# Project Description: Data Validation Application

## Objective

The primary objective of this project is to validate data from two sources: the client (Senpiper) and our UK server. The validation process includes checking if the files have the exact count of rows, identifying missing data in mandatory fields/columns, and ensuring the column names are consistent. Additionally, the application will compare the date ranges in both files to ensure they cover the same period.

## Tools and Environment

- Programming Language: Python
- Development Environment: Visual Studio Code
- Libraries Used: pandas, tkinter, datetime, os

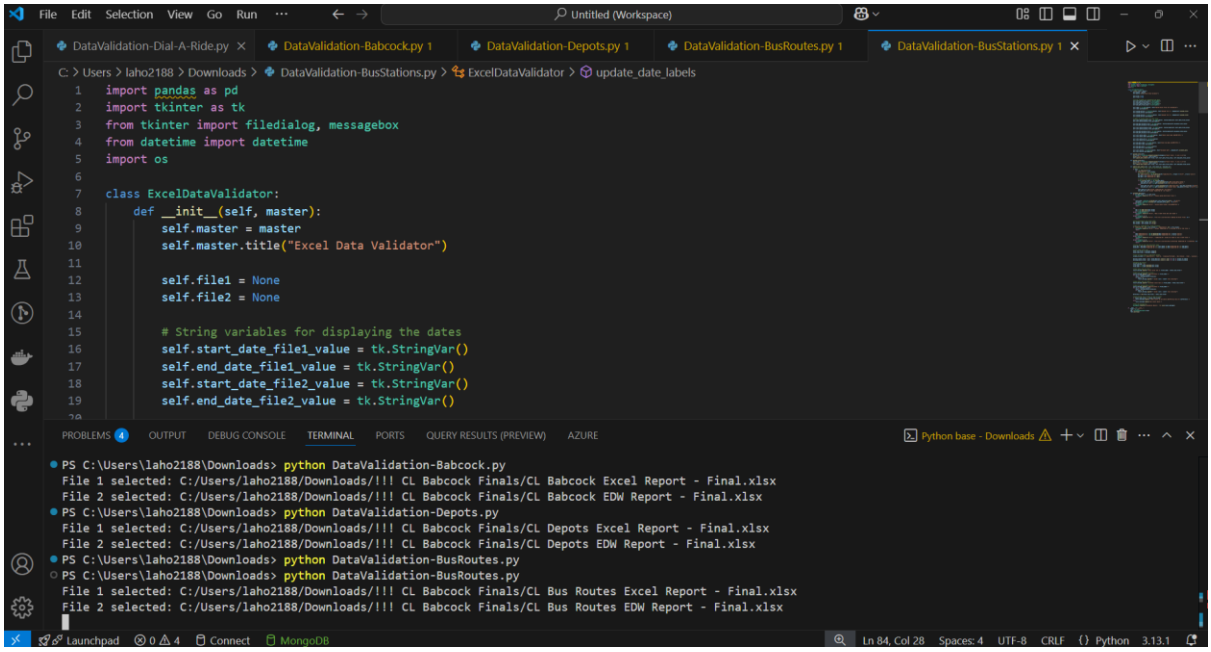


**Image 1. Using Visual Studio Code to create the Python application**

## Key Features

1. **File Upload and Comparison:**
   - Users can upload two Excel files for comparison: one from the client (Senpiper) and one from our UK server/tables.
   - The application reads both Excel files and loads the data into pandas DataFrames.

2. **Row Count Validation:**
    - The application checks if both data sources have the same number of rows.
    - If the row counts differ, the application highlights the discrepancy.
3. **Mandatory Fields/Columns Check:**
    - The application identifies mandatory fields/columns that must not have missing data.
    - It checks for missing data in these fields and generates a report listing any missing values.
4. **Date Range Validation:**
    - The application extracts the earliest and latest dates from both data sources.
    - It compares the date ranges to ensure both cover the same period.
5. **Column Name Consistency Check:**
    - The application checks if the column names in both data sources are identical.
    - It identifies any missing columns or columns with different names in the UK server file compared to the client (Senpiper) file.

6. **User Interface:**
    - The application provides a user-friendly interface for uploading files and viewing validation results.
    - Users can easily navigate the interface to upload files, initiate validation, and view the generated reports.

## Detailed Steps and Implementation

1. **File Upload:**
    - The application interface allows users to select and upload two Excel files for comparison. This feature ensures that users can quickly provide the necessary data for validation without any technical hurdles.
    - The panda tkinter library creates a graphical user interface (GUI) for file upload. Users can select files using a file dialogue.
2. **Data Loading:**
    - Once the files are uploaded, the application reads the data using the panda's library. This step involves loading the data into pandas DataFrames, which are efficient data structures for handling large datasets and performing various data manipulation tasks.
    - The application prints the selected file names to the console for confirmation.
3. **Row Count Validation:**
    - The application compares the number of rows in both DataFrames. This step is crucial because it ensures that both data sources contain the same amount. A

discrepancy in the row counts could indicate missing or extra data in one of the sources, which needs to be addressed.

- The application generates a report indicating the difference in row counts if they are unequal.
- In this project, it was found that the UK server files have fewer rows than the client (Senpiper) files, indicating potential data loss or incomplete data transfer.

4. **Date Range Validation:**
   - The application extracts the earliest and latest dates from both DataFrames. This step involves parsing the date columns and identifying the minimum and maximum dates in each data source. The application then compares these date ranges to ensure both cover the same period. A mismatch in the date ranges could indicate that one of the sources is missing data for specific periods, which needs to be addressed.
   - The application uses the 'Submitted On' column to determine the date range. If this column is missing, the application notifies the user.
   - The user can check different date ranges using the Start and End date input fields.

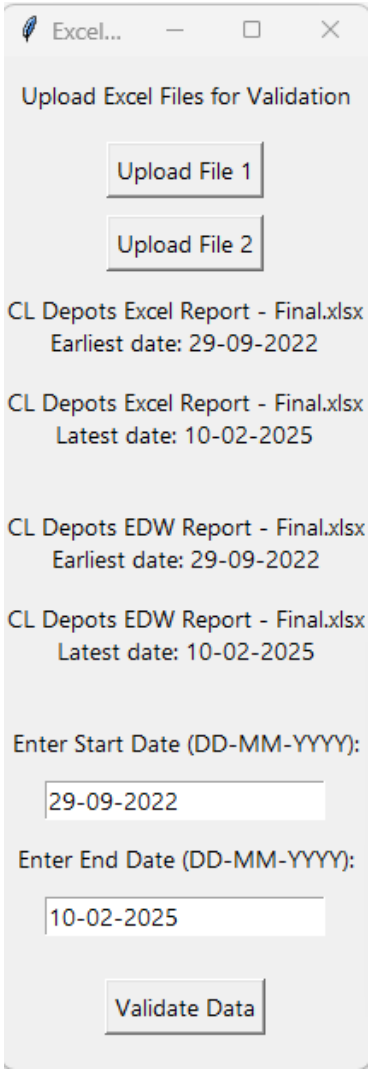


**Image 2. Earliest/latest date from both files and input fields**

5. **Mandatory Fields/Columns Check:**
    - The application identifies mandatory fields/columns that must not have missing data. These fields are predefined based on the requirements of the data validation process. The application checks for missing values in these fields and generates a report listing discrepancies. This step ensures that all critical data is present and accounted for.
    - Some mandatory fields checked in this project are: 'Location ID', 'Team', 'Job / BU', 'Building', 'Site', 'Depot', 'Location' and 'Area'.
    - It was discovered that both data sources have many mandatory fields/columns with missing data, which needs to be addressed to ensure data completeness and accuracy.

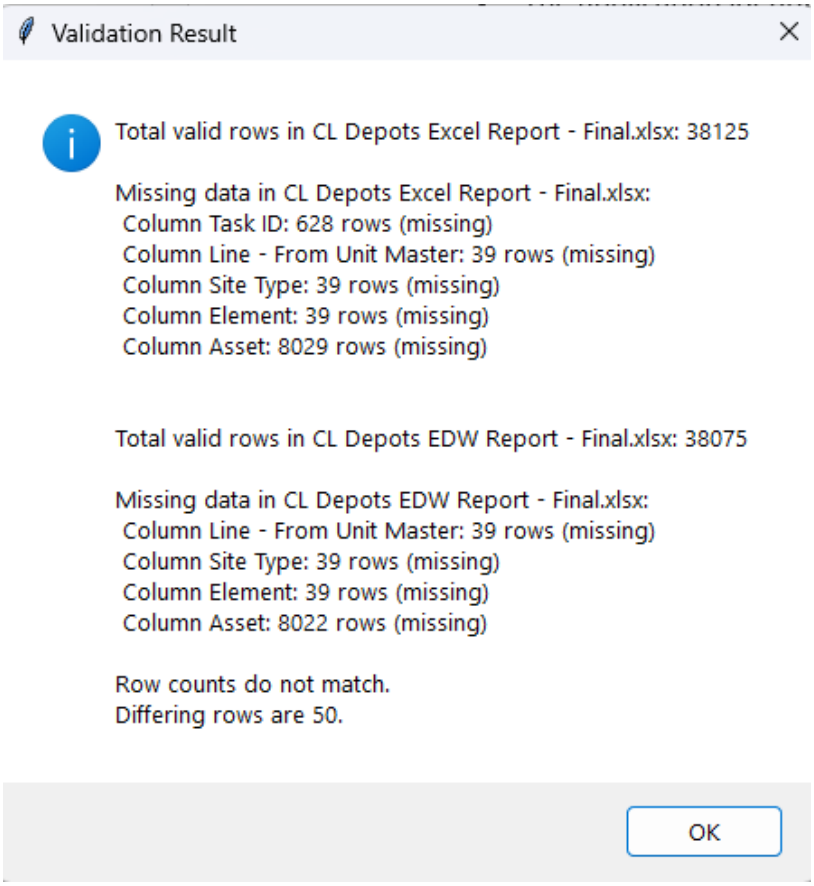


**Image 3. Count of rows from both files and a list of columns with missing data**

6. **Report Generation:**
    - The application generates a comprehensive report summarising the validation results. This report includes details on row count discrepancies, missing data in mandatory fields, and date range mismatches. It is designed to be user-friendly and easy to understand, providing clear insights into any issues that must be addressed.
    - The report is displayed in a message box using the tkinter library.

## Benefits and Impact

1. **Data Consistency:**
   - By validating the data from two sources, the application ensures that both sources contain consistent and accurate data. This consistency is crucial for maintaining data integrity and reliability, which is essential for any data-driven decision-making process.
2. **Efficiency:**
   - The application automates the data validation process, saving time and effort for users. Instead of manually comparing the data sources and identifying discrepancies, users can rely on the application to perform these tasks quickly and accurately.
3. **Error Detection:**
   - The application identifies missing data in mandatory fields and highlights discrepancies in row counts and date ranges. This error detection capability helps users identify and address any issues in the data, ensuring that the final dataset is complete and accurate.
4. **User-Friendly Interface:**
   - The application provides a user-friendly interface that makes it easy for users to upload files, initiate validation, and view the results. This ease of use ensures that users can quickly and efficiently perform data validation tasks without technical hurdles.
5. **Comprehensive Reporting:**
   - The application generates a comprehensive report summarising the validation results. This report provides clear insights into any issues that need to be addressed, helping users make informed decisions about their data.

**Conclusion**

During the validation process, we found that the data from our UK server has fewer rows than the client (Senpiper) files. This discrepancy needs to be investigated to determine whether the data transfer from the client is inefficient or if the client is sending less data. Additionally, we discovered missing data in mandatory columns, which needs to be addressed with the client to ensure these issues are fixed.

The user-friendly interface and comprehensive reporting capabilities make it easy for users to perform data validation tasks quickly and efficiently, highlighting the issues.