2026-05-01

pkgassistR: A Shiny/RAG framework for natural-language help for statistical R packages

Nikita Mozgunov
MRC Biostatistics Unit, University of Cambridge

pkgassistR will provide reusable R infrastructure for package maintainers to build Shiny-based, retrieval-augmented natural-language assistants over their package documentation, examples, and supporting materials. The framework will support local and API-based LLMs, answers with citations to retrieved package materials, visible retrieved evidence, low-evidence warnings, tests, and documentation. The first full implementation will use the `MAMS` package as the minimum milestone, with `survivalMPL` and `onlineFDR` used as additional motivating examples for developing a more general framework.

Executive Summary

In my work developing statistical R packages for biostatistics, I often see the same problem: the methods are useful, the software is available, but the first step for a non-statistician user is still too hard. Clinicians, trialists, and other applied researchers may have a clear idea of the clinical or scientific question they want to answer, but they may not know the exact package terminology, function names, arguments, examples, or statistical assumptions needed to use the software correctly.

The aim of **pkgassistR** is not to create a silver bullet that makes complex statistical packages usable in one second. Instead, the aim is more practical: to reduce the time between having an idea in mind and finding a correct first route towards implementation. In many cases the user and the package use different lexicons. A clinician may describe a problem in clinical language, while the package documentation describes the same problem using statistical and software terminology. This project focuses on helping users communicate with a package across that gap.

My aim is to develop **pkgassistR** as an open-source R framework for building package-specific natural-language assistants. Users should be able to ask questions in ordinary language, while the assistant is required to answer from the package's own documentation, help files, vignettes, examples, textbooks or teaching material where available, and selected source metadata. The framework

will combine a Shiny front end, retrieval-augmented generation (RAG), local and API-based LLM connectors, answers with citations to retrieved package materials, and inspectable evidence panels.

There is no magic in this process. The quality of the assistant will depend on the quality of the material available for the package: help pages, examples, vignettes, tutorials, websites, textbooks, and other supporting resources. Better material in should lead to better answers out. This is also why the framework will show citations and retrieved evidence, so users and maintainers can see what the answer is based on and where the documentation may need improvement.

This is important because many users do not read package help pages in full, and some users are not aware that vignettes exist. Even when they want to search, they may not know what terms to search for. At the same time, many users now naturally turn to AI tools for help in other domains. This is not necessarily a bad thing; it is the current reality of how people look for support. The R community can respond by providing AI-assisted interfaces that are grounded in the package materials we want users to rely on.

The first full implementation will be an interface for the `MAMS` package, which supports multi-arm multi-stage clinical trial designs and already has a working prototype. However, the proposal is not only about `MAMS`. I also plan to use `survivalMPL` and `onlineFDR` as further examples. The `survivalMPL` package extends the possibilities of survival modelling by using penalised maximum likelihood to estimate regression coefficients and smooth baseline hazards, including settings with left, right, and interval censored data. These models can be useful in many studies, but practical uptake can be limited because users may not know how to handle observation schemes such as interval censoring, left censoring, left truncation, or related data structures. My colleagues have also developed teaching material and textbook-style resources covering some of these technical topics, but moving from theory to practical computation, or from clinical practice back to the theoretical model, is not always straightforward.

The `onlineFDR` package is another motivating example. It combines several methods for online error control, where hypotheses arrive sequentially and decisions must be made without knowing the future tests. The package is powerful, but choosing the most appropriate method for a particular setting can be difficult for new users. A natural-language interface with citations to the relevant package materials could help users identify useful methods and documentation before they start coding.

The long-term goal is to create a universal package, similar in spirit to how `pkgdown` helps package maintainers build documentation websites. In this case, **pkgassistR** would help maintainers build Shiny applications and API connectors to LLMs over their own package documentation, examples, and supporting material. The expected outcome of this grant is a tested R package, reusable Shiny modules, provider-neutral LLM interfaces, documentation, and a full `MAMS` reference implementation. `survivalMPL` and `onlineFDR` will be used as additional motivating examples to check that the framework is not over-fitted to one package, with pilot interfaces developed if time allows.

Signatories

Project team

Dr Nikita Mozgunov, Statistical Programmer, MRC Biostatistics Unit, University of Cambridge. Project lead. Maintainer/developer of statistical R packages used in biostatistics and clinical-trials research. Responsible for project design, implementation, package architecture, reference implementations, evaluation, and community engagement.

Contributors

The main contributors will be co-authors and maintainers of the packages used as the initial test cases: `MAMS`, `survivalMPL`, and `onlineFDR`. These packages will provide the first practical background for developing and testing the framework. In summer 2026, we will also work with an undergraduate intern through the NIHR Undergraduate Internship Programme. We are currently reviewing candidates, so the intern cannot yet be named. The internship project will focus on scaffolding the system that will later be developed and generalised within `pkgassistR`. Colleagues in the MRC Biostatistics Unit with expertise in clinical trials, communication with clinical PIs, software development, and PPIE will be invited to provide feedback on whether the proposed interface is understandable and useful for applied users. Additional contributors will be invited from clinical trials, biostatistics, and statistical package maintainer communities.

Consulted

At this stage, the proposal has been developed from my own experience maintaining and developing statistical R packages for biostatistics, and from discussions within my research environment about package usability, clinical-trials methodology, and software implementation.

I have not yet discussed this proposal directly with R Consortium ISC members. I would welcome feedback from the ISC before or during review.

The Problem

The proposal

Overview

This proposal is to develop **pkgassistR**, a reusable R framework for building natural-language assistants for statistical R packages. A package maintainer should be able to point the framework at a package, build an index from its documentation and examples, connect a local or API-based model, and deploy a Shiny app that answers user questions with citations and visible retrieved context.

The project will use the existing `MAMS` assistant prototype as the first full demonstrator. `MAMS` is a useful reference case because it has specialised statistical terminology and real clinical-trials relevance. However, the funded work will generalise beyond `MAMS`. The final package should support other CRAN, GitHub, or local R packages with a documented workflow, and the design will be checked against `survivalMPL` and `onlineFDR` as additional package examples.

Detail

The main focus of the work will be to turn the current prototype into package-level infrastructure with stable data structures and tested functions. In practical terms, the framework will include:

- Corpus ingestion from R package sources, including Rd files, vignettes, README files, examples, and selected source-code metadata.
- Chunking and provenance tracking so each retrieved passage records source file, topic, function name, package version, chunk ID, and retrieval score.
- Index building using local embeddings initially, with an interface that can support alternative embedding providers.

- Retrieval logic that prioritises package documentation and examples over raw source code for user-facing explanations.
- Prompt construction that requires the model to answer from retrieved context, cite sources inline, provide uncertainty language when evidence is weak, and separate draft code examples from prose explanation.
- Provider-neutral LLM interfaces supporting local Ollama models and at least one API-based provider through explicit user configuration.
- Shiny modules for question input, answer display, citations, retrieved context, runtime status, and diagnostics.
- Tests for corpus extraction, retrieval ranking, prompt assembly, citation parsing, and low-evidence behaviour.
- Documentation showing package maintainers how to build and customise an assistant for their own package.

Minimum Viable Product

The minimum viable product will be a working R package and Shiny reference app that can:

1. Build a local corpus from a specified R package.
2. Create an index from package documentation and examples.
3. Accept a natural-language question in a Shiny interface.
4. Retrieve relevant package-specific evidence.
5. Generate an answer using a configured LLM.
6. Display inline citations and a separate evidence panel.
7. Warn users when the retrieved evidence is weak.
8. Demonstrate the full workflow using the `MAMS` package, and document how the same framework can be applied to `survivalMPL`, `onlineFDR`, or other packages.

Architecture

The proposed architecture has five layers.

First, a package ingestion layer will read package documentation, vignettes, README files, examples, and selected source metadata. It will normalise these into a corpus table with stable identifiers and provenance.

Second, an indexing and retrieval layer will create embeddings, store an R-native index suitable for package-sized corpora, and rank retrieved chunks using a combination of semantic similarity, lexical overlap, and documentation-first weighting.

Third, an LLM provider layer will handle chat and embedding calls. Local Ollama support will be included for privacy-sensitive and offline workflows. API-based support will be configurable and optional.

Fourth, a prompting and answer layer will assemble retrieved context, require citations, parse model output into answer/code/function sections, and apply low-evidence guards.

Fifth, a Shiny interface layer will provide reusable modules for package assistants, including status checks, question input, answer output, citations, retrieved context, and debugging views.

Assumptions

The assistant will be advisory and educational. It will not provide autonomous clinical decision support, replace statistical consultation, or guarantee correctness of generated code. Generated examples will be clearly presented as drafts that require review.

The assistant will not create knowledge that is absent from the package materials. Its usefulness for a particular package will depend on the quality and coverage of the documentation, examples, vignettes, tutorials, websites, and other materials provided to the retrieval layer. This is a "better in, better out" workflow, not a replacement for good documentation.

The first implementation will focus on package-sized corpora rather than very large document collections. This keeps the scope appropriate for an ISC grant and allows the system to run on a single machine. Future work may add support for external vector databases or larger institutional documentation collections.

Many package maintainers and users will prefer local-first workflows, especially where clinical or sensitive data may be involved. For that reason, local model support is a core feature rather than an optional extra.

External dependencies

The framework will build on existing R ecosystem tools where appropriate, including Shiny, bslib, testthat, httr2, jsonlite, dplyr, purrr, stringr, tibble, readr, fs, and related packages. Local LLM workflows will use Ollama through R client tooling (eg. ollamar). API-based LLM support will be optional and configured by users.

No grant funding is requested for cloud compute, hosted LLM usage, or hardware purchases.

Project plan

Start-up phase

During the first phase, I will review the existing MAMS prototype and refactor it into a package-oriented design. The main outputs will be a public repository structure, package skeleton, design notes, stable corpus/index object definitions, and an initial issue tracker with milestones.

This phase will also build on related scaffolding work planned for summer 2026 through the NIHR Undergraduate Internship Programme. The intern project will provide an early opportunity to test the basic workflow, identify implementation issues, and prepare components that can be generalised during the R Consortium-funded work.

I will seek early feedback from R package maintainers, biostatisticians, and potential clinical users. This will help ensure that the framework supports realistic package-maintainer workflows rather than only the original MAMS use case.

Technical delivery

The technical delivery will proceed through five milestones.

Milestone 1: Framework extraction and corpus API. Refactor the prototype into package-level functions, define corpus and index structures, and implement ingestion for Rd files, vignettes, README files, examples, and selected source metadata.

Milestone 2: Retrieval, citations, and guardrails. Implement retrieval scoring, documentation-first weighting, citation tracking, answer parsing, and low-evidence warnings. Add deterministic tests for retrieval and citation behaviour.

Milestone 3: LLM provider interface. Add provider-neutral chat and embedding interfaces. Support local Ollama models and at least one explicitly configured API-based provider.

Milestone 4: Shiny modules and MAMS reference assistant. Build reusable Shiny modules and rebuild the MAMS assistant using the framework. Include runtime status checks and retrieved-context inspection.

Milestone 5: Documentation, evaluation, and release. Write vignettes, maintainer guidance, example workflows, and an evaluation report based on clinician-style questions reviewed by domain experts.

Other aspects

The project will be developed openly. Issues and discussions will be used to gather feedback. Documentation will emphasise responsible use, including citations, uncertainty, privacy, and the need for expert review of statistical outputs.

This work focuses on a full biostatistical reference implementation for `MAMS`, but the same framework could support other specialised R domains where users need help navigating complex package documentation. The documentation will also discuss how maintainers can improve the quality of assistant outputs by improving the quality and structure of their package materials.

The project outputs can also be promoted through the MRC Biostatistics Unit communication channels. The Unit is supportive of promoting software and methodological work developed by its staff, including through social media and website news items. I also plan to write a blog post, or a short series of posts, covering different stages of the project development, including the motivation, the first `MAMS` implementation, lessons from applying the framework to other packages, and practical guidance for package maintainers.

Budget & funding plan

Funding requested: **US $12,000**

The budget will primarily support developer time for package extraction, testing, Shiny module development, documentation, and evaluation. The proposed allocation is:

- Software design and implementation: **US $5,000**
- Retrieval, citation, and LLM provider interfaces: **US $2,500**
- Shiny modules and MAMS reference implementation: **US $1,500**
- Testing, documentation, and vignettes: **US $2,000**
- Evaluation, community feedback, and release preparation: **US $1,000**

No funds are requested for hardware or cloud software purchases. Any API-based LLM examples will be optional and configured by users with their own credentials. The primary reference workflow will support local models.

Success

Definition of done

The project will be considered complete when:

- A public R package repository is available with documented functions for corpus ingestion, indexing, retrieval, prompting, and answer parsing.
- Reusable Shiny modules are available for building package-specific assistants.

- The framework supports local Ollama workflows and at least one optional API-based provider.
- The `MAMS` reference assistant is rebuilt using the reusable framework.
- Tests cover corpus extraction, retrieval, prompt construction, citation extraction, and low-evidence behaviour.
- Documentation includes a vignette showing how a maintainer can build an assistant for another R package.
- Documentation explains that assistant quality depends on the quality of package help pages, examples, vignettes, tutorials, websites, and other source materials.
- An evaluation report summarises performance on a set of clinician-style questions and records known limitations.

Measuring success

Success will be measured by both technical and usability criteria.

Technical measures include passing tests, reproducible corpus/index building, stable citation provenance, successful local model execution, and clear failure modes when models or indexes are unavailable.

Usability measures include whether applied users can ask questions in their own language, whether answers identify relevant package functions and documentation, whether citations help users inspect evidence, and whether maintainers can adapt the framework to another package using the documented workflow.

Documentation-quality measures will include whether the assistant exposes gaps in the available material, for example questions where retrieval is weak because the package lacks examples, vignettes, or terminology that matches applied users' questions.

Community measures include feedback from package maintainers and applied users, issue tracker activity, and interest from related R communities such as R/Medicine and biostatistics package developers.

Future work

Future work could include support for larger documentation collections, hosted deployment patterns, additional LLM providers, integration with pkgdown sites, support for Bioconductor packages, richer evaluation tooling, and templates for package-specific teaching assistants. Longer term, this could become a general pattern for natural-language help based on package documentation and examples.