Paper OS08

Deceptive Twins: An Interactive Python Tool for Automated RTF Document Comparison in Clinical Submissions

Aravind Gajula, Bristol Myers Squibb, Lawrenceville, NJ, USA
Indraneel Narisetty, Gilead Sciences Inc., Foster City, USA

ABSTRACT

In clinical trial programming, routine updates ranging from a minor footnote edit to a full dataset refresh can trigger wholesale regeneration of tables, listings, and figures. While the underlying content often remains unchanged, refreshed timestamps and formatting requirements force reviewers to re-examine outputs line by line to confirm that nothing meaningful has shifted. As documents progress through multiple regulatory revisions, this problem intensifies, giving rise to what we call “deceptive twins”: files that look virtually identical at first glance, yet may conceal subtle but critical differences.

To address this challenge, we developed Deceptive Twins, an interactive Python application that automates the comparison of Rich Text Format (RTF) files commonly used in clinical deliverables. The tool features a PyQt5-based graphical interface for selecting document versions, leverages the striprtf library for text extraction, and applies Python’s difflib to perform word-level comparisons. It generates color-coded HTML reports that clearly highlight insertions, deletions, and unchanged text. By minimizing unnecessary manual review and improving change visibility, Deceptive Twins streamlines quality control and enhances efficiency across clinical reporting and regulatory submissions.

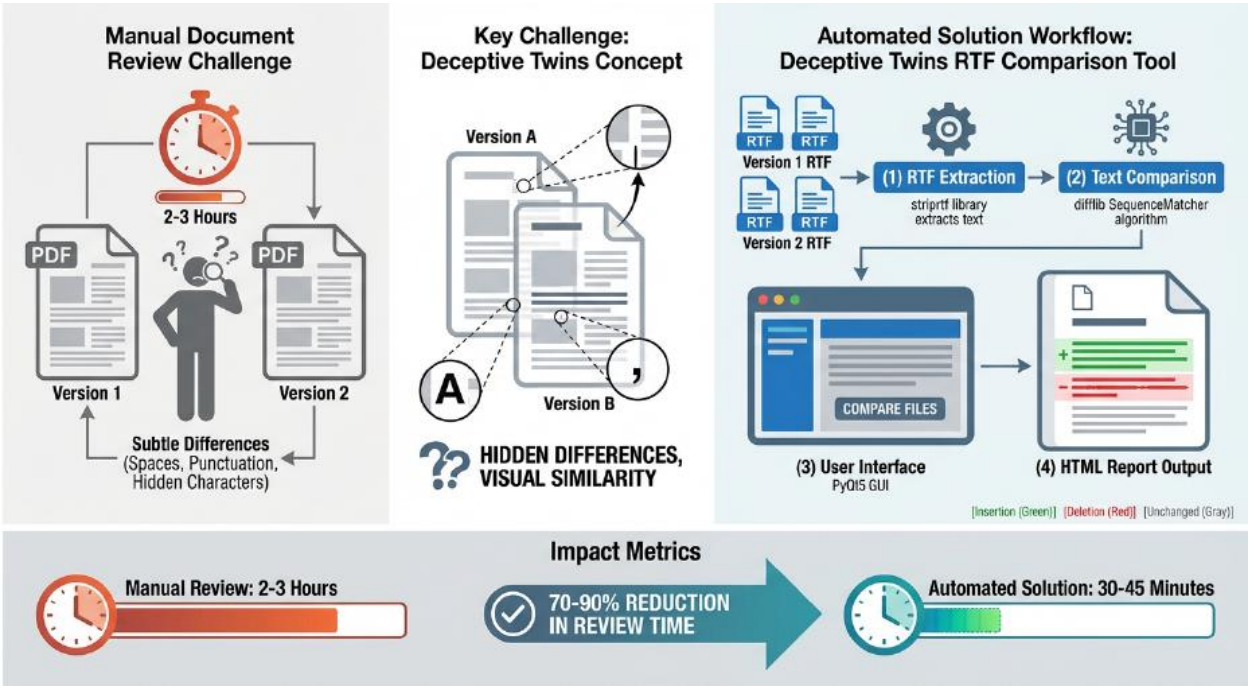

Figure 1: Graphical Abstract: The Deceptive Twins workflow transforms the challenging task of RTF document comparison into an automated, reliable process.

INTRODUCTION

Clinical trial programming generates extensive documentation throughout the study lifecycle. Tables, listings, and figures (TLFs) are produced in Rich Text Format (RTF) to meet regulatory requirements and sponsor specifications. During a study, these outputs undergo numerous revisions from database updates, protocol amendments, programming updates, to minor formatting corrections.

Rich Text Format remains the de facto standard for clinical deliverables despite the emergence of newer document formats. Regulatory agencies accept RTF files for submission, and legacy statistical programming systems continue to generate outputs in this format. However, RTF files present unique comparison challenges due to their binary encoding, formatting metadata, and timestamp information embedded within the document structure.

A fundamental challenge arises from this iterative process: distinguishing meaningful content changes from incidental modifications. When a single data point changes in a large dataset, the entire output may be regenerated. File timestamps update, metadata changes, and the document appears as a new version even when 99% of its content remains identical to the previous version.

THE “DECEPTIVE TWINS” PROBLEM

We define "deceptive twins" as document versions that:
• Look identical at first glance
• Have different file metadata or timestamps
• Contain subtle but important text differences
• Require careful, line-by-line comparison to spot changes

This problem is especially critical in regulatory submissions, where accuracy is paramount. Quality control (QC) programmers must verify that all changes are intentional and correct. Manual comparison of RTF documents has several drawbacks:

• Time-consuming, particularly for large tables with many data points
• Prone to human error—reviewers can miss subtle changes
• Inefficient, wasting effort on unchanged content
• Inconsistent, as different reviewers may apply different standards

SOLUTION OVERVIEW

Deceptive Twins solves these problems by automating the comparison process. By extracting and comparing text word by word, the tool:

• Eliminates manual visual inspection of document versions
• Clearly highlights specific changes between versions
• Creates permanent HTML comparison reports for audit trails
• Reduces QC review time by 70–90% for typical documents

SYSTEM ARCHITECTURE

The architecture of the Deceptive Twins is composed of three distinct layers, each designed to enhance modularity, scalability, and maintainability.

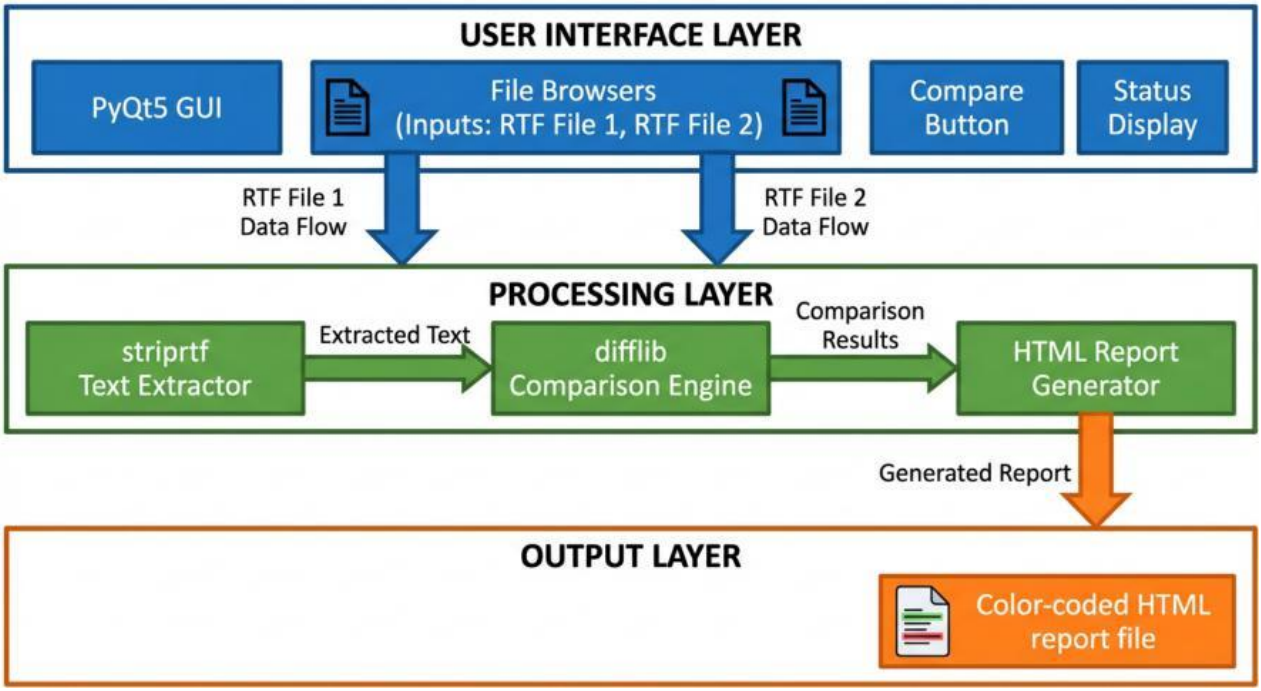

Figure 2: System Architecture: The application consists of three layers: (1) User Interface Layer providing the PyQt5 GUI, (2) Processing Layer containing text extraction and comparison modules, and (3) Output Layer generating the HTML comparison report.

The application comprises the following layers:

- User Interface Layer: Implements the PyQt5 graphical user interface
- Processing Layer: Encompasses text extraction and comparison modules
- Output Layer: Generates detailed HTML comparison reports

USER INTERFACE LAYER

The graphical user interface is built with PyQt5, making it work on Windows, macOS, and Linux. It looks native on each platform, uses standard file selection dialogs, and responds naturally to user actions. The interface includes two file selection fields with "Browse" buttons, a "Compare" button, and a status bar showing progress.

PROCESSING LAYER

This layer consists of two principal modules.

• Text Extraction Module (striprtf): Converts RTF documents to plain text by parsing RTF control codes. It handles complex and nested formatting better than regular expression methods.
• Comparison Engine (difflib): Uses the Ratcliff/Obershelp algorithm to detect fine-grained differences. The SequenceMatcher class highlights insertions, deletions, and changes at the word level.

OUTPUT LAYER

The output layer generates HTML reports with distinct color coding for ease of review.

- Green: Inserted text present only in the new version
- Red with strikethrough: Deleted text present only in the old version
- Black: Unchanged content

PRACTICAL APPLICATION:

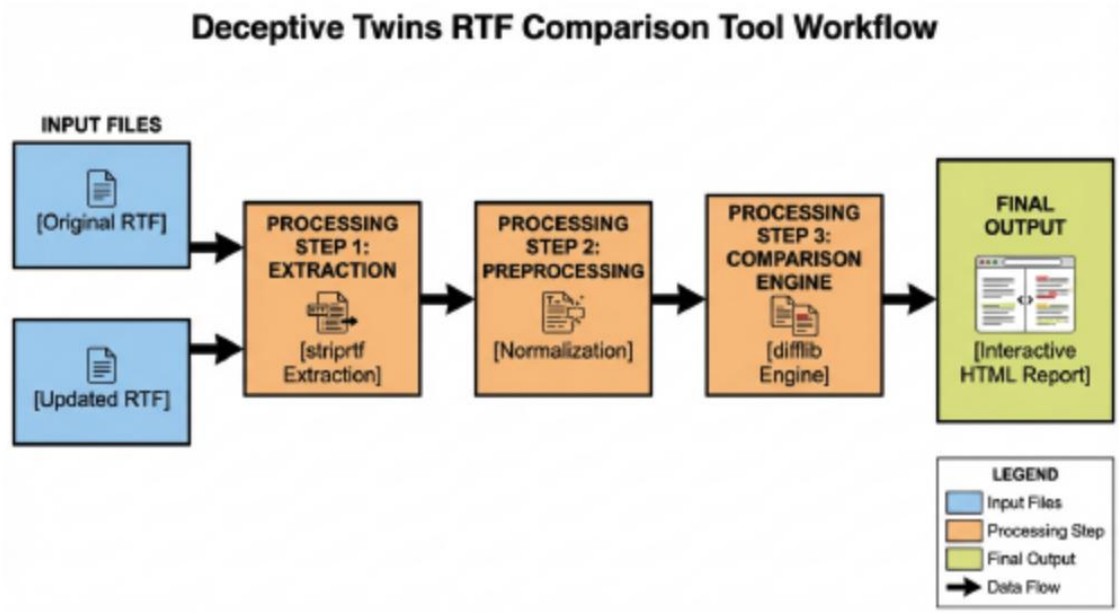

Figure 3: Practical Application: Step1- striprtf extraction, Step2 - Preprocessing, Step3- Comparison engine (difflib engine), HTML output.

The application window divides into three functional areas. The left panel contains mode selection controls (folder vs. file comparison) and path selectors for old and new versions. A comprehensive options section provides checkboxes for customizing comparison behavior, including case sensitivity, whitespace handling, blank line treatment, line numbering, and change-only highlighting. The right panel displays inline comparison results with color-coded differences, allowing immediate visual inspection without leaving the application.
For single-file comparisons, users select the "Compare Files" mode and browse to select old and new RTF versions. After clicking Compare, the tool extracts text, applies selected options, and displays results in the right panel. If differences exist, they appear color-coded with character-level highlighting showing precisely what changed within modified lines.

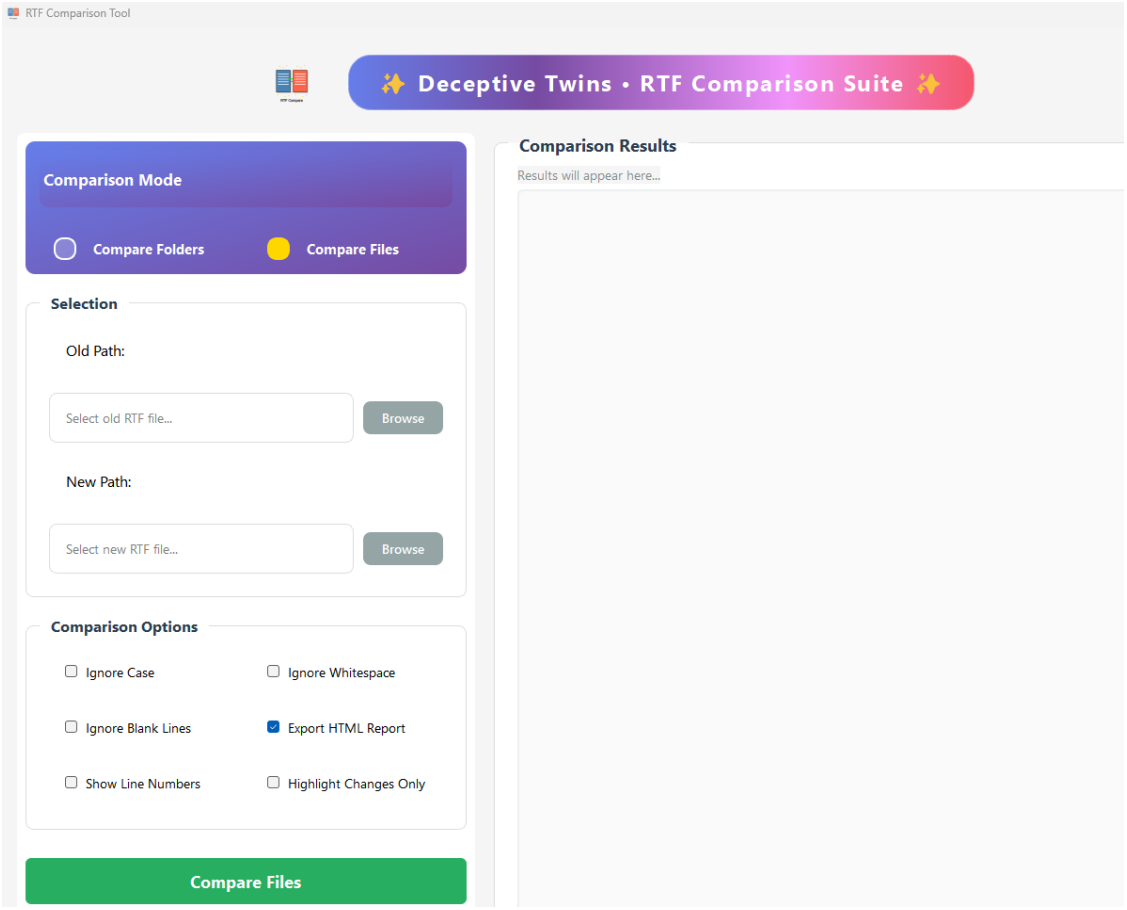

Figure 4: User Interface: The Deceptive Twins application window provides simple file selection and one-click comparison functionality.

Folder mode enables batch comparison of multiple RTF files, matching files by name between old and new directories. The tool iterates through all RTF files in the old folder, locating corresponding files in the new folder and comparing them sequentially. Results aggregate into a single scrollable view with file headers separating each comparison.

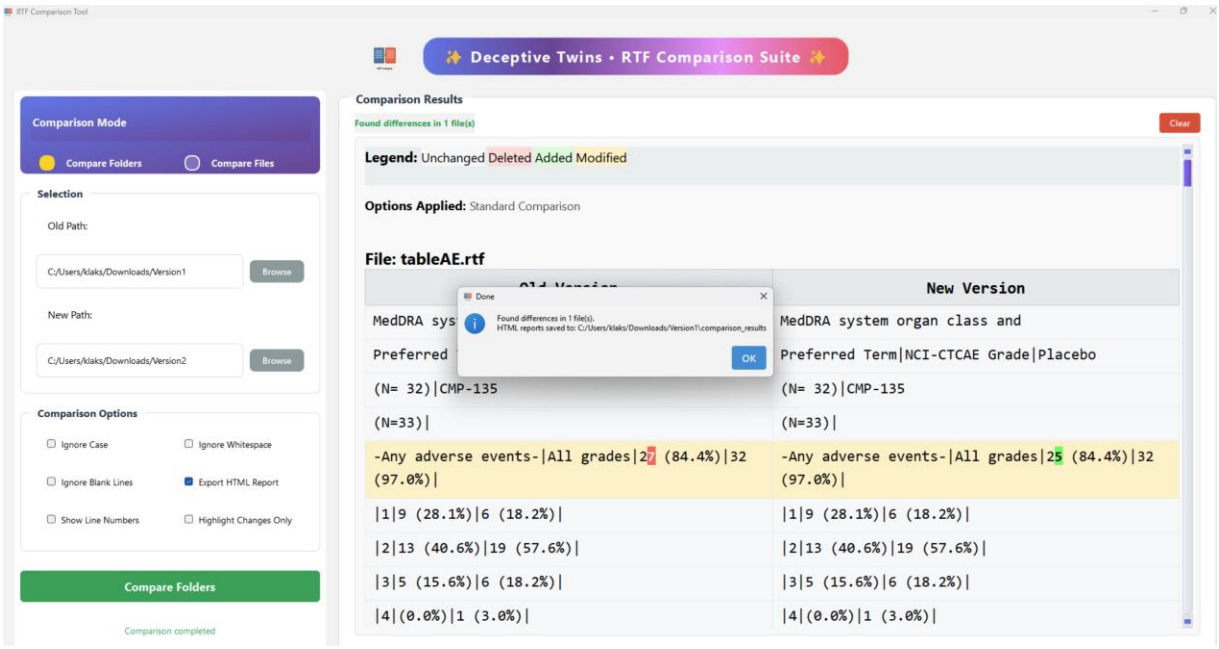

Figure 5: User Interface: File vs Folder mode selection, Comparison options, Immediate visual feedback without opening the browser, HTML file autosaves to the location.

When the "Export HTML Report" option is enabled, the tool generates a standalone HTML file in a timestamped subdirectory named "comparison results" adjacent to the source files. This report opens automatically in the default web browser, providing a shareable artifact for documentation and review.

A summary counter reports the total number of files with detected differences. Individual HTML reports generate for each changed file when export is enabled, allowing reviewers to focus on specific files of interest. Files present in the old folder but missing from the new folder receive special notation, alerting users to potentially deleted outputs.

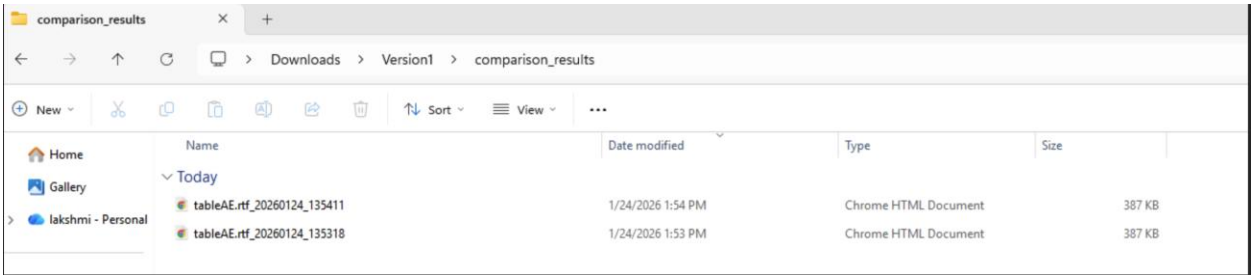

Figure 6: Comparison results: The comparison results save as individual HTML files in the folder.

The HTML structure uses a four-column table layout with line numbers and content for both old and new versions. Color coding applies at the table cell level, with character-level highlighting provided by nested span elements for modified lines.

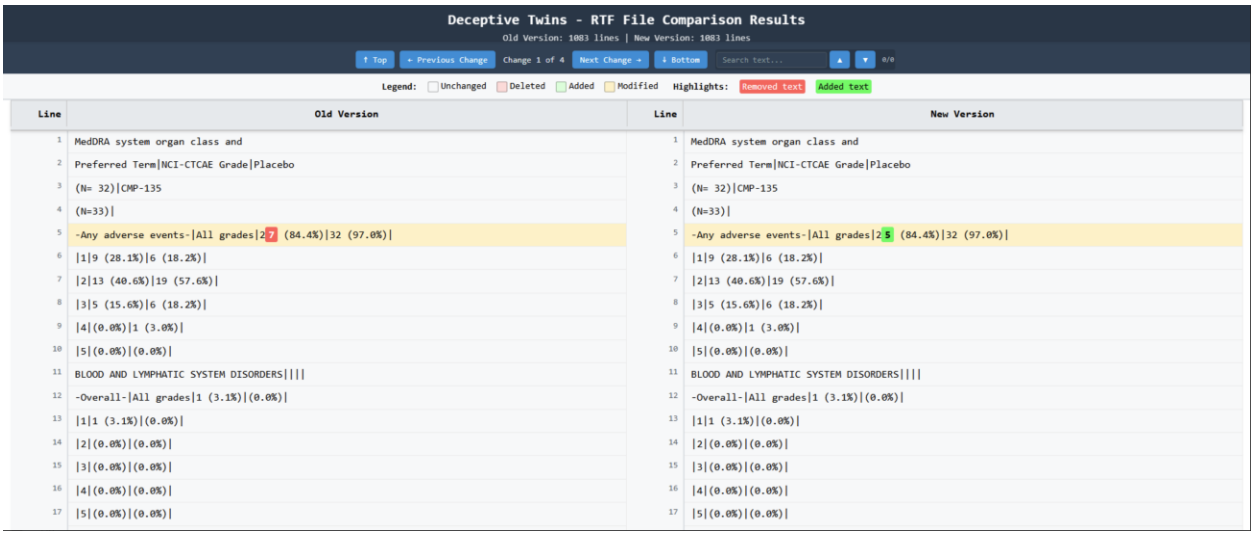

Figure 7: Sample Output: A color coded HTML report showing insertions (green), deletions (red), and unchanged text (black) in a Adverse event table.

CONCLUSION

Deceptive Twins demonstrates how targeted automation can meaningfully improve quality and efficiency in clinical trial programming. By pairing an intuitive PyQt5 interface with reliable text extraction and comparison techniques, the solution shifts RTF review from a repetitive manual task to a streamlined, repeatable workflow. More importantly, it enables programmers and reviewers to focus on what truly matters understanding and validate meaningful change supporting higher-quality outputs and more confident decision-making across clinical reporting and regulatory submissions.

Key benefits include:

• 70–90% reduction in document review time
• Fewer missed changes through systematic word-level analysis
• Permanent, auditable QC documentation via HTML reports
• Works across platforms, fitting into diverse IT environments

This tool shows the value of purpose-built solutions in regulated industries. Deceptive Twins is designed specifically for clinical programming: it handles RTF files, allows customizable comparison logic, can process entire folders, and presents clear visual differences between documents.

As regulatory submissions grow in volume and complexity, automated document comparison will become essential for maintaining quality standards, meeting project timelines, and controlling clinical trial costs. Clinical programming teams using Deceptive Twins see measurable improvements in QC efficiency, error detection, and documentation quality.

Future enhancements will include support for additional document formats. Its open architecture, built on standard Python libraries, allows teams to customize it for their specific needs.

REFERENCES

1. Python Software Foundation, “Python Language Reference.” Online: https://docs.python.org/3/reference/ .
2. Python Software Foundation, “difflib — Helpers for computing deltas.” Online: https://docs.python.org/3/library/difflib.html.
3. Riverbank Computing Limited, “PyQt5 Reference Guide.” Online: https://www.riverbankcomputing.com/static/Docs/PyQt5/ .
4. striprtf developers, “striprtf (Python package).” PyPI. Online: https://pypi.org/project/striprtf/.

ACKNOWLEDGMENTS

The authors wish to thank colleagues for their valuable collaboration and support. We also acknowledge the PHUSE community for fostering knowledge exchange and providing useful resources. Special thanks to reviewers for their constructive feedback during paper development.

CONTACT INFORMATION

Aravind Gajula
Bristol Myers Squibb (BMS)
Lawrenceville, New Jersey, USA
Aravind.gajula@bms.com

Indraneel Narisetty
Gilead Sciences
Foster City, California, USA
Indraneel.Narisetty@gilead.com