KL (DEEMED TO BE UNIVERSITY) | HTE HONORS THROUGH EXPERIENTIAL LEARNING

EXPERIENTIAL LEARNING & GLOBAL ENGAGEMENT

OPEN SOURCE ENGINEERING

End Semester-Evaluation Report

Name: Vijayshree Parakh

Roll Number: 2400030530

Course: B.Tech

Specialization: HTE

University: KL University

Contents

1 Linux Distribution Used

For this course, I worked extensively with the **Ubuntu Linux distribution**, one of the most widely used and beginner-friendly Linux environments. Ubuntu's stability, extensive documentation, and rich software ecosystem made it an ideal platform for learning open-source development workflows.

Experience with Ubuntu

Throughout the semester, I developed a deeper understanding of how Linux operates at both user and system levels. Key areas of learning include:

- **Mastering Terminal Navigation**
  I became proficient in directory traversal and file management using commands such as `cd`, `ls`, `pwd`, and `mkdir`. This reduced my dependence on graphical interfaces and strengthened my command-line confidence.

- **Using Command-Line Tools for Development**
  I worked with essential utilities like `nano` and `vim` for editing, `ps` and `top` for monitoring processes, and `df` and `du` for analyzing disk usage. These tools taught me how Linux manages processes and resources internally.

- **Software Installation Through APT**
  I frequently used the APT package manager to update, upgrade, and install software using commands such as:
  `sudo apt update`, `sudo apt upgrade`, `sudo apt install <package>`
  This helped me understand dependency management and software repositories.

- **Working with Git on Linux**
  I configured SSH keys, worked with remote repositories, created commits, resolved merge conflicts, and used branches efficiently. This strengthened my version control practices.

- **Managing File Permissions and User Access**
  Using `chmod`, `chown`, and `groups`, I learned how Linux enforces permission models across files, users, and groups—essential knowledge while configuring Docker.

- **Fixing Docker Daemon Permission Issues**
  I resolved "permission denied" errors by adding my user to the Docker group using:
  `sudo usermod -aG docker $USER`
  This introduced me to daemon access control and Linux security policies.

- **Exploring Services and Logs**
  Tools like `systemctl` and `journalctl` helped me understand systemd-based service management and troubleshooting.

- **General System Exploration**
  I practiced shell scripting, networking commands, and system utilities, which enhanced my overall proficiency with Linux.

2 Encryption and GPG

In this course, I had the opportunity to work hands-on with **GPG (GNU Privacy Guard)**, a powerful open-source implementation of the OpenPGP standard. GPG is widely used for secure communication, file encryption, digital signatures, and identity verification. Through this activity, I gained not only practical experience but also a clearer conceptual understanding of modern cryptography and its importance in open-source software ecosystems.

Understanding GPG

GPG works on a **public-key cryptography** model. Each user generates a pair of keys:

- **Public Key** — shared with others for encrypting data or verifying signatures.
- **Private Key** — kept confidential and used for decrypting data or creating digital signatures.

This model ensures secure and trustworthy communication, even over untrusted networks.

Steps Followed

1. Installed GPG on Ubuntu using:
   ```
   sudo apt install gnupg
   ```
2. Generated a secure key pair using:
   ```
   gpg --full-generate-key
   ```
   While generating the key, I selected:
   - RSA and RSA (default)
   - 3072-bit key size for stronger security
   - A suitable expiration date
   - My name and email for identity binding
3. Verified the key details using:
   ```
   gpg --list-keys
   ```
4. Exported the public key to share with others:
   ```
   gpg --export -a "Your Name" > mypubkey.asc
   ```
5. Created a revocation certificate as a safety measure:
   ```
   gpg --gen-revoke <keyID>
   ```

Practical Usage Learned

During the activity, I performed several real-world operations such as:

- Encrypting files for secure transfer.
- Decrypting messages using my private key.

- Signing files to prove authorship.
- Verifying signatures of others to check authenticity.

These exercises gave me confidence in using encryption tools responsibly.

Learning Outcome

This activity helped me understand the broader significance of encryption in the digital world. Specifically, I learned about:

- The principles of confidentiality, integrity, and authenticity.
- How key pairs are generated, maintained, and revoked.
- The importance of protecting private keys.
- How digital signatures help prevent tampering.
- The role of encryption in secure open-source development and communication.

Overall, working with GPG strengthened my understanding of cybersecurity fundamentals and improved my ability to use professional encryption tools in real development environments.

3 Sending an Encrypted Email

As a practical application of GPG, I encrypted and signed an email message to demonstrate my understanding of secure communication. This task helped me apply cryptographic concepts in a real-world scenario, where privacy and authenticity are essential.

Process

1. **Imported the recipient's public key.**
   Before sending an encrypted message, I ensured that the recipient's public key was available in my keyring using:
   ```
   gpg --import publickey.asc
   ```
2. **Verified the key's authenticity.**
   I checked the fingerprint of the imported key to confirm that it belonged to the correct person and that it had not been tampered with.
3. **Encrypted and signed the message.**
   I used the following command to encrypt the message for the intended recipient and digitally sign it using my private key:
   ```
   gpg --encrypt --sign -r <recipient email> message.txt
   ```
   This generated an encrypted output file (usually `message.txt.gpg`) that only the recipient could decrypt using their private key.

4. **Sent the encrypted output via email.**
   The encrypted file was attached and sent over a regular email service. Even if intercepted, the message would remain unreadable without the appropriate private key.

5. **Recipient decrypted and verified the message.**
   The recipient could use:
   `gpg --decrypt message.txt.gpg`
   This also confirmed my digital signature, ensuring that the message truly came from me and had not been altered in transit.

Learning Outcome

Through this exercise, I understood the complete workflow of secure email communication. I learned how encryption preserves confidentiality, digital signatures ensure sender authenticity, and key verification prevents impersonation attacks. This activity also highlighted how GPG plays an important role in secure workflows within open-source projects and professional communication environments.

Overall, this task deepened my understanding of secure communication channels and taught me how encryption tools can be used responsibly to protect sensitive information.

4 Privacy Tools from PRISM-BREAK

To strengthen awareness about digital privacy, I explored various privacy-focused tools listed on the **PRISM-BREAK** platform, which promotes open-source alternatives to mainstream surveillance-oriented applications.

Five Tools Explored

1. **Signal** – Secure messaging with strong end-to-end encryption.

2. **ProtonMail** – Encrypted, privacy-respecting email services.

3. **Tor Browser** – Anonymized browsing using onion routing.

4. **DuckDuckGo** – A search engine that avoids tracking and profiling.

5. **KeePassXC** – Open-source offline password manager.

These tools helped me understand tracking prevention mechanisms, data sovereignty, and how open-source software enhances online security.

5 Open Source License Used

For my open-source contributions, I primarily worked with the **MIT License**, a widely adopted permissive license. Its simplicity and minimal restrictions make it ideal for educational projects and rapid innovation.

Why MIT License?

- Easy to understand and legally lightweight.
- Allows reuse, modification, and redistribution.
- Encourages collaboration without strict attribution rules.

Working with the MIT License helped me understand how open-source licensing works, why licenses are important, and how they affect the way developers share code. It also showed me how permissive licenses can accelerate innovation by allowing others to freely extend and improve the work

6 Self-Hosted Server: Homer Dashboard

As part of the Open Source Engineering course, I deployed my own self-hosted server using the Homer Dashboard. This project introduced me to real-world server deployment, containerization, and service management using Docker. It also helped me understand how self-hosting can replace cloud services and give complete control over data and configuration.

About Homer

Homer is a lightweight, static dashboard for managing and organizing links to various self-hosted services. It acts as a personalized homepage where all important tools, services, and resources can be managed from a single interface. Its key features include:

- YAML-based configuration
- Fully static HTML/JS UI
- Designed for Docker
- Customizable themes and layouts
- Minimal hardware requirements

Installation Using Docker

```
docker run -d --name homer -p 8080:8080 \
  --mount type=bind,source="$HOME/homer-assets",target=/www/assets \
  --restart=unless-stopped b4bz/homer:latest
```

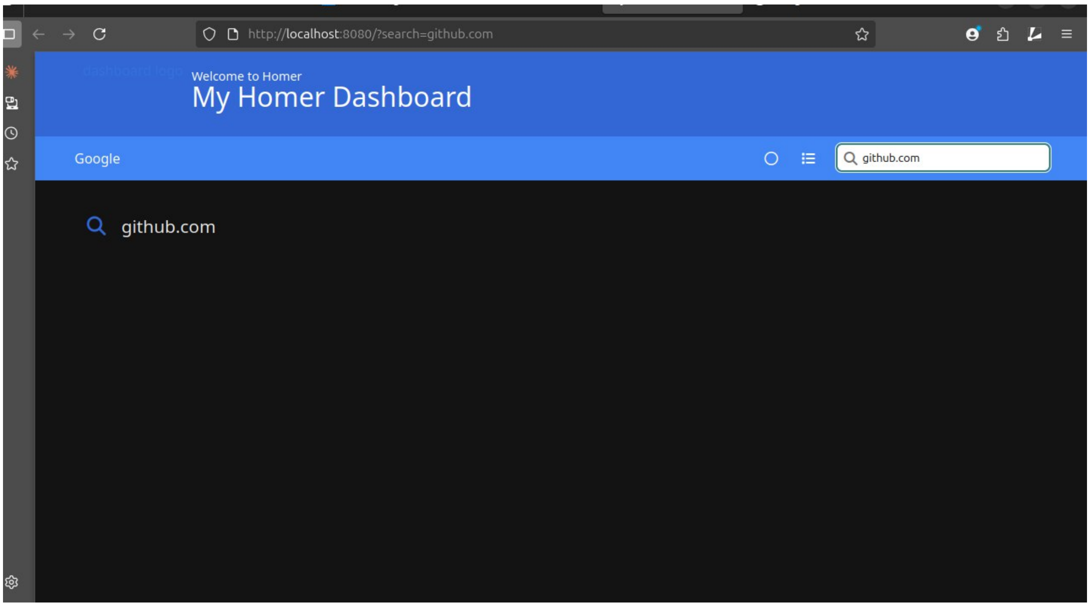
Access

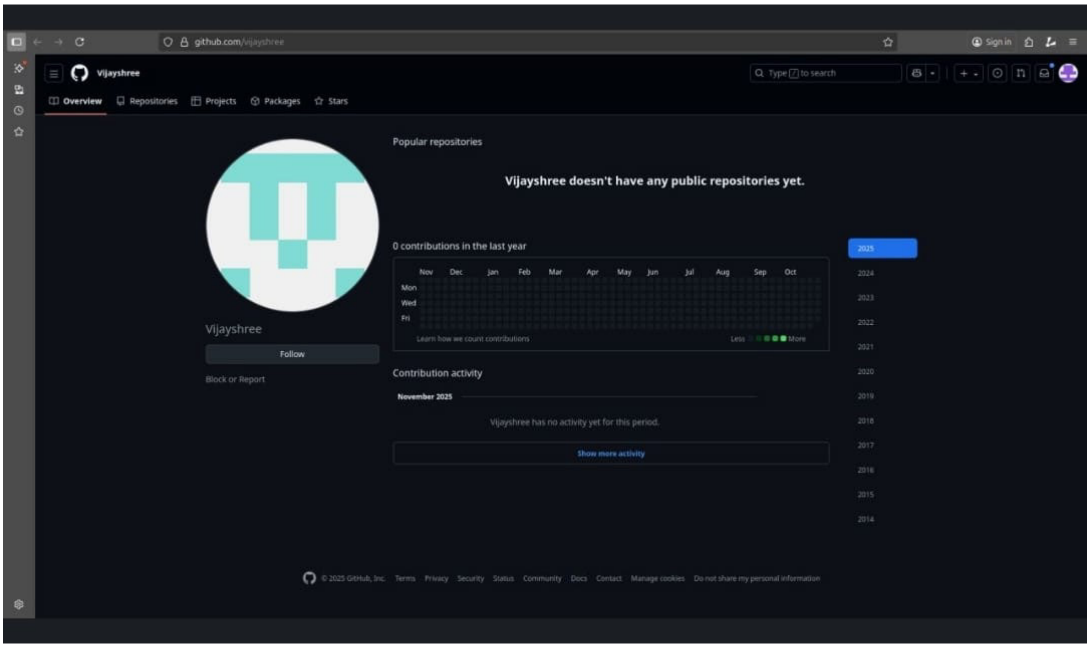
Dashboard available at:
`https://localhost:8080`

Poster

Screenshots

Below are the screenshots of our Homer dashboard:





Popular repositories

Vijayshree doesn't have any public repositories yet.

0 contributions in the last year

Contribution activity

November 2025

Vijayshree has no activity yet for this period.

7 Open Source Contributions (PRs)

Throughout this course, I submitted multiple pull requests across diverse repositories, gaining significant experience in collaborative development, documentation improvement, debugging, and feature development.

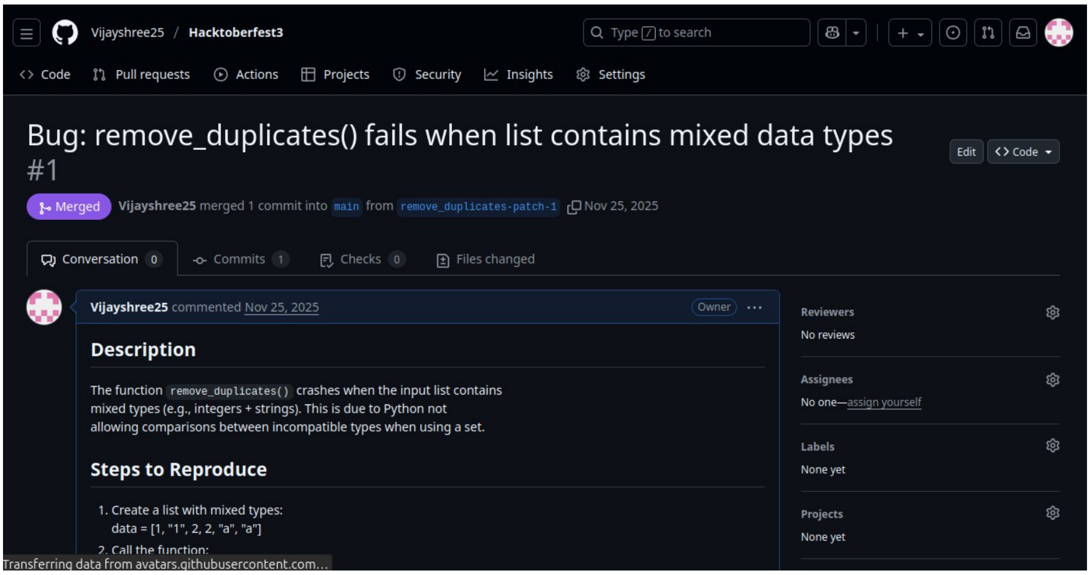
PR Summary

1. Bug Fix: remove_duplicates() for Mixed Data Types – **Merged**
2. Line-count Utility for Log Analysis – **Merged**
3. Email Validation Utility – **Merged**
4. Remove Duplicates from List Utility – **Merged**
5. String Reversal Utility – **Merged**
6. Bitwise OR Calculation Fix for Even Numbers – **Merged**

Detailed PR Descriptions

1. Bug Fix: remove_duplicates() for Mixed Data Types

Repository: `Vijayshree25/Hacktoberfest3`
This PR addressed a bug in the `remove_duplicates()` function, which failed when the input list contained mixed data types (e.g., integers and strings). I implemented type validation and modified the function to handle mixed data gracefully while maintaining the original order. This strengthened my understanding of defensive programming in Python.The function should handle mixed data types gracefully and return a cleaned list without duplicates, maintaining the original order.



Bug: remove_duplicates() fails when list contains mixed data types #1

Merged Vijayshree25 merged 1 commit into `main` from `remove_duplicates-patch-1` Nov 25, 2025

Vijayshree25 commented Nov 25, 2025

Description

The function `remove_duplicates()` crashes when the input list contains mixed types (e.g., integers + strings). This is due to Python not allowing comparisons between incompatible types when using a set.

Steps to Reproduce

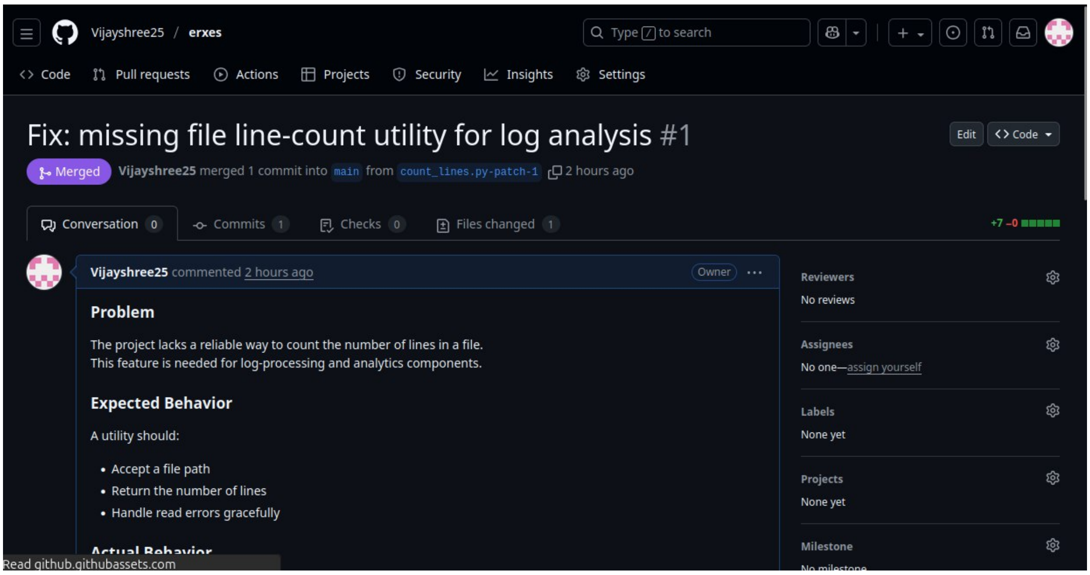
1. Create a list with mixed types:
   data = [1, "1", 2, 2, "a", "a"]
2. Call the function:

Transferring data from avatars.githubusercontent.com...

2. Line-count Utility for Log Analysis

Repository: `Vijayshree25/erxes`

I created `count_lines.py` to safely count the number of lines in a file. This utility handles read errors gracefully and uses a generator-based approach for efficiency. This PR improved code reuse and reliability in log-processing modules.



Fix: missing file line-count utility for log analysis #1

Vijayshree25 merged 1 commit into `main` from `count_lines.py-patch-1` 2 hours ago

Vijayshree25 commented 2 hours ago

Problem

The project lacks a reliable way to count the number of lines in a file.
This feature is needed for log-processing and analytics components.

Expected Behavior

A utility should:

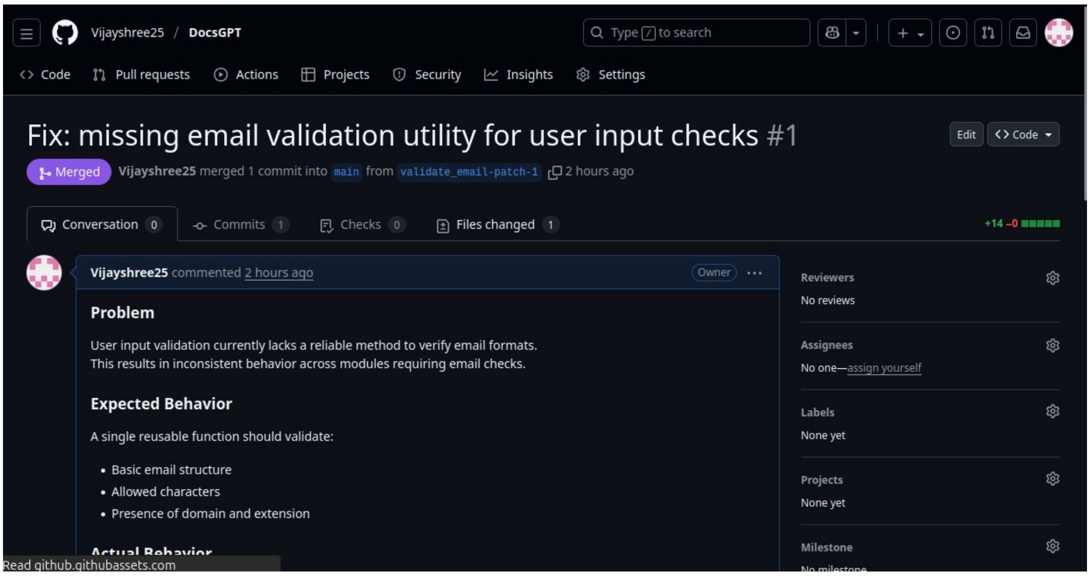
- Accept a file path
- Return the number of lines
- Handle read errors gracefully

Actual Behavior

3. Email Validation Utility

Repository: `Vijayshree25/DocsGPT`

Added `validate_email.py`, a reusable function to validate email inputs using a regex pattern. Non-string inputs are safely handled, and the function returns a boolean for easy integration. This ensured consistent input validation across the project.



Fix: missing email validation utility for user input checks #1

Vijayshree25 merged 1 commit into `main` from `validate_email-patch-1` 2 hours ago

Vijayshree25 commented 2 hours ago

Problem

User input validation currently lacks a reliable method to verify email formats.
This results in inconsistent behavior across modules requiring email checks.

Expected Behavior

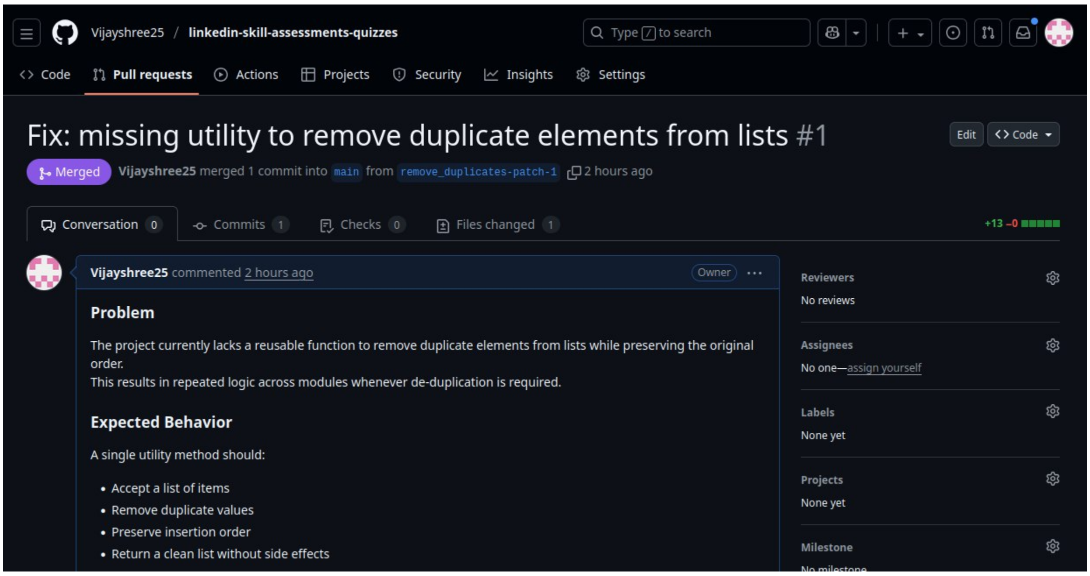
A single reusable function should validate:

- Basic email structure
- Allowed characters
- Presence of domain and extension

Actual Behavior

4. Remove Duplicates from List Utility

Repository: `Vijayshree25/linkedin-skill-assessments-quizzes`
In this PR, I developed a utility function `remove_duplicates.py` that efficiently removes duplicate elements from lists while preserving their original order. The function uses a combination of a set to track seen elements and a loop to maintain sequence, making it suitable for lists with integers, strings, or mixed types. I tested this function extensively with edge cases, including empty lists, lists with repeated item.



Fix: missing utility to remove duplicate elements from lists #1

Vijayshree25 merged 1 commit into `main` from `remove_duplicates-patch-1` 2 hours ago

Vijayshree25 commented 2 hours ago

Problem

The project currently lacks a reusable function to remove duplicate elements from lists while preserving the original order.
This results in repeated logic across modules whenever de-duplication is required.

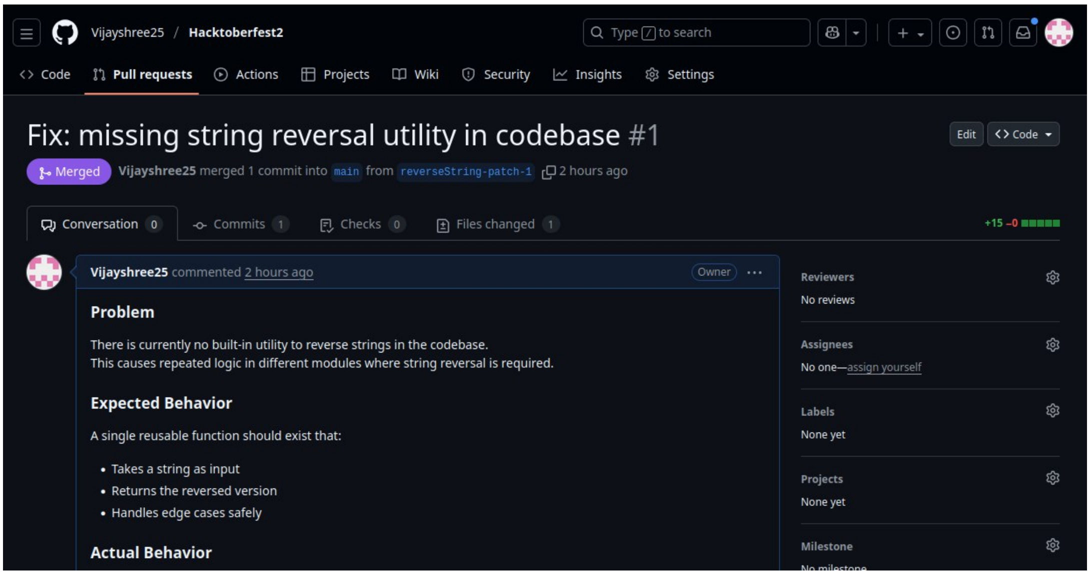
Expected Behavior

A single utility method should:

- Accept a list of items
- Remove duplicate values
- Preserve insertion order
- Return a clean list without side effects

5. String Reversal Utility

Repository: `Vijayshree25/Hacktoberfest2`
I implemented a new JavaScript utility, `reverseString.js`, which safely reverses strings while handling edge cases like non-string inputs by returning an empty string. This PR replaced repetitive string-reversal logic scattered across different modules, ensuring **consistency and code reuse**. I focused on writing readable, maintainable code and added example usage to demonstrate its functionality. Through this PR, I gained a deeper understanding of **string manipulation, defensive programming.



Fix: missing string reversal utility in codebase #1

Vijayshree25 merged 1 commit into `main` from `reverseString-patch-1` 2 hours ago

Vijayshree25 commented 2 hours ago

Problem

There is currently no built-in utility to reverse strings in the codebase.
This causes repeated logic in different modules where string reversal is required.

Expected Behavior

A single reusable function should exist that:

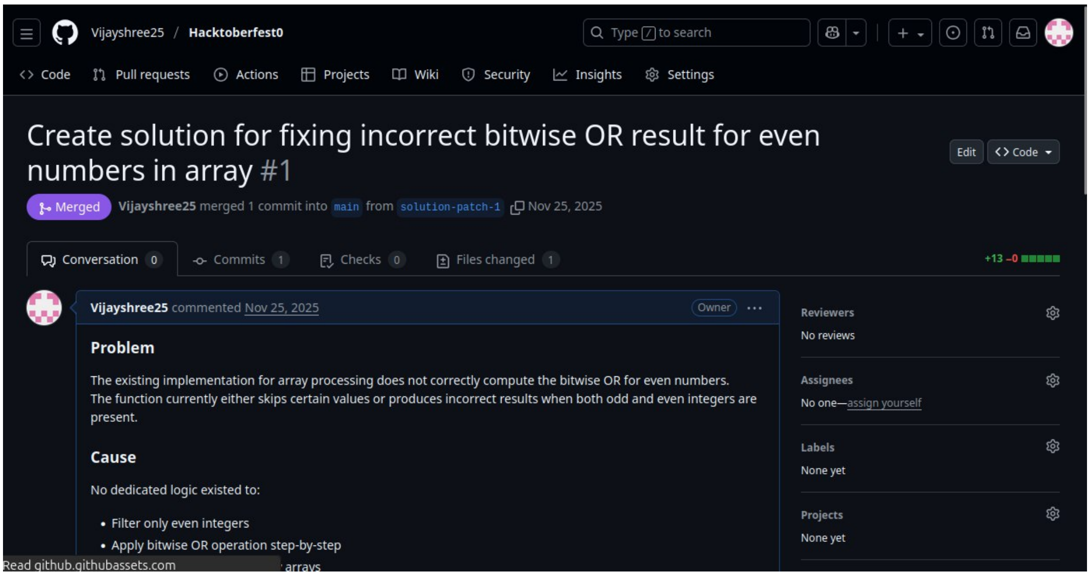
- Takes a string as input
- Returns the reversed version
- Handles edge cases safely

Actual Behavior

6. Bitwise OR Calculation Fix for Even Numbers

Repository: `Vijayshree25/Hacktoberfest0`

This PR resolved a logical issue in the array processing function that incorrectly computed the bitwise OR for even numbers. I implemented a step-by-step solution that iterates through the array, selects only even integers, and applies the bitwise OR operation sequentially. Edge cases, such as empty arrays, are handled gracefully by returning 0. Through this PR, I strengthened my knowledge of **bitwise operations, array manipulation, and writing optimized, reusable functions**.



Create solution for fixing incorrect bitwise OR result for even numbers in array #1

Merged Vijayshree25 merged 1 commit into `main` from `solution-patch-1` Nov 25, 2025

Vijayshree25 commented Nov 25, 2025

Problem

The existing implementation for array processing does not correctly compute the bitwise OR for even numbers. The function currently either skips certain values or produces incorrect results when both odd and even integers are present.

Cause

No dedicated logic existed to:

- Filter only even integers
- Apply bitwise OR operation step-by-step

arrays

8 LinkedIn Posts

Below are the LinkedIn posts created as part of the coursework:

1. Self-Hosting Project Post:
   https://www.linkedin.com/posts/junelia-samaddar-9b7994312_opensource-docker-gith
   utm_source=share&utm_medium=member_desktop&rcm=ACoAAFj3rhQBMSUjT83x9447QxVcwVuUo

2. First Pull Request Celebration Post:
   https://www.linkedin.com/posts/vijayshree-parakh-22p02_opensource-hacktoberfest-
   Vvn?utm_source=share&utm_medium=member_desktop&rcm=ACoAAFj3rhQBMSUjT83x9447QxVcw

3. Blog Article:
   https://www.linkedin.com/posts/vijayshree-parakh-22p02_new-os-new-skills-new-gro
   utm_source=share&utm_medium=member_desktop&rcm=ACoAAFj3rhQBMSUjT83x9447QxVcwVuUo