



Enhancing Real-Time Clinical Reporting with R Shiny and Large Language Models: Automated Code Generation and Dynamic TLG Visualization

Mahesh Divakaran

Research Scholar

Amity University, Lucknow, India

Agenda



Background



Motivation



System Framework



Prototype



Demo



Conclusion

Clinical Reporting Background



TLGs are core clinical deliverables



Manual coding dominates workflows



High effort and slow iteration

Challenges in Current Workflow



**REPEATED
PROGRAMMING**



**DELAYED REVIEW
CYCLES**



**LIMITED
INTERACTIVITY**

Role of R in Clinical Analytics

Base R for data handling

- Load SDTM and ADaM datasets into session memory
- Inspect data structure, variable classes, and dimensions
- Subset rows and columns using index and logical conditions
- Handle missing values and data type conversions
- Control execution using functions, loops, and conditional logic
- Provide stable and predictable behavior for clinical data processing

tidyverse for data transformation

- Use dplyr for filtering populations and grouping by treatment arms
- Calculate summary statistics using grouped operations
- Use tidyr to reshape datasets for tables and figures
- Manage categorical variables using forcats
- Standardize labels and text fields using stringr
- Improve readability and consistency of transformation logic

R for visualization

- Map clinical variables to axes and plot aesthetics
- Create box plots, bar charts, and line plots for TLGs
- Overlay subject-level points for distribution inspection
- Control axis labels, titles, and legends explicitly
- Apply consistent themes across figures
- Support reproducible and review-ready clinical graphics

R Shiny Capabilities



Build

Build interactive dashboards for clinical data review



Support

Support real-time rendering of tables, listings, and graphs



React

React automatically to user inputs and prompt execution



Manage

Manage session-level data without repeated reloads



Separate

Separate user interface and server-side logic clearly



Display

Display results, generated code, and logs in dedicated tabs



Enable

Enable rapid iteration during internal clinical review meetings

Limitations of Shiny Alone



Requires manual development of server-side logic



Each new output needs additional coding effort



No native support for automated code generation



Maintenance increases as application complexity grows



Changes in requirements require redeployment



Limited support for prompt-driven workflows

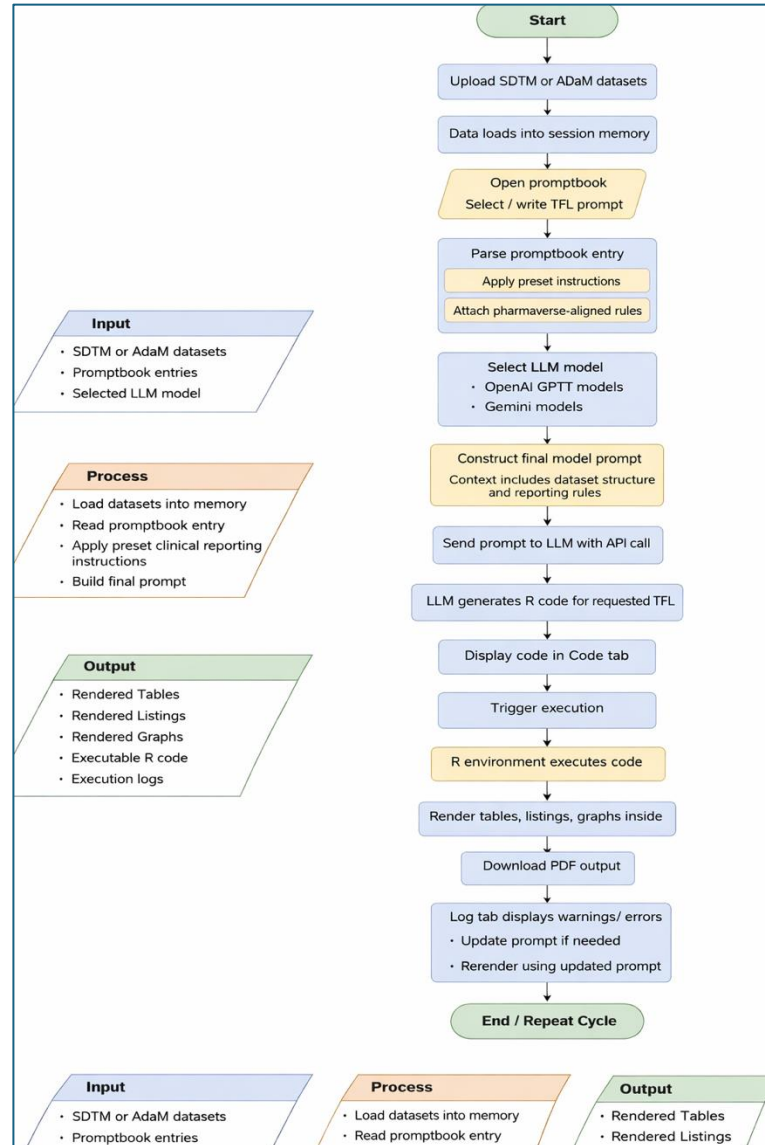


Productivity depends heavily on programmer expertise

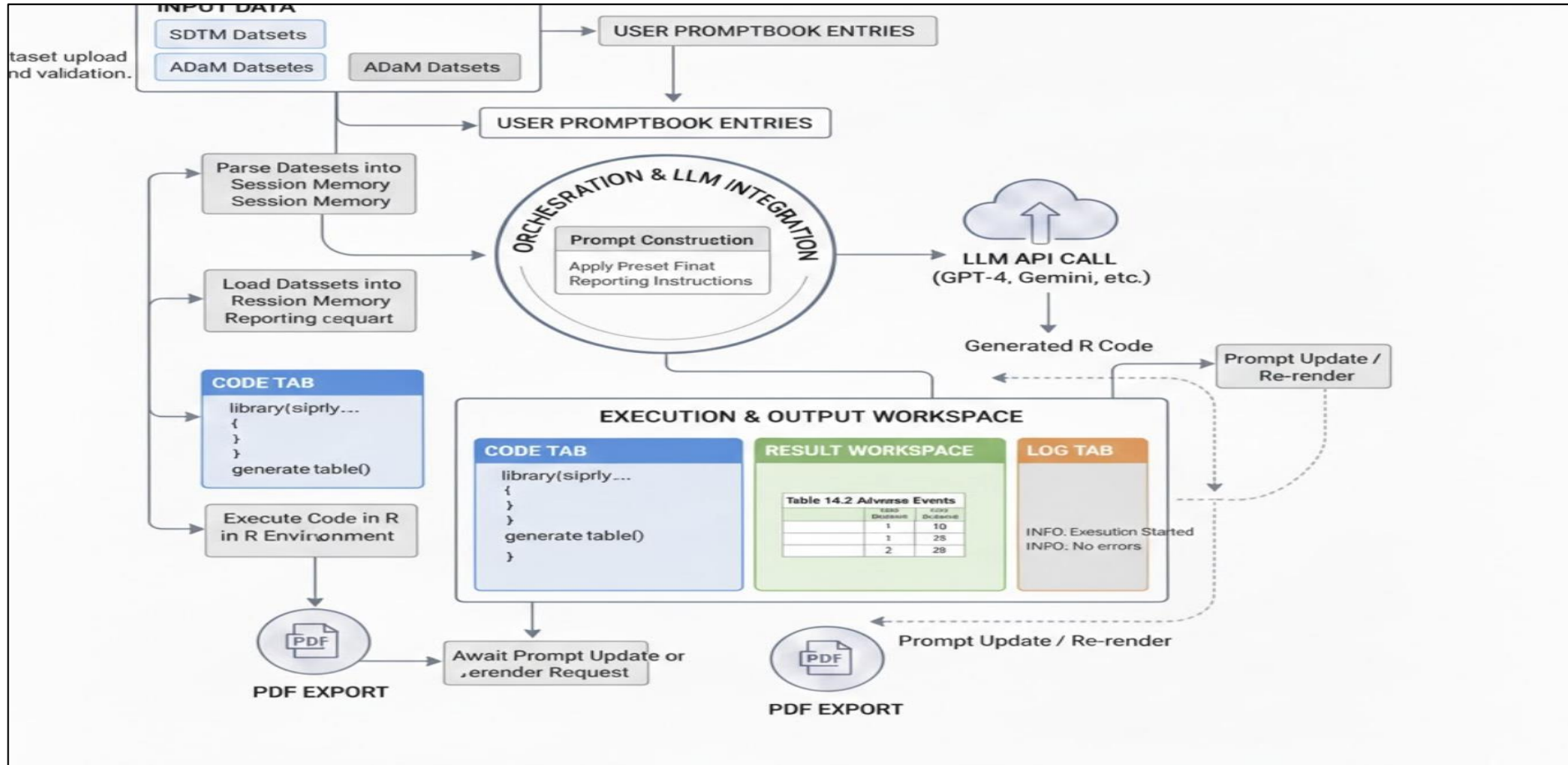
LLMs + R Shiny in Regulated Settings

Keep	Keep all clinical data inside controlled R Shiny sessions
Send	Send only prompts and metadata through secured API calls
Expose	Expose full generated R code for review and validation
Require	Require explicit user action before code execution
Capture	Capture logs, warnings, and execution status for audit review
Separate	Separate exploratory outputs from validated production deliverables
Support	Support compliance through transparency, traceability, and control

Proposed Framework



System Architecture



Promptbook Concept



Store

Store structured prompts for each Table, Listing, or Graph



Describe

Describe reporting intent instead of writing R code



Define

Define population rules, variables, summaries, and formatting



Reuse

Reuse prompts across studies and review cycles



Apply

Apply preset clinical reporting instructions automatically



Support

Support consistent output generation across sessions



Enable

Enable quick updates and rerendering during review

App Interface

PharmaTFL Agent

Browse...

Drag .xpt files here

Example: Create a demographic table summarizing Age, Sex, and Race by Treatment Arm (ARM).

Generate TFL

Result Workspace

Rendered Output

Generated R Code

Logs / Errors

Actions

Export to PDF

Downloads the currently rendered TFL.

Example Prompts

Create a demographic summary table using the rtables package based on the adsl dataset.

- 1. Column Layout:
 - Split the columns by Treatment Arm (ARM).
 - Include an 'Overall' column summarizing all patients.
 - Display the subject count (N) in the column headers.
- 2. Row Analysis:
 - Age (AGE): Summarize as a continuous variable. Show Mean, Standard Deviation (SD), Median, and Range (Min-Max).
 - Sex (SEX) and Race (RACE): Summarize as categorical variables showing counts and percentages.
- 3. Titles & Footnotes:
 - Title: 'Table 14.1.1: Demographic Characteristics'
 - Subtitle: 'Population: Intent-to-Treat (ITT)'
 - Footer: 'Source: ADSL dataset | Programmed by PharmaTFLAgent'
- 4. Output:
 - Assign the final table object to the variable final_output."

"Create a box plot figure using the adsl dataset to visualize the distribution of Age (AGE) by Treatment Arm (ARM).

- Requirements:
 - X-axis: Treatment Arm (ARM)
 - Y-axis: Age in years (AGE)
- Style: Use ggplot2. Overlay individual data points (jitter) on top of the box plots to show subject-level distribution.
- Formatting: Add a clear title 'Distribution of Age by Treatment Arm', label the axes appropriately, and use a minimal theme (theme_minimal).
- Output: Assign the final plot object to final_output."

Execution Environment

PharmaTFL Agent

Browse...

adsl.rda

Upload complete

adsl

"Create a box plot figure using the adsl dataset to visualize the distribution of Age (AGE) by Treatment Arm (ARM)."

Requirements:

Generate TFL

Success! TFL Generated.

Result Workspace

Actions

Export to PDF

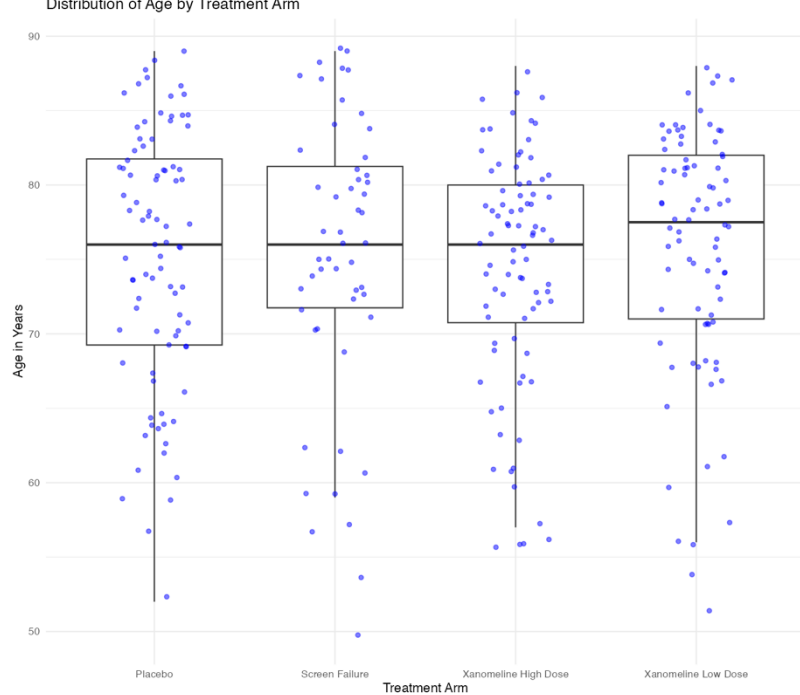
Downloads the currently rendered TFL.

Rendered Output

Generated R Code

Logs / Errors

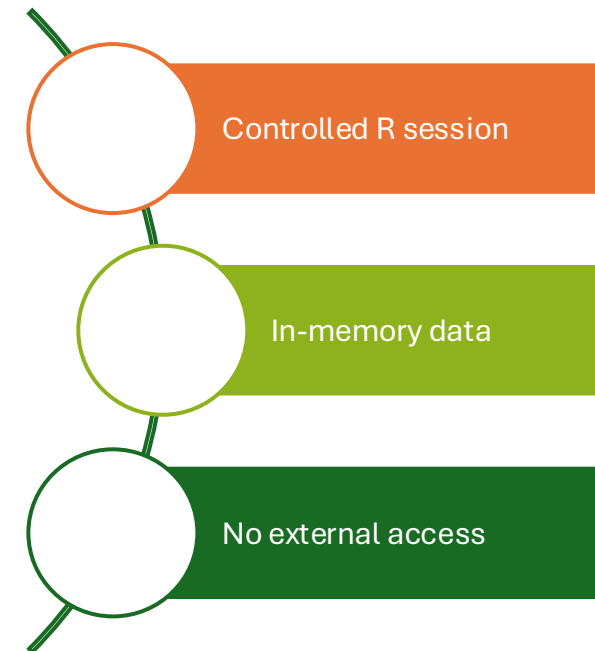
Distribution of Age by Treatment Arm



Age in Years

Placebo Screen Failure Xanomeline High Dose Xanomeline Low Dose

Treatment Arm



DEMO

CONCLUSION

Prompt-driven
automation
improves clinical
reporting efficiency

R Shiny provides an
interactive and
controlled execution
layer

Large Language
Models reduce
manual
programming effort

Full code visibility
supports traceability
and audit review

Rapid re-rendering
enables effective
collaborative review

Exploratory outputs
remain separate
from validated
deliverables

The PharmaTFL
Agent demonstrates
practical value in
real workflows

Questions?





Thank You.