**PROJECT REPORT**

**Project Title:** Excel Report Generator  
**Domain:** Data Processing & Automation  
**Tools Used:** Python, pandas, openpyxl, matplotlib, Tkinter

**1. Introduction**

Data is one of the most valuable assets in today’s world. Organizations, researchers, and students often work with CSV files to store and share data. However, raw CSV files are not presentation-friendly and require transformation into meaningful reports.

The **Excel Report Generator** is designed to automate this process by converting raw CSV inputs into well-structured and styled Excel files. This reduces manual effort, increases efficiency, and ensures accuracy in reporting.

**2. Objectives**

* To develop a **Python-based application** that reads CSV data and generates professional Excel reports.
* To automate **data summarization** using pivot tables and descriptive statistics.
* To enhance reports with **visual charts** for better insights.
* To implement a **user-friendly GUI** for selecting input files and saving outputs.
* To ensure the generated reports are **styled, organized, and ready-to-use**.

**3. Tools & Technologies**

* **pandas** – For data loading, cleaning, transformation, and pivot tables.
* **openpyxl** – For exporting, styling, and formatting Excel files.
* **matplotlib** – For generating visual charts (bar, line, pie, etc.).
* **Tkinter** – For GUI support with file upload and save dialogs.
* **Python 3.x** – Core programming language.

**4. Methodology**

The project follows a modular approach:

1. **Load CSV Data**
   * User selects a CSV file using the Tkinter file dialog.
   * Data is read into a pandas DataFrame.
2. **Data Processing**
   * Perform basic cleaning if required.
   * Generate pivot tables for summarized insights.
   * Calculate descriptive statistics (mean, median, max, min).
3. **Visualization**
   * Generate charts (bar, line, pie) using matplotlib.
   * Save charts as images and embed into Excel.
4. **Excel Report Generation**
   * Export raw data, pivot tables, and summary stats to Excel.
   * Apply styling (headers, colors, borders) using openpyxl.
5. **User Interaction**
   * File dialogs for selecting input CSV and saving output report.
   * Simple GUI for ease of use.

**5. Features**

✔ CSV import via GUI  
✔ Automated pivot table generation  
✔ Summary statistics (mean, median, count, etc.)  
✔ Visual charts for quick insights  
✔ Styled Excel export with multiple sheets  
✔ Simple and interactive Tkinter interface

**6. Project Structure**

excel-report-generator/

│── main.py # Entry point (Tkinter GUI + logic)

│── report\_generator.py # Core functions (pandas, openpyxl, charts)

│── requirements.txt # Required Python libraries

│── sample\_data.csv # Example input file

│── outputs/ # Generated Excel reports

│── README.md # Documentation

**7. Sample Output**

* **Raw Data Sheet** → Contains the original CSV data.
* **Summary Sheet** → Pivot tables and descriptive statistics.
* **Charts Sheet** → Bar, line, and pie charts for visualization.
* **Styled Report** → Professionally formatted Excel file.

**8. Applications**

* **Business Analytics** – Sales, marketing, and financial data reports.
* **Education & Research** – Student performance and research data summarization.
* **Data Automation** – Reducing manual reporting tasks in organizations.
* **Personal Productivity** – Generating quick insights from CSV logs (expenses, records, etc.).

**9. Future Enhancements**

* Multi-file CSV import and merging.
* More advanced chart customization.
* Option to generate PDF reports along with Excel.
* Integration with databases (MySQL, SQLite).
* Scheduling automated reports.

**10. Conclusion**

The **Excel Report Generator** successfully automates the conversion of raw CSV data into structured Excel reports with pivot tables, summary statistics, and charts. This project not only saves time but also ensures accuracy, consistency, and professional presentation of data.

It can be extended in the future for advanced business intelligence tasks and integrated with cloud platforms for wider applicability.