phuse·connect
connect·share·advance

APAC 2026

19–21 February

Novotel Hyderabad International Convention Centre

FROM DOCUMENT TO DERIVATION: AUTOMATING ADAM VARIABLE AND ENDPOINT CODE GENERATION USING STUDY SPECIFICATIONS AND CLINICAL DOCUMENTS

Alisha D, Sruthy Biju
Pfizer, India

DISCLAIMER

I confirm that, the opinions and thoughts discussed in this presentation are subject to my own independent views and are not subject to the opinions of the organization that I represent.

AGENDA

1. INTRODUCTION
2. OBJECTIVES OF FRAMEWORK
3. OVERVIEW OF MODELS & ALGORITHMS
4. VARIABLE METADATA → R CODE GENERATION WORKFLOW
5. ENDPOINT REPOSITORY & RETRIVAL WORKFLOW
6. USER INTERFACE & DASHBOARD DEMONSTRATION
7. COMPARISON EVALUATION WITH OTHER MODELS
8. PERFORMANCE EVALUATION
9. FUTURE ENHANCEMENTS
10. CONCLUSION

INTRODUCTION

- Translating clinical dataset specifications into executable derivation code is typically a manual, interpretation-heavy task, often hindered by complex logic and evolving rules. To streamline this, we introduce an offline, automated framework that converts derivation descriptions into validated, reproducible R code, handling mappings, derived variables, and optimized logic.

- The system uses a locally deployed code-generation model with rule-aware preprocessing, ensuring traceability, repeatability, and alignment with CDISC standards. It also offers controlled endpoint support for rule retrieval and modification, serving as a supplementary capability.

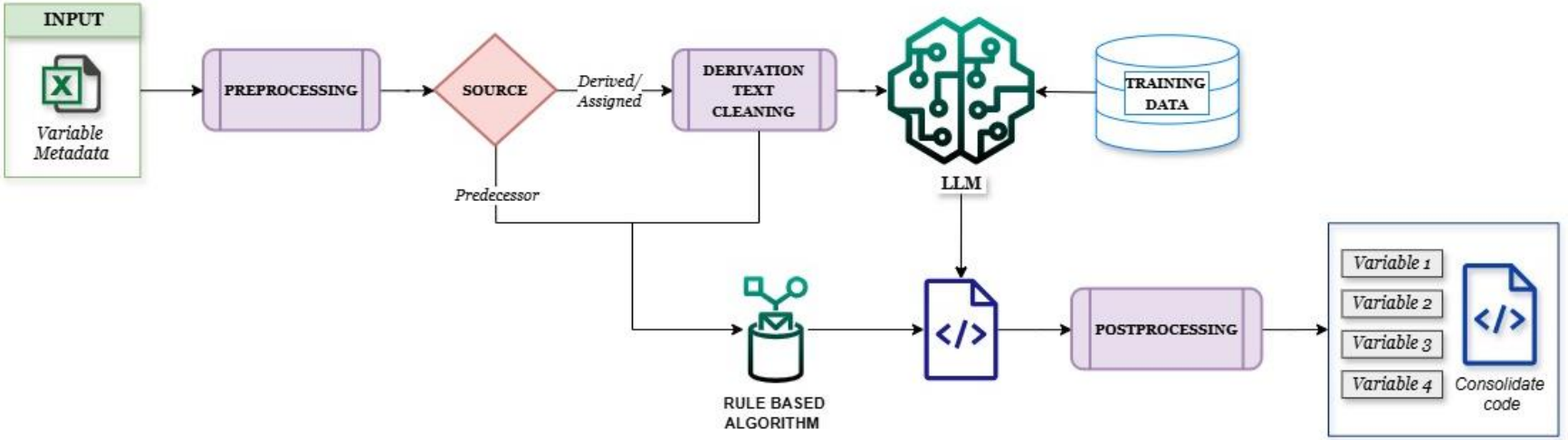
OBJECTIVES OF FRAMEWORK

- Automate the conversion of clinical dataset specifications into executable R code.
- Ensure reproducibility and traceability of endpoint derivations.
- Handle derivation logic, dataset dependencies, and optimized mutation logic.
- Ensure alignment with CDISC standards throughout the process.

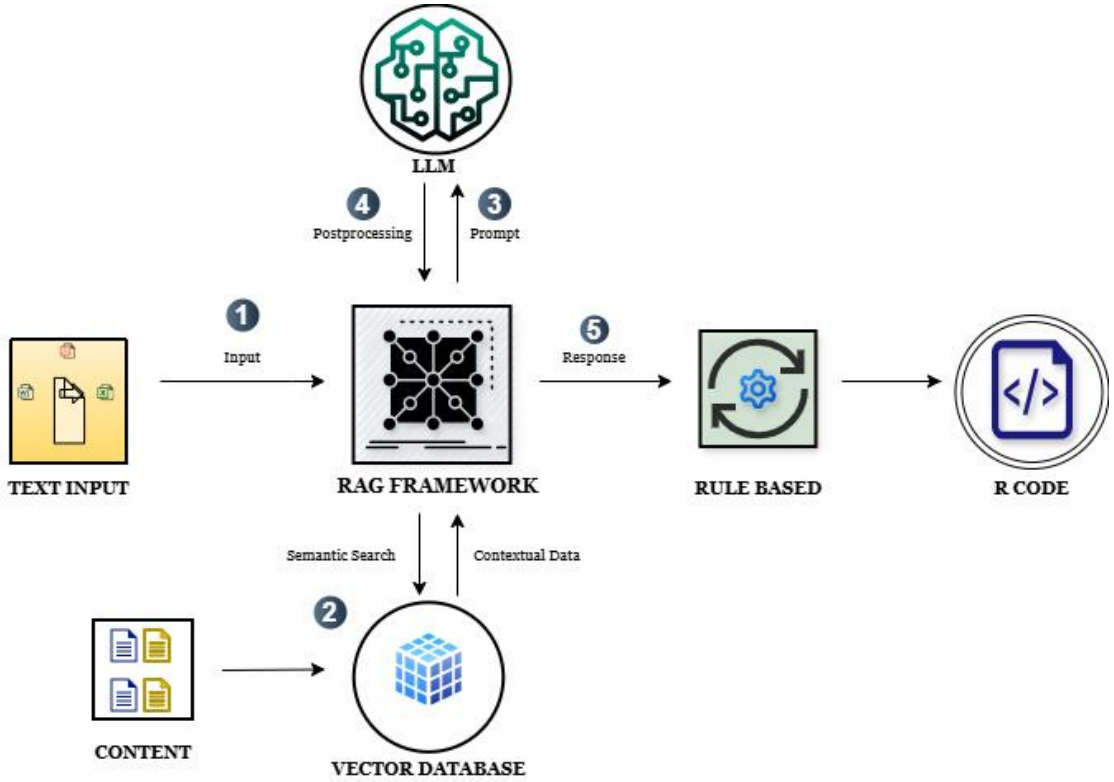
OVERVIEW OF MODELS & ALGORITHMS

COMPONENT	PURPOSE	KEY FEATURES
CodeGen-350M-Multi	Generate R code for derived variables	Multilingual training, fine-tuned on CDISC patterns
Sentence Transformer	Create embeddings for endpoint definitions	Dense semantic representation, cosine similarity
FAISS Index	Efficient retrieval of similar endpoints	Inner-product search, scalable indexing
Rule-Based Engine	Direct mapping of predecessor variables	Deterministic logic, reproducibility

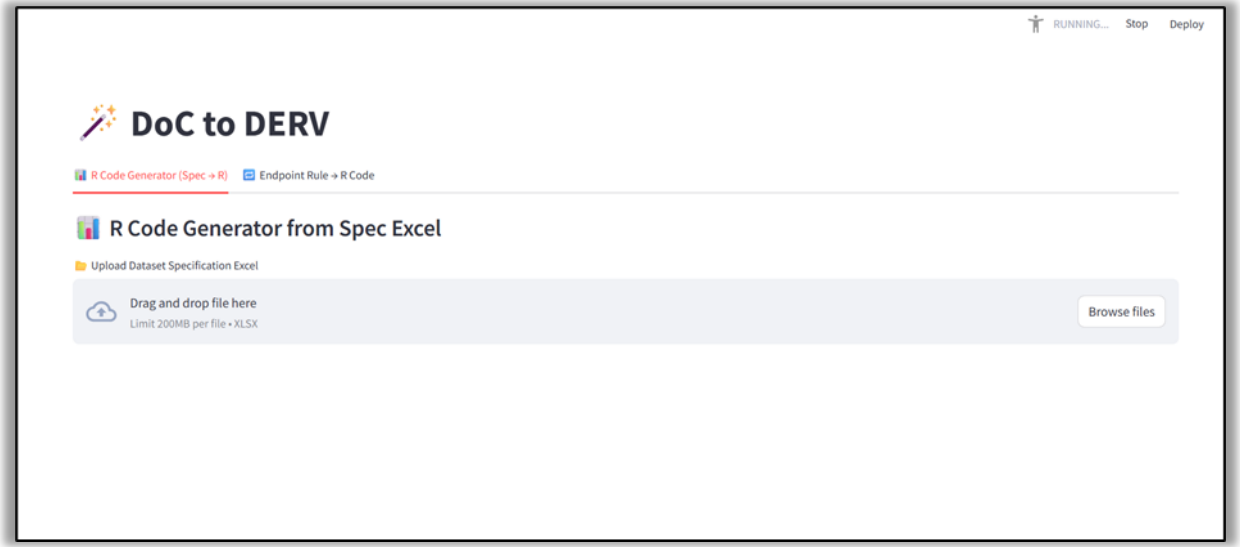
VARIABLE METADATA ⟶ R CODE GENERATION WORKFLOW



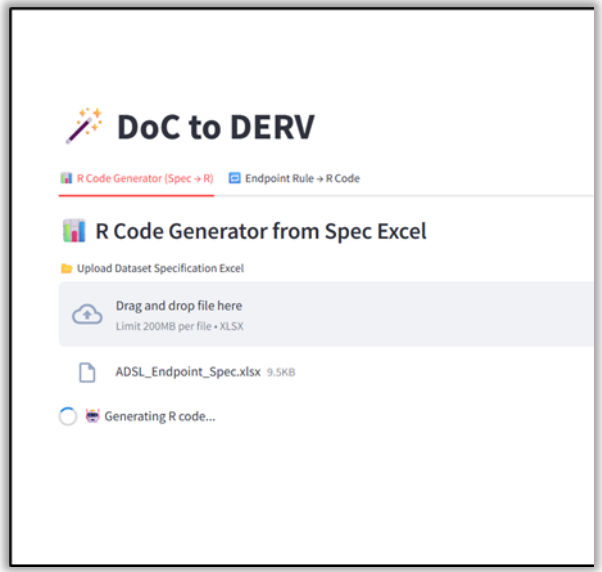
ENDPOINT REPOSITORY & RETRIVAL WORKFLOW



USER INTERFACE & DASHBOARD DEMONSTRATION



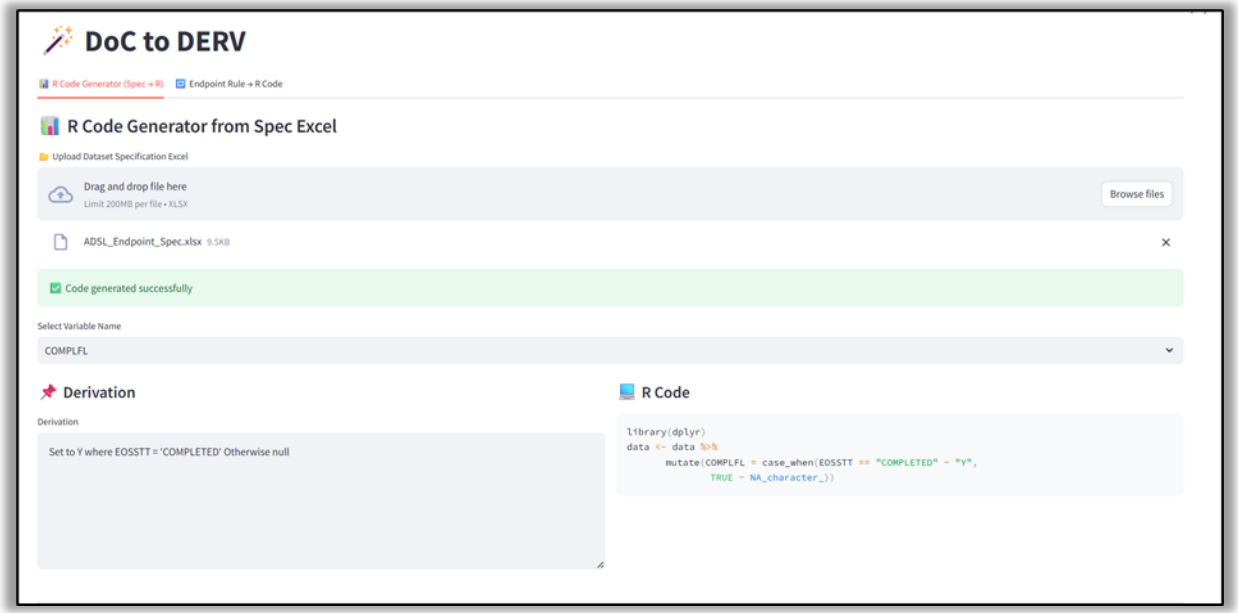
USER INTERFACE & DASHBOARD DEMONSTRATION



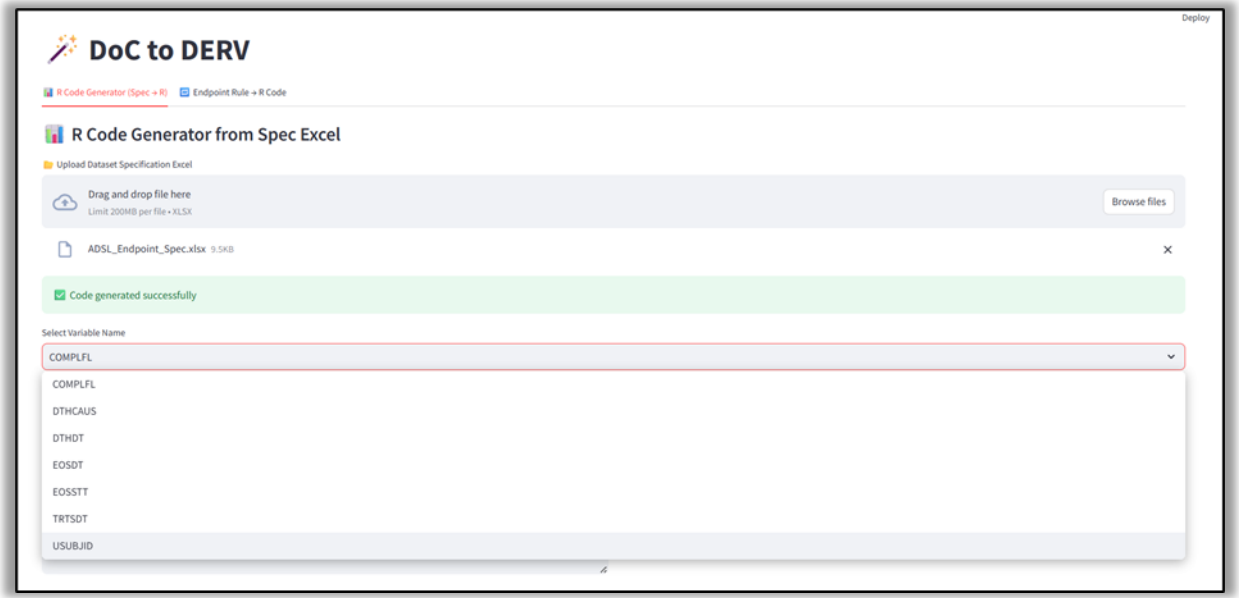
ADSL Variable Metadata

Variable Name	Variable Label	Type	Codelist/Controlled Terms	Source/Derivation/Comment
USUBJID	Unique Subject Identifier	text		Predecessor: DM.USUBJID
DTHDT	Date of Death	integer		Derived: DM.DTHDTC converted from ISO to numeric
DTHCAUS	Cause of Death	text		Derived: DD.DDORRES where DD.DDTESTCD is 'PRCDTH'.
EOSDT	End of Study Date	integer		Derived: DS.DSSTDTC where DS.DSCAT = 'DISPOSITION EVENT' then converted to numeric.
EOSSTT	End of Study Status	text	COMPLETED; DISCONTINUED	Derived: if DS.DSDECOD = 'COMPLETED' where DS.DSCAT = 'DISPOSITION EVENT' set to 'COMPLETED' if DS.DSDECOD = 'LOST TO FOLLOW-UP' where DS.DSCAT = 'DISPOSITION EVENT' set to 'DISCONTINUED' if DS.DSDECOD = 'DEATH' where DS.DSCAT = 'DISPOSITION EVENT' set to 'COMPLETED'
COMPLFL	Completed Population Flag	text	Y	Assigned: Set to Y where EOSSTT = 'COMPLETED' Otherwise null
TRTSDT	Date of First Exposure to Treatment	integer		Predecessor: DM.RFXSTDTC converted from ISO formatted text to numeric

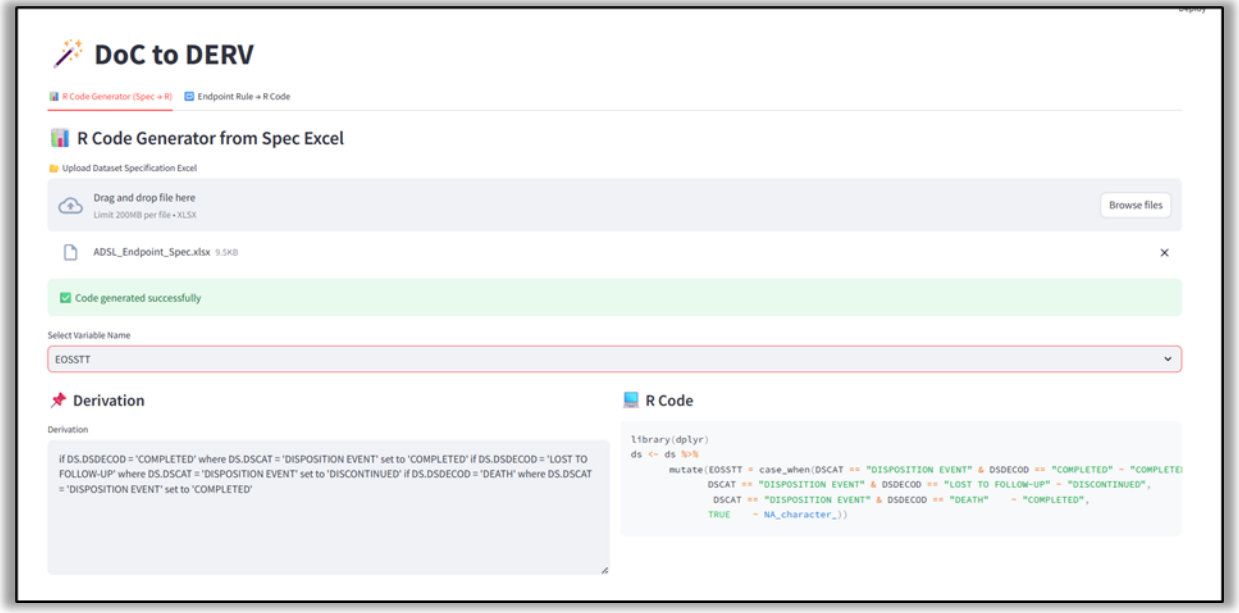
USER INTERFACE & DASHBOARD DEMONSTRATION



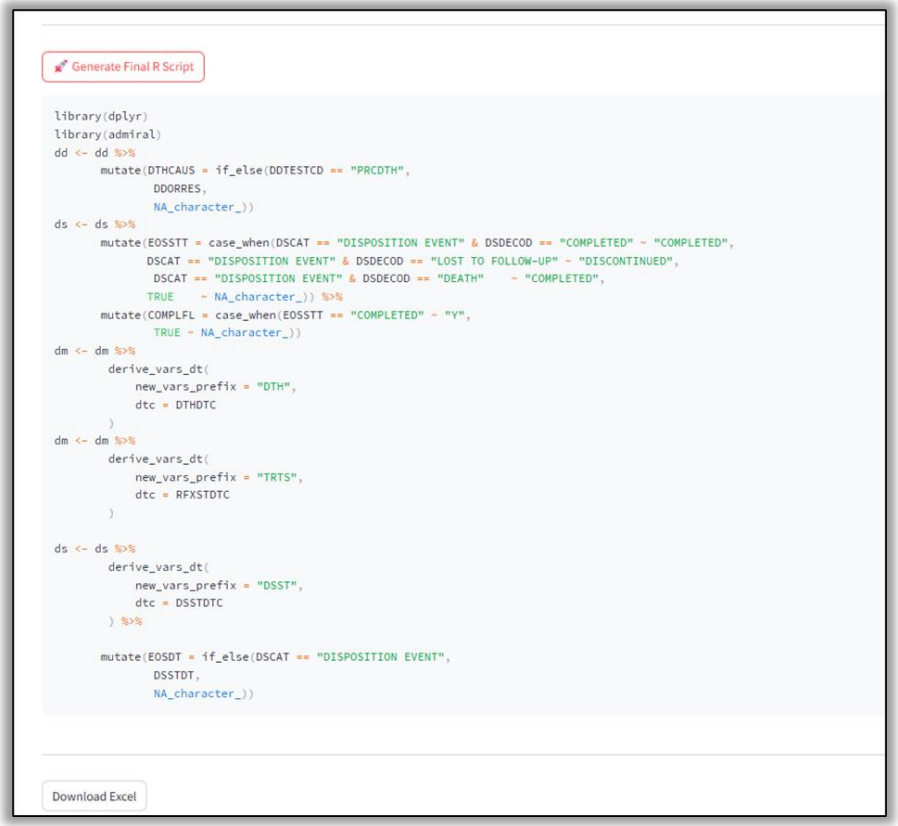
USER INTERFACE & DASHBOARD DEMONSTRATION



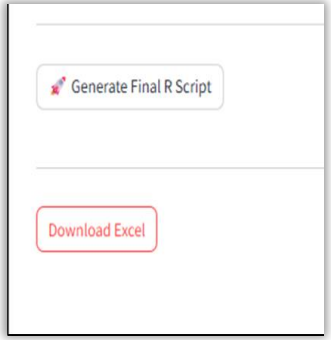
USER INTERFACE & DASHBOARD DEMONSTRATION



USER INTERFACE & DASHBOARD DEMONSTRATION

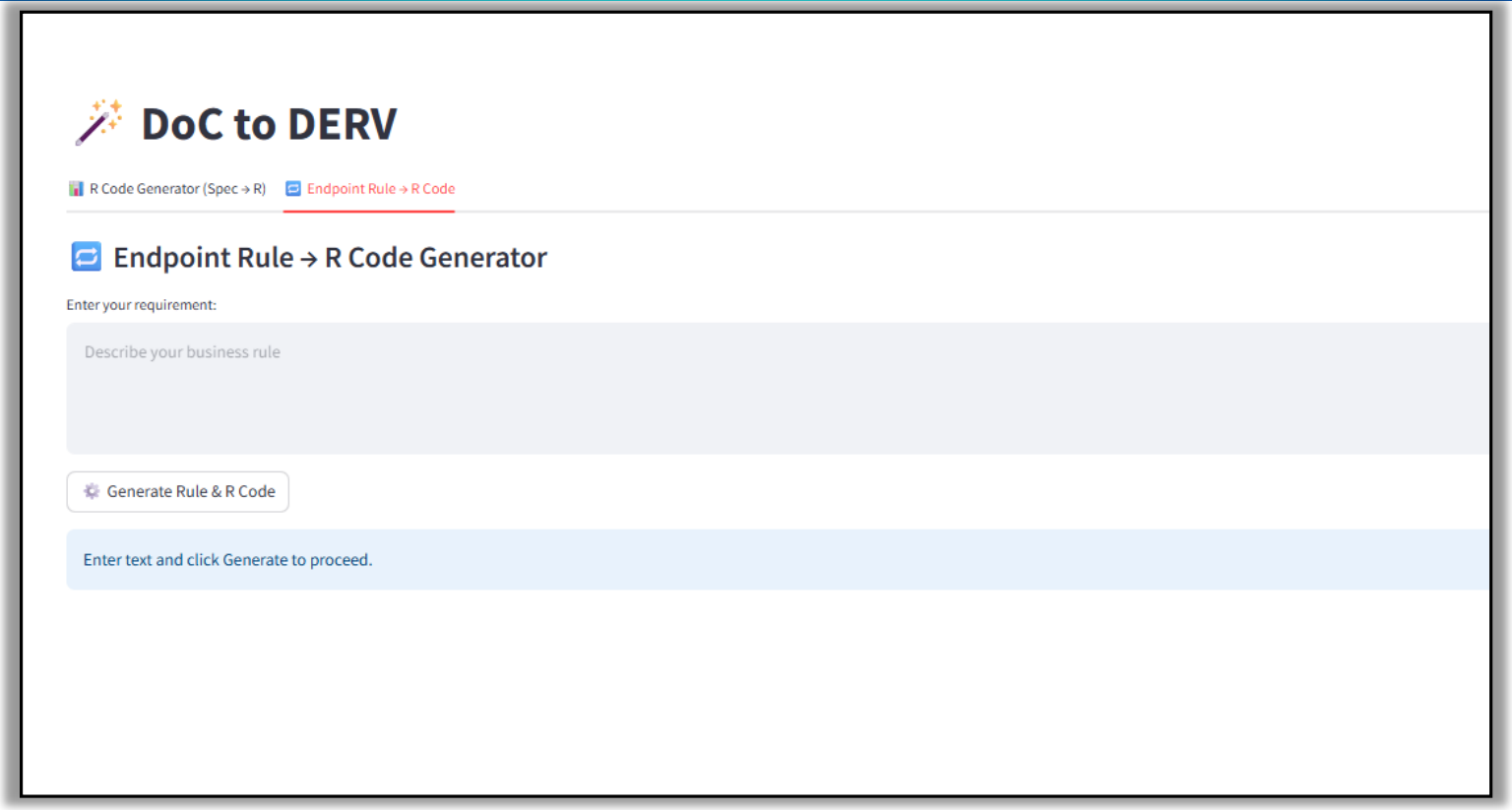


USER INTERFACE & DASHBOARD DEMONSTRATION

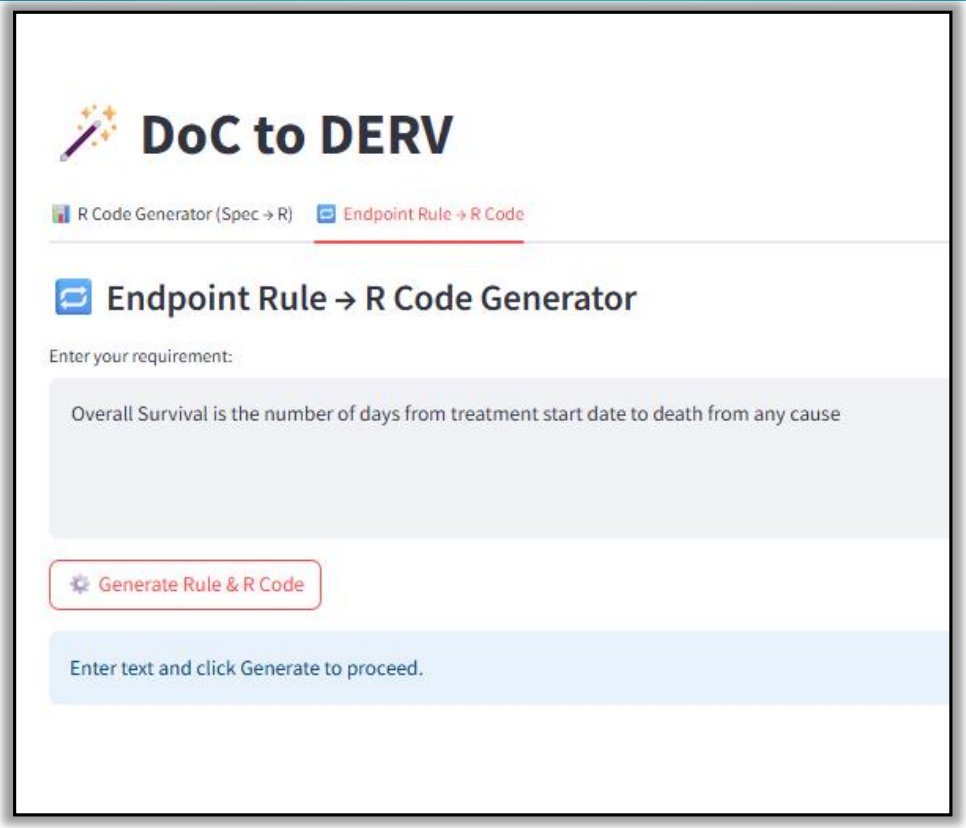


	A	B	C	D	E	F
1	**Variable Name**	**Variable Label**	**Type**	**Codelist/Controlled Terms**	**Source/Derivation/Comment**	**R_code**
2	USUBJID	Unique Subject Identifier	text		Predecessor: DM.USUBJID	
3	DTHDT	Date of Death	integer		Derived: DM.DTHDTC converted from ISO to numeric	dm <- dm %>% derive_vars_dt(new_vars_prefix = "DTH", dtc = DTHDTC)
4	DTHCAUS	Cause of Death	text		Derived: DD.DDORRES where DD.DDTESTCD is 'PRCDTH'.	dd <- dd %>% mutate(DTHCAUS = if_else(DDTESTCD == "PRCDTH", DDORRES, NA_character_))
5	EOSDT	End of Study Date	integer		Derived: DS.DSSTDTC where DS.DSCAT is 'DISPOSITION EVENT' then converted to numeric.	ds <- ds %>% derive_vars_dt(new_vars_prefix = "DSST", dtc = DSSTDTC) %>% derive_vars_dt(new_vars_prefix = "DSST", dtc = DSSTDT) %>% mutate(EOSDT = if_else(DSCAT == "DISPOSITION EVENT", DSSTDT, NA_character_))

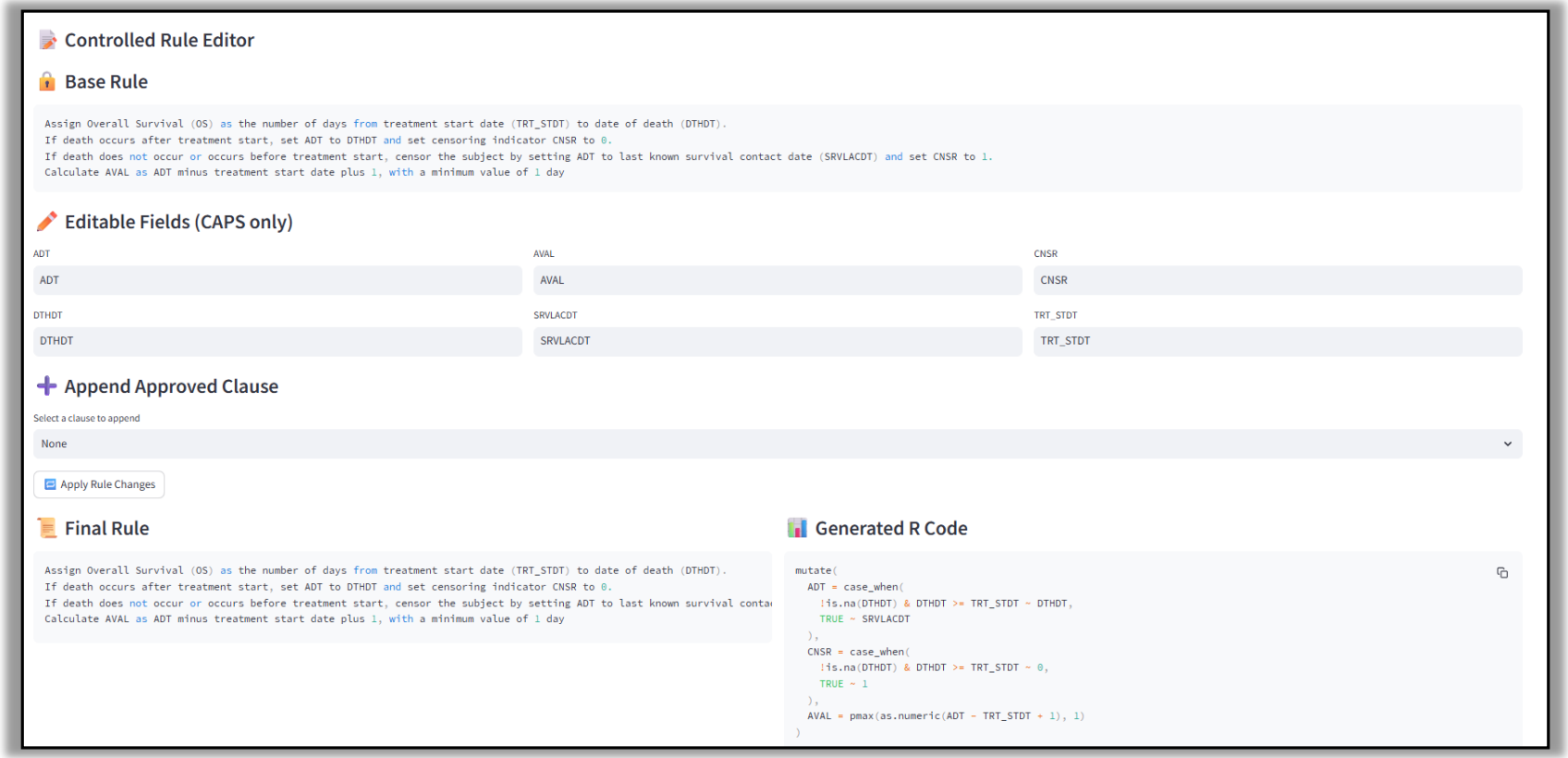
USER INTERFACE & DASHBOARD DEMONSTRATION



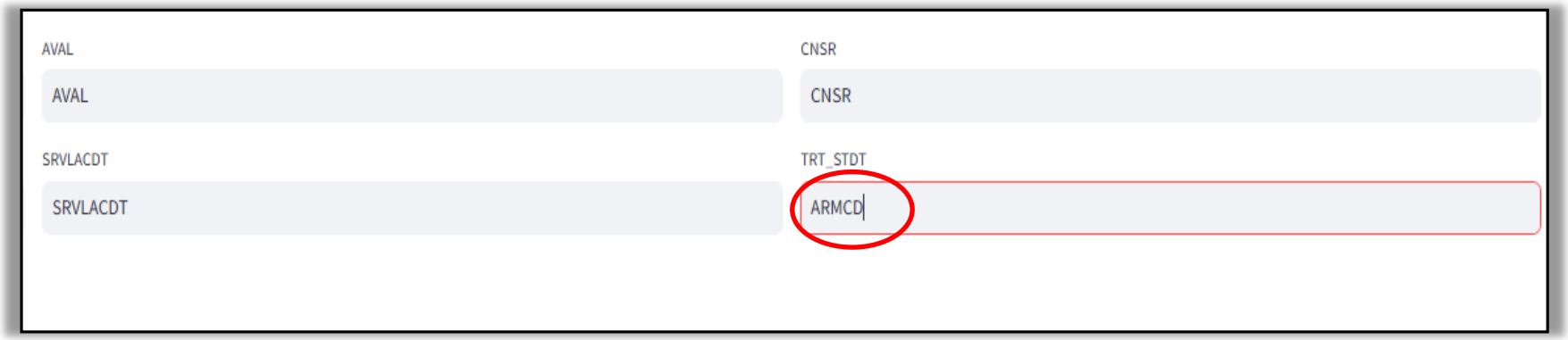
USER INTERFACE & DASHBOARD DEMONSTRATION



USER INTERFACE & DASHBOARD DEMONSTRATION



USER INTERFACE & DASHBOARD DEMONSTRATION



USER INTERFACE & DASHBOARD DEMONSTRATION

COMPARISON EVALUATION WITH OTHER MODELS

CodeGen-350M-Multi	Salesforce / CodeT5-Base	CodeLlama-7B
CODEGEN	salesforce	
✓ Balanced Performance	✓ Faster Generation	✗ High Quality Code
✓ Accurate & Efficient	✓ But Less Accurate	✗ But Slower Inference
✓ Low Resource Usage	✓ Lower Quality Code	✗ High Resource Usage

Key Takeaway: CodeGen-350M-Multi: Optimal Balance of Speed and Accuracy

Effective R Code Derivation with Reduced Computational Demands

PERFORMANCE EVALUATION

Code Generation Accuracy

70%

Fully Functional R Code Generated

- Aligned with Reference Logic
- Minor Errors Due to Limited Data

Endpoint Retrieval Efficiency

MRR: 0.80

Mean Reciprocal Rank

- High Accuracy in Top Results
- Effective Endpoint Retrieval

Reliable Performance in Code Generation and Endpoint Retrieval

Ensuring ADaM-Compliant Workflow

FUTURE ENHANCEMENTS

- Expand training to all BDS data structures
- Handle complex post-processing scenarios
- Enable advanced derivation logic
- Integrate validation, QC, and traceability
- Support adaptive learning for improvement

CONCLUSION: KEY TAKEAWAYS

1. Framework for R Code Generation

- Automates translation of clinical documents into executable R code, reducing manual effort.
- Ensures standardization and traceability aligned with **CDISC ADaM** principles.

2. Hybrid Automation Approach

- Combines deterministic rules and a lightweight code generation model.
- Balances automation with regulatory compliance and auditability.

3. Governance with Customization

- Separates rule governance from code generation to prevent logic drift.
- Allows study-specific customization within clear boundaries.

REFERENCES

- Jiang, J., Wang, F., et al. (2024). A Survey on Large Language Models for Code Generation. arXiv preprint.
- Nijkamp, E., Pang, B., Hayashi, H., et al. (2022). CodeGen: An Open Large Language Model for Code with Multi-Turn Program Synthesis. arXiv preprint.
- Tenneti, L. S. D., Gayatri, M. K., Prabha, M. D., & Shravani, D. (2025). Code Generation Using LLMs. Future Engineering Journal.
- E. Dehaerne, B. Dey, S. Halder, S. De Gendt and W. Meert, "Code Generation Using Machine Learning: A Systematic Review," in IEEE Access, vol. 10, pp. 82434-82455, 2022, doi: 10.1109/ACCESS.2022.3196347.
- Nijkamp, Erik, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. "A Conversational Paradigm for Program Synthesis." arXiv preprint (2022).
- CDISC. (2026, Jan 20). CDISC standards. Clinical Data Interchange Standards Consortium. https://www.cdisc.org/

THANK YOU