Paper OS19

# Future of Data Surveillance in Clinical Studies: A comprehensive approach on data monitoring using R Shiny application for study health

Ruchitbhai Patel, Daiichi Sankyo Inc, Basking Ridge, USA
Jacky Jin, Daiichi Sankyo Inc, Basking Ridge, USA

## ABSTRACT

The Data Surveillance R Shiny Application enhances study health monitoring by integrating dynamic data processing with standardized reporting modules. It is designed to improve operational efficiency and ensure consistency in generating customized data surveillance reports across multiple studies. The application's data processing layer creates standardized input datasets from various sources, including SDTM, ADaM, and their combinations. Custom-built R functions utilize these standardized datasets to produce interactive summary reports and advanced visual analytics, requiring updates only to the data processing layer for new studies. This approach simplifies report generation across multiple studies while maintaining flexibility. With several interactive R Shiny features, the application also supports on-demand custom report generation, aiding in assessing study health and evaluating impacts on key outcomes. Additionally, an in-built robust administration layer enables controlled user access, ensuring data security and regulatory compliance in clinical research.

## INTRODUCTION

Effective data surveillance is essential in clinical research to maintain study integrity and quality. The Data Surveillance R Shiny Application addresses this need with a robust administration layer that regulates user access, ensuring data security and regulatory compliance. It integrates diverse input data sources, including SDTM and ADaM, to create standardized datasets that serve as the foundation for analysis. The data engineering layer processes these inputs, facilitating the generation of customizable and interactive summary reports. By streamlining this workflow, the application enhances monitoring capabilities and supports timely decision-making across multiple clinical studies.

## ADMINISTRATIVE LAYER CONFIGURATIONS

The Administrative Layer is implemented using the *shinymanager* package, which creates a landing page requiring user credentials for access. Its built-in account management functionality facilitates access control, user management, and monitoring of application usage. With *shinymanager*, the username can be extracted during a session, enabling the determination of study access for the active user. If the extracted username matches the study access information stored in a CSV file, the user will see the corresponding studies listed on the dashboard after login.

As illustrated in Figure 1, the landing page of the R Shiny application presents a list of studies accessible to the user, based on their specific access permissions. Upon selecting a study, the application dynamically adjusts the post-processing workflows and analysis to align with the parameters of the chosen study. This design ensures that the user's actions are contextually relevant and tailored to the selected study, improving the precision and efficiency of subsequent data handling.

After a study is selected from the dashboard, the application generates study-specific outputs, utilizing configuration data from the corresponding config.json, study_para.R, and adam_functions.R files. These files contain critical information such as study-specific parameters, analysis functions, and settings, which guide the data processing pipeline to ensure accuracy and consistency in the results.

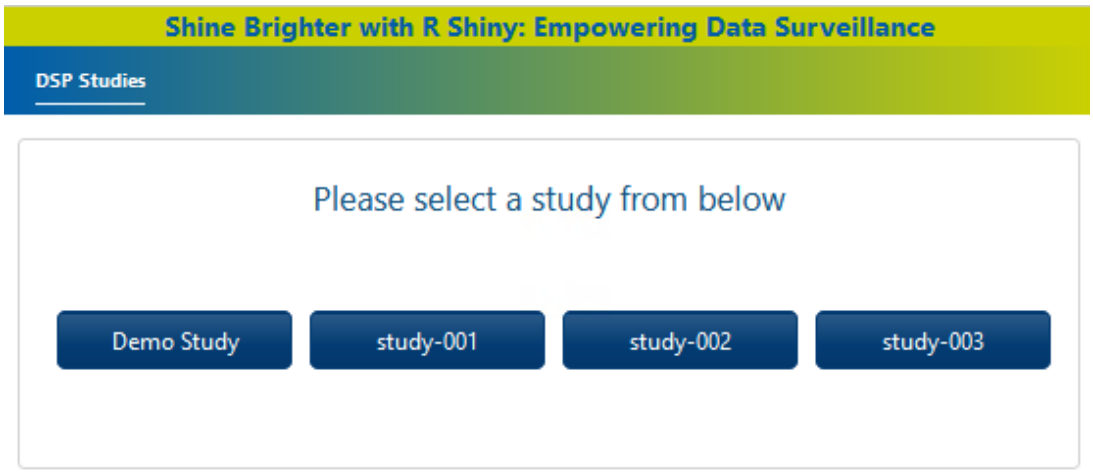


**Figure 1:** Landing page of the R Shiny application following successful user authentication (example for illustration purposes).

## CONFIGURATION AND STUDY PARAMETER LAYER

Processing the configuration and the study parameter layers triggers study-level implementation. The configuration layer defines the input data folder path and contents, the output folder path, and the list of study-specific custom reports. The study parameter layer defines global variables, such as population flags, subgroup variables, and study-specific parameters. Together, these layers ensure flexibility within a highly standardized framework.

### CONFIG.R AND CONFIG.JSON

A `config.R` file is generated for each individual study. The contents of the `config.R` file are then saved as a `config.json` file within the predefined folder structure. Upon selection of a study by the user in the dashboard interface, the application begins executing the `config.json` file. This process reads the input datasets and keywords within the active session, with the availability of input datasets and keywords being restricted to the duration of the session.

### STUDY_PARA.R

Within the `stuyd_para.R` file, the `rlang` programming package is utilized to store study-specific programming requirements. This approach ensures study-specific programming consistency within the application for each study. Although the `stuyd_para.R` is used to define default values for reports or study-specific filtering conditions, it can be adapted to accommodate any other study-specific requirements.

## DATA ENGINEERING LAYER

The data engineering layer is a collection of ADaM functions designed to ingest SDTM and/or ADaM datasets and generate ADaM datasets that are compatible with Shiny report functions. If the input dataset is SDTM, only the required ADaM datasets with the necessary variables are generated. If the input consists of ADaM datasets, the data engineering layer can be employed to standardize variable names and/or create formatted variables, ensuring seamless compatibility with Shiny report functions.

The data engineering layer standardizes the input data for the shiny report functions, ensuring that the shiny report functions do not require updates to accommodate different input data structures. Study-specific updates can be managed at `config.R`, `study_para.R` and data engineering layer. This approach minimizes the need for modifications to the shiny report functions or shiny modules for study-specific requirements. A key aspect of the data engineering layer is the `adam_functions.R` file, which consolidates all the functions responsible for generating ADaM datasets. This approach enhances the consistency of inputs to Shiny functions and modules. There is one

`adam_functions.R` file per study, and each file can reference a different number of ADaM datasets, tailored to the specific needs of the study.

## FUNCTION AND MODULE

In the Data Surveillance R Shiny Application, functions and modules serve as key building blocks that promote reusability, maintainability, and code clarity. Functions are reusable scripts designed to derive variables or perform calculations. They can be called repeatedly throughout the application to avoid code duplication and simplify complex workflows. Modules are self-contained components that bundle Shiny UI and server logic together. They integrate specific functions into UI components to enable user interactivity. Functions and modules work together to create a clean, modularized framework.

## HOW IT FITS TOGETHER

The Data Surveillance R Shiny Application was designed and developed as a modularized framework, leveraging module-specific namespaces to create self-contained, reusable components and prevent conflicts. User accounts and study access are managed using the R package *shinymanager*. The initialization of global variables and study parameters triggers study-level implementation, starting with data processing, followed by the utilization of functions and modules to create interactive Shiny outputs. The `main.R` script serves as the central coordinator, orchestrating the integration of all modules into a unified Shiny application. Figure 2 demonstrates the integration of the different components discussed earlier, showing how they are connected and function together within a single R Shiny application.

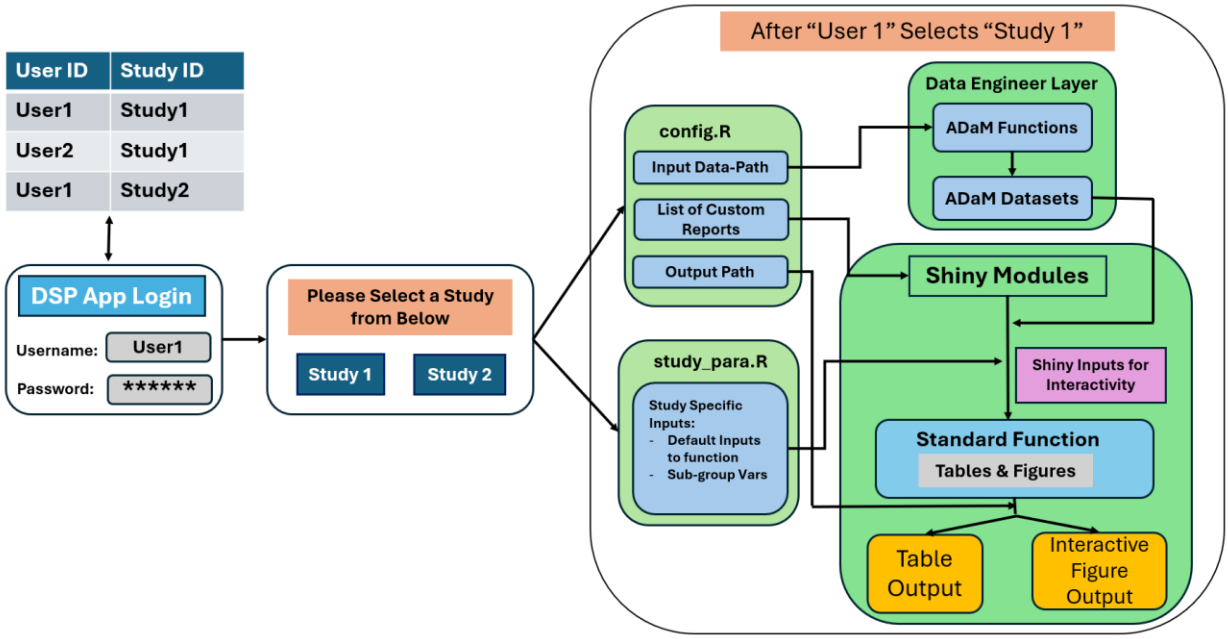


**Figure 2:** Illustration of how the various components, as discussed earlier, are integrated and connected within a single R Shiny application.

## FOLDER STRUCTURE

To support a standardized reporting tool while enabling study-specific programming flexibility, a unique folder structure is employed for the application. Within this structure, any changes to reporting functions and Shiny modules are applied to all studies, thereby isolating study-specific programming and preventing interaction between studies.

Reporting functions are stored in the 'logic' folder, and shiny modules are stored in the 'modules' folder. All study-specific programming is stored within their respective study folders. Each study can have a different set of ADaM functions, and

the Shiny application will identify which report to generate based on the available data. Figure 3 displays the folder structure designed to support a single R Shiny application across multiple studies, illustrating how the organization of files and directories enables seamless management of study-specific programming, configurations, and scripts. This structure ensures that the application can efficiently handle multiple studies simultaneously, allowing for easy customization and scalability while maintaining consistency in data processing and analysis workflows.

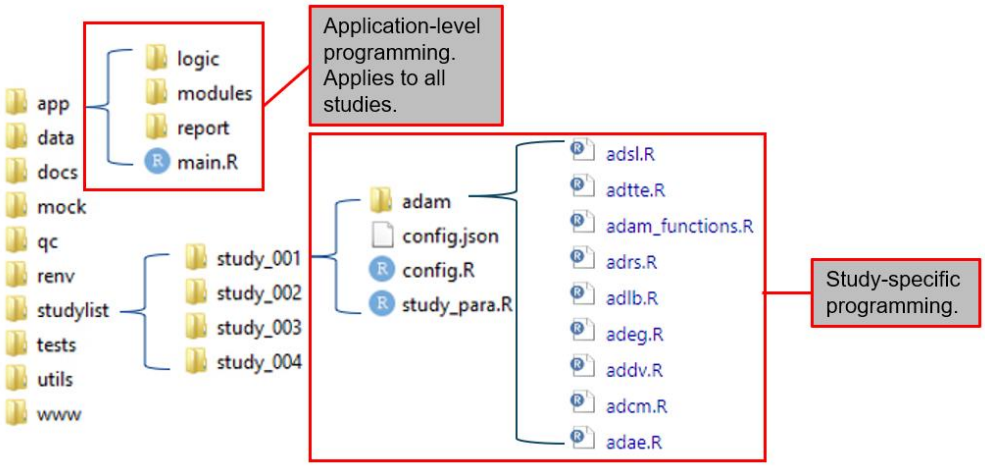


**Figure 3:** Folder structure supporting a single R Shiny application for multiple studies, demonstrating how the organization of files and directories facilitates efficient management, customization, and scalability of study-specific data and configurations.

## CONCLUSION

The Data Surveillance R Shiny Application serves as an innovative and interactive tool for efficient and flexible data monitoring in clinical research. By streamlining complex workflows, it simplifies integrated summary generation and standardizes reporting for data surveillance. Its modular framework not only enhances current study monitoring capabilities but also establishes a foundation for future scalability and adaptability. This approach demonstrates the potential of technology-driven solutions to improve operational efficiency, support critical decision-making, and advance the integrity of clinical research.

## ACKNOWLEDGMENTS

We would like to express our gratitude to the R Shiny development team and the DSP working group at Daiichi Sankyo, Inc. for their invaluable support. It is our honor to recognize the key contributors: Didier Murillo Florez (R Shiny Developer), Jake Klyn (R Shiny Developer), Loan Robinson (R Shiny Developer), Meng Qian (Biostatistician), and Ronan Fougeray (Biostatistician). We would also like to thank Vijay Koduru and Li-An Xu for their guidance and feedback throughout the development process.

## CONTACT INFORMATION

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

Author Name: Ruchitbhai Patel
Company: Daiichi Sankyo, Inc
Address: 211 Mt. Airy Road, Basking Ridge, NJ 07920-2311
Email: ruchitbhai.patel@daiichisankyo.com

Author Name: Jacky Jin
Company: Daiichi Sankyo, Inc
Address: 211 Mt. Airy Road, Basking Ridge, NJ 07920-2311
Email: jacky.jin@daiichisankyo.com

## DISCLAIMER

Please note the views and opinions expressed in this paper are those of the authors and are not intended to reflect the views and/or opinions of their employers.