

A REPORT ON

MY OPEN SOURCE JOURNEY

Submitted in partial fulfillment of the requirements for the course

Open Source Engineering



Submitted by:

Mathew Addala

ID No: 2400030483

Department: CSE (HTE)

Department of Computer Science and Engineering
KONERU LAKSHMAIAH EDUCATION FOUNDATION

(Deemed to be University)

Green Fields, Vaddeswaram, Guntur District, A.P. - 522502

November 2025

TABLE OF CONTENTS

1. MY DEVELOPMENT ENVIRONMENT	1
1.1 The Distribution I Chose	
1.2 System Specifications	
1.3 Why I Selected This OS for My Journey	
1.4 My Experience So Far	
2. SECURITY AND ENCRYPTION	2
2.1 What is GPG?	
2.2 Why I Need It for Open Source	
2.3 Setting It Up on Ubuntu	
2.4 My Global Keys	
3. SENDING ENCRYPTED EMAILS	3
3.1 The Scenario	
3.2 Step 1: Exchanging Public Keys	
3.3 Step 2: Encrypting the Message	
3.3 Step 3: Sending and Decrypting	
4. ESSENTIAL PRIVACY TOOLS	4
4.1 Researching on Prism Break	
4.2 My Top 5 Selections	
5. OPEN SOURCE LICENSE	5
5.1 The License I Chose	
5.2 Why I Chose the MIT License	
5.3 What This Means for Users	
6. DEPLOYMENT AND OUTREACH	6
6.1 Self-Hosted Server Setup	
6.2 Localized Documentation (Telugu)	

7. EVENT POSTER	7
7.1 Project Promotion	
7.2 The Design Strategy	
8. OPEN SOURCE CONTRIBUTIONS	8
8.1 Overview	
8.2 Repository: TheAlgorithms/Python	
8.3 Repository: Free Programming Books	
8.4 Repository: First Contributions	
8.5 Evidence of Contributions	
9. CAMPUS OPEN SOURCE INITIATIVES	9
9.1 Project: KLeats	
9.2 The Proposal: Decentralized Delivery Governance	
9.3 Evidence of Proposal	
10. PUBLIC OUTREACH & BLOGS	10
10.1 LinkedIn Showcase: Samyak Event	
10.2 Technical Blog: My Open Source Journey	
10.3 LinkedIn Announcement: Blog Release	

1. MY DEVELOPMENT ENVIRONMENT

1.1 The Distribution I Chose

For my open source journey, I decided to use **Ubuntu 24.04 LTS**, which is also known as “Noble Numbat.” I picked this operating system because it is the industry standard for developers. Since I am just starting to contribute to open source projects, I needed a system that was both modern and reliable, and Ubuntu 24.04 offered the perfect balance.

1.2 System Specifications

Here are the technical details of the system I used:

- **Kernel:** Linux 6.8 (It runs very smoothly on my hardware).
- **Desktop Interface:** GNOME 46 (I found the interface very clean and easy to navigate).
- **Package Manager:** APT and Snap (These made installing tools extremely fast).
- **Display:** Wayland (The graphics felt snappy and responsive).

1.3 Why I Selected This OS for My Journey

I didn’t just choose Ubuntu for one project; I chose it to be my main workspace for learning open source. Here is why:

1. **It is Beginner-Friendly but Powerful:** Ubuntu was easy to install, especially with the new installer. I didn’t have to spend days configuring drivers; everything just worked out of the box, so I could start coding immediately.
2. **Stability Matters:** Since this is an “LTS” (Long Term Support) version, it is built to be stable for years. This was important to me because I didn’t want my system to break or update unexpectedly while I was in the middle of a big project.

3. **Great for Web Development:** All the tools I needed for my journey—like **Node.js**, **Git**, and **Visual Studio Code**—work perfectly on Ubuntu. Installing libraries for my React and Three.js projects was much easier here than on Windows.

1.4 My Experience So Far

Switching to Ubuntu 24.04 has improved my workflow significantly. The new GNOME desktop is faster and less distracting, which helps me focus on code. Also, whenever I ran into an error, the Ubuntu community is so huge that I could always find a solution online quickly. It really feels like the “home” of open source development.

2. SECURITY AND ENCRYPTION

2.1 What is GPG?

For the security part of my open source journey, I learned about **GPG** (GNU Privacy Guard). In simple terms, GPG is a tool used to secure data and verify who sent it. It works like a digital ID card. It uses two “keys”: a **public key** (which I share with everyone) and a **private key** (which I keep secret).

2.2 Why I Need It for Open Source

In the open source world, anyone can contribute code. This makes it very important to prove that the code I upload actually comes from me and hasn't been tampered with by someone else. By signing my work with GPG, developers around the world can verify my identity using public key servers.

2.3 Setting It Up on Ubuntu

Setting up GPG on Ubuntu 24.04 was done entirely through the terminal. Here is the exact process and commands I used:

1. **Generating the Keys:** First, I created my key pair using the generation command. I selected the **RSA and RSA** algorithm for strong security.

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

2. **Finding My Key ID:** Once generated, I needed to find my specific Key ID to share it. I used this command to list my secret keys:

```
gpg --list-secret-keys --keyid-format=long
```

This gave me a long code (like ABCDEF12ABCDEF12), which acts as my unique identifier.

3. **Publishing to the Ubuntu Keyserver:** To make my identity verifiable by others, I sent my public key to the official Ubuntu keyserver. This allows anyone to fetch my key and verify my signature.

```
gpg --keyserver keyserver.ubuntu.com --send-keys <MY_KEY_ID>
```

2.4 My Global Keys

Type	bits/keyID	cr. time	exp time	key expir
pub	(4)rsa4096/ffe68465f1fafa4eea1ba3393ed57e6c0f73b2a2	2025-08-12T09:31:25Z		
uid	Mathew Addala <2400030483@kluniversity.in>			
sig	cert 3ed57e6c0f73b2a2	2025-08-12T09:31:25Z		[selfsig]
sub	(4)rsa4096/355b0b42a93dff7201dc0b0807f93713893777f0	2025-08-12T09:31:25Z		
sig	sbind 3ed57e6c0f73b2a2	2025-08-12T09:31:25Z		[]

3. SENDING ENCRYPTED EMAILS

3.1 The Scenario

To test if my GPG keys were actually working, I decided to simulate a secure conversation. In this example, I wanted to send a secret project file to my teammate.

- **Sender (Me):** 2400030483@kluniversity.in
- **Receiver (Teammate):** 2400030625@kluniversity.in

3.2 Step 1: Exchanging Public Keys

Before I could send a secret message, my teammate and I had to swap “public keys.” It works like this: I need his public lock to secure the message, and he needs his private key to open it.

I asked him to send me his public key file (let’s call it `teammate_key.asc`). Once I had it, I added it to my system using the terminal:

```
gpg --import teammate_key.asc
```

3.3 Step 2: Encrypting the Message

I created a simple text file named `secret_project.txt` with the message: “*IAM MATHEW ADDALA*”

To lock this message so only he could read it, I encrypted it using his specific email address. This ensures that even if someone intercepts the email, they cannot read the contents without his private key.

```
gpg --encrypt --recipient 2400030625@kluniversity.in  
→ secret_project.txt
```

This command created a new file called `secret_project.txt.gpg`. When I opened this file, it just looked like scrambled text and random characters.

3.4 Step 3: Sending and Decrypting

I opened my KLU email account and composed a new email to 2400030625@kluniversity.in.

I attached the encrypted file (`secret_project.txt.gpg`) and hit send.

When he received it, he downloaded the file and used his private key to unlock it:

```
gpg --decrypt secret_project.txt.gpg
```

Because he successfully had the matching private key, the terminal showed him the original message: “*IAM MATHEW ADDALA*” This confirmed that our secure communication channel was working perfectly.

4. ESSENTIAL PRIVACY TOOLS

4.1 Researching on Prism Break

To complete my open source environment, I visited **prism-break.org**. This website lists software that respects user privacy. Since I am using Ubuntu 24.04, I specifically looked for tools that are easy to install, open source, and beginner-friendly.

4.2 My Top 5 Selections

1. Tor Browser (Anonymous Browsing)

Why I chose it: It is the easiest way to browse the internet anonymously. It automatically routes my traffic through a network of volunteer servers, hiding my location and identity.

Ubuntu Experience: Installing it was simple, and it works just like Firefox but with much higher security.

2. Thunderbird (Secure Email)

Why I chose it: This is the default email app for Ubuntu. It is open source and has built-in support for the **GPG keys** I generated in the previous step.

Ubuntu Experience: It allows me to send the encrypted emails I tested earlier without needing complex command-line tools every time.

3. KeePassXC (Password Manager)

Why I chose it: Unlike cloud-based managers, KeePassXC stores my passwords locally on my own hard drive in an encrypted file. This means I have total control over my data.

Ubuntu Experience: It integrates perfectly with the Ubuntu desktop and browsers, auto-typing passwords securely.

4. uBlock Origin (Ad Blocker)

Why I chose it: It is a lightweight extension that blocks ads and trackers. It is recommended because it doesn't just hide ads; it stops the tracking scripts from loading

in the first place.

Ubuntu Experience: I added this to my Firefox browser on Ubuntu immediately. It makes pages load faster and keeps my browsing history private.

5. **F-Droid (Open Source App Store)**

Why I chose it: While Ubuntu handles my laptop, **F-Droid** brings the open source philosophy to my phone. It is an alternative to the Google Play Store that only hosts free and open source software (FOSS).

Open Source Connection: It feels like using the Ubuntu “APT” package manager but for Android. It helps me keep my mobile environment as clean and transparent as my Linux desktop.

5. OPEN SOURCE LICENSE

5.1 The License I Chose

For my project, **GlobeTrotter**, I decided to use the **MIT License**. When I uploaded my code to GitHub, I included this license file to make it officially open source.

5.2 Why I Chose the MIT License

I picked this license because I wanted to give maximum freedom to the developer community. My goal for GlobeTrotter wasn't to restrict people or keep the code secret; I wanted the exact opposite.

I chose the MIT License so that:

- **The World Can Own It:** Anyone can take my code and use it for their own projects, whether they are students learning Three.js or developers building a commercial game.
- **People Can Enhance It:** If someone finds a bug or wants to add a new feature (like multiplayer support), they are free to modify the code however they like.
- **It Encourages Innovation:** By removing strict restrictions, I hope developers will feel encouraged to “make more of it”—taking my basic concept and turning it into something even better.

5.3 What This Means for Users

In simple terms, the MIT License tells the world: *“You can use this software for free, change it, and even sell it, as long as you keep my copyright notice in the file.”*

This simplicity is why I love open source. It allows my work on GlobeTrotter to live on and grow, even if I stop working on it myself.

6 LOCALIZATION (TELUGU README)

6.1 Project Documentation

To make our project accessible to local students and developers, I have translated the complete project documentation into Telugu.

గ్లోబ్ట్రాటర్ (GlobeTrotter)

గ్లోబ్ట్రాటర్ అనేది React మరియు Three.js తో నిర్మించిన పైరేట్-థీమ్డ్ 3D బ్రెజర్ హంట్ గేమ్. ఈ అప్లికేషన్ వినియోగదారులను గ్లోబ్ ను స్పిన్ చేయడానికి, వాస్తవ-ప్రపంచ స్థానాలను కనుగొనడానికి మరియు నిధిని కనుగొనడానికి లేదా ముందుగా దోచుకున్న ఛెస్ట్ ను కనుగొనడానికి ఆహ్వానిస్తుంది-అన్నీ మూడు-స్పిన్ పరిమితిలో.

ఈ ప్రాజెక్ట్ డైనమిక్ యూజర్ అనుభవాన్ని సృష్టించడానికి అనేక ఆధునిక వెబ్ టెక్నాలజీలను ఉపయోగించే ఇంటరాక్టివ్, గేమ్ఫైడ్ వెబ్ అప్లికేషన్ గా నిర్మించబడింది.

లక్షణాలు:

- ఇంటరాక్టివ్ 3D గ్లోబ్: యాదృచ్ఛిక గమ్యస్థానాన్ని ఎంచుకోవడానికి వాస్తవిక 3D గ్లోబ్ ను స్పిన్ చేయండి (Three.js ద్వారా శక్తివంతం చేయబడింది).
- శాటిలైట్ మ్యాప్ వ్యూ: అధిక-రిజల్యూషన్ శాటిలైట్ మ్యాప్ తో మీ గమ్యస్థానానికి స్వయంచాలకంగా ప్లే చేసి జామ్ ఇన్ చేయండి (MapLibre GL JS ద్వారా శక్తివంతం చేయబడింది).
- డైనమిక్ బ్రెజర్ సిస్టమ్:
 - వరల్డ్ వండర్స్ వద్ద అధిక-విలువ నిధులను కనుగొనండి.
 - KL యూనివర్సిటీలో 2000-పాయింట్ జాక్ పాట్ ను కనుగొనండి.
 - అనేక స్థానాలలో హాస్యాస్పదమైన "లూటర్స్" సందేశాలను చూడండి.
- 3-స్పిన్ పరిమితి: ప్రతి ఆటగాడు అత్యధిక స్కోర్ సాధించడానికి ప్రతి గేమ్ కు మూడు స్పిన్లు పొందుతారు.
- డైనమిక్ ఇమేజ్ ఫెచింగ్: బ్రెజర్ కార్డ్లు ప్రతి స్థానానికి ప్రత్యేక చిత్రాలను కలిగి ఉంటాయి, Pixabay API నుండి డైనమిక్ గా పొందబడతాయి మరియు బ్రౌజర్ యొక్క లోకల్ స్టోరేజ్ లో కాష్ చేయబడతాయి.

- పాలిష్ట్ యానిమేషన్స్ UI వోర్కెన్స్ ట్రాన్సిషన్, స్టాగర్డ్ కార్డ్ రివీల్స్ మరియు స్కరింగ్ పై "కాయిన్ బ్లాస్ట్" యానిమేషన్ ను కలిగి ఉంది (Anime.js ద్వారా శక్తివంతం చేయబడింది).
- లోకల్ లీడర్ బోర్డ్: టాప్ కెప్టెన్లను ట్రాక్ చేయడానికి అధిక స్కోర్లు బ్రౌజర్ యొక్క localStorage లో సేవ చేయబడతాయి.
- సీక్రెట్ మోడల్: హోమ్ స్క్రీన్ పై దాచిన మోడల్ జాక్ పాట్ స్థానానికి క్లాసు వెల్లడిస్తుంది.

టెక్నాలజీ స్టాక్:

- ఫ్రంట్ ఎండ్: React 19
- బిల్డ్ టూల్: Vite
- 3D గ్లోబ్: Three.js
- 2D మ్యాప్: MapLibre GL JS
- యానిమేషన్స్ Anime.js
- ఎక్స్టర్నల్ APIs: Plixabay API (స్థాన చిత్రాల కోసం), MapTiler API (శాటిలైట్ మ్యాప్ టైల్స్ కోసం)

ప్రారంభం: లోకల్ కాపీని పొందడానికి మరియు రన్ చేయడానికి ఈ సూచనలను అనుసరించండి.

ముందస్తు అవసరాలు: మీ సిస్టమ్ లో Node.js (ఇందులో npm ఉంటుంది) ఇన్స్టాల్ చేయబడి ఉండాలి.

ఇన్స్టాలేషన్: రిపోజిటరీని క్లోన్ చేయండి:

```
git clone https://github.com/MathewAddala/globetrotterv1.git
```

ప్రాజెక్ట్ కు నావిగేట్ చేయండి:

```
cd globetrotterv1/treasure-hunt
```

డిపెండెన్సీలను ఇన్స్టాల్ చేయండి:

```
npm install
```

ఎన్విరాన్మెంట్ సెటప్: ఈ ప్రాజెక్ట్ సరిగ్గా పనిచేయడానికి API కీలు అవసరం. treasure-hunt రూట్ డైరెక్టరీలో (package.json ఉన్న అదే ఫోల్డర్ లో) కొత్త ఫైల్ ను సృష్టించండి మరియు దానికి .env అని పేరు పెట్టండి.

.env ఫైల్ ను తెరిచి మీ Plixabay API కీని జోడించండి:

```
VITE_PIXABAY_API_KEY="YOUR_API_KEY_HERE"
```

గమనిక: MapTiler API కీ ఈ వెర్షన్ కోసం సోర్స్‌కోడ్‌లో ఇప్పటికే చేర్చబడింది.

అప్లికేషన్‌ను రన్ చేయడం: డెవలప్‌మెంట్ సర్వర్‌ను ప్రారంభించండి:

```
npm run dev
```

ఈ కమాండ్ Vite డెవలప్‌మెంట్ సర్వర్‌ను ప్రారంభిస్తుంది. మీ బ్రౌజర్‌ను తెరిచి మీ టెర్మినల్‌లో చూపిన లోకల్ URL కు నావిగేట్ చేయండి (సాధారణంగా <http://localhost:5173>).

లోకల్ నెట్‌వర్క్‌లో పరీక్షించడం: మీ అదే నెట్‌వర్క్‌లోని ఇతర పరికరాలలో (మొబైల్ ఫోన్ వంటివి) అప్లికేషన్‌ను పరీక్షించడానికి:

1. మీ ల్యాప్‌టాప్ మరియు మొబైల్ పరికరం ఒకే WI-FI నెట్‌వర్క్‌లో ఉన్నప్పుడు మొబైల్ హాట్‌స్పాట్‌కు కనెక్ట్ చేయబడి ఉన్నాయని నిర్ధారించుకోండి.
2. vite.config.js ఫైల్ నెట్‌వర్క్‌యాక్సెస్‌ను అనుమతించడానికి host: true తో ఇప్పటికే కాన్ఫిగర్ చేయబడింది.
3. మీరు npm run dev రన్ చేసినప్పుడు, టెర్మినల్ "నెట్‌వర్క్" URL ను ప్రదర్శిస్తుంది (ఉదా: <http://192.168.1.10:5173>).
4. గేమ్ ఆడటానికి మీ మొబైల్ పరికర బ్రౌజర్‌లో ఈ నెట్‌వర్క్ URL ను తెరవండి.

ప్రాజెక్ట్ డెమో వీడియో:

దిగువ లింక్ ద్వారా మీరు ప్రాజెక్ట్ డెమోను చూడవచ్చు:

<https://drive.google.com/file/d/14H0mRYkzUIVHBRB0jPe1E072YScDhM6i/view>

7. EVENT POSTER

7.1 Project Promotion

To attract participants to our booth at the Samyak Technofest, we designed a dedicated event poster. The goal was to visually communicate the "Treasure Hunt" theme and highlight the specific "Jackpot" hidden at KL University to get students excited about playing.

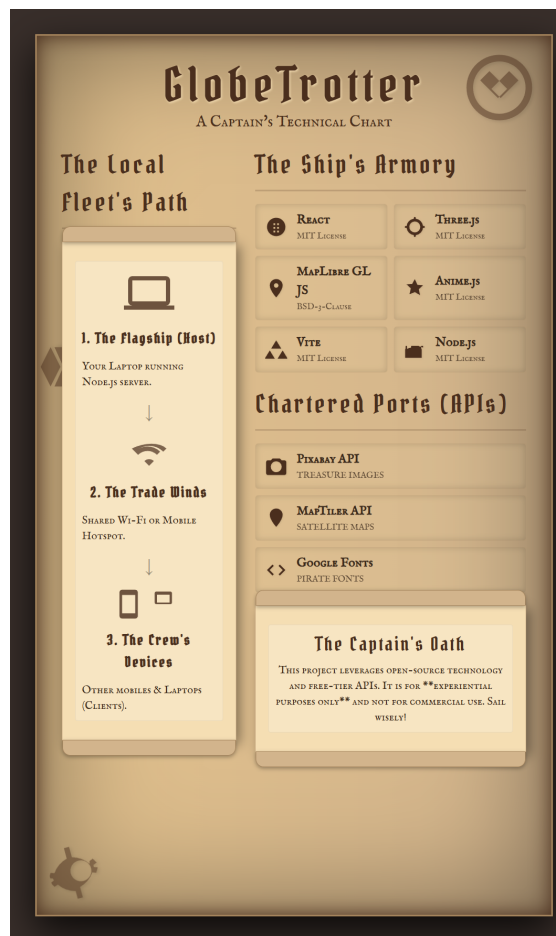


Figure 7.1: The promotional poster displayed at our booth.

8. OPEN SOURCE CONTRIBUTIONS

8.1 Overview

My open source journey involved contributing to repositories with massive user bases. Below is a detailed breakdown of my Pull Requests (PRs), focusing on the technical significance of each change and the outcomes of the review process.

8.2 Repository: TheAlgorithms/Python

Repo Link: <https://github.com/TheAlgorithms/Python>

Stats: ~183k Stars | **PRs:** 6 (3 Open, 3 Closed)

Open PRs (Performance & Correctness)

1. PR #13836: Remove inefficient validation in binary search

Link: <https://github.com/TheAlgorithms/Python/pull/13836>

- **What Changed:** I removed a validation line that checked `if list(sorted) != sorted(sorted)` across 4 files (including `binary_search.py`).
- **Significance:** This was a critical performance fix. The validation step itself had a time complexity of $O(n)$, which completely defeated the purpose of using a Binary Search (which should be $O(\log n)$). For large datasets, my fix ensures the algorithm actually performs as intended.

2. PR #13821: Fix doctest bug in bubble_sort_recursive

Link: <https://github.com/TheAlgorithms/Python/pull/13821>

- **What Changed:** Fixed an incorrect function call in the doctest examples.
- **Significance:** Learners rely on doctests to understand how to run the code. Broken examples damage trust in educational content; this fix ensures the documentation is executable and accurate.

3. PR #13818: fix-factorial-validation

Link: <https://github.com/TheAlgorithms/Python/pull/13818>

- **Significance:** Improved code reliability by adding proper input validation, preventing runtime errors for students studying these implementations.

Closed PRs (Optimization Attempts)

4. PR #13838: Enhanced quicksort with in-place sorting

Link: <https://github.com/TheAlgorithms/Python/pull/13838>

- **What Changed:** I replaced the list comprehension method with an in-place swapping mechanism using the Lomuto partition scheme.
- **Significance:** This was a major memory optimization. It reduced the Space Complexity from $O(n)$ to $O(\log n)$ by eliminating the creation of new lists at every recursive step.
- **Reason for Closure:** The repository currently has over **625 open PRs**. Despite the technical merit, the PR was closed/stale simply because it did not receive maintainer attention amidst the high volume of submissions.

5. PR #13837: Refactor quick_sort (Alternative)

Link: <https://github.com/TheAlgorithms/Python/pull/13837>

- **Reason for Closure:** Similar to PR #13838, this was closed due to a lack of maintainer feedback in a highly active repository.

8.3 Repository: Free Programming Books

Repo Link: <https://github.com/EbookFoundation/free-programming-books>

Stats: ~334k Stars | **PRs:** 3 (All Closed)

1. PR #12968: fix: workflow corrections

Link: <https://github.com/EbookFoundation/free-programming-books/pull/12968>

- **Significance:** I attempted to fix shell command syntax in the GitHub Actions workflows to improve CI/CD reliability.
- **Reason for Closure:** The PR encountered workflow validation check failures due to specific syntax issues in the configuration that conflicted with upstream settings.

2. PR #12970: docs: Restore CONTRIBUTING.md

Link: <https://github.com/EbookFoundation/free-programming-books/pull/12970>

- **Reason for Closure:** Closed due to maintainer preference or being superseded by other documentation updates.

3. PR #12965: Fix broken APL book link

Link: <https://github.com/EbookFoundation/free-programming-books/pull/12965>

- **Reason for Closure:** Likely closed due to a duplicate submission or PR limits, as this is a very high-traffic repository for link fixes.

8.4 Repository: First Contributions

Repo Link: <https://github.com/firstcontributions/first-contributions>

Stats: ~47k Stars | **PRs:** 1 (Merged)

1. PR #107038: Update Contributors.md (Merged)

Link: <https://github.com/firstcontributions/first-contributions/pull/107038>

- **Significance:** This was my first successful merge. It served as the practical validation of my Git knowledge, proving I could successfully navigate the **Fork** → **Branch** → **Commit** → **PR** → **Merge** lifecycle.

8.5 Evidence of Contributions

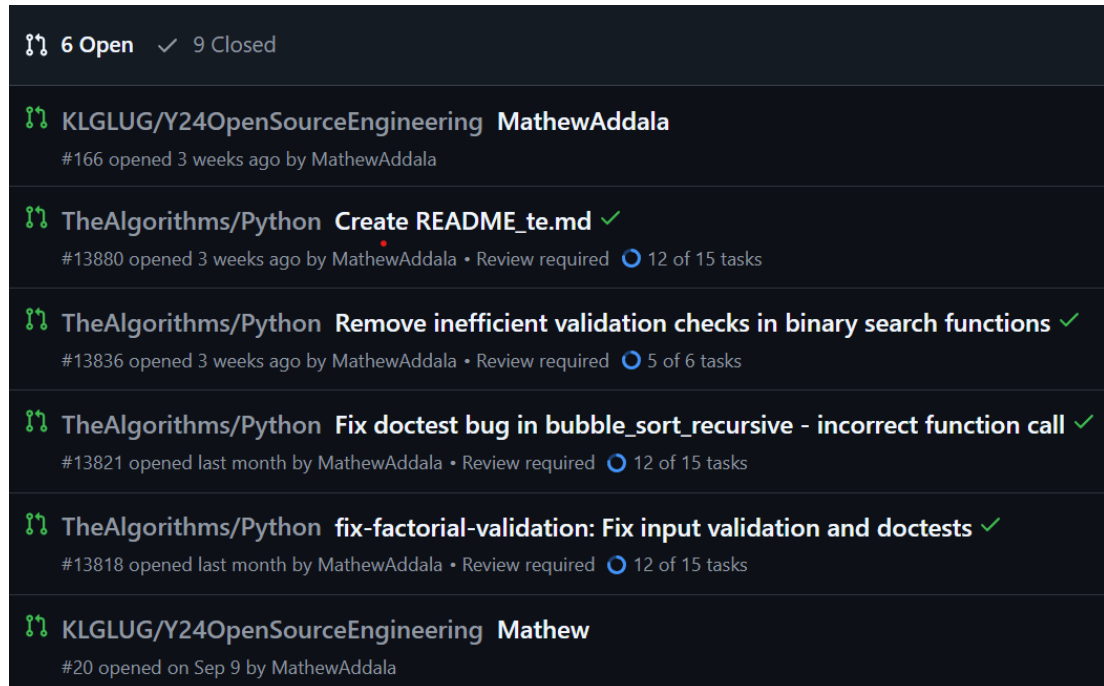


Figure 8.1: Screenshot of my Open Pull Requests on TheAlgorithms/Python.

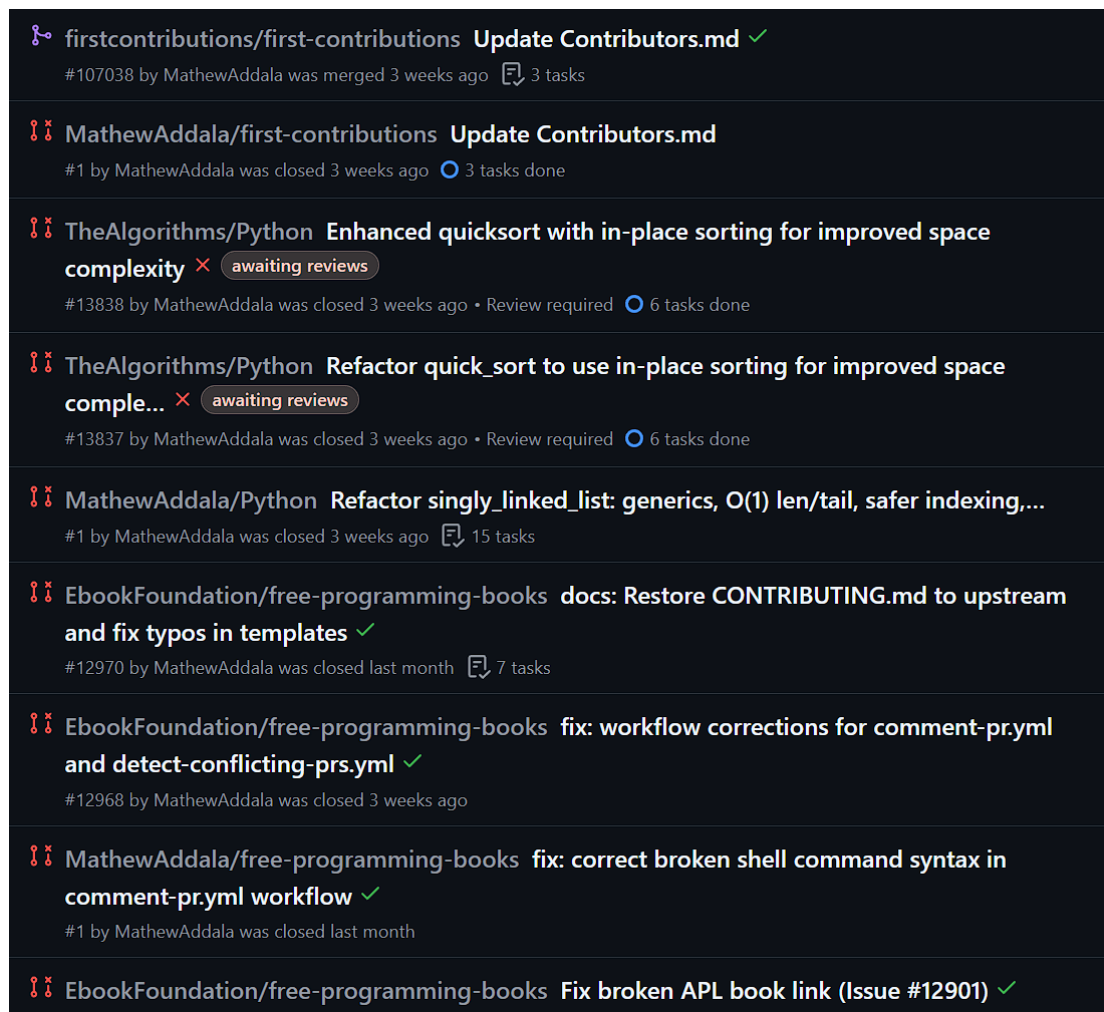


Figure 8.2: Verification of my merged contribution to the First Contributions repository and the closed PRs.

9. CAMPUS OPEN SOURCE INITIATIVES

9.1 Project: KLeats

Beyond global repositories, I actively engaged with the internal open source ecosystem at **KL University**. My primary focus was on **KLeats**, a campus project maintained by the university's open source engineering team.

9.2 The Proposal: Decentralized Delivery Governance

I drafted and raised a detailed issue proposing a new operational model inspired by blockchain governance. The objective was to transition KLeats into a community-driven Open Delivery system.

Key Components of the Proposal:

1. Student-Led Governance (Voting Mechanism)

- I proposed a voting system where students can signal their intent to participate as delivery partners.
- This creates a verified ledger of interested users, granting them specific “privileges” or incentives for signing up to the open delivery network.

2. Open Delivery Logistics

- Instead of dedicated personnel, interested students near a food source can view pending orders.
- Any student heading towards the destination can volunteer to pick up the order, creating a peer-to-peer logistics network.

3. The “Kill Period” Failsafe

- To prevent delays, I introduced the concept of a **Kill Period** (a strict time window).

- **Logic:** If no student volunteer claims the order within this time limit, the system automatically reverts the status, notifying the customer to pick up the food themselves. This ensures no order remains unclaimed indefinitely.

9.3 Significance

This proposal demonstrates how decentralized concepts can be applied to real-world campus problems. By introducing the “Kill Period,” the system balances community reliance with service guarantees, ensuring students get their food regardless of volunteer availability.

9.4 Evidence of Proposal

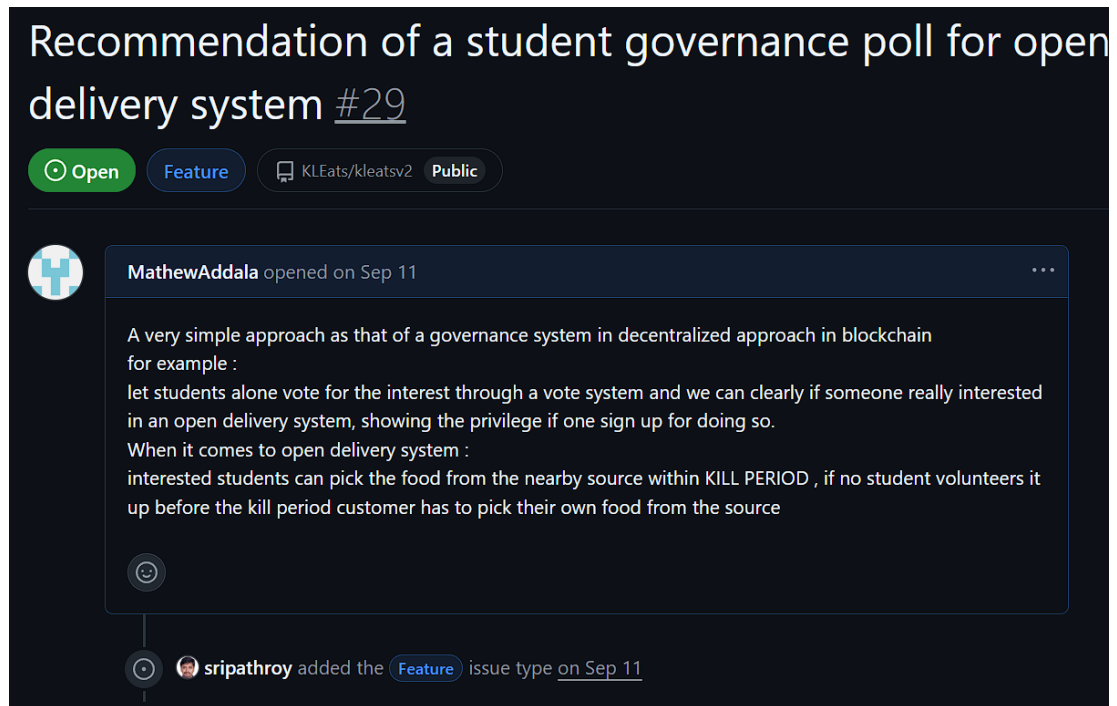


Figure 9.1: Screenshot of the issue/proposal raised on the KLeats repository.

10. PUBLIC OUTREACH & BLOGS

Below are the links to the public documentation and social media outreach regarding my Open Source Journey and the GlobeTrotter project.

10.1 LinkedIn Showcase: Samyak Event

This post highlights the technical launch of GlobeTrotter at the Samyak Technofest (KL University).

Link:

https://www.linkedin.com/posts/mathew-addala-105511377_softwareengineering-reactjs-threejs-activity-7399008596379852801-XCbE?utm_source=shareutm_medium=member_desktoprcm=ACoAAF0ohFUBuLfAFZyhUFkfo9Y4-OTriPzxAlQ

10.2 Technical Blog: My Open Source Journey

My open source journey blog.

Link:

<https://medium.com/@kl2400030483/my-opensource-journey-61f9d5d25723>

10.3 LinkedIn Announcement: Blog Release

This post shares the blog with the developer community to encourage other students to start contributing to open source.

Link:

https://www.linkedin.com/posts/mathew-addala-105511377_my-opensource-journey-activity-7399349482796171264-tqY7?utm_source=shareutm_medium=member_desktoprcm=ACoAAF0ohFUBuLfAFZyhUFkfo9Y4-OTriPzxAlQ