Virtunexa Task 1

Project Report: Python Web Scraper and Calculator Application

1. Introduction

This project integrates two functionalities into a single Python application: a **Web Scraper** and a **Calculator**. The web scraper extracts headlines and links from a given website, while the calculator performs basic arithmetic operations and maintains a history of calculations.

2. Technologies Used

- **Python**: Core programming language
- **BeautifulSoup & Requests**: Web scraping libraries
- **SQLite**: For storing calculation history
- **Pandas**: For exporting history to CSV
- **Logging**: For keeping track of performed calculations
- **Tkinter (optional)**: GUI support for the calculator

3. Features

3.1 Calculator

- Supports **addition, subtraction, multiplication, and division**
- **Error handling** (e.g., division by zero)
- Logs history in **SQLite database**
- Exports history to **CSV format**
- Logs all calculations into a **text file**

3.2 Web Scraper

- Extracts **headlines and links** from websites
- Saves data in **CSV** or **JSON** format
- Handles network errors and invalid URLs
- Uses `requests` to fetch the website data and `BeautifulSoup` to parse HTML

3.3 User Interface

- **Console-based UI** for menu-driven interaction
- Option to switch between calculator and web scraper

4. Implementation Details

4.1 Calculator Functionality

1. The user selects an arithmetic operation.
2. The program validates the input and performs the calculation.
3. The result is displayed and logged.
4. History is stored in **SQLite** and can be exported to a **CSV file**.

4.2 Web Scraper Functionality

1. The user enters a **website URL**.
2. The scraper extracts **headlines (<h2>) and links**.
3. The user selects a **storage format** (CSV/JSON).
4. The extracted data is saved and displayed.

5. Reference Data

5.1 CSV Preview

headline	link
JioMart: Your Go-to Online Grocery Store	No link available

5.2 JSON Preview

```
[
  {
    "headline": "Skip to",
    "link": "No link available"
  },
  {
    "headline": "Keyboard shortcuts",
    "link": "No link available"
  },
  {
    "headline": "Makeup products",
    "link": "No link available"
  },
  {
    "headline": "New looks for the new season",
    "link": "No link available"
  },
  {
    "headline": "Do up your home",
    "link": "No link available"
  }
```

]

6. Strengths & Improvements

Strengths

✔️ Well-structured and modular code ✔️ Provides both functional and analytical capabilities ✔️ Handles user input and errors effectively ✔️ Supports data storage and logging

Potential Enhancements

🔹 Expand scraper functionality to include other HTML elements 🔹 Improve **error handling** for network issues 🔹 Add a **Tkinter-based GUI** for a better user experience

7. Conclusion

This project successfully combines **web scraping** and **arithmetic calculations** in a single Python application. With future enhancements like a GUI and expanded data extraction, it can be further improved for more practical use cases.