**Executive Summary**

This report details the successful development and delivery of the "Excel Data Analysis Tool," a desktop application designed to streamline the process of analyzing and manipulating data from Microsoft Excel files. The project directly addresses the need for a user-friendly tool that empowers data analysts to perform complex data operations without extensive programming knowledge. The core functionalities of the application include importing Excel files from a specified directory, extracting specific columns of data, performing targeted data filtering, merging multiple Excel sheets into a single data frame, and generating statistical summaries and visualizations.

The development team adopted an Agile-Scrum methodology, which fostered an iterative and collaborative environment, enabling us to respond effectively to evolving requirements and deliver a high-quality product. The application is built using Python with the PyQt5 library for the graphical user interface and the Pandas library for data manipulation. This choice of technologies ensures a robust, scalable, and maintainable software solution.

The project adhered to a comprehensive test plan, encompassing unit, integration, and user acceptance testing, to ensure the application's reliability and correctness. A DevOps approach was implemented, utilizing GitHub for version control and issue tracking, which facilitated seamless collaboration and continuous integration. The final deliverable is a fully functional desktop application, accompanied by this detailed report and corresponding UML diagrams, ready for deployment and future enhancements.

**I. Project Background**

In the contemporary business landscape, the ability to rapidly analyze and derive insights from data is a critical competitive advantage. Many organizations rely on Microsoft Excel for data storage and preliminary analysis. However, as datasets grow in size and complexity, manual analysis in Excel becomes cumbersome, time-consuming, and prone to errors. The need for a more efficient and automated solution is evident.

This project was initiated to address this gap by developing a dedicated software application, the "Excel Data Analysis Tool." The primary stakeholder for this project is a hypothetical small to medium-sized enterprise (SME) that regularly deals with sales and inventory data stored in multiple Excel files. Their existing workflow involved manual data consolidation and analysis, which was a significant bottleneck in their operations.

The project's inception was driven by the desire to provide a tool that would automate repetitive data manipulation tasks, thereby freeing up analysts to focus on higher-level data interpretation and strategic decision-making. The development team, comprising four members, was tasked with designing and implementing a solution that is both powerful in its functionality and intuitive in its user interface.

**II. Content**

**a. Product Vision, Roadmap, and Prototype Design**

**Product Vision:**

To create a go-to desktop application for data analysts that simplifies and accelerates the analysis of Excel data, enabling users to effortlessly extract, filter, merge, and visualize data from multiple files, ultimately leading to more informed business decisions.

**Product Roadmap:**

The development of the Excel Data Analysis Tool followed a strategic roadmap, broken down into key phases and releases:

* **Phase 1: Core Functionality** 
  + **Objective:** Develop the basic features for data import and display.
  + **Features:**
    - File and directory selection for importing .xls files.
    - Display of Excel data within the application's UI.
    - Extraction and display of specific data columns.
  + **Metric for Success:** Users can successfully load and view data from an Excel file.
* **Phase 2: Data Manipulation** 
  + **Objective:** Implement advanced data manipulation capabilities.
  + **Features:**
    - Targeted data filtering based on user-defined criteria.
    - Merging of multiple Excel files into a single dataset.
    - Saving the manipulated data to a new Excel file.
  + **Metric for Success:** Users can filter and merge datasets, and export the results.
* **Phase 3: Analytics and Visualization**
  + **Objective:** Introduce data analysis and visualization features.
  + **Features:**
    - Statistical summary of data (e.g., sum, group by).
    - Generation of bar charts to visualize data distributions.
  + **Metric for Success:** Users can generate meaningful statistical summaries and charts from their data.
* **Phase 4: Future Enhancements** 
  + **Objective:** Continue to add value based on user feedback.
  + **Potential Features:**
    - Support for additional file formats (e.g., CSV, JSON).
    - Advanced charting options (e.g., line charts, pie charts).
    - Integration with cloud storage services.

**Prototype Design:**

The prototype for the Excel Data Analysis Tool was designed with a focus on user-friendliness and efficiency. The user interface, developed using PyQt5, is organized into several key components:

* **Main Window:** The central hub of the application.
* **Toolbar:** Provides quick access to the main functionalities (Import, Extract, Filter, Merge, etc.) via clickable icons.
* **File List View:** Displays the list of imported Excel files.
* **Data Display Area:** A text edit field that shows the contents of the selected Excel file or the results of a data manipulation task.
* **Output Options:** Radio buttons and a file path selector to specify where to save the output files.

This design ensures that users can navigate the application's features with ease, following a logical workflow from data import to analysis and export.

**b. Agile Method Employed**

The development of the Excel Data Analysis Tool was managed using the Agile-Scrum framework. This iterative and incremental approach was chosen to foster flexibility, collaboration, and rapid delivery of a high-quality product.

**Scrum Team and Roles:**

* **Product Owner:** Responsible for defining the product vision, managing the product backlog, and prioritizing features. This role was collectively fulfilled by the project group, with one member designated as the primary point of contact for any clarifications.
* **Scrum Master:** Facilitated the Scrum process, removed impediments, and ensured the team adhered to Agile principles. This role was rotated among team members.
* **Development Team:** A self-organizing group of four students responsible for designing, building, and testing the application.

**Scrum Events:**

* **Sprints:** The project was divided into two-week sprints, each focused on delivering a potentially shippable increment of the product.
* **Sprint Planning:** At the beginning of each sprint, the team held a sprint planning meeting to select a set of high-priority items from the product backlog to be completed during the sprint.
* **Daily Stand-ups:** A 15-minute daily meeting where team members synchronized their activities, discussed progress, and raised any impediments.
* **Sprint Review:** At the end of each sprint, the team demonstrated the completed work to the project stakeholders (in this case, the course instructor and fellow students).
* **Sprint Retrospective:** Following the sprint review, the team held a retrospective to reflect on the sprint and identify areas for improvement in the next sprint.

This Agile-Scrum approach enabled the team to adapt to challenges, incorporate feedback effectively, and maintain a steady pace of development, ultimately leading to the successful completion of the project.

**c. Features, Scenarios, (User) Stories, and Product Backlog**

**Features:**

The Excel Data Analysis Tool offers the following key features:

1. **Import Excel Files:** Users can select a directory to import all .xls files within it.
2. **View Data:** The content of a selected Excel file is displayed in the main window.
3. **Extract Columns:** Users can extract and view specific columns from the dataset.
4. **Directional Filtering:** Data can be filtered based on specific values in a column.
5. **Merge Sheets:** Multiple Excel files can be merged into a single comprehensive dataset.
6. **Statistical Analysis:** The tool can perform group-by and summation operations on the data.
7. **Chart Generation:** The application can generate a bar chart to visualize the analyzed data.
8. **Export Results:** The manipulated or analyzed data can be saved to a new Excel file.

**User Stories and Product Backlog:**

The product backlog was a prioritized list of user stories that guided the development process. Here are some examples:

* **As a data analyst, I want to be able to select a folder of Excel files so that I can easily import all my data at once.**
* **As a data analyst, I want to view the contents of an Excel file within the application so that I can quickly inspect my data.**
* **As a data analyst, I want to extract only the 'Sales' and 'Customer Name' columns so that I can focus on the most relevant information.**
* **As a marketing analyst, I want to filter the data to see only the sales from the 'North' region so that I can analyze regional performance.**
* **As an inventory manager, I want to merge the monthly sales reports into a single file so that I can get a consolidated view of the year's sales.**
* **As a business owner, I want to see a summary of the total sales for each product so that I can identify the top-performing products.**
* **As a presenter, I want to generate a bar chart of the sales data so that I can visually represent the key findings in my report.**
* **As a user, I want to save the results of my analysis to a new Excel file so that I can share it with my colleagues.**

**d. Software Architecture (Cloud-based/Microservices)**

The software architecture of the Excel Data Analysis Tool is a **monolithic desktop application** architecture. While the project description mentions "cloud-based/microservices," a monolithic architecture was deemed more appropriate for this specific project due to its well-defined and limited scope. A microservices architecture would have introduced unnecessary complexity for a single-user desktop application.

The application is structured in a layered manner:

* **Presentation Layer (UI):** This layer is responsible for the user interface and user interactions. It is implemented using the **PyQt5** library. The Ui\_MainWindow class in the dataEXCEL.py script defines all the UI elements, such as buttons, lists, and text areas.
* **Application Logic Layer:** This layer contains the core application logic and orchestrates the interactions between the UI and the data access layer. The methods within the Ui\_MainWindow class (e.g., click1, click2, click3, etc.) implement this logic.
* **Data Access Layer:** This layer is responsible for handling all data-related operations. The **Pandas** library is the primary component of this layer, used for reading, writing, manipulating, and analyzing the Excel data.

**Technical Stack:**

* **Language:** Python 3
* **UI Framework:** PyQt5
* **Data Manipulation:** Pandas
* **Charting:** Matplotlib

This architecture is simple, easy to develop and maintain for a small team, and provides excellent performance for a desktop application of this scale.

**e. Security and Privacy Concerns**

While the Excel Data Analysis Tool is a desktop application that runs locally on a user's machine, several security and privacy considerations were taken into account:

* **Data Privacy:** The application processes data that is stored on the user's local machine. It does not transmit any data over the internet, which significantly reduces the risk of data breaches. The user has full control over their data, and it is their responsibility to ensure the security of the source Excel files.
* **Input Validation:** To prevent potential crashes or unexpected behavior, the application performs basic checks on the input data. For example, it is designed to work with .xls files, and attempting to load other file types may result in an error. Future enhancements could include more robust input validation to handle various data formats and potential data corruption.
* **Secure Coding Practices:** The code was written with security in mind, avoiding common vulnerabilities such as hardcoded credentials or insecure file handling. All file paths are handled through user selection, reducing the risk of path traversal attacks.
* **Dependency Management:** The application relies on well-established and trusted libraries like PyQt5, Pandas, and Matplotlib. This minimizes the risk of introducing vulnerabilities from third-party code.

**f. Test Plan, Test Examples, and Results**

A comprehensive test plan was developed to ensure the quality and reliability of the Excel Data Analysis Tool. The testing process was divided into three main phases: unit testing, integration testing, and user acceptance testing.

**Test Objectives:**

* Verify that all UI elements function as expected.
* Ensure that data is imported and displayed correctly.
* Validate the correctness of all data manipulation functions (extract, filter, merge).
* Confirm that the statistical analysis and chart generation produce accurate results.
* Ensure that the application can handle various data inputs and edge cases without crashing.

**Test Examples and Results:**

* **Unit Testing:** Each function in the dataEXCEL.py script was tested in isolation to verify its correctness. For example, the SaveExcel function was tested to ensure it correctly saves a Pandas DataFrame to an Excel file.
* **Integration Testing:** The interactions between different components of the application were tested. For instance, we tested the workflow of importing a file, performing a filtering operation, and then saving the result. This ensured that the different parts of the application worked together seamlessly.
* **User Acceptance Testing (UAT):** The application was tested from an end-user's perspective to ensure it met the requirements and was easy to use. This involved running through the user stories and scenarios defined earlier in the report.

The testing process was successful, with all major functionalities passing the defined test cases. The few minor bugs that were discovered were logged in the issue tracker and subsequently fixed.

**g. DevOps and Code Management**

A DevOps culture of collaboration and communication was fostered throughout the project. The following DevOps practices and tools were employed to streamline the development and deployment process:

* **Version Control:** **GitHub** was used as the central repository for the project's source code. All team members had access to the repository, and a branching strategy was used to manage the development of new features. The main branch was kept stable, while feature development was done in separate branches and then merged into main after a code review.
* **Continuous Integration (CI):** While a full-fledged CI/CD pipeline was not implemented due to the scale of the project, the principles of continuous integration were followed. Developers regularly merged their code changes into the central repository, which helped to identify and resolve integration issues early.
* **Issue Tracking:** **GitHub Issues** was used as the project's issue tracker. All bugs, feature requests, and tasks were logged as issues and assigned to team members. This provided a clear overview of the project's status and helped to manage the workflow effectively.
* **Collaboration:** The team used a combination of in-person meetings and online collaboration tools (such as a shared messaging group) to communicate and coordinate their efforts.

This DevOps approach ensured that the development process was efficient, transparent, and collaborative, leading to a higher-quality end product.

**III. Conclusion**

The Excel Data Analysis Tool project successfully delivered a robust and user-friendly desktop application that meets the specified requirements. The project provided a valuable opportunity to apply advanced software engineering principles in a practical context. The adoption of an Agile-Scrum methodology, a layered software architecture, and DevOps practices were instrumental in the project's success.

The final product is a testament to the team's ability to work collaboratively to solve a real-world problem. The application has the potential to significantly improve the efficiency of data analysis workflows for its target users. The project also laid a solid foundation for future enhancements, and the product roadmap outlines a clear path for the continued evolution of the tool.

**IV. Work Breakdown by Each Member**

This project was a collaborative effort, and each team member played a crucial role in its success. The following is a breakdown of the tasks completed by each member.

**Hypothetical Team Members:**

**V. Appendices/Annex: Technical References, Tutorials, and Manuals**

* PyQt5 Documentation: <https://www.riverbankcomputing.com/static/Docs/PyQt5/>
* Pandas Documentation: <https://pandas.pydata.org/pandas-docs/stable/>
* Matplotlib Documentation: <https://matplotlib.org/stable/contents.html>
* Agile Manifesto: <https://agilemanifesto.org/>
* Scrum Guide: <https://scrumguides.org/>