!AS FOR THE CURRENT CUSTOMER CRITERIA!

Project Scope

This test plan covers the current milestone of the Excel Data Combiner prototype. The main objectives are to verify that the system properly reads two Excel files (hospital and Medicaid datasets), merges them based on predefined criteria, and displays the merged data in the user interface. The functionality to view individual profiles and display unmatched data will also be tested.

Test Objectives

1. **File Reading:**
    - Verify that the system can read two Excel files (hospital and Medicaid datasets).
    - Ensure that appropriate error messages are displayed when files are not selected or are in an incorrect format.
2. **Data Merge:**
    - Verify that the merge is performed based on the correct fields: **Mother's First Name**, **Mother's Last Name**, and **Child's DOB**.
    - Ensure that the merged data contains only valid matches between the two datasets.
    - Verify that unmatched records are logged to the console and saved in the unmatched file.
3. **GUI Functionality:**
    - Verify that the user interface displays the combined data in the correct format (Mother ID, Child Name, DOB).
    - Test double-click functionality on each child entry to ensure it opens a detailed profile window with the correct data.
    - Test navigation between windows and proper layout display.
4. **Data Integrity:**
    - Check that the data in the merged output reflects the correct combined fields and is consistent with the input data.
    - Ensure that no data is lost or misrepresented during the merge process.
5. **Save Files:**
    - Verify that the system saves the merged data as *combined_matched_data.xlsx*.
    - Check that unmatched data is saved correctly in a separate file, *combined_unknown_data.xlsx*.
6. **Performance:**
    - Test the time taken to process and combine large datasets.
    - Ensure the system remains responsive and does not crash when handling larger data volumes.

!AS FOR THE CURRENT CUSTOMER CRITERIA!

Test Cases

Test Case 1: File Reading

Objective: Verify that the system correctly reads two Excel files.

- **Steps:**
  - Launch the application.
  - Click on "Read Excel File 1" and select a valid hospital dataset.
  - Click on "Read Excel File 2" and select a valid Medicaid dataset.
- **Expected Results:**
  - The system successfully reads both files without errors.
  - Error messages appear if an invalid file is selected or no file is selected.

Test Case 2: Merge Operation

Objective: Verify that the system merges the two datasets based on Mother's Name and Child's DOB.

- **Steps:**
  - After reading the two Excel files, click on "Combine Data."
  - Observe the console for any unmatched records.
  - Verify the correctness of the *combined_matched_data.xlsx* file.
- **Expected Results:**
  - Only matched records are merged.
  - The combined data file should have the merged entries for records that match based on the three fields.
  - Unmatched records are displayed in the console and saved separately.

Test Case 3: Display Combined Data

Objective: Verify that the combined data is displayed in the UI.

- **Steps:**
  - After clicking "Combine Data," check the UI window for the list of combined names.
  - Ensure each entry contains the **Mother ID**, **Child Name**, and **DOB**.
- **Expected Results:**
  - Combined names appear in the list.
  - The data in the UI matches the merged data in the *combined_matched_data.xlsx*.

Test Case 4: View Child Profile

Objective: Verify that double-clicking an entry opens the correct profile.

- **Steps:**

    - Double-click on any child entry in the combined data window.
    - Ensure a new window opens displaying the full profile data for that child.
- **Expected Results:**
    - The correct profile data from the merged sheet is displayed in the profile window.

Test Case 5: Handling Errors

Objective: Ensure that the system handles errors gracefully.

- **Steps:**
    - Try reading files with missing columns (e.g., no Mother_First_Name or Child_DOB).
    - Select a file in an incorrect format (e.g., a CSV instead of an Excel file).
    - Try running the merge without selecting both files.
- **Expected Results:**
    - Appropriate error messages are shown for missing or incorrect data.
    - The system should not crash or freeze in case of invalid input.

Test Case 6: File Saving

Objective: Verify that the system saves merged and unmatched data properly.

- **Steps:**
    - Perform the merge.
    - Check the directory for *combined_matched_data.xlsx*.
    - Open both files and verify their contents.
- **Expected Results:**
    - The matched file contains all the correctly merged data.
    - The unmatched file contains the records that could not be merged.

Test Case 7: Performance Test

Objective: Ensure that the application performs efficiently with large datasets.

- **Steps:**
    - Run the application with large Excel files (e.g., 1000+ rows).
    - Measure the time taken for the merge operation.
    - Test if the UI remains responsive during the merge.
- **Expected Results:**
    - The system should perform the merge in a reasonable time frame.
    - The UI should remain responsive and not freeze during processing.

Test Environment

- **Operating System:** Test across different platforms (Windows, macOS, Linux).

!AS FOR THE CURRENT CUSTOMER CRITERIA!

- **Python Version:** Python 3.7 or higher.
- **Excel Versions:** Ensure compatibility with different Excel file formats (.xls and .xlsx).
- **Test Data:** Use sample datasets provided for testing or generate mock data using tools like Faker.

Risk Assessment

- **Incorrect Merge:** There is a risk that the merging process might combine incorrect records due to mismatched names or DOB. String normalization (case, spaces) should mitigate this risk.
- **File Format Errors:** If files are in the wrong format or contain missing columns, the system should handle these gracefully without crashing.
- **Performance Bottlenecks:** Larger datasets might cause the application to slow down or crash if not handled efficiently.

Acceptance Criteria

The application will be considered successful if:

1. The system correctly reads both datasets.
2. The merge operation only includes exact matches and handles errors.
3. The combined data is displayed correctly in the UI, and detailed profiles are accessible.
4. Unmatched data is logged and marked in the console separately.
5. The system remains stable and responsive under heavy load.