uc3m | Universidad Carlos III de Madrid

Master Degree in Computational Social Sciences
2024-2025

Master Thesis

From Data to Report: automatizing personalized analysis in data analytics

Andrea Rivera Mateos

Iñaki Ucar Marques

Madrid, 2025

SUMMARY

Every hour spent formatting slides is an hour not spent thinking. In data analytics, reporting remains one of the most repetitive, error-prone, and under-optimized parts of the workflow—especially in teams where technical expertise varies and deadlines are tight. This thesis addresses that gap by developing a modular, user-centered reporting application using the `R Shiny` framework. The application automates the generation of analytical reports—from data upload to structured `PowerPoint` export—without requiring any programming skills.

Built through an iterative, prototype-based methodology, the application integrates key principles of human-centered design, reproducible computing, and reactive programming. It supports essential reporting tasks within a single, interactive environment. In addition, emphasis was placed on usability, modularity, and adaptability, ensuring the tool remains accessible to non-programmers while flexible enough to support diverse analytical contexts.

Beyond its technical contribution, the project engages broader debates in open science, including the balance between usability and rigor, the risks of human error in collaborative environments, and the methodological demands of reproducible analysis. By transforming reporting into a more automated, interpretable, and communicable process, the application offers a practical, scalable solution for improving analytic workflows across academic, public, and private settings.

Keywords: *Reproducible Analytics, Automated Reporting, R Shiny, User-Centered Design, Analytic Communication*

CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The generation of data-driven reports is a fundamental yet often tedious task. Despite increasing computational capacity and the widespread availability of statistical software, many analytic environments still rely on manual workflows for functions such as data cleaning, analysis, and reporting. These manual processes are not only time-consuming but also prone to inconsistencies, reducing the efficiency and reproducibility of research outputs. Therefore, the convergence of computer-assisted qualitative data analysis with more structured analytical tools presents an opportunity to streamline workflows and reduce redundancy in analytical practice.

In the data analytics business, the repetitive nature of tasks such as exporting tables, creating plots, and integrating these into standardized slide decks introduces a significant administrative burden. Furthermore, many of the individuals involved in such work—research assistants, project managers, field coordinators, or policy analysts- may lack advanced programming skills. This creates a gap between data and insight, where valuable time is spent formatting results rather than interpreting them. Similar challenges are also prevalent in the private sector, where teams frequently encounter bottlenecks when translating raw data into actionable reports.

Therefore, this thesis addresses this gap in the market by developing an interactive, user-centered application using the `R Shiny` framework to automate the production of analytical reports. The idea for this project emerged not as an abstract technical exercise, but directly from the challenges I encountered in my own work. Over the past three years on Research team, I often found myself producing reports that followed nearly identical structures, differing only in the data they contained. Despite having access to powerful statistical software, the process of transforming analysis outputs into polished, stakeholder-ready reports remained highly manual—time-consuming, repetitive, and prone to error. I began to wonder: why, in an increasingly data-rich environment, was so much of this work still being done by hand?

This question became the foundation of the project. The goal was to create a solution that could relieve the burden of repetitive reporting, freeing analytics to focus on interpretation and insight. At the same time, the tool needed to be intuitive and flexible—something that non-technical users could adopt with ease, yet robust enough to handle the complexity of real-world data. That complexity posed one of the main challenges. Data is rarely straightforward; it is often multi-layered, longitudinal, and highly contextual. The application had to account for this dimensionality while producing outputs that were standardized enough to be reusable across projects.

Designing for both flexibility and structure required a modular approach, where users could upload different datasets but still generate reports that adhered to a consistent visual and analytical format. However, striking the right balance was not easy. Too much rigidity, and the tool would be unusable for anything but the most templated work; too much flexibility, and it would lose the efficiency gains that automation is meant to provide. Much of the development process was therefore spent navigating this tension—testing, iterating, and revising based on both technical constraints and practical needs.

Although the application was initially built with my own team's workflow in mind, scalability was an essential part of the design from the beginning. The core logic was abstracted in a way that allows it to be adapted by other units facing similar challenges—whether in academic institutions or private sector analytics teams. The idea was never to solve a single reporting problem, but to create a framework for solving many of them, across different settings and datasets.

In this way, the app reflects more than just a response to an isolated inefficiency. It represents a broader reflection on how we work as data scienctists—how much of our time is consumed by formatting rather than thinking, by re-creating rather than refining. This tool is, at its core, a practical answer to a recurring professional frustration. But more than that, it is an attempt to reclaim space for analysis, creativity, and clarity in the everyday practice of data analytics.

This thesis unfolds in five main chapters. Following this introduction, the *Literature Review* situates the project within ongoing academic conversations around reproducibility, usability, and open science, highlighting the methodological and practical tensions it seeks to address. The *Objectives* chapter then defines the specific aims guiding the development of the tool, framed both technically and conceptually. The *Methodology* chapter outlines the design logic, development choices, and user-centered principles that shape the application, while the *Application Overview* offers a detailed walkthrough of its technical architecture and interactive features. Finally, the *Challenges* chapter reflects on the practical difficulties encountered—ranging from interface design to state management—and evaluates how successfully the tool met its intended goals.

Together, these sections trace the project from its initial motivation through to implementation and reflection, offering both a functional prototype and a contribution to broader discussions on improving workflows.

LITERATURE REVIEW

In recent years, the demand for reproducibility has reshaped expectations around research quality and transparency across disciplines. In the social sciences in particular—where research often involves complex, context-dependent data and nuanced analytical decisions—this reinforcement has arrived later than in other fields and has exposed long-standing tensions between methodological rigor and everyday practice. Without reproducible workflows, analytical processes become difficult to verify, extend, or replicate, leaving findings vulnerable to both error and misinterpretation (Goodman et al., 2016; Peng, 2011).

Despite increasing awareness, many research environments still rely on fragmented workflows, often spread across `Excel`, `SPSS`, and presentation software. These tools may be familiar and widely available, but they rarely promote transparency or consistency. Decisions are made informally, transformations are undocumented, and results are often copied manually into final reports—leaving behind little trace of how conclusions were actually reached (Rule et al., 2019; Sandve et al., 2013). As teams grow, change, or collaborate across institutions, the risks of human error and miscommunication compound. In my own experience, this disconnect between analysis and reporting is one of the most persistent challenges facing applied research teams.

Addressing this gap is not simply a technical challenge—it is a methodological one. Reproducibility depends not only on code and data, but also on how tools are used, shared, and understood across team members. In multidisciplinary projects, for instance, researchers may bring different assumptions about versioning, documentation, or software. Some may rely on code; others may prefer point-and-click interfaces. These differences are rarely acknowledged in tool design, yet they shape whether workflows are adopted, understood, and maintained in practice (Beaton et al., 2000; Wilson et al., 2014).

In response, the open science movement has advanced a range of solutions aimed at making data workflows more transparent and accountable. Tools like `R Markdown`, `Jupyter Notebooks`, and platforms like *OSF* and *GitHub* offer powerful frameworks for literate programming, documentation, and sharing (Stodden et al., 2014; Xie, 2015). Yet many of these tools assume a certain level of technical expertise—creating a divide between those who can access these benefits and those who cannot. In effect, some of the very tools designed to democratize research risk excluding practitioners with limited programming experience (Fecher & Friesike, 2014).

This usability gap is particularly evident in the debate between open-source and proprietary software. On one hand, open tools like `R` or `Python` offer flexibility, transparency, and community-driven development. On the other hand, proprietary platforms like `SPSS` or `Tableau` often provide more intuitive user experiences but obscure the underlying logic of analysis. This trade-off is especially relevant in collaborative environments, where interpretability and traceability are critical not only for validation but also for communication and handover. Good reproducible practices in these contexts involve more than code-sharing—they include modular project structures, version control, and the use of trusted data repositories such as *GESIS*, *Harvard Dataverse*, *OpenMX*, or *arXiv*. These practices facilitate reuse, minimize human error, and reinforce transparency across the research lifecycle. Reproducibility, in this sense, is best understood not as a binary outcome, but as a spectrum—ranging from opaque results with no supporting metadata to dynamic, fully documented workflows that integrate code, narrative, and data in a coherent structure.

It is precisely at this intersection—between openness and usability, between individual expertise and team collaboration—that tools like `R Shiny` become relevant. `Shiny` enables the development of interactive, web-based applications built entirely within the `R` environment. It allows researchers to design tools that encapsulate complex analytical workflows but present them through accessible interfaces. Recent studies show its effectiveness in fields like public health, education, and government analytics, where timely decisions depend on both analytical robustness and communicative clarity (Chang et al., 2021; Kassambara, 2018).

Yet the potential of `Shiny` to support end-to-end analytical workflows—especially those that include structured, presentation-ready outputs—remains underexplored. In my own work, I found that while existing `Shiny` applications often focus on exploration or visualization, few address the equally important task of turning those insights into standardized reports. This is a crucial gap. In applied settings, the final output is often not a publication or dataset, but a presentation, dashboard, or slide deck. The reporting phase is where research becomes communicable—where analysis meets policy, practice, or funding decisions.

This literature not only highlights a set of persistent limitations in current research workflows but also points toward a growing consensus: that achieving reproducibility, clarity, and usability must become integrated goals, not competing ones. By situating this thesis within that ongoing conversation, the project aims to offer a concrete, practice-oriented contribution—one that explores how accessible, modular, and automated tools can support transparent reporting without placing undue technical burdens on users. In doing so, it responds to both the methodological and practical demands of applied social research, where the stakes of miscommunication are high, and the value of well-designed workflows is increasingly clear.

OBJECTIVES

Given the repetitive, time-consuming, and error-prone nature of manual reporting workflows in applied research, this project aims to develop a practical and scalable solution that supports automation, enhances reproducibility, and improves accessibility for users with varying levels of technical expertise. The application addresses both, reporting and respond, not only to streamline the reporting process but also to respond to broader concerns around consistency, transparency, and the effective communication of research findings.

In this context, the following general and specific objectives guide the development and implementation of the tool:

General Objective

To critical respond to recurring inefficiencies in data analytics reporting by developing a transparent, replicable, and user-oriented application that bridges the gap between data analysis and communication, while promoting automation and reproducibility in the reporting workflow.

Specific Objectives

1. **Streamline Reporting Workflows** identifying inefficiencies in manual reporting processes and transforming them into user-oriented application features that reduce time and effort.

2. **Automate Analysis and Output Generation** to enhance consistency and reduce manual workload.

3. **Improve Data Accessibility and Exploration** enabling interactive visualizations and dynamic exploration of results, helping users gain deeper insights regardless of their technical background.

4. **Promote Usability, Reproducibility, and Flexibility** designing an interface that is accessible to non-programmers, supports complex datasets, and embeds consistent procedures to ensure reproducibility and transparency.

5. **Ensure Scalability and Long-Term Use** building a modular, reusable architecture that supports future adaptation across datasets, organizations, and reporting needs—facilitating institutional knowledge-sharing and collaboration.

Beyond fulfilling its functional goals, the application aims to reshape reporting as a more communicative, collaborative, and sustainable process in data analytics.

It promotes the **homogenization** of standardized reporting practices, helping teams maintain consistency in logic, format, and structure across time and users. This reduces variation and embeds shared practices into daily workflows.

The application also strengthens **organizational learning and capacity-building** by replacing fragmented processes with reusable logic—making on-boarding easier and ensuring continuity even as teams change.

In addition, it **enhance clearer research communication** by producing outputs that are both analytically sound and visually coherent—especially valuable in interdisciplinary or policy settings where clarity and efficiency are critical.

Finally, the application reflects a broader push in computational social science toward **human-centered design**. By prioritizing usability and accessibility, it bridges the gap between technical infrastructure and everyday research needs.

In this way, the application not only addresses recurring challenges in data analytics workflows, but also models how accessible, modular tools can support better reporting practices at scale. It serves as a foundation for continued development, inviting further iteration, integration, and institutional adoption.

METHODOLOGY

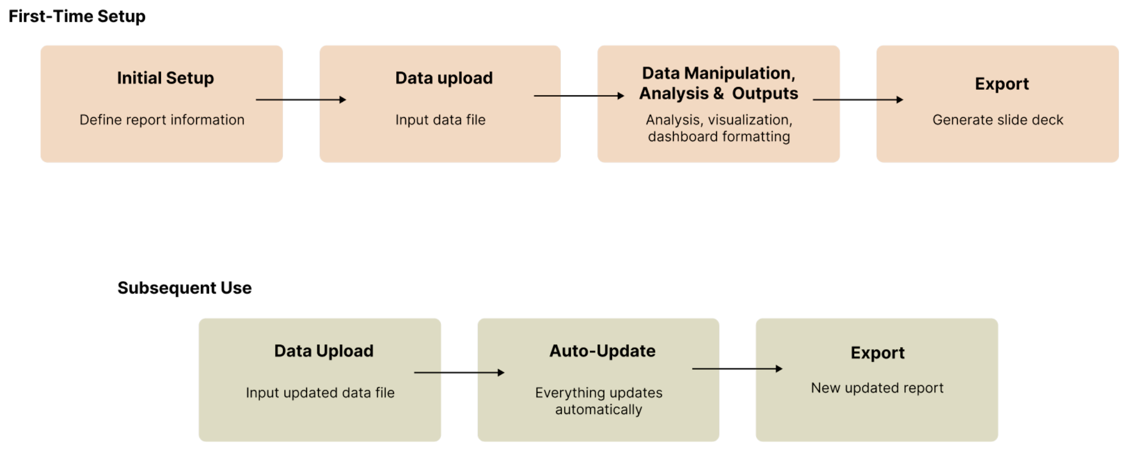
To operationalize the objectives outlined above, the development of this application follows a structured and iterative methodology approach outlined to balance conceptual clarity, technical feasibility, and user-centered design. The approach integrates principles from user experience design, data pipeline automation, and reproducible computing to build a functional prototype using the `R Shiny` framework.

Development Framework and Methodological Rationale

The development of this application is focous on the need to address recurring inefficiencies in how data analytics teams produce analytical reports. Given the project's limited timeframe, it adopts a **prototype-based methodology** rather than pursuing a full, production-ready build. This approach is both strategic and pragmatic, enabling the creation of a functional **beta version** that demonstrates core features, design logic, and potential for future scalability.

Instead of implementing an exhaustive suite of analytical tools, the prototype emphasizes **breadth over depth**—offering key functionalities such as data import, table and plot generation, and dashboard composition. The goal is to deliver a working **proof of concept** that meets typical reporting needs and lays the groundwork for future development or institutional integration.

Figure 1
First-time setup versus subsequent use of the application workflow.



This methodological choice is anchored in a **user-centered design** philosophy. Recognizing that many users lack programming experience, the application is built to be usable without writing code. Dropdowns, guided inputs, and automatic outputs are employed to support intuitive interaction and reduce cognitive load, making accessibility a central design concern.

The tool also follows a **modular structure**. Core components—including data upload, statistical summaries, visualization, and export—display discrete but interconnected modules. This not only streamlines the development process but also facilitates future extension and customization (e.g., new data formats or analytical outputs).

Finally, the approach prioritizes **reproducibility and transparency**. In many analytic environments, data scientists carry out workflows manually and in fragmented steps, making them prone to error and difficult to replicate. By centralizing key tasks within a single coded interface, the application promotes procedural uniformity and minimizes undocumented variation—enhancing both the reliability and standardization of outputs.

In summary, this methodological framework reflects a pragmatic response to real-world reporting needs: building a working prototype that centers users, favors clarity and modularity, and creates space for future development beyond the scope of the thesis.

Tools and Technologies

The decision to use `R` (R Core Team, 2023) as the primary language is both practical and strategic. With strong support for data cleaning, transformation, visualization, and statistical analysis, `R` aligns well with the core needs of the application. Its open-source nature and extensive package ecosystem also reflect the project's broader goals of transparency, accessibility, and collaboration.

To bring these capabilities into an interactive interface, the application is built using the **`Shiny`** framework (Chang et al., 2021), which enables the creation of web-based tools entirely within `R`. This integration allows data processing, user interaction, and result generation to take place in a single environment—ideal for rapid prototyping and minimizing technical overhead.

A key advantage of `Shiny` is its reactive programming model: user inputs automatically update related outputs like tables, plots, or dashboards. This behavior mirrors real-world reporting workflows, where small data changes often require time-consuming revisions. With reactivity, those updates happen instantly and seamlessly.

Supporting this architecture is a curated set of `R` packages that span the full reporting pipeline—from data import and transformation to visualization and formatted export. Packages were selected based on their documentation quality, compatibility with `Shiny's` reactivity, and relevance to applied practices.

Package(s)	**Purpose**
`shiny`, `shinydashboard`	Web application and dashboard development using the Shiny framework.
`shinyjs`	Enables JavaScript-based interactivity within the `Shiny` UI.
`DT`, `rhandsontable`	Interactive and editable data tables.
`readxl`, `haven`	Data import from *Excel*, *SPSS*, and *Stata* formats.
`dplyr`, `tidyr`, `rlang`	Data transformation and manipulation following the tidyverse paradigm.
`lubridate`	Parsing and manipulating date and time data.
`ggplot2`	Generation of statistical plots and data visualizations.
`officer`, `flextable`	Exporting results into *PowerPoint* and formatting tables professionally.

Table 1
Main R packages used in the development of the application

All development work was designed and programmed in `RStudio`, an integrated development environment (IDE) designed specifically for R. Despite being developed individually, the project is available on GitHub, laying the groundwork for future collaborative development, contribution, and version control. This open structure not only ensures transparency and traceability but also reinforces the tool's potential to evolve beyond this thesis into a broader, community-driven resource for applied analytical teams.

Workflow Logic and Adaptability

The application's overall logic is ruled by a reactive pipeline:

Data Upload → Cleaning → Analysis → Dashboard → Export

Although this workflow is not strictly linear in the codebase, reactivity ensures that user interactions in one part of the app immediately propagate across dependent components. For instance, a change in selected variables automatically updates summary tables, plots, and export content.

In addition, it is structured with modularity in mind. Each core task is handled through **self-contained functions**, making it easier to trace bugs, make improvements, and extend functionality later on without breaking what's already working.

One of the most important aspects is how the app **adapts dynamically** to the user's dataset. As soon as data is uploaded, the app checks its structure and only shows options that make sense—for example, numeric inputs for histograms or categorical ones for frequency tables. This kind of built-in logic helps reduce errors and supports users who may not be familiar with the technical requirements of each method.

Design and User Experience Considerations

Usability, accessibility, and clarity are central to the application's design, particularly for users with limited technical expertise. These are methodological choices as much as design ones, aimed at making data analysis more approachable and communication-ready in applied research contexts.

To support this aim, the interface adopts a **tab-based layout** that mirrors the reporting workflow: upload, clean, analyze, visualize, and export. This sequential structure helps guide users through the process, reducing confusion while supporting a logical and intuitive progression.

Building on this foundation, **interactive elements** like dropdowns and auto-filtered inputs respond dynamically to the structure of the uploaded dataset—displaying only relevant variables depending on the task at hand. To avoid overwhelming users, the interface also follows a **progressive disclosure approach**: advanced features such as filters or stratification appear only when needed, maintaining a clean layout while still enabling more complex use cases when appropriate.

A key component reinforcing this design logic is the **dashboard preview**, which displays saved outputs in a layout that mirrors the final *PowerPoint* export. This gives users immediate feedback on how results will appear in their reports, strengthening the link between analysis and presentation in a seamless way.

Ultimately, the design treats the interface not as a superficial layer, but as a core component of the tool's methodology. Its design enables users to produce reliable, well-structured outputs without technical expertise—making user experience a foundational element of the application from the start.

Iterative Testing and Refinement

Although formal usability testing is limited due to time constraints, the application benefits from ongoing informal feedback gathered throughout the development process. Academic peers, instructors, and colleagues with experience in data analytics workflows interacted with early versions of the tool and provided targeted suggestions regarding layout logic, button placement, labeling clarity, and the handling of user errors. This feedback directly informed several refinements, including clearer upload instructions, improved navigation flow, more intuitive labeling of UI components, and stronger export reliability.

APPLICATION OVERVIEW

To effectively support automated and reproducible reporting, the application uses a modular architecture that integrates backend logic and user-facing components through the `R Shiny` framework. This section details how the technical structure and interface work together to create a responsive, accessible tool for applied analytics.

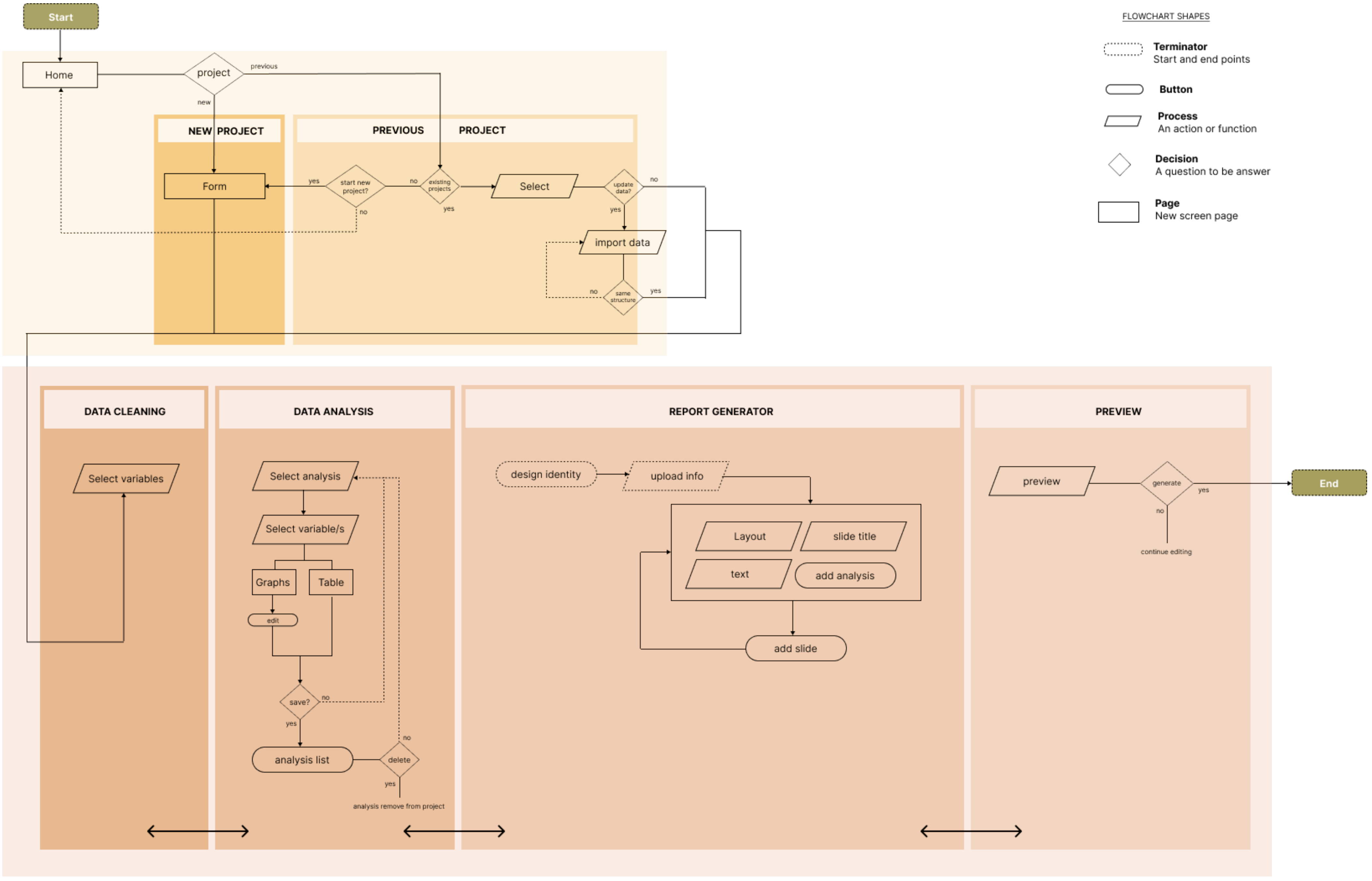
Technical foundations

The application adopts the standard structure of the `R Shiny` framework, which separates the user interface (UI) from the server-side logic. The `ui.R` file controls the layout—organizing tabs, input elements, and output containers—while the `server.R` file manages how data is processed, analyzed, and displayed. A third file, `global.R`, holds shared functions and libraries used throughout the app. These components work together through the `shinyApp(ui, server)` function, which brings the entire application to life.

To keep things organized and flexible, the app organizes its features into a series of **functional modules: data upload, cleaning and variable selection, statistical analysis, dashboard creation, and report export**. Each module handles a specific part of the reporting workflow, making the app easier to maintain and expand over time.

Figure 2

First internal architecture and module workflow sketch of the application.

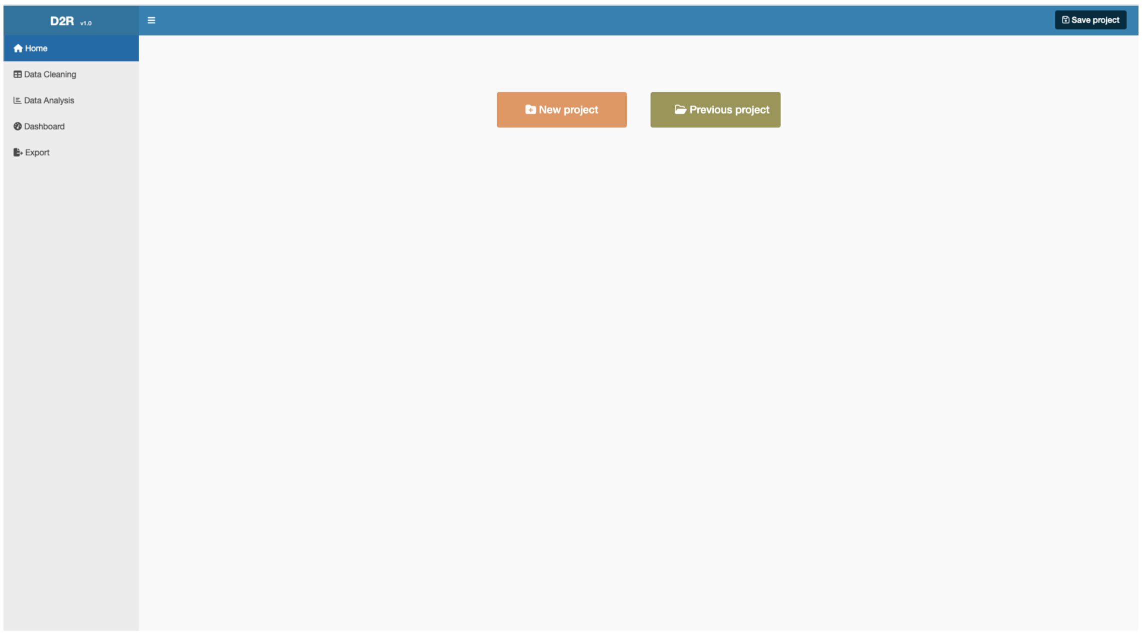


UI and Functional Features

The **beta-stage** application developed in this project is designed to support a modular and accessible workflow for automating the analysis and reporting of social science data. The user interface (UI) implemented reflects the core objective of enabling non-technical users to conduct structured data analysis and export results in a reproducible and standardized format.

Figure 3

General layout of the application interface



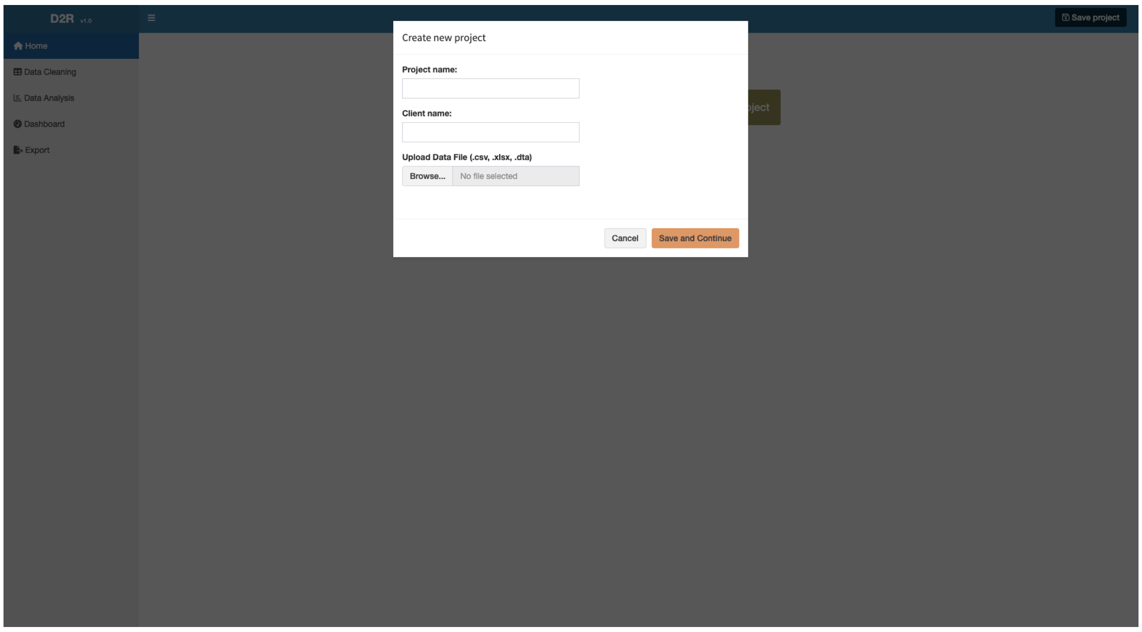
The interface is structured around five core interactive components:

Project Creation and Data Upload

The user begins by choosing either to create a new project or load a previous one. Upon creating a new project, the app prompts for basic metadata (e.g., project title, client) and allows the upload of datasets in `.csv`, `.xlsx`, and `.sav` formats. The upload process uses a custom *read_datafile*() function, with built-in validation and error handling. Once uploaded, the data is automatically cleaned and stored in a reactive container. The app then redirects the user to the next tab, ensuring a smooth workflow.

Figure 4

File upload interface with project creation modal and supported file formats.



Create new project

Project name:

Client name:

Upload Data File (.csv, .xlsx, .dta)

Browse... No file selected

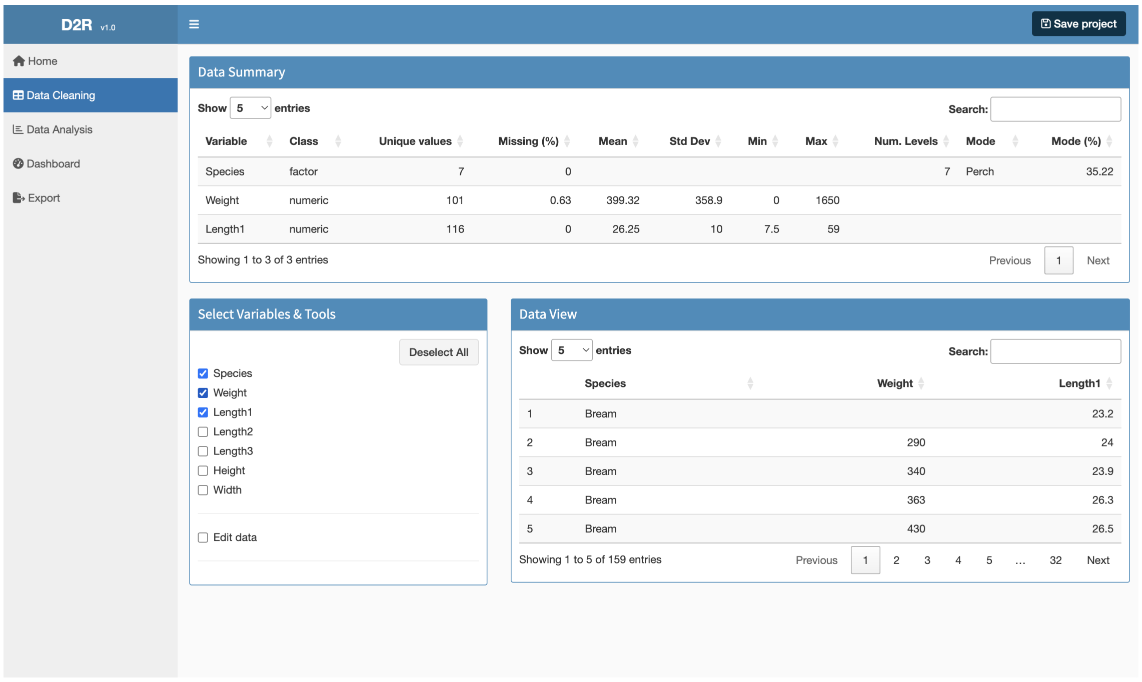
Cancel Save and Continue

Data Cleaning and Variable Selection

This section presents a dual-pane layout: selectors on the left, an editable or preview table on the right. Users can rename variables, recode values, or handle missing data. Changes are confirmed via dialog and tracked for reversibility. Summary statistics and variable metadata are automatically generated to support data understanding.

Figure 5

Data cleaning interface with editable dataset grid and variable selection controls.



Data Summary

Show 5 entries Search:

Variable	Class	Unique values	Missing (%)	Mean	Std Dev	Min	Max	Num. Levels	Mode	Mode (%)
Species	factor	7	0					7	Perch	35.22
Weight	numeric	101	0.63	399.32	358.9	0	1650			
Length1	numeric	116	0	26.25	10	7.5	59			

Showing 1 to 3 of 3 entries Previous 1 Next

Select Variables & Tools

Deselect All

- [x] Species
- [x] Weight
- [x] Length1
- [ ] Length2
- [ ] Length3
- [ ] Height
- [ ] Width

- [ ] Edit data

Data View

Show 5 entries Search:

	Species	Weight	Length1
1	Bream		23.2
2	Bream	290	24
3	Bream	340	23.9
4	Bream	363	26.3
5	Bream	430	26.5

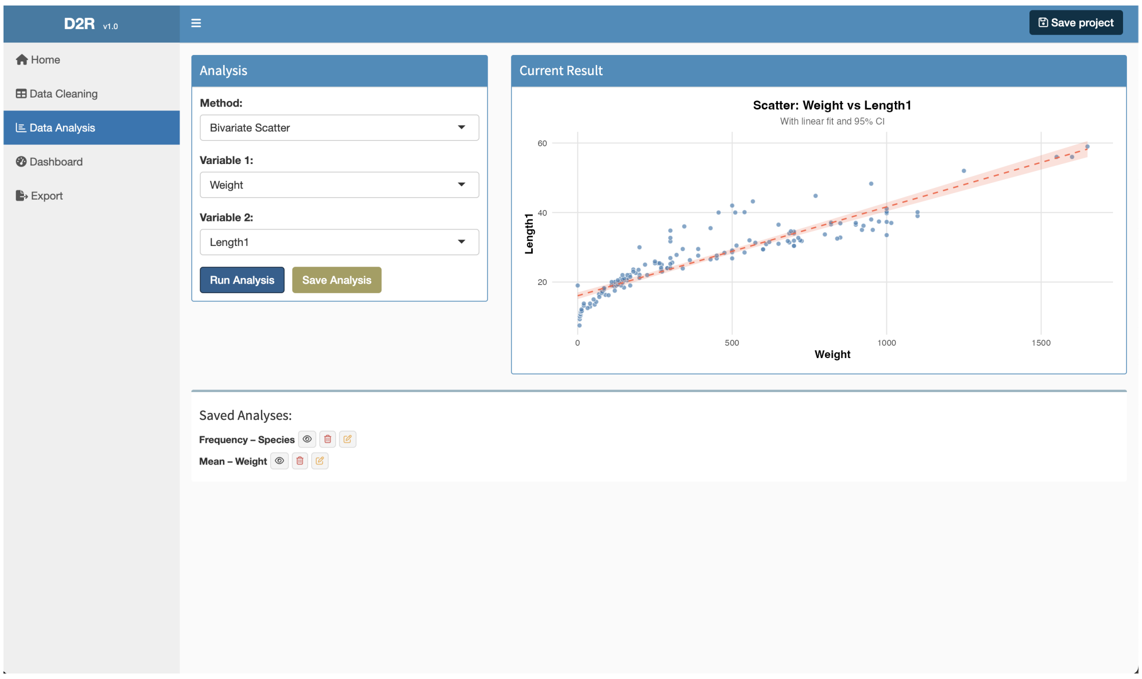
Showing 1 to 5 of 159 entries Previous 1 2 3 4 5 ... 32 Next

Data Analysis and Visualization

In this tab, users select from drop-down menus to run analyses such as frequencies, means, or bivariate plots. The UI adapts based on analysis type. Each result can be labeled and saved for later use. Then, the saved outputs feed into the dashboard and export modules.

Figure 6

Analysis interface with variable inputs and resulting plots or summary tables.



Method: Bivariate Scatter

Variable 1: Weight

Variable 2: Length1

Saved Analyses:

Frequency – Species

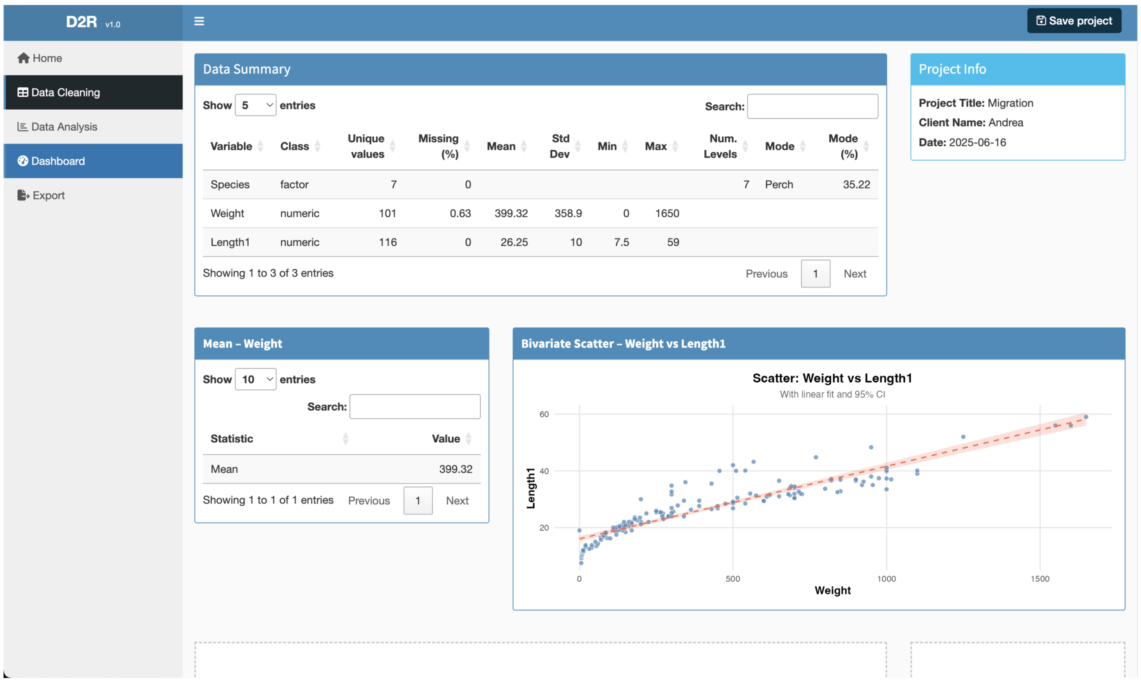
Mean – Weight

Dashboard Composition

Saved results are placed into panels that simulate a slide layout. Users can assign a table or plot to each panel and preview the final structure.

Figure 7

Dashboard composition view with layout panel and output assignments.



Data Summary

Show 5 entries Search:

Variable	Class	Unique values	Missing (%)	Mean	Std Dev	Min	Max	Num. Levels	Mode	Mode (%)
Species	factor	7	0					7	Perch	35.22
Weight	numeric	101	0.63	399.32	358.9	0	1650			
Length1	numeric	116	0	26.25	10	7.5	59			

Showing 1 to 3 of 3 entries Previous 1 Next

Project Info

Project Title: Migration
Client Name: Andrea
Date: 2025-06-16

Mean – Weight

Show 10 entries Search:

Statistic	Value
Mean	399.32

Showing 1 to 1 of 1 entries Previous 1 Next

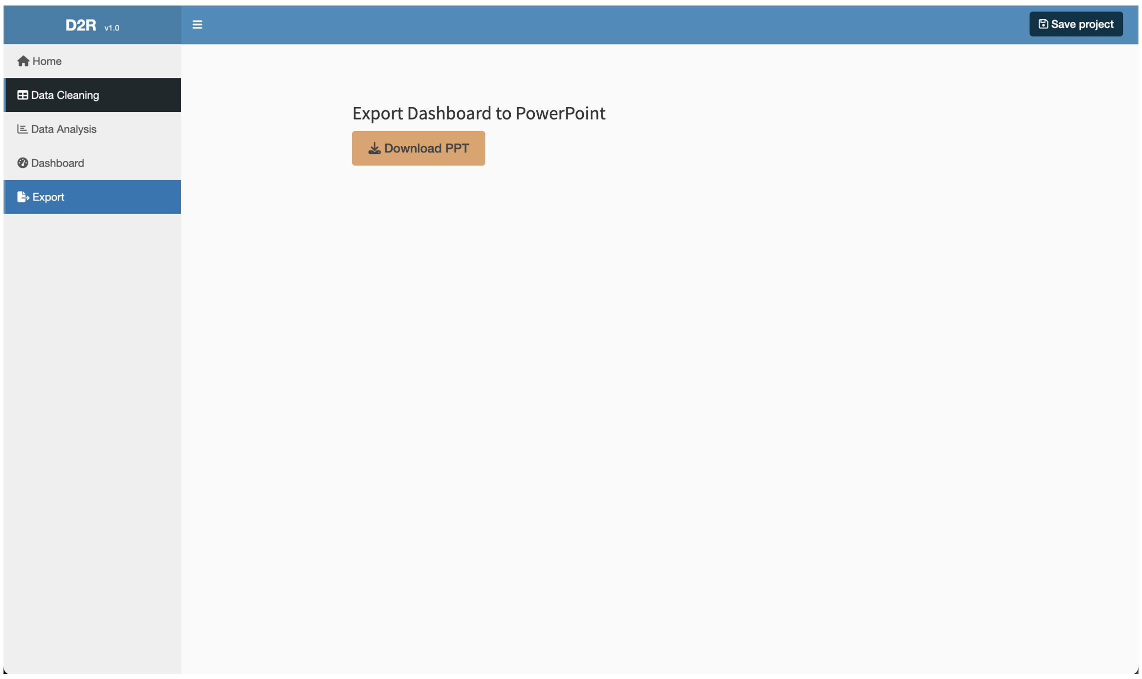
Bivariate Scatter – Weight vs Length1

Report Export

In the final step, users export the dashboard to *PowerPoint*. Slides are automatically formatted to include titles, captions, and plots or tables. This automates a process that typically involves hours of manual formatting in applied analytics workflows.

Figure 8

Export tab interface with export button.



CHALLENGES

The development of this application posed a series of practical and conceptual challenges, closely aligned with its broader goals: building a reproducible, accessible, and automated reporting tool for social science workflows. Balancing technical feasibility with user-centered design required continuous iteration and problem-solving. This section outlines the key difficulties encountered, the solutions implemented, and reflects briefly on how well the core objectives are being met.

Designing for Non-Technical Users

Creating an interface that guides users through complex analytical workflows—without requiring statistical or programming expertise—was a central challenge. The solution involved breaking the process into clear, sequential tabs and using simple UI components like dropdowns, modals, and pre-filtered inputs. Features such as inline prompts, editable tables, and real-time previews helped reduce cognitive load while preserving control. Overall, this structure successfully made the application approachable to non-technical users while supporting analytical rigor.

Managing Reactivity and Session State

Enabling users to save and reload full projects—including datasets, settings, and dashboards—was essential for reproducibility. This process relies on serializing user sessions into *.rds* files, which store both content and structure. When reloaded, the application restores the state and recalculates outputs to ensure consistency. This functionality met its intended goal, offering a practical way to preserve and revisit analytical work without manual reconstruction.

Saving and Reloading Complex Projects

One of the more complex features of the app is its ability to save and reload entire projects, including the dataset, selected variables, user-generated analyses, and dashboard layout. This functionality was critical for enabling reproducibility, but presented both conceptual and technical challenges.

To implement this, the solution involved serializing the project state into *.rds* files using custom save and load functions. Each project file includes metadata such as the project title, client name, selected variables, saved analyses, and dashboard layout. Upon loading, the application restores these elements and re-renders relevant UI components. To ensure consistency, the system recalculates all analyses in real time using the current data, reducing the risk of outdated or inconsistent outputs.

Exporting to Structured Output Formats

Exporting results into a structured `PowerPoint` presentation introduced notable design and technical challenges. A key difficulty was converting interactive, `HTML`-based elements—such as tables and plots—into static formats suitable for slide output. To address this, the application uses the `officer` and `flextable` packages to render plots as high-resolution *PNG* images and format tables with consistent styles. Although the layout limits flexibility and dimension adjustments require iterative refinement, the export feature ultimately proved effective. It delivered on one of the tool's core objectives: reducing the time spent assembling reports manually while ensuring consistency and readability across outputs.

Limited User Testing and Iterative Refinement

Due to time constraints, the project limits user testing to informal peer feedback. However, this feedback informed meaningful improvements—particularly in layout, labeling, and error-handling. The emphasis on modular design allowed for agile updates and validated many usability decisions even without formal testing protocols.

Many of the project's key goals—automation, accessibility, reproducibility, and structured reporting—were successfully demonstrated in this prototype. While technical challenges, particularly around reactivity and state management, required substantial effort to address, the final application provides a strong foundation for future development and institutional use. The process underscored the importance of designing for real users, not ideal ones, and offered valuable lessons for bridging technical tools with practical analytical workflows.

CONCLUSION

This thesis began with a simple but persistent question: why, in an era of increasingly sophisticated analytical tools, does the task of producing analytical reports in applied social science remain so repetitive, fragmented, and prone to error? This challenge—encountered firsthand through years of professional experience—reflects a broader structural gap between the way data is analyzed and the way results are ultimately communicated.

The development of the `R Shiny` application presented here responds directly to that challenge. By offering a structured yet flexible interface for automating the analysis and export of outputs, the application addresses core inefficiencies in current workflows while promoting transparency and standardization. Importantly, it does so without requiring users to write a single line of code—thus opening the door to wider adoption among non-technical practitioners.

The solution is not without limitations. While the prototype demonstrates the feasibility and value of such a tool, further testing, customization, and refinement will be necessary for full-scale deployment. Usability testing, performance optimization, and expanded functionality (e.g., additional export formats or analytical options) remain opportunities for future work. Still, the application succeeds in proving that reproducibility and automation need not come at the cost of accessibility or design.

Beyond the technical contributions, this application also engages a broader methodological conversation around reproducibility and usability. Literature on open science, workflow transparency, and team-based research emphasizes that reproducibility is not just about making code public—it is about designing tools, habits, and infrastructures that are adoptable across diverse roles and contexts. By embedding good practices—such as version control, structured inputs, and standardized outputs—into a user-friendly interface, this tool translates abstract principles into practical routines that can scale across teams and institutions.

In a landscape increasingly shaped by interdisciplinary teams, policy deadlines, and the demand for clear, communicable insights, tools like this one serve a dual role: they reduce the friction of doing analysis, and they raise the floor for what transparent, efficient, and replicable reporting can look like. The application presented here is an early step in that direction—practical, imperfect, and open to iteration, but grounded in the real needs and guided by a commitment to clarity, usability, and reproducibility. It is an invitation to continue building tools that not only process data but also support the people who make sense of it.

BIBLIOGRAPHY

Beaton, D. E., Bombardier, C., Tugwell, P., & Guillemin, F. (2000). Guidelines for the process of cross-cultural adaptation of self-report measures. *Spine*, *25*(24), 3186–3191. https://doi.org/10.1097/00007632-200012150-00014

Chang, W., Cheng, J., Allaire, J., Xie, Y., & McPherson, J. (2021). Shiny: Web application framework for r [R package version 1.x].

Fecher, B., & Friesike, S. (2014). Open science: One term, five schools of thought. *Opening Science*, 17–47. https://doi.org/10.1007/978-3-319-00026-8_2

Goodman, S. N., Fanelli, D., & Ioannidis, J. P. (2016). What does research reproducibility mean? *Science Translational Medicine*, *8*(341), 341ps12. https://doi.org/10.1126/scitranslmed.aaf5027

Kassambara, A. (2018). Ggpubr: 'ggplot2' based publication ready plots [R package version 0.1.8].

Peng, R. D. (2011). Reproducible research in computational science. *Science*, *334*(6060), 1226–1227. https://doi.org/10.1126/science.1213847

Rule, A., Birmingham, A., Hinsen, K., Perez, F., & Barnes, P. J. (2019). Ten simple rules for writing and sharing computational analyses in jupyter notebooks. *PLoS Computational Biology*, *15*(7), e1007007. https://doi.org/10.1371/journal.pcbi.1007007

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLOS Computational Biology*, *9*(10), e1003285. https://doi.org/10.1371/journal.pcbi.1003285

Stodden, V., Guo, P., & Ma, Z. (2014). Toward reproducible computational research: An empirical analysis of data and code policy adoption by journals. *PLOS ONE*, *8*(6), e67111. https://doi.org/10.1371/journal.pone.0067111

Wilson, G., Aruliah, A., Brown, C. T., Chue Hong, N. P., Davis, M., Guy, R. T., Haddock, S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., et al. (2014). Best practices for scientific computing. *PLOS Biology*, *12*(1), e1001745. https://doi.org/10.1371/journal.pbio.1001745

Xie, Y. (2015). *Dynamic documents with r and knitr* (2nd ed.). Chapman; Hall/CRC. https://doi.org/10.1201/9781315382487

APPENDIX

Github repository:View repository

Application documentation: View application documentation

Interactive initial prototype: View prototype on Figma uc3m | Universidad Carlos III de Madrid

DECLARACIÓN DE USO DE IA GENERATIVA EN EL TRABAJO DE FIN DE MÁSTER

He usado IA Generativa en este trabajo

Marca lo que corresponda:

[✓ **SI**] **NO**

Si has marcado SI, completa las siguientes 3 partes de este documento:

Parte 1: reflexión sobre comportamiento ético y responsable

Ten presente que el uso de IA Generativa conlleva unos riesgos y puede generar una serie de consecuencias que afecten a la integridad moral de tu actuación con ella. Por eso, te pedimos que contestes con honestidad a las siguientes preguntas (*Marca lo que corresponda*):

Pregunta		
1. En mi interacción con herramientas de IA Generativa he remitido **datos de carácter sensible** con la debida autorización de los interesados.		
SÍ, he usado estos datos con autorización	NO, he usado estos datos sin autorización	[✓ NO, no he usado datos de carácter sensible]
2. En mi interacción con herramientas de IA Generativa he remitido **materiales protegidos por derechos de autor** con la debida autorización de los interesados.		
SÍ, he usado estos materiales con autorización	NO, he usado estos materiales sin autorización	[✓ NO, no he usado materiales protegidos]
3. En mi interacción con herramientas de IA Generativa he remitido **datos de carácter personal** con la debida autorización de los interesados.		
SÍ, he usado estos datos con autorización	NO, he usado estos datos sin autorización	[✓ NO, no he usado datos de carácter personal]

4. Mi utilización de la herramienta de IA Generativa ha **respetado sus términos de uso**, así como los principios éticos esenciales, no orientándola de manera maliciosa a obtener un resultado inapropiado para el trabajo presentado, es decir, que produzca una impresión o conocimiento contrario a la realidad de los resultados obtenidos, que suplante mi propio trabajo o que pueda resultar en un perjuicio para las personas.	
[✓ SI]	NO

Si **NO** has contado con la autorización de los interesados en alguna de las preguntas 1, 2 ó 3, explica brevemente el motivo *(por ejemplo, "los materiales estaban protegidos pero permitían su uso para este fin" o "los términos de uso, que se pueden encontrar en esta dirección (…), impiden el uso que he hecho, pero era imprescindible dada la naturaleza del trabajo".*

Parte 2: declaración de uso técnico

Utiliza el siguiente modelo de declaración tantas veces como sea necesario, a fin de reflejar todos los tipos de iteración que has tenido con herramientas de IA Generativa. Incluye un ejemplo por cada tipo de uso realizado donde se indique: *[Añade un ejemplo].*

Declaro haber hecho uso del sistema de IA Generativa (ChatGPT- 4.1 mini) **para:**

Documentación y redacción:

- *Soporte a la reflexión en relación con el desarrollo del trabajo: proceso iterativo de análisis de alternativas y enfoques utilizando la IA*

During the early stages of the thesis, I asked ChatGPT: *"How does automated reporting relate to the concept of reproducibility in open science?"*. The response helped me frame the introduction and literature review, connecting the technical goal of the app with ongoing academic discussions about transparency and reproducible research.

- *Revisión o reescritura de párrafos redactados previamente*

When finalizing the abstract, I asked ChatGPT: *"Can you tell me some keywords that best reflect the core themes of my thesis on automated reporting in R Shiny?"*. Based on its suggestions, I selected terms like *Reproducible Analytics*, *Automated Reporting*, *User-Centered Design*, and *Analytic Communication*, which were later included as the official keywords of the thesis to improve its discoverability and accurately reflect its scope.

- *Búsqueda de información o respuesta a preguntas concretas*

I asked ChatGPT: *"What R packages are commonly used to export outputs to PowerPoint from a Shiny app?"*. Based on the response, I incorporated the `officer` and `flextable` packages into the export module of the application, as documented in the Application Overview chapter of the thesis.

- *Búsqueda de bibliografía*
- *Resumen de bibliografía consultada*
- *Traducción de texto consultados*

Desarrollar contenido específico

Se ha hecho uso de IA Generativa como herramienta de soporte para el desarrollo del contenido específico del TFM, incluyendo:

- *Asistencia en el desarrollo de líneas de código (programación)*

I used ChatGPT to troubleshoot R code related to reactivity in Shiny. Specifically, I asked: *"Why is my reactive plot not updating when I change the input variable?"*. ChatGPT explained the concept of reactive dependencies and suggested restructuring the observer logic, which helped me resolve issues in the `server.R` component of the app.

- *Generación de esquemas, imágenes, audios o vídeos*

Although no generative media like audio or video was used, I asked ChatGPT for help creating a conceptual diagram for the application's internal architecture. Based on its suggestions, I sketched the modular workflow (shown in Figure 2 of the thesis), which helped clarify the system logic during the design phase.

- *Procesos de optimización*

During development, I consulted ChatGPT on improving the performance of the data cleaning module. I asked: *"How can I avoid unnecessary re-rendering in a Shiny app with multiple dependent inputs?"*. It suggested using `isolate()` and more efficient reactive expressions, which led to smoother interactivity and reduced processing time.

- *Tratamiento de datos: recogida, análisis, cruce de datos…*

While building the analysis module, I used ChatGPT to clarify best practices for handling user-uploaded datasets. I asked: *"What's a safe way to handle missing values and variable types dynamically in Shiny?"*. Based on the guidance received, I implemented automated checks that adapt input options based on data types, which streamlined data transformation and reduced error rates.

- *Inspiración de ideas en el proceso creativo*

While designing the user interface, I prompted ChatGPT with: *"What UX design principles should I follow to make an analytics tool accessible for non-technical users?"*. The feedback influenced key decisions like using a tab-based workflow, progressive disclosure of options, and real-time preview components.

- *Otros usos vinculados a la generación de puntos concretos del desarrollo específico del trabajo*

Si crees necesario incluirlo, añade, en cada uno de los casos:

empleando el *prompt* …
(escribe la petición realizada a la IAG)
teniendo como interacción …
(describe qué interacción realizaste con la IAG tras la respuesta del prompt)

Ejemplo: ***Declaro** haber hecho uso del sistema de IA Generativa **Chat GPT 3.5** para **buscar información** empleando **el prompt: "Dime un ejemplo concreto que ilustre la aplicación de la propuesta urbanística de la Ciudad de los 15 minutos en España"** teniendo como interacción **la proposición del ejemplo concreto del distrito de Chamartín que se ha reflejado en la memoria**.*

Parte 3: reflexión sobre utilidad

Por favor, aporta una valoración personal (formato libre) sobre las fortalezas y debilidades que has identificado en el uso de herramientas de IA Generativa en el desarrollo de tu trabajo. Menciona si te ha servido en el proceso de aprendizaje, o en el desarrollo o en la extracción de conclusiones de tu trabajo.

The use of generative AI, particularly ChatGPT, played a valuable but measured role in the development of my thesis. Its main strength was in offering on-demand support during moments of uncertainty—whether technical (e.g., debugging code or structuring reactive logic in Shiny) or conceptual (e.g., articulating the link between reproducibility and usability). In this sense, the tool served as a brainstorming partner, accelerating certain phases of development and offering reassurance when navigating unfamiliar challenges.

Another important strength was its flexibility in adjusting to the evolving needs of the project: from initial idea exploration, to refining arguments, to polishing academic writing. The ability to iterate rapidly and receive feedback in natural language reduced the friction typically associated with technical documentation or problem-solving.

However, its limitations were also clear. While ChatGPT provided useful code snippets and design advice, the suggestions often required additional validation or adaptation to fit the specific structure and logic of my application. Moreover, when dealing with edge cases in Shiny or advanced package interactions, the AI's responses were sometimes too generic or outdated. This underlines the importance of maintaining a critical perspective and combining AI assistance with rigorous testing and human judgment.

Overall, I found it particularly useful in the learning process—it helped me solidify my understanding of both programming and methodological decisions by forcing me to articulate precise questions and evaluate multiple solutions. Although it did not replace human insight or independent work, it served as a valuable complement, making the research process more fluid, interactive, and at times, less solitary.