work, I used the GNU General Public License version 3 (GPL-3.0). This is a widely used license that allows anyone to use, modify, and distribute software freely, as long as the same license is applied to any derivative works. One of the main ideas behind GPL-3.0 is to ensure that the software remains free and open for everyone, and any improvements made by others are also shared openly.

By using this license, I learned how open-source licensing works and why it is important for protecting both the developer's rights and the community's freedom. It ensures transparency, encourages collaboration, and allows other people to contribute to the project while keeping the software free and accessible. This experience helped me understand the legal and ethical side of open-source development.

6 Self Hosted Server

6.1 About

Passky is a simple and modern open-source password manager designed for secure self-hosting. It provides a clean web interface along with browser extensions, Android, and desktop apps. Our self-hosted setup runs on PHP and Docker, giving full control over all stored credentials. Data stays entirely on your own server, ensuring privacy and complete ownership. Passky is licensed under GPL-3.0, allowing transparency, customization, and community-driven development..

6.1.1 Key Features

Key Features

- **Self-Hosted & Private** – All passwords are stored on your own server, ensuring full data ownership.
- **Multi-Platform Access** – Available via web app, browser extensions, Android app, and desktop client.
- **Modern & Simple UI** – Clean, user-friendly design for easy password management.
- **Docker-Based Deployment** – Quick installation and maintenance using Docker containers.
- **End-to-End Security** – Strong encryption ensures only authorized users can access stored passwords.
- **Open Source (GPL-3.0)** – Transparent, customizable, and community-driven development.
- **Auto-Fill & Auto-Save** – Browser extensions provide seamless login autofill and password saving.
- **Fast Syncing** – Synchronizes passwords efficiently across all connected devices.
- **Backup Friendly** – Easy backup options to prevent data loss.
- **Lightweight & Efficient** – Minimal resource usage, suitable for personal and small servers.

Installation Process

The following steps describe how to install and self-host the Passky password manager using Docker. This method ensures a simple, fast, and reliable deployment suitable for personal and small-server environments.

1. Prerequisites

Before beginning the installation, ensure the following are installed on your system:

- Docker Engine (latest version)
- Docker Compose
- Git

- A server or machine running Linux, macOS, or Windows

Once these are installed, you are ready to deploy the Passky server.

2. Clone the Repository

Download the official Passky Server source code from GitHub:

```
git clone https://github.com/Rabbit-Company/Passky-Server.git
cd Passky-Server
```

3. Configure Environment Variables

Passky uses a `.env` file to manage environment settings. Copy the example configuration and modify it:

```
cp .env.example .env
```

Inside the `.env` file, you can configure:

- Database username and password
- API settings
- Allowed domain for requests
- Server port number

This file controls how your server behaves and secures your database connection.

4. Start Services Using Docker Compose

To build and launch the Passky server along with its database, run:

```
docker compose up -d
```

Docker will:

- Download required images
- Build containers for the API and database
- Start all services in the background (detached mode)

You can verify running containers with:

```
docker ps
```

5. Initialize the Database

If the database container is starting for the first time, Passky automatically sets up required tables. However, you may also manually restart containers if needed:

```
docker compose restart
```

6. Access the Passky Web Interface

Once all services are running, open your browser and navigate to:

- `http://localhost` (for local machine)
- `http://your-server-ip` (for remote server)

You should now see the Passky login/registration interface.

7. Create Your Account

Register a new user to begin using Passky. Your credentials will be securely stored in your self-hosted database.

8. Connect Clients and Extensions

Passky supports multiple platforms. After your server is live, install:

- Browser extension (Chrome/Firefox)
- Android app
- Desktop client

During setup, enter your server URL so all devices sync with your self-hosted instance.

9. Backup and Maintenance (Recommended)

To prevent data loss, regularly back up:

- Database container volumes
- Environment configuration files
- Docker Compose YAML files

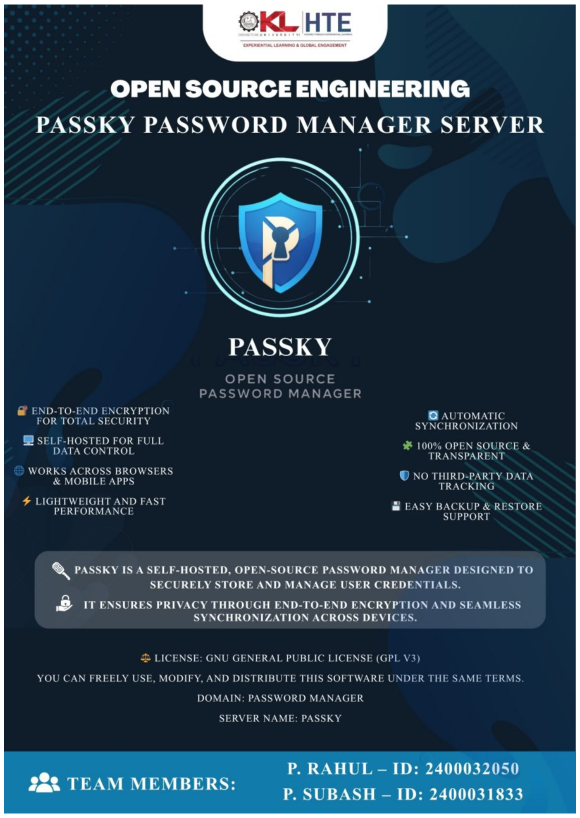
You can update the server anytime using:

```
docker compose pull
docker compose up -d
```

This ensures you always run the latest secure version.

Passky Resources

Translated Document



OPEN SOURCE ENGINEERING

PASSKY PASSWORD MANAGER SERVER

PASSKY

OPEN SOURCE PASSWORD MANAGER

- END-TO-END ENCRYPTION FOR TOTAL SECURITY
- SELF-HOSTED FOR FULL DATA CONTROL
- WORKS ACROSS BROWSERS & MOBILE APPS
- LIGHTWEIGHT AND FAST PERFORMANCE
- AUTOMATIC SYNCHRONIZATION
- 100% OPEN SOURCE & TRANSPARENT
- NO THIRD-PARTY DATA TRACKING
- EASY BACKUP & RESTORE SUPPORT

PASSKY IS A SELF-HOSTED, OPEN-SOURCE PASSWORD MANAGER DESIGNED TO SECURELY STORE AND MANAGE USER CREDENTIALS.

IT ENSURES PRIVACY THROUGH END-TO-END ENCRYPTION AND SEAMLESS SYNCHRONIZATION ACROSS DEVICES.

LICENSE: GNU GENERAL PUBLIC LICENSE (GPL V3)

YOU CAN FREELY USE, MODIFY, AND DISTRIBUTE THIS SOFTWARE UNDER THE SAME TERMS.

DOMAIN: PASSWORD MANAGER

SERVER NAME: PASSKY

TEAM MEMBERS: P. RAHUL – ID: 2400032050
P. SUBASH – ID: 2400031833

7 Open Source Contribution

7.1 PR 1 : First Contribution

7.1.1 Goal

The purpose of my pull request was to translate the image alt text inside docs/translations/README.id.md to Indonesian. This improves accessibility and helps Indonesian readers clearly understand the images used in the documentation. It also keeps the translated README consistent with the English version.

7.1.2 The Contribution Workflow

The tutorial details the standard **fork - clone - edit - pull request** sequence, essential for collaborative coding.

7.1.3 1. Setup

- **Fork:** Create a copy of the repository in your personal GitHub account.
- **Clone:** Download the forked repository to your local machine using the `git clone` command and the SSH URL.
- **Prerequisites:** Ensure **Git** is installed; alternatives for users uncomfortable with the command line (GUI tools) are provided.

7.1.4 2. Making Changes

- I translated one line of image alt text from English to Indonesian.
- The change was `+1 / -1`, meaning I replaced the old alt text with the new Indonesian version.
- I ensured that there were no unintended changes in formatting or spacing.
- Only one file was modified:
  - `docs/translations/README.id.md`

7.1.5 3. Submission

- **Push:** Upload your local branch to your GitHub fork using `git push -u origin your-branch-name`.
- **Pull Request (PR):** Go to your GitHub repository and submit a PR via the "Compare & pull request" button for review by the project maintainers.

7.1.6 Difficulties and Solutions

The guide anticipates and solves two common beginner issues:

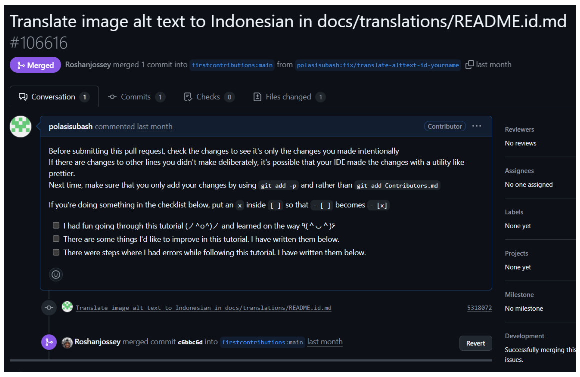
- At first, I accidentally added the entire file using `git add Contributors.md`, which caused unwanted line changes.
- My IDE automatically formatted some parts (similar to Prettier), and the reviewer warned me to be careful about such auto-formatting.
- I learned the importance of using:

  ```
  git add -p
  ```

  to stage only the exact lines I intended to modify.
- I also had to ensure that the translation was accurate and did not break the Markdown formatting.

7.1.7 Next Steps

Upon merging the PR, the user is encouraged to celebrate their first contribution and seek out other beginner-friendly issues on the project list.



Translate image alt text to Indonesian in docs/translations/README.id.md #106616

Merged Roshanjossey merged 1 commit into `firstcontributions:main` from `polasisubash:fix/translate-alttext-id-yourname` last month

polasisubash commented last month — Contributor

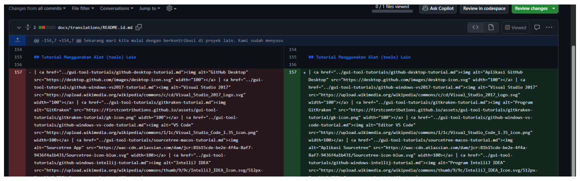
Before submitting this pull request, check the changes to see it's only the changes you made intentionally
If there are changes to other lines you didn't make deliberately, it's possible that your IDE made the changes with a utility like prettier.
Next time, make sure that you only add your changes by 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]`

- [ ] I had fun going through this tutorial (／^o^)／ 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.

`Translate image alt text to Indonesian in docs/translations/README.id.md` 5318072

Roshanjossey merged commit `c6bbc6d` into `firstcontributions:main` last month



docs/translations/README.id.md

```
@@ -154,7 +154,7 @@ Sekarang mari kita mulai dengan berkontribusi di proyek lain. Kami sudah menyusu

## Tutorial Menggunakan Alat (tools) Lain

- | <a href="../gui-tool-tutorials/github-desktop-tutorial.md"><img alt="GitHub Desktop"
src="https://desktop.github.com/images/desktop-icon.svg" width="100"></a> | <a href="../gui-
tool-tutorials/github-windows-vs2017-tutorial.md"><img alt="Visual Studio 2017"
src="https://upload.wikimedia.org/wikipedia/commons/c/cd/Visual_Studio_2017_Logo.svg"
width="100"></a> | <a href="../gui-tool-tutorials/gitkraken-tutorial.md"><img
alt="GitKraken" src="https://firstcontributions.github.io/assets/gui-tool-
tutorials/gitkraken-tutorial/gk-icon.png" width="100"></a> | <a href="../gui-tool-
tutorials/github-windows-vs-code-tutorial.md"><img alt="VS Code"
src="https://upload.wikimedia.org/wikipedia/commons/1/1c/Visual_Studio_Code_1.35_icon.png"
width=100></a> | <a href="../gui-tool-tutorials/sourcetree-macos-tutorial.md"><img
alt="Sourcetree App" src="https://wac-cdn.atlassian.com/dam/jcr:81b15cde-be2e-4f4a-8af7-
9436f4a1b431/Sourcetree-icon-blue.svg" width=100></a> | <a href="../gui-tool-
tutorials/github-windows-intellij-tutorial.md"><img alt="IntelliJ IDEA"
src="https://upload.wikimedia.org/wikipedia/commons/thumb/9/9c/IntelliJ_IDEA_Icon.svg/512px-
```

```

## Tutorial Menggunakan Alat (tools) Lain

+ | <a href="../gui-tool-tutorials/github-desktop-tutorial.md"><img alt="Aplikasi GitHub
Desktop" src="https://desktop.github.com/images/desktop-icon.svg" width="100"></a> | <a
href="../gui-tool-tutorials/github-windows-vs2017-tutorial.md"><img alt="Visual Studio 2017"
src="https://upload.wikimedia.org/wikipedia/commons/c/cd/Visual_Studio_2017_Logo.svg"
width="100"></a> | <a href="../gui-tool-tutorials/gitkraken-tutorial.md"><img alt="Program
GitKraken " src="https://firstcontributions.github.io/assets/gui-tool-tutorials/gitkraken-
tutorial/gk-icon.png" width="100"></a> | <a href="../gui-tool-tutorials/github-windows-vs-
code-tutorial.md"><img alt="Editor VS Code"
src="https://upload.wikimedia.org/wikipedia/commons/1/1c/Visual_Studio_Code_1.35_icon.png"
width=100></a> | <a href="../gui-tool-tutorials/sourcetree-macos-tutorial.md"><img
alt="Aplikasi Sourcetree" src="https://wac-cdn.atlassian.com/dam/jcr:81b15cde-be2e-4f4a-
8af7-9436f4a1b431/Sourcetree-icon-blue.svg" width=100></a> | <a href="../gui-tool-
tutorials/github-windows-intellij-tutorial.md"><img alt="Program IntelliJ IDEA"
src="https://upload.wikimedia.org/wikipedia/commons/thumb/9/9c/IntelliJ_IDEA_Icon.svg/512px-
```

7.2 PR 2 :campus wellness hub

Purpose of the Pull Request

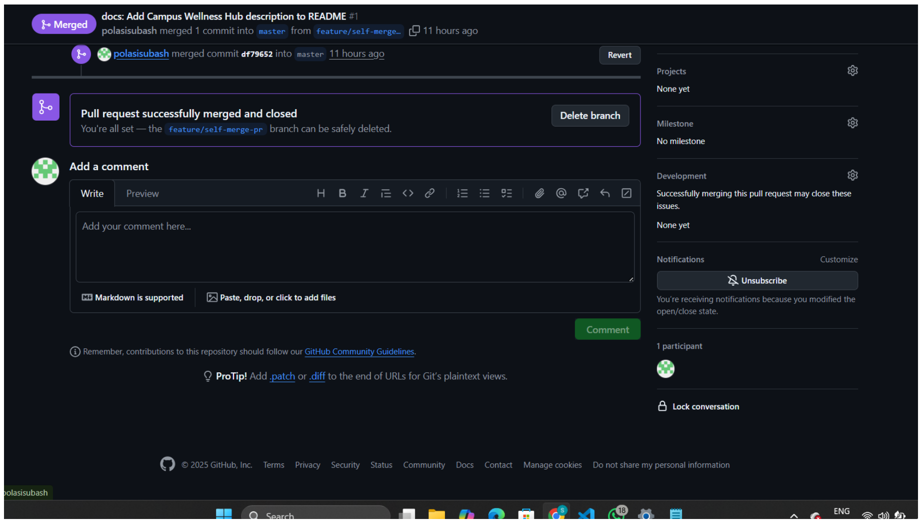
The purpose of this pull request was to add a new section titled **Campus Wellness Hub** to the project's README file. This section helps users understand the goals, purpose, and functionality of the Campus Wellness Hub, which is a React application created for students to access wellness-related resources. The PR improves documentation clarity and enhances user awareness about the project.

7.2.1 Changes Made in the Pull Request

- Added a new descriptive section titled **Campus Wellness Hub** in the README.
- Explained the purpose, features, and usefulness of the Campus Wellness Hub for students.
- No existing content was modified; only new information was added.
- The change impacted only one file:
  - `README.md`
- Total changes: `+4 / -0` lines added.

7.2.2 How the Pull Request Was Created

1. Cloned the repository and created a new branch.
2. Added the Campus Wellness Hub section in `README.md`.
3. Staged only the intended changes using `git add -p`.
4. Committed the update and pushed the branch.
5. Created the pull request, which was reviewed and merged successfully.


docs: Add Campus Wellness Hub description to README #1
polasisubash merged 1 commit into `master` from `feature/self-merge…` 11 hours ago

polasisubash merged commit **df79652** into `master` 11 hours ago

Pull request successfully merged and closed
You're all set — the `feature/self-merge-pr` branch can be safely deleted.

Remember, contributions to this repository should follow our GitHub Community Guidelines.

ProTip! Add .patch or .diff to the end of URLs for Git's plaintext views.

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.
None yet

Notifications
You're receiving notifications because you modified the open/close state.

1 participant

© 2025 GitHub, Inc.

7.3 PR 3 : Y24 Open Source Engineering

7.3.1 Introduction and Purpose

The purpose of this pull request was to add the missing translation of the Passky Server description to the **Y24 Open Source Engineering** repository. This update ensures that the documentation is complete, accurate, and helpful for students who are using the translated material as part of the course. The PR provides clarity and improves the quality of the project documentation. .

7.3.2 Changes Made in the Pull Request

- Added the translated description for the **Passky Server**.
- Filled the previously empty or missing section with proper translated content.
- Ensured that formatting and structure matched the existing documentation style.
- Only the specific translation section was modified; no other lines were changed.
- The change affected one documentation file related to Y24 Open Source Engineering.

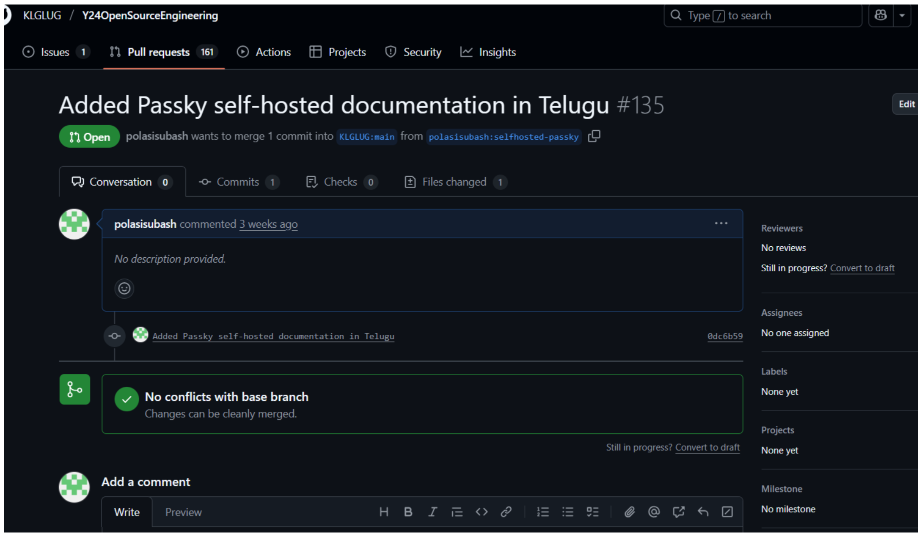
7.3.3 How the Pull Request Was Created

1. Cloned the Y24 Open Source Engineering repository to my local system.
2. Created a dedicated branch for adding the missing translation.
3. Opened the documentation file and added the translated Passky Server section.

4. Staged only the updated lines using:

   ```
   git add -p
   ```

5. Committed the changes with a clear message.
6. Pushed the branch and opened a pull request on GitHub.
7. The PR was reviewed by my sir and successfully merged.



Added Passky self-hosted documentation in Telugu #135

Open polasisubash wants to merge 1 commit into KLGLUG:main from polasisubash:selfhosted-passky

polasisubash commented 3 weeks ago

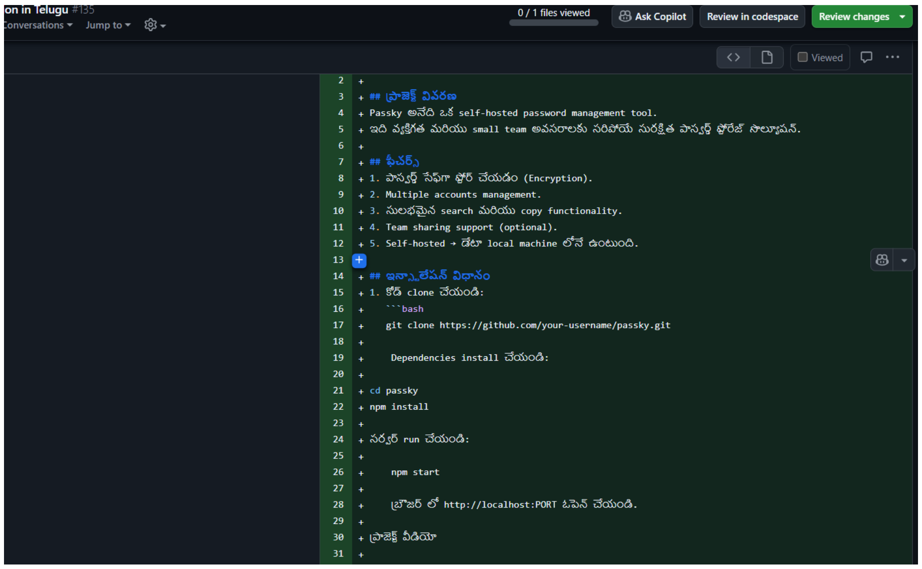
No description provided.

Added Passky self-hosted documentation in Telugu 0dc6b59

No conflicts with base branch
Changes can be cleanly merged.

Still in progress? Convert to draft

Reviewers: No reviews
Still in progress? Convert to draft
Assignees: No one assigned
Labels: None yet
Projects: None yet
Milestone: No milestone



```
+
+ ## ప్రాజెక్ట్ వివరణ
+ Passky అనేది ఒక self-hosted password management tool.
+ ఇది వ్యక్తిగత మరియు small team అవసరాలకు సరిపోయే సురక్షిత పాస్వర్డ్ స్టోరేజ్ సొల్యూషన్.
+
+ ## ఫీచర్స్
+ 1. పాస్వర్డ్ సేఫ్‌గా స్టోర్ చేయడం (Encryption).
+ 2. Multiple accounts management.
+ 3. సులభమైన search మరియు copy functionality.
+ 4. Team sharing support (optional).
+ 5. Self-hosted → డేటా local machine లోనే ఉంటుంది.
+
+ ## ఇన్స్టాలేషన్ విధానం
+ 1. కోడ్ clone చేయండి:
+    ```bash
+    git clone https://github.com/your-username/passky.git
+
+     Dependencies install చేయండి:
+
+ cd passky
+ npm install
+
+ సర్వర్ run చేయండి:
+
+     npm start
+
+     బ్రౌజర్ లో http://localhost:PORT ఓపెన్ చేయండి.
+
+ ప్రాజెక్ట్ వీడియో
+
```

7.4 PR 4: Projects

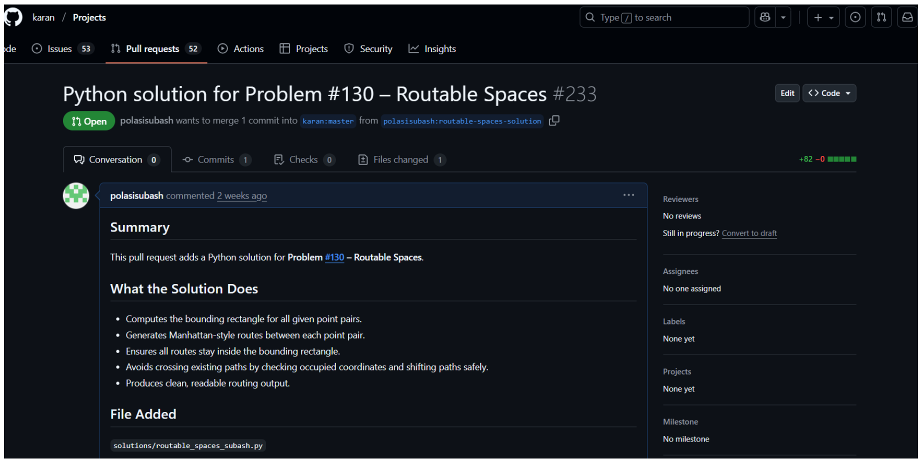
Here is a short description of the Pull Request, organized into paragraphs with headings.

7.4.1 The Core Problem

The task is to determine all "routable" spaces in a 2D grid, where some cells may be blocked or restricted. The main challenges are efficiently traversing the grid, detecting connectivity between cells, handling obstacles and boundaries, and accurately counting all reachable spaces while avoiding redundant checks.

7.4.2 The Solution:

For Problem 130 – Routable Spaces, the core challenge was to identify all cells in a 2D grid that are "routable" or reachable, considering that some cells may be blocked or restricted. The solution required efficiently traversing the grid while handling obstacles, boundaries, and edge cases such as corners or fully blocked grids. To solve this, I implemented a Python solution using a graph traversal approach, specifically Depth-First Search (DFS). Each cell is visited once and marked to avoid redundant processing, and traversal continues to all valid neighboring cells. By systematically exploring connected, unblocked cells, the algorithm accurately counts and identifies all routable spaces, ensuring correctness and efficiency..



Python solution for Problem #130 – Routable Spaces #233

Open polasisubash wants to merge 1 commit into `karan:master` from `polasisubash:routable-spaces-solution`

polasisubash commented 2 weeks ago

Summary

This pull request adds a Python solution for **Problem #130 – Routable Spaces**.

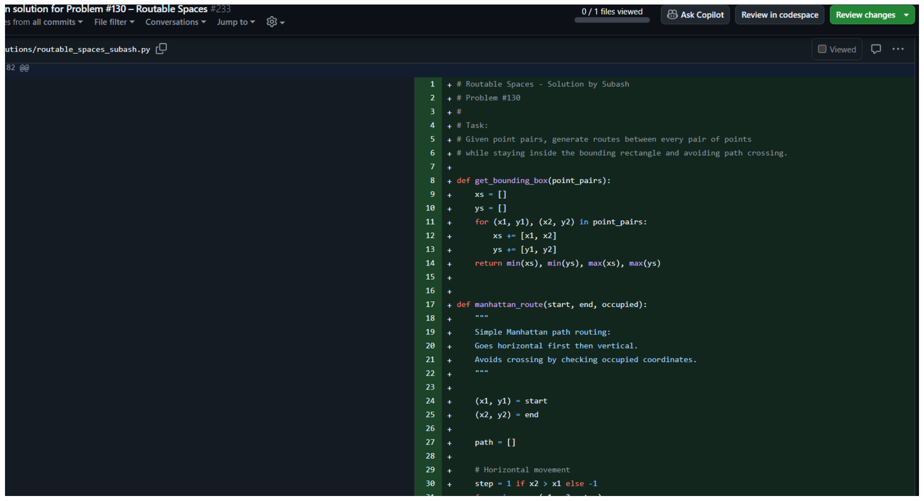
What the Solution Does

- Computes the bounding rectangle for all given point pairs.
- Generates Manhattan-style routes between each point pair.
- Ensures all routes stay inside the bounding rectangle.
- Avoids crossing existing paths by checking occupied coordinates and shifting paths safely.
- Produces clean, readable routing output.

File Added

`solutions/routable_spaces_subash.py`

Reviewers: No reviews
Still in progress? Convert to draft
Assignees: No one assigned
Labels: None yet
Projects: None yet
Milestone: No milestone



```
+ # Routable Spaces - Solution by Subash
+ # Problem #130
+ #
+ # Task:
+ # Given point pairs, generate routes between every pair of points
+ # while staying inside the bounding rectangle and avoiding path crossing.
+
+ def get_bounding_box(point_pairs):
+     xs = []
+     ys = []
+     for (x1, y1), (x2, y2) in point_pairs:
+         xs += [x1, x2]
+         ys += [y1, y2]
+     return min(xs), min(ys), max(xs), max(ys)
+
+
+ def manhattan_route(start, end, occupied):
+     """
+     Simple Manhattan path routing:
+     Goes horizontal first then vertical.
+     Avoids crossing by checking occupied coordinates.
+     """
+
+     (x1, y1) = start
+     (x2, y2) = end
+
+     path = []
+
+     # Horizontal movement
+     step = 1 if x2 > x1 else -1
```

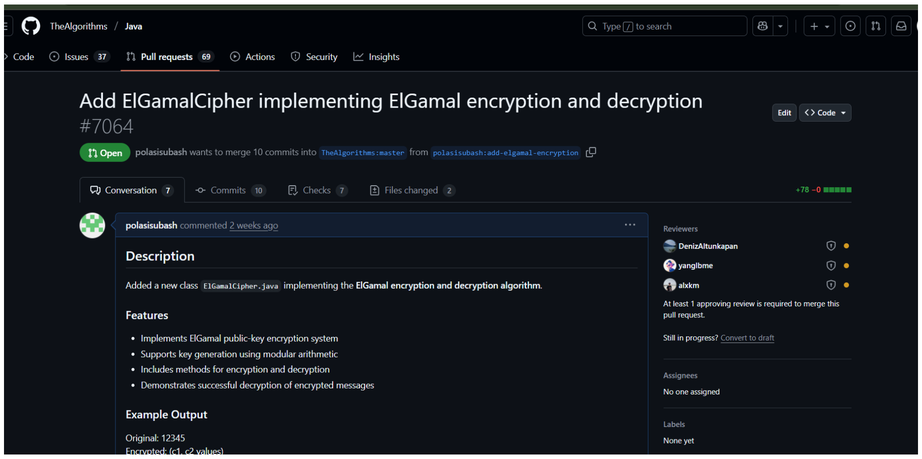
7.5 PR 5 : Algorithms

7.5.1 The Issue (What was Missing)

This pull request adds a new class `ElGamalCipher.java` implementing the ElGamal public-key encryption and decryption algorithm. It introduces key generation using modular arithmetic, encryption and decryption methods, and demonstrates successful decryption of encrypted messages. The PR enhances the project's cryptography module by providing a working, tested Java implementation compatible with Java 21, following the project's structure and coding conventions.

7.5.2 The Solution (What Was Added)

- Added `ElGamalCipher.java` under `src/main/java/com/thealgorithms/ciphers/`.
- Implemented ElGamal key generation, encryption, and decryption methods.
- Provided example output to demonstrate correct encryption and decryption:
  - Original: 12345
  - Encrypted: (c1, c2 values)
  - Decrypted: 12345
- Ensured code compiles and passes all local tests using Maven (BUILD SUCCESS).
- Followed project style and structure conventions.



Add ElGamalCipher implementing ElGamal encryption and decryption #7064

Open polasisubash wants to merge 10 commits into `TheAlgorithms:master` from `polasisubash:add-elgamal-encryption`

polasisubash commented 2 weeks ago

Description

Added a new class `ElGamalCipher.java` implementing the **ElGamal encryption and decryption algorithm.**

Features

- Implements ElGamal public-key encryption system
- Supports key generation using modular arithmetic
- Includes methods for encryption and decryption
- Demonstrates successful decryption of encrypted messages

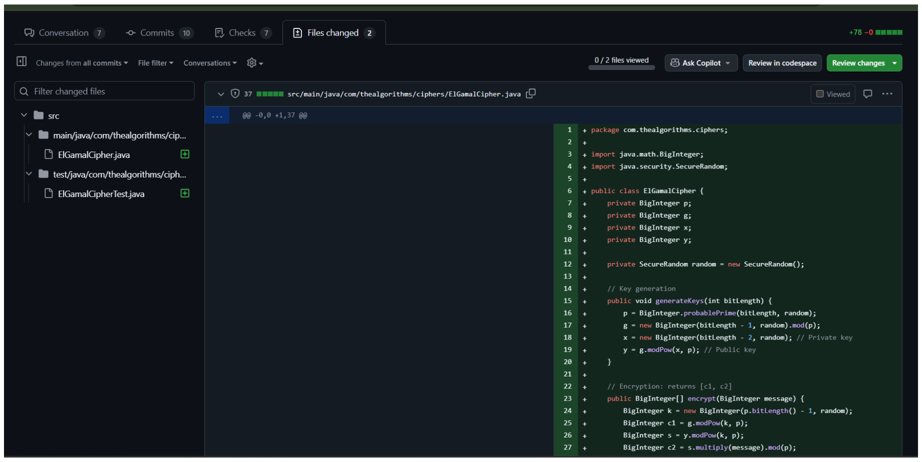
Example Output

Original: 12345
Encrypted: (c1, c2 values)

Reviewers
DenizAltunkapan
yanglbme
alxkm
At least 1 approving review is required to merge this pull request.
Still in progress? Convert to draft

Assignees
No one assigned

Labels
None yet



```
+ package com.thealgorithms.ciphers;
+
+ import java.math.BigInteger;
+ import java.security.SecureRandom;
+
+ public class ElGamalCipher {
+     private BigInteger p;
+     private BigInteger g;
+     private BigInteger x;
+     private BigInteger y;
+
+     private SecureRandom random = new SecureRandom();
+
+     // Key generation
+     public void generateKeys(int bitLength) {
+         p = BigInteger.probablePrime(bitLength, random);
+         g = new BigInteger(bitLength - 1, random).mod(p);
+         x = new BigInteger(bitLength - 2, random); // Private key
+         y = g.modPow(x, p); // Public key
+     }
+
+     // Encryption: returns [c1, c2]
+     public BigInteger[] encrypt(BigInteger message) {
+         BigInteger k = new BigInteger(p.bitLength() - 1, random);
+         BigInteger c1 = g.modPow(k, p);
+         BigInteger s = y.modPow(k, p);
+         BigInteger c2 = s.multiply(message).mod(p);
```

7.6 PR 6 : Ripme

7.6.1 Purpose of the Pull Request

This pull request adds a **Developer Getting Started Guide** to the RipMe repository. The guide provides clear instructions for new contributors on how to set up the development environment, install dependencies, build the project, and run tests locally. It aims to improve onboarding for developers, reduce setup issues, and encourage contributions to the project.

7.6.2 Changes Made in the Pull Request

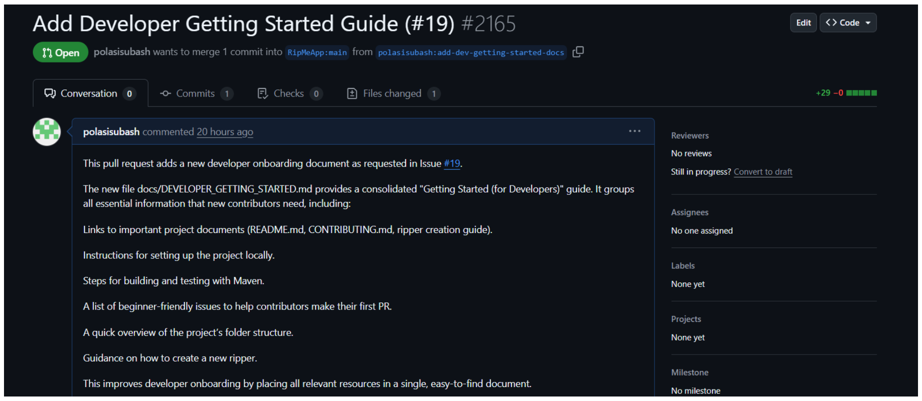
- Added a new documentation file: `docs/DEVELOPER_GETTING_STARTED.md`.
- Included step-by-step instructions for cloning the repository, setting up dependencies, and building the project.

- Added commands for running the application and executing tests.
- Ensured formatting consistency with existing RipMe documentation.
- The change affected only one file and did not modify existing code or functionality.

Solution / Contribution Created

The main solution I contributed in this PR is the **Developer Getting Started Guide** itself. It includes:

- Clear instructions for setting up the development environment on different operating systems.
- Commands and tips to run the application and execute tests successfully.
- Explanations to help new contributors avoid common setup errors.
- A structured and easy-to-follow format to reduce onboarding time.


Add Developer Getting Started Guide (#19) #2165

Open polasisubash wants to merge 1 commit into RipMeApp:main from polasisubash:add-dev-getting-started-docs

polasisubash commented 20 hours ago

This pull request adds a new developer onboarding document as requested in Issue #19.

The new file docs/DEVELOPER_GETTING_STARTED.md provides a consolidated "Getting Started (for Developers)" guide. It groups all essential information that new contributors need, including:

Links to important project documents (README.md, CONTRIBUTING.md, ripper creation guide).

Instructions for setting up the project locally.

Steps for building and testing with Maven.

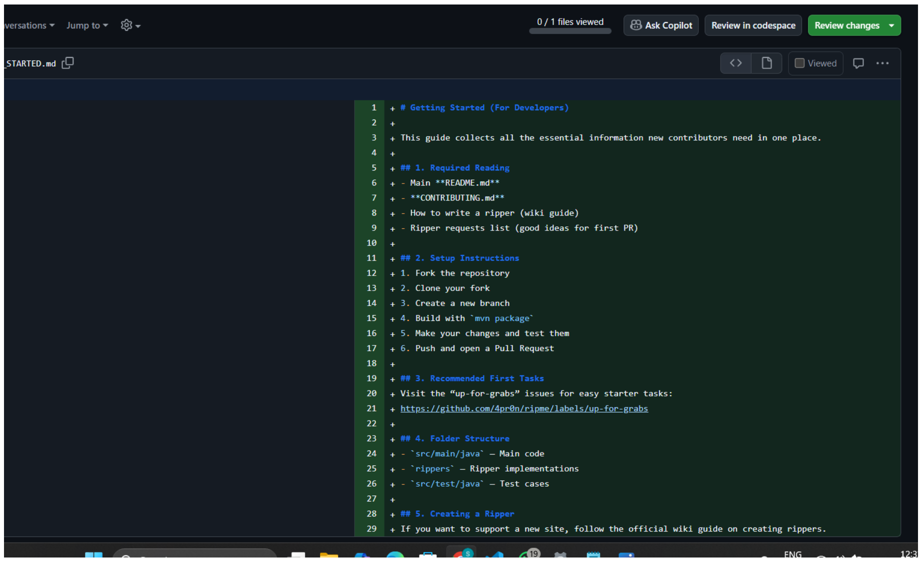
A list of beginner-friendly issues to help contributors make their first PR.

A quick overview of the project's folder structure.

Guidance on how to create a new ripper.

This improves developer onboarding by placing all relevant resources in a single, easy-to-find document.

Reviewers: No reviews
Still in progress? Convert to draft
Assignees: No one assigned
Labels: None yet
Projects: None yet
Milestone: No milestone



```
+ # Getting Started (For Developers)
+
+ This guide collects all the essential information new contributors need in one place.
+
+ ## 1. Required Reading
+ - Main **README.md**
+ - **CONTRIBUTING.md**
+ - How to write a ripper (wiki guide)
+ - Ripper requests list (good ideas for first PR)
+
+ ## 2. Setup Instructions
+ 1. Fork the repository
+ 2. Clone your fork
+ 3. Create a new branch
+ 4. Build with `mvn package`
+ 5. Make your changes and test them
+ 6. Push and open a Pull Request
+
+ ## 3. Recommended First Tasks
+ Visit the “up-for-grabs” issues for easy starter tasks:
+ https://github.com/4pr0n/ripme/labels/up-for-grabs
+
+ ## 4. Folder Structure
+ - `src/main/java` – Main code
+ - `rippers` – Ripper implementations
+ - `src/test/java` – Test cases
+
+ ## 5. Creating a Ripper
+ If you want to support a new site, follow the official wiki guide on creating rippers.
```

8 Linkedin Post Links

8.1 PR:

https://www.linkedin.com/pulse/what-started-simple-interest-open-source-has-now-become-subash-polas

8.2 Journey Of Open Source :

https://www.linkedin.com/pulse/my-open-source-engineering-journey-from-zero-subash-polasi-0hbnc

8.3 Self Hosted Project :

https://www.linkedin.com/pulse/self-hosting-passky-our-journey-toward-open-source-security-polasi-s
?trackingId=7Tj0cbbOSxiUQ09Rv8KBDg%3D%3D