(DEEMED TO BE UNIVERSITY) HONORS THROUGH EXPERIENTIAL LEARNING

EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

# OPEN SOURCE ENGINEERING

**Student ID:** 2400032836
**Student Name:** Kona Spandana
**Semester:** Odd
**Academic Year:** 2025-2026
**Course Code:** 24CS02EF

# 1 Understanding the Core Ubuntu Linux Distribution

#### 1.0.1 1. Overview and Philosophy

Ubuntu is a widely used open-source Linux distribution known for its simplicity, stability, and user-friendly design. It is based on Debian and provides a well-maintained ecosystem suitable for developers, students, and enterprise environments. Ubuntu follows a six-month release cycle, ensuring frequent updates, security patches, and long-term support (LTS) versions for production use. The philosophy of Ubuntu is deeply rooted in the African concept of "humanity to others," emphasizing community, collaboration, and inclusiveness. It promotes free access to software, transparency in development, and the idea that technology should be shared and improved collectively. Ubuntu encourages contributions from users worldwide, making it a truly community-driven platform. Its design principles focus on accessibility, openness, and ease of use, ensuring that both beginners and advanced users can operate efficiently. Ubuntu prioritizes security through robust package management, permission control, and open-source auditing.

#### 1.0.2 2. The Desktop Experience (GNOME)

The default Ubuntu desktop is powered by the GNOME environment, offering a refined, intuitive, and performance-oriented user interface. It features a sleek, left-aligned dock that keeps frequently used applications readily accessible, supporting a smooth and organized workflow. At the heart of GNOME's usability is the Activities Overview, which can be opened using the Super (Windows) key. This centralized space enables users to view all active windows, switch between multiple workspaces, and perform system-wide searches with ease. The interface emphasizes productivity through its clean layout, fluid animations, and minimal visual distractions. Additionally, Ubuntu delivers excellent hardware compatibility right from installation, ensuring that most devices, drivers, and peripherals work seamlessly without requiring manual configuration. This strong integration between GNOME and Ubuntu results in a polished, user-friendly desktop experience that caters to both new users and experienced developers.

#### 1.0.3 3. Software Management and Packaging

Ubuntu provides a powerful and versatile approach to software management through its dual packaging ecosystem. The traditional Advanced Packaging Tool (APT) remains at the core of system management, using DEB packages to deliver stable applications and essential system components directly from trusted repositories. Alongside this, Ubuntu integrates Snap packages, a modern and universal packaging method introduced by Canonical. Snaps include all necessary libraries and dependencies within a single bundle, ensuring the application behaves consistently on any Ubuntu release. They also operate inside a secure sandbox, preventing them from interfering with other system processes and adding an additional layer of security. This combination of APT and Snaps provides users with both reliability and modern flexibility, giving access to a broad collection of software that is secure, frequently updated, and optimized for diverse use cases.

# 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:** Full Disk Encryption (FDE) protects an entire disk or large partition by encrypting all contents, including system components, swap space, and user files. **How it works:** Ubuntu implements FDE using **LUKS** (Linux Unified Key Setup). During boot, the user is asked to enter a **passphrase**. Once authenticated, LUKS unlocks the drive, and all cryptographic operations happen seamlessly in the background. **Purpose:** The main goal is to defend against data exposure caused by **device theft** or any form of **unauthorized physical access**. Without the correct passphrase, the encrypted drive remains completely unreadable. **Implementation:** FDE is most easily configured during installation by selecting the "Encrypt the Ubuntu installation" option. Enabling it after installation is technically possible but considerably more challenging.

### 2.1.2 2. File and Directory Encryption

**What it is:** This approach focuses on encrypting particular files, folders, or messages, giving users precise control over which pieces of data receive protection. **Tools:**

- **GPG (GNU Privacy Guard):** A widely used tool for securing individual files and enabling encrypted communication through **public-key cryptography**.
- **eCryptfs (older):** An earlier solution for protecting a user's Home directory; however, it is now mostly replaced by full-disk encryption methods.

## 2.2 GPG (GNU Privacy Guard) Explained

**GPG** is the GNU version of the **OpenPGP** standard (derived from Pretty Good Privacy – PGP). It plays a crucial role in encrypting specific files and enabling secure, verified communication between users.

### 2.2.1 1. Core GPG Concepts

GPG is built on **asymmetric cryptography**, meaning it operates using two interconnected keys:

- **Public Key:** This key can be shared openly. Others use it to **encrypt** data intended for you or to **validate** digital signatures that you create.
- **Private (Secret) Key:** This key must remain **confidential** and is secured with a strong passphrase. It allows you to **decrypt** messages sent to you and to **sign** files to prove their authenticity.

### 2.2.2 2. Basic GPG Command-Line Usage

On Ubuntu, GPG is typically available by default and is mainly operated through the Terminal for generating keys and managing encrypted files.

**A. Generating a Key Pair** Before using GPG, you need to create your own public–private key pair:

Bash

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

During this process, you will choose the key algorithm (commonly RSA and RSA), the key length (4096 bits is a strong option), an expiration period, and then provide your Name, Email, and a secure **passphrase** to safeguard your private key.

**B. Encrypting a File for Yourself (Symmetric Encryption)** If you want to quickly protect a file using just one passphrase, you can perform symmetric encryption:

Bash

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

GPG will ask for a passphrase and will produce an encrypted version named `myfile.txt.gpg`.

**C. Encrypting a File for Someone Else (Asymmetric Encryption)** For securely sending a file to another person, you must encrypt it using their **Public Key** (which must already be imported into your keyring via `gpg --import`):

Bash

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

This command outputs `mysecretfile.doc.gpg`, which only the intended recipient can decrypt using their Private Key.

**D. Decrypting a File** To access the contents of a file that has been encrypted for you:

Bash

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

GPG will request the passphrase associated with your Private Key. You can also specify a custom output file using the `--output` option.

## 3 Sending Encrypted Email

### 3.1 Prerequisite: Setting Up GPG

Before encrypted email can be exchanged, both you and the other person must have functioning GPG key pairs and share your public keys with one another:

1. **Generate Keys:** Each user must create their own GPG public/private key pair (using the earlier command `gpg --full-generate-key`).
2. **Exchange Public Keys:** You must obtain your recipient's **Public Key**, and they must have yours. This can be done in multiple ways:
   - **Exporting** your key manually: `gpg --armor --export "Recipient Name" > recipient_key.asc` and sending them the generated `.asc` file.
   - **Publishing** the key to a public key server so others can fetch it.
3. **Import Key:** Add the recipient's public key to your keyring using: `gpg --import recipient_key.asc`.

### 3.2 Sending the Encrypted Email

The easiest and most widely used method for sending GPG-encrypted emails on Ubuntu is through **Mozilla Thunderbird**, which includes built-in OpenPGP support (previously provided by the **Enigmail** extension).

#### 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 rely on the mail client's integrated GPG features to complete two main operations:

1. **Encryption:** The message must be encrypted using the **Public Key** of the intended recipient. Only their matching **Private Key** is capable of decrypting it. For messages sent to multiple people, the email must be encrypted with the Public Key of *each* recipient.
2. **Digital Signature:** You **sign** the email using **your Private Key**. This allows the receiver to confirm the sender's identity and ensures the message has not been altered during delivery.

Within Thunderbird, these actions are usually enabled through the **OpenPGP** or **Security** options in the compose window, where you simply activate both the **"Encrypt"** and **"Sign"** settings.

#### 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 encrypted message in its unreadable form.
2. Their mail application then employs their **Private Key** (unlocked with their passphrase) to decrypt the content and display the original message.
3. At the same time, the client checks your **Public Key** to validate the digital signature, ensuring the message is genuine and unaltered.

## 4 Privacy Tools From Prism Break

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

- **What it is:** A privacy-enhanced browser based on Firefox that sends all web traffic through the Tor network, which is powered by a global system of volunteer relays.
- **Privacy Focus:** Offers **high anonymity** by masking your real IP address and physical location. It also incorporates protections against browser fingerprinting and tracking.
- **PRISM Break Note:** PRISM Break recommends Tor Browser as the preferred option for users who require the highest level of anonymity while browsing the internet.

#### 4.0.2 2. Debian (Operating Systems)

- **What it is:** A well-known and principled GNU/Linux distribution recognized for its strong commitment to Free Software ideals and its community-driven ethical guidelines.
- **Privacy Focus:** In contrast to closed-source platforms such as Windows or macOS (which PRISM Break avoids recommending), Debian is entirely open-source, making it fully inspectable. Its long-standing dedication to transparency and software freedom enhances user trust.
- **PRISM Break Note:** Frequently suggested as an excellent GNU/Linux option for newcomers leaving proprietary systems, valued for its reliability and strict free-software foundation.

#### 4.0.3 3. Thunderbird (Email Clients)

- **What it is:** An open-source, cross-platform email application created by Mozilla and widely used for managing personal and professional mail.

- **Privacy Focus:** Thunderbird is favored for secure communication because it is fully open-source and offers robust **built-in OpenPGP** (GPG) capabilities, enabling users to encrypt messages and apply digital signatures directly within the client.
- **PRISM Break Note:** Strongly endorsed as a secure email solution thanks to its integrated PGP tools and privacy-respecting design.

#### 4.0.4 4. KeePassXC (Password Managers)

- **What it is:** An open-source, cross-platform password management tool available at no cost.
- **Privacy Focus:** All credentials are saved inside a single, strongly encrypted database file that resides **locally** on your machine, ensuring you maintain complete ownership and control over your private information. No external cloud services are involved.
- **PRISM Break Note:** Recommended for its powerful encryption, transparent open-source nature, and reliance on local storage, which reduces dependence on third-party platforms.

#### 4.0.5 5. Firefox (Web Browsers)

- **What it is:** A quick, customizable, and security-focused web browser created by the Mozilla Foundation, a non-profit organization dedicated to an open internet.
- **Privacy Focus:** Firefox is fully open-source and offers strong privacy protections, such as Enhanced Tracking Protection (ETP), site isolation through container tabs, and a wide range of add-ons that improve security—such as uBlock Origin.
- **PRISM Break Note:** Though Tor Browser is preferred for maximum anonymity, Firefox is suggested for everyday browsing or when certain websites do not function correctly with Tor, as long as users adjust its privacy settings and switch to a privacy-respecting search engine.

## 5 Open Source License

Certainly. Here is the information about the **MIT License** organized into clear, descriptive headings, strictly maintaining a paragraph-only format within each section.

### 5.1 The Core Purpose and Classification

The **MIT License** is renowned as one of the most permissive and concise open-source licenses currently in use. Originating from the **Massachusetts Institute of Technology (MIT)**, its primary goal is to encourage maximum adoption and reuse of software with minimal legal friction. It is formally classified as a **permissive license**, meaning it grants users broad rights to **use**, **modify**, and **distribute** the software without imposing the reciprocal sharing obligations seen in **copyleft licenses**, such as the **GNU General Public License (GPL)**. This makes the MIT License highly favorable for both **commercial enterprises** and **proprietary software development**.

### 5.2 Granted Rights and Permissions

The license grants **blanket permission** to any individual or entity obtaining a copy of the software and its associated documentation to deal with the Software **without restriction**. Specifically, users are granted explicit rights to **use, copy, modify, merge, publish, distribute, sublicense, and/or sell** copies of the software. This expansive grant allows developers to incorporate **MIT-licensed code** into projects that may ultimately be **closed-source** and sold **commercially**, provided they meet the few mandated conditions.

### 5.3 The Only Two Conditions for Distribution

Unlike licenses that enforce reciprocal sharing, the MIT License has only two critical requirements that must be met when the software is distributed or included in a larger work. The first condition is the mandatory inclusion of the original **Copyright Notice** (e.g., `Copyright <YEAR> <COPYRIGHT HOLDER>`). The second requirement is the mandatory inclusion of the full **License Text** itself. If these two simple obligations are satisfied, the user is free to treat the code in virtually any manner they choose, including releasing their modifications under a **proprietary license**.

### 5.4 Disclaimer of Warranty and Liability

A key component of the MIT License is its comprehensive liability disclaimer, which serves to protect the original authors. The license emphatically states that the software is provided **"AS IS,"** meaning it comes without any guarantee or warranty of any kind, whether express or implied, including warranties of merchantability or fitness for a particular purpose. Furthermore, the license explicitly protects the authors and copyright holders, asserting they **shall not be held liable** for any claim, damages, or other liability arising from the use or other dealings in the software. This places the entire risk associated with the software onto the end-user.

# 6 Self Hosted Server

### 6.1 About

Jitsi Meet is an open-source, self-hosted video conferencing platform that makes online meetings easy and secure. It provides a clean and user-friendly interface for real-time audio and video communication, without depending on any paid or closed-source services. Built to give users full freedom and control, it is released under the Apache 2.0 License, which allows anyone to use, modify, and host it on their own servers.

#### 6.1.1 Key Features

- **Meal  Grocery Management Approach:** KitchenOwl follows a simple and efficient **recipe–meal–grocery** workflow, allowing users to add ingredients, plan meals, and generate shopping lists easily.
- **Organized Data Structure:** A self-hosted setup includes **users** – **recipes** – **meal plans** – **grocery lists**, helping families or teams collaborate smoothly.

- **User-Friendly Interface:** It provides a clean, modern UI with **Dark Mode**, visually rich **recipe cards**, and **intuitive navigation**, making kitchen planning enjoyable.
- **Power Features:** KitchenOwl includes tools like **automatic grocery list generation**, **ingredient grouping**, **recipe scaling**, **local storage**, and full offline-friendly access.
- **Secure Data Storage:** All data is stored securely on your own server using **Docker-based deployment**, ensuring complete privacy and ownership.
- **Real-Time Sync:** Changes to recipes, ingredients, or grocery lists are updated instantly across all connected users without refreshing the page.

## 6.2 Installation Process (Docker Compose)

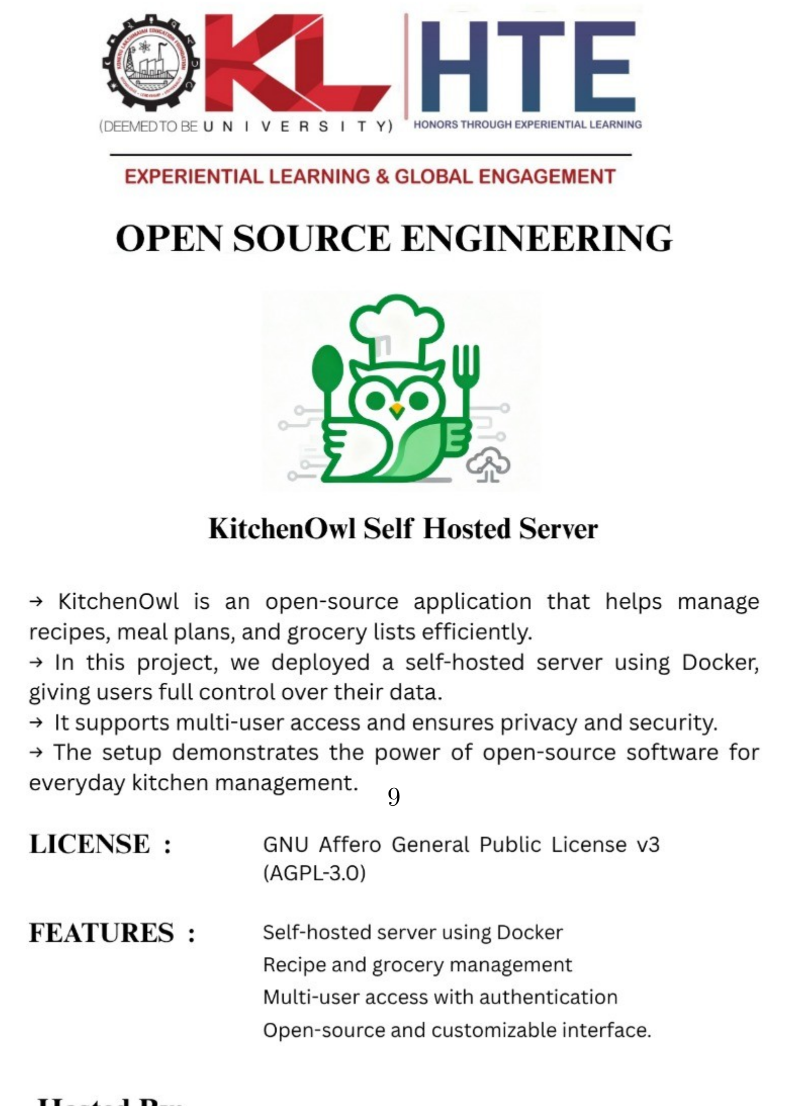
The recommended and most convenient way to self-host **KitchenOwl** is by using **Docker Compose**, which manages all required services such as the **backend API**, **frontend web UI**, and the **database** in a single structured multi-container setup. Before installation, ensure that **Docker** and **Docker Compose** are installed on your system.

The setup begins by **downloading the official Docker Compose files**. You can clone the KitchenOwl repository or obtain the required `docker-compose.yml` file from GitHub. After downloading, the most important step is to configure the environment variables inside the `.env` file. You must set the `BASE_URL` (e.g., `http://your-server-ip:8080`) and define secure credentials for the database and server components. Passwords can be generated securely using commands like `openssl rand -hex 32`.

Once the configuration is complete, the next step is to **start the service stack**. Running `docker compose up -d` will automatically download the necessary Docker images and launch the KitchenOwl frontend, backend, and database containers in detached mode. Docker Compose ensures that every component communicates correctly without manual configuration.

After the services are running, accessing **KitchenOwl** is simple: open a browser and navigate to the URL you specified in the `BASE_URL` variable or your server's IP address. From there, you can immediately begin adding recipes, planning meals, or inviting other users. KitchenOwl does not require separate login services unless you configure advanced authentication manually.

## KitchenOwl Resources


(DEEMED TO BE UNIVERSITY)
HTE
HONORS THROUGH EXPERIENTIAL LEARNING
EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

**OPEN SOURCE ENGINEERING**

**KitchenOwl Self Hosted Server**

→ KitchenOwl is an open-source application that helps manage recipes, meal plans, and grocery lists efficiently.
→ In this project, we deployed a self-hosted server using Docker, giving users full control over their data.
→ It supports multi-user access and ensures privacy and security.
→ The setup demonstrates the power of open-source software for everyday kitchen management.

| | |
|---|---|
| **LICENSE :** | GNU Affero General Public License v3 (AGPL-3.0) |
| **FEATURES :** | Self-hosted server using Docker<br>Recipe and grocery management<br>Multi-user access with authentication<br>Open-source and customizable interface. |

**Hosted By**

# 7 Open Source Contribution

## 7.1 PR 1 : First Contribution

### 7.1.1 Goal

The goal of the project is to help beginners understand the open-source contribution workflow by allowing them to add their name to the repository's `Contributors.md` file.

### 7.1.2 The Contribution Workflow

The tutorial explains the standard **fork – clone – edit – pull request** workflow used across open-source development.

### 7.1.3 1. Setup

- **Fork:** Create a personal copy of the repository in your GitHub account.
- **Clone:** Download the forked repository to your system using `git clone` with the SSH URL.
- **Prerequisites:** Ensure **Git** is installed; GUI alternatives are suggested for beginners.

### 7.1.4 2. Making Changes

- **Branch:** Create a new branch using `git switch -c your-branch-name`.
- **Edit:** Add your name to the `Contributors.md` file.
- **Commit:** Save the change with `git add Contributors.md` and `git commit -m "Add your-name to Contribut`

### 7.1.5 3. Submission

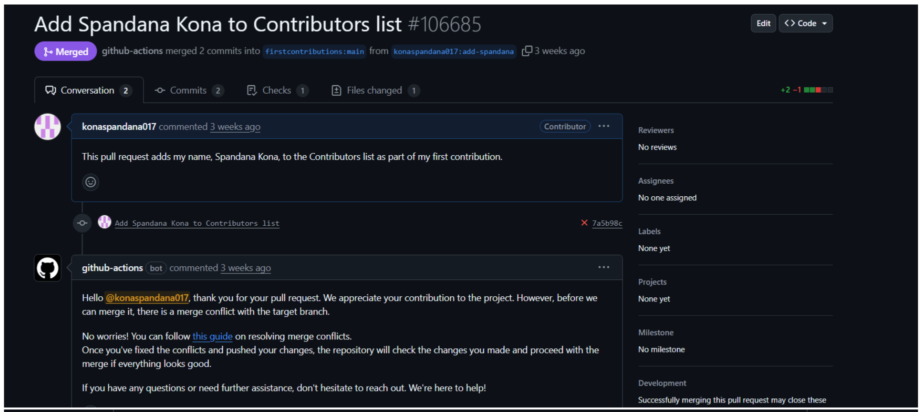
- **Push:** Upload your local branch to GitHub using `git push -u origin your-branch-name`.
- **Pull Request (PR):** Open your fork on GitHub and submit a PR through "Compare & pull request."

### 7.1.6 Difficulties and Solutions

- **Old Git Version:** If `git switch` fails, use `git checkout -b your-branch-name`.
- **Authentication Error:** If `git push` fails, configure an **SSH key** or **GitHub PAT** and ensure your remote URL uses SSH: `git remote set-url origin git@github.com:...`

### 7.1.7 Next Steps

After your PR is merged, you are encouraged to celebrate your first contribution and explore more beginner-friendly issues.



Add Spandana Kona to Contributors list #106685

Merged github-actions merged 2 commits into `firstcontributions:main` from `konaspandana017:add-spandana` 3 weeks ago

konaspandana017 commented 3 weeks ago · Contributor

This pull request adds my name, Spandana Kona, to the Contributors list as part of my first contribution.

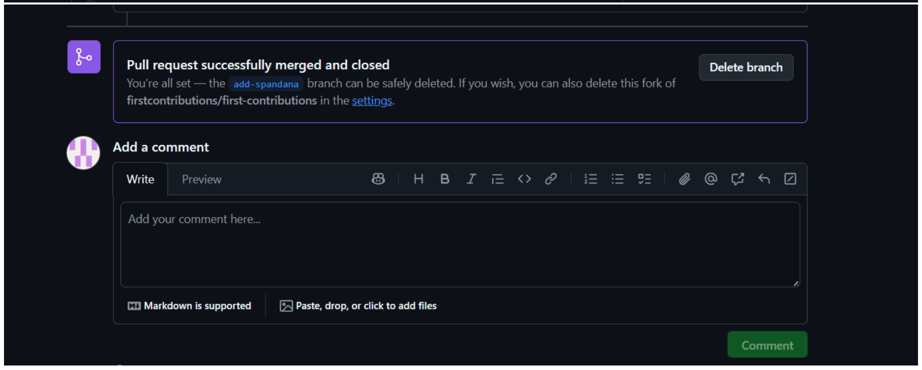
`Add Spandana Kona to Contributors list` ✕ 7a5b98c

github-actions bot commented 3 weeks ago

Hello @konaspandana017, thank you for your pull request. We appreciate your contribution to the project. However, before we can merge it, there is a merge conflict with the target branch.

No worries! You can follow this guide on resolving merge conflicts.
Once you've fixed the conflicts and pushed your changes, the repository will check the changes you made and proceed with the merge if everything looks good.

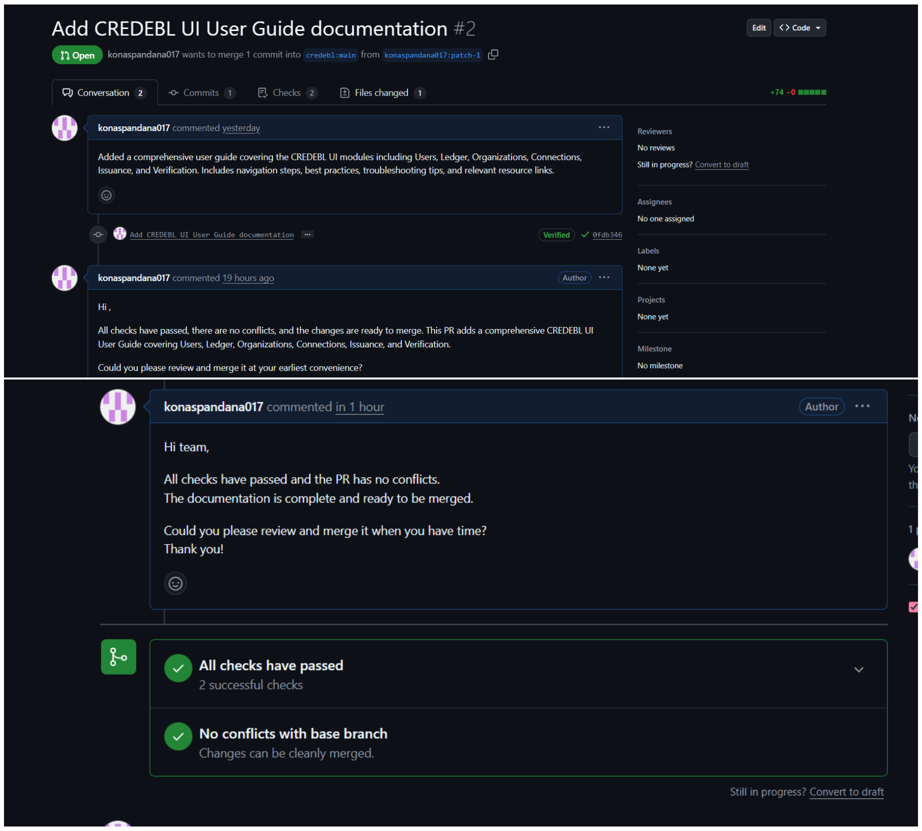
If you have any questions or need further assistance, don't hesitate to reach out. We're here to help!



Pull request successfully merged and closed

You're all set — the `add-spandana` branch can be safely deleted. If you wish, you can also delete this fork of **firstcontributions/first-contributions** in the settings.

## 7.2 PR 2 : credebl



Add CREDEBL UI User Guide documentation #2

Open konaspandana017 wants to merge 1 commit into `credebl:main` from `konaspandana017:patch-1`

Conversation 2 · Commits 1 · Checks 2 · Files changed 1 · +74 −0

**konaspandana017** commented yesterday

Added a comprehensive user guide covering the CREDEBL UI modules including Users, Ledger, Organizations, Connections, Issuance, and Verification. Includes navigation steps, best practices, troubleshooting tips, and relevant resource links.

`Add CREDEBL UI User Guide documentation` Verified ✓ `0fdb346`

**konaspandana017** commented 19 hours ago · Author

Hi ,

All checks have passed, there are no conflicts, and the changes are ready to merge. This PR adds a comprehensive CREDEBL UI User Guide covering Users, Ledger, Organizations, Connections, Issuance, and Verification.

Could you please review and merge it at your earliest convenience?

Reviewers: No reviews. Still in progress? Convert to draft

Assignees: No one assigned

Labels: None yet

Projects: None yet

Milestone: No milestone

**konaspandana017** commented in 1 hour · Author

Hi team,

All checks have passed and the PR has no conflicts.
The documentation is complete and ready to be merged.

Could you please review and merge it when you have time?
Thank you!

All checks have passed
2 successful checks

No conflicts with base branch
Changes can be cleanly merged.

Still in progress? Convert to draft

## 7.3 PR 2 : CREDEBL

CREDEBL is an open-source, decentralized identity and credential management platform designed to simplify the issuance, verification, and lifecycle management of digital credentials. It provides a secure and transparent way for organizations to manage users, ledgers, verifiable credentials, and cross-organization connections. The platform focuses on delivering a **user-friendly UI**, **modular architecture**, and **interoperable identity standards**, making it suitable for enterprises, institutions, and developers aiming to adopt trust-based digital identity solutions.

CREDEBL plays several important roles across decentralized identity ecosystems:

- **End-to-End Credential Management:** Through its UI, organizations can manage users, create issuers, generate credentials, and perform verifications—all through streamlined and guided workflows.
- **Interoperable DID and VC Platform:** CREDEBL supports decentralized identifiers (DIDs), verifiable credentials (VCs), and trust registries, enabling secure communication and credential exchanges across different entities.

- **Developer- and Admin-Friendly Interface:** The platform includes intuitive modules such as Users, Ledger, Organizations, Connections, Issuance, and Verification, allowing administrators and developers to operate the system without requiring deep protocol-level knowledge.

Overall, CREDEBL stands as a **transparent**, **secure**, and **enterprise-ready** platform for digital credential ecosystems, giving institutions complete control over identity workflows while maintaining full openness through its community-driven development model.

### 7.3.1 Licensing and Self-Hosting Options

CREDEBL is released as an open-source project under the **Apache License 2.0**, a highly permissive and industry-standard license. This allows individuals and organizations to freely use, modify, and distribute the software with flexibility for both research and production use.

The platform supports multiple deployment models:

- **Local Development:** Developers can run CREDEBL modules locally using Node.js and Docker-based services. This setup is helpful for testing UI changes, contributing documentation, or experimenting with DID/VC workflows.
- **Self-Hosted Deployments:** CREDEBL offers container-based deployment using **Docker Compose** or Kubernetes, enabling organizations to run the system on their own servers or cloud environments. This includes hosting ledger services, identity agents, and the UI dashboard.
- **Production-Ready Architecture:** With modular services and extensible APIs, CREDEBL can be scaled to enterprise workloads involving multiple issuers, registries, and verification flows.

### 7.3.2 Community and Support

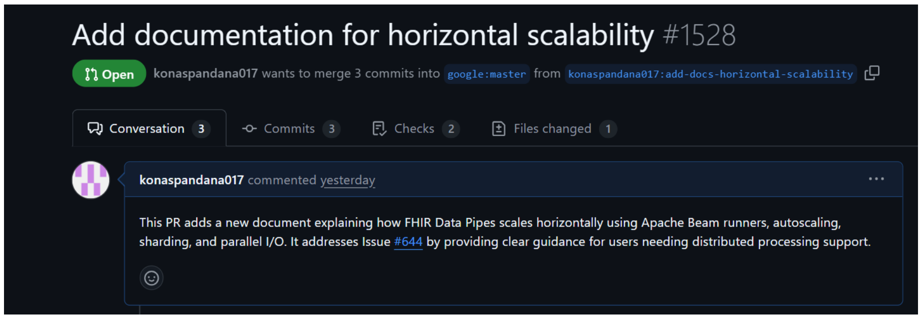
CREDEBL is actively maintained through its GitHub organization, where contributors collaborate on improving documentation, UI modules, DID/VC workflows, and platform security. Users can raise issues, submit feature requests, or contribute pull requests directly through the repository.

Support and collaboration typically occur through:

- GitHub Issues and Pull Requests
- Official documentation in the `credebl/docs` repository
- Community discussions and roadmap updates through GitHub

For security-related concerns, CREDEBL follows a responsible disclosure workflow, allowing sensitive issues to be reported privately to the maintainers. This ensures that vulnerabilities are handled professionally and promptly.

## 7.4 PR 3 : google/fhir-data-pipes


Add documentation for horizontal scalability #1528

Open konaspandana017 wants to merge 3 commits into google:master from konaspandana017:add-docs-horizontal-scalability

konaspandana017 commented yesterday

This PR adds a new document explaining how FHIR Data Pipes scales horizontally using Apache Beam runners, autoscaling, sharding, and parallel I/O. It addresses Issue #644 by providing clear guidance for users needing distributed processing support.

### 7.4.1 Introduction and Purpose

**FHIR Data Pipes** is an open-source data processing framework developed by Google to enable scalable, reliable, and efficient transformation of **HL7® FHIR® healthcare data**. It is built to support large healthcare workloads such as analytics, data warehousing, reporting pipelines, and interoperability use cases. The primary purpose of the project is to simplify end-to-end FHIR data movement by leveraging distributed processing engines, cloud-native infrastructure, and automated pipelines that can run at massive scale.

This PR contributes to that objective by adding comprehensive documentation explaining how the system supports **horizontal scalability**, enabling users to scale their data pipelines across multiple workers, machines, and compute environments.

### 7.4.2 Technical Components

Horizontal scalability in FHIR Data Pipes is achieved through its underlying architecture built on **Apache Beam**. Apache Beam runners—such as Dataflow, Spark, and Flink—allow pipelines to automatically distribute workloads across multiple worker nodes. Key technical components include:

- **Autoscaling Mechanisms:** Beam runners automatically increase or decrease the number of workers based on input load, CPU metrics, and backlog size.
- **Sharding and Parallelism:** Large FHIR datasets are divided into smaller shards, enabling many parallel workers to process separate batches without interfering with each other.
- **Parallel I/O Operations:** Reads and writes to FHIR stores, databases, or cloud buckets are parallelized to reduce bottlenecks and improve throughput.
- **Distributed Transform Execution:** Each step in the pipeline—mapping, validation, transformation, aggregation—is executed simultaneously across many nodes using Beam's parallel processing model.

The documentation added in this PR explains these components in a simple and structured manner so that users can understand how to configure, deploy, and optimize horizontally scalable pipelines.

### 7.4.3 Operation and Usage

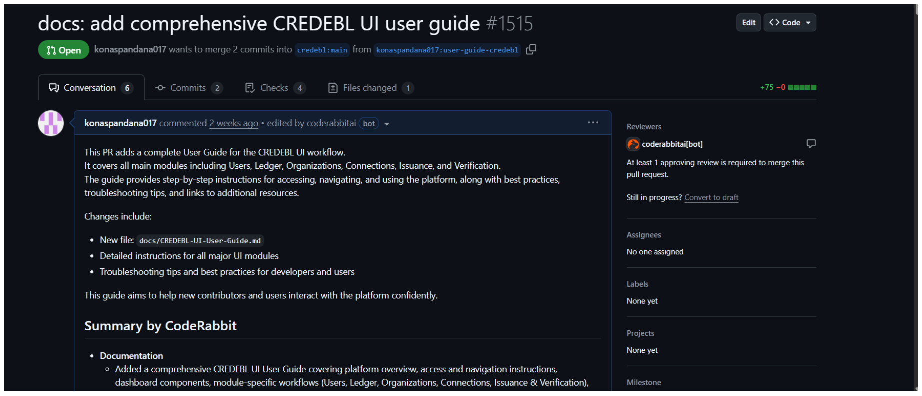
FHIR Data Pipes operates by executing Apache Beam pipelines on any supported distributed runner. Users can define their FHIR workflows in Python or Java and then deploy them to cloud or on-premise environments suitable for large-scale processing. Typical workloads include:

- processing large batches of FHIR resources,
- performing transformations for analytics,
- migrating data between FHIR servers,
- building real-time or batch healthcare data pipelines.

With horizontal scalability, users can run these workflows across hundreds of parallel workers, ensuring that even extremely large datasets—such as nationwide clinical records—can be processed efficiently.

The PR enhances the project by providing a detailed, user-friendly guide on how to enable this scalability, configure Beam runners, understand autoscaling behavior, and tune performance for distributed environments.

## 7.5 PR 4: Credebl/platform



docs: add comprehensive CREDEBL UI user guide #1515

Open konaspandana017 wants to merge 2 commits into `credebl:main` from `konaspandana017:user-guide-credebl`

konaspandana017 commented 2 weeks ago • edited by coderabbitai bot

This PR adds a complete User Guide for the CREDEBL UI workflow.
It covers all main modules including Users, Ledger, Organizations, Connections, Issuance, and Verification.
The guide provides step-by-step instructions for accessing, navigating, and using the platform, along with best practices, troubleshooting tips, and links to additional resources.

Changes include:

- New file: `docs/CREDEBL-UI-User-Guide.md`
- Detailed instructions for all major UI modules
- Troubleshooting tips and best practices for developers and users

This guide aims to help new contributors and users interact with the platform confidently.

**Summary by CodeRabbit**

- **Documentation**
  - Added a comprehensive CREDEBL UI User Guide covering platform overview, access and navigation instructions, dashboard components, module-specific workflows (Users, Ledger, Organizations, Connections, Issuance & Verification),

Reviewers
coderabbitai[bot]
At least 1 approving review is required to merge this pull request.

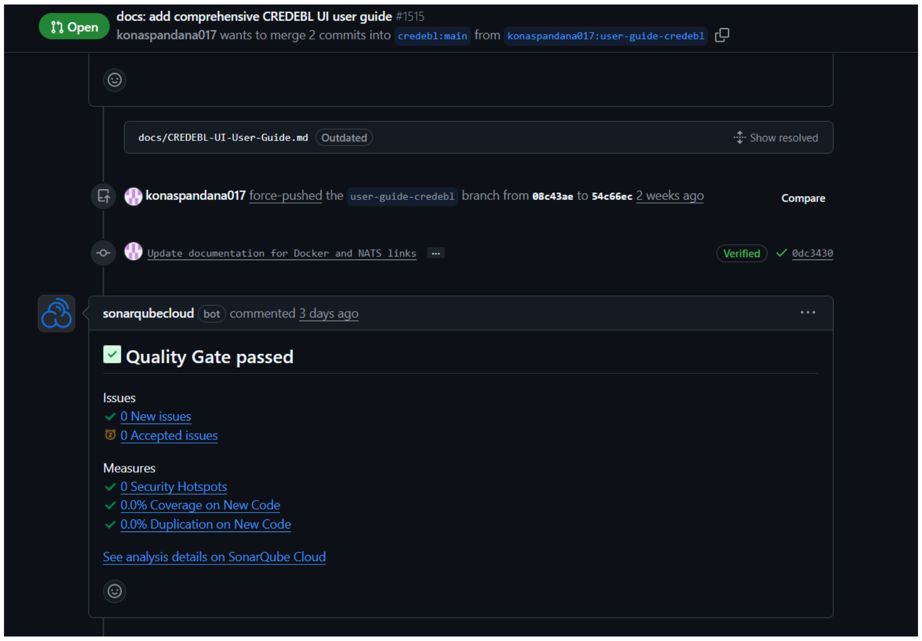
Still in progress? Convert to draft

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone



Open docs: add comprehensive CREDEBL UI user guide #1515
konaspandana017 wants to merge 2 commits into credebl:main from konaspandana017:user-guide-credebl

docs/CREDEBL-UI-User-Guide.md Outdated Show resolved

konaspandana017 force-pushed the user-guide-credebl branch from 08c43ae to 54c66ec 2 weeks ago Compare

Update documentation for Docker and NATS links ... Verified ✓ 0dc3430

sonarqubecloud bot commented 3 days ago

✓ Quality Gate passed

Issues
✓ 0 New issues
0 Accepted issues

Measures
✓ 0 Security Hotspots
✓ 0.0% Coverage on New Code
✓ 0.0% Duplication on New Code

See analysis details on SonarQube Cloud

### 7.5.1 The Core Problem

The primary issue addressed in this Pull Request was the absence of a structured and accessible **CREDEBL UI User Guide** within the `credebl/platform` repository. New contributors, end users, and integrators often struggled to understand how the platform's UI modules worked, as the existing documentation was either scattered, incomplete, or missing key workflow explanations. This created onboarding challenges, reduced productivity, and made it difficult for users to interact confidently with the system's features such as Users, Ledger, Organizations, Connections, Issuance, and Verification.

Additionally, the project lacked a standardized, comprehensive reference explaining navigation steps, dashboard behavior, troubleshooting, and best practices. This led to repeated questions in discussions, inconsistent understanding across contributors, and unnecessary dependency on maintainers for clarifications.

### 7.5.2 The Solution: A Comprehensive CREDEBL UI User Guide

This Pull Request introduces a complete and well-structured **CREDEBL UI User Guide** that consolidates all necessary information into a single Markdown document. The guide provides a full overview of the platform's interface, covering every major module: Users, Ledger, Organizations, Connections, Issuance, and Verification.

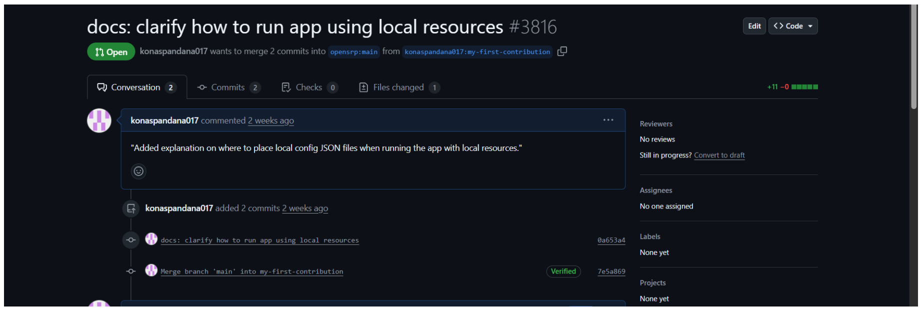
The documentation includes:

- **Module-Specific Workflows:** Clear instructions for performing actions within each UI module.
- **Navigation and Dashboard Guidance:** Step-by-step paths for accessing features and understanding the layout.

- **Best Practices:** Recommended workflows for administrators, developers, and end users.
- **Troubleshooting Tips:** Common issues and solutions designed to reduce friction and support requests.
- **Resource Links:** Helpful references for Docker setup, NATS connections, and additional documentation.

The PR adds the new file `docs/CREDEBL-UI-User-Guide.md` with no code changes, ensuring that it enhances usability without affecting the platform's functionality. These improvements significantly reduce onboarding time, improve contributor efficiency, and help maintain documentation consistency across the CREDEBL ecosystem.

## 7.6 PR 5 : opensrp/fhircore


docs: clarify how to run app using local resources #3816

Open konaspandana017 wants to merge 2 commits into `opensrp:main` from `konaspandana017:my-first-contribution`

konaspandana017 commented 2 weeks ago

"Added explanation on where to place local config JSON files when running the app with local resources."

konaspandana017 added 2 commits 2 weeks ago

`docs: clarify how to run app using local resources` 0a653a4

`Merge branch 'main' into my-first-contribution` Verified 7e5a869

Reviewers: No reviews

Still in progress? Convert to draft

Assignees: No one assigned

Labels: None yet

Projects: None yet

### 7.6.1 The Issue (What Was Missing)

The main issue addressed in this Pull Request was the lack of clear documentation on **how to run the FHIRCore application using local resource files**. Developers who wanted to test or customize the app with their own configuration JSON files did not have proper guidance on where those files should be placed or how the application reads them during runtime. This gap made local development confusing, especially for new contributors who needed to understand the folder structure, resource-loading behavior, and required configuration formats.

Without this clarification, developers often struggled to set up their local environment correctly, leading to misconfigurations, runtime errors, or unnecessary back-and-forth with maintainers.

### 7.6.2 The Solution (What Was Added)

This Pull Request adds a clear and helpful explanation to the documentation outlining the **exact location and structure** required for local configuration JSON files when running the app with local resources. The updated documentation now includes:

- **Precise file placement instructions** (e.g., which directories under `assets/` or resource folders should contain the JSON files).
- **Guidance on local resource loading**, explaining how the application detects and uses these files.

- **Improved clarity for developers** setting up their environment for offline, test, or custom configurations.

By adding this missing documentation, the PR makes it significantly easier for contributors to run the application locally, reduces confusion during setup, and ensures consistent workflows across the development community. The update does not modify any application code—only the documentation—making it safe, simple, and valuable for all users.

## 8 Linkedin Post Links

### 8.1 PR :

`https://www.linkedin.com/posts/spandana-kona-57117a349_cse-hte-kluniversity-activity-73991434347179`
`utm_source=share&utm_medium=member_desktop&rcm=ACoAAFcOWaUBwhun6Zt2Oahlj5P9lKRArNWL-pg`

### 8.2 Journey Of Open Source :

`https://www.linkedin.com/pulse/my-journey-opensource-engineering-spandana-kona-wpige`

### 8.3 Self Hosted Project :

`https://www.linkedin.com/pulse/kitchen-owl-self-hosted-server-spandana-kona-roroe?`
`lipi=urn%3Ali%3Apage%3Ad_flagship3_pulse_read%3BD7UhkyG5QbeeG7MDICKP%2BQ%3D%3D`