OST Project Report

Name	**PRN**
Aditya Basak	23070123007
Anshuman Singh	23070123021
Shresht	23070123076
Lakshay Tanwar	23070123079

Title: Development of an Automated Currency Converter using Python and Flask API

Introduction:

This project demonstrates the design and implementation of a **Currency Converter** using the **Flask web framework** and the **Frankfurter API**, enabling real-time exchange rate conversion between world currencies. When a user sends a request through the API, the Flask backend fetches live currency data, calculates the converted amount, and returns the result in structured JSON format. The system ensures accuracy, eliminates manual conversions, and offers a reliable, automated alternative to traditional exchange calculators. The application, developed in Python using Flask, Requests, and Pandas libraries, utilizes RESTful API principles and file handling for maintaining conversion history. This project serves as a cost-effective educational tool for learning web development, financial data processing, and backend integration techniques.
This project illustrates the utilization of **Flask**, **Frankfurter API**, and **GitHub** for efficient data retrieval, collaboration, and project management in creating a real-time Currency Converter application. Flask was used to manage routes, handle HTTP requests, and deliver JSON responses, while the Frankfurter API provided authentic live exchange rate data for accurate conversions. Git and GitHub were employed to track changes in the source code, maintain version control, and allow parallel development of backend logic, documentation, and testing.

Each team member worked within their own development branch — such as API design, data handling, and testing modules — and merged their changes into the main branch after review and testing. This ensured a clean, transparent, and traceable development workflow where all updates were properly documented and verified before integration.

By combining Flask with Git-based collaboration and external APIs, the team achieved a well-structured, efficient, and scalable backend system. The repository now preserves the complete version history of the project, from the initial prototype to the final tested version, which can be further enhanced, deployed, or integrated with front-end interfaces in the future.

1.Tools: Git and Github

2. Programming language: Python/Flask API

3. Contributions:

S.No	Name	role/contribution	Github profile
1	Aditya Basak	Coding and Debugging with error handling	https://github.com/theadityabasak
2	Anshuman Singh	API request enhancements and error handling	https://github.com/anshumansingh
3	Shresht	README file and Integration of API	https://github.com/Shresht44
4	Lakshya Tanwar	Flask API Integration	https://github.com/Lak5hay

4.Challenges Faced and Solved:

1.API Connection Errors:
Faced issues with unstable internet and incorrect API endpoints.
Solved by adding timeout handling and validating API responses using `try-except` *blocks.*

2.Invalid Input Parameters:
Missing or incorrect currency codes caused request failures.
Solved by adding input validation and structured error messages in JSON format.

3.Data Accuracy:
Early conversions returned outdated or partial rates.
Solved by fetching live data from the Frankfurter API and verifying each response before calculation.

4.File Handling Errors:
Writing to the CSV log caused duplication and access issues.
Solved by using Pandas append mode and timestamp-based unique entries.

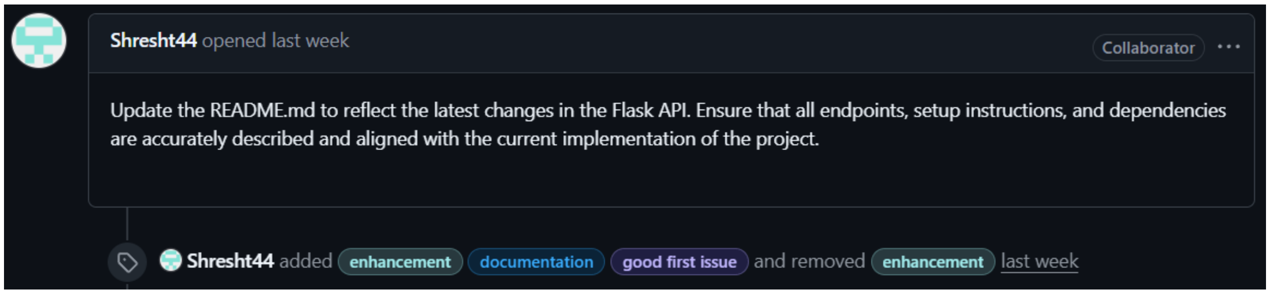
5.Version Control Conflicts:
Multiple contributors caused merge conflicts during updates.
Solved by maintaining separate Git branches and merging through reviewed pull requests

5. Pushing the project to github

a. Github repository link: Currency converter Github Link

Issues:

1) Label: enhancement and documentation - README.md file update

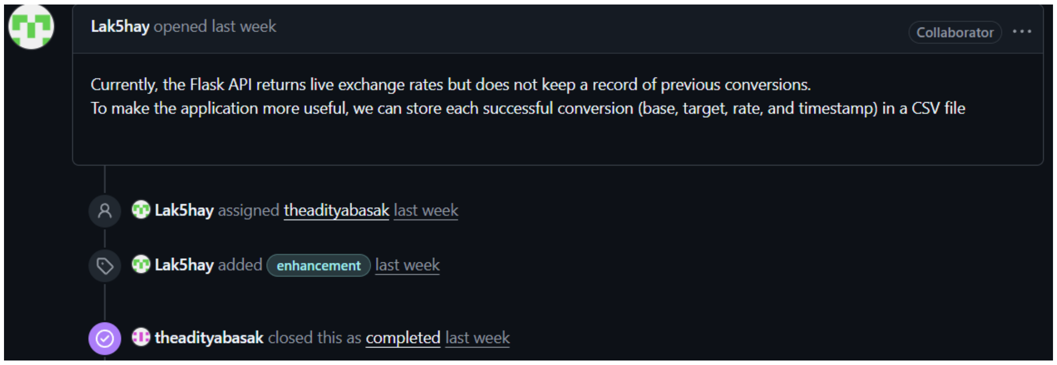


Shresht44 opened last week — Collaborator

Update the README.md to reflect the latest changes in the Flask API. Ensure that all endpoints, setup instructions, and dependencies are accurately described and aligned with the current implementation of the project.

Shresht44 added enhancement documentation good first issue and removed enhancement last week

2) Label: enhancement – Flask API integration



Lak5hay opened last week — Collaborator

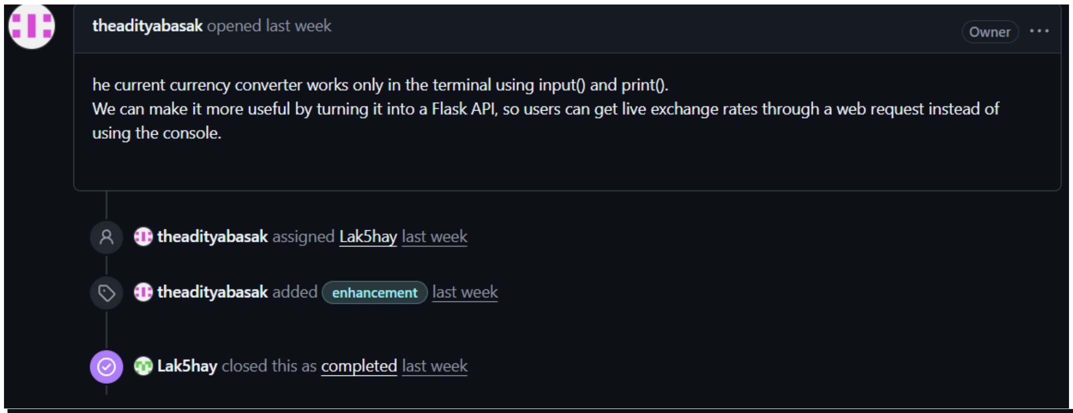
Currently, the Flask API returns live exchange rates but does not keep a record of previous conversions.
To make the application more useful, we can store each successful conversion (base, target, rate, and timestamp) in a CSV file

Lak5hay assigned theadityabasak last week

Lak5hay added enhancement last week

theadityabasak closed this as completed last week

3) Label: enhancement/owner – Code Implementation



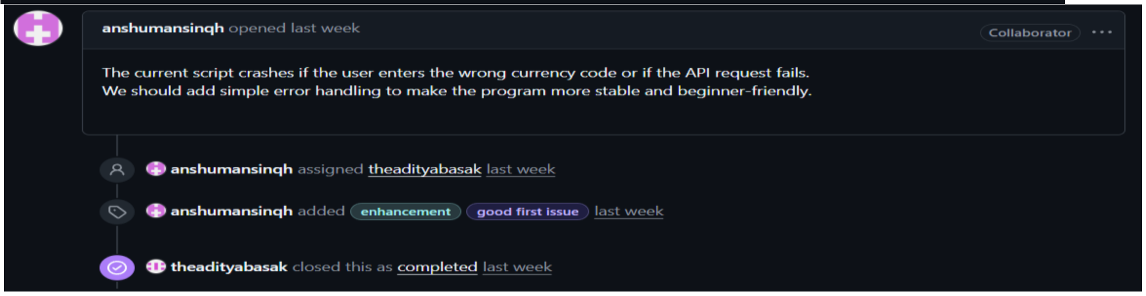
theadityabasak opened last week — Owner

he current currency converter works only in the terminal using input() and print().
We can make it more useful by turning it into a Flask API, so users can get live exchange rates through a web request instead of using the console.

theadityabasak assigned Lak5hay last week

theadityabasak added enhancement last week

Lak5hay closed this as completed last week



anshumansingh opened last week — Collaborator

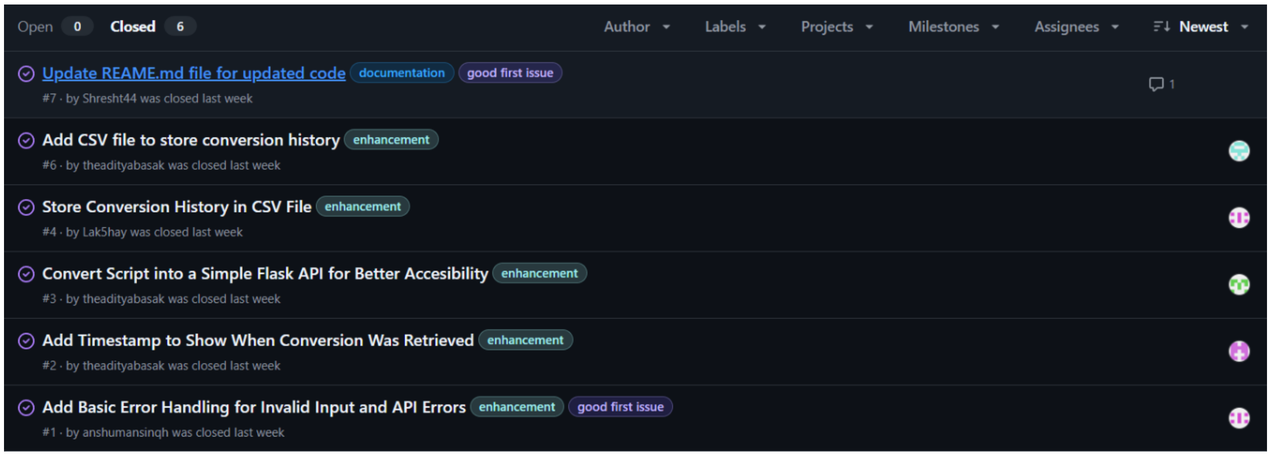
The current script crashes if the user enters the wrong currency code or if the API request fails.
We should add simple error handling to make the program more stable and beginner-friendly.

anshumansingh assigned theadityabasak last week

anshumansingh added enhancement good first issue last week

theadityabasak closed this as completed last week

6. Commits:



Open 0 Closed 6

Author Labels Projects Milestones Assignees Newest

Update REAME.md file for updated code documentation good first issue
#7 · by Shresht44 was closed last week

Add CSV file to store conversion history enhancement
#6 · by theadityabasak was closed last week

Store Conversion History in CSV File enhancement
#4 · by Lak5hay was closed last week

Convert Script into a Simple Flask API for Better Accesibility enhancement
#3 · by theadityabasak was closed last week

Add Timestamp to Show When Conversion Was Retrieved enhancement
#2 · by theadityabasak was closed last week

Add Basic Error Handling for Invalid Input and API Errors enhancement good first issue
#1 · by anshumansinqh was closed last week

README File: It contains all the essential information related to the Currency Converter project, explaining its objective, functionality, and implementation. It helps users understand the purpose of the project, how real-time currency conversion is achieved using Flask and the Frankfurter API, and the overall working process of the system.

Git commands used:

- **git init** – Initializes a new Git repository in the project folder.
- **git add .** – Adds all modified and new files to the staging area before committing.
- **git commit -m "message"** – Saves the staged changes with a descriptive commit message.
- **git push origin main** – Uploads local commits to the remote repository (GitHub).
- **git pull origin main** – Fetches and merges the latest updates from the remote repository into the local copy.

7. Conclusion:

The **Currency Converter using Flask API** project successfully demonstrates the development of a real-time currency conversion system using Python and modern web technologies. By integrating the **Frankfurter API**, the project is capable of fetching live exchange rates and performing accurate conversions between global currencies.

The application showcases the effective use of **Flask** for building RESTful APIs, **Requests** for API communication, and **Pandas** for data storage and management. Through this project, essential concepts such as backend development, API handling, and data logging were implemented efficiently.

Overall, the project achieved its objective of providing a lightweight, reliable, and easily extendable platform for real-time currency conversion. It serves as a strong foundation for future improvements, such as adding a graphical interface, authentication features, or multi-currency support, making it a valuable learning and practical implementation of web-based automation using Python.

.