PharmaSUG China 2025 - Paper AA-111

MediSum: An R Shiny-based Web Platform for Automated Analysis and Reporting of EDC Data in Clinical Trials

Jingya Wang, Jiangsu Hengrui Pharmaceuticals Co. Ltd.

ABSTRACT

Clinical data review requires timely access to critical analyses, including subject disposition, adverse event (AE) profiles, oncology efficacy endpoints (e.g., PFS/ORR/DCR), and survival analysis (KM curves). Traditional workflows relying on manual calculation or SAS programming for EDC data processing consume at least 2-3 days per tumor project for these analyses.

To address the growing data volume and urgent medical review needs, we developed MediSum – an R Shiny-based web platform featuring: 1) Direct parsing of EDC raw data with automated ADaM-like dataset generation; 2) Interactive visualization modules enabling dynamic subgroup filtering (e.g., by treatment arm/baseline characteristics) and real-time updates of waterfall/swimmer plots; 3) Standardized output templates (DOCX/PNG/SVG) compliant with Oncology Mockup Shell specifications. The platform also ensures full traceability from raw EDC data to final reports by providing intermediate outputs—including ADaM-like datasets and analysis-related listings—accessible via one-click XLSX downloads.

Currently deployed on Hengrui's enterprise servers, MediSum has reduced these summary analysis times from days to only 15 minutes and has generated 9000+ analysis reports across 100+ different clinical trials in the first half of 2025.

INTRODUCTION

Medical review serves as the cornerstone for ensuring data integrity and patient safety in clinical trials, particularly in oncology studies where adverse event (AE) profiles and tumor response metrics directly impact regulatory submissions. These analyses are critical during safety monitoring committee meetings, interim database locks, and protocol amendment discussions, requiring immediate access to dynamically updated summaries such as MedDRA-coded AE distributions and stratified survival endpoints (e.g., by tumor type or biomarkers). However, traditional workflows struggle to meet the demands for rapid analyses and data updates.

To bridge this gap, we developed MediSum—an automated EDC-to-report workflow platform for oncology data review. The platform leverages R Shiny, an interactive web application framework, which offers a transformative solution by providing interactive user interfaces that utilize predefined analytical frameworks and data structures. Through Shiny applications, study-specific parameters are extracted to enable real-time user adjustments. Furthermore, the applications provide analysis templates and visualization styles developed for common medical review needs.

This paper introduces MediSum's development techniques, internal workflow and designs, its functional capabilities (including user-configurable parameters, templates, and styles), along with current limitations and future optimization strategies.

WORKFLOW AND DESIGNS

To achieve end-to-end automation from raw data ingestion to interactive insights, we architected the platform as three functional layers:

1. DATA INTEGRATION LAYER

Our data integration layer offers three coordinated pathways for flexible clinical data ingestion.

First, we provide two EDC raw data acquisition methods: API connections to a secondary EDC data storage system or manual file upload. API connections enable the system to automatically fetch EDC and AE coding data based on user permissions. For manual uploads, users directly upload project data files.

Second, we develop a Custom Data Upload pipeline for supplementary subgroup integration. When users upload Excel files containing subject-level variables, the system performs automated validation, verifying mandatory SUBJID fields and preventing naming conflicts with core datasets. Validated subject-level variables become analytical dimensions for stratified analysis (used in group summarization or comparison), while invalid uploads generate diagnostic warnings to facilitate corrections.

Third, we implement cutoff date configuration. This feature dynamically segments datasets by study phases or snapshots, with boundary validation preventing illogical date ranges. All configurations remain editable before final submission. Users initiate data merging and trigger downstream processing engines by clicking the Generate button. Figure 1.1 is a flowchart for Data Integration Layer.

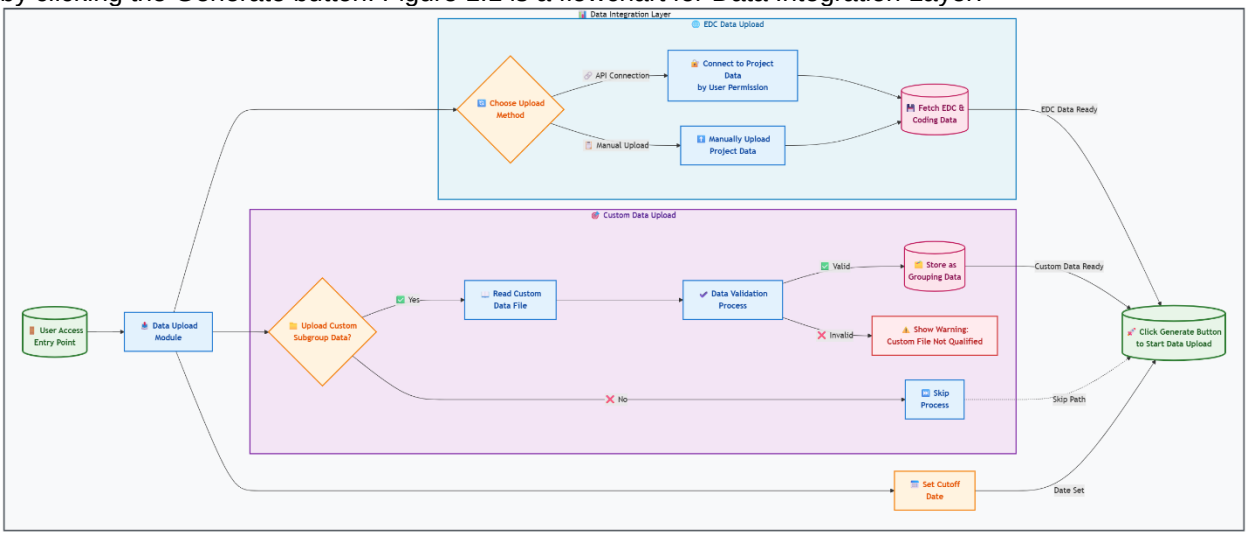

Figure 1.1 Data Integration Layer Workflow

2. DATA PROCESSING LAYER

We design an adaptive ADaM-like dataset generator that dynamically builds analysis-ready datasets through a multi-stage workflow. For oncology studies, users configure protocol-specific parameters like cycle duration, TEAE windows, and SD/PR confirmation periods; non-oncology studies utilize default settings. Figure 3.4 shows the configuration interface. Our engine then sequentially generates nine core ADaM domains: ADSL, ADAE, ADPR, ADMH, ADCM, ADRS, ADTR, ADRESP, and ADTTE.

Before generating each domain, we verify prerequisite variables through dependency checks using specification files designed from CRF design standards and basic ADaM data generation rules. Successful generation triggers automatic storage and initiates the next domain's processing, while unmet conditions generate domain-specific diagnostic warnings displayed on the web interface for user visibility. Variable derivation and cutoff handling strictly follow the company's generally accepted SAP standards. The entire process completes validation and ADaM dataset generation in <20 seconds, delivering analysis-ready datasets with real-time logs and export options. Figure 1.2 is a flowchart for Data Integration Layer.

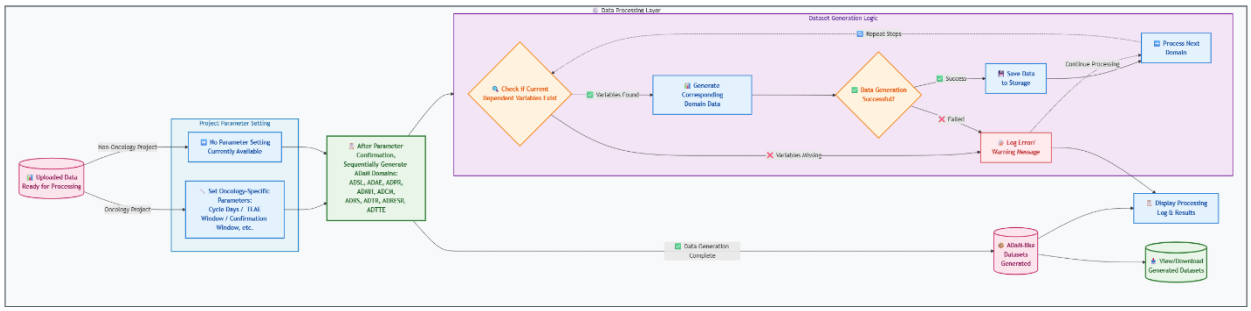

Figure 1.2 Data Processing Layer Workflow

3. ANALYSIS & VISUALIZATION LAYER

The website UI is designed primarily with bs4Dash and related Shiny packages (e.g., shinyWidgets, shinyalert, shinyjs). We use the {golem} package framework and implement a modular architecture where each visualization functions as an independent Shiny module; this structure allows seamless addition of new modules.

When user accesses any module, the system first verifies the existence of required ADaM datasets (e.g., ADSL for demographic summaries), triggering real-time alerts if dependencies are unmet. Only after successful validation the analysis interface will be launched, providing users with clear visibility of available functional modules. Figure 1.3 is a flowchart for Interactive Visualization Layer.

All modules include essential controls: a filtered/non-filtered data toggle, an analysis set selector, a grouping variable selector, and table/plot export options. For specialized functionality, module-specific enhancements are provided—for example, AE table modules incorporate SAE/TRAE filters, incidence cutoffs, and custom PT filters. Additionally, visualization styles and detail adjustment tools are added to all plot modules, and subgroup annotations are included in waterfall and swimmer plots.

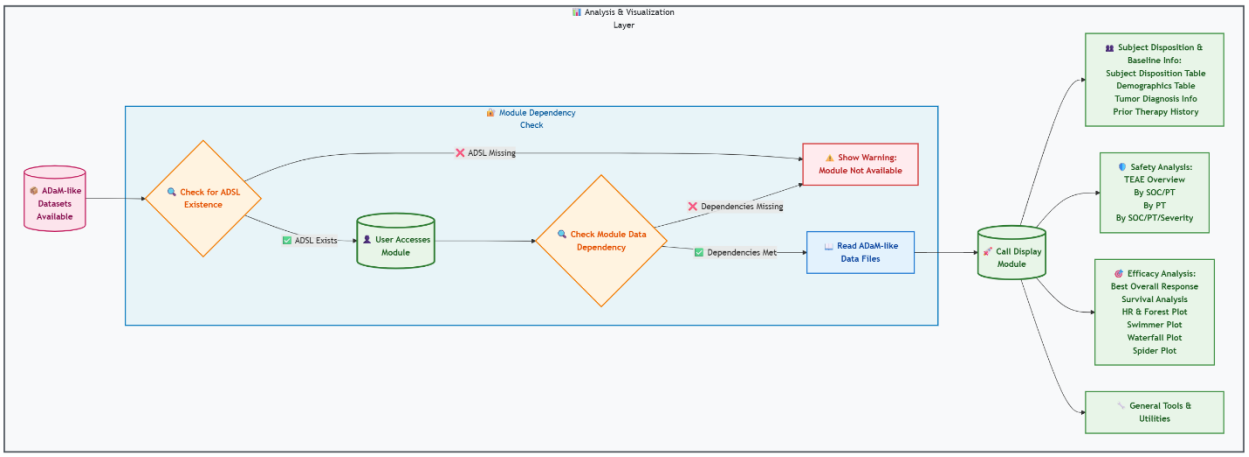

Figure 1.3 Analysis & Visualization Layer Workflow

This stratified design ensures computational isolation—each layer handles specialized tasks (data ingestion, transformation, visualization) while maintaining seamless interoperability through automated handoffs.

PLATFORM OVERVIEW

MAIN INTERFACE OVERVIEW

Figure 2.1 is the main interface of MediSum, and it consists of four areas. The left navigation bar (blue box part) facilitates the display and switching of analysis views. The central workspace (green box part) presents individual analysis module operations and displays generated charts. The right panel (orange

box part) provides specialized tools - ADSL-based subject filtering and Adverse Event (AE) encoding auto-completion. The top toolbar (red box part) offers supplementary resources, including feedback submission, historical update logs and links platform usage guides.

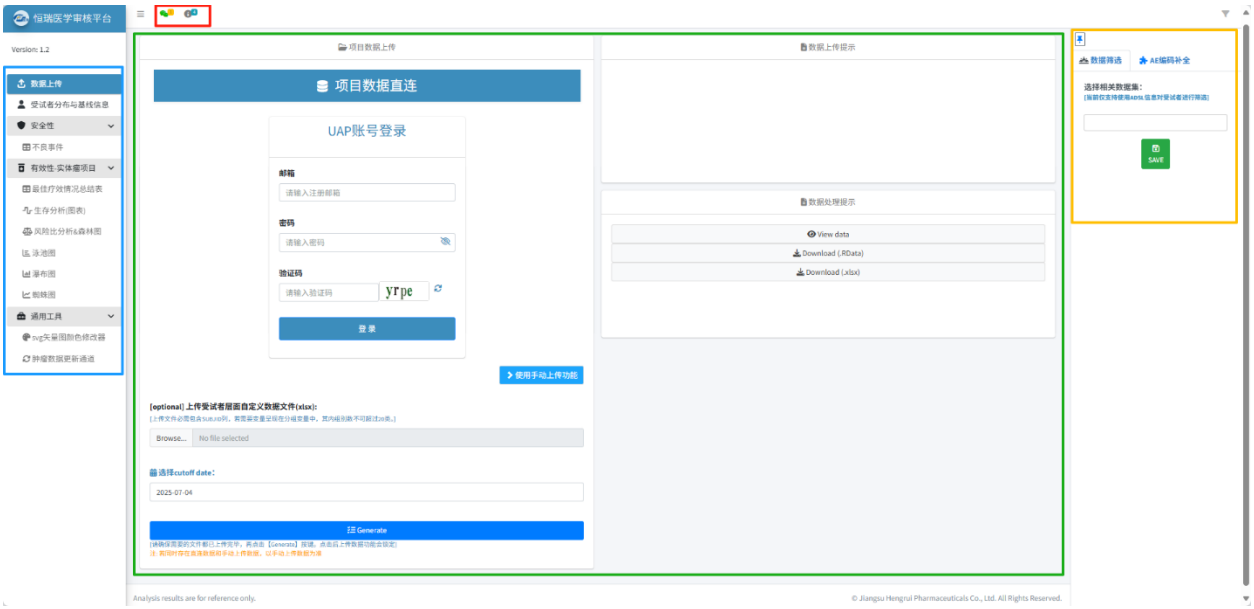

Figure 2.1 MediSum Main Interface

FOLDER STRUCTURE OVERVIEW

As described, MediSum organizes into three functional layers, each supported by dedicated code modules.

In Data Integration Layer, we manage EDC data imports from sources like BioKnown EDC systems. We implement functions to validate file integrity and convert inputs to RData format for downstream processing. Raw data ingestion is handled by 3 Shiny module files (e.g., mod_01_DataUpload.R) supported by 3 related function files (mod_01_DataUpload_fct_readEDCtxt.R for EDC reading, mod_01_DataUpload_API_fct_loaddata.R for specific version fetching, mod_01_DataUpload_API_fct_logic.R for user login).

In Data Processing Layer, we implement 11 R functions to generate standardized ADaM-like datasets (e.g., fct_gen_adsl.R for subject-level data, fct_gen_adae.R for adverse events), supplemented by critical supporting functions (e.g., fct_bor_confirm_resist.R for tumor response logic, fct_simple_adam_gen.R for simple dependency variable extraction). Additionally, we include utility functions for extracting dependency relationships from JSON specifications (utils_spec_related.R) and validating variable dependencies (utils_var_dependency.R).

In Analysis & Visualization Layer, we employ a modularized structure using 16 Shiny module files to provide interactive Tables, Figures, and Listings (TFLs). Examples include mod_32_KM.R for survival analysis, mod_34_waterfall.R for waterfall plots, and mod_35_spider.R for spider plots. To ensure standardized outputs, we provide 3 formatting function files (e.g., fct_ggplot_mycolor.R for color themes, fct_hr_rtf_style.R for RTF styling).

The core application files (app_ui.R, app_server.R, app_config.R) act as the central integration hub, dynamically orchestrating modules across all three layers to form the complete MediSum application. Within these files, we implement dynamic module integration through programmatic calls (e.g., callModule(mod_35_spider_server, ...)), and consolidate data integration, processing, and analysis workflows into a unified application framework. This design ensures seamless interaction between layers while maintaining structural integrity. Figure 2.2 show the files and folder structure for MediSum.

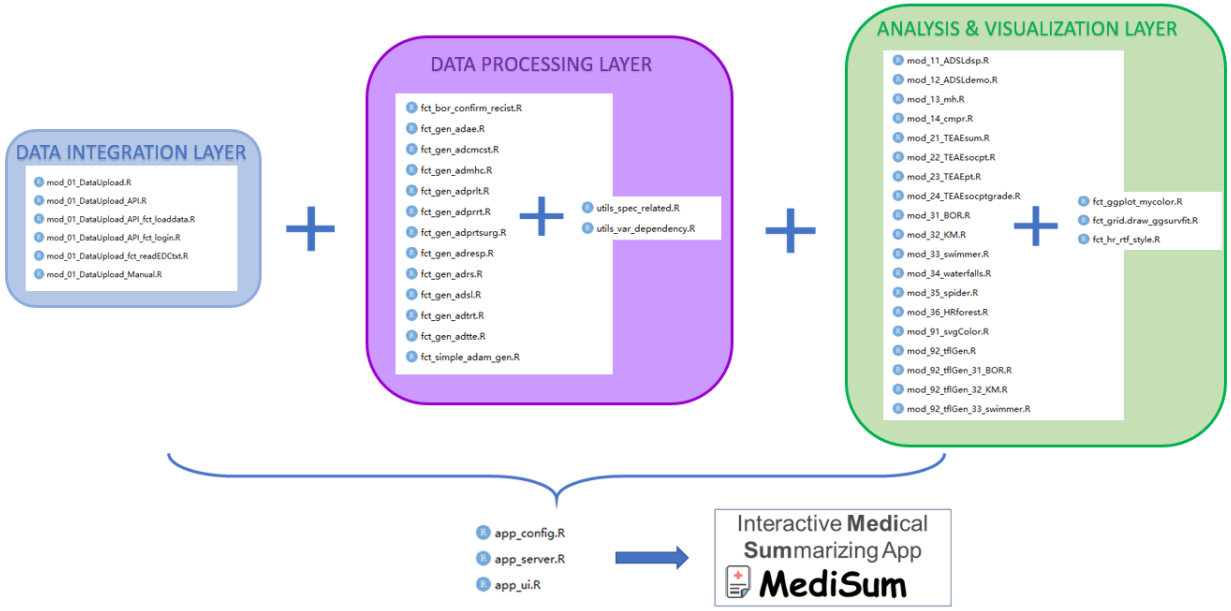

Figure 2.2 MediSum Folder Structure

MODULE STRUCTURE OVERVIEW

The MediSum platform enforces strict modular design principles across its analytical components.

UI components (see Figure 2.3) implement mandatory interactive controls including a binary switch (switchInput("filter_yn")) for toggling filtered/non-filtered data sources, a dynamic selector (pickerInput("ds_flag")) for choosing analysis sets (SAF/EES/FAS), and a hierarchical variable picker (pickerInput("var_groupby")) supporting clinical grouping parameters such as treatment arm ("TRT01A"). These are complemented by essential action triggers: an analysis execution button (actionBttn("generate")) initiates computational workflows, while format-specific download handlers (downloadBttn("download_*")) enable SVG visualization exports alongside RTF/DOCX document outputs. Supplementary UI elements incorporate dropdown selectors (pickerInput()), vertically aligned checkbox groups (checkboxGroupButtons()), and context-sensitive conditional displays (conditionalPanel()) that dynamically expose advanced options based on user selections.

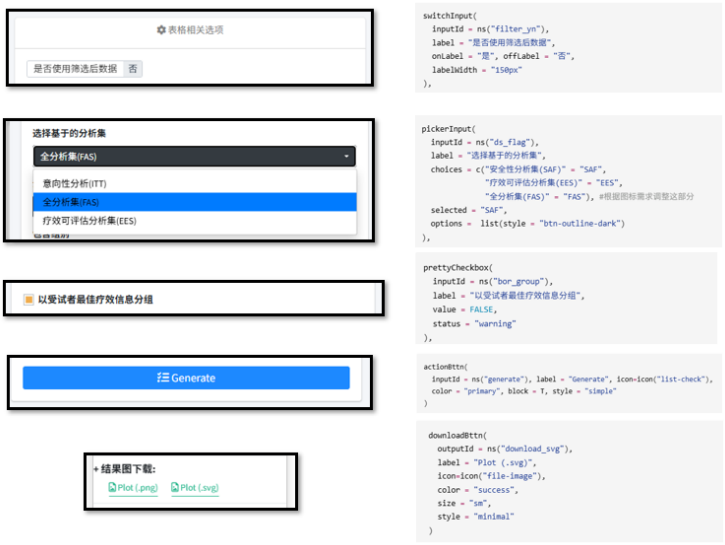

Figure 2.3 UI Components Examples

Server-layer processing follows a rigorous three-phase reactive architecture (see Figure 2.4). This sequence ensures deterministic data handling while maintaining runtime stability through error isolation.

```r
mod_xx_XXXX_server <- function(input, output, session, nofilter, filter) {
    ## Step 1: Data Preprocessing ----
    adsl <- reactive({
      ### Handles filtered/non-filtered data selection & analysis set flagging
    })
    adxx <- reactive({...})
    table_data <- reactive({
      ### Data merging via left_join
    })

    ## Step 2: Interactive Component Management ----
    observeEvent(table_data(), {
      ### Dynamically updates UI options based on dataset characteristics
    })

    ## Step 3: Analysis Execution Pipeline ----
    observeEvent(input$generate, {
      ### Loads TFL specification if applicable
      tryCatch(  # Error handling for analysis failures
        {
          ### 1. Core computation & output generation
          ### 2. Dynamic download filename configuration
          ### 3. Interactive table rendering
          output$tableout <- renderUI({})
          ### 4. Document export handler
          output$download_docx <- downloadHandler()
        },
        error = function(e) {
          message('Error: Failed to generate xx_XXXX (code issue)')
          message(conditionMessage(e))
          shinyalert::shinyalert(...)
        }
      )
    })
  }
```

Figure 2.4 Server Part Template

TFL specification governs analytical outputs via JSON-based configuration files (see Figure 2.5), which structurally define table metadata and ADaM-like dataset dependencies.

```json
{
    "type": "Table",
    "category": "Part 1: Subject Disposition and Baseline",
    "Number": "T1_2",
    "title": "Demographic Characteristics",
    "footnote1": "N is the number of subjects of each treatment group included in the analysis set. Percentage is calculated only if the numerator is not 0, with N as denominator.",
    "footnote2": "BMI = weight (kg)/ (height (cm)*0.01){^2}",
    "adamDataDependency": {
        "SEX": {
            "datasetName": "ADSL",
            "variableName": "SEX",
            "label": "Sex n(%)",
            "type": "freq",
            "required": "Y"
        },
        "AGE": {
            "datasetName": "ADSL",
            "variableName": "AGE",
            "label": "Age (years)",
            "type": "stat",
            "required": "Y"
        },
        "AGEGR1": {
            "datasetName": "ADSL",
            "variableName": "AGEGR1",
            "label": "Age (years) Group n(%)",
            "type": "freq",
            "required": "Y"
        },
        "ETHNIC": {
            "datasetName": "ADSL",
            "variableName": "ETHNIC",
```

Figure 2.5 Table JSON-Configuration File Template

Critical parameters include:

- **adamDataDependency**: mapping clinical variables to analysis functions
- **required**:"Y" flags enforcing mandatory field inclusion
- **type**:"freq" declarations specifying statistical computation methods

This declarative approach abstracts clinical rules from code logic, enabling standardized table generation across therapeutic domains.

Output standardization employs dual rendering pathways: The tern/rtables workflow utilizes export_as_docx(..., section_properties=landscape()) for publication-ready document generation, while custom data.frame pipelines apply hr_rtf_style() for regulatory-compliant RTF formatting. Visual outputs strictly adhere to centralized color management through fct_ggplot_mycolor.R implementations, mandating usage of scale_fill_cvi_d("mycol") for discrete scales and scale_colour_cvi_c("mycol") for continuous gradients to ensure brand-compliant visualization aesthetics.

This integrated framework establishes a reproducible analytical environment where clinical configuration drives computational execution, enabling consistent production of regulatory-grade outputs.

KEY FUNCTIONAL DEMONSTRATIONS

DATA UPLOAD & PROCESSING

We offer two distinct data upload options:

1. **Authenticated Upload:** Users log in with their credentials. Upon login, the platform retrieves project permissions associated with the user's account and provides access to the corresponding project's EDC raw data and associated AE coding data. Users can optionally select a specific version date for analysis (see Figure 3.1).

Figure 3.1 Authenticated Upload Interface

2. **Manual Upload:** Users select and upload project data files directly from their local machine (see Figure 3.2).

Figure 3.2 Manual Upload Interface

Additionally, we also allow for the upload of custom subject grouping data and a selection of cutoff-date for analysis. Following data selection/upload, clicking Generate initiates the platform's data processing workflow. During this stage, the platform imports and processes the user's data while displaying a progress bar (see Figure 3.3) to indicate ongoing operation.

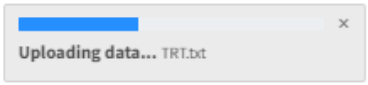

Figure 3.3 Upload Progress Bar

Upon successful data ingestion, the system provides an upload confirmation message verifying the data's compatibility with platform requirements. Concurrently, a dedicated configuration interface (see Figure 3.4) becomes available, allowing users to define protocol-specific parameters such as cycle duration, TEAE windows, and SD/CRPR confirmation windows.

Figure 3.4 Project Parameter Configuration Interface

Once the data generation completes, the Data Processing Notification interface (see Figure 3.5) clearly lists the processing outcomes. This interface also provides the ability to download the generated ADaM-like datasets for further analysis.

Figure 3.5 Data Upload & Processing Notification Interface

SUMMARY TABLE MODULES

MediSum includes multiple summary table modules:

- **Subject Distribution and Baseline Information:**

  Subject Disposition, Subject Disposition at the End of Study, Subject Disposition at the End of Treatment, Reasons for Screen Failures, Tumor Diagnosis at Baseline, Prior Anti-Cancer Therapy.

- **Safety Analysis - Adverse Event Summaries:**

  Overall Summary of TEAEs, Summary of TEAEs by SOC and PT, Summary of TEAEs by PT, Summary of TEAEs by SOC, PT and Maximum Severity/Maximum CTCAE Grade.

- **Oncology Efficacy Analysis:**

  Best Overall Response, Objective Response Rate, Disease Control Rate, Overall Survival, Progression-Free Survival, Duration of Response, Summary of Duration of Survival Follow-up, Risk ratio analysis on OS/PFS/DoR.

In this section, we will introduce the design and implementation of our most frequently used safety analysis page - Summary of Treatment-Emergent Adverse Events by System Organ Class and Preferred Term (SS) table. This table categorizes treatment-emergent adverse events (TEAEs) by MedDRA System Organ Class (SOC) and Preferred Term (PT), with summaries grouped by cohort and the total cohort. See Figure 4.1 for the options interface and result display.

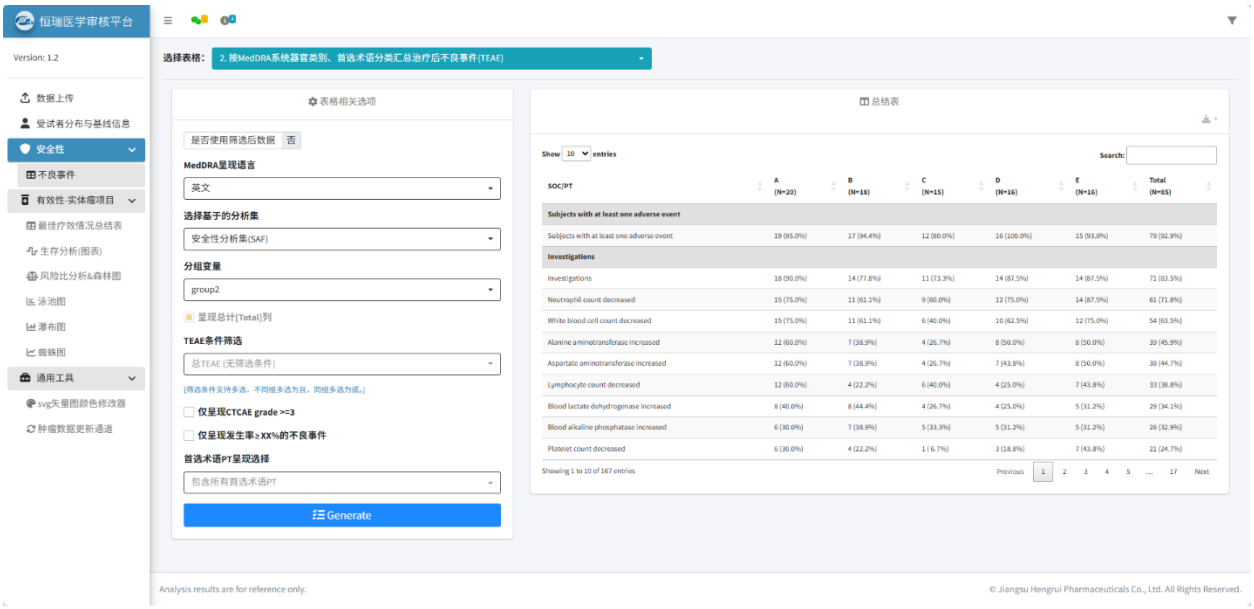

Figure 4.1 Safety Analysis Page - AE SOC/PT

1. Basic Options for All modules

We implement four core controls available across all modules:

- **Data Switch Toggle:** Switches between filtered data (user-defined subsets) and full analysis data.
- **Analysis Sets Selector:** Enables selection of clinical trial populations (ITT, FAS, SAF, EES) for endpoint calculations.
- **Dynamic Grouping Variable selector:** Supports grouping by user-defined variables (e.g., biomarker status/demographic strata).
- **Multi-Format Export:** Generates three report types: RTF Tables, XLSX Summary Tables, XLSX Listings.

2. Module Special Functions

Beyond the above basic options, we add a TEAE condition filter to filter for the different hierarchies in TEAE summary. The default choice is "Total TEAEs (no filters)". Users can combine filter conditions using drop-down menus: selecting options from different groups applies AND logic (all must be satisfied), while selecting options within the same group uses OR logic (any one is sufficient). The drop-down provides four option groups (see Figure 4.2 left):

(1) Severity: serious adverse events, TEAEs leading to death.

(2) Relatedness: related to any drug, related to a specific drug.

(3) Action taken: action taken on any drug, action taken on a specific drug.

(4) Other: includes AESI (Adverse Events of Special Interest), AEIRAE (Immune-Related Adverse Events), and AEDLT (Adverse Events with Dose-Limiting Toxicity), if available in the data.

Figure 4.2 TEAE Condition Filter

If user choose Option3-Action taken, a new drop-down appears for selecting the action type. Similarly, choosing option4-Other adds a drop-down for specific conditions. See Figure 4.2 right part for these new menus, which vary based on the data.

3. User-Feedback-Driven Feature Enhancements

Based on user feedback, we add three key features to this summary table (see Figure 4.3):

- **Display only CTCAE grade ⩾3:** When selected, only adverse events with a CTCAE grade of 3 or higher will be summarized and displayed in this table.
- **Display only adverse events with incidence ⩾XX%:** User can filter this table to PT entries with an incidence rate greater or equal to XX%, which XX can be defined by user. The default is 5%.
- **Preferred Term (PT) selection:** User can select preferred terms (PTs) by partial input and decide which ones to display. The final table shows only the summaries of the chosen PTs and their SOCs.

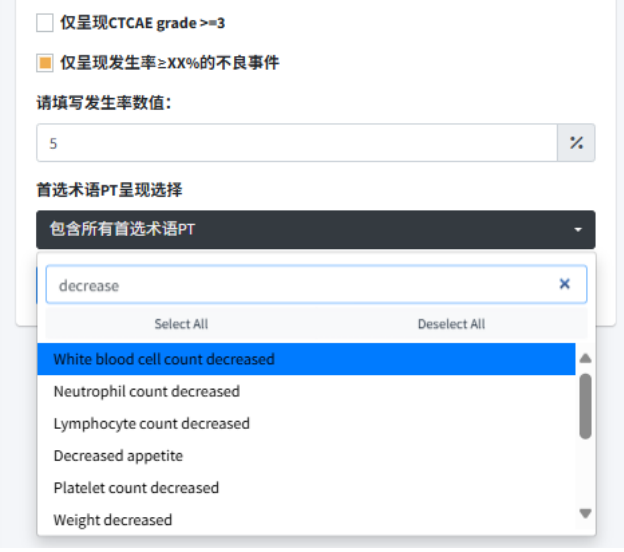

Figure 4.3 User-Feedback-Driven Options

INTERACTIVE PLOT MODULES

MediSum provide the following five types of interactive plot modules:

- Kaplan-Meier Plot of OS/PFS/DoR
- Forest Plot for Subgroup Analysis on OS/PFS/DoR.
- Swimmer Plot of the Duration of Objective Response
- Waterfall Plot of Optimal Percentage Change in Target Lesion Diameter Relative to Baseline
- Spider Plot of Target Lesion Diameter Relative to Baseline

Figure 5.1 displays some outputs of above plots.

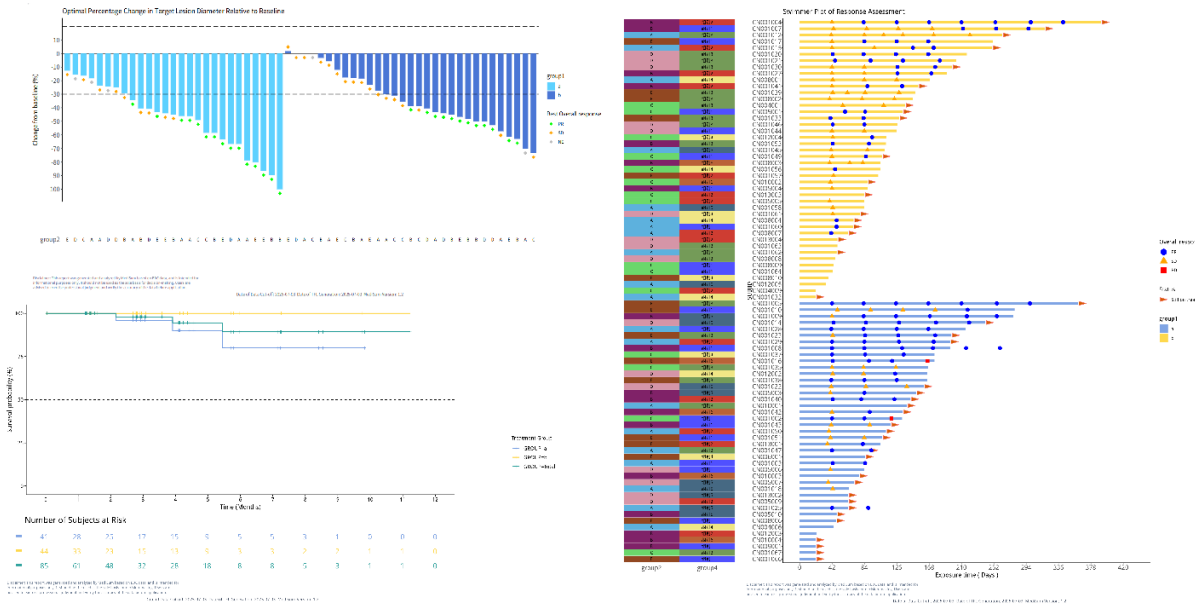

Figure 5.1 Plot Outputs

In this section, we will take Swimmer Plot as an example (see Figure 5.2).

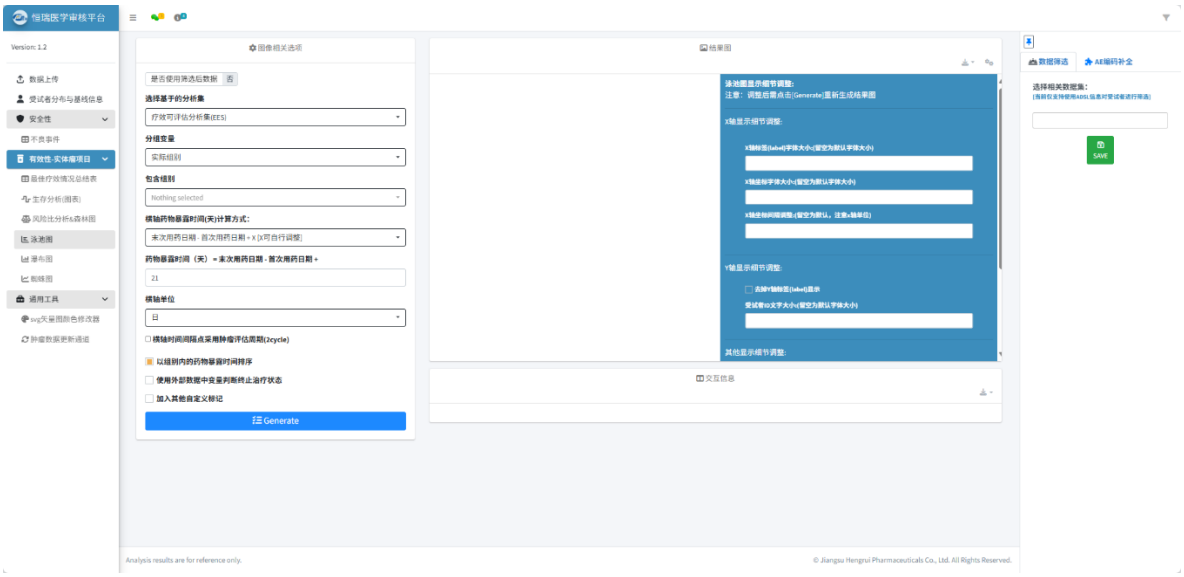

Figure 5.2 Simmer Plot Page

1. Basic Options for All modules

We implement five core controls available across all plot modules:

- **Data Switch Toggle:** Switches between filtered data (user-defined subsets) and full analysis data.
- **Analysis Sets Selector:** Enables selection of clinical trial populations (ITT, FAS, SAF, EES) for endpoint calculations.
- **Dynamic Grouping Variable Selector:** Supports grouping by user-defined variables (e.g., biomarker status/demographic strata).
- **Grouping Visibility Selector:** Provides multi-select dropdown for grouping value selection; defaults to all groups and enables custom plot display configurations.
- **Multi-Format Export:** Generates three report types: PNG/SVG Plots with editable height and width, XLSX Listings.

2. Module Special Functions

Beyond the above basic options, we set many specific options for each plot modules. Here's the special functions in swimmer plot:

- **X-axis Value Options:** Adjustment options for the calculation method of drug exposure time (days) on the x-axis, x-axis unit adjustment, and x-axis coordinate point adjustment. (see Figure 5.3)

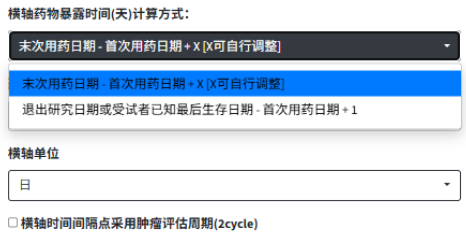

Figure 5.3 X-axis Value Options

- **Presenting Options:** Adjustment option for changing data presenting orders.
- **Customize for End of Treatment status:** Provide multiple adjustment methods for variables determining therapy termination status. (see Figure 5.4)

Figure 5.4 Customize for End of Treatment status Options

3. User-Feedback-Driven Feature Enhancements

During use, we continuously iterate based on user feedback and questions. And here we demonstrate the two common requests for swimmer plots:

- **Adding Subject-level information:** Support adding custom markers to plot based on user-uploaded files.
- **Modify Plot details:** Based on different project, user often require adjustments to text size, axis

interval, angles. Thus, we provide these adjustment options. (see Figure 5.5).

Figure 5.5 Visualization Plot Outputs

These features ensure we provide users with the quickest and most commonly used templates while guaranteeing a certain degree of flexible customization.

CONCLUSION

MediSum provides an end-to-end solution for accelerating clinical data review in oncology trials. By architecting an R Shiny-based platform with three specialized layers—Data Integration, Processing, and Visualization—we enable automated transformation of EDC raw data into ADaM-like datasets and interactive reports within 15 minutes. The platform's modular design supports real-time subgroup filtering, dynamic visualization, and compliance with Oncology Mockup Shell standards. Future work will focus on enhancing cross-trial compatibility, developing personalized analytical features, and extending platform adaptability to additional therapeutic areas and non-oncology clinical trials—addressing evolving medical review and summarization needs.

REFERENCES

Becker G, Waddell A, Garolini D, de la Rua E, Zhu J (2025). rtables: Reporting Tables. R package version 0.6.13, https://github.com/insightsengineering/rtables.

Miller E, Stackhouse M, Tarasiewicz A (2024). Tplyr: A Traceability Focused Grammar of Clinical Data Summary. R package version 1.2.1, https://github.com/atorus-research/Tplyr.

Granjon D (2025). bs4Dash: A 'Bootstrap 4' Version of 'shinydashboard'. R package version 2.3.4, https://github.com/RinteRface/bs4Dash.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Jingya Wang
Jiangsu Hengrui Pharmaceuticals Co. Ltd.
Jingya.wang@hengrui.com