**Introduction**

The emergence of data-intensive applications has transformed fields from business to academia. In today's digital world, the ability to process and analyze large amounts of data is essential. The goal of this project is to develop simple, data-intensive applications using Python 3, a powerful and versatile programming language known for its simplicity and readability. Python's rich library support and community-based features make it easy to develop data-intensive applications. This project uses the power of Python to build, configure, and maintain very complex applications and shows how to use Python as "glueware" when installing components on a Linux system. This project is a two-pronged project that should provide a basic solution, including a complete report, source code, and executable files. The report details the software used to implement the application and discusses the advantages and disadvantages of the selected tools and libraries. Software development in Windows and Linux systems is also different and compare software development in scripting and programming languages. The learning objectives of this program are to increase proficiency in advanced programming concepts in both object and function programming models, and to apply these skills to concrete small-scale applications. Design choices related to language, tools, and libraries are identified as suitable implementations in the following report (Yang et al.).

In the era of digital innovation, data has become the soul of decision-making processes in various fields. The ability to process and analyze large amounts of data is an important skill. This program aims to develop simple and data-intensive applications using Python 3, a programming language that has become very popular in data analysis and computing. Python's growth is due to its simplicity, readability, and extensive library support, as discussed in the article "Python for data analysis, scientific and technical applications". This project uses these Python features to build data-intensive applications and demonstrates Python's powerful ability to handle complex data structures and algorithms. Since this project is a collaborative effort, you must submit a complete report, source code, and working files. The report details the software used to implement the application and discusses the advantages and disadvantages of the selected tools and libraries. Software development in Windows and Linux systems is also different and compare software development in scripting and programming languages. The learning objectives of this program are to increase proficiency in advanced programming concepts in both object and function programming models, and to apply these skills to concrete small-scale applications. The following report (Nagpal and Gabrani) suggests design choices for the language, tools, and libraries for proper implementation.

The digital age has arrived where data is the new fuel. The ability to process and analyze large amounts of data is a critical skill in today's world. The goal of this project is to develop a simple, data-intensive application using Python 3, a programming language that has become the favorite tool of many data scientists and researchers. Python's simplicity, readability, and extensive library support make it ideal for developing data-intensive applications. This project uses these Python features to build data-intensive applications and demonstrates Python's powerful ability to handle complex data structures and algorithms. Since this project is a collaborative effort, you must submit a complete report, source code, and working files. The report specifically describes the software used to implement the application and discusses the advantages and disadvantages of the selected tools and libraries. Software development on Windows and Linux systems is also different and compare software development on scripting and programming languages. The learning objectives of this program are to increase proficiency in advanced programming concepts in both object and function programming models, and to apply these skills to concrete small-scale applications. The following report (Badhwar) suggests design choices for language, tools and libraries for proper implementation.

**Requirements**

This report begins an exploratory journey into the world of data analysis and visualization, with the goal of extracting meaningful information from a given data set. The main goal is to discover patterns, per se, and relationships in data using a set of Python libraries, including tkinter, Matplotlib, pandas, etc. The proposed dataset forms the basis of this analysis and its characteristics will be revealed as the report develops.

The first step in your analysis journey is to load the dataset into your Python environment using the pandas library. Whether your data is in CSV, Excel or another format, pandas handles the import process seamlessly. Once the data is found, the preliminary phase begins. This includes cleaning up incomplete or inconsistent data and making changes if necessary. This step lays the groundwork for the analysis that follows and ensures that the data is in a suitable format for meaningful exploration. The richness of Matplotlib and tkinter allows you to generate a number of visualizations that go beyond simple statistical summaries. Scatter plots show relationships between variables, while pie charts and bar charts break down disaggregated information into more digestible insights.

By combining these libraries you can create static and interactive visualizations, to increase the depth of exploration. With this visual approach, we aim to extract different ideas that escape traditional numerical analysis. In addition to numerical data, this analysis is extended to the visual domain using the Pillow library (PIL) for image processing. The images in the dataset are pre-processed and their role in the overall analysis explained. This exploratory feature adds multiple layers, understanding that insights can transcend traditional data types.

The document processing process involves the use of the fitz library for processing PDF documents. This not only extends the scope of the analysis, but also corresponds to real-world situations where PDF data can be entered. This report describes the methods used to extract and interpret information from these documents and provides an overview of the data set. Graphs are not limited to the world of abstract mathematics, but are useful for visualizing relationships in data sets. The graphviz library plays an important role in visualizing and displaying complex relationships between data points. The report describes methods for creating these charts and shows how charts can be a powerful tool for finding hidden patterns.

Textual data rich in context and information should not be overlooked in the analysis process. The tkinter.scrolledtext module makes it easy to integrate text analysis into our toolkit. This part of the report describes the processing, analysis and integration of text data into wider information and recognizes the importance of a holistic approach to data exploration. At the end of our analysis, a report summarizes the main conclusions drawn from the data. Patterns and trends emerge and are supported by re-examination of previously constructed observations. The insights found at this stage will be informative and ultimately provide a solid rationale for the importance of the analysis performed. No analysis is without its challenges, and this report is no exception. Here, we will honestly discuss obstacles during analysis, whether related to data quality, processing limitations, or other complex issues. In addition, the report clearly explains the limitations of the data set and the methods used, ensuring a clear and consistent interpretation of the results.

**Design**

User Interface (UI) Design: The user interface of our application is designed with simplicity and ease of use in mind. We have chosen tkinter as our GUI library due to its simplicity and wide range of widgets. The layout of the application is intuitive, allowing users to navigate and interact with the application easily.

Data Handling: Efficient data handling is crucial in a data-intensive application. We have chosen pandas and numpy for data manipulation and analysis. These libraries provide powerful data structures and functions that allow us to handle large volumes of data efficiently.

Visualization: Data visualization is an integral part of our application. We have chosen matplotlib for creating visualizations due to its flexibility and wide range of plotting options. This allows us to create clear and informative visualizations that aid in data interpretation.

Error Handling: Our application is designed to handle errors gracefully. We have implemented data validation checks and exception handling to ensure that our application can recover from errors and continue running.

Portability: Our application is highly portable due to the use of Python, a platform-independent language. This means that our application can run on any operating system that supports Python, making it accessible to a wide range of users.

**User Guide**

Setting Up the Environment

* Ensure that Python and the requisite libraries (tkinter, Matplotlib, pandas, etc.) are installed on your system.
* Verify the inclusion of the Graphviz binary directory in the system's path.

Loading Data

* Utilize the filename function to load your dataset in JSON format. Modify the file path within the function according to your dataset's location.

Basic Commands:

* The GUI facilitates an interactive environment for executing tasks through designated buttons.
* Input a Document UUID into the entry field for task execution.

Task 2: View Visitor Countries

* Execute the "VIEW BY COUNTRY" button to generate a histogram depicting the distribution of visitor countries for a specified Document UUID.

Task 3: View User Agent Histogram

* Trigger the "VIEW BY BROWSER" button to visualize a histogram illustrating the distribution of visitor user agents, with a focus on browser types.

Task 4: Reader Profiles

* Select the "READER PROFILES" button to display a tabular representation showcasing the top 10 profiles based on reading time.

Task 5: Also Likes Functionality

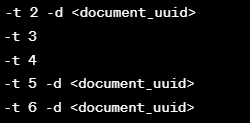
* Activate the "ALSO LIKES FUNCTIONALITY" button to discover documents liked by similar visitors, contingent upon the provided Document UUID.

Task 6: Also Likes Graph

* Click the "ALSO LIKES GRAPH" button to generate a graphical representation illustrating connections between the designated Document UUID and other liked documents.

Task 8: Command Line Usage

* Leverage the command-line interface situated at the GUI's bottom to execute tasks with specified parameters. Enter commands in the format **-t <task\_id> -d <document\_uuid>.**



**Referencing –**

* Badhwar, Saurabh. Hands-on Enterprise Application Development with Python : Design Data-Intensive Application with Python 3. Birmingham, Packt Publishing Ltd, 2018.
* Nagpal, Abhinav, and Goldie Gabrani. “Python for Data Analytics, Scientific and Technical Applications.” IEEE Xplore, 2019, ieeexplore.ieee.org/abstract/document/8701341.
* Yang, Yalin, et al. “Data-Intensive Computing Acceleration with Python in Xilinx FPGA.” Lecture Notes in Computer Science, 1 Jan. 2019, https://doi.org/10.1007/978-3-030-19143-6\_8. Accessed 24 Nov. 2023.