**Book Recommendation System**

# Introduction:

This documentation details a Python-based project designed to automate the extraction and recommendation of book data. It consists of two main components: a data scraping script using Selenium for extracting book details from a designated website, and a Streamlit-based recommendation system that suggests books based on user preferences using advanced text processing techniques.:

* Book Data Scraping: A script to scrape book data using Selenium from the TIME's collection of must-read books of 2023.
* Book Recommendation System: A Streamlit application that provides book recommendations based on user preferences, utilizing TF-IDF for text vectorization and cosine similarity for finding the most similar books.

# Requirements:

* Python: Python 3
* Libraries: pandas, selenium, Streamlit, scikit-learn
* Chrome web driver: Version depends on your chrome version

## Installation Commands:



Figure : Libraries Installation

# Code Overview:

* Scraper.py: Utilizes Selenium WebDriver to navigate to "https://time.com/collection/must-read-books-2023/", handling pop-ups and extracting book titles, authors, descriptions, and image URLs.
* Book Recommendation App (Streamlit): Processes the scraped data to create a feature matrix using TF-IDF vectorization. It then uses cosine similarity to provide book recommendations based on user inputs such as genre, author, and keywords.

# Usage:

* **Running Scarper.py**

**A black background with white text

Description automatically generated**

Figure : Running Scraper File

Ensure you have a compatible WebDriver for Selenium.

* **Running app.py**

****

Figure : Running App

After setting up the environment with required libraries.

# Testing File by pytest:

This test file contains test cases for the recommendation system implemented in the “app” module.

## How it works:

The test file contains several test functions that evaluate different aspects of the recommendation system implementation. These test functions ensure that the system behaves as expected and meets the specified requirements.

* **“test\_load\_data”:** Checks if the data loading function works correctly and applies preprocessing steps as expected.
* **“test\_compute\_cosine\_similarity”:** Tests the correctness of the cosine similarity matrix computation.
* **“test\_recommend\_books”:** Evaluates the recommendation function by verifying that it returns non-empty recommendations with the required columns. –
* **“test\_recommend\_books\_with\_empty\_input”:** Tests the handling of an empty DataFrame input in the recommendation function.

## How to run:

To run the tests, ensure that the necessary dependencies are installed, and the test file is located in the appropriate directory. First install pytest using command:



Figure : Install pytest

Then, execute the following command in the terminal:

A black and white sign with white text

Description automatically generated

Figure : Run test

# Code Explanation:

* Scraper.py: Details on Selenium operations, pop-up handling, data extraction, and saving to CSV.
* Recommendation System: Explains preprocessing with TF-IDF, filtering mechanism based on user inputs, and how recommendations are determined using cosine similarity.

# Troubleshooting

Common issues may include WebDriver compatibility, changes in the target website’s structure, or missing data fields. Verify that the latest version of Selenium WebDriver is installed and check if the website’s structure has been altered.